DESENVOLVIMENTO DE UMA PLATAFORMA DE APRENDIZAGEM DE SISTEMAS EMBARCADOS BASEADA EM ESP32

Otávio Evaristo Cardoso – otavioevaristocardoso@hotmail.com
Otávio Cosme Matias – otaviocmatias@gmail.com
Lucas Frederico Jardim Meloni – lucas.meloni@ifmg.edu.br
Rodrigo Menezes Sobral Zacaroni - rodrigo.zacaroni@ifmg.edu.br
Gustavo Lobato Campos - gustavo.lobato@ifmg.edu.br
Instituto Federal de Minas Gerais, IFMG – Campus Formiga
Rua Padre Alberico, 440 – Bairro São Luiz
35570-000 – Formiga – MG

Resumo: *O uso de plataformas como o Arduino em cursos tecnológicos mostrou-se eficaz e motivador, ao facilitar a compreensão prática de vários tópicos de eletrônica, automação, sistemas de informação e robótica. Apesar de ser excelente em aplicações simples, as placas básicas de entrada utilizam microcontroladores relativamente antigos, contendo recursos periféricos simples e a capacidade de processamento limitada. Também se torna difícil o desenvolver de aplicações modernas, ou mais elaboradas nessas placas, por exemplo, dispositivos conectados à Internet (Internet of Things, IoT) e comunicação Bluetooth®, pois são necessários acessórios extras (Shields) que podem encarecer o projeto. Neste contexto, este artigo mostrará as etapas de planejamento e produção de materiais didáticos para o desenvolvimento de uma plataforma didática para ensino de microcontroladores, baseada nos módulos ESP32. A motivação para criação desta plataforma é possibilitar que os alunos estendam seus conhecimentos sobre sistemas embarcados, aperfeiçoem projetos desenvolvidos anteriormente em Arduino e sejam capazes de criar novos projetos mais complexos, que exijam maior capacidade de processamento e recursos. Serão comentadas as metodologias para desenvolvimento dos materiais didáticos, a produção de materiais didáticos no formato de roteiros de laboratório, a proposta para criação de um portal para acesso a esses materiais e também as dificuldades encontradas até o momento pelos autores.*

Palavras-chave: *Sistemas Embarcados. Metodologias de Ensino. Aprendizagem Baseada em Problemas.*

1 INTRODUÇÃO

Recursos tecnológicos utilizados em aulas de laboratório são uma alternativa pedagógica capaz de produzir resultados extremamente satisfatórios, pois possibilitam demonstrar a aplicação direta de conceitos básicos vistos em sala de aula, atrai a atenção dos alunos e motiva a implementação de novas tecnologias (CAMPOS, 2019). Isso evita a proliferação de problemas comumente relatados em cursos de graduação como, por exemplo, evasão,

desinteresse e descrença dos discentes sobre o conteúdo aprendido (BARBOSA, MEZZOMO e LODER, 2011).

Neste âmbito, plataforma de desenvolvimento como a Arduino vem sendo usadas em diversos cursos de graduação, como física (MOREIRA, 2018) ou Engenharia Elétrica mostrado em (VILAR *et al*, 2018), (MATOS, 2018) e (CAMPOS, 2019). Trata-se de uma plataforma gratuita (*Opensouce*), de baixo custo, que utiliza placas de desenvolvimento baseadas em microcontroladores, com uma interface de desenvolvimento (IDE) amigável e de fácil programação (GEDDES, 2017). No ensino em engenharia elétrica, essa plataforma revelou-se versátil e dinâmica, permitindo que os estudantes possam compreender melhor as disciplinas do curso e potencializar seu desempenho nas mesmas (BRIDI *et al*, 2013).

