

PROYECTO FINAL DE CARRERA

Ingeniería de Telecomunicaciones

Implementación de terminales de Radio Cognitiva en la banda de TV

(Cognitive Radio Terminals Implementation on TV band)

Autor: Israel Perejil Sabaté

Director: Ferran Casadevall Palacio

23 de Febrero de 2016

AGRADECIMIENTOS

En primer lugar, quisiera expresar mi más sincero agradecimiento a mi tutor, Ferran Casadevall Palacio, por todas las horas dedicadas a ayudarme y a orientarme en la consecución y elaboración de este proyecto final de carrera, así como también por su constante apoyo para que pudiera sacar adelante con éxito este trabajo.

Gracias a Gaby, Víctor, Miri, Eric, Isa, Carles, Jose y Pau por echarme una mano (o dos, o tres...) siempre que han podido y, en general, a todos los del grupo “electro”. No se puede tener mejores amigos.

Gracias a todos los compañeros de la ETSETB que he ido conociendo a lo largo de mi andadura y con los que he forjado una bonita amistad.

Y por último, y más importante, gracias a mis padres. Gracias por tanta generosidad y por haber hecho de mí quien soy. Este proyecto final de carrera os lo dedico a vosotros.

RESUMEN

La aparición de nuevas tecnologías inalámbricas, unido a la ineficiente asignación del espectro por parte de los organismos oficiales encargados de ello, vislumbra un escenario de escasez espectral en un futuro próximo. En este contexto, la puesta en marcha de técnicas de acceso oportunista (dinámico) al espectro basadas en la Radio Cognitiva irrumpen como posible solución a tal panorama.

En este proyecto final de carrera se ha implementado una comunicación inalámbrica (*wireless*) entre dos transceptores destinada a poner de manifiesto las principales características de la Radio Cognitiva. Para ello, ha sido necesario configurar un sistema *Software Defined Radio* (SDR) formado por 2 PCs de propósito general, 2 USRPs N200 y 4 antenas VERT400, y utilizar el *software GNU Radio* como programa principal.

El sistema de comunicaciones implementado es un sistema de transmisión bidireccional no simultáneo (*half-duplex*), que utiliza como protocolo el método de *Stop&Wait*, y que opera en la banda de televisión (470-790 MHz). El sistema permite visualizar de forma experimental tres de las cuatro fases del Ciclo Cognitivo, formalmente: *Spectrum Awareness*, *Spectrum Selection* y *Spectrum Sharing*.

RESUM

L'aparició de noves tecnologies sense fils, unit a l'ineficient assignació de l'espectre per part dels organismes oficials encarregats d'aquest, albira un escenari d'escassetat espectral en un futur pròxim. En aquest context, la posta en marxa de tècniques d'accés oportunista (dinàmic) a l'espectre basades en la Ràdio Cognitiva irromp com a possible solució a aquest panorama.

En aquest projecte final de carrera s'ha implementat una comunicació sense fils (*wireless*) entre dos transceptors destinada a posar de manifest les principals característiques de la Ràdio Cognitiva. Per això, ha estat necessari configurar un sistema *Software Defined Radio* (SDR) format per 2 PCs de propòsit general, 2 USRPS N200 i 4 antenes VERT400, i utilitzar el *software GNU Radio* com a programa principal.

El sistema de comunicacions implementat és un sistema de transmissió bidireccional no simultani (*half-duplex*), que utilitza com a protocol el mètode *Stop&Wait*, i que opera a la banda de televisió (470-790 MHz). El sistema permet visualitzar de forma experimental tres de les quatre fases del Cicle Cognitiu, formalment: *Spectrum Awareness*, *Spectrum Selection* y *Spectrum Sharing*.

ABSTRACT

The emergence of new wireless technologies, together with the inefficient spectrum allocation taken by government agencies, suggests a scenario of spectrum scarcity in the near future. In this context, the initiation of opportunistic (dynamic) spectrum access techniques based on Cognitive Radio is a possible solution to this scenario.

In this final project a wireless communication has been made between two transceivers with the aim of highlighting the main features of Cognitive Radio. For that reason, it has been necessary to set up a Software Defined Radio (SDR) system composed of 2 PCs, 2 USRPs N200 and 4 antennas VERT400, and to use GNU Radio software as main program.

The communications system implemented is a half-duplex system, which uses Stop&Wait method as protocol, and operates on the television band (470-790 MHz). The system can experimentally display three of the four phases of Cognitive Cycle, formally: Spectrum Awareness, Spectrum Selection and Spectrum Sharing.

X

ÍNDICE GENERAL

1. Introducción	1
1.1 Motivación.....	1
1.2 Objetivos.....	2
1.3 Organización de la memoria.....	2
2. Estado del arte	4
2.1 La Radio Cognitiva.....	4
2.1.1 Definiciones.....	4
2.1.2 Arquitectura.....	7
2.1.3 El Ciclo Cognitivo	9
2.1.4 Acceso Dinámico al Espectro	10
2.1.4.1 <i>Spectrum Awareness</i>	12
2.1.4.2 <i>Spectrum Selection</i>	15
2.1.4.2.1 <i>Spectrum Analysis</i>	16
2.1.4.2.2 <i>Spectrum Decision</i>	16
2.1.4.3 <i>Spectrum Sharing</i>	17
2.1.4.4 <i>Spectrum Mobility</i>	21
2.1.5 Estandarización	22
2.2 <i>TV White Spaces</i> (TVWS).....	24
2.2.1 Contextualización	24
2.2.2 Concepto de <i>TV White Space</i>	26
2.3 <i>Software Defined Radio</i> (SDR)	27
2.3.1 Concepto	27
2.3.2 Sistemas <i>Software Defined Radio</i>	28
3. Equipo físico utilizado (<i>hardware</i>)	32
3.1 Ordenador de propósito general.....	32

3.2 Universal Software Radio Peripheral (USRP)	33
3.2.1 La placa madre (<i>motherboard</i>)	35
3.2.2 La placa hija (<i>daughterboard</i>)	37
3.3 Antenas	39
4. Equipo lógico utilizado (<i>software</i>).....	40
4.1 <i>GNU Radio</i>	41
4.1.1 <i>GNU Radio Companion</i> (GRC).....	43
5. Implementación de los terminales cognitivos.....	46
5.1 Identificación de las oportunidades de acceso al espectro	46
5.1.1 El Detector de Energía.....	47
5.1.2 Criterio de decisión: Canal Libre/Canal Ocupado.....	54
5.2 Asignación del espectro.....	57
5.3 Diseño del Sistema Digital	58
5.3.1 Transmisor	60
5.3.2 Receptor.....	71
5.3.3 Cabeceras en la transmisión.....	75
5.4 Esquema conceptual de los <i>scripts</i>	77
6. Conclusiones y trabajo futuro	80
Apéndices.....	83
A. Instalación y configuración del USRP N200 en Ubuntu	83
B. Instalación de <i>GNU Radio</i> en Ubuntu.....	90
C. Módulos, carpetas y archivos de <i>GNU Radio</i>	95
D. Construcción de Módulos y Bloques en <i>GNU Radio Companion</i>	98
Bibliografia	104

ÍNDICE DE FIGURAS

Figura 2.1.2-1 Concepto de Arquitectura de Radio Cognitiva	8
Figura 2.1.3-1 Ciclo Cognitivo.....	9
Figura 2.1.3-2 Ciclo Cognitivo simplificado.....	10
Figura 2.1.4-1 Modelos DSA: a) <i>interweave</i> ; b) <i>underlay</i> ; c) <i>overlay</i>	12
Figura 2.1.4.1-1 Ilustración del concepto de agujero espectral (<i>spectrum hole</i>).....	13
Figura 2.1.4.2.2-1 Ejemplo de mecanismo de decisión espectral	17
Figura 2.1.4.3-1 Infraestructura Centralizada.....	18
Figura 2.1.4.3-2 Infraestructura Distribuida	19
Figura 2.1.4.3-3 Compartición del espectro <i>Inter-Network</i> e <i>Intra-Network</i> en una Red Cognitiva	20
Figura 2.1.5-1 Arquitectura del protocolo IEEE 802.22.....	24
Figura 2.2.1-1 Banda espectral asignada a la televisión y Dividendo Digital	25
Figura 2.2.2-1 TVWS en edificio D4 del Campus Nord (UPC).....	27
Figura 2.3.2-1 Sistema SR ideal.....	28
Figura 2.3.2-2 Arriba, estructura SDR de un Transmisor. Abajo, estructura SDR de un Receptor.....	30
Figura 2.3.2-3 Gráfico orientativo para la elección de ASICs, FPGAs o DSPs	31
Figura 3-1 Arquitectura del sistema SDR implementado.....	32
Figura 3.2-1 Panel frontal del USRP N200	35
Figura 3.2.1-1 Arquitectura de la <i>motherboard</i> del USRP N200.....	35
Figura 3.2.1-2 Ejecutando <code>./uhd_usrp_probe</code> en un terminal de Ubuntu, se obtiene información sobre los componentes del USRP	37
Figura 3.2.2-1 Rango de frecuencias de operación alcanzados por modelos de <i>daughterboards</i>	38
Figura 3.3-1 Antena VERT400.....	39
Figura 4.1-1 Arquitectura <i>GNU Radio</i>	42
Figura 4.1.1-1 Abriendo el programa <i>GNU Radio Companion</i>	43
Figura 4.1.1-2 Apariencia de la interfaz GRC	44
Figura 4.1.1-3 Parámetros por defecto del bloque <i>Signal Source</i>	44
Figura 5.1.1-1 Bloques que participan en el programa <i>Sectrum_Awareness.py</i>	47
Figura 5.1.1-2 Ejemplo del espectro de una señal al pasar por los bloques <i>Front-end</i> y ADC	48
Figura 5.1.1-3 Ejemplo del paso de una señal a través de los bloques ADC y FFT	50
Figura 5.1.1-4 Ejemplo de los casos a), b) y c)	51
Figura 5.1.1-5 Respuesta en frecuencia del módulo y la fase de un filtro paso bajo.....	52

Figura 5.1.2-1 A la izquierda, criterio de decisión con 2 estados. A la derecha, criterio de decisión con 4 estados	55
Figura 5.1.2-2 Arriba, representación gráfica del espectro analizado con el programa <i>Spectrum_Awareness.py</i> . El eje horizontal representa el número de la muestra y el eje vertical el nivel de potencia de la señal (en dBm). Abajo, lista de la ocupación de los canales de la banda de TV según el criterio del programa <i>Spectrum_Awareness.py</i>	57
Figura 5.2-1 La elección del canal la toma el usuario cognitivo	58
Figura 5.3-1 Escenario del sistema de comunicación inalámbrico. Foto tomada en el laboratorio D4112 de la UPC.....	59
Figura 5.3.1-1 <i>Flow graph</i> de la cadena de transmisión.....	61
Figura 5.3.1-2 Parámetros del bloque <i>File Soure</i>	61
Figura 5.3.1-3 Asignación de los colores de los puertos de un bloque según el tipo de dato	62
Figura 5.3.1-4 Trama generada en el <i>Packet Encoder</i>	62
Figura 5.3.1-5 Parámetros del bloque <i>Packet Encoder</i>	64
Figura 5.3.1-6 A la izquierda, símbolos en una modulación MSK. A la derecha símbolos en una modulación GMSK.....	65
Figura 5.3.1-7 Parámetros del bloque <i>GSMK Mod</i>	66
Figura 5.3.1-8 Parámetros del bloque <i>UHD: USRP Sink</i>	67
Figura 5.3.1-9 Parámetros del bloque <i>Multiply Const</i>	68
Figura 5.3.1-10 Parámetros del bloque <i>WX GUI FFT Sink</i>	69
Figura 5.3.1-11 Densidad espectral de potencia de la señal GMSK	70
Figura 5.3.1-12 Parámetros del bloque <i>WX GUI Waterfall Sink</i>	70
Figura 5.3.1-13 Espectrograma de la señal GMSK transmitida	70
Figura 5.3.2-1 <i>Flow graph</i> de la cadena de transmisión.....	71
Figura 5.3.2-2 Parámetros del bloque <i>UHD:USRP Source</i>	72
Figura 5.3.2-3 Arriba, el espectrograma de la señal recibida. Abajo la densidad espectral de potencia de la señal recibida	72
Figura 5.3.2-4 Parámetros del bloque <i>GMSK Demod</i>	73
Figura 5.3.2-5 Parámetros del bloque <i>Packet Decoder</i>	74
Figura 5.3.2-6 Parámetros del bloque <i>File Sink</i>	75
Figura 5.3.3-1 A la izquierda, contenido del archivo transmitido. A la derecha, contenido del archivo recibido	76
Figura 5.3.3-2 Añadiendo las cabeceras @INI*y &FIN\$ con el programa GHex.....	77
Figura 5.4-1 Esquema conceptual del sistema de Radio Cognitiva	78
Figura 5.4-2 El receptor muestra por pantalla un resumen de los paquetes que va recibiendo	79
Figura A-1 Configuración de la red <i>Ethernet</i>	85
Figura A-2 Ejemplo de información proporcionada por el comando <i>uhd_find_devices</i>	85
Figura A-3 Ejemplo de información proporcionada por el comando <i>uhd_usrp_probe</i>	85

Figura A-4 A la izquierda, ejemplo del estado de los LEDs cuando el USRP transmite. A la derecha, ejemplo del estado de los LEDs cuando el USRP recibe.....	86
Figura A-5 Comandos utilizados para cargar las imágenes en el USRP	87
Figura A-6 Ejemplo del comando <i>sudo uhd_images_downloader</i>	87
Figura A-7 Interfaz del programa <i>usrp_n2xx_net_burner_gui.py</i>	87
Figura A-8 Configuración de la dirección IP del dispositivo y de la red <i>Ethernet</i> , para el segundo USRP.....	89
Figura B-1 Interfaz gráfica del programa <i>app_store.py</i>	94
Figura D-1 Creación de un módulo OOT	99
Figura D-2 Opciones del comando <i>gr_moodtol</i>	99
Figura D-3 Contenido en la carpeta del módulo <i>Ejemplos</i>	99
Figura D-4 Creación de un bloque de procesado	100
Figura D-5 Edición del archivo <i>bloque_ejemplo1_impl.cc</i>	101
Figura D-6 Edición del archivo <i>Ejemplos_bloque_ejemplo1.xml</i>	102
Figura D-7 <i>bloque_ejemplo1</i> en <i>GNU Radio Companion</i>	103

ÍNDICE DE TABLAS

Tabla 2.1.4.1-1 Comparación de las técnicas del <i>Spectrum Awareness</i>	14
Tabla 2.1.5-1 Estandarización de la Radio Cognitiva (IEEE SCC 41)	23
Tabla 2.1.5-2 Estandarización de la Radio Cognitiva (IEEE 802).....	23
Tabla 3.2-1 Comparación entre modelos USRP de sus características más relevantes	34
Tabla C-1 Clasificación de los módulos de <i>GNU Radio</i>	96
Tabla C-2 Listado de carpetas de los módulos de <i>GNU Radio</i>	96
Tabla C-3 Listado de archivos de los módulos de <i>GNU Radio</i>	97

ACRÓNIMOS

A

ACK: *Acknowledgment*

ADC: *Analog-to-Digital Converter*

API: *Application Program Interface*

ASIC: *Application-Specific Integrated Circuit*

B

BB: *Baseband*

C

CPLD: *Complex Programmable Logic Device*

CRC: *Cyclic Redundancy Check*

D

D8PSK: *8-Ary Differential Phase-Shift Keying*

DAC: *Digital-to-Analog Converter*

DBPSK: *Differential Binary Phase Shift Keying*

DDC: *Digital Downconverter*

DFT: *Discrete Fourier Transform*

DQPSK: *Differential Quadrature Phase Shift Keying*

DSP: *Digital Signal Processor*

DUC: *Digital Up-Converter*

DVB-H: *Digital Video Broadcast - Handheld*

E

ECMA: *European Computer Manufacturers Association*

ETSI: *European Telecommunications Standards Institute*

F

FCC: *Federal Communications Commission*

FFT: *Fast Fourier Transform*

FPGA: *Field-Programmable Gate Array*

G

GRC: *GNU Radio Companion*

GRCM: *Grupo de Investigación en Comunicaciones Móviles*

GRCon: *GNU Radio Conference*

GSMK: *Gaussian Minimum Shift Keying*

I

IEEE: *Institute of Electrical and Electronics Engineers*

IF: *Intermediate frequency*

IP: *Internet Protocol*

ISI: *Inter-Symbol Interference*

ISM: *Industrial, Scientific and Medical*

ITU: *International Telecommunication Union*

L

LAN: *Local Area Network*

LED: *Light Emitting Diode*

LOS: *Line Of Sight*

M

MAC: *Media Access Control*

MAN: *Metropolitan Area Network*

MIMO: *Multiple Input Multiple Output*

MSK: *Minimum Shift Keying*

N

NLOS: *No Line of Sight*

NLPS: *Side Lobe Level*

NTIA: *National Telecommunication and Information Agency*

O

OOT: *Out-Of-Tree*

P

PC: *Personal Computer*

PGA: *Programmable Gain Amplifier*

PU: *Primary User*

Q

QAM8 : *8-Ary Quadrature Amplitude Modulation*

QAM16: *16-Ary Quadrature Amplitude Modulation*

QAM64: *64-Ary Quadrature Amplitude Modulation*

QAM256: *256-Ary Quadrature Amplitude Modulation*

R

RAM: *Random-Access Memory*

REM: *Radio Environment Map*

RF: *Radio Frequency*

S

SDR: *Software Defined Radio*

SMA: *SubMiniature version A*

SU: *Secundary User*

SWIG: *Simplified Wrapper and Interface Generator*

T

TDT: *Televisión Digital Terrestre*

TVWS: *TV White Space*

U

UDP: *User Datagram Protocol*

UHD: *USRP Hardware Driver*

UHF: *Ultra High Frequency*

UPC: *Universitat Politècnica de Catalunya*

USB : *Universal Serial Bus*

USRP: *Universal Software Radio Peripheral*

V

VHDL: *VHSIC Hardware Description Language*

VHSIC: *Very High Speed Integrated Circuit*

W

WRAN: *Wireless Regional Access Network*

1. Introducción

1.1 Motivación

En los últimos años, y debido al auge de las comunicaciones inalámbricas, se ha producido un crecimiento exponencial del número de usuarios y volumen de tráfico de datos gestionado por dichos sistemas, lo que conlleva un uso exhaustivo del espectro radioeléctrico por debajo de los 3 GHz. En este contexto, y ante la creciente escasez de espectro radioeléctrico libre, los expertos se plantean y debaten si se está haciendo un uso correcto, o no, del mismo.

Como es sabido, el espectro radioeléctrico es un bien de dominio público cuya titularidad, gestión, planificación, administración y control corresponde al Estado. Cada país en su legislación (más precisamente, en los denominados *Cuadros Nacionales de Atribución de Frecuencias*) delimita el uso del espectro mediante la descripción de la tecnología utilizada y el tipo de servicio ofrecido en las distintas bandas de frecuencia que lo integran.

Uno de los puntos que parece ser clave para un mejor aprovechamiento del uso del espectro reside en constatar que, hasta ahora y en todos los países, se utiliza una política de asignación de espectro fija en la práctica totalidad del espectro disponible. Es decir, las distintas bandas de frecuencia y radiocanales asociadas a los diversos servicios de radiocomunicación son asignadas a empresas, usuarios y/o servicios mediante autorización, permiso o licencia durante un periodo de tiempo largo (años habitualmente) y en amplias regiones geográficas.

Esta situación provoca que, teóricamente, ya no haya banda libre en frecuencias inferiores a 3 GHz, que es el rango de frecuencia más interesante del espectro tanto por lo que respecta a simplicidad tecnológica como a proporcionar zonas de cobertura razonablemente grandes. Sin embargo, en realidad ocurre que, por debajo de 3 GHz, hay bandas que se utilizan muy intensivamente y que se encuentran prácticamente congestionadas y otras, en cambio, que están parcial o totalmente desocupadas la mayor parte del tiempo, es decir, están desaprovechadas. Por otra parte, aunque más allá de 3 GHz sí que hay bandas libres, trabajar a decenas de GHz proporciona peores condiciones por lo que

respecta a pérdidas de propagación y cobertura, y además los equipos son más onerosos.

Por tanto, cabe concluir que la explotación de una parte relevante del espectro radioeléctrico asignado bajo licencia se realiza de forma ineficiente. Frente a esta incapacidad evidente de gestionar de forma eficiente el espectro, es necesario introducir un nuevo paradigma de gestión del espectro que permita, no solo mejorar su utilización, sino que dé cabida a nuevas tecnologías y servicios inalámbricos. Es, en el contexto de un acceso dinámico al espectro, donde entra en juego el concepto de Radio Cognitiva y el desarrollo de transceptores *Software Defined Radio* como los diseñados en este proyecto fin de carrera.

1.2 Objetivos

Este proyecto tiene por finalidad cumplir tres puntos esenciales. En primer lugar, explicar qué es la Radio Cognitiva y comprender cuáles son sus principios. En segundo lugar, proporcionar una primera aproximación a dicha tecnología, implementando dos transceptores capaces de establecer una comunicación inalámbrica (*Wireless*) que utilice un hueco espectral en la banda de TV o TVWS (*TV White Space*). Para ello se dispone de dos dispositivos USRP N200 con placas WBX que cubren el rango de los 50 MHz a los 2.2 GHz, que serán la base sobre la que se implementarán los dos transceptores. Y finalmente, el tercer objetivo es documentar, de manera exhaustiva, el *software* utilizado para llevar a cabo dicha comunicación (*GNU Radio*), con la finalidad de agilizar y facilitar su manejo a aquellos posibles usuarios que vayan a utilizar la herramienta en un futuro.

1.3 Organización de la memoria

La memoria de este proyecto final de carrera está estructurada en 6 capítulos y 4 apéndices:

- Capítulo 1: Introducción.

Se explica la situación actual del problema espectral y la importancia que tiene solucionar dicho problema. A partir de estos elementos se describen los objetivos del proyecto.

- Capítulo 2: Estado del arte.

Se describe el ámbito de la Radio Cognitiva, ofreciendo definiciones, investigando sobre su arquitectura y sus principios, documentando el conocimiento acumulado hasta el momento y sintetizando toda la información. Además se presentan los sistemas de *Software Defined Radio* (SDR) que, junto a los TVWS, conforman la piedra angular de este proyecto final de carrera.

- Capítulo 3: Equipo físico utilizado (*hardware*).

Se describen los componentes físicos que forman parte del sistema SDR implementado: el ordenador, el USRP y las antenas.

- Capítulo 4: Equipo lógico utilizado (*software*).

Se describen los programas informáticos empleados tanto para poner en marcha el equipo físico (*software UHD*) como para llevar a cabo la implementación del sistema SDR (entorno *GNU Radio*).

- Capítulo 5: Implementación de los terminales cognitivos.

Se explica en detalle los procedimientos específicos de la tecnología de la Radio Cognitiva desarrollados en este proyecto. En concreto, se detalla la implementación del detector de energía, que permite al usuario cognitivo identificar las oportunidades espectrales en la banda de televisión; posteriormente se presentan los esquemas diseñados para realizar la decisión espectral, para finalmente pasar a describir los *scripts* creados para establecer el protocolo de comunicación entre los dos dispositivos USRPs que implementan el sistema de comunicación *wireless*.

- Capítulo 6: Conclusiones y trabajo futuro.

Este capítulo hace una recapitulación del contenido del documento, se evalúan los resultados y se comentan las posibles líneas futuras para el caso de que se quiera seguir indagando posteriormente en el tema de la Radio Cognitiva.

2. Estado del arte

En este capítulo se da una revisión de la tecnología Radio Cognitiva y de sus técnicas de acceso dinámico al espectro y, además, se introduce al lector en los conceptos de *TV-White Spaces* y en la tecnología denominada *Software Defined Radio*.

2.1 La Radio Cognitiva

2.1.1 Definiciones

De una manera sencilla y, a la vez, breve, se podría definir la Radio Cognitiva como:

“Radio inteligente capaz de aprovechar de forma oportunista banda desocupada del espectro radioeléctrico”.

Sin embargo, atribuir una facultad propia de los seres vivos como es la inteligencia a algo inanimado provoca cierta inexactitud. Desde que surgió el término de Radio Cognitiva se han propuesto varias definiciones, algunas muy precisas. A continuación se exponen unas cuantas:

El creador de la idea original y el que acuñó el término de Radio Cognitiva fue Joseph Mitola III y la definió como:

“El ámbito tecnológico en el cual las PDAs inalámbricas y las redes relacionadas son, en términos computacionales, lo suficientemente inteligentes con respecto a los recursos de radio y las correspondientes comunicaciones de terminal a terminal como para detectar las necesidades eventuales de comunicación del usuario en función del contexto de uso y proporcionarle los recursos de radio y servicios inalámbricos más adecuados a sus necesidades.” [1]

Más tarde, Simon Haykin revisó el concepto, y definió la Radio Cognitiva como:

“Un sistema de comunicaciones inalámbricas inteligente que es consciente de su entorno y emplea la metodología ‘comprender y construir’ para aprender de su

entorno y adaptar su estado interno a las variaciones estadísticas presentes en los estímulos de radiofrecuencia de entrada, haciendo los correspondientes cambios en ciertos parámetros de operación (por ejemplo, la potencia de transmisión, la frecuencia portadora y el tipo de modulación) en tiempo real; con dos objetivos fundamentales: hacer un uso eficiente del espectro y proporcionar una comunicación altamente confiable.” [2]

El FCC, por su parte, la definió basándose esencialmente en el funcionamiento del transmisor:

“Una radio que puede cambiar sus parámetros de transmisión en base a la interacción con el entorno en el que opera.” [3]

La NTIA, el principal organismo regulador del espectro en Estados Unidos, también propuso su definición:

“Una radio o sistema que detecta su entorno electromagnético operativo y puede autónomamente y dinámicamente ajustar sus parámetros de operación radio para modificar el funcionamiento del sistema como, por ejemplo, mitigar la interferencia, proporcionar acceso al “mercado” secundario o facilitar la interoperabilidad y maximizar la capacidad.” [4]

La Unión Internacional de Telecomunicaciones, en el segmento destinado a la normalización de las Radiocomunicaciones, también conocido como ITU-R, dijo que es:

“Una radio o sistema que detecta y es consciente de su entorno de operación y en consecuencia puede ajustar de forma dinámica y autónoma sus parámetros de operación.” [5]

Por otra parte, el IEEE, propuso lo siguiente:

“Un transmisor/receptor de radiofrecuencia que está diseñado para detectar de forma inteligente si un segmento particular del espectro radioeléctrico está actualmente en uso, y para saltar dentro y fuera de él instantáneamente, siempre que el tiempo de desocupación del espectro lo permita, sin que haya ningún tipo de interferencia con las transmisiones de otros usuarios autorizados.” [6]

Del mismo modo, el *SDR Forum*, cuyo nombre actual es *Wireless Innovation Forum*, la define en su página web como:

“La radio en la cual los sistemas de comunicación son conscientes de su estado interno y del entorno, como serían la ubicación y la utilización del espectro de radiofrecuencia en ese lugar y, además, pueden tomar decisiones sobre su comportamiento operativo confrontando esta información con los objetivos predefinidos.” [7]

Y, para finalizar, el grupo *IEEE 1900.1* la define como:

“Un tipo de radio que puede detectar y razonar de forma autónoma sobre su entorno y adaptarse al mismo. [...] Esta radio podría emplear técnicas de representación del conocimiento, razonamiento autónomo y mecanismos de aprendizaje automático en los procesos de establecimiento, mantenimiento y finalización de una comunicación o en los mecanismos de interconexión con otras radios. [...] Los terminales de Radio Cognitiva pueden ser entrenados para ajustar sus parámetros de funcionamiento de forma dinámica y autónoma.” [8]

Una vez vistas las descripciones de lo que entienden algunos autores y organismos oficiales por Radio Cognitiva, podemos concluir este apartado dando una última definición, que sea lo más precisa posible y que combine las características que se han ido mencionando. Así pues, nosotros entendemos por Radio Cognitiva a:

“Dispositivos inalámbricos totalmente programables que pueden detectar su entorno y que son capaces de adaptar dinámicamente una serie de funcionalidades, como por ejemplo: el método de acceso al canal, la forma de onda utilizada para la transmisión, los protocolos de red, el uso del espectro necesario para no generar interferencias a otros dispositivos, etc.”

2.1.2 Arquitectura

Según [9], en un sistema de Radio Cognitiva hay dos grandes subsistemas: una unidad cognitiva que toma decisiones basadas en varios inputs o entradas (internas y externas) y una unidad SDR (*Software Defined Radio*) flexible, cuyo software operativo proporciona un rango de posibles modos de funcionamiento. Además, también es común incluir un subsistema de detección del espectro para medir las señales que hay en el entorno radio y así determinar la presencia de otros servicios o usuarios.

Con estos tres subsistemas se logran tres aspectos fundamentales que distinguen a una Radio Cognitiva [2]:

1. Capacidad cognitiva
2. Capacidad de aprendizaje y adaptación
3. Capacidad de auto-reconfiguración.

Por capacidad cognitiva se entiende a la habilidad de obtener información del entorno y del propio estado interno del sistema a través de múltiples “sensores”¹. Mientras que la capacidad de aprendizaje y adaptación permite utilizar esta información para ajustar, dinámicamente y autónomamente, los parámetros de operación a través de mecanismos de auto-reconfiguración, con el objetivo de optimizar las prestaciones del sistema.

Es importante señalar que las funcionalidades antes indicadas no tienen por qué definirse necesariamente en un solo dispositivo o subsistema del mismo, sino que se pueden incorporar componentes que se extiendan a través de una red completa. Esto quiere decir que las tres capacidades mencionadas anteriormente pueden estar distribuidas a través de múltiples capas de protocolos y dispositivos en una red. Como resultado, se refiere a menudo a la Radio Cognitiva como a un sistema de Radio Cognitiva o a una red Cognitiva.

En una arquitectura ideal de Radio Cognitiva encontrariamos el siguiente conjunto de componentes funcionales [10]:

1. Funciones de interfaz de usuario.

¹ Aquí el concepto “sensores” se entiende de forma amplia, es decir, cualquier equipo electrónico capaz

2. Funciones sensoriales de detección del entorno.
3. Aplicaciones del sistema.
4. Funciones de *Software Defined Radio*.
5. Funciones cognitivas.

Entre cada uno de estos componentes funcionales se establecen varias interfaces a través de las cuales se produce el intercambio de datos y señales de control que definen el funcionamiento del sistema.

En la figura 2.1.2-1 se muestra un ejemplo del concepto de arquitectura de Radio Cognitiva. En dicha figura, vemos como la unidad cognitiva se separa en dos bloques. El primero, llamado “*Cognitive Engine*”, tiene la función de encontrar una solución u optimizar las prestaciones del sistema basándose en los inputs o entradas recibidas que definen el propio estado interno del sistema y el estado del entorno radio en el que opera este dispositivo. El segundo bloque, llamado “*Policy Engine*”, es utilizado para asegurar que la solución proporcionada por el primer bloque cumpla con las normas regulatorias.

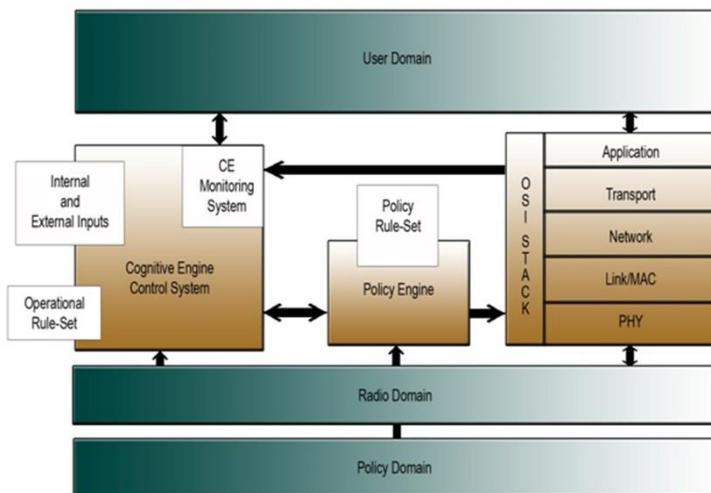


Figura 2.1.2-1. Concepto de Arquitectura de Radio Cognitiva. Extraída de [7].

