



UNIVERSIDAD AUTÓNOMA DE YUCATÁN

FACULTAD DE MATEMÁTICAS

IMPLEMENTACIÓN DE UN SISTEMA DE
NAVEGACIÓN CINÉTICA SATELITAL PARA
ASISTIR EN EL CONTROL DE POSICIÓN DE
VEHÍCULOS AÉREOS NO TRIPULADOS

T E S I S

QUE PARA OPTAR POR EL GRADO DE:

Ingeniero en Computación

PRESENTA:

Alex Antonio Turriza Suárez

ASESORES DE TESIS:

Dr. Arturo Espinosa Romero

Dr. Anabel Martín González



Mérida, Yucatán, 2017

Declaración de Autenticidad

Yo, ALEX ANTONIO TURRIZA SUÁREZ, declaro que este trabajo de tesis titulado, «Implementación de un Sistema de Navegación Cinética Satelital para Asistir en el Control de Posición de Vehículos Aéreos No Tripulados» y los resultados presentados son de mi autoría. Declaro que:

- Este trabajo fue realizado durante mi estancia como estudiante de la Facultad.
- Ninguna parte de esta tesis ha sido previamente presentada para la obtención de un grado u otra promoción en esta Universidad u otra institución.
- Todas las consultas a trabajos de otras personas han sido claramente atribuidos a sus respectivos autores.
- Donde se hayan realizado citas textuales del trabajo de otras personas, la fuente siempre estará dada. Con la excepción de dichas citas, esta tesis es totalmente mía.

Firma:

Fecha:

«*Si quieres hacer una tarta de manzana desde cero, primero debes inventar un universo.*»

Carl Sagan

Resumen

El conocimiento del entorno que nos rodea ha sido tema de interés humano desde antaño, así como la interacción con éste en el día a día, para poder investigar diferentes maneras en cómo se pueden aprovechar los recursos disponibles. Unas herramientas para describir el medio geográfico son los mapas realizados a mano; sin embargo, en lugar de ser una herramienta de exactitud, son sólo una referencia del entorno, dado el gran error derivado de la inexactitud humana en las mediciones.

En el presente trabajo se propone la implementación de un sistema de navegación cinética satelital para vehículos aéreos no tripulados (UAV, por sus siglas en inglés), que sea capaz de proporcionar una precisión muy alta (medible en centímetros) de forma que represente un apoyo en la planeación de rutas y ejecuciones de determinadas tareas cada cierta distancia que permitan la creación de mapas de imágenes aéreas más precisos.

Para alcanzar el objetivo, se trabajará con ayuda de software de posicionamiento en tiempo real (Real-Time Kinematics), que servirá de apoyo al momento de alcanzar la precisión deseada, además de dispositivos de hardware tales como un par de receptores GPS (Sistema de posicionamiento global, GPS por sus siglas en inglés) comunicados de forma inalámbrica, indispensables para implementar una corrección de tipo diferencial.

Agradecimientos

A mis padres Antonio y Lucy, aquellos que sin importar las circunstancias me apoyaron de principio a fin en esta odisea de estudiar un nivel superior en una ciudad aparte de donde soy originario. A mi *hermana menor Haydeé Krystel*, quien estaba ahí para poder platicar conmigo y animarme en aquellos momentos donde lo necesité en estos años. Mención especial a mis abuelitos y mis tíos, tías, primos y primas, que siempre me apoyaron y me enseñaron muchas cosas y utilidades de sus vidas.

A mis asesores de tesis, Dr. Arturo Espinosa Romero y Dr. Anabel Martín González, por la innumerable cantidad de conocimientos transmitidos que me serán de utilidad en el resto de la vida, y por su infinita paciencia ante mis dudas y adversidades especialmente durante la realización de este trabajo. Mención especial a todos mis profesores, no sólo de la universidad.

A mi grupo de cercanos amigos: Ernesto [Nestor9224], Kevin [Kevan357], Rafael [RFer93], Oscar [MadOsc], George [El Tío], Beto [Ax Owl xA] por estos años de reuniones para despejar la mente con los videojuegos como excusa. Junto conmigo, Alex [AlexRT07] somos conocidos con diversos nombres tales como *los inútiles, los amigos, los Youtube 3, Ingenieros 0*, o últimamente de forma oficial: [TeamGG]**Los Gansos Galácticos**.

A mis compañeros de la carrera por compartir cada momento no sólo en la estancia en la facultad y hacer de esta experiencia universitaria algo más agradable.

Muchas gracias a todos.

Índice general

Declaración de Autenticidad	II
Resumen	IV
Agradecimientos	V
1. Introducción	2
1.1. Preliminares	2
1.2. Importancia del tema	2
1.3. Planteamiento del Problema	2
1.4. Trabajos previos	3
1.4.1. Propuesta	4
1.5. Hipótesis	4
1.6. Objetivo	4
1.6.1. General	4
1.6.2. Objetivos específicos	4
1.7. Publicaciones	5
1.8. Descripción del documento	5
2. Marco Teórico	6
2.1. Introducción	6
2.2. El Sistema de Posicionamiento Global - GPS	6
2.2.1. Historia	7
Antecedentes	7
Con satélites en órbita	7
2.2.2. La constelación NAVSTAR	8
2.2.3. Estructura	9
Segmento espacial	9
Segmento de control	10
Segmento de usuario	10
2.2.4. Dispositivos GPS	10
GPS NavSpark RAW GPS	11
GPS Ublox C94-M8P	12
2.2.5. Principios de funcionamiento	12
2.2.6. Fundamento matemático	13
2.2.7. Causas de error	16
Error causado por el satélite	17
Error causado por la atmósfera	18
Error causado por rutas múltiples	19
Error causado por el receptor	20
Error por Disponibilidad Selectiva	20
2.3. Comunicación inalámbrica	20
2.3.1. Protocolo ZigBee	21

2.3.2. Módulo UHF del Ublox C94-M8P GPS	21
2.4. Sistemas de cómputo embebidos	22
2.4.1. BeagleBone Black	22
2.4.2. BlackLib	23
2.5. Software Libre	23
2.5.1. GNU/Linux	23
2.6. Real Time Kinematics	24
2.7. Formato RTCM-3	25
2.8. RTKLIB	25
2.8.1. Modos de funcionamiento	25
Modo estático	26
Modo cinematográfico	26
2.9. Vehículos aéreos no tripulados - VANT	26
2.9.1. Instrumentación de vuelo	27
2.10. Conclusión	27
3. Diseño del hardware y software	28
3.1. Introducción	28
3.2. Hardware	30
3.2.1. Prototipo	30
3.2.2. Descripción de la estación base	33
3.2.3. Descripción de la estación móvil	34
Obtención de datos de GPS	34
Visualización de resultados	35
3.3. Software	37
3.3.1. RTKLIB	37
3.3.2. Biblioteca BlackGPIO	40
Biblioteca GPS	40
Biblioteca LCD	40
3.4. Conclusión	40
4. Diseño del experimento	41
4.1. Introducción	41
4.2. Bosquejo del experimento	41
4.2.1. Bosquejo de la figura	41
4.3. Procedimiento	53
4.4. Conclusión	53
5. Resultados	54
5.1. Introducción	54
5.2. Conclusión	54
6. Conclusiones	55
6.1. Introducción	55
6.2. Resultado	55
A. Sistema de Archivos en Red entre una PC y una BeagleBone Black	56
A.1. Introducción	56
A.2. Definición de NFS	56
A.3. Descarga e instalación	56
A.3.1. Instalación en host / PC	56
Seguridad	58

A.3.2. Instalación en cliente / BeagleBone	60
A.4. Ejecución	60
B. Máquina Virtual con Sistema de Arquitectura ARM en PC	61
B.1. Introducción	61
B.2. Software a utilizar	61
B.2.1. QEMU	61
B.2.2. Debootstrap	62
B.3. Descarga e instalación	62
B.3.1. Configuración del sistema	63
B.3.2. Ejemplo: compilando un programa en C++	65
C. Programación de módulos XBEE	67
C.1. Introducción	67
C.2. El software XCTU	67
C.2.1. Descarga e instalación	67
C.2.2. Apertura del programa	69
C.3. Conexión y programación de los XBEE	70
C.3.1. Instalación de controladores	71
Requerimientos para la instalación	71
Instalación	73
C.3.2. Reconocimiento del XBEE con XCTU	76
Identificación del puerto	76
Añadir XBEE a XCTU	78
C.3.3. Programación del XBEE	81
Configurando los modos	81
Configurando el Coordinador	83
Configuración mediante comandos AT	84
Configurando el Router	86
C.4. Prueba: Un chat	88
C.4.1. Preparativos	88
C.4.2. Chateando	89
D. Configuración del GPS Ublox C94-M8P para su uso en RTKLIB	90
D.1. Introducción	90
D.2. Descripción de los componentes	90
D.2.1. Preparando el dispositivo	90
D.3. Instalación en PC	90
D.3.1. Descargas e Instalaciones	91
Software	91
Drivers	92
D.4. Configurando los GPS	94
D.4.1. Configurar la estación base	94
D.4.2. Configurar la estación móvil	100
D.5. Conclusión	102

Índice de figuras

1.1. Posición de celular en mapa	3
2.1. Sistema de Posicionamiento Global	6
2.2. Satélite de la constelación NAVSTAR	8
2.3. Segmento espacial	9
2.4. GPS Navspark	11
2.5. GPS Ublox C94-M8P	12
2.6. El rol que juega cada variable	14
2.7. Error del reloj atómico satelital	17
2.8. Error por capas de la Atmósfera	18
2.9. Error por rutas múltiples	19
2.10. Error del receptor	20
2.11. Equipo XBEE	22
2.12. BeagleBone Black	22
2.13. Esquema de funcionamiento en RTK	24
2.14. Vehículo aéreo no tripulado	26
3.1. Diagrama de componentes	28
3.2. Prototipo	30
3.3. Estación base	33
3.4. Convertidor de RS232 a CMOS RX/TX	34
3.5. Conexión de la LCD	35
3.6. LCD proporcionando información de posicionamiento	36
3.7. Panel del Control	37
3.8. RTKLIB mostrando información de observaciones en Windows	38
3.9. RTKRCV funcionando en GNU_Linux PC	39
4.1. Primer paso del trazado de la figura	41
4.2. Primer arco de circunferencia	42
4.3. Segundo arco de circunferencia	42
4.4. Tercer arco de circunferencia	42
4.5. Cuarto y quinto arco de circunferencia	43
4.6. Segundo lado del cuerpo geométrico	44
4.7. Tercer lado del cuerpo geométrico	46
4.8. Acotamiento del segundo lado de la figura	48
4.9. Acotamiento del tercer lado de la figura	50
4.10. Trazado del último lado de la figura	52
A.1. Archivo /etc/default/nfs-kernel-server ya modificado	57
A.2. Archivo /etc/idmapd.conf	57
A.3. Archivo /etc/exports	58
A.4. Archivo /etc/hosts.deny	59
A.5. Archivo /etc/hosts.allow	59

B.1. Logotipo del software QEMU	61
B.2. Deboostrap dentro de los repositorios.	62
B.3. Salida del comando uname	63
B.4. Versión del sistema instalado.	64
B.5. Contenido del archivo <i>/etc/apt/sources.list</i>	64
B.6. Compilación, ejecución y verificación de la arquitectura en que fue compilado un programa de c++.	65
C.1. Página de descarga de XCTU	67
C.2. Instalador del programa.	68
C.3. Ventana principal del instalador de XCTU.	68
C.4. Carpeta de destino de la instalación.	69
C.5. El instalador informa que el proceso ha terminado correctamente.	69
C.6. Archivo de inicio del software XCTU.	70
C.7. Ventana principal de XCTU.	70
C.8. Dispositivo que permite comunicación entre el XBEE y la PC.	71
C.9. Ventana que indica una correcta instalación de drivers.	71
C.10. Tabla de controladores.	72
C.11. Ubicación del administrador de dispositivos.	72
C.12. Administrador de Dispositivos	73
C.13. Actualizar controlador.	73
C.14. Asistente de instalación del controlador.	74
C.15. Carpeta del controlador.	74
C.16. Selección de carpeta.	75
C.17. Instalación exitosa.	75
C.18. Segunda instalación.	76
C.19. Instalación exitosa de los controladores.	76
C.20. Ubicación del administrador de dispositivos.	77
C.21. Localización del Device Manager.	77
C.22. Identificador del puerto COM utilizado por el XBEE.	78
C.23. Botón para añadir un dispositivo.	78
C.24. Ventana donde se especifica la conexión al dispositivo.	79
C.25. Botón para buscar un dispositivo.	79
C.26. Indique el puerto que el programa muestreará.	80
C.27. Ventana de configuración de búsqueda.	80
C.28. Lista de XBEE encontrados por la herramienta de búsqueda.	81
C.29. Ventana principal de XCTU con un XBEE añadido.	81
C.30. Información del XBEE asociado a XCTU.	82
C.31. Apartado de información y herramientas.	82
C.32. Ventana de programación del firmware.	83
C.33. Botón de inicio del modo terminal.	84
C.34. XCTU en modo consola.	84
C.35. Botón de apertura de comunicación.	85
C.36. Botón de modo de configuración gráfico.	85
C.37. Configuraciones finales en el Coordinador.	86
C.38. Configuraciones finales en el Router, 1.	87
C.39. Configuraciones finales en el Router, 2.	87
C.40. Configuración de sesión en PuTTY.	88
C.41. Terminal PuTTY.	89
C.42. El chat funcionando.	89

D.1. Página web de Ublox	91
D.2. Software U-Center	92
D.3. Asistente de instalación de drivers	92
D.4. Lista desplegable del menú Inicio	93
D.5. Ventana principal del administrador de dispositivos	94
D.6. Ventana principal de U-Center	94
D.7. Seleccionar la opción "Messages View" del menú "View"	95
D.8. Ventana de la opción "Messages"	96
D.9. Opciones de la ventana "TMODE3"	97
D.10. Botón "Send"	97
D.11. Configuración de mensajes enviados inalámbricamente	98
D.12. Configuración de mensajes enviados mediante protocolo RTCM3	99
D.13. Enviar mensaje 1005 mediante RTCM3	99
D.14. Enviar mensaje 1077 mediante RTCM3	99
D.15. Enviar mensaje 1087 mediante RTCM3	100
D.16. Configuración de comunicación del GPS rover	100
D.17. Configuración de datos de navegación	101
D.18. Guardado de la configuración	102

Índice de cuadros

2.1. Variables presentes en el cálculo de posiciones de satélites.	13
2.2. Bandas de frecuencia en comunicación inalámbrica.	21
2.3. Estructura del mensaje RTCM-3.	25
2.4. Contenido del mensaje en formato RTCM-3.1	25

Lista de Abreviaciones

GPS	Global Positioning System
RTK	Real-Time Kinematics
UAV	Unmanned Aerial Vehicle
UHF	Ultra High Frequency

Constantes Físicas

Rotación de la Tierra

$$\omega_{ie} = 7.292\,115\,1 \times 10^{-5} \text{ rad s}^{-1}$$

Velocidad de la Luz

$$c = 2.997\,924\,58 \times 10^8 \text{ m s}^{-1}$$

F

$$F = -4.442\,807\,633 \times 10^{-10} \text{ s}/\sqrt{\text{m}}$$

Excentricidad de la órbita de la Tierra

$$e = 0.017$$

Eje semi-mayor terrestre

$$A = 149.60 \times 10^6$$

Dedicado a todas aquellas personas que hacen especial mi vida...

Capítulo 1

Introducción

1.1. Preliminares

En este documento de tesis se presenta la implementación de un sistema de navegación RTK (Real-Time Kinematics¹) con el apoyo de un software dedicado a navegación en una computadora portátil de bajo costo, basándose en el trabajo de [Takasu y Yasuda \(2009\)](#).

1.2. Importancia del tema

La humanidad ha estado interesada en obtener información acerca del contexto en el que interactúa para poder utilizarlo a su favor.

A través del tiempo, se ha aumentado la certeza de los datos contenidos, por ejemplo, en un mapa, debido al aumento de la precisión de las herramientas de medición utilizadas. Un ejemplo de dichas herramientas son las computadoras, sensores y actuadores, así como de algoritmos que permiten filtrar errores aleatorios.

Mediante el uso de un sistema de navegación en tiempo real (RTK), se tiene como finalidad obtener la información de navegación de un vehículo de forma que este sistema pueda aportar datos confiables medibles en pocos decímetros para apoyar en el control de posicionamiento del mismo.

1.3. Planteamiento del Problema

A pesar del avance de la tecnología en las herramientas de medición, aún persisten algunos errores en la precisión, que de forma práctica, afectan al implementar sistemas basados en ellos.

¹Sistema de Navegación en Tiempo Real por sus siglas en inglés.

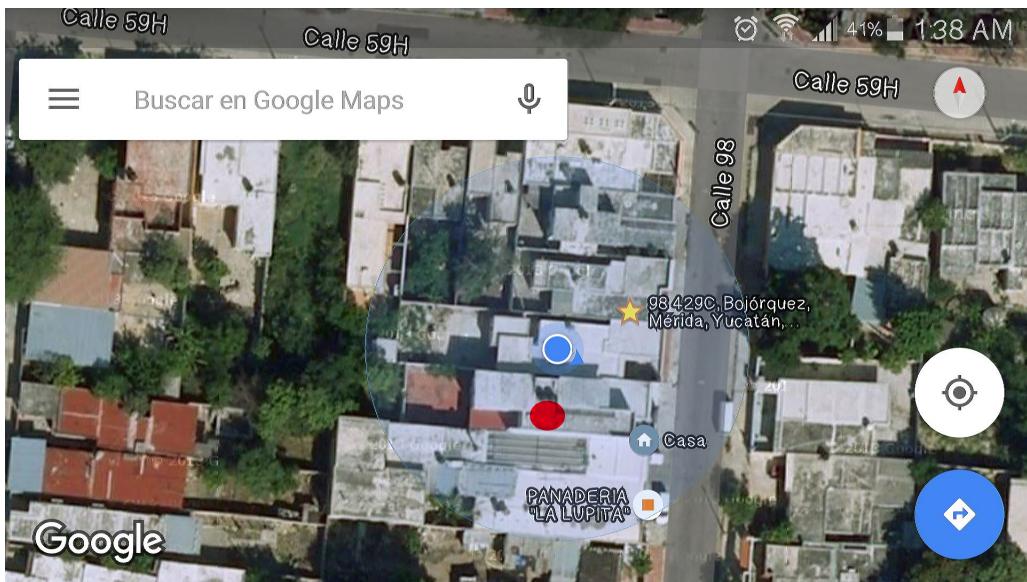


FIGURA 1.1: Posición aproximada de la ubicación de un celular en un mapa de Google Maps.

En una planeación de ruta de un sistema de navegación autónomo, una incertidumbre de n metros a la redonda converge en una inexactitud significativa del sistema. El dispositivo puede encontrarse dentro de cualquier punto de la circunferencia de $2n$ metros de diámetro. Como se puede observar en la figura 1.1, todo el círculo azul denota la posición aproximada de un dispositivo celular con GPS, y en donde la pequeña elipse roja muestra la posición real en dicho mapa. La circunferencia del círculo azul abarca prácticamente la mitad de una manzana, haciendo notorio un problema de impracticidad al intentar implementar un sistema de posicionamiento de, por ejemplo, un automóvil autónomo.

1.4. Trabajos previos

En la tesis de [De la Cruz Bautista et al. \(2011\)](#), se usa un sistema GPS (Global Positioning System²) para apoyar en la logística de distribución de farmacéuticos, como control de rutas, tiempos y seguridad, mencionando ante todo la capacidad de poder obtener los datos de localización y tiempo, además de la velocidad de desplazamiento de un vehículo. Sin embargo, de acuerdo al artículo de [Mendoza Diaz et al. \(2004\)](#), donde se propone una actualización del ancho de carriles carreteros a una amplitud adecuada máxima de 3.6 m, dado que un GPS mide su incertidumbre en una escala de metros, podría indicar que el vehículo se encuentra fuera de la cinta asfáltica o circulando en una vía contraria. Por tanto, se hace importante que un sistema tenga un menor grado de incertidumbre al ofrecido de forma estándar con GPS, como el Real-Time Kinematics, medido en pocos decímetros de error.

También, la tesis escrita por [Rönnbäck \(2000\)](#), menciona la implementación de un sistema de navegación inercial INS/GPS para un UAV (Unmanned Aerial Vehicle³). Entre los datos a destacar, la posición de este sistema es estimada con un error de 2 m con un 95 % de confianza, además de otros datos para asistir al vuelo como velocidad y su altitud. Una incertidumbre de 2 m hace inviable que un drone (vehículo

²Sistema de Posicionamiento Global, por sus siglas en inglés.

aéreo no tripulado) pudiese mantenerse cuando se le requiere que esté estático, por ejemplo, en fotografía aérea.

Revisando el artículo de [Maldonado Hidalgo *et al.* \(2010\)](#), los autores afirman que las coordenadas recibidas de un dispositivo GPS solamente son usadas como aproximación a la posición del vehículo, y nunca como un dato sólido que sirviese al computar datos, dada la exactitud mínima de 10 metros. Sin embargo, mencionan limitantes tales como el peso total de la carga del UAV y el consumo de energía causado por la integración de todos los demás sensores, de donde el GPS aporta entonces una cantidad mínima de información y utilidad en general. Sería ideal que el sistema de posicionamiento aportara información más precisa y que aporte a las observaciones realizadas con el vehículo de forma general.

1.4.1. Propuesta

Dado que en los trabajos previos se presentan algunos inconvenientes con la precisión del sistema de navegación inercial de los dispositivos debido a la naturaleza del sensor GPS común utilizado, se propone la realización de un sistema de navegación que utilice una aplicación de cómputo denominada RTKLIB, que nominalmente reduce el error a una escala medible en centímetros a partir de dos señales de GPS, en un sistema portátil de arquitectura ARM (Advanced RISC Machine), ampliando las posibilidades de aprovechamiento de los datos.

1.5. Hipótesis

Mediante corrección diferencial de tipo Real-Time Kinematics, se puede reducir la incertidumbre de un equipo GPS a niveles de escasos decímetros.

1.6. Objetivo

1.6.1. General

El objetivo general de este trabajo es el diseño y la implementación de un sistema de navegación capaz de conocer su ubicación en un entorno geográfico con una alta precisión para apoyar en proyectos que requieran de dicha información, utilizando dispositivos pequeños y portátiles.

1.6.2. Objetivos específicos

- Configurar los GPS en dos estaciones: una móvil y una base.
- Configurar la transferencia de datos entre estaciones en UHF (Ultra-High Frequency³)
- Acondicionamiento de una microcomputadora BeagleBone con sus respectivos periféricos de apoyo.
- Programación de una biblioteca que controle los periféricos a través de la microcomputadora.

³Vehículo Aéreo No Tripulado, por sus siglas en inglés.

- Integración de módulos.
- Evaluación de los datos obtenidos.

1.7. Publicaciones

Los avances de este proyecto fueron publicados en las memorias en extenso del Encuentro Universitario de Sistemas Computacionales - EUSICS 2016, editado por [?](#).

1.8. Descripción del documento

El presente trabajo se divide en las siguientes secciones:

- **Introducción:** Se describen tanto la importancia del tema, los problemas a resolver, estado del arte, y una lista de objetivos a seguir durante el desarrollo del tema.
- **Marco Teórico:** En este apartado se sientan las bases de funcionamiento de los dispositivos a utilizar, así como información útil acerca de los mismos. Se da una descripción general tanto del hardware como del software utilizado.
- **Diseño del Hardware y Software:** En esta sección se enlistan diversos recursos implementados en el trabajo, tanto de software como de hardware. Se da una descripción de cada uno y el aporte que tienen a la estructura final durante el funcionamiento.
- **Diseño del Experimento:** Se describen las condiciones a las que será sometido el prototipo para indicar un adecuado funcionamiento de acuerdo a los objetivos listados en la introducción.
- **Análisis de Resultados:** Se realiza una evaluación de los resultados obtenidos durante la ejecución de la rutina experimental. Se enfatizan diferencias de los distintos modos de funcionamiento y su impacto en el rendimiento del sistema.
- **Conclusiones:** En forma de síntesis, se da un resumen de los resultados obtenidos de acuerdo al modo de funcionamiento, las observaciones y el rendimiento en general del sistema.

Además, al final se añade una sección de anexos en donde se adjuntan guías de configuración o programación de distintos dispositivos de hardware o de software necesarios para la reproducción del proyecto.

⁴Ultra Alta Frecuencia, por sus siglas en inglés.

Capítulo 2

Marco Teórico

2.1. Introducción

En esta sección se presenta la teoría de varios elementos clave en el prototipo a utilizar. Partiendo desde la definición de GPS, pasando por una breve historia, su fundamento matemático y las causas de error en las mediciones, hasta el software utilizado y los dispositivos específicos utilizados en este trabajo, se presentan datos de funcionamiento y su relación con el proyecto.

2.2. El Sistema de Posicionamiento Global - GPS



FIGURA 2.1: Sistema de Posicionamiento Global.¹

GPS fue iniciado en 1973 para su uso con fines militares por los Estados Unidos de Norteamérica. Su objetivo principal es la determinación de las coordenadas espaciales de un dispositivo bajo una referencia mundial. Para dichos propósitos, se necesita una recepción de señales de un mínimo de cuatro satélites, cuyas coordenadas son plenamente conocidas (*Huerta et al., 2005*).

¹Imagen tomada de: <https://upload.wikimedia.org/>

En la figura 2.1 se observa un instrumento GPS que es empleado para mostrar una ruta entre dos puntos en un mapa y el lugar en que se encuentra el equipo dentro de dicho trazado.

2.2.1. Historia

Antecedentes

Antes de que GPS funcionara mediante satélites, ya estaban implementados varios sistemas de localización radio-terrestres. Uno de ellos era conocido como LO-RAN (*Long Range Navigation*, Navegación de largo alcance, por sus siglas en inglés). Se fundamentaba por el envío de una señal desde distintos emisores. Al percibir señal de tres distintos transmisores, podía determinar su posición.

Fue tras el lanzamiento del primer satélite ruso Sputnik, que investigadores norteamericanos descubrieron que era posible la determinación de la posición gracias a la deformación de una señal por efecto Doppler. Así, conociendo la posición del satélite a través de la posición del observador en la Tierra; a la inversa, sería posible conocer la posición de un observador conociendo la posición del satélite 

Con satélites en órbita

El primer sistema que utilizaba la triangulación mediante satélites fue *TRANSIT* en 1960, entrando en operaciones hasta 1965. Constaba de seis satélites en seis planos, contando con cobertura mundial.

Requería que el observador realizara un seguimiento de 15 minutos a la constelación satelital, pudiendo acceder a los satélites cada hora y media. Su error de precisión rondaba los 250 metros.

En 1973 se propuso el *DNSS* (Defense Navigation Satellite System, Sistema de Navegación Satelital de Defensa por sus siglas en inglés), cambiando de nombre a *NAVSTAR* (Navigation System Time and Ranging, Sistema de Navegación por Tiempo y Distancia, por sus siglas en inglés), después a *NAVSTAR-GPS* y finalmente a *GPS* ([Termal y Jairo, 2014](#)).

2.2.2. La constelación NAVSTAR



FIGURA 2.2: Satélite de la constelación NAVSTAR².

La constelación *NAVSTAR* está conformada por los satélites que se utilizan para triangular la posición de un dispositivo GPS situado dentro del planeta Tierra. Se propuso que fueran 24 satélites situados en seis planos de cuatro satélites cada uno, con cobertura en toda la Tierra. En la figura 2.2 se muestra una representación de un satélite de la constelación orbitando al planeta.

La constelación ha sido lanzada en conjuntos de satélites llamados *Blocks* (bloques en inglés), distribuidos de la siguiente manera, de acuerdo al artículo de [Termal y Jairo \(2014\)](#).

- Block I (1978 - 1985): 11 satélites lanzados, [10 puestos en órbita con éxito](#).
- Block II (1989 - 1990): [9 satélites puestos en órbita con éxito](#).
- Block IIA (1990 - 1997): [19 satélites lanzados con éxito](#).
- Block IIR (1997 - 2004): 13 satélites lanzados, [12 satélites lanzados con éxito](#).
- Block IIR-M (2005 - 2009): [8 satélites lanzados con éxito](#).
- Block IIF (2010): [1 satélite lanzado con éxito](#).

²Imagen tomada de: <https://upload.wikimedia.org/>

2.2.3. Estructura

GPS está conformado por tres segmentos:

- Segmento espacial: los satélites.
- Segmento de control: estaciones terrestres.
- Segmento de usuario: los receptores.

Segmento espacial

Se conoce como segmento espacial al sistema de satélites que orbitan al planeta Tierra, emitiendo señales que permiten al receptor calcular su posición en el marco de referencia terrestre.

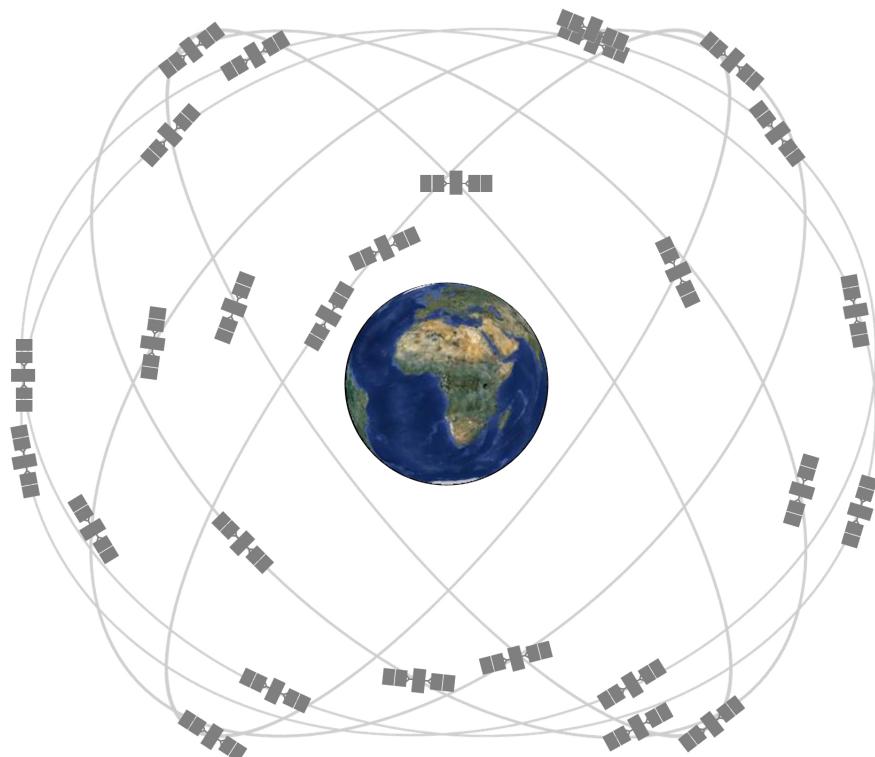


FIGURA 2.3: Segmento espacial.³

Se reparten en seis órbitas sincronizadas y se desplazan a una altitud de 20'000 Km. Cada uno de ellos da una vuelta a la Tierra en 12 horas.

