

# Empowering STEAM Activities with Artificial Intelligence and Open Hardware: the BitDogLab

Fabiano Fruett <sup>1</sup>, Fernanda Pereira Barbosa <sup>2</sup>, Pedro Ivo Aragão Guimarães <sup>2</sup>, and Samuel Cardoso Fraga Fraga <sup>2</sup>

<sup>1</sup>University of Campinas

<sup>2</sup>Affiliation not available

October 31, 2023

## Abstract

The paper “Empowering STEAM Activities with Artificial Intelligence and Open Hardware: the BitDogLab” discusses the design and development of BitDogLab, a tool intended to enhance Science, Technology, Engineering, Arts, and Mathematics (STEAM) activities. The device is based on the Raspberry Pi Pico board, employing an easy-to-use programming interface powered by the GPT model for artificial intelligence.

BitDogLab leverages open-source platforms, facilitating the expansion and enhancement of its functionalities. It is capable of interfacing with various software resources and more sophisticated hardware, such as robotic systems or advanced sensors. This ability broadens its scope of use in educational applications.

BitDogLab fosters a collaborative learning culture by encouraging community contributions, thereby strengthening the innovation process. Furthermore, BitDogLab is particularly useful in demystifying complex programming concepts, promoting critical thinking, and problem-solving skills. This transforms it from just an educational tool to a catalyst that inspires lifelong learning in the ever-evolving landscape of STEAM education.

# Empowering STEAM Activities with Artificial Intelligence and Open Hardware: the BitDogLab

Fabiano Fruett, Fernanda Pereira Barbosa, Samuel Cardoso Z. Fraga, Pedro Ivo Aragão Guimarães

**Abstract**—Open-source hardware and software platforms have played an important role in democratizing access to technology and education in computer science and engineering. They have significantly reduced costs and lowered barriers to entry for enthusiasts, educators, and students to start working on electronics, programming and other technologies. Recent advancements in open-source tools such as KiCad, MicroPython and the Thonny development environment have the potential to accelerate low-budget educational applications, providing a smooth and consistent learning curve for users. Using these open-source platforms, we designed and developed the BitDogLab, a tool for STEAM activities development using embedded systems based on the Raspberry Pi Pico board, with uncomplicated programming supported by artificial intelligence through the GPT model. BitDogLab is an open-source hardware solution with a flexible design that allows for continuous expansion for future developments in both software and hardware areas. Being based on open-source code, the functionality of BitDogLab can be enhanced by integrating it with a variety of software resources. This includes, for example, programming the board to interact with databases, a functionality that can be crucial in applications involving artificial intelligence. Additionally, BitDogLab can be adapted to work in conjunction with more sophisticated hardware, such as robotic systems or advanced sensors, expanding its range of educational applications. The project's open distribution also encourages community contributions, promoting a culture of collaborative learning and innovation.

**Index Terms**—Open Hardware, STEAM Education, Artificial Intelligence in Education, Project-Based Learning

## I. INTRODUCTION

**B**RAZIL, along with other developing countries, faces a significant challenge regarding the inclusion of technology in elementary and secondary school classrooms, especially in activities or projects involving Science, Technology, Engineering, Arts, and Mathematics (STEAM) [1]. This challenge is influenced by various factors such as teacher training, IT infrastructure, social and economic inequality, and more [2]. In Brazil, one particular factor is the reality of cascading taxes (federal, state, and municipal) that make technologically dependent solutions, subject to importation, prohibitively expensive for many schools, especially public ones. These taxes result in an approximately 120% increase in the final

We would like to express our profound gratitude to the IEEE Electron Device Society (EDS) for the generous grant that enabled us to develop STEM activities. Our appreciation also extends to INCT Namitec (National Institute of Science and Technology on Nano and Microelectronics). This initiative, conducted by CNPq (National Council for Scientific and Technological Development) in Brazil, provided valuable resources and expertise in the development of semiconductor devices that significantly contributed to the success of our project.

cost for consumers [3]. Furthermore, elementary and high school teachers face difficulties in using electronic components connected to a breadboard [4]. These difficulties mainly arise from common problems of poor contact on breadboards and the fragility of connections, which can lead to errors and frustration during practical teaching activities. Ideally, electronic boards designed for educational purposes should be easy to use and operate under the principle of open-source hardware, providing their manufacturing "recipe" to encourage sharing and continuous improvement. It is also important for them to be well-documented to facilitate independent learning for students [5]. The programming environment of the board should prioritize logical thinking over the syntax of the programming language, as the latter can become an obstacle for students who are starting to develop their programming and computational thinking skills [6].

An initiative evaluated the effectiveness of Fab Lab-based learning and shows that students are more interested in science lessons after using Fab Lab-based learning [7]. This research corroborates an analysis of 225 studies comparing student performance in STEM courses using traditional learning versus active learning methods, revealing a significant positive effect of active learning on student performance [8]. Given this scenario, it is essential to seek low-cost solutions that meet the requirements while providing quality learning experiences and effective development of students' skills. Therefore, we have developed BitDogLab: an open-source, low-cost, and user-friendly hardware board designed specifically to teach STEAM concepts in an interactive and engaging manner. This article will describe the development and capabilities of the BitDogLab, demonstrating how it can be programmed with the support of artificial intelligence to facilitate and expedite the learning process.

