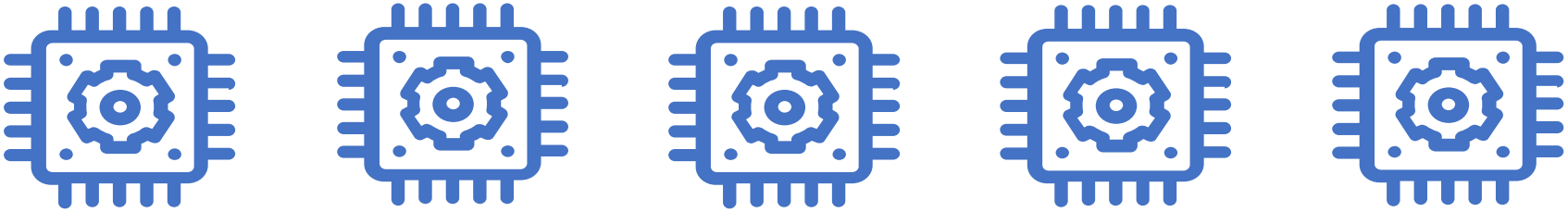
Aula 4 – Explorando o Arduino UNO

Disciplina: Microprocessadores e Microcontroladores

Professor: Daniel Gueter



CENTRO UNIVERSITÁRIO UniBTA Campus Paulista

Cronograma

08/05 – Aula 1 - Introdução da disciplina, revisão de conceitos e histórico da área
15/05 – Aula 2 - Microprocessadores: Arquitetura e instruções
22/05 – Aula 3 – Microcontroladores: Arquitetura e o Arduino
29/05 – Aula 4 – Explorando o Arduino UNO
05/06 – Aula 5
12/06 – Prova
19/06 – Feriado – Corpus Christi
26/06 – Exame

