GUÍA PASO A PASO DE LABORATORIO 06: NODOS INALÁMBRICOS E INTEGRACIÓN HÍBRIDA (WI-FI, BLE, LORA Y DEEP SLEEP)

Curso: Internet de las Cosas (IoT) con Arduino y Raspberry Pi — Ingeniería Mecatrobótica

Módulo 06 / Video 06: Nodos Inalámbricos Autónomos, Servidor BLE GATT (`0x181A`), Radio de Largo Alcance LoRa (`SX1278`) y Sueño Profundo (`Deep Sleep`) con Sensor Ambiental Bosch `BME280`

Plataformas: Arduino Nano ESP32 (`3.3 V` Nativo) + Raspberry Pi 4 Model B / 5 (`64-bit`)

---

1. Objetivo de Ingeniería del Laboratorio

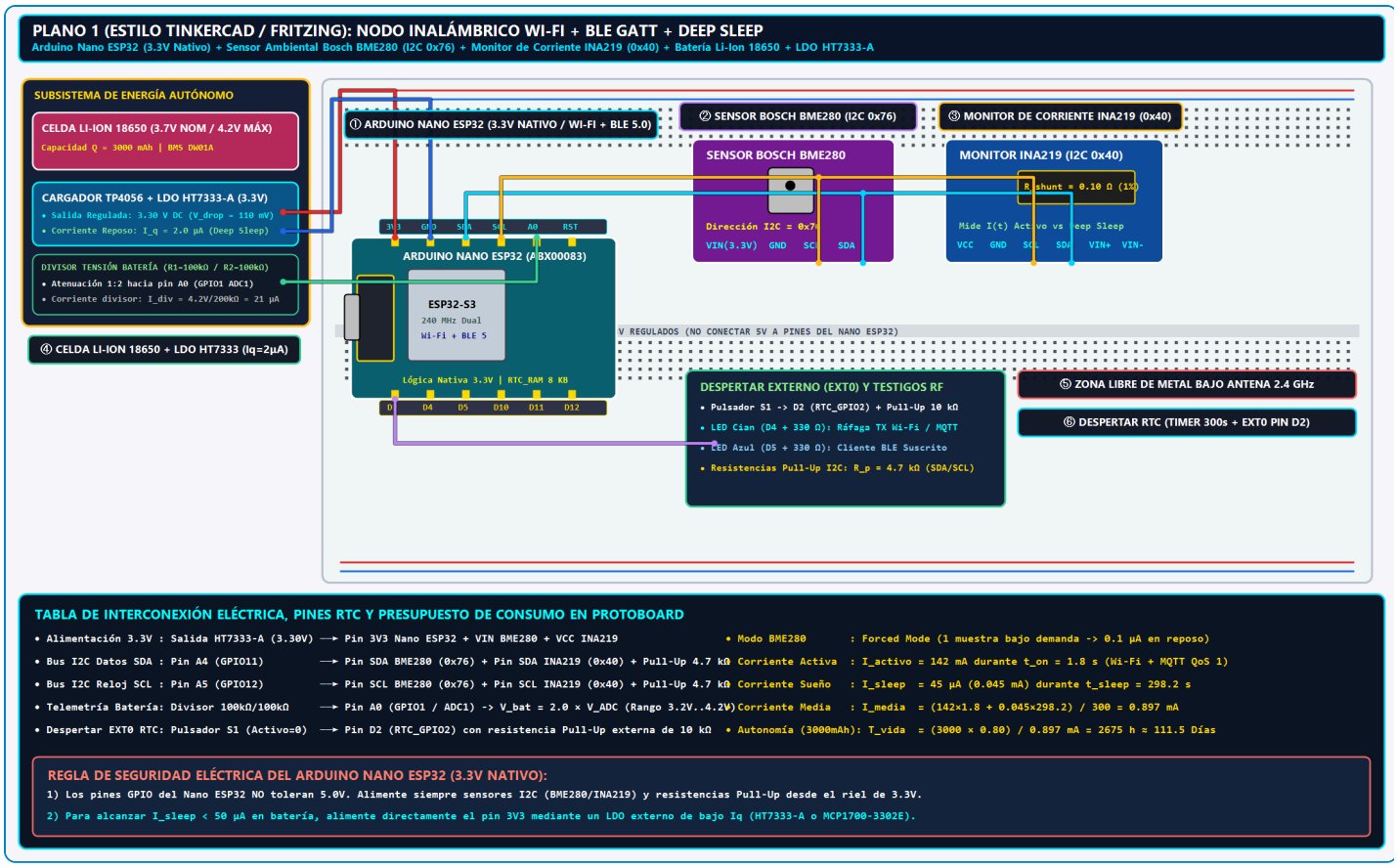
Diseñar, ensamblar en protoboard (estilo Tinkercad / Fritzing) y en placa de circuito impreso soldada (`PCB FR-4`), programar y auditar una **arquitectura inalámbrica híbrida de tres bandas/protocolos (`Wi-Fi` , `BLE 5.0` y `LoRa Sub-GHz`)** convergente hacia el Broker Eclipse Mosquitto en la Raspberry Pi:

- **Nodo Wi-Fi + MQTT de Ultrabajo Consumo (`Deep Sleep`)**: Basado en Arduino Nano ESP32 alimentado por celda `Li-Ion 18650 (3.7 V / 3000 mAh)` y regulador LDO `HT7333-A (I_q = 2.0 µA)` , leyendo temperatura, humedad y presión barométrica mediante el sensor Bosch `BME280` en **Modo Forzado (`Forced Mode`)** y reconectando con **Backoff Exponencial + Jitter**.
- **Nodo Periférico Bluetooth Low Energy (`BLE GATT 0x181A`)**: Exponiendo las características estándar `0x2A6E` (Temperatura), `0x2A6F` (Humedad) y `0x2A6D` (Presión) con notificaciones asíncronas (`CCCD 0x2902`) hacia el demonio Python `bleak` en la Raspberry Pi.
- **Concentrador de Largo Alcance** `LoRa SX1278 (433 / 915 MHz)` : Acoplado por bus `SPI0` e interrupción física `DIO0 (GPIO 25)` a la Raspberry Pi para enlaces rurales de `2 km` a `15 km` .

---

2. Montaje Físico en Protoboard (Estilo Tinkercad) y en Placa PCB Soldada

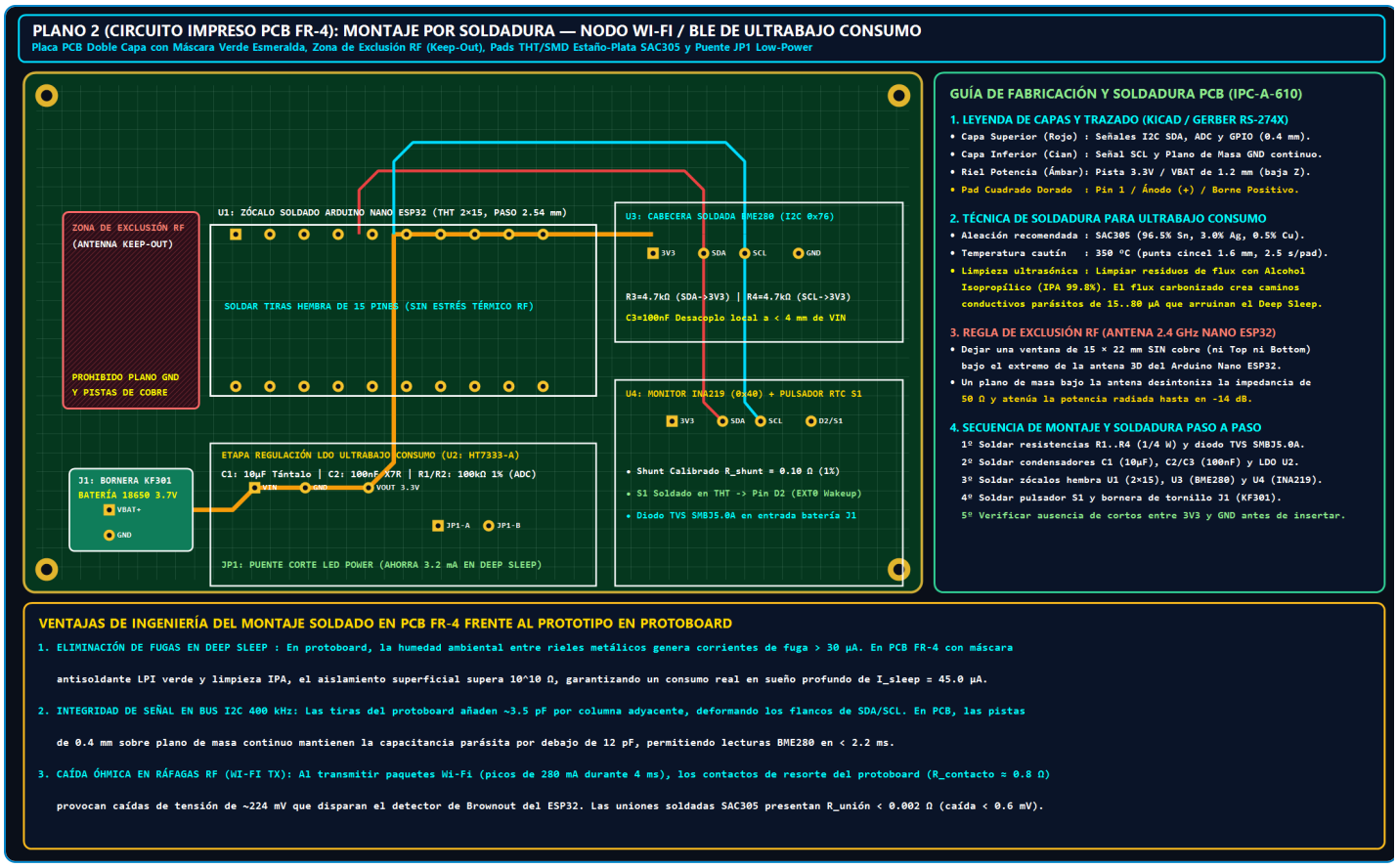
📐 **Plano 1: Montaje en Protoboard Estilo Tinkercad del Nodo Wi-Fi / BLE con BME280, INA219 y Batería 18650**



