DOI: 10.64302/joshusc.v30n1e.1332

NGHIÊN CỨU PHÁT TRIỂN KHỐI LỆNH LẬP TRÌNH KÉO THẢ CHO NỀN TẢNG ARDUINO DỰA TRÊN MÃ NGUỒN MỞ BLOCKLY

Nguyễn Phan Nguyên Bảo[1], Nguyễn Văn Hướng[2], Nguyễn Văn Thái Bảo[1], Nguyễn Đức Nhật Quang[1], Trần Thị Kiều[1*]

[1] Khoa Điện, Điện tử và Công nghệ vật liệu, Trường Đại học Khoa học, Đại học Huế

[2] Khoa Công nghệ thông tin, Trường Đại học Khoa học, Đại học Huế

*Email: kieutran@husc.edu.vn

Ngày nhận bài: 24/5/2025; ngày hoàn thành phản biện: 8/6/2025; ngày duyệt đăng: 24/6/2025

TÓM TẮT

Bài báo trình bày quy trình thiết kế và phát triển hệ thống lập trình kéo thả cho Arduino dựa trên mã nguồn mở Google Blockly, được triển khai trực tuyến tại địa chỉ https://robocon.husc.edu.vn/. Hệ thống cho phép người dùng lập trình và sinh mã C/C++ tương thích với Arduino ngay trên web, không cần cài đặt phần mềm. Các khối lệnh được tổ chức thành 11 nhóm chức năng, có thể mở rộng một cách dễ dàng và linh hoạt bằng cách định nghĩa mới qua JavaScript và XML, đáp ứng hiệu quả các bài toán điều khiển, giao tiếp và xử lý tín hiệu trong lập trình nhúng. Thử nghiệm trên Arduino Uno và Nano cho thấy mã sinh ra đảm bảo độ chính xác cú pháp và ổn định khi thực thi, chứng minh tính đúng đắn và tin cậy của hệ thống. So với các nền tảng như ArduBlockly, mBlock hay CloverBlock – vốn bị giới hạn về khả năng tùy biến và yêu cầu cài đặt cục bộ – hệ thống đề xuất thể hiện tính mở và khả năng mở rộng vượt trội, thuận tiện cho giảng dạy, nghiên cứu và triển khai STEM. Hệ thống cũng đã được ứng dụng thành công tại cuộc thi Robocon HUSC 2025, khẳng định hiệu quả và độ ổn định cao trong môi trường thực hành và thi đấu robot.

Từ khóa: Arduino, Google Blockly, Drag-and-drop programming.

1. MỞ ĐẦU

Trong bối cảnh giáo dục STEM ngày càng chú trọng đến việc phát triển tư duy lập trình và kỹ năng tính toán, nhu cầu học lập trình điều khiển nhúng, đặc biệt với nền tảng Arduino – một vi điều khiển mã nguồn mở phổ biến và chi phí thấp – đang gia tăng mạnh mẽ. Tuy nhiên, việc lập trình bằng ngôn ngữ truyền thống C/C++ vẫn là thách thức lớn đối với người mới bắt đầu do yêu cầu cao về cú pháp và tư duy logic. Để giảm rào

cản này, các mô hình lập trình trực quan dạng khối (block-based programming) đã được nghiên cứu và ứng dụng rộng rãi, góp phần đơn giản hóa quá trình học lập trình và khơi gợi tư duy sáng tạo cho người học.

Trong số đó, Google Blockly [1] – thư viện mã nguồn mở do Google phát triển – đã chứng minh tính linh hoạt và hiệu quả trong việc xây dựng giao diện lập trình trực quan, là nền tảng cho nhiều công cụ phổ biến như Scratch for Arduino (S4A) [2], mBlock [3], và ArduBlockly [4]. Các nghiên cứu của Fraser (2013), Sun & Looi (2024) và Fagerlund & Häkkinen (2020) đều chỉ ra rằng lập trình kéo thả giúp nâng cao tư duy thuật toán, giảm lỗi cú pháp và cải thiện hiệu quả học tập so với lập trình văn bản.

Mặc dù vậy, các công cụ hiện có vẫn bộc lộ nhiều hạn chế về khả năng triển khai trực tuyến và mở rộng khối lệnh.

