RBECT
Revista Brasileira de Ensino de Ciência e Tecnologia

Revista Brasileira de Ensino de Ciência e Tecnologia
ISSN: 1982-873X

https://periodicos.utfpr.edu.br/rbect

Construção de um transmissor e de um receptor de código Morse através de sinais luminosos com uma placa Arduino

Thiago Costa Caetano
tccaetano@unifei.edu.br
0000-0002-2304-487X
Universidade Federal de Itajubá, Itajubá, Minas Gerais, Brasil.

Newton de Figueiredo Filho
newton@unifei.edu.br
0000-0001-6833-3125
Universidade Federal de Itajubá, Itajubá, Minas Gerais, Brasil.

Camila Cardoso Moreira
milacardoso.fisica@gmail.com
0000-0003-1118-0687

RESUMO

Neste artigo apresentamos uma atividade que tem sido desenvolvida com estudantes do primeiro ano do curso de Licenciatura em Física da Universidade Federal de Itajubá: a construção de um transmissor e de um receptor de código Morse através de sinais luminosos utilizando a placa Arduino. Descrevemos em detalhes a construção dos circuitos que são utilizados e também os códigos, todos disponíveis *on-line* gratuitamente. A atividade faz parte da disciplina "Prática de Ensino de Física II – FIS262" que teve sua ementa modificada no ano de 2016 para tratar de tópicos relacionados ao uso de tecnologias no ensino de Ciências. Seu principal objetivo é introduzir conceitos básicos tanto de programação como de circuitos de uma forma contextualizada, além de promover embasamento teórico e discussões acerca da temática.

PALAVRAS-CHAVE: Arduino. Código Morse. Tecnologias no ensino.

INTRODUÇÃO

O conteúdo programático da disciplina "Prática de Ensino de Física II – FIS262", inserida no segundo período da matriz curricular mais recente do curso de Licenciatura em Física da Universidade Federal de Itajubá/Brasil, foi modificado no ano de 2016 e passou a abranger tópicos ligados ao uso de novas tecnologias no ensino. Tal modificação foi motivada pelos inúmeros estudos que revelam que, de uma forma geral, os professores se sentem inseguros quanto à utilização dessas tecnologias em sala de aula ou não têm familiaridade com os recursos, algo que quase sempre está relacionado a uma deficiência em sua formação (PERALTA e COSTA, 2007; SANCHES, RAMOS e COSTA, 2014; COSTA, RIBEIRO e FERREIRA, 2016; CHIOSSI e COSTA, 2018). Sabe-se que o contato com essas tecnologias não é algo que deve ocorrer de forma estanque durante a formação inicial do professor, mas sim estar presente durante toda essa fase e persistir mesmo após ela através de um processo de formação continuada, tendo em vista a velocidade com que ocorrem os avanços no campo tecnológico (BAPTISTA da SILVA, 2013; NUNES, GUERINO e STANZANI, 2014; VASCONCELOS e OLIVEIRA, 2017; FONSECA, 2018). Apesar disso, a criação de um espaço dedicado a estudos mais pontuais, focados e com mais aprofundamento durante a formação inicial é válida e sem dúvida representa um elemento fundamental à sua formação. Espera-se contribuir dessa forma para que os futuros professores se familiarizem com os recursos tecnológicos – não somente aqueles mais básicos – e para que se municiem não só do ponto de vista instrumental, mas também do ponto de vista teórico-metodológico, permitindo assim que sejam capazes de integrar os recursos tecnológicos em sua prática de maneira consciente e significativa.

De forma simplificada, o conteúdo da disciplina passou a apresentar a seguinte estrutura híbrida, onde alterna-se entre aulas presenciais, realizadas em um laboratório da universidade, e aulas disponibilizadas no Ambiente Virtual de Aprendizagem (AVA):

1. (Presencial/Laboratório) – Introdução ao Arduino: noções básicas, identificando componentes, nomenclatura básica, estrutura básica de um algoritmo, biblioteca de exemplos;
2. (AVA) – Determinação da aceleração da gravidade: avaliar diferentes formas de realizar esse experimento analisando a utilização dos recursos tecnológicos em cada uma delas e mantendo o foco na aprendizagem de conceitos. Mostrar que nem sempre mais tecnologia traz mais resultados;
3. (Presencial/Laboratório) – Utilizando uma *protoboard*: Construção de circuitos simples, noções básicas sobre os componentes eletrônicos e ligação com o Arduino;
4. (AVA) – Utilização do programa *Fritzing* e discussão sobre a aprendizagem de conceitos com auxílio de experimentos;
5. (Presencial/Laboratório) – Aquisição de dados: o uso de sensores e o tratamento de dados, modelos teóricos envolvidos;
6. (AVA) – Atividades investigativas: o *Inquiry Based Science Education* como uma estratégia de ensino articulada aos recursos experimentais e tecnológicos;

7. (Presencial/Laboratório) – Circuitos integrados e sensores;
8. (AVA) – Experimento ou instrumento: o que faz com que um conjunto de instrumentos de medidas se torne um experimento? Análise do Laboratório Remoto de Física da Unifei[1], o uso da internet articulada aos recursos vistos;
9. (Presencial/Laboratório) – Saída de dados: controlando elementos através de sinais, *Physical Programming*;
10. (AVA) – Alguns referenciais bibliográficos para a prática;
11. (Presencial/Laboratório) – Motores e atuadores mecânicos: programação específica e eletrônica envolvida; desenvolvimento de circuitos analógicos e digitais em PCB;
12. (AVA) – Discussão e aprofundamento teórico;
13. (Presencial/Laboratório) – Aulas dedicadas à apresentação dos trabalhos finais da disciplina: projetos desenvolvidos pelos estudantes a partir dos elementos constituintes da disciplina.