2.1.3 El Ciclo Cognitivo

Joseph Mitola III, en su tesis doctoral [11], estima que hay nueve niveles en el incremento de la capacidad cognitiva de un sistema:

En el nivel 0, el sistema no tiene capacidad de razonamiento basado en modelo alguno. Representa a un sistema *Software Defined Radio* básico (véase apartado 2.4). En el nivel 1, el sistema es consciente del entorno que le rodea, mientras que, en el nivel 2, el sistema es consciente de lo que su usuario desea. En el nivel 3, el sistema tiene conocimiento tanto de sus componentes como de los de distintas redes en las que pueda operar. En el nivel 4, el sistema ya es capaz de analizar la situación (niveles 2 y 3). En el nivel 5, el sistema es capaz de negociar con otros dispositivos radio, mediante algún protocolo. Finalmente, en los niveles 6, 7 y 8, el sistema logra asumir el mayor grado de inteligencia.

Con el objetivo de explicar el proceso por el cual una Radio Cognitiva consigue estos niveles de funcionalidad, Mitola creó el Ciclo Cognitivo (figura 2.1.3-1). El Ciclo Cognitivo representa una máquina de estados de las diferentes etapas del proceso cognitivo.

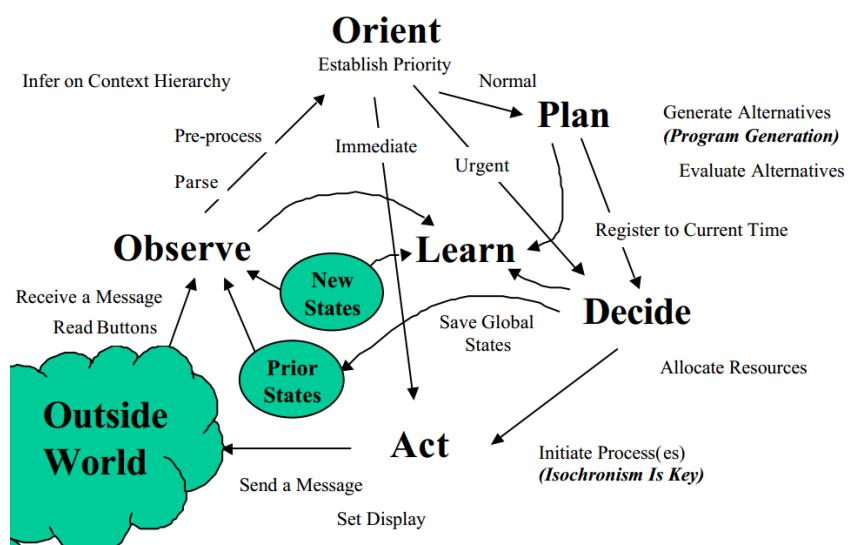


Figura 2.1.3-1 Ciclo Cognitivo. Extraída de [11].

El funcionamiento del Ciclo Cognitivo es el siguiente: en primer lugar, por medio de la observación (*Observe*), el sistema recibe información de su entorno (*Outside World*). Posteriormente, esta información se analiza y se pre-procesa

(*Orient*) con el objetivo de establecer una prioridad o relevancia (normal, urgente o inmediata). Según la prioridad establecida, el sistema genera y evalúa alternativas (*Plan*), elige la que considera más adecuada (*Decide*) y, una vez asignados los recursos necesarios, inicia el proceso y la lleva acabo (*Act*). Por consiguiente, se producen cambios en el entorno (*Outside World*) y el sistema aprovecha las observaciones y decisiones tomadas (*Prior States*) para aprender (*Learn*). En este proceso, el sistema es capaz de mejorar, logrando crear nuevos estados (*New States*), generando nuevas alternativas y definiendo nuevos criterios de evaluación.

Sin embargo, esta propuesta inicial de Ciclo Cognitivo se ha ido modificando por diversos autores para adaptarla al concepto de Radio Cognitiva como tecnología de acceso dinámico al espectro, véase la figura 2.1.3-2. De esta manera, el Ciclo Cognitivo para el acceso dinámico al espectro incluye el análisis y detección del espectro (*Spectrum Sensing* o *Spectrum Awareness*), la selección de las bandas de frecuencias que más se adecuen a los requerimientos de los usuarios (*Spectrum Decision*), la coordinación del acceso al espectro con otros usuarios (*Spectrum Sharing*) y la movilidadpectral de las frecuencias usadas cuando son utilizadas por los usuarios autorizados (*Spectrum Mobility*).

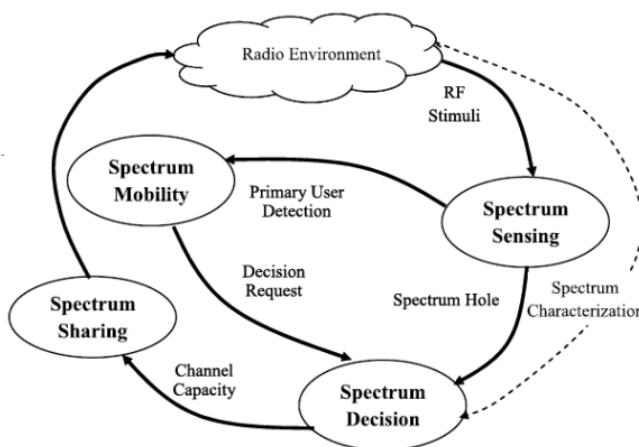


Figura 2.1.3-2 Ciclo Cognitivo simplificado. Extraída de [12].

2.1.4 Acceso Dinámico al Espectro

En función de las bandas de frecuencias utilizadas, pueden distinguirse entre dos modelos de acceso dinámico al espectro: modelo de acceso libre y modelo de acceso jerárquico [10].

En el modelo de acceso libre, el uso del espectro queda delimitado a aquellas bandas de frecuencia que están abiertas a todo el mundo sin necesidad de licencias. En dichas bandas, los usuarios tienen los mismos derechos, pero han de respetar las regulaciones impuestas como, por ejemplo, los niveles de potencia transmitida, las frecuencias portadoras o los límites espectrales. Un ejemplo de bandas que entrarían en el modelo de acceso libre serían las bandas ISM.

En el modelo de acceso jerárquico se distinguen dos tipos de usuarios: usuarios primarios (PU), aquellos que disponen de licencia para explotar el espectro, y usuarios secundarios (SU), aquellos que no disponen de ningún tipo licencia y que, por consiguiente, tienen restringido el uso del espectro correspondiente a bandas que requieren autorización. Para el acceso de los usuarios secundarios al espectro asignado legalmente a los usuarios primarios, pueden distinguirse tres tipos de estrategias: *interweave*, *underlay* y *overlay*.

En el esquema *interweave*, los usuarios secundarios tienen la capacidad de identificar porciones disponibles del espectro, comúnmente denominados “agujeros espectrales” (*spectrum holes* o *white spaces*), que utilizan de manera oportunista sin interferir a los usuarios primarios [13]. En el esquema *underlay*, los usuarios primarios y secundarios operan concurrentemente, sin embargo estos últimos han de restringir su potencia de transmisión con el objetivo de que la interferencia en el receptor primario se encuentre por debajo de un umbral preestablecido [13]. Además, para que sea posible la coexistencia con los usuarios primarios, los usuarios secundarios han de ser capaces de estimar el nivel de interferencia generado en los receptores primarios. En el esquema *overlay*, se permite la transmisión simultánea de los usuarios primarios y secundarios en las mismas bandas de frecuencia y a niveles comparables de potencia. Este hecho se logra gracias a la combinación de procedimientos de comunicación cooperativa y técnicas de manejo de interferencias [14].

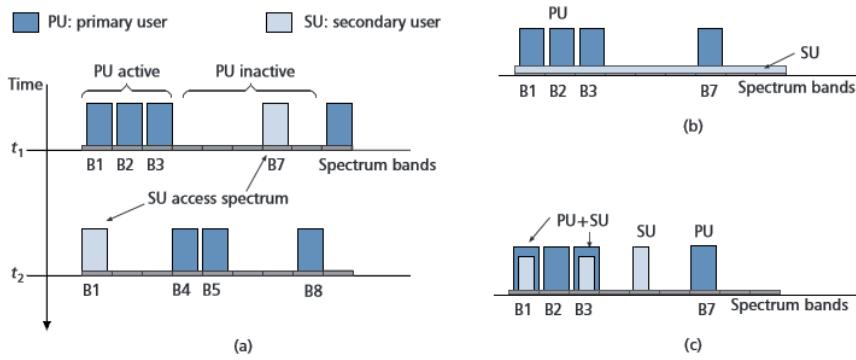


Figura 2.1.4-1 Modelos DSA: a) *interweave*; b) *underlay*; c) *overlay*. Extraída de [13].

Aunque estas distinciones son útiles desde el punto de vista teórico, no todo el modelo de acceso dinámico al espectro se puede únicamente clasificar como *underlay*, *interwave* u *overlay*. En sentido general, el acceso dinámico al espectro requiere de las cuatro fases del ciclo cognitivo:

1. *Spectrum Awareness*: Determinar qué partes de una porción del espectro de interés están disponibles y detectar la presencia de usuarios primarios cuando el usuario secundario opera en una banda licenciada.
2. *Spectrum Selection*: Seleccionar el mejor canal disponible.
3. *Spectrum Sharing*: Coordinar el acceso al canal en el que se está operando con otros usuarios secundarios.
4. *Spectrum Mobility*: Desocupar el canal cuando se detecta el acceso a dicho canal por parte de un usuario primario.

2.1.4.1 *Spectrum Awareness*

El Conocimiento del Espectro (o *Spectrum Awareness*) es la función responsable de la obtención de información relevante sobre el entorno radio. Con este fin, las radios cognitivas deben:

- 1) Rastrear de forma inteligente el espectro radioeléctrico buscando bandas de frecuencia no utilizadas (*spectrum holes* o *white spaces*). Estas bandas de frecuencias no utilizadas cambian dinámicamente tanto en tiempo como en espacio (véase figura 2.1.4-1-1).
- 2) Detectar la aparición de usuarios primarios en huecos espectrales utilizados de forma oportunista por usuarios secundarios. Es decir, implementar un control de interferencias.

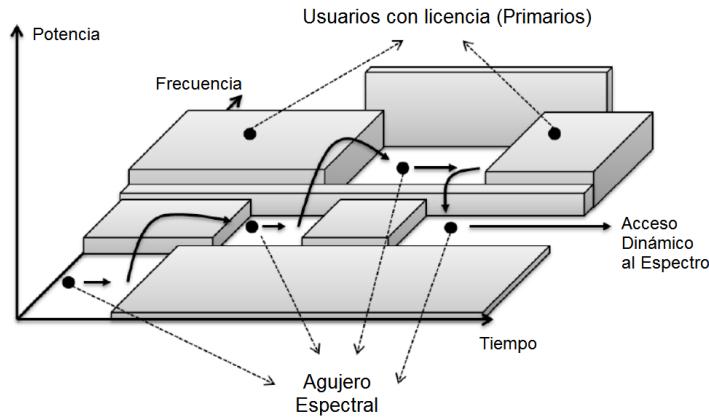


Figura 2.1.4.1-1 Ilustración del concepto de agujero espectral (*spectrum hole*). Extraída de [15].

Las técnicas empleadas por una radio cognitiva para obtener información sobre el espectro pueden ser:

- Observar (“sensar”): Se trata de determinar, mediante técnicas de procesado de señal, si la señal primaria está presente en la banda del espectro observada. Entre las técnicas más comunes se encuentran el detector de energía, el detector cicloestacionario y el detector de filtro adaptado.
- Utilizar bases de datos: El terminal cognitivo tiene permiso para acceder a una base de datos, mantenida por las autoridades regulatorias, y obtener información sobre el uso del espectro. La base de datos, además de proporcionar información sobre los huecos espectrales en un determinado punto o zona geográfica puede incluir información de localización de los transmisores primarios y una estimación del rango de interferencia ocasionada por los usuarios secundarios. Cuando un usuario secundario (SU) necesita transmitir, consulta la base de datos, selecciona una banda de frecuencia disponible y se le reserva para su uso. Cuando un usuario primario o secundario finaliza las transmisiones, la banda asociada es liberada y queda disponible para otros usuarios. Los usuarios primarios pueden iniciar las transmisiones en una frecuencia reservada por un SU en cualquier momento y, por tanto, los usuarios secundarios tienen que comprobar la base de datos periódicamente para evitar interferencias al sistema primario.
- Usar señales de balizamiento (*beacon signals*): Consiste en la difusión periódica de mensajes para proporcionar, en tiempo real, información sobre los sistemas primarios ubicados en la zona geográfica en donde la señal “*beacon*” es distribuida y así autorizar o denegar el acceso al espectro a los usuarios

secundarios. Aunque el uso de las “*beacon signals*” elimina la necesidad de detección del espectro por parte de los usuarios secundarios, las prestaciones de este método se degradan significativamente cuando los mensajes no pueden alcanzar a los usuarios secundarios debido al comportamiento del canal inalámbrico, como por ejemplo por la aparición de desvanecimientos.

En la tabla 2.1.4.1-1 se comparan las características de estas tres técnicas. Como se aprecia, a nivel de complejidad y coste de infraestructura, la implementación de una técnica del sensado espectral por parte del terminal cognitivo sería la mejor opción, ya que no resultaría muy costoso. En cambio, optar por utilizar una base de datos, precisaría del desarrollo de protocolos y requeriría de un mantenimiento. Mientras que utilizar balizas requeriría la implementación de transmisores que emitieran esas señales y las hicieran llegar a los usuarios secundarios. Por otra parte, a nivel de complejidad y coste del terminal cognitivo ocurre lo contrario. La opción más compleja la proporcionaría la técnica de sensado espectral ya que sería necesario integrar un conjunto de sensores dentro del terminal para que este pudiera “sensar”, mientras que, por ejemplo, utilizando una base de datos, el terminal únicamente necesitaría un enlace con dicha base. Finalmente, otro aspecto interesante a destacar, esta vez respecto a la compatibilidad, es la dificultad que tendrían las balizas, que deberían prever qué radiocanales estarían utilizando los usuarios secundarios para acceder a su información.

Características	Sensado Espectral	Base de Datos	Balizas
Complejidad de infraestructura y coste	Bajo	Alto	Alto/Medio
Complejidad del terminal y coste	Alto	Medio	Bajo
Compatibilidad sistemas previos	Alto	Medio/Bajo	Bajo
Fiabilidad	Bajo/Medio	Alto	Alto/Medio
Dinámica espectral	Alto	Bajo	Alto
Necesidad de un sistema/proveedor externo	No	Sí	Sí
Necesidad de banda adicional	No	No	Sí/No
Aspectos específicos	Consumo de tiempo y energía	Sistema de Posición	Solución Estandarizada

Tabla 2.1.4.1-1 Comparación de las técnicas del *Spectrum Awareness*. Extraída de [16].

2.1.4.2 *Spectrum Selection*

La Selección del Espectro (o *Spectrum Selection*) es la función responsable de la selección de las partes del espectro más apropiadas para su uso por parte de los usuarios cognitivos (secundarios).

La decisión de utilizar una banda de frecuencia determinada se toma en función de las características estimadas del canal, tales como su capacidad, ancho de banda, nivel de interferencia, retardo y patrones de uso del canal por parte de los usuarios primarios, etc.

La selección del espectro puede realizarse a nivel de banda o de radiocanal y de forma centralizada o distribuida. En los mecanismos centralizados de decisión del espectro debe intercambiarse una gran cantidad de información entre una unidad central de control y los usuarios de la red, produciendo sobrecargas en el tráfico de señalización. Por tanto, la asignación de los canales a todos los usuarios secundarios se propone como un problema de optimización. En el caso de la asignación distribuida del espectro, los canales son seleccionados por los usuarios secundarios de forma individual y dinámica, por lo cual debe existir algún tipo de mecanismo de coordinación para evitar colisiones entre ellos.

La Selección del Espectro comprende dos procesos:

- Análisis del Espectro (*Spectrum Analysis*): En este proceso se identifican las características de los agujeros o huecos espectrales mediante determinadas métricas.
- Decisión sobre el uso del Espectro (*Spectrum Decisión*): En este procedimiento se toma la decisión sobre qué huecos espectrales deben utilizar los usuarios cognitivos. Esta decisión no sólo se realiza a partir de las características (métricas) de los huecos espectrales sino también tomando en consideración otros requerimientos (por ejemplo, requisitos de QoS o capacidad estimada del canal) y/o políticas (coste económico del acceso a la banda, prohibición de uso bandas destinadas a usos militares de los terminales cognitivos, etc.).

2.1.4.2.1 *Spectrum Analysis*

El objetivo del Análisis Espectral (o *Spectrum Analysis*) es identificar las características de los huecos espectrales con el objeto de proporcionar “métricas” que permitan su selección en el procedimiento de decisión (*Spectrum Decision*).

Básicamente pueden identificarse dos tipos de métricas: métricas asociadas a parámetros de RF y métricas asociadas a la actividad de los primarios.

- Métricas asociadas a parámetros de RF:
 - Ancho de Banda disponible
 - Nivel de Interferencias en la banda
 - Límites de Emisión
 - Frecuencia
- Métricas asociadas a la actividad de los primarios:
 - Ciclo de Trabajo (*Duty Cycle*)
 - Tasa de aparición de los periodos de ocupación/vacío
 - Duración media de los periodos de canal ocupado/canal libre

2.1.4.2.2 *Spectrum Decision*

El objetivo de la Decisión Espectral (o *Spectrum Decision*) es seleccionar la mejor banda, de entre todas las seleccionadas o disponibles, con objeto de garantizar la calidad de servicio a los usuarios secundarios a medio y largo plazo.

Los procedimientos de decisión espectral deben asignar las bandas diferenciando entre dos situaciones distintas:

1. Un nuevo SU quiere unirse a la red. En consecuencia hay asignarle espectro para poder transmitir.
2. Un PU utiliza el canal o banda asignados. En este caso múltiples usuarios secundarios deben abandonar dicho canal y hay que reubicarlos en nuevos radiocanales.

En la figura 2.1.4.2.2-1 se muestra un marco de referencia para los mecanismos de decisión espectral:

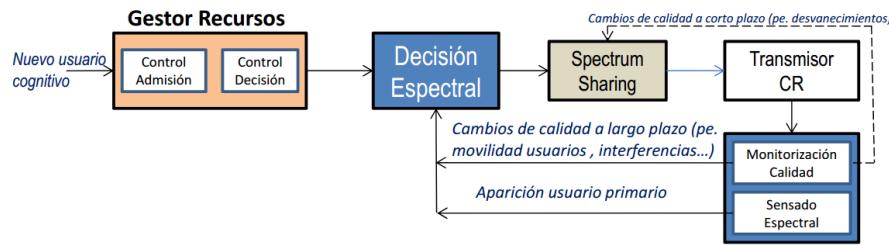


Figura 2.1.4.2.2-1 Ejemplo de mecanismo de decisión espectral. Extraída de [16].

Es importante observar que el responsable de abordar y resolver una situación en la que la calidad de canal se degrada y múltiples usuarios secundarios deben ser reubicados en mejores radiocanales es el mecanismo de *Spectrum Sharing*.

2.1.4.3 *Spectrum Sharing*

La Compartición del Espectro (o *Spectrum Sharing*) es la función destinada a proporcionar un acceso equitativo tanto a nivel de redes como de usuarios secundarios que coexistan en una misma región geográfica, mediante la coordinación del acceso a los huecos espectrales disponibles.

Así pues, cuando múltiples usuarios secundarios comparten una misma banda de frecuencia, el acceso al espectro debe coordinarse entre los SUs de modo que se minimicen las colisiones y la interferencia mutua. Convencionalmente, siempre ha sido la capa MAC la responsable de la coordinación del acceso al espectro. Por tanto, el *Spectrum Sharing* se enfrenta a problemas similares:

- Permitir a múltiples usuarios cognitivos acceder simultáneamente al espectro radioeléctrico.
- Evitar colisiones entre usuarios mediante un acceso coordinado.

Por el contrario, las diferencias principales con respecto a la capa MAC son:

- Se debe gestionar un amplio rango de bandas/canales libres disponibles en el espectro.
- Mantener y garantizar la calidad de servicio a aquellos usuarios con licencia (PU).

El trabajo abordado por los mecanismos de la Compartición del Espectro (*Spectrum Sharing*) tiene como objetivo enfrentarse a estos desafíos. Según [17], la Compartición del Espectro se puede clasificar tomando en consideración cuatro aspectos: la infraestructura, el algoritmo de compartición espectral, la técnica de acceso al espectro y el alcance.

La primera clasificación está basada en la infraestructura, la cual puede ser centralizada o distribuida:

- Centralizada: Los procedimientos de asignación y acceso al espectro son controlados por una entidad central, denominada *Spectrum Server* o *Spectrum Broker*. Cada usuario secundario de la red cognitiva envía sus mediciones sobre el uso del espectro a este nodo central, de manera que le permite construir un mapa de asignación de espectro. Los criterios utilizados por el *Spectrum Server* para realizar la asignación óptima de los agujeros espectrales a los usuarios secundarios pueden ser, por ejemplo: *Maximum Sum Rate Scheduling*, *Max-Min Scheduling*, *Proportional Fair Scheduling*, etc.

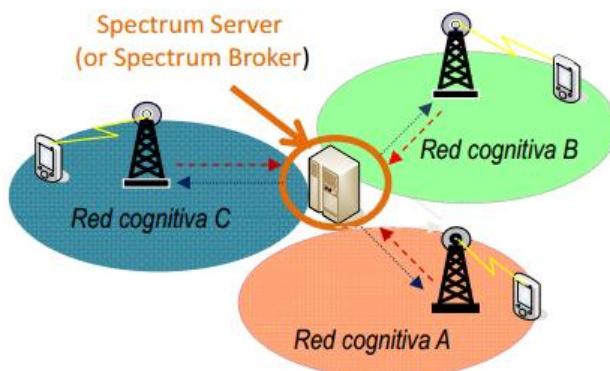


Figura 2.1.4.3-1 Infraestructura Centralizada. Extraída de [16].

- Distribuida: Se basa en las estructuras Ad hoc. Por lo cual, cada usuario es responsable de la asignación de espectro, y su acceso se decide mediante políticas locales. Es decir, los usuarios cognitivos han intercambiar información con otros usuarios vecinos para gestionar el acceso al espectro.

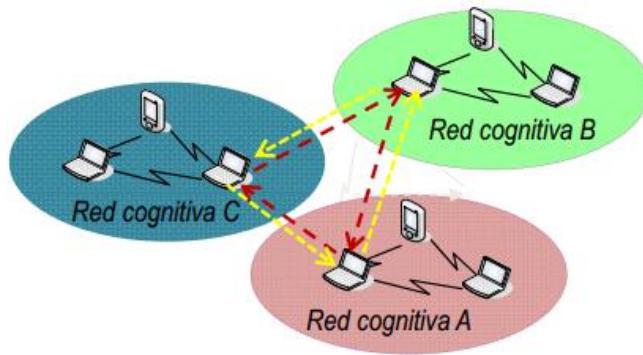


Figura 2.1.4.3- 2 Infraestructura Distribuida. Extraída de [16].

La segunda clasificación se basa en el comportamiento que tienen los usuarios a la hora asignar y acceder a los recursos. En este caso, encontramos algoritmos cooperativos y no cooperativos:

- Cooperativos (o colaborativos): Son algoritmos que toman en consideración como la asignación de recursos a un determinado usuario cognitivo afecta al resto. Una técnica común en este tipo de algoritmos es compartir las medidas de interferencia de un usuario cognitivo con el resto de usuarios a nivel local.

Este esquema no es recomendable para redes con limitaciones de energía o ancho de banda, debido a que la cooperación entre usuarios aumenta el número de mensajes de control. Tampoco resultan convenientes para usuarios que no deseen revelar, por razones de privacidad, cómo hacen uso del espectro o busquen evitar ataques por interferencia provocada (*jamming*).

- No cooperativos (o no colaborativos): Son algoritmos que no consideran la interferencia de los demás usuarios cognitivos. Por ello, ya no se requiere de un frecuente intercambio de mensajes entre SUs tal como se producía, por ejemplo, en la solución cooperativa. En contrapartida la eficiencia en la utilización del espectro, en general, puede verse reducida.

Así pues, en este caso, cada usuario cognitivo es responsable de la asignación de espectro. Su objetivo es seleccionar un canal que le permita

maximizar la velocidad de transmisión, sin tomar en consideración a los otros usuarios, mientras que el acceso es basado en políticas locales.

La tercera clasificación para la Compartición del Espectro en una red cognitiva se basa en la técnica de acceso. En este caso se distinguen Compartición del Espectro *Underlay* y Compartición del Espectro *Overlay* (véase el punto 2.1.4 *Acceso Dinámico al Espectro* donde se explican estas técnicas).

Por último, las técnicas de Compartición del Espectro se centran generalmente en dos tipos de soluciones: la compartición del espectro dentro de una red cognitiva (compartición del espectro *Intra-Network*) y la compartición entre múltiples redes cognitivas coexistentes (compartición del espectro *Inter-Network*):

- *Intra-Network*: En este caso se comparte el espectro entre usuarios de la misma red cognitiva. En consecuencia, los algoritmos se centran en distribuir los huecos espectrales entre los usuarios de la red. Se debe coordinar el acceso entre múltiples usuarios cognitivos con el fin de prevenir colisiones. También se debe asegurar que los usuarios cognitivos acceden al espectro sin causar interferencia a los usuarios primarios.
- *Inter-Network*: En este caso múltiples sistemas cognitivos se despliegan en la misma área y utilizan las mismas bandas del espectro. Por ello se necesita un mecanismo de coordinación entre redes.

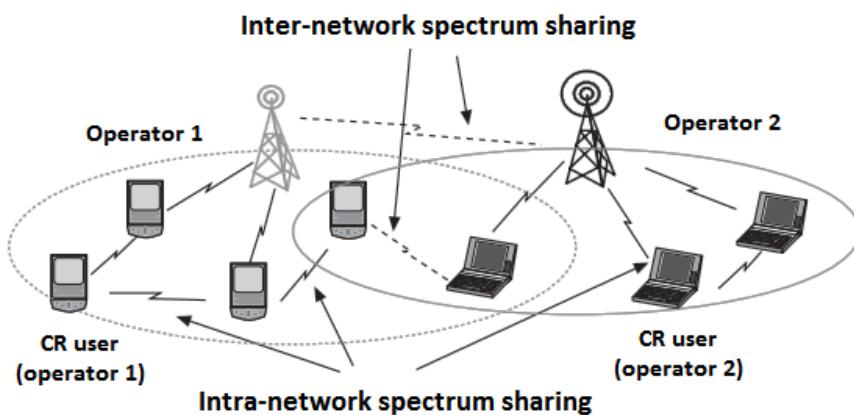


Figura 2.1.4.3-3 Compartición del espectro *Inter-Network* e *Intra-Network* en una Red Cognitiva. Extraída de [17].

2.1.4.4 *Spectrum Mobility*

La Movilidad Espectral (o *Spectrum Mobility*) es la función responsable de evitar interferir a los usuarios con licencia (PU).

En un entorno de radiocomunicaciones de acceso dinámico al espectro, la disponibilidad y la calidad de los canales inalámbricos pueden cambiar frecuentemente a lo largo del tiempo. Cuando en un sistema radio cognitivo se detecta la presencia de un PU en un radiocanal en uso, dicho radiocanal debe ser liberado inmediatamente por parte de los usuarios secundarios para permitir su uso por parte de los usuarios con licencia. En esta situación, los usuarios secundarios desocupan el radiocanal y se desplazan a un hueco espectral alternativo.

El procedimiento que permite la transición de un radiocanal a otro, con una degradación mínima en las prestaciones del sistema, se conoce como *Spectrum Handover* o *Spectrum Handoff*. El retardo que se produce durante el *Spectrum Handover* tiene un impacto negativo en las prestaciones del sistema percibidas por el usuario, y es un elemento a tener en cuenta en el diseño de protocolos de comunicación para la radio cognitiva. Otro factor importante es el retardo que se produce entre el momento que se detecta la presencia de las transmisiones primarias en una banda de frecuencia y el momento en que se desocupa dicha banda, ya que las transmisiones de los usuarios secundarios durante este período pueden provocar interferencia perjudicial a los usuarios primarios.

Según [18], los mecanismos de Movilidad Espectral pueden clasificarse en reactivos o proactivos:

- **Reactivo:** El usuario cognitivo espera a la aparición del usuario primario o al empeoramiento del comportamiento del canal para tomar decisión de cambiar de frecuencia (canal) y continuar la comunicación. En este caso se debe monitorizar de forma regular el canal y hay probabilidad de interferencia con el usuario primario.
- **Proactivo:** El usuario cognitivo utiliza el conocimiento generado a partir de las observaciones del canal realizadas en el pasado (modelado estadístico del comportamientos de los usuarios primarios) para predecir la disponibilidad espectral futura; en particular cual es la probabilidad de que el canal

continúe vacío en la siguiente observación. El objetivo de este tipo de algoritmo es minimizar las interferencias con los usuarios primarios.

Otra clasificación posible para los algoritmos de *Spectrum Handover* puede ser diferenciarlos entre centralizados o distribuidos:

- Centralizado: Se informa a la estación base o al punto de acceso cuando se detecta la presencia del usuario primario.
- Distribuido: Se informa al resto de los nodos (usuarios cognitivos) cuando se detecta la presencia del usuario primario.

2.1.5 Estandarización

Las principales instituciones implicadas en los procesos de estandarización de los sistemas de Radio Cognitiva son la Unión Internacional de Telecomunicaciones (ITU), el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), la Asociación Europea para la Estandarización de los Sistemas de la Información y las Comunicaciones (ECMA) y el Instituto Europeo de Normas de Telecomunicaciones (ETSI).

Por parte del IEEE, el Comité de Coordinación de Estándares (SCC) 41 lleva a cabo las actividades de estandarización sobre las Redes de Acceso Dinámico al Espectro, mientras que el Comité de Estándares 802 lleva a cabo las actividades de estandarización sobre redes LAN/MAN (véase tabla 2.1.5-1 y tabla 2.1.5-2).

En el estándar IEEE 802.22 se especifican las interfaces inalámbricas y las capas física y MAC (control de acceso al medio) de redes inalámbricas punto a multipunto de área regional (WRAN), compuestas por estaciones base y dispositivos móviles/fijos operando en la banda espectral que va desde los 54 MHz hasta los 862 MHz. Como es sabido, dicho rango espectral es usado por canales de televisión, telefonía móvil, redes de emergencias, emisoras de radio, etc. Para operar en los *TV White Spaces* (véase el punto 2.2) de las bandas asignadas a los sistemas de televisión, se definen los siguientes requisitos: una probabilidad de detección mayor del 90%, una probabilidad de falsa alarma menor del 10% y que los sistemas secundarios tengan una capacidad de detectar señales de -116dBm y -94 dBm para los sistemas de TV digitales y analógicos, respectivamente.