- S4A hoạt động hoàn toàn offline, yêu cầu cài đặt phần mềm và không hỗ trợ chạy trên nền web.
- mBlock có phiên bản web tại ide.mblock.cc, nhưng chủ yếu phục vụ ngôn ngữ Scratch và Python, chưa tích hợp đầy đủ chức năng lập trình Arduino, nên chưa đáp ứng tốt cho các bài toán điều khiển nhúng phức tạp.
- ArduBlockly cho phép triển khai trực tuyến thông qua máy chủ cục bộ, song việc cấu hình và mở rộng đòi hỏi kỹ thuật phức tạp.
- CloverBlock, một công cụ được phát triển độc lập, có giao diện thân thiện và hỗ trợ lập trình kéo thả trực quan; tuy nhiên, hệ thống chỉ chạy cục bộ và việc tùy biến hoặc mở rộng thêm các khối lệnh mới tương đối phức tạp, đòi hỏi can thiệp trực tiếp vào mã nguồn. Điều này hạn chế khả năng mở rộng và thích ứng của công cụ trong môi trường giảng dạy, nghiên cứu và phát triển ứng dụng thực tế.

Xuất phát từ thực tiễn đó, nghiên cứu này tập trung thiết kế và phát triển hệ thống lập trình kéo thả cho Arduino dựa trên mã nguồn mở Blockly, với mục tiêu khắc phục các hạn chế của những công cụ hiện có. Hệ thống được triển khai hoàn toàn trực tuyến tại địa chỉ https://robocon.husc.edu.vn, cho phép người dùng lập trình, sinh mã C/C++ tương thích với Arduino trực tiếp trên nền web mà không cần cài đặt phần mềm, đồng thời dễ dàng mở rộng và tùy chỉnh các khối lệnh thông qua cấu trúc JavaScript và XML của Blockly.

Kết quả triển khai thực tế cho thấy hệ thống hoạt động ổn định trên các bo mạch Arduino Uno và Nano, sinh mã C/C++ chính xác, đảm bảo tính tương thích và khả năng thực thi tin cậy. Đặc biệt, hệ thống đã được ứng dụng thành công trong cuộc thi Robocon HUSC 2025, nơi các đội thi sử dụng trực tiếp nền tảng này để lập trình và hoàn thiện chương trình điều khiển robot. Kết quả cuộc thi đã chứng minh tính ổn định, độ tin cậy

và khả năng mở rộng linh hoạt của hệ thống, qua đó khẳng định tính khả thi và hiệu quả ứng dụng cao của giải pháp trong giảng dạy, nghiên cứu và thi đấu kỹ thuật trong lĩnh vực STEM.

2. THIẾT KẾ KHỐI LỆNH LẬP TRÌNH KÉO THẢ DỰA TRÊN MÃ NGUỒN MỞ BLOCKLY

2.1. Lập trình kéo thả và mã nguồn mở Blockly

Lập trình kéo thả (drag-and-drop programming) là một phương pháp lập trình trực quan, trong đó chương trình được xây dựng bằng cách sắp xếp các khối lệnh đồ họa thay vì viết mã văn bản. Mỗi khối lệnh biểu diễn một đơn vị chức năng tương đương với cú pháp trong ngôn ngữ lập trình truyền thống. Phương pháp này giúp người học, đặc biệt là người mới bắt đầu, dễ dàng tiếp cận với các khái niệm lập trình cơ bản mà không cần hiểu rõ cú pháp, vốn là rào cản phổ biến trong quá trình học ngôn ngữ như C/C++.

