UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE CUENCA

CARRERA DE INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo
a la obtención del título
de Ingeniero Electrónico

PROYECTO TÉCNICO:
"DESARROLLO DE UN MANUAL DE PRÁCTICAS PARA EL USO DE EQUIPOS LORAWAN EN REDES DE SENSORES INALÁMBRICOS"

AUTOR:

FERNANDO XAVIER VEGA LAZZO

TUTOR:

ING. JUAN DIEGO JARA SALTOS, MgT.

CUENCA - ECUADOR

2020

CESIÓN DE DERECHOS DE AUTOR

Yo, Fernando Xavier Vega Lazzo con documento de identificación N° 0105736896, manifiesto mi voluntad y cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del trabajo de titulación: **"DESARROLLO DE UN MANUAL DE PRÁCTICAS PARA EL USO DE EQUIPOS LORAWAN EN REDES DE SENSORES INALÁMBRICOS"**, mismo que ha sido desarrollado para optar por el título de: *Ingeniero Electrónico,* en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, suscribo este documento en el momento que hago entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, junio del 2020

Fernando Vega

Fernando Xavier Vega Lazzo
C.I. 0105736896

CERTIFICACIÓN

Yo, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: **"DESARROLLO DE UN MANUAL DE PRÁCTICAS PARA EL USO DE EQUIPOS LORAWAN EN REDES DE SENSORES INALÁMBRICOS"**, realizado por Fernando Xavier Vega Lazzo, obteniendo el *Proyecto Técnico***,** que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana.

Cuenca, junio del 2020

Ing. Juan Diego Jara Saltos, MgT.

C.I. 0103543658

DECLARATORIA DE RESPONSABILIDAD

Yo, Fernando Xavier Vega Lazzo con documento de identificación N° 0105736896, autor del trabajo de titulación: **"DESARROLLO DE UN MANUAL DE PRÁCTICAS PARA EL USO DE EQUIPOS LORAWAN EN REDES DE SENSORES INALÁMBRICOS"** certifico que el total contenido del *Proyecto Técnico,* es de mi exclusiva responsabilidad y autoría.

Cuenca, junio del 2020

Fernando Xavier Vega Lazzo

C.I. 0105736896

AGRADECIMIENTOS

Lo primero de todo, me gustaría agradecer mi tutor Juan Diego Jara por sus valiosos comentarios y opiniones para el desarrollo de este Trabajo de titulación y a los docentes que me ayudaron con ideas. Finalmente me gustaría agradecer personalmente a mi mamá por su apoyo incondicional y por haber creído en mí.

Fernando Xavier Vega Lazzo

DEDICATORIAS

Este trabajo de titulación lo dedico a mi persona por el esfuerzo, dedicación y sacrificio que representa cualquier objetivo que se impone en nuestra vida.

Fernando Xavier Vega Lazzo

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

GLOSARIO

ABP: Activación por Personalización.

AppEUI: Identificador de la Aplicación.

AppSkey: Clave de Sesión de Aplicación.

ARCOTEL: Agencia de Regulación y Control de las Telecomunicaciones.

BSON: Binary JSON.

CSS: Espectro Ensanchado por Chirp – Chirp Spread Spectrum.

DevAddr: Dirección del Dispositivo.

DevEUI**:** Identificador de dispositivo final globalmente único.

E/S: Entradas o Salidas

ID: Identificación de Dato.

IMT: Telecomunicaciones Móviles Internacionales.

JSON: Notación de objeto de JavaScript - JavaScript Object Notation

LoRa: Largo alcance – Long Range.

LoRaWAN: Red de área Inalámbrica de Largo Alcance – Long Range Wireless Area Network.

MAC: Capa de control de acceso a medios- Media Access Control.

MCC: MPLAB Code Configurator

NwkSkey: Clave de Sesión de la Red.

Npm: Node Package Manager.

OTTA**:** Activación por Aire.

UDBL: Uso Determinado en Bandas Libres.

VM: Virtual Machine – Máquina Virtual.

WSN: Red de Sensores Inalámbricos.

RESUMEN

En el presente trabajo de titulación como objetivo es el desarrollo de un manual de prácticas las cuales están orientadas al manejo y configuración de los equipos LoRaWAN.

El primer capítulo se centra en los fundamentos teóricos de la tecnología LoRaWAN, se detalla sus características principales entre una de ellas las bandas de frecuencias en las que se puede utilizar, los tipos de dispositivos existentes y la estructura en la que se implementa esta tecnología.

En segundo capítulo se enfoca en la descripción de los equipos LoRaWAN que se encuentran en el laboratorio, se detalla su funcionamiento y las partes del equipo.

En el tercer capítulo se describe el desarrollo del sistema de recepción, especificando los softwares para su desarrollo, y los módulos que se implementan para el funcionamiento del Backend y Frontend.

En el cuarto capítulo se describe las prácticas de laboratorio, enfocándose en explicar la configuración de los programas que se utilizan para los equipos LoRaWAN, la especificación de los módulos y periféricos requeridos para una configuración especifica.

El quinto capítulo se presenta la conclusión del trabajo realizado

Finalmente, en los anexos, se adjuntas los módulos de prácticas donde se especifica las instrucciones y el desarrollo para los estudiantes.

INTRODUCCIÓN

En los últimos años, ha habido un interés creciente en la red de área Inalámbrica de Largo Alcance (LoRaWAN), por su bajo consumo, ancho de banda escalable, alta robustez y su capacidad de amplia cobertura, de forma que aprender a utilizar esta tecnología es un paso importante para un buen desarrollo académico [1].

La tecnología LoRaWAN hace hincapié en la comunicación de largo alcance con la alta capacidad de sensibilidad de recepción que le permite trabajar bajo la interferencia de ruido o el piso de ruido de manera efectiva. Además, la seguridad del sistema LoRa puede garantizarse ya que la transmisión se extiende de forma pseudoaleatoria que se presenta como un ruido, por lo tanto, la técnica de modulación proporciona la seguridad básica para un sistema LoRa [1], [2].

Esta tecnología puede ser implementada en las redes sensores inalámbricos teniendo en cuenta su estructura, su tipo de conexión y sobre todo la aplicación en la cual se implementará.

Este proyecto presenta los conceptos más importantes de la tecnología LoRaWAN, la utilización de los módulos LoRaWAN y la configuración de los mismos.

ANTECEDENTES DEL PROBLEMA DE ESTUDIO

En la actualidad, las comunicaciones inalámbricas son las más utilizadas en hogares, oficinas, universidades, etc. y con el avance de la tecnología, hoy en día, cualquier dispositivo (sensores, actuadores) pueden conectarse a una red de comunicación a través de las llamadas Redes de Sensores Inalámbricas (WSN) [1]. El objetivo generalmente es enviar información específica a los usuarios o a un servidor para mantener monitoreo y/o control de eventos dentro de la red. Dicho esto, una de las tecnologías que se muestra como alternativa para el despliegue de este tipo de conexiones y que ha ganado peso es LoRaWAN (que significa red de área de largo alcance) que aplica una técnica de modulación derivada de las tecnologías de espectro ensanchado por chirp (CSS) en la transferencia de datos [1] [2].

Por otra parte, la Universidad Politécnica Salesiana – Sede Cuenca, ha adquirido diversos equipos para el análisis, desarrollo y experimentación con diferentes tecnologías para el uso de WSN. Entre estas tecnologías está LoRaWAN que presenta oportunidades como integración con otras tecnologías (industria, biología, agricultura), menor uso de recursos, mayor cobertura.

JUSTIFICACIÓN (IMPORTANCIA Y ALCANCES)

La Universidad Politécnica Salesiana - Sede Cuenca, realizó una inversión para equipar el laboratorio de Telecomunicaciones con equipos de tecnología LoRaWAN; con el objetivo de impulsar el desarrollo Tecnológico en dispositivos de comunicaciones para Internet de las Cosas, este proyecto de titulación pretende desarrollar un manual de guías de prácticas de laboratorio para pregrado, que especifiquen paso a paso las actividades para desarrollarlas. Todo esto con la finalidad de tener una Guía paso a paso y escalable, de manera que el estudiante pueda comprender la parte práctica desde un enfoque básico hasta un enfoque avanzado, en lo que respecta a la implementación de una red de sensores. Se pretende generar varias prácticas de manera que el estudiante entienda desde el montaje del hardware, hasta la recepción de los datos en tiempo real en una plataforma básica de Software que se desarrollará mediante programación con sockets.

OBJETIVOS

OBJETIVO GENERAL

- Desarrollar un manual de prácticas para el uso de equipos LoRaWAN en redes de sensores inalámbricos para la Universidad Politécnica Salesiana - Sede Cuenca.

OBJETIVOS ESPECÍFICO

- Analizar a profundidad el estándar de comunicación LoRaWAN.
- Identificar características, componentes y puertos del paquete LoRa(R) Technology Evaluation Kit-800.
- Realizar pruebas de funcionamiento de la tarjeta LoRa Gateway Core Board y tarjeta LoRa Gateway Radio Board.
- Realizar una revisión sobre el uso de las bandas de frecuencias para que se puedan utilizar estos equipos con fines industriales.
- Diseñar prácticas enfocadas en redes de sensores inalámbricos a través del uso de LoRa(R) Technology Evaluation kit-800.
- Desarrollar una aplicación básica de recepción de datos usando código abierto.
- Estructurar y desarrollar el módulo de guías de prácticas para redes y sensores.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA O ESTADO DEL ARTE

Este capítulo se centra principalmente en la teoría que se requiere para la comprensión de las redes de sensores inalámbricos y su implementación mediante dispositivos LoRaWAN, además se revisa las características principales de una red de sensores.

1.1 REDES DE SENSORES INALÁMBRICOS

Hoy en día con los avances de la tecnología, se han podido realizar diseños de sensores, multifuncionales, de bajo consumo, inalámbricos y de costos accesibles, que pueden ser incorporados en una red de sensores [5].

En [6] los autores definen una Red de sensores inalámbricos o WSN, por sus siglas en inglés, Wireless Sensor Network, como un conjunto de nodos compuestos por un microcontrolador, diversos sensores, dispositivos de comunicación, que permiten recoger y transmitir datos del entorno físico en el que se encuentran.

En otras palabras una WSN es un grupo de nodos finales distribuidos, que poseen la capacidad de monitorear, procesar y transmitir los parámetros físicos o ambientales a un nodo central o Gateway que es el encargado de recibir los datos que generen los nodos finales, de igual manera, sirven como puntos de acceso a otras redes o hacia un servidor central de procesamiento de datos [6]–[9].

1.1.1 CARACTERÍSTICAS GENERALES DE UNA RED DE SENSORES INALÁMBRICOS

Las WSN son redes ya estandarizadas por la ITU-T, tienen diferentes ámbitos de aplicación, debido a que dan la posibilidad de desarrollar los nodos de manera personalizada para su utilización en un área específica y adaptar la topología de la red a las necesidades del problema, hay que tener en cuenta que no existe un patrón específico para la distribución de nodos, por lo tanto, no existe una topología determinada para este tipo de red.[6], [8], [9]

Las características más destacadas de las WSN se pueden decir que es su baja tasa de transferencia, bajo consumo, escalabilidad y seguridad, además, los componentes de una WSN necesariamente requieren estar conectados a una red pública [6], [8].

1.2 MODULACIÓN DE LARGO ALCANCE (LORA)

La modulación de largo alcance o LoRa, por sus siglas en inglés, Long Range, define únicamente la capa física y a su vez proporciona las característica de modulación y las especificaciones como ancho de banda, frecuencia portadora, y potencia que debe de cumplir para poder realizar una emisión [4], [10], [11].

LoRa está basada en una modulación de espectro ensanchado con chirps o CSS, por sus siglas en inglés, Chirp Spread Spectrum, este tipo de modulación utiliza una señal chirp que varía continuamente en frecuencia para lograr el espectro ensanchado. Un parámetro importante es el Spread Factor, con él se configura la duración de cada símbolo y el ensanchamiento que se le otorga, este tipo de modulación aporta con una gran robustez a interferencias y una mejora en la sensibilidad [4], [7], [10], [11].

Las propiedades principales de LoRa de acuerdo con [4] y [7] son:

- Ancho de banda Escalable

- Baja potencia
- Alta robustez
- Capacidad amplia de cobertura
- Capacidad de red mejorada

1.3 TECNOLOGÍA LORAWAN

LoRaWAN es la capa de acceso al medio para las LP-WAN, define el protocolo de comunicación y la arquitectura de la red, por lo tanto, determina de forma decisiva la vida útil de la batería de los nodos, la calidad del servicio, los tipos de aplicaciones para la red, y su seguridad [1], [4], [7], [10].

1.3.1 CLASES DE DISPOSITIVOS FINALES

LoRa Alliance define tres clases de dispositivos finales según los requerimientos y la aplicación que se realizara, cada dispositivo realiza un compromiso respecto al gasto energético y latencia [7], [12], [13].

- **Dispositivos finales bidireccionales (Clase A):** Estos dispositivos permanecen a la escucha después de cada emisión del Gateway, quiere decir, que permiten una comunicación bidireccional nodo-servidor (uplink) y servidor-nodo (downlink), además con un menor consumo energético.
- **Dispositivos finales bidireccionales con ranuras de recepción programadas (Clase B):** Estos dispositivos permiten más ventanas de recepción adicionales a la clase A, estas ventanas de recepción se abren en tiempos programados o fijos, para ello, se envía mensajes denominados ‘‘beacons’’ desde el Gateway y así fijar el tiempo en modo de escucha del dispositivo final.
- **Dispositivos finales bidireccionales con ranuras de recepción máxima (Clase C):** Son dispositivos que tienen las ventanas de recepción continuamente abiertas, únicamente se cierran al momento de transmitir, posee una fuerte restricción de latencia de recepción y utiliza más energía que las clases A y B.

1.3.2 MODOS DE ACCESO A LA RED LORAWAN

Los dispositivos LoRAWAN ya sean de cualquier clase para poder ser parte de una red LoRa deben ser activados, los modos de activación que pueden tener estos dispositivos pueden ser Activación por Aire (OTTA) y Activación por Personalización (ABP) [1], [14], [15].

1.3.2.1 Activación por Personalización

Este tipo de activación no necesita mensajes de solicitud y aceptación de la red, agrega directamente los dispositivos a la red (***Figura 1***).

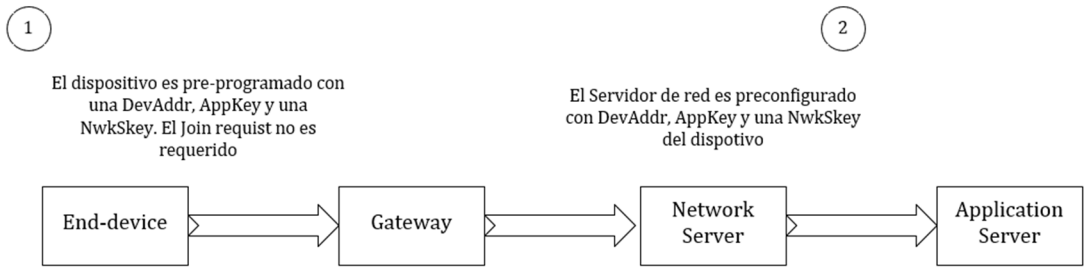


Figura 1: Comunicación entre nodo final y red LoRa con activación ABP.

Fuente: Microchip Technology, Inc.

Para la activación ABP es necesario la clave de sesión de la red (NwkSkey), clave de sesión de aplicación (AppSkey) y la dirección del dispositivo (devAddr), además hay que tener en cuenta que NwkSkey y AppSkey deben ser únicas dentro de la red. El hecho de que se agreguen los dispositivos directamente, es debido a que en el servidor de red se preconfiguran las claves NwkSkey, AppSkey y devAddr [7], [13]–[15].

Tabla 1: Descripción de parámetros para una activación ABP

Clave	Descripción
DevAddr	Es el identificador del dispositivo, que se utiliza durante toda la comunicación
NwkSkey	Se utiliza para encriptar y desencriptar de la carga útil en mensajes MAC
AppSkey	Se utiliza para encriptar y desencriptar de la carga útil en mensajes de datos específicos de una aplicación (FRMPayload)

1.3.2.2 Activación por Aire

En este tipo de activación si se requiere de un mensaje de solicitud y aceptación para que el nodo final pueda ingresar a la red LoRa (*Figura 2*).

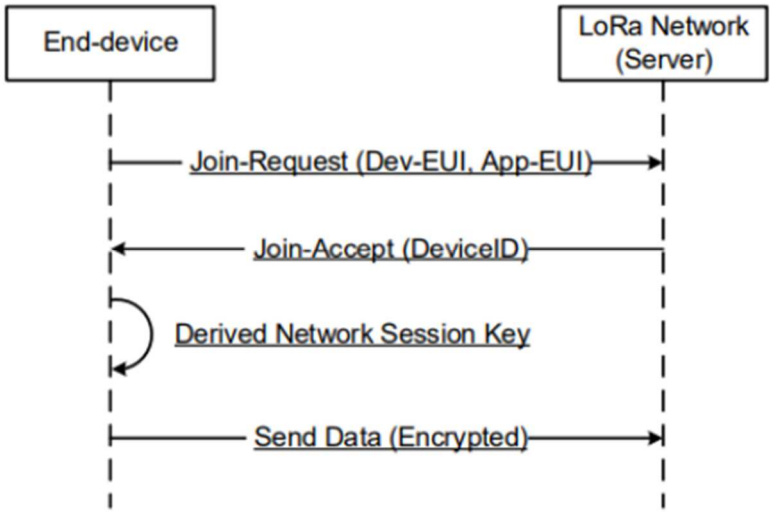


Figura 2: Comunicación entre nodo final y red LoRa con activación OTAA.

Fuente: Microchip Technology, Inc.

El procedimiento para la activación OTAA en los dispositivos LoRaWAN requiere del envió de un mensaje de ‘‘Join Request’’ que contiene el Identificador de dispositivo final globalmente único (DevEUI), el Identificador de la aplicación (AppEUI) y la Autenticación con clave de aplicación (AppKey). El servidor de red envía un mensaje de ‘‘Join Accept’’ al nodo final, este lo verifica y descifra, para extraer la dirección del dispositivo y así conectarse a la red para él envió de datos.[7], [13], [15]–[17]

Tabla 2: Descripción de parámetros para una activación OTAA

Clave	Descripción
DevEUI	Es el identificador del dispositivo (este debe ser único)
AppEUI	Identificador de aplicación único de 64 bits, utilizado para clasificar los dispositivos por la aplicación a la que pertenecen
Appkey	Es una clave AES-128 única del dispositivo

1.4 ESTRUCTURA DE LA RED LORA

En una red LoRa (*Figura 3*) su estructura está definida por los servidores de red que centralizan y procesan la información, los concentradores (gateways) que hacen de punto de acceso entre el servidor , y los nodos finales, que se encargan de recolectar la información y transmitirla, además el tipo de topología más común que se utiliza en estas redes es estrella como se puede ver en la Figura 3 [6].

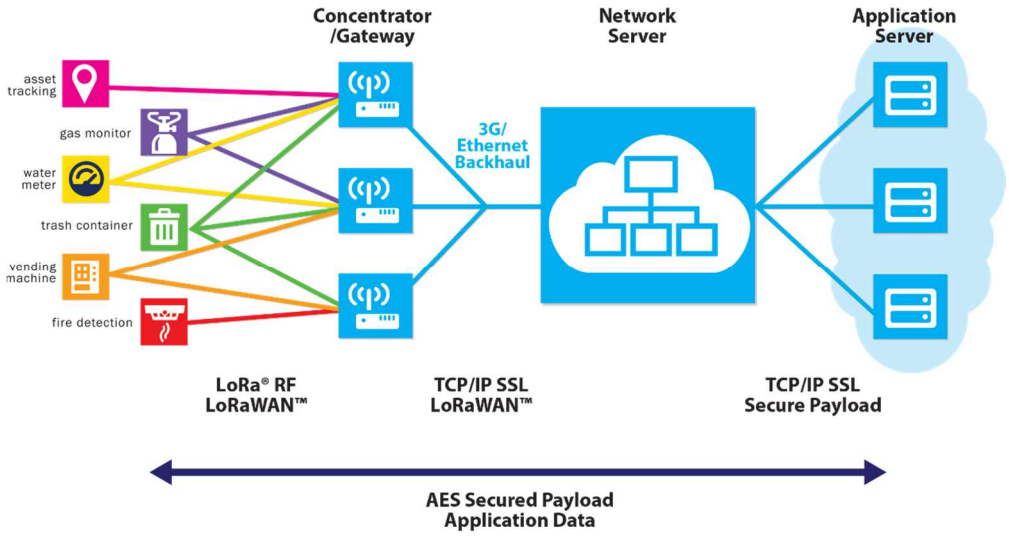


Figura 3*:* Topología de una Red LoRaWAN, especificada por LoRa Alliance

Fuente: LoRa Alliance

La estructura de red en la que se basa este trabajo de titulación se presenta en la Figura 4, la cual consta de un servidor, una base de datos, un concentrador o gateway, los nodos finales y la aplicación.

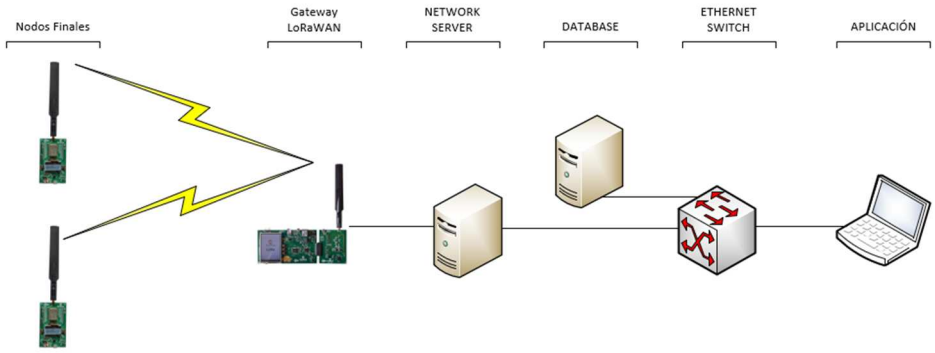


Figura 4: Estructura de la Red LoRa para las prácticas de laboratorio

Fuente: Autor

1.5 BANDAS DE FRECUENCIAS DE UTILIZACIÓN DE EQUIPOS LORAWAN

LoRaWAN opera en bandas de frecuencia no licenciadas, conocidas como bandas ISM (Industrial, Scientific and Medical), son bandas reservadas para uso no comercial, una de las bandas más conocidas es la de 2.4GHz que es utilizada para la tecnología Wi-Fi. En el caso de LoRaWAN utiliza frecuencias más bajas con respecto a Wi-Fi, pero con un mayor rango de alcance, por lo tanto, conlleva a nuevas restricciones que a menudo son especificas en cada país [18], [19].

LoRa Alliance propone un plan de canales regional, se puede ver en la Tabla 3, pero cada país tiene una agencia regulatoria en el caso de Ecuador, es la Agencia de Regulación y Control de las Telecomunicaciones (ARCOTEL), por lo que se debe acatar las normas que tiene esta agencia regulatoria para la implementación de los diferentes equipos [15], [20].

Tabla 3: Plan de Canales Regionales, realizado por LoRa Alliance

Nombre Común	Plan de canales (MHz)
EU868	EU 863-870
US915	US 902-928
CN779	CN 779-787
EU433	EU 433
AU915	AU 915-928
CN470	CN 470-510
AS923	AS 923
KR920	KR 920-923
IN865	IN 865-867
RU864	RU 864-870

Para la utilización de los equipos LoRaWAN en el país se debe revisar primero el plan nacional de frecuencias, que establece las normas para la atribución de las bandas, sub-bandas y canales radioeléctricos para los diferentes servicios de radiocomunicaciones [20], como se puede ver en la Tabla 4 los rangos en los que transmite LoRaWAN ya tienen un servicio definido, con la ayuda de la Tabla 3 y Tabla 4 podemos definir que la banda AU915, es la más óptima para la utilización de los equipos LoRaWAN en el país.

Tabla 4: Rango de Frecuencias Nacionales aplicables a LoRaWAN

Frecuencia MHz	**Descripción**
399,9 - 400,05	Móvil por satélite (tierra-espacio), Radionavegación por Satélite
400,05 - 400,15	Frecuencias patrón y señales horarias por satélite (400,1 MHz)
400,15 - 401	Ayuda a la meteorología, meteorología por satélite (espacio-tierra), Móvil por satélite, investigación espacial.
698 - 806	Móvil mod 5.317, Eqa.85: fijo y móvil (IMT), radiodifusión, fijo
806-890	Fijo, Móvil, Radiodifusión
902 - 928	Fijo, Aficionados, Móvil, Móvil Aeronáutico, Radiolocalización

Además en el informe de canalización de las bandas de 900 MHz [21], ARCOTEL identifico las bandas 869-915 MHz (Uplink) y 940-960 MHz (Downlink) como los rangos de frecuencias para la operación de las telecomunicaciones móviles internacionales (IMT), en el Anexo 2: Comunicado de ARCOTEL - Bandas de Frecuencias, indica todas las bandas utilizadas para las telecomunicaciones móviles internacionales, quiere decir, que esas bandas están en uso y no se pueden utilizar. ARCOTEL en [22] especifica que las bandas 915 – 928 MHz, 2 400 – 2 483,5 MHz, 5 150 – 5 350 MHz, 5 470 – 5 725 MHz y 5 725 – 5 850 MHz y 24,05 – 24,25 GHz operan, a título secundario, sistemas que ocupan espectro radioeléctrico para Uso Determinado en Bandas Libres (UDBL), hay que tener en cuenta que para utilizar las

UDBL para prestación de servicios o como parte de redes privadas se debe sacar un título habilitante [22], por lo tanto, se podría incorporar la banda AU915 que LoRA Alliance propone para la implementación de los equipos LoRaWAN en el país en el caso de la industria, en el caso del equipo LoRa® Technology Evaluation Kit 800 que funciona en las bandas de 400 MHz y 868 MHz como se puede ver en la Tabla 4 y con la confirmación del Anexo 2: Comunicado de ARCOTEL - Bandas de Frecuencias, se confirma que el equipo no podría ser utilizado de manera comercial en el país, este equipo puede ser utilizado para pruebas de laboratorio e investigación dentro de la institución [23].

CAPÍTULO 2: DESCRIPCIÓN DE EQUIPOS

En este capítulo se describe las características, funcionamiento y especificaciones de los equipos utilizados para el desarrollo de las prácticas de laboratorio.

2.1 KIT DE EVALUACIÓN DE LORA - 800

El kit de evaluación de LoRA – 800 (Figura 4) facilita al usuario la utilización y configuración de la tecnología LoRa. El kit este compuesto por la tarjeta LoRa Core Boar, Radio Boars y dos tarjetas MOTEs (nodos finales) que tienen el módulo RN2483.

Figura 5: Kit de Evaluación LoRA – 800

Fuente: Microchip Technology, Inc.

2.2 LORA RADIO BOARD

La tarjeta LoRa Radio (Figura 6) es la encargada de capturar los paquetes ascendentes (nodo-gateway) usando dos transceptores SX157, concentrados en un procesador de banda base SX1301 para la comunicación a través de SPI a la tarjeta LoRa Gateway.

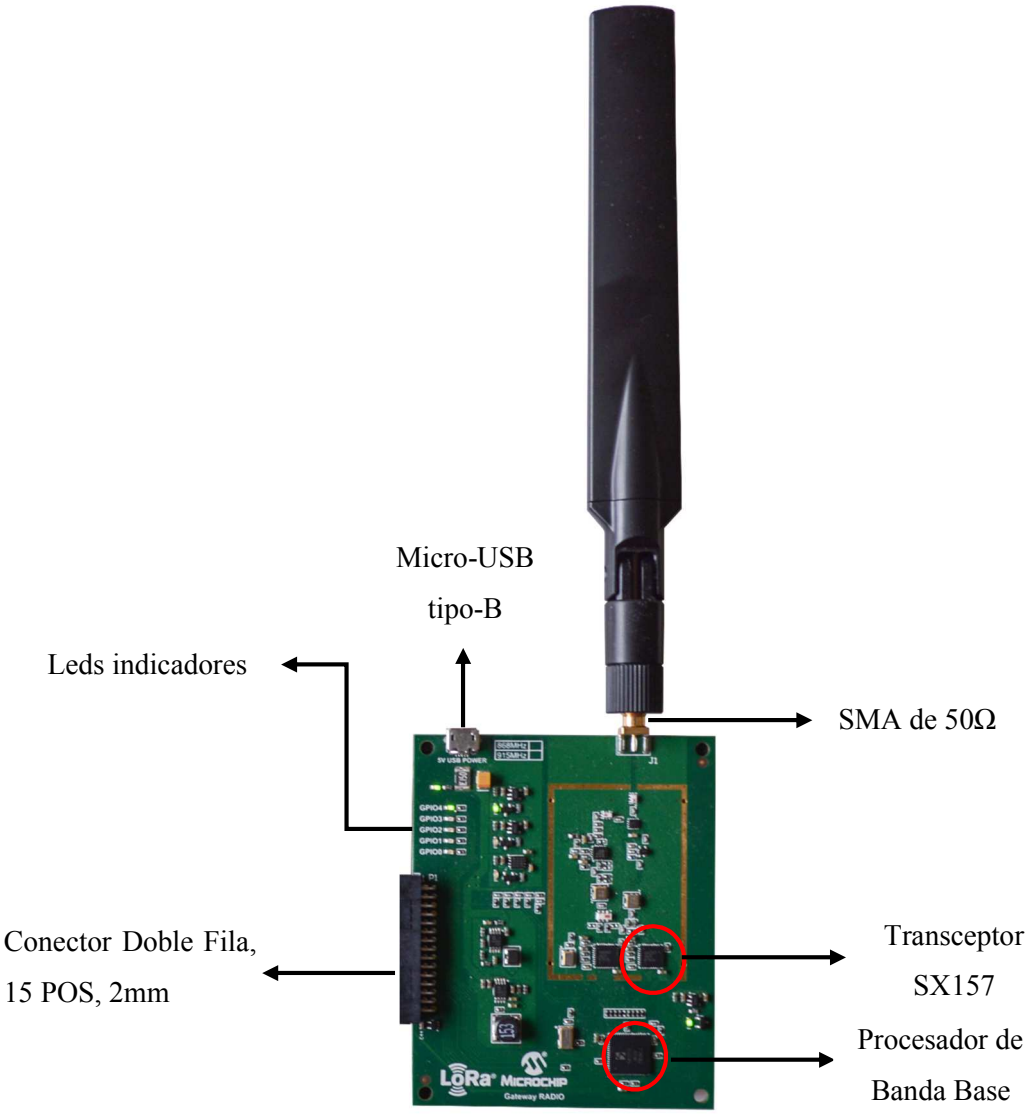


Figura 6: Tarjeta LoRa Radio

Fuente: Microchip Technology, Inc.

2.2.1 DESCRIPCIÓN DEL HARDWARE

La señal RF enviada por el nodo final (MOTEs) es captada por tarjeta LoRa Radio, esta filtra la señal RF por medio de un filtro paso bajo y envía a un switch

RFSW1012 que separa en dos señales RF que se filtran en dos frecuencias diferentes, cada señal es enviada a un transceptor SX157 y su ves estos concentran los datos en el procesador SX1301.

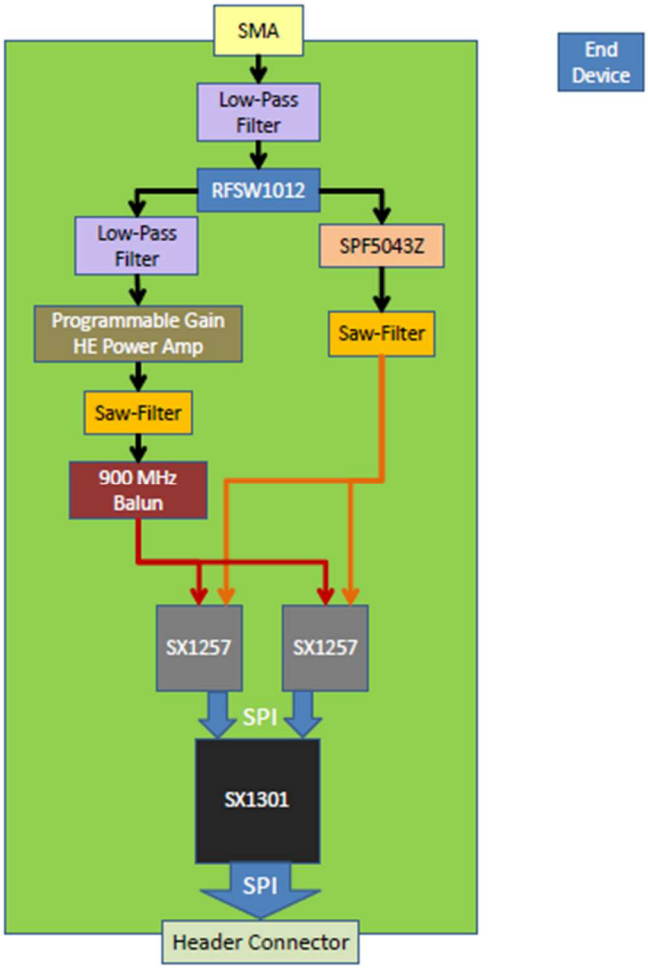


Figura 7: Proceso de Comunicación del hardware de la tarjeta LoRa Radio

Fuente: Microchip Technology, Inc.

2.3 TARJETA LORA GATEWAY CORE

La tarjeta LoRa Gateway Core es una placa demostrativa que puede ser utilizada para el desarrollo de aplicaciones que incorporen los módulos RN de Michrochip. Esta tarjeta es la encargada de recibir los datos capturados de la tarjeta LoRa Radio y de reenviar la información mediante el protocolo TCP/IP al servidor.

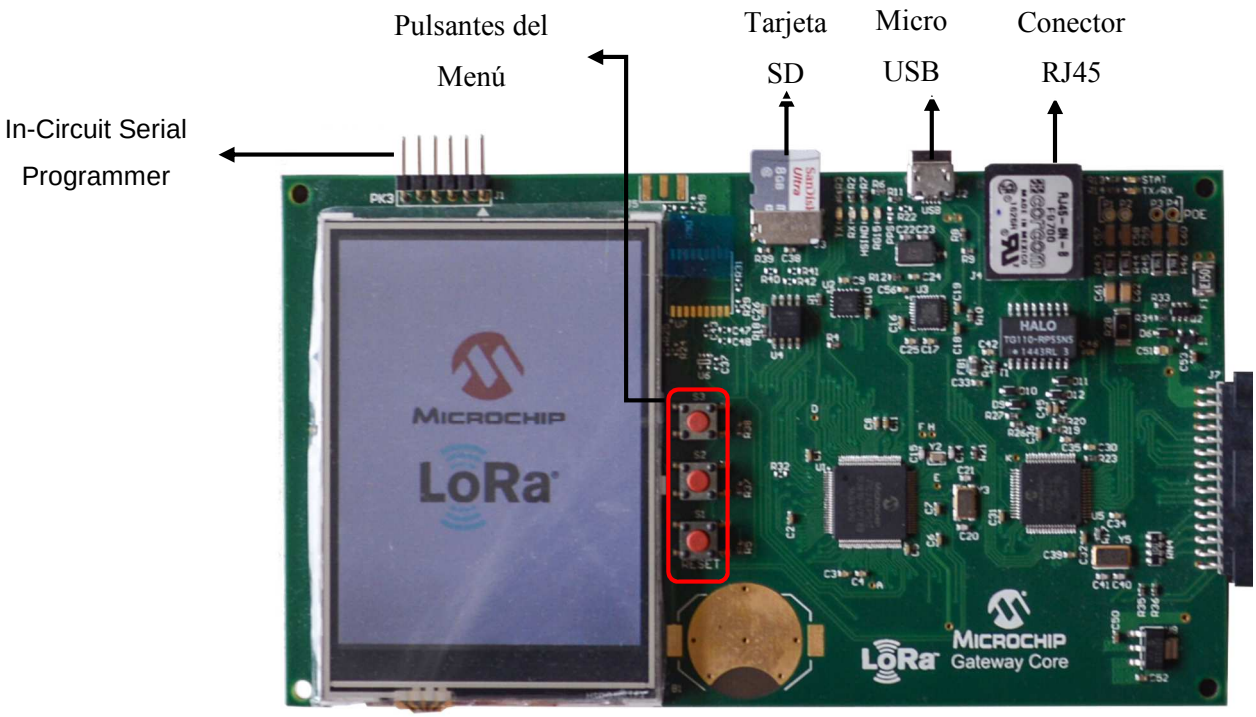


Figura 8: Tarjeta LoRa Gateway Core

Fuente: Microchip Technology, Inc.

2.3.1 DESCRIPCIÓN DEL HARDWARE

La tarjeta LoRa Gateway Core está compuesta por un PIC24EP512GU810 (MCU), este es el responsable de la captura de los datos enviados por la tarjeta LoRa Radio y del reenvió de los datos por medio del controlador Ethernet.

La comunicación entre el MCU y el controlador ethernet se realiza por medio de una comunicación PMP, por sus siglas en inglés, Parallel Master Port, entre el PIC24 y el encapsulado ENC624J600, además el periférico PMP es utilizado por el PIC24 el control de la comunicación y visualización de los datos de la LCD. Esta tarjeta tiene la posibilidad de comunicación con una PC anfitriona, mediante la comunicación USB-Serial, debido al encapsulado USB2412 la información USB es decodificada antes de ser comunicada al MCP2221, además para realizar esta comunicación el MCU tiene pines periféricos dedicados.

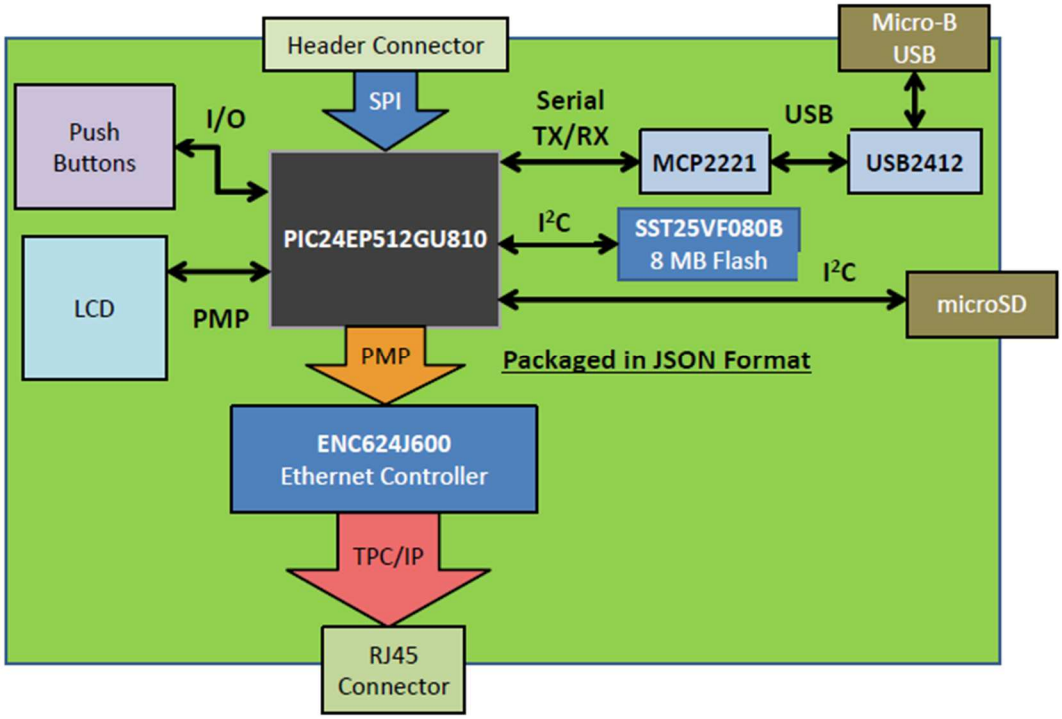


Figura 9: Proceso de Comunicación del hardware de la tarjeta LoRa Gateway Core

Fuente: Microchip Technology, Inc.

2.4 TARJETA LORA MOTE

La tarjeta LoRa MOTE es una placa demostrativa que ayuda al acercamiento a la tecnología LoRa, permite la conexión de una PC por medio de la comunicación USB a UART, que puede realizarse por medio de un PIC 18.

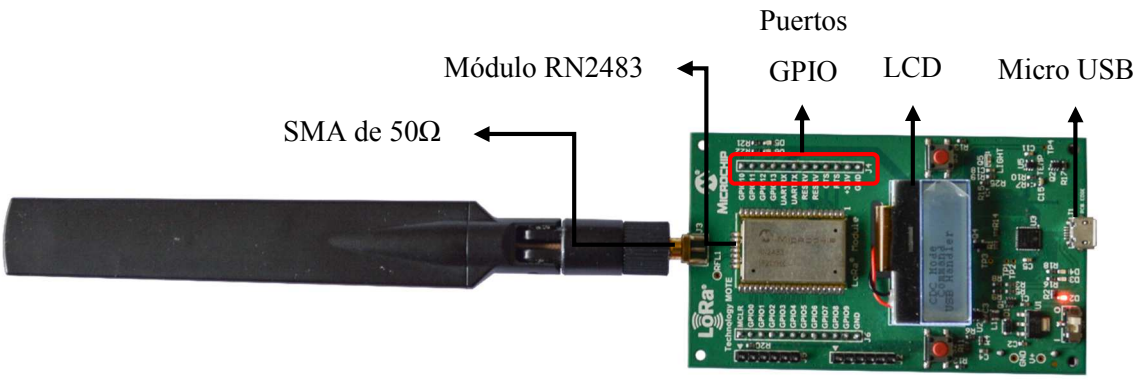


Figura 10: Tarjeta LoRa MOTE

Fuente: Microchip Technology, Inc.

2.4.1 DESCRIPCIÓN DEL HARDWARE

La tarjeta LoRa MOTE está compuesta por un módulo RN2483, que es el encargado de realizar el envió de los datos captados por los sensores conectados en los puertos GPIO, además tiene una comunicación serial Tx/Rx con un PIC18LF45k50 que es el encargado de realizar la comunicación a un host para la configuración del RN2483 con los programas que Microchip dispone en el kit, también dispone de un sensor de temperatura y de luz incorporados los cuales se visualizan en el LCD.

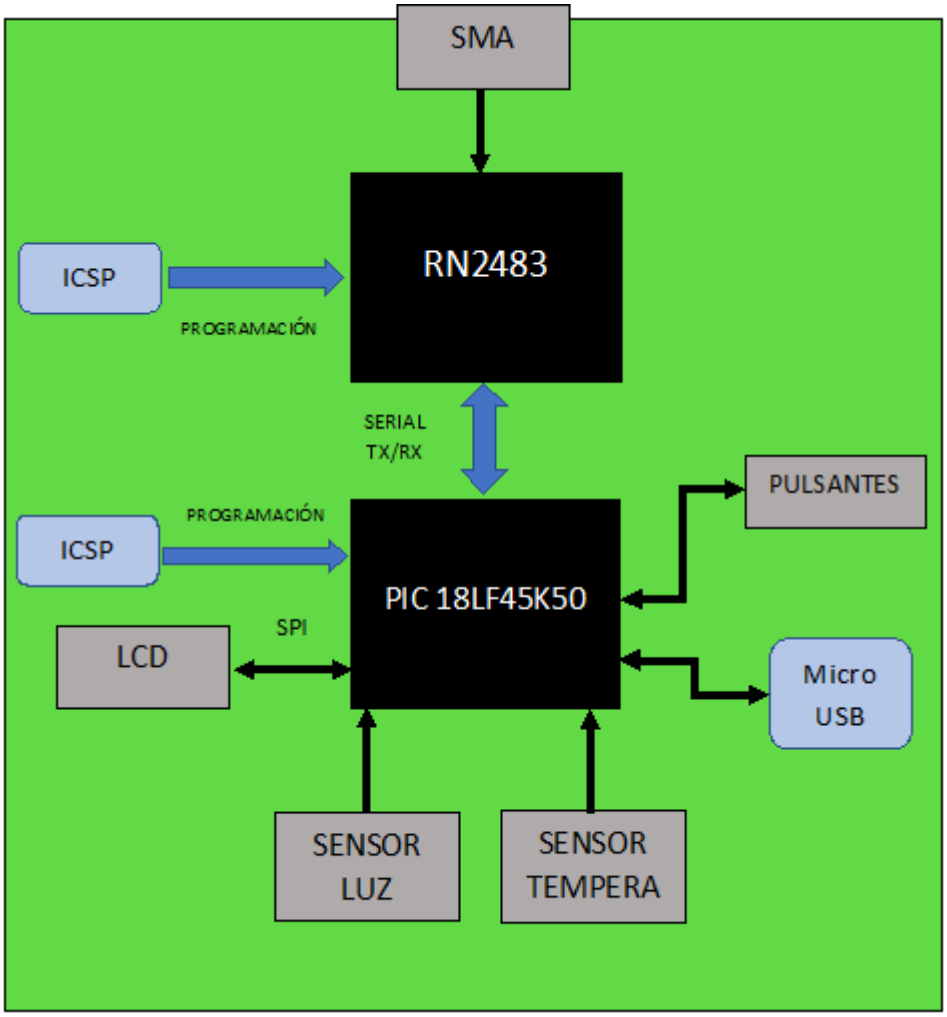


Figura 11: Proceso de Comunicación del hardware de la tarjeta LoRa MOTE

Fuente: Autor

CAPÍTULO 3: DESARROLLO DE APLICACIÓN DE RECEPCIÓN

En este capítulo se describe el desarrollo de la aplicación web para la recepción de los datos enviados por el dispositivo LoRa MOTE. La aplicación se llama S.A.D, que significa Sistema de Adquisición de Datos, esta permite mostrar los datos de los sensores conectados a la red de los diferentes usuarios, en la Figura 12 se representa como está formada la red desde el punto de vista de desarrollo web.

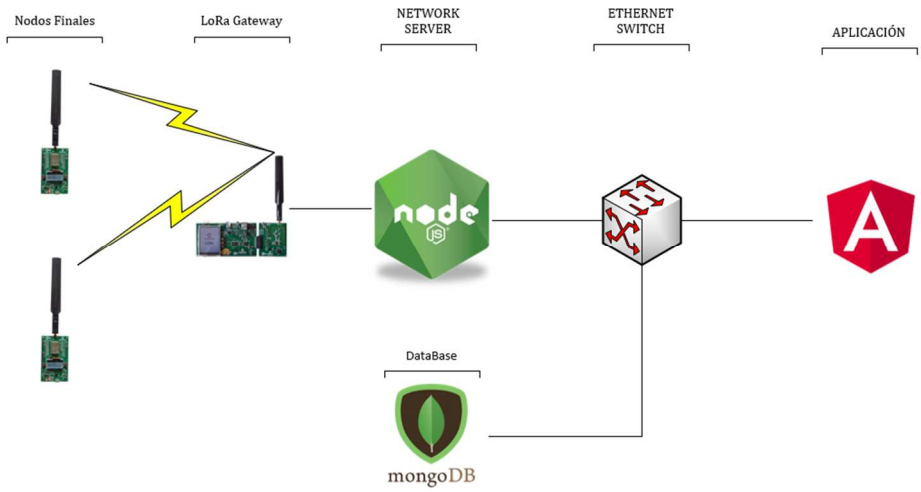


Figura 12: Red LoRaWAN vista desde un punto de desarrollo web

Fuente: Autor

En el siguiente diagrama de flujo se puede apreciar el funcionamiento del sistema S.A.D.

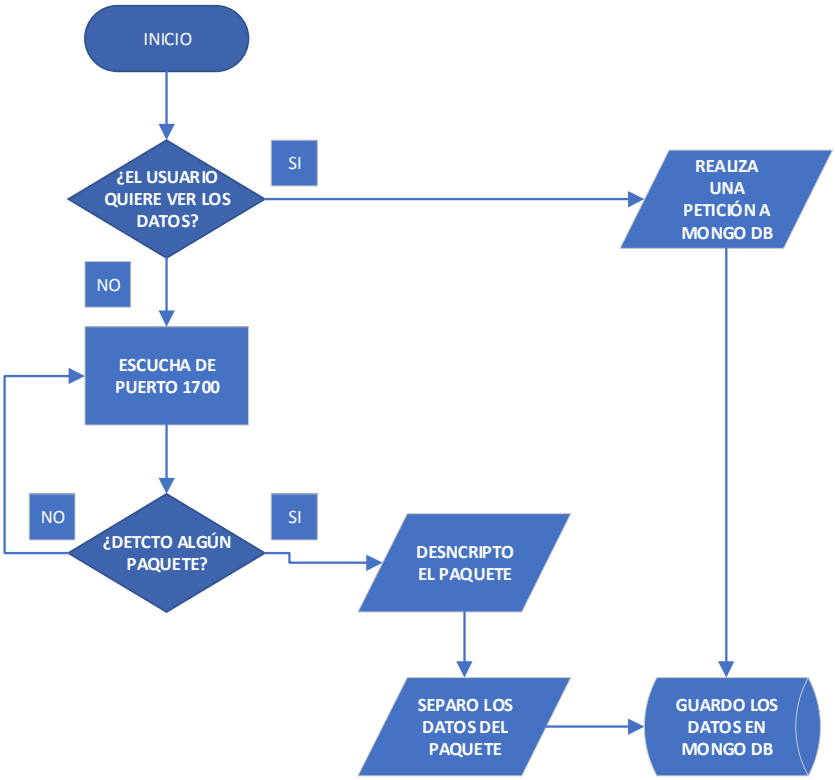


Diagrama de Flujo 1: Funcionamiento de sistema S.A.D

Fuente: Autor

3.1 NODE JS

Una de las características más importantes de Node Js es que está impulsado por eventos, por lo tanto, el código de programación responderá a un evento o disparara un evento, además estos eventos son asíncronos ya que puede estar haciendo un número de conexiones dependientes de la red a diferentes servicios [24].

La utilización de Node Js para el desarrollo de esta aplicación es utilizada para el desarrollo del servidor, este se encargará de la recepción de los datos mediante sockets, en el puerto UDP 1700 escucha y recibe los datos que son enviados por una simulación de programación con sockets que reemplaza al Gateway y por el puerto TCP 3900 es donde corre el servidor web de la aplicación.

La instalación de Node Js se detalla en el Apéndice B: Instalación de node js, se recomienda revisar el apéndice D.7 Configuración del Servidor y Apéndice Q: Inicio del Sistema S.A.D.

3.2 MONGO DB

Es una base de datos no relacional y no SQL diseñada para facilitar el desarrollo, además permite almacenar documentos BSON que básicamente son documentos JSON binarios, de manera que es más sencillo el trabajo con este tipo de base de datos.

Para lograr conectarse a la base de datos hay que tener en cuenta el puerto y la dirección de la base de datos, en este caso el puerto es el 27017 y la dirección es la 'local host', la base de datos se llama 'LoRaWAN_Database'. La instalación de Mongo DB se detalla en el Apéndice C: Instalación Base de datos.

3.3 BACKEND

El Backend es el encargado de realizar todos los procesos lógicos, peticiones a las bases de datos, conexión con el servidor, entre otras tareas, en otras palabras, se encarga de toda la lógica de una página web y funcionamiento en general.

El backend desarrollado para este proyecto de titulación está formado por un servidor en Node JS y MongoDB como base de datos, como se puede ver la Figura 13, el lenguaje de programación utilizado es JavaScript y se utiliza Visual Studio Code como editor de código.

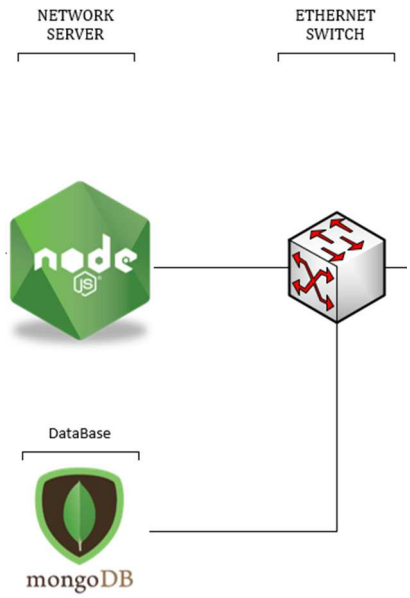


Figura 13: Softwares para el desarrollo del Backend

Fuente: Autor

Para un mejor desarrollo en el backend se realizó la instalación de paquetes con la herramienta 'npm' que es un administrador de paquetes para Node JS, todos los paquetes npm están definidos en archivos package.json dentro de los archivos de registro.

Los paquetes que se utilizan para facilitar los procesos en el backend son:

- **Body-parser:** Permite acceder a la información de las peticiones que se realizan a los servidores.
- **Express:** Proporciona la Escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes caminos URL.
- **Lora-packet:** Permite la desencriptación de los paquetes enviados por la tarjeta MOTE al servidor.
- **Moogoose:** Permite la comunicación entre el servidor y la base de datos.
- **Validator:** Realiza la validación de los datos enviados por la URL.

Para mayor detalle en el Apéndice D: Desarrollo del Backend, se encuentra detallada la programación de cada uno de los módulos.

3.4 MODULOS DEL BACKEND

El backend está diseñado para realizar la comunicación con el Frontend y la base datos, está formado por módulos que ejecutan procesos específicos como el inicio de los sistemas web y la captura de datos por un puerto especifico con una dirección IP especifica.