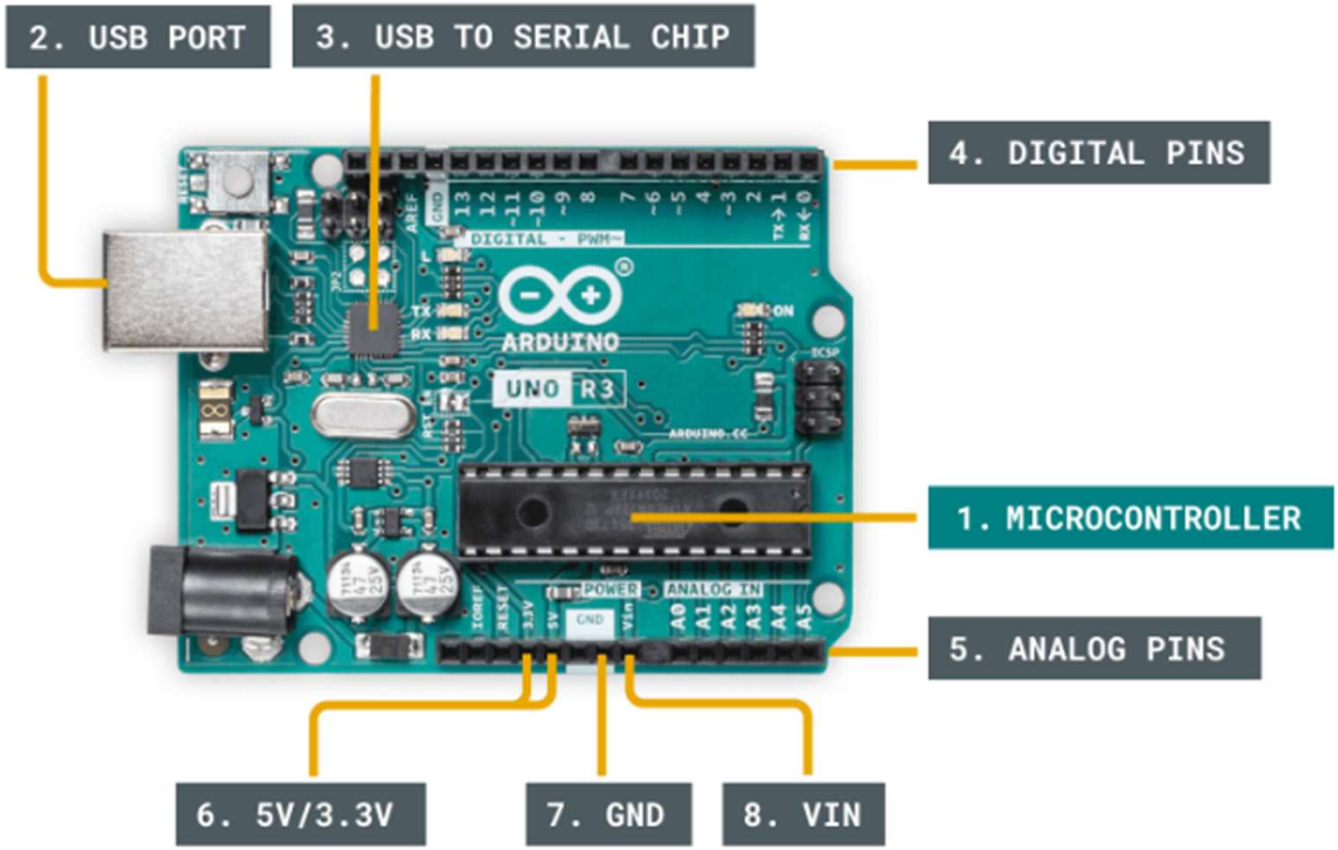
Arduino

- Arduino é uma série de placas de desenvolvimento de projetos eletrônicos baseados em microcontroladores.

- A empresa Arduino foi criada em 2005 na Itália visando a utilização em fins didáticos, e foi um sucesso sendo vendidas mais de 50 mil placas até 2008.

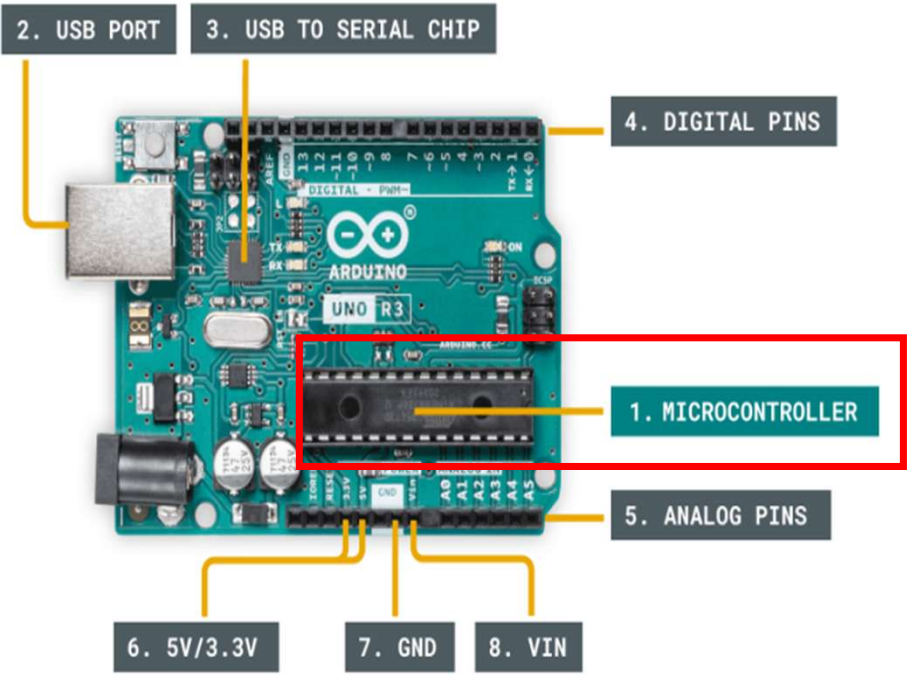
- Seu principal produto é o **Arduino UNO**

