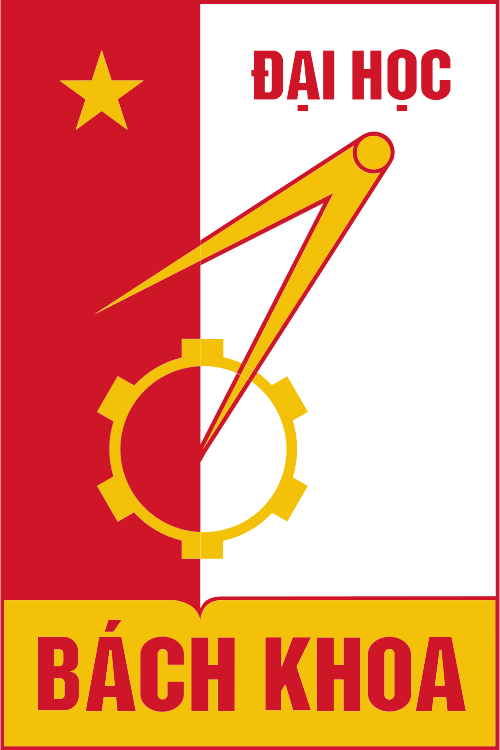
ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ───────

****

**BÀI TẬP LỚN**

HỌC PHẦN: HỆ NHÚNG

ĐỀ TÀI: GAME 2048

Nhóm : 8

Mã lớp học : 157539

Giáo viên hướng dẫn : TS. Ngô Lam Trung

Danh sách sinh viên thực hiện:

| Trần Xuân Phúc | 20225379 |
| --- | --- |
| Phan Lê Hải Đăng | 20225273 |
| Phạm Đăng Đạt | 20215563 |

***Hà Nội, tháng 6 năm 2025***

# **GIỚI THIỆU ĐỀ TÀI**

## **M**ô tả sản phẩm

Game 2048 là một game giải đố logic dựa trên các con số, người chơi sẽ di chuyển các ô trên một lưới ô vuông 4x4, các khối vuông sẽ trượt theo một trong bốn hướng tương ứng (lên, xuống, trái, phải). Mục tiêu của trò chơi là hợp nhất các ô có cùng giá trị, các ô sau khi gộp sẽ có giá trị tăng lên, và được tính điểm Nếu không còn ô nào di chuyển được sẽ tính là thua. Project tập trung phát triển một phiên bản game 2048 trên nền tảng vi điều khiển STM32F429ZIT6 kết hợp sử dụng hai thiết bị ngoại vi là module Joystick và module Speaker (Loa) giúp tăng khả năng tương tác và trải nghiệm người dùng (UX).

## **Mục tiêu** của **project**

Project “Game 2048” của nhóm chúng em hướng tới đạt được các mục tiêu cụ thể sau:

* Phát triển một phiên bản game 2048 hoạt động ổn định và đầy đủ chức năng trên nền tảng vi điều khiển STM32F429ZIT6. Điều này bao gồm việc triển khai chính xác thuật toán game, quản lý các trạng thái trò chơi (khởi tạo, di chuyển, hợp nhất, kết thúc), và đảm bảo tính ngẫu nhiên của việc xuất hiện các ô mới trên ma trận 4x4.
* Tích hợp thành công module Joystick để điều khiển và module âm thanh để cung cấp phản hồi làm tăng tính tương tác của trò chơi. Đảm bảo game hoạt động mượt mà với độ trễ thấp trên phần cứng giới hạn của vi điều khiển (RAM/ Flash, tài nguyên CPU), bao gồm việc xử lý tín hiệu điều khiển từ Joystick và phát âm thanh đồng thời.
* Thiết kế và triển khai giao diện đồ họa (UI) trực quan trên màn hình LCD tích hợp trên vi điều khiển STM32F429ZIT6, hiển thị rõ ràng rõ ràng trạng thái màn chơi, điểm số và các thông báo cần thiết. Đồng thời, đảm bảo trải nghiệm người dùng thân thiện thông qua việc điều khiển dễ dàng bằng Joystick và cung cấp phản hồi âm thanh phù hợp, trực quan.
* Nâng cao kỹ năng lập trình nhúng và làm việc với các thiết bị phần cứng. Thông qua quá trình phát triển project, sinh viên sẽ được củng cố và mở rộng kiến thức về lập trình C/C++ cho hệ thống nhúng thông qua nền tảng STM32Cube IDE, làm việc với các thiết bị ngoại vi của STM32F429ZIT6 (LCD, Joystick, Speaker), gỡ lỗi phần cứng/ phần mềm, và quản lý dự án phát triển một sản phẩm.
* Chứng minh khả năng vận dụng các kiến thức đã học về học phần Hệ nhúng, kiến trúc máy tính, xử lý tín hiệu, và các kỹ thuật lập trình nâng cao để giải quyết một bài toán thực tế trong lĩnh vực hệ nhúng.