## II. DESIGN OF THE BITDOGLAB PRINTED CIRCUIT BOARD

While the use of protoboards is a common educational tool, it can pose technical challenges for elementary and high school teachers and students. Therefore, we consider it advisable to avoid the protoboard as the first form of contact with electronic components. In this way, our project is based on a specially designed printed circuit board (PCB) for educational activities. We have named the board BitDogLab, in honor of our mascot BitDog, which is a playful element of the Escola 4.0 project [9] to which this initiative belongs. This PCB uses as its

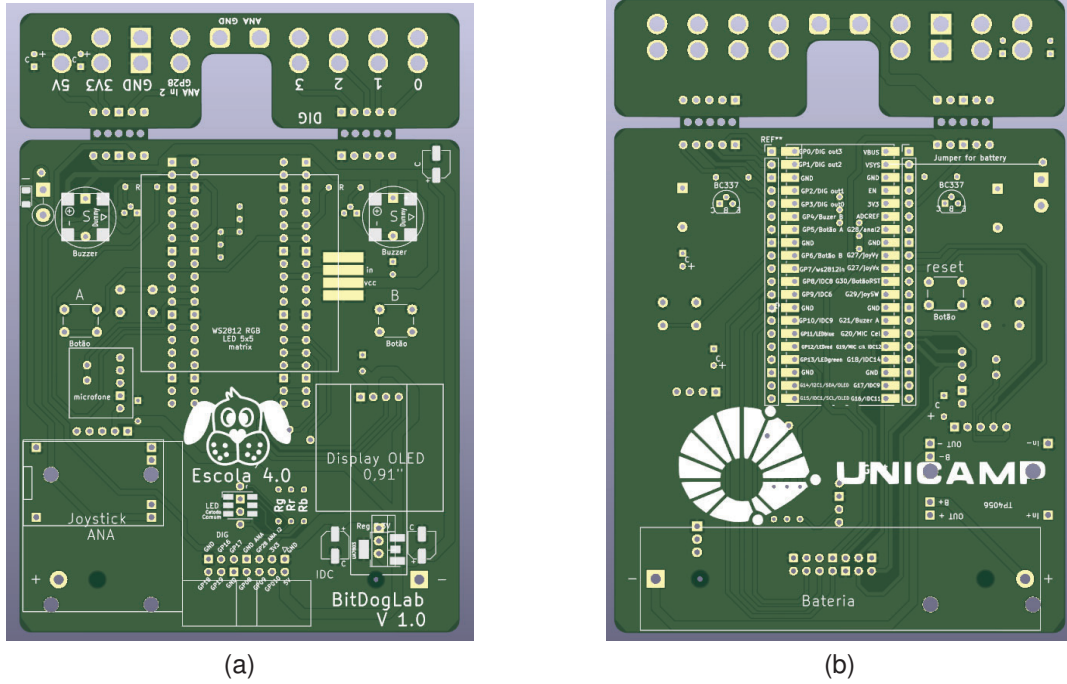


Fig. 1. Layout design of the BitDogLab PCB: (a) front and (b) back side.

central component a Raspberry Pi Pico, a module with a low-cost (US\$6) and high-performance ARM microcontroller, which offers programming flexibility and robust features for teaching concepts of Science, Technology, Engineering, Arts, and Mathematics (STEAM) [10]. The BitDogLab PCB was designed to have two parts that can be easily separated. Its layout is shown in Figure 1a and Figure 1b. The upper part, which has several drilled holes, is a terminal bar to facilitate connection with alligator-type clamps. The main core of the board is where the components are soldered. The main core of the board measures 98 mm in length, 88 mm in width, and 1.6 mm in thickness. The terminal bar is 23 mm in length, while retaining the same width and thickness as the main core.

The BitDogLab PCB connects the microcontroller with various peripheral components that can be soldered onto the board as pedagogical needs demand or as the user's interest increases. These connections are made with two levels of metallization, one on each face of the board. The board was designed to offer flexible and accessible alternatives for electronic components, both in the through-hole assembly (PTH) mode and in the surface-mount device (SMD) mode. This approach allows the user to choose the best assembly strategy according to their skills and learning goals. For example, PTH components may be more suitable for beginners due to the ease of soldering, while SMD components can offer a more advanced experience, closer to what is found in the industry. This feature is shown in Figure 2.

