Prova

Resumo da Ópera

Jan K. S. – janks@puc-rio.br

ENG4051 – Projeto Internet das Coisas

Sensor de Luz

```
int leitura = analogRead(pino);
int porcentagemLuz = map(leitura, 0, 4095, 0, 100);
```

LED

```
void setup () {
  pinMode(pinoLED, OUTPUT);
  digitalWrite(pinoLED, HIGH);
}

  digitalWrite(pinoLED, LOW);
```

Millis

```
unsigned long instanteAnterior = 0;

void loop () {
  unsigned long instanteAtual = millis();
  if (instanteAtual > instanteAnterior + 1000) {
    Serial.println("+1 segundo");
    instanteAnterior = instanteAtual;
  }
}
```

Serial

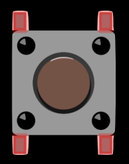
```
void setup () {
  Serial.begin(115200); while(!Serial);
}
void loop () {
  if (Serial.available() > 0) {
    String texto = Serial.readStringUntil('\n');
    Serial.println(texto);
  }
}
```

String

```
String texto1 = "Olá, mundo!";
int numero = 100 * 2;
String texto2 = String(numero);
int numero2 = texto2.toInt() + 42;

String texto3 = "aaa" + texto2;

bool ehIgual = texto2 == texto3;
bool comecaComOla = texto1.startsWith("Olá");

char caracter = texto1[2];              // 'á'
int totalCaracteres = texto1.length(); // 11

String trecho = texto1.substring(0, 3);   // "Olá"
String trechoFinal = texto1.substring(5); // "mundo!"

String texto4 = "  abc abc  \n";
texto4.replace("ab", "AB"); // "ABc ABc"
```

ARDUINO™

Botão



```
#include <GFButton.h>

GFButton botao(1);

void botaoPressionado (GFButton& botaoDoEvento) {
  Serial.println("Botão foi pressionado!");
}
void botaoSolto (GFButton& botaoDoEvento) {
  Serial.println("Botão foi solto!");
}

void setup () {
  Serial.begin(115200);
  botao.setPressHandler(botaoPressionado);
  botao.setReleaseHandler(botaoSolto);
}
void loop () {
  botao.process();
}
```

LED RBG

```
neopixelWrite(RGB_BUILTIN, vermelho, verde, azul);
```

Sensor de Movimento

```
#include <GFButton.h>

GFButton sensor(21);

void movimento (GFButton& sensor) {
  Serial.println("Movimento detectado!");
}
void inercia (GFButton& sensor) {
  Serial.println("Inércia detectada!");
}

void setup () {
  Serial.begin(115200);
  sensor.setPressHandler(inercia);
  sensor.setReleaseHandler(movimento);
}
void loop () {
  sensor.process();
}
```

Botão com Passagem de Parâmetro

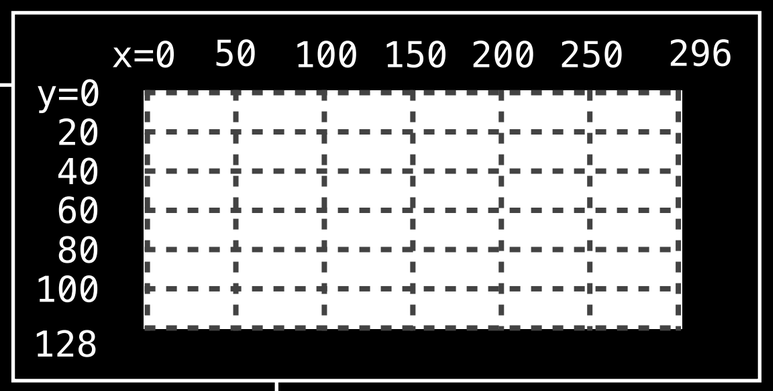
```
void minhaFuncao (int x) {
  Serial.println(x);
}

void setup () {
  Serial.begin(115200);
  botao.setPressHandler([](GFButton &b){ minhaFuncao(42); });
}
```