Entretanto, as placas Arduino mais acessíveis, tais como os modelos UNO e MEGA, utilizam microcontroladores da família AVR com capacidade de processamento limitada e poucos recursos periféricos integrados. Isso obriga os usuários a utilizarem placas adaptadoras (*Shields*), circuitos integrados externos, ou recorrer a modelos de placas mais caros, elevando o custo inicial para desenvolvimento de alguns projetos como, por exemplo, dispositivos IoT que necessitem de conexão com a Internet. A montagem de protótipos também torna-se mais complicada e demorada, por exigir a integração de diversos componentes.

Em 2016 surgiram os módulos ESP-32, que comparados aos modelos básicos de placas Arduino, apresentam mais recursos disponíveis como rádio para comunicação IEEE 802.11 b/g/n e *Bluetooth* 4.2 BLE, maior capacidade de memórias e núcleo de processamento *Dual Core* (ESPRESSIF, 2016). Os módulos ESP-32 também são programáveis no ambiente de desenvolvimento Arduino IDE, através das linguagens de programação C/C++, sem a necessidade de circuitos de programação externos. Seu custo também não é elevado, quando comparado a outros modelos de placas com recursos para comunicação sem fio já integradas.

Desde então, várias aplicações interessantes foram desenvolvidas nesta plataforma, como transmissores sem fio inteligentes, para aplicações em sistemas de supervisão, controle e aquisição de dados (AGHENTA e IQBAL, 2019), sistemas de automação residencial (THIRUPATHI e SAGAR, 2018) ou medidores de energia (ALEXANDRE, 2019).

Este artigo discutirá o planejamento e desenvolvimento de materiais didáticos, no formato de roteiros para experiências de laboratório, que futuramente serão incorporados em uma plataforma didática, desenvolvida para o ensino de sistemas embarcados através dos módulos ESP-32. A plataforma consiste em um portal de informações, na forma de um aplicativo ou *site*, cujo objetivo é auxiliar os alunos durante aulas laboratoriais, promovendo fácil acesso a experimentos, montagens, informações técnicas e de programação dos microcontroladores contidos nos módulos ESP. Além disso, também espera-se que estes materiais e o portal consigam aprimorar o conhecimento de alunos que já tiveram contato com o desenvolvimento de sistemas embarcados simples, utilizando as linguagens de programação C/C++ através da plataforma Arduino. Também serão discutidas a organização elaborada para os roteiros, qual a aprendizagem esperada dos alunos e quais dificuldades foram encontradas no desenvolvimento destes materiais didáticos. Tanto os roteiros, quanto a plataforma serão futuramente utilizados na criação de disciplinas laboratoriais para o curso de Engenharia Elétrica.

Na seção 2 apresentar-se-á metodologia que foi estabelecida para selecionar os temas dos roteiros, através de uma comparação entre as placas Arduino UNO e ESP32 Devkit C. Na seção 3 será discuto o planejamento dos roteiros, bem como o fluxo de aprendizagem esperado. Por fim, na seção 4, serão discutidos os resultados preliminares e as conclusões. Também serão

dadas as perspectivas para o curso proposto e relatadas algumas dificuldades na elaboração dos experimentos.

2 METODOLOGIA

Assim como feito em (OLIVEIRA, 2014), antes de iniciar o planejamento do curso de ESP-32, foi necessário pesquisar e conhecer a plataforma. Segundo o manual (ESPRESSIF, 2020) define-se o ESP32 como um circuito integrado dedicado ao desenvolvimento de aplicações portáteis e dispositivos IoT conectados à Internet (*Internet of Things*). O fabricante destes circuitos também disponibiliza duas famílias de módulos programáveis denominados ESP32 e ESP32-S2, ambos também separados em subgrupos classificados como Wroom, mostrado na "Figura 1 (a)" e Wrover, mostrado na "Figura 1 (b)".

Figura 1 – Módulos ESP32. (a) Módulo ESP32 – Wroom 32. (b) Módulo ESP32-S2 – Wrover 32.

(a) **(b)**

Fonte: retirado e adaptado de (ESPRESSIF, 2020).

