ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

“ANÁLISIS Y EVALUACIÓN DE LA PLATAFORMA DE RADIO DEFINIDO POR SOFTWARE DE BAJO COSTO: ADALM-PLUTO SDR ACTIVE LEARNING MODULE”

EXAMEN DE GRADO (Complexivo)

Previo a la obtención del título de

MAGISTER EN TELECOMUNICACIONES

Presentado por

CÉSAR AUGUSTO NOBOA TERÁN

GUAYAQUIL – ECUADOR

2020

AGRADECIMIENTOS

A Dios por haberme permitido llegar a vivir este momento.

A mi adorada familia, por haberme dado las fuerzas y amor para culminar esta meta.

A la ESPOL por el orgullo haber sido formado en sus aulas.

A mis profesores por haber recibido de ellos su conocimiento.

A mis amigos y compañeros de tantos y sobrehumanos esfuerzos académicos, y en especial a aquellos que me impulsaron a llegar a la meta y cumplir este objetivo:
Ernesto, José Luis, Janeth, Amalia, Tanya, Grace, José Stalyn.

DEDICATORIA

Este trabajo se lo dedico a mi adorada familia. Y con especial dedicatoria a la persona que me dio la vida: Dora Terán, mi madre querida.

TRIBUNAL DE EVALUACIÓN

MSc. Alfredo Nuñez

PhD. María Antonieta Alvarez

DECLARACIÓN EXPRESA

"La responsabilidad y la autoría del contenido de este Trabajo de Titulación, me corresponde exclusivamente; y doy mi consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

CÉSAR AUGUSTO NOBOA TERÁN

RESUMEN

En este trabajo se investigó la tecnología de radio definido por software la cual permite experimentar y verificar de manera práctica los conceptos teóricos de telecomunicaciones vistos en aulas de clase. Partiendo de un marco teórico sobre esta tecnología, se analizó la plataforma de Radio Definido por Software Adalm-Pluto, la cual corresponde a un segmento de dispositivos de bajo costo. Se describió las características técnicas, operativas y además sus interfaces de programación compatibles con el dispositivo.

Se implementó un sistema trasmisor y receptor QPSK completo, utilizando dos dispositivos Adalm-Pluto, una computadora y software Matlab y Simulink, se experimentó y documentó este sistema. Luego se hizo una comparativa con otros dispositivos del mercado y se realizó una evaluación comparativa de los mismos.

Descriptores: Radio definido por software, Adalm-Pluto, Matlab, Simulink.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. ANTECEDENTES Y DESCRIPCIÓN DEL PROBLEMA

La teoría de las telecomunicaciones tiene muchos conceptos que en su mayoría son abstractos, basados en fórmulas matemáticas complejas, por lo cual una parte importante del aprendizaje lo constituyen los experimentos en laboratorio, mediante los cuales podemos obtener una mayor percepción y conocimiento de la teoría y fundamentos básicos de las telecomunicaciones. No obstante, los experimentos de laboratorio necesitan de equipo experimental costoso, mientras las universidades adolecen de falta de recursos para equipamiento [1].

Además, bajo la coyuntura de confinamiento obligada por la pandemia mundial de Covid-19, surgen nuevas metodologías de enseñanza online, que si bien es cierto resuelven la enseñanza de la teoría, no reemplazan la experimentación realizada en los laboratorios.

Ante esta situación, se plantean escenarios accesibles al estudiante, mediante el uso de plataformas de Radio Definido por Software (SDR por sus siglas en inglés) de bajo costo como RTL-SDR, Adalm-Pluto, HackRF One, entre otras plataformas, apoyadas con herramientas informáticas de modelaje como MATLAB y GNU Radio. Mediante su uso, el estudiante puede llevar a la práctica y comprobar conceptos de comunicaciones digitales, de tratamiento digital de señales, antenas, propagación de señales, así como de electrónica de comunicaciones. Para lograr esto necesita un hardware SDR de bajo costo que le permita recibir o trasmitir una señal y un computador con el software adecuado para controlar y configurar dicho hardware [2].

Sin embargo, existe la barrera de la falta de documentación completa al respecto, debido a que son dispositivos de relativamente reciente aparición, en un mercado dominado por plataformas ya consolidadas, como, por ejemplo: los dispositivos USRP de National Instruments.

1.1. Descripción de la Propuesta:

El dispositivo Adalm-Pluto fabricado por Analog Devices, es una plataforma de radio definido por software autónomo y portable que permite la trasmisión y recepción de señales de radiofrecuencia, a un costo accesible. Lo que permite poner en práctica las etapas completas de radio, diferenciándolo de dispositivos como el RTL-SDR que solo permiten la recepción de señales.

En este trabajo en una primera instancia se analizará las características internas y operativas del Adalm-Pluto, sus formas de uso y posibles aplicaciones.

En una segunda instancia, se describirán los ambientes e interfaces de programación con los que se puede interactuar con el Adalm-Pluto, el mismo que provee librerías Libiio, para implementación desde C, C++, C# y Python API. Además, se describirá la compatibilidad con GNU Radio, Matlab y Simulink [3].

Luego, a través del soporte ofrecido por Matlab y Simulink se implementará un entorno de transmisión-recepción real basado en QPSK, para el cual el transmisor generará un mensaje de texto ASCII, luego convertirá los caracteres en bits y anteponiendo un código Barker para la sincronización de trama del receptor. Estos datos se modularán utilizando QPSK y se filtrarán con un filtro de coseno alzado. Los símbolos QPSK filtrados se trasmitirán al aire utilizando el sistema transmisor del dispositivo ADALM-PLUTO.

Con un segundo dispositivo ADALM-PLUTO se recibirán datos de la transmisión por aire y serán procesadas por el sistema receptor QPSK.

Al final se procederá a evaluar los resultados obtenidos, en función de calidad de trasmisión, nivel de error, facilidad de implementación y se los comparará teóricamente con otros dispositivos RDS de bajo costo como el RTL-SDR y el Hack One RF.

1.2. Objetivos

Objetivo general:

- Analizar y evaluar plataforma de Radio Definido por Software de bajo costo: Adalm-Pluto SDR Active Learning Module.

Objetivos específicos:

- Analizar las características del dispositivo Adalm-Pluto e identificación de sus capacidades, formas de uso y limitaciones del dispositivo.
- Describir las interfaces de programación del dispositivo para su uso con GNU Radio, Matlab y Simulink.
- Implementar una trasmisión y recepción de un mensaje ASCII con modulación QPSK mediante el uso de dos dispositivos Adalm-Pluto, Matlab y Simulink.
- Evaluar los resultados, las prestaciones del dispositivo y comparación teórica con otros dispositivos RDS de bajo costo.

1.3. Resultados Esperados

Los resultados a obtener de este trabajo son los siguientes;

- Información detallada de las características, capacidades, limitaciones y modos de uso del dispositivo Adalm-Pluto.
- Información de las diferentes interfaces de programación disponibles para el dispositivo, que servirán de referencia para futuras implementaciones.
- Documentación de la implementación de un caso práctico de trasmisión y recepción QPSK utilizando el dispositivo Adalm-Pluto.
- Información evaluativa y comparativa del dispositivo con respecto a otros dispositivos RDS de bajo costo.

1.4. Marco teórico

El concepto de Radio definida por software (o SDR por sus siglas en inglés) hace referencia a sistemas de radio en los que casi todas las funciones asociadas con la capa física se implementan en software que utilizando algoritmos de procesamiento de señales digitales (DSP) y a través de una

pequeña interfaz de hardware; solo una antena y un muestreador de señales de alta velocidad capaz de capturar y digitalizar una amplia banda de frecuencias de radio. Cualquier demodulación, sincronización, decodificación o descifrado necesario para recuperar la información contenida en la señal recibida se debería realizar en un software que se ejecuta en un dispositivo de procesamiento dedicado y con alta prestación [4]. Es decir, todas las funciones de parametrización del sistema deben ser configuradas por software.

La tecnología SDR apareció en 1998 con el proyecto llamado SpeakEasy I y II en los EE. UU., el cual utilizó un procesador de Texas Instruments TMS320C40 de 40 MHz incluyendo por primera vez circuitos con matriz de puertas lógicas programable (FPGA), siendo su uso restringido al ámbito militar. Conforme evolucionó la tecnología, con circuitos de compuertas lógicas y procesadores digitales más potentes, generó nuevas alternativas al mercado de comunicaciones comercial, experimental y educativo [5].

A partir de esta tecnología, se origina el Wireless Innovation Forum (antes conocido como SDR Forum), para promover estándares e innovaciones en el desarrollo de aplicaciones SDR de uso civil y militar. En el año 2001 surge el proyecto GNU Radio, el cual provee un ambiente de desarrollo que usa software de código abierto con bloques para procesamiento de radio y análisis de señales, con compatibilidad con hardware externo. Con lo que surge la oportunidad para primeros desarrollos en código abierto de dispositivos SDR de carácter experimental, permitiendo realizar no solo simulaciones, sino también interacción con equipamiento de comunicaciones reales.

En 2005 la compañía Ettus Research, subsidiaria de National Instruments produce un dispositivo llamado USRP, creando una plataforma de SDR de bajo coste, impulsando el auge de esta tecnología [6].

En el mercado actual, consultado en septiembre del año 2020, tenemos que un USRP Ettus inicia su precio desde los $925, siendo de tipo experimental y vendidos como módulos no ensamblados. National Instruments provee sus dispositivos USRP, ya ensamblados y con soporte, pero sus precios inician en $1285, y algunos alcanzan decenas de miles de dólares, si bien es cierto, representan valores cada vez menores, aún representan una inversión económica alta para estudiantes y profesionales de países en vías de desarrollo.

1.5. Partes de un SDR

Un SDR está compuesto por tres bloques funcionales: sección de RF, sección de IF y sección banda base, tal como se muestra en la Figura 1. 1. La parte de RF e IF se implementan en hardware mientras que la sección de banda base en software.

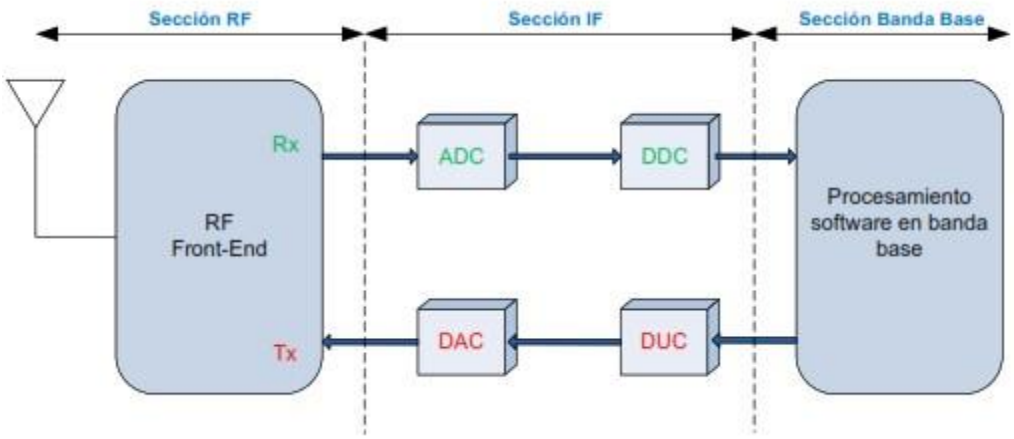

Figura 1. 1 Partes de un radio definido por software

La sección de RF recibe o trasmite las señales de radiofrecuencia, en el caso de recepción, es encargada de adecuar las señales y para llevarlas a un bloque de frecuencia intermedia IF. En el caso de trasmisión, se encarga de modular las señales de IF. Conforme avanzan la tecnología, se puede hablar de componentes ZERO-IF, donde no existe procesamiento en frecuencia intermedia. La sección de IF pasa la señal hacia la banda base para

digitalizarla en la recepción o ejecuta la conversión digital analógica en caso de trasmisión.

Existen los módulos convertidores analógico digitales ADC y convertidores digital analógicos DAC encargados de este proceso. Para aumentar o bajar la tasa de muestreo tenemos los módulos DUC y DDC es decir módulos para interpolación y diezmado. La sección de banda base se encarga del procesamiento en banda base de la señal, es decir aquí se realiza la modulación o demodulación, codificación o decodificación, análisis espectral de la señal, ayudándose de software especializado [2].

Utilizando estos dispositivos es posible desarrollar algoritmos de modulación y demodulación de señales a una velocidad de muestreo específica, todo esto controlado en un ambiente de desarrollo, normalmente una computadora host, donde la depuración y la visualización son mucho más fáciles. Se necesita entonces un software eficiente para desarrollar y probar las funciones de transmisión de datos y procesamiento de señales digitales en un sistema de comunicaciones inalámbricas. Un entorno de software que cumple con este requisito es MATLAB de MathWorks. MATLAB es un entorno informático técnico y lenguaje de programación, que permite un desarrollo fácil de usar y excelentes mecanismos de visualización. Un producto adicional, Simulink y el bloque Communications Systems Toolbox, agrega algoritmos de capa física, modelos de canal, modelos de referencia y conectividad al hardware SDR para transmitir y recibir señales en vivo.

La tecnología SDR es impulsada por la constante búsqueda de nuevos servicios y rentabilidad de los equipos, ya que esta tecnología permite ahorrar costos, y con ella se favorecen la facilidad de adaptabilidad y de configuración de los equipos, factores claves en el mercado de telecomunicaciones.

CAPÍTULO 2

2. Dispositivo Adalm-Pluto SDR

El dispositivo Adalm-Pluto es una plataforma de bajo costo, orientado a la educación y experimentación acorde a la definición de su fabricante Analog Devices. No obstante, proporciona capacidad suficiente para abordar una amplia gama de aplicaciones fuera del ámbito educativo, ya que contiene en su interior componentes de comunicaciones configurables por software.

Figura 2. 1 Dispositivo Adalm-Pluto SDR

Las especificaciones de este dispositivo están descritas en la Tabla 2. 1.

Especificación	Descripción	Valor típico
Power	DC Input (USB)	4.5 V a 5.5 V
Reloj y muestreo	Tasa de Muestreo ADC y DAC	65.2 KSPS a 61.44 MSPS
	Resolución de ADC y DAC	12 bits
	Precisión de frecuencia	±25 ppm
Radiofrecuencia	Rango cobertura	325 MHz a 3800 MHz

	Potencia Salida Tx	7 dBm
	Figura de ruido Rx	<3.5 dB
	Rx , Tx Modulation Accuracy (EVM)	–34 dB (2%)
	RF Shielding	No
Digital	Circuito RF	Tranceiver AD9363-- Highly Integrated RF Agile Transceiver de Analog Devices
	FPGA	FPGA Xilinx® Zynq Z-7010
	Núcleo	Single ARM Cortex® -A9 @ 667 MHz
	USB	2.0
	FPGA Logic Cells	28k
	DSP Slices	80
	DDR3L	512 MB
	QSPI Flash	256 Mb (32 MB)
Especificaciones físicas	Dimensiones	117 mm × 79 mm × 24 mm
	Peso	114 g
	Temperatura	10°C a 40°C

Tabla 2. 1 Especificaciones del dispositivo Adalm-Pluto

El circuito Tranceiver AD9363, que consta internamente de un trasmisor y un receptor, con modo de operación semidúplex o dúplex completo. El dispositivo está concebido como un módulo de aprendizaje de radiofrecuencia autónomo y portátil, teniendo suporte y compatibilidad con Matlab y Simulink, siendo

importante destacar que dichas librerías de programación consiguen exceder sus especificaciones teóricas en el rango de cobertura, obviando un bloqueo interno, ampliando su rango de cobertura desde 70 MHz a 6 GHz.

Tiene compatibilidad con bloques para GNU radio y tiene además una interface de programación de tipo software abierto, llamada Libiio, la cual permite la configuración en lenguajes como C, C ++, C # y Python.

Posee una interfaz alimentada por USB 2.0 con conector Micro-USB 2.0, con soporte para Windows y Linux. El dispositivo tiene un almacenamiento interno de 512 MB. Este espació de memoria, puede ser accedido por el puerto USB primario a través de una computadora externa y se puede ser utilizado para colocar programas de control, cuando el dispositivo funciona de manera autónoma, es decir alimentado a través del segundo puerto USB por batería externa y arrancando su operación desde su sistema Linux.