The repository with all Gerber files for manufacturing and also the KiCad source file for BitDogLab is available at [11]. There are several manufacturers around the world that offer PCB prototyping. The manufacturing cost of this board

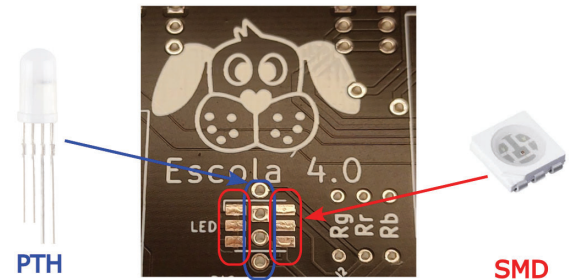


Fig. 2. Detail of the choice of the type of component and how to solder the colored LED on the BitDogLab PCB. The choice depends on the user's skills and knowledge, it can be: through-hole (PTH) or surface mount (SMD).

depends on the quantity requested. For less than a dozen units, the approximate cost is \$1 per board. The components of the project are detailed in the next section.

#### A. Materials

In addition to the BitDogLab board, the materials used in this project are: a five by five matrix of colored (RGB) LEDs, two buzzers, three push buttons, a microphone, a joystick, an RGB LED, a mini display, and an Insulation-Displacement Connector (IDC) that allows externally connecting other components or boards, expanding its functions. In addition to components such as resistors and capacitors. All of these components are commercially available and easy to acquire. The list of materials is shown in Table I.

TABLE I  
BITDOGLAB'S BILL OF MATERIALS

item	tag	description	quantity
1		20-pin female header	2
2		Raspberry pi pico (H or W)	1
3	LED	5mm RGB LED common cathode	1
4	Rb	100 ohm resistor 1/8 W 5%	1
5	Rg, Rr	200 ohm resistor 1/8 W 5%	2
6	A, B, Reset	button	3
7	A, B	passive buzzer	2
8	Q1, Q2	NPN transistor BC337	2
9	R4, R5	330 ohm resistor 1/8W 5 %	2
10	M	5 x 5 Matrix RGB LEDs 5050 WS2812B	1
11	JOY	KY023 analog joystick	1
12	reg	Voltage regulator 3V3 LM1117	1
13	MIC	PDM microphone module MP34DT01	1
14	C1-C6	100uF 16V electrolytic capacitor	6
15	DSP	0.96" I2C 128 x 64 OLED display	1
16	bat	rechargeable battery holder for PCB	1
17	CHM	charge module tp4056 mini USB	1

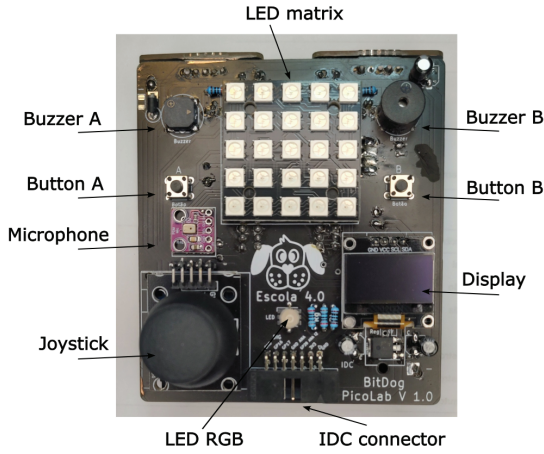


Fig. 3. Front view of the assembled BitDogLab with all its components.

### B. Method for Hardware Implementation

With the BitDogLab PCB in hand, we suggest that each component be included, by soldering it onto the board, in the order that appears in Table I. In this way, after incorporating items 1 and 2, the user has the possibility to progress and program using the approach presented in this article. As more components are included, your options for use in STEAM activities increase along with your programming options. The soldering of the components onto the board is a task that requires appropriate guidance and safety precautions. With proper supervision, young people from 12 years old are capable of performing quality soldering work. There are numerous tutorials and online videos that offer step-by-step instructions on how to solder correctly. Although this task requires some practice, once acquired, soldering skills become a lifelong useful competency. Furthermore, with the right guidance, the activity can become not only educational but also fun. The PCB with all the soldered components is shown in Figure 3.

The board has some terminals for connecting alligator clips. In this way, the board's expandability is facilitated by these terminals and also by a Insulation-Displacement Connector (IDC) with 14 vias, which are detailed in Figure 4. An IDC is



Fig. 4. Close-up view of the connection terminals of the BitDogLab that can be connected with alligator clips.

an electrical connector that allows connections to be made with insulated cables without the need to remove the insulation. In practice, this means that electronic components can be easily connected to cables, without the need for complex cable preparation. This simplifies the assembly and maintenance process, making it ideal for educational applications where ease of use and expansibility are fundamental.

## III. THE PROGRAMMING ENVIRONMENT

The hardware we described earlier is the physical implementation of our tool for empowering STEAM Activities. The core of this tool is a microcontroller that must be programmed to command or receive signals from components, sensors, and peripherals. This microcontroller can be programmed in a variety of ways [12]. However, the main contribution of this article is to propose a method that allows anyone, even without any prior programming experience, to use this tool in STEAM projects in the classroom.

The programming environment comprises the Integrated Development Environment (IDE), Thonny [13], and a Large Language Model (LLM) of the GPT type. Thonny is a Micropython IDE that is incredibly beginner-friendly, with features like step-by-step debugging and variable monitoring. The main interface of Thonny is shown in Figure 5 [13].

