UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

IMPLEMENTACIÓN Y ANÁLISIS DE UNA RED GPRS BASADA EN OPENBTS Y EQUIPO USRP

TESIS

PARA OPTAR POR EL GRADO DE:

MAESTRO EN INGENIERÍA (COMPUTACIÓN)

PRESENTA:

MARCO ANTONIO JUÁREZ AGUIRRE

TUTOR

DR. VÍCTOR RANGEL LICEA

FACULTAD DE INGENIERA – UNAM

MÉXICO, D.F. MAYO DE 2015

Jurado asignado

Presidente: **Dr. Gerardo Vega Hernández**

Vocal: **Dr. Víctor Rangel Licea**

Secretario: **Dr. Javier Gómez Castellanos**

Suplente: **Dr. Jorge Luis Ortega Arjona**

Suplente: **Dra. María Elena Lágarra Ramírez**

Lugar donde se realizó la tesis: México D.F.

Tutor de tesis

Dr. Víctor Rangel Licea

Dedicatoria

A mis padres: María y Antonio por educarme, apoyarme y forma a la persona que soy ahora.

A mis hermanos: Andrea, Ani y Eduardo por el apoyo y el cariño que me han brindado siempre.

A mi novia Claudia por su cariño y apoyo a seguir preparándome.

Agradecimientos

Al Dr. Víctor Rangel Licea por sus enseñanzas, dedicación y paciencia durante el desarrollo de este trabajo de tesis.

Al CONACYT, por la beca otorgada durante mis estudios de maestría.

A la UNAM por mi formación profesional.

Resumen

En los últimos años, el software libre se ha utilizado en el sector de las telecomunicaciones, como herramienta para la implementación de infraestructura celular, esto debido al bajo costo con respecto a la infraestructura tradicional.

El proyecto *Open Base Tranceiver Station* (OpenBTS) basado en software libre, es una aplicación para sistemas operativos Linux, la cual utiliza un radio programable (*Universal Software Radio Peripheral* USRP) para implementar una radio base telefónica de segunda generación.

El presente trabajo de tesis enfatiza en conceptos, implementación y análisis del desempeño de una radio base GPRS (*General Packet Radio Service*), utilizando OpenBTS versión 5 y un dispositivo USRP modelo N210, con el objetivo de evaluar su factibilidad en zonas rurales de México, donde no se cuenta con infraestructura telefónica.

Para la evaluación se realizó el análisis en la capa de transporte, concentrándonos en los valores de *throughput* y retardo en función de su esquema de codificación y configuración *mulitslot*. Se midió además, la cobertura máxima que el equipo USRP N210 puede alcanzar utilizando la banda más baja de GSM (850 MHz). Para ello se obtuvo el nivel de fuerza recibida por parte del dispositivo móvil a diferentes distancias y se calculó la relación (the *energy per bit to noise power spectral density ratio* (E_b/N_0).

Índice Principal

Resumen ..i
Índice de figuras ..v
Índice de tablas ..vii
1 Capítulo 1: Introducción ..1
1.1 Antecedentes ..1
1.2 Planteamiento del problema ..2
1.3 Objetivo ..2
1.3.1 Objetivo general ..2
1.3.2 Objetivos específicos ..2
1.4 Relevancia y contribución ..3
1.5 Estructura de la tesis ..3
2 Capítulo 2: Estado del arte ..5
2.1 Introducción ..5
2.2 Breve historia de las telecomunicaciones ..5
2.3 Software Defined Radio (SDR) ..7
2.4 Conceptos SDR ..9
2.4.1 Arquitectura SDR ..9
3 Capítulo 3: General Packet Radio Service (GPRS) ..12
3.1 Introducción ..12
3.2 Arquitectura GPRS ..13
3.2.1 Packet Control Unit (PCU) ..14
3.2.2 Serving GPRS Support Node (SGSN) ..14
3.2.3 Gateway GPRS Support Node (GGSN) ..14
3.2.4 Border Gateway (BG) ..15
3.2.5 Interfaces ..15
3.3 Plano de usuario ..16
3.3.1 GPRS Tunneling Protocol (GTP) ..17
3.3.2 Sub Network Dependent Convergence Protocol (SNDCP) ..17
3.3.3 Logical Link Control (LLC) ..17

3.3.4 Radio Link Control (RLC) 18
3.3.5 Medium Access Control (MAC) 19
3.4 Plano de control GPRS 21
3.4.1 GPRS mobility management (GMM) 21
3.4.2 Session Management (SM) 21
3.4.3 Base Station System GPRS Protocol (BSSGP) 22
3.4.4 Activación de contexto Packet Data Protocol (PDP) 22
3.4.5 Administración de calidad de servicio (QoS Management) 23
3.5 Interfaz de radio 23
3.5.1 Frame básico 23
3.5.2 Multiframe 24
3.5.3 Canales físicos y lógicos 26
3.5.4 Asignación Multislot 28
3.5.5 Burst Normal 28
3.5.6 Burst de acceso 29
3.6 Codificación del canal y Throughput 29
4 Capítulo 4: Entendiendo OpenBTS 35
4.1 Introducción 35
4.2 El proyecto OpenBTS 35
4.3 Aplicaciones 36
4.3.1 Transceiver 36
4.3.2 Base de datos 36
4.3.3 Asterisk 37
4.3.4 SIPAuthServe 37
4.4 Arquitectura 39
4.5 Requerimientos de Hardware 40
4.6 Software Defined Radio 41
4.6.1 Universal Software Radio Peripheral (USRP) 41
4.6.2 Teléfonos móviles 44
4.7 Configuración de GPRS 44
4.7.1 Configuración Multislot GPRS 47

4.7.2 Asignaciones de canales GPRS 47
4.7.3 Configuración del tamaño de buffer en Linux 47
5 Capítulo 5: Optimización de GPRS 43
5.1 Introducción 43
5.2 Transmission Control Protocol (TCP) 43
5.2.1 Control de Congestión TCP 44
5.2.2 Inicio lento (Slow Start) 44
5.2.3 Retransmisión y recuperación rápida 44
5.2.4 ACK Selectivo 45
5.3 Optimización TCP/IP 45
5.4 TCP Window Size 46
5.5 BDP (Bandwidth Delay Product) 46
5.6 MTU – Maximum Transmission Unit y MSS – Maximum Segment Size 46
5.7 Clasificación de enlaces GPRS. 47
6 Capítulo 6: Análisis del desempeño de GPRS 48
6.1 Introducción 48
6.2 Implementación del escenario 49
6.3 Configuración de banda GSM 50
6.4 Desempeño RTT con tamaño de paquete IP 52
6.5 Optimización del MTU (Maximum Transfer Unit) 54
6.5.1 FTP (File Transfer Protocol) 54
6.5.2 Optimización del MTU 55
6.6 Throughput en downlink utilizando multislot 57
6.7 Análisis en la ventana de recepción TCP 60
6.8 Cobertura del USRP N210 61
6.8.1 Ajustes de potencia 61
6.8.2 Selección del área geográfica 64
6.8.3 Mediciones de cobertura 64
6.8.4 Throughput 68
6.8.5 RTT 69
6.8.6 Retransmisiones 70

7 Capítulo 7: Conclusiones 61
7.1 Introducción 61
7.2 Contribución 61
7.3 Trabajo futuro 62
7.4 Conclusiones finales 63
8 Bibliografía 61
9 Glosario 63
10 Apéndices 63
10.1 Instalando OpenBTS V5 63
10.1.1 Crear una nueva llave 64
10.1.2 Instalación de transceiver 66
10.1.3 Configuración de OpenBTS 67
10.1.4 Instalación del registro de suscriptores y sipauthserve 68
10.1.5 Debugging OpenBTS 69

Índice de figuras

Figura 2.1: Diagrama de bloques SDR 10
Figura 3.1: Arquitectura GPRS 13
Figura 3.2: Pila de protocolos en el plano de usuario. 16
Figura 3.3: Combinación de TDMA con FDMA. 24
Figura 3.4: Estructura de 52 multiframes GPRS. 25
Figura 3.5: Burst normal. 29
Figura 3.6: Burst de acceso. 29
Figura 3.7: Jerarquía de encabezados. 30
Figura 3.8: Codificación del radio bloque. 31
Figura 4.1: Parámetros de configuración SIPAuthServe. 38
Figura 4.2: Arquitectura OpenBTS. 40
Figura 4.3: Panel frontal N210. 42
Figura 4.4: Antena VERT900 y Tarjeta Daughterboard SBX. 43
Figura 4.5: interfaz sgsntun. 45
Figura 6.1: Implementación del escenario. 49
Figura 6.2: Analizador de espectro Aaronia HF-4060. 50
Figura 6.3: Barrido en rango de 848 a 895 MHz. 51
Figura 6.4: Script para creación de paquetes de diferentes tamaños. 52
Figura 6.5: RTT vs tamaño del paquete IP. 53
Figura 6.6: Establecimiento de conexión FTP. 54
Figura 6.7: MTU de 1500 bytes utilizado por defecto en OpenBTS 55
Figura 6.8: Configuración *multislot* por defecto. 55
Figura 6.9: MTU 1500 y 3 PDCHs. 55
Figura 6.10: Porcentaje de retransmisiones 56
Figura 6.11: MTU de 788 bytes con 3 PDCHs. 57
Figura 6.12: *Throughput* con diferente configuración *multislo*t. 58
Figura 6.13: Throughput 3 PDCHs vs 4 PDCHs con CS-4. 59
Figura 6.14: Throughput con 3 PDCHs vs PDCHs con CS-1. 59

Figura 6.15: Ventana de recepción del MS. 61
Figura 6.16: Área seleccionada para realizar las mediciones. 64
Figura 6.17: RSL vs Distancia 67
Figura 6.18: Throughput VS Distancia con CS-4. 68
Figura 6.19: Throughput VS Distancia con CS-1. 69
Figura 6.20: RTT VS Distancia utilizando CS-4 69
Figura 6.21: RTT VS Distancia utilizando CS-1 70
Figura 6.22: Retransmisiones y ACKs duplicados con CS-4. 71
Figura 6.23: Tamaño de paquetes ACK duplicados y retransmisiones. 71
Figura 6.24: Retransmisiones y ACKs duplicados con CS-1. 72
Figura 10.1: Salida del comando *uhd_usrp_probe* 67
Figura 10.2: Ancho de banda de 200 KHz. 68

Índice de tablas

Tabla 2.1: Proyectos SDR. 9
Tabla 3.1: Cálculo de frecuencias en *uplink* y *downlink* 23
Tabla 3.2: Canales lógicos GPRS 26
Tabla 3.3: Clases *multislot.* 28
Tabla 3.4: Throughput de datos por cada CS. 30
Tabla 3.5: Parámetros de codificación. 31
Tabla 3.6: Máximo throughput por *radio block* 32
Tabla 3.7: Máximo throughput por número de TSs. 32
Tabla 4.1: LEDs del USRP N210. 42
Tabla 4.2: Características USRP N210. 42
Tabla 4.3: Antena VERT900 y Tarjeta Daughterboard SBX. 43
Tabla 6.1: Características de host 49
Tabla 6.2: Parámetros OpenBTS. 53
Tabla 6.3: Características en downlink. 57
Tabla 6.4: Throughput con diferente número de time slots 58
Tabla 6.5: Eb/N0 mínimo requerido pasa CS-1 y CS-4 65
Tabla 6.6: SNR y E_b/N_0 a partir del RSL medido. 67
Tabla 6.7: Eb/N0 dB teórico y práctico. 68

Capítulo 1: Introducción

1.1 Antecedentes

Con la llegada de varios estándares de transmisión de datos a principios de 1990, se originó una problemática de incompatibilidad y actualización de estos, debido a que su programación se encontraba en el hardware, lo que llevo a una solución llamada *Software Define Radio* (SDR), el cual inició un impacto revolucionario en el desarrollo de procesamiento de señales, donde la lógica del hardware es sustituido por software [1]. SDR lleva a cabo el procesamiento de señales tales como modulación, demodulación y codificación entre otros por medio de software, como resultado, su costo es significativamente más rentable que los sistemas implementados únicamente con hardware.

Uno de estos desarrollos; en el cual se centra este trabajo de investigación es OpenBTS [2] (*Open Base Transceiver Station*); una aplicación Unix de software libre que junto con un equipo *Universal Software Radio Peripheral* [3] (USRP) implementa la pila de protocolos de *Global System for Mobile Communications* (GSM) y *General Packet Radio Service* (GPRS).

Existen artículos y tesis sobre la implementación de OpenBTS que se centran en el análisis y desempeño de GSM [4, 5], así como su instalación y puesta en operación en zonas aisladas como es el caso de África e Indonesia [6,7]. Con la llegada de GPRS a OpenBTS se abrió un nuevo interés en el diseño y programación de la tecnología SDR para servicios de telefonía por conmutación de paquetes.

El entendimiento de la implementación de GPRS con SDR es de vital importancia para entender y diseñar generaciones de telefonía más avanzada (3G Y 4G).

Esto nos motiva a realizar estudios del comportamiento del equipo USRP con OpenBTS y valorar el uso de esta tecnología en zonas rurales de México, que permitan posibles mejoras para optimizar su operación e instalación.

1.2 Planteamiento del problema

La demanda de servicios de datos ha superado a las tradicionales llamadas telefónicas y mensajes de texto. La alta demanda de usuarios ha propiciado que las redes de telefonía se extiendan en nuevas regiones, sin embargo existen problemas para llevar esta tecnología a zonas rurales, donde la instalación de infraestructura es costosa y no es viable para zonas con poca población.

Esta problemática nos motiva a realizar investigaciones en OpenBTS y hardware programable (USRP N210), y sentar las bases para un análisis de viabilidad de esta tecnología en zonas rurales.

1.3 Objetivo

1.3.1 Objetivo general

Implementar y analizar el comportamiento de una red GPRS con OpenBTS y un equipo USRP N210 que permita caracterizarla.

1.3.2 Objetivos específicos

- Entender la plataforma SDR.
- Entender el principio de operación de los equipos USRP N210.
- Instalar y configurar una red GPRS con OpenBTS.
- Realizar mediciones de *throughput* en el enlace descendente de los esquemas de codificación CS-1 (Coding Scheme 1) y CS-2 implementados hasta el momento en OpenBTS.
- Estudiar el comportamiento TCP en la red GPRS utilizando una aplicación FTP.
- Obtener mediciones de la cobertura máxima que la red alcanza con el equipo USRP N210 utilizando una aplicación FTP.

1.4 Relevancia y contribución

Este trabajo de investigación de campo proporciona un nuevo conocimiento en el área de telecomunicaciones en México y abre una nueva línea de investigación en tecnología SDR, la cual es una tendencia que crece año con año.

Los resultados permiten evaluar estas tecnologías como nuevas alternativas para desplegar telefonía y servicios de internet en zonas aisladas de México, donde es difícil la instalación de infraestructura actual.

Por ser una tecnología desarrollada con software libre permite la enseñanza en las universidades y futuras implementaciones en otros estándares de redes inalámbricas como WiFi, UMTS y LTE.

1.5 Estructura de la tesis

Este trabajo de investigación está dividido en 7 capítulos.
El capítulo 2 describe el estado del arte, en ella se describe una breve historia de las telecomunicaciones, el surgimiento SDR, y algunos de los desarrollos de esta tecnología. Se describen los conceptos de la tecnología SDR y presenta los antecedentes y desarrollo del proyecto OpenBTS.

En el capítulo 3 se define los conceptos más importantes de la tecnología GPRS como arquitectura, pila de protocolos, señalización y esquemas de codificación.

En el capítulo 4 se describe el concepto de OpenBTS, su arquitectura, instalación y configuración. Se describen los requerimientos de hardware mínimos y las características el equipo USRP N210 para que la red opere con éxito.

En el capítulo 5 se explican los parámetros TCP más importantes en GPRS que de tomar en cuenta para que la red tenga buen desempeño.

En el capítulo 6 se explica la implementación del escenario de la red GPRS, la descripción de cada uno de los componentes de conforma la red, así como la recolección de los resultados y análisis de su comportamiento en términos de *throughput,* retardo, ruido y cobertura.

Finalmente, el capítulo 7 muestra la contribución que aporta esta tesis, los trabajos futuros derivados de esta investigación y conclusiones finales.

Capítulo 2: Estado del arte

2.1 Introducción

Durante las últimas dos décadas, la industria de las telecomunicaciones ha desarrollado numerosas tecnologías. Las comunicaciones móviles han sido un éxito junto con los avances en computación, proporcionando a los dispositivos móviles de nuevos servicios. Para entender las nuevas tecnologías de comunicaciones móviles de hoy en día, es importante entender de donde vienen y cómo han evolucionado.

En este capítulo, se da una breve historia de las telecomunicaciones y el surgimiento de la tecnología SDR. También se discuten trabajos relacionados con implementaciones de esta tecnología en diferentes estándares de comunicaciones inalámbricas y se provee una introducción a los conceptos SDR más importantes para entender su funcionamiento.

Conocer la tecnología SDR es uno de los puntos clave para entender el funcionamiento de OpenBTS sobre el cual está enfocado este trabajo.

2.2 Breve historia de las telecomunicaciones

En 1946 la *Federal Communications Commission* (FCC) de Estados Unidos aprobó el servicio de telefonía de AT&T. En 1947 AT&T introduce el concepto celular para la reutilización de frecuencias. En 1969 la Federal *Communications Commision* (FCC) adjudicó proporciones del espectro de radio frecuencia para teléfonos móviles. En la década de los 70 el *Post Office Code Standarization Advisory* (POCSAG) estandarizó el número de marcación.

La primera generación de telefonía celular inició en la década de 1980. Esta generación utilizaba técnicas de transmisión analógicas para tráfico de voz. Los estándares más exitosos fueron: *Total Access Communications System* (TACS), *Advance Mobile Phone Service* (AMPS) y *Nordic Mobile Telephone* (NMT) [8].

Con la segunda generación de telefonía se dio un salto a las técnicas de transmisión digitales. Existen 4 estándares: *Global System for Mobile Telecommunications* (GSM), *Digital AMPS* (D-AMPS), *Code Division Multiplex Access* (CDMA) y *Personal Digital Celluar* (PDC). Estos sistemas soportaban una tasa de transmisión menor a 10 kbps. Esta generación tuvo su éxito masivo en la década de 1990 (junto con Internet) lo que llevó a difundirse por todo el mundo e hizo posible servicios de datos tales como *Short Message Service* (SMS), Wireless Application Protocol (WAP), *bluetooth* y WiFi [8].

En la generación de telefonía 2.5G y 3G se amplían características de ancho de banda y manejo de datos. Se ofrecen nuevos servicios como teleconferencia, televisión, acceso a internet y descarga de archivos con tasas de transferencia desde 171 en GPRS hasta 42 Mbps en *Evolved High-Speed Packet Access* (HSPA+) [9].

Con la llegada del estandar LTE-advance para redes 4G a finales de 2009, se aumentan las tasas de transferencia hasta 1Gbps, utilizando *Orthogonal Frequency-Division Multiple Access* (OFDMA) como nueva técnica en la capa física.

Con las tecnologías mencionadas surge la problemática de que sus radios y protocolos son basados en hardware, por lo que su reprogramación y reconfiguración es mínima. Esta carencia de flexibilidad provoca problemas para corregir errores en el firmware y hardware al no ser modificable. Las diferentes implementaciones en algunos casos crean incompatibilidades entre los estándares desarrollados y sus equipos, complicando aspectos en la prestación y actualización de servicios. Además, estos equipos son limitados en cuanto a funcionalidad y capacidades de hardware y no pueden ser reconfigurados con nuevos protocolos.

La tecnología SDR surge para solucionar estos problemas de incompatibilidad, actualización e interoperabilidad, logrando una eficiente manera de desarrollar, actualizar y modificar tecnologías sin incurrir en gastos costosos de nuevos equipos.

2.3 Software Defined Radio (SDR)

El término *Software Defined Radio* (SDR) se le acredita al Dr. Joseph Mitola [10] quien publicó el primer artículo en 1992 y se refirió a una clase de radios que pueden ser programados y reconfigurados vía software.

La primera implementación SDR fue dentro de un proyecto militar en Estados Unidos llamado *SpeakEasy,* el cual tenía como objetivo implementar más de 10 tecnologías en comunicaciones inalámbricas en un equipo programable el cual operaria dentro un rango de frecuencia de 2 MHz hasta 200 MHz. Este proyecto inicia en 1991 y toma 4 años para cumplir con los objetivos, más 15 meses de mejoras. SpeakEasy es el primer proyecto conocido que trabajado con *Field Programmable Gate Array* (FPGA) para procesamiento de datos.

Actualmente, SDR es una tecnología que ha tomado gran auge en todo el mundo y sobre el cual se han realizado diferentes investigaciones y desarrollos. A continuación, se mencionan algunos de los proyectos más influyentes.

2.3.1.1 GNU Radio

GNU Radio es un proyecto de código abierto bajo la licencia *General Public License* (GPL) , que proporciona una librería para implementar procesamiento de señales mediante programación gráfica por bloques y permite el control de todo el hardware USRP como: tasa de muestreo, parámetros de frecuencia, filtros digitales, moduladores, en otros [11]. Su interfaz gráfica está implementada en lenguaje XML.

Existen varios artículos enfocados a realizar investigaciones con GNU Radio para el desarrollo de diferentes tecnologías inalámbricas. Uno de estos es "*An IEEE 802.11a/g/p OFDM Receiver for GNU Radio"* de la Universidad de Innsbruck en Austria [12], donde se implementa un receptor OFDM utilizando GNU radio junto con equipos USRP N210.

2.3.1.2 LabVIEW

Laboratory Virtual Instrumentation Engineering Workbench (LabVIEW) es una plataforma de desarrollo para procesamiento de señales y sistemas de control. La programación es visual y fue creado por *Nacional Instruments* (NI) [13].

Las investigaciones con LabVIEW han sido extensas. EL artículo "*LabVIEW-Based Software-Defined Radio: 4-QAM Modem*" de la universidad de Texas en Dallas [14], diseñan y analizan resultados simulados de un *modem* utilizando *Quadrature Amplitude Modulation* (4-QAM).

2.3.1.3 Software-developed Radio (SORA)

Software desarrollado por Microsoft para el desarrollo de tecnologías inalámbricas. El software fue escrito en lenguaje C y ensamblador [15]. El artículo "*Sora: High Performance Software Radio Using General Purpose Multi-core Processors*" de la Universidad de Beijing en China [16], describe la implementación del sistema *SoftWiFi* utilizando la plataforma SORA, el cual es compatible con tarjetas 802.11a/b/g y logra un rendimiento equivalente a las tarjetas comerciales.