Esto ejemplifica la figura 2.3, los planos orbitales están equitativamente separados de forma que se garantice virtualmente que desde cualquier punto del planeta, un mínimo de 4 satélites sean visibles, siendo éste el número mínimo para un funcionamiento adecuado. Cada órbita contiene 4 satélites, haciendo un total de 24 equipos rodeando al planeta. Hasta la fecha, existen 3 satélites denominados extra. Estos equipos están hechos tanto para mejorar la cobertura de los 24 originales, o bien, sustituir a alguno en caso de presentarse una falla o termine su ciclo de vida

³Imagen tomada de <http://www.gps.gov/>

S. National Coordination Office for space-based positioning navigation and timing, 2017).

Los satélites son ajustados mediante mensajes NAV enviados desde el segmento de control.

Segmento de control

El segmento de control es un conjunto de sistemas instalados en la Tierra que permiten el funcionamiento óptimo del segmento espacial. Conformada por 12 estaciones, de las cuales la principal MCS (Master Control Station⁴) se encuentra en Colorado, en la Base Schriever de la Fuerza Aérea.

Tras el envío de nueva información, cada satélite sincroniza su reloj atómico y ajusta las efemérides de su órbita. El cálculo de esta última es a través de un filtro de Kalman, permitiendo una estimación precisa a pesar de múltiples factores externos.

Segmento de usuario

El segmento de usuario está conformado por equipos receptores diseñados para recibir, decodificar y procesar las señales de los satélites. Los parámetros para controlar la calidad de los receptores utilizados son los siguientes, de acuerdo a (Termal y Jairo, 2014):

- La antena debe estar bajo cielo abierto recibiendo la señal de los satélites con suficiente potencia.
- El reloj interno del receptor debe mantener la sincronización entre receptor y satélites.
- El número de canales debe habilitar al receptor para sintonizar un número suficiente de señales.
- El procesador debe tener una frecuencia tal que garantice un cálculo correcto de la posición a partir de las señales captadas.

El segmento espacial y el segmento de usuario están conectadas por dos frecuencias inalámbricas: L1 y L2. La primera es de 1575.42 MHz y la segunda de 1227.60 MHz (Farrell, 2008).

2.2.4. Dispositivos GPS

Los dispositivos GPS son aparatos receptores que obtienen la información de los satélites y realizan un procesamiento del mismo para ubicarse en el marco de referencia terrestre.

Existen GPS de distintos tamaños y marcas. La diferencia entre éstas consiste en la velocidad del procesador, las frecuencias de los mensajes de los satélites que puede captar, su rango de operación y su velocidad de muestreo y actualización, y si es capaz de aplicar algún tipo de correcciones al momento. Algunos equipos pueden otorgar la información de sus observaciones sin procesar.

⁴Estación de Control Principal por sus siglas en inglés.

GPS NavSpark RAW GPS

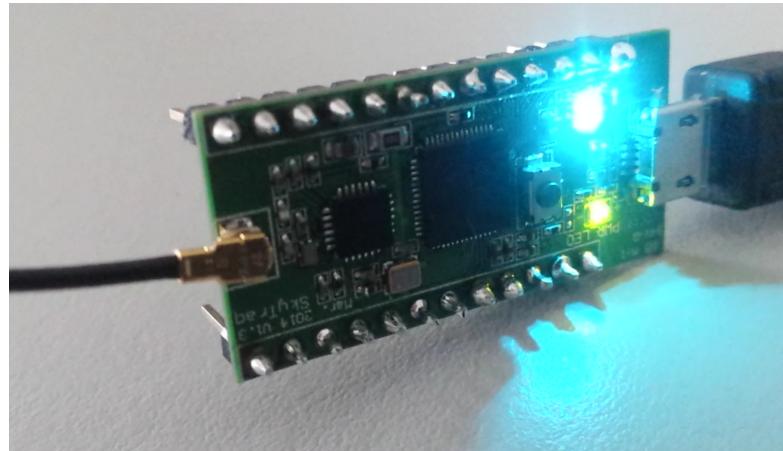


FIGURA 2.4: GPS Navspark.

Fabricado por NavSpark, mostrado en la figura 2.4. Sus características son las siguientes ([NavSpark, 2016](#)):

- Procesador: 100MHz 32bit LEON3 Sparc-V8 + IEEE-754 Compliant FPU.
- 17 Digital I/O.
- GPS con actualización de hasta 20 Hz.
- Rango de operación: ($h < 18000 \text{ msnm}$) & ($v < 515 \text{ m/s}$).
- Precisión de hasta 2.5 m.
- Consumo: 15 ma @ 3.3V.

GPS Ublox C94-M8P



FIGURA 2.5: GPS Ublox C94-M8P.

Fabricado por Ublox, mostrado en la figura 2.5. Sus características son las siguientes ([U-Blox, 2016](#)):

- 72 channel u-blox M8 engine GPS L1C/A, GLONASS L1OF, BeiDou B1.
- GPS con actualizaciones de hasta 10 Hz.
- Rango de operación: $(h < 50000 \text{ msnm}) \& (v < 500 \text{ m/s})$.
- Precisión de hasta 2.5 m.
- Consumo: 23 ma @ 3.3V.
- [Incluye módulo de radiofrecuencia.](#)

2.2.5. Principios de funcionamiento

El objetivo de GPS como sistema de localización es el de calcular la posición de un punto cualquiera en un espacio de coordenadas (x,y,z) ([Sonnenberg, 2013](#)), comenzando con el cálculo de las distancias del punto a un mínimo de tres satélites con localización conocida. Dicha distancia se obtiene multiplicando el tiempo de propagación de la señal emitida por su velocidad de difusión.

Al error de distancias causado por el desfase temporal del reloj del receptor se le llama pseudodistancia ([Pozo-Ruz et al., 2000](#)).

2.2.6. Fundamento matemático

En esta subsección se muestra una síntesis del libro de Farrell (2008), del cálculo de la posición de los satélites.

A través de las frecuencias L1 y L2, el segmento espacial provee de cobertura a todo el planeta. Farrell (2008) menciona al **pseudorango** como la distancia entre un satélite y un dispositivo GPS que incluye errores de medición debidos al reloj del dispositivo. El **pseudotiempo** lo define como el tiempo de tránsito del mensaje de los satélites a los aparatos GPS, incluyendo el error.

El objetivo de un receptor GPS en cualquier parte de la Tierra es calcular su posición a través de la determinación de la posición de los satélites que enviaron sus datos de localización en un determinado momento.

Las variables importantes a utilizar están listados en la tabla 2.1.

CUADRO 2.1: Variables presentes en el cálculo de posiciones de satélites.

Variables	
t'_r	El pseudotiempo en que el receptor toma una medición.
t_{sv}	El pseudotiempo en el que un satélite emite la señal hacia los receptores.
t'_p	El pseudotiempo de propagación de la señal emitida por el satélite hasta llegar a la antena del receptor.
Δt_{sv}	El sesgo del reloj satelital.
t	El tiempo de GPS en el que el satélite emite la señal hacia los receptores.
b	El sesgo del reloj del receptor.
t_r	El tiempo de GPS en el que el receptor toma una medición.
t_p	El tiempo de propagación de la señal desde la antena del satélite hasta la antena del receptor.

En la figura 2.6 se puede observar a cada variable relacionada con el objeto que juega durante el proceso del cálculo de posición de los satélites.

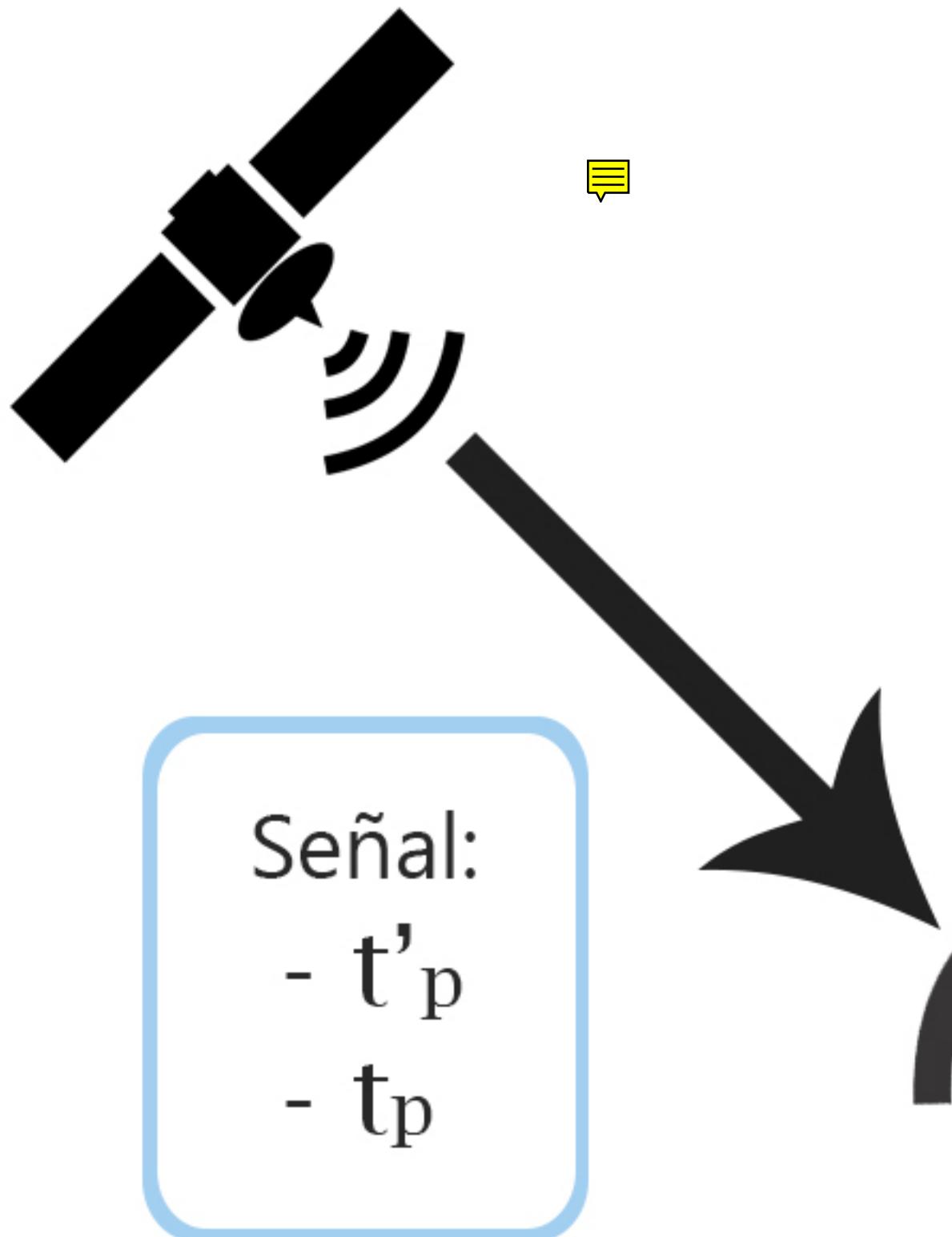


FIGURA 2.6: El rol que juega cada variable.

Las variables t y t_{sv} están dadas por:

$$t_{sv} = t + \Delta t_{sv} \quad (2.1)$$

donde t_{sv} es llamado el *tiempo de canal*, porque el receptor mantiene un valor distinto para cada canal.

Las variables t_r y t'_r están dadas por:

$$t'_r = t_r + b \quad (2.2)$$

donde t'_r es el tiempo del receptor y es común para todos los canales.

El tiempo de propagación t_p está dado por:

$$t_p = t_r - t \quad (2.3)$$

en el cual los valores típicos para t_p rondan entre 60 ms y 90 ms.

Ahora, asumiendo que b inicialmente es desconocida, el cálculo se resume como sigue:

1. En el tiempo t'_r , el receptor toma una muestra de todos los canales en los que encuentra señal.
2. En cada canal, el procesador del receptor calcula t_{sv} .
3. El procesador del receptor calcula el pseudotiempo de propagación y el pseudorango ρ para cada canal.

$$t'_p = t'_r - t_{sv} \quad (2.4)$$

$$\rho = ct'_p \quad (2.5)$$

donde c es la velocidad de la luz: $c \approx 2,998 \times 10^8 \text{ ms}^{-1}$.

4. Se calcula Δt_{sv} utilizando la siguiente ecuación:

$$\Delta t_{sv} = a_{f0} + a_{f1}(t_{sv} - t_{oc}) + a_{f2}(t_{sv} - t_{oc})^2 + \Delta t_r \quad (2.6)$$

donde t_{oc} y los coeficientes polinomiales a_{fi} , $i \in \{0, 1, 2\}$ son parte del mensaje de navegación proporcionado por los satélites. El término Δt_r es una corrección relativista:

$$\Delta t_r = FeA^{1/2} \sin(E_k) \quad (2.7)$$

F , e y A son constantes cuyos valores son: $F \approx -4,443 \times 10^{-10} s/\sqrt{m}$, $e = 0,017$ y $A = 149,60 \times 10^6$. E_k es la ecuación de Kepler de la anomalía de la excentricidad, dada por:



$$E_k = M_k + e \sin(E_k) \quad (2.8)$$

donde M_k es la anomalía media en radianes, dada por:

$$M_k = M_0 + t_k n$$

del que t_k es el tiempo desde la época referenciada, dada por:

$$t_k = t - t_{oe}$$

en el cual t_{oe} es una constante dada en la observación satelital. M_0 es una constante dada en la observación satelital y n es la media de corrección de movimiento, en radianes por segundo (rps), dada por:

$$n = n_0 + \Delta n$$

Donde n_0 es la media de compensación de movimiento en rps, dada por:

$$n_0 = \sqrt{\mu/A^3}$$

y Δn es una constante dada por la observación.

5. Partiendo de que es posible calcular Δt_{sv} con la ecuación 2.6, se puede calcular t utilizando la igualdad 2.1, de la siguiente manera:

$$t = t_{sv} - \Delta t_{sv} \quad (2.9)$$

Una vez obtenida t , es posible el cálculo de la posición de los satélites. Contando con dichas posiciones y las mediciones de pseudorangos, el receptor puede calcular su propia posición y el sesgo de reloj b .

6. El receptor ahora puede calcular t_r y t_p .

2.2.7. Causas de error

Como en todo sistema de medición, es probable que un cálculo de una posición se vea afectado por cualquiera de los siguientes factores:

- Satélites.
- Atmósfera.
- Rutas múltiples.
- Receptor.

Error causado por el satélite

FIGURA 2.7: Error del reloj atómico satelital⁵.

Los satélites, representados en la figura 2.7, incorporan relojes atómicos de gran exactitud. Como el tiempo es crítico al momento de la triangulación de cualquier dispositivo, un error de apenas un nanosegundo equivale a un error en distancia de 30 cm. Los relojes atómicos acumulan un error de esta magnitud cada tres años.

También, al error en la posición de los satélites sobre sus órbitas se le atribuye un error de 2.1 metros.

⁵Imagen tomada de: <https://pixabay.com/>

Error causado por la atmósfera

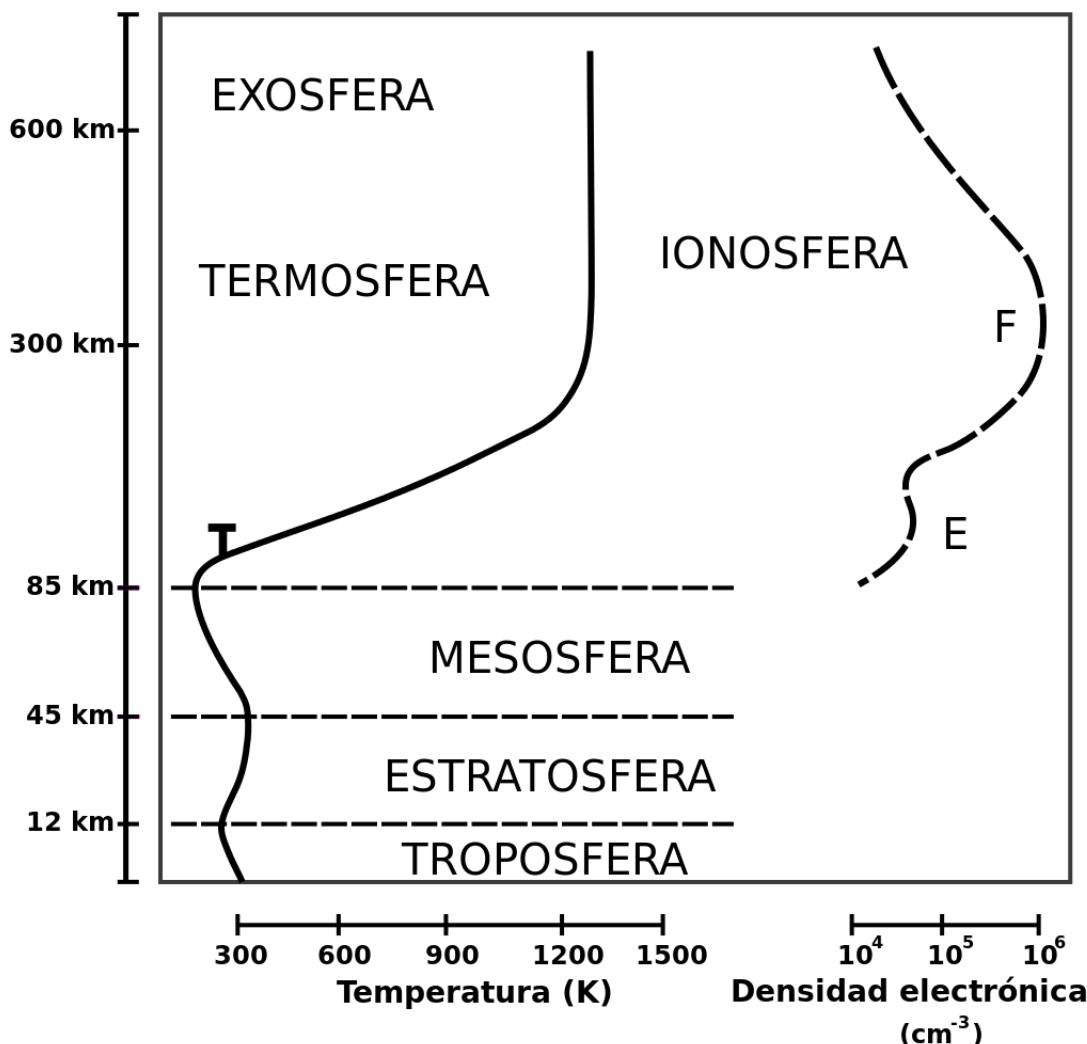


FIGURA 2.8: Gráfica de densidad electrónica y temperatura de las capas atmosféricas⁶.

Las señales de radio que comunican a los satélites y los receptores deben atravesar a la atmósfera a través de considerables kilómetros. El sólo hecho de atravesar partículas cargadas en la ionósfera⁷ y entrar en contacto con el vapor de agua de la tropósfera causa variaciones de velocidad en la transmisión. La figura 2.8 muestra una gráfica que muestra la densidad electrónica y la temperatura de las capas.



El error en distancia atribuible a esta etapa es de 4 metros.

⁶Imagen tomada de: <https://upload.wikimedia.org/>

⁷El error obtenido en la ionósfera puede eliminarse utilizando receptores de frecuencia doble L1 y L2.

Error causado por rutas múltiples

FIGURA 2.9: Error por rutas múltiples⁸.

El sistema está diseñado para funcionar idealmente a cielo abierto. Sin embargo, en condiciones normales, esto no puede ser del todo reproducible, ya sea por el uso en áreas forestales como el de la figura 2.9, o áreas urbanas. Normalmente, la señal directa llega primero al receptor, y después, arriban las que proceden de las rutas múltiples. Esta diferencia de tiempo ocasiona otro error de medición. Este mismo efecto era visible en las señales análogas de televisión, en las imágenes dobles. Las nuevas antenas exteriores pueden filtrar este efecto.

⁸Imagen tomada de: <https://pixabay.com/>

Error causado por el receptor

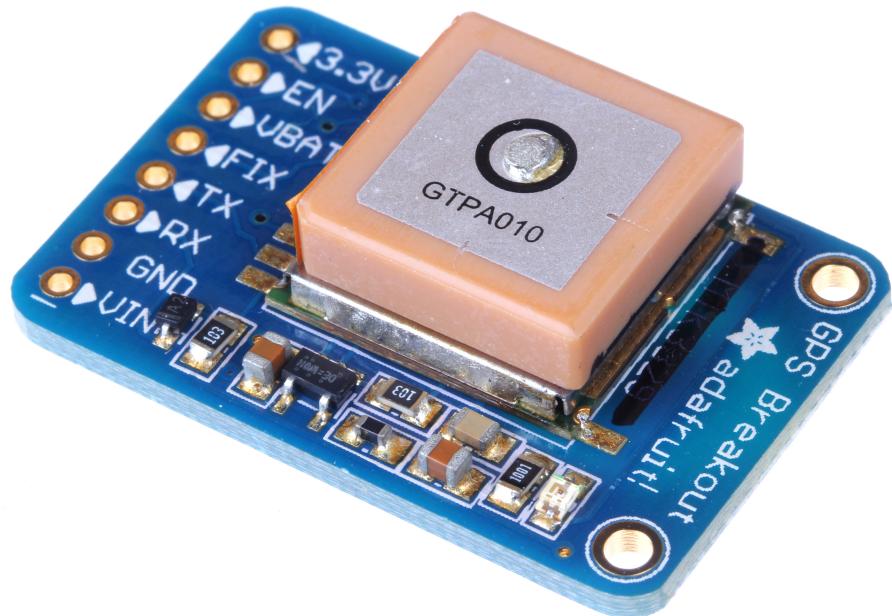


FIGURA 2.10: Error del receptor⁹.

Así como los satélites, los receptores, como el de la figura 2.10 cuentan con sus propios relojes. Debido a costos y dimensiones, éstos no pueden ser atómicos y por tanto son menos exactos, siendo otra fuente de error en la medición. El error asociado a esta causa suele ser de 0.5 metros (Fallas, 2002).

Error por Disponibilidad Selectiva

GPS nació siendo un proyecto de tipo militar. Tras liberarse para uso civil en 1983 por autorización del presidente Ronald Reagan, se habilitaron dos secuencias codificadas llamadas códigos: C/A para uso civil y P para uso militar, con mayor precisión. En el código C/A, se habilitó un error intencionado para evitar un alto grado de precisión, conocido como *Disponibilidad Selectiva*. El presidente Bill Clinton ordenó eliminarle en el año 2000 (Termal y Jairo, 2014). GPS actualmente mantiene incorporada la opción de disponibilidad selectiva, cuyo error ronda los 100 metros, en caso de que el gobierno de Estados Unidos desee reabilitarlo. En 2007, el presidente de Estados Unidos anunció que los satélites de los bloques III no incorporarían más dicha opción (CHÁFER, 2017).



2.3. Comunicación inalámbrica

Se dice que dos dispositivos se comunican de forma **inalámbrica** cuando éstos interactúan sin un contacto sólido entre sus masas. En los aparatos electrónicos, se suelen usar ondas de radiofrecuencia, que a su vez se subclasifican dependiendo de la velocidad a la que son emitidas. A menor frecuencia, se obtiene una gran cobertura pero la capacidad se ve mermada. **Conforme aumenta, se pierde capacidad de**



⁹Imagen tomada de: <https://upload.wikimedia.org/>



alcance pero la carga de información que puede tener, aumenta. Se dice entonces que existe una relación inversamente proporcional entre capacidad y frecuencia.

Actualmente, no existe algún organismo internacional que regule las frecuencias utilizadas por los dispositivos de comunicación inalámbrica, por lo que cada país ha de adoptar una regulación propia. En Estados Unidos es la FCC (Federal Communications Commission¹⁰) quien determina dichas regulaciones. En México, es el Instituto Federal de Telecomunicaciones quien se encarga de dichas tareas regulatorias ([Instituto Federal de Telecomunicaciones, 2015](#)). Otras organizaciones reguladoras son la ISO (International Organization for Standardization¹¹) y la EPCglobal.

La clasificación del uso de las frecuencias por zona geográfica se muestra en la tabla 2.2.

CUADRO 2.2: Bandas de frecuencia en comunicación inalámbrica.

País/Región	LF	HF	UHF	Microondas
USA	125-134 KHz	13.56 MHz	902-928 MHz	2400-2483.5 MHz 5725-5850 MHz
Europa	125-134 KHz	13.56 MHz	865-868 MHz	2.45 GHz
Japón	125-134 KHz	13.56 MHz	No permitida	2.45 GHz
China	125-134 KHz	13.56 MHz	No permitida	2446-2454 MHz

Tanto los sistemas LF y HF son para libre uso en todo el planeta. Sin embargo, UHF necesita de autorización y certificación dependiendo de la zona. En EUA, el uso de UHF no requiere de licencia, pero tiene algunas restricciones ([Tapia et al., 2007](#))

2.3.1. Protocolo ZigBee

El estándar IEEE 802.15.4, mejor conocido como ZigBee, es una especificación para aplicaciones de control remoto para cualquier equipo que requiera de un bajo costo y un bajo consumo de potencia en entornos reducidos. ZigBee puede funcionar a tres bandas de frecuencia diferentes: 868 MHz, 915 MHz y 2.4 GHz.

Los módulos XBEE, fabricados por Digi International, siguen el protocolo ZigBee. En la figura 2.11 se muestran dos de estos dispositivos. De entre todos esos módulos, destacan los de la serie PRO, ya que poseen una mayor potencia en la señal y en consecuencia, pueden hasta duplicar la capacidad de alcance en la distancia de transmisión. El módulo requiere una alimentación que va desde los 2.8 V hasta los 3.4 V ([Oyarce et al., 2010](#)).

2.3.2. Módulo UHF del Ublox C94-M8P GPS

Los dispositivos GPS Ublox C94-M8P están diseñados para trabajar en pares y vienen incorporados con módulos de comunicación inalámbrica que operan en los 915 MHz de frecuencia en el continente americano. En él, se puede configurar el envío y recepción de diferentes contenidos. El manual de Ublox recomienda usar el formato RTCM3, que será explicado más adelante ([U-Blox, 2016](#)).

¹⁰Comisión Federal de Comunicaciones, por sus siglas en inglés.

¹¹Organización Internacional para la Estandarización, por sus siglas en inglés.

¹²Imagen tomada de: <https://www.digi.com/>



FIGURA 2.11: Equipo XBEE¹².

2.4. Sistemas de cómputo embebidos

Los Sistemas Embebidos son sistemas programables, que realizan tareas específicas determinadas por el usuario, con el objetivo de optimizar los procesos para mejorar su desempeño y eficiencia, reduciendo tamaño y costos de producción. Se caracterizan por el bajo consumo de energía. Están compuesto por tres componentes principales: Procesador, Dispositivos de almacenamiento y Periféricos ([Caballero Paz, 2014](#)).

2.4.1. BeagleBone Black

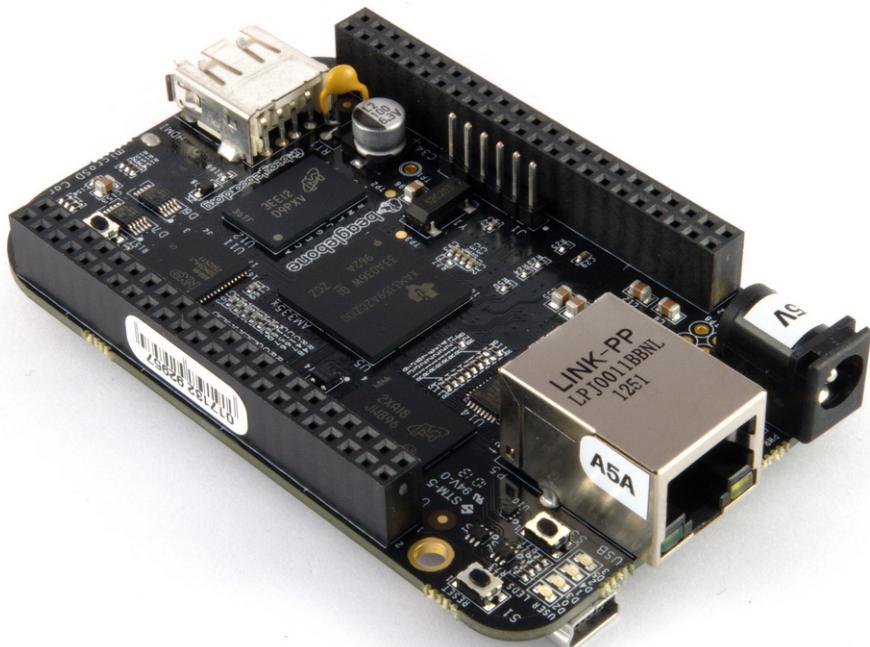


FIGURA 2.12: BeagleBone Black¹³.

La BeagleBone es una plataforma de desarrollo de bajo costo, desarrollada por la BeagleBone Foundation de los Estados Unidos, una fundación sin fines de lucro, cuyo objetivo es la promoción de hardware y software de código abierto para el desarrollo de sistemas embebidos.

Por su diseño, la BeagleBone posee una arquitectura ARM, que posee soporte de varias distribuciones Linux ([Coronado Vallés, 2014](#)). Por su condición de hardware y software abierto, tanto sus esquemáticos del hardware, como las códigos fuente de su software están disponibles a todos los usuarios. En la figura 2.12, se muestra una tarjeta BeagleBone Black.

De las ventajas que posee una arquitectura como la de la BeagleBone, basada en microprocesadores, es que su uso conduce a plataformas más poderosas, capaces de realizar tareas de gran carga computacional ([Coley, 2013](#)).

2.4.2. BlackLib

BlackLib es una biblioteca que permite controlar el hardware y los puertos de la BeagleBone Black a través del lenguaje C++. Puede leer entradas análogas, generar señales PWM (Modulación de Ancho de Pulso), usar pines de propósito general o GPIO, y comunicarse con otros dispositivos a través de comunicación serial ([Yiğit Yüce, 2017](#)).

2.5. Software Libre

El **software** es un conjunto de instrucciones en lenguaje máquina para que una computadora realice funciones específicas.

El código máquina llega a través de un código más entendible para humanos, llamado código fuente, que un compilador se encarga de traducir a lenguaje máquina. Cuando el código fuente no es accesible para nadie, se dice que se trata de código cerrado ([Hernández, 2005](#)). 

El software libre es aquél cuyo código fuente puede ser usado, copiado, estudiado, modificado y redistribuido libremente. A pesar de dicha capacidad de propagación del código, el software puede seguir siendo vendido comercialmente ([García y Cuello, 2007](#)).

2.5.1. GNU/Linux

Linux es un sistema operativo. Un sistema operativo es un conjunto de programas que permiten la interacción con el hardware, además de ejecutar otros programas, así como crear una interfaz con el usuario. En un sistema GNU/Linux, Linux es el núcleo o kernel, y GNU es el conjunto de programas ([The Debian Installer team, 2015](#)).

