A UTILIZAÇÃO DA PLACA DE PROTOTIPAGEM ARDUINO COMO FERRAMENTA DE INCENTIVO E APRIMORAMENTO DA FORMAÇÃO DO ENGENHEIRO EM PROGRAMAÇÃO APLICADA

Área Temática: Formação do Engenheiro

Ewerton C. L. de Oliveira[1], Michelle P. Farias[2], Vitor O. G. Barbosa[3], Danielle L. Guedes[4], Orlando F. Silva[5]

[1] Universidade Federal do Pará – UFPA – Campus Belém – Belém-PA, ewerton.o43@gmail.com
[2] Universidade Federal do Pará – UFPA – Campus Belém – Belém-PA, michellefarias12@hotmail.com
[3] Universidade Federal do Pará – UFPA – Campus Belém – Belém-PA, vitorogb@gmail.com
[4] Universidade Federal do Pará – UFPA – Campus Belém – Belém-PA, danielle.lguedes@gmail.com
[5] Universidade Federal do Pará – UFPA – Campus Belém – Belém-PA, orfosi@ufpa.br

Resumo

Este artigo objetiva descrever as experiências desenvolvidas com os alunos do curso de Engenharia Elétrica da Universidade Federal do Pará por meio de minicurso sobre a utilização do microcontrolador Arduino como ferramenta no aprendizado dos graduandos em técnicas de linguagens de programação aplicadas a sistemas de controle. Assim como, visa explanar a metodologia utilizada para o ensino destes conteúdos por meio da descrição das aulas teóricas e experimentos práticos em laboratório utilizando o Arduino e diversos componentes eletrônicos para a construção dos circuitos. Também são abordados os resultados da avaliação dos alunos por meio de dados estatísticos retirados de questionários, os quais se referem aos impactos gerados pelo ensino deste microcontrolador no aprendizado de programação e na formação do engenheiro em graduação.

Palavras-chave: Programação; Arduino; Microcontrolador; Sistemas de controle.

1 Introdução

O ensino e o nível de conhecimento fornecido ao aluno de engenharia em sua vida acadêmica nas diversas universidades, públicas ou privadas, torna-se a cada ano um novo desafio a ser enfrentado perante as inúmeras dificuldades encontradas no aprendizado de disciplinas básicas e específicas. Entre essas disciplinas, encontram-se as linguagens de programação para computadores, as quais em geral abrangem o currículo de quase todo curso de engenharia, ao mesmo tempo em que se inclui como um assunto não tão bem compreendido, e muitas vezes, abstrato.

Segundo LAUDARES e RIBEIRO (2001) a mudança no conteúdo das atividades dos engenheiros e nas suas atribuições tem como consequência necessidades de qualificações mais específicas, como conhecimentos na área de informática, que são cada vez mais necessários. Em virtude disto, este artigo aborda os aspectos teóricos e metodológicos de um projeto de ensino desenvolvido por graduandos em engenharia elétrica na Universidade Federal do Pará envolvendo a aplicação da programação em placa de prototipagem Arduino como elemento motivador e pedagógico para a melhoria do aprendizado do engenheiro graduando nesta disciplina, assim como as suas possíveis aplicabilidades em sistemas de controle e automação.

A plataforma de prototipagem Arduino foi desenvolvida por estudantes na cidade italiana de Ivrea. O intuito do projeto era criar uma plataforma que pudesse realizar a programação e comunicação entre objetos interativos de forma acessível para estudantes e a um baixo custo. A placa Arduino consiste em um micro controlador ATMEL, portas de entrada e saída digitais, portas de entrada analógicas além de pinos capazes de fornecer níveis de tensão e aterrar pontos do circuito (MCROBERTS, 2011). A programação do microcontrolador é feita através do computador, utilizando um ambiente de desenvolvimento denominado *Arduino IDE*. A comunicação da placa é feita através de um cabo USB, utilizando uma porta serial do computador. A figura 1 ilustra o Arduino UNO, modelo utilizado durante as aulas, discriminando as portas presentes na placa.