Verificação Manual

```
botao.isPressed();
sensor.isPressed();
```

Setup

```
#include <GxEPD2_BW.h>
#include <U8g2_for_Adafruit_GFX.h>

U8G2_FOR_ADAFRUIT_GFX fontes;
GxEPD2_290_T94_V2 modeloTela(10, 14, 15, 16);
GxEPD2_BW<GxEPD2_290_T94_V2, GxEPD2_290_T94_V2::HEIGHT> tela(modeloTela);

void setup() {
  tela.init();
  tela.setRotation(3);
  tela.fillScreen(GxEPD_WHITE);
  tela.display(true);

  fontes.begin(tela);
  fontes.setForegroundColor(GxEPD_BLACK);
}
```



Fontes de Símbolos

u8g2_font_open_iconic_all_4x_t

↓

https://github.com/olikraus/u8g2/wiki/fntpic/u8g2_font_open_iconic_all_4x_t.png

Fontes de Texto

```
u8g2_font_helvB24_te
u8g2_font_helvB18_te
u8g2_font_helvB14_te
u8g2_font_helvB12_te
```

Textos

```
fontes.setFont( u8g2_font_helvB24_te );
fontes.setFontMode(1);
fontes.setCursor(x, y);
fontes.print("Meu texto");

tela.display(true); // SEMPRE CHAMAR NO FINAL!
```

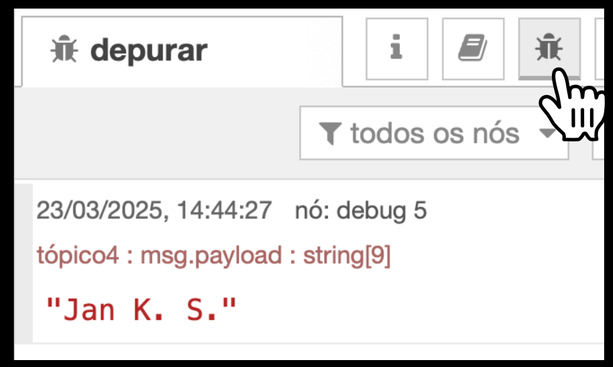
Desenhos

```
tela.drawLine(x1, y1, x2, y2, cor);

tela.fillCircle(x, y, raio, cor);
tela.drawCircle(x, y, raio, cor);

tela.fillRect(x, y, comprimento, altura, cor);
tela.drawRect(x, y, comprimento, altura, cor);

tela.fillTriangle(x1, y1, x2, y2, x3, y3, cor);
tela.drawTriangle(x1, y1, x2, y2, x3, y3, cor);

tela.display(true); // SEMPRE CHAMAR NO FINAL!
```

Display ePaper
2.9" WeAct

mqtt → { topic: "tópico1/10", payload: "conteúdo"}

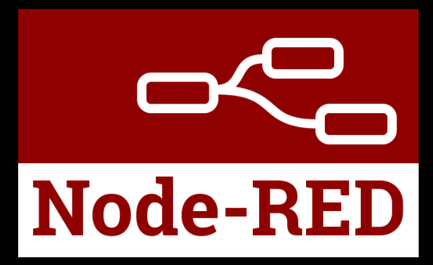
{ topic: "tópico2", payload: "conteúdo"} → mqtt



{ topic: "tópico4", payload: "Jan K. S."} → debug

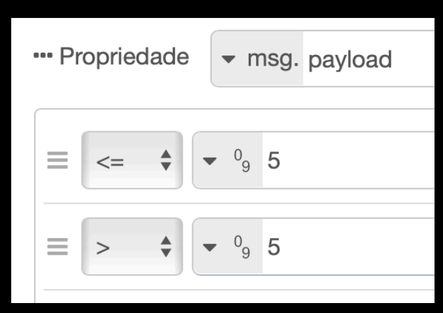
inject → { topic: "tópico3", payload: "teste!"}

