PROGRAMAÇÃO MOBILE

Aline Zanin

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

Introdução à Internet das Coisas na plataforma Arduino

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Identificar os elementos arquiteturais do Arduino.
- Reconhecer os componentes e protocolos que possibilitam a comunicação com o Arduino.
- Ilustrar aplicações de IoT no Arduino.

Introdução

O Arduino é uma placa de microcontrolador que se destacou no mercado por permitir efetuar diversos trabalhos de automação com pouco conhecimento de eletrônica associado a conhecimento de programação. Além disso, o projeto da placa é aberto (*open source*), e com isso é liberada a construção de placas compatíveis com o Arduino, resultando na oferta de placas de baixo custo, o que popularizou a sua aquisição.

Um dos locais de aplicação de Arduino é em projetos de Internet das Coisas (*Internet of Things*) (IoT), a qual se fundamenta na conexão e na comunicação dos dispositivos através da internet. Em IoT não apenas computadores se comunicam, mas também diversos outros dispositivos. Através da conexão destes dispositivos e do uso de Arduino, podem ser feitos diversos tipos de automações residenciais e comerciais, como iluminação e eletrodomésticos inteligentes e sistemas de segurança.

Neste capítulo, você vai estudar sobre os conceitos fundamentais do Arduino. Exploraremos as suas diferentes versões e a aplicação do Arduino para IoT. Você lerá sobre os elementos arquiteturais do Arduino,

aprenderá a reconhecer os componentes e protocolos utilizados pela plataforma para comunicação e, por fim, verá como dar os primeiros passos em IoT utilizando Arduino.

1 Elementos arquiteturais do Arduino

O Arduino, conforme mencionado anteriormente, é uma placa de microcontrolador. Ela contém um plugue de conexão USB que pode ser utilizado, entre outras coisas, para energizar a placa através da conexão com o computador. Além dessa placa, possui diversos pinos de conexão, que permitem a ligação da placa com dispositivos eletrônicos, como: motores, relés, sensores luminosos, diodos a *laser*, alto-falantes, microfones e outros (MONK, 2017).

Além do USB, as placas de Arduino podem ser energizadas por bateria ou por fonte de alimentação. As placas de Arduino podem ser complementadas por componentes, chamado *shields*, que são acoplados às placas dando outras funções a elas, como por exemplo, acoplando sensor de temperatura e movimento (MONK, 2017).

Fique atento

Relé é um componente eletromecânico, um tipo de atuador que comuta mecanicamente circuitos elétricos. Os relés têm papel fundamental em sistemas de acionamento de motores, fornecendo a lógica de acionamento pela comutação de múltiplos circuitos de acionamento (MALVINO; BATES, 2011).

Na Figura 1 é apresentada de forma lúdica a arquitetura do Arduino. É possível visualizar a existência de elementos de entrada e elementos de saída, e ao centro o microcontrolador, que é o principal elemento de uma placa Arduino, que possui pinos para conexão dos elementos de entrada e de saída. Existem diversas placas de Arduino, contudo a existência destes três elementos (entrada, microcontrolador e saída é universal e existirá em todas as placas).

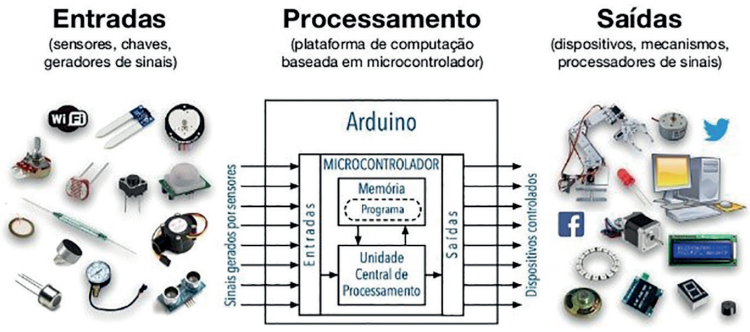

Figura 1. Arquitetura Arduino.

Fonte: Adaptada de ETACARINAE (2017).

A seguir vamos apresentar a descrição de cada um dos elementos arquiteturais que são comuns no Arduino, mas com ênfase aqui nas placas de Arduino Uno. A Figura 2 apresenta uma placa Arduino Uno na qual estão sinalizados cada um dos componentes que serão descritos a seguir.

