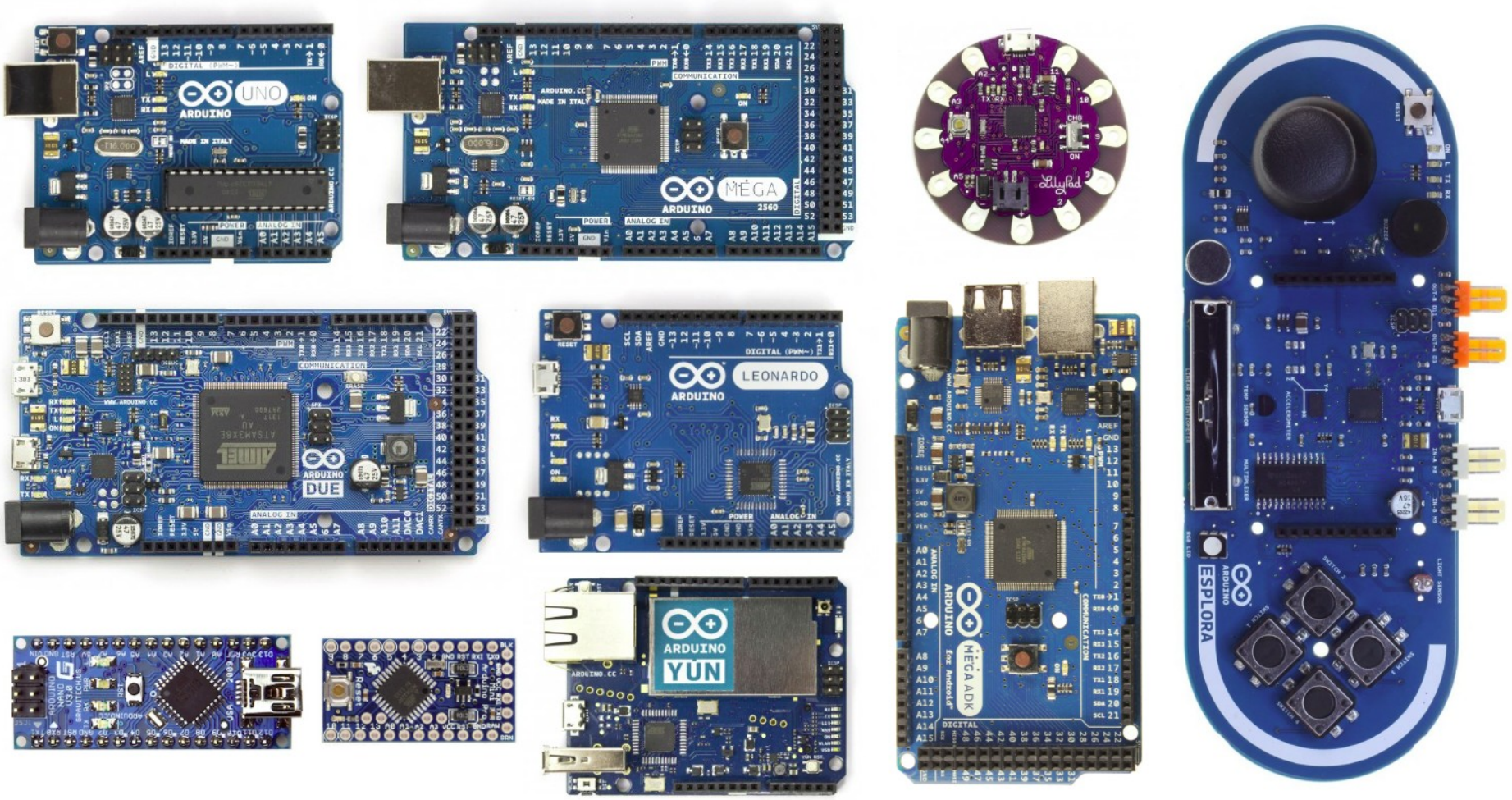
Curso Arduíno – Aula 1



Objetivo do Curso de Arduíno

Capacitar os alunos a utilizarem as ferramentas oferecidas na plataforma Arduíno, bem como os sensores e atuares compatíveis, para ajudá-los na realização de projetos acadêmicos e trabalhos na área da robótica e automação.

Objetivos Específicos do Curso

- Conhecer as especificações do Arduíno UNO;
- Aprender sobre os conceitos elétricos que evolve os sinais de entrada/saída deste dispositivo;
- Aprender os comandos básico em linguagem C e comandos específicos utilizados pelo Arduíno;
- Realizar diversas práticas para que estes comandos sejam empregados ao longo do curso;

Práticas do Curso de Arduíno

- Acionamento de LED, Semáforo, Botões;
- Dispositivo BlueTooth;
- Motor DC, Servomotor e Motor de Passo;
- Sensores de Temperatura, Umidade, Ultrassom e iluminação (LDR);
- Display LCD e display de 7 segmentos;
- Controle de Carro e Braço Robótico.

O que é o Arduíno

É uma plataforma de prototipagem eletrônica de hardware livre e de placa única, projetada com um microcontrolador Atmel com suporte de entrada/saída embutido, uma linguagem de programação padrão, que é essencialmente C/C++.

Fonte: https://pt.wikipedia.org/wiki/Arduino

Tipos de Arduíno

Há vários tipos de Arduíno no mercado:

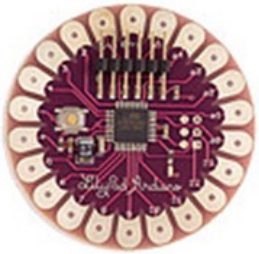
Arduino Uno

Arduino Leonardo

Arduino Ethernet

Arduino Pro

Arduino Mega 2560



Arduino LilyPad

Arduino BT

Arduino Nano

Arduino Mega ADK

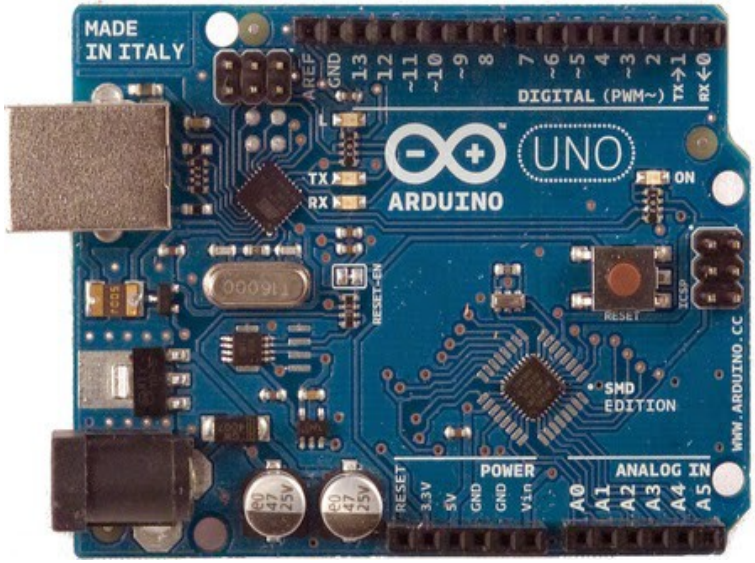
Arduino Fio

USB/Serial Light Adapt

Arduino Mini

Arduíno UNO

O modelo UNO conta com um microcotrolador Atmel Atmega328 de 8 bit, 32kb de memoria flash e 2kb de ram, operando em até 20 Mhz, possui também 14 pinos digitais de 5V estes podem ser de entrada ou saída e 6 entradas analógicas também de 5V.



Definições e Conceitos

Memória RAM: é um tipo de memória volátil que serve para rodar aplicações depois que o Arduíno já está ligado, e cujas informações são perdidas depois do desligamento do mesmo.

Memória Flash: mantém informações armazenadas dentro dela, sem a necessidade de uma fonte de energia, também é conhecida como uma memória não volátil.

Definições e Conceitos

Sinal Digital : é uma sequência discreta no tempo e em amplitude.

