Dedicatoria

Este proyecto está dedicado a nuestras familias y amigos que nos apoyaron incondicionalmente desde el comienzo de nuestra carrera, como así también, durante nuestra estadía en Francia que nos permitió poder llevarlo a cabo.

Agradecimientos

Queremos agradecer, para comenzar, a todas las personas que contribuyeron al correcto desarrollo de nuestro proyecto y nuestro aprendizaje.

Principalmente, agradecemos a nuestros tutores de tesis en la universidad francesa, por sus consejos y por habernos guiado a lo largo de todo el trabajo.

A continuación, nos gustaría agradecer a todos aquellos que dedicaron parte su tiempo para ayudarnos a afrontar diversas situaciones que fueron surgiendo durante el proyecto.

También, agradecemos al Instituto Universitario Aeronáutico por todos los conocimientos que nos brindó durante todos los años de cursado de la carrera y por permitirnos el acceso a la beca ARFITEC para crear un lazo entre nuestra universidad y la Universidad de Valenciennes.

Trabajo Final de Grado

Radio Definida por Software sobre BladeRF

Paulo DELRIVO
Francisco MALEM

IEMN-IUA

Ingeniería en Telecomunicaciones

Contenido

Índice de Imágenes

1- Prólogo

Desde septiembre de 2015 a enero de 2016 hemos realizado nuestro proyecto de fin de estudios en el laboratorio IEMN-DOAE que pertenece a la universidad de Valenciennes. El objetivo del proyecto fue el desarrollo de un demostrador pedagógico sobre Radio Definida por Software.

El proyecto surge gracias a que el Laboratorio adquirió recientemente las placas BladeRF (Nuand), que permiten trabajar sobre una plataforma flexible con calidad profesional, con un costo razonable. Las placas pueden trabajar tanto en Linux, Windows o Mac.

Para desarrollar el trabajo, el software utilizado primariamente fue Matlab, permitiendo implementar las diferentes etapas de una cadena de comunicaciones haciendo uso de todos los parámetros correspondientes.

El trabajo fue desarrollado de manera conjunta en su primera etapa y aquellas en las que se ha requerido mayor dedicación. Luego se realizó una división de tareas con el objetivo de avanzar más rápidamente.

2- Introducción

Tradicionalmente todos los equipos receptores y emisores de radiocomunicaciones estaban constituidos por multitud de componentes electrónicos, los cuales formaban circuitos sintonizadores, etapas de frecuencia intermedia, detectores, amplificadores de baja frecuencia, etc., es decir, estaban constituidos por "Hardware". Posteriormente, en los años 1980 y 1990 se introdujeron microprocesadores en estos equipos para el control de funciones internas (controles desde teclados y pulsadores) y para añadir nuevas prestaciones (relojes, pantallas informativas, programadores), y también se introdujo la posibilidad de controlar los equipos de radio desde un ordenador, añadiendo al equipo de radio puertos de comunicación o interfaces para la conexión al ordenador. En estos casos, y usando el software adecuado, es posible controlar desde el ordenador numerosas funciones del equipo de radio, igual o mejor que desde los controles del propio equipo. También en la década de 1990 comenzó la introducción en los modernos equipos de radio de los chips DSP o "Procesadores Digitales de Señal", los cuales permiten mediante técnicas digitales realizar filtros de paso de banda y de supresión de ruidos, entre otras posibilidades, muy eficaces, mejor que los realizados tradicionalmente con circuitos analógicos.

En cualquier caso, siempre se trata de equipos de radio realizados enteramente con componentes electrónicos, o sea, en términos informáticos se definirían como "radios hardware". Pero desde principios de la década del 2000 un grupo de radioaficionados están investigando y desarrollando un nuevo concepto de equipos de radiocomunicaciones, los equipos de radio desarrollados por programa o "radios Software", en siglas SDR (Software Defined Radio).

Básicamente todos los usos implican tomar la señal del receptor y procesar la codificación contenida en esta para reconstruir el mensaje digital originalmente codificado en este por el proceso, a modo inverso, en el otro extremo de la comunicación.

3- Presentación del proyecto

3.1- Objetivo Principal

El laboratorio en el cual desarrollamos nuestra tesis (IEMN DOAE) ha adquirido recientemente placas de radio definida por software BladeRF (Nuand). El objetivo de nuestra práctica en la universidad francesa fue de desarrollar un demostrador con las diferentes posibilidades ofrecidas por la placa.

Con este hardware es posible de llevar a cabo los siguientes procesos: análisis espectral, transmisión mono portadora (BPSK, QPSK, 16QAM, FSK), transmisión OFDM, transmisión MIMO 2x2.

Durante la práctica, el interés principal ha sido la programación de la cadena de emisión y recepción a través del software Matlab, utilizando el tipo de modulación QPSK, parametrizando cada etapa del sistema e identificando todas las variables intervinientes a modo de comprender el funcionamiento de dicha tecnología, integrando los conceptos sobre modulación adquiridos en la carrera de grado.

3.2- Objetivos específicos

- Poner en práctica los diferentes tipos de modulación estudiados durante la carrera de ingeniería;
- Aprender a programar en Matlab;
- Conocer los parámetros de funcionamiento de la placa BladeRF;
- Parametrizar cada etapa de una cadena de comunicaciones;
- Analizar los diferentes efectos que tiene el canal sobre la señal transmitida.

4- IEMN: Un gran centro de Investigación Europeo

4.1- Presentación del Instituto

EL IEMN es una Unidad Mixta de investigación que asocia el CNRS (Centre National de la Recherche Scientifique) y tres establecimientos de Enseñanza Superior públicas y privadas. Sus equipos de diseño, fabricación y caracterización de dispositivos se sitúan en el mejor nivel europeo.

La actividad científica de 20 grupos de investigación del instituto es estructurada en cinco ejes [Figura 1]:

1. Los materiales y las nanoestructuras,
2. Las microtecnologías y los microsistemas,
3. Micro, nano y optoelectrónica,
4. **Los circuitos y los sistemas de telecomunicaciones**,
5. La acústica.

Cada eje consta de varios grupos. En ciertos casos, un grupo está formado por diferentes equipos.

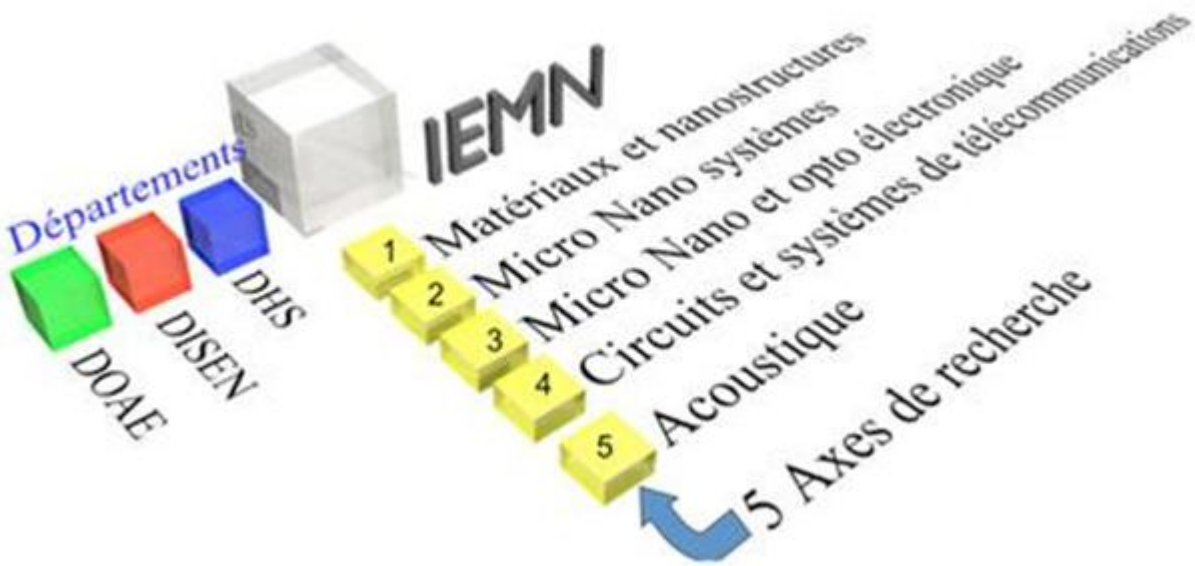

Ilustración 1-Organización del IEMN

4. SISTEMAS DE COMUNICACIÓN Y APLICACIÓN DE LAS MICROONDAS

Las principales actividades del IEMN en este eje son reagrupadas en 4 acciones unificadoras:

Acción 1

Objetos móviles comunicantes, redes de captadores y redes de proximidad de alta velocidad

Esta acción interdisciplinaria se desarrolla en colaboración estrecha con LIFL (Laboratorio de Informática de Lille) en el marco de la Federación de Investigación IRCICA (FR CNRS 3024). EL IEMN está a cargo de las capas física y MAC mientras que el LIFL desarrolla Middleware (lógica de intercambio de información entre aplicaciones "interlogical") de comunicación y los algoritmos de envío. Los trabajos esencialmente se refieren a las redes de proximidad que utilizan el canal de 60 GHz y enlaces mixtos fibra-radio para la cobertura global intraedificio.

Este estudio incluye:

El sondeo y la modelización de canal de 60 GHz así como la optimización de la división de los recursos.

El desarrollo de módulos radio en miniatura y de bajo consumo para nudos de redes de captadores que utilizan una técnica de banda ultra ancha impulsional y un sector BiCMOS avanzado (ST-M).

El desarrollo de redes locales (WLAN / WPAN) de muy de alta velocidad (> 7Gb/) utilizando esquemas de modulación vectorial complejos y técnicas de multiplexado frecuencial.

El desarrollo de tecnologías de integración heterogénea para la realización de Systems In Package (SIP) que integra antenas ágiles a base de MEMS.

La recuperación y la transformación de energía.

Acción 2

Telecomunicaciones y radiolocalización para los transportes

Esta acción es principalmente dedicada a los transportes terrestres en entorno automóvil y ferroviario. Contiene:

Trabajos sobre los fenómenos complejos de acoplamiento entre haces de cables y conectores automóviles que reposan a la vez en estudios electromagnéticos teóricos.

La caracterización del entorno electromagnético a bordo de trenes y el estudio de las perturbaciones electromagnéticas inducidas por una subestación ferroviaria.

El estudio de las comunicaciones intra-vehículo sobre la red eléctrica (Power Line Communication) con vistas a caracterizar y modelizar el canal de propagación para desarrollar la codificación más adaptada para librarse de parásitos como el ruido impulsivo.

El despliegue de comunicación MIMO en túneles basados en el concepto de diversidad modal para mejorar la capacidad del canal y el desarrollo de un nuevo banco de prueba para simular un entorno multitrayectos y someter a un test de sistemas MIMO.

El diseño y la realización de sistemas de radio localización y de medida de distancia que utiliza técnicas de banda ultra ancha o escalonamiento espectral (AOA, TOA, TDOA).

Acción 3

Comunicaciones numéricas RF y diseño software de radio y radar

Esta acción concierne:

El desarrollo de técnicas numéricas específicas de tratamiento de señales para mejorar las realizaciones de los enlaces multiportadoras de tipo DMT y OFDM.

El diseño y la realización de digitalizadores de banda ultra ancha (a DC - 20 GHz) alta resolución (> 8 bits efectivos) y gran dinámica (> 60 dB) para el tratamiento digital de señales de radar complejas.

El desarrollo de interfaz RF para radio software en los terminales móviles que se apoyan en tecnología CMOS, el filtrado BAW y la integración SIP / REJA (colaboración ST-M).

El desarrollo de transmisiones fibras-radios multiestándar que utiliza componentes acústico-ópticos para el multiplexado en longitudes de ondas, la propagación de señal ULB y el acceso múltiple por CDMA óptico.

Acción 4

Dispositivos específicos y aplicaciones de las microondas

Estos estudios incluyen:

El desarrollo de sistemas de estampería pasiva en gama mm. y sub-mm.

El desarrollo de dispositivos de caracterización no destructiva en gama microonda y milimétrico, en particular microscopios microondas de campo próximo a barrido y radiómetros para aplicaciones biológicas y el control de procesos industriales.[1]

4.2- El Departamento OAE del IEMN

El Departamento OAE del IEMN deriva del Laboratorio "Ultrasonidos e Hipersonidos" creado por el profesor Michel Moriamez en 1963 en la Universidad de Lille 1 y trasladado en gran parte al Centro Universitario de Valenciennes (creado en 1966) en el trascurso de los años 1969 y 1970.

Este Centro Universitario se hizo Universidad en 1971 y el Laboratorio tomó luego el título "Opto-Acústico-Electrónico" y ha sido asociado con CNRS como asi también con URA 832 en 1976. El laboratorio se hizo por fin parte integral (Departamento) del IEMN (El Instituto de Electrónica, Microelectrónica y de Nanotecnología, UMR 8520) creado en enero de 1992. En efecto, el IEMN ha sido creado por el Centro Nacional de Investigación Científica (CNRS), la Universidad de Ciencias y tecnología de Lille (USTL), la Universidad de Valenciennes y de Hainaut Cambrésis (UVHC) y el Instituto Superior de Electrónica y del Numérico (ISEN).

Su objetivo es reagrupar en una estructura única lo esencial de la investigación regional en un dominio que va de la física a las aplicaciones de la Electrónica y de crear así en Norte-Pas-de-Calais un Instituto de talla europea.

El Departamento de Opto-Acústica y Electrónica (DOAE) del IEMN está situado sobre el campus universitario de Valenciennes. El DOAE es un componente de la UVHC.

Sobre el plan científico, el DOAE es estructurado en 3 grupos y un equipo de investigación [Figura 2].

Ilustración 2-Estructura del DOAE

[2]

5- Memoria descriptiva

5.1- Diagrama FAST

Para asegurar una mejor organización de trabajo, hemos creado un diagrama FAST (Function analysis system technique) [Figura 3]. Este es un tipo de diagrama que presenta los datos esenciales que permiten tener conocimiento de un proyecto como así también de poder mejorar la solución propuesta. El mismo se lee de izquierda a derecha, con una lógica de "cómo" y de derecha a izquierda con una lógica de "por qué".

A continuación se puede observar el diagrama FAST que ha sido realizado para nuestro proyecto. Este muestra los objetivos a cumplir, y los métodos y medios utilizados.

Cada parte será completamente detallada luego del diagrama para una mejor comprensión del proyecto.