Estándar	Propósito
P1900.1	Terminologías y conceptos relacionados con los sistemas de radio de próxima generación y redes de acceso dinámico al espectro.
P1900.2	Guía técnica para el análisis de la interferencia y coexistencia entre sistemas de radio operando en las mismas bandas de frecuencias.
P1900.3	Ánalisis y pruebas técnicas para la evaluación de los sistemas de radio con capacidades de acceso dinámico al espectro.
P1900.4	Arquitectura para la optimización del uso de los recursos de radio en redes inalámbricas heterogéneas.
P1900.4.1	Interfaces y protocolos para la optimización del uso de los recursos de radio en redes inalámbricas heterogéneas.
P1900.4a	Arquitecturas e interfaces de redes de accesos dinámico al espectro.
P1900.5	Requerimientos de los lenguajes de especificación de regulaciones y arquitectura para el acceso dinámico al espectro.
P1900.6	Interfaces y estructuras de datos para el intercambio de información de detección del espectro.

Tabla 2.1.5-1 Estandarización de la Radio Cognitiva (IEEE SCC 41). Extraída de [10].

Estándar	Propósito
P802.22	Políticas y procedimientos para la operación de las WRANs en la bandas de TV.
P802.11y	Mecanismos de coexistencia en las bandas de 3650 a 3700 MHz. Estos mecanismos incluyen la especificación de nuevas regulaciones, detección de otros transmisores, TPC y DFS.
P802.11af	Es una modificación de las especificaciones de las capas física y MAC del estándar 802.11, con el objetivo de cumplir con los requerimientos legales para el acceso y coexistencia en las bandas de TV.
P802.22.1	Métodos de protección de los dispositivos de baja potencia en las bandas de TV.
P802.19.1	Mecanismos de coexistencia en las bandas de TV.

Tabla 2.1.5-2 Estandarización de la Radio Cognitiva (IEEE 802). Extraída de [10].

Además, éste estándar ha adoptado muchas de las características del estándar 802.16 en las especificaciones de las capas física, control de acceso al medio y convergencia. En la figura 2.1.5-1 se muestra la arquitectura de protocolos propuesta en el estándar 802.22, donde se separa el sistema en: Plano Cognitivo, Plano de Datos/Control y Plano de Administración. Las funciones de geolocalización (GL) y detección del espectro (SSF), que interactúan con la etapa de RF del dispositivo, proporcionan, respectivamente, información de localización e información sobre la presencia de señales primarias al módulo de Administración

del Espectro (SM). Por tanto, es en éste módulo donde reside la mayor parte de la inteligencia y la capacidad de tomar decisiones. Las funciones de seguridad para los planos Cognitivo, de Información/Control y Administración incluyen la integridad de los datos, identificación, autenticación, autorización y confidencialidad.

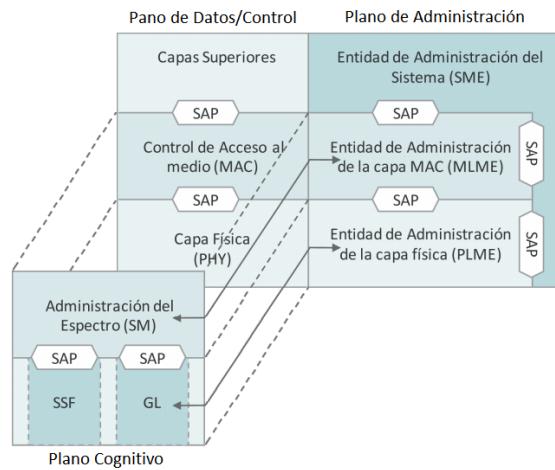


Figura 2.1.5-1 Arquitectura del protocolo IEEE 802.22. Extraída de [10].

2.2 TV White Spaces (TVWS)

2.2.1 Contextualización

El *Apagón Analógico*, nombre con el que se conoce al cese de las emisiones analógicas de los operadores de televisión alrededor del mundo, corroboró que, en las últimas décadas, se ha tomado un camino vertiginoso en la transición de la tecnología analógica a la tecnología digital.

En España, la migración de la televisión analógica a la digital llegó el 3 de abril del 2010, cuando todas las emisiones de TV terrestres comenzaron a realizarse mediante técnicas digitales y quedó definitivamente implantada la denominada TV Digital Terrestre (TDT).

Respecto a la emisión analógica, la TDT ofrece una transmisión más eficiente, utilizando menos espectro para ofrecer más canales de televisión. En particular, con el ancho de banda utilizado, 8 MHz, para la difusión de un único

programa de tecnología analógica en un canal UHF, la TDT permite la transmisión de hasta 6 programas².

Gracias a la implementación de la TDT, se logró el denominado Dividendo Digital, que es el nombre otorgado a la banda de frecuencias del espectro asignado a la televisión analógica que quedó disponible para su uso como banda de comunicaciones móviles. En España, la banda de frecuencias de 800 MHz estaba ocupada, en parte, por algunos canales de la TDT (canales del 61 al 69 de UHF, véase figura 2.2.1-1). La liberación del Dividendo Digital fue el proceso de reordenación de frecuencias necesario para que la banda 800 MHz quedara disponible en toda Europa. De esta manera podía utilizarse dicho Dividendo Digital para uso de otros servicios (DVB-H, telefonía móvil, etc.).

Tras la liberación, el 31 de marzo de 2015, la banda de 800 MHz dejó de utilizarse para la transmisión de la TDT y se asignó a los operadores de telefonía móvil para prestar nuevos servicios de banda ancha de cuarta generación (4G), tal y como ya anunció el Gobierno en el plan Marco del Dividendo Digital:

“El Dividendo Digital brinda la oportunidad de introducir la 4ª Generación de telefonía móvil (LTE), así como asegurar la cobertura de la banda ancha móvil ultrarrápida al 98% de la población, facilitando el cumplimiento de los objetivos de la Agenda Digital para Europa y contribuyendo decididamente a reducir la brecha digital.” [19]

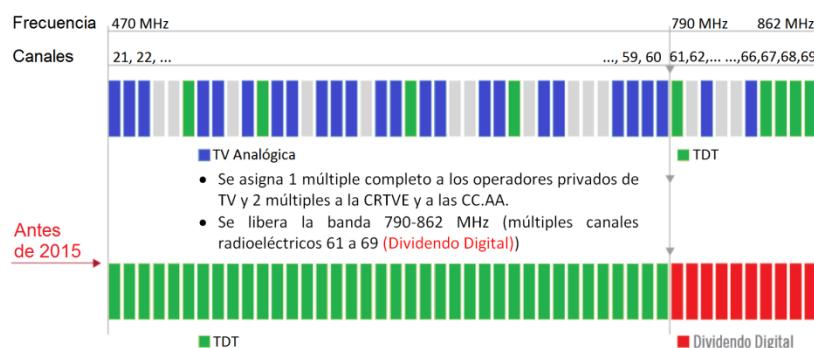


Figura 2.2.1-1 Banda espectral asignada a la televisión y Dividendo Digital. Extraída de [16].

² Nota: A cada canal de la TDT se le llama múltiple (cada múltiple transmite diferentes programas de televisión a la vez). En el estado español se hace una distinción entre tres tipos de múltiples: estatales, autonómicos y locales.

2.2.2 Concepto de *TV White Space*

Una vez completada la reordenación del espectro en la banda de TV (470 MHz – 782 MHz; canales del 21 al 60 con ancho de banda de canalización de 8 MHz c/u) y finalizada la transición a la TDT, en una zona geográfica dada existe un cierto número de canales de TV que no están siendo utilizados por estaciones de TDT.

Sin embargo, un transceptor, operando en uno de dichos canales de televisión localmente vacantes y con un nivel de potencia mucho más bajo, no necesitaría una gran separación física respecto a la ubicación de las estaciones de TV para evitar causar interferencias tanto cocanal, como por canal adyacente sobre la señal de TV. A estos canales de televisión vacantes, en los que pueden operar dispositivos de baja potencia, se conocen como *TV White Spaces* (TVWS).

Análogamente, los *TV White Spaces* han sido definidos por el comité técnico RRS³ del ETSI como: *“La parte del espectro de la banda asignada a TV, que está disponible para una aplicación de radiocomunicaciones (servicio, sistema) en un momento dado en una zona geográfica determinada de forma que no interfiera ni requieran protección los servicios primarios y otros servicios con una mayor prioridad a nivel nacional.”* [20]

Obsérvese que la banda de TV es una banda muy estable, debido a la difusión de los programas de forma constante, es decir, que nunca cesa. Consecuentemente, los TVWS se caracterizan por ser muy estáticos, permitiendo así conocer con mayor facilidad y fiabilidad su ubicación. En la siguiente figura se muestra la ocupación que tienen los canales de TV desde la azotea del edificio D4 del Campus Nord (UPC) [21]:

³ RRS (Reconfigurable Radio Systems).

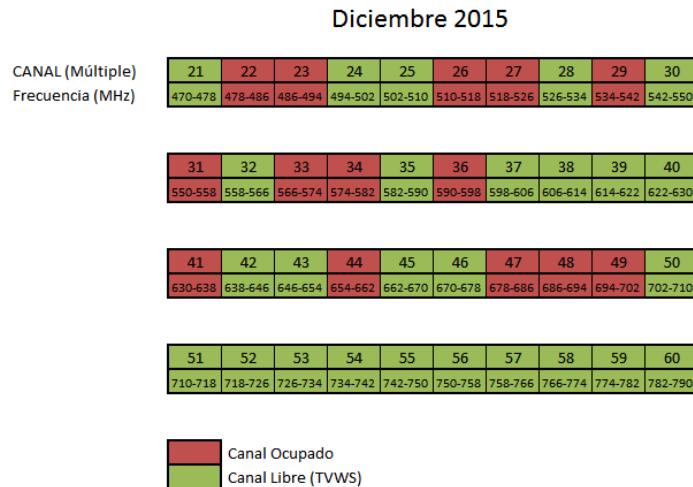


Figura 2.2.2-1 TVWS en edificio D4 del Campus Nord (UPC).

2.3 Software Defined Radio (SDR)

Como se ha explicado en los anteriores apartados de este capítulo, la meta de la Radio Cognitiva es conseguir un dispositivo inalámbrico que adquiera la inteligencia suficiente como para permitirle analizar el entorno radio y tomar decisiones autónomamente, en lo que hace referencia a la elección de la mejor banda espectral, la utilización de un protocolo óptimo de comunicación, la reconfiguración de los parámetros de transmisión, etc., y siempre intentando que los niveles de energía consumidos para estas acciones sean lo más bajos posibles. La tecnología que soporta dicho paradigma se la denomina *Software Defined Radio* (SDR).

2.3.1 Concepto

El término *Software Defined Radio* (SDR) fue introducido por Joseph Mitola III en 1991 para referirse a un tipo de radios reprogramables donde un mismo componente *hardware* podía utilizarse para realizar diferentes funciones, en distintos instantes de tiempo, cuando se aplicaban cambios en su configuración a través de *software*. Mitola lo definió de la siguiente manera:

“El software radio es una radio donde las señales portadoras que se van a modular y transmitir por un canal son definidas mediante software. Es decir, las formas de onda son generadas como señales digitales muestreadas, convertidas de digital a analógico a través de un Conversor Digital Analógico (DAC) de banda ancha y con

la posibilidad de subir la señal de Frecuencia Intermedia (IF) a Radiofrecuencia (RF). De la misma manera, el receptor emplea un Conversor Analógico Digital (ADC) de banda ancha que captura todos los canales del nodo del software radio. Entonces, el receptor extrae, baja y demodula la señal utilizando software en un ordenador de propósito general.” [22]

Por lo tanto, se puede ver el concepto de *Software Defined Radio* como una tecnología cuyo principal objetivo es que el procesado de la señal se realice a través de *software* y que, además, dicho procesado se acerque lo máximo posible a la antena, convirtiendo los problemas que hasta ahora habían recaído en el *hardware* (mezcladores, filtros, amplificadores, moduladores, demoduladores, detectores, etc.) en problemas de tipo *software*.

2.3.2 Sistemas *Software Defined Radio*

En un sistema SDR ideal la digitalización de la señal se realizaría a pie de antena. Es decir, tanto en el transmisor como en el receptor, todo sería configurado mediante *software* a excepción de los conversores y de la antena, tal y como se muestra en la siguiente figura:

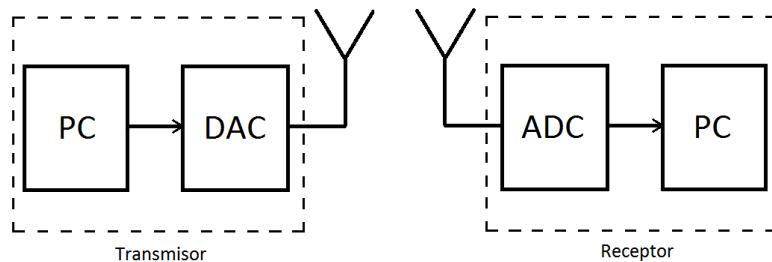


Figura 2.3.2-1 Sistema SDR ideal

Por consiguiente, los problemas actuales surgidos del comportamiento no ideal de mezcladores, filtros, amplificadores, etc., se solucionarían y la distorsión introducida en la señal transmitida se vería minimizada en recepción. Sin embargo, el estado actual de la tecnología hace que un sistema SDR ideal no pueda implementarse dado que hay una serie de limitaciones:

- Limitación en el ADC: Como sabemos, el teorema de Nyquist establece que, para evitar el *aliasing* (*solapamiento en el espectro*) de una señal muestreada, la frecuencia mínima de muestreo (denominada también

frecuencia de Nyquist) ha de ser superior al doble de la frecuencia máxima de la señal a muestrear.

$$f_{Nyquist} > 2f_{máxima}$$

Esto implica que, sin filtrado previo, si trasmitiéramos directamente una señal de RF en la banda de 2.5 GHz al conversor analógico digital, este debería soportar un ancho de banda de 5 GHz. Y, a pesar de que los conversores actuales soportan tasas de muestreo altas, de momento los anchos de banda máximo soportados son de cientos de MHz.

Por otra parte, en la práctica, hay que eliminar el ruido y todas aquellas señales indeseadas que se encuentren fuera del espectro de Nyquist, ya que si no se suprimen provocan *aliasing* al realizar el muestreo. Por tanto, es necesario utilizar un filtro paso-bajo analógico de frecuencia de corte $f_{Nyquist}/2$ (filtro anti-aliasing) antes de muestrear.

Todo esto es en relación a la etapa de muestreo de un conversor analógico digital. En cuanto a la etapa de cuantificación, también tenemos limitaciones. Si la señal recibida sufre una gran atenuación y los niveles de potencia son muy bajos en comparación al rango dinámico del ADC (bit de resolución), diferentes muestras que correspondían a diferentes niveles podrían ser cuantificados en un mismo estado y, por tanto, la señal no sería cuantificada correctamente.

- Limitación en los buses: La velocidad de transferencia máxima de los buses oscila normalmente entre varios cientos de Mbps hasta algún Gbps. Este hecho, junto al hecho de que no todos bits transmitidos son de datos, crean una restricción en la tasa binaria máxima del sistema.
- Limitación en el PC: El procesamiento en tiempo real de todos los datos que el sistema genera requiere de ordenadores potentes que sean capaces de realizar cálculos matemáticos avanzados con velocidades elevadas y de forma concurrente.

Debido a estas limitaciones y con el objetivo de mejorarlas, a la arquitectura del sistema que se muestra en la figura 2.3.2-1 se le introduce unos cuantos bloques de *hardware* adicional que permiten, entre otras cosas, filtrar en RF, amplificar,

bajar la señal a IF y procesarla en banda base. En la siguiente figura se muestra la estructura de un posible SDR⁴:

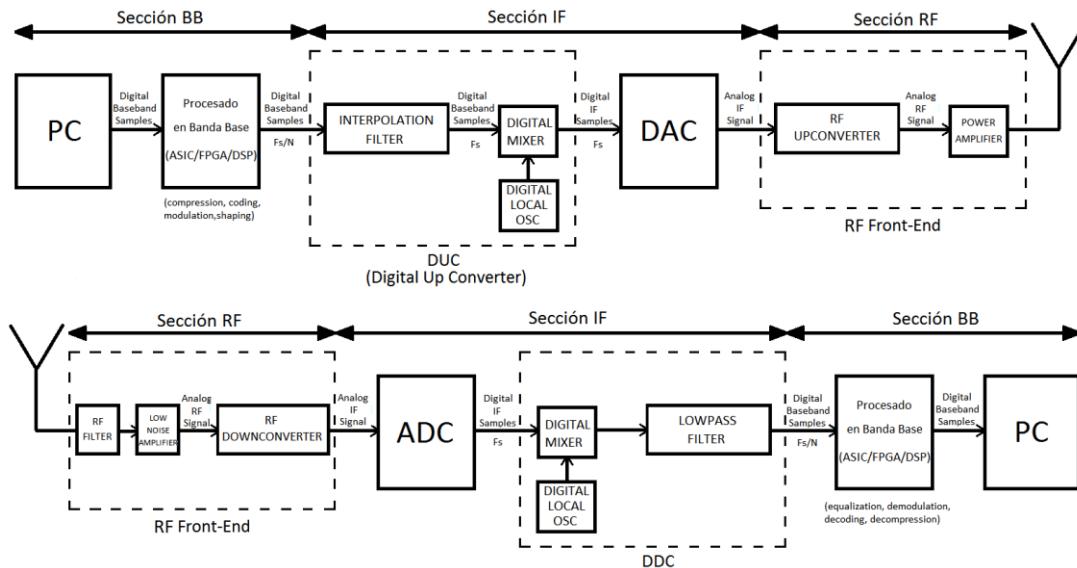


Figura 2.3.2-2 Arriba, estructura SDR de un Transmisor. Abajo, estructura SDR de un Receptor.

La estructura del sistema SDR de la figura 2.3.2-2 se compone de tres etapas:

- La etapa, o sección, de Radiofrecuencia (RF) tiene como función:
 - En transmisión, modular y amplificar las señales de IF ajustándolas de manera que, posteriormente, puedan ser transmitidas vía radio.
 - En recepción, adecuar las señales de RF mediante filtrado y convertirlas a frecuencia intermedia.
- La etapa, o sección, de Frecuencia Intermedia (IF) se ocupa de:
 - En transmisión, convertir las señales de banda base a IF (DUC, Digital Up Conversion) y, posteriormente, convertir dichas señales digitales en señales analógicas (DAC).

⁴ Nota: aunque en la actualidad se utiliza indistintamente los términos SR y SDR, se considera que un *Software Defined Radio* (SDR) es una versión de un *Software Radio* (SR) implementable con la tecnología disponible actualmente.

- En recepción, digitalizar las señales de IF (ADC) y, después, convertirlas a banda base (DDC, Digital Down Conversion).
- La etapa, o sección, de Banda Base (BB) es la encargada, tanto en transmisión como en recepción, de todo el procesado digital en banda base de las señales.

Cabe especificar que, mientras que la etapa de RF, generalmente, se constituye toda ella con *hardware* analógico, las etapas de IF y Banda Base se implementan con módulos *hardware* digitales.

Por otra parte, para minimizar el potencial problema que conlleva las operaciones que tiene que soportar la unidad de procesado se utiliza un *hardware* que normalmente está constituido por una combinación de DSP (*Digital Signal Processor*) con FPGA (*Field Programmable Gate Array*) y ASIC (*Application Specific Integrated Circuit*). Las operaciones de procesado que conlleven un alto coste computacional serán realizadas por dispositivos ASIC o bien FPGA, liberando así de carga computacional al ordenador.

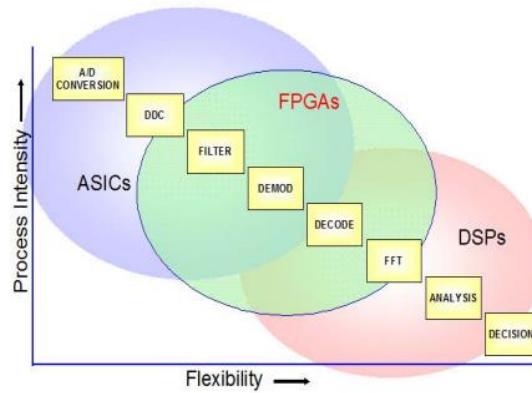


Figura 2.3.2-3 Gráfico orientativo para la elección de ASICs, FPGAs o DSPs. Extraído de [23].

La configuración o reconfiguración del *hardware*, es decir, lo que realmente proporciona flexibilidad al sistema y permite acercarnos al concepto de Radio Cognitiva, se realiza mediante *software*. Actualmente, existen muchas herramientas que dan soporte al desarrollo de dicho *software* y son muchos los lenguajes de programación que se pueden emplear para ello como, por ejemplo, C/C++, VHDL, etc.

3. Equipo físico utilizado (*hardware*)

En este capítulo se explican los componentes *hardware* utilizados para construir el sistema SDR y llevar a cabo el demostrador de Radio Cognitiva. En esencia, el equipo *hardware* está formado por:

- 2 ordenadores de propósito general
- 2 USRPs
- 4 antenas (2 antenas para cada USRP, de las cuales una se utiliza para emitir y otra para recibir)

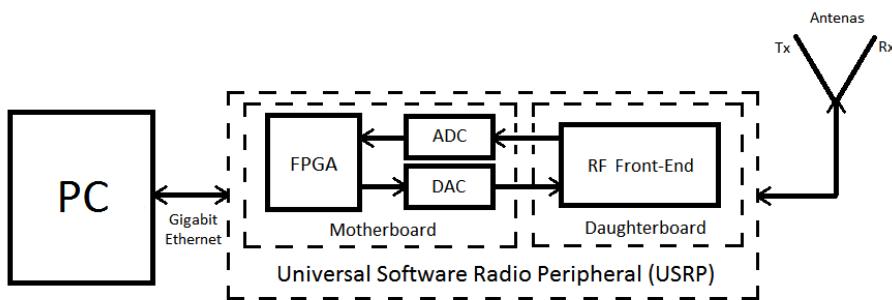


Figura 3-1 Arquitectura del sistema SDR implementado.

3.1 Ordenador de propósito general

La principal función del ordenador de propósito general, denominado comúnmente PC (*Personal Computer*), es procesar la señal en banda base y, por tanto, es en este PC donde se va a llevar a cabo las principales funciones implicadas en un sistema de comunicación digital (codificación, decodificación, modulación, demodulación, sincronismo, etc.), permitiendo que lo que antes se implementaba a través de *hardware*, ahora se realice mediante *software*.

En el laboratorio se van a utilizar 2 ordenadores; uno al que denominaremos A que desempeñará las funciones de Estación Base y de receptor y otro ordenador al que denominaremos B y que tendrá la función de usuario cognitivo (SU).

Las especificaciones del ordenador A son:

- Sistema Operativo: Ubuntu 14.04 LTS (64 bits)
- Procesador: Intel Core i5-3330 @ 3.00 GHz x4

- Memoria RAM: 8 GB

Las especificaciones del ordenador B son:

- Sistema Operativo: Ubuntu 14.04 LTS (64 bits)
- Procesador: Intel Core i3 CPU 540 @ 3.07 GHz x4
- Memoria RAM: 4 GB

Como se puede apreciar, son ordenadores de gama media, que podría tener cualquier persona en su domicilio. Queremos remarcar esto para expresar que no vamos a trabajar con supercomputadores y que, por tanto, la implementación del *software* en un sistema *Software Defined Radio* es algo realizable, incluso a nivel de aficionado, y no necesita de grandes infraestructuras para su desarrollo.

3.2 Universal Software Radio Peripheral (USRP)

El *Universal Software Radio Peripheral* (USRP) es básicamente un cabezal de radiofrecuencia, de configuración flexible, comercializado por *Ettus Research*, y que está diseñado para trabajar en conjunto con un procesador externo (PC, *Workstation*, etc.) a través de una FPGA, a la que se conecta mediante un bus *Ethernet*, y que permite la realización de sistemas *Software Defined Radio*.

Hay diferentes tipos de USRP con diferentes características y coste. En la tabla 3.2-1 se muestra una comparativa de las características más relevantes de los diferentes modelos. En nuestro caso, el Grupo de Investigación en Comunicaciones Móviles (GRCM) de la UPC dispone de 2 USRPs N200 y, por ello, son los que se han utilizado en este proyecto.

El USRP N200 está diseñado para aplicaciones que requieren un gran ancho de banda. La arquitectura del producto incluye una *Xilinx Spartan 3A-DSP 1800* FPGA, un conversor DAC de doble canal de 16 bits de resolución y 400 MS/s de tasa de muestreo, un conversor ADC de doble canal de 14 bits de resolución y una tasa de muestreo de 100 MS/s, un *Digital Down Converter* (DDC) y un *Digital Up Converter* (DUC), ambos de 25 MHz de resolución. Además, mediante el puerto expansión, es posible la sincronización de varios USRP N200 y, en consecuencia, la implementación de una configuración 2x2 MIMO (*Multiple Inputs Multiple*

Outputs). La comunicación entre el USRP y el PC se realiza a través del protocolo *Gigabit Ethernet*.

	USRP1	USR B100	USR N200	USR N210	USR E100	USR E110
Interfaz	USB 2.0	USB 2.0	Gigabit Eth	Gigabit Eth	Embebido	Embebido
Daughterboard Slots	2	1	1	1	1	1
MIMO	Sí	No	Sí	Sí	No	No
Resolución ADC (bits)	12	12	14	14	12	12
Tasa muestreo ADC (MS/s)	64	64	100	100	64	64
Resolución DAC (bits)	14	14	16	16	14	14
Tasa muestreo DAC (MS/s)	128	128	400	400	128	128

Tabla 3.2-1 Comparación entre modelos USRP de sus características más relevantes.

En el panel frontal de la interfaz del dispositivo (figura 3.2-1) se encuentran:

- 6 LEDS que informan sobre el estado del dispositivo (véase el Apéndice A).
- 4 Conectores SMA:
 - RF1 y RF2 son los conectores para los dipolos (Tx/Rx).
 - REF CLOCK sirve para utilizar una señal de reloj de referencia externa de 10 MHz.
 - PPS IN es utilizado para mejorar la sincronización de la señal.
- El puerto de expansión MIMO.
- El puerto *Gigabit Ethernet*.
- El conector de la fuente de alimentación.



Figura 3.2-1 Panel frontal del USRP N200. Extraída de [24].

Al USRP se le puede ver como un dispositivo formado, en su esencia, por dos placas: una placa madre (*motherboard*) y una placa hija (*daughterboard*). En la *motherboard* se encuentra, principalmente, la FPGA, los conversores ADC/DAC, la alimentación y la conexión vía *Ethernet*. La *daughterboard*, por su parte, es la encargada de trasladar de banda de base a RF, y viceversa, la señal de interés. A continuación se describe de forma más detallada las funciones que realizan estas placas dentro del USRP.

3.2.1 La placa madre (*motherboard*)

La placa madre, o *motherboard*, es el “cerebro” del USRP y ejerce de intermediario en el tratamiento de la señal entre el host (PC) y la placa hija. Sus funciones principales son: la conversión de la señal del dominio analógico al digital (cuando se recibe), la conversión de la señal del dominio digital al analógico (cuando se transmite) y el ajuste de la tasa binaria del sistema. En la figura 3.2.1-1 se muestra el diagrama de bloques de la placa madre del USRP N200:

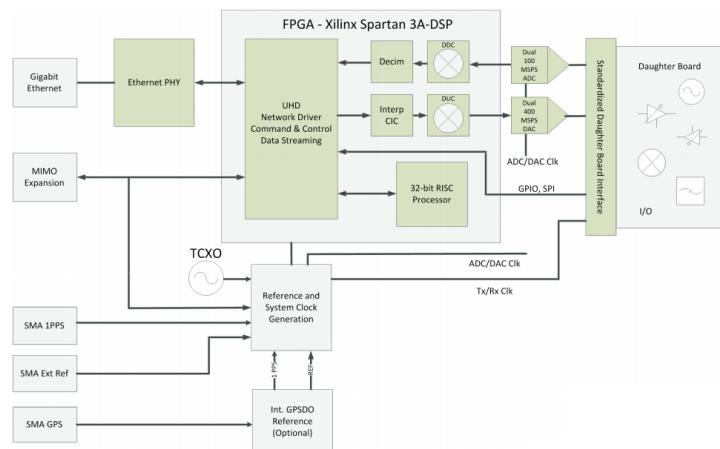


Figura 3.2.1-1 Arquitectura de la *motherboard* del USRP N200. Extraída de [24].

En el caso de que se desee transmitir una señal a través del canal radioeléctrico, lo primero que se hace es generar una señal en banda base mediante *software*. Posteriormente, las componentes en fase y cuadratura (I/Q) de la señal se envían desde el PC a la *motherboard* del USRP N200 entrelazadas a través de un cable que soporta el estándar *Gigabit Ethernet*. Una vez las componentes I/Q de la señal han llegado a la FPGA, se desentrelazan y son tratadas mediante sendos filtros (proceso de interpolación) para finalmente entregarse al DUC, quien las convertirá en una señal digitalizada a frecuencia intermedia⁵ (IF). Por último, desde la FPGA, la señal, todavía digitalizada, se envía al DAC, quien la transforma del dominio digital al dominio analógico para ser entregada a la placa hija.

En el caso de que se estuviera recibiendo, la señal a IF llega al ADC procedente de la placa hija. Por consiguiente, las muestras de la señal pasan del dominio analógico al dominio digital y son enviadas a la FPGA. Una vez en la FPGA, la señal es convertida a banda base por el DDC y tratada mediante filtros digitales (proceso de diezmado), que entrelazan las componentes en fase y cuadratura. Finalmente, las muestras digitales de la señal en banda base se envían a través del medio físico (interfaz *Gigabit Ethernet*) hacia el ordenador, donde serán procesadas por *software*.

Cabe hacer mención que la conversión de frecuencia que realizan el DUC y el DDC (de banda base a FI y de FI a banda de base, respectivamente) depende del modelo de placa hija que contenga el USRP. Es decir, hay modelos de placas hija que operan en modo *direct conversión* y, por tanto, son capaces de llevar a cabo la conversión de banda base a RF (o viceversa) directamente, sin utilizar el DDC o el DUC.

En cuanto a los conversores ADC y DAC, el conversor analógico digital (ADC) de doble canal (ADS62P44) tiene un ganancia programable de hasta 6.5 dB, mediante un ajuste grueso en pasos de 0.5 dB y un ajuste fino en pasos de 0.1 dB.

⁵ En algunas placas hijas, existe la posibilidad de trabajar en modo “*direct conversión*”, de modo que los conversores ADC y DAC son de doble canal (un canal para las componentes en fase y otro canal para las componentes en cuadratura) para facilitar este modo de operación. En este caso, lo que se entrega a la salida de los conversores son muestras procesadas de las señales.

Por el contrario, el conversor digital analógico (DAC) de doble canal (AD9777) no presenta ganancia alguna.

```

/
|   RX Codec: A
|   Name: ads62p44
|   Gain range digital: 0.0 to 6.0 step 0.5 dB
|   Gain range fine: 0.0 to 0.5 step 0.1 dB
/
|   TX Codec: A
|   Name: ad9777
|   Gain Elements: None

```

Figura 3.2.1-2 Ejecutando */uhd_usrp_probe* en un terminal de Ubuntu, se obtiene información sobre los componentes del USRP.

Por parte de los filtros diezmadores e interpoladores, cabe resaltar que su función es la de adecuar las tasas binarias del sistema para que puedan ser soportadas tanto por la interfaz *Ethernet* como por el ordenador.

Finalmente, sobre la señal de reloj mencionar que puede ser una señal de reloj externa (SMA 1PPS, SMA Ext Ref, SMA GPS) o, por el contrario, una interna (TCXO). Esta señal de reloj se utiliza en: los conversores ADC y DAC, la FPGA, y también en la placa hija para mejorar la sincronización del dispositivo.

