

# Tarea 2

## Comunicación BLE

**Profesor: Luciano Radrigan F.**  
Auxiliar: Ricardo Aravena P.

Cada grupo contara con una Raspberry Pi3 y un microcontrolador ESP32, con estos elementos se debera lograr programar:

## Objetivos

- Una comunicación BLE, en la cual el ESP32 sea un servidor BLE y la Raspberry corresponderá al cliente (recomendado la librería de Pygatt), usando los mismos paquetes de datos que la tarea anterior (Tabla 1, 2 y 3) y que cuente con los siguientes modos de comunicación (transport layer):
  - **BLE configuración (status=10):** El ESP32 estará en un modo de espera, en el cual la Raspberry deberá comunicarse con él y enviarle una configuración. Deberá estar mostrándose claramente que espera una configuración. Este es el modo en el que se debe prender por defecto.
  - **BLE Continua (status=30):** El ESP32 deberá enviar el paquete de datos de forma continua hasta que desde el cliente se detenga el envío.
  - **BLE discontinua (status=31):** Luego de enviar los datos, según un tiempo definido en la variable 'Discontinuous\_time' el ESP32 entrara en modo 'Deep Sleep'. Se recomienda que esta variable tenga como unidad minutos y valor mínimo de 1. Este modo de envíos también se deberá poder detener desde el cliente.
- Para mejorar la intermitencia en la conexión inicial entre la Raspberry y el ESP32, se debe implementar un sistema de maquina de estados (se subirá un ejemplo de esta a U-cursos).
- Los datos recepcionados deberán ser almacenados en una base de datos SQL (de la misma forma que la Tarea 1).
- La base de datos deberá contener:
  1. **Tabla de datos:** con timestamp y el identificador del dispositivo (Id\_device y MAC) como llaves.
  2. **Tabla de Logs:** donde se anotara toda comunicación, considerando el Id\_protocol y el Transport\_layer. Mismas llaves que la tabla anterior.
  3. **Tabla de configuraciones:** misma función que la tarea 1, aquí se guardan las configuraciones de los dispositivos a conectar. En caso de que valores cambien, se debe enviar el cambio de configuración a la ESP32.
  4. **Tabla de Loss:** se anotara los tiempos de comunicación cada vez que intente conectarse a un server BLE para hacer lecturas y la cantidad de intentos de conexión que lo tomo lograrlo (solo una entrada por intento de conexión).

Tabla 1: Forma de los paquetes

Header					Data			
2 bytes	6 bytes	1 byte	1 byte	2 bytes	-	-	-	-
ID Device	MAC	Transport Layer	ID Protocol	Leng Msg	Data 1	Data 2	Data ...	Data N

- Los protocolos de envío que deberán probar son:

Tabla 2: Protocolos 0,1,2 y 3 .

Tamaño de los Datos ->		1 bytes	1 bytes	4 bytes	1 bytes	4 bytes	1 bytes	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes
ID Protocol	Leng Msg (bytes)	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8	Data 9	Data 10	Data 11	Data 12	Data 13	Data 14
0	6	Val: 1	Batt level	Timestamp											
1	16	Val: 1	Batt level	Timestamp	Temp	Press	Hum	Co							
2	20	Val: 1	Batt level	Timestamp	Temp	Press	Hum	Co	RMS						
3	44	Val: 1	Batt level	Timestamp	Temp	Press	Hum	Co	RMS	Amp x	Frec x	Amp y	Frec y	Amp z	Frec z

## Generación de datos

Para generar los datos deberá implementar funciones que emulen el funcionamiento de los sensores, usando las siguientes indicaciones (estos deben acomodar a su tamaño dentro del paquete, trunque el valor de ser necesario):

- **THPC\_Sensor** (Temperatura-Humedad-Presión-*CO*): representa un sensor de cada uno de estos aspectos, genere su medición usando los siguientes valores:
  - *Temp* = Valor aleatorio entre 5 y 30
  - *Hum* = Valor aleatorio entre 30.0 a 80.0
  - *Pres* = Valor aleatorio entre 1000 y 1200
  - *CO* = Valor aleatorio entre 30.0 y 200.0
- **Batt\_sensor**: representando el nivel de batería del aparato, deberá ser un valor entre 1 y 100.
- **Acelerometer\_kpi**: representa un sensor de vibraciones, midiendo en los tres ejes y sacando su promedio (RMS, root mean square), para sus valores use:
  - $Amp_x$  = valor aleatorio entre 0.0059 y 0.12
  - $Frec_x$  = valor aleatorio entre 29.0 y 31.0
  - $Amp_y$  = valor aleatorio entre 0.0041 y 0.11
  - $Frec_y$  = valor aleatorio entre 59.0 y 61.0
  - $Amp_z$  = valor aleatorio entre 0.008 y 0.15
  - $Frec_z$  = valor aleatorio entre 89.0 y 91.0
  - $RMS = \sqrt{(Amp_x^2 + Amp_y^2 + Amp_z^2)}$

## Entrega

Con fecha de entrega del Lunes 29 de Mayo, esta consistirá:

1. Un vídeo demo del funcionamiento de lo pedido, que muestre una conexión normal y al menos un cambio de modo de funcionamiento.
2. Subir el código a U-cursos (misma fecha de entrega que la demostración). De ser posible entregue un link a un repositorio Git donde se trabajó la tarea.
3. Incluir dentro de la entrega los datos de las tablas de Loss para experimentos que incluyan pruebas con maquina de estado y otros que no la usen (al menos tres experimentos de cada uno).