A nivel macro este dispositivo está compuesto de dos grandes bloques acorde a la Figura 2. 2. Se puede apreciar dos bloques principales: el circuito AD9363 encargado de la trasmisión o recepción y el FPGA Xilinx Zynq, encargado del control del dispositivo.

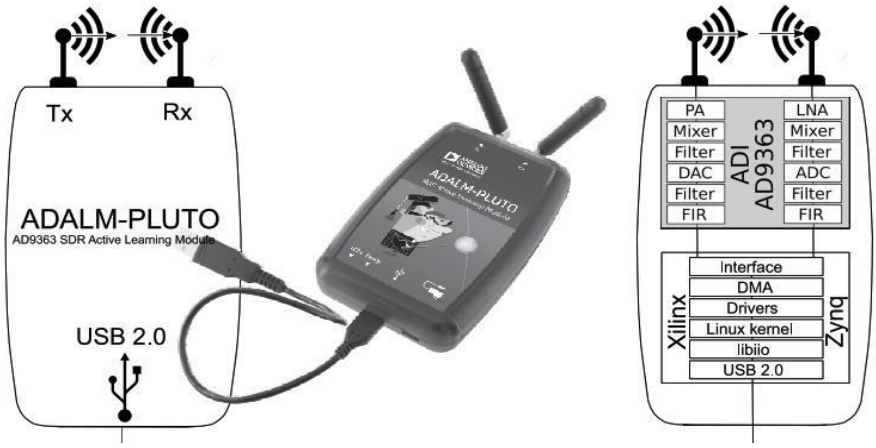

Figura 2. 2 Bloques del dispositivo Adalm-Pluto

2.1. Detalles del circuito AD9363

El circuito AD9363 es responsable de capturar y digitalizar los datos de RF. Este circuito proporciona amplificación, traslación y mezcla de frecuencias, conversión digital y filtrado de señales transmitidas y filtrado de señales recibidas. Tiene tres secciones principales: una sección de RF analógica, una sección de banda base analógica y una para procesamiento de señales recibidas o señales por transmitir. La Figura 2. 3 muestra un diagrama de este circuito.

En la parte delantera del AD9363 hay un amplificador de bajo ruido (LNA) que proporciona una ganancia analógica que es parte del componente de control automático de ganancia (AGC) del receptor. Después del LNA se encuentra el mezclador, que es responsable de los cambios de frecuencia.

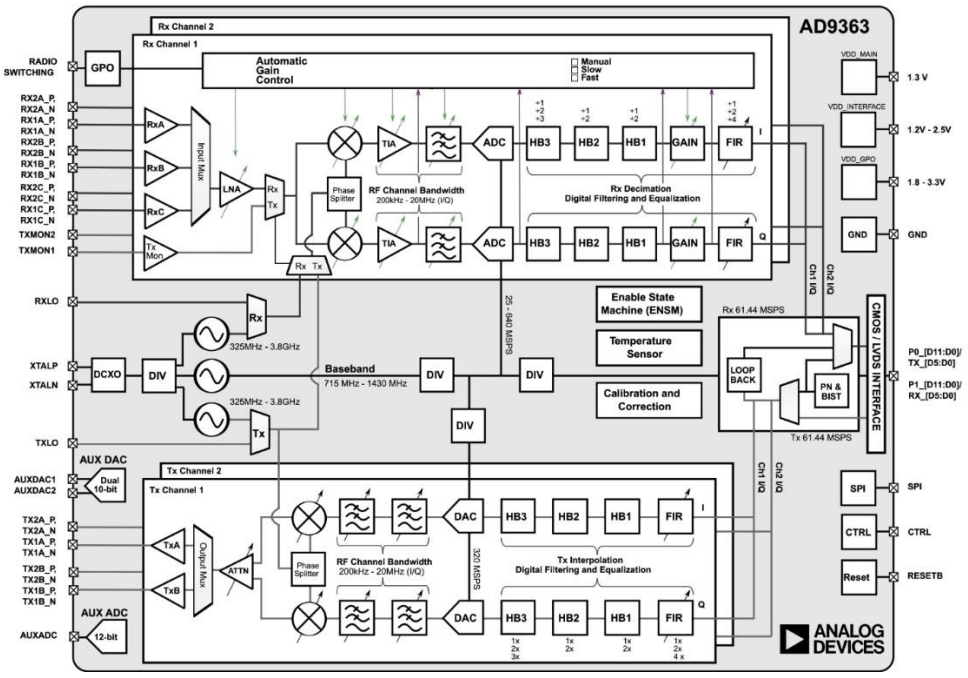

Figura 2. 3 Diagrama del circuito AD9363

Antes de este proceso de mezcla, la señal se divide y se alimenta a lo largo de dos caminos diferentes pero idénticos, para los componentes en fase y cuadratura de la señal mediante una simple rotación de fase del reloj del

mezclador. Efectivamente, esto duplica el ancho de banda efectivo del receptor ya que las señales en fase y en cuadratura son ortogonales.

Después de mezclar, la señal se filtra para eliminar los efectos de aliasing de la señal mezclada y reducir la interferencia y el ruido fuera de banda. El amplificador de transimpedancia combinado (TIA) y el filtro analógico se configuran juntos para mantener el ancho de banda analógico. Al utilizar un ADC de muy alta velocidad, la señal de recepción se puede digitalizar a 12 bits a la frecuencia de muestreo configurada deseada. Por lo tanto, la mejor resolución de la señal se logra mediante un gran sobremuestreo de la señal de entrada y luego seguido de varias etapas de diezmado etapa de frecuencia intermedia (IF), que puede oscilar entre 200 kHz y 20 MHz. El circuito TIA actúa como un filtro unipolar y el filtro programable analógico es un Butterworth de tercer orden. La etapa final del AD9363 es la etapa de conversión y diezmado digital. Aquí, el ADC normalmente se ejecutará a una velocidad mucho más alta que el ancho de banda de recepción deseado, pero el ADC en sí mismo no proporcionará los 12 bits definidos en las especificaciones. Los bits adicionales se obtienen en las etapas del filtro de media banda (HBF), lo que permitirá el crecimiento de bits [7].

El AD9363 también proporciona sistemas de auto-calibración y AGC para mantener un alto nivel de rendimiento bajo diferentes temperaturas y condiciones de señal de entrada. Además, el dispositivo incluye varios modos de prueba que permiten a los diseñadores de sistemas insertar tonos de prueba y crear modos de bucle invertido interno para depurar sus diseños durante la creación de prototipos y optimizar su configuración de radio para una aplicación específica. Acorde a la hoja de especificaciones, a nivel de trasmisión y recepción, el circuito AD9363 tiene la capacidad teórica para trabajar con señales de diferentes fuentes, lo que permitiría que el dispositivo se utilice en sistemas de múltiples entradas y múltiples salidas (MIMO), no obstante esta capacidad está bloqueada por firmware en el Adalm-Pluto, el cual solo tiene habilitado un solo canal de trasmisión y un solo canal de recepción [8].

2.2. Detalles del FPGA Xilinx Zynq Z-7010

Una vez que los datos se digitalizan, se pasan al sistema Xilinx Zynq, el cual es un FPGA que proporciona un procesamiento basado en ARM Cortex-A9 con lógica programable, el Zynq es utilizado en Adalm-Pluto como controlador principal. El núcleo de este circuito está basado en Linux, permitiendo utilizar los controladores de dispositivo Linux IIO ya existentes, y utilizados en otros dispositivos, lo cual permite controlar aspectos del dispositivo tales como frecuencias de muestreo, la configuración de filtros FIR, la configuración del oscilador local LO o filtros de diezmado adicionales. Una vez que los datos de señales en fase y cuadratura (IQ) se receptan de circuito AD9363, son transferidos a la estructura FPGA, el núcleo AXI DMAC las guarda en memoria buffer del Adalm-Pluto.

2.3. Modos de operación del dispositivo

La arquitectura utilizada por el Adalm-Pluto es llamada Industrial Input Output (IIO) el cual corre dentro de un núcleo de Linux, y brinda soporte a dispositivos tales como convertidores ADC o DAC, siendo un estándar de código abierto adoptado por muchos fabricantes diferentes para proporcionar una API común. En la Figura 2. 4 se muestra la arquitectura de este sistema

• El controlador IIO se ejecuta dentro del núcleo de Linux, en este caso en ARM en Pluto SDR.

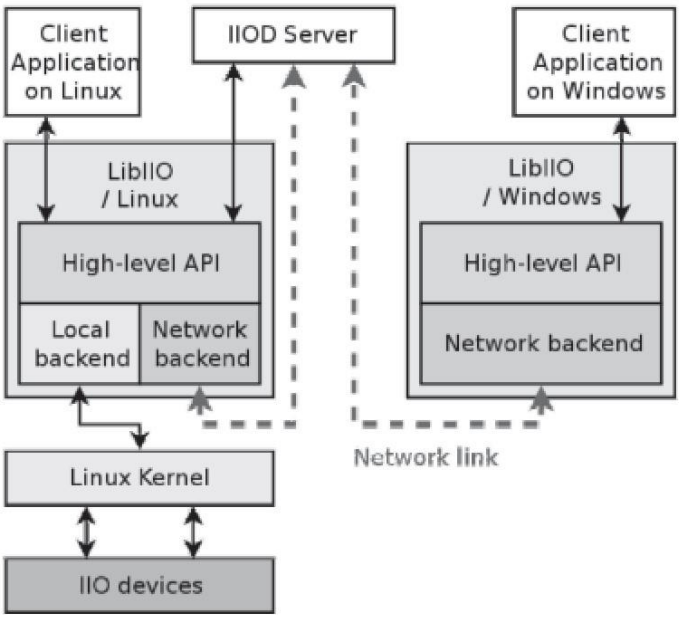

Figura 2. 4 Arquitectura de interfaces del Adalm-Pluto

- Libiio, es la biblioteca para acceder a dispositivos IIO locales y remotos, implementadas para correr en el dispositivo ARM, o en el host. Es de acotar que el dispositivo es totalmente autosuficiente, necesita ser alimentado por el puesto USB 2, y puede correr programas en ambiente Linux.

- IIOD, es un proceso tipo agente, responsable de permitir la conexión remota a los clientes IIO externos hacia el Adalm-Pluto.

Convenientemente, la interface API de Libiio que accede al transceptor trabaja de la misma manera en el dispositivo Adalm-Pluto ARM como host o corriendo en una computadora PC como host, con los controladores Libiio instalados. Por lo tanto, el código se puede implementar en la máquina host conectada a Pluto SDR y luego desplegado en el ARM con el mismo código [7].

2.4. Interfaces de programación

Matlab puede ser usado como una cliente IIO de tipo multi-plataforma hacia el Adalm-Pluto, incluyendo el objeto de comunicación para recepción y trasmisión. En la Tabla 2. 2 se muestra la estructura de objetos para recepción Comm.SDRRxPluto.

Atributo	Descripción
RadioID	Atributo identificador del dispositivo: USB: # donde # es un número asociado de USB. ip: <ipddress> si está conectado por Ethernet.
CenterFrequency	Define la frecuencia central de operación en Hz. Las frecuencias de trasmisión y recepción pueden ser diferentes.
BasebandSampleRate	Define la frecuencia de muestreo de la Fase/Quadratura. Tener en cuenta que solo hay un oscilador local, por lo que esta frecuencia afecta al convertidor ADC y al convertidor DAC.
GainSource	Tiene tres modos: Manual: El atributo Ganancia se puede configurar manualmente. AGC Slow Attack y AGC Fast Attack: mediante esta configuración los valores de ganancia se autoregulan, acorde al nivel de potencia recibida en el receptor (RSSI).
ChannelMapping	Debe ser establecido en 1, ya que solo está habilitada una sola antena de trasmisión y una sola antena de recepción.
OutputDataType	Determina el formato de datos que se proporcionan el objeto. 8 o 16 bits.
SamplesPerFrame	Determina el número de muestras en el búfer o marco que se pasa a MATLAB desde iiod. Es decir, el tamaño del vector proporcionado.

Tabla 2. 2 Estructura de objeto de recepción Adalm-Pluto en Matlab

El Objeto para trasmisión Comm.SDRTxPluto usado para trasmisión, utiliza los mismos atributos del objeto de recepción, a excepción de los atributos:

GainSource, SamplesPerFrame, and OutputDataType, los cuales solo existen en el contexto de recepción.

2.5. Interface con Simulink

El paquete de soporte de Matlab y Simulink llamado Communications Toolbox Support Package for Analog Devices ADALM-Pluto Radio, está disponible en el sitio web de Mathwoks y le permite utilizar el dispositivo para crear prototipos, verificar y probar sistemas inalámbricos prácticos.

Luego de instalado es posible utilizar los siguientes bloques en Simulink mostrados en la Figura 2. 5.

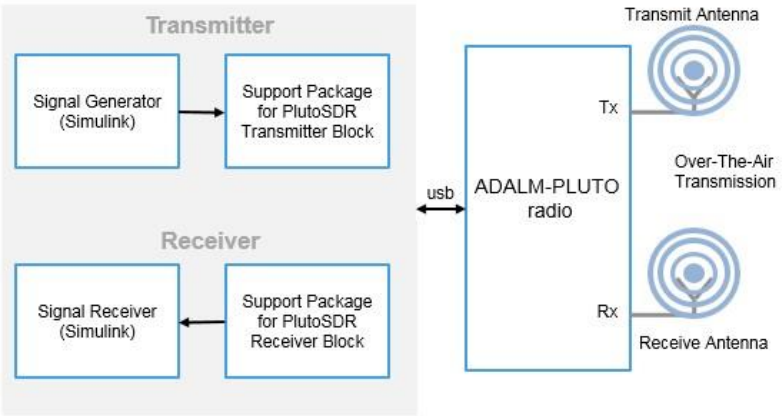

Figura 2. 5 Diagrama de bloques de componentes para Simulink

2.6. Bloque receptor de Adalm-Pluto para Simulink

El bloque receptor en Simulink se llama Adalm-Pluto radio receiver y se muestra en la Figura 2. 6.

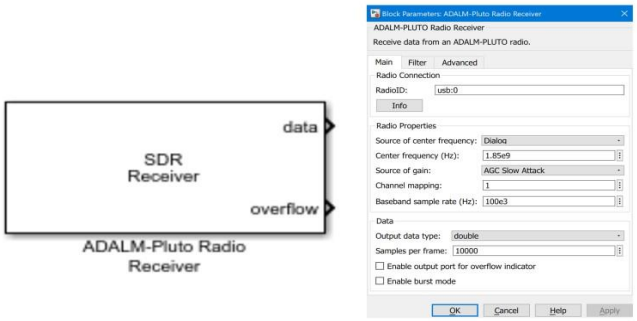

Figura 2. 6 Bloque Adalm-Pluto Radio receiver

A continuación, se muestra la Tabla 2. 3, la cual describe las entradas , salidas y parámetros del bloque receptor del dispositivo en Simulink.

Puertos de Entrada del bloque receptor		
Nombre	Funcionalidad	Observaciones
Center frequency	Ajuste de la frecuencia central de RF en Hz	Escalar de 70.0e6a 6.0e9.
`Gain`	Ajuste de ganancia del receptor en dB	Escalar de -4a 71. La configuración de ganancia mínima y máxima aceptable depende de la frecuencia central. Una combinación incompatible de ganancia y frecuencia central devuelve un error.
Puertos de Salida		
Data	Vector de datos recibidos	Datos recibidos, devueltos como un vector de valores complejos. El rango de valores depende del tipo de datos de la salida. int16- Los datos de salida constan de valores enteros de 16 bits con signo en el rango [–2048, 2047]. El chip RF AD936X tiene un ADC de 12 bits. Los datos de 12 bits del ADC se almacenan en los 12 bits inferiores del valor de salida y se extienden por signo a 16 bits.
Overflow	Indicador lógico de muestras perdidas	Indicador de muestras perdidas, devuelto como lógico (1= Muestras perdidas/0= No se perdieron muestras)
Parámetros		

Source of center frequency	Fuente de frecuencia central	Dialog: Se especifica el valor de Frecuencia central por parámetro fijo en el bloque. Input Port: Se especifica el valor de Frecuencia central mediante una puerta de entrada al bloque.
Source of gain	Fuente de ganancia	AGC Slow Attack: Para señales con señales cuya potencia varía lentamente con el tiempo. AGC Fast Attack: Para señales con un nivel de potencia que cambia rápidamente. Dialog: Ganancia fija especificada en el parámetro Gain (dB). Input Port: Ganancia especificada utilizando una entrada que se agrega al bloque del receptor de Pluto .
Baseband sample rate	Frecuencia de muestreo de banda base (Hz.)	Escalar de 65105 a 61.44e6.
Samples per frame	Número de muestras por frame	Especificado como un número entero positivo par de 2 a 16.777.216. El uso de valores inferiores a 3660 puede producir un rendimiento deficiente. Cuando se transmite al host, el uso de marcos de mayor tamaño puede brindar un rendimiento más eficiente.

