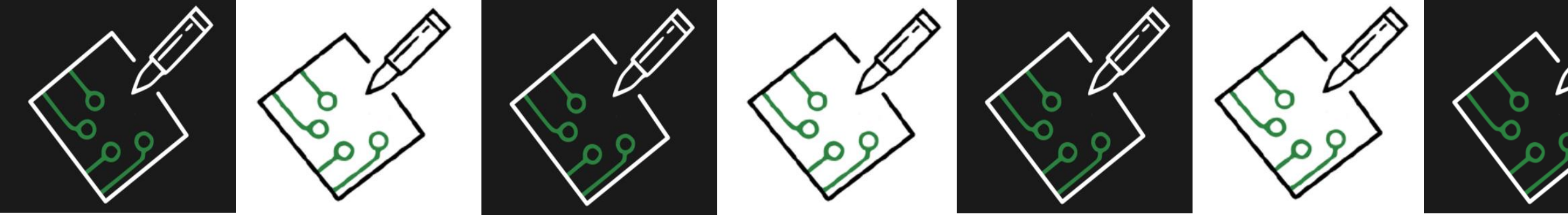
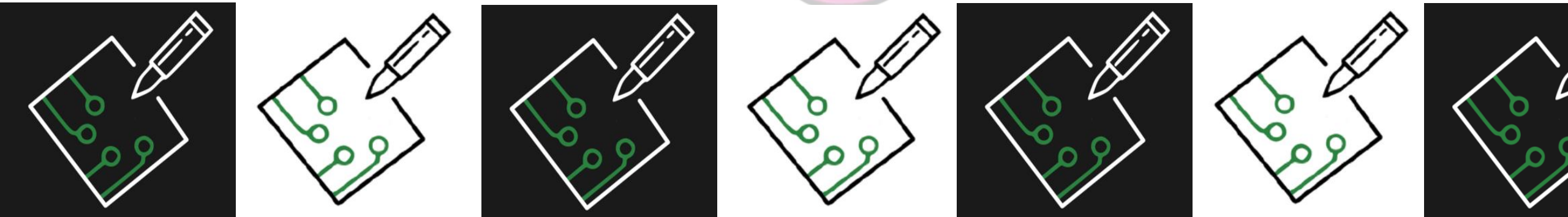


Franzinando Na FATEC



Aula 02:

Sinal Digital

Introdução ao Python

- Python é uma linguagem bastante poderosa e apresenta uma estrutura sintática muito simples em comparação as principais linguagens do mercado.
- É uma linguagem interpretada e fracamente tipada (não precisamos declarar o tipo de uma variável, por exemplo).

Estrutura Condicional (if-else)

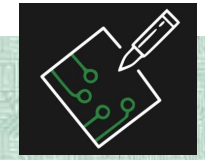
- A estrutura condicional é uma seção que ajuda a definir condições para a execução de determinados blocos de comandos.
- Em vez de executar tudo de vez, sem nenhuma interrupção, o programa deve parar para executar um teste e decidir qual caminho seguir.

Estrutura Condicional (if-else)

```
name = 'Jason'
if name == 'Jason':
    print("Hello Jason, Welcome")
else:
    print("Sorry, I don't know you")
```

Operadores Lógicos

- Operadores nos possibilitam construir um tipo de teste muito útil e muito utilizado em qualquer programa: os testes lógicos.
- Os três tipos de Operadores Lógicos são: and, or e not.

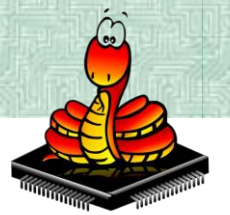


Operadores Lógicos

- **AND**: Retorna True se ambas as afirmações forem verdadeiras
- **OR**: Retorna True se uma das afirmações for verdadeira
- **NOT**: Retorna Falso se o resultado for verdadeiro