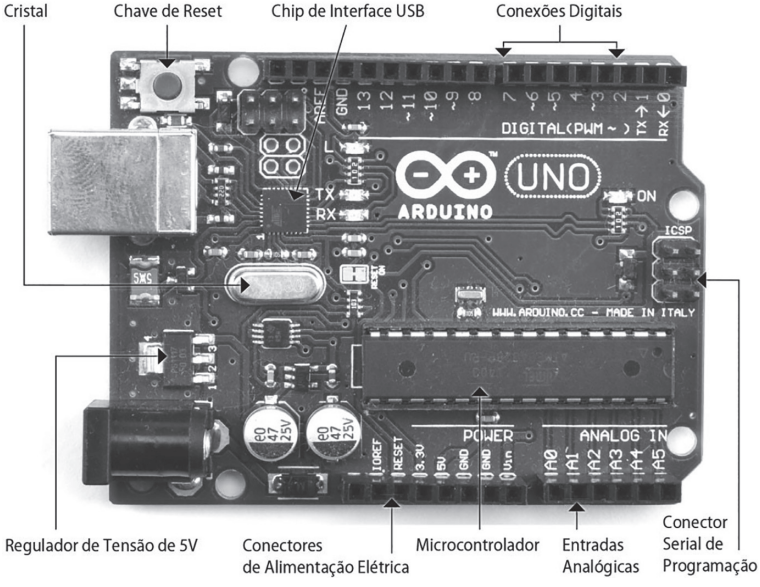

Figura 2. Placa Arduino Uno.

Fonte: Monk (2017).

Microcontrolador — O coração do nosso Arduino é um microcontrolador. Praticamente todos os demais componentes da placa destinam-se ao fornecimento de energia elétrica e à comunicação com o seu computador (MONK, 2017). Eles são componentes pequenos, mas com diversas funcionalidades. Os microcontroladores são, na verdade, diversos computadores associados a uma placa. Assim como o computador, os microcontroladores possuem processador, memória RAM para guardar dados, memória EPROM ou memória *flash* (para armazenar os programas) e pinos de entrada e saída. São esses pinos de entrada/saída que permitem ligar os microcontroladores aos demais componentes dos circuitos (MONK, 2017):

> O microcontrolador de uma placa de Arduino Uno é um chip (circuito integrado) de 28 pinos que está encaixado em um soquete no centro da placa. [...] Ele é fabricado pela empresa Atmel, que é uma das maiores fabricantes de microcontroladores (MONK, 2017).

Conexões de alimentação elétrica — Os pinos de conexão de alimentação elétrica estão localizados na parte inferior da Figura 2, e um deles é o pino de Reset. Ao clicar neste botão, um pulso lógico é enviado ao pino Reset do microcontrolador, obrigando o microcontrolador a começar a execução do programa desde o início e a limpar a sua memória (MONK, 2017). Os demais pinos desta seção fornecem diversas tensões: 3,3V, 5V, GND (*ground* ou terra), conforme estão indicadas na placa, sendo que GND significa simplesmente zero volts e é a tensão que serve de referência a todas as demais tensões da placa (MONK, 2017).

Entradas analógicas — Grandezas analógicas são aquelas que podem assumir infinitos valores de amplitude dentro de uma faixa de valores, como por exemplo, o velocímetro de um carro que tem o ponteiro oscilando pelas faixas de velocidade; caso o ponteiro girasse em saltos, o velocímetro seria considerado digital.

Localizados ao lado das conexões elétricas, existem seis pinos denominados *Analog In* (Entrada Analógica), indo de A0 a A5, eles podem ser usados para receber dados provenientes de grandezas analógicas. Essas conexões têm capacidade de medir a tensão que está sendo aplicada a cada um desses pinos, de modo que os seus valores podem ser usados em um programa de Arduino (MONK, 2017).

Conexões digitais — Localizados na parte superior da Figura 2, no lado direito, podem ser usadas como entradas ou saídas. Quando são usados como **saídas**, os pinos comportam-se como se fossem pinos de alimentação elétrica, todos de 5V, podendo fornecer 40 mA. A voltagem fornecida por esses pinos pode ser utilizada, por exemplo, para acender um LED comum, contudo não é suficiente para acionar diretamente um motor elétrico. Os primeiros dois pinos (0 e 1), também denominados RX e TX, são para recepção e transmissão, utilizados, por exemplo, pelo Arduino para se comunicar com seu computador através de recepção e transmissão (MONK, 2017).

