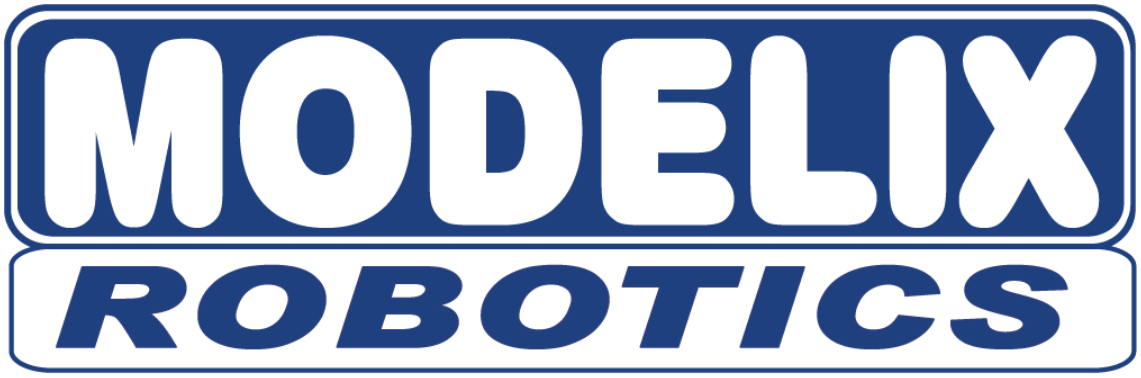



# Aulas Arduino

# Aula Introdutória

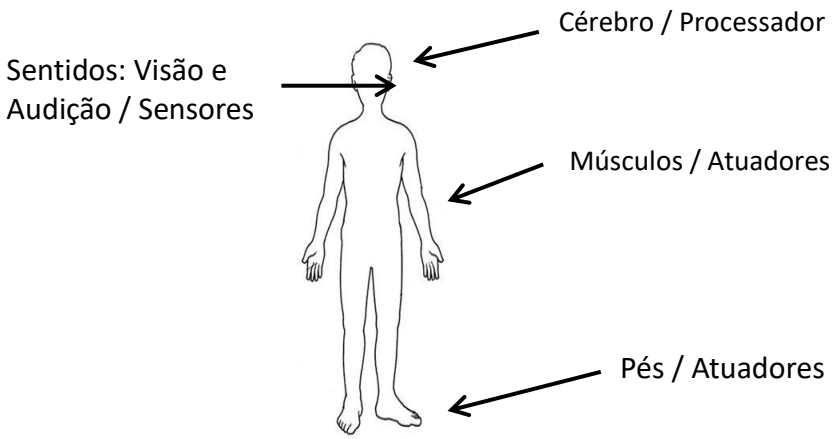
Existe três grandes pilares que constituem a robótica, são eles: **mecânica**, **eletrônica** e **programação**.

**Mecânica:** responsável pela estrutura física do projeto. O corpo do robô ou do projeto propriamente dito.

**Eletrônica:** são os sensores, luzes, motores e as suas ligações com o "cérebro", que é uma placa com circuito do microcontrolador.

**Programação:** O programa como mais conhecido. Este nós gravamos dentro do microcontrolador quantas vezes desejarmos e ele é o responsável pela lógica das funções do projeto.

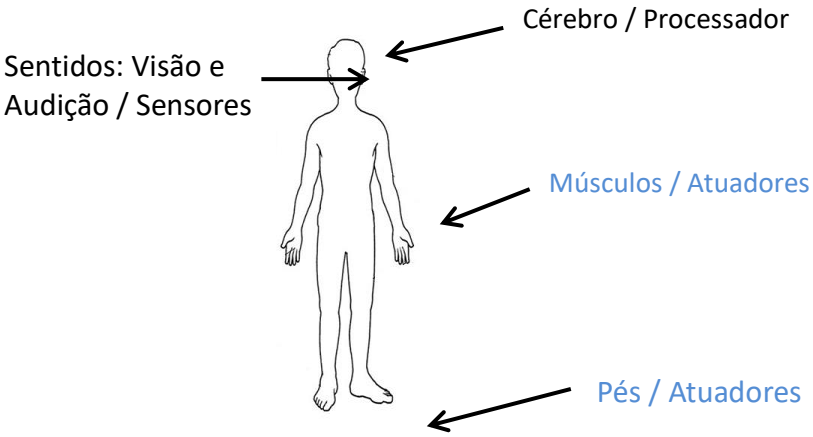
Na robótica o que programamos é o microcontrolador que atua como cérebro do nosso robô. Existe 3 componentes necessários para realizarmos a programação: o microcontrolador (processador), atuadores e sensores. Vamos ver cada um com mais detalhes durante as aulas. Abaixo está uma analogia entre o corpo humano e os componentes da robótica.



# Aula 1 – Led

Antes de começar a montagem, vamos conhecer os componentes. Os componentes são divididos em **Atuadores** e **Sensores.** Nessa aula vamos conhecer os atuadores.

Atuadores também são conhecidos como **saídas**, ou, ***outputs.*** Eles podem ser comparados com nossos pés ou músculos, pois eles são responsáveis por fazer o robô se movimentar (no caso de motores). Também existem outros tipos de atuadores como os luminosos e sonoros que seriam os LEDs e BIPs.




Agora que já conhecemos um pouquinho sobre os componentes, vamos saber mais sobre o Led especificamente.

# LED:

**Funcionalidade:** Emitir luz quando receber energia.

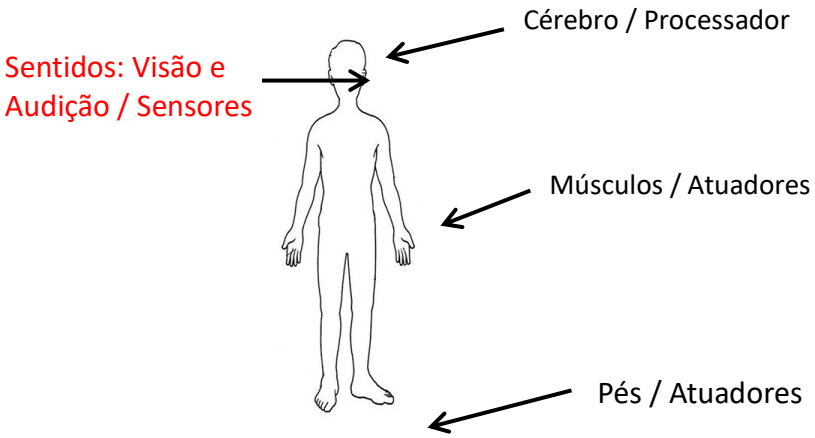
**Quantidade de pinos:** 2 pinos. Possui polaridade, portanto deve – se ligar o pino de menor comprimento (na cabeça do LED existe um chanfrado) no GND. O pino maior é conectado no resistor e desse para o microcontrolador.

**Microcontrolador:** Quando utilizado com o microcontrolador esse componente é configurado como uma saída digital.

## Aula 2 – Led e botão

Agora vamos aprender sobre os sensores.

Sensores também são conhecidos como **entradas**, ou, ***inputs.*** Eles podem ser comparados com nossos sentidos. O sensor de obstáculos tem uma função semelhante ao nosso olho, pois ele consegue captar se existe obstáculo na sua frente e passar essa informação ao microcontrolador para que o robô não venha a colidir com essa barreira. Exemplo de sensores: Botão, sensor de ímã, sensor de temperatura, sensor de luz, sensor de obstáculo.



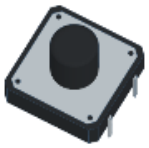

Agora que já conhecemos um pouquinho sobre os componentes, vamos saber mais sobre o PushButton (botão) especificamente.

# PushButton:

**Funcionalidade:** Quando acionado pode impedir ou permitir a passagem de corrente pelo circuito.

**Quantidade de pinos:** 4 pinos.

**Microcontrolador:** Quando utilizado com o microcontrolador esse componente é configurado como entrada digital.



**Objetivo:** Conhecer os componentes chamados de sensores e aprender a programá-los. Quando o botão (pushButton) é acionado o LED é aceso. Para acionar um LED é preciso alimentá-lo. Ele opera com nível de tensão de 3.3 V (Volts). O microcontrolador fornece 5V, então utilizamos um resistor de 300 Ω para evitar que o LED seja danificado.

## Material Utilizado

1x Led (qualquer cor)

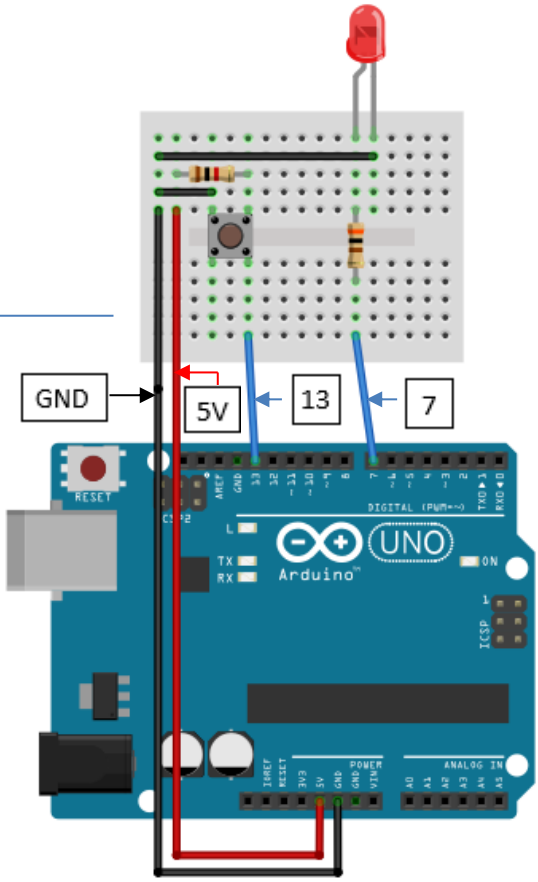
1x PushButton

Fios Conectores

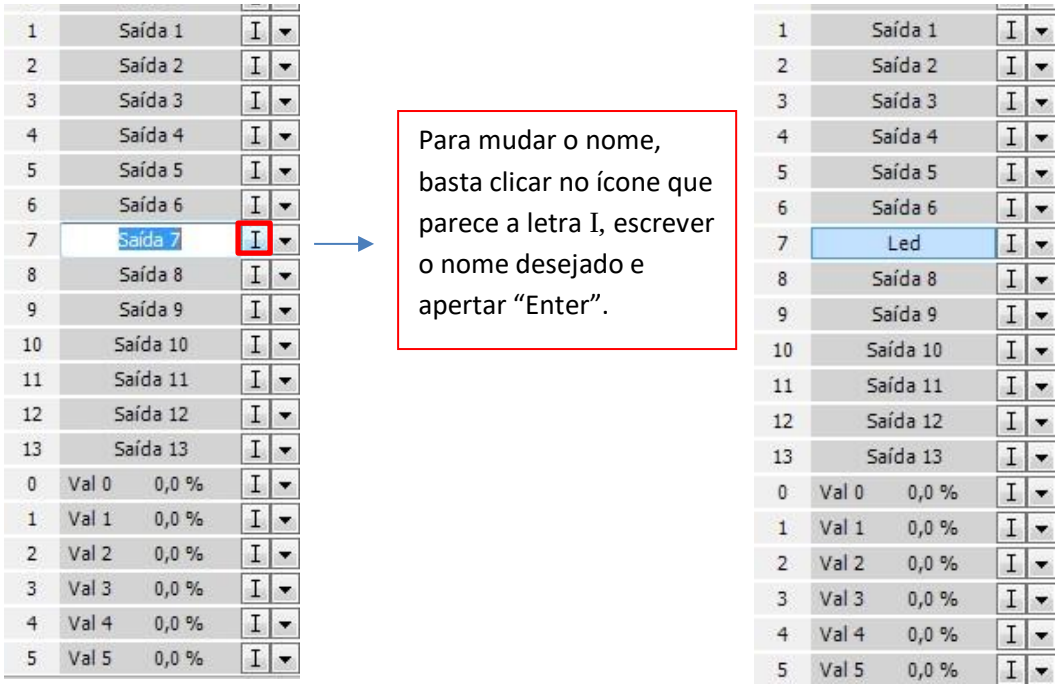
1x Resistor 10kΩ

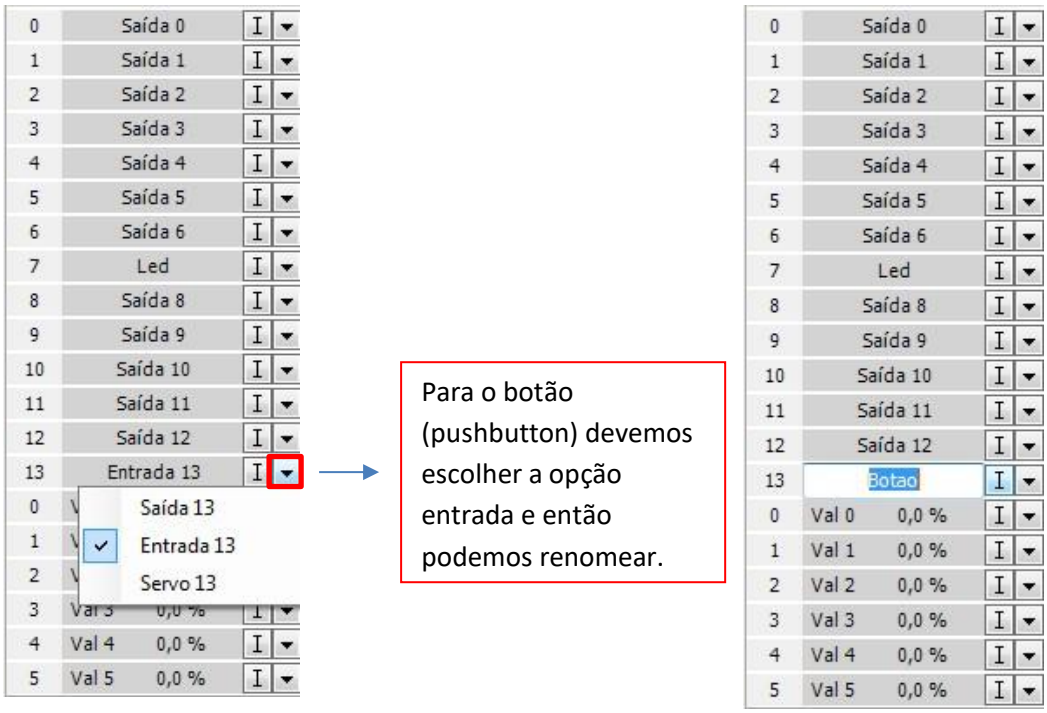
1x Resistor 330Ω

A perna reta do LED mostrada no esquema representa a parte que possui um chanfro na cabeça do mesmo. Essa perna menor deve ser ligada sempre no GND.

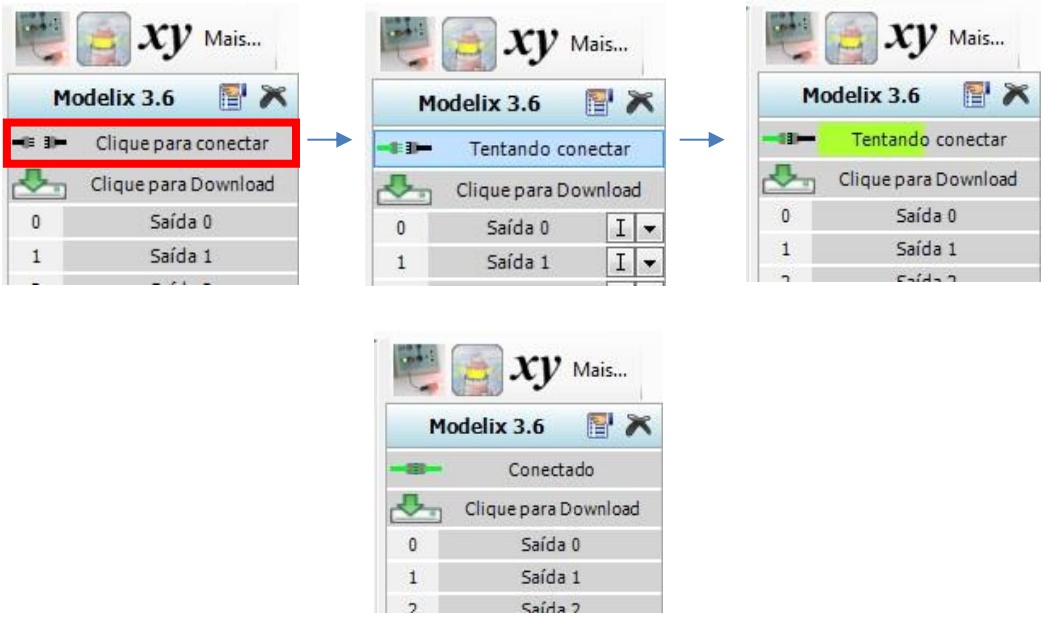


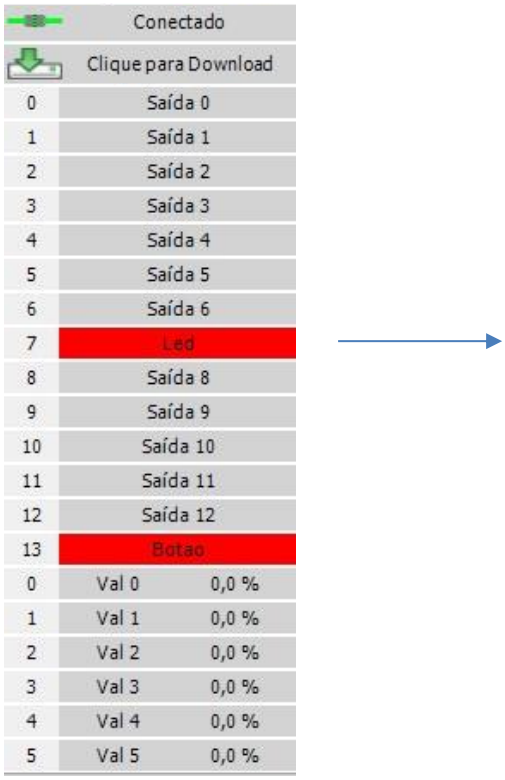
Com o circuito elétrico já montado vamos iniciar as configurações no software. Primeiramente verifique em quais portas do microcontrolador foram conectados o LED e o botão. Nesse exercício o LED está na porta 7 e o Botão na porta 13. Com isso definido, podemos mudar os nomes dessas variáveis e configurá-las:



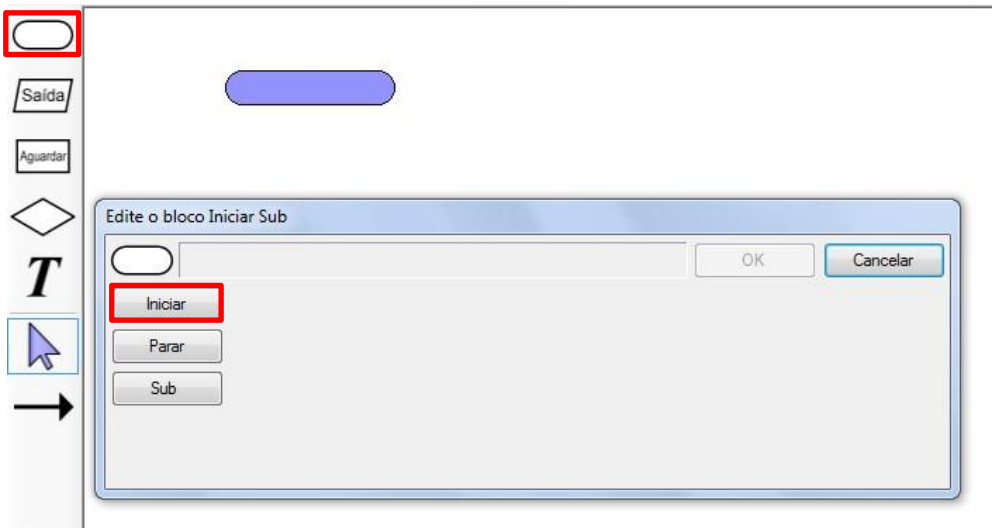
Antes de começarmos a programação podemos verificar se todas as ligações foram feitas corretamente. Para isso conecte o cabo USB no microcontrolador e no computador. No menu lateral direito escolha a opção “Clique para conectar” para iniciar a comunicação entre o software e o microcontrolador.



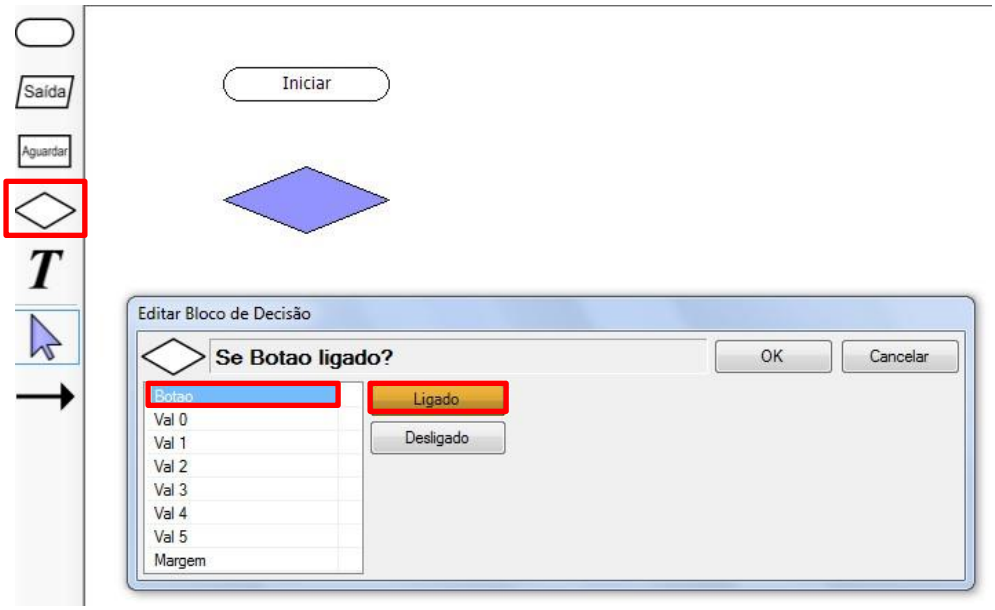



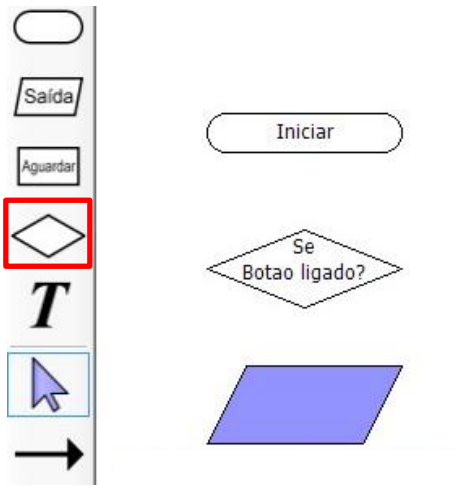

Agora podemos verificar se as ligações estão corretas. Se acionarmos o bloco 7 Led, devemos ver o LED no protoboard ligado. Se acionarmos o botão no circuito, devemos ver o bloco 13 Botão Acionado. Caso não aconteça os eventos esperados verifique se tudo está conectado corretamente e nos pinos corretos. Esse modo de conexão é chamado Modo Conectado.

Após concluir essa configuração e teste podemos iniciar a programação. Para utilizarmos os blocos basta arrastá-los até a tela de programação e escolher a opção “Iniciar”.

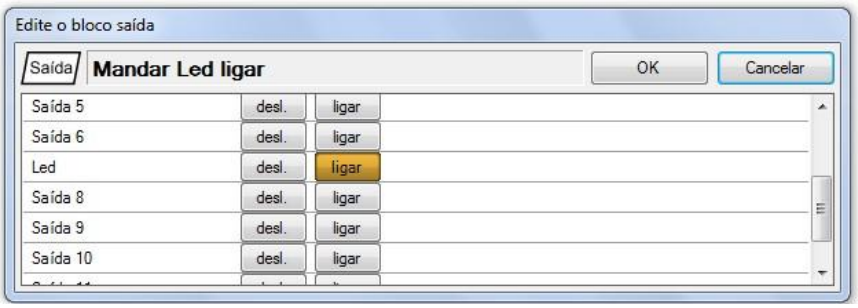


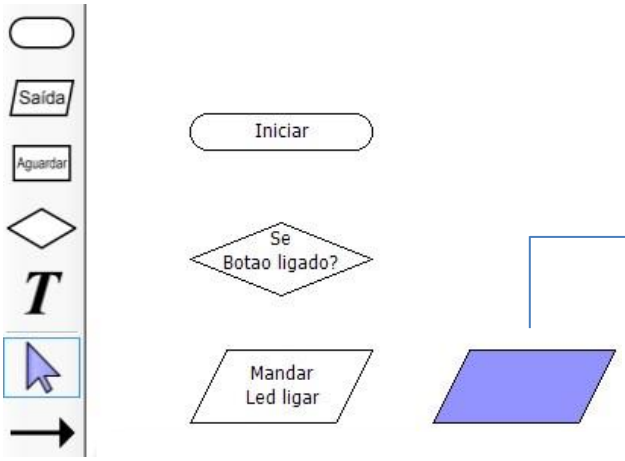

Agora temos que verificar se a chave está acionada ou não. O bloco que utilizaremos é o de Decisão. Ele é usado para comparar valores de entradas. Para utilizar basta arrastar abaixo do Bloco Iniciar. Feito isso já podemos verificar se o botão está acionado, como mostra a figura abaixo:



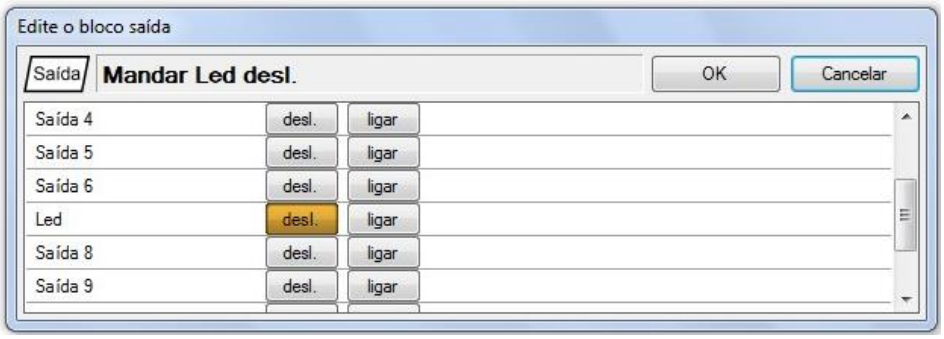



Se o botão estiver acionado, acionaremos o LED. O bloco usado será “Saída”.