2.3.1.4 Simulink

Es un entorno de programación visual que trabaja sobre el entorno de programación de Matlab [17]. Su uso es amplio en el área de procesamiento digital se señales, ingeniería biomédica entre otros. El artículo "*Implementation of WIMAX IEEE 802.16e Baseband Transceiver on Multi-Core Software Defined Radio Platform*" [18], describe la plataforma de desarrollo *The Small Form Factor* (SFF) desarrollado por Texas Instruments [19], que junto con el entorno de desarrollo Simulink implementan un transmisor 802.16e.

2.3.1.5 OpenBSC

Proyecto de software libre bajo la licencia GPL de GNU enfocado a la implementación del *Base Station Controller* (BSC) de GSM para propósitos de investigación en el área de seguridad [20]. El software es compatible con equipos. Siemens BS-11 microBTS [19]. La implementación con este software se describe en el artículo "Development of an open-source GSM femtocell and integrated core infrastructure" [22], utilizando un equipo USRP E100 [23]. A continuación se muestra en la tabla 2.1, el año de inicio los proyectos SDR descritos anteriormente.

Proyecto	Año
MathWorks - MATLAB / Simulink	2011
OpenBSC	2009
Microsoft – SORA	2009
GNU Radio	2001
National Instruments – LabVIEW	1986

Tabla 2.1: Proyectos SDR.

2.4 Conceptos SDR

En esta sección se describe la arquitectura de hardware sobre la que opera la tecnología SDR y se explica cada una de sus componentes.

2.4.1 Arquitectura SDR

La tecnología SDR trabaja sobre un transmisor/receptor digital genérico el cual está compuesto por tres bloques básicos: sección IF (tarjeta madre), *front end* (tarjeta hija) y la sección banda base [24]. En la figura 2.1 se observa esta arquitectura. Un sistema SDR es aquel en el cual el procesamiento banda base así como los módulos *DDC/DUC* son programables. Los protocolos de capa de enlace y capa física son implementados en software.

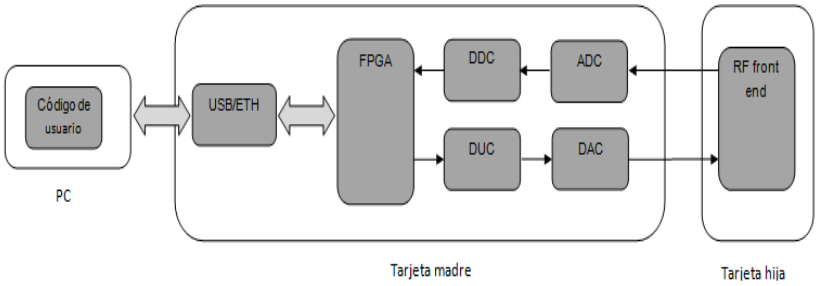

Figura 2.1: Diagrama de bloques SDR

- La sección *front end* utiliza un circuito analógico de radio frecuencia y es responsable de recibir y transmitir señales a una determinada frecuencia.
- La sección IF realiza la conversión digital a analógica y viceversa, a través de convertidores *Analog to Digital Converter* (*ADC) / Digital to Analog Converter (*DAC*)*. Está formado por una FPGA que se encarga de realizar todo el proceso digital requerido y reemplaza el tradicional procesamiento de radio.
- Los módulos *DDC/DUC* se encargan de convertir las señales de banda base a frecuencia intermedia (IF) y viceversa. Esencialmente estos bloques realizan la modulación de la señal para transmitir y demodulación al recibir la señal.
- La sección banda base se encuentra en la computadora y es donde se encuentra la toda la complejidad, aquí el código es ejecutado para realizar todo el proceso banda base.

Capítulo 3: General Packet Radio Service (GPRS)

3.1 Introducción

En este capítulo se describe los fundamentos de GPRS como su arquitectura, pila de protocolos y capa física.

En la arquitectura se describen los nuevos componentes que son agregados a la ya existente arquitectura GSM para operar por conmutación de paquetes.

En la pila de protocolos describe los protocolos de plano de usuario y control, así los procedimientos para establecer una conexión lógica entre los *Mobile Stations* (MSs) y la red GPRS.

En la capa física, se describe la estructura del *frame*[1] así como los canales lógicos y físicos. Se realiza además un análisis del *throughput*[2] alcanzado en esta tecnología.

La tecnología GPRS junto con SDR son las bases para entender la implementación de OpenBTS, y con ello realizar una correcta configuración y análisis de su desempeño.

[1] Frame. Secuencia de bits transmitida por unidad de tiempo.
[2] Throughput. Cantidad de datos exitosamente entregados sobre un canal de comuncación.

3.2 Arquitectura GPRS

GPRS es un servicio de conmutación de paquetes 2.5G estandarizado por la *European Telecommunications Standards Institute* (ETSI), el cual soporta velocidades de hasta 171.2 Kbps en canales de radio GSM [25]. Está diseñado para compartir recursos de la capa física junto con GSM. Sobre la capa física, la pila de protocolos es distinta y existe junto a la tecnología de conmutación de paquetes de GSM. Una radio base 2.5G contiene dos sistemas independientes; uno para servicios de conmutación de circuitos y otro para conmutación de paquetes.

En la interfaz de radio, los recursos son asignados a los MS temporalmente en contraste con conmutación de circuitos, en donde los recursos son asignados a los MSs durante toda la duración de la llamada. En GPRS los recursos sólo son asignados durante de la duración del flujo de los paquetes IP.

La arquitectura está diseñada para proveer una interconexión con conmutación de circuitos de la red GSM. GPRS requiere cambios en la arquitectura GSM. En la figura 3.1, se describen estos nuevos componentes con bordes discontinuos agregados a la arquitectura GSM.

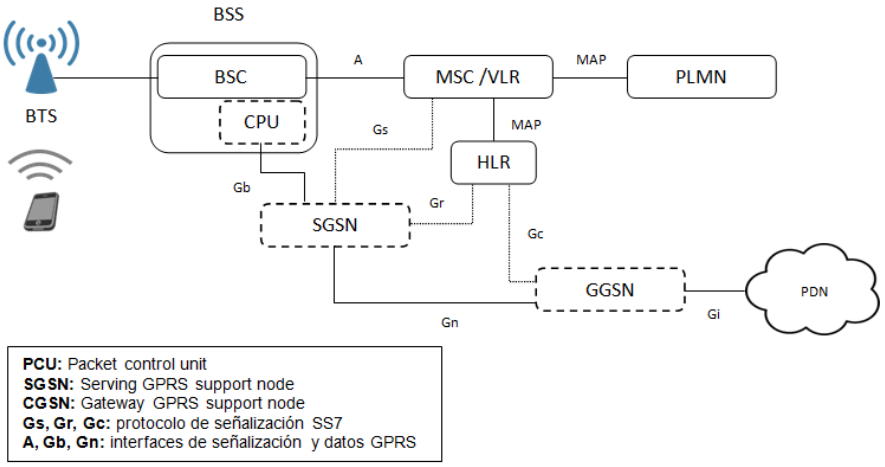

Figura 3.1: Arquitectura GPRS

3.2.1 Packet Control Unit (PCU)

El *Base Station Subsystem* (BSS) se extiende agregando un módulo llamado PCU, el cual separa el tráfico entre conmutación de circuitos y paquetes que los usuarios envían. Además, realiza funciones de administración de recursos de la red GPRS [26].

3.2.2 Serving GPRS Support Node (SGSN)

Es un componente primario, el cual se encarga del enrutamiento los paquetes de entrada y salida hacia cualquier subscriptor que se encuentra físicamente dentro del área de cobertura del SGSN [26].

Este componente provee:

- Cifrado y autenticación.
- Administración y configuración de sesiones de los suscriptores.
- Administración de movilidad como *roaming* y *handover* entre redes.
- Administración del enlace lógico del dispositivo móvil suscriptor.

3.2.3 Gateway GPRS Support Node (GGSN)

Es la interfaz externa hacia redes TCP/IP, accede a funciones de los *Internet Service Providers* (ISPs) tales como *routers*[3] y acceso remoto a servidores RADIUS [26]. Existen usualmente dos o más GGSNs con propósitos de redundancia, por si alguno de ellos falla. Este componente provee:

- Enrutamiento de los paquetes del usuario destino que vienen de redes externas hacia el SGSN.
- Enrutamiento de paquetes del usuario interno hacia la red externa.
- Proporciona interfaces a redes TCP/IP.
- Asigna IPs dinámicas y estáticas apoyandose de servidores DHCP y RADIUS.
- Está involucrado en establecer túneles con SGSN y con otras redes externas y VPN.

[3] Routers. Dispositivos que se encarga de encaminar paquetes IP en la red.

3.2.4 Border Gateway (BG)

Es un *router* que proporciona un túnel directo GPRS entre diferentes operadores de red GPRS. Proporciona mayor seguridad que enviar el tráfico a través de internet. El BG operará una vez que se haya establecido el acuerdo de servicio de *roaming*[4] entre las operadoras. Esencialmente permite a un suscriptor de *roaming* acceso a la intranet de una compañía a través de GGSN por medio de una red PLMN [26].

3.2.5 Interfaces

GPRS cuenta con un nuevo conjunto de interfaces las cuales han sido asignadas como G_x donde x denota una variedad de interfaces las cuales son descritas a continuación [27]:

- G_b Entre el SGSN y el PCU. Transporta tráfico GPRS y señalización entre GSM y GPRS. Frame Relay es utilizado para esta interface.
- G_r Entre el SGSN y HLR, señalización *Mobile Application Part* (MAP) para el almacenamiento y recuperación de datos de los abonados.
- G_n Entre el SGSN y GGSN para señalización de control en la administración de sesión y movilidad, esta interfaz hace uso de GTP para enviar la señalización.
- G_i Entre GGSN y PDNs, utilizado para el envío de paquetes IP entre la estación móvil y la red externa.
- G_s Entre SGSN y MSC/VLR para el uso de operaciones simultáneas entre GPRS y GSM. Provee administración de movilidad para las estaciones móviles que están registrados en GSM y GPRS.
- G_c Entre GGSN y HLR para la obtención de datos del MS.

[4] Roaming. Capaciadad del dispositivo móvil de conectarse a una radio base diferente a su operadora contratada.

3.3 Plano de usuario

La arquitectura de protocolos está organizada en dos planos, el plano de usuario y el plano de control. El plano de usuario es responsable de la transmisión de los datos de usuario en *downlink* y *uplink*. Para que una transferencia se lleve a cabo, el plano de usuario necesita información de las entidades involucradas como direcciones de red, estado de la red, actualización de opciones de protocolos, entre otras. Esta información debe ser obtenida al inicio de la sesión y debe estar siendo actualizada durante ésta.

El plano de usuario consiste en una pila de protocolos para la transmisión de datos. La figura 3.2 describe la pila de protocolos únicamente de los nodos involucrados en el flujo IP.

Esta pila está diseñada principalmente para el transporte de *Packet Data Protocol* (PDP) basado en los protocolos TCP/IP o UDP/IP. La ruta de transmisión punto a punto es asegurada mediante procedimientos de control.

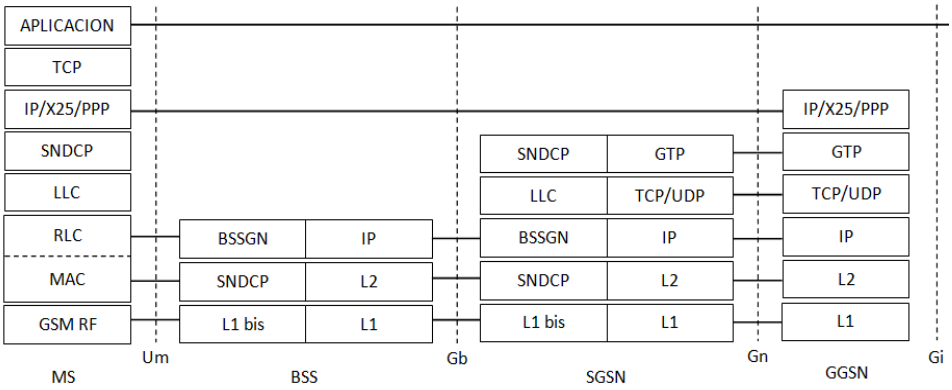

Figura 3.2: Pila de protocolos en el plano de usuario.

GPRS utiliza la capa de red para transportar paquetes IP entre el SGSN y GGSN. Esta descripción muestra el flujo de datos en *downlink* de una PDN al MS. Una vez que la conexión lógica entre el MS y GGSN es establecida, el MS obtiene una dirección IP y puede enviar paquetes vía GPRS. Este procedimiento es conocido como activación de contexto *Packet Data Protocol* (PDP) [28]. A continuación se describen cada uno de los protocolos involucrados.

3.3.1 GPRS Tunneling Protocol (GTP)

GTP mejora las funciones de ruteo estático IP para transmitir paquetes en ambas direcciones. Los PDUs llegan al GGSN sobre la interfaz Gi. Estos paquetes están encapsulados dentro de paquetes GTP y son transferidos al SGSN.

GTP utiliza TCP o UDP dependiendo si se necesita una conexión confiable. GTP provee por cada paquete un identificador de túnel *Tunnel Identifier* (TID), el cual identifica el destino y transacción del paquete [29].

3.3.2 Sub Network Dependent Convergence Protocol (SNDCP)

La tarea principal de SNDCP es transportar datos de la capa de red de forma transparente [30]. SNDCP recibe los paquetes IP llamados *Network Protocol Data Unit* (N-PDU) y los adapta a *Sub Network Protocol Data Unit* (SN-PDU) para que puedan ser compatibles con la red GPRS [9]. Los SN-PDUs formados por SNDCP son enviados a través de la interfaz de radio en un *Temporary Block Flow* (TBF).

SNDCP provee compresión de datos y cabeceras de las capas TCP/IP con el objetivo de mejorar la eficiencia del canal. Segmenta los paquetes N-PDUs de máximo 1520 bytes (octetos). Los procedimientos y compresión y segmentación son realizados cuando excede el valor del parámetro n201 en la capa LLC.

Existen dos tipos de cabeceras SN-PDU, para modo acuse y sin acuse de recibo. Estos modos son definidos en la capa LLC [9].

3.3.3 Logical Link Control (LLC)

La capa LLC es una derivación del protocolo *High-Level Data Link Control* (HDLC). Provee un enlace lógico entre el SGSN y el MS. El *Temporary Logical Link Identifier* (TLLI) es utilizado para identificar el enlace lógico entre la estación móvil y el SGSN.

La capa LLC provee servicios como el cifrado entre el enlace, control de flujo y es independiente de los protocolos de interfaz de radio, con el objetivo de ser compatible con nuevos protocolos [31]. La capa LLC provee entrega de PDUs LLC punto a punto en modo acuse y sin de recibo. El modo acuse de recibo tiene un mecanismo de recuperación de errores, en este modo se pueden enviar peticiones de retransmisión de las tramas para las cuales el acuse de recibido no ha sido recibido.

Provee detección de errores de PDUs corruptos verificando el *Frame Check Sequence* (FCS) en el *frame* LLC [31]. El FCS contiene el valor *Cyclic Redundancy Check* (CRC) el cual es calculado sobre la trama completa.

La capa LLC define un tamaño máximo en un frame LLC por medio del parámetro n201, el cual varía de 500 bytes para modo sin acuse de recibo y 1503 bytes para acuse de recibo [9]. Si los paquetes exceden estos tamaños serán segmentados.

El parámetro n202 define el número de bytes permitidos en la cabecera de capa 3 con un valor máximo de 5 [9].

3.3.4 Radio Link Control (RLC)

La capa RLC es responsable de la segmentación y reensamblado de los *frames* LLC [32]. Ésta segmenta las tramas LLC en bloques y los encapsula formando un bloque RLC. El tamaño de las tramas LLC segmentadas depende del esquema de codificación utilizado (CS-1 CS-2 CS-3 o CS-4).

El campo *Block Sequence Number* (BSN) se encuentra dentro de la cabecera RLC y es utilizado por el receptor para ordenar los bloques RLC en un orden secuencial, después la cabecera RLC es eliminada de los bloques y son reensamblados en tramas LLC.

Esta capa soporta dos modos de operación: con acuse de recibo y sin acuse de recibo. El modo acuse de recibo activa la operación de retransmisión selectiva. Es este modo donde el campo BSN es utilizado para las retransmisiones de un bloque perdido [33]. El modo sin acuse de recibo no garantiza la llegada de cada bloque RLC. Este modo es importante para aplicaciones que requieren un retardo constante. La capa RLC provee *Backward Error Correction* (BEC), el cual es encargado de activar las transmisiones selectivas.

Un bloque RLC de datos consiste en su cabecera, campo de datos y bits de reserva (*spare bits*). Cada bloque RLC de datos es codificado por alguno de los cuatro esquemas de codificación (CS-1 a CS-4). Si el contenido de un frame LLC no llena por completo un bloque RLC, la siguiente trama LLC se utilizara para llenarlo. Sin embargo, si la trama LLC fue la última en la transmisión del bloque RLC, ésta será llenada con bits de reserva. La estructura del bloque RLC depende de la dirección de la transmisión.

3.3.5 Medium Access Control (MAC)

La capa MAC se encarga de administrar el medio de transmisión compartido entre las diferentes estaciones móviles [32]. Esta capa interactúa directamente con la capa física. En la dirección *uplink*, la capa MAC administra los recursos entre las diferentes estaciones móviles que compiten por los recursos del canal. El protocolo de reservación que se utiliza para la contención está basado en el protocolo Aloha ranurado. La capa MAC administra de igual forma los recursos para los servicios que compiten entre ellos para servir a una estación móvil.

En la dirección *downlink* no existen contención, ya que solo un transmisor está presente. La capa MAC otorga prioridad a los datos de usuario. La señalización de datos tiene la mayor prioridad. La señalización también es multiplexada dentro del medio de transmisión.

Realiza también la corrección de errores, *interleaving*[5], modulación y monitoreo de la calidad del enlace de radio y control de potencia.

La capa MAC utiliza varios parámetros para transferir datos. Los dos más importantes se describen a continuación:

Temporary Block Flow. Es utilizado para identificar un conjunto de bloques RLC/MAC de una estación móvil en *downlink* o *uplink*. El TBF es único por cada dirección. Cada estación móvil que ocupe los recursos de radio se le asigna un TBF únicamente por el tiempo que dura la transferencia de datos. Por lo que durará hasta que todos los bloques RLC/MAC hayan sido transferidos y acusados de recibido.

[5] Interleaving. Técnica que consiste en organizar los bits de forma no contigua para proteccion de errores.

Temporary Flow Identify. Es un identificador para cada TBF en una dirección dada. El conjunto de un TBF, TFI y una dirección (*downlink o uplink*) identifica de manera única un bloque de datos RLC. El mensaje de control RLC/MAC contiene este conjunto de identificadores junto con el tipo de mensaje. El indicador TFI se incluye en la cabecera de los paquetes RLC para permitir la implantación del protocolo ARQ selectivo.

3.3.5.1 Asignación del canal en la capa MAC

El campo *Uplink State Flag* (USF) es un campo que la red utiliza para el control de multiplexación entre las estaciones móviles [32]. El USF está incluido en la cabecera MAC del bloque RLC/MAC en la dirección *downlink*. Este campo designa a la estación móvil que podrá transmitir en ese momento después del siguiente bloque en la dirección *uplink*. Este campo está compuesto por 3 bits por lo que 8 posibles usuarios pueden ser multiplexados en el mismo time slot. Si USF=7 indica que el slot está libre y puede utilizarse para el acceso. Para cualquier otro valor identifica a uno de las estaciones móviles activas en esa portadora. USF se puede usar en su forma normal o agregar redundancia como protección (precoded USF) a los bits USF.

Existen tres modos de asignación de recursos: asignación fija, dinámica y dinámica extendida. Estos modos aplican al canal *uplink*.

Asignación fija. La red asigna un conjunto de canales físicos al móvil para la transferencia de datos. Puede ser un rango de canales físicos de forma consecutiva o intercalada.

Asignación dinámica. El móvil lee el USF de la cabecera RLC/MAC del bloque de datos. Cuando este detecta su asignación USF, puede transmitir en uno o cuatro bloques RLC/MAC. Al estar en constante monitoreo el campo USF por el móvil, la asignación cambia dinámicamente.

Asignación dinámica extendida. Funciona en parte como la asignación dinámica. La diferencia radica en que el sistema puede especificar un rango de canales físicos al móvil para transmitir sus datos. Esto provee una mayor transferencia de datos.

3.4 Plano de control GPRS

El plano de control realiza la administración de estas funciones por medio de GPRS Mobility Management (GMM), Session Management (SM) y Quality of Service (QoS) Management.

3.4.1 GPRS Mobility Management (GMM)

GPRS mobility management se encarga del registro GPRS, administración de localización, autenticación y cifrado. GMM permite combinar procedimientos para GSM y GPRS y reducir la carga en la señalización [34]. Para obtener acceso a los servicios GPRS el MS debe iniciar un procedimiento de registro.

Las funciones de GMM están basadas en tres diferentes estados definidos entre el MS y SGSN. Estos estados son los siguientes:

1. **Idle**. No se encuentra registrado. No existen procedimientos asociados con el MS por lo que no es parte de la red GPRS.
2. **Standby**. Se encuentra registrado en la red. El MS y SGSN han establecido un contexto y se le han asignado recursos al MS.
3. **Ready**. Cuando un MS se encuentra registrado y se le han asignado recursos para realizar una transferencia de datos.

3.4.2 Session Management (SM)

EL SM comprende todas las funciones de señalización para acceder a *Packet Data Networks* (PDNs) externas. Soporta interconexión con redes basadas en IP las cuales puede ser internet o intranet [34].

La función principal de SM es la administración de contextos PDP en el MS. La negociación de *Quality of Service* (QoS) es realizada entre el MS y la red dentro de los procedimientos de activación o modificación de PDP. Por lo que los procedimientos de SM son de vital importancia para mecanismos de QoS.

3.4.3 Base Station System GPRS Protocol (BSSGP)

BSSGP es utilizado para el control de flujo de datos y control entre el SGSN y BSS [9]. Este mecanismo de control de flujo es utilizado para prevenir congestión y pérdida de datos. Sus principales funciones son:

- Provee *Quality of Service* (QoS) a los usuarios.
- Provee información de ruteo entre el BSS y el SGSN.
- Encapsulación de *frames* LLC en paquetes BSSGP para el envío entre el BSS y el SGSN.

3.4.4 Activación de contexto Packet Data Protocol (PDP)

La activación de contexto establece el inicio de una transferencia de paquetes. Este contexto puede permanecer activo después de la transferencia de paquetes.

La capa de red es la encargada de activar el contexto PDP en el MS, esta pregunta al SM para establecerla [34].