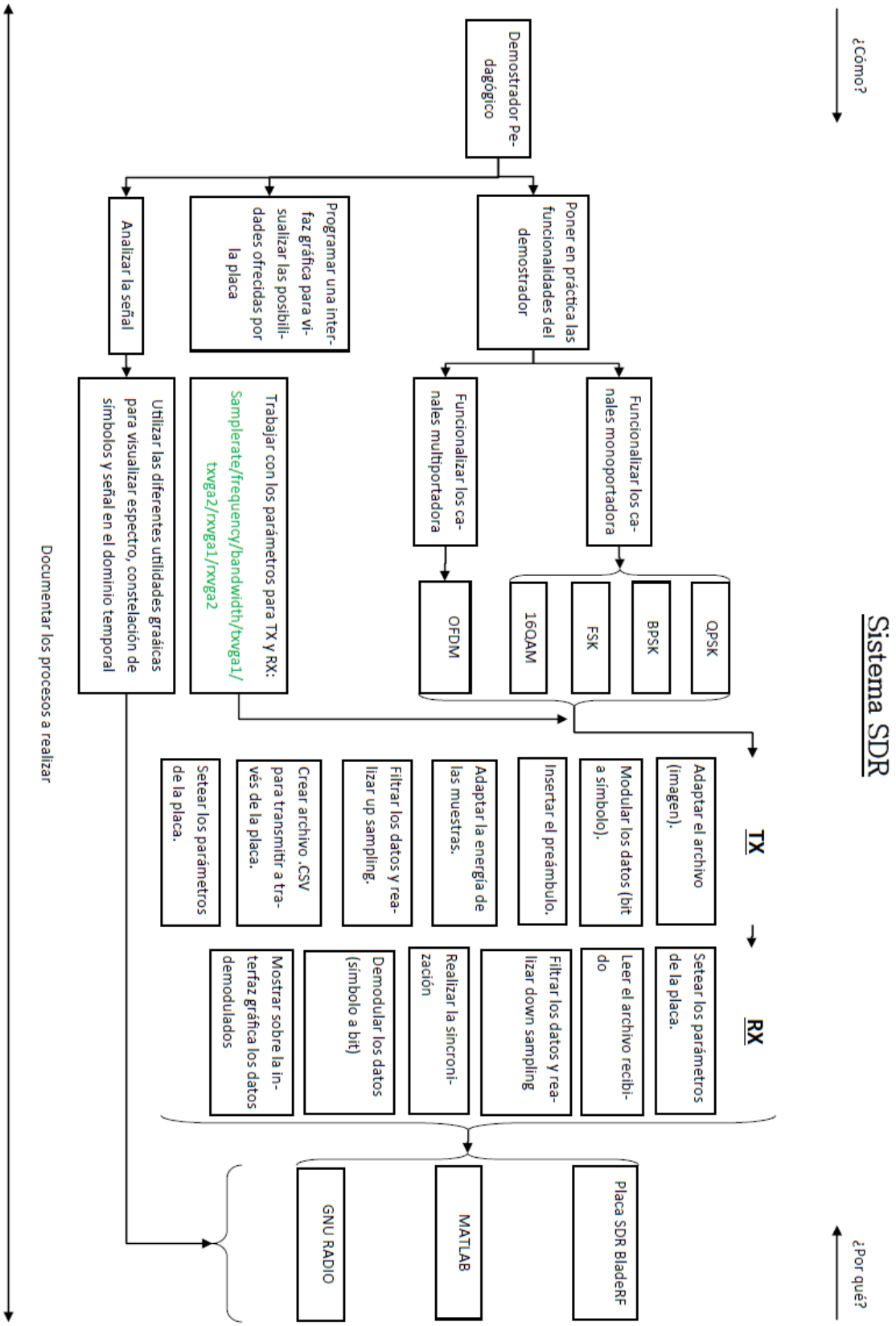

Ilustración 3- Diagrama FAST

5.2- SDR – Radio definida por Software

Radio definida por software, en inglés Software Radio o Software Defined Radio (SDR), es un receptor y emisor de radio realizado principalmente por software y en menor medida por parte material [Figura 4].

En la recepción, la parte material consiste en la digitalización directa, por un convertidor analógico-digital (CAN), de las señales en banda de alta frecuencia que se van a recibir, o su conversión en una banda de frecuencia Intermedia (FI) antes de la digitalización.

Los tratamientos que siguen pueden ser realizados mediante un software: filtrado, decimación, demodulación, desciframiento. Estos tratamientos son realizados con la ayuda de un microprocesador dedicado al tratamiento de la señal, del componente electrónico programable, o directamente sobre el procesador de una computadora tradicional. Esto le confiere una universalidad y una gran adaptabilidad al emisor / receptor. En efecto, basta con cambiar o con adaptar el software para funcionar con un sistema de radio diferente.

En un sistema de radio clásico, la emisión / recepción es asegurada por componentes materiales específicos y adaptados a los sistemas a los cuales es destinado. Por lo tanto, muchas veces no es posible utilizar otros sistemas sin cambiar el material y la integridad del receptor.

El campo de radio definida por software está en desarrollo constante. [3]

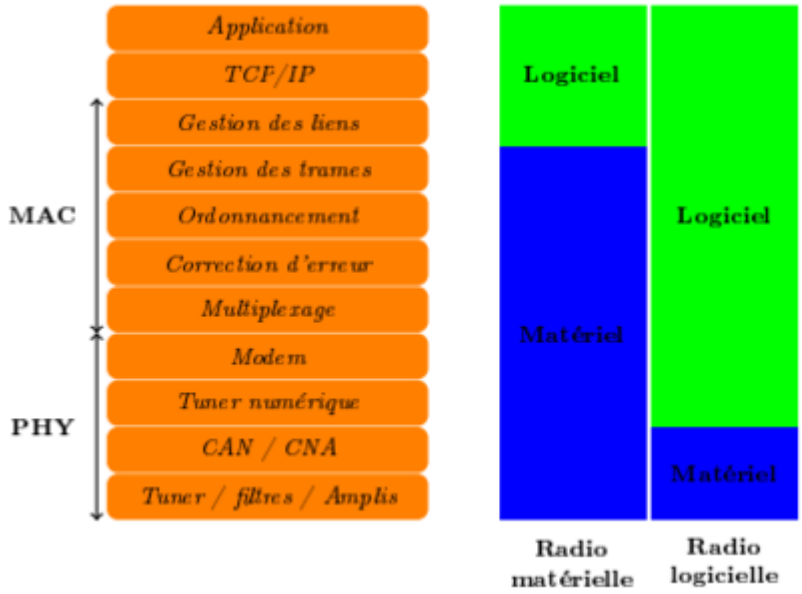

Ilustración 4-Radio material- Radio software

La figura 5 muestra los conceptos de receptor de radio donde la mayoría de la cadena de tratamiento de la señal está hecha sobre la computadora o software específico:

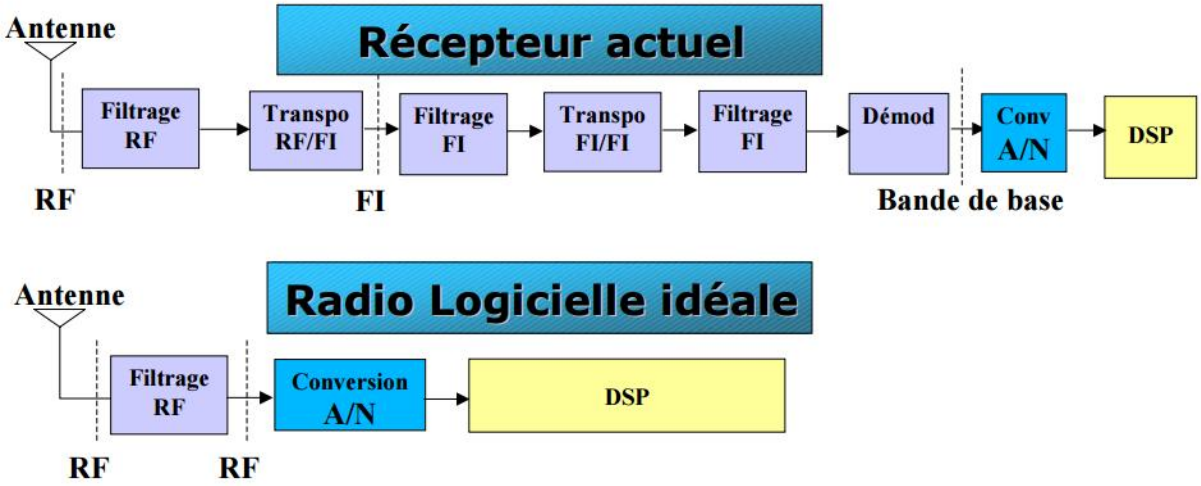

Ilustración 5-SDR ideal

[4]

Para poder trabajar con la mayoría de los estándares de transmisión de radiofrecuencia, un SDR debe ser capaz de operar dentro del espectro de radiofrecuencia más grande posible. Un terminal de SDR multi-estandar debe poder tratar diversas normas de comunicación de radio teniendo diferentes especificaciones, sobre los canales de comunicaciones de banda estrecha o ancha, centrados sobre frecuencias portadoras que pueden alcanzar 6 GHz.

EL tratamiento digital debe estar adaptado a la norma seleccionada y responder a las exigencias en términos de modulación y técnicas de acceso.

5.2.1- Historia de SDR

Empezado el 1991, el proyecto militar SpeakEasy, fue el primero sobre radio definido por software. Había sido iniciado con el fin de paliar los problemas de incompatibilidades entre los medios de comunicación de las diferentes ramas del ejército americano. Este proyecto era muy ambicioso ya que tenía como objetivo permitir el establecimiento de 10 formas de ondas diferentes a partir de un solo equipo. En 1994, una demostración fue quién reveló el éxito del proyecto. El problema era el estorbo del equipo ya que éste ocupaba toda la parte trasera de un camión y su complejidad de diseño (integraba varias centenas de procesadores). Ha sido concebido para ser un sistema multibandas en una gama de 2 a 400 MHz. Las muestras de este prototipo fueron quienes aportaron atención particular al establecimiento de un entorno abierto con el fin de permitir el desarrollo rápido de nuevas aplicaciones. En efecto, el software presente en esta plataforma fue codificado en código ensamblador específico al material utilizado, lo que era pues lejos de ofrecer una portabilidad extensa. Otro punto destacado es que "fue sobredimensionado" y eso permite comprender que al prototipo debía faltarle flexibilidad con el fin de adaptarse a futuras limitaciones. Además, el establecimiento de tal equipo fue extremadamente costoso ya que estaba compuesto por los productos más avanzados disponibles para el mercado en la época.

Spectrum Ware

Spectrum Ware es un proyecto de investigación del MIT empezado en 1994 y que tiene por objeto revelar el potencial de implantación transportable (en el sentido de la implantación sobre varias plataformas) para el SDR. La arquitectura estaba compuesta por un front-end que ponía en ejecución la parte RF y la conversión analógica/digital y digital/analógica. La parte de tratamiento fue implementada sobre una PC bajo Linux. La diferencia notable con el proyecto SpeakEasy era la atención apoyada sobre la portabilidad del software y su reutilización utilizando un sistema operativo dedicado y un diseño basado en un enfoque software orientado a objeto. Así la elección del material que había que utilizar emanaba del software utilizado.

Lo que volvió a salir de este estudio fue que aunque no siendo la plataforma más poderosa del momento, en efecto, la utilización de un sistema operativo y un diseño orientado a objeto inducen indudablemente estados latentes (u overhead), la progresión en término de capacidad de cálculo, de espacio de almacenamiento y de rapidez de ejecución de los recursos materiales que en el futuro lo harán volverse mucho más atractivo. Una evolución será llevada a este prototipo en 1998 por una separación de la señal de tratamiento y de la señal de control y de configuración. Lo que tiene por objeto mejorar todavía la portabilidad y el reutilizabilidad del software. [5]

5.3- Análisis económico

En el mercado existen diferentes alternativas en cuanto a hardware de Radio Definida por Software, siendo factores diferenciadores importantes la capacidad de proceso de la FPGA que incorporan de forma nativa, el ancho de banda en el que pueden operar y las interfaces que disponen. Siendo un factor no menor el precio de venta que puede variar significativamente.

En el caso de nuestro trabajo en Francia, se utilizó la placa BladeRF de Nuand modelo X40, con un costo aproximado de 450€ el cual no debimos afrontar ya que nos fueron provistas dos unidades por la universidad para el desarrollo de nuestro proyecto. Este dispositivo fue de interés para el laboratorio ya que presenta una relación de calidad/precio razonable para una institución educativa en la que podrían precisarse numerosas placas.

De cualquier forma, el proyecto no tuvo como finalidad el desarrollo de un producto comercial que genere potenciales beneficios económicos, siendo los objetivos fundamentales de finalidad educativa.

5.4- Elaboración de nuestro proyecto

5.4.1- Medios utilizados

BladeRF

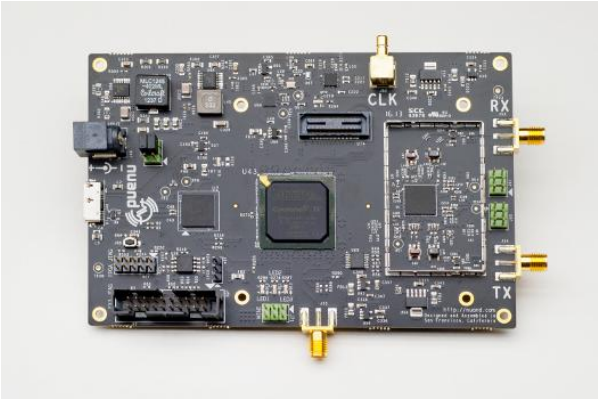

Ilustración 6-Placa BladeRF

BladeRF es una plataforma de radio definida por software diseñada para permitir a una comunidad de amateurs y profesionales, explorar y experimentar las facetas multidisciplinarias de la comunicación RF.