Nesse artigo apresentamos em detalhes uma das atividades que é realizada pelos estudantes durante o curso: a construção de um transmissor e de um receptor de código Morse por meio de sinais luminosos utilizando uma placa Arduino, componente este que tem sido amplamente empregado em atividades de ensino devido ao seu aspecto notavelmente didático. O leitor poderá encontrar mais exemplos através dos trabalhos de Cavalcante, Tavolaro e Molisani (2011); Souza *et al*. (2011); Carvalho e Amorim (2014); Rocha, Marranghello e Lucchese (2014); da Silva *et al*. (2016) e Silveira e Girardi (2017).

A atividade apresentada aqui foi concebida e planejada de forma a introduzir conceitos básicos de programação e de circuitos ao mesmo tempo em que apresenta uma dimensão histórico-teórica, que permite compreender, de forma contextualizada, a necessidade de se codificar informações utilizando um sistema binário; e uma componente lúdica, ao passo que a dinâmica adotada em sala de aula baseia-se na introdução de certos desafios entre os estudantes, o que, conforme observado, tem grande potencial para despertar o seu envolvimento e sua motivação.

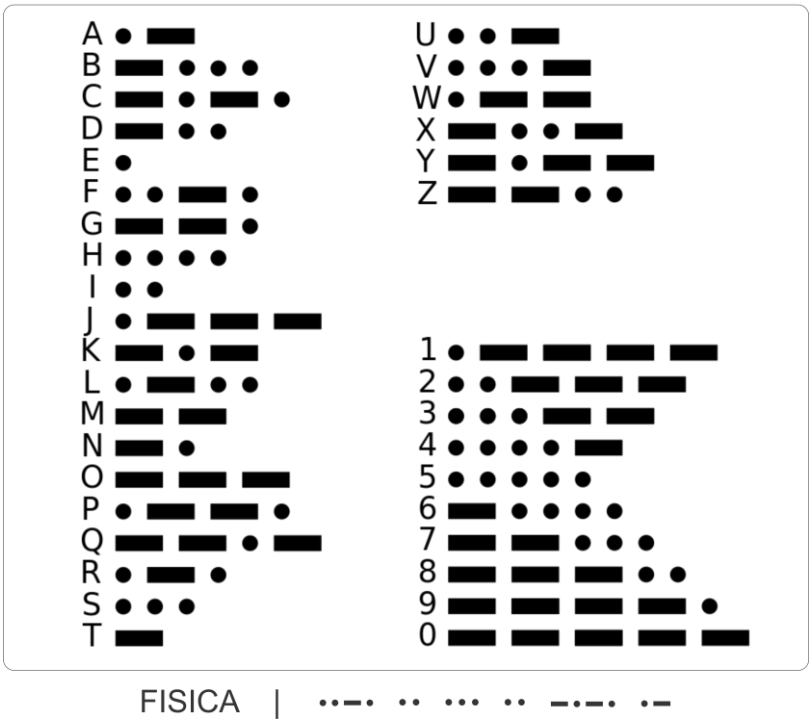
A mesma atividade é utilizada em diferentes momentos do curso, explorando-se diferentes recursos ao passo que os estudantes avançam com o conteúdo. Os algoritmos utilizados vão gradualmente incorporando novos elementos e os circuitos também ficam pouco a pouco mais elaborados, embora neste caso não exista nada complexo do ponto de vista eletrônico. Para descrever a atividade como um todo será feita antes uma breve introdução sobre o código Morse, após o que passaremos a apresentar o transmissor. Esta parte é constituída por duas seções que tratam respectivamente da primeira e da segunda versão do código, construídas em momentos diferentes do curso empregando-se diferentes técnicas de programação. O receptor (leitor) de código Morse com sinais luminosos é apresentado em seguida e, finalmente, são feitas algumas breves considerações a partir da experiência com o curso ao longo destes anos em que vem sendo oferecido.

O CÓDIGO MORSE

A partir da segunda metade do século XVIII, diversos experimentos com a eletricidade realizados na Europa levaram à construção de equipamentos de comunicação capazes de transmitir mensagens a distâncias curtas, que foram genericamente denominados telégrafos elétricos (FAHIE, 1884). Em 1837 foi instalada a primeira linha telegráfica da Europa, que ligava Londres a Birmingham, com cerca de 160 km de comprimento. Já nos Estados Unidos a primeira linha foi instalada em 1844 e ligava Washington a Baltimore (ITU, 2015). No Brasil o telégrafo elétrico chegou em 1852 quando foi inaugurada no Rio de Janeiro a linha que conectava a Quinta Imperial ao Quartel General do Exército (MACIEL, 2001).

Nos anos que se seguiram, as conexões telegráficas experimentaram uma rápida expansão, mas não havia ainda uma padronização para o modo com que as mensagens eram codificadas. Essa padronização somente viria a ser adotada em 1865, com a assinatura da primeira Convenção Telegráfica Internacional por delegados de vinte países reunidos em Paris. Nesse evento foi fundada a União Telegráfica Internacional (ITU), que recebeu a incumbência de supervisionar as futuras versões desse documento (ITU, 2015).

Figura 1 – Representação gráfica dos caracteres alfanuméricos do Código Morse aprovada em 2009 pela ITU



Fonte: Adaptado pelos autores através da plataforma Wikimedia Commons (2020).

O padrão adotado teve por base o código usado por Samuel Morse em 1844 que, por essa razão, é denominado Código Morse. Cada letra, algarismo ou sinal de pontuação é representado por uma sequência de sinais curtos e longos separados por pausas de duração variável. Esses sinais podem ser sonoros, luminosos ou elétricos e são representados graficamente por pontos e traços, como mostra a Figura 1. A duração dos sinais e das pausas obedece ao seguinte padrão:

a) um traço equivale a três pontos;

b) a pausa entre dois sinais que formam um caractere é igual a um ponto;

c) a pausa entre dois caracteres é igual a três pontos;

d) a pausa entre duas palavras é igual a sete pontos.

