ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELECTRÍCA Y ELECTRÓNICA

INGENIERIA EN TELECOMUNICACIONES

IMPLEMENTACIÓN DE UN PROTOTIPO DE ALARMA COMUNITARIA EN EL BARRIO DE CHILLOGALLO EN EL SUR DE QUITO CON TECNOLOGÍA SIGFOX

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES

WILSON IVAN TAPIA BASTIDAS

wilson.tapia01@epn.edu.ec

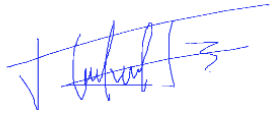
DIRECTOR: CARLOS ROBERTO EGAS ACOSTA

carlos.egas@epn.edu.ec

DMQ, octubre 2022

DECLARACIÓN

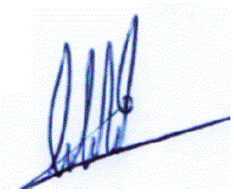
Yo, Wilson Iván Tapia Bastidas declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.



WILSON IVAN TAPIA BASTIDAS

CERTIFICACIÓN

Certifico que el presente trabajo de integración curricular fue desarrollado por Wilson Iván Tapia Bastidas, bajo mi supervisión.



M.Sc. Carlos Roberto Egas Acosta

DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el producto resultante del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Wilson Iván Tapia Bastidas

M.Sc. Carlos Roberto Egas Acosta

DEDICATORIA

A mi familia, mis Padres y hermana que apoyaron cada una de mis decisiones, me dieron su confianza, y la constancia de fortalecerme para seguir creciendo como persona, ya que, he ganado muchas virtudes y destrezas que me han ayudado a elegir todo lo que es correcto y a su vez brindar lo aprendido a quien lo requiera.

Para aquellos que contribuyen en la información necesaria que a su vez sirve para una guía de las personas que están comenzando su proyecto de grado.

A todos los expertos en el área asignada que, por su dedicación y su habilidad para la enseñanza, ahora existen varios profesionales que aportan en el crecimiento de la sociedad, a los técnicos que se siguen preparando y brindando sus conocimientos para más jóvenes que se incorporan a la vida universitaria.

AGRADECIMIENTO

Principalmente agradezco a Dios que guía cada paso de fortaleza, dedicación, cuidado, y la habilidad que me brinda para continuar preparándome. La sabiduría de observar el camino correcto y darme el ánimo para no desfallecer ante adversidades.

A mis padres Luis Tapia y Yolanda Bastidas que, con sus enseñanzas, sacrificio, entereza, la habilidad para darme fuerzas y que gracias a ellos no me ha faltado nada, he logrado ser una persona con la debida preparación, y poder enfrentarme a varios problemas cotidianos y resolverlos. A mi hermana Myrian Tapia por su compañía y crecimiento juntos, tanto en la laboral como personal, compartir el lazo tan grande para superar todos los obstáculos que se han presentado.

A mi tutor Carlos Egas que me dio la oportunidad de poder realizar el presente proyecto, dedicando su tiempo, explicaciones, recomendaciones y constante seguimiento ante el análisis y resultados que se exponen.

A las personas que compartieron cada jornada Universitaria, por todos los años impartiendo conocimiento de manera conjunta, mis compañeros de clase, que fueron ese sustento del día a día al estar mayor parte del tiempo con ellos, se aprende varias situaciones en las cuales se puede contribuir, y así conocer de cada uno.

RESUMEN

Debido a la constante inseguridad que se presenta en hogares de la ciudad de Quito, la ineficiencia de las unidades de vigilancia al no constatar en qué lugar específico se da el conflicto. La mejor solución es integrar un sistema de aviso pronto que permitirá realizar la visita óptima ante los hechos.

En el presente proyecto se desarrolló un prototipo de sistema de alarma comunitaria, compuesta de un módulo principal Sipy y como dispositivo secundario el chip ESP8266. El sistema se accionará al presionar un pulsador incorporado en los pines de adecuado acceso del módulo Sipy que es triple portador, ya que, consta de wifi, bluetooth y acceso a la red Sigfox, y que se encargará del envío de datos al backend que es la interfaz propia de Sigfox donde se interpreta cada uno de los mensajes de forma ascendente, adicional a esto el módulo principal proporciona el enlace vía wifi a nuestro chip ESP8266 que actúa como dispositivo secundario que posteriormente activará la bocina. Los datos que fueron enviados del módulo Sipy al backend corresponden a las coordenadas de altitud y latitud del lugar en conflicto, dichos datos se representan en la interfaz de visualización para el usuario llamada Ubidots, que mediante un Dashboards adaptable, mostrará el mapa de la ubicación exacta del lugar donde fue presionado el pulsador, que será presentada en un computador o dispositivo móvil.

PALABRAS CLAVE: Sigfox, backend, Ubidots, Dashboards.

ABSTRACT

Due to the constant insecurity that occurs in homes in the city of Quito, the inefficiency of the surveillance units by not verifying in which specific place the conflict occurs. The best solution is to integrate an early warning system that will allow the optimal visit to be made in light of the facts.

In this project, a prototype of a community alarm system was developed, consisting of a main Sipy module and the ESP8266 chip as a secondary device. The system will be activated by pressing a button incorporated in the appropriate access pins of the Sipy module, which is a triple carrier, since it consists of Wi-Fi, Bluetooth and access to the Sigfox network, and which will be in charge of sending data to the backend, which is Sigfox's own interface where each of the messages is interpreted from the bottom up, in addition to this the main module provides the link via Wi-Fi to our ESP8266 chip that acts as a secondary device that will later activate the horn. The data that was sent from the Sipy module to the backend corresponds to the altitude and latitude coordinates of the place in conflict, said data is represented in the visualization interface for the user called Ubidots, which through an adaptable Dashboards, will show the map of the location exact location where the button was pressed, which will be presented on a computer or mobile device.

KEYWORDS: Sigfox, backend, Ubidots, Dashboards.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 INTRODUCCIÓN

El gran avance tecnológico que se presenta en las últimas décadas ha contribuido en la facilidad de comunicarse globalmente, la gran cantidad de elementos automatizados que permiten recopilar y analizar datos a distancia, referentes a la humedad, temperatura, domótica, control de viviendas, luminancia. Etc. Por esa razón se debe implementar tecnologías que proporcionen eficiencia, cobertura extensa, costo bajo, seguridad, etc. Existen redes que ya lo hacen y generan este tipo de alternativas, sin embargo, Sigfox es la más viable al ser una red LPWAN que facilita cada característica de manera óptima, dado que, utiliza una modulación de banda estrecha de 100 Hz, lo cual es una ventaja ya que el ruido en el canal de comunicación será mínimo. Las conexiones inalámbricas han sido hoy en día de uso cotidiano para enlazar ciudades, países y continentes, proporcionando un gran alcance en el desarrollo.

Con el pasar de los años, la variedad de componentes que registran las personas es cada vez mayor, debido a que se presenta gran uso de las redes inalámbricas, y que, dependiendo de su infraestructura, se requiere de componentes compatibles con Sigfox, varios de estos al conectarse a la red consumen una cantidad mínima de energía y operan a grandes distancias. A medida que avanza la tecnología se busca alternativas para la comunicación de varios dispositivos en un solo entorno de desarrollo. En la actualidad debido a varios inconvenientes de inseguridad se plantean mecanismos de rastreo, monitoreo y control, los cuales son de gran ayuda para la sociedad, sin embargo, implementar este tipo de módulos registran también deficiencias en su funcionalidad ya que, trabajan con internet de forma local, no manejan protocolos de seguridad, costo muy alto, transferencia de datos alta, y consumo de energía extensa. El elegir un componente que presente características autónomas y eficientes en varias ocasiones es muy complicado por programadores, debido a desconocimiento de alternativas viables. El dispositivo Sipy proporciona tres componentes en un solo, ya que consta de Wifi, Bluetooth y acceso a red Sigfox, por esa razón es de gran utilidad para el presente proyecto.

Dada la problemática de varios sucesos delincuenciales en la ciudad capital se trata el análisis del módulo Sipy y la red Sigfox que, en conjunto con el lenguaje de programación adecuados, se puede enviar datos a la nube propia de Sigfox y decodificarlos en plataformas auxiliares que permitan ser visibles para los usuarios. La utilización de interfaces de salida de datos proporciona soluciones más amigables, y que pueden ser ilustradas en el propio dispositivo móvil o computador.

1.1 Objetivo general

Implementar un prototipo de alarma comunitaria inteligente para hogares en conflicto, utilizando la red Sigfox.

1.2 Objetivos específicos

1. Estudiar la base teórica necesaria para realizar la implementación del prototipo.
2. Realizar el registro del dispositivo Sipy en la red de Sigfox para el envío de datos.
3. Implementar el pulsador para la activación de la alarma comunitaria.
4. Diseñar y estudiar un prototipo de red con el chip ESP8266 para la activación vía wifi de la alarma.
5. Diseñar y construir las PCB para integración en los módulos de control.
6. Programar la transferencia de datos entre la red Sigfox y la plataforma Ubidots para tener la interfaz gráfica de usuario.
7. Realizar las pruebas para validar la integración de los componentes y la funcionalidad del prototipo.

1.3 Alcance

Se implemento un sistema de alarma inteligente, que se acciona al presionar un botón, el cual está instalado en el mecanismo que contiene al módulo Sipy, esté se activa y envía información a la nube Sigfox y mediante comunicación inalámbrica se realiza el enlace vía URL e IP al módulo ESP8266 que posteriormente activa la alarma.

Al momento que se activa el módulo Sipy se envía datos de ubicación específica del lugar en conflicto, y son mostrados en la interfaz de visualización para el usuario llamada Ubidots, la cual permite observar en un mapa virtual de tiempo real las coordenadas del domicilio.

El lugar donde se instale el mecanismo es de fácil acceso e ilustrando las normas en las cuales se debe presionar el botón no se presentará inconvenientes en el desarrollo. El sonido que emite la alarma tiene una duración ilimitada y se desactiva una vez que se presione nuevamente el botón de interrupción.

Con los datos que se presentan en el dispositivo final como celular o computador y mediante la notificación enviada a las unidades de Vigilancia, se espera la ayuda pronta para brindar la solución óptima.

1.4 Marco teórico

1.4.1 Tecnología Sigfox

La Tecnología Sigfox se basa en una red inalámbrica de la familia de tecnologías LPWAN, y está dedicada a aplicaciones que requieren transmisión de datos con un consumo de potencia muy bajo y que puedan cubrir grandes distancias. La red Sigfox está desplegada a nivel mundial con un gran alcance al momento del envío de información, todos los mensajes que llegan pueden ser reenviados hacia otros servidores propietarios, mediante enlaces o conectores con otras plataformas en la nube.

1.4.2 Características de la tecnología Sigfox

Sigfox presenta una variedad de características que optimizan el envío de información, estás se presentan a continuación:

Se basa en una transmisión de radio que utiliza la tecnología UNB (del inglés: Ultra Narrow Band), con modulación D-BPSK (del inglés: Differential Binary Phase-Shift Keying), gracias a esta modulación y banda estrecha la información enviada presenta mayor inmunidad al ruido.

Presenta bandas de frecuencia no licenciadas ISM (del inglés Industrial, Scientific and Medical), las cuales no requieren de autorización para ser utilizadas.

Para la trasmisión de datos el consumo de energía es bajo, en el orden de 10 a 50 miliamperios, y en estado de reposo tiende a cero.

Al utilizar UNB, funciona a bajas velocidades con una transferencia de datos de a 10 a 1000 bps, además que se basa en una transmisión de canal de espectro muy estrecho menor a 1 KHz, esto proporciona un alcance inalámbrico de 20 km en lugares abiertos y 1.5 km en lugares urbanos, en la figura 1 se presenta un esquema del alcance a varios dispositivos en cientos de kilómetros. [1]

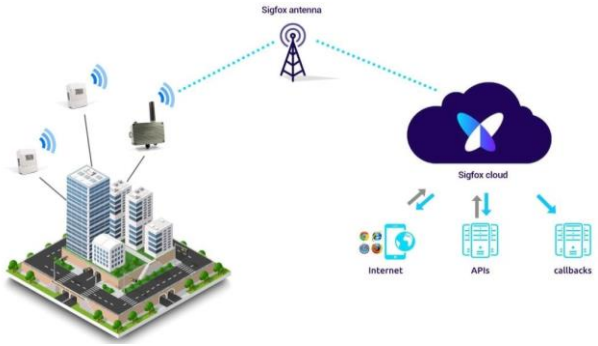


figura 1.1 Funcionamiento de la red Sigfox en lugares urbanos, proporcionando alcances inalámbricos extensos.

1.4.3 Cobertura Sigfox

Para que se establezca un enlace entre los dispositivos que son compatibles con la red Sigfox, se debe tener en cuenta la cobertura a nivel mundial, en la siguiente figura se visualiza la cobertura que existe en varios lugares del mundo, marcado de color celeste. [2]

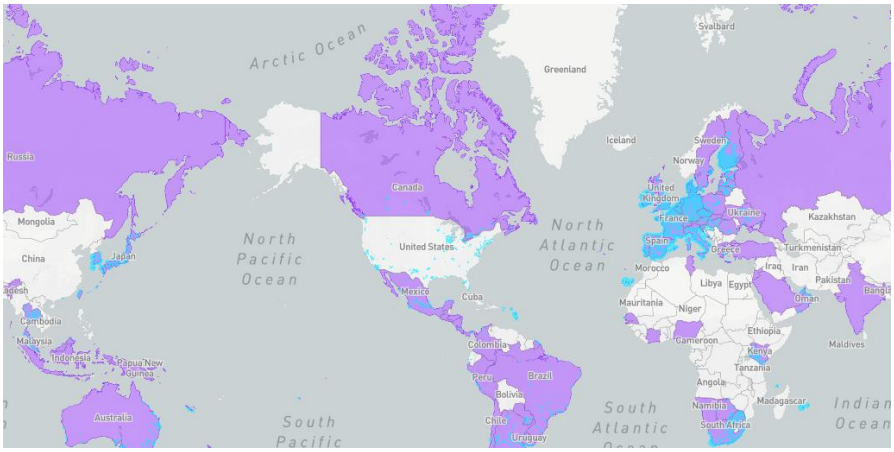


figura 1.2 Cobertura que proporciona Sigfox a nivel mundial.

1.4.3.1 Cobertura Sigfox a nivel local

Existe una gran cobertura que proporciona Sigfox a nivel nacional, en la ciudad capital existe un amplio despliegue, por lo que no se presentará inconveniente para el envío de información a largo alcance, se debe tomar en cuenta que en todos los lugares no se establece cobertura, ya que la colocación de antenas base debe ser estratégico para que se presente una óptima comunicación entre los mensajes enviados y la nube de Sigfox.

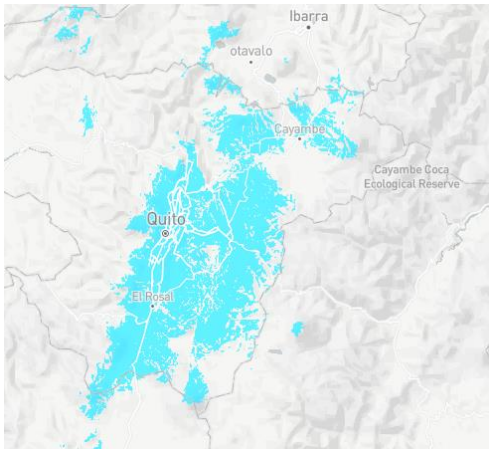


figura 1.3 Cobertura en la ciudad de Quito.

1.4.4 Modo de operación Sigfox

La tecnología Sigfox se basa en estaciones base que constantemente están monitoreando el mensaje uplink por parte de dispositivos sensores que a su vez sean compatibles con Sigfox, estas estaciones posteriormente envían los datos transmitidos a la nube de Sigfox (backend), en el siguiente esquema se puede visualizar el modo de operación relacionado.

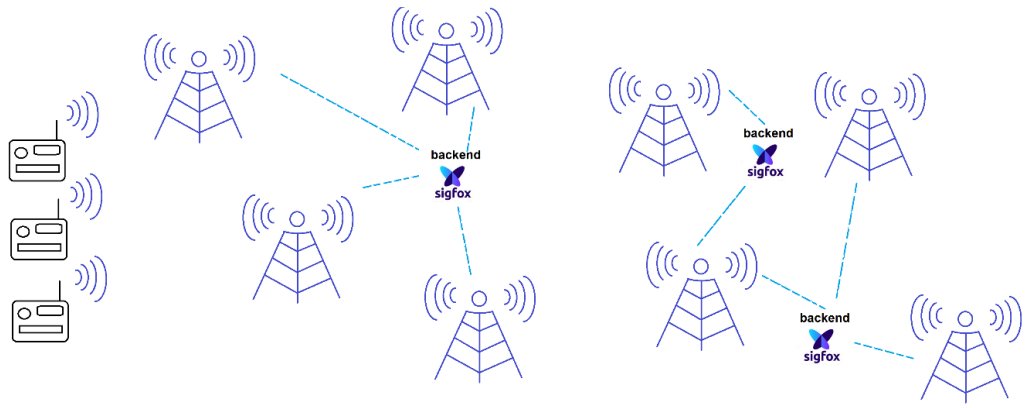


figura 1.4 Modo de operación de las antenas base, que se encuentran en modo espera hasta recibir los mensajes enviados por dispositivos, para poderlos subir a la nube de Sigfox (backend).