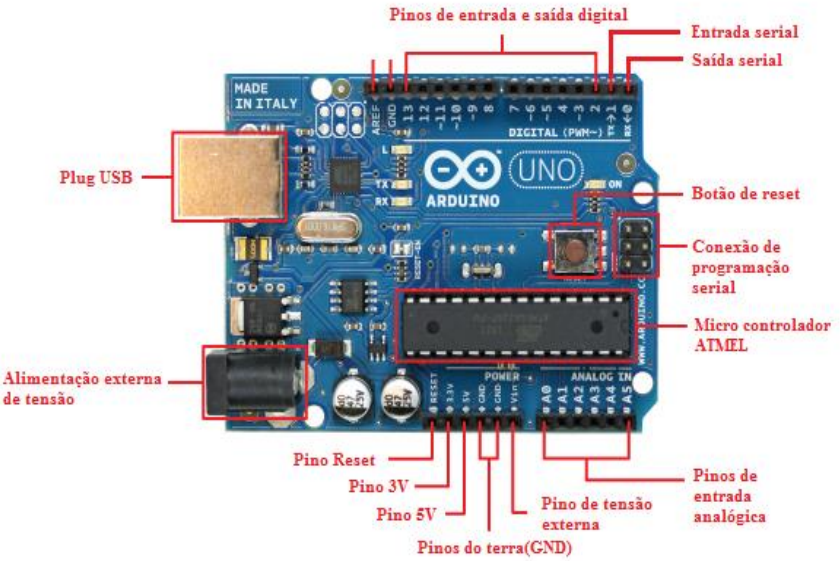

Figura 1- Arduino UNO. Fonte: Arquivo do Autor

Visto a dificuldade e, até mesmo, desistência dos discentes de engenharia elétrica nas disciplinas de programação devido a poucas aplicações interessantes no contexto do mesmo, surgiu a ideia de ministrar um minicurso capaz de aplicar os conhecimentos obtidos nas disciplinas de programação nas de sistemas de controle, visando assim uma conexão entre as entre elas.

2 Metodologia

O planejamento prévio do minicurso consistiu em elaborar cinco apostilas, apresentações em slides contendo os esquemáticos dos projetos e imagens dos componentes para facilitar o entendimento do aluno e em reunir nove kits Arduinos. A dinâmica de ensino consistiu em dividir os dezoito inscritos em duplas e entregar para cada uma, um único kit Arduino para manuseio. Cada aula foi ministrada por um bolsista do Programa de Educação Tutorial de Engenharia Elétrica (PETEE) e monitorada por mais três bolsistas presentes que respondiam dúvidas.

O minicurso ocorreu durante 5 dias com duração de uma hora e vinte minutos cada dia, numa sala de laboratório da Universidade Federal do Pará (UFPA), pois o minicurso exigia o uso de computadores. Durante essa semana, foram abordadas as funções básicas da linguagem de programação da placa, tal como a utilização de alguns componentes para a obtenção de dados. O foco das aulas foi demonstrar como a programação do microcontrolador, definindo as ações a serem realizadas a partir das entradas do sistema, é o princípio básico do controle e automação realizado em indústrias e residências.

2.1 Aula 1: Aula de programação da placa

2.1.1 Instalação e utilização do Arduino IDE

Na aula inicial, apresentaram-se os procedimentos básicos para realizar a gravação no microcontrolador da placa Arduino. Explicou-se sobre o ambiente de programação, o software Arduino IDE (Figura 1), que já havia sido instalado previamente em todos os computadores, e mencionou-se que qualquer usuário poderia fazer o download do software no site http://arduino.cc/. Com o software aberto, explicou-se sobre a sua janela principal, que é semelhante às plataformas de outras linguagens de programação, onde o código é escrito para posteriormente ser gravado no microcontrolador presente na placa Arduino.

