

**ĐẠI HỌC BÁCH KHOA HÀ NỘI  
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**BÁO CÁO ĐỒ ÁN MÔN HỌC**

**Đề tài: Xây dựng game Battle Ship  
sử dụng KIT STM32F429I-DISC1**

Lớp : 141341  
Học phần : Hệ nhúng  
Mã học phần : IT4210  
Giảng viên hướng dẫn : TS. Đỗ Công Thuần

Danh sách thành viên nhóm 40:

Họ và tên	Mã số sinh viên	Email
Vũ Đình Tiên	20194384	<a href="mailto:tien.vs194384@sis.hust.edu.vn">tien.vs194384@sis.hust.edu.vn</a>
Trương Văn Hiển	20194276	<a href="mailto:hien.tv194276@sis.hust.edu.vn">hien.tv194276@sis.hust.edu.vn</a>
Nguyễn Hữu Huân	20194288	<a href="mailto:huan.nh194288@sis.hust.edu.vn">huan.nh194288@sis.hust.edu.vn</a>

*Hà Nội, tháng 8 năm 2023*

## MỤC LỤC

LỜI NÓI ĐẦU .....	3
CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI .....	5
1.1. Lý do chọn đề tài .....	5
1.2. Mục tiêu .....	6
1.3. Ý tưởng thực hiện .....	7
1.4. Phương pháp nghiên cứu .....	7
CHƯƠNG 2: HỆ THỐNG PHẦN CỨNG .....	8
2.1. Kiến trúc hệ thống .....	8
2.2. Tổng quan về Kit STM32F429I-DISC1 .....	8
2.2.1. Phần cứng của Kit .....	9
2.2.2. Tính năng, giao thức kết nối của Kit .....	10
2.2.3. Vị trí các thành phần trên Kit và kích thước .....	10
2.3. Mạch nút bấm .....	12
CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH VÀ TRIỂN KHAI CÀI ĐẶT .....	13
3.1. Sơ đồ hệ thống .....	13
3.2. Luồng hệ thống .....	14
3.2.1. Nhận dữ liệu từ tác nhân người chơi .....	14
3.2.2. Xử lý logic của game và xây dựng giao diện .....	16
CHƯƠNG 4: ĐÁNH GIÁ TỔNG QUAN .....	18
4.1. Sản phẩm demo .....	18
4.2. Đánh giá hệ thống .....	19
4.3. Hướng phát triển .....	20
PHỤ LỤC .....	21
1. Phân công nhiệm vụ .....	21
2. Tài liệu tham khảo .....	22
3. Mã nguồn (source code) và các resource của đồ án .....	22

## LỜI NÓI ĐẦU

Công nghệ thông tin ngày càng phát triển và có vai trò hết sức quan trọng không thể thiếu trong cuộc sống hiện đại, nó đã thay đổi mọi khía cạnh của cuộc sống, từ việc giao tiếp, giải trí, đến học tập và công việc. Chúng ta sống trong một thế giới với hàng tỷ thiết bị kết nối với Internet, tạo nên mạng lưới thông tin phức tạp. Nhưng không chỉ có những thiết bị lớn như máy tính cá nhân và điện thoại thông minh, công nghệ thông tin còn hiện diện ẩn mình trong những hệ thống nhỏ gọn và tích hợp, được gọi là hệ thống nhúng.

Hệ thống nhúng là những thiết bị và hệ thống tích hợp các vi xử lý, cảm biến, và giao tiếp mạng. Điển hình cho các ứng dụng của hệ thống nhúng là các thiết bị di động, các thiết bị y tế thông minh, đồ gia dụng kết nối Internet (Internet of Things - IoT) và nhiều ứng dụng khác.

Sự phát triển của công nghệ đã thúc đẩy sự tiến bộ của hệ nhúng. Các bộ vi xử lý ngày càng mạnh mẽ và nhỏ gọn, công nghệ bộ nhớ và lưu trữ phát triển nhanh chóng, đồng thời với đó là xu hướng phần mềm mã nguồn mở và các công cụ phát triển hệ thống nhúng. Tất cả những tiến bộ này đã tạo điều kiện thuận lợi cho việc phát triển các ứng dụng hệ thống nhúng phức tạp và đa dạng.

Với sự hấp dẫn của lĩnh vực và những thách thức còn đang ở phía trước, với niềm đam mê, mong muốn được học hỏi các công nghệ, tiếp xúc với các bài toán của Hệ nhúng, nhóm chúng em đã quyết định lựa chọn đề tài “**Xây dựng game Battle Ship sử dụng KIT STM32F429I-DISC1**” cho Đồ án môn học của mình.

Đồ án môn học của nhóm chúng em bao gồm 4 nội dung chính:

1. Tổng quan đề tài
2. Hệ thống phân cứng
3. Xây dựng chương trình và triển khai cài đặt
4. Đánh giá tổng quan

*Đồ án môn học: Hệ nhúng – IT4210*

Mặc dù đã cố gắng hoàn thiện sản phẩm nhưng không thể tránh khỏi những thiếu hụt về kiến thức và sai sót trong kiểm thử. Chúng em rất mong nhận được những nhận xét thẳng thắn, chi tiết đến từ thầy để tiếp tục hoàn thiện hơn nữa. Cuối cùng, nhóm chúng em xin được gửi lời cảm ơn đến thầy **TS. Đỗ Công Thuần** đã hướng dẫn chúng em trong suốt quá trình hoàn thiện Đồ án môn học. Nhóm chúng em xin chân thành cảm ơn thầy.

## CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

### 1.1. Lý do chọn đề tài

Các thiết bị điều khiển và các hệ thống nhúng ngày một nhiều và trở nên phổ biến quanh cuộc sống thường ngày với nền văn minh hiện đại. Thực tế mỗi khía cạnh của hoạt động hằng ngày đều bị chi phối bởi một vài loại hệ thống điều khiển. Đề dàng tìm thấy các thiết bị điều khiển cho các máy móc, công cụ, điều khiển máy tính, các hệ thống giao thông, hệ thống năng lượng robot,... rồi các ứng dụng vĩ mô hơn như công nghệ kỹ thuật liên quan đến vũ khí công nghệ cao, ngành hàng không vũ trụ, hay đơn giản là các ví dụ về các phương tiện đi lại, đồ gia dụng, dân dụng mỗi người sử dụng hằng ngày.

Việc xây dựng hệ thống nhúng là một thách thức thú vị và có ý nghĩa rất lớn trong việc nâng cao kỹ năng và hiểu sâu hơn về cách hoạt động của các thiết bị điện tử thông minh. Trong Đồ án học phần này, nhóm chúng em quyết định lựa chọn đề tài “Xây dựng game Battle Ship sử dụng KIT STM32F429I-DISC1” với lý do sau:

- Battle Ship: trò chơi kinh điển được nhiều người yêu thích. Việc xây dựng phiên bản chơi game này trên một hệ thống nhúng với hi vọng sẽ mang đến thử thách và sự mới lạ cho nhóm chúng em trong quá trình nghiên cứu và hoàn thiện Đồ án.
- Ứng dụng thực tiễn: KIT STM32F429I-DISC1 là 1 board phổ biến và mạnh mẽ được sử dụng rộng rãi trong việc phát triển các hệ thống nhúng. Xây dựng trò chơi Battle Ship trên nền tảng này có thể sẽ giúp chúng em hiểu rõ hơn về cách làm việc với kit phần cứng và tận dụng tối đa tiềm năng của nó.
- Học hỏi và phát triển kỹ năng: Đề tài này đòi hỏi nhóm chúng em phải áp dụng kiến thức về vi điều khiển, cấu trúc dữ liệu, giao tiếp ngoại vi và lập trình nhúng. Qua việc học tập trên lớp Lý thuyết và làm việc tại lớp Thực hành, chúng em có cơ hội rèn luyện và nâng cao kỹ năng lập trình phần cứng cũng như phát triển tư duy logic và sáng tạo.

- Xây dựng sản phẩm cuối cùng: Mục tiêu cuối cùng của đề tài này là tạo ra một trò chơi Battle Ship hoàn chỉnh trên hệ thống nhúng, có thể hoạt động ổn định và trải nghiệm được. Điều này sẽ giúp chúng em hiểu quy trình phát triển phần mềm nhúng và chuẩn bị cho việc thực hiện các dự án thực tế trong tương lai.

Với những lý do trên, trong khuôn khổ Đồ án môn học, nhóm em với đề tài trên rất mong muốn triển khai một hệ thống nhúng sẽ mang lại những trải nghiệm học tập và thực hành đáng giá, đồng thời góp phần thúc đẩy sự phát triển và ứng dụng của công nghệ nhúng trong cuộc sống hàng ngày.

## 1.2. Mục tiêu

Trong quá trình học tập học phần Hệ nhúng, nhận thấy bối cảnh và những nhu cầu kể trên, nhóm chúng em đặt ra mục tiêu phải xây dựng được sản phẩm với các tiêu chí:

1. Ứng dụng được kiến thức về xây dựng các hệ thống nhúng đơn giản.
2. Phải là một sản phẩm được xem là “thông minh”.
3. Xây dựng giao diện người dùng (User Interface - UI): Phát triển giao diện người dùng trực quan và thân thiện để người dùng có thể tương tác và chơi trò chơi Battle Ship một cách dễ dàng.
4. Xử lý luật chơi: Xác định và cài đặt các luật chơi của trò chơi Battle Ship.
5. Điều khiển ngoại vi: Sử dụng KIT STM32F429I-DISC1 để điều khiển các ngoại vi như màn hình hiển thị, bàn phím, và các thiết bị ngoại vi khác cần thiết cho trò chơi. Thông qua việc tương tác với các ngoại vi này, người chơi có thể thấy và thao tác với trò chơi một cách trực quan và linh hoạt.
6. Kiểm thử và sửa lỗi: Tiến hành kiểm thử chéo để đảm bảo tính đúng đắn và ổn định của trò chơi. Nếu phát hiện lỗi, tiến hành sửa chúng và kiểm tra lại để đảm bảo hoạt động chính xác.

### **1.3. Ý tưởng thực hiện**

Trò chơi Battle Ship là trò chơi dạng shoot-em-up, trong đó người chơi điều khiển nhân vật di chuyển để tránh hỏa lực của kẻ thù và tiêu diệt các mục tiêu.

Trò chơi sẽ được xây dựng trên kit STM32F429I-DISC1 kết hợp với thư viện đồ họa TouchGFX. Bộ 32F429IDISCOVERY tận dụng khả năng của các vi điều khiển hiệu suất cao STM32F429, cho phép người dùng dễ dàng phát triển các ứng dụng phong phú với giao diện người dùng đồ họa tiên tiến.