A utilização do Código Morse para o envio de mensagens transmitidas por estações de rádio teve um grande desenvolvimento ao longo do século XX, mas foi gradativamente substituída pela telefonia e pelos modos digitais. Atualmente ele é utilizado principalmente na radionavegação aérea, em que estações em terra transmitem sua posição por meio de radiofaróis não direcionais para orientação das aeronaves (BRASIL, 2017). Também é empregado por estações de radioamadores em regiões do espectro eletromagnético especificamente alocadas para essa finalidade (BRASIL, 2018). Na navegação marítima o Código Morse é utilizado na forma de pulsos de luz para orientação de embarcações (USCG, 2019).

PRIMEIRA VERSÃO DO CÓDIGO

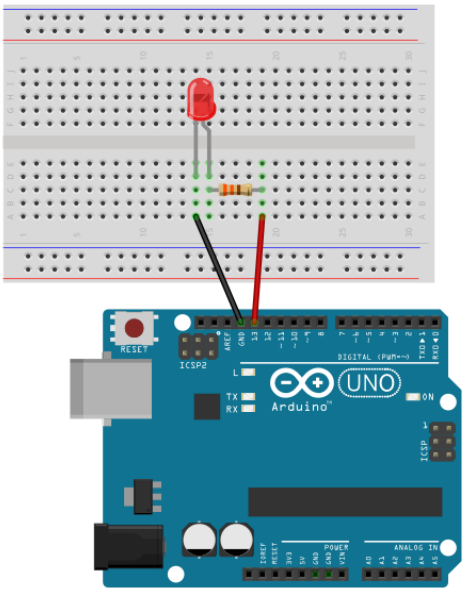
A primeira versão do código é aplicada logo na primeira aula da disciplina *Prática de Ensino de Física II – FIS262*, oferecida no segundo período conforme a matriz curricular do curso. Por essa razão, os comandos explorados nessa versão são os mais básicos possíveis e a estrutura do código não apresenta nenhuma sofisticação em termos de lógica ou de sintaxe. O objetivo é exercitar o uso de alguns comandos através da sua utilização repetitiva, tornando propícia a sua memorização e ao mesmo tempo familiarizando os estudantes com procedimentos corriqueiros como: compilar o código, depurá-lo e transferi-lo para o Arduino. Através deste exercício os estudantes também começam a desenvolver sua técnica para a programação, observando particularidades da linguagem utilizada, características da sua sintaxe, a grafia sensível ao uso de maiúsculas e minúsculas, pensamento lógico, estruturas de controle, entre diversos outros aspectos.

O programa *Blink* é utilizado como ponto de partida para o desenvolvimento do código. Esse é apenas um entre os muitos exemplos que podem ser encontrados no ambiente de desenvolvimento do próprio Arduino (IDE – *Integrated Development Environment*). O programa ativa e desativa a diferença de potencial em uma das portas do Arduino de forma intermitente, fazendo-a alternar entre 0 e 5 Volts. Se conectarmos um LED a essa porta, da forma como mostra a Figura 2, ele irá piscar. As placas Arduino já vêm com um LED conectado à porta 13, o que permite que um programa como o *Blink* seja testado sem a necessidade de nenhum componente externo. Mesmo o código que os estudantes irão construir poderia ser testado dessa forma. Contudo, como um dos objetivos da disciplina é que os estudantes tenham familiaridade com a montagem de circuitos, é pedido que eles construam o circuito mostrado na Figura 2, como uma espécie de introdução, até porque eles deverão fazer isso ao longo de toda a disciplina e os circuitos serão mais e mais complexos.

Outro fator pelo qual é conveniente que os estudantes construam o circuito é o fato de que, na atividade proposta com o código Morse, posteriormente, é pedido a um colega que tente decifrar a mensagem que será transmitida através de sinais luminosos e isso torna-se uma tarefa muito mais fácil com um sinal

luminoso mais potente. Aproveita-se a oportunidade para a apresentação de algumas informações e mesmo para a introdução e discussão de algumas características do circuito como: componentes com e sem polaridade, funcionamento do LED, código de cores de resistores, potência máxima suportada por um resistor, precisão relacionada à resistência nominal, alimentação do circuito, características do Arduino, cuidados com o equipamento, utilização correta das *protoboards*, associação em série e em paralelo utilizando uma *protoboard*, entre outros.

Figura 2 – Ilustração da montagem do circuito utilizado para o transmissor



Fonte: Elaborado pelos autores através do *software Fritzing* (2019).

Antes de iniciar a construção do código é preciso decidir que mensagem será transmitida. De acordo com o esquema adotado na disciplina, os alunos estão organizados em duplas em cada bancada de experimentos. É pedido a cada dupla que escolha o nome (primeiro nome ou sobrenome), de um cientista famoso e mantenha essa informação em segredo. Essa será a mensagem a ser transmitida e outra dupla irá tentar decifrá-la. Assim é possível avaliar se a dupla obteve sucesso na construção do código. Se a mensagem cifrada contiver falhas, não será possível lê-la, ou a informação lida não fará sentido e saberemos que a dupla cometeu algum erro.

À guisa de ilustração, iremos supor a partir deste ponto que a mensagem a ser transmitida na forma de código Morse seja o nome NEWTON. A construção do código a partir do programa *Blink* consiste em replicar os comandos já existentes na sequência correta empregando os intervalos apropriados. Por convenção, na disciplina, a duração do ponto (dit) é de 333 ms. Por conseguinte, o traço (dah) tem duração de 1000 ms. Em ambos, os sinais o LED deve permanecer aceso. O intervalo de tempo entre sinais dit e dah, em uma mesma letra, corresponde à duração de um dit. O intervalo de tempo entre duas letras corresponde à duração de um dah. Seja entre sinais ou entre letras, o LED deve permanecer apagado. Ao final de uma palavra, o LED deve permanecer apagado por um intervalo equivalente a sete vezes a duração de um dit.