La BladeRf es un dispositivo impulsado totalmente sobre USB que no necesita estar conectado a otra fuente de energía para su funcionamiento normal. Para los usuarios que desean hacer procesamiento de host, USB 3.0 Super Speed es el alto rendimiento ideal, y a la vez, una interface de baja latencia que conecta la PC con la antena. Para aquellos que buscan una solución independiente, la BladeRF cuenta con una entrada de 5V de corriente continua y funciona de una forma autónoma utilizando la FPGA para el tratamiento de la señal. [6]

DIAGRAMA DE BLADERF

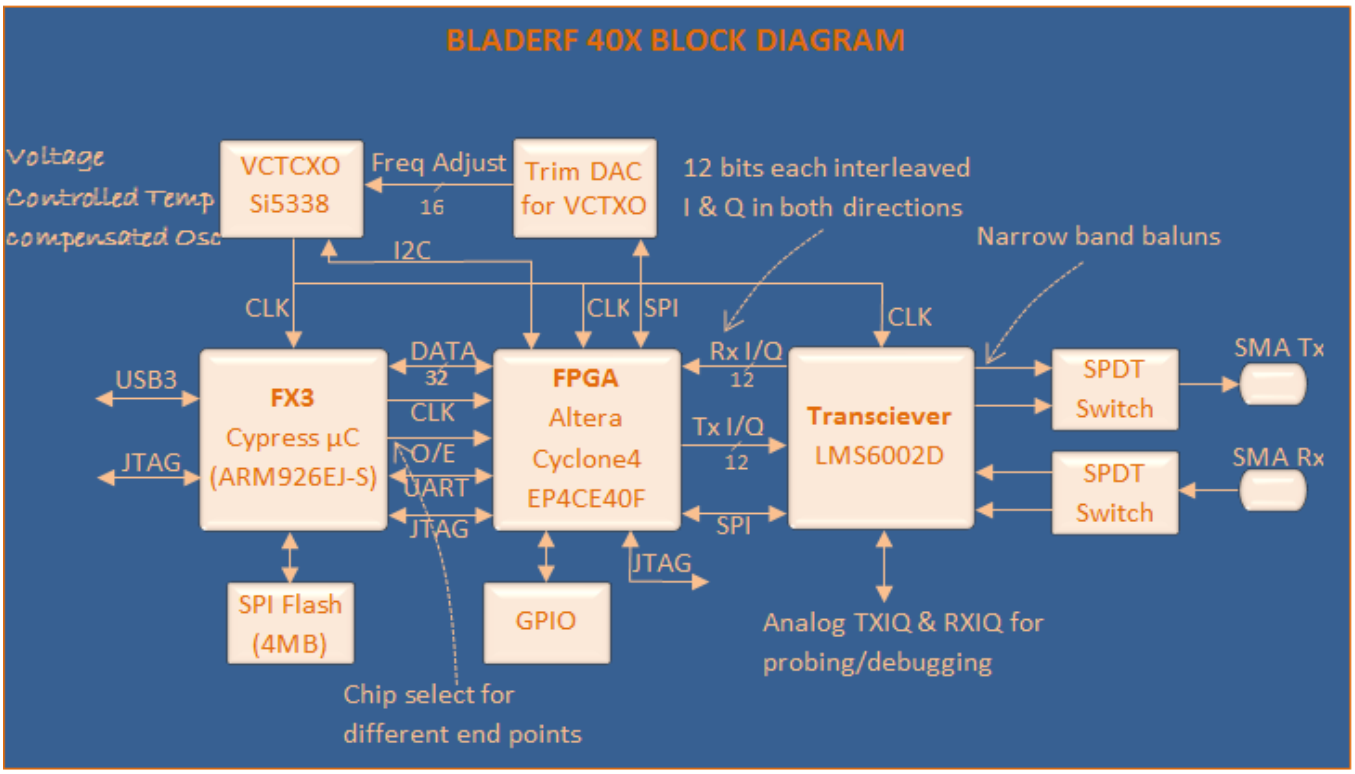

Ilustración 7-Diagrama de la placa BladeRF

TRANSCEIVER LMS6002D

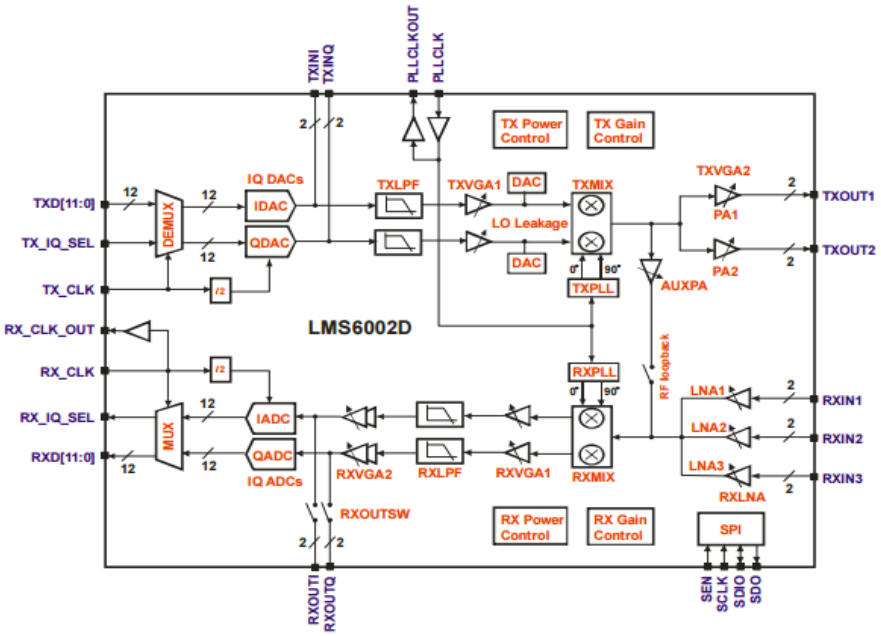

Ilustración 8-Diagrama del transceiver

FX3

El kit de exploración FX3 SuperSpeed de Cypress es una plataforma de desarrollo barato y de simple utilización, que permite a los desarrolladores añadir una funcionalidad de dispositivo USB 3.0 a todo sistema. El kit de exploración FX3 SuperSpeed es concebido para evaluar y desarrollar soluciones con la familia CYUSB301x de controladores de periférico USB 3.0 EZ-USB FX3. El dispositivo EZ-USB ® FX3 es alimentado por un corazón ARM9 totalmente accesible con 512 Kb de RAM. El dispositivo FX3 está dotado de una interfaz totalmente configurable, programable y general (GPIF ™ II) pudiendo interactuar con todo procesador, ASIC, detector de imagen o FPGA.

La interfaz FX3 es quien va a hacer la comunicación entre el ordenador y el FPGA que está sobre la placa, adaptando los datos al formato correcto en las dos direcciones.

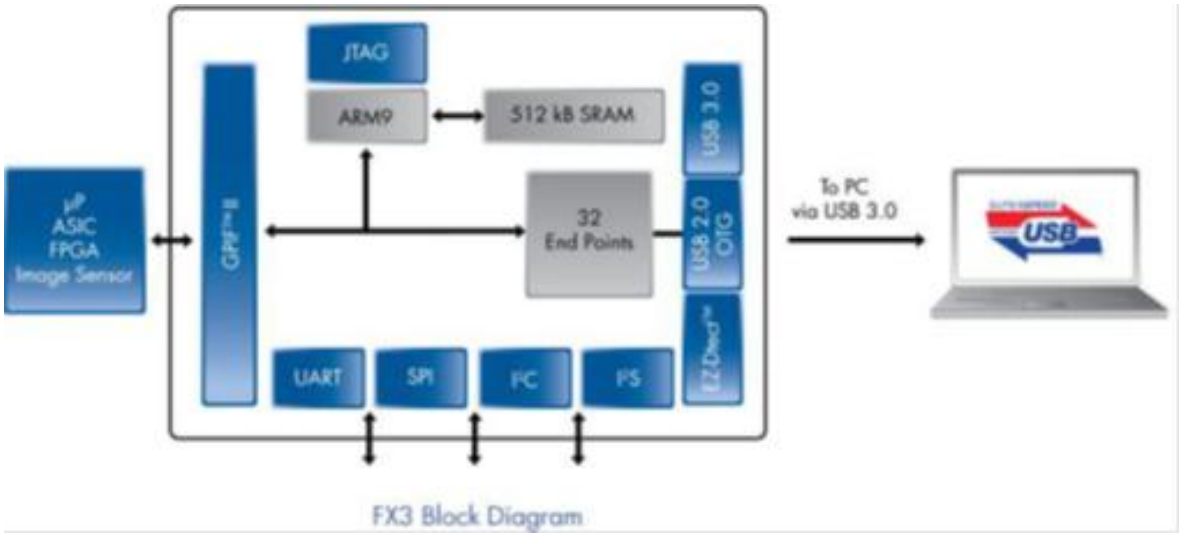

Ilustración 9-Diagrama del FX3

[7]

CLI

Para programar la placa BladeRF, tanto en Windows como en Linux, la interface CLI ha sido utilizada.

Con CLI (en inglés: Command Line Interface) [Figura 10] es posible de insertar los parámetros que van a configurar el emisor y el receptor. Es también posible enviar y recibir archivos en formato .bin o .csv. Este medio nos ha permitido realizar las diferentes simulaciones sobre la cadena de comunicación digital.

- Ganancia RX
- Ganancia TX
- Ancho de Banda
- Frecuencia
- Frecuencia de muestreo
- Loopback
- Version de Firmware
- Calibración

En conclusión, CLI es la interface entre la computadora y la placa de radio definida por software que permite hacer y modificar todo lo que concierne a la placa.

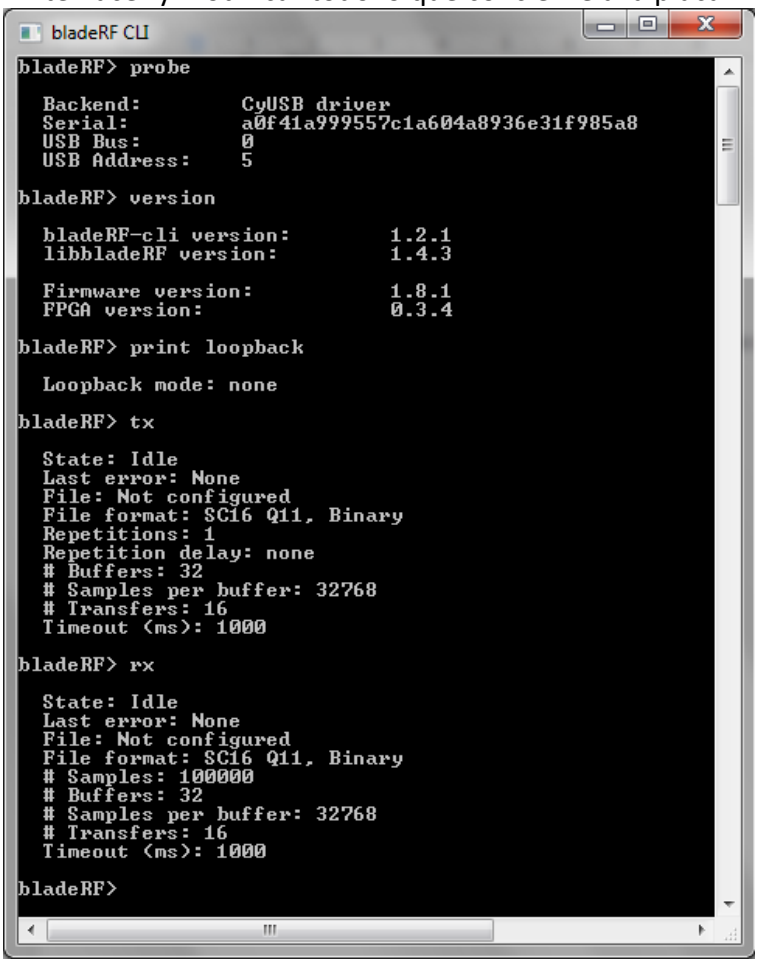

Ilustración 10-Command Line Interface

El comando Print [Figura 11] es particularmente útil en el CLI, ya que permite visualizar las configuraciones actuales de los diferentes parámetros de la placa:

```
print
    Usage: print <param>

    The print command takes a parameter to print.  The parameter is one of:

         Parameter   Description
    ---------------------------------------------------------------------------
         bandwidth   Bandwidth settings
         frequency   Frequency settings
              gpio   FX3 <-> FPGA GPIO state
          loopback   Loopback settings
           lnagain   Gain setting of the LNA, in dB
            rxvga1   Gain setting of RXVGA1, in dB
            rxvga2   Gain setting of RXVGA2, in dB
            txvga1   Gain setting of TXVGA1, in dB
            txvga2   Gain setting of TXVGA2, in dB
          sampling   External or internal sampling mode
        samplerate   Samplerate settings
           trimdac   VCTCXO Trim DAC settings
```

Ilustración 11-Comando Print

Para ajustar las diferentes configuraciones de parámetros de la placa, el comando SET [Figura 12] ha sido utilizado porque esa es la forma más práctica para configurar las etapas siguientes:

```
set
    Usage: set <param> <arguments>

    The set command takes a parameter and an arbitrary number of arguments
    for that particular command.  The parameter is one of:

         Parameter   Description
    ---------------------------------------------------------------------------
         bandwidth   Bandwidth settings
         frequency   Frequency settings
              gpio   FX3 <-> FPGA GPIO state
          loopback   Loopback settings.  Run 'set loopback' for available
                     modes.
           lnagain   Gain setting of the LNA, in dB.  Valid values: 0, 3, 6
            rxvga1   Gain setting of RXVGA1, in dB.  Range: [5, 30]
            rxvga2   Gain setting of RXVGA2, in dB.  Range: [0, 30]
            txvga1   Gain setting of TXVGA1, in dB.  Range: [-35, -4]
            txvga2   Gain setting of TXVGA2, in dB.  Range: [0, 25]
          sampling   External or internal sampling mode
        samplerate   Sample rate settings
           trimdac   VCTCXO Trim DAC settings
```

Ilustración 12-Comando Set

Durante las simulaciones realizadas, los siguientes parámetros han sido utilizados:

- Ganancia RX : RxVga1= 30db
  RxVga2= 3db
- Ganancia TX : TxVga1= -14db
- TxVga2= 0db
- Ancho de Banda: 2MHz
- Frecuencia : 866MHz
- Frecuencia de Muestreo : 10MHz

La elección de los parámetros ha sido hecha sobre la base de los mejores resultados utilizando diferentes configuraciones.

En el caso de la frecuencia, la elección ha sido hecha para el tipo de antena utilizado. Por otro lado, el sample rate y el ancho de banda, han sido elegidos de una manera que permiten evitar el aliasing, como así también, respetando los márgenes de funcionamiento óptimo de la placa.

El teorema de Nyquist menciona que para una correcta transferencia de información, la frecuencia de muestreo debe ser, al menos, el doble de la frecuencia más alta que contiene la señal original. La componente frecuencia corresponde a la mitad de la frecuencia de muestreo, y se denomina frecuencia de Nyquist. La representación de frecuencias por encima de la correspondiente al factor de Nyquist como frecuencias negativas, significa que cuando el número de muestras esté por debajo del doble de la máxima frecuencia presente en la onda analizada, éstas componentes de frecuencias altas pueden mimetizar (imitar) componentes por debajo de la frecuencia de Nyquist y, por tanto, introducir errores en el análisis. Las componentes de frecuencias elevadas, pueden incluso completar varias revoluciones entre muestras consecutivas, lo que implica una pérdida de información. Esta mala interpretación de las frecuencias más altas y más bajas que la frecuencia de Nyquist, se denomina “Aliasing”.

MATLAB

MATLAB es un lenguaje de computación técnica de alto nivel y un entorno interactivo para desarrollo de algoritmos, visualización de datos, análisis de datos y cálculo numérico. Está presente en sistemas de seguridad activa de automóviles, naves espaciales interplanetarias, dispositivos de monitorización de la salud, redes eléctricas inteligentes y redes móviles LTE. Se utiliza para aprendizaje automático, procesamiento de señales, procesamiento de imágenes, visión artificial, comunicaciones, finanzas computacionales, diseño de control, robótica y muchos otros campos.

La plataforma de MATLAB está optimizada para resolver problemas de ingeniería y científicos. El lenguaje de MATLAB, basado en matrices, es la forma más natural del mundo para expresar las matemáticas computacionales. Los gráficos integrados facilitan la visualización de los datos y la obtención de información a partir de ellos. Una vasta librería de toolboxes preinstaladas le permiten empezar a trabajar inmediatamente con algoritmos esenciales para su dominio. [8]

GNU Radio

GNU Radio es un software de desarrollo de herramientas de código libre y abierto que proporciona bloques de procesamiento de señales para implementar software radio. Este puede ser usado con hardware de RF externo para crear radios definidos por software o sin hardware utilizando el entorno de simulación. Es ampliamente utilizado en entornos de aficionados, académicos y comercial para contribuir tanto en la investigación de comunicaciones inalámbricas y sistemas de radio del mundo real.

GNU Radio está licenciado bajo la GNU General Public License (GPL) versión 3. Todo el código es propiedad de la Fundación del Software Libre.