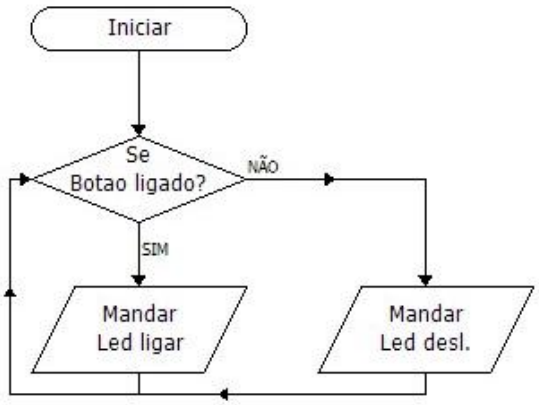


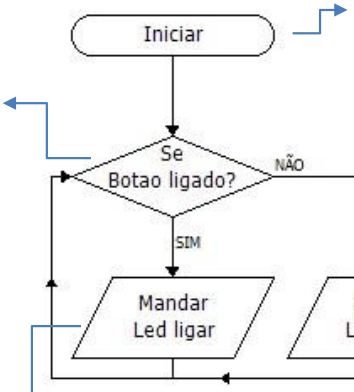

Caso contrário o Led será desligado. Para isso utiliza-se outro bloco “Saída”.



Com tudo esquematizado, devemos ligar os blocos. Para fazer a conexão entre dois blocos, basta selecionar a seta preta no painel esquerdo, e então selecionar sempre primeiro, o bloco em que está, e em seguida selecionar o bloco que deseja ir sempre ligando em ordem sequencial. Outra forma é, se um bloco for colocado exatamente embaixo do bloco anterior, ele ficará com a borda amarela e colocará a linha automaticamente. No final deve-se ligar os blocos de saída com o condicional, para que essa verificação seja contínua.


Esse bloco verifica se o botão está acionado, caso ele esteja, executa o comando “Sim”, caso contrário “Não”.



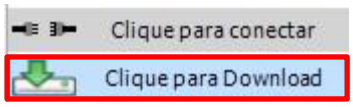

Bloco que indica o início do programa.

Esse bloco desliga o LED.

Esse bloco liga o LED.

Para simular o fluxograma é necessário que todos os blocos estejam conectados. Por fim é só clicar na seta vermelha de simulação, como na figura abaixo.

O modo anterior é feito utilizando um cabo USB, desse modo o computador fornece energia para alimentar o microcontrolador. A segunda maneira de executar o programa é fazendo o download para a placa. Desse modo o cabo USB só é necessário enquanto a transferência está sendo feita, mas após isso pode ser desconectado que o fluxograma continuará salvo. Contudo é necessário utilizar uma fonte externa para alimentar o microcontrolador (pilhas), ou mesmo o cabo, porém sem conectar no software.




No Arduino é necessário 5v para a alimentação, ou seja, 4 pilhas. O fio vermelho das pilhas deve ser colocado na porta Vin do Arduino, enquanto o fio preto vai na porta GND.

Já no microcontrolador Modelix 2.8, a alimentação pode ser feita em qualquer porta, desde que, o fio vermelho esteja no 5v e o fio preto no GND.

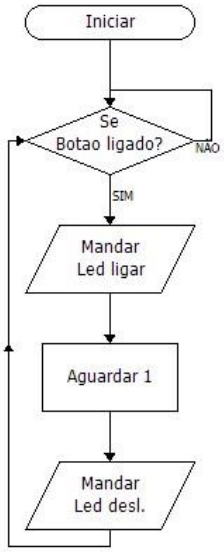
## Projeto 1

**Objetivo:** Quando o botão é acionado, o LED deve ficar ligado por 1 segundo e em seguida apagar. Como não é necessário fazer modificações nas ligações eletrônicas, iremos utilizar o esquema anterior, de modo que só iremos modificar a lógica.

### Material Utilizado

**Mesmo material que o projeto anterior.**

Para manter o LED acionado pelo tempo determinado, usaremos a função “Aguardar”. Esse bloco permite gerar um delay, ou seja, o microcontrolador irá esperar um tempo antes de continuar a sequência. Desse modo é gerado o fluxograma abaixo.




Quando o botão é acionado o LED deve ficar aceso por 1 segundo, apagado por 0,5 segundos, acender novamente por 2 segundos e apagar definitivamente. Essa sequência deve acontecer sempre que se acionar o botão

## Projeto 2

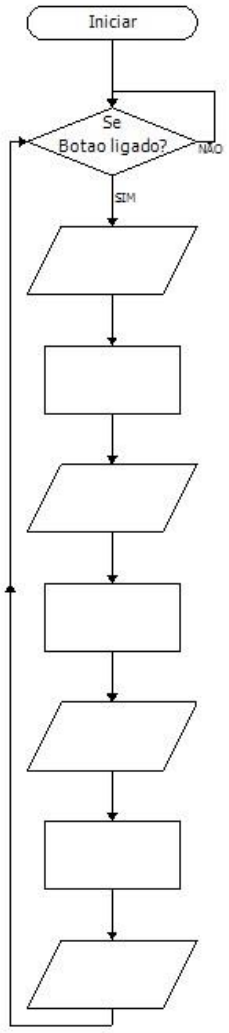
**Objetivo:** Quando o botão é acionado, o LED deve ficar ligado por 1 segundo, apagado por 0.5 segundos, ligar novamente por 2 segundos e apagar definitivamente. Essa sequência deve acontecer sempre que se acionar o botão.

Esse projeto possuí muitas semelhanças com o anterior, só aumentamos a quantidade de etapas que o LED irá fazer quando pressionar o botão.

## Material Utilizado

**Mesmo material que o projeto anterior.**

Deve-se completar os blocos que estão vazios com as informações necessárias para conseguir fazer o projeto 2. (A resposta está na apostila gabarito.)




Deve-se completar os blocos que estão vazios com as informações necessárias para conseguir fazer o projeto 2. (A resposta está na apostila gabarito.)

## Desafio

**Objetivo:** Montar a lógica de um farol de carros. Os três LEDs devem ficar acendendo e apagando em períodos diferentes, simulando um farol. O botão irá iniciar o sistema.

## Material Utilizado

3x Led (qualquer cor)

1x PushButton

Fios Conectores

1x Resistor 10kΩ

3x Resistor 330Ω

## 3.1.4 Exercícios

**3.1.4.2.** O que é delay? Qual bloco gera esse delay?

**3.1.4.3.** O LED deve ser configurado como entrada ou saída digital? E o botão? Por que?

As respostas do desafio e dos exercícios estão na apostila gabarito.

# Aula 3 – Sofisticando a programação

**Objetivo:** Agora que já conhecemos o atuador Led e o sensor botão, vamos evoluir um pouco mais nossa programação. Agora iremos utilizar um botão para cada led, ou seja, cada botão aciona um led diferente.

## Material Utilizado

1x Led (qualquer cor)

1x PushButton

Fios Conectores

1x Resistor 10kΩ

1x Resistor 330Ω

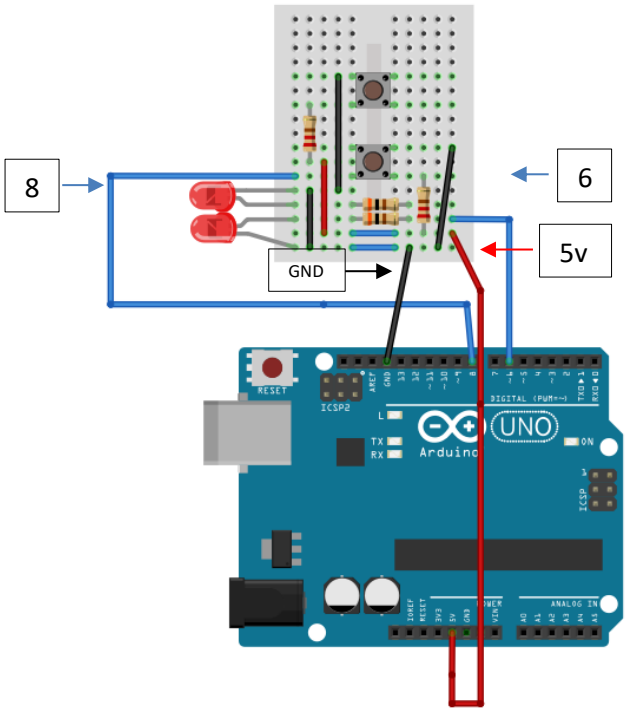
## Ligações Elétrica

Primeiramente vamos fazer as ligações elétricas. Com essa etapa concluída iremos renomear as entradas e saídas do software de acordo com as ligações. Após isso basta desenvolver o fluxograma.

| | | |
|---|---|---|
| 0 | Saída 0 | |
| 1 | Saída 1 | |
| 2 | Saída 2 | |
| 3 | Saída 3 | |
| 4 | Saída 4 | |
| 5 | Led1 | |
| 6 | Led2 | |
| 7 | Botao1 | |
| 8 | Botao2 | |
| 9 | Saída 9 | |
| 10 | Saída 10 | |
| 11 | Saída 11 | |
| 12 | Saída 12 | |
| 13 | Saída 13 | |
| 0 | Val 0 | 0,0 % |
| 1 | Val 1 | 0,0 % |
| 2 | Val 2 | 0,0 % |
| 3 | Val 3 | 0,0 % |
| 4 | Val 4 | 0,0 % |
| 5 | Val 5 | 0,0 % |

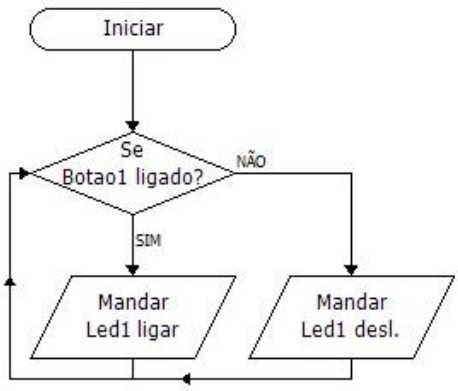
Saídas

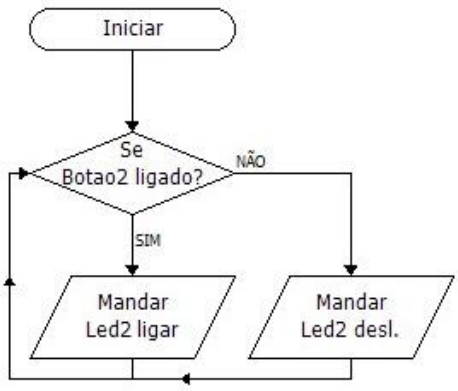
Entradas




Nesse programa iremos trabalhar com execuções em paralelos. Programação em paralelo significa que o software fará leituras ou executará uma ação ao mesmo tempo sem a interferência de outra parte do fluxograma. Desse modo será possível verificar qual botão está acionado e acender o LED correspondente.

Cada bloco “Iniciar” indica o início de um fluxo independente, ou seja, cada um desse bloco está fazendo a leitura de um botão.



## Projeto 1

**Objetivo:** Agora iremos conhecer outro atuador, o Buzzer, também conhecido como Bip. Quando for pressionado um dos botões, um LED liga e toca o buzzer. Quando o outro botão for acionado, o outro led liga e o buzzer deve tocar em outro intervalo.

Buzzer é um componente eletrônico capaz de produzir som. Quando ele é alimentado por uma fonte os seus componentes internos vibram e produzem som. Devemos lembrar que o buzzer possui polaridade, então não pode inverter a alimentação de em seus terminais.

# BUZZER:

**Funcionalidade:** Emitir som quando acionado por uma corrente elétrica.

**Quantidade de pinos:** 2 pinos. Possui polaridade, portanto deve-se ligar o pino de menor comprimento no GND. Em cima do componente indicando qual o lado positivo.

**Microcontrolador:** Quando utilizado com o microcontrolador esse componente é configurado como uma saída digital

## Material Utilizado

2x Led (qualquer cor)

2x PushButton

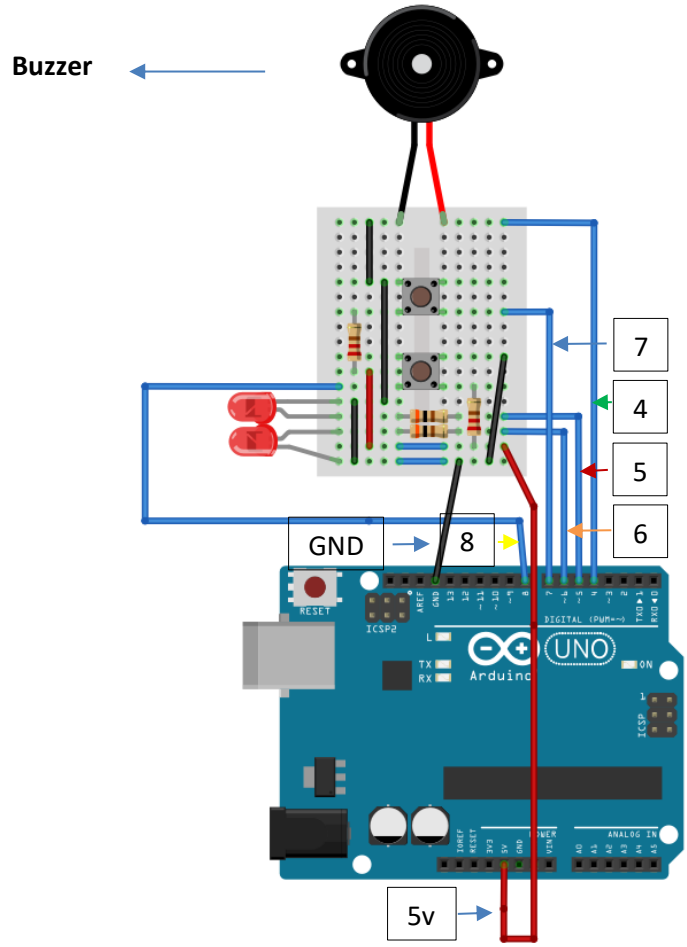
1x Buzzer

Fios Conectores

2x Resistor 10kΩ

2x Resistor 330Ω

# Ligações Elétrica

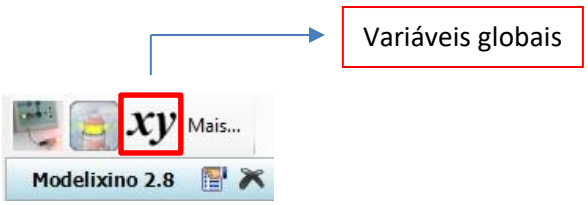



| | |
|---|---|
| 0 | Saída 0 |
| 1 | Saída 1 |
| 2 | Saída 2 |
| 3 | Saída 3 |
| 4 | Buzzer |
| 5 | Led1 |
| 6 | Led2 |
| 7 | Botao1 |
| 8 | Botao2 |
| 9 | Saída 9 |
| 10 | Saída 10 |
| 11 | Saída 11 |
| 12 | Saída 12 |
| 13 | Saída 13 |

Nesse projeto foram usadas as saídas 5 e 6 para os LEDS. Os pinos 7 e 8 para os botões (Configuradas como entradas) e o pino 4 para o buzzer

Para fazermos o buzzer tocar em intervalos diferentes, usaremos uma nova função: Variáveis. Variáveis podem ser comparadas a caixas que guardam itens dentro de si. A diferença é que no nosso caso iremos guardar um valor dentro dessa variável.

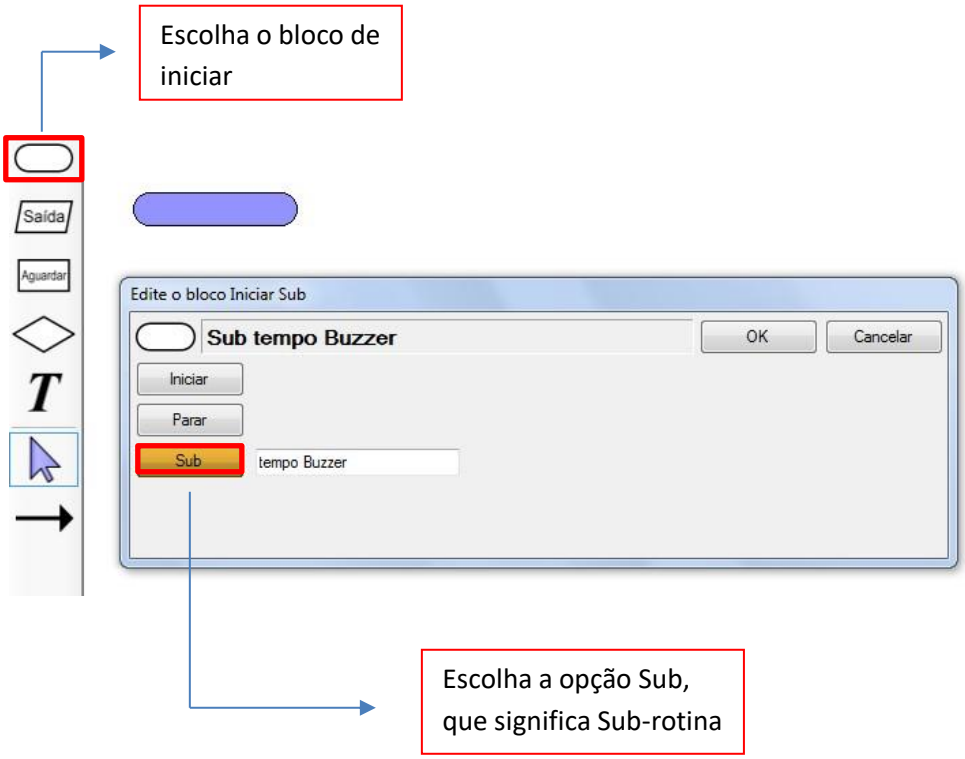
Para criar uma variável, basta clicar no símbolo XY (variáveis) e em seguida alterar seu nome, nesse exemplo foi colocado o nome “tempo”, pois será usada para armazenar o tempo que o buzzer ficará acionado.

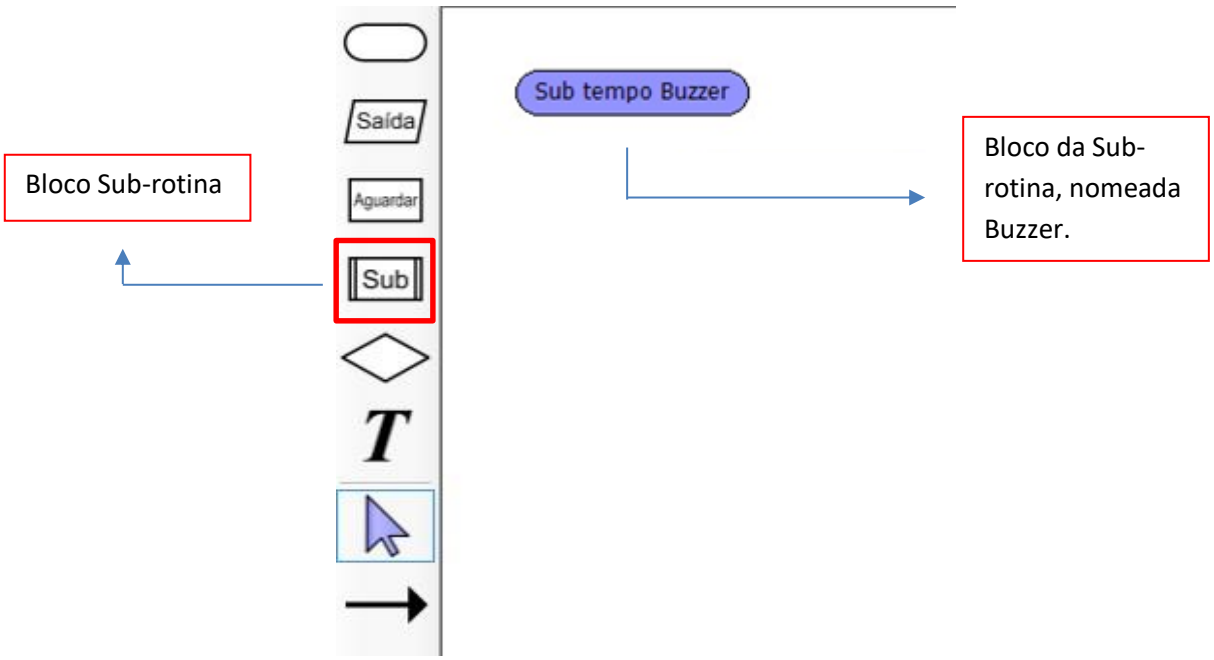



Observe que foi criado o bloco “Definir”. Utilizado para definir o valor que será guardado em nossa variável.

Quando estamos programando e precisamos repetir uma ação várias vezes podemos fazer uma única sub-rotina e ficar chamando-a várias vezes. Para criar essa função basta escolher o bloco de Iniciar e escolher a opção “Sub”.

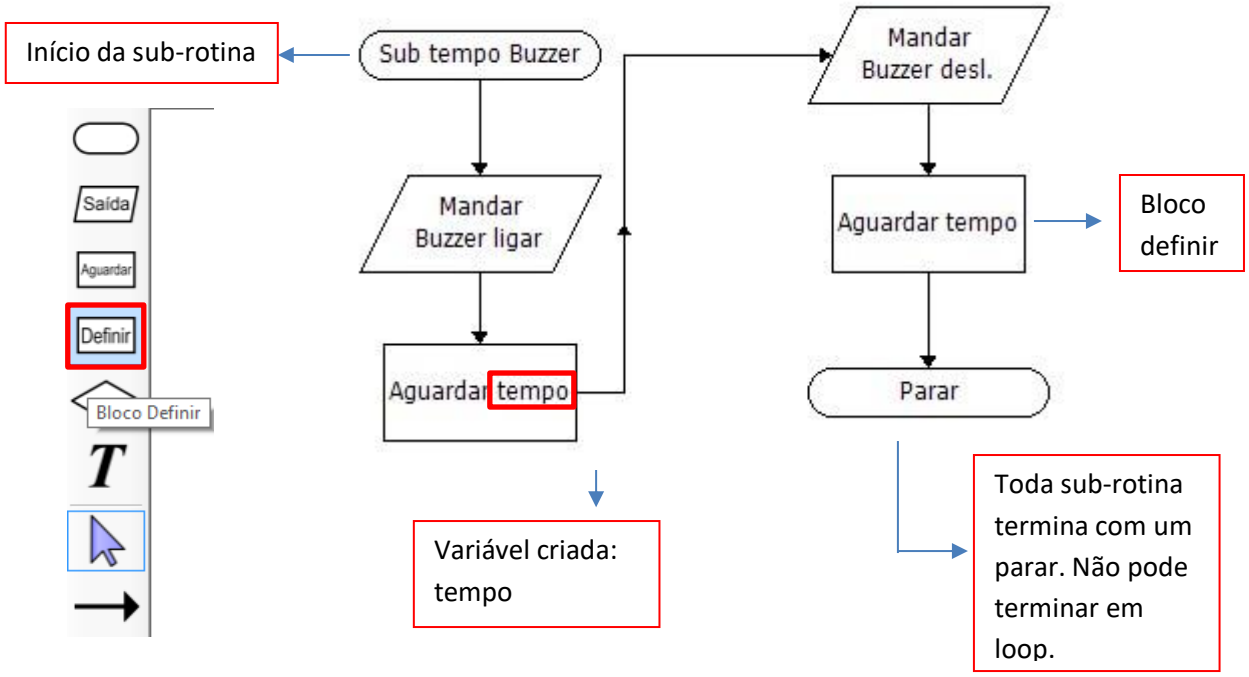



Após criar esse componente irá aparecer um novo bloco no painel da esquerda, o bloco Sub. Como é mostrado na figura abaixo.

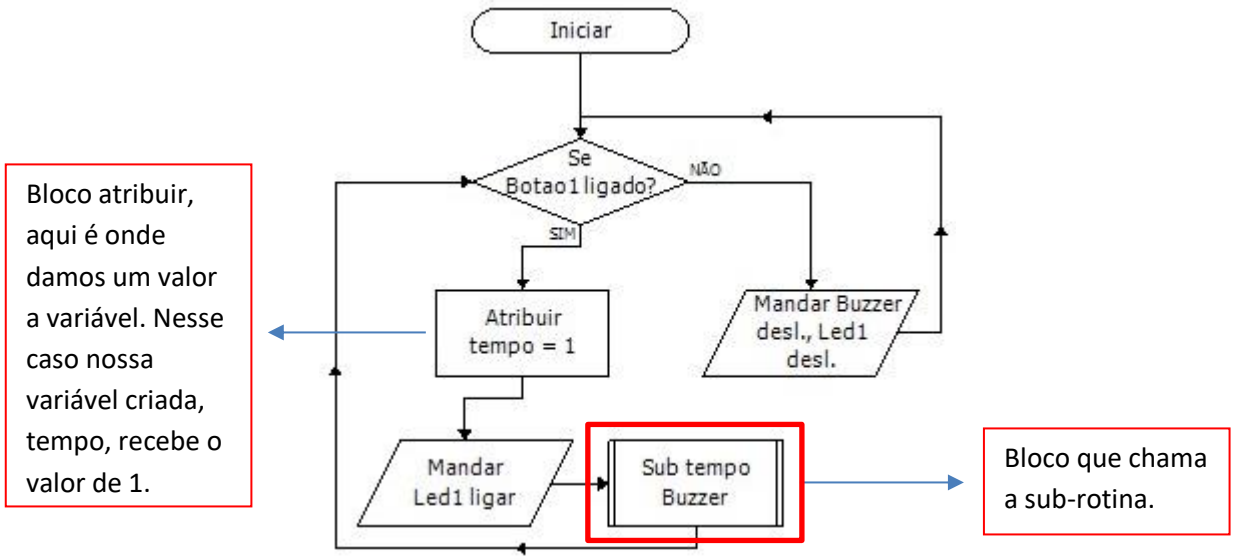


Após essa etapa já podemos desenvolver a lógica. O Buzzer será ativado e ficará assim por um tempo = “tempo” (nossa variável).

