Programação de Sistemas Supervisórios

Aula 05 - Aplicação no ScadaBR

Apresentação

Olá, seja bem-vindo(a) à Aula 05! Epa! A última desta disciplina! 😁 Parabéns pelo seu empenho até aqui, mas espera... ainda há muito o que você aprender.

Nessa aula, você irá colocar em prática todos os conhecimentos adquiridos nas aulas anteriores. Vai compreender como usar o protocolo de rede *modbus* e aprenderá um pouco sobre o uso da plataforma microcontrolada do Arduino, desenvolvendo um sistema de automação na plataforma e fazendo sua integração ao *ScadaBR*.

Objetivos

> Integrar o *ScadaBR* à plataforma microcontrolada;
>
> Compreender como fazer uso do protocolo de rede *modbus*;
>
> Conhecer a plataforma microcontrolada do Arduino.

Componentes Eletrônicos

Para aprender a integrar o *ScadaBR* à plataforma microcontrolada e também como montar um sistema de automação simples, você precisará conhecer alguns componentes novos e relembrar outros.

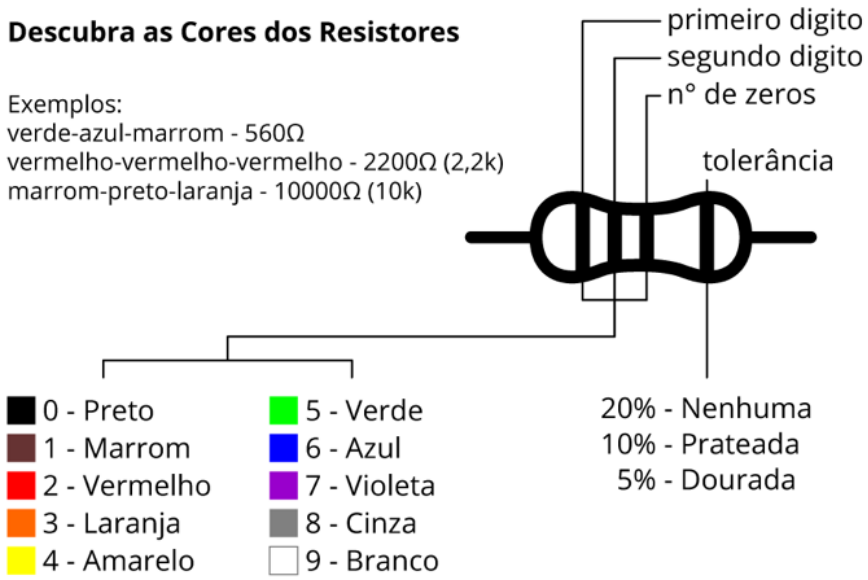
São os componentes que fazem parte de qualquer circuito elétrico ou eletrônico e que estão interligados entre si, isto é, eles são a estrutura de um circuito.

Conheça agora alguns desses componentes abaixo e para que eles servem.

Resistor

O resistor você já conhece das aulas de circuitos eletrônicos e sabe que ele serve para limitar a corrente elétrica que passa pelo circuito. Para limitar mais ou menos corrente, o valor desse componente pode variar. Para saber o valor de cada resistor você pode medir sua resistência com o auxílio do multímetro e consultar a tabela mostrada na Figura 01.

Figura 01 - Tabela de resistências por cores



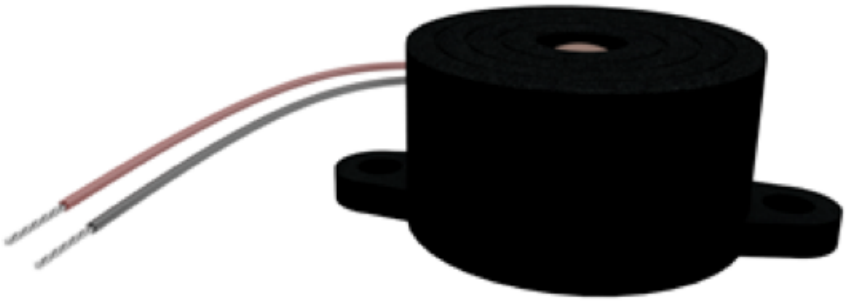
Fonte: ROBOCORE (S.d.).

No experimento dessa aula você utilizará somente dois tipos de resistores, o de 330 Ω e o de 10 KΩ, assim será fácil identificá-los por cor.

Buzzer

Outro componente que vai ser utilizado é o *Buzzer* (Buzina). Esse componente possui o funcionamento bem simples: quando uma corrente elétrica passa por ele, é emitido um som. Pode ser usado também para emitir a sonoridade em várias frequências diferentes, para isso, você deve fazer uso do PWM (*Pulse Wide Modulation*), o que não é o foco dessa aula, mas se quiser conhecer mais sobre ele, vale a pena a pesquisa. Um ponto importante do *Buzzer* é a sua polaridade, percebam na foto da Figura 02 que um dos lados possui um sinal de +, é importante que ele seja ligado dessa forma para que não queime o componente nem a placa.