¹³Imagen tomada de: <https://www.flickr.com/>

2.6. Real Time Kinematics

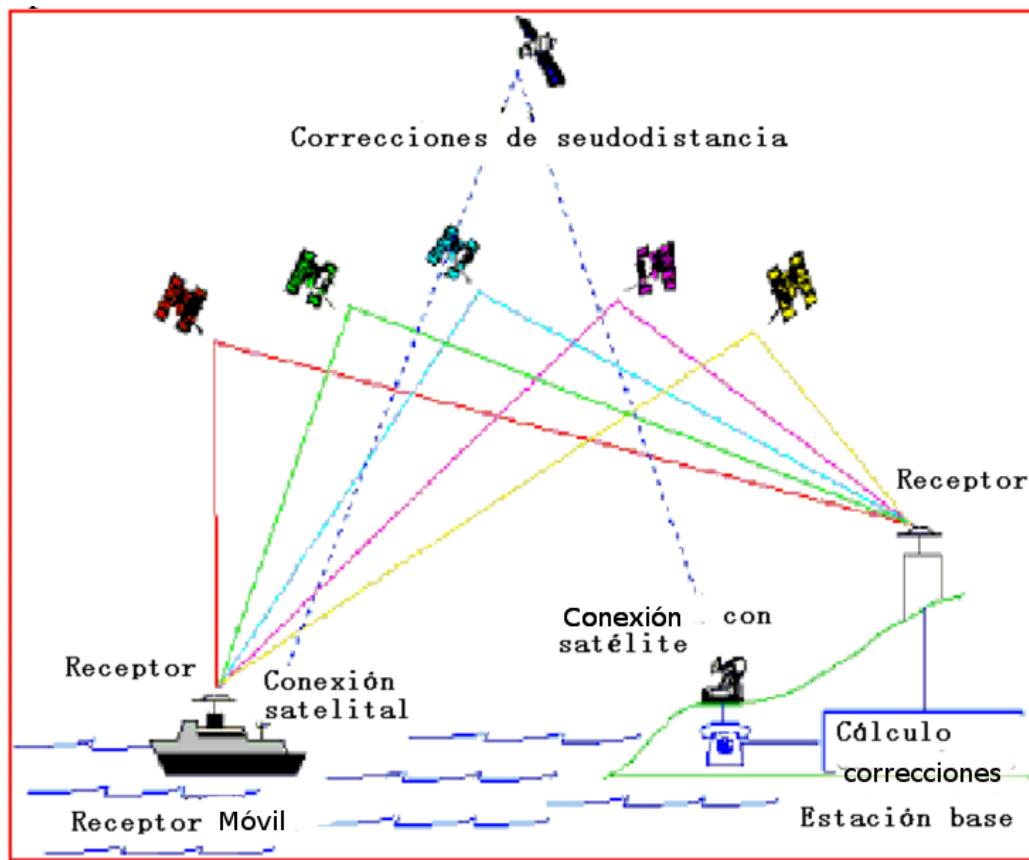


FIGURA 2.13: Esquema de funcionamiento en RTK, por Fallas (2002).

Se le llama Sistema de Navegación Cinética Satelital en Tiempo Real (RTK, Real Time Kinematics por sus siglas en inglés), a las correcciones de la señal del GPS basadas en las señales L1 y L2. Todo está en torno al siguiente supuesto, mostrado también de forma gráfica en la figura 2.13:

Se tienen dos receptores a una distancia de pocos kilómetros entre sí. En esta condición, se podría esperar que los errores causados por el reloj atómico del satélite, por la ionósfera y la tropósfera afectarían de igual manera y con la misma magnitud a ambos receptores, por su proximidad. Si la posición exacta de uno de los receptores es conocida, entonces esta información puede ser usada para determinar el error asociado a las lecturas de dicho receptor y después aplicar la corrección al otro dispositivo. El primer GPS, cuya posición es conocida, recibe el nombre de **receptor base** y el segundo es llamado **receptor móvil**. La estación base calcula la distancia entre cada uno de los satélites de los que recibe señal y su posición (también conocida) para determinar el error asociado a la medición de distancia. Esa información la envía al receptor móvil, quien aplica la corrección hacia dicho satélite, obteniendo así un conocimiento más exacto sobre su posición (Fallas, 2002).

Con todas las correcciones aplicadas de forma ideal, se alcanza una precisión mejor a los 10 cm (Cerrato Miranda, 2011). Conforme el receptor móvil se aleja gradualmente de la estación base, paulatinamente, la utilidad de las correcciones proporcionadas de la segunda a la primera, disminuye (Mueller *et al.*, 1994).

2.7. Formato RTCM-3

El formato RTCM-3 es un estándar internacional para transmitir datos de posicionamiento en tiempo real. Es utilizado sólo por las estaciones base para proporcionar sus observaciones. Su estructura consiste en un mensaje de 42 bits fijos más otros bits que pueden variar de acuerdo al o los mensajes que se requiera enviar ([Rubinov et al., 2011](#)), como muestra la tabla 2.3.

CUADRO 2.3: Estructura del mensaje RTCM-3.

Preámbulo	Reservado	Tamaño del mensaje	Datos del mensaje	Checksum
8 bits	6 bits	10 bits	n bits	24 bits
0xD3	Sin definir	Tamaño del mensaje en bytes	Tamaño variable en bytes	Definición QualComm CRC-24Q

El mensaje utilizado en este trabajo contendrá los datos mostrados en la tabla 2.4:

CUADRO 2.4: Contenido del mensaje en formato RTCM-3.1

Mensaje RTCM-3.1
<ul style="list-style-type: none"> • 1005: (X,Y,Z) Coordenadas fijas de la antena. • 1077: Observaciones de GPS. • 1087: Observaciones de GLONASS.¹⁴

2.8. RTKLIB

Desarrollado por Tomoji Takasu, RTKLIB es un paquete de programas de código abierto escrito en el lenguaje de programación C, para posicionamiento tanto estándar como preciso con sistemas de bajo costo. Soporta varios modos de posicionamiento tales como: Simple, Diferencial, Cinemático, entre otros. En todos sus modos soporta tanto procesamiento en tiempo real así como postprocesamiento ([Takasu y Yasuda, 2009](#)).

2.8.1. Modos de funcionamiento

RTKLIB puede funcionar en distintos modos. Se explicarán un par de ellos en conformidad con su relevancia en este proyecto:

- Modo estático.
- Modo cinemático.

RTKLIB utiliza un filtro de Kalman extendido (*EKF*, extended Kalman filter, por sus siglas en inglés), para obtener los resultados de sus cálculos de aproximación.

¹⁴Homólogo ruso del sistema americano GPS.

Modo estático

En este modo es necesaria una larga observación del cielo y de recolección de datos. Este requisito está fundamentado dado el cambio en la geometría del trayecto de los satélites, que apoyan en la resolución de ambigüedades (Wiśniewski *et al.*, 2013).

Modo cinemático

Requerido cuando el objeto al que se quiere conocer su posición, se encuentra en movimiento. Permite obtener decímetros de precisión. Para un funcionamiento óptimo, necesita las coordenadas de una estación base conocida en el archivo de configuración. Los datos de esta estación fija pueden ser obtenidos mediante mensajes RTCM (Wiśniewski *et al.*, 2013).

2.9. Vehículos aéreos no tripulados - VANT



FIGURA 2.14: Vehículo aéreo no tripulado¹⁵.

Un VANT o Vehículo Aéreo no Tripulado (**UAV**, por sus siglas en inglés) es definido como un vehículo con operaciones y sistemas mecatrónicos, con computadoras a bordo y supervisado por humanos desde tierra (Haluani, 2015). Un ejemplo de éstos se muestra en la figura 2.14. Los vehículos aéreos no tripulados estuvieron orientados inicialmente para operaciones de contexto militar (Fahlstrom y Gleason, 2012). En usos civiles, se pueden destacar empresas que investigan formas de entregar paquetes mediante el uso de esta tecnología, así como observaciones meteorológicas.



Algunas áreas de aplicación de los VANT son (Addati y Lance, 2014):

¹⁵Imagen tomada de <https://upload.wikimedia.org/>

- Imágenes y vídeo aéreo.
- Monitoreo y vigilancia.
- Inspección de infraestructuras.
- Búsqueda y rescate.
- Gestión de emergencias.
- Mapeo de terrenos.

2.9.1. Instrumentación de vuelo

La instrumentación de vuelo es la encargada de obtener el estado de vuelo del UAV de manera precisa, así como el entorno en el que se encuentra haciendo uso de distintos sensores. Dependiendo de la finalidad de una misión de vuelo, se pueden encontrar los siguientes sistemas de sensado ([Barrientos et al., 2007](#)):

Para un marco de referencia inercial.

- Acelerómetros.
- Giroscopios.

Para posicionamiento y velocidad.

- GPS.
- Compás. 

2.10. Conclusión

Tras una revisión de los conceptos y componentes necesarios, se detalla la forma en cómo cada uno de ellos conforma un elemento clave para la obtención del objetivo.

En el capítulo 3, se habla acerca de la integración de los componentes descritos en el capítulo 2. Se presentan descripciones acerca de los componentes, fotografías, especificaciones de funcionamiento y modos de operación, así como las condiciones de uso para un correcto desempeño.

Capítulo 3

Diseño del hardware y software

3.1. Introducción

En este capítulo se presenta información referente al funcionamiento general del sistema. Se divide en dos secciones: el hardware y el software, donde en cada uno se muestran diagramas o imágenes de funcionamiento, junto a una breve explicación.

En el diagrama de la figura 3.1 se muestran los componentes de cada una de las estaciones.

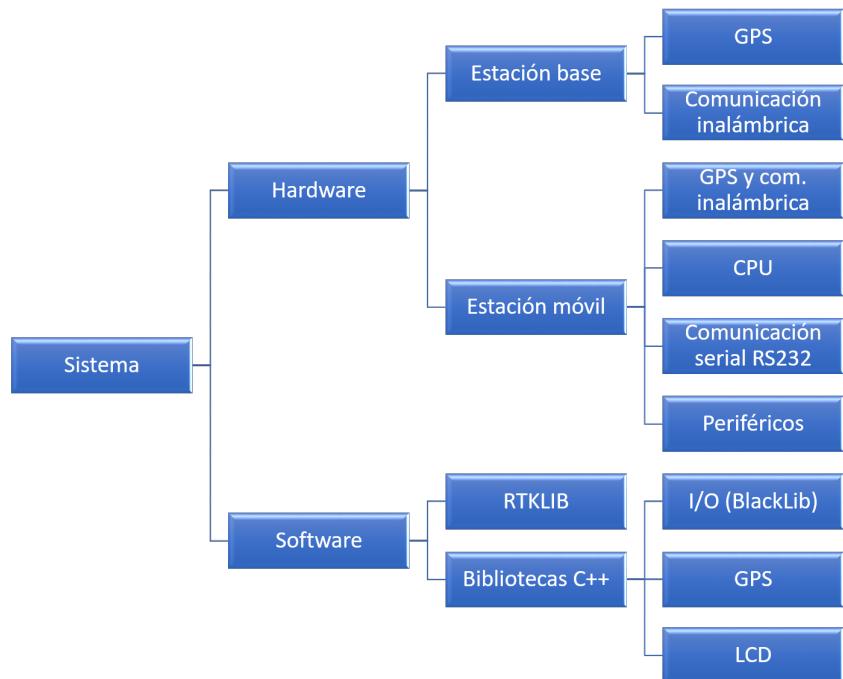
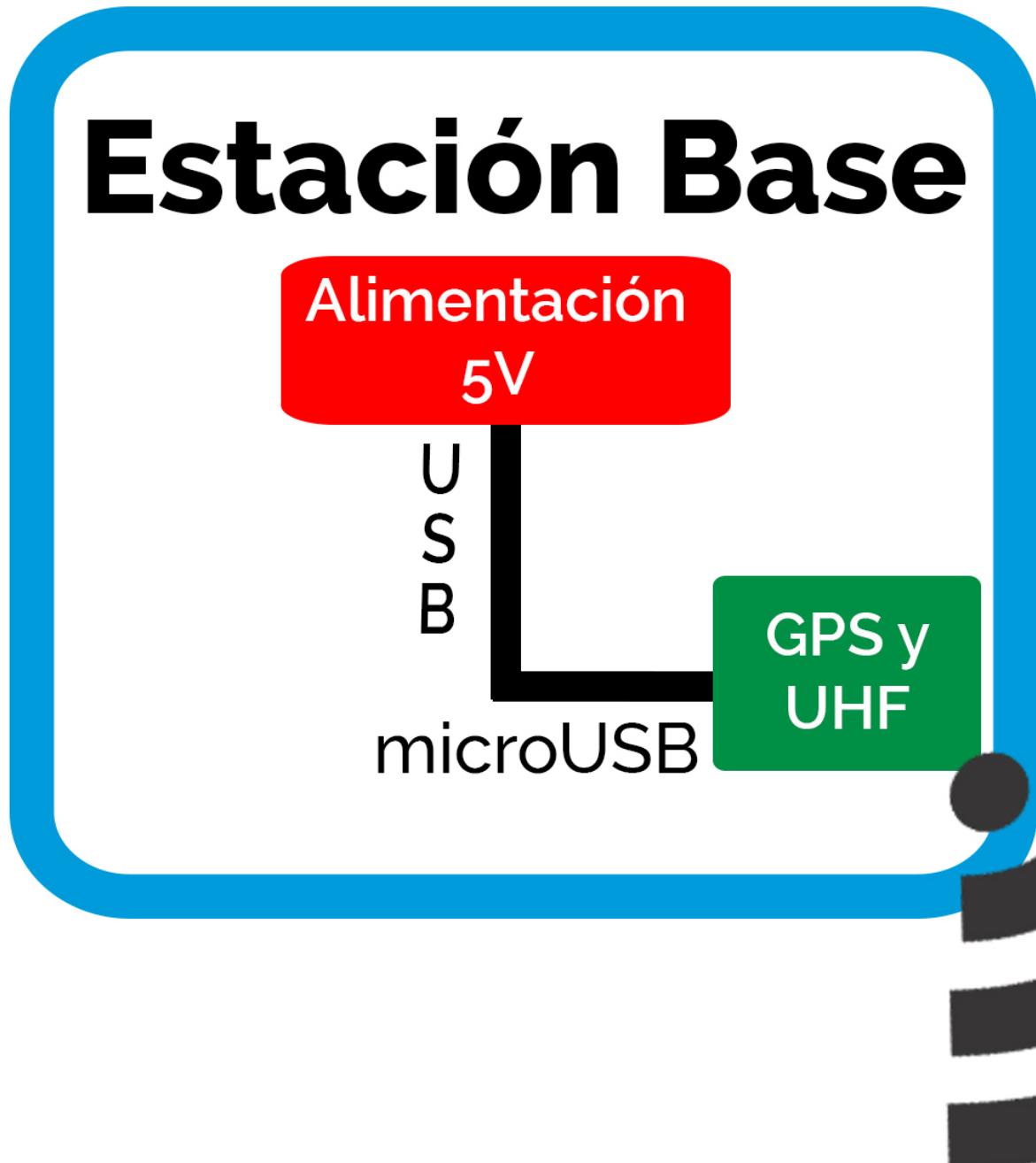


FIGURA 3.1: Diagrama de componentes.

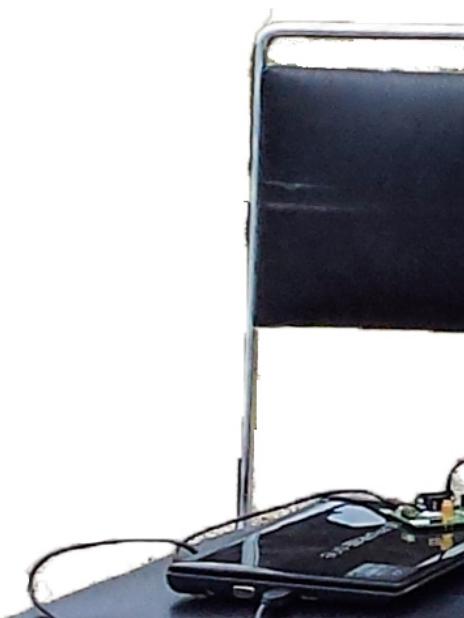
3.2. Hardware

3.2.1. Prototipo



El sistema cuenta con dos estaciones, tal y como se puede observar en la figura 3.2: una que es conocida como *estación base* y es la encargada de ofrecer datos de corrección de las observaciones de GPS, y una estación móvil, cuyo posicionamiento se desea conocer.

3.2.2. Descripción de la estación base



La estación base cuenta con dos componentes principales:

- Tarjeta GPS Ublox C94-M8P.
- Módulo de Comunicación inalámbrica.

En la figura 3.3 se puede observar la estación base en una prueba realizada en el exterior. En la memoria de la tarjeta Ublox C94-M8P de esta estación se encuentran las coordenadas de su posición actual (se pueden actualizar vía PC utilizando el Anexo de Configuración de los Dispositivos GPS Ublox C94-M8P para su uso en RTKLIB en el apéndice D, adjunto a esta tesis).

Mediante la comunicación inalámbrica, los datos de posicionamiento necesarios para aplicar correcciones son enviadas a la tarjeta Ublox de la estación móvil, utilizando el protocolo RTCM3.

3.2.3. Descripción de la estación móvil

Obtención de datos de GPS

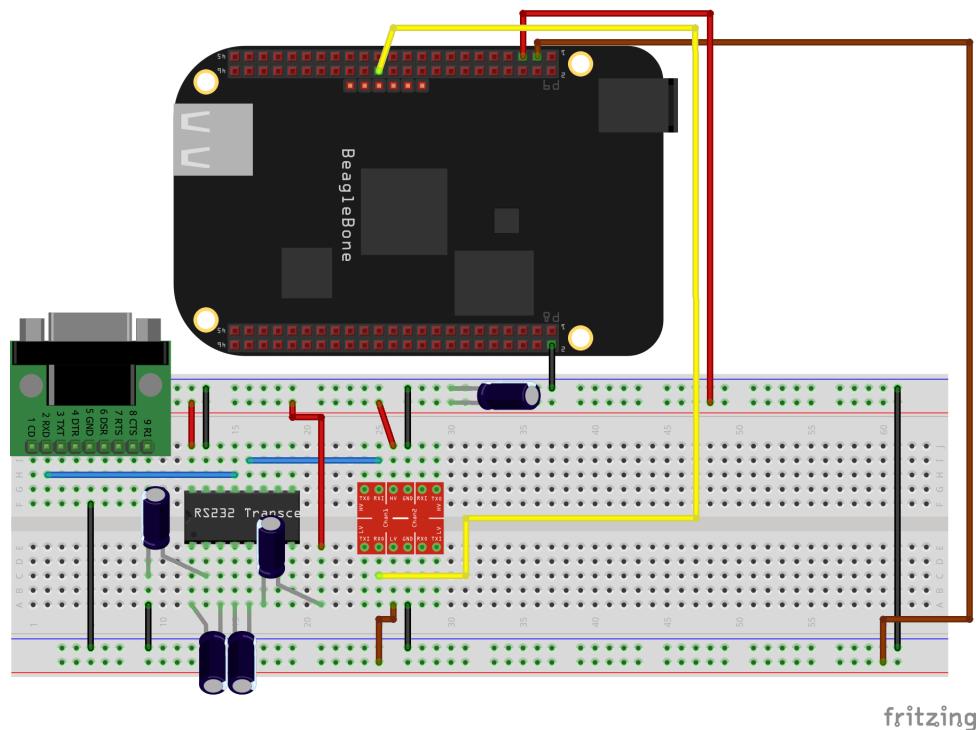


FIGURA 3.4: Convertidor de RS232 a CMOS RX/TX.¹

El diseño de los componentes de la estación móvil gira en torno a las capacidades y al diseño del GPS Ublox C94-M8P, de quien se tomarán dos datos importantes: **datos de posicionamiento de esta misma tarjeta**, ofrecidos por un **puerto microUSB**, y los **datos de corrección de posición** proporcionados por la estación base a través de comunicación inalámbrica, obtenidos por **protocolo RS232**, a través de un periférico como el que se muestra en la figura 3.4. Por tanto, en la tarjeta BeagleBone Black, son reservados el puerto USB y un puerto (dos pines) de comunicación serial.

¹Todos los diagramas de conexión electrónica son para referencia y no muestran el estado real del cableado. Todos fueron realizados con el software fritzing.

Visualización de resultados

Para la visualización de los datos durante el funcionamiento del sistema, se dispone de una pantalla LCD 16x2 RGB conectada a pines de entrada/salida de uso general de la BeagleBone de la forma que muestra la siguiente figura 3.5. En la figura 3.6, se puede observar dicha LCD proporcionando datos ya procesados de localización:

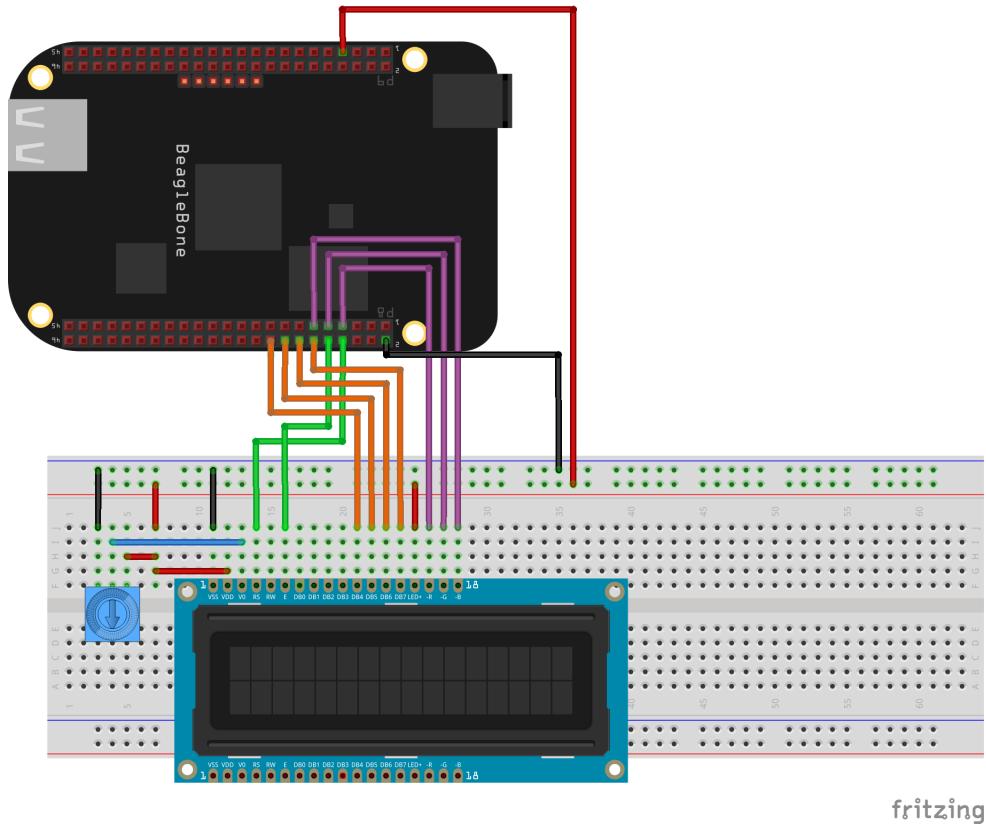


FIGURA 3.5: Conexión de la LCD.

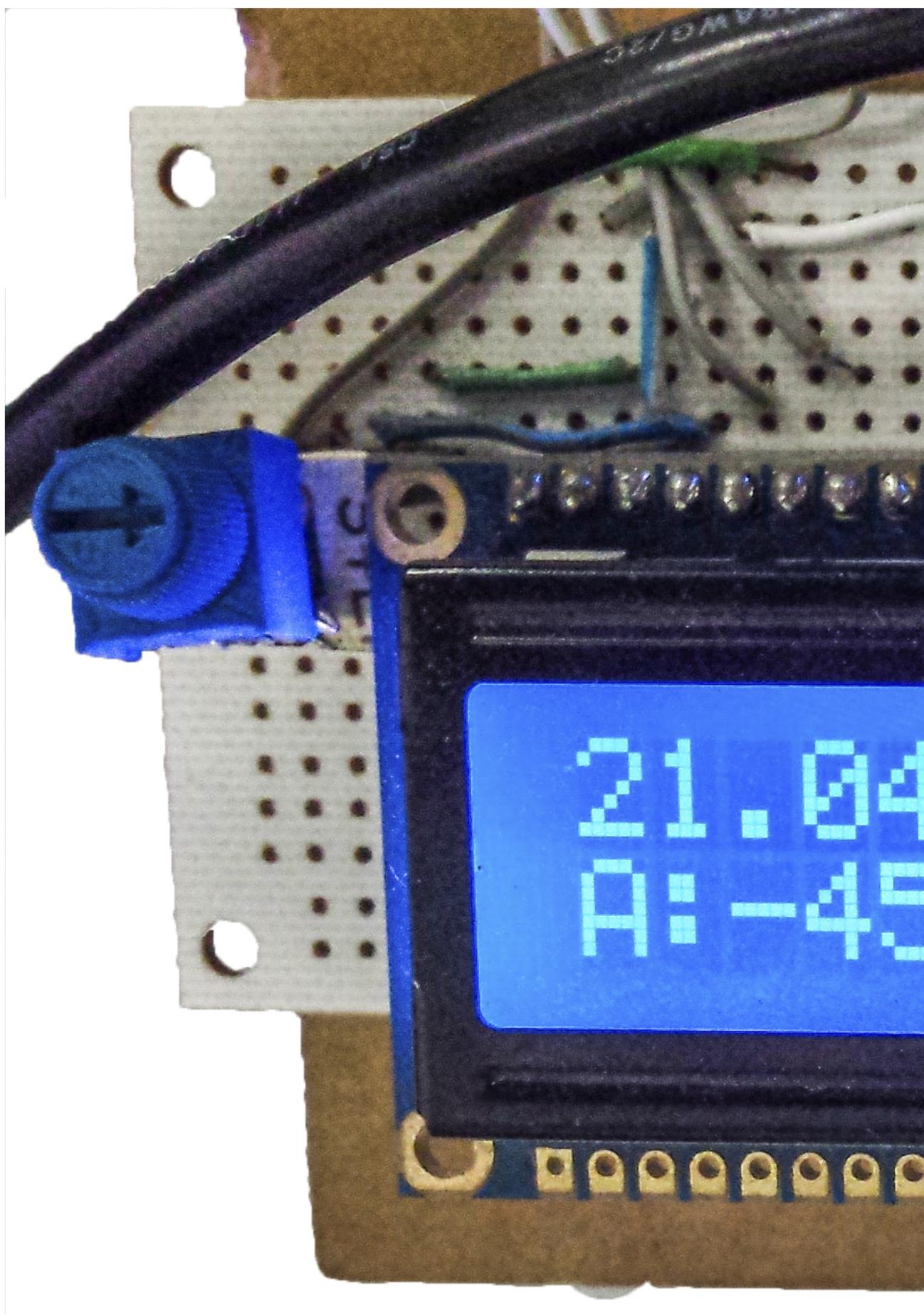


FIGURA 3.6: LCD proporcionando información de posicionamiento.

También, para un mejor control de estados del programa, se dispone de dos push buttons y un diodo LED conectados cada uno a pines de entrada/salida de uso general en la BeagleBone, tal y como muestra el circuito de la figura 3.7.

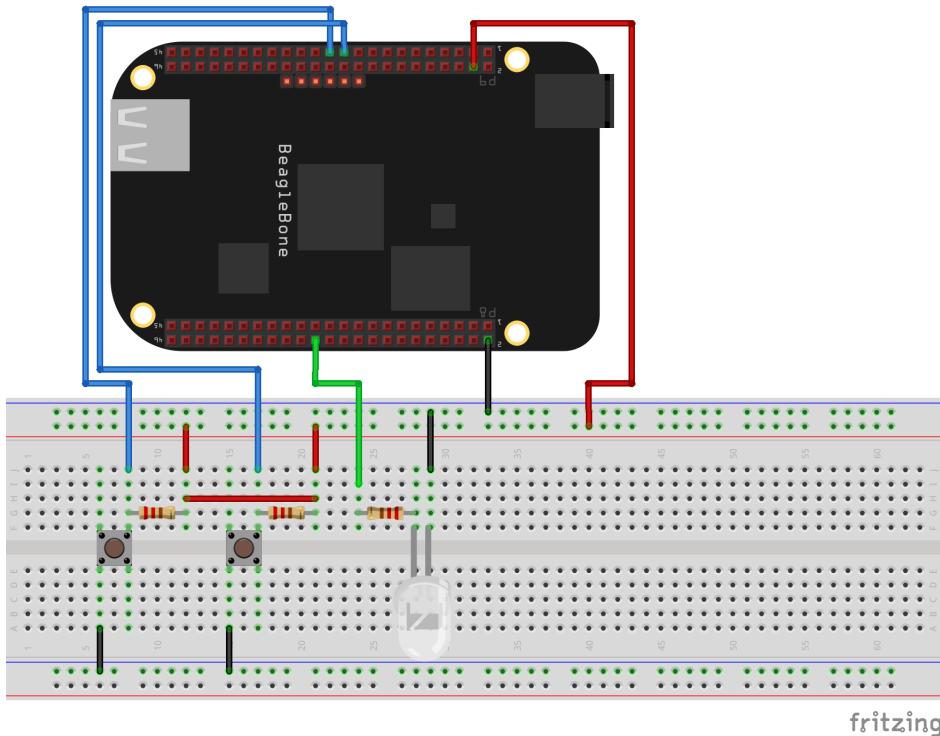


FIGURA 3.7: Panel de Control.

3.3. Software

En el apartado de software, se cuenta con dos conjuntos importantes de código: **RTKLIB** y bibliotecas para manejo de los pines de entrada/salida de uso general. Mediante estos últimos, fueron implementados los programas que controlan a los diversos periféricos de hardware conectados, incluido el GPS.

3.3.1. RTKLIB

Para una correcta implementación de RTKLIB²en la BeagleBone Black, basada en arquitectura ARM, se pasó por distintas etapas:

- **Configuración en Windows x86_64:** Dadas las facilidades ofrecidas por RTKLIB en este sistema operativo (dos de ellas y las más importantes, el entorno gráfico y la cantidad de documentación) se decidió configurar el sistema primero bajo Windows, como se puede apreciar en la figura 3.8.

²La versión utilizada es la beta 26 de RTKLIB en su versión 2.4.3, dado que es a partir de estas betas que se da soporte a los mensajes proporcionados por Ublox C94-M8P



Mediante RTKNAVI, una aplicación de RTKLIB, se configuró el entorno para que reconociese los mensajes de GPS a través de USB, los distintos modos de funcionamiento, y cómo otorgaba los datos ya procesados, además de observación de geolocalización mediante un mapa al momento de la puesta en marcha.

- **Hacer funcionar el sistema en GNU/Linux x86_64:** Una vez funcionando RTKLIB en Windows, el objetivo de este punto es el de transportarse a un sistema intermedio entre el utilizado por el dispositivo de destino y el de donde se configuró inicialmente, sabiendo que todo el hardware estaba correctamente conectado y funcionando.

Bajo este Sistema Operativo, la aplicación RTKRCV, que es la equivalente en entorno consola a RTKNAVI en Windows, es la encargada de procesar los datos de la tarjeta GPS Ublox C94-M8P. Se realizaron los ajustes necesarios para hacer operativo el hardware, tales como indicar las rutas de obtención de datos.

