Aumentando as capacidades de programação com microcontroladores

Desenvolvido por Mirela TIBU

"Grigore Moisil" Theoretical Highschool of Informatics , Iasi, Romania

A Trainers Toolkit To Foster STEM Skills Using Microcontroller Applications

Co-funded by the Erasmus+ Programme of the European Union

Project No. 2019-1-RO01-KA202-063965

This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Conteúdos

Informática

Objetivo
Descrição
Objectivos de Aprendizagem
Metodologias de Aprendizagem
Grupo Alvo
Microprocessador VS Microcontrolador
Arquitetura de Sistemas Embebidos
Compreender a programar
Conceitos usando Microcontroladores
Áreas Científicas cobertas
Avaliação
Bibliografia

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Objetivo

Expressar uma forma criativa de pensar, na estruturação e resolução de problemas

Criar hábitos para usar conceitos e métodos algorítmicos específicos de computação na abordagem de uma variedade de problemas

Manifestar atitudes em relação à ciência e ao conhecimento

Manifestar iniciativa e vontade de abordar várias tarefas

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Descrição

- Abordagem ITC - Microprocessador VS Microcontrolador
  - identificando áreas onde os computadores/sistemas embebidos são usados na vida diária
  - descrevendo a arquitetura de hardware para um computador e um sistema embebido
  - comparando recursos de microprocessadores e microcontroladores
- Abordagem de programação - Entendendo os conceitos de programação, usando aplicações de microcontroladores
  - instruções de programação estruturada – decisões, ciclos (IF, WHILE, FOR)
  - declaração e chamada de funções void e non-void
  - usando arrays nas aplicações
  - analisando a funcionalidade de dispositivos Arduino para reconhecer as etapas de execução de instruções de programação

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Objetivos de Aprendizagem

- Identificar aplicações informáticas na vida social – tomar consciência do impacto dos sistemas embebidos na vida quotidiana
  - Identificar as semelhanças e diferenças entre um microprocessador e um microcontrolador na arquitetura de computação e sistemas embebidos
  - Praticar a implementação dos elementos de programação estruturada - decisões, ciclos, funções; representação e uso de dados num array
  - Visualizar o efeito da execução de várias sequências de programas através de dispositivos baseados em microcontroladores

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Metodologias de Aprendizagem

- Explicação
- Demonstração
- Conversação
- Algoritmização
- Implementações

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Grupo Alvo

Alunos do Ensino Secundário – 9º e 10º anos

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

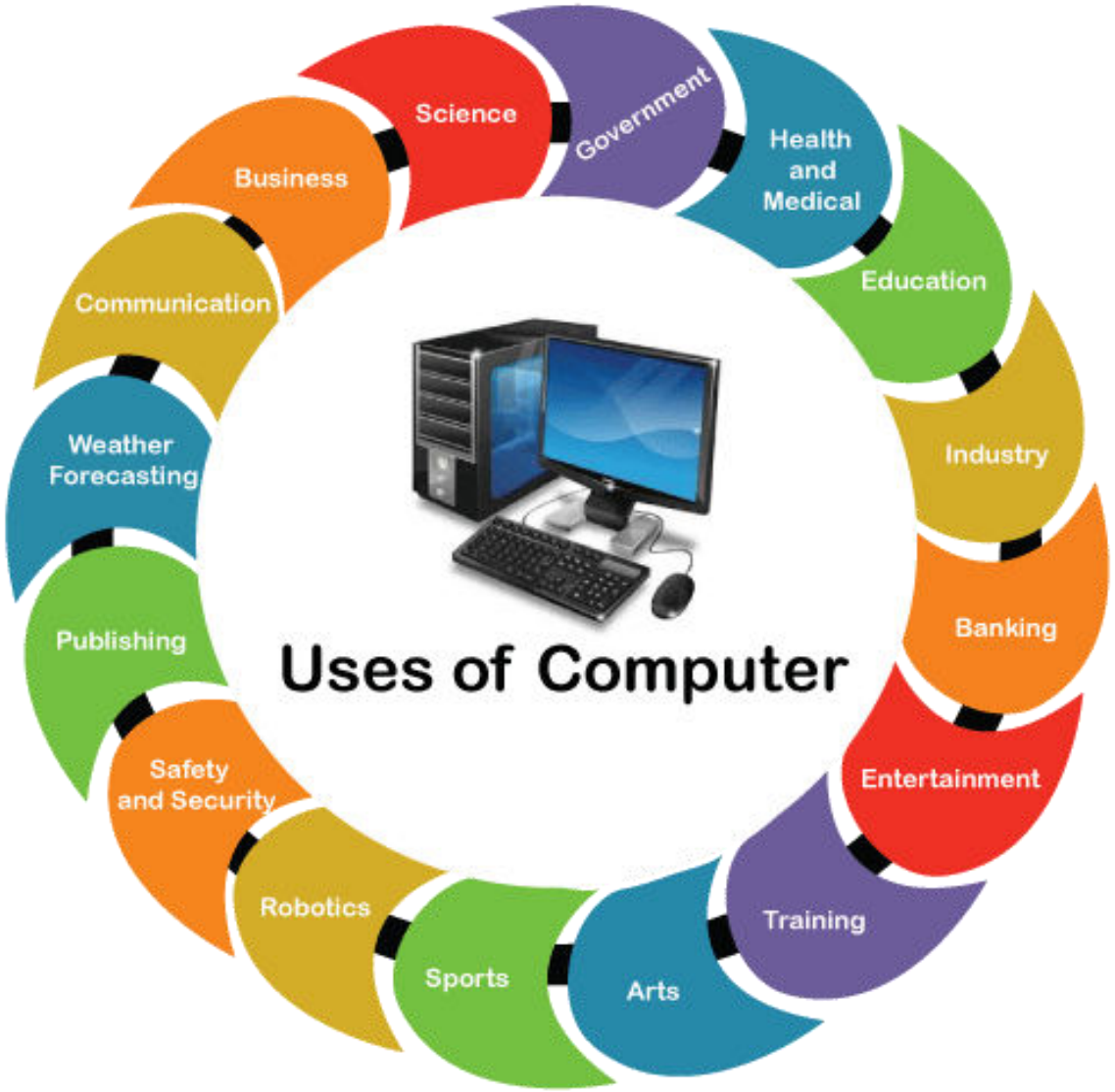
Co-funded by the Erasmus+ Programme of the European Union