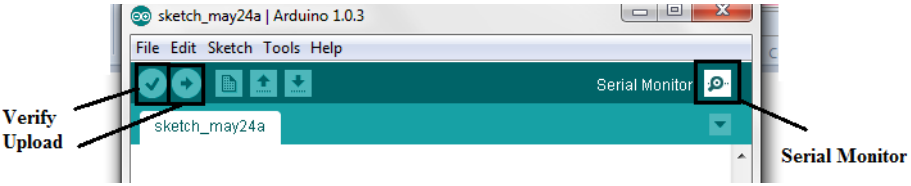

Figura 2 - Interface de Programação. Fonte: Arquivo do Autor

Além disso, os botões da janela principal foram discutidos e focou-se nos três principais o *Verify*, *Upload* e *Serial Monitor*. O *Verify* verifica o código escrito a procura de erros de sintática ou semântica no programa e identifica erros e discrimina para o usuário na aba inferior da janela principal. Depois de verificado, o programa pode ser gravado no controlador através do botão *Upload*, porém o usuário deve se certificar

que a placa está conectada ao computador através da entrada USB e que o dispositivo está sendo reconhecido pelo sistema.

É importante ressaltar que, após a gravação, o programa fica salvo no controlador mesmo com a desconexão do cabo USB. O botão Serial Monitor disponibiliza uma interface onde é possível visualizar variáveis pertinentes do código de programação e também quaisquer dados transmitidos entre o computador e a placa através da porta de comunicação serial.

2.1.2 Linguagem de programação

A linguagem de programação utilizada para a gravação de programas é baseada na linguagem de programação C++. Logo, muitas características e sintaxes da linguagem que iremos utilizar são análogas as da linguagem C++. Porém, existem funções criadas especialmente para a programação em Arduino. Muitas destas funções fazem referência as portas que aplaca possui e também permitem utilizar a comunicação serial para transferência de dados entre o microcontrolador e o computador. As principais funções utilizadas na aula estão presentes na Tabela 1.

Tabela 1 – Principais funções utilizadas para programar

Função	**Funcionalidade**
pinMode(porta,tipo da porta)	Função que declara o número da porta digital que será utilizada pela placa e se a porta deve operar como entrada (INPUT) ou saída (OUTPUT) de dados. Toda a porta digital que for utilizada deve ser declarada. Portas analógicas não precisam ser declaradas.
digitalWrite(porta, nível lógico)	Envia um sinal digital para uma porta de saída. Este sinal possui apenas dois valores possíveis: HIGH (1) ou LOW (0).
digitalRead(porta)	Identifica o valor que está sendo mandado para uma porta digital de entrada. Este valor precisa ser salvo em uma variável para ser visualizado.
analogRead(porta)	Lê um valor de tensão que está sendo

	aplicado em uma porta analógica de entrada. A porta analógica representa os valores lidos que podem variar entre 0 e 1023. Este valor precisa ser salvo em uma variável para ser visualizado.
Setup ()	Função sem parâmetros, na qual deve ser declarada todos os pinos que serão utilizados.
Loop ()	Funções em parâmetros, a qual deve ser escrito todas as linhas de código que descrevem as ações a serem executadas pelo microcontrolador. Estas linhas de código serão executadas sem loop, semelhante a uma função while de outras linguagens de programação.

Fonte: Arquivo do Autor

2.2 Aula 2: Componentes eletrônicos e acionamento de LED's

2.2.1 Componentes eletrônicos

O primeiro foco desta aula foi abordar os componentes eletrônicos básicos utilizados no aprendizado do microcontrolador Arduino. Entre esses componentes podem ser citados: Resistores, LED's, Buzzer e Termistor.

Os resistores são elementos bastante encontrados em circuitos eletrônicos, pois sua principal característica é promover resistência à passagem de corrente elétrica, além de proporcionar quedas de tensões (HALLIDAY; RESNICK; WALKER, 2009). O LED, como pode ser visto na Figura 3, (***Light-Emitting-Diode*****) é um componente que permite a passagem da corrente elétrica por apenas um sentido, impedindo a passagem da mesma no sentido contrário (Diodo). Este componente também emite luz quando uma corrente o atravessa.**

O Buzzer é um componente eletrônico constituída de duas placas de metal e um cristal piezoelétrico. Quando o Buzzer recebe um sinal de tensão em uma determinada frequência, o mesmo começa a vibrar funcionando como uma sirene.

O termistor, Figura 4, é um sensor de temperatura fabricado com materiais semicondutores que possuem sua resistência alterada como efeito direto da temperatura. Entre esses componentes, os mais utilizados estão o

LM35, NTC (*Negative TemperatureCoefficient*) e PTC (*Positive TemperatureCoefficient*).