Una captura de pantalla de RTKRCV funcionando puede verse en la figura 3.9.

```

alexrt07@HP-Omen15-GTX: ~/RTKLIB-rtklib_2.4.3/app/rtkrcv/gcc

Parameter          : Value
rtklib version     : 2.4.3 b26
rtk server thread  : 1945511680
rtk server state   : run
processing cycle (ms) : 10
positioning mode   : kinematic
frequencies        : L1+L2
accumulated time to run : 00:06:17.5
cpu time for a cycle (ms) : 3
missing obs data count : 0
bytes in input buffer : 0,0
# of input data rover : obs(378),nav(9),gnav(9),ion(96),sbs(0),pos(0),dgps(0),ssr(0),err(0)
# of input data base : obs(377),nav(0),gnav(0),ion(0),sbs(0),pos(377),dgp(0),ssr(0),err(0)
# of input data corr : obs(0),nav(0),gnav(0),ion(0),sbs(0),pos(0),dgps(0),ssr(0),err(0)
# of rtkm messages rover : 1005(377),1077(377),1087(377)
# of rtkm messages base : 1000(377)
# of rtkm messages corr : 1000
solution status     : float
time of receiver clock rover: 2017/01/25 00:23:09.999822897
time sys offset (ns) : 0.000,0.000,0.000,0.000
solution interval (s) : 1.000
age of differential (s) : 1.003
ratio for ar validation : 2.893
# of satellites rover : 7
# of satellites base : 18
# of valid satellites : 7
GDOP/PDOP/HDOP/VDOP : 2.4,2.0,1.2,1.6
# of real estimated states : 3
# of all estimated states : 325
pos xyz single (m) rover : 36864.518,-5955007.947,2276471.158
pos llh single (deg,m) rover: 21.04909199,-89.64531461,7.383
vel enu (m/s) rover : -0.059,-0.011,0.021
pos xyz float (m) rover : 36864.518,-5955007.947,2276471.158
pos xyz float std (m) rover : 0.038,0.049,0.019
pos xyz fixed (m) rover : 36864.688,-5955008.503,2276471.266
pos xyz fixed std (m) rover : 0.008,0.014,0.011
pos xyz (m) base : 36862.876,-5955009.424,2276471.848
pos llh (deg,m) base : 21.04909305,-89.64533050,9.000
# of average single pos base: 0
ant type rover : 
ant delta rover : 0.000 0.000 0.000
ant type base : 
ant delta base : 0.000 0.000 0.000
vel enu (m/s) base : 0.000,0.000,0.000
baseline length float (m) : 2.314
baseline length fixed (m) : 2.114
monitor port : 0

```

FIGURA 3.9: RTKRCV funcionando en GNU_Linux PC.

- **Hacer funcionar el sistema en GNU/Linux ARM:** Una vez configurado correctamente para GNU/Linux en el punto anterior, sólo quedaba conectar todo a la microcomputadora BeagleBone y verificar que todo funcionase como en la

PC. Los datos de salida de RTKLIB son ofrecidos vía socket y son recibidos a través de otro programa programado en C++ del que se hablará más adelante.

3.3.2. Biblioteca BlackGPIO

La biblioteca BlackGPIO fue desarrollada para poder tener un control ordenado de los puertos de la BeagleBone a través de programación en C++. Utilizándole, se crearon otras dos bibliotecas a partir de las cuales se pueden instanciar objetos conectados a la microcomputadora.

Biblioteca GPS

La biblioteca GPS *gps.h* es utilizada para instanciar un objeto de tipo GPS que contiene en sus atributos distintos strings que contendrán datos de observaciones tales como longitud y latitud. Incluye también un buffer de lectura que utiliza internamente para la asignación de los anteriores valores.

Entre sus métodos están conjuntos de sets y gets para manejo de información. Todos están protegidos con exclusiones mutuas que evitan que información errónea sea entregada.

Biblioteca LCD

La biblioteca LCD *lcd.h* permite instanciar una LCD RGB de 16x2 conectada a los pines que se le indiquen al constructor siguiendo el orden de declaración que se indica: rojo, verde, azul, RS, enable, db7, db6, db5, db4. Durante el proceso de instanciaión del objeto, automáticamente se inicializa para el inmediato despliegue de información en la siguiente línea de código. Entre sus métodos se encuentra el apagar y encender la pantalla, cambiar el color, escribir un string en la pantalla, saltos de línea, etcétera.

3.4. Conclusión

Una vez detallado cómo funciona el sistema y el papel que juega cada componente, se procede a definir el ejercicio al cual es sometido el sistema para la obtención de resultados.

En el capítulo 4 se hablará acerca de los experimentos realizados y el cómo fueron diseñados, además de los resultados esperados.

Capítulo 4

Diseño del experimento

4.1. Introducción

En este capítulo, se muestra a detalle la prueba a la que será sometido el equipo de posicionamiento RTK implementado en la BeagleBone, así como la justificación de dicho experimento y se plantean las medidas que se tomarán para conocer la precisión que el sistema tiene.

4.2. Bosquejo del experimento

El sistema Real-Time Kinematics ofrece una precisión medible en distancia. Por lo tanto, se pretende dibujar una figura geométrica de medidas conocidas en una determinada área donde las antenas de GPS estén de la forma menos afectada posible por interferencias planteadas en el Marco Teórico en la sección 2.2.7 en la página 16. Por tanto, se utiliza la siguiente técnica para el trazado de la figura:

4.2.1. Bosquejo de la figura



FIGURA 4.1: Primer paso del trazado de la figura.

Se traza una línea recta de medida a . En la figura 4.1, la **O** indica el punto de origen del recorrido a seguir.

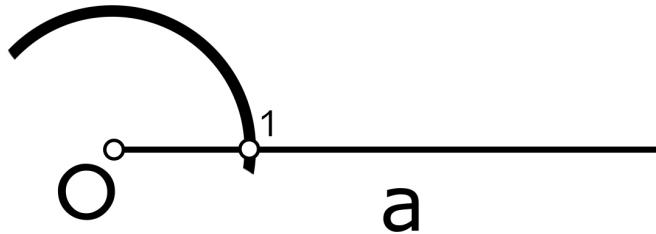


FIGURA 4.2: Primer arco de circunferencia.

Utilizando una cuerda tensa, ahora se dibuja un segmento de circunferencia o arco teniendo como centro al origen **O** y un radio arbitrario r que debe de mantenerse constante, tal y como muestra la figura 4.2. El punto donde el arco interseca a la primera línea será llamado punto 1.

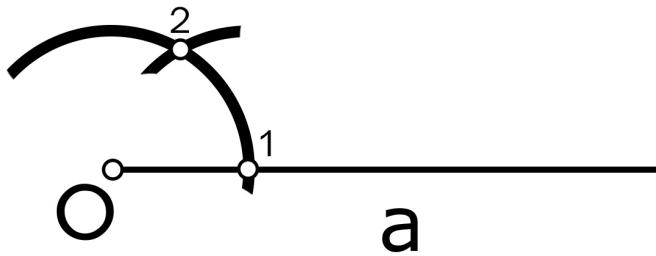


FIGURA 4.3: Segundo arco de circunferencia.

Se trazará otro arco, manteniendo el mismo radio r , pero ahora teniendo como origen de la circunferencia al punto que se denotó como 1, como sugiere la figura 4.3. Al punto de la intersección de los dos arcos dibujados anteriormente se le llamará punto 2.

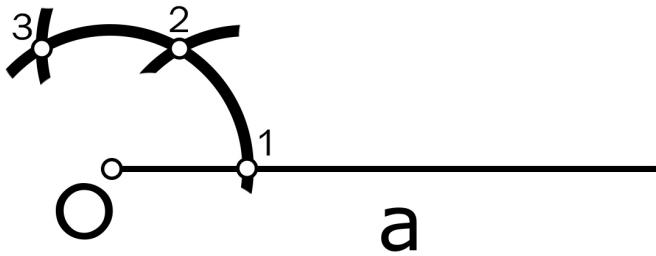


FIGURA 4.4: Tercer arco de circunferencia.

Se repite el paso anterior con el arco de radio r , tomando como origen al punto 2, y buscando intersecarse con el primer arco de circunferencia trazado, como muestra la figura 4.4. Al punto de intersección se le llamará punto 3.

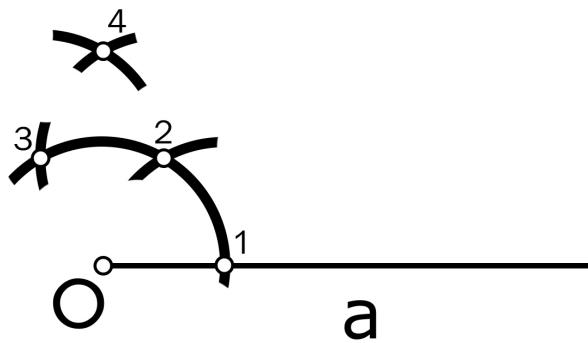
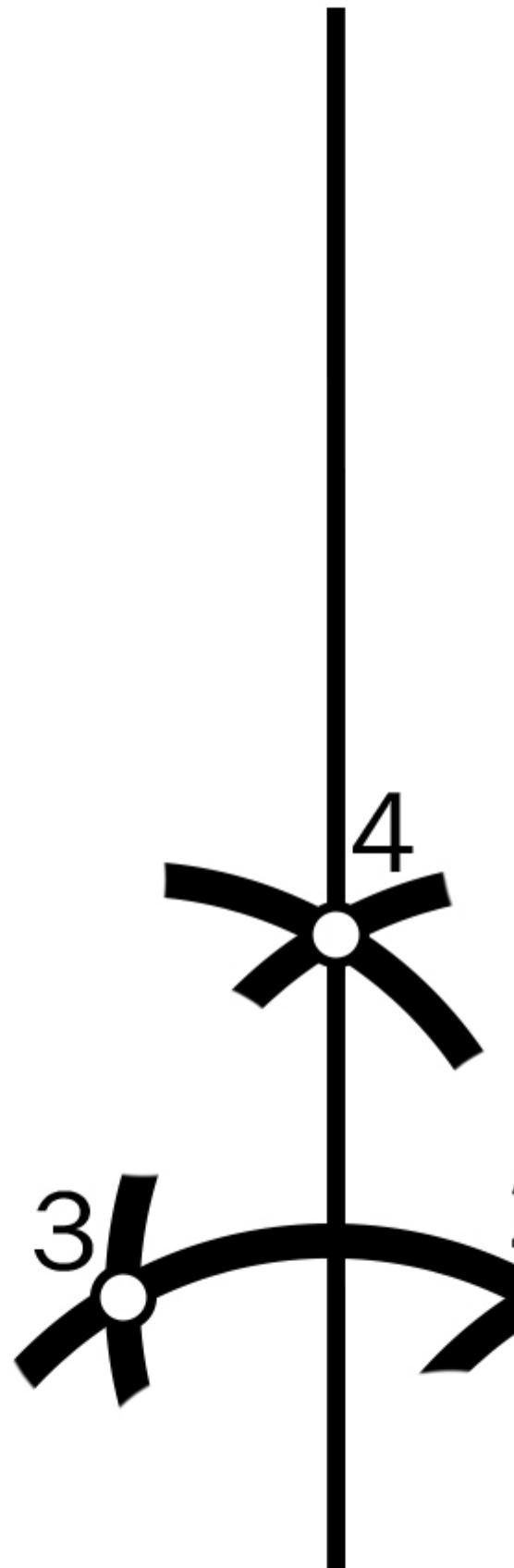
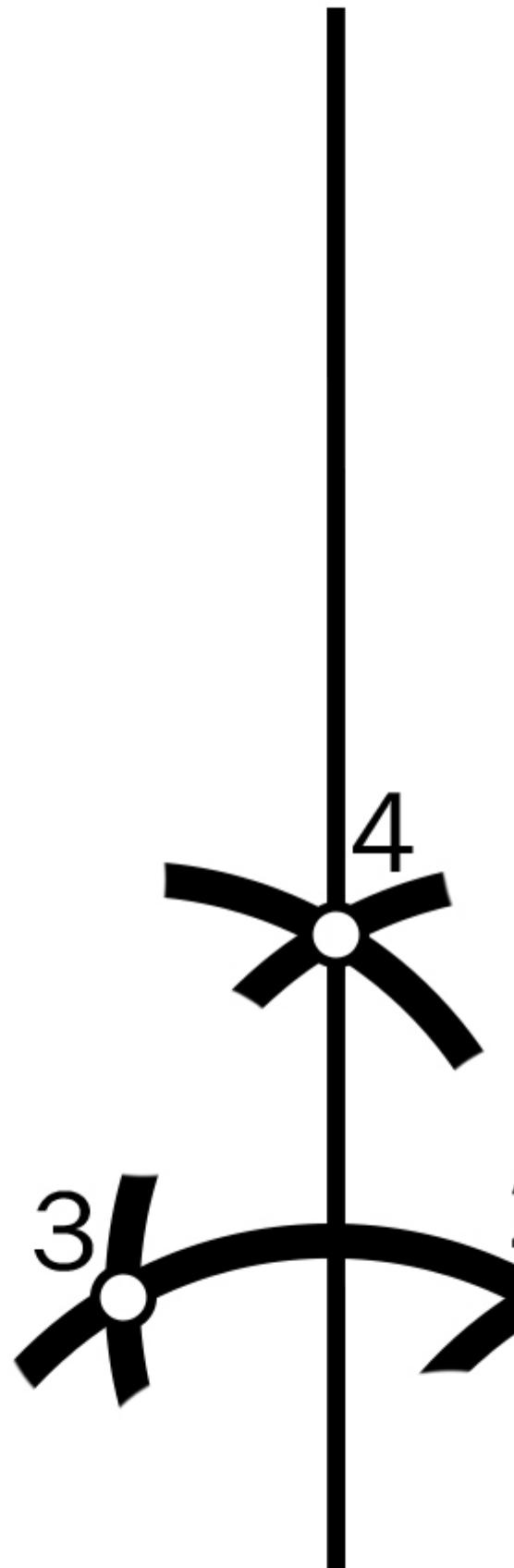


FIGURA 4.5: Cuarto y quinto arco de circunferencia.

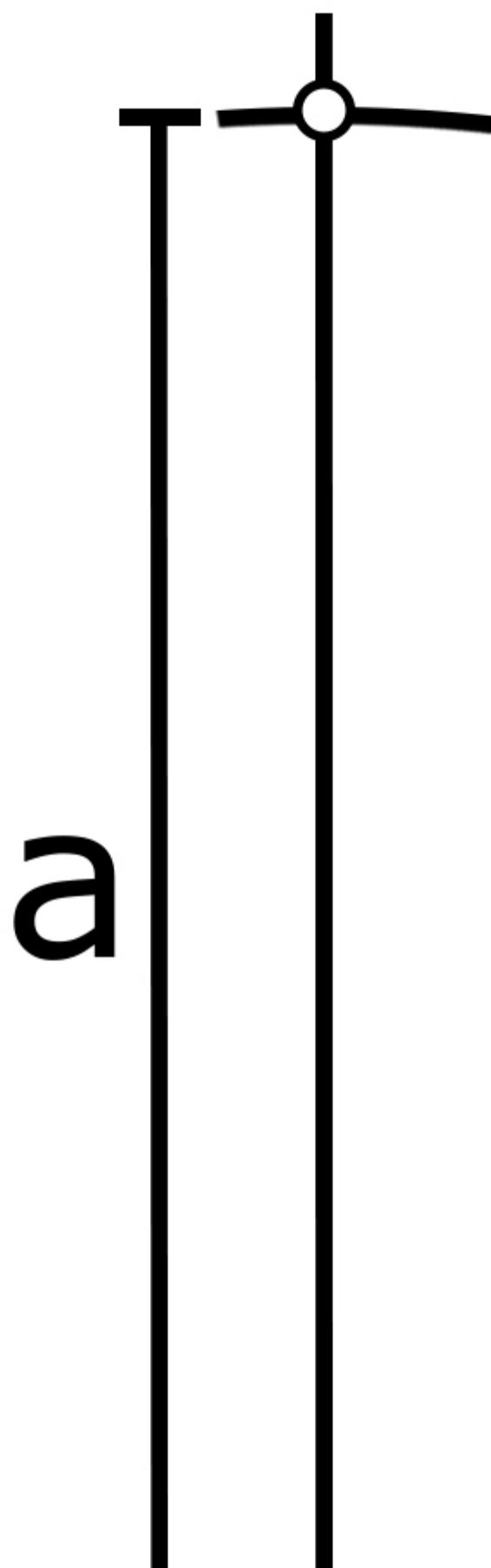
Ahora, con dos arcos de circunferencia de radio r , pero con orígenes en los puntos 2 y 3, se trata de localizar un cuarto punto tal que, trazando una recta entre el punto de origen y esta cuarta marca, el segmento dibujado sea perpendicular a la recta original de tamaño a dibujada al principio. Lo anterior se logra trazando los arcos que marca el punto número 4 en la figura 4.5.



Así, se traza una recta entre el punto 4 y el origen **O** para obtener el segundo lado de la figura geométrica a utilizar, como indica la figura 4.6. Los anteriores pasos se repiten simétricamente en el otro extremo de la recta inicial de tamaño a para obtener el tercer lado del cuerpo geométrico, como muestra la figura 4.7.



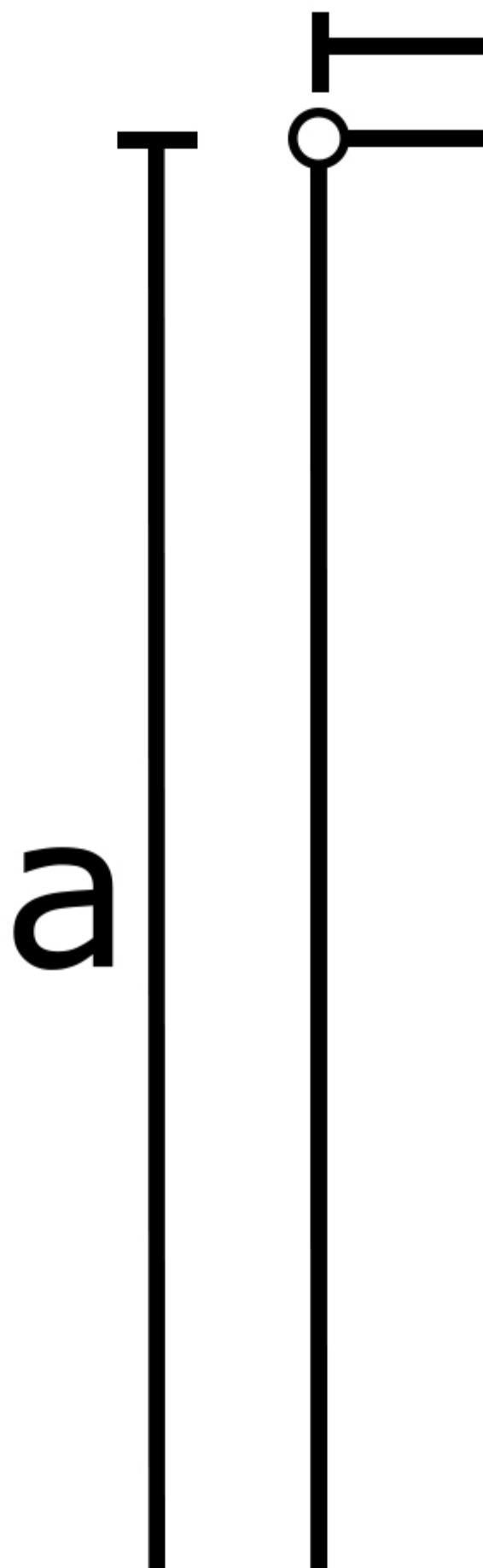
Para garantizar que las medidas de los segmentos obtenidos en estos últimos pasos correspondan con el primer segmento a , se trazará nuevamente otro arco, pero ahora de radio a y centrado en el punto de origen **O**, de forma que el arco a trazar se interseque con el lado perpendicular a la recta y que está sobre el origen, tal y como sugiere la figura 4.8. El segmento acotado a como medida.



El paso anterior se repite de forma simétrica en el otro extremo del primer segmento dibujado, como indica la figura 4.9.



Para obtener el último lado de la figura geométrica, basta con trazar un segmento de recta entre los puntos que acotaron las dos rectas anteriores, como indica la figura 4.10. Ese segmento tendrá a como medida.



Así, se tiene un cuadro de tamaño a en cada lado para poder realizar las mediciones del sistema.

4.3. Procedimiento

Una vez trazada la figura, se identificará la localización de uno de los vértices de la figura en coordenadas de latitud y longitud bajo el marco de referencia terrestre, tomando a Google Maps como herramienta de apoyo. Entonces, la posición del resto de vértices y algunos puntos seleccionados del cuadrado es calculada utilizando la conversión de distancia en metros a coordenadas geográficas.

Una vez verificados los cálculos, el sistema RTK será puesto en marcha con la estación base en un punto cercano con una vista libre del cielo, y el *rover* en el vértice de la figura con posición conocida. Tras el paso de cierto tiempo en que el sistema se estabilice, se comenzará el recorrido de la figura.

En la memoria de la BeagleBone Black, se guardará un registro de las posiciones, calculadas por RTKLIB, del *rover*, mismas que serán utilizadas para determinar el tamaño del cuadrado de acuerdo a la solución de la implementación de Real-Time Kinematics y posteriormente comparado con los cálculos previos a la sesión de experimentación.

4.4. Conclusión

Una vez conocidas las características del experimento a realizar, se procede a la realización de las pruebas, y al análisis de los datos obtenidos.

En el capítulo 5 se mostrarán datos de la rutina seguida y un análisis de la mismo.

Capítulo 5

Resultados

5.1. Introducción

Al final, los resultados obtenidos quedaron de esta manera:

5.2. Conclusión

 Lorem ipsum dolor...

Capítulo 6

Conclusiones

6.1. Introducción

Pepe...

6.2. Resultado

Lorem ipsum dolor...

Apéndice A

Sistema de Archivos en Red entre una PC y una BeagleBone Black

A.1. Introducción

Cuando dos máquinas de diferente arquitectura deben trabajar en un sólo proyecto, suele suceder que es mucho más cómodo realizar código y documentación en una, a pesar de que los archivos estén destinados a ser usados en la otra.

Para ello, se mostrará la forma de configurar un sistema de archivos en red NFS que facilite la tarea de compartir archivos en un directorio.

En este trabajo se mostrará la instalación del sistema en una máquina host en una PC y un cliente en una BeagleBone Black, aprovechando que al conectar mediante USB, se crea una red entre ambas plataformas.

A.2. Definición de NFS

Sistema de archivos en red (NFS, "Network File System" por sus siglas en inglés), es un protocolo que permite acceder mediante una conexión remota a un sistema de archivos [El Manual del Administrador de Debian¹, consultado en Octubre 2016].

En su funcionamiento, permite que un equipo host comparta determinado directorio con otros equipos clientes, pudiendo determinar qué equipos tienen permisos de lectura, escritura o ambas.

A.3. Descarga e instalación

A.3.1. Instalación en host / PC

Bajo Ubuntu en sus últimas versiones en el momento de la redacción de éste documento, se abre una terminal con los comandos *Ctrl + Alt + t*.

Lo primero, es actualizar los repositorios con:

```
$ sudo apt-get update
```

Una vez actualizados, se procede a la instalación de un paquete mediante el siguiente comando:

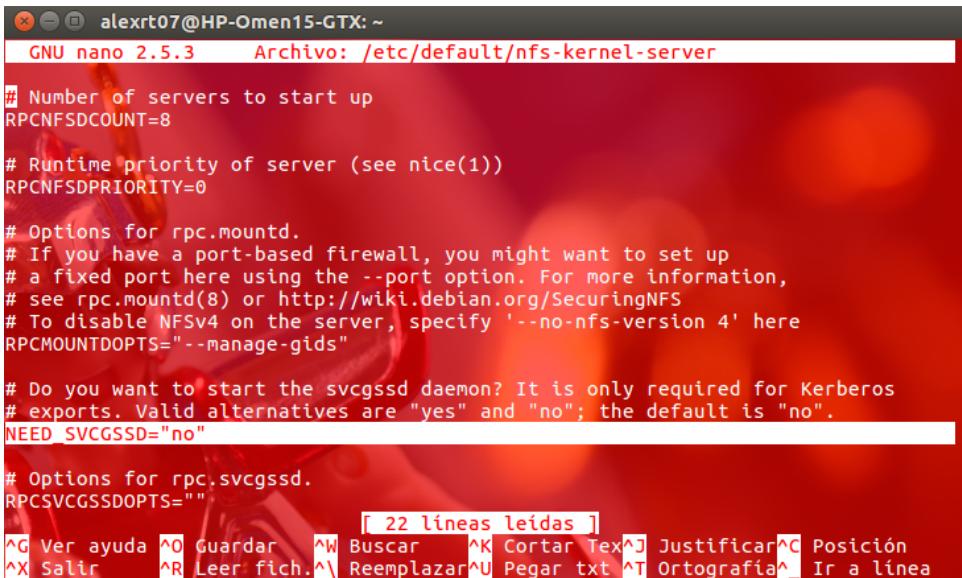
```
$ sudo apt-get install nfs-kernel-server
```

Al finalizar la descarga e instalación, se debe modificar un archivo. Copiar en la terminal el siguiente comando y colocar la contraseña:

```
$ sudo nano /etc/default/nfs-kernel-server
```

¹<https://debian-handbook.info/browse/es-ES/stable/sect.nfs-file-server.html>

Se debe modificar la línea **NEED_SVCGSSD=** y colocar *"no"* en el entrecolumnado, como muestra la figura A.1. Cuando se termine de modificar, guardar con la combinación de teclas *Ctrl + O* y regresar a la terminal con *Ctrl + X*.



```
alexrt07@HP-Omen15-GTX: ~
GNU nano 2.5.3          Archivo: /etc/default/nfs-kernel-server

# Number of servers to start up
RPCNFSDCOUNT=8

# Runtime priority of server (see nice(1))
RPCNFSDPRIORITY=0

# Options for rpc.mountd.
# If you have a port-based firewall, you might want to set up
# a fixed port here using the --port option. For more information,
# see rpc.mountd(8) or http://wiki.debian.org/SecuringNFS
# To disable NFSv4 on the server, specify '--no-nfs-version 4' here
RPCMOUNTDOPTS="--manage-gids"

# Do you want to start the svcgssd daemon? It is only required for Kerberos
# exports. Valid alternatives are "yes" and "no"; the default is "no".
NEED_SVCGSSD="no"

# Options for rpc.svcgssd.
RPCSVCGSSDOPTS=""
```

FIGURA A.1: Archivo /etc/default/nfs-kernel-server ya modificado.

Lo siguiente es abrir el archivo ubicado en /etc/idmapd.conf:

```
$ sudo nano /etc/idmapd.conf
```

Verificar que existan las líneas *Nobody – User = nobody* y *Nobody – Group = nogroup* como muestra la figura A.2.



```
alexrt07@HP-Omen15-GTX: ~
GNU nano 2.5.3          Archivo: /etc/idmapd.conf

[General]
Verbosity = 0
Pipes-Directory = /run/rpc_pipefs
# set your own domain here, if id differs from FQDN minus hostname
# Domain = localdomain

[Mapping]
Nobody-User = nobody
Nobody-Group = nogroup
```

FIGURA A.2: Archivo /etc/idmapd.conf.

Cuando se realiza una conexión con la BeagleBone Black mediante un cable USB, se crea una red con las siguientes direcciones: 192,168,7,2 para la BeagleBone y 192,168,7,1 para el PC host. Entonces, tomando en cuenta lo anterior, se modifica el archivo /etc/exports de la siguiente manera:

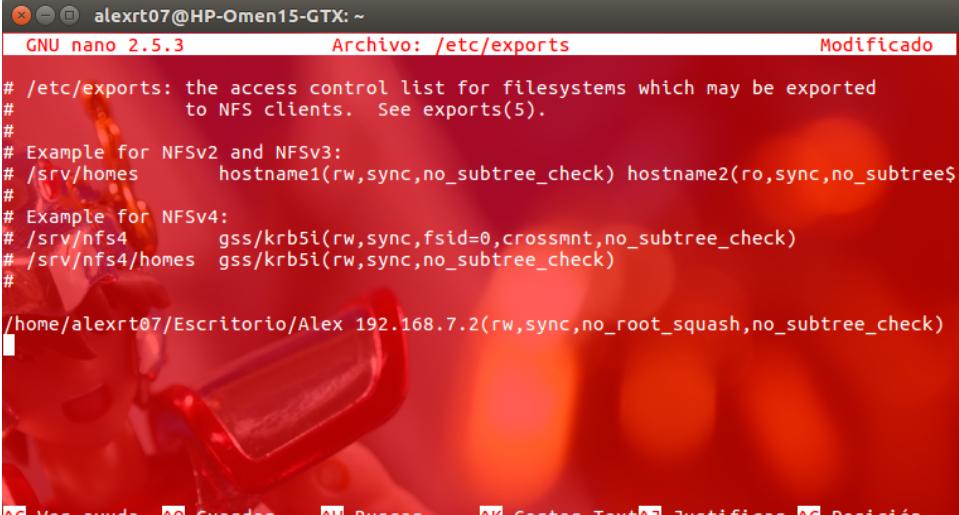
Se abre el archivo con nano, en la terminal:

```
$ sudo nano /etc/exports
```

En el archivo que se abre, añadir la siguiente línea (note que dentro del paréntesis, entre los comandos no existen espacios):

```
/home/alexrt07/Escritorio/Alex 192.168.7.2(rw,sync,  
no_root_squash,no_subtree_check)
```

Donde /home/alexrt07/Escritorio/Alex es el directorio a compartir y 192.168.7.2 es la dirección ip de la BeagleBone. Así, el archivo queda como muestra la figura A.3.



```
alexrt07@HP-Omen15-GTX: ~
GNU nano 2.5.3                               Archivo: /etc/exports                         Modificado
# /etc/exports: the access control list for filesystems which may be exported
#                   to NFS clients. See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes      hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree$#
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/home/alexrt07/Escritorio/Alex 192.168.7.2(rw,sync,no_root_squash,no_subtree_check)

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Texto ^J Justificar ^C Posición
^X Salir    ^R Leer fich. ^\ Reemplazar ^U Pegar txt ^T Ortografía ^L Ir a linea
```

FIGURA A.3: Archivo /etc/exports.

Finalmente, resta reiniciar el servidor con el siguiente comando:

```
$ /etc/init.d/nfs-kernel-server restart
```

Se deberá mostrar una confirmación de reinicio exitoso.