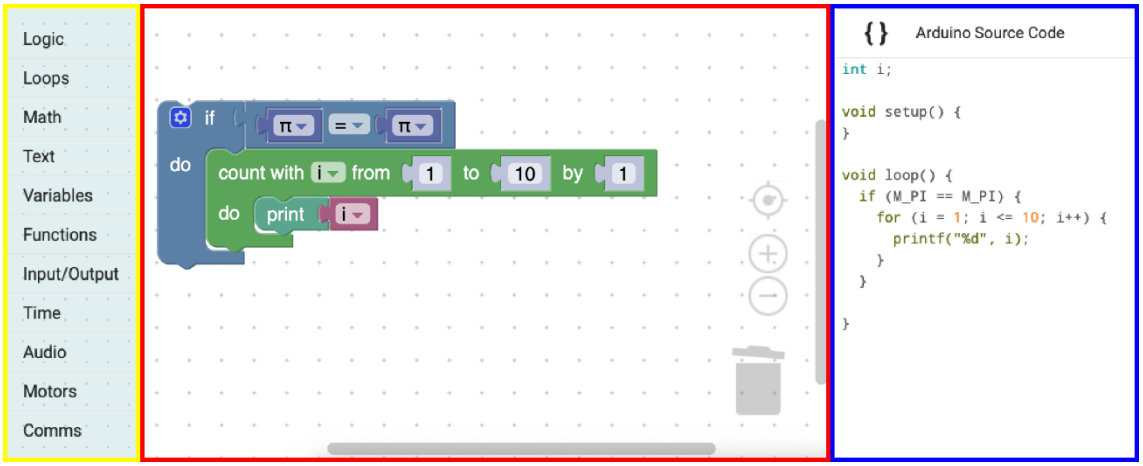
Google Blockly là một thư viện mã nguồn mở do Google phát triển, cung cấp các công cụ để xây dựng môi trường lập trình trực quan dưới dạng khối kéo thả. Thư viện này cho phép người dùng, đặc biệt là người mới bắt đầu, tạo chương trình bằng cách sắp xếp các khối lệnh đồ họa mà không cần viết mã văn bản, rất phù hợp cho giáo dục STEM và lập trình nhúng trên vi điều khiển Arduino [1]. Blockly không cung cấp giao diện cố định, mà bao gồm các thành phần linh hoạt để nhà phát triển tùy chỉnh theo nhu cầu. Các thành phần cốt lõi của Blockly bao gồm:

Bảng công cụ (Toolbox): Chứa danh sách các nhóm khối lệnh, mỗi nhóm khối lệnh chứa nhiều khối lệnh cùng chức năng. Người dùng kéo thả các khối lệnh từ Toolbox vào khu vực làm việc.

Khu vực làm việc (Workspace): Không gian nơi người dùng ghép nối các khối lệnh để tạo chương trình. Các khối được thiết kế với các đầu nối (connectors) đảm bảo tính hợp lệ về cú pháp, giảm thiểu lỗi lập trình.

Trình tạo mã (Code Generator): Chuyển đổi các khối lệnh thành mã nguồn của ngôn ngữ đích, thông qua các hàm ánh xạ được định nghĩa sẵn [10].

Nhóm tác giả phát triển môi trường lập trình kéo thả với giao diện đơn giản hóa cho học sinh, sinh viên. Hệ thống gồm 3 phần chính như Hình 1: Bảng công cụ (Toolbox) - khu vực màu vàng, khu vực làm việc (Workspace) - khu vực màu đỏ, trình tạo mã (Code Generator) - khu vực màu xanh.



Hình 1. Cấu trúc giao diện lập trình kéo thả.

Ngoài ra còn có thể bổ sung một số khối theo yêu cầu của người dùng để thuận tiện nhất cho lập trình.

2.2. Thiết kế các khối lệnh kéo thả

2.2.1. Xây dựng các nhóm khối lệnh trong Toolbox

Nhóm khối lệnh trong Toolbox là tập hợp các khối lệnh cùng chức năng, nhằm hỗ trợ người học tiếp cận lập trình cho vi mạch Arduino và điều khiển robot một cách hiệu quả và đơn giản nhất. Cách tiếp cận phân nhóm này được xây dựng dựa trên các nghiên cứu trước đây về thiết kế giao diện lập trình trực quan, chẳng hạn như công trình của Fraser, Moreno-León và Robles [1], [4], cũng như các nền tảng lập trình kéo thả phổ biến như mBlock và ArduBlockly [3], [4]. Những nền tảng này cũng áp dụng phương pháp phân loại khối lệnh theo chức năng nhằm tối ưu hóa trải nghiệm học tập và nâng cao hiệu quả.

Từ đó, nhóm tác giả chia các khối lệnh thành 11 nhóm khối lệnh chính như sau:

1. **Logic**: Phép so sánh, toán tử AND, OR, NOT...
2. **Loops**: Vòng lặp for, while...
3. **Math**: Phép toán đơn giản
4. **Text**: Xử lý chuỗi, biến kiểu chuỗi...
5. **Variables**: Khai báo, gán giá trị biến...
6. **Functions**: Định nghĩa hàm con
7. **Input/Output**: Đọc/ghi tín hiệu digital và analog
8. **Time**: Tạo độ trễ, lấy thời gian thực (delay(), millis())...