Explorando o Arduino UNO



Principais componentes de um Arduino UNO

Explorando o Arduino UNO

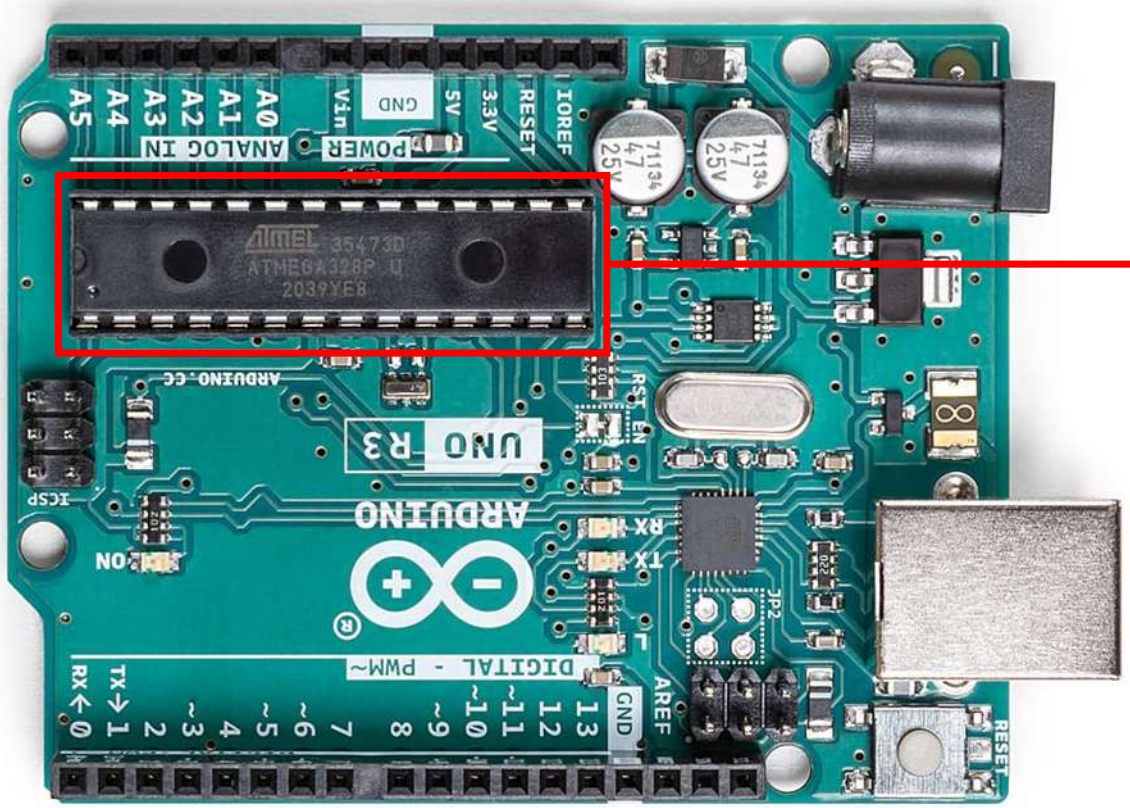


Principais componentes de um Arduino UNO

1 – Microcontrolador → ATmega328P

