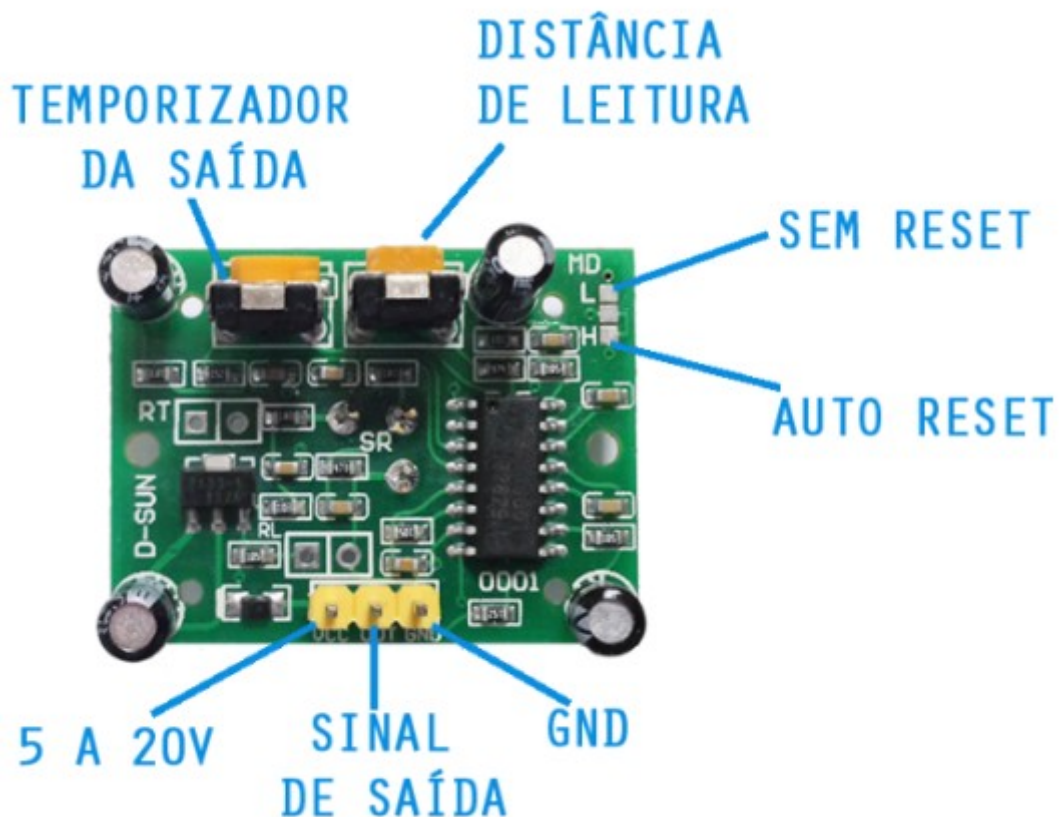


**MICROCONTROLADORES ESP8266 e ESP32  
(TEORIA E PRÁTICA)**

Edição  
Julho/2025



## 8. Sistema detector de presença com Infra Vermelho



Entendendo o sensor **PIR HC-SR501**

O **sensor PIR (Passive InfraRed)** detecta variações de calor (movimento de pessoas/animais). Ele possui **3 pinos**:

1. **VCC** → alimentação (3.3V ou 5V)
2. **OUT** → saída digital (HIGH = presença detectada, LOW = ausência)
3. **GND** → terra

➔ Ele também possui **2 trimpots** (potenciômetros):

- **Sx (Sensibilidade)**: ajusta a distância de detecção (aprox. de 3m a 7m).
- **Tx (Tempo)**: ajusta o tempo que a saída ficará em nível HIGH após detectar movimento (de ~2s a 200s).

- **Jumper de modo** (às vezes presente): seleciona se o sensor reinicia a contagem do tempo ao detectar novo movimento (*retriggerable*) ou se apenas espera o tempo acabar (*non-retriggerable*).

### Ligação do PIR ao NodeMCU ESP8266

O NodeMCU funciona a 3.3V nos pinos GPIO, e o PIR suporta 3.3V ou 5V.

#### Conexões:

- PIR **VCC** → 3V3 (3.3V) do NodeMCU
- PIR **GND** → GND do NodeMCU
- PIR **OUT** → D5 (GPIO14) do NodeMCU

### Ligação do módulo relé ao NodeMCU

O módulo relé também tem **3 pinos de controle**:


1. **VCC** → alimentação (use 5V para garantir acionamento do relé)
2. **IN** → pino de controle (baixo nível ou alto nível, depende do módulo)
3. **GND** → terra

#### Conexões:

- Relé **VCC** → VIN ou VU do NodeMCU (quando o NodeMCU está sendo alimentado por 5V via USB ou fonte)
- Relé **GND** → GND do NodeMCU
- Relé **IN** → D6 (GPIO12) do NodeMCU

**Observação:** A entrada de controle do relé é compatível com 3.3V do ESP8266, então pode ser ligada diretamente.

---



### Diagrama resumido de ligações

HC - SR501		NodeMCU
VCC	-----	3V3
OUT	-----	D5 (GPIO14)
GND	-----	GND

### Relé

VCC	-----	VIN (5V)
IN	-----	D6 (GPIO12)
GND	-----	GND

➔ O **dispositivo a ser controlado** (lâmpada, ventilador etc.) deve ser ligado aos contatos **COM** e **NO/NC** do relé.



### **Código para Arduino IDE**

```
#define PIR_PIN D5    // Pino do sensor PIR

#define RELE_PIN D6   // Pino do relé


void setup() {

    pinMode(PIR_PIN, INPUT);

    pinMode(RELE_PIN, OUTPUT);

    digitalWrite(RELE_PIN, LOW); // Relé desligado inicialmente
    Serial.begin(115200);
}

void loop() {

    int estadoPIR = digitalRead(PIR_PIN);

    if (estadoPIR == HIGH) {

        Serial.println("Movimento detectado!");

        digitalWrite(RELE_PIN, HIGH); // Liga o relé

    } else {

        Serial.println("Sem movimento");

        digitalWrite(RELE_PIN, LOW); // Desliga o relé

    }

    delay(200); // pequena pausa para estabilidade
}
```



## Ajustando os trimpots do PIR

- **Trimpot da sensibilidade (S):**  
Gire no sentido horário → maior alcance (até 7m).  
Gire no sentido anti-horário → menor alcance (~3m).
- **Trimpot do tempo (T):**  
Gire no sentido horário → maior tempo em nível HIGH após detecção (até ~200s).  
Gire no sentido anti-horário → menor tempo (2s).

## Como funciona

1. O sensor PIR monitora o ambiente.
2. Quando detecta movimento, ele envia **HIGH** (3.3V) no pino OUT.
3. O ESP8266 lê o sinal e ativa o **relé**.
4. Passado o tempo definido no trimpot, se não houver movimento, o PIR volta a LOW e o relé desliga.

## Bibliografia:

[https://www.espressif.com/sites/default/files/documentation/esp32\\_datasheet\\_en.pdf](https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf)  
<https://docs.espressif.com/projects/esp-idf/en/stable/esp32/index.html>  
<https://randomnerdtutorials.com/>  
<http://www.practical-arduino.com/>