Figura 3 - Led Vermelho. Fonte: Arquivo do Autor

Figura 4 - Termistor LM35. Fonte: Arquivo do Autor

2.2.2 Acionamento intervalar de LED's

Neste experimento foi utilizada uma placa Arduino, três LED's das cores vermelho, verde e amarela, e três resistências de 300 Ohm. O principal objetivo deste experimento consistiu em ensinar aos alunos como controlar o acionamento dos LED's supracitados em intervalos diferentes. O sistema resumia-se em acionar primeiramente o LED verde, após certo intervalo de tempo o primeiro se apagava e o amarelo era acionado, e somente após outro intervalo de tempo o segundo se apagava e o vermelho era aceso, de forma que o ciclo se repetisse continuamente.

Para este experimento foi utilizada, além das funções básicas de declarar e escrever em uma porta digital com *pinMode()* e *digitalWrite()* respectivamente, a instrução *delay()* que recebe um parâmetro do intervalo de atraso em milissegundos dentro da função *loop()* principal na programação do Arduino, conforme é ilustrado abaixo uma parcela do programa utilizado. A Figura 3 abaixo ilustra parte do código utilizado para a realização deste experimento.

```
void loop(){

digitalWrite(LED1,HIGH);
digitalWrite(LED2,LOW);  //acionamento do LED 1
digitalWrite(buz,LOW);

delay(2000);       //delay de 2 segundos

digitalWrite(LED1,LOW);
digitalWrite(LED2,HIGH);  //acionamento do LED 2
digitalWrite(buz,LOW);
```

Figura 5 - Código do programa controlador de LED's. Fonte: Arquivo do Autor

Percebe-se no trecho acima que o LED1 (verde) é acionado enquanto os outros não, e após 2 segundos o mesmo se apaga e o LED2 (amarelo) é acionado. Abaixo é ilustrado na Figura 6 o esquema eletrônico montado na protoboard para a realização deste experimento na aula 2.

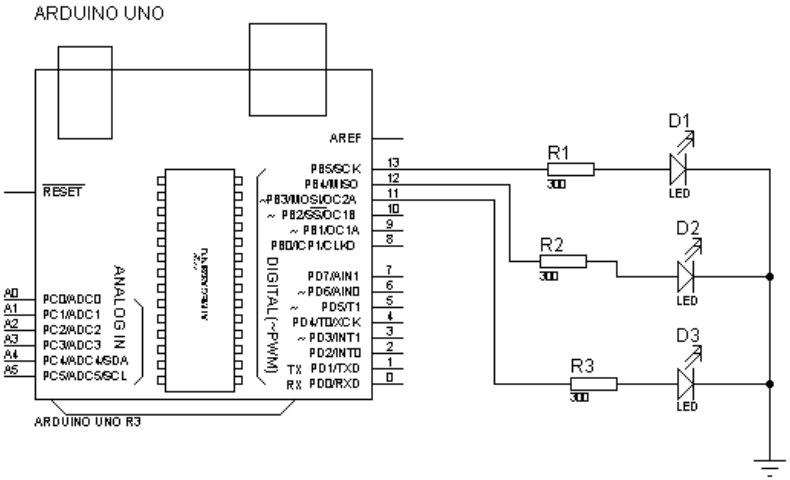

Figura 6 - Circuito eletrônico para o acionamento dos LED's. Fonte: Arquivo do Autor

2.3 Aula 3: Utilização do display LCD

Na terceira aula, abordou-se sobre Liquid Crystal Display (LCD), um painel fino de extrema utilidade para visualizar informações, como textos. Durante a aula, explicou-se sobre a existência de vários tipos de LCDs, porém focou-se no LCD de 16 pinos, o qual seria utilizado. Primeiramente, explicou-se a funcionalidade de cada pino (Tabela 2) e ressaltou-se a importância de conectá-los de forma correta para não danificar o componente. Após isso, pediu-se aos alunos para montar o esquemático da Figura 7.