Seguridad

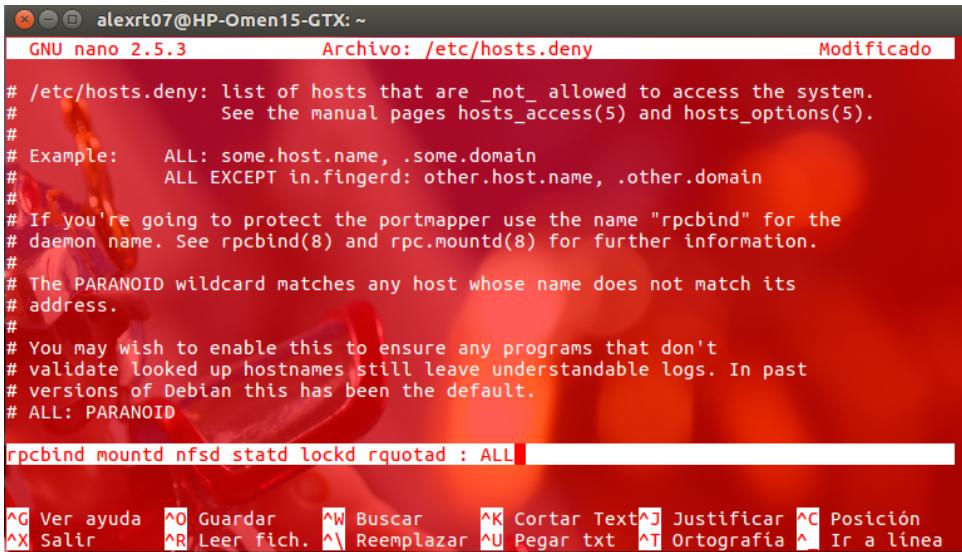
Para evitar dejar hoyos de seguridad de acceso a los archivos personales, es altamente recomendable modificar los archivos /etc/hosts.deny y /etc/hosts.allow para permitir acceso solamente a los clientes conocidos.

Abrir el archivo /etc/hosts.deny con:

```
$ sudo nano /etc/hosts.deny
```

Y añadir la siguiente línea:

```
rpcbind mountd nfsd statd lockd rquotad : ALL
```



```

alexrt07@HP-Omen15-GTX: ~
GNU nano 2.5.3          Archivo: /etc/hosts.deny          Modificado
# /etc/hosts.deny: list of hosts that are _not_ allowed to access the system.
# See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: some.host.name, .some.domain
#              ALL EXCEPT in.fingerd: other.host.name, .other.domain
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#
# The PARANOID wildcard matches any host whose name does not match its
# address.
#
# You may wish to enable this to ensure any programs that don't
# validate looked up hostnames still leave understandable logs. In past
# versions of Debian this has been the default.
# ALL: PARANOID

rpcbind mountd nfsd statd lockd rquotad : ALL

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Texto ^J Justificar ^C Posición
^X Salir   ^R Leer fich. ^V Reemplazar ^U Pegar txt ^T Ortografía ^L Ir a línea

```

FIGURA A.4: Archivo /etc/hosts.deny

Como muestra la figura A.4. Ahora, abrir el archivo /etc/hosts.allow con el comando:

```
$ sudo nano /etc/hosts.allow
```

Y añadir la siguiente línea:

```
rpcbind mountd nfsd statd lockd
rquotad : 192.168.7.2 127.0.0.1
```

Como muestra la figura A.5.



```

alexrt07@HP-Omen15-GTX: ~
GNU nano 2.5.3          Archivo: /etc/hosts.allow
# /etc/hosts.allow: list of hosts that are allowed to access the system.
# See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: LOCAL @some_netgroup
#              ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#
rpcbind mountd nfsd statd lockd rquotad : 192.168.7.2 127.0.0.1

[ 11 líneas escritas ]

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Texto ^J Justificar ^C Posición
^X Salir   ^R Leer fich. ^V Reemplazar ^U Pegar txt ^T Ortografía ^L Ir a línea

```

FIGURA A.5: Archivo /etc/hosts.allow

Reiniciar el servidor con:

```
$ service nfs-kernel-server restart
```

Ahora, el PC está preparado para compartir vía red el directorio:

```
/home/alexrt07/Escritorio/Alex/
```

A.3.2. Instalación en cliente / BeagleBone

Asumiendo que la BeagleBone Black tiene un Debian con su archivo */etc/apt/sources.list* correctamente configurado, ejecutamos en la terminal de nuestro host para conectarnos:

```
$ ssh -l root 192.168.7.2
```

donde *ssh* es el comando para conectarse por el protocolo secure shell, *-l* es el comando que indica que se hará un login con el usuario *root*, y *192.168.7.2* es la dirección IP de la BeagleBone en la red que se creó a través del cable USB.

Entonces, una vez hecho el loggin, ejecutar:

```
$ apt-get install nfs-common
```

Que instalará y preconfigurará los archivos necesarios para una correcta comunicación a través de NFS.

Es recomendable crear un directorio en donde se montarán los archivos que compartirá con la PC:

```
$ mkdir /home/debian/Alex_tesista
```

A.4. Ejecución

Se procede a montar el sistema de archivos con el siguiente comando:

```
$ mount -t nfs -o proto=tcp, port=2049  
192.168.7.1:/home/alexrt07/ARM-Root/home/Alex  
/home/debian/Alex_tesista/
```

En donde *mount* es el comando para montar el directorio, *-t nfs* indica que se trata de un sistema de archivos por red, *-o proto=tcp, port=2049* indica que se utilizará el protocolo de transferencia de archivos a través del puerto 2049 (mirar el manual de nfs en su página 5 con **\$ man 5 nfs** para más opciones e información), *192.168.7.1*: es la dirección IP de la PC host, */home/alexrt07/ARM-Root/home/Alex* es el directorio que contiene los archivos a compartir, y */home/debian/Alex_tesista/* es el directorio creado en donde se encontrarán los archivos.

Para cerrar esta conexión, utilice

```
$ umount /home/debian/Alex_tesista/
```

Tome en cuenta que al finalizar la conexión, no se mantendrán los archivos compartidos por nfs. Desaparecerán y contendrá los archivos originales que esa carpeta contenía antes de montar el sistema de archivos por red.

Apéndice B

Máquina Virtual con Sistema de Arquitectura ARM en PC

B.1. Introducción

Los equipos de cómputo actuales se pueden categorizar de acuerdo a su arquitectura. Se define a la *arquitectura de una computadora* como la estructura operacional de este dispositivo. Incluye los formatos de información, conjunto de instrucciones y técnicas de direccionamiento de memoria [Mano, M. MORRIS (1994). *Arquitectura de Computadoras*. Pearson Educación].

Entre las arquitecturas más conocidas están la x86, la x86_64 o AMD64, ARM, PowerPC, entre otras. Un programa codificado para cierta arquitectura no podrá ser funcional en otra, salvo que la misma explícitamente indique la compatibilidad (por ejemplo, el caso de x86_64, que permite la ejecución de programas codificados en x86, por ser sólo una extensión de este último).

Cuando en un determinado proyecto coexisten dispositivos de cómputo de distinta arquitectura, es lógico pensar que alguna puede ser no muy cómoda para programar grandes cantidades de líneas de código, o bien, requiere de mucho tiempo al momento de compilar y obtener un ejecutable. En estos casos, toma sentido el ser capaz de compilar todo el código de determinado equipo. Esto es posible mediante la emulación de un sistema con determinada arquitectura, en otro equipo no necesariamente con la misma configuración.

En este anexo, se mostrarán los pasos a realizar para emular un dispositivo BeagleBone (con sistema operativo GNU/Linux Debian de arquitectura ARM) en una computadora convencional (con sistema operativo Ubuntu de arquitectura x86_64).

B.2. Software a utilizar

B.2.1. QEMU



FIGURA B.1: Logotipo del software QEMU.

QEMU es un emulador y virtualizador de máquinas, de código abierto. Cuando es usado como emulador, puede correr sistemas operativos y programas hechos para determinada arquitectura. Tiene un buen rendimiento por usar lo que llama *traducción dinámica*.

B.2.2. Debootstrap

Debootstrap es una herramienta cuyo propósito es la instalación de un sistema basado en Debian en un directorio determinado de un sistema ya instalado. Posee la capacidad de instalar un sistema de diferente arquitectura a la máquina *host* o anfitriona.

B.3. Descarga e instalación

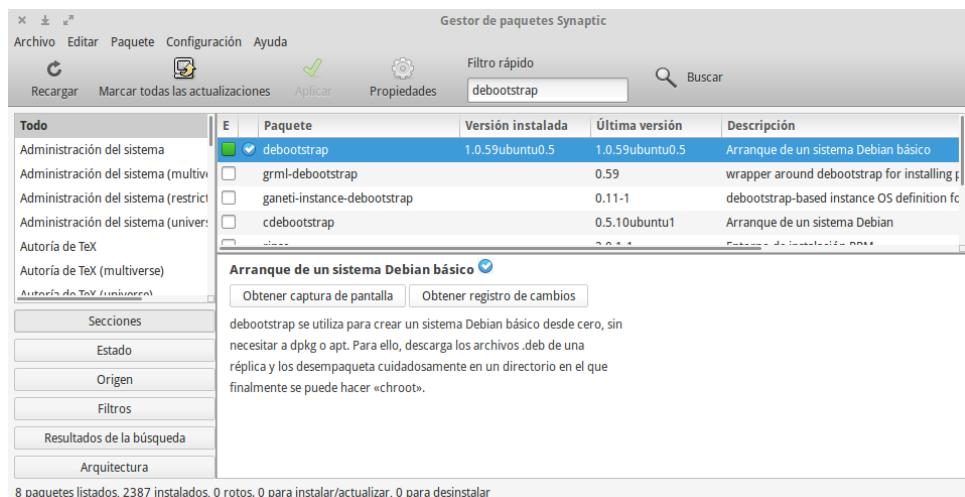


FIGURA B.2: Deboostrap dentro de los repositorios.

Utilizando Ubuntu como sistema host, las herramientas ya mencionadas pueden ser encontradas dentro de los repositorios oficiales del sistema, como se muestra en la figura B.2. Pueden ser instalados por consola o por cualquier gestor de paquetes. Se sugiere actualizar los repositorios mediante el comando:

```
$ sudo apt-get update
```

Una vez actualizados, se procede a la instalación de los repositorios:

```
$ sudo apt-get install debootstrap qemu-user-static
```

El proceso durará dependiendo de la velocidad de descarga del internet. Al finalizar, se deberá solicitar permiso de *root* al sistema:

```
$ sudo -s
```

Ahora, se deberá crear el directorio donde se almacenarán los archivos del sistema que se emulará. Se sugiere que el nombre de dicha carpeta haga referencia al tipo de sistema que contendrá. En este caso, se usará el nombre *ARM-Root*.

```
$ mkdir ~/ARM-Root
```

Al finalizar, se procederá a la instalación del sistema:

```
$ debootstrap --foreign --arch=armhf stable ARM-Root
$ cp /usr/bin/qemu-arm-static ARM-Root/usr/bin
$ chroot ARM-Root /debootstrap/debootstrap --second-stage
```

El proceso tardará nuevamente dependiendo de la velocidad del internet. Se sugiere estar pendiente del proceso de descarga e instalación por cualquier error que surgiera durante el mismo¹.

Al finalizar, salga del modo *root* con :

```
$ exit
```

Ya dispone de un sistema Debian ARM totalmente funcional, pudiendo acceder mediante el comando:

```
$ sudo chroot ~/ARM-Root /bin/bash
```

B.3.1. Configuración del sistema

Se abre una terminal. En ella, ingrese los comandos para acceder al sistema ARM:

```
$ sudo chroot ~/ARM-Root /bin/bash
```

```
sudo chroot ~/ARM-Root /bin/bash
...
root@Dell-Inspiron23:/# uname -a
Linux Dell-Inspiron23 3.16.0-73-generic #95~14.04.1-Ubuntu SMP Thu
Jun 9 08:00:04 UTC 2016 armv7l GNU/Linux
root@Dell-Inspiron23:/#
```

FIGURA B.3: Salida del comando *uname*.

La terminal se debe mostrar como un usuario root dentro del mismo equipo que usted utiliza. Sin embargo, es un sistema de diferente arquitectura. Escriba el siguiente comando para corroborarlo:

```
$ uname -a
```

El sistema entregará una salida que contiene el nombre del equipo, versión del kernel, entre otros. Lo interesante está en la salida del tipo de arquitectura del sistema, que dice *armv7l*, como se muestra en la figura B.3, lo que indica que la instalación fue exitosa. En la ejecución de una terminal común, sin haber ejecutado el comando *chroot*, la salida ofrecerá en el apartado de arquitectura, a *x86_64* ó *x86*, dependiendo del sistema anfitrión.

Ahora, conviene actualizar este sistema ARM e instalar los repositorios necesarios que se van a utilizar en el proyecto actual en la tarjeta ARM física original. Para hacer esto, se verifica la versión de Debian instalada mediante el comando:

```
$ cat /etc/debian_version
```

¹En ocasiones, suelen salir mensajes de que el archivo a descargar no se encuentra ya en el servidor, pero que lo intentará en otro servidor espejo. Es un mensaje normal y no requiere de la atención del usuario.

```
sudo chroot ~/ARM-Root/ /bin/bash
...
root@Dell-Inspiron23:/# cat /etc/debian_version
8.5
root@Dell-Inspiron23:/#
```

FIGURA B.4: Versión del sistema instalado.

Teniendo el número de versión, se debe conocer con qué nombre fue "bautizada" dicha versión de la distribución. Durante la escritura de este documento, la versión instalada fue Debian Jessie (versión 8.*), como se muestra en la figura B.4. El archivo `/etc/apt/sources.list` es un archivo vacío y en su contenido se debe indicar las direcciones en donde debe buscar por paquetes. Es dependiente de la versión instalada.

En la web [LinuxConfig.org²](https://linuxconfig.org/debian-apt-get-jessie-sources-list) se encuentran listadas las entradas para este archivo en Debian Jessie.

Para este caso particular, se colocarán las entradas para las actualizaciones de seguridad, y del espejo mexicano. Se hace mediante la entrada del comando:

```
$ nano /etc/apt/sources.list
```

```
GNU nano 2.2.6      File: /etc/apt/sources.list      Modified

# /etc/apt/sources.list :
#Security-Updates
deb http://security.debian.org/ jessie/updates main contrib non-free
deb-src http://security.debian.org/ jessie/updates main contrib non-free

#Mexican-Mirror
deb http://ftp.mx.debian.org/debian/ jessie main contrib non-free
deb-src http://ftp.mx.debian.org/debian/ jessie main contrib non-free
```

FIGURA B.5: Contenido del archivo `/etc/apt/sources.list`.

Se vuelve a enfatizar que el contenido de este archivo es totalmente dependiente de la versión de Debian instalada, ya que colocar las fuentes para un sistema diferente puede comprometer el sistema de paquetes de Debian. Una vez colocadas las entradas como muestra la figura B.5, se procede a actualizar la lista de repositorios mediante el comando:

```
$ apt-get update
```

El proceso puede tardar dependiendo de la cantidad de cosas a descargar y de la velocidad a internet. Una vez terminado, se procede a la instalación de los paquetes necesarios para el proyecto en el que se tenga que utilizar este sistema.

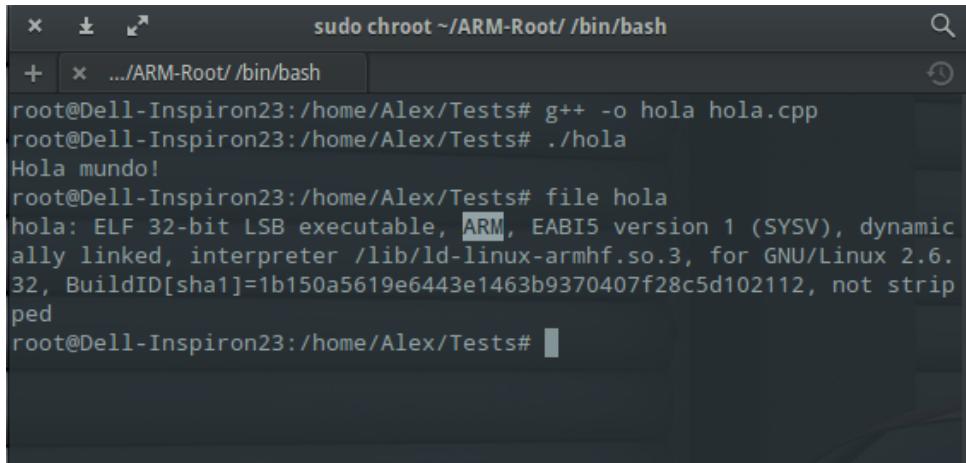
²<https://linuxconfig.org/debian-apt-get-jessie-sources-list>

B.3.2. Ejemplo: compilando un programa en C++

Para instalar los compiladores de c y c++, se usa el comando:

```
$ apt-get install build-essential
```

El sistema mostrará la cantidad de bytes a descargar y pregunta si se desea proseguir. Con la tecla y/Y se le indica que continúe y mostrará la descarga. Al acabar, el sistema ya será capaz de compilar programas en los lenguajes ya mencionados.



The screenshot shows a terminal window titled "sudo chroot ~/ARM-Root/ /bin/bash". The terminal content is as follows:

```
root@Dell-Inspiron23:/home/Alex/Tests# g++ -o hola hola.cpp
root@Dell-Inspiron23:/home/Alex/Tests# ./hola
Hola mundo!
root@Dell-Inspiron23:/home/Alex/Tests# file hola
hola: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux-armhf.so.3, for GNU/Linux 2.6.32, BuildID[sha1]=1b150a5619e6443e1463b9370407f28c5d102112, not stripped
root@Dell-Inspiron23:/home/Alex/Tests#
```

FIGURA B.6: Compilación, ejecución y verificación de la arquitectura en que fue compilado un programa de c++.

A modo de prueba, se codificó un archivo en c++ que imprima *Hola mundo*, llamado *hola.cpp* y se procedió a compilarlo y ejecutarlo:

```
$ g++ -o hola hola.cpp
$ ./hola
```

Mediante el comando *file*³, se procedió a verificar que el archivo estaba codificado para arquitectura ARM, como indica la figura B.6. Esto significa que el ejecutable *hola* puede ser ejecutado perfectamente en una tarjeta BeagleBone, Raspberry o similares, de arquitectura ARM con sistema GNU/Linux. Como nota, este archivo no podrá ser ejecutado por una terminal de la computadora host sin haber ejecutado el comando *chroot* mencionado al principio de la subsección B.3.1, en la página 63, debido a que su arquitectura no soporta la codificación de dicho archivo.

³ Instalable mediante: \$ apt-get install file

Apéndice C

Programación de módulos XBEE

C.1. Introducción

Una incorrecta configuración de los módulos XBEE conlleva a que éstos sean incapaces de establecer comunicación, y a manera de evitar dicho escenario, la presente guía se ofrece como apoyo para la correcta configuración de los dispositivos XBEE en una topología denominada Par.

C.2. El software XCTU

El fabricante, Digi International Inc., proporciona un software como herramienta para la configuración y puesta a punto de los dispositivos XBEE, llamado XCTU. Está disponible para funcionar en Windows, GNU-Linux (ambas tanto en arquitecturas 32 y 64 bits) y MacOS X.

C.2.1. Descarga e instalación

El software está habilitado para su descarga desde la página oficial de [Digi International, Inc.](#)¹, mostrada en la figura C.1.

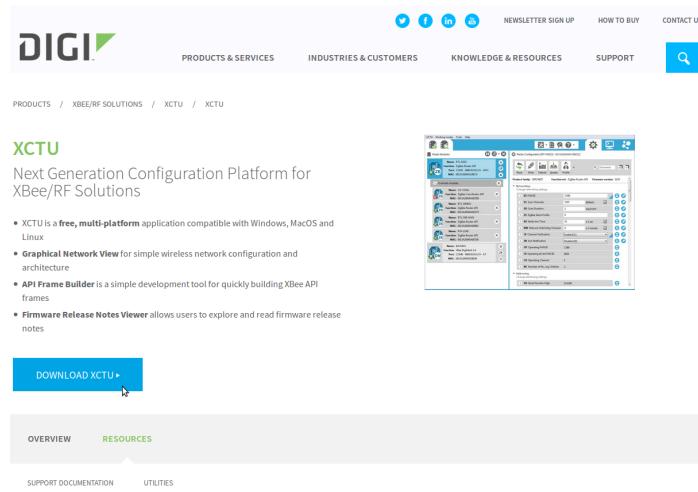


FIGURA C.1: Página de descarga de XCTU.

Se debe dar clic al botón "Download XCTU", y después, escoger la versión a instalar de entre la lista que aparece, dependiendo del sistema operativo y arquitectura en uso. Note que también se ofrece a descargar una versión llamada "Legacy", que es

¹<http://www.digi.com/products/xbee-rf-solutions/xctu-software/xctu#resources>

conocida como la versión antigua de XCTU, disponible solamente para Windows. En esta guía, se mostrará la configuración de un XBEE para la versión "nueva" de XCTU bajo Windows con arquitectura de 64 bits. Aparecerá una ventana que pide ingresar datos para un registro, que es opcional. Dar clic en el link **No thanks, register later** en la parte inferior de la página, o llene los datos y regístrese si así lo desea. La descarga iniciará en breve.

Para la instalación, una vez concluida la descarga, se debe ubicar en la carpeta de destino de la misma el archivo que se observa en la figura C.2.

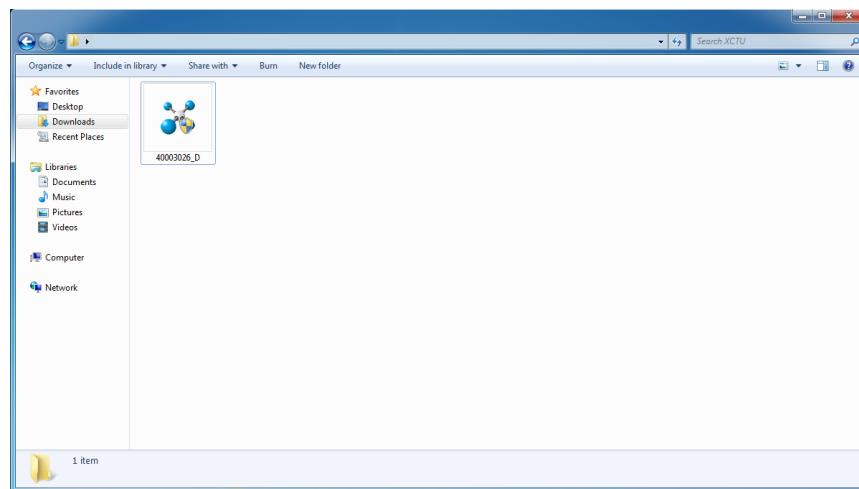


FIGURA C.2: Instalador del programa.

Al ejecutar dicho archivo, el instalador mostrará la pantalla de bienvenida como se muestra en la figura C.3 de la página 68. Pulse el botón *Next*.

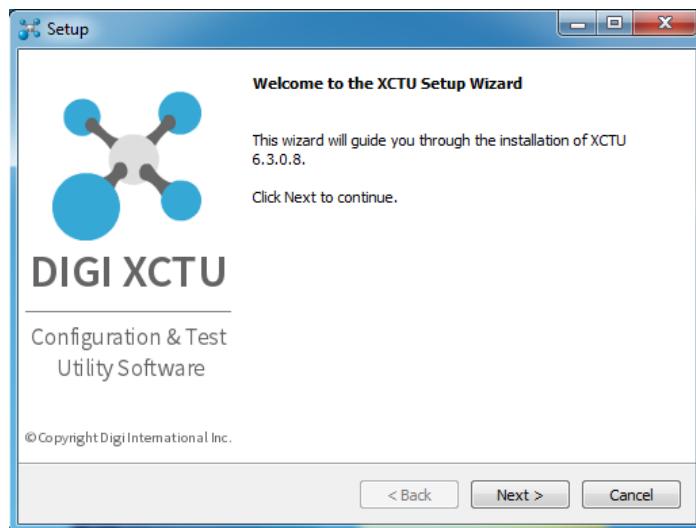


FIGURA C.3: Ventana principal del instalador de XCTU.

Se presentarán al usuario los términos y condiciones de uso del software, mismos que deberán ser aceptados para continuar utilizando el software. Tras ello, dar clic en el botón *Next*. En la siguiente ventana, el usuario deberá indicar la carpeta donde el software se instalará, como se muestra en la figura C.4 dar clic en el botón *Next*.

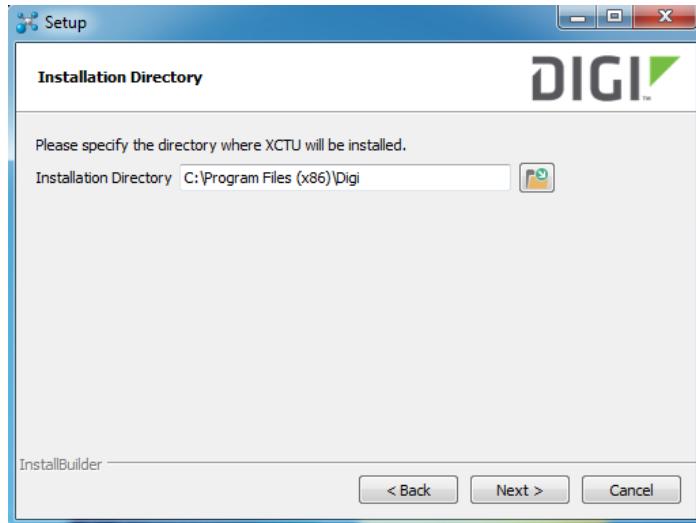


FIGURA C.4: Carpeta de destino de la instalación.

La copia de archivos tendrá lugar y habrá que esperar a que la misma termine, siendo la ventana del instalador quien se actualizará mostrando el progreso y después el aviso de que ha terminado. Una vez que dicho aviso aparezca, dar clic en el botón *Next*. El proceso de instalación habrá terminado con una ventana como la de la figura C.5 y solo resta dar clic en el botón *Finish*.

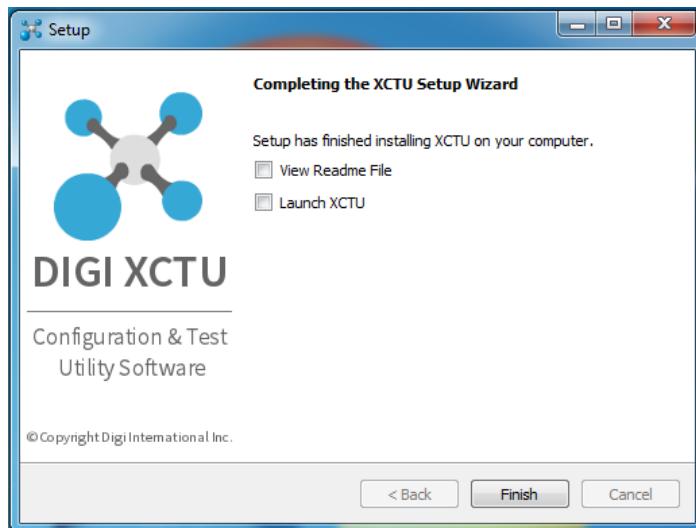


FIGURA C.5: El instalador informa que el proceso ha terminado correctamente.

C.2.2. Apertura del programa

El software estará disponible para su ejecución siguiendo la siguiente ruta: Menú Inicio > Todos los programas > Digi > XCTU > XCTU.exe, como se muestra en la figura C.6 de la página 70.

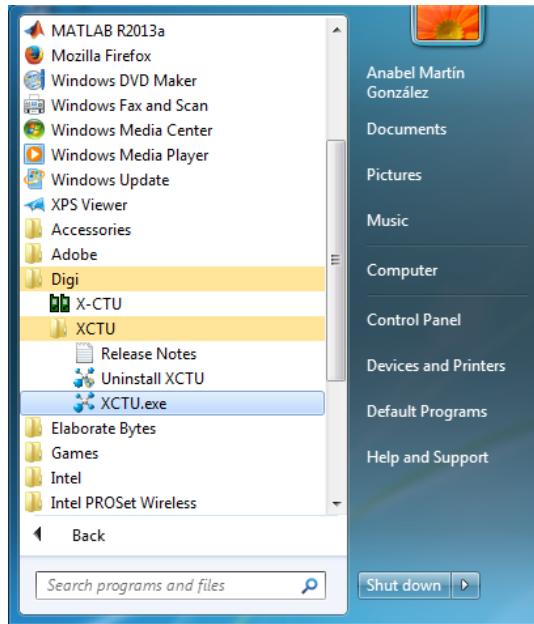


FIGURA C.6: Archivo de inicio del software XCTU.

El programa cargará los archivos necesarios hasta desplegar la ventana principal del software, mostrado en la figura C.7.

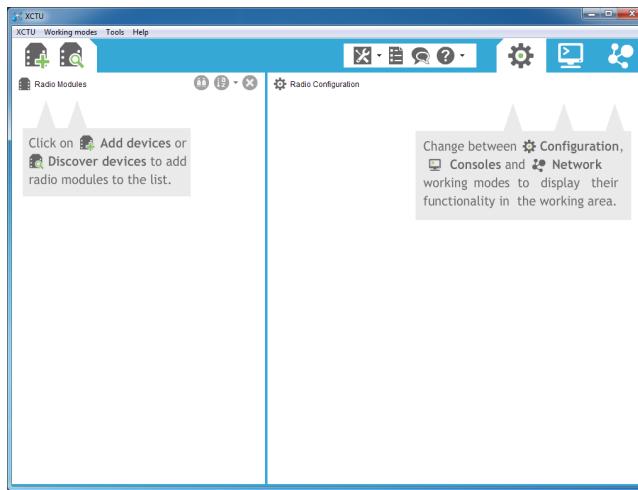


FIGURA C.7: Ventana principal de XCTU.

C.3. Conexión y programación de los XBEE

Para conectar los XBEE a la PC, es necesario contar con un dispositivo que permita la comunicación serial mediante puertos USB, como el que se muestra en la figura C.8. Existen distintos modelos en el mercado que varían en precio como en diseño, pero al final la funcionalidad es la misma. El XBEE deberá ser colocado con cuidado sobre la placa.

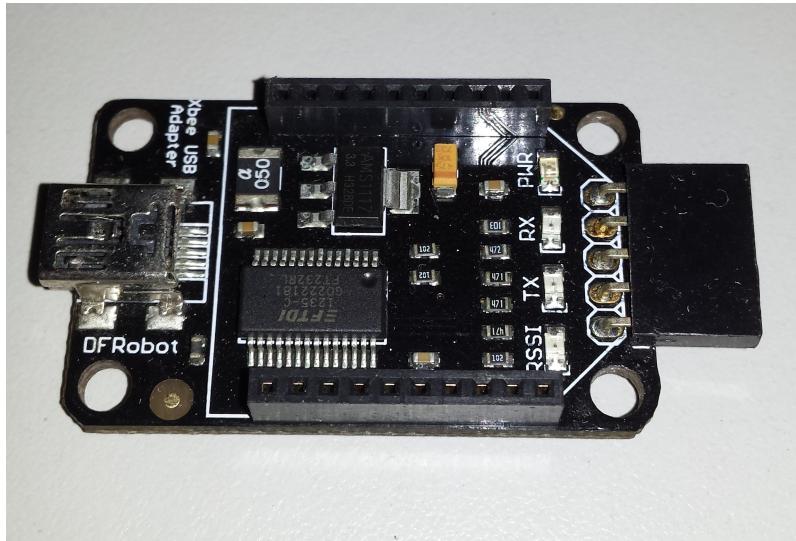


FIGURA C.8: Dispositivo que permite comunicación entre el XBEE y la PC.

Al conectarle con el cable a la PC, el sistema operativo comenzará con la instalación de los controladores necesarios, desplegando una ventana como la que se muestra en la figura C.9 una vez sea realizada con éxito.