### **1.4. Phương pháp nghiên cứu**

- Đi từ những kiến thức cơ bản về board mạch, cách vận hành cấu hình đến lập trình và nhúng.
- Sử dụng STM32CubeIDE để code, debug và biên dịch.
- Chuẩn bị kit STM32F429I-DISC1, các linh kiện cần thiết.
- Tìm hiểu về Kit STM32F429I-DISC1, các linh kiện, thiết bị ngoại vi thông qua các datasheet của Kit và các tài liệu liên quan trên Internet và Youtube.

## CHƯƠNG 2: HỆ THỐNG PHẦN CỨNG

### 2.1. Kiến trúc hệ thống

Hệ thống gồm 4 thành phần chính:

1. Kit STM32F429I-DISC1
2. Mini USB-cable
3. Mạch nút bấm
4. Dây nối

### 2.2. Tổng quan về Kit STM32F429I-DISC1



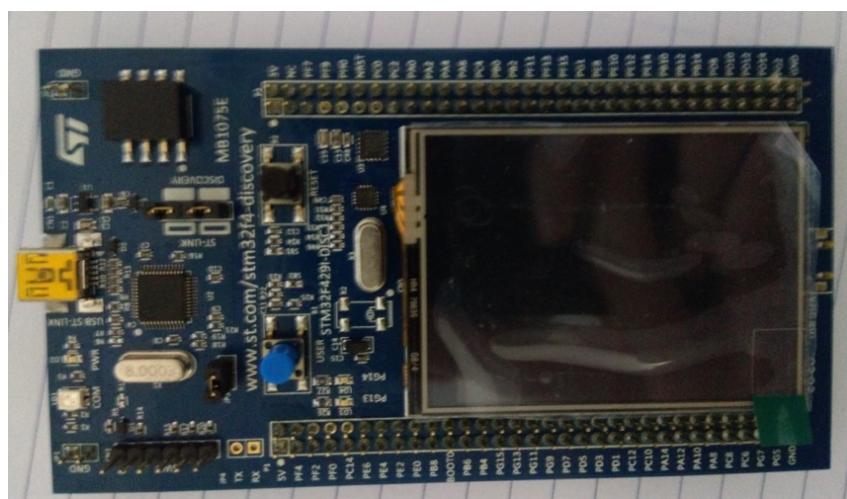
Hình 1: Mặt trước và mặt sau của Kit

Bộ STM32F429I-DISC1 cho phép người dùng dễ dàng phát triển các ứng dụng với MCU hiệu suất cao STM32F429 với lõi ARM® Cortex®-M4.

Kit bao gồm một công cụ gỡ lõi nhúng ST-LINK / V2-B, màn hình LCD TFT QVGA 2,4 ", hỗ trợ SDRAM 64 Mbit, con quay hồi chuyển ST MEMS hỗ trợ trong việc nhận diện góc quay, đầu nối micro-AB USB OTG, đèn LED và nút ấn.

### 2.2.1. Phần cứng của Kit

- Kit có tích hợp Micro Vi điều khiển (Micro Controller Unit – MCU) là đơn vị xử lý nhỏ, nó được tích hợp toàn bộ các bộ nhớ như ROM, RAM, các Port truy xuất, giao tiếp ngoại vi trực tiếp trên một con chip hết sức nhỏ gọn.
- Màn hình QVGA TFT LCD 2.4" hiển thị màu.
- 6 Led bao gồm:
  - o LD1 (đỏ/xanh) cho giao tiếp USB
  - o LD2 (đỏ) dành cho cấp nguồn 3.3V
  - o 2 Led người dùng: LD3 (xanh) và LD4 (đỏ)
  - o 2 Led USB OTG: LD5 (xanh) VBUS và LD6 (đỏ) OC
- 2 nút bấm: User và Reset
- Cổng kết nối USB OTG với chuẩn micro-AB
- Cảm biến con quay hồi quy xoay 3 trục xyz
- Cổng USB cấp nguồn 3V hoặc 5V



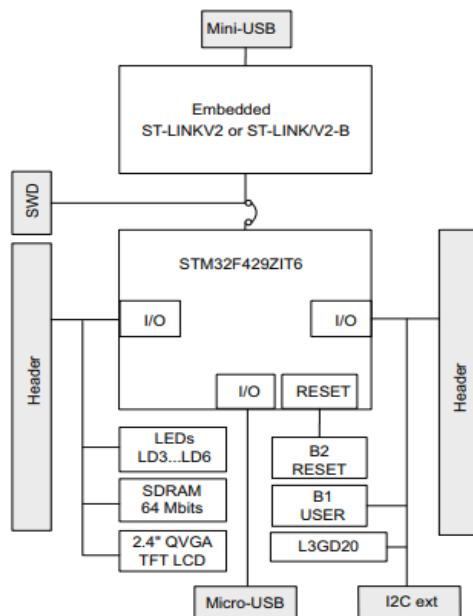
Hình 2: KIT STM32F429I-DISC1

### 2.2.2. Tính năng, giao thức kết nối của Kit

- Tính năng tích hợp SD-Link/V2 hỗ trợ 2 chế độ nạp code chạy và debug
- Tính năng mbed-enabled : nhằm đảm bảo các nhà phát triển có thể sử dụng các công nghệ Mbed (Mbed OS, Mbed Linux và các công cụ khác) trên một loạt các phần cứng đủ điều kiện.
- Tính năng cảm biến con quay hồi quy xoay Kit theo 3 trục xyz
- Tích hợp màn hình Led có hỗ trợ cảm ứng
- Kết nối USB có các chức năng chính:
  - o Cổng debug
  - o Cổng COM ảo
  - o Lưu trữ Mass
  - o Nạp nguồn cho Kit
  - o Giao tiếp USB OTG chuẩn micro-AB