Tabela 2 – Função de cada pino do LCD

Pinos	Símbolos	Função	Pinos	Símbolos	Função
1	Vss	GND (alimentação)	9	DB2	Dado
2	Vdd	5V (alimentação)	10	DB3	Dado
3	V0	Ajuste de Contraste	11	DB4	Dado
4	Rs	Habilita/Desabilita Seletor de Registrador	12	DB5	Dado
5	R/W	Leitura/Escrita	13	DB6	Dado
6	E	Habilita Escrita no LCD	14	DB7	Dado
7	DB0	Dado	15	A	5V (Backlight)
8	DB1	Dado	16	K	GND (Backlight)

Fonte: Arquivo do Autor

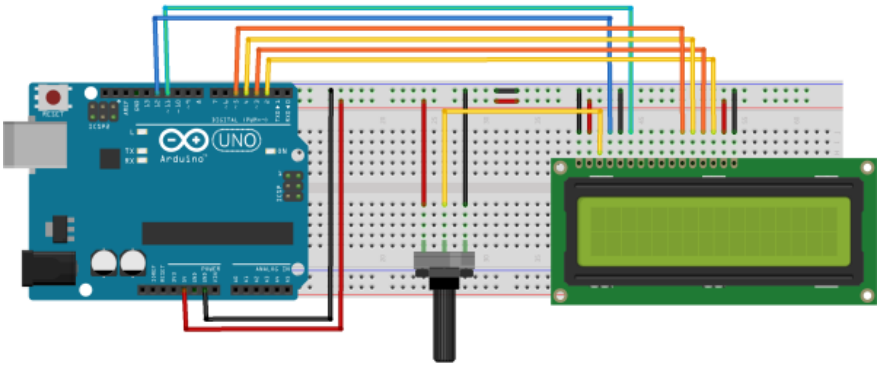

Figura 7 - Esquemático proposto na aula. Fonte: Arquivo do Autor

2.3.1 Programação para o display

Para utilização do componente, primeiramente explicou-se sobre as bibliotecas e como incluí-las, depois se declarou a biblioteca do LCD que contém as principais funções para utilização do componente. Após isso, apresentaram-se as funções fundamentais para o funcionamento, tais como a função que declara os pinos do Arduino conectados ao LCD e a função que especifica o tamanho da matriz utilizada, nesse caso especificou-se uma matriz de 16 colunas por 2 linhas. Terminadas as especificações para manuseio, foram discutidas as funções de uso geral, apresentadas na Tabela 3 e, por fim, foi proposto o código da Figura 8.

Tabela 3 – Principais funções explanadas

Função	Funcionalidade
.begin(colunas, linhas)	Especificaas dimensões do display. Esse comando precisa ser escrito antes de qualquer outro comando da biblioteca do LCD.
.clear()	Posiciona o cursor no canto superior esquerdo e limpa a tela
.setCursor(coluna, linha)	Posiciona o cursor numa coluna x e linha y
.write() / .print()	Escreve um caractere / texto no LCD
.cursor() / .noCursor()	Mostra / Esconde o cursor do LCD
.display() / .noDisplay()	Liga o display do LCD
.scrollDisplayLeft() / .scrollDisplayRight()	Desloca o que contem no display um espaço para a esquerda / direita
.autoscroll()/ .noAutoscroll()	Liga / Desliga o deslocamento automático do LCD
.leftToRight() / . rightToLeft()	Significa que a escrita no display será da esquerda para direita / direita para esquerda
.createChar(numero,dados)	Cria um caractere personalizado. Um caractere possui uma matriz 5x8 pixels, numerados de 0 à 7

Fonte: Arquivo do autor.

```
#include <LiquidCrystal.h> // inclusão da biblioteca do LCD
LiquidCrystal lcd(12,11,5,4,3,2); // Pinos do Arduino conectados ao LCD
void setup (){
lcd.begin(16,2); // Declaração do tipo de Matriz (16 colunas x 2 linhas)
lcd.setCursor(0,0); // Posiciona o Cursor na coluna 0 e linha 0
lcd.print("Aula de LCD"); // Printa no Lcd a frase "Aula de LCD"
lcd.setCursor(0,1);
lcd.print("PET EE");}
void loop(){}
```