3.4.1 MODELO DE DATOS

El modelo de datos es la estructura que tienen los datos enviados a Mongo DB, contiene los parámetros necesarios para la identificación del Gateway al cual pertenece, quiere decir, guarda la AppSkey, DevAddr, NwkSkey, ID y la información capturada por el puerto especificado, que son los datos que se almacenan en 'content'; la ID es única dentro de la base de datos y es asignada automáticamente.

Dato

ID: 5e9b3adf107f34045c80d8dc
AppSkey: 1100000F
DevAddr: 3C8F262739BFE3B7BC0826991AD0504D
NwkSkey: 2B7E151628AED2A6ABF7158809CF4F3C
Content: 30 cm

Figura 14: Forma del dato guardado en la base de datos

Fuente: Autor

3.4.2 MÓDULOS

Existen dos módulos que realizan dos procesos el primero es el desencriptado y el segundo la separación de la información principal del paquete enviado.

- **Módulo Desencryp:** Este módulo realiza la desencriptación de los datos enviados por el dispositivo MOTE, para realizar este proceso es necesario la instalación del paquete 'lora-packet', de manera que se obtiene desglosados todos los parámetros.
- **Módulo Separador:** Permite separa la información ya desencriptada (desglosada) en un arreglo de longitud 'n', por lo que permite obtener los datos enviados del sensor de una forma más rápida y sencilla.

3.4.3 RUTAS

Las rutas son las encargadas de poder comunicar el frontend con el backend, además permiten realizar peticiones que permiten modificar los datos guardados en MongoDB, mediante peticiones post, get y delete.

3.4.4 CONTROLADORES

Este módulo es el encargado de guardas, eliminar y obtener los datos que se envían a Mongo DB, para realizar los procesos mencionados se deben hacer subprocesos de validación para que el backend pueda decidir el proceso es el que debe realizar.

3.4.4.1 Proceso de Guardar

Los datos que se envían al servidor se deben guardar, para ello el controlador recibe los datos por la URL y realiza la validación según los parámetros del modelo de datos, si la validación de los datos es correcta envía una respuesta de estatus 200, que significa que han sido guardados correctamente en la base de datos.

3.4.4.2 Proceso de Eliminar

Para acceder a los datos que se encuentran en Mongo DB y poder eliminarlos se deben buscar con el ID con el que se guardaron, para realizar este proceso se creó un método que busca el ID, luego realiza la eliminación del mismo y envía una respuesta de estatus 200 o también puede utilizar directamente la herramienta Robo 3T para realizar el mismo procedimiento de manera gráfica. Este proceso solo puede realizar el administrador de la red, el usuario no puede acceder a este proceso.

3.4.4.3 Proceso de Obtener

Este proceso es el encargado de tomar los datos que existen en Mongo DB, para ello se puede identificar por el ID de cada dato o directamente tomar todos los datos existentes, al realizar correctamente la toma de datos se envía una respuesta de estatus 200.

3.5 DESARROLLO DE FRONTEND

El Frontend es la interfaz o el diseño gráfico de un sitio web (Figura 15), se encarga de la visualización de varios tipos de contenido y debe agradar al usuario, además permite la interacción del usuario con el Backend de una manera interactiva y sin percepción de cambio.

Los lenguajes principales para el desarrollo de un Frontend son HTML, CSS y JavaScript, una manera de facilitar el desarrollo es utilizar frameworks y librerías desarrolladas con esos lenguajes de manera que pueda expandir las capacidades para crear una interfaz de usuario, algunos frameworks más reconocidos son Angular, React, Vue entre otros, para esta aplicación se escogió Angular para el desarrollo del Frontend.

Figura 15: Frontend de la aplicación S.A.D

Fuente: Autor

3.5.1 ANGULAR

Angular es un framework y una plataforma de desarrollo para crear aplicaciones eficientes y sofisticadas de una sola página utilizando HTML y TypeScript.

La estructura de una aplicación en angular está basada en un conjunto de bloques de construcción llamados NgModules, permitiendo la compilación de los componentes existentes en la misma.

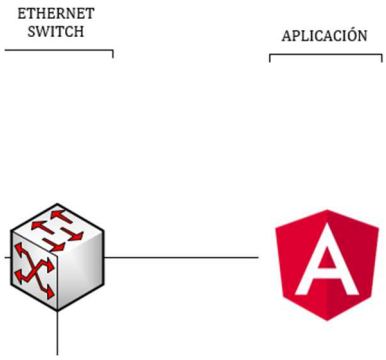


Figura 16: Software para el desarrollo del Frontend

Fuente: Autor

3.5.2 MÓDULOS DEL FRONTEND

3.5.2.1 Componentes

Cada componente define una clase que contiene datos y lógica de la aplicación (Figura 17), y está asociada con una plantilla HTML que define una vista que se mostrará en un entorno de destino. El trabajo de un componente es permitir la experiencia a usuario, este presenta propiedades, métodos para la vinculación de datos y permiten mediar entre la representación gráfica y la lógica de la aplicación.

Figura 17: Inicio de la página principal de la aplicación, con sus componentes.

Fuente: Autor

Los componentes creados para la aplicación son los siguientes:

1. **Header:** Es la cabecera que tiene la aplicación web, aquí se inserta el logo y los parámetros de la barra de menú
2. **Home:** Es la página inicial donde se muestra los componentes de Slider, footer, header.
3. **Sistema:** Es la página donde el usuario puede interactuar con el backend, además permite la observación de los datos guardados en Mongo DB.
4. **Slider:** Es un componente decorador que se agregó para una mejor vista de los títulos de las páginas.

5. **Terminal:** Es el componente que permite ver de una manera más llamativa los datos guardados en Mongo DB.
6. **Footer:** Es un componente decorador, se agregó para una mejor presentación de la aplicación.

3.5.2.2 Servicio

Un servicio es típicamente una clase con un propósito bien definido, separa la funcionalidad relacionada con la vista de un componente de otros tipos de procesamiento realizando una mejora en eficiencia. El servicio que se realizó para esta aplicación es la comunicación del Frontend con Mongo DB mediante peticiones por la URL al servidor.

3.5.2.3 Rutas

Las rutas son las encardas de direccionar de forma instantánea, quiere decir, se pude navegar a las diferentes vistas que posee la aplicación, para realizar ese proceso se debe usar el 'Angular Router' que permite la navegación interpretando la URL de un navegador como una instrucción para cambiar la vista.

Las rutas existentes de esta aplicación son '/inicio' que es la encargada de la vista inicial, '/sistema' que muestra la vista donde se puede hacer la petición a Mongo DB para mostrar los datos y '**' esta ruta es la encargada de mostrar un mensaje cuando no existen rutas definidas.

CAPÍTULO 4: DESCRIPCIÓN DE LAS PRÁCTICAS DE LABORATORIO

Este capítulo está enfocado a describir las prácticas que se elaboraron con la utilización del equipo LoRa(R) Technology Evaluation Kit-800, que se puede ver en el Anexo 1: Prácticas de Laboratorio. El módulo de prácticas desarrollado se planteó de forma que el estudiante pueda comprender de manera más práctica la configuración de los equipos dentro de la red de sensores, considerando desde un enfoque básico a uno avanzado, para lograrlo, se enfocó de manera que se comprenda cuáles son los softwares básicos y que desempeña cada uno de ellos dentro de la red, además la configuración correcta de los mismos y la configuración de los equipos que se utilizan en la red de sensores.

4.1 EXPLICACIÓN DE LOS SOFTWARES PARA EL DESARROLLO DE PRÁCTICAS

En el desarrollo de cualquier práctica lo fundamental es la preparación de todas las herramientas necesarias para su funcionamiento, en este caso particular, son los softwares LoRa Suite, Docker Toolbox y Virtual Box, los cuales, permiten la interacción del usuario con los dispositivos LoRaWAN, capturar y guardar los datos enviados y el arranque del servidor, respectivamente, se puede observar en la Figura 18.

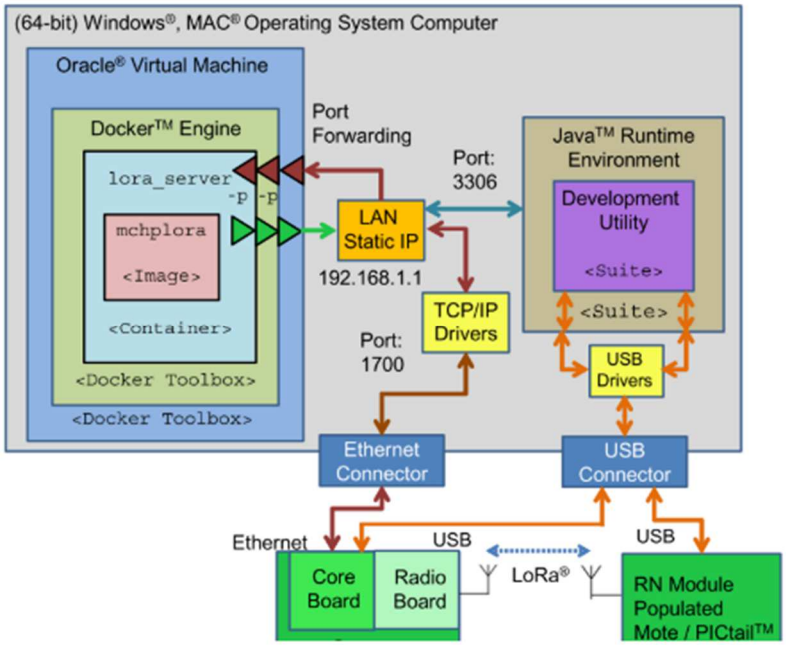


Figura 18: Configuración de la red LoRa de evaluación

Fuente: Microchip Technology, Inc.

4.1.1 LORA SUITE

Es un paquete de instalación basado en Java que contiene los componentes necesarios para la evaluación de un sistema LoRa, además incluye una imagen Docker de instalación y el software LoRa Tecnology Development Utility, que sirve para la evaluación de la red que Microchip propone (Figura 18).

La instalación del paquete LoRa Suite se detallada en el Apéndice E: Instalación de LoRa Utility.

4.1.2 LORA TECHNOLOGY DEVELOPMENT UTILITY

La aplicación presenta una interfaz amigable (Figura 19), permite configurar los dispositivos LoRa Gateway y MOTEs que Microchip dispone, además permite que las instancias de los dispositivos puedan ejecutarse de forma independiente, quiere decir, permite al usuario operar el Gateway y los MOTEs de forma paralela.

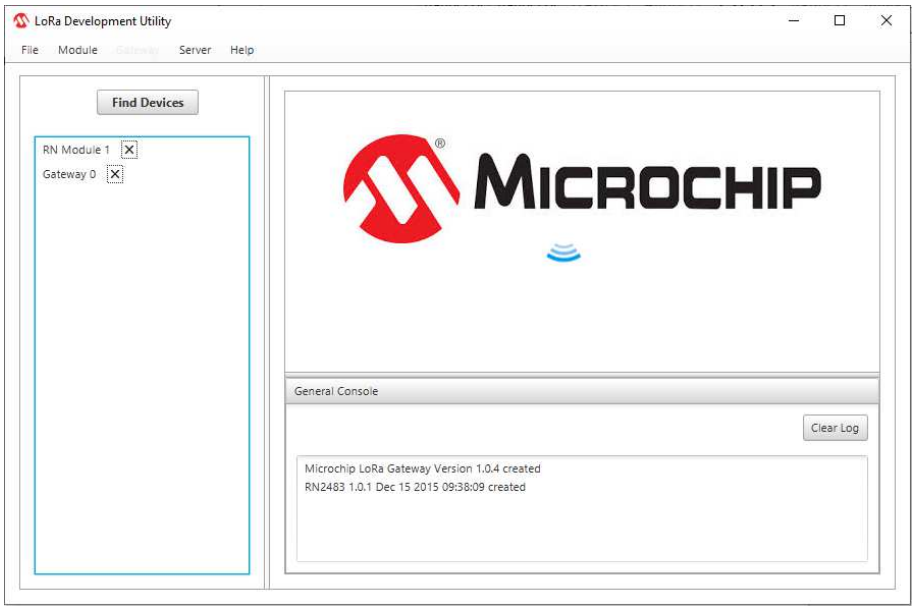


Figura 19: Interfaz Visual de LoRa Tecnology Development Utility

Fuente: Microchip Technology, Inc

El Funcionamiento de LoRa Tecnology Development Utility consiste en escanear todos los puertos COM USB, al encontrar uno o varios puertos disponibles solicita la versión de los dispositivos, dependiendo de la repuesta, el dispositivo es creado como un Módulo RN o Gateway, por lo tanto, cada dispositivo tiene su propia interfaz dentro de la aplicación (Figura 19), en el caso de recibir una respuesta incorrecta o no recibir ninguna respuesta la aplicación considera que no es un dispositivo LoRa y no mostrara los dispositivos en pantalla.

4.1.3 DOCKER TOOLBOX

Esta herramienta proporciona una forma de utilizar Docker en sistemas operativos con Windows que no cumplen con los requisitos mínimos para la instalación de Docker, en la Figura 20 se puede observar la ventana de configuración que permite la instalación de un contenedor o la carga de una imagen Docker.

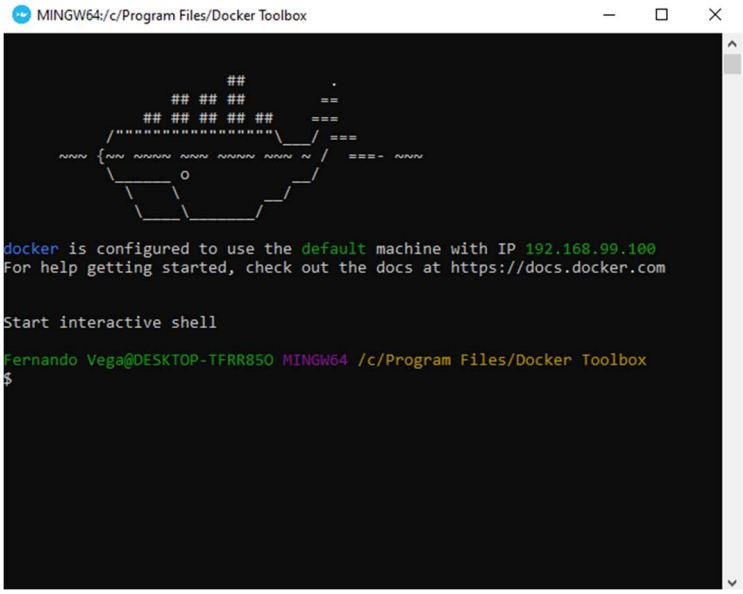


Figura 20: Ventana de comando de Docker Toolbox

Fuente: Autor

Docker permite crear contenedores ligeros y portables para diversos tipos de software, que pueden ejecutarse en cualquier ordenador con cualquier sistema operativo, pero debe tener pre instalado Docker para su funcionamiento, esto quiere decir, el contenedor es el encargado de guardar todos los tipos de softwares, drivers etc., que la aplicación necesite para su funcionamiento y poder llevar la aplicación a cualquier ordenador que tenga instalado Docker.

Los pasos para la instalación se encuentran en el

Apéndice F: Instalación de Docker .

4.1.4 VIRTUAL BOX

Es un software de virtualización que permite la ejecución de otros sistemas operativos (Figura 21) llamados sistemas invitados que son almacenados en el sistema anfitrión, para este caso en particular el sistema anfitrión es Windows 10 y el sistema invitado es Linux.

El motivo de la virtualización es debido a que Microchip recomienda la instalación del Docker dentro del máquina virtual para descartar cualquier error, el sistema que se ejecuta dentro de la máquina virtual permite cargar la imagen del servidor LoRa para tener comunicación con el equipo LoRa(R) Technology Evaluation Kit-800.

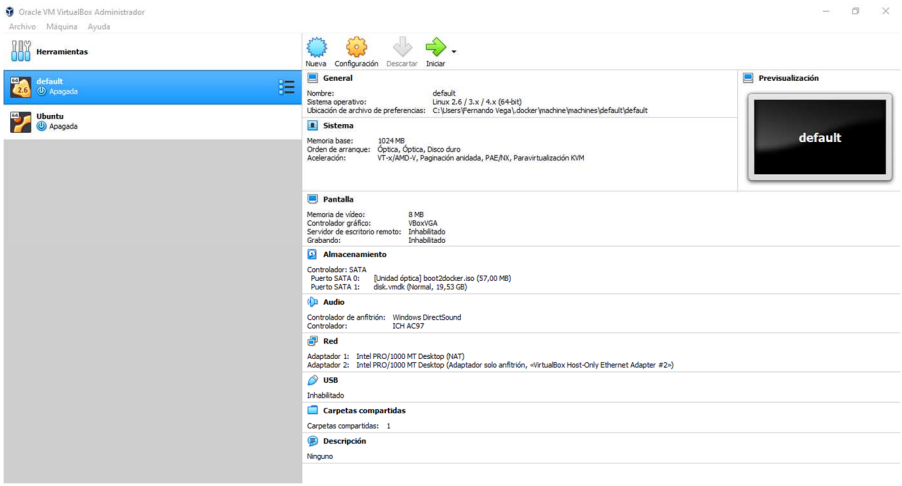


Figura 21: Software Virtual Box

Fuente: Autor

4.1.5 MPLAB X IDE

Este software (Figura 22) permite la configuración de las tarjetas MOTE mediante la selección del microcontrolador que posee el RN2483 y facilita la programación por su lenguaje C, además permite la transferencia del programa conectando los pines de salida del Snap a la tarjeta MOTE, para revisar su instalación véase el

Apéndice M: Instalación de MPLAB X IDE.

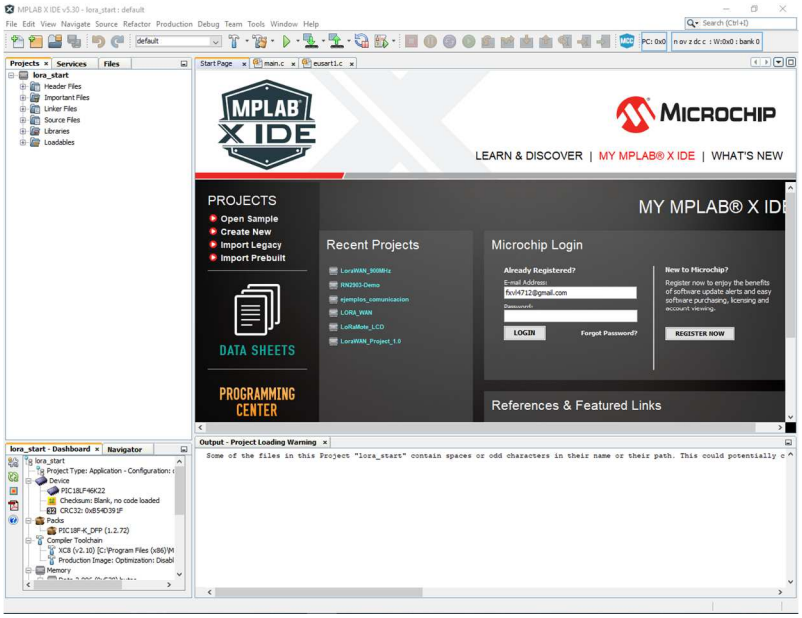


Figura 22: Ventana de inicio del software MPLAB IDE X

Fuente: Microchip Technology, Inc.

4.2 DESCRIPCIÓN DE LA CONFIGURACIÓN DE LOS SOFTWARES

En esta sección se describen las configuraciones correspondientes de los diferentes programas de la sección anterior para el desarrollo de las prácticas de laboratorio.

4.2.1 CONFIGURACIÓN DE REENVÍO DE PUERTOS EN VIRTUAL BOX

La configuración de reenvió de puertos se realiza para el forwarding de datos de enlace descendentes (downlink) y ascendentes (uplink), para ello se debe ir a la configuración de la VM, los datos por defecto que Microchip propone para la red de evaluación se pueden ver en la Tabla 5.

Tabla 5: Valores de configuración de reenvió de puertos

Nombre	Protocolo	Puerto Anfitrión	Puerto Invitado
Customer Server Traffic	UDP	5000	5000
Gateway Traffic	UDP	1700	1700

MySQL Traffic	TCP	3306	3306

Los valores de la Tabla 5 son modificables y hay que tener presente que los valores son modificados, se deben cambiar en el Docker container para la comunicación con el servidor y la aplicación LoRa Development Utility, para realizar la configuración se recomienda revisar el Apéndice G: Configuración de Puertos en Virtual Box.

4.2.2 CONFIGURACIÓN DE DOCKER

El equipo LoRa(R) Technology Evaluation Kit-800 incorpora ya una imagen del servidor que Microchip Technology propone, para utilizar esta imagen se debe cargar desde la interfaz de comando de Docker (Figura 20) y para lograr abrir la imagen del servidor hay que tener la ruta en donde se encuentra, por defecto está en 'C: \Users\ nombre-del-usuario\ Microchip\ LoRa Suite\ Docker', para cargar la imagen en la consola de Docker se debe indicar el comando 'docker load < ruta-por-defecto' y verificar si la imagen se cargó correctamente con el comando 'docker images', además se debe configurar los puertos de reenvió que se vio en la sección 4.2.1 para que pueda tener comunicación la aplicación LoRa Devoloment Utility con el servidor, para la configuración se puede ver en el

Apéndice H: Carga de Imagen Docker.

4.3 DESCRIPCIÓN DE LA CONFIGURACIÓN CON LORA DEVELOMENT UTILITY

4.3.1 CONFIGURACIÓN DE LORA GATEWAY

El software LoRa Develoment Utility permite la configuración del Gateway y las tarjetas MOTEs, para ello se debe conectar las tarjetas por su puerto USB al ordenador (Figura 23) y esperar que el software reconozca el equipo.

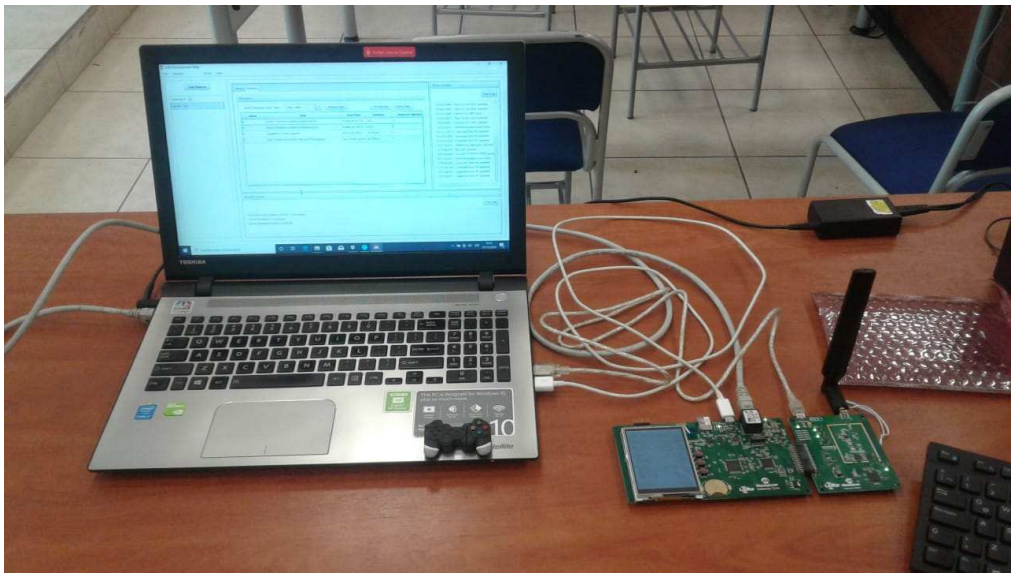


Figura 23: Conexión de la tarjeta LoRa Gateway al ordenador

Fuente: Autor

Para la configuración del Gateway se debe tener presente las direcciones IPs de red, en la Figura 24 se puede ver que la IP de red con cuerda con los datos de la Figura 18, ya que son valores por defecto para probar el funcionamiento de la red de evaluación LoRa, además se debe recordar que la dirección del Gateway debe estar en el rango de la dirección IP de la red.

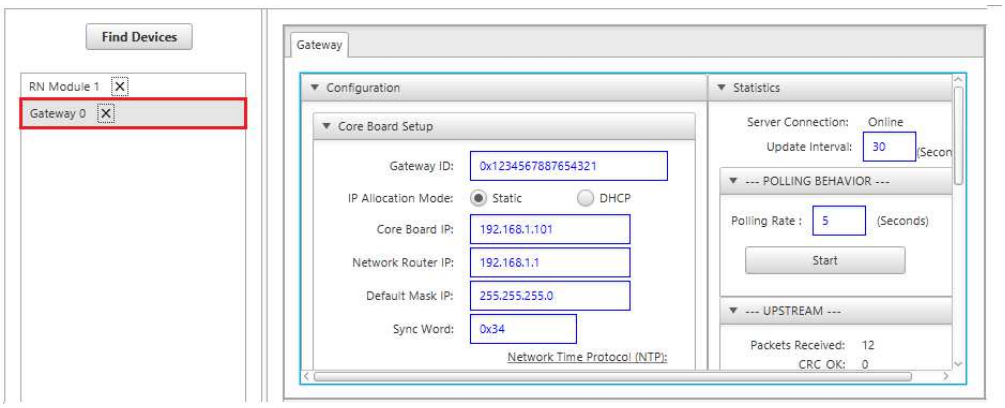


Figura 24: Panel de configuración de la tarjeta LoRa Gateway

Fuente: Microchip Technology, Inc.

El valor que no se debe cambiar es el 'Gateway ID' ya que este es el identificador del Gateway y cada uno tiene un valor único. Los valores de configuración que hay que recordar son los puertos de reenvió que se configuraron en el sistema invitado como en el sistema anfitrión de la VM (sección 4.2.1) y la dirección IP del servidor, como se puede ver en la Figura 25 se asignan los mismos valores que de la Tabla 5.

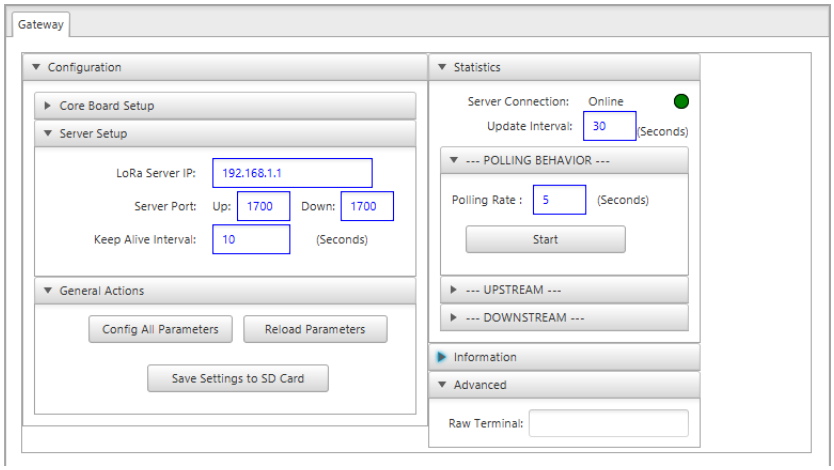


Figura 25: Configuración de puertos de reenvío y de la dirección del servidor

Fuente: Microchip Technology, Inc.

4.3.1 CONFIGURACIÓN AUTOMATICA ABP PARA LORA MOTE

La configuración de la tarjeta LoRa MOTE se realiza seleccionando el módulo que aparece en la ventana izquierda de la aplicación LoRa Development Utility (Figura 26), el menú 'LoRa WAN' permite realizar la configuración del modo de transmisión

ya sea OTAA (sección 1.3.2.1) o ABP (sección 1.3.2.2) y la configuración de las llaves NwkSKey, AppSKey y DevAdd.

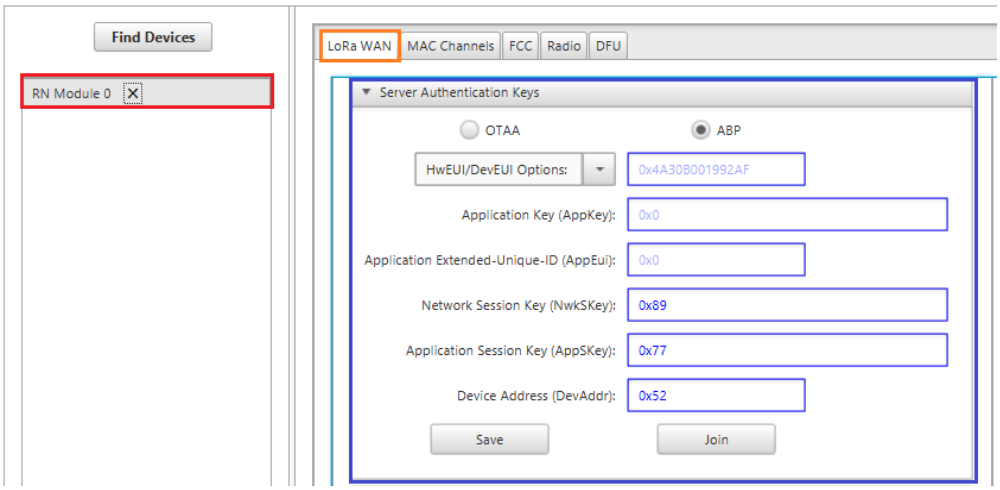


Figura 26: Panel de configuración de LoRa MOTE

Fuente: Microchip Technology, Inc.

Las llaves se pueden configurar de manera automática si el usuario realiza la configuración de auto creación, considerando que estas llaves deben ser únicas dentro de toda la red; en la configuración automática la aplicación genera las llaves como se puede ver en la Figura 27, se recomienda revisar el

Apéndice I: Autoconfiguración de tarjeta LoRa MOTE.

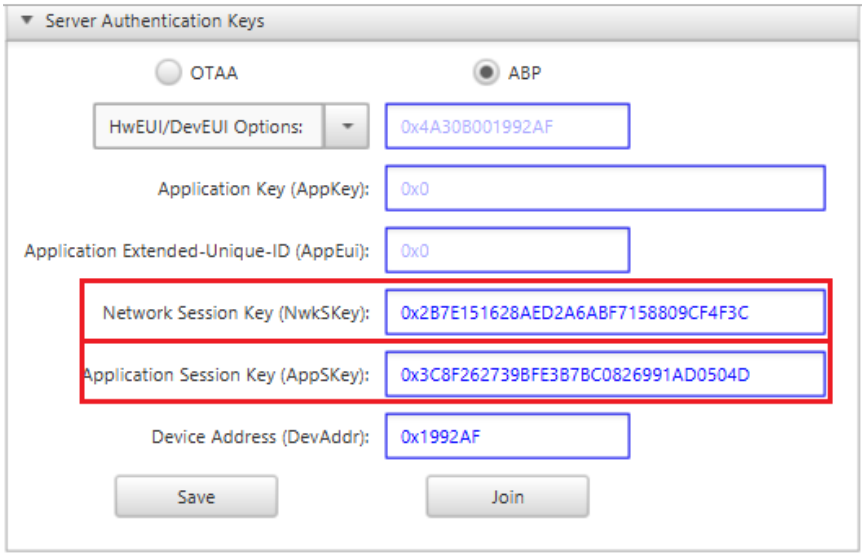


Figura 27: Configuración automática de las llaves de sesión y dirección

Fuente: Microchip Technology, Inc.

4.3.2 CONFIGURACIÓN ABP MANUAL PARA LORA MOTE

En la configuración manual se debe tomar en cuenta que llaves de sesión, aplicación y dirección del dispositivo deben ser ingresadas por él usuario, en el caso de que se repitan la red va a tener un fallo, en la Figura 28 se puede ver los campos de NwkSKey, AppSKey y DevAdd ingresados estos se guardan (color verde) y cargar (color morado), recordar que estos campos son la identidad de la tarjeta MOTE en la red.

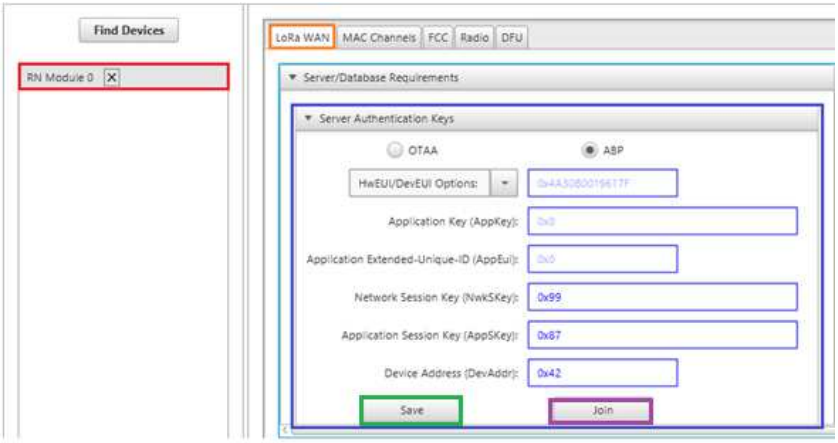


Figura 28: Configuración Manual de la clave de sesión y dirección de dispositivo

Fuente: Microchip Technology, Inc.

4.3.3 CONFIGURACIÓN OTAA PARA LORA MOTE

En la configuración OTAA se debe seleccionar en la pestaña 'Server Authentification Keys' la opción de OTAA y así poder ingresar los valores de AppKey, AppEUI y DevEUI, estos valores deben guardar y cargarlos (Figura 29), recordar que deben ser únicos ya que son la identidad del dispositivo dentro de la red, esta conexión es más segura por la forma de autentificación que tiene el dispositivo, se pude revisar el Apéndice J: Implementación de conexión OTAA, para mayor comprensión.

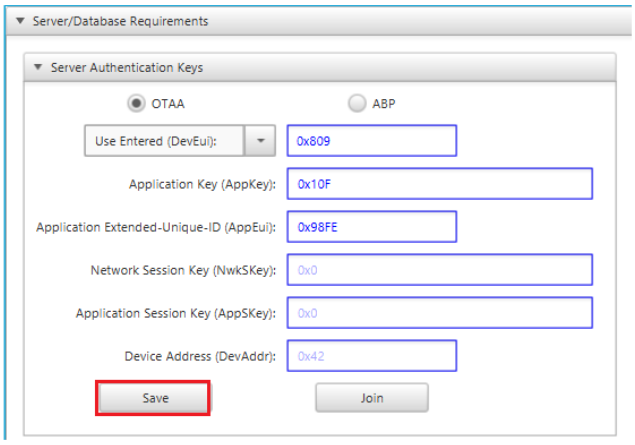


Figura 29: Configuración de conexión OTAA

Fuente: Microchip Technology, Inc.

4.3.4 CONFIGURACIÓN DE CANALES PARA LORA MOTE

Los canales se deben configurar para lograr la transmisión de datos de la tarjeta MOTE al Gateway, para ello se debe ingresar a la pestaña 'MAC Channels', en el cual se debe configurar el 'data rate', se aconseja colocar de 1 como mínimo y 5 como máximo, además el duty cycle se recomienda de 302, estos valores son dados por Microchip Technology, en la Figura 30 se puede ver la ventana de configuración de los canales.

Esta configuración se debe realizar para la conexión OTAA y ABP, para ello se debe deshabilitar todos los canales y luego habilitar los 8 canales principales para lograr la transmisión sin ningún problema.

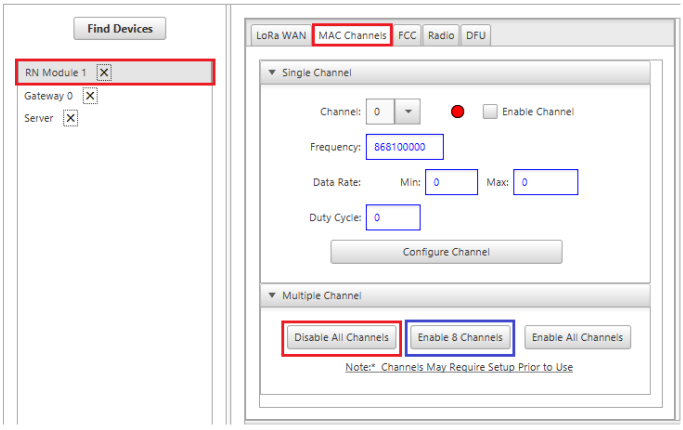


Figura 30: Configuración de canales en tarjetas MOTES

Fuente: Microchip Technology, Inc.

4.3.5 CONFIGURACIÓN DE SERVIDOR Y ADICIÓN DE DISPOSITIVOS MOTES

La aplicación LoRa Development Utility para tener comunicación con el servidor Docker debe tener una vía de comunicación para obtener y guardas los datos que se envían desde las tarjetas MOTEs, por lo tanto, se debe agregar a la lista de dispositivos el servidor y configurar con la dirección IP de la red, la configuración se enfoca en la conexión entre la aplicación y el servidor.

Para saber si la conexión fue exitosa la aplicación LoRa Development Utility muestra un mensaje de la versión del servidor (cuadro de color azul) como se puede ver en la Figura 31.

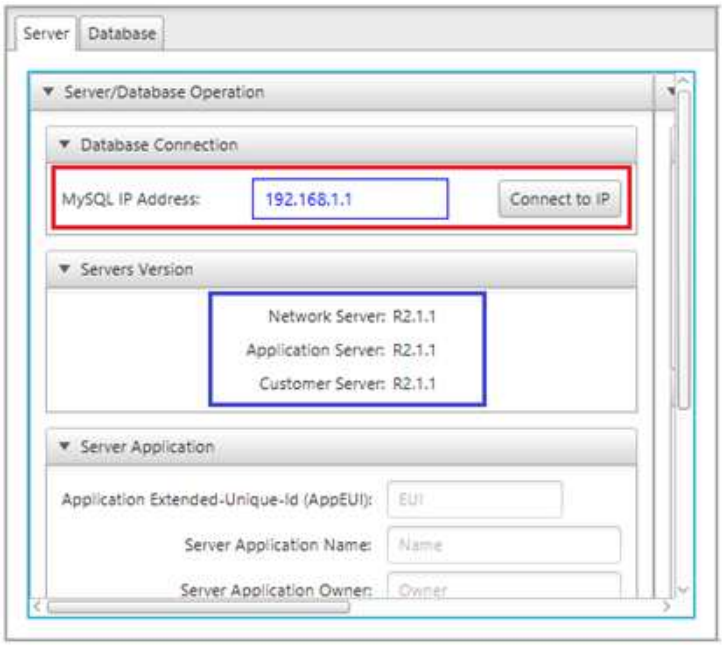


Figura 31: Configuración de comunicación entre LoRa Development Utility y Servidor Docker

Fuente: Microchip Technology, Inc.

Establecida la conexión con el servidor se debe cargar los datos de los MOTES que se establecieron en la sección 4.3.2 y 4.3.3 para diferenciar los dispositivos que se conectan y el tipo de conexión que realizan cada uno de ellos, en la Figura 32, se puede ver en el recuadro azul que se ingresa los datos de los dispositivos ABP y en el recuadro de morado se ingresan los dispositivos OTAA, en el caso de agregar más dispositivos se deben ingresar nuevos identificadores en el servidor con la finalidad que dispositivo estén dentro de la red LoRa.

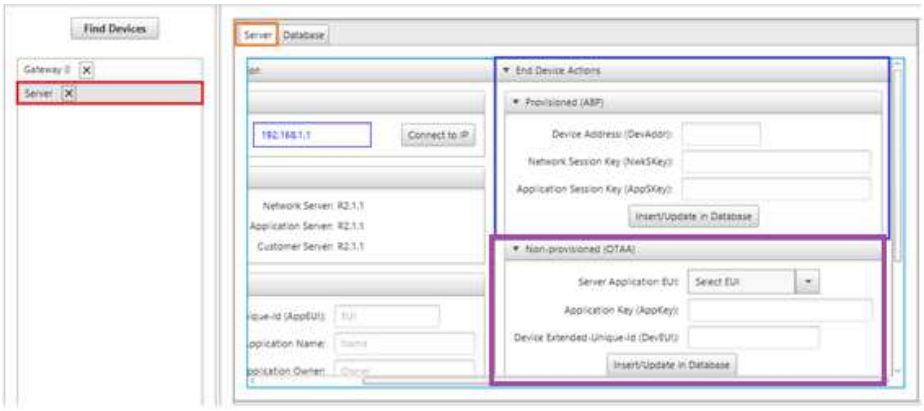


Figura 32: Ingreso de Identificadores de dispositivos MOTEs al servidor

Fuente: Microchip Technology, Inc.

4.3.6 ENVÍO Y CONFIRMACIÓN DE DATOS CON LORA DEVELOPMENT UTILITY

Realizada las configuraciones descritas anteriormente y para confirmar que los dispositivos MOTEs están ya conectados a la red LoRa se debe enviar datos para verificarlos dentro de la base de datos.

El envío de datos se realiza desde la ventana de los dispositivos MOTEs en la pestaña LoRaWAN, donde se configura el puerto, el tipo de datos que se va a enviar, los datos, la velocidad de datos y la potencia de transmisión como se observa en la Figura 33.

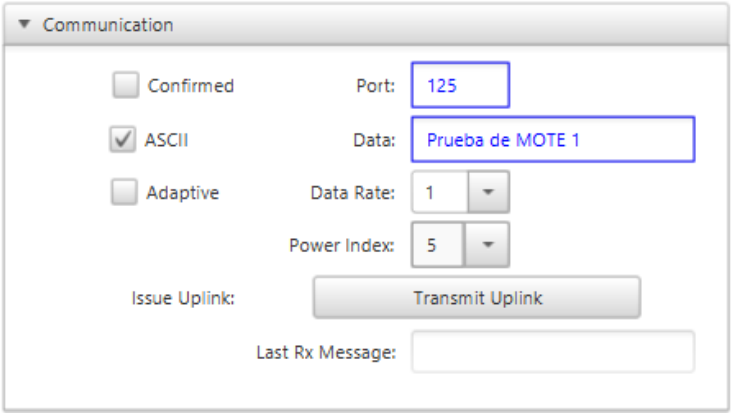


Figura 33: Configuración de envío de datos al servidor

Fuente: Microchip Technology, Inc.

La verificación de los datos enviados se puede obtener ingresando en la ventana del servidor, el cual dispone de la base de datos. Al ingresar podemos ver todos los datos enviados a la base de datos (Figura 34), se aconseja revisar el

Apéndice K: Confirmación de conexión oTAA.

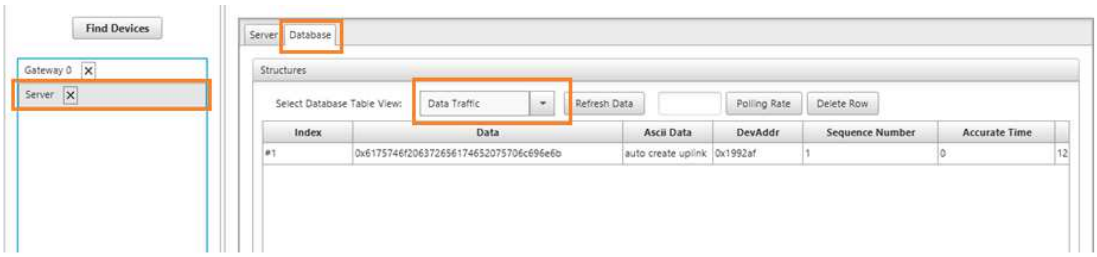


Figura 34: Confirmación de envió de datos y conexión exitosa de MOTEs a la red

Fuente: Microchip Technology, Inc.

4.3.7 USO DE PUERTOS GPIO DE LA TARJETA MOTE

La tarjeta MOTE permite la configuración de los puertos GPIO como entradas y salidas digitales, de manera que se pueda realizar una verificación del estado de los puertos GPIO, para ello LoRa Development Utility permite su configuración de manera manual como se puede ver en la Figura 35, revisar el

Apéndice L: Uso de Puertos GPIO de la Tarjeta MOTE.

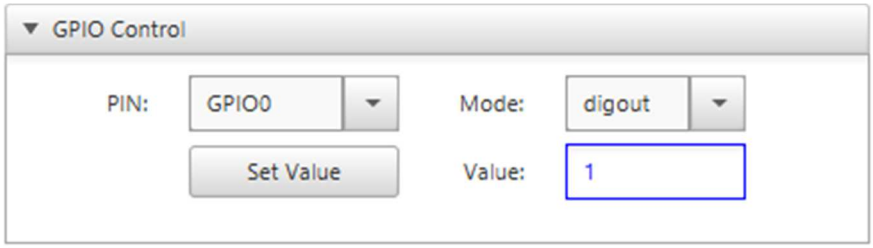


Figura 35: Configuración de los puertos GPIO de la tarjeta MOTE

Fuente: Microchip Technology, Inc.

4.4 CONFIGURACIÓN DE MPLAB X IDE

El software MPLAB permite la programación del PIC interno que posee el RN2483, lo primero que se debe realizar es la creación de un nuevo proyecto que se explica en el

Apéndice N: Creación de un Proyecto en MPLAB X IDE y la instalación del plugin de LoRaWAN y así agregar la librería que Microchip Technology posee para la programación de LoRaWAN, revisar el Apéndice Ñ.1: Instalación de Plugin Lorawan.

Una de las herramientas que Microchip Technology tiene para una mayor facilidad de programación es MCC, este es un entorno grafico que permite la configuración del microcontrolador, entre estas la activación de los puertos, selección de los periféricos que posee, además realiza la generación de los archivos con las configuraciones realizadas por el usuario, vea se el Apéndice Ñ.2: Instalación de MCC.

En la Figura 36 se puede ver como MCC genera una serie de ventanas de configuración que son System Module, Interrupt Module, Pin Module, Available Resources, además los paneles de Project Resources, Device Resources y Pin Manager.

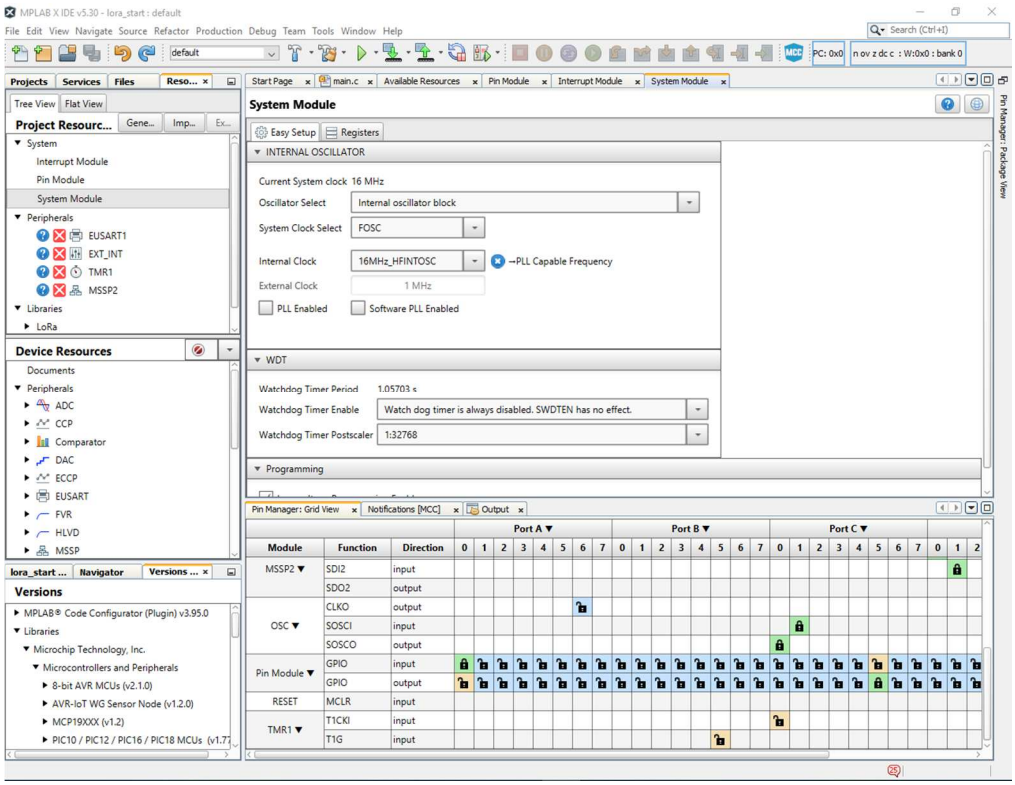


Figura 36: Ventana de configuración de MCC

Fuente: Microchip Technology, Inc.

4.4.1 MÓDULO DEL SISTEMA

Establece los ajustes de configuración básica de microcontrolador que son el tipo de oscilador, el reloj interno, la habilitación del perro guardián y la habilitación de sleep (Figura 37), esta interfaz permite al usuario configurar interactivamente y con mayor facilidad.

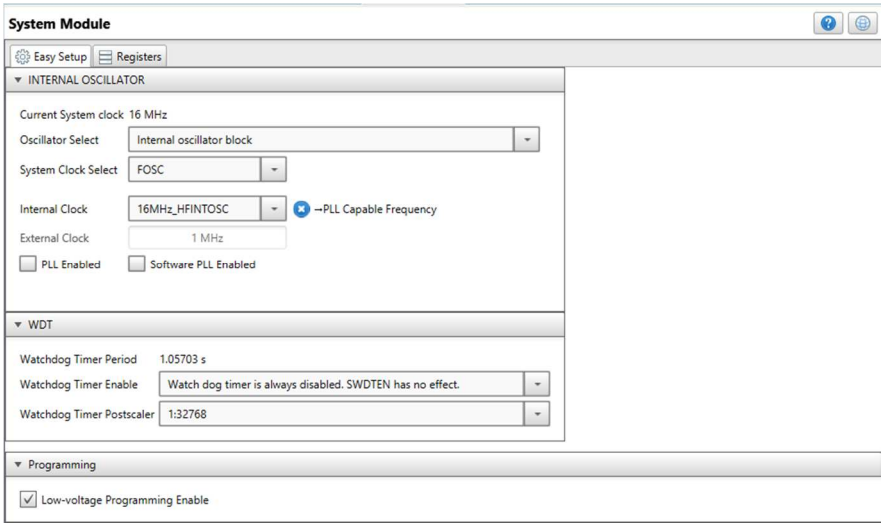


Figura 37: Interfaz gráfica para la configuración del sistema

Fuente: Microchip Technology, Inc.

4.4.2 MÓDULO DE INTERRUPCIONES

Este módulo presenta todas las interrupciones que se configuran para los diferentes periféricos del microcontrolador y las interrupciones globales, la configuración consiste en habilitar y deshabilitar las interrupciones, para ello se debe seleccionar la interrupción requerida, como se puede ver en la Figura 38.

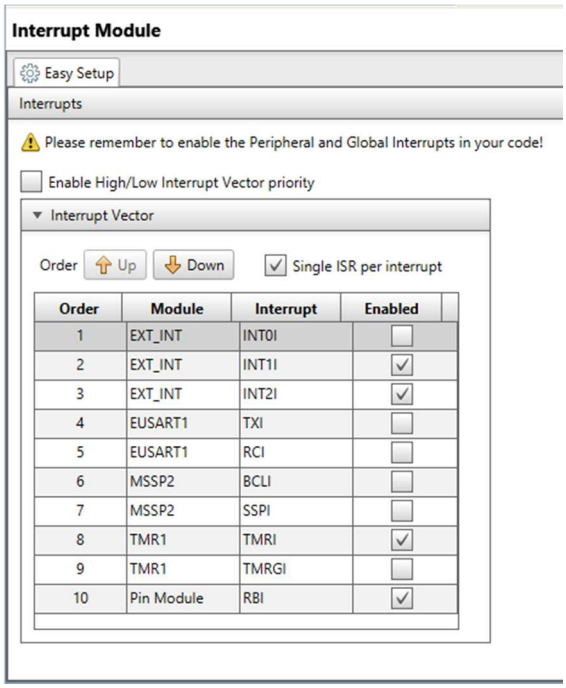


Figura 38: Habilitación o Deshabilitación de interrupciones en MCC

Fuente: Microchip Technology, Inc.

4.4.3 MÓDULO DE PINES

Es el encargado de la configuración de los pines como E/S, además está estrechamente unida a la función del pin manager ya que permite bloquear o desbloquear las E/S. Además, se puede cambiar el nombre del Pin en la casilla 'Custom Name'.

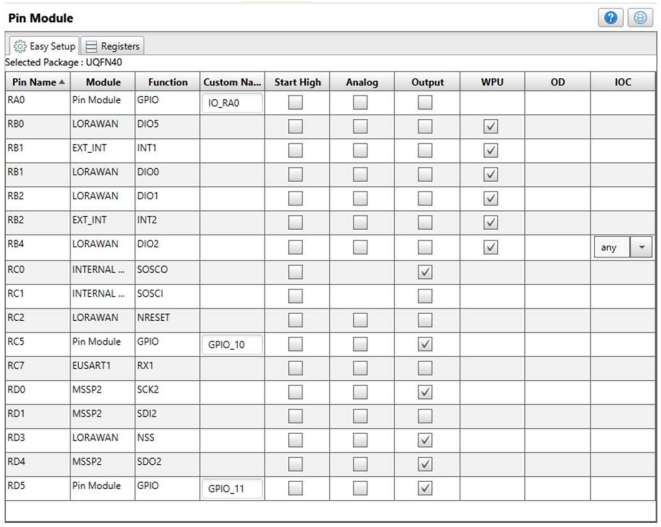


Figura 39: Configuración de Pines como entradas y salidas digitales

Fuente: Microchip Technology, Inc

El pin manager presenta que pines se están habilitando, deshabilitando o si pueden ser ocupados y se muestra en una interfaz como la Figura 40.