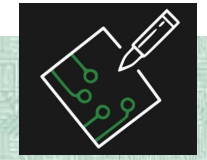
Exemplos Operadores Lógicos

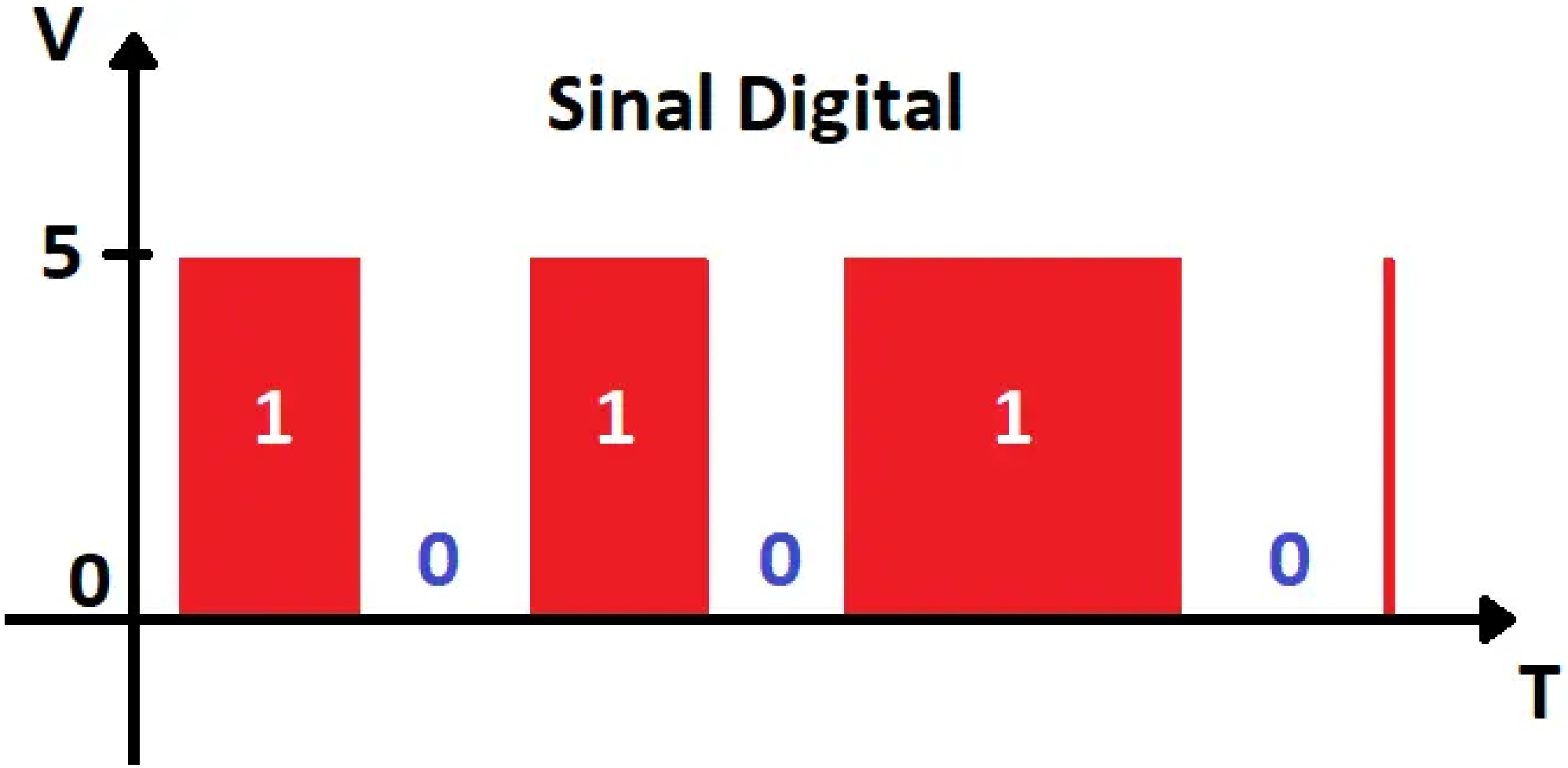
```python
#exemplo da utilização de cada um:
num1 = 7
num2 = 4

# Exemplo and
if num1 > 3 and num2 < 8:
    print("As Duas condições são verdadeiras")

# Exemplo or
if num1 > 4 or num2 <= 8:
    print("Uma ou duas das condições são verdadeiras")

# Exemplo not
if not (num1 < 30 and num2 < 8):
    print('Inverte o resultado da condição entre os parânteses')
```

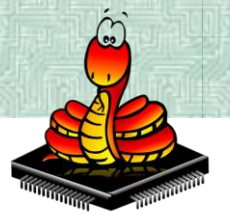


Sinal Digital

- Sinal Digital possui uma quantidade limitada, geralmente é representado por dois números, sendo eles 0 e 1. Assim o Sinal digital é definida para instantes de tempo e o conjunto de valores quem podem assumir é finita.