Las aplicaciones de GNU Radio son escritas utilizando el lenguaje de programación Python, mientras que el suministro de herramientas críticas de procesamiento de señales que requieren alto rendimiento son implementados en C++ usando extensiones de procesamiento de punto flotante, cuando este está disponible. Así, el desarrollador es capaz de implementar, de manera simple, sistemas de radio de alto rendimiento funcionando a tiempo real aprovechando el ambiente de desarrollo de aplicaciones de manera inmediata.

Aunque no es una herramienta principalmente de simulación, GNU Radio complementa el desarrollo de algoritmos de procesamiento de señales a partir de datos previamente grabados o generados, evitando la necesidad de hardware de RF. [9]

5.4.2- Los métodos

A fin de implementar la cadena de transmisión, cada bloque que intervino en las Figuras 13-14, ha sido programado en Matlab.

Durante su desarrollo, como es posible ver sobre el diagrama FAST, hubo una división de actividades. Por un lado, Francisco ha programado cada parte del proceso de transmisión, y por otro lado, Paulo ha hecho la programación del procedimiento de recepción. La sincronización, una de las etapas más difíciles, ha sido llevada a cabo conjuntamente porque es una parte muy importante para el correcto funcionamiento del sistema entero.

En la parte emisora, la primer etapa consistió en adaptar el formato del archivo a transmitir. Un preámbulo, ha sido añadido a la información "útil" para hacer posible la detección del comienzo de la trama en el receptor, y también, fue de gran importancia al momento de corregir la fase de la señal recibida. Luego, esta información ha sido modulada asignándole un tipo de constelación particular.

A partir de la información modulada, se realizó un agregado de muestras en un factor N.

Enseguida, la señal debió ser filtrada por un filtro de raíz de coseno sobrealzado con el objetivo de evitar la interferencia entre símbolos en el momento de la toma de decisiones en el receptor, todo controlando el espectro que ha sido ocupado en el canal. La última etapa consistió en adaptar la energía de las muestras a la banda del convertidor digital/analógico para aprovechar la resolución que ofrece el CDA.

TX

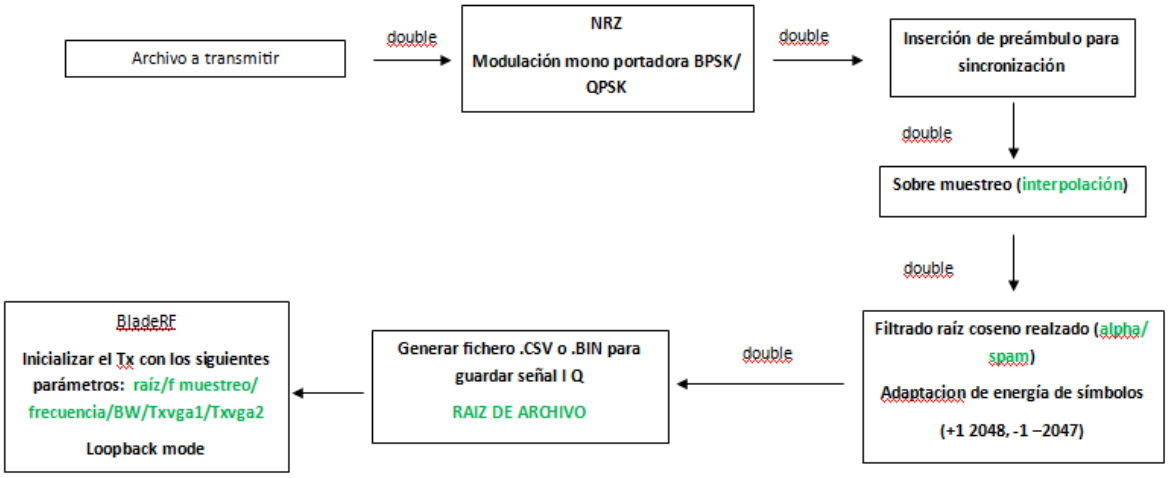

Ilustración 13-Diagrama TX

Del lado del receptor, la primer acción a realizar luego de abrir el archivo recibido, consistió en hacer el filtrado adaptado (igualmente, filtro de raíz de coseno sobrealzado.

Después, la detección del comienzo de la trama ha sido realizada, utilizando el preámbulo, quien también fue útil para estimar el offset de fase de la portadora y su corrección.

Cuando el proceso está finalizado, a la señal se le deben quitar las muestras agregadas en el mismo factor N. A continuación, el proceso de mapeo inverso fue hecho para poder recuperar la información “útil” que ha sido enviada. Esta información es adaptada según el tipo de archivo original.

RX

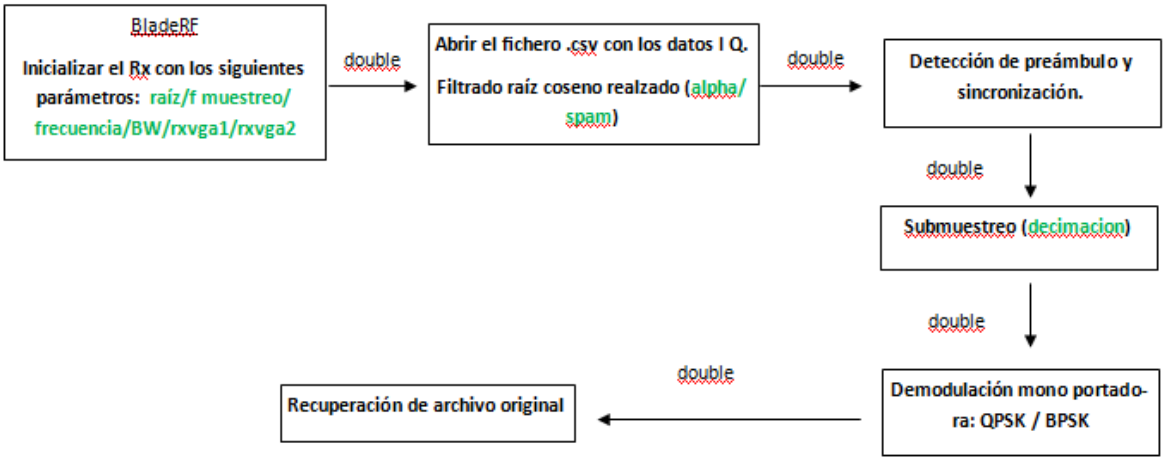

Ilustración 14-Diagrama RX

Modulación de la señal

Modulación engloba el conjunto de técnicas que se usan para transportar información sobre una onda portadora, típicamente una onda sinusoidal. Estas técnicas permiten un mejor aprovechamiento del canal de comunicación lo que posibilita transmitir más información de forma simultánea además de mejorar la resistencia contra posibles ruidos e interferencias. Según la American National Standard for Telecommunications, la modulación es el proceso, o el resultado del proceso, de variar una característica de una onda portadora de acuerdo con una señal que transporta información. El propósito de la modulación es sobreponer señales en las ondas portadoras.

Básicamente, la modulación consiste en hacer que un parámetro de la onda portadora cambie de valor de acuerdo con las variaciones de la señal moduladora, que es la información que queremos transmitir.

La modulación es de orden N, si cada símbolo transmitido puede comprender N valores posbiles. En la mayoría de los casos, el símbolo esta formado a partir de un vector de K bits, lo que conduce a N=2K.

El modulador es un modulo de la cadena de comunicación digital que forma símbolos ak (en general complejos), en función de vectores de bits en su entrada. El demodulador es un modulo situado sobre el terminal receptor, que estima los valores correspondientes a los símbolos ak, asi logrando una estimación ak a su entrada. Según el criterio de diseño del modulador, dos parámetros son importantes de definir:

- tipo de modulacion o constelacion sobre el plano complejo,
- étiquetas de puntos de la constelacion (mapping).

Dentro de las comunicaciones digitales, la modulacion de amplitud y la modulacion I Q son utilizadas. Por lo tanto, el tipo de modulacion IQ es una manera eficiente de transferir la informacion. Un modulador IQ puede generar AM, FM y PM.

El grafico IQ muestra [Figura 15]:

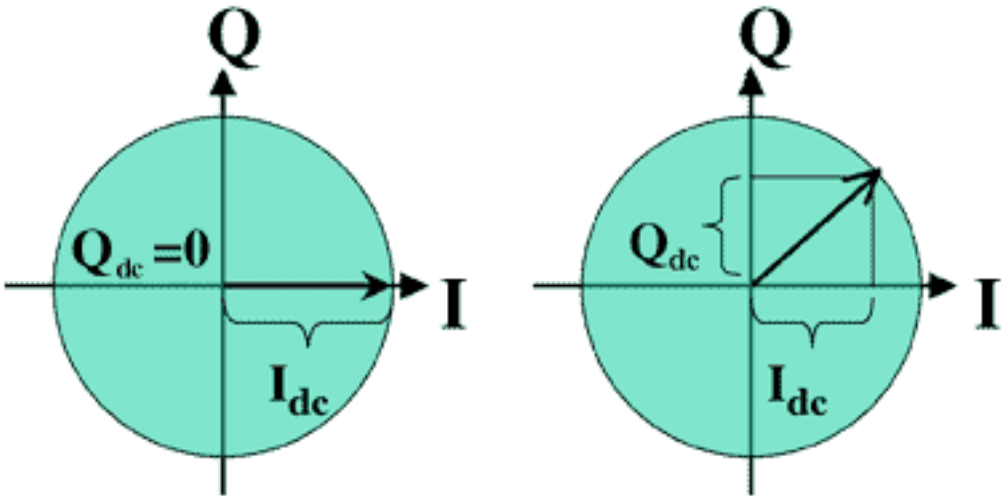

Ilustración 15-Plan I/Q

Tipos de modulación:

Phase-shift keying (PSK)

La modulación de fase se encuentra dentro de una familia de modulaciones digitales, en la cual la información a transmitir en la señal en banda base será transmitida variando la fase de la señal portadora de referencia.

Como para todo tipo de modulación digital, la fase involucrada no puede adoptar más que un número finito de valores. Cada uno de esos valores representa un único numero binario, por lo que dependiendo del número utilizado será la capacidad del sistema para transmitir el mensaje binario.

Binary phase-shift keying (BPSK)

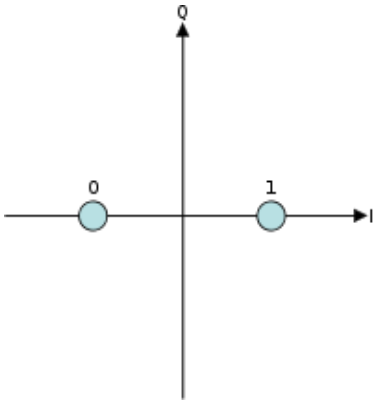

Ilustración 16-Constelación BPSK

BPSK es la forma más sencilla de PSK. Este esquema utiliza dos fases que son separadas 180º [Figura 16]. Esta modulación es la más robusta de todas las PSK porque es necesaria una gran deformación de la señal para que el demodulador cometa un error al realizar el reconocimiento del símbolo.

Debido a que solo se utilizan dos fases diferentes no se puede modular más que un bit por símbolo.

En lo que concierne al preámbulo utilizado, sus 128 bits fueron modulados en BPSK, por lo que todos tienen la misma fase. Dicha elección hará más sencillo el proceso de estimación del canal realizado en el receptor.

Quadrature phase-shift keying (QPSK)

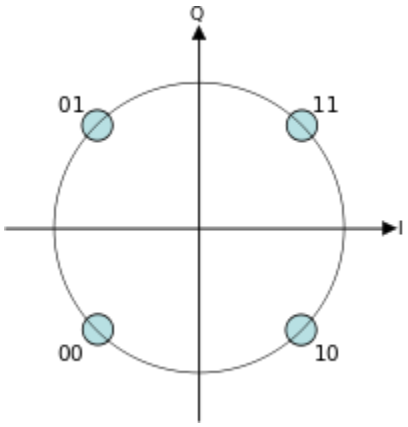

Ilustración 17-Constelación QPSK

Esta modulación utiliza un diagrama de constelación a cuatro puntos, equidistantes alrededor de un círculo. Con 4 fases, QPSK puede codificar dos bits por simbolo, dicho esquema de gray mostrado permite de aumentar la tasa de bits a razón de 2 comparado con un sistema BPSK manteniendo la banda pasante de la señal o, mantener la tasa de bits constante y reducir el ancho de banda en mitad.

Bien que QPSK puede considerarse como una modulación en cuadratura, es además fácil de ser considerada como dos modulaciones independientes. Considerando dicha interpretación, los bits pares (o impares) son utilizados para modular la componente en fase (I), siendo los bits impares (o pares) utilizados para la componente en cuadratura (Q).

Dicha consideración presenta el tipo de modulación utilizado en la cadena transmisión implementada ya que la arquitectura de la placa BladeRF fue diseñada para trabajar con señales del tipo I Q.

Siendo esta además una modulación robusta que permite realizar un buen análisis de los cambios de fase e imperfecciones introducidas por los sistemas I Q en la señal.

Luego, para recuperar la información transmitida en el lado receptor, el proceso de de-mapping fue realizado. La toma de decisión para reasignar los valores originales a los símbolos recibidos estuve basada en un criterio de decisión directa basada en umbrales bien definidos, luego de realizar las correcciones fase pertinentes.

Sobre-Submuestreo

El sobremuestreo consiste en muestrear la señal a una frecuencia mucho más elevada de lo que exigido por el teorema de Shannon.

Es posible de utilizar el oversampling(Sobremuestreo) para simplificar los filtros de emisión y reconstrucción y también, para la reducción de ruido que está agregado. El ruido de cuantificación aparece aleatoriamente sobre la señal de entrada y se distribuye regularmente sobre todo el espectro. Utilizar el sobremuestreo permite distribuir regularmente el ruido entre 0Hz y la mitad de la frecuencia de muestreo. Cuando la señal digital atraviesa el filtro pasa bajos de decimación, la mayor parte de ese ruido será eliminado, lo que aumenta la relación señal/ruido efectiva; la señal permanece igual pero el ruido se reduce. Este enfoque permite utilizar en la entrada, convertidores A/D rápidos y de buena resolución, que harán lo que normalmente hacen convertidores A/D de resolución más elevada.

Por lo tanto, el tiempo de símbolo será mucho más grande que el tiempo de un solo bit de información original. En las simulaciones que hemos desarrollado, ese tiempo es 20 veces el tiempo de bit.

Filtrado adaptado

Los filtros adaptados raíz de coseno sobrealzado [Figura 18] son una clase de filtros de Nyquist. Su rol es de imponer a los símbolos una forma tal que el riesgo de ISI ((Inter-Symbol Interference) sea fuertemente reducido al momento de la toma de decisión sobre el valor del símbolo transmitido (la señal digital está por lo tanto, en su banda base). Una forma de base elemental ha sido impuesta, de manera que la forma de onda correspondiente a un símbolo tenía una amplitud máxima al momento de la toma de decisión sobre el valor del símbolo, y una amplitud nula en los instantes de la toma de decisión sobre los otros símbolos. Desde el punto de vista temporal, la forma de onda impuesta por este tipo de filtro se parece a una función de "seno cardinal".