Figura 8 - Código Proposto. Fonte: Arquivo do Autor

2.4 Aula 4: Controle de temperatura e luminosidade

Esta aula foi direcionada ao desenvolvimento de um sistema de controle mais abrangente, uma vez que o mesmo demandou a utilização de todos os outros componentes ensinados nas aulas anteriores. O

funcionamento deste sistema procede da seguinte maneira, primeiro display LCD exibirá constantemente os valores de temperatura e luminosidade aquisitados nas entradas analógicas, caso a temperatura seja menor que a faixa 29 a 32 ºC, o LED vermelho é acionado, se a temperatura for maior que a faixa 29 a 32 ºC, o LED vermelho e o Buzzer serão acionados, caos a temperatura estiver na faixa 29 a 32 ºC, o LED verde será acionado, se a luminosidade estiver abaixo de 400, o LED amarelo é acionado e o branco não, caso contrário, o LED branco é acionado e o amarelo desativado.

Neste experimento procurou-se incentivar a habilidade dos alunos em trabalhar tanto com a comunicação serial pelo microcontrolador Arduino, uma vez que são utilizados dois tipo de sensores para aquisitar sinais de duas variáveis ambientais distintas por meio da leitura nas portas analógicas com o comando *analogRead()*, além de que são exploradas algumas estruturas de condições na programação por meio da instrução *if()*, cujo parâmetro envolvem tato as faixas limites de temperatura e luminosidade.

2.5 Aula 5: Acionamento de motor DC

O objetivo principal desta aula foi realizar o acionamento e controle da velocidade um motor DC utilizando uma porta digital da placa e o princípio de modulação por tamanho de pulso (PWM). O circuito montado está ilustrado na Figura 9.

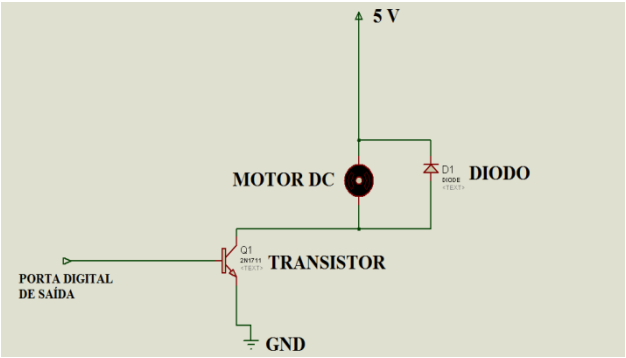

Figura 9 - Circuito proposto. Fonte: Arquivo do Autor

2.5.1 Modulação por tamanho de pulso (PWM)

A técnica de modulação por tamanho de pulso permite obter vários níveis de operação de um dado componente, mesmo dispondo apenas de um gerador de sinais digital, que só pode fornecer dois níveis de tensão. Isso ocorre com a alteração do tempo em que o sinal se mantém em nível alto, também chamado de *duty cycle.* O valor do duty cycle é representando em porcentagem.

Um gerador de tensão DC de 5 volts submetido á técnica PWM com 80% de duty cycle mantém o seu sinal no nível alto (5 volts) durante 80% de um período de tempo pequeno pré-estabelecido. A carga conectada a essa fonte não perceberia os efeitos da mudança constante de tensão. A queda de tensão em seus terminais seria simplesmente 4 volts, que é exatamente 80% de 5V, a tensão máxima do sinal.

As portas digitais do Arduino que permitem o uso do PWM são aquelas identificadas com o símbolo "~" ao lado do número. A função para utilizá-las é analogwrite (). Esta função possui dois parâmetros. O primeiro é o número da porta sendo utilizada e o segundo é o ciclo de trabalho desejado. Assim como as portas analógicas possuem 10 bits para representação de valores, as portas PWM possuem oito bits, possibilitando-as a representação de valores entre 0 e 255(2^8-1). O parâmetro duty cycle deve ser inserido não em porcentagem, mas com valores inteiros na faixa entre 0 e 255, sendo 255 correspondente a um duty cycle de 100%. A Figura 10 mostra o código proposta na aula para a variação da velocidade de um motor a cada cinco segundos.

```
void setup()
{}

void loop()
{
analogWrite(3,255);
delay(5000);
analogWrite(3,127);
delay(5000);
analogWrite(3,63);
delay(5000);
}
```

Figura 10 - Código proposto na aula. Fonte: Arquivo do Autor

3 Resultados e Considerações Finais

Ao final da quinta e última aula, aplicou-se um questionário a fim de saber se os objetivos propostos haviam sido alcançados. O questionário continha oito perguntas de sim ou não, uma pergunta avaliando o minicurso por conceitos com as opções Ruim, Regular, Bom e Excelente e, por fim, um campo para o discente sugerir, criticar e/ou elogiar as aulas aplicadas.