Figura 02 - *Buzzer*

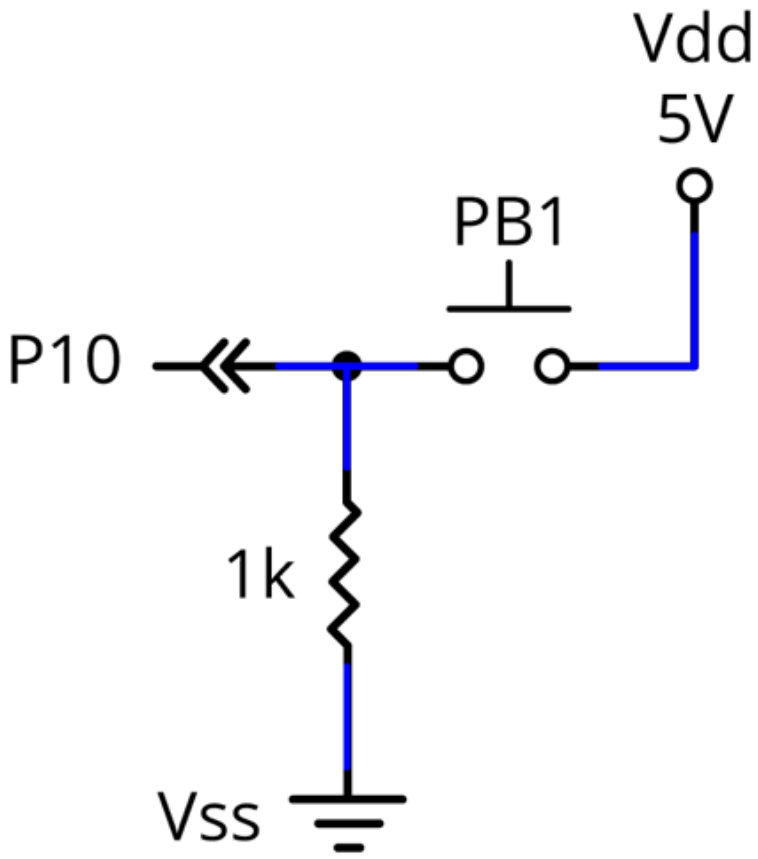


Fonte: adaptado de BAÚ DA ELETRÔNICA. Disponível em: <http://www.baudaeletronica.com.br/buzzer-5v.html> Acesso em: 28 maio 2018.

Push Button

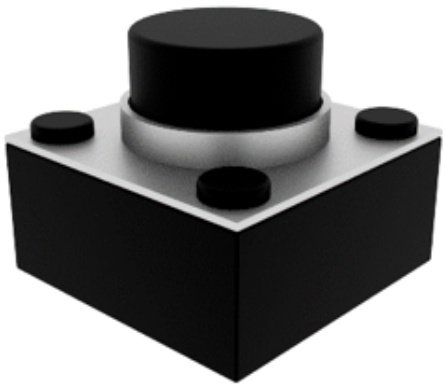
O *push button* (chave momentânea) é um botão que, quando apertado, faz os contatos dos lados se ligarem entre si, seu funcionamento é bem simples, mas sua ligação requer um pouco de cuidado, pois é necessário fazer um divisor de tensão (lembra das aulas de conceitos de eletricidade?), como você pode ver na Figura 03.

Figura 03 - Circuito do *push button*



O botão que você vai utilizar nessa aula é o que está abaixo, na Figura 04. Ele tem quatro pinos: são dois pinos de cada lado que estão em contato e ficam normalmente abertos, após o botão ser pressionado são fechados todos os contatos.

Figura 04 - *Push button*

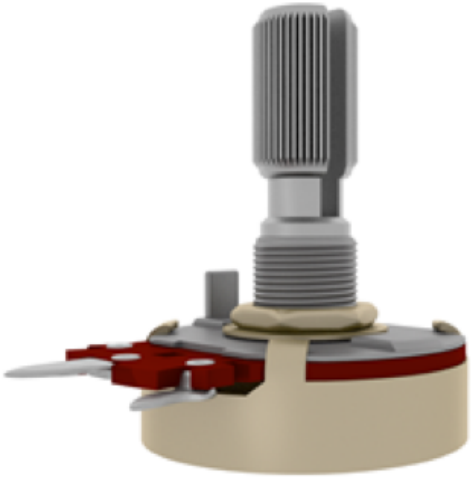


Fonte: adaptado de FILIPEFLOP. Disponível em: <https://www.filipeflop.com/produto/chave-tactil-push-button-x10-unidades/>. Acesso em: 28 maio 2018.

Potenciômetro

O potenciômetro funciona como um resistor variável, que aumenta ou diminui a resistência conforme a haste é rotacionada. Na Figura 05 você ver o modelo que será usado na prática. O potenciômetro possui 3 pinos e a resistência varia entre um dos pinos mais da extremidade para com o do centro, quando utilizá-lo você irá entender melhor o seu funcionamento.