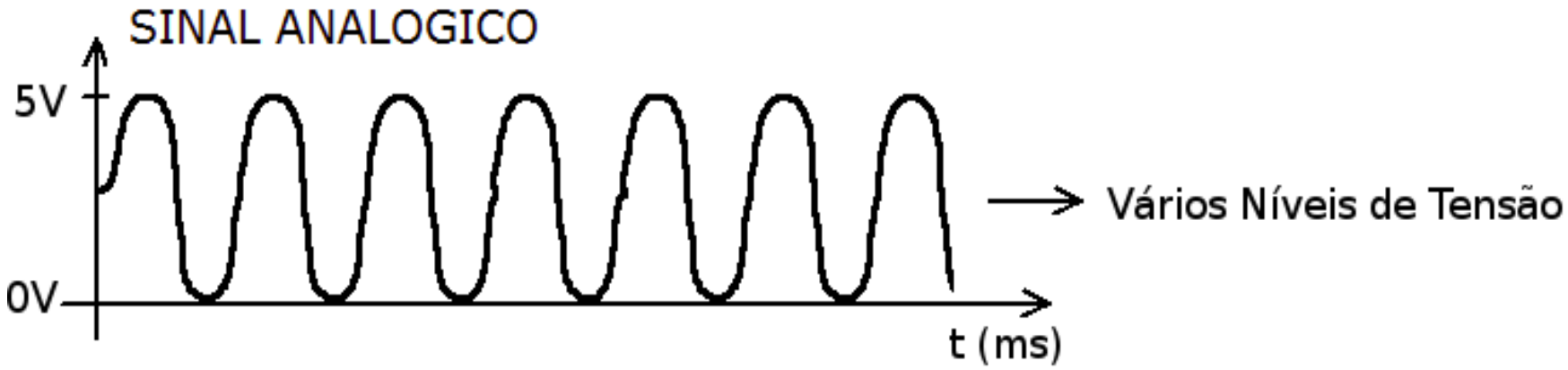
Isso significa que um sinal digital só é definido para determinados instantes de tempo, e que o conjunto de valores que pode assumir é finito.

Fonte:https://pt.wikipedia.org/wiki/Sinal_digital

Definições e Conceitos

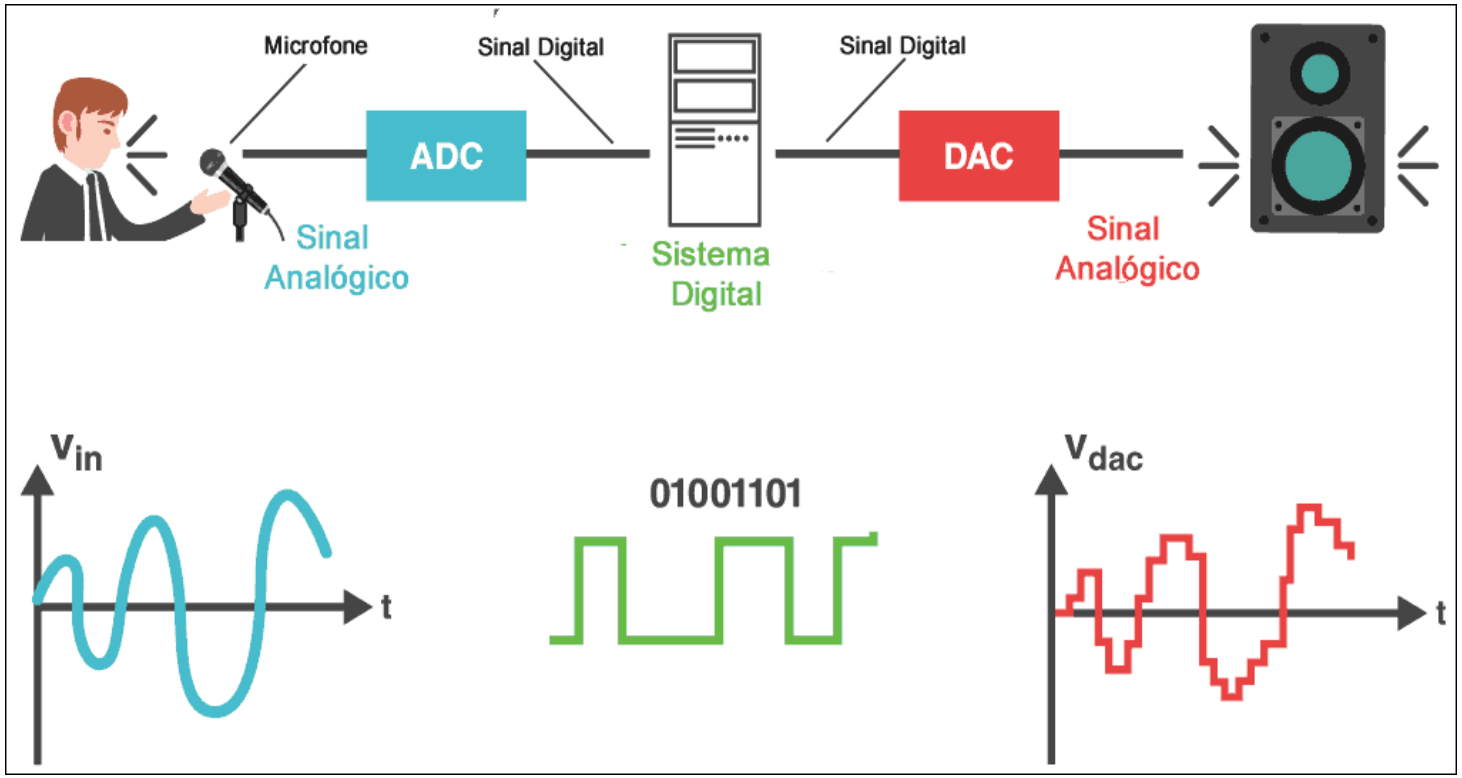
Sinal Analógico: é uma sequência contínua no tempo e em amplitude.

Isso significa que um sinal analógico possui infinitos níveis de amplitude ao longo do tempo, e não possui formato definido, apenas formas de onda conhecidas.



Definições e Conceitos

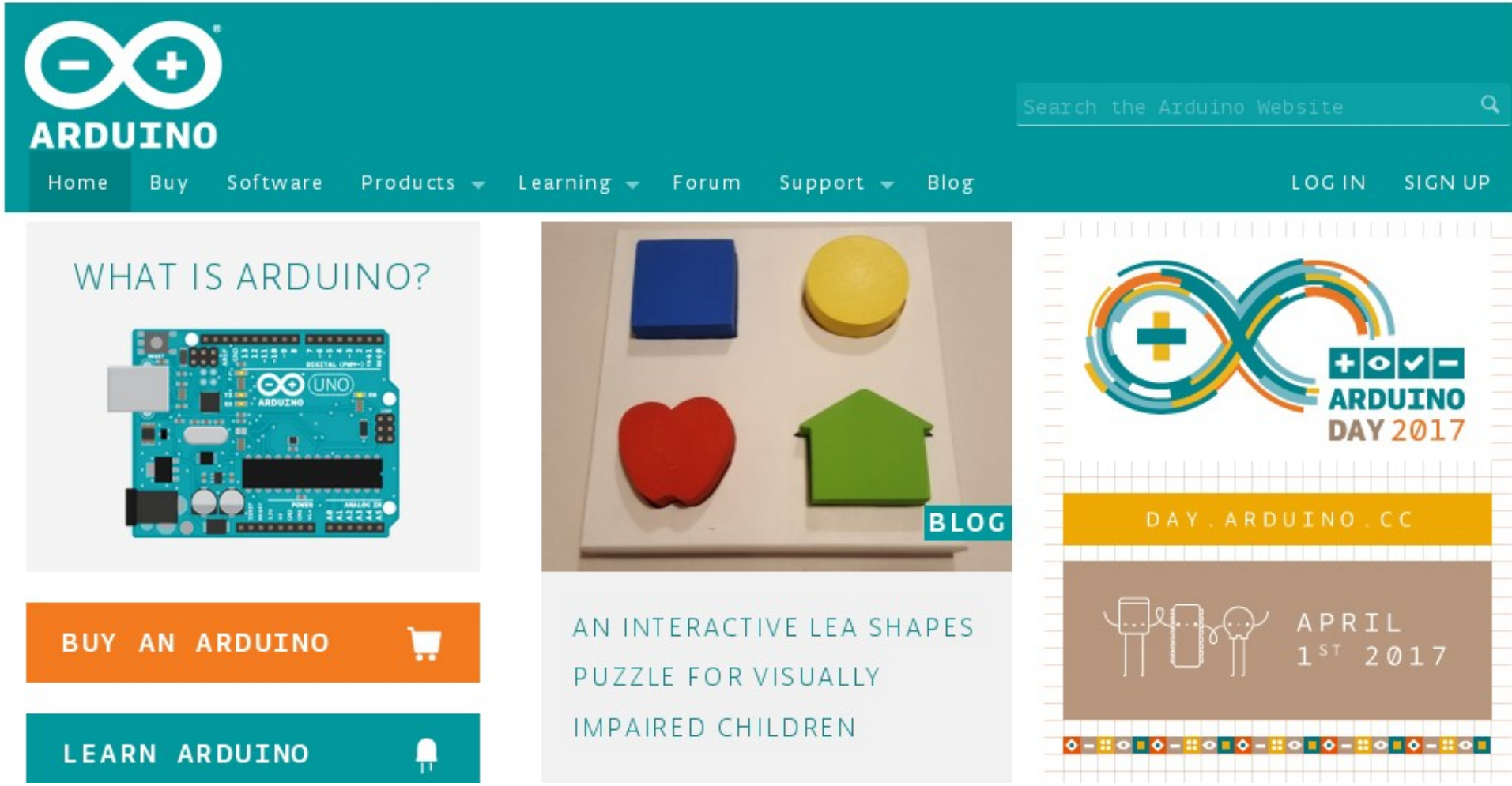
Exemplo de Sinais Analógicos e Digitais em nosso dia a dia.



