**O QUE É O ESP32?**

**-**O **ESP32** é uma **plaquinha inteligente** que funciona como o **"cérebro"** de projetos eletrônicos.  
Você pode **programá-la** para **ligar coisas, ler sensores** e até **se conectar ao Wi-Fi ou Bluetooth**, como um celular.

**O QUE É GIPIOs?**

**-GPIO** é o nome dos **pinos** que ficam na placa do **ESP32** (ou Arduino).  
Esses pinos servem para **ligar coisas** ou **ler informações** de sensores e botões.

**O QUE DÁ PRA FAZER COM GPIOs?**

**Como entrada (INPUT):**

* Detectar se um **botão** foi apertado.
* Ler o valor de um **sensor** (ex: PIR, gás, umidade, etc).
* Receber sinais digitais (1 ou 0) ou analógicos (variações de tensão).

**Como saída (OUTPUT):**

* Acender um **LED**.
* Ligar um **buzzer** ou **motor**.
* Acionar um **relé** que liga/desliga lâmpadas, ventiladores, etc.

**DIFERENÇA DO ARDUINO PARA O ESP32:**

### ****Arduino (ex: Uno)****

* É como um **cérebro simples**.
* Não tem Wi-Fi nem Bluetooth.
* Funciona bem com **coisas básicas**, como:
  + Acender LED
  + Ler sensores
  + Controlar motores

### ****ESP32****

* É como um **cérebro mais esperto e moderno**.
* Já vem com **Wi-Fi e Bluetooth**.
* É mais rápido, tem mais memória e mais funções.
* Ideal para projetos **com internet**, celular, ou mais sensores ao mesmo tempo.

Se você busca um microcontrolador com maior poder de processamento e conectividade integrada para projetos mais complexos, o ESP32 é a escolha ideal. Para projetos mais simples ou para quem está começando no mundo da eletrônica, o Arduino oferece uma plataforma acessível e fácil de usar.

**DIFERENÇA DOS PROCESSADORES**

| **Característica** | **Arduino Uno** | **ESP32** |
| --- | --- | --- |
| Núcleos do processador | 1 | 2 |
| Velocidade | 16 MHz | Até 240 MHz |
| Memória RAM | Pouca | Muito mais |
| Uso típico | Projetos simples | Projetos complexos, internet, comunicação |

 Arduino = um trabalhador que faz **uma coisa de cada vez, devagarzinho, mas direitinho**.

 ESP32 = **dois trabalhadores rápidos** que fazem muitas coisas ao mesmo tempo.

ESP32 ARDUINO UNO

Processador Dual-core 32 bits a até 240 MHz 8 bits a 16 MHz

Conectividade Wi-Fi e Bluetooth integrados Necessita módulos externos

GPIOs Até 34 pinos 14 digitais, 6 analógicos

**CODIGO CONVERTIDO PARA ESP32:**

**#include <Servo.h>**

**#include <LiquidCrystal\_I2C.h>**

**#include <DHT.h>**

**// Definição dos pinos para ESP32 (mudados para GPIO disponíveis e compatíveis)**

**const int mq135 = 36; // GPIO36 (ADC1\_CH0), entrada analógica**

**const int buzzer = 14; // GPIO14, saída digital**

**const int PIR = 27; // GPIO27, entrada digital**

**const int LED = 26; // GPIO26, saída digital**

**const int rele = 25; // GPIO25, saída digital**

**#define DHTPINO 4 // GPIO4 para sensor DHT11**

**#define DHTTYPE DHT11**

**DHT dht(DHTPINO, DHTTYPE);**

**LiquidCrystal\_I2C Lcd(0x27, 20, 4);**

**Servo motor;**

**void verificarVazamentoDeGas() {**

**int estadoMQ135 = analogRead(mq135);**

**Serial.println(estadoMQ135);**

**if (estadoMQ135 >= 50) {**

**alarme\_dois\_tons();**

**} else {**

**noTone(buzzer);**

**}**

**}**

**void alarme\_dois\_tons() {**

**int freqAlta = 2000;**

**int freqBaixa = 800;**

**int duracaoTom = 250;**

**tone(buzzer, freqAlta, duracaoTom);**

**delay(duracaoTom);**

**tone(buzzer, freqBaixa, duracaoTom);**

**delay(duracaoTom);**

**}**

**void verificarTemperaturaEUmidade() {**

**float temperatura = dht.readTemperature();**

**float umidade = dht.readHumidity();**

**Lcd.clear();**

**Lcd.setCursor(0, 0);**

**Lcd.print("temperatura: " + String(temperatura) + "C");**

**Lcd.setCursor(0, 1);**

**Lcd.print("umidade: " + String(umidade) + "%");**

**delay(5000);**

**}**

**void abrirPortaAutomatica() {**

**motor.write(180);**

**delay(1500);**

**}**

**void fecharPortaAutomatica() {**

**motor.write(0);**

**delay(1500);**

**}**

**void trancarPorta() {**

**digitalWrite(rele, HIGH);**

**Serial.println("Porta trancada...");**

**delay(1500);**

**}**

**void destrancarPorta() {**

**digitalWrite(rele, LOW);**

**Serial.println("Porta destrancada...");**

**}**

**void setup() {**

**Serial.begin(115200); // ESP32 suporta velocidades maiores**

**dht.begin();**

**Lcd.init();**

**Lcd.backlight();**

**pinMode(LED, OUTPUT);**

**pinMode(PIR, INPUT);**

**pinMode(buzzer, OUTPUT);**

**pinMode(mq135, INPUT);**

**motor.attach(13); // Servo no GPIO13 (PWM compatível no ESP32)**

**pinMode(rele, OUTPUT);**

**digitalWrite(rele, HIGH);**

**motor.write(0);**

**Serial.println("Calibrando o sensor PIR...");**

**Serial.println("Por favor, nao se mova na frente do sensor.");**

**delay(15000);**

**Serial.println("Sensor PIR calibrado e pronto!");**

**Lcd.setCursor(0, 0);**

**Lcd.print("Iniciando leituras");**

**Lcd.setCursor(0, 1);**

**Lcd.print("Obrigada");**

**}**

**void loop() {**

**// acenderLEDaoDetectarPresenca();**

**// verificarVazamentoDeGas();**

**// alarme\_dois\_tons();**

**// verificarTemperaturaEUmidade();**

**abrirPortaAutomatica();**

**fecharPortaAutomatica();**

**}**

**MUDANÇAS:**

Pinos

Tensão dos pinos

Velocidade da serial

PWM e Servo

Sem mudança na lógica do código