Arduino

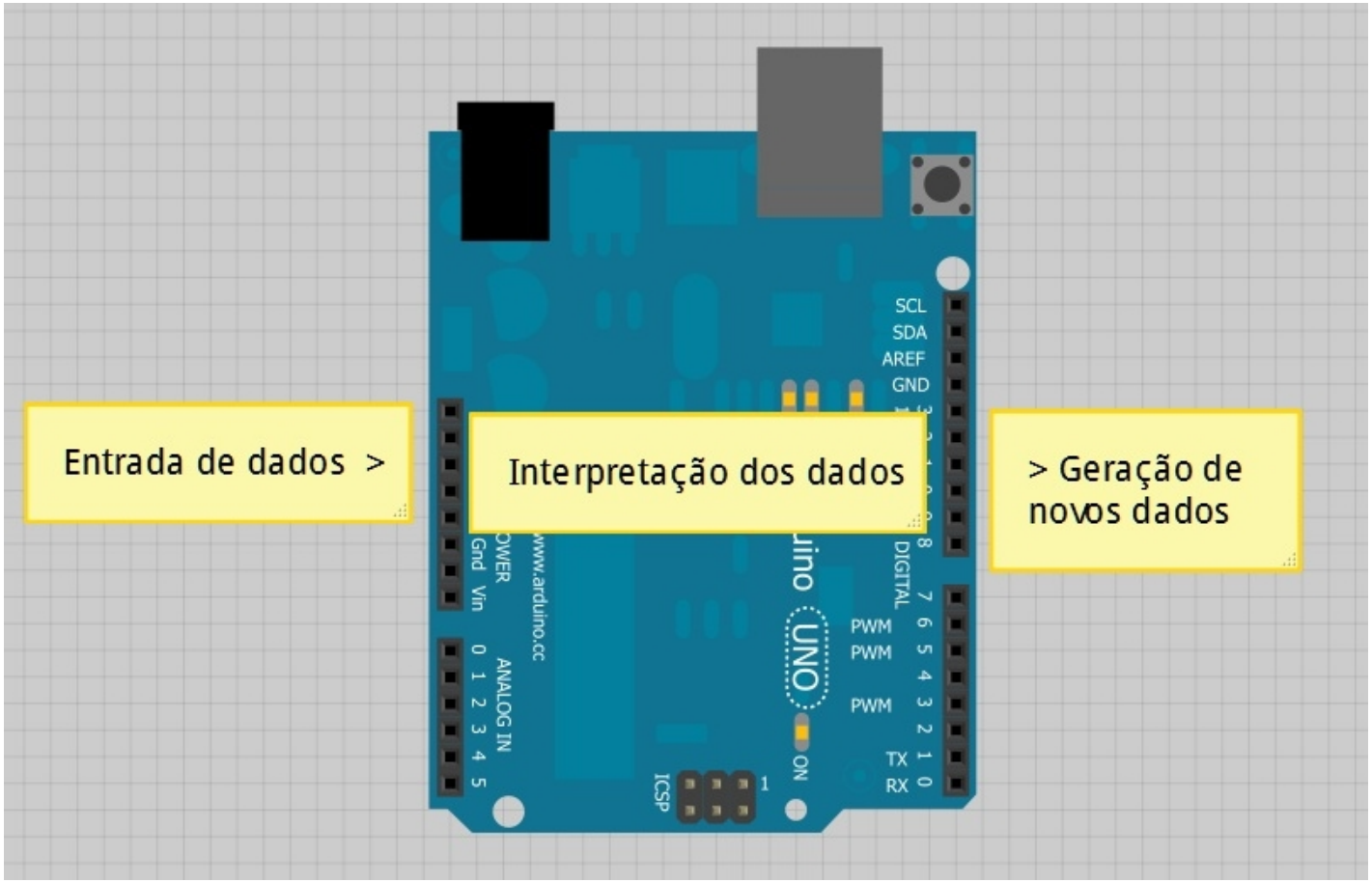
Introdução à plataforma de desenvolvimento Arduino