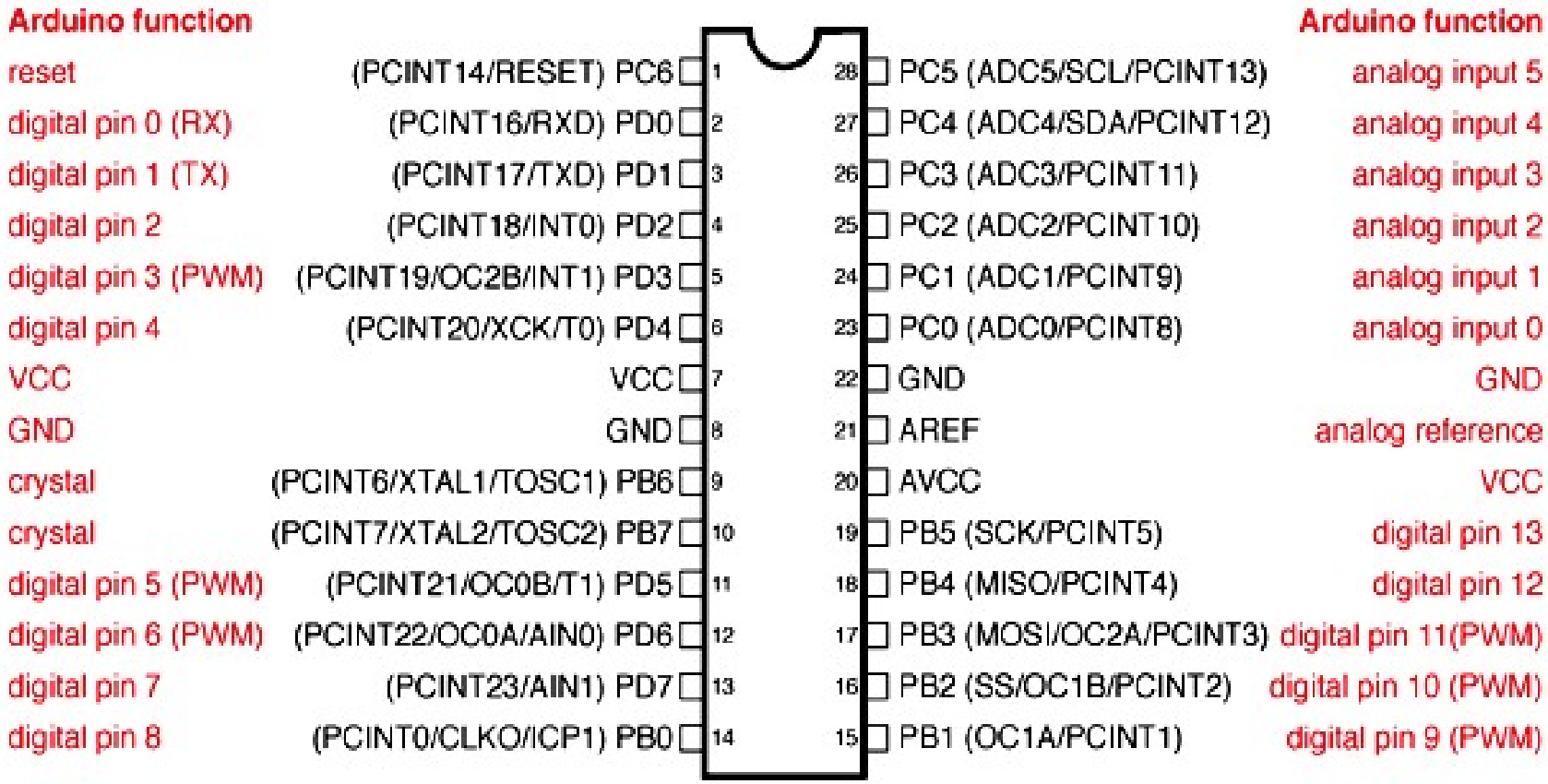
- Microcontrolador AVR RISC de chip único com arquitetura Harvard de 8 bits.
- 32K bytes de memória Flash programável
- 32 x 8 registradores
- 16MIPS a 16MHz
- 1K bytes de memória EEPROM
- 2K bytes de memória SRAM