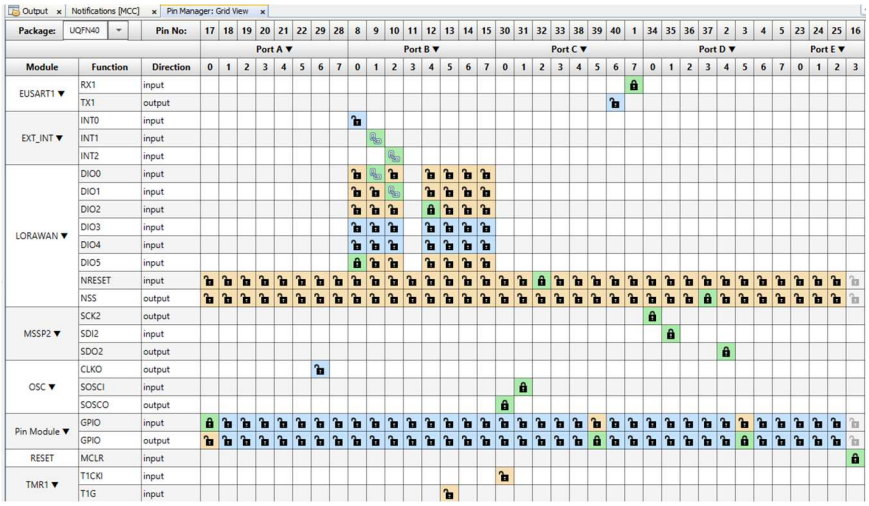


Figura 40: Representación del estado de los pines del microcontrolador

Fuente: Microchip Technology, Inc

4.4.4 GENERACIÓN DE CÓDIGO MEDIANTE MCC

La generación de código se realiza una vez ya terminada las configuraciones de las secciones 4.4.1, 4.4.2 y 4.4.3, se puede ir al

Apéndice O: Desarrollo de ejemplos con MCC para ver un ejemplo de la configuración en general. Para que el usuario genere el código se debe dirigir a la ventana 'Project Resources' y dar clic en 'Generate' cómo se puede ver en la Figura 41.

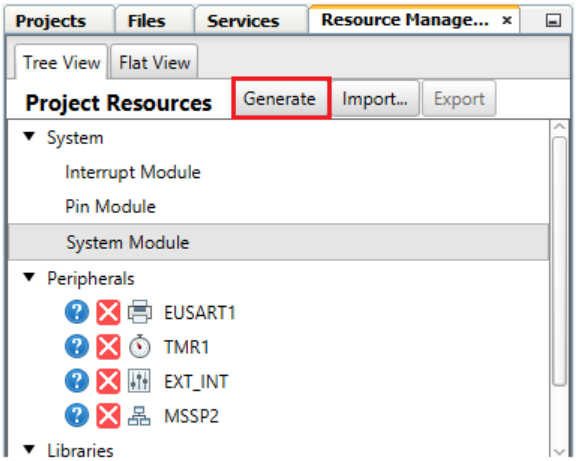


Figura 41: Ventana de Project Resources

Fuente: Microchip Technology, Inc

Los archivos que se generan se pueden ver en la pestaña de 'Projects' (Figura 42), los archivos de extensión '.h' son cabeceras generadas de los periféricos para el PIC18 y los archivos '.c' son las funciones principales donde se configura los registros de los periféricos del PIC18.

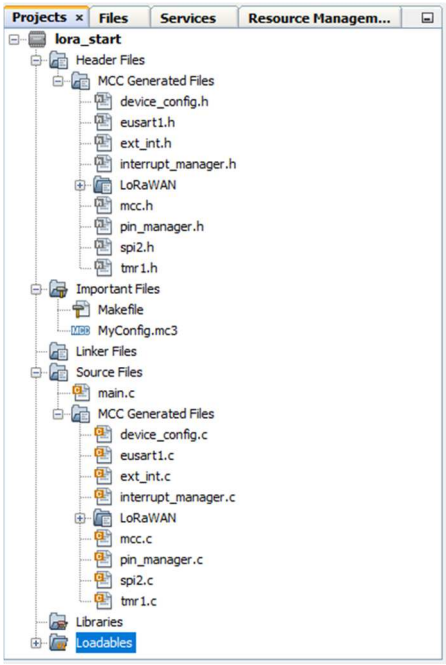


Figura 42: Archivos generados por MCC con extensión '.c' y '.h'

Fuente: Autor

4.4.5 CONFIGURACIÓN DE PERIFÉRICOS CON MCC

Los periféricos que MCC dispone al usuario dependen del microcontrolador utilizado, en este caso se utilizara el PIC18LF46k22, debido a que este PIC es el que se incorpora en el RN2483, en la Figura 43 se puede ver los periféricos disponibles. Hay que tener en cuenta que no todos los periféricos se pueden utilizar cuando se programa el microcontrolador con la librería LoRaWAN, ya que ocupa demasiados recursos, además algunos de los pines del PIC18LF46k22 ya están conectados internamente con el SX1276 y no se pueden utilizar.

Start Page | main.c | Available Resources | Pin Module | Interrupt Module | System Module

Module	Type	Library (Available Libraries)
ADC	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
CCP4	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
CCP5	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
CMP1	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
CMP2	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
DAC (5 bit)	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
ECCP1	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
ECCP2	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
ECCP3	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
EUSART2	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
FVR	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
HLVD	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
MEMORY	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
MSSP1	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
SRLATCH	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
TMR0	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
TMR2	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
TMR3	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
TMR4	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
TMR5	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs
TMR6	Peripheral	PIC10 / PIC12 / PIC16 / PIC18 MCUs

Figura 43: Periféricos disponibles en el PIC18LF46k22

Fuente: Microchip Technology, Inc

La selección de los periféricos se puede realizar agregando desde la ventana 'Device Resources' cómo se puede ver en la Figura 44, hay que tener en cuenta la hoja de datos del PIC para poder realizar la configuración del periférico que se agregue.

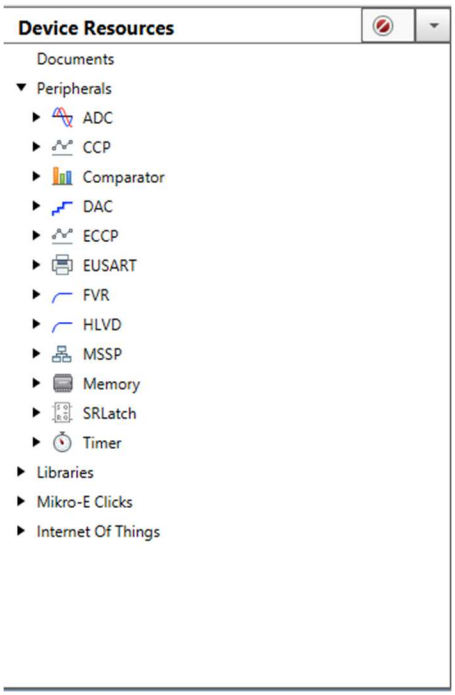


Figura 44: Ventana de selección de periféricos

Fuente: Microchip Technology, Inc

Para agregar el periférico se debe seleccionar en este caso el ADC, luego agregarlo realizando clic en el cuadro verde tal como se muestra en la Figura 45.

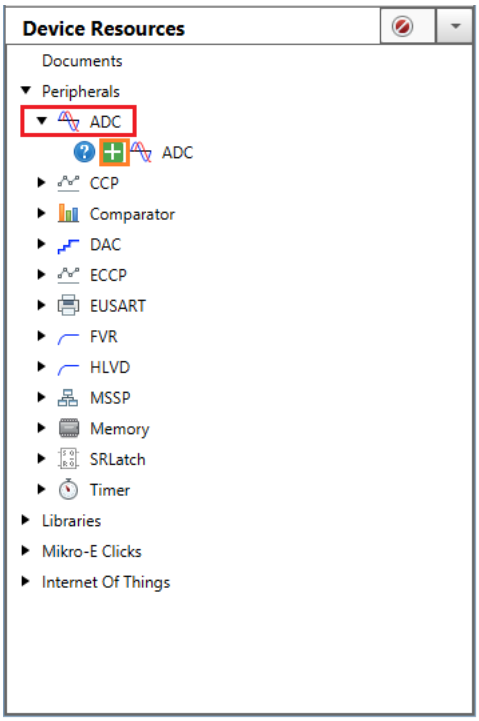


Figura 45: Ventana de Device Resources

Fuente: Autor

Agregado el periférico saldrá una ventana (Figura 46) en la cual se deberá configurar el ADC, se aconseja que se revise la hoja de datos del PIC, además MCC nos ayuda con notificaciones y alertas que presenta o requiere el módulo antes de generar el código. Cualquier periférico que se agrega se deberá realizar el mismo procedimiento que se indicó.

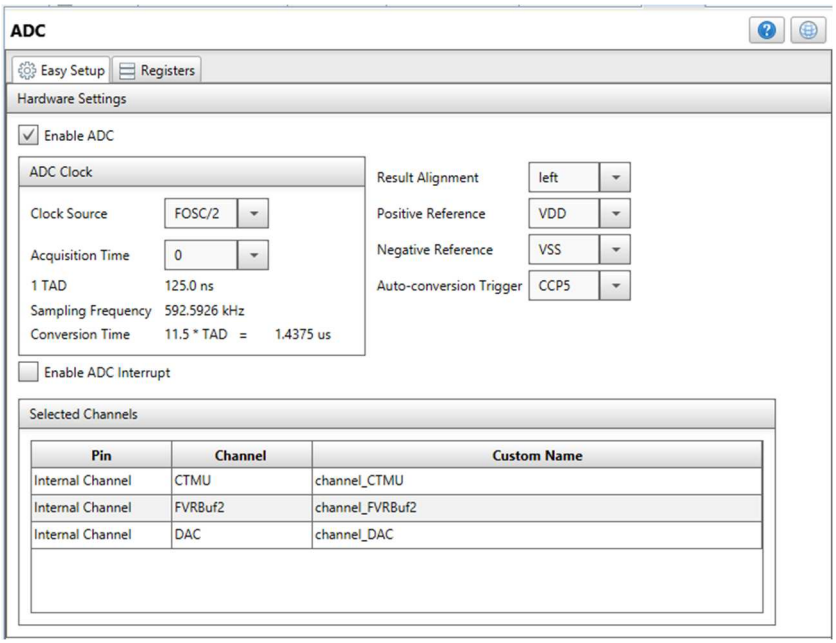


Figura 46: Ventana de configuración del ADC

Fuente: Microchip Technology, Inc

4.4.6 CONFIGURACIÓN DE LORAWAN CON CONFIGURACIÓN DE CODIGO DE MPLAB (MCC)

Las configuraciones que se deben realizar dependen del módulo que se esté utilizando, en este caso es el RN2483, por lo tanto, la frecuencia en la que trabaja está en los 400 MHz y 868 MHZ, incorpora el módulo de radio SX1276 que tiene la configuración por defecto para LoRaWAN , también se debe configurar el Timer 1 para las interrupciones que necesita el sistema, los módulos SPI para la comunicación con el SX1276, la configuración del pin Manager para bloquear los pines para la comunicación con el módulo de radio y la implementación de una encriptación que Microchip Technology recomienda; se debe recordar que la instalación del plugin 'Lorawan' permite una configuración mucho más fácil del RN2483 y hay que tener presente que MCC facilita la programación ya que genera un código según las configuraciones que se realizan.

La configuración que se realiza entre el PIC18LF46K22 y el módulo SX1276 es basada en la comunicación SPI, para ello es necesario seis puertos GPIO, un pin de control de restablecimiento (NRESET) y un pin de comunicación con el transceptor, esto se puede ver en Figura 47, también se debe configurar el sistema de reloj interno en 16 MHz para el funcionamiento correcto del dispositivo.

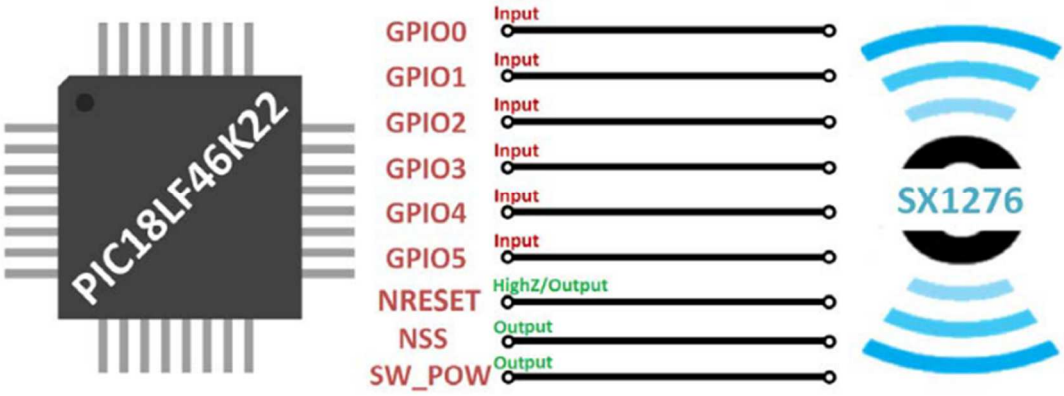


Figura 47: Comunicación de PIC18LF46K22 y SX1276, Módulo RN2483

Fuente: Microchip Technology, Inc

Los pines que se deben configurar se presentan en la Tabla 6, además una vez que se configuran estos pines algunas funciones del PIC ya no se podrán realizar como en el caso de las detecciones por flancos, debido a la utilización para la implementación de LoRaWAN.

Tabla 6: Tabla de Pines para la comunicación SPI

Nombre	Puerto
GPIO 0	RB1
GPIO 1	RB2
GPIO 2	RB4
GPIO 3	RESERVADO
GPIO 4	RESERVADO
GPIO 5	RB0
NRESET	RC2
NSS	RD3

La encriptación que se realiza se basa en las librerías de encriptación para los MCU de 8-bits y 16-bits que Microchip Technology dispone (Figura 48, se pueden

descargas directamente de la página oficial), este proceso se realiza después de generar el código de programación en MCC. Los datos que se envían por un radio enlace desde la tarjeta MOTE al Gateway deben ser encriptados para que no puedan identificar los datos transmitidos otros dispositivos o personas de malas intenciones.

```
...ew  AES.c  AES.h  Pin Module  System Module  Interrupt Module  LORAWAN  EXT_INT  MSSP2  TMR1
Source  History
1   /*...55 lines */
56
57  /****************************** Headers ******************************/
58  #include <p18CXXX.h>
59  #include "AESdef.h"
60  #include "AES.h"
61
62  /****************************** Constants ****************************/
63  rom const unsigned char STable[] =
64  {
65      0x63,0x7C,0x77,0x7B,0xF2,0x6B,0x6F,0xC5,0x30,0x01,0x67,0x2B,0xFE,0xD7,0xAB,0x76,
66      0xCA,0x82,0xC9,0x7D,0xFA,0x59,0x47,0xF0,0xAD,0xD4,0xA2,0xAF,0x9C,0xA4,0x72,0xC0,
67      0xB7,0xFD,0x93,0x26,0x36,0x3F,0xF7,0xCC,0x34,0xA5,0xE5,0xF1,0x71,0xD8,0x31,0x15,
68      0x04,0xC7,0x23,0xC3,0x18,0x96,0x05,0x9A,0x07,0x12,0x80,0xE2,0xEB,0x27,0xB2,0x75,
69      0x09,0x83,0x2C,0x1A,0x1B,0x6E,0x5A,0xA0,0x52,0x3B,0xD6,0xB3,0x29,0xE3,0x2F,0x84,
70      0x53,0xD1,0x00,0xED,0x20,0xFC,0xB1,0x5B,0x6A,0xCB,0xBE,0x39,0x4A,0x4C,0x58,0xCF,
71      0xD0,0xEF,0xAA,0xFB,0x43,0x4D,0x33,0x85,0x45,0xF9,0x02,0x7F,0x50,0x3C,0x9F,0xA8,
72      0x51,0xA3,0x40,0x8F,0x92,0x9D,0x38,0xF5,0xBC,0xB6,0xDA,0x21,0x10,0xFF,0xF3,0xD2,
73      0xCD,0x0C,0x13,0xEC,0x5F,0x97,0x44,0x17,0xC4,0xA7,0x7E,0x3D,0x64,0x5D,0x19,0x73,
74      0x60,0x81,0x4F,0xDC,0x22,0x2A,0x90,0x88,0x46,0xEE,0xB8,0x14,0xDE,0x5E,0x0B,0xDB,
75      0xE0,0x32,0x3A,0x0A,0x49,0x06,0x24,0x5C,0xC2,0xD3,0xAC,0x62,0x91,0x95,0xE4,0x79,
76      0xE7,0xC8,0x37,0x6D,0x8D,0xD5,0x4E,0xA9,0x6C,0x56,0xF4,0xEA,0x65,0x7A,0xAE,0x08,
77      0xBA,0x78,0x25,0x2E,0x1C,0xA6,0xB4,0xC6,0xE8,0xDD,0x74,0x1F,0x4B,0xBD,0x8B,0x8A,
78      0x70,0x3E,0xB5,0x66,0x48,0x03,0xF6,0x0E,0x61,0x35,0x57,0xB9,0x86,0xC1,0x1D,0x9E,
EncodeShiftRow
```

Figura 48: Código de Encriptación AES128 para datos LoRaWAN

Fuente: Microchip Technology, Inc

Realizadas las configuraciones respectivas y la generación del código en MPLAB se puede visualizar todos los documentos creados, en la ventana 'Projects' los archivos con extensión '.h' presentan una descripción de las funciones creadas por MCC y los archivos con extensión '.c' son los que llaman a esas funciones para los diferentes procesos que se deben ejecutar. Para realizar él envió de un mensaje se debe ingresar la nwkSkey, appKey, devAddr (Figura 49), habilitar las interrupciones globales (Figura 50) y llamar a las funciones correspondientes para el inicio de la transmisión LoRaWAN (Figura 51).

```
#include "mcc_generated_files/mcc.h"

uint8_t nwkSKey[16] = {0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB,
0xF7, 0x15, 0x88, 0x09, 0xCF, 0x4F, 0x3C};
uint8_t appSKey[16] = {0x3C, 0x8F, 0x26, 0x27, 0x39, 0xBF, 0xE3, 0xB7, 0xBC,
0x08, 0x26, 0x99, 0x1A, 0xD0, 0x50, 0x4D};
uint32_t devAddr = 0x1100000F;

void RxData(uint8_t* pData, uint8_t dataLength, OpStatus_t status)
{}

void RxJoinResponse(bool status)
{}
```

Figura 49: Ingreso de llaves y funciones para transmisión LoRaWAN

Fuente: Microchip Technology, Inc

```
// Disable low priority global interrupts.
//INTERRUPT_GlobalInterruptLowDisable();

// Enable the Global Interrupts
INTERRUPT_GlobalInterruptEnable();

// Enable the Peripheral Interrupts
INTERRUPT_PeripheralInterruptEnable();

// Disable the Global Interrupts
//INTERRUPT_GlobalInterruptDisable();

// Disable the Peripheral Interrupts
//INTERRUPT_PeripheralInterruptDisable();
```

Figura 50: Habilitación de las Interrupciones Globales

Fuente: Microchip Technology, Inc

```
LORAWAN_Init(RxData, RxJoinResponse);

LORAWAN_SetNetworkSessionKey(nwkSKey);
LORAWAN_SetApplicationSessionKey(appSKey);
LORAWAN_SetDeviceAddress(devAddr);

LORAWAN_Join(ABP);
```

Figura 51: Funciones de inicialización del sistema LoRaWAN

Fuente: Microchip Technology, Inc

Para enviar caracteres se debe ingresar el código dentro del bucle 'while' la función LORAWAN_Send(), se debe el tipo de mensaje que es 'UNCNF' quiere decir, un mensaje no confirmado, ingresar el puerto para la transmisión, este puede ser de 0 a 255, los datos a enviar y la longitud del buffer de datos, como se puede ver en la

Figura 52, se recomienda ver el Apéndice P: Configuración LoRaWAN en PIC18LF46k22.

```
while (1)
{
    // Add your application code
    LORAWAN_Mainloop();

    // All other function calls of the user-defined
    // application must be made here
    LORAWAN_Send(UNCNF, 2, "LoRa", 4);
}
```

Figura 52: Función de transmisión de datos para LoRaWAN

Fuente: Microchip Technology, Inc

CAPÍTULO 5: CONCLUSIONES Y RECOMENDACIONES

En este proyecto se desarrolló una serie de prácticas de laboratorio enfocadas a redes de sensores empleando el módulo LoRa® Technology Evaluation Kit 800, el cual puede ser configurado mediante la aplicación LoRa Development Utility o MPLAB X IDE, el módulo de guías de prácticas esta desarrollado y estructurado en base a la utilización de las aplicaciones mencionadas.

La aplicación LoRa Development Utility es de gran ayuda para comprender la configuración y utilización del MOTE, Gateway y la conexión con el Servidor, además al realizar las pruebas de funcionamiento especialmente en los MOTEs, la aplicación solo permite transmitir caracteres ASCII, logra cambiar el estado de los puertos del RN2483 y permite la configuración de las llaves de sesión y aplicación, pero el problema que se presenta es la restricción de la configuración de los periféricos del microcontrolador 18LF46K22, por lo tanto, este programa sirve para la orientación y comprensión de la tecnología LoRaWAN.

En MPLAB X IDE la configuración del RN2483 se realiza mediante la herramienta MCC que permite configurar por medio de una interfaz gráfica al microcontrolador 18LF46k22 y generar el código en lenguaje 'C'. Al realizar las pruebas de funcionamiento se percató que el RN2483 pierde la comunicación con la aplicación LoRa Development Utility y con el microcontrolador 18LF45K50 al cargar el nuevo Firmeware, esto se debe a que se cambia la el Firmeware de fabrica que permite la comunicación con LoRa Development Utility, esta dificultad se superó con

el desarrollo de la aplicación en Node JS y Angular, que se denominó S.A.D, está desencripta y separa los datos importantes capturados en el puerto 1700 para su visualización.

La configuración de la tarjeta MOTE en MPLAB X IDE es más detallada, ya que se debe tener en cuenta la arquitectura del RN2483, la hoja de datos del microcontrolador y los periféricos que se pueden utilizar, por lo tanto, este programa es bueno para comprender de manera detallada la configuración de un dispositivo LoRaWAN.

Las bandas de frecuencias libres que el país dispone para la utilización de la tecnología LoRaWAN se las puede ver en el Anexo 2: Comunicado de ARCOTEL - Bandas de Frecuencias, las bandas en las que LoRaWAN funcionan son 400 MHz, 868MHz y 900MHz. El módulo LoRa Technology Evaluation Kit 800 funciona en bandas denominadas por LoRa Alliance y The Things Network como EU863-870 y EU433 que se utilizan en los países europeos como Austria, Bélgica, Croacia, Dinamarca, Estonia, Finlandia, Francia, entre otros, además esas frecuencias ya están utilizadas por las Telecomunicaciones móviles Internaciones en el país, por lo tanto, el módulo se puede utilizar para pruebas de laboratorio e investigación dentro de la institución, de manera personalmente realizara un cambio en el módulo MOTE, donde incorpore el chip RN2903, para poder utilizar en la industria a nivel nacional ya que este chip si funciona en las bandas permitidas en el país.

REFERENCIAS BIBLIOGRÁFICAS

[1] LoRa Alliance, "LoRaWAN," 2018. [Online]. Available: https://lora-alliance.org/about-lorawan.

[2] J. P. Shanmuga Sundaram, W. Du, and Z. Zhao, "A Survey on LoRa Networking: Research Problems, Current Solutions, and Open Issues," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 1, pp. 371–388, 2020.

[3] A. Zourmand, "Internet of Things (IoT) using LoRa technology," *2019 IEEE Int. Conf. Autom. Control Intell. Syst.*, no. June, pp. 324–330, 2019.

[4] K. Olsson and S. Finnsson, "Exploring LoRa and LoRaWAN A suitable protocol for IoT weather stations ?," 2017.

[5] I. Akyildiz, *Wireless Sensor Network*, 2010th ed. 1395.

[6] M. Santiago, R. Fernando, D. G. Armando, and T. F. G, "Análisis para Despliegue de una Red de Sensores Heterogénea," pp. 787–796, 2018.

[7] P. Avila, "UNIVERSIDAD DE CUENCA - TESIS.pdf," *Artic. Ecuador*, vol. 1, no. 5, pp. 1–127, 2017.

[8] J. S. Rueda and J. M. Talavera Portocarrero, "Similitudes y diferencias entre Redes de Sensores Inalámbricas e Internet de las Cosas: Hacia una postura clarificadora," *Rev. Colomb. Comput.*, vol. 18, no. 2, pp. 58–74, 2017.

[9] ITU-T, "Requisitos para el soporte de los servicios y aplicaciones de redes de sensores ubicuos en el entorno de las redes de próxima generación." 2010.

[10] J. Merino, "Despliegue Y Evaluación de una Red On- Site LoRaWAN basada en The Things Network Stack versión 3," 2019.

[11] E. Sáenz, "Estudio de los parámetros principales de la tecnología LoRa para el despliegue de redes e implementación de servicios IoT en la CAV-EAE," pp. 2017–2018, 2018.

[12] T. M. Workgroup, "A technical overview of LoRa ® and LoRaWAN ™ What is it?," no. November, 2015.

[13] N. S. Sornin, M. S. Luis, T. I. Eirich, T. I. Kramp, and O. A. Hersent, "LoRaWAN™ 1.0.3 Specification," 2018.

[14] Microchip, *LoRa Technology Evaluation Suite User's Guide*. 2016.

[15] LoRa Alliance, "LoRaWAN Backend Interfaces 1.0.4 Specification," *LoRa Alliance*, no. 2010, p. 97331, 2017.

[16] Microchip Technology, “Connecting a SAM R34 LoRaWAN™ End-Device to a LoRaWAN Network Server.” [Online]. Available: https://microchipdeveloper.com/lora:connecting-to-a-lorawan-network.

[17] Microchip, *LoRa Technology Gateway User’s Guide*. 2016.

[18] S. Sector and O. F. Itu, “ITU-T Rec. K.79 (03/2015) Electromagnetic characterization of the radiated environment in the 2.4 GHz ISM band,” vol. 79, 2015.

[19] T.KRAMP and A.YEGIN, “RP002-1.0.0 LoRaWAN® Regional Parameters,” 2019.

[20] MTC, “Plan Nacional de Atribución de Frecuencias,” p. 46, 2016.

[21] Agencia de regulacion y Control de las telecomunicaciones, “Informe de Canalizacion de las bandas de 900 MHz y AWS.” 2017.

[22] ARCOTEL, “RESOLUCIÓN ARCOTEL-2018-,” vol. 2. QUITO, pp. 3–4, 2018.

[23] H. Miranda, “El Plan Nacional de Frecuencias vigente.” QUITO, p. 1, 2020.

[24] K. Jason, *Web Development with MongoDB and Node JS*, no. February 2019. Packt Publishing Ltd., 2014.

APÉNDICES

APÉNDICE A: INSTALACIÓN DE VISUAL STUDIO

Desde la página principal se puede descargar la última versión estable, al descargar el software ejecutamos el instalador

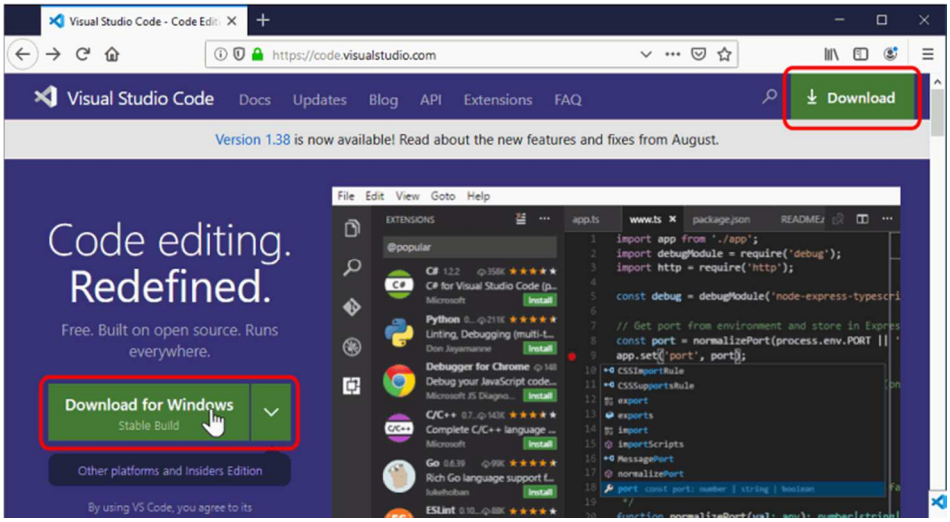


Figura A 1: Instalador de Visual Studio Code

La primera pantalla exige aceptar la licencia de Visual Studio Code, damos clic en siguiente.

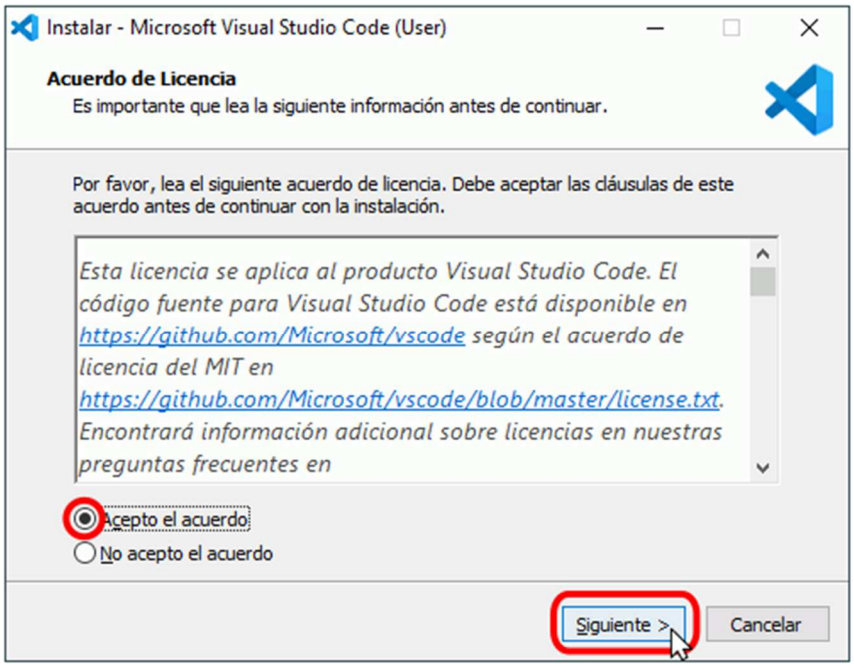


Figura A 2: Acuerdo de licencia de Visual Studio Code

La segunda pantalla permite elegir el directorio de instalación, en ese caso dejamos en el predeterminado.

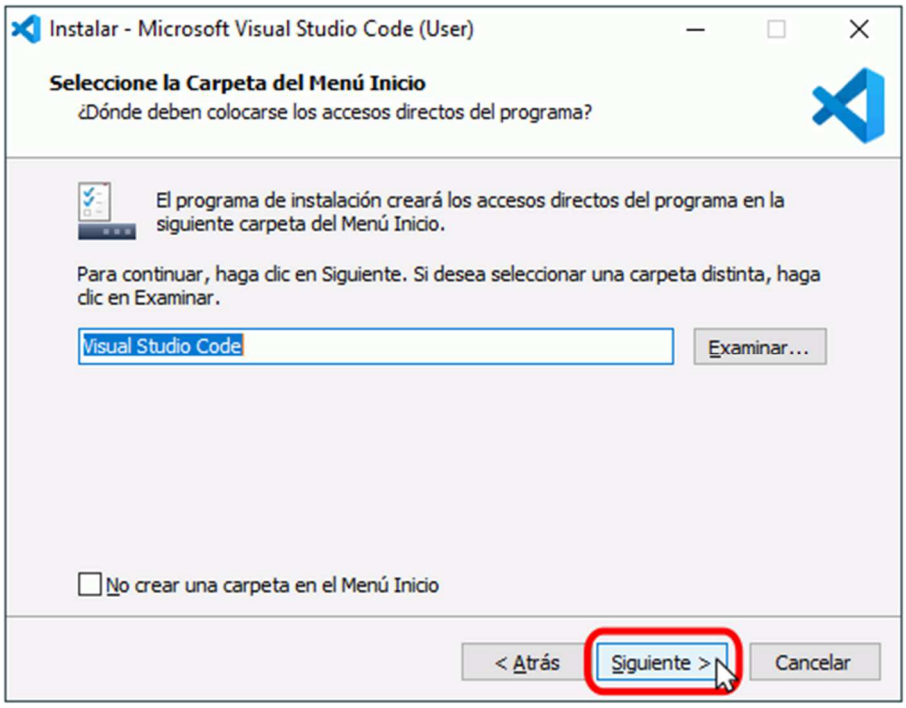


Figura A 3: Creación del acceso directo de Visual Studio Code

La cuarta pantalla permite elegir algunas tareas adicionales agregamos las que están marcadas.

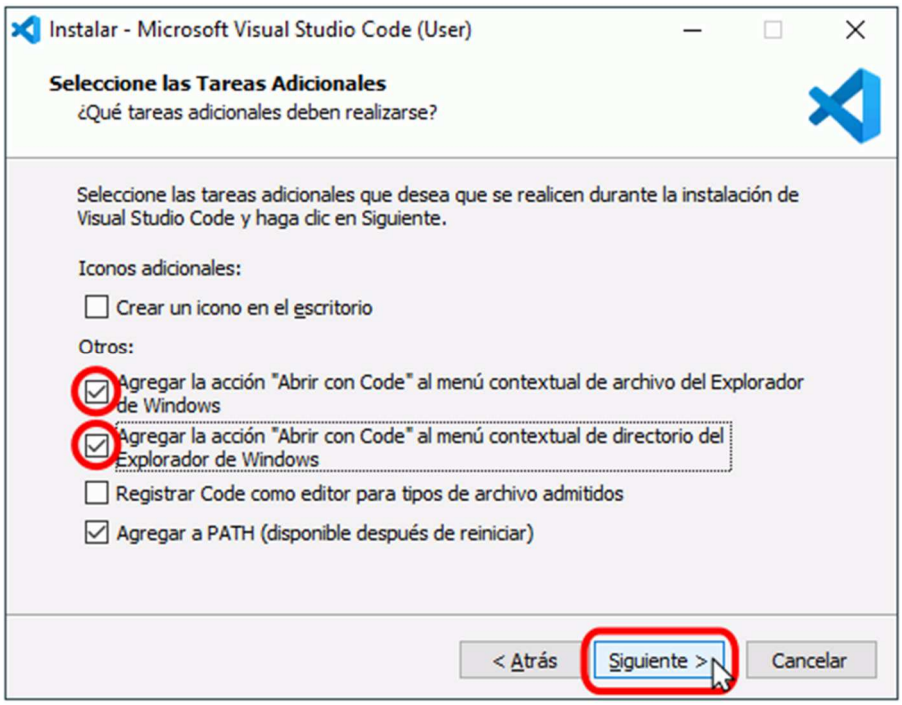


Figura A 4: Selección de Tareas Adicionales

Finalmente se muestran las opciones elegidas en las pantallas anteriores e iniciamos la instalación, haga clic en Instalar.

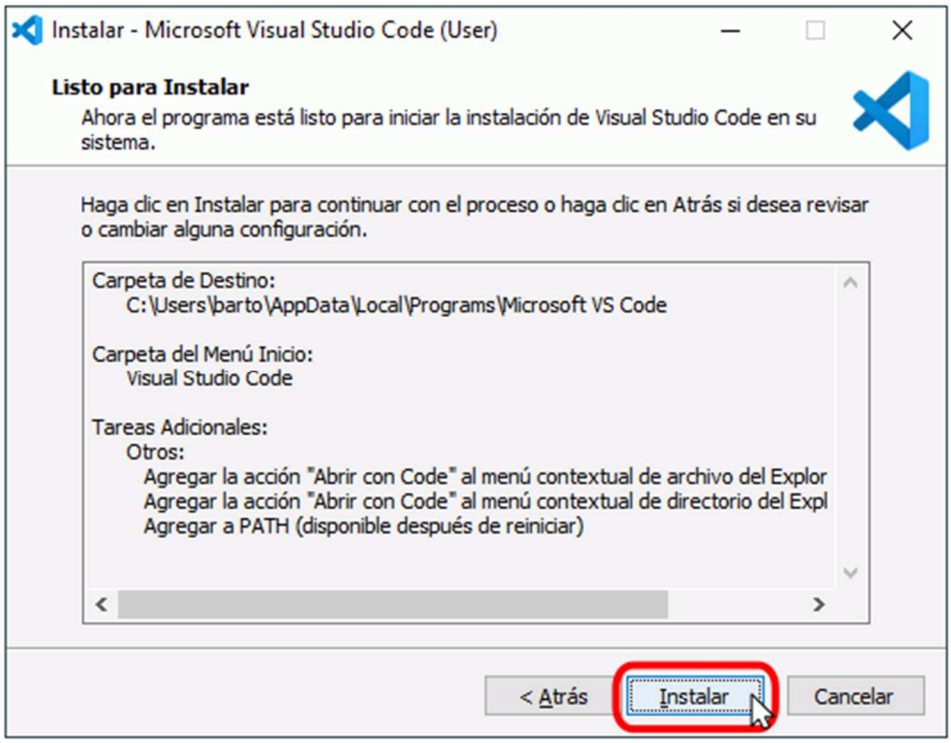


Figura A 5: Ventana de confirmación de instalación de Visual Studio Code

APÉNDICE B: INSTALACIÓN DE NODE JS

Se debe descargar de la página oficial el instalador de node js

Figura B 1: Página oficial de descarga de Node Js

Para la instalación damos doble clic en instalador de Node Js y damos clic en next

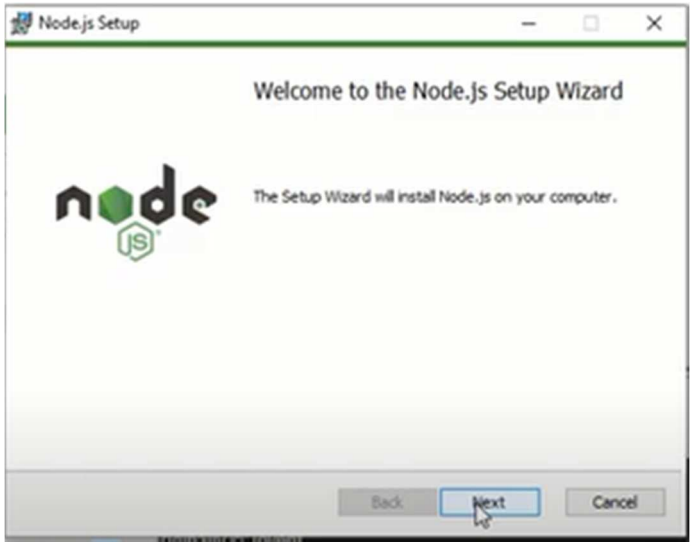


Figura B 2: Ventana de Instalación de Node Js

Aceptamos los terminas de licencia y damos clic en next

Figura B 3: Aceptación de Acuerdos de licencia de Node Js

Dejamos el directorio por defecto de la instalación

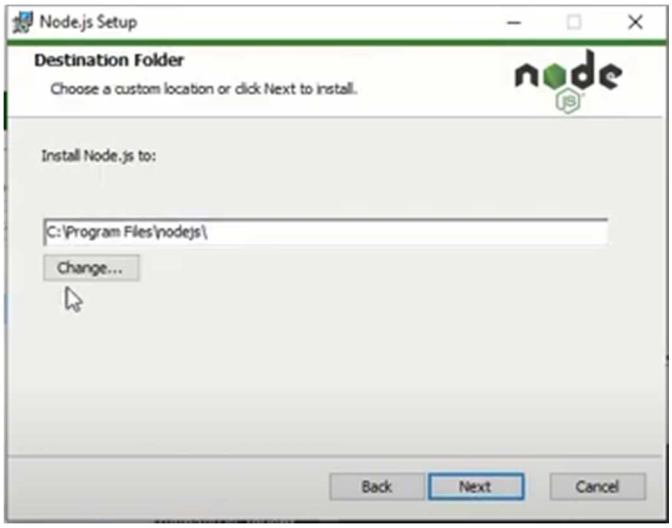


Figura B 4: Directorio de Instalación de Node Js

Seleccionamos los paquetes para la instalación, damos clic en Next y esperamos que se instale

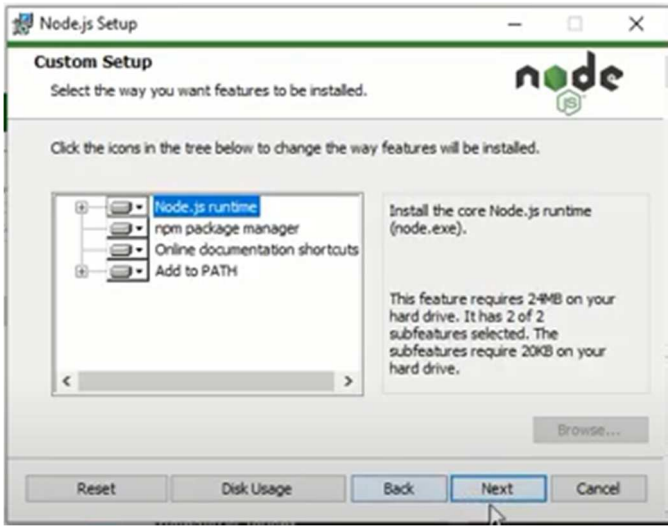


Figura B 5: Configuración de tipo de instalación

Para finalizar damos clic en finish

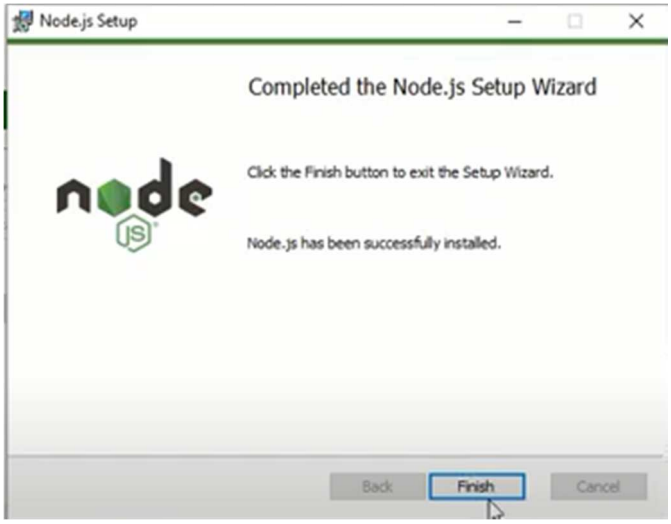


Figura B 6: Finalización de instalación

APÉNDICE C: INSTALACIÓN BASE DE DATOS

C.1: INSTALACIÓN DE MONGO DB

Se ejecuta le instalador y se da clic en next

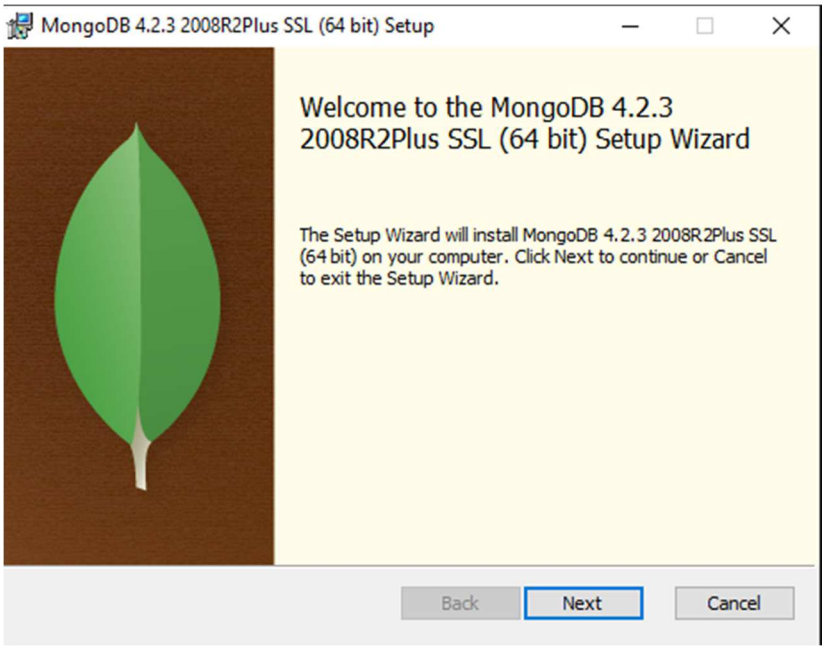


Figura C 1: Ventana de inicio de instalación de MongoDB

Aceptamos los términos de la licencia y clic en next

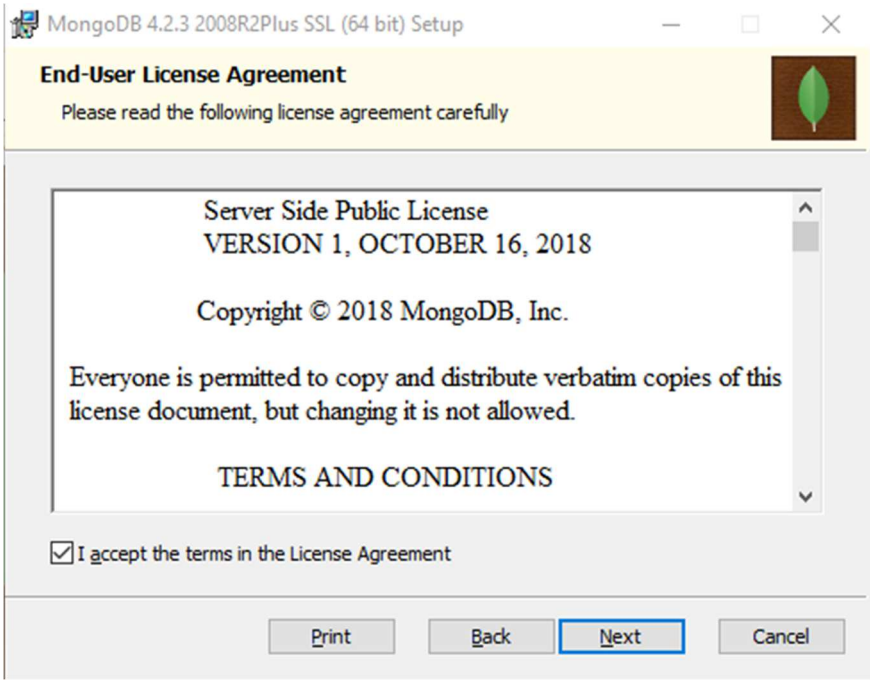


Figura C 2: Acuerdo de aceptación de licencia de MongoDB

Elegimos la opción custom

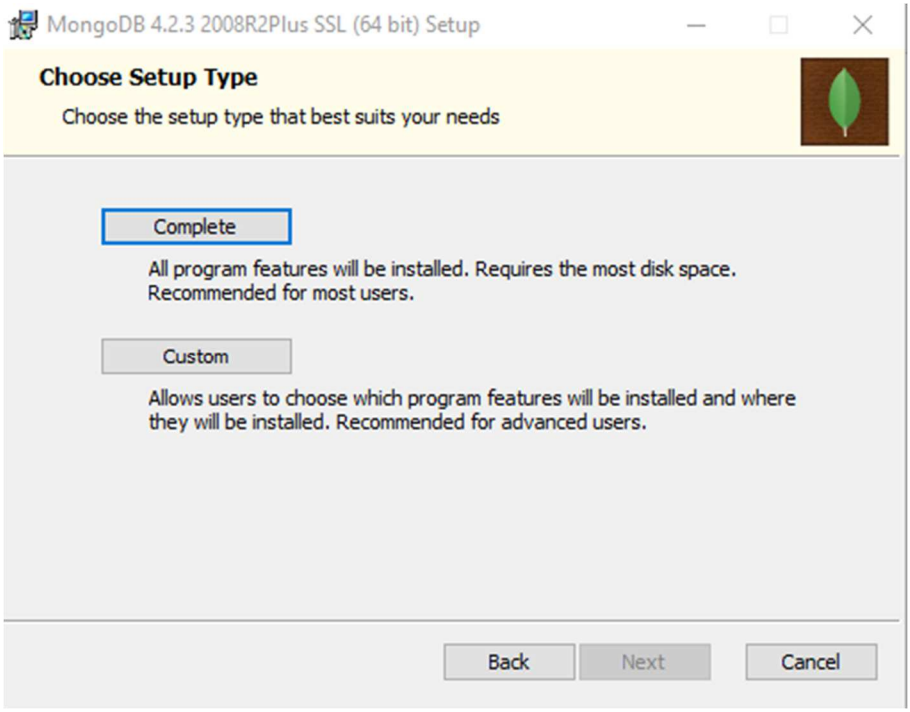


Figura C 3: Selección de tipo de instalación

Nos aparece una ventana donde nos muestra las características de instalación, damos clic en next.

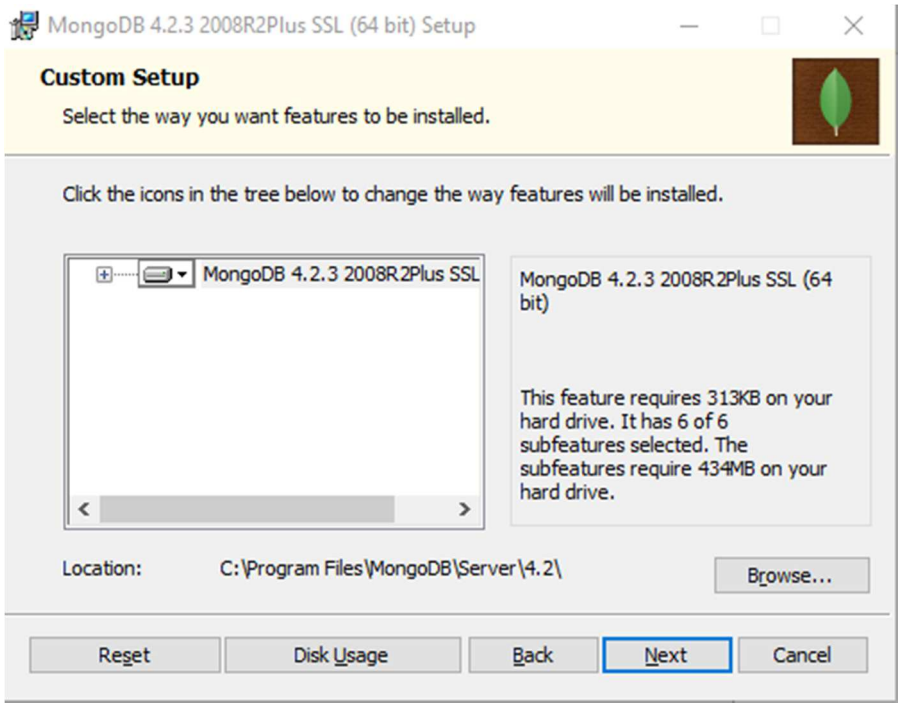


Figura C 4: Selección de características para la instalación

En la ventada de configuración de servicio desmarcamos la pestaña Install MongoDB as a service, ya que lo vamos a utilizar de una manera no frecuente, damos clic a next, luego next.

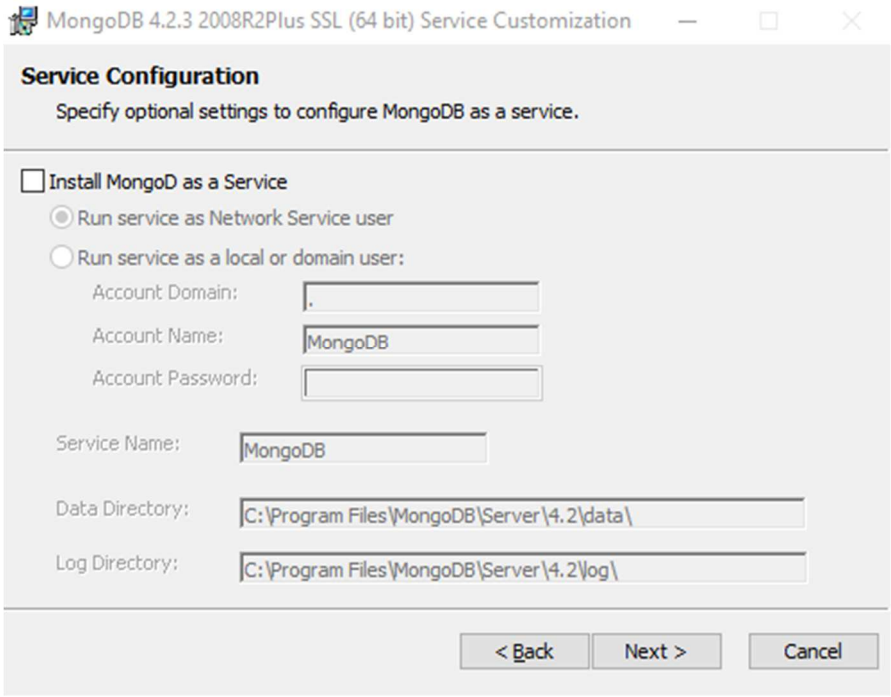


Figura C 5: Configuración de servicios de Mongo DB

Por último, damos clic en install

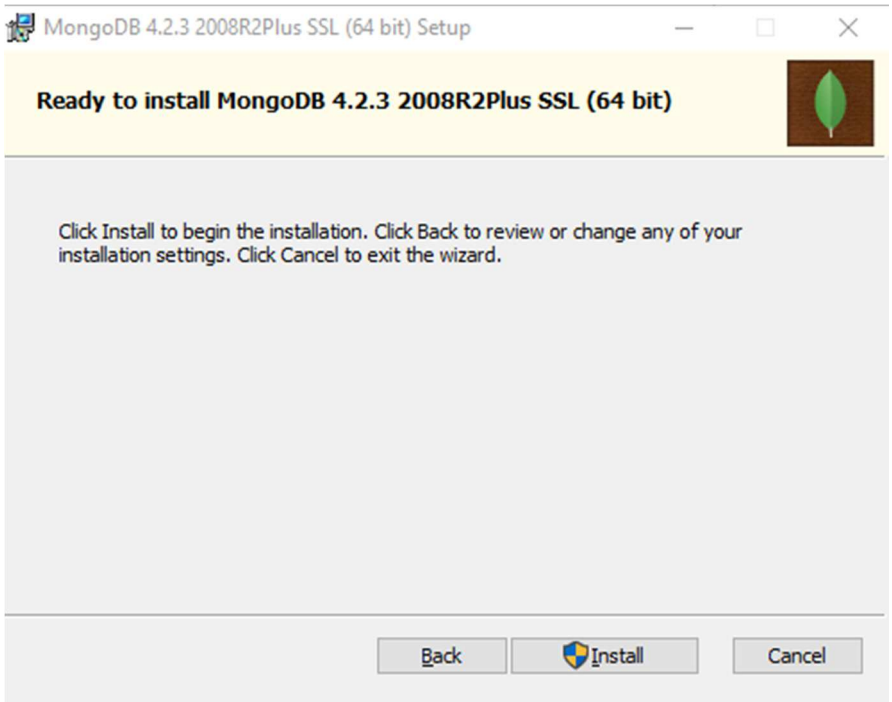


Figura C 6: Aceptación de instalación de Mongo DB

C.2: INSTALACIÓN DE ROBO 3T

Se debe instalar Robo 3T para realizar de manera más sencilla la administración de la base de datos.

