



Minicurso de Arduino

Aula 03



UNIVERSIDADE
FEDERAL DO CEARÁ
CAMPUS QUIXADÁ



Revisão de FUP

- Constantes:
 - **const int x = 10;**
 - **#define x 100**
 - **True/False.**
 - **HIGH/LOW.**
 - **INPUT/OUTPUT**
- Comentário:
 - **// Este é um comentário de linha**
 - **/*Este é um comentário permite mais de uma linha */**



Revisão de FUP

- Tipo de Variáveis:
 - As variáveis são lugares na memória principal que servem para armazenar dados.
 - São acessadas por meio de um identificador único.
 - O seu valor pode ser alterado ao longo da execução do programa.
 - A variável só pode armazenar um valor a cada instante.
 - Obedecendo a regra: o primeiro caractere do nome de uma variável deve, obrigatoriamente, ser uma letra e não pode ter caracteres especiais ou palavras reservadas.

Tipo de dados	RAM	Intervalo numérico
void keyword	N/A	N/A
boolean	1 byte	0 a 1 (false ou true)
byte	1 byte	0 a 255
char	1 byte	-128 a 127
unsigned char	1 byte	0 a 255
int	2 bytes	-32.768 a 32.767
unsigned int	2 bytes	0 a 65.535
word	2 bytes	0 a 65.535
long	4 bytes	-2.147.483.648 a 2.147.483.647
unsigned long	4 bytes	0 a 4.294.967.295
float	4 bytes	-3,4028235E+38 a 3,4028235E+38
double	4 bytes	-3,4028235E+38 a 3,4028235E+38
string	1 byte + x	Sequência de caracteres
array	1 byte + x	Coleção de variáveis



Revisão de FUP

- Tipo de operadores:
 - Aritméticos:

Símbolo	Significados
-	Subtração
+	Adição
*	Multiplicação
/	Divisão
%	Resto da divisão (módulo)

- Relacionais:

Operação	Operador	Exemplo
Igualdade	==	x == y
Diferença	!=	x != y
Maior	>	x > y
Menor	<	x < y
Maior ou igual	>=	x >= y
Menor ou igual	<=	x <= y



Revisão de FUP

- Tipo de operadores:
 - Lógicos:
 - Compostos:

Operação	Operador
E	&
Ou	
Negação	!
Ou Exclusivo	^

- ++ (incremento)
- -- (decremento)
- += (adição com atribuição)
- -= (subtração com atribuição)
- *= (multiplicação com atribuição)
- /= (divisão com atribuição)



Entradas e Saídas



Sinais Digitais e Analógicos

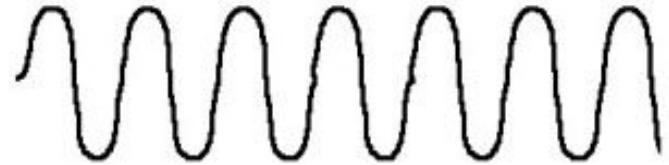
Sinal Digital: Possui uma quantidade limitada, geralmente representada por dois níveis.

Sinal Analógico: Pode assumir infinitos valores em um intervalo de tempo

Sinal digital



Sinal analógico



Pinos Digitais



O Arduino possui 14 pinos que podem ser utilizadas como **entrada** ou **saída** digital, numeradas de 0 a 13. Antes de utilizá-las é necessário configurá-la conforme a necessidade.

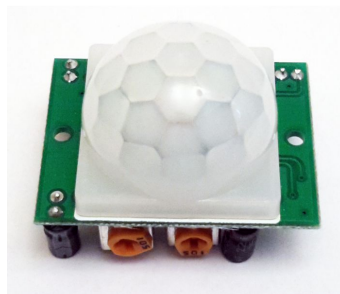
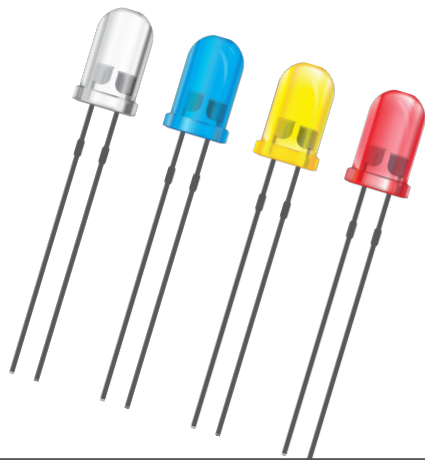
Estão limitadas a dois estados, ou seja, o Arduino reconhece apenas dois valores de tensão: 0V e 5V.