3.2.2 La placa hija (*daughterboard*)

La placa hija, o *daughterboard*, realiza prácticamente las mismas funciones que un cabezal de radiofrecuencia (*RF Front-End*), por tanto, su cometido es adecuar las señales para que puedan ser recibidas o transmitidas. La placa madre y la placa hija se comunican mediante dos ranuras que hay en la placa madre. Una ranura para la tarjeta de transmisión, etiquetada como TX, y otra ranura para la tarjeta de recepción, etiquetada como RX. La ranura TX tiene acceso al DAC, mientras que la ranura RX tiene acceso al ADC.

Al igual que ocurría con las familias de USRP, hay distintos modelos de *daughterboards* para cubrir distintas necesidades del usuario (véase figura 3.2.2-1). En nuestro caso, los USRPs N200 disponibles en el laboratorio vienen cada uno con una placa WBX.

La placa WBX, utilizada en este proyecto, es un transceptor (transmisor y receptor) de banda ancha que proporciona una potencia típica de salida de 100 mW y una figura de ruido de 5 dB. Por defecto, opera en *direct conversión*, por lo que la conversión de la señal de RF a banda base (o viceversa) se hace sin pasar por frecuencia intermedia. El oscilador de la cadena de transmisión y el oscilador de la cadena de recepción son independientes, lo que permite operar en modo *full-duplex* (transmitir y recibir en ambas direcciones al mismo tiempo) con señales complejas. No obstante, ambos osciladores pueden ser sincronizados para soportar MIMO. El transceptor proporciona un ancho de banda de 40 MHz y un rango de operación de frecuencia que va de los 50 MHz a los 2.2 GHz. La ganancia que presenta en transmisión tiene como rango de 0 dB a 31 dB en pasos de 1 dB, mientras que la ganancia en recepción va de 0 dB a 31.5 dB en pasos de 0.5 dB.

En este punto cabe mencionar que, cuando se especifica una ganancia del USRP, esta es la suma de las contribuciones del PGA (*Programmable Gain Amplifier*) del conversor ADC más la del amplificador de potencia programable de la placa WBX. Por ejemplo, si se indica una ganancia de 25 dB, el PGA del conversor ADC se ajustará a 10 dB y la placa WBX a 15 dB.

Finalmente, indicar que la placa WBX tiene dos conectores SMA, uno de ellos se puede configurar bien como transmisor o bien como receptor (TX/RX), mientras que el otro sólo puede actuar como receptor (RX2).

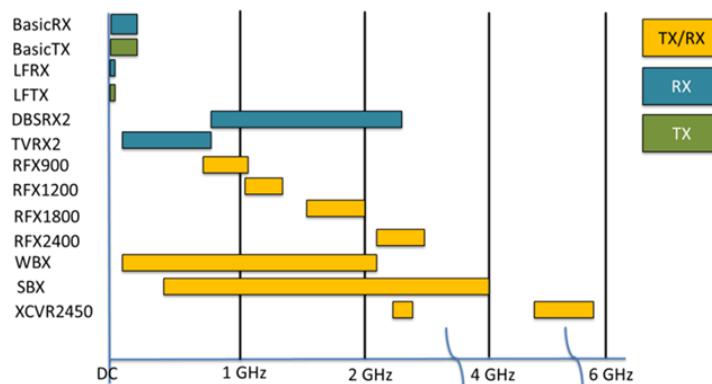


Figura 3.2.2-1 Rango de frecuencias de operación alcanzados por modelos de *daughterboards*. Extraída de [25].

3.3 Antenas

Las antenas serán las encargadas de transmitir o recibir las ondas de radiofrecuencia. En nuestro caso, los USRPs vienen con antenas VERT400.

Sus especificaciones son las siguientes:

- Dipolo omni-direccional de polarización vertical.
- Rango de funcionamiento tri-banda: 144 MHz (~2m), 400 MHz (75 cm) y 1200 MHz (25 cm).
- Rango de recepción extendido: 118-160 MHz, 250-290 MHz, 360-390 MHz, 420-470 MHz, 820-960 MHz, 1260-1300 MHz.
- Ganancia:
 - 146 MHz 0dBi 1/4 wave
 - 446 MHz 0dBi 1/4 wave
 - 1200 MHz 3.4dBi 5/8 wave
- Potencia máxima: 10 watts
- Longitud: 17 cm (6.5 pulgadas)
- Conector: SMA



Figura 3.3-1 Antena VERT400. Extraída de [26].

4. Equipo lógico utilizado (*software*)

En este capítulo se va a explicar el *software* que se ha utilizado para llevar a cabo todo el procesado de la señal en banda base y que se corresponde una parte fundamental en la implementación del sistema SDR.

La empresa que comercializa el USRP, *Ettus Research*, ofrece soporte *software* con el objetivo de proporcionar, a la plataforma *hardware*, la mayor cantidad de posibilidades y flexibilidad en cuanto a entornos de desarrollo y a soporte de sistema operativo. En este sentido, encontramos el *USRP Hardware Driver* (UHD) que es el *driver*⁶ necesario para trabajar con el USRP en los sistemas operativos de Windows, Mac OS o Linux.

A su vez, *Ettus Research* recomienda también utilizar otros programas que son compatibles con la mayoría de USRPs y que sirven como potentes herramientas para el desarrollo de aplicaciones en sistemas SDR. Entre ellos están: *GNU Radio*, *LabVIEW*, *Matlab*, etc. En nuestro caso, se ha optado por utilizar *GNU Radio* como plataforma *software* principal para este proyecto final de carrera. *GNU Radio* tiene una gran comunidad activa, ofrece un servicio de *mainling list* donde se puede consultar cualquier duda y, además, cada año organiza un congreso (*GRCon*) donde se presentan ponencias de proyectos que se están llevando a cabo y se discuten los avances que se están logrando gracias a dicho *software*, entre otras muchas cosas.

Por tanto, y recapitulando, se utilizará *GNU Radio* como programa para realizar todas las operaciones de procesado de señal en banda base y para configurar el USRP y será el *USRP Hardware Driver* (UHD) el que maneje el USRP y posibilite la modificación de sus parámetros (frecuencia RF deseada, ganancia, etc.).

⁶ *Driver*: programa que controla un dispositivo.

4.1 *GNU Radio*

GNU Radio es un marco *framework*⁷ de desarrollo de *software libre* creado por Eric Blossom en 2001 y que tiene como principales objetivos proporcionar funciones de procesado de señal para implementar *Software Defined Radios* y apoyar la investigación de las comunicaciones inalámbricas y de los sistemas radio basados en tecnología SDR [27].

Por tanto, lo que tenemos es un conjunto de archivos y aplicaciones agrupados en librerías, que permiten manipular señales mediante procesado digital. A través de estas librerías se puede realizar el diseño de sistemas SDR. *GNU Radio* corre sobre los sistemas operativos Linux, Mac OS y Windows y puede ser usado conjuntamente con *hardware* externo como, por ejemplo, los dispositivos USRP, para implementar físicamente el sistema SDR, o bien, sin incluir *hardware*, para trabajar en un entorno de simulación.

La creación de una aplicación en *GNU Radio* se lleva a cabo mediante la programación en alto nivel, usualmente Python. Sin embargo, existe la posibilidad de utilizar una interfaz gráfica, llamada *GNU Radio Companion* (GRC), que permite realizar diseños completos sin escribir líneas de código, simplificando así el nivel de complejidad y eliminando la necesidad de conocimientos de lenguajes de programación por parte del usuario.

Para construir un sistema radio con *GNU Radio* se debe crear una especie de grafo, donde los vértices o nodos son los bloques de procesado de señal y las aristas o arcos que unen esos nodos representan el flujo de datos entre ellos. Conceptualmente un bloque procesa señales de forma continua, desde puertos de entrada hasta puertos de salida. Por consiguiente, un bloque viene determinado por el número de puertos de entrada, el número de puertos de salida y el tipo de datos que fluirán de un puerto al otro. De esta manera, se hace una distinción entre:

⁷ *Framework*: entorno o ambiente de trabajo para desarrollo; dependiendo del lenguaje normalmente integra componentes que facilitan el desarrollo de aplicaciones como el soporte de programa, bibliotecas, plantillas y más.

- Fuentes (*Sources*), caracterizadas por tener solo puerto de salida. Por ejemplo: un fichero, un micrófono, un dispositivo USRP, un generador de señal, etc.
- Sumideros (*Sinks*), caracterizados por tener solo puerto de entrada. Por ejemplo: un fichero, la tarjeta de sonido, un dispositivo USRP, visualizadores gráficos (*FFT sink*, *Oscilloscope sink*, *Constellation sink...*), etc.
- Bloques de procesado de señal, caracterizados por realizar un tratamiento de la señal. Por ejemplo: filtros, amplificadores, moduladores, operadores lógicos, etc.

En cualquier aplicación de *GNU Radio* siempre ha de haber algún tipo de fuente y algún tipo de sumidero. Los tipos de datos que se manejan son *byte* (1 byte de datos), *short* (2 bytes de datos), *int* (4 bytes de datos), *float* (4 bytes de datos para números en punto flotante) y *complex* (8 bytes de datos, un float para cada componente).

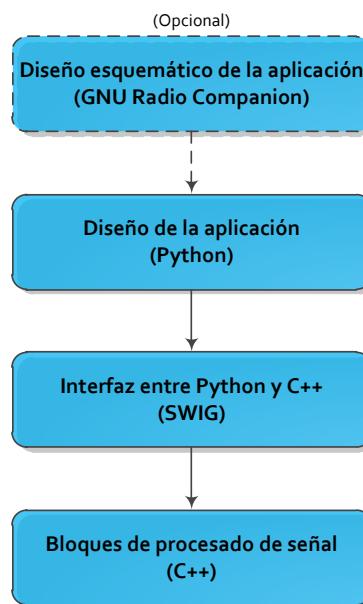


Figura 4.1-1 Arquitectura *GNU Radio*.

Los bloques de procesado de la señal se implementan en C++ y son convertidos a Python mediante el programa SWIG (*Simplified Wrapper and Interface Generator*). Esto quiere decir que, cuando se cree un módulo⁸ en Python

⁸ Módulo (referido al lenguaje de programación Python): es un archivo que contiene definiciones y declaraciones. El nombre del archivo es el nombre del módulo con el sufijo .py agregado.

para hacer una aplicación, este llamará a los métodos y funciones necesarios de los bloques implementados en C++. En la figura 4.1-1 se muestran las interacciones entre las diferentes capas o niveles de abstracción de la arquitectura *GNU Radio*.

4.1.1 *GNU Radio Companion* (GRC)

Una vez presentada la manera en que funciona el programa *GNU Radio*, se procede a explicar con más detalle su interfaz gráfica, *GNU Radio Companion*, ya que esta es la herramienta que se ha utilizado principalmente para implementar la cadena de transmisión y recepción del sistema SDR, tal como se explica en el apartado 5.3 de este documento.

La construcción de aplicaciones utilizando *GNU Radio Companion* se basa en añadir bloques e interconectarlos de manera esquemática. Para acceder a la herramienta, simplemente hay que llamarla desde un terminal de Linux escribiendo el comando *gnuradio-companion*.



```
israel@israel-VirtualBox: ~
israel@israel-VirtualBox: ~$ gnuradio-companion
linux; GNU C++ version 4.9.2; Boost_105400; UHD_003.010.git-80-g777352e7
<<< Welcome to GNU Radio Companion 3.7.9git-226-ge0a70a92 >>>
Preferences file: /home/israel/.gnuradio/grc.conf
Block paths:
    /usr/local/share/gnuradio/grc(blocks
    /home/israel/.grc_gnuradio
Showing: ""
```

Figura 4.1.1-1 Abriendo el programa *GNU Radio Companion*.

Una vez dentro, el aspecto de la interfaz se divide en cuatro partes:

- En la parte superior se encuentra la barra de herramientas, en donde aparecerán opciones para ejecutar/parar la aplicación, guardar el esquemático, etc.
- En la parte central está el *Workspace* que será el lugar donde se trabajará con los bloques elegidos y se creará la configuración deseada para la aplicación.
- En la parte derecha se sitúa la librería, donde aparece toda la lista de bloques de procesado de señal agrupados por categorías.

- Finalmente, en la parte inferior aparece un “terminal de la interfaz”, en donde irán apareciendo los *logs* o los procesos que lleve a cabo la aplicación una vez ésta sea ejecutada.

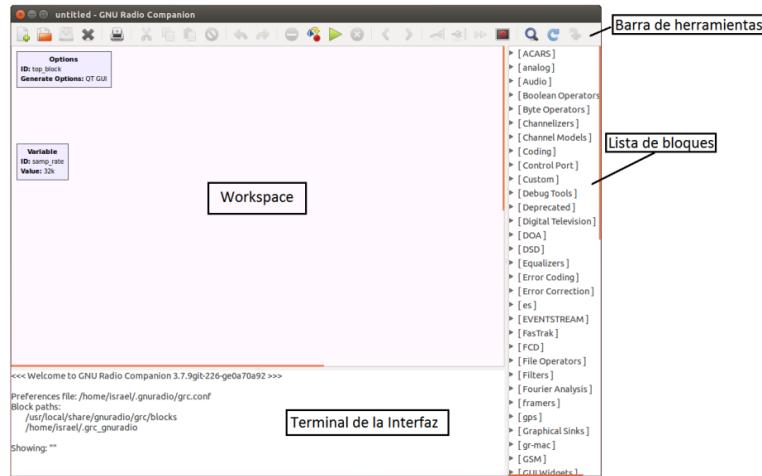


Figura 4.1.1-2 Apariencia de la interfaz GRC.

Para añadir un bloque en el esquemático basta con hacer doble clic sobre aquel elemento que se desea añadir. Dependiendo del bloque elegido, éste tendrá más o menos parámetros de configuración. Si se desea modificar los parámetros que vienen por defecto en el bloque, hay que hacer doble clic sobre él para que aparezca la ventana de configuración y se puedan cambiar los valores preestablecidos.

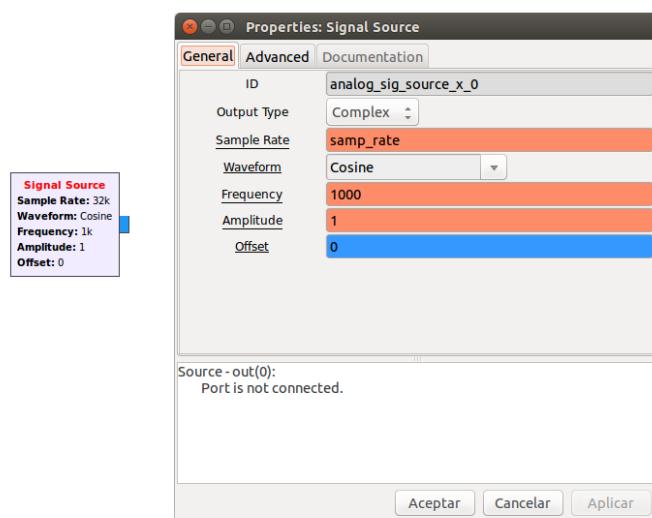


Figura 4.1.1-3 Parámetros por defecto del bloque *Signal Source*.

Una vez que se han incluido en el esquemático todos los bloques que se ha considerado pertinentes, lo único que queda por hacer es interconectarlos. Para interconectar bloques, únicamente es necesario seleccionar los bloques que se quieren unir siguiendo el orden que imponga la dirección en que se quiere que fluyan los datos.

Cuando se tiene el diseño completado (*flow graph*), éste se guarda en un archivo “*.grc” que al compilarlo generará un archivo “*.py”. De esta forma, se consigue automáticamente el código Python de cualquier aplicación que se diseñe de manera esquemática. Sin embargo, si se desea modificar la aplicación a partir del código generado, hay que ser consciente que cada vez que se ejecute la aplicación desde la interfaz (archivo “*.grc”), ésta sobrescribirá el *script* Python generado. Por lo que si se va a trabajar con el archivo “*.py” generado, lo más aconsejable es renombrarlo. Por defecto, si no se modifica el parámetro “ID” del bloque *Options* (bloque fijo que aparece en el *Workspace* y que permite modificar parámetros globales del programa), siempre que se ejecuta una aplicación mediante la interfaz se genera un archivo Python llamado *top_block.py*.

En síntesis, con *GNU Radio Companion* se puede llevar a cabo de forma simple el diseño de muchos sistemas/aplicaciones, ya que dispone de una amplia librería de bloques para el procesado de señal y, además, el proceso de diseño se lleva a cabo de una manera simple y comprensible, gracias a que la interfaz gráfica es intuitiva. Así mismo, el diseñador también se ahorra el trabajo de programar directamente en Python.

5. Implementación de los terminales cognitivos

En este capítulo se explicará el proceso que se ha llevado a cabo para implementar el demostrador de Radio Cognitiva. Para ello se crearán unos *scripts* que correrán en cada uno de los ordenadores y que permitirán la comunicación inalámbrica entre los dos URSP.

Puesto que la elaboración de dichos *scripts* está ligada al ciclo cognitivo (analizar el entorno, elegir el canal y transmitir), se ha dividido el capítulo en cuatro grandes bloques. Primeramente, un apartado destinado a explicar cómo se ha llevado a cabo el proceso de *Spectrum Awareness* en nuestro sistema, a continuación un segundo apartado para esclarecer los criterios utilizados en el proceso *Spectrum Selection* y, finalmente, dos últimos apartados destinados a describir el diseño y el protocolo de comunicación implementados.

5.1 Identificación de las oportunidades de acceso al espectro

Para realizar esta primera etapa, a partir del código *usrp_spectrum_sense.py* que se encuentra en la carpeta gr-uhd de *GNU Radio*, se ha creado un programa en Python llamado *Spectrum_Awareness.py*.

Spectrum_Awareness.py es un programa que analiza todo el espectro comprendido entre los 470 MHz y los 790 MHz y proporciona como resultado el nivel de energía de la señal que hay en cada radiocanal de TV. Posteriormente las estimaciones del nivel de energía que contiene cada radiocanal se comparan con un umbral preestablecido y se decide sobre la presencia (o ausencia) de usuarios primarios en el correspondiente radiocanal.

Debido a las limitaciones del USRP⁹, el análisis se hace en bloques de 25 MHz sobre una banda a observar que va desde 470 MHz a 790 MHz. La observación por bloques implica que no toda la banda de TV es evaluada al mismo tiempo. Sin embargo esto no es un problema grave debido a que las emisiones de TV tienen un comportamiento muy estático, es decir, si un canal está ocupado por una emisión de TV lo estará durante horas o incluso días, y si un canal está libre

⁹ Según las especificaciones del equipo N200 sólo se pueden observar 25 MHz de banda contigua, centrados alrededor de la frecuencia de sintonía elegida.

también lo estará durante horas o días, de modo que la evaluación simultánea de todos los radiocanales dentro de la banda de TV no es un problema relevante en nuestra aplicación.

A continuación, se explica de manera conceptual el funcionamiento del programa.

5.1.1 El Detector de Energía

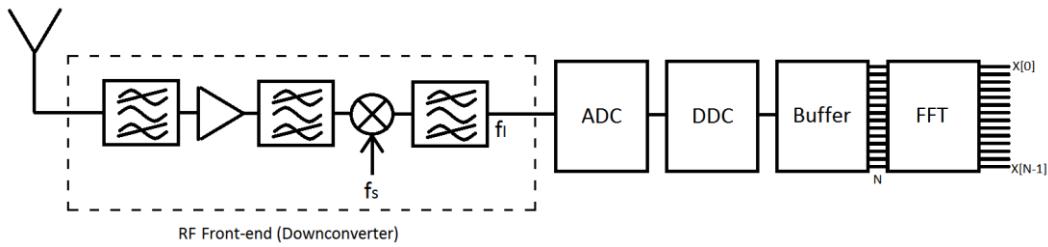


Figura 5.1.1-1 Bloques que participan en el programa *Sectrum_Awareness.py*.

En primer lugar la antena capta las ondas electromagnéticas¹⁰. Posteriormente, un filtro, sintonizado dentro de la banda de 50 MHz a 2.2 GHz, reduce el ancho de banda captado por la antena al ancho de banda que permite la *daughterboard* (25 MHz) y se amplifica la señal obtenida.

En este punto, se procede a trasladar la señal a frecuencia intermedia. El motivo es el siguiente. Si pasáramos directamente la señal captada por la antena a banda base, lo que ocurriría es que, después de que nuestra señal atravesara el bloque ADC, y como consecuencia del teorema de muestreo, veríamos que los espectros de señales centradas en armónicos de la frecuencia de sintonía ($2f_c$, $3f_c$, etc.) aparecerían también en banda base e interferirían con el espectro de la señal que queremos procesar (señal principal). Para evitar este problema, a la entrada del cabezal, con un filtro RF (filtro analógico), se elimina las señales colindantes a la zona del espectro que se quiere analizar/procesar, incluyendo por lo tanto señales centradas en múltiplos de la frecuencia de sintonía (f_c), antes de pasar a banda base, y posteriormente se traslada la señal recibida a una frecuencia intermedia (f_i).

¹⁰ Obviamente, la antena tiene su ancho de banda y, por tanto, sólo será capaz de captar el rango de frecuencias que se ajuste a sus especificaciones

Todo este proceso de filtrado y traslación del espectro de la señal a frecuencia intermedia (*dowconverter*) ocurre en el bloque cabezal de RF (*Front-end*) y, en resumidas cuentas, lo que nos va a quedar a su salida es una señal de 25 MHz de ancho de banda, como máximo, sintonizada a una frecuencia intermedia (f_i) que puede ser cero Hz si resulta conveniente. De “facto”, el componente que limita el ancho de banda de la señal es realmente el conversor ADC que viene a continuación, cuya tasa de muestreo es de $50 \cdot 10^6$ muestras/s. Por tanto, para no tener *aliasing* en las muestras de la señal muestreada, hemos de respetar el Teorema de Nyquist. Obsérvese que a la salida del conversor, vemos la señal que teníamos en banda base más todos sus alias (copias del espectro separadas entre ellas la frecuencia de muestreo). Por lo tanto el máximo ancho de banda a procesar sin *aliasing* es $B \leq 25$ MHz, tal como se indica en la figura 5.1.1-2.

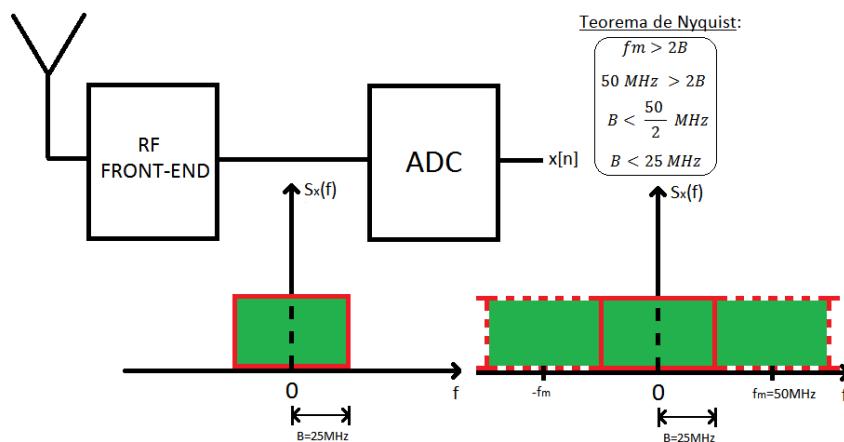


Figura 5.1.1- 2 Ejemplo del espectro de una señal al pasar por los bloques *Front-end* y ADC.

De esta manera, para observar toda la banda de TV, comprendida entre los 470 a los 790 MHz, el cabezal de RF se sintoniza progresivamente a frecuencia $f_c = (482,5 + 25*(N-1))$ MHz, con $N= 1, \dots, 12$. Es decir, y para entenderlo mejor, es como si un filtro fuera recorriendo la banda en pasos de 25 MHz, seleccionando una primera porción del espectro y muestreándolo, luego una segunda porción del espectro y pasando otra vez por el ADC, etc.

Una vez la señal está muestreada, se le aplica la FFT (Transformada Rápida de Fourier)¹¹ para darnos a una aproximación del contenido en frecuencia de la señal discreta.

Más concretamente, el cálculo de la Transformada Discreta de Fourier (DFT) de una secuencia $x[n]$ de N puntos se realiza según la siguiente ecuación:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{\frac{-j2\pi kn}{N}} \text{ para } k = 0, 1, 2, \dots, N-1$$

Si se define $W_N = e^{\frac{-j2\pi}{N}}$, la expresión anterior queda como:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{k \cdot n} \text{ para } k = 0, 1, 2, \dots, N-1$$

Por tanto, el cálculo requiere de una suma compleja de N multiplicaciones complejas para cada una de las salidas. En total, N^2 multiplicaciones complejas y $N(N-1)$ sumas complejas. El algoritmo FFT consigue simplificar el cálculo de la DFT y reduce el número de operaciones aprovechando las propiedades de simetría y periodicidad de los términos W_N . Eligiendo un valor de N de manera que $N=2^m$ resulta:

$$W_N^{n+N} = W_N^n \quad W_N^{k \cdot N} = 1$$

$$W_N^{n+N/2} = -W_N^n \quad W_N^2 = W_{N/2}$$

Las muestras cogidas de la señal se almacenan en un buffer que generará tantas salidas en paralelo como puntos tenga la FFT. Sería el equivalente de coger una ventana temporal de X muestras y en lugar de entregarlas en serie (dominio del tiempo), pasárlas en paralelo (dominio de la frecuencia).

El valor de los coeficientes obtenidos tras la FFT son los niveles de energía de la señal en N distintas frecuencias (realmente sub-bandas) equiespaciadas en la

¹¹ La Transformada Rápida de Fourier es un algoritmo especializado que permite a un procesador digital hacer el cálculo de la DFT (Transformada Discreta de Fourier) de una forma eficiente, en lo que respecta a carga computacional y tiempo de procesamiento. Puede ser aplicada cuando el número de muestras de la señal es una potencia de dos.

banda de observación de la FFT. Para que la anterior afirmación sea cierta existe un compromiso (*Trade-Off*) entre: puntos de la FFT (N), ancho de banda de la señal (B) y ancho de banda de resolución del canal o de la sub-banda (B_{res}). En efecto, fijados B y N, el ancho de banda de resolución B_{res} viene dado por: $B_{res} = \frac{B}{N}$, tal como se muestra en la siguiente figura:

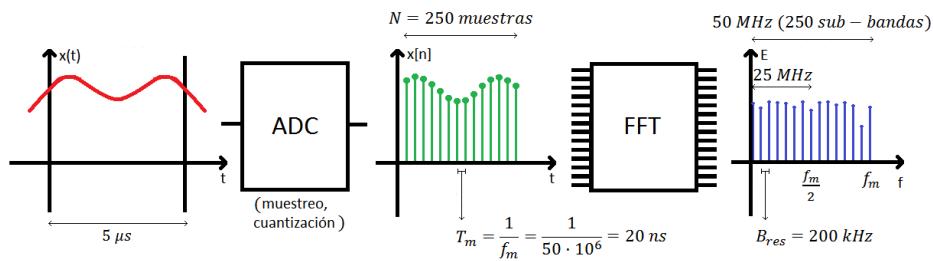


Figura 5.1.1-3 Ejemplo del paso de una señal a través de los bloques ADC y FFT.

Obsérvese que:

- El primer valor de la salida de FFT ($\text{bin}0 \equiv X[0]$)¹² es la frecuencia central de la señal paso banda sintonizada. Por ejemplo, si analizamos el bloque espectral de $25\ MHz$ de ancho de banda, comprendido entre $470\ MHz$ y $495\ MHz$, y consideramos una DFT de 250 puntos ($N = 250$), resulta una resolución de $B_{res} = 100\ kHz$. En este caso, la muestra $X[0]$ a la salida de la DFT corresponde a la energía del espectro concentrado alrededor de la frecuencia $482.5\ MHz$.
- La primera mitad de la FFT (de $X[1]$ a $X[(N/2)-1]$) contiene las frecuencias positivas de la señal paso banda sintonizada, que corresponde desde la frecuencia central a la frecuencia máxima de la zona espectral observada. Siguiendo con el ejemplo anterior: $X[1] \leftrightarrow 482.6\ MHz$, ..., $X[124] \leftrightarrow 495\ MHz$ (sobre unos $12.5\ MHz$ por encima de la frecuencia central).
- La segunda mitad de la FFT (de $X[N/2]$ a $X[N-1]$) contiene las frecuencias negativas de la señal paso banda sintonizada, que corresponden desde la frecuencia mínima hasta la frecuencia central. Análogamente, siguiendo con el ejemplo anterior: $X[125] \leftrightarrow 470\ MHz$, ..., $X[249] \leftrightarrow 482.4\ MHz$ (sobre unos $12.5\ MHz$ por debajo de la frecuencia central).

¹² Bin = barra del espectro

Obsérvese que garantizar que $B = N \cdot B_{res}$ también es crítico si queremos efectivamente obtener una representación espectral correcta de la zona del espectro observada. En efecto, si $N < \frac{B}{B_{res}}$, conceptualmente es como si al “bin” en el dominio de la frecuencia se le estuviera asignado una energía resultante del valor promedio de diversas muestras en el dominio del tiempo (véase figura 5.1.1-4), aunque en la práctica lo que ocurriría es que se descartarían a la entrada de la DFT diversas muestras de la señal en el dominio del tiempo y por lo tanto se perdería energía, resultando en una estimación errónea del nivel de energía del “bin” en el dominio frecuencial.

En el caso opuesto, es decir, $N > \frac{B}{B_{res}}$, lo que estamos haciendo es añadir muestras cero a la entrada de la DFT, cuando realmente estas muestras tendrían un cierto valor siguiendo la evolución de la señal de entrada (véase figura 5.1.1-4), de modo que nuevamente daría como resultado una estimación errónea del nivel de energía del “bin” en el dominio frecuencial.

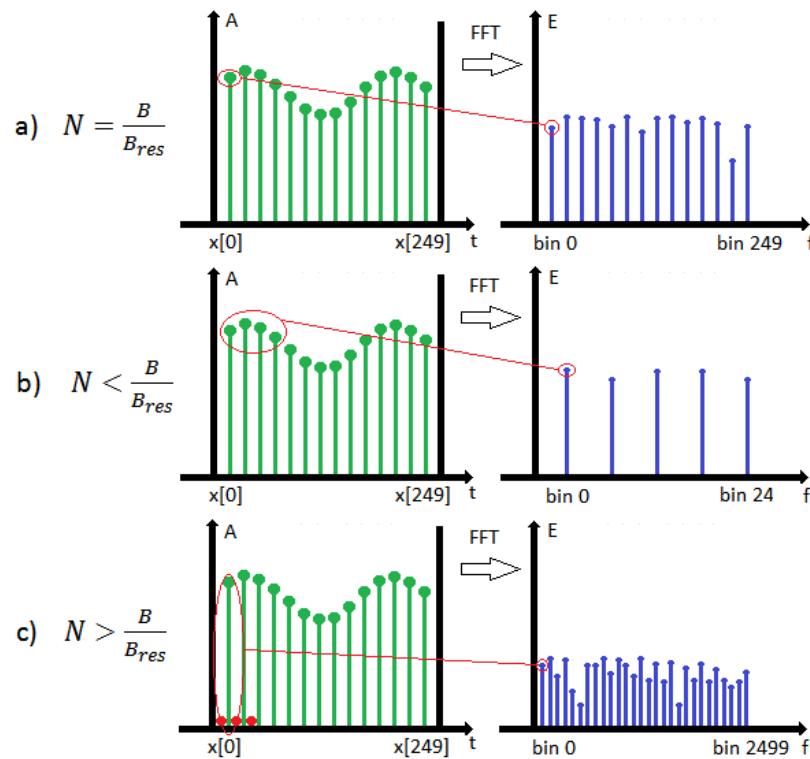


Figura 5.1.1-4 Ejemplo de los casos a), b) y c).