El contexto PDP describe características en la conexión hacia las redes externas como: el tipo de red, dirección de red, *Access Point Name* (APN), QoS y prioridad entre otras. La activación PDP se realiza en los siguientes pasos descritos a continuación:

- La estación móvil realiza una petición de activación de contexto PDP.
- El SGSN valida la petición basándose en la información de subscripción recibida del HLR durante el registro GPRS.
- El APN es enviado hacia el *Domain Name Server* (DNS) el cual se encuentra en el SGSN para encontrar la dirección del GGSN.
- Una conexión lógica es establecida por medio de GTP entre el SGSN y GGSN.
- El SGSN asigna una dirección dinámica a la estación móvil dentro de un rango de IPs o por medio de una autenticación remota a través de un servidor RADIUS.

3.4.5 Administración de calidad de servicio (QoS Management)

Para la administración de QoS, GPRS define atributos como: clase de precedencia, clase de retardo y clase de fiabilidad entre otros. Con las combinaciones de estos atributos se definen varios perfiles de QoS. Cada atributo es negociado por el MS y el SGSN mediante la administración de contextos PDP así como por GTP entre el SGSN y el GGSN [35].

3.5 Interfaz de radio

La interfaz Um es considerada uno de los aspectos centrales de GPRS ya que determina los aspectos principales de su rendimiento. Esta sección describe los canales físicos y lógicos que están envueltos en la comunicación de la interfaz de radio de GPRS, así como la estructura del *frame*.

3.5.1 Frame básico

GPRS utiliza la misma estructura *Time Division Multiple Access* / *Frequency Division Multiple Access* (TDMA/FDMA) de GSM para formar los canales físicos. La dirección de la estación base a la estación móvil se define como *downlink* mientras que de la estación móvil a la estación base es *uplink*. Los canales de frecuencia *downlink* y *uplink* tienen un ancho de banda de 200 KHz y son definidos a través de FDMA por medio de un índice llamado *Absolute Radio Frequency Channel Number* (ARFCN) [36]. La tabla 3.1 se muestra el par de frecuencias *downlink* y *uplink* asociadas a un ARFCN en particular así como su cálculo a partir de este.

Banda	**Designación**	**ARFCN**	**F_{UL}**	**F_{DL}**
GSM 850	GSM 850	128 - 251	824.2 + 0.2*(n – 128)	F_{UL} (n) + 45
GSM 900	P-GSM	1 − 124	890 + 0.2*n	F_{UL} (n) + 45
	E-GSM	0 − 124 975 − 1023	890 + 0.2*n 890 + 0.2*(n - 1024)	F_{UL} (n) + 45
	R-GSM	0−124 955 − 1023	890 + 0.2*n 890 + 0.2*(n – 1024)	F_{UL} (n) + 45
GSM 1800	DCS 1800	512 − 885	1710.2 + 0.2*(n - 512)	F_{UL} (n) + 95
GSM 1900	DCS 1900	512−810	1850.2 + 0.2*(n − 512)	F_{UL} (n) + 80

Tabla 3.1: Cálculo de frecuencias en *uplink* y *downlink.*

En la figura 3.3, se muestra como los canales son divididos en *frames* TDMA con un tamaño de 4.615 ms [25] .

Cada *frame* TDMA está dividido en ocho *Time Slots* (TS) o *burts*[6] numerados del 0 al 7 de igual tamaño, con una duración de 577 µs. Un *burst* normal contiene 156.25 bits durante el periodo de 577 µs. Este es el más común para transportar información como el tráfico de usuarios. La tasa de datos formada por los 8 time slots es 156.26 bits / 577 µs = 270.81 Kbps [9].

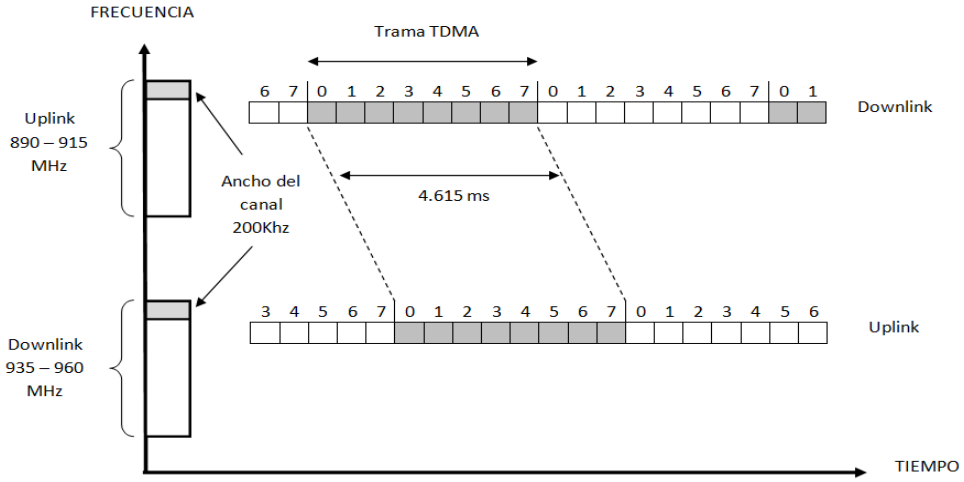

Figura 3.3: Combinación de TDMA con FDMA.

GPRS utiliza las mismas bandas de frecuencia y comparte los mismos canales físicos de GSM. Cada TS puede ser asignado a GSM o GPRS. Los TSs utilizados para GPRS se les llama *Packet Data Channel* (PDCH).

3.5.2 Multiframe

La unidad básica de un PDCH se llama *radio block*. Para transmitir un *radio block* se utiliza cuatro TSs en cuatro consecutivos *frames* TDMA, como puede observarse en la figura 3.4. Un PDCH es estructurado en *multiframes* lo cuales comprenden 52 *frames* TDMA y tienen una duración de 240 ms [28].

[6] Burst. transmisión de datos modulada en un time slot

Cada 13° *burst* es llamado idle *burst* y no es usado para transmitir datos dejando así 12 radio block en un multiframe.

Un *frame* TDMA dura:

$$576.9\,\mu s * 8 = 40615\,ms$$

Un bloque RLC son 4 frames TDMA:

$$4.615ms * 4 = 18.460\,ms$$

Por lo que el tiempo total del *multiframe* es:

$$18.460\,ms * 12 + 18.460\,ms = 240\,ms$$

Con esto el tiempo de transmisión de un *radio block* es de 20 ms. Un *radio block* contiene 456 bits pero debido al FEC menos bits de datos son transmitidos.

Un block RLC tiene una duración $\frac{240ms}{12} = 20ms$

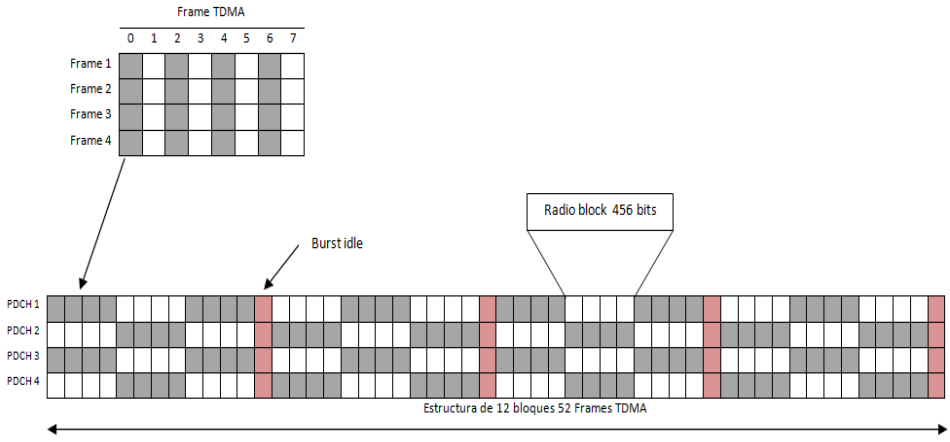

Figura 3.4: Estructura de 52 multiframes GPRS.

3.5.3 Canales físicos y lógicos

Existen dos tipos de canales: Los físicos y los lógicos. Un canal físico corresponde a un TS en un par de canales uno asignado para el *downlink* y otro para el *uplink*. Mientras que los canales lógicos son un tipo de información que es transportado en un canal físico. Un canal físico tendrá una tasa máxima de 270.81 Kbps / 8 = 33.85 Kbps. Existen dos tipos de canales lógicos; de tráfico o control, los cuales son multiplexados a diferentes canales físicos. Entre los canales lógicos de control tenemos tres subtipos que han sido definidos para GPRS: Broadcast (difusión), common control (Control común) y Associated (asociados). Estos canales lógicos junto con los de GSM (BCCH, CCCH, RANCH) son utilizados para acceder a la red y establecer una transferencia de paquetes. A continuación en la tabla 3.2, se muestra los diferentes canales lógicos y su función.

Grupo	nombre	Dirección	Función
PBCCH	PBCCH	Downlink	Broadcast
PCCCH	PRACCH	Uplink	Random access
	PPCH	Downlink	Paging
	PAGCH	Downlink	Access grant
	PNCH	Downlink	multicast
PTCH	PDTCH	Ambos	Data
	PACCH	Ambos	Control

Tabla 3.2: Canales lógicos GPRS

Packet Broadcast Control channel (PBCCH). Transmite información del sistema a todas las terminales en una celda.

Packet Common Control Channel (PCCCH). Es utilizado por el MS para iniciar una transferencia de paquetes o responder a los mensajes de *paging*[7].

Packet Paging Channel (PPCH). Es utilizado para buscar a un MS con el cual desea hacer una transferencia de paquetes en la dirección *downlink*.

Packet Access Grant Channel (PAGCH). Es utilizado para solicitar recursos para la transferencia de paquetes.

[7] Paging. Procedimiento de aviso para establecer una comunicación

Packet Notificación Channel (PNCH). Es utilizado para enviar una notificación mulitcast hacia un grupo de MSs. La notificación tiene la forma de asignación de recursos para una transferencia de paquetes.

Packet Data Traffic Channel (PDTCH). Es utilizado para transmitir datos.

Packet Associated Control Channel (PACCH). Es usado para enviar información de señalización referente a un MS como el ACK, control de potencia, mensajes de asignación de PDTCHs. Un PACCH es asociado con uno o más PDTCHs concurrentes a un MS.

Packet Timing Advance Control Channel / UPLINK (PTCCH). Es utilizado para estimación del temporizador en una estación móvil.

Packet Timing Advance Control Channel / DOWNLINK (PTCCH). Es utilizado para transmitir el temporizador a varias estaciones móviles.

3.5.3.1 Mapeo de canales lógicos a canales físicos

Múltiples canales lógicos pueden ser transportados dentro de canales físicos utilizando la estructura *multiframe* [32]. A continuación se muestran las tres posibles combinaciones:

$$\mathbf{1.}\ \boldsymbol{BROADCAST\ CONTROL + COMMON\ CONTROL + TRAFFIC + DEDICATED\ CONTROL}$$

$$\boldsymbol{PBCCH + PCCCH + PDTCH + PACCH + PTCCH}$$

$$\mathbf{2.}\ \boldsymbol{COMMON\ CONTROL + TRAFFIC + DEDICATED\ CONTROL}$$

$$\boldsymbol{PCCCH + PDTCH + PACCH + PTCCH}$$

$$\mathbf{3.}\ \boldsymbol{TRAFFIC + DEDICATED\ COMMON\ CONTROL}$$

$$\boldsymbol{PDTCH + PACCH + PTCCH}$$

3.5.4 Asignación Multislot

En GPRS se agrupan PDCHs para proveer una mayor tasa de transferencia en ambas direcciones, esta característica se le conoce como clase *multislot*. La clase *multislot* se define como el número de PDCHs para *downlink* y *uplink*.

El límite de cada clase es especificado como el número de PDCHs en *uplink* y *downlink* por *frame* TDMA. La clase *multislot* es enviada a la red durante el registro GPRS.

En la tabla 3.3 se muestran las configuraciones de las clases *multislot* que soporta OpenBTS para equipos de tipo 1 los cuales no pueden recibir y transmitir al mismo tiempo.

Clase multislot	Slots Downlink	Slots Uplink	Slots activos	configuración
1	1	1	2	1+1
2	2	1	3	2+1
3	2	2	3	2+2
4	3	1	4	3+1
5	2	2	4	2+2
6	3	2	4	3+2
7	3	3	4	3+3
8	4	1	5	4+1
9	2	3	5	2+3
10	4	2	5	4+2
11	4	3	5	4+3
12	4	4	5	4+4

Tabla 3.3: Clases *multislot.*

3.5.5 Burst Normal

El *burst* normal es utilizado en los canales PDTCH, PACCH, PAGCH, PTCCH, PAGCH, PPCH, PBCCH [32]. La información de estos canales es transmitida en *radio blocks* mapeados en cuatro consecutivos de estos *bursts*, en la figura 3.5 se muestra su estructura.

El *burst* normal contiene 26 bits para el *training sequence*, 2 boques de 57 bits de información y 2 bits para las *stealing flags*, las cuales son usadas para por GPRS para conocer el esquema de codificación usado. Contiene 3 bits como cola (*tail*) y 8.25 bits como periodo de guarda.

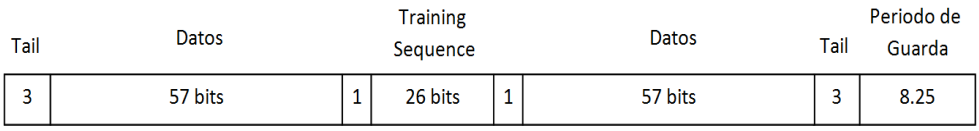

Figura 3.5: Burst normal.

3.5.6 Burst de acceso

El *burst* de acceso es utilizado para el acceso a la red y actualización de rastreo de área en el enlace *uplink* [32]. Este solo es utilizado por el PRANCH. Un *burst* de acceso contiene 41 bits de *training sequence*, 36 bits de información, 8 bits de cola y 68.25 bits de periodo de guarda. La figura 3.6 muestra su estructura.

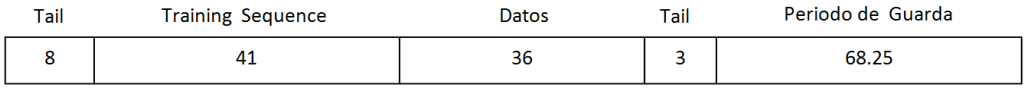

Figura 3.6: Burst de acceso.

3.6 Codificación del canal y Throughput

En la transmisión de datos sobre GPRS una jerarquía de protocolos está involucrada. Cada uno de estos protocolos agrega encabezados a los datos. Como se observa en la figura 3.7 los datos de la capa de aplicación son enviados a la capa TCP, agregando un encabezado de 32 bytes por cada segmento creado. En la capa IP se añaden 20 bytes a cada paquete. La capa SNDCP segmenta los paquetes IP en *frames* LLC agregando 4 bytes, posteriormente en la capa LLC son agregados 6 bytes más [37].

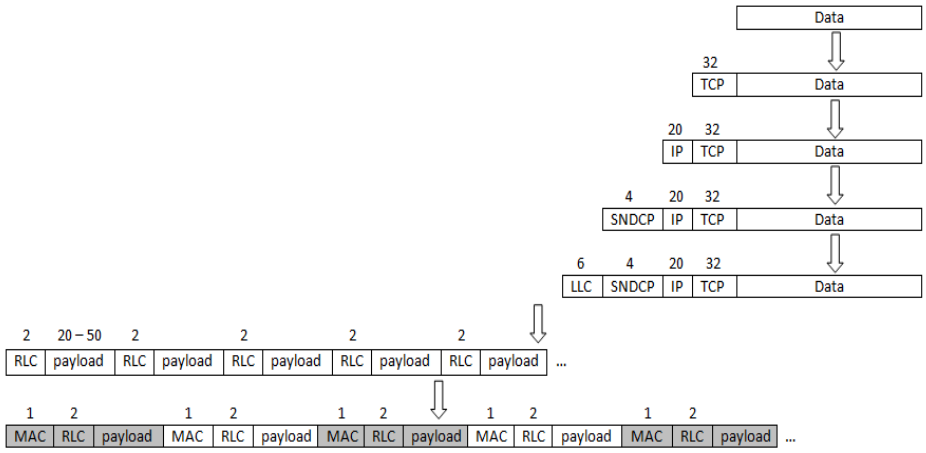

Figura 3.7: Jerarquía de encabezados.

La capa RLC segmenta cada *frame* LLC en *radio blocks*. La estructura y el número de bits de datos de un *radio block* depende del tipo de mensaje y esquema de codificación (20 para CS-1 a 50 bytes para CS-4). Cada *radio block* comienza con cabeceras MAC y RLC agregando en total 3 bytes de encabezado. La cantidad de bits de datos por cada esquema de codificación se muestra en la tabla 3.4.

Esquema de codificación	PAYLOAD RLC (bytes)	PAYLOAD RLC (bits)	Throughput de datos (bits)
CS-1	20	20*8 = 160	8000
CS-2	30	30*8 = 240	12000
CS-3	36	39*8 = 312	14400
CS-4	50	50*8 = 400	20000

Tabla 3.4: Throughput de datos por cada CS.

La figura 3.8 muestra como a cada *radio block* se le agrega un *Block Check Sequenc*e (BCS) el cual es usado para detectar errores que no pueden ser corregidos por *Forward Error Control* (FEC). Dependiendo del tipo de mensaje transmitido en un *radio block*, una secuencia de *radio blocks* formarán un canal lógico [25].

GPRS cuenta con 4 *Coding Schemes* (CS), desde CS-1 a CS-4. El nivel de cada uno de los estos esquemas trae cambios en el nivel de protección de los datos y *throughput*. El esquema a ser usado depende de un mecanismo que toma en cuenta las condiciones del ambiente llamado “adaptación de enlace” [25].

Los primeros 3 (CS-1 a CS-3) están basados en *Cyclic Redundancy Codec* (CRC) seguido de codificación convolucional, mientras que para CS-4 únicamente se utiliza CRC. La técnica *punctuaring* es aplicada a la salida del codificador convolucional para adaptarse al tamaño del *radio block*. En la figura 3.8, se muestra este proceso [25].

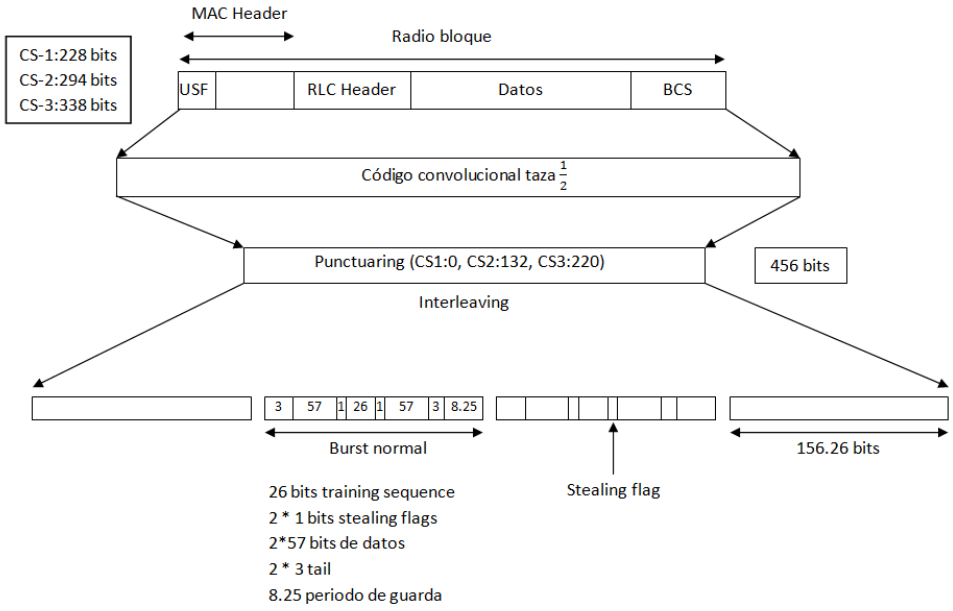

Figura 3.8: Codificación del radio bloque.

Las características de los cuatro esquemas se específica en la tabla 3.5. Se especifica el total de bits por cada CS [38]. En el caso del esquema de codificación CS-1 cada bloque consiste 181 bits de información, 40 bits de BCS y 4 bits de control/*tail*. Con un solo PDCH el esquema de codificación CS-1 tiene una tasa de 9.05 Kbps. De manera similar se puede observar en la tabla 3.5 para los esquemas de codificación con mayor tasa de datos.

CS	PDU	BCS	USF	Tail	Codificador Convolucional		Bits rem	Tasa efectiva	Tasa Kbps
CS-1	181	40	3	4	228	456	0	0.5	9.05
CS-2	268	16	6	4	294	588	132	0.64	13.4
CS-3	312	16	6	4	338	676	220	0.74	15.6
CS-4	428	16	12	*	*	456	0	1	21.4

Tabla 3.5: Parámetros de codificación.

En CS-2 el número de bits codificados es mayor que en CS1 (258 bits), su tasa efectiva es 0.64, el USF es precodificado a 6 bits. Agregados a los 268 bits y 6 bits del USF tenemos el BCS con 16 bits y 4 bits de cola (tail).

Estos 294 bits producen una salida de 588 en el codificador convolucional, al ser una cantidad mayor a la que puede transportar los 4 *burst*, se utilizan técnicas de perforado (*punctuaring*) para remover los bits sobrantes, en este caso 132 bits. Posteriormente son enviados sobre los 4 *bursts*. Con CS-3 el proceso es similar a CS-2 la diferencia radica en el número de bits de datos (338 bits) el cual hace uso nuevamente de la técnica de perforado.

CS-4 ofrece la tasa máxima de datos ya no hace uso del codificador convolucional, a estos bits únicamente se agrega 16 bits de BCS y 12 bits de pre codificación USF en lugar de 6 bits ya que este campo debe llevar más protección que en los casos de CS-2 Y CS-3. En la tabla 3.6 se muestra el *throughput* total por cada esquema de codificación.

Esquema de codificación	PDU-MAC/RLC (bytes)	PDU-MAC/RLC (bits)	Throughput RLC/MAC (bits)
CS-1	23	(23*8) = 184	184*50 = 9200
CS-2	33	(33*8) = 264	264*50 = 13200
CS-3	39	(39*8) = 312	312*50 = 15600
CS-4	53	(53*8) = 424	424*50 = 21200

Tabla 3.6: Máximo throughput por *radio block*

Una mayor tasa de datos se puede lograr utilizando asignación *multislot*. En la tabla 3.7 se muestran las tasas al aumentar el número de PDCHs.

Número de PDCHs	CS-1	CS-2	CS-3	CS-4
1	9200	13200	15600	21200
2	18400	26400	31200	42400
3	27600	39600	46800	63600
4	36800	52800	62400	84800

Tabla 3.7: Máximo throughput por número de PDCHs.

Capítulo 4: Entendiendo OpenBTS

4.1 Introducción

Para realizar el análisis de OpenBTS, es importante conocer su funcionamiento y su interacción con el equipo USRP N210 así como la configuración de ambos.