ATmega238P → Microcontrolador do Arduino UNO



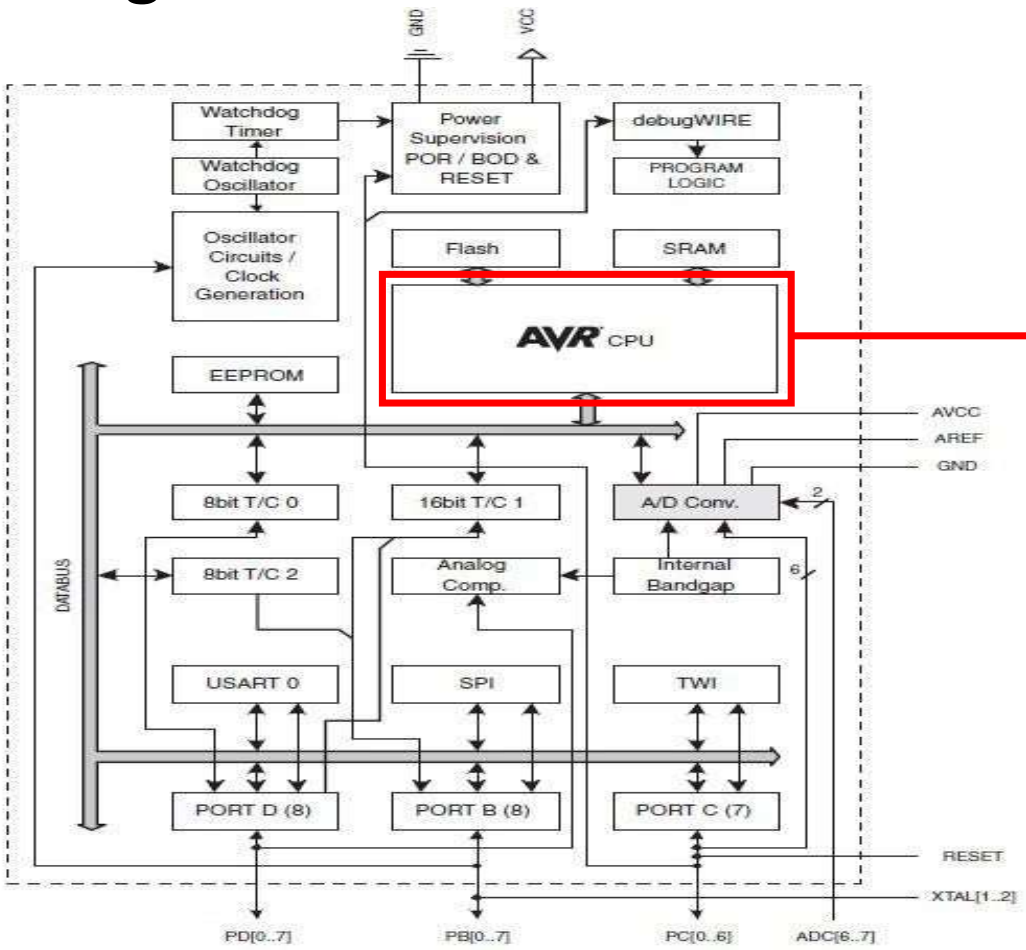
Arduino UNO

ATmega238P → Microcontrolador do Arduino UNO



Pinagem do microcontrolador ATmega238P

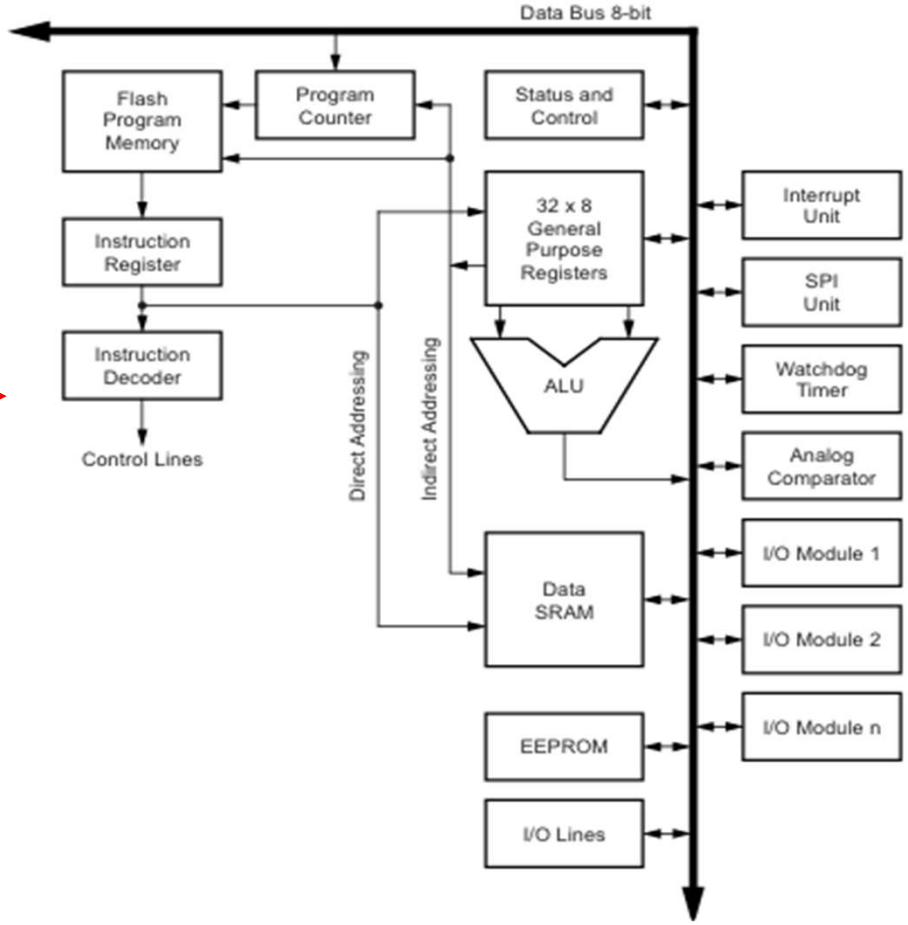
ATmega238P → Microcontrolador do Arduino UNO