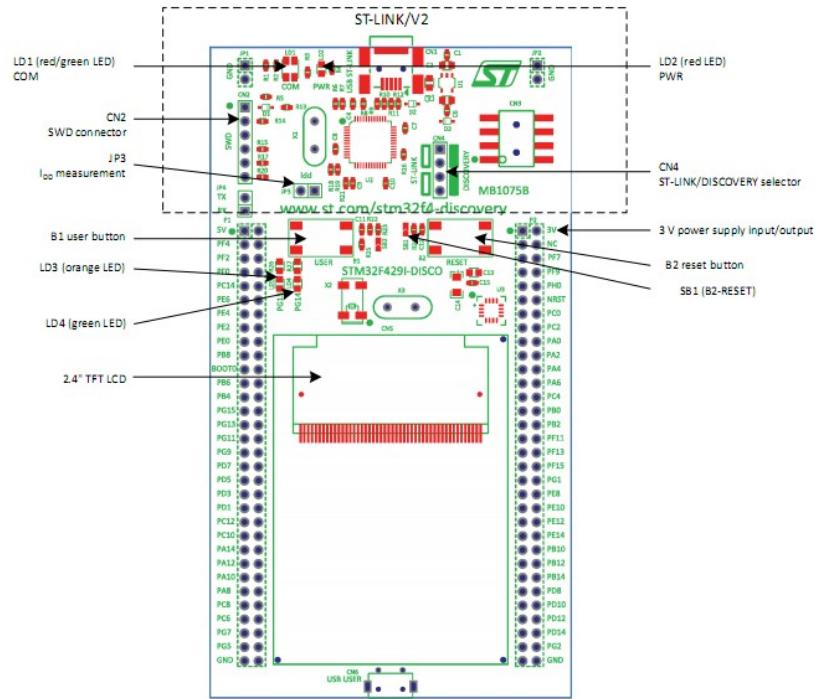
### 2.2.3. Vị trí các thành phần trên Kit và kích thước