Antes de iniciar o desenvolvimento dos roteiros, foi estabelecido como pré-requisito que os alunos já tivessem contato prévio com a plataforma Arduino. Também foi assumido que este contato foi estabelecido através do modelo de placa Arduino UNO Rev 3, sendo o mais comumente encontrado. Assim, foi estabelecida uma comparação entre as placas Arduino UNO e a ESP32-DevKitC, que resultou na "Tabela 1", preenchida com auxílio de informações disponibilizadas pelos seus fabricantes em (ARDUINO, 2020) e (ESPRESSIF, 2020).

Nota-se pela "Tabela 1" que a placa ESP32 possui um processador *dualcore*, com capacidade superior em frequência de operação, e processamento de instruções executadas por segundo. A placa também conta com maior capacidade de memória para desenvolvimento de programas e manipulação de dados. Há também um modo de baixo consumo de energia, o qual pode estender a autonomia de sistemas que utilizem baterias (ESPRESSIF, 2020).

Quanto a capacidade de componentes periféricos comuns, o ESP32 também supera o Arduino UNO Rev 3, disponibilizando maior número de portas digitais (Entradas/Saídas) e entradas analógicas, temporizadores e saídas PWM, assim como entradas configuráveis através de interrupção externa. Também se nota recursos diferentes como a presença de um sensor de efeito Hall, sensor de efeito capacitivo e uma saída analógica, baseada em um conversor Digital para Analógico.

O suporte a padrões de comunicação também é um destaque para o ESP, que além de suportar os mesmos protocolos I2C, SPI e UART serial, também possui I2S, CAN, Ethernet, IEEE 802.11 b/g/n/e/i e Bluetooth v4.2 BR/EDR e BLE. Diante de tantos recursos, foi possível elaborar o diagrama da "Figura 2", que ilustra uma possível organização dos roteiros de

laboratório, baseada na comparação entre os recursos periféricos do ESP32 DevKitC e Arduino UNO.

Tabela 1 – Aspectos comparativos entre as placas Arduino UNO e ESP32 DevKit C.

	Arduino UNO 	**ESP32 DevKit C** 
Tensão de Operação	5 V	3,3 V
Operação em modo de baixa energia	Não possui	Modo de hibernação (*Deep Sleep*)
Arquitetura do processador	AVR® (8-bit)	Xtensa® LX6 (32-bit) Dual Core
Frequência de operação	16 MHz	80 – 240 MHz
Instruções por segundo	16 milhões	Até 600 milhões
Memória Flash (programa)	32 kB	4 MB
Memória RAM/SRAM	2 kB	530 kB
Memória EEPROM	4 kB	448 kB
Pinos de Entrada/Saída	23	34
Pinos com suporte a Interrupção Externa	2	34
Saídas PWM	6 saídas	16 saídas
Entradas Analógicas	6 Unidades de 10-bit	18 Unidades de 12-bit
Saídas Analógicas	Não possui	2 Unidades de 8-bit
WiFi integrado	Não possui	IEEE 802.11 b/g/n/e/i
Bluetooth integrado	Não possui	Bluetooth v4.2 BR / EDR e BLE
Temporizadores	3 Timers	4 Timers (64-bit)
Relógio de Tempo Real dedicado	Não possui	RTC com suporte a UNIX Timestamp
Sensores Internos	Temperatura	Temperatura, Efeito Hall e Toque Capacitivo
Protocolos de comunicação	I2C, SPI, UART	SPI, UART, I2S, I2C, CAN, Leitor de cartão SD/MMC, Ethernet, Leitor infravermelho de controle remoto.