Entrando no microprocessador

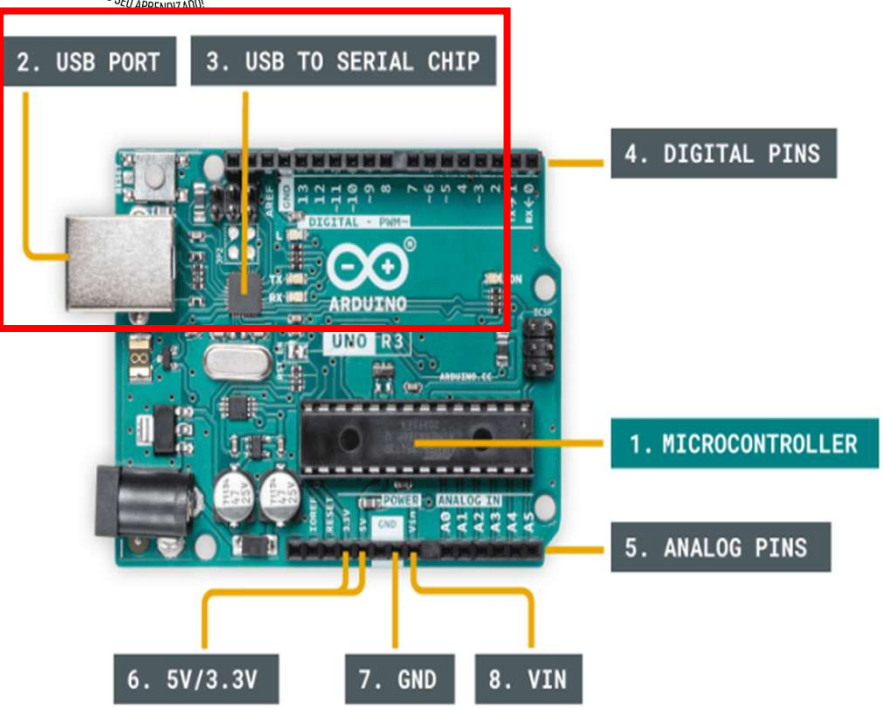
Diagrama de blocos do microcontrolador ATmega238P

ATmega238P → Microcontrolador do Arduino UNO



Arquitetura do microprocessador do ATmega238P

Explorando o Arduino UNO



Principais componentes de um Arduino UNO

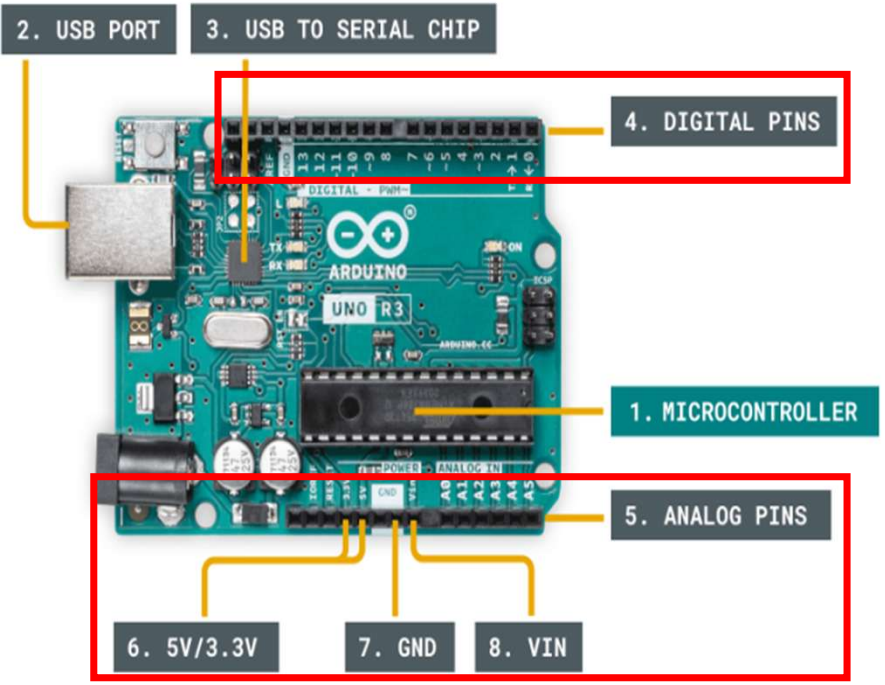
2 – Porta USB

- Usada para conectar a placa Arduino a um computador

3 – Chip USB para Serial

- O conversor USB para Serial é um componente importante, pois ajuda a traduzir os dados que vêm, por exemplo, de um computador para o microcontrolador da placa. É isso que possibilita programar a placa Arduino a partir do computador.

Explorando o Arduino UNO



Principais componentes de um Arduino UNO

4 – Pinos digitais

- Pinos que usam lógica digital (0,1 ou LOW/HIGH). Normalmente usados para interruptores e para ligar/desligar um LED.

5 – Pinos analógicos

- Pinos que podem ler valores analógicos em uma resolução de 10 bits (0-1023).

6 – Pinos 5V / 3.3V

- Pinos usados para alimentar componentes externos.

7 – GND (Terra)

- Também conhecido como terra, negativo ou simplesmente -, é usado para completar um circuito, onde o nível elétrico está a 0 volt.