1.4.4.1 Configuración de la zona de radio en Sigfox

Sigfox al estar desplegada en más de 72 países, proporciona un enlace dispositivo-red según el país en cada continente, el dispositivo que sea compatible con Sigfox utiliza bandas de frecuencia no licenciadas ISM, para el Sur de América que es la banda de interés es 950 MHz, a continuación, se presenta la configuración de radio (RC) según el país de estudio. [3]

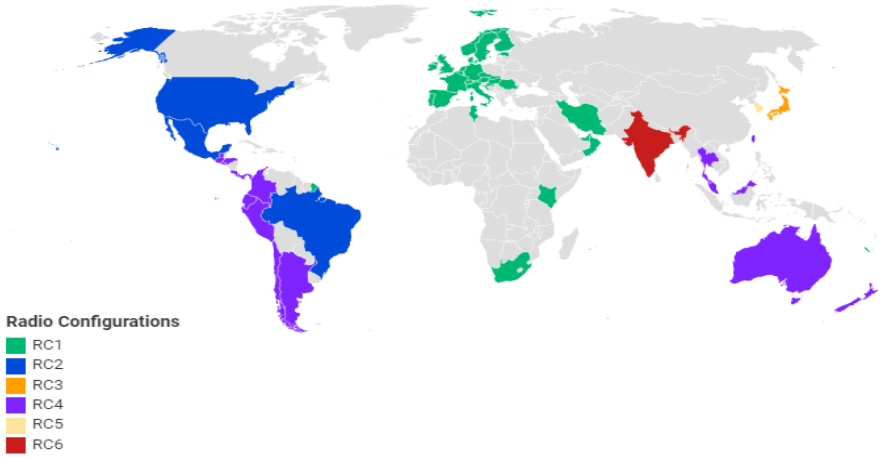


figura 1.5 Países que cuentan con red Sigfox según el continente.

Las bandas de frecuencia según su color y país de análisis (Ecuador) es lila, por lo tanto, se tiene la RC 920 MHz.

Configuración de radio	Frecuencias
RC1	868 MHz
RC2	902 MHz
RC3	923 MHz
RC4	920 MHz
RC5	923 MHz
RC6	865 MHz

Tabla 1.1 Parámetros que se recomiendan para cada continente, según la zona específica utilizada en la tecnología Sigfox.

1.4.5 Estructura de la Trama Sigfox

Los mensajes que se envían en Sigfox están diseñados para que sean pequeños y que requieran un uso mínimo de energía al ser transmitidos.

La trama Sigfox consta de un preámbulo que establece la sincronización de los mensajes uplink, con una extensión de 4 de bytes.

La trama de sincronización que consta de 2 bytes, que es especifica el tipo de trama.

El ID, que nos indica el identificador de dispositivo, presenta 4 bytes.

Payload, que es el campo de la trama que indica el mensaje que se encuentra en el dispositivo, el cual envía los datos a ser transmitidos, consta de 12 bytes los cuales se presentan de manera hexadecimal en la nube (backend) de Sigfox, y dependiendo la extensión de los mensajes pueden ser utilizados todos en forma óptima.

El FCS que especifica la secuencia de verificación de trama, y tiene una extensión de 2 bytes.

A continuación, se presenta la estructura de la trama Sigfox. [4]

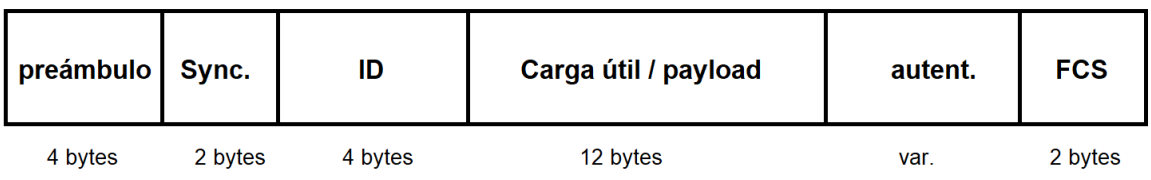


figura 1.6 Trama de datos Sigfox.

1.4.6 Backend de Sigfox

Sigfox establece una comunicación de extremo a extremo, el primero que es el dispositivo que envía información, la cual se enlaza a las antenas base, estás demodulan los datos y los envían al segundo extremo, el cual es final y se denomina backend, esté recepta los datos y son visualizados en una interfaz de usuario de manera hexadecimal en duplas de bytes.

El acceso al backend de Sigfox se logra mediante la actualización de firmware en el dispositivo compatible con Sigfox, esté se registra y proporciona un ID y un PAC únicos del dispositivo, los cuales dan uso de la interfaz de usuario.

figura 1.7 Interfaz de usuario, página principal de acceso al backend de Sigfox.

En la interfaz de usuario se puede observar características esenciales como el tipo de dispositivo registrado, localización del dispositivo, estado en el que se encuentra, mensajes que llegaron, callbacks, etc.

1.4.6.1 Mensajes en el backend Sigfox

Los mensajes enviados del dispositivo emisor al backend son previamente cargados mediante el lenguaje de programación que sea compatible con el mismo, una vez que llegan al backend se receptan de manera hexadecimal, los datos que se receptan se expresan en la interfaz de usuario según la siguiente figura:

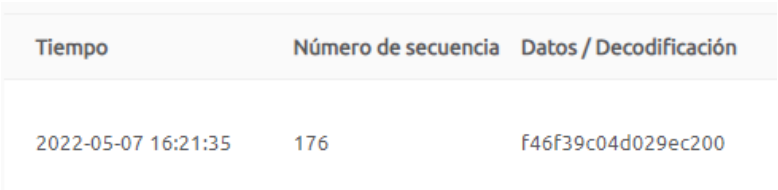


figura 1.8 Datos en el backend de Sigfox de manera hexadecimal en duplas de bytes.

Los datos que se visualizan son coordenadas de altitud y latitud de pruebas realizadas para verificar el enlace con el dispositivo y el backend de Sigfox.

1.4.7 Callbacks (Devolución de llamada)

Los callbacks son notificaciones que se activan cuando el dispositivo emisor envía un mensaje, el backend al recibir los datos, automáticamente responde con una devolución de llamada (callbacks). Se presenta tres tipos de callbacks, los cuales son descritos en la siguiente tabla.

TIPOS DE CALLBACKS	ANÁLISIS DE TIPOS	CARACTERÍSTICAS
DATOS	ENLACE ASCENDENTE	Los datos que son enviados del dispositivo son receptados en el backend, y luego publicados en servidor compatible.
	BIDIR	Además, que los datos son enviados en forma ascendente, se responde con mensaje de manera descendente desde el servidor. (bidireccional)
SERVICIO	ESTADO	En el estado se establece información acerca del dispositivo, también conocidos como mensajes de mantenimiento.
	RECONOCER	Trabaja como mensaje de verificación, el cual es enviado desde el destino hacia al origen, para confirmar la recepción.
	DATOS AVANZADOS	Más utilizado para mensajes de geolocalización.
ERROR	Se informa en la red o en el dispositivo si existió un corte en la comunicación.	

Tabla 1.2 Tipos de datos en la devolución de llamada en Sigfox.

1.4.7.1 Representación de la interfaz de devolución de llamada en el backend de Sigfox

Los parámetros que son enviados del dispositivo y llegan al backend, se configuran de acuerdo con lo que se solicite, pueden valores de enteros, flotantes y caracteres, y se los puede modificar en la siguiente interfaz de usuario. [5]

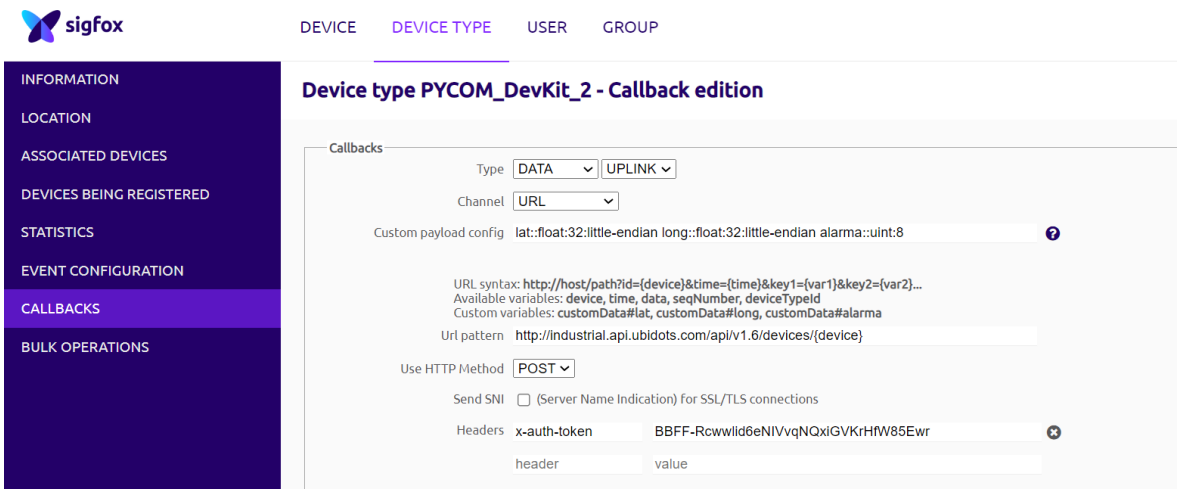


figura 1.9 Interfaz gráfica, representa los parámetros de la devolución de llamada.

1.4.8 Módulo Sipy

Pycom que es una empresa que se basa en la fabricación de módulos de última generación los cuales provén varias alternativas de conectividad, entre algunos se tiene a Sipy, Lopy y Fipy, todos los mencionados tienen características de conexión a extensas redes, el enfoque que se realiza será en el módulo Sipy, por su costo, facilidad de acceso al mercado, y conexión a múltiples redes. [6]

El módulo Sipy es un microcontrolador que opera en múltiples redes, como son Wifi, Bluetooth y Sigfox, también conocido como triple portador, orientado hoy en día a variedad de aplicaciones que dan parte a soluciones de IoT. La programación requerida es MicroPython, el cual es un lenguaje basado en Python específicamente para microcontroladores, además, consta de un Chipset Espressif ESP32 que proporciona un mayor alcance en la red Wifi y Bluetooth de ser necesario.

Presenta características esenciales como consumo mínimo de energía en el orden de los 25 microamperios (uA), en la transmisión de datos 120 mA, en la recepción de datos 12 mA y en modo sleep 0.5 uA. [7]

1.4.9 Características generales del Módulo Sipy

Característica	funcionalidad
Conectividad	Wifi, Bluetooth, Red Sigfox
Puertos de entrada y salida GPIO	24 puertos
Voltaje	3.3 Voltios – 5.5Voltios
Potencia de transmisión	22 dBm
Frecuencia de operación	920 MHz-922 MHz

Tabla 1.3 Generalidades del módulo Sipy, correspondientes a Sudamérica.

1.4.9.1 Partes esenciales del Módulo Sipy

Botón reset: Reinicia el módulo en modo de operación, si presenta errores en la programación se restablece al presionar el botón reset.

RGB LED: En modo de operación indica un correcto funcionamiento, al palpitar cada segundo de color característico azul, así se puede comprobar si la conexión se estableció de manera óptima, y se procede a programar.

Conector de antena Sigfox externa: Al conectar la antena externa, permite establecer una mejor comunicación con el backend de Sigfox, y actualizaciones previas al acceso.

ESP32: Es una serie de SoC (del inglés System on Chip), microcontrolador dual, que consta de Wifi y bluetooth, que opera a bajo costo y con consumo bajo de energía.

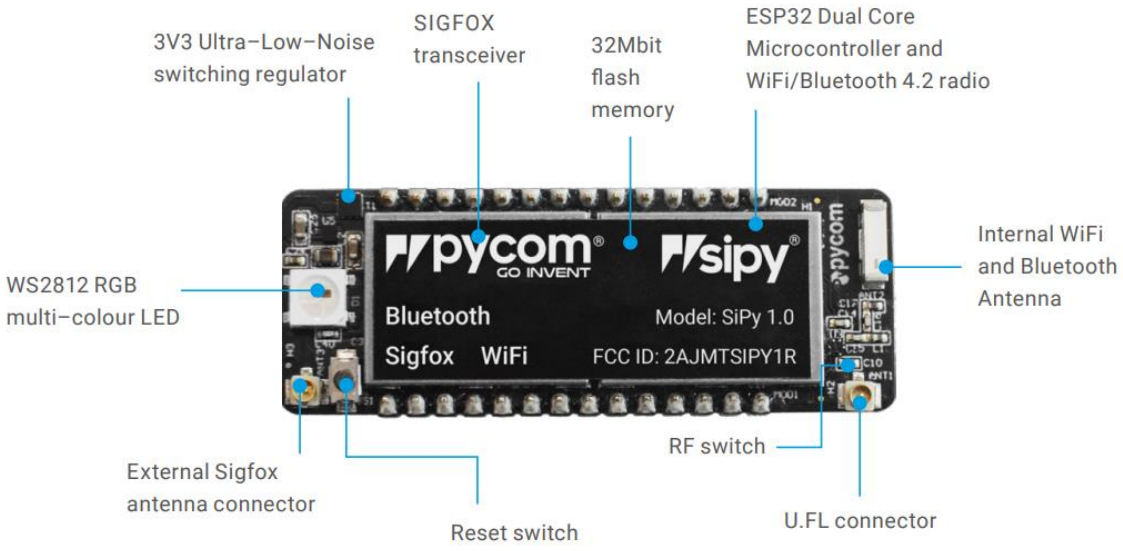


figura 1.10 Partes características del módulo Sipy

1.4.9.2 Pytrack

Es una placa de expansión para el módulo Sipy, al contener terminales únicos para el acceso y establecer la comunicación microcontrolador y máquina (computador), proporciona la conexión para la accesibilidad del cable USB, que permite grabar y subir la información que posteriormente se escribe en un lenguaje de programación compatible con el dispositivo. Mediante la Pytrack se puede establecer la actualización de firmware, que es de vital importancia para el registro de componentes Pycom, en el caso de análisis el dispositivo Sipy, además de proporcionar datos específicos como son el ID y el PAC para el acceso al backend de Sigfox. [8]

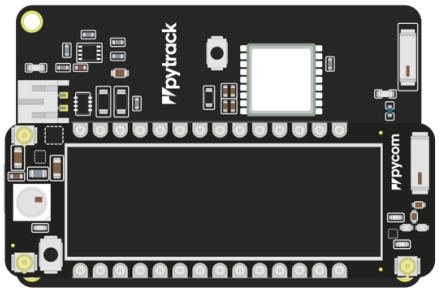


figura 1.11 Placa de expansión Pytrack

1.4.10 Módulo ESP8266

Es un chip que proporciona una solución autosuficiente en las redes Wifi, por su alta capacidad para integrarse a dispositivos externos mediante los GPIO (Pin de propósito general de entrada y salida), que permiten el envío y la recepción de datos a los pines de los extremos del componente emisor y posibilita el control inalámbrico a grandes distancias.

1.4.10.1 Características y especificaciones del módulo ESP

Característica	**Especificación**
Alimentación	3.3 V
Soporte de red	2.4 GHz
Stack	TCP/IP
Comunicación	Serial
Consumo de energía	Menor a 10 uA

Tabla 1.4 Especificaciones y características técnicas del ESP8266

El ESP se alimenta a 3,3 Voltios, no es de uso común, sin embargo, se puede colocar módulos de fuente conversores de tensión que proporcionen la salida de voltaje que es necesaria para el funcionamiento del módulo.

1.4.10.2 Distribución de pines en el ESP8266

En el ESP se utiliza los 6 pines de distribución:

- El pin 1 que es para la transmisión de datos (Tx), que se establece mediante comunicación serial con el dispositivo externo el cual presenta entrada USB y permite grabar el código que se programa en el IDE de Arduino.
- El pin 2 es GND o conexión a tierra.
- El pin 3 y pin 7 son la alimentación a 3,3 Voltios (VCC).
- El pin 4 y pin 6 son pines de entrada y salida de propósito general (GPIO).
- El pin 5 permite reiniciar (RST) el módulo.
- El pin 8 recepción de datos (Rx).

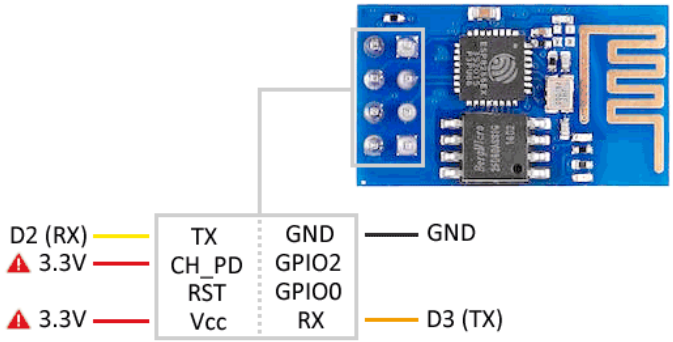


figura 1.12 Representación y especificación de cada Pin en el módulo ESP8266

1.4.11 Interfaz Ubidots

Debido a la cantidad de proyectos los cuales necesitan de una interfaz para la visualización de datos enviados por cualquier dispositivo a la nube, y luego ser proyectados, Ubidots proporciona la interpretación de información en su interfaz, mediante tableros (del inglés Dashboards), teniendo una lectura óptima por parte del usuario en el punto de recepción.

Ubidots minimiza el proceso de creación de aplicaciones, las cuales tendrían el mismo fin que una interfaz de salida, las circunstancias de análisis y tiempo de operación para proyectos hacen de Ubidots el panel que proporcione de una manera interpretativa y entendible la información enviada por mecanismos de origen.