Para analizar toda la banda del espectro correspondiente entre los 470 MHz y 790 MHz, es decir, 320 MHz de espectro radioeléctrico, utilizando, por ejemplo,

una banda B de 25 MHz, una resolución B_{res} de 100 kHz y 250 puntos de la DFT (como en el caso analizado anteriormente), el USRP debería repetir 13 veces el sensado del espectro radioeléctrico, desplazando la banda de sensado en pasos de 25 MHz cada vez. Es decir, teóricamente, el dispositivo sintonizaría la frecuencia central de RF de la siguiente manera:

El primer paso utilizaría una frecuencia central de la banda de 482.5 MHz (cubriría la banda de frecuencias de 470 MHz a 495 MHz), el segundo paso sería 507.5 MHz (cubriría la banda de 495 MHz a 520 MHz),..., el duodécimo paso, la frecuencia central sería 757.5 MHz (cubriría la banda de 745 MHz a 770 MHz) y, finalmente, en el decimotercer paso sería 782,5 (cubriría la banda de 770 MHz a 795 MHz). Como el objetivo sería medir hasta la frecuencia de 790 MHz, se descartaría algunos puntos de la FFT en el último paso.

Sin embargo, en la práctica, no es posible coger bloques de 25 MHz. La no linealidad del filtro paso bajo del DDC (la respuesta en frecuencia del filtro no es plana de $-f_m/2$ a $+f_m/2$) obliga a utilizar una superposición de puntos FFT que permita estimar correctamente la energía en los bordes del análisis (470 MHz, 495 MHz, 520 MHz, 545 MHz, 570 MHz, 595 MHz, 620 MHz, 645 MHz, 670 MHz, 695 MHz, 720 MHz, 745 MHz, 770 MHz, 795 MHz en el ejemplo previo), ya que, de lo contrario, en estas frecuencias los valores de energía serían erróneos, pudiendo resultar, incluso, nulos.

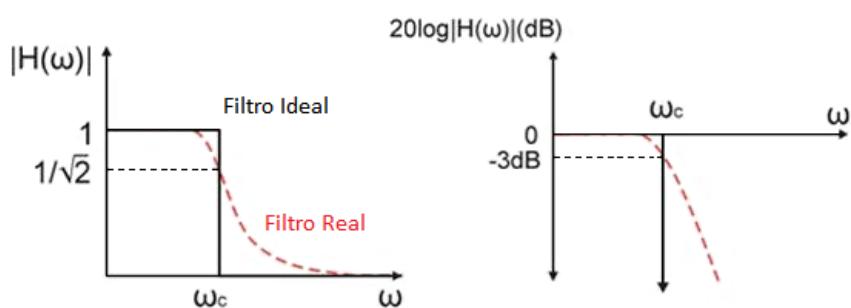


Figura 5.1.1-5 Respuesta en frecuencia del módulo y la fase de un filtro paso bajo.

Así pues, para compensar estos efectos, se opta por utilizar una superposición de puntos del 25%, que provoca que el tamaño de paso en el análisis frecuencial sea de 18.75 MHz ($25 \text{ MHz} * (1-0,25)$). Por consiguiente, el dispositivo

va a sintonizar la frecuencia central de la siguiente manera (en total, habrá 18 pasos):

El primer paso utiliza una frecuencia central de la banda de 479.375 MHz (cubrirá la banda de frecuencias de 470 MHz a 488.75 MHz), el segundo paso, de 498.125 MHz (cubrirá la banda de 488.75 MHz a 507.5 MHz),..., el decimoséptimo paso, de 779.375 MHz (cubrirá la banda de 770 MHz a 788.75 MHz) y, finalmente, el decimoctavo paso, de 798.125 MHz (cubrirá la banda de 788.75 MHz a 807.5 MHz). Como se quiere solo hasta la frecuencia 790 MHz, se descartarán los puntos de la FFT correspondientes (de 790 MHz a 807.5 MHz).

Por otra parte, además de los parámetros estudiados anteriormente, en el análisis del espectro hay que utilizar otros parámetros importantes, que son los tiempos *Tune Delay* (t_d) y *Dwell Delay* (t_n).

- *Tune Delay* (t_d): Es el período de tiempo en el que las muestras de FFT son descartadas por el cabezal de radiofrecuencia (*Front-end*) cada vez que se sintoniza una nueva frecuencia central. Esto se hace para tomar en consideración la existencia de estados transitorios que ocurren en los dispositivos de RF cada vez que se cambian sus parámetros y/o modos de operación. Es decir, cuando mandamos la orden a la *daughterboard* del USRP para que este cambie de frecuencia central, tenemos que esperar t_d segundos hasta que las muestras del ADC (sin efectos de los transitorios) lleguen al bloque de la FFT. Esto se hace simplemente descartando las muestras recibidas entrantes durante un tiempo de *Tune Delay* especificado.
- *Dwell delay* (t_n): Es el período de tiempo durante el cual, para cada frecuencia central, se calculan los valores (potencia) medios en cada bin de la FFT. Esta operación se realiza después de descartar las muestras en el *Tune Delay*.

En la práctica, para implementar un buen detector de energía, que ofreciera de manera aproximada y fiel valores de potencia comparables a los que se podían medir en el laboratorio con un analizador de espectros, se ha elegido los siguientes valores para los parámetros fundamentales del programa:

$$\triangleright \min_freq = 470 \cdot 10^6$$

- $\max_freq = 790 \cdot 10^6$
- $\text{sample_rate} (f_m \equiv B) = 25 \cdot 10^6$
- $\text{channel_bw} (B_{res}) = 24.414 \cdot 10^3$
- $\text{fft-size} (N) = 1024$
- $t_d = 0.25$ segundos
- $t_n = 0.25$ segundos
- Se considera superposición de puntos de la FFT del 25%

5.1.2 Criterio de decisión: Canal Libre/Canal Ocupado

A la salida del bloque FFT obtenemos el espectro¹³ de la señal. Para poder decidir sobre la disponibilidad, o no, de un canal, lo que se hace es comparar la potencia de la señal recibida en el radiocanal con un umbral predeterminado.

Este método de detección de radiocanales libres y/o ocupados es uno de los más utilizados debido a su generalidad y a su facilidad de implementación y, también, a los bajos requerimientos computacionales que precisa [28]. Puesto que vamos a detectar la presencia de señales primarias, sólo cuando su energía se encuentre por encima del umbral de detección, la selección de un valor apropiado para el umbral resulta crítica, ya que además de la señal primaria va a existir un cierto nivel de ruido y potencialmente la presencia de señales interferentes que pueden ocasionar errores en la detección. En nuestro caso, como el entorno donde se va a realizar el demostrador de Radio Cognitiva es un entorno estable (el laboratorio del Grupo de Investigación de Comunicaciones Radio), podremos estimar tanto el nivel de ruido como el de interferencia y, por ende, fijar un valor de umbral lo más óptimo posible.

Dado que un múltiplex de la TDT ocupa 8 MHz y hay 320 MHz de espectro radioeléctrico por analizar, el criterio de decisión se debe aplicar a 40 canales. Puesto que el espectro de la señal recibida va a estar, en general, afectada por la presencia de desvanecimientos selectivos en frecuencia¹⁴, en lugar de comparar con el umbral considerando la potencia total del radiocanal de 8 MHz, lo que se hace es

¹³ De hecho, es la densidad espectral.

¹⁴ La presencia de desvanecimientos selectivos en frecuencia puede dar lugar a pérdidas relevantes del nivel de potencia del radiocanal, un aspecto realmente importante cuando se trabaja con radiocanales ocupados por un usuario primario que llega con niveles muy pequeños de potencia.

seleccionar los 8 MHz (del bloque de 25 MHz) que corresponden a un radiocanal y contar cuantas muestras (bins) hay en esos 8 MHz, descartar el 10% las muestras (5% a cada lado¹⁵) y empezar a comparar la potencia de cada muestra de la señal con el valor umbral. Si la potencia media de la muestra es superior al valor del umbral, se incrementa el valor de un contador. De esta forma, si hay un determinado % de muestras que superan el valor del umbral, se decide que el canal está ocupado, en caso contrario, se decide que el canal está libre¹⁶.

En realidad, después de implementar este criterio, se comprobó que en ocasiones, la decisión era incorrecta (canales que estaban libres aparecían ocupados y viceversa). El motivo es que, al tratarse de un criterio binario, era muy factible dar resultados erróneos cuando en un canal ocupado la señal primaria recibida tiene un nivel muy bajo o bien en el caso de un canal libre que presente un nivel de ruido más interferencia relativamente alto. La solución a este problema fue añadir en el proceso de decisión dos estados más: Dudososo Ocupado y Dudososo Libre).

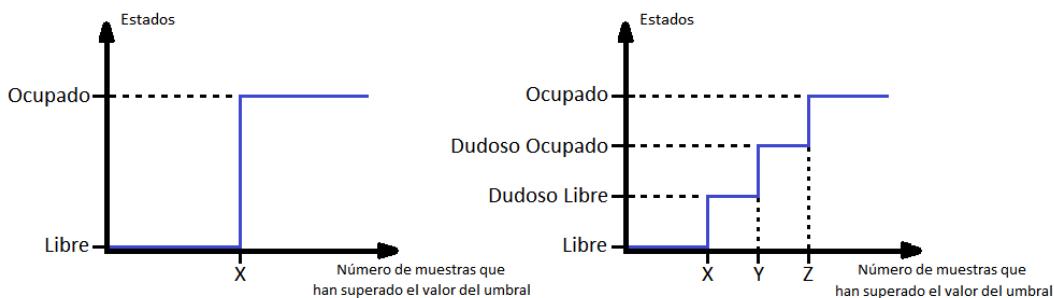


Figura 5.1.2-1 A la izquierda, criterio de decisión con 2 estados. A la derecha, criterio de decisión con 4 estados.

Por otra parte, emplear el criterio de los 4 estados no implica absoluta seguridad de que la lista de ocupación de canales obtenida corresponda con la regulación teórica del espectro en el punto de observación, es decir, que detectemos todos los canales que teóricamente tienen licencia para transmitir en el entorno

¹⁵ Se descartan el 5% de las muestras en ambos extremos de la banda estudiada ya que estas muestras corresponden a la banda de guarda de la señal de TV y por lo tanto no son muestras de señal realmente, sino de ruido y otras posibles señales espúreas.

¹⁶ Se considera un cierto % y no el número total de muestras evaluadas ya que la presencia de desvanecimientos selectivos en frecuencia, que pueden ser muy importantes en ciertas regiones del espectro, puede ocasionar que un número relativamente significativo de muestras estén por debajo del umbral aun existiendo un nivel de señal primaria significativamente relevante en el canal.

observado como ocupados y al resto como libres. La demostración se hace en un laboratorio situado en un primer piso de un edificio del Campus Nord de UPC. Por tanto, es posible que haya señales de televisión cuya potencia de recepción en el tejado del edificio (en donde realmente estaría una antena de TV) sea relativamente baja y que, en consecuencia, en la antena del USRP (situada en el interior de un laboratorio ubicado en el primer piso del edificio) se capten como ruido. En ese caso, el canal será designado como Libre (o Dudosamente Libre). Obsérvese que, a efectos prácticos, esta decisión no cabe considerarla como errónea, ya que efectivamente el radiocanal puede no ser percibido como ocupado en el interior del edificio debido a las pérdidas por penetración en edificios y, por lo tanto, ser utilizado para comunicaciones en interiores (*indoor*) sin que ello afecte a la calidad de la recepción de TV mediante una antena externa situada en la azotea del edificio. En resumen, durante la demostración del concepto de Radio Cognitiva puede ocurrir que canales, que teóricamente están ocupados, aparezcan en el análisis como libres y puedan ser utilizados como canal de comunicación *indoor* entre los dos USRPs durante la demostración.

Finalmente, para implementar el criterio de decisión, se ha elegido:

- Como umbral: un valor de potencia de -90 dBm¹⁷.
- Como primer umbral de decisión (estado: Libre): que el contador esté entre 0 y 74 muestras (equivalente inferior al 25% de las muestras posibles¹⁸).
- Como segundo umbral de decisión (estado: Dudosamente Libre): que el contador esté entre 75 y 148 muestras (equivalentemente entre el 25 y 50 % de las muestras posibles).
- Como tercer umbral de decisión (estado: Dudosamente Ocupado): que el contador esté entre 149 y 221 muestras (equivalentemente entre el 50 y 75 % de las muestras posibles).
- Como cuarto umbral de decisión (estado: Ocupado): que el contador esté por encima de 221 muestras (equivalente superior al 75% de las muestras posibles).

¹⁷ Valor promedio resultante de haber calibrado los USRPs utilizando una carga adaptada en bornas de la antena receptora y medido el nivel de ruido generado por los propios USRPs.

¹⁸ El número máximo de muestras posibles a contar en un canal de 8 MHz (una vez despreciado el 10% de la banda) es 295.

Con esto finaliza la descripción correspondiente a la implementación de la etapa de *Spectrum Awareness* del ciclo cognitivo. A modo de ejemplo de esta etapa, a continuación se ofrece una figura donde se puede ver la salida del programa *Spectrum_Awareness.py* una vez ha sido ejecutado en uno de los ordenadores del laboratorio. En la misma figura también se ha representado lista de la ocupación de los canales de la banda de TV obtenida aplicando el criterio descrito anteriormente.

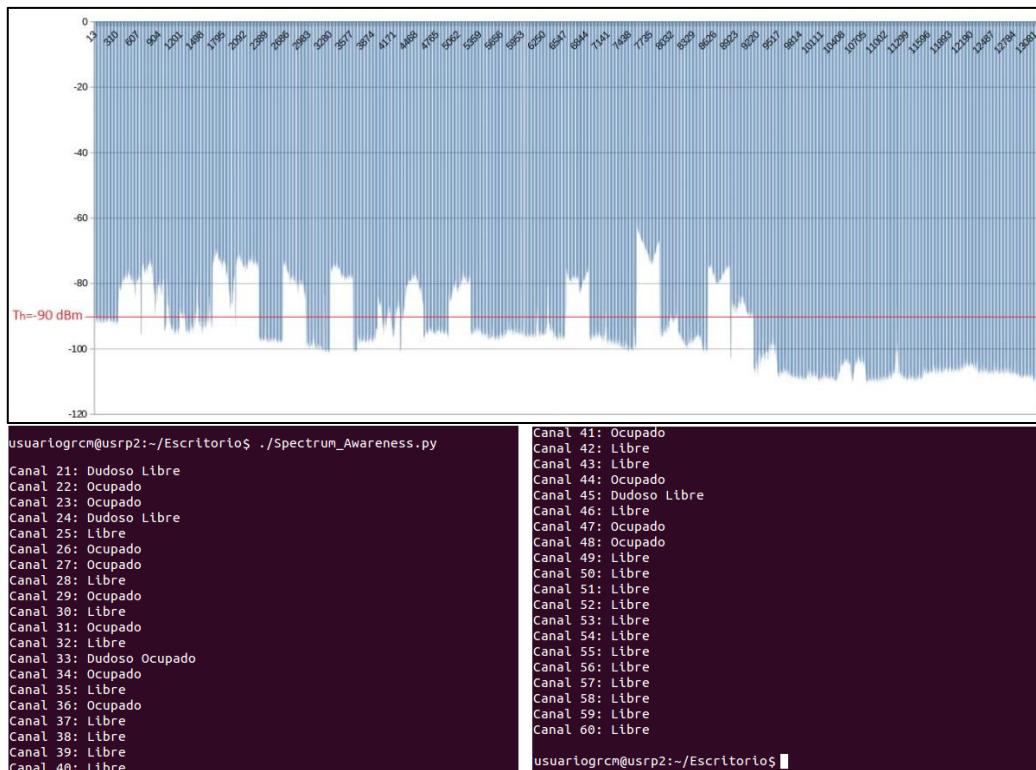


Figura 5.1.2-2 Arriba, representación gráfica del espectro analizado con el programa *Spectrum_Awareness.py*. El eje horizontal representa el número de la muestra y el eje vertical el nivel de potencia de la señal (en dBm). Abajo, lista de la ocupación de los canales de la banda de TV según el criterio del programa *Spectrum_Awareness.py*.

5.2 Asignación del espectro

Una vez se conoce qué canales hay disponibles en la banda de TV, se ha de decidir cuál de ellos será el canal por el que el usuario cognitivo transmitirá. Aunque en general la decisión de asignación del espectro será un procedimiento automatizado, en este caso vamos a dejar que la decisión la tome el usuario cognitivo, ya que lo que se pretende con el demostrador es visualizar las

funcionalidades del ciclo cognitivo y realizar una demostración de la comunicación entre dos USRPs, configurados como radios cognitivas.

En esta situación no habrá un análisis espectral previo que ayude al SU a elegir el TVWS que le proporcione las mejores prestaciones. En este caso, el usuario tendrá como información la potencia media de cada canal y podrá elegir únicamente entre aquellos canales que tengan el estado “Libre”¹⁹. Obsérvese que esta información es suficiente para tomar una decisión ya que los huecos son muy estables (periodos de canal libre u ocupado muy grandes) y por lo tanto la potencia media del canal es un buen indicador de su estado.

El hecho de que sea el usuario cognitivo el que elija el canal, es una decisión tomada con objeto de dar más dinamismo a la presentación de la demostración. Perfectamente se podría haber creado un programa que eligiera aleatoriamente el canal de entre aquellos que estuvieran libres o se podría haber optado por algún otro criterio (por ejemplo elegir el canal con menor nivel de señal detectado, tener en cuenta que el canal elegido tuviera otros canales adyacentes libres, etc.).

Los canales que se encuentran libres son:

[25, 28, 30, 32, 35, 37, 38, 39, 40, 42, 43, 46, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60]

Elija el canal en el que quiere operar:

Figura 5.2-1 La elección del canal la toma el usuario cognitivo.

5.3 Diseño del Sistema Digital

Llegados a este punto, tenemos como objetivo diseñar un sistema de comunicación inalámbrico que permita transmitir los archivos necesarios para que se pueda establecer finalmente una comunicación entre el usuario cognitivo y su destinatario (USRPs), utilizando alguno de los principios de la Radio Cognitiva, como es la utilización oportunista de un canal.

¹⁹ Obsérvese que los canales declarados como Libres Dudosos no son considerados en esta demostración como posibles canales cognitivos, ya que en general existen muchos canales libres y sólo hay que elegir un único canal.

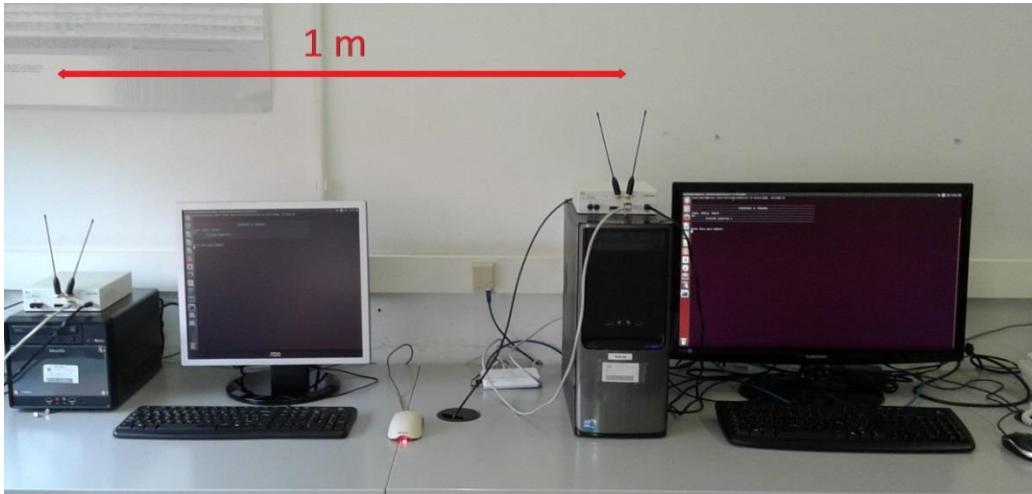


Figura 5.3-1 Escenario del sistema de comunicación inalámbrico. Foto tomada en el laboratorio D4112 de la UPC.

En el proceso previo a la comunicación, es decir, en el proceso de establecimiento de la comunicación, el usuario primeramente deberá enviar una petición de sensado a la Estación Base y esperar a que ésta le envíe la información sobre el estado de la banda de TV, para saber qué canales aparecen como libres (TVWS) y son opciones a utilizarse en la comunicación o, por otro lado, qué canales aparecen como ocupados y, por tanto, hay que vigilar, tanto como sea posible, de no interferir en su emisión²⁰. Una vez el usuario cognitivo tenga esta información, que estará en un archivo cuyo nombre será *tabla de canales*, éste deberá elegir uno de entre todos los canales que haya disponibles. Dicho canal, será el canal que, cuando se establezca la comunicación entre el usuario cognitivo y su destinatario, se utilizará de manera temporal y oportunista. El siguiente paso al que se procederá, será enviar al receptor la información del canal elegido por el usuario secundario; en este caso, el archivo tendrá como nombre *canal de operación*. Una vez el receptor obtenga correctamente este archivo y sepa la frecuencia a la que ha de operar, se dará por finalizado el proceso de establecimiento de la comunicación y se llevará a cabo, por parte del SU, la transmisión de una cantidad determinada de paquetes.

²⁰ Obsérvese que, a efectos de demostración, la Estación Base actúa como un pseudo-REM, ya que recibe una petición del terminal y le informa de los canales libres. No obstante no puede considerarse un verdadero REM ya que, entre otras informaciones a proporcionar al REM, el terminal no dice ni su posición ni sus características de TX/RX que finalmente van a limitar qué canales de los libres en su zona de operación se van a utilizar.

Cuando el SU finalice de transmitir, abandonará el TVWS; de la misma manera, cuando el receptor de la Estación Base obtenga todos los paquetes dejará el canal²¹.

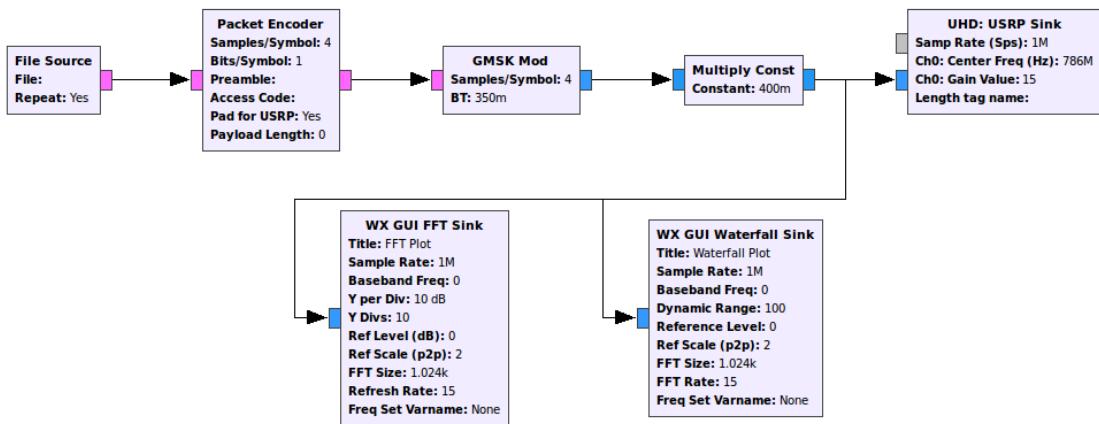
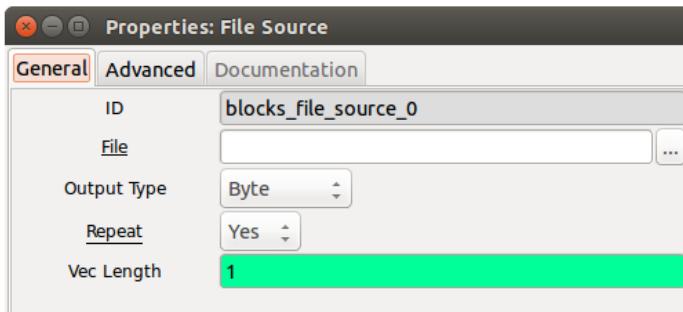
Las transmisiones llevadas a cabo serán transmisiones *half-duplex*, dado que el modo de envío de información será bidireccional pero no simultáneo. Por otra parte, hasta que se establezca el canal de comunicación de datos de usuario, cualquier comunicación entre el SU y la Estación Base se realizará a través de un canal común de control. Para este fin se ha elegido el canal 60 (comprendido entre 782-790 MHz) de la banda de TV, ya que, por normativa, se sabe que dicho canal siempre está libre. Por tanto, los archivos *tabla de canales* y *canal de operación* se enviarán y recibirán por la frecuencia 786 MHz, que es la frecuencia central del canal 60 y, posteriormente, cuando se establezca la comunicación, los USRPs operarán en la frecuencia elegida por el SU.

Para diseñar la cadena transmisora y receptora que utilizarán tanto el usuario secundario, como la Estación Base, se ha utilizado la herramienta *GNU Radio Companion*, presentada en el Capítulo 4. Así pues, a continuación se va a describir ambos diseños y se va a explicar paso a paso los bloques elegidos en cada uno de ellos.

5.3.1 Transmisor

El esquema diseñado para el transmisor es el mostrado en la figura 5.3.1-1. Dicho esquema está formado por una fuente (*File Source*), 3 bloques de procesado de señal (*Packet Encoder*, *GSMK Mod* y *Multiply Const*) y 3 sumideros (*USRP Sink* y 2 visualizadores gráficos: *FFT Sink*, *Waterfall Sink*).

²¹ En este demostrador básico no se han previsto mecanismos de repetición (ARQ) dada la corta distancia entre TX y RX que garantiza una SNR suficientemente grande para que los errores introducidos por del canal (LOS) sean despreciables o incluso nulos.

Figura 5.3.1-1 *Flow graph* de la cadena de transmisión.Figura 5.3.1-2 Parámetros del bloque *File Soure*.

El bloque *File Source* lee los bits de un archivo especificado. Por tanto, uno de los parámetros que hay que configurar de este bloque es la ruta (*path*) donde se encuentra el archivo en nuestro ordenador. Otro de los parámetros es el de “*Repeat*”, que permite que, una vez se hayan leído los bits del archivo y entregado al puerto de salida, se pueda repetir la acción una y otra vez, por tanto, permite una continua retransmisión de los datos. La opción “*Vec. Length*” define la longitud del vector de salida. Normalmente se deja a 1 para que la salida sea un flujo de bytes individuales. Y finalmente está el parámetro “*Output Type*” para definir el tipo de dato que se desea a la salida. Este parámetro, o el parámetro “*Input Type*”, lo encontraremos en casi todos los bloques y normalmente se dará a elegir entre las opciones *complex*, *float*, *int*, *short* o *byte*²². Se puede saber inmediatamente el tipo

²² Es importante saber que en *GNU Radio* existen tres tipos de datos que “equivalen” a bits: *Byte*, *Char* y *Unsigned Char (UChar)*. El tipo *Char* es un byte con signo, cuyo valor numérico va entre -128 y 127 (aritmética complemento a 2). El tipo *UChar* es un carácter sin signo, por lo que su valor va entre 0 y 255. Por último, el tipo *Byte* se puede considerar exactamente lo mismo que el tipo *UChar*.

de dato que se transmite entre bloque y bloque gracias al color del puerto de entrada o del puerto salida de cada bloque.

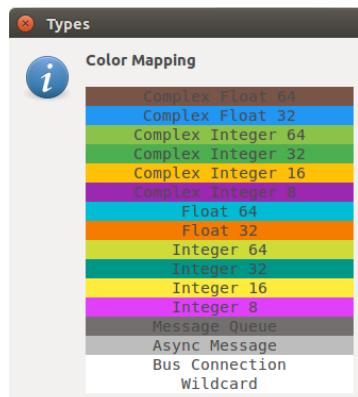


Figura 5.3.1-3 Asignación de los colores de los puertos de un bloque según el tipo de dato.

Así pues, elegiremos el archivo que queremos transmitir (en nuestro caso será el archivo *tabla de canales* o el archivo *canal de operación*) y el bloque *File Source* extraerá un tren de bytes, pertenecientes al archivo elegido, por el puerto de salida. Los bytes entrarán al bloque *Packet Encoder*.

El *Packet Encoder* agrega los bytes en una trama. Esta trama está formada por un preámbulo, un código de acceso, una cabecera (longitud del *payload* más offset), el *payload* (carga útil), un CRC-32²³, y el octeto \x55 que se utiliza para marcar el final de la trama. En la siguiente figura se muestra la estructura de dicha trama²⁴:

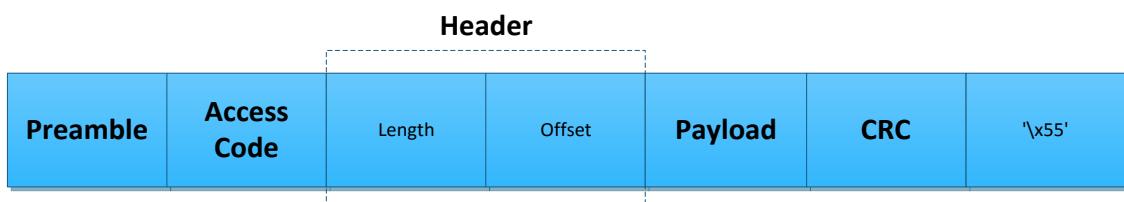


Figura 5.3.1-4 Trama generada en el *Packet Encoder*.

Esto es importante a la hora de convertir tipos (con los bloques de la categoría *Type Converters*). Por ejemplo, no existe un conversor de tipo *Float* a *Byte*, pero sí de *Float* a *UChar*.

²³ Verificación por redundancia cíclica (CRC): Código de detección de errores para detectar cambios accidentales en los datos.

²⁴ Para más detalles, puede consultar el método *make_packet* definido en el archivo *packet_utils.py*; por defecto, se encuentra en /usr/local/lib/python2.7/dist-packages/gnuradio/digital.

Uno de los objetivos de este bloque es ayudar en la sincronización de la trama y esto se consigue gracias al preámbulo (una secuencia de bits) que permite al receptor detectar el inicio de una transmisión y sincronizar relojes. El bloque además incluye un código de acceso, cuya función es descartar la trama si ocurren más errores del prefijado, esto se realiza mediante el *threshold* del bloque *Packet Decoder* del receptor. Si estos parámetros se dejan en blanco, se configuran los valores por defecto: 0xA4F2 en el preámbulo y 0xACDDA4E2F28C20FC en el código de acceso, en cuyo caso el *Packet Decoder* descartará la trama si aparecen más de 12 errores.

El *Packet Encoder* es un bloque que se utiliza junto con bloques moduladores, como son *GMSK Mod*, *DPSK Mod* o *QAM Mod*. El parámetro “*bits/symbol*” hace referencia a ello y se configura de la siguiente manera:

- gmsk → 1
- dbpsk → 1
- dqpsk → 2
- d8psk → 3
- qam8 → 3
- qam16 → 4
- qam64 → 6
- qam256 → 8

Por otra parte, el parámetro “*samples/symbol*” indica cuántas muestras se tomarán por símbolo, mientras que el parámetro “*payload length*” sirve para configurar la longitud que se desea del *payload* (el rango permitido está entre 0 y 4096 bits), si se deja a 0, la longitud se asigna automáticamente. Finalmente, el parámetro “*pad for URSP*” tiene como posible opciones de configuración “Yes” o “No”, si se utiliza la condición afirmativa se permite que el programa rellene de 0’s las tramas para que estas terminen en un múltiple de 512 bytes. Debido a que el *Packet Encoder* lo que hace es añadir un entramado a la información que se quiere transmitir, es lógico que salgan más bytes de los que entran a dicho bloque y esto es importante saberlo para tener en cuenta la tasa de generación de símbolos.