Figura 05 - Potenciômetro



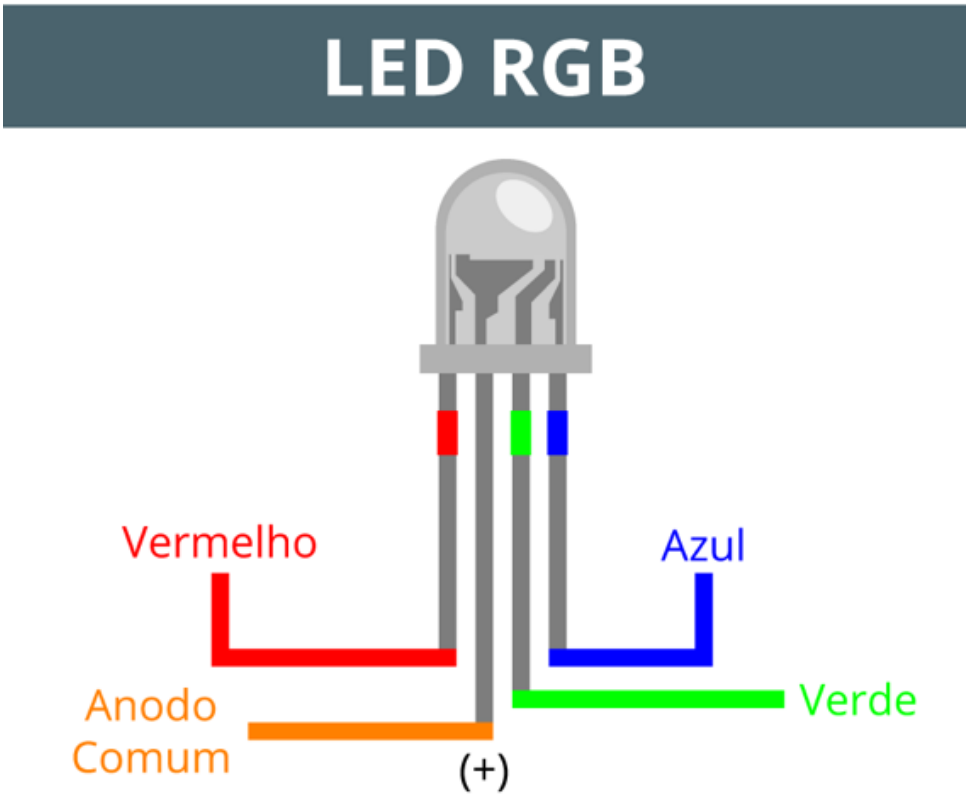
Fonte: Adaptada de HU INFINITO COMPONENTES ELETRÔNICOS. Disponível em: <http://www.huinfinito.com.br/potenciometros/682-potenciometro-linear-5k-cursor-

metal.html>. Acesso em: 28 maio 2018.

LED RGB

O LED (*Light Emitting Diode*) RGB (*Red, green e blue*) possui três LEDs juntos no mesmo encapsulamento. De alto brilho e com cores diferentes, obedece o padrão RGB, assim há um bulbo vermelho, um verde e um azul - pois com o padrão RGB pode-se montar luzes de qualquer cor. Fisicamente são quatro pinos, um para cada uma das cores, e o maior deles (o Anodo), existe uma saída para uma entrada, nesse caso há uma saída para as três entradas, ele é comum a todas.

Figura 06 - LED RGB

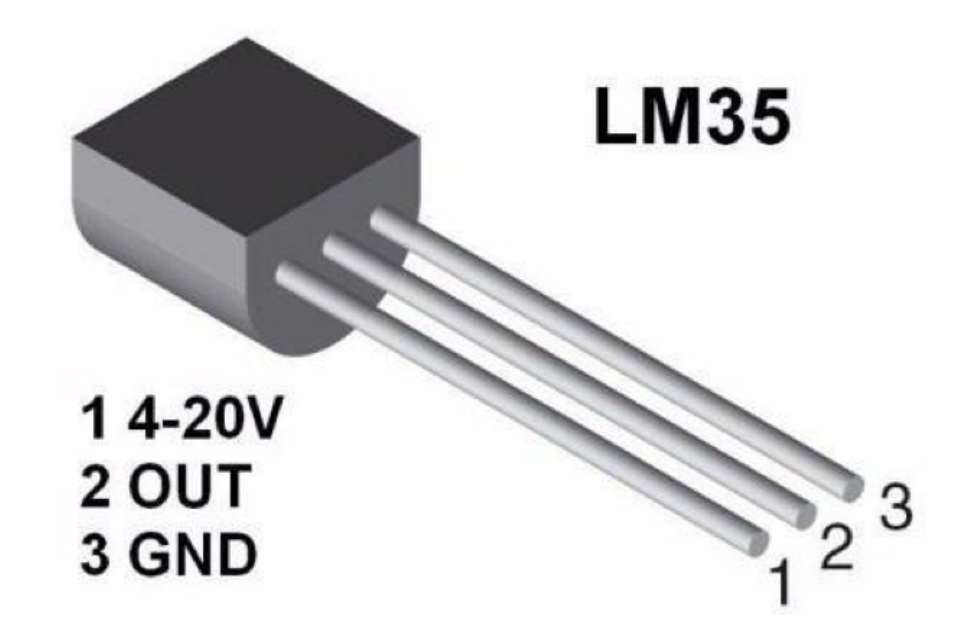


Fonte: ROBOCORE (S. d.).

Sensor de Temperatura

Você precisará também utilizar o sensor de temperatura LM35, um circuito integrado (CI) que faz a medição de temperatura em graus Celsius (°C). Como você pode ver na Figura 07 o CI possui três pinos do mesmo comprimento, cada pino tem uma função: alimentação, saída e terra. É importante você ficar atento para sua ligação.

Figura 07 - LM35



Fonte: INSTRUCTABLES. Disponível em: <http://www.instructables.com/id/LM35-Temperature-Sensor/>. Acesso em: 28 maio 2018.