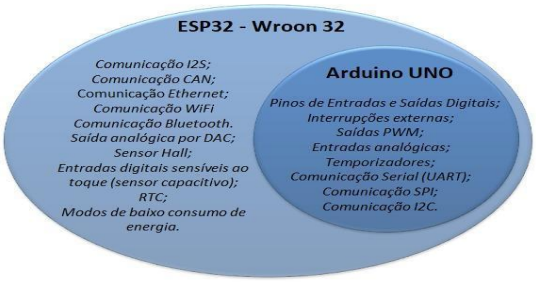
Fonte: Elaborado pelos autores, com auxílio informações contidas em (ARDUINO, 2020) e (ESPRESSIF, 2020).

Na "Figura 2" foram formadas duas categorias denominadas recursos comuns (RC) e recursos exclusivos (RE). A categoria RC é formada por periféricos comuns ao ESP-32 DevkitC e Arduino UNO, como Entradas/Saídas digitais, Interrupções Externas, Saídas PWM, entradas analógicas (ADC) e Temporizadores e os protocolos de comunicação UART, SPI e I2C. A categoria RE é formada por recursos presentes apenas no ESP32, como saída analógica DAC,

o sensor Hall, RTC e as entradas digitais sensíveis ao toque e os demais protocolos de comunicação suportados.

Através da organização estabelecida na "Figura 2" foi possível estabelecer critérios para elaborar a ordem de sugestão na execução dos roteiros de laboratório. Como foi assumido que os alunos já tivessem contato com programação em C/C++ através das placas Arduino UNO, optou-se por elaborar inicialmente roteiros que abordem recursos básicos, comuns entre as duas placas.

Figura 2 - Organização dos recursos a serem abordados nos roteiros de laboratório.

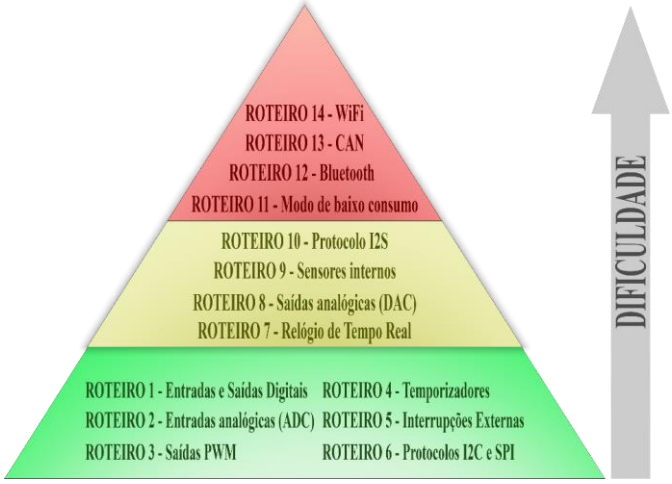


Fonte: Elaborado pelos autores.

Posteriormente, foi feita uma pesquisa sobre aplicações básicas e exemplos disponibilizados pelo fabricante do ESP32 e desenvolvedores terceiros. Foi verificada uma grande diferença de complexidade entre estes periféricos, avaliada através do tamanho dos programas, circuitos necessários e informações sobre o funcionamento dos *Datasheets.* Assim, também foi possível classificar os recursos exclusivos em níveis Intermediários e Avançados.

O resultado final da classificação dos roteiros se encontra na "Figura 3". Pela quantidade proposta de roteiros, considerando um curso com carga horária de duas horas semanais, verifica-se que este curso deverá ser ministrado em um semestre ou pode ser dividido em módulos.

Figura 3 – Organização proposta para os roteiros de laboratório.



Fonte: Elaborado pelos autores.

3 PLANEJAMENTO DOS ROTEIROS

Para elaborar cada roteiro, foram estruturadas quatro partes. Para cada uma dessas quatro divisões foi proposto um questionário, ilustrado na "Figura 4", o qual deverá ser respondido antes da escolha de cada experiência.

Na primeira parte, recorre-se à interdisciplinaridade para elaborar uma fundamentação teórica do assunto proposto no roteiro. Neste momento, ocorre a explicação do funcionamento de sistemas periféricos abordados, quais as principais aplicações práticas e os princípios físicos de operação de cada recurso estudado.