O que é o Arduino

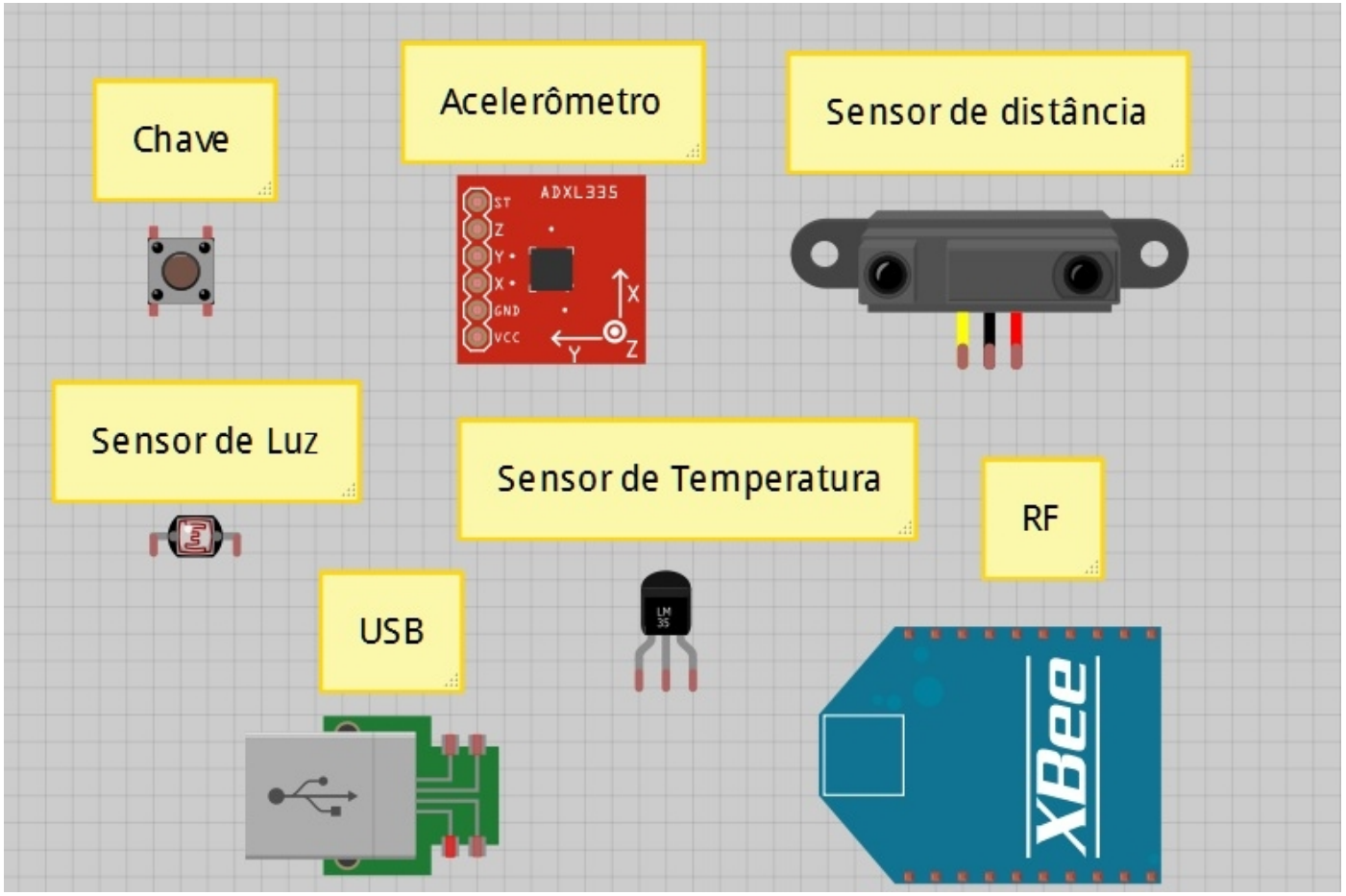
- É uma plataforma de desenvolvimento de hardware, microcontrolada de código aberto.

- Em termos práticos o Arduino é um pequeno computador em que é possível se interagir com o ambiente.