Tabla 2. 3 Entradas, salidas y parámetros del objeto receptor en Simulink

El dispositivo Adalm-Pluto tiene un solo canal para recibir datos y enviarlos al bloque Pluto Receiver. El bloque emite una señal de vector de columna de longitud fija.

Si las características de ganancia o ancho de banda del filtro predeterminado no satisfacen los requisitos de la aplicación, es posible utilizar configurador de filtro que cumpla con los requisitos específicos, tal como se muestra en la Figura 2. 7.

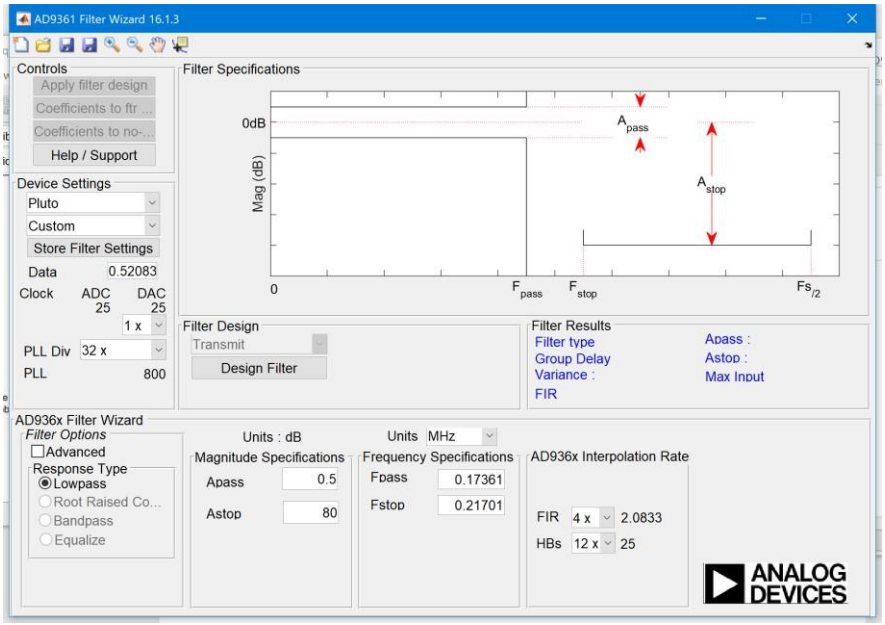

Figura 2. 7 Asistente para configuración de filtro

Al utilizar el asistente de filtro, se permite diseñar un filtro personalizado para el chip de RF AD9363 de Analog Devices basado en el parámetro de frecuencia de muestreo de banda base (Hz) .Se puede ajustar y optimizar la configuración para calcular los filtros analógicos, los filtros de interpolación , diezmado y los coeficientes FIR.

2.7. Bloque trasmisor de Adalm-Pluto para Simulink

El bloque trasmisor Adalm-Pluto se muestra en la Figura 2. 8.

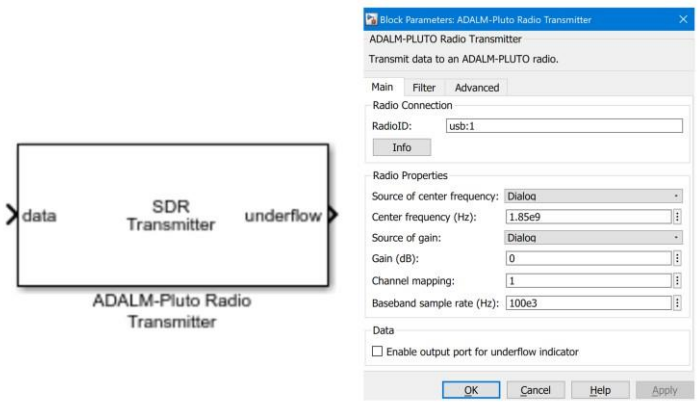

Figura 2. 8 Bloque Adalm-Pluto Radio Trasmitter

En la Tabla 2. 4 se muestran las entradas y salidas del bloque trasmisor en Simulink.Tabla 2. 4 Entradas y salidas del bloque trasmisor de Simulink

Puertos de Entrada		
Data	Datos para transmitir	Vector de columna con un número par de elementos de 2 a 16.777.216. Los datos de transmisión deben ser complejos.
Center frequency	Frecuencia central en Hz	Escalar de 70.0e6a 6.0e9.
Gain (dB)	Ajuste de ganancia del trasmisor en dB	Escalar de -4a 71. La configuración de ganancia mínima y máxima aceptable depende de la frecuencia central. Una combinación incompatible de ganancia y frecuencia central devuelve error.
Puertos de salida		
Overflow	Indicador lógico de muestras perdidas	Indicador de muestras perdidas, devuelto como lógico (1= Muestras perdidas/0= No se perdieron muestras).
Parámetros		
Source of center frequency	Fuente de frecuencia central	Dialog: Se especifica el valor por parámetro fijo en el bloque. Input Port: Se especifica el valor mediante una puerta de entrada al bloque.
Source of gain	Fuente de ganancia	AGC Slow Attack: Para señales cuya potencia varía lentamente con el tiempo. AGC Fast Attack: Para señales con un nivel de potencia que cambia rápidamente. Dialog: Ganancia fija especificada. Input Port: Ganancia especificada utilizando una entrada que se agrega al bloque.

Tabla 2. 4 Entradas y salidas del bloque trasmisor de Simulink

2.8. Interface con GNU Radio.

El fabricante Analog Devices ofrece soporte para uso de sus dispositivos con GNU Radio, a través del módulo Gr-iio, el cual está disponible para bajar en el sitio web destinado a educación y soporte de la compañía.

Solo está soportada la compatibilidad con la versión GNU Radio instalada sobre sistemas operativos Linux, por lo que no es posible usar el dispositivo con GNU Radio para Windows 10.

En un ambiente Linux, para instalar el paquete libiio, se debe realizar los siguientes pasos, acorde a la Figura 2. 9.

```
git clone https://github.com/analogdevicesinc/libiio.git
cd libiio
cmake .
make
sudo make install
cd ..
```

Figura 2. 9 Instalación de librerías Libiio en Linux

Aunque existe actualmente la versión GNU Radio 3.8, se recomienda usar la versión GNU Radio3.7, para garantizar la compatibilidad de las librerías del paquete Gr-iio, lo cual puede ser realizado con los siguientes comandos mostrados en la Figura 2. 10.

```
git clone https://github.com/analogdevicesinc/gr-iio.git
cd gr-iio
cmake .
make
sudo make install
cd ..
sudo ldconfig
```

Figura 2. 10 Instalación de paquete de librerías Gr-iio

Una finalizada la instalación podemos utilizar los bloques PlutoSDR: Source y Sink acorde a la Figura 2. 11.

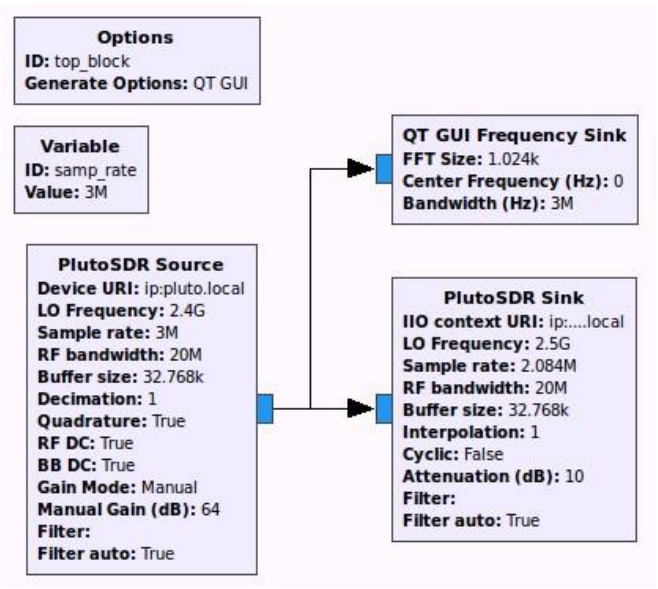

Figura 2. 11 Bloques PlutoSDR Source y PlutoSDR Sink para GNU Radio

CAPÍTULO 3

3. Implementación de un trasmisor y un receptor QPSK con Adalm-Pluto SDR

El modelo implementado es un trasmisor de mensajes indexados de la palabra "Hola mundo!", en una frecuencia central especificada. Ese mensaje se procedió a demodularlo utilizando un receptor QPSK.

Se implementó un sistema de comunicaciones completo con modulación QPSK utilizando dos dispositivos Adalm-Pluto, conectados a la computadora a través de puertos USB. Para configurar e implementar este sistema se utilizó Matlab, Simulink y los bloques de comunicación Adalm-Pluto Toolbox. Con esta implementación se pone a prueba la capacidad del dispositivo como asistente en el aprendizaje y puesta en práctica de la teoría de las telecomunicaciones.

Una de las consideraciones iniciales, es que debido a las variaciones de hardware entre las radios ADALM-PLUTO, es probable que sea necesario una compensación de frecuencia entre el hardware del transmisor y el hardware del receptor. En las pruebas ejecutadas no fue necesario, no obstante, el fabricante aconseja realizar una calibración manual de la frecuencia.

Esta implementación aborda cuestiones prácticas de las comunicaciones inalámbricas, tales como frecuencias portadoras desplazamiento de fase, desplazamiento de tiempo y sincronización de frames, codificación de mensaje y decodificación del mismo, enviando como resultado el mensaje recibido al visor de resultados del Simulink.

3.1. Implementación del trasmisor

Basados en la documentación existente en el sitio de soporte, se implementó el sistema trasmisor, el cual trasmitió las cien cadenas de texto de "Hola mundo" consecutivas y numeradas en la frecuencia central especificada en el archivo de inicialización llamado plutoradioqpsktransmitter_init, el cual consta en los anexos de este trabajo. En la Figura 3. 1 se muestra el esquema de bloques del trasmisor.

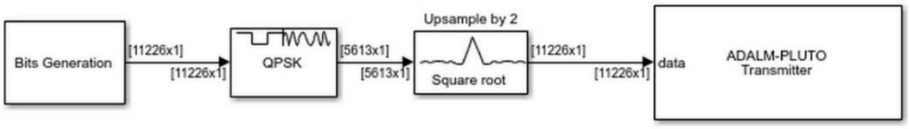

Figura 3. 1 Bloque Trasmisor QPSK

El primer bloque llamado generador de bits se encarga de la producción de la secuencia de bits a trasmitir, en este caso una transferencia de código ASCII.

Se necesita un método para identificar el inicio del mensaje y permitir la interpretación del mismo, por lo que se debe definir un protocolo de trasmisión, siendo necesario dividir el mensaje en frames. Un frame de información contiene un elemento de cabecera o header y un elemento de datos o payload, como se muestra en la Figura 3. 2.

Figura 3. 2 Encabezado y carga útil de un frame.

En esta implementación la cabecera del frame tiene la función de permitir el proceso de sincronización en el receptor y asegurar la correcta alineación del segmento de datos, por consiguiente, poder conseguir la correcta extracción de la información.

En la sección de datos del frame se trasmitieron las cadenas de caracteres “Hola mundo!” concatenados con la numeración del frame, en este caso fueron numerados desde 001 hasta 100, lo cual permitió mayor facilidad de verificación de la trasmisión.

Para lograr la sincronización a nivel de frame se agregó una secuencia de preámbulo, la cual debe tener una correlación adecuada para ser fácilmente identificable.

El proceso estadístico de correlación determina el grado de similitud entre dos señales, en este caso una secuencia de bits. Por definición la

correlación entre dos secuencias s1 y s2, ambas de longitud N, puede ser expresada como se muestra en la Figura 3. 3.

$$Corr_{1\,2} = \sum_{n=0}^{N-1} s1[n]s2[n]$$

Figura 3. 3 Definición de correlación de dos señales

Esta es usada para generar una medida de tiempo, para ayudar a recuperar los datos de los frames recibidos, para esto se tiene una secuencia conocida de bits, la cual se autocorrelaciona con la secuencia recibida, en este caso teóricamente las dos señales son iguales, produciéndose un máximo de correlación cuando las dos señales están alineadas, y se produce un mínimo cuando las secuencias están desalineadas, tal como se muestra en la Figura 3. 4.

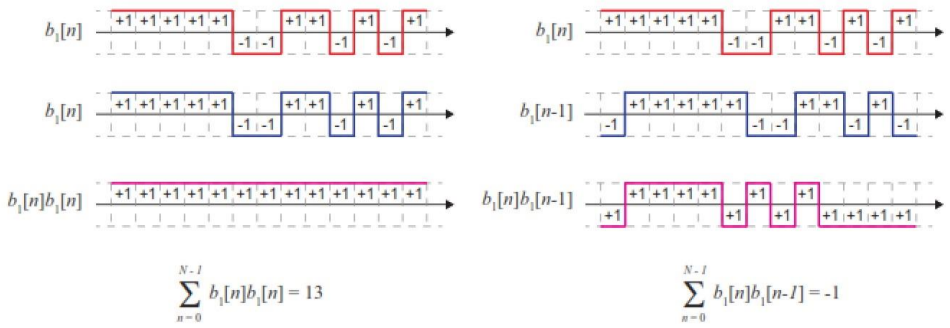

Figura 3. 4 Comparación por autocorrelación de dos señales

Los códigos de secuencia Barker son usados para propósitos de sincronización y de carácter determinístico, siendo ideales para aplicar autocorrelación [9]. Estos códigos están definidos por la fórmula mostrada en la Figura 3. 5.

$$R_{ap}(n) = \begin{cases} \sum_{i=1}^{P-n} c_{i+n}c_i & \text{for } 0 \le n \le P-1 \\ \sum_{i=1}^{P+n} c_i c_{i-n} & \text{for } -P+1 \le n \le 0 \\ 0 & \text{for } |n| \ge P \end{cases}$$

$$\mathbf{c}_2 = (+1, -1)$$
$$\mathbf{c}_3 = (+1, +1, -1)$$
$$\mathbf{c}_4 = (+1, +1, -1, +1)$$
$$\mathbf{c}_5 = (+1, +1, +1, -1, +1)$$
$$\mathbf{c}_7 = (+1, +1, +1, -1, -1, +1, -1)$$
$$\mathbf{c}_{11} = (+1, +1, +1, -1, -1, -1, +1, -1, -1, +1, -1)$$
$$\mathbf{c}_{13} = (+1, +1, +1, +1, +1, -1, -1, +1, +1, -1, +1, -1, +1)$$

Figura 3. 5 Códigos Barker de tamaño 13

Los códigos Barker solo están limitados a longitudes de tamaño 2,3,4,5,7,11,13, no existiendo en otras longitudes, para este trasmisor se utilizó un código Barker de tamaño 13.

Cuando se aplica autocorrelación a una secuencia Barker, se obtiene un pico de +13 en el momento que las secuencias están alineadas, mientras que se obtiene un bajo nivel de autocorrelación mientras no existe alineamiento, tal como se muestra en la Figura 3. 6.