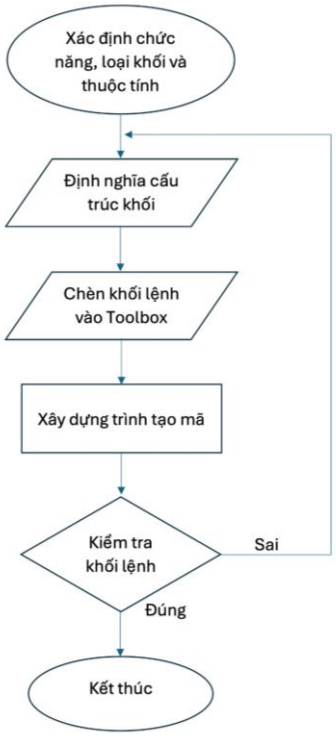
9. **Audio**: Xử lý tín hiệu âm thanh (liên quan gián tiếp đến I/O hoặc cảm biến)

10. **Motors**: Điều khiển servo, động cơ DC, stepper...

11. **Comms**: Giao tiếp (như Serial, liên quan đến I/O)...

2.2.2. Xây dựng khối lệnh trong từng nhóm khối lệnh tương ứng

Để xây dựng khối lệnh trong từng nhóm khối lệnh, nhóm tác giả đưa ra 5 bước chính thể hiện ở lưu đồ thuật toán trong Hình 2.



Hình 2. Quy trình xây dựng khối lệnh kéo thả

Bước 1: Xác định chức năng, loại khối và thuộc tính:

- Chức năng: Xác định hành vi của khối, ví dụ: thực hiện phép toán, điều khiển luồng chương trình...

- Loại khối: Statement block (khối lệnh hành động), value block (khối lệnh giá trị), hat block (khối lệnh khởi đầu).

- Quyết định các thuộc tính như kiểu dữ liệu đầu vào/đầu ra (số, Boolean), màu sắc và thông tin mô tả (tooltip).

Bước 2: Định nghĩa cấu trúc khối

Trong bước này, nhóm tác giả sử dụng mã nguồn mở Blockly để định nghĩa các khối lệnh bằng mã JavaScript. Các khối lệnh này sẽ được tích hợp vào nhóm khối lệnh

tương ứng trong Toolbox, nhằm phục vụ cho lập trình kéo-thả [1], [3]. Việc định nghĩa khối lệnh gồm các thành phần sau:

- **Khai báo khối**: Được định danh duy nhất trong Blockly.Blocks (Ví dụ: io_digitalread).

- **Khởi tạo giao diện (init)**: Thiết lập thuộc tính hiển thị như màu sắc (setColour) để phân biệt nhóm khối.

- **Cấu trúc đầu vào**:

  ● appendDummyInput: đầu vào tĩnh.

  ● appendValueInput: nhận giá trị từ khối khác.

  ● appendStatementInput: chứa khối lệnh con.

- **Liên kết khối**: setOutput, setPreviousStatement, setNextStatement: xác định khả năng kết nối giữa các khối.

- **Hỗ trợ người dùng**: setTooltip, setHelpUrl: cung cấp mô tả và tài liệu tham khảo.

- **Kiểu dữ liệu đầu ra**: getBlockType().

- **Thích ứng phần cứng**: updateFields() dùng để cập nhật giao diện khi có thay đổi về phần cứng.

Bước 3: Chèn khối lệnh vào Toolbox

Để hiển thị khối lệnh trong giao diện lập trình kéo thả, nhóm tác giả chèn mã XML tương ứng vào phần Toolbox. Mỗi khối lệnh cần được đặt trong một thẻ <block> và nhóm vào một thẻ <category> theo chức năng [1], [10].

Bước 4: Xây dựng trình tạo mã (Code Generator)

Mỗi khối lệnh sẽ được ánh xạ với một đoạn mã Arduino C/C++ tương ứng, được thực hiện thông qua các hàm Generator được viết bằng JavaScript [1], [4].

1. **Định nghĩa hàm**: Gán hàm JavaScript cho khối (ví dụ: Blockly.Arduino ['ten_khoi']) để xác định logic tạo mã cho khối cụ thể.

2. **Lấy đầu vào**: Truy xuất giá trị từ trường hoặc khối con (ví dụ: block.getFieldValue('PIN')) để lấy thông tin người dùng nhập, như số chân hoặc tham số.

3. **Tạo mã cài đặt**: Thêm mã khởi tạo (như pinMode) vào setup() bằng addSetup để cấu hình tài nguyên trước khi chương trình chạy.

4. **Tạo mã thực thi:** Trả về mã thực thi (chuỗi hoặc [code, ORDER_ <MỨC_ƯU_TIÊN>]) để thực hiện chức năng khối trong chương trình chính.

