



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 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

Universidad Autónoma de Yucatán

Resumen

Facultad de Matemáticas

Ingeniero en Computación

Implementación de un Sistema de Navegación Cinética Satelital para Asistir en el Control de Posición de Vehículos Aéreos No Tripulados

por ALEX ANTONIO TURRIZA SUÁREZ

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 procesamiento para RTK, que servirá de apoyo al momento de alcanzar la precisión deseada, además de dispositivos de hardware tales como un par de receptores GPS 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, en futuros retos, 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], Rafa[RFer93], Oscar[MadOsc], George[El Tío], Beto[Ax Owl xA] por estos años reuniéndonos para despejar la mente con los videojuegos como excusa. Junto conmigo, Alex[AlexRT07] somos los *inútiles*, o mejor dicho, [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
1.1.1. Descripción del documento	2
1.2. Importancia del tema	3
1.3. Planteamiento del Problema	3
1.4. Trabajos previos	4
1.4.1. Propuesta	4
1.5. Objetivo	4
1.5.1. General	4
1.5.2. Objetivos específicos	5
1.6. Conclusiones	5
2. Marco Teórico	6
2.1. Introducción	6
2.2. El sistema de Posicionamiento Global - GPS	6
2.2.1. Historia	6
Antecedentes	6
Con satélites en órbita	7
2.2.2. La constelación NAVSTAR	7
2.2.3. Estructura	8
Segmento espacial	8
Segmento de control	8
Segmento de usuario	8
2.2.4. Dispositivos GPS	9
2.2.5. Fundamento matemático	10
2.2.6. Causas de error	10
Error causado por el satélite	10
Error causado por la atmósfera	11
Error causado por rutas múltiples	11
Error causado por el receptor	12
Error por Disponibilidad Selectiva	12
2.3. Comunicación inalámbrica	13
2.3.1. Protocolo ZigBee	14
2.3.2. Módulo UHF del Ublox C94-M8P GPS	14
2.4. Sistemas de cómputo embebidos	14
2.4.1. BeagleBone Black	14
2.5. Real Time Kinematics	15

2.6. Formato RTCM-3	16
2.7. RTKLIB	17
2.7.1. Modos de funcionamiento	17
Modo estático	17
Modo cinemático	17
2.8. Conclusión	17
3. Diseño del hardware	18
3.1. Prototipo	18
3.2. Descripción del funcionamiento	18
3.3. Descripción del experimento	18
4. Resultados	19
5. Conclusiones	20

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.	7
2.3. Error del reloj atómico satelital.	10
2.4. Error por capas de la Atmósfera.	11
2.5. Error por rutas múltiples.	11
2.6. Presidente Ronald Reagan.	12
2.7. Símbolo de conectividad inalámbrica.	13
2.8. Equipo XBEE.	14
2.9. BeagleBone Black.	15
2.10. Esquema de funcionamiento en RTK.	15

Índice de cuadros

2.1. Características del GPS Navspark RAW	9
2.2. Características del GPS Ublox C94 M8P	9
2.3. Bandas de frecuencia en comunicación inalámbrica.	13
2.4. Estructura del mensaje RTCM-3.	16
2.5. Contenido del mensaje en formato RTCM-3.1	16

Lista de Abreviaciones

GPS Global Positioning System
RTK Real-Time Kinematics

Constantes Físicas

Rotación de la Tierra $r_{Earth} = 460 \text{ m s}^{-1}$

Lista de símbolos

a	distance	m
P	power	$\text{W} (\text{J s}^{-1})$
ω	angular frequency	rad

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 Kinematics, Sistema de Navegación en Tiempo Real) con el apoyo del software RTKLIB en una BeagleBone Black, basándose en [14], para poder realizar control sobre la posición de un aparato que esté en constante movimiento.

1.1.1. Descripción del documento

El presente trabajo se dividirá 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.

1.2. Importancia del tema

El conocimiento del medio ha sido tema de interés humano desde antaño. Lo que nos rodea, cómo se interactúa con ello en el día a día, así como poder investigar diferentes maneras en cómo se pueden aprovechar los recursos disponibles.

Los mapas realizados a mano, por ejemplo, en lugar de ser una herramienta de exactitud, son solo una referencia del entorno, dado el gran error derivado de la inexactitud humana en las mediciones.

Con el avance de la tecnología y el establecimiento de estándares, obtener herramientas de mejor presición ha sido cada vez más viable, y con ello, se puede obtener una mejor descripción del medio, acercándose a lo que en realidad es, y no a cómo lo interpreta la persona que realiza las mediciones. También, el desarrollo de computadores, sensores y actuadores ha permitido un menor sesgo en las mismas.

1.3. Planteamiento del Problema

Los sistemas de medición actuales poseen cierto grado de incertidumbre, que en ocasiones resulta crítico dado el propósito para el que son utilizados, por ejemplo, en sistemas de tiempo real.

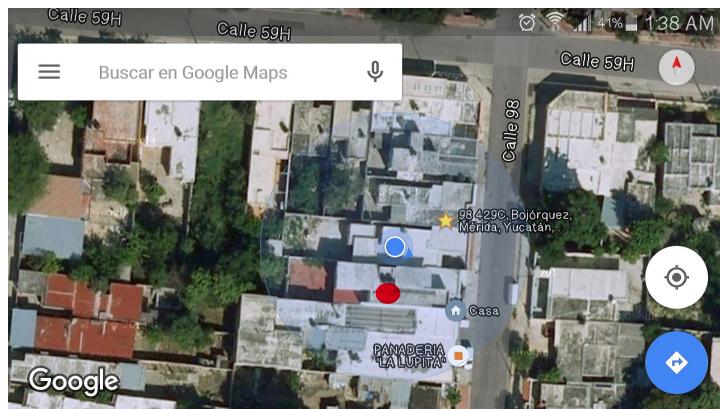


FIGURA 1.1: Posición aproximada de la ubicación de un celular en un mapa de Google Maps.

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. Revise la figura 1.1, donde 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. Además, cuando el sistema no está limitado a sólo seguir una ruta, sino que debe de ejecutar cierta acción cada determinados metros, entonces la incertidumbre se convierte en algo aún más

crítico. Es evidente que el sistema no arrojará los resultados esperados debido a la falta de precisión.

1.4. Trabajos previos

En el trabajo [6], se usa un sistema GPS (Global Positioning System, sistema de posicionamiento global, por sus siglas en inglés) 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 trabajo [10], se propone una actualización del ancho de carriles carreteros a una amplitud adecuada máxima de 3.6 m, y dado que un GPS mide su incertidumbre en metros dependiendo del ruido, entonces se está ante un problema que requiere de modelos y cálculos matemáticos extra para reducir dicho error de medición, que podría incluso indicar que el vehículo se encuentra fuera de la cinta asfáltica o circulando en una vía contraria.

También, la tesis [12], menciona la implementación de un sistema de navegación inercial INS/GPS para un UAV (Unmanned Aerial Vehicle, vehículo aéreo no tripulado, por sus siglas en inglés). Entre los datos a destacar, la posición de este sistema es estimada con un error de 2 metros con un 95 % de confianza, además de otros datos para asistir al vuelo como velocidad y su altitud.

Revisando el desarrollo [9], 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.

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 dada la naturaleza del sensor GPS común utilizado, o bien, se requiere una fuerte cantidad de cálculos matemáticos en filtros para aún así seguir recibiendo errores medibles en metros, 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, ampliando las posibilidades de aprovechamiento de los datos.

1.5. Objetivo

1.5.1. 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 un equipo pequeño y portátil.

1.5.2. Objetivos específicos

- Configurar la transferencia de datos en UHF (Ultra-High Frequency, ultra alta frecuencia por sus siglas en inglés).
- Configuración de los equipos GPS.
- Acondicionamiento de una microcomputadora BeagleBone con sus respectivos periféricos de apoyo.
- Unificación de todos los módulos.
- Evaluación de los datos obtenidos.

1.6. Conclusiones

Un sistema de localización de alta precisión incluso para objetivos móviles tiene un alto potencial dadas las posibles aplicaciones. Como un sensor en un sistema de control, GPS posee un error muy grande para emplearse en trabajos que requieren exactitud y estabilidad, como la toma de fotografías aéreas. Mediante corrección diferencial de tipo Real-Time Kinematics, se puede reducir dicha incertidumbre a niveles de escasos decímetros. El objetivo de este trabajo es el de implementar dicha corrección en un sistema móvil controlado por una BeagleBone Black.

En el capítulo 2, se expone la teoría que se encuentra detrás de la metodología implementada en el proyecto de tesis, partiendo desde la definición de un sistema GPS, pasando por su modo de funcionamiento y las principales causas de error en sus mediciones. También, se presenta una breve descripción del funcionamiento de una comunicación inalámbrica, del software y del hardware utilizado.

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.

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 bajo una referencia mundial. Para dichos propósitos, se necesita una recepción de señales de un mínimo de cuatro satélites, cuyas coordenadas son plenamente conocidas [8].

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 LORAN (*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 emisores, podía determinar su posición.

Fue tras el lanzamiento del *Sputnik*¹, que investigadores norteamericanos descubrieron que era posible la determinación de la posición gracias a la deformación de una señal por efecto Doppler. Así, si era posible conocer la posición del satélite conociendo 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 ellos 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* [16].

2.2.2. La constelación NAVSTAR



FIGURA 2.2: Satélite de la constelación NAVSTAR.

La constelación *NAVSTAR* está conformada por los satélites que se utilizan para triangular la posición. Se propuso que fueran 24 satélites situados en seis planos de cuatro satélites cada uno, con cobertura en toda la Tierra.

La constelación ha sido lanzada en conjuntos llamados *Blocks* (bloques en inglés), distribuidos de la siguiente manera.

- 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): 11 satélites lanzados, 12 satélites lanzados con éxito.

¹Satélite ruso, el primero de la historia.

- Block IIR-M (2005 - 2009): 8 satélites lanzados con éxito.
- Block IIF (2010): 1 satélite lanzado con éxito [16]

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.

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

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

Segmento de control

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 Station, Estación de Control Principal por sus siglas en inglés) 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

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:

- 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.

[16]

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.

CUADRO 2.1: Características del GPS Navspark RAW.

Navspark RAW GPS
 <p>Características:</p> <ul style="list-style-type: none"> • Procesador: 100MHz 32bit LEON3 Sparc-V8 + IEEE-754 Compliant FPU. • 17 Digital I/O. • GPS con actualización de hasta 20 Hz. • Rango de operación: ($h < 18000$ msnm) & ($v < 515$ m/s). • Precisión de hasta 2.5 metros. • Consumo: 15 ma @ 3.3V.

CUADRO 2.2: Características del GPS Ublox C94 M8P.

Ublox C94 M8P GPS
 <p style="text-align: right;">2</p> <p>Características:</p> <ul style="list-style-type: none"> • 72 channel u-blox M8 engine GPS L1C/A, GLONASS L1OF, BeiDou B1. • GPS con actualizaciones de hasta 10 Hz. • Rango de operación: ($h < 50000$ msnm) & ($v < 500$ m/s). • Precisión de hasta 2.5 metros. • Consumo: 23 ma @ 3.3V. • Incluye módulo de radiofrecuencia.