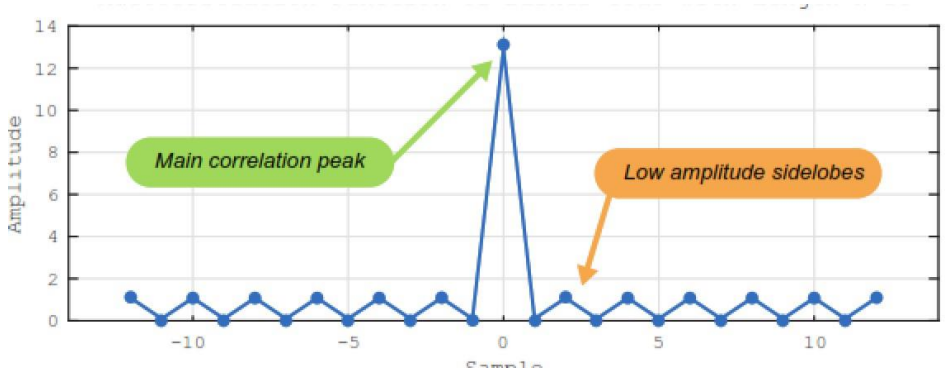

Figura 3. 6 Detección de alineamiento con código Barker de tamaño 13

Cuando la secuencia Barker es detectada, se identifica como el inicio de un frame, y se puede extraer la información contenida en la sección de datos del frame. El bloque generador de bits se muestra en la Figura 3. 7.

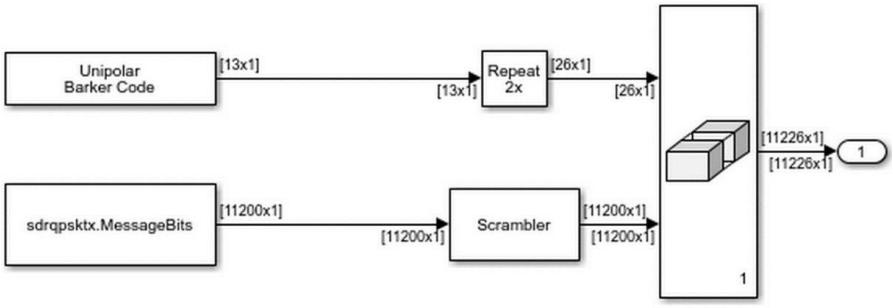

Figura 3. 7 Bloque de generación de bits.

Esta secuencia de bits genera una matriz de 13 filas por 1 columna, la cual pasa por un módulo repetidor que la convierte a una matriz de 26 filas por 1 columna. Esto se realiza con la finalidad tenga un tamaño par.

Para generar la sección del bloque de datos, el modelo de Simulink inicializó las variables en la mediante una llamada a la función de Matlab plutoradioqpsktransmitter_init.

En esta función se definió la estructura TrasmisorQPSK, la cual consta principalmente de las siguientes variables internas:

BarkerCode = [+1 +1 +1 +1 +1 -1 -1 +1 +1 -1 +1 -1 +1];

Message = 'Hola mundo!;

MessageLength = length(SimParams.Message) + 5;

NumberOfMessage = 100;

A través de codificación en matlab se concatena el número de frame al mensaje “Hola Mundo!”, estando estos numerados desde 000 hasta el 099, luego esta cadena se la convierte a una secuencia de bits. Es de notar que los caracteres ASCII imprimibles, solo requieren 7 bits por carácter, por lo que la longitud total del mensaje es de 11200 bits (16 caracteres x 7 bits x 100 mensajes). En la Figura 3. 8 se muestra la configuración del bloque de entrada.

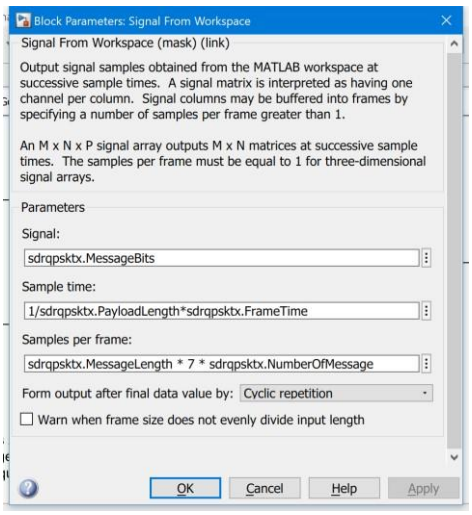

Figura 3. 8 Entrada de bloque generador de bits

Esta secuencia de bits fue asignada a la variable MessageBits y alimenta el bloque Scrambler del trasmisor. Este bloque viene incluido dentro de las librerías del communications toolbox de Simulink.

Existen secuencias de bits problemáticas, tales como largas secuencias de ceros o unos, que pueden formar parte de una trasmisión, sobre la cual no existe forma de controlar su trasmisión. Estas cadenas de bits podrían ser fuente de de problemas como interferencias, intermodulación, diafonía, dificultad para poder realizar la sincronización y la ecualización. Un método para evitar estos problemas es utilizar un bloque de scrambler aleatorizador, eliminando estas cadenas repetitivas de ceros y unos, mapeándose en una secuencia codificada que parezca más aleatoria, menos problemática y así poder asegurar la correcta recuperación del sincronismo del mensaje [10].

El bloque Scrambler forma parte del communications toolbox de Simulink y fue configurado acorde a la Figura 3. 9.

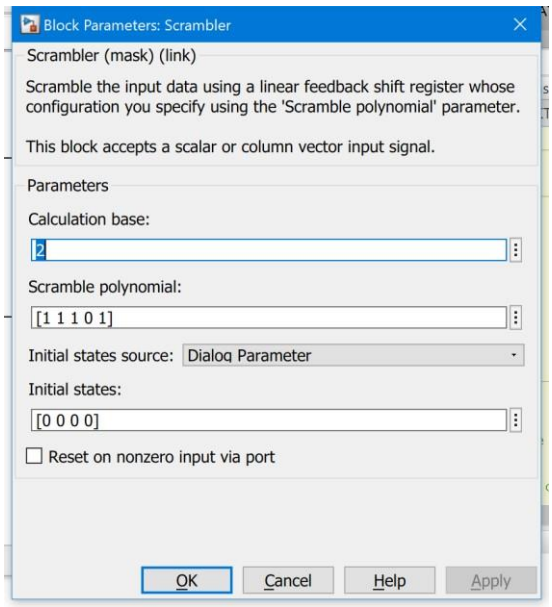

Figura 3. 9 Configuración de bloque Scrambler

Luego de haber pasado por el bloque aleatorizador, la sección de cabecera y de datos se unen a través de un bloque concatenador de matrices llamado en Simulink como Matrix Concatenate, el resultado es una matriz de 11226 filas x 1 columna que lleva un mensaje completo a trasmitir.

La matriz con el mensaje completo es insumo del bloque QPSK Modulator Baseband el cual se describe a continuación.

Los sistemas de comunicación digital operan mapeando datos binarios en símbolos y el método de mapeo es definido por el esquema de modulación elegido. Para la implementación de este sistema de trasmisión se escogió un sistema de modulación QPSK (Quaternary Phase Shift Keying) el cual es basado en fase, y lo podemos visualizar en un plano cartesiano con una componente real en el eje-x y una componente compleja Q en el eje-y. El esquema QPSK mapea 2 bits por símbolo y se lo puede apreciar en la Figura 3. 10, la cual representa su constelación usando codificación Gray.

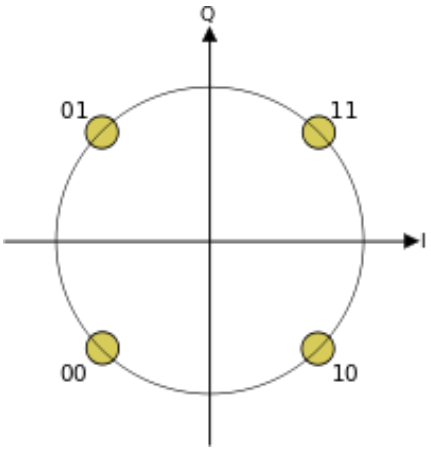

Figura 3. 10 Constelación QPSK con codificación Gray

El proceso de mapeo separa los grupos de bits y los trasmite con un símbolo apropiado, el proceso inverso debe existir en el receptor, donde basado en el símbolo recibido, se extrae la información. Normalmente existe un grado de error en la trasmisión, producto del ruido u otros efectos indeseables del canal o error en la sincronización. Esta decodificación es basada en fronteras de decisión, basado en la cercanía a los puntos teóricos de la constelación. Para QPSK estas fronteras están dadas acorde a la Figura 3. 11.

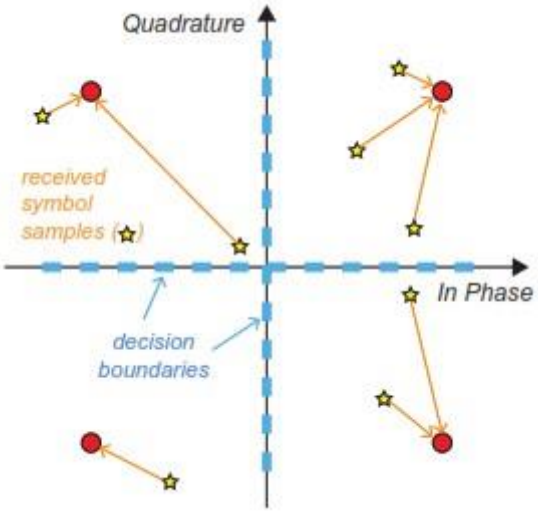

Figura 3. 11 Fronteras de decisión en QPSK

Se procedió a configurar el módulo QPSK Modulator baseband acorde a la Figura 3. 12.

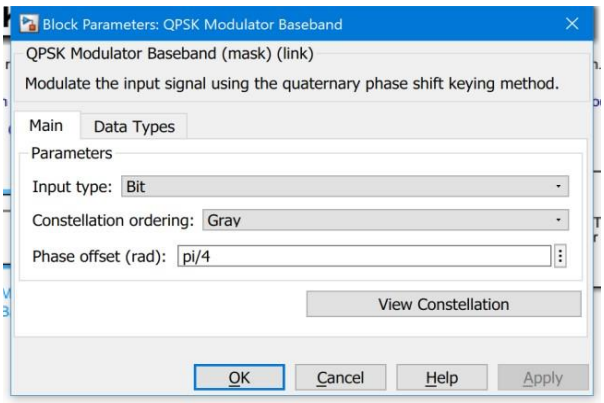

Figura 3. 12 QPSK Modulator Baseband

El siguiente bloque es el filtrado de los símbolos generados por el modulador, se utilizó el bloque Raised Cosine Trasmit Filter, configurado como filtro de raíz de coseno alzado en el cual se aumentó la tasa de muestreo de los símbolos modulados en factor de upsampling de 2 con un factor de roll-of de 0,5. La tasa de salida del filtro raiz de coseno alzado se estableció en 400k muestras / segundo con una tasa de símbolos de 200k símbolos por segundo, debiendo en la implementación del receptor coincidir con la tasa de símbolos del transmisor, acorde a la Figura 3. 13.

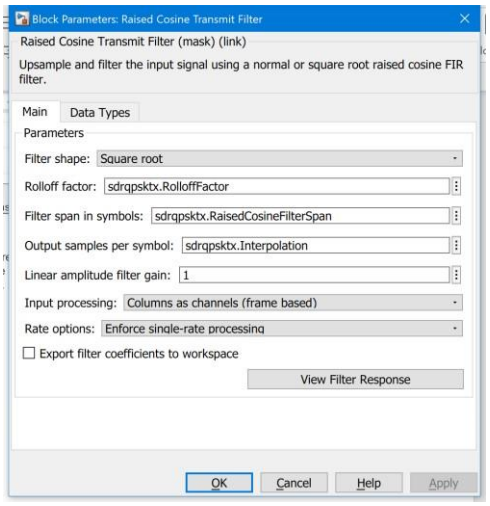

Figura 3. 13 Configuración de bloque Filtro de Coseno Alzado

El principal propósito del filtro del trasmisor es limitar el ancho de banda de la señal, y evitar emisiones en bandas adyacentes, ajustándose a las especificaciones definidas, también llamada máscara espectral. Este filtrado debe ser diseñado de tal manera que no introduzca interferencia intersímbolo (ISI). Es decir, el filtro no debe provocar que los sucesivos símbolos filtrados interfieran entre sí. El filtro Raiz de coseno alzado tiene la propiedad deseada de ISI cero. Específicamente, a pesar de que la respuesta del coseno elevado generalmente se extiende a lo largo de varios períodos de símbolo, la contribución de todos los demás símbolos es cero en los puntos de muestreo ideales, como lo ilustra la Figura 3. 14 [4].

Los bloques de filtro de transmisión de coseno elevado y filtro de recepción de coseno elevado están diseñados para su uso en el transmisor y el receptor, respectivamente. El filtro de transmisión emite una señal muestreada (interpolada), mientras que el filtro de recepción espera que su señal de entrada sea muestreada diezmada antes de enviarla al puerto de salida.

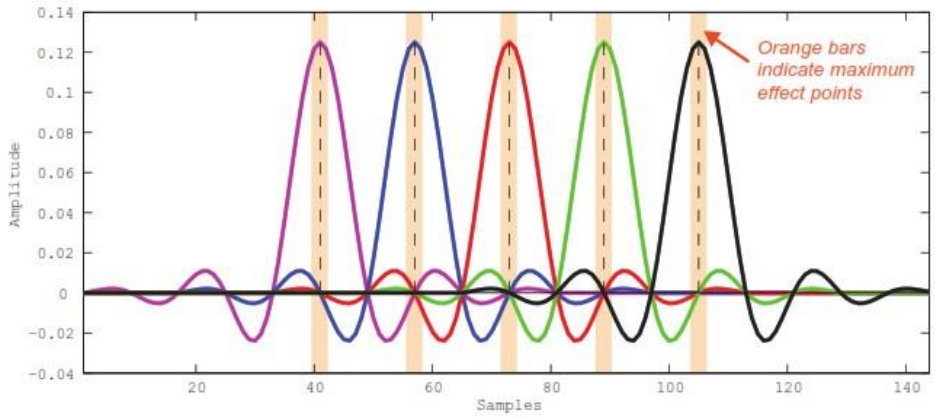

Figura 3. 14 Propiedad de ISI=0 con filtro de coseno alzado

Dependiendo de la frecuencia portadora de RF involucrada, actualmente puede ser posible realizar todo el procesamiento digitalmente, se debe considerar los métodos para la conversión ascendente y descendente

digital. Estos métodos de Upsampling o downsampling se utilizan para modular y demodular una señal respectivamente, al mismo tiempo que realizan las transiciones multivelocidad necesarias para moverse entre la velocidad de símbolo, y la velocidad de muestreo de los convertidores DAC o ADC.

En el trasmisor ocurre el proceso de upsampling o sobremuestreo a través de una serie de pasos de filtrado que pasa de una velocidad de símbolo hasta alcanzar una velocidad o tasa de muestreo del convertidor digital analógico DAC, en la Figura 3. 15 se ilustra este proceso

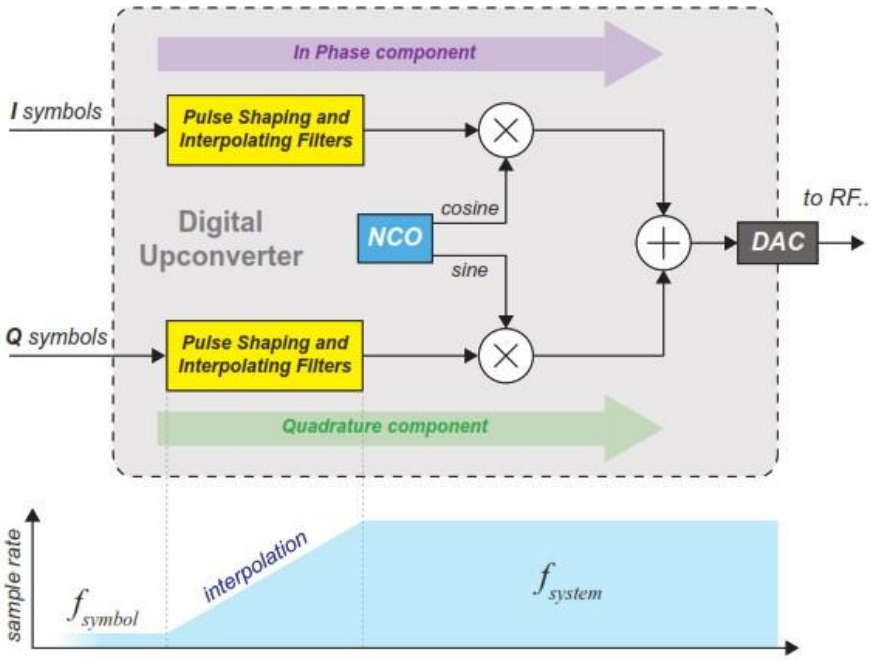

Figura 3. 15 Filtrado e interpolación de las señales

Se necesita generar imágenes espectrales que sean múltiplos de la tasa original de muestreo, luego pasada la señal por un filtro paso bajo, esta operación en conjunto se denomina interpolación.