#### 2.2.3.1. Sơ đồ khái niệm của Kit

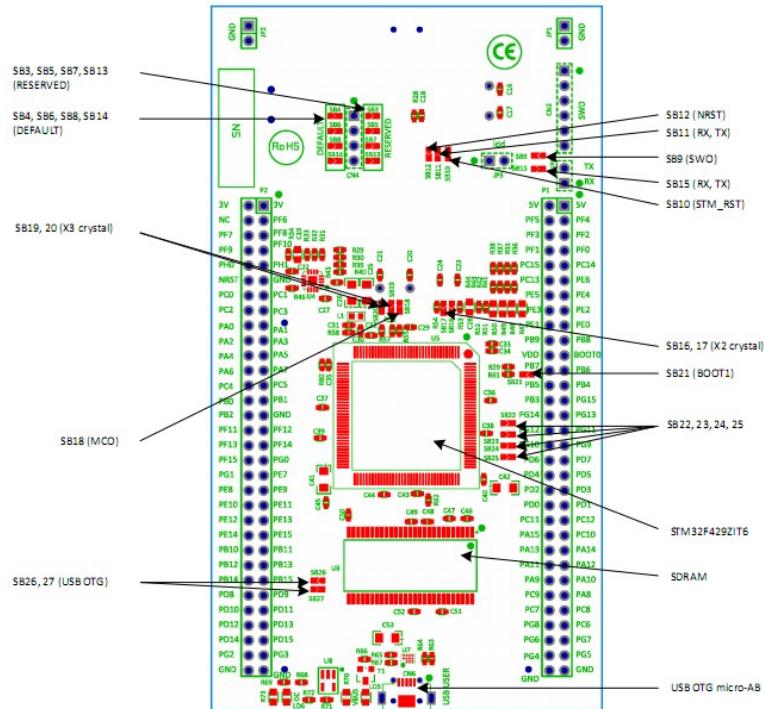


Hình 3: Sơ đồ khái niệm các thành phần của Kit

### 2.2.3.2. Vị trí của các thành phần trên mạch

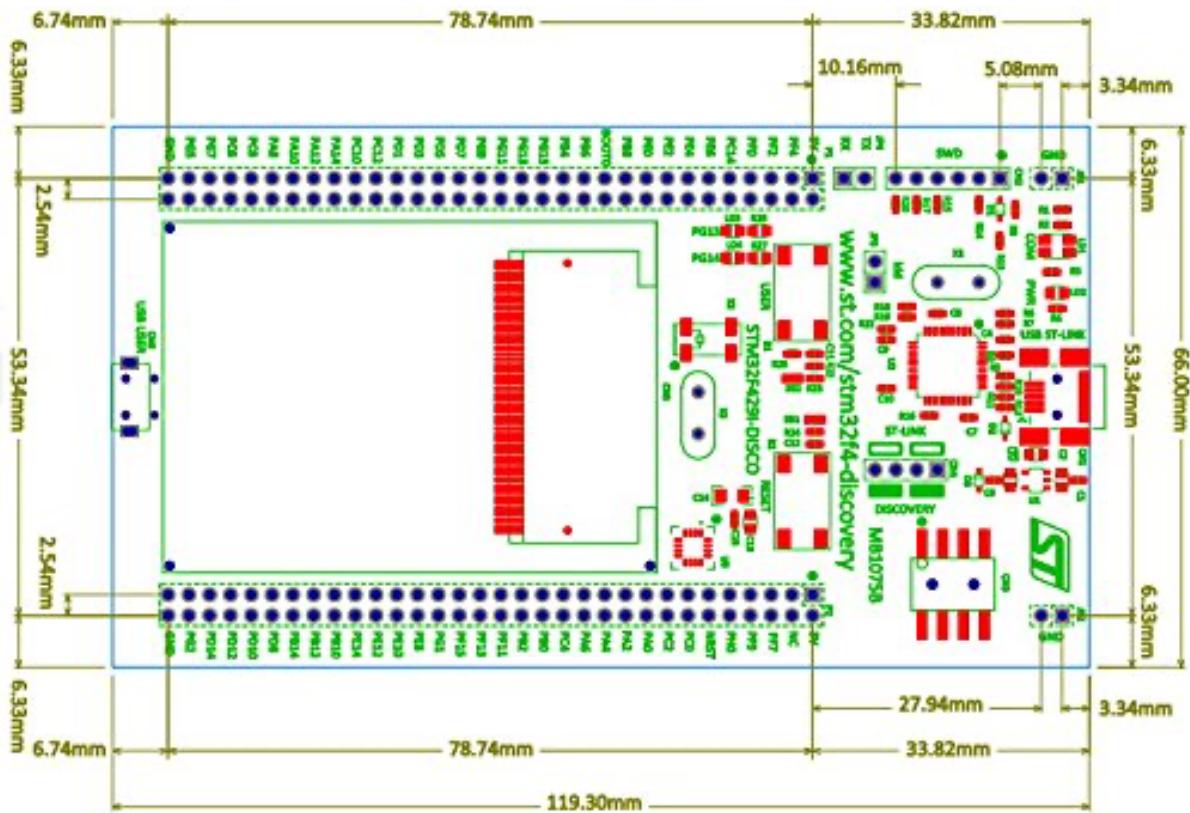


Hình 4: Vị trí các thành phần của Kit từ phía trên



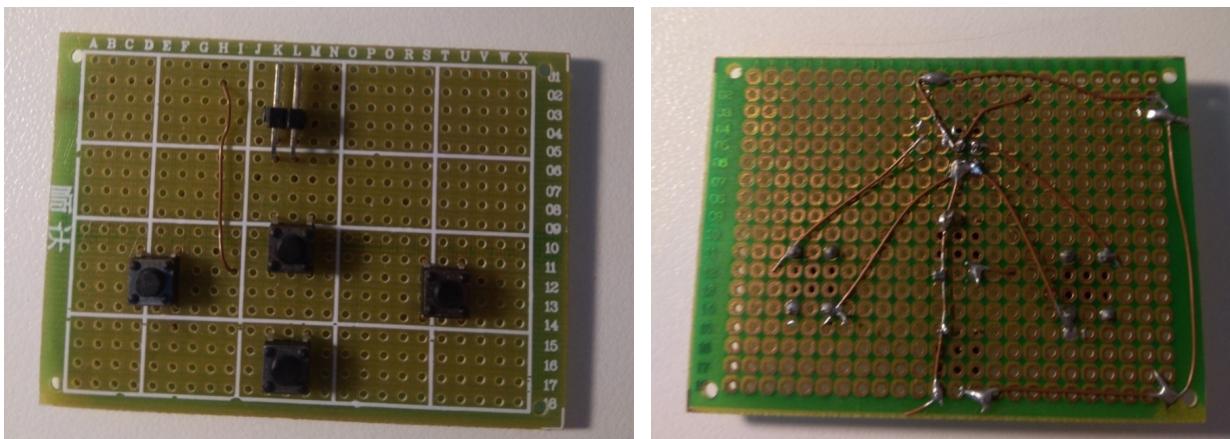
Hình 5: Vị trí các thành phần của Kit từ phía dưới

### 2.2.3.3. Kích thước của Kit



Hình 6: Kích thước của Kit (đơn vị: mm)

### 2.3. Mạch nút bấm

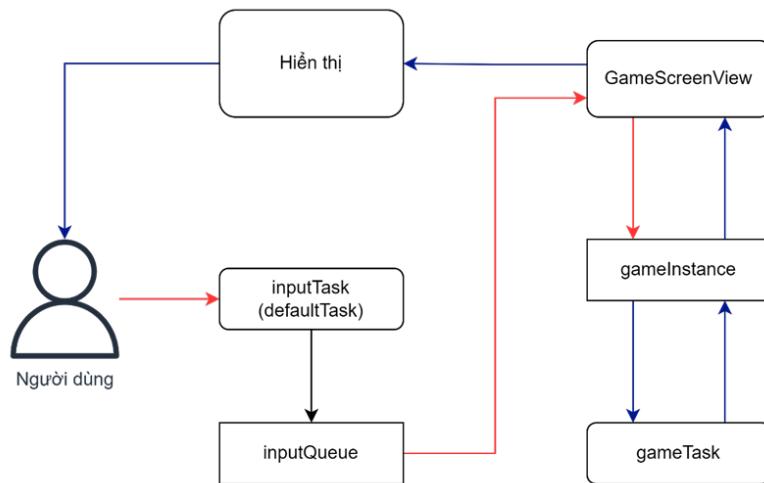


Hình 7: Mạch nút bấm

## CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH VÀ TRIỂN KHAI CÀI ĐẶT

### 3.1. Sơ đồ hệ thống

Với những yêu cầu đã đặt ra khi lên ý tưởng về đề tài, nhóm chúng em đã xây dựng sơ đồ toàn cảnh hệ thống như sau:



Hình 8: Sơ đồ hệ thống

Kiến trúc hệ thống gồm có 5 thành phần:

1. **GameScreenView**: Tiếp nhận dữ liệu từ người chơi và hiển thị hình ảnh lên màn hình
2. **inputTask**: Nhận dữ liệu từ người dùng và gửi vào **inputQueue** để chờ xử lý
3. **gameTask**: Cập nhật thông tin các đối tượng xử lý logic trò chơi
4. **inputQueue**: Chứa dữ liệu được người dùng gửi tới trò chơi thông qua các nút bấm
5. **gameInstance**: Đối tượng trò chơi của chương trình, chứa thông tin về trò chơi

## 3.2. Luồng hệ thống

Hệ thống bao gồm 2 luồng hoạt động chính:

1. Nhận dữ liệu từ tác nhân người chơi
2. Xử lý logic của game và xây dựng giao diện

### 3.2.1. Nhận dữ liệu từ tác nhân người chơi

Các button sẽ được cài đặt để gửi tín hiệu tới các cổng GPIO PE2, PE3, PE4, PE5.

Chương trình sẽ duy trì 1 default Task để liên tục đọc dữ liệu từ các cổng này và gửi lên hàng chờ.

```
618 // User input
619 GPIO_InitStruct.Pin = GPIO_PIN_2;
620 GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
621 GPIO_InitStruct.Pull = GPIO_PULLUP;
622 GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
623 HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);
624
625 GPIO_InitStruct.Pin = GPIO_PIN_3;
626 GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
627 GPIO_InitStruct.Pull = GPIO_PULLUP;
628 GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
629 HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);
630
631 GPIO_InitStruct.Pin = GPIO_PIN_4;
632 GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
633 GPIO_InitStruct.Pull = GPIO_PULLUP;
634 GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
635 HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);
636
637 GPIO_InitStruct.Pin = GPIO_PIN_5;
638 GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
639 GPIO_InitStruct.Pull = GPIO_PULLUP;
640 GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
641 HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);
642
643
```

Hình 9: Khởi tạo các chân vào ra của Kit

```
976     /* @retval None
977     */
978     /* USER CODE END Header_StartDefaultTask */
979     void StartDefaultTask(void *argument)
980     {
981         /* USER CODE BEGIN 5 */
982         /* Infinite loop */
983         uint32_t count1, count2, count3, count4;
984         uint8_t x1,x2,x3,x4;
985         for(;;)
986         {
987             //pg2
988             count1 = osMessageQueueGetCount(Queue1Handle);
989             if(count1 < 1){
990                 if(HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_2) == GPIO_PIN_RESET){
991                     x1 = 'R';
992                 } else {
993                     x1 = 'N';
994                 }
995                 osMessageQueuePut(Queue1Handle, &x1, 0 , 100);
996             }
997             //pg3
998             count2 = osMessageQueueGetCount(Queue2Handle);
999             if(HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_3) == GPIO_PIN_RESET){
1000                 if(count2 < 1){
1001                     x2 = 'L';
1002                 } else {
1003                     x2 = 'N';
1004                 }
1005                 osMessageQueuePut(Queue2Handle, &x2, 0 , 100);
1006             }
1007             //pg4
1008             count3 = osMessageQueueGetCount(Queue3Handle);
1009             if(count3 < 1){
1010                 if(HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_4) == GPIO_PIN_RESET){
1011                     x3 = 'U';
1012                 } else {
1013                     x3 = 'N';
1014                 }
1015                 osMessageQueuePut(Queue3Handle, &x3, 0 , 100);
1016             }
1017             //pg5
1018             count4 = osMessageQueueGetCount(Queue4Handle);
1019             if(count4 < 1){
1020                 if(HAL_GPIO_ReadPin(GPIOE, GPIO_PIN_5) == GPIO_PIN_RESET){
1021                     x4 = 'D';
1022                 } else {
1023                     x4 = 'N':
```

Hình 10: Hàm StartDefaultTask() để nhận dữ liệu từ tác nhân người dùng

Dữ liệu nằm trong hàng đợi sẽ được GameScreenView poll và lấy ra với mỗi lượt tick trong hàm.

```

103     // Get input
104     uint8_t res = 0;
105     uint32_t count = osMessageQueueGetCount(Queue1Handle);
106     if(count > 0) {
107         osMessageQueueGet(Queue1Handle,&res, NULL, osWaitForever);
108         if(res == 'R') {
109             gameInstance.ship.updateVelocityX(gameInstance.ship.VELOCITY);
110             shipImage.setBitmap(touchgfx::Bitmap(BITMAP_SHIP_RIGHT_ID));
111             osMessageQueueReset(Queue2Handle);
112         } else if(res == 'N'){
113             gameInstance.ship.updateVelocityX(0);
114             shipImage.setBitmap(touchgfx::Bitmap(BITMAP_SHIP_MAIN_ID));
115         }
116     }
117     uint32_t count2 = osMessageQueueGetCount(Queue2Handle);
118     if(count2 > 0) {
119         osMessageQueueGet(Queue2Handle,&res, NULL, osWaitForever);
120         if(res == 'L') {
121             gameInstance.ship.updateVelocityX(-gameInstance.ship.VELOCITY);
122             shipImage.setBitmap(touchgfx::Bitmap(BITMAP_SHIP_LEFT_ID));
123             osMessageQueueReset(Queue1Handle);
124         } else if(res == 'N'){
125             gameInstance.ship.updateVelocityX(0);
126             shipImage.setBitmap(touchgfx::Bitmap(BITMAP_SHIP_MAIN_ID));
127         }
128     }
129
130     uint32_t count3 = osMessageQueueGetCount(Queue3Handle);
131     if(count3 > 0) {
132         osMessageQueueGet(Queue3Handle,&res, NULL, osWaitForever);
133         if(res == 'U') {
134             gameInstance.ship.updateVelocityY(gameInstance.ship.VELOCITY);
135             osMessageQueueReset(Queue4Handle);
136         } else if(res == 'N'){
137             gameInstance.ship.updateVelocityY(0);
138         }
139
140     uint32_t count4 = osMessageQueueGetCount(Queue4Handle);
141     if(count4 > 0) {
142         osMessageQueueGet(Queue4Handle,&res, NULL, osWaitForever);
143         if(res == 'D') {
144             gameInstance.ship.updateVelocityY(-gameInstance.ship.VELOCITY);
145             osMessageQueueReset(Queue3Handle);
146         } else if(res == 'N'){
147