Fonte: https://www.citisystems.com.br/pwm/

Onde encontrar Informações sobre o Arduíno ?

https://www.arduino.cc/



Comandos em Linguagem C

https://www.arduino.cc/en/Reference/HomePage

Structure

- setup()
- loop()

Control Structures

- if
- if...else
- for
- switch case
- while
- do... while
- break
- continue
- return
- goto

Variables

Constants

- HIGH | LOW
- INPUT | OUTPUT | INPUT_PULLUP
- LED_BUILTIN
- true | false
- integer constants
- floating point constants

Data Types

- void
- boolean
- char
- unsigned char
- byte

Functions

Digital I/O

- pinMode()
- digitalWrite()
- digitalRead()

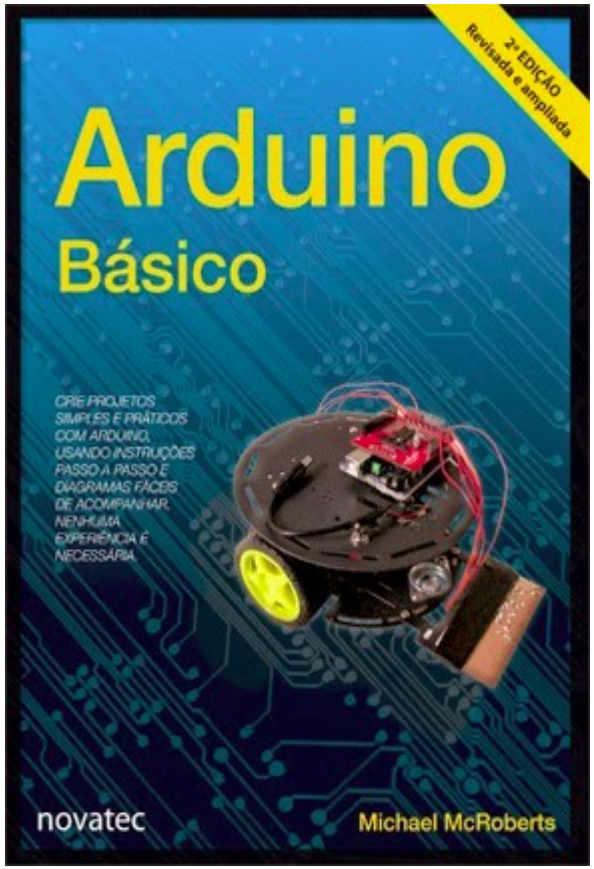
Analog I/O

- analogReference()
- analogRead()
- analogWrite() - *PWM*

Due & Zero only

- analogReadResolution()
- analogWriteResolution()

Onde encontrar mais Informações sobre o Arduíno ?



http://www.arduinoecia.com.br/

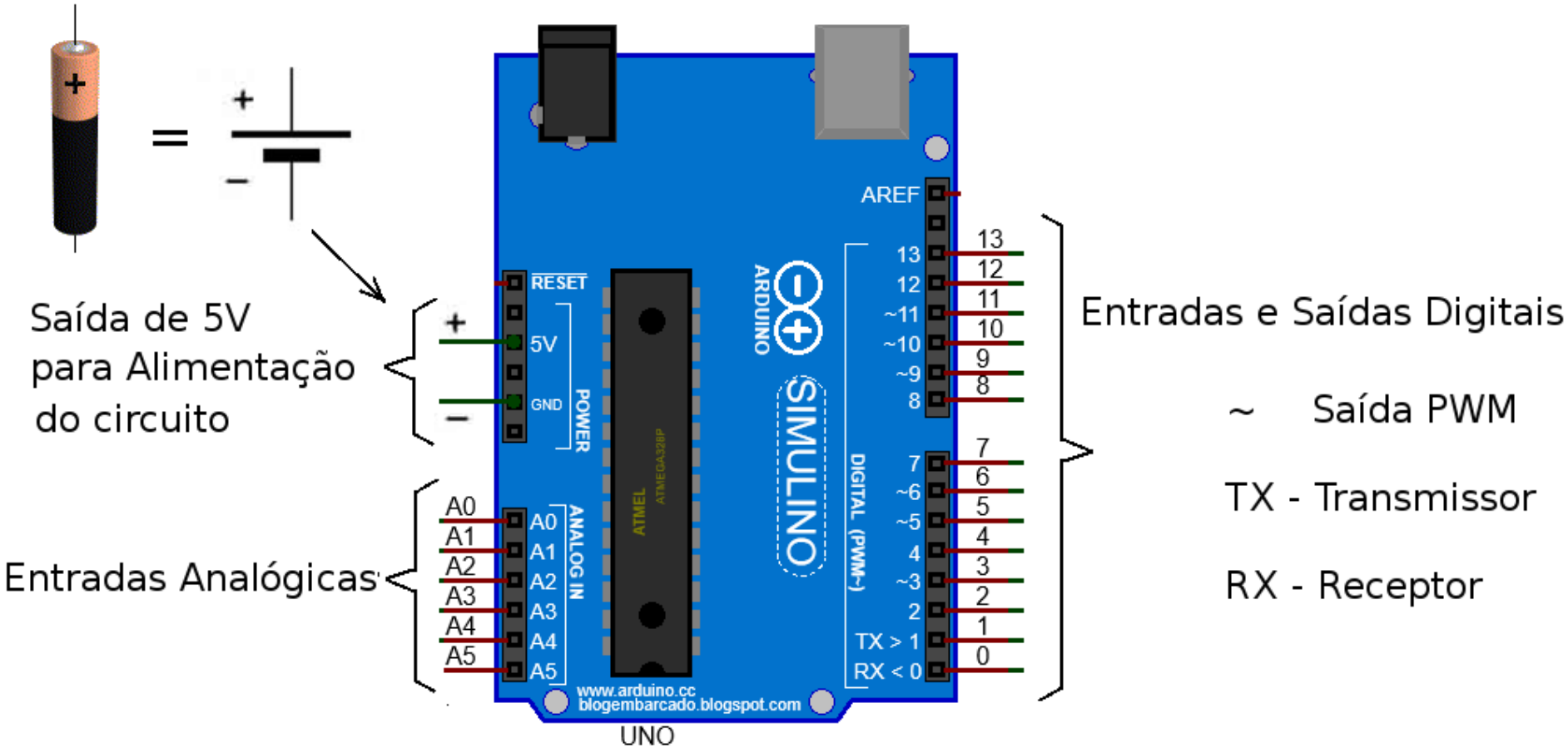
http://blog.filipeflop.com/arduino/o-que-e-arduino.html