Bước 5: Kiểm thử khối lệnh trên cả phần mềm và phần cứng thực tế

Mục đích: Kiểm thử và xác minh tính chính xác, độ tin cậy của các khối lệnh được định nghĩa ở bước 2 khi hoạt động trên phần cứng thực tế (bo mạch vi điều khiển Arduino Uno).

Để kiểm thử khối lệnh vừa tạo ra, đầu tiên ta tạo chương trình đơn giản bằng cách kéo thả các khối lệnh vào khu vực làm việc, sau đó kiểm tra mã C/C++ được sinh ra và copy mã vào Arduino IDE, biên dịch và nạp vào thiết bị. Cuối cùng theo dõi hoạt động, debug và chỉnh sửa nếu cần [4].

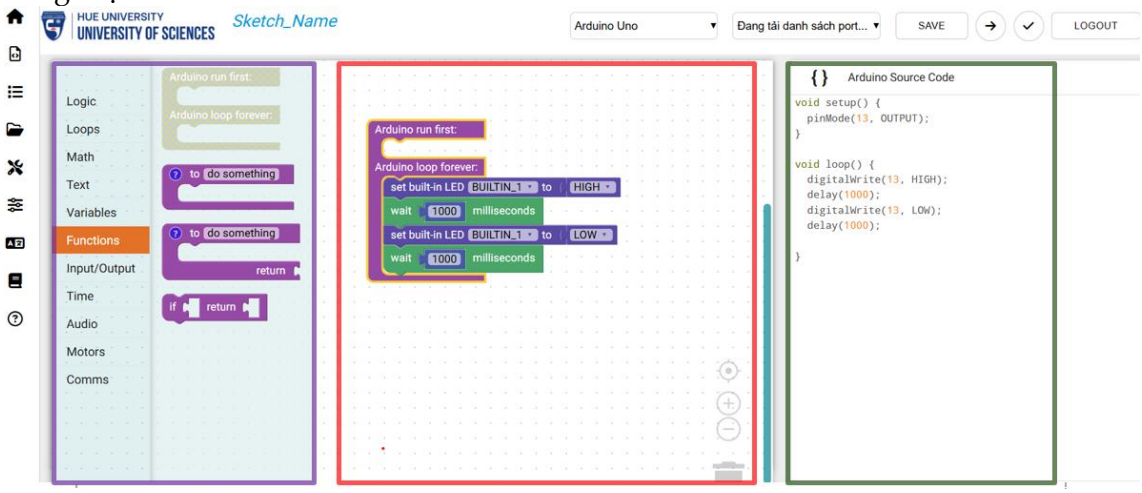
3. KẾT QUẢ

Để kiểm tra tính chính xác của các khối lệnh, nhóm tác giả thực thi thử nghiệm trên phần cứng (Arduino UNO) với quy trình gồm 4 bước như sau:

Bước 1: Kéo thả các khối lệnh để tạo chương trình hoàn chỉnh (Hình 3**).**

Người dùng truy cập bảng công cụ (Toolbox – khung tím) để lựa chọn và kéo các khối lệnh cần thiết, sau đó đưa các khối này vào không gian làm việc (Workspace – khung đỏ) để ghép nối và hình thành chương trình điều khiển Arduino. Trong quá trình kéo thả, mã Arduino tương ứng được hệ thống tự động sinh ra đồng thời trong khung xanh Arduino Source Code – khung xanh, thể hiện sự ánh xạ trực tiếp giữa từng khối lệnh và đoạn mã C/C++ tương ứng.