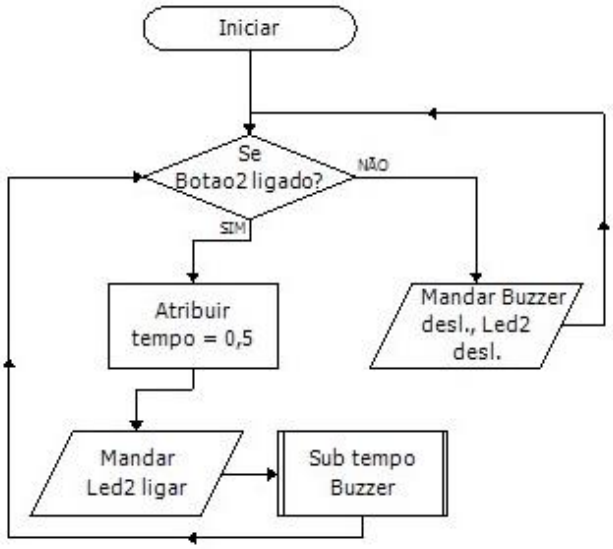
Após isso ficará desligado pelo mesmo tempo. Abaixo segue o esquema da nossa Sub-rotina:




Com isso feito já podemos desenvolver a lógica dos botões. Seguiremos o mesmo pensamento que os projetos anteriores, acrescentando o bloco “Definir”, para colocar um valor para a nosso variável tempo e chamando a sub-rotina.



Devemos fazer o mesmo procedimento para o botão 2. As únicas diferenças são: iremos ligar o Led 2 ao invés do 1 e a variável tempo recebe, nesse caso o valor de 0,5.




## Projeto 2

**Objetivo:** Quando apenas um botão for pressionado, os LEDs devem piscar com período de 1 segundo. Quando os dois botões forem acionados o buzzer dever ser acionado e o LED desligado.

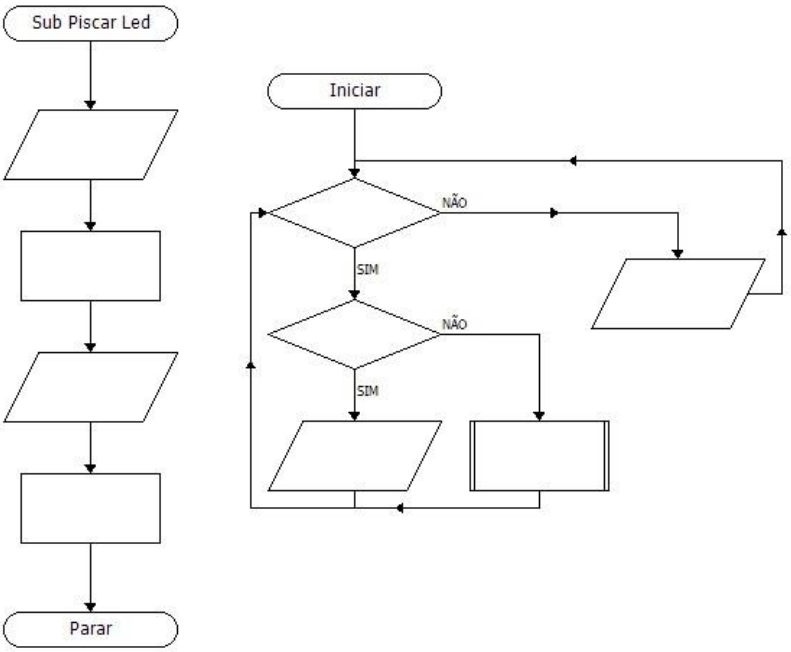
Nesse projeto teremos que trabalhar com dois conceitos aprendidos: Sub-rotinas e programação em paralelo. A sub-rotina será usada para fazer o LED piscar e em paralelo vamos verificar o estado dos botões. Um cuidado que precisamos ter é verificar se os dois botões estão acionados ao mesmo tempo, para isso pode-se utilizar dois blocos de decisão em sequência, cada um conferindo o valor de um dos botões. Se um dos botões não estiver acionado o buzzer deve ser desligado.

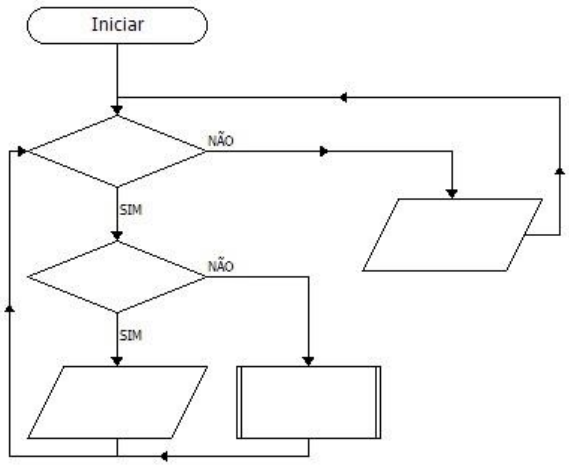
## Material Utilizado

**Mesmo material que o projeto anterior.**

## Ligações Elétrica

Deve-se completar os blocos que estão vazios com as informações necessárias para conseguir fazer o projeto 2. (A resposta está na apostila gabarito.)



## Desafio

Quando apenas um botão for pressionado, um dos LEDs deve ficar acesso continuamente e o outro piscar em intervalos de 1 segundo. Nessa situação o Buzzer fica desligado. Quando o outro botão é acionado, ambos os LEDs e o buzzer devem piscar em intervalos de 1,5 segundos. Quando ambos os botões são acionados os LEDs e o buzzer ficam acionados continuamente.

**Dica 1:** Deve-se seguir a mesma lógica do projeto anterior, usando programação em paralelo e verificar se ambos os botões estão acionados com dois blocos de condição em sequência.

**Dica 2**: Para facilitar a lógica, pode piscar os LEDs/Buzzer sem usar uma sub-rotina.

## Material Utilizado no Desafio

**2 x Led**

**2 x PushButton**

**1 x Buzzer**

**2 x Resistor 300 Ω**

**2 x Resistor 10k Ω**

**Fios**

## Exercícios

**3.2.4.1.** O que é programação paralela?

**3.2.4.2.** O Buzzer é um elemento de entrada ou saída digital?

As respostas do desafio e dos exercícios estão na apostila gabarito.

## Aula 4 – Sensor analógico de temperatura

**Objetivo:** Conhecer os sensores analógicos e seu funcionamento, nessa aula iremos aprender a fazer a leitura do sensor de temperatura em °C, utilizando as entradas analógicas.

**Sensor analógico** possui uma faixa de valores que podem ser trabalhados, a determinação de qual valor será usado irá depender do objetivo que dará para seu robô. Ele difere do sensor digital que só tem 2 valores, 0 e 1, ou seja, ligado e desligado. Por exemplo: o sensor de temperatura usado nessa aula pode medir de 0° C a 100° C, então qualquer valor dentro de

O LM35 é um sensor que faz a leitura da temperatura em °C, porém a informação que ele fornece ao microcontrolador (pino do meio do sensor) é uma tensão que varia de acordo com a temperatura ambiente. Esse modelo possui um incremento de 10mV/°C, ou seja, cada vez que aumentar 1°C da temperatura a tensão que o sensor envia ao microcontrolador tem um aumento de 10mV.

# LM 35:

**Funcionalidade:** Medir a temperatura do ambiente que se encontra.
Pode fazer medições de -55 °C à 150 °C.

**Quantidade de pinos:** 3 pinos. Dois para alimentação e o terceiro (central) é a saída do sinal.

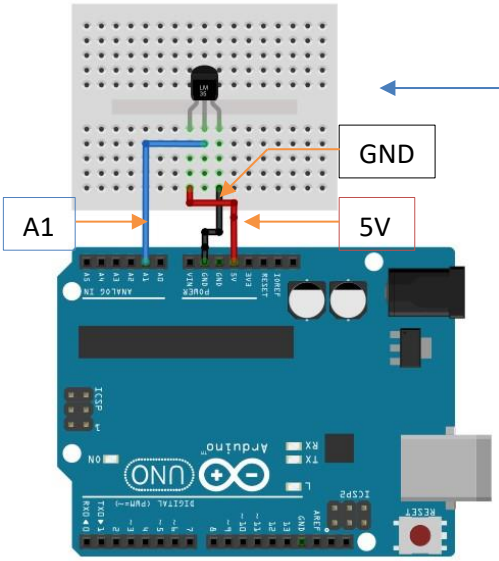
**Microcontrolador:** Quando utilizado com o microcontrolador esse componente é configurado como entrada e o pino central é ligado à entrada analógica.

## Material Utilizado

1x Sensor de temperatura LM35

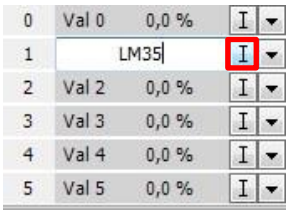
Fios Conectores

## Ligações Elétrica

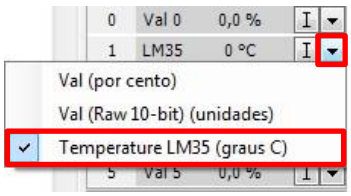



Como sempre iniciaremos com a montagem do circuito elétrico, contudo devemos estar atentos para ligarmos o sensor na entrada analógica do microcontrolador, conforme mostra a figura.

Como estamos trabalhando com um sensor analógico, será usado as entradas analógicas do software (Val 0 ao Val 5). Podemos renomear a entrada utilizada (olhar em qual pino o sensor está conectado no microcontrolador – nesse caso no pino 1) e alterar a leitura para graus C.

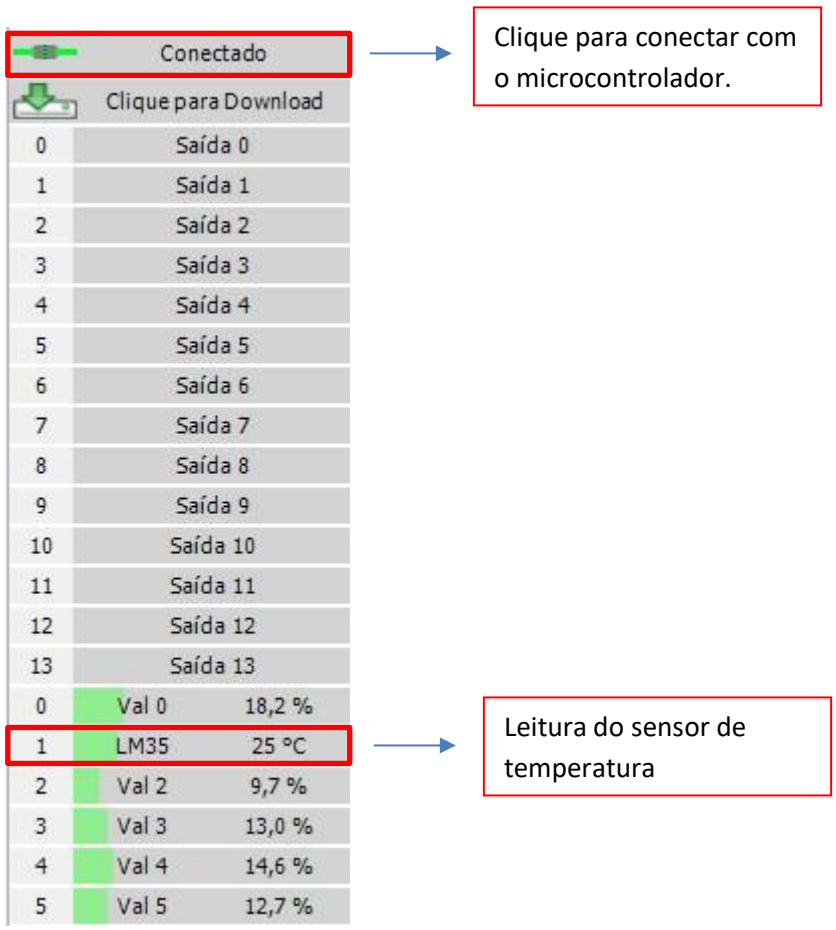



Para mudar o nome é necessário clicar no ícone que parece um I.




O sensor de temperatura é o único sensor que para fazer a medição em graus é necessário clicar na seta e escolher a opção “Temperature LM35 (graus C)”.

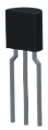
Nesse projeto só queremos fazer a leitura do sensor, então só iremos conectar o microcontrolador ao software e já fazemos a leitura do sensor.




## Projeto 1

Desenvolver um projeto que quando a temperatura for maior que 25°C o alarme soará.

## Material Utilizado



1x Sensor de temperatura LM35

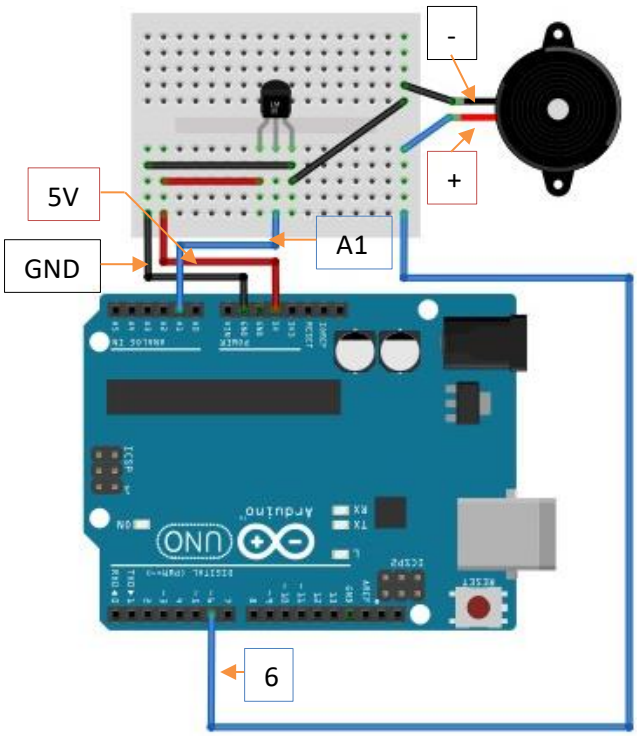
1x Buzzer

Fios Conectores

## Ligações Elétrica

Primeiramente iremos fazer a montagem do circuito. Fique atento que esses componentes possuem polaridade.

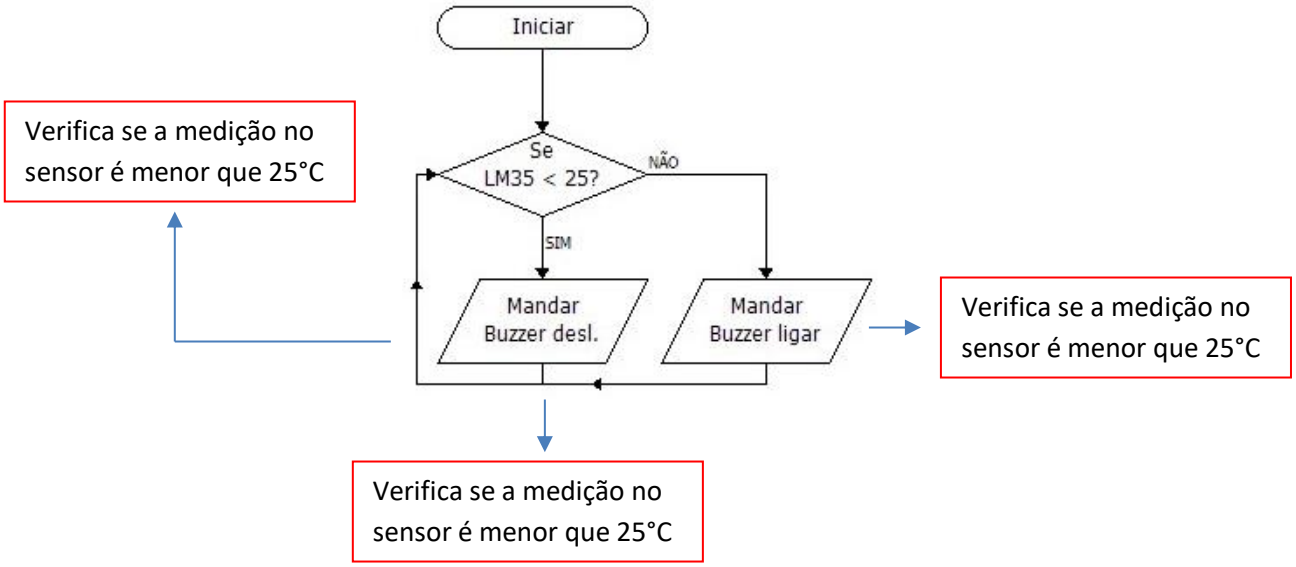
A lógica desse projeto é bem simples, basta verificar se a temperatura é menor que 25°C, caso seja não aciona o alarme, caso contrário o alarme é ativado.




| | | |
|---|---|---|
| 0 | Saída 0 | |
| 1 | Saída 1 | |
| 2 | Saída 2 | |
| 3 | Saída 3 | |
| 4 | Saída 4 | |
| 5 | Saída 5 | |
| 6 | Buzzer | |
| 7 | Saída 7 | |
| 8 | Saída 8 | |
| 9 | Saída 9 | |
| 10 | Saída 10 | |
| 11 | Saída 11 | |
| 12 | Saída 12 | |
| 13 | Saída 13 | |
| 0 | Val 0 | 24,9 % |
| 1 | LM35 | 25 °C |
| 2 | Val 2 | 13,3 % |
| 3 | Val 3 | 18,9 % |
| 4 | Val 4 | 21,6 % |
| 5 | Val 5 | 21,5 % |

O buzzer está conectado no pino digital 6.

O LM35 está conectado na entrada analógica 1.



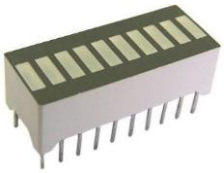

## 3.3.2 Projeto 2

Acender os LEDs da barra gráfica conforme aumenta a temperatura e se a temperatura estiver muito alta, aciona um alarme.

### Material Utilizado

**1 x Sensor LM35**

**1 x Barra gráfica de Leds**



**1 x Buzzer**

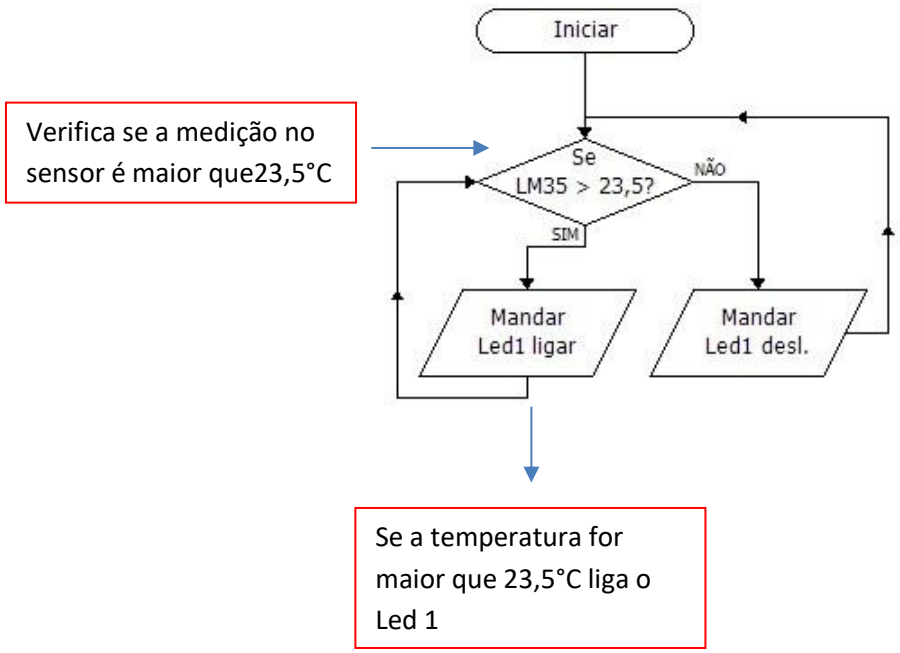
**1 x Resistor 300 Ω**

**Fios Conectores**

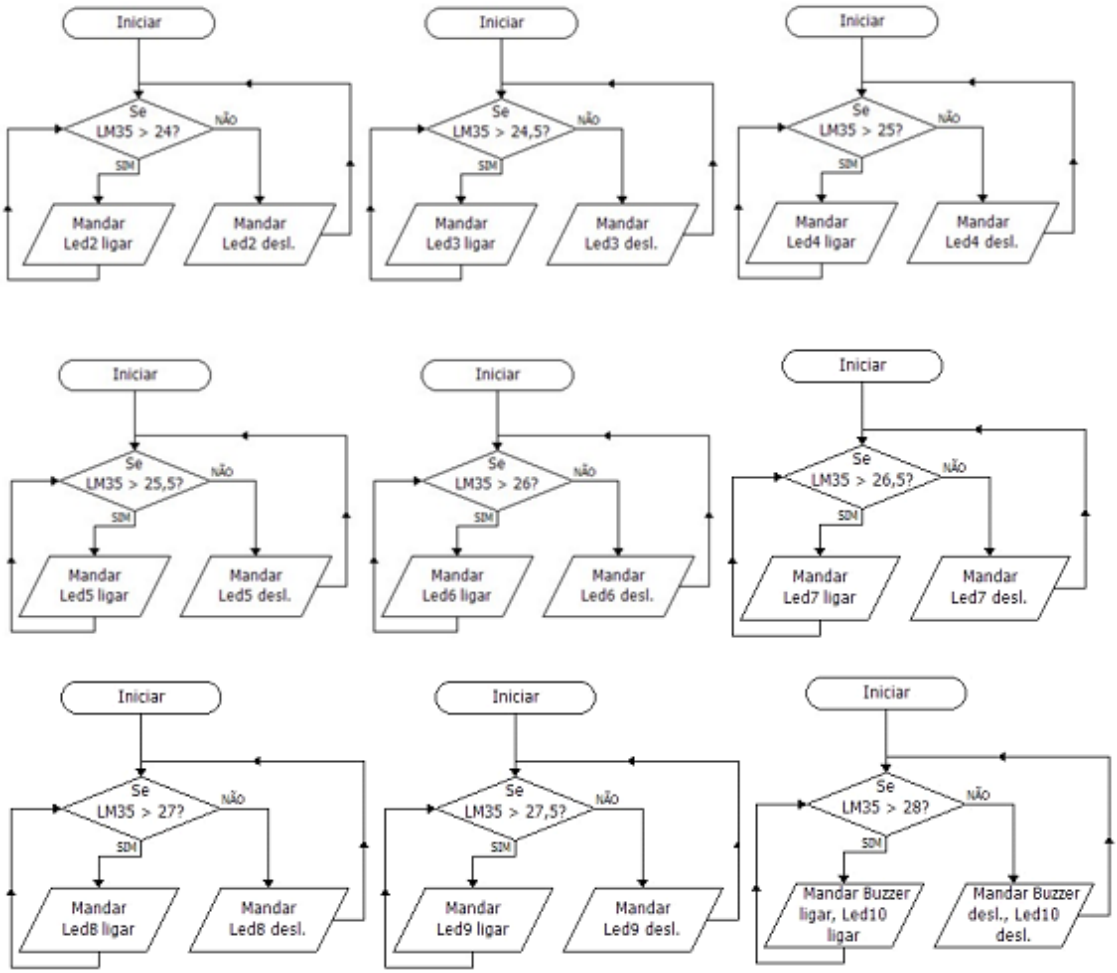
# Ligações Elétrica

A montagem desse circuito exige muita atenção, pois são utilizados muitos cabos. Para ter certeza que tudo está ligado corretamente, não esqueça de testar cada LED como vimos no Módulo 1.

A lógica desse projeto é simples, pois só precisamos verificar se a temperatura aumentou e ir acionando os LEDs, caso a temperatura seja maior que 28°C é acionado o buzzer.



Repetimos esse procedimento para todos os outros 9 Leds, mudando apenas o valor do sensor que iremos comparar. Como podemos ver na imagem abaixo:




| | | |
|---|---|---|
| | Conectado | |
| | Clique para Download | |
| 0 | Saída 0 | |
| 1 | Saída 1 | |
| 2 | Saída 2 | |
| 3 | Buzzer | |
| 4 | Led1 | |
| 5 | Led2 | |
| 6 | Led3 | |
| 7 | Led4 | |
| 8 | Led5 | |
| 9 | Led6 | |
| 10 | Led7 | |
| 11 | Led8 | |
| 12 | Led9 | |
| 13 | Led10 | |
| 0 | Val 0 | 20,0 % |
| 1 | LM35 | 26 °C |
| 2 | Val 2 | 10,5 % |
| 3 | Val 3 | 14,1 % |
| 4 | Val 4 | 16,0 % |
| 5 | Val 5 | 17,1 % |

Atenção em cada fluxograma muda-se o valor do sensor de temperatura e o Led que deve ativar.

Não se esqueça de trocar a unidade para °C (como já aprendemos no início do módulo 3).

## 3.3.3 Desafio

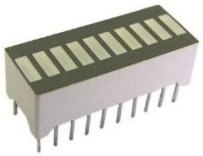
Simular um alarme de incêndio. Quando a temperatura for maior que 80°C, o Buzzer deve tocar por 1,5 segundos e ficar desligado 0,5 segundo. Os LEDs, devem piscar com esse mesmo intervalo. Se não houver incêndio, a barra de LEDs deve ser acionada progressivamente.

**Dica 1:** A lógica é muito semelhante com o projeto anterior, contudo, teremos que verificar o valor da temperatura antes de ativar ou desligar uma saída, para evitar que os dois blocos de fluxogramas executem ao mesmo tempo.

### Material Utilizado no Desafio

**1 x Sensor**

**1 x Barra gráfica de**



**1 x Buzzer**

**1 x Resistor 300 Ω**

**Fios Conectores**

## 3.3.4 Exercícios

**3.3.4.1.** Qual a diferença de entradas analógicas e entradas digitais?

**3.3.4.2.** Quando trabalhamos com entradas analógicas no software Modelix, quais as unidades que podemos ter?

**3.3.4.3.** Por que usamos apenas um resistor com a barra gráfica em vez de 10 resistores (um para cada segmento)?

As respostas do desafio e dos exercícios estão na apostila gabarito.

## 3.4 Módulo 4

Desenvolver um projeto que permita fazer um LED piscar mais rápido ou mais devagar conforme muda o valor do potenciômetro.