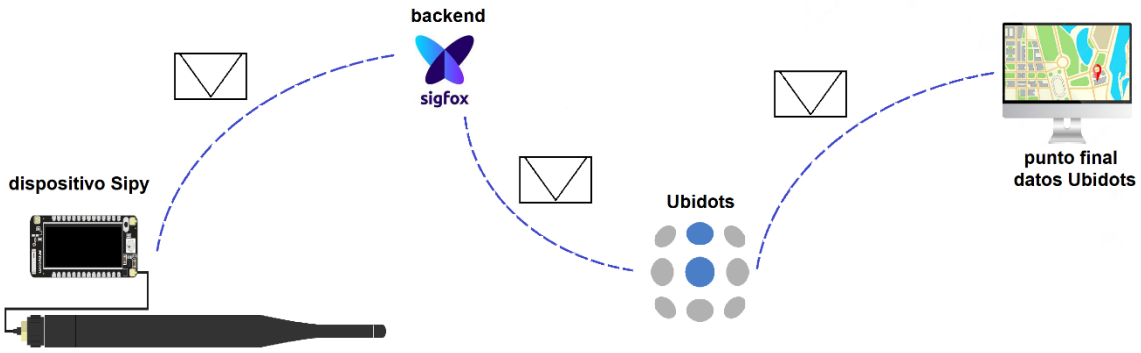


figura 1.13 Modo de operación en Ubidots, mediante un dispositivo inicial, backend de Sigfox, interfaz Ubidots y el mecanismo de visualización final (computador o celular).

1.4.11.1 Panel principal en Ubidots

En la interfaz de usuario Ubidots se presenta una variedad pestañas enlazadas a la necesidad del programador, como los dispositivos agregados, datos enviados por terceros, perfil de modificación de usuario y la fecha de emisión la cual se inicia el uso del panel principal.

figura 1.14 Representación del tablero y variables de demostración en Ubidots

1.4.11.2 Parámetros de operación en Ubidots

Para la adaptabilidad de cualquier componente externo es necesario contar con varios mecanismos de operación, ya sea para lectura e interpretación de información, la cual es requerida al momento de ser presentada en forma visible y entendible.

Los parámetros iniciales que deben ser configurados para el componente origen son los siguientes:

- Dispositivo: representa cualquier hardware o servicio de terceros que crea datos, estos son decodificados a través de la API de Ubidots en forma de variables.
- Datos: consta del acceso a una variedad de tableros que permiten ilustrar la información cualitativa y cuantitativa de la información receptada.

Los parámetros secundarios que deben ser configurados para el dispositivo y los datos son los siguientes:

- Widget: corresponde a la visualización específica de un dato enviado por el componente origen, se presenta en forma de tabla, planos cartesianos, medidas, mapas, gráficos de barra, indicadores, etc.
- Credencial de la API: permite la comunicación dispositivo-Ubidots, mediante el código fuente programado en el dispositivo primario.
- Token: es un identificador o serial único que se autentifica al crear la cuenta en Ubidots y que se enlaza a cada uno de los dispositivos registrados en análisis.
- Variables: una vez establecida el Widget, está pasa a ser variable y es visible en el tablero de Ubidots. La variable puede compartida mediante un enlace URL y ser extraíble en tiempo real a usuarios externos.

1.4.11.3 Conectividad en Ubidots

Ubidots enlaza varias tecnologías para la conectividad he interpretación de información enviada por terceros, como, por ejemplo, Ethernet, Lora WAN, LTE, Wifi y Sigfox.

1.4.11.4 Adaptabilidad a dispositivos en Ubidots

Es adaptable para una variedad de dispositivos, y con una gran facilidad al momento de manejar los Dashboards.

Según el fabricante y la tecnología de cada dispositivo, Ubidots integra cada tipo en una interfaz única para la interpretación de datos, el módulo en operación para el presente proyecto es Sipy de su fabricante Pycom.

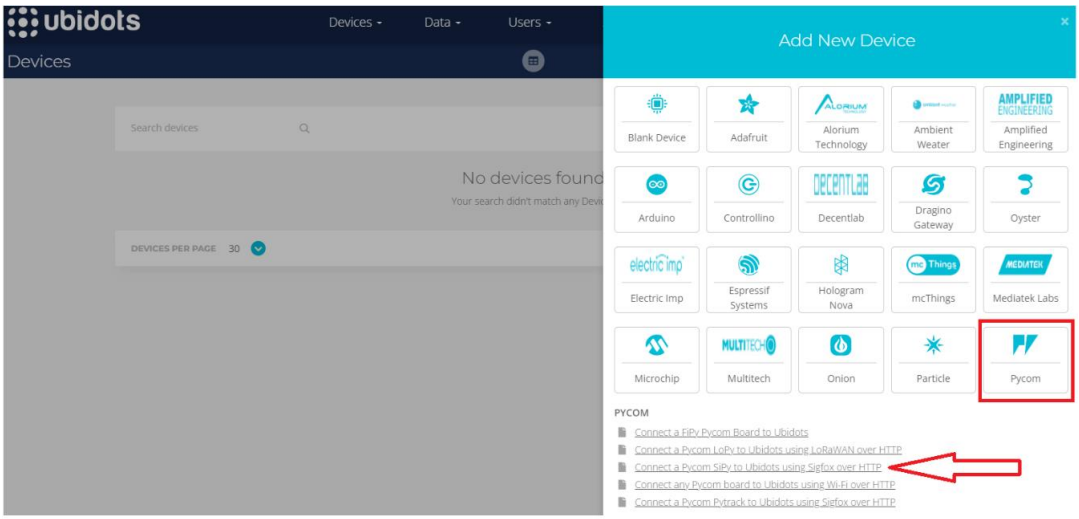


figura 1.15 Elección de tipo de dispositivo según el fabricante y modelo específico.

1.4.12 Módulo Fuente de voltaje

Es una fuente de energía universal que proporciona 3.3 y 5 voltios de alimentación a circuitos que requieren salidas de voltaje variada. Es muy útil para circuitos electrónicos especialmente con carácter digital con versiones de microcontroladores PIC y además que consta de una entrada USB con salida de voltaje de 5 V.

1.4.12.1 Especificaciones técnicas

Para los requerimientos del sistema como el módulo Sipy y el chip ESP8266 es necesario una salida de voltaje que proporcione varias alternativas de alimentación, a continuación, algunas características acerca del módulo fuente: [9]

Característica	Especificación
Voltaje de enterada	6.5 V y 12 V
Voltaje de salida	3.3 V y 5 V
Salida de tensión USB	5 V
Corriente máxima	500 mA
Diodo LED	Indicador de encendido y apagado

Tabla 1.5 Características técnicas del módulo de alimentación múltiple.

El módulo fuente de alimentación representa un gran avance para diversos proyectos de diseño de baja consumo de potencia. [10]

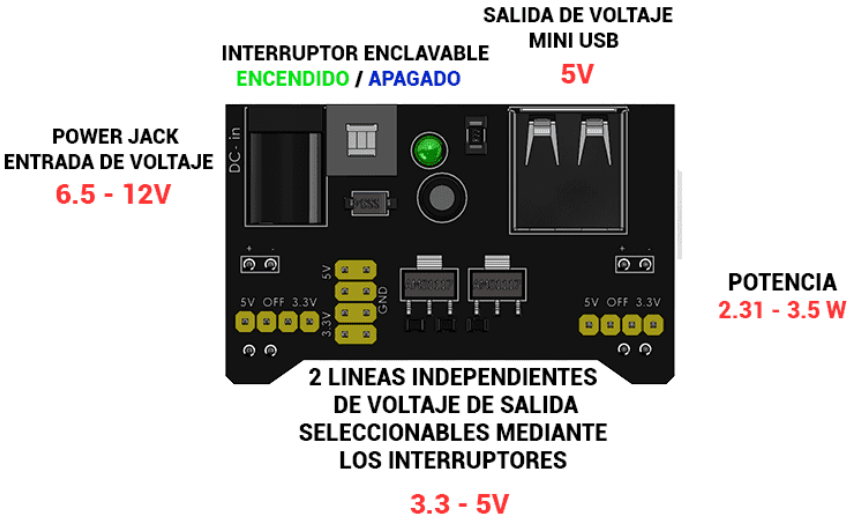


figura 1.16 Partes esenciales del módulo fuente para una salida de tensión a 3.3 V y 5 V

1.4.13 Módulo Relé

Se requiere de este módulo para el accionamiento de la alarma y como una protección de dispositivos de bajo consumo de potencia como los microcontroladores, y además soporta un máximo de 250 V, por lo que no se tendrá problema con la alimentación de 110 V para la alarma.

1.4.13.1 Representación de salidas y entradas del módulo Relé

Las características que cumple el Relé son las siguientes: [11]

Especificaciones	Características
Señal de control	5 V
Máxima AC	250 V
Máxima DC	30 V
Tipos de contacto	NO (normalmente abierto), NC (normalmente cerrado), C (común)

Tabla 1.6 Características técnicas del módulo Relé.

1.4.13.2 Componentes específicos del módulo Relé

Las partes más relevantes del módulo Relé se representan a continuación. [11]

figura 1.17 Salidas y Entradas específicas del módulo Relé y alimentación VCC a 5V

1.4.14 Motor de sirena

Adecuada para alertar a una gran distancia, fácil implementación y tamaño muy reducido para espacios específicos. Practica para proyectos de bajo costo, utilizada para señales de alerta, robos a vehículos, emergencia, ambulancias, etc. [12]

1.4.14.1 Especificaciones técnicas

Característica	Especificación
Nivel de ruido	75 dB
Modo de transmisión	Cable de alimentación

Alimentación	110 V
Corriente nominal	3 A

Tabla 1.7 Descripciones básicas del motor de sirena, utilizada como alarma para alertas de emergencia.

2 METODOLOGÍA

2.1 Autenticación del dispositivo en el backend de Sigfox

La variedad de dispositivos Pycom requieren una actualización previa y registro en la nube de Sigfox, para esto se proporciona:

- Registro del módulo Sipy en la red Sigfox
- Actualización de firmware y obtención de ID y PAC del dispositivo
- Creación de cuenta para el acceso a la red Sigfox

2.1.1 Registro del módulo Sipy en la red Sigfox

El registro de cualquier módulo Pycom se lo realiza desde la página oficial de Sigfox [13], al ser una red que requiere paga mensual, se debe activar la suscripción por un cierto lapso (un año), que proporciona al enlazar por primera vez el dispositivo a la red.

Presionamos en activar mi DevKit, como se muestra en la siguiente imagen:

figura 2.1 Página oficial Sigfox, activación de suscripción por un año.

Al seleccionar en activar mi DevKit, se redirecciona a una nueva página donde se muestra el país en el cual se establecerá el análisis, en el caso de estudio será Ecuador, como se muestra en la siguiente la figura, en la cual se despliega si se encuentra o no activo en el país de selección.

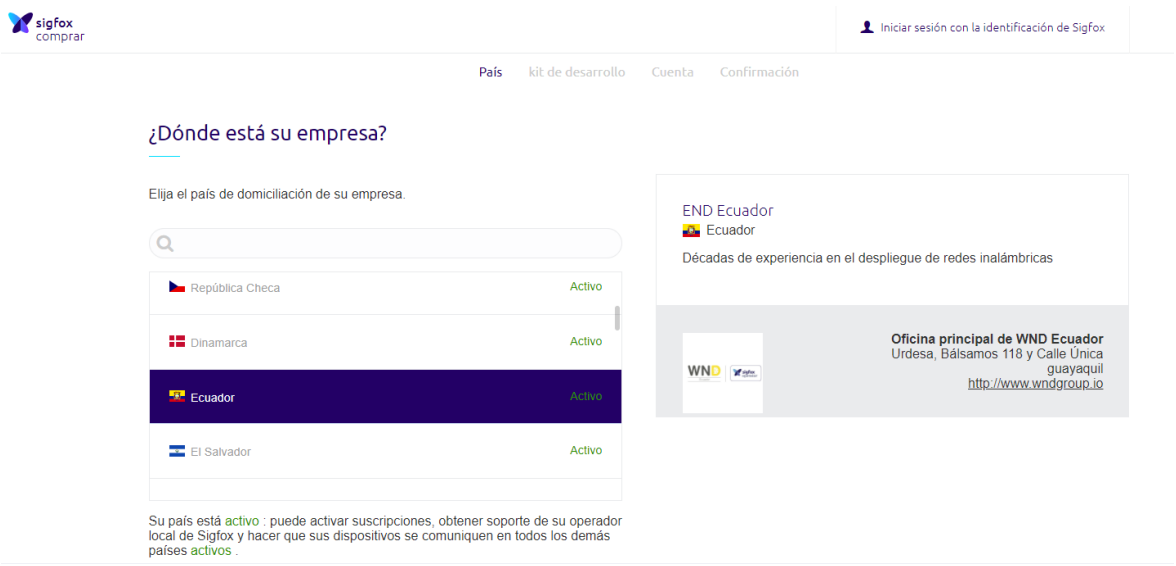


figura 2.2 Selección de país y actividad correspondiente.

Una vez que se establece el país de origen, se debe realizar la identificación del dispositivo con el ID y PAC, los cuales son únicos de cada módulo Pycom.

2.1.2 Actualización del firmware en el módulo Sipy

Las credenciales como el ID y PAC del dispositivo se obtienen con la actualización del firmware siguiendo algunos pasos característicos como son los siguientes:

- Descargar e instalar la herramienta de actualización [14], de acuerdo con el sistema operativo, para el caso de estudio será Windows.

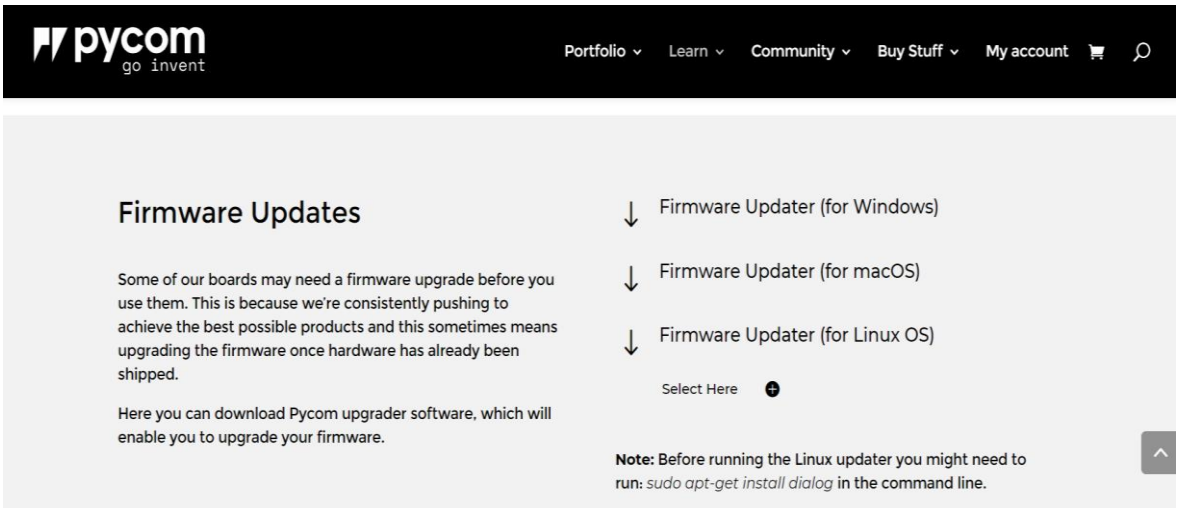


figura 2.3 Sistema operativo compatible con la actualización de firmware para el registro en el backend de Sigfox.

- Obtener de credenciales ID y PAC, previos a la actualización de firmware.

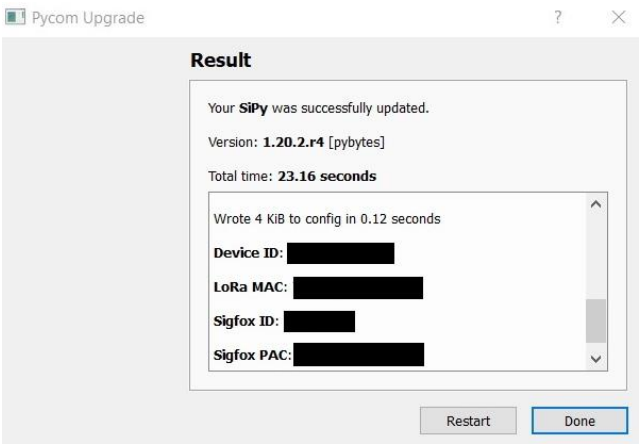


figura 2.4 Credenciales ID y PAC del dispositivo Sipy.

Es de vital importancia conectar la antena al terminal del módulo Sipy al momento que se está actualizando el firmware, esto permite validar los datos únicos como credenciales y posterior registro al backend de Sigfox.

2.1.3 Identificación del módulo Sipy mediante el ID y PAC

Cuando se obtiene el ID y PAC que son únicos del dispositivo Sipy, se puede continuar con el proceso de identificación, que nos permitirá el ingreso al backend de Sigfox, se coloca los datos solicitados y procedemos a presionar en siguiente, esto redirecciona al proceso de creación de cuenta, la cual nos establecerá el acceso al backend.

Las credenciales para el acceso al backend de Sigfox como usuario y clave, son creadas mediante el correo electrónico (usuario) que sea de nuestra autoridad, se identifica una clave con caracteres que se solicite y posteriormente se tiene dicho acceso.

figura 2.5 Colocación de ID y PAC previos a la obtención de credenciales únicas del dispositivo al actualizar el firmware.