## Phân tích yêu cầu

1.3.1. Yêu cầu chức năng

Các yêu cầu chứng năng xác định những gì hệ thống phải làm để đáp ứng mục tiêu của project. Đối với game 2048 trên STM32F429ZIT6, các yêu cầu chức năng chính bao gồm:

* Quản lý Logic Game:
* Khởi tạo Game: Hệ thống khởi tạo một bàn chơi là một lưới ô vuông có kích thước 4x4 trống khi bắt đầu game mới hoặc game được khởi động lại. Sau đó, hệ thống đặt ngẫu nhiên một ô số “2” trên bàn chơi khi khởi tạo
* Di chuyển Ô và Hợp nhất: Hệ thống cho phép người chơi di chuyển tất cả các ô theo một trong bốn hướng: Lên, Xuống, Trái, Phải khi nhận được lệnh điều khiển từ module Joystick. Khi di chuyển, các ô cùng giá trị nằm kề nhau theo hướng di chuyển phải được hợp nhất thành một ô có giá trị gấp đôi. Các ô phải trượt đến vị trí xa nhất có thể theo hướng di chuyển cho đến khi gặp ô khác hoặc biên bàn chơi. Hệ thống phải tính điểm cho người chơi mỗi khi các ô được hợp nhất (ví dụ: hợp nhất 2+2=4 cộng 4 điểm). Sau mỗi lần di chuyển hợp lệ, hệ thống ngẫu nhiên đặt thêm một ô “2” hoặc “4” tại một vị trí còn trống trên bàn chơi.
* Điều kiện Thua: Hệ thống phải phát hiện khi không còn nước đi hợp lệ (không thể di chuyển hoặc hợp nhất bất kỳ ô nào) và hiển thị thông báo “You Lose”.
* Khởi động lại Game: Hệ thống cung cấp chức năng cho phép người chơi khởi động lại game từ đầu bất kỳ lúc nào.
* Giao diện Người dùng (UI) và Phản hồi:
* Hiển thị bàn chơi: Hệ thống phải hiển thị bàn chơi là một lưới ô vuông có kích thước 4x4 rõ ràng trên màn hình LCD của vi điều khiển STM32F429ZIT6. Mỗi ô trên bàn chơi phải hiển thị giá trị số của nó. Các ô có giá trị khác nhau phải được hiển thị với màu nền hoặc chữ khác nhau để dễ phân biệt.
* Hiển thị điểm số: Hệ thống phải hiển thị điểm số hiện tại của người chơi trên màn hình LCD. Điểm số được cập nhật ngay lập tức sau mỗi lần hợp nhất ô.
* Hiển thị thông báo: Hệ thống hiển thị các thông báo “You Lose” rõ ràng trên màn hình khi đạt điều kiện tương ứng. Hệ thống hiển thị giao diện sản chờ hoặc hướng dẫn ban đầu khi game mới được bật lên.
* Điều khiển và Tương tác (Joystick & LCD):
* Nhận tín hiệu từ Joystick: Hệ thống đọc được tín hiệu từ module Joystick để xác định hướng di chuyển (Lên, Xuống, Trái, Phải). Mỗi hướng di chuyển của Joystick phải tương ứng với một lệnh di chuyển ô trong game.
* Nhận tín hiệu cảm ứng từ màn hình LCD: Hệ thống đọc tín hiệu cảm ứng (nhấn nút) từ người dùng tương tác với màn hình, được sử dụng để lựa chọn bắt đầu game, khởi động lại game, xác nhận, và thoát game.
* Phản hồi âm thanh (Loa):
* Phát âm thanh khi Game kết thúc: Hệ thống phát ra hiệu ứng âm thanh huýt sáo chê bai khi đạt trạng thái “You Lose”
* Quản lý trạng thái:
* Lưu/ Tải trạng thái Game: Hệ thống có khả năng lưu lại trạng thái bàn chơi và điểm số hiện tại (lưu vào bộ nhớ Flash của vi điều khiển). Hệ thống có khả năng tải lại tháng thái game đã lưu khi khởi động lại hoặc theo yêu cầu của người chơi.