Es este capítulo, se da una introducción a OpenBTS y se describe cada uno de los componentes que conforman su arquitectura así como los requerimientos de hardware mínimos que debe cumplir el equipo para la instalación de OpenBTS.

Se menciona el *harware* de diferentes marcas que soportan OpenBTS y se describe los componentes que conforman el equipo USRP N210 de la marca Ettus Research, el cual es utilizado en este trabajo para poner en operación la radio base GPRS. Finalmente, se explica la configuración de OpenBTS realizada para que la red GPRS pueda ser desplegada con éxito.

4.2 El proyecto OpenBTS

El proyecto OpenBTS nace con el objetivo de revolucionar las redes móviles y sustituir las arquitecturas tradicionales por medio de software libre flexible. La primera versión del código fue liberado en 2008 [39], desde entonces ha proporcionado servicios en todo el mundo y ha sido adoptado en laboratorios y universidades.

OpenBTS hace uso de software definido por radio y ha redefinido la manera de diseñar redes móviles. Esta nueva tecnología nos permite la construcción de complejos transmisores de radio puramente con software. OpenBTS es una aplicación escrita en C++ basado en la plataforma Unix e implementa la pila de protocolos GSM/GPRS [39].

La flexibilidad de ser puramente código permite a programadores realizar innovaciones de manera más fácil, que utilizando hardware propietario.

El proyecto OpenBTS está liderado por la compañía Range Networks [40] bajo la licencia AGPLv3 [41].

4.3 Aplicaciones

OpenBTS es portable en cualquier sistema Unix de 32 bits. Éste comprende tres aplicaciones principales:

- **Transceiver**. software que controla la capa física del radio modem.
- **Base de datos**. Contiene todo los parámetros de configuración de OpenBTS.
- **Asterisk**. Conmutador *Voice over IP* (VoIP).
- **SIPAuthServe**. Administración de los subscriptores.

4.3.1 Transceiver

Controla el radio USRP N210 por medio de la interfaz Ethernet. Esta aplicación tiene implementada toda la capa física de GSM.

4.3.2 Base de datos

La aplicación OpenBTS administra su configuración en una base de datos en sqlite3 [42], la cual puede ser modificada por medio de *Command Line Interface* (CLI). Las tablas implementadas en OpenBTS son:

- **Tabla de configuración**. Los parámetros en esta tabla controlan toda la configuración de OpenBTS, como asignación IP para la red GPRS, ARFCN, control de potencia y registro entre otros.

- **Tabla IMSI**. Esta tabla lleva el control de TMSI-IMSI de cada MS registrado.
- **Tabla de canales.** Contiene el estatus en tiempo real de los canales físicos activos.

4.3.3 Asterisk

Asterisk es el conmutador VoIP el cual remplaza al *Mobile Switching Center* (MSC) en una red GSM convencional. Asterisk ve a cada MS como un usuario SIP con nombre definido como "IMSIxxxxxxxxxxxxxxx" donde las "x" representan un valor de 14 a 15 dígitos del IMSI extraídos del SIM de cada dispositivo. OpenBTS es invisible para la red VoIP. Asterisk depende de una base de datos Sqlite3 para su registro *Session Initiation Protocol* (SIP) y parte de su *dial plan*[8]. En OpenBTS la base de datos de registro es parte de la aplicación SIPAuthServe, el cual se encarga del registro de suscriptores. La aplicación SIPAuthServe, reemplaza el *Home Location Register* (HLR) y *Visitor Location Register* (VLR) en una red GSM convencional.

4.3.4 SIPAuthServe

SipAuthServe es el servidor de registro y autorización de usuarios SIP el cual procesa las peticiones de actualizaciones de OpenBTS y realiza las correspondientes actualizaciones en su base de datos de registro de subscriptores. En OpenBTS esta aplicación es llamada SIPAuthServe (*Subscriber Registry*). El registro a la base de datos de subscriptores es un registro SIP de Asterisk siguiendo el formato estándar de la tabla *sip_buddies*[9]. Los campos más importantes se muestran en la figura 4.1.

[8] Dial plan. Colección de contextos en los cuales se define la marcación y su comportamiento.
[9] sip_buddies. Tabla de asterisk donde es guardada la información de los usuarios.

username varchar(80),
-- nombre de usuario SIP, el cual tendra el formato "IMSI..." en OpenBTS
WhiteListFlag timestamp not null default ’0’,
-- true si MS se encuetra en la lista blanca
WhiteListCode varchar(8) not null default ’0’,
-- Codigo de acceso en la lista blanca
rand varchar(33) default ’’,
-- token de autenticación en caché
sres varchar(33) default ’’,
-- token de autenticación en caché
ki varchar(33) default ’’,
-- Ki, clave secreta del SIM
kc varchar(33) default ’’,
-- Kc, clave de cifrado de sesión (aun no es utilizada)
prepaid int(1) default 0 not null,
-- ’1’ para suscriptores de prepago (aun no es soportada)

Figura 4.1: Parámetros de configuración SIPAuthServe.

4.3.4.1 Autenticación

SIPAuthServe soporta autenticación directa SIP en el registro de subscripción (Subscriber Registry) realizando una transacción después de la autenticación de IMS y SIP utilizando:

- RAND
- A3/A8 en lugar de una función hash MD5

SIPAuth soporta dos formas de autenticación RAND-SRES:

Autenticación completa. Es una autenticación que utiliza A3[10] cuando la clave ki[11] es conocida en el registro. OpenBTS incluye COMP128v1[12] como algoritmo A3. El registro invoca a A3 como una aplicación de Unix externa, lo que proporciona una manera fácil de agregar otros algoritmos A3.

Autenticación cache. Este tipo de autenticación utiliza el mismo protocolo en Um como el estándar de autenticación. Pero el comportamiento es diferente y el algoritmo A3 no se utiliza. La autenticación es más débil que la autenticación completa pero puede usarse si la clave ki no es conocida.

SIPAuthServe selecciona automáticamente el tipo de autenticación basado en la disponibilidad de la clave ki. Si la clave ki es conocida utiliza autenticación completa. Si no es así utiliza la autenticación cache. Esta es la autenticación por defecto y la que fue utilizada al configurar OpenBTS.

4.4 Arquitectura

La arquitectura de OpenBTS es una emulación de GSM/GPRS mostrada en la figura 4.2. En la dirección *uplink* los datos del MS son digitalizados en el USRP por un convertidor Analógico-Digital (A/D). Posteriormente son empaquetados con cabeceras *User Datagram Protocol* e *Internet Protocol* (UDP/IP) y enviados a las capas superiores dentro host sobre una interfaz Gigabit Ethernet.

El transceiver se encarga procesamiento de capa física en el USRP. OpenBTS transfiere la información de control sobre la interfaz Ethernet utilizando UDP. En la dirección *downlink* la información es transmitida desde la aplicación pasando por la capa de transporte y de red del sistema operativo. A continuación es procesada por OpenBTS en sus capas SNDCP/LLC/RLC/MAC para finalmente enviarse al USRP por medio de la aplicación transceiver la cual procesara la información para enviarla sobre la interfaz de radio hacia el MS.
OpenBTS Interactúa con Sipauthserver y Asterisk para la administración de registro de los MS. OpenBTS tiene acceso a internet mediante una implementación *Network Address Translation* (NAT) entre la IP del host donde se encuentra corriendo OpenBTS y los MS.

[10] A3. Algoritmo de autenticación utilizado por GSM.
[11] Ki. Llave de autenticación individual de cada subscriptor, con un tamaño de 128 bits.
[12] COMP128v1. Implementación del algoritmo A3.

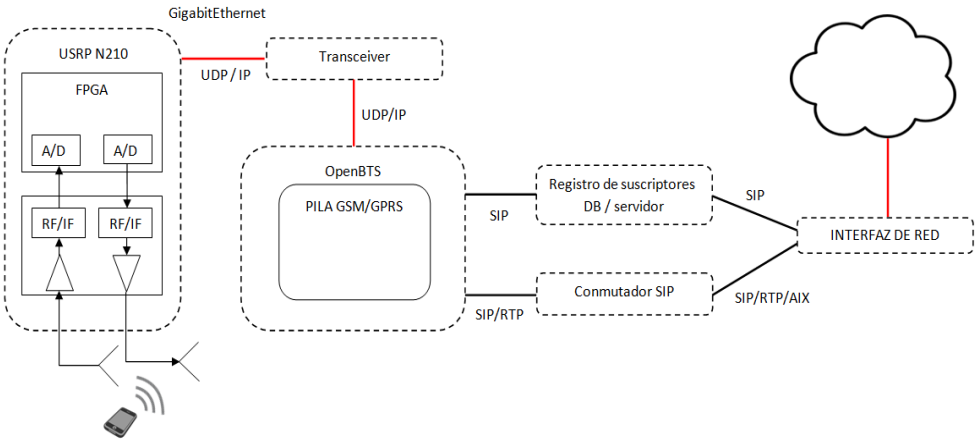

Figura 4.2: Arquitectura OpenBTS.

4.5 Requerimientos de Hardware

El primer requerimiento es un sistema operativo Linux trabajando con un procesador x86, corriendo a 32 bits el cual ha mostrado mejores resultados hasta el momento. Se han realizado pruebas con distribuciones Ubuntu Long-Term Support (LTS), Debian y CentOS.

Se está mejorando el código para que otras arquitecturas en el futuro puedan ser soportadas. Los requerimientos mínimos de *Random-Access Memory* (RAM) y procesamiento no han sido claramente definidos debido a muchas variables involucradas, como el número de portadoras concurrentes, carga en la red, tipo de uso de la red, ambiente de radio entre otras [39].

OpenBTS soporta la creación de múltiples portadoras de forma concurrente, aumentando la capacidad de la red pero la demanda de procesamiento es muy alta. Para una configuración estable con una sola portadora (máximo 7 canales concurrentes), se recomienda un procesador intel i5 junto con 2GB de RAM. Se debe tomar en cuenta que se cuente en la computadora con un puerto red Gigabit Ethernet para un mejor desempeño. Si la computadora cuenta con un puerto de red Fast Ethernet se deberá utilizar un switch que trabaje con Fast y Gigabit Ethernet para que OpenBTS pueda operar, sin embargo el desempeño será menor debido a los retardos en el switch.

4.6 Software Defined Radio

SDR es la llave lo que hace posible la interacción de la pila de protocolos con el radio. OpenBTS soporta el hardware de varias marcas:

- Ettus Research [3]
- Fairwaves [43]
- Nuand [44]
- Range Networks [40].

Para este trabajo se utilizó el equipo USRP N210 de Ettus Research. En la siguiente sección se describe cada una de las partes que conforman este equipo.

4.6.1 Universal Software Radio Peripheral (USRP)

Universal Software Radio Peripheral (USRP) es una familia de productos de hardware manipulado por medio de una PC o laptop. Es una plataforma SDR flexible y de bajo costo desarrollado por Matt Ettus [3]. USRP está compuesto por una tarjeta madre, una tarjeta hija y antenas. La siguiente figura muestra un producto USRP dividido en dos partes; una tarjeta madre con un FPGA para procesamiento de señales y una o más tarjetas hijas las cuales cubren diferentes rangos de frecuencias. La serie USRP cubre todas las frecuencias desde AM pasando por el estándar IEEE 802.11. Los diferentes componentes son descritos a continuación.

4.6.1.1 Motherboard (Tarjeta madre)

La función principal es el muestreo *Intermediate Frequency* (IF) y la conversión entre señales banda base y IF. En la figura 4.3 se muestra el panel frontal del USRP N210.

Figura 4.3: Panel frontal N210.

El equipo cuenta con un conjunto de LEDS que son de gran ayuda para conocer el estatus del dispositivo y son descritos en la tabla 4.1.

LED	descripción
A	En transmisión
B	cable MIMO activado
C	En recepción
D	Firmware cargado
E	Reloj de referencia
F	CPLD (Complex Programmable Logic Device) activo

Tabla 4.1: LEDs del USRP N210.

En la tabla 4.2, se muestran las características de la tarjeta madre del dispositivo.

Frecuencia de muestreo ADC	100 MS/s 14-bits
Frecuencia de muestreo DAC	400 MS/s 16-bits
Rango de frecuencia	SBX 400 - 4400 MHz Rx/Tx
FPGA	Spartan 3A-DSP 3400
Ancho de banda máximo	SBX BW 40MHz
Conexión a PC	Gigabit Ethernet (1000 Mb/s)

Tabla 4.2: Características USRP N210.

4.6.1.2 Daughterboard (Tarjeta hija)

La tarjeta hija es el *front-end* de radio frecuencia. En la mayoría de las tarjetas hijas la señal ya está filtrada, amplificada y sintonizada a la frecuencia de banda base dependiendo en el ancho de banda IF de las tarjetas y su oscilador de frecuencia. Existen también tarjetas de Recepción-transmisión (Rx/Tx) sin conversión de frecuencia o filtrado, las cuales solo proveen conexión RF directa a la tarjeta madre.

La tarjeta utilizada en este trabajo es el modelo SBX [45]. SBX es un transmisor de banda ancha la cual genera 100 mW de salida de potencia y una figura de ruido de 5 dB [46]. Su oscilador para enviar y recibir opera independientemente lo cual proporciona una operación de banda dual. La SBX tiene capacidades MIMO, provee un ancho de banda de 40MHz. Es ideal para aplicaciones que requieren acceso a una gran variedad de de bandas dentro del rango de 400 – 4400 MHz como WiFi, WiMax y GSM. Opera con antenas omnidireccionales con una ganancia de 3 dBi. En la figura 4.4 se muestra la antena y tarjeta hija.

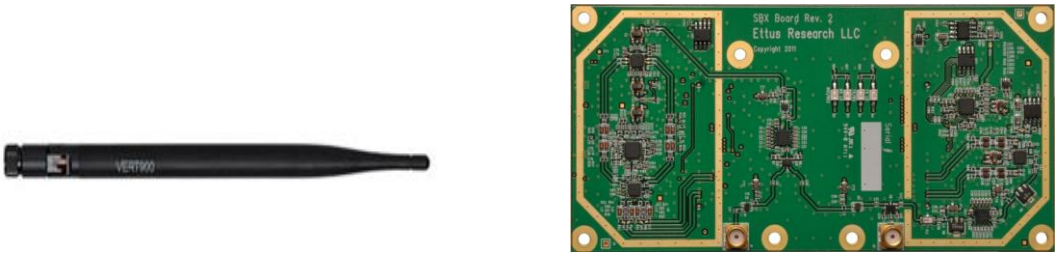

Figura 4.4: Antena VERT900 y Tarjeta Daughterboard SBX.

4.6.1.3 UHD (USRP Hardware Driver)

UHD es el controlador de hardware que provee una API para controlar todos los dispositivos USRP. Trabaja en la mayoría de las plataformas y puede ser compilado con compiladores *GNU Compiler Collection* (GCC) y *Microsoft Visual C++ compiler* (MSVC). Es desarrollado por Ettus Research para el desarrollo de aplicaciones con estos equipos USRP. UHD puede interactuar con plataformas de desarrollo como:

- GNU Radio
- LabVIEW
- Simulink
- OpenBTS
- Iris
- Redhawk
- Amarisoft LTE eNodeB

4.6.2 Teléfonos móviles

Los teléfonos deben ser compatibles con las 4 bandas GSM (850, 900, 1800, o 1900 MHz). Esta información viene en la especificación técnica de cada producto.

Las pruebas pueden hacerse con cualquier *Suscriber Indentify module* (SIM) de cualquier compañía telefónica ya que las pruebas se realizaron con un *Mobile Country Code* (MCC) de 001 y *Mobile Network Code* (MNC) de 01 lo cual permite a cualquier SIM registrarse a la red.

4.7 Configuración de GPRS

El servicio GPRS en OpenBTS está deshabilitado por defecto. Para activarlo se modifica el valor *GPRS.Enable* con valor a 1. Con esto el sistema GPRS advierte a los usuarios por medio del canal *beacon* C0T0[13].

```
OpenBTS> config GPRS.Enable 1
GPRS.Enable changed from "0" to "1"
GPRS.Enable is static; change takes effect on restart
```

Para que tenga efecto se debe ejecutar nuevamente OpenBTS. La aplicación automáticamente iniciara una instancia de la aplicación *transceiver* y se conectara el USRP N210. La información capturada por el radio será intercambiado con el *transceiver* mediante un socket utilizando UDP.

```
$ sudo ./OpenBTS
```

Una vez que se ha reiniciado OpenBTS se puede ejecutar *gprs list* para confirmar que OpenBTS ha configurado algunos canales para GPRS.

```
OpenBTS> gprs list
 PDCH ARFCN=166 TN=1 FER=0%
 PDCH ARFCN=166 TN=2 FER=0%
```

[13] Beacon C0T0. Corresponde al canal de control de difusión enviado en el primer TS del primer canal.

Debajo de la pila IP OpenBTS implementa una interfaz de red lógica [47] en la cual el tráfico de los MS es enviado a través de esta. Esta interfaz puede observarse con la salida del comando *ifconfig*.

```
sgsntun   Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Figura 4.5: interfaz sgsntun.

OpenBTS utiliza la función de NAT del kernel de Linux para asignar IPs a los MS. Estas reglas son aplicadas a la interfaz virtual mediante *iptables* .Estas reglas están localizadas en */etc/OpenBTS/iptables.rules*

Para que las reglas sean aplicadas al sistema operativo se deberá modificar el archivo de configuración de red localizado en */etc/network/interfaces* el cual cargara estas reglas al iniciar el sistema operativo.

```
mark@mark-UX32VD:~$ more /etc/network/interfaces
auto lo
iface lo inet loopback
pre-up iptables-restore < /etc/OpenBTS/iptables.rules
```

El rango de direcciones IP está definido en el parámetro *GGSN.MS.IP.Base* y *GGSN.MS.IP.MaxCount.*

Para que el MS pueda registrarse en la red GPRS debe estar activado el uso de datos en modo *roaming*. Una vez que el MS está exitosamente registrado se puede utilizar el comando *sgsn list* para mostrar información del dispositivo.

```
OpenBTS> sgsn list
 GMM Context: imsi=334020231147002 ptmsi=0xb2001 tlli=0xc00b2001 state=GmmRegisteredNormal age=146 idle=135 MS#1,TLLI=c00b
2001,8007b001 IPs=192.168.99.1
```

Para obtener información de la interacción de los MS con GGSN y SGSN se configura un archivo log el cual se debe definir el OpenBTS.

```
OpenBTS> devconfig GGSN.Logfile.Name /tmp/GGSN.log
GGSN.Logfile.Name changed from "" to "/tmp/GGSN.log"
GGSN.Logfile.Name is static; change takes effect on restart
```

Para monitorear el contenido del archivo log se utiliza el comando tail.

```
$ tail -f /tmp/GGSN.log
 19:44:04.5:  GGSN.MS.IP.Base=192.168.99.1
 19:44:04.5:  GGSN.MS.IP.MaxCount=254
 19:44:04.5:  GGSN.MS.IP.Route=192.168.99.0/24
 19:44:04.5:  GGSN.IP.MaxPacketSize=1520
 19:44:04.5:  GGSN.IP.ReuseTimeout=180
 19:44:04.5:  GGSN.Firewall.Enable=0
 19:44:04.5:  GGSN.IP.TossDuplicatePackets=0
 19:44:04.5:GGSN: DNS servers: 8.8.8.8 0.0.0.0
 19:44:04.5:ip link set sgsntun up
 19:44:04.7:ip route add to 192.168.99.0/24 dev sgsntun
```

En OpenBTS se pueden ajustar el número de PDCHs disponibles para *Traffic Channel* (TCH). El valor por defecto es 2.

```
OpenBTS> config GPRS.Channels.Min.C0 7
GPRS.Channels.Min.C0 changed from "2" to "7"
GPRS.Channels.Min.C0 is static; change takes effect on restart
```

Con el comando *load* se puede observar el total de canales TCH asignados a GPRS.

```
OpenBTS> load
== GSM ==
SDCCH load/available: 4/4
TCH/F load/available: 7/7
PCH load: active, total: 0, 0
AGCH load: active, pending: 0, 0
== GPRS ==
current PDCHs: 0
utilization: 0%
```

El tráfico de voz y datos utiliza TSs que transportan los canales lógicos de tráfico. GPRS puede ajustar el número de TSs utilizados para el tráfico con el parámetro *GPRS.Channels.Min.C0*. Su valor por defecto es 2. Para saber el número de canales disponibles se utiliza en comando *load*. Existen 7 canales de tráfico disponibles los cuales son:

- SDCCH – número de SDCCHs activos disponibles.
- TCH/F - número de TCH/Fs activos disponibles.
- PDCHs -Canales activos GPRS.
- AGCH/PCH número de mensajes encolados esperando a ser transmitidos en AGCH o PCH.
- Utilization % Utilización de canales GPRS.

4.7.1 Configuración Multislot GPRS

Los parámetros *GPRS.Multislot.Max.Uplink* y *GPRS.Multislot.Max.Downlink* controlan el número máximo de slots para un MS en la dirección *downlink* y *uplink*. La configuración *multislot* es limitada por el número de PDCHs adyacentes disponibles en el servicio. Los MSs más modernos soportan hasta 5 slots en algunas combinaciones como 1+4.

Una clase *multislot* más grande proporciona una entrega de datos mayor a un solo MS. Esto no significa que un MS pueda usar los 5 PDCHs concurrentemente, sin embargo la red estará más congestionada al momento que intenten acceder más MSs. Para ajustar este parámetro se modifica la variable *GPRS.Multislot.*

```
OpenBTS> config GPRS.Multislot
GPRS.Multislot.Max.Downlink 3     [default]
GPRS.Multislot.Max.Uplink 2     [default]
```

OpenBTS soporta 2 esquemas de codificación CS-1 y CS-4. Para coberturas más grandes se utiliza CS-1 y para coberturas más pequeñas CS-4. Por defecto OpenBTS soporta ambos.

```
OpenBTS> devconfig Codecs
GPRS.Codecs.Downlink 1,4     [default]
GPRS.Codecs.Uplink 1,4     [default]
```

4.7.2 Asignaciones de canales GPRS

Para obtener el mejor rendimiento de GPRS, el número de canales asignados por ARFCN debe ser un múltiplo de la configuración *multislot*. Si *GPRS.Multislot.Max.Downlink* es 3 entonces los valores óptimos para *GPRS.Channels.Min.C0* son 3 o 6.

4.7.3 Configuración del tamaño de buffer en Linux

Por defecto el buffer TCP en Linux no está configurada para transferencias altas, esto es por el hecho de ahorrar recursos de memoria. Para modificar esta configuración se debe incrementar el tamaño de los *buffers* para soportar una transferencia más alta de paquetes. La memoria TCP es calculada en base a la memoria del sistema y puede consultarse en el archivo /proc/sys/net/ipv4/tcp_mem.

El máximo tamaño del buffer de envió (wmem) y recepción (rmem) es de 12 MB para todos los protocolos. Esto representa la cantidad de memoria que es asignada a cada socket TCP cuando esta es abierta.