En la práctica es más eficiente realizar este proceso utilizando filtros en cascada, es decir si necesitamos interpolar en un múltiplo de 200, es más eficiente hacerlo en pasos, por ejemplo: primero 20,2 y 5 respectivamente.

Una vez que la se ha realizado el proceso de interpolación, la señal pasa al bloque ADALM-Pluto Radio Transmitter, el cual es configurado acorde a la Figura 3. 16.

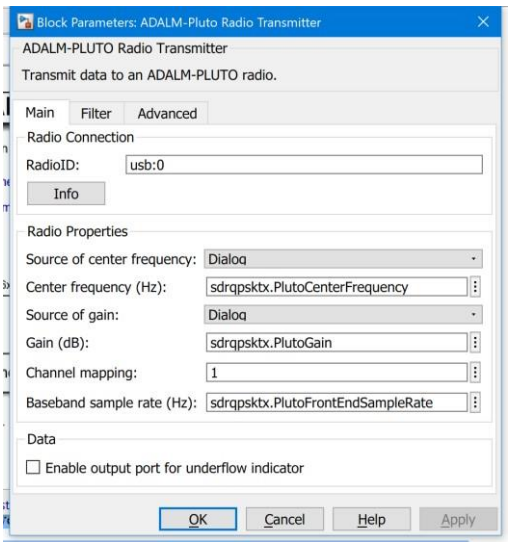

Figura 3. 16 Configuración del bloque trasmisor Adalm-Pluto

La frecuencia portadora fue configurada al valor de 915 MHz, el cual está en los rangos de especificación del dispositivo, y con tasa de muestreo 400KHz.

3.2. Implementación del receptor

Este ejemplo muestra cómo utilizar los objetos del sistema de radio ADALMPLUTO para implementar un receptor QPSK. El receptor demodula los símbolos recibidos e imprime un mensaje simple en la línea de comandos de MATLAB. Se implementó con el receptor QPSK utilizando un ADALM-PLUTO con Simulink, en el cual se abordó aspectos prácticos tales como frecuencia portadora y desplazamiento de fase, desplazamiento de tiempo y sincronización de trama. Este modelo recibe la señal enviada por el Transmisor QPSK implementado previamente.

El esquema del receptor es el siguiente se muestra en la Figura 3. 17.

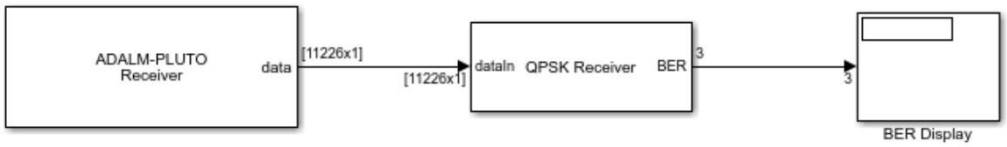

Figura 3. 17 Bloque principal de recepción

El primer nivel del esquema tenemos bloque Adalm-Pluto Receiver, el cual fue configurado bajo los mismos parámetros que los configurados en el correspondiente bloque trasmisor acorde a la Figura 3. 18.

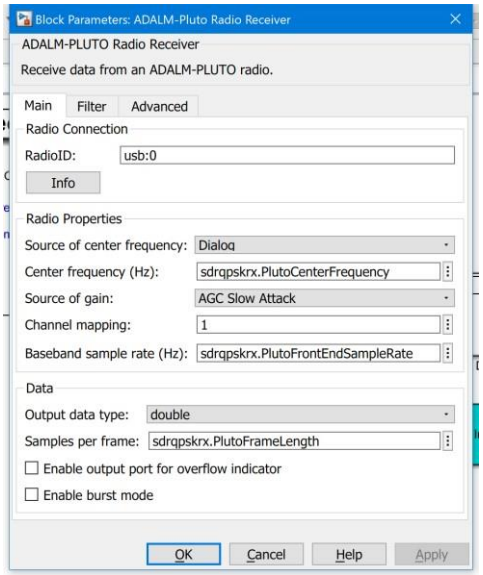

Figura 3. 18 Configuración de bloque receptor Adalm-Pluto

Debido a que puede haber ligeros desajustes en la calibración de frecuencia, es recomendable realizar el proceso de calibración de frecuencia, entre el dispositivo trasmisor y receptor.

El bloque Adalm-Pluto tiene como salida una matriz de 11226 filas por 1 columna, la cual fue enviada al bloque QPSK Receiver, el mismo que se muestra en la figura Figura 3. 19.

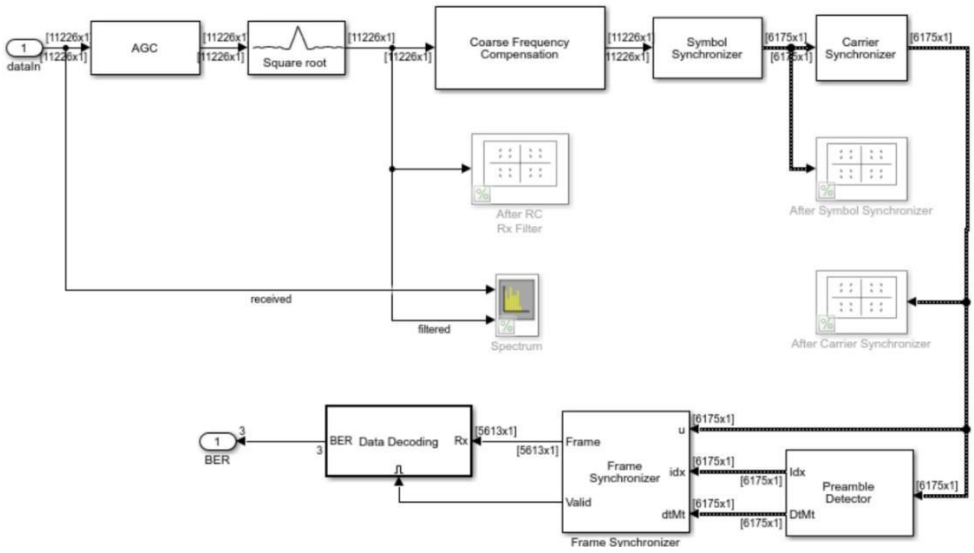

Figura 3. 19 Bloque QPSK del receptor

La señal luego fue ajustada por medio con ayuda del bloque de control automático de ganancia AGC, en donde se ajustó y estabilizó la amplitud de la señal recibida para asegurar precisión en el proceso de sincronización de portadora y sincronización de símbolo, asegurando que estos valores se mantengan constantes en el tiempo. Este bloque de ajuste de ganancia fue colocado antes del bloque de filtrado. En el siguiente bloque ocurre el filtrado de la señal , para esto se utiliza un filtro Raíz de Coseno Alzado, con factor de rollof de 0.5, acorde al utilizado en la etapa de trasmisión.

Luego esta señal pasa el bloque de ajuste compensador de frecuencia, este bloque fue el encargado de corregir la señal de entrada mediante una estimación aproximada del desplazamiento de frecuencia. En la Figura 3. 20 se puede notar, que el desplazamiento de frecuencia se estima promediando la salida del algoritmo basado en correlación del bloque Compensador de frecuencia aproximada. Puede subsistir un desplazamiento de frecuencia residual, el cual luego es compensado en el bloque de sincronización de portadora.

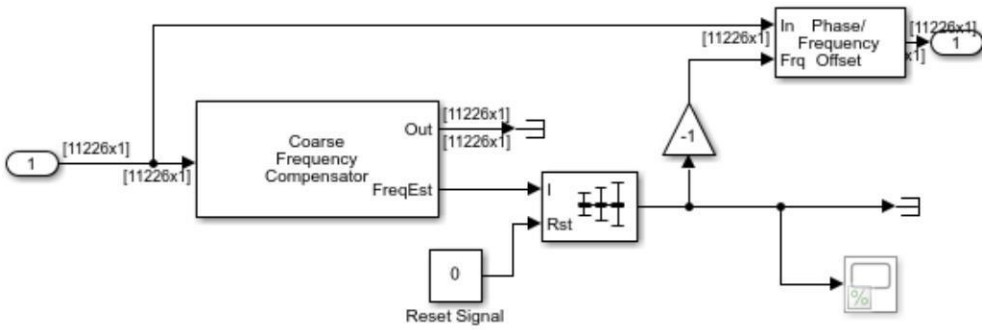

Figura 3. 20 Bloque de ajuste compensador de frecuencia

Se puede suponer que hay tres tipos de problemas cuando no hay sincronización de frecuencia

1. Donde el oscilador local LO en el receptor tiene exactamente la misma frecuencia que la portadora recibida, pero está desfasado.

2. Donde el LO en el receptor tiene una frecuencia ligeramente diferente a la de la portadora recibida.

3. Donde las frecuencias del oscilador local LO de recepción y la señal portadora recibida cambian entre sí.

Esto puede verse en la Figura 3. 21.

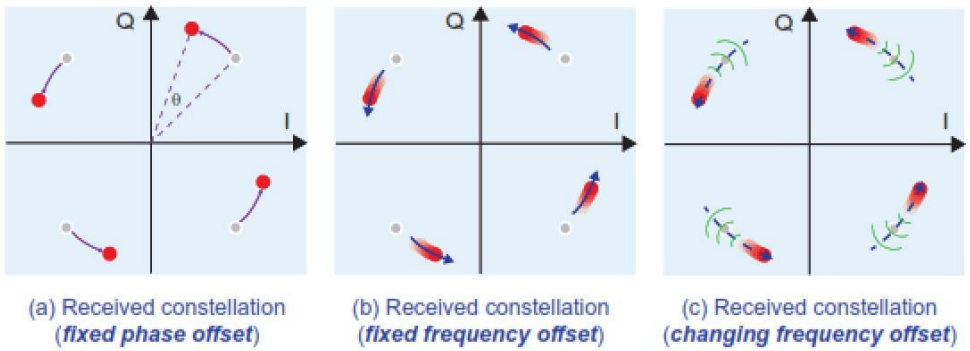

Figura 3. 21 Diferentes tipos de errores en fase y frecuencia

Luego de ajustada la frecuencia, la señal pasa al bloque Symbol Synchronizer encargado de recuperación de la sincronización de símbolos, el cual utiliza un circuito PLL (Phase Locked Loop) para realizar esta corrección. Sin esta

sincronización, los desplazamientos en las fases harían imposible la recuperación de los datos trasmitidos, este circuito consigue igualar la fase del oscilador local con la fase de la señal de entrada.

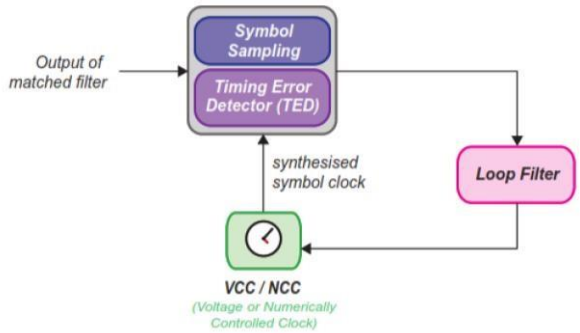

Figura 3. 22 Modelo de sincronizador genérico

En la Figura 3. 22 se muestra un sincronizador genérico, el lazo de sincronización consigue ajustar el tiempo de muestreo de los símbolos basado en mediciones grado de error en los tiempos.

El muestreo de la señal en el receptor puede tener imperfecciones que pueden dar lugar a muestreos no óptimos tales como: tiempos de fase erróneos, el cual ocurre cuando la muestra es tomada a con la tasa correcta de símbolos, pero con la fase incorrecta. El resultado en el muestreo es que este es tomado antes o después de los puntos apropiados, significando valores SNR poco óptimos, como consecuencia puede ocurrir interferencias inter-símbolos ISI.

En la Figura 3. 23 se ilustra una comparación entre un muestreo de una comunicación BPSK, en la fase correcta y la siguiente figura con el muestreo en la fase incorrecta.

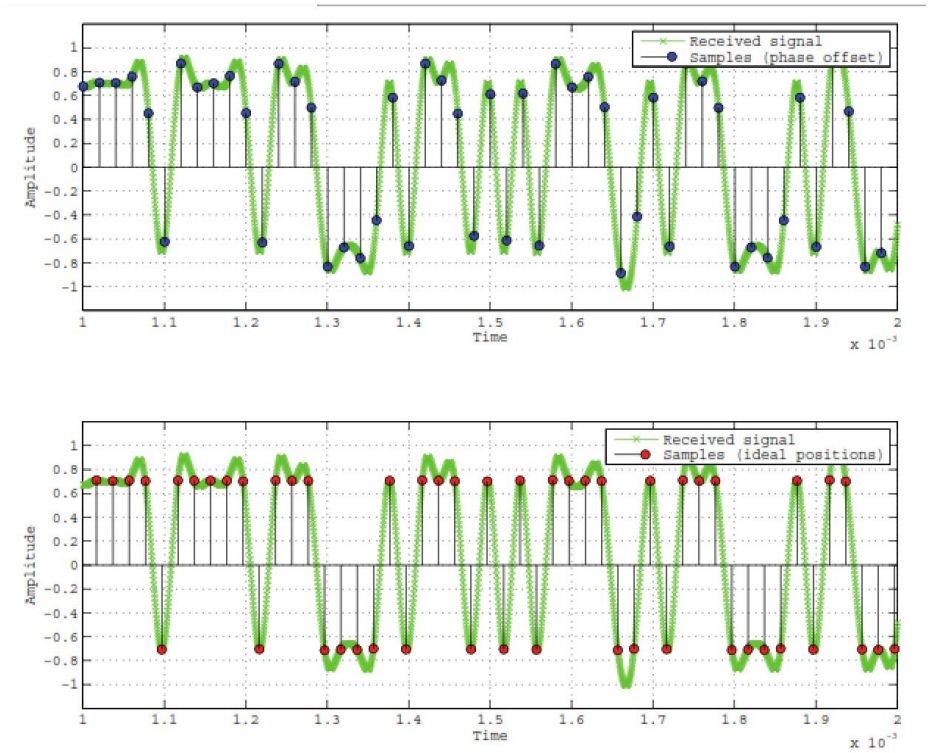

Figura 3. 23 Arriba: Muestreo BPSK en la fase correcta y abajo muestreo en la fase incorrecta

Existe también el error que puede darse cuando las muestras son tomadas a una frecuencia incorrecta, más rápido o lento que la tasa de símbolos, ocasionando que la muestra esté en los puntos correctos solo una fracción del tiempo. Otro efecto indeseable es el tiempo de jitter, ocasionado por variaciones aleatorias que pueden ser atribuidas a las características físicas de los convertidores ADC, siendo este un efecto práctico que ocurre en los dispositivos reales [4].

Una vez sincronizados los tiempos, se procede a realizar un ajuste o compensación pequeño de frecuencia, esto es desarrollado en el bloque Carrier Syncronicer, el cual utiliza un circuito PLL, para tratar desplazamientos residuales de frecuencia y fase en la señal de entrada.