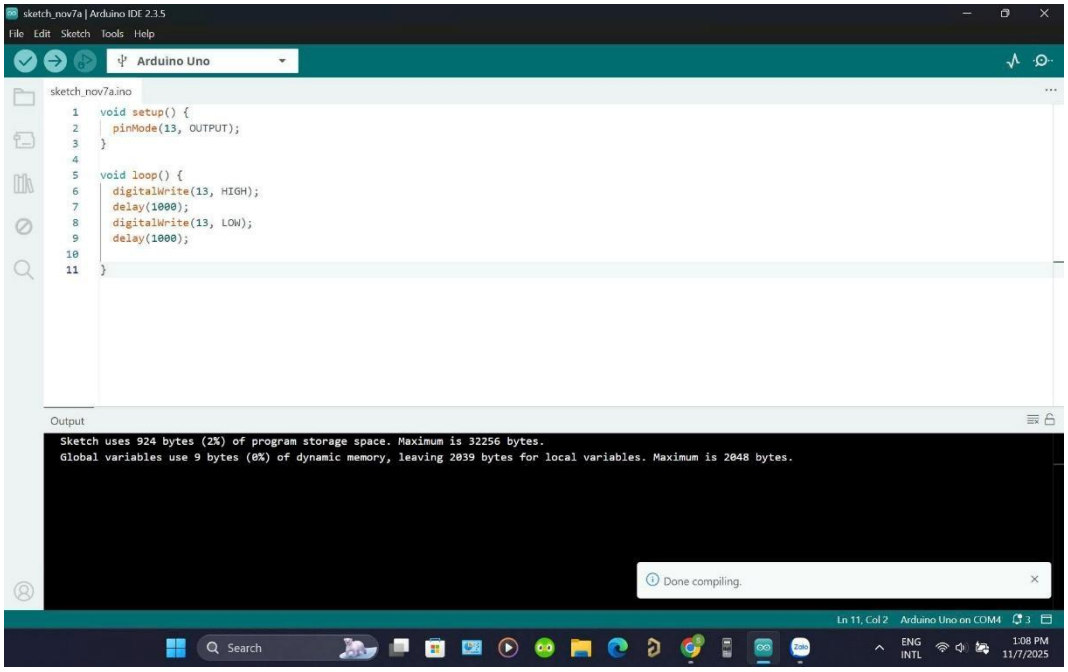
Trong ví dụ minh họa, chương trình gồm hai khối lệnh điều khiển LED tích hợp: khối thứ nhất bật LED trong 1 giây, khối thứ hai tắt LED trong 1 giây. Khi nạp chương trình vào bo mạch Arduino Uno, đèn LED tích hợp nhấp nháy với chu kỳ 1 giây, chứng minh quy trình sinh mã và hoạt động của các khối lệnh hoàn toàn chính xác trên phần cứng thực tế.



Hình 3. Giao diện Tooolbox (khung tím) – không gian ghép nối (khung đỏ) – phần sinh mã (khung xanh)

Bước 2: Kiểm tra cú pháp từ phần sinh mã tương ứng.

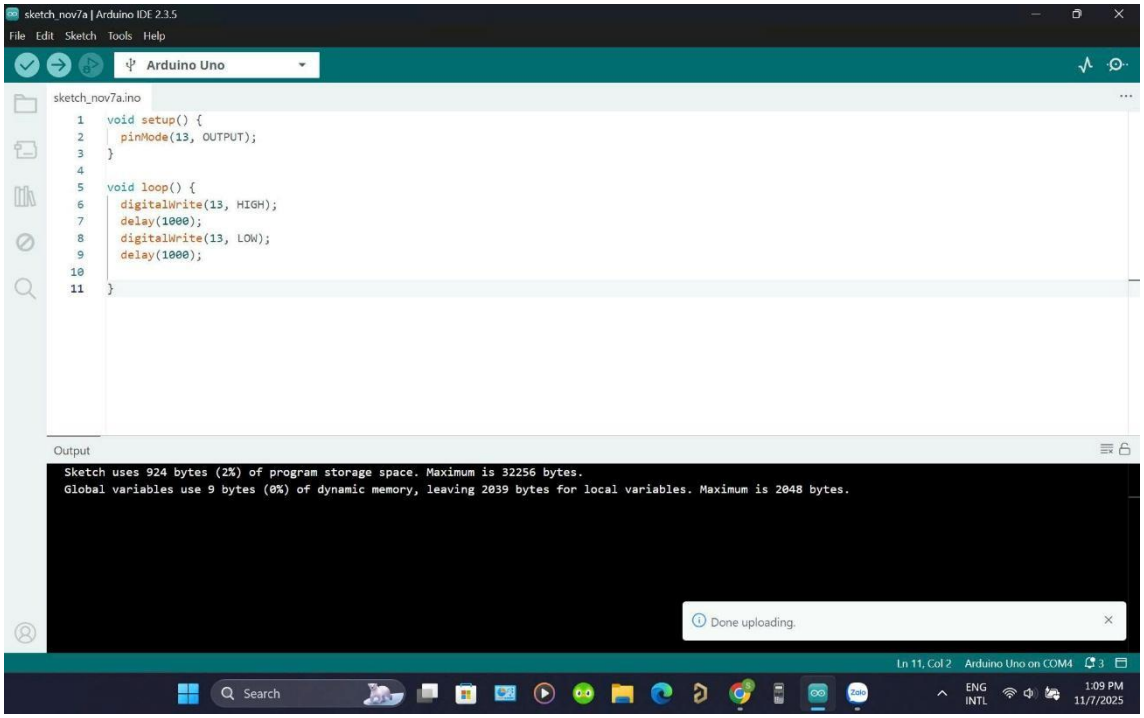
Xem mã C/C++ từ phần sinh mã ở Arduino Source Code – khung xanh, kiểm tra cú pháp bằng mắt hoặc Arduino IDE (chức năng Verify - dấu ✓) [4] (Hình 4).