²Imagen alojada en www.u-blox.com

2.2.5. Fundamento matemático

Aquí irá el texto mareador contenido en el Aided Navigation donde se de un breve fundamento del funcionamiento.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec posuere vitae ex quis blandit. In eu libero ante. Maecenas tempus, massa eget convallis sagittis, ligula nisl lobortis lectus, nec efficitur ligula velit non enim. Curabitur iaculis consequat vehicula. Donec tempus sapien laoreet eros hendrerit, quis sodales leo porttitor. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec tincidunt elementum elementum. Donec viverra condimentum vulputate. Cras eget arcu ac odio facilisis convallis. Donec et est eget ipsum aliquet ultrices.

Aliquam egestas a elit a hendrerit. Quisque finibus, quam non faucibus porta, nulla libero blandit elit, in dignissim mauris nulla a erat. Fusce eget massa ac arcu dictum fermentum. Vestibulum iaculis vel nibh eu elementum. Morbi laoreet, metus ut dignissim dictum, risus sem euismod nunc, sed laoreet urna tellus sit amet elit. Nam lectus nisl, interdum sit amet metus at, malesuada imperdiet erat. Sed hendrerit fermentum pretium. Nulla lacinia elementum leo, a vulputate libero congue et. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

2.2.6. Causas de error

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

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

Error causado por el satélite



FIGURA 2.3: Error del reloj atómico satelital.

Los satélites 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.

Error causado por la atmósfera

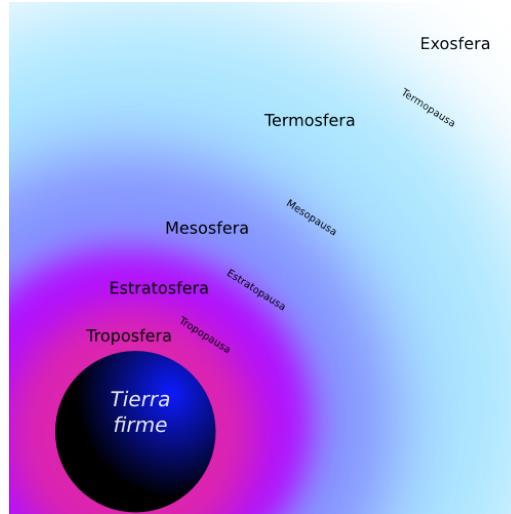


FIGURA 2.4: Error por capas de la Atmósfera.

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 ionósfera³ y entrar en contacto con el vapor de agua de la tropósfera causa variaciones de velocidad en la transmisión.

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

Error causado por rutas múltiples

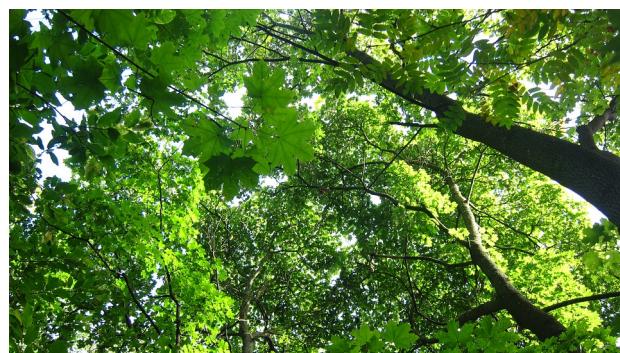


FIGURA 2.5: Error por rutas múltiples.

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 o áreas urbanas.

³El error obtenido en la ionósfera puede eliminarse utilizando receptores de frecuencia doble L1 y L2.

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.

Error causado por el receptor

Así como los satélites, los receptores 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 [7].

Error por Disponibilidad Selectiva



FIGURA 2.6: Presidente Ronald Reagan.

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 eliminarlo en el año 2000. de [16].

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 [3].

2.3. Comunicación inalámbrica



FIGURA 2.7: Símbolo de conectividad 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 frecuencia a la que son emitidas.

A menor frecuencia, se obtiene una gran cobertura pero la capacidad se ve mermada. Conforme se aumenta la frecuencia, se pierde capacidad de cobertura pero la carga que puede tener una banda, aumenta. Se dice entonces que es una relación inversamente proporcional.

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 Commission, Comisión Federal de Comunicaciones, por sus siglas en inglés) quien determina dichas regulaciones. Otras organizaciones reguladoras son la *ISO* (International Organization for Standardization, Organización Internacional para la Estandarización, por sus siglas en inglés) y la *EPCglobal*.

La clasificación del uso de las frecuencias por zona geográfica se muestra en la tabla 2.3.

CUADRO 2.3: 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 [15]

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.



FIGURA 2.8: Equipo XBEE.

Los módulos XBEE, fabricados por Digi International siguen el protocolo ZigBee. 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 [11].

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.

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 compuestos por tres componentes principales: Procesador, Dispositivos de almacenamiento y Periféricos [1].

2.4.1. BeagleBone Black

La BeagleBone es una plataforma de desarrollo de bajo costo.

Desarrollada por la BeagleBone Foundation de los Estados Unidos, una fundación sin ánimos de lucro, cuyo objetivo es la promoción de hardware y software de



FIGURA 2.9: BeagleBone Black.

código abierto para el desarrollo de sistemas embebidos.

Por su diseño, posee una arquitectura ARM, que posee soporte de varias distribuciones Linux [5].

Por su condición de hardware y software abierto, tanto sus esquemáticos del hardware, como los códigos fuente de su software están disponibles a todos los usuarios.

De las ventajas que posee una arquitectura como la del BeagleBone, basada en microprocesadores, es que su uso conduce a plataformas más poderosas, capaces de realizar tareas de gran carga computacional [4].

2.5. Real Time Kinematics

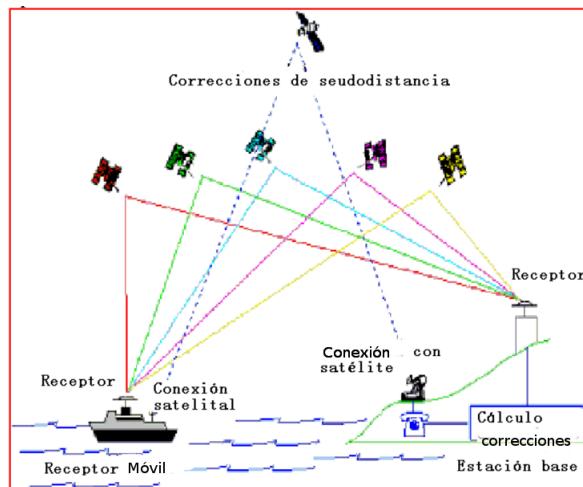


FIGURA 2.10: Esquema de funcionamiento en RTK.

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:

Se tienen dos receptores a una distancia de pocos km 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.

Al 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 [7].

Con todas las correcciones aplicadas de forma ideal, se alcanza una precisión mejor a los 10 cm [2].

2.6. Formato RTCM-3

Este formato 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, como muestra la tabla 2.4[13].

CUADRO 2.4: 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 siguientes datos:

CUADRO 2.5: 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.⁴

⁴Homólogo ruso del sistema americano GPS.

2.7. RTKLIB

Desarrollado por Tomoji Takasu, RTKLIB es un paquete de programas de código abierto escrito en 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 post-procesamiento[14].

2.7.1. 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.

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[17].

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[17].

2.8. Conclusión

Tras una revisión de los conceptos y componentes necesarios, se procede a continuación a detallar 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

3.1. Introducción

Lorem ipsum dolor...

3.2. Prototipo

La interconexión general del sistema quedará como sigue: Es posible observar a la estación base o de tierra en la parte inferior izquierda de la imagen. La estación móvil o Rover se encontrará navegando libremente en un espacio cercano en función a la potencia de los módulos de radiofrecuencia usados para comunicar a ambas plataformas.

3.3. Descripción del funcionamiento

A la estación base se le asignará un conjunto de coordenadas fijas de latitud, longitud y altura sobre el nivel del mar, que es donde ella deberá ser colocada. El GPS realizará las triangulaciones necesarias para obtener su posición estimada, también en coordenadas, a través de los satélites. Así, comparando a las coordenadas previamente fijadas de su posición exacta con las obtenidas de las triangulaciones, la estación base calculará el error que existe en ese instante en las mediciones y procederá a informar, mediante los módulos de comunicación de radiofrecuencia, a la estación móvil, para que proceda a hacer las correcciones necesarias y obtener una aproximación depurada de su posición y sus movimientos.

Se realizó un experimento que consistió en trazar y seguir una determinada ruta para poder observar las diferencias entre un sistema RTK y uno con el GPS triangulando por sí solo.

También fueron colocadas marcas en el suelo para poder tener una guía precisa del camino a seguir y repetir en ambos experimentos.

Se usaron las coordenadas que Google Maps proporciona, como referencia. De este modo, los resultados irán conforme a los mapas de este servicio.

Al final, la ruta a seguir quedaría de este modo:

3.4. Descripción del experimento

 Lorem ipsum dolor...

3.5. Conclusión

 Lorem ipsum dolor...

 En el capítulo siguiente...

Capítulo 4

Resultados

Al final, los resultados obtenidos quedaron de esta manera:

Capítulo 5

Conclusiones

Pepe

UNIVERSIDAD AUTÓNOMA DE YUCATÁN

FACULTAD DE MATEMÁTICAS

ANEXO DE TESIS DE ALEX ANTONIO TURRIZA SUÁREZ

**Configuración de un Sistema de
Archivos en Red [NFS] entre una
PC x86-64 y una BeagleBone
Black**

Autor:

Alex Antonio TURRIZA SUÁREZ

10 de marzo de 2017

Índice

1. Introducción	2
2. Definición de NFS	2
3. Descarga e instalación	2
3.1. Instalación en host / PC	2
3.1.1. Seguridad	5
3.2. Instalación en cliente / BeagleBone	7
4. Ejecución	8

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.

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.

3. Descarga e instalación

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:

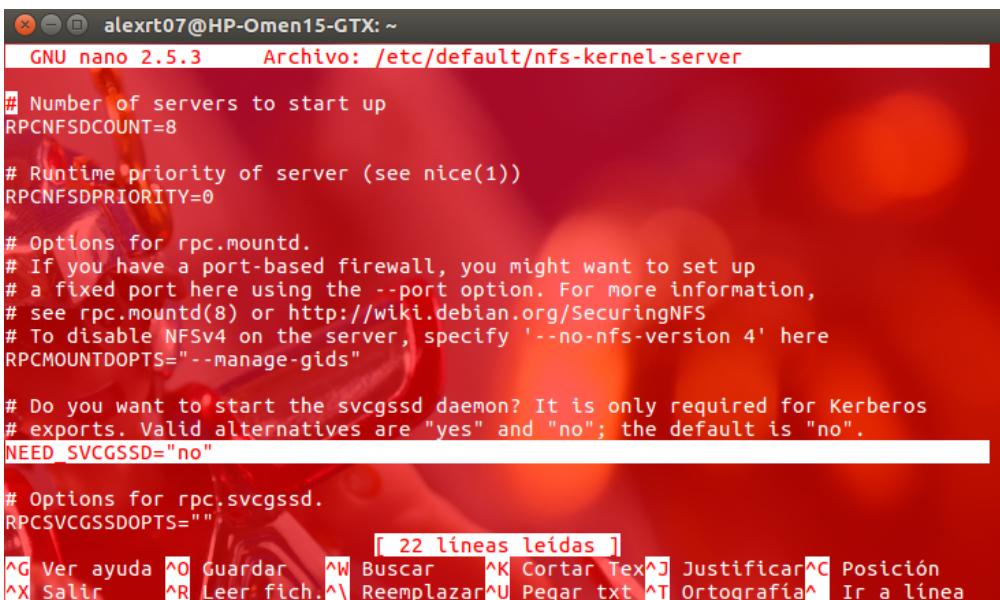
¹<https://debian-handbook.info/browse/es-ES/stable/sect.nfs-file-server.html>

```
$ 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
```

Se debe modificar la línea **NEED_SVCGSSD=** y colocar ”*no*” en el entrecomillado, como muestra la figura 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 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 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

[ 11 líneas leídas ]
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Tex^J Justificar^C Posición
^X Salir ^R Leer fich.^V Reemplazar^U Pegar txt ^T Ortografía^I Ir a línea

```

Figura 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 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. ^I Reemplazar ^U Pegar txt ^T Ortografía ^L Ir a línea

```

Figura 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.