2.1 Tabla Maestra de Conexiones del Nodo Arduino Nano ESP32 (`Lógica Estricta 3.3 V`)

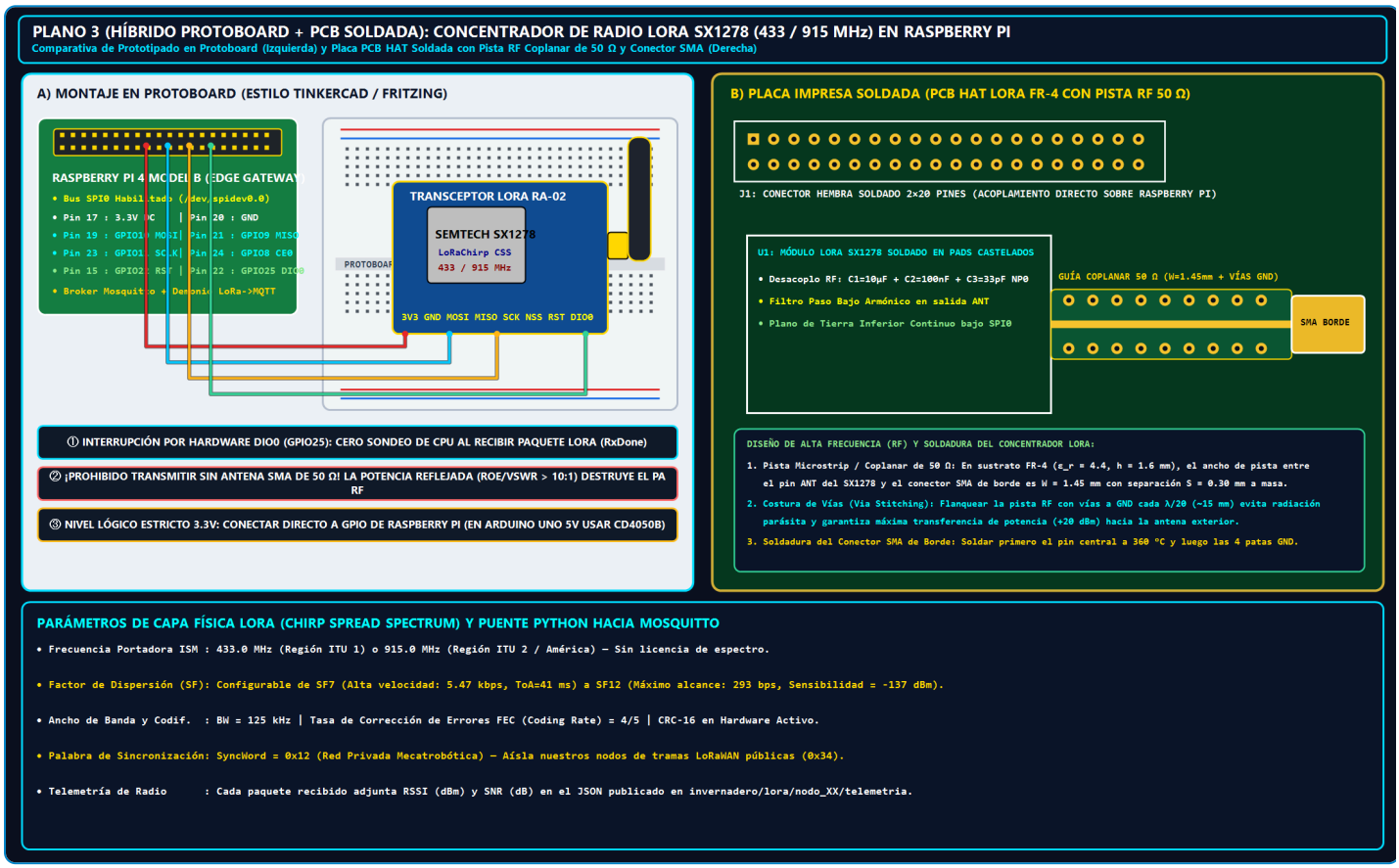
Subsistema / Componente	Pin del Componente	Pin en Arduino Nano ESP32	Función Eléctrica y Observaciones de Ingeniería
Regulador LDO HT7333-A	`VOUT (3.30 V)`	`Pin 3V3`	Alimentación directa regulada con corriente de reposo $I_q = 2.0\ \mu A$
Sensor Bosch BME280 (`0x76`)	`VIN / GND / SDA / SCL`	`3V3 / GND / A4 / A5`	Bus I^2C a $400\ kHz$ con resistencias Pull-Up de $4.7\ k\Omega$ a $3.3\ V$
Monitor Corriente INA219 (`0x40`)	`VCC / GND / SDA / SCL`	`3V3 / GND / A4 / A5`	Shunt de precisión $R_{shunt} = 0.10\ \Omega\ (1\%)$ en serie con el riel de $3.3\ V$
Divisor Tensión Batería	Punto medio `100kΩ/100kΩ`	`Pin A0 (GPIO1 / ADC1)`	Lectura calibrada con `analogReadMilliVolts(A0)` ($V_{bat} = 2.0 \times V_{ADC}$)
Pulsador Despertar RTC (`S1`)	Contacto NA a `GND`	`Pin D2 (RTC_GPIO2)`	Interrupción externa `EXT0` con resistencia Pull-Up externa de $10\ k\Omega$ a $3.3\ V$
Indicadores LED RF	Ánodo vía $330\ \Omega$	`Pin D4 (Wi-Fi)` / `D5 (BLE)`	Testigos luminosos activos únicamente durante la ventana despierta