Saiba Mais

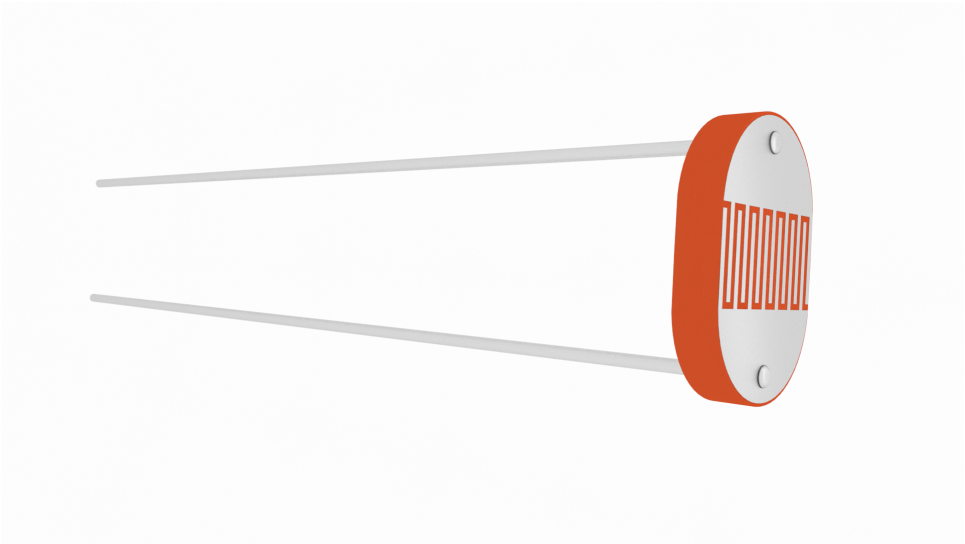
Para conhecer mais sobre a *data sheet* do sensor e suas características acesse aqui. <http://www.ti.com/lit/ds/symlink/lm35.pdf>

Ah... Também será uma boa oportunidade para você aplicar o que aprendeu nas aulas de Inglês Técnico.

Sensor de Luminosidade

Você também irá utilizar outro sensor para mensurar a luminosidade. O sensor que vai ser utilizado é o LDR (*Light Dependent Resistor*), que basicamente é um resistor que varia de acordo com a luminosidade. Como você pode ver na Figura 08 ele possui dois pinos da mesma forma que uma resistência e não há polaridade para ligar ao circuito, mas em sua montagem é importante fazer uso de um divisor de tensão.

Figura 08 - LDR

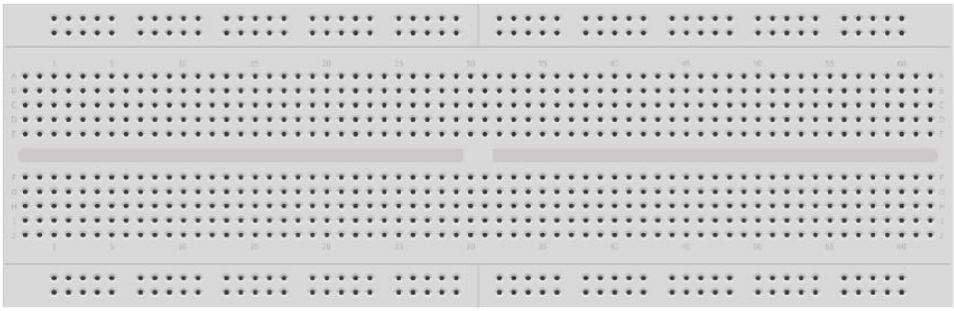


Fonte: IMPORTADORA ARDUNE. Disponível em: <http://importadoraardunel.com/sensores/46-sensor-de-luz-fotorresistencias-ldr.html>. Acesso em: 28 maio 2018.

Protoboard

Para fazer todas as ligações dos componentes ao Arduino é necessário uma *protoboard* (Figura 09), que se trata de uma placa de plástico, cheia de pequenos furos com ligações internas, onde você irá fazer as ligações elétricas.

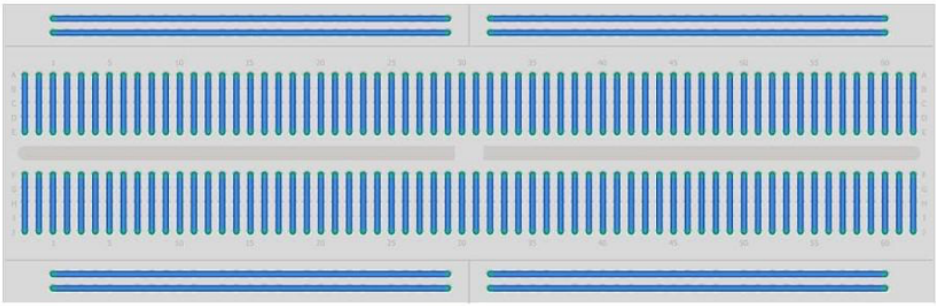
Figura 09 - *Protoboard*



Fonte: adaptado de MERCADO LIVRE. <https://produto.mercadolivre.com.br/MLB-926578752-protoboard-arduino-placa-de-circuito-de-teste-830-pontos-JM> Acesso em: 28 maio 2018.

É muito comum na hora das montagens haver dificuldade para entender as ligações da *protoboard*, mas basicamente os furos nas extremidades superior e inferior são ligados entre si na horizontal, enquanto as barras do meio são ligadas na vertical. Veja na Figura 10 as ligações internas da *protoboard*.