{ payload: "texto" } → change → { payload: "novo texto" }



{ payload: 8 } → switch (não emite nada)

{ payload: 8 }



{ payload: [10, 20, 30] } → split →
{ payload: 10 }
{ payload: 20 }
{ payload: 30 }

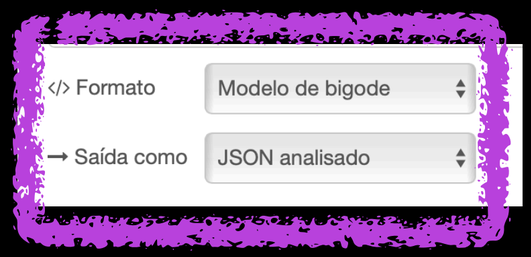
{ payload: 10 }
{ payload: 20 }
{ payload: 30 } → join → { payload: [10, 20, 30] }

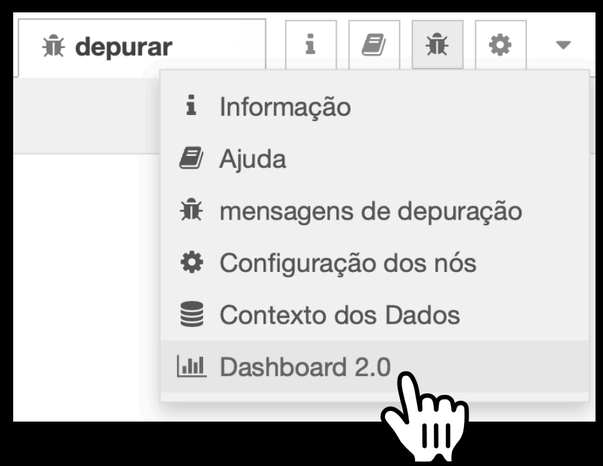
receiver → { content: "Olá, Node-RED!", ... }

{ payload: <imagem> } → image

{ payload: 42 } → { template → sender

```
{
  "content": "Valor = {{payload}}",
  "chatId": "ID DO SEU CHAT",
  "type": "message"
}
```