```
sudo sysctl -w net.core.rmem_max=50000000
sudo sysctl -w net.core.wmem_max=1048576
```

Capítulo 5: Optimización de GPRS

5.1 Introducción

En este capítulo se discutirá los parámetros TCP que deben ser considerados en el análisis de la red GPRS. Estos parámetros son importantes para comprender el desempeño de la red, y optimizarlos para obtener un mejor desempeño.

Se describe el protocolo TCP y sus 4 mecanismos de control de flujo principales así como los parámetros principales en TCP que deben tomarse en cuenta para su optimización en redes GPRS. Finalmente se discute la clasificación de los enlaces GPRS con respecto a su BDP.

5.2 Transmission Control Protocol (TCP)

TCP fue originalmente diseñado para trabajar sobre enlaces guiados, los cuales proveen una probabilidad baja de error. El mecanismo de evasión de congestión interpreta los paquetes perdidos como congestión en un enlace causando que la tasa de transmisión decaiga. Junto con la alta latencia de enlaces inalámbricos como GPRS, el mecanismo de evasión de congestión de TCP afecta gravemente el *throughput* cuando los paquetes se pierden, debido a los errores de transmisión en los enlaces inalámbricos.

5.2.1 Control de Congestión TCP

El principal objetivo de TCP es evitar una sobre carga en la red que cause rebasar los tamaños de buffer. TCP interpreta paquetes perdidos como una congestión en la red. TCP cubre cuatro grandes componentes.

- Inicio lento (Slow Start)
- Evasión de congestión (Congestion avoidance)
- Retransmisión rápida (Fast retransmisión)
- Recuperación rápida (Fast Recovery)

5.2.2 Inicio lento (Slow Start)

Inicio lento de TCP es mandatorio para evitar paquetes perdidos en redes congestionadas. *Slow start* limita el número de segmentos dependiendo de la Configuración utilizada en GPRS (Número de PDCHs y CS).

5.2.3 Retransmisión y recuperación rápida

Cada segmento fuera de orden causa que se envíen al transmisor *ACKnowledgement*s (ACKs) duplicados. Con el mecanismo de recuperación rápida el transmisor retransmite los segmentos perdidos después de recibir un determinado número de ACKs duplicados en lugar de esperar a que expire el tiempo límite para retransmitir. La retransmisión rápida está ajustada a recibir cuatro ACKs duplicados para lanzar una retransmisión.

El mecanismo de retransmisión es implementada en el mayor de los casos con recuperación rápida, el cual tiene como objetivo evitar el inicio lento cuando solo un segmento está perdido y la congestión de la red no ha sido asumida. El mecanismo de recuperación rápida es un algoritmo para aumentar rápidamente el tamaño de la ventana TCP del transmisor, después de perderse un segmento.

Este algoritmo está basado en la idea que la recepción de ACKs duplicados después de un segmento perdido indica los demás segmentos han sido transmitidos exitosamente por lo tanto los ACKs pendientes por ser recibidos es menor.

5.2.4 ACK Selectivo

El mecanismo de ACK selectivo es de suma importancia para que trabaje el debidamente el mecanismo de retransmisión. ACK selectivo informa al transmisor de todos los segmentos fuera de orden que han sido recibidos de forma exitosa. Por lo que el transmisor sólo tiene que retransmitir los segmentos que no han sido acusados de recibidos. La información de los segmentos que son selectivamente acusados se encuentran en las opciones de campo TCP. Este mecanismo es establecido en el saludo de tres vías de la sesión de control TCP.

5.3 Optimización TCP/IP

El rendimiento de un dispositivo final a otro, requiere de dos parámetros fundamentales: la latencia o *Round-Trip delay Time* (RTT) de los paquetes que son transmitidos de un punto final a otro y el *throughput*. Con la variación de latencia y *throughput* la tasa de paquetes perdidos es de suma importancia en la evaluación de la red GPRS.

Los parámetros que afectan el rendimiento es primero, la configuración de canal de radio y segundo, las características del ancho de banda definido por el número de PDCHs asignados al MS. Finalmente, la eficiencia en la capa RLC determinada por el esquema de codificación usado y *Block Error Rate*[14] (BLER) son los parámetros clave.

[14] Block Error Rate. Medida de tasa de error definido por el número erróneo de bloques.

5.4 TCP Window Size

La ventana TCP es uno de los parámetros más importantes del lado del receptor. El transmisor puede enviar tantos datos como el receptor envié ACKs al transmisor, por cada uno de los segmentos TCP enviados. La ventana TCP debe ser igual al *Bandwidth Delay Product* (BDP) o mayor para poder utilizar todo el ancho de banda disponible [48]. Ventanas demasiado grandes provocaran altos valores RTT debido al encolamiento de paquetes en el buffer y tiempo de procesamiento del *Central Processing Unit* (CPU). En el peor de los casos los segmentos TCP son descartados al rebasar el buffer.

5.5 Bandwidth Delay Product (BDP)

Define un espacio en memoria necesario para una óptima utilización del canal en la red [49]. Esto representa la cantidad de datos que pueden ser enviados al receptor antes de recibir un acuse de recibo. BDP es calculado multiplicado su ancho de banda (BW) con el *Round Trip Time* (RTT). La ecuación 5.1 define el cálculo BDP.

$$\boldsymbol{BDP\ [KB] = BW\ \left[\frac{KB}{s}\right] * RTT\ [s]} \qquad (\mathbf{5.1})$$

5.6 MTU – Maximum Transmission Unit y MSS – Maximum Segment Size

MTU es un parámetro IP que define el máximo tamaño de un paquete de datos dentro de un frame. Cada paquete consiste en el encabezado y segmento de datos. El máximo tamaño del segmento de datos se define como *Maximum Segment Size* (MSS). La ecuación 5.2 define el tamaño del MTU.

$$MTU = IPHeader + TCPHeader + MSS \qquad (5.2)$$

Donde MTU representa la suma del encabezado IP (IPHeader), encabezado TCP (TCPHeader) y el MSS. El tamaño del MSS debe ser considerado al definir la ventana TCP. Mientras más grande es el MTU, la congestión de la ventana crece más rápido [28]. En un enlace con errores, mientras el tamaño del PDU es más pequeño es mucho mejor en términos de transmisión. El protocolo TCP sobre GPRS permite elegir el tamaño del MTU desde 576 hasta 1500 bytes [49].

5.7 Clasificación de enlaces GPRS.

Considerando un MS con 4 PDCHs y un esquema de codificación de CS-4, el máximo ancho de banda que puede lograr es alrededor de los 80 kbps. Los valores típicos de retardo RTT varían entre 600 ms hasta 3s [28], [48], dependiendo de la asignación de los time slots, calidad del enlace y tamaño del N-PDU. El BDP típico cubre un rango de 4 a 20 KB. Basándonos en estas mediciones, los enlaces GPRS pueden ser clasificados como LTN (Long Thin Network) [28]. Estos enlaces reducen el rendimiento del protocolo TCP.

Un transmisor TCP adapta la cantidad de datos basándose en el tamaño de la ventana del receptor. Por lo que una latencia elevada en la red reduce la adaptación dinámica de los algoritmos TCP [28].

Capítulo 6: Análisis del desempeño de GPRS

6.1 Introducción

Este capítulo se presenta un estudio del desempeño TCP y cobertura de la red GPRS utilizando OpenBTS y el equipo USRP N210.

Se describe el escenario utilizado para desplegar la red y la elección de la banda donde se presenta menos interferencia.

Se realiza un análisis en la capa TCP en términos de *throughput* y *delay*, para posteriormente realizar una comparación con los datos teóricos vistos en el capítulo 3. Esto en función del esquema de codificación y configuración *multislot*.

Se realizan mediciones del nivel de señal recibida en el equipo móvil por parte del equipo USRP N210 en función de la distancia, esto para evaluar la cobertura máxima soportada por este equipo.

Los resultados obtenidos proporcionan la base para evaluar esta tecnología como una opción de telefonía celular en zonas rurales del país con escasa cobertura, debido a su bajo costo de instalación y precios accesibles para personas con bajos recursos.

6.2 Implementación del escenario

Para la implementación del escenario se utilizó el siguiente equipo:

- Equipo USRP N210
- Una antena VERT 900
- Laptop ASUS zenbook
- Interfaz USB a Gigabit Ethernet
- Smart Phone Xperia ZL

En la figura 6.1 se describen el escenario utilizado en el desarrollo de esta tesis.

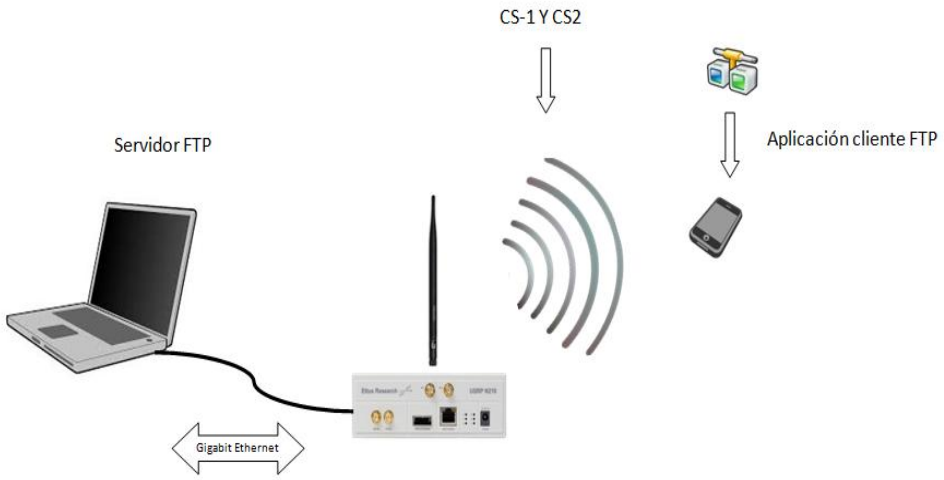

Figura 6.1: Implementación del escenario.

La laptop ASUS tiene instalado un sistema operativo Ubuntu 12.04 LTS de 32 bits. OpenBTS fue instalado en este equipo así como el servidor FTP [51]. La conexión con el equipo USRP N210 se hizo mediante una interfaz Gigabit Ethernet a USB 3.0. Las características del host se pueden ver en la tabla 6.1.

Laptop ASUS	
Procesador	Intel i7 Quad Core
RAM	4GB
Sistema operativo	Ubuntu 12.04 LTS
Versión OpenBTS	5

Tabla 6.1: Características de host.

El dispositivo móvil Xperia ZL utilizó una aplicación cliente FTP llamada *AndFTP* [52], la cual se puede descargar de forma gratuita por medio de *google play*.

El dispositivo SONY Xperia ZL describe en sus especificación una tasa máxima GPRS de 70.4 Kbps en ambas direcciones por lo que no se espera alcanzar mayores tasas que las descritas por estas especificaciones [53].

6.3 Configuración de banda GSM

Para realizar las pruebas, se buscó un espacio libre del espectro para transmitir en cualquiera de las 4 bandas GSM (850, 900,1800 y 1900 MHz) con el objetivo de evitar interferencia por las radio bases telefónicas cercanas. Para ello se utilizó el equipo *Aaronia Spectran HF-4060* [54] mostrado en la figura 6.2.

Figura 6.2: Analizador de espectro Aaronia HF-4060.

Monitoreando la banda de 850 se encontró espacio poco utilizado para transmitir. En la figura 6.3 se muestra el barrido entre 448 a 895 MHz.

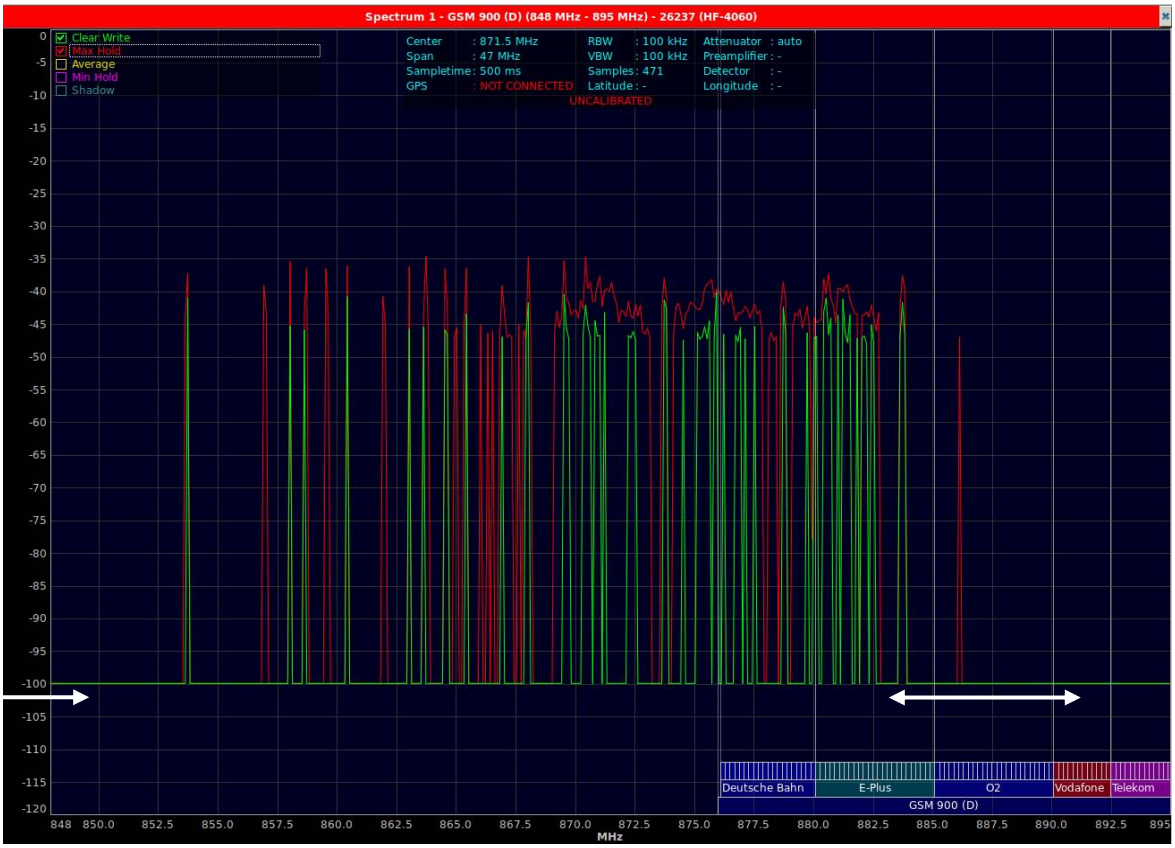

Figura 6.3: Barrido en rango de 848 a 895 MHz.

Como se puede observar, tenemos un espacio de aproximadamente de 4 MHz entre (848 a 852.5 MHz) y de 7 MHz (887 a 895 MHz). Tomando el ARFCN de 251 y calculando el par de frecuencias *uplink* y *downlink* para la banda de 850 tenemos:

$$F_{UL} = 824.2 + 0.2(251 - 128) \tag{6.1}$$

$$\boldsymbol{F_{UL} = 848.8\ MHz}$$

$$F_{DL} = 848.8 + 45$$

$$\boldsymbol{F_{DL} = 893.8\ MHz}$$

Este par de frecuencias fueron las más idóneas para la realización de las pruebas.

6.4 Desempeño RTT con tamaño de paquete IP

Estas mediciones fueron realizadas dentro del laboratorio de redes inalámbricas del edificio Valdez Vallejo.

Las primeras mediciones fueron enfocadas en el parámetro RTT, el cual es crucial para aplicaciones en Internet. Estas mediciones fueron realizadas utilizando el escenario de la figura 6.1 con ambos esquemas de codificación (CS-1 y CS-4). El objetivo es conocer el desempeño de la interfaz de radio GSM utilizando el equipo USRP N210. El tamaño del paquete fue evaluado en un rango de 500 a 1500 bytes debido a que un valor menor a 500 bytes nos proporciona un *throughput* demasiado bajo mientras que mayor a 1500 bytes los retardos son demasiado y el desempeño es muy pobre.

Para realizar las pruebas se creó un script en bash [55], el cual se muestra en la figura 6.4.

```
#!/bin/bash

# icmp message IP header 20 bytes + ICMP header 8 = 28 bytes + packetSize #

if [ -f graphRttPacketSize.txt ];
then
     rm -f graphRttPacketSize.txt
     echo "reinicio del archivo"
fi

for i in `seq 50 150` #rango del tamaño del paquete
do

  packetSize = $(expr $i \* 10)

  icmpPayload = $(expr $packetSize - 28)

  RTT = $(ping -c 5 -s $icmpPayload -i 1 192.168.99.1 | tail -1| awk '{print $4}' | cut -d '/' -f 2)

   echo "$packetSize,$RTT" >> graphRttPacketSize.txt

   echo "$packetSize -- $RTT"

done
```

Figura 6.4: Script para creación de paquetes de diferentes tamaños.

Este script invoca al comando ping el cual tiene como argumentos: número de paquetes *Internet Control Message Protocol* (ICMP), tamaño del paquete (*icmpPayload*), intervalo y dirección IP destino que corresponde al GGSN de OpenBTS. Este bucle produce un tamaño de paquetes IP en un rango de 500 a 1500 bytes y es guardado en la variable *icmpPayload*. Finalmente la salida estándar del comando ping pasa por un conjunto de filtros o tuberías para obtener el valor promedio RTT y es guardado en el archivo *graphRttPacketSize.txt.* Las pruebas fueron realizadas utilizando la configuración de la tabla 6.2.

PDCH Downlink	3
PDCH Uplink	2
Tamaño del paquete (bytes)	500 - 1500
Esquema de codificación	CS-1,CS-2

Tabla 6.2: Parámetros OpenBTS.

En la figura 6.5, se muestra el comportamiento RTT[15] de la interfaz de radio GSM, con respecto al tamaño del paquete IP con CS-1 y CS-2. Como se observa, el valor RTT sufre con muchas fluctuaciones, debido al tamaño del paquete IP y los retardos de procesamiento en el equipo USRP N210. A pesar de las fluctuaciones se puede observar un crecimiento del RTT en ambos esquemas de codificación a medida que se incrementa su tamaño. Cada valor en la gráfica representa el promedio de 10 pruebas realizadas.

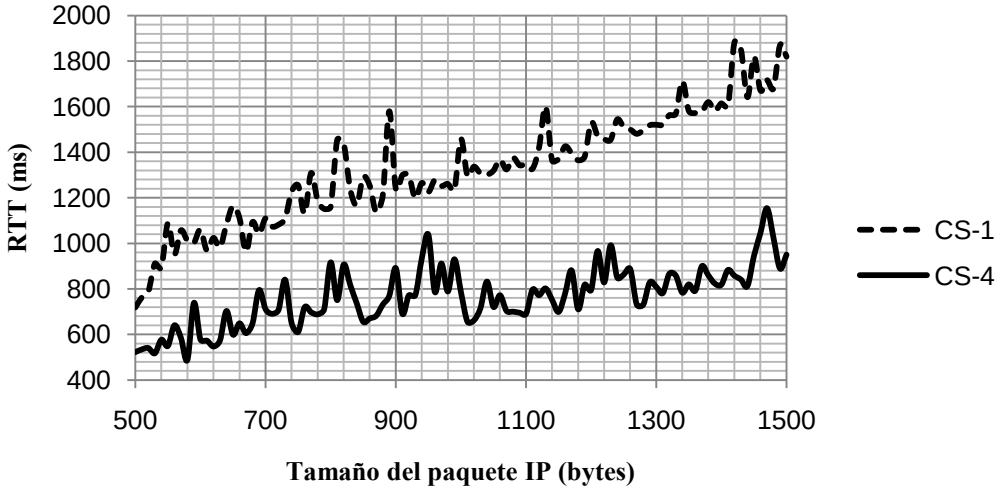

Figura 6.5: RTT vs tamaño del paquete IP.

[15] El valor RTT representa el tiempo que tarda un mensaje de recibido en llegar al teléfono.

6.5 Optimización del MTU (Maximum Transfer Unit)

Para realizar las pruebas de descarga hicimos uso del servidor FTP junto con la aplicación cliente *AndFTP*. A continuación se describe el protocolo FTP.

6.5.1 FTP (File Transfer Protocol)

File Transfer Protocol (FTP) proporciona garantía en la transferencia de datos en una arquitectura cliente-servidor. Este protocolo está estandarizado en [56] y es uno de los más utilizados debido a su control del lado del cliente, haciéndolo una buena aplicación para la evaluación de redes inalámbricas.

FTP utiliza una sesión de control en donde se realiza un intercambio de comandos basados en el protocolo *telnet*. La sesión de control usa el puerto 21 por defecto. Para la transferencia de datos, FTP utiliza una conexión full dúplex entre el cliente y servidor utilizando el puerto 20. FTP es eficiente dentro de una red GPRS ya que solo necesita una sola conexión para transferir datos y con ello una tamaño de encabezado pequeño.

Antes y después de realizar la transferencia de datos se requiere de un conjunto de mensajes de control para que la sesión sea exitosa. Se debe tomar en cuenta que al intercambiar mensajes de control el *throughput* decrece. En la figura 6.6 se muestra el intercambio de mensajes entre el cliente y el servidor para realizar una transferencia de un archivo.

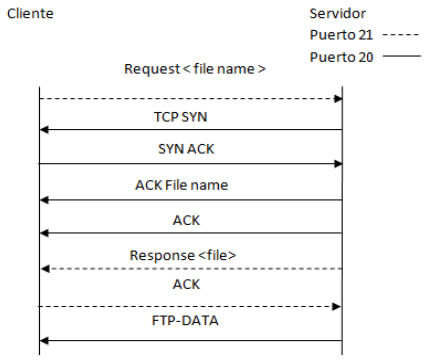

Figura 6.6: Establecimiento de conexión FTP.

6.5.2 Optimización del MTU

OpenBTS ajusta por defecto un MTU en la interfaz *sgsntun* de 1500 bytes como se observa en la figura 6.7. En esta sección se analizó el desempeño utilizando el MTU por defecto.

```
sgsntun   Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Figura 6.7: MTU de 1500 bytes utilizado por defecto en OpenBTS

Para realizar las pruebas se utilizó la configuración *multislot* por defecto (3+2), la cual se puede observar en la figura 6.8, utilizando CS-4.

```
OpenBTS> config GPRS.Multislot
GPRS.Multislot.Max.Downlink 3     [default]
GPRS.Multislot.Max.Uplink 2     [default]
```

Figura 6.8: Configuración *multislot* por defecto.

En la figura 6.9, se muestra la descarga de un archivo de 545 KB con la configuración descrita anteriormente. El retardo promedio RTT de CS-1 y CS-4 se encuentra en el eje vertical, mientras que el eje horizontal corresponde al tiempo transcurrido de la descarga.