https://www.embarcados.com.br/arduino-uno/

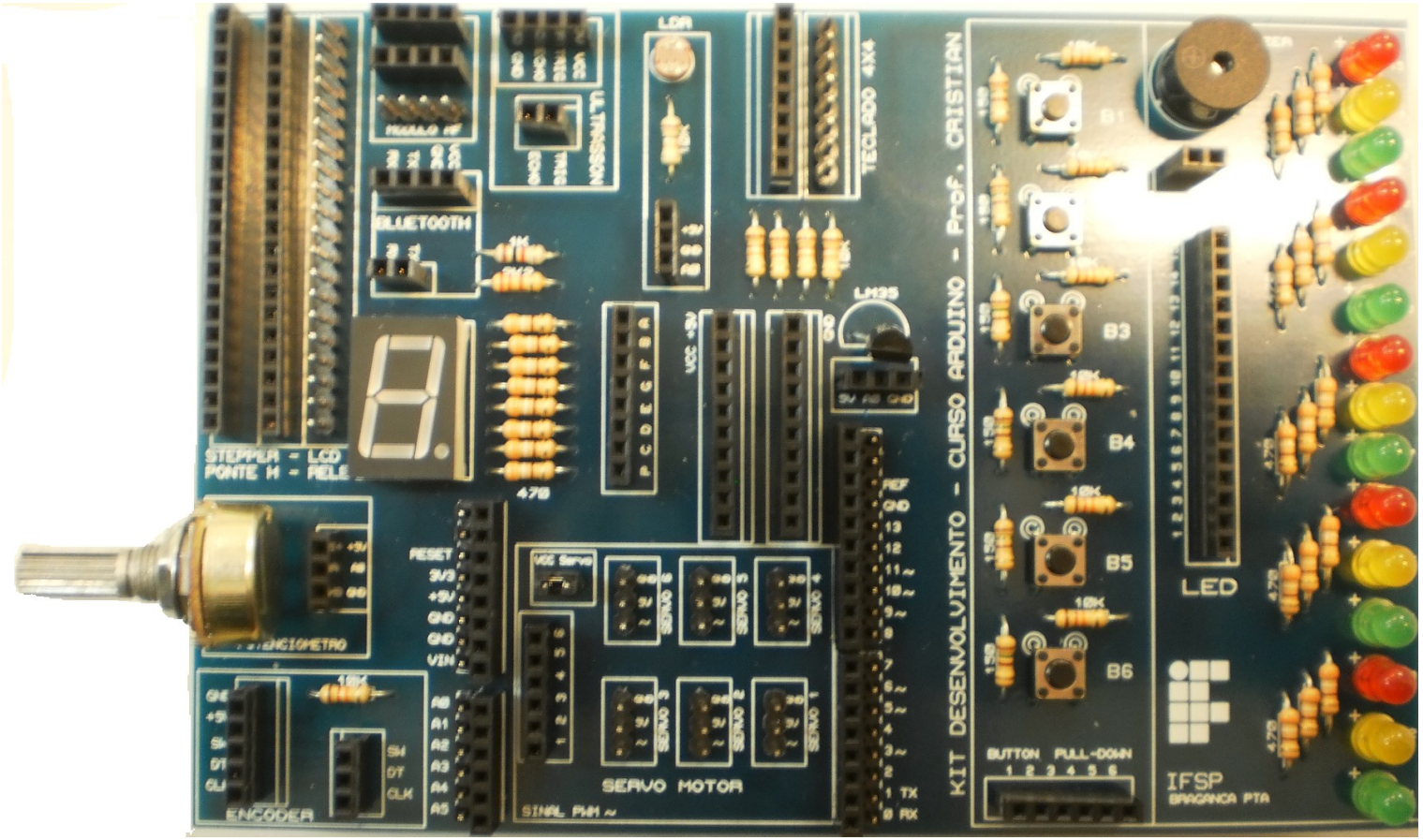
Material do Curso Arduíno

goo.gl/fDGoLk

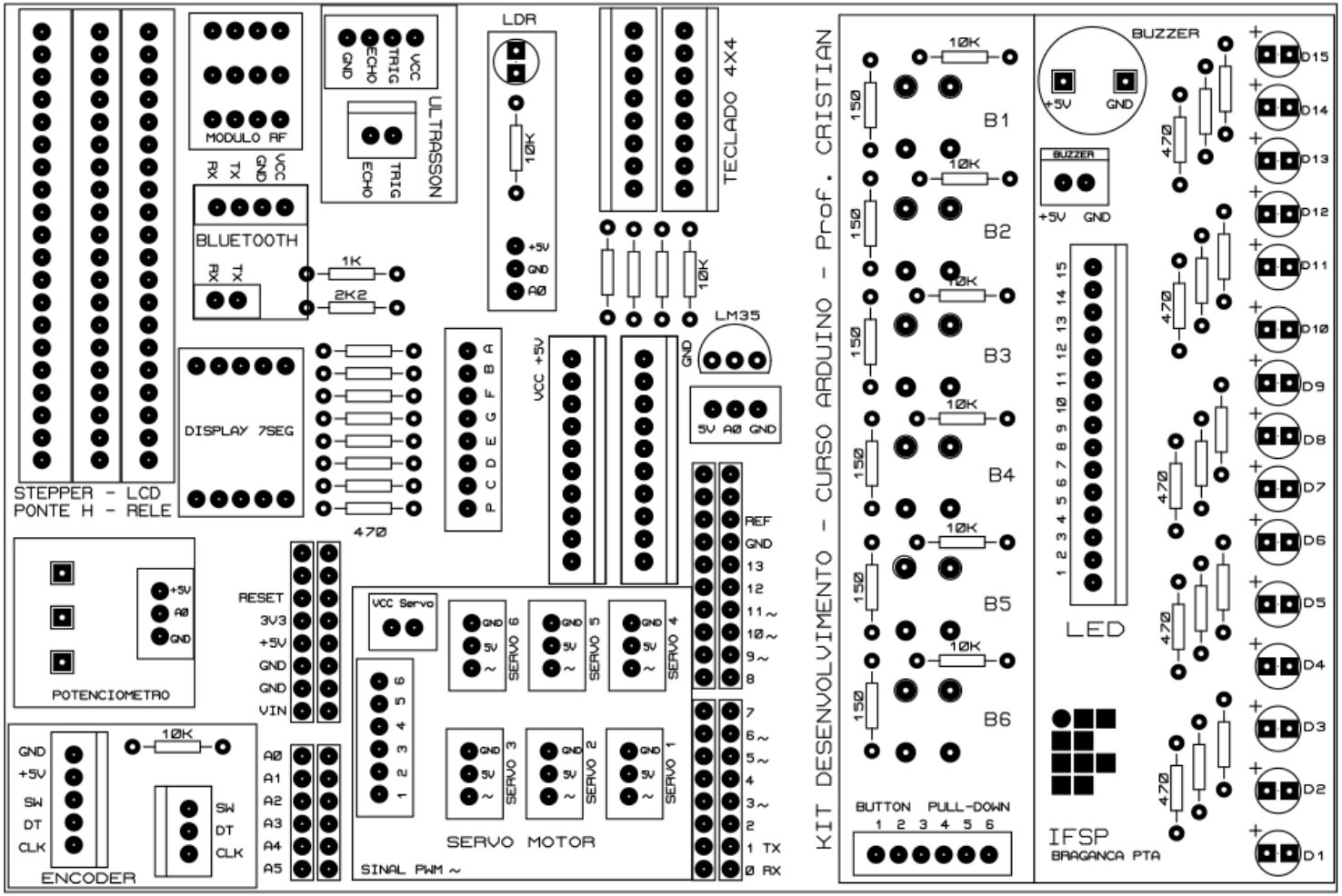
Entradas e Saídas do Arduíno UNO



Placa Desenvolvida para o Curso à ser conectada ao Arduíno UNO

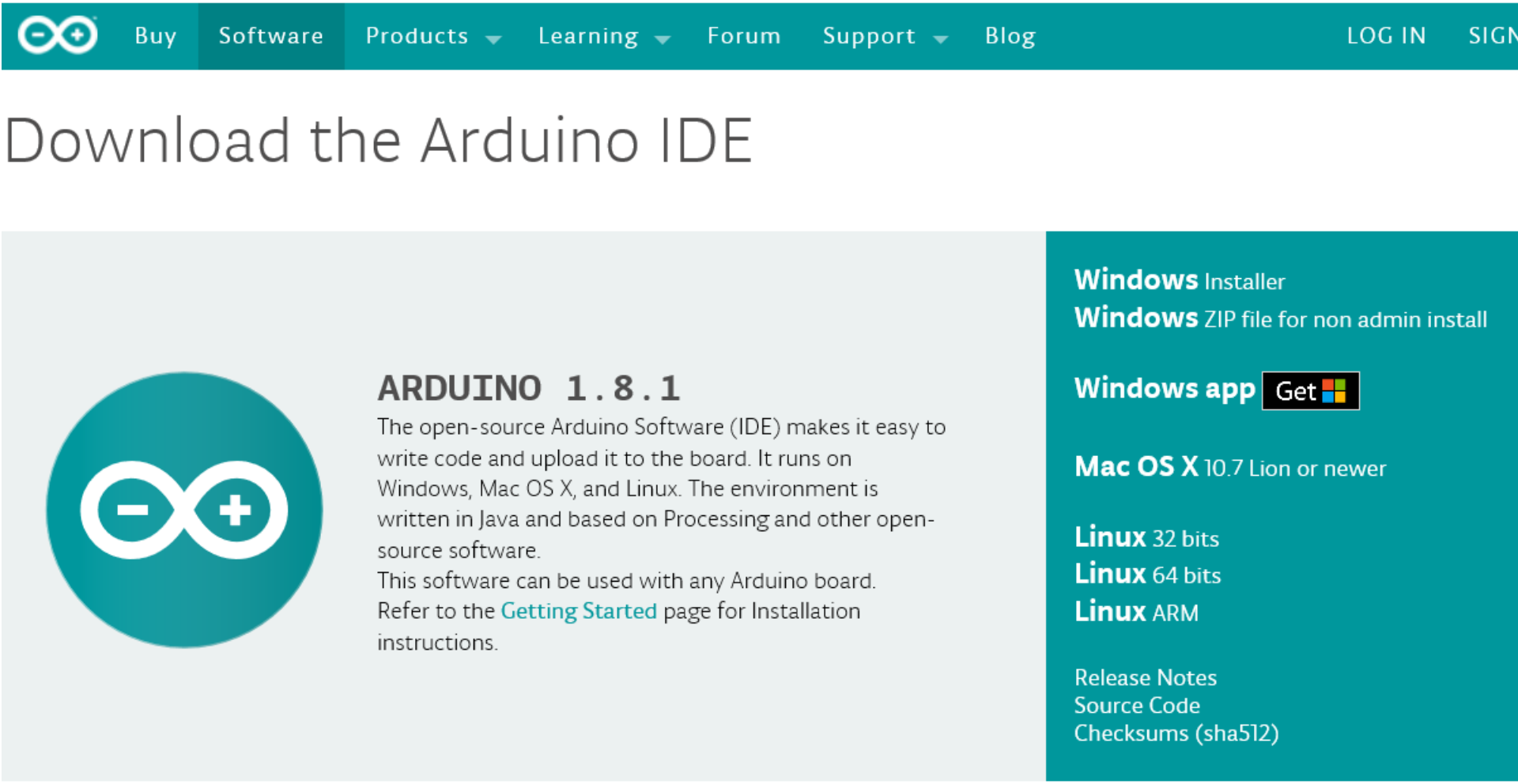


Placa Desenvolvida para o Curso à ser conectada ao Arduíno UNO

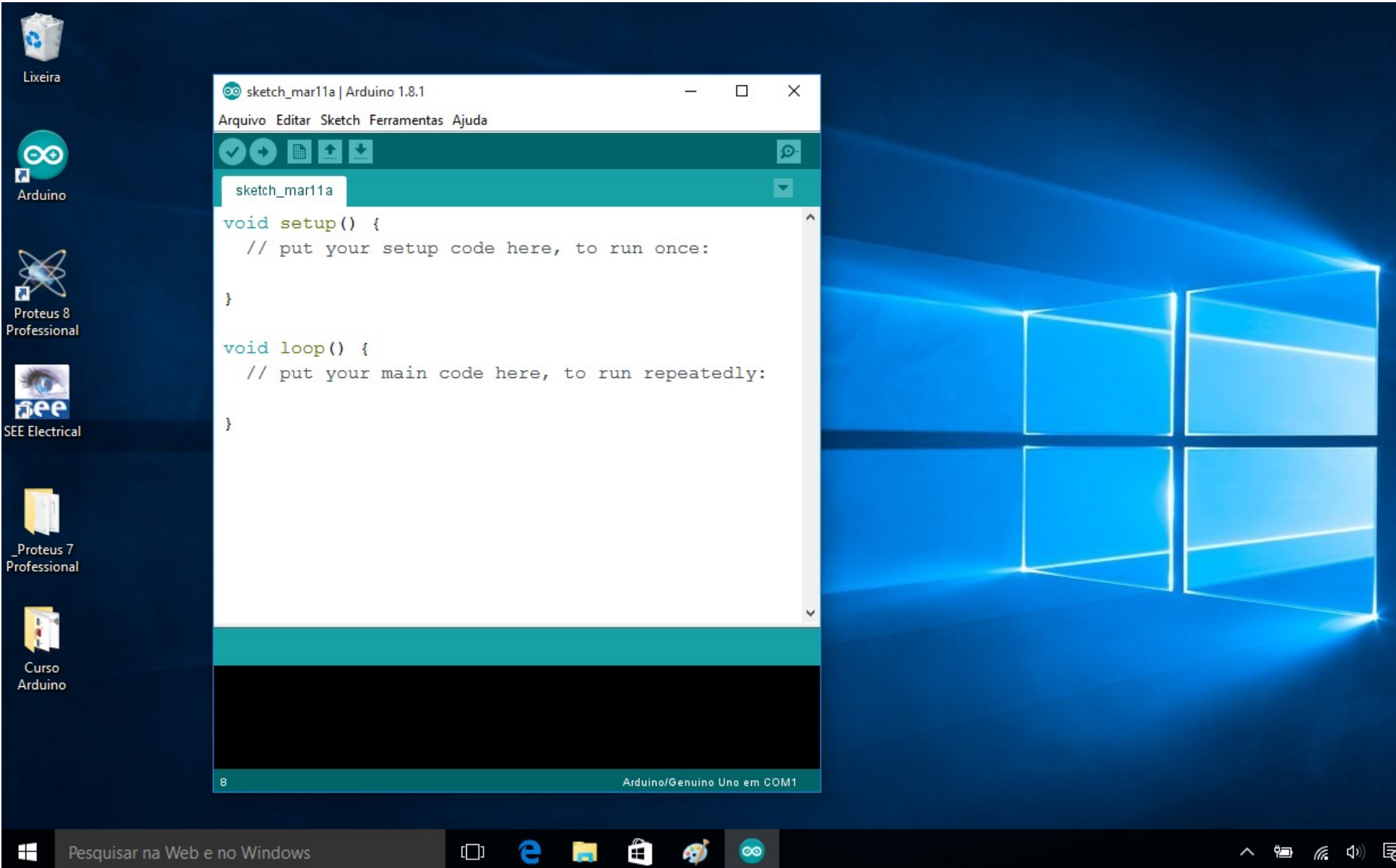


Instalando o Software do Arduíno

https://www.arduino.cc/en/Main/Software



Executando o Software do Arduíno



Estrutura de Programação do Arduíno