No início do código são feitas algumas definições para simplificar a sua construção. Por exemplo, atribuímos o numeral 333 como significado para o termo dit. Ou seja, sempre que esse termo for empregado no código, o compilador entenderá que o seu significado é o número inteiro positivo 333. Para efeitos práticos, seria o equivalente a declarar uma variável com nome dit e atribuir a ela o valor 333. Essa é uma prática largamente empregada em programação. Considere, a título de exemplo, que a variável dit seja utilizada cem vezes em todo o código. Para alterar o seu valor, basta alterar a sua declaração no início do código utilizando o novo valor. Mas se, ao invés disso, tivéssemos empregado o seu valor cem vezes em todo o código, caso fosse necessário alterá-lo, teríamos que realizar a alteração em cada ponto do código onde o valor foi empregado. Imagine isso para um programa contendo 10 mil linhas!

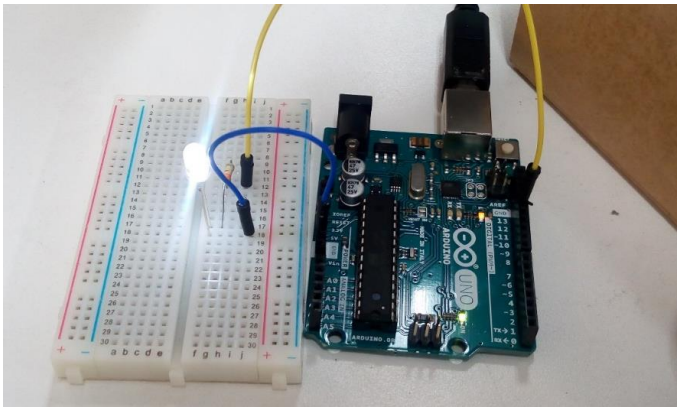
Todo código para Arduino deve conter obrigatoriamente as funções *setup* e *loop*. A primeira é executada uma única vez quando a placa é conectada à energia ou o botão Reset é pressionado. É o local apropriado para configurarmos o funcionamento das portas através da função *pinMode*. Nesse caso, a porta em que o LED está conectado deve funcionar no modo *output* (saída), pois irá fornecer uma tensão para o circuito. A segunda, a função *loop*, é executada ciclicamente enquanto a placa estiver conectada à energia e é nela que iremos inserir os comandos para acender e apagar o LED (ativar ou desativar a tensão na porta em que o LED está conectado). Como é possível perceber, ao colocarmos a mensagem cifrada dentro da função *loop*, ela irá se repetir indefinidamente. A função *digitalWrite* é utilizada para controlar a tensão na porta e a função *delay* é empregada para interromper a execução do código por um tempo determinado, fornecido em milissegundos. Essa última é utilizada para controlar os intervalos de espera em que o LED deve permanecer aceso ou apagado. O código completo que foi desenvolvido para transmissão da mensagem NEWTON através de código Morse com sinais luminosos pode ser obtido através do endereço http://labremoto.unifei.edu.br/materiais. Embora tenhamos optado por não o inserir aqui, é recomendado que o leitor consulte o código para melhor compreensão dos aspectos que serão discutidos a seguir.

Esse é o momento adequado para algumas considerações relativas à programação. Diferentemente do que acontece na maioria dos contextos em que se discute programação, o código a ser elaborado aqui é capaz de promover uma ação no meio físico, ou interagir com ele de alguma forma, e esse aspecto tem sido a pedra angular do termo *Physical Programming*. No caso da atividade proposta aqui, com os instrumentos que estão sendo utilizados, há um conjunto específico de funções que tornam isso possível, sendo as mais utilizadas: *pinMode*, *digitalWrite/digitalRead* e *analogWrite/analogRead*. Essas funções, em conjunto com o hardware adequado, oferecem inúmeras possibilidades, desde ações mecânicas até a leitura das mais diversas grandezas em sistemas físicos. Um bom exemplo de como o domínio dessas funções básicas e dos princípios por trás do *Physical Programming* podem levar ao desenvolvimento de projetos sofisticados e ao surgimento de novas tecnologias é o desenvolvido um laboratório didático de Física em que todos os experimentos podem ser controlados à distância através da comunicação via internet. Recomenda-se visitar o site http://labremoto.unifei.edu.br para informações mais detalhadas do projeto. Além disso, ao leitor que tiver interesse, recomenda-se consultar mais informações sobre *Physical Programming* e tutoriais de projetos através do endereço

https://learn.sparkfun.com/tutorials (essa é apenas uma sugestão de fonte a ser consultada).

A Figura 3 mostra uma imagem real do transmissor construído por um dos alunos da disciplina. Foi utilizada uma *protoboard* pequena, pois há pouquíssimos componentes nesse projeto: um LED e um resistor ligados em série. O circuito está alimentado com 5.0 V, fornecido pelo próprio Arduino. O LED não deve ser conectado diretamente à fonte, ou ele irá dissipar tanta potência quanto lhe for fornecida e será danificado. Para limitar a corrente utiliza-se um resistor conectado em série com ele, digamos, um resistor de 330 Ohms.

Figura 3 – Imagem real do circuito transmissor



Fonte: Autores (2020).