Micropython is a compact and efficient implementation of the Python language that can be used to program microcontrollers [14]. Chat GPT is an artificial intelligence feature [15] that can assist users in programming the BitDogLab based on its Hardware Configuration Database.

## IV. HARDWARE CONFIGURATION DATABASE

The Hardware Configuration Database (HCD), which we propose, is a structured textual list containing information about the BitDogLab board's hardware configuration. This database includes details about the arrangement and function



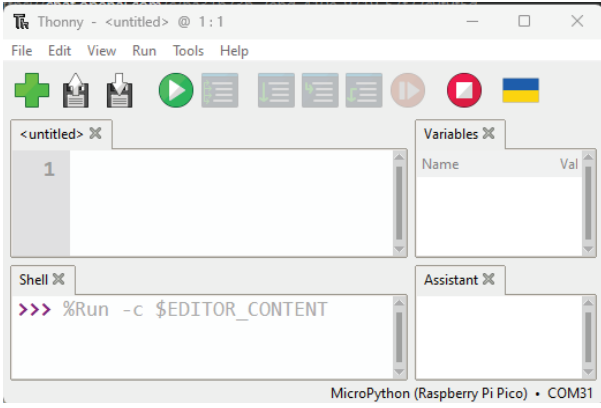


Fig. 5. Thonny's Integrated Development Environment (IDE).

of the hardware components, including but not limited to: connected peripherals and the GPIO pins of the Raspberry Pi Pico microcontroller. This set of information simplifies understanding the board's architecture and allows users, and even Artificial Intelligence models like ChatGPT, to interact efficiently with the hardware. For each peripheral, the database includes details about the connected GPIO pin, the expected function of this component, and any specific parameters associated with its use. The HCD not only serves as a reference guide but also aids in the development of STEAM (Science, Technology, Engineering, Arts, and Mathematics) applications and activities with the BitDog PicoLab board. Moreover, it is essential for customizing the AI model's responses, enabling it to provide specific and relevant information, particularly concerning programming code, based on the board's hardware configuration. The BitDogLab Hardware Configuration Database is the textual list shown below:

- A common-cathode RGB LED has the red electrode connected to GPIO 12 through a 200-ohm resistor, the green pin connected to GPIO 13 through a 200-ohm resistor, and the blue pin to GPIO 11 through a 100-ohm resistor.
- A button, identified as Button A, is connected to the Raspberry Pi Pico's GPIO5. The other terminal of the button is connected to the board's GND.
- Another button, identified as Button B, is connected to the Raspberry Pi Pico's GPIO6. The other terminal of the button is also connected to the board's GND.
- A buzzer, identified as Buzzer A, is connected to the Raspberry Pi Pico's GPIO21.
- Another buzzer, identified as Buzzer B, is connected to the Raspberry Pi Pico's GPIO4.
- The input pin of a 5-line by 5-column 5050 RGB LED matrix type WS2812B (Neopixel) is connected to GPIO7.
- A KY023 type analog joystick has its VRx output connected to GPIO26 and its VRy output to GPIO27. Its SW button is connected to GPIO22.
- A 0.96-inch 128-column by 64-line OLED display with I2C communication has its SDA pin connected to GPIO14 and the SCL pin to GPIO15.
- An Insulation-Displacement Connector (ICD) box of 14

pins is used for hardware expansion and is connected to the Raspberry Pi Pico as follows: pin 1 with GND, pin 2 with 5V, pin 3 with 3V3, pin 4 with GPIO10, pin 5 with GPIO28, pin 6 with GPIO9, pin 7 with analog GND. Pin 8 with GPIO8, pin 9 with GPIO17, pin 10 with GND, pin 11 with GPIO16, pin 12 with GPIO19, pin 13 with GND, pin 14 with GPIO18.

- A terminal bar that allows the connection of alligator clip-type terminals is connected to the Raspberry Pi Pico as follows: the DIG 0, 1, 2, and 3 electrodes are connected to GPIOs 3, 2, 1, and 0, respectively. Additionally, this terminal bar has five more electrodes connected to: analog GND, GPIO 28, GND, 3V3, and 5V of the Raspberry Pi Pico.

This Hardware Configuration Database is the key to making the BitDogLab board an accessible and versatile tool for STEAM education, providing all the necessary information for users and AI models to interact efficiently with the hardware.

## V. AI-ASSISTED PROGRAMMING

In the following, we will describe how an artificial intelligence algorithm can be used as a Programming Support Tool for BitDogLab. When we say "programming using Artificial Intelligence (AI)", this may be interpreted as if the AI is executing the programming autonomously. However, current AI models, including the GPT model, do not yet have this capability. They cannot write complex computer programs independently, and more specifically, they do not have the ability to program hardware boards. What AI models like GPT can do is assist humans in the task of programming. They can provide code suggestions, explain programming concepts, assist in debugging errors, generate small snippets of code in response to specific requests, and so forth. In the case of the BitDogLab project, from the Hardware Configuration Database, the AI can help transcribe and explain the code to control different components on the board, such as the RGB LEDs, the buzzers, or the display from commands of buttons, joysticks or sensors. Therefore, when we say that BitDogLab board is "programmed using AI", this means that AI is being used as a tool to assist humans in programming the board. In this case, AI is used as an intelligent assistant that can suggest, guide, and explain, but the final task of writing, testing, debugging, and implementing the code is guided and developed by humans.