2.1.4 Acceso al backend de Sigfox

Se coloca el usuario (correo electrónico que corresponda), y contraseña, estos datos nos permiten acceder a la interfaz principal del backend, donde se puede verificar, el estado del dispositivo, devolución de llamada (callbacks), mensajes uplink, recepción de datos que llegan del dispositivo según la programación establecida. Etc. [15]

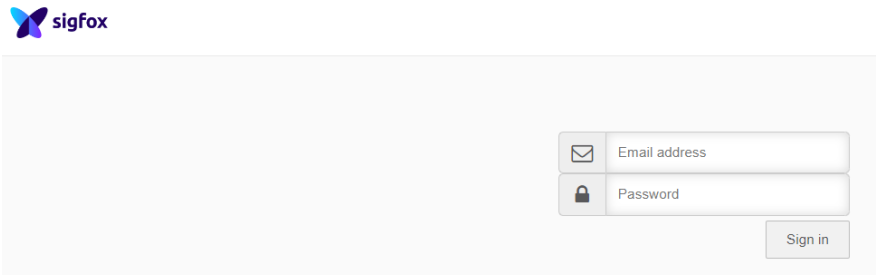


figura 2.6 Acceso al backend de Sigfox, colocación de credenciales (usuario y contraseña).

2.2 Instalación de Atom

Los módulos propios de Pycom tienen editores de código compatibles para cada uno de sus dispositivos, como, por ejemplo; Atom, Pybytes y Visual Studio Code. Atom y VS Code son editores gratis disponibles en las páginas oficiales y descargables según el sistema operativo que se tiene. Pybytes es un editor propio de los fabricantes de Pycom, que tiene una licencia y paga mensual.

Para programar el código fuente que proporciona el envío de datos e información respectiva se utilizará Atom, que mediante un complemento Pymark nos permite establecer el lenguaje compatible en MicroPython, que es de sencilla instalación y amigable para el código programable. [16]

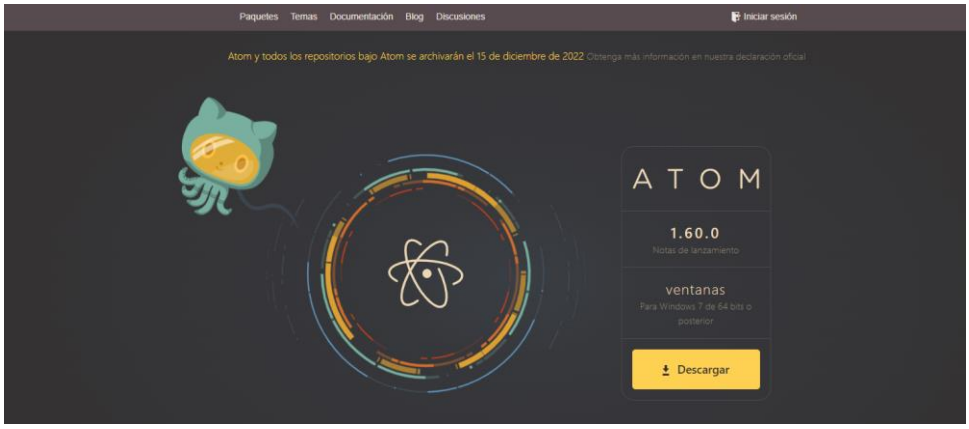


figura 2.7 Interfaz de usuario para descargar Atom según el sistema operativo, Windows, Ubuntu y Mac.

2.2.1 Instalación de complementos en Atom

Los complementos son descargables e instalables para el procesamiento de acuerdo con el lenguaje de programación, constan de una serie de librerías funcionales para el módulo del desarrollo, y una correcta lectura del código fuente.

2.2.1.1 Complemento Pymark

La instalación del plugin Pymark permite la compatibilidad del dispositivo Sipy con el lenguaje de programación en MicroPython, se instala siguiendo los pasos siguientes:

- Ingresar a la interfaz principal de Atom.

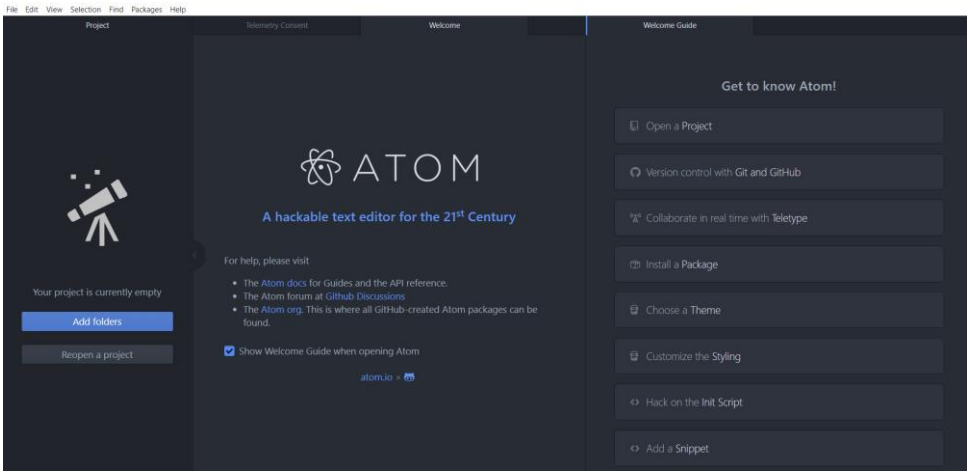


figura 2.8 Interfaz de usuario Atom

- Dirigirse a Settings, install y colocar Pymakr, previo a que aparezca la extensión, se procede a descargar e instalar.

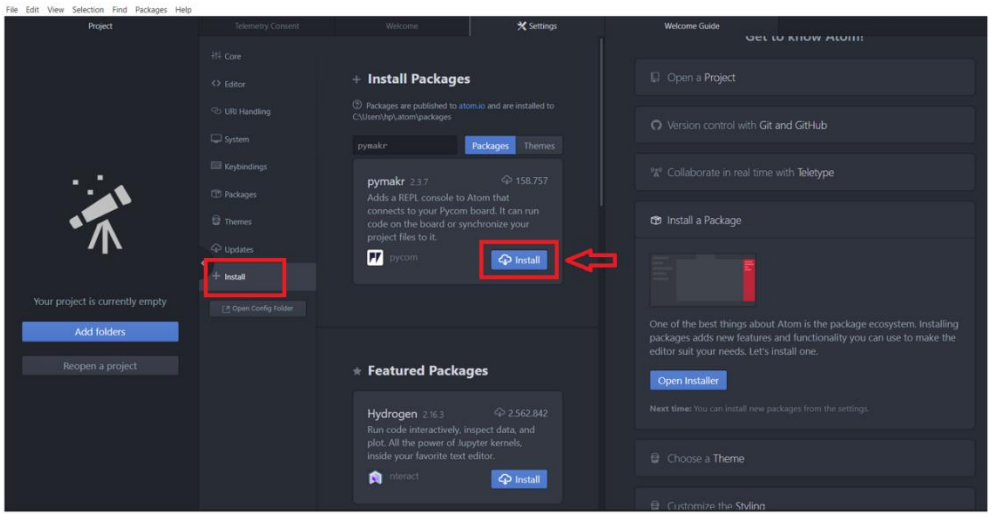


figura 2.9 Instalación de extensión Pymakr

2.2.2 Conexión del dispositivo Sipy a través de USB

Cuando ya se encuentra configurado e instalado el complemento Pymakr, se debe seguir los pasos siguientes para la conexión y asignación de un puerto COM para el dispositivo:

- Mediante la placa de expansión Pytrack y colocación del módulo Sipy, conectar el dispositivo a la PC a través del cable de alimentación USB.
- Colocar en Conecct Device y buscar el puerto COM que le asigna el computador.

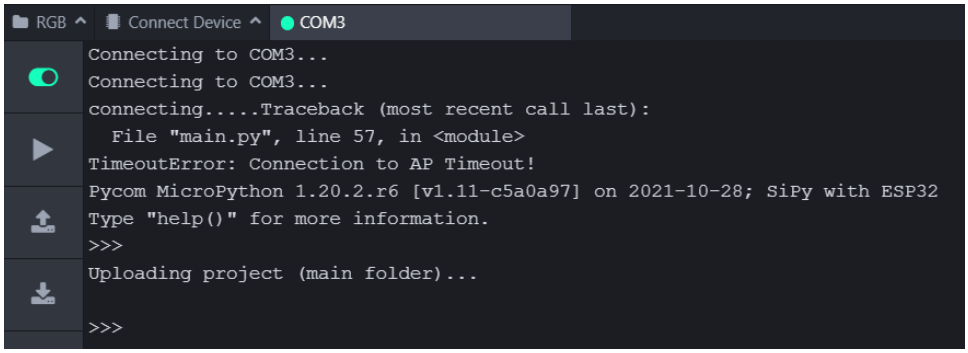


figura 2.10 Asignación de puerto COM al dispositivo Sipy

Atom asigna automáticamente la comunicación del dispositivo con la interfaz donde se proporciona acceso a los resultados de complicación. Si todo está correcto se puede comenzar a programar el código fuente.

2.3 Lenguaje programable en el módulo Sipy

El código que interpreta y que es único compatible con el dispositivo de desarrollo, es Mycrophython, implementado por el lenguaje de programación Python, empleado para potenciar y ejecutar el código fuente en un microcontrolador.

Para los primeros pasos que se debe establecer con el módulo Sipy, Pycom proporciona ejemplos para la correcta funcionalidad del dispositivo, por ejemplo, palpitación del RGB LED en un determinado tiempo [17], el código de implementación es el siguiente:

Código Fuente	Descripción
import pycom	Librería principal Pycom
import time	Librería que permite el salto de tiempo.
pycom.heartbeat(False)	Función de apagado del RGB LED.
while True:	Función repetitiva while, ciclo infinito.

pycom.rgbled(0xFF0000) # Red
time.sleep(1)
pycom.rgbled(0x00FF00) # Green
	Funciones designadas para los colores, los cuales se expresan de

time.sleep(1)
pycom.rgbled(0x0000FF) # Blue
time.sleep(1)
	manera hexadecimal, con un tiempo de encendido de un segundo.

Tabla 2.1 Código de verificación para el dispositivo Sipy, encendido y apagado del RGB LED.

2.3.1 Librerías recomendadas para el código fuente en el dispositivo Sipy

El módulo Sipy al ser programable en MicroPython requiere de ciertas librerías que se denotan a continuación:

Librerías	Descripción
import Sigfox	Enlace con el backend de Sigfox, definen la zona de radiación RCZ.
import struct	Interpreta bytes como dupla de datos binarios en paquetes.
import machine	Permite la comunicación con los pines conectados a un LED externo.
Import WLAN	Proporciona el acceso a la red Wifi del domicilio, lectura de usuario y contraseña.
import time	Librería que permite el salto de tiempo.

Tabla 2.2 Librerías específicas para el funcionamiento tanto de la red a la que nos conectemos vía Wifi y pines utilizados en el dispositivo.

2.3.2 Etapa de programación en el módulo Sipy

2.3.2.1 Botón de activación y diodos LED de verificación

El dispositivo Sipy consta de pines de propósito general de entrada y salida (GPIO), y es necesario realizar la verificación, tanto del botón, al momento que fue presionado y la señal Wifi que se establezca al conectarse a la red del domicilio.

- Para la verificación del botón se prende un led el cual está conectado al pin P12 del módulo Sipy GPIO12.
- Mientras tanto para la verificación de que la señal Wifi está vigente y existe comunicación entre la red y el dispositivo, se prenderá un led conectado al pin P9 del módulo Sipy GPIO21.

- Adicional a los pines de acceso, se debe tener en cuenta el botón que proporciona el enlace al dispositivo secundario ESP y posterior activación de la alarma al ser presionado, el cual está conectado al pin P10 del módulo Sipy GPIO13.

Las líneas de código que establecen la conexión de pines, tanto de los leds de verificación y el botón que es presionado son las siguientes:

```
led=Pin('P12', mode=Pin.OUT)

boton=Pin('P10', mode=Pin.IN, pull=Pin.PULL_UP)

led1=Pin('P9', mode=Pin.OUT)
```

2.3.2.2 Notificación de comunicación hacia la alarma

Cuando se inicializa la activación de la red y se conectan tanto el Sipy y la alarma vía Wifi, está actúa como servidor, la cual espera hasta que el requerimiento se solicite, mediante una notificación IP y URL, para esto se crea una variable bandera=0, que permite verificar si se presionó el botón de inicio, si es presionado se enciende el led de color verde y se procede a enviar el aviso de activación hacia la alarma, correspondiente a una IP única. Mediante el código siguiente se observa el proceso de envío y función creada:

```
bandera=0
while True:
    if bandera==0:
        if boton()==1:
            led.value(1)
            resp=http_get('http://192.168.1.20/LEDON1?')
            print(resp)
            bandera=1
            enviar_dato(bandera)
```

la información anteriormente enviada se realiza mediante la función http_get, la cual para realizar el enlace con la alarma es necesario un puerto de acceso número 80 (puerto de enlace para servidores web).

Los parámetros de salida son el host con la URL e IP 192.168.1.20, y la función socket que es un parámetro para la conexión de comunicación.

La función *s.send,* que permite el envío de la URL en un formato entendible según lo solicitado, para esto nos ayuda el parámetro *utf8* que proporciona un entendimiento

correcto de los caracteres, para especificar cada uno de estos parámetros y funciones específicas se presenta las siguientes líneas de código:

```
def http_get(url):
    import socket
    _, _, host, path = url.split('/', 3)
    addr = socket.getaddrinfo(host, 80)[0][-1]
    s = socket.socket()
    s.connect(addr)
    s.send(bytes('GET /%s HTTP/1.0\r\nHost: %s\r\n\r\n' % (path, host), 'utf8'))
```

2.3.2.3 Conexión del módulo Sipy a la red Wifi

Pycom proporciona una variedad de ejemplos en los cuales se analiza la conexión vía Wifi para los módulos de desarrollo, en el caso de estudio se da análisis en el domicilio el cual presenta una dirección IP estática, y que el dispositivo principal permite la comunicación con el enrutador inalámbrico (ESP8266). [18]

También se debe configurar la IP estática, la subred o máscara y la puerta de enlace (Gateway), que este último debe ser el mismo en la configuración de código para el ESP, de lo contrario no se podrá asignar una conexión y comunicación entre dispositivos.

Una vez establecida la dirección IP estática, se procede a colocar el SSID (usuario de la red Wifi en el domicilio), y la contraseña, espera alrededor de 5 segundos para que se proporcione la conexión y reenvía una notificación de conectado.

El código que permite el enlace del dispositivo con la red domiciliaria es el siguiente:

```
wlan = WLAN()
if machine.reset_cause() != machine.SOFT_RESET:
    wlan.init(mode=WLAN.STA)
        wlan.ifconfig(config=('192.168.1.10', '255.255.255.0', '192.168.1.1', '8.8.8.8')) # (ip, subnet_mask, gateway, DNS_server)
if not wlan.isconnected():
    wlan.connect('SSID', auth=(WLAN.WPA2, 'PASSWORD'), timeout=5000)
    print("connecting",end='')
```

2.3.2.4 Envío de datos al backend de Sigfox

Para el envío de información se crea una función enviar_dato (), y como paso primordial es la verificación de la zona en la que opera el módulo Sipy, en este caso es Sudamérica con la RCZ4.

El parámetro socket que permite la conexión de comunicación entre el dispositivo y el backend de Sigfox,

El parámetro *s.send* que envía la coordenadas del domicilio, como son únicas y exactas no existe variación en los datos, y adicionalmente el envío de un byte de bandera cuando se encuentra en estado de encendido o apagado la alarma. En total se envía 9 Bytes al backend de Sigfox en dupla de bytes en formato hexadecimal, especificando cada dato de la coordenada que se envió, como es la altitud y latitud.

Al ser enviado un dato a la nube Sigfox, está admite como máximo 12 bytes en la información enviada por lo que los 9 bytes que son registrados en el backend para el caso de estudio no tendrán inconvenientes en la recepción.

Los datos que se envían son de tipo float/double o boolean, el rango de valores se especifica en la siguiente tabla: [19]

Tipos de datos	Memoria que ocupa	Rango de valores
boolean	1 byte	0 o 1 (verdadero o falso)
float/double	4bytes	-3.4028235E+38; 3.4028235E+38

Tabla 2.3 Especificación de datos utilizados para el envío al backend de Sigfox, coordenadas del domicilio y bandera si está apagada o encendida la alarma.

El código que establece el envío de datos hacia el backend de Sigfox es el siguiente:

```
def enviar_dato(bandera):
import socket
sigfox = Sigfox(mode=Sigfox.SIGFOX, rcz=Sigfox.RCZ4)
s = socket.socket(socket.AF_SIGFOX, socket.SOCK_RAW)
s.setblocking(True)
s.setsockopt(socket.SOL_SIGFOX, socket.SO_RX, False)
s.send(struct.pack("<f",CD_1)+struct.pack("<f",CD_2)+struct.pack("<B",bandera)
```

2.4 Esquema de conexión con el Módulo Sipy

Para una mejor visualización se diseñó un esquema en Proteus, tomando en cuenta los pines de conexión en los terminales GPIO y alimentación a 3,3 Voltios del módulo Sipy.

Se presenta un boton (B1), que proporciona la activación de la alarma e interrupción al módulo Sipy que posteriormente envía los datos que corresponden a coordenadas del lugar donde se presionó el pulsador.

Dos diodos LED (D1 y D2) que están conectados a los terminales GPIO, que son activados cuando se acciona la alarma y si se estableció conexión con la red Wifi del domicilio.

Un bloque (J1) donde se conecta las extensiones de alimentación y GND.