3.1.1. 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
```

The screenshot shows a terminal window titled "alexrt07@HP-Omen15-GTX: ~". The title bar also displays "GNU nano 2.5.3", "Archivo: /etc/hosts.deny", and "Modificado". The main content of the terminal is the /etc/hosts.deny file, which contains the following text:

```
# /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
```

In the bottom right corner of the terminal window, there is a status bar with various keyboard shortcuts:

^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 linea

Figura 4: Archivo /etc/hosts.deny

Como muestra la figura 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 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. ^A Reemplazar ^U Pegar txt ^T Ortografía ^L Ir a linea

```

Figura 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/

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
```

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.

UNIVERSIDAD AUTÓNOMA DE YUCATÁN

FACULTAD DE MATEMÁTICAS

ANEXO DE TESIS DE ALEX ANTONIO TURRIZA SUÁREZ

**Configuración de una Máquina
Virtual de arquitectura ARM bajo
GNU/Linux**

Autor:

Alex Antonio TURRIZA SUÁREZ

26 de octubre de 2016

Índice

1. Introducción	2
2. Software a utilizar	3
2.1. QEMU	3
2.2. Debootstrap	3
3. Descarga e instalación	4
3.1. Configuración del sistema	5
3.2. Ejemplo: compilando un programa en C++	8

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).

2. Software a utilizar

2.1. QEMU



Figura 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*.

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.

3. Descarga e instalación

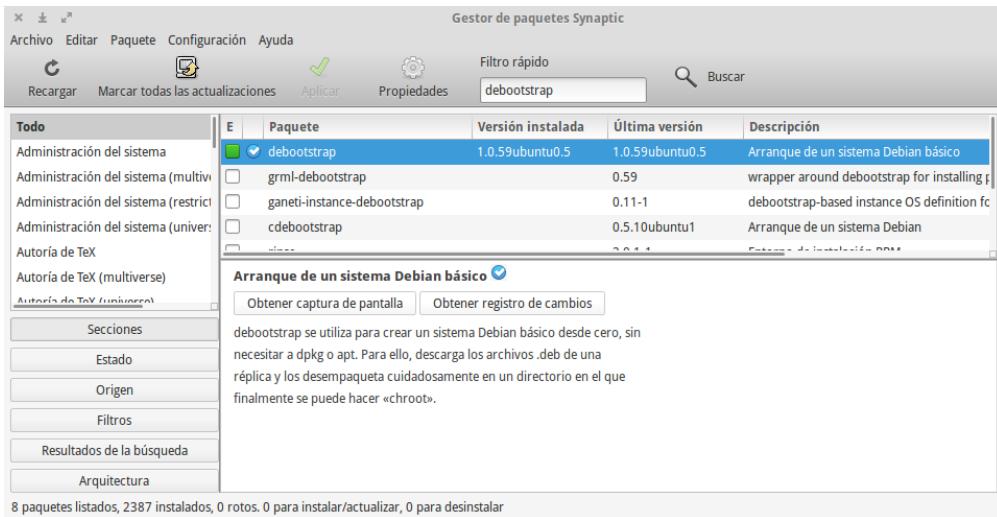


Figura 2: Deboostrap dentro de los repositorios.

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 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
```

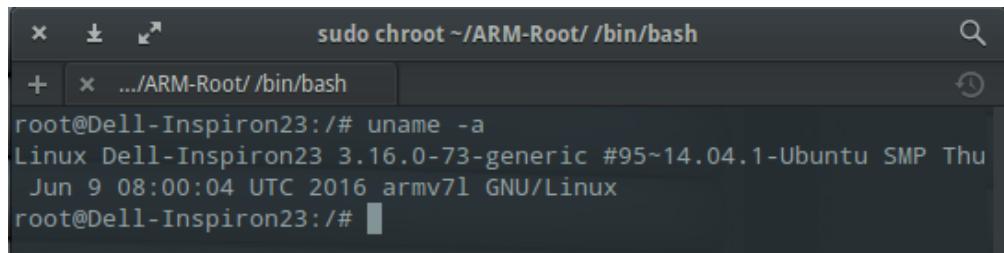
Ya dispone de un sistema Debian ARM totalmente funcional, pudiendo acceder mediante el comando:

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

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
```



The screenshot shows a terminal window titled "sudo chroot ~/ARM-Root/ /bin/bash". The window has a single tab labeled ".../ARM-Root/ /bin/bash". The terminal displays the output of the "uname -a" command, which includes the kernel version (3.16.0-73-generic), the distribution (Ubuntu 14.04.1), the architecture (armv7l), and the date (Jun 9 08:00:04 UTC 2016). The prompt at the end is "root@Dell-Inspiron23:/#".

Figura 3: Salida del comando *uname*.

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.

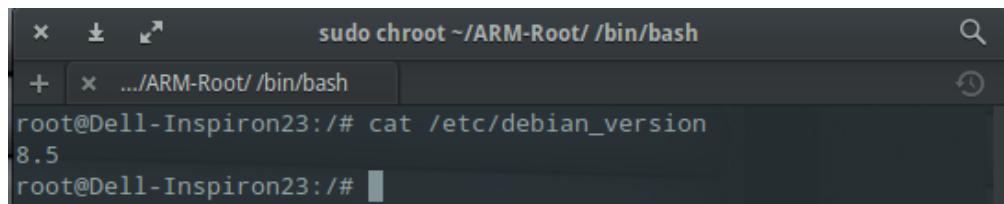
La terminal se debe mostrar como un usuario root dentro del mismo equipo que usted utiliza. Sin embargo, es un sistema de diferente arquitectura. Escriba el siguiente comando para corroborarlo:

```
$ 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 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
```

A screenshot of a terminal window titled "sudo chroot ~/ARM-Root/ /bin/bash". The window shows the command "cat /etc/debian_version" being run and its output "8.5" displayed. The terminal has a dark theme with light-colored text. The title bar also includes ".../ARM-Root/ /bin/bash".

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

Figura 4: Versión del sistema instalado.

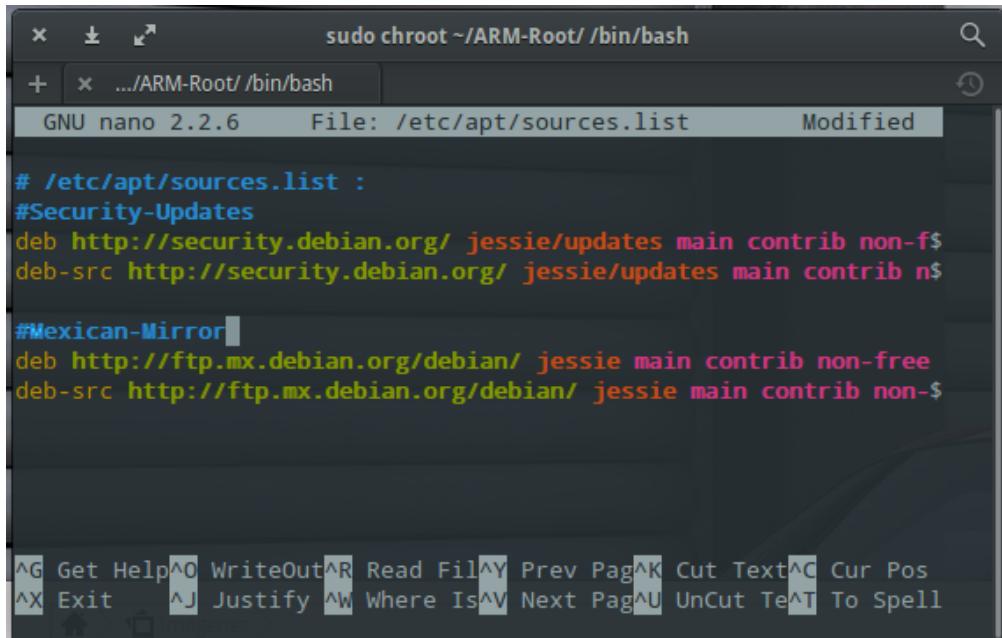
Teniendo el número de versión, se debe conocer con qué nombre fue "bautizada" dicha versión de la distribución. Durante la escritura de este documento, la versión instalada fue Debian *Jessie* (versión 8.*), como se muestra en la figura 4. El archivo */etc/apt/sources.list* es un archivo vacío y en su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

En la web LinuxConfig.org² se encuentran listadas las entradas para este archivo en Debian Jessie.

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

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
```



```
# /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 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 5, se procede a actualizar la lista de repositorios mediante el comando:

```
$ apt-get update
```

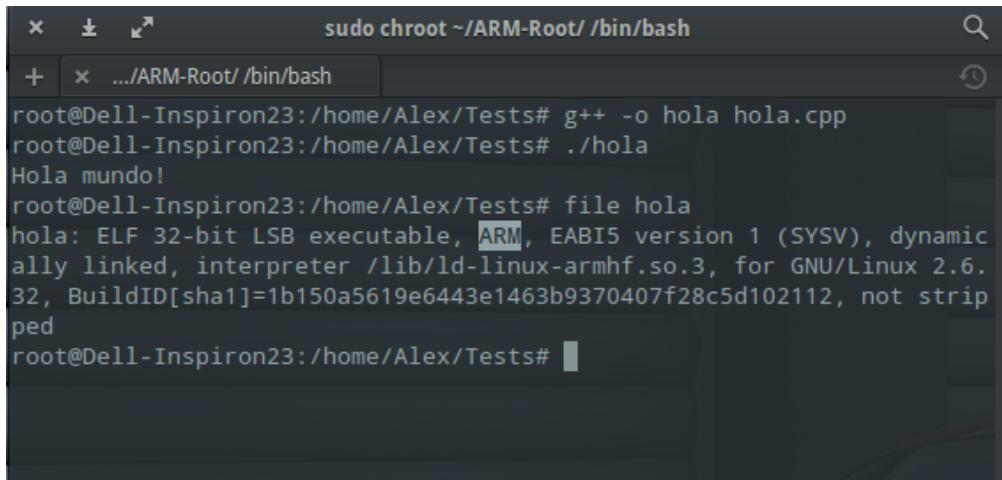
El proceso puede tardar dependiendo de la cantidad de cosas a descargar y de la velocidad a internet. Una vez terminado, se procede a la instalación de los paquetes necesarios para el proyecto en el que se tenga que utilizar este sistema.

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 terminal window shows the following session:

```
sudo chroot ~/ARM-Root/ /bin/bash
+ x .../ARM-Root/ /bin/bash
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 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 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 3.1, en la página 5, debido a que su arquitectura no soporta la codificación de dicho archivo.