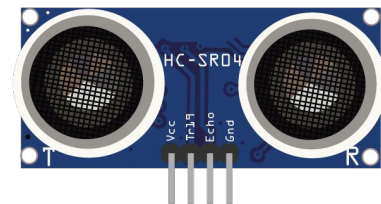
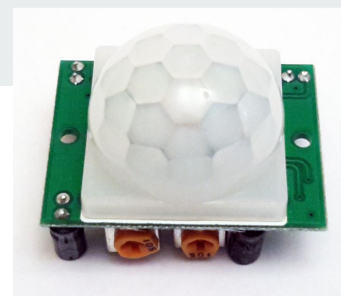
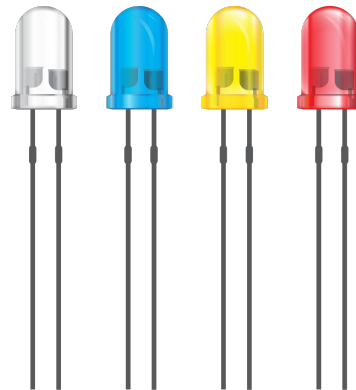
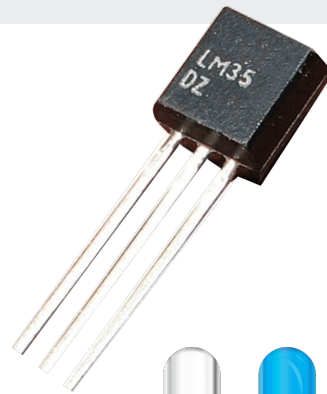
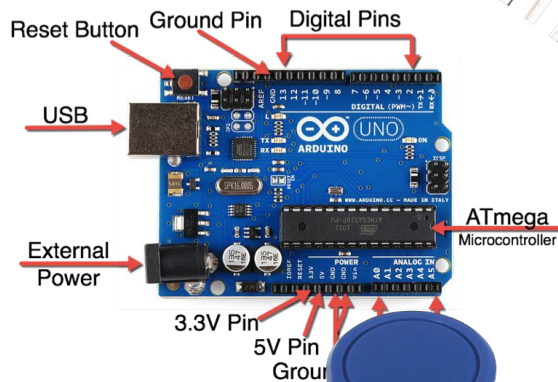
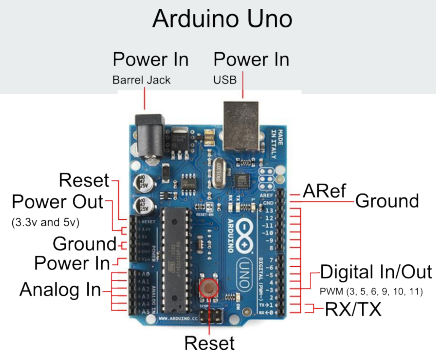
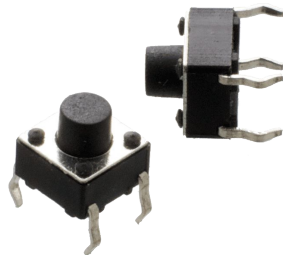
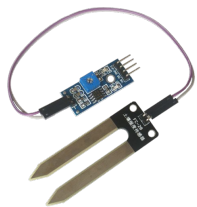
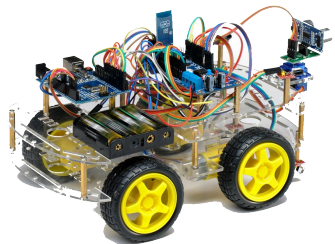
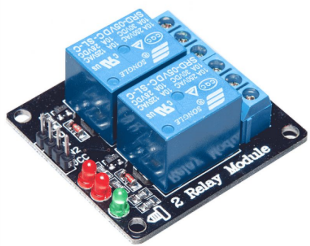


UNIVERSIDADE
FEDERAL DO CEARÁ
CAMPUS QUIXADÁ



Dispositivos com sinal digital





Funções para usar os pinos digitais

- **void pinMode()**
 - Sintaxe: pinMode(**pino**, **modo**);
 - **Modo:**
 - **INPUT**: Entrada digital;
 - **INPUT_PULLUP**: Entrada digital como resistor interno PULL-UP habilitado;
 - **OUTPUT**: Saída digital;
- **int digitalRead();**
 - Sintaxe: digitalRead(**pino**);
 - Retorno: HIGH ou LOW;
- **void digitalWrite()**
 - Sintaxe: digitalWrite(**pino**, **valor**);
 - **Valor:**
 - **HIGH**(nível lógico alto)
 - **LOW**(nível lógico baixo)