Na segunda parte são discutidos quais experimentos serão elaborados para exemplificar o funcionamento do recurso estudado, quais serão os componentes necessários para este experimento, como elaborar os diagramas elétricos da montagem e qual o tempo necessário para concluir esta montagem. Para elaboração dos diagramas elétricos e de montagem, recorreu-se ao uso do *software Fritzing*®.

A terceira parte trata o funcionamento do *Firmware* e da programação. Nesta fase, são mostrados como utilizar bibliotecas prontas, realizar a configuração básica do recurso, quais as funções utilizadas e qual o princípio lógico do algoritmo desenvolvido. Para compreender o funcionamento de alguns algoritmos, foi preciso recorrer ao uso de fluxogramas. Todos os códigos desenvolvidos foram comentados passo-a-passo com a explicação das funcionalidades.

Por fim, na quarta parte, são feitas discussões sobre a prática elaborada, além da aplicação de um questionário avaliativo, onde os alunos darão suas opiniões, discutirão as dificuldades encontradas e irão propor sugestões de melhoria.

Figura 4 – Estruturação básica de cada roteiro.

PARTE 1 – INTRODUÇÃO E FUNDAMENTAÇÃO TEÓRICA
● Qual a utilidade deste recurso?
● Como é feita a configuração deste recurso no ESP32?
● Quais exemplos de aplicações práticas envolveriam o uso deste recurso?
PARTE 2 – HARDWARE NECESSÁRIO
● Qual experimento será proposto para demonstrar o funcionamento do recurso abordado?
● Quantos componentes são necessários para demonstrar o funcionamento deste recurso?
● Como é realizada a conexão destes componentes com o ESP32?
● Qual será e como poderá ser interpretado o diagrama elétrico do experimento?
● Quanto tempo será gasto para o aluno realizar a montagem do experimento?
PARTE 3 – SOFTWARE E CONFIGURAÇÕES
● Existem bibliotecas prontas para este recurso?
● Como explicar o algoritmo do código? Será utilizado um fluxograma?
● Como carregar e inicializar as bibliotecas?
● Quais funções e variáveis serão declaradas para esse experimento?
PARTE 4 – CONCLUSÕES E PERCEPÇÕES DOS ALUNOS
● O que foi possível concluir após a realização deste experimento?
● É possível utilizar as informações apresentadas neste roteiro para desenvolver outras aplicações?
● O recurso estudado operou da forma esperada?
● O tempo proposto para realizar a montagem e compreender o código foi suficiente?

Fonte: elaborado pelos autores.

4 RESULTADOS PRELIMINARES E DISCUSSÕES

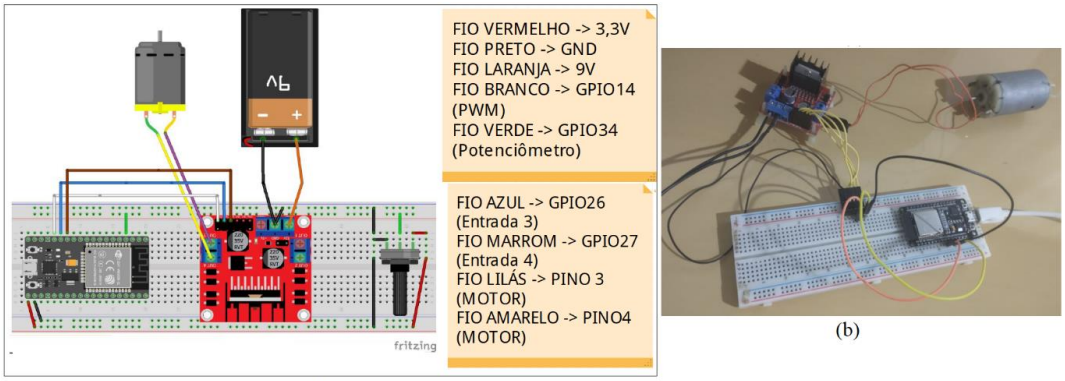
O projeto que resulta o presente artigo teve início em março de 2020, segue em desenvolvimento até esta submissão e tem encerramento previsto para dezembro de 2020. Foram desenvolvidos oito roteiros de laboratório, abordando a instalação dos *softwares* e da placa, configuração de entradas e saídas digitais, configuração das entradas analógicas através do conversor ADC de aproximações sucessivas, uso das saídas PWM para controle de luminosidade de LEDs e de velocidade de motores, configuração e utilização do sensor embarcado de efeito Hall e saída analógica DAC. Ainda estão em desenvolvimento roteiros que abordam a comunicação Bluetooth e WiFi, o relógio de tempo real e uso de interrupções externas. Os demais temas, presentes na "Figura 3", serão abordados futuramente.