8 – VIN

- *Voltage In* (Tensão de Entrada), onde você pode conectar fontes de alimentação externas.

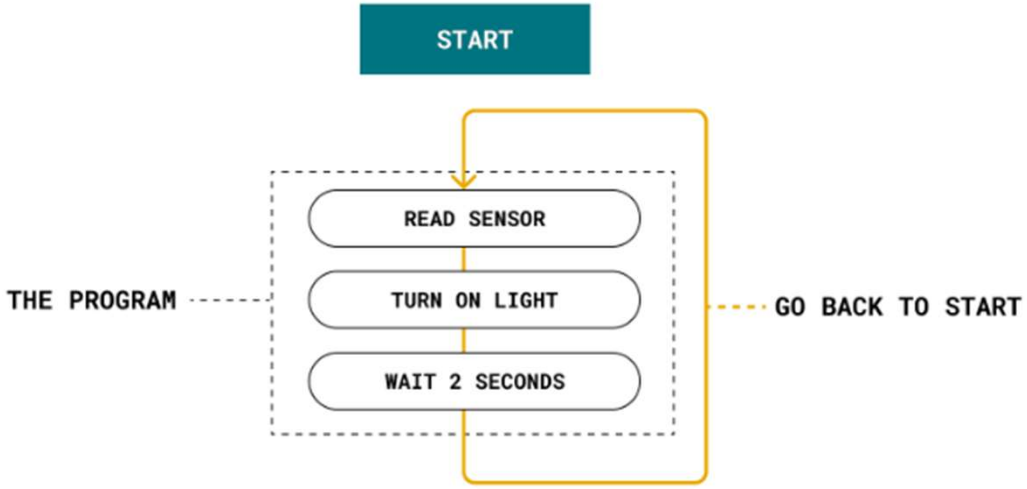
Explorando o Arduino

Funcionamento

- O Arduino UNO, como os demais microcontroladores, funciona rodando o código em looping.
- O programa compilado no Arduino será executado a partir do momento que o Arduino for ligado.

```
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
```

Código em branco de um Arduino



Operações básica do Arduino

Interrupt ou Polling

- Interrupt e Polling são métodos para executar um evento de um microcontrolador (Ex: Executar algo a partir de um botão apertado).
- Uma vez que o microcontrolador trabalha com um loop, ele executa as suas ações em ordem de linha código.
- Dependendo da aplicação, é necessário que o microcontrolador pare tudo que ele está fazendo para executar esse evento.
- O ato de fazer o microcontrolador parar tudo que ele está fazendo se chama **Interrupt.**
- Caso contrário, é utilizado o método de **Polling**, onde há a necessidade de aguardar a ordem do código para que essa execução ocorra.

Interrupt

Função Interrupt do Arduino

```
1  const int buttonPin = 2;  // Usando pino 2 (INT0)
2  const int ledPin = 13;
3  volatile bool buttonPressed = false;
4
5  void setup() {
6    pinMode(ledPin, OUTPUT);
7    pinMode(buttonPin, INPUT_PULLUP);
8    attachInterrupt(digitalPinToInterrupt(buttonPin), buttonISR, FALLING);
9    Serial.begin(9600);
10 }
11
12 void loop() {
13   // Operação 1: Leitura analógica do pino A0
14   int sensorValue = analogRead(A0);
15   Serial.print("Sensor: ");
16   Serial.println(sensorValue);
17
18   // Operação 2: Cálculo simples
19   float calc = sensorValue * 0.4887;
20   Serial.print("Calculo: ");
21   Serial.println(calc, 2);
22
23   // Verifica interrupção
24   if(buttonPressed) {
25     digitalWrite(ledPin, !digitalRead(ledPin));
26     Serial.println("LED toggle (INTERRUPT)");
27     buttonPressed = false;
28   }
29
30   delay(300); // Atraso para visualização
31 }
32
33 void buttonISR() {
34   static unsigned long last = 0;
35   if(millis() - last > 200) { // Debounce
36     buttonPressed = true;
37     last = millis();
38   }
39 }
```