1.3.2. Yêu cầu phi chức năng

Các yêu cầu phi chức năng xác định những thuộc tính chất lượng của hệ thống, bao gồm hiệu suất, độ tin cậy, khả năng sử dụng, bảo mật, khả năng bảo trì và khả năng mở rộng. Đối với dự án này, các yêu cầu phi chức năng chính bao gồm:

* Yêu cầu về hiệu suất:
* Thời gian phản hồi điều khiển: Hệ thống phản hồi thao tác từ Joystick trong vòng không quá 100ms kể từ khi nhận được tín hiệu di chuyển hợp lệ. Điều này đảm bảo trải nghiệm chơi game mượt mà và không có độ trễ đáng kể.
* Tốc độ hiển thị: Màn hình LCD cập nhật trạng thái bàn chơi và điểm số trong vòng không quá 200ms sau mỗi lần di chuyển và hợp nhất các ô.
* Tần suất phát âm thanh: Các hiệu ứng âm thanh được phát ngay lập tức và đồng bộ với sự kiện tương ứng (thua), với độ trễ không đáng kể (<50ms)
* Tốc độ khởi tạo game: Hệ thống hoàn tất quá trình khởi tạo game (hiển thị sảnh chờ, bàn chơi) trong vòng <= 2 giây kể từ khi bật nguồn hoặc khởi động lại.
* Yêu cầu về độ tin cậy:
* Hoạt động ổn định: Hệ thống hoạt động liên tục mà không gặp lỗi treo hoặc reset đột ngột trong ít nhất 2 giờ chơi liên tục.
* Tính chính xác của Game Logic: Thuật toán game (di chuyển, hợp nhất, tạo ô mới, tính điểm, kiểm tra thắng/ thua) phải hoạt động chính xác 100% theo quy tắc game 2048.
* Yêu cầu về khả năng sử dụng
* Giao diện trực quan: Giao diện người dùng trên LCD rõ ràng, dễ hiểu, các ô số và thông bảo dễ đọc.
* Dễ dàng thao tác: Người dùng có thể điều khiển dễ dàng bằng Joystick, các thao tác di chuyển và khởi động lại game đơn giản và trực quan
* Phản hồi rõ ràng: Các phản hồi từ hệ thống (hình ảnh trên LCD, âm thanh) rõ ràng và gia
* Yêu cầu về khả năng bảo trì
* Mã nguồn dễ hiểu: Mã nguồn được viết rõ ràng, có cấu trúc tốt
* Khả năng cấu hình: Các thông số quan trọng của game (kích thước bàn chơi, tốc độ game) được định nghĩa rõ ràng, có thể dễ dàng cấu hình nếu cần
* Yêu cầu về khả năng mở rộng:
* Thiết kế module: Hệ thống phải được thiết kế theo module để cho phép dễ dàng thêm các tính năng mới (ví dụ: lưu điểm cao, bảng xếp hạng) mà không làm ảnh hưởng lớn đến các chức năng hiện có.
* Hỗ trợ thay đổi phần cứng: Kiến trúc phần mềm có sự tách biệt rõ ràng giữa logic game và lớp driver phần cứng để có thể dễ dàng port sang các vi điều khiển STM32 khác hoặc các module LCD/ Joystick tương tự
* Yêu cầu về tài nguyên:
* Sử dụng bộ nhớ Flash: Tổng kích thước firmware của game nằm trong giới hạn bộ nhớ Flash của STM32F429ZIT6 (~1MB). Tối ưu để không vượt quá 80% dung lượng khả dụng.
* Tối ưu CPU: Tải CPU sử dụng trong quá trình game hoạt động bình thường được giữ ở mức thấp để đảm bảo không bị giật lag và có thể dành tài nguyên cho các tác vụ nền khác.

# THIẾT KẾ HỆ THỐNG

## 2.1. Tổng quan kiến trúc hệ thống

Hệ thống game 2048 được xây dựng dựa trên kiến trúc nhúng truyền thống, nơi vi điều khiển STM32F429ZIT6 đóng vai trò trung tâm, điều khiển và quản lý tất cả các thiết bị ngoại vi và thực thi logic game.

* Lớp phần cứng: Bao gồm vi điều khiển STM32F429ZIT6 và các thiết bị ngoại vi được kết nối trực tiếp, chịu trách nhiệm xử lý các tác vụ vật lý và cung cấp dữ liệu thô cho phần mềm.
* Lớp phần mềm: Chạy trên vi điều khiển, chịu trách nhiệm xử lý logic game, quản lý giao diện người dùng, đọc dữ liệu từ phần cứng và điều khiển các thiết bị đầu ra.