Todos as montagens e códigos propostos foram testados em bancada. O tempo médio esperado para que os alunos concluam a montagem e teste dos códigos depende da dificuldade de cada tema. O menor tempo esperado é de 40 minutos, enquanto que o tempo máximo foi de 90 minutos para sua realização.

Como exemplo, um dos roteiros propostos aborda a saída PWM para controle de velocidade de um motor de corrente contínua, de ímã permanente. O objetivo foi discutir as diferenças entre as funções *analogWrite()*, utilizada no Arduino UNO, e *ledcWrite()* para o ESP32. O diagrama de montagem e o teste da montagem em bancada são mostrados na "Figura 5" e a "Figura 6" apresenta o código comentado.

Quanto as percepções dos autores durante a elaboração dos roteiros, relata-se grande dificuldade encontrada em utilizar documentos técnicos e manuais do ESP32, bem como compreender os códigos exemplo fornecidos pelo fabricante ESPRESSIF. Em muitas ocasiões, como ocorreu durante a elaboração do roteiro sobre entradas analógicas e ADC, percebeu-se que a documentação estava incompleta, pois não demonstrava qual a estrutura deste recurso dentro do microcontrolador. Segundo (LIBÂNEO, 1999) o professor deve dominar seus métodos de ensino com segurança, para que o aluno tenha tranquilidade e saber a quem recorrer quando tiver dúvidas. Portanto, os autores acreditam que a carência de algumas informações nestes documentos pode trazer insegurança aos docentes durante suas atividades. Assim, constantemente recorreu-se ao estudo de aplicações elaboradas por terceiros, para visualizar diferentes formas de se compreender o funcionamento de alguns recursos. Por isso, gastou-se tempo considerável para formular a primeira parte de quase todos os roteiros.

Figura 5 – Diagramas de conexão e teste em bancada para o roteiro proposto sobre PWM.



(a) (b)

Fonte: elaborado pelos autores, com auxílio do software *Fritzing*®.

Figura 6 – Código para programação e avaliação do funcionamento da montagem da Figura 5.

```
#include <esp32-hal-gpio.h>
#include <esp32-hal-ledc.h>
#include <esp32-hal-adc.h>
/* Declarando pinos do motor*/
uint8_t pino_adc = 34;
uint8_t pino1_motor = 26;
uint8_t pino2_motor = 27;
uint8_t pino_ativacao = 14;
/* Configurando propriedades do PWM*/
const int freq = 30000;
const int canal_PWM = 0;
const int resolucao = 12;
int ciclo_trabalho = 200;
void setup() { /* Configura a resolução do conversor ADC;*/
    analogReadResolution(resolucao);
/*Configura a resolução do conversor ADC;*/
    adcAttachPin(pino_adc);
/*Configura os pinos do motor como saídas*/
 pinMode(pino1_motor, OUTPUT);
 pinMode(pino2_motor, OUTPUT);
 pinMode(pino_ativacao, OUTPUT);
/*Configura o sinal PWM*/
    ledcSetup(canal_PWM, freq, resolucao);
/*Seta a GPIO do PWM*/
    ledcAttachPin(pino_ativacao, canal_PWM);
/*Inicia a comunicação Serial*/
    Serial.begin(115200); }
void loop() {
/*Ler o potenciômetro e determinar o ciclo de trabalho*/
    ciclo_trabalho = analogRead(34);
/*Move o motor DC para frente*/
    digitalWrite(pino1_motor, LOW);
    digitalWrite(pino2_motor, HIGH);
/*Laço de repetição da velocidade ajustada por sinal PWM*/
    while (ciclo_trabalho <= 4090){
/*Ajusta o ciclo de trabalho do sinal PWM*/
        ledcWrite(canal_PWM, ciclo_trabalho);
        /* Printa no monitor serial o ciclo de trabalho do sinal PWM */
        Serial.print("\nMovendo para frente - Ciclo de trabalho:");
        Serial.println(ciclo_trabalho);
/*Aumenta o ciclo de trabalho a cada repetição do laço*/
        ciclo_trabalho = ciclo_trabalho + 200; delay(500); }
}
```