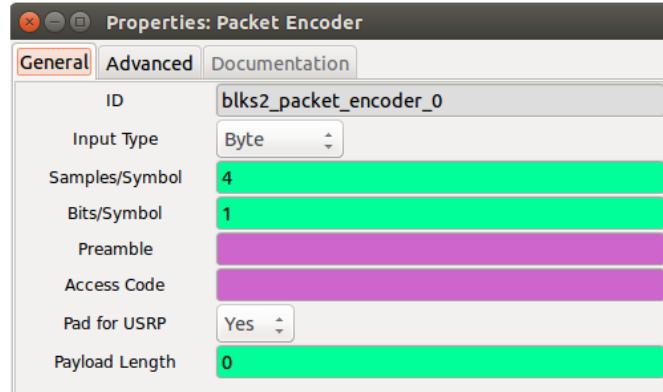


Figura 5.3.1-5 Parámetros del bloque *Packet Encoder*.

Una vez se ha configurado el *Packet Encoder* tal y como aparece en la figura 5.3.1-5, los bytes entran al bloque modulador *GMSK Mod*.

El bloque *GMSK Mod* es un bloque jerárquico, es decir, un bloque hecho con otros bloques. Los bloques jerárquicos añaden un nivel más de abstracción, permitiendo englobar un conjunto numeroso de bloques en solamente uno y simplificando la tarea de diseño. Por ejemplo, si no se utilizara un modulador genérico, como ocurre en este caso, habría que diseñar el modulador a partir de los siguientes bloques: *Packet to Unpacket* → *Map* → *Chunk to Symbols* → *Polyphase Arbitrary Resampler*.

Básicamente, GMSK (*Gaussian Minimum Shift Keying*) es un tipo de modulación derivada de la modulación MSK (*Minimum Shift Keying*). Se trata de una técnica que consigue suavizar las transiciones de fase y frecuencia entre estados de la señal, logrando así reducir los niveles de lóbulos laterales y, en consecuencia, los requisitos de ancho de banda. Concretamente, los bits de entrada representados de forma rectangular (± 1) son transformados en pulsos gaussianos (se suavizan las transiciones entre bits) mediante un filtro gaussiano (filtro de pre-modulación), para posteriormente ser entregados a un modulador de frecuencia. El filtro gaussiano suaviza los cambios de frecuencia en la señal MSK (véase figura 5.3.1-6). Esto tiene el efecto de reducir considerablemente los niveles de los lóbulos laterales en el espectro transmitido.

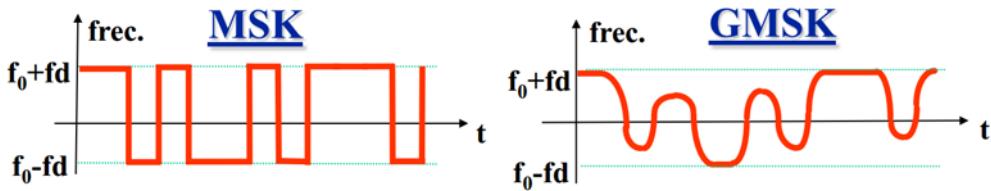


Figura 5.3.1-6 A la izquierda, símbolos en una modulación MSK. A la derecha símbolos en una modulación GMSK. Extraída de [29].

En la mayoría de los casos, la duración del pulso Gaussiano supera la duración de un símbolo, dando lugar a lo que se conoce como ISI (Interferencia Intersimbólica). El grado de esta superposición es determinado por el producto del ancho de banda del filtro Gaussiano y la duración de un símbolo, al que se le conoce normalmente como producto BT. BT está comprendido entre $0 < BT < 1$. Cuanto menor sea el valor de BT mayor será el solapamiento entre pulsos Gaussianos. La portadora resultante es una señal continua en fase, lo cual es importante porque las señales con transiciones suaves entre fases requieren menor ancho de banda para ser transmitidas. Por otra parte, este suavizado de la señal hace que el receptor tenga que realizar un trabajo mayor en la demodulación de la señal, ya que las transiciones entre bits no están bien definidas²⁵.

Por tanto, y recapitulando, se ha elegido emplear GMSK como técnica de modulación en el diseño del transmisor porque utiliza un ancho de banda razonablemente limitado en transmisión, ofrece robustez a la señal en medios hostiles, es comúnmente utilizada en entornos móviles (GMS) y presenta envolvente constante, por lo que es insensible a las no linealidades que puedan presentar los amplificadores de potencia.

De esta forma, al bloque *GMSK Mod* le entran un flujo de bytes y éste extrae a su salida la señal modulada compleja en banda base. Esta señal será enviada al USRP, que en la herramienta *GNU Radio Companion* se representará colocando el bloque *UHD: USRP Sink* como bloque sumidero.

²⁵ La degradación en términos de SNR en el proceso de detección de la señal GMSK con respecto a una señal 2-FSK es del orden de 0.5 a 1 dB, siendo estos valores absolutamente asumibles en una transmisión, tanto más cuanto se consideran las ventajas aportadas por la señal GMSK: mayor robustez frente a no linealidades y mejor comportamiento en términos de lóbulos laterales.

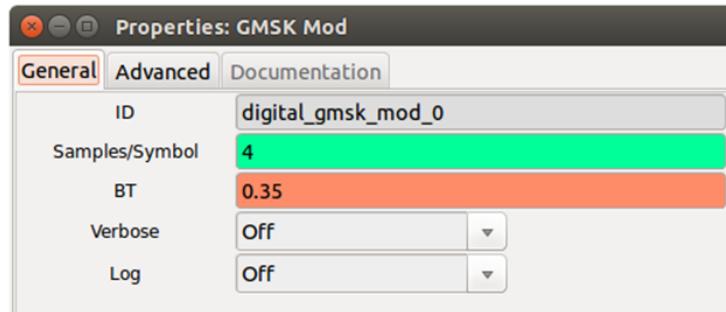


Figura 5.3.1-7 Parámetros del bloque *GSMK Mod*.

El *UHD: USRP Sink* tiene varios parámetros, pero los que nos suscitan más interés son los siguientes tres:

- “Samp Rate (Sps)”: Es la tasa a la que el USRP le pasa muestras al conversor DAC.
- “Ch0: Center Freq (Hz)": Es la frecuencia portadora de la señal (frecuencia de operación).
- “Ch0: Gain Value”: Es la ganancia del amplificador de bajo ruido que está justo antes de la antena (*PGA* de la *daughterboard*). Podemos diferenciar entre ganancia baja (0dB), media (15dB) y alta (25dB).

Por otra parte, en este bloque se observa que en el parámetro “*Input Type*” nos dan a elegir entre “*Complex float 32*” o “*Complex int 16*”. Esto lo que significa es que, o se van a utilizar 16 bits para representar cada muestra I y cada muestra Q (caso de *Complex float 32*), o bien, que se van a utilizar 8 bits para cada una de ellas (caso de *Complex int 16*). Dependiendo de qué caso se escoja, el máximo *sample rate* soportado por el dispositivo USRP será distinto. En nuestro caso, el USRP N200 permite un *sample rate* máximo de 25 Msps si se utilizan 16 bits por muestra y 50 Msps si se utilizan 8 bits por muestra.

Para realizar la comunicación con el USRP se ha de indicar en el parámetro “*Device Address*” la IP del dispositivo, siguiendo el formato: ‘addr=IP’. Si se deja en blanco, se utilizará el primer dispositivo USRP que encuentre el *software UHD* conectado al ordenador.

Los demás parámetros no son relevantes para nuestro diseño de transmisor, por tanto, no los tocaremos y dejaremos que actúen los valores por defecto. Los parámetros a los que nos referimos son: “*Sync*”, “*Clock Rate (Hz)*”, “*Num Mboards*”,

“*Mb0: Clock Source*”, “*Mb0: Time Source*”, “*Mb0: Subdev Spec*” y “*Num Channels*” que sirven para especificar la sincronización del dispositivo y para configurar un sistema MIMO (más de un USRP a la vez); al parámetro “*Ch0: Antenna*”, que sirve para especificar con qué antena del USRP se quiere trabajar, y al parámetro “*Ch0 Bandwidth (Hz)*”, que sirve para configurar el ancho de banda de los filtros de la *daughterboard* (si los hubiera).

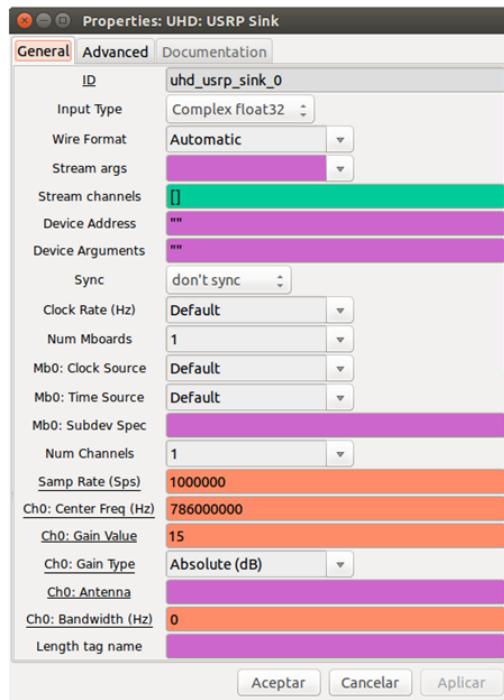


Figura 5.3.1-8 Parámetros del bloque *UHD: USRP Sink*.

Por último, es necesario aclarar que la señal que entra al USRP debe tener una magnitud máxima de 1 voltio. De lo contrario, el DAC, que presenta un rango de voltaje de -1 a +1 V, acabará saturado y la señal podrá verse comprimida. Para controlar este efecto se hace uso de un bloque multiplicador (*Multiply Const*) que se colocará entre el bloque *GMSK Mod* y el bloque *UHD: USRP Sink*.

Llegados a este punto, donde ya se han elegido los bloques que conformarán el transmisor y los valores de sus parámetros, se va a calcular la velocidad de transmisión de la señal y su ancho de banda para comprobar que el diseño es factible.

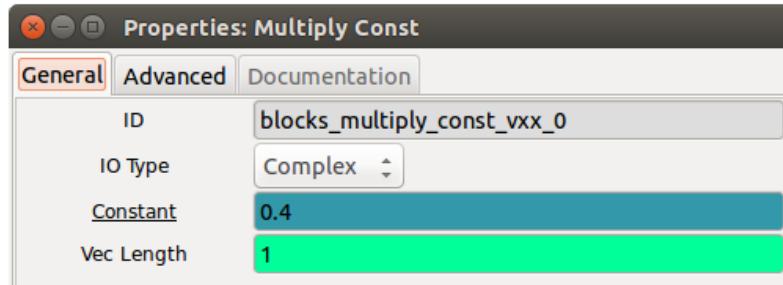


Figura 5.3.1-9 Parámetros del bloque *Multiply Const*.

La velocidad de transmisión viene dada por el producto de dos factores: los bits por muestra (N) y las muestras por segundo ($samp_rate$)²⁶. En nuestra implementación:

$$N \text{ (bit/sample)} = \frac{\text{bit}}{\text{symbol}} \cdot \frac{\text{symbol}}{\text{sample}} \Big|_{\text{mod GSMK}} = 1 \cdot \frac{1}{4} = \frac{1}{4}$$

$$v_{TX}(\text{bps}) = N \left(\frac{\text{bit}}{\text{sample}} \right) \cdot samp_rate \left(\frac{\text{sample}}{\text{second}} \right) = \frac{1}{4} \cdot (1 \cdot 10^6) = 250 \text{ kbps}$$

Obsérvese que la velocidad de transmisión del sistema está limitada por la capacidad de transmisión del estándar *Gigabit Ethernet* (1 Gbps). De hecho, la velocidad máxima (hipotética) a la que lograría trabajar el dispositivo es:

$$v_{USRPMAX}(\text{bps}) = \left(16 \left(\frac{\text{bit}}{\text{I sample}} \right) + 16 \left(\frac{\text{bit}}{\text{Q sample}} \right) \right) \cdot 25 \cdot 10^6 \left(\frac{\text{sample}}{\text{second}} \right) = 800 \text{ Mbps}$$

$$v_{USRPMAX}(\text{bps}) = \left(8 \left(\frac{\text{bit}}{\text{I sample}} \right) + 8 \left(\frac{\text{bit}}{\text{Q sample}} \right) \right) \cdot 50 \cdot 10^6 \left(\frac{\text{sample}}{\text{second}} \right) = 800 \text{ Mbps}$$

Por tanto, podemos concluir que:

$$v_{TX}(\text{bps}) \ll v_{USRPMAX}(\text{bps}) < v_{GigaE}(\text{bps})$$

²⁶ Obsérvese que en el modulador se ha elegido trabajar a una velocidad de muestreo de 1 Msample/s ($samp_rate = 1 \cdot 10^6$ sample/s), es decir, a una velocidad inferior a la máxima que permite el USRP (25 Msample/s).

Lo que quiere decir que la velocidad de transmisión de la señal está dentro de los márgenes establecidos y el transmisor, en este aspecto, no presentará problemas.

Por otra parte, sabemos que el ancho de banda de la modulación GMSK es aproximadamente 1.5 veces la velocidad de transmisión, así pues se obtiene que:

$$B_{T_X}|_{GMSK} \approx 1.5 \cdot v_{T_X} = 1.5 \cdot (250 \cdot 10^3) = 375 \text{ kHz}$$

Este valor está muy cerca del valor del ancho de banda de transmisión que se aprecia, a ojo, en el espectro de la figura 5.3.1-11.

Finalmente, para cerrar este apartado, debemos comentar las características de los bloques que se han utilizado para visualizar gráficamente la señal transmitida. Por una parte hemos utilizado el bloque *WX GUI FFT Sink* que nos permitía ver el espectro de la señal modulada en GMSK y, por otra parte, hemos utilizado el bloque *WX GUI Waterfall Sink* que nos mostraba el espectrograma de la señal transmitida. En la figura 5.3.1-10 y en la figura 5.3.1-12 se pueden ver los valores elegidos para configurar los parámetros de estos bloques. Cabe destacar que dichos parámetros están asociados a la visualización de la señal y no afectan en nada a la cadena del transmisor. Por tanto, se han escogido aquellos valores que permitían una mejor representación.

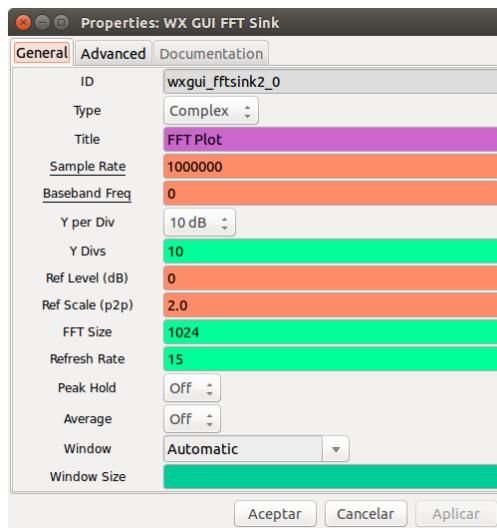


Figura 5.3.1-10 Parámetros del bloque *WX GUI FFT Sink*.

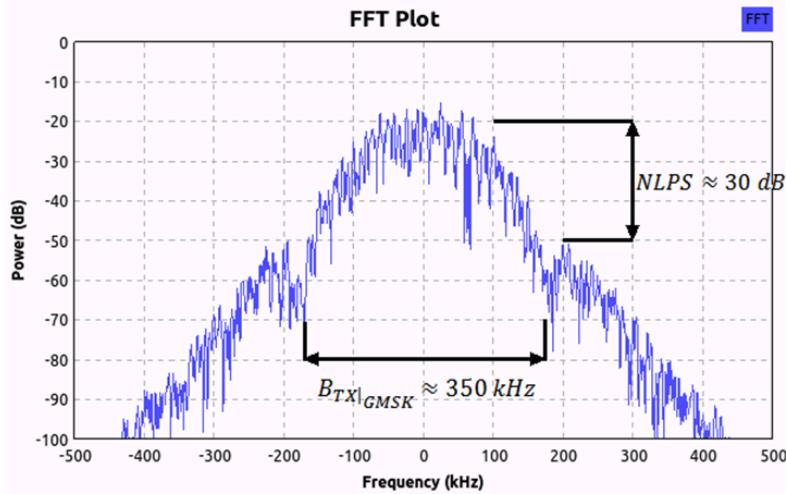


Figura 5.3.1-11 Densidad espectral de potencia de la señal GMSK.

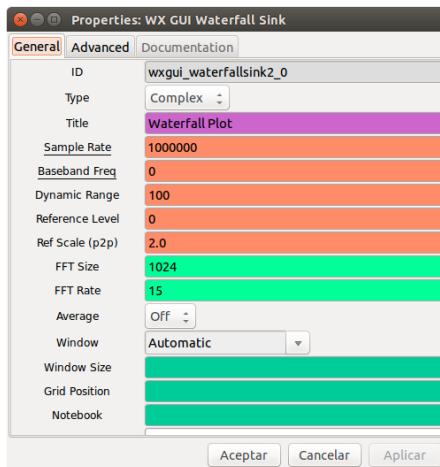


Figura 5.3.1-12 Parámetros del bloque *WX GUI Waterfall Sink*.

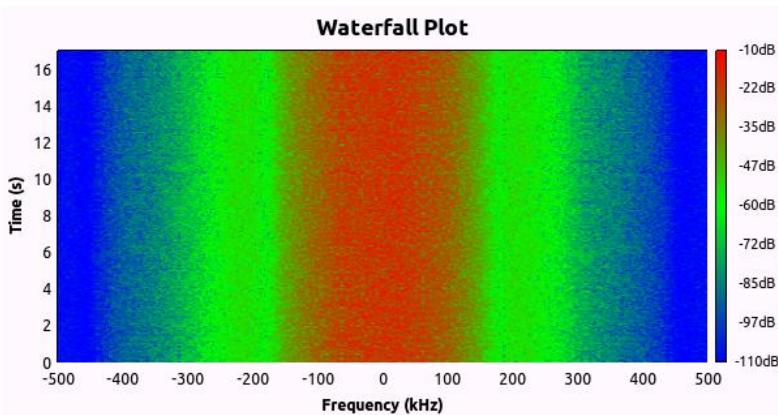


Figura 5.3.1-13 Espectrograma de la señal GMSK transmitida.

5.3.2 Receptor

El esquema diseñado para el receptor es el siguiente:

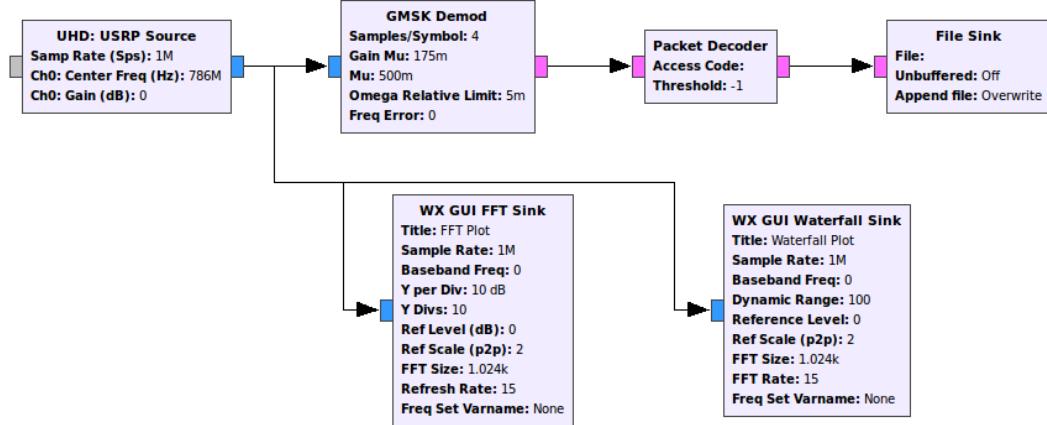


Figura 5.3.2-1 *Flow graph* de la cadena de transmisión.

El esquema es muy similar al del transmisor, pero esta vez se utilizan los bloques antagonistas, cuyas funciones principales son: recibir la señal, demodular, deshacer el entramado y almacenar los bits de información en un archivo. Por tanto, en este diagrama de flujo (*flow graph*) nos encontramos una fuente (*USRP Source*), dos bloques de procesado de señal (*GMSK Demod* y *Packet Decoder*) y tres sumideros (*File Sink* y otra vez los dos visualizadores gráficos: *FFT Sink*, *Waterfall Sink*).

El bloque *UHD: USRP Source* se encarga de aplicar un filtro pasa-banda, bajar la señal a banda-base y muestrear. Los parámetros a configurar son los mismos que en el bloque *UHD:USRP Sink* del transmisor.

Los bloques *WX GUI FFT Sink* y *WX GUI Waterfall Sink* nos muestran la señal recibida. Como se aprecia en la figura 5.3.2-3, la señal se ha visto atenuada debido a su paso por el interfaz aire (espectro radioeléctrico). Sin embargo, el nivel de lóbulo principal a secundario (NLPS) del espectro de la señal recibida sigue manteniéndose, como también se mantiene el ancho de banda de transmisión, que corresponde al lóbulo principal de la señal y donde se concentra la mayor parte de la energía.

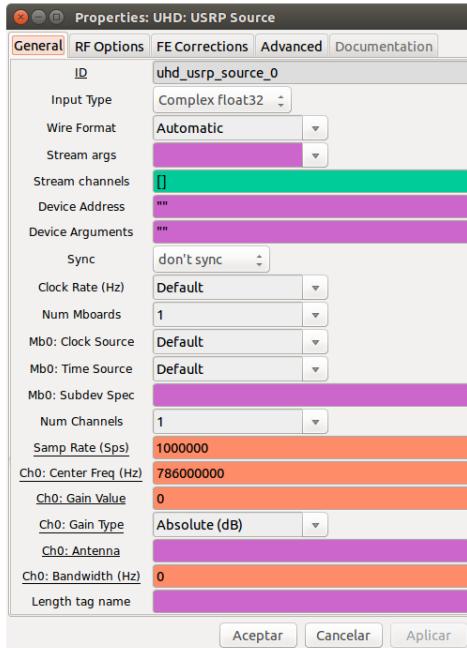


Figura 5.3.2-2 Parámetros del bloque *UHD:USRP Source*.

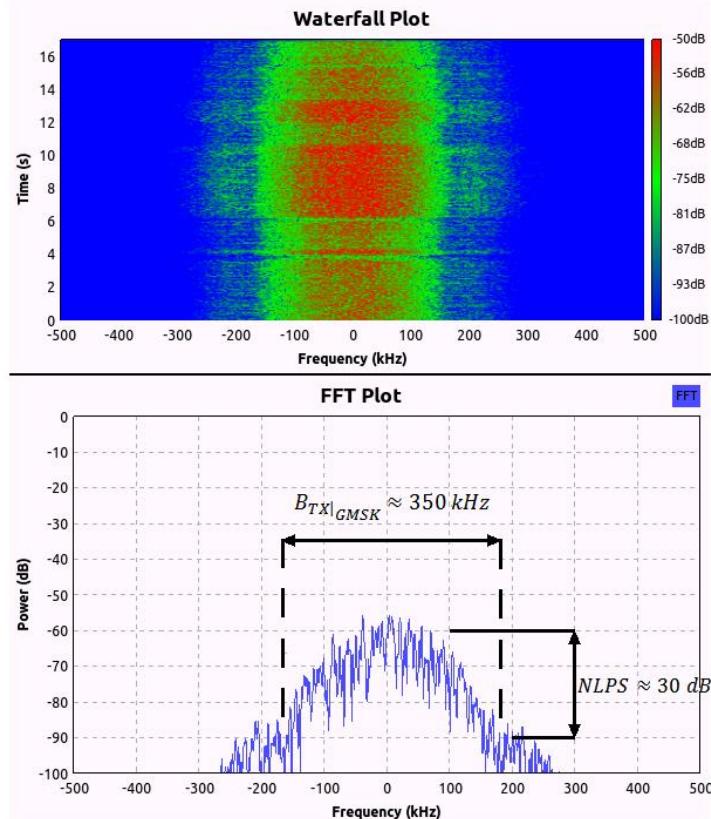


Figura 5.3.2-3 Arriba, el espectrograma de la señal recibida. Abajo la densidad espectral de potencia de la señal recibida.

Así pues, la señal recibida pasa ahora por el bloque *GMSK Demod*, que es el bloque que lleva a cabo el proceso de demodulación de la señal GMSK. De esta manera, lo que se introduce en el puerto de entrada es la señal modulada compleja en banda base y lo que se extrae por el puerto de salida es un flujo de bits, donde cada bit es empaquetado en un byte. A modo de ejemplo, si se quiere sacar el byte 00100011, lo que saldrá realmente son los siguientes ocho bytes: 00000000 00000000 00000001 00000000 00000000 00000000 00000001 00000001. De los parámetros a configurar de este bloque, destacan los que corresponden a mejorar el *digital clock recovery* del receptor. Los más importantes son el parámetro “*Gain Mu*”, que es un factor de corrección basado en el desfase temporal entre símbolos, y el parámetro “*Mu*”, que sirve para ajustar el *sampling* (tiempos de muestreo). Para configurar estos y la resta de parámetros de este bloque nos hemos ayudado de las indicaciones que se dan en http://wiki.spench.net/wiki/GNU_Radio_Notes. Los valores elegidos aparecen en la siguiente figura:

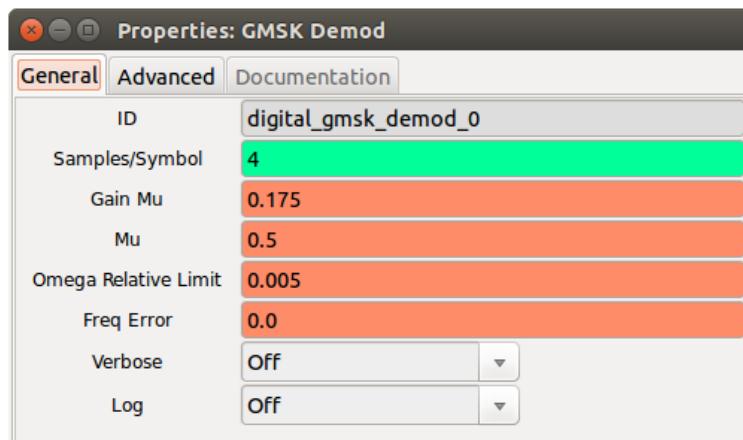


Figura 5.3.2-4 Parámetros del bloque *GMSK Demod*.

El siguiente bloque con el que se encuentra el flujo de bits que ha sido demodulado es el *Packet Decoder*. Recordemos que el *Packet Encoder* del transmisor agregaba los bits de información en tramas. Estas tramas tenían, a parte del *payload*, otros bits que conformaban campos como, por ejemplo, el preámbulo, el código de acceso, etc. (véase la figura 5.3.1-4). Entonces, la función del *Packet Decoder* es analizar el código de acceso de las tramas que le van llegando y decidir. Si el número de bits erróneos de este campo es menor al umbral (*Threshold*), lee el campo de la cabecera, obtiene la longitud del *payload*, extrae el *payload* y envía dichos datos al puerto de salida del bloque. Por el contrario, si el

número de bits erróneos es mayor que el umbral prestablecido, se descarta la trama y se pierden los datos.

Tal y como se dijo el apartado 5.3.1, si el parámetro “Access Code” del bloque *Packet Encoder* queda en blanco, por defecto se asigna el valor 0xACDDA4E2F28C20FC como código de acceso de 64 bits. En este caso, si la trama que le llega al *Packet Decoder* tiene 12 bits erróneos o más en el código de acceso, la trama es descartada.

Por tanto, los parámetros a configurar en el bloque *Packet Decoder* son “Access Code” y “Threshold”. En el “Access Code” hay que poner la misma secuencia de bits que la utilizada en el *Packet Encoder* y en el “Threshold” hay que poner el número de bits erroneos que se desea como umbral de decisión. Para que se utilicen los valores por defecto, el primer parámetro se deja en blanco y al segundo se le da el valor “-1”.

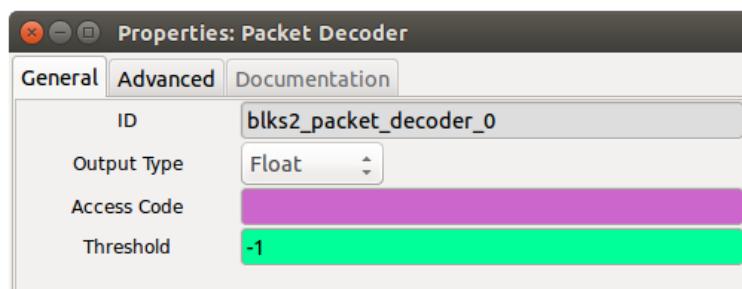


Figura 5.3.2-5 Parámetros del bloque *Packet Decoder*.

Finalmente, los bits pertenecientes al *payload* de las tramas irán llegando al bloque *File Sink* que creará un archivo para almacenar la información recibida. Una vez se acabe la recepción, el archivo podrá ser leído.

En cuanto a los parámetros a configurar de este bloque, en el parámetro “File” hay que indicar la ruta en el ordenador donde se quiere almacenar el archivo que contendrá los datos recibidos y el nombre que tendrá dicho archivo. El parámetro “Unbuffered” nos da a elegir entre dos opciones: “Off” y “On”. Si elegimos “Off” los datos que vayan entrando al bloque serán almacenados en un buffer en memoria y, una vez estén todos, serán volcados al fichero. Si elegimos “On” los datos serán almacenados en un buffer en memoria y se volcarán al fichero cada vez

que se llame a la función “*work*”²⁷ del bloque. Esta última opción no es muy recomendable puesto que puede ocasionar que el programa se ejecute más lentamente ya que requiere acceder al disco duro cada vez que se aplica. Por último, el parámetro “*Append file*” configurado con la opción “*Append*” permite que los datos que se quieren almacenar sean anexados al fichero, permitiendo así no sobreescribir el contenido inicial que pudiera tener; configurado con la opción “*Overwrite*” los datos sobreescriben cualquier contenido del fichero.

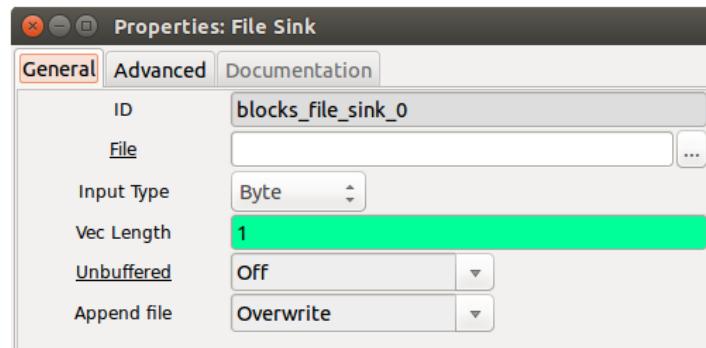


Figura 5.3.2-6 Parámetros del bloque *File Sink*.

5.3.3 Cabeceras en la transmisión

Uno de los problemas que nos encontramos al poner en marcha los diseños implementados fue que, en el transcurso de la transmisión, se perdía una parte de la información contenida en el archivo que se enviaba. Y esto sucedía siempre, tuviera el archivo el tamaño que tuviera, llegando incluso a no recibir nada si el archivo tenía un tamaño pequeño (~5kb). Un ejemplo de lo que sucedía se muestra en la figura 5.3.3-1.

Observamos, además, que la información que perdíamos era siempre la del contenido final del archivo y que esta pérdida era variable (a veces se perdían más caracteres, a veces menos). Entonces, se llegó a la conclusión de que la causa de este problema se debía a un proceso de sincronización que necesitan los USRPs

²⁷*Work function*: Es una rutina implementada en C++ que tienen todos los bloques en *GNU Radio* y que, en esencia, lo que hace es leer los *inputs* y escribir los *outputs* de dichos bloques. Para más información:

<https://gnuradio.org/redmine/projects/gnuradio/wiki/BlocksCodingGuide#The-work-function>

para empezar a transmitir y recibir entre ellos. La explicación de por qué siempre se perdía los últimos caracteres del archivo es sencilla: el transmisor almacena los datos que se quieren enviar por radio (*over the air*) en un buffer, de manera que los primeros datos que salen del transmisor son los últimos que llegan al buffer. Por consiguiente, si al comienzo el USRP transmisor y el USRP receptor necesitan un tiempo aleatorio indeterminado para sincronizarse entre ellos, los primeros bytes que se pierden son los del final del archivo.