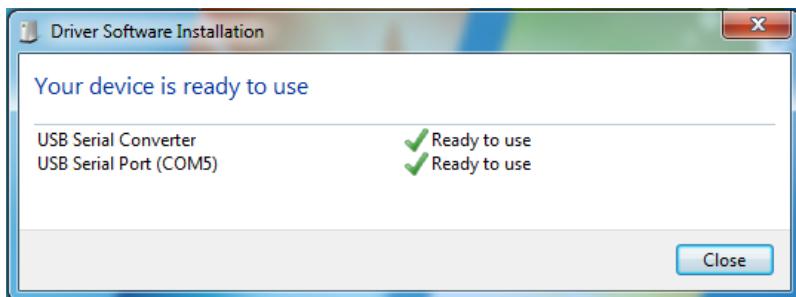


FIGURA C.9: Ventana que indica una correcta instalación de drivers.

En caso de que la ventana muestre que Windows no encontró los drivers, será necesario que se refiera a la sección C.3.1. En otro caso, puede saltarse esta guía hasta la sección C.3.2.

C.3.1. Instalación de controladores

Requerimientos para la instalación

Si la ventana de instalación de drivers despliega el siguiente mensaje: "FT232R USB UART No driver found", entonces debe visitar la página web de **FTDI CHIP²**, que contiene una tabla como la que se muestra en la figura C.10.

²<http://www.ftdichip.com/Drivers/VCP.htm>

Currently Supported VCP Drivers									
Operating System	Release Date	Processor Architecture						Comments	
		x86 (32-bit)	x64 (64-bit)	PPC	ARM	MIPSII	MIPSIV	SH4	
Windows*	2016-02-02	2.12.14	2.12.14	-	-	-	-	-	2.12.14 is WHQL Certified. Release Notes
Linux	2009-05-14	1.5.0	1.5.0	-	-	-	-	-	All FTDI devices now supported in Ubuntu 11.10, kernel 3.0.5+19 Refer to TN-105 if you need a custom VCP VID/PID in Linux
Mac OS X 10.3 to 10.8	2012-08-10	2.2.18	2.2.18	2.2.18	-	-	-	-	Refer to TN-105 if you need a custom VCP VID/PID in Mac OS
Mac OS X 10.9 and above	2015-04-15	-	2.3	-	-	-	-	-	This driver is signed by Apple
Windows CE 4.2/5.2**	2012-01-06	1.1.0.20	-	-	1.1.0.20	1.1.0.10	1.1.0.10	1.1.0.10	For use of the CAT files supplied for ARM and x86 builds refer to AN_319
Windows CE 6.0/7.0	2012-01-06	1.1.0.20 CE 6.0 CAT CE 7.0 CAT	-	-	1.1.0.20 CE 6.0 CAT CE 7.0 CAT	1.1.0.10	1.1.0.10	1.1.0.10	BETA VCP Driver Support for WinCE2013
Windows CE 2013	2015-03-06	BETA	-	BETA	-	-	-	-	

FIGURA C.10: Tabla de controladores.

Se seleccionará el link correspondiente a la celda cuyos encabezados indiquen sus correspondientes sistema operativo y arquitectura. Se iniciará un proceso de descarga. Es necesario conocer el directorio donde estarán ubicados estos archivos, dado que deberán ser descomprimidos y luego indicados explícitamente en un asistente de Windows. Ahora, deberá abrir el *Menú Start*, dar clic derecho a *Computer* y seleccionar *Manage*, como se indica en la figura C.11 de la página 72. Este último paso requiere permisos de administrador. Si usted no los posee, contacte con el administrador del sistema.

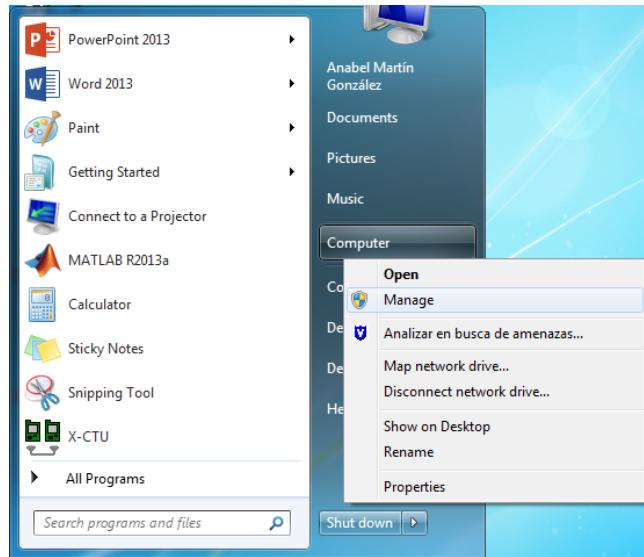


FIGURA C.11: Ubicación del administrador de dispositivos.

Se abrirá una ventana. En el listado del lado izquierdo, seleccionar *Device Manager*, entonces la pantalla se mostrará como indica la figura C.12.

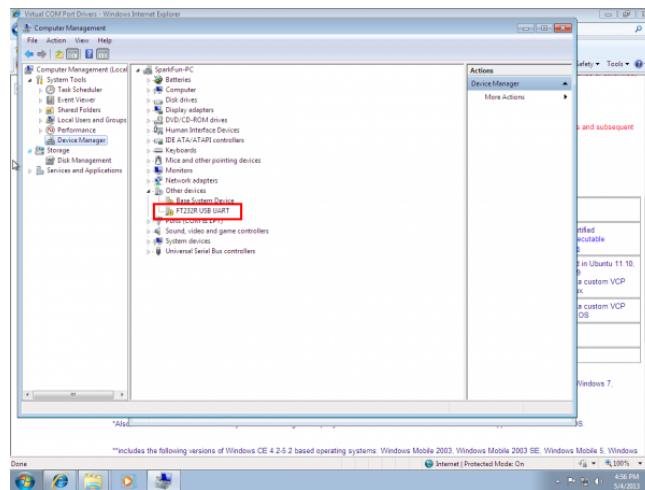


FIGURA C.12: Administrador de Dispositivos. Imagen tomada de [Sparkfun](#).³

Instalación

Debe encontrar el dispositivo cuyo controlador no pudo ser instalado (*FT232R USB UART*) en la sección de *Other devices*, por lo que se deberá dar clic derecho en él y seleccionar la opción *Update Driver Software*, como indica la figura C.13.

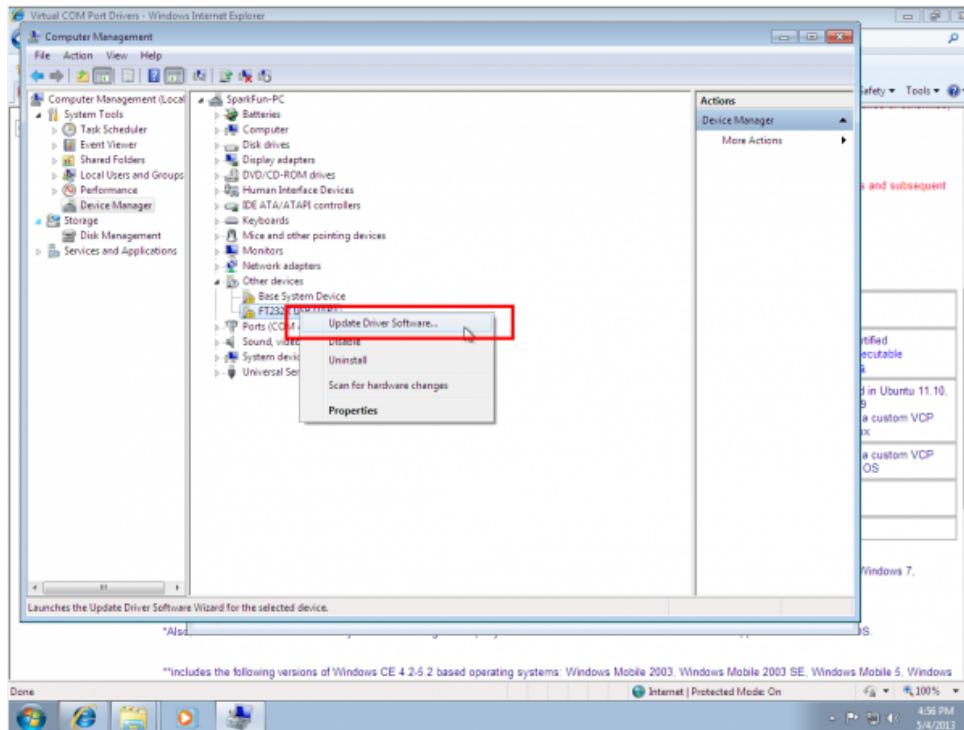


FIGURA C.13: Actualizar controlador.

³<https://learn.sparkfun.com/tutorials/how-to-install-ftdi-drivers/all>. A partir de aquí, el resto de las figuras de la sección C.3.1 serán tomadas de dicha web salvo que se indique lo contrario.

La ventana que ahora aparece es el asistente de instalación de los controladores descargados en la tabla de la figura C.10. Seleccionar **Browse my computer for driver software**, como se indica en la figura C.14 de la página 74.

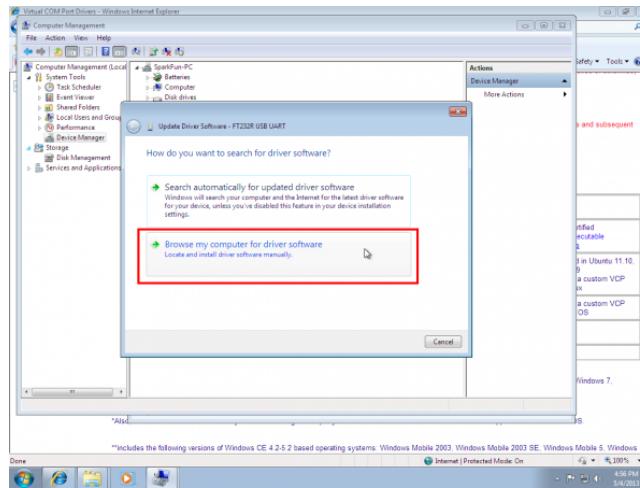


FIGURA C.14: Asistente de instalación del controlador.

Es entonces en la ventana que ahora debe mostrarse, que se accederá a la carpeta que se creó tras descomprimir la descarga del controlador. Se hará dando clic al botón **Browse** y se navegará hasta seleccionarla en la ventana emergente. Clic en **OK**, como se muestra en la figura C.15.

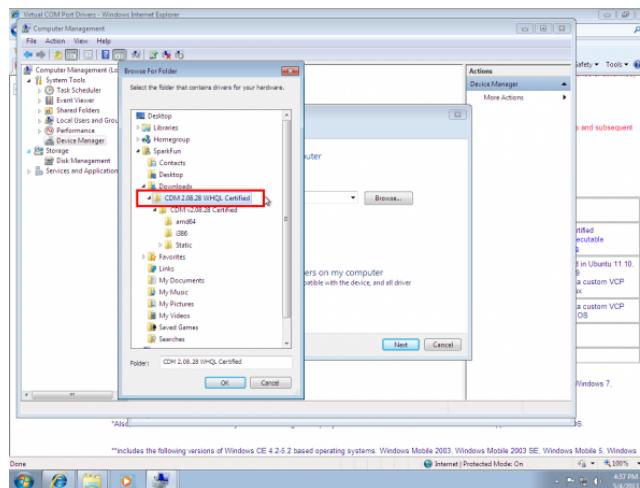


FIGURA C.15: Carpeta del controlador.

Verifique también que la opción *Include subfolders* esté palomeada como en la figura C.16 de la página 75, y dar clic al botón **Next**.

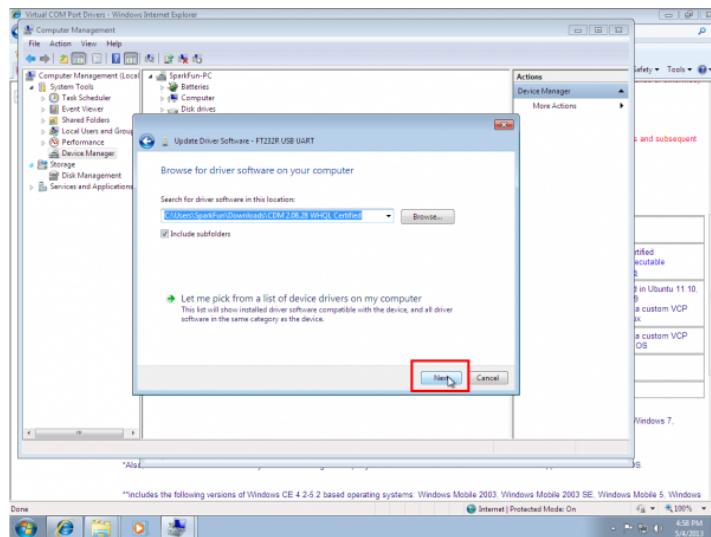


FIGURA C.16: Selección de carpeta.

Tras un proceso que tomará algunos segundos, se mostrará una ventana de éxito en la instalación, tal y como se ve en la figura C.17, y el asistente podrá cerrarse.

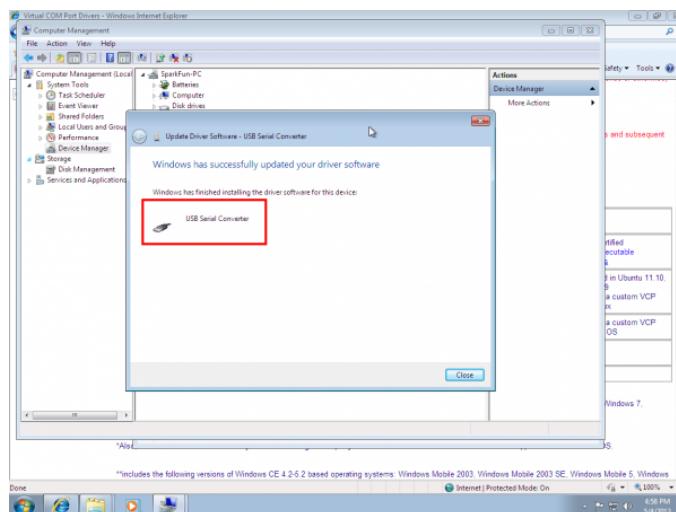


FIGURA C.17: Instalación exitosa.

Sin embargo, la instalación aún no está terminada, por lo que el proceso deberá repetirse desde el inicio de esta subsección C.3.1, pero ahora, en vez de actualizar el driver de *FT232R USB UART*, se hará con el de *USB Serial Port*, como se observa en la figura C.18 usando nuevamente la misma carpeta de controladores que antes se descomprimió.

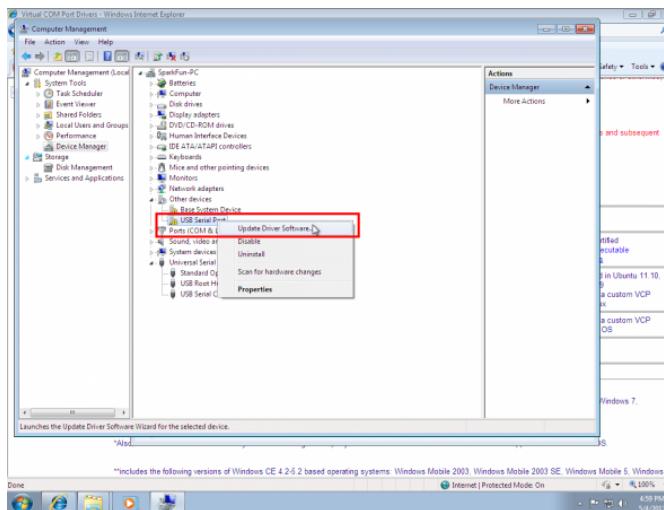


FIGURA C.18: Segunda instalación.

Al finalizar, se mostrará un dispositivo conectado como un puerto serial y el número asignado al mismo, como se muestra en la figura C.19. Tomar nota del número de puerto asignado al XBEE.

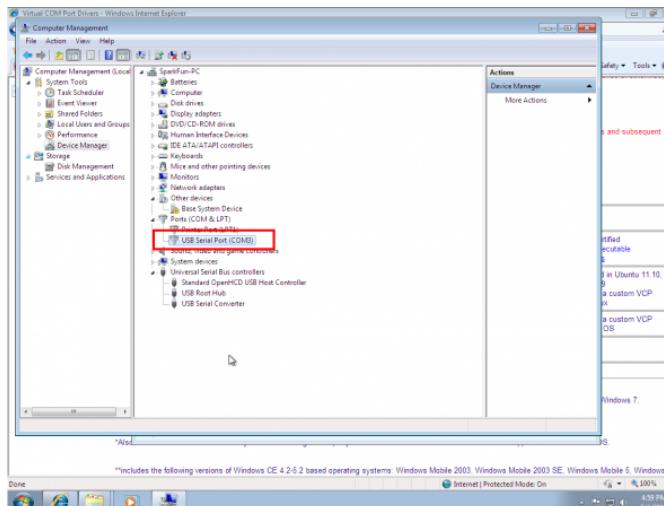


FIGURA C.19: Instalación exitosa de los controladores.

No será necesario repetir esta subsección C.3.1 cada que un XBEE se conecte, ya que a partir de ahora, todos los dispositivos XBEE serán reconocidos automáticamente como una interfaz serial.

Este proceso de instalación fue tomado de los tutoriales de Sparkfun⁴.

C.3.2. Reconocimiento del XBEE con XCTU

Identificación del puerto

Ante todo, es necesario reconocer mediante qué puerto un XBEE está conectado a la PC. Por lo tanto, se deberán, seguir los siguientes pasos: abrir el menú *Start*, clic

⁴<https://learn.sparkfun.com/tutorials/how-to-install-ftdi-drivers/all> con último acceso el 3 de Febrero de 2016

derecho en *Computer*, clic en *Manage*, tal y como se muestra en la figura C.20. Se requieren permisos de administrador para acceder a esta ventana.

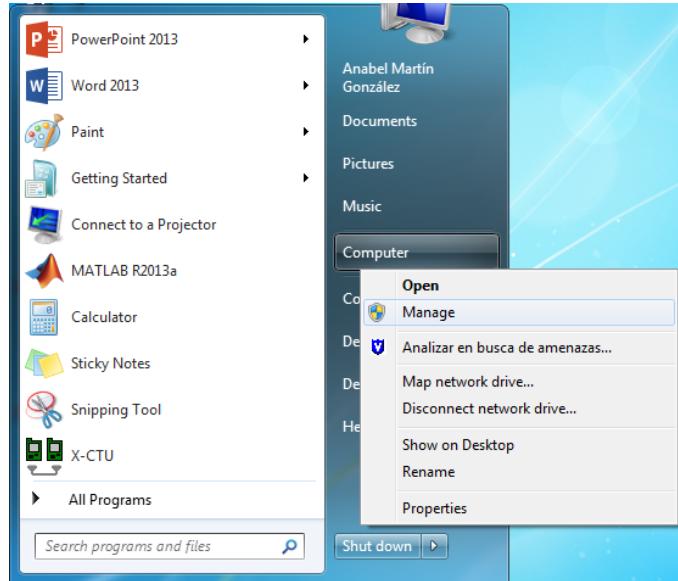


FIGURA C.20: Ubicación del administrador de dispositivos.

En la ventana que se abre, en el menú de la izquierda, dar clic a *Device Manager*, tal como se observa en la figura C.21 de la página 77.

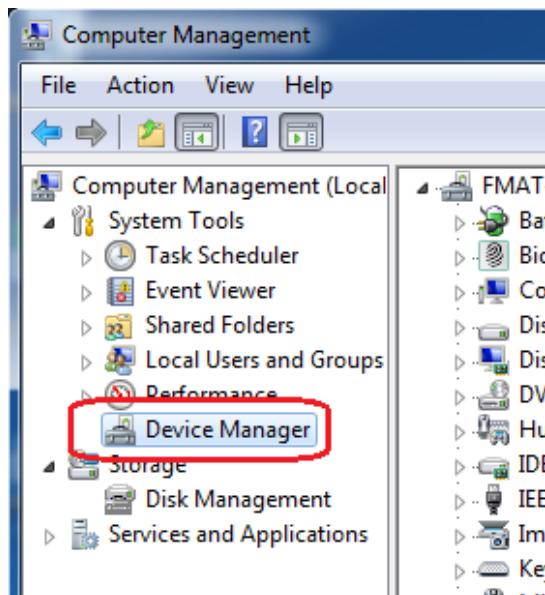


FIGURA C.21: Localización del Device Manager.

Entonces, ubicar el menú desplegable *Ports (COM & LPT)*, y ahí debe indicarse el número de puerto utilizado, como se indica en la figura C.22.

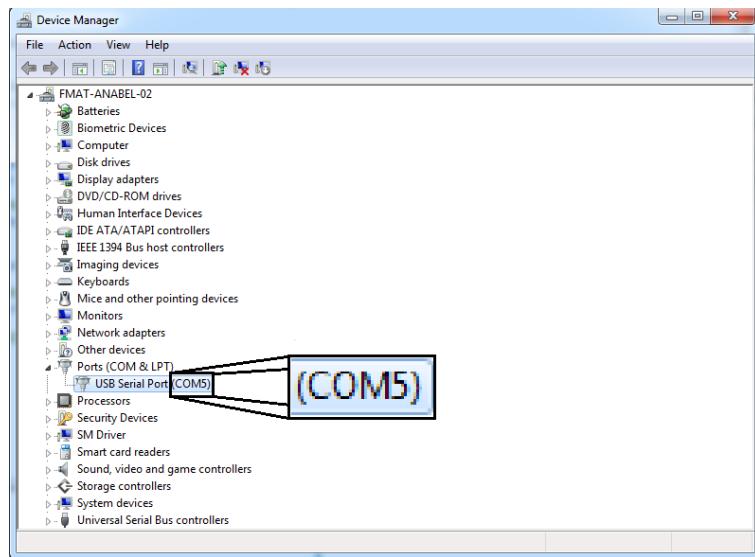


FIGURA C.22: Identificador del puerto COM utilizado por el XBEE.

Añadir XBEE a XCTU

Se debe abrir el software XCTU (mire la subsección C.2.2). Una vez en la ventana principal, se debe ubicar el botón para añadir un dispositivo, que es como se muestra en la figura C.23, que se encuentra en la parte superior izquierda de la ventana del programa.

Tras darle clic, se abrirá una ventana como la que se muestra en la figura C.24. En esta ventana se observa una sección donde se debe de indicar el puerto al que el XBEE está conectado, y otra donde se pide especificar ciertos valores de la conexión, y dar clic al botón *Finish*. Si el XBEE es nuevo, se recomienda utilizar los valores por defecto de esta ventana, y de ser el caso, indicar explícitamente en la casilla que el dispositivo es programable.



FIGURA C.23: Botón para añadir un dispositivo.

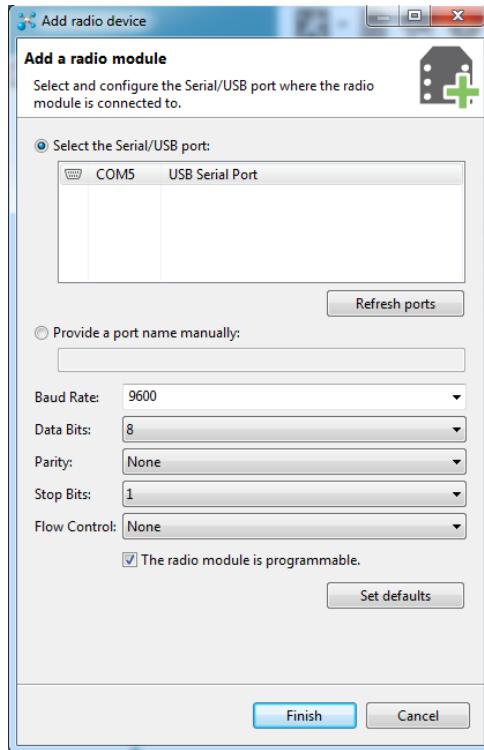


FIGURA C.24: Ventana donde se especifica la conexión al dispositivo.

En caso de que tras seguir los pasos del párrafo anterior el dispositivo aún no sea detectado, entonces utilice el botón para descubrir dispositivos conectados, como el que muestra la figura C.25. La función que realiza es muestrear el puerto a distintas configuraciones hasta establecer una conexión exitosa con el XBEE.

Por tanto, al dar clic a dicho botón, se abre una ventana preguntando por el número de puerto a muestrear, como se indica en la figura C.26. Se pueden escoger múltiples puertos en el caso de que dos o más XBEE estén conectados para que el sistema les descubra y se conozcan sus configuraciones de interfaz serial. Se sugiere escribir y marcar de forma visible a los dispositivos con su configuración (baud rate, bits de parada, etc.) para que sea más sencillo identificarlos y añadirlos al software XCTU mediante la herramienta de añadir dispositivo de la figura C.24 de la página 79. Indique el puerto y de clic al botón *Next*.



FIGURA C.25: Botón para buscar un dispositivo.

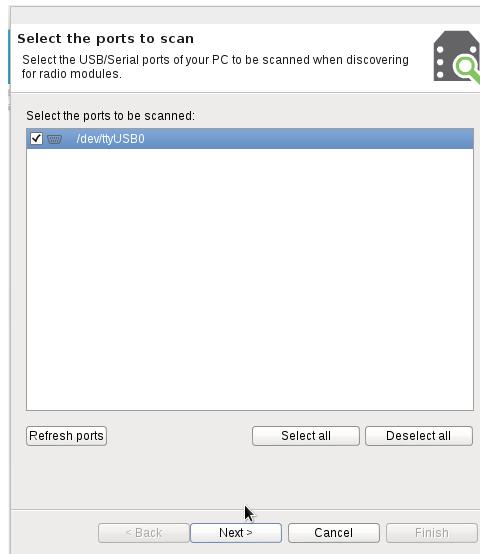


FIGURA C.26: Indique el puerto que el programa muestreará.

Se abrirá una segunda ventana donde se pueden indicar varias configuraciones a evaluar, como el de la figura C.27 de la página 80. En ella, se indica también el tiempo estimado que le tomará al software explorar a través de todas esas configuraciones. Si no se conoce ningún dato sobre el XBEE, puede marcar todas las opciones. De clic al botón *Finish* para iniciar la búsqueda. Tomará varios minutos.

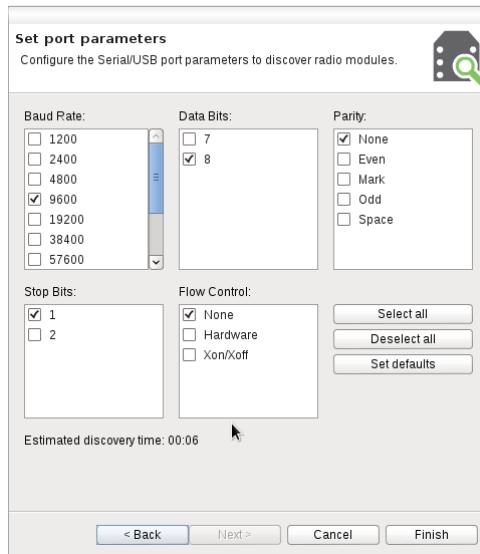


FIGURA C.27: Ventana de configuración de búsqueda.

Una vez finalizada la búsqueda, otra ventana mostrará todos los XBEE detectados por el software, como se muestra la figura C.28. Seleccione todos los dispositivos que desea añadir al software XCTU y de clic al botón *Add selected devices*.

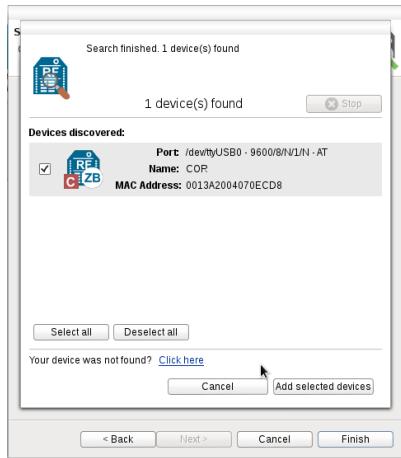


FIGURA C.28: Lista de XBEE encontrados por la herramienta de búsqueda.

Tras cerrar todas las ventanas relacionadas con la búsqueda del dispositivo, se puede observar en la ventana principal de XCTU que, en la parte izquierda, se encuentra el XBEE descubierto y está listo para ser programado, como se observa en la figura C.29.

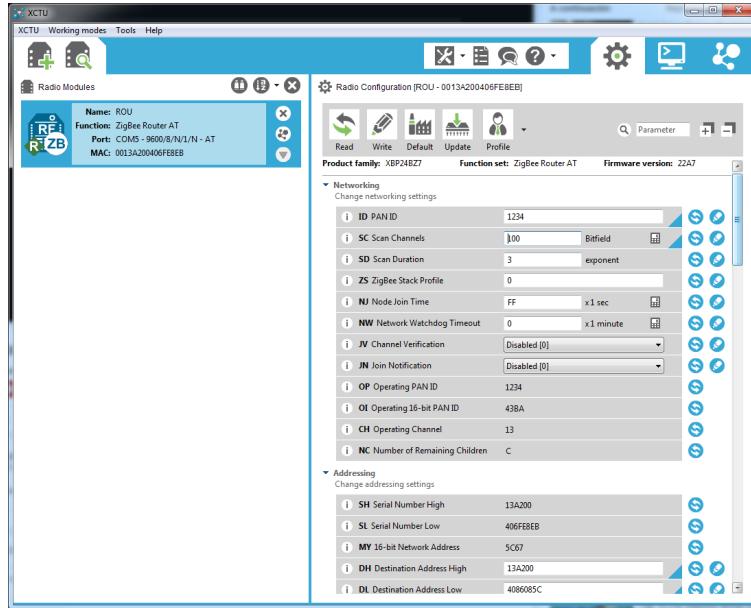


FIGURA C.29: Ventana principal de XCTU con un XBEE añadido.

C.3.3. Programación del XBEE

Configurando los modos

Una vez asociado por lo menos un XBEE a XCTU, como se observa en la figura C.29, se puede comenzar con la configuración del mismo. En el lado izquierdo de la ventana, podrá ubicar una sección con algunos datos relacionados al XBEE que se encuentra conectado, como muestra la figura C.30.

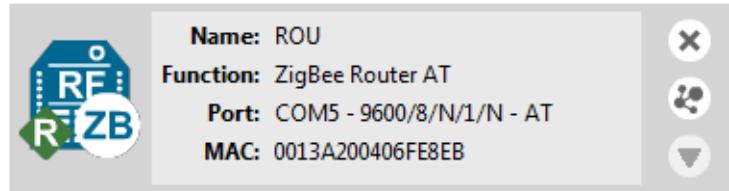


FIGURA C.30: Información del XBEE asociado a XCTU.

Es importante que tenga en cuenta los dígitos del apartado **MAC**, ya que serán útiles en la posterior configuración de los dispositivos. Los primeros ocho dígitos serán conocidos como la parte alta de la dirección física (en el caso de la figura C.30 de la página 82, 0013A200), y los últimos ocho como la parte baja de la dirección física (en la figura C.30, 406FE8EB).

Si se da clic en este apartado, XCTU leerá toda la configuración del XBEE para ser presentado al usuario. La información es separada mediante secciones y tablas, y se diferencian ligeramente dependiendo de la versión de firmware y el tipo de función programada.