Fonte: elaborado pelos autores.

5 CONSIDERAÇÕES

O presente trabalho teve como objetivo apresentar fases e resultados parciais de projeto em desenvolvimento desde março de 2020. Destaca-se que este projeto tem como um de seus objetivos capacitar discentes do Curso de Engenharia Elétrica do IFMG Campus Formiga a trabalhar com uma nova ferramenta e tecnologia. Além disso, tem-se como objetivo futuro efetivamente a aplicação de um curso, por exemplo, em formato de extensão para alunos não só da Engenharia Elétrica, mas também de outras Engenharias, visto que aplicações como IoT e Manufatura 4.0, são temas cada vez mais presentes e necessários para inserção no mercado atual.

Até o momento os roteiros elaborados não foram utilizados em sala de aula, mas os autores acreditam que os mesmos são promissores para a utilização da plataforma ESP32 como uma ferramenta didática. Espera-se que os mesmos sejam capazes de suprir os problemas relatados sobre os manuais e aplicações exemplo fornecidas pelo fabricante da plataforma, promovendo uma fácil compreensão e capacitação aos futuros alunos.

A plataforma será criada no período restante de duração do projeto, contendo todos os roteiros laboratoriais e informações úteis para o aprendizado sobre o ESP32.

Por fim, destaca-se que os roteiros desenvolvidos, e ainda em desenvolvimento, tem por base aplicar conceitos da metodologia de aprendizagem ativa, pois cada vez mais faz-se necessário deixar os alunos capacitados a aprender por si sós.

Agradecimentos

Os autores agradecem ao IFMG – Campus Formiga, em especial ao GSE – Grupo de Soluções em Engenharia, pela interação e colaboração no desenvolvimento do presente trabalho.

REFERÊNCIAS

AGHENTA, L. O.; IQBAL, M. T. Low-Cost, Open Source IoT-Based SCADA System Design Using Thinger.IO and ESP32 Thing. **Electronics**, v.8, n.822, 2019.

ALEXANDRE, Gerônimo Barbosa; FILHO, Múcio d'Emery Alves; SANTANA, Vitoria Borges. **Projeto e montagem de um medidor de energia elétrica didático e de baixo custo.** XLVII Congresso Brasileiro de Educação em Engenharia e II Simpósio Internacional de Educação em Engenharia da ABENGE. Fortaleza, 2019.

ARDUINO. **Arduino UNO Rev 3 Tech Specs.** Arduino, 2020. Disponível em: < https://store.arduino.cc/usa/arduino-uno-rev3>. Acessado em 07 de Agosto de 2020.

BARBOSA, P. V.; MEZZOMO, F.; LODDER, L. L. **Motivos de Evasão no curso de Engenharia Elétrica – Realidade e perspectivas**. XXXIX Congresso Brasileiro de Educação em Engenharia. Blumenau:[s.n.]. 2011.

