COMUNICACIÓN DE 2 CUBECELL POR LoRaWAN A TRAVÉS DE MQTT

1. Introducción

Este proyecto permite que dos dispositivos Heltec CubeCell HTCC-AB02 se comuniquen de forma indirecta usando LoRaWAN, un Gateway Dragino LPS8v2 y The Things Network (TTN).

- El CubeCell Emisor envía periódicamente un mensaje (en este ejemplo, "Hello") a TTN usando la activación OTAA.
- El CubeCell Receptor se une a TTN y, a través de downlinks generados, recibe y muestra el mensaje en formato legible.
- La comunicación entre ambos dispositivos se realiza a través de TTN; el Gateway solo reenvía los paquetes.
- Se utiliza una integración MQTT (implementada mediante un script en Python) para suscribirse a los uplinks del emisor y, al detectar un mensaje, publicar un downlink dirigido al receptor.
- Además, se ha configurado un *Custom JavaScript Formatter* en TTN para decodificar el downlink y mostrar el mensaje (por ejemplo, "Hello") en formato ASCII.

---

2. Requisitos

Hardware:

- 2 placas Heltec CubeCell HTCC-AB02
- 2 antenas LoRa (una por placa)
- 1 Gateway Dragino LPS8v2 (con conexión a Internet)

Software y Herramientas:

- Arduino IDE con soporte para CubeCell (instalado desde el Gestor de Placas)
- Cuenta en The Things Network (TTN)
- Python 3 instalado en tu sistema
- La librería Python *paho-mqtt* (se instala ejecutando pip install paho-mqtt)

---

3. Configuración en TTN

A. Registro del Gateway

1. Conecta el Gateway Dragino LPS8v2 a Internet mediante WiFi configurándolo en modo WAN Client.
2. Accede a la interfaz web del Gateway, configura la red y el modo LoRaWAN y verifica que el Gateway esté conectado (indicador verde).
3. En TTN, registra el Gateway ingresando su Gateway EUI y configurándolo para la región *EU868*.

B. Creación de la Aplicación y Registro de Dispositivos

1. En la consola TTN, crea una nueva aplicación (por ejemplo, "lorawan-app-iot").
2. Registra dos dispositivos:
    - **CubeCell-1 (Emisor):** Configúralo en OTAA y asigna sus claves (DevEUI, JoinEUI y AppKey).
    - **CubeCell-2 (Receptor):** Regístralo en OTAA con claves distintas (se recomienda un JoinEUI diferente, por ejemplo, generando uno nuevo para distinguirlo).

C. Payload Formatters

1. En el dispositivo receptor (CubeCell-2) en TTN, accede a la sección *Payload Formatters > Downlink*.
2. Selecciona "Custom JavaScript Formatter" y pega el siguiente código (el cuál convierte el arreglo de bytes recibido a un string ASCII):

```
function decodeDownlink(input) {
 var bytes = input.bytes;
 var message = "";
 for (var i = 0; i < bytes.length; i++) {
  message += String.fromCharCode(bytes[i]);
 }
 return {
  data: {
   message: message
```

```
  },
  warnings: [],
  errors: []
 };
}
```

3. Guarda la configuración. Este formatter se encargará de decodificar los mensajes enviados al receptor, mostrando "Hello" (por ejemplo) en texto legible en la consola de TTN.

---

4. Configuración de MQTT mediante Script Python

Se utilizará un script en Python con la librería *paho-mqtt* para automatizar la publicación de downlinks en TTN al detectar uplinks del emisor.

A. Instalación:

- Ejecuta en la terminal:

```
pip install paho-mqtt
```

B. Configuración y Uso:

1. Define en el script tus credenciales de TTN:
    - **Application ID** (por ejemplo, “lorawan-app-iot”)
    - **API Key** (obtenida en TTN→Other integrations→MQTT→Generate new API Key)
    - **IDs de los dispositivos:** Ejemplo: “cubecell-1” (emisor) y “cubecell-2” (receptor).
2. Configura los tópicos MQTT (están en el script de Python):
    - Tópico de uplink del emisor: v3/lorawan-app-iot@ttn/devices/cubecell-1/up
    - Tópico de downlink del receptor: v3/lorawan-app-iot@ttn/devices/cubecell-2/down/push
3. El script se conectará al broker MQTT de TTN (usando el dominio eu1.cloud.thethings.network en el puerto 1883 para conexión no cifrada, o 8883 si se desea TLS).

4. Al recibir un mensaje en el tópico de uplink del emisor, el script extraerá el payload (en Base64) y lo utilizará para construir un mensaje JSON de downlink, que se publicará en el tópico del receptor.
5. El script convierte el payload de Base64 a ASCII (para que veas en la consola del script el mensaje legible) y lo vuelve a enviar tal como está al receptor.

Nota: No es necesario configurar MQTT en el Gateway. El Gateway únicamente reenvía los paquetes hacia TTN, que cuenta con su propio broker MQTT integrado.

---

5. Funcionamiento del Sistema

- **CubeCell-1 (Emisor):**
  - Se une a TTN mediante OTAA y envía uplinks periódicos (por ejemplo, cada 15 segundos) con el mensaje "Hello".
- **TTN:**
  - Recibe el uplink y, a través del script Python o mediante la consola TTN, se programa un downlink dirigido al CubeCell-2.
  - El Custom JavaScript Formatter en el receptor decodifica el payload binario a texto ASCII para su visualización.
- **CubeCell-2 (Receptor):**
  - Se une a TTN y, para abrir su ventana de recepción, envía uplinks periódicos (por ejemplo, cada 30 segundos).
  - Cuando TTN envía el downlink, el receptor lo recibe y, mediante su función de manejo de downlink, imprime el mensaje en el Monitor Serie en formato legible (mostrando "Hello").

---

6. Proceso de Prueba y Verificación

1. **Programación de los Dispositivos:**
   - Carga el código del Emisor en CubeCell-1.
   - Carga el código del Receptor en CubeCell-2.
   - (El código de emisor y receptor están en el repositorio.)

2. **Configuración en TTN:**
    - Asegúrate de que ambos dispositivos se han unido exitosamente a TTN y aparecen en Live Data.
    - Revisa los campos en TTN para confirmar que las claves y la configuración regional son correctas.
3. **Ejecución del Script MQTT (Python):**
    - Ejecuta el script Python, que se conectará al broker MQTT de TTN, se suscribirá al tópico de uplink del emisor y enviará automáticamente un downlink al receptor cada vez que reciba un mensaje.
    - Observa en la consola del script que se imprime el mensaje en ASCII ("Hello") antes de enviarlo.
4. **Verificación en el Receptor:**
    - Abre el Monitor Serie del CubeCell-2.
    - Verifica que aparezca el mensaje de downlink recibido, tanto en formato hexadecimal como en texto ASCII, gracias al *Custom JavaScript Formatter* en TTN y al código de procesamiento en el receptor.

---

7. Conclusiones

Con este sistema, se demuestra la comunicación indirecta entre dos CubeCell usando TTN y MQTT, donde:

- El Emisor envía mensajes uplink.
- Un script Python automatiza el reenvío de esos mensajes como downlinks al Receptor.
- El Receptor, mediante su ventana de recepción, capta el mensaje y lo procesa, mostrando el contenido de forma legible (por ejemplo, "Hello").

Esta documentación, junto con los códigos de Emisor y Receptor (que se encuentran en archivos separados dentro del repositorio), permite replicar y probar el sistema de comunicación LoRaWAN completo.