Basicamente a estrutura de programação do Arduíno é formada como visto abaixo:

# include < ?????.h >	→ biblioteca desejada;
int ????;	→ declarar variáveis;
void setup ()	→ função de inicialização;
void loop ()	→ loop principal.

Comandos utilizando nas Práticas

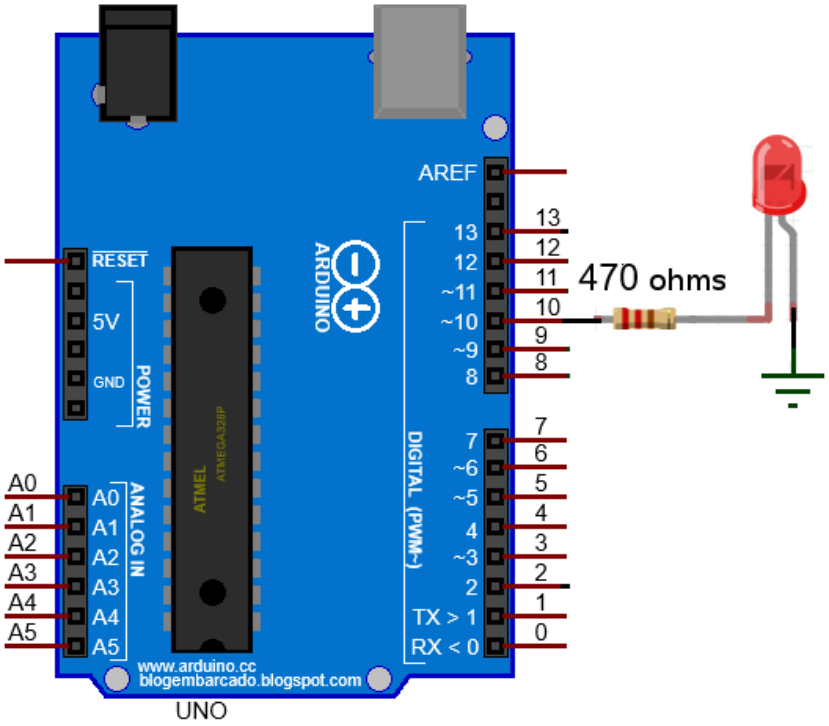
- pinMode()
Configura o pino especificado para se comportar como uma entrada (INPUT) ou uma saída (OUTPUT);

- digitalWrite( )
Escreva um valor HIGH ou LOW para um pino digital.

- digitalRead( )
Lê o valor de um pino digital especificado, HIGH ou LOW.