Figura C 7: Página de descarga de Robo 3T

Damos clic en download y nos dirigimos a Robomongo y damos clic en download

Figura C 8: Selección de instalador

Seleccionamos el sistema operativo para la descarga

Figura C 9: Descarga del instalador

Al momento que la descarga se ha completado se debe proceder a la instalación

Figura C 10: Asistente de instalación de Robo 3T

Se debe dar clic en siguiente, aceptamos el acuerdo de licencia, damos clic en siguiente y esperamos a que se instale el software.

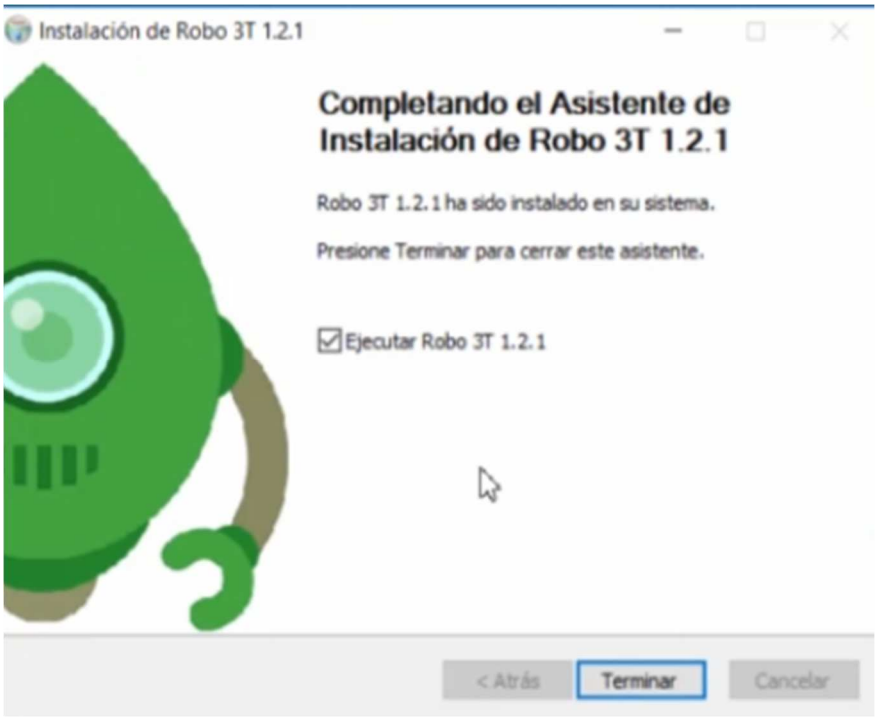


Figura C 11: Finalización de instalación de Robo 3T

Para finalizar se realiza clic en terminar.

APÉNDICE D: DESARROLLO DEL BACKEND

D.1 MODELOS

El modelo de datos que se desarrolló se presenta en la siguiente línea de código, donde se configura los parámetros necesarios para guardar los datos en Mongo DB.

```
'use strict'
var mongoose = require('mongoose');
var Schema = mongoose.Schema;
var datosSchema = Schema({
   nwskey: String,
   appskey: String,
   devaddr: String,
   content: String,
   date:{type: Date, default:Date.now}
});
module.exports = mongoose.model('Datos',datosSchema);
```

D.2 MÓDULOS

MÓDULO DESENCRYP

Este módulo permite la desencriptación de los paquetes enviados por la tarjeta LoRa MOTE.

```
const desencryp = {};
const lorapacket = require('lora-packet');
function decrypt(data,nwkSKey,appSKey) {
   var packet = lorapacket.fromWire(data);
   var valid = lorapacket.verifyMIC(packet, nwkSKey);
   var payload = lorapacket.decrypt(packet, appSKey, nwkSKey);
  return  payload.toString();
};
desencryp.decrypt = decrypt;
module.exports = desencryp;
```

MÓDULO SEPARADOR

Este módulo es el encargo de separar los datos más relevantes para guardar en Mongo DB.

```
const separador = {};
function splitStr(str) {
   var datos = str.split("");
   return datos
};
```

```
separador.splitStr = splitStr;
module.exports = separador;
```

D.3 CONTROLADORES

Los controladores son los encargados de realizar la toma de decisiones para que Mongo DB pueda realizar algunos procesos como guardar, obtener y eliminar.

```
'use strict'

var validator = require('validator');
var DataModel = require('../models/datos');

var controller = {
   datosCurso: (req, res) => {

      return res.status(200).send({
         curso: 'Tesis',
         profersor: 'Fernando Vega'
      });
   },

   test: (req, res) => {
      return res.status(200).send({
         message: 'Soy la accion test de mi controlador de datos'
      });
   },

   save: (req, res) => {
      // recoger parametros por post
      var params = req.body;

      // validar datos (validator)

      try {
         var validate_nwskey = !validator.isEmpty(params.nwskey);
         var validate_appskey = !validator.isEmpty(params.appskey);
         var validate_devaddr = !validator.isEmpty(params.devaddr);
         var validate_content = !validator.isEmpty(params.content);

      } catch (error) {
         return res.status(200).send({
            status:'error',
            message: 'Faltan datos por enviar',
         });
      }
      if (validate_nwskey && validate_appskey && validate_devaddr && validate_content)
{
         // Crear el objeto a guerdar
         var datamodel = new DataModel();
```

```
        // asignar valores
        datamodel.nwskey = params.nwskey;
        datamodel.appskey = params.appskey;
        datamodel.devaddr = params.devaddr;
        datamodel.content = params.content;

        // guardar el dato
        datamodel.save((err,dataStored)=>{
            if(err || !dataStored){
                return res.status(404).send({
                    status:'error',
                    message: 'Los Datos NO se an guardado'
                });
            }
            // devolver respuesta
            return res.status(200).send({
                status:'success',
                dataStored
            });
        })

    } else {
        return res.status(200).send({
            status:'error',
            message: 'Datos no validos'
        });

    }
}, // end save

getDatos:(req,res) => {
    // find
    DataModel.find({}).sort('-_id').exec((err,datos) => {

        if(err){
            return res.status(500).send({
                status:'error',
                message: 'Error al devolver los Datos'
            });
        }

        if(!datos){
            return res.status(404).send({
                status:'error',
                message: 'No hay Datos que mostrar'
            });
        }

        return res.status(200).send({
            status:'success',
            datos
        });

    });
```

```
},// end getdatos

getdato: (req,res)=>{
   // recoger el ide de la URL
   var datoId = req.params.id;

   // comprobar que existe
   if(!datoId || datoId == null){
      return res.status(404).send({
         status:'error',
         message: 'No existe el dato'
      });
   }

   //buscar el dato
   DataModel.findById(datoId,(err,dato) =>{

      if(err || !dato){
         return res.status(404).send({
            status:'error',
            message: 'No existe el dato'
         });
      }

        //Devolver en json
      return res.status(200).send({
      status:'success',
      dato
      });

   });

}, // end updata

delete:(req,res) => {
   // recoger el id de la url
   var datoId = req.params.id;

   // find and delete
   DataModel.findOneAndDelete({_id:datoId},(err,datoRemoved)=>{
      if(err){
         return res.status(500).send({
            status:'error',
            message: 'No al borrar'
         });
      }
      if(!datoRemoved){
         return res.status(404).send({
            status:'error',
            message: 'No se ha borrado el dato, posiblemente no exista'
         });
      }
      return res.status(200).send({
         status:'success',
         dato: datoRemoved
```

```
            });
        });

    }, // end delete

}; // end controller

module.exports = controller;
```

D.4 RUTAS

Las rutas permiten la comunicación del frontend con el backend.

```
'use strict'

var express = require('express');
var datosController = require('../controllers/datos');

var router = express.Router();

// rutas de pruebas
router.post('/datos-curso',datosController.datosCurso);
router.get('/test-de-controlador',datosController.test);

// rutas de datos
router.post('/save',datosController.save);
router.get('/datos',datosController.getDatos);
router.get('/dato/:id',datosController.getdato);
router.delete('/dato/:id',datosController.delete);

module.exports = router;
```

D.5 CONFIGURACIÓN DE WEB

La configuración para la web se realizó mediante el siguiente código

```
'use strict'

// cargar modulos de node para crear el servidor
var express = require('express');
var bodyParser = require('body-parser');

// ejecutar expres
var app = express();

// cargar rutas o ficheros
var datos_routes = require('./routes/datos')

// Middlewares
app.use(bodyParser.urlencoded({extended:false})); //cargar el bodyparser
app.use(bodyParser.json());

// Cors
```

```
// Configurar cabeceras y cors
app.use((req, res, next) => {
   res.header('Access-Control-Allow-Origin', '*');
   res.header('Access-Control-Allow-Headers', 'Authorization, X-API-KEY, Origin, X-
Requested-With, Content-Type, Accept, Access-Control-Allow-Request-Method');
   res.header('Access-Control-Allow-Methods', 'GET, POST, OPTIONS, PUT, DELETE');
   res.header('Allow', 'GET, POST, OPTIONS, PUT, DELETE');
   next();
});

//Anadir prefijos o rutas / cargar rutas
app.use('/api',datos_routes);

//exportar modulos
module.exports = app;
```

D.6 CONFIGURACIÓN DE GATEWAY

Para la captura de los datos que envía el programa de simulación del Gateway se realizó la siguiente configuración en el servidor.

```
'use strict'
// cargar modulos
const dgram = require('dgram');
const gateway = dgram.createSocket('udp4');
const separador = require('./modulos/separador');
const desencryp = require('./modulos/desencryp');
const modelo = require('./models/datos');

var aux = [];
var resp = [];

gateway.on('message', (msg, rinfo) => {
   if (rinfo.size > 108) {

      aux = separador.splitStr(msg.toString()); // separando datos del paquete enviado
      aux = aux[aux.length - 2]; // seleciono la informacion principal
      resp = desencryp.decrypt(new Buffer.from(aux.toString(), 'base64'), new
Buffer.from('2B7E151628AED2A6ABF7158809CF4F3C', 'hex'), new
Buffer.from('3C8F262739BFE3B7BC0826991AD0504D', 'hex'));
      resp = resp;
      let dato = new modelo({
         nwskey:'2B7E151628AED2A6ABF7158809CF4F3C',
         appskey:'3C8F262739BFE3B7BC0826991AD0504D',
         devaddr: '1100000F',
         content: resp})
      dato.save();

   } else {
   }
});

module.exports = gateway;
```

D.7 CONFIGURACIÓN DEL SERVIDOR

Aquí se presenta la configuración global del servidor

```
'use strict'

var mongoose = require('mongoose'); // pide la libreria mongoose
var app = require('./app_web');
var port = 3900;

var gateway= require('./app_gateway');
var port_1 = 1700;

mongoose.set('useFindAndModify', false);
mongoose.Promise = global.Promise;
mongoose.connect('mongodb://localhost:27017/LoRaWAN_Database', { useNewUrlParser:
true })
   .then(() => {
      // pasa varios parametros al servidor
      /*'mongodb://localhost:27017/LoRaWAN_Database' --> es la direccion
      del servidor, el puerto y el nombre de la base de datos*/
      /* { useNewUrlParser:true} --> permite utilizar las nuevas funcionalidades de
mongoose */

      // crear servidor y escuchar peticiones http
      app.listen(port, () => {
         console.log('Servidor corriendo en http://localhost: ' + port);
      });
      gateway.on('listening', () => {
         const address = gateway.address();
         console.log(`server listening ${address.address}:${address.port}`);
        });
      gateway.bind(port_1);

   });
```

D.8 SIMULACIÓN DEL GATEWAY

```
const dgram = require('dgram');

const message = Buffer.from('
å4VxeC!{"rxpk":[{"tmst":2354794796,"chan":2,"rfch":0,"freq":868.500000,"stat":1,
"modu":"LORA","datr":"SF12BW125","codr":"4/5","lsnr":9.3,"rssi":-
14,"size":15,"data":"QA8AABEACAB4kpX6WCI4"}]}');

const client = dgram.createSocket('udp4');

client.send(message, 1700, '0.0.0.0', (err) => {

  client.close();

});
```

APÉNDICE E: INSTALACIÓN DE LORA UTILITY

Iniciamos el instalador dando doble clic, y esperamos que se ejecute.

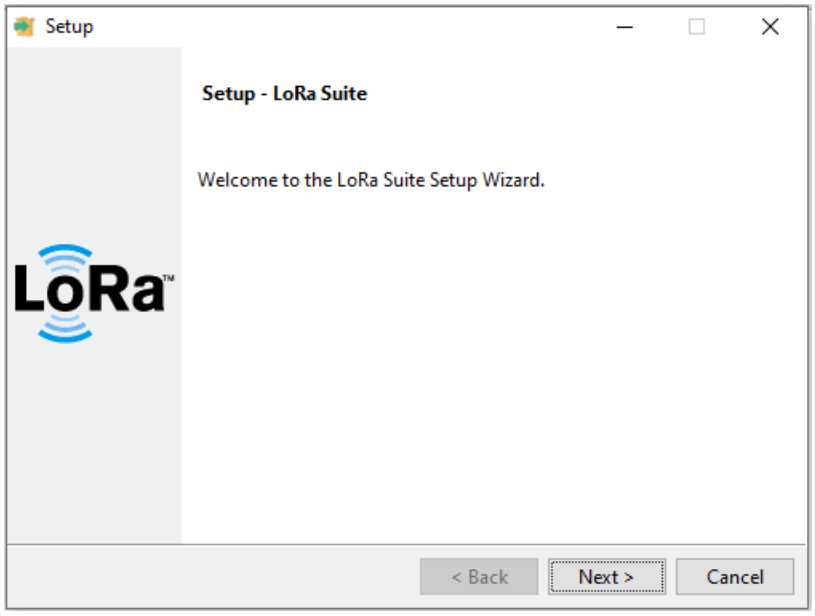


Figura E 1: Ventana de Instalación de LoRa Suite

Aceptamos la licencia y damos clic en next

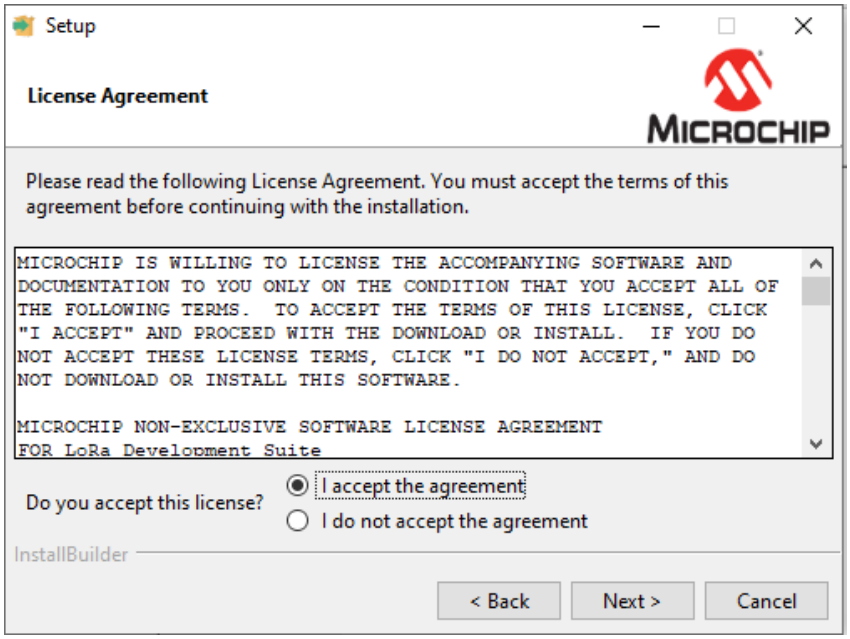


Figura E 2: Ventana de condición de licencia de Microchip

Muestra el directorio en el cual se va a instalar el software, lo dejamos por defecto el mismo y damos clic en next.

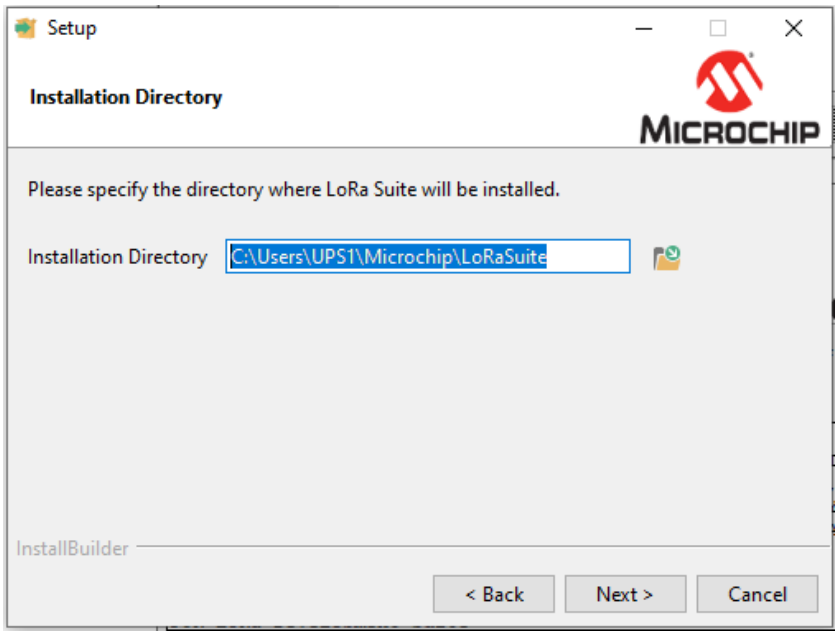


Figura E 3: Directorio de instalación de LoRa Suite

Seleccionamos todos los componentes para que se instale correctamente, damos clic en next.

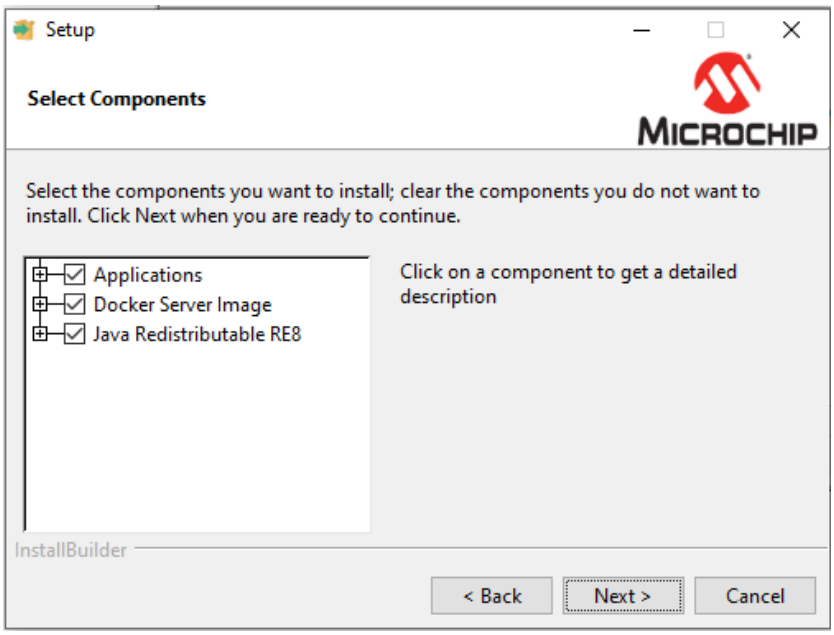


Figura E 4: Componentes de Instalación del paquete LoRa Suite

Presenta una ventana la cual nos avisa que la instalación se va a realizar, damos clic en next, y esperamos que se instale.

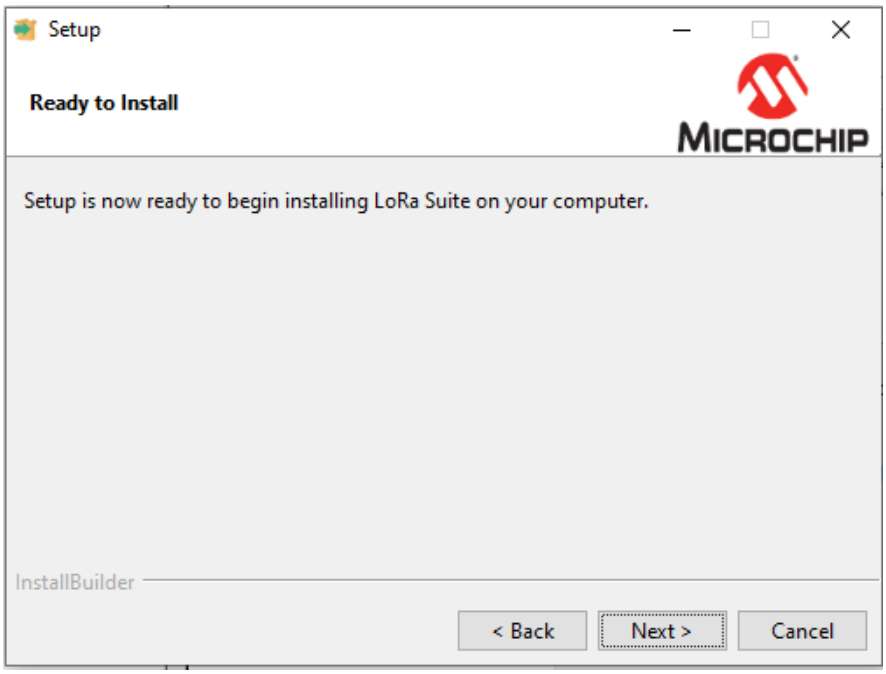


Figura E 5: Preparación de Instalación del paquete LoRa Suite

Para finalizar damos clic en Finish

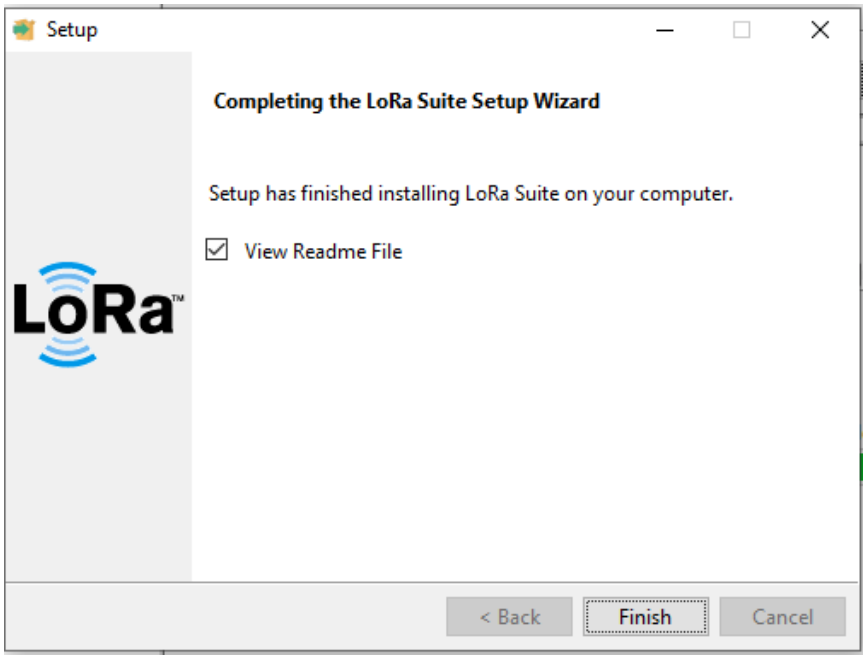


Figura E 6: Ventana de finalización de Instalación

APÉNDICE F: INSTALACIÓN DE DOCKER TOOLBOX

Ejecutamos el instalador y damos clic en next.

Figura F 1: Ventana de Instalación de Docker Toolbox

Mostrará una ventana con la dirección en la cual se va a instalar, la dejamos esa misma por defecto y damos clic en next.

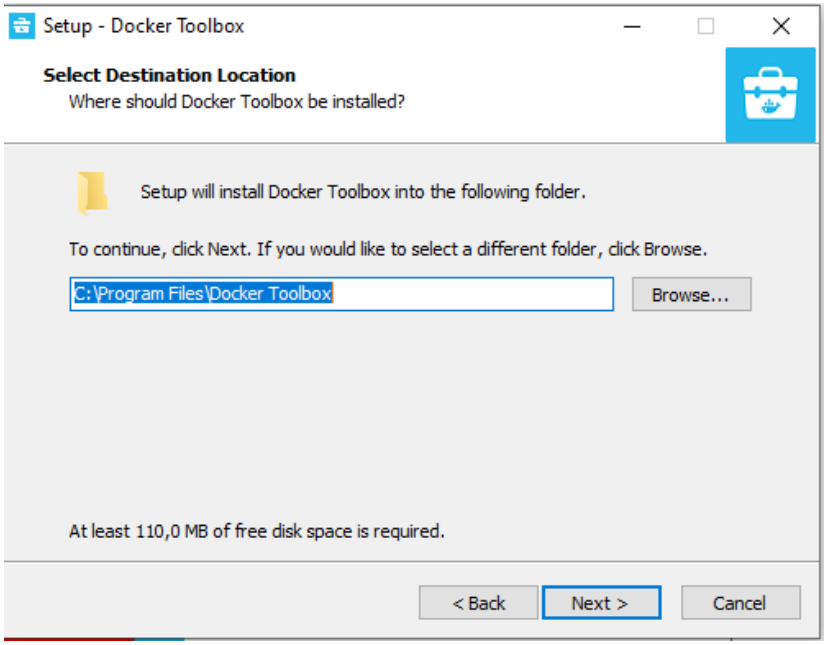


Figura F 2: Ventana de directorio de instalación

Seleccionamos los componentes que deseemos instalar y damos clic en next, en el caso de que uno de ellos ya este instalado no es necesario marcarlo,

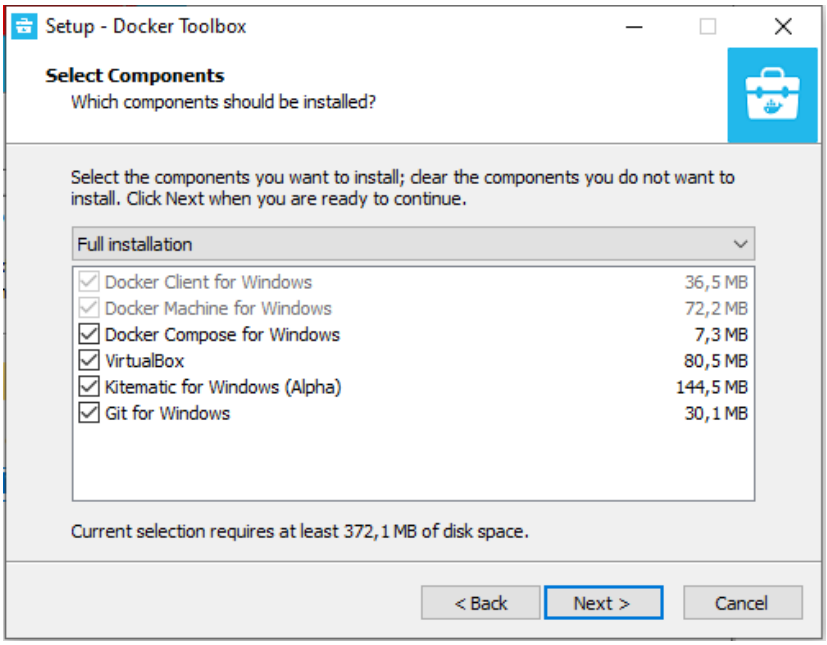


Figura F 3: Selección de componentes a instalar

Seleccionamos las herramientas adicionales que necesitemos y damos clic en next.

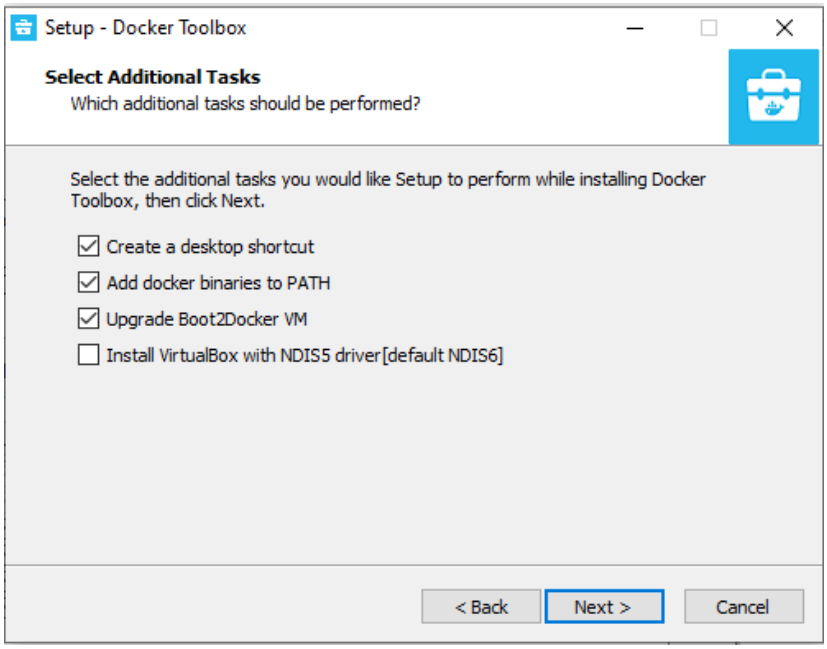


Figura F 4: Selección de tareas adicionales

Muestra una ventana en la cual se presenta todas las configuraciones que se realizaron para la instalación del software, en el caso de que todo esté bien damos clic en Install y esperamos que se ejecute.

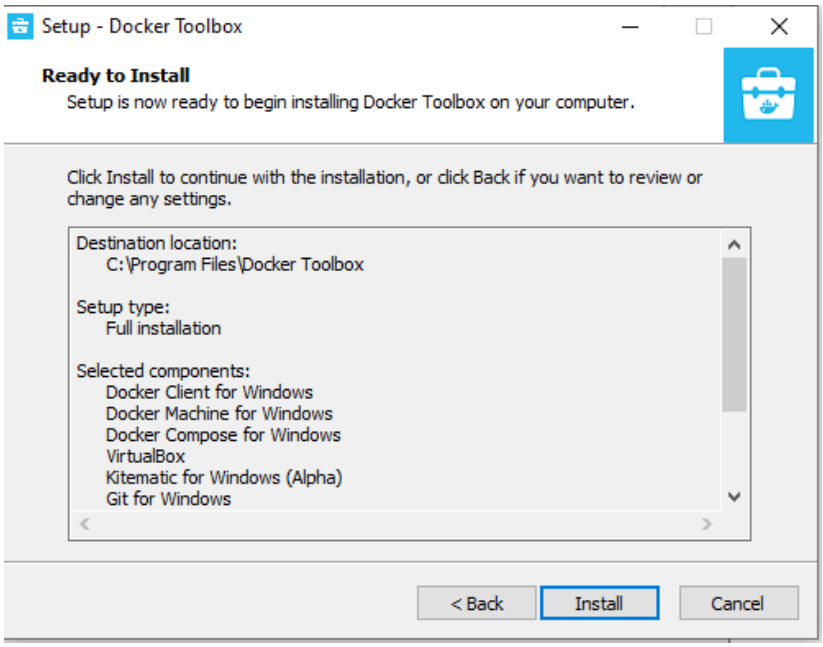


Figura F 5: Ventana de preparación de paquetes a instalar

Para finalizar la instalación damos clic en Finish

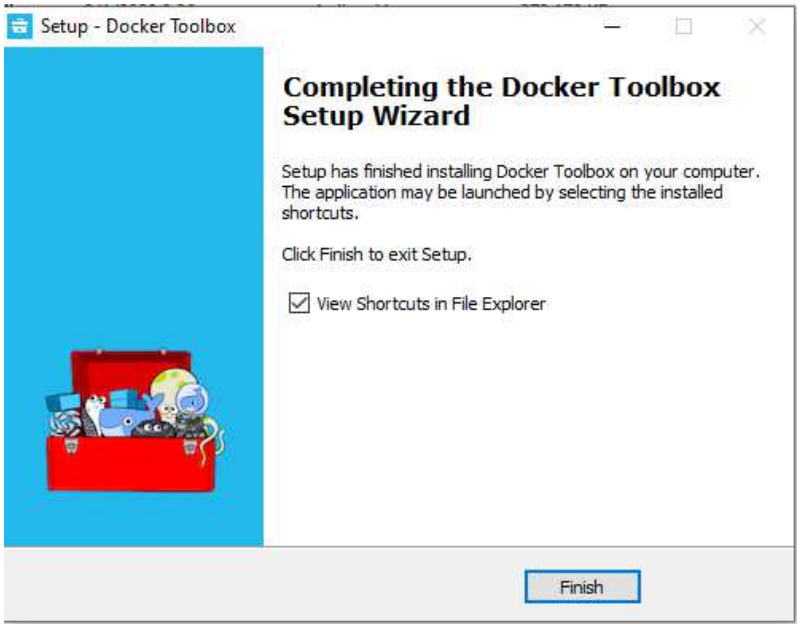


Figura F 6: Ventana de finalización de instalación

Para que no se tenga problemas por los cortafuegos, se debe ejecutar como administrador a Docker Quickstart Terminal.

APÉNDICE G: CONFIGURACIÓN DE PUERTOS EN VIRTUAL BOX

La configuración de reenvió de puertos en la máquina virtual se realiza para el forwarding de datos de enlace descendentes (downlink) y ascendentes (uplink), para ello se debe ir a la configuración de la VM.

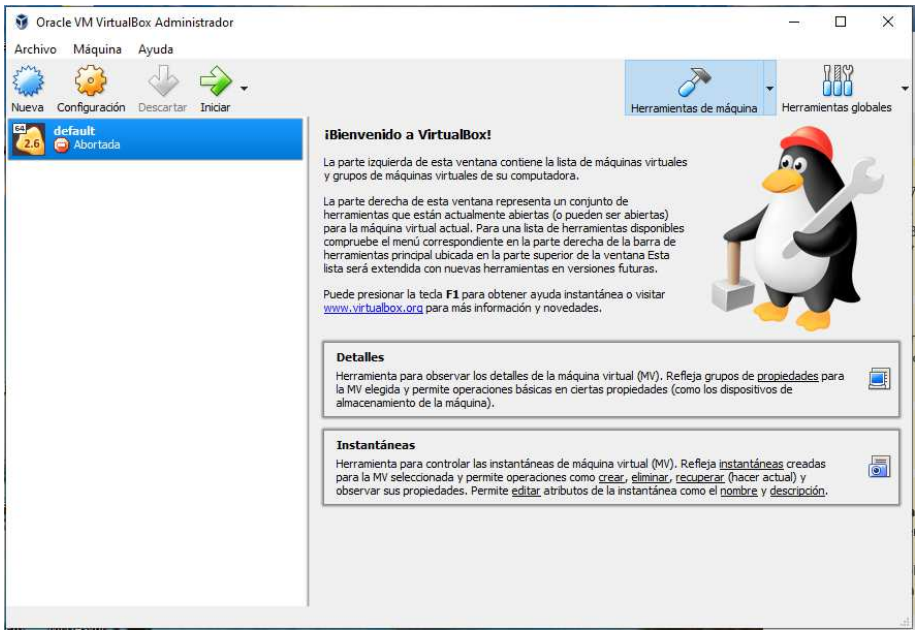


Figura G 1: Ventana de Virtual Box

Se desplegará una ventada de configuraciones en la cual seleccionamos Red como en la figura

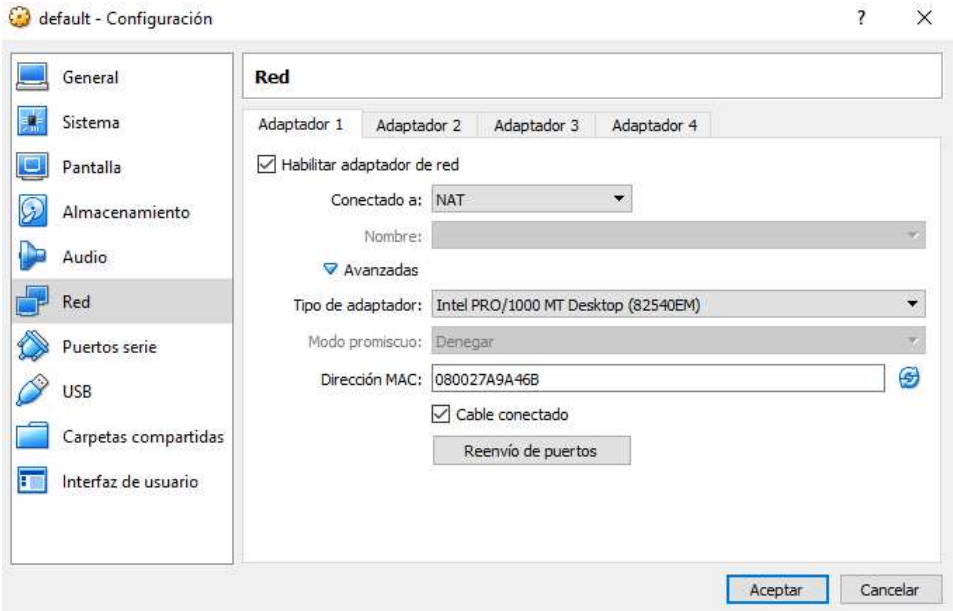


Figura G 2: Panel de configuración de Red en Virtual Box

Al seleccionar reenvió de puertos se despliega una ventada en la cual se debe agregar nuevas reglas para el forwarding de datos, haciendo clic en el símbolo como se muestra en la figura, añadimos las nuevas reglas

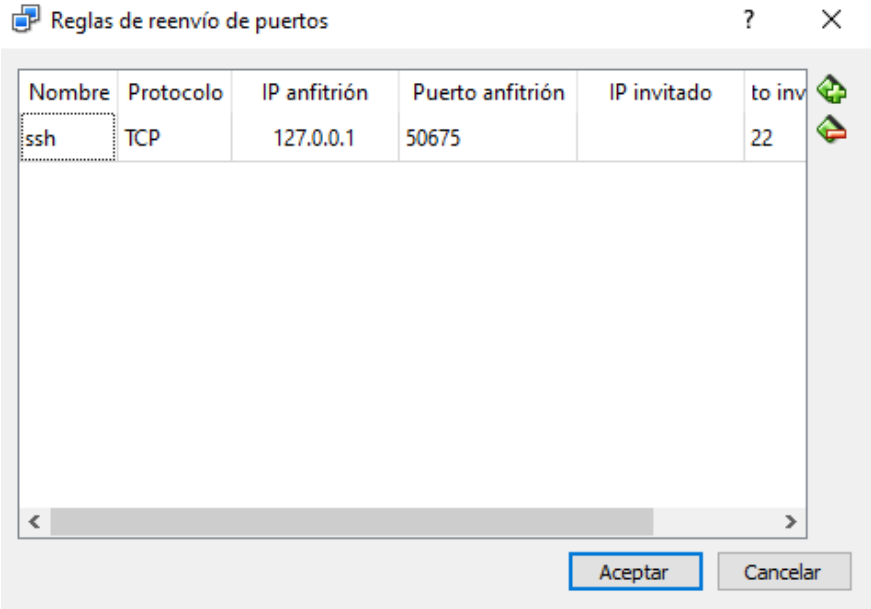


Figura G 3: Ventana de configuración de reenvió de puertos

Las reglas que se deben añadir se presentan en la figura

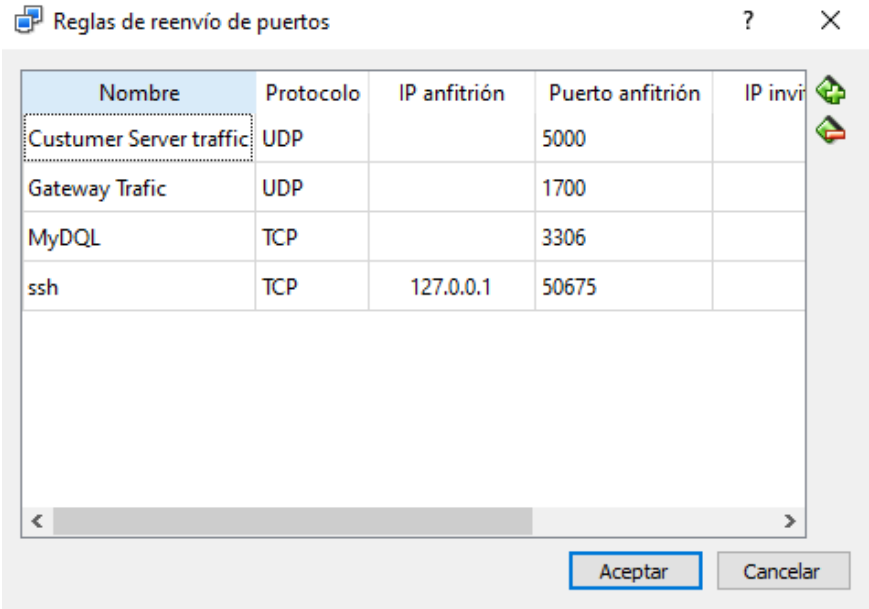


Figura G 4:Reenvío de Puertos configurados

APÉNDICE H: CARGA DE IMAGEN DOCKER

Se debe salir del directorio predeterminado con “cd” y se debe ingresar el directorio donde se encuentra la imagen a cargar “mchplora”.

NOTA: hay que tener en cuenta ya está dentro del directorio “USER”.

```
Fernando Vega@DESKTOP-TFRR85O MINGW64 /c/Program Files/Docker Toolbox
$ cd

Fernando Vega@DESKTOP-TFRR85O MINGW64 ~
$ docker load < microchip\LoRaSuite\Docker\mchplora
bash: microchipLoRaSuiteDockermchplora: No such file or directory

Fernando Vega@DESKTOP-TFRR85O MINGW64 ~
$ docker load < microchip/LoRaSuite/Docker/mchplora
628a915caf0d: Loading layer [==================================================>]  130.9MB/130.9MB
5f70bf18a086: Loading layer [==================================================>]  1.024kB/1.024kB
6e36e19d0ae3: Loading layer [==================================================>]  344.6kB/344.6kB
72e9172ed167: Loading layer [==================================================>]  1.536kB/1.536kB
fd23cd6fd9b0: Loading layer [==================================================>]  33.78MB/33.78MB
7778062a69d0: Loading layer [==================================================>]  25.09kB/25.09kB
aa5dbe59dec3: Loading layer [==================================================>]  3.584kB/3.584kB
edbcfddf7fa8: Loading layer [==================================================>]  166.3MB/166.3MB
867ef509b54b: Loading layer [==================================================>]   5.12kB/5.12kB
3d27850363bf: Loading layer [==================================================>]  4.608kB/4.608kB
993b22a1c8f9: Loading layer [==================================================>]  14.21MB/14.21MB
ae08779d6105: Loading layer [==================================================>]  2.615MB/2.615MB
8f80b9b6fe34: Loading layer [==================================================>]  17.41kB/17.41kB
c5ed15ed2bc6: Loading layer [==================================================>]  4.096kB/4.096kB
6bdcc537b10e: Loading layer [==================================================>]  22.02kB/22.02kB
7e337c4e8a05: Loading layer [==================================================>]   2.56kB/2.56kB
9ba2d91b08d1: Loading layer [==================================================>]  20.48kB/20.48kB
f77d57993b74: Loading layer [==================================================>]  22.53kB/22.53kB
Loaded image: mchplora:1.2
```

Figura H 1: Carga de imagen del servidor LoRa

Se comprueba que la imagen se cargó con el comando “docker images”

```
Fernando Vega@DESKTOP-TFRR85O MINGW64 ~
$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mchplora            1.2                 36173b91626b        3 years ago         341MB
```

Figura H 2: Verificación de la imagen del servidor

Se debe crear el docker container

```
Fernando Vega@DESKTOP-TFRR85O MINGW64 ~
$ docker create --name lora_server -p 1700:1700/UDP -p 3306:3306/TCP -p 5000:5000/UDP mchplora:1.2
87926bc2358924ab2d9c155f73bf7594b76a826814f10fef9da8120a50b5fc7c
```

Figura H 3: Configuración de los puertos de reenvío en Docker

Creado el docker container se debe se debe iniciar al servidor para ello se debe ingresar la línea de comando “docker start lora_server”.

```
Fernando Vega@DESKTOP-TFRR85O MINGW64 ~
$ docker start lora_server
lora_server
```

Figura H 4: Inicio de servidor lora_server

Para detener el servidor se debe ingresar la siguiente línea de comando “docker stop lora_server”

```
Fernando Vega@DESKTOP-TFRR85O MINGW64 ~
$ docker stop lora_server
lora_server
```

Figura H 5: Paro de servidor lora_server

En el cado de reiniciar el servidor se debe ingresar el comando “docker restart lora_server”

APÉNDICE I: AUTOCONFIGURACIÓN DE TARJETA LORA MOTE

En la configuración de LoRa MOTE se debe ingresar dando clic en el dispositivo "RN Module", seleccionar la pestaña "Lora WAN", seleccionar ABP y dar clic en "config for auto-create application server"

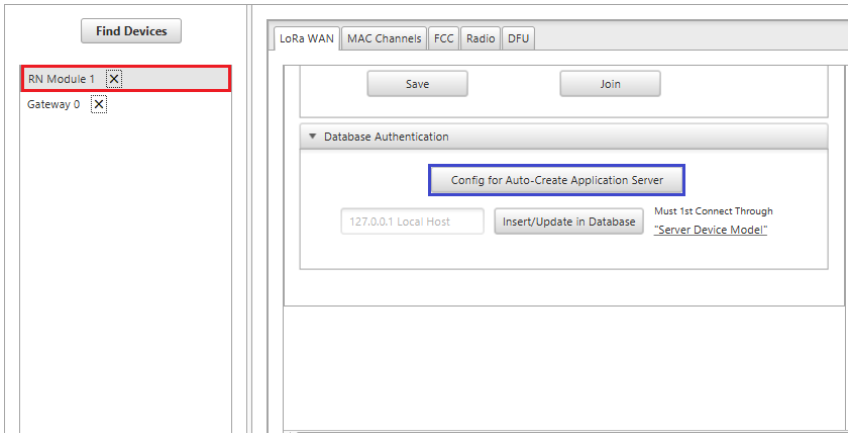


Figura I 1: Autoconfiguración de módulos MOTEs

El botón "config for auto-create application server" preconfigura el dispositivo con la clave de sesión de red especificada (NwSKey) y la clave de sesión de aplicación (AppSKey), permitiendo el acceso instantáneo de las llaves al servidor.

La dirección del dispositivo se completará automáticamente, utilizando los ocho bytes inferiores del identificador único extendido de hardware (HwEUI). El módulo emitirá automáticamente un comando SAVE y JOIN al presionar el botón

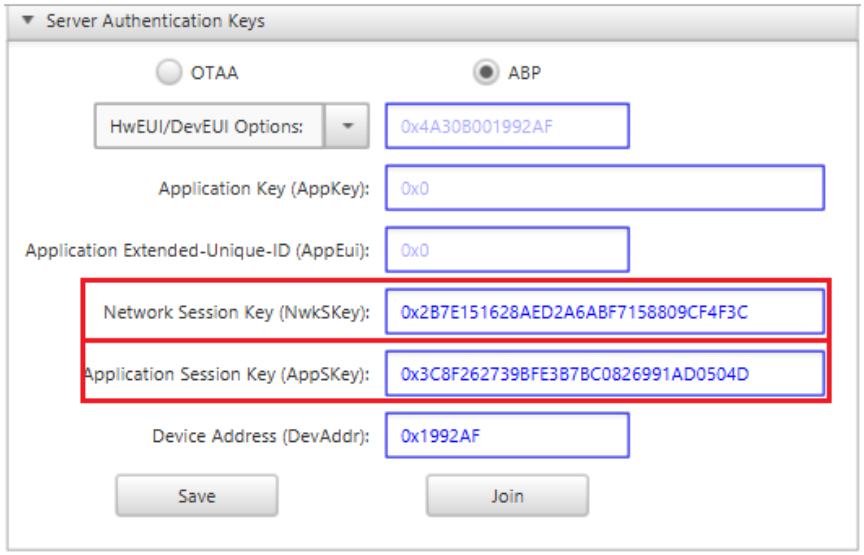


Figura I 2: Parámetros rellenados automáticamente por Lora Utility

APÉNDICE J: IMPLEMENTACIÓN DE CONEXIÓN OTAA

La configuración se realiza dentro de la pestaña del servidor, por tanto, dentro de LoRa Development Utility, seleccionamos server.

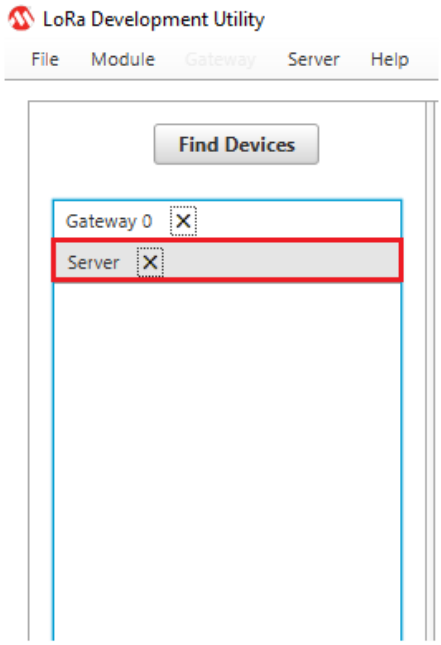


Figura J 1: Selección de servidor

Debemos configura los parámetros de la pestaña Server Aplication donde se colocarán los siguientes parámetros de la tabla.

Tabla J 1: Parámetros de configuración para conexión OTAA

Descripción	Parámetros
Application Extended-Unique-Id	0x98fe
Sever Application Name	Mi Aplicación OTAA
Server Application Owner	Microchip:C1230

Al momento de ingresar todos los parámetros damos clic en insert/update in Database, ya ingresados los parámetros se debe reiniciar el servidor Docker.

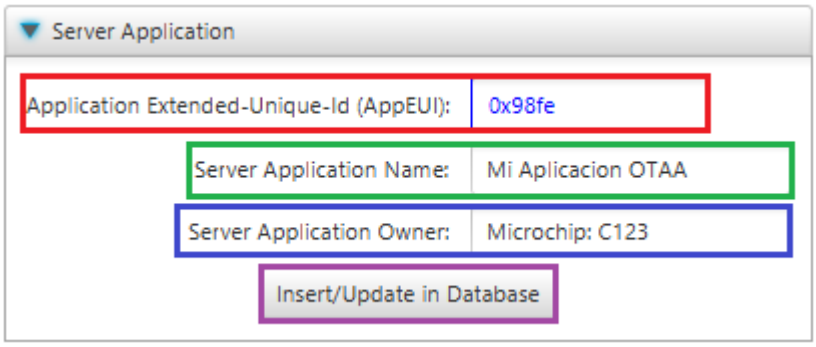


Figura J 2: Ingreso de parámetros del servidor OTAA al servidor Docker

Se crearán las credenciales de los end-device (MOTEs) en el servidor, por lo tanto, dentro de Server configuramos los parámetros de la pestaña Non-provisioned (OTAA) como se especifica en la tabla.

Tabla J 2: Parámetros de ingreso para dispositivos MOTEs

Descripción	Parámetros
Sever Application EUI	0x98fe
Application key (APPkey)	0x10f
Device Extended- Unique-Id (DevEUI)	0x809

Como se muestra en la figura, se debe ingresar los datos del end-device para que lo reconozca el servidor.