Figura 10 - Ligações internas da *Protoboard*

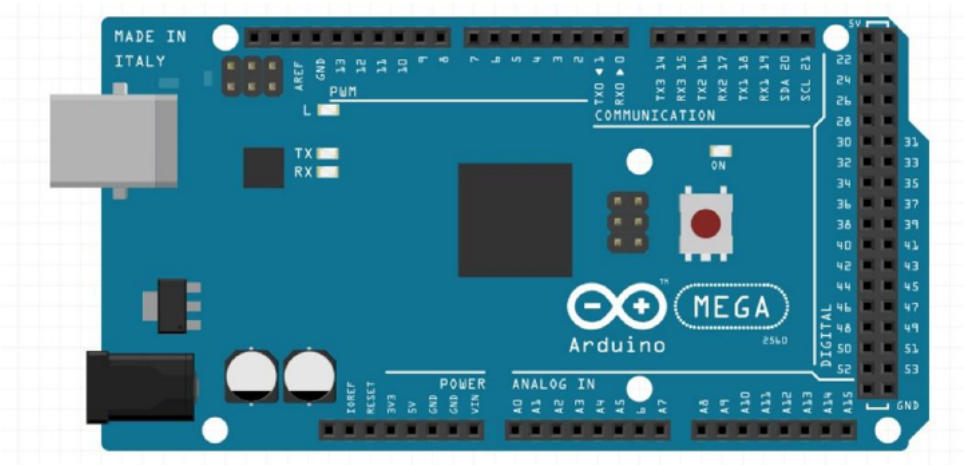


Fonte: adaptado de ROBOCORE. Disponível em: <https://www.robocore.net/tutoriais/como-utilizar-uma-protoboard.html> Acesso em: 28 maio 2018.

Arduino Mega 2560

Para essa prática você vai fazer uso do Arduino Mega 2560, que é uma placa de microcontrolador baseada no ATmega2560. Ela possui 54 pinos de entrada/saída digitais (dos quais 15 podem ser utilizados como saídas PWM), 16 entradas analógicas, 4 UARTs (portas seriais de hardware), um oscilador de cristal de 16 MHz, uma conexão USB, uma tomada de energia, um cabeçalho ICSP, e um botão de reinicialização, como você vê na Figura 11.

Figura 11 - Arduino Mega 2560



Fonte: FRITZING. Disponível em: <http://fritzing.org>. Acesso em: 28 maio 2018.

A placa contém tudo o que é necessário para suportar o microcontrolador; apenas conecte-a a um computador com um cabo USB ou a ligue com um adaptador ou bateria AC-to-DC. A placa Mega 2560 é compatível com a maioria das *shields* projetadas para o Arduino Uno.

Não é o foco do curso de automação conhecer sobre microcontroladores, mas é importante detalhar um pouco o desempenho dessa placa, veja na Figura 12 as características técnicas:

Figura 12 - Características técnicas do Arduino Mega

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm
Width	53.3 mm
Weight	37 g

Fonte: ARDUINO. Disponível em: <https://www.arduino.cc/en/Main/ArduinoBoardMega/>. Acesso em: 28 maio 2018.

Saiba Mais

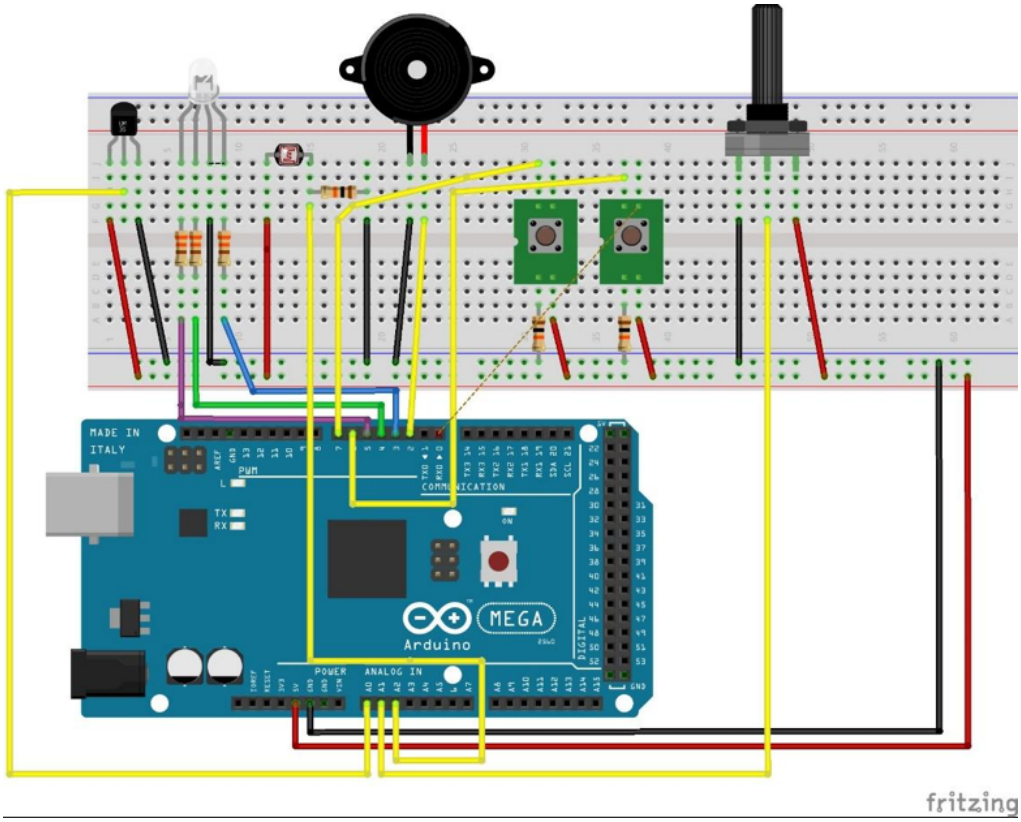
Para conhecer mais sobre o Arduino e toda sua família de placas visite este site. <https://www.arduino.cc/en/Main/arduinoBoardMega2560/>

Montagem

Agora que você conheceu cada componente do sistema junte-os na *protoboard* e ligue a placa. Como não é foco do curso de automação uma eletrônica muito complexa, vou colocar o sistema como você deve montar em sala de aula. Para criar a imagem, fiz uso do *Fritzing*, que é um software que auxilia na montagem e prototipação de circuito, para conhecer mais sobre ele basta entrar em www.fritzing.org.