Computadores e Sistemas Embebidos

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Computadores e Vida Diária



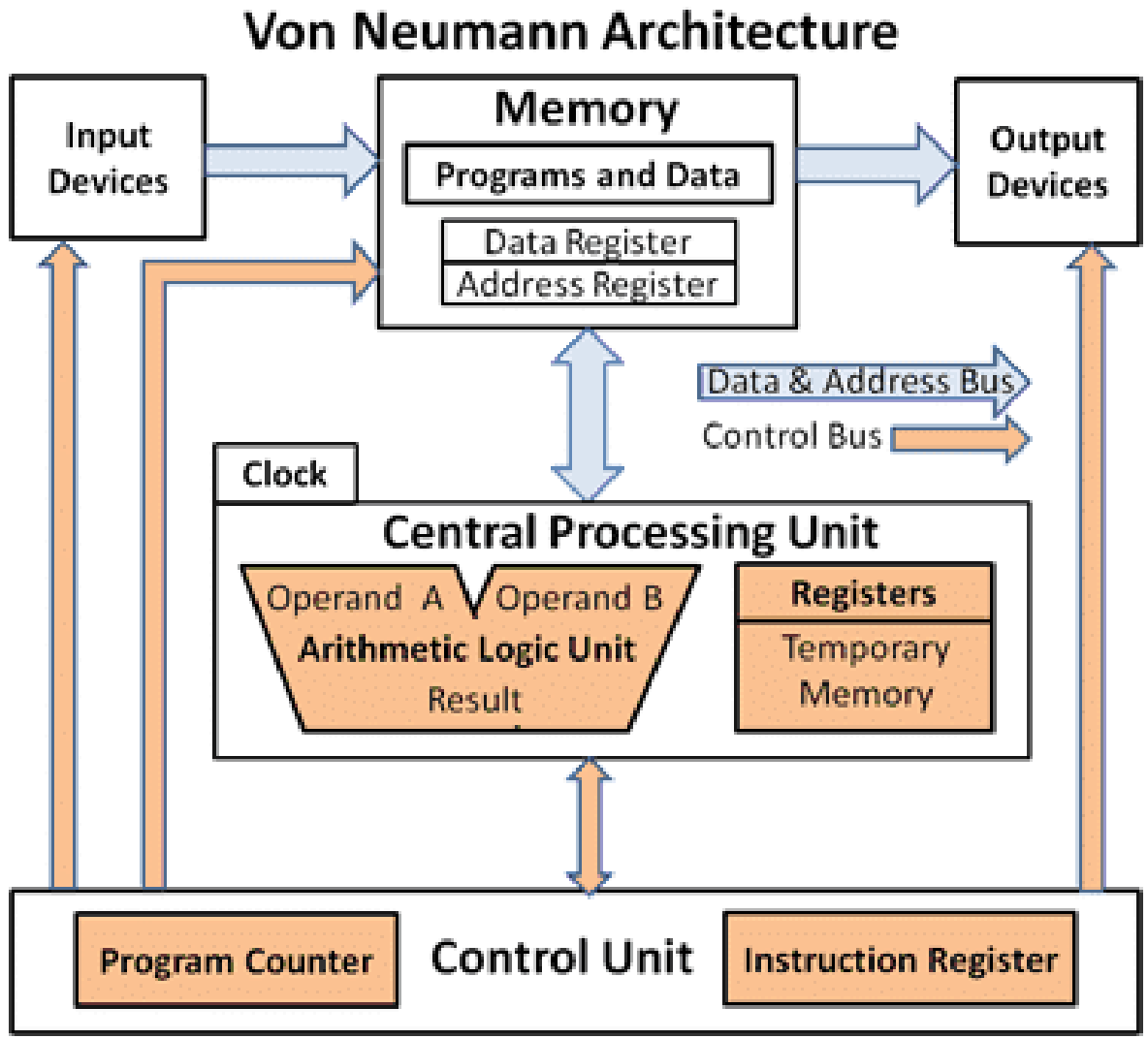
- Os computadores fazem parte do nosso dia a dia.
- Computador = Hardware + Software
- Hardware - componentes físicos
- Software - programas, procedimentos e rotinas que informam o computador o que fazer
- ONDE usamos os computadores?

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Arquitetura do Computador Hardware

- ✓ CPU = Microprocessador - “o cérebro” do nosso computador – faz todas as operações aritméticas e lógicas (ALU) e controla todas as atividades do sistema
- ✓ Unidade de Memória – armazena dados e programas
  - RAM – Memória de acesso aleatório
    ROM – Memória Somente de Leitura
- ✓ Dispositivos de entrada/saída



Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

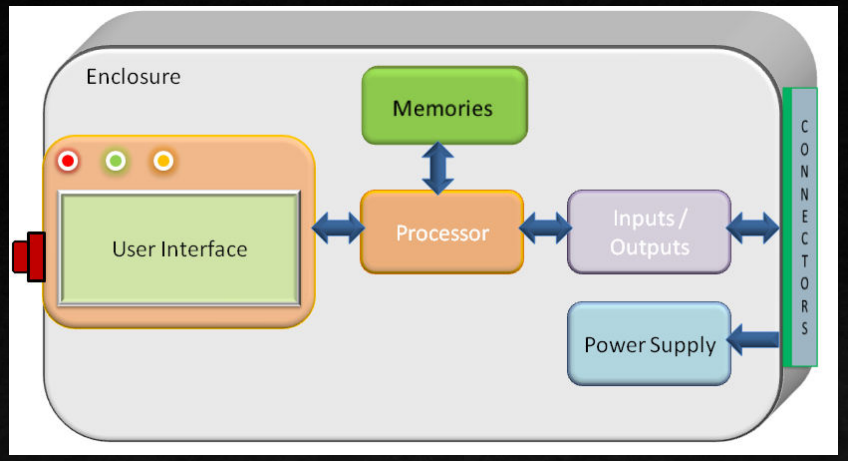
Sistemas Embebidos na Vida Diária

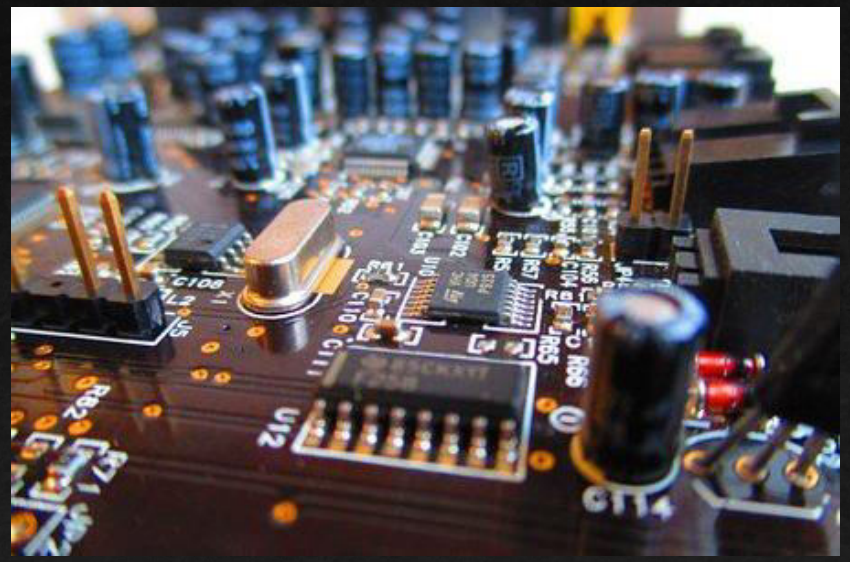
- Um sistema embebido é um computador com um propósito especial, que é usado dentro de um dispositivo
- é baseado num microcontrolador que é um chip otimizado para controlar dispositivos eletrónicos; é armazenado num único circuito integrado, dedicado a realizar uma determinada tarefa e executar uma aplicação específica
- ONDE usamos os sistemas embebidos?

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

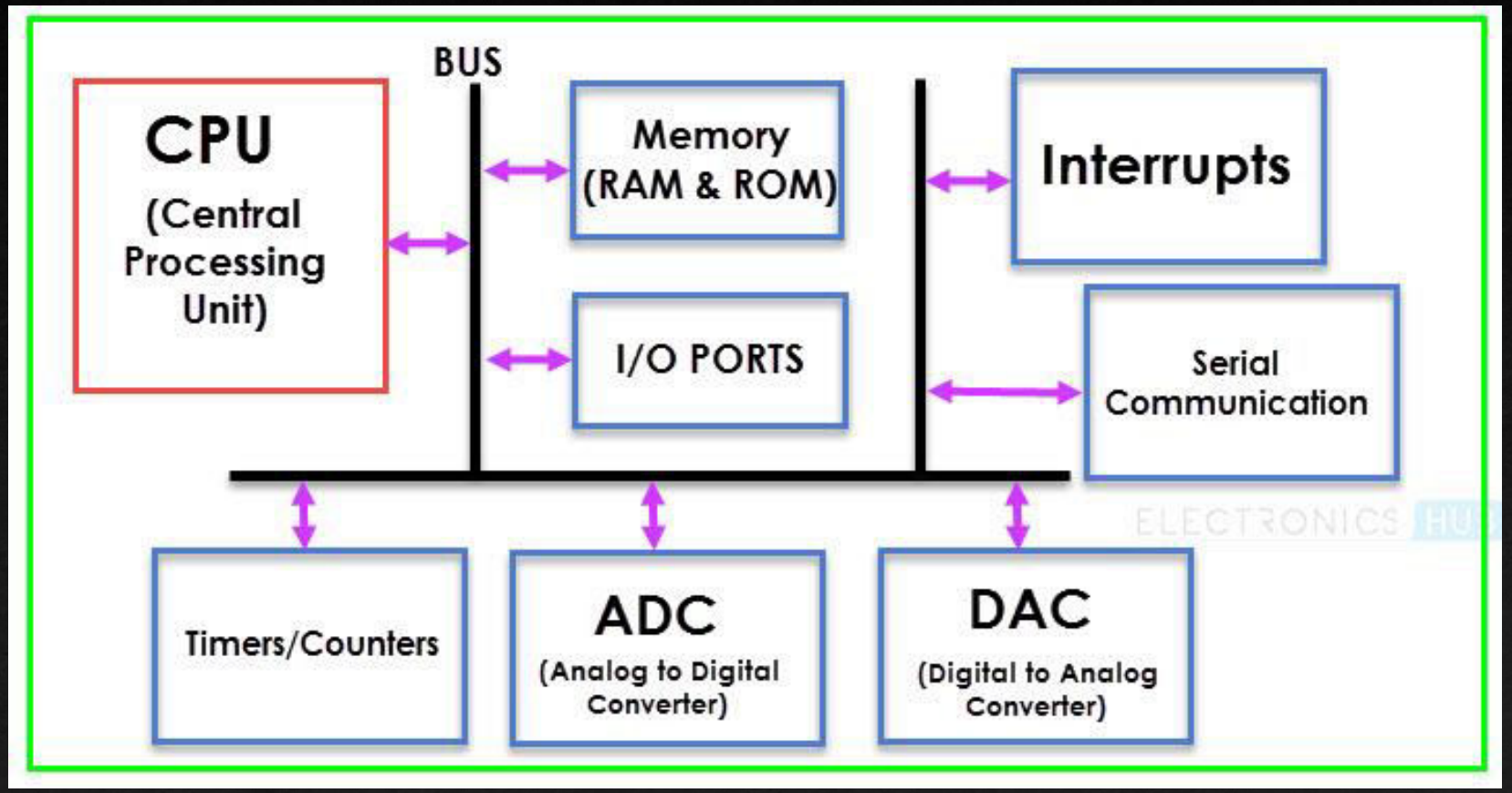
Co-funded by the Erasmus+ Programme of the European Union

Arquitetura dos Sistemas Embebidos





- CPU – é um microcontrolador ou um microprocessador



Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Qual é o melhor?