## 2.2. Phân chia chức năng

Để đạt được yêu cầu chức năng và phi chức năng đã đề ra, các chức năng của project được phân chia rõ ràng như sau:

2.2.1. Chức năng thực hiện bằng Phần cứng

Các thành phần phần cứng đóng vai trò cung cấp khả năng tương tác vật lý và hiển thị, là nền tảng cho hoạt động của game.

* Vi điều khiển STM32F429ZIT6: Thực thi Core Logic, chịu trách nhiệm chính trong việc thực thi toàn bộ mã lệnh của game, từ logic tính toán đến quản lý I/O. Thêm vào đó, vi điều khiển còn quản lý lưu trữ mã chương trình, các dữ liệu tĩnh và động, trạng thái game thông qua bộ nhớ Flash và RAM. Cuối cùng, thực hiện giao tiếp ngoại vi với các chuẩn GPIO (General Purpose Input/Output), ADC (Analog-to-Digital Converter), TIM (Timer/Counter), SPI (Serial Peripheral Interface), và I2S (Inter-Integrated Circuit Sound)
* Màn hình LCD: Hiển thị đồ hoạ, nhận dữ liệu hình ảnh (pixel, màu sắc) từ vi điều khiển và chuyển đổi thành tín hiệu quang học để hiển thị bàn chơi, điểm số và các thông báo.
* Module Joystick: Tạo các tín hiệu điện áp analog trong các biến trở tương ứng với vị trí trục X, Y.
* Module Âm thanh (Speaker với module khuếch đại âm thanh MAX98357A I2S): Nhận tín hiệu từ vi điều khiển và chuyển đổi chúng thành sóng âm thanh tương ứng với tần số và cường độ để tạo ra các hiệu ứng âm thanh.

2.2.2. Chức năng thực hiện bằng Phần mềm

Phần mềm đóng vai trò xử lý tất cả các logic nghiệp vụ và quản lý tương tác với phần cứng. Các module phần mềm chính bao gồm:

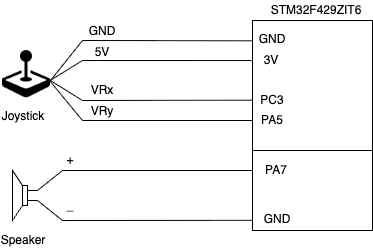
* Tầng Drivers phần cứng
* Module Game Logic
* Module quản lý giao diện người dùng (UI)
* Module quản lý âm thanh
* Module quản lý sự kiện/Input
* Module quản lý bộ nhớ/ lưu trữ