No circuito montado (Figura 13), foram utilizados os fios vermelhos para a alimentação positiva, os fios pretos para a negativa e os fios amarelos para a ligação ao Arduino, já para o LDR, os fios foram colocados de acordo com a cor dos LEDs. Na prática pode ser que você não tenha os *jumpers* (fios para interligação) dessas cores, não importa, o intuito aqui é apenas facilitar a montagem.

Figura 13 - Circuito proposto



Fonte: FRITZING. Disponível em: <fritzing.org>. Acesso em: 28 maio 2018.

Atenção

Grande parte dos problemas encontrados em circuitos estão em sua montagem, então tenha bastante atenção ao ligar os componentes. Em caso de dúvida, é sempre bom consultar as *data sheets* e revisar todas as ligações antes de testar.

O Software

O foco dessa prática não é ensinar a programar em Arduino, mas como você já tem um bom conhecimento em programação não será difícil entender o código. Desse modo, vou apresentá-lo, com todos os comentários para facilitar o entendimento e a manutenção.

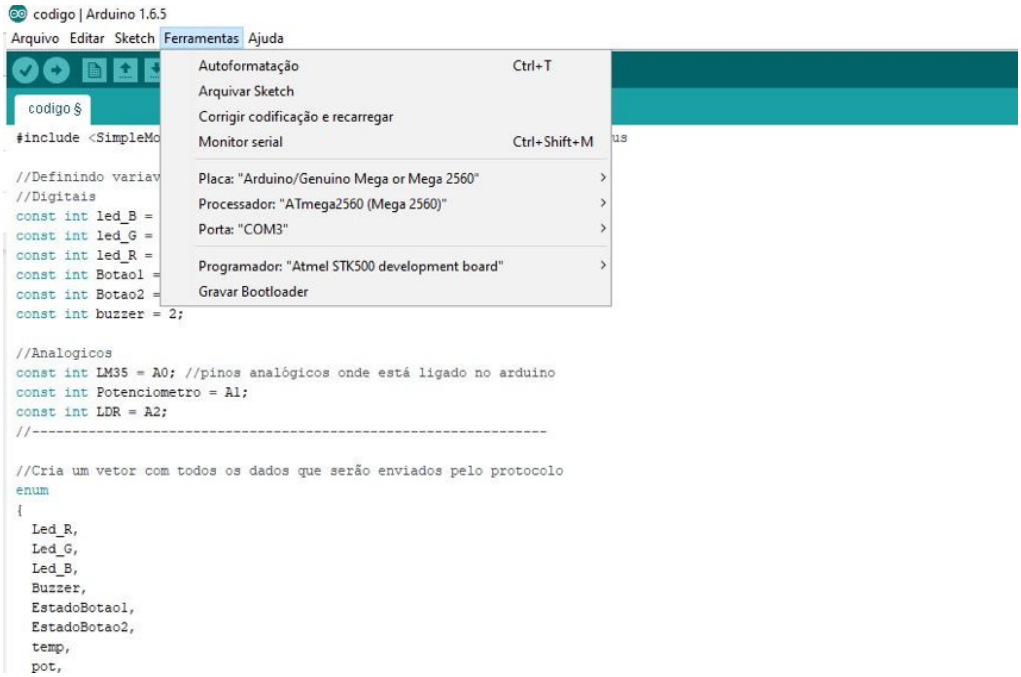
Código 1 – A programação do Arduino

```
#include <SimpleModbusSlave.h> // inclusão da biblioteca do protocolo modbus

//Definindo variaveis e seus respectivos pinos no arduino
//Digitais
const int led_B = 3; //pinos digitais onde está ligado no arduino
const int led_G = 4;
const int led_R = 5;
const int Botao1 = 6; //
const int Botao2 = 7; //
const int buzzer = 2;

//Analogicos
const int LM35 = A0; //pinos analógicos onde está ligado no arduino
const int Potenciometro = A1;
const int LDR = A2;
//---------------------------------------------------------------

//Cria um vetor com todos os dados que serão enviados pelo protocolo
enum
{
   Led_R,// endereço offset 0
   Led_G, // endereço offset 1
   Led_B, // endereço offset 2
   Buzzer, // endereço offset 3
   EstadoBotao1, // endereço offset 4
   EstadoBotao2, // endereço offset 5
   temp,// endereço offset 6
   pot,// endereço offset 7
   luz, // endereço offset 8
   HOLDING_REGS_SIZE
};
unsigned int holdingRegs[HOLDING_REGS_SIZE];


// No setup definimos as variaveis que são entradas e saídas
void setup() {

   Serial.begin(9600); // inicia a porta serial
   modbus_configure(&Serial, 9600, SERIAL_8N1, 1, 2, HOLDING_REGS_SIZE, holdingRegs);// configu
   modbus_update_comms(9600, SERIAL_8N1, 1);


   pinMode(Buzzer, OUTPUT);// configuraos pinos como entrada ou saída
   pinMode(led_R, OUTPUT);
   pinMode(led_G, OUTPUT);
   pinMode(led_B, OUTPUT);
   pinMode(Botao1, INPUT);
   pinMode(Botao2, INPUT);

}