SEGUNDA VERSÃO DO CÓDIGO

Embora a primeira versão funcione perfeitamente e seja suficiente para completar a tarefa de transmitir uma mensagem através de código Morse, é possível explorar um pouco mais os recursos da programação em prol do ensino de técnicas relativamente mais sofisticadas e de otimização do código, as quais serão úteis para a construção de projetos mais complexos posteriormente. É possível notar que, caso queiramos alterar a mensagem na primeira versão do código, é necessário reescrevê-lo quase totalmente. Isso torna o código demasiadamente específico, porém, essa versão cumpre o seu papel quando nos limitamos a questões didáticas.

Passaremos a apresentar agora uma segunda versão do código, mais versátil, que é construída em uma fase posterior da disciplina, quando os estudantes já possuem familiaridade com os fundamentos da programação. A essa altura espera-se que eles utilizem as funções básicas sem dificuldades e que eles sejam capazes de criar e manipular variáveis com certa destreza. Nessa versão faz-se a introdução da técnica de criação e utilização de funções específicas.

A criação de funções é útil para tornar o código e a programação mais simples e deve ser empregada sempre que existirem blocos de comandos recorrentes, como é o caso da primeira versão. Independentemente da mensagem escolhida, a mensagem transmitida é cifrada em termos de apenas dois sinais: dit e dah. Sempre que o programador realiza um dit, ele escreve os mesmos comandos, assim como o faz para o dah. O código pode tornar-se muito mais compacto e o trabalho de programação mais simples se o programador construir funções

específicas para cada sinal. Como no caso da primeira versão, o código completo da segunda versão pode ser obtido no endereço http://labremoto.unifei.edu.br/materiais.

As linhas iniciadas em #define foram mantidas. As linhas seguintes definem protótipos para as funções f_dah e f_dit, nomes estes que podem ser definidos pelo programador. Observa-se, contudo, que há algumas regras que se aplicam aos nomes das funções: i) não são permitidos espaços em branco; ii) não se deve utilizar um numeral no início do nome; iii) não se deve utilizar caracteres especiais e iv) deve-se ter o cuidado para não escolher o nome de uma função já existente.

```
void f_dah( bool lend = false );
void f_dit( bool lend = false );
```

A declaração inicial *void* é utilizada antes do nome da função, no momento da sua declaração, para indicar que aquela função não irá retornar nenhum valor. Entre parênteses são informados aquilo que chamamos de parâmetros da função. Nesse caso, cada uma das duas funções aceita um parâmetro, o qual recebe o nome *lend* (uma contração de *letter end*) dentro da função. O fato de esse parâmetro ter um valor atribuído a ele – o valor *false* – na declaração acima, significa que esse parâmetro é opcional. Ou seja, quando a função for utilizada, o programador poderá ou não informar o parâmetro. Caso não o faça, ele receberá o valor *false* automaticamente. Agora que o protótipo foi definido, é preciso definir a função em si. Optou-se por fazer essa definição na última parte do código, mas isso não constitui uma regra. Vejamos como fica a função f_dah.

```
void f_dah( bool lend ){
  digitalWrite( led, HIGH );
  delay( dah );
  digitalWrite( led, LOW );
  if ( lend ) {
    delay( dah );
  } else {
    delay( dit );
  }
}
```

Note que a função f_dah, por exemplo, acende o LED, espera um tempo equivalente à duração de um dah e então o apaga. O tempo que o LED ficará apagado depende do valor da variável *lend*. Se *lend* for igual a *false*, que é o seu valor padrão, então o código será interrompido por um tempo equivalente à duração de um dit. Esse é o tempo que o LED ficará apagado. Se, por outro lado, o valor de *lend* for *true*, então o LED permanecerá apagado por um tempo equivalente à duração de um dah. Essa característica da função é útil pois esse intervalo varia dependendo da posição do sinal na mensagem. Por exemplo, entre sinais de uma mesma letra, o intervalo em que o LED permanece apagado equivale à duração de um dit. Já entre letras na mensagem, a duração equivale a um dah.

Uma vez que as funções foram definidas adequadamente, basta utilizá-las conforme a mensagem escolhida. O código acima baseia-se no exemplo anterior, em que a mensagem adotada é a palavra NEWTON. Ao final da palavra adota-se

um intervalo cuja duração é sete vezes a duração de um dit, no qual o LED permanece apagado.

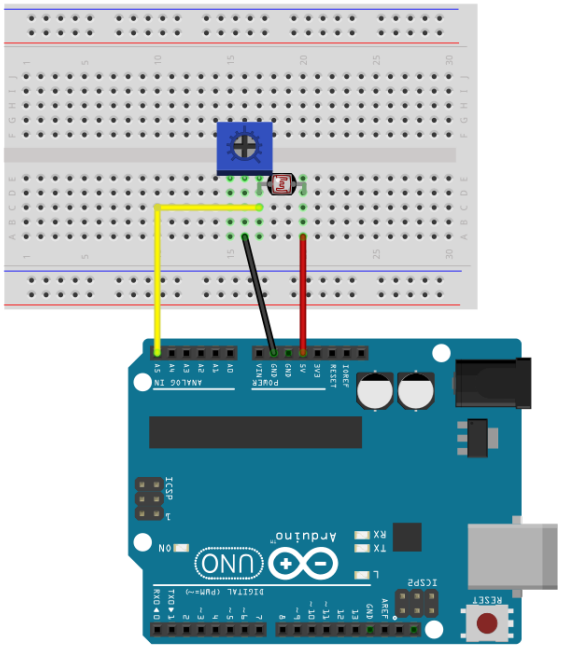
RECEPTOR DE CÓDIGO MORSE POR MEIO DE SINAIS LUMINOSOS

Nas aulas finais da disciplina propõe-se a construção de um leitor de código Morse baseado em sinais luminosos, para ser utilizado com o transmissor que foi apresentado nas sessões anteriores. Apresentaremos a sua construção e a sua programação separadamente.