### A. Method for hardware programming using AI

We propose a programming methodology, with AI support, that can be utilized in the development of a variety of interdisciplinary learning projects and activities. This methodology is characterized by an iterative approach, focused on the use of a Hardware Configuration Database of open-source hardware, which is incorporated into the conversational platform of the AI model. Before starting to use the methodology, it is assumed that the user is already engaged in a STEAM project that may involve electronics, sensors, actuators, and programming in its implementation. The methodology is summarized

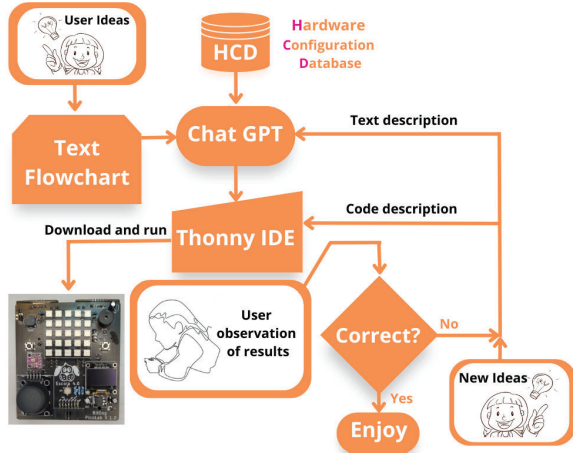


Fig. 6. Flow diagram of the programming methodology with AI assistance.

in the flow diagram shown in Figure 6. The steps of the methodology are detailed as follows:

- 1) **Loading the Hardware Configuration Database into Chat GPT:** This involves inserting the detailed information, in the form of a textual list, of the BitDogLab board into the GPT conversational platform. This includes information about the layout of the hardware components, the functions of each GPIO pin, etc.
- 2) **Use of a Textual Flowchart:** Editing or writing a Textual Flowchart: The user's ideas for employing the BitDogLab in the STEAM project should be transcribed in the form of a textual flowchart, using logical thinking. This flowchart serves as a guide for the AI model, directing it on how to utilize the Hardware Configuration Database to assist in the creation of MicroPython code.
- 3) **Copying and Loading the Code into Thonny:** After the code is created in collaboration with the AI, it is copied and loaded into the Thonny development environment, which is specific for MicroPython programming and supports the Raspberry Pi Pico.
- 4) **Download the Code to the Raspberry Pi Pico:** The code is then downloaded to the Raspberry Pi Pico board through the Thonny IDE.
- 5) **Execution of the Program:** From a command given by the user in Thonny, the Raspberry Pi Pico board executes the loaded code, performing the programmed actions.
- 6) **Observation of Results:** The results of the execution are observed by the user who analyzes and contrasts them with their expectations and needs. At this point, opportunities to increase both the hardware, by soldering or connecting a new component, as well as modifying the code, can also be identified.
- 7) **Corrections with the AI:** Any deviation from the expected behavior or need for change is discussed with the AI model, which uses the Hardware Configuration Database to suggest possible corrections to the code.

This approach allows the student to go through all the stages of a programming cycle and obtain results quickly, driving the process. Even without knowing details of the programming

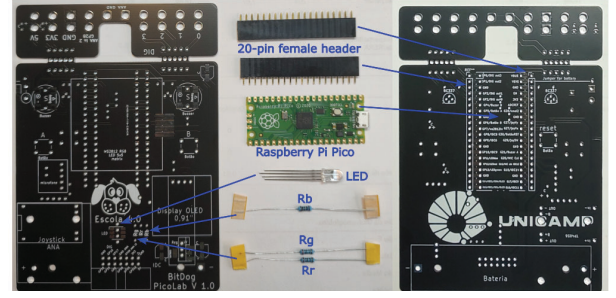


Fig. 7. Front and back view of the PCB showing the basic components to be soldered in the positions indicated on the board to carry out one of the STEAM activities.

language syntax, the user can develop skills related to logic and computational thinking. As their interest increases, the board can be expanded with new on-board components as well as off-board ones.

## VI. STEAM ACTIVITY EXAMPLE WITH BITDOGLAB

In this section, we show an example of a STEAM activity using the BitDogLab, which can be programmed using the proposed methodology. As an initial application example, we prepared the PCB by soldering a few components, such as: two 20-pin rows (to connect the Raspberry Pi Pico), a single RGB-colored LED, and three resistors on the BitDogLab board. These components are identified in Table 1 by their tags: LED, Rr, Rg, and Rb. They are also identified on the PCB in the position they should be soldered, as shown in Figure 7.