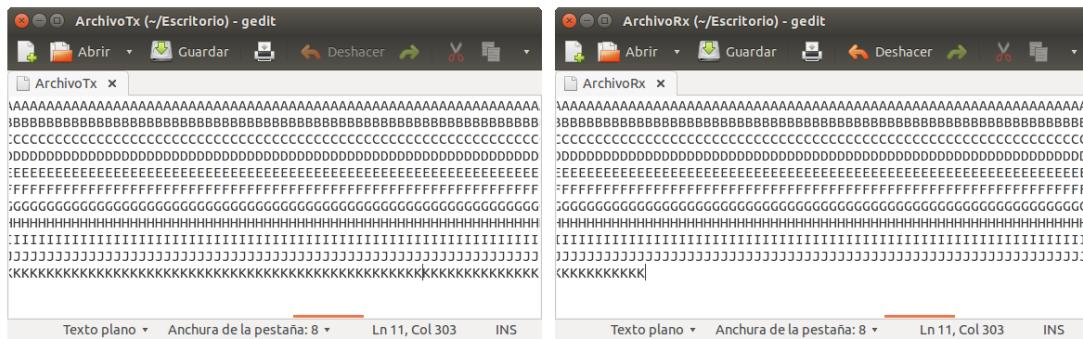


Figura 5.3.3-1 A la izquierda, contenido del archivo transmitido. A la derecha, contenido del archivo recibido.

Para dar solución a este reto, se propuso hacer un tratamiento de pre-procesado al archivo que se iba enviar que consistía en añadir una cabecera inicial y una cabecera final de 40 bits cada una. Para tratar al archivo se utilizó un editor de código hexadecimal de manera que se introdujo los valores 40494E492A (caracteres `@INI$` en ASCII) como cabecera inicial y los valores 2646494E24 (caracteres `&FIN$` en ASCII) como cabecera final. Una vez el archivo era modificado, se enviaba mediante el transmisor con la opción “*Repeat*” del bloque *File Source* activada, para poder así retransmitir repetidamente la información. Finalmente el receptor obtenía la información y al archivo recibido se le hacía un tratamiento de post-procesado que consistía en buscar la cabecera inicial y la cabecera final, copiar la información entre ellas (que es la información útil) y volcarla en el archivo final. De esta manera, creando los *scripts* de pre-procesado y post-procesado, se consiguió solventar el problema de la sincronización entre USRPs y la consecuente pérdida de información.

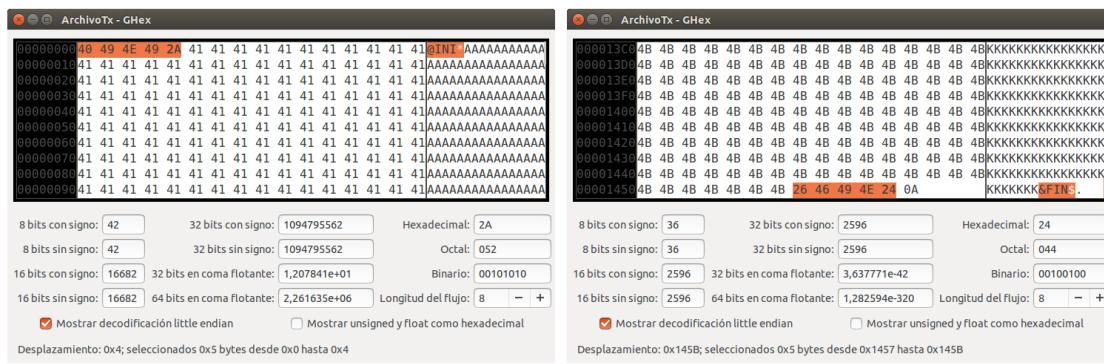


Figura 5.3.3-2 Añadiendo las cabeceras @INI*y &FIN\$ con el programa GHex.

5.4 Esquema conceptual de los scripts

Por último, se presenta en la figura 5.4-1 el esquema conceptual del protocolo de comunicación que lleva a cabo el demostrador de Radio Cognitiva implementado.

Como se observa, se ha utilizado un método de *Stop and Wait* para controlar los errores que puedan producirse en la comunicación entre los dos hosts. De esta manera, cada vez que se envía una petición o un archivo, el proceso se detiene hasta confirmar la recepción con el correspondiente ACK. Si pasado un tiempo determinado (*Time Out*) no se recibe el ACK, el proceso, dependiendo del caso, o puede volver a repetir el envío de la petición o del archivo, o bien puede seguir esperando.

Para implementar el envío de las peticiones y de los ACKs que aparecen en la fase de establecimiento de la comunicación y para implementar la comunicación final, que es una comunicación *simplex* (el único sentido de la comunicación que hay es del usuario secundario al receptor de la Estación Base), se han creado unos programas en Python, utilizando los fundamentos del código *benchmark* que se puede encontrar en la carpeta gr-digital de *GNU Radio*.

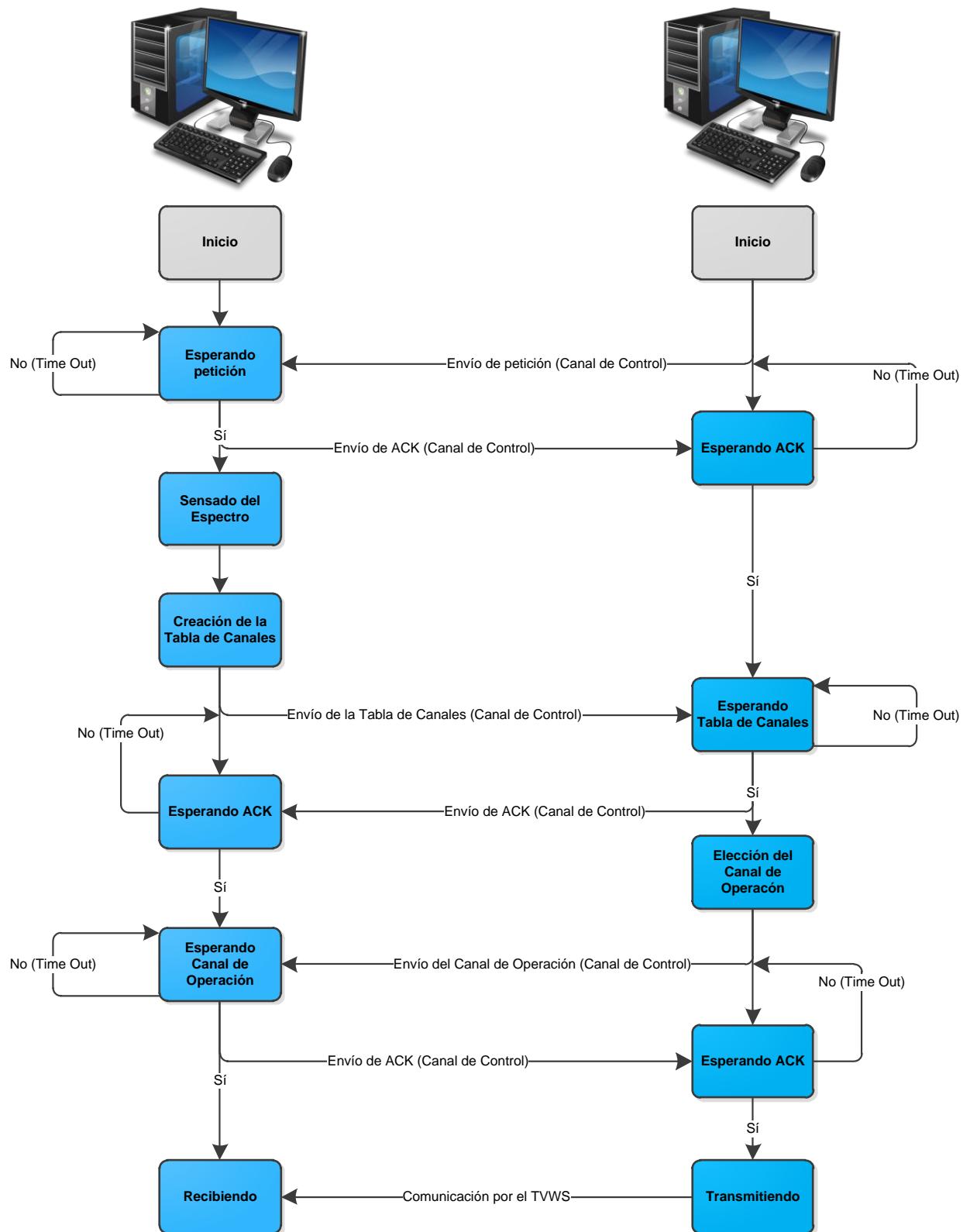


Figura 5.4-1 Esquema conceptual del sistema de Radio Cognitiva.

En la comunicación *simplex* del final, el usuario secundario lo que hace es transmitir, por el TVWS elegido, 125 paquetes de 12 bytes cada uno, esperando 2 segundos cada vez que ha transmitido 5 paquetes. En total se transmiten 1.5 kB. Los paquetes son generados por el propio programa y para transmitirlos se encapsulan en la misma estructura de trama que la de la figura 5.3.1-4 y se utiliza la modulación GMSK.

El receptor, por su parte, escucha la llegada de los paquetes, va mostrando por pantalla un resumen de cada uno de ellos y comprueba si tienen errores. Como se observa en la figura 5.4-2, el término “ok” que aparece puede tomar el valor de “True” o el valor de “False”. “True” indica que el CRC ha verificado los datos y no hay errores, mientras que “False” indica se han encontrado uno o más errores en dichos datos; “*pktno*” da el número del paquete recibido (del 0 al 124), “*n_rcvd*” cuenta los paquetes recibidos hasta el momento y “*n_right*” hace lo mismo que este último pero solo lleva la cuenta de los recibidos correctamente.

```

ok = True  pktno = 112  n_rcvd = 113  n_right = 113
ok = True  pktno = 113  n_rcvd = 114  n_right = 114
ok = True  pktno = 114  n_rcvd = 115  n_right = 115
ok = True  pktno = 115  n_rcvd = 116  n_right = 116
ok = True  pktno = 116  n_rcvd = 117  n_right = 117
ok = True  pktno = 117  n_rcvd = 118  n_right = 118
ok = True  pktno = 118  n_rcvd = 119  n_right = 119
ok = True  pktno = 119  n_rcvd = 120  n_right = 120
ok = True  pktno = 120  n_rcvd = 121  n_right = 121
ok = True  pktno = 121  n_rcvd = 122  n_right = 122
ok = True  pktno = 122  n_rcvd = 123  n_right = 123
ok = True  pktno = 123  n_rcvd = 124  n_right = 124
ok = True  pktno = 124  n_rcvd = 125  n_right = 125

```

Comunicación finalizada. TV White Space desocupado

Figura 5.4-2 El receptor muestra por pantalla un resumen de los paquetes que va recibiendo.

6. Conclusiones y trabajo futuro

Para llevar a cabo el demostrador de Radio Cognitiva, en primer lugar hemos profundizado en nuestros conocimientos sobre la Radio Cognitiva. En este aspecto, hemos visto que una **Radio Cognitiva** es un dispositivo inalámbrico programable capaz de percibir el entorno que le rodea y aprender de él, con el objetivo de adaptar sus parámetros y optimizar lo máximo posible la operatividad de sus funcionalidades. Las singulares propiedades que caracterizan a los terminales de radio cognitiva hacen de ellos un dispositivo óptimo para el empleo de **técnicas de acceso dinámico al espectro**, siendo esta una solución eficaz para el vigente problema de **escasez espectral** en el que nos encontramos. En efecto, mediante el denominado **Ciclo Cognitivo**, que representa los estados de las diferentes etapas del proceso cognitivo, los llamados **usuarios secundarios** (usuarios sin licencia para explotar el espectro) son capaces de aprovechar de manera oportunista una porción del espectro desocupada (**spectrum hole**) durante un tiempo limitado y sin interferir de ningún modo a los **usuarios primarios** (usuarios con autorización). Hemos visto que este proceso consta de cuatro fases: **Spectrum Awareness** y **Spectrum Decision**, **Spectrum Sharing** o **Spectrum Mobility** y que la banda de TV ofrece unos huecos espetrales muy convenientes para desarrollar esta tecnología: los **TV White Spaces**.

A partir de las bases teóricas de la Radio Cognitiva y de la caracterización de las propiedades de los *TV White Spaces*, en segundo lugar, hemos configurado un sistema **Software Defined Radio** (SDR), compuesto por 2 USRPs, 2 PCs y 4 antenas dipolo, capaz de implementar y demostrar algunas de las principales funcionalidades de la Radio Cognitiva. Los sistemas SDR permiten la reconfiguración de la mayoría de sus parámetros y funcionalidades radio gracias a que todo el procesado de la señal en banda base se realiza por **software**, dejando únicamente las conversiones de digital a analógico (y viceversa) y las funciones propias del *Front-End* de radiofrecuencia al equipo **hardware**. En nuestro caso, el componente *hardware* principal ha sido el **USRP N200**, que se compone de una placa madre (**motherboard**) donde se localiza la **FPGA** y los conversores, y de una placa hija (**daughterboard**) que, en esencia, es el cabezal de radiofrecuencia. Por otra parte, para el desarrollo *software* de la radio cognitiva, se ha utilizado el

programa ***GNU Radio***, que dispone de una gran librería de funciones de procesado de señal y, además, facilita al usuario el proceso de programación ofreciendo una interfaz gráfica sencilla e intuitiva.

Posteriormente, se ha diseñado, implementado y automatizado la fase de *Spectrum Awareness* creando en Python el programa *Spectrum_Awareness.py*, en el cual se configura un **detector de energía** que permite analizar qué canales de la **banda de TV** están siendo utilizados y qué canales pueden ser útiles para el acceso oportunista por parte del SU. A través de un **criterio de decisión**, los 40 canales de la banda de TV son designados como “Libre” “Ocupado”, “Dudoso Libre” y “Dudoso Ocupado”, aunque, en última instancia, el usuario cognitivo sólo puede elegir entre aquellos canales etiquetados como “Libres” en la fase de *Spectrum Selection*. En esta fase no se ha automatizado la selección del canal ya que, al ser la plataforma desarrollada un demostrador de laboratorio, se ha buscado potenciar los aspectos demostrativos haciendo que la evaluación de las características de los canales libres y la elección del radiocanal sean ambos manuales.

Puesto que sólo se ha implementado una **Estación Base** y un usuario del sistema Cognitivo, los aspectos relacionados con la funcionalidad de *Spectrum Sharing* se limitan a la implementación de sendos transceptores con parámetros de operación programables. En particular, mediante el entorno ***GNU Radio Companion*** hemos diseñado los esquemas del **transmisor** y el **receptor** para hacer posible el envío de archivos entre los dos USRPs y garantizar que los dispositivos puedan **reconfigurarse**.

Finalmente, en el contexto de construir un demostrador con finalidades didácticas, se han creado dos **scripts**, uno para cada PC. Dichos *scripts* permiten que, en primer lugar, haya una fase de establecimiento de la comunicación, en la que el usuario cognitivo manda una petición de sensado a la Estación Base, recibe la *tabla de canales*, elige el canal de operación y envía dicha información a su destinatario, todo a través de un **canal de control** preestablecido; y, en segundo lugar, se lleve a cabo una comunicación *simplex* entre SU y su receptor a través el TVWS, en donde se envía 1.5 kB de información generada por el propio programa.

En resumen, y a partir de las actividades desarrolladas, el Proyecto Fin de Carrera me ha permitido:

- Concebir, diseñar, implementar y evaluar un sistema de comunicaciones inalámbricas cognitivo utilizando tecnología *Software Radio* basada en dispositivos USRP N200 conectados a un PC.
- Ampliar mis conocimientos sobre el tema de la Radio Cognitiva.
- Aprender a programar en Python.
- Aprender a utilizar el programa *GNU Radio*.
- Asimilar y utilizar conceptos adquiridos durante la carrera como, por ejemplo, los dados en las asignaturas de *Sistemas y Señales 2* (SIS2) *Comunicaciones 2* (Com2) y *Radiocomunicaciones* (Radiocom).

Evidentemente, a partir de esta estructura básica del demostrador, se pueden hacer y mejorar muchas cosas. En este sentido, y como líneas de trabajo futuro, se propone:

- Añadir al sistema más usuarios secundarios (USRPs) e implementar las funcionalidades específicas de la etapa de *Spectrum Sharing*.
- Añadir al sistema usuarios primarios (USRPs) e implementar la etapa de *Spectrum Mobility*.
- Mejorar la etapa de *Spectrum Awareness*, aumentando las funcionalidades del detector de energía o implementando cualquier otra técnica (detector cicloestacionario, detector de filtro adaptado, etc.).
- Mejorar la etapa de *Spectrum Selection*, añadiendo un proceso automático de análisis, para decidir la mejor opción posible, en la elección del radiocanal.
- Incrementar la robustez de los esquemas diseñados en transmisión y recepción, añadiendo nuevos bloques que proporcionen alineación en frecuencia, corrección de errores, sincronización y ecualización, etc.
- Modificar el escenario en el que se trabaja, añadiendo obstáculos de manera que se dificulte la propagación con línea de vista (LOS) o, incluso, configurando un entorno NLOS.
- Transmitir y recibir otro tipo de archivos, como por ejemplo vídeos, imágenes, pdfs, etc.

Apéndices

A. Instalación y configuración del USRP N200 en Ubuntu

Instalación del UHD

Para poder utilizar el USRP N200, se ha de instalar el *USRP Hard Driver* (UHD). La instalación del UHD se puede llevar a cabo de 3 maneras diferentes:

1. Mediante el **gestor de paquetes** que proporcione la distribución de Ubuntu (normalmente, la mayoría de distribuciones proporcionan el *UHD* como parte del sistema de gestión de paquetes).

Pasos a realizar:

- 1) Abrir un Terminal de Linux usando la combinación de teclas CTRL+ALT+T
- 2) Ejecutar el siguiente comando:

```
sudo apt-get install libuhd-dev libuhd003 uhd-host
```

2. Mediante los **instaladores binarios** que proporciona *Ettus Research*.

Pasos a realizar:

- 1) Abrir un Terminal de Linux.
- 2) Ejecutar los siguientes comandos:

```
sudo bash -c 'echo "deb  
http://files.ettus.com/binaries/uhd/repo/uhd/ubuntu/`lsb_release -  
cs` `lsb_release -cs` main" >/etc/apt/sources.list.d/ettus.list'
```

```
sudo apt-get update
```

```
sudo apt-get install -t `lsb_release -cs` uhd
```

3. Mediante **PyBOMBS** (RECOMENDADO). En el Anexo B se explica cómo descargar y configurar PyBOMBS.

Pasos a realizar:

- 1) Abrir un Terminal de Linux.
- 2) Ejecutar los siguientes comandos:

cd pybombs

sudo ./pybombs install uhd

Una vez se ha instalado el *driver* que permite manejar el equipo *hardware*, se pasa a explicar la configuración del USRP que se ha llevado a cabo en el laboratorio.

Configuración del USRP N200

1 –Conectar entre sí el USRP y el ordenador mediante el cable *Gigabit Ethernet* (en nuestro caso, USRP y host están comunicados a través de un *switch*). Posteriormente, conectar el USRP a la fuente de alimentación.

2 – El USRP N200 se comunica en la capa IP/UDP sobre *Gigabit Ethernet*. Por defecto, la IP del USRP es 192.168.10.2. Por tanto, para comunicarse con él, se ha de configurar la interfaz *Ethernet* del host en el que se opera, utilizando una dirección IP estática del tipo 192.168.10.X. Se recomienda la dirección 192.168.10.1 y una máscara de red 255.255.255.0.

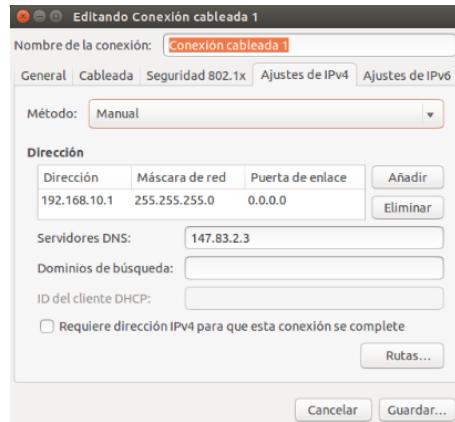


Figura A-1 Configuración de la red *Ethernet*.

3 – El comando ***uhd_find_devices*** busca los dispositivos USRP conectados al ordenador y muestra algo de información sobre él, como el tipo, el nombre, el número de serie y la dirección.

```
usuario@rcm@usrp2: ~
usuario@rcm@usrp2:~$ uhd_find_devices
linux; GNU C++ version 4.8.4; Boost_105400; UHD_003.009.git-219-gd9656de8

-- UHD Device 0
-----
Device Address:
    type: usrp2
    addr: 192.168.10.2
    name:
    serial: F50263
```

Figura A- 2 Ejemplo de información proporcionada por el comando *uhd_find_devices*.

4 – El comando ***uhd_usrp_probe*** muestra información adicional sobre el USRP conectado (conversores, placa hija, etc.).

```
/ RX DSP: 0
| Freq range: -50.000 to 50.000 MHz
|
/ RX DSP: 1
| Freq range: -50.000 to 50.000 MHz
|
/ RX Dboard: A
| ID: WBR v3, WBR v3 + Simple GDB (0x0057)
| Serial: F46631
|
/ RX Frontend: 0
| Name: WBR v3 RXv2D
| Antennas: TX/RX, RX2, CAL
| Sensors: lo_locked
| Freq range: 68.750 to 2200.000 MHz
| Gain range PGA: 0.0 to 31.5 step 0.5 dB
| Bandwidth range: 4000000.0 to 40000000.0 step 0.0 Hz
| Connection Type: IQ
| Uses LO offset: No
|
/ RX Codec: A
| Name: adso2p44
| Gain range digital: 0.0 to 6.0 step 0.5 dB
| Gain range fine: 0.0 to 0.5 step 0.1 dB
```

Figura A-3 Ejemplo de información proporcionada por el comando *uhd_usrp_probe*.

5 – Los LEDs que se observan en el panel frontal del USRP revelan lo siguiente sobre el estado del dispositivo:

- LED A: Se está transmitiendo.
- LED B: Se está utilizando MIMO.
- LED C: Se está recibiendo.
- LED D: Se ha cargado el firmware de la FPGA.
- LED E: El oscilador interno está en estado *lock* de la referencia.
- LED F: Se ha cargado el CPLD (*Complex Programmable Logic Device*)



Figura A-4 A la izquierda, ejemplo del estado de los LEDs cuando el USRP transmite. A la derecha, ejemplo del estado de los LEDs cuando el USRP recibe.

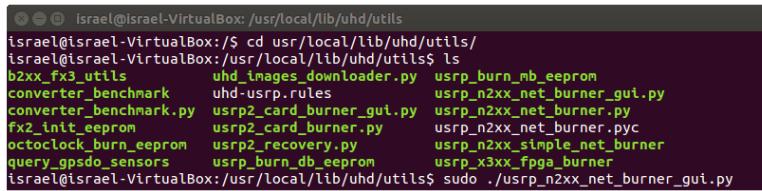
6 – Es necesario cargar el *firmware* y las imágenes de la FPGA en el USRP. La serie N del USRP puede ser reprogramada por la red para actualizar o cambiar el *firmware* y las imágenes de la FPGA. Cuando se actualizan las imágenes, se ha de cargar tanto las imágenes de la FPGA como del *firmware* antes de reiniciar el dispositivo, sólo así el dispositivo se asegura de tener las imágenes compatibles una vez se inicie de nuevo.

En primer lugar, ejecutar los siguientes comandos:

```
cd <uhd_installation_path>/uhd/utils
```

```
sudo ./usrp_n2xx_net_burner_gui.py
```

“*uhd_installation_path*” es la ruta donde está instalado el UHD y *usrp_n2xx_net_burner_gui.py* es el programa que cargará las imágenes de la FPGA y del *firmware* en el USRP.



```

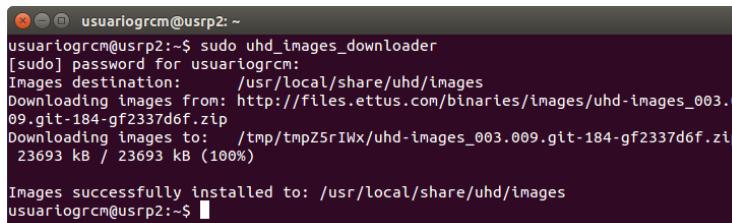
israel@israel-VirtualBox:/usr/local/lib/uhd/utils$ cd /usr/local/lib/uhd/utils/
israel@israel-VirtualBox:/usr/local/lib/uhd/utils$ ls
b2xx_fx3_utils      uhd_images_downloader.py  usrp_burn_mb_eeprom
converter_benchmark  uhd_usrp.rules          usrp_n2xx_net_burner_gui.py
converter_benchmark.py usrp2_card_burner_gui.py  usrp_n2xx_net_burner.py
fx2_init_eeprom       usrp2_card_burner.py    usrp_n2xx_net_burner.pyc
octoclock_burn_seeprom usrp2_recovery.py   usrp_n2xx_simple_net_burner
query_gpsdo_sensors   usrp_burn_db_eeprom    usrp_x3xx_fpga_burner
israel@israel-VirtualBox:/usr/local/lib/uhd/utils$ sudo ./usrp_n2xx_net_burner_gui.py

```

Figura A-5 Comandos utilizados para cargar las imágenes en el USRP.

En segundo lugar, descargar las imágenes del *firmware* y de la *FPGA*, ejecutando el siguiente comando en otro terminal:

sudo uhd_images_downloader



```

usuariogrcm@usrp2:~$ sudo uhd_images_downloader
[sudo] password for usuariogrcm:
Images destination: /usr/local/share/uhd/images
Downloading images from: http://files.ettus.com/binaries/images/uhd-images_003.09.git-184-gf2337d6f.zip
Downloading images to: /tmp/tmpZ5rIWx/uhd-images_003.009.git-184-gf2337d6f.zip
23693 kB / 23693 kB (100%)
Images successfully installed to: /usr/local/share/uhd/images
usuariogrcm@usrp2:~$ 

```

Figura A-6 Ejemplo del comando *sudo uhd_images_downloader*.

Teniendo el *USRP-N2XX Net Burner* abierto, seleccionar las imágenes del *firmware* y de la *FPGA* descargadas. Si el dispositivo USRP instalado no aparece en la ventana, ni tampoco haciendo clic en la opción “*Rescan for Devices*”, simplemente escribir su dirección IP manualmente en el apartado “*Network Address*” (por defecto es 192.168.10.2).

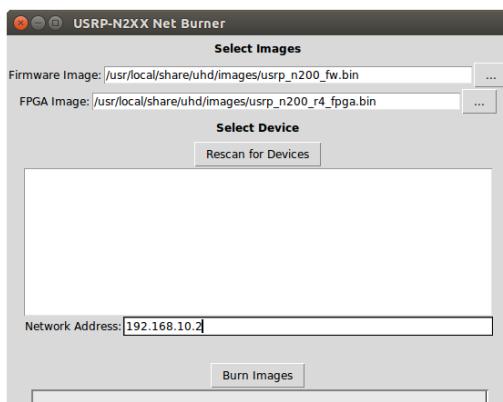


Figura A-7 Interfaz del programa *usrp_n2xx_net_burner_gui.py*.

Finalmente, hacer clic en “*Burn Images*” para cargar las imágenes en el USRP. Esto puede tomar un par de minutos. Posteriormente el programa debería

preguntar si se desea reiniciar el dispositivo. No hay que reiniciarlo manualmente, simplemente al presionar ‘ok’ se realizará. No cerrar el *USRP-N2XX Net Burner* antes de reiniciar el dispositivo, de lo contrario el dispositivo podría ponerse en estado inutilizable por cargar de mala manera las imágenes.

Si ocurriera esto, afortunadamente los USRPs de la serie N pueden ser iniciados en modo seguro (*read-only*). Si se inicia de esta manera, el usuario puede volver a cargar las imágenes en el dispositivo. El botón del modo seguro es un interruptor en forma de pulsador (S2) ubicado en el interior de la carcasa. Para arrancar en este modo, mantener pulsado el interruptor mientras apaga y enciende el dispositivo. Mantener presionado el interruptor hasta que los LEDs de panel frontal dejen de parpadear y se mantengan fijos. En el modo seguro, el dispositivo USRP de la serie N siempre tendrá la dirección IP 192.168.10.2 por defecto.

*7– A la hora de crear un sistema de comunicaciones basado en el SDR que emplee mínimo dos USRPs N200, se ha de tener en cuenta que todos los USRPs de esta familia tienen la misma IP asignada por defecto. Por lo tanto, puede haber un conflicto si los dos ordenadores que utilizan dichos USRPs están conectados dentro de una misma red, por ejemplo, a través de un *switch*. Para solucionar este problema específico, se ha de modificar la IP de uno de los dos USRPs y también la IP de uno de los host. Los comandos a utilizar para modificar la IP del dispositivo son:

```
cd <uhd_installation_path>/uhd/utils
```

```
sudo ./usrp_burn_mb_eeprom --args=<optional devide args> --values = “ip-addr=192.168.20.2”
```

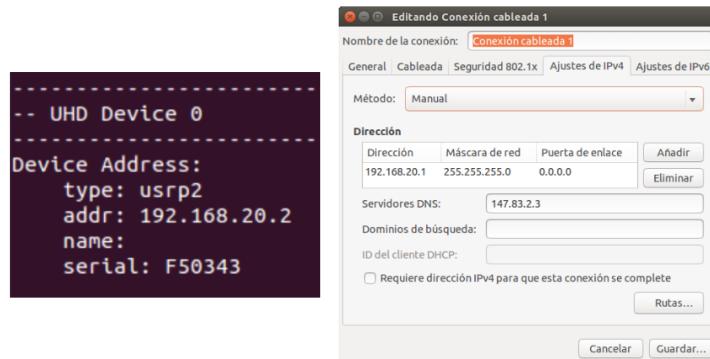


Figura A-8 Configuración de la dirección IP del dispositivo y de la red *Ethernet*, para el segundo USRP.

IMPORTANTE: El equipo no tiene interruptor de encendido y apagado, por lo que no hay que olvidar desenchufarlo cuando se vaya a dejar de usar durante un tiempo prolongado.

Para más información sobre la configuración, se deja a continuación el manual que ofrece el fabricante y la página de soporte técnico:

- files.ettus.com/manual/
- <http://www.ettus.com/support>

B. Instalación de *GNU Radio* en Ubuntu

Hay varias formas de obtener *GNU Radio* en nuestro ordenador. Se destacan las siguientes cuatro:

1. Mediante ***GNU Radio Live SDR Environment***, el cual tiene como ventaja que no requiere ninguna instalación en el ordenador del usuario. Lo único que se ha de hacer es arrancar un sistema operativo desde un USB. En ese sistema operativo, *GNU Radio* viene instalado completamente y está listo para ser utilizado. Además el usuario puede realizar cambios en él de manera persistente y almacenar archivos (todo perdurará después de reiniciar el sistema operativo). La desventaja de esta opción es que su ejecución es más lenta con respecto a la ejecución convencional en el disco duro ya que, en este caso, los tiempos de acceso al medio son mayores.