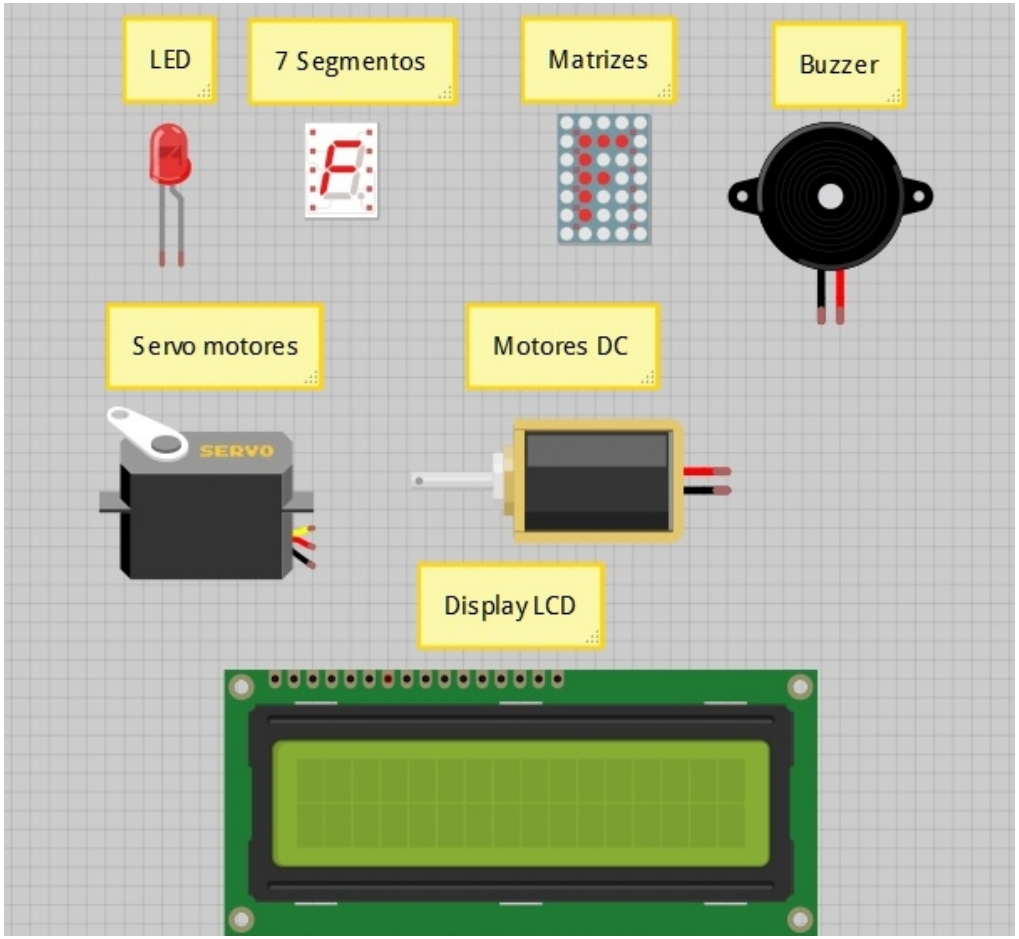
Princípio de Funcionamento



Entrada de Dados

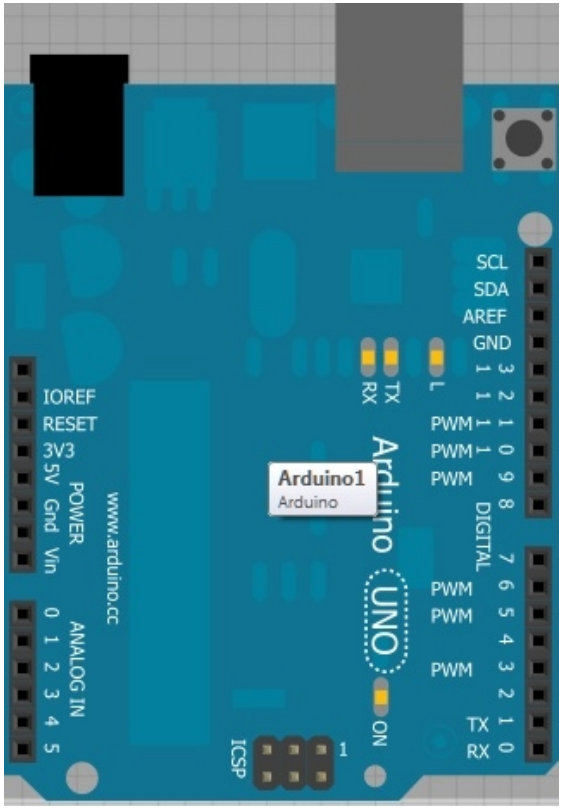


Geração de novos dados



Estrutura do Hardware

Microcontrolador	**ATmega328 ou ATmega168**
Tensão operacional	5 V
Tensão de alimentação (recomendada)	7-12 V
Pinos I/O digitais	14 (dos quais 6 podem ser Saídas PWM e RX/TX)
Pinos Analógicos	6
Memória flash	32K/16K
Clock	16Mhz
Corrente por pino I/O	40 mA