Luego la señal es enviada al bloque detector del prefijo y sincronización de frame, llamado Preamble Detector, el cual utiliza el encabezado de trama conocido, en este caso un código Barker conocido y modulado por el trasmisor

QPSK, el cual se correlaciona con los símbolos recibidos, con la finalidad de encontrar la ubicación del encabezado de la trama, también transforma la trama de tamaño variable en una trama de tamaño fijo, lo cual es necesario para el procesamiento posterior, esta señal puede ser enviada así al bloque de decodificación de datos, Data Ber Decoding, el cual es mostrado en la Figura 3. 24.

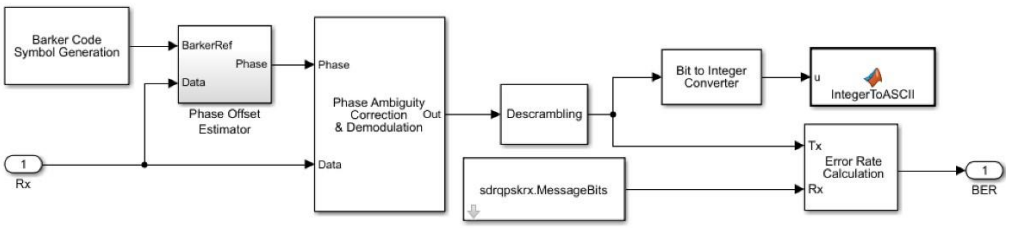

Figura 3. 24 Bloque decodificador de datos

Este subsistema es encargado de hacer la resolución de ambigüedad de fase, la demodulación y la decodificación de mensajes de texto. El bloque del sincronizador de portadora puede bloquearse en la portadora no modulada con un desplazamiento de fase de 0, 90, 180 o 270 grados, lo que puede provocar una ambigüedad de fase. El bloque Phase Offset Estimator determina este cambio de fase. Con los datos corregidos, se procede a realizar el proceso inverso al scrambler y se decodifican los datos, obteniendo el mensaje ASCII trasmitido, estos se almacenan y se muestran en el visualizador de diagnóstico de Simulink. Como tenemos el mensaje teórico trasmitido en variables del entorno de Matlab, se puede proceder a mostrar el grado de error de la trasmisión.

3.3. Resultados de la implementación

Para la ejecución correcta de la trasmisión y recepción, se instanció dos sesiones diferentes de Matlab y Simulink, para que la asignación de tiempos de procesador de la computadora sea lo más equitativo posible, ya que son aplicativos que consumen altos recursos en la computadora.

Se utilizó como host de los dispositivos, una computadora con procesador Intel core i7 6600U con 8 GB de Ram.

Los parámetros de inicio del sistema se encuentran en dos scripts de Matlab, los cuales son ejecutados al principio de la comunicación.

En el caso del bloque de trasmisión se hace una llamada en el evento PostLoadFcn, es decir cuando el modelo de Simulink esta cargado en memoria, configurado tal como indica la Figura 3. 25. Este bloque de inicialización se lo puede ver en el anexo.

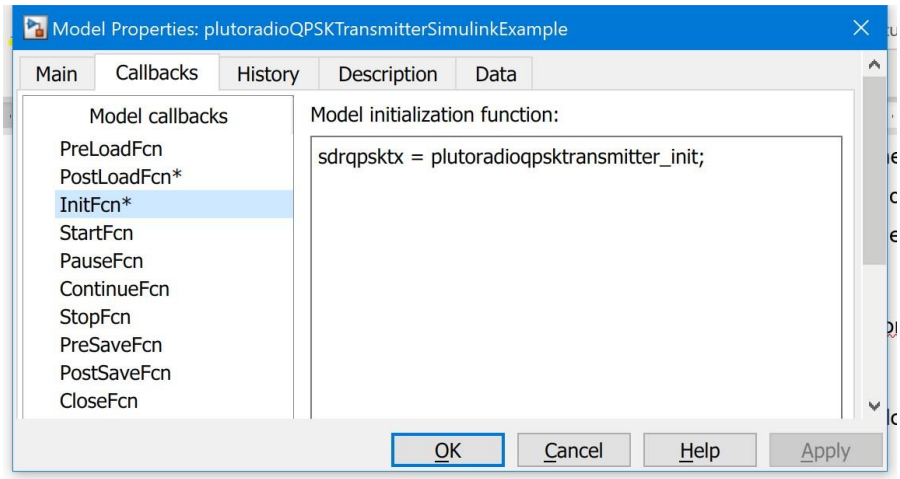

Figura 3. 25 Llamada de inicialización de trasmisor

Entre los parámetros de trasmisión inicializados importantes tenemos:

Message = 'Hola Mundo!'

NumberOfMessage = 100

RolloffFactor = 0.5

PlutoCenterFrequency = 915e6;

PlutoGain= 0;

En el bloque de recepción también se inicializan variables, acorde a la llamada PostLoadFcn, tal como se muestra en la Figura 3. 26. Este código de inicialización puede ser consultado en el anexo.

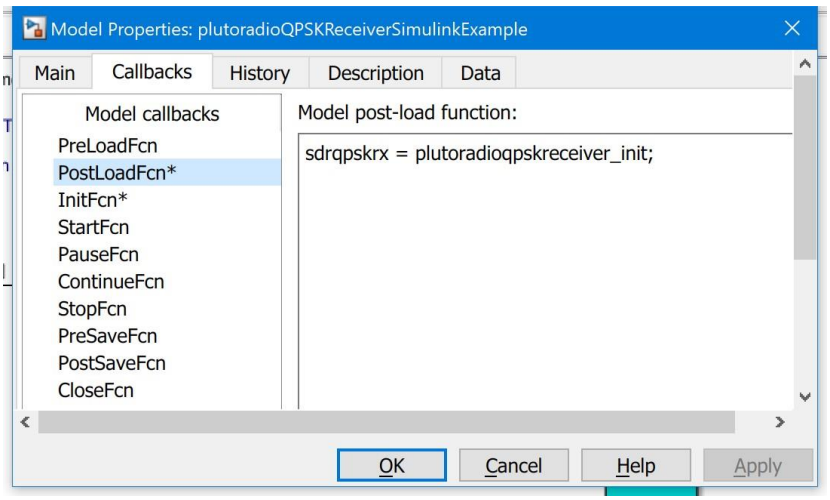

Figura 3. 26 Llamada de inicialización de receptor

Entre las variables de inicio mas importantes tenemos:

Message= 'Hola Mundo!';

RolloffFactor= 0.5;

PlutoCenterFrequency = 915e6;

PlutoGain = 30;

Es de notar que el mensaje original se lo requiere para hacer la comparación y cálculo de la tasa de error de bits (BER).

Una vez ejecutados los dos modelos, se puede apreciar en un analizador de espectro la señal, luego de aplicado el filtro raíz de coseno alzado. Puede notarse en la señal de color celeste, mostrada en la Figura 3. 27, la señal luego de aplicado el filtro versus la señal no aplicado el filtro. Este filtro permite ajustar el ancho de banda para evitar ISI y emisiones en bandas adyacentes.

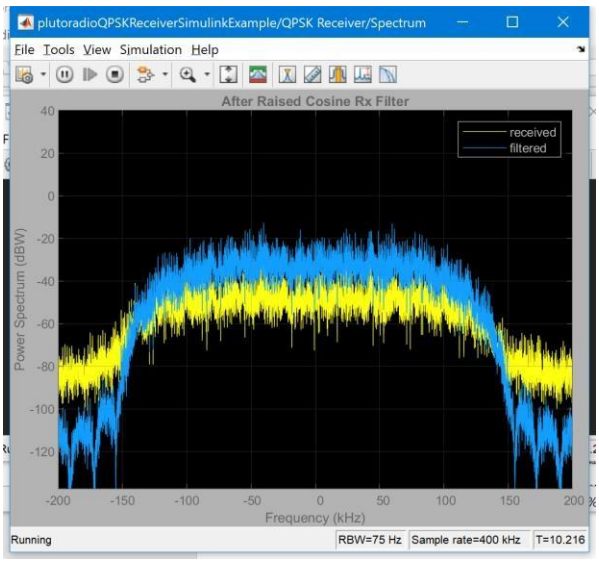

Figura 3. 27 Señal sin filtrar versus señal filtrada

En la Figura 3. 28 podemos notar el diagrama de constelación QPSK, luego de aplicado el filtro, pero la señal tiene desfaces, por lo que es necesario hacer un ajuste corrector de frecuencia.

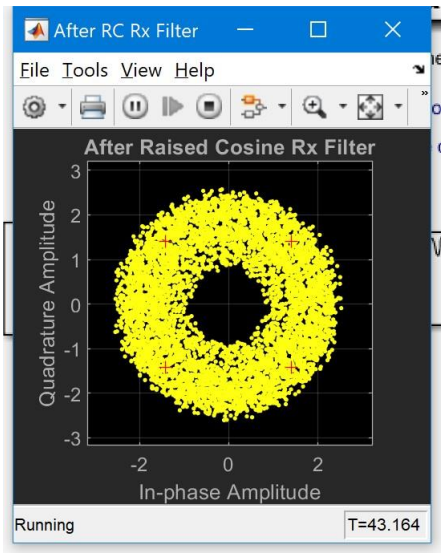

Figura 3. 28 Constelación QPSK aplicado el filtro

En la Figura 3. 29, se puede apreciar que la señal ha sido ajustada, mediante un proceso de sincronización de portadora y de símbolo, pero aún existen desfaces residuales de frecuencia.

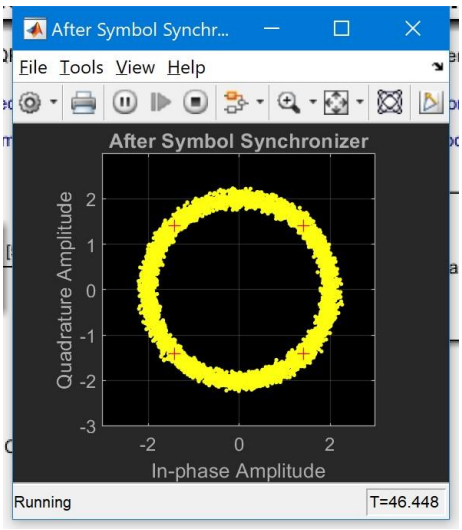

Figura 3. 29 Constelación después de sincronización de símbolos

En la Figura 3. 30, notamos la constelación QPSK, luego de realizado el ajuste residual de frecuencia, permitiendo decodificar con certeza los mensajes

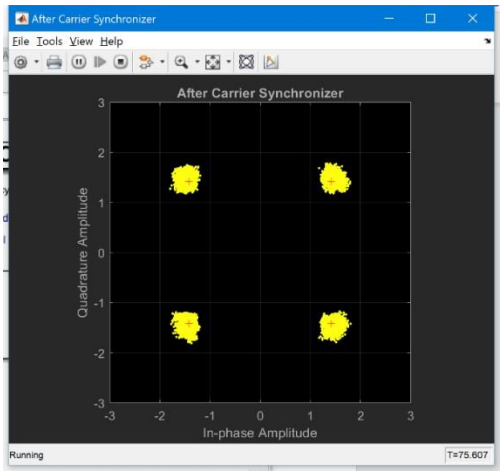

Figura 3. 30 Constelación QPSK luego de sincronización de portadora

La Figura 3. 31 muestra el visor de Simulink, donde podemos apreciar la cadena de texto recibidos

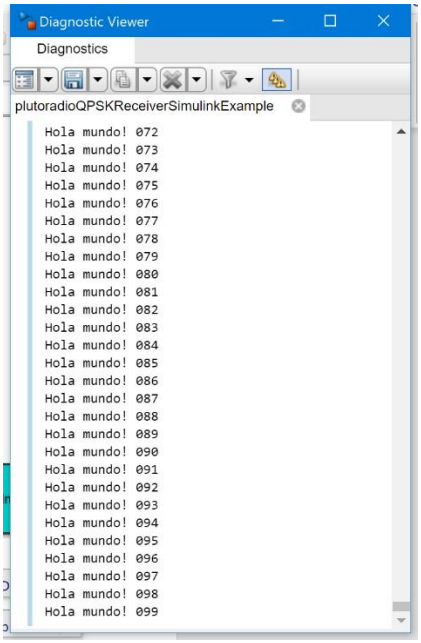

Figura 3. 31 Visor con datos recibidos

En la Figura 3. 32 se muestra el visor, cuando ocurren errores, por inestabilidad del sistema, debido a interferencias o atenuaciones por obstáculos. No obstante, dichas variables serían motivo de otro estudio.

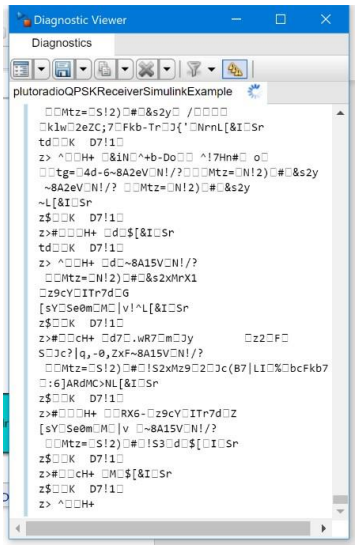

Figura 3. 32 Visor: Resultado cuando se pierde sincronización

El BER se calcula con el bloque Error Rate Calculation de Simulink mostrado en la Figura 3. 33, el cual tiene 2 entradas, en las cuales una es la señal Recibida Rx y la otra corresponde a la señal Trasmitida Tx teórica, es decir la que consta en la variable de inicialización del receptor.

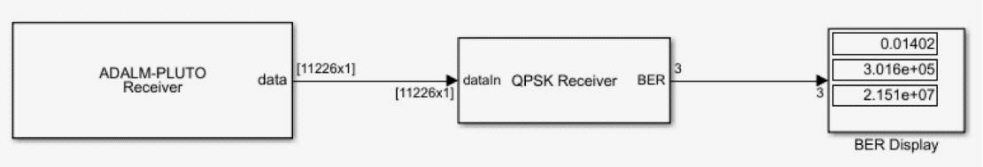

Figura 3. 33 Bloque para cálculo de BER

La tasa de error de bits se calcula dividiendo El número total de errores, es decir, el número de casos en que un elemento Rx no coincide con el elemento Tx correspondiente.

En las ejecuciones realizadas, se obtuvo una tasa de error promedio de 0.014, es decir un porcentaje de error en el orden del 1,4%, el cual podría considerarse alto. No obstante, se cumplió uno de los objetivos del experimento, al implementar un sistema de comunicaciones QPSK con el dispositivo Adalm-Pluto.

CAPÍTULO 4

4. Evaluación Comparativa de Dispositivos SDR

Ahora que la tecnología SDR se encuentra al alcance comercialmente y que sus costos han decrecido, se procedió a evaluar algunas marcas de dispositivos muy populares, tomando en consideración los siguientes aspectos:

- Que tenga un costo accesible para estudiantes, investigadores o profesionales.
- Que el equipo requiera menos potencia computacional y pueda trabajar en una amplia gama de frecuencias portadoras y sea capaz de manejar anchos de banda adecuados y prácticos. Es decir que opere dentro de un rango de frecuencias adecuado y que puedan operar con las bandas para FM, DAB, TDT, Televisión por Satélite, GPRS(2.5G), UMTS(3G), LTE(4G), 5G, Wi-Fi (2.4 y 5 GHz) y Bluetooth.
- Que la compañía fabricante proporcione algún grado de soporte, mediante componentes, algoritmos, módulos, y documentación de los mismos, con la finalidad de que la curva de aprendizaje sea la menor posible.
- Que sea fiable en los experimentos en tiempo real, a nivel de hardware y software, de tal manera que proporcione un aprendizaje y experimentación con alto nivel de confiabilidad.
- Que tenga una base amplia de conocimiento creada por usuarios de sus sistemas.

Se procedió a evaluar las características principales de los dispositivos SDR representativos en el mercado.

4.1. USRP

En el mercado actual, a nivel de equipos SDR, existe un líder indiscutible: National Instruments, con sus equipos denominados USRP (Universal Software Radio Peripheral). Esta marca ha permitido a ingenieros y profesionales, el diseño, prototipado e implementación rápida de sistemas complejos de comunicaciones, contando además con el respaldo comercial,

tecnológico y de soporte propio de una empresa multinacional y líder de su segmento.