CALIBRAÇÃO E CONSTRUÇÃO

A Figura 4 contém uma ilustração do circuito utilizado para a calibração do receptor, que consiste em um LDR (*Light Dependent Resistor*) ligado em série com um resistor variável de 10 kΩ. O circuito é alimentado por uma tensão de 5.0 Volts, fornecida pelo próprio Arduino. Conecta-se o terminal do LDR a uma entrada analógica do Arduino, nesse caso, à porta A5, conforme mostra a figura. Uma observação: existem diversos modelos de resistores variáveis, ou potenciômetros, ou ainda, *trimpots*. O modelo mostrado na figura é o que está disponível no programa *Fritzing*. Já a imagem real da montagem irá mostrar um modelo diferente, mas que possui a mesma função.

Figura 4 – Circuito utilizado para calibração do receptor



Fonte: Elaborado pelos autores através do *software Fritzing* (2019).

A calibração do receptor é necessária para determinar o nível de detecção do sinal luminoso, permitindo eliminar a contaminação devida à luz ambiente e garantindo que apenas o sinal da fonte seja detectado. Dessa forma não será preciso um ambiente escuro para que o projeto funcione corretamente.

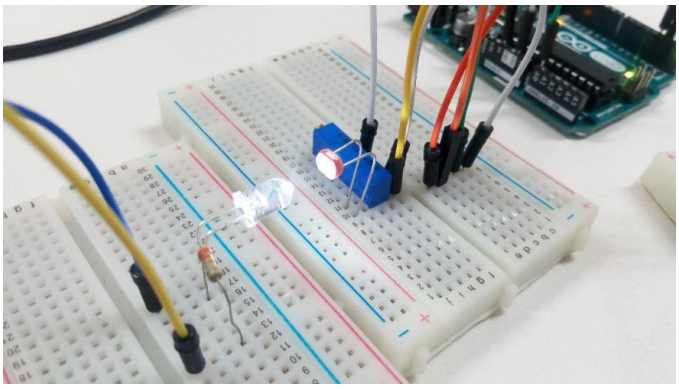
O código abaixo deve ser carregado para o Arduino do receptor. Em seguida, abre-se o monitor da porta serial, disponível no menu *ferramentas* (*tools*) ou através das teclas de atalho Ctrl + Shift + M, e acompanha-se os resultados das leituras.

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  Serial.print("Analog signal: ");
  Serial.print(  analogRead(A5) );
  Serial.print("   ddp:");
  Serial.print( analogRead(A5)*5./1023 );
  Serial.print("    Digital signal: ");
  Serial.println( digitalRead(A5) );
  delay(100);
}
```

É possível observar que, conforme variamos a resistência, obtemos valores diferentes para o sinal analógico e para a ddp naquele ponto do circuito que está conectado à porta A5 do Arduino. Começando com um sinal analógico igual a zero (e uma ddp nula, consequentemente) veremos que o sinal digital muda de valor em algum ponto quando aumentamos o nível de sinal progressivamente, passando de 0 para 1. Esse ponto é o que definimos como limiar de detecção. A calibração do receptor é feita colocando-se o nível de sinal analógico próximo deste limiar e abaixo dele. Para ter certeza que a calibração foi realizada corretamente, é recomendado fazer um teste utilizando o transmissor, conforme mostra a Figura 5. No monitor da porta serial, observe se o sinal digital muda de valor conforme o LED acende e apaga. Em caso positivo, a calibração está boa. Caso contrário, tente ajustar a resistência do receptor até conseguir o resultado desejado.

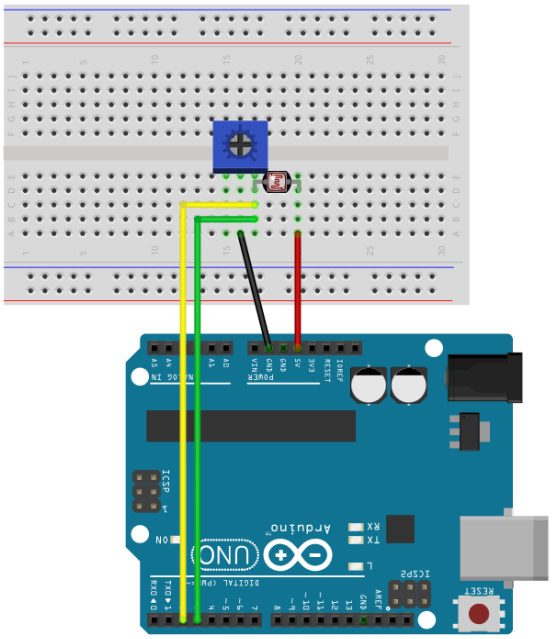
Figura 5 – Imagem dos circuitos transmissor e receptor



Fonte: Autores (2019).

O próximo passo após a calibração é a montagem do circuito final mostrado na Figura 6. Note que ambas as portas, 2 e 3 do Arduino estão conectadas ao mesmo terminal do LDR.

Figura 6 – Ilustração para a montagem do receptor



Fonte: Elaborado pelos autores através do *software Fritzing* (2020).

PROGRAMAÇÃO

O código para o receptor agrega diversos novos elementos da programação. Por essa razão é tratado apenas ao final da disciplina. Entre os elementos introduzidos pelo código destacamos o uso de variáveis do tipo voláteis e o uso da função attachInterrupt.

De forma breve, essa função possibilita que a execução de uma função definida no código seja controlada por um sinal externo, denominado *trigger*. No código a seguir, a função riseEvent é executada sempre que o sinal digital na porta 2 passa de 0 para 1 (por isso a calibração é um procedimento importante). De forma análoga, a função fallEvent é executada sempre que o sinal digital na porta 3 passa de 1 para 0 (contrário ao que acontece com o sinal na porta 2).

Agora analisemos o comportamento do código em termos do que é observado no circuito. Quando o LED do transmissor acende o nível de sinal digital no receptor muda de 0 para 1 fazendo com que a função riseEvent seja executada e o instante em que isso ocorre é registrado na variável *start*. Quando o contrário acontece a função fallEvent é executada e o tempo transcorrido desde *start* é computado. O resultado é armazenado na variável *interval* e corresponde ao tempo que o LED do transmissor permaneceu aceso. O valor da variável *interval* é interpretado dentro da função loop. Se menor que 500 ms, será impresso um sinal dit no monitor da porta serial, caso contrário será impresso um sinal dah.

A mesma lógica – mas de forma invertida – é empregada para determinar o tempo que o LED permanece apagado. Quando o sinal luminoso do transmissor cessa e o sinal digital no receptor muda de 1 para 0, a função fallEvent é executada e o instante em que isso acontece fica registrado na variável *start_2*. Quando o LED acende novamente, o sinal digital no receptor muda de 0 para 1 e a função riseEvent é executada, computando o tempo transcorrido desde *start_2* e

registrando o resultado na variável *interval_2*. Esse valor será analisado dentro da função loop para determinar se o tempo que o LED permaneceu apagado indica o intervalo entre dois sinais em uma mesma letra, entre duas letras ou o fim da mensagem.

Embora a descrição do comportamento do código tenha sido feita de forma compartimentada, na tentativa de tornar o texto o mais claro possível, é fácil notar que ambas as ações descritas anteriormente ocorrem paralelamente. Ou seja, quando a função riseEvent é executada, ela não só registra o momento em que a execução teve início como também determina o tempo transcorrido desde a última execução da função fallEvent. Ao abrir o monitor da porta serial em que o receptor está conectado, com o receptor e o transmissor posicionados de acordo com a Figura 5, lê-se o seguinte resultado:

```
– . . .– – – – – – . 
```

CONSIDERAÇÕES FINAIS

A atividade descrita nesse trabalho foi realizada pelo terceiro ano consecutivo em 2018, tendo sofrido certos ajustes do ponto de vista didático-pedagógico durante esse período, naturalmente. Como era de se esperar, sempre houve excelente aceitação por parte dos estudantes, o que é possível afirmar graças à experiência adquirida com a oferta do curso e às observações feitas. Isso apenas confirma o fato de que estes estudantes advêm de uma sociedade em que os recursos tecnológicos permeiam as mais variadas atividades do dia a dia, o que acaba por gerar entre eles uma expectativa de que estes recursos não estejam ausentes ou menos frequentes nas atividades ligadas ao ensino. Sobre esse aspecto é interessante salientar que a inserção – estratégica – de atividades como a que foi descrita neste trabalho logo nos períodos iniciais do curso de licenciatura representa um aspecto extremamente positivo pois responde a essa expectativa. O entusiasmo, a motivação e o interesse que tais atividades suscitam entre os estudantes são elementos imprescindíveis para o sucesso da sua formação.

Embora boa parte dos estudantes tenha familiaridade com o computador e com ferramentas básicas como processadores de texto e planilhas eletrônicas, o mesmo não pode ser dito com relação às linguagens de programação e à eletrônica. Nesse aspecto a atividade tem se mostrado didaticamente adequada, pois, mesmo estudantes sem nenhuma familiaridade com alguma linguagem de programação podem rapidamente compreender a sintaxe utilizada no caso da Arduino IDE e os princípios básicos de lógica de programação. A biblioteca de exemplos do próprio Arduino e o vasto acervo de tutoriais e materiais gratuitos disponíveis na internet certamente contribuem para uma aprendizagem acelerada. Quanto à eletrônica destaca-se que, apesar da simplicidade dos componentes eletrônicos que foram citados neste trabalho, a maioria dos estudantes possui apenas – quando possui – uma noção teórica sobre o seu funcionamento, de forma que atividades como a que foi descrita aqui contribuem para a consolidação do conhecimento que possuem. A simplicidade do projeto torna a atividade viável mesmo para aqueles que nunca tiveram contato com eletrônica, seja na prática ou seja teoricamente.

Com exceção da placa Arduino, todos os materiais empregados são baratos, custando apenas alguns centavos em alguns casos. Podem ser encontrados em quase todas as casas de materiais elétricos e eletrônicos ou adquiridos através da internet. A placa Arduino pode ser encontrada normalmente em lojas especializadas, lojas de materiais eletrônicos ou ligadas à robótica. Apesar de ser o item mais caro, há algumas alternativas mais baratas disponíveis no mercado e também pode ser adquirido através da internet.

Alternativamente, como é possível que o leitor tenha notado, parte da atividade pode ser realizada apenas com uma placa Arduino e um computador, não sendo possível, contudo, a construção do receptor. Essa alternativa pode ser a única opção em alguns casos, principalmente quando não se dispõe dos materiais ou de uma estrutura física adequada. Reitera-se que todos os códigos e esquemas de montagem citados neste trabalho estão disponíveis gratuitamente através do site https://labremoto.unifei.edu.br/materiais.

BUILDING A LIGHT-SIGN BASED MORSE CODE TRANSMITTER AND RECEIVER WITH AN ARDUINO BOARD

ABSTRACT

In this paper, we present an activity that has been developed amongst the students in the first year of the Physics undergraduate course at Federal University of Itajubá/Brazil: the development of the Morse code transmitter and receiver based on light signals using the Arduino board. We describe with details the circuits and the algorithms that were developed, all of it being available online freely. This activity is part of the subject "Physics Teaching Practice II – FIS262" which has been modified in 2016 to deal with topics concerning the technology in the Science Teaching. Its main goal is to introduce basic concepts from both programming and electronic circuits in a contextualized manner, besides promoting a theoretical background and discussions around the theme.

KEYWORDS: Arduino. Morse Code. Technologies in teaching.

NOTAS

1 O laboratório remoto pode ser acessado gratuitamente 24 horas por dia, todos os dias da semana, através do endereço https://labremoto.unifei.edu.br.

REFERÊNCIAS