Hình 4: Kiểm tra code được sinh ra trong khối Arduino Source Code

Bước 3: Copy mã C/C++ từ phần sinh mã và nạp vào IDE Arduino.

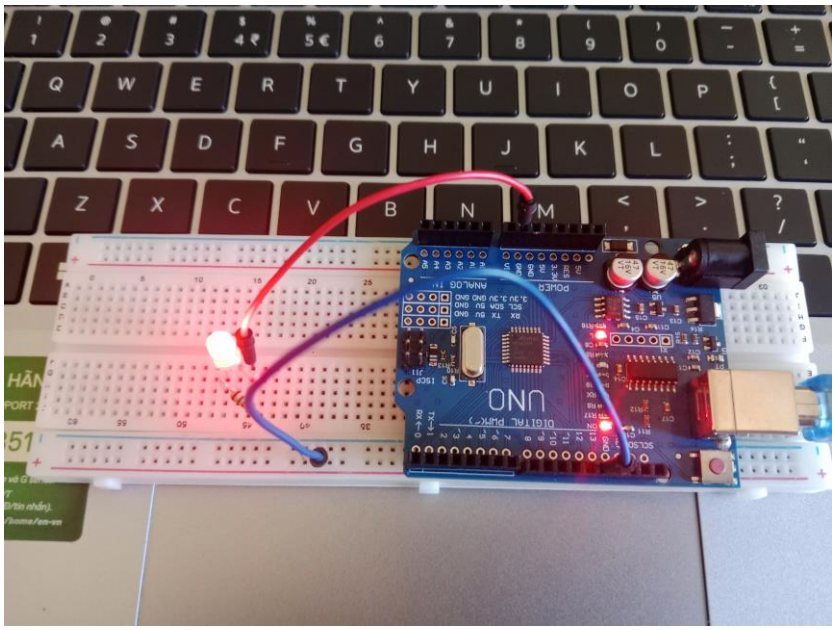
Sao chép mã C/C++ từ khối sinh mã, dán vào Arduino IDE, biên dịch và nạp lên vi điều khiển Arduino [1] Hình 5.



Hình 5: Nạp mã được sinh ra trong khối Arduino Source Code

Bước 4: Quan sát kết quả và hoàn thiện khối lệnh.

Quan sát hoạt động phần cứng (LED, động cơ, cảm biến), debug qua Serial Monitor, sửa lỗi nếu cần (Hình 6)



Hình 6: Quan sát hoạt động phần cứng

Nhóm cũng đã thử nghiệm nhiều chương trình như điều khiển servo, nút nhấn điều khiển LED và sử dụng vòng lặp for trên bo mạch Arduino Uno, tất cả đều cho kết quả chính xác. Độ tin cậy của hệ thống tiếp tục được khẳng định qua việc áp dụng thành công trong Cuộc thi Robocon HUSC 2025 (tháng 10/2025), nơi các đội thi sử dụng trực tiếp môi trường kéo thả để lập trình và điều khiển robot.

4. KẾT LUẬN

Nhóm tác giả đã thiết kế và phát triển hệ thống lập trình kéo thả cho Arduino tích hợp trực tuyến trên nền web, giúp người dùng dễ dàng truy cập và thao tác mà không cần cài đặt phần mềm. Hệ thống gồm 11 nhóm khối lệnh do nhóm tự xây dựng, có khả năng tùy biến cao, cho phép xây dựng và định nghĩa khối lệnh mới linh hoạt.

Các khối lệnh được sinh mã C/C++ chính xác, hoạt động ổn định trên phần cứng Arduino và đã được ứng dụng thành công trong Cuộc thi Robocon HUSC 2025, chứng minh tính khả thi, hiệu quả và độ tin cậy của thiết kế. Giải pháp này góp phần đơn giản hóa lập trình nhúng, hỗ trợ hiệu quả cho đào tạo và phát triển tư duy điều khiển trong giáo dục STEM.