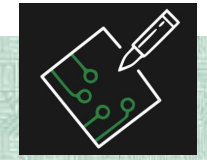



Sinal Digital
V
5
0
1
0
1
0
1
0
T

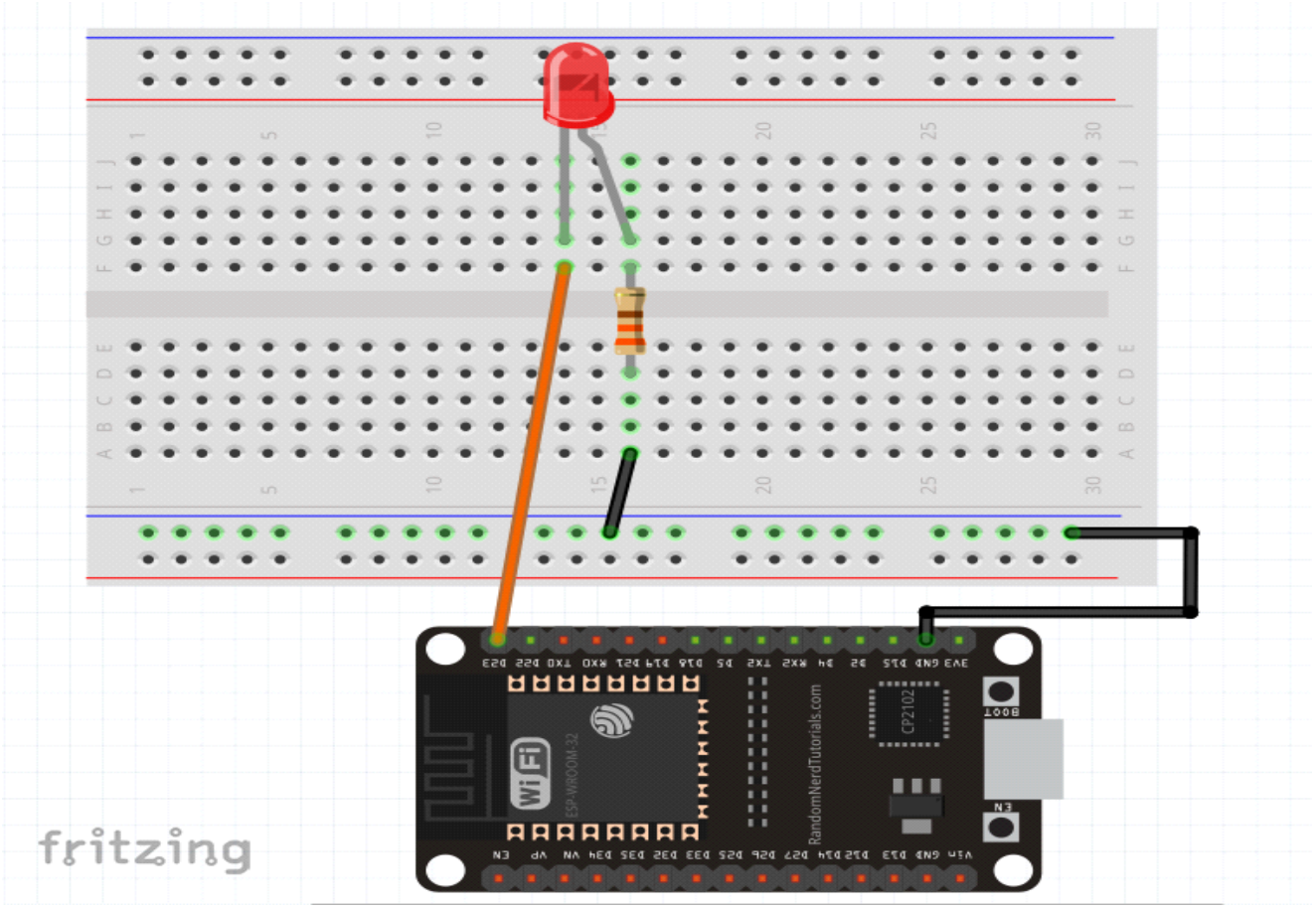


Hello World

- Agora vamos fazer o nosso ‘Hello World’ das plaquinhas microcontroladas, mais conhecido como Blink (pisca-pisca), onde vamos visualizar o conceito de uma saída digital.