BRIDI, E. et al. **Oficina de Arduino como ferramenta interdisciplinar no curso de engenharia elétrica da UFMT – A experiência do Pet-Elétrica**. XLI Congresso Brasileiro de Educação em Engenharia. Gramado: [s.n.]. 2013.

CAMPOS, Gustavo Lobato; SANTOS, Ana Paula Lima dos; OLIVEIRA, Patrick Santos de. **Arduino: uma nova ferramenta para a aprendizagem, estudo de caso no IFMG Campus Formiga.** XLVII Congresso Brasileiro de Educação em Engenharia e II Simpósio Internacional de Educação em Engenharia da ABENGE. Fortaleza, 2019.

ESPRESSIF. **Espressif announces the launch of esp32 cloud on chip and funding by fosun group**. Disponível em: https://www.espressif.com/en/media_overview/news/20160907-esp32briefing. Setembro 2016. Acesso em: 07 de Agosto de 2020.

ESPRESSIF. **ESP32 Series Datasheet – Version 3.4.** Disponível em: https://www.espressif.com/en/products/devkits/esp32-devkitc/overview. Agosto de 2020. Acessado em: 07 de Agosto de 2020.

ESPRESSIF. **ESP32 Modules.** Disponível em: <https://www.espressif.com/en/products/modules/esp32>, 2020. Acessado em 07 de Agosto de 2020.

GEDDES, Mark. **Manual de Projetos do Arduino: 25 projetos práticos para começar.** São Paulo: Novatec, 2017.

LIBÂNEO, José Carlos. **Didática.** 2 ed. São Paulo: Editora Cortez, 1999.

MATOS, B. T. M. *et al.* **Ensino da Robótica – O Arduino como ferramenta didática.** V Congresso Nacional de Educação. Olinda: [s.n.]. 2018.

MOREIRA, Michele Maria Paulino. *et al.* **Contribuições do Arduino no ensino de Física: uma revisão sistemática de publicações na área do ensino.** Caderno Brasileiro de Ensino de Física, v. 35, p. 721-745, dez. 2018.

OLIVEIRA, Alison Lopes de.; GONÇALVES, Willian Antonio; HOED, Raphael Magalhães. **Arduino: uma proposta para o ensino introdutório de programação.** Congresso Brasileiro de Ensino em Engenharia. Juiz de Fora, 2014.

THIRUPATHI, V.; SAGAR, K. Implementation of Home Automation System using MQTT Protocol and ESP32. **International Journal of Engineering and Advanced Technology**, v. 8, n. 2C2, p. 111–113, 1 dez. 2018.

VILAR, Sammara Raquel, *et al.* **Utilização da plataforma Arduino para a solução de problemas por alunos da disciplina de introdução à Engenharia Elétrica e algoritmos e lógica de programação do IFPB.** Revista Principia, n. 39, p. 72-78, 2018.

DEVELOPMENT OF A PLATFORM FOR LEARNING OF EMBARKED SYSTEMS BASED ON ESP32

Abstract: *The use of platforms like Arduino in technological courses proved to be effective and motivating, as it facilitated the practical understanding of various topics of electronics, automation, information systems and robotics. Despite being excellent in simple applications, basic entry cards use relatively old microcontrollers, containing simple peripheral resources and limited processing capacity. It is also difficult to develop modern applications on these cards, for example, devices connected to the Internet (Internet of Things) and Bluetooth® communication, as extra accessories (Shields) are needed which can make the project more expensive. This article will present a proposal for an ESP32 laboratory course, in an initiative to extend students' knowledge of embedded systems, improve projects previously developed in Arduino and create new projects in a new environment, with greater processing capacity and more resources available. The methodologies for developing the teaching materials for the classes, the production of laboratory scripts and the difficulties encountered by students in preparing these activities will be discussed.*

Keywords: *Embedded systems. Teaching methodologies. Problem-Based Learning.*