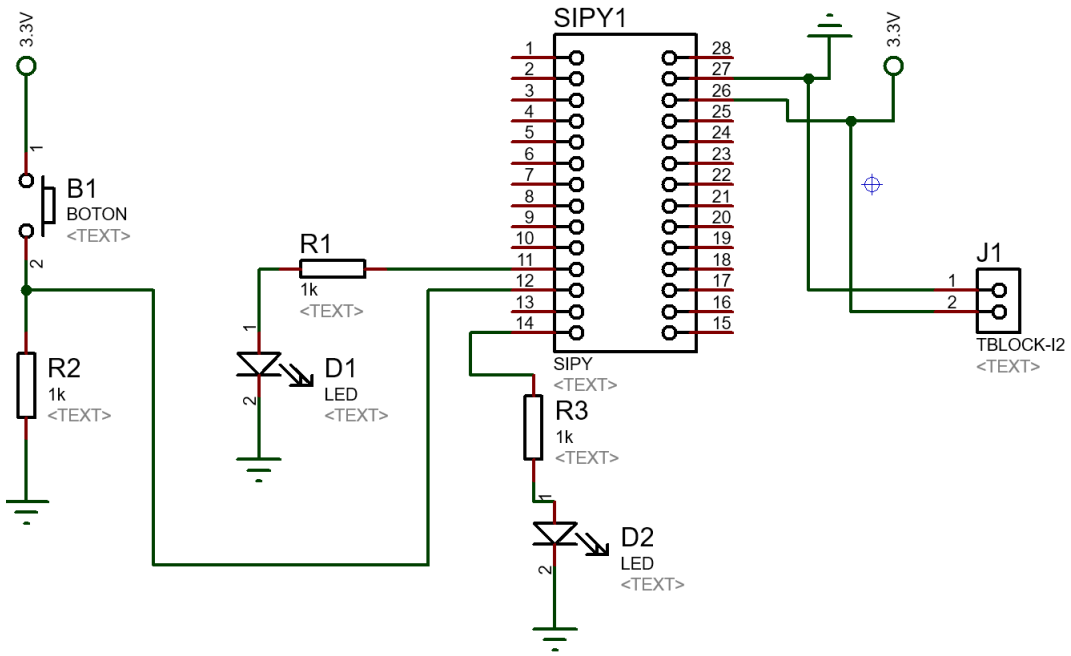


figura 2.11 Diseño del circuito principal mediante los pines específicos para el boton de interrupción hacia el módulo Sipy y diodos LED de verificación.

2.4.1 Estructura del circuito principal con el Módulo Sipy

Para realizar el esquema de la placa electrónica, el programa de diseños electrónicos Proteus nos permite exportar un circuito impreso que permite extraer el dimensionamiento de una forma más profesional.

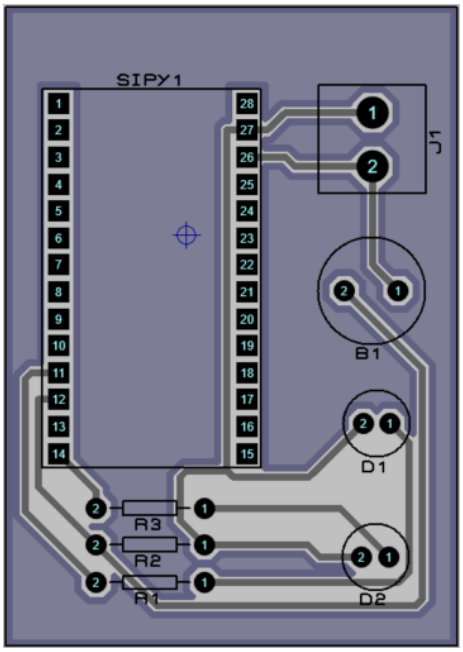


figura 2.12 Circuito esquemático con pistas y componentes esenciales para el diseño.

2.4.2 PCB del circuito principal

Para la estructuración de los componentes y funcionamiento del circuito principal se presenta en forma física la placa que es necesaria para la colocación de elementos electrónicos.

figura 2.13 Lámina de cobre terminada para la PCB del circuito principal.

2.4.3 Mecanismo de implementación módulo Sipy y antena Sigfox

Según el esquema de conexión y estructura del circuito implementado se puede finalizar el montaje del mecanismo, como se muestra a continuación:

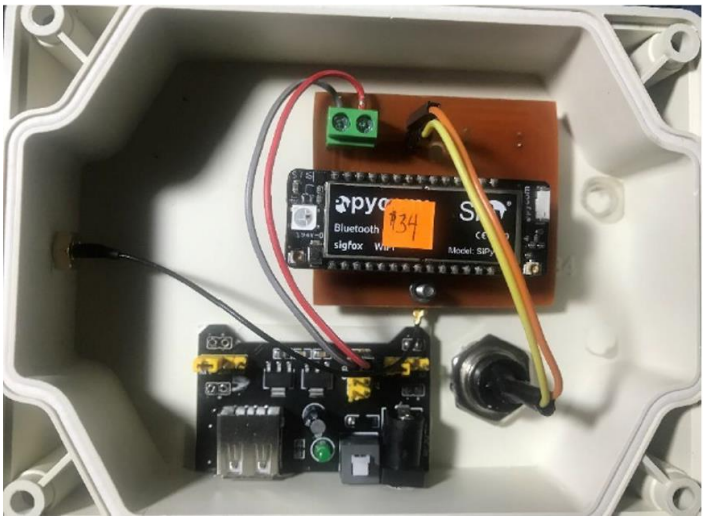


figura 2.14 Conexión e implementación de módulo fuente, pulsador, módulo Sipy y antena Sigfox.

2.5 Instalación del IDE de Arduino

Es una aplicación de lectura de código programable para dispositivos microcontroladores que a su vez sean compatibles con Arduino. En el caso de estudio si utilizará el módulo ESP8266 que tiene familiaridad con el IDE y posterior código de programación.

Para descargar e instalar el IDE debemos ingresar a la página oficial de Arduino. [20]

Se debe elegir el sistema operativo Windows y automáticamente se da la descarga del IDE de Arduino.

figura 2.15 Página de descarga oficial para el IDE de Arduino

Se instala el IDE y se obtiene acceso a su interfaz inicial.

2.5.1 Configuración mediante el IDE para el ESP8266

En la interfaz gráfica de Arduino se visualiza el panel principal, en el cual se debe presionar en Herramientas y se busca Placa, finalmente en Gestor de Tarjetas. [21]

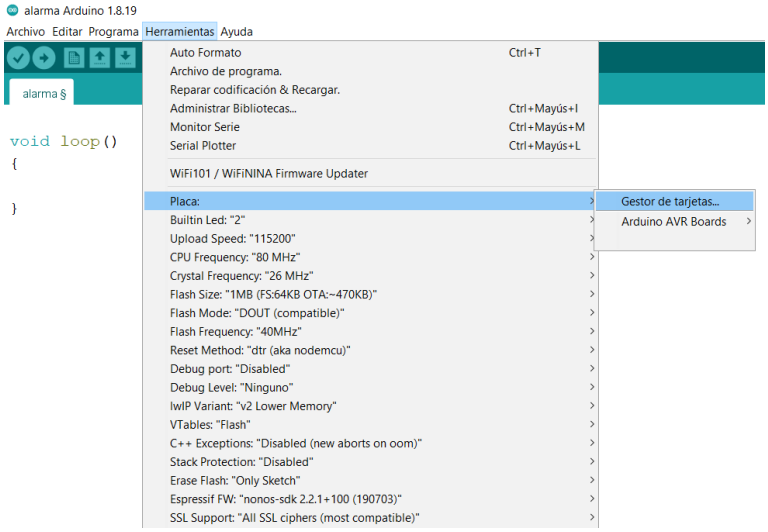


figura 2.16 Interfaz de configuración en el IDE de Arduino.

Al ingresar al Gestor de tarjetas buscamos el ESP8266, y se instala el complemento para la lectura de código compatible con el módulo, además que incluye las librerías necesarias para el proceso de compilación.

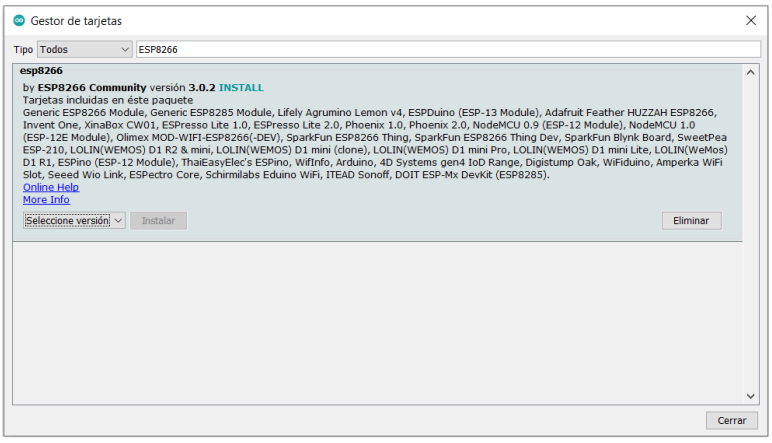


figura 2.17 Instalación del complemento ESP8266 en el IDE de Arduino.

Una vez configurados los parámetros de inicio, ya se puede programar el código fuente.

2.6 Lenguaje programable en el módulo ESP8266

Mediante la configuración de parámetros al momento de la instalación del IDE de Arduino, el lenguaje de programación que es compatible con el ESP es Java, que proporciona una

variedad de librerías y acceso a ejemplos ilustrativos que generan un mejor entendimiento del código a utilizar.

2.6.1 Librerías recomendadas para el código fuente en el Módulo ESP

Con el complemento ya instalado y la última versión para el ESP, solo falta añadir una librería adicional que proporciona la conexión Wifi con el módulo, la cual es *#include <ESP8266WiFi.h>.*

2.6.2 Etapa de programación para el ESP8266

2.6.2.1 Asignación de dirección IP estática, puerta de enlace y subred

Se define dos constantes que corresponden al SSID y contraseña, y mediante la dirección IP que proporciona la red Wifi del domicilio se establece los parámetros y direcciones correspondientes, las líneas de código que permiten la asignación son las siguientes:

```
const char* ssid = "SSID";
const char* password = "PASSWORD" ;
IPAddress staticIP(192,168,100,20);
IPAddress gateway(192,168,100,1);
IPAddress subnet(255,255,255,0);
```

2.6.2.2 Conexión a la red domiciliaria mediante el ESP8266

Mediante la función *serial.printf* se verifica si el ESP se está conectando a la red Wifi con el SSID propio del servidor domiciliario.

La función *Wifi.config* que es propia del ESP proporciona la conexión mediante las direcciones de red establecidas en las variables iniciales constantes (IP, Máscara y Subred).

Una vez conectado al servidor, la función que inicializa la comunicación es *Wifi.begin* y nos proporciona los datos de verificación como son el SSID y la contraseña en la salida de la interfaz, si todo es correcto aparece el mensaje de CONECTADO. El código para cada función de análisis se menciona de la siguiente manera:

```
Serial.begin(115200);
Serial.println();
Serial.printf("Connecting to %s\n", ssid);
WiFi.config(staticIP, gateway, subnet);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED)
```

2.6.2.3 Petición entrada hacia el ESP en modo servidor

Cuando ya existe conexión a la red del servidor, el ESP permanece en espera hasta que exista el requerimiento de envío que proporciona el Módulo Sipy al ser presionado el botón de inicio.

Mediante la función *WifiClient* se establece la variable *client* que en conjunto con la función *return* se crea un lazo repetitivo que permanece en modo espera hasta que no se presente el requerimiento de petición.

Una vez presionado el botón de inicio en el circuito principal donde se encuentra el módulo Sipy, se inicia la petición mediante la función *request* la cual nos indica estados lógicos programables, como son:

- LEDON1, cuando accede la señal de aviso para encender la alarma.
- LEDOFF1, cuando se vuelve a presionar el botón de inicio y se apaga la alarma.

La lectura se proporciona en el pin de acceso GPIO2 del ESP8266.

Todo lo mencionado se realiza con las siguientes líneas de código:

```
WiFiClient client = server.available();
if (!client) {
dato=digitalRead(Sensor);
return;}
request = client.readStringUntil('\r');
if ( request.indexOf("LEDON1") > 0 ) { digitalWrite(LED1_Pin, HIGH);}
else if ( request.indexOf("LEDOFF1") > 0 ) { digitalWrite(LED1_Pin, LOW);}
```

2.7 Esquema de conexión con el módulo ESP8266

La aplicación utilizada para el diseño del circuito integrado es Proteus, permite la correcta distribución de pines y visualización para la PCB que se establecerá de forma física.

El botón (B1) que permite el reinicio del módulo ESP8266, al presionarlo se cierra y permite el acceso a GND que hace reset al dispositivo, de esa manera se comprueba si el código subido está funcionando correctamente.

El transistor 2N3904 que se conecta al pin número 4 (GPIO2) del módulo ESP que permite la regulación y control del ingreso de una corriente muy grande, en el caso de estudio ingresa la petición de acceso al servidor cuando se presionó el botón de inicio.

La regleta auxiliar (J1) que permite la comunicación serial para poder subir el código fuente y visualizar el funcionamiento de la conexión Wifi. Las regletas (J2 y J3) que son extensiones las cuales establecen las conexiones de alimentaciones y GND respectivas al circuito.

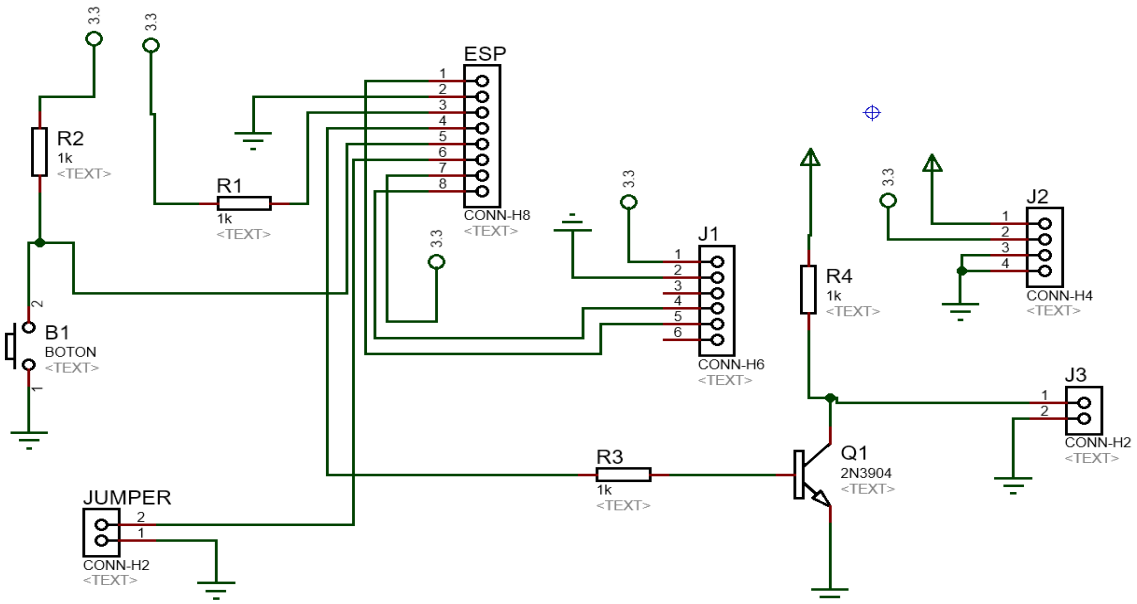


figura 2.18 Diseño del circuito secundario mediante los pines específicos del ESP8266 y elementos de auxiliares de conexión.

2.7.1 Estructura del circuito principal con el Módulo ESP8266

Las pistas y modelo específico del circuito con el esquema y colocación previa a los elementos se presenta mediante la siguiente figura:

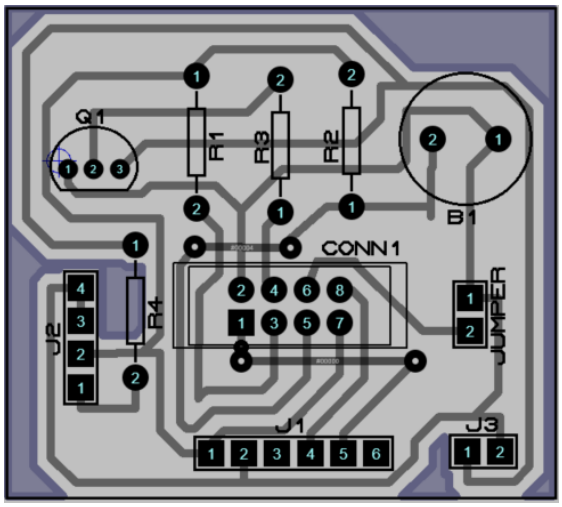


figura 2.19 Circuito representativo de pistas y componentes esenciales para el diseño del módulo ESP.

2.7.2 PCB del circuito principal para el Módulo ESP8266

La PCB para la colocación de cada elemento se presenta en la lámina de cobre terminada, con cada una de las entradas respecto a los componentes de estudio.

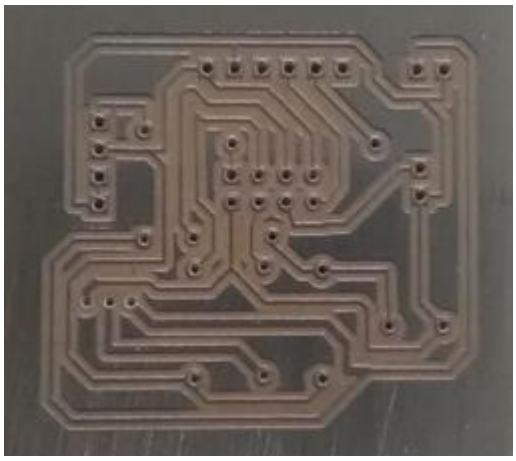


figura 2.20 Lámina de cobre terminada para la PCB del circuito principal con el ESP8266.

2.7.3 Mecanismo de implementación módulo ESP8266

Una vez finalizada la configuración y conexiones en la PCB se puede concluir el montaje del mecanismo con todos los elementos y módulos representativos para el funcionamiento, como se muestra a continuación:

figura 2.21 Conexión e implementación de módulo fuente, relé, módulo ESP8266 y alimentación al motor de sirena.