## 2.3. Thiết kế chi tiết

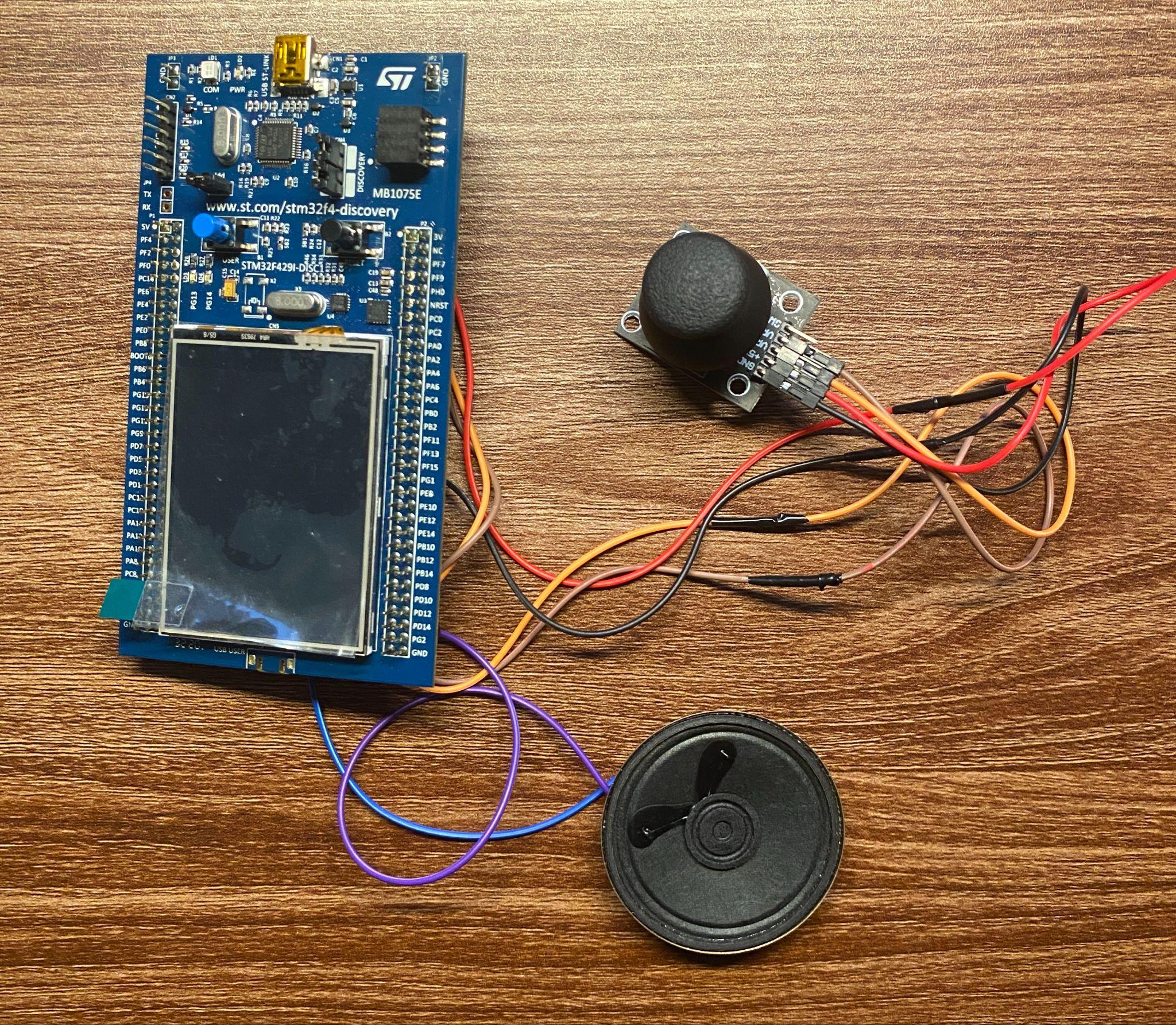
## 

2.3.1. Thiết kế phần cứng

* Module/ Linh kiện:
* Joystick:
* Nguồn cấp: Cung cấp nguồn (5V) và GND cho module Joystick.
* Kết nối Analog (Trục X, Y): Hai chân đầu ra analog của Joystick (VRx, VRy) sẽ được kết nối đến hai chân ADC (Analog-to-Digital Converter) của STM32F429ZIT6 (PC3, PA5)
* Module Speaker:
* Nguồn cấp: Cung cấp nguồn (2V) và GND
* Chân kết nối với chân PA7 của vi điều khiển được cấu hình (TIM3/ Channel 2) để xuất ra tín hiệu PWM
* Sơ đồ mạch:



* Sơ đồ ghép nối:

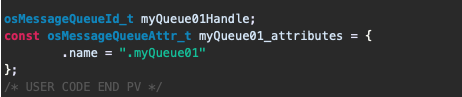


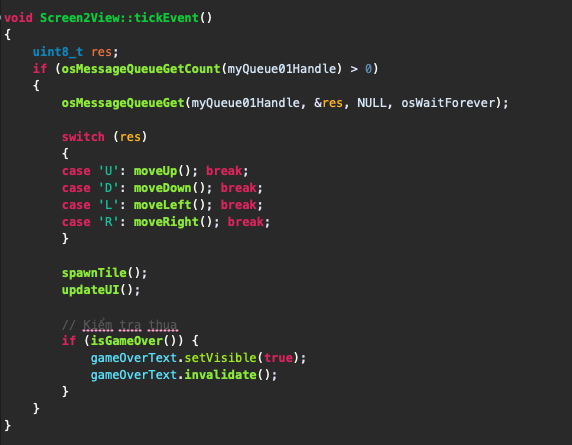
2.3.2. Thiết kế phần mềm

* Xử lý Input từ Joystick:
* Trong file main.c, hàm StartDefaultTask() sẽ liên tục polling tín hiệu từ Joystick sau đó gọi hàm getJoystickDirection() chuyển đổi từ số nguyên sang các kí tự “R”, “L”, “D”, “U” sau đó gửi vào hàng đợi myQueue01Handle.
* Trong file Screen2View.cpp, hàm tickEvent() lấy dữ liệu của myQueue01Handle từ file main.c sau đó xử lý sự kiện di chuyển và update màn hình. Các sự kiện gồm lên, xuống, trái, phải, cập nhật màn hình, và hiển thị khi Game Over. Lần lượt gọi các hàm moveUp(), moveDown(), moveLeft(), moveRight(), spawnTile(), updateUI(), isGameOver()