Outros componentes — Além dos componentes descritos anteriormente, o Arduino também tem outros componentes com relevante significado, como os seguintes (MONK, 2017):

- oscilador a cristal: realiza 16 milhões de ciclos ou oscilações por segundo e, em cada um desses ciclos, o microcontrolador pode executar uma operação — de adição, subtração ou alguma outra operação matemática;
- chave de Reset: quando se aperta essa chave, um pulso lógico é enviado ao pino de Reset do microcontrolador, fazendo com que o microcontrolador seja reiniciado;
- conector serial de programação: ele proporciona um meio alternativo para programar o Arduino sem que a porta USB seja usada;
- soquete USB: aqui encontra-se o *chip* de interface USB. Esse *chip* converte os níveis de sinal usados pelo padrão USB em níveis que podem ser usados diretamente pela placa do Arduino.

2 Protocolos utilizados para comunicação

Protocolos de comunicação podem ser comparados a contratos: são regras que controlam a comunicação entre dois ou mais sistemas. Através de protocolos são definidos, por exemplo: configurações de parâmetros, sintaxe, semântica e padrões que os dispositivos conectados devem obedecer. Os sistemas microcontrolados — e aqui damos uma ênfase especial ao Arduino — também utilizam protocolo de comunicação. Nesta seção conheceremos os protocolos Universal Asynchronous Receiver Transmitter (UART), Inter-Integrated Circuit (I2C) e Serial Peripheral Interface (SPI), que são os que têm maior destaque na comunicação do Arduino com periféricos externos,

como por exemplo, sensores e atuadores e com outros microcontroladores. Exploraremos como eles funcionam e que componentes do Arduino eles utilizam (MONK, 2015).

Protocolo Universal Asynchronous Receiver Transmitter

A UART é um tipo de comunicação serial e assíncrona, que transmite dados através de barramento de comunicação. Ela utiliza níveis de tensão TTL, que possui valores 0V e 5V para representar os níveis lógicos 0 e 1, respectivamente (MONK, 2015).

Este protocolo pode ser utilizado de três formas:

- *simplex*, onde apenas um dispositivo pode transmitir dados;
- *half duplex*, onde os dois dispositivos conectados podem transmitir dados, contudo um de cada vez;
- *full-duplex*, onde existem canais separados de modo que os dispositivos podem enviar dados simultaneamente (MONK, 2015).

A comunicação serial UART (Figura 3) é utilizada para comunicar o Arduino com o computador e com módulos eletrônicos, como o Bluetooth. Por padrão, é ligada com os pinos digitais 0 e 1 (MONK, 2015).

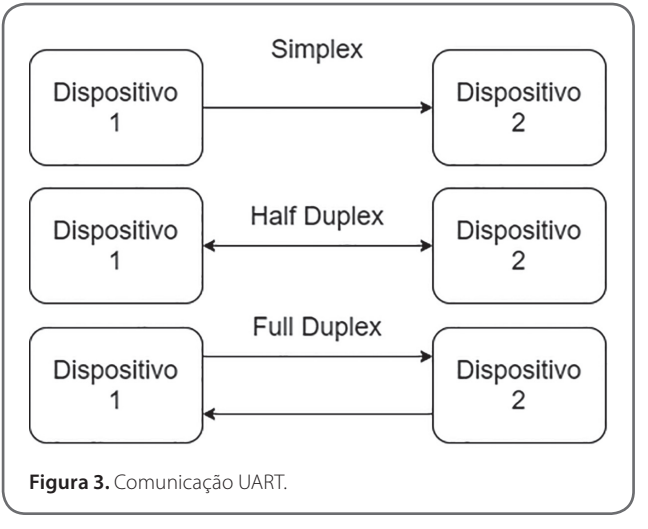

Figura 3. Comunicação UART.