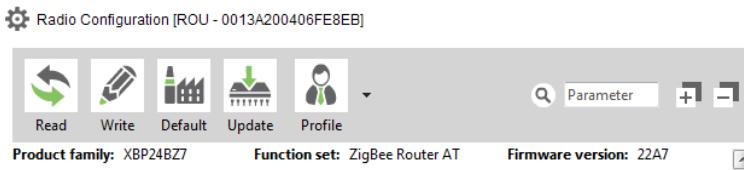


FIGURA C.31: Apartado de información y herramientas.

En la parte superior de esta ventana, se halla un apartado como el de la figura C.31, que ofrece al usuario tanto herramientas así como información sobre el estado de la configuración del XBEE.

A continuación, una breve explicación de esta área:

- **Read:** Se lee la configuración del XBEE y se muestran en las tablas.
- **Write:** Se guardan y escriben al dispositivo todos los cambios realizados a los parámetros.
- **Default:** Restaura al XBEE a las configuraciones por defecto del firmware instalado.
- **Update:** Para cambiar la función asignada y la versión de firmware.
- **Profile:** Para guardar un conjunto de configuraciones en un archivo externo, o para leer uno y desplegarlo en las tablas.
- **Product family:** Muestra el modelo físico del hardware.
- **Function set:** Muestra su función asignada.
- **Firmware version:** Muestra el número de su última versión.

Para el propósito de tener una topología de tipo *Par*, es necesario contar con un XBEE programado como *ZigBee Router AT* o como *ZigBee End Device AT*, y otro XBEE como un *ZigBee Coordinator AT*. Para obtenerlos, se da clic al botón **Update**. Se abrirá una ventana como la de la figura C.32.

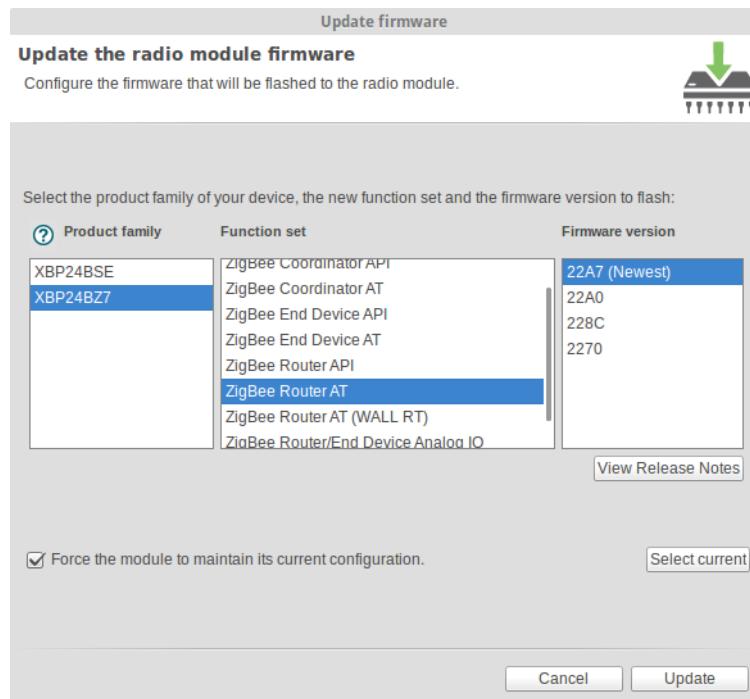


FIGURA C.32: Ventana de programación del firmware.

En la lista del lado izquierdo, llamada *Product family*, se seleccionará el modelo correspondiente al hardware que se tiene, es el mismo que se muestra en el área mostrada en la figura C.31 de la página 82. En la segunda lista, llamada *Function set*, se seleccionará el tipo de dispositivo. Se necesitan ya sea uno programado como ZigBee End Device AT o ZigBee Router AT, y un ZigBee Coordinator AT, así que se selecciona alguno de ellos (si el primer XBEE fue programado como End Device o Router, el segundo deberá ser un Coordinator, o viceversa). En la tercera lista, llamada *Firmware version*, se selecciona alguno de la lista. El texto *Newest* al lado de cierta versión, indica que es la más reciente y por tanto la más recomendada. Puede revisar las notas de los desarrolladores acerca de estas versiones en el botón **View Release Notes**. Si marca la opción *Force the module to maintain its current configuration*, la nueva versión mantendrá configuraciones tales como el área de trabajo, sus identificadores, etc., por lo que se recomienda tenerla seleccionada. Se da clic al botón **Update**. El asistente realizará el proceso de grabado de firmware y le tomará algunos minutos. Repita por cada XBEE a programar. Al terminar, podrá regresar a la ventana principal de XCTU para las últimas configuraciones.

Configurando el Coordinador

Los XBEE cuentan tanto con direcciones físicas, dadas por el fabricante y que son imposibles de cambiar, y de direcciones lógicas, que pueden ser configuradas de acuerdo a las necesidades del proyecto. Tenga en cuenta siempre la dirección física de los XBEE, ya que será necesario colocarlos en la configuración. En el coordinador, los parámetros a configurar son:

- ID Personal Area Network ID o PAN ID (identificador de red de área personal).
- OI Operating 16-bit PAN ID (identificador operativo de red de área personal de 16 bits).

- **CH Operating Channel** (Canal operativo).
- **DH Destination Address High** (Parte alta de la dirección de destino).
- **DL Destination Address Low** (Parte baja de la dirección de destino).

Si dos terminales se encuentran en diferentes configuraciones (no aplica para las direcciones de destino), no serán capaces de encontrarse.

Para configurarlas, hay que introducir los siguientes datos en XCTU:

En la sección *Networking*:

- PAN ID: 1234.
- Scan Channels: 100.

En la sección *Addressing*:

- Destination Address High: [Parte alta de la dirección física del Router o End Device].
- Destination Address Low: [Parte baja de la dirección física del Router o End Device].

Para terminar, de clic al botón *Write* de la parte superior de la ventana.

Configuración mediante comandos AT



FIGURA C.33: Botón de inicio del modo terminal.

A continuación, para programar el parámetro restante, es necesario ingresar al modo terminal. Para ello, se debe dar clic en el botón que se muestra en la figura C.33. En el modo terminal o consola, se pueden enviar datos a otro XBEE y también se pueden leer datos recibidos de otro dispositivo, así como entrar al modo de configuración mediante comandos AT (mire la figura C.34), que es lo que a continuación se realizará.

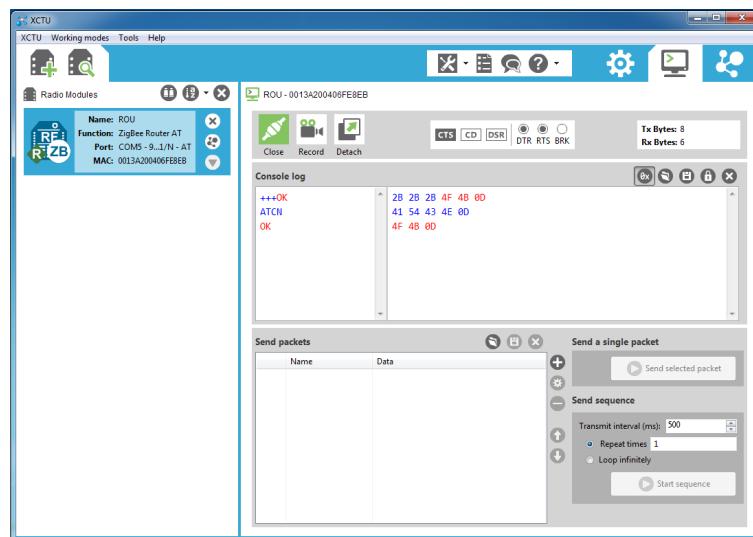


FIGURA C.34: XCTU en modo consola.

Para iniciar con este modo, de clic al botón *Open*, que es como el que se muestra en la figura C.35 de la página 85. Esto

habilitará la consola (*Console log*), donde se podrá trabajar. En textos de color azul, se muestra lo que el usuario teclea en esta terminal. En textos de color rojo, se encuentran los datos recibidos del XBEE, que puede ser tanto respuesta de la comunicación con otro XBEE, así como respuesta del propio dispositivo conectado a la PC cuando se esté configurando.

En el recuadro blanco del lado izquierdo, teclee tres veces el carácter de suma (+++). En color rojo, el XBEE deberá responder con un **OK**. Al recibir esta respuesta, el XBEE estará en modo de configuración por comandos AT⁵. Ahora teclee el comando **ATII 43BA**. El dispositivo deberá responder con otro mensaje de **OK**. Para confirmar que dicho valor fue establecido, teclee el comando **ATII**. El XBEE deberá responder con el valor que le fue dicho explícitamente en la instrucción anterior: **43BA**. En otro caso, vuelva a indicárselo. Para finalizar, teclee el comando **ATCN**. El XBEE responderá con un **OK**. Una vez que el dispositivo de esta última respuesta, retornará al modo de transmisión y recepción normal de datos. Es ahora que el Coordinador XBEE está configurado correctamente y está listo para ser utilizado. Cierre la comunicación de la terminal con el XBEE con el botón Close (quien sustituyó al botón Open de la figura C.35 en la misma posición). Los valores explícitamente proporcionados pueden modificarse para adecuarse a algún proyecto que lo requiriese, para ello, lea el manual que Digi u otros autores ofrecen sobre la programación de estos dispositivos.



FIGURA C.36: Botón de modo de configuración gráfico.

Debe regresar al modo en que se configuraba todo de forma gráfica, para ello, de clic al botón que se muestra en la figura C.36, que se ubica en la parte superior derecha de la ventana.

de la página 86.

Para actualizar todas las configuraciones que se muestran, de clic al botón **Read** que se encuentra en las herramientas de la parte superior de la ventana. Al final, las configuraciones deben aparecer como que se muestra en la figura C.37

⁵Si a mitad del proceso de configuración por comandos AT el XBEE deja de dar respuestas, es porque muy probablemente haya pasado un tiempo sin recibir comandos, por lo que el XBEE retornará a su estado de transmisión y recepción. Deberá volver a iniciar el modo de configuración por comandos AT con los tres caracteres de suma e iniciar el proceso de nuevo.



FIGURA C.35: Botón de apertura de comunicación.

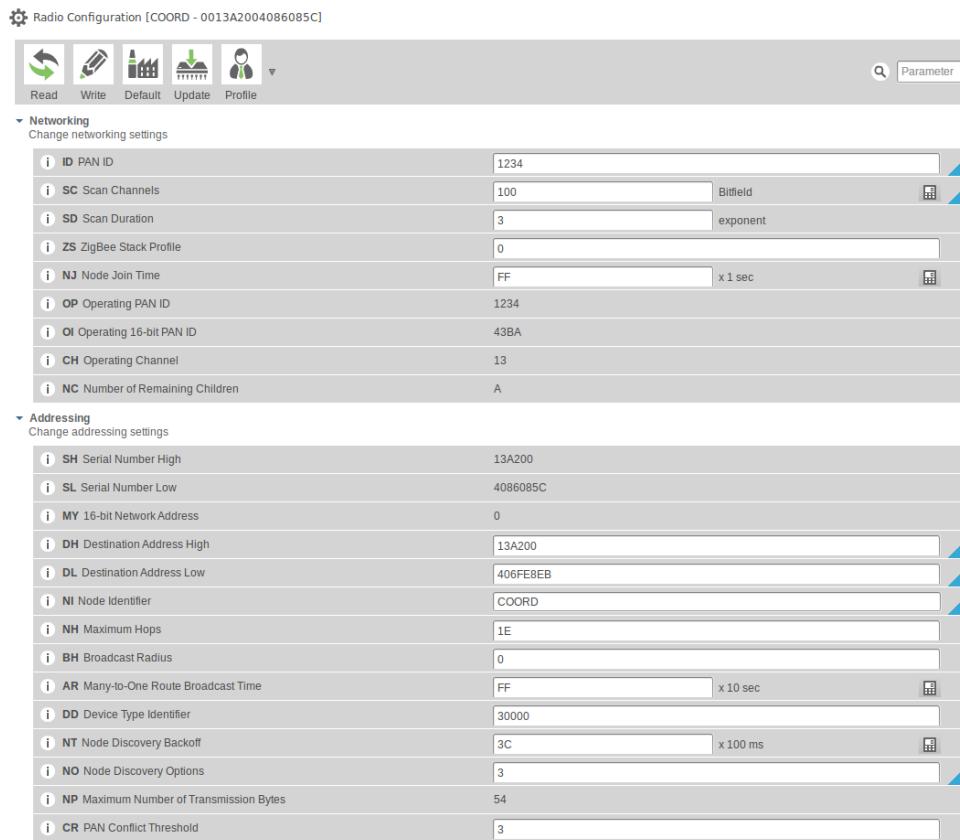


FIGURA C.37: Configuraciones finales en el Coordinador.

Tras verificar que todo esté debidamente configurado, se procede a realizar lo propio con el dispositivo Router o End Device. Para efectos de esta guía, se usará un Router.

Configurando el Router

La configuración del Router se hará de forma similar al coordinador. Se abre la ventana de configuración por tablas (la que se abre por defecto al seleccionar el dispositivo). Los parámetros a configurar son:

- **ID Personal Area Network ID o PAN ID** (identificador de red de área personal).
- **CH Operating Channel** (Canal operativo).
- **DH Destination Address High** (Parte alta de la dirección de destino).
- **DL Destination Address Low** (Parte baja de la dirección de destino).

Como nota, el identificador operativo de red de área personal de 16 bits (**OI Operating 16-bit PAN ID**) debería ser capaz de ajustarse automáticamente⁶.

Bastará con introducir los siguientes datos en XCTU:

En la sección *Networking*:

⁶En caso de que esto no suceda, ajuste manualmente este dato en el XBEE Coordinador, mediante el comando ATII, a la 16-bit PAN ID ofrecida por el Router, mediante el método que se mencionó en la subsección C.3.3. Esto debido a que el firmware de un Router no permite realizar este ajuste aún por comandos AT.

- PAN ID: 1234.

- Scan Channels: 100.

En la sección *Addressing*:

- Destination Address High: [Parte alta de la dirección física del Coordinador].

- Destination Address Low: [Parte baja de la dirección física del Coordinador].

Ahora, debe dar clic al botón de *Write* de la parte superior de la ventana. Para verificar que los ajustes dieron resultados, actualice las cajas de los parámetros dando clic al botón *Read*, que está junto a al botón anterior, en la parte superior de la ventana.

Finalmente, los parámetros deben estar de forma similar a los de las figuras C.38 y C.39 de la página 87. Ahora la programación y configuración de los módulos XBEE están terminadas.

Parameter	Value
ID PAN ID	1234
SC Scan Channels	100
SD Scan Duration	3
ZS ZigBee Stack Profile	0
NJ Node Join Time	FF
NW Network Watchdog Timeout	0
JV Channel Verification	Disabled [0]
JN Join Notification	Disabled [0]
OP Operating PAN ID	1234
OI Operating 16-bit PAN ID	43BA
CH Operating Channel	13
NC Number of Remaining Children	C

FIGURA C.38: Configuraciones finales en el Router, 1.

Parameter	Value
SH Serial Number High	13A200
SL Serial Number Low	406FE8EB
MY 16-bit Network Address	5C67
DH Destination Address High	13A200
DL Destination Address Low	4086085C
NI Node Identifier	ROU
NH Maximum Hops	1E
BH Broadcast Radius	0
AR Many-to-One Route Broadcast Time	FF
DD Device Type Identifier	30000
NT Node Discovery Backoff	3C
NO Node Discovery Options	0
NP Maximum Number of Transmission Bytes	54
CR PAN Conflict Threshold	3

FIGURA C.39: Configuraciones finales en el Router, 2.

C.4. Prueba: Un chat

C.4.1. Preparativos

Esta prueba puede realizarse utilizando el modo consola de XCTU en dos computadoras (mire la subsección C.3.3, donde se explica el modo de ingresar a dicha ventana). Sin embargo, puede probar con una terminal y XCTU en una misma PC, o dos sesiones distintas de terminales. Existen diversidad de ellas en internet, que se pueden usar para evitar instalar XCTU en cada PC o dispositivo que requiera el uso de los XBEE. Una de ellas es PuTTY⁷, que se usará en esta guía, junto a XCTU.

PuTTY no es necesario que sea instalado como cualquier programa de Windows, ya que solo es un ejecutable que al ser invocado, pedirá inmediatamente los datos para establecer una conexión, tal y como muestra la figura C.40.

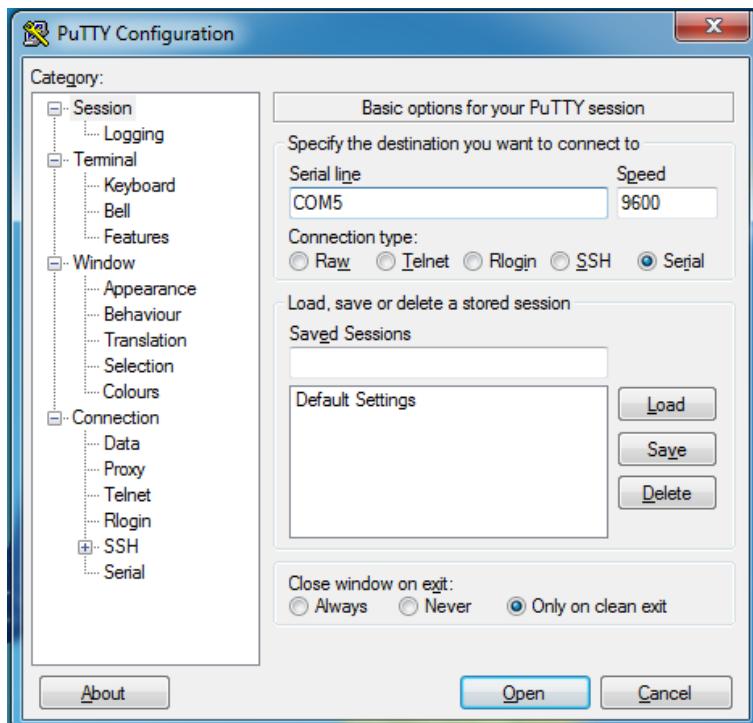


FIGURA C.40: Configuración de sesión en PuTTY.

No importa si que XBEE se elija para comunicarse con XCTU o una terminal, es indistinto. En la ventana de configuraciones de PuTTY que se abre se deberá seleccionar, en el área llamada *Connection type*, a **Serial**. Entonces, en la caja de texto de *Serial line* deberá colocar el puerto a través del cual se comunica la PC con el XBEE (mire la subsección C.3.2). En el área demominada *Speed* deberá colocar la velocidad, en baudios, a la cual una PC se comunica con el XBEE. Es la misma a la cual XCTU reconoció dicho XBEE. En este caso, se trata de 9600 baudios a través del puerto COM5. El resto de la ventana se deja por defecto, ahora de clic en **Open**. Se abrirá una ventana como la que se muestra en la figura C.41.

⁷Descargable de <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

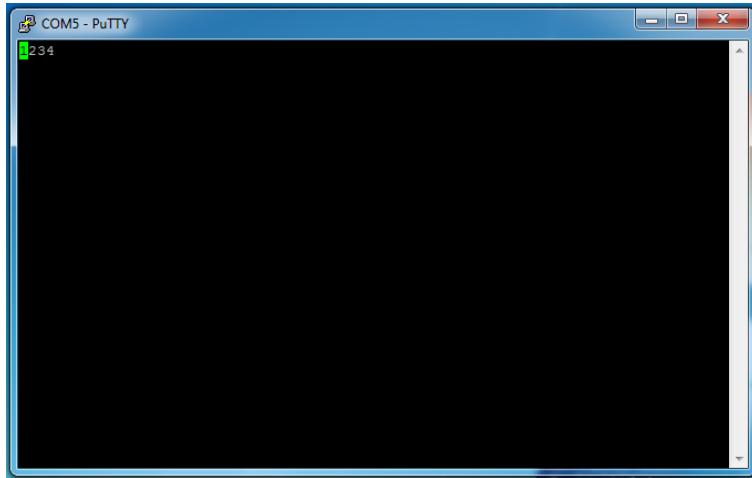


FIGURA C.41: Terminal PuTTY.

Para comprobar que la comunicación se estableció correctamente, teclee los tres signos de suma (**+++**, como cuando se utilizó el modo consola de XCTU en la subsección C.3.3). El XBEE deberá responder con un **OK**. De ser el caso, teclee **ATCN** para ingresar al modo de recepción y envío de datos para estar listos para el chat.

Para el otro XBEE, repita los pasos de apertura de consola en la subsección C.3.3, dejando abierta la consola y verificando que el XBEE de respuesta al comando **+++**, y de ser ese el caso, teclee **ATCN**.

C.4.2. Chateando

Proceda a teclear un texto cualquiera tanto en PuTTY como en la consola de XCTU. Lo que se escriba en la primera deberá de aparecer en la segunda, y viceversa, como se muestra en la figura C.42. Como nota, en PuTTY puede leer lo que recibe, mas no el texto que teclea y envía, a diferencia de la consola de XCTU que muestra tanto los textos de envío y recepción.

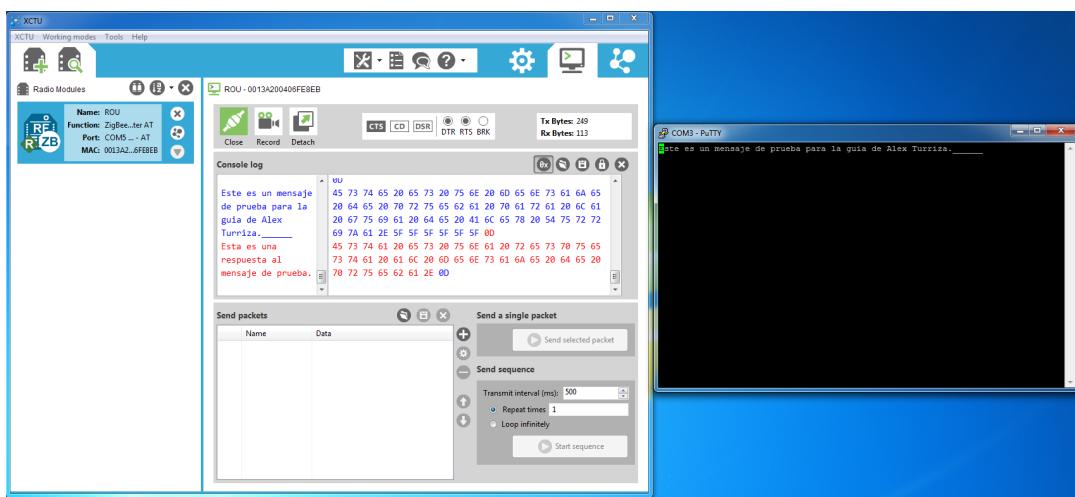


FIGURA C.42: El chat funcionando.

Habiendo verificado que todo funcione correctamente y los XBEE estén comunicándose, ya podrán usarse en los proyectos. En caso contrario, verifique la configuración en la sección C.3.3.

Apéndice D

Configuración del GPS Ublox C94-M8P para su uso en RTKLIB

D.1. Introducción

Para el uso del sistema GPS en modo cinemático, es necesario contar con dos dispositivos en dos roles diferentes: uno que brinde información de posicionamiento en modo estático en un marco de referencia terrestre, llamado *Estación base*, con coordenadas conocidas; y otro, que brinde información de posicionamiento de un equipo en movimiento, llamado *Rover* o *Estación móvil*.

Configuradas de esta manera, RTKLIB puede procesar los datos de ambas para ofrecer una información depurada de posicionamiento del Rover.

D.2. Descripción de los componentes

El kit de GPS Ublox C94-M8P proporciona:

- Un par de equipos GPS.
- Un par de antenas UHF.
- Un par de antenas GNSS.
- Un par de cables de conexión USB-microUSB.

D.2.1. Preparando el dispositivo

Para obtener un dispositivo funcional, hay que tomar un GPS y conectarle una antena UHF al conector marcado para dicha funcionalidad, una antena GNSS al conector indicado, y finalmente, colocarle el cable USB-microUSB al conector micro-USB. De igual manera, se realiza para el otro GPS.

D.3. Instalación en PC

En una PC con Windows, realizar los pasos descritos en las siguientes subsecciones.

D.3.1. Descargas e Instalaciones

Software

El primer paso es descargar el software de Ublox, de la liga: <https://www.u-blox.com/en/product/u-center-windows>. Se abrirá una página web tal y como muestra la figura D.1.



FIGURA D.1: Página web de Ublox.

Dar clic al botón de "Descargar", se obtendrá un archivo con extensión .exe. Instalar siguiendo los pasos indicados por el asistente y esperar a que termine la copia de archivos e instalación de drivers.

Al final, buscar el lanzador de la aplicación de U-Center en el menú Inicio o bien en el escritorio, con un ícono como muestra la figura D.2.

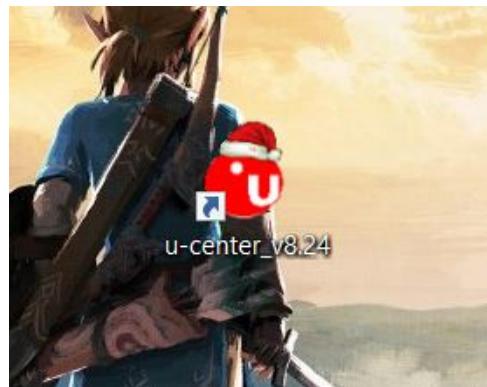


FIGURA D.2: Software U-Center.

Drivers

Para la correcta interpretación de los datos en formato RAW propietario enviados por el GPS, es necesario que los drivers sean instalados en Windows.

Tomar el cable USB-microUSB y conectar a un puerto USB de la computadora con Windows. Esperar a que detecte y se debe mostrar una ventana de instalación de drivers automático, como muestra la figura D.3.

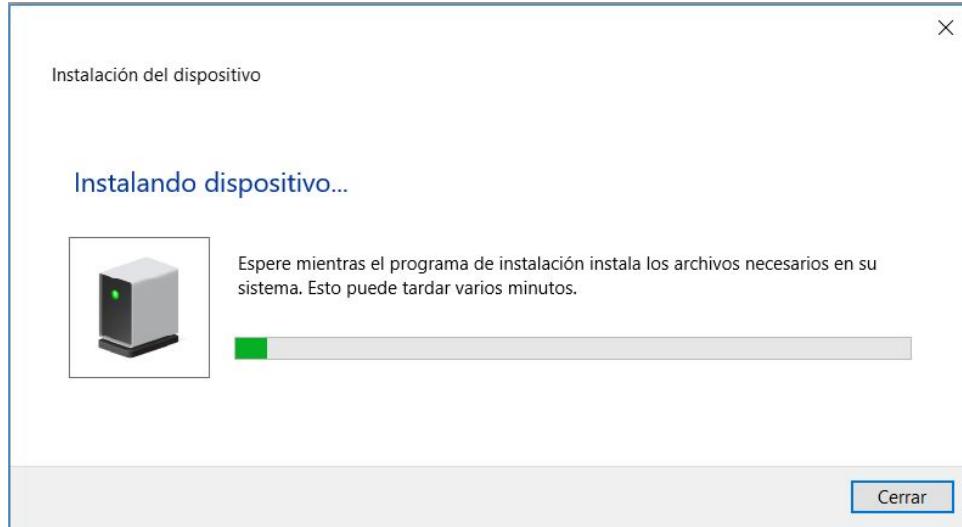


FIGURA D.3: Asistente de instalación de drivers.

Al finalizar esa instalación, cerrar la ventana y verificar que haya sido correcta de la siguiente manera: Dar clic derecho al menú de inicio, y escoger **Administrador de Dispositivos**, tal y como muestra la figura D.4.

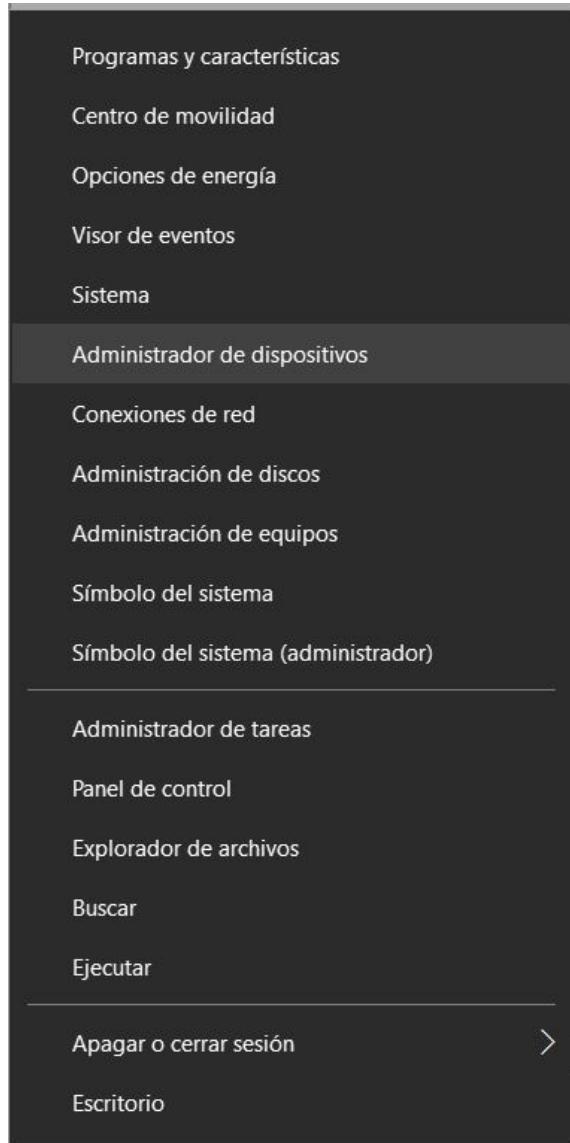


FIGURA D.4: Lista desplegable del menú Inicio.

En la ventana que se abre, en la lista, desplegar *Puertos (COM y LPT)* y verificar que se encuentra sublistado dentro de dicha categoría **u-blox Virtual COM Port (COMX)**, como muestra la figura D.5.

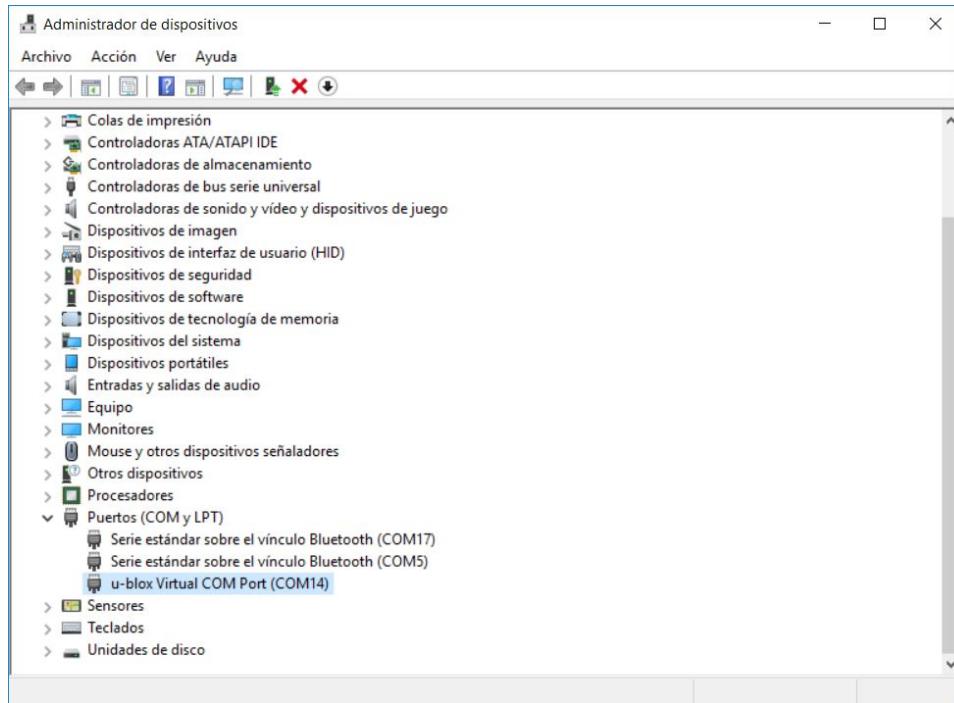


FIGURA D.5: Ventana principal del administrador de dispositivos.