```

```
52 void loop(){
53
54    modbus_update();
55
56    //Leitura dos botões
57    holdingRegs[EstadoBotao1] = digitalRead(Botao1);
58    holdingRegs[EstadoBotao2] = digitalRead(Botao2);
59
60    //Leitura dos sensores
61    holdingRegs[luz] = analogRead(LDR);// Ler valor do sensor de luminosidade e envia para o Scada
62    holdingRegs[pot] = analogRead(Potenciometro);// Ler valor do potenciometro e envia para o Scad
63    holdingRegs[temp] = analogRead(LM35);// Ler valor do sensore de temperatura e envia para o Sc
64
65    //Recebendo os dados digitais
66    digitalWrite(buzzer, holdingRegs[Buzzer]); //recebe o dado enviado pelo ScadaBr e envia para por
67    digitalWrite(led_R, holdingRegs[Led_R]);
68    digitalWrite(led_G, holdingRegs[Led_G]);
69    digitalWrite(led_B, holdingRegs[Led_B]);
70
71 }
```

A programação do Arduino é feita em sua própria IDE (*Integrated Development Environment*), mas também tem suporte para ser usada em outras. Após digitar o código e conectar sua placa, você deverá fazer algumas configurações no menu ferramentas, lá você deve configurar a placa (Mega), o processador (Atmega2560) e a porta. A seleção da porta vai depender em geral da escolha computador, mas desde já adianto que não é a COM1, o ideal é verificar quais portas existem antes de colocar a placa, e ver qual passa a existir quando a placa é conectada, pois é nesse instante que o computador atribui a ela uma porta. Na Figura 14 observe o menu. Após a configuração, basta clicar no ícone  , localizado no canto superior esquerdo, e seu programa será gravado na placa.

Figura 14 - Configurações no IDE do Arduino



ScadaBR

Agora que você conheceu os componentes, já sabe como fazer as ligações no circuito e como programar o microcontrolador, falta apenas aprender a integrar a placa ao *ScadaBR*. Para isso, você precisa configurar o protocolo de comunicação *modbus* serial que será usado na interligação da placa com o computador. Veja na Figura 15 os campos já preenchidos de acordo com o código colocado em sua a placa.

Figura 15 - *Data source modbus* serial

Fonte: SCADABR (2015).

A *data source Modbus* serial é usada para adquirir dados a partir de uma rede local *modbus*, acessível via comunicação RS232 ou RS485 (você já deve ter visto isso na aula de redes). Toda *data source* requer um **Nome**, o qual pode ser qualquer descrição. O **Período de atualização** determina com que frequência a rede *Modbus* será consultada por dados. Os campos ***Timeout*** e **Tentativas** determinam o comportamento do sistema no caso de uma falha em uma consulta. A *data source* aguarda um determinado número de milissegundos por uma resposta da rede. Se nenhuma resposta for recebida, a requisição será refeita um dado número de vezes. A opção **Apenas quantidades contínuas** pode ser usada para especificar que a implementação *modbus* não deve tentar otimizar diferentes requisições de valores em uma única requisição. Selecionar essa opção fará com que a implementação somente faça requisições para múltiplos valores quando esses valores formarem um espaço de registro contínuo (SCADABR, 2015).

Um ponto muito importante de configuração é comum a qualquer comunicação serial que é controlada com os valores de **Baud rate** (deve ser igual ao dispositivo), de **Controle de fluxo de entrada**, de **Controle de fluxo de saída**, dos **Bits de dados** (o tamanho do dado enviado), dos **Bits de parada**, e da **Paridade** (verifica se há erro). O ***Echo*** pode ser usado com redes com redes RS485 quando for apropriado.

O valor da **codificação** determina como as requisições *Modbus* são formatadas. A maioria dos hardwares de produção usa mensagens RTU formatadas, mas é importante sempre ver a documentação *Modbus* do seu equipamento para determinar como definir esse campo. No seu caso, utilize a RTU.

Uma configuração importante é a **Porta**, é necessário que você verifique qual porta o computador atribui a sua placa.

Após a configuração coloque os *data points* referentes a cada sensor e atuador que estão na placa, os quais são:

- Led na cor vermelha (endereço offset 0);
- Led na cor verde (endereço offset 1);
- Led na cor azul (endereço offset 2);
- *Buzzer* (endereço offset 3);

- Botão 01 (endereço offset 4);
- Botão 02 (endereço offset 5);
- Sensor de temperatura (endereço offset 6);
- Potenciômetro (endereço offset 7);
- Sensor de luminosidade (endereço offset 8).

Na Figura 16, você vê os *data points* já configurados e é interessante que você tenha sempre uma atenção especial com os endereços. Só é necessário marcar como **configurável** os *data points* que possuem comunicação do *ScadaBR* -> Placa, no caso os LEDs, os botões e o *buzzer*.