³ Instalable mediante: \$ apt-get install file

UNIVERSIDAD AUTÓNOMA DE YUCATÁN

FACULTAD DE MATEMÁTICAS

ANEXO DE TESIS DE ALEX ANTONIO TURRIZA SUÁREZ

Programación de módulos XBEE

Autor:

Alex Antonio TURRIZA SUÁREZ

10 de febrero de 2016

Índice

1. Introducción	2
2. El software XCTU	2
2.1. Descarga e instalación	2
2.2. Apertura del programa	5
3. Conexión y programación de los XBEE	7
3.1. Instalación de controladores	8
3.1.1. Requerimientos para la instalación	8
3.1.2. Instalación	10
3.2. Reconocimiento del XBEE con XCTU	14
3.2.1. Identificación del puerto	14
3.2.2. Añadir XBEE a XCTU	16
3.3. Programación del XBEE	19
3.3.1. Configurando los modos	19
3.3.2. Configurando el Coordinador	22
3.3.3. Configuración mediante comandos AT	23
3.3.4. Configurando el Router	25
4. Prueba: Un chat	28
4.1. Preparativos	28
4.2. Chateando	30

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.

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.

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 1.

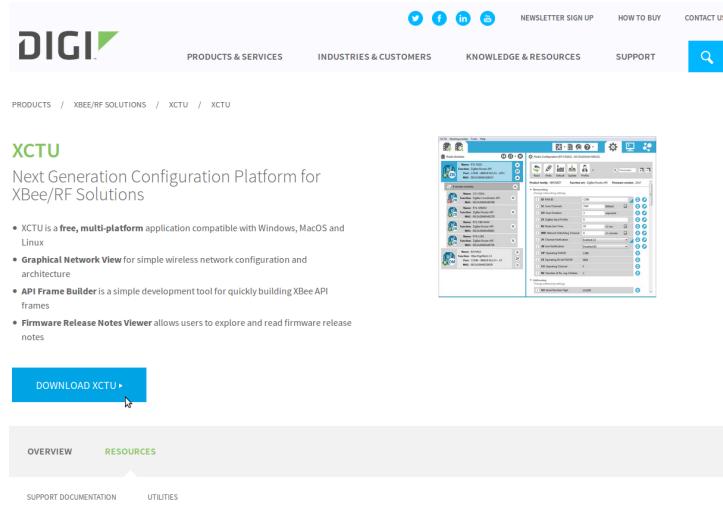


Figura 1: Página de descarga de XCTU.

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

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 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 2.

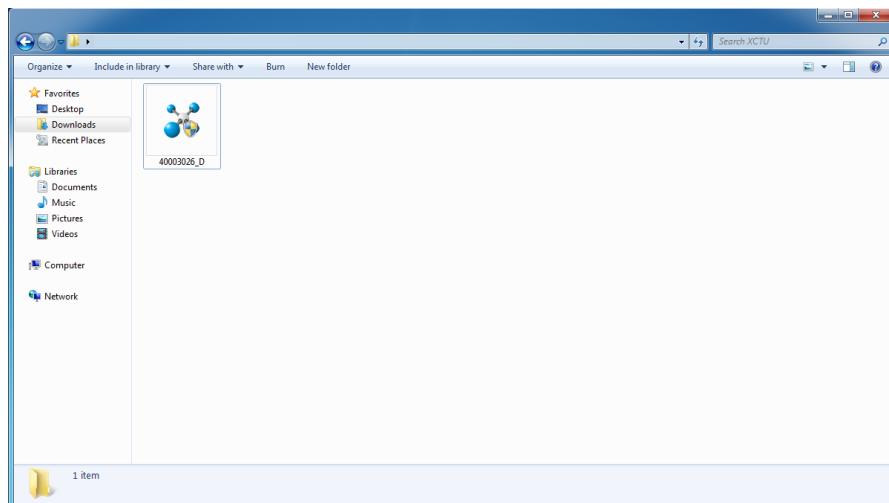


Figura 2: Instalador del programa.

Al ejecutar dicho archivo, el instalador mostrará la pantalla de bienvenida como se muestra en la figura 3 de la página 4. Pulse el botón **Next**.

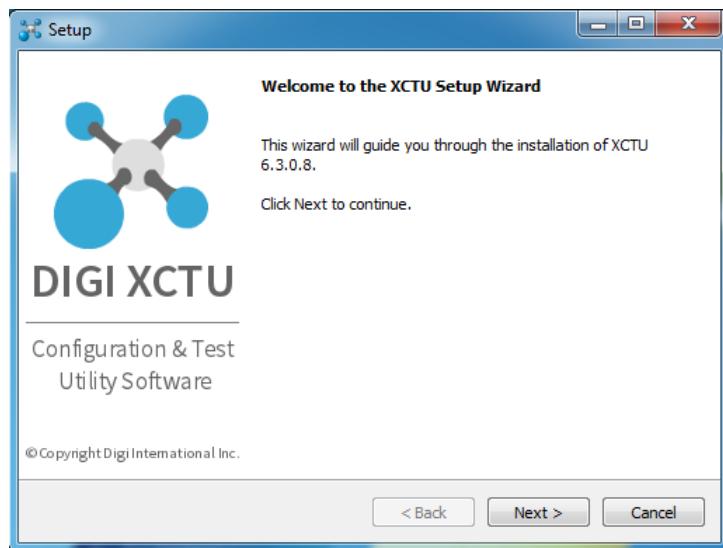


Figura 3: Ventana principal del instalador de XCTU.

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 4 dar clic en el botón **Next**.

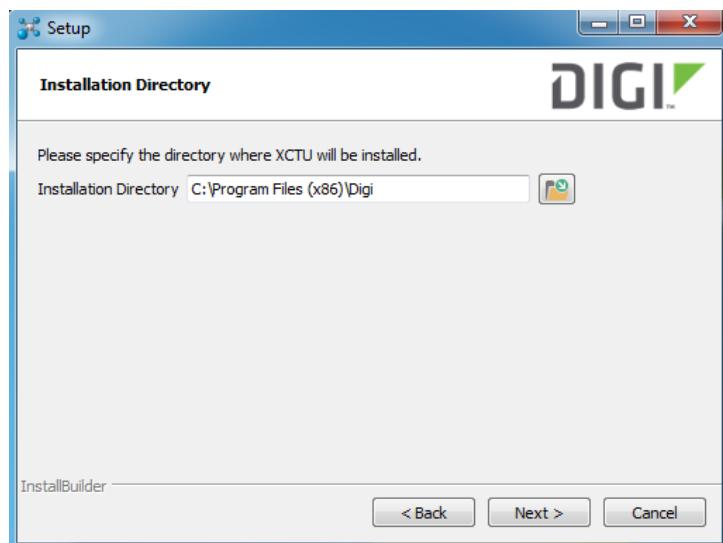


Figura 4: Carpeta de destino de la instalación.

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 5 y solo resta dar clic en el botón **Finish**.

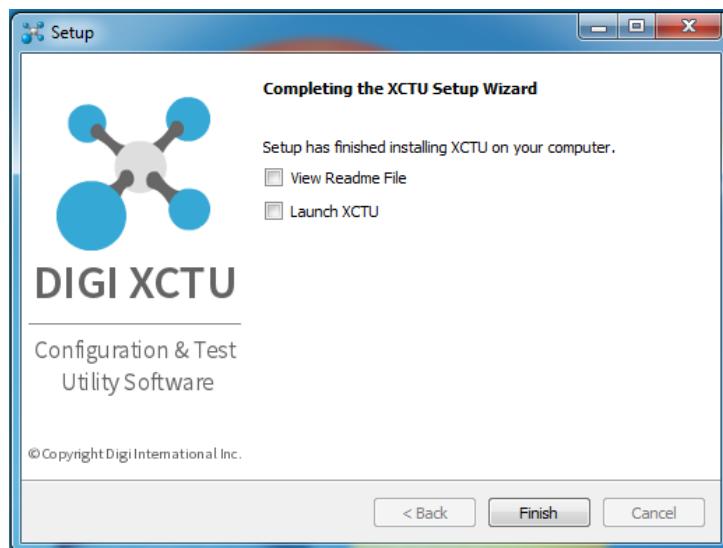


Figura 5: El instalador informa que el proceso ha terminado correctamente.

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 6 de la página 6.

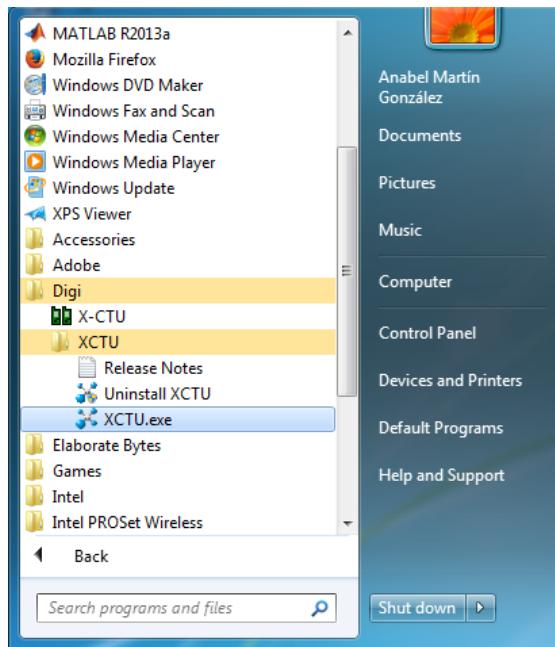


Figura 6: Archivo de inicio del software XCTU.

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

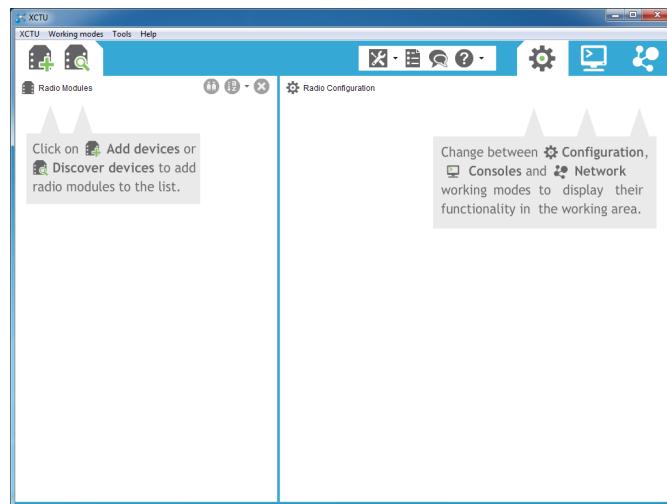


Figura 7: Ventana principal de XCTU.

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 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.