Microprocessador	Microcontrolador
É o coração do sistema do Computador.	O coração de um Sistema embebido
É apenas um processador, por isso a memória e os componentes de I/O precisam de ser conectados externamente	Tem um processador com uma memória interna e componentes de I/O
A Memória e os I/O têm de ser conectados externamente, para que o circuito se torne maior	Memória e I/O já estão presentes, e o circuito interno é pequeno
Pode ser usado em sistemas compactos	É usado em sistemas compactos
O custo de todo o sistema é elevado	O custo de todo o sistema é baixo
Devido às componentes externas, o consumo total de energia é elevado. Não é o ideal para os dispositivos que funcionam com energia armazenada como baterias.	Como os componentes externos são baixos, o consumo total de energia é menor. Portanto, podem ser usados com dispositivos que funcionam com energia armazenada, como baterias
A maioria deles não tem recursos de economia de energia	A maioria deles oferece modo de economia de energia
Maioritariamente usado em computadores pessoais	Maioritariamente usado em sistemas embebidos.
Baseados no modelo de Von Neumann	Baseados na arquitetura Harvard

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Qual é o melhor?

tem um número menor de registradores, então mais operações são baseadas em memória	tem mais registro, então os programas são mais fáceis de escrever
é uma unidade de processamento central em um único chip integrado baseado em silício	é um subproduto do desenvolvimento de microprocessadores com um CPU junto com outros periféricos
não tem RAM, ROM, unidades de entrada-saída, temporizadores e outros periféricos no chip	tem um CPU junto com RAM, ROM e outros periféricos embutidos em um único chip
usa um barramento externo para fazer interface com RAM, ROM e outros periféricos	usa um barramento de controle interno
Os sistemas baseados em microprocessadores podem funcionar a uma velocidade muito alta devido à tecnologia envolvida	Sistemas baseados em microcontroladores funcionam até 200 MHz ou mais, dependendo da arquitetura
é usado para aplicações de uso geral que permitem lidar com cargas de dados	é usado para sistemas específicos de aplicações
É complexo e caro, com um grande número de instruções para processar	É simples e barato com menor número de instruções para processar

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Instrução IF, declaração e chamada de funções void e definição const

Tarefa: Programe um dispositivo capaz de ler o estado de um potenciómetro (uma entrada analógica) e acenda um LED somente se o potenciómetro.
Deve imprimir o valor analógico independente do nível.

```
const int analogPin = A0;
   // pino ao qual o sensor está conectado
const int ledPin = 13;
   // pino ao qual o LED está conectado
const int valmin = 400;
   // um nível arbitrário de valmin

void setup() {
   // inicializa o pino LED pin como output:
  pinMode(ledPin, OUTPUT);
   // inicializa comunicações em série:
  Serial.begin(9600); }
```

Entendendo os conceitos de programação usando aplicações de microcontroladores

```
void loop() {
  // lê o valor do potentiómetro:
  int analogValue = analogRead(analogPin);

  // se o valor analógico for suficientemente alto,
  // ligar o LED:
  if (analogValue > valmin) {
    digitalWrite(ledPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
  }
  // imprime o valor analógico:
  Serial.println(analogValue);
  delay(1);    // atraso entre leituras
}
```


Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Entendendo os conceitos de programação usando aplicações de microcontroladores

Definição da Variável

```
<typeVar> nameVar [= value];
```

declara uma variável de um tipo específico, que determinará o tamanho dos valores, o comprimento e o tipo de representação da memória

Tipos usuais de variáveis usadas em aplicações Arduino

int = tipo numérico para variáveis/constantes;
é representado em 4 bytes e pode armazenar valores entre aprox. $-2*10^9 \cdots 2*10^9$

bool = tipo booleano para variáveis/constantes;
é representado em 1 byte e pode armazenar valores falso (0) e verdadeiro (1)

Definição da Constante

```
const <typeConst> nameConst = value;
```

define uma constante de typeConst com um valor específico

Estrutura de decisão

```
if (Condition) {
  instructions_A
    // fazer instruções_A se a
    // Condição for verdadeira
} else {
 instructions_B
  // fazer instruções_A se a
  // Condição for falsa
}
```

Funções void – declaração

```
void nameFunction(list of formal parameters)
{declaração de variáveis locais
   instruções}
```


Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Funções void – declaração

```
void nameFunction(parâmetros formais)
{ declaração de variáveis locais
  instruções
}
```

Funções non-void - declaração

```
resultType nameFunction(parâmetros formais)
{ declaração de variáveis locais
  instruções
  expressão de retorno;  //
}
```

onde parâmetros formais é uma lista de tipos e nomes de parâmetros usados em instruções de função

Chamar uma função

```
nameFunction(lista de parâmetros atuais)
```

- Funções void – A chamada é uma instrução
- Funções non-void – A chamada é um operando na expressão com o mesmo tipo que o resultType

! Os parâmetros formais e reais devem ter o mesmo tipo, número e devem estar na mesma ordem

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Funções específicas Arduino

setup()

Função void - chamado quando um sketch é iniciado e será executado apenas uma vez, após cada inicialização ou reinicialização da placa Arduino semelhante a main(). Use-o para inicializar variáveis, fixar modos, começar a usar bibliotecas, etc.

loop()

Função void - loops consecutivamente, permitindo que o programa Arduino mude e responda Ele é chamado após a função setup() , que inicializa e define os valores iniciais.

pinMode(pin, mode)

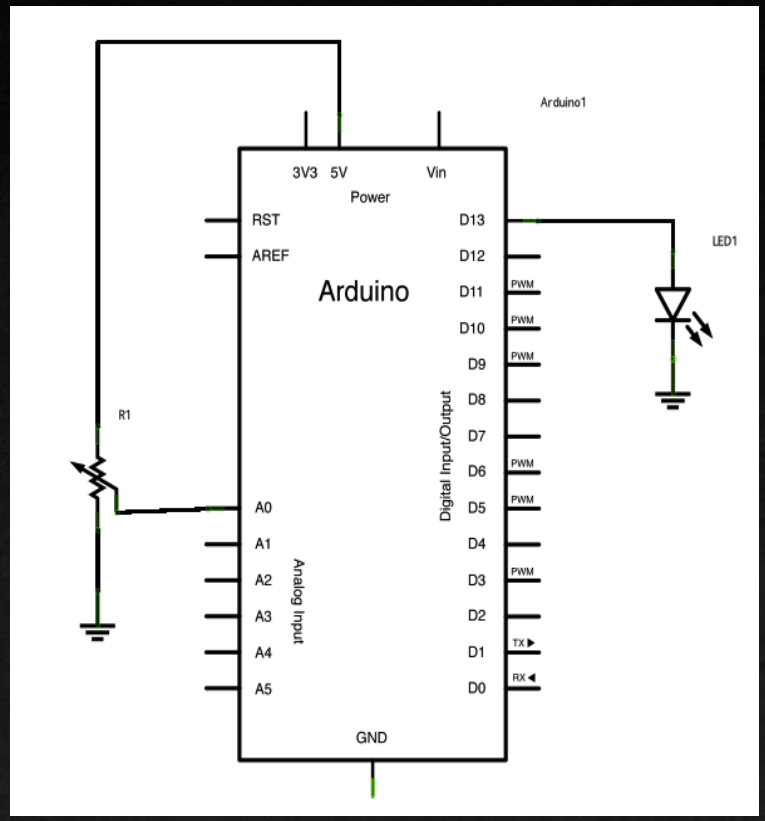
Função void com parâmetros

- pin: o número do pino do Arduino para desativar o modo
  mode: INPUT, OUTPUT, or INPUT_PULLUP

delay(milisec)