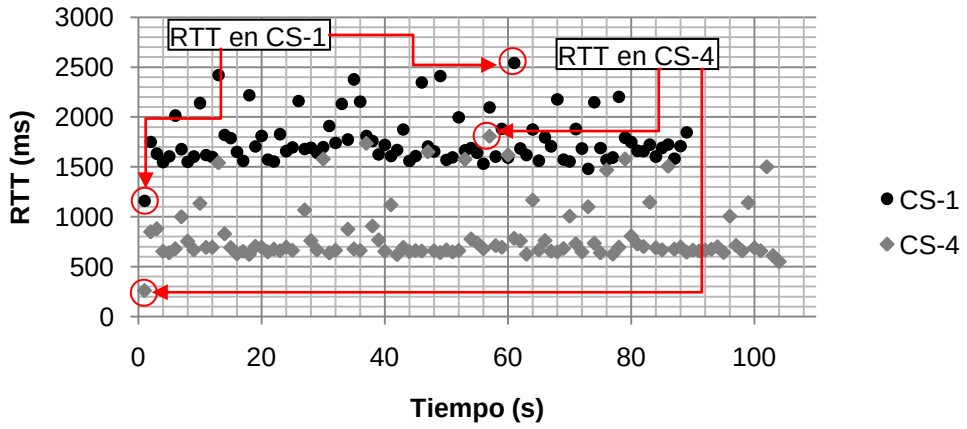

Figura 6.9: MTU 1500 y 3 PDCHs.

Como se puede observar los retados varían drásticamente de 250 ms a 1.82 s en CS-4 mientras que en CS-1 es de 1.2 s a 2.5 s. TCP es tolerante a retardos mientras estos no varíen demasiado. Debido a los cambios drásticos de RTT, la capa TCP retransmite varios segmentos al expirar el tiempo de espera de los ACKs. En la figura 6.10 se muestra el porcentaje de retransmisiones para ambos esquemas de codificación.

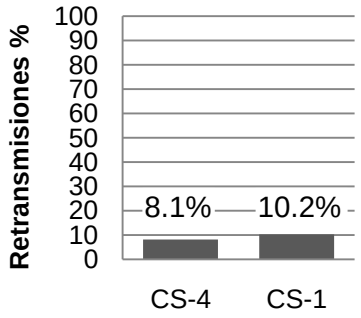

Figura 6.10: Porcentaje de retransmisiones

Para disminuir los retardos se modificó el tamaño del MTU basándonos en la figura 6.5. Realizando varias pruebas con diferentes tamaños de MTU se ajustó a un valor de 788 bytes el cual es obtenido mediante la ecuación 6.2.

$$MTU = IPHeader + TCPHeader + MSS \quad (6.2)$$

$$MTU = 20 + 32 + 736$$

$$MTU = 788 \ bytes$$

El MTU fue reducido de 1500 a 788 bytes. Con este ajuste disminuyen considerablemente los retardos y se establecen en 480 ms y 1.1 s para CS-4 y CS-1 respectivamente. En la figura 6.11 se observa la descarga del mismo archivo utilizando CS-1 y CS-4. El retardo es mayor en CS-1 debido a que por cada segmento debe enviar más *radio blocks,* ya que transportan la mitad de datos en comparación con CS-4. Esto provoca mayores retardos al enviar los segmentes TCP. El tiempo de descarga en CS-4 es de 90 s, mientras que con CS-1es de 220 s.

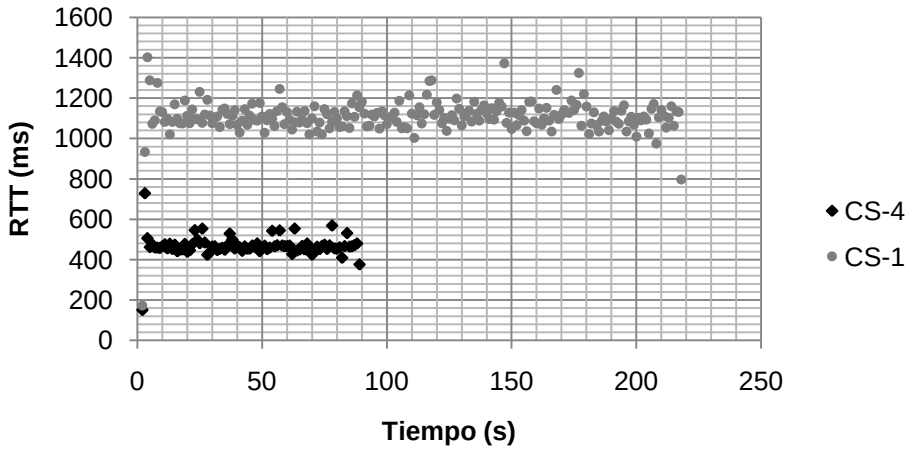

Figura 6.11: MTU de 788 bytes con 3 PDCHs.

Ajustando el tamaño del MTU a 788 bytes evitamos todas las retransmisiones. Este valor fue utilizado en todas las pruebas para evaluar el *throughput* en *downlink* en la siguiente sección.

6.6 Throughput en downlink utilizando multislot

Otro parámetro importante es el *throughput* en *downlink* en función a su configuración *multislot.* Medir estos valores y compararlos con teóricos nos permite conocer la mejor configuración del sistema.

Para realizar estas mediciones, se descargó un archivo de 545 KB del servidor FTP usando diferentes configuraciones *multislot* en combinación con los esquemas de codificación. El tamaño del MTU fue ajustado a 788 bytes por darnos un mejor desempeño. El *throughput* se midió con la aplicación *AndFTP* instalada en el dispositivo móvil. En la tabla 6.3 se describen las características de esta medición.

Window Size	3776
MTU	788 bytes
Tamaño del archivo	545kB
Time slots en *uplink*	2
codificación	CS-1 / CS-2

Tabla 6.3: Características en *downlink*.

A continuación se ilustra en la tabla 6.4 los resultados de la descarga FTP. Las mediciones se realizaron incrementando de 1 hasta 4 PDCHs en *downlink*, en un espacio cerrado a una distancia máxima de 5 metros.

CS-4				
Número de canales	1 PDCH	2 PDCHs	3 PDCHs	4 PDCHs
Duración (min)	4:11	2:07	1:27	1:20
Throughput (Kbps)	17.3	34.3	50.1	54.5
CS-1				
Número de canales	1 PDCH	2 PDCHs	3 PDCHs	4 PDCHs
Duración (min)	10:05	5:10	3:20	3:02
Throughput (Kbps)	7.2	14	21.8	23.9

Tabla 6.4: *Throughput* con diferente número de time slots

En la tabla 6.4 se observa que estos datos son obtenidos a partir del tamaño del archivo dividido por el tiempo de su descarga. En la figura 6.11, se muestra únicamente el *throughput* alcanzado al incrementar los PDCHs (TSs).

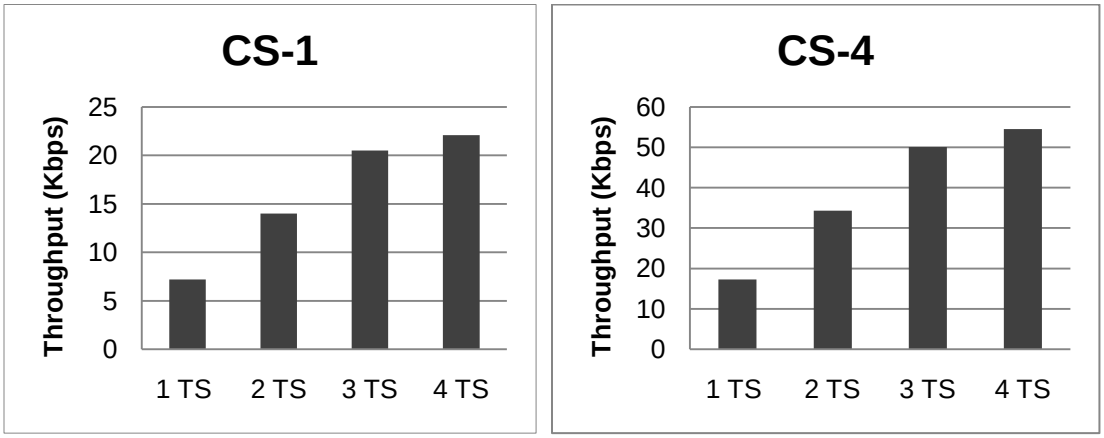

Figura 6.12: *Throughput* con diferente configuración *multislo*t.

En la figura 6.13 y 6.14 se muestran gráficas donde se observa el *throughput* de 3 y 4 PDCHs con CS-4 y CS-1 respectivamente. En CS-4 hay un incremento de 4.4 Kbps, mientras que para CS-1 es de 2.1 Kbps. Para obtener estas mediciones utilizamos wireshark [57].

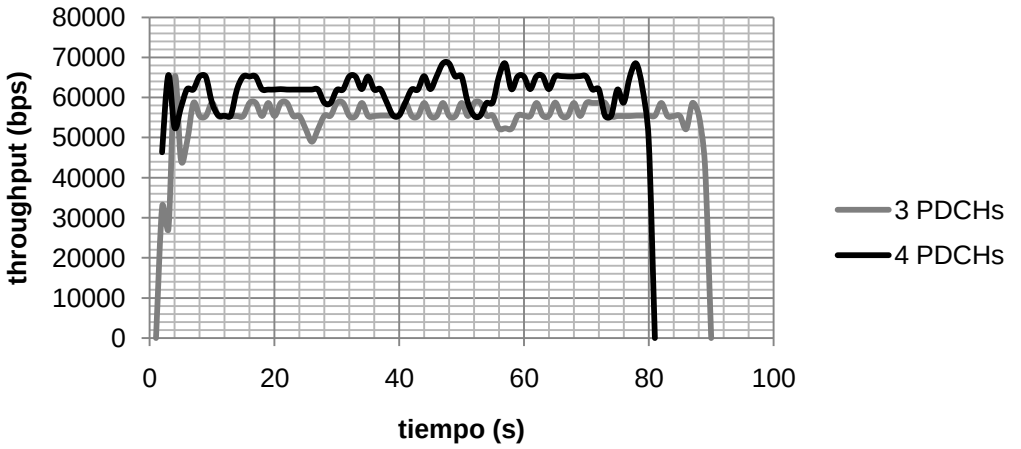

Figura 6.13: Throughput 3 PDCHs vs 4 PDCHs con CS-4.

Los resultados muestran que OpenBTS tiene un mejor desempeño utilizando la configuración 3+2.

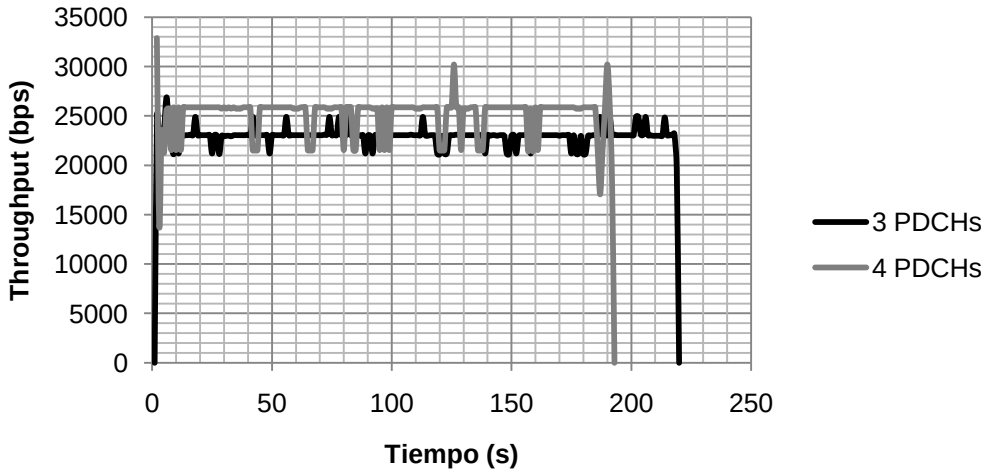

Figura 6.14: Throughput con 3 PDCHs vs PDCHs con CS-1.

6.7 Análisis en la ventana de recepción TCP

Para estas pruebas utilizamos un equipo Xperia ZL, el cual utiliza android como sistema operativo. El equipo cuenta con el mecanismo Dynamic Right-Sizing (DRS), el cual deja al receptor ajustar su ventana TCP de forma dinámica en función de la satisfacción de la demanda en la conexión. Basándose en el *Bandwith Delay Product* (BDP) medido en la red. El objetivo de DRS es asignar suficiente espacio en memoria en la ventana TCP, por lo que el *throughput* de la conexión no está limitado por este valor sino únicamente por la congestión de la red, así DRS evita asignar más memoria de lo necesario [58]. Para utilizar de forma eficiente en canal, la ventana es calculada a partir del BDP con la siguiente ecuación.

$$Windowsize[KB] = BW\ \left[\frac{KB}{s}\right] * RTT\ [s] \tag{6.3}$$

El RTT promedio con CS-4 es de 467 ms, mientras que con CS-1 es de 1.12 s. Tomando estos datos y el uso 3 PDCH obtenemos la ventana óptima para ambos.

$$Windowsize_{CS-4} = \left(\frac{21.4\ kbps * 3\ PDCHs}{8}\right) * 0.467\ s \tag{6.4}$$

$$Windowsize_{CS-4} = 8.025\ KB * 0.467\ s$$

$$\boldsymbol{Windowsize_{CS-4} = 3.748\ KB}$$

$$Windowsize_{CS-1} = \left(\frac{9.05\ kbps * 3\ PDCHs}{8}\right) * 0.1.12\ s \tag{6.5}$$

$$Windowsize_{CS-1} = 3.393\ KB * 1.12\ s$$

$$Windowsize_{CS-1} = \boldsymbol{3.8\ KB}$$

Como podemos observar, en la ecuación 6.4 y 6.5, el valor de la ventana TCP en ambos casos es casi igual (3.748 KB y 3.8 KB). La figura 6.15 fue extraída del monitoreo con wireshark de una descarga con FTP utilizando CS-4.

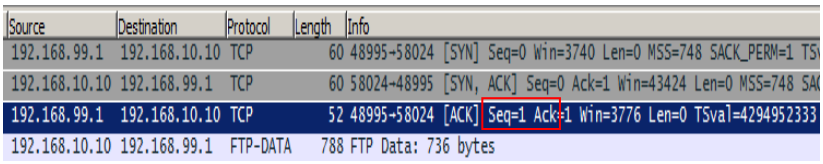

Figura 6.15: Ventana de recepción del MS.

Como se observar en la figura 6.15, la aplicación cliente FTP con dirección 192.168.99.1 advierte su ventana de 3776 bytes al servidor FTP con dirección 192.168.10.10. Cabe mencionar que el valor para CS-1 fue el mismo que en CS-4.

6.8 Cobertura del USRP N210

Otro aspecto importante para llevar esta tecnología a lugares rurales es el análisis de la cobertura de los equipos USRP N210.

El objetivo en esta sección es medir la cobertura máxima con respecto al Eb/N0 registrado del lado del receptor. Estos valores serán comparados con el estudio realizado en el artículo [59], y se determinará los esquemas de codificación que se pueden ofrecer a ciertas distancias alrededor del USRP N210.

Antes de realizar las mediciones se hicieron ajustes en OpenBTS en el control de potencia, la cual se describe a continuación.

6.8.1 Ajustes de potencia

Para realizar las pruebas se cambiaron parámetros en el control de potencia, ya que con la configuración por defecto a una distancia mayor de 5 m entre el USRP N210 y el MS la conexión fallaba debido a la baja potencia del MS. A continuación se define el concepto y su configuración.

6.8.1.1 Control de potencia

El control de potencia se refiere al control de transmisión de potencia del MS y BTS en la interfaz de radio. En la dirección *uplink* permite a los MSs transmitir a niveles bajos de potencia cuando no es necesario niveles altos en la transmisión, aumentando el tiempo de vida de la batería. En la dirección *downlink* una potencia baja de transmisión permite a la BTS reducir el consumo de energía y con ello el costo de operación.

6.8.1.2 Control de potencia Uplink

Los procedimientos de control de potencia son utilizados para calcular la potencia de transmisión de salida P_T de cada asignación de PDCH en la dirección uplink basado en la siguiente ecuación [9].

$$P_T = \min(\gamma_0 - \gamma_{CH} - \alpha * (C + 48), P_{MAX}) \quad (6.6)$$

- $\boldsymbol{P_T}$ Potencia de transmisión en dBm.
- $\boldsymbol{\gamma_0}$ Es un valor constante de 39 dBm en las bandas bajas (850 y 900 MHz) y 36 dBm en las bandas altas (1800 y 1900 MHz).
- $\boldsymbol{\gamma_{CH}}$ Este parámetro determina la mínima salida de potencia del MS por medio de un valor de 0 al 64 dB.
- $\boldsymbol{\alpha}$ Es un valor escalar que determina el cambio de salida de potencia con respecto al valor *C.* Su rango es de 0.1 a 1.
- $\boldsymbol{C}$ Es el valor RSL en la dirección *downlink* medido por el MS.
- $\boldsymbol{P_{MAX}}$ Máxima potencia permitida del MS. El rango varía de 5 a 33 dB [60]. Sin embargo los dispositivos, trabajan con una potencia mucho más baja que va de 3.2 mW a 20 mW.

6.8.1.3 Configuración del control de potencia en OpenBTS

OpenBTS implementa *Open Loop Power Control* el cual está basado en la pérdida de la señal *downlink,* asumiendo que esa misma pérdida sucede en *uplink* por lo que el parámetro $\boldsymbol{\alpha}$ es ajustado a 1. El parámetro $\boldsymbol{\gamma_0}$ se mantiene constante ya que la radio base no actualiza este valor. Con esto cuando la señal recibida C decrece, la potencia del salida del MS aumenta y viceversa.

Los parámetros $\boldsymbol{\alpha}$ y $\boldsymbol{\gamma_0}$ en OpenBTS se muestran a continuación.

GPRS.MS.Power.Alpha
GPRS.MS.Power.Gamma

GPRS.MS.Power.Alpha se configura dentro de un rango de 1 a 10, cada valor representa en valor escalar de 0.1 por lo que el rango es de 0.1 a 1.
GPRS.MS.Power.Gamma se configura con un valor de 0 a 32. Cada valor representa un diferencia 2 dB por lo que los valores varían entre 0 a 64 dB.

Los parámetros de configuración $\boldsymbol{\alpha = 1}$ y $\boldsymbol{\gamma_0 = 62}$ por defecto en OpenBTS fueron ajustados con $\boldsymbol{\alpha = 1}$ y $\boldsymbol{\gamma_0 = 0}$ obteniendo así la máxima potencia de transmisión del MS.

6.8.2 Selección del área geográfica

Las pruebas fueron realizadas enfrente del anexo de ingeniería de Ciudad Universitaria. En la figura 6.15 se observa el área geográfica, esta zona fue escogida debido a la poca obstrucción de árboles.

Figura 6.16: Área seleccionada para realizar las mediciones.

El equipo fue instalado en un extremo. La descarga del archivo inició a menos de un metro del equipo USRP N210. Iniciando la descarga se procedió a caminar en línea recta alejándonos del equipo hasta que la descarga presentara problemas de conexión y fallara.

6.8.3 Mediciones de cobertura

Para servicios de datos el *bit energy to noise density* (E0/N0) y *BLock Error Rate* (BLER) son factores determinantes para la detección de la señal recibida.

Un E_b/N_0 mínimo para un BLER dado es especificado en cada tipo de modulación para una recepción exitosa. Un error de bloque ocurre cuando uno o más bits de un bloque codificado (1 bloque = 4 burts) resulta erróneo después de su decodificación. El BLER para un PDCH es de 10% a 30% [25]. En la tabla 6.5 se muestra los valores mínimos de E_b/N_0 con CS-1 y CS-4 obtenidos en el artículo [58].

Codificación	Eb/N0 [dB]
CS-1	14
CS-4	35

Tabla 6.5: Eb/N0 mínimo requerido pasa CS-1 y CS-4

6.8.3.1 SNR (Signal to noise ratio)

SNR representa la relación de la fuerza de la señal que se transmite con respecto a la potencia de ruido que deteriora la señal. A medida que la distancia entre el USRP N210 y el MS aumenta el SNR disminuirá, por lo que las mediciones de SNR representan factor de la máxima cobertura [61]. Para calcular los valores SNR a partir del RSL medido, utilizamos la siguiente ecuación.

$$SNR = RSL[dB] + N[dBm] \tag{6.7}$$

Donde:

- RSL (Received Signal Level) representa la potencia de recepción medida en dBm y está dada por:

$$RSL[dBm] = EIRP[dBm] - PL[dB] + G_{rx}[dB] \tag{6.8}$$

  - $EIRP[dBm]$: Potencia Isotrópica radiada efectiva (*Effective Isotropic Radiated Power*).
  - $PL[dB]$: Pérdida de trayectoria.
  - $G_{rx}[dB]$: Ganancia de la antena receptora.

- N representa la potencia de ruido de recepción determinada por la ecuación.

$$N[dBm] = 10\log(k * T_0 * B_w) + F_N + 30 \tag{6.9}$$

  - k: Constante de Boltzman (1.38 x 10^{-23} J/K).
  - T_0: Temperatura ambiente en grados Kelvin (290º K).
  - B_w: Ancho de banda (Hz).
  - F_N: Figura de ruido en el receptor (dB).

Calculando N para GPRS tenemos:

$$N[dBm] = 10\log\left(1.38 \text{ x } 10^{-23}\ \left[\frac{\text{J}}{\text{K}}\right] * 290[K] * 200[kHz]\right) + 5[dB] + 30 \qquad (6.10)$$

$$N[dBm] = -150.966 + 5[dB] + 30$$

$$\boldsymbol{N = -115.966\ [dBm]}$$

6.8.3.2 E_b/N_0 (Energy per bit to Noise power spectral density ratio)

E_b/N_0 (*Energy per bit to Noise power spectral density ratio*) representa el SNR por bit [61] y puede ser determinado a partir del SNR utilizando la siguiente ecuación.

$$SNR = \frac{R_b}{B} * \frac{E_b}{N_0} \qquad (6.11)$$

Donde:

- R_b Data rate de la modulación
- B Ancho de banda del canal
- E_b Energía por bit
- N_0 Representa la densidad de ruido

GSM utiliza el esquema de modulación GMSK, su *data rate* R_b es de 270.833 kbps y su ancho de banda de 200 kHz. Calculando R_b/B tenemos.

$$\frac{R_b}{B} = \frac{270.833\ [kbps]}{200\ [kHz]} = 1.35\ [dB] \qquad (6.12)$$

Este resultado significa que $E_b/N0$ es 1.35 dB menos con respecto al valor SNR medido. A continuación se observa los valores RSL medidos en la figura 6.16. Estas mediciones fueron tomadas con la aplicación *GSM signal monitoring* [62].