A pedagogical activity example, using only these components, involves programming so that the LED displays the seven colors of the rainbow. This activity, although simple, serves as a programming exercise, and also as a multidisciplinary teaching tool. In this programming exercise, one can explore mathematical concepts by involving the calculation of the percentages that determine the intensity of each color displayed by the LED. The same activity can also be used to introduce science and art concepts related to color theory, including primary colors and their blends. Thus, the activity incorporates STEAM disciplines concepts in an integrated way. After soldering these few components, we follow the methodology previously presented. After opening the free GPT Chat interface, we load the Hardware Configuration Database (HDC) into the dialog window. In the same GPT, using the next dialog window, we load the textual flowchart of operations to be performed. As an example, we can write: **Considering our Hardware Configuration Database, could you make a code to vary the colors of the RGB LED with the sequence of the seven colors of the rainbow, starting with red and going to violet? The transition of the seven colors should take 21 seconds and restart again.**

After a few seconds, the response followed by the commented code appears in the Chat GPT response window, as shown in Figure 8.

The code can be copied simply by clicking on the 'Copy code' icon and pasting it into the Thonny's window [11] as shown in Figure 9. The step-by-step guide on how to configure

```

python
Copy code

from machine import Pin, PWM
import utime

# Frequency for PWM control
FREQUENCY = 1000

# Colors of the rainbow (red, orange, yellow, green, blue, indigo, violet)
# The values are adjusted for a common cathode RGB LED
rainbow_colors = [
    (1023, 0, 0),      # Red
    (1023, 165, 0),    # Orange
    (1023, 1023, 0),   # Yellow
    (0, 1023, 0),      # Green
    (0, 0, 1023),      # Blue
    (75, 0, 130),      # Indigo
    (238, 130, 238)    # Violet
]

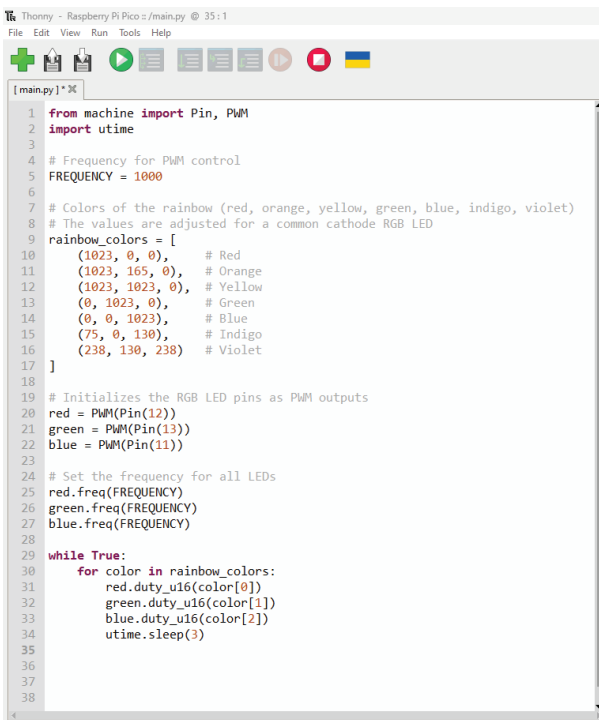
# Initializes the RGB LED pins as PWM outputs
red = PWM(Pin(12))
green = PWM(Pin(13))
blue = PWM(Pin(11))

# Set the frequency for all LEDs
red.freq(FREQUENCY)
green.freq(FREQUENCY)
blue.freq(FREQUENCY)

while True:
    for color in rainbow_colors:
        red.duty_u16(color[0])
        green.duty_u16(color[1])
        blue.duty_u16(color[2])
        utime.sleep(3)

```

Fig. 8. Chat GPT model's response to the structured question suggested by the method of this article.



```

Thonny - Raspberry Pi Pico 35:1
File Edit View Run Tools Help

[main.py]*K

1 from machine import Pin, PWM
2 import utime
3
4 # Frequency for PWM control
5 FREQUENCY = 1000
6
7 # Colors of the rainbow (red, orange, yellow, green, blue, indigo, violet)
8 # The values are adjusted for a common cathode RGB LED
9 rainbow_colors = [
10     (1023, 0, 0),      # Red
11     (1023, 165, 0),    # Orange
12     (1023, 1023, 0),   # Yellow
13     (0, 1023, 0),      # Green
14     (0, 0, 1023),      # Blue
15     (75, 0, 130),      # Indigo
16     (238, 130, 238)    # Violet
17 ]
18
19 # Initializes the RGB LED pins as PWM outputs
20 red = PWM(Pin(12))
21 green = PWM(Pin(13))
22 blue = PWM(Pin(11))
23
24 # Set the frequency for all LEDs
25 red.freq(FREQUENCY)
26 green.freq(FREQUENCY)
27 blue.freq(FREQUENCY)
28
29 while True:
30     for color in rainbow_colors:
31         red.duty_u16(color[0])
32         green.duty_u16(color[1])
33         blue.duty_u16(color[2])
34         utime.sleep(3)
35
36
37
38

```

Fig. 9. Thonny IDE with code generated by Chat GPT.

the Thonny IDE along with the microcontroller (Raspberry Pi

Pico) is out of the scope of this article's innovation and will not be detailed. For users less familiar with the initial setup of the Raspberry Pi Pico using the Thonny IDE, including the installation of Micropython, we recommend the "Getting started with Raspberry Pi Pico" document from the Raspberry Pi Foundation [16]. After loading the code into the BitDogLab, the program begins to execute the requested tasks. The result is shown in Figure 10.