Figura 16 - Configuração dos *data points*

Nome	Tipo de dado	Status	Escravo	Faixa	Offset (baseado em 0)
botao_1	Binário		1	Registrador holding	4/0
botao_2	Binário		1	Registrador holding	5/0
buzzer	Binário		1	Registrador holding	3/0
led_b	Binário		1	Registrador holding	2/0
led_g	Binário		1	Registrador holding	1/0
led_r	Binário		1	Registrador holding	0/0
luz	Numérico		1	Registrador holding	8
Potenciometro	Numérico		1	Registrador holding	7
Temperatura	Numérico		1	Registrador holding	6

Fonte: SCADABR (2015).

Como se está recebendo dados de sensores e você sabe que esses dados chegam em forma de bits (0-1023), pode-se fazer na entrada a transformação. Na figura abaixo, se vê o multiplicador sendo utilizado para converter os dados de bits para °C. Esse processo de conversão é bem demorado e em alguns casos são necessários vários ensaios, bem como o uso de equipamentos padronizados. Por exemplo, você precisaria de um termômetro confiável para configurar com precisão a leitura do sensor.

Figura 17 - Manipulação dos dados na entrada

Fonte: *SCADABR* (2015).

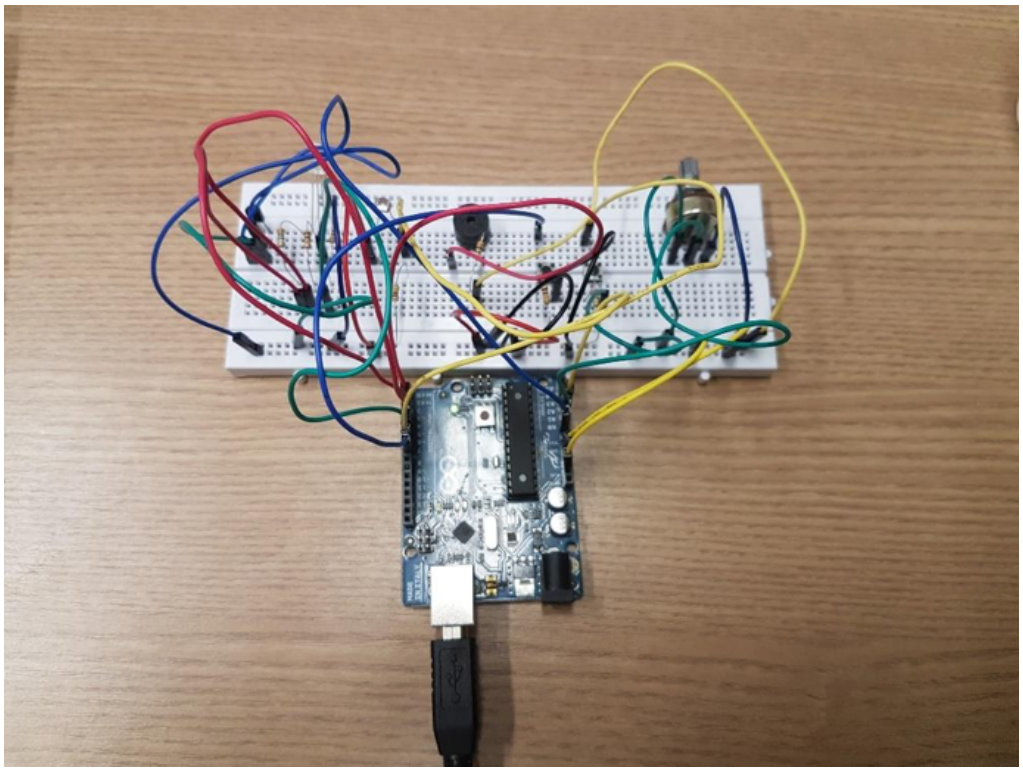
Para ver os data points gerados e a comunicação da placa com *ScadaBR* basta olhar na *watch list*, como mostra a Figura 18.

Figura 18 - Watch List

Fonte: SCADABR (2015).

A Figura 19 tem o circuito que, quando montado, fica bem mais bagunçado que o da Figura 13, mas é importante que todas as ligações estejam corretas, por isso, é bom ter cuidado e paciência para montar, pois depois de pronto dá mais trabalho para encontrar erros.

Figura 19 - Circuito montado



Fonte: adaptado de <https://lh3.googleusercontent.com/uTeb-rePZ8mFPpidVMMYavyWgfdPrKnVCKSHIqZ36a68ypqyPD3jt8PwKVEV4fcv9dFJjA=s113>. Acesso em: 28 maio 2018.

Você já deve ter visto até agora que só os dados no *ScadaBR* foram recebidos, falta ainda:

- Configurar as variáveis;
- Tratar as variáveis;
- Criar os *point links*;
- Criar as telas;
- Configurar os relatórios.

Hum... muitas coisas novas, não? Você deve ter enriquecido bastante o seu conhecimento com essas aulas!

Então, vá ao laboratório e coloque tudo o que você aprendeu em prática! 😉

Resumo

Nesta aula você aprendeu como configurar o protocolo *modbus* em um Arduino e como fazer sua interligação ao *ScadaBR* simulando um sistema de automação. Conheceu também alguns componentes eletrônicos e como utilizá-los em um circuito.

Autoavaliação

Agora faça todos os passos apresentados na aula para fixar o conhecimento que você adquiriu. Você já pode até adiantar o código e testá-lo em algum simulador de Arduino.

Referências

SCADABR. **Automação para todos**. 2015. Disponível em: <http://www.scadabr.com.br/?q=downloads>. Acesso em: 16 maio 2018.

ROBOCORE. Apostila: kit iniciante v7.2: para Arduino. S. d. Disponível em: <www.robocore.net>. Acesso em: 16 maio 2018.