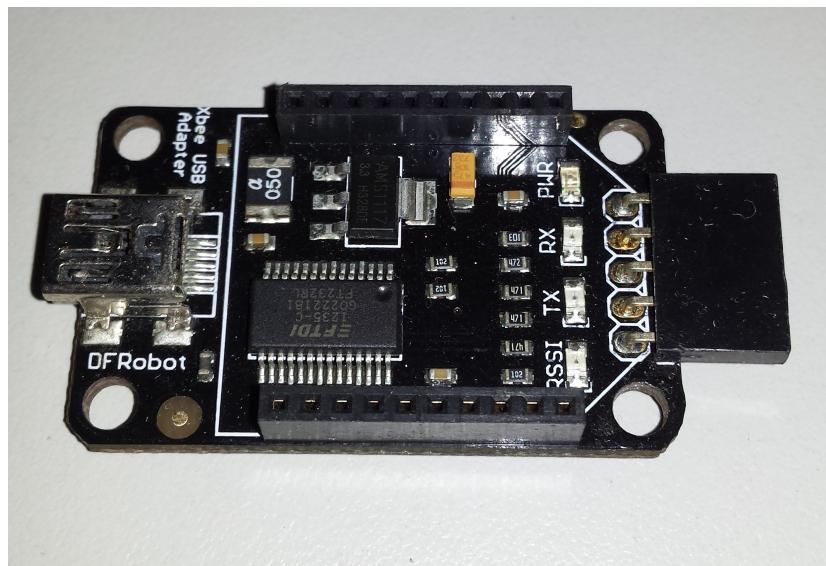


Figura 8: Dispositivo que permite comunicación entre el XBEE y la PC.

Al conectarle 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 9 una vez sea realizada con éxito.

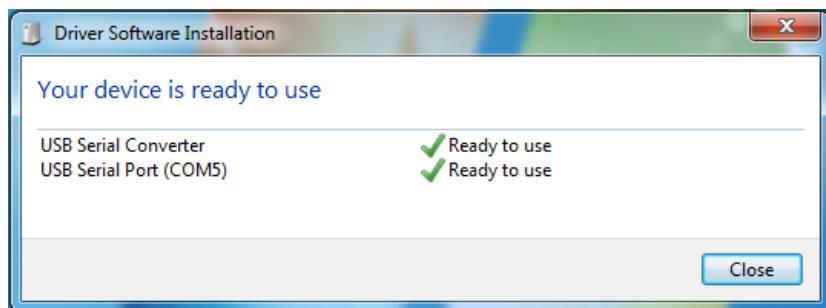


Figura 9: Ventana que indica una correcta instalación de drivers.

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

3.1. Instalación de controladores

3.1.1. 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 10.

Currently Supported VCP Drivers:								
Operating System	Release Date	Processor Architecture						Comments
		x86 (32-bit)	x64 (64-bit)	PPC	ARM	MIPSII	MIPSIV	
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 3.0.0-19 Refer to TN-102 if you need a custom VCP VID/PID in Linux
Mac OS X 10.3 to 10.8	2015-08-10	2.2.18	2.2.18	2.2.18	-	-	-	Refer to TN-102 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
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
Windows CE 2013	2015-03-06	BETA		BETA				BETA VCP Driver Support for WinCE2013

Figura 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 11 de la página 9. Este último paso requiere permisos de administrador. Si usted no los posee, contacte con el administrador del sistema.

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

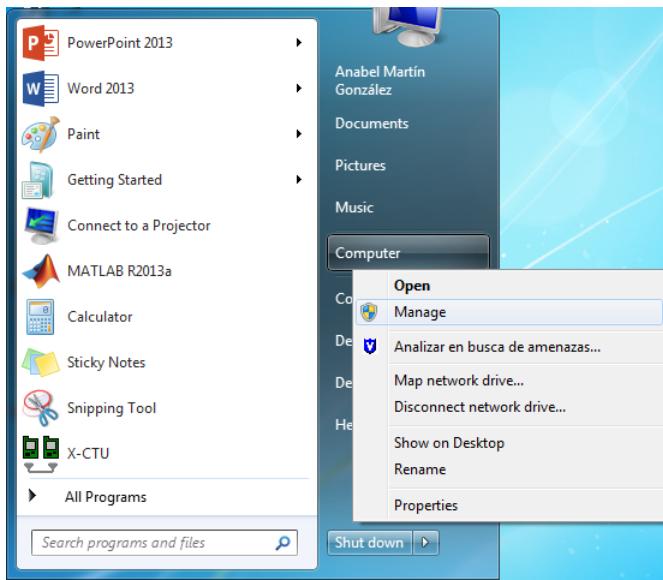


Figura 11: Ubicación del administrador de dispositivos.

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

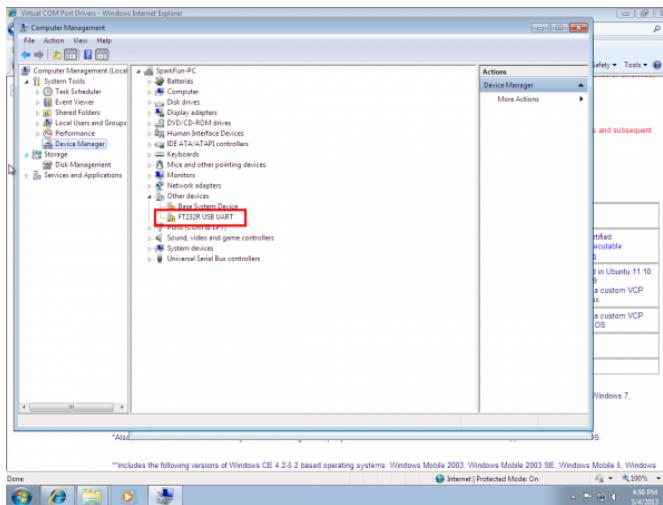


Figura 12: Device Manager. Tomada de Sparkfun³.

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

3.1.2. 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 13.

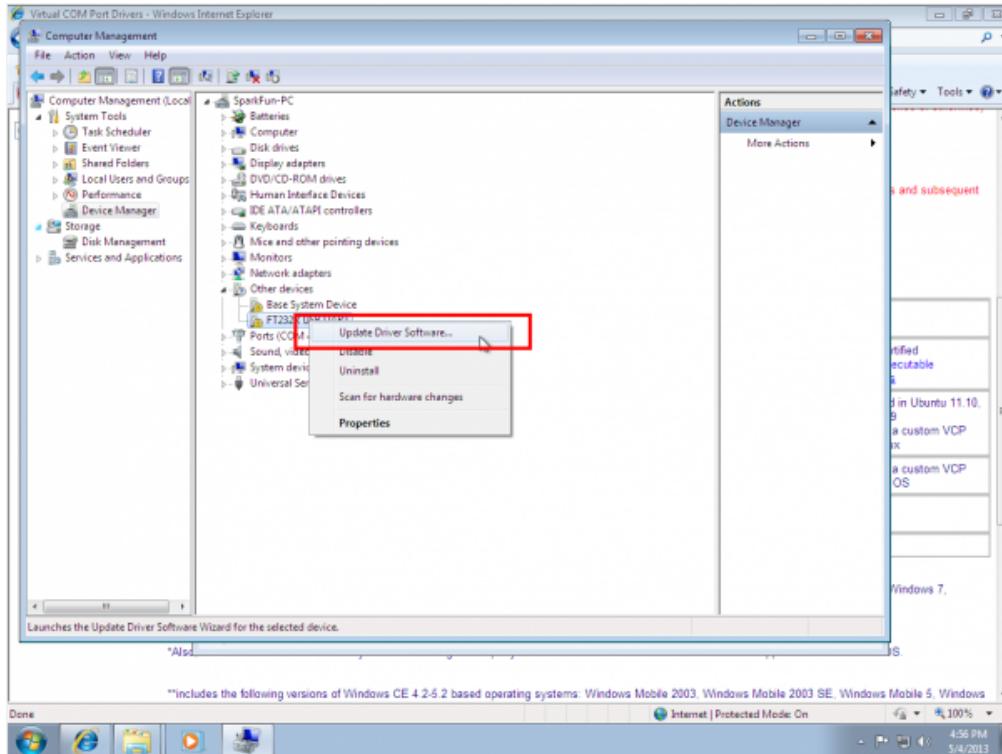


Figura 13: Actualizar controlador.

La ventana que ahora aparece es el asistente de instalación de los controladores descargados en la tabla de la figura 10. Seleccionar ***Browse my computer for driver software***, como se indica en la figura 14 de la página 11.

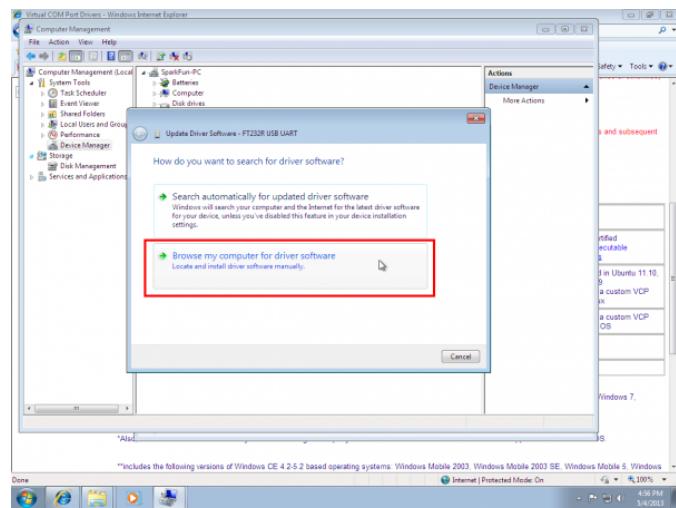


Figura 14: Asistente de instalación del controlador.

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 15.

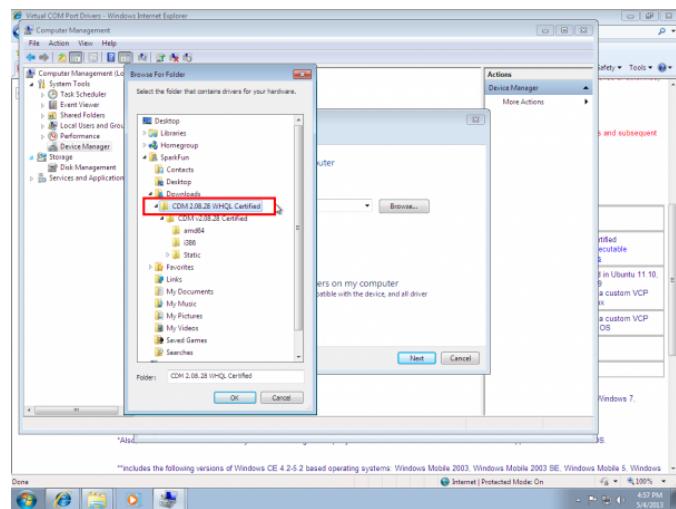


Figura 15: Carpeta del controlador.

Verifique también que la opción *Include subfolders* esté palomeada como en la figura 16 de la página 12, y dar clic al botón **Next**.