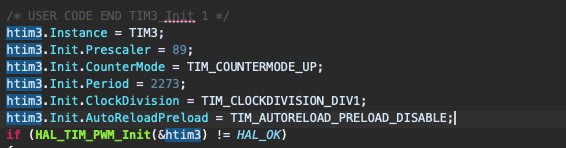


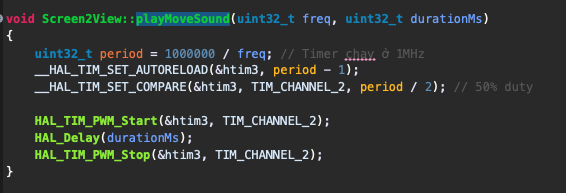






* Quản lý phát âm thanh bằng module Speaker:
* Ở file screen2View extern htim3 từ file main.c để tạo xung âm thanh, hàm playMoveSound tạo âm thanh dạng sóng Tạo ra âm thanh dạng sóng vuông có tần số freq (Hz) và kéo dài trong durationMs mili giây, sử dụng PWM (Pulse Width Modulation) của timer TIM3 để điều khiển một loa buzzer hoặc buzzer thụ động được kết nối với kit.

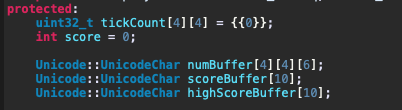




* Module Lưu trạng thái:
* Hàm saveHighScoreToFlash(): Lưu giá trị điểm cao nhất vào bộ nhớ Flash của vi điều khiển. Đầu tiên, mở khoá bộ nhớ Flash để chuẩn bị dữ liệu. Sau đó, ghi vào Flash rồi khoá Flash lại.
* Hàm loadHighScoreFromFlash(): Lấy giá trị highScore từ bộ nhớ Flash đã lưu từ trước đó.

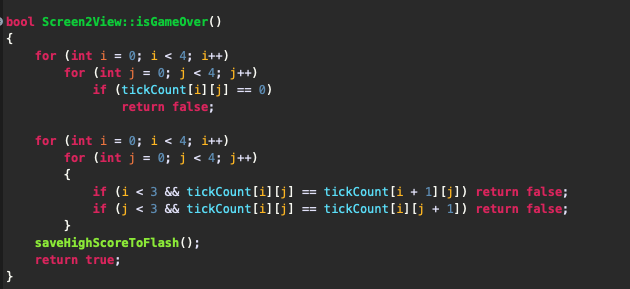


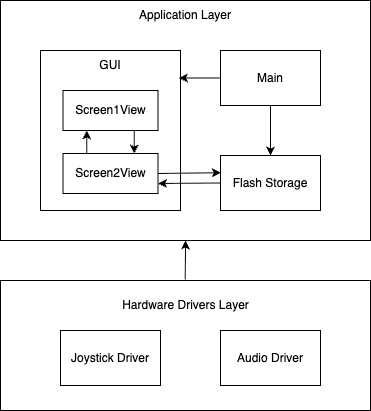
* Logic Game:
* Khởi tạo các trạng thái hiển thị và logic trò chơi:
* uint32\_t tickCount[4][4] = {{0}}: Ma trận logic để lưu trạng thái game. Dùng để lưu thời điểm (tick) của mỗi ô trong lưới 4x4 (mỗi ô số của 2048)
* int score = 0: Lưu điểm hiện tại của trò chơi
* Unicode::UnicodeChar numBuffer[4][4][6]: Lưu chuỗi ký tự đại diện cho số của từng ô (file) trong bàn chơi
* Unicode::UnicodeChar scoreBuffer[10]: Chuỗi Unicode để hiển thị điểm hiện tại
* Unicode::UnicodeChar highScoreBuffer[10]: Tương tự như scoreBuffer nhưng dùng cho điểm cao nhất



* Sinh ngẫu nhiên ô vuông: Hàm spawnTile(): Sử dụng hàm HAL\_GetTick() lấy thời gian từ hệ thống để tạo random vị trí xuất hiện và giá trị xuất hiện.



* Các hàm moveUp(), moveDown(), moveLeft(), moveRight() thay đổi mảng tickCount[4][4]
* Xử lý sự kiện Game Over: sử dụng hàm isGameOver() kiểm tra xem trên bàn cờ còn tạo được ô vuông ngẫu nhiên không dựa vào việc kiểm tra lần lượt các giá trị tickCount[i][j] trên bàn chơi. Nếu đúng sẽ gọi hàm saveHighScoreToFlash() để lưu điểm hiện tại là điểm cao nếu hợp lệ 
* Sơ đồ khối:



# CÀI ĐẶT HỆ THỐNG

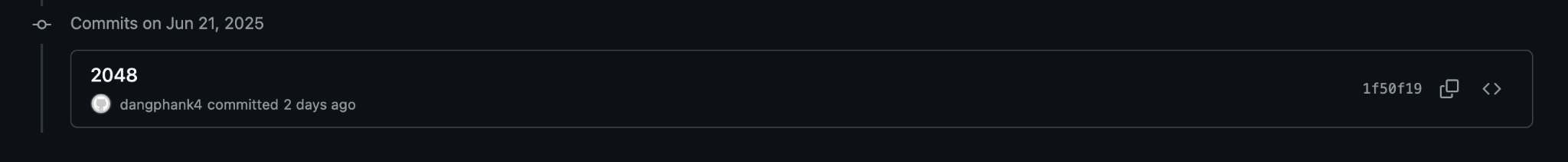
## 

Link Github Project: <https://github.com/dangphank4/game_2048.git>

## 3.1. Phân chia công việc

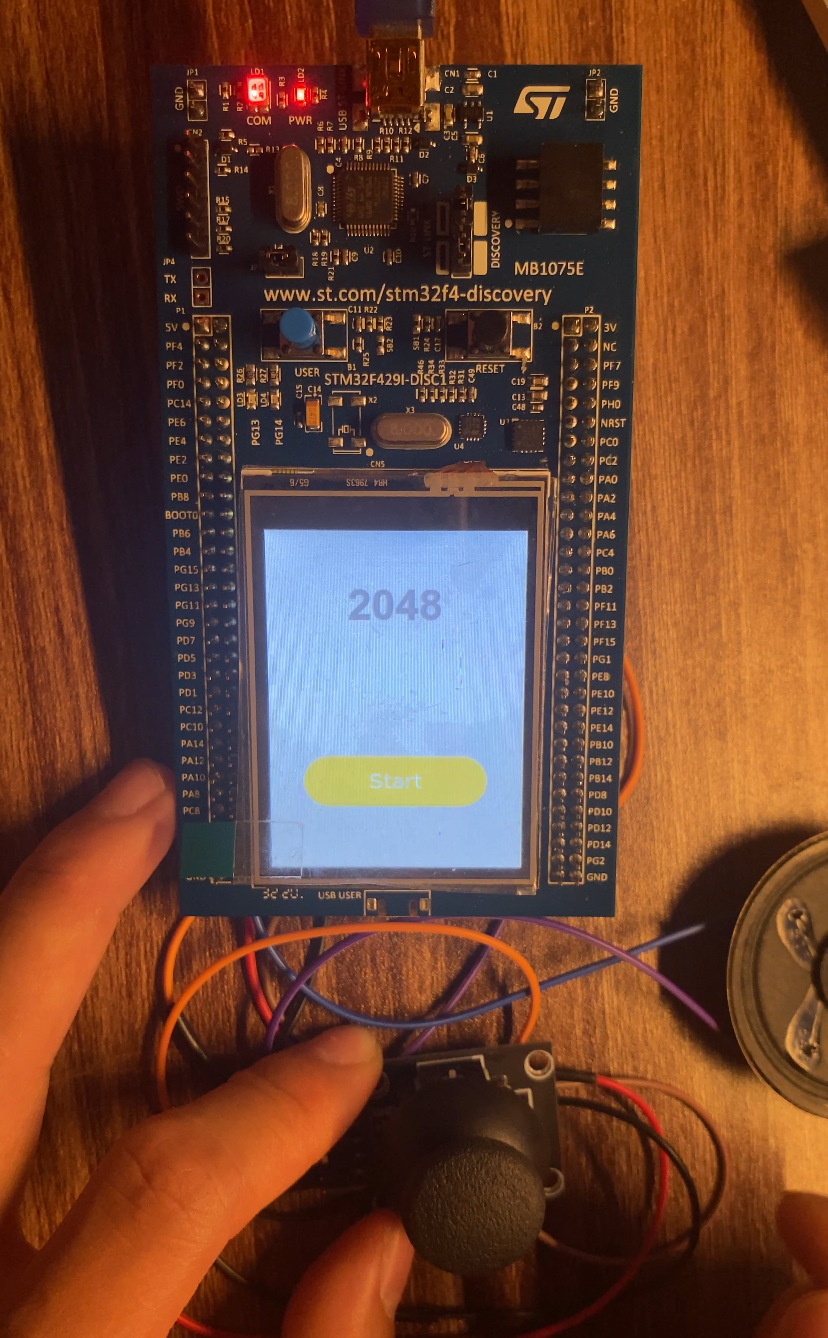
| **Họ và tên** | **Nhiệm vụ** | **Đóng góp** |
| --- | --- | --- |
| Trần Xuân Phúc | * Viết báo cáo * Thiết kế giao diện trên TouchGFX * Thiết kế kịch bản trò chơi * Xây dựng logic trò chơi | 33,33% |
| Phan Lê Hải Đăng | * Viết báo cáo * Xây dựng module Joystick * Xây dựng logic trò chơi + điểm | 33,33% |
| Phạm Đăng Đạt | * Viết báo cáo * Xây dựng module Speaker * Xây dựng chức năng lưu trạng thái khi chơi và điểm cao | 33,33% |