Should there be any errors before the desired outcome is achieved, the error code will appear in the Thonny Shell window. This error information can be copied into a GPT Chat window that will rewrite the program, with the user simply observing and repeating the process. Observing and comparing the achieved result with the desired one, based on the textual flowchart, is a very important stage of the process for the advancement of students' STEAM skills.

For example, note that the difference between orange and yellow, as shown in Figure 10, is very subtle and may cause difficulty in distinguishing one from the other. This particular result can become a challenge for students to improve the code. Thus, they may be motivated to adjust the intensity of each color in the code to deliver a result where yellow is more distinguishable than orange. For this, they should exercise concepts of proportion, rule of three, color blending, and logical thinking. From this example, the user can expand to include new possibilities, such as: playing the Buzzer at a frequency proportional to the wavelength of each color's electromagnetic wave. This would be a suitable example to explain the electromagnetic spectrum and relate its frequencies to a practical outcome. Thus, in a more intuitive way, students could understand the relationship between frequency and wavelength in the electromagnetic spectrum.

Alternatively, they could include the display to show the wavelength (in nm) of each color being displayed by the colored LED. The commands to change the color could also come from the joystick or from claps that would be picked up by the microphone.

## VII. DISCUSSION AND CONCLUSIONS

This work highlights the importance of an integrated and suitable approach to the development of STEAM projects, promoting a holistic and practical understanding, as well as including hands-on experience with electronics and programming. The experimental nature of the BitDogLab enables students to apply theoretical knowledge, by creating and programming real devices, thus solidifying meaningful learning. The BitDogLab also offers interactive features that enhance the educational experience. To start, we highlight an example where the correlation of buzzer frequencies with electromagnetic wavelengths emitted by the LED provides a practical and multimodal approach to teaching complex concepts, such as the electromagnetic spectrum. The implementation of various user interfaces, involving: lights, sounds, and movement, increases the interactivity of the learning experience provided by BitDogLab and demonstrates its flexibility. Future research can assess the effectiveness of this methodology in stimulating interdisciplinary learning and the development of programming



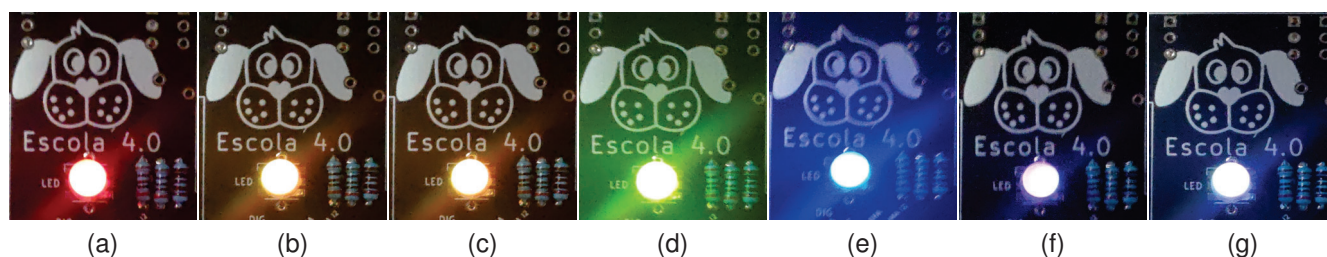


Fig. 10. Result of the RGB LED displaying the seven colors of the rainbow in sequence.

skills. The application of this methodology in other contexts and with different types of hardware is also a promising area for future investigation. Therefore, the central contribution of this work to the field of science and technology education is the proposal of an iterative programming process that integrates AI into the development process, using the Hardware Configuration Database (HCD). This approach, which harmoniously unites artificial intelligence and open hardware in the form of BitDogLab, has been specifically designed to simplify the programming task for both students and educators. In doing so, it not only demystifies a complex subject, such as dealing with complex coding syntax, but also actively encourages the development of critical thinking and problem-solving skills. In this way, BitDogLab can be an empowering tool for STEAM activities, providing an environment where curiosity is rewarded with knowledge and challenges serve as stepping stones to understanding. This confluence of elements propels BitDogLab beyond a mere educational tool, shaping it as a catalyst to inspire lifelong learners in the ever-evolving landscape of STEAM education.