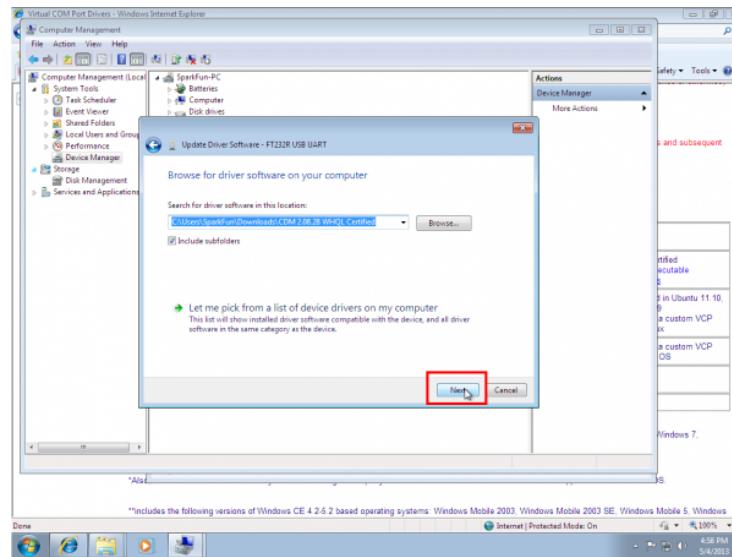


Figura 16: Selección de carpeta.

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

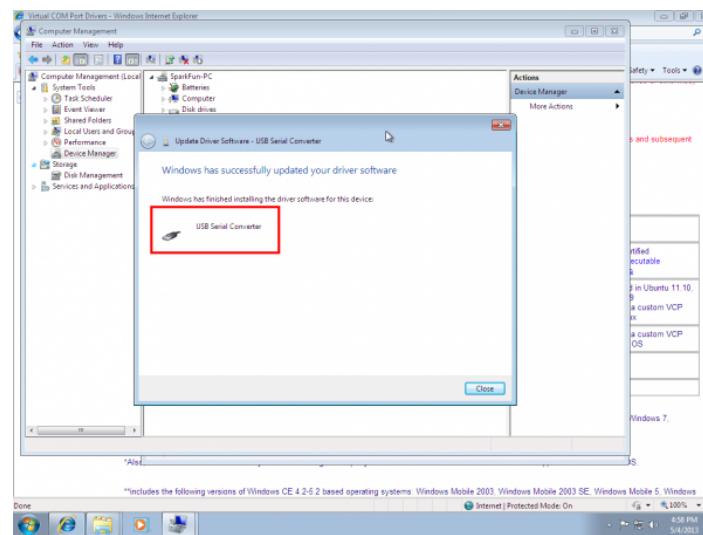


Figura 17: Instalación exitosa.

Sin embargo, la instalación aún no está terminada, por lo que el proceso deberá repetirse desde el inicio de esta subsección 3.1.2, 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 18 usando nuevamente la misma carpeta de controladores que antes se descomprimió.

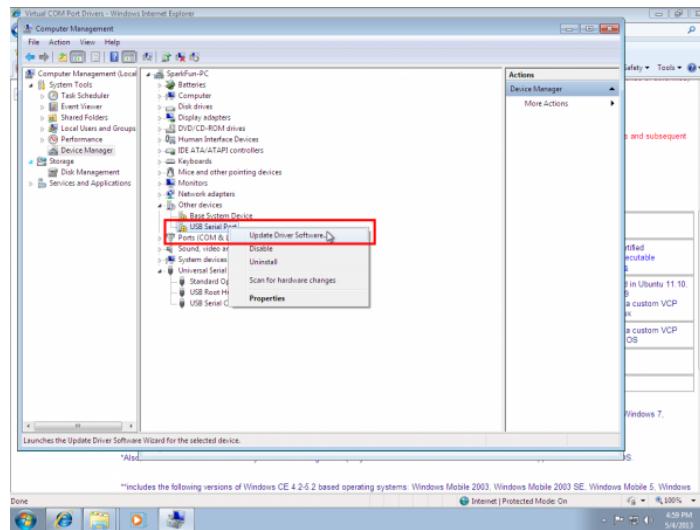


Figura 18: Segunda instalación.

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

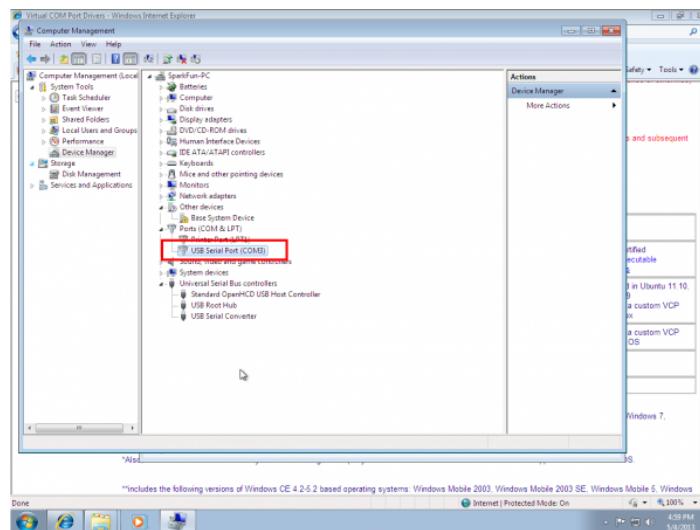


Figura 19: Instalación exitosa de los controladores.

No será necesario repetir esta subsección 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.

3.2. Reconocimiento del XBEE con XCTU

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

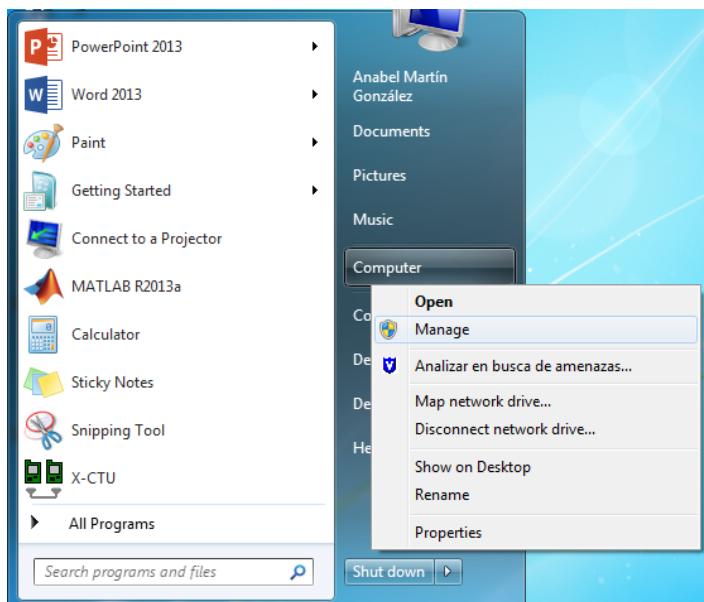


Figura 20: Ubicación del administrador de dispositivos.

En la ventana que se abre, en el menú de la izquierda, dar clic a **Device Manager**, tal como se observa en la figura 21 de la página 15.

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

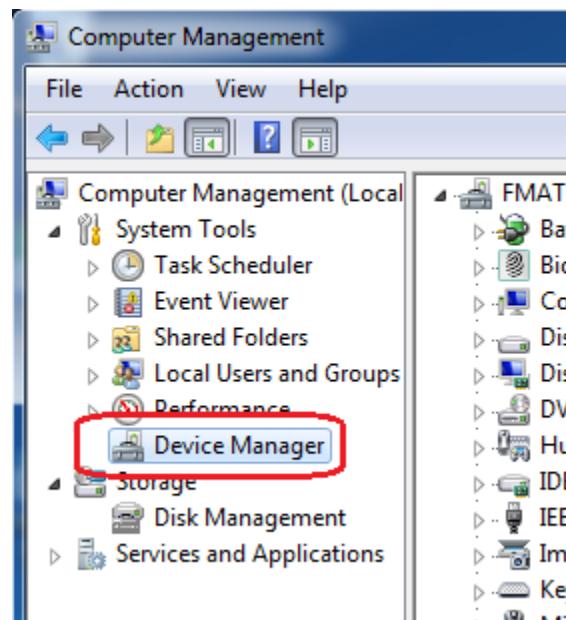


Figura 21: Localización del Device Manager.

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

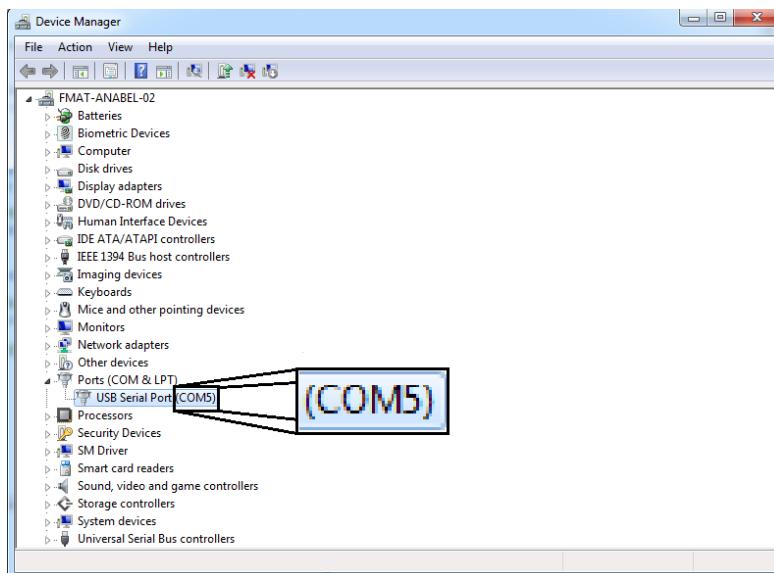


Figura 22: Identificador del puerto COM utilizado por el XBEE.

3.2.2. Añadir XBEE a XCTU

Se debe abrir el software XCTU (mire la subsección 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 23, que se encuentra en la parte superior izquierda de la ventana del programa.



Figura 23: Botón para añadir un dispositivo.

Tras darle clic, se abrirá una ventana como la que se muestra en la figura 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.

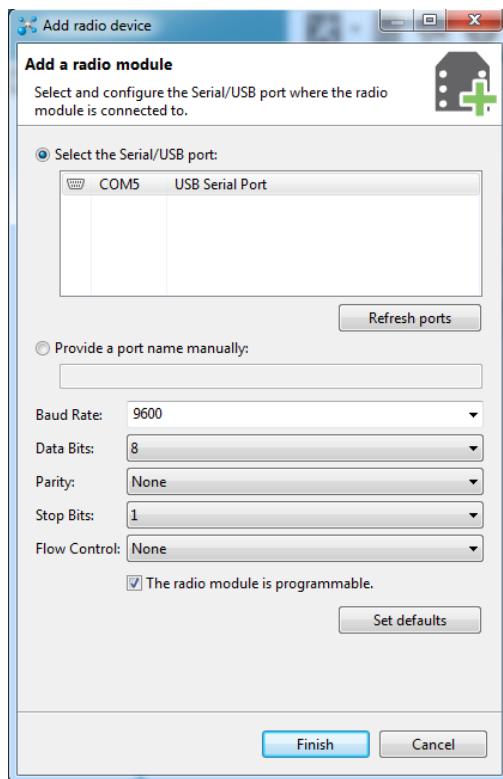


Figura 24: Ventana donde se especifica la conexión al dispositivo.