Resistores Pull-Up e Pull-Down

Pull-Up:

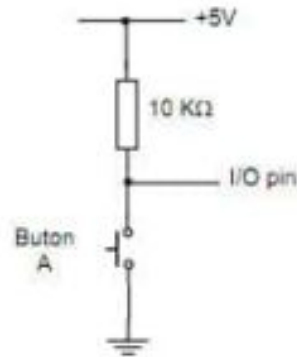
Chave aberta: 5V

Chave fechada: 0V

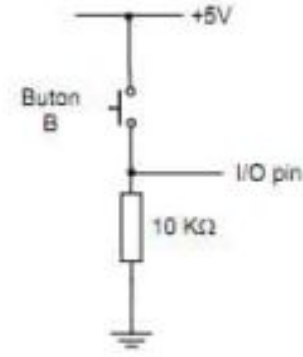
Pull-Down:

Chave aberta: 0V

Chave fechada: 5V



Pull-Up



Pull-Down



Pinos Analógicos



O Arduino possui 6 entradas analógicas, numeradas de A0 a A5. São utilizadas para medir variações de tensão.

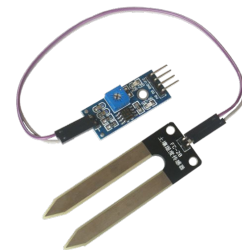
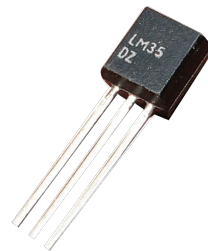
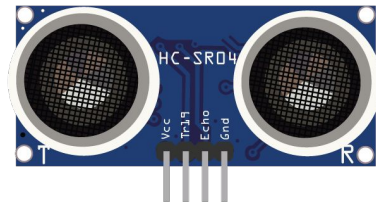
Para a leitura dos sinais analógicos, o Arduino traduz os valores para um valor digital, visto que internamente ele trabalha com dados digitais. Esse processo é feito pelo conversor analógico digital, ADC ou A/D.



UNIVERSIDADE
FEDERAL DO CEARÁ
CAMPUS QUIXADA



Dispositivos com sinal analógico

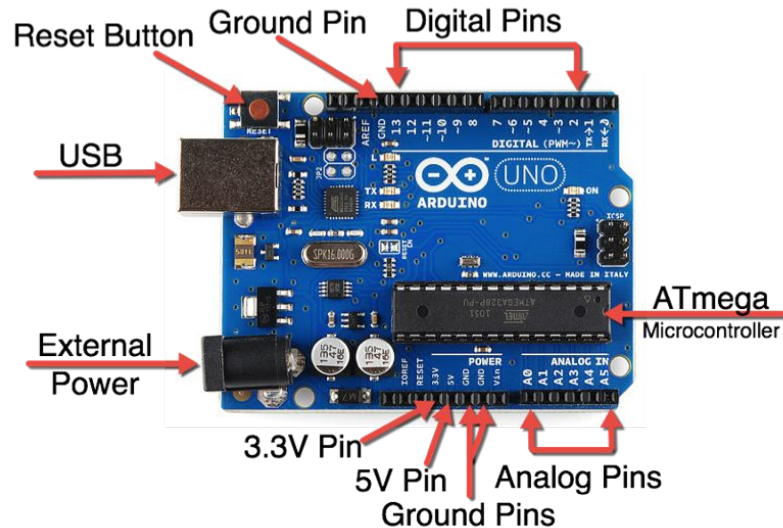


Funções para usar os pinos analógicos

- **int analogRead();**
 - Sintaxe:
analogRead(**pino**);
 - Retorno: int(0 a 1023);



Pinos Arduino - Esquemático



Ambiente - Arduino IDE



O ambiente de desenvolvimento do Arduino é gratuito e pode ser baixado no site.

As principais funcionalidades são:

- Escrever código do programa.
- Salvar o código.
- Compilar o código.
- Transportar código o compilado para a placa do Arduino.