Protocolo Inter-Integrated Circuit

A I2C é uma comunicação serial que se baseia no modelo mestre e escravo (*máster/slave*). Neste protocolo pode existir mais de um dispositivo mestre, sendo que o mestre em geral é um microcontrolador ou outro dispositivo programável que é responsável por coordenar a comunicação e requisitar informação dos escravos, e cada escravo deve possuir um endereço único na comunicação. Para a comunicação fluir neste tipo de protocolo, são utilizados dois dispositivos, um *Serial Data Line* (SDA) e um SCL (*Serial Clock Line*). O SDA é um canal bidirecional onde os dados são transmitidos entre mestre e escravo, e o SCL é controlado pelo mestre e utilizado para comunicação (MONK, 2015).

A transmissão de dados na comunicação I2C (Figura 4) é iniciada com o sinal de *Start* enviado pelo dispositivo mestre. Através deste sinal todos os dispositivos conectados no barramento irão escutar as mensagens que estão sendo transmitidas no barramento (MONK, 2015).

O mestre irá enviar no barramento o endereço do dispositivo com o qual ele quer estabelecer a comunicação, todos os dispositivos irão comparar o endereço recebido com o seu endereço e então o escravo que está no endereço enviado pelo mestre irá "se apresentar" através de um sinal de *Acknowledge* (ACK), e então a comunicação passa a acontecer exclusivamente entre ele e o mestre. Por fim, o mestre envia o sinal de Stop no barramento para finalizar a comunicação. A comunicação I2C do Arduino, também chamada de *Two Wire Interface*, está disponível nos pinos analógicos A4(SDA0 e A5(SCL) (MONK, 2015).

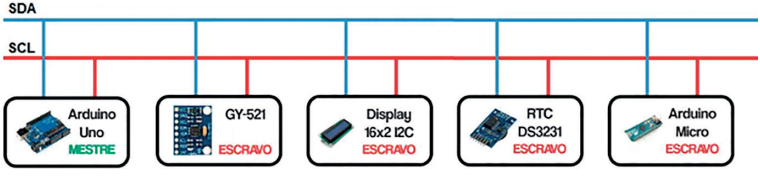

Figura 4. Comunicação I2C.

Fonte: Madeira (2017, documento *on-line*).

Protocolo de comunicação SPI

A SPI, ou comunicação serial, é umas das comunicações seriais mais rápidas, contudo utilizada apenas para pequenas distâncias. Assim como acontece na

comunicação I2C, no SPI também é utilizada a hierarquia mestre-escravo, como a I2C, mas no modo de transmissão *full-duplex*, onde os dispositivos podem enviar e receber mensagens simultaneamente (MONK, 2015).

Para que a comunicação *full-suplex* possa fluir, normalmente a SPI utiliza quatro canais, sendo eles:

1. MISO (*Master In Slave Out*): utilizada para o escravo para enviar dados para o dispositivo mestre;
2. MOSI (*Master Out Slave In*): utilizada para o mestre para enviar dados para o escravo;
3. SCK ou SCLK (*Serial Clock*): pulsos de *clock* de sincronismo da transmissão gerada pelo dispositivo mestre;
4. SS (*Select Slave*): utilizada quando existe mais de um dispositivo escravo no barramento.

A comunicação SPI (Figura 5) é utilizada para comunicação com velocidades rápidas e curtas distancias, e no Arduino uno está disponível nos pinos 11 MOSI, pino 12 MISO e 12 SCK, e pino 10 SS (MONK, 2015).

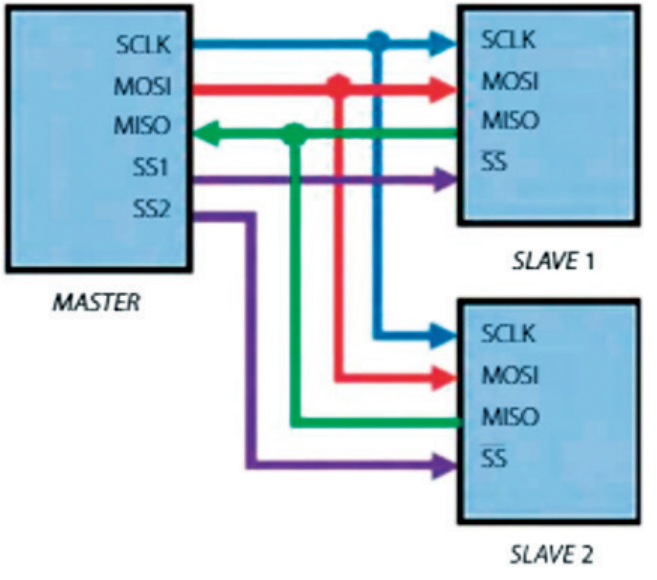

Figura 5. Comunicação SPI.
Fonte: Adaptada de Corelis ([201–]).