Pasos a realizar:

- 1) Descargar la imagen ISO de:

<https://gnuradio.org/redmine/projects/gnuradio/wiki/GNURadioLive>
DVD

- 2) Copiar la imagen ISO en un USB. Herramientas para ello pueden ser: *Ubuntu Startup Disk Creator* (Ubuntu Linux OS) o *Unetbootin* (Windows, MacOS, Linux), entre otras.
- 3) Acceder a la BIOS del ordenador. En la mayoría de casos, se consigue presionando F11 durante la fase POST (pocos segundos después de encender el ordenador). En el SETUP del ordenador, especificar como dispositivo de inicio (*boot sequence*) el puerto USB.
- 4) A partir del momento que indique al ordenador que arranque desde USB, siempre que inserte la memoria USB con la imagen ISO y encienda el ordenador, el sistema operativo que le aparecerá será el de la imagen ISO y en él tendrá el programa *GNU Radio* instalado.

2. Mediante el *script* de instalación ***build-gnuradio***²⁸. Este *script* se encarga de todo para hacer funcionar *GNU Radio* en su versión master: bajar las fuentes, chequear las dependencias, compilar las fuentes, asignar las variables de entorno, bajar los drivers del USRP, copiar los archivos a los lugares donde correspondan, etc.

Hay que prestar atención desde dónde se ejecuta porque ahí quedarán las fuentes y el resultado de la compilación. En particular, ahí estará la carpeta *build* que será necesario en caso de querer desinstalar *GNU Radio*.

IMPORTANTE: Antes de proceder a la instalación de *GNU Radio* mediante el *script*, hay que tener ciertas dependencias instaladas en el ordenador. Se puede consultar la lista de dependencias necesarias en la siguiente dirección:

https://gnuradio.org/doc/doxygen/build_guide.html

Pasos a realizar:

- 1) Ir hacia el Tablero (*Dash Home*), escribir “Terminal” en la barra de búsqueda y clicar encima de su ícono (también puede utilizar el atajo de teclas: CTRL+ALT+T). Una vez en el Terminal, escribir el siguiente comando, el cual crea un directorio llamado “gnuradio”:

mkdir gnuradio

- 2) Cambiar el directorio de instalación al directorio que se acaba de crear a través del comando:

cd gnuradio

- 3) Escribir el siguiente comando, que descarga el *script* *build-gnuradio*²⁹, le cambia las propiedades a ejecutable y lo ejecuta:

```
sudo wget http://www.sbrac.org/files/build-gnuradio && chmod a+x  

./build-gnuradio && ./build-gnuradio
```

²⁸ <http://gnuradio.org/redmine/projects/gnuradio/wiki/InstallingGR#Using-the-build-gnuradio-script>.

²⁹ Una vez se ejecute, este *script* tardará, aproximadamente, una hora en finalizar.

Durante la instalación, se hacen dos preguntas: *Proceed?* y *Do you have SUDO privileges?* Simplemente se ha de presionar la tecla “Y” para continuar.

- 4) Una vez finalizada la instalación, para abrir la interfaz gráfica *GNU Radio Companion*, escribir en la ventana del terminal :

gnuradio-companion

- 5) Si se desea desinstalar *GNU Radio*, el proceso es sencillo y toma poco tiempo. Basta con situarse en la carpeta *build* del directorio de instalación y ejecutar el comando:

sudo make uninstall

IMPORTANTE: si se desea utilizar otra versión de *GNU Radio*, se ha de desinstalar la versión anterior.

3. Mediante **PyBOMBS** (*Python Build Overlay Managed Bundle System*)³⁰.
IMPORTANTE: este es el método más aconsejable para instalar *GNU Radio*, ya que ofrece la última versión de *GNU Radio*, resuelve dependencias y hace más sencillo añadir módulos OOT (*Out-Of-Tree Modules*) de proyectos que dan soporte a *GNU Radio* (todo el repositorio desarrollado por empresas *3rd party* para aplicaciones *GNU Radio* se puede encontrar en *The Comprehensive GNU Radio Archive Network (CGRAN)*: <http://www.cgran.org/>).

Pasos a realizar:

- 1) Instalar GIT. Para ello, abrir un terminal y ejecutar el siguiente comando:

sudo apt-get install git-all

- 2) Descargar PyBOMBS, ejecutando el comando:

git clone git://github.com/pybombs/pybombs

- 3) Configurar PyBOMBS. Para ello, ejecutar los comandos:

³⁰ <http://gnuradio.org/redmine/projects/pybombs/wiki>

```
cd pybombs
```

```
./pybombs config
```

A continuación, saldrán los siguientes parámetros a configurar:

- *gituser*
- *gitcache*
- *gitoptions*
- *prefix*
- *satisfy_order*
- *forcepkgs*
- *forcebuild*
- *timeout*
- *cmakebuildtype*
- *builddocs*
- *cc*
- *cxx*
- *makewidth*

Puede dejar la mayoría de parámetros en blanco para que se asigne el valor por defecto, pero ha de configurar el parámetro *prefix* para indicar el directorio donde, posteriormente, serán instalados todos los programas (entre ellos *GNU Radio*). Para que todo sea más sencillo, se recomienda:
prefix: /usr/local

- 4) Instalar *GNU Radio* ejecutando el comando (este proceso tardará un tiempo):

```
sudo ./pybombs install gnuradio
```

- 5) Actualizar los links hacia las recientes librerías creadas:

```
sudo ldconfig
```

A partir de aquí, ya se puede invocar la herramienta gráfica *GNU Radio Companion* (mediante el comando ***gnuradio-companion***).

- 6) Instalar los íconos en los menús de Linux:

```
cd usr/local/libexec/gnuradio/
```

```
sudo grc_setup_freedesktop install
```

- 7) Para instalar módulos OOTs hay dos opciones:

(En el directorio raíz: `~/pybombs`)

- a. Ejecutar el comando: `sudo ./pybombs install nombre_OOT`
 - b. Ejecutar el comando: `sudo ./app_store.py`

Entonces, aparecerá una interfaz gráfica donde se mostrarán todas las aplicaciones que pueden ser instaladas. Hacer clic en la que se desee instalar.



Figura B-1 Interfaz gráfica del programa *app_store.py*.

4. Mediante el comando ***apt-get install gnuradio***.

Pasos a realizar:

- 1) Abrir un terminal
 - 2) Ejecutar el comando ***sudo apt-get install gnuradio***
 - 3) Una vez finalizada la instalación, ejecutar el comando ***gnuradio-companion***.

C. Módulos, carpetas y archivos de *GNU Radio*

Los principales módulos³¹ que conforman *GNU Radio* son:

Módulos <i>GNU Radio</i> (<i>In-Tree components</i>)	
gr-analog	<i>Analog Modulation.</i> Contiene todos los bloques de modulación analógica, utilidades y ejemplos.
gr-audio	<i>Audio Interface.</i> Proporciona los bloques audio_source y audio_sink, que permiten enviar y recibir señales de audio a través de la tarjeta de sonido.
gr-blocks	<i>Standard GNU Radio Blocks.</i> Contiene la mayoría de los bloques simples, estándares o genéricos utilizados en la creación de <i>flowgraphs</i> .
gr-channel	<i>Channel Model Blocks.</i> Contiene los bloques de procesamiento de señal para simular modelos de canal.
gr-digital	<i>Digital Modulation.</i> Contiene todos los bloques de modulación digital, utilidades y ejemplos.
gr-dtv	<i>Digital Tv.</i> Contiene bloques que implementan varios estándares de televisión digital.
gr-fcd	<i>FunCube Dongle Source.</i> Contiene un bloque fuente para el <i>hardware</i> FUNcube Dongle.
gr-fec	<i>Forward Error Correction.</i> Contiene todo los bloques de corrección de errores hacia delante (FEC), utilidades y ejemplos.
gr-fft	<i>FFT Signal Processing Blocks.</i> Contiene bloques de procesamiento de señal para realizar Transformadas Rápidas de Fourier (FFTs) y funciones relacionadas.
gr-filter	<i>Filter Signal Processing Blocks.</i> Contiene bloques de procesamiento de señal para llevar a cabo operaciones de filtrado.
gr-qtgui	<i>QT Graphical User Interface.</i> Contiene varios bloques de interfaz gráfica de usuario Qt que añaden sumideros gráficos al <i>flowgraph</i> .
gr-trellis	<i>Trellis Coding.</i> Contiene bloques para realizar codificaciones convolucionales.
gr-uhd	<i>UHD Interface.</i> Es la interfaz a la librería UHD para poder enviar o recibir datos a través USRP.

³¹ Para más información, consultar:

https://gnuradio.org/doc/doxygen/page_components.html#components_blocks

gr-vocoder	<i>Voice Coders and Decoders (Vocoders).</i> Contiene todos los <i>vocoders</i> disponibles en <i>GNU Radio</i> .
gr-wxgui	<i>WX Graphical User Interface.</i> <i>Contiene varios bloques de interfaz gráfica de usuario Wx que añaden sumideros gráficos al flowgraph.</i>
gr-zeromq	ZeroMQ. Contiene todos los bloques zeromq, utilidades y ejemplos.
Módulos OOT (<i>Out-Of-Tree</i>)	
Todo los del repositorio CGRAN (http://www.cgran.org/)	

Tabla C-1 Clasificación de los módulos de *GNU Radio*.

A su vez, los módulos de *GNU Radio* son estructurados en carpetas:

Carpetas halladas en los módulos de <i>GNU Radio</i>	
grc	Contiene los diferentes archivos *.xml de los bloques para poder ser utilizarlos en la herramienta gráfica <i>GNU Radio Companion</i> .
include	Contiene los archivos fuente (*.h) de las librerías de los bloques de procesado.
lib	Contiene los archivos fuente (*.cc) de los bloques de procesado.
python	Contiene diferentes <i>scripts</i> de Python.
swig	Contiene los archivos swig (*.i) con la configuración del intérprete de C++ y Python.
apps (opcional)	Contiene aplicaciones ya hechas que utilizan bloques del módulo.
cmake (opcional)	Contiene archivos de configuración necesarios para la correcta instalación del módulo.
doc (opcional)	Contiene archivos que proporcionan información sobre el módulo.
examples (opcional)	Contiene ejemplos que muestran las funcionalidades de los bloques del módulo.

Tabla C-2 Listado de carpetas de los módulos de *GNU Radio*

Y los diferentes tipos de archivos encontrados en las respectivas carpetas son clasificados en:

Archivos encontrados en las carpetas de <i>GNU Radio</i>	
Archivos *.cc	Son los que contienen la función que desempeña el bloque de procesado de señal. Están escritos en lenguaje de programación C++.
Archivos *.dox	Son los que contienen información general de los módulos e información más específica de sus bloques.
Archivos *.grc	Son los que contienen el <i>flowgraph</i> creado a partir de la herramienta gráfica <i>GNU Radio Companion</i> .
Archivos *.h	Son las bibliotecas de los bloques de procesado de señal.
Archivos *.i	Son los encargados de la comunicación entre los bloques de procesado de señal y la interfaz en Python.
Archivos *.py	Son programas (normalmente aplicaciones y ejemplos) escritos en lenguaje de programación Python.
Archivos *.xml	Son los archivos necesarios para que los bloques de procesado aparezcan en <i>GNU Radio Companion</i> . En ellos se definen los parámetros del bloque, el tipo de dato a utilizar, el número de puertos de entrada y de salida, etc.

Tabla C-3 Listado de archivos de los módulos de *GNU Radio*.

Además, en todos los módulos y en sus respectivas carpetas hay un fichero llamado *CMakeList.txt*. Este fichero es el encargado de indicar qué ficheros compilar, qué librerías y ejecutables generar, qué programas externos utilizar, etc.

Para más información sobre el funcionamiento de *GNU Radio*, se deja el manual y la documentación de su API:

- <https://gnuradio.org/doc/doxygen/>

D. Construcción de Módulos y Bloques en *GNU Radio Companion*

Como ya sabemos, *GNU Radio* es una herramienta de desarrollo libre que provee bloques de procesado de señal para implementar sistemas SDR, pero también es una herramienta de desarrollo abierta que permite, por tanto, crear y añadir nuevos módulos y bloques. Estos módulos que se agregan a posteriori no forman parte del núcleo de componentes de *GNU Radio* (*In-Tree components*), sino que son instalados fuera del directorio raíz, motivo por el cual reciben el nombre de módulos OOT (*Out-Of-Tree*).

Gracias a los módulos de *GNU Radio*, se extiende la funcionalidad de la plataforma, ya que, al construir bloques de procesado con código propio, se logra mayor poder y flexibilidad.

Este apéndice tiene por objetivo indicar los pasos a seguir para llevar a cabo la inclusión de nuevos módulos (*Out-Of-Tree*) y bloques en el proyecto *GNU Radio*.

A modo de ejemplo, se va a crear un módulo llamado *Ejemplos* y un bloque llamado *bloque_ejemplo1*. IMPORTANTE: en este texto se asume que se ha completado satisfactoriamente la instalación de *GNU Radio* mediante PyBOMBS.

Generación de un nuevo módulo OOT

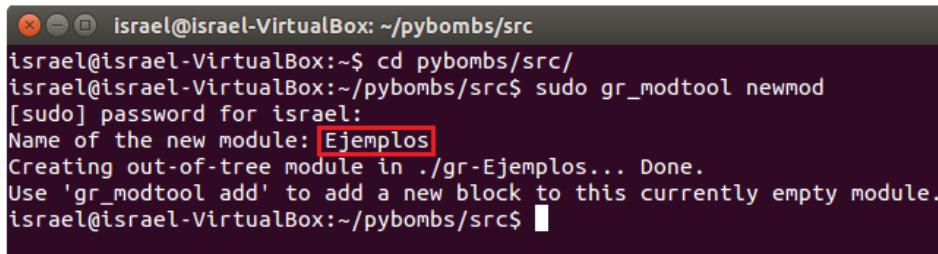
Primeramente nos situamos en el la carpeta *src* que es donde está instalado *GNU Radio* y los demás módulos OOT (en el caso de que se hubiera instalado alguno). Para ello ejecutamos en un terminal el siguiente comando:

```
cd pybombs/src
```

Para crear un nuevo módulo bastará con escribir en pantalla el siguiente comando:

```
sudo gr_modtool newmod
```

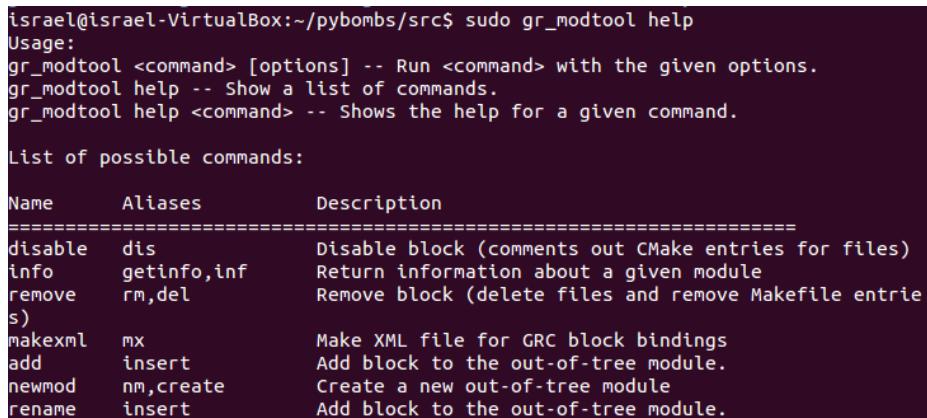
Tal y como se muestra en la siguiente figura, posteriormente se pedirá que ponga nombre al nuevo módulo.



```
israel@israel-VirtualBox:~/pybombs/src
israel@israel-VirtualBox:~$ cd pybombs/src/
israel@israel-VirtualBox:~/pybombs/src$ sudo gr_modtool newmod
[sudo] password for israel:
Name of the new module: Ejemplos
Creating out-of-tree module in ./gr-Ejemplos... Done.
Use 'gr_modtool add' to add a new block to this currently empty module.
israel@israel-VirtualBox:~/pybombs/src$
```

Figura D-1 Creación de un módulo OOT.

El comando `gr_modtool` permite crear, modificar y eliminar módulos y/o bloques de procesos. Para ver la lista de opciones que ofrece, escribir **`sudo gr_modtool help`**.



```
israel@israel-VirtualBox:~/pybombs/src$ sudo gr_modtool help
Usage:
gr_modtool <command> [options] -- Run <command> with the given options.
gr_modtool help -- Show a list of commands.
gr_modtool help <command> -- Shows the help for a given command.

List of possible commands:

Name      Aliases      Description
=====  ======  ======
disable   dis          Disable block (comments out CMake entries for files)
info      getinfo,inf  Return information about a given module
remove    rm,del       Remove block (delete files and remove Makefile entries)
makexml   mx           Make XML file for GRC block bindings
add       insert        Add block to the out-of-tree module.
newmod   nm,create    Create a new out-of-tree module
rename   insert        Add block to the out-of-tree module.
```

Figura D-2 Opciones del comando `gr_modtool`.

Una vez se ha creado el módulo *Ejemplos*, aparecerá en el directorio (*pybombs/src*) una carpeta llamada *gr-Ejemplos*. Si la revisamos, veremos que nos aparecen todas las carpetas propias de un módulo de *GNU Radio* (véase tabla C-2)

**Figura D-3 Contenido en la carpeta del módulo *Ejemplos*.**

Generación de un nuevo bloque de procesado:

Ahora vamos a generar el bloque de procesado *bloque_ejemplo1*. Estando en la carpeta del módulo (gr-Ejemplos), ejecutamos el siguiente comando:

```
sudo gr_modtool add
```

A continuación, nos pedirán que configuremos una serie de parámetros, tal y como se muestra en la siguiente figura:

```

israel@israel-VirtualBox:~/pybombs/src$ cd gr-Ejemplos/
israel@israel-VirtualBox:~/pybombs/src/gr-Ejemplos$ sudo gr_modtool add
[sudo] password for israel:
GNU Radio module name identified: Ejemplos
('sink', 'source', 'sync', 'decimator', 'interpolator', 'general', 'tagged_strea
m', 'hier', 'noblock')
Enter block type: general
Language (python/cpp): cpp
Language: C++
Enter name of block/code (without module name prefix): bloque_ejemplo1
Block/code identifier: bloque_ejemplo1
Enter valid argument list, including default arguments:
Add Python QA code? [Y/n]
Add C++ QA code? [y/N]
Adding file 'lib/bloque_ejemplo1_impl.h'...
Adding file 'lib/bloque_ejemplo1_impl.cc'...
Adding file 'include/Ejemplos/bloque_ejemplo1.h'...
Editing swig/Ejemplos_swig.i...
Adding file 'python/qa_bloque_ejemplo1.py'...
Editing python/CMakeLists.txt...
Adding file 'grc/Ejemplos_bloque_ejemplo1.xml'...
Editing grc/CMakeLists.txt...
israel@israel-VirtualBox:~/pybombs/src/gr-Ejemplos$
```

Figura D-4 Creación de un bloque de procesado.

El parámetro *type* sirve para definir el tipo de bloque. En *GNU Radio*, existen diferentes tipos de bloques: *general*, *sync*, *interpolator/decimator*, *source/sink*, *Hierarchical*, etc. Dependiendo de la elección del bloque, *gr_modtool* añadirá el código y las funciones correspondientes. Como *bloque_ejemplo1* no tendrá ninguna función, elegimos que sea de tipo general. También, elegimos que el lenguaje de programación del bloque sea C++ y escribimos su nombre. Respecto a las demás preguntas que se hacen (añadir los argumentos de bloque y códigos QA), no son relevantes en este ejemplo, así que las dejamos en blanco (pulsar la tecla *Intro*).

Cuando acabemos con las preguntas estándares y finalice el proceso, se tendrá creado *bloque_ejemplo1* y estará añadido en el módulo *Ejemplos*. Lo único que faltarán por hacer es modificar algunos archivos para poder llevar a cabo la implementación del bloque (definir entrada, salidas, tipos datos, función que desempeña, etc.).

Implementación del bloque de procesado:

Primero nos situamos dentro de la carpeta raíz del módulo. Una vez en ese directorio buscamos el archivo *bloque_ejemplo1_impl.cc* (que es el código fuente del bloque) ubicado en la carpeta *lib* y lo editamos con un editor de texto cualquiera (en este caso, se utiliza el programa *sublime text*).

Las partes del código que vamos a modificar son: <+MIN_IN+> (número de entradas mínimas), <+MAX_IN+> (número de entradas máximas), <+MIN_OUT+> (número de salidas mínimas), <+MAX_OUT+> (número de salidas máximas), <+ITYPE+> (tipo de flujo de datos de la entrada), <+OTYPE+> (tipo de flujo de datos de la salida). El bloque que se pretende implementar va a tener un puerto de entrada y un puerto de salida. Al puerto de entrada le llegarán datos del tipo *int* y dichos datos serán enviados directamente al puerto de salida, sin ser tratados, pues no se va a implementar ninguna función al bloque. En caso de querer atribuir una función al bloque, se ha de escribir el código correspondiente dentro del método *general_work()* que aparece en el archivo.

```
israel@israel-VirtualBox:~/pybombs/src/gr-Ejemplos$ cd lib
israel@israel-VirtualBox:~/pybombs/src/gr-Ejemplos/lib$ ls
bloque_ejemplo1Impl.cc  CMakeLists.txt  qa_Ejemplos.h
bloque_ejemplo1Impl.h   qa_Ejemplos.cc  test_Ejemplos.cc
israel@israel-VirtualBox:~/pybombs/src/gr-Ejemplos/lib$ sudo sublime-text bloque_ejemplo1Impl.cc
```

```

39     * The private constructor
40     */
41     bloque_ejemplo1Impl::bloque_ejemplo1Impl()
42     : gr::block("bloque_ejemplo1",
43                 gr::io_signature::make(1, 1, sizeof(int)),
44                 gr::io_signature::make(1, 1, sizeof(int)))
45     {}
46
47     /*
48     * Our virtual destructor.
49     */
50     bloque_ejemplo1Impl::~bloque_ejemplo1Impl()
51     {
52     }
53
54     void
55     bloque_ejemplo1Impl::forecast (int noutput_items, gr_vector_int &ninput_items_required)
56     {
57         /* <+forecast+> e.g. ninput_items_required[0] = noutput_items */
58     }
59
60     int
61     bloque_ejemplo1Impl::general_work (int noutput_items,
62                                         gr_vector_int &ninput_items,
63                                         gr_vector_const_void_star &input_items,
64                                         gr_vector_void_star &output_items)
65     {
66         const int *in = (const int *) input_items[0];
67         int *out = (int *) output_items[0];
68
69         // Do <->signal processing>
70         // Tell runtime system how many input items we consumed on
71         // each input stream.
72         consume_each (noutput_items);
73
74         // Tell runtime system how many output items we produced.
75         return noutput_items;
76     }
77
78 } /* namespace Ejemplos */
79 } /* namespace gr */
```

Figura D-5 Edición del archivo *bloque_ejemplo1Impl.cc*.

Una vez hemos editado el *bloque_ejemplo1Impl.cc*, tenemos que editar el archivo *Ejemplos_bloque_ejemplo1.xml* ubicado en la carpeta *grc* para que nos

aparezca como bloque disponible en la interfaz gráfica *GNU Radio Companion*. A continuación se muestra el archivo una vez modificado:

```

1  <?xml version="1.0"?>
2  <block>
3    <name>bloque_ejemplo1</name>
4    <key>Ejemplos.bloque_ejemplo1</key>
5    <category>Ejemplos</category>
6    <import>Import EJEMPLOS</import>
7    <make>EJEMPLOS.bloque_ejemplo1</make>
8  <!-- param node for every parameter you want settable from the GUI. -->
9    Sub-nodes:
10   * name
11   * key (makes the value accessible as $keyname, e.g. in the make node)
12   * type -->
13  <param>
14    <name>Ejemplos</name>
15    <key></key>
16    <type>int</type>
17  </param>
18
19  <!-- Make one 'sink' node per input. Sub-nodes:
20   * name (an identifier for the GUI)
21   * type
22   * vlen
23   * optional (set to 1 for optional inputs) -->
24  <sink>
25    <name>in</name>
26    <type>int</type>
27  </sink>
28
29  <!-- Make one 'source' node per output. Sub-nodes:
30   * name (an identifier for the GUI)
31   * type
32   * vlen
33   * optional (set to 1 for optional inputs) -->
34  <source>
35    <name>out</name>
36    <type>int</type>
37  </source>
38 </block>
39 |

```

Figura D-6 Edición del archivo *Ejemplos_bloque_ejemplo1.xml*.

Finalmente, el último paso es compilar el módulo y el bloque e instalarlos en *GNU Radio*. Para ello, nos situamos en la carpeta gr-Ejemplos y ejecutamos los siguientes comandos en el terminal:

sudo mkdir build

cd build

sudo cmake ./

sudo make

sudo make install

sudo ldconfig

A partir de este punto, ya se puede trabajar con el bloque en *GNU Radio Companion*.

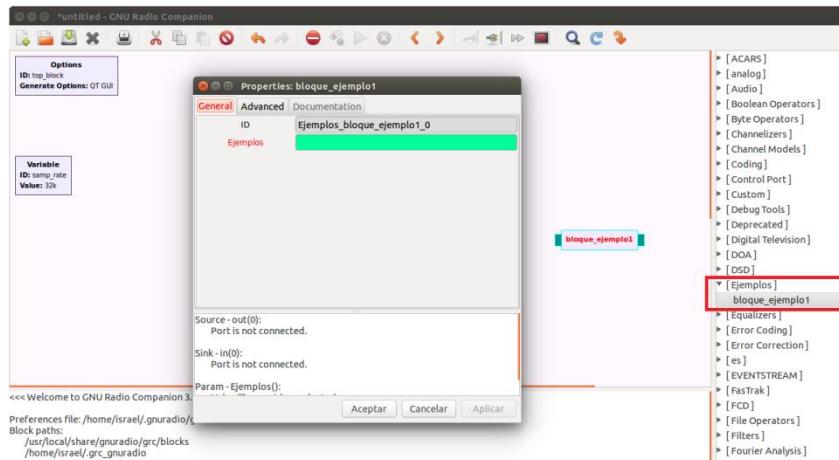


Figura D-7 *bloque_ejemplo1* en *GNU Radio Companion*.

Para más información se remite a los siguientes tutoriales:

- <https://gnuradio.org/redmine/projects/gnuradio/wiki/OutOfTreeModules>
- http://gnuradio.org/redmine/projects/gnuradio/wiki/Guided_Tutorial_GNU_Radio_in_Python
- http://gnuradio.org/redmine/projects/gnuradio/wiki/Guided_Tutorial_GNU_Radio_in_C++

Bibliografía

- [1] J. Mitola, G. Q. Maguire, “Cognitive Radio: Making Software Radios More Personal”, Personal Communications, IEEE (Volume: 6, Issue: 4), Aug 1999, 13-18.
- [2] S. Haykin, “Cognitive Radio: Brain-Empowered Wireless Communications”, IEEE Journal on Selected Areas in Communications, Vol. 23, No.2, Feb. 2005.
- [3] Federal Communications Commission, “FCC- ET Docket No. 03-108”, March 11, 2005.
- [4] National Telecommunications and Information Administration on FCC ET Docket No. 03-108, “Facilitating Opportunities for Flexible, Efficient, and Reliable Spectrum Use Employing Cognitive Radio Technologies”, February 15, 2005.
- [5] International spectrum regulatory community, ITU Wp8A, International Telecommunication Union.
- [6] “Improving Spectrum Usage through Cognitive Radio Technology”, IEEE USA Position, Nov 13, 2003.
- [7] Wireless Innovation Forum (<http://www.wirelessinnovation.org/what-are-cr-and-dsa>).
- [8] 1900.1, “IEEE Standard Definitions and Concepts for Dynamic Spectrum Access: Terminology Relating to Emerging Wireless Networks, System Functionality, and Spectrum Management Amendment 1: Addition of New Terms and Associated Definitions”, 2012.
- [9] Wireless Innovation Forum
([http://www.wirelessinnovation.org/Cognitive Radio Architecture](http://www.wirelessinnovation.org/Cognitive%20Radio%20Architecture)).
- [10] Raikel Bordón López, Samuel Montejo Sánchez, “La Radio Cognitiva y su Impacto en el Uso Eficiente del Espectro de Radio”, RIELAC, Vol.XXXVI, p.42-55, Enero – Abril, 2015, ISSN: 1815-5928.

- [11] J. Mitola III, “Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio”, PhD thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2000.
- [12] Željko Tabaković, “A Survey of Cognitive Radio Systems”, Croatian Post and Electronic Communications Agency, Jurišićeva 13, Zagreb, Croatia.
- [13] Min Song, Chunsheng Xin, Yanxio Zhao, Xiuzhen Cheng, “Dynamic Spectrum Access: From Cognitive Radio To Network Radio”, Wireless Communications, IEEE (Volume: 19, Issue: 1), Feb. 2012, 23 – 29.
- [14] Héctor Poveda, Fernando Merchan, Jouvett García, “Sistemas de Radio Inteligencia de Tipo Overlay: Análisis de Técnicas de Mitigación de Interferencia”, July 29-31, 2015, Santo Domingo, Dominican Republic.
- [15] Beibei Wang, “Advances in Cognitive Radio Networks: A Survey”, Selected Topics in Signal Processing, IEEE Journal of (Volume:5 , Issue: 1), Feb. 2011, 5 – 23.
- [16] Ferran Casadevall, “Curs Radio Cognitiva”, Seminario de Radio Cognitiva, UPC, 2014.
- [17] Akyildiz, I.F., Won-Yeol Lee; Vuran, Mehmet C.; Mohanty, S., “A Survey on Spectrum Management in Cognitive Radio Networks”, Communications Magazine, IEEE (Volume:46 , Issue: 4), April 2008, 40 – 48.
- [18] Li-Chun Wang, Chung-Wei Wang, “Spectrum management techniques with QoS provisioning in cognitive radio networks”, Wireless Pervasive Computing (ISWPC), 2010 5th IEEE International Symposium on, 5-7 May 2010, 116 – 121.
- [19] Plan Marco de Actuaciones para la liberación del Dividendo Digital, se puede descargar de:
<http://www.minetur.gob.es/telecomunicaciones/es-ES/Novedades/Paginas/PlanMarcoActuacionesDividendoDigital.aspx>
- [20] “Reconfigurable Radio Systems (RRS); System requirements for Operation in UHF TV Band White Spaces”, ETSI TS 102 946 v1.1.1, July 2014.

- [21] Estuardo Rodríguez, “Implementación de un REM (Radio Environment MAP) en la Banda de TV para el Campus Nord”, Proyecto Fin de Carrera, Febrero 2016.
- [22] Christophe Bobda, “Introduction to Reconfigurable Computing: Architectures, Algorithms, and Applications”, ISBN:1402060882 9781402060885, página 313.
- [23] Rodger H. Hosking, “Software Defined Radio Handbook (10th Edition)”, Pentek, Inc., April 2013.
- [24] “USRP N200 Datasheet”, Ettus Research, September 2012, descargable desde <http://www.ettus.com/product/details/UN200-KIT>
- [25] “Selecting an RF Daughterboard”, Ettus Research, descargable desde <http://www.ettus.com/kb>
- [26] <http://www.ettus.com/product/details/VERT400>
- [27] Mohamed Abd El Ghany, Mostafa Amr, Mohamed Gamal, “Communication Systems Using USRP Kits”, ISBN: 978-3-659-56172-6, página 36.
- [28] Lu, L., X. Zhou, U. Onunkwo and G. Y. Li, “Ten years of research in spectrum sensing and sharing in cognitive radio. EURASIP”, Journal on Wireless Communications and Networking, 2012, (28), 1 -16.
- [29] Ferran Casadevall, “Sistemas Celulares: GSM y UMTS”, Diapositivas Tema 6 de Radiocomunicaciones, UPC, 2014.