A IDE

Verifica o programa

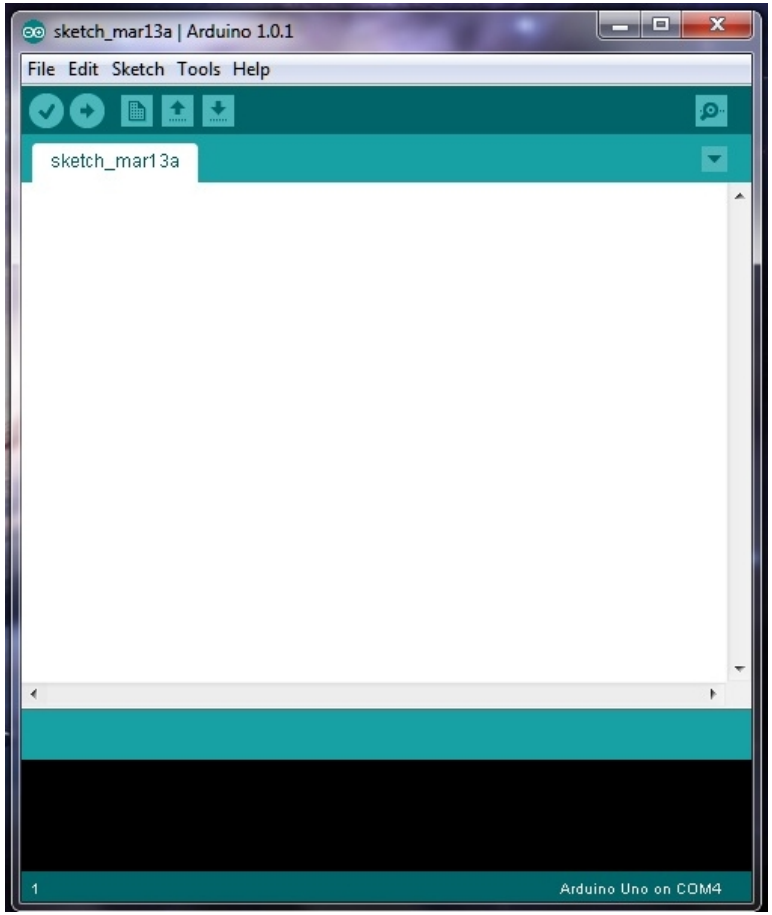
Grava o programa

Novo programa

Abre programa

Salva programa

Monitor Serial



Principais recursos da IDE

- A IDE do Arduino conta com diversos programas de exemplos em que se é utilizado os mais diversos dispositivos externos. (Ex: Teclados, sensores, display, etc.).
- Ela também conta com um aplicativo em que é possível se interagir com a placa através da usb do computador.

Linguagem de programação

- Nessa plataforma é utilizada como referência a linguagem C++.

- Com isso temos as referências de linguagem que são: estruturas, valores, funções.

Estruturas

- Estruturas de controle (if, else, break ...).
- Sintaxe básica (define, include...).
- Operadores aritméticos e de comparação(+, -, *, /, >>,<<...).
- Operadores Booleanos (||, &&, !...).
- Operadores aritméticos (++,--...).

Valores

- Tipos de dados (byte, int, char, float...).
- Conversões (char(), byte(), int()...).

Funções

- As funções são ferramentas com o intuito de direcionar e exemplificar as funcionalidades do microcontrolador. E já existem diversas funções prontas em bibliotecas que veem junto com o programa.

Exemplo de funções

- Digital: pinmode(), digitalwrite(), digitalread().
- Analógico: analogReference(), analogwrite().
- Tempo: millis(), micros(), delay().
- Matemáticas: min(), max(), abs(), pow().
- Números aleatórios: randomSeed(), random().
- Interrupções: interrupts(), nointerrupts().

Algumas considerações

- O programa possuí duas funções indispensáveis, são elas:

- void setup(){} – Que é responsável por configurar o hardware do arduino inicialmente.

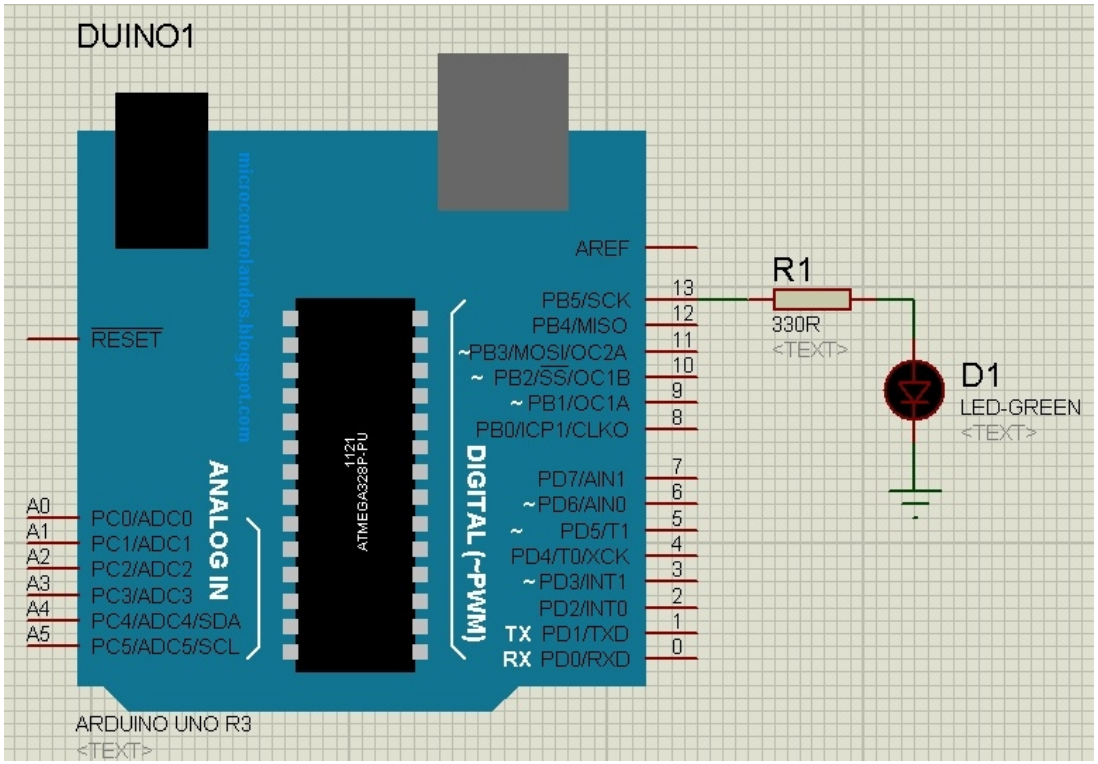
- void loop(){} – Função principal, responsável por rodar o programa repetidamente.