## Material Utilizado

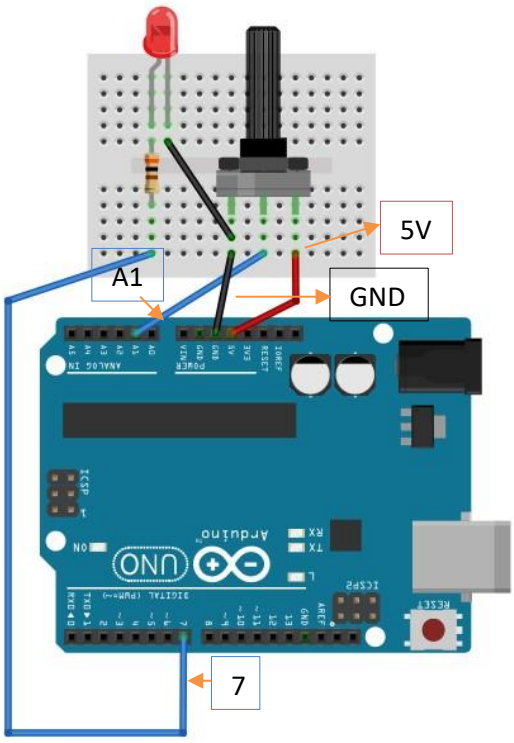
**1 x Potenciômetro 10k Ω**

**1 x Resistor 300 Ω**

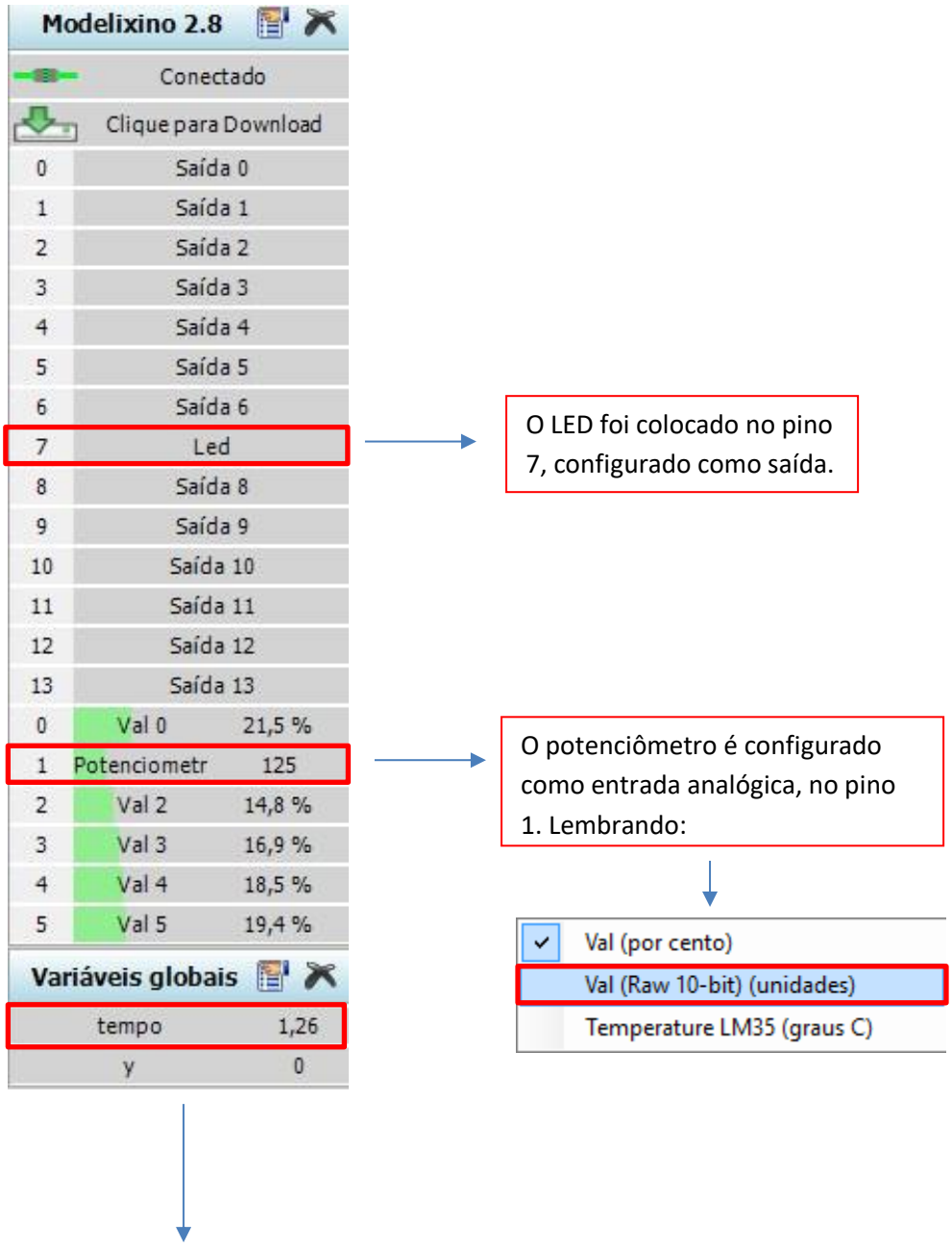
**1 x LED (Qualquer cor)**

**Fios Conectores**

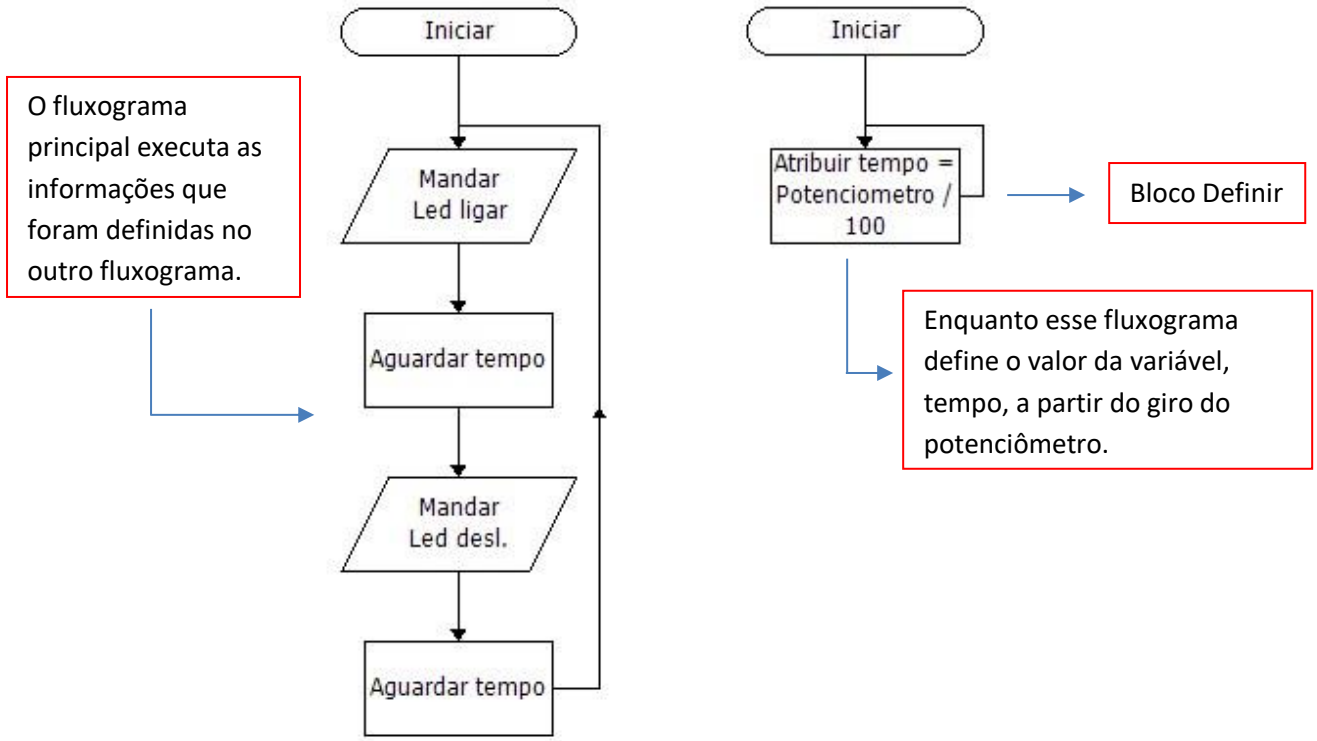
## Ligações Elétrica




A montagem desse circuito é mais simples que o anterior, mas ainda devemos estar atentos na montagem. Ligamos o potenciômetro na entrada analógica e o LED em um pino digital que configuraremos como saída.


Modelixino 2.8
Conectado
Clique para Download
0 Saída 0
1 Saída 1
2 Saída 2
3 Saída 3
4 Saída 4
5 Saída 5
6 Saída 6
7 Led
8 Saída 8
9 Saída 9
10 Saída 10
11 Saída 11
12 Saída 12
13 Saída 13
0 Val 0 21,5 %
1 Potenciometr 125
2 Val 2 14,8 %
3 Val 3 16,9 %
4 Val 4 18,5 %
5 Val 5 19,4 %
Variáveis globais
tempo 1,26
y 0
O LED foi colocado no pino 7, configurado como saída.
O potenciômetro é configurado como entrada analógica, no pino 1. Lembrando:
Val (por cento)
Val (Raw 10-bit) (unidades)
Temperature LM35 (graus C)
Como teremos que controlar o tempo em que o LED ficará aceso ou apagado, usaremos uma variável que foi chamada de “tempo”. (NÃO ESQUECER DE CRIAR A VARIÁVEL NO BOTÃO XY).

Como queremos que o valor da nossa variável “tempo” seja atualizada constantemente, iremos fazer essa leitura em paralelo.  E mandar o LED piscar infinitamente.



## 3.4.1 Projeto 1

Desenvolver um fluxograma que faça com que a intensidade luminosa do LED mude de acordo com o valor do potenciômetro.

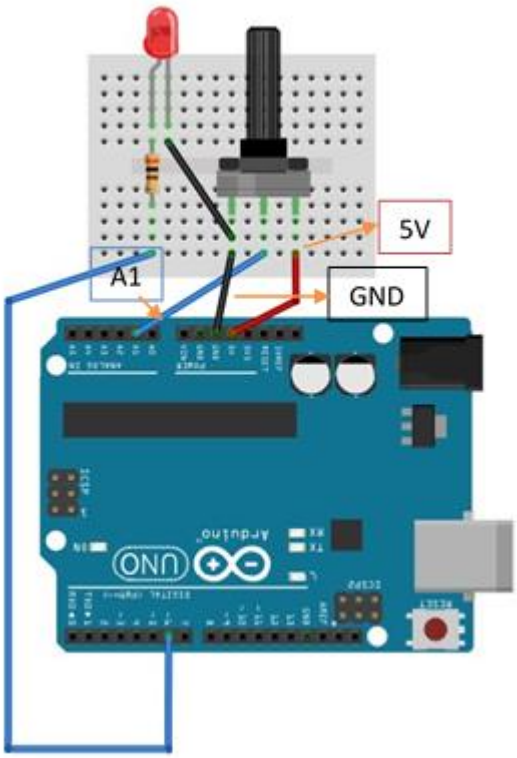
Para controlar a intensidade luminosa emitida pelo LED, iremos utilizar um PWM. Esta sigla vem do inglês *Pulse Width Modulation* que significa modulação por largura de pulso, ou seja, através da largura do pulso é feito o controle de potência ou velocidade.

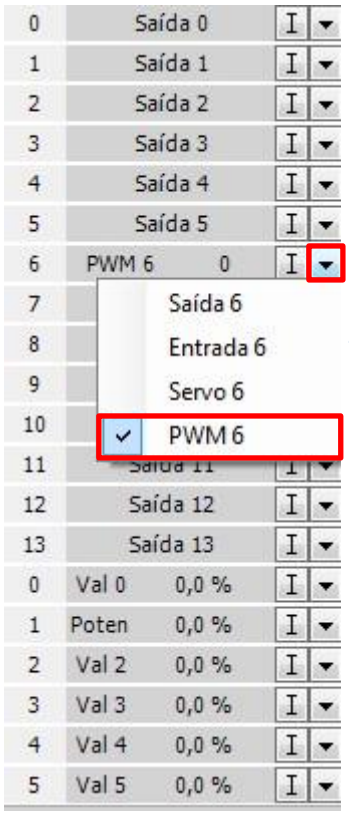
## Material Utilizado

**Mesmo material que o projeto anterior.**

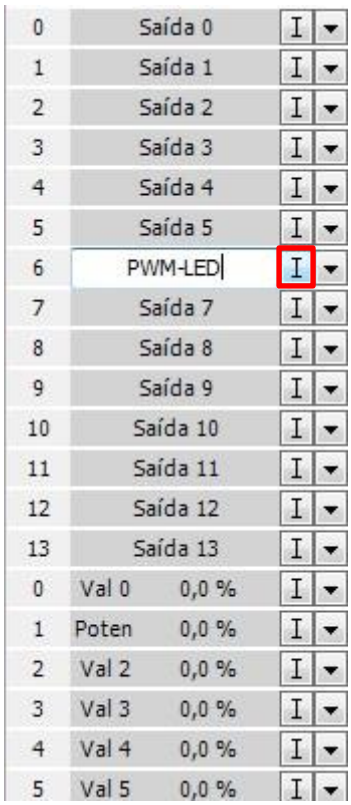
## Ligações Elétrica

Primeiramente faremos a montagem do circuito, contudo não podemos colocar o LED em qualquer pino digital, pois precisamos que esse pino permita uma modulação PWM. Esses pinos são 3, 5, 6, 9, 10 e 11. Nesse caso foi usado o 6.

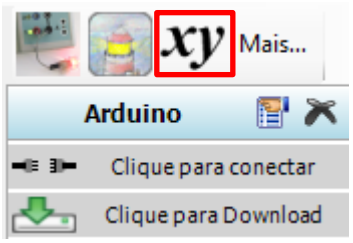





Para renomear o campo, basta clicar no ícone que parece a letra I e colocar o nome desejado.



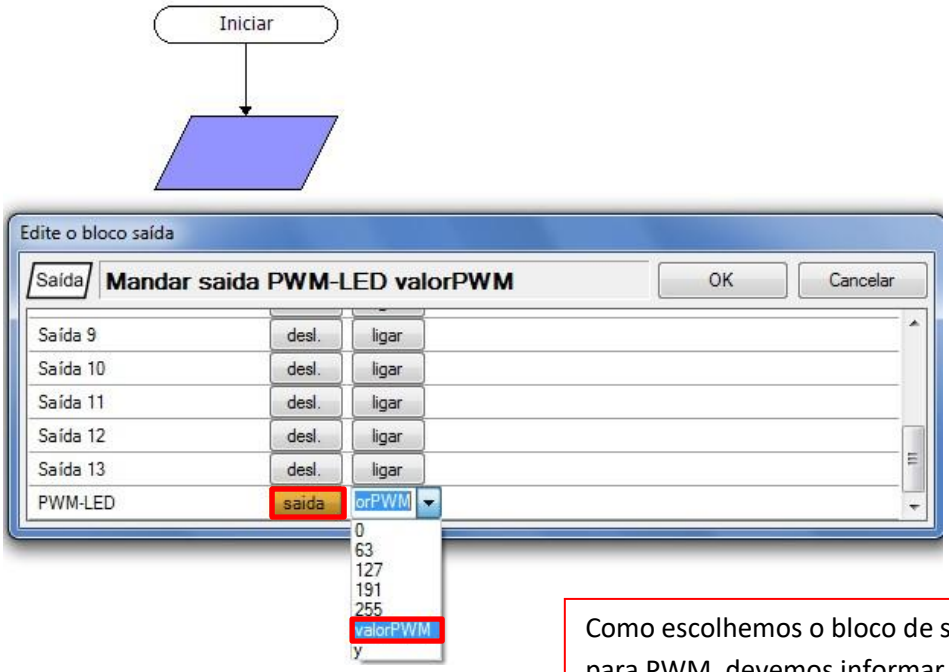

Para usar o PWM do microcontrolador devemos escolher a opção PWM, mesmo o Led sendo uma saída




Deve-se clicar no XY para criar uma nova variável.

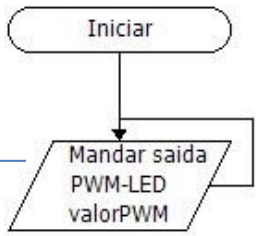

Para controlar o PWM, utilizaremos uma variável para guardar o valor do potenciômetro, pois não é possível mudar o valor do PWM com o valor direto deste. Essa variável foi chamada de “valor PWM”.

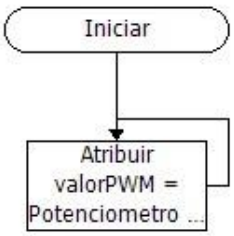
Para renomear a variável valor PWM.




Como escolhemos o bloco de saída para PWM, devemos informar qual valor iremos assumir. No nosso caso, iremos escolher a variável criada "valor PWM".

Mandar o LED ligar com uma potência igual ao valor da variável "valor PWM"



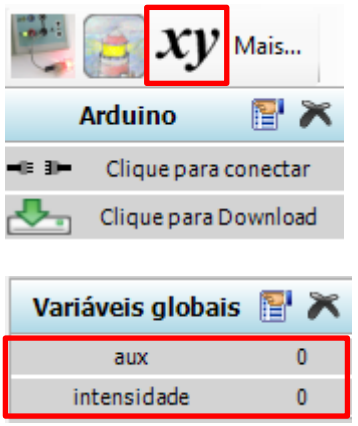




Conforme gira o potenciômetro um valor é atribuído no fluxograma e esse valor é salvo na variável "valor PWM".

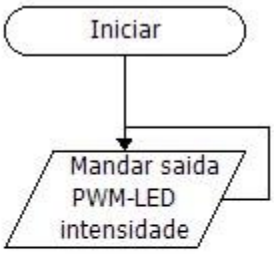
Para termos um controle mais preciso da intensidade luminosa devemos considerar que a intensidade máxima do LED ocorre quando o valor do PWM é 255. A variação do potenciômetro é de 0 a 1023. Utilizando fórmulas matemáticas chegamos na seguinte equação:

$$Intensidade_{LED} = \frac{255 * Valor_{potênciometro}}{1023}$$

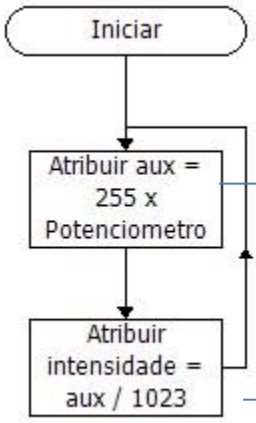
Para utilizar a fórmula mostrada anteriormente vamos criar mais duas variáveis, uma auxiliar para receber o valor do potenciômetro e outra que irá receber a intensidade.




Para criar as novas variáveis basta clicar no “XY” e renomear as mesmas.




Fluxograma principal, que recebe a informação da variável intensidade.




Variável que recebe o valor do potenciômetro, conforme a fórmula.

Variável que armazena o resultado da conversão do PWM para intensidade luminosa.

## 3.4.2 Desafio 1

Desenvolver um projeto que permita fazer o Buzzer apitar mais rápido ou mais devagar conforme muda o valor do potenciômetro.

**Dica 1:** Modifique as ligações eletrônicas para inserir o potenciômetro.

**Dica 2:** Siga a mesma lógica usada para os LEDs.

### Material Utilizado no Desafio

**1 x Potenciômetro 10k Ω**

**1 x Buzzer**

**Fios Conectores**

## 3.4.3 Desafio 2

Desenvolver um projeto que mude a intensidade sonora do buzzer conforme muda o valor do potenciômetro.

### Material Utilizado no Desafio

**1 x Potenciômetro 10k Ω**

**1 x Buzzer**

**Fios Conectores**

## 3.4.4 Exercícios

**3.4.4.1.** Um potenciômetro é um componente de entrada ou saída? Digital ou analógico?

**3.4.4.2.** Qual o valor da resistência máxima do potenciômetro?

**3.4.4.3.** Por que não podemos ligar um LED sem a presença de um resistor?

## 3.5 Módulo 5

Desenvolver um projeto que se a iluminação ambiente diminuir o LED deve acender gradativamente.

### Material Utilizado

**1 x Resistor 10k Ω**

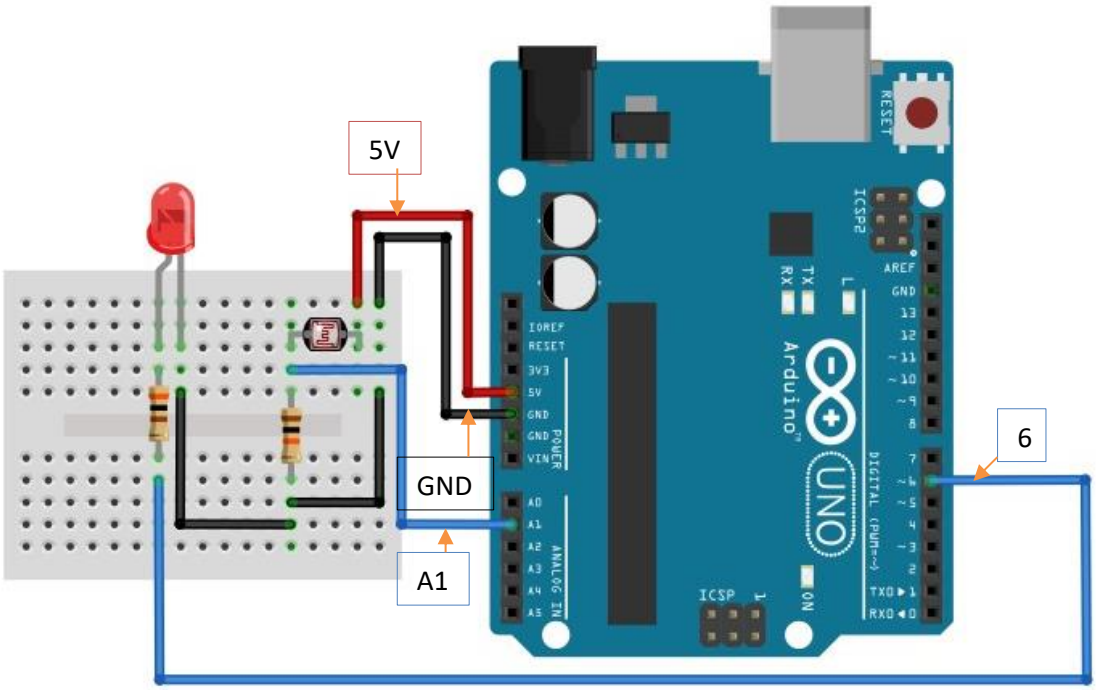
**1 x Resistor 300 Ω**

**Fios Conectores**

**1 x Sensor LDR**

**1 x LED (Qualquer cor)**

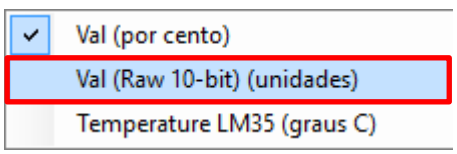
# Ligações Elétrica




O Led é conectado na porta digital 6. Configurado como PWM.

| | | |
|---|---|---|
| 0 | Saída 0 | |
| 1 | Saída 1 | |
| 2 | Saída 2 | |
| 3 | Saída 3 | |
| 4 | Saída 4 | |
| 5 | Saída 5 | |
| 6 | LedPWM | 0 |
| 7 | Saída 7 | |
| 8 | Saída 8 | |
| 9 | Saída 9 | |
| 10 | Saída 10 | |
| 11 | Saída 11 | |
| 12 | Saída 12 | |
| 13 | Saída 13 | |
| 0 | Val 0 | 67,8 % |
| 1 | SensorLuz | 780 |
| 2 | Val 2 | 72,9 % |
| 3 | Val 3 | 70,8 % |
| 4 | Val 4 | 69,1 % |
| 5 | Val 5 | 71,6 % |
| Variáveis globais | | |
| ValorLDR | | 77,4 |

O sensor LDR é configurado como uma entrada analógica, no pino 1. O tipo de leitura é bits



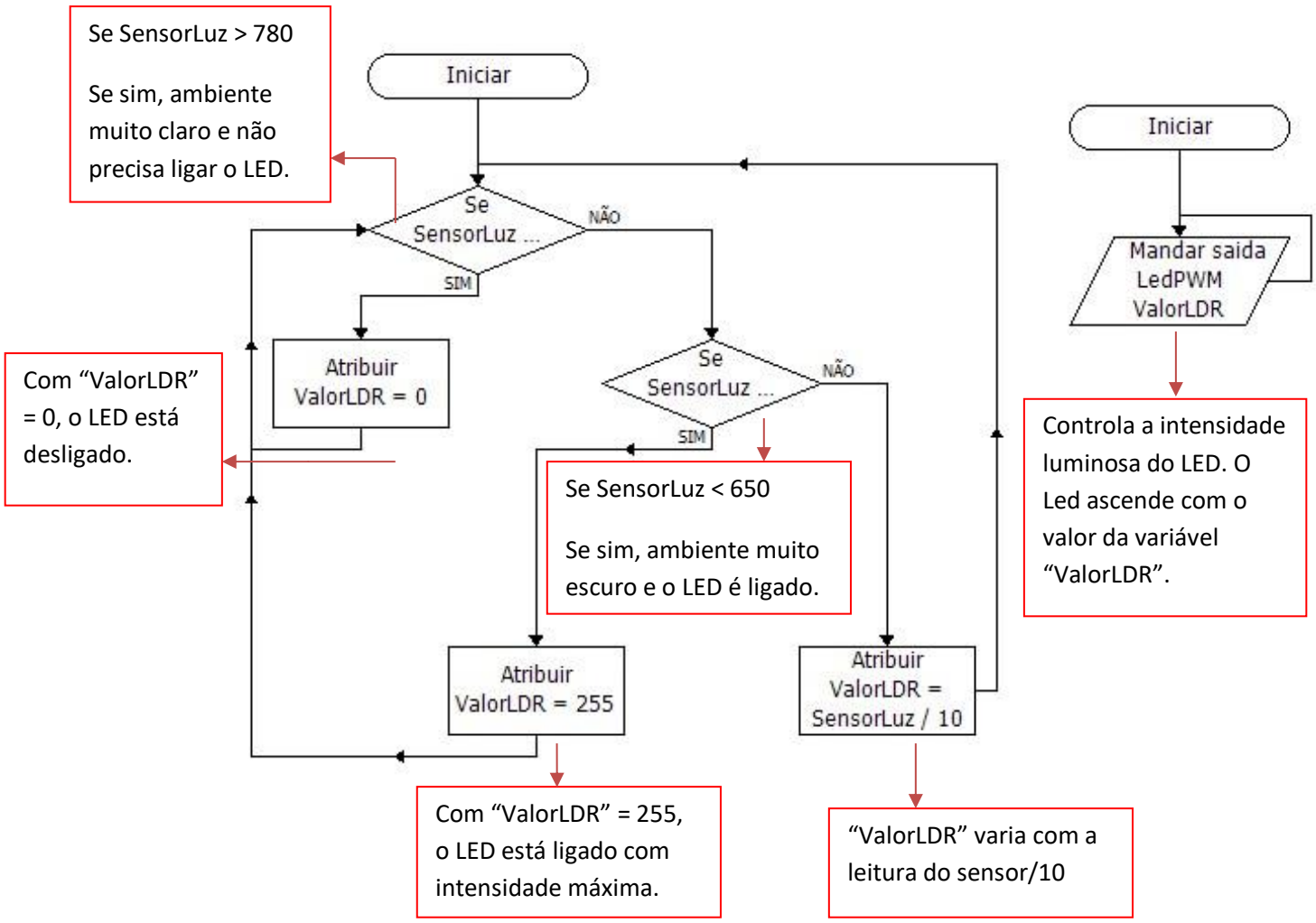

Variável que controla a intensidade luminosa do LED.