Ação do Interrupt fora do Loop

Polling

```
const int buttonPin = 2;  // Qualquer pino digital
const int ledPin = 13;
int lastButtonState = HIGH;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop() {
  // Operação 1: Leitura de temperatura simulada
  float temp = analogRead(A0) * 0.4887 - 50.0;
  Serial.print("Temp: ");
  Serial.print(temp, 1);
  Serial.println("C");

  // Operação 2: Geração de número aleatório
  int randNum = random(100);
  Serial.print("Random: ");
  Serial.println(randNum);

  // Verificação do botão (polling)
  int reading = digitalRead(buttonPin);
  if(reading != lastButtonState && reading == LOW) {
    digitalWrite(ledPin, !digitalRead(ledPin));
    Serial.println("LED toggle (POLLING)");
    delay(50); // Debounce
  }
  lastButtonState = reading;

  delay(300); // Atraso para visualização
}
```

Ação desejada dentro do loop (Pooling)

CENTRO UNIVERSITÁRIO UniBTA Campus Paulista

Interrupt ou Polling

Interrupt	Polling
Resposta imediata	Resposta no próximo loop
Ideal para eventos críticos	Adequado para projetos simples
Complexidade moderada	Implementação mais simples
Não bloqueia o loop principal	Pode afetar desempenho se muito busy
Requer pinos específicos (2 ou 3 Uno)	Funciona em qualquer pino digital

Diferenças entre Interrupt e Polling

Interrupt ou Polling

Tipo de Interrupt do Arduino UNO e suas prioridades

0 – Botão Reset

1 - Interrupts Externas (2 pinos)
INT0 - Pino Digital 2
INT1 - Pino Digital 3

2- Pin Change Interrupts (todos os pinos)
Grupo PCINT0 (Pinos D8-D13)
Grupo PCINT1 (Pinos A0-A5)
Grupo PCINT2 (Pinos D0-D7)

3 - Interrupts de Temporizador (Timer)
TIMER0 - Usado para funções delay(), millis()
TIMER1 - 16-bit
TIMER2 - 8-bit

4 - Interrupts de Comunicação
USART RX Complete - Recepção serial
USART UDRE - Buffer de transmissão vazio
SPI STC - Transferência SPI completa
TWI (I2C) - Interrupt de interface I2C

5 - .....

Explorando o Arduino

Sinal Analógico

- Um sinal analógico geralmente está limitado a uma faixa específica. No Arduino, essa faixa é tipicamente de 0-5V ou 0-3.3V.

- Por exemplo, se usarmos um potenciômetro (um componente analógico utilizado para alterar a resistência em um circuito), podemos ajustar manualmente essa faixa (0-5V). No programa, isso é representado em uma escala de 0-1023, que corresponde a uma resolução de 10 bits.

Potenciômetro

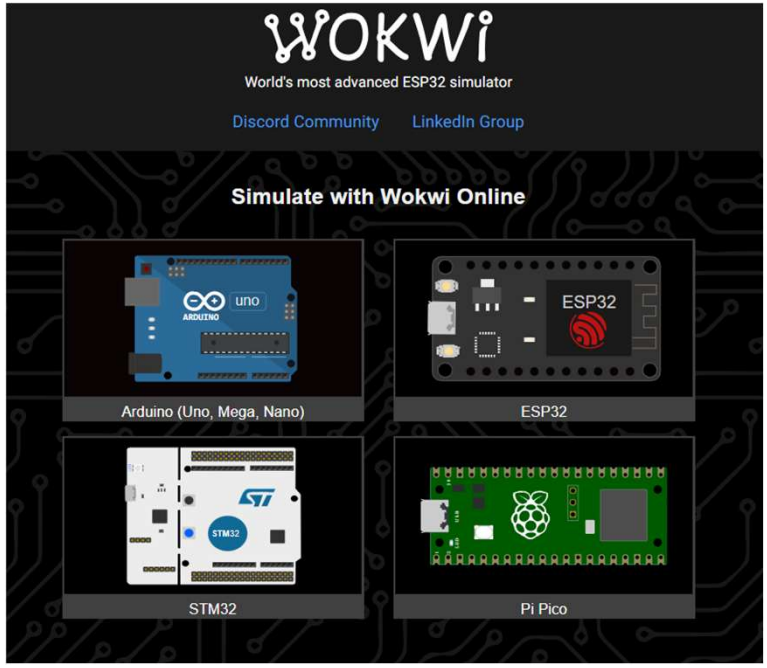
Explorando o Arduino

Interface de programação do Arduino – IDE (Integrated development environment)

- Para programar o microcontrolador do Arduino UNO, utilizamos uma interface de programação, chamada de IDE.

IDE do Arduino

Para começarmos a trabalhar com o Arduino UNO, vamos utilizar um simulador online chamado: **Wokwi**



Simulador Wokwi

https://wokwi.com/