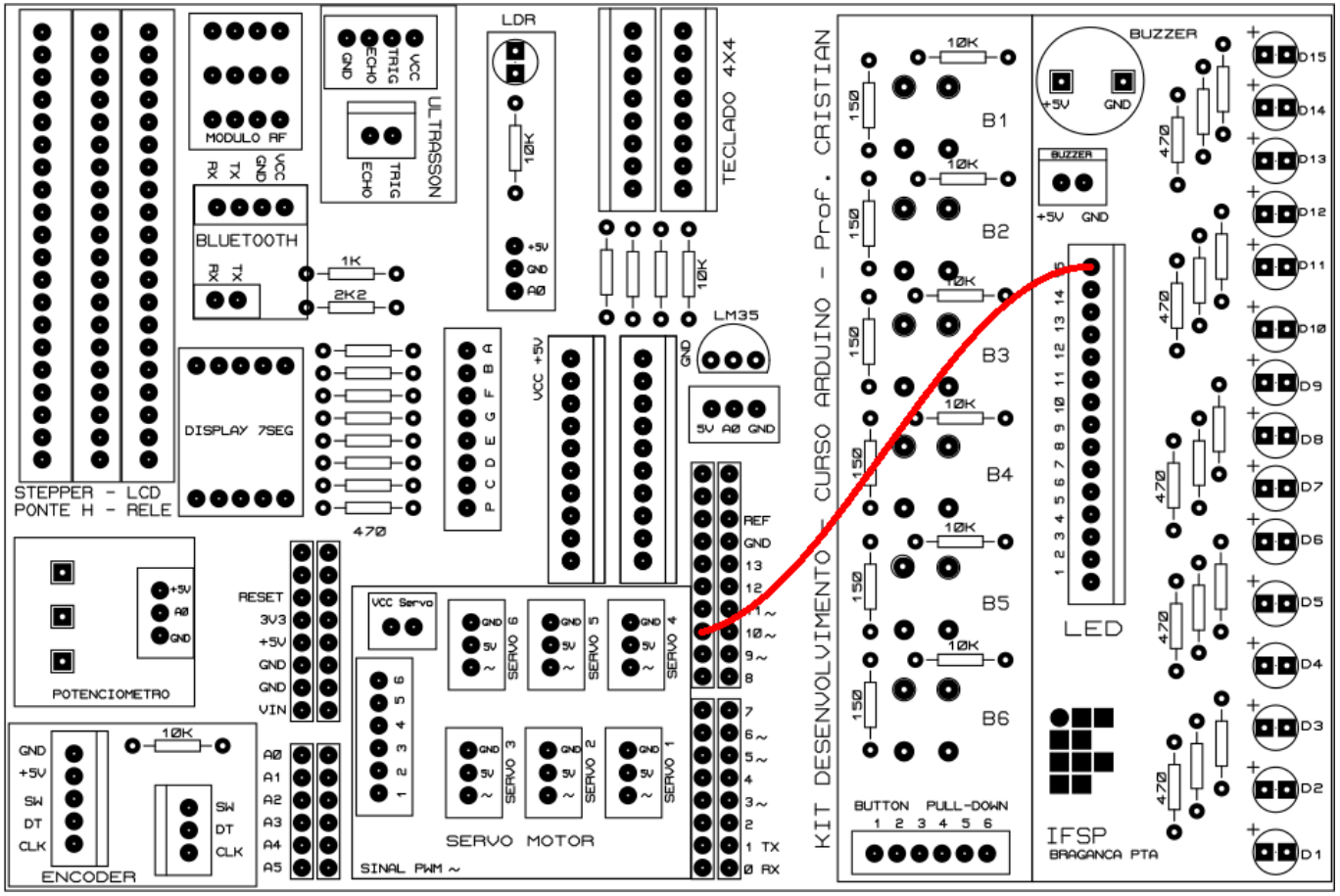
Prática 1 – pisca LED

Ligaremos o pino de saída digital 10, à um dos LEDs da placa. Veja a seguir a ligação.



Prática 1 – pisca LED

A ligação do circuito deve ser realizada como visto a seguir:



Prática 1 – pisca LED

```
// Projeto 1 – LED piscante          → comentário do código

int ledPin = 10;                     → variável de tipo inteiro

void setup() {                       → executada somente uma vez
                                       no início do programa

pinMode(ledPin, OUTPUT);  → definir o pino 10 como saída de
                                       sinal digital

}
```

Prática 1 – pisca LED

void loop() { → executa continuamente enquanto o Arduíno estiver ligado

digitalWrite(ledPin, HIGH); → escreve nível alto na saída do pino 10 (ledPin)

delay(1000); → esperar 1 segundos

digitalWrite(ledPin, LOW); → escreve nível baixo na saída do pino 10 (ledPin)

delay(1000); } → esperar 1 segundos

→ Ao final do programa, ele retorno ao início e executa novamente

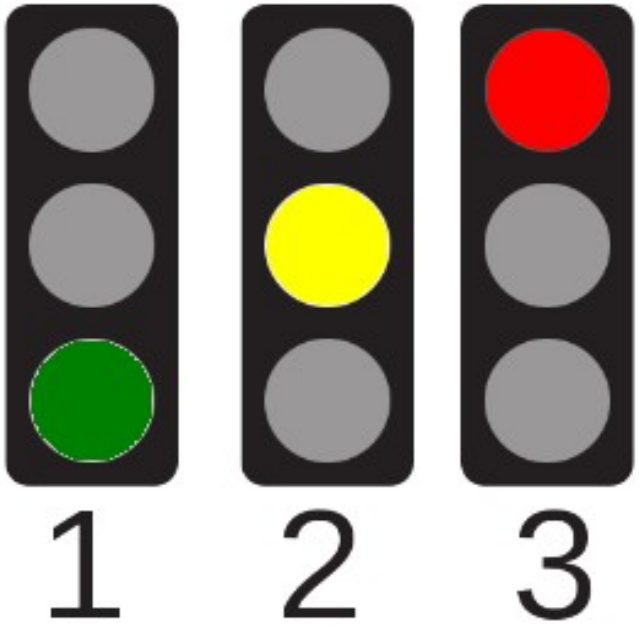
Prática 1 – pisca LED

```
// Projeto 1 –pisca LED

int ledPin = 10;
void setup() {
          pinMode(ledPin, OUTPUT);
}
void loop() {
          digitalWrite(ledPin, HIGH);
          delay(1000);
          digitalWrite(ledPin, LOW);
          delay(1000);
}
```

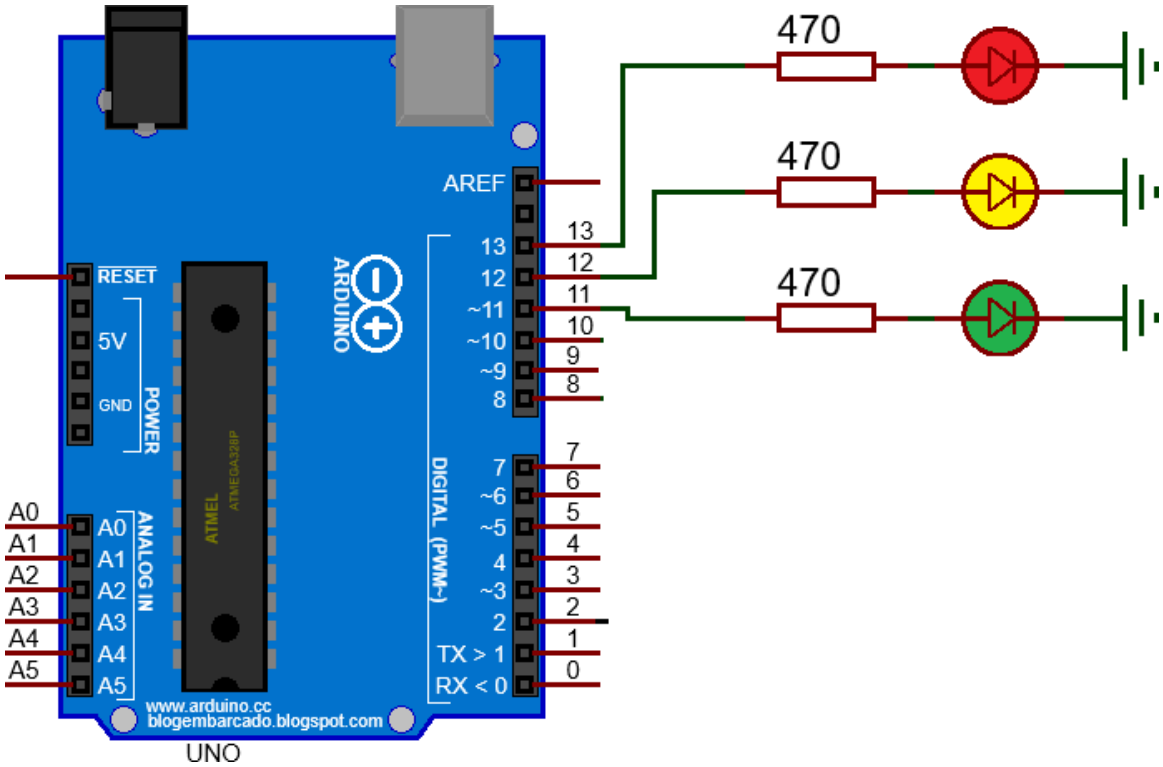
Prática 2 – Semáforo

Crie um semáforo que irá do verde ao vermelho, passando pelo amarelo, e que retornará depois de um intervalo de tempo.