Es de esperar que sus productos muy precisos y confiables, tengan un costo que se consideraría elevado para un estudiante o investigador que no esté auspiciado por una institución. No obstante, National Instruments tiene una segunda marca llamada Ettus Research USRP, las cuales ofrecen menores costos para una gama de modelos equivalentes entre sí.

En la Figura 4. 1 se establecen las diferencias entre las dos marcas.

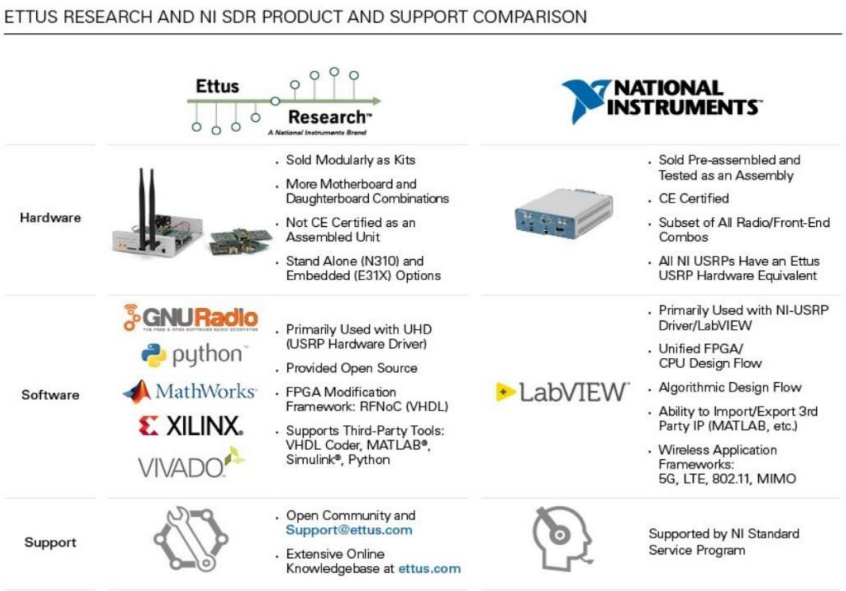

Figura 4. 1 Diferencias clave entre marca Ettus y National Instruments

4.2. USRP National Instruments

Acorde a información de la página web de este fabricante, podemos notar que la marca National Instruments USRP provee equipos certificados, lo cual garantiza los parámetros ofrecidos de sus equipos y un alto nivel de calidad. Además, sus productos se venden como módulos o preensamblados, siempre han pasado pruebas de calidad rigurosas. Poseen un software propio para programarlos: LabView, no obstante, si existe compatibilidad con Matlab y Simulink. Permite además programar sus circuitos FPGA. Además, hay

programas de soporte a nivel mundial, que garantizan acompañamiento en los proyectos. Obviamente esto posiciona estos equipos en una categoría premium, en cuanto a calidad y precios.

Como ejemplo, consultamos el USRP 2920, el cual tiene un rango de operación desde los 50 MHz hasta los 2,2 GHz, y con un ancho de banda de 20 MHz, acorde la Figura 4. 2, notamos que su precio es de $ 3855 USD [11]. Teniendo en cuenta que para un sistema de comunicaciones completo con trasmisión y recepción se requieren dos equipos, este valor total se duplicaría, resultado un valor alto para la economía de países latinoamericanos. No obstante, esta marca es ampliamente usada en universidades de todo el mundo.

Figura 4. 2 National Instruments USRP 2920

4.3. USRP Ettus Research

Por otro lado, tenemos los dispositivos USRP Ettus, subsidiaria de National Instruments, con equipos equivalentes, aunque existen diferencias que describimos a continuación. Ettus vende principalmente módulos no empaquetados, y no tienen certificación. Lo cual significa que no está orientado al mercado profesional que requiera alta prestación y garantía de funcionamiento. Tiene su base de usuarios en el ámbito de laboratorios de universidades, investigadores o entusiastas del software y hardware libre. Tiene compatibilidad con herramientas tales como GNU Radio, Matlab, Simulink, Phyton. No existe un programa de soporte pagado, pero hay grupos, comunidades y foros que son alentados por especialistas de Ettus.

Como ejemplo, tenemos el USRP B205mini-i, el cual tiene un rango de operación que va desde 70 MHz – 6 GHz, con capacidad Full duplex y un ancho de banda de 56 MHz. El soporte a la compatibilidad con GNU Radio es mantenido por especialistas de Ettus y su precio es de $1,105.00 USD [12]. Ver Figura 4. 3. Esta marca es también muy usada en las universidades.

Figura 4. 3 Ettus B205mini-i

4.4. Hack One RF

HackRF One de la empresa Great Scott Gadgets es un dispositivo de radio definido por software con capacidad de trasmisión o recepción de señales de radio en el rango desde 1 MHz a 6 GHz. Diseñado para permitir la experimentación y el desarrollo de tecnologías de radio modernas. Es además una plataforma de hardware de código abierto que puede usarse como un periférico USB o programarse para un funcionamiento independiente, para lo cual puede ser alimentado por batería externa a través de la conexión USB [13]. En sus características descritas en su sitio web, tenemos las siguientes:

- Frecuencia de funcionamiento de 1 MHz a 6 GHz
- Transceptor semidúplex
- Muestreo de 20 millones de muestras por segundo
- Muestras de cuadratura de 8 bits (I de 8 bits y Q de 8 bits)
- Compatible con GNU Radio, SDR #
- Filtro de banda base y ganancia RX y TX configurable por software
- Alimentación del puerto de antena controlada por software (50 mA a 3,3 V)

- Conector de antena SMA hembra
- Entrada y salida de reloj SMA hembra para sincronización
- Conexión USB 2.0
- Alimentado por USB

La documentación disponible es basada principalmente en la experimentación de la comunidad de internet e investigadores del tema. La compañía fabricante nació en el año 2014, producto de una campaña de recolección de fondos de inversión tipo crowfunding, por lo que se puede considerar una compañía pequeña, la cual, aunque tiene un producto innovador, no tiene la capacidad para ofrecer un soporte especializado, ni la documentación requerida sobre sus productos.

No tiene soporte para uso en Matlab y Simulink, estando solo posicionado para su uso en plataformas de software libre como GNU Radio. Su precio consultado en Amazon fue de $ 324.95 USD. Ubicándose en un precio intermedio para el mercado. El dispositivo HackRF One se muestra en la Figura 4. 4.

Figura 4. 4 HackRF One

4.5. RTL-SDR

Es basado en RTL2832U de la compañía Realtek, el cual es un demodulador DVB-T COFDM de alto rendimiento que admite una interfaz USB 2.0. Tiene solamente capacidad de recepción, ya que fue diseñado para uso de recepción de televisión digital. La comunidad investigadora descubrió que podrían ser utilizados como receptores genéricos simplemente poniéndolos en un modo diferente, es decir son capaces de recibir cualquier señal en el rango funcional de su sintonizador. Debido a que la empresa Realtek vende sus circuitos a múltiples fabricantes, existen muchas marcas de SDR las cuales varían sus características dependiendo de los componentes adicionales que utilicen. No obstante, el rango más común de recepción va desde los 25 MHz a 1,75 GHz. [4]

Este circuito SDR recibió el respaldo de MathWorks, que en conjunto con el fabricante NooElec, lanzaron un paquete de soporte de hardware para el RTLSDR a principios de 2014, lo que permitió a MATLAB y Simulink poder interactuar y controlar el RTL-SDR. Con este complemento, las muestras y la salida del dispositivo se puede capturar y trabajar en el software, permitiendo a los usuarios implementar cualquier tipo de receptor o sistema de detección de espectro [14]. Este dispositivo está en la gama inferior de precios, como referencia el modelo de RTL-SDR fabricado bajo la marca NooElec tiene un precio de $ 29,95 USD., acorde a la Figura 4. 5.

Figura 4. 5 NooElec (RTL-SDR)

Este dispositivo tiene una comunidad creciente y muy activa que han formado sitios tales RTL-SDR.com, donde existe amplia información de proyectos, y bases de conocimiento, apropiados para investigadores y estudiantes.

Debido a que el dispositivo no tiene capacidad de trasmisión de señales, no es posible construir un sistema de comunicaciones completo, por lo cual es necesario contar con otro equipo con capacidad de trasmisión para este fin, o como alternativa hacer la simulación de la trasmisión a través de señales grabadas previamente.

El RTL2832U de última generación cuenta con algoritmos de Realtek, que incluyen estimación de canal superior, rechazo de interfaz cocanal, recepción de canal de eco largo y cancelación de ruido impulsivo, y proporciona una solución ideal para una amplia gama de aplicaciones para PC-TV, lo cual le asegura un continuo crecimiento [15].

4.6. Adalm-Pluto

Tenemos también el dispositivo Adalm-Pluto, fabricado por la compañía Analog Devices, el cual como resumen de lo visto anteriormente, tiene un rango de operación nominal de 325 MHZ a 3,8 GHz. Pero este rango puede ser actualizado para abarcar 70 MHz a 6 GHz, esto se logra con el soporte dado por Matlab. Siendo posible esto debido a que los dispositivos tienen

componentes que tienen ciertos bloqueos de fábrica que pueden ser cambiados por software. Además, Tiene capacidad de trasmisión y recepción de señales en modo Full duplex. Aunque para hacer esto hay que tomar en cuenta los retardos acumulados a través de sus capas de hardware y software, por lo que se recomienda solo usarlo en una función simultáneamente.

El ancho de banda es de 20 MHz, con una velocidad de muestreo de 61,44 MSPS. La documentación actual es escasa, no obstante, la compañía Analog Devices, tiene un sitio base de conocimiento, tipo foro, donde sus especialistas de producto responden de manera gratuita a preguntas técnicas sobre el dispositivo. Fue creado con la finalidad de asistir en el aprendizaje, y existen algunos institutos y universidades que lo utilizan en sus laboratorios. El costo es relativamente bajo, el dispositivo tiene un precio sugerido por parte de Analog Devices de $249 USD. No obstante, es posible encontrarlo a un precio de $ 150 USD. en el distribuidor electrónico Digikey.com [16]. Ver Figura 4. 6.

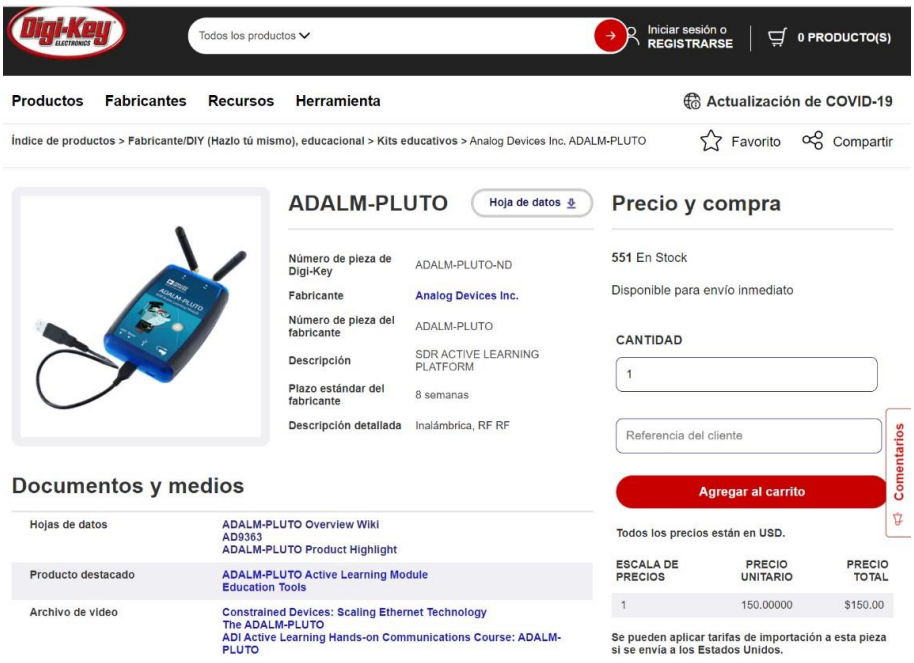

Figura 4. 6 Adalm Pluto

Para la implementación del trasmisor QPSK, se utilizaron 2 dispositivos Adalm Pluto, y se pudo poner en práctica conceptos importantes de la teoría de telecomunicaciones de datos, utilizando el soporte que ofrece la compañía

Matlab al producto. Además ofrece soporte a través de componentes de para GNU Radio.

Es el único dispositivo de bajo costo que tiene capacidad trasmisión y recepción y que además cuenta con el respaldo de una compañía grande en electrónica, como lo es Analog Devices, la cual tiene un staff de ingenieros que dedicados a impulsar el desarrollo y soporte del mismo. Por el precio del dispositivo, se convierte en una opción atractiva para el estudiante.

A continuación, se presenta la Tabla 4. 1, con el resumen de esta evaluación comparativa.

Equipo	**USRP B205mini-i**	**USRP-2920**	**Hack RF One**	**Noo Elec Rtl-Sdr**	**Adalm-Pluto**
Fabricante	Ettus	National Instruments	Great Scott Gadgets	NooElec	Analog Devices
Rango de operación	70 MHz – 6 GHz	50 MHz a 2.2 GHz	1 MHz- 6 GHz		325 MHZ a 3,8 GHz ampliable a 70 MHz – 6 GHz
Ancho de banda	56 MHz	20 MHz	20 Mhz		20 Mhz
Modos de operación	Trasmisión y recepción Full Duplex	Trasmisión y recepción Full Duplex	Trasmisión y recepción Half duplex	Solo Recepción	Trasmisión y recepción Full Duplex
Compatibilidad Software	Componentes de GNU Radio mantenido por Ettus Research	Lab view, Matlab, Simulink, GNU Radio	GNU Radio, SDR#	GNU Radio, Componentes Matlab y Simulink	GNU Radio, Componentes Matlab y Simulink
Interface USB	USB 3.0	USB 3.0		USB 2.0	USB 2.0
Disponibilidad de soporte	Amplia documentación, comunidades apoyadas por el fabricante	Soporte especializado y amplia documentación	Foros de internet	Foros de Internet	Comunidades apoyadas por soporte en foros del fabricante
Precio	$1,105 USD	Desde $ 3,855 USD.	$ 319 USD.	$ 29.95 USD.	$ 150 USD.

Tabla 4. 1 Resumen de la evaluación comparativa

CONCLUSIONES

En este trabajo se analizaron aspectos generales de la tecnología de radio definida por software y aspectos específicos del dispositivo de bajo costo Adalm-Pluto SDR, con su correspondiente experimentación, evaluación comparativa y conclusiones que se detallan a continuación:

- Actualmente es posible experimentar la recepción de señales con dispositivos muy económicos como el RTL-SDR, los cuales, por ser receptores modificados de televisión digital, tienen capacidades avanzadas de configuración y son fabricados en volúmenes grandes. Su precio inicia desde los $30 USD.
- Se puede experimentar la trasmisión y recepción de señales con dispositivos de costo considerado bajo como el Adalm-Pluto SDR, el cual tiene un precio de $150 USD.
- El dispositivo Adalm-Pluto SDR, permite verificar ideas estudiadas teóricamente, de manera práctica. Lo cual se comprobó a través de la implementación del sistema trasmisor y receptor QPSK. El cual tenía bloques completos de trasmisión y recepción.
- La compañía Analog Devices, fabricante del dispositivo Adalm-Pluto SDR, provee objetos de programación para Matlab, Simulink y GNU Radio, a diferencia de otros dispositivos de bajo costo que no gozan de un respaldo a nivel de software para sus dispositivos.
- El dispositivo maneja los parámetros prácticos para experimentación, no obstante, se evidenció retardos, propagados a través de sus capas de software, que interferían en la comunicación. Por lo que el dispositivo se consideraría solo para uso experimental y educativo.
- Para un uso profesional e intensivo, se recomienda otro tipo de productos, en lo posible certificados y diseñados para uso intensivo.
- El nivel de error BER en la comunicación, estuvo en el orden del 1,4%, el cual podría considerarse alto, no obstante, está fuera de los objetivos de este trabajo, profundizar en la mejora del nivel de error.