2.8 Enlace módulo Sipy, red Sigfox y Ubidots

Simplificar el tiempo que implica crear una aplicación que proporcione los datos de salida enviados por un dispositivo emisor, proporcionar mecanismos digitales para la interpretación de información envidad, Ubidots lo facilita enlazando el módulo en análisis y la red de estudio, permitiendo la decodificación de variables y proyectándolas en dispositivos como un móvil, tableta y computador.

2.8.1 Creación de cuenta en Ubidots

Ubidots presenta su página oficial para la creación y registro de la cuenta [22], no es una aplicación descargable y su suscripción es de un mes gratis para estudiantes, la cual es muy completa y no se presenta problemas para la interpretación de datos enviados.

Para el registro en la nube de Ubidots se selecciona en llévame a Ubidots Stem, según la siguiente figura:

figura 2.22 Página inicial para el registro en la nube Ubidots.

Para el siguiente paso se presenta la página de creación de usuario y contraseña, se registra el correo electrónico personal y se recibe la notificación de activación gratuita por treinta días.

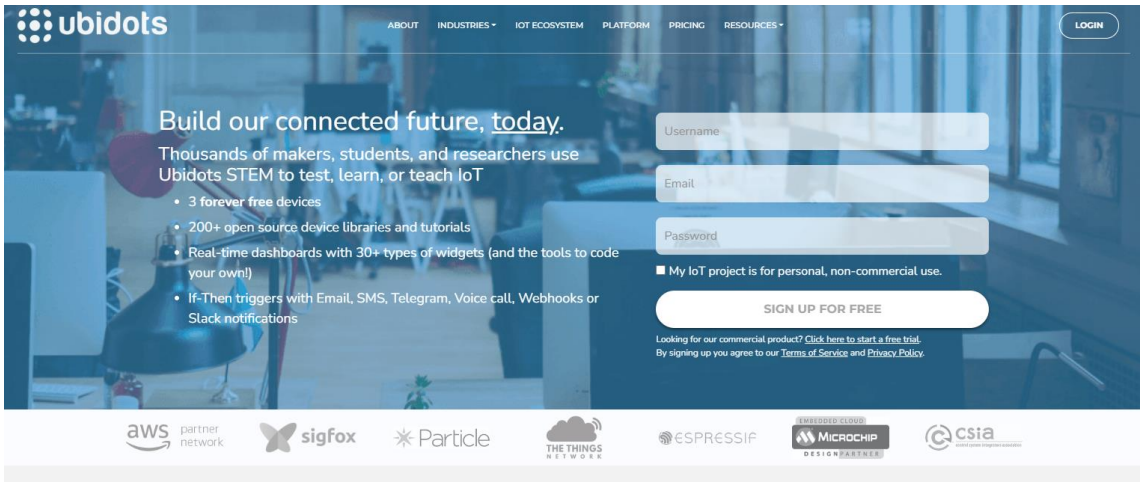


figura 2.23 Credenciales únicas para el acceso a la plataforma de Ubidots.

Con los pasos realizados correctamente se puede ingresar al panel principal de Ubidots, constatar la pestaña de dispositivos, datos y algunos pasos generales para la creación e integración de mecanismos microcontroladores.

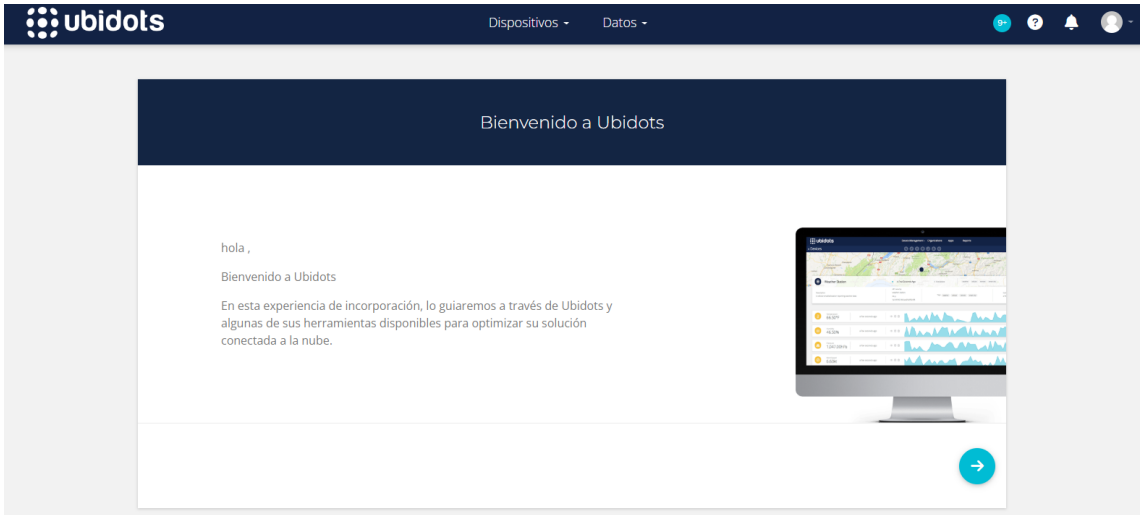


figura 2.24 Panel principal, interfaz Ubidots.

2.8.2 Integración del dispositivo Sipy a Ubidots

En el panel principal de Ubidots se selecciona dispositivos, crear un nuevo dispositivo y seleccionar la conectividad en análisis la cual es Sigfox. Se despliega los fabricantes y compatibilidad de cada dispositivo, en el caso de estudio se establece Pycom y el módulo Sipy.

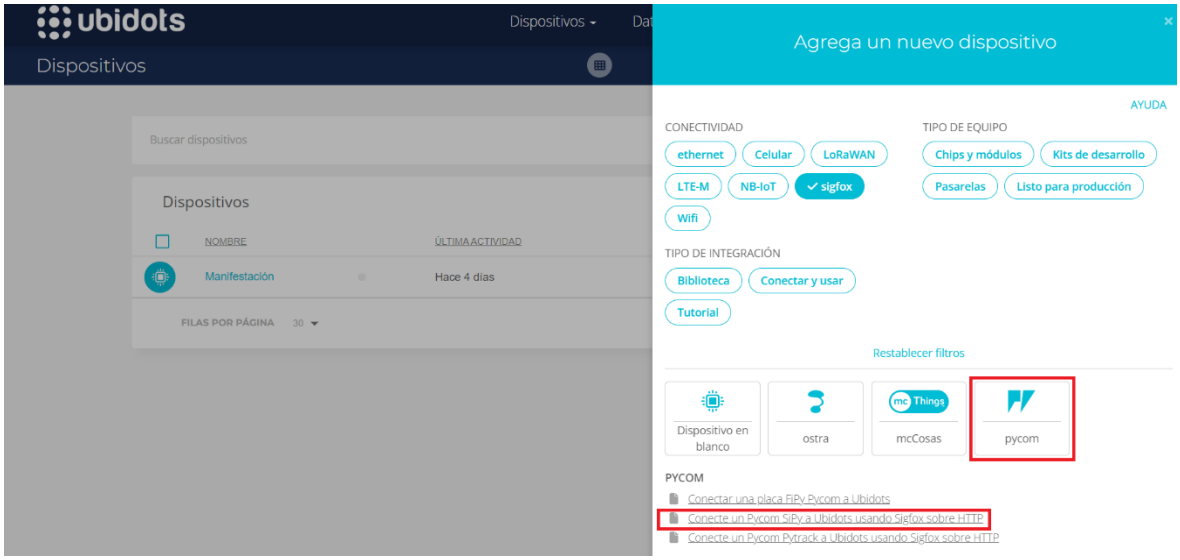


figura 2.25 Selección de conectividad, fabricante y dispositivo.

2.8.3 Enlace de datos a Ubidots

El enlace de mensajes enviados hacia Ubidots se lo realiza mediante una URL única y peticiones de devolución de llamada (Callbacks), cuando el dispositivo se integra a Ubidots mediante el fabricante y el módulo requerido, se procede a ingresar al backend de Sigfox con las credenciales como son usuario y contraseña, y se continua con los siguientes: [23]

- Seleccionar en PYCOM_Devkit:

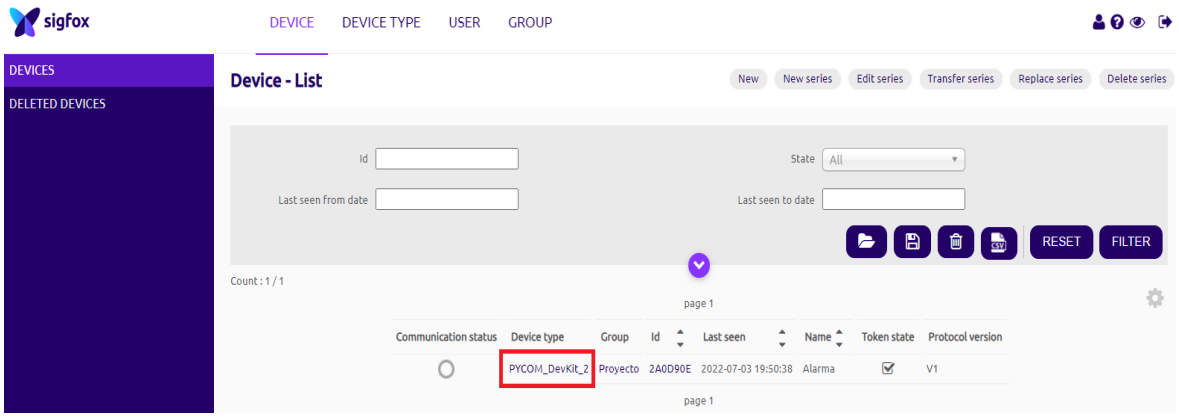


figura 2.26 Selección del tipo de dispositivo en el backend de Sigfox y enlace a las callbacks.

- Dirigirse a Callbacks

figura 2.27 Creación de parámetros en la petición callbacks.

Al momento que se crea una petición callback, se configura el tipo de dato y el enlace ascendente (uplink) o descendente (downlink) según el mecanismo de envío, y por último el tipo de canal (URL) que permite el enlace hacia la plataforma de Ubidots.

Los datos que se envían del dispositivo origen corresponden a la ubicación del domicilio como altitud y latitud, además de un indicador que especifica si la alarma esta encendida o apagada, la configuración de carga útil (Payload), se realiza según los siguientes tipos de decodificación:

- float: los parámetros de estudio corresponden a longitud en bits de valor (32 o 64 bits), y el parámetro endian que es utilizado para datos flotantes de varios bytes.
- uint: corresponde a un entero sin signo, establece la notificación de encendido o apagado de la alarma.

El dominio que indica el enlace a Ubidots es:
http://industrial.api.ubidots.com/api/v1.6/devices/{device}

El siguiente parámetro que indica el establecimiento del enlace en la comunicación con el medio de desarrollo es la autenticación de token.

2.8.3.1 Autenticación token

Ubidots presenta una llave de acceso (API Key) para las solicitudes enviadas a su entorno, cada solicitud enviada a Ubidots requiere un token, el cual es una contraseña única que se genera al registrar y crear una cuenta en Ubidots, los pasos para la obtención del token son los siguientes:

- Dirigirse al panel principal de Ubidots y seleccionar en credenciales API.

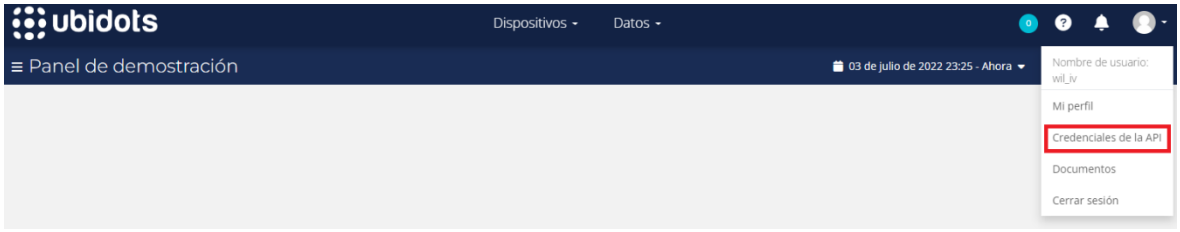


figura 2.28 Obtención de credenciales de acceso según las solicitudes enviadas por el dispositivo inicial.

- Seleccionar en copiar token.

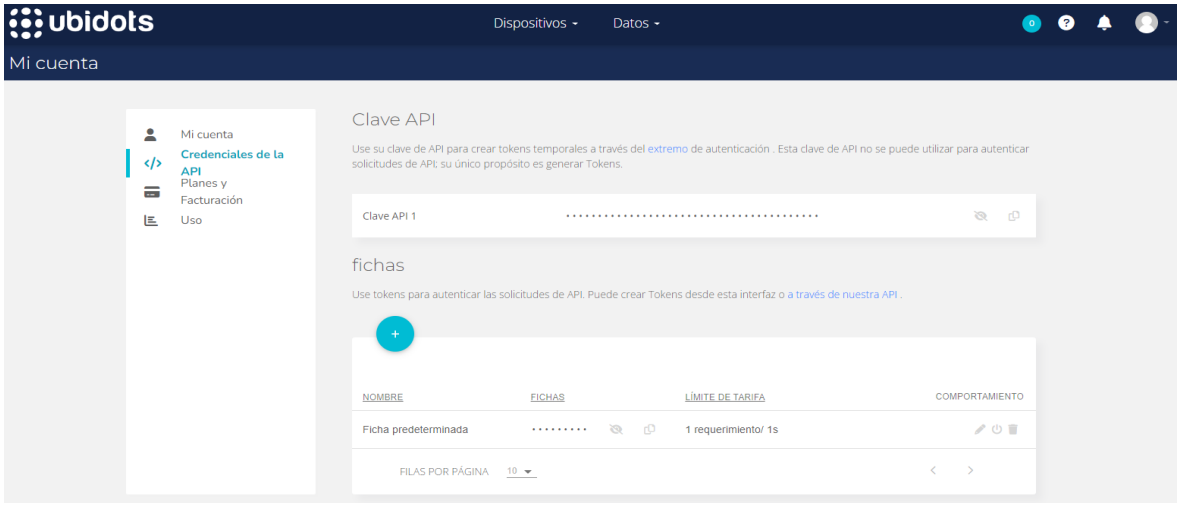


figura 2.29 Visualización de token y clave API

Cuando la configuración de parámetros en la devolución de llamada (callbacks) se complete de manera satisfactoria, ya se puede registrar el módulo Sipy en el entorno de Ubidots.

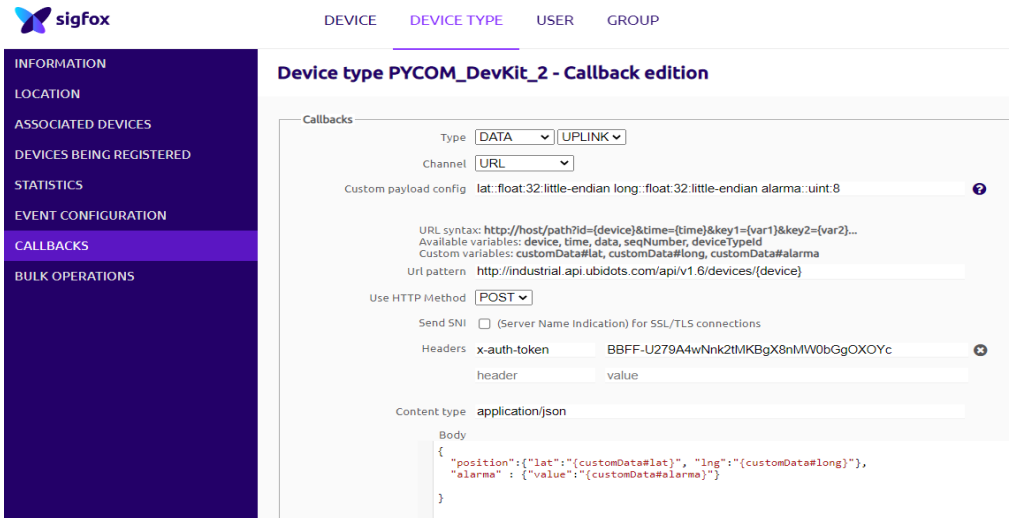


figura 2.30 Configuración final de parámetros de acceso del módulo Sipy a Ubidots.

El tipo de contenido es específico para la configuración del widget en el panel principal de Ubidots, en el caso de estudio se crea la posición y el indicador de alarma. Estos parámetros son de uso temporal, ya que, serán necesarios cuando se visualicen por el usuario final en el punto de recepción.

2.8.4 Registro de módulo Sipy en Ubidots

Para la autenticación y registro del módulo Sipy en Ubidots, se envía los datos de acuerdo al caso de estudio que corresponden a las coordenadas de ubicación, 4 bytes de latitud, 4 bytes de altitud y un byte de verificación de la alarma al estar encendida o apagada, y son registrados en el backend de Sigfox, cada uno de los bytes enviados son decodificados en duplas, y al mismo tiempo el proceso de verificación de token es ejecutado y posteriormente el dispositivo se enlaza en la plataforma de Ubidots, con un nombre específico.