Aplicações

- 1º Exemplo - Pisca Led
- Objetivo: Entender as funções “setup()” e “loop()”, assim como conhecer a função “delay()”.
- Funcionamento: Um led piscará intermitentemente em intervalos de 1 segundo.

Aplicações

- Circuito



Aplicações

- Programa

```
int led = 13;

void setup()
{
  pinMode(led, OUTPUT);
}

void loop()
{
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

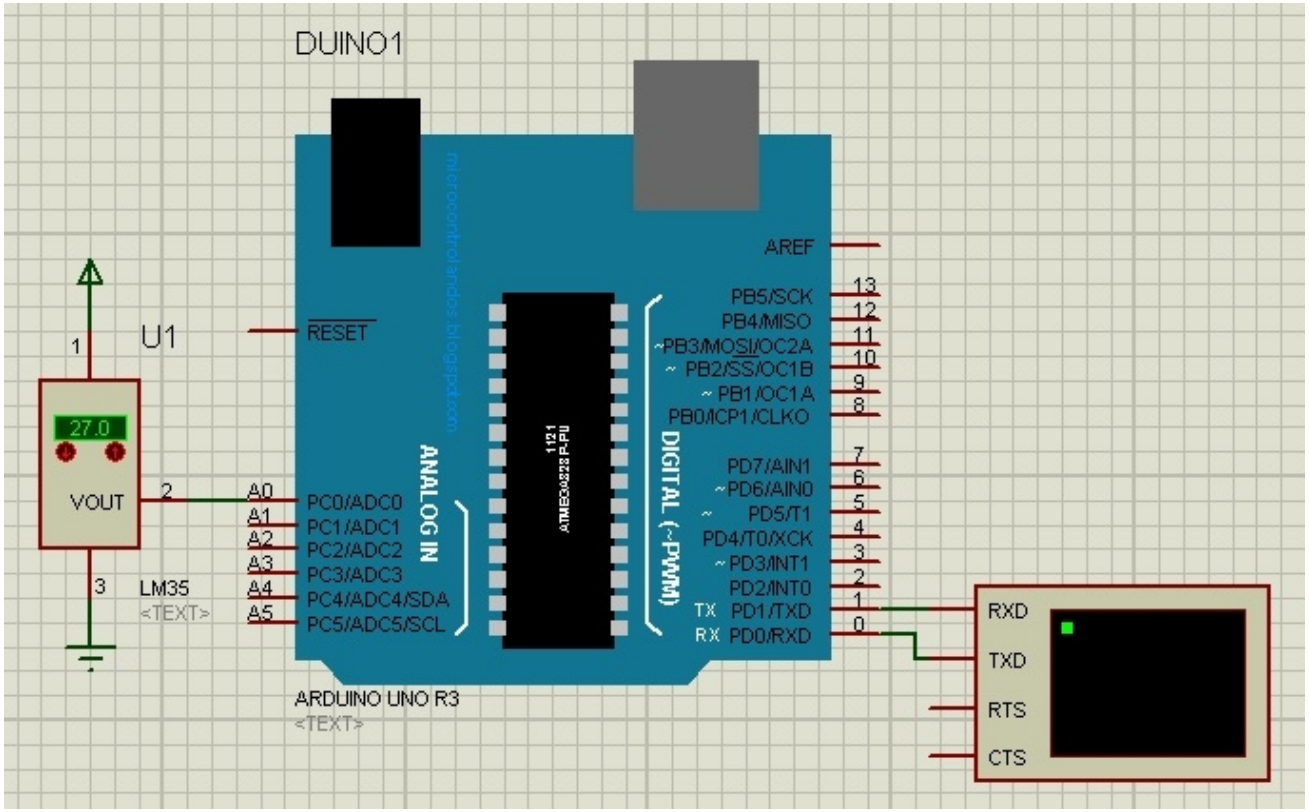
Aplicações

- 2º Exemplo – Sensor de Temperatura
- Objetivo: Aprender a utilizar canais AD do arduino, e comunicação serial.

- Funcionamento: O arduino lê constantemente um sensor de temperatura LM35, e fica enviando o valor da temperatura em graus celsius para o computador via serial.

Aplicações

- Circuito



Aplicações

- Programa