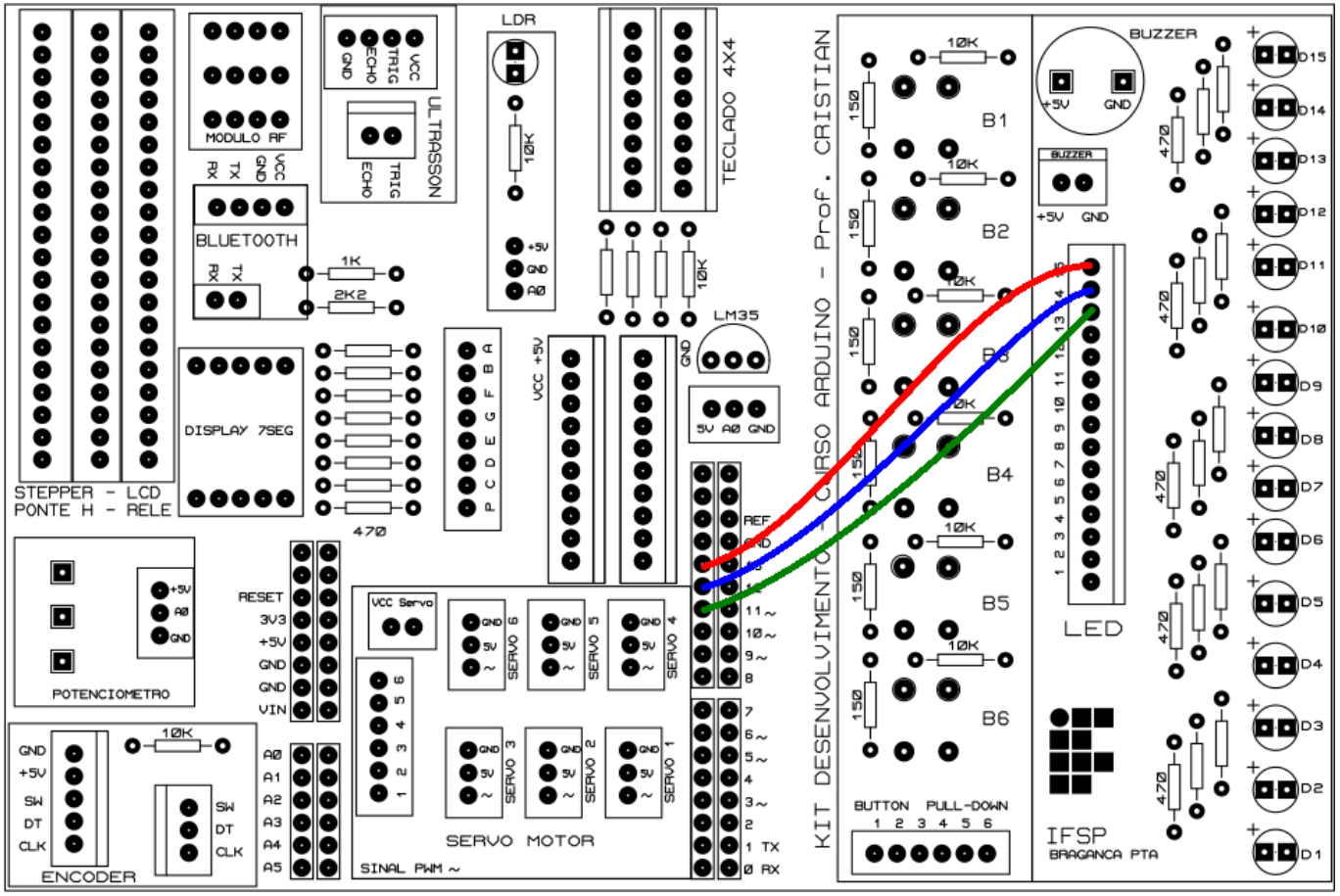
Prática 2 – Semáforo

Ligaremos os pinos da saídas digitais 11, 12 e 13 à LEDs vermelho, amarelo e verde da placa. Veja a seguir a ligação



Prática 2 – Semáforo

A ligação do circuito deve ser realizada como visto a seguir:



Prática 2 – Semáforo

```
int red = 13;      // declara saida 13 digital como  variavel inteira
int yellow = 12;   // declara saida 12 digital como  variavel inteira
int green = 11;    // declara saida 11 digital como  variavel inteira

void setup() {          // laço inicial (executa uma vez)

  pinMode(red, OUTPUT);      // declara como variavel de saida
  pinMode(yellow, OUTPUT);   // declara como variavel de saida
  pinMode(green, OUTPUT);    // declara como variavel de saida
}

void loop() {          // laço de loop (executa continuamente)
  digitalWrite(green, HIGH);   //acende LED verde
  delay(2000);                 //espera 2 segundos
  digitalWrite(green, LOW);    //apaga LED verde
  digitalWrite(yellow, HIGH);  //acende LED amarelo
  delay(2000);                 //espera 2 segundos
  digitalWrite(yellow, LOW);   //apaga LED amarelo
  digitalWrite(red, HIGH);     //acende LED vermelho
  delay(2000);                 //espera 2 segundos
  digitalWrite(red, LOW);      //apaga LED amarelo
}
```

Acionamento por chaves

O acionar de dispositivos a partir do Arduíno, podemos realizado por dois tipos de chaves externas:

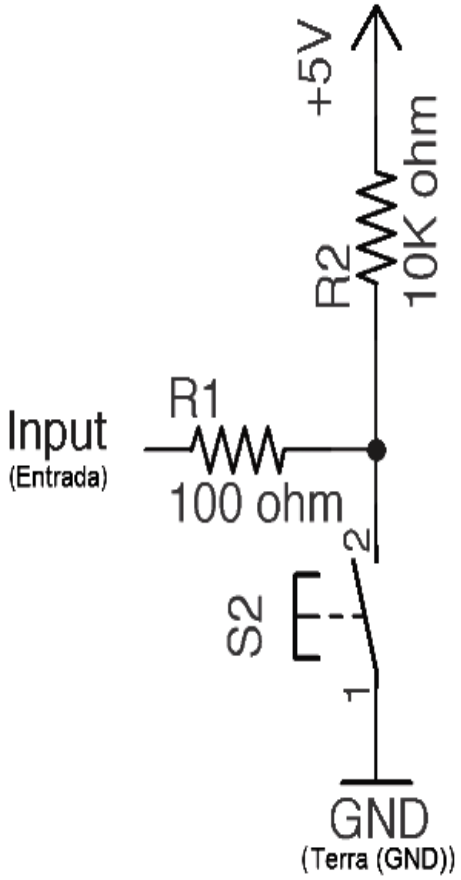


Figura 2.11 – Circuito de resistor pull-up.