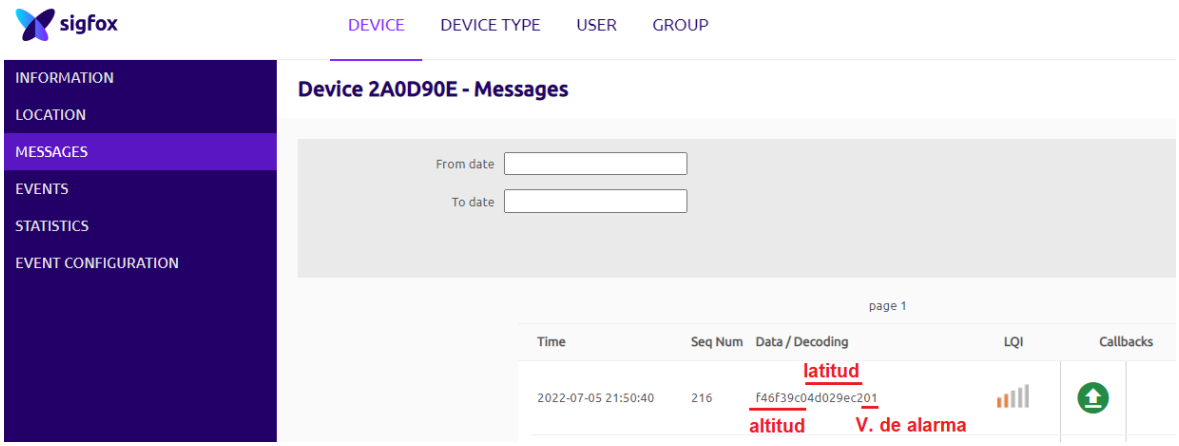


figura 2.31 Datos obtenidos en el backend de Sigfox correspondientes a los 4 bytes de altitud, latitud y el byte de verificación de alarma

Ya verificado el token y los parámetros de devolución de llamada, el módulo ya se puede visualizar en el panel principal de Ubidots.

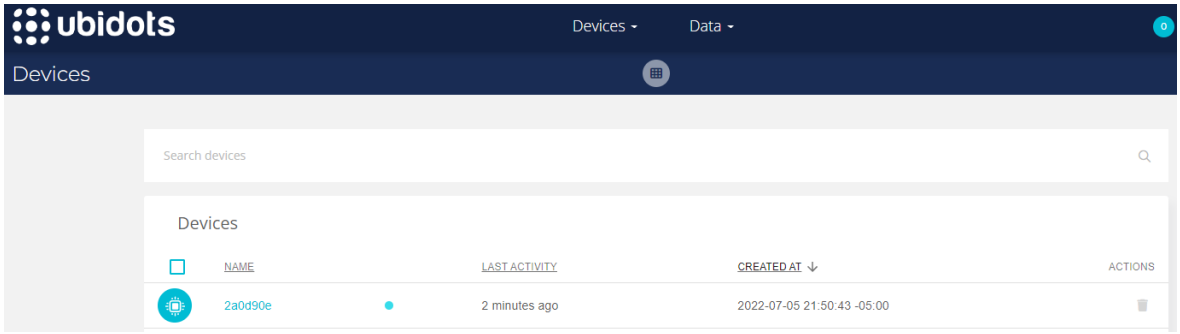


figura 2.32 Verificación de dispositivo integrado al panel principal de Ubidots.

Tanto la hora de verificación y llegada de los datos es en tiempo real, por esa razón se interpreta la misma en el backend de Sigfox y Ubidots.

2.8.5 Integración de widget en Ubidots

En la plataforma de Ubidots se puede interpretar de una mejor manera los datos de llegada en la interfaz principal, para esto es necesario dos widgets, que son tableros de visualización e interpretación, el primer widget es un indicador, el cual anuncie si la alarma esta encendida o apagada, y el segundo es el mapa de ubicación donde se encuentre el Sipy que es accionado por el botón de inicio.

Se toma en cuenta que los widgets fueron integrados como parámetros, en el cuerpo del tipo de contenido de la petición de llamada (callbacks).

2.8.5.1 Indicador de alarma

Es necesario integrar un indicador que avise la actividad de la alarma al ser presionado el botón en el módulo principal, para constatar el aviso se debe añadir un widget indicador al tablero o Dashboards de Ubidots.

En el panel de inicio Ubidots se añade el widget de interés y se configura de acuerdo con el parámetro de estudio, en el caso respectivo la alarma.

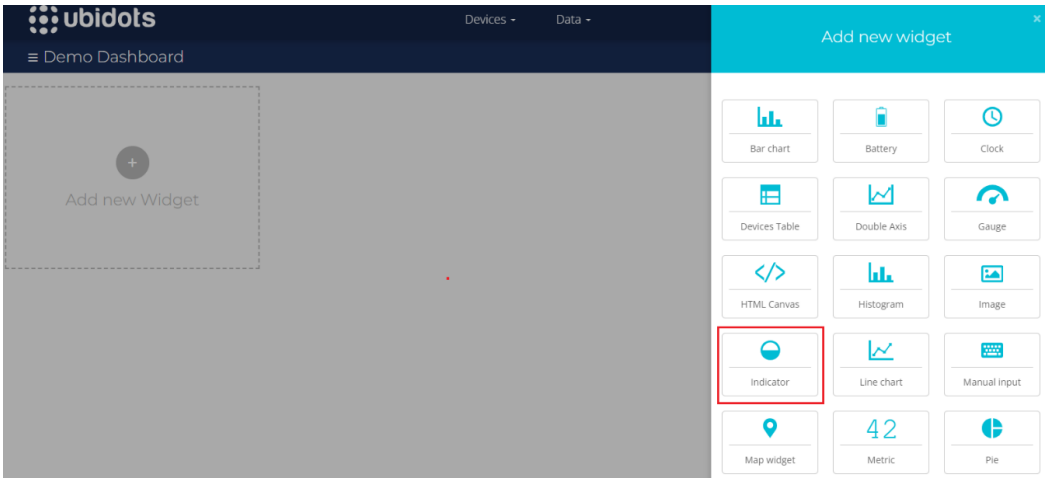


figura 2.33 Selección del widget, indicador de actividad para la alarma.

Para proporcionar el acceso correcto a los parámetros de la alarma es necesario asignar la variable creada en el backend de Sigfox, con el nombre correspondiente según la siguiente figura:

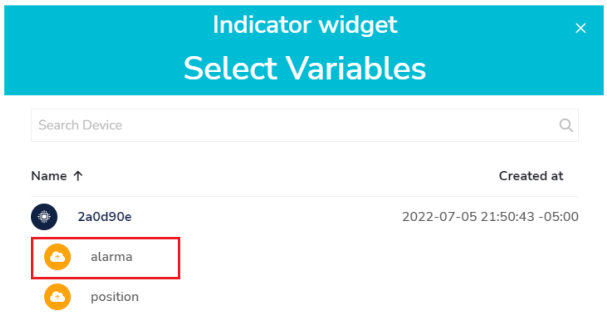


figura 2.34 Variable de asignación según el parámetro de salida, indicador alarma.

Finalizada la configuración de asignación de variable y elección del parámetro correcto, ya se puede visualizar el indicador en el tablero principal.

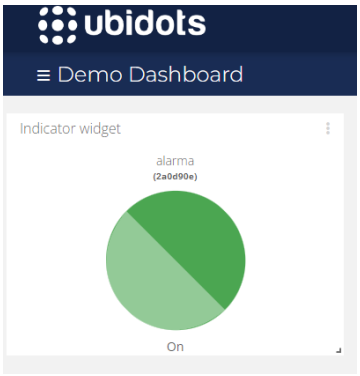


figura 2.35 widget indicador alarma

2.8.5.2 Indicador Geográfico

Integrar un indicador geográfico permite la llegada pronta de unidades de vigilancia al domicilio, ya que, se podrá obtener el lugar exacto donde se presente inconvenientes solo con seleccionar el punto de marcación.

Nuevamente en el panel de inicio se puede agregar el widget que se solicita según la siguiente figura:

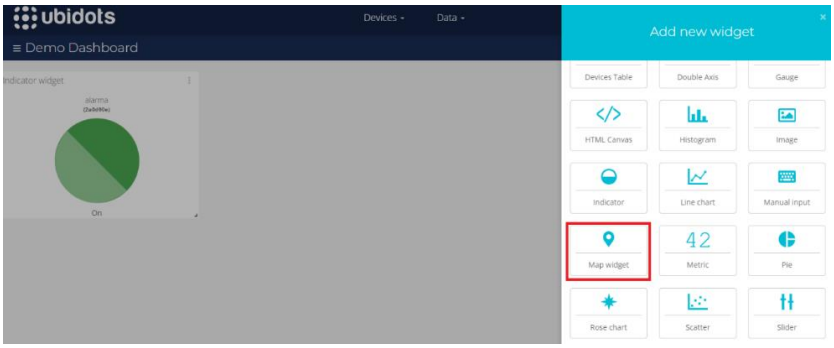


figura 2.36 Selección del widget, indicador geográfico del lugar donde fue presionado el botón de activación.

Se indica el dispositivo agregado que corresponde al módulo Sipy y se guarda el cambio generado, como se muestra a continuación:

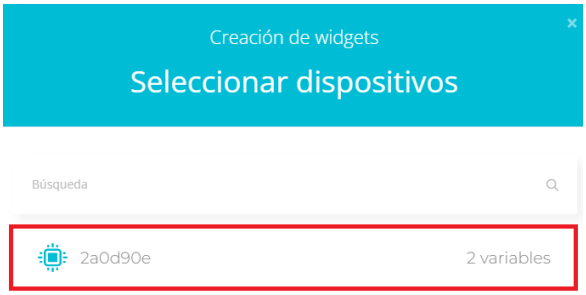


figura 2.37 Selección del dispositivo y creación del widget geográfico.

Terminado cada uno de los parámetros, se proporciona la ubicación en tiempo real del lugar donde se generó alguna anomalía, y se presenta el widget del mapa geográfico correspondiente.

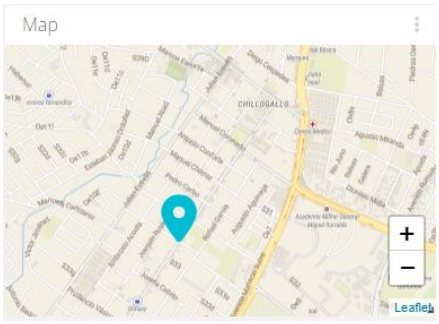


figura 2.38 widget geográfico, ubicación en tiempo real del domicilio.

Ubidots permite obtener sus resultados e indicadores en un dispositivo celular o a su vez un computador, lo que proporciona un mayor punto de acople al momento de observar la información enviada.

3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.1 Resultados

Los datos obtenidos se realizaron en el norte y sur de Quito, Sigfox proporciona una gran cobertura en la ciudad capital, por lo que, los resultados fueron óptimos al realizar pruebas de comunicación entre el dispositivo Sipy y el backend de Sigfox.

3.1.1 Entorno sin línea de vista en el interior del domicilio

En el interior del domicilio no se presenta una línea de vista óptima por lo que, los datos que fueron enviados al backend de Sigfox se registran, sin embargo, no se establece actividad en la devolución de llamada para el previo registro en Ubidots, por está razón no es recomendable que el mecanismo que contiene el módulo Sipy y la antena de Sigfox estén en el interior del domicilio son línea de vista.

Ya sea en cualquier estado de tiempo, mientras el mecanismo de activación este en el interior, no se puede establecer comunicación con la plataforma de Ubidots.

Se realizo algunas pruebas de verificación en varios puntos del domicilio y horas del día, para un mejor análisis y registros previos.

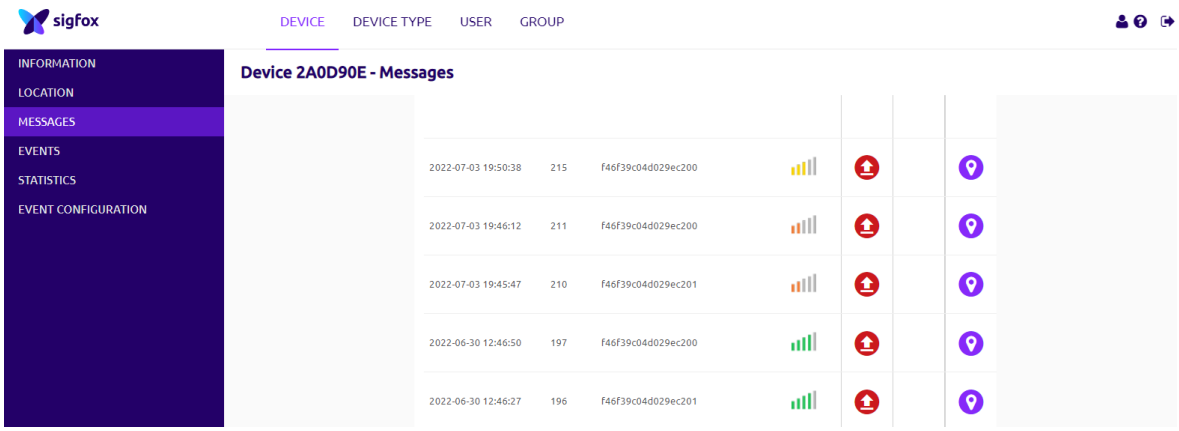


figura 3.1 Datos en forma hexadecimal de las coordenadas y verificación de alarma, registro en el backend de Sigfox, sin actividad en la devolución de llamada (callbacks).

3.1.1.1 Comparación de datos correspondientes a los estados de tiempo estudiados

La actividad en la devolución de llamada es nula, no habrá registro en Ubidots y la representación de datos en los widgets no tendrá correspondencia.

Estado del día	Pruebas	Hora	Callbacks	Registro en Ubidots
tarde	Prueba 1	12:46:27	Registro sin éxito	No
tarde	Prueba 2	12:46:50	Registro sin éxito	No
noche	Prueba 3	19:45:47	Registro sin éxito	No
noche	Prueba 4	19:46:12	Registro sin éxito	No
noche	Prueba 5	19:50:38	Registro sin éxito	No

Tabla 3.1 Resultados en el interior del domicilio, actividad de devolución de llamada nula.

3.1.2 Entorno con línea de vista al exterior

El mecanismo que contiene al módulo Sipy y la antena Sigfox, presentan mayor alcance a las antenas base si están en un entorno externo, los datos son enviados sin inconvenientes tanto en el día, noche y madrugada según las pruebas realizadas en el presente caso de estudio.

Al presionar el botón de inicio, los datos como altitud, latitud y byte de verificación de alarma, fueron enviados sin ningún problema, y a su vez el previo registro en el backend de Sigfox:

- En el día

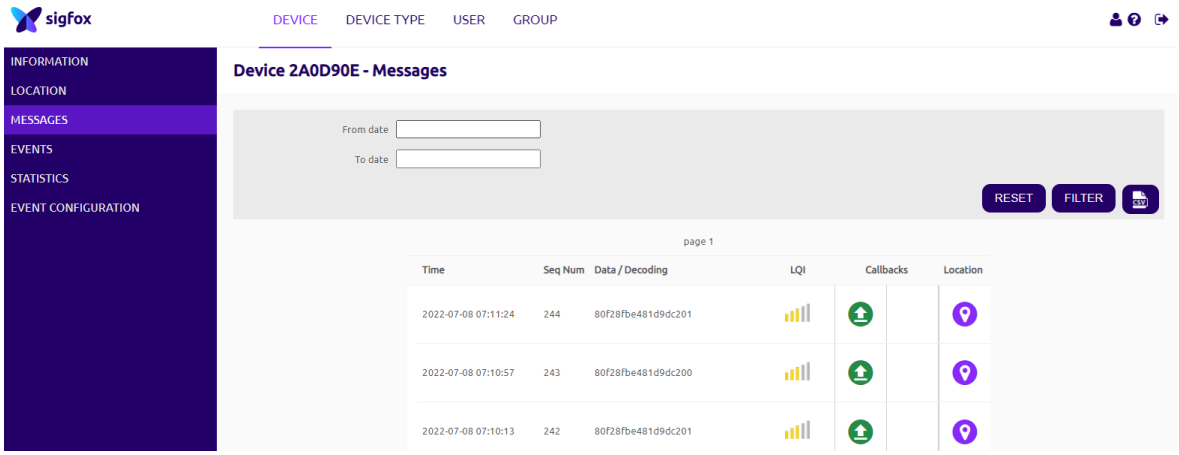


figura 3.2 Datos en forma hexadecimal de las coordenadas y verificación de alarma, registro en el backend de Sigfox correspondientes a la mañana.

Los datos esperados fueron satisfactorios, el backend de Sigfox registro las coordenadas de altitud y latitud de manera hexadecimal, así como el byte de verificación de alarma, la información que es receptada en la nube de Sigfox debe presentarse en el panel de análisis del widget alarma según la siguiente tabla:

DATE	VALUE
2022-07-08 07:11:28 -05:00	1.00
2022-07-08 07:11:00 -05:00	0.00
2022-07-08 07:10:16 -05:00	1.00

Tabla 3.2 Datos enviados y registrados en Ubidots, con actividad pasiva y activa de la alarma, correspondientes a la mañana.

En cada uno de los estados de tiempo, día, noche y madrugada debe coincidir en tiempo real y el retardo por mensaje enviado debe ser mínimo de aproximadamente 3 segundos, lo que queda por verificar en los demás resultados.