📐 Plano 2: Placa de Circuito Impreso PCB FR-4 con Montaje por Soldadura del Nodo Inalámbrico



3. Montaje del Concentrador LoRa `SX1278` en Raspberry Pi (`SPI0` + `IRQ DIO0`)

📐 Plano 3: Interconexión en Protoboard y Placa PCB HAT Soldada del Transceptor LoRa SX1278



> **Regla Crítica de Radiofrecuencia (ROE / VSWR):** Jamás alimente ni ejecute el script de transmisión/recepción del módulo `SX1278` sin haber enroscado previamente su **antena helicoidal o dipolo de $50\ \Omega$ en el conector SMA**. Transmitir a $+20\ \text{dBm}$ con el puerto RF abierto refleja el 100% de la energía hacia el transistor de salida del amplificador de potencia (PA), destruyendo el chip en milisegundos.

Pin del Módulo LoRa `SX1278 (Ra-02)`	Pin Físico en Raspberry Pi (40 Pines)	GPIO BCM	Función en el Bus `SPI0` y Control de Radio
`VCC (3.3V)`	**Pin 17**	Riel `3.3V DC`	Alimentación limpia capaz de entregar picos de $120\ \text{mA}$
`GND`	**Pin 20**	Riel `GND`	Masa común de referencia digital y blindaje RF
`MOSI`	**Pin 19**	`GPIO 10 (SPI0_MOSI)`	Escritura de registros y carga de FIFO hacia el SX1278
`MISO`	**Pin 21**	`GPIO 9 (SPI0_MISO)`	Lectura de registros, RSSI, SNR y paquete recibido
`SCK`	**Pin 23**	`GPIO 11 (SPI0_SCLK)`	Reloj síncrono SPI a $4.0\ \text{MHz}$ (Modo `0,0`)
`NSS (CS)`	**Pin 24**	`GPIO 8 (SPI0_CE0_N)`	Selección de esclavo activa en nivel bajo
`RST`	**Pin 15**	`GPIO 22`	Reinicio físico por hardware del transceptor
`DIO0`	**Pin 22**	`GPIO 25`	Interrupción por flanco de subida (`RxDone` / `TxDone`)

---

4. Despliegue de Software y Verificación en Laboratorio

Paso 1: Habilitar `SPI0` y `Bluetooth BlueZ` en Raspberry Pi e instalar dependencias

```
sudo raspi-config nonint do_spi 0
sudo systemctl enable --now bluetooth
sudo apt update && sudo apt install -y python3-pip python3-spidev python3-rpi.gpio bluez
pip3 install --break-system-packages bleak paho-mqtt
```

Paso 2: Compilar y cargar los firmwares en Arduino Nano ESP32

- En Arduino IDE 2.x, instalar el paquete de placas **Arduino ESP32 Boards** y las librerías `PubSubClient`, `Adafruit BME280 Library` y `Adafruit INA219`.
- Para ensayar el nodo **Wi-Fi + MQTT con Deep Sleep**, abrir `codigo_fuente/nodo_wifi_mqtt_bme280_deepsleep.ino`, ajustar `WIFI_SSID`, `WIFI_PASSWORD` y `MQTT_BROKER_IP`, y subir al Nano ESP32.
- Para ensayar el nodo **Periférico BLE GATT (`0x181A`)**, subir `codigo_fuente/periferico_ble_gatt_ambiental.ino`.

Paso 3: Lanzar las pasarelas de borde en la Raspberry Pi y auditar en Mosquitto

```
# Terminal 1: Iniciar pasarela asíncrona BLE GATT -> MQTT
python3 codigo_fuente/pasarela_ble_a_mqtt_bleak.py

# Terminal 2: Iniciar concentrador LoRa SX1278 SPI0 -> MQTT
python3 codigo_fuente/transceptor_lora_sx1278_mqtt_bridge.py

# Terminal 3: Auditar en tiempo real la convergencia de las 3 tecnologías RF en el Broker
mosquitto_sub -h localhost -u scada_central -P Clave -t 'invernadero/#' -q 1 -v
```