El valor del factor de redondeo α, comprendido entre 0 y 1, depende del sistema puesto en marcha. El es determinado notablemente en función de un compromiso a efectuar entre la banda de frecuencia ocupada (óptima para un α próximo a 0) y la reducción de riesgo de IS (óptima para α próxima a 1). [11]

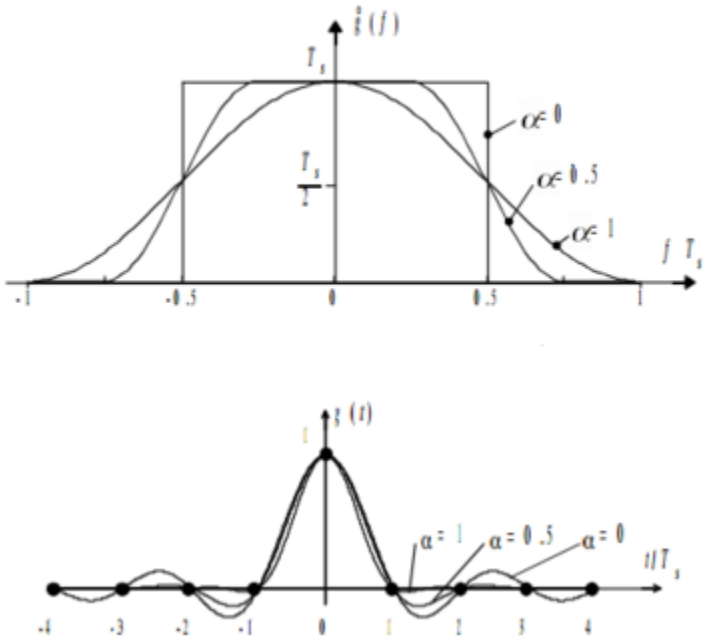

Ilustración 18-Filtro en raíz de coseno sobrealzado

Para optimizar la interferencia entre símbolos y la puesta en forma de la señal que hay que transmitir, la siguiente ventana de filtrado [Figura 19] ha sido utilizada [α=0.2 Span=5*Ts M=12]:

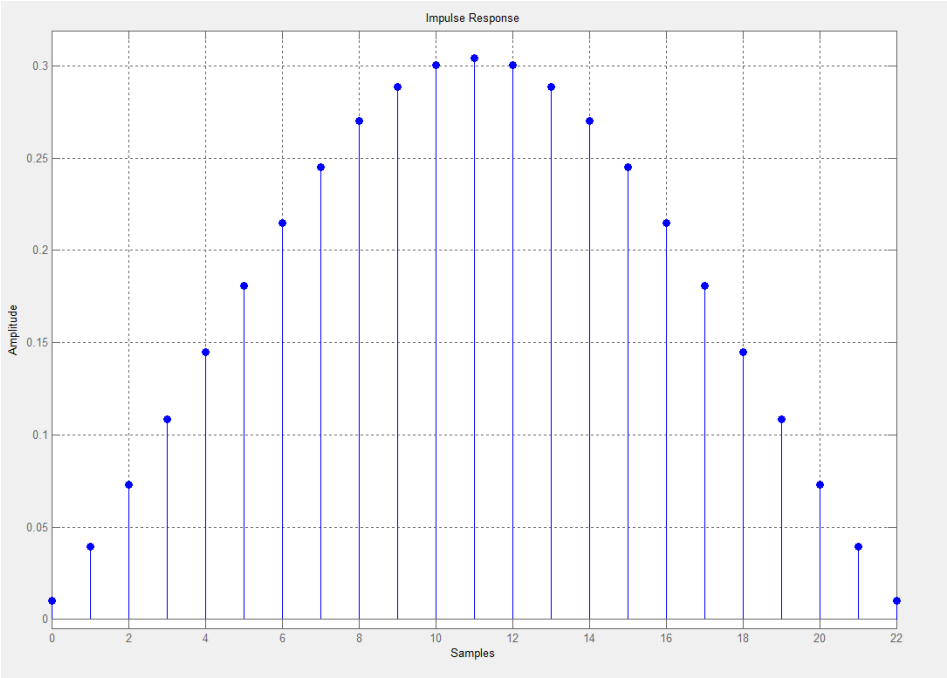

Ilustración 19-Ventana de filtrado utilizada

Adaptación de la amplitud de las muestras

Luego de haber realizado el filtrado de emisión de la señal I y de la señal Q, es necesario adaptar la amplitud de las muestras para obtener la mejor resolución en la conversión Digital-Analógica que va a ser hecha en la placa. Para hacer esto, la amplitud de las muestras debe estar entre 2047 (valor decimal) para representar +1, y -2048 (valor decimal) para el -1.

El convertidor presentara en la salida un número finito de códigos numéricos, correspondiendo a una gama de tensión analógica de entrada limitada: esta es la banda de conversión (o tensión de fondo de escala) del convertidor.

El CAD-CDA que está sobre la placa BladeRF tiene una resolución de 16 bit y una banda pasante de 38.4Mhz +/-1ppm.

La resolución corresponde a la variación de una unidad del código binario, esta unidad es igual a la variación del bit de menor peso (LSB=least signifiant bit en inglés). Si VMAX representa la banda de conversión y N el número de bits del convertidor, surge la relación siguiente:

$$LSB = \frac{\Delta V_{max}}{2^n}$$

Del lado receptor, a la salida del convertidor analógico-digital había también valores que oscilan entre [-2048 y 2047]. Estos valores fueron normalizados para continuar con el proceso del tratamiento de la señal.

En función del método para enviar el archivo, dependerá si el tamaño de los datos debe estar entre [-2048 2047] o [-1 1] al momento de enviarlos por el CLI.

La interface CLI debe hacer una conversión automática de formato .CSV a .BIN, en este caso los datos a su entrada serán entre [-2048 2047], pero si el formato binario es el elegido, el tamaño será entre [-1 1] porque ese es el formato nativo del convertidor. [12]

Diagrama de ojo

El diagrama del ojo [Figura 20] es un oscilograma que representa datos digitales surgidos de un receptor. Éstos son muestreados de manera repetitiva y son aplicados sobre la entrada de desviación vertical, mientras que el disparo (desviación horizontal) es sincronizado con flujo de la señal. El nombre de este diagrama viene por el hecho de que para número de codificación, el motivo obtenido se parece a una continuación de los ojos encuadrados por dos carriles horizontales.

Los numerosos criterios de realización pueden ser deducidos de este análisis. Si las señales son demasiado largas, demasiado cortas, mal sincronizadas con relación al reloj del sistema, de nivel demasiado importante o demasiado débil, demasiado afectadas por ruido, demasiado lentas en el momento de los cambios de estado, o que contienen demasiados adelantamientos o inercia, el diagrama del ojo las pondrá en evidencia. Un ojo abierto corresponderá a una señal que contiene un mínimo de distorsión. La distorsión de la forma de onda de la señal, pudiendo ser atribuida a la interferencia intersímbolo o al ruido, se traduce por un cierre del ojo.

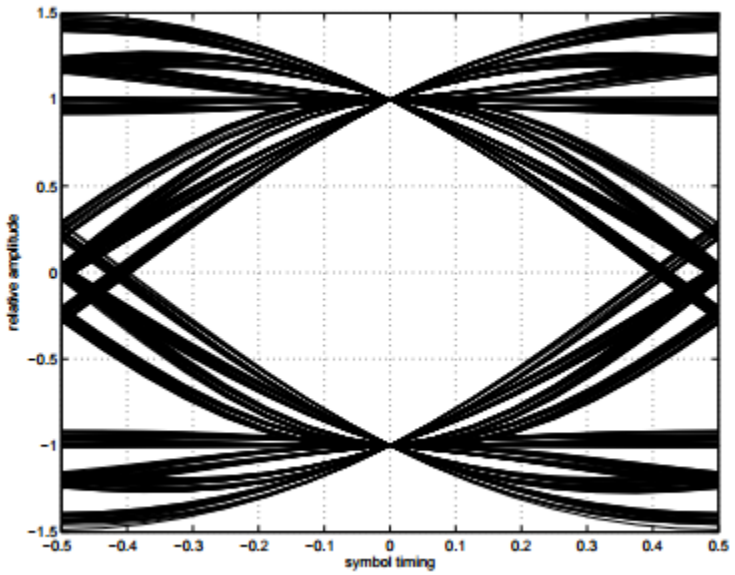

Ilustración 20-Diagrama de ojo

Durante el desarrollo de nuestro proyecto, hemos realizado diferentes pruebas para comprobar el correcto funcionamiento de la programación, y en cada una de ellas, aplicamos el diagrama de ojo. La siguiente imagen muestra un diagrama obtenido en la recepción de la información que se asemeja al ideal:

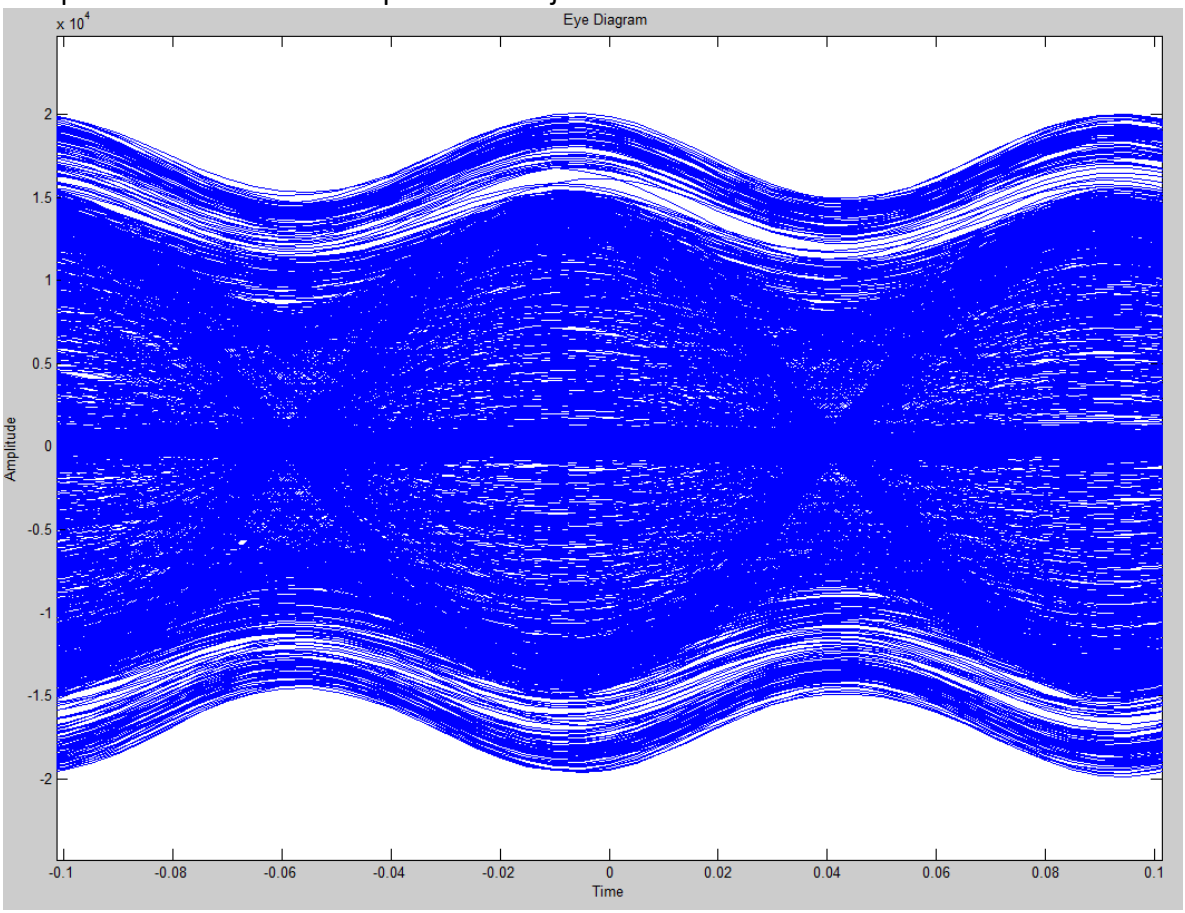

Ilustración 21-Diagrama de ojo obtenido

Diagrama de transición

El diagrama de transición [Figura 21] se parece al diagrama de la constelación, porque los dos muestran la señal sobre el plano complejo.

Sin embargo, el diagrama de transición muestra las transiciones de la señal entre los diferentes símbolos que forman la constelación.

La forma que toma el diagrama depende del coeficiente de roll off del filtro, como así también del número de muestras por símbolo.

La imagen siguiente representa una modulación QPSK, quien tiene sus cuatro símbolos en cuadratura.

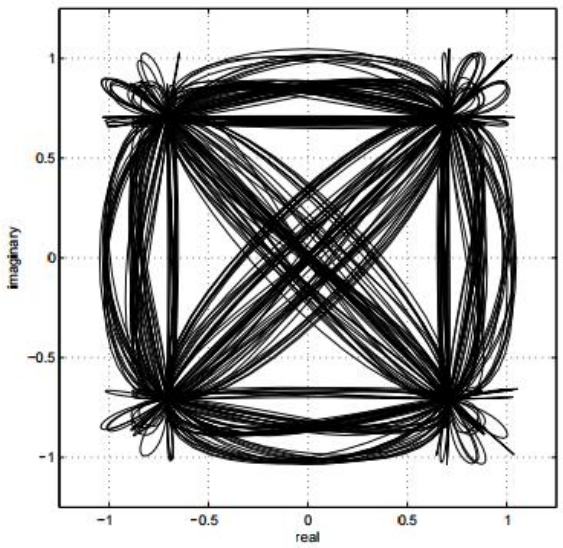

Ilustración 22-Diagrama de transición

Particularidades de un sistema I Q

En los sistemas I-Q, habrá fuentes de error [Figura 22] ya que tanto las señales I y Q siguen caminos diferentes sobre la capa física. El offset de amplitud es una consecuencia de utilizar diferentes amplificadores para cada uno. Otro aspecto que importa es el ruido de fase que va a ser añadido a las señales por el oscilador local. Así también, si la diferencia de fase entre el canal I y Q no es exactamente 90 °, esto añadirá un cambio notable sobre la constelación.

Otro factor a considerar es el offset de DC que producirá un desfase de amplitud constante entre los dos.

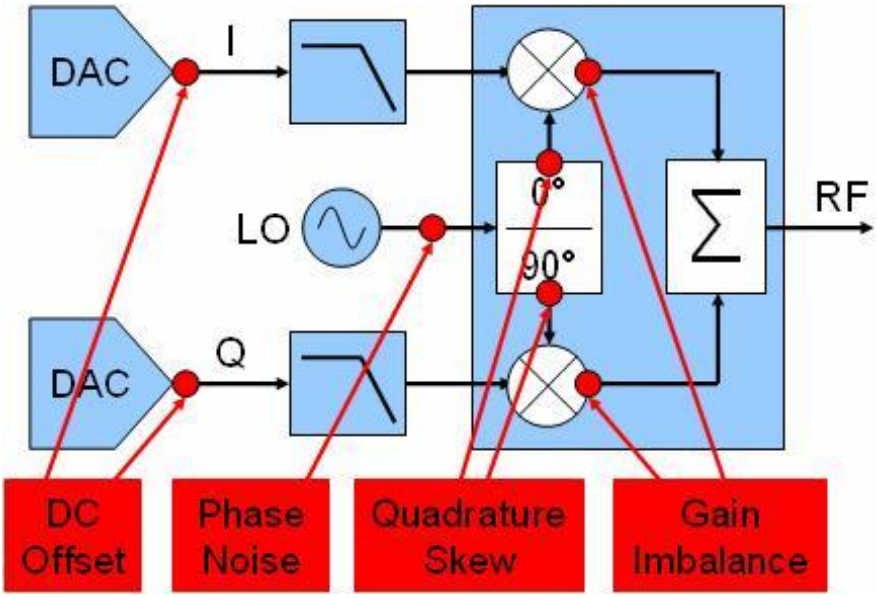

Ilustración 23-Fuentes de error de las señales I Q

Las figuras 23-24 muestran los efectos sobre la constelación, tanto de la parte material que interviene como así también del canal real.

