**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**Khoa Điện – Điện tử**

**Bộ môn Tự động**



**BÁO CÁO BÀI TẬP LỚN**

**Môn học**

**HỆ THỐNG ĐIỀU KHIỂN NHÚNG**

**Đề tài**

**Điều chế xung PWM để điều khiển động cơ DC sử dụng vi điều khiển STM32F407**

|  |  |
| --- | --- |
| **Sinh viên thực hiện** | **MSSV** |
| Lê Thành Thiện | 1713295 |
| Phan Nguyễn | 1712407 |
| Trần Thiên Ân | 1710557 |
| Vũ Ngọc Nguyên | 1712405 |

*Giáo viên hướng dẫn: Thầy Nguyễn Vĩnh Hảo.*

Thành phố Hồ Chí Minh, tháng 7 năm 2020

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**Khoa Điện – Điện tử**

**Bộ môn Tự động**



**BÁO CÁO BÀI TẬP LỚN**

**Môn học**

**HỆ THỐNG ĐIỀU KHIỂN NHÚNG**

**Đề tài**

**Điều chế xung PWM để điều khiển động cơ DC sử dụng vi điều khiển STM32F407**

|  |  |
| --- | --- |
| **Sinh viên thực hiện** | **MSSV** |
| Lê Thành Thiện | 1713295 |
| Phan Nguyễn | 1712407 |
| Trần Thiên Ân | 1710557 |
| Vũ Ngọc Nguyên | 1712405 |

*Giáo viên hướng dẫn: Thầy Nguyễn Vĩnh Hảo.*

Thành phố Hồ Chí Minh, tháng 7 năm 2020

**MỤC LỤC**

**Đề mục Trang**

[Tóm tắt v](#_Toc45277696)

[1) Giới thiệu 1](#_Toc45277697)

[1.1) Tổng quan 1](#_Toc45277698)

[1.2) Nhiệm vụ đề tài 2](#_Toc45277699)

[1.3) Phân chia công việc trong nhóm 2](#_Toc45277700)

[2) Lý thuyết 3](#_Toc45277701)

[2.1) Giới thiệu dòng vi điều khiển STM32 3](#_Toc45277702)

[2.1.1) Sự tinh vi: 3](#_Toc45277703)

[2.1.2) Khả năng an toàn 4](#_Toc45277704)

[2.1.3) Tính bảo mật 4](#_Toc45277705)

[2.1.4) Phát triển phần mềm 4](#_Toc45277706)

[2.2) Giới thiệu KIT STM32F407 Discovery 5](#_Toc45277707)

[3) Thiết kế và thực hiện phần cứng 6](#_Toc45277708)

[4) Thiết kế và thực hiện phần mềm 7](#_Toc45277709)

[4.1) Yêu cầu đặt ra cho phần mềm 7](#_Toc45277710)

[4.2) Lưu đồ giải thuật 7](#_Toc45277711)

[5) Kết quả thực hiện 8](#_Toc45277712)

[5.1) Điều khiển vị trí 8](#_Toc45277713)

[5.2) Điều khiển vận tốc 10](#_Toc45277714)

[6) Kết luận và hướng phát triển 13](#_Toc45277715)

[6.1) Kết luận 13](#_Toc45277716)

[6.2) Hướng phát triển 13](#_Toc45277717)

**Danh mục hình**

[Hình 3.1: Sơ đồ kết nối phần cứng vẽ trên phần mềm Proteus (Schematic Capture). 6](#_Toc45277685)

[Hình 3.2: Sơ đồ kết nối phần cứng vẽ trên phần mềm Proteus (PCB Layout). 6](#_Toc45277686)

[Hình 4.1: Lưu đồ giải thuật của chương trình lập trình. 7](#_Toc45277687)

[Hình 5.1: Kết quả điều khiển vị trí với tần số 10 kHz. 8](#_Toc45277688)

[Hình 5.2: Kết quả điều khiển vị trí với tần số 1 kHz. 8](#_Toc45277689)

[Hình 5.3: Kết quả điều khiển vị trí với tần số 50 Hz. 9](#_Toc45277690)

[Hình 5.4: Kết quả điều khiển vị trí với tần số 65 kHz. 9](#_Toc45277691)

[Hình 5.5: Kết quả điều khiển vận tốc với tần số 10 kHz. 10](#_Toc45277692)

[Hình 5.6: Kết quả điều khiển vận tốc với tần số 1 kHz. 11](#_Toc45277693)

[Hình 5.7: Kết quả điều khiển vận tốc với tần số 50 Hz. 11](#_Toc45277694)

[Hình 5.8: Kết quả điều khiển vận tốc với tần số 65 kHz. 12](#_Toc45277695)

# Tóm tắt

Bài báo cáo này trình bày về các vấn đề về sử dụng STM32F407 Discovery trong việc điều khiển một động cơ DC bằng xung PWM và có giao tiếp UART với máy tính, trình bày về kết nối phần cứng cũng như chương trình lập trình mà nhóm đã thực hiện. Cụ thể, bài báo cáo bao gồm các mục như sau:

1. Giới thiệu:

Giới thiệu về đề tài mà nhóm đã thực hiện trong bài tập lớn này, về việc sử dụng vi xử lý ARM trong hệ thống điều khiển nhúng.

1. Lý thuyết:

Trình bày những lý thuyết cơ bản về dòng vi điều khiển STM32.

1. Thiết kế và thực hiện phần cứng:

Trình bày về kết nối phần cứng, sơ đồ mạch đã thực hiện.

1. Thiết kế và thực hiện phần mềm:

Trình bày về lưu đồ giải thuật và chương trình lập trình đã thực hiện.

1. Kết quả thực hiện:

Trình bày về các kết quả mà nhóm đã đạt được.

1. Kết luận và hướng phát triển:

Trình bày những kết luận từ kết quả thực hiện và hướng phát triển cho đề tài.

# Giới thiệu

## Tổng quan

Hệ thống điều khiển tự động có nhiều ứng dụng trong thực tế ở hầu hết các lĩnh vực như điều khiển, đo đạc, viễn thông, quân sự, nghiên cứu khoa học,... xây dựng 1 hệ thống điều khiển tự động đòi hỏi một pham vi rộng kiến thức về phần cứng, phần mềm. Cùng với sự phát triển của khoa học kỹ thuật, công nghệ điều khiển tự động ngày càng được phát triển cả về thiết bị phần cứng và các giải pháp phần mềm.

Cùng với sự phát triển của công nghệ vi điện tử, các vi điều khiển đã có nhiều cải thiện về cấu trúc, thiết kế hệ thống, khả năng xử lý, tái lập hệ thống giúp giảm chi phí sản xuất, rút ngắn thời gian cải tiến, nâng cấp, sản xuất hệ thống.

Cấu trúc vi xử lý ARM (viết tắt từ tên gốc là Advanced RISC Machine) là một loại cấu trúc vi xử lý 32 bit và 64 bit kiểu RISC được sử dụng rộng rãi trong các thiết kế hệ thống nhúng. Do có đặc điểm tiết kiệm năng lượng, các bộ vi xử lý ARM chiếm ưu thế trong các sản phẩm điện tử di động, mà với các sản phẩm này việc tiêu thụ công suất thấp là một mục tiêu thiết kế quan trọng hàng đầu. Ngày nay, hơn 75% CPU nhúng 32-bit là thuộc họ ARM, điều này khiến ARM trở thành cấu trúc 32-bit được sản xuất nhiều nhất trên thế giới. Giải pháp hệ thống trên chip (System-On-Chip) dựa trên bộ vi xử lý nhúng ARM được ứng dụng vào rất nhiều thị trường khác nhau bao gồm các ứng dụng doanh nghiệp, các hệ thống ô tô, mạng gia đình và công nghệ mạng không dây. Nhiều thiết bị xử lý ảnh chuyên dụng đã được thiết kế sử dụng vi điều khiển ARM. Dòng vi xử lý ARM Cortex dựa trên một kiến trúc chuẩn đủ để đáp ứng hầu hết các yêu cầu về hiệu năng làm việc trong tất cả các lĩnh vực trên. Thêm vào đó là việc lập trình được đơn giản hóa đáng kể giúp kiến trúc ARM trở thành một lựa chọn tốt cho ngay cả những ứng dụng đơn giản nhất. Những đặc điểm nổi trội của dòng ARM Cortex đã thu hút các nhà sản xuất IC, hơn 240 dòng vi điều khiển dựa vào nhân Cortex đã được giới thiệu. Không nằm ngoài xu hướng đó, hãng sản xuất chip ST Microelectronic đã nhanh chóng đưa ra dòng STM32 là vi điều khiển dựa trên nền tảng lõi ARM Cortex®- M thế hệ mới do hãng ARM thiết kế.

Khả năng kết hợp trong thiết kế hệ thống vi điều khiển STM32 à các giải thuật phần mềm cho phép xây dựng một hệ thống điều khiển xung PWM chuẩn xác đáp ứng được nhu cầu của thiết kế.

## Nhiệm vụ đề tài

**Tên đề tài:** Điều chế xung PWM để điều khiển động cơ DC sử dụng vi điều khiển STM32F407.

**Nhiệm vụ đề tài:**

* Nội dung 1: Tìm hiểu về vi điều khiển STM32F407, module điều khiển động cơ L298, động cơ DC và các thiết bị, linh kiện liên quan.
* Nội dung 2: Tìm hiểu về nguyên lý hoạt động của điều chế xung PWM.
* Nội dung 3: Tìm hiểu về kết nối phần cứng giữa KIT STM32F407 Discovery với module điều khiển động cơ L298, động cơ DC và các thiết bị, linh kiện liên quan.
* Nội dung 4: Phát triển phần mềm nhúng cho phép cấu hình hệ thống và xuất ra xung PWM.
* Nội dung 5: Chạy mô phỏng chương trình lập trình đã thực hiện, thực hiện kiểm tra và chỉnh sửa, bổ sung chương trình lập trình.
* Nội dung 6: Thi công mạch phần cứng và thực hiện các kết nối phần cứng cần thiết.
* Nội dung 7: Thực hiện chạy chương trình điều khiển với các trường hợp khác nhau và kiểm tra kết quả.
* Nội dung 8: Thực hiện các chỉnh sửa, bổ sung cần thiết để hoàn thiện chương trình lập trình.

## Phân chia công việc trong nhóm

1. Phan Nguyễn:
   * Thực hiện viết chương trình cho vi điều khiển STM32F407 trên phần mềm Keil.
2. Lê Thành Thiện:
   * Hỗ trợ bạn Phan Nguyễn viết chương trình cho vi điều khiển STM32F407 trên phần mềm Keil.
3. Trần Thiên Ân:
   * Thực hiện viết chương trình tạo giao diện GUI trên phần mềm Visual Studio.
4. Vũ Ngọc Nguyên:
   * Thực hiện phần lý thuyết trong báo cáo.
   * Thực hiện thi công mạch phần cứng và thực hiện các kết nối phần cứng cần thiết.

# Lý thuyết

## Giới thiệu dòng vi điều khiển STM32

STM32 là vi điều khiển do hãng ST Microelectronic sản xuất dựa trên nền tảng lõi vi xử lý ARM Cortex®-M. Là một dòng sản phẩm vi xử điều khiển 32 bit kết hợp các ưu điểm về hiệu suất cao, khả năng xử lý thời gian thực, xử lý tín hiệu số, tiêu thụ ít năng lượng, hoạt động điện áp thấp, trong khi duy trì khả năng tích hợp đầy đủ và dễ dàng phát triển ứng dụng.

Vi điều khiển STM32 dựa trên một lõi tiêu chuẩn công nghiệp, có thể sử dụng nhiều công cụ và phần mềm để phát triển ứng dụng. Điều này làm cho dòng STM32 là sự lựa chọn lý tưởng đối với các dự án nhỏ hoặc cho thiết kế nền tảng. ST đã đưa ra thị trường 4 dòng vi điều khiển dựa trên ARM7 và ARM9, nhưng STM32 là một bước tiến quan trọng trên đường cong chi phí và hiệu suất (price/performance), giá chỉ gần 1 Euro với số lượng lớn, STM32 là sự thách thức thật sự với các vi điều khiển 8 và 16 bit truyền thống. STM32 đầu tiên gồm 14 biến thể khác nhau, được phân thành hai dòng: dòng Performance có tần số hoạt động của CPU lên tới 72 Mhz và dòng Access có tần số hoạt động lên tới 36 Mhz. Các biến thể STM32 trong hai nhóm này tương thích hoàn toàn về cách bố trí chân (pin) và phần mềm, đồng thời kích thước bộ nhớ FLASH ROM có thể lên tới 512K và 64K SRAM.

### Sự tinh vi:

Thoạt nhìn thì các ngoại vi của STM32 cũng giống như những vi điều khiển khác, như hai bộ chuyển đổi ADC, timer, I2C, SPI, CAN, USB và RTC. Tuy nhiên mỗi ngoại vi trên đều có rất nhiều đặc điểm thú vị. Ví dụ như bộ ADC 12 bit có tích hợp một cảm biến nhiệt 12 độ để tự động hiệu chỉnh khi nhiệt độ thay đổi và hỗ trợ nhiều chế độ chuyển đổi. Mỗi bộ định thời có 4 khối capture compare (dùng để bắt sự kiện với tính năng input capture và tạo dạng sóng ở ngõ ra với output compare), mỗi khối định thời có thể liên kết với các khối định thời khác để tạo ra một mảng các định thời chính xác hơn. Một bộ định thời chuyên hỗ trợ điều khiển động cơ với 6 đầu ra tín hiệu điều biến độ rộng xung PWM với dead time (khoảng thời gian được chèn vào giữa hai đầu tín hiệu xuất PWM bù nhau trong điều khiển mạch cầu H) lập trình được và một đường break input (khi phát hiện điều kiện dừng khẩn cấp) sẽ buộc tín hiệu PWM sang một trạng thái an toàn đã được cài sẵn. Ngoại vi nối tiếp SPI có một khối kiểm tổng (CRC) bằng phần cứng cho 8 và 16 word hỗ trợ tích cực cho giao tiếp thẻ nhớ SD hoặc MMC. STM32 có hỗ trợ thêm tối đa 12 kênh DMA (Direct Memory Access). Mỗi kênh có thể được dùng để truyền dữ liệu đến các thanh ghi ngoại vi hoặc từ các thanh ghi ngoại vi với kích thước từ (word) dữ liệu truyền đi có thể là 8/16 hoặc 32 bit. Mỗi ngoại vi có thể có một bộ điều khiển DMA (DMA controller) đi kèm dùng để gửi hoặc truy vấn dữ liệu như yêu cầu. Một bộ phân xử bus nội (bus arbiter) và ma trận bus (bus matrix) tối thiểu hoá sự tranh chấp bus giữa truy cập dữ liệu thông qua CPU (CPU data access) và các kênh DMA. Điều đó cho phép các đơn vị DMA hoạt động linh hoạt, dễ dùng và tự động điều khiển các luồng dữ liệu bên trong vi điều khiển. STM32 là một vi điều khiển tiêu thụ năng lượng thấp và đạt hiệu suất cao. Nó có thể hoạt động ở điện áp 2V, chạy ở tần số 72 MHz và dòng tiêu thụ chỉ có 36mA với tất cả các khối bên trong vi điều khiển đều được hoạt động. Kết hợp với các chế độ tiết kiệm năng lượng của Cortex, STM32 chỉ tiêu thụ 2μA khi ở chế độ Standby. Một bộ dao động nội RC 8 MHz cho phép chip nhanh chóng thoát khỏi chế độ tiết kiệm năng lượng trong khi bộ dao động ngoài đang khởi động. Khả năng nhanh đi vào và thoát khỏi các chế độ tiết kiệm năng lượng làm giảm nhiều sự tiêu thụ năng lượng tổng thể.

### Khả năng an toàn

Ngày nay các ứng dụng hiện đại thường phải hoạt động trong môi trường khắt khe, đòi hỏi tính an toàn cao, cũng như đòi hỏi sức mạnh xử lý và càng nhiều thiết bị ngoại vi. Để đáp ứng các yêu cầu 13 khắt khe đó, STM32 cung cấp một số tính năng phần cứng hỗ trợ các ứng dụng một cách tốt nhất. Chúng bao gồm một bộ phát hiện điện áp thấp, một hệ thống bảo vệ xung Clock và hai bộ Watchdogs. Bộ đầu tiên là một Watchdog cửa sổ (windowed watchdog). Watchdog này phải được làm tươi trong một khung thời gian xác định. Nếu nhấn nó quá sớm, hoặc quá muộn, thì Watchdog sẽ kích hoạt. Bộ thứ hai là một Watchdog độc lập (independent watchdog), có bộ dao động bên ngoài tách biệt với xung nhịp hệ thống chính. Hệ thống bảo vệ xung nhịp có thể phát hiện lỗi của bộ dao động chính bên ngoài (thường là thạch anh) và tự động chuyển sang dùng bộ dao động nội RC 8 MHz.

### Tính bảo mật

Một trong những yêu cầu khắt khe khác của thiết kế hiện đại là nhu cầu bảo mật mã chương trình để ngăn chặn sao chép trái phép phần mềm. Bộ nhớ Flash của STM32 có thể được khóa để chống truy cập đọc Flash thông qua cổng gỡ lỗi (Debug). Khi tính năng bảo vệ đọc được kích hoạt, bộ nhớ Flash cũng được bảo vệ chống ghi để ngăn chặn mã không tin cậy được chèn vào bảng vector ngắt. Hơn nữa bảo vệ ghi có thể được cho phép trong phần còn lại của bộ nhớ Flash. STM32 cũng có một đồng hồ thời gian thực và một khu vực nhỏ dữ liệu trên SRAM được nuôi nhờ nguồn pin. Khu vực này có một đầu vào chống giả mạo (anti-tamper input), có thể kích hoạt một sự kiện ngắt khi có sự thay đổi trạng thái ở đầu vào này. Ngoài ra một sự kiện chống giả mạo sẽ tự động xóa dữ liệu được lưu trữ trên SRAM được nuôi bằng nguồn pin.

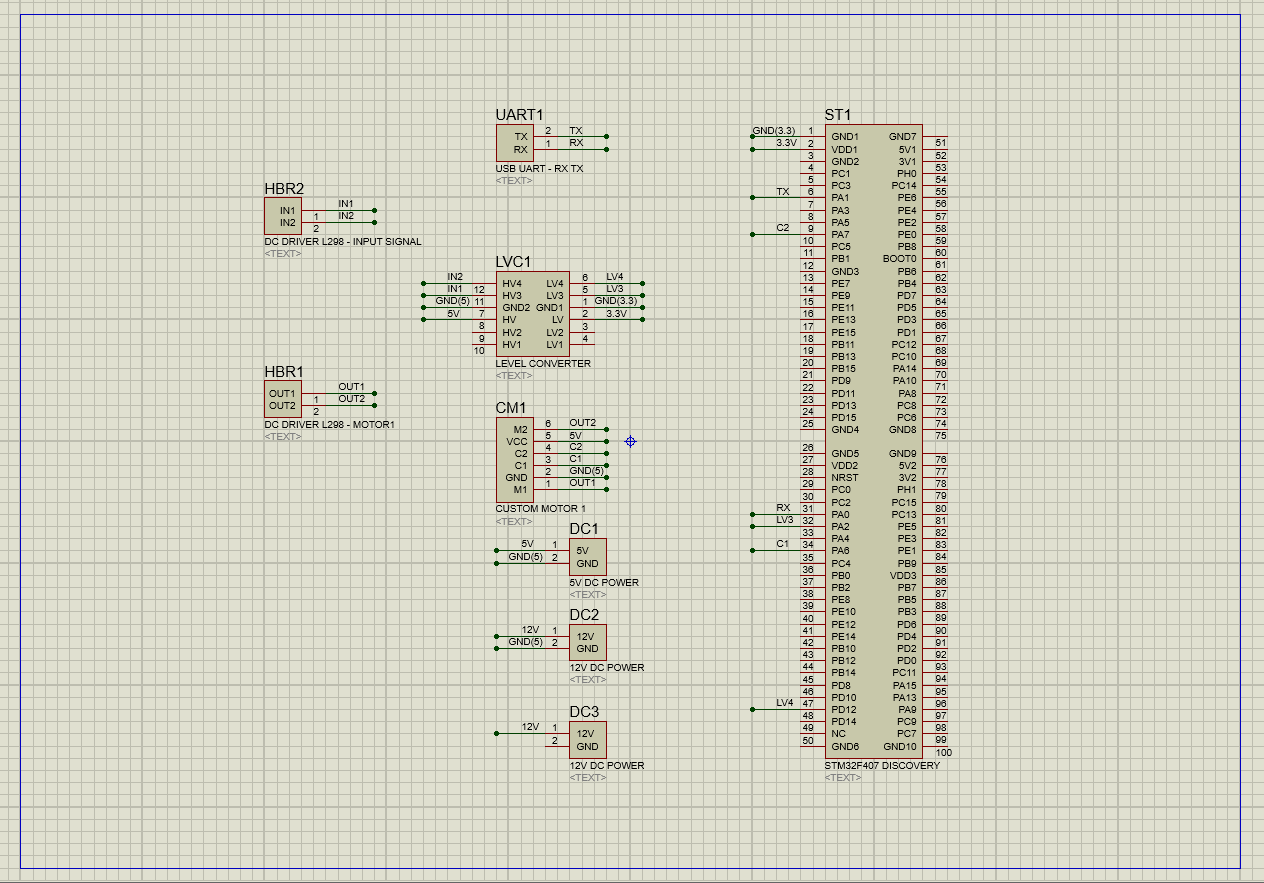
### Phát triển phần mềm

Các công cụ phát triển cho ARM hiện có đã được hỗ trợ tập lệnh Thumb-2 và dòng Cortex. Ngoài ra ST cũng cung cấp một thư viện điều khiển thiết bị ngoại vi, một bộ thư viện phát triển USB như là một thư viện ANSI C và mã nguồn tương thích với các thư viện trước đó được công bố cho vi điều khiển STR7 và STR9. Có rất nhiều hệ điều hành thời gian thực RTOS (Real Time Operating System) mã nguồn mở và thương mại và middleware (TCP/IP, hệ thống tập tin, v.v.) hỗ trợ cho họ Cortex. Dòng Cortex-M3 cũng đi kèm với một hệ thống gỡ lỗi hoàn toàn mới gọi là CoreSight. Truy cập vào hệ thống CoreSight thông qua cổng truy cập gỡ lỗi (Debug Access Port), cổng này hỗ trợ kết nối chuẩn JTAG hoặc giao diện 2 dây (serial wire-2 14 Pin), cũng như cung cấp trình điều khiển chạy gỡ lỗi, hệ thống CoreSight trên STM32 cung cấp hệ thống điểm truy cập (data watchpoint) và một công cụ theo dõi (instrumentation trace). Công cụ này có thể gửi thông tin về ứng dụng được lựa chọn đến công cụ gỡ lỗi. Điều này có thể cung cấp thêm các thông tin gỡ lỗi và cũng có thể được sử dụng trong quá trình thử nghiệm phần mềm. STM32 có sẵn một bộ thư viện ngoại vi chuẩn và mẫu, ví dụ hỗ trợ lập trình mà không cần kiến thức chuyên sâu hay hiểu rõ datasheet của chip, giúp nhanh chóng tập trung vào việc viết chương trình, tiết kiệm thời gian phát triển sản phẩm.

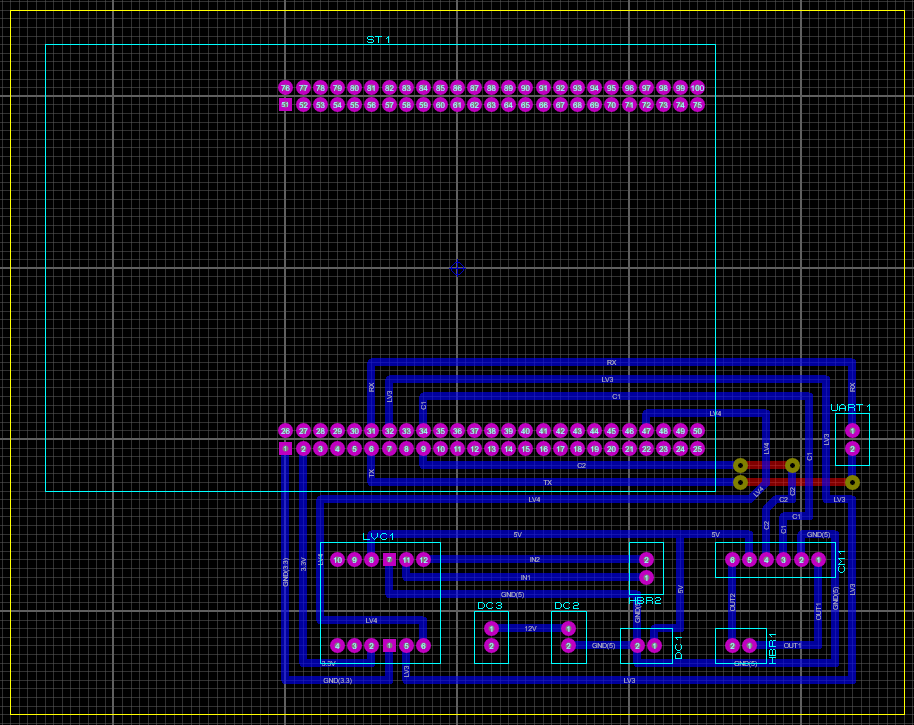
## Giới thiệu KIT STM32F407 Discovery

Nhóm sử dụng Kit STM32F407VG Discovery cho bắt ảnh qua Camera OV7670. Hình 1.3: Kit STM32F407VG Discovery Bộ kit STM32F4-DISCOVERY với vi điều khiển hiệu suất cao STM32F407VGT6, cho phép người dùng dễ dàng phát triển các ứng dụng xử lý tín hiệu số (hình ảnh, video…). Nó bao gồm một công cụ ST-LINK tích hợp sẵn trên bảng mạch giúp nạp chương trình, gỡ lỗi nhanh chóng. 15 Các tính năng chính - Vi điều khiển 32-bit ARM Cortex®-M4 STM32F407VGT6 với lõi FPU hỗ trợ xử lý tính toán dấu phẩy động, 1-MB bộ nhớ Flash, 192 Kbyte RAM. - On-board ST-LINK/V2 trên STM32F4-DISCOVERY giúp nạp chương trình, gỡ lỗi. - Nguồn điện cung cấp cho bảng mạch: thông qua cổng USB hoặc từ một nguồn cung cấp điện áp 5V bên ngoài. - Từ bảng mạch, có thể cấp nguồn 3,3 V và 5 V cho các ứng dụng. - Cảm biến chuyển động LIS302DL, ST MEMS 3 trục gia tốc. - Cảm biến âm thanh MP45DT02 ST-MEMS, mic cảm biến âm thanh vô hướng kỹ thuật số. - Bộ chuyển đổi DAC âm thanh CS43L22. - Tám đèn LED: + LD1 (đỏ / xanh lá cây) để giao tiếp USB + LD2 (màu đỏ) báo hiệu nguồn 3,3 V on + Bốn đèn LED màu: LD3 (màu cam), LD4 (màu xanh lá cây), LD5 (màu đỏ) và LD6 (màu xanh dương). + Hai USB OTG LED LD7 (màu xanh lá cây) VBUS và LD8 (màu đỏ). - Hai nút bấm (nút bấm User màu xanh, nút bấm Reset màu đen). - OTG FS USB với cổng nối micro-AB. - Header mở rộng cho tất cả LQFP100 I/O.

# Thiết kế và thực hiện phần cứng



Hình 3.1: Sơ đồ kết nối phần cứng vẽ trên phần mềm Proteus (Schematic Capture).



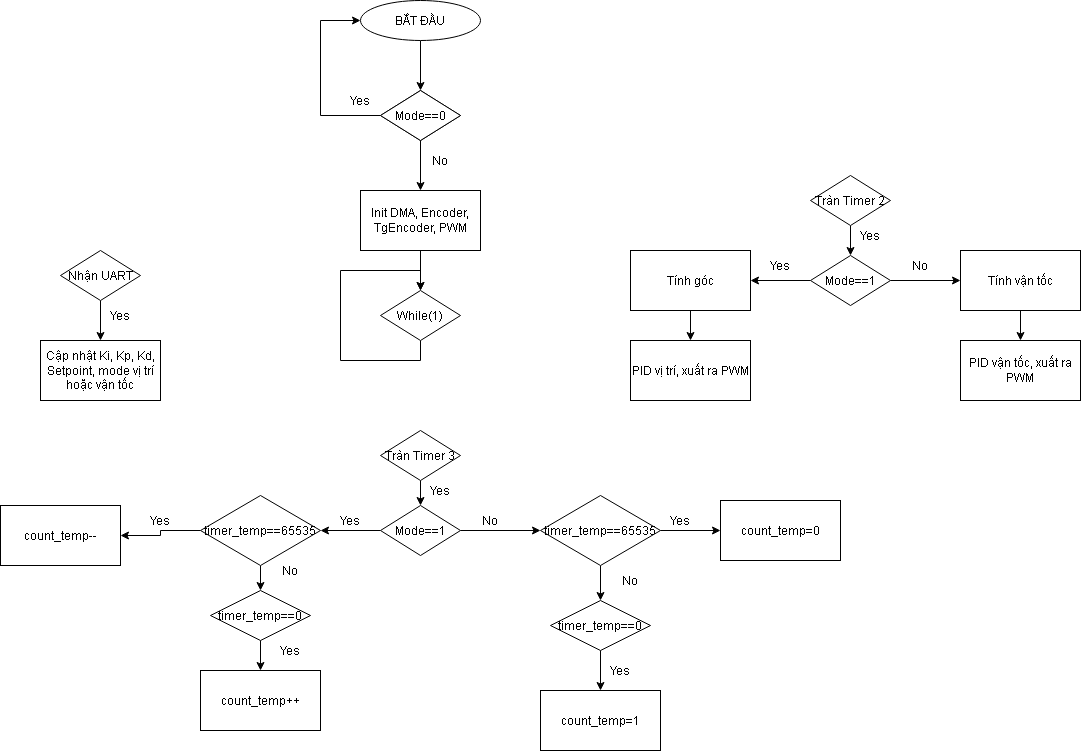
Hình 3.2: Sơ đồ kết nối phần cứng vẽ trên phần mềm Proteus (PCB Layout).

# Thiết kế và thực hiện phần mềm

## Yêu cầu đặt ra cho phần mềm

* Chương trình lập trình cho vi điều khiển để đọc được xung encoder.
* Chương trình lập trình cho vi điều khiển xuất xung PWM ra điều khiển PID động cơ DC motor.
* Viết chương trình lập trình giao diện GUI trên máy tính và giao diện này có khả năng:
* Cho phép kết nối cổng COM với các thông số tự chọn.
* Cho phép chọn chế độ điều khiển vị trí và vận tốc động cơ.
* Cho phép cài đặt các thông số Ki, Kp, Kd để điều khiển động cơ.
* Hiển thị tốc độ hoặc vị trí và vẽ đồ thị liên tục trong quá trình điều khiển để quan sát được đáp ứng của nhiệt độ theo thời gian.

## Lưu đồ giải thuật

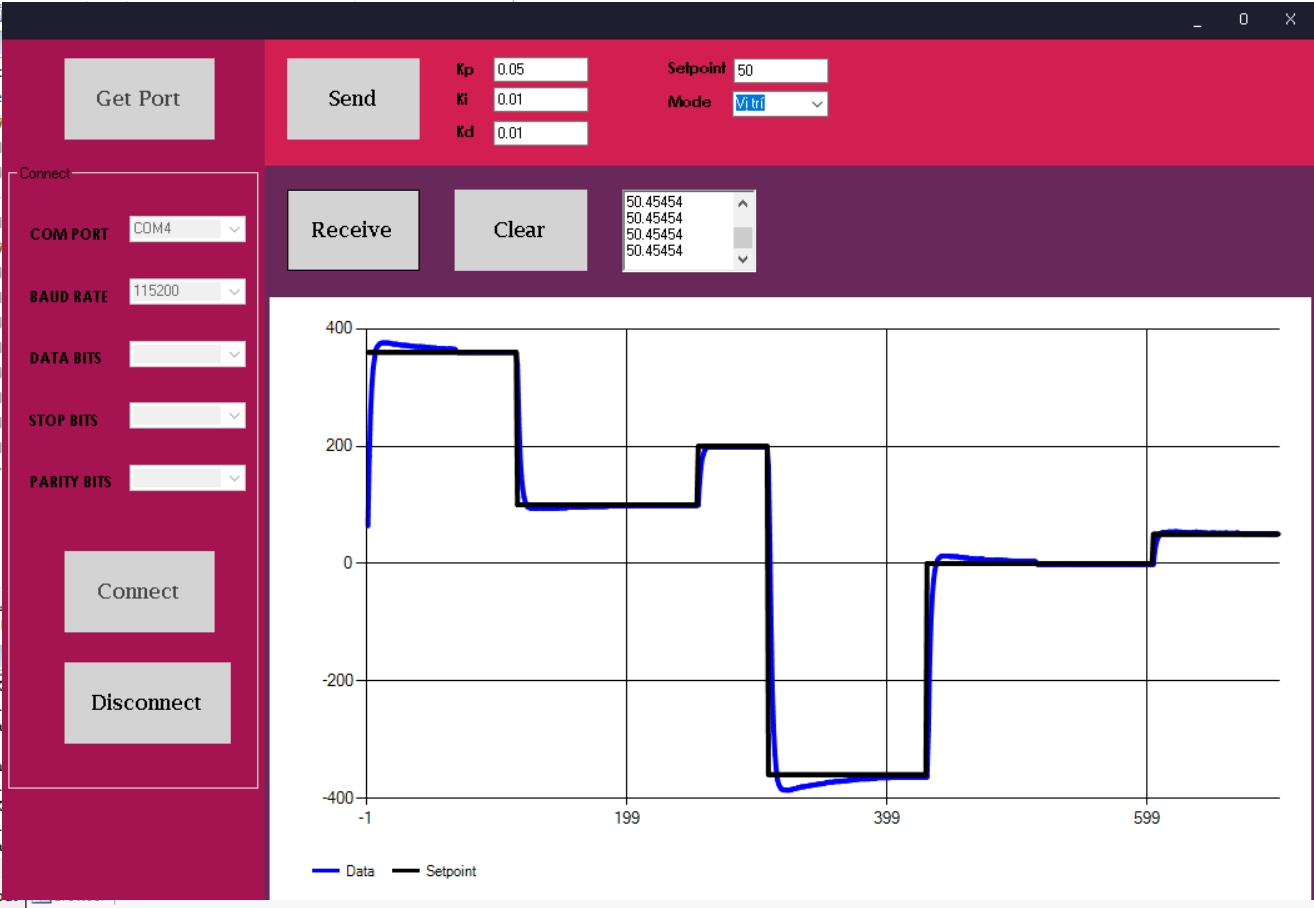


Hình 4.1: Lưu đồ giải thuật của chương trình lập trình.

# Kết quả thực hiện

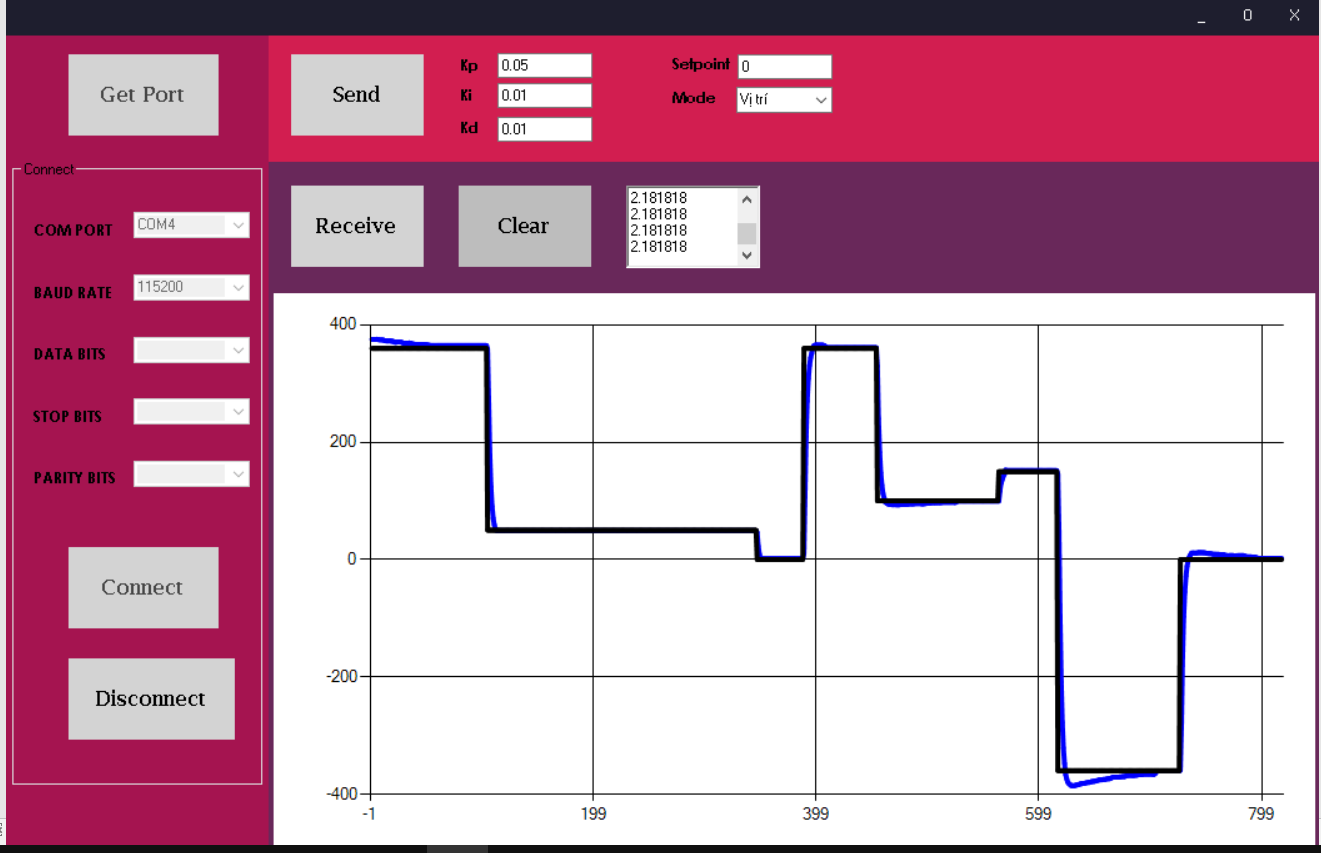
## Điều khiển vị trí

**Tần số 10 kHz:**



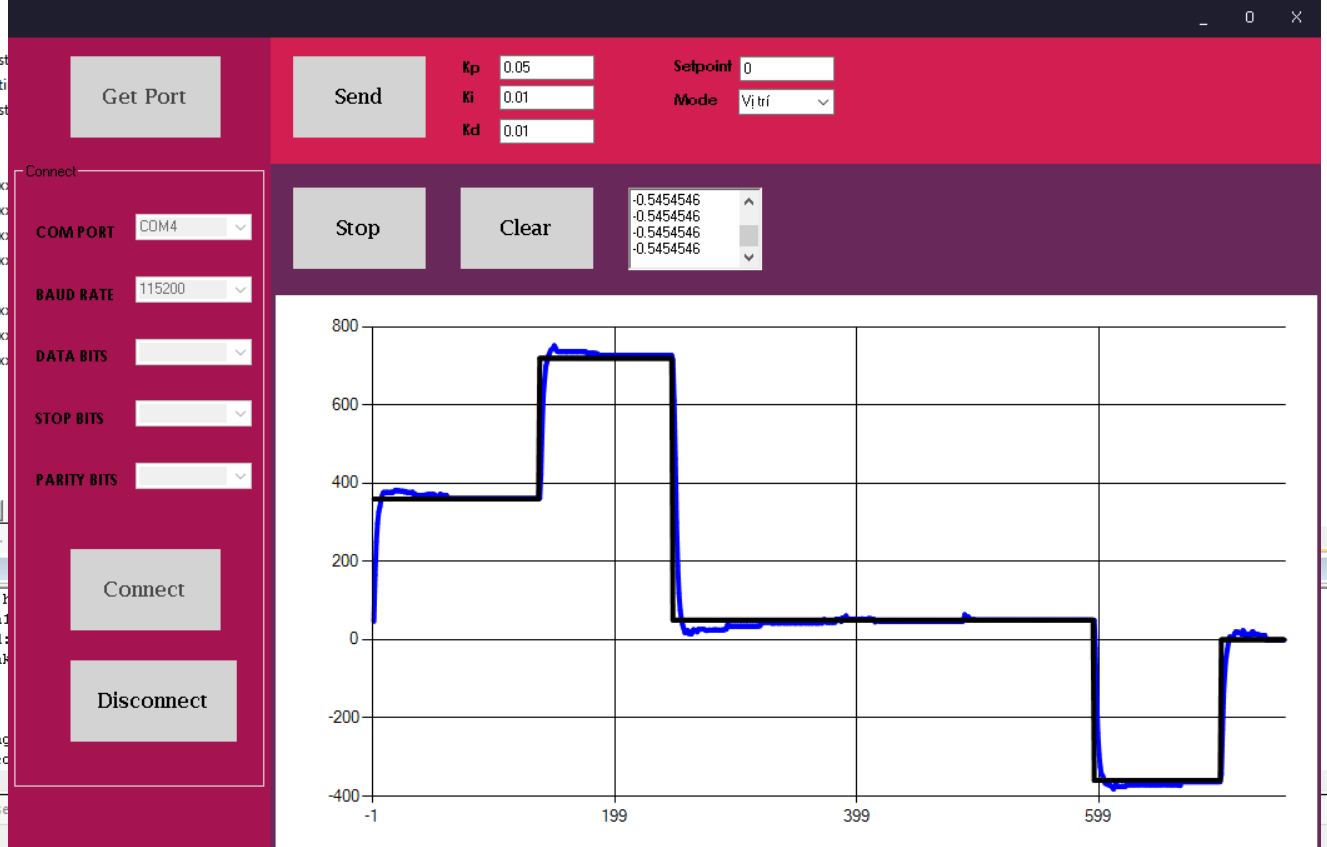
Hình 5.1: Kết quả điều khiển vị trí với tần số 10 kHz.

**Tần số 1 kHz:**



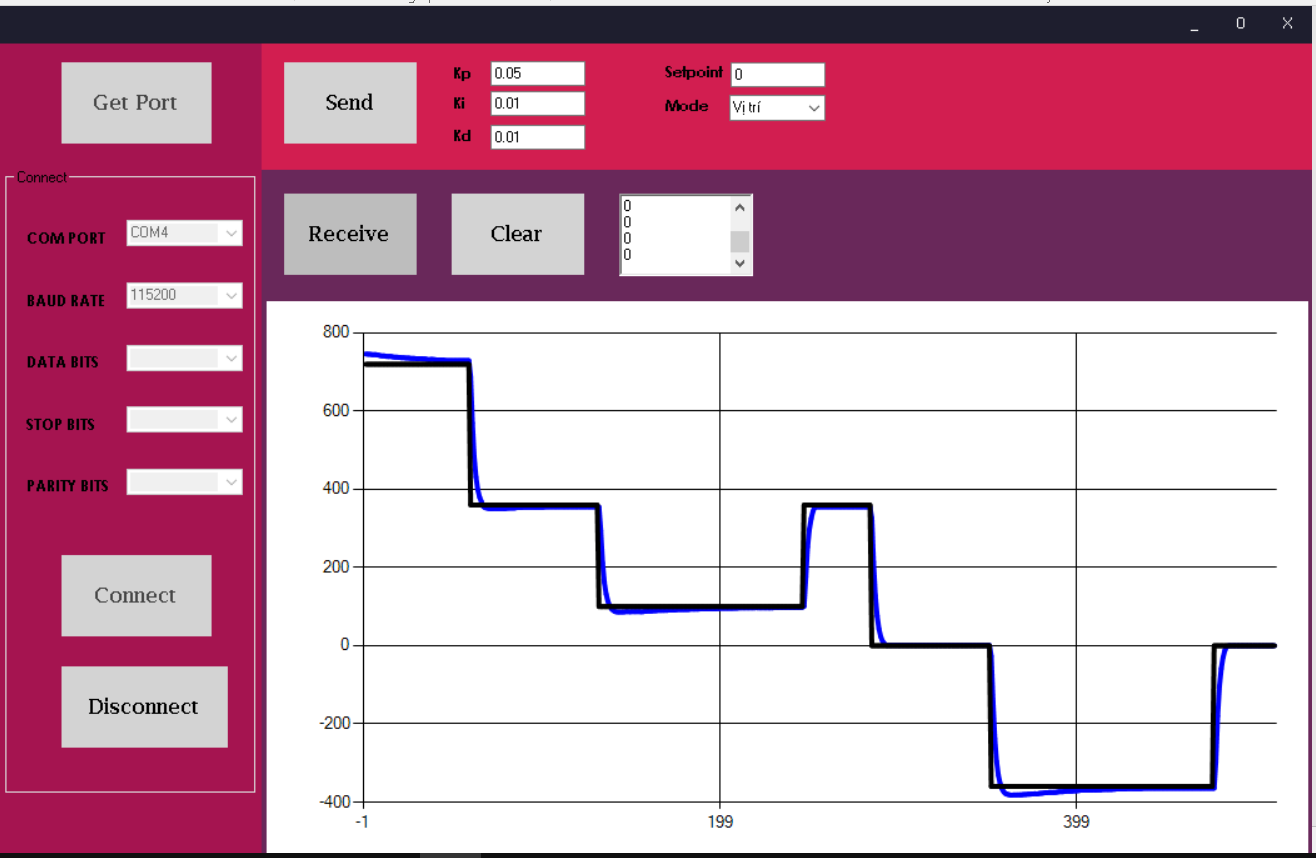
Hình 5.2: Kết quả điều khiển vị trí với tần số 1 kHz.

**Tần số 50 Hz:**



Hình 5.3: Kết quả điều khiển vị trí với tần số 50 Hz.

**Tần số 65 kHz:**



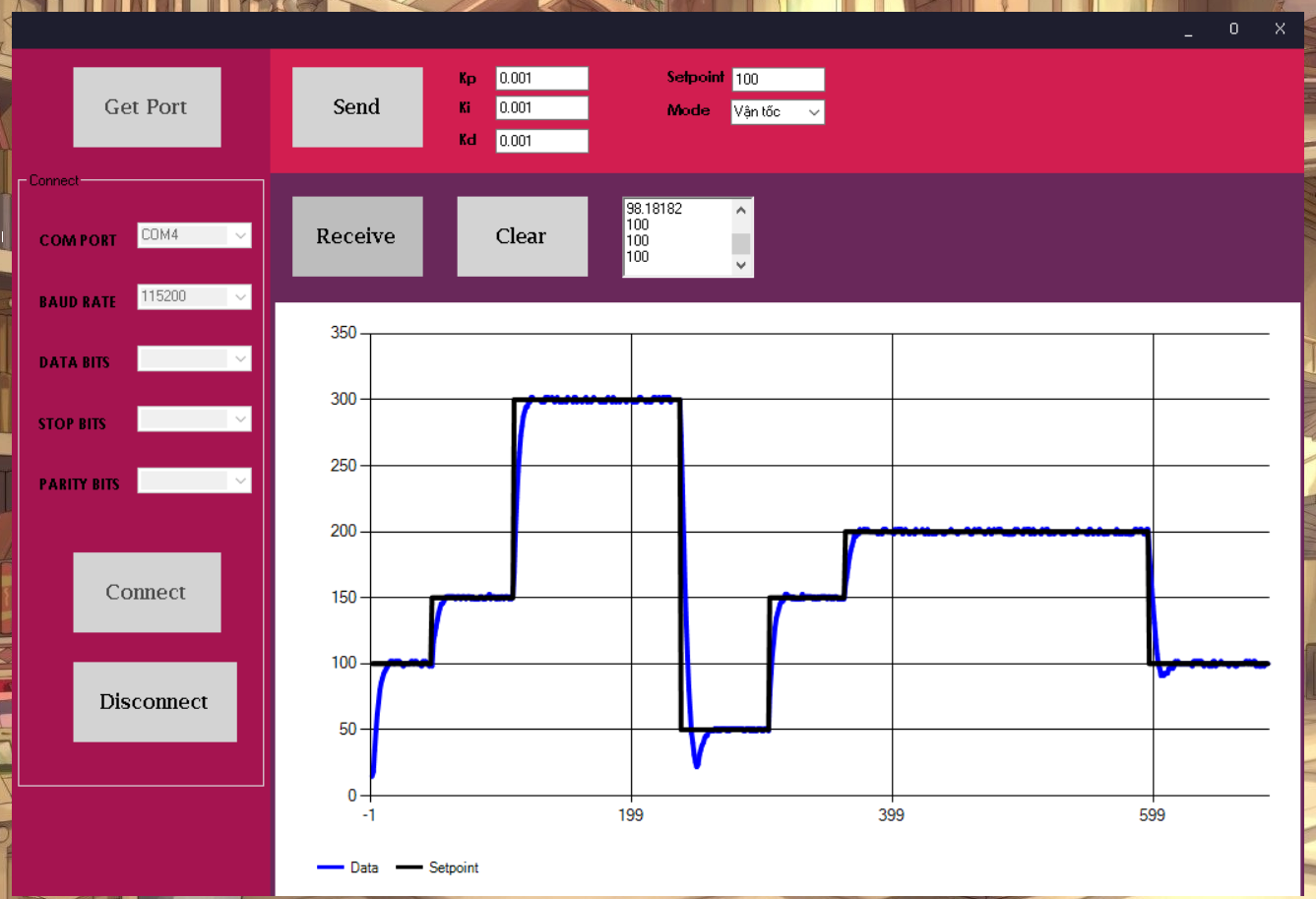
Hình 5.4: Kết quả điều khiển vị trí với tần số 65 kHz.

**Nhận xét:**

Kết quả điều khiển khá tốt, không bị dao động nhiều và sai số nhỏ. Tuy nhiên thời gian xác lập còn hơi dài. Tần số thay đổi không ảnh hưởng quá nhiều đến chất lượng hệ thống.

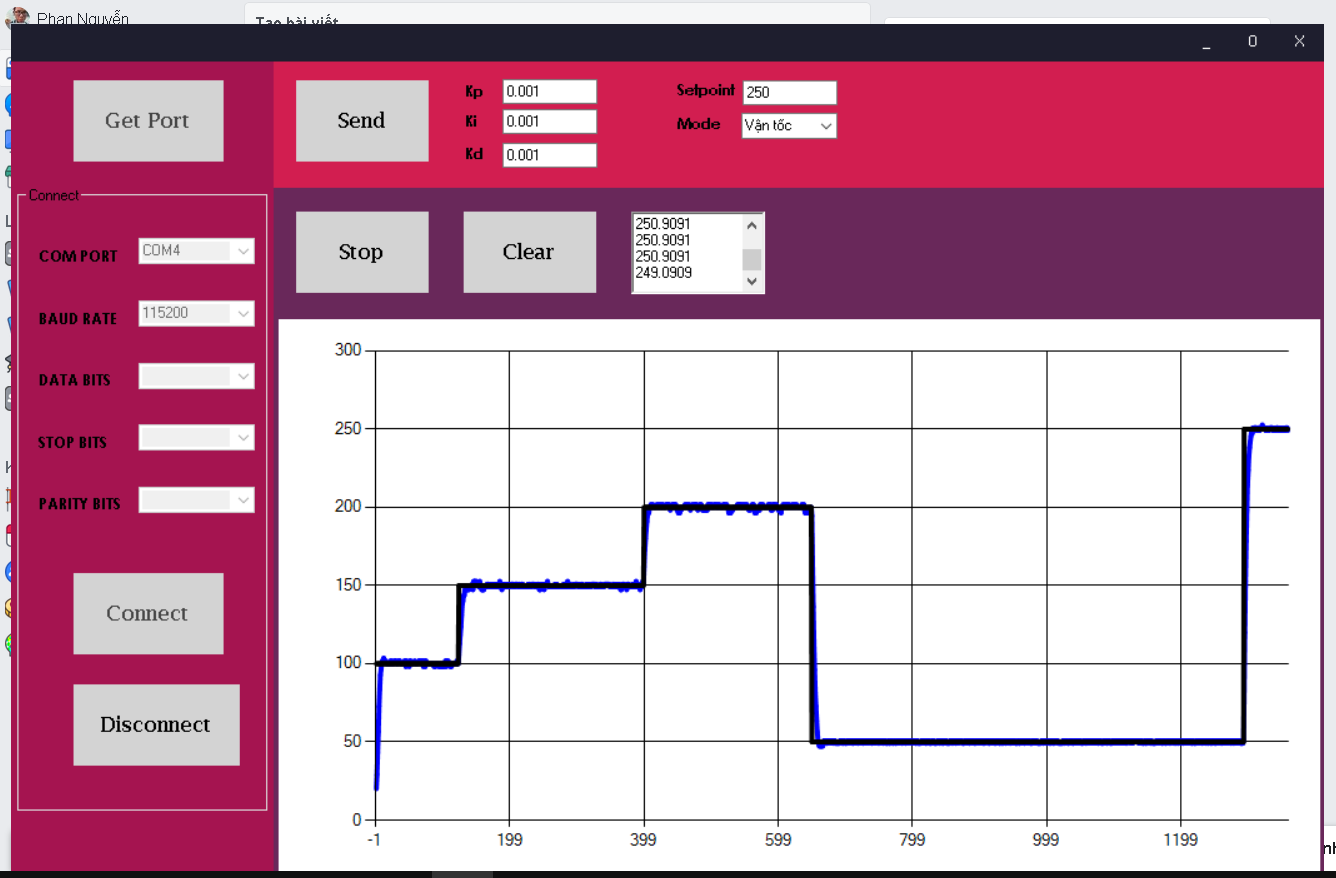
## Điều khiển vận tốc

**Tần số 10 kHz:**



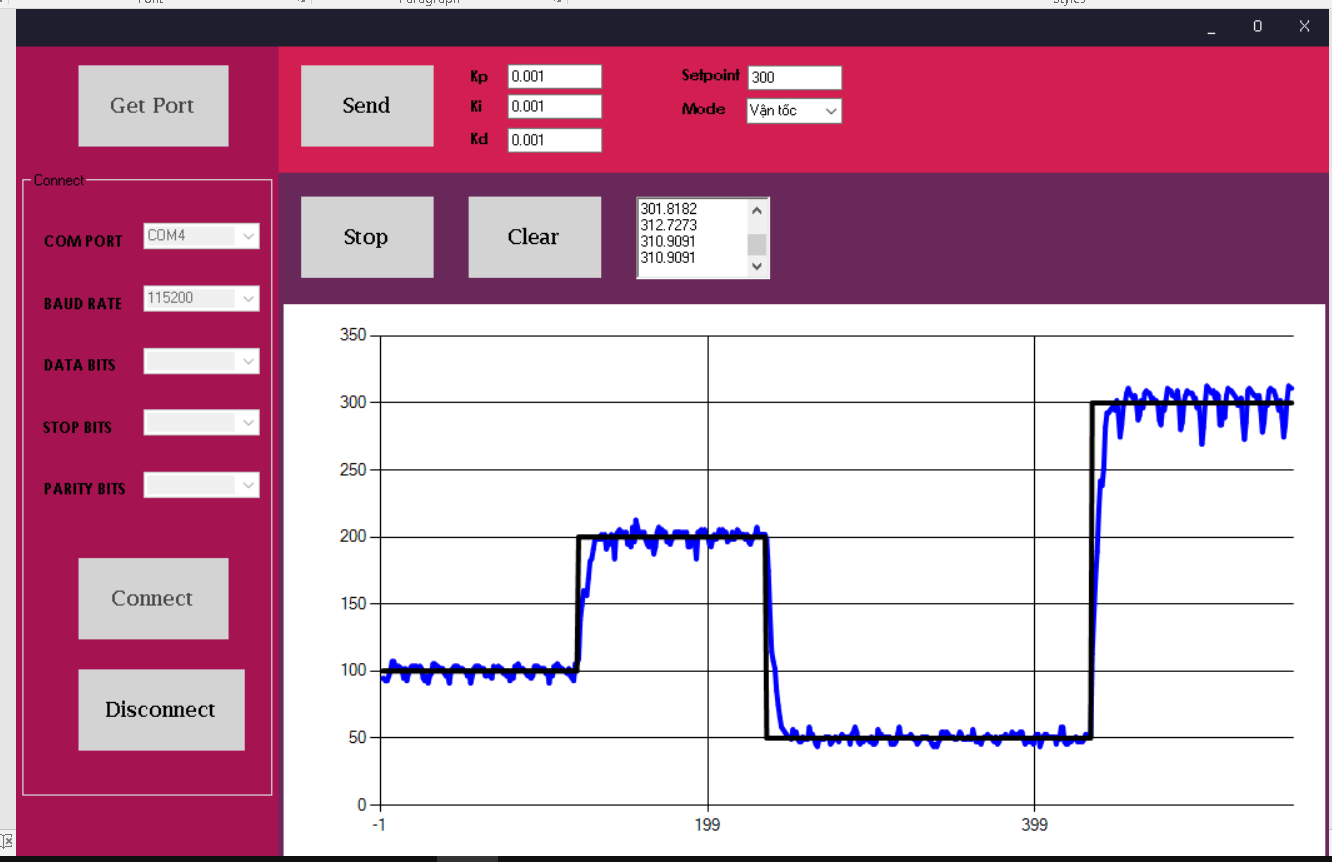
Hình 5.5: Kết quả điều khiển vận tốc với tần số 10 kHz.

**Tần số 1 kHz:**



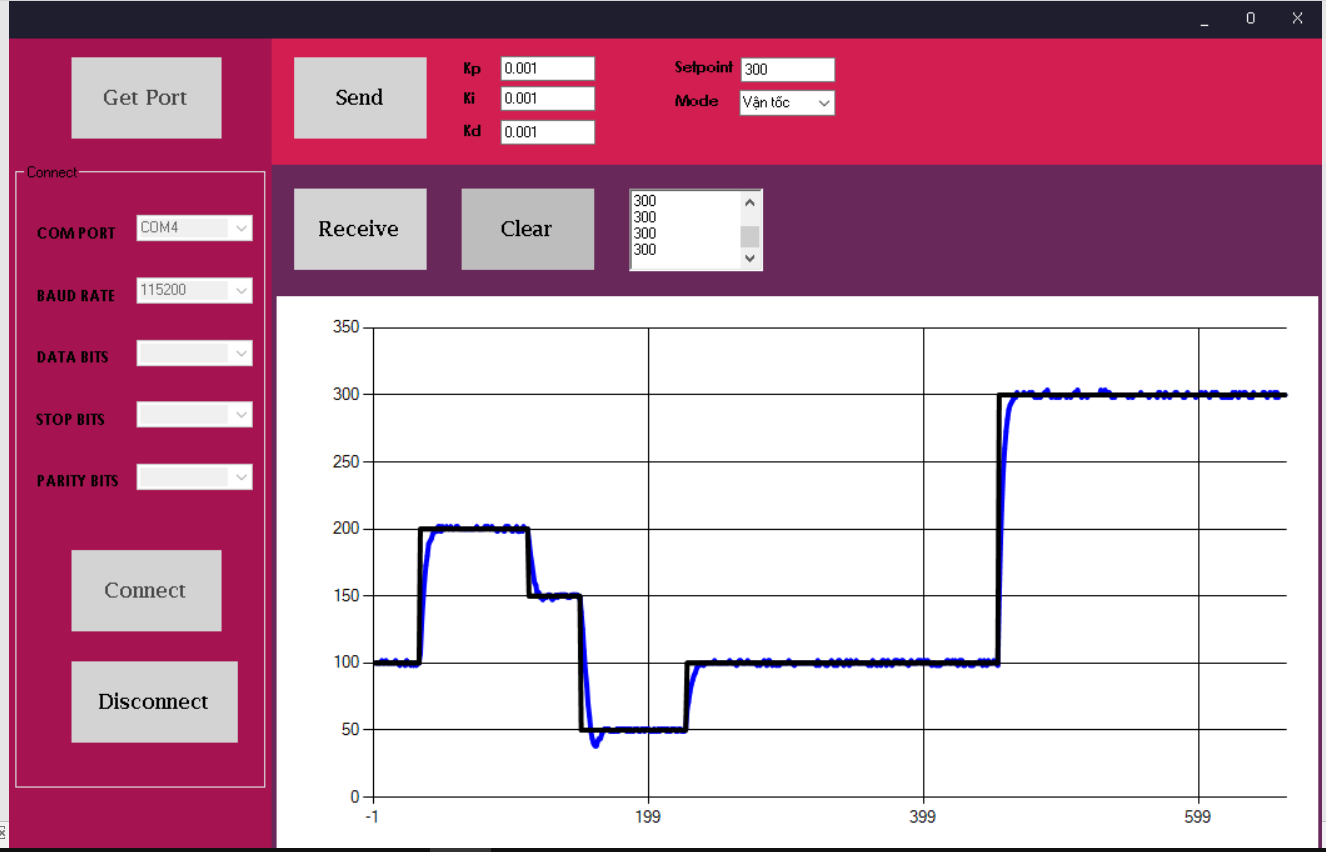
Hình 5.6: Kết quả điều khiển vận tốc với tần số 1 kHz.

**Tần số 50 Hz:**



Hình 5.7: Kết quả điều khiển vận tốc với tần số 50 Hz.

**Tần số 65 kHz:**



Hình 5.8: Kết quả điều khiển vận tốc với tần số 65 kHz.

**Nhận xét:**

Kết quả điều khiển bám theo vận tốc đã đặt từ trước, thời gian đáp ứng nhanh, đôi khi còn vọt lố và bị nhiễu khá nhiều. Tần số nhỏ thì sẽ khiến cho vận tốc bị dao dộng khá nhiều, khi qua một mức tần số thì chất lượng gần như không thay đổi.

# Kết luận và hướng phát triển

## Kết luận

Nhóm đã thiết kế thành công phần cứng và chương trình phần mềm cho đề tài. Kết quả chạy chương trình của nhóm là ổn định, tương đối chính xác với yêu cầu của đề tài.

## Hướng phát triển

Nhóm sử dụng PID bình thường để điều khiển động cơ DC, do đó tốn rất nhiều thời gian để tìm ra các thông số của PID. Hướng phát triển cho đề tài là sử dụng điều khiển PID mờ để điều khiển động cơ cho ra đáp ứng tốt hơn.