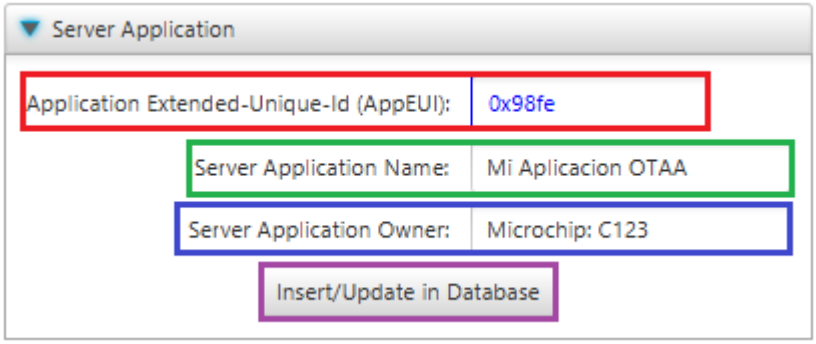


Figura J 3: Ingreso de parámetros de MOTEs en el servidor

Para finalizar la configuración se debe ingresar al RN MODULE, en la pestaña server/ LoRAWAN/ Authentification Keys realizamos las configuraciones respectivas.

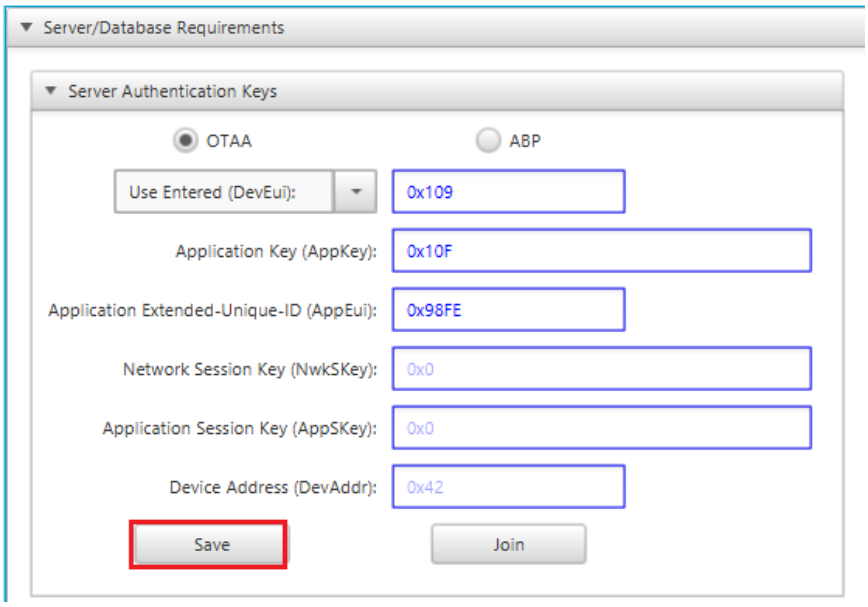


Figura J 4: Configuración de parámetros OTAA en los módulos MOTEs

APÉNDICE K: CONFIRMACIÓN DE CONEXIÓN OTAA

Seleccionamos Server/Database y nos dirigimos a la pestaña Selec Database Table View y seleccionamos ServerAplication, donde nos muestra nombre de la nueva base de datos para almacenar los datos de los dispositivos OTAA.

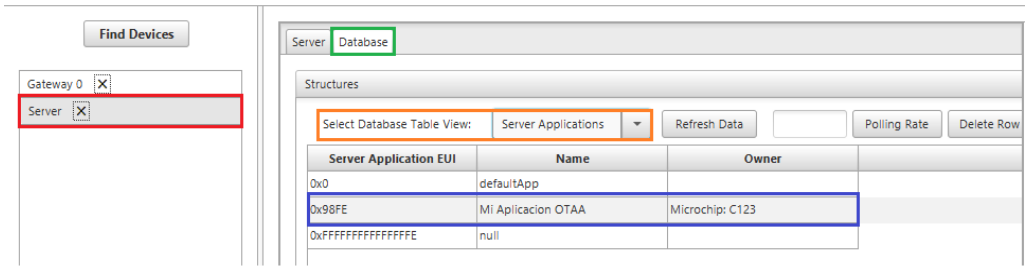


Figura K 1: Confirmación de creacion de servidor para conexion OTAA

Seleccionamos Server/Database y nos dirigimos a la pestaña Selec Database Table View y seleccionamos OTAA Devices, esto permite ver cuántos dispositivos OTAA están dentro de la red.

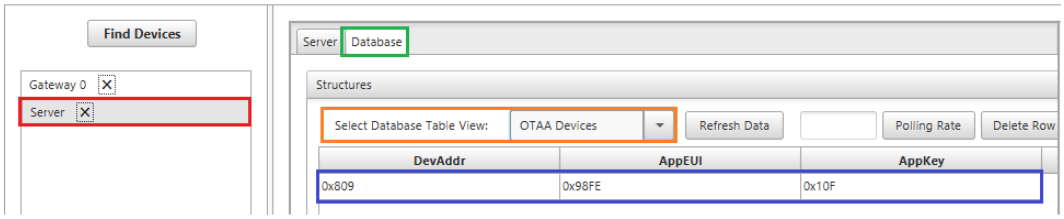


Figura K 2: Confirmación de ingreso de parámetros MOTEs al servidor

Confirmado la configuración correcta se puede ya ejecutar las configuraciones dentro del MOTE.

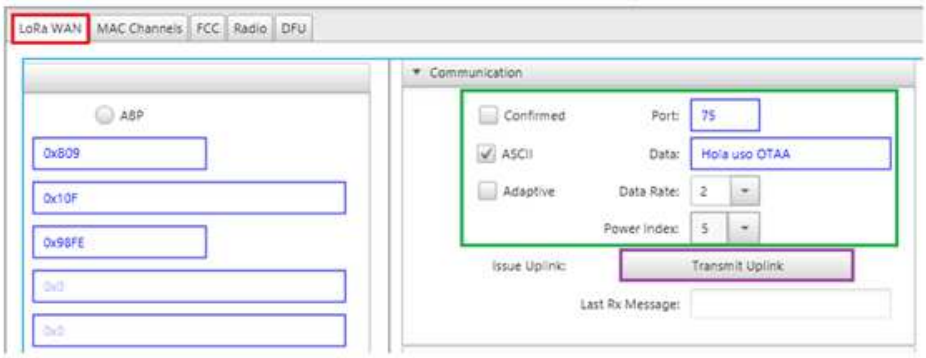


Figura K 3: Envío de caracteres por medio de Lora Utility

La confirmación del mensaje se puede ver en server/database y en la pestaña Selec Database Table View, seleccionamos ServerAplication

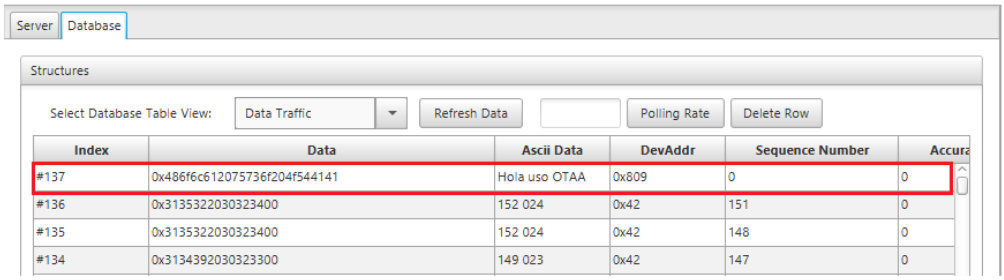


Figura K 4: Confirmación de envió de caracteres al servidor

APÉNDICE L: USO DE PUERTOS GPIO DE LA TARJETA MOTE

Abrimos el programa LoRa Development Utility conectamos la tarjeta mote, abrimos el RN Module, y nos dirigimos a la pestaña Radio, GPIO Control, como se puede ver en la figura.

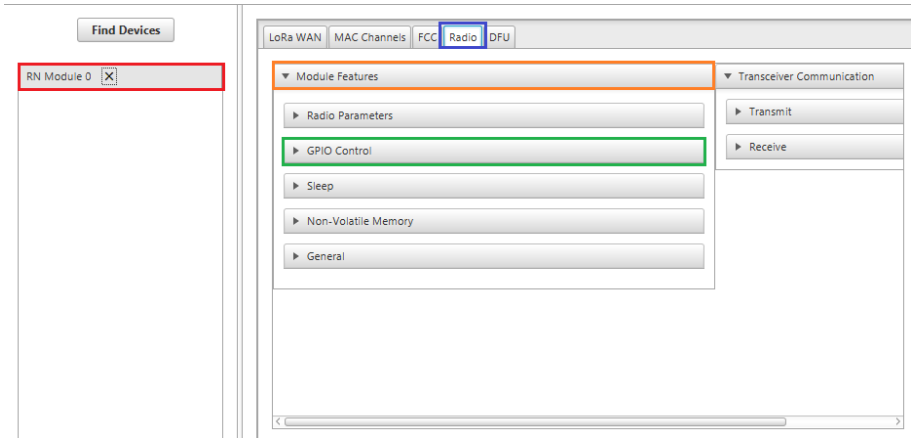


Figura L 1: Panel de configuración para puertos GPIO

Dentro del sub-panel tenemos los nombres de los pines los cuales se pueden configurar entradas digitales, salidas digitales y pines analógicos.

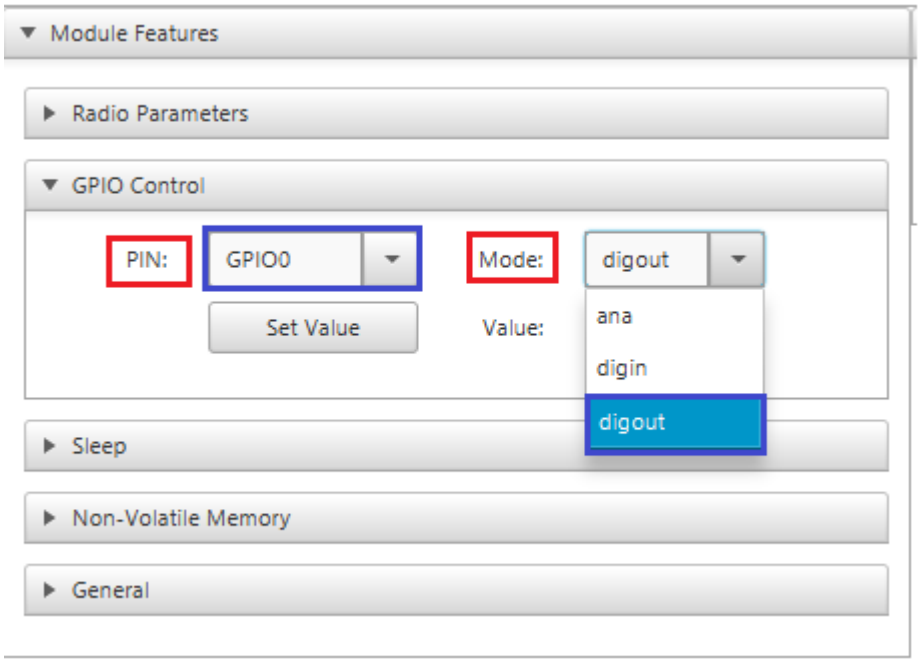


Figura L 2: Configuración de GPIO

Seleccionamos en PIN el GPIO 0, el modo digOut y en value se cambia a 1

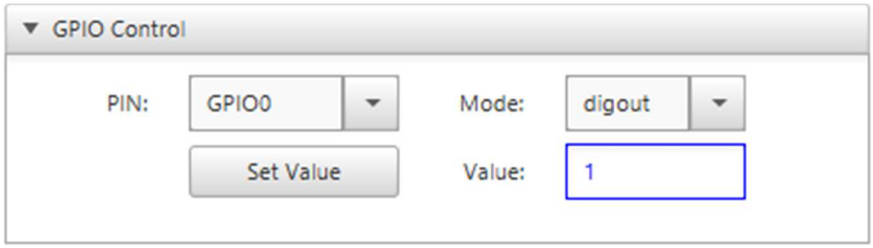


Figura L 3: Estado del GPIO

Para confirmar los cambios realizados, damos clic en set Value.

APÉNDICE M: INSTALACIÓN DE MPLAB X IDE

Al hacer doble clic en el instalador saldrá la siguiente ventana y damos clic en next

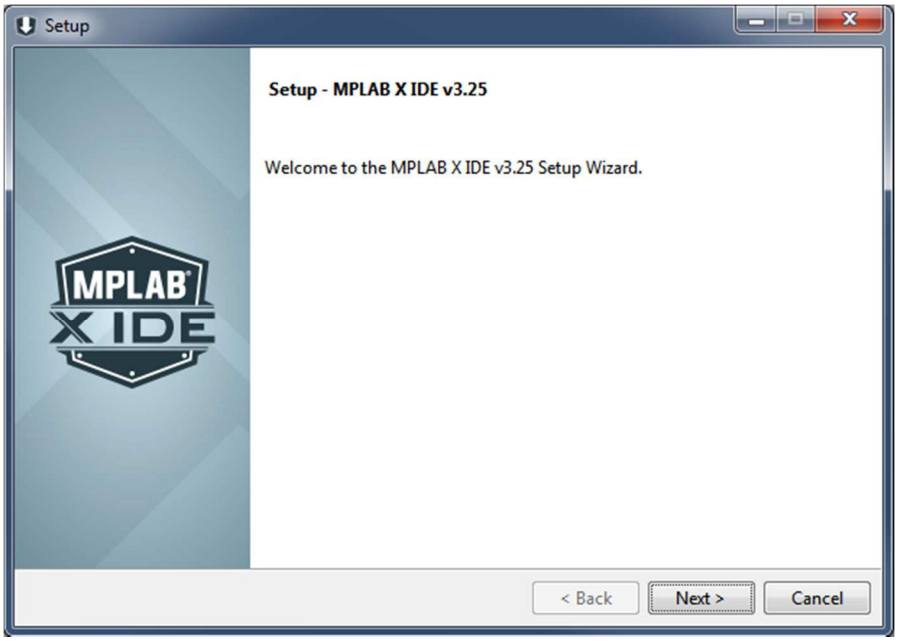


Figura M 1: Ventana de Instalación de MPLAB

Aceptamos los términos de la licencia y damos clic en next

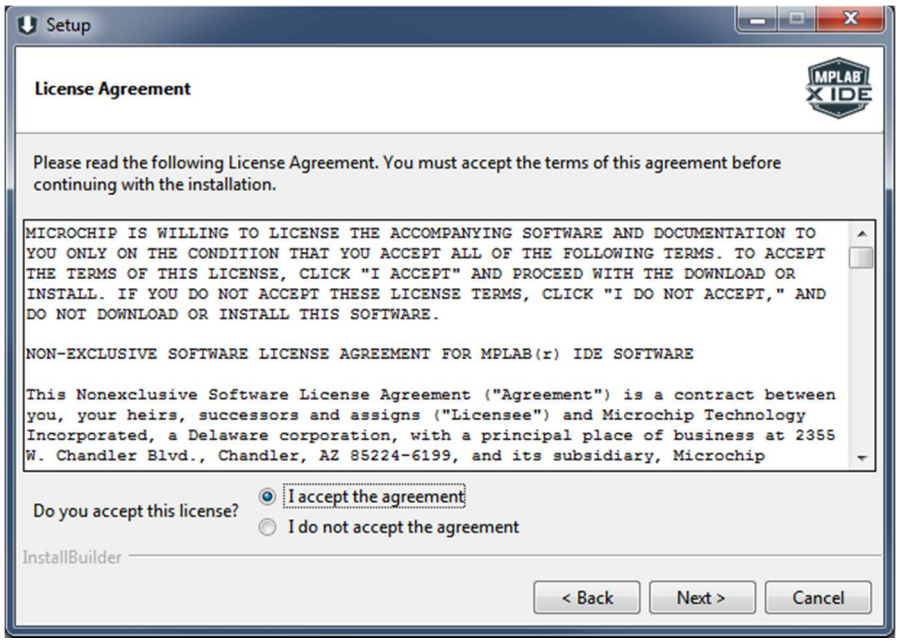


Figura M 2: Acuerdo de licencia

Dejamos la ruta por defecto que realiza MPLAB X IDE, en el caso de cambiarla solo se tiene que dirigir a la ruta que el usuario quiera.

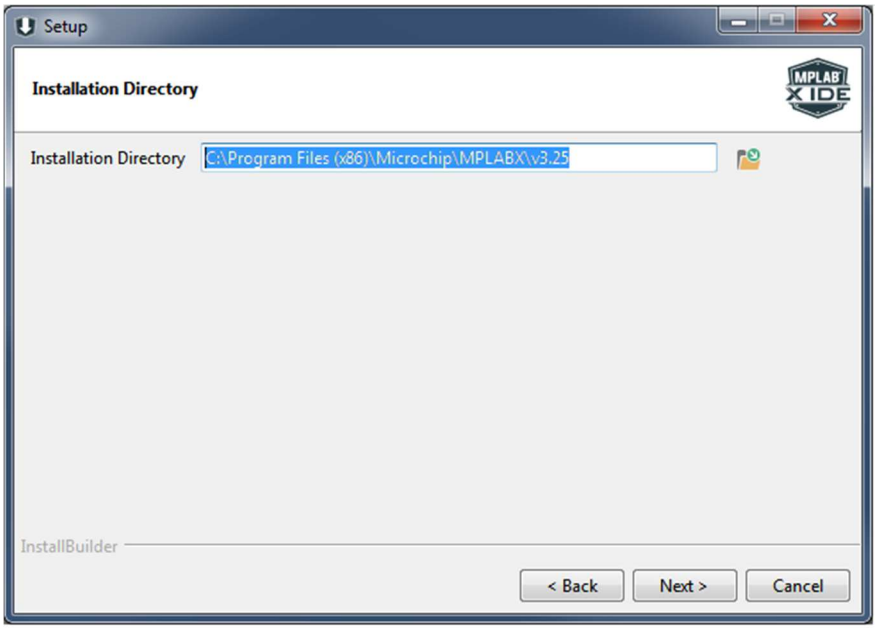


Figura M 3: Directorio de Instalación

Señalar los como en la imagen, para la instalación de los dos softwares, damos clic en next.

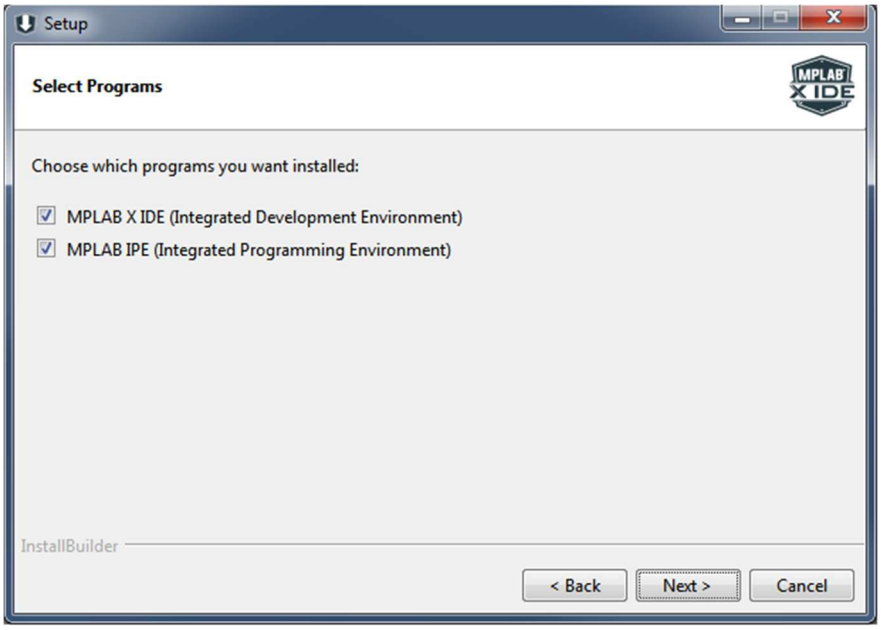


Figura M 4: Selección de programas adicionales

El instalador nos indica que está listo para la instalación, damos clic en next

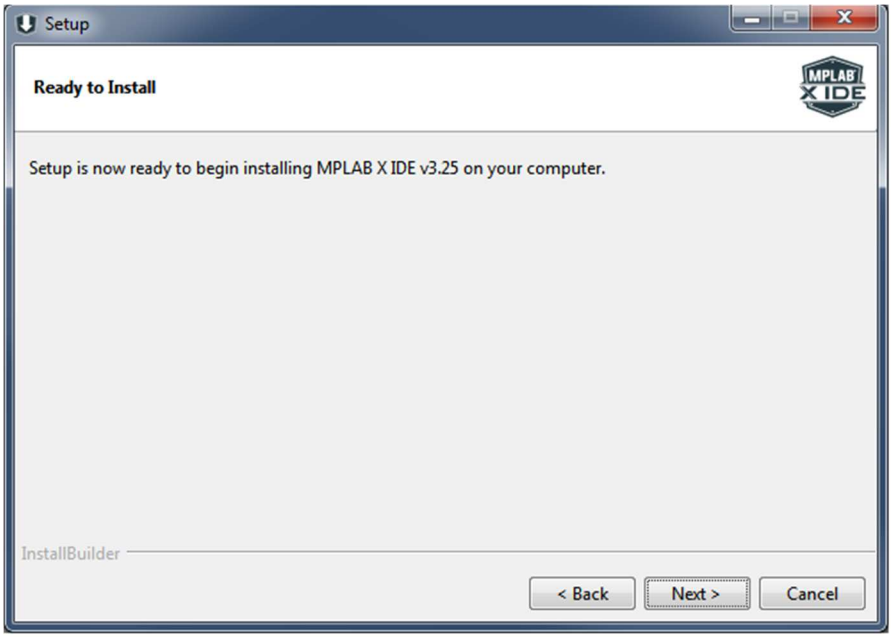


Figura M 5: Panel de preparación de instalación de MPLAB

Esperamos que se instale el software, casi al terminar la instalación nos saldrá un cuadro de la instalación de un driver, aceptamos para permitir la comunicación con cualquier tarjeta de microchip.

Figura M 6: Instalación de driver para placas de MPALB

Para finalizar damos clic en Finish.

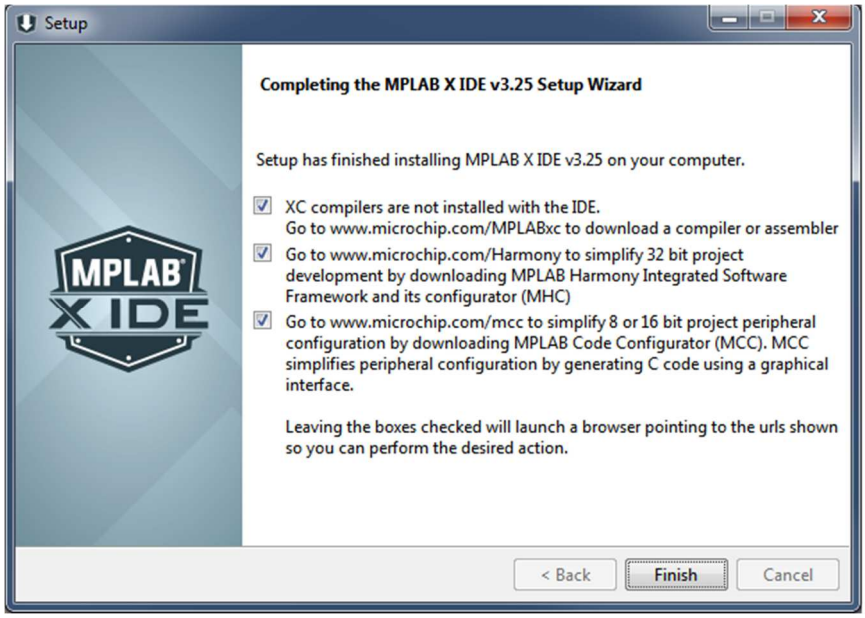


Figura M 7: Finalización de Instalación

APÉNDICE N: CREACIÓN DE UN PROYECTO EN MPLAB X IDE

Para realizar un nuevo proyecto se de abrir el MPLAB IDE X e ir a 'File/New Project'

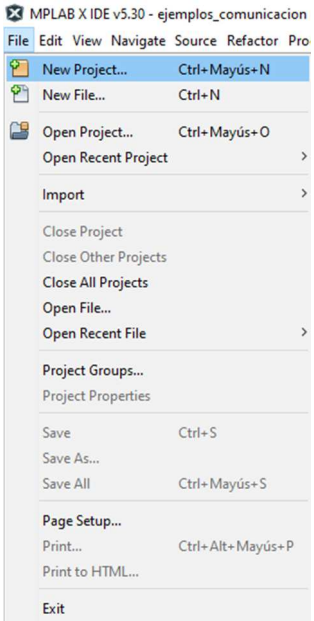


Figura N 1: Creación de un nuevo proyecto

Se abrirá una nueva ventana en la cual se seleccionar la 'Microchip Embedded' y 'Standalone Project' y clic en 'Next'.

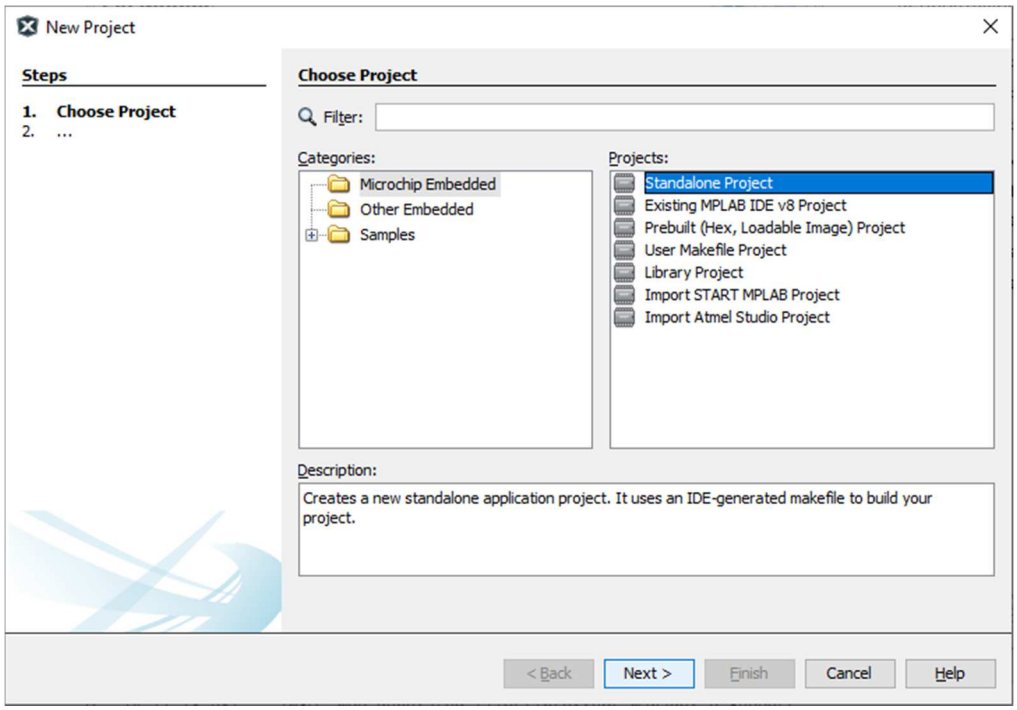


Figura N 2: Selección de un proyecto estándar

Se debe seleccionar la familia del MCU que se debe utilizar, como se puede ver en la figura y clic en Next.

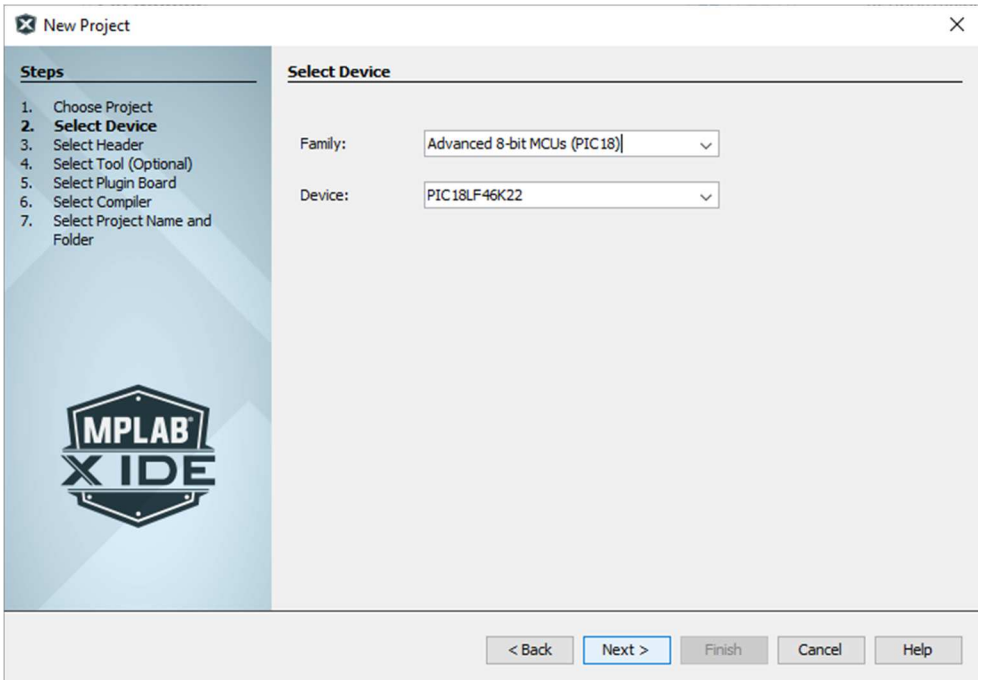


Figura N 3: Selección del microcontrolador a configurar

Ahora debe seleccionar la herramienta con la cual se va a grabar el RN2483 se selecciona el SNAP (en los laboratorios de la Universidad se tiene el SNAP)

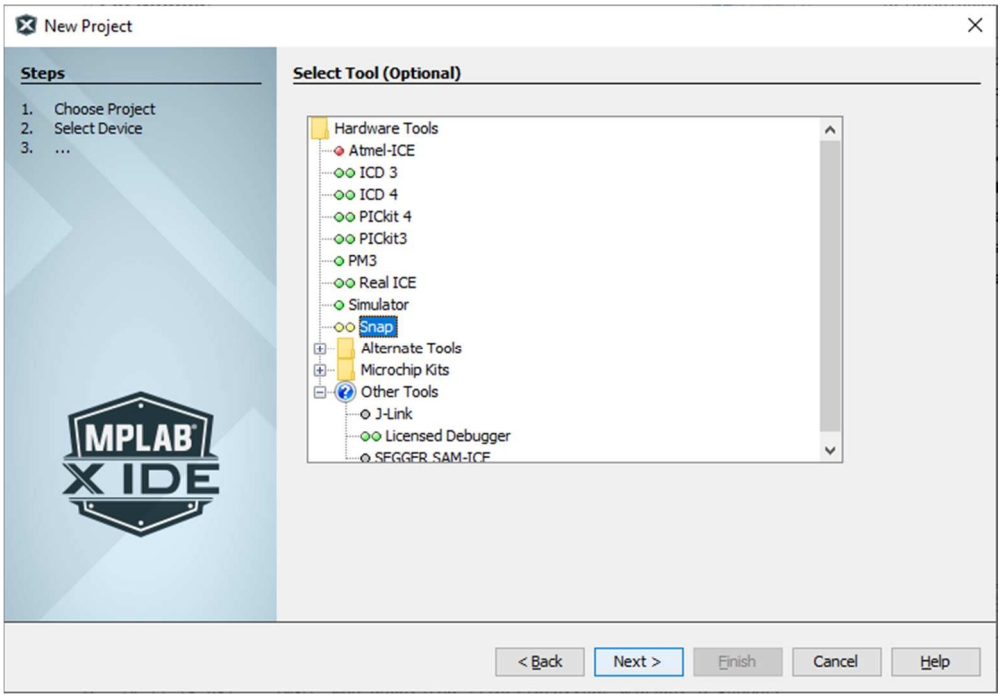


Figura N 4: Selección del programador

Seleccionamos el compilador XC8, por defecto viene el mpasm, en el caso de no tener el compilador indicado ir a la página de Microchip y descargar el compilador, clic en Next.

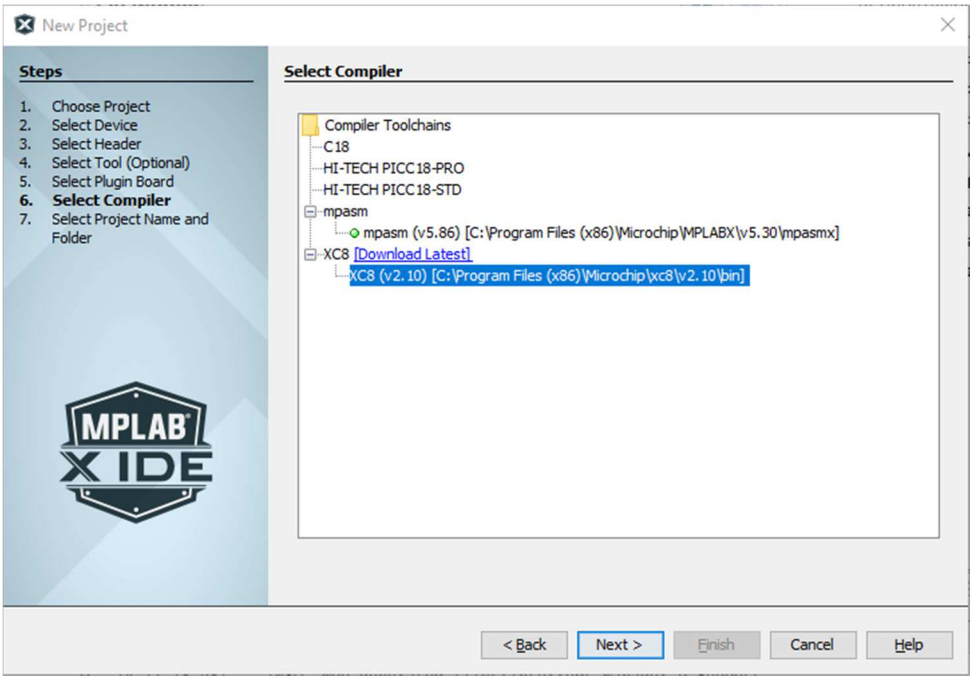


Figura N 5: Selección del tipo de compilador

En el último paso se debe colocar el nombre del proyecto, la ubicación donde se va a guardar el proyecto, y clic en ‘finish´

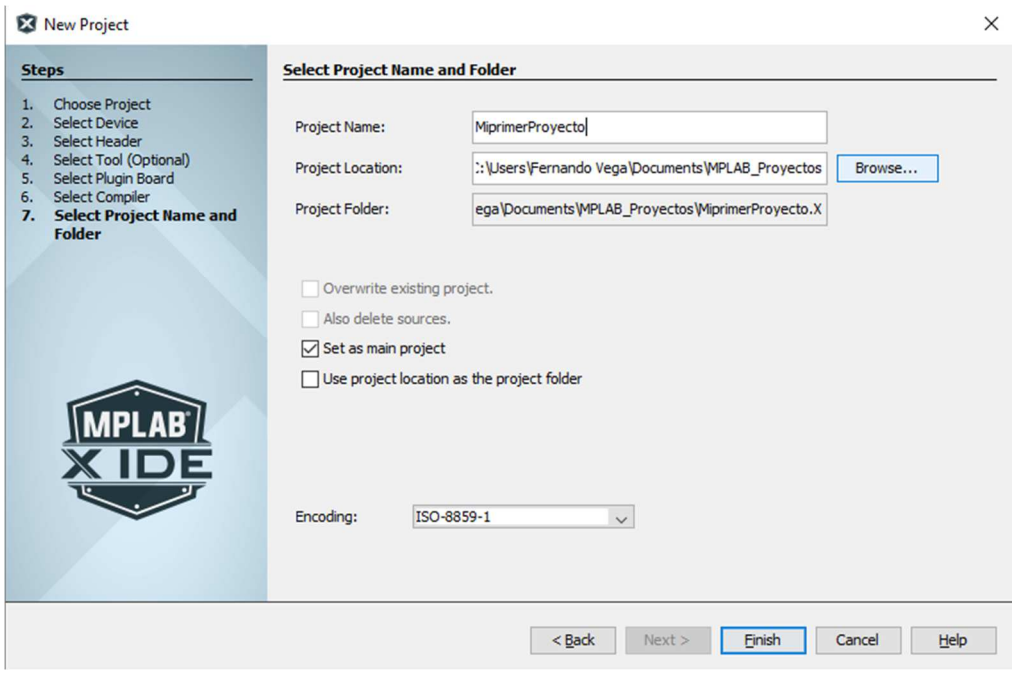


Figura N 6: Nombre del proyecto y su localización

APÉNDICE Ñ: INSTALACIÓN DE PLUGIN LORAWAN Y MCC

APÉNDICE Ñ.1: INSTALACIÓN DE PLUGIN LORAWAN

Para la configuración se debe descargar la librería de lorawan_v01.10.00_beta, y para instalar se debe abrir el MPLAB X IDE hacer clic en tools/options.

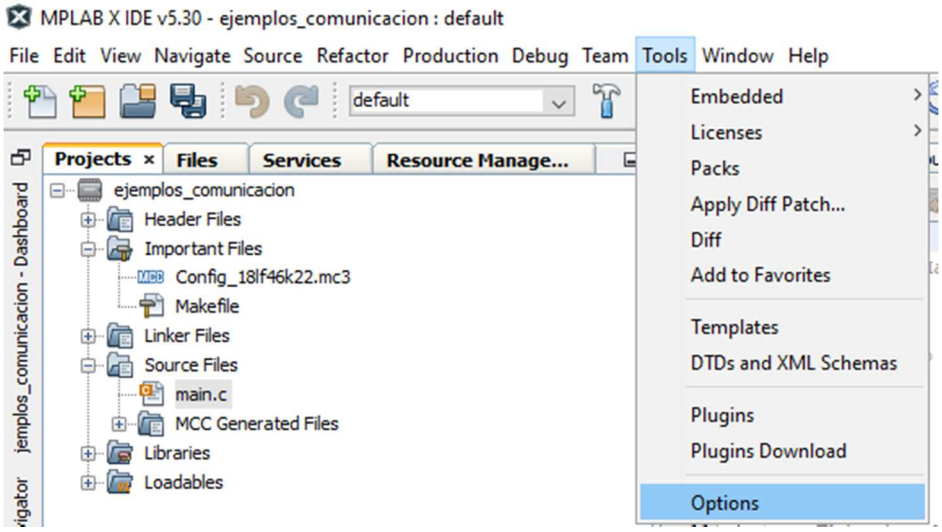


Figura Ñ 1: Ingreso a la pestaña Opciones

Dentro de "Options" dirigirse a Plugins/Install Library

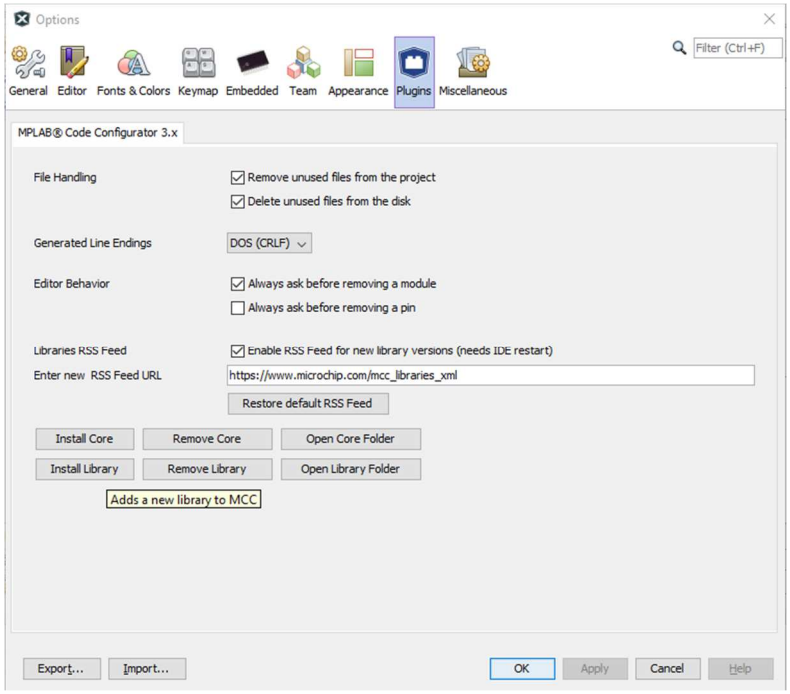


Figura Ñ 2: Configuración de la librería Lorawan

Buscar el archivo donde se encuentra la librería y dar clic en Abrir y clic en OK.

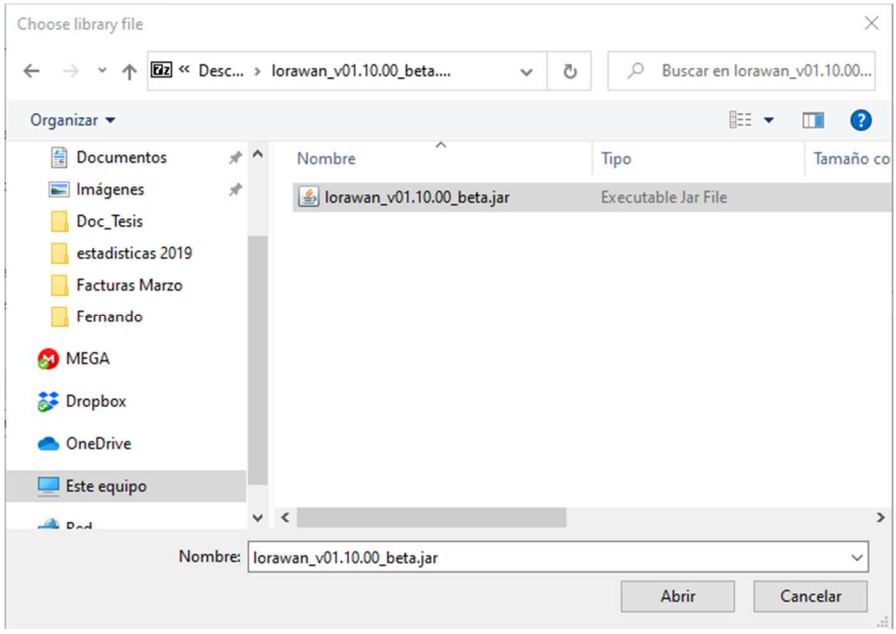


Figura Ñ 3: Selección e instalación de la librería Lorawan

APÉNDICE Ñ.2: INSTALACIÓN DE MCC

AL momento de programar la tarjeta LoRa MOTE, hay comandos que son muy difíciles de programar manualmente para ello se utiliza una herramienta llamada MCC, la cual se debe instalar, para ello se debe ir a Tools/Plugin.

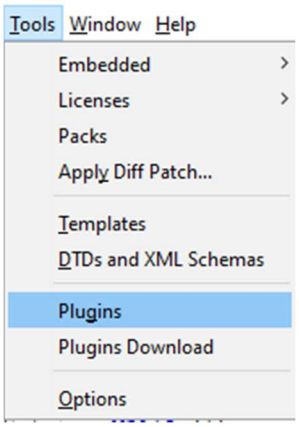


Figura Ñ 4: Ingreso al panel de Plugind

Seleccionar la pestaña Available Plunings y buscar Mplab Code Configurator he instalar.

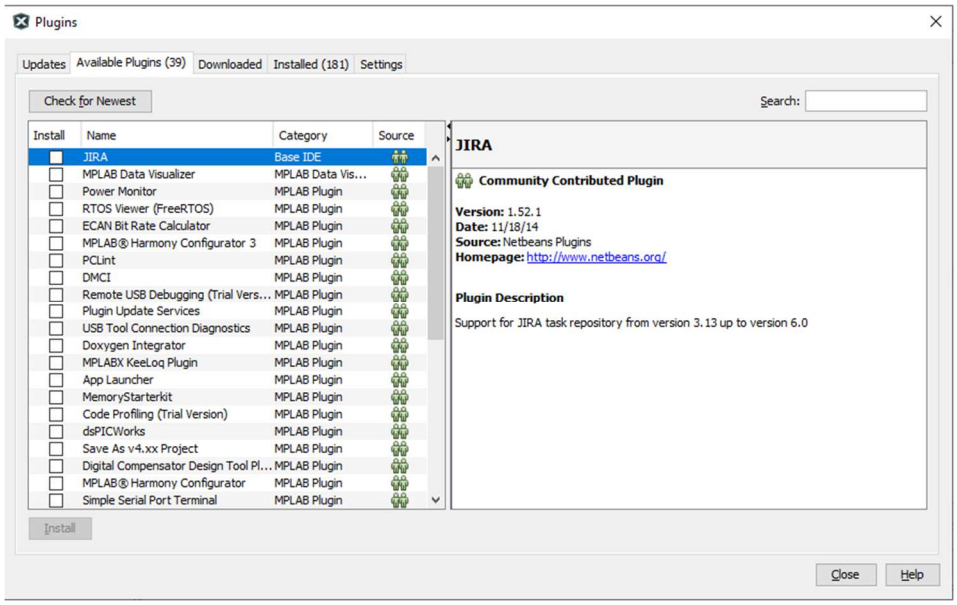


Figura Ñ 5: Instalación del plugin MCC

APÉNDICE O: DESARROLLO DE EJEMPLOS CON MCC

APÉNDICE O.1: CONFIGURACIÓN DE PIC18LF46K22 COMO SALIDA DIGITAL

Iniciar el Programa MPLAB IDE X, dirigirse a la pestaña de Windows/MPLAB Code Configurator, asignarle un nombre y guardarlo.

Figura O 1: Creación de un archivo MCC

Tener en cuenta que los módulos de interrupción, pines y sistema se abren automáticamente al iniciar el MCC

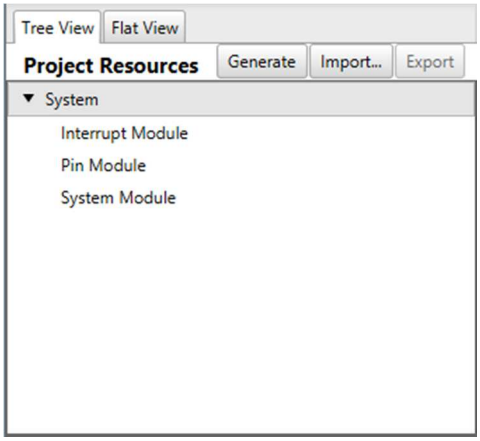


Figura O 2: Módulos del MCC

Abrimos el Pin Manager y seleccionamos los siguientes pines:

Tabla O 1: Nombre de los puertos GPIO

Costum Name	Puerto
GPIO 0	RA0
GPIO 1	RA1
GPIO 2	RA2
GPIO 3	RA3

GPIO 4	RA4
GPIO 5	RA5
GPIO 6	RE0
GPIO 7	RE1
GPIO 8	RD2
GPIO 9	RE2
GPIO 10	RC5
GPIO 11	RD5
GPIO 12	RD7
GPIO 13	RD6
USARTX	RC6
USARRX	RC7

Por tanto, debe aparecer de la siguiente manera el Pin Manager

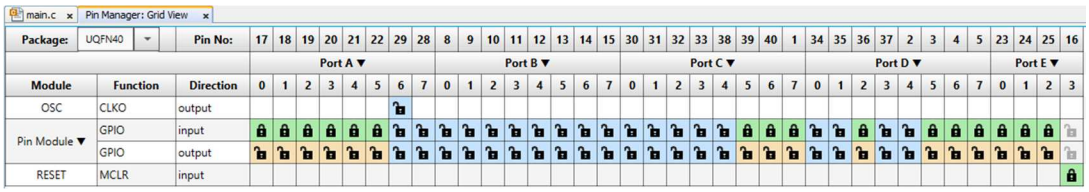


Figura O 3: Pin Manager configurado

Configurado el pin manager, se configura el módulo del sistema donde se escoge el oscilador interno, el sistema de reloj se escoge el 'FOSC', el reloj interno e n 16MHz.

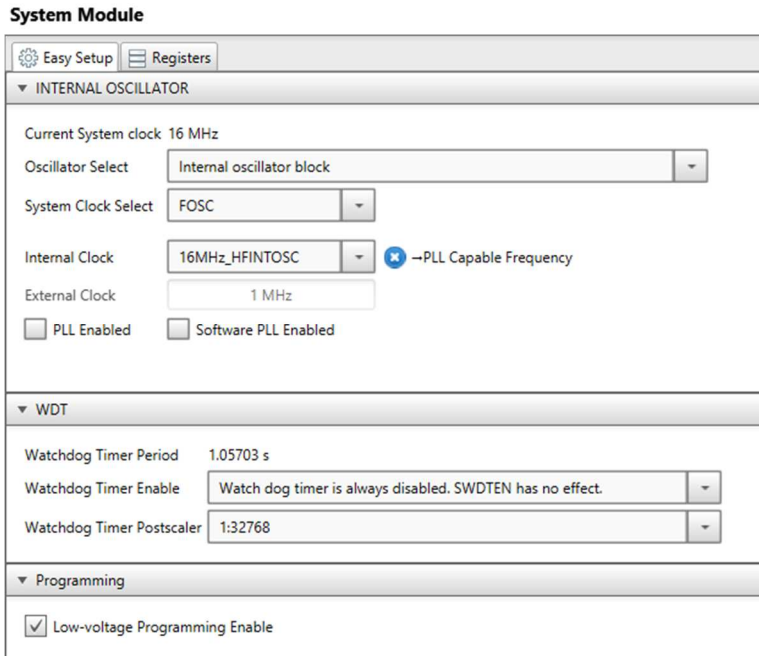


Figura O 4: Configuración del 'System Module'

Configurado todos los parámetros se debe iniciar con la generación de código para ello nos dirigimos a Resource Mangement/ Tree View/ Generate.

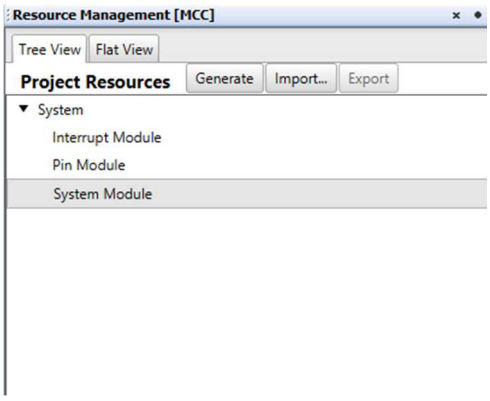


Figura O 5: Panel de Generación de código

Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Source Files), como se puede ver en la figura.

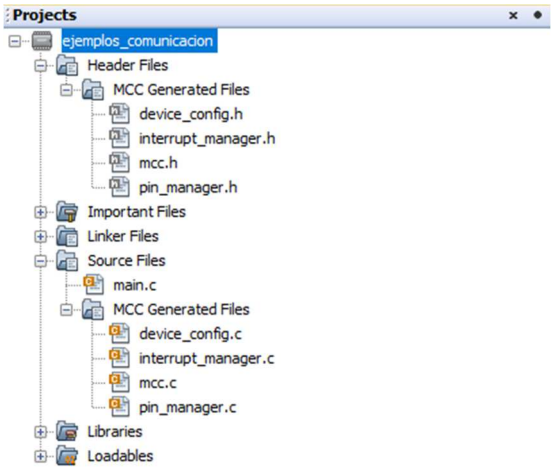


Figura O 6: Archivos Generados por MCC

Tomar en cuenta que los archivos '.h' son los APIs que Microchip realiza como ayuda al usuario, para la configuración de los puertos debemos ir al main.c y llamar a las funciones.

```
void main(void)
{// Initialize the device
   SYSTEM_Initialize();
   while (1) // cualquier código ingresado se repite indefinidamente
   {
      // Add your application code
      GPIO_0_SetDigitalMode(); // el puerto se configura como digital
      GPIO_0_SetDigitalOutput(); //el puerto se configura como salida

      GPIO_1_SetDigitalMode(); // el puerto se configura como digital
      GPIO_1_SetDigitalOutput(); //el puerto se configura como salida

      GPIO_2_SetDigitalMode(); // el puerto se configura como digital
      GPIO_2_SetDigitalOutput(); //el puerto se configura como salida

      GPIO_0_SetHigh(); // el puerto gpio_0 se activa en 1 lógico
      __delay_ms(500); // el estado se mantiene 500ms
      GPIO_0_SetLow(); // el puerto gpio_0 cambia a 0 lógico
      __delay_ms(400); // el estado se mantiene 400ms

      GPIO_1_SetHigh(); // el puerto gpio_0 se activa en 1 lógico
      __delay_ms(500); // el estado se mantiene 500ms
      GPIO_1_SetLow(); // el puerto gpio_0 cambia a 0 lógico
      __delay_ms(400); // el estado se mantiene 400ms

      GPIO_2_SetHigh(); // el puerto gpio_0 se activa en 1 lógico
      __delay_ms(500); // el estado se mantiene 500ms
      GPIO_2_SetLow(); // el puerto gpio_0 cambia a 0 lógico
      __delay_ms(400); // el estado se mantiene 400ms
   }
}
/**
```

End of File
*/

APÉNDICE O.2: CONFIGURACIÓN DE PIC18LF46K22 COMO ENTRADA DIGITAL

Iniciar el Programa MPLAB IDE X, dirigirse a la pestaña de Windows/MPLAB Code Configurator, asignarle un nombre y guardarlo.