D.4. Configurando los GPS

D.4.1. Configurar la estación base

Abrir el software U-Center y conectar el GPS que será designado a ser de la estación base. Con el GPS correctamente conectado, se mostrará una imagen como la siguiente figura D.6 en la ventana principal del programa.

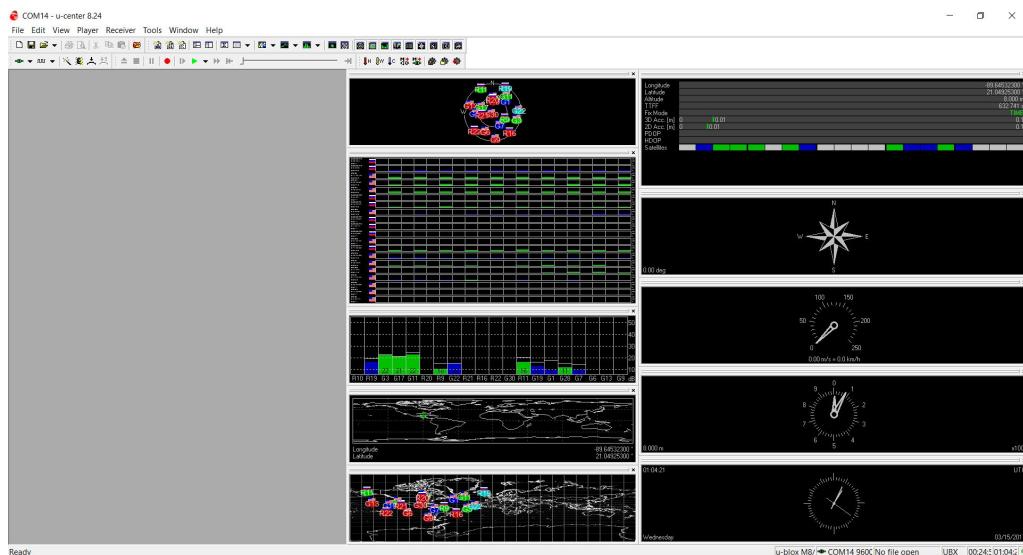


FIGURA D.6: Ventana principal de U-Center.

Del lado derecho de la ventana se despliega información diversa acerca de las observaciones actuales del GPS. En la parte inferior, en la barra de estado, se muestra info acerca del status actual del dispositivo (el tipo de mensaje enviado por el puerto

USB, la tasa de baudios, el puerto al que está conectado a la PC, la hora del sistema satelital, entre otros.

Seleccionar la opción *Messages View* a través del menú *View*: **View → Messages View**, como muestra la figura D.7.

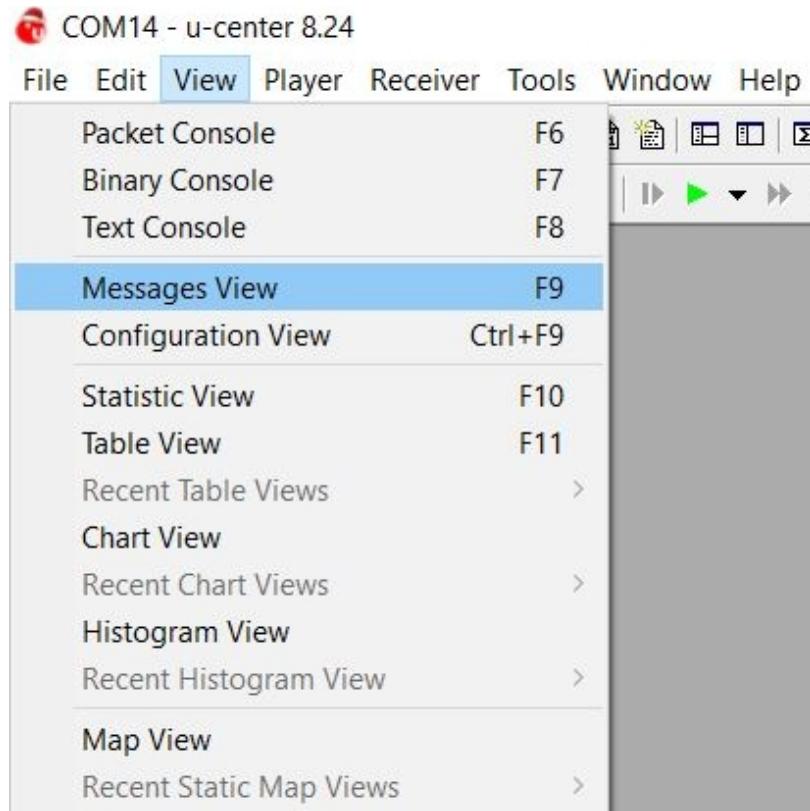


FIGURA D.7: Seleccionar la opción "Messages View" del menú "View".

Se abrirá una ventana en el área izquierda del programa U-Center, como la de la figura D.8.

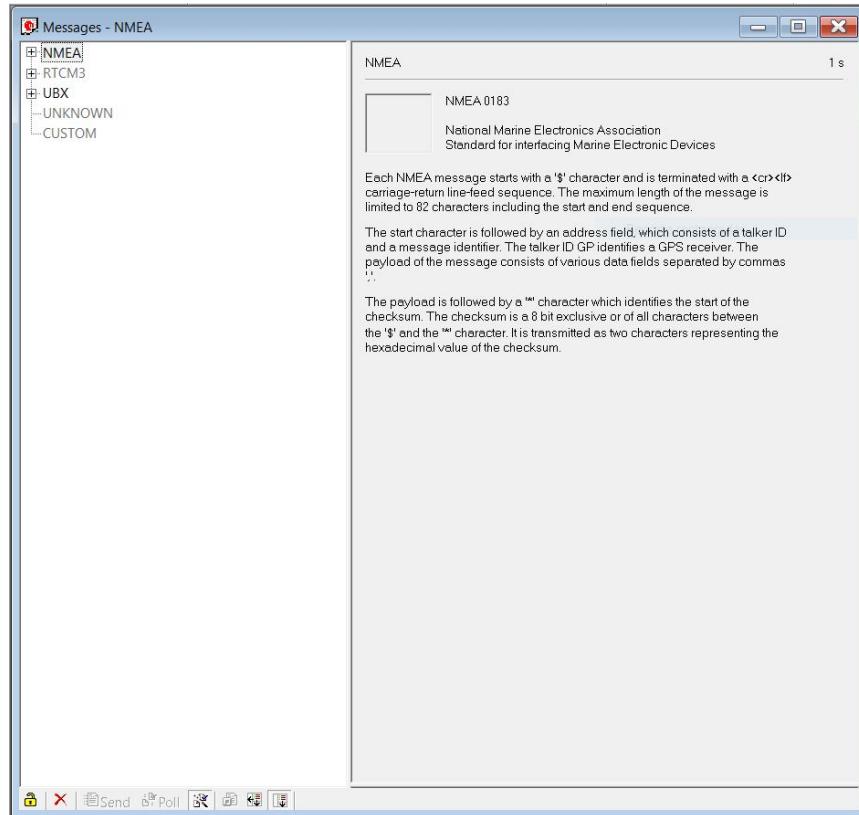


FIGURA D.8: Ventana de la opción "Messages".

Dar clic derecho a **NMEA**, de la lista de la parte izquierda, y seleccionar *Deshabilitar* del menú contextual. Ahora dar clic derecho a **UBX** y seleccionar *Habilitar*.

En la lista de la izquierda, seguir la ruta: **UBX - CFG (Config) - TMODE3**. Saldrán unos cuadros de texto como muestra la figura D.9.

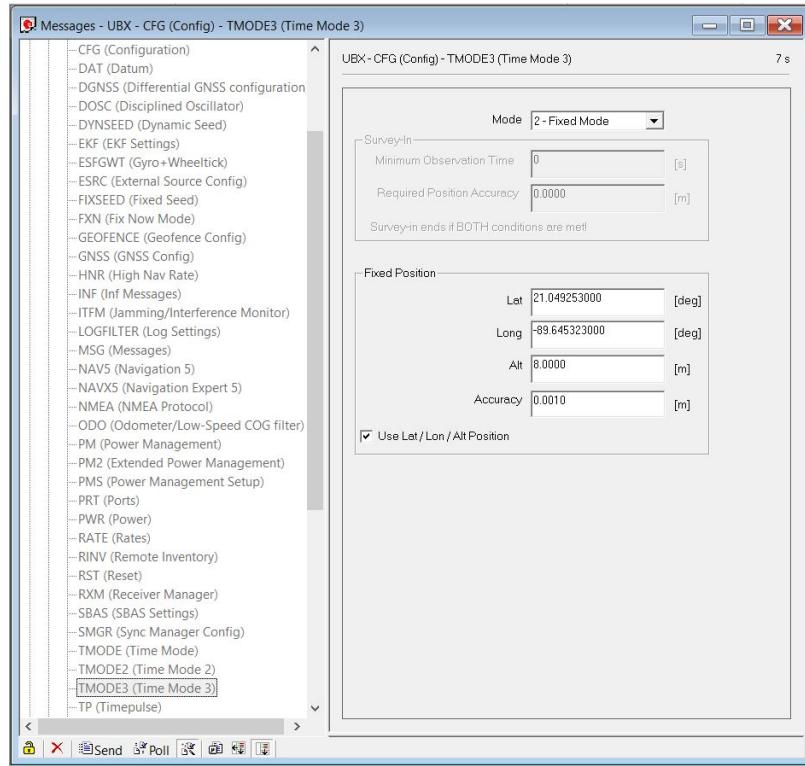


FIGURA D.9: Opciones de la ventana "TMODE3".

Ahí, se introducirán los datos de localización de la estación base. Tal y como indica la figura D.9, se ingresarán los siguientes datos a los cuadros de texto:

- **Mode:** 2 - *Fixed Mode*, para colocar coordenadas fijas de la estación base a utilizar.
- Marcar *Use Lat/Lon/Alt Position* en la parte inferior de la ventana.
- **Lat:** Ingresar la latitud en grados.
- **Long:** Ingresar la longitud en grados.
- **Alt:** Ingresar la altura en metros sobre el nivel del mar.
- **Accuracy:** Ingresar un factor de precisión en metros. Se recomienda utilizar 0.0010.

Finalmente, en la barra de herramientas situada en la barra inferior de la ventana, dar clic al botón **Send**, como el que muestra la figura D.10.



FIGURA D.10: Botón "Send".

A continuación, en la lista de la izquierda, seguir la ruta: **UBX - CFG (Config) - PRT (Ports)**.

Aparecerá una ventana como la de la figura

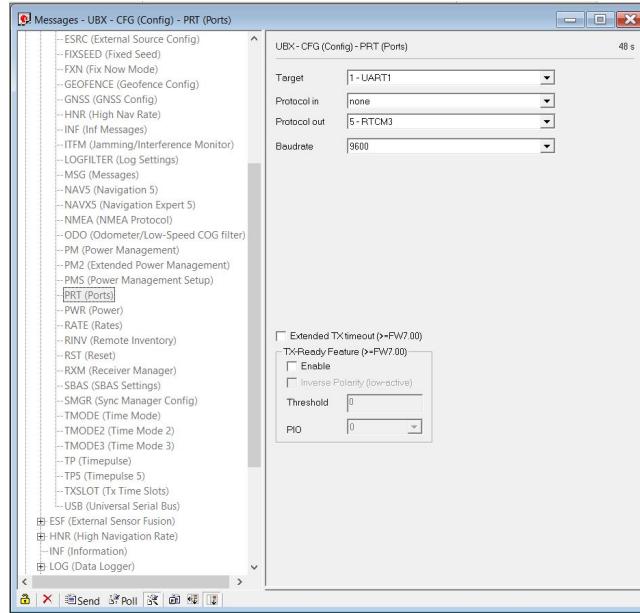


FIGURA D.11: Configuración de mensajes enviados inalámbricamente.

Aquí se ingresarán los datos a enviar mediante transmisión inalámbrica a través del puerto UART1. Se seleccionarán los siguientes parámetros:

- **Target:** 1 - *UART1*
- **Protocol in:** *none*
- **Protocol out:** *5-RTCM3*
- **Baudrate:** *9600*

Deshabilitar en caso de estar marcado *Extended TX timeout(>=FW7.00)*.

Dar clic al botón **Send** de la barra de herramientas inferior.

A continuación, en la lista de la izquierda, seguir la ruta: **UBX - CFG (Config) - MSG (Messages)**.

Se muestra una ventana como la de la figura [D.12](#).

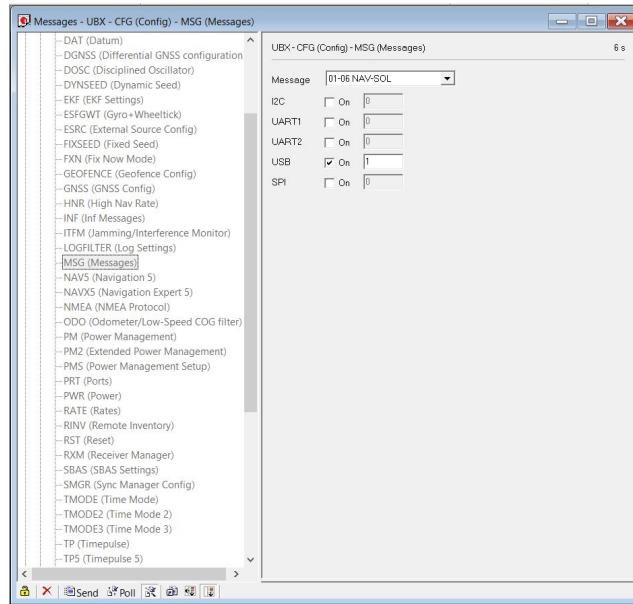


FIGURA D.12: Configuración de mensajes enviados mediante protocolo RTCM3.

En el cuadro de lista desplegable **Message**, seleccionar *F5-05 RTCM3.1 1005*, *F5-4D RTCM3.1 1077* y *F5-05 RTCM3.1 1087*. Seleccionar en todos ellos únicamente el cuadro de *UART1*, tal y como muestran las figuras D.13, D.14, D.15.

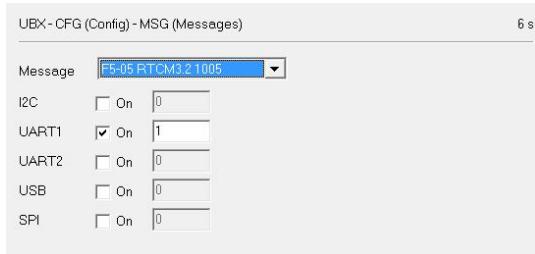


FIGURA D.13: Enviar mensaje 1005 mediante RTCM3.

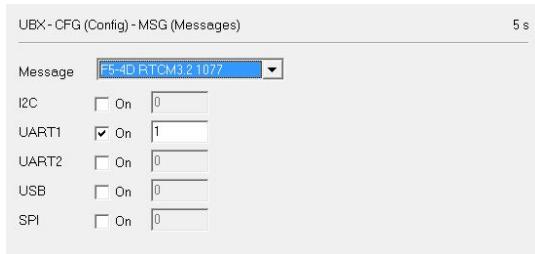


FIGURA D.14: Enviar mensaje 1077 mediante RTCM3.

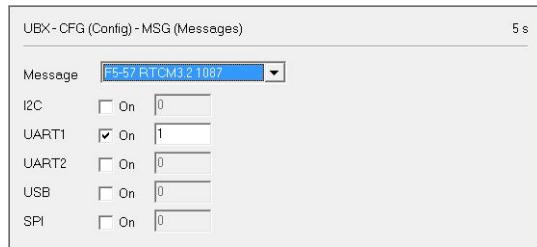


FIGURA D.15: Enviar mensaje 1087 mediante RTCM3.

Dar clic al botón **Send** de la barra de herramientas inferior.

D.4.2. Configurar la estación móvil

Abrir el software U-Center y conectar el GPS que será designado a ser parte del Rover. Abrir la opción **Messages View** como en la subsección D.4.1.

Dar clic derecho a **NMEA**, de la lista de la parte izquierda, y seleccionar *Deshabilitar* del menú contextual. Ahora dar clic derecho a **UBX** y seleccionar *Habilitar*.

En la lista de la izquierda, seguir la ruta: **UBX - CFG (Config) - PRT (Ports)**. Saldrá una ventana como muestra la figura D.16.

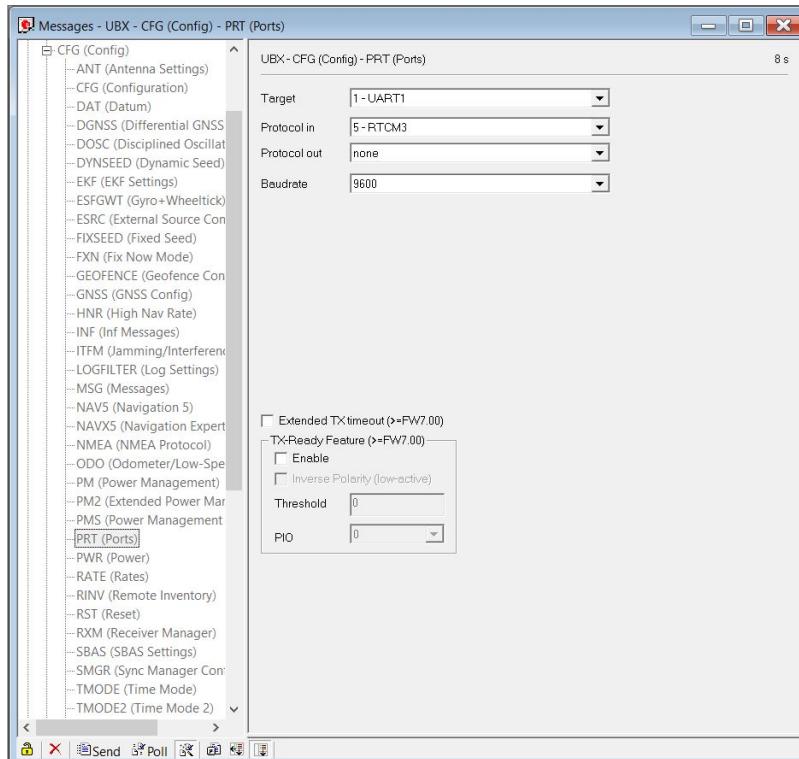


FIGURA D.16: Configuración de comunicación del GPS rover.

Se seleccionarán los siguientes parámetros:

- **Target:** 1 - *UART1*
- **Protocol in:** 5-*RTCM3*

- **Protocol out:** *none*
- **Baudrate:** 9600

Deshabilitar en caso de estar marcado *Extended TX timeout(>=FW7.00)*.

Dar clic al botón **Send** de la barra de herramientas inferior.

Ahora, en la lista de la izquierda, seguir la ruta: **UBX - CFG (Config) - NAV5 (Navigation 5)** y se abrirá una ventana como la de la figura D.17.

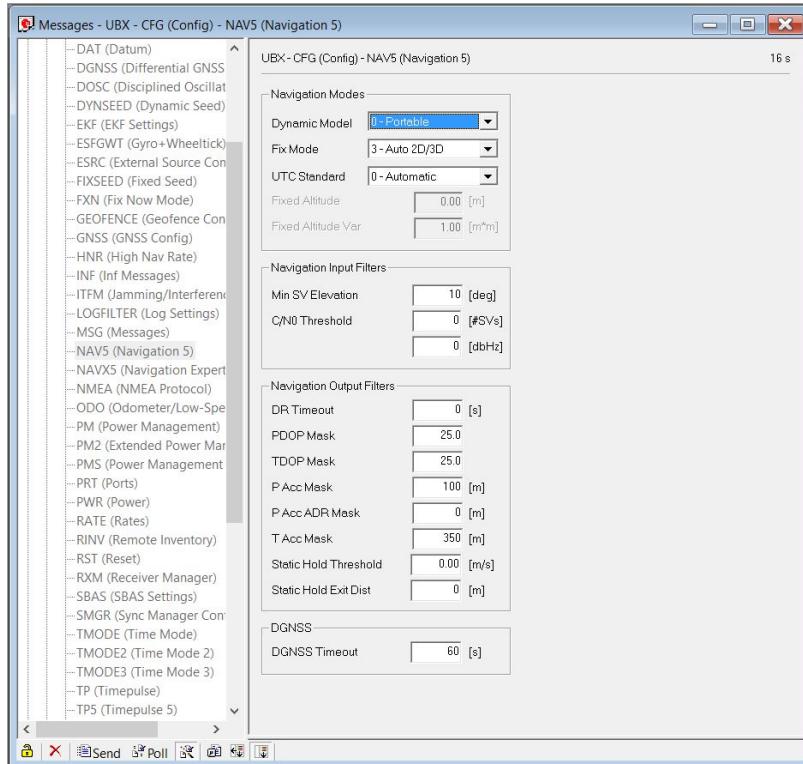


FIGURA D.17: Configuración de datos de navegación.

Ingresar los siguientes parámetros:

En el área de *Navigation Modes*

- **Dynamic Model:** 0-*Portable*
- **Fix Mode:** 3-*Auto 2D/3D*
- **UTC Standard:** 0-*Automatic*

El resto de parámetros dejarlos por default¹.

Dar clic al botón **Send** de la barra de herramientas inferior.

Lo único restante es guardar los datos de configuración en la memoria del GPS. Visitar en la lista de la izquierda la ruta: **UBX - CFG (Config - CFG (Configuration))**. Se abrirá una ventana como la de la figura D.18.

¹Los parámetros por default se muestran en la figura D.17.

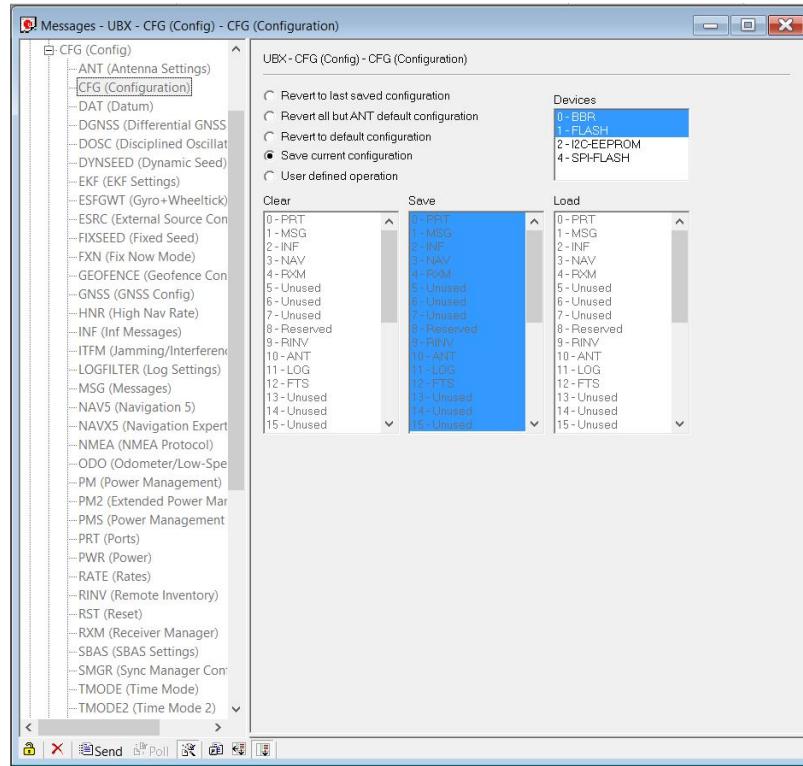


FIGURA D.18: Guardado de la configuración.

Pulsar por última vez el botón **Send** de la barra de herramientas inferior.

D.5. Conclusión

Tras seguir los pasos descritos en esta guía, los GPS Ublox C94-M8P se encuentran listos para ser conectados en sus respectivas estaciones para desempeñar su función. El designado a estar en la estación base enviará datos a su GPS par en el rover para que este último pueda depurar su posición, de acuerdo al fundamento de Real-Time Kinematics.

Bibliografía

- Addati, G. A. y Lance, G. P. (2014). Introducción a los uavs, drones o vants de uso civil. Technical report, Universidad del CEMA.
- Anabel Martin-Gonzalez, Ali Bassam, Carlos Brito-Loeza, Israel Sánchez Domínguez, editores (2016). *Encuentro Universitario de Sistemas Computacionales EUSICS 2016, Memorias en extenso*. Numero 2.
- Barrientos, A., del Cerro, J., Gutiérrez, P., San Martín, R., Martínez, A., y Rossi, C. (2007). Vehículos aéreos no tripulados para uso civil. tecnología y aplicaciones. *Universidad politécnica de Madrid, Madrid*.
- Caballero Paz, A. (2014). Desarrollo de un controlador midi no convencional, implementado en un sistema embebido, utilizando el kinect.
- Cerrato Miranda, J. (2011). Diseño e implementación de una aplicación visual para el control de flotas basado en gps.
- CHÁFER, L. S. (2017). *Diseño de procedimiento Point-In-Sapce LPV para el helipuerto de airbus helicopters (Albacete, España) basado en tecnología GNSS y utilizando EGNOS como sistema de satélites de aumentación europeo*. Tesis de doctorado.
- Coley, G. (2013). Beaglebone black system reference manual. *Texas Instruments, Dallas*.
- Coronado Vallés, J. (2014). *Desarrollo de aplicaciones embebidas de control en robots móviles*. Tesis de doctorado.
- De la Cruz Bautista, V., Luna Vazquez, M., Navarro Tapia, T., Robles Valdez, E. O., y Ramírez Arguelles, M. A. (2011). *Diseño de una estrategia logística de rutas para la distribución de productos farmacéuticos*. Tesis de doctorado.
- Fahlstrom, P. y Gleason, T. (2012). *Introduction to UAV systems*. John Wiley & Sons.
- Fallas, J. (2002). Sistema de posicionamiento global. *Universidad Nacional, Laboratorio de teledetección y sistemas de información geográfica. Escuela de Ciencias Ambientales y Programa Regional en Manejo de Vida Silvestre. Universidad Nacional, Heredia, Costa Rica*.
- Farrell, J. (2008). *Aided navigation: GPS with high rate sensors*. McGraw-Hill, Inc.
- García, A. M. D. y Cuello, R. O. (2007). La promoción del uso del software libre por parte de las universidades. *Revista de Educación a Distancia*, (17).
- Haluani, M. (2015). La tecnología aviónica militar en los conflictos asimétricos: historia, tipos y funciones de los drones letales. *Cuestiones Políticas*, 30(52).
- Huerta, E., Mangiaterra, A., y Noguera, G. (2005). Gps: posicionamiento satelital. *Rosario: UNR Editora, Universidad Nacional de Rosario*.
- i Hernández, J. M. (2005). *Software libre: técnicamente viable, económico sostenible y socialmente justo*. Infonomia.
- Instituto Federal de Telecomunicaciones (2015). Acuerdo por el que el pleno del instituto federal de telecomunicaciones expide la disposición técnica ift-008-2015: Sistemas de radiocomunicación que emplean la técnica de espectro disperso-equipos de radiocomunicación por salto de frecuencia y por modulación digital a operar en las bandas 902-928 mhz, 2400-2483.5 mhz y 5725-5850 mhz-especificaciones, límites y métodos de prueba.

- Maldonado Hidalgo, D. A., Ramírez Acosta, C. F., y Villarreal Quintero, M. E. (2010). Controlador de posición para un vehículo aéreo de 4 rotores realimentado por gps.
- Mendoza Diaz, A., Abarca Perez, E., Mayoral Grajeda, E., y Quintero Pereda, F. (2004). Recomendaciones de actualización de algunos elementos del proyecto geométrico de carreteras. *Publicación técnica*, (244).
- Mueller, K. T., Loomis, P. V., Kalafus, R. M., y Sheynblat, L. (1994). Networked differential gps system. US Patent 5,323,322.
- NavSpark (2016). *NavSpark User Guide*.
- Oyarce, A., Aguayo, P., y Martin, E. (2010). Guía del usuario xbee series 1. *Ingeniería MCI Ltda.*
- Pozo-Ruz, A., Ribeiro, A., García-Alegre, M., García, L., Guinea, D., y Sandoval, F. (2000). Sistema de posicionamiento global (gps): Descripción, análisis de errores, aplicaciones y futuro. *ETS Ingenieros de Telecomunicaciones. Universidad de Málaga*.
- Rönnbäck, S. (2000). Developement of a ins/gps navigation loop for an uav. *Master's thesis*, 81.
- Rubinov, E., Collier, P., Fuller, S., y Seager, J. (2011). Review of gnss formats for real-time positioning. *Africa GEO*, 2011.
- Sonnenberg, G. J. (2013). *Radar and electronic navigation*. Elsevier.
- Takasu, T. y Yasuda, A. (2009). Development of the low-cost rtk-gps receiver with an open source program package rtklib. En *international symposium on GPS/GNSS*, páginas 4–6. International Convention Centre Jeju, Korea.
- Tapia, D. I., Cueli, J. R., García, Ó., Corchado, J. M., Bajo, J., y Saavedra, A. (2007). Identificación por radiofrecuencia: fundamentos y aplicaciones. *Proceedings de las primeras Jornadas Científicas sobre RFID. Ciudad Real, Spain*, páginas 1–5.
- Termal, Z. y Jairo, J. (2014). Prototipo de comunicación entre taxis-policía para la seguridad ciudadana.
- The Debian Installer team (2015). *Guía de instalación de Debian GNU/Linux*.
- U-Blox (2016). *C94-M8P u-blox RTK Application Board Package, User Guide*.
- U.S. National Coordination Office for space-based positioning navigation and timing (2017). Official U.S. government information about the Global Positioning System (GPS) and related topics. <http://www.gps.gov/systems/gps/space/>. Accesado el 04 Abril, 2017.
- Wiśniewski, B., Bruniecki, K., y Moszyński, M. (2013). Evaluation of rtklib's positioning accuracy usingn low-cost gnss receiver and asg-eupos. *TransNav: International Journal on Marine Navigation and Safety of Sea Transportation*, 7(1):79–85.
- Yiğit Yüce (2017). BlackLib reference. <https://github.com/yigityuce/BlackLib>. Accesado el 28 Abril, 2017.