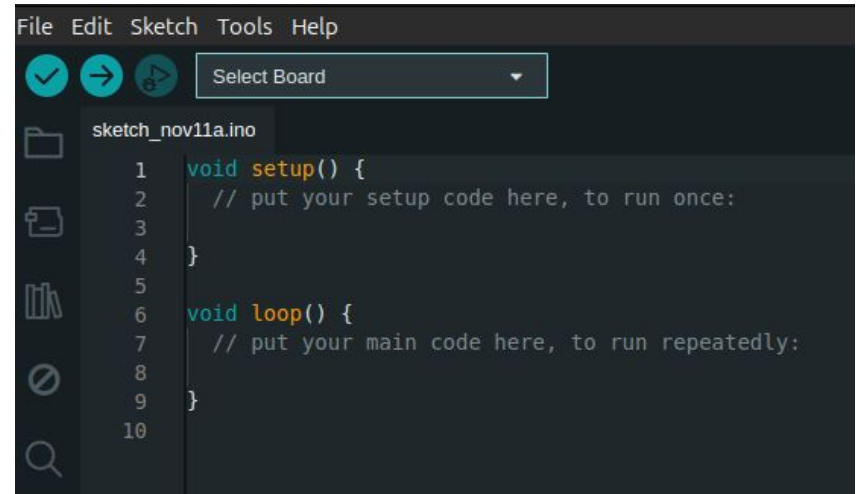
UNIVERSIDADE
FEDERAL DO CEARÁ
CAMPUS QUIXADÁ



Funções `setup()` e `loop()`

`setup()`: onde devem ser definidas algumas configurações iniciais do programa. Ele executa uma única vez.

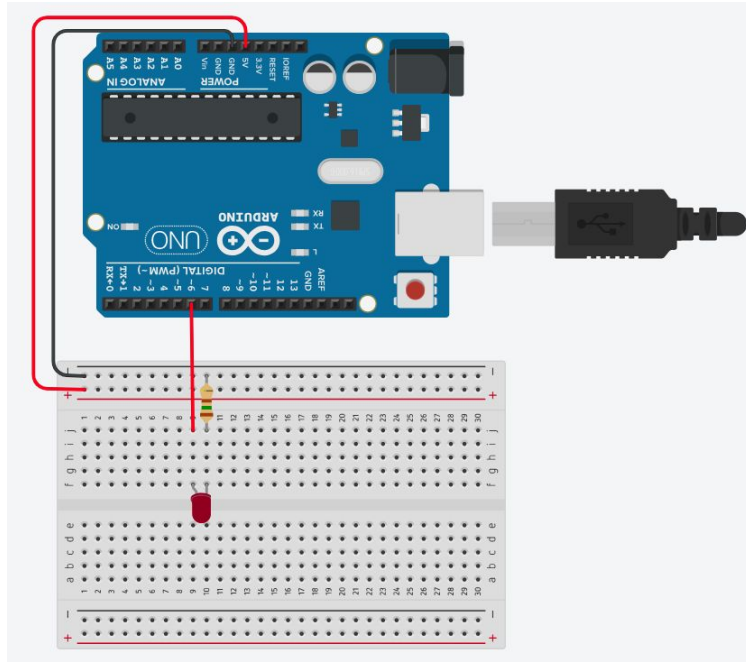
`loop()`: função principal do programa. Fica executando indefinidamente.



```
File Edit Sketch Tools Help
[Checkmark] [Next] [Upload] Select Board
sketch_nov11a.ino
1 void setup() {
2   // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7   // put your main code here, to run repeatedly:
8
9 }
10
```



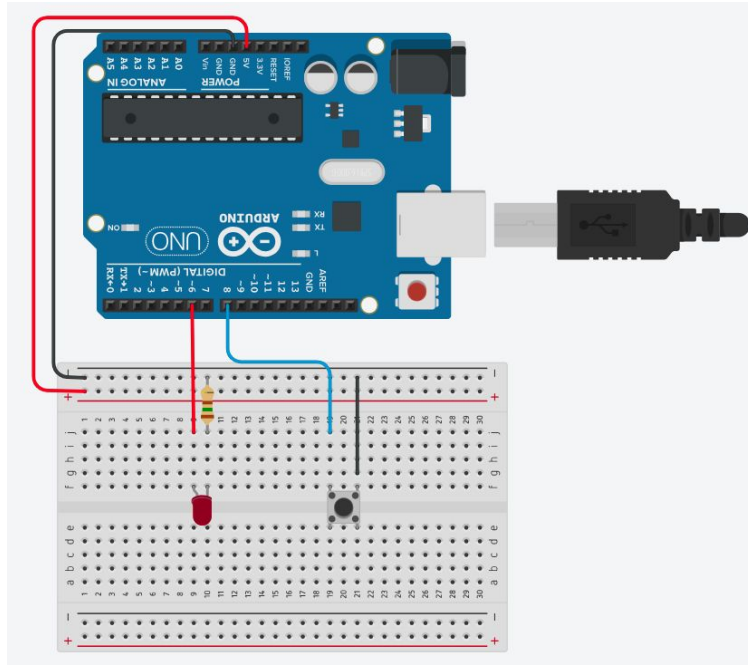
Prática no Tinkercad



```
1  #define led 6;
2
3  void setup()
4  {
5      pinMode(led, OUTPUT);
6  }
7
8  void loop()
9  {
10     digitalWrite(led, LOW);
11     delay(1000);
12     digitalWrite(led, HIGH);
13     delay(1000);
14 }
15
16
17
```