Figura O 7: Creación de un archivo MCC

Tener en cuenta que los módulos de interrupción, pines y sistema se abren automáticamente al iniciar el MCC

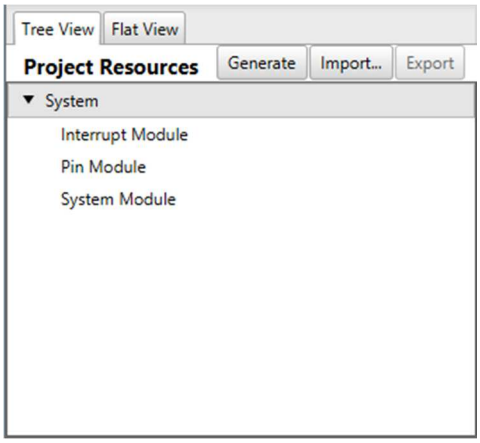


Figura O 8: Módulos del MCC

En este programa se modifica el pin GPIO_3 como entrada digital para ello se debe configurar en el pin manager como se muestra en el recuadro rojo.

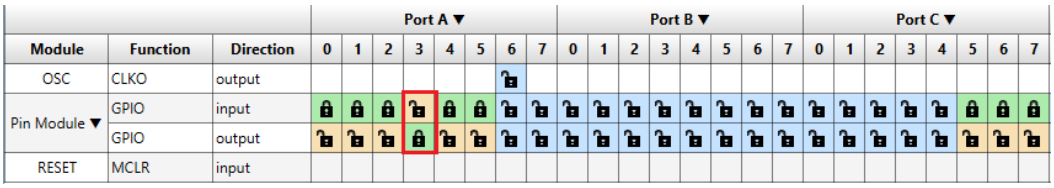


Figura O 9: Configuración del Pin Manager

Configurado el pin manager, se configura el módulo del sistema donde se escoge el oscilador interno, el sistema de reloj se escoge el 'FOSC', el reloj interno en 16MHz.

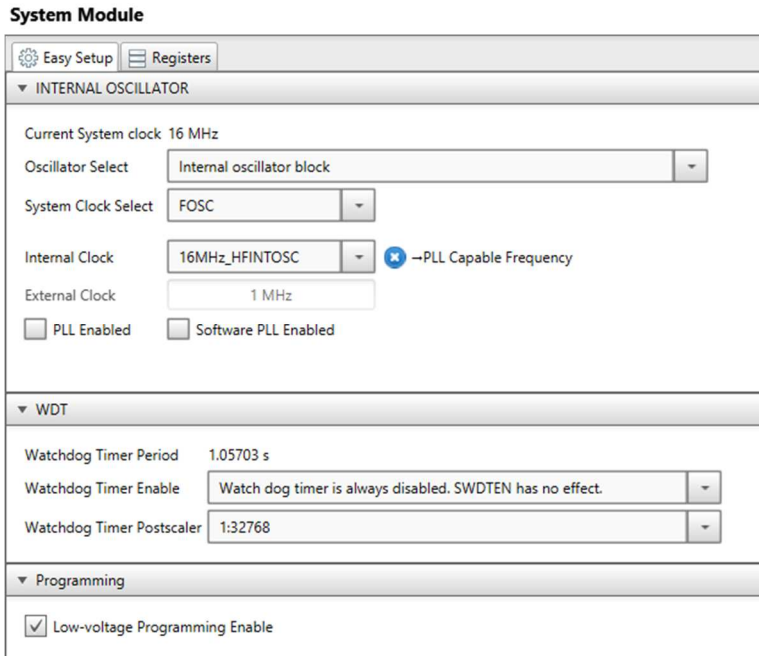


Figura O 10: Configuración del 'System Module'

Realizamos la generación del código y en el main.c debemos llamar las funciones correspondientes, aquí se describe el código del programa.

```
void main(void)
{// Initialize the device
   SYSTEM_Initialize();
   while (1) // cualquier código ingresado se repite indefinidamente
   {
      GPIO_0_SetDigitalMode(); // el puerto se configura como digital
      GPIO_0_SetDigitalOutput(); //el puerto se configura como salida

      if (GPIO_3_GetValue() == 0)
      {
         GPIO_0_SetHigh(); // el puerto gpio_0 se activa en 1 lógico
      }
      else {GPIO_0_SetLow(); // el puerto gpio_0 cambia a 0 lógico}
}
```

APÉNDICE O.3: CONFIGURACIÓN DE PIC18LF46K22 CON ENTRADA ANALÓGICA Y SALIDA DIGITAL

Para el desarrollo se debe primero tener en cuenta cuales son los pines de canal analógico, en este caso se tiene el GPIO_0 (RA0) y GPIO_1(RA1), para ello se debe incorporar el ADC y configurarlo, nos dirigimos a 'Resource Management/ Device Resources' y agregamos el periférico ADC y CCP5.

- Los registros para ADC son ADRESL y ADRESH

- Al usar el ADC se debe tener en cuenta: configuración del puerto, selección del canal, Selección del voltaje de referencia ADC, Fuente de reloj de conversión ADC, control de interrupción, formato de resultados.
- El registro ADCON2 permite al usuario seleccionar un tiempo de adquisición que ocurre cada vez que se establece el bit GO / DONE
- El tiempo de adquisición es establecido por el registro ADCON2
- El CCP5 es un Triger de evento especial, es utilizado en el ADC

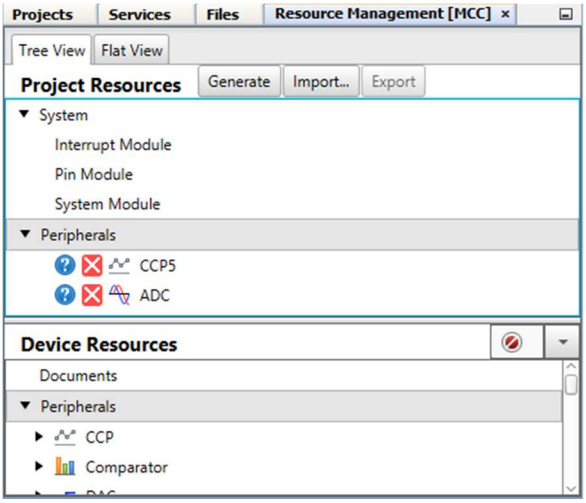


Figura O 11: Periféricos CCP5 y ADC agregados

Configuramos el puerto RA0 como canal analógico y el puerto RA4 como salida digital de forma que en el pin manager se presente de la siguiente forma

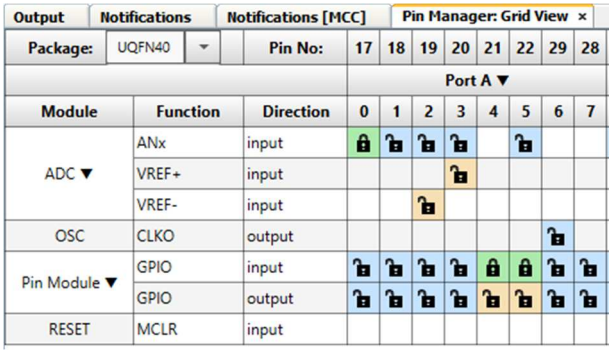


Figura O 12: Configuración de ADC en Pin Manager

Configuramos los datos del Clock Source, Acquisition Time, Result Aligment, positive Reference, Negative Reference y Auto-conversion Triger como se muestra en la figura.

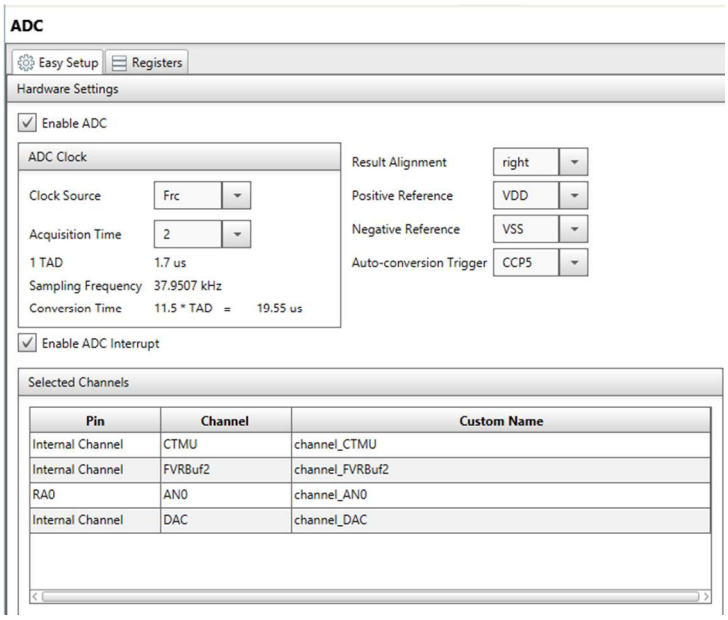


Figura O 13: Configuración de ADC

Configuramos el CCP5 debido a que es un requisito para el funcionamiento del ADC, con la diferencia que se configura de la siguiente manera

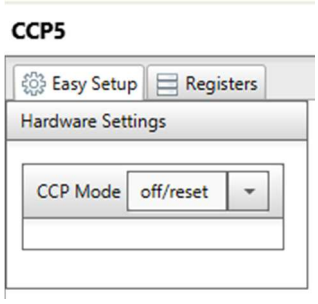


Figura O 14: Configuración del CCP5

Al terminar las configuraciones se debe generar los archivos de cabecera (Header Files) y los archivos fuente (Source Files).

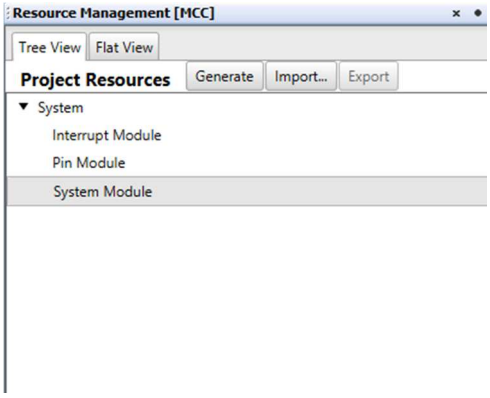


Figura O 15: Generación de Código con MCC

La programación de para una entrada analógica es la siguiente

```
void main(void)

{// Initialize the device

   SYSTEM_Initialize();

   while (1) // cualquier código ingresado se repite indefinidamente
   {
      IO_RA4_SetDigitalOutput(); //el puerto se configura como salida
      Channel_AN0_SetAnalogMode(); //el puerto RA0 se configura como analógico

     if (ADC_GetConversion(channel_AN0) < 512)
     {
        IO_RA4_SetHigh(); // el puerto gpio_4 se activa en 1 lógico
     }
     else { IO_RA4_SetLow(); // el puerto gpio_4 cambia a 0 lógico}
}
```

APÉNDICE O.4: CONFIGURACIÓN DE PIC18LF46K22 CON TMR0

Para realizar un nuevo proyecto se de abrir el MPLAB IDE X e ir a 'File/New Project', realizamos todos los pasos y ejecutamos el MCC y agregamos el TMR0.

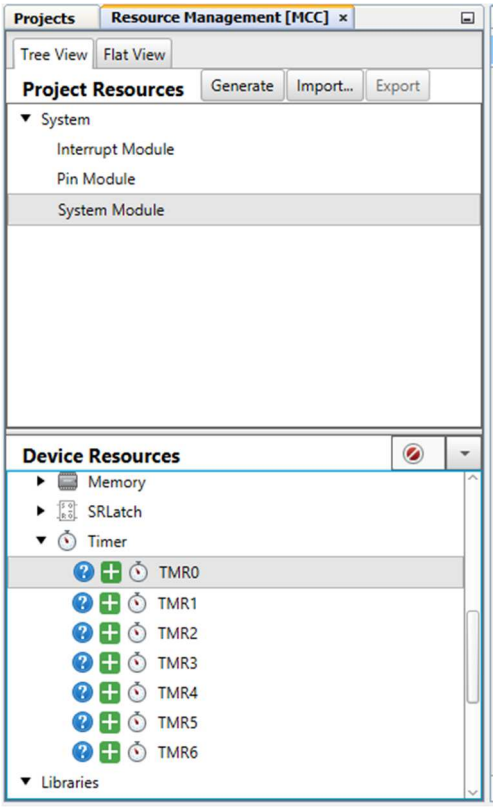


Figura O 16: Módulo TMR que posee el microcontrolador

Ahora configuramos el RA5 como salida digital por lo tanto en el pin manager debe quedar de la siguiente forma.

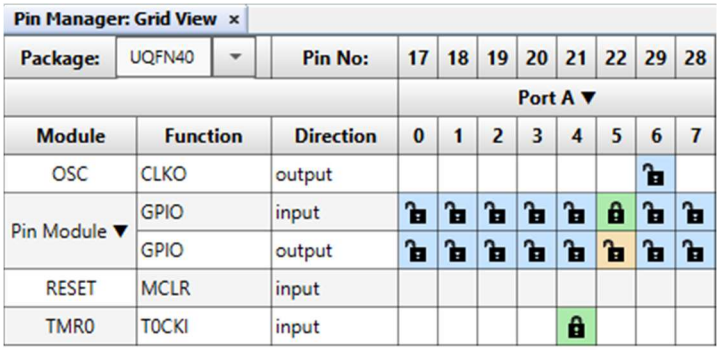


Figura O 17: Configuración del TMR0 en Pin Manager

Configurado el pin manager, se configura el módulo del sistema donde se escoge el oscilador interno, el sistema de reloj se escoge el 'FOSC', el reloj interno en 16MHz.

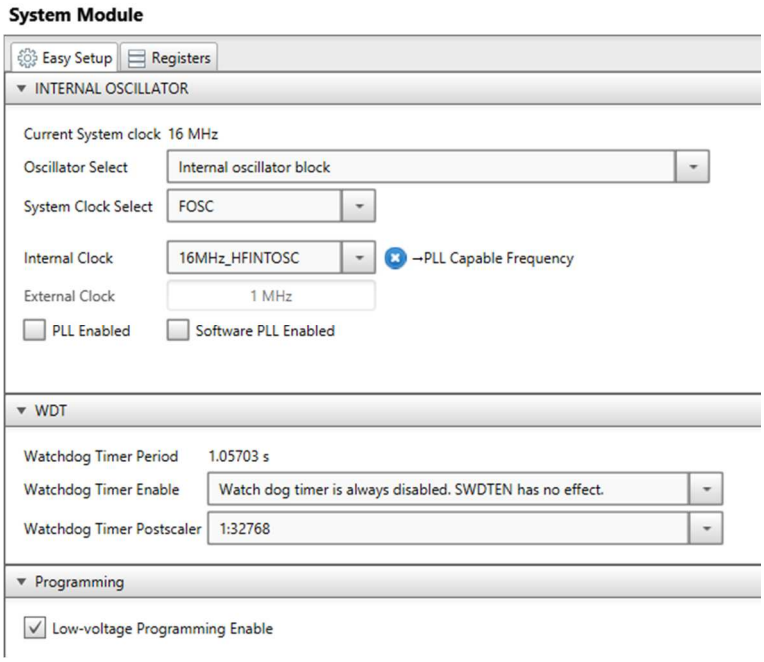


Figura O 18: Configuración del System Module

Nos dirigimos al TMR0 y lo configuramos de la siguiente manera

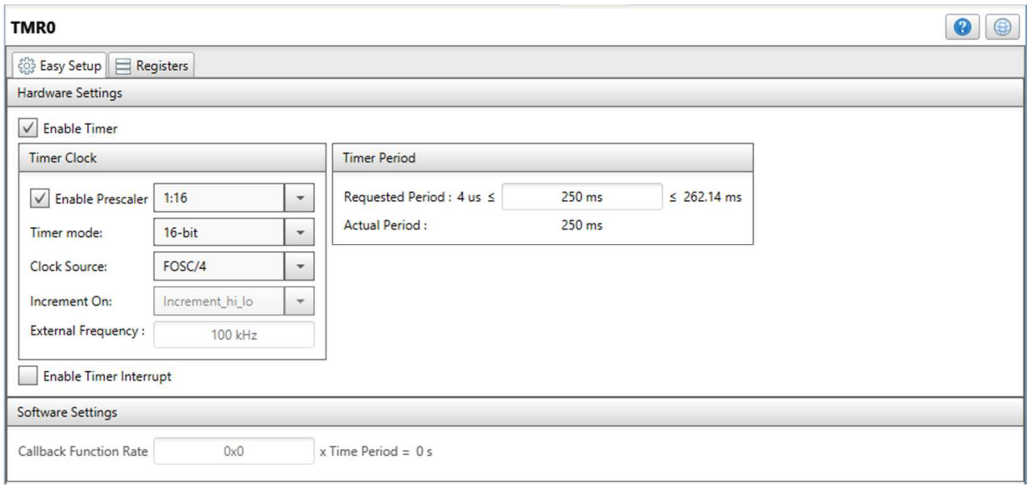


Figura O 19: Configuración del TMR0

Terminado la configuración se genera los documentos.

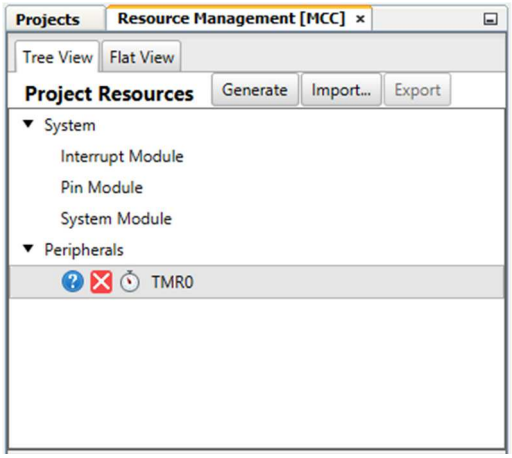


Figura O 20: Generación de código con MCC

EL Código de programación sería el siguiente

```
void main(void){// Initialize the device

   SYSTEM_Initialize();
   TMR0_Initialize();
   IO_RA5_SetDigitalMode();
   IO_RA5_SetDigitalOutput();

   while (1) // cualquier código ingresado se repite indefinidamente
   {
         If (TMR0_hasOverflowOccured()){
   IO_RA5_Toggle ();
            TMR0IF=0;}
   }
}
```

APÉNDICE P: CONFIGURACIÓN LORAWAN EN PIC18LF46K22

Crear un nuevo proyecto y dirigirse a 'Tools/Embedded/ MPLAB Code Configurator' para ejecutar MCC.

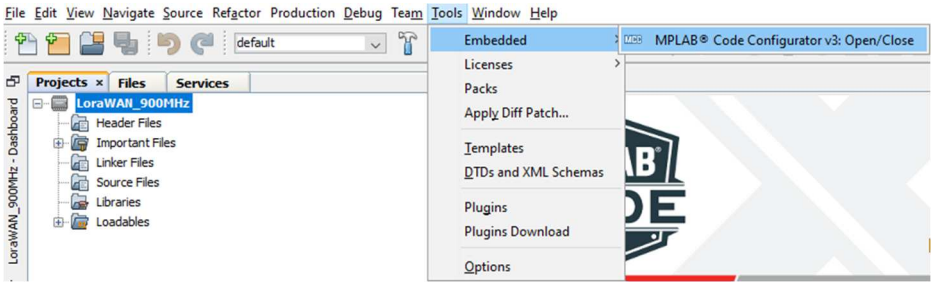


Figura P 1: Creación de un nuevo proyecto MCC

En la pestaña Resource Management/ Device Resource buscar la librería Lora y agregarla dando doble clic en el cuadro verde.

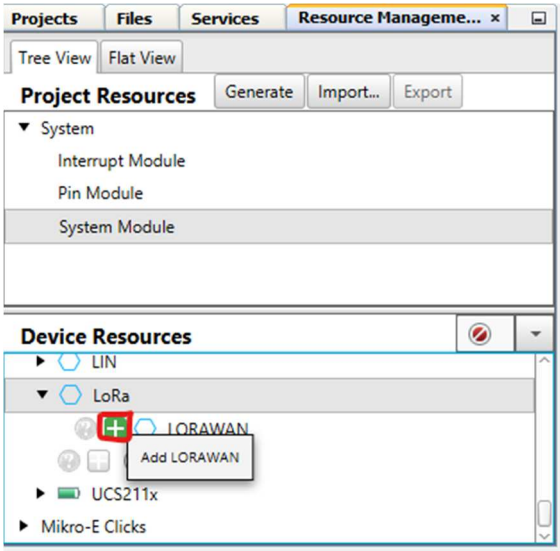


Figura P 2: Agrega la librería LORAWAN

En LORAWAN aparecen cinco parámetros que se deben configurar de la siguiente manera:

1. Base Timer: TMR1
2. Radio Module: SX1276
3. SPI Module: MSSP2
4. ISM Band: Europe 868
5. Default class: A

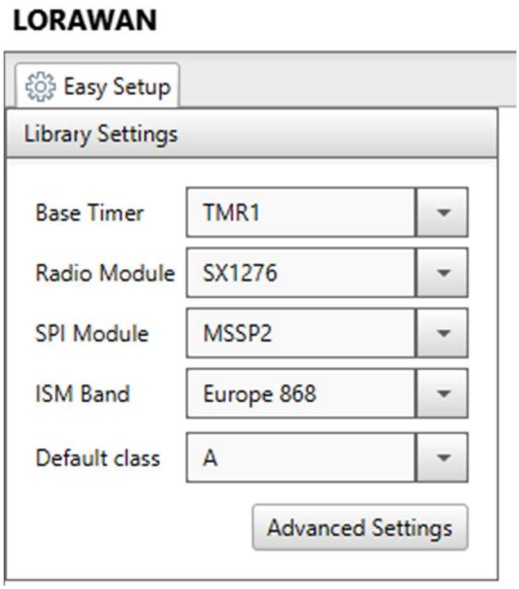


Figura P 3: Configuración de la librería LORAWAN

Una vez configurado la librería LORAWAN, aparecen dos notificaciones

- Configuring TMR1 for LoRaWAN to use a 2-second interrupt request period.
- Configuring MSSP2 for LoRaWAN in SPI Master mode.

Estas notificaciones se resuelven al momento de añadir y configurar las librerias EXT_INT, TMR1 y MSSP2, estos se encuentras en Device Resource.

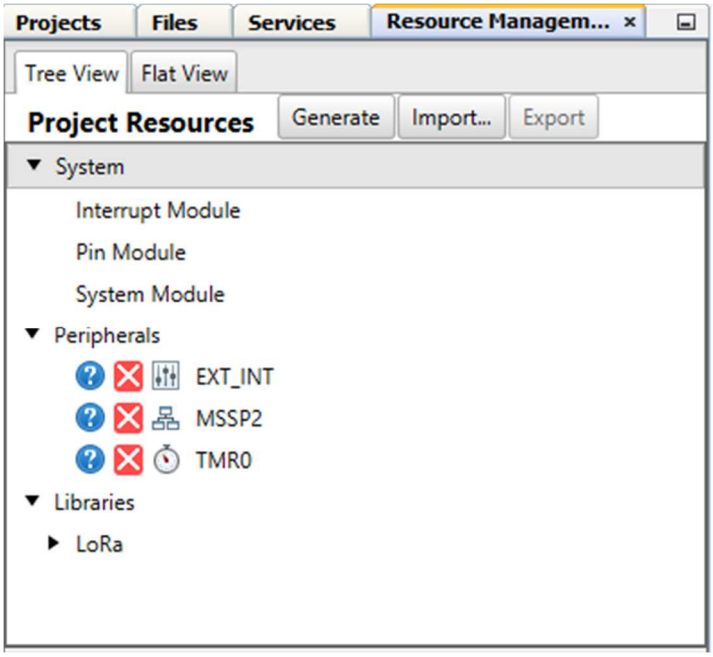


Figura P 4: Módulos necesarios para el funcionamiento de LoRaWAN

Configuración de System Module

En Project Resource/ System Module, configuramos los siguientes parámetros

- Oscillator Select: Internal oscillator block
- System Clock Select: FOSC
- Internal Clock: 16MHz_HFINTOSC/4
- Low-voltage programming Enable (checked)
- Watchdog Timer Enable: Watchdog timer is always disabled; SWDTEN has no effect
- Watchdog Timer Postscaler: 1: 32768

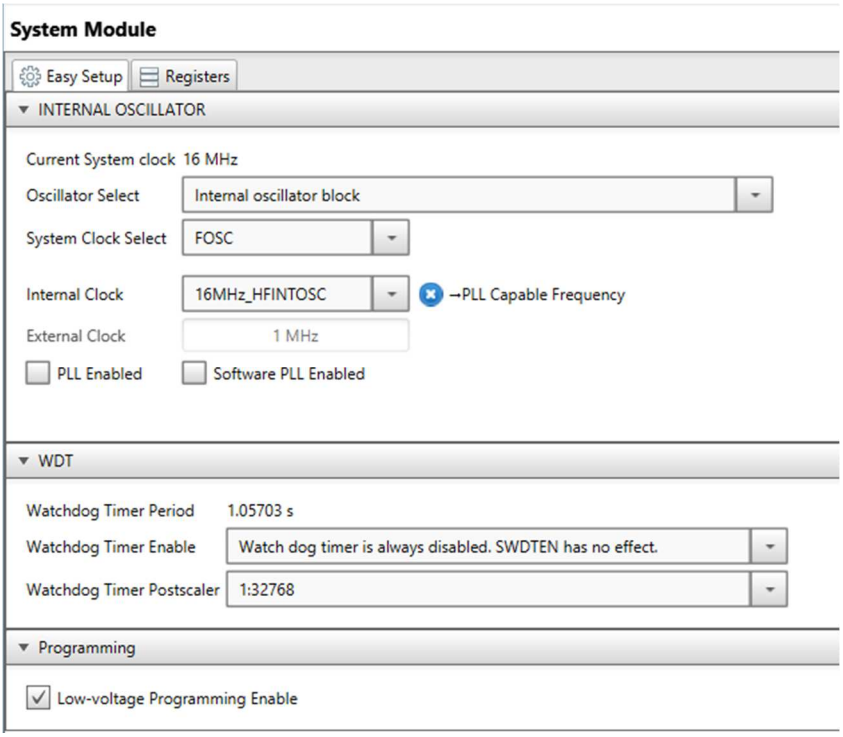


Figura P 5: Configuración del System Module

Configuración de TMR1

En Project Resource/ Pheripherals, configuramos los siguientes parámetros:

- Enable Timer: checked
- Timer Clock
  - Clock Source: External @ 32.768 kHz
  - Prescaler: 1:1
  - Enable Synchronization: Not checked
  - Enable Oscillator Circuit: Checked
- Timer Period
  - Timer Period: 2s
  - Period Count: (automatically filled by MCC)
  - Enable 16-bit read: not checked
- Enable Gate: not checked

- Enable Timer Interrupt: checked
- Software Settings
    - Callback Function Rate: 1

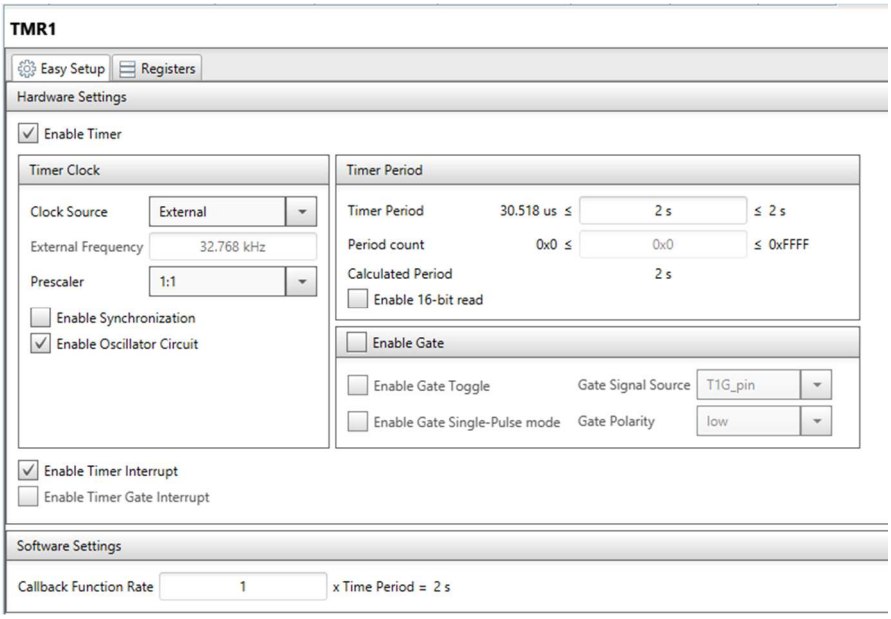


Figura P 6: Configuración del TMR1

Configuración de MSSP2

En Project Resource/ Pheripherals, configuramos los siguientes parámetros:

- Mode: SPI Master
- Enable MSSP: checked
- Input Data Sampled at: Middle
- Serial Protocol: SPI Mode
- SPI Mode: 0 (automatically filled by MCC)
- Clock Source: FOSC/4
    - SPI Clock: 4000.0 kHz (automatically filled by MCC)

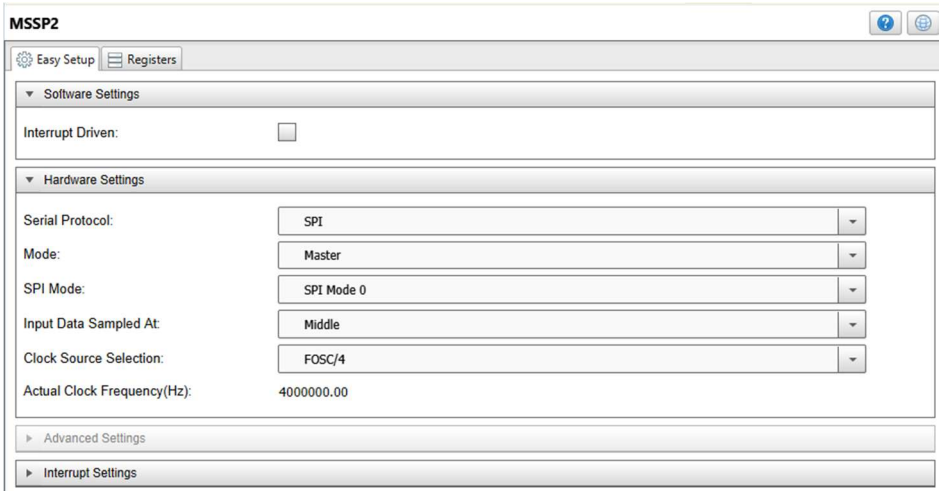


Figura P 7: Configuración del MSSP2

Configuración de Pin Manager

La comunicación del MCU con el radio transceiver se realiza configurando los GPIOs necesarios para la transferencia de datos, por lo tanto, los configuramos los siguientes pines.

- DIO0 – PORTB Pin 1 (RB1)
- DIO1 – PORTB Pin 2 (RB2)
- DIO2 – PORTB Pin 4 (RB4)
- DIO3 – not needed (Reserved for future use)
- DIO4 – not needed (Reserved for future use)
- DIO5 – PORTB Pin 0 (RB0)
- NRESET – PORTC Pin 2 (RC2)
- NSS – PORTD Pin 3 (RD3)
- SW_POW – PORTB Pin 3 (RB3). This pin is needed only for RN2903A module. For RN2483/RN2903, configuration for this pin is not needed.

En la ventana de MPLAB buscar la pestaña Pin Manager y realizar la configuración de los pines para la comunicación entre el MCU y el radio tranceiver, se configura los pines como se muestra en la figura.

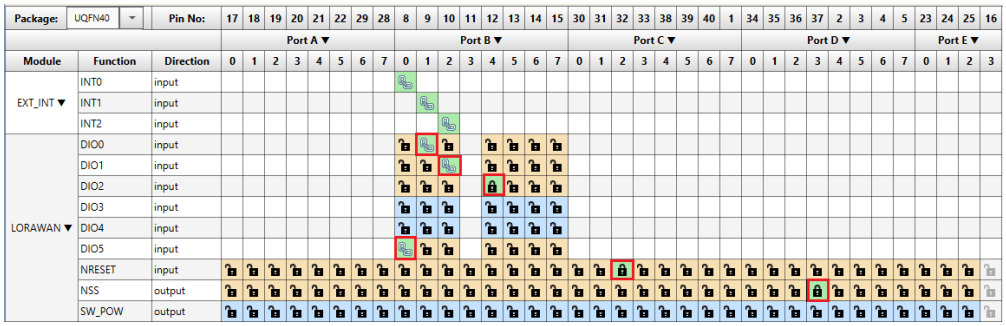


Figura P 8: Configuración del Pin Manager

Configuración de PIN Module

El Pin Module contiene todos los pines que se configuraron en Pin Manager, en este caso modificamos el IOC del RB4 a any.

Pin Module

Easy Setup | Registers

Selected Package : UQFN40

Pin Name	Module	Function	Custom Name	Start High	Analog	Output	WPU	OD	IOC
RB0	EXT_INT	INT0		☐	☐	☐	☑		
RB0	LORAWAN	DIO5		☐	☐	☐	☑		
RB1	EXT_INT	INT1		☐	☐	☐	☑		
RB1	LORAWAN	DIO0		☐	☐	☐	☑		
RB2	EXT_INT	INT2		☐	☐	☐	☑		
RB2	LORAWAN	DIO1		☐	☐	☐	☑		
RB4	LORAWAN	DIO2		☐	☐	☐	☑		any
RC0	INTERNAL OSCILLATOR	SOSCO		☐		☑			
RC1	INTERNAL OSCILLATOR	SOSCI		☐		☐			
RC2	LORAWAN	NRESET		☐	☐	☐			
RD1	MSSP2	SDI2		☐	☐	☐			
RD3	LORAWAN	NSS		☐	☐	☑			
RD4	MSSP2	SDO2		☐	☐	☑			

Figura P 9: Configuración del Pin Module

Configuración de Interrup Module

Deshabilitar la interrupción INT0. Debido a que el pin RB0, correspondiente a la interrupción INT0, solo es sondeado por la pila LoRaWAN siempre que sea necesario, no es necesario generar también una interrupción en INT0.

Haga clic en Módulo de interrupción y desactive la casilla de verificación EXT_INT - INT0I

Interrupt Module

Easy Setup

Interrupts

Please remember to enable the Peripheral and Global Interrupts in your code!

☐ Enable High/Low Interrupt Vector priority

Interrupt Vector

Order Up Down ☑ Single ISR per interrupt

Order	Module	Interrupt	Enabled
1	EXT_INT	INT0I	☐
2	EXT_INT	INT1I	☑
3	EXT_INT	INT2I	☑
4	MSSP2	BCLI	☐
5	MSSP2	SSPI	☐
6	TMR1	TMRI	☑
7	TMR1	TMRGI	☐
8	Pin Module	RBI	☑

Figura P 10: Configuración del Inerrupt Module

Para finalizar la configuración de LORA debemos generar los archivos

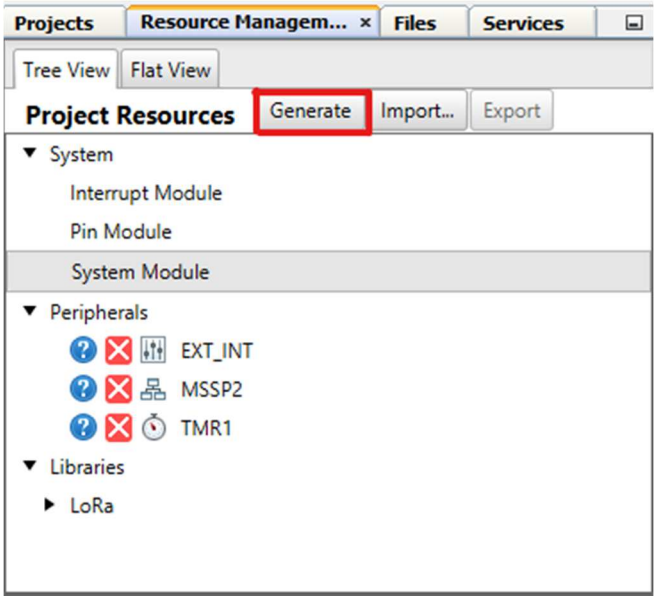


Figura P 11: Generación del código con MCC

Configuración de Encriptación AES

Microchip tiene una librería de encriptación para los MCU de 8-bits y 16-bits, la cual se puede encontrar en https:// www.microchip.com/ Developmenttools/ ProductDetails/ SW300052, para esta configuración específicamente necesitamos la carpeta 'Data Encryption Libraries V2.6'. en esta carpeta buscamos los archivos AES, que son AES.c, AES.h y AESdef.h y copiamos el contenido de estos en los archivos creados por el MCC.

En la pestaña Projects/Headers Files/MCC Generated Files/LoRaWAN vamos a encontrar los archivos que se deben modificar, para ello se abre AES.h y se pega el contenido.

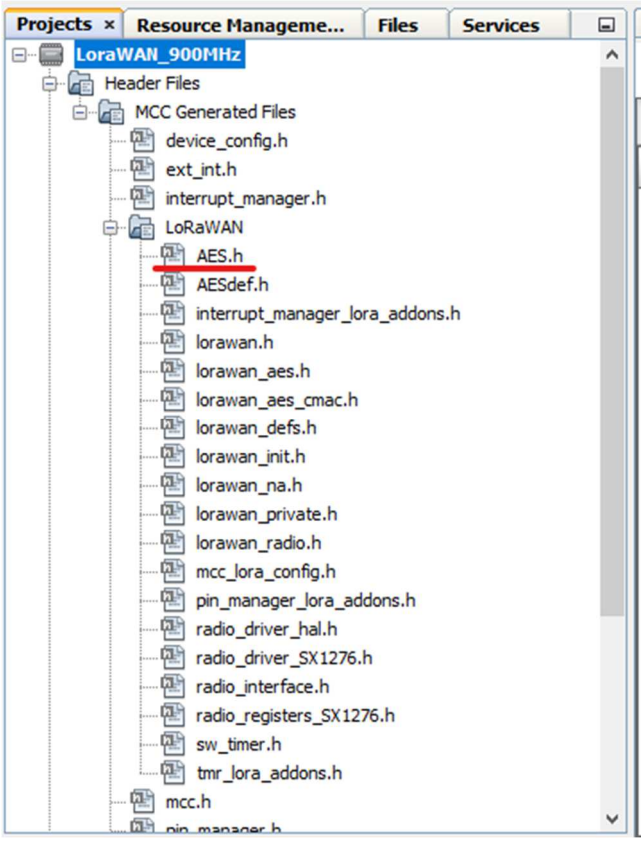


Figura P 12: Archivos Generados por MCC

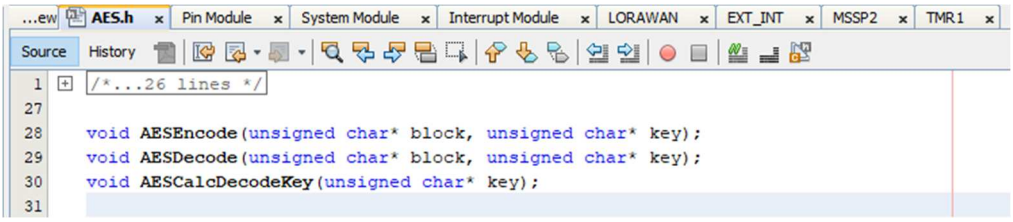


Figura P 13: Archivo cambiado para encriptación AES

En la pestaña Projects/Source Files/MCC Generated Files/LoRaWAN vamos a encontrar los archivos que se deben modificar, para ello se abre AES.c y se pega el contenido.

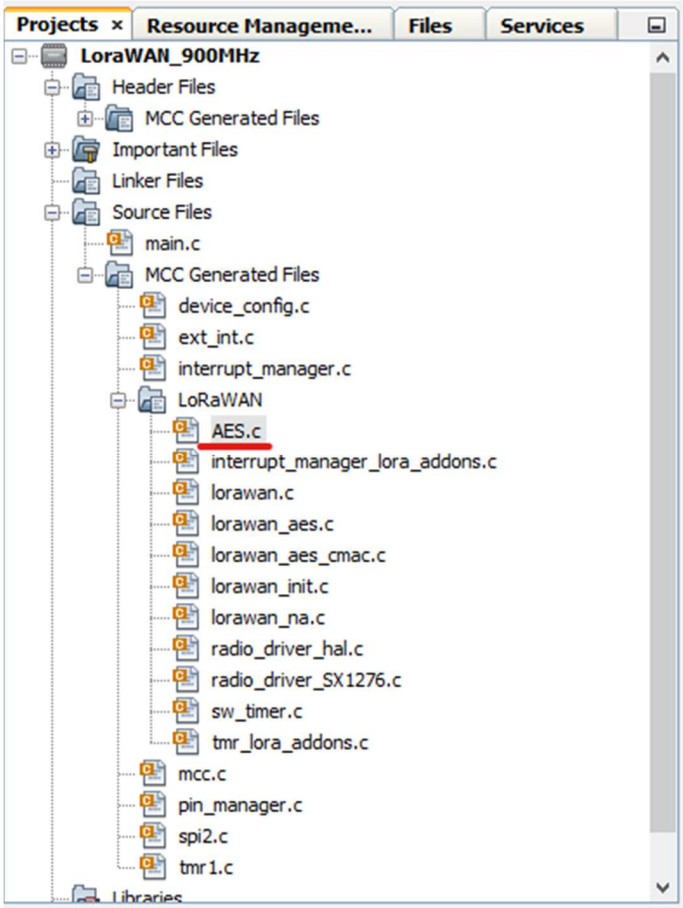


Figura P 14: Archivos Generados por MCC

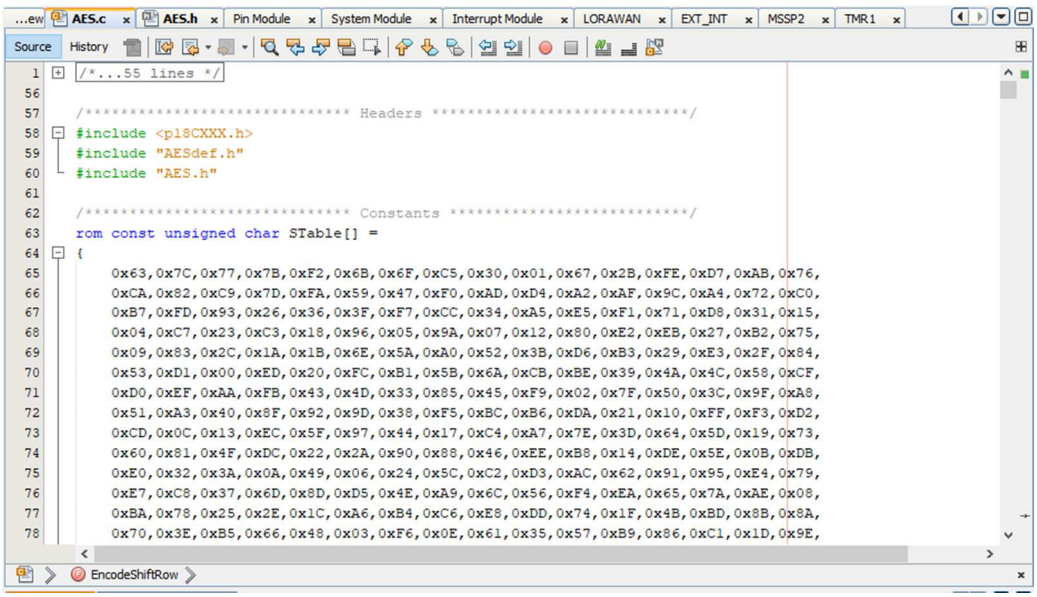


Figura P 15: Archivo cambiado para encriptación AES

Realizado el cambio en los archivos de encriptación debemos cambiar el Linker a C90, para ello se dirige a la pestaña Dashboard, Project Properties.

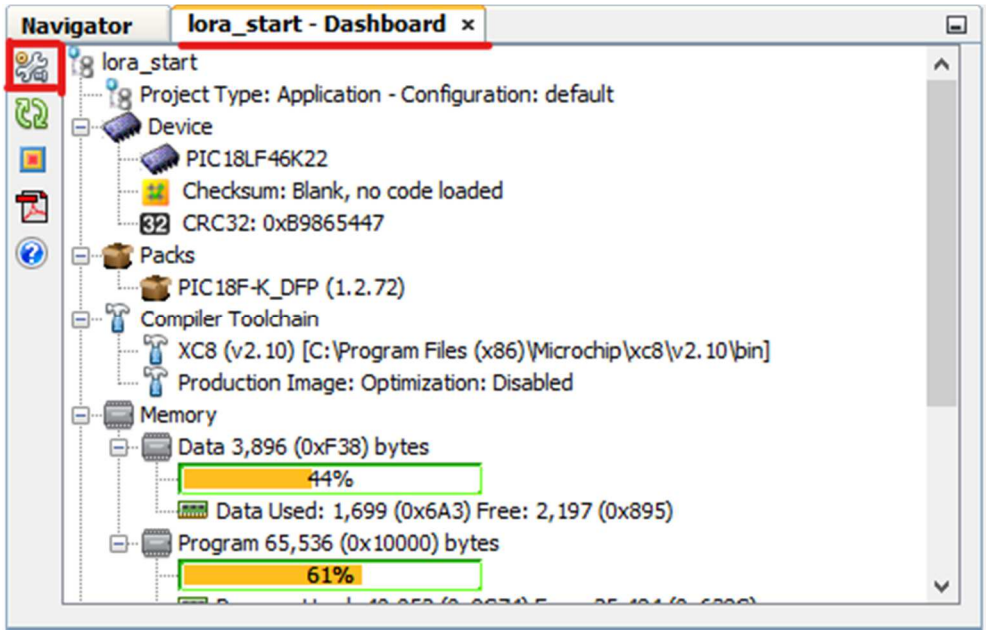


Figura P 16: Ventana de características del PIC

Ahora nos dirigimos 'XC8' Global compiler/ C standar' y cambiamos por C90, damos aplicar y Ok.

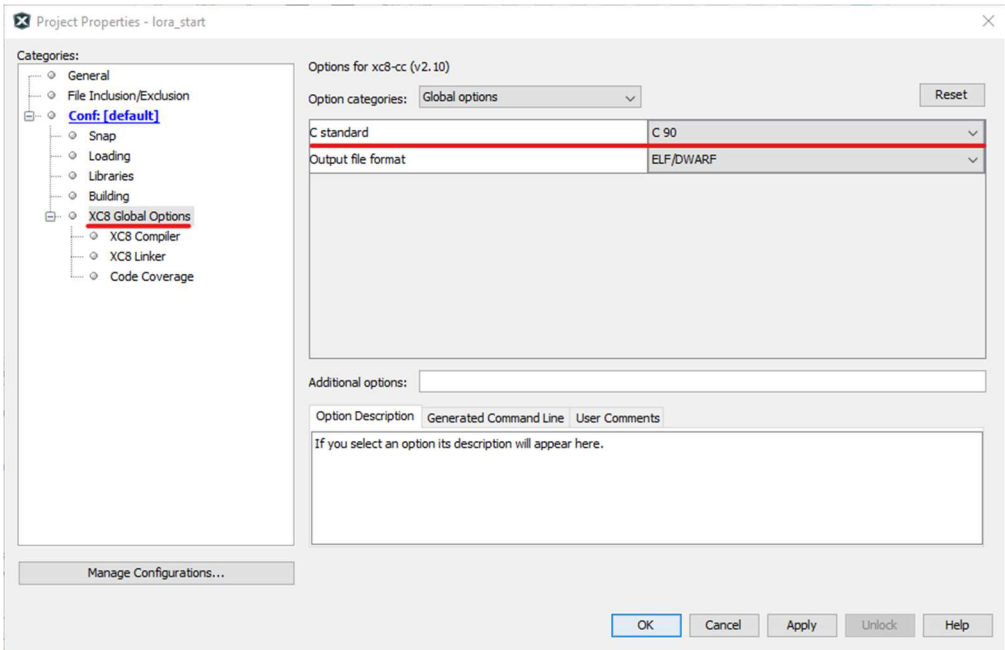


Figura P 17: Ventana de propiedades del proyecto, se cambia la versión del compilador

Para finalizar debemos comentar las líneas de código de las librerías lorawan_init.c y lorawan.c como se muestran en las siguientes imágenes y guardamos los cambias realizados.

```
67  void DIO5_ISR_Lora_Init(void)
68  {
69      //INT0_SetInterruptHandler(DIO5_ISR_Lora);
70  }
```

Figura P 18: Comentado de líneas de código

En lorawan.c se debe borrar 'reentrant'

```
reentrant void LORAWAN_ReceiveWindow2Callback(uint8_t param)
{
829     // Make sure the radio is not currently receiving (because a long packet is being recei
        if (loRa.macStatus.macPause == DISABLED)
        {
            if((RADIO_GetStatus() & RADIO_FLAG_RECEIVING) == 0)
            {
                loRa.macStatus.macState = RX2_OPEN;
835
836             RADIO_ReceiveStop();
837             RADIO_ReleaseData();
```

Figura P 19: Eliminación de código agregado por MCC

Insertamos el periférico EUSART1 para configurar una conexión UART con el Arduino nano.

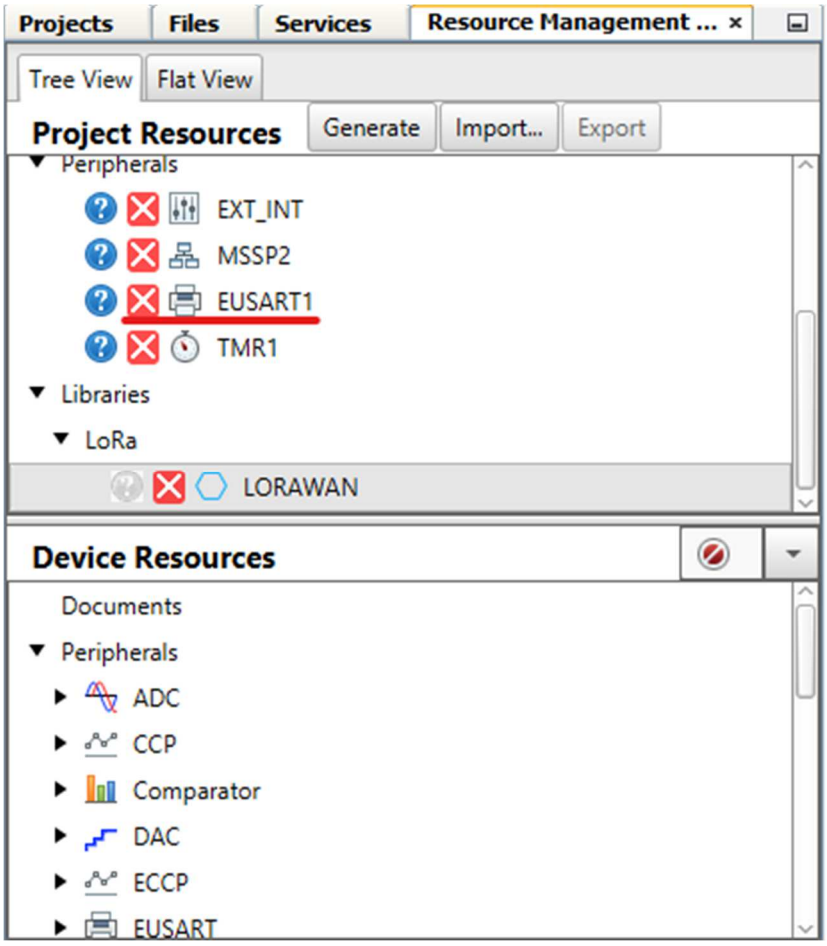


Figura P 20: Aumento del periférico EUSART1

Configuramos el periférico como se muestra en la figura, recordar que estos parámetros pueden variar dependiendo de la necesidad del usuario.