Primeira Montagem



Código - Saída digital

- Na primeira linha, vamos importar a função Pin do módulo machine, que nos possibilita ter acesso aos pinos da placa:

```
1 from machine import Pin #acessar os pinos da placa
```

Código - Saída digital

- Agora, vamos importar a função sleep da biblioteca time, utilizada para intervalos de temporização.

```
2 from time import sleep#intervalos de temporização
3 #import time
```

Código - Saída digital

- Agora, vamos importar a função sleep da biblioteca time, utilizada para intervalos de temporização.

```
2  from time import sleep#intervalos de temporização
3  #import time
```

Código - Saída digital

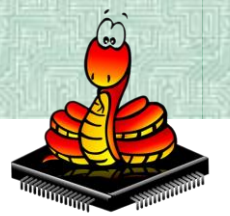
- Vamos configurar o nosso pino como uma saída digital. Para isso, vamos chamar a função Pin, entre parênteses, primeiro vamos identificar qual o pino estamos utilizando e depois dizer se uma saída ou uma entrada digital (OUT / IN). Todas essas informações devem estar atribuídas ao objeto led

```
5 led = Pin(23, Pin.OUT) #Configurar o pino 2 como uma saída digital
```

Código - Saída digital

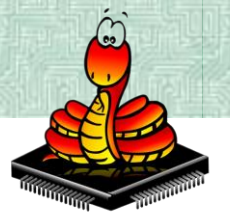
- Por fim, criarmos o laço de repetição infinita, mais conhecido como loop e para esse objetivo utilizamos o while True. Dentro do laço, começaremos ligando o nosse led (colocando ele no estado 1) para isso vamos chamar a função led.value( )

```
 7  while True:
 8      led.value(1) #liga o led
 9      sleep(1) #aguarda 1 segundo
10      led.value(0) #desliga o led
11      sleep(1)
```



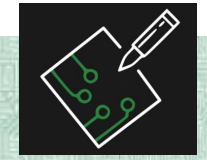
Código - Saída digital

```
1  from machine import Pin #acessar os pinos da placa
2  from time import sleep#intervalos de temporização
3  #import time
4
5  led = Pin(23, Pin.OUT) #Configurar o pino 2 como uma saída digital
6
7  while True:
8      led.value(1) #liga o led
9      sleep(1) #aguarda 1 segundo
10     led.value(0) #desliga o led
11     sleep(1)
```