```
void setup()
{
  Serial.begin(9600);
}

void loop()
{
  float temperatura;
  int sensor = analogRead(A0);
  temperatura = (sensor * 500.0)/1024.0;
  Serial.print("Temperatura = ");
  Serial.println(temperatura);
  delay(10);
}
```

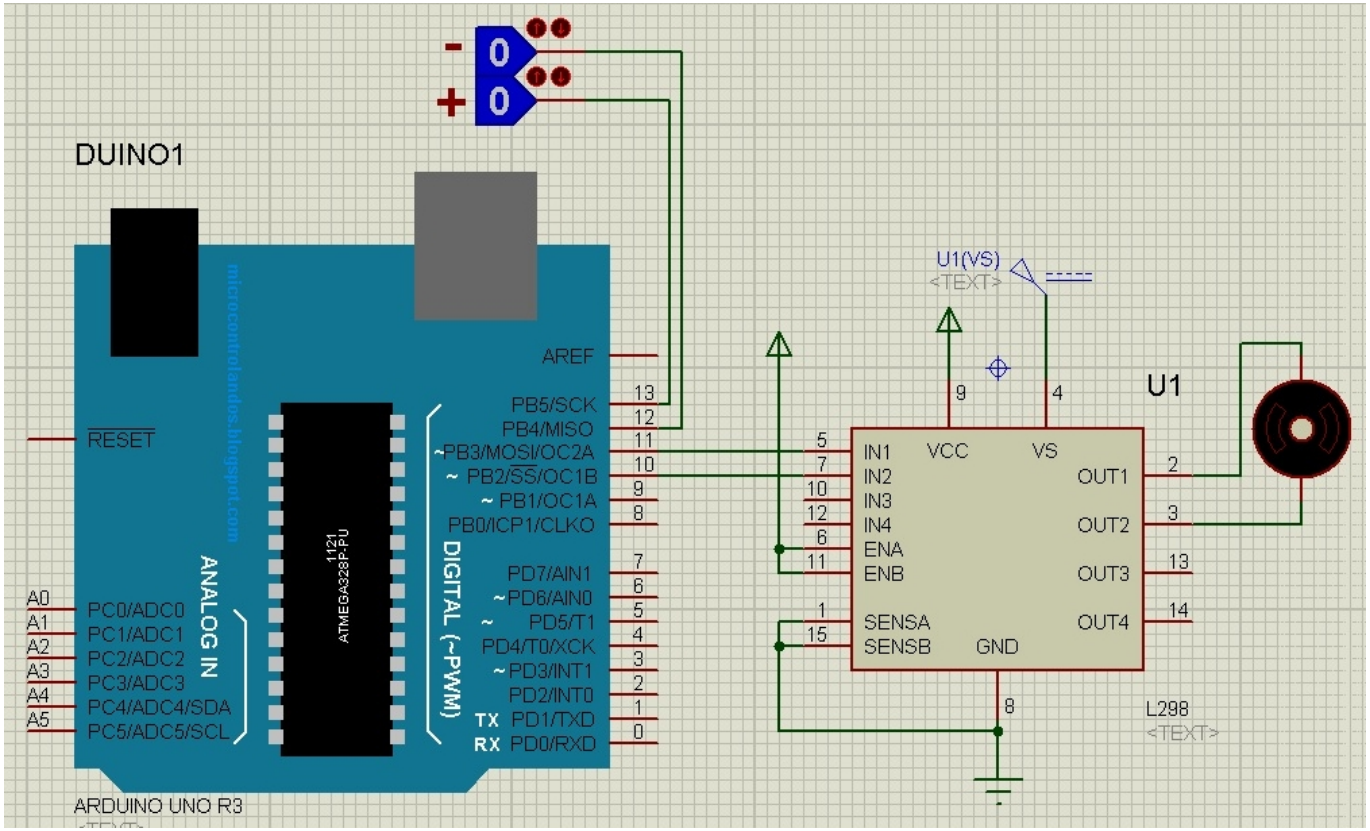
Aplicações

- 3º Exemplo – Controle de Motor
- Objetivo: Aprender o funcionamento do PWM, e também da leitura de botões.

- Funcionamento: Dois botões controlam a velocidade de um motor DC, sendo um botão pra aumentar a velocidade e o outro para diminuir.

Aplicações

- Circuito



Aplicações

- Programa

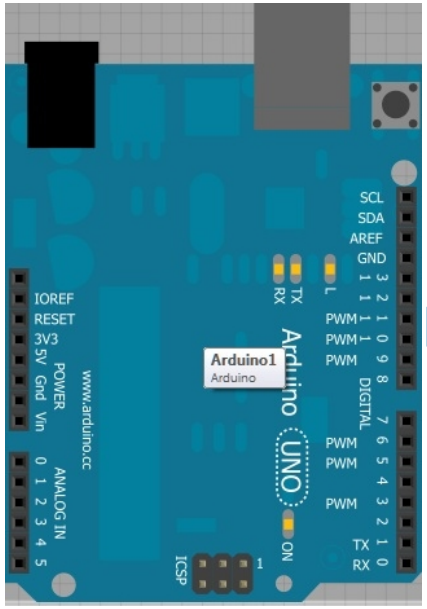
```
int aumenta_velocidade;
int diminui_velocidade;
int velocidade=0;