Comunicação via rede

Além dos protocolos descritos acima, que são utilizados para comunicação do Arduino com o mundo através dos pinos da placa, o Arduino permite ainda o uso de placas de rede cabeada e sem fio. A placa de rede pode ser uma placa conectada através de um *shield*, ou também pode-se utilizar uma versão do Arduino, o Arduino Yun, que foi desenvolvida preparada para interação com a internet e por isso já possui as placas de rede embutidas. Desta forma, ao desenvolver soluções para Arduino o programador pode também usar os protocolos de nível de aplicação (como HTTP) e de nível de rede (como TCP e UDP) na comunicação do Arduino com o mundo exterior.

3 Desenvolvimento de aplicações no Arduino

O Arduino se mostrou um excelente componente para IoT, o seu baixo custo e facilidade de implementação facilitaram a sua popularização. O Arduino pode ser utilizado em ações simples, como acender e apagar um LED, até ações mais complexas, como uma automação residencial completa. A seguir, são listados exemplos de projetos que podem ser realizados no contexto de IoT utilizando Arduino.

- Arduino Uno como servidor *web*: utilizando um *shield* de ethernet e a rede local pode ser criado um servidor simples de *web* que permite que através do navegador do computador você envie uma requisição para o Arduino e veja os resultados no navegador (MONK, 2017).
- Arduino Controlado pela *web*: utilizando também um *shield* de ethernet é possível que os pinos digitais 3 a 7 do Arduino sejam ligados ou desligados utilizando um formulário *web* simples (MONK, 2017).
- Controle remoto de TV: você pode com Arduino transformar o seu *smartphone* em um controle remoto para utilizar na sua televisão (MONK, 2014).
- Registrador de temperatura: com Arduino você pode utilizar um sensor para medir a temperatura de alguma coisa ou local e exibir esta temperatura no celular (MONK, 2014). A partir disso você pode enviar outros comandos, como ligar o ar condicionado, por exemplo.
- Medir distâncias: com Arduino você pode criar um medidor de distância para, por exemplo, medir terremos sem precisar locomover-se ou utilizar uma fita métrica (MONK, 2014).

Além da placa de Arduino e dos sensores associados a ela, outro elemento extremamente importante para que o Arduino possa operar é o *software* que é programado baseado nele. Os *software* são baseados em algoritmos que no Android são chamados de *sketchs*. Neste capítulo vamos conhecer o código utilizado para acender e apagar um LED, bem como a estrutura de *hardware* necessária para este fim. Além da placa de Arduino para que você possa manipular o LED, você vai precisar de 1 LED, 1 resistor de 220 ohms ou 120 ohms (dependendo do LED que for utilizar) e 1 *protoboard*.

Protoboard

A *protoboard* é uma placa para prototipação utilizada para inserir os componentes eletrônicos simulando um circuito, sem necessidade de soldar os componentes na placa. A Figura 6 apresenta uma *protoboard*, onde você pode ver que a placa tem seus furos centrais marcados com as letras A até J nas linhas e números nas colunas. As colunas entre A e E e entre F e J são ligadas na vertical sem existência de um fio, e a corrente circula entre elas automaticamente. As colunas não se comunicam entre si verticalmente, sendo que caso se deseje que essa conexão exista é necessário colocar um cabo (*jumper*) ou mesmo um componente (como por exemplo, um resistor) entre as duas colunas. Já os furos das extremidades são ligados na horizontal. Ou seja, uma linha inteira é ligada em série como se fosse um único fio. Contudo, uma linha não é interligada com outra (CASSIO, [2019?]).