Função void com parâmetros

- milisec: número de milissegundos para pausar o programa



esquema elétrico do dispositivo Arduino

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Funções específicas Arduino

digitalWrite(pin, value)

Função void com parâmetros

- pin: o número pin do Arduino
- value: ALTO ou BAIXO

digitalRead(pin)

Função com parâmetros

- pin: o número pin do Arduino
- return value: ALTO ou BAIXO

analogRead(pin)

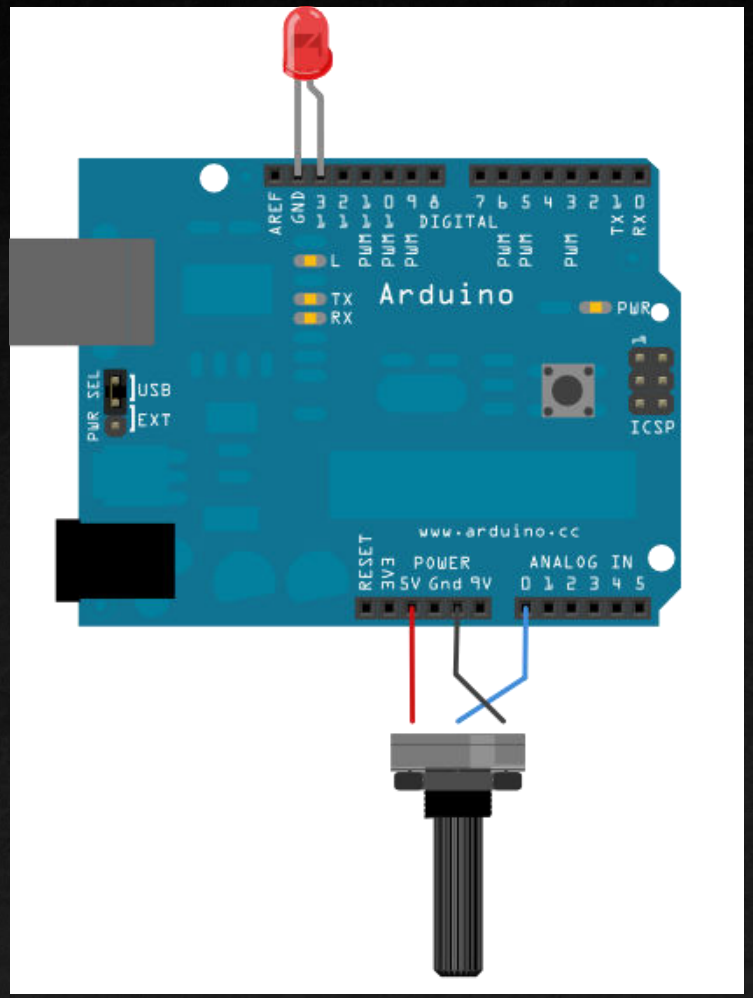
Função int com parâmetro

- pin: o nome do pino de entrada analógica a ser lido (A0 a A5 na maioria das placas)
- Retorna a leitura analógica no pino.

Constantes de Níveis de Pino do Arduino

ALTO e BAIXO

pino	INPUT	OUTPUT
ALTO	voltagem > 3.0V	5V
BAIXO	voltagem > 3.0V	0V



Dispositivo Arduino

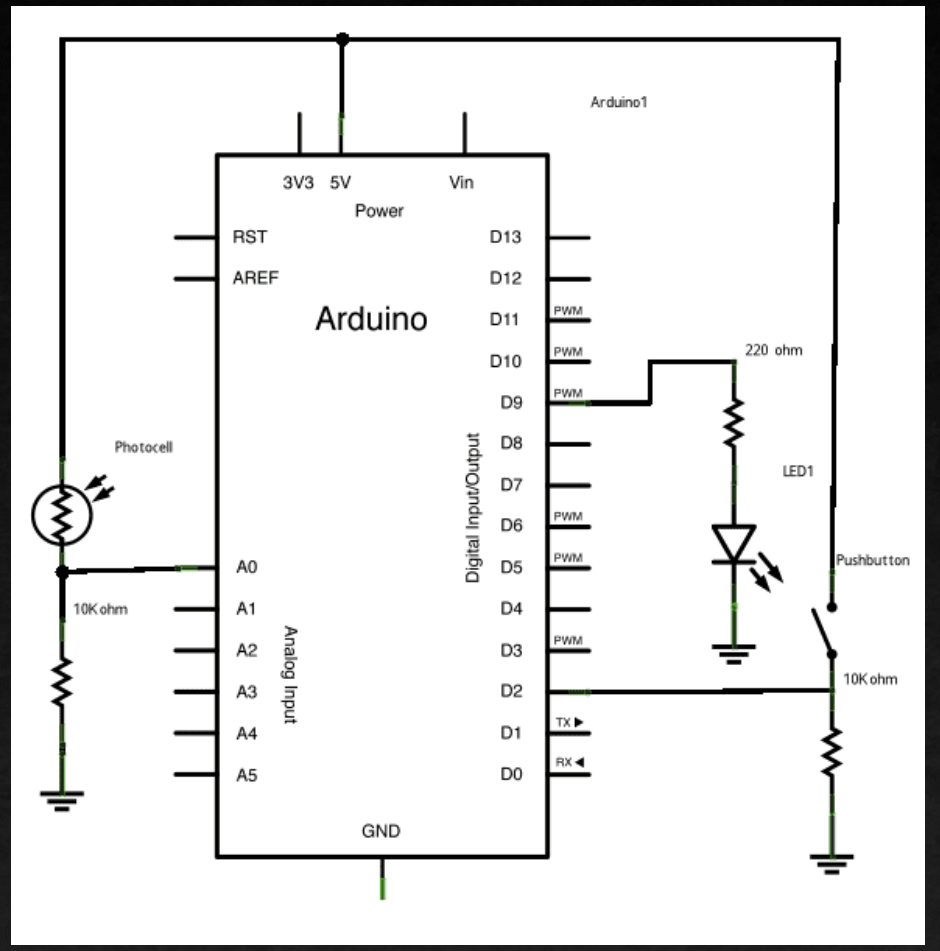
Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Declaração WHILE