## REFERENCES

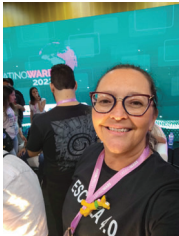
- [1] S. Marginson, R. Tytler, B. Freeman, and K. Roberts, "STEM: country comparisons: international comparisons of science, technology, engineering and mathematics (STEM) education. Final report," 1 2013. [Online]. Available: [https://dro.deakin.edu.au/articles/book/STEM\\_country\\_comparisons\\_international\\_comparisons\\_of\\_science\\_technology\\_engineering\\_and\\_mathematics\\_STEM\\_education\\_Final\\_report\\_/20951737](https://dro.deakin.edu.au/articles/book/STEM_country_comparisons_international_comparisons_of_science_technology_engineering_and_mathematics_STEM_education_Final_report_/20951737)
- [2] S. Mejias, N. Thompson, R. M. Sedas, M. Rosin, E. Soep, K. Pepler, J. Roche, J. Wong, M. Hurley, P. Bell, and B. Bevan, "The trouble with steam and why we use it anyway," *Science Education*, vol. 105, no. 2, pp. 209–231, 2021. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/sce.21605>
- [3] "Brazilian Tax aspects on import of services," <https://www.pnst.com.br/en/articles/tax-consulting-and-litigation/brazilian-tax-aspects-import-services/>, accessed: 2023-06-03.
- [4] K. Pepler, M. Sedas, and N. Thompson, "Paper circuits vs. breadboards: Materializing learners' powerful ideas around circuitry and layout design," *Journal of Science Education and Technology*, pp. 1–24, 05 2023.
- [5] R. Heradio, J. Chacon, H. Vargas, D. Galan, J. Saenz, L. De La Torre, and S. Dormido, "Open-source hardware in education: A systematic mapping study," *IEEE Access*, vol. 6, pp. 72 094–72 103, 2018.
- [6] P. Trivedi, P. Kajgaonkar, A. Kulkarni, N. Kolte, and B. Kanawade, "System model for syntax free coding," in *2019 Global Conference for Advancement in Technology (GCAT)*, 2019, pp. 1–5.
- [7] M. A. Togou, C. Lorenzo, G. Cornetta, and G.-M. Muntean, "Assessing the effectiveness of using fab lab-based learning in schools on k–12 students' attitude toward steam," *IEEE Transactions on Education*, vol. 63, no. 1, pp. 56–62, 2020.
- [8] S. Freeman, S. L. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt, and M. P. Wenderoth, "Active learning increases student performance in science, engineering, and mathematics," *Proceedings of the National Academy of Sciences*, vol. 111, no. 23, pp. 8410–8415, 2014. [Online]. Available: <https://www.pnas.org/content/111/23/8410>
- [9] "Inspire seus alunos com steam," <https://escola4pontozero.fee.unicamp.br/>, accessed: 2023-06-03.
- [10] "Buy a raspberry pi pico," <https://www.raspberrypi.com/products/raspberry-pi-pico/>, accessed: 2023-06-03.
- [11] "Bitdoglab repository," <https://github.com/Fruett/BitDogLab>, accessed: 2023-06-03.
- [12] "Raspberry pi os," <https://www.raspberrypi.com/software/>, accessed: 2023-06-03.
- [13] A. Annamaa, "Thonny: A python ide for learning programming," in *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*, ser. ITiCSE '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 343. [Online]. Available: <https://doi.org/10.1145/2729094.2754849>
- [14] "Micropython," <https://micropython.org/>, accessed: 2023-06-03.
- [15] OpenAI, "Gpt-4 technical report," 2023.
- [16] "Getting started with raspberry pi pico," <https://projects.raspberrypi.org/en/projects/getting-started-with-the-pico/2>, accessed: 2023-06-03.





**Fabiano Fruett** Fabiano Fruett received his degree in electrical engineering in 1994 from the Universidade Estadual Paulista - UNESP-Ilha Solteira, and his Master's degree in microelectronics in 1997 from the Universidade Estadual de Campinas - UNICAMP. In the same year, he began his work as a researcher in the field of microelectronic sensors at the Delft University of Technology, Netherlands. Fruett obtained his Ph.D. in September 2001, where the primary findings of his doctoral research were published in over a dozen scientific articles and a

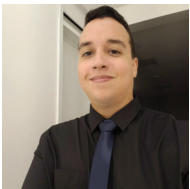
book authored by him. Since 2002, he has been working as a faculty member at the School of Electrical and Computer Engineering (FEEC) at UNICAMP. He is currently an Associate Professor III in the renamed Department of Electronics and Biomedical Engineering of the FEEC. His research interests are in the areas of electronic instrumentation, Internet of Things (IoT), and open-source STEAM learning tools.



**Fernanda Pereira Barbosa** Master's degree in Chemistry from the Department of Chemistry at UFSCar. Supporter of the Escola 4.0 project. Curator of social networks for the project Escola 4.0.



**Samuel Cardoso Zampolli Fraga** Undergraduate student in Electrical Engineering at State University of Campinas - Unicamp.



**Pedro Ivo Aragão Guimarães** Mr. Guimarães is currently a M.Sc. student in Electronic Engineering at the Federal Institute of Education, Science and Technology of Paraíba—IFPB, Brazil. He received the B.Sc. in Electronic Engineering from the Federal University of Campina Grande-UFCG, Brazil, in 2021.