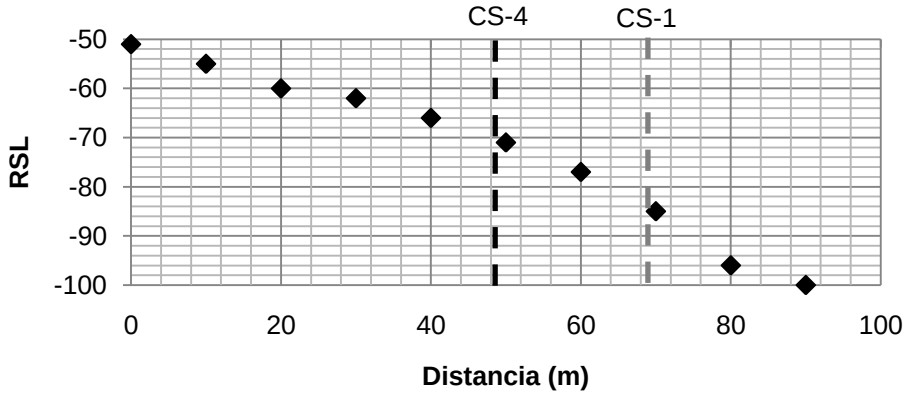

Figura 6.17: RSL vs Distancia.

En la gráfica 6.17 se muestra el RSL medido con respecto a la distancia. Para CS-4 la distancia máxima alcanzada fue de 60 m, con un valor RSL de -77 dBm, mientras que para CS-1 la cobertura fue de 80 m, con un valor RSL de -96 dBm. A continuación en la tabla 6.6 se presenta el valor SNR y E_b/N_0 calculado a partir del RSL medido. Para el valor de -77 dBm de CS-4 su valor E_b/N_0 es de 37.65 dB y para CS-1 es de 18.65 dB.

RSL	SNR	E_b/N_0
-51	65	63.65
-55	61	59.65
-60	56	54.65
-62	54	52.65
-66	50	48.65
-71	45	43.65
-77	39	37.65
-85	31	29.65
-93	23	21.65
-96	20	18.65
-100	16	14.65

Tabla 6.6: SNR y E_b/N_0 a partir del RSL medido.

En la tabla 6.7 se muestran los valores medidos con respecto a los teóricos. La comparación de los resultados prácticos con los teóricos muestra que la implementación en OpenBTS al estudio realizado en [59].

Codificación	Eb/N0 dB teórico	Eb/N0 dB teórico práctico
CS-1	14	18.65
CS-4	35	37.65

Tabla 6.7: Eb/N0 dB teórico y práctico.

6.8.4 Throughput

En esta sección se presenta las mediciones de *throughput* de CS-1 y CS-4 obtenido con respecto a la distancia.

En la figura 6.18, se muestra el comportamiento de CS-4, éste se mantiene estable hasta los 60 m donde la conexión comienza a tener fallas, alejándose de 3 a 4 metros la conexión se cierra. Como se observa, el *throughput* decae debido a las retransmisiones.

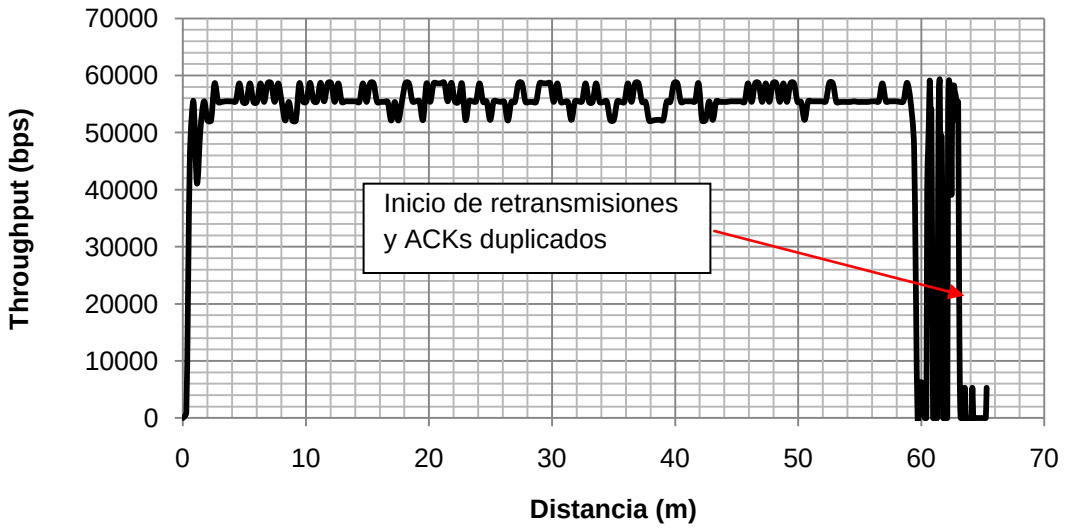

Figura 6.18: Throughput VS Distancia con CS-4.

Las mediciones con CS-1 presentan un comportamiento similar con respecto a CS-4. En la figura 6.19 se muestra como el *throughput* comienza a decaer a partir de los 80 metros. Al alejar el MS 5 metros más, la conexión falla y se cierra.

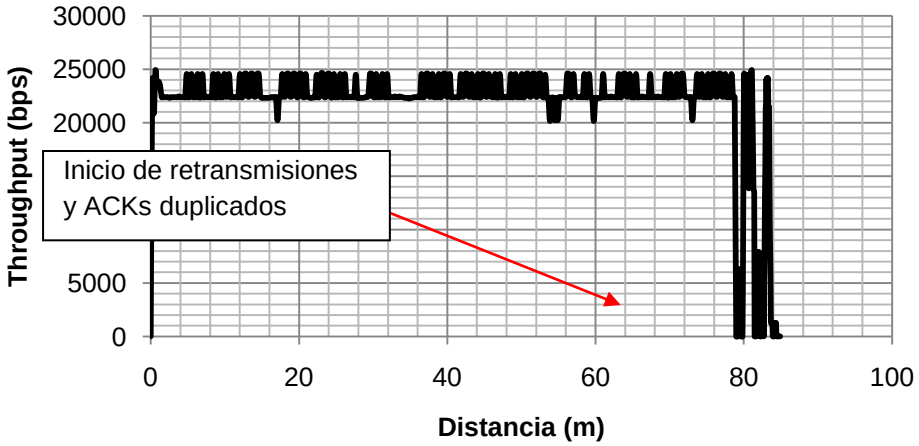

Figura 6.19: Throughput VS Distancia con CS-1.

6.8.5 RTT

A continuación se presenta el RTT de ambos esquemas de codificación. En la figura 6.19 utilizando CS-4 el aumento del RTT se dispara a los 60 m mientras que para CS-1 en la figura 6.20 es a partir de 80 m.

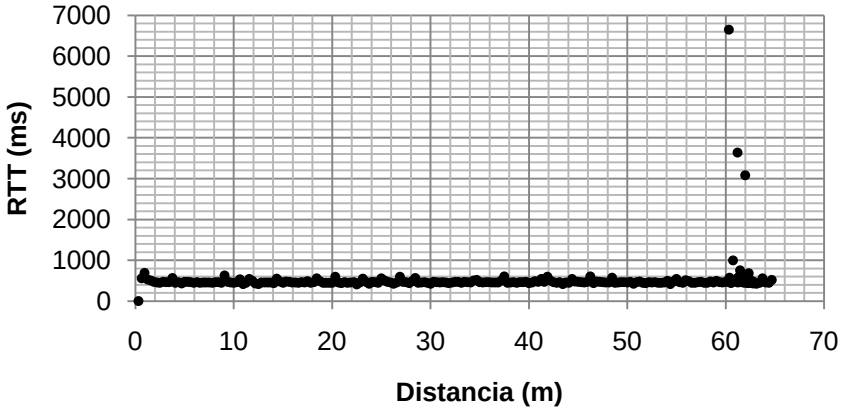

Figura 6.20: RTT VS Distancia utilizando CS-4.

El aumento del RTT es debido al bajo Eb/N_0 en el receptor, provocadas por las condiciones del canal, estos altos retardos ocasionan que espire el tiempo de retransmisiones TCP *Retransmission TimeOut* (RTO). Si los tiempos de retransmisión expiran, TCP no sabe distinguir si un paquete ha sido perdido o si su ACK correspondiente únicamente sufrió de un elevado RTT. Por lo que el rendimiento TCP se degrada al retransmitir segmentos.

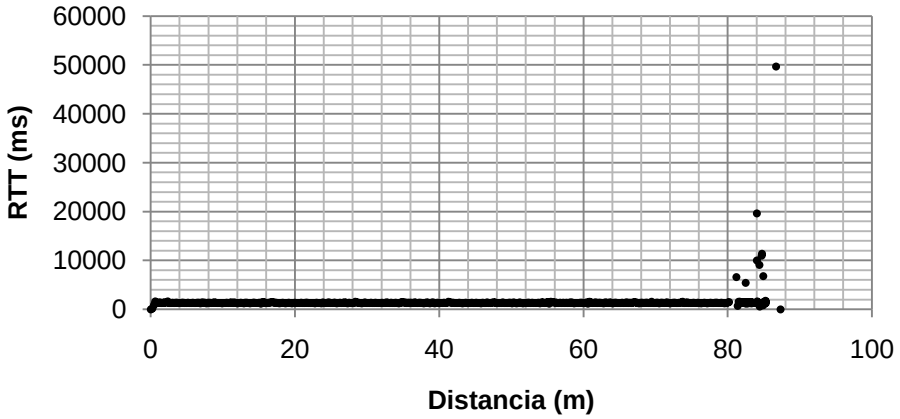

Figura 6.21: RTT VS Distancia utilizando CS-1.

Para ambos esquemas de codificación se observa que el throughput se degrada conforme las retransmisiones y ACKs duplicados. Esto al alcanzar la distancia máxima, los ACKs y ACKs duplicados son transmitidos por el receptor.

6.8.6 Retransmisiones

TCP maneja la pérdida de paquetes por medio de un mecanismo de retransmisión de paquetes. Estas retransmisiones suceden debido a las malas condiciones del canal. La pérdida de paquetes se refleja en la disminución de *throughput* y con ello se alarga el tiempo de transferencia debido a las retransmisiones.

En la figura 6.22 se muestra el comportamiento descrito en el párrafo anterior, donde se presentan las retransmisiones y ACKs duplicados enviados a partir de una distancia entre 60 m a 65 m utilizando CS-4.

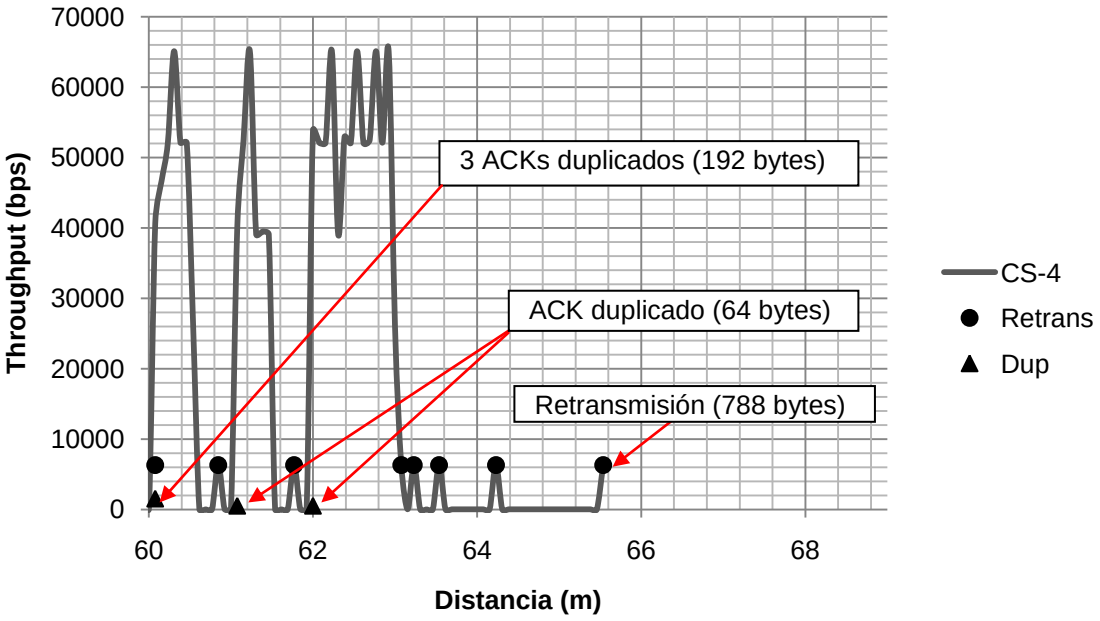

Figura 6.22: Retransmisiones y ACKs duplicados con CS-4.

El tamaño de los paquetes ACK duplicados son de 64 bytes o 512 bits mientras que las retransmisiones tienen un tamaño de 788 o 6304 bits. En la figura 6.22 se observa el tiempo y tamaño de estos mensajes utilizando wireshark.

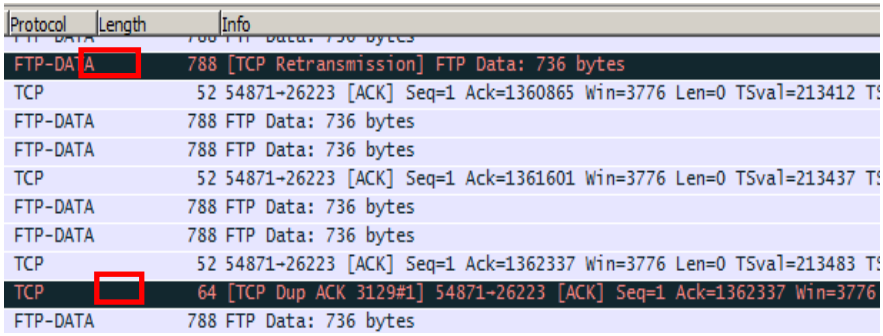

Figura 6.23: Tamaño de paquetes ACK duplicados y retransmisiones.

En la figura 6.24 se presenta en caso para CS-1 donde a partir de los 80 m se inician las retransmisiones y ACKs duplicados. La tolerancia a un bajo Eb/N0 permite una cobertura de 20 m más, con respecto a CS-4.

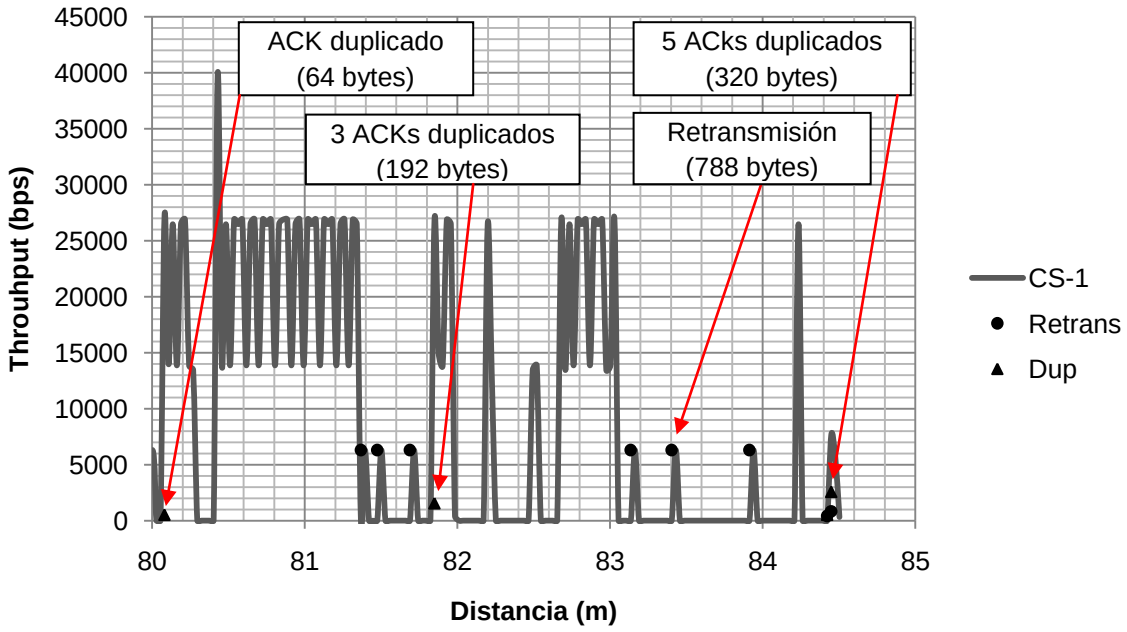

Figura 6.24: Retransmisiones y ACKs duplicados con CS-1.

Capítulo 7: Conclusiones

7.1 Introducción

En este capítulo, se presenta las contribuciones derivadas de este trabajo de investigación así como trabajos futuros derivados de este y conclusiones finales.

7.2 Contribución

El presente trabajo de investigación realizó un estudio del desempeño de una red GPRS utilizando OpenBTS versión 5 y un equipo USRP N210. Se describieron los requerimientos de hardware y el software necesario para su implementación y la configuración para que la red esté en operación.

Este trabajo abre una nueva línea de investigación en el desarrollo de telecomunicaciones utilizando tecnología SDR. Los resultados obtenidos nos proporcionan un panorama del potencial de esta tecnología y del hardware utilizado.

Este trabajo nos proporciona nuevos conocimientos que servirán para la enseñanza a las futuras generaciones de ingenieros en telecomunicaciones para poner en práctica sus conocimientos teóricos así como en futuras implementaciones de otras tecnologías inalámbricas.

7.3 Trabajo futuro

La implementación de GPRS no es perfecta todavía, voz y mensajes de textos están a punto de ser terminados sin embargo GPRS continua teniendo mejoras en su optimización de algoritmos de acceso al medio y calendarización de los recursos.

Los resultados de este trabajo nos proporcionan un panorama inicial para realizar estudios en zonas urbanas, donde el tipo de terreno presenta mayores problemas de propagación de la señal y se deberá contar con antenas más sofisticadas para un despliegue exitoso.

El número de usuarios soportados por el equipo USRP N210 es un estudio abierto, y el cual es de suma importancia para el análisis de viabilidad en zonas rurales.

El entendimiento de GNU Radio en este tipo de proyectos proporciona las bases para trabajos futuros en la implementación de radios en otras tecnologías como UMTS y LTE.

Proyectos con OpenBTS como el análisis en el comportamiento de *handoff* entre dos o más equipos USRP requieren de un estudio minucioso para desplegar más de una celda.

En octubre de 2014 el proyecto OpenBTS libero la versión 1.0 de UMTS la cual cuenta hasta el momento con manejo de datos. Esto nos motiva a continuar haciendo investigaciones ahora con este nuevo software y aportar mejoras a estas tecnologías.

La investigación en nuevas plataformas de hardware SDR para implementación de radios como Rasperry Pi basada en Linux, es una de las siguientes plataformas en programación de radios. Range Networks está trabajando con su software OpenBTS y Rasperry Pi en la implementación de nuevas plataformas en telefónica celular.

OpenBTS permitirá aportar conocimiento práctico a la docencia, donde los estudiantes en telecomunicaciones podrán interactuar y comprender estas tecnologías en laboratorios.

7.4 Conclusiones finales

Se realizaron varias mediciones con diferente configuración *multislot* y se llegó a la conclusión que OpenBTS proporciona un *throughput* más cercano al teórico con la configuración *multislot* 3+2. Con el uso de más PDCHs el *throughput* muestra ligeros aumentos por arriba de los 64 Kbps utilizando CS-4.

Se realizó un análisis de desempeño en la capa TCP donde la configuración por defecto de GPRS sufre de altas variaciones de retardo entre 400 ms y 2 s en la interfaz de radio desplegada por el USRP N210. Para solucionar este problema se realizó un análisis del comportamiento de los retardos con respecto al tamaño del paquete donde observamos que un tamaño de MTU de 788 bytes nos proporciona menores retardos comparado con un tamaño 1500 bytes con el cual GPRS trabaja por defecto.

Las mediciones de distancia de propagación de la señal con el equipo USRP N210 mostraron que la máxima distancia que puede abarcar este equipo es de 80 m utilizando CS-1 y 60 m utilizando CS-4, Esto se logra configurando los parámetros de control de potencia en *uplink* ya que al dejar la configuración por defecto la potencia del MS es baja y tiene un alcance únicamente de 5 m de distancia entre el MS y el USRP N210.

Los resultados mostraron que las variaciones de los retardos se disparan al alcanzar la máxima distancia, lo que provoca que TCP dispare retransmisiones y ACKs duplicados y finalmente la conexión falle.

Los equipos USRP N210 utilizando una antena con más ganancia pueden ser idóneos para comunicaciones en zonas rurales, donde los alcances podrían ser alrededor de cientos de metros. OpenBTS no es ideal para aplicaciones en tiempo real debido a su poco ancho de banda y altos retardos, sin embargo para momentos de emergencia en donde no se provee de infraestructura será la elección idónea.

Bibliografía

[1] Wipro, «Software-Defined Radio», ago-2002.

[2] «OpenBTS - Open Source Cellular Infrastructure». [En línea]. Disponible en: http://openbts.org/. [Accedido: 05-feb-2015].

[3] «Ettus Research». [En línea]. Disponible en: http://www.ettus.com/. [Accedido: 05-feb-2015].

[4] H. A. M. Hussein Magdy Hussein Ali, «OpenBTS - Network Design & System Analysis», Faculty of Engineering at Cairo University, Giza. Egipto, 2012.

[5] A. Azad, «Open BTS Implementation with Universal Software Radio Peripheral», *Dep. Electr. Comput. Eng. Va. Polytech. State Univ. Va. VA USA*, 2011.

[6] J. Mpala y G. van Stam, «Open BTS, a GSM experiment in rural Zambia», en *e-Infrastructure and e-Services for Developing Countries*, Springer, 2013, pp. 65–73.

[7] K. Heimerl, S. Hasan, K. Ali, E. Brewer, y T. Parikh, «Local, sustainable, small-scale cellular networks», 2013, pp. 2-12.

[8] Erik Dahlman, *4G: LTE/LTE-Advanced for Mobile Broadband*, 1st ed. 2011.

[9] Peter McGuiggan, *GPRS in Practice. A companion to the specifications.* John Wiley & Sons, 2004.

[10] Shiwen Mao, Yingsong Huang, y Yihan Li, «Introducing Software Defined Radio into Undergraduate Wireless Engineering Curriculum through a Hands-on Approach», presentado en 120th ASEE Annual Conference and Exposition, Atlanta Georgia, Estados unidos, 2013.

[11] «WikiStart - GNU Radio - gnuradio.org». [En línea]. Disponible en: http://gnuradio.org/redmine/projects/gnuradio/wiki. [Accedido: 22-ene-2015].

[12] B. Bloessl, M. Segata, C. Sommer, y F. Dressler, «An IEEE 802.11 a/g/p OFDM Receiver for GNU Radio», en *Proceedings of the second workshop on Software radio implementation forum*, 2013, pp. 9–16.