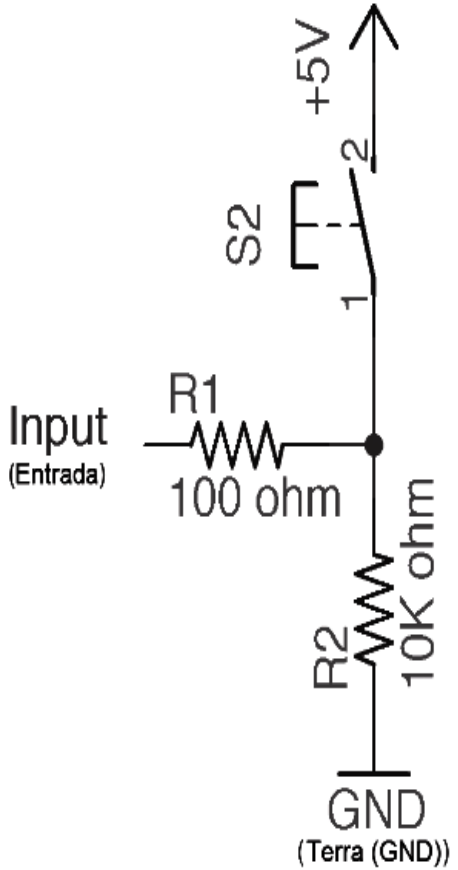
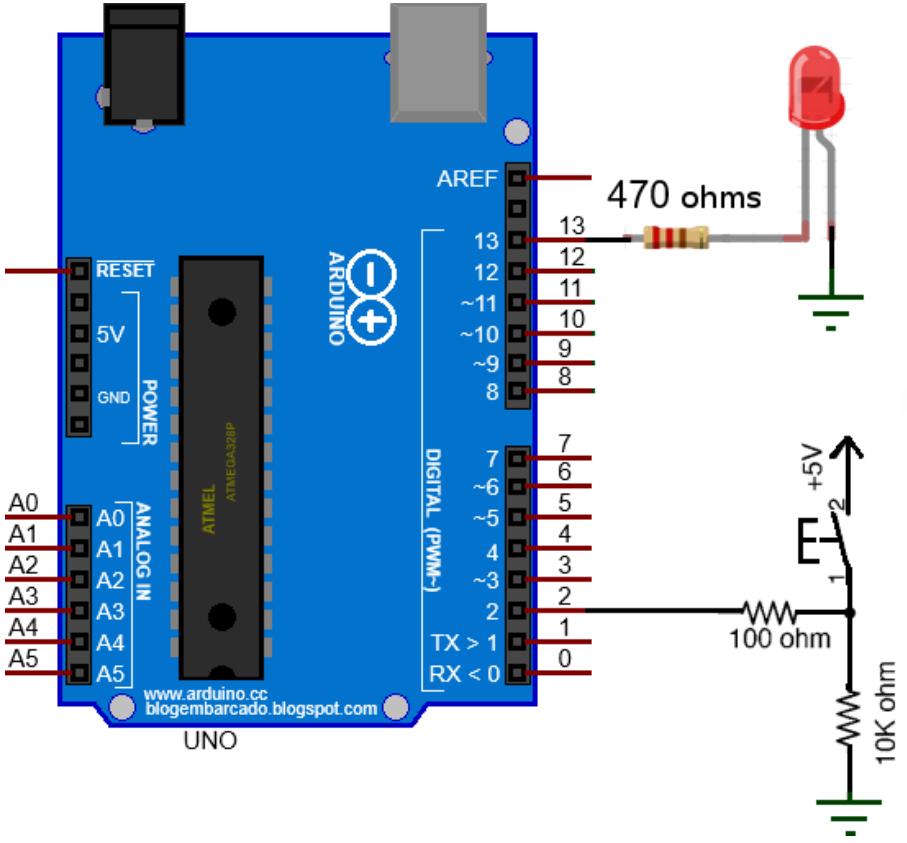


Figura 2.10 – Circuito de resistor pull-down.

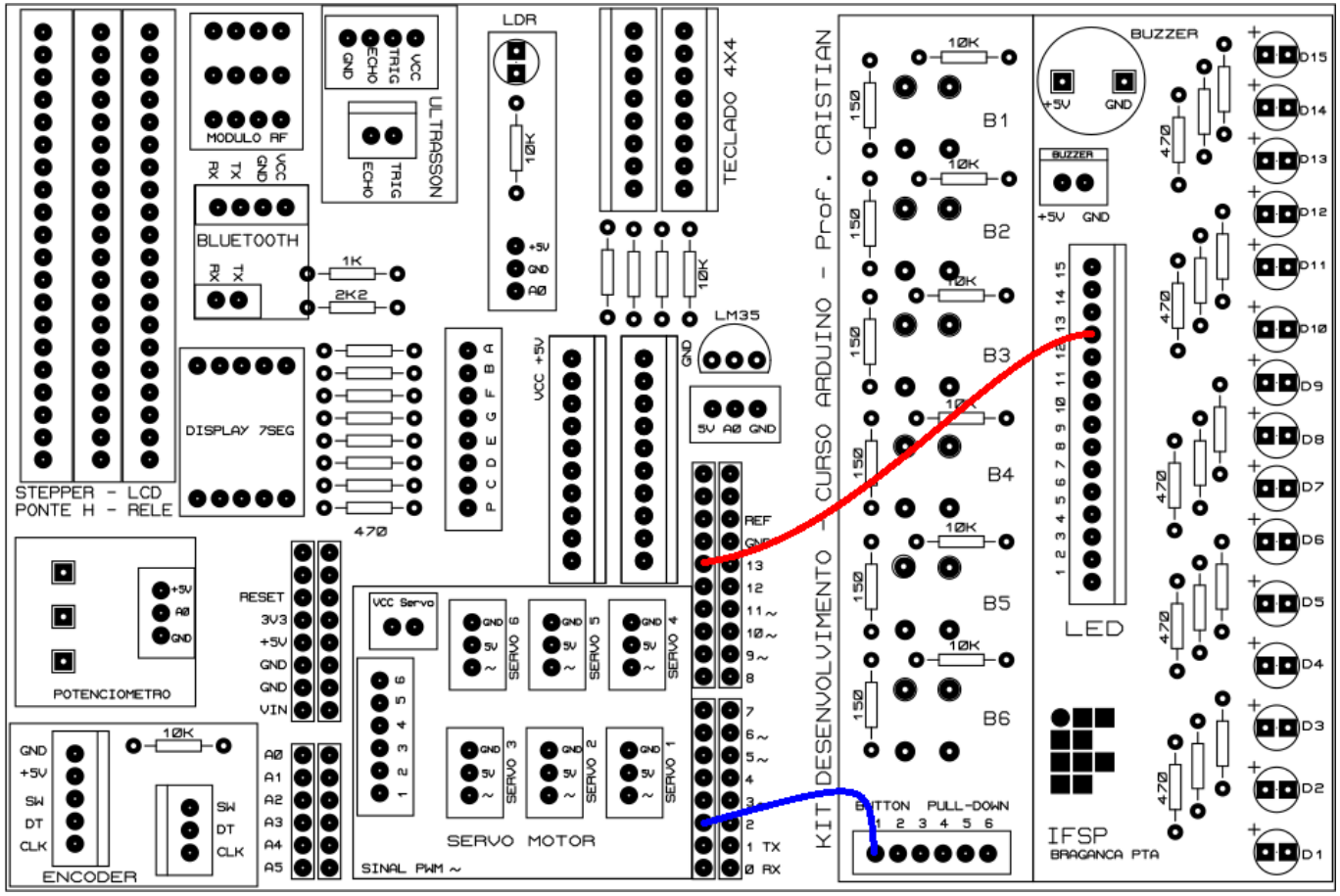
Prática 3 - Acionamento por chaves

Na placa utilizada no curso há seis chaves pull-down, vamos acionar um LED a partir de uma destas chaves, para entendermos como utilizá-las.



Prática 3 - Acionamento por chaves

A ligação do circuito deve ser realizada como visto a seguir:



Comando **if**

Nesta prática utilizaremos o comando **if**, que realiza o teste se uma condição foi atingida ou não.

If (condicional) ==,! =, <,> (Operador de comparação)

x == y (x é igual a y)
x ! = y (x não é igual a y)
x < y (x é menor que y)
x > y (x é maior que y)
x <= y (x é menor que ou igual a y)
x >= y (x é maior ou igual a y)

Prática 3 - Acionamento por chaves

```
int botao = 2;   // atribuiu pino 2 a palavra botao
int led =13;     // atribuiu pino 13 a palavra led
int estado = 0; // atribuiu o valor 0 a palavra estado e declarou
                //  como variavel inteira

void setup(){              // laço inicial (executa uma vez)
  pinMode(led,OUTPUT);     // declara como variavel de saida
  pinMode(botao,INPUT);    // declara como variavel de entrada
}

void loop(){                  // laço de loop (executa continuamente)
  estado=digitalRead(botao);// lê o estado do botao(0 ou 1)e
                            //  armazena na palavra estado

  if(estado == HIGH) {        // Se estado igual a nivel baixo (0) continue;
    digitalWrite(led,HIGH); // acender o LED
  }
  if(estado == LOW){          // Se estado igual a nivel alto (1) continue;
    digitalWrite(led,LOW);  // apagar o LED
  }
}
```

Prática 4 – Semáforo com botão

Crie um programa que quando um pedestre for atravessar a rua, ele tenha que apertar um botão e o semáforo mude de estado, permitindo que o pedestre possa seguir e os carros parem.

Ao lado a sequência das luzes dos semáforos.