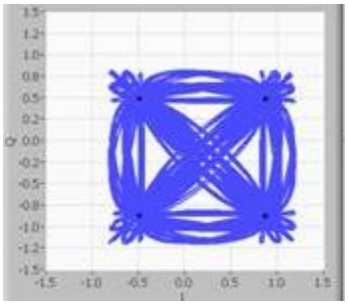

Ilustración 24-Offset de amplitud

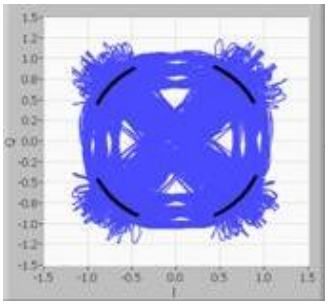

Ilustración 25-Offset de fase

A causa de que cada placa tiene diferente 'clock', un offset de fase ha sido encontrado. Eso quiere decir que entre dos símbolos sucesivos de la información había una diferencia de fase a corregir.

Sincronización

Los datos son formateados bajo la forma de una trama compuesta de un preámbulo, seguido de la información propiamente dicha. Este preámbulo ha sido necesario en dos casos al momento de la recepción, ya que su composición es conocida (su cantidad de símbolos y la amplitud y la fase de cada símbolo). Como indica más arriba, él está compuesto por 128 bits que son modulados en el tipo BPSK.

Por un lado, pudo ser utilizado para conocer el offset de fase que la señal recibida tenía, aplicando la ecuación siguiente:

$$\widehat{\Phi} = \frac{1}{pre} + \sum_{n=k}^{k+pre-1} (\angle[S_{n+1}] - \angle[S_n])$$

pre = numero de muestras del preámbulo
k = índice del primer símbolo del preámbulo
S = símbolo recibido en banda base

El offset de fase ha sido por lo tanto, estimado como la media de la suma de todas las diferencias de fase entre dos símbolos sucesivos del preámbulo. Cuando este valor fue obtenido, el offset de fase pudo ser compensado al nivel de la señal recibida:

$$S'_n = \frac{S_n}{e^{i\widehat{\Phi}(n-k)}} \quad \forall n \in \{k, \dots, k+M\}$$

S_n '=símbolo en banda base compensado

Antes de la corrección de fase, hizo falta encontrar el preámbulo que ha sido utilizado para realizar eso, y es por eso que es necesaria una correlación entre la información recibida y una cadena de datos que sea igual al preámbulo en el receptor. Esta comparación es posible de realizar porque la cadena contiene solamente valores "1" y ellos tienen la amplitud más grande de toda la información recibida (los otros datos recibidos tienen amplitudes entre -0,707 y 0,707).

Por otro lado, el preámbulo fue útil para detectar el comienzo de la trama para realizar la demodulación, cuando la fase de toda la señal esté ya corregida. Y por consecuencia, tomar solamente la parte necesaria para continuar con el proceso de mapping inverso, para reconstruir la señal original. [13]

5.3.3- Loopback

Después de haber programado en Matlab todos los bloques que van a formar parte de la cadena de transmisión mono portadora, la transmisión de datos en modo Firmware Loopback ha sido realizada para analizar la señal y también la constelación de símbolos QPSK a transmitir.

La posibilidad de implementar el Loopback presentó una gran herramienta al momento de hacer las simulaciones con el objetivo de optimizar la cadena como así también el valor de los parámetros.

La placa BladeRF permite testear los diferentes modos de loopback que son los siguientes:

- `bb_txlpf_rxvga2  Baseband loopback: TXLPF output --> RXVGA2 input`
- `bb_txlpf_rxlpf  Baseband loopback: TXLPF output --> RXLPF input`
- `bb_txvga1_rxvga2  Baseband loopback: TXVGA1 output --> RXVGA2 input.`
- `bb_txvga1_rxlpf  Baseband loopback: TXVGA1 output --> RXLPF input`
- `rf_lna1  RF loopback: TXMIX --> RXMIX via LNA1 path.`
- `rf_lna2  RF loopback: TXMIX --> RXMIX via LNA2 path.`
- `rf_lna3  RF loopback: TXMIX --> RXMIX via LNA3 path.`
- `firmware  Firmware-based sample loopback.`
- `none  Loopback disabled - Normal operation.`

Hay dos tipos diferentes de Loopback, aquellos que comienzan con ‘bb’ y los otros que comienzan con ‘rf’. Eso quiere decir que el modo bb realiza una comunicación con la señal en banda base y el modo rf realiza una comunicación de la señal en banda transpuesta para analizar los efectos de los amplificadores.

La placa también permite realizar un tercer modo de loopback, Firmware Loopback, que funciona a nivel de la interface FX3.

La figura 25 muestra el diagrama interno de la placa para una mejor comprensión de su funcionamiento.

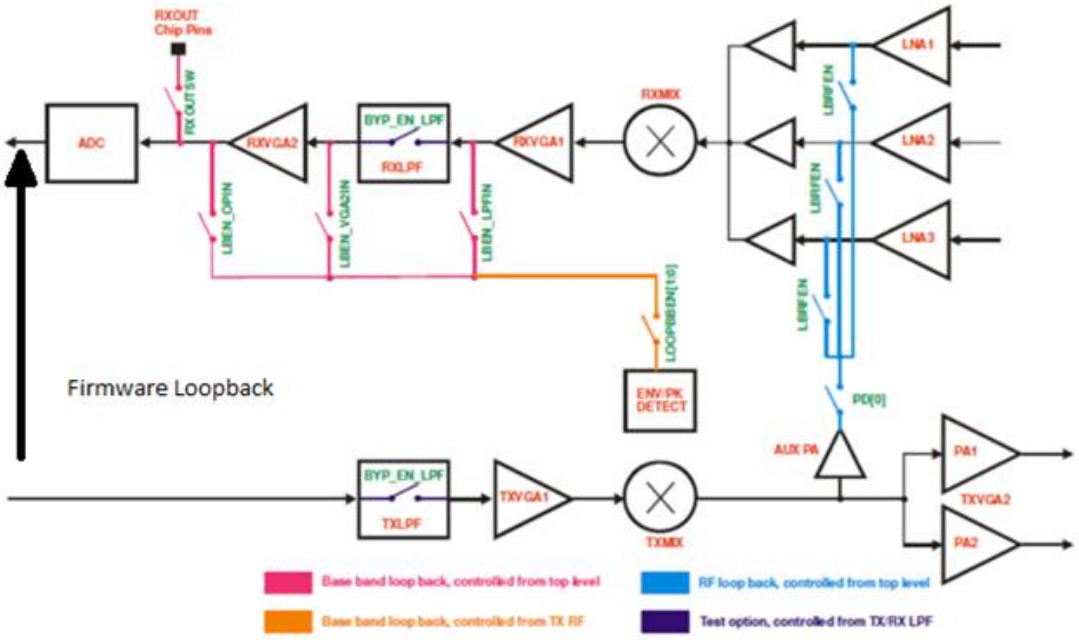

Ilustración 26-Transceiver Loopback

Para configurar la placa, utilizando CLI, con el modo de loopback elegido, se deben seguir los siguientes pasos:

- Loopback mode: none
- bladeRF> set loopback
- set loopback <mode>, donde mode es el tipo de loopback a utilizar.
- bladeRF> print loopback, este comando es utilizado para visualizar el modo activo.

A continuación, las figuras 26-27-28-29 muestran una simulación quien ha sido hecha con el modo firmware, y con los siguientes parámetros:

- TX/RX Frequency: 1 GHz
- Sample rate: 10 MHz; Tsymbol/Tbit= 20
- Bandwidth: 2.5 MHz

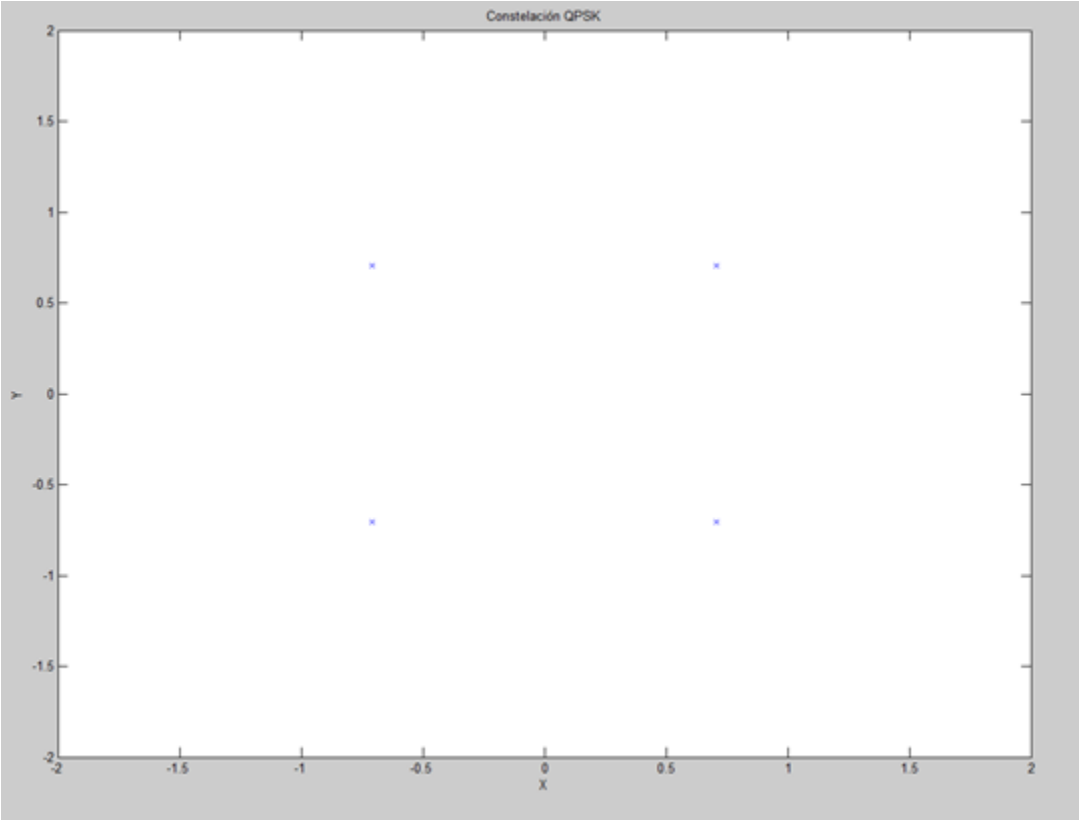

Ilustración 27-Constelación de la modulación QPSK

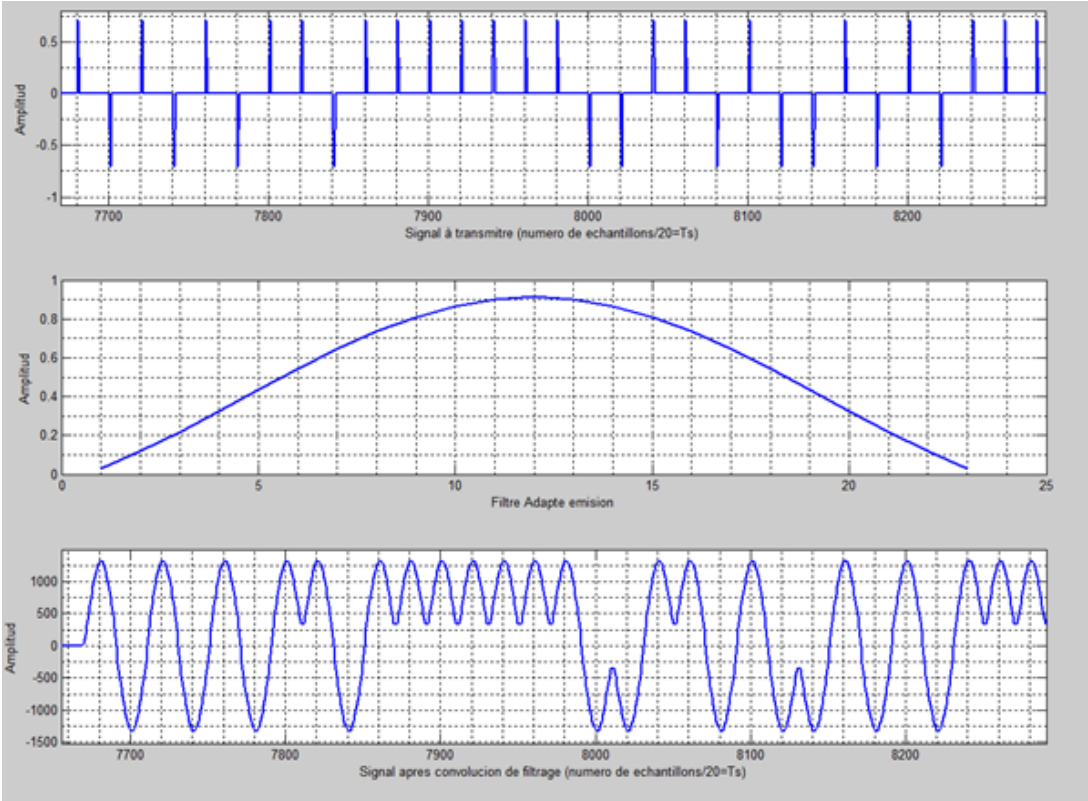

Ilustración 28-Señal a transmitir antes y después del filtro adaptado-filtro adaptado de emisión