- En la noche

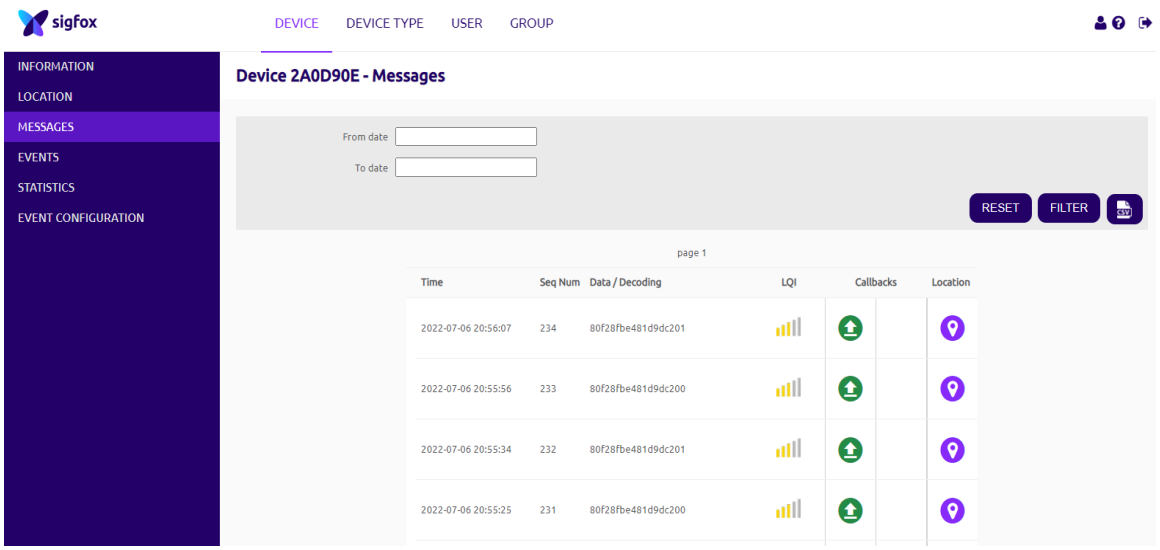


figura 3.3 Datos en forma hexadecimal de las coordenadas y verificación de alarma, registro en el backend de Sigfox correspondientes a la noche.

Para realizar el análisis correcto, se establece la comparación de datos enviados, los cuales fueron obtenidos de manera satisfactoria, como se observa en la figura 3.3 y la deben ser los mismos en la tabla que se presenta a continuación, coincidiendo la hora y minutos en los cuales se registran tanto en el backend de Sigfox como Ubidots.

2022-07-06 20:56:11 -05:00	1.00
2022-07-06 20:55:59 -05:00	0.00
2022-07-06 20:55:37 -05:00	1.00
2022-07-06 20:55:28 -05:00	0.00

Tabla 3.3 Datos enviados y registrados en Ubidots, con actividad pasiva y activa de la alarma, correspondientes a la noche.

Los resultados en el día y la noche están en la relación de tiempo correcta, no se presentan inconvenientes y es necesario realizar pruebas en cualquier hora del día, hasta ahora la alarma comunitaria se acciona y al mismo tiempo se registra la información en Ubidots del lugar donde se presentan anomalías, para una mejor constancia se continua con el estado de tiempo en la madrugada.

- En la madrugada

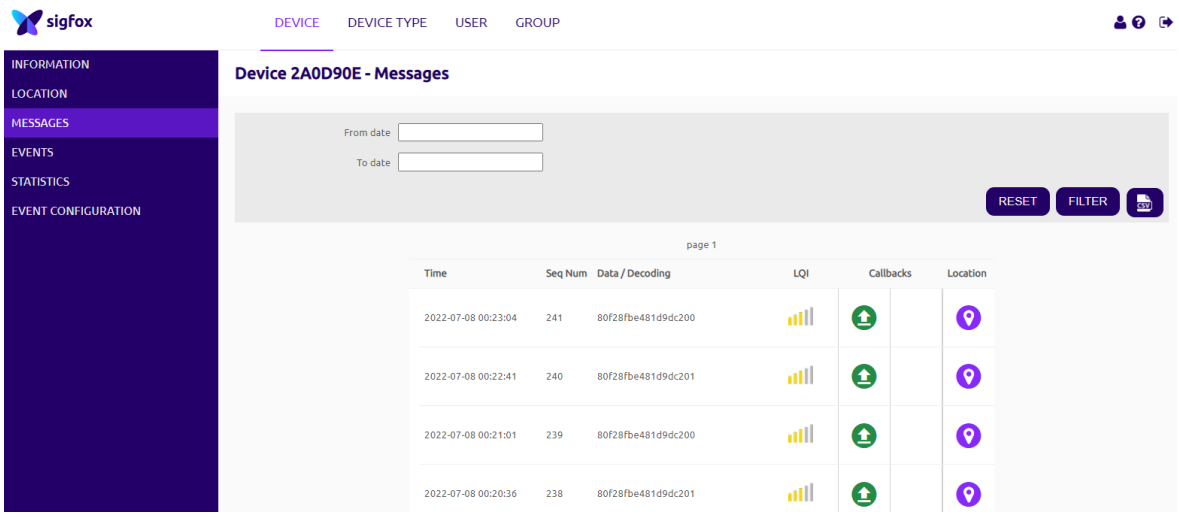


figura 3.4 Datos en forma hexadecimal de las coordenadas y verificación de alarma, registro en el backend de Sigfox correspondientes a la madrugada.

La comprobación con los datos que recepta Ubidots, se representan al mismo tiempo que fueron registrados en el backend de Sigfox con una diferencia de aproximadamente 3 segundos, es óptimo en la prueba ya que, se genera en tiempo real la recepción de datos y posterior publicación en cada widget del panel principal.

DATE	VALUE
2022-07-08 00:23:07 -05:00	0.00
2022-07-08 00:22:44 -05:00	1.00
2022-07-08 00:21:04 -05:00	0.00
2022-07-08 00:20:39 -05:00	1.00

Tabla 3.4 Datos enviados y registrados en Ubidots, con actividad pasiva y activa de la alarma, correspondientes a la madrugada.

Se puede observar en la figura 3.4 y la tabla 3.4, los datos de verificación tanto en el registro del backend de Sigfox y Ubidots que fueron marcados en el instante en el que se receptan en cada tablero.

3.1.2.1 Comparación de datos correspondientes a los estados de tiempo estudiados

De manera óptima la valoración de cada uno de los datos enviados fue transparente, sin retardos extensos y autónomos, lo que indica que el proyecto es viable para incorporarse en el domicilio y ser proyectado a mayor escala.

Si la devolución de llamada es efectiva, el enlace a Ubidots se presenta sin ningún problema. A continuación, la tabla comparativa del registro y establecimiento de devolución de llamada (callbacks).

Estado del día	Pruebas	Hora	Callbacks	Registro en Ubidots
mañana	Prueba 1	7:10:13	Registro exitoso	Si
mañana	Prueba 2	7:10:13	Registro exitoso	Si
mañana	Prueba 3	7:10:13	Registro exitoso	Si
noche	Prueba 4	20:55:25	Registro exitoso	Si
noche	Prueba 5	20:55:34	Registro exitoso	Si
noche	Prueba 6	20:55:56	Registro exitoso	Si
noche	Prueba 7	20:56:07	Registro exitoso	Si
madrugada	Prueba 8	00:20:36	Registro exitoso	Si
madrugada	Prueba 9	00:21:01	Registro exitoso	Si
madrugada	Prueba 10	00:22:41	Registro exitoso	Si
madrugada	Prueba 11	00:23:04	Registro exitoso	Si

Tabla 3.5 Resumen de registros en el backend de Sigfox y la plataforma de Ubidots, actividad de devolución de llamada exitosa, registro de datos en los widgets de Ubidots.

3.2 Conclusiones

La red Sigfox está disponible las 24 horas del día, sin embargo, si el módulo Sipy está en interiores del domicilio no se puede generar el envío de datos al backend de Sigfox, la señal que emite el dispositivo va a ser mínima, lo que ocasiona un retardo en la recepción, para la alarma de alerta es una desventaja porque los datos que recibe Ubidots deben ser constatados en tiempo real y dar aviso pronto en los dispositivos finales. Debe existir una línea de vista a exteriores por parte del mecanismo que contiene la antena Sigfox del lugar origen.

Para el entorno de una alarma comunitaria es necesaria la ayuda pronta por las unidades de vigilancia, por esa razón al momento que se presiona el botón de inicio, se registra en tiempo real las coordenadas del domicilio en el backend de Sigfox, y al mismo instante los datos que se decodifican en Ubidots, para proyectarlos en los widgets, de lo contrario no sería óptimo implementar el mecanismo de alerta, ya que, presentaría retardos en la recepción de datos.

Todos los datos obtenidos reflejaron mayor viabilidad en el entorno del exterior del domicilio, esto se debe a que existe una línea de vista transparente al momento de presionar el botón de inicio, no es necesario que el mecanismo este por completo fuera del lugar donde se instalará, sin embargo, si debe estar cerca de una ventana donde no haya obstáculos.

3.3 Recomendaciones

Antes de realizar las pruebas de funcionamiento se debe constatar una línea de vista adecuada, que no se presente obstáculos en el medio en el que se coloque la antena de comunicación Sigfox, hasta que no se establezca un enlace óptimo entre el dispositivo y el backend puede generarse una conexión obsoleta, aunque exista una buena señal, primero se debe enviar un mensaje de verificación, y una vez enlazados los dos medios ya se puede enviar mensajes las veces que sea necesario.

Para realizar la comunicación de los dispositivos Sipy y ESP8266 se debe verificar la dirección de Gateway en el código fuente de cada lenguaje de programación, el cual debe ser el mismo para enlazar la conexión vía Wifi, las coordenadas de ubicación son enviadas al backend de Sigfox sin presentar mayor problema, sin embargo, no se activará la alarma de aviso, ya que, es activada con la IP de origen en el domicilio.

4 Referencias

[1] A. Ibarrola, «TEISA,» 2021. [En línea]. Available: https://sigfox.com.py/que-es-sigfox/.

[2] T. Sigfox, «Sigfox,» [En línea]. Available: https://www.sigfox.com/en/coverage. [Último acceso: 12 01 2022].

[3] ALFAIOT, «ALFAIOT,» 27 01 2021. [En línea]. Available: https://agelectro904833371.files.wordpress.com/2019/09/sigfox_zone.png.

[4] A. E. S. d. CV, «AG electrónica,» 19 02 2018. [En línea]. Available: https://agelectronica.blog/2019/09/18/sigfox-la-red-del-iot/.

[5] Sigfox, «Calbacks en Sigfox,» 12 02 2022. [En línea]. Available: https://backend.sigfox.com/devicetype/6264aed97a36e3251564dc84/callbacks/626b3733c3045d23c7f589a6/edit.

[6] crunchbase, «pycom,» 11 01 2022. [En línea]. Available: https://www.crunchbase.com/organization/pycom-ltd.

[7] PYCOM, «sipy-datasheet,» 22 03 2018. [En línea]. Available: https://pycom.io/wp-content/uploads/2018/08/sipy-specsheet.pdf.

[8] AliExpress, «AliExpress,» 15 02 2010. [En línea]. Available: https://es.aliexpress.com/item/32861464677.html.

[9] Tecnopura, «Fuente de alimentacion para protoboard,» 13 06 2019. [En línea]. Available: https://www.tecnopura.com/producto/fuente-de-alimentacion-para-protoboard-5v-3-3v-arduino-pic/.

[10] F. Electronic, «Fuente de Alimentación para Protoboard,» 15 06 2018. [En línea]. Available: https://fcelectronik.com/fuente-alimentacion-protoboard/.

[11] U. Electronics, «Componentes electrónicos,» 05 01 2022. [En línea]. Available: https://uelectronics.com/producto/modulo-relevador-5v-ky-019/.

[12] ALIExpress, «ALIExpress,» 14 03 2022. [En línea]. Available: https://es.aliexpress.com/item/32635510211.html.

[13] Sigfox, «sigfox comprar,» 16 01 2022. [En línea]. Available: https://buy.sigfox.com/.

[14] Pycom, «pyco go ivent,» 19 02 2021. [En línea]. Available: https://docs.pycom.io/updatefirmware/device/.

[15] Sigfox, «backend de Sigofx,» 9 05 2022. [En línea]. Available: https://backend.sigfox.com/welcome/news.

[16] Atom, «Atom,» 18 05 2022. [En línea]. Available: https://atom.io/.

[17] Pycom, «Pycom go ivent,» 26 02 2022. [En línea]. Available: https://docs.pycom.io/gettingstarted/.

[18] Pycom, «pycom go ivent,» 16 02 2022. [En línea]. Available: https://docs.pycom.io/tutorials/networks/wlan/.

[19] LONGREADS, «Aprendiendo Arduino,» 12 05 2009. [En línea]. Available: https://aprendiendoarduino.wordpress.com/2015/03/26/tipos-de-datos/.

[20] Arduino, «Arduino Web Editor,» 14 03 2022. [En línea]. Available: https://www.arduino.cc/en/software.

[21] EPA, «Electrónica Practica Aplicada,» 12 02 2017. [En línea]. Available: https://www.diarioelectronicohoy.com/blog/el-esp8266-como-arduino/preferencias367x384.

[22] Ubidots, «Ubidots,» 3 07 2022. [En línea]. Available: https://industrial.ubidots.com/accounts/signup_industrial/.

[23] I. Lopez, «Go to Ubidots,» 15 03 2021. [En línea]. Available: https://help.ubidots.com/en/articles/694725-connect-a-pycom-sipy-to-ubidots-using-sigfox-over-http?_gl=1*16y5i54*_ga*MzI5NzQ5NDcyLjE1OTk2NzAzODA.*_ga_VEME7QQ5EZ*MTY1Njk5MzIyNy4xNi4xLjE2NTY5OTMzNTQuMA... [Último acceso: 10 01 2022].

ANEXO I

Código fuente, Módulo Sipy

```
from network import Sigfox
import struct
import machine
from machine import Pin
from network import WLAN
import time
led=Pin('P12', mode=Pin.OUT)
boton=Pin('P10', mode=Pin.IN, pull=Pin.PULL_UP)
led1=Pin('P9', mode=Pin.OUT)
def http_get(url):
    import socket
    _, _, host, path = url.split('/', 3)
    addr = socket.getaddrinfo(host, 80)[0][-1]
    s = socket.socket()
    s.connect(addr)
    s.send(bytes('GET /%s HTTP/1.0\r\nHost: %s\r\n\r\n' % (path, host), 'utf8'))
    while True:
        data = s.recv(100)
        if data:
            print(str(data, 'utf8'), end='')
        else:
            break
    s.close()
```

```
def enviar_dato(bandera):
    import socket
    # init Sigfox for RCZ1 (Europe)
    sigfox = Sigfox(mode=Sigfox.SIGFOX, rcz=Sigfox.RCZ4)
    # create a Sigfox socket
    s = socket.socket(socket.AF_SIGFOX, socket.SOCK_RAW)
    # make the socket blocking
    s.setblocking(True)
    # configure it as uplink only
    s.setsockopt(socket.SOL_SIGFOX, socket.SO_RX, False)
    s.send(struct.pack("<f",     -0.281147)+struct.pack("<f",     -78.557193)+struct.pack("<B", bandera))
    s.close()
wlan = WLAN() # get current object, without changing the mode
if machine.reset_cause() != machine.SOFT_RESET:
    wlan.init(mode=WLAN.STA)
    # configuration below MUST match your home router settings!!
    wlan.ifconfig(config=('192.168.100.10', '255.255.255.0', '192.168.100.1', '8.8.8.8')) # (ip, subnet_mask, gateway, DNS_server)
if not wlan.isconnected():
    # change the line below to match your network ssid, security and password
    wlan.connect('USUARIO',auth=(WLAN.WPA2, 'PASSWORD'), timeout=5000)
    print("connecting",end='')
    led1.value(0)
    while not wlan.isconnected():
        led1.value(1)
```

```
        time.sleep(0.1)
        led1.value(0)
        time.sleep(0.9)
        print(".",end='')
    print("connected")
    led1.value(1)
bandera=0
while True:
    if bandera==0:
        if boton()==1:
            led.value(1)
            resp=http_get('http://192.168.100.20/LEDON1?')
            print(resp)
            bandera=1
            enviar_dato(bandera)
            time.sleep(1)
        else:
            led.value(1)
            time.sleep(0.1)
            led.value(0)
            time.sleep(1)
    if bandera==1:
        if boton()==1:
            led.value(0)
            resp=http_get('http://192.168.100.20/LEDOFF1?')
```

```
        print(resp)
        bandera=0
        enviar_dato(bandera)
        time.sleep(1)
```

ANEXO II

Código fuente, Módulo ESP8266

```
#include <ESP8266WiFi.h>
const char* ssid = "USUARIO";
const char* password = "PASSWORD" ;
IPAddress staticIP(192,168,100,20);
IPAddress gateway(192,168,100,1);
IPAddress subnet(255,255,255,0);
WiFiServer server(80);
int LED1_Pin = 2;
void setup()
{
   Serial.begin(115200);
   Serial.println();
   Serial.printf("Connecting to %s\n", ssid);
   WiFi.config(staticIP, gateway, subnet);
   WiFi.begin(ssid, password);
   while (WiFi.status() != WL_CONNECTED)
   {
     delay(500);
     Serial.print(".");
```

```
    }
    Serial.println();
    Serial.print("Connected, IP address: ");
    Serial.println(WiFi.localIP());
    server.begin();
    Serial.println("Servidor inicializado");
   pinMode(LED1_Pin, OUTPUT);
}
  void loop()
{
    WiFiClient client = server.available();
    if (!client)  {
     return;}
     request = client.readStringUntil('\r');
     if       ( request.indexOf("LEDON1") > 0 )  { digitalWrite(LED1_Pin, HIGH);}
    else if  ( request.indexOf("LEDOFF1") > 0 ) { digitalWrite(LED1_Pin, LOW);}
    delay(500);
```

ANEXO III

Enlaces de videos y representación de complementos

- Conectando Pycom a Sigfox & Ubidots a través de HTTP

https://www.youtube.com/watch?v=g4FWYtuJT2k&t=577s

- Parámetros en Ubidots

https://www.youtube.com/watch?v=0wdl7qWS4F4&t=4542s

- Integración de ESP8266 con Ubidots

https://www.youtube.com/watch?v=eji3gmLHfNE&t=586s