Tarefa: Programar um dispositivo capaz de ler por cinco segundos a entrada do sensor e calibrar definindo o mínimo e o máximo dos valores esperados para as leituras realizadas durante o loop.

```
const int sensorPin = A0;
    // pino ao qual o sensor está
conectado
const int ledPin = 9;
    // pino ao qual o LED está conectado
    // variáveis:
int sensorValue = 0;
    // o valor do sensor
int sensorMin = 1023;
    // valor mínimo do sensor
int sensorMax = 0;
    // valor máximo do sensor
```



Esquema para o dispositivo Arduino

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Declaração WHILE

```
void setup() {
  // liga o LED para sinalizar o início
  // do período de calibração:
     pinMode(13, OUTPUT);
     digitalWrite(13, HIGH);
  // calibra durante os primeiros cinco segundos
  while (millis() < 5000) {
       sensorValue = analogRead(sensorPin);
       if (sensorValue > sensorMax){
           sensorMax = sensorValue;
       }
       if (sensorValue < sensorMin){
          sensorMin = sensorValue;
       }
    }
  digitalWrite(13, LOW);    // sinaliza o fim do período de calibração
}
```


Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Entendendo os conceitos de programação usando aplicações de microcontroladores

Determinar o Mínimo e o Máximo valor
De uma série de valores

Algoritmo

Passo 1. Defina as variáveis para sensorMin
com o valor máximo possível e sensor Max com o mínimo valor possível
Passo 2. Compare o valor atual com sensorMin e, se for menor, atualizar sensorMin
Passo 3. Compare o valor atual com sensorMax e, se for maior, atualizar sensorMax

Declaração WHILE

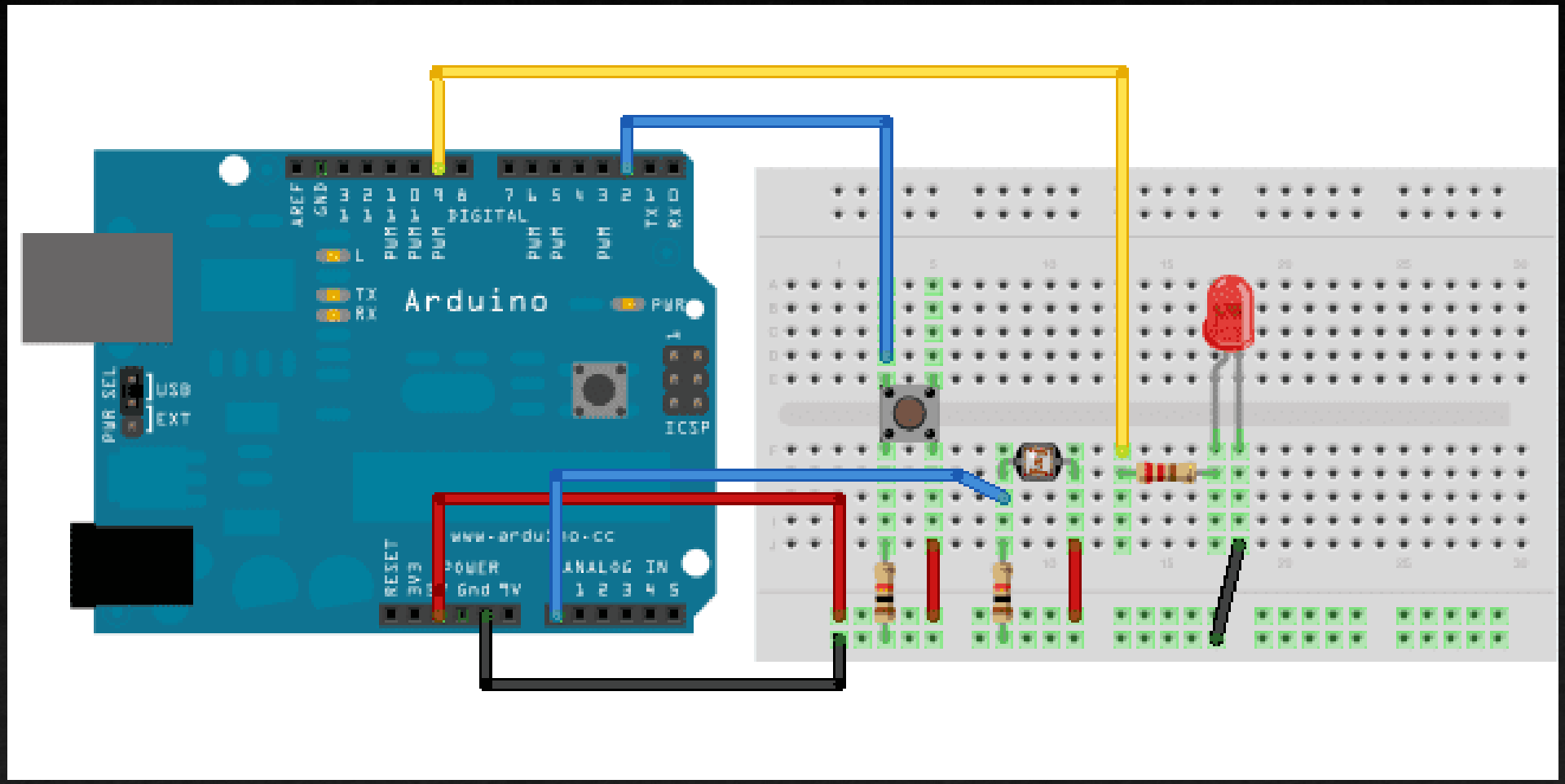
```
while (Condition) {
  instructions_A
}
```

Execução

Passo 1. A Condição é avaliada
Passo 2. Se a Condição for Verdadeira
2.1. Instruções A serão executadas
2.2. Ir para o Passo 1.
Se Condição for Falsa, a execução do programa sai do ciclo

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.
Co-funded by the Erasmus+ Programme of the European Union

Entendendo os conceitos de programação usando aplicações de microcontroladores



Explicação do Dispositivo Arduino para o ciclo WHILE

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Declaração FOR e manipulação de ARRAYs

```
int timer = 100;
   // Quanto maior o número,
   // mais lento o tempo.
int ledPins[] = { 2, 7, 4, 6, 5, 3};
   // um array de números de pinos aos quais os LEDs estão
conectados
int pinCount = 6;
  // o número de pinos (o comprimento do array)
void setup() {
  // os elementos do array estão numerados de 0 a
(pinCount-1)
  // use um loop for para inicializar cada pino como uma
saída:
  for (int thisPin = 0; thisPin < pinCount; thisPin++) {
    pinMode(ledPins[thisPin], OUTPUT);
  }
}
```

Tarefa: Programa um dispositivo capaz de acender um série de LEDs ligados a pinos cujos números não são contíguos nem necessariamente sequenciais. Para fazer isso, os números PIN serão armazenados num ARRAY e, em seguida, usa ciclos FOR para iterar sobre o array.