BAPTISTA da SILVA, J. M., **TIC no ensino e na formação de professores: reflexões a partir da prática docente**, 2013. Dissertação (Mestrado em Educação) - Faculdade de Motricidade Humana, Universidade Técnica de Lisboa, 2013.

BRASIL. MINISTÉRIO DA DEFESA. **Manual Brasileiro de Inspeção em Voo**. Brasília: Departamento de Controle do Espaço Aéreo, p. 172, 2017.

BRASIL. **AGÊNCIA NACIONAL DE TELECOMUNICAÇÕES**. Resolução nº 697, de 28 de agosto de 2018.

CAVALCANTE, M. A.; TAVOLARO, C. R. C.; MOLISANI, E., Física com Arduino para iniciantes. **Rev. Bras. Ensino Fís**., São Paulo, vol. 33, n. 4, p. 4503, 2011.

CARVALHO, L. R. M.; AMORIM, H. S. Observando as marés atmosféricas: uma aplicação da placa Arduino com sensores de pressão barométrica e temperatura. **Rev. Bras. Ensino Fís.**, São Paulo, vol. 36, n. 3, p. 1-7, 2014.

CHIOSSI, R. R.; COSTA, C. S., Novas formas de aprender e ensinar: a integração das Tecnologias de Informação e Comunicação (TIC) na formação de professores da educação básica, **Texto Livre**, v. 11, n. 2, p. 160-176, Belo Horizonte, 2018

COSTA, F. J.; RIBEIRO, P. C.; FERREIRA, J. R., A Distância das Tecnologias Digitais de Informação e Comunicação do Ambiente Escolar e a Formação de Professores, **Revista Formação@Docente**, vol. 8, n. 2, Belo Horizonte, 2016

DA SILVA, S. L. *et al*. Avaliação do módulo da aceleração da gravidade com Arduino. **Caderno Brasileiro de Ensino de Física**, Florianópolis, vol. 33, n. 2, p. 619-640, 2016.

FAHIE, J.J**. A history of electric telegraphy, to the year 1837**. London: E.&F.N. Spon, 1884.

FONSECA, M. G. R., **As tecnologias de informação e comunicação na formação inicial de professores do 1º ciclo do ensino básico – crenças e perspectivas dos formadores**, Tese (Doutorado em Educação) - Instituto de Educação, Universidade de Lisboa, 2018.

ITU – INTERNATIONAL TELECOMMUNICATIONS UNION. **Paris, 1865: the birth of the Union**. Genebra: ITU, 2015.

MACIEL, L.A. Cultura e tecnologia: a constituição do serviço telegráfico no Brasil. **Revista Brasileira de História**, São Paulo, vol. 21, n. 41, p. 127-144, 2001.

NUNES, M. O.; GUERINO, M. F.; STANZANI, E. L., La utilización de las TIC en la formación continua: Iniciativas y experiencias presentes en la producción académica brasileña. **Revista Iberoamericana de Educación**, v. 65, p. 111-126, 1 maio 2014.

PERALTA, H.; COSTA, F., Competência e confiança dos professores no uso das TIC. Síntese de um estudo internacional (versão on-line). **Sísifo/Revista de Ciências da Educação**, n. 3, pp. 77-86, 2007.

ROCHA, F. S.; MARANGHELLO, G. F.; LUCCHESE, M. M. Acelerômetro eletrônico e a placa Arduino para ensino de física em tempo real. **Caderno Brasileiro de Ensino de Física**, Florianópolis, v. 31, n. 1, p. 98-123, 2013.

SANCHES, K. S.; RAMOS, A. O.; COSTA, F. J, As tecnologias digitais e a necessidade da formação continuada de professores de Ciências e Biologia para tecnologia: um estudo realizado em uma escola de Belo Horizonte, **Revista Tecnologias na Educação**, vol. 6, n. 11, Belo Horizonte, 2014.

SILVEIRA, S.; GIRARDI, M. Desenvolvimento de um kit experimental com Arduino para o ensino de Física Moderna no Ensino Médio. **Rev. Bras. Ensino Fís**., São Paulo, vol. 39, n. 4, 2017.

SOUZA, A. R. *et al*. A placa Arduino: uma opção de baixo custo para experiências de física assistidas pelo PC. **Rev. Bras. Ensino Fís**., São Paulo , vol. 33, n. 1, p. 01-05, 2011.

USGC. UNITED STATES COAST GUARD. **Light List: Atlantic Coast**. USGC Navigation Center, Alexandria, VA, 2019.

VASCONCELOS, C. A., OLIVEIRA, E. V., TIC no ensino e na formação de professores: reflexões a partir da prática docente, **Revista Brasileira de Ensino Superior**, vol. 3, n. 1, p. 112-132, 2017.

WIKIMEDIA COMMONS. **File:International_Morse_Code.svg**. 2019. Disponível em: https://commons.wikimedia.org/wiki/File:International_Morse_Code.svg. Acesso em: 20 de abril de 2020.

Recebido: 26 mar. 2019.
Aprovado: 11 mai. 2020.
DOI: 10.3895/rbect.v13n2.9904
Como citar: CAETANO, T. C.; FILHO, N. F.; MOREIRA, C. C. Construção de um transmissor e de um receptor de código Morse através de sinais luminosos com uma placa Arduino. **Revista Brasileira de Ensino de Ciência e Tecnologia**, Ponta Grossa, v.13, n. 2, p. 415-432, mai./ago. 2020. Disponível em: <https://periodicos.utfpr.edu.br/rbect/article/view/9904>. Acesso em: XXX.
Correspondência: Thiago Costa Caetano - tccaetano@unifei.edu.br