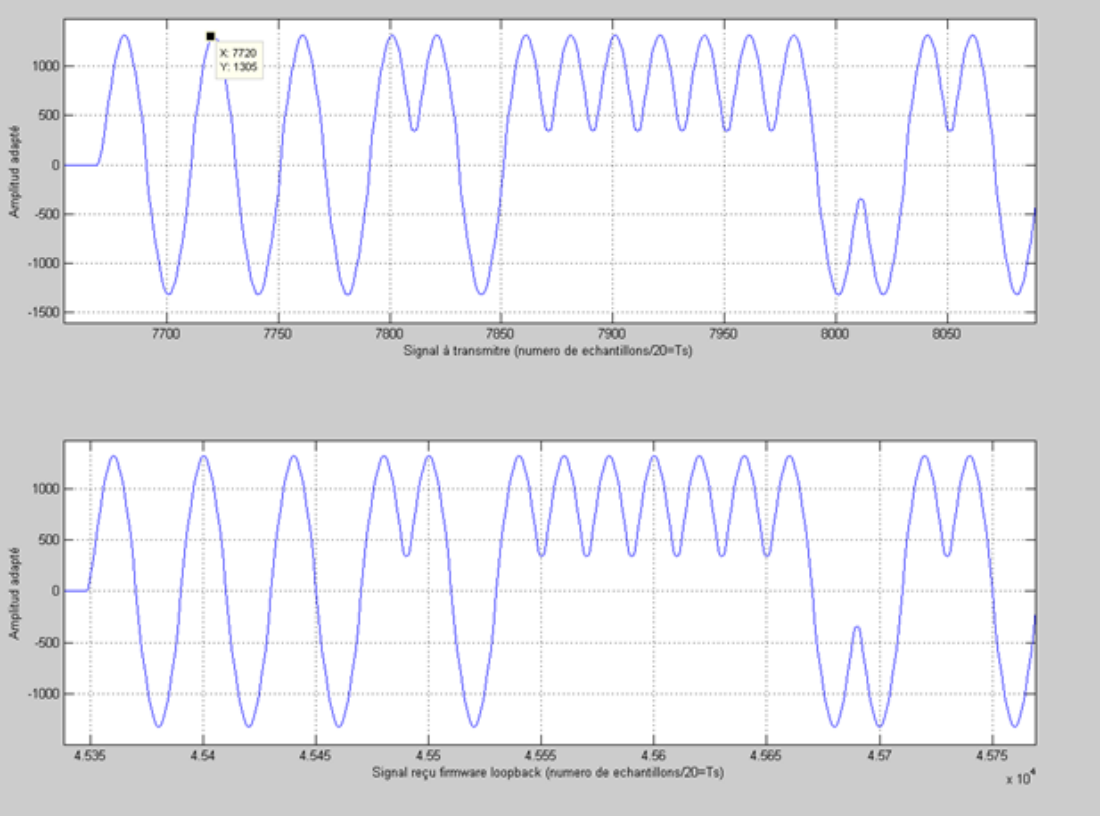

Ilustración 29-Señal enviada y señal recibida

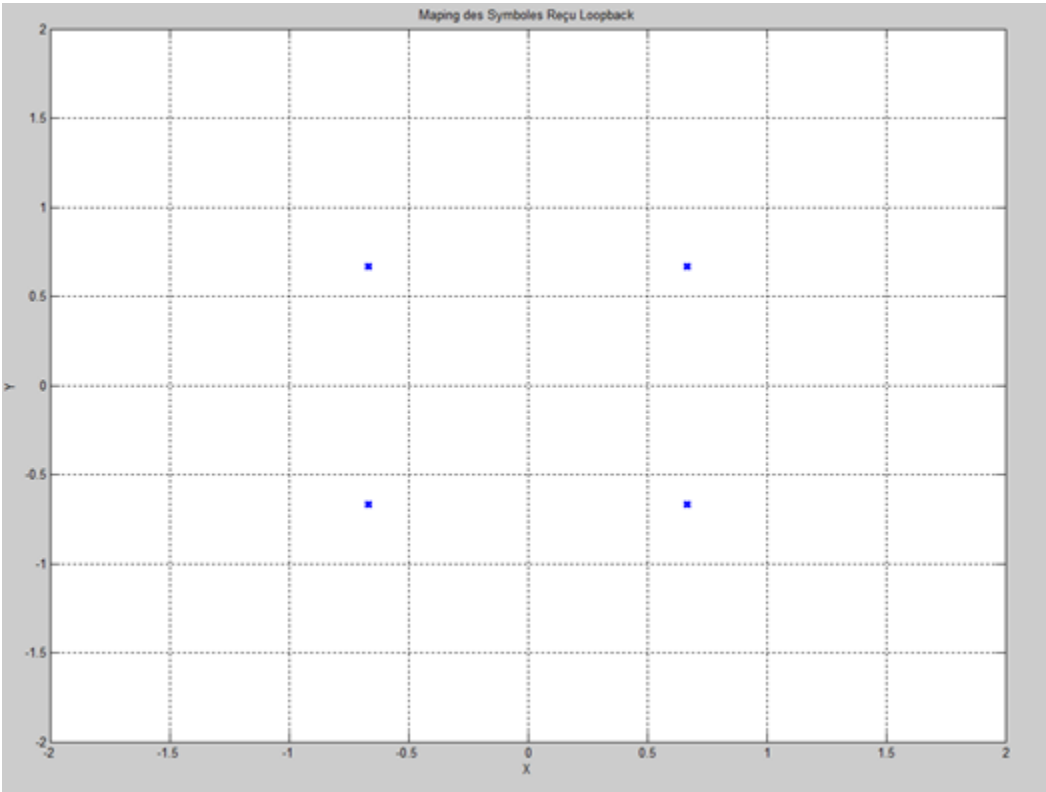

Ilustración 30-Constelación después del filtrado y submuestreo

Otra simulación [Figura 30] ha sido realizada para visualizar la constelación que resulta después de la conversión de la señal a alta frecuencia y viceversa. Este modo de loopback permite estimar los cambios que se van a producir sobre la señal entre los canales I y Q: (rf_lna3 loopback)

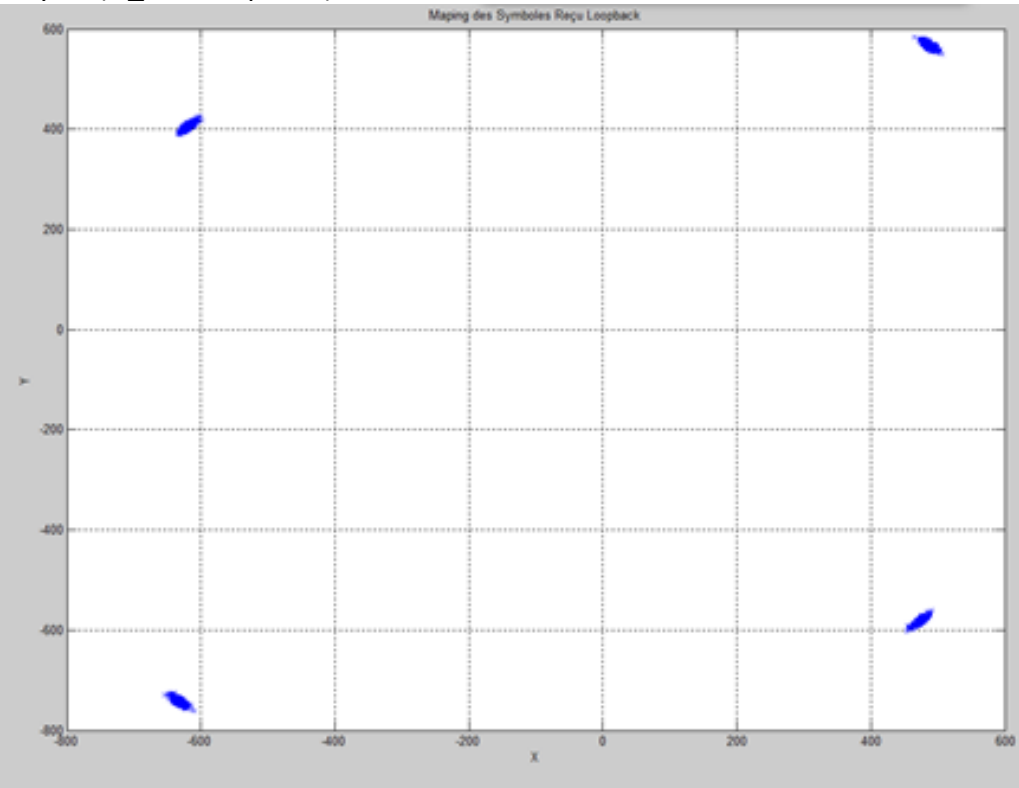

Ilustración 31-Loopback modo rf_lna3

A partir de las imágenes mostradas, es posible ver que los símbolos cambian de lugar a causa de etapas electrónicas de conversión sobre la placa. Eso muestra un primer enfoque para visualizar las fuentes de error mencionadas anteriormente. El demodulador es capaz de reconocer los símbolos que han cambiado de posición sobre la constelación para reasignarles su valor original.

La próxima etapa del proyecto ha sido realizar una estimación del canal real para contrarrestar esos efectos que desnaturalizaron la señal.

La siguiente imagen muestra los resultados de una transmisión en modo Loopback, siendo la parte superior el esquema de una "trama" con su respectivo preámbulo en ensayo, que contaba con 128 unos más 128 ceros a fin de evaluar el mejor preámbulo. Mientras que en la parte inferior de la imagen se puede visualizar la señal recibida en una ventana temporal mayor, teniendo múltiples tramas contenidas en la misma, analizando una mínima distorsión no significativa.

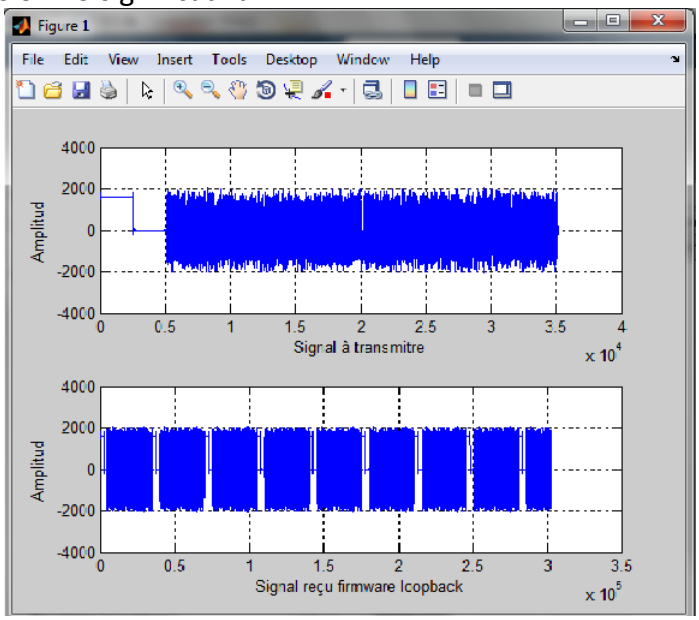

Ilustración 32-Prueba Loopback

6- Problemas y dificultades eventualmente encontrados

Instalación de softwares (GNU Radio, CLI, Gr-Osmosdr):

Antes de comenzar con el proyecto fue necesaria la instalación de diversos controladores en los ordenadores para llevar a cabo las actividades previstas. Es por ello que fue necesario esperar a que el encargado de informática del laboratorio nos proveyera todo lo necesario. La instalación de algunos programas demandó más tiempo del necesario debido a problemas de compatibilidad con los sistemas operativos y licencias. A causa de ello, el avance del trabajo se vio retrasado considerablemente en su primera etapa.

Fue requerido mayor tiempo del que había sido estimado debido a la falta de experiencia suficiente para trabajar con los programas propuestos, y además, siendo el dominio de la radio definida por software un tema desconocido hasta el momento de aplicación.

Formato de archivo:

La utilización de la interface CLI para enviar el fichero en formato csv por defecto nos dio diversos problemas en las simulaciones. En primer lugar, cuando configuramos el transmisor para trabajar estrictamente con el formato csv, la interfaz nos otorgaba un error que no hacía posible si quiera efectuar la transmisión.

Después de numerosas simulaciones pudimos notar que, sobre la señal recibida, la cantidad de datos era diferente a la transmitida originalmente. Dicho análisis nos hizo comprender que había una aparente no correspondencia entre los formatos utilizados a ambos lados del sistema. Con el objetivo de solucionar el inconveniente se decidió trabajar con el formato nativo de los convertidores AD-DA para asegurar una correcta operación. Para ello, en el transmisor se utilizó el formato por defecto que es binario, asumiendo que la interfaz realizaría una correcta adaptación para obtener una conversión digital a analógica correcta. Para implementar lo dicho anteriormente la función « save_sc16q11 » disponible en Matlab fue utilizada.

Sincronización (estimación de canal y offset de fase):

La etapa de sincronización y detección, nos ha requerido la mayor parte del tiempo, como resultado del inconveniente de formato anteriormente mencionado y debido a la falta de conocimientos específicos sobre dicho sistema e implementación.

7- Análisis crítico de los resultados obtenidos como así también, conclusiones del proyecto y perspectivas que surgen del mismo

7.1- Análisis y Conclusiones

Durante el desarrollo de las actividades, tuvimos un mes para interiorizarnos con los medios que eran necesarios, como los softwares, las interfaces y la placa BladeRF. En ese tiempo hemos incorporado diversos conceptos sobre el proceso de puesta en marcha del demostrador, así como también, nuevos métodos de trabajo y organización para alcanzar los objetivos fijados, de una manera más eficiente.

Hemos trabajado con la ayuda de bibliografía digital e impresa, además de la información provista por nuestra tutora y otros ingenieros, quienes tienen suficiente experiencia sobre el tema.

El proyecto ha sido realizado de una manera conjunta entre los dos, con ciertas divisiones que nos permitieron avanzar más rápidamente, aunque hubo otras etapas que hemos llevado a cabo juntos para tener una idea más precisa en esos procesos que no conocíamos con profundidad. Además, realizamos un diagrama FAST, que ha sido muy útil para una mejor organización de las actividades, así como también para un seguimiento ordenado de los procesos a ejecutar.

Después de la división de actividades, la organización y la investigación previa, hemos comenzado con la programación sobre los softwares propuestos, para iniciar, a continuación, una de las partes más importantes de nuestro trabajo: realizar las pruebas con las placas Blade RF, que nos ha permitido interactuar con la realidad y tener una idea más concisa sobre lo sucedido, como así también hemos podido continuar con el desarrollo de nuestro proyecto a través de las modificaciones necesarias para acercarnos al objetivo final.

Luego de todo lo realizado, y a pesar de las dificultades y contratiempos enfrentados, hemos podido arribar al resultado propuesto, para así mostrar la funcionalidad de las cadenas de transmisión y de recepción programadas. Por lo tanto, es posible observar que el conjunto de los métodos y medios utilizados y nombrados anteriormente, el proyecto realizado funciona correctamente.