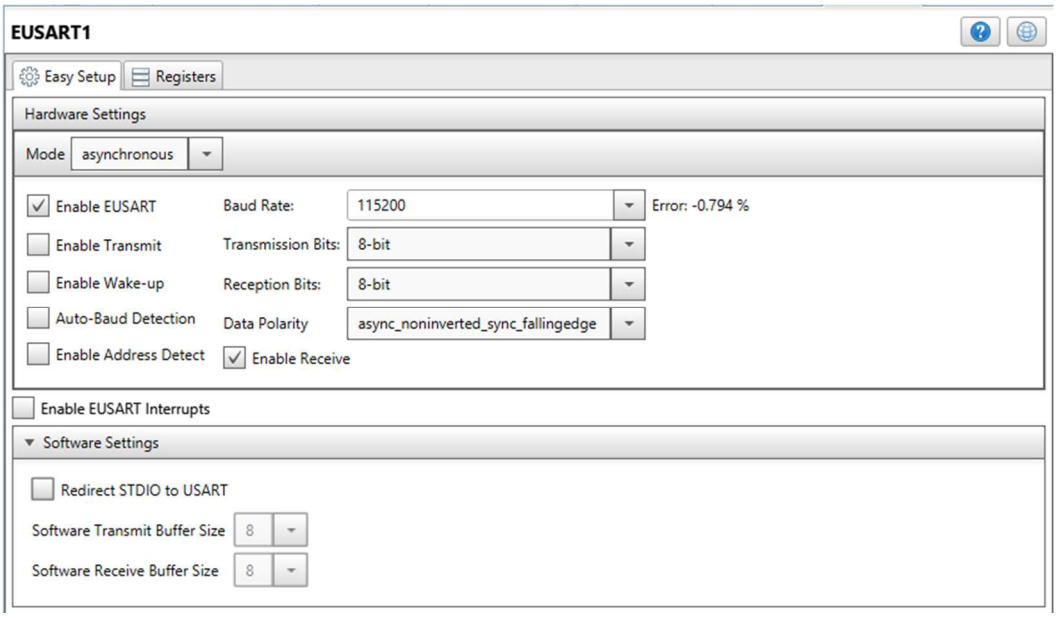


Figura P 21: Configuración de EUSART1

Configurado el EUSART1 se genera el código nuevamente y nos dirigimos a la pestaña 'main.c' y realizamos el algoritmo para la comunicación, como se describe a continuación.

```
#include "mcc_generated_files/mcc.h"
# define MAX_LENGTH_UART 16

/*
                    Main application
 */

uint8_t nwkSKey[16] = {0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB,
0xF7, 0x15, 0x88, 0x09, 0xCF, 0x4F,
0x3C};//0X2B7E151628AED2A6ABF7158809CF4F3C
uint8_t appSKey[16] = {0x3C, 0x8F, 0x26, 0x27, 0x39, 0xBF, 0xE3, 0xB7, 0xBC,
0x08, 0x26, 0x99, 0x1A, 0xD0, 0x50,
0x4D};//0X3C8F262739BFE3B7BC0826991AD0504D
uint32_t devAddr = 0x1100000F;

/*Funciones para el inicio de LORA*/
void RxData(uint8_t* pData, uint8_t dataLength, OpStatus_t status){}
void RxJoinResponse(bool status){}


uint8_t UARTReadString(char *buf, uint8_t max_length) {
   uint8_t i = 0;
   char tmp = 1;
```

```
    for (i=0 ; i<max_length-1 ; i++) {
        tmp = EUSART1_Read();
        // Stop reading if end of string is read
        if (tmp == '\0' || tmp == '\n' || tmp == '\r') {
            break;
        }
        buf[i] = tmp;
    }

    buf[i+1] = '\0';

    return i;
}

void main(void){
    // cargo variables
    char* readBuf[MAX_LENGTH_UART];
    char* tmpBuf[8];
    int nRead = 0;

    // Initialize the device
    SYSTEM_Initialize();

    // If using interrupts in PIC18 High/Low Priority Mode you need to enable the Global
High and Low Interrupts
    // If using interrupts in PIC Mid-Range Compatibility Mode you need to enable the Global
and Peripheral Interrupts
    // Use the following macros to:

    // Enable the Global Interrupts
    INTERRUPT_GlobalInterruptEnable();

    // Enable the Peripheral Interrupts
    INTERRUPT_PeripheralInterruptEnable();

    // Disable the Global Interrupts
    //INTERRUPT_GlobalInterruptDisable();

    // Disable the Peripheral Interrupts
    //INTERRUPT_PeripheralInterruptDisable();

    /*Inicio de lorawWAN*/

    LORAWAN_Init(RxData, RxJoinResponse);/*Esta función inicia la pila de LORAWAN
y el radio Transceiver */

    /*ASIGNACION DE LOS PARAMETROS DE CONFIGURACION DE LORA*/
    /* 1. llave de sesion
       2. llave de aplicación
       3. direcciòn del dispositivo */
    LORAWAN_SetNetworkSessionKey(nwkSKey);
    LORAWAN_SetApplicationSessionKey(appSKey);
    LORAWAN_SetDeviceAddress(devAddr);

    /*SELECCION DE MODO DE UTILIZACION*/
```

```
    LORAWAN_Join(ABP);
    /* led de aviso de funcionamiento*/
    GPIO_10_SetHigh();
    __delay_ms(400);
    GPIO_10_SetLow();
    __delay_ms(100);

    while (1){

      // Add your application code

      LORAWAN_Mainloop();
      if (EUSART1_is_rx_ready()){
        nRead = UARTReadString(readBuf, MAX_LENGTH_UART);
        //itoa(tmpBuf, nRead, 10);
        LORAWAN_Send (UNCNF, 120, readBuf, nRead);
        GPIO_10_SetHigh();
        __delay_ms(200);
        GPIO_10_SetLow();
        __delay_ms(100);
      }
    }
}
```

APÉNDICE Q: INICIO DEL SISTEMA S.A.D

Abrir el programa Visual Studio Code, abrimos la carpeta donde se encuentra los archivos del sistema SAD, haciendo clic en 'Open Folder'

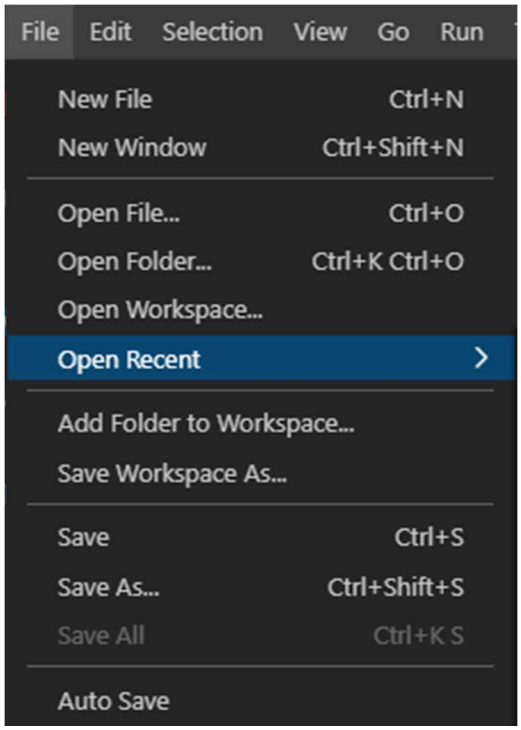


Figura Q 1: Abrir carpeta contenedora del Proyecto

Se debe dirigir al directorio donde está la carpeta principal para este caso en particular es 'C:\Users\Fernando Vega\Documents\NODE_JS\Tesis', seleccionamos tesis y damos clic en abrir.

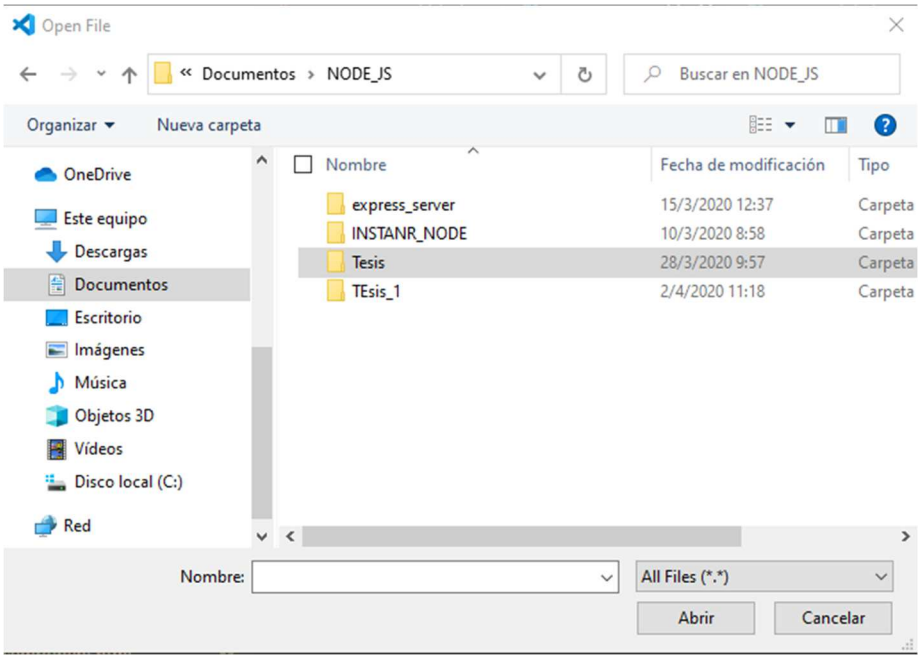


Figura Q 2: Selección de directorio del proyecto

Ahora abrimos dos ventanas CMD, puede presionar la tecla windows + r, escribir cmd

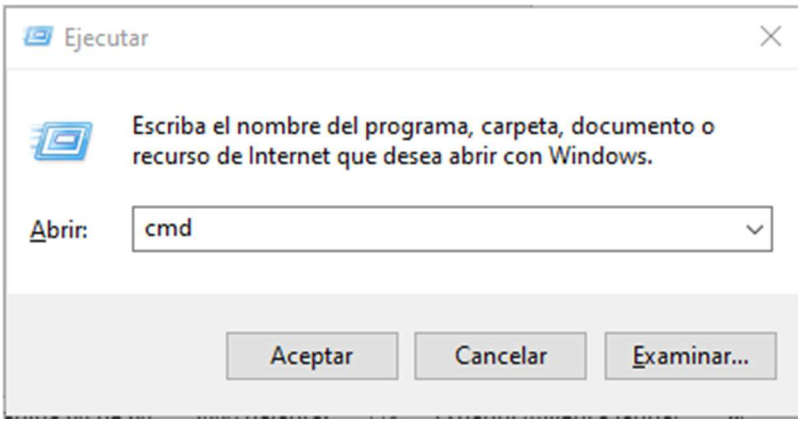


Figura Q 3: Comando para abrir símbolo del sistema

Iniciamos el sistema de Mongo DB para ello nos dirigimos a la carpeta donde se instaló el archivo que se presenta en la figura

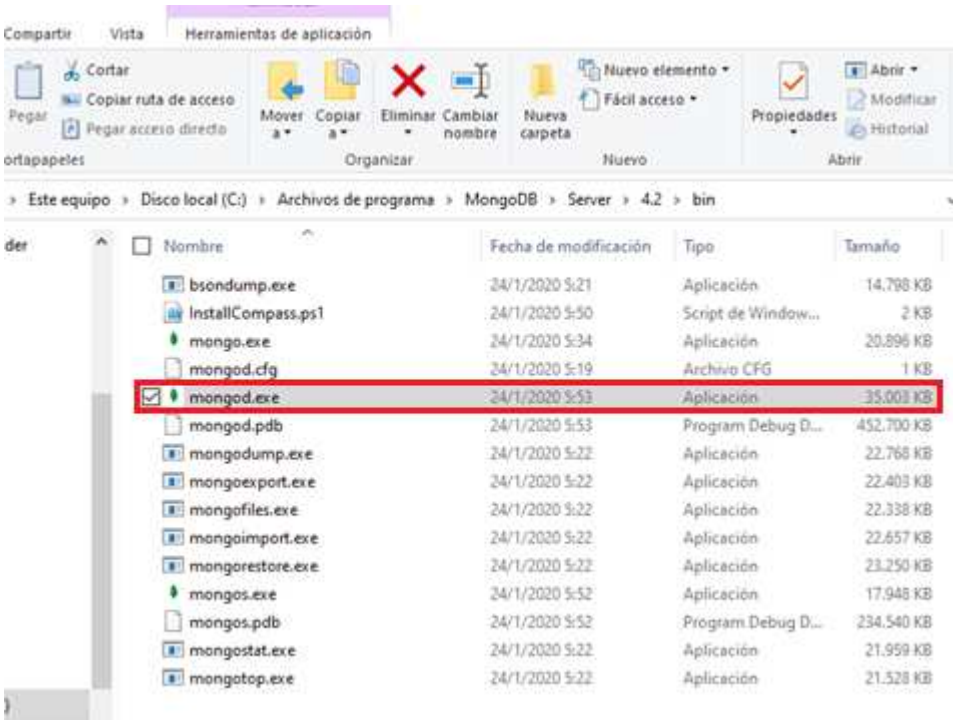


Figura Q 4: Inicio del servidor Mongo DB

Al dar doble clic se ejecutará el sistema de mongo db

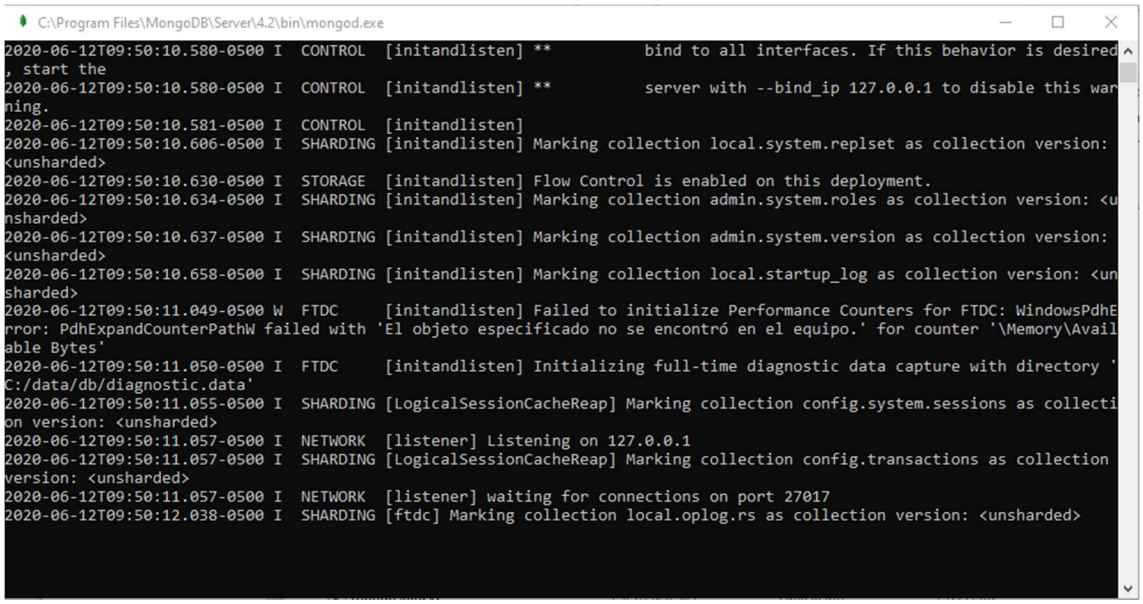


Figura Q 5: Ventana de ejecución de Mongo DB

Ahora para poner en marcha el backend se debe ir a la misma dirección, pero el directorio en ‘cmd’ se cambia a Backend

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Versión 10.0.18363.900]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Fernando Vega>cd C:\Users\Fernando Vega\Documents\NODE_JS\Tesis

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>dir
 El volumen de la unidad C no tiene etiqueta.
 El número de serie del volumen es: 205B-8025

 Directorio de C:\Users\Fernando Vega\Documents\NODE_JS\Tesis

28/03/2020  09:57    <DIR>          .
28/03/2020  09:57    <DIR>          ..
28/03/2020  09:59    <DIR>          AngularProject
14/05/2020  10:36    <DIR>          Backend
27/03/2020  21:22    <DIR>          Maquetacion
25/03/2020  20:30            53.294 tesis.png
               1 archivos         53.294 bytes
               5 dirs  181.876.477.952 bytes libres

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>cd Backend

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\Backend>
```

Figura Q 6: Cambio de directorio para ejecutar el BackEnd

Ahora para ejecutar se debe escribir ‘npm start’

```
npm

 Directorio de C:\Users\Fernando Vega\Documents\NODE_JS\Tesis

28/03/2020  09:57    <DIR>          .
28/03/2020  09:57    <DIR>          ..
28/03/2020  09:59    <DIR>          AngularProject
14/05/2020  10:36    <DIR>          Backend
27/03/2020  21:22    <DIR>          Maquetacion
25/03/2020  20:30            53.294 tesis.png
               1 archivos         53.294 bytes
               5 dirs  189.217.951.744 bytes libres

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>cd Backend

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\Backend>npm start

> tesis@1.0.0 start C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\Backend
> nodemon index_app_web.js

[nodemon] 2.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index_app_web.js`
(node:9712) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a f
uture version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the Mon
goClient constructor.
Servidor corriendo en http://localhost: 3900
server listening 0.0.0.0:1700
```

Figura Q 7: Ejecución del Backend

Debemos colocar la dirección donde está la carpeta Tesis y cambiar de directorio en la ventana de cmd

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Versión 10.0.18363.900]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Fernando Vega>cd C:\Users\Fernando Vega\Documents\NODE_JS\Tesis

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>dir
 El volumen de la unidad C no tiene etiqueta.
 El número de serie del volumen es: 205B-8025

 Directorio de C:\Users\Fernando Vega\Documents\NODE_JS\Tesis

28/03/2020  09:57    <DIR>          .
28/03/2020  09:57    <DIR>          ..
28/03/2020  09:59    <DIR>          AngularProject
14/05/2020  10:36    <DIR>          Backend
27/03/2020  21:22    <DIR>          Maquetacion
25/03/2020  20:30            53.294 tesis.png
               1 archivos         53.294 bytes
               5 dirs  181.876.494.336 bytes libres

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>cd AngularProject

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\AngularProject>
```

Figura Q 8: Cambio de directorio para ejecutar el Frontend

Ahora para ejecutar se debe escribir ‘ng serve’

```
ng serve

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>dir
 El volumen de la unidad C no tiene etiqueta.
 El número de serie del volumen es: 205B-8025

 Directorio de C:\Users\Fernando Vega\Documents\NODE_JS\Tesis

28/03/2020  09:57    <DIR>          .
28/03/2020  09:57    <DIR>          ..
28/03/2020  09:59    <DIR>          AngularProject
14/05/2020  10:36    <DIR>          Backend
27/03/2020  21:22    <DIR>          Maquetacion
25/03/2020  20:30            53.294 tesis.png
               1 archivos         53.294 bytes
               5 dirs  189.222.014.976 bytes libres

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>cd Angularproject

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\AngularProject>ng serve

chunk {main} main.js, main.js.map (main) 58.8 kB [initial] [rendered]
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 141 kB [initial] [rendered]
chunk {runtime} runtime.js, runtime.js.map (runtime) 6.15 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 35.1 kB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 3.41 MB [initial] [rendered]
Date: 2020-06-18T22:36:01.314Z - Hash: a255cf96de803bc06caa - Time: 13628ms
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
: Compiled successfully.
```

Figura Q 9: Ejecución del Frontend

Una ves que se ejecute MongoDB, Backend y Frontend se puede acceder a la interfaz colocando en el navegador ‘http://localhost/4200’, esto direcciona a la página principal.

Figura Q 10: Ventana principal o de inicio

En la estaña 'Sistemas' se encuentra una el panel de control para actualizar los datos que llegan al servidor MongoDB, como se puede ver en la figura

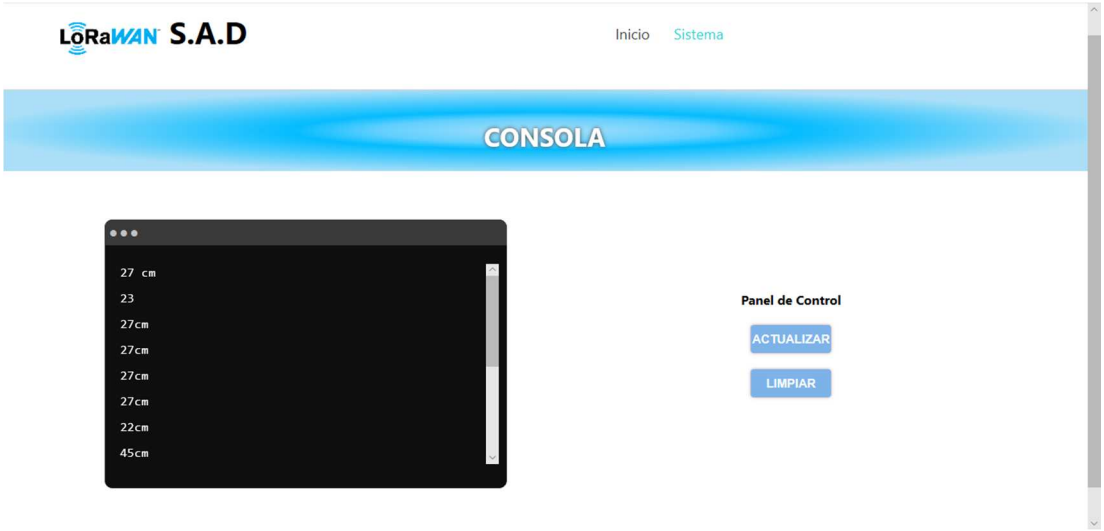


Figura Q 11: Ventana de visualización de datos enviados por los sensores

Para cancelar la ejecución de los servidores se debe teclear 'CTRL+c' y luego colocar 'S', este proceso se realiza para la Figura Q 7 y Figura Q 9.

```
C:\WINDOWS\system32\cmd.exe
C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>dir
 El volumen de la unidad C no tiene etiqueta.
 El número de serie del volumen es: 205B-8025

 Directorio de C:\Users\Fernando Vega\Documents\NODE_JS\Tesis

28/03/2020  09:57    <DIR>          .
28/03/2020  09:57    <DIR>          ..
28/03/2020  09:59    <DIR>          AngularProject
14/05/2020  10:36    <DIR>          Backend
27/03/2020  21:22    <DIR>          Maquetacion
25/03/2020  20:30            53.294 tesis.png
               1 archivos         53.294 bytes
               5 dirs  189.222.014.976 bytes libres

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis>cd Angularproject

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\AngularProject>ng serve

chunk {main} main.js, main.js.map (main) 58.8 kB [initial] [rendered]
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 141 kB [initial] [rendered]
chunk {runtime} runtime.js, runtime.js.map (runtime) 6.15 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 35.1 kB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 3.41 MB [initial] [rendered]
Date: 2020-06-18T22:36:01.314Z - Hash: a255cf96de803bc06caa - Time: 13628ms
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
: Compiled successfully.
^C¿Desea terminar el trabajo por lotes (S/N)? S

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\AngularProject>
```

Figura Q 12: Cierre de aplicación S.A.D

APÉNDICE R: IMPLEMENTACIÓN DE CONEXIÓN ABP

El ingreso de los dispositivos se debe realizar en el servidor, para que se puedan autentificar e identificar los datos de cada end-point, que se ingrese a la red.

Para ello se debe ingresar “Server/End-Device-Action/Provisioned ABP”

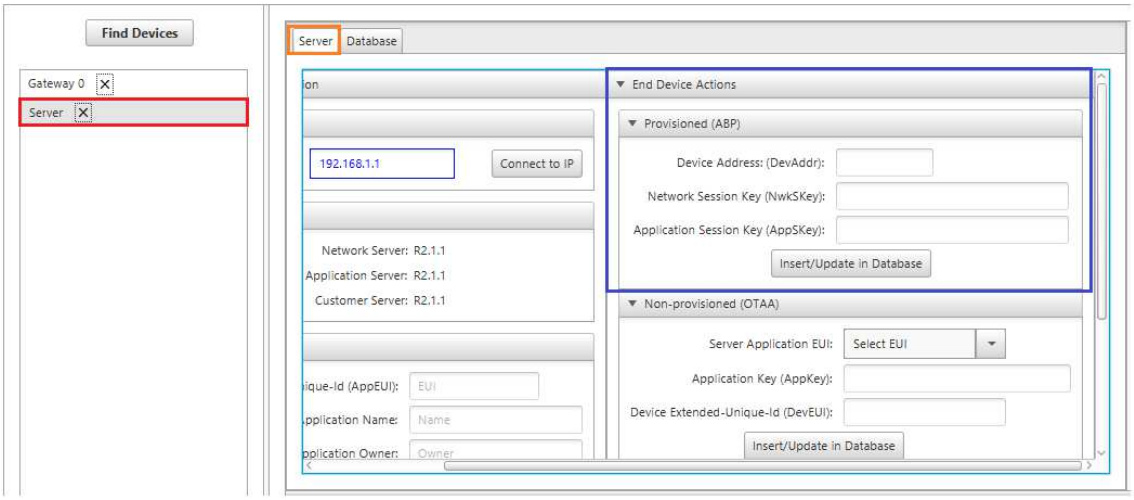


Figura R 1: Configuración de APB en LoRa Develoment

En los datos de ingreso en el cuadro de Provisioned (ABP) se debe ingresar lo siguiente:

Tabla R 1: Direcciones de los dispositivos LoRaWAN

Configuración	MOTE 1	MOTE 2
Device Address	0x42	0x52
Network Session Key	0x99	0x89
Application Session Key	0x87	0x77

Una vez configurado en el servidor los parámetros ABP de los MOTE´s se debe ingresar "RN Module/Server autentification Key" y configurar los mismos parámetros que en el servidor.

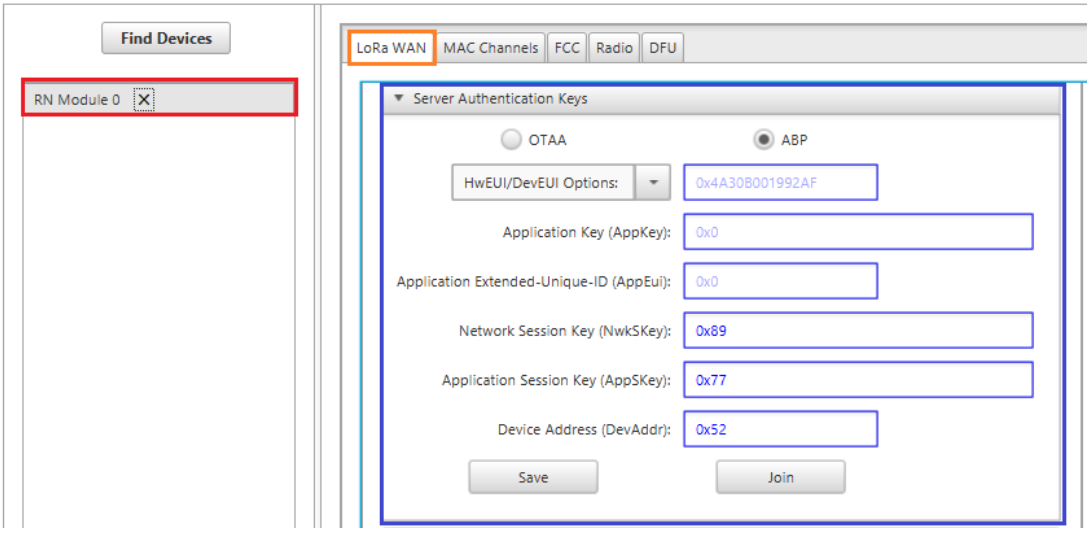


Figura R 2: Datos de dirección del MOTE 1

Después de ingresar se debe presionar Save y después Join, para su óptimo funcionamiento.

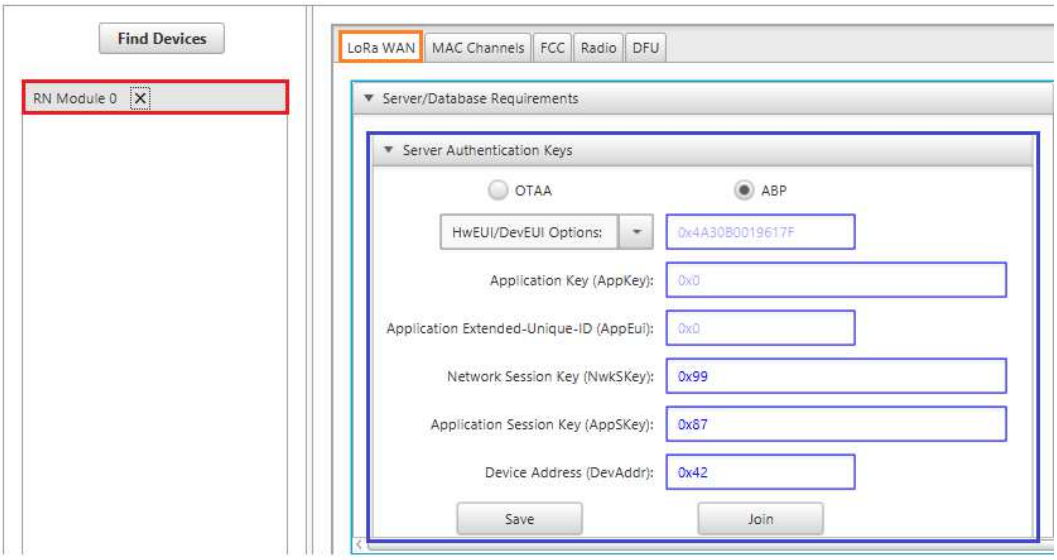


Figura R 3: Datos de dirección del MOTE 2

Para comprobar la autentificación de los MOTE´s se envía símbolos ASCII al servidor.

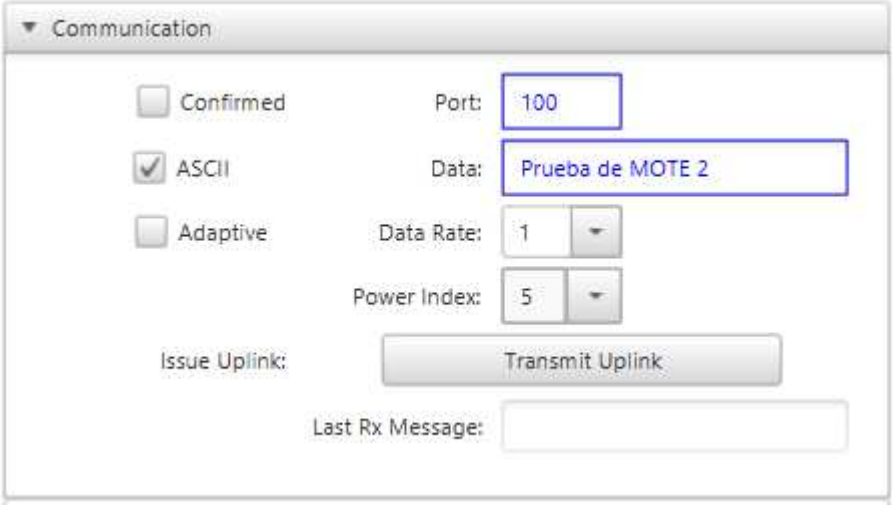


Figura R 4: Envió de dato ASCII del MOTE 1

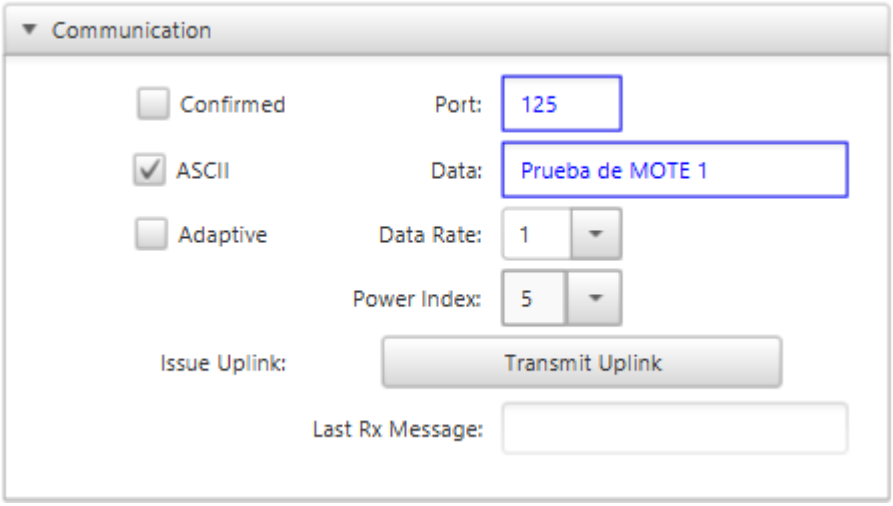
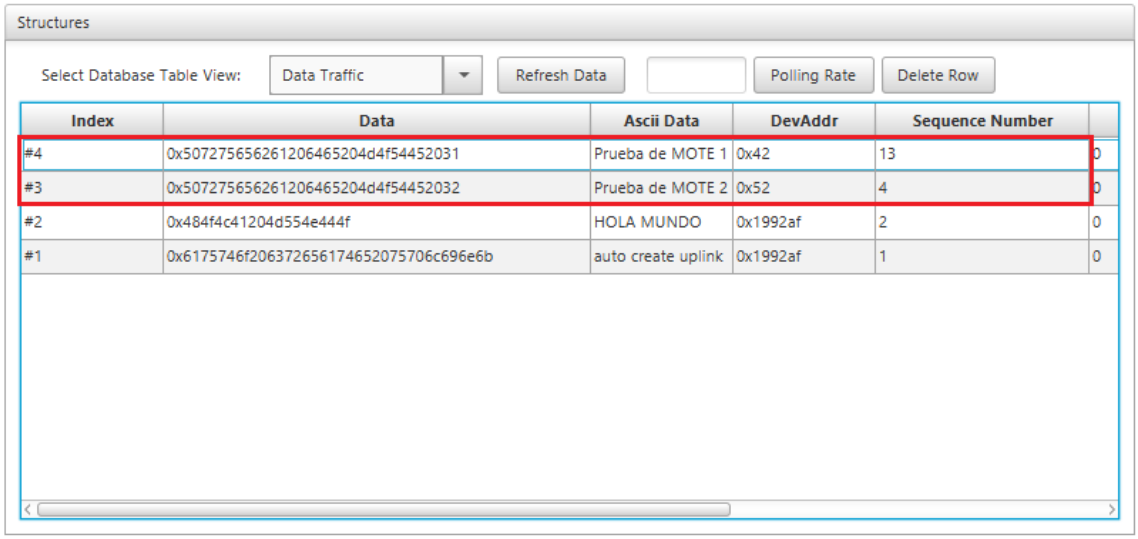


Figura R 5: Envió de dato ASCII del MOTE 2

Una vez que ya se han transmitido debe aparecer en el servidor los datos enviados



Index	Data	Ascii Data	DevAddr	Sequence Number	
#4	0x50727565626120646520 4d4f54452031	Prueba de MOTE 1	0x42	13	0
#3	0x50727565626120646520 4d4f54452032	Prueba de MOTE 2	0x52	4	0
#2	0x484f4c41204d554e444f	HOLA MUNDO	0x1992af	2	0
#1	0x6175746f206372656174652075706c696e6b	auto create uplink	0x1992af	1	0

Figura R 6: Verificación de datos enviados al servidor

APÉNDICE S: ENVÍO DE DATOS MEDIANTE SOCKETS A SERVIDOR DE RED

Iniciada la base de datos, el backend y frontend, el sistema comienza en la escucha del puerto 1700 para la captura de paquetes que proviene del Gateway LoRa, como se puede ver **¡Error! No se encuentra el origen de la referencia.**, el puerto 3900 se encarga de las peticiones del frontend y de la base de datos.

```
npm
(node:9664) [DEP0018] DeprecationWarning: Unhandled promise rejections are deprecated. In the future, promise rejections
 that are not handled will terminate the Node.js process with a non-zero exit code.
[nodemon] clean exit - waiting for changes before restart
¿Desea terminar el trabajo por lotes (S/N)? S

C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\Backend>npm start

> tesis@1.0.0 start C:\Users\Fernando Vega\Documents\NODE_JS\Tesis\Backend
> nodemon index_app_web.js

[nodemon] 2.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index_app_web.js`
(node:17764) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a
future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the Mo
ngoClient constructor.
Servidor corriendo en http://localhost: 3900
server listening 0.0.0.0:1700
```

Figura S 1: Inicio de Backend del sistema S.A.D

Para verificar que el backend y la base de datos están en funcionamiento, se debe hacer peticiones ‘get’, ‘post’ desde el software ‘Postman’, en el caso de una petición ‘get’ en el puerto 3900, que pertenece al backend, este se comunica con la base de datos para que le entregue todos los datos guardados como se puede ver en la figura 2.

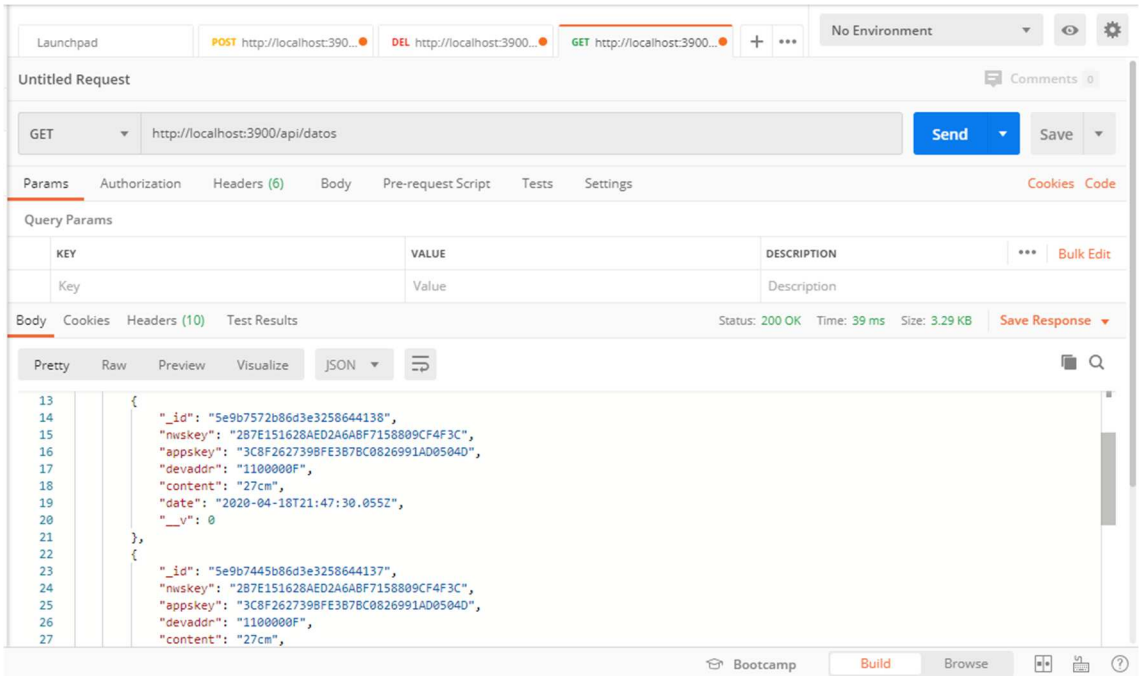


Figura S 2:Petición 'GET' del Backend a la Bade de datos

En el caso de una petición post, se envía el esquema del paquete completando los datos requeridos que son la clase de sesión, aplicación y la dirección del dispositivo, esto se guarda directamente en la base de datos dando un mensaje de 'success' (**¡Error! No se encuentra el origen de la referencia.**). De esta manera se comprueba que el backend y la base de datos tienen una comunicación correcta.

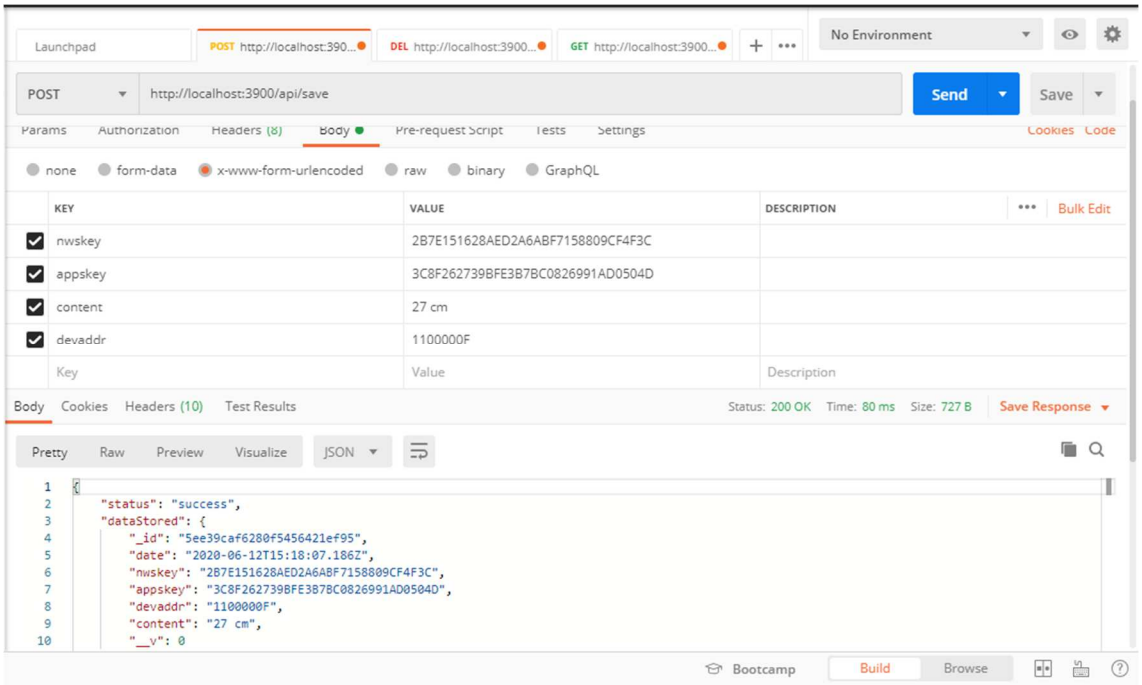


Figura S 3:Confirmación de dato guardado en Mongo DB

En las figuras anteriores se presenta la comunicación entre el Frontend con el Backend, ahora para lograr una simulación del Gateway LoRaWAN se debe realizar una comunicación por sockets, para ello se realizó el siguiente código

```
const dgram = require('dgram');
const message = Buffer.from(',`ÈÓÄ95>&EßªU@@À¨eÀ¨§ù¤Ë#¶F¯4VxeC!{"rxpk":
[{"tmst":3557271876,"chan":0,"rfch":0,"freq":868.100000,"stat":1,"modu"
:"LORA","datr":"SF12BW125","codr":"4/5","lsnr":7.8,"rssi":-14,
"size":15, "data":"QA8AABEADAB4n3gqcamy"}]}');
const client = dgram.createSocket('udp4');
client.send(message, 1700, '0.0.0.0', (err) => {
  client.close();
});
```

Donde el paquete enviado por el Gateway LoRaWAN se guarda en la constante 'message' que se envía por el puerto 1700 con la dirección IP 0.0.0.0, al enviar el mensaje el sistema SAD, separa, desencripta y guarda la información necesaria.

```
npm
(node:2916) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a f
uture version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the Mon
goClient constructor.
Servidor corriendo en http://localhost: 3900
server listening 0.0.0.0:1700
[nodemon] restarting due to changes...
[nodemon] starting `node index_app_web.js`
(node:8100) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a f
uture version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the Mon
goClient constructor.
Servidor corriendo en http://localhost: 3900
server listening 0.0.0.0:1700
dato encriptado: QA8AABEADAB4n3gqcamy
Desencrypt 20
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
[nodemon] starting `node index_app_web.js`
(node:3056) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a f
uture version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the Mon
goClient constructor.
Servidor corriendo en http://localhost: 3900
dato encriptado: QA8AABEADAB4n3gqcamy
Desencrypt data: 20
dato encriptado: QA8AABEACAB4kpX6WCI4
Desencrypt data: 19
```

Figura S 4: Desencriptación de datos enviaos por sockets

En este caso en particular son datos de un sensor de temperatura, en el frontend se puede ver el dato recibido

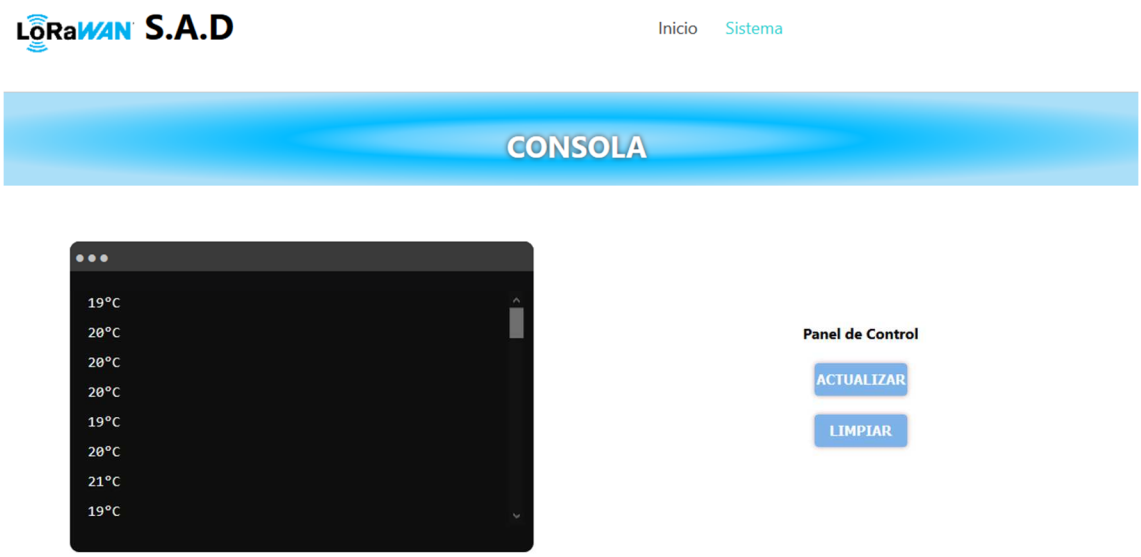


Figura S 5:Ventada de Frontend, donde se muestras los datos que posee Mongo DB

ANEXOS

ANEXO 1: PRÁCTICAS DE LABORATORIO

El tiempo que se considera para el desarrollo de las prácticas es de dos horas.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR		**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:			**ASIGNATURA**:
NRO. PRÁCTICA:	1	**TÍTULO PRÁCTICA**: Instalación y configuración del servidor Docker para LORAWAN	
OBJETIVOS: • **Instalación del software LoRaSuite, Virtual Box y Docker Toolbox** • **Selección de imagen del servidor** • **Configuración de container** • **Inicialización del servidor**			
INSTRUCCIONES:		1. Descargar e instalar el software LoRa Suite(1), Virtual Box (2) y Docker ToolBox(3)	
		2. Cargar imagen del servidor LoRa **"mchplora"**, que se encuentra dentro del directorio "microchip/LoRaSuite/Docker/mchplora", la dirección se debe cargar en la consola de Docker Quickstart Terminal.	
		3. Comprobar la carga correcta de la imagen, utilizar el comando "docker images"	
		4. Crear Docker Container, con las siguientes características • Name: lora_server • **Gateway traffic:** En el puerto 1700, • **MySQL traffic:** En el puerto 3306 • **Costumer server traffic:** En el puerto 5000.	
		5. Iniciar el servidor lora_server	
		6. Detener el servidor lora_server	
MARCO TEORICO (Investigar los siguientes conceptos para el desarrollo de la práctica)			
1. **Investigar el funcionamiento de las máquinas virtuales** 2. **Investigar que es Docker y su utilidad** 3. **Concepto de containers** 4. **Diferencia entre un container y una máquina virtual**			
ACTIVIDADES DESARROLLADAS			

(Anotar las actividades que siguió para el desarrollo de la práctica)
CONCLUSIONES:
BIBLIOGRAFIA:

Para su desarrollo revisar el, Apéndice A: Instalación de visual studio y Apéndice B: Instalación de node js.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:	**ASIGNATURA**:

NRO. PRÁCTICA:	2	**TÍTULO PRÁCTICA**: Configuración de Puertos de reenvió y de direcciones IPs estáticas de la red local.

OBJETIVO:

- **Asignación de puertos de reenvió en Máquina Virtual (MV).**
- **Configuración de IPs estática en el servidor.**
- **Conexión entre MV y Gateway del servidor.**

INSTRUCCIONES:	1. Ingresar a la configuración de la MV (nombre por defecto "default") • Seleccionar Red • Reenvió de puertos • Configurar con los datos del Docker Container.
	2. Ingresar a panel de control/ • Redes e Internet • Centro de redes y recursos compartidos • Cambiar configuración de adaptador, donde seleccionamos el adaptador ethernet y cambiamos la dirección a x.x.x.x con la máscara x.x.x.x
	3. Realizar ping entre la Pc y la máquina virtual, se verifica que hay comunicación.

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Que son las IPs Estáticas y IPs Dinámicas**
2. **Que es la máscara de red**
3. **Investigar que es el reenvió de puertos y su utilidad**

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice C: Instalación Base de datos, Apéndice E: Instalación de LoRa Utility, Apéndice F: Instalación de Docker Toolbox y Apéndice G: Configuración de Puertos en Virtual Box

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:	**ASIGNATURA**:

NRO. PRÁCTICA:	3	**TÍTULO PRÁCTICA**: Configuración de puertos GPIO's de la tarjeta lora MOTE

OBJETIVO:

- **Configuración de Puertos Digitales GPIO con LoRa Devolopment Utility**
- **Comprobación de funcionamiento de puertos.**

INSTRUCCIONES:	1. Abrir la aplicación LoRa Development Utility, conectar la tarjeta MOTE y pulsar Find Devices.
	2. Ingresar a RN Module X, y seleccionar la pestaña Radio
	3. En el protoboard armamos una fila de 13 leds, los cuales se conectan en orden a los puertos GPIOX.
	4. En el panel GPIO Control, seleccionamos el PIN, cambiamos el modo a salida digital, y value cambia a 1, realizar a todos los GPIOs de las tarjetas.
	5. Comprobar que los pines funcionan correctamente

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Protoboard
3. (13) Leds
4. (13) Resistencias 220 ohm
5. (13) Conectores (hembra-macho)
6. (1) Cables USB 'tipo A' a Micro USB tipo B

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar las características del módulo RN2483**
2. **Detallar las partes que componen la tarjeta LoRA MOTE**

Esquema de la práctica:

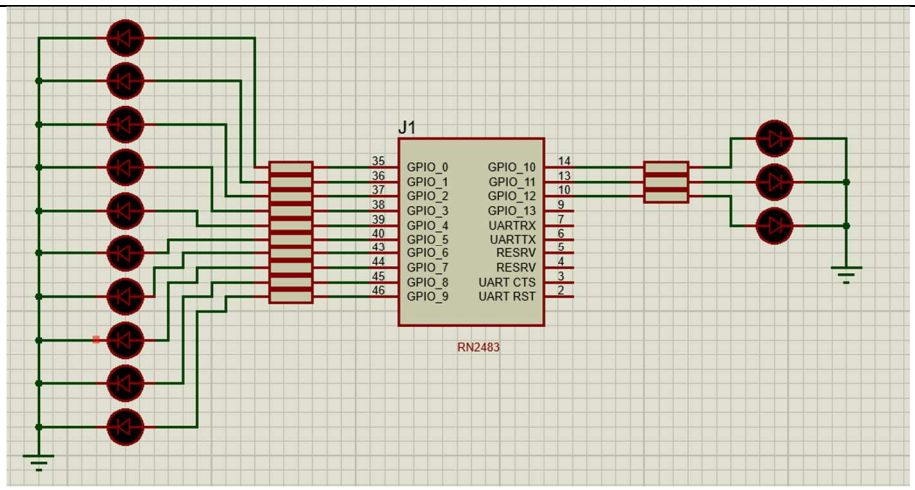


Figura: Conexión de leds con RN2483

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice I: Autoconfiguración de tarjeta LoRa MOTE y Apéndice L: Uso de Puertos GPIO de la Tarjeta MOTE.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:	**ASIGNATURA**:

NRO. PRÁCTICA:	4	**TÍTULO PRÁCTICA**: Comunicación entre tarjeta LoRa Core Board Gateway y Servidor Docker.

OBJETIVO:

- **Confirmación de comunicación entre servidor y LoRa Gateway.**

INSTRUCCIONES:	1. Abrir la aplicación LoRa Development Utility, conectar los dispositivos y pulsar Find Devices.
	2. Seleccionar Gateway X, configurar las direcciones IPs del Core Board Setup • Core Board IP: x.x.x.x • Network Router IP: x.x.x.x • Default Mask IP: x.x.x.x **Nota: El Gateway IP se deja por defecto, IP Allocation Mode se deja en static y el Sync Word no se cambia**
	3. Configurar los parámetros de la pestaña Server Setup • LoRa Server IP: x.x.x.x • Server Port: UP → 1700 Down → 1700 • Keep Alive interval: 10
	4. En la pestaña General Actions debemos hacer clic en: • Config all Parameters • Save Setting to SD Card.
	5. Realizar un ping desde el servidor a la dirección IP del LoRa Gateway.

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta Gateway Core
2. (1) Tarjeta Gateway Radio
3. (1) Computador
4. (1) Cable de red
5. (2) Cables USB 'tipo A' a Micro USB tipo B

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar que es LoRa**
2. **Investigar que es LoRaWAN**

3. Detallar las partes que componen la tarjeta LoRA Core Board Gateway

Esquema de la práctica:

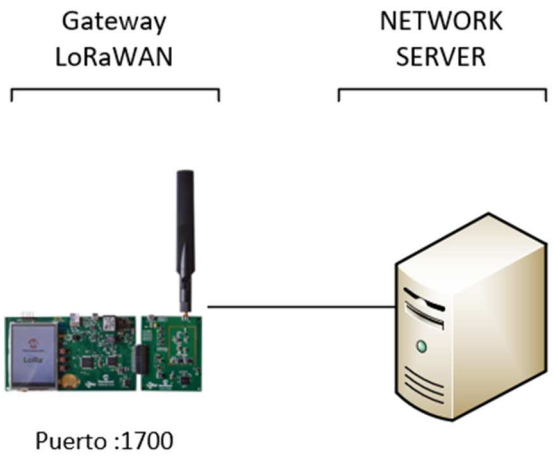


Figura: Conexión de Gateway LoRaWAN con Servidor Docker

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el Capítulo 1: Fundamentación Teórica o Estado del Arte y Apéndice I: Autoconfiguración de tarjeta LoRa MOTE.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	5	**TÍTULO PRÁCTICA**: Comunicación entre LoRa Gateway y LoRa MOTE con autoconfiguración.