Como a luminosidade do LED depende da intensidade luminosa ambiente, temos que começar verificando o valor que o sensor está medindo. Fazendo testes com o microcontrolador no modo conectado, chegamos na conclusão que valores maiores que 780 indicam um ambiente claro em que o LED não precisa ascender.

A próxima verificação é para informar se o sensor está captando um dado menor que 650, pois iremos considerar um ambiente muito escuro e o LED tem que ascender com intensidade máxima (255).

Se o valor medido não for nenhuma das opções, fazemos uma pequena fórmula para adequar o valor lido pelo sensor para o valor do PWM.

A ultima etapa é ligar o LED com o valor da variável “ValorLDR”. Esse valor irá mudar dependendo do valor lido pelo LDR.



## 3.5.1 Projeto 1

Desenvolver um projeto que se a iluminação ambiente diminuir o Buzzer deve soar com intervalos maiores.

### Material Utilizado

**1 x Resistor 10k Ω**

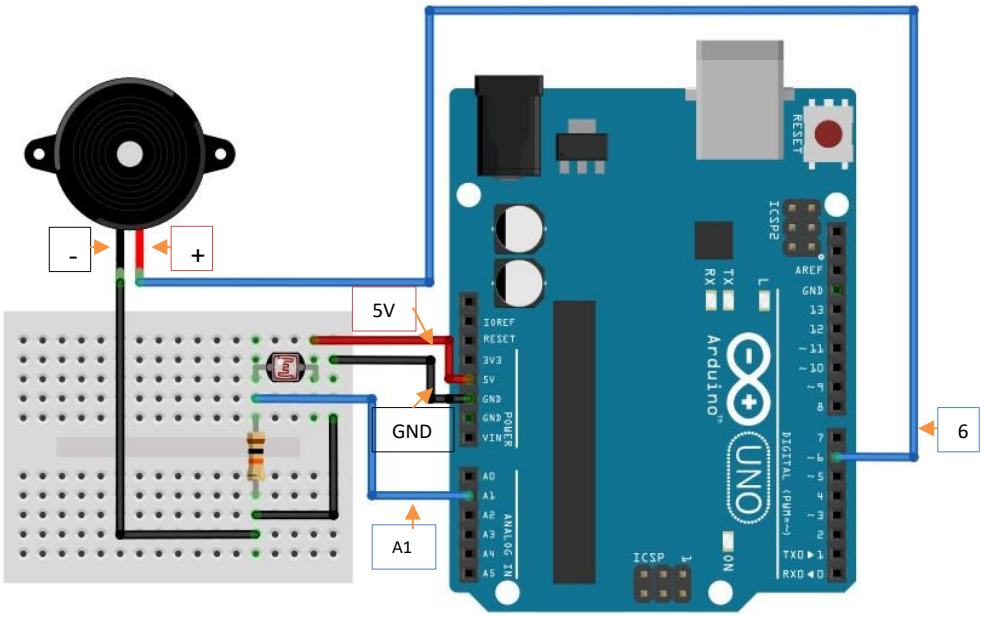
**Fios Conectores**

**1 x Sensor LDR**

**1 x Buzzer**

Deve-se completar os blocos que estão vazios com as informações necessárias para conseguir fazer o projeto 2. (A resposta está na apostila gabarito.)

# Ligações Elétrica




| | | |
|---|---|---|
| 0 | Saída 0 | |
| 1 | Saída 1 | |
| 2 | Saída 2 | |
| 3 | Saída 3 | |
| 4 | Saída 4 | |
| 5 | Saída 5 | |
| 6 | Buzzer | |
| 7 | Saída 7 | |
| 8 | Saída 8 | |
| 9 | Saída 9 | |
| 10 | Saída 10 | |
| 11 | Saída 11 | |
| 12 | Saída 12 | |
| 13 | Saída 13 | |
| 0 | Val 0 | 0,0 % |
| 1 | LDR | 0 |
| 2 | Val 2 | 0,0 % |
| 3 | Val 3 | 0,0 % |
| 4 | Val 4 | 0,0 % |
| 5 | Val 5 | 0,0 % |
| Variáveis globais | | |
| | tempo | 0 |
| | aux | 0 |

O buzzer é conectado na porta digital 6.

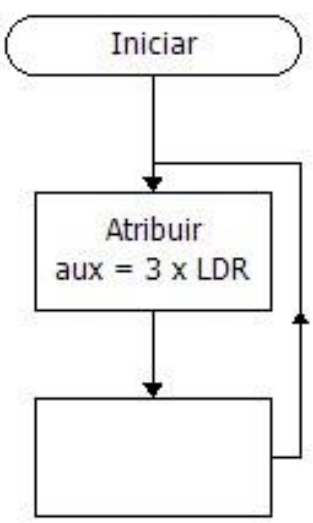
O LDR é conectado na entrada analógica 1. Configurado para bits.

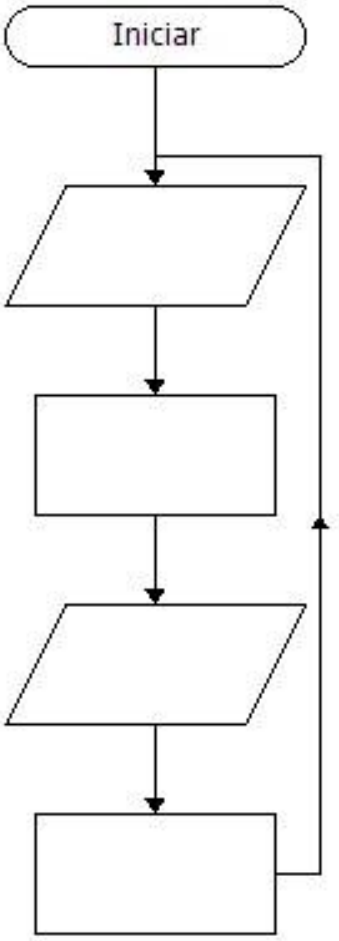
Variáveis auxiliares.

O software segue a mesma lógica do Módulo 4, só que nesse caso o controle será feito pelo LDR e não pelo potenciômetro. Outra característica é que também existirá 2 fluxogramas em paralelos.

Os intervalos máximos dos bips serão de 3 segundos. O valor máximo lido pelo LDR é 1023. Com esses dados chegamos na formula matemática abaixo e concluímos a lógica.

$$tempo = \frac{3 * Valor_{LDR}}{1023}$$



## 3.5.2 Desafio 1

Agora vamos variar a intensidade sonora do buzzer, conforme mudamos a iluminação local.

**Dica 1:** Utilize a lógica aprendida nos projetos anteriores.

### Material Utilizado no Desafio

**1 x Resistor 10k Ω**

**Fios Conectores**

**1 x Sensor LDR**

**1 x Buzzer**

## 3.5.3 Desafio 2

Desenvolver um projeto que mude o intervalo que o LED pisque e que o Buzzer apite de acordo com a luminosidade ambiente. O intervalo máximo deve ser 2 segundos.

**Dica 1:** Modifique as ligações eletrônicas para inserir o LED.

**Dica 2:** Siga a mesma lógica usada para os LEDs.

## Material Utilizado no Desafio

1 x Resistor 10k Ω

1 x Resistor 300 Ω

Fios Conectores

1 x Buzzer

1 x Sensor LDR

1 x LED (Qualquer cor)

## 3.5.4 Exercícios

**3.4.4.1. O LDR é um componente de entrada ou saída? Analógica ou digital?**

**3.4.4.2. Queremos fazer que um LED pisque em intervalos (nos exercícios anteriores chamamos esses intervalos com a variável "tempo") de no máximo 5 segundos quando o valor do LDR é máximo. Qual a fórmula matemática que utilizaríamos para isso?**

As respostas do desafio e dos exercícios estão na apostila gabarito.

## 3.5 Módulo 6

Desenvolver um projeto que o RGB liga cada cor por 0.5 segundos**.** Um LED RGB é a junção de três leds, um vermelho (R de Red), um verde (G de Green) e um azul (B de Blue). Seu funcionamento é igual aos dos LEDs comuns, para ascender é preciso ter uma alimentação. O pino maior é o que vai para o GND e serve para as três cores. Os demais pinos vão para o microcontrolador que enviará 5V para que acione a cor desejada.

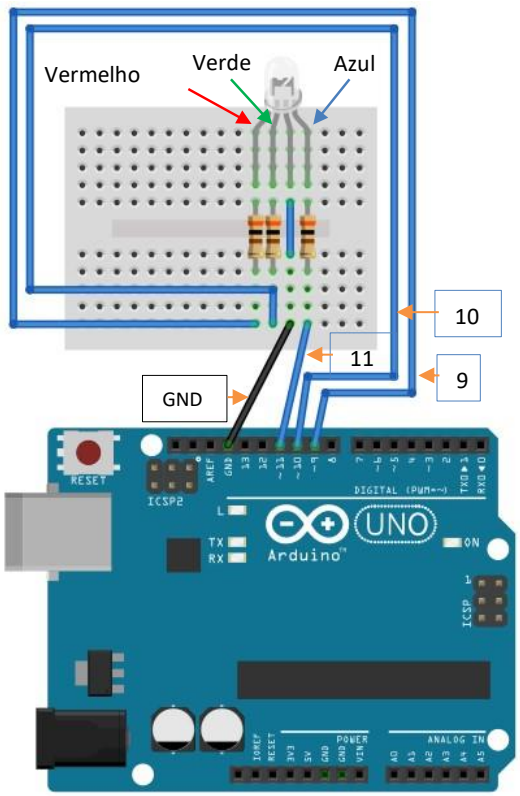
## Material Utilizado

**3 x Resistor 300 Ω**

**1 x Led RGB**

**Fios Conectores**

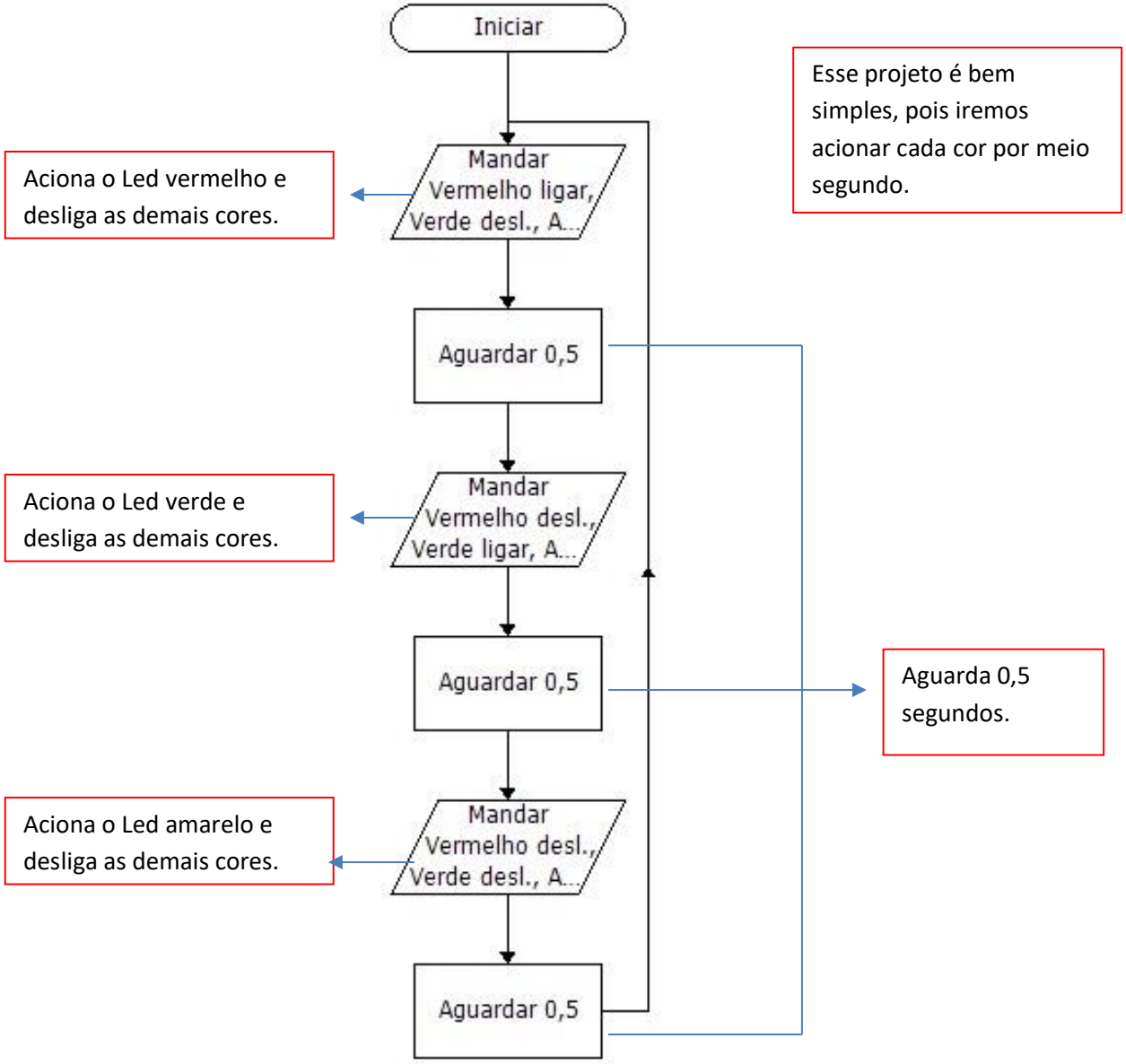
## Ligações Elétrica




**OBS.:** Para confirmar qual cor corresponde a cada pino, basta fazer a ligação eletrônica e pelo software ir acionando a saída 9, depois a 10 e por fim a 11. Com isso o RGB irá ascender com a cor ligada em cada saída.
Só deve ligar uma cor de cada vez (apenas uma saída deve ser acionada por vez).

| | |
|---|---|
| 0 | Saída 0 |
| 1 | Saída 1 |
| 2 | Saída 2 |
| 3 | Saída 3 |
| 4 | Saída 4 |
| 5 | Saída 5 |
| 6 | Saída 6 |
| 7 | Saída 7 |
| 8 | Saída 8 |
| 9 | Vermelho |
| 10 | Verde |
| 11 | Azul |
| 12 | Saída 12 |
| 13 | Saída 13 |

Cada pino do RGB é conectado em uma saída digital.




Esse projeto é bem simples, pois iremos acionar cada cor por meio segundo.

Aciona o Led vermelho e desliga as demais cores.

Aciona o Led verde e desliga as demais cores.

Aciona o Led amarelo e desliga as demais cores.

Aguarda 0,5 segundos.

## 3.6.1 Projeto 1

Desenvolver um projeto que acione cada cor do RGB gradativamente.

### Material Utilizado

**Mesmo material que o projeto anterior.**

### Ligações Elétricas

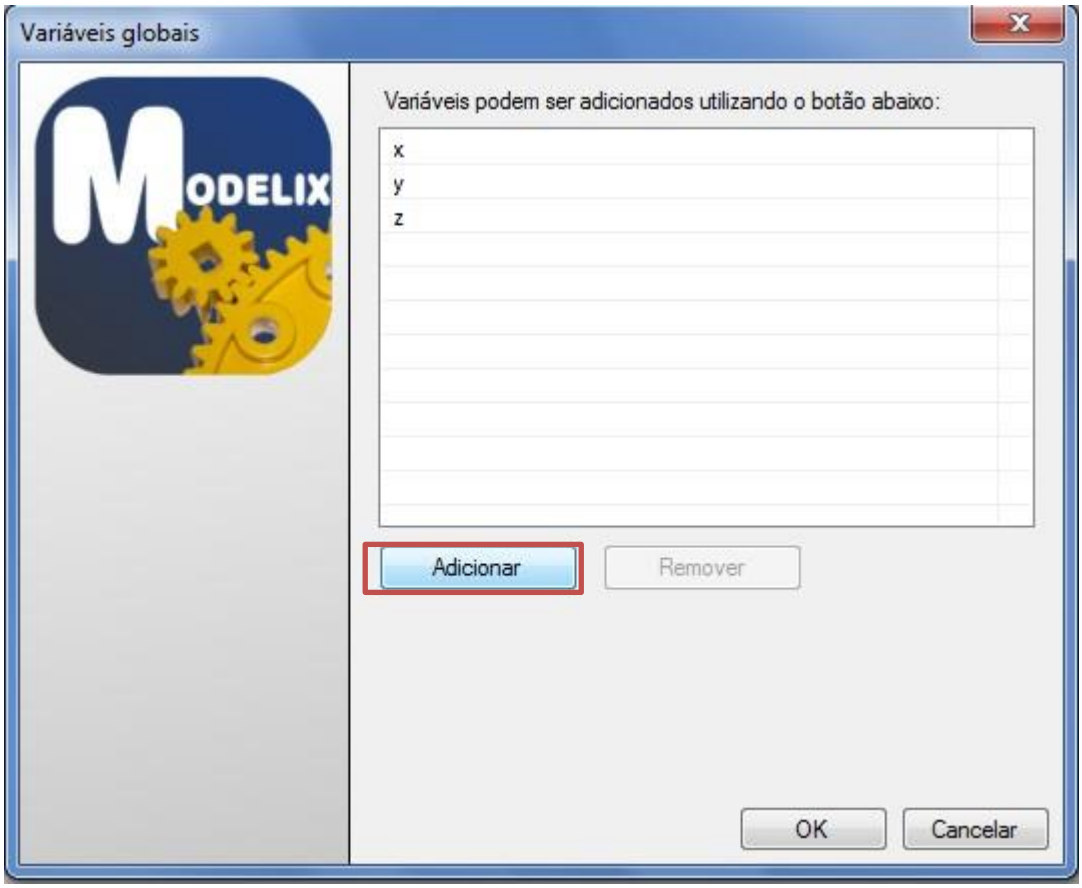
**Mesmas ligações que o projeto anterior.**

Como queremos controlar a intensidade de luminosidade, teremos que utilizar uma saída PWM. Faremos os ajustes nos tipos de saídas.


| | | |
|---|---|---|
| 9 | VermelhoPWM | 0 |
| 10 | VerdePWM | 0 |
| 11 | AzulPWM | 0 |

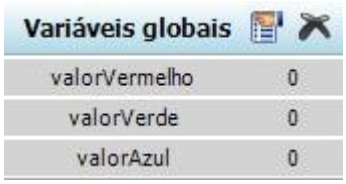
Para a intensidade luminosa alterar automaticamente será necessário a utilização de variáveis. Como vimos anteriormente o botão “XY” cria duas variáveis, porém nesse caso usaremos três. Para adicionar outras seguiremos o procedimento:


Ao clicar no botão “Variáveis” gera por padrão duas variáveis globais.

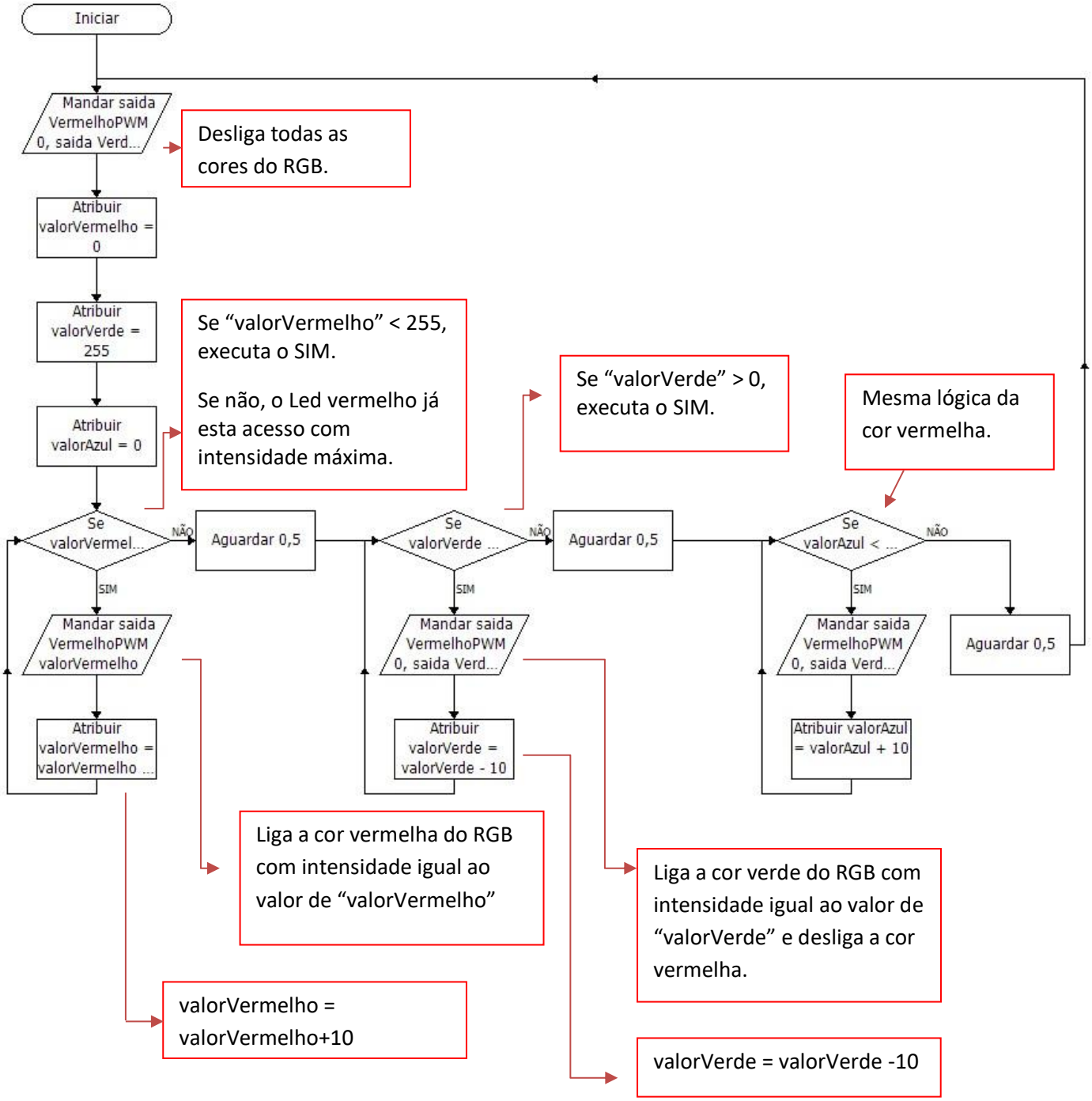





Renomeando as variáveis, temos o seguinte resultado.



Com tudo ajustado faremos a lógica. O primeiro passo é desligar todas as saídas PWM e definir que as variáveis “valorVermelho” e “valorAzul” irão começar zeradas e a “valorVerde” no valor máximo. Ou seja, quando ligarmos o nosso projeto os Leds vermelho e azul irão ascender progressivamente, enquanto o Led verde irá apagar gradualmente.



## 3.6.2 Desafio

Desenvolver um projeto que possibilite a variação das cores do RGB com o potenciômetro.

**Dica 1:** Modifique as ligações eletrônicas para inserir o potenciômetro.

**Dica 2:** O valor máximo lido pelo potenciômetro é 1023.

**Dica 3:** Cada cor deve ser acesa se entrar em um intervalo de valores do potenciômetro.

## Material Utilizado no Desafio

**1 x Led RGB**

**3 x Resistor 300 Ω**

**1 x Potenciômetro 10k Ω**

**Fios Conectores**

## 3.6.3 Exercícios

**3.6.3.1. O LED RGB é uma entrada ou uma saída? Digital ou analógica?**

**3.6.3.2. O que significa RGB?**

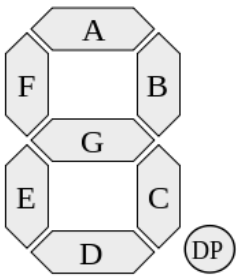
As respostas do desafio e dos exercícios estão na apostila gabarito.

## 3.7 Módulo 7

Desenvolver um projeto que faça a leitura da temperatura ambiente e exiba essa mensagem em um display de 7 segmentos.

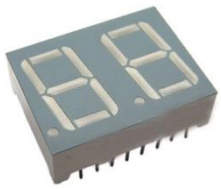
Display de 7 segmentos, nada mais é do que LEDs interligados, sendo cada segmento um LED. Como sabemos, para ascender um LED é preciso fornecer em um terminal sinal positivo e no outro GND. No caso do display de 7 segmentos um dos terminais de todos os LEDs estão conectados juntos, ou seja, ou o GND é comum a todos os segmentos (nesse caso o display é conhecido como catodo comum, pino negativo) logo, para acionar cada segmento é necessário enviar um sinal positivo no outro terminal, ou o pino de VCC é comum a todos (nesse caso o display é conhecido como anodo comum, pino positivo) e para acionar cada LED é necessário enviar GND ao outro terminal. No nosso caso o display utilizado é anodo comum.

Para cada segmento do display é normalmente atribuído uma letra para facilitar sua identificação. Uma representação muito comum é a ilustração abaixo.