Tuy nhiên, hệ thống còn hạn chế về số lượng khối lệnh cho các bài toán phức tạp và chưa tích hợp trình nạp tự động mã C/C++, gây bất tiện trong quá trình nạp code. Trong tương lai, nghiên cứu sẽ mở rộng thư viện khối lệnh và phát triển tính năng nạp tự động để nâng cao tính ứng dụng và hiệu quả đào tạo, góp phần thúc đẩy kỹ năng lập trình và tư duy sáng tạo trong giáo dục STEM.

TÀI LIỆU THAM KHẢO

[1]. Fraser, N. (2013). *Blockly*. Google Developers. https://developers.google.com/blockly

[2]. Citilab. (n.d.). *S4A – Scratch for Arduino*. http://s4a.cat/

[3]. Makeblock. (n.d.). *mBlock: Coding platform based on Scratch and Arduino*. https://mblock.makeblock.com/

[4]. Perate, C. (n.d.). *ArduBlockly: Arduino programming using visual blocks*. GitHub. https://github.com/carlosperate/ardublockly

[5]. D. Sun, C.-K. Looi, Y. Li, C. Zhu, C. Zhu, and M. Cheng,"Block-based versus text-based programming: a comparison of learners' programming behaviors, computational thinking skills and attitudes toward programming,"*Educational Technology Research and Development*, vol. 72, no. 1, pp. 1–24, Jan. 2024. doi: 10.1007/s11423-023-10328-8.

[6]. J. Fagerlund, P. Häkkinen, M. Vesisenaho, and J. Viiri,"Computational thinking in programming with Scratch in primary schools: A systematic review," *Computer Applications in Engineering Education*, vol. 28, no. 6, pp. 1–17, 2020. doi:10.1002/cae.22255.

[7]. Bak, N., Chang, B.-M., & Choi, K. (2020). Smart Block: A visual block language and its programming environment for IoT. *Journal of Computer Languages, 60*, 100999. https://doi.org/10.1016/j.cola.2020.100999

[8]. J. J. López and P. Lamo, "Rapid IoT Prototyping: A Visual Programming Tool and Hardware Solutions for LoRa-Based Devices," *Sensors*, vol. 23, no. 17, pp. 7511, 2023. doi: 10.3390/s23177511.

[9]. Ahmad, A., Idrees, M., Butt, M. A., & Danish, H. M. (2021). BBVPL: A block-based visual programming language built on Google's Blockly. *International Journal of Advanced Trends in Computer Science and Engineering, 10*(3), 2524–2532. https://doi.org/10.30534/ijatcse/2021/1441032021

[10]. Trower, J., & Gray, J. (2015, February). Creating new languages in Blockly. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education (SIGCSE '15)* (pp. 607–612). ACM. https://doi.org/10.1145/2676723.2691916

RESEARCH AND DEVELOPMENT OF DRAG-AND-DROP PROGRAMMING BLOCKS FOR THE ARDUINO PLATFORM BASED ON THE OPEN-SOURCE BLOCKLY FRAMEWORK

Nguyen Phan Nguyen Bao[1], Nguyen Van Huong[2], Nguyen Van Thai Bao[1], Nguyen Duc Nhat Quang[1], Tran Thi Kieu[1*]

[1] Faculty of Electronics, Electrical Engineering and Material Technology, University of Sciences, Hue University

[2] Faculty of Information Technology, University of Sciences, Hue University

*Email: kieutran@husc.edu.vn

ABSTRACT

This paper presents the design and development process of a drag-and-drop programming system for Arduino, built upon the open-source Google Blockly platform. The system is deployed online at https://robocon.husc.edu.vn, allowing users to create and generate C/C++ code compatible with Arduino directly on the web without requiring any software installation. The programming blocks are organized into 11 functional groups, which can be flexibly extended and customized by defining new blocks in JavaScript and XML, effectively supporting tasks related to control, communication, and signal processing in embedded programming. Experiments conducted on Arduino Uno and Nano demonstrate that the generated code maintains syntactic correctness and stable execution, confirming the accuracy and reliability of the proposed system. Compared to existing platforms such as ArduBlockly, mBlock, and CloverBlock—which are limited in customization capabilities and require local installation—the proposed system offers greater openness, scalability, and accessibility, making it more suitable for teaching, research, and STEM applications. Moreover, the system was successfully applied in the Robocon HUSC 2025 competition, where it proved to be stable, efficient, and highly effective in supporting robot programming and practical implementation.

Keywords: Arduino, Google Blockly, Drag-and-drop programming.