#include <WiFi.h>

#include <PubSubClient.h>

#include <Wire.h>

// Definições para o ADXL345

#define ADXL345\_ADDRESS 0x53

#define REG\_DATAX0 0x32

#define REG\_DATAX1 0x33

#define REG\_DATAY0 0x34

#define REG\_DATAY1 0x35

#define REG\_DATAZ0 0x36

#define REG\_DATAZ1 0x37

#define REG\_POWER\_CTL 0x2D

#define POWER\_CTL\_MEASURE\_MODE 0x08

// Wi-Fi

const char\* ssid = "SUA\_REDE\_WIFI";

const char\* password = "SUA\_SENHA\_WIFI";

// MQTT

const char\* mqtt\_server = "IP\_DO\_BROKER"; // ex: "192.168.1.100"

WiFiClient espClient;

PubSubClient client(espClient);

// ================= SETUP =================

void setup() {

Serial.begin(115200);

Wire.begin();

// Conectar Wi-Fi

setup\_wifi();

// Configurar MQTT

client.setServer(mqtt\_server, 1883);

// Inicializar ADXL345

initADXL345();

}

// ================= LOOP =================

void loop() {

if (!client.connected()) {

reconnect();

}

client.loop();

int16\_t x, y, z;

readADXL345(&x, &y, &z);

// Montar mensagem JSON

char msg[100];

snprintf(msg, sizeof(msg), "{\"x\": %d, \"y\": %d, \"z\": %d}", x, y, z);

// Publicar no tópico MQTT

client.publish("sensor/vibracao", msg);

Serial.println(msg);

delay(500);

}

// ================= FUNÇÕES WIFI =================

void setup\_wifi() {

delay(10);

Serial.println("Conectando ao WiFi...");

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("\nWiFi conectado");

Serial.print("IP: ");

Serial.println(WiFi.localIP());

}

// ================= FUNÇÕES MQTT =================

void reconnect() {

while (!client.connected()) {

Serial.print("Conectando ao MQTT...");

if (client.connect("ESP32Client")) {

Serial.println("conectado!");

} else {

Serial.print("falha, rc=");

Serial.print(client.state());

Serial.println(" tentando novamente em 5s");

delay(5000);

}

}

}

// ================= FUNÇÕES ADXL345 =================

void initADXL345() {

writeRegister(REG\_POWER\_CTL, POWER\_CTL\_MEASURE\_MODE);

}

void readADXL345(int16\_t \*x, int16\_t \*y, int16\_t \*z) {

\*x = readRegister(REG\_DATAX0, REG\_DATAX1);

\*y = readRegister(REG\_DATAY0, REG\_DATAY1);

\*z = readRegister(REG\_DATAZ0, REG\_DATAZ1);

}

int16\_t readRegister(byte reg\_low, byte reg\_high) {

Wire.beginTransmission(ADXL345\_ADDRESS);

Wire.write(reg\_low);

Wire.endTransmission(false);

Wire.requestFrom(ADXL345\_ADDRESS, 2);

byte lowByte = Wire.read();

byte highByte = Wire.read();

return (int16\_t)((highByte << 8) | lowByte);

}

void writeRegister(byte reg, byte value) {

Wire.beginTransmission(ADXL345\_ADDRESS);

Wire.write(reg);

Wire.write(value);

Wire.endTransmission();

}