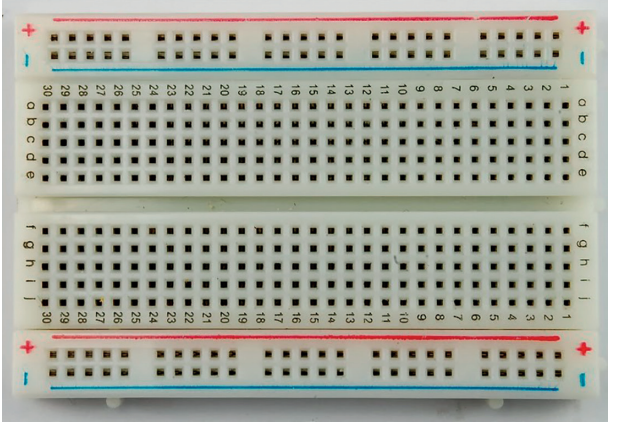

Figura 6. *Protoboard.*

Fonte: RaspDuino (2013, documento *on-line*).

LED e resistores

Diodos emissores de luz (LED, do inglês *light emissor diode*) são componentes que convertem energia elétrica em energia luminosa. Neste componente a energia flui apenas em um sentido e por este motivo é necessário, no momento de colocar o componente na placa, conectar a entrada de energia no polo positivo (com a ponta maior) e o terra (onde é fechado o circuito) no lado negativo (com a ponta menor) (CASSIO, [2019?]).

Os **resistores** são componentes eletrônicos que controlam e limitam a passagem de corrente pela placa e para os demais componentes. Ele é utilizado, por exemplo, para controlar a energia, impedindo que o LED queime.

Na Figura 7 podemos ver a conexão da *protoboard* do LED, do resistor com a placa de Arduino energizada com USB. O fio colorido é energizado, saindo da porta 12 do Arduino e o fio preto é neutro, saindo da porta GND. Estão ligados na mesma linha vertical a "perna" maior do LED (C15) e o fio colorido (b15). Abaixo da outra ponta do LED (C14) está o resistor (B14). Existe ainda um fio terra ligado na mesma linha horizontal que uma das pontas do resistor (-13) e à saída GND da placa do Arduino.

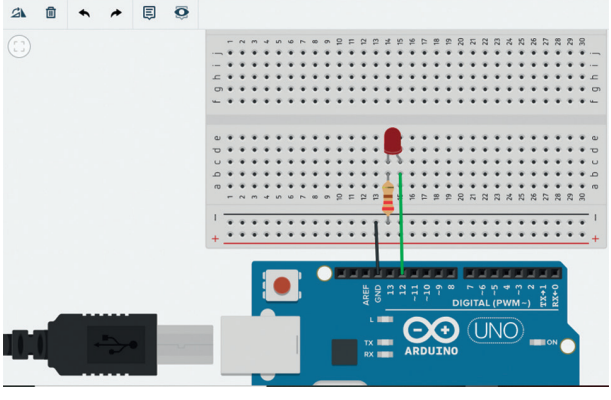

Figura 7. Circuito necessário para o acendimento do LED.
Fonte: Adaptada de AUTODESK (c2019).

Desenvolvendo o *software* para o Arduino

O Arduino pode se comunicar com seu computador através de uma conexão USB; enquanto o computador estiver conectado é possível enviar e receber mensagens. Um Arduino é diferente de um computador porque tem pouca memória, não contém sistema operacional nem interfaces para teclado, *mouse* ou monitor. Contudo é feito para atividades específicas, podendo, por exemplo, ligar ao Arduino um sensor para medir temperatura e um relé para ligar e desligar um aquecedor (MONK, 2015).

Saiba mais

Para que você possa desenvolver seus sistemas para Arduino, você precisa instalar o *software* do Arduino em seu sistema operacional de preferência. Todas as informações sobre a instalação do *software* do Arduino você encontra na documentação oficial do Arduino, no *site* oficial da plataforma.

A programação de Arduino não tem um grande nível de complexidade, pelo contrário, é muito mais simples construir um sistema para Arduino do que criar um *software* para *web* que depende de diversos fatores associados, como responsividade e portabilidade das páginas, por exemplo. Vamos analisar o código necessário para acender e apagar um LED com Arduino. Você pode ver a seguir que são necessárias apenas dez linhas de código.

```
1.  int led = 13;
2.  void setup() {
3.      pinMode(led, OUTPUT);
4.  }
5.  void loop() {
6.      digitalWrite(led, HIGH);
7.      delay(1000);
8.      digitalWrite(led, LOW);
9.      delay(1000);
10. }
```

Na linha 1 é criada uma variável que recebe o número da porta digital do Arduino em que está conectado o LED. Na linha 2 foi iniciada a função `setup` que é uma das funções principais do Arduino. Esta função é executada apenas uma vez, e com ela são realizadas as configurações iniciais do Arduino. Na linha 3 é configurada a porta digital 13 como SAÍDA de energia, isto é feito através da função `pinMode`, que exige dois parâmetros separados por vírgula: o primeiro é o número da porta e o segundo é o tipo que será a porta, ou seja, ENTRADA de energia (`INPUT`) ou SAÍDA (`OUTPUT`). Como queremos fornecer energia para acender um LED, a porta 13 deverá ser do tipo `OUTPUT` (CASSIO, [2019?]).