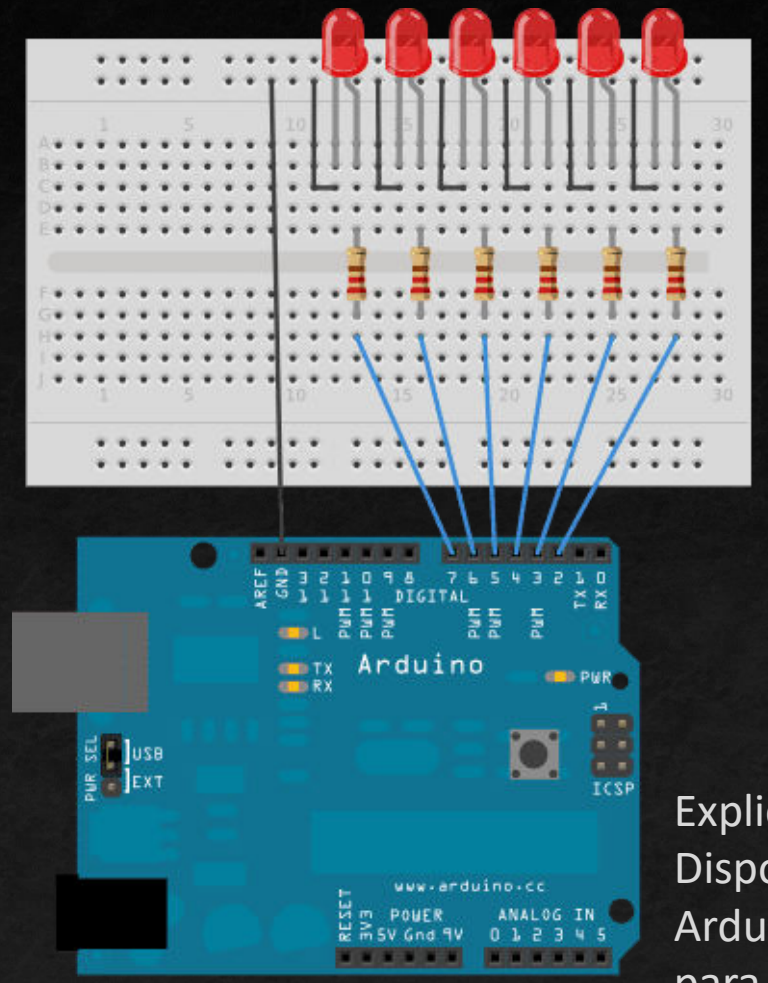
Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Declaração FOR e manipulação de ARRAYs

```
void loop() {
// loop do pino mais baixo para o mais alto:
 for (int thisPin = 0; thisPin < pinCount; thisPin++) {
   digitalWrite(ledPins[thisPin], HIGH);
      // ligar o pino:
   delay(timer);
   digitalWrite(ledPins[thisPin], LOW);
      // desligar o pino:
 }
      // loop do pino mais alto para o mais baixo:
 for (int thisPin=pinCount-1; thisPin >= 0; thisPin--){
      // ligar o pino:
    digitalWrite(ledPins[thisPin], HIGH);
    delay(timer);
      // desligar o pino:
    digitalWrite(ledPins[thisPin], LOW);
 }
}
```

Entendendo os conceitos de programação usando aplicações de microcontroladores

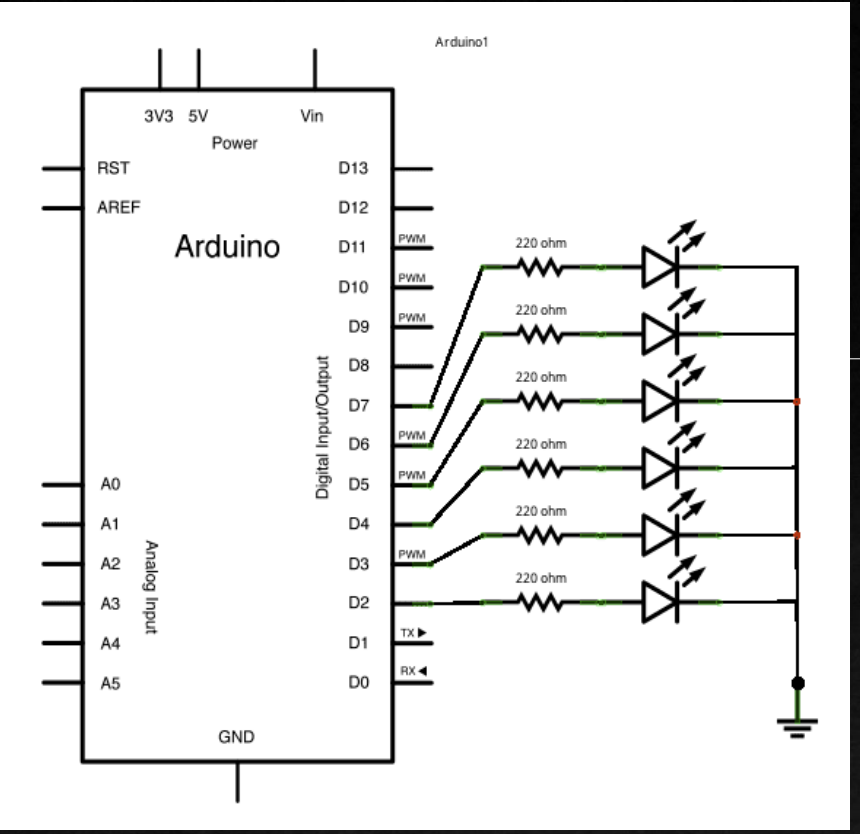


Explicação do Dispositivo Arduino para o ciclo FOR

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Entendendo os conceitos de programação usando aplicações de microcontroladores



Esquema para o dispositivo Arduino

Declaração FOR

```
for(int counter = initialVal; counter <= finalVal; counter++) {
    instructions_A
}
```

Execução

Passo 1. O contador é definido com initialValue

Passo 2. Se contador <= finaValue for Verdadeiro

- 2.1. Instrução A será executada
- 2.2. O contador é incrementado com 1
- 2.3. Ir para o Passo 2

Passo 3. Se contador <= finaValue for Falso, a execução do programa sai do ciclo

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Entendendo os conceitos de programação usando aplicações de microcontroladores

Organizando os dados no ARRAY

Array = uma coleção de dados do mesmo tipo, organizados numa zona de memória contígua e referenciados com um único nome, que é um ponteiro (endereço de memória) do primeiro elemento do array.