### Material Utilizado

**1 x Display de 7 segmentos**



**2 x Resistor 300 Ω**

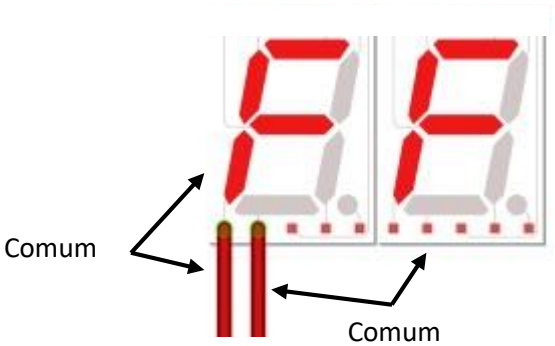
**1 x Sensor LM35**

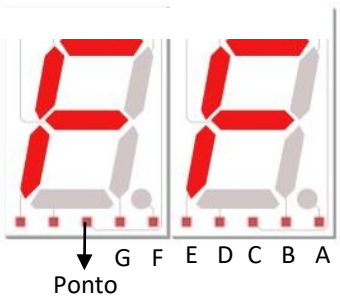
**Fios Conectores**

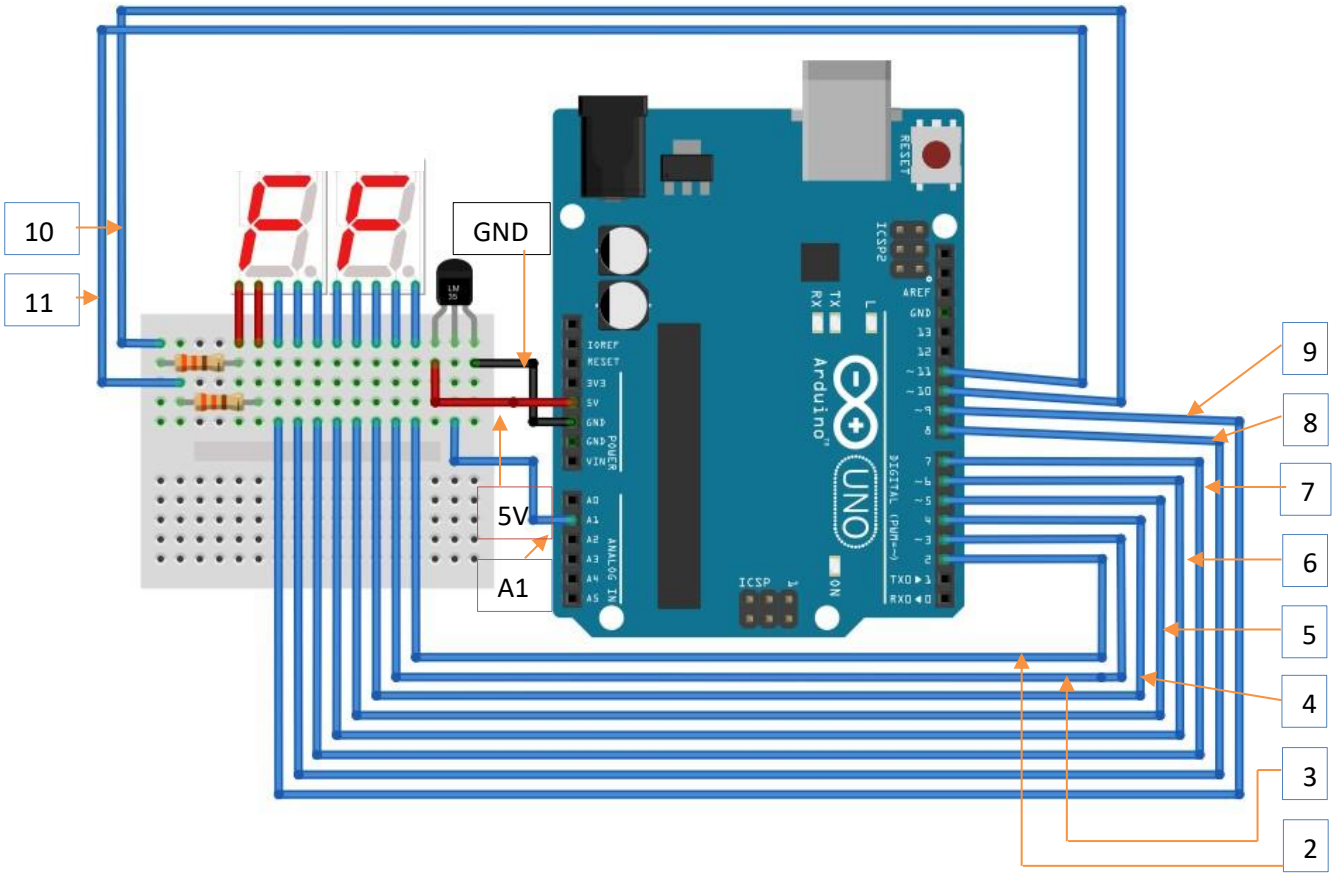
# Ligações Elétrica

O nosso componente possuí dois displays, porém a quantidade de pinos corresponde aos segmentos de apenas um, ou seja, se ligarmos os dois displays juntos eles exibirão o mesmo número. Contudo é possível alterar essa funcionalidade.

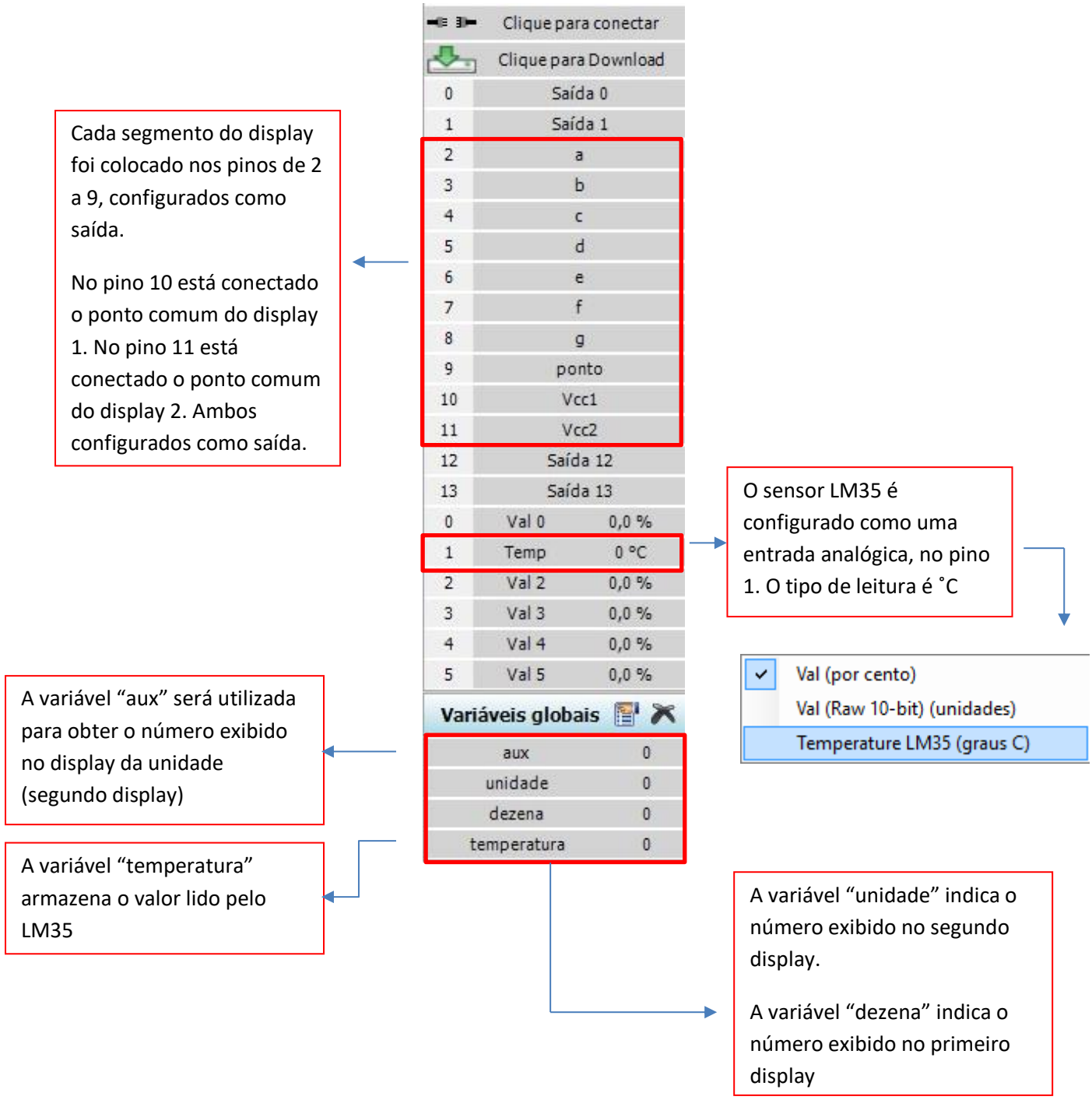
Os dois primeiros pinos são os pontos comum. O primeiro é o ponto comum do primeiro display e o segundo é o ponto comum do segundo display. Como vimos anteriormente para o display ascender é preciso que esse ponto comum esteja energizado, então se desligar o segundo ponto o display dois apaga e a mesma coisa acontece para o primeiro display se desligar o primeiro pino. Por conta desta funcionalidade, os pontos comuns são o controle do display.



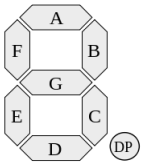



Com tudo conectado já podemos iniciar as configurações do software



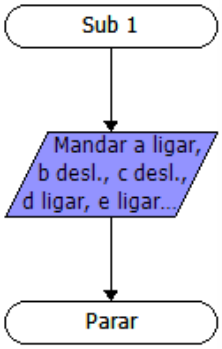
Primeiro vamos fazer a lógica para mostrar os dígitos no display e para isso é fundamental o uso da nomenclatura dos segmentos. O número 1 é representado pelo acionamento dos segmentos B e C e assim sucessivamente. Para facilitar, montamos a tabela abaixo e para cada número criamos uma sub-rotina.

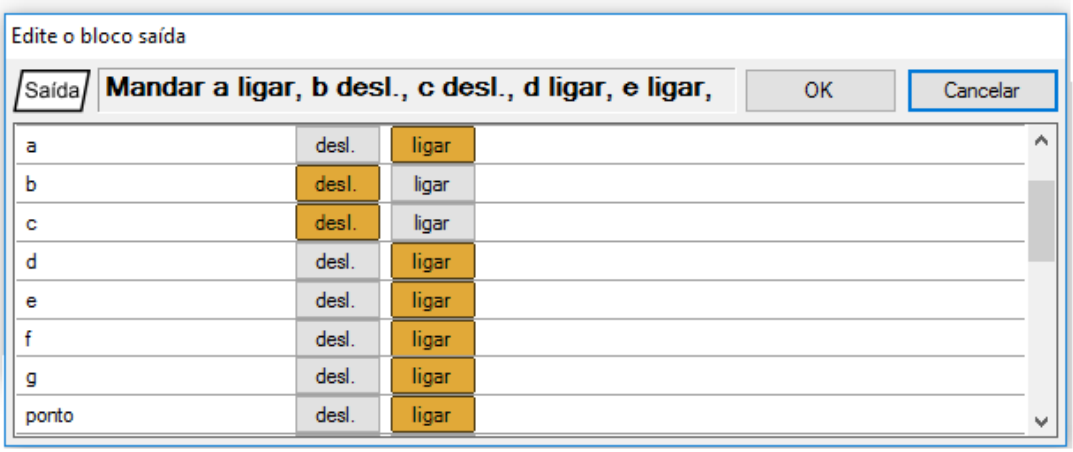
| NÚMERO\SEGMENTO | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 0 | X | X | X | X | X | X | |
| 1 | | X | X | | | | |
| 2 | X | X | | X | X | | X |
| 3 | X | X | X | X | | | X |
| 4 | | X | X | | | X | X |
| 5 | X | | X | X | | X | X |
| 6 | X | | X | X | X | X | X |
| 7 | X | X | X | | | | |
| 8 | X | X | X | X | X | X | X |
| 9 | X | X | X | | | X | X |




**Atenção:** Como falamos anteriormente o display é anodo comum, ou seja, ele funciona ao contrário. Todo segmento que queremos ligar, devemos desliga-los e vice-versa.

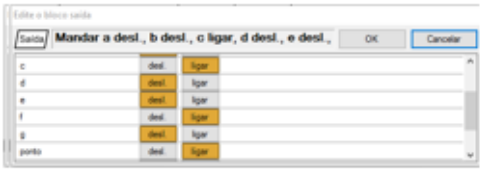
Para exibir cada número no display iremos criar uma sub-rotina. A seguir está a sub-rotina para do número 1.



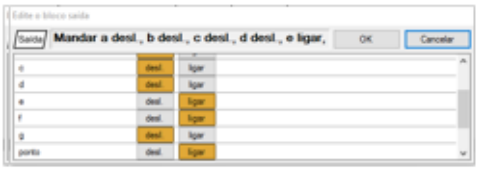




O estado ligado indica que essa saída não será acionada. Logo só serão acessos os LEDs da saída “b” e “c”.

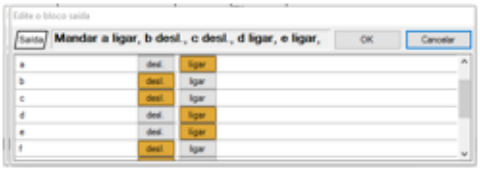
2

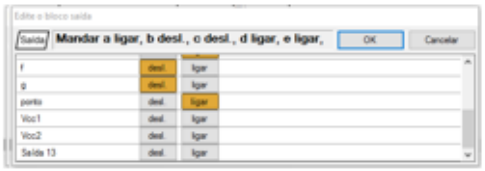

Edite o bloco saída
Saída
Mandar a desl., b desl., c ligar, d desl., e desl.,
OK
Cancelar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar

3

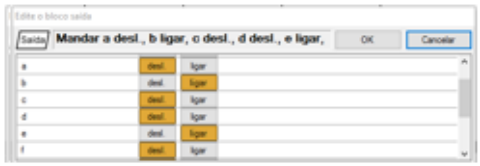

Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d desl., e ligar,
OK
Cancelar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar

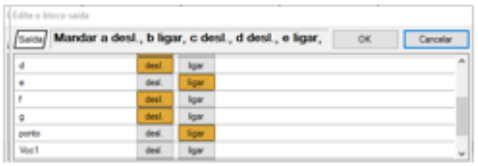
4


Edite o bloco saída
Saída
Mandar a ligar, b desl., c desl., d ligar, e ligar,
OK
Cancelar
a desl. ligar
b desl. ligar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar

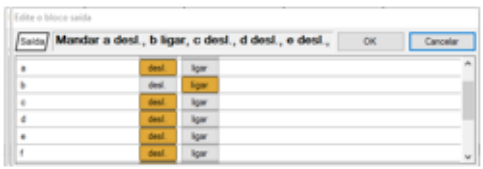

Edite o bloco saída
Saída
Mandar a ligar, b desl., c desl., d ligar, e ligar,
OK
Cancelar
f desl. ligar
g desl. ligar
ponto desl. ligar
Vcc1 desl. ligar
Vcc2 desl. ligar
Saída 13 desl. ligar

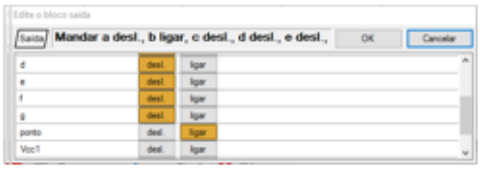
5


Edite o bloco saída
Saída
Mandar a desl., b ligar, c desl., d desl., e ligar,
OK
Cancelar
a desl. ligar
b desl. ligar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar

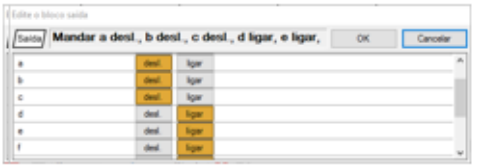

Edite o bloco saída
Saída
Mandar a desl., b ligar, c desl., d desl., e ligar,
OK
Cancelar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar
Vcc1 desl. ligar

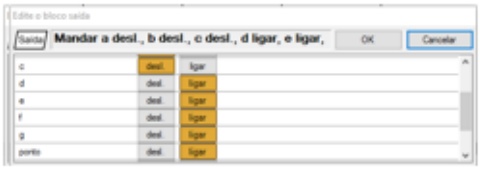
6


Edite o bloco saída
Saída
Mandar a desl., b ligar, c desl., d desl., e desl.,
OK
Cancelar
a desl. ligar
b desl. ligar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar

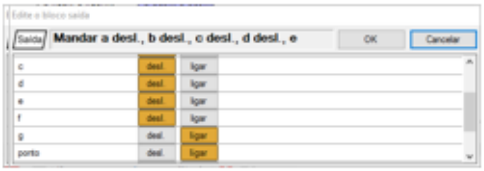

Edite o bloco saída
Saída
Mandar a desl., b ligar, c desl., d desl., e desl.,
OK
Cancelar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar
Vcc1 desl. ligar

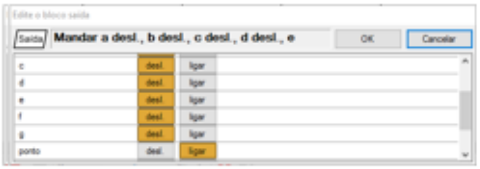
7


Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d ligar, e ligar,
OK
Cancelar
a desl. ligar
b desl. ligar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar

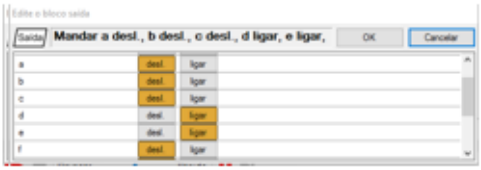

Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d ligar, e ligar,
OK
Cancelar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar

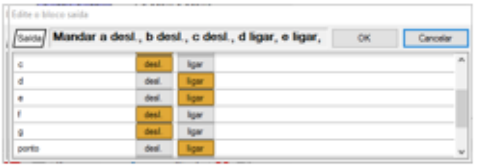
8


Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d desl., e
OK
Cancelar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar

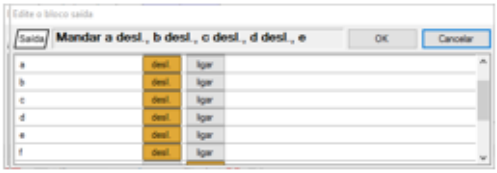

Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d desl., e
OK
Cancelar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar

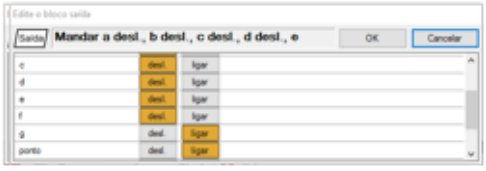
9

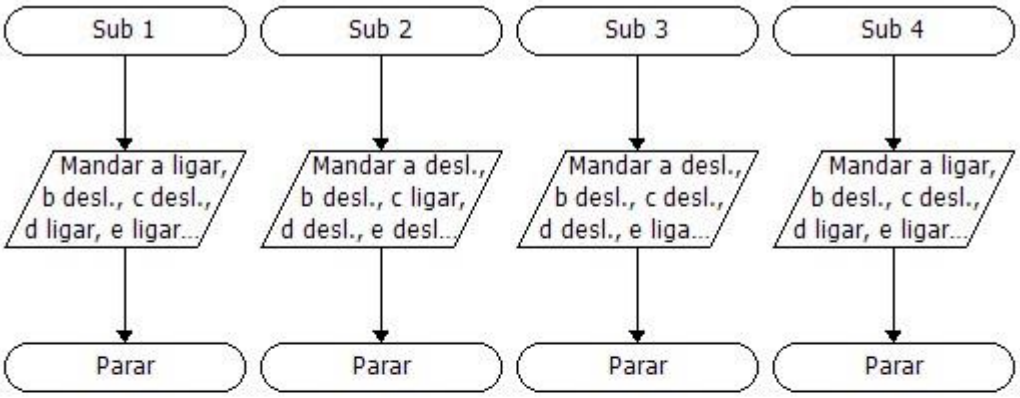

Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d ligar, e ligar,
OK
Cancelar
a desl. ligar
b desl. ligar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar

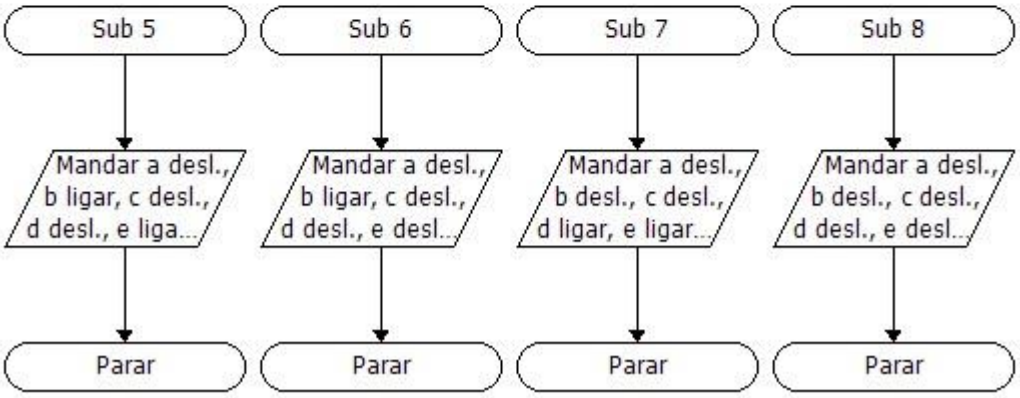

Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d ligar, e ligar,
OK
Cancelar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar

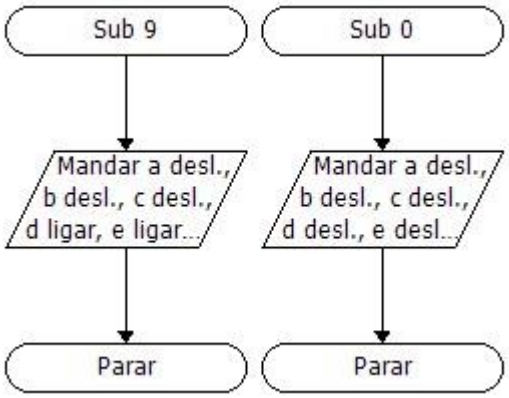
0


Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d desl., e
OK
Cancelar
a desl. ligar
b desl. ligar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar


Edite o bloco saída
Saída
Mandar a desl., b desl., c desl., d desl., e
OK
Cancelar
c desl. ligar
d desl. ligar
e desl. ligar
f desl. ligar
g desl. ligar
ponto desl. ligar

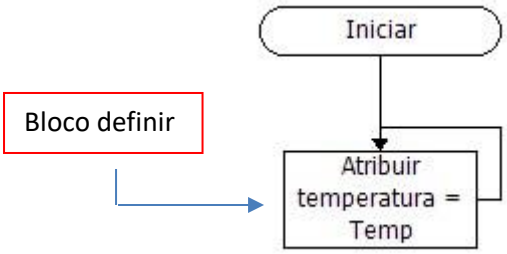


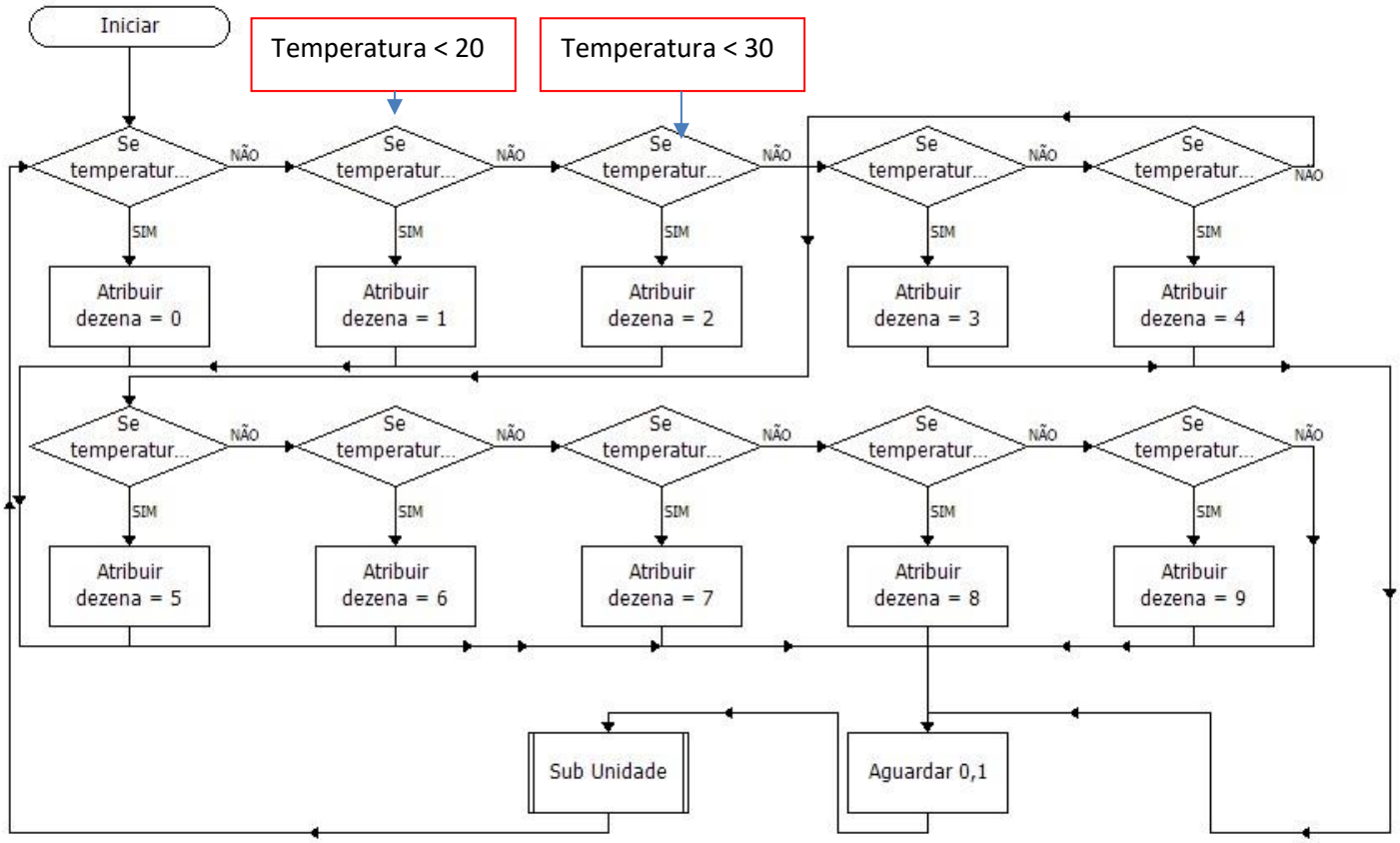

Teste as sub-rotinas para verificar se estão aparecendo todos os números corretamente.

Pronto, agora já temos todos os números para aparecerem no display. Porém temos um problema. Cada display só pode exibir um número, então se o sensor ler o valor 25 teremos que separar o 2 e o 5 para que o display da dezena (primeiro display) exiba o número 2 e o display da unidade exiba o número 5. Para fazer essa separação precisamos, primeiro, do valor lido pelo sensor.




Como queremos que o valor da temperatura seja capturado a todo instante, essa leitura irá ocorrer em paralelo e armazenará o valor na variável “temperatura”.