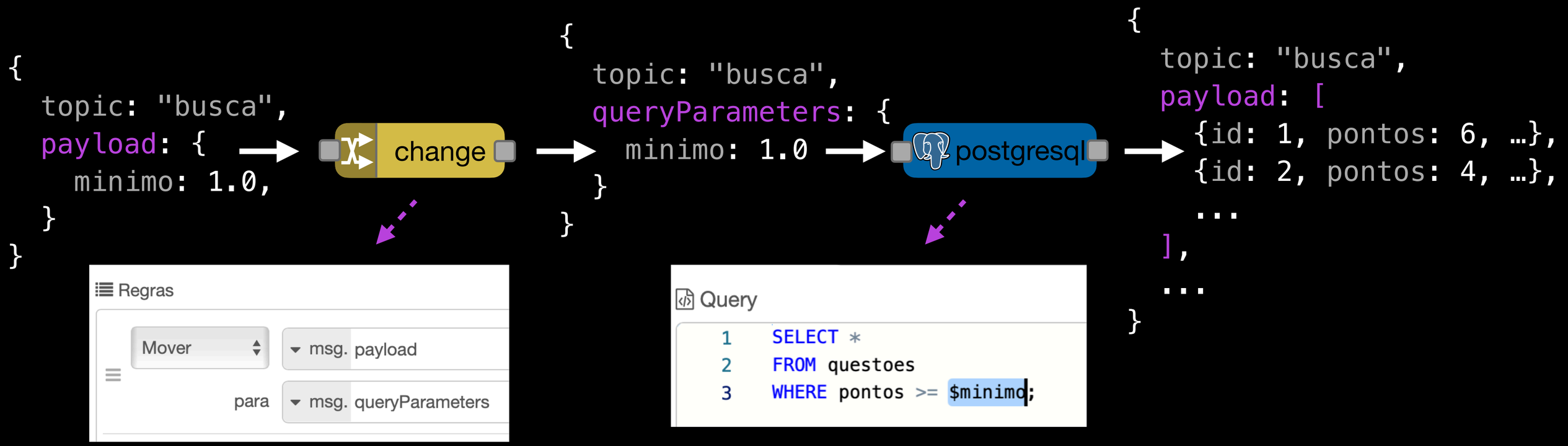


Inserção de Dados

```sql
INSERT INTO provas (id, nome, inicio) VALUES
(1, 'P1 de Programação', '2024-04-22 17:00:00'),
(2, 'P2 de Programação', '2024-05-17 17:00:00');
```

Busca de Dados

```sql
SELECT *
FROM questoes;


SELECT numero, pontos, enunciado
FROM questoes
WHERE pontos >= 2.0 AND id = 1
ORDER BY id_prova ASC, numero ASC;
```



Timescale

Busca com Janelamento

```sql
SELECT
  time_bucket('1 hour', data_hora) AS time,
  AVG(luz) AS media_luz,
  SUM(movimento) AS soma_movimento
FROM dados
WHERE data_hora > NOW() - INTERVAL '1 day'
GROUP BY time
ORDER BY time ASC;
```

Outras Funções de Janelamento

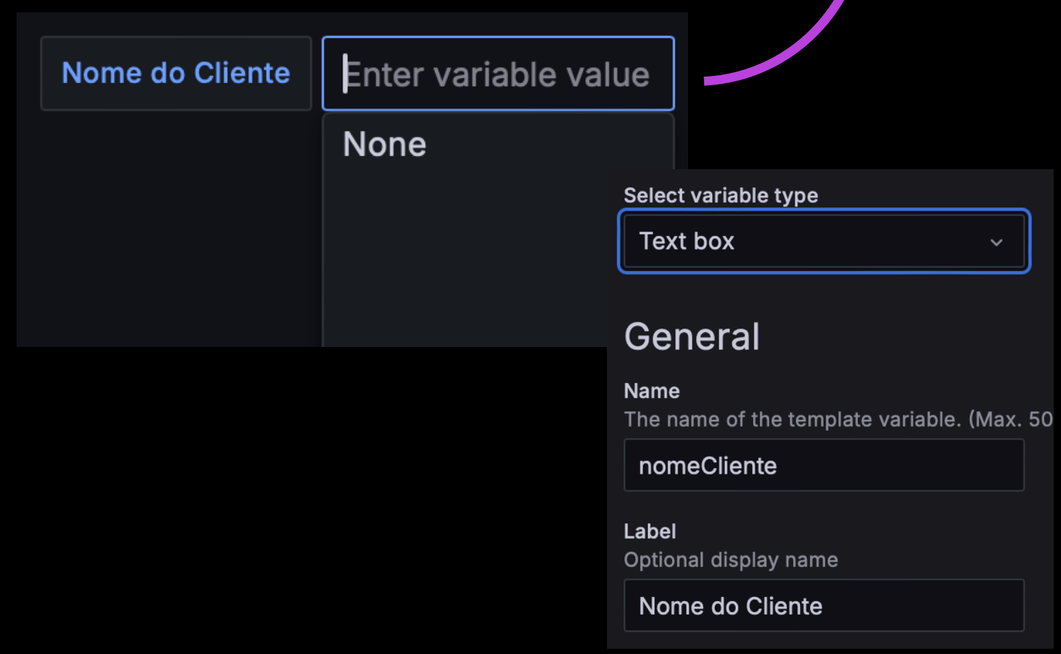
```sql
time_bucket('1 hour', data_hora) AS time
first(temperatura, time) AS primeira_temperatura
last(temperatura, time) AS ultima_temperatura

time_bucket_gapfill('1 hour', data_hora) AS time
```

Grafana

Filtro de Tempo do Grafana

```sql
SELECT
  time_bucket('1 minute', data_hora) AS time,
  AVG(luz) AS media_luz
FROM dados
WHERE $__timeFilter(data_hora)
GROUP BY time
ORDER BY time ASC;
```

```sql
SELECT saldo
FROM clientes
WHERE nome = '$nomeCliente'
```