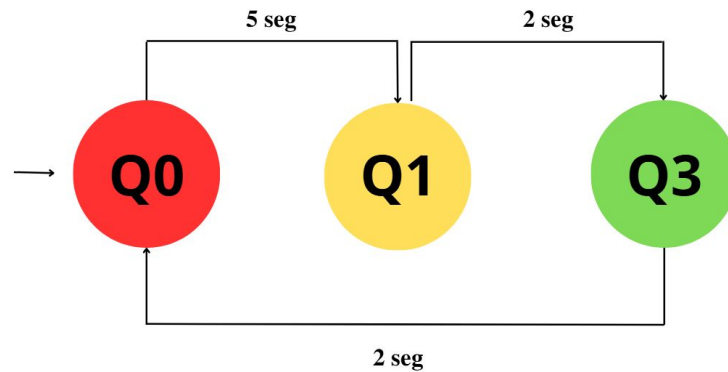
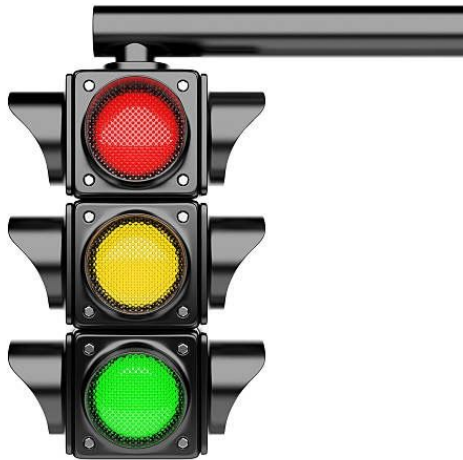
Prática no Tinkercad



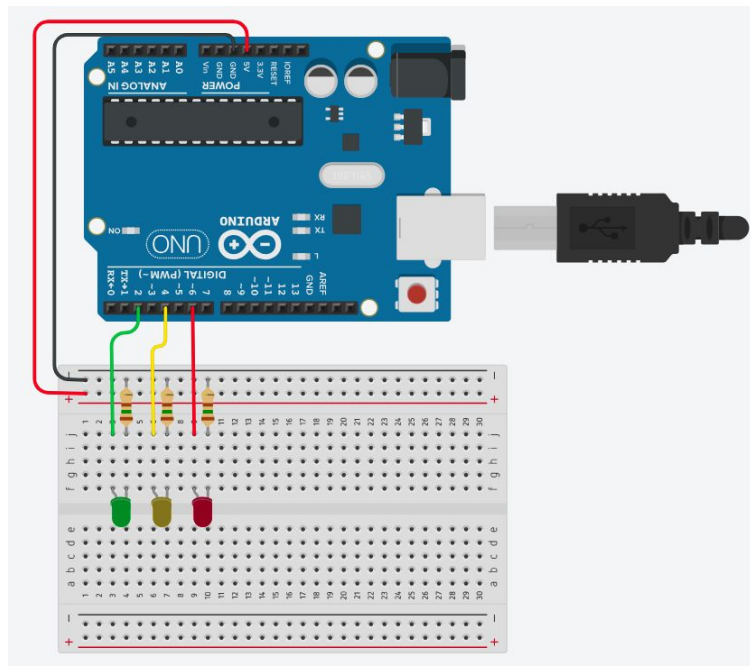
```
1 #define led 6
2 #define botao 8
3
4 void setup()
5 {
6   pinMode(led, OUTPUT);
7   pinMode(botao, INPUT_PULLUP);
8 }
9
10 void loop()
11 {
12   if(!digitalRead(botao)){
13     digitalWrite(led, LOW);
14   } else {
15     digitalWrite(led, HIGH);
16   }
17 }
```



Semáforo



Prática no Tinkercad



```
1 //Agora é com vocês! :)
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
```