void setup()
{
  pinMode(13, INPUT);
  pinMode(12, INPUT);
  pinMode(11, OUTPUT);
  pinMode(10, OUTPUT);
}

void loop()
{
  aumenta_velocidade = digitalRead(13);
  diminui_velocidade = digitalRead(12);
 if(aumenta_velocidade)
 {
    velocidade++;
    delay(5);
  }
  if(diminui_velocidade)
  {
    velocidade--;
    delay(5);
  }
    analogWrite(11, velocidade);
    digitalWrite(10,LOW);
    delay(3);
    if(velocidade==255)velocidade=0;
}
```

Aplicações

- 4º Exemplo – ESC (Electronic Speed Controler)
- Objetivo: Aprender a manusear as funções pulseIn(), map().
- Funcionamento: O arduino irá monitorar constantemente uma de suas entradas que estará recebendo um sinal de servo pulso, e com isso irá controlar um motor DC.

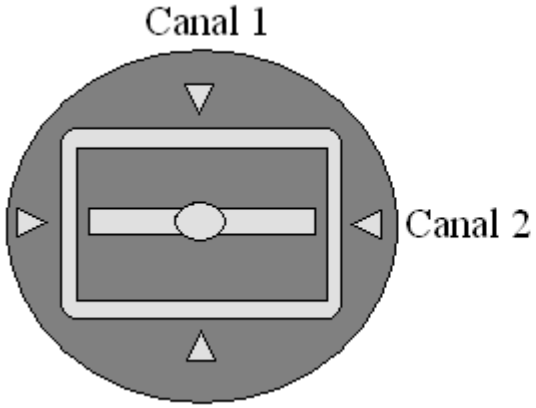
Aplicações

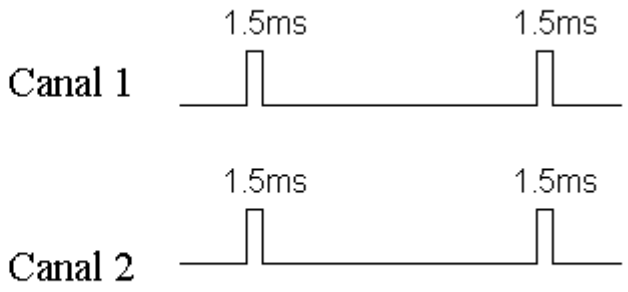
- Esquemático

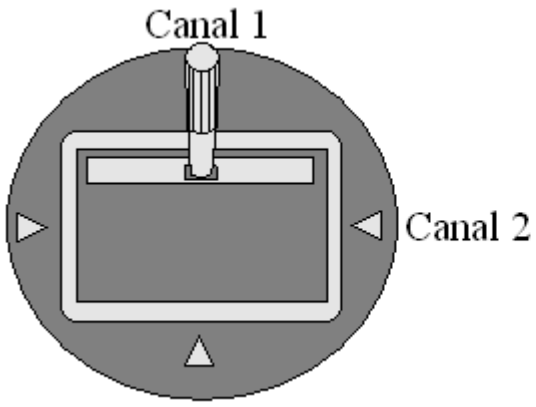


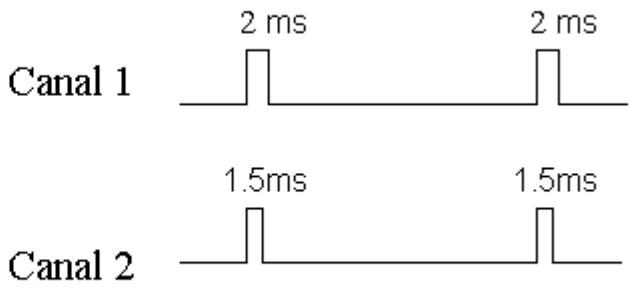