* Lịch sử commit của từng thành viên:  
  

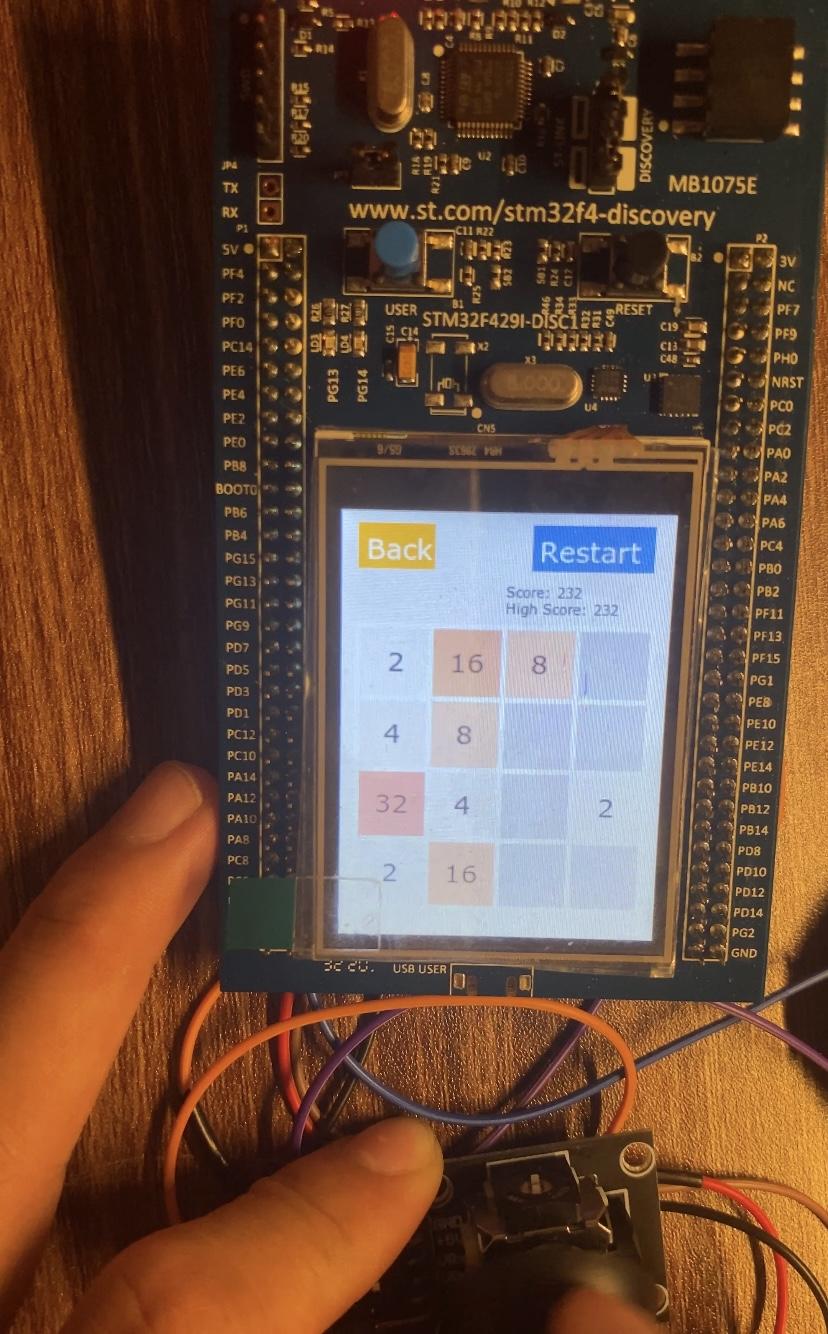


## 3.2. Hình ảnh & Video demo dự án

* Link video: [Project Nhúng](https://drive.google.com/drive/folders/1KH0EtIzelvZy90hKnZjd_fQvRbbQqKLi?usp=sharing)
* Một số hình ảnh demo:



*Hình 1: Màn hình chính*



*Hình 2: Màn hình chơi*