```

Hình 11: Lấy dữ liệu từ trong `inputQueue` của `GameScreenView`

Sau đó GameScreenView sẽ xử lý dữ liệu và cập nhật vào đối tượng trong gameInstace (ở đây là hướng di chuyển của ship – đối tượng mà người chơi điều khiển).

### 3.2.2. Xử lý logic của game và xây dựng giao diện

Khi 1 GameScreenView được tạo, chương trình sẽ tạo ra gameTask.

```

62@ void GameScreenView::setupScreen()
63 {
64     GameScreenViewBase::setupScreen();
65
66     // Kết thúc task game trước
67     osThreadTerminate(gameTaskHandle);
68     const osThreadAttr_t gameTask_attributes = {
69         .name = "gameTask", // Tên của task là "gameTask"
70         .stack_size = 8192 * 2, // Kích thước của stack được cấp phát cho task là 8192 * 2 bytes
71         .priority = (osPriority_t) osPriorityNormal,
72     };
73
74     // Tạo một task mới
75     gameTaskHandle = osThreadNew(gameTask, NULL, &gameTask_attributes);
76     shouldEndGame = false;
77     shouldStopTask = false;
78     shouldStopScreen = false;
79 }
80

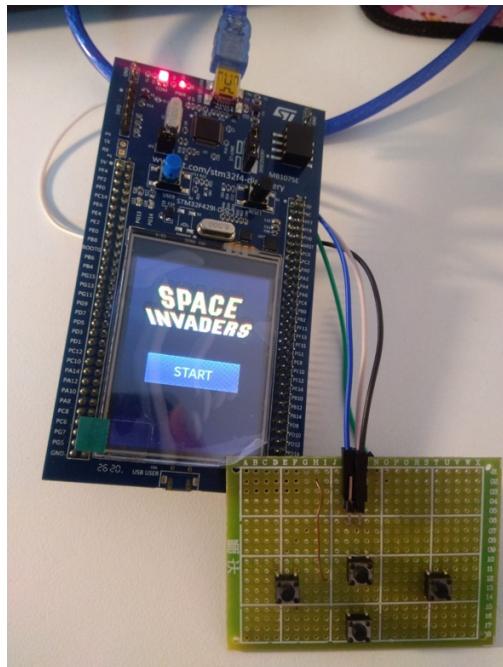
```

Hình 12: Hàm `GameScreenView::setupScreen()` xây dựng `gameTask`

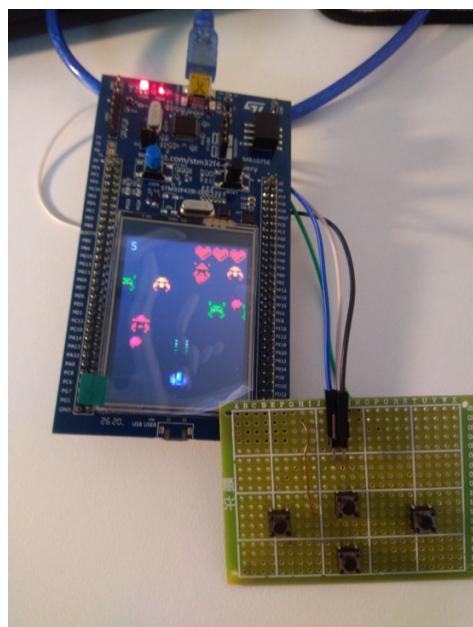
gameTask sẽ dựa trên các dữ liệu có được từ gameInstance để xử lý logic và cập nhật lại thông tin của gameInstace cho phù hợp. GameScreenView sẽ liên tục polling dữ liệu từ gameInstance để cập nhật lại hiển thị của các đối tượng trên màn hình và hiển thị cho người chơi.

## CHƯƠNG 4: ĐÁNH GIÁ TỔNG QUAN

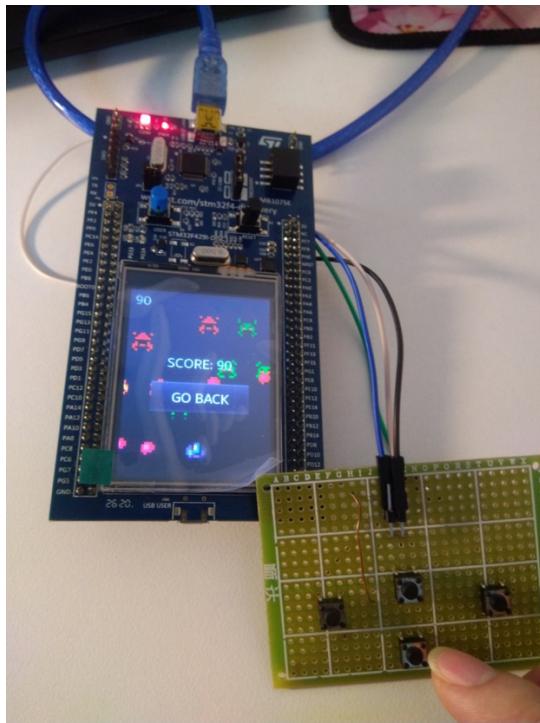
### 4.1. Sản phẩm demo



Hình 13: Giao diện mở đầu game



Hình 14: Giao diện khi chơi game



Hình 15: Giao diện khi kết thúc lượt chơi

## 4.2. Đánh giá hệ thống

Trong khuôn khổ đồ án môn học, nhóm chúng em đã thiết kế được mô hình đạt được các yêu cầu đề ra:

- Chức năng và luồng trò chơi: Quá trình chơi và kiểm tra vị trí đang hoạt động một cách chính xác, đảm bảo tính chính xác và logic trong cách thức thực hiện.
- Tương tác và độ phản hồi: Việc tương tác với trò chơi thông qua các nút bấm trên KIT STM32F429I-DISC1 diễn ra một cách suôn sẻ và có độ trễ thấp, sự tương tác mượt mà và dễ dàng.
- Hiệu suất: Trò chơi hoạt động một cách ổn định trên KIT STM32F429I-DISC1, không gây lag và không gây quá tải cho vi xử lý.
- Tích hợp phần cứng: Tích hợp của trò chơi với KIT STM32F429I-DISC1 được thực hiện thành công. Các phần cứng như nút bấm và màn hình hoạt động một cách chính xác và đáp ứng đúng yêu cầu của trò chơi.

Các vấn đề còn thiếu sót:

- Giao diện: Màn hình nhỏ của KIT STM32F429I-DISC1 làm hạn chế không gian hiển thị, gây khó khăn trong việc hiển thị một cách rõ ràng và tiện lợi.
- Khả năng tương tác hạn chế: Các nút bấm có sẵn trên kit có giới hạn, dẫn đến sự hạn chế trong việc cung cấp tương tác đa dạng và thuận tiện cho người chơi. Điều này có thể làm giảm đi sự thoải mái và trải nghiệm tương tác.

### **4.3. Hướng phát triển**

- Cải thiện giao diện người chơi: Tối ưu hóa giao diện cho màn hình nhỏ của KIT STM32F429I-DISC1.
- Thêm tính năng cho game: Tạo thêm chế độ chơi hoặc biến thể của trò chơi để tăng tính thú vị và sự đa dạng.
- Tối ưu hiệu suất: Tối ưu hóa mã nguồn để cải thiện hiệu suất và tối ưu hóa việc sử dụng tài nguyên phần cứng.

## PHỤ LỤC

### 1. Phân công nhiệm vụ

Nhiệm vụ	Người thực hiện
Tìm hiểu và lên ý tưởng trò chơi	Vũ Đình Tiến Trương Văn Hiển Nguyễn Hữu Huân
Xử lý tín hiệu nút bấm	Vũ Đình Tiến
Tìm hiểu thu thập hình ảnh đồ họa	Trương Văn Hiển Nguyễn Hữu Huân
Thiết kế đồ họa bằng TouchGFX	Trương Văn Hiển Nguyễn Hữu Huân
Phát triển phần mềm điều khiển	Vũ Đình Tiến Trương Văn Hiển Nguyễn Hữu Huân
Xử lý logic và hiển thị trò chơi	Trương Văn Hiển Nguyễn Hữu Huân
Viết báo cáo thành phần	Vũ Đình Tiến Trương Văn Hiển Nguyễn Hữu Huân
Tổng hợp báo cáo và in quyền đồ án	Trương Văn Hiển

## 2. Tài liệu tham khảo

- Bài giảng học phần “Hệ nhúng” – IT4210 của TS. Đỗ Công Thuần.
- <https://gociter.wordpress.com/2018/12/09/he-thong-nhung-giao-tiep-chuot-va-ban-phim-su-dung-kit-stm32f429-discovery/>
- <https://hocarm.org/huong-dan-stm32cubeide/>
- Youtube Lập trình STM32 trên STM32CubeIDE:  
[https://www.youtube.com/playlist?list=PLyD\\_mbw\\_VznOxKMBg5CjJ6OWvDzPNR8F4](https://www.youtube.com/playlist?list=PLyD_mbw_VznOxKMBg5CjJ6OWvDzPNR8F4)
- Schematic Discovery kit with STM32F429ZI MCU:  
<https://docs.rs-online.com/b72f/0900766b815247be.pdf>

## 3. Mã nguồn (source code) và các resource của đồ án

- Mã nguồn chương trình: <https://github.com/TienHunter/space.git>
- Link video demo:  
<https://drive.google.com/file/d/1YjCQygTOjeLaZpwpCA-5x5VgkKaZbc6H/view?usp=sharing>
- IDE lập trình: STM32CubeIDE
- Công cụ vẽ sơ đồ, biểu đồ: <https://app.diagrams.net>
- Bản thiết kế mạch điện KIT STM32F429I-DISC1:  
<https://dtxvn.com/san-pham/stm32f429i-disc1/>  
<https://www.waveshare.com/32f429idiscovery.htm>