[13] «NI LabVIEW - National Instruments». [En línea]. Disponible en: http://www.ni.com/labview/esa/. [Accedido: 05-feb-2015].

[14] N. Kim, N. Kehtarnavaz, y M. Torlak, «LabVIEW-Based Software-Defined Radio: 4-QAM Modem», *framework*, vol. 10, p. 11, 2007.

[15] «Sora - Microsoft Research». [En línea]. Disponible en: http://research.microsoft.com/en-us/projects/sora/. [Accedido: 05-feb-2015].

[16] K. Tan, H. Liu, J. Zhang, Y. Zhang, J. Fang, y G. M. Voelker, «Sora: high-performance software radio using general-purpose multi-core processors», *Commun. ACM*, vol. 54, n.º 1, pp. 99–107, 2011.

[17] «Simulink - Simulation and Model-Based Design». [En línea]. Disponible en: http://www.mathworks.com/products/simulink/. [Accedido: 05-feb-2015].

[18] Mohammed Aboud Kadhim y Widad Ismail, «Implementation of WIMAX IEEE 802.16e Baseband Transceiver on Multi-Core Software Defined Radio Platform», *Int. J. Comput. Theory Eng.*, oct. 2010.

[19] Texas Instruments, «Product Bulletin. SFF SDR Development Platform». 2007.

[20] «OpenBSC». [En línea]. Disponible en: http://openbsc.osmocom.org/trac/wiki/OpenBSC. [Accedido: 05-feb-2015].

[21] «BS11 – OpenBSC». [En línea]. Disponible en: http://openbsc.osmocom.org/trac/wiki/BS11. [Accedido: 06-abr-2015].

[22] Tsou, T, Cooper, T, y McGwier, R, «Development of an open-source GSM femtocell and integrated core infrastructure», presentado en MILITARY COMMUNICATIONS CONFERENCE, 2012.
[23] «Ettus Research -USRP E100». [En línea]. Disponible en: http://www.ettus.com/product/details/UE100-KIT. [Accedido: 06-abr-2015].
[24] A. P. S. Kalsi, «SMT-8036 based implementation of secured adaptive Software Defined Radio system for LDPC coded modulation techniques», THAPAR UNIVERSITY, 1956.
[25] E. Seurre, *EDGE for mobile Internet*. Boston: Artech House, 2003.
[26] NOKIA Networks, «GPRS Architecture: Interfaces and Protocols». NOKIA Networks, 2004.
[27] Lars Ekeroth, «GPRS support nodes review No 3». Ericcson, 2000.
[28] Kurt Wagentristl, «GPRS Performance Evaluation», Universidad Tecnica de Viena, Viena Austria, 2004.
[29] Agilent Technologies, «Understanding General Packet Radio Service GPRS». Agilent Technologies, 28-jun-2001.
[30] T. Halonen, J. Romero, y J. Melero, Eds., *GSM, GPRS, and EDGE performance: evolution towards 3G/UMTS*, 2nd ed. Chichester, West Sussex, England ; Hoboken, NJ, USA: J. Wiley, 2003.
[31] R. K. INGO MEIRICK y ERICSSON RESEARCH, «Wireless Internet Access Based on GPRS», *IEEE Pers. Commun.*, abr. 2000.
[32] REGIS J. BATES, *GPRS General Packet Radio Service Professional TELECOM*. McGraw-Hill, 2004.
[33] Emmanuel Seurre, *GPRS for Mobile Internet*. Mobile Communications series, 2002.
[34] Peter Stuckmann, «Traffic Engineering Concepts for Cellular Packet Radio Networks with Quality of Service Support», Universidad Técnica de Renania-Westfalia, 2003.
[35] G. Karagiannis y G. Heijenk, «QoS in GPRS», 2000.
[36] 3GPP, «3rd Generation Partnership Project; Technical Specification Group GSM/EDGE Radio Access Network; Radio transmission and reception (Release 8)». 3GPP TS 45.005, mar-2010.
[37] Ascom, «Protocol Overhead in GPRS». Ascom, 2009.
[38] K. Premkumar y A. Chockalingam, «Performance Analysis of RLC/MAC and LLC Layers in a GPRS Protocol Stack», *IEEE Trans. Veh. Technol.*, vol. 53, n.º 5, pp. 1531-1546, sep. 2004.
[39] M. Iedema, *Getting started with openbts*. [S.l.]: O'Reilly Media, 2015.
[40] «Range Networks». [En línea]. Disponible en: http://www.rangenetworks.com/. [Accedido: 05-feb-2015].
[41] «GNU Affero General Public License - GNU Project». [En línea]. Disponible en: http://www.gnu.org/licenses/agpl-3.0.html. [Accedido: 05-feb-2015].
[42] «SQLite Home Page». [En línea]. Disponible en: https://www.sqlite.org/. [Accedido: 09-abr-2015].
[43] «Fairwaves». [En línea]. Disponible en: https://fairwaves.co/wp/. [Accedido: 14-feb-2015].
[44] «Nuand | bladeRF Software Defined Radio». [En línea]. Disponible en: http://nuand.com/. [Accedido: 14-feb-2015].
[45] «serie SBX». [En línea]. Disponible en: http://files.ettus.com/manual/page_dboards.html#dboards_sbx. [Accedido: 05-feb-2015].
[46] «Ettus Research SBX - Product Detail». [En línea]. Disponible en: http://www.ettus.com/product/details/SBX. [Accedido: 19-feb-2015].
[47] «Universal TUN/TAP device driver». [En línea]. Disponible en: https://www.kernel.org/pub/linux/kernel/people/marcelo/linux-2.4/Documentation/networking/tuntap.txt. [Accedido: 14-feb-2015].

[48] R. Chakravorty, S. Katti, I. Pratt, y J. Crowcroft, «Using TCP flow-aggregation to enhance data experience of cellular wireless users», *IEEE J. Sel. Areas Commun.*, vol. 23, n.º 6, pp. 1190-1204, jun. 2005.
[49] Juha Leppilahti, «TCP Receive Window Size Adaptation in a Mobile Device», HELSINKI UNIVERSITY OF TECHNOLOGY, 2009.
[50] M. Hara, «TCP over 2.5G and 3G Wireless Networks», 23-ago-2001. [En línea]. Disponible en: http://www.ietf.org/proceedings/50/I-D/pilc-2.5g3g-00.txt. [Accedido: 19-feb-2015].
[51] «FTP Server». [En línea]. Disponible en: https://help.ubuntu.com/10.04/serverguide/ftp-server.html. [Accedido: 16-feb-2015].
[52] «AndFTP es es un cliente FTP - Aplicaciones Android en Google Play». [En línea]. Disponible en: https://play.google.com/store/apps/details?id=lysesoft.andftp&hl=es_419. [Accedido: 12-feb-2015].
[53] «Xperia ZL Full Specifications | Xperia Devices». [En línea]. Disponible en: http://www.xperiadevices.com/xperia-zl-full-specifications. [Accedido: 19-feb-2015].
[54] «Spectran HF-4060 V3». [En línea]. Disponible en: http://www.aaronia.co.uk/HF-4060.htm. [Accedido: 05-mar-2015].
[55] «Bash - GNU Project - Free Software Foundation». [En línea]. Disponible en: http://www.gnu.org/software/bash/. [Accedido: 21-feb-2015].
[56] «FTP». [En línea]. Disponible en: https://www.ietf.org/rfc/rfc959.txt. [Accedido: 14-abr-2015].
[57] «Wireshark · Go Deep.» [En línea]. Disponible en: https://www.wireshark.org/. [Accedido: 16-feb-2015].
[58] Haiqing Jiang, «Tackling bufferbloat in 3G/4G networks», *IMC 12 Proc. 2012 ACM Conf. Internet Meas. Conf.*, 2002.
[59] ETSI TC SMG, «EDGE Feasibility Study Work Item 184; Improved Data Rates through Optimised Modulation Version 0.2». 13-oct-1997.
[60] Range Networks, «OpenBTS Application Suite v 4.0». 15-abr-2014.
[61] J. S. Seybold, *Introduction to RF propagation*. Hoboken, N.J: Wiley, 2005.
[62] «Monitorización de señal GSM - Aplicaciones Android en Google Play». [En línea]. Disponible en: https://play.google.com/store/apps/details?id=com.signalmonitoring.gsmsignalmonitoring&hl=es_419. [Accedido: 22-abr-2015].
[63] «Gqrx SDR | A software defined radio powered by GNU-Radio and Qt». [En línea]. Disponible en: http://gqrx.dk/. [Accedido: 12-feb-2015].

Glosario

2.5G	2.5 Generation
3G	third Generation
3GPP	Third-generation Partnership Project
4G	fourth generation
ACK	acknowledgement
ADC	Analog-to-Digital Conversion
AM	Amplitude modulation
AMPS	Advance Mobile Phone Service
API	Application Programming Interface
APN	Access point name
ARFCN	Absolute Radio Frequency Number
BCCH	Broadcast Control Channel
BCCH	Broadcast common control
BDP	Bandwidth Delay Product
BEC	Backward Error Correction
BG	Border gateway
BGP	Border gateway protocol
GMM	GPRS mobility management
BSC	Base station controller
BSN	Block Sequence Number
BSS	Base station subsystem
BTS	Base transceiver station

CDMA	Code Division Multiplex Access
CLI	Command Line Interface
CRC	Cyclic Redundancy Codec
CS	Coding Scheme
DAC	Digital-to-Analog Conversion
D-AMPS	Digital AMPS
DDC	Digital Down Conversion
DHCP	Dynamic host configuration
DNS	Domain name server
DUC	Digital Up Conversion
ETSI	European Telecommunications Standards
FCC	Federal Communications Commission
FCS	Frame Check Sequence
FDMA	Frequency Division Multiple
FEC	Forward Error Correction
FPGA	Field Programmable Gate
FTP	File transfer protocol
GCC	GNU Compiler Collection
GGSN	Gateway GPRS Support Node
GMM	GPRS Mobility Management
GMSC	The Gateway Mobile
GNU	GNU Is not Unix
GPRS	General Packet Radio Service
GSM	Global System for
GSN	GPRS support node

GTP	GPRS tunneling protocol
GUI	Graphical user interface
HDLC	High-level Data Link
HLR	Home Location Register
HTTP	Hypertext transfer protocol
IP	Internet protocol
ISP	Internet service provider
ICMP	Internet control message
IF	Intermediate frequency
IMEI	International mobile equipment
IMSI	International mobile subscriber
IP	Internet protocol
ISP	Internet service provider
IWMSC	Inter Working Mobile Switching Centre
LLC	Logical Link Control
LTE	Long Term Evolution
LTN	Long Thin Network
LTS	Long Term Sopport
MAC	Medium access control
MAP	Mobile application part
MCC	Mobile Country Code
MIMO	Multiple-input Multiple Output
MNC	Mobile Network Code
MS	Mobile station
MSC	Mobile switching center
MTU	Maximum Segment Size

NMT Nordic Mobile Telephone
N-PDU Network PDU

OFDMA Orthogonal Frequency-Division Multiple Access

PACCH Packet Associated Control
PAGCH Packet Access Grant
PBCCH Packet Broadcast Control
PCCCH Packet Common Control
PCU Packet control unit
PDC Personal Digital Celluar
PDCH Packet data channel
PDN Public Data Network
PDP Packet Data Protocol
PDU Protocol Data Unit
PLMN Public land mobile
PNCH Packet Notificación Channel
POCSAG Post Office Code Standarization Advisory
PPCH Packet Paging Channel
PPP Point-to-point protocol
PTCCH Packet Timing Advance Control Channel
PDTCH Packet Data Traffic Channel

QoS Quality of service

RA Routing area
RACH Random access channel
RADIUS Remote access dial-in
RAM Random Access Memory
RF Radio Frequency

RLC	Radio link control
RTT	Round Trip Time
RTO	TCP retransmission Timeout
RSL	Received Signal Level
SDR	Software Defined Radio
SFF	The Small Form Factor
SGSN	Serving GPRS Support
SIM	Subscriber Identity Module
SIP	Session Initiation Protocol
SM	Session Management
SMS	Short message service
SN-PDU	SNDCP PDU
SNDCP	Sub Network Dependent
SNMP	Simple network management
SS7	Signaling system no.7
SORA	Software-developed Radio
TACS	Total Access Communications System
TBF	Temporaly Block Flow
TCP	Transmission control protocol
TDMA	Time-division multiple access
TFI	Temporary Flow Identity
TIA	Telecommunications Industry Association
TID	Tunnel Endpoint Identifier
TRX	Transceiver
TTLI	Temporaly Logical Link
UHD	USRP Hardware Driver

UDP	User datagram protocol
UMTS	Universal mobile telecommunications
USF	Uplink State Flag
USRP	Universal Software Radio
VLR	Visitor location register
VoIP	Voice Over IP
VPN	Virtual Private Network
WAP	Wireless Application Protocol
WiFi	Wireless Fidelity
WiMAX	Worldwide Interoperability for
XML	eXtensible Markup Language

Apéndices

10.1 Instalando OpenBTS V5

Esta sección describe la instalación de OpenBTS.

Debe estar previamente instalado Ubuntu 12.0 debe estar previamente instalado Ubuntu 12.04 i386.

OpenBTS utiliza nuevas características en Git por lo que se debe verificar que la aplicación cliente es mayor a 1.8.2 de la siguiente forma.

```
$ git --version
  git version 1.9.1
```

De lo contrario de debe actualizar la aplicación ejecutando lo siguiente

```
$ sudo apt-get install software-properties-common python-software-properties
$ sudo add-apt-repository ppa:git-core/ppa
  (press enter to continue)
$ sudo apt-get update
$ sudo apt-get install git
```

Para obtener el código de debe tener una cuenta GitHub y generar la llave SSH. Se debe verificar las llaves existentes con el siguiente comando:

```
ls -al ~/.ssh
# Lists the files in your .ssh directory, if they exist
```

Este comando mostrara si existe el archivo. Los nombres de los archivos por defecto que contiene la llave pública son:

- id_dsa.pub
- id_ecdsa.pub
- id_ed25519.pub
- id_rsa.pub

Para generar una nueva llave se ejecuta el siguiente comando con un correo electrónico valido y presionar ENTER.

```
$ ssh-keygen -t rsa -C "tu_correo@ejemplo.com"
# Creates a new ssh key, using the provided email as a label
# Generating public/private rsa key pair.
# Enter file in which to save the key (/c/Users/you/.ssh/id_rsa): [Press enter]
```

10.1.1 Crear una nueva llave

Se pedirá que se ingrese una contraseña

```
Enter passphrase (empty for no passphrase): [Type a passphrase]
# Enter same passphrase again: [Type passphrase again]
```

Lo que mostrara una salida como la siguiente

```
Your identification has been saved in /c/Users/you/.ssh/id_rsa.
# Your public key has been saved in /c/Users/you/.ssh/id_rsa.pub.
# The key fingerprint is:
# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db correo@ejemplo.com
```

Se agrega la nueva llave a ssh-agent

```
# start the ssh-agent in the background
ssh-agent -s
# Agent pid 59566
ssh-add ~/.ssh/id_rsa
```

Agregar la llave a la cuenta GitHub

Se debe copiar la llave del archivo id_rsa.pub y agregarse a tu cuenta GitHub dentro del menú configuración -> SSH keys -> add SSH key y finalmente dar click en Add key. Para verificar que todo funciona correctamente se teclea.

```
$ ssh -T git@github.com
# Attempts to ssh to GitHub
```

Se pedirá la contraseña y se observara lo siguiente:

```
The authenticity of host 'github.com (207.97.227.239)' can't be established.
# RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.
# Are you sure you want to continue connecting (yes/no)?
```

Al teclear “yes” se enviará un mensaje de autenticación exitosa.

Obtener el código

El siguiente comando descargara el proyecto

```
$ git clone https://github.com/RangeNetworks/dev.git
```

Para descargar los componentes necesarios ejecutamos el siguiente script

```
$ cd dev
$ ./clone.sh
```

Para seleccionar la versión 5 de OpenBTS

```
$ ./switchto.sh master
```

Para la compilación exitosa se deben tener instaladas las siguientes librerías:

- autoconf
- libtool
- libosip2
- libortp
- libusb-1.0
- g++
- sqlite3
- libsqlite3-dev
- libreadline6-dev
- libncurses5-dev
- liba53

Para ello ejecutamos el siguiente comando

```
$ sudo apt-get install autoconf libtool libosip2-dev libortp-dev libusb-1.0-0-dev
g++ sqlite3 libsqlite3-dev erlang libreadline6-dev libncurses5-dev
```

Para instalar liba53 se ejecuta lo siguiente

```
$ cd liba53/trunk
$ sudo make install
```

Con el siguiente script se compilaran e instalaran todas las dependencias dependiendo el tipo de radio que se esté utilizando. Los paquetes compilados se guardaran en la carpeta BUILDS Para este caso se utiliza el USRP N210.

```
$./build.sh N210
```

10.1.2 Instalación de transceiver

Para instalar los controladores del USRP N210 se ejecuta lo siguiente:

```
$ 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
```

Esto instalara el software UHD y permitirá recibir actualizaciones. Para verificar el correcto funcionamiento de los controladores utilizamos el comando ***uhd_usrp_probe***, es debe desplegar las características del dispositivo N210.

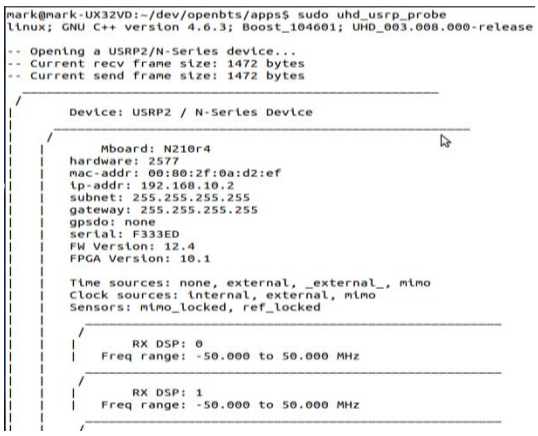

Figura 9.1: Salida del comando *uhd_usrp_probe*

10.1.3 Configuración de OpenBTS

Se debe crear la base de datos ***OpenBTS.db*** la cual almacenara toda la configuración del sistema, esta debe ser instalada en /***etc/OpenBTS.*** Crearemos la base de datos por defecto con el ***archivo OpenBTS.example.sql*** dentro del directorio ***apps*** así como la ruta mencionada anteriormente.

Esta base de datos está implementada en SQLite3 y permite una rápida configuración de OpenBTS. Existen las configuraciones dinámicas que permite que no se interrumpa el sistema mientras que en el caso de las estáticas se necesita reiniciar el sistema.

```
$ sudo mkdir /etc/OpenBTS
$ sudo sqlite3 -init ./apps/OpenBTS.example.sql /etc/OpenBTS/OpenBTS.db ".quit"
```

Probamos la base de datos con

```
sqlite3 /etc/OpenBTS/OpenBTS.db .dump
```

Deben mostrarse todas las variables de configuración. Para correr OpenBTS ejecutamos el siguiente script desde la raíz de OpenBTS.

```
$ cd apps
$ sudo ./OpenBTS
```

Se desplegara una salida en la consola como la siguiente

```
system ready
use the OpenBTSCLI utility to access CLI
```

Se puede observar el despliegue de frecuencia de OpenBTS en la dirección downlink. Para ello se utilizó una herramienta llamada Gqrx [63]. Los parámetros de configuración se realizar mediante CLI (Command line Interface). Para ejecutar el CLI se ejecuta lo siguiente.

```
$ sudo /OpenBTS/OpenBTSCLI
```

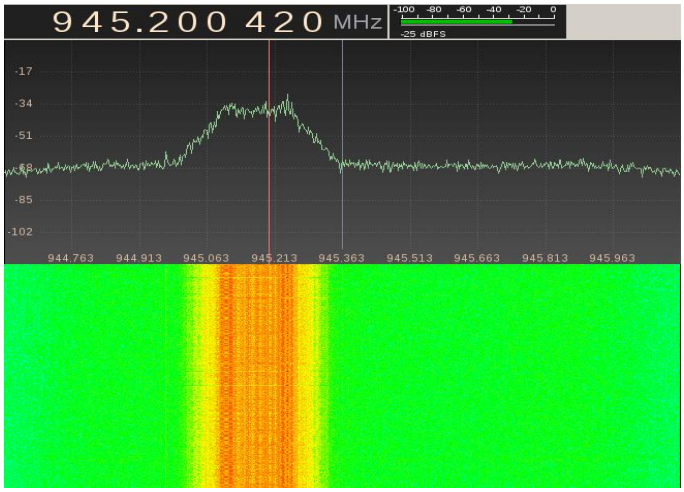

Figura 9.2: Ancho de banda de 200 KHz.

10.1.4 Instalación del registro de suscriptores y sipauthserve

OpenBTS depende de sipauthserve para la autorización y registro de los dispositivos móviles. Para Asterisk el Subscriber Registry database forma parte del registro SIP y del dialplan. Este registro remplaza el rol de HRL y VLR in una red convencional GSM, almacena la información del suscriptor y provee de autenticación A3 RAND-SRES para los suscriptores con ki conocida. Esta base de datos se encuentra localizada en /etc/OpenBTS/sipauthserve.db.

Ya que Asterisk y SIPAuthServe acceden a la base de datos sqlite3 a través de un archivo, ambos deben estar en el mismo servidor físico. La dirección de esta base de datos se encuentra en SubscriberRegistry.db el cual es un parámetro de la configuración de SIPAuthServe.

La ruta por defecto es ***/var/lib/asterisk/sqlite3dir/sqlite3.db.*** Asterisk espera encontrar la base de datos en esta dirección.

Para configurar la base de datos de suscriptor de registro se debe crear la dirección donde la base de datos estará almacenada.

```
$ sudo mkdir -p /var/lib/asterisk/sqlite3dir
```

sipauthserve es un demonio el cual realiza servicios de autenticación. Para compilar Sipathserve ejecutamos:

```
$ cd subscriberRegistry/
$ sudo ./autogen.sh
$ sudo ./configure
$ sudo make
```

Configuramos sipauthserve con el siguiente comando desde la carpeta de apps. La dirección ***/etc/OpenBTS*** debe estar creada.

```
$ sudo sqlite3 -init subscriberRegistry.example.sql /etc/OpenBTS/sipauthserve.db
".quit"
$ sudo ./sipauthserve
```

Procedemos a ejecutarlo desde la carpeta de apps. Si se ha ejecutado correctamente se podrá ver

```
ALERT 139639310980928 sipauthserve.cpp:214:main: ./sipauthserve (re)starting
```

10.1.5 Debugging OpenBTS

Para realizar un debugging del sistema ejecutamos el siguiente comando.

tail -f /var/log/syslog | grep OpenBTS