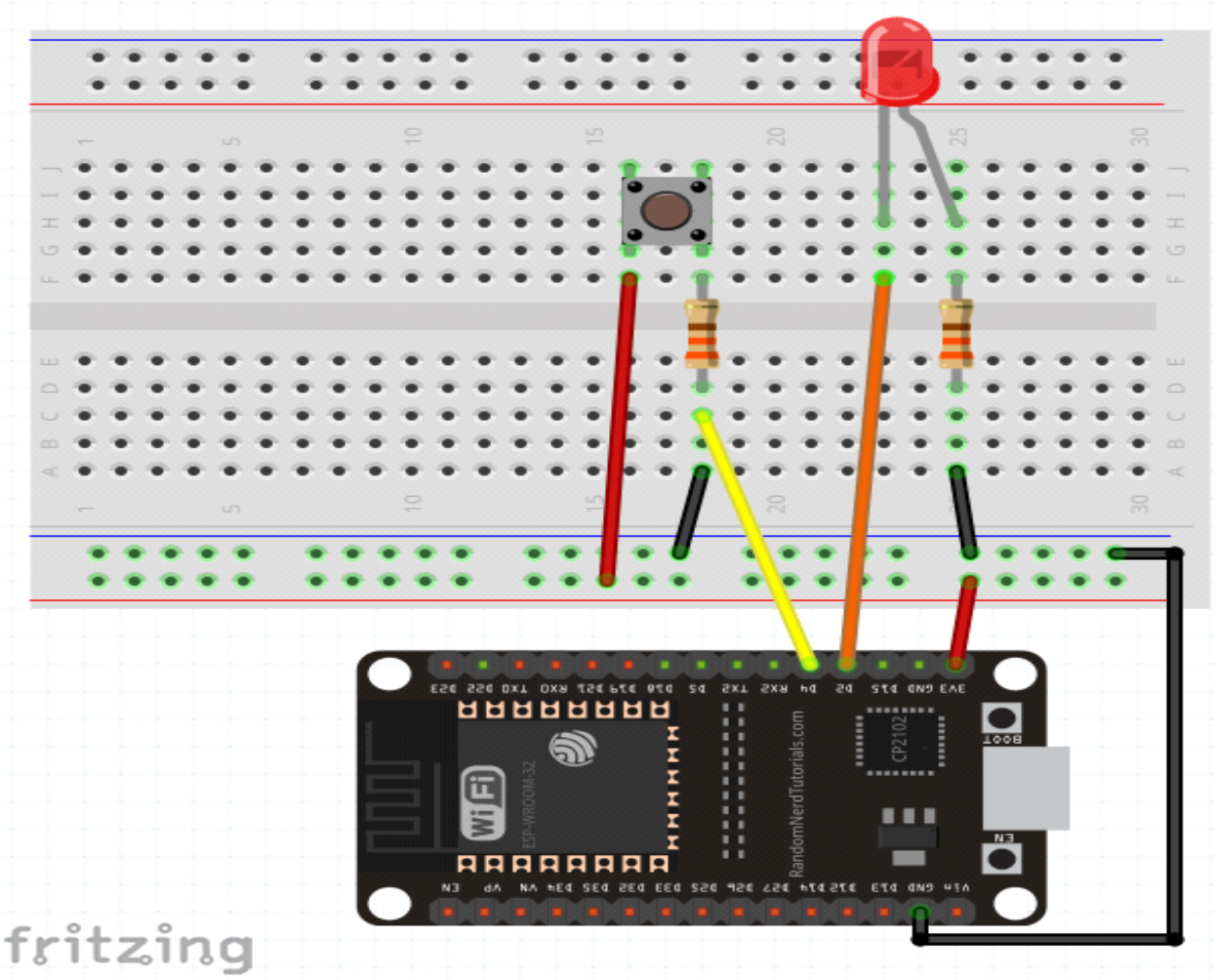


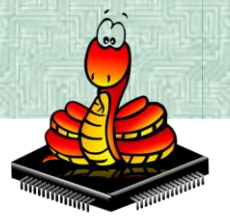
Entrada digital

- No segundo código, aprenderemos a configurar o pino como uma entrada digital, que é muito similar ao que já vimos, apenas com algumas mudanças.



Montagem





Código 2 – Entrada Digital

- Aqui, importaremos os mesmos módulos usados no exemplo anterior e configuraremos um pino como uma saída digital:

```
4  from machine import Pin              #acessar os pinos da placa
5  from time import sleep  #intervalos de temporização
6
7  #Configurando o pino como saída digital
8  led = Pin(2, Pin.OUT)    #Associar o pino 2 da placa a variável led
9                           #e identificá-lo como um pino digital
```

Código 2 – Entrada Digital

- Agora, vamos configurar o pino como uma entrada digital. Repetiremos o mesmo procedimento seguido para uma saída digital, a única diferença é que usaremos o IN invés de OUT:

```
12 botao = Pin(4, Pin.IN) #mapea para pino 3 e configura como entrada
```

- Em sequência, vamos criar uma variável chamada estado que será iniciada com o valor 0:

```
14 estado = 0
```

Código 2 – Entrada Digital

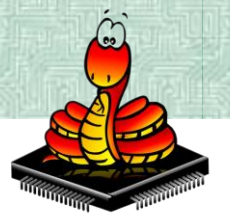
- Agora, , criaremos o nosso loop. Dentro do laço, utilizaremos a condição if, onde caso o botão seja pressionado, o valor armazenado na variável estado é trocado e, consequentemente, o estado do led também é trocado:

```
16 while True: #laço de repetição infinito (loop)
17
18     if botao.value():          #Se houver alteração no valor do botão,
19                                #ou seja, se o botão for pressionado
20         estado = not estado    #troca o estado
21
22     led.value(estado)         #led se comporta de acordo com a variável estado
23     print(estado)
24     sleep(1)
```