RECOMENDACIONES

Para futuros experimentos con la tecnología de radio definida por software y en particular el dispositivo Adalm-Pluto SDR, se detallan las siguientes recomendaciones:

- Para probar sistemas completos de trasmisión, es preferible utilizar computadores separados con alto nivel de procesamiento, ya que software como Matlab y Simulink son demandantes de intensivos recursos de procesamiento.
- Para un uso profesional, no se recomienda usar dispositivos de bajo costo, ya que podría haber inestabilidades y márgenes de error no aceptables.
- Se recomienda a los estudiantes de telecomunicaciones invertir en dispositivos SDR de bajo costo, ya que el beneficio supera con creces el costo del dispositivo.

BIBLIOGRAFÍA

[1] L. Guangpu y G. Yuchun, «The application of Matlab in communication theory,» *Procedia Engineering,* vol. 29, 2012.

[2] I. Pinar Domínguez y J. Murillo Fuentes, Laboratorio de Comunicaciones Digitales Radio Definida por Software, Sevilla: Universidad de Sevilla, 2011.

[3] Analog Devices, Inc, "analog.com," Analog Devices, Inc, 2020. [Online]. Available: https://www.analog.com/en/design-center/evaluation-hardware-andsoftware/evaluation-boards-kits/adalm-pluto.html#eb-overview. [Accessed 27 Julio 2020].

[4] R. Stewart, K. Barlee, D. Atkinson y L. Crockett, Software Defined Radio using MATLAB® & Simulink® and the RTL-SDR, Glasgow, Scotland: University of Strathclyde, 2015.

[5] P. Sowjanya y P. Satyanarayana, «Implementation of Transceiver module for SDR system using ADALM PLUTO platform,» *International Journal of Engineering & Technology,* pp. 279-284, 2018.

[6] A. Padilla Esquivel, Diseño de sistema de comunicaciones usando software defined radio (Tesis de Pregrado), Sevilla: Universidad de Sevilla, 2015.

[7] T. Collins, R. Gets, A. Wyglinski and D. Pu, Software defined radio for engineers, Analog devices, 2018.

[8] Analog Devices. Inc, «RF Agile Tranceiver Data Sheet AD9363,» 2016. [En línea]. Available: https://www.analog.com/en/products/ad9363.html.

[9] K. Wesolowski, Introduction to Digital Communication Systems, West Sussex: John Wiley & Sons, Incorporated, 2009.

[10] I. Hernáez Rioja , «Procesado Digital de la Señal en Comunicaciones. Universidad del País Vasco,» 2010. [En línea]. Available: https://aholab.ehu.eus/users/inma/psc/PSC20102011.pdf.

[11] National Instruments, «USRP Software Defined Radios,» 2020. [En línea]. Available: http://www.ni.com/pdf/product-flyers/usrp-software-defined-radio.pdf.

[12] Ettus Research, «Ettus Research a National Instruments brand,» 2020. [En línea]. Available: https://www.ettus.com/all-products/usrp-b205mini-i/.

[13] Great Scott Gadgets, «Hack One RF,» 2020. [En línea]. Available: https://greatscottgadgets.com/hackrf/one/.

[14] NooElec, «NooElec store,» 2020. [En línea]. Available: https://www.nooelec.com/store/sdr/sdr-receivers/nesdr-mini-two-plus.html.

[15] Realtek Semiconductor Corp., «Realtek product search RTL2832U,» Septiembre 2020. [En línea]. Available: https://www.realtek.com/en/products/communications-networkics/item/rtl2832u.

[16] Digikey Electronics, «Digikey Electronics.Indice de productos: Adalm-Pluto,» Septiembre 2020. [En línea]. Available: https://www.digikey.com/productdetail/es/analog-devices-inc/ADALM-PLUTO/ADALM-PLUTO-ND/6624230.

[17] Analog devices, «ADALM-PLUTO Overview,» 20 Agosto 2020. [En línea]. Available: https://wiki.analog.com/university/tools/pluto.

[18] J. Rodríguez de Haro, Análisis software y hardware del SDR HackRF One (Tesis de Pregrado), Granada: Universidad de Granada, 2017.

ANEXOS
ANEXO1
CÓDIGO DE INICIALIZACIÓN DEL TRASMISOR

```
function SimParams = plutoradioqpsktransmitter_init
%   Copyright 2017 The MathWorks, Inc.

%% General simulation parameters
SimParams.Rsym = 0.2e6;             % Symbol rate in Hertz
SimParams.ModulationOrder = 4;      % QPSK alphabet size
SimParams.Interpolation = 2;        % Interpolation factor
SimParams.Decimation = 1;           % Decimation factor
SimParams.Tsym = 1/SimParams.Rsym;  % Symbol time in sec
SimParams.Fs   = SimParams.Rsym * SimParams.Interpolation; % Sample rate

%% Frame Specifications
% [BarkerCode*2 | 'Hello world 000\n' | 'Hello world 001\n' ...];
SimParams.BarkerCode      = [+1 +1 +1 +1 +1 -1 -1 +1 +1 -1 +1 -1 +1];     % Bipolar Barker Code
SimParams.BarkerLength    = length(SimParams.BarkerCode);
SimParams.HeaderLength    = SimParams.BarkerLength * 2;                   % Duplicate 2 Barker codes to be as a header
SimParams.Message         = 'Hola mundo!';
SimParams.MessageLength   = length(SimParams.Message) + 5;                % 'Hello world 000\n'...
SimParams.NumberOfMessage = 100;                                          % Number of messages in a frame
SimParams.PayloadLength   = SimParams.NumberOfMessage * SimParams.MessageLength * 7; % 7 bits per characters
SimParams.FrameSize       = (SimParams.HeaderLength + SimParams.PayloadLength) ...
    / log2(SimParams.ModulationOrder);                                    % Frame size in symbols
SimParams.FrameTime       = SimParams.Tsym*SimParams.FrameSize;
```

```
%% Tx parameters
SimParams.RolloffFactor     = 0.5;                                        % Rolloff Factor of Raised Cosine Filter
SimParams.ScramblerBase     = 2;
SimParams.ScramblerPolynomial           = [1 1 1 0 1];
SimParams.ScramblerInitialConditions    = [0 0 0 0];
SimParams.RaisedCosineFilterSpan = 10; % Filter span of Raised Cosine Tx Rx filters (in symbols)
%% Message generation
msgSet = zeros(100 * SimParams.MessageLength, 1);
for msgCnt = 0 : 99
    msgSet(msgCnt * SimParams.MessageLength + (1 : SimParams.MessageLength)) = ...
        sprintf('%s %03d\n', SimParams.Message, msgCnt);
end
integerToBit = comm.IntegerToBit(7, 'OutputDataType', 'double');
SimParams.MessageBits = integerToBit(msgSet);
% Pluto transmitter parameters
SimParams.PlutoCenterFrequency      = 915e6;
SimParams.PlutoGain                 = 0;
SimParams.PlutoFrontEndSampleRate   = SimParams.Fs;
SimParams.PlutoFrameLength          = SimParams.Interpolation * SimParams.FrameSize;
% Simulation Parameters
SimParams.FrameTime = SimParams.PlutoFrameLength/SimParams.PlutoFrontEndSampleRate;
SimParams.StopTime  = 1000;
```

ANEXO2

CÓDIGO DE INICIALIZACIÓN DEL RECEPTOR

```
function SimParams = plutoradioqpskreceiver_init
%   Copyright 2017 The MathWorks, Inc.

%% General simulation parameters
SimParams.Rsym = 0.2e6;             % Symbol rate in Hertz
SimParams.ModulationOrder = 4;      % QPSK alphabet size
SimParams.Interpolation = 2;        % Interpolation factor
SimParams.Decimation = 1;           % Decimation factor
SimParams.Tsym = 1/SimParams.Rsym;  % Symbol time in sec
SimParams.Fs   = SimParams.Rsym * SimParams.Interpolation; % Sample rate

%% Frame Specifications
% [BarkerCode*2 | 'Hello world 000\n' | 'Hello world 001\n' ... | 'Hello world 099\n'];
SimParams.BarkerCode      = [+1 +1 +1 +1 +1 -1 -1 +1 +1 -1 +1 -1 +1];     % Bipolar Barker Code
SimParams.BarkerLength    = length(SimParams.BarkerCode);
SimParams.HeaderLength    = SimParams.BarkerLength * 2;                    % Duplicate 2 Barker codes to be as a header
SimParams.Message         = 'Hola Mundo!';
SimParams.MessageLength   = length(SimParams.Message) + 5;                 % 'Hello world 000\n'...
SimParams.NumberOfMessage = 100;                                           % Number of messages in a frame
SimParams.PayloadLength   = SimParams.NumberOfMessage * SimParams.MessageLength * 7; % 7 bits per characters
SimParams.FrameSize       = (SimParams.HeaderLength + SimParams.PayloadLength) ...
    / log2(SimParams.ModulationOrder);                                     % Frame size in symbols
SimParams.FrameTime       = SimParams.Tsym*SimParams.FrameSize;
```

```
%% Rx parameters
SimParams.RolloffFactor     = 0.5;                      % Rolloff Factor of Raised Cosine Filter
SimParams.ScramblerBase     = 2;
SimParams.ScramblerPolynomial           = [1 1 1 0 1];
SimParams.ScramblerInitialConditions    = [0 0 0 0];
SimParams.RaisedCosineFilterSpan = 10;                  % Filter span of Raised Cosine Tx Rx filters (in symbols)
SimParams.DesiredPower                  = 2;            % AGC desired output power (in watts)
SimParams.AveragingLength               = 50;           % AGC averaging length
SimParams.MaxPowerGain                  = 60;           % AGC maximum output power gain
SimParams.MaximumFrequencyOffset        = 6e3;
% Look into model for details for details of PLL parameter choice.
% Refer equation 7.30 of "Digital Communications - A Discrete-Time Approach" by Michael Rice.
K = 1;
A = 1/sqrt(2);
SimParams.PhaseRecoveryLoopBandwidth    = 0.01;         % Normalized loop bandwidth for fine frequency compensation
SimParams.PhaseRecoveryDampingFactor    = 1;            % Damping Factor for fine frequency compensation
SimParams.TimingRecoveryLoopBandwidth   = 0.01;         % Normalized loop bandwidth for timing recovery
SimParams.TimingRecoveryDampingFactor   = 1;            % Damping Factor for timing recovery
% K_p for Timing Recovery PLL, determined by 2KA^2*2.7 (for binary PAM),
% QPSK could be treated as two individual binary PAM,
% 2.7 is for raised cosine filter with roll-off factor 0.5
SimParams.TimingErrorDetectorGain       = 2.7*2*K*A^2+2.7*2*K*A^2;
SimParams.PreambleDetectorThreshold     = 0.8;
```

```
%% Message generation and BER calculation parameters
msgSet = zeros(100 * SimParams.MessageLength, 1);
for msgCnt = 0 : 99
    msgSet(msgCnt * SimParams.MessageLength + (1 : SimParams.MessageLength)) = ...
        sprintf('%s %03d\n', SimParams.Message, msgCnt);
end
integerToBit = comm.IntegerToBit(7, 'OutputDataType', 'double');
SimParams.MessageBits = integerToBit(msgSet);
% For BER calculation masks
SimParams.BerMask = zeros(SimParams.NumberOfMessage * length(SimParams.Message) * 7, 1);
for i = 1 : SimParams.NumberOfMessage
    SimParams.BerMask( (i-1) * length(SimParams.Message) * 7 + ( 1: length(SimParams.Message) * 7) ) = ...
        (i-1) * SimParams.MessageLength * 7 + (1: length(SimParams.Message) * 7);
end
% Pluto receiver parameters
SimParams.PlutoCenterFrequency      = 915e6;
SimParams.PlutoGain                 = 30;
SimParams.PlutoFrontEndSampleRate   = SimParams.Fs;
SimParams.PlutoFrameLength          = SimParams.Interpolation * SimParams.FrameSize;
% Experiment parameters
SimParams.PlutoFrameTime = SimParams.PlutoFrameLength / SimParams.PlutoFrontEndSampleRate;
SimParams.StopTime = 10;
```

ANEXO3

DATASHEET AD9363

THEORY OF OPERATION

GENERAL

The AD9363 is a highly integrated radio frequency (RF) transceiver capable of being configured for a wide range of applications. The device integrates all RF, mixed-signal, and digital blocks necessary to provide all transceiver functions in a single device. Programmability allows this broadband transceiver to be adapted for use with multiple communication standards, including FDD and TDD systems. This programmability also allows the device to interface to various BBPs using a single 12-bit parallel data port, dual 12-bit parallel data ports, or a 12-bit low voltage differential signaling (LVDS) interface.

The AD9363 also provides self calibration and AGC systems to maintain a high performance level under varying temperatures and input signal conditions. In addition, the device includes several test modes that allow system designers to insert test tones and create internal loopback modes to debug their designs during prototyping and optimize their radio configuration for a specific application.

RECEIVER

The receiver section contains all blocks necessary to receive RF signals and convert them to digital data that is usable by a BBP. Two independently controlled channels can receive signals from different sources, allowing the device to be used in multiple input, multiple output (MIMO) systems while sharing a common frequency synthesizer.

Each channel has three inputs that can be multiplexed to the signal chain, making the AD9363 suitable for use in diversity systems with multiple antenna inputs. The receiver is a direct conversion system that contains a low noise amplifier (LNA) followed by matched in-phase (I) and quadrature (Q) amplifiers, mixers, and band shaping filters that downconvert received signals to baseband for digitization. External LNAs can also be interfaced to the device, allowing designers the flexibility to customize the receiver front end for their specific application.

Gain control is achieved by following a preprogrammed gain index map that distributes gain among the blocks for optimal performance at each level. This gain control can be achieved by enabling the internal AGC in either fast or slow mode or by using manual gain control, allowing the BBP to make the gain adjustments as needed. Additionally, each channel contains independent RSSI measurement capability, dc offset tracking, and all circuitry necessary for self calibration.

The receivers include 12-bit, sigma-delta (Σ-Δ) ADCs and adjustable sample rates that produce data streams from the received signals. The digitized signals can be conditioned further by a series of decimation filters and a fully programmable 128-tap FIR filter with additional decimation settings. The sample rate of each digital filter block can also be adjusted by changing the decimation factors to produce the desired output data rate.

TRANSMITTER

The transmitter section consists of two identical and independently controlled channels that provide all digital processing, mixed-signal, and RF blocks necessary to implement a direct conversion system while sharing a common frequency synthesizer. The digital data received from the BBP passes through a fully programmable 128-tap FIR filter with interpolation options. The FIR output is sent to a series of interpolation filters that provide additional filtering and data rate interpolation prior to reaching the DAC. Each 12-bit DAC has an adjustable sampling rate. Both the I and Q channels are fed to the RF block for upconversion.

After being converted to baseband analog signals, the I and Q signals are filtered to remove sampling artifacts and provide band shaping, and then they are passed to the upconversion mixers. At this point, the I and Q signals are recombined and modulated on the carrier frequency for transmission to the output stage. The output stage provides attenuation control that provides a range of output levels while keeping the output impedance at 50 Ω. A wide range of attenuation adjustment with fine granularity is included to help designers optimize SNR.

Self calibration circuitry is included in the transmit channel to provide internal adjustment capability. The transmitter also provides a Tx monitor block that receives the transmitter output and routes it back through an unused receiver channel to the BBP for signal monitoring. The Tx monitor blocks are available only in TDD mode operation while the receiver is idle.

CLOCK INPUT OPTIONS

The AD9363 uses a reference clock provided by an external oscillator or clock distribution device (such as the AD9548) connected to the XTALN pin. The frequency of this reference clock can vary from 10 MHz to 80 MHz. This reference clock supplies the synthesizer blocks that generate all data clocks, sample clocks, and local oscillators inside the device.

SYNTHESIZERS

RF PLLs

The AD9363 contains two identical synthesizers to generate the required LO signals for the RF signal paths—one for the receiver and one for the transmitter. PLL synthesizers are fractional N designs that incorporate completely integrated VCOs and loop filters. In TDD mode, the synthesizers turn on and off as appropriate for the Rx and Tx frames. In FDD mode, the Tx PLL and the Rx PLL can be activated at the same time. These PLLs require no external components.