OBJETIVO:

- **Configuración de LoRa MOTE**
- **Habilitación de canales disponibles en LoRa MOTE**
- **Comunicación entre servidor y LoRa MOTE**

INSTRUCCIONES:	1. Abrir la aplicación LoRa Development Utility, conectar los dispositivos y pulsar Find Devices.
	2. Configurar el Gateway con las direcciones Ip correspondientes
	3. Añadir el servidor en LoRa Development Utility y conectar con la puerta de enlace predeterminada. **Nota: En la aplicación LoRa Development Utility, seleccionar la pestaña Server/add y configurar en Database Connection la dirección IP del Gateway del servidor.**
	4. Configuración de los parámetros de LoRa MOTE, sebe presionar el botón Config for Auto-Create Application Server y habilitar los canales **Nota: preconfigura el dispositivo con la clave de sesión de red especificada (NwSKey) y la clave de sesión de aplicación (AppSKey).**
	5. Envió de mensajes tipo ASCII al servidor. • Ingresar a RN MODULE X, en la pestaña LoRa WAN • En Communication configurar Puerto, ASCII, Data Rate y Data.
	6. Verificar en LoRa Gateway los datos enviados.

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta Gateway Core
2. (1) Tarjeta Gateway Radio
3. (1) Tarjeta LoRa MOTE

4. (1) Computador
5. (1) Cable de red
6. (3) Cables USB 'tipo A' a Micro USB tipo B

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Clases de LoRaWAN**
2. **Cuantos canales tienen cada clase de LoRaWAN**
3. **Investigar cuales son las frecuencias en las que se pueden trabajar con LoRaWAN**

Esquema de la practica

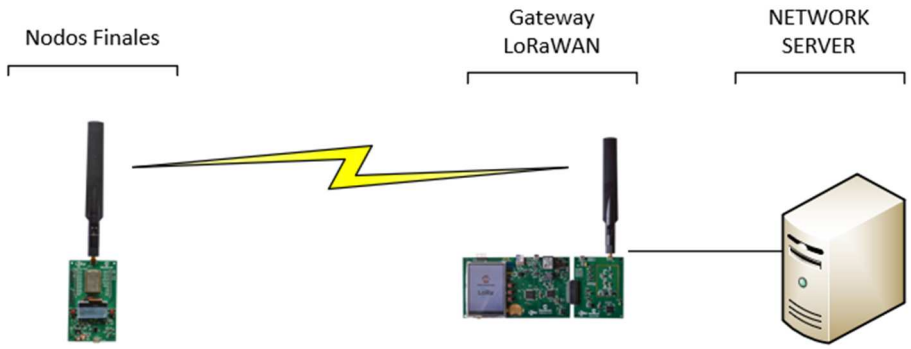


Figura: Comunicación de LoRa MOTE y LoRa Gateway

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el capítulo 1, sección 1.3 Tecnología LoRaWAN y Apéndice I: Autoconfiguración de tarjeta LoRa MOTE.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:			**ASIGNATURA**:
NRO. PRÁCTICA:	6	**TÍTULO PRÁCTICA**: Comunicación entre LoRa Gateway y LoRa MOTE con configuración ABP.	

OBJETIVO:

- **Configuración de Lora MOTE mediante ABP.**
- **Envío de mensajes ASCII de MOTE a Servidor.**
- **Verificación de mensajes ASCII enviados al servidor.**
- **Verificación de credenciales del dispositivo.**

INSTRUCCIONES:	1. Abrir la aplicación LoRa Development Utility, conectar los dispositivos y pulsar Find Devices
	2. Configurar el Gateway con las direcciones IP correspondientes
	3. Añadir el servidor en LoRa Development Utility y conectar con la puerta de enlace predeterminada.
	4. Ingresar a Server, dentro de la pestaña End-Device-Action, ir a Provisioned ABP e ingresar los datos de: • Device Address • Network Session Key • Application Session Key. **Nota: la longitud máxima es de 16 bytes**
	5. Ingresar a RN Module, en la pestaña Server Autentification Key y configurar los mismos parámetros que se ingresaron en el servidor.
	6. Presionar Save y después Join, para su óptimo funcionamiento
	7. Comprobar la autentificación de los MOTE´s transmitiendo símbolos ASCII al servidor • Ingresar a RN MODULE X, en la pestaña LoRa WAN • En Communication configurar Puerto, ASCII, Data Rate y Data.
	8. Verificar los datos enviados de las tarjetas MOTEs en el servidor. • Ingresar Server, en la pestaña Database • En Selec Database Table View, elegimos data Traffic

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta Gateway Core
2. (1) Tarjeta Gateway Radio
3. (1) Tarjeta LoRa MOTE

4. (1) Computador
5. (1) Cable de red
6. (3) Cables USB 'tipo A' a Micro USB tipo B

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar que es la conexión ABP**
2. **Investigar que son los siguientes parámetros para un dispositivo LoRaWAN**
   - Device Address
   - Network Session Key
   - Application Session Key

Esquema de la práctica a desarrollar

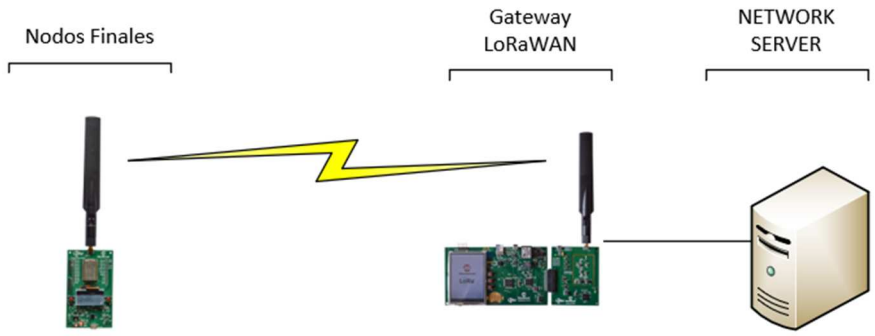


Figura: Comunicación de LoRa MOTE y LoRa Gateway

Anotar los parámetros de los dispositivos en la tabla

Configuración	MOTE 1
Device Address	0x
Network Session Key	0x
Application Session Key	0x

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BLIOGRAFIA:

Para su desarrollo revisar el capítulo 1, sección 1.3 Tecnología LoRaWAN y Apéndice I: Autoconfiguración de tarjeta LoRa MOTE.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	7	**TÍTULO PRÁCTICA**: Comunicación entre lora gateway y dos LoRa mote con configuración ABP.

OBJETIVO:

- **Configuración de Lora MOTE mediante ABP.**
- **Envío de mensajes ASCII de MOTE a Servidor.**
- **Verificación de mensajes ASCII enviados al servidor.**
- **Verificación de las credenciales de cada dispositivo**

INSTRUCCIONES:

1. Abrir la aplicación LoRa Development Utility, conectar los dispositivos y pulsar Find Devices
2. Configurar el Gateway con las direcciones Ip correspondientes
3. Añadir el servidor en LoRa Development Utility y conectar con la puerta de enlace predeterminada.
4. Ingresar a Server, dentro de la pestaña End-Device-Action, ir a Provisioned ABP e ingresar los datos de:
   - Device Address
   - Network Session Key
   - Application Session Key

   Para los dos dispositivos MOTEs

   Nota: la longitud máxima es de 32 bits
5. Ingresar a RN Module/Server Autentification Key y configurar los mismos parámetros que se ingresaron en el servidor en los dos dispositivos MOTEs
6. Presionar Save y después Join, para su óptimo funcionamiento
7. Comprobar la autentificación de los MOTE´s transmitiendo símbolos ASCII al servidor
   - Ingresar a RN MODULE X, en la pestaña LoRa WAN
   - En Communication configurar Puerto, ASCII, Data Rate y Data.
8. Verificar los datos enviados de las tarjetas MOTEs en el servidor.
   - Ingresar Server, en la pestaña Database
   - En Selec Database Table View, elegimos data Traffic

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

Esquema de la práctica a desarrollar

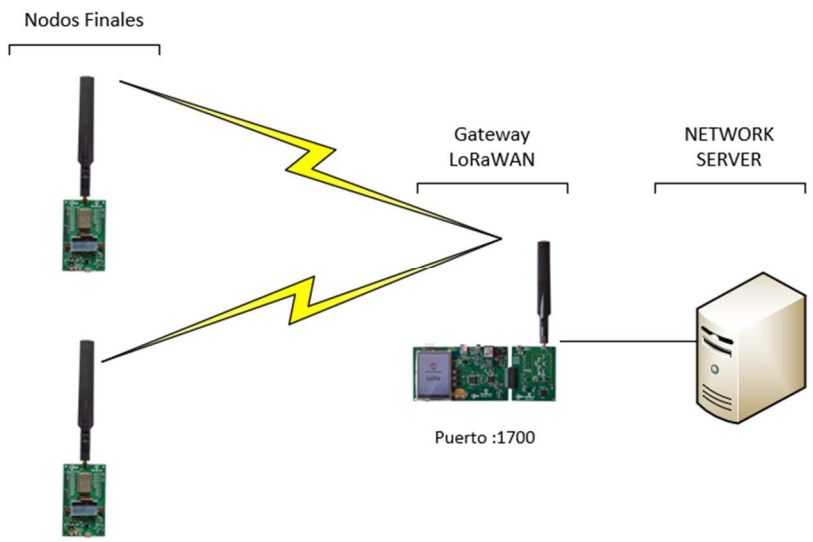


Figura : Comunicación de dos dispositivos LoRa MOTE y LoRa Gateway mediante ABP

Anotar los parámetros de los dispositivos en la tabla

Configuración	MOTE 1	MOTE 2
Device Address	0x	0x
Network Session Key	0x	0x
Application Session Key	0x	0x

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BLIOGRAFIA:

Para su desarrollo revisar el capítulo 1, sección 1.3 Tecnología LoRaWAN y Apéndice R: Implementación de conexión ABP.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR		**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:			**ASIGNATURA**:
NRO. PRÁCTICA:	8	**TÍTULO PRÁCTICA**: Comunicación entre Lora Gateway y LoRa MOTE con configuración OTTA.	
OBJETIVO: • **Configuración de parámetros para el Server Application** • **Creación de credenciales OTAA en el servidor** • **Configuración de MOTE con Credenciales OTAA** • **Verificación de envió de mensajes ASCII al servidor.**			
INSTRUCCIONES:		1. Abrir la aplicación LoRa Development Utility, conectar los dispositivos y pulsar Find Devices	
		2. Configurar el Gateway con las direcciones Ip correspondientes	
		3. Añadir el servidor en LoRa Development Utility y conectar con la puerta de enlace predeterminada.	
		4. Configurar los parámetros de la pestaña Server Aplication • **Aplication Extended-Unique-Id**: 0x98fe • **Server Aplication Name:** Mi Aplicación OTAA • **Server Aplication Owner:** UPS_LAB: P1234	
		5. En la pestaña Non-Provisioned (OTAA) configuramos los siguientes parámetros: • Server Aplication EUI: • Application Key (APPKEY) • Device Extended-Unique-Id (DevEUI) **NOTA: El Server Aplication EUI debe ser el mismo valor que se ingresó en el Server Aplication**	
		6. Ingresamos al dispositivo RN MODULE X, seleccionamos la pestaña LoRa WAN, nos dirigimos a Server Authentification Keys y escogemos OTAA. **NOTA: Los parámetros que se configuraron el servidor en la pestaña Non-Provisioned (OTAA) se deben colocar en el RN MODULE X.**	
		7. Guardar la configuración del RN MODULE al presionar Save y después Join, para su óptimo funcionamiento	
		8. Comprobar la autentificación de los MOTE´s transmitiendo símbolos ASCII al servidor. • Ingresar a RN MODULE X, en la pestaña LoRa WAN	

	• En Communication configurar Puerto, ASCII, Data Rate y Data.
	9. Verificar los datos enviados de las tarjetas MOTEs en el servidor. • Ingresar Server, en la pestaña Database • En Selec Database Table View, elegimos data Traffic

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar que es la conexión OTAA**
2. **Investigar que son los siguientes parámetros para la conexión OTAA**
   - Server Aplication EUI:
   - Application Key (APPKEY)
   - Device Extended-Unique-Id (DevEUI)

Esquema de la práctica a desarrollar

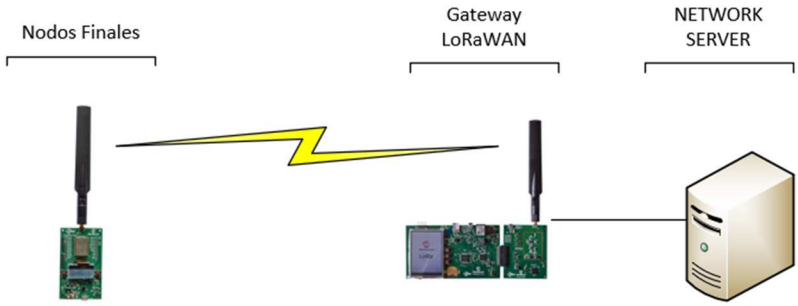


Figura: Comunicación de LoRa MOTE y LoRa Gateway mediante OTAA

Anotar los parámetros de los dispositivos en la tabla

Configuración	MOTE 1
Server Aplication EUI	0x
Application Key (APPKEY)	0x
Device Extended-Unique-Id (DevEUI)	0x

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BLIOGRAFIA:

Para su desarrollo revisar el Apéndice J: Implementación de conexión OTAA y Apéndice K: Confirmación de conexión oTAA

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	9	**TÍTULO PRÁCTICA**: Instalación de MPLAB X IDE, XC8, MCC y librería LORAWAN.

OBJETIVO:

- Instalar software para la configuración personalizada de la Tarjeta LoRaWAN MOTE.

INSTRUCCIONES:	1. Descargar el Software MPLAB X IDE (https://www.microchip.com/mplab/mplab-x-ide), el software XC8 (https://www.microchip.com/mplab/compilers) y la librería LoRaWAN (http://en.pudn.com/Download/item/id/3389426.html).
	2. Instalar el Software MPLAB X IDE
	3. Instalar el software XC8
	4. Abrir el Software MPLAB X IDE y agregar la librería lorawan. • Abrir la pestaña Tools/ Options • Ir a Plugins/ Install Library • Buscar el directorio donde se guardó la librería lorawan • Seleccionar la librería y pulsar abrir • Dar clic en OK
	5. Instalar la herramienta MCC • Abrir la pestaña Tools / Plugins • Ir a la pestaña Available Plugins • Seleccionar MPLAB Code Configurator • Clic en install y luego en close

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar que es la herramienta MCC y para que se utiliza**
2. **Investigar los archivos que se crean al generar un proyecto MCC**
3. **Indicar los módulos que se abren al momento de generar un proyecto MCC**

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:
BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice M: Instalación de MPLAB X IDE, Apéndice Ñ.1: Instalación de Plugin Lorawan y Apéndice Ñ.2: Instalación de MCC

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	10	**TÍTULO PRÁCTICA**: Configuración de estado de un pin en LoRa MOTE mediante MPLAB X IDE y herramienta MCC.
OBJETIVO: • Configuración de puerto de salida digital de un GPIO		
INSTRUCCIONES:	1. Iniciar el software MPLAB X IDE y crear un nuevo proyecto.	
	2. En la barra de tareas, nos dirigimos a Windows • MPLAB Code Configurator • Damos clic en MCC MPLAB Code Configurator : Open/Close.	
	3. Colocar un nombre al proyecto MCC y guardar	
	4. Abrimos el Pin Manager y seleccionamos como salida • GPIO 0	
	5. En Pin Module cambiaos el nombre a GPIO_0	
	6. Configuramos el System Module • **Oscillatr Selectec:** Internal oscilator block • **System Clock Select:** FOSC • **Internal Clock:** 16MHz_HFINTOSC • **Wachtdog timer Enable:** always disable. • **Wachtdog Timer Postscaler:** 1:32768 • **Low-voltage Programming Enable:** En chek	
	7. Nos dirigimos a la ventana izquierda Resource Management. • En la pestaña Tree View • Pulsamos el botón Generate **NOTA: Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Sources Files)**	
	8. En la ventana Projects abrimos los archivos fuente, y damos doble clic al main.c	
	9. Desarrollar la siguiente secuencia: • GPIO_0 estado en 1 espera 500 ms y cambian a 0	

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Protoboard
3. (1) Snap
4. (1) Leds
5. (1) resistencia 220 ohm
6. (10) Conectores (hembra-macho)
7. (3) Cables USB tipo A a Micro USB tipo B
8. (1) Cable de Red

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar que es la tarjeta SNAP, las características principales, sus componentes y su funcionamiento.**
2. **Diagrama de conexión del SNAP y tarjeta LoRa MOTE**

Esquema de la práctica a desarrollar

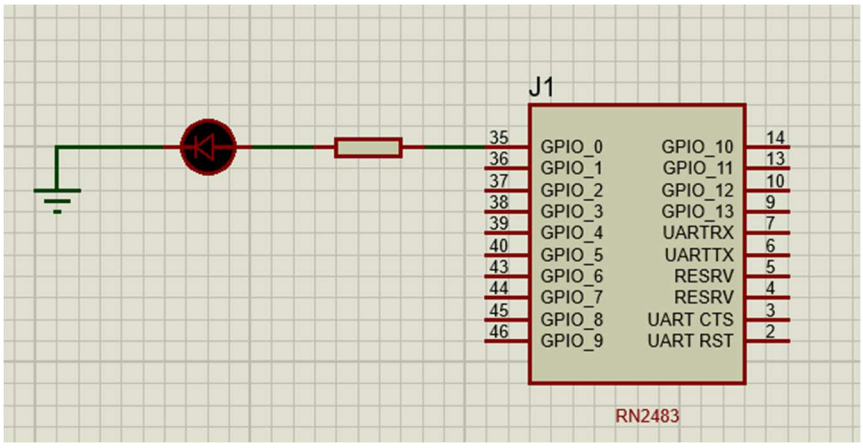


Figura: Diagrama de conexión de led con el módulo RN2483

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice O.1: Configuración de PIC18LF46k22 como Salida Digital

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	11	**TÍTULO PRÁCTICA**: Programación de secuencia con leds en LoRa MOTE mediante MPLAB X IDE y herramienta MCC.
OBJETIVO: • Configuración de puertos como salidas digitales GPIOs		

INSTRUCCIONES:	1. Iniciar el software MPLAB X IDE y crear un nuevo proyecto.
	2. En la barra de tareas, nos dirigimos a Window • MPLAB Code Configurator • Damos clic en MCC MPLAB Code Configurator : Open/Close.
	3. Colocar un nombre al proyecto MCC y guardar
	4. Abrimos el Pin Manager y seleccionamos como salidas • GPIO 1 – GPIO 13
	5. En Pin Module cambiamos los nombres que salen por defecto a GPIO_X **Nota: X son los números que los GPIOS que están en la tarjeta LoRa MOTE**
	6. Configuramos el System Module • **Oscillatr Selectec:** Internal oscilator block • **System Clock Select:** FOSC • **Internal Clock:** 16MHz_HFINTOSC • **Wachtdog timer Enable:** always disable. • **Wachtdog Timer Postscaler:** 1:32768 • **Low-voltage Programming Enable:** En chek
	7. Nos dirigimos a la ventana izquierda Resource Management. • En la pestaña Tree View • Pulsamos el botón Generate **NOTA: Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Sources Files)**
	8. E la ventana Projects abrimos los archivos fuente, y damos doble clic al main.c
	9. Desarrollar la siguiente secuencia:

	• GPIO_0 – GPIO_4 estado en 1 esperan 500 ms y cambian a 0 • GPIO_5 – GPIO_9 estado en 1 esperan 200 ms y cambian a 0 • GPIO_10 – GPIO_13 estado en 1 esperan 500 ms y cambian a 0

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Protoboard
3. (1) Snap
4. (13) Leds
5. (13) Resistencia de 220ohm
6. (19) Conectores (hembra-macho)
7. (3) Cables USB tipo A a Micro USB tipo B
8. (1) Cable de red

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

Esquema de la practica

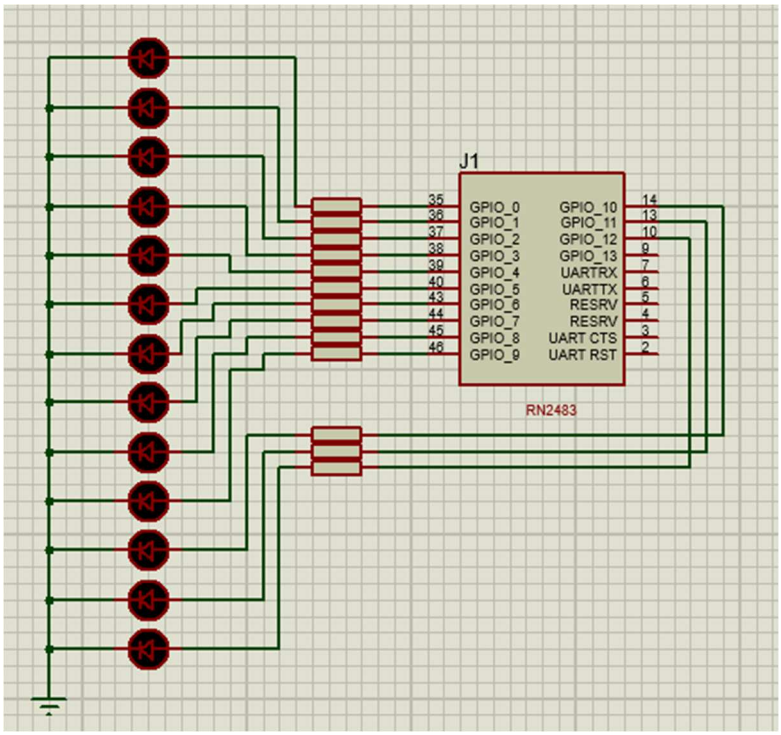


Figura: Conexión de leds al módulo RN2483

ACTIVIDADES DESARROLLADAS (Anotar las actividades que siguió para el desarrollo de la práctica)
CONCLUSIONES:
BIBLIOGRAFIA:

Para su desarrollo revisar el capítulo 1, sección 1.3 Tecnología LoRaWAN y Apéndice R: Implementación de conexión ABP.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	12	**TÍTULO PRÁCTICA**: Configuración de un puerto analógico como entrada.
OBJETIVO: • Configurar el puerto analógico como entrada.		
INSTRUCCIONES:	1. Iniciar el software MPLAB X IDE y crear un nuevo proyecto.	
	2. En la barra de tareas, nos dirigimos a Window • MPLAB Code Configurator • Damos clic en MCC MPLAB Code Configurator : Open/Close.	
	3. Colocar un nombre al proyecto MCC y guardar	
	4. En la Ventana de Reosurce Management, nos dirigimos a Device Resources • Agregamos el periférico ADC y el CCP5	
	5. Abrimos el Pin Manager, en el módulo ADC: • Configuramos el puerto GPIO_0 como canal analógico	
	6. Abrimos el Pin Manager, en el módulo Pin Module: • Configuramos el puerto GPIO_4 como salida digital	
	7. En Pin Module cambiamos el nombre a GPIO_0 y GPIO_4 **Nota: X son los números que los GPIOS que están en la tarjeta LoRa MOTE**	
	8. En la pestaña ADC, configuramos los siguientes parámetros • Enable ADC: check • Clock Source: FRC • Acquisition Time: 2 • Auto-conversion Trigger: CCP5 • Enable ADC Interrup: check • Los demos parámetros los dejamos.	
	9. Ingresamos a CCP5 y lo configuramos en off/reset	
	10. Configuramos el System Module • **Oscillator Selectec:** Internal oscilator block	

	• **System Clock Select:** FOSC • **Internal Clock:** 16MHz_HFINTOSC • **Wachtdog timer Enable:** always disable. • **Wachtdog Timer Postscaler:** 1:32768 • **Low-voltage Programming Enable:** En chek
	11. Nos dirigimos a la ventana izquierda Resource Management. • En la pestaña Tree View • Pulsamos el botón Generate **NOTA: Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Sources Files)**
	12. En la ventana Projects abrimos los archivos fuente, y damos doble clic al main.c
	13. Desarrollar un algoritmo donde el puerto A0 es la entrada analógica y active al puerto A4 (salida digital) en un determinado valor.

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Protoboard
3. (1) Snap
4. (1) Leds
5. (10) Conectores (hembra-macho)
6. (1) potenciómetro de 20K

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

Esquema de la práctica:

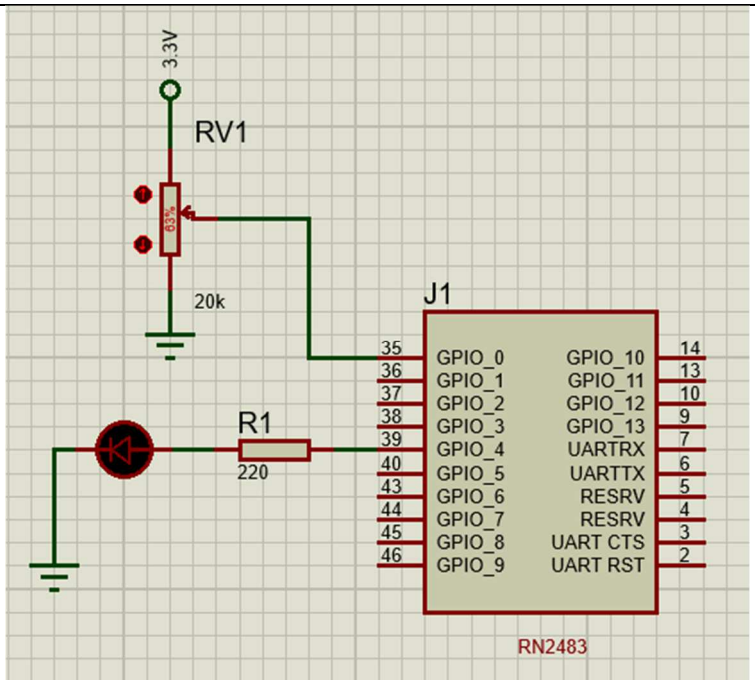


Figura: Conexión de Potenciómetro y led al módulo RN2483

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice O.3: Configuración de PIC18LF46k22 con Entrada Analógica y Salida Digital.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA: 13	**TÍTULO PRÁCTICA**: Configuración del TMR0 del Módulo LoRa MOTE	
OBJETIVO: • Configurar los parámetros necesarios para el funcionamiento del TMR0		
INSTRUCCIONES:	1. Iniciar el software MPLAB X IDE y crear un nuevo proyecto.	
	2. En la barra de tareas, nos dirigimos a Window • MPLAB Code Configurator • Damos clic en MCC MPLAB Code Configurator : Open/Close.	
	3. Colocar un nombre al proyecto MCC y guardar	
	4. En la Ventana de Reosurce Management, nos dirigimos a Device Resources • Agregamos el TMR0	
	5. Abrimos el Pin Manager, en el módulo Pin Module: • Configuramos el puerto A5 como salida digital	
	6. En Pin Module cambiamos el nombre a GPIO_X del puerto A5 **Nota: X son los números que los GPIOS que están en la tarjeta LoRa MOTE**	
	7. En la pestaña TMR0, configuramos los siguientes parámetros • **Enable Timer:** check • **Enable Prescaler:** check – 1:16 • **Timer mode:** 16-bit • **Clock Source:** FOSC/4 • **Requested Period:** X ms	
	8. Ingresamos a CCP5 y lo configuramos en off/reset	
	9. Configuramos el System Module • **Oscillator Selectec:** Internal oscilator block • **System Clock Select:** FOSC • **Internal Clock:** 16MHz_HFINTOSC • **Wachtdog timer Enable:** always disable. • **Wachtdog Timer Postscaler:** 1:32768	

- **Low-voltage Programming Enable:** En chek

10. Nos dirigimos a la ventana izquierda Resource Management.
    - En la pestaña Tree View
    - Pulsamos el botón Generate

NOTA: Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Sources Files)

11. En la ventana Projects abrimos los archivos fuente, y damos doble clic al main.c
12. Realizar el cambio de estado con un delay de 250ms, 300ms, 400ms y 500ms de un led en el puerto A5 con el Timer0

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Protoboard
3. (1) Snap
4. (1) Leds
5. (1) resistencia de 220ohm
6. (10) Conectores (hembra-macho)

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar el funcionamiento del TMR0 en el datasheet del microcontrolador**
2. **Investigar cuales son las banderas de desbordamiento que tiene el TMR0**

Esquema de la práctica:

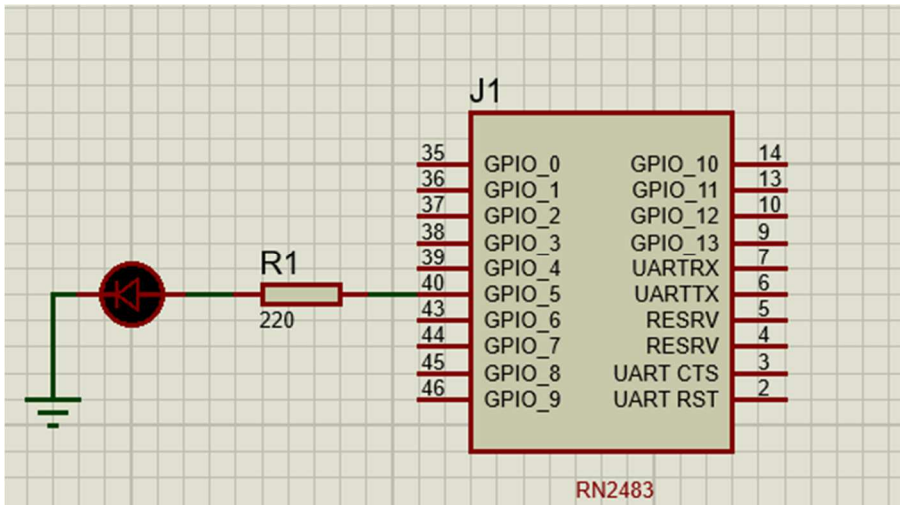


Figura: Conexión de led al módulo RN2483

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:
BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice O.4: Configuración de PIC18LF46k22 con TMR0

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR		**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:			**ASIGNATURA**:
NRO. PRÁCTICA:	14	**TÍTULO PRÁCTICA**: Modulación de ancho de Pulso (PWM) con Timer2 y CCP5 en LoRa MOTE.	
OBJETIVO: • **Configuración de los parámetros TMR2 y CCP5 para el PWM**			
INSTRUCCIONES:		1. Iniciar el software MPLAB X IDE y crear un nuevo proyecto.	
		2. En la barra de tareas, nos dirigimos a Window • MPLAB Code Configurator • Damos clic en MCC MPLAB Code Configurator : Open/Close.	
		3. Colocar un nombre al proyecto MCC y guardar	
		4. En la Ventana de Reosurce Management, nos dirigimos a Device Resources y agregamos los periféricos: • Agregamos el TMR2 • CCP5	
		5. Abrimos el Pin Manager, en el módulo Pin Module: • Configuramos el puerto A5 como salida digital	
		6. En la pestaña TMR2, configuramos los siguientes parámetros • **Enable Timer:** check • **Postscaler:** 1:X	

	• **Prescaler:** 1:X • **Timer Period:** X ms
	7. Ingresamos a CCP5 y configuramos • CCP Mode: PWM • Select Timer: Timer 2 • Duty Cycle: X %
	8. En Pin Manager seleccionamos el pin RE2 como salida digital del módulo CCP5
	9. Configuramos el System Module • **Oscillator Selectec:** Internal oscilator block • **System Clock Select:** FOSC • **Internal Clock:** 16MHz_HFINTOSC • **Wachtdog timer Enable:** always disable. • **Wachtdog Timer Postscaler:** 1:32768 • **Low-voltage Programming Enable:** En chek
	10. Nos dirigimos a la ventana izquierda Resource Management. • En la pestaña Tree View • Pulsamos el botón Generate **NOTA: Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Sources Files)**
	11. En la ventana Projects abrimos los archivos fuente, y damos doble clic al main.c
	12. Realizar la configuración del TMR2 y CCP5, viendo los cambios que se producen en la salida del pin RE2, además tomar capturas en el osciloscopio de las señales.

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Protoboard
3. (1) Snap
4. (1) Leds
5. (1) resistencia de 220ohm
6. (10) Conectores (hembra-macho)

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar el funcionamiento del TMR2 en el datasheet del microcontrolador**
2. **Investigar el funcionamiento del CCP5**

Esquema de la práctica:

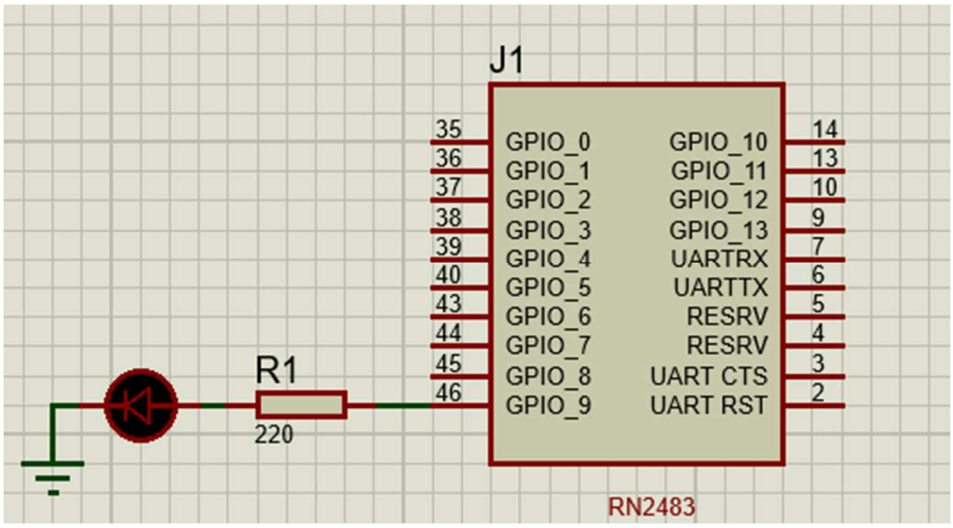


Figura: Conexión de led al módulo RN2483 con configuración de PWM

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice O: Desarrollo de ejemplos con MCC

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	15	**TÍTULO PRÁCTICA**: Configuración de protocolo LoRaWAN con MPLAB y MCC

OBJETIVO:

- **Configuración de periféricos y pines para la comunicación con el radio tranceiver**

INSTRUCCIONES:	1. Iniciar el software MPLAB X IDE y crear un nuevo proyecto.
	2. En la barra de tareas, nos dirigimos a Window • MPLAB Code Configurator • Damos clic en MCC MPLAB Code Configurator : Open/Close.
	3. Colocar un nombre al proyecto MCC y guardar
	4. En la pestaña Resource Management nos dirigimos • Device resource en donde se busca la librería LORAWAN • Agregar la Librería LORAWAN
	5. En la pestaña LORAWAN configurar los parámetros siguientes • Base Timer: TMR1 • Radio Module: SX1276 • SPI Module: MSSP2 • ISM Band: Europe 868 • Default class: A
	6. En Device Resoures agregamos lo periféricos: • EXT_INIT • MSSP2 • TMR1
	7. En la pestaña System Module configuramos: • **Oscillator Select:** Internal oscillator block • **System Clock Select:** FOSC • **Internal Clock:** 16MHz_HFINTOSC • **Wachtdog timer Enable:** always disable. • **Wachtdog Timer Postscaler:** 1:32768 • **Low-voltage Programming Enable:** cheked
	8. En ventana Project Resource, nos dirigimos a la pestaña Pheripherals damos doble clic en Timer1 y configuramos: • **Enable Timer:** checked

	• **Timer Clock** ➢ **Clock Source:** External @ 32.768 kHz ➢ **Prescaler:** 1:1 ➢ **Enable Synchronization:** Not checked ➢ **Enable Oscillator Circuit:** Checked • **Timer Period** ➢ **Timer Period:** 2s ➢ **Period Count:** (automatically filled by MCC) ➢ **Enable 16-bit read:** not checked • **Enable Gate:** not checked • **Enable Timer Interrupt:** checked • **Software Settings** ➢ **Callback Function Rate:** 1
	9. En ventana Project Resource, nos dirigimos a la pestaña Pheripherals damos doble clic en MSSP2 y configuramos: • **Mode:** SPI Master • **Enable MSSP:** checked • **Input Data Sampled at**: Middle • **Serial Protocol:** SPI Mode • **SPI Mode:** 0 (automatically filled by MCC) • **Clock Source:** FOSC/4 ➢ **SPI Clock:** 4000.0 kHz (automatically filled by MCC)
	10. En Pin manager, nos dirigimos al módulo LORAWAN y configuramos: ➢ DIO0 – PORTB Pin 1 (RB1) ➢ DIO1 – PORTB Pin 2 (RB2) ➢ DIO2 – PORTB Pin 4 (RB4) ➢ DIO3 – no se usa ➢ DIO4 – no se usa ➢ DIO5 – PORTB Pin 0 (RB0) ➢ NRESET – PORTC Pin 2 (RC2) ➢ NSS – PORTD Pin 3 (RD3) ➢ SW_POW – no se usa
	11. En la ventana Pin Module configuramos: • En la columna WPU del RB0 al RB4 damos en cheked • En la columna IOC en la fila del RB4 configuramos en **<u>any</u>**
	12. En la ventana Interrup Module, se deshabilita la INT0, debido a que el pin RB0, correspondiente a la interrupción ya mencionada.
	13. Nos dirigimos a la ventana izquierda Resource Management.

	• En la pestaña Tree View • Pulsamos el botón Generate **NOTA: Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Sources Files)**
ACTIVIDADES DESARROLLADAS (Anotar las actividades que siguió para el desarrollo de la práctica)	
CONCLUSIONES:	
BIBLIOGRAFIA:	

Para su desarrollo revisar el Apéndice P: Configuración LoRaWAN en PIC18LF46k22.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUAD	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**	
CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	16	**TÍTULO PRÁCTICA**: Configuración de Encriptación AES dentro protocolo LoRaWAN con MPLAB
OBJETIVO: • **Configuración de la librería de encriptación para el MCU de 8-bits**		
INSTRUCCIONES:	1. Iniciar el software MPLAB X IDE y abrir la práctica 15	
	2. Descargar la librería de encriptación para MCUs de 8-bits de https://www.microchip.com/Developmenttools/ProductDetails/SW300052, dentro de los archivos buscar la carpeta Data Encryption Libraries V2.6 • Abrir la carpeta AN0953 - Data Encryption Routines for the PIC18 • Dirigirse a AES • Abrimos los archivos AES.c y AES.h	
	3. En el panel izquierdo en Projects dirigirse a: • Headers Files • MCC Generaed Files • LoRaWAN • Abrimos el archivo AES.h,	
	4. Copiamos el contenido de AES.h del paso 2 y lo pegamos en el archivo AES.H del paso 3	
	5. En el panel izquierdo en Projects dirigirse • Sources Files • MCC Generaed Files • LoRaWAN • Abrimos el archivo AES.c,	
	6. Copiamos el contenido de AES.c del paso 2 y lo pegamos en el archivo AES.c del paso 5	
	7. En la barra de menú ir: • File • Project Properties • XC8 Global options • C stadard cambiamos a **C90** • Aplicar y aceptar	
	8. En el panel izquierdo en Projects dirigirse • Sources Files • MCC Generaed Files • LoRaWAN • Abrir los archivos	

	➢ Lorawan_init.c: comentar el contenido del DIO5_ISR_Lora_Init ➢ Lorawan.c: Borrar el código ´**reentrant**´
ACTIVIDADES DESARROLLADAS (Anotar las actividades que siguió para el desarrollo de la práctica)	
CONCLUSIONES:	
BIBLIOGRAFIA:	

Para su desarrollo revisar el Apéndice P: Configuración LoRaWAN en PIC18LF46k22.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**		
CARRERA:		**ASIGNATURA**:	
NRO. PRÁCTICA:	17	**TÍTULO PRÁCTICA**: Configuración de parámetros para la transmisión de datos en modo ABP con los módulos MOTEs.	
OBJETIVO: • **Inicialización del protocolo LoRaWAN para la transmisión de datos**			
INSTRUCCIONES:	1. Iniciar el software MPLAB X IDE y abrir la práctica 16		
	2. Habilitar las interrupciones Globales y periféricas		
	3. Creamos las funciones para el inicio de LoRAWAN • RxData • RxJoinResponse		
	4. Dentro de la función main, inicializamos LoRaWAN con la función • LORAWAN_Init (RxData, RxJoinResponse)		
	5. Asignamos los parámetros de configuración • LORAWAN_SetNetworkSessionKey( ) • LORAWAN_SetApplicationSessionKey() • LORAWAN_SetDeviceAddress( )		
	6. Declaramos el modo de operación de LoRaWAN • LORAWAN_Join(ABP)		
	7. Dentro del bucle infinito declarar • LORAWAN_Mainloop() • LORAWAN_Send (UNCNF, puerto, datos, longitud de datos);		
	8. Guardar y Programar el módulo MOTE		
	9. Verificar en el Gateway él envió del dato		
Herramientas			
Herramientas necesarias para realizar la práctica. 1. (1) Tarjeta LoRa MOTE 2. (1) Tarjeta Gateway Core Board 3. (1) Tarjeta Gateway Radio			
MARCO TEORICO (Investigar los siguientes conceptos para el desarrollo de la práctica)			
1. Investigar que hace cada una de las funciones que se especifican en las instrucciones.			
ACTIVIDADES DESARROLLADAS (Anotar las actividades que siguió para el desarrollo de la práctica)			

CONCLUSIONES:
BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice P: Configuración LoRaWAN en PIC18LF46k22.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	18	**TÍTULO PRÁCTICA**: Captura de datos LoRAWAN con la ayuda de la herramienta Wireshark

OBJETIVO:

- **Capturar los datos enviados al Gateway.**

INSTRUCCIONES:	1. Iniciar el software MPLAB X IDE y abrir la práctica 17
	2. Conectar el Gateway al PC y modificar la dirección IP del puerto ethernet
	3. Abrimos el software Wireshark, y comenzamos a capturar los datos
	4. Cargamos el programa realizado en la practica 17 en la Tarjeta LoRa MOTE
	5. Conectamos la tarjeta LoRa MOTE y esperamos que envié los datos
	6. En Wiresahark vemos los paquetes con la dirección IP del Gateway
	7. Tomar una captura de pantalla de los datos enviados

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Tarjeta Gateway Core Board
3. (1) Tarjeta Gateway Radio
4. (1) Cable de red

MARCO TEORICO

(Investigar los siguientes conceptos para el desarrollo de la práctica)

1. **Investigar que hace el Wireshark**
2. **Investigar la trama de datos de LoRaWAN**

ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice P: Configuración LoRaWAN en PIC18LF46k22.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	19	**TÍTULO PRÁCTICA**: Comunicación UART entre LoRa MOTE y Arduino nano para él envió de datos del sensor DHT_22

OBJETIVO:

- **Configuración del periférico UART en LoRA MOTE**
- **Configuración del sensor en Arduino**
- **Recepción de datos en consola**

INSTRUCCIONES:

1. Iniciar MongoDB, Backend y FrondEnd
2. Iniciar el Sistema de Adquisición de Datos (S.A.D)
3. Iniciar el software MPLAB X IDE y abrir la práctica 17
4. Abrimos el MCC ya creado
   - En la pestaña Projects
   - Important Files
   - Hacer doble clic en nombre_archivo.mc3
5. Añadimos el periférico EUSART 1
   - **Mode:** asynchronous
   - **Enable EUSART:** checked
   - **Baud Rate:** X
   - **Transmission Bit:** 8-bit
   - **Reception Bits:** 8-bit
   - **Data Polarity:** async_noninverted
   - **Enable Recevie:** checked
6. En la ventana Pin Manager, cambiamos el estado a salida al pin RC7 del módulo EUSART1 RX1
7. Nos dirigimos a la ventana izquierda Resource Management.
   - En la pestaña Tree View
   - Pulsamos el botón Generate

   NOTA: Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Sources Files)
8. Abrimos el software Arduino IDE, descargamos e instalamos la librería para el sensor DHT_22 y realizamos el programa para la comunicación del sensor y el envío de datos por el puerto serial.
9. Realizar un análisis de trama de datos usando un sniffer.

	10. Conectamos los dos dispositivos y vemos los datos en consola en S.A.D
	11. Revisar los datos capturados en sniffer y enviar los mismo mediante un programa de sockets por el puerto 1700 al servidor de red del sistema SAD.

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Protoboard
3. (1) Snap
4. (1) Arduino nano
5. (1) resistencia de 5.2 kohm
6. (1) sensor DHT_22
7. (1) Cable de red
8. (10) Conectores (hembra-macho)

ESQUEMAS

Esquema de la práctica:

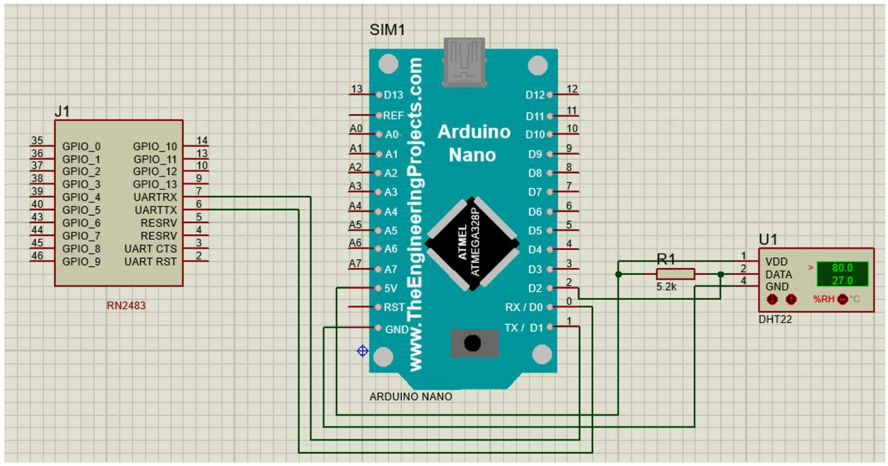
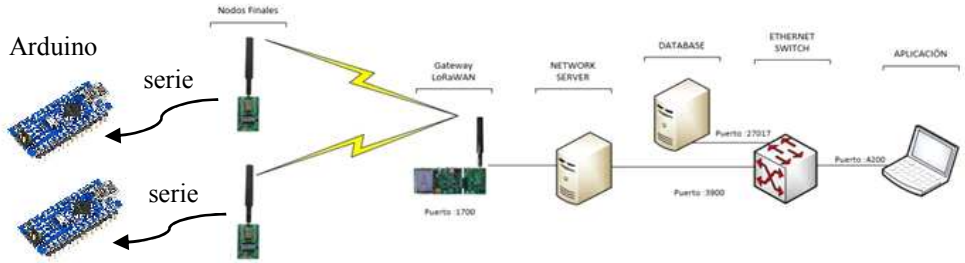


Figura: Comunicación UART de RN2483 y Arduino

Esquema de la red:



ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)
CONCLUSIONES:
BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice P: Configuración LoRaWAN en PIC18LF46k22 y revisar el Apéndice S: Envío de datos mediante sockets a servidor de red.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN**

CARRERA:		**ASIGNATURA**:
NRO. PRÁCTICA:	20	**TÍTULO PRÁCTICA**: Comunicación UART entre LoRa MOTE y Arduino nano para él envió de datos del sensor HC-SR04

OBJETIVO:

- **Configuración de la librería de encriptación para el MCU de 8-bits**

INSTRUCCIONES:

1. Iniciar MongoDB, Backend y FrondEnd
2. Iniciar el Sistema de Adquisición de Datos (S.A.D)
3. Iniciar el software MPLAB X IDE y abrir la práctica 17
4. Abrimos el MCC ya creado
   - En la pestaña Projects
   - Important Files
   - Hacer doble clic en nombre_archivo.mc3
5. Añadimos el periférico EUSART 1
   - **Mode:** asynchronous
   - **Enable EUSART:** checked
   - **Baud Rate:** X
   - **Transmission Bit:** 8-bit
   - **Reception Bits:** 8-bit
   - **Data Polarity:** async_noninverted
   - **Enable Recevie:** checked
6. En la ventana Pin Manager, cambiamos el estado a salida al pin RC7 del módulo EUSART1 RX1
7. Nos dirigimos a la ventana izquierda Resource Management.
   - En la pestaña Tree View
   - Pulsamos el botón Generate

   NOTA: Al generar se crean archivos de cabecera (Header Files) y los archivos fuente (Sources Files)
8. Abrimos el software Arduino IDE, realizamos el programa para la comunicación del sensor y el envió de datos por el puerto serial.
9. Realizar un análisis de trama de datos usando un sniffer.
10. Conectamos los dos dispositivos y vemos los datos en consola en S.A.D
11. Revisar los datos capturados en el sniffer y enviar los mismo mediante un programa de sockets por el puerto 1700 al servidor de red del sistema SAD.

Herramientas

Herramientas necesarias para realizar la práctica.

1. (1) Tarjeta LoRa MOTE
2. (1) Protoboard
3. (1) Snap
4. (1) Leds

5. (1) resistencia de 220ohm
6. (10) Conectores (hembra-macho)

ESQUEMAS

Esquema de la práctica:

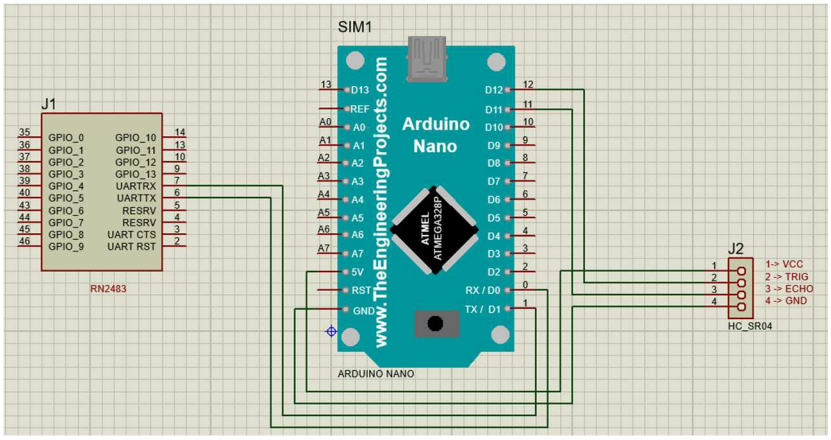
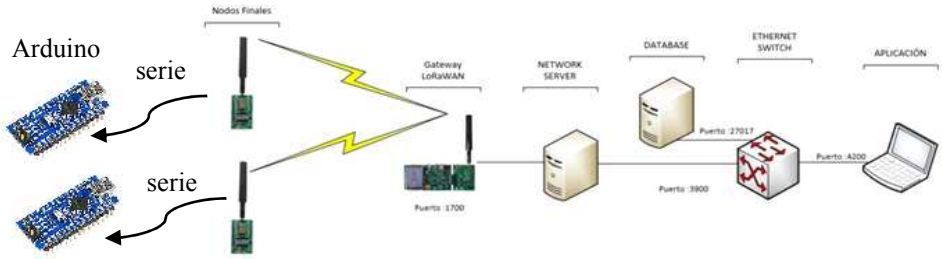


Figura: Comunicación UART de RN2483 y Arduino

Esquema de la red:



ACTIVIDADES DESARROLLADAS

(Anotar las actividades que siguió para el desarrollo de la práctica)

CONCLUSIONES:

BIBLIOGRAFIA:

Para su desarrollo revisar el Apéndice P: Configuración LoRaWAN en PIC18LF46k22 y revisar el Apéndice S: Envío de datos mediante sockets a servidor de red.

ANEXO 2: COMUNICADO DE ARCOTEL - BANDAS DE FRECUENCIAS

MIRANDA GRIJALVA HAROLD ESTUARDO 21 de junio de 2020 a las

<harold.miranda@arcotel.gob.ec> 21:05

Para: "fxvl4712@gmail.com" <fxvl4712@gmail.com>

CC: PAREDES MOLINA WILLIAM DAVID <david.paredes@arcotel.gob.ec>

Estimado,

El Plan Nacional de Frecuencias vigente establece:

EQA.40 Las bandas 450 – 470 MHz, 698 – 806 MHz, 824 – 849 MHz, 869 – 915 MHz, 940 – 960 MHz, 1427 – 1518 MHz, 1710 – 1780 MHz, 1850 – 1910 MHz, 1930 – 1990 MHz, 2110 – 2180 MHz, 2500 – 2690 MHz y 3300 – 3600 MHz se han identificado para su utilización por parte de las Telecomunicaciones Móviles Internacionales (IMT) de conformidad con las Resoluciones 212, 223, 224 (Rev.CMR15) y las notas internacionales aplicables a cada banda.

EQA.45 En las bandas 915 – 928 MHz, 2400 – 2483,5 MHz, 5150 – 5350 MHz, 5470 – 5725 MHz y 5725 – 5850 MHz y 24,05 – 24,25 GHz operan, a título secundario, sistemas que ocupan espectro radioeléctrico para Uso Determinado en Bandas Libres (UDBL), para los servicios fijo y móvil.

La normativa vigente establece diferencias entre espectro de uso libre y espectro para uso determinado en bandas libres.

Adjunto remito para su referencia, la NORMA TÉCNICA DE ESPECTRO DE USO LIBRE Y DE ESPECTRO PARA USO DETERMINADO EN BANDAS LIBRES.

Saludos cordiales,

Ing. Harold Miranda G., Mgtr.

Dirección Técnica de Regulación del Espectro Radioeléctrico

Agencia de Regulación y Control de las Telecomunicaciones

Av. Diego de Almagro N31-95 entre Whymper y Alpallana

Telf.: +(593) 2 294 7800 ext. 2656

Quito - Ecuador

AGENCIA DE REGULACIÓN Y CONTROL
DE LAS TELECOMUNICACIONES