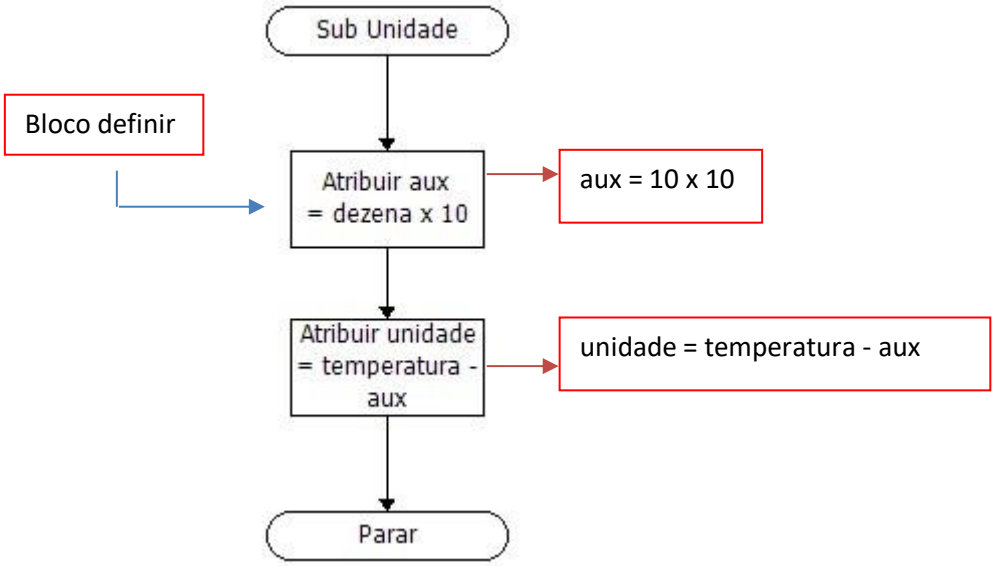
Para separar a dezena e a unidade precisaremos de uma lógica de condições. Primeiro verificaremos se a variável “temperatura” é menor que 10, se for verdadeira a afirmação a dezena vale 0, se “temperatura” for menor que 20, a dezena vale 1, pois não entrou na condição de ser menor que 10, e assim sucessivamente até o último bloco verificar se a temperatura é menor que 100. Então esse número pode variar entre 11 e 19. Seguindo essa lógica fizemos os seguintes blocos:




Após separar a dezena temos que verificar o valor da unidade. Faremos isso na sub-rotina “Unidade”.

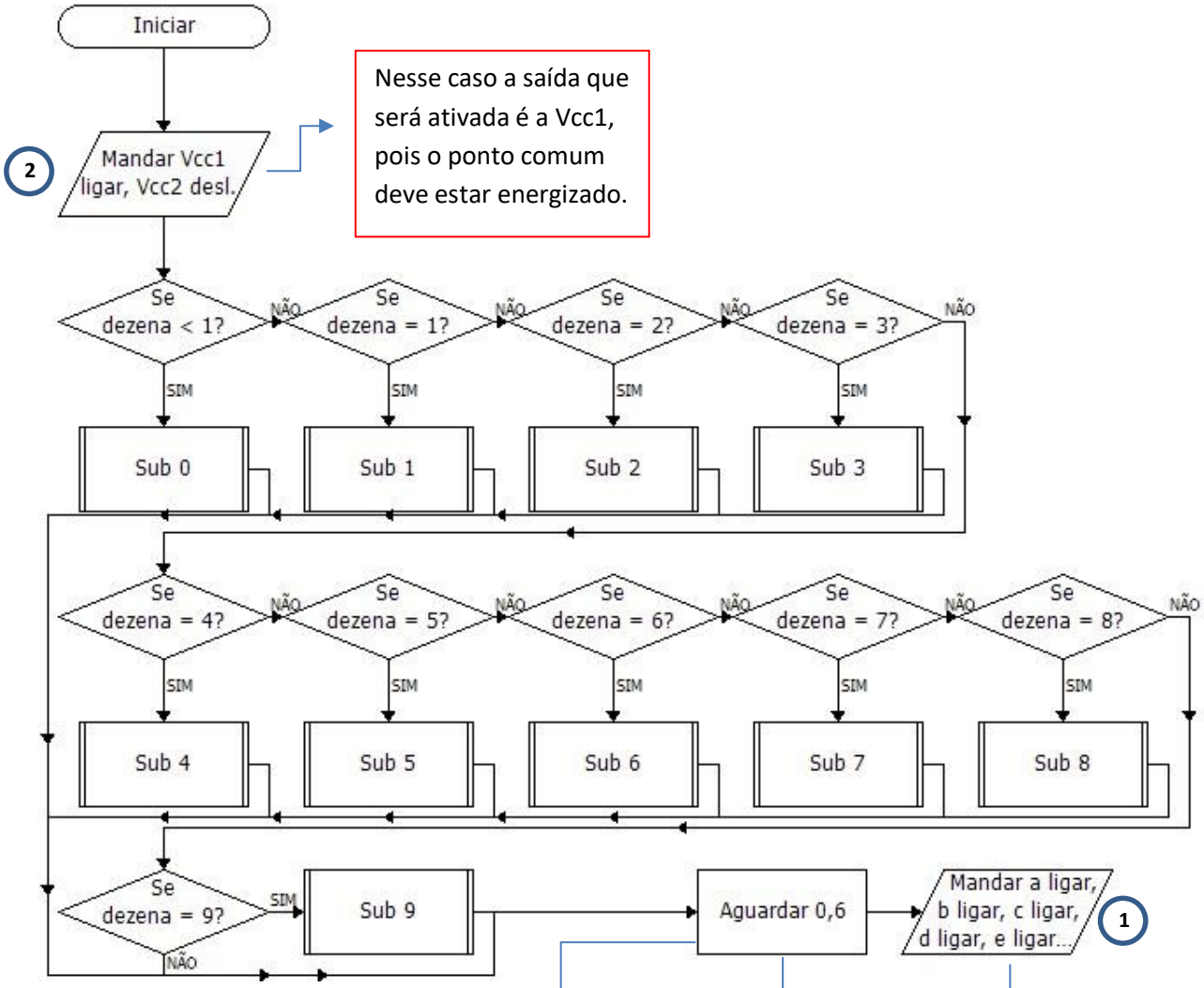
Como temos o valor da dezena é simples pegar o valor da unidade, basta usar a formula a baixo.

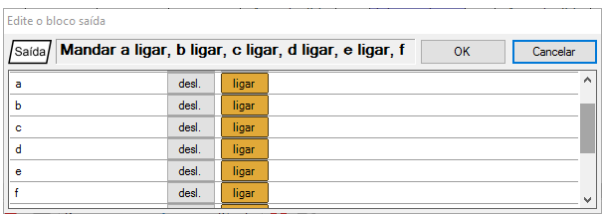
$$valor_{unidade} = valor_{total} - (10 * valor_{dezena})$$

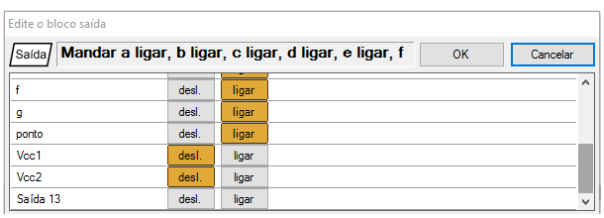


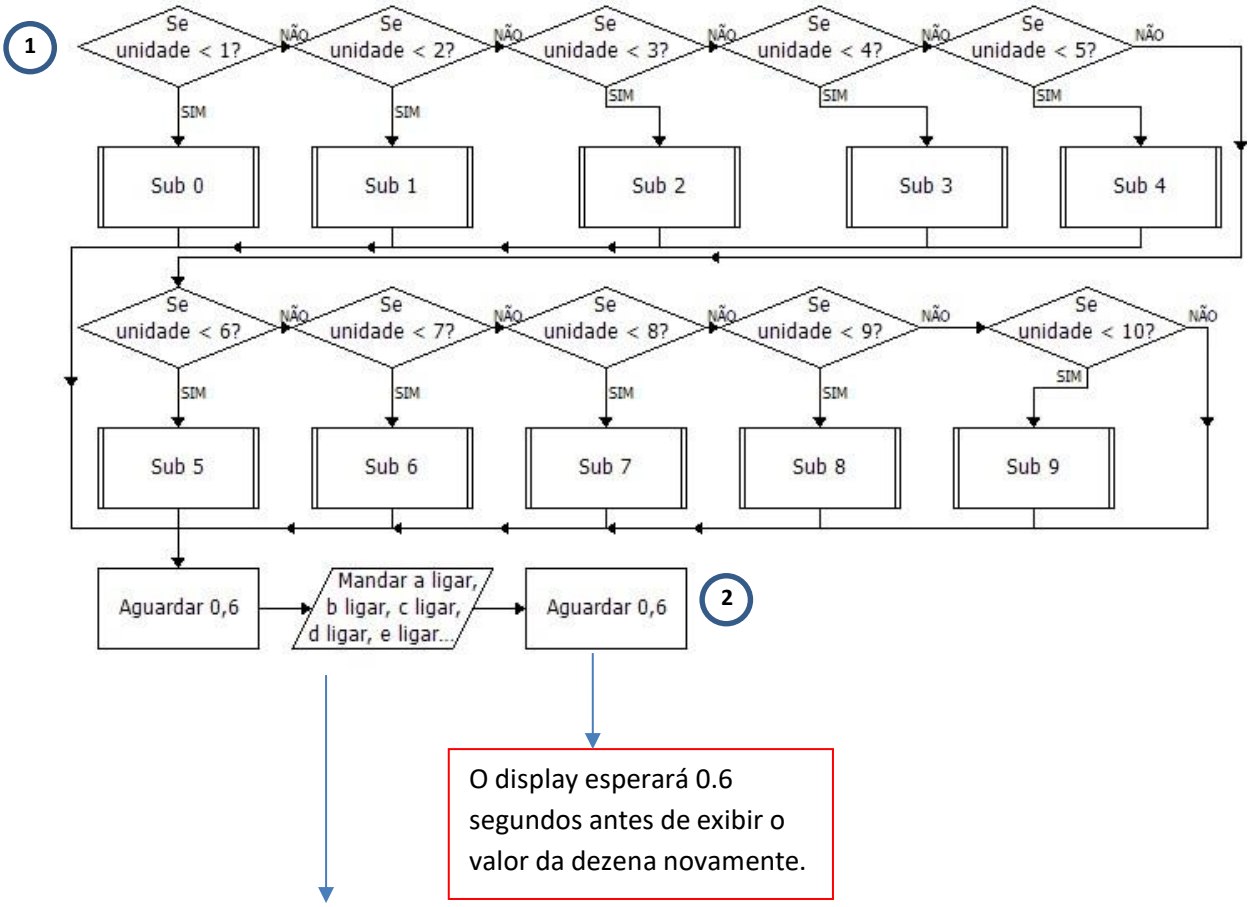

Com isso já temos o valor da unidade e da dezena, então só precisamos exibi–las nos displays. Para exibir o número 2 é necessário chamar a sub-rotina “2”, mas isso só deve acontecer se o valor da unidade ou dezena for 2, portanto temos que verificar qual o valor da dezena/unidade para chamar a sub-rotina correta. Essa verificação é feita com os blocos condicionais.

Como vimos no começo do projeto se ligarmos os dois displays juntos eles exibirão o mesmo número, então temos que ligar um de cada vez. Para isso iremos controlar a alimentação. Primeiro vamos exibir o número que indica dezena, então temos que acionar o primeiro display que possui o ponto comum conectado na entrada 10 (Vcc1) e desligar o segundo.

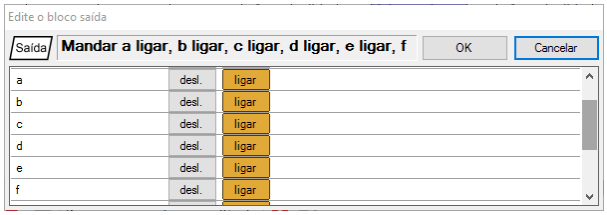

Iniciar
2
Mandar Vcc1 ligar, Vcc2 desl.
Nesse caso a saída que será ativada é a Vcc1, pois o ponto comum deve estar energizado.
Se dezena < 1?
Se dezena = 1?
Se dezena = 2?
Se dezena = 3?
Se dezena = 4?
Se dezena = 5?
Se dezena = 6?
Se dezena = 7?
Se dezena = 8?
Se dezena = 9?
NÃO
SIM
Sub 0
Sub 1
Sub 2
Sub 3
Sub 4
Sub 5
Sub 6
Sub 7
Sub 8
Sub 9
Aguardar 0,6
Mandar a ligar, b ligar, c ligar, d ligar, e ligar...
1
Desliga o Vcc1 e liga o Vcc2 para ativar o segundo display.
Tempo que o número será exibido no display.
Desliga todos os segmentos para exibir o valor da unidade.

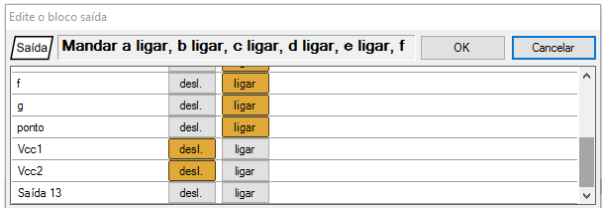

Edite o bloco saída
Saída
Mandar a ligar, b ligar, c ligar, d ligar, e ligar, f
OK
Cancelar
a
b
c
d
e
f
desl.
ligar


Edite o bloco saída
Saída
Mandar a ligar, b ligar, c ligar, d ligar, e ligar, f
OK
Cancelar
f
g
ponto
Vcc1
Vcc2
Saída 13
desl.
ligar




O display esperará 0.6 segundos antes de exibir o valor da dezena novamente.




A lógica para exibir a unidade ou a dezena é muito semelhante e para juntar as duas partes do fluxograma basta ligar os pontos indicados com (1) e (2).

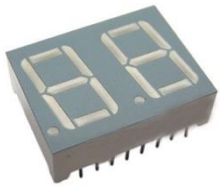
A primeira etapa para exibir a unidade é verificar se o número é menor que 1, caso seja a unidade irá valer 0. Caso o valor seja menor que 2, significa que a unidade vale 1, pois pode variar entre 1 até 1.9999. Usamos esse pensamento para todos os números.

## 3.7.1 Desafio

Desenvolver um contador de 0 até 99. O incremento será através do botão. Quando chegar em 99, volta para zero.

### Material Utilizado no Desafio

**1 x Display de 7 segmentos**



**2 x Resistor 300 Ω**

**1 x Resistor 10k Ω**

**1 x PushButton**

**Fios Conectores**

## 3.7.2 Exercícios

**3.7.2.1. O display de 7 segmentos é configurado como uma entrada ou saída no software Modelix?**

**3.7.2.2. O display usado é um anodo comum ou catodo comum?**

As respostas do desafio e dos exercícios estão na apostila gabarito.

## 3.8 Módulo 8

Desenvolver um projeto que conforme muda a intensidade luminosa do ambiente o motor gira em sentido horário ou anti-horário.

## Material Utilizado

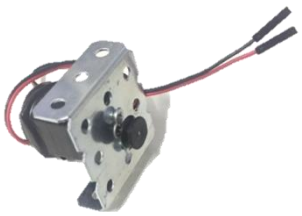
**1 x Sensor LDR**

**1 x Resistor 10k Ω**

**1 x Circuito HUB**


**2 x Circuito Relé Modelix**


**1 x Motor de Giro**



**Fios Conectores**

## Ligações Elétrica

Relé é um componente eletrônico que funciona como um interruptor, ou seja, possuí pinos de entradas que controlam os pinos de saída. Outra característica dos relés é o fato de possuírem duas saídas. Uma saída é conhecida como normal aberta (NA) e a outra normal fechada (NF). A saída NA funciona como uma lâmpada. Quando os pinos de entrada recebem

um sinal, a saída NA é ativada e quando desativa o sinal da entrada, desliga a saída, do mesmo modo que as lâmpadas e os interruptores. Já a saída NF tem o funcionamento contrário ao NA. Quando os pinos de entrada recebem um sinal a saída NF é desativada e quando a entrada para de receber sinal, a saída NF é ativada. Para o relé funcionar é necessário que o mesmo esteja energizado.

A Modelix desenvolveu duas placas para facilitar o uso dos reles. A primeira foi o Circuito Rele v.1.1 e a mais recente, o Circuito Rele v.3.7. A diferença da nova versão é o fato de possuir alimentação na própria placa, sem a necessidade de ligar o circuito relé à outra placa de alimentação. Além disso a entrada do sinal é feita através dos pinos INT e a saída nos pinos de NA e NF, simplificando sua utilização.

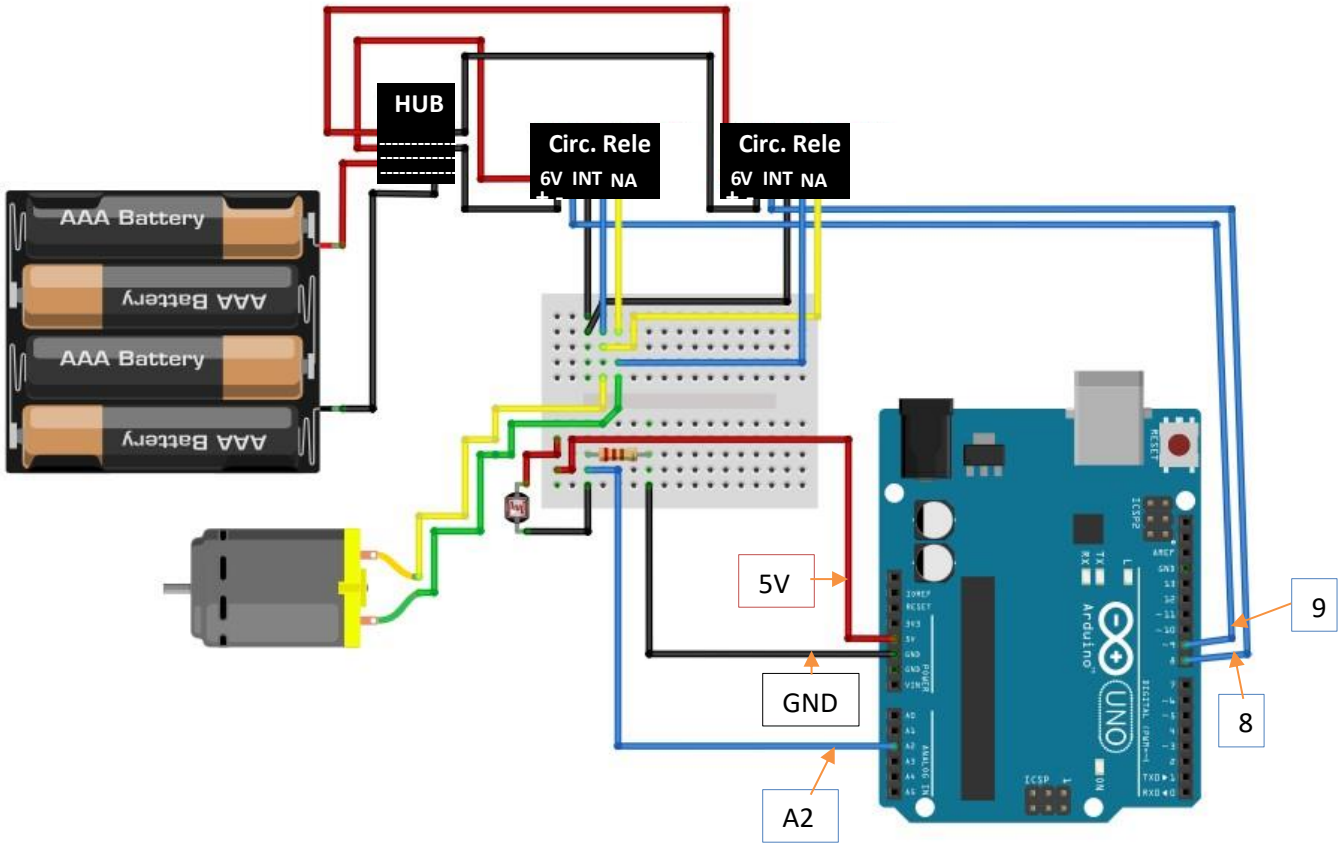

Circuito Rele v.1.1


Circuito Rele v.3.7

Outra placa Modelix que utilizaremos é o Circuito HUB. Essa placa funciona como um Protoboard, em que suas saídas verticais estão conectadas. Essa placa é muito utilizada para fazer alimentações externas em motor.


Circuito HUB

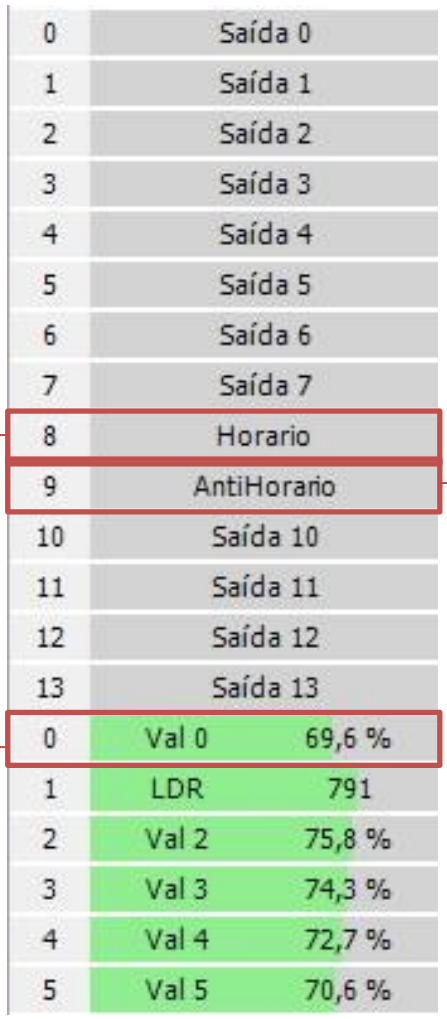

Ligação do Circuito HUB




Como podemos perceber, o HUB possibilita que se utilize menos pinos no protoboard.

O Circuito Relé funciona como uma chave e usaremos ele para fazer o controle do sentido de rotação do motor. Para acionar um LED, como já falamos anteriormente, é necessário que um de seus terminais receba um sinal Vcc e o outro receba um sinal Vcc e o outro receba GND. O motor funciona da mesma maneira, a diferença é que quando inverte o terminal que recebe o GND com o que com o que recebe Vcc o motor gira no sentido contrário.

A primeira etapa para esse controle é conectar as saídas do motor com as saídas NA dos circuitos reles, com o auxílio do protoboard. Desse modo o motor é controlado pelos dois reles, pois enquanto o circuito 1 estiver acionado, é ele que controla o motor e quando o circuito 2 ativar, o motor gira em sentido contrário.

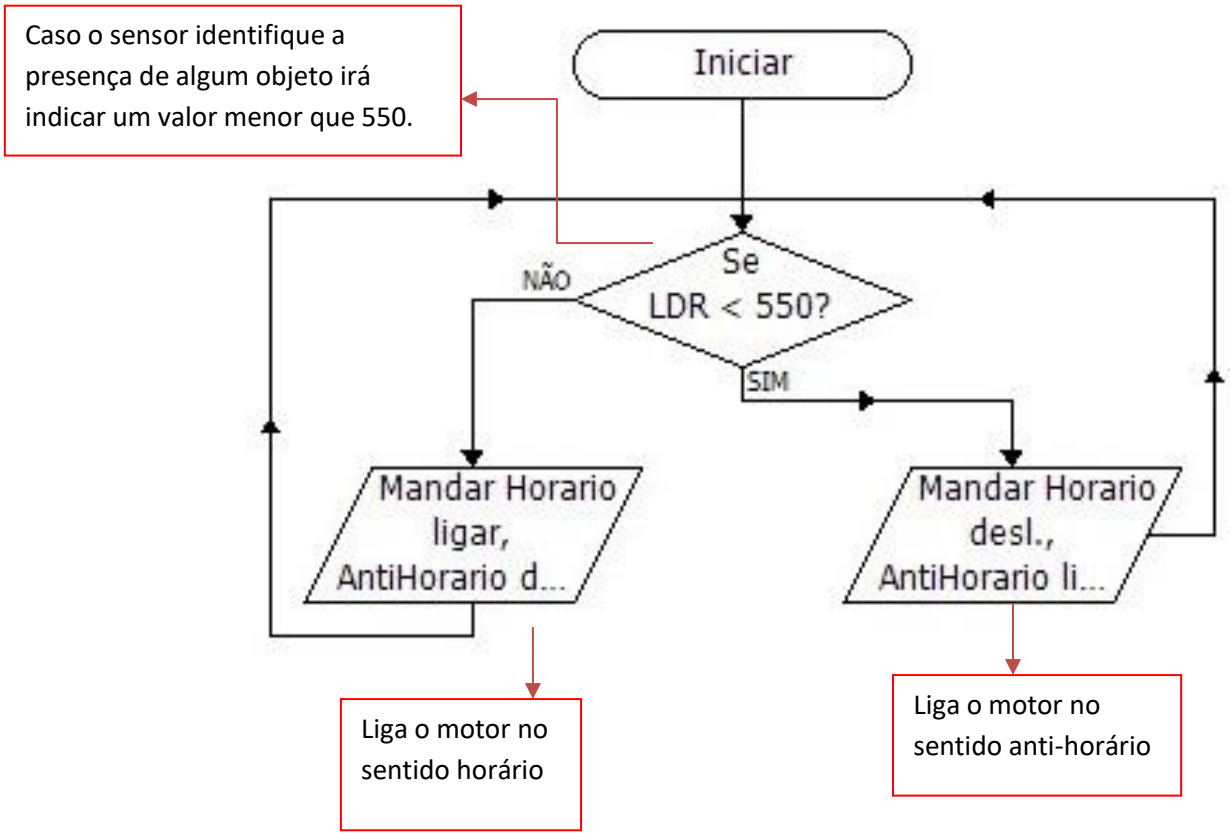



O Circuito Rele responsável pela rotação no sentido horário foi conectado no pino 8. Configurado como saída digital.

O Circuito Rele responsável pela rotação no sentido anti-horário foi conectado no pino 9. Configurado como saída digital.

O LDR é configurado como entrada analógica no pino1.

**OBS.:** Antes de começarmos a programação faremos testes com o microcontrolador conectado no software. O primeiro teste é verificar se o motor está girando no sentido correto quando ligamos as saídas 8 e 9. Se não estiver, basta inverter as ligações da saída NA com o motor do circuito relé que estiver incorreto.




### 3.8.1 Desafio 1

Desenvolver um projeto que ao pressionar o botão o motor gira em sentido horário, pressionando novamente, gira em sentido anti-horário.

## Material Utilizado no Desafio 1

**1x PushButton**

**1 x Resistor 10k Ω**

**1 x Circuito HUB**


**2 x Circuito Relé Modelix**


**1 x Motor de Giro**

**Fios Conectores**

## 3.8.2 Desafio 2

Desenvolver um projeto que girando o potenciômetro para um dos lados o motor deve girar no sentido horário. Girando para o outro lado, o motor gira em sentido anti-horário.