A continuación se muestra una serie de Figuras [31-32-33] que representan los resultados obtenidos de las simulaciones de los caminos de transmisión como de recepción:

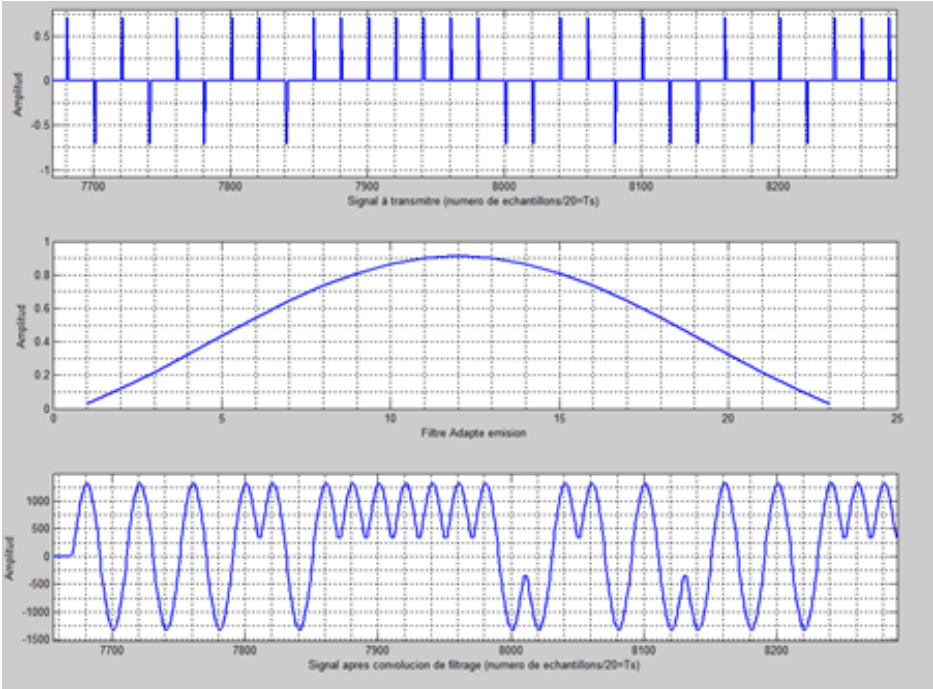

Ilustración 33-Señal sobremuestreada – Ventana del filtro adaptado – Señal filtrada y modulada

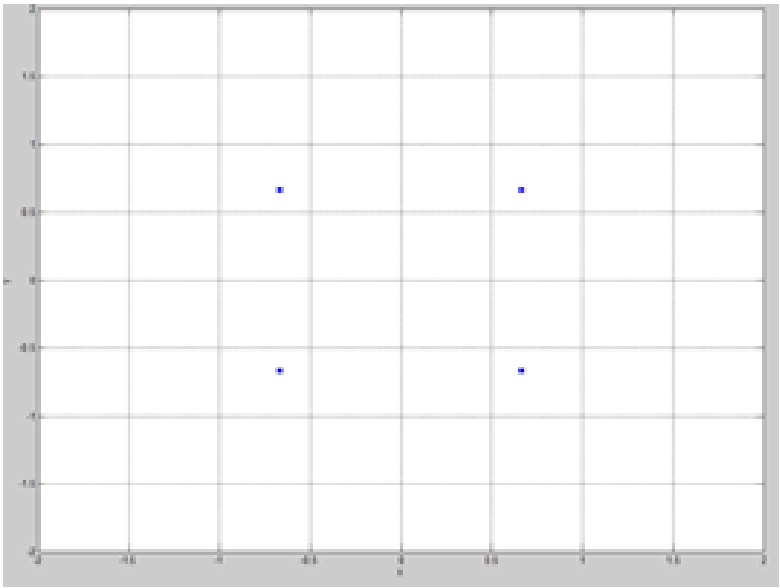

Ilustración 34-Constelación QPSK a transmitir

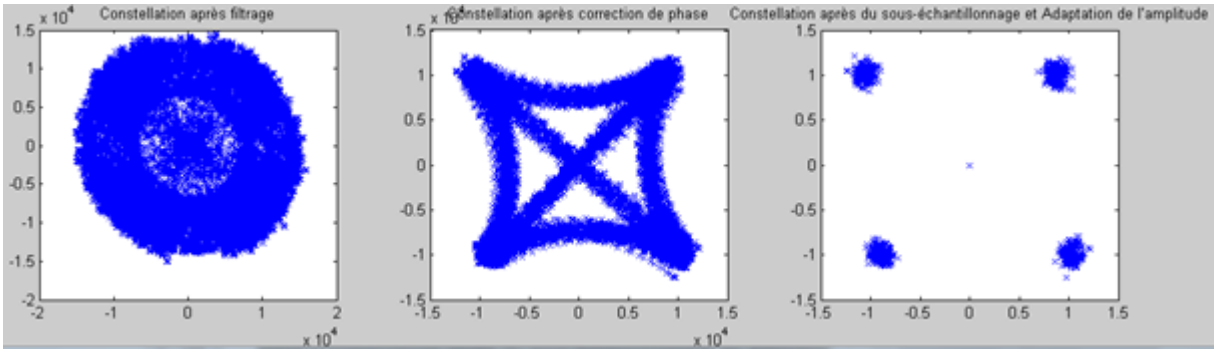

Ilustración 35- Constelación recibida - Constelación con fase corregida - Constelación submuestreada

Con los resultados mostrados, se puede observar que hemos realizado y también comprendido la importancia de cada parte de una cadena de transmisión real, como así, todas las problemáticas implicadas sobre un sistema de comunicación RF.

7.2- Nuestra experiencia personal

La oportunidad de trabajar en un dominio particular, en pleno crecimiento y que ofrece un gran potencial de trabajo, nos ha abierto la posibilidad de transmitir nuestro interés sobre SDR a nuestra universidad de origen en Argentina.

Creemos que habrá aplicaciones interesantes sobre los proyectos actuales de investigación, que son principalmente sobre los UAV, siendo sujetos de interés común en el marco de la Aeronáutica y Telecomunicaciones.

En lo que concierne a nuestra experiencia personal, la misma ha sido muy importante e interesante ya que a través de este proyecto hemos incorporado muchos conocimientos sobre los softwares que no sabíamos utilizar cuando comenzamos y sobre el hardware BladeRf. Además, hemos aprendido a desempeñarnos dentro de un grupo de trabajo y a realizar un proyecto de investigación en un laboratorio. Es necesario remarcar que hemos realizado nuestra práctica en un país distante al nuestro en muchos aspectos, con diferentes costumbres y con una lengua que al comienzo nos ha presentado un ambicioso desafío. Pero eso nos ha hecho más que ayudarnos a enriquecer nuestra capacidad de adaptación y aprendizaje.

A causa de lo anterior, estamos muy agradecidos de haber tenido esta gran oportunidad, ya que todo lo aprendido será muy útil, no solamente por los conocimientos científicos adquiridos sino también por las modalidades de trabajo que hemos podido desarrollar para aplicarlas en empleos futuros, permitiendo mejorar nuestra performance.

8- Bibliografía

Referencias

[1] http://exploit.iemn.univ-lille1.fr
[2] http://www.univ-valenciennes.fr/DOAE/index-doae
[3] https://fr.wikipedia.org/wiki/Radio_logicielle
[4] http://f6ikyradioamateur.pagesperso-orange.fr/pdf/sdrf8ghe.pdf
[5] https://tel.archives-ouvertes.fr/tel-00355352/document
[6] https://www.nuand.com/
[7] http://www.cypress.com/products/ez-usb-fx3-superspeed-usb-30-peripheral-controller
[8] https://fr.wikipedia.org/wiki/MATLAB
[9] https://fr.wikipedia.org/wiki/GNU_Radio
[10] https://fr.wikipedia.org/wiki/Modulation_du_signal
[11] http://www.ief.u-psud.fr/~bournel/PDF_enseignement/Intro-sys-telecom.pdf
[12] http://www.bedwani.ch/electro/ch21/
[13] Leonardo S. Cardoso, Raul Lacerda Neto, Pierre Jallon, Merouane Debbah. SDR4all: a Tool for Making Flexible Radio a Reality. COGnitive Systems with Interactive Sensors 2009, Nov 2009, France. 6 p., 2009.

- Digital Communication Systems Using MATLAB and Simulink- Dennis Silage - 2009- Bookstand Publishing

Sitios de internet:

- nuand BladeRF- https://nuand.com/forums/
- GitHub, Inc- https://github.com/Nuand/bladeRF/
- The MathWorks, Inc.http://fr.mathworks.com/
- http://www.academia.edu/6168741/Modulation_Techniques_MATLAB_Code
- National Instruments Corporation-http://www.ni.com/tutorial/5657/en/(IQ erreurs)
- Wikipedia- https://www.wikipedia.org/
- OsmocomSDR- http://sdr.osmocom.org/
- BladeRF- https://sites.google.com/site/sdrbladerf/home

9- Anexos

A continuación, se encuentra el código de la cadena implementada sobre el software Matlab que corresponde a lo detallado anteriormente y realizado durante nuestro proyecto en Francia.

```
%Modulateur QPSK

clc;
clear all;
close all;


rng default
data = randi([0 1],1, 2048);          %información aleatoire


pre1=ones(1,128);
pre2=zeros(1,128);
pre3=(pre1*-1);
M=20;                                 %échantillons per symbole
span=100;
rolloff=0.2;
data1=[];

pre=[pre2 pre1];
prea=[pre2 pre2];
pre1=pskmod(pre,2,pi);    %preamble BPSK
pre=real(pre1);

figure(1)
stem(data,'linewidth',3), grid on;
title(' Información antes de transmitir ');
axis([ 0 30 0 1.5]);

[datag,MAP] = bin2gray(data,'psk',4);

data_NRZ=2*datag-1;      % Data Represented at NZR form for QPSK
modulation
s_p_data=reshape(data_NRZ,2,length(data_NRZ)/2);  % S/P convertion of
data

                              %%% Modulacion %%%
y=[];
y_in=[];
y_qd=[];

tic
for i=1:length(data_NRZ)/2;
    y1=s_p_data(1,i)*cos(pi/4); % inphase
```

```
    y2=s_p_data(2,i)*sin(pi/4); % Quadrature
    y_in=[y_in y1]; % inphase vector
    y_qd=[y_qd y2]; % quadrature vector
end
toc

test = cast(data_NRZ, 'uint8');

figure(2)
plot(y_in,y_qd,'X'), axis ([-2 2 -2 2]);
title('Constelación QPSK');
xlabel('X');
ylabel('Y');


data2_in=[pre y_in];
data2_in=upsample(data2_in,M);           %Upsample head and data

rrcFilter=rcosdesign(rolloff,span,12);   %Raised cosine FIR filter design
rrcFilter=3*rrcFilter(1,590:612);
txSignal_in=conv(data2_in,rrcFilter,'same'); % Filtrage I

data2_qd=[pre y_qd ];
data2_qd=upsample(data2_qd,M);            %Upsample head and data

rrcFilter=rcosdesign(rolloff,span,12); % Raised cosine FIR filter design
rrcFilter=3*rrcFilter(1,590:612);
txSignal_qd=conv(data2_qd,rrcFilter,'same'); % Filtrage Q

fvtool(rrcFilter,'Analysis','Impulse'); %Filter Impulse response

txSignal_in=(txSignal_in); %Adaptation d'amplitud des symboles (DAC-ADC)
txSignal_qd=(txSignal_qd);

txSignal=[txSignal_in;txSignal_qd];  %Signal a transmetre

txSignalc=[txSignal_in+1j*txSignal_qd];

figure(15);
plot(txSignal','b','linewidth',1);
xlabel('Signal à transmitre');
ylabel('Amplitud');
grid on;

subplot(311);
plot(data2_in,'linewidth',1.5);
grid on;
grid minor;
xlabel('Signal à transmitre (numero de echantillons/20=Ts)');
ylabel('Amplitud');

subplot(312);
plot(rrcFilter,'linewidth',1.5);
grid on;
```

```
grid minor;
xlabel('Filtre Adapte emision');
ylabel('Amplitud');

subplot(313);
plot(txSignal_in,'linewidth',1.5);
grid on;
grid minor;
xlabel('Signal apres convolucion de filtrage (numero de
echantillons/20=Ts)');
ylabel('Amplitud');

txSignal=txSignalc;

txSignalca=txSignal;

save_sc16q11('mybin.bin',txSignalca);  %write the .bin file with message



%Demodulateur QPSK

clc;
clear all;
close all;
rng default

rxFiltSignal=[];
Message=[];
M=20; %échantillons par symbole
span=100;
rolloff=0.2;
Fs=1e6;

pre_1=ones(1,128);
pre_0=zeros(1,128);
pre1=[pre_1 pre_0];
pre=upsample(pre1,M);
pre=pre*2047;

RxSignal = csvread('test_correcto.csv'); %signal reçu
Rx_sig=RxSignal;

Rx_sig1= Rx_sig(:,1);
Rx_sig2= Rx_sig(:,2);

figure(1)
plot(Rx_sig2) % axis ([-2 2 -2 2]);
title('Information reçu ');
xlabel('X');
ylabel('Y');

figure(1)
```

```
subplot(2,1,1)
plot(Rx_sig1);
subplot(2,1,2)
plot(Rx_sig2)

                        %%% Filtre de réception %%%

rrcFilter=rcosdesign(rolloff,span,12); % Raised cosine FIR filter design
rrcFilter=4*rrcFilter(1,590:612);
rxFiltSignal_in = conv(Rx_sig1,rrcFilter,'same'); % Filtrage I

eyediagram(rxFiltSignal_in,200);

rrcFilter=rcosdesign(rolloff,span,12); % Raised cosine FIR filter design
rrcFilter=4* rrcFilter(1,590:612);
rxFiltSignal_qd = conv(Rx_sig2,rrcFilter,'same'); % Filtrage Q

                            %%% Corrélation %%%

[acor,lag] = xcorr(rxFiltSignal_in,pre);
[~,I] = max(abs(acor));
lagDiff = lag(I);
timeDiff = lagDiff/Fs;

figure (3)
plot(acor), grid on;

rxFiltSignal_in=rxFiltSignal_in';
rxFiltSignal_qd=rxFiltSignal_qd';

                           %%% Synchronisation %%%

comple=rxFiltSignal_in+1j*rxFiltSignal_qd;

x1=0;
s1=0;

for i=lagDiff+1:1:lagDiff+2560
    x=(angle(comple(i+1))- angle(comple(i)));
    x1=x1+x;
end

h=x1/2560;

tic
for i=lagDiff+2561:1:lagDiff+23039
    a=comple(1,i);
    b=exp(1j*h*(i-lagDiff));%correction d'angle
    s=a/b;
    s1=[s1 s];
end
toc

z=real(s1);
```

```
w=imag(s1);

rxFiltSignal_in1p= s1(:).'*exp(1j*(pi/4)); %correction de phase

                        %%% Sous Échantillonnage %%%

rxFiltSignal_in1 = downsample(rxFiltSignal_in1p,M);

                      %%% Adaptation de l'amplitude %%%

rxFiltSignal_in1=rxFiltSignal_in1/1e4;

                             %%% Maping inverse %%

a_re= real(rxFiltSignal_in1p); % inphase
b_im= imag(rxFiltSignal_in1p); %Quadrature

for i=1:1:length(a_re)

                                    %detector real
    Z_in=a_re(i)/cos(pi/4);

    if (Z_in>=0)
        Z_in=1;
    else
        Z_in=0;
    end
                                  % detector imag
      Z_qd=b_im(i)/sin(pi/4);

     if (Z_qd>=0)
        Z_qd=1;
    else
        Z_qd=0;
     end

        Message=[Message Z_in  Z_qd];
end

figure(8)
subplot(1,3,1)
plot(comple(lagDiff+2561:1:lagDiff+23039),'X');
%axis([-1.5e4 1.5e4 -1.5e4 1.5e4]);
title('Constellation après filtrage');
subplot(1,3,2)
plot(rxFiltSignal_in1p,'X')
title('Constellation après correction de phase');
subplot(1,3,3)
plot(rxFiltSignal_in1,'X') % axis ([-2 2 -2 2]);
title('Constellation après du sous-échantillonnage et Adaptation de
l'amplitude');

figure(9)
stem(Message,'linewidth',3)
```

```
title('Données demodulées');
axis([0 30 0 1.5]), grid on;
fvtool(Message,'Analysis','Impulse')
```