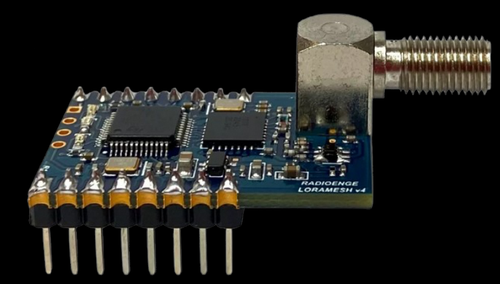


Modem LoRaWAN

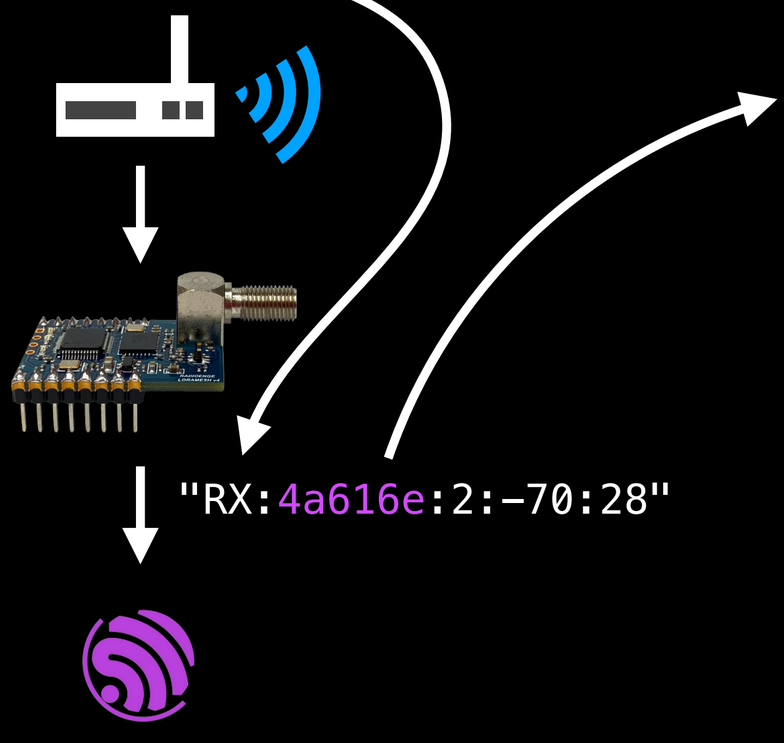
```ino
void setup() {
  Serial1.begin(9600, SERIAL_8N1, 47, 48);
  Serial.begin(115200); delay(500);

  Serial1.println("AT+JOIN");
}

void loop() {
  if (Serial1.available() > 0) {
    String texto = Serial1.readStringUntil('\n');
    texto.trim();
    Serial.println("Resposta do módulo LoRaWAN: " + texto);
  }
}
```

Envio de Dados

```ino
Serial1.println("AT+SEND=1:Oi!"); // porta : texto
Serial1.println("AT+SENDB=1:A0FF4D"); // porta : dado hexadecimal
```