Primeiramente, perguntou-se ao aluno se o mesmo possuía afinidade com as disciplinas propostas no minicurso, 87,5% afirmou possuir afinidade nas disciplinas de programação e sistemas de controle. Também através do questionário, conclui-se que 56,25% ainda não haviam tido contado com o microcontrolador Arduino, porém 93,5% afirmou que houve pouca ou quase nenhuma dificuldade em operar o IDE Arduino, pois a consideraram uma plataforma bem intuitiva, com layout simples e bem similar a outros ambientes de programação.

Sobre a metodologia, 100% dos alunos afirmaram que a apresentação do minicurso havia sido bem empregada, conseguindo atingir os objetivos propostos, e 93.65% afirmou que a experiência ajudou a reforçar os conceitos teóricos das disciplinas de programação e sistemas de controle.

Logo, os alunos avaliaram o minicurso através de quatro conceitos e se considerava uma boa ideia a utilização do Arduino em aulas de programação, o resultado pode ser visto nas Figuras11 e 12.

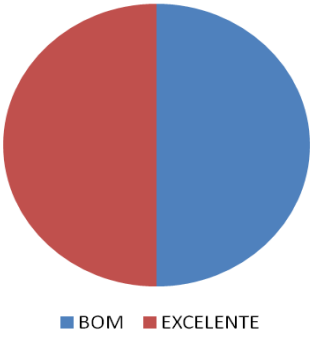

Figura 11 - Avaliação do minicurso feita pelos alunos. Fonte: Arquivo do Autor

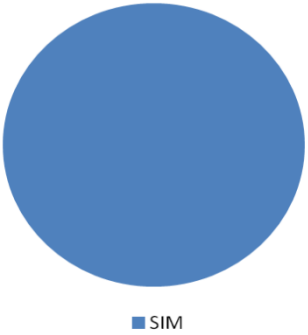

Figura 12 - Aceitação dos alunos para o uso do Arduino em disciplinas de programação. Fonte: Arquivo do Autor

Analisando-se as sugestões feitas pelos alunos, deparou-se com algumas sugestões que poderão ser aplicas para as próximas turmas, tais como: alteração da duração do minicurso que passaria para duas semanas, aumento do número de monitores para sanar as dúvidas das duplas e tentaremos ainda conseguir um número maior de kits Arduino, pois, como a procura pelo foi tão grande, fez se necessária a criação de uma lista de espera para o próximo minicurso.

Dessa forma, através do questionário, pode-se concluir que o minicurso conseguiu aplicar conceitos das disciplinas de programação nas de sistemas de controle e ajudou fortalecer a correlação entre elas, fato que tornou o minicurso dinâmico e incentivou os alunos a serem participativos. Logo, pode-se concluir que o minicurso atingiu os objetivos propostos e proporcionou aos acadêmicos um fortalecimento dos conhecimentos teóricos prévios e os aplicou-os a outra disciplina muito importante no currículo deles.

4 Referências Bibliográficas

ARDUINO OFFICIAL WEBSITE. **LanguageReference.** Disponível em: <http://arduino.cc/en/Reference/HomePage> Acesso em: 20 fev. 2014.

HALLIDAY, D.; RESNICK; WALKER, J. **Fundamentos de Física, Volume 3: Eletromagnetismo.** Rio de Janeiro: LTC, 2009.

LAUDARES, J. B.; RIBEIRO, S. **Trabalho e Formação do Engenheiro. Anais do XXIX COBENGE**, Porto Alegre – RS, 2001.

MCROBERTS, Michael. **Arduino Básico**; [tradução Rafael Zanolli]. – São Paulo: Novatec Editora, 2011.