Aplicações

- Sinal de Servo pulso



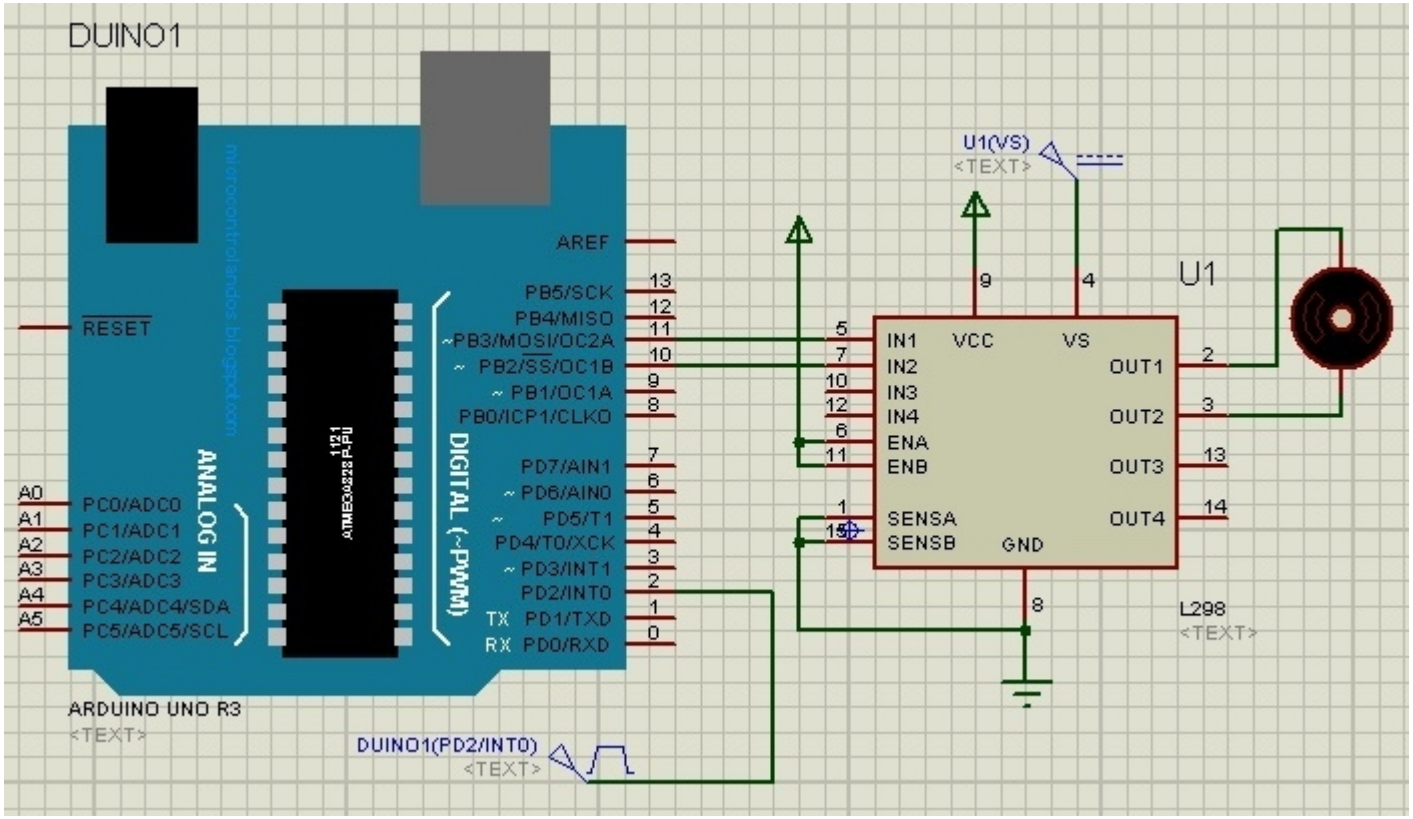






Aplicações

- Circuito



Aplicações

- Programa

```
int canal_a = 2;
int MA1 = 11;
int MA2 = 10;
float T_A = 0.0;
float velocidade;
int sentidoA;

void setup()
{
  Serial.begin(9600);
  pinMode(canal_a, INPUT);
  pinMode(MA2, OUTPUT);
  pinMode(MA1, OUTPUT);
}

void loop()
{
  T_A = pulseIn(2, HIGH);
  T_A = (T_A/1000.0) - 1.5;

  if(T_A<0.0)sentidoA=0;
  if(T_A>0.0)sentidoA=1;
  if(T_A==0.0)sentidoA=2;
  if(sentidoA==2)
  {
    digitalWrite(MA1,LOW);
    digitalWrite(MA2,LOW);
  }
  if(sentidoA==1)
  {
    velocidade = T_A * 510.0;
    analogWrite(MA1, velocidade);
    digitalWrite(MA2, LOW);
  }
  if(sentidoA==0)
  {
    velocidade =-(T_A * 510.0);
    analogWrite(MA2, velocidade);
    digitalWrite(MA1, LOW);
  }
  Serial.print("Potencia= ");
  int potencia = map(velocidade,0,255,0,100);
  Serial.println(potencia);
}
```