**Dica 1:** Siga a mesma lógica do exercício anterior.

**Dica 2:** Quando giramos o potenciômetro o seu valor aumenta ou diminui, então podemos identificar o sentido de rotação utilizando isso.

## Material Utilizado no Desafio 2

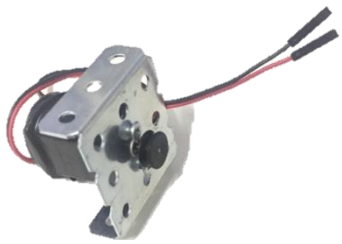
**1x Potênciometro de 10k Ω**

**1 x Resistor 10k Ω**

**1 x Circuito HUB**


**2 x Circuito Relé Modelix**


**1 x Motor de Giro**



**Fios Conectores**

## 3.8.3 Exercícios

**3.8.3.1. Por que o circuito rele é necessário?**

**3.8.3.2. Quais pinos do circuito relé são entradas digitais e quais são saídas digitais?**

As respostas do desafio e dos exercícios estão na apostila gabarito.

## 3.9 Módulo 9

Desenvolver um projeto que quando o sensor detectar um imã ele acione o LED.

## Material Utilizado

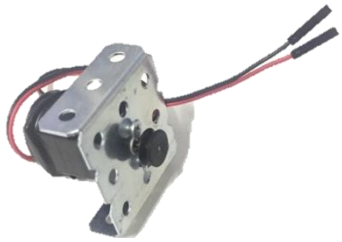
**1 x Resistor 10k Ω**

**1 x Resistor 300 Ω**

**1 x LED**

**1 x Sensor Magnético**

**1 x Motor de Giro**

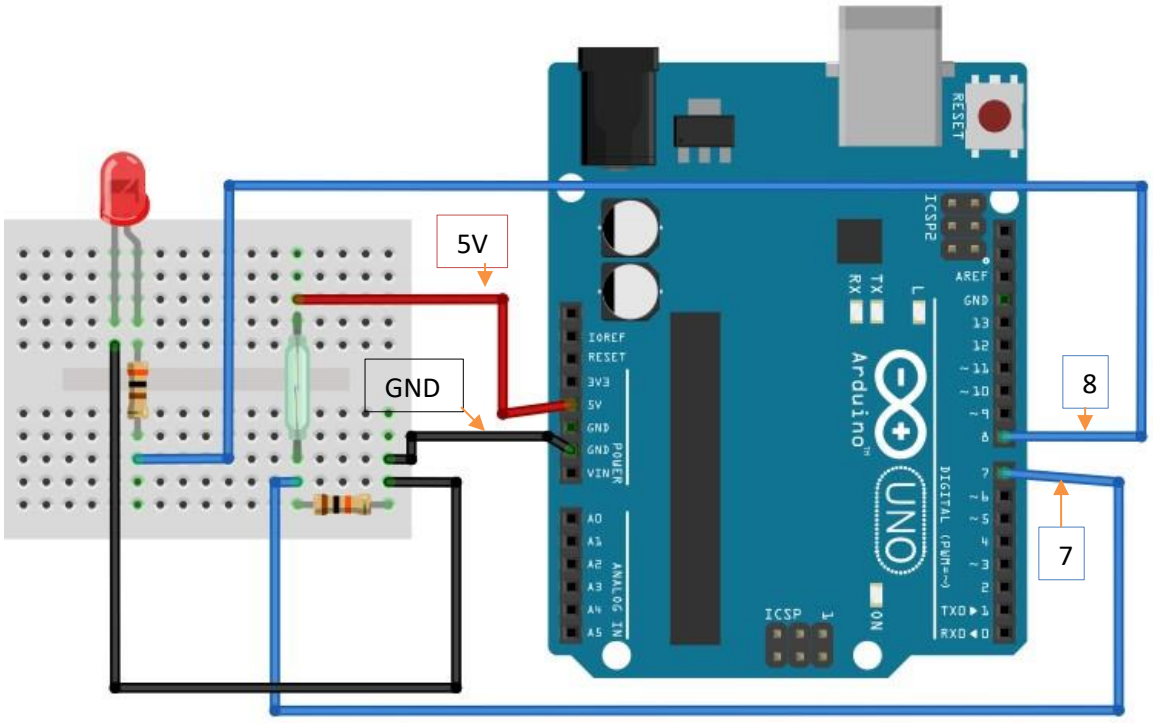


**Fios Conectores**

## Ligações Elétrica

Sensor Reed Switch ou sensor Magnético tem um funcionamento semelhante às chaves PushButton, ou seja, são interruptores. A grande diferença é que enquanto as chaves são acionadas pressionando sua alavanca os sensores magnéticos são acionados por campo magnético gerado por imãs.

O circuito eletrônico desse sistema é bem simples, muito semelhante aos esquemas de botões.



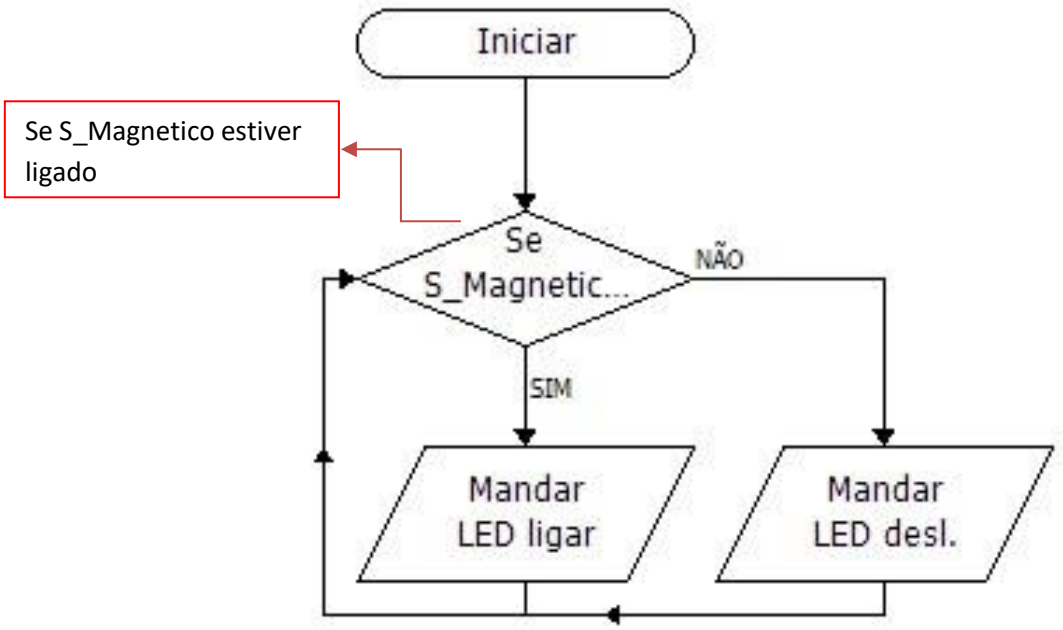

No software o sensor é tratado igual a uma chave normal, ou seja, é configurado como uma entrada (pino 8). A lógica do fluxograma é muito simples, basta verificar se a o sensor está acionado, se tiver deverá acender o LED e caso contrário, deverá apaga-lo.

Como nosso kit não inclui um imã, usaremos o corpo do motor de giro para ativar nosso sensor magnético.

| | |
|---|---|
| 0 | Saída 0 |
| 1 | Saída 1 |
| 2 | Saída 2 |
| 3 | Saída 3 |
| 4 | Saída 4 |
| 5 | Saída 5 |
| 6 | Saída 6 |
| 7 | LED |
| 8 | S_Magnetico |
| 9 | Saída 9 |
| 10 | Saída 10 |
| 11 | Saída 11 |
| 12 | Saída 12 |
| 13 | Saída 13 |

O LED está conectado no pino digital 7.

Sensor magnético está conectado como entrada no pino digital 8.




Se S_Magnetico estiver ligado

## 3.9.1. Desafio

Desenvolver um projeto que quando o sensor detectar um imã ele acione o buzzer duas vezes.

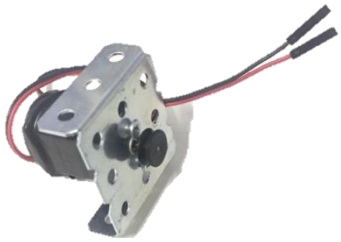
### Material Utilizado no Desafio

**1 x Resistor 10k Ω**

**1 x Buzzer**

**1 x Sensor Magnético**

**1 x Motor de Giro**



**Fios Conectores**

## 3.9.2. Exercícios

**3.9.2.1. O sensor magnético é uma entrada ou saída? Digital ou analógica?**

**3.9.2.2. O sensor magnético tem a funcionalidade muito semelhante a outro componente já estudado. Qual é esse componente?**

As respostas do desafio e dos exercícios estão na apostila gabarito.

## 3.10 Módulo 10

Desenvolver um projeto que funcione como uma senha de porta. Quando baterem três vezes na porta essa irá ser destravada e acionará o LED. Se o intervalo entre as batidas for muito grande, será considerado senha errada e tocará o bip.

## Material Utilizado

**1 x Resistor 300 Ω**

**1 x Resistor 1 MΩ**

**1 x Buzzer**

**1 x LED**

**1 x Sensor Piezoelétrico**

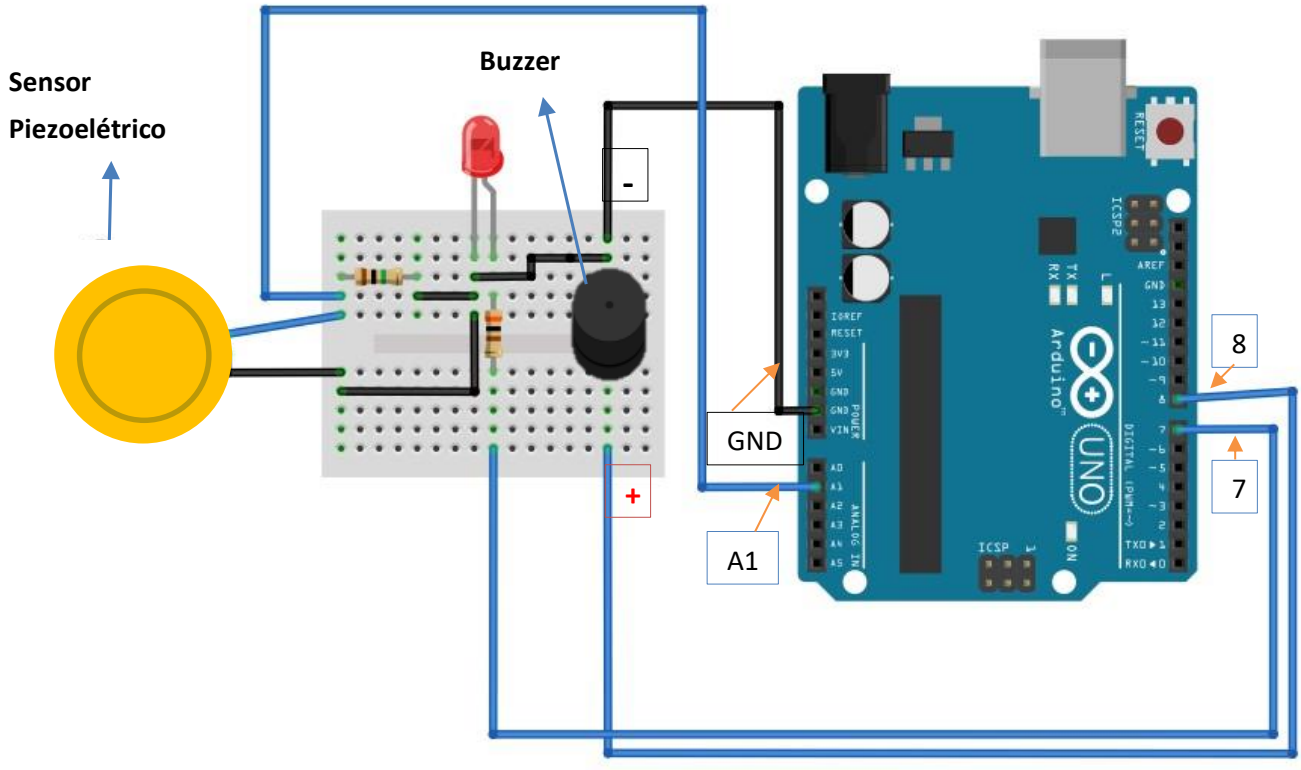
**Fios Conectores**

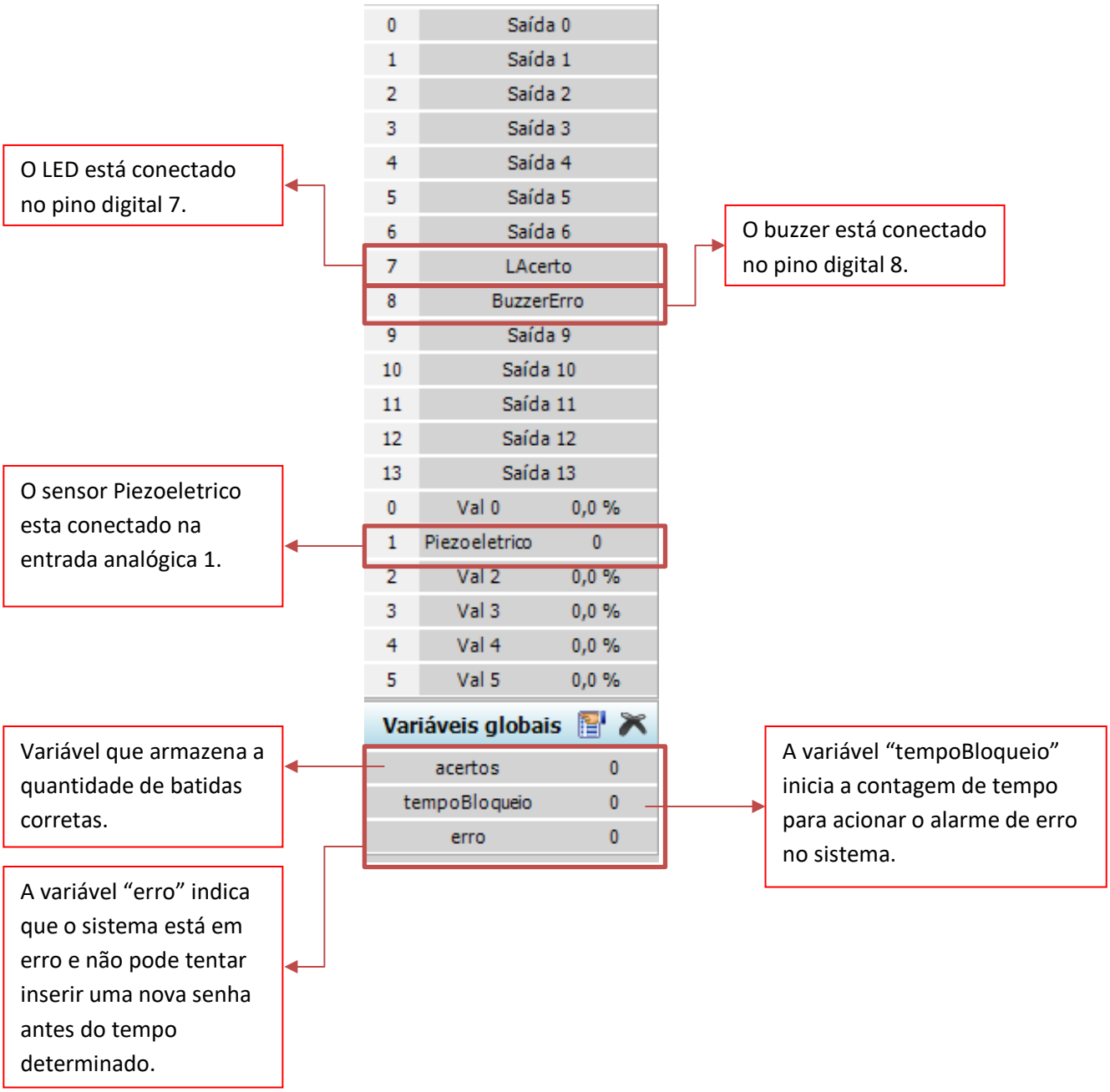
# Ligações Elétrica

Sensor Piezoelétrico é um dispositivo que utiliza o efeito da piezeletricidade para medir pressão ou tensão. Esse efeito ocorre quando o material analisado pode gerar uma corrente elétrica. Nesse caso consegue medir vibrações.

Para simularmos o ambiente descrito no objetivo, usaremos um LED sendo a porta, um buzzer indicando senha incorreta e o sensor piezoelétrico para fazer as medições.

**ATENÇÃO:** O Piezoelétrico deve ser fixado em uma superfície oca para conseguir fazer as medições.







Para ativar a variável, basta clicar em XY no menu superior direito.

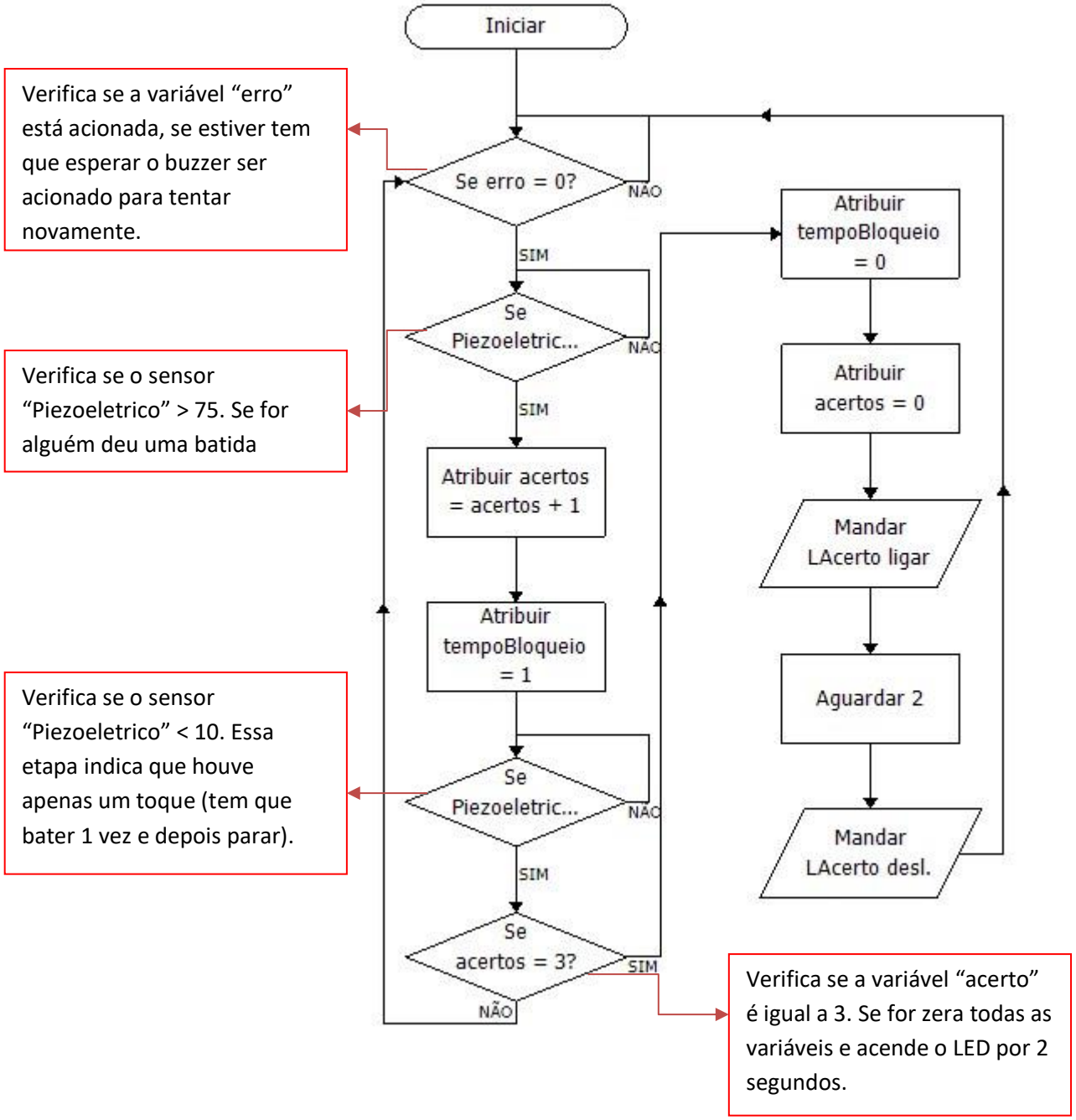
Vamos considerar que a senha para abrir a porta é bater três vezes rápido. Se o intervalo entre cada batida for muito longo, deve–se considerar senha errada e acionar o buzzer. Enquanto o buzzer estiver acionado não pode tentar inserir a senha novamente.

A primeira etapa é verificar a leitura do sensor piezoelétrico. Quando o sensor não recebe nenhum estimulo ele envia o valor 0 e caso receba estimulo, envia o valor 100. Para garantir que sempre capturaremos os estímulos mesmo que aja variação nesse valor, vamos considerar que quando o piezoelétrico captura o valor 75, já é uma batida.

Quando for detectado as vibrações (batidas) iremos incrementar uma variável de acertos (responsável por contar os três toques – "acertos"). E vamos acionar o tempo para indicar erro ("tempoBloqueio"). Quando for acionada essa variável teremos 1,3 segundos para inserir a senha corretamente.

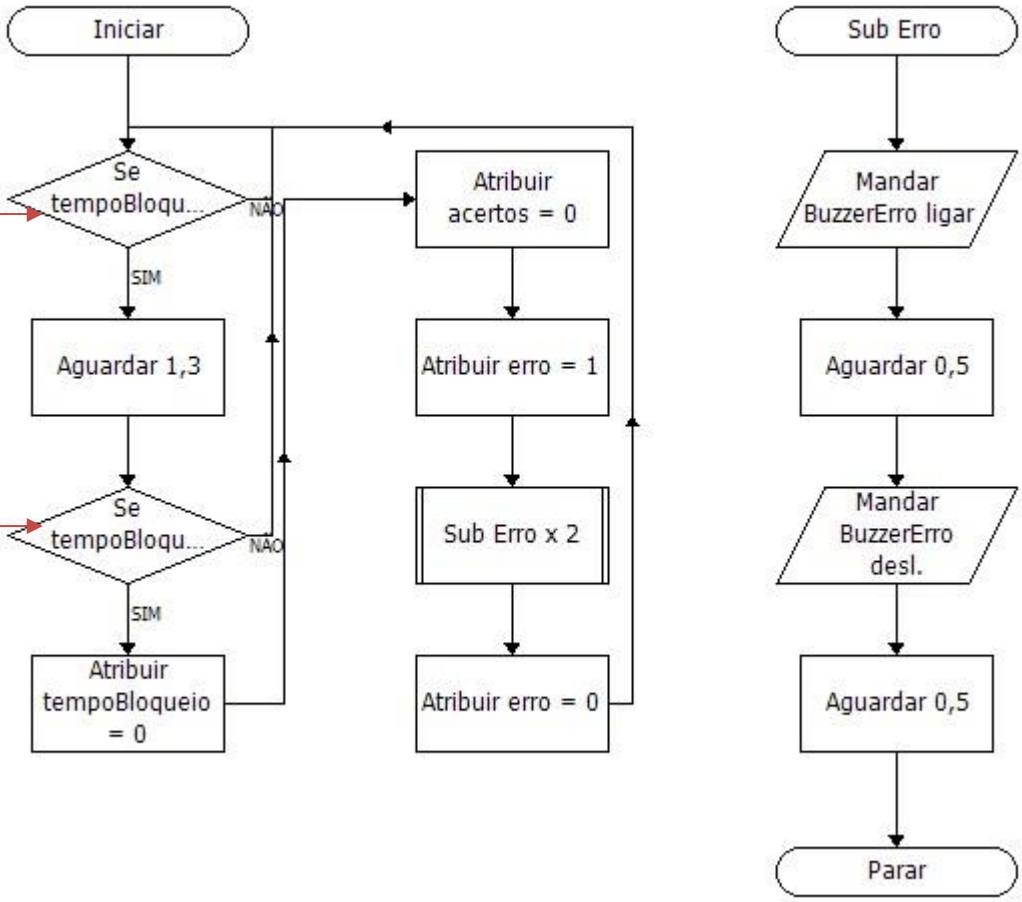
A próxima etapa é verificar se a contagem já chegou em 3, caso tenha chegado acionamos o LED, zeramos as variáveis e acionamos um tempo para recomeçar a lógica. Caso não seja, devemos voltar a verificar o sensor.

Para executar todas essas etapas serão utilizados dois programas em paralelo (dois iniciar) e uma sub-rotina. O primeiro fluxograma é a rotina de verificar os acertos. O segundo é responsável por indicar senha incorreta.


Iniciar
Se erro = 0?
NAO
SIM
Se Piezoeletric...
NAO
SIM
Atribuir acertos = acertos + 1
Atribuir tempoBloqueio = 1
Se Piezoeletric...
NAO
SIM
Se acertos = 3?
SIM
NÃO
Atribuir tempoBloqueio = 0
Atribuir acertos = 0
Mandar LAcerto ligar
Aguardar 2
Mandar LAcerto desl.
Verifica se a variável “erro” está acionada, se estiver tem que esperar o buzzer ser acionado para tentar novamente.
Verifica se o sensor “Piezoeletrico” > 75. Se for alguém deu uma batida
Verifica se o sensor “Piezoeletrico” < 10. Essa etapa indica que houve apenas um toque (tem que bater 1 vez e depois parar).
Verifica se a variável “acerto” é igual a 3. Se for zera todas as variáveis e acende o LED por 2 segundos.

tempoBloquei = 1
Se for verdadeira a condição, inicia a sequência de erro.

tempoBloquei = 1
Essa verificação é feita logo que se encerra o 1,3 segundos. Se essa variável ainda for igual a 1, a pessoa errou a senha. Caso contrário a senha está correta.



## 3.10.1 Desafio

Desenvolver um projeto em que quando o sensor Piezoelétrico detectar uma batida deve acender a cor verde, se detectar duas batidas acende a cor azul e três batidas a vermelho.

### Material Utilizado no Desafio

**3 x Resistor 300 Ω**

**1 x Resistor 1 MΩ**

**1 x LED RGB**

**1 x Sensor Piezoelétrico**

**Fios Conectores**

## 3.10.2. Exercícios

**3.10.2.1. O sensor piezoelétrico é conectado numa entrada ou saída? Digital ou analógica?**

As respostas do desafio e dos exercícios estão na apostila gabarito.