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 25. La función que realiza es muestrear el puerto a distintas configuraciones hasta establecer una conexión exitosa con el XBEE.



Figura 25: Botón para buscar un dispositivo.

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 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 añadirles al software XCTU mediante la herramienta de añadir dispositivo de la figura 24 de la página 16. Indique el puerto y de clic al botón **Next**.

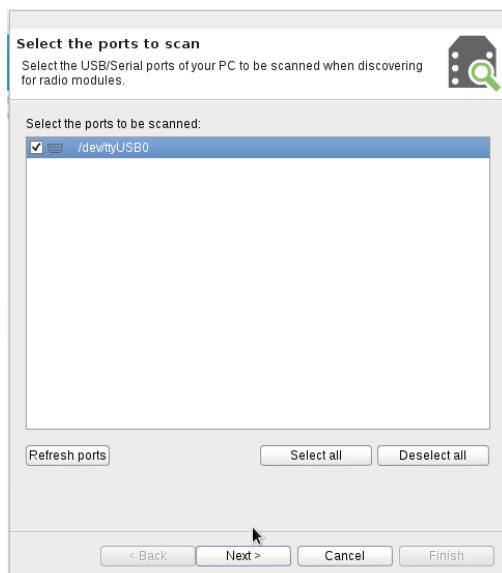


Figura 26: Indique el puerto que el programa muestreará.

Se abrirá una segunda ventana donde se pueden indicar varias configuraciones a evaluar, como el de la figura 27 de la página 18. 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.

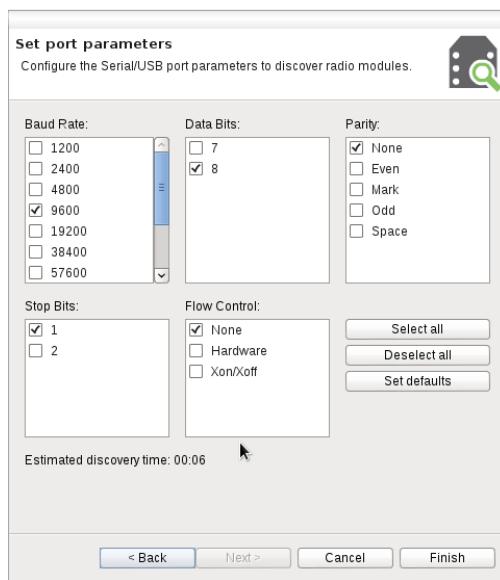


Figura 27: Ventana de configuración de búsqueda.

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

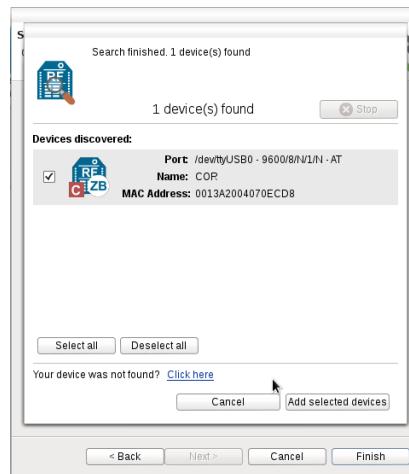


Figura 28: Lista de XBEE encontrados por la herramienta de búsqueda.

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 29.

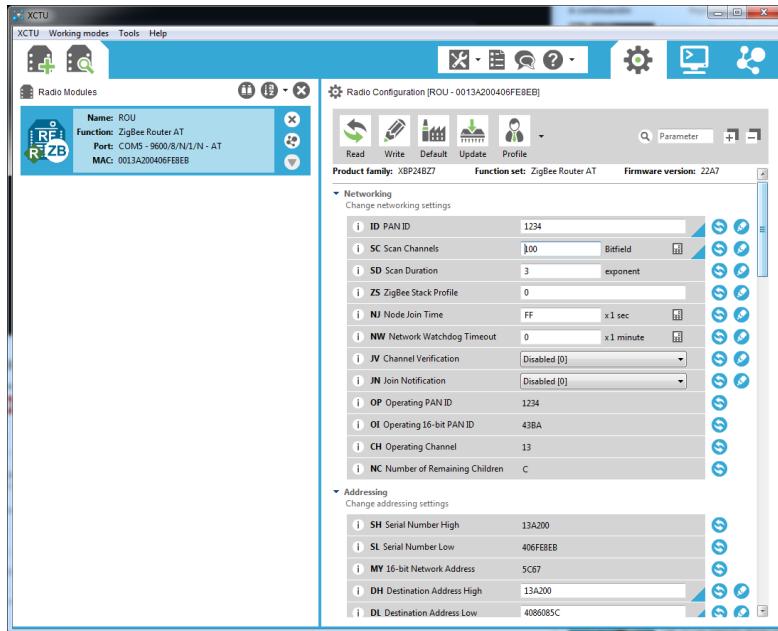


Figura 29: Ventana principal de XCTU con un XBEE añadido.

3.3. Programación del XBEE

3.3.1. Configurando los modos

Una vez asociado por lo menos un XBEE a XCTU, como se observa en la figura 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 30.

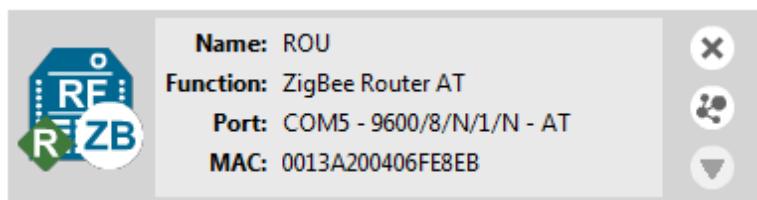


Figura 30: Información del XBEE asociado a XCTU.

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 30 de la página 19, 0013A200), y los últimos ocho como la parte baja de la dirección física (en la figura 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.

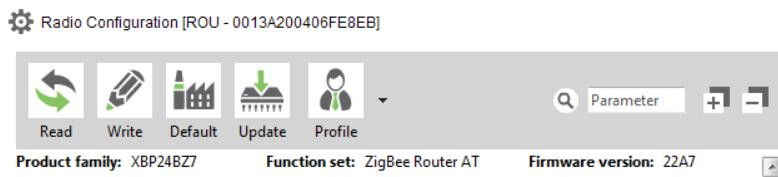


Figura 31: Apartado de información y herramientas.

En la parte superior de esta ventana, se halla un apartado como el de la figura 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 32.

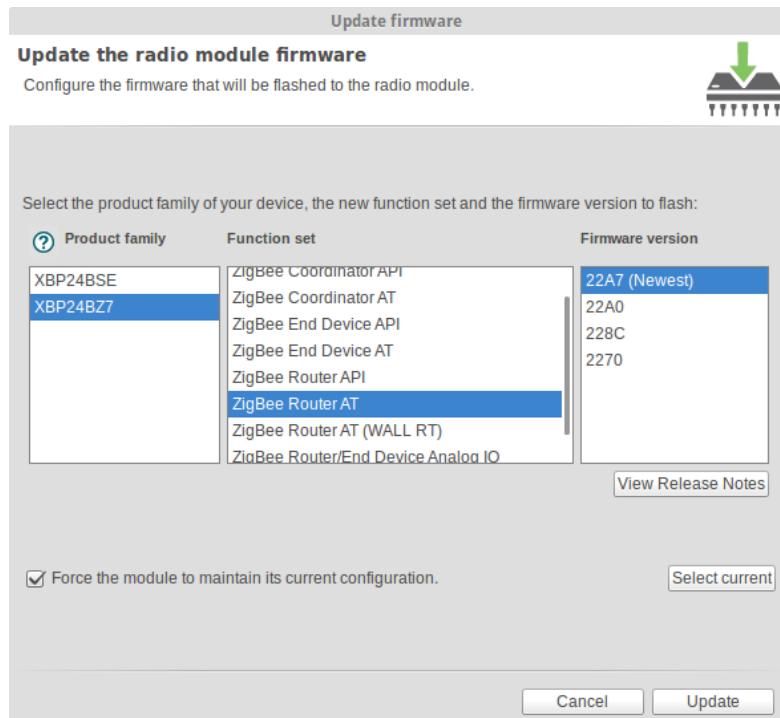


Figura 32: Ventana de programación del firmware.

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 31 de la página 20. 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.

3.3.2. 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.

3.3.3. Configuración mediante comandos AT



Figura 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 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 34), que es lo que a continuación se realizará.

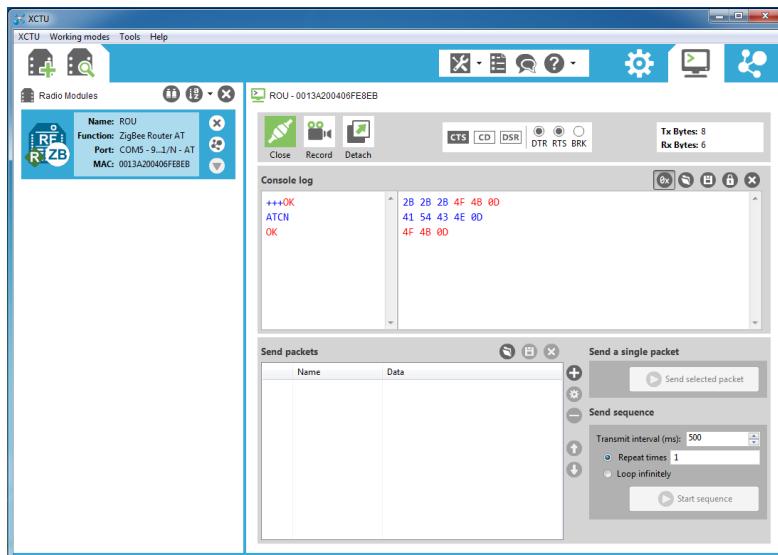


Figura 34: XCTU en modo consola.

Para iniciar con este modo, de clic al botón **Open**, que es como el que se muestra en la figura 35 de la página 24. 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, vuelve 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 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 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 36, que se ubica en la parte superior derecha de la ventana.

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 37 de la página 25.

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



Figura 35: Botón de apertura de comunicación.

