

Programando com GPIO no Python

Transcrição

Aprendemos a interagir com o GPIO via console, mas também conseguimos interagir via Python!

Para isso, podemos utilizar a interface gráfica do Raspberry PI. Para uso remoto, precisamos do **VNC**, conforme vimos no primeiro curso. Podemos abrir o Python clicando em **Menu -> Programming -> Python 2 (IDLE)**.

Começamos criando um novo arquivo, clicando em **File -> New File**. Agora podemos escrever o nosso código. Primeiramente, para podermos nos comunicar com o GPIO, precisamos importar a sua biblioteca:

```
import RPi.GPIO as GPIO
```

Vamos importar também a biblioteca de temporização **time**:

```
import RPi.GPIO as GPIO
import time
```

Neste curso estamos usando como referência a numeração da pinagem do GPIO (e não o número do GPIO em si), mas precisamos "dizer" isso ao GPIO:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
```

Se quisermos utilizar o número do GPIO como referência, basta trocar **BOARD** por **BCM**.

Outro detalhe que merece ser falado, é que há mensagens que o GPIO fica jogando na tela, por exemplo se a porta já estiver alocada. Para não exibi-las, adicionamos a seguinte linha:

```
GPIO.setwarnings(False)
```

O próximo passo é definir o pino 7 como saída (**OUT**). Então fazemos:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

GPIO.setup(7, GPIO.OUT)
```

Para definir como entrada, basta trocar **OUT** por **IN**. Após isso, precisamos acender o LED, mas antes vamos exibir uma mensagem, utilizando o `print` :

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

GPIO.setup(7,GPIO.OUT)

print('LED ON')
```

E colocamos o pino 7 em **alta (HIGH)**:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

GPIO.setup(7,GPIO.OUT)

print('LED ON')
GPIO.output(7,GPIO.HIGH)
```

Isso já faz com que o LED fique aceso. Vamos deixá-lo aceso por 2 segundos e depois apagá-lo. Para apagá-lo, basta colocar o pino 7 em baixa (**LOW**) e para aguardar os 2 segundos, utilizamos o `time.sleep(2)`

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

GPIO.setup(7,GPIO.OUT)

print('LED ON')
GPIO.output(7,GPIO.HIGH)

time.sleep(2)

print('LED OFF')
GPIO.output(7,GPIO.LOW)
```

Com isso tudo feito, podemos testar o nosso código. Mas antes vamos criar um diretório, com o nome de **GPIO** e salvar o nosso arquivo Python dentro deste diretório, com o nome `teste.py` .

Agora podemos executar o arquivo, clicando em **Run -> Run Module**. Repare no LED ligando e desligando, e nas respectivas mensagens no console.

Podemos ainda deixar o LED piscando, envolvendo o código em um loop `while` e adicionando mais um `time.sleep(2)` após desligamento do LED:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

GPIO.setup(7,GPIO.OUT)

while(1):
    print('LED ON')
    GPIO.output(7,GPIO.HIGH)

    time.sleep(2)

    print('LED OFF')
    GPIO.output(7,GPIO.LOW)
    time.sleep(2)
```

Salvamos e executamos novamente o arquivo, e vemos o LED piscando infinitamente.

Uma vez entendida essa codificação e conseguindo controlar o LED, acabaram os nossos problemas, qualquer dispositivo é passível do nosso controle, basta ele estar dimensionado e alimentado, mas a lógica que usaremos é bem parecida com a feita neste capítulo. Nos próximos capítulos vamos começar a construir a automação, só que ao invés de utilizar o LED, utilizaremos o **Módulo Relé**.