Declaração:

```
valuesType arrayName[numberOfElements];
```

Inicialização:

- junto com a declaração

```
int ledPins[] = { 2, 7, 4, 6, 5, 3};
```

- por atribuição

```
digitalWrite(ledPins[thisPin], HIGH);
```

- lendo os valores da entrada

Fazendo referência a um valor específico do array

```
A[expressionIndex]
```

expressionIndex é um inteiro de [0,count-1], indicando a posição do elemento no array

Analisando o ARRAY para analisar e processar os seus elementos:

left-rigt

```
for (int it = 0; it < count; it++)
    process(A[it]);
```

rigt-left

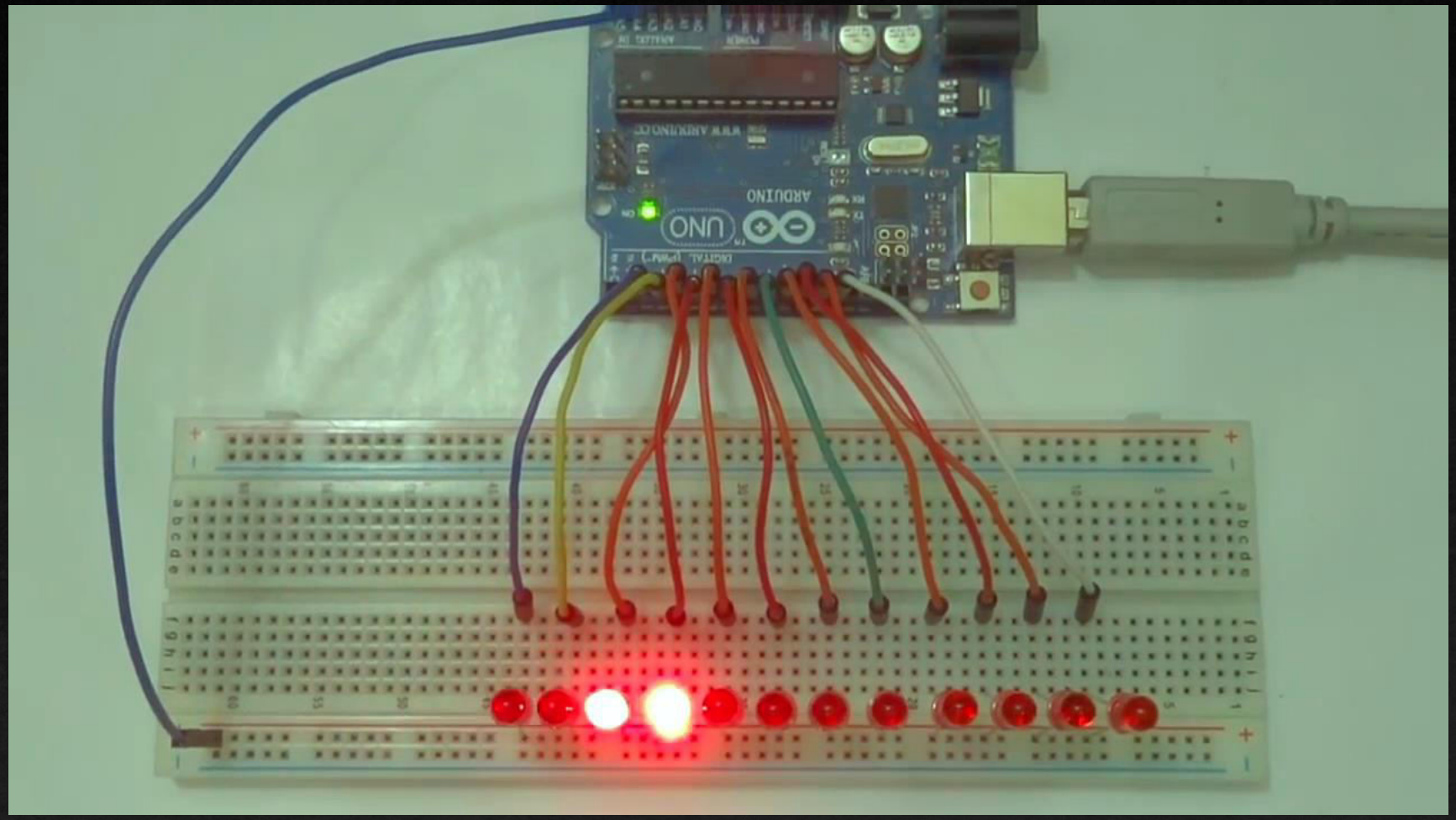
```
for (int it = count - 1; it >= 0; it--)
    process(A[it]);
```

onde count é o número de elementos do array e it é o índice usado para analisar o array

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Entendendo os conceitos de programação usando aplicações de microcontroladores



Aprendendo os ciclos FOR e os ARRAYs com o dispositivo Arduino

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Áreas Científicas Cobertas

Arquiteturas de Hardware para Computadores e Sistemas Embebidos

Programação Estruturada – tipos de dados, declarações (IF, WHILE, FOR), funções definidas para o utilizador

Programação de dispositivos baseados em microcontroladores(ex. Arduino)

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Avaliação

- Teste de escolha múltipla
- mini-projeto em grupos de 2-3 alunos – programação de dispositivos Arduino que:

✓ descrevam a operação de outras instruções específicas em programação estruturada

✓ para aplicar em situações reais - por exemplo, a operação de um semáforo RGV

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union

Bibliografia

Webgrafia

- https://creativecommons.org/2008/10/22/wired-on-arduino-and-open-source-computing/
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/WhileStatementConditional
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/ForLoopIteration
- https://commons.wikimedia.org/wiki/Category:Fritzing
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/ForLoopIteration
- https://commons.wikimedia.org/wiki/Category:Arduino_projects

Project No. 2019-1-RO01-KA202-063965
This project has been funded with support from the European Commission. The content reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the Erasmus+ Programme of the European Union