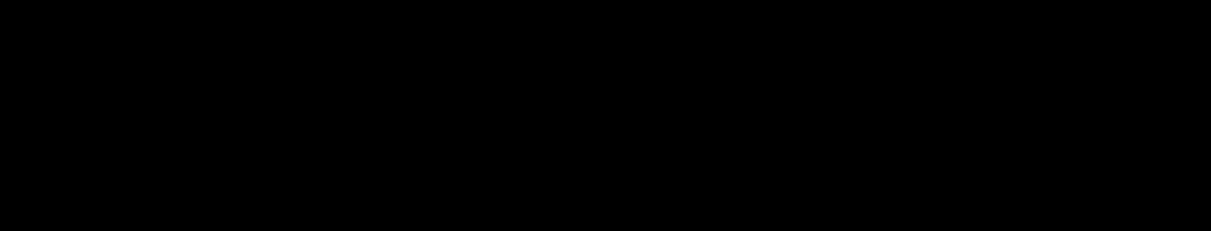
Modem LoRaWAN



Conversão de Hex para Texto

```ino
String hexadecimalParaTexto(String textoHex) {
  String resultado = "";

  textoHex.replace(" ", "");

  for (int i = 0; i < textoHex.length(); i += 2) {
    String par = textoHex.substring(i, i + 2);
    char caractere = (char)strtol(par.c_str(), NULL, 16);
    resultado += caractere;
  }

  return resultado;
}
```

```ino
#include <CayenneLPP.h>

CayenneLPP dados(50); // limite de 50 bytes

dados.addTemperature(1, 27.5);  // temperatura 1 (canal 1)
dados.addTemperature(2, 25.5);  // temperatura 2 (canal 2)
dados.addPresence(1, true);
dados.addBarometricPressure(1, 1035);
dados.addRelativeHumidity(1, 76);
dados.addAnalogInput(1, 34.55);

uint8_t* buffer = dados.getBuffer();
String mensagem = "";
for (int i = 0; i < dados.getSize(); i++) {
  if (buffer[i] < 16) {
    mensagem += "0";
  }
  mensagem += String(buffer[i], HEX);
}
mensagem.toUpperCase();
Serial.println(mensagem);

dados.reset();
```



v3/ID_APP/devices/ID_DO_DISPOSITIVO/down/push

```json
{
  "downlinks": [{
    "f_port": 5,
    "frm_payload": "VGVycmVtb3RvIQ==",
    "priority": "HIGH"
  }]
}
```

Setup Hibernação

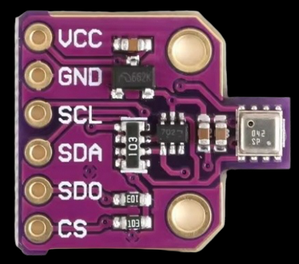
```ino
int pinoParaAcordar = 4;

// mantém valor mesmo após dormir
RTC_DATA_ATTR int contador = 0;

void setup() {
  Serial.begin(115200); delay(500);

  // agenda para acordar depois de 10000000 µs (10 segundos)
  esp_sleep_enable_timer_wakeup(10e6);

  // ou... acorda quando tiver HIGH no pino desejado
  pinMode(pinoParaAcordar, INPUT);
  esp_sleep_enable_ext0_wakeup((gpio_num_t) pinoParaAcordar, HIGH);

  Serial.printf("Contador: %d\n", contador2);
  contador++;
}
```

Ativar Hibernação

```ino
esp_deep_sleep_start();
```

Setup

```ino
#include <Adafruit_BME680.h>

Adafruit_BME680 sensorBME;

void setup() {
  Serial.begin(115200); delay(500);

  if (!sensorBME.begin()) {
    Serial.println("Erro no sensor BME");
    while (true);
  }

  // aumenta amostragem dos sensores (1X, 2X, 4X, 8X, 16X ou NONE)
  sensorBME.setTemperatureOversampling(BME680_OS_8X);
  sensorBME.setHumidityOversampling(BME680_OS_2X);
  sensorBME.setPressureOversampling(BME680_OS_4X);

  sensorBME.setIIRFilterSize(BME680_FILTER_SIZE_3);
  sensorBME.setGasHeater(320, 150); // °C e ms, (0, 0) para desativar
}
```



Sensor BME650

Medição dos Dados

```ino
sensorBME.performReading();

float temperatura = sensorBME.temperature;                    // °C
float pressao = sensorBME.pressure / 100.0;                   // hPa
float altitude = sensorBME.readAltitude(1013.25);             // m
float umidade = sensorBME.humidity;                           // %
float resistencia_gas = sensorBME.gas_resistance / 1000.0; // kΩ
```