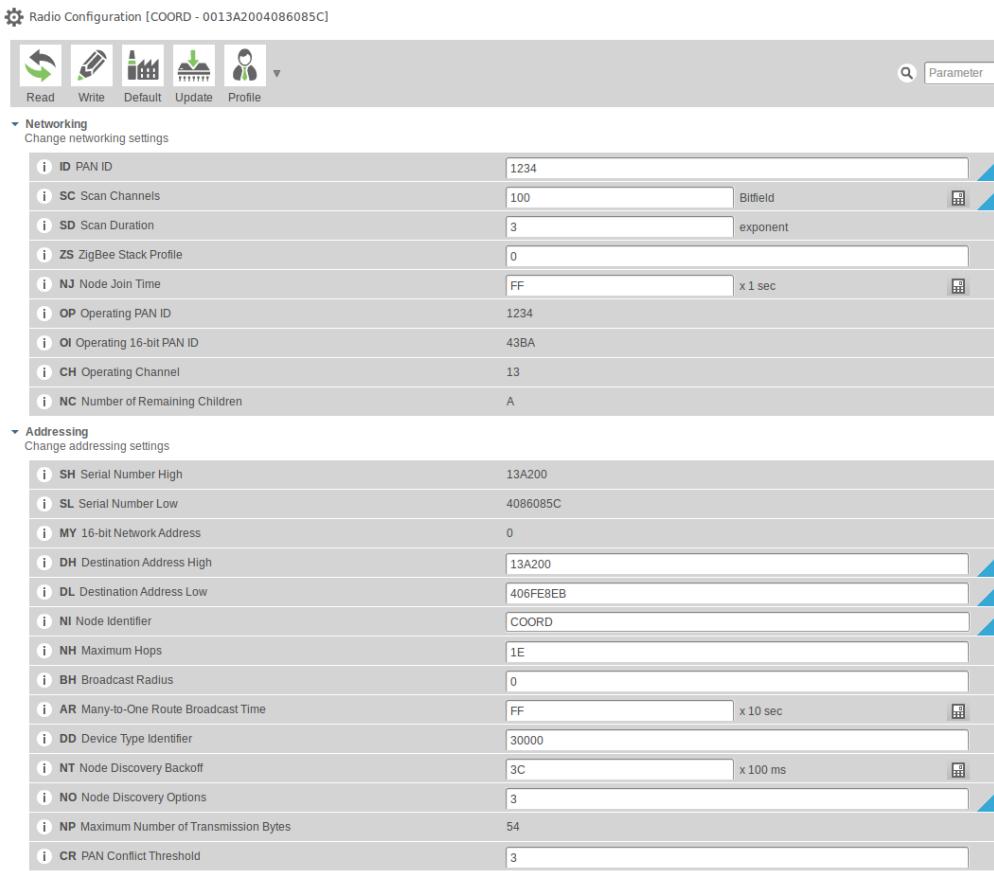


Figura 37: Configuraciones finales en el Coordinador.

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.

3.3.4. 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).

por comandos AT con los tres caracteres de suma e iniciar el proceso de nuevo.

- **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*:

- 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 38 y 39 de la página 27. Ahora la programación y configuración de los módulos XBEE están terminadas.

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 3.3.3. Esto debido a que el firmware de un Router no permite realizar este ajuste aún por comandos AT.

▼ Networking
Change networking settings

ID PAN ID	1234	
SC Scan Channels	100	Bitfield
SD Scan Duration	3	exponent
ZS ZigBee Stack Profile	0	
NJ Node Join Time	FF	x 1 sec
NW Network Watchdog Timeout	0	x 1 minute
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 38: Configuraciones finales en el Router, 1.

▼ Addressing
Change addressing settings

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	x 10 sec
DD Device Type Identifier	30000	
NT Node Discovery Backoff	3C	x 100 ms
NO Node Discovery Options	0	
NP Maximum Number of Transmission Bytes	54	
CR PAN Conflict Threshold	3	

Figura 39: Configuraciones finales en el Router, 2.

4. Prueba: Un chat

4.1. Preparativos

Esta prueba puede realizarse utilizando el modo consola de XCTU en dos computadoras (mire la subsección 3.3.3, 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 40.

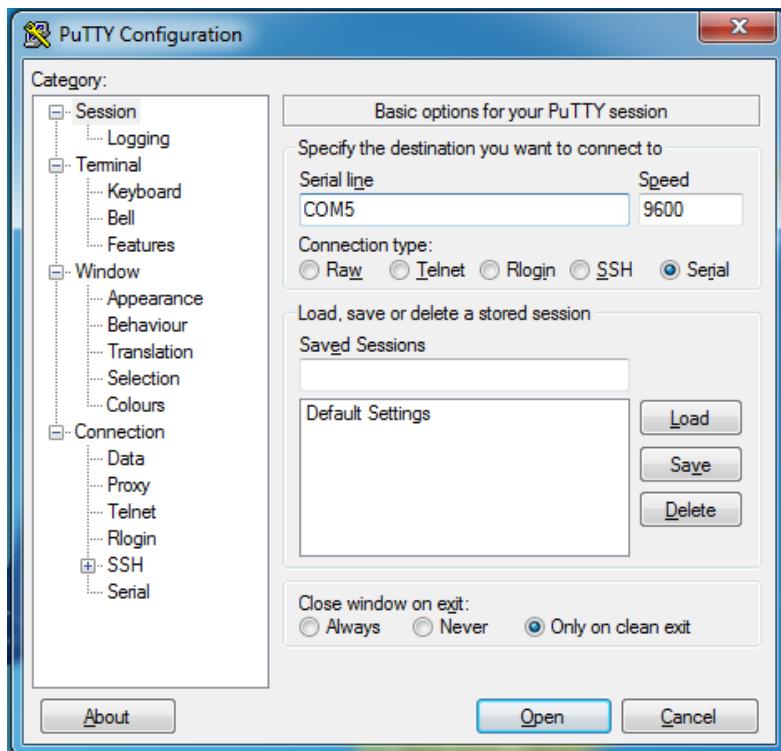


Figura 40: Configuración de sesión en PuTTY.

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

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 3.2.1). En el área denominada *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 41.

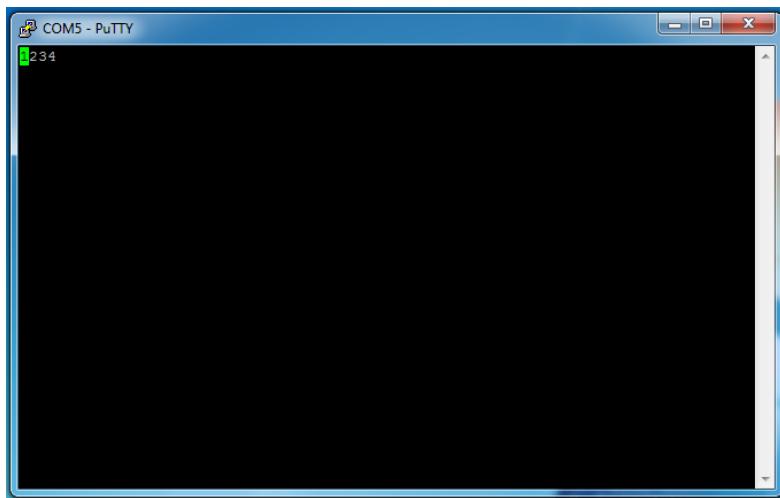


Figura 41: Terminal PuTTY.

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 3.3.3). 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 3.3.3, dejando abierta la consola y verificando que el XBEE responda al comando **+++**, y de ser ese el caso, teclee **ATCN**.

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 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.

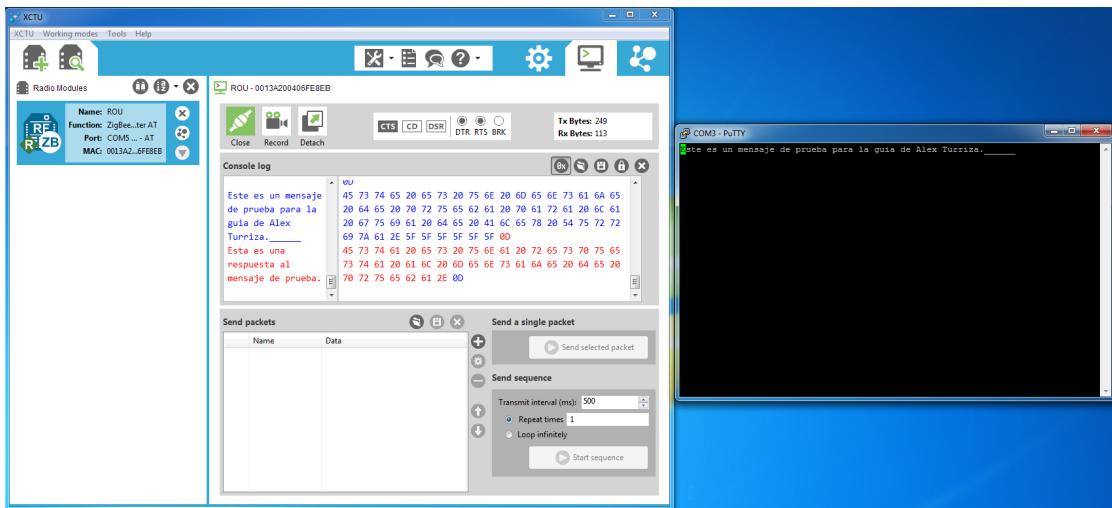


Figura 42: El chat funcionando.

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 3.3.1.

Bibliografía

- [1] Andrés Caballero Paz. «Desarrollo de un controlador midi no convencional, implementado en un sistema embebido, utilizando el kinect». En: (2014).
- [2] Javier Cerrato Miranda. «Diseño e implementación de una aplicación visual para el control de flotas basado en GPS». En: (2011).
- [3] LUCÍA SENCHERMÉS CHÁFER. «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 doct. 2017.
- [4] Gerald Coley. «Beaglebone black system reference manual». En: *Texas Instruments, Dallas* (2013).
- [5] Jorge Coronado Vallés. «Desarrollo de aplicaciones embebidas de control en robots móviles». Tesis doct. 2014.
- [6] Verónica De la Cruz Bautista y col. «Diseño de una estrategia logística de rutas para la distribución de productos farmacéuticos.» Tesis doct. 2011.
- [7] Jorge Fallas. «Sistema de Posicionamiento Global». En: *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* (2002).
- [8] Eduardo Huerta, Aldo Mangiaterra y GUSTAVO Noguera. «GPS: posicionamiento satelital». En: *Rosario: UNR Editora, Universidad Nacional de Rosario* (2005).
- [9] Diego Armando Maldonado Hidalgo, Christian Felipe Ramírez Acosta y Miguel Eduardo Villarreal Quintero. «Controlador de posición para un vehículo aéreo de 4 rotores realimentado por GPS.» En: (2010).
- [10] A Mendoza Diaz y col. «Recomendaciones de actualización de algunos elementos del proyecto geométrico de carreteras.» En: *Publicación técnica 244* (2004).
- [11] Andrés Oyarce, Paul Aguayo y Eduard Martin. «Guía del usuario Xbee series 1». En: *Ingeniería MCI Ltda* (2010).
- [12] Sven Rönnbäck. «Developement of a INS/GPS navigation loop for an UAV». En: *Master's thesis 81* (2000).
- [13] Eldar Rubinov y col. «Review of GNSS formats for real-time positioning». En: *Africa GEO 2011* (2011).
- [14] Tomoji Takasu y Akio Yasuda. «Development of the low-cost RTK-GPS receiver with an open source program package RTKLlib». En: *international symposium on GPS/GNSS*. International Convention Centre Jeju, Korea. 2009, págs. 4-6.
- [15] Dante I Tapia y col. «Identificación por radiofrecuencia: fundamentos y aplicaciones». En: *Proceedings de las primeras Jornadas Científicas sobre RFID. Ciudad Real, Spain* (2007), págs. 1-5.
- [16] Zambrano Termal y Jhon Jairo. «Prototipo de comunicación entre taxis-policía para la seguridad ciudadana». En: (2014).

- [17] B Wiśniewski, Krzysztof Bruniecki y Marek Moszyński. «Evaluation of RT-
KLIB's Positioning Accuracy Usingn low-cost GNSS Receiver and ASG-EUPOS». En: *TransNav: International Journal on Marine Navigation and Safety of Sea Trans-*
portation 7.1 (2013), págs. 79-85.