Código 2 – Entrada Digital

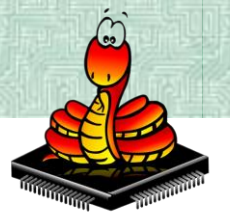
- Agora, , criaremos o nosso loop. Dentro do laço, utilizaremos a condição if, onde caso o botão seja pressionado, o valor armazenado na variável estado é trocado e, consequentemente, o estado do led também é trocado:

```
16 while True: #laço de repetição infinito (loop)
17
18     if botao.value():          #Se houver alteração no valor do botão,
19                                #ou seja, se o botão for pressionado
20         estado = not estado    #troca o estado
21
22     led.value(estado)         #led se comporta de acordo com a variável estado
23     print(estado)
24     sleep(1)
```



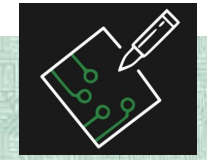
Código 2 – Entrada Digital

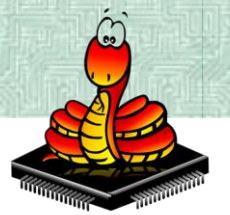
```
#Aula 2: Sinal Digital

#Módulos Necessários
from machine import Pin           #acessar os pinos da placa
from time import sleep  #intervalos de temporização

#Configurando o pino como saída digital
led = Pin(2, Pin.OUT)    #Associar o pino 2 da placa a variável led
                         #e identificá-lo como um pino digital

#Configurando o pino como entrada digital
botao = Pin(4, Pin.IN) #mapea para pino 3 e configura como entrada

estado = 0

while True: #laço de repetição infinito (loop)

    if botao.value():          #Se houver alteração no valor do botão,
                               #ou seja, se o botão for pressionado
        estado = not estado    #troca o estado

    led.value(estado)         #led se comporta de acordo com a variável estado
    print(estado)
    sleep(1)
```



Exercício para fazer em sala

- Ao pressionar um botão, faça um led piscar. Para esse exercício, você vai utilizar a montagem do exemplo 2.

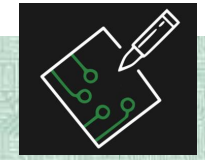


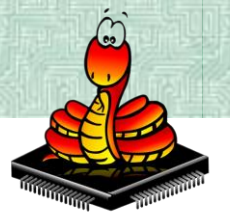


Exercício 1

Faça um semáforo de trânsito usando três LEDs (vermelho, verde e amarelo), cada um em um pino digital do Arduino. Faça-os acender conforme a sequência:

- LED verde aceso por 55, enquanto outros LEDs estão apagados.
- LED verde apaga, LED amarelo acende por 2s, LED vermelho continua apagado.
- LED amarelo apaga, LED vermelho acende por 5s, LED verde continua apagado.
- LED vermelho apaga, LED verde acende por 5s, LED amarelo continua apagado.
- [repetir a sequência]

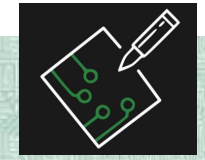


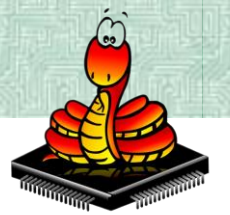


Exercício 2

Faça um semáforo de trânsito para carro e pedestres usando cinco LEDs (2x vermelho, 2x verde e 1x amarelo), cada um em um pino digital do Arduino. Faça-os acender conforme a sequência:

- LED verde carro e LED vermelho pedestre iniciam acesos. Demais LEDs
- apagados.
- Se botão de pedestre for pressionado, então:
- LED verde carro apaga; LED vermelho de pedestre se mantém aceso.
- LED amarelo pisca 2x em intervalo de 0,5s.
- LED amarelo e LED vermelho pedestre apagam: LED verde pedestre e
- LED vermelho carro acendem por 5s.
- LED verde pedestre e LED vermelho carro apagam.
- LED vermelho pedestre e LED verde carro acendem.
- Se o botão não for pressionado, não acontece nada.
- Fim do código.





Referências

SOUZA, Fábio. **Entradas e Saídas Digitais.** Disponível em: https://docs.franzininho.com.br/docs/franzininho-wifi/exemplos-circuitpython/entradas-saidas-digitais