Na linha 5 é iniciada a função `loop`, e na linha 6 é feito com que o Arduino ligue a porta digital 13, utilizando a função `digitalWrite`. Esta função é utilizada exclusivamente para escrever valores para LIGAR (`HIGH`) e DESLIGAR (`LOW`) portas digitais. Essa função tem dois parâmetros, o primeiro é o número da porta, e o segundo, se esta deverá ficar ligada ou desligada. Na linha 7 é utilizada uma função `delay` para o Arduino efetuar uma parada, e isto é configurado conforme o parâmetro enviado par a função em milissegundos. Nas linhas 8 e 9 o processo é repetido, mas para apagar a luz que foi acesa (CASSIO, [2019?])

Para enviar o código criado para o Arduino é necessário utilizar o *software* que foi instalado em seu sistema operacional, o mesmo que foi utilizado para programar. Nele deve ser localizado o botão de *upload* de arquivo. Para que esse *software* possa funcionar, o Arduino precisa estar conectado na porta USB.

Utilizando Arduino, com um investimento financeiro baixo você pode construir coisas que facilitarão a sua vida e o seu trabalho. Utilizando os primeiros passos demonstrados aqui você pode construir os circuitos Arduino de sua preferência. Utilize a sua criatividade e facilite a sua vida com Arduino.

Referências

AUTODESK. *Tinkercad*. c2019. Disponível em: https://www.tinkercad.com. Acesso em: 18 maio 2020.

CASSIO, H. *Como acender LEDs com Arduino em 4 passos*. [2019?]. Disponível em: https://protovie.com/2019/03/13/como-acender-leds-com-arduino-em-4-passos/. Acesso em: 18 maio 2020.

CORELIS. *SPI tutorial*. [201–]. Disponível em: https://www.corelis.com/education/tutorials/spi-tutorial/. Acesso em: 18 maio 2020.

ETACARINAE. *Arduino 1*: introdução. 2017. Disponível em: http://eletronicaparaartistas.com.br/arduino-1-introducao. Acesso em: 18 maio 2020.

MADEIRA, D. *Protocolo I2C*: comunicação entre Arduinos. 2017. Disponível em: https://portal.vidadesilicio.com.br/i2c-comunicacao-entre-arduinos/. Acesso em: 18 maio 2020.

MALVINO, A.; BATES, D. J. *Eletrônica*: diodos, transistores e amplificadores. 7. ed. Porto Alegre: AMGH, 2011.

MONK, S. *Programação com Arduino*: começando com sketches. 2. ed. Porto Alegre: Bookman, 2017. (Tekne).

MONK, S. *Programação com Arduino II*: passos avançados com sketckes. Porto Alegre: Bookman, 2015. (Tekne).

MONK, S. *Projetos com Arduino e Android*: use seu smartphone ou tablet para controlar o Arduino. Porto Alegre: Bookman, 2014. (Tekne).

RASPDUINO. *Breadboards, resistências e circuitos*: as bases. 2013. Disponível em: http://raspduino.blogspot.com/2013/09/breadboards-resistencias-e-circuitos-as.html. Acesso em: 18 maio 2020.

Leitura recomendada

DIAS, K. *Comunicação serial Java + Arduino*. 2014. Disponível em: https://www.embarcados.com.br/comunicacao-serial-java-arduino/. Acesso em: 18 maio 2020.

Fique atento

Os *links* para *sites* da *web* fornecidos neste livro foram todos testados, e seu funcionamento foi comprovado no momento da publicação do material. No entanto, a rede é extremamente dinâmica; suas páginas estão constantemente mudando de local e conteúdo. Assim, os editores declaram não ter qualquer responsabilidade sobre qualidade, precisão ou integralidade das informações referidas em tais *links*.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS