BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM

KHOA ĐIỆN – ĐIỆN TỬ

BÁO CÁO ĐỒ ÁN

ĐỀ TÀI:

ỔN ĐỊNH ĐỘNG CƠ DC SỬ DỤNG BIẾN TRỞ VÀ THUẬT TOÁN PID

GVHD: Ths.NGUYỄN ĐÌNH PHÚ

SVTH1: NGUYỄN ĐỨC QUÂN

MSSV: 20161360

SVTH2: NGUYỄN HỒNG QUANG

MSSV: 20161357

TP.HỒ CHÍ MINH, ngày tháng năm 2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM

KHOA ĐIỆN – ĐIỆN TỬ

BÁO CÁO ĐỒ ÁN

ĐỀ TÀI:

ỔN ĐỊNH ĐỘNG CƠ DC SỬ DỤNG BIẾN TRỞ VÀ THUẬT TOÁN PID

GVHD: Ths.NGUYỄN ĐÌNH PHÚ

SVTH1: NGUYỄN ĐỨC QUÂN

MSSV: 20161360

SVTH2: NGUYỄN HỒNG QUANG

MSSV: 20161357

TP.HỒ CHÍ MINH, ngày tháng năm 2023

LỜI CAM ĐOAM

Đề tài này là do nhóm tự thực hiện dựa vào một số tài liệu và không sao chép từ tài liệu hay công trình đã có trước đó. Nếu có sao chép thì nhóm hoàn toàn chịu trách nhiệm.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

Chương 1:

GIỚI THIỆU YÊU CẦU – GIỚI HẠN CỦA ĐỀ TÀI

1.1 Giới thiệu đề tài

Trong bối cảnh ngày càng phức tạp và đa dạng của công nghiệp và tự động hóa, động cơ DC (Direct Current) đóng vai trò quan trọng trong việc cung cấp nguồn năng lượng và thực hiện các tác vụ quan trọng. Điều khiển hiệu quả động cơ DC để đảm bảo sự ổn định và hiệu suất là một thách thức đối với nhiều ngành công nghiệp. Đồng thời, sự phát triển của công nghệ và các thuật toán điều khiển đã mở ra nhiều cơ hội để cải thiện hệ thống điều khiển động cơ DC. Đề tài ”Ổn Định Động Cơ DC Sử Dụng Biến Trở và Thuật Toán PID” ra đời với mục tiêu góp phần giải quyết những vấn đề hiện tại trong việc điều khiển động cơ DC. Đồ án này tập trung vào việc nghiên cứu và triển khai thuật toán PID (Proportional-Integral-Derivative) cùng với sự ứng dụng linh hoạt của biến trở để đạt được điều khiển chính xác và ổn định cho động cơ DC.

Những lý do quan trọng khiến cho việc nghiên cứu và phát triển hệ thống điều khiển động cơ DC sử dụng biến trở và thuật toán PID trở nên quan trọng và hấp dẫn:

- **Tối ưu hóa Hiệu suất:** Động cơ DC thường không hoạt động ở mức hiệu suất tối ưu mặc dù có tiềm năng. Sử dụng biến trở và thuật toán PID có thể giúp điều chỉnh động cơ để đạt được hiệu suất cao hơn, tiết kiệm năng lượng và giảm thiểu sự hao mòn của hệ thống.
- **Ổn Định Hệ Thống:** Việc điều khiển động cơ DC đôi khi gặp khó khăn trong việc duy trì sự ổn định. Sử dụng thuật toán PID có khả năng điều khiển tốt trong các tình huống biến đổi.
- **Ứng dụng Rộng Rãi:** Công nghệ điều khiển động cơ DC có ứng dụng đa dạng từ công nghiệp đến tự động hóa, từ xây dựng đến y tế. Vì vậy, việc nghiên cứu và phát triển hệ thống điều khiển hiệu quả có thể mang lại lợi ích lớn cho nhiều lĩnh vực.

Để thực hiện đồ án này, nhóm đã tìm hiểu các nghiên cứu trước đó liên quan đến điều khiển động cơ DC và áp dụng kiến thức này để phát triển một hệ thống điều khiển hoàn

thiện. Sự kết hợp giữa việc sử dụng biến trở và thuật toán PID có thể đem lại lợi ích to lớn và cung cấp giải pháp ổn định và hiệu quả cho việc điều khiển động cơ DC.

1.2 Mục đích đề tài

Mục tiêu của đồ án này là xây dựng một hệ thống điều khiển tốc độ động cơ DC bằng cách sử dụng thuật toán PID, kết hợp với đọc giá trị setpoint từ biến trở và đo tốc độ thực tế từ encoder. Cụ thể, các mục tiêu chính bao gồm:

- Phát triển một hệ thống điều khiển tốc độ cho động cơ DC sử dụng thuật toán PID (Proportional-Integral-Derivative).
- Đảm bảo rằng hệ thống có khả năng đọc giá trị đặt (setpoint - SP) từ biến trở và đo tốc độ thực tế (Process Variable - PV) từ encoder.
- Tính toán sai số (error) giữa SP và PV để định rõ sự chênh lệch và áp dụng thuật toán PID để điều khiển tốc độ động cơ.
- Giới hạn giá trị điều khiển(control variable - CV) để đảm bảo an toàn và hiệu suất tốt.
- Hiển thị thông tin SP và PV trên màn hình LCD và truy cập dữ liệu thông qua Serial Monitor.

Mục tiêu chính của độ án là xây dựng một hệ thống điều khiển tốc độ động cơ DC sử dụng PID, cung cấp khả năng điều khiển tốc độ một cách ổn định và hiệu quả, và theo dõi hiệu suất thông qua giao diện người dùng.

1.3 Phạm vi nghiên cứu

Đồ án này có phạm vi tập trung vào việc thiết kế và phát triển hệ thống điều khiển tốc độ cho một động cơ DC bằng cách sử dụng thuật toán PID. Cụ thể, phạm vi nghiên cứu sẽ bao gồm:

- **Kích thước và cấu trúc hệ thống:** Hệ thống điều khiển sẽ được triển khai và kiểm tra trên một động cơ DC cụ thể. Kích thước và cấu trúc của hệ thống điều khiển sẽ được xác định để đảm bảo tích hợp dễ dàng vào ứng dụng thực tế.
- **Thiết bị và linh kiện:** đồ án sẽ sử dụng các thành phần chính như động cơ DC, biến

trở, encoder, mô-đun PWM, màn hình LCD, và các linh kiện khác để xây dựng hệ thống điều khiển.

- **Hệ số điều chỉnh PID:** Các hệ số điều chỉnh Kp, Ki, và Kd của thuật toán PID sẽ được điều chỉnh để đảm bảo hiệu suất tốt nhất cho việc điều khiển tốc độ động cơ DC.
- **Phạm vi của setpoint (SP) và giới hạn Control Variable (CV):** Phạm vi giá trị setpoint (SP) và giới hạn giá trị Control Variable (CV) sẽ được xác định để đảm bảo an toàn và hiệu suất hệ thống.
- **Môi trường sử dụng:** Hệ thống điều khiển này sẽ được áp dụng trong môi trường thí nghiệm và chỉ sử dụng trong bối cảnh thí nghiệm, không dự kiến triển khai trong môi trường thực tế.

Phạm vi nghiên cứu này sẽ giúp xác định và giới hạn các yếu tố cụ thể và thông số kỹ thuật của hệ thống điều khiển động cơ DC, tạo ra sự rõ ràng và giúp đánh giá hiệu suất của đồ án dễ dàng hơn.

Chương 2:

CƠ SỞ LÝ THUYẾT

2.1 GIỚI THIỆU

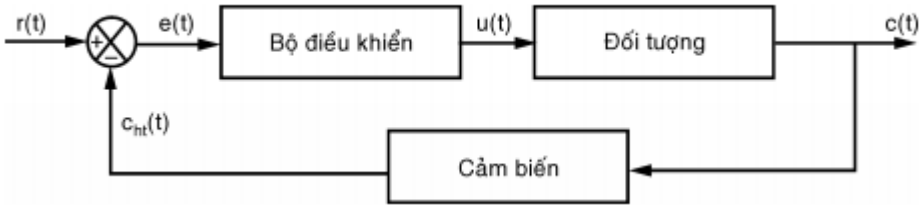
Lĩnh vực điều khiển tự động ngày càng phát triển, đặc biệt là điều khiển một cách chính xác, đã trở thành một phần không thể thiếu của nền công nghiệp hiện đại. Phần lớn các loại máy móc, thiết bị dân dụng hay trong công nghiệp sử dụng động cơ điện đều yêu cầu độ chính xác cao, tiết kiệm năng lượng, tuổi thọ và chu kì bảo dưỡng dài. Một trong những yêu cầu cần được đáp ứng là điều khiển được tốc độ động cơ điện ổn định, đáp ứng nhanh và vận hành trơn tru. .

2.2 NHỮNG KHÁI NIỆM CƠ BẢN

2.2.1 Khái niệm điều khiển

Điều khiển là quá trình thu thập thông tin, xử lý thông tin và tác động lên hệ thống để đáp ứng của hệ thống "gần" với mục đích định trước. Vậy "Tại sao cần phải điều khiển?". Có hai lý do chính là con người không thỏa mãn với đáp ứng của hệ thống hay muốn hệ thống hoạt động tăng độ chính xác, tăng năng suất, tăng hiệu quả kinh tế. Ví dụ trong lĩnh vực dân dụng, chúng ta cần điều chỉnh nhiệt độ và độ ẩm cho các căn hộ và các cao ốc tạo ra sự tiện nghi trong cuộc sống. Trong vận tải cần điều khiển các xe hay máy bay từ nơi này đến nơi khác một cách an toàn và chính xác. Trong công nghiệp, các quá trình sản xuất bao gồm vô số mục tiêu sản xuất thỏa mãn các đòi hỏi về sự an toàn, độ chính xác và hiệu quả kinh tế.

Trong những năm gần đây, các hệ thống điều khiển càng có vai trò quan trọng trong việc phát triển và sự tiến bộ của kỹ thuật công nghệ và 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. Dễ dàng tìm thấy hệ thống điều khiển máy công cụ, kỹ thuật không gian và hệ thống vũ khí, điều khiển máy tính, các hệ thống giao thông, hệ thống năng lượng, robot,...



Hình 2.1: Sơ đồ khối hệ thống điều khiển

Chú thích các ký hiệu viết tắt:

- **r(t) (reference input)**: tín hiệu vào, tín hiệu chuẩn
- **c(t) (controlled output)**: tín hiệu ra
- c_{ht}**(t)**: tín hiệu hồi tiếp
- **e(t) (error)**: sai số
- **u(t)** : tín hiệu điều khiển.

Để thực hiện được quá trình điều khiển như định nghĩa ở trên, một hệ thống điều khiển bắt buộc gồm có ba thành phần cơ bản là thiết bị đo lường (cảm biến), bộ điều khiển và đối tượng điều khiển. Thiết bị đo lường có chức năng thu thập thông tin, bộ điều khiển thực hiện chức năng xử lý thông tin, ra quyết định điều khiển và đối tượng điều khiển chịu sự tác động của tín hiệu điều khiển.

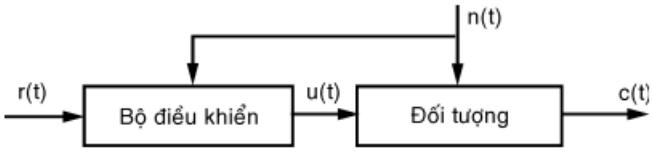
Các nguyên tắc điều khiển có thể xem là kim chỉ nam để thiết kế hệ thống điều khiển đạt chất lượng cao và có hiệu quả kinh tế nhất:

Nguyên tắc 1: *Nguyên tắc thông tin phản hồi*

Muốn quá trình điều khiển đạt chất lượng cao, trong hệ thống phải tồn tại hai dòng thông tin: một từ bộ điều khiển đến đối tượng và một từ đối tượng ngược về bộ điều khiển (dòng thông tin ngược lại gọi là hồi tiếp). Điều khiển không hồi tiếp (điều khiển vòng hở) không thể đạt chất lượng cao, nhất là khi có nhiễu.

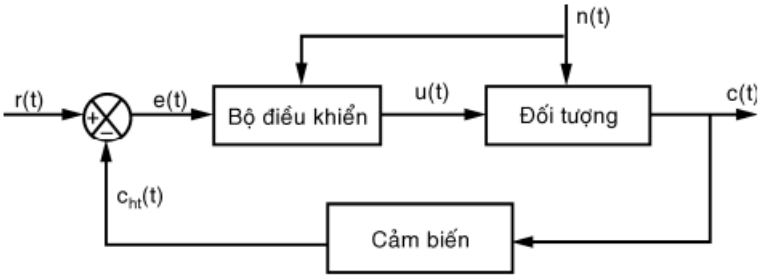
Các sơ đồ điều khiển dựa trên *nguyên tắc thông tin phản hồi* là:

Điều khiển bù nhiễu: là sơ đồ điều khiển theo nguyên tắc bù nhiễu để đạt đầu ra c(t) mong muốn mà không cần quan sát tín hiệu ra c(t). Về nguyên tắc, đối với hệ phức tạp thì điều khiển bù nhiễu không thể cho chất lượng tốt.



Hình 2.2: Điều khiển bù nhiễu

Điều khiển phối hợp: Các hệ thống điều khiển chất lượng cao thường phối hợp sơ đồ điều khiển bù nhiễu và điều khiển san bằng sai lệch



Hình 2.3: Điều khiển phối hợp

Nguyên tắc 2: *Nguyên tắc đa dạng tương xứng*

Muốn quá trình điều khiển có chất lượng thì sự đa dạng của bộ điều khiển phải tương xứng với sự đa dạng của đối tượng. Tính đa dạng của bộ điều khiển thể hiện ở khả năng thu thập thông tin, lưu trữ thông tin, truyền tin, phân tích xử lý, chọn quyết định,... Ý nghĩa của nguyên tắc này là cần thiết kế bộ điều khiển phù hợp với đối tượng.

Nguyên tắc 3: *Nguyên tắc bổ sung ngoài*

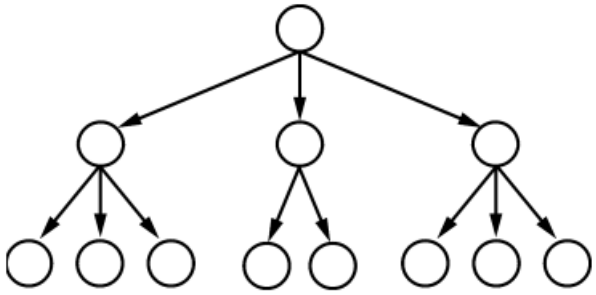
Một hệ thống luôn tồn tại và hoạt động trong môi trường cụ thể và có tác động qua lại chặt chẽ với môi trường đó. Nguyên tắc bổ sung ngoài thừa nhận có một đối tượng chưa biết tác động vào hệ thống và ta phải điều khiển cả hệ thống lẫn hộp đen. Ý nghĩa của nguyên tắc này là khi thiết kế hệ thống tự động, muốn hệ thống có chất lượng cao thì không thể bỏ qua nhiễu của môi trường tác động.

Nguyên tắc 4: *Nguyên tắc dự trữ*

Vì nguyên tắc 3 luôn coi thông tin chưa đầy đủ phải đề phòng bất trắc xảy ra và không được dùng toàn bộ lực lượng trong điều kiện bình thường. Vốn dự trữ không sử dụng, nhưng cần để đảm bảo cho hệ thống vận hành an toàn.

Nguyên tắc 5: *Nguyên tắc phân cấp*

Đối với hệ thống điều khiển phức tạp cần xây dựng nhiều lớp điều khiển bổ sung cho trung tâm. Cấu trúc phân cấp thường sử dụng là cấu trúc hình cây, ví dụ như hệ thống điều khiển giao thông đô thị hiện đại, hệ thống điều khiển dây chuyền sản xuất.



Hình 2.4: Nguyên tắc phân cấp

Nguyên tắc 6: *Nguyên tắc cân bằng nội*

Mõi hệ thống cần xây dựng cơ chế cân bằng nội để có khả năng tưh giải quyết những biến động xảy ra. Có nhiều cách để phân loại hệ thống điều khiển tùy theo mục đích của sự phân loại. Nếu căn cứ vào phương pháp phân tích và thiết kế có thể phân hệ thống điều khiển thành các loại tuyến tính và phi tuyến, biến đổi theo thời gian và bất biến theo thời gian; nếu căn cứ vào dạng tín hiệu trong hệ thống ta có hệ thống liên tục và hệ thống rời rạc; nếu căn cứ vào mục đích điều khiển ổn định hóa, điều khiển theo chương, điều khiển theo dõi,...

Hệ thống tuyến tính và hệ thống phi tuyến:

Hệ thống tuyến tính không tồn tại trong thực tế, vì tất cả các hệ thống vật lý đều là phi tuyến. Hệ thống điều khiển tuyến tính là mô hình lý tưởng để đơn giản hóa quá trình phân tích và thiết kế hệ thống. Khi giá trị của tín hiệu nhập vào hệ thống còn nằm trong giới hạn mà các phần tử còn hoạt động tuyến tính, thì hệ thống còn là tuyến tính. Nhưng khi giá trị của tín hiệu vào vượt ngoài vùng hoạt động tuyến tính của các phần tử và hệ thống

2.2.2 Điều khiển học

Điều khiển học (cybernetics) là khoa học về việc điều khiển, thu thập, truyền và xử lý thông tin, thường bao gồm liên hệ điều chỉnh ngược trong các cơ thể sống, trong máy móc và các tổ chức và các kết hợp của chúng (Ví dụ hệ thống kỹ thuật xã hội, các máy móc do máy tính điều khiển, chẳng hạn robot).

Từ khi thuật ngữ "điều khiển học" xuất hiện, các nhà khoa học đã tập trung vào những lỗi sai trong hệ thống truyền thông thông tin và kiểm soát phức tạp. Trong điều khiển học, khái niệm về liên lạc và kiểm soát có mối quan hệ mật thiết với nhau. Các thông tin liên quan đến chức năng và kiểm soát được trao đổi giữa các bộ phận của hệ thống cũng như giữa hệ thống với môi trường xung quanh. Mục đích của những thông tin này là đạt được điều kiện cân bằng hay nói cách khác là duy trì được khả năng hoạt động của hệ thống.

Trên thực tế, những hệ thống kiểm soát điều khiển tiêu thụ năng lượng ít hoặc không đáng kể khi vận hành thường cho hiệu quả cao. Điều này xảy ra khi chức năng cơ bản của nó là xử lý thông tin chứ không phải là truyền năng lượng. Việc kiểm soát điều khiển cần phải được phân biệt với sự khuyếc đại của dòng chảy bị tác động, điều này cũng có thể xảy ra khi có sự tồn tại của sự khuyếc đại. Sự kiểm soát thuộc điều khiển học thường được thực hiện cùng với những phương pháp đo lường hiệu quả. Có ba phương pháp đo lường thường được sử dụng như sau:

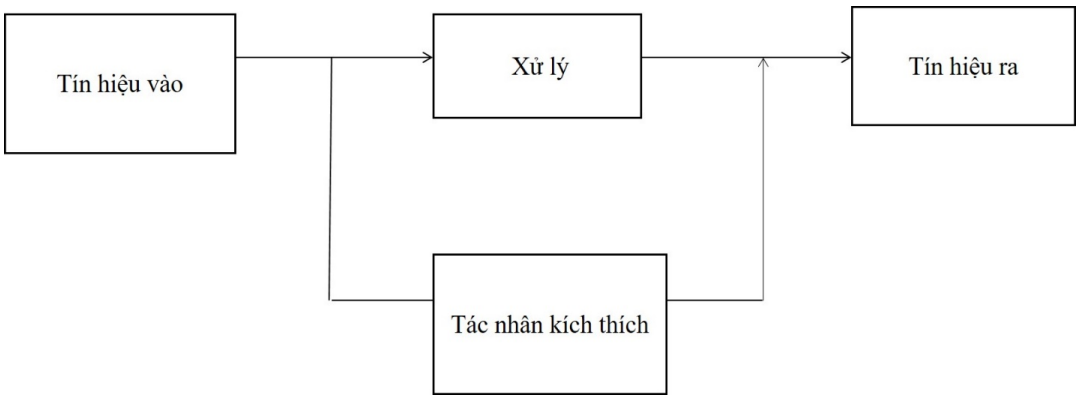
- **Đo lường hiệu quả chất lượng:** là thước mức độ hệ thống đạt được chuyển đổi dự kiến;
- **Đo lường hiệu quả sử dụng tài nguyên:** thước đo trong phạm vi mà hệ thống đạt được sự chuyển đổi như mong muốn mà chỉ sử dụng nguồn tài nguyên ở mức tối thiểu;
- **Hiệu quả công suất/ khả năng thực hiện:** đây là thước đo trong phạm vi mà ở đó hệ thống đóng góp vào mục đích của một hệ thống ở mức độ cao hơn mà hệ thống này là một tiểu hệ thống trực thuộc. Điều khiển có thể được định nghĩa là sự tác động có mục đích hướng tới mục tiêu đã được định sẵn liên quan tới việc so sánh liên tục những trạng thái hiện tại với mục đích trong tương lai.

Điều khiển là: Xử lý thông tin, Lên chương trình, Quyết định, Trao đổi thông tin (đối ứng). Một chương trình được lập được mã hóa hoặc chuẩn bị trước các thông tin điều khiển quá trình (hay hành vi) dẫn tới một kết quả đã được xác định trước.

Để hiểu rõ hơn về điều khiển học trước hết cần hiểu về hệ thống. Tuy nhiên, khi cần có những thông tin hữu ích về toàn bộ quá trình chuyển đổi của hệ thống, 5 thành phần sau đây có thể được tính toán:

- **Dữ liệu đầu vào:** đây là những thang đo có thể thay đổi được quan sát để tác động tới hoạt động của hệ thống;
- **Dữ liệu đầu ra:** đây là những thang đo được quan sát có ảnh hưởng tới mối quan hệ giữa hệ thống và môi trường xung quanh;
- **Chuỗi trạng thái:** đây là những thang đo nội bộ của hệ thống xác định mối quan hệ giữa nguyên liệu và sản phẩm;
- **Chức năng chuyển đổi trạng thái:**chức năng này sẽ quyết định trạng thái thay đổi như thế nào khi có quá nhiều dữ liệu được đưa vào hệ thống;
- **Chức năng của dữ liệu đầu ra:** chức năng này cho biết sản phẩm của hệ thống với một dữ liệu đầu vào cho sẵn trong một trạng thái nhất định.

Cơ chế điều chỉnh của hệ thống khép kín bao gồm điều khiển hồi tiếp (feedback) và điều khiển tiếp tới (feedforward). Điều khiển tiếp tới là hành động kiểm soát mang tính chất tiên liệu nhằm mục đích tạo ra trạng thái mong muốn hoặc được dự đoán trong tương lai. Quá trình này sử dụng tín hiệu vào chứ không sử dụng tín hiệu ra như trạng thái hồi tiếp âm. Trong một hệ thống tiếp tới, hoạt động của hệ thống được thiết lập lại theo như một số mô hình kết nối nguyên liệu hiện tại với kết quả được dự đoán trong tương lai. Chính vì vậy, những thay đổi trạng thái hiện tại được quyết định bởi trạng thái được dự đoán trong tương lai được tính toán theo một số mô hình nội bộ trên thê giới.



Hình 2.5: Hệ thống điều khiển

Hệ thống điều khiển nói chung với năm bước điều khiển cơ bản hoạt động theo tiến trình như sau:

1. Một trung tâm điều khiển giúp tạo nên những thang đo mục tiêu mong muống và công cụ để đạt được những mục tiêu đó.
2. Những quyết định về mục tiêu được chuyển thành những tín hiệu đầu vào hành động, những tín hiệu này dẫn đến những tác động nhất định vào trạng thái của hệ thống và môi trường xung quanh nó.
3. Thông tin về những tác động này được ghi lại và đưa phản hồi về cho trung tâm.
4. Trung tâm điều khiển kiểm tra lại trạng thái mới này của hệ thống so với các thang đo mục tiêu mong muốn để đo lường những lỗi sai hay độ sai lệch của phản hồi đầu ra đầu tiên.
5. Nếu những lỗi sai này làm cho hệ thống nằm ngoài những giới hạn được đặt ra bởi thang đo mục tiêu thì trung tâm điều khiển sẽ thực hiện hành động sửa chữa tín hiệu đầu ra.

Thành phần cơ bản đầu tiên của cơ chế điều khiển trong vòng tuần hoàn điều khiể cơ bản là bộ tiếp nhận (đôi khi được gọi là cảm biến hoặc máy dò), một dụng cụ đăng kí các kích thích khác nhau và sau khi chuyển đổi sang thông tin thì tiếp vận đơn vị điều khiển.

Khi so sánh giá trị của bộ tiếp nhận và các tiêu chuẩn cần thiết được lưu trữ trong bộ so sánh ta thấy rằng sự khác biệt cung cấp thông tin đính cính được thực hiện bởi bộ phận tác động. thông qua việc giám sát và hồi đáp lại bộ tiếp nhận, cơ chế tự điều khiển được hình thành. Hình dưới đây cho thấy cơ chế điều khiển xảy ra ở phía tín hiệu vào

và cơ chế cảm biến được đặt ở bên tín hiệu đầu ra. ở những hệ thống phức tạp hơn với hồi tiếp bậc ba, bộ điều khiển có thể bao gồm cả bộ phần đạt mục tiêu với các tiêu chuẩn tham khảo, bộ lọc và có thể là bộ thiết kế để xây dựng các mục tiêu và quy luật đưa ra quyết định của hệ thống.

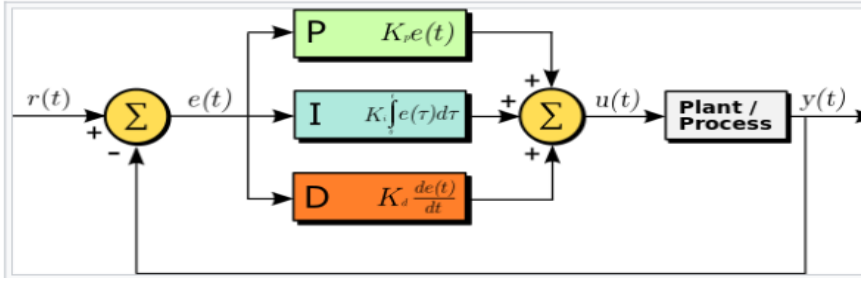
Hệ thống được kiểm soát phải có khả năng đọc được trạng thái của một biến quan trọng và xác định được liệu nó đang nằm trên nằm dưới hay nằm tại giá trị cho phép, việc này được thực hiện bởi bộ dò. Vì vậy, hệ thống thực hiện một số ưu tiên nhất định được sắp xếp bởi bộ lọc. Ngoài ra, hệ thống cũng cần tạo ra những phản hồi ưu tiên này trong trường hợp chúng không tồn tại. Việc này được tiến hành bởi các bộ phận kích hoạt và giám sát.

Ổn định phân chia đều là một phần tự nhiên của vòng đời hệ thống trong hệ thống của sinh vật. Đối với những hệ thống nhân tạo tiên tiến, vòng đời hệ thống có thể được chia ra thành những giai đoạn sau: hoạch định hệ thống, khảo sát hệ thống, thiết kế hệ thống, xây dựng hệ thống, đánh giá hệ thống, sử dụng hệ thống, và kết thúc vòng đời hệ thống.

2.3 LÝ THUYẾT BỘ ĐIỀU KHIỂN PID

PID là viết tắt của Proportional-Integral-Derivative, và đó là một thuật toán điều khiển hệ thống được sử dụng trong kỹ thuật và tự động hóa để điều chỉnh một quá trình hoặc hệ thống. Điều khiển PID là một hệ thống phản hồi liên tục tính toán và điều chỉnh đầu ra, hệ thống được điều khiển dựa trên sự khác biệt giữa một giá trị thiết lập mong muốn (mục tiêu) và một biến số quá trình đo lường (phản hồi).

Điểm đặc trưng của bộ điều khiển PID là khả năng sử dụng ba thành phần điều khiển tỷ lệ, tích phân và vi phân để điều chỉnh đầu ra của bộ điều khiển một cách chính xác và tối ưu.



Hình 2.6: sơ đồ khối của bộ diều khiển PID

Sơ đồ khối bên trên cho thấy cách nguyên tắc hoạt động của các thành phần này và cách chúng được áp dụng. Nó thể hiện một bộ điều khiển PID, liên tục tính toán giá trị sai số e(t) giữa giá trị thiết lập mong muốn SP = r(t) và giá trị hiện tại PV = y(t): e(t) = r(t) - y(t), và sửa đổi dựa trên các khâu tỉ lệ, tích phân và vi phân. Bộ điều khiển sẽ cố gắng làm giảm sai số theo thời gian bằng cách điều chỉnh biến số điều khiển u(t).

Luật điều khiển PID được định nghĩa:

$$u(t) = K_p(e(t) + \frac{1}{T_1}\int_0^t e(\tau)d\tau + T_D\frac{de(t)}{dt}) \tag{2.1}$$

Hàm truyền của bộ điều khiển PID:

$$G_{PID}(s) = \frac{U(s)}{E(s)} = K_P + K_I\frac{1}{s} + K_D s = K_P(1 + \frac{1}{T_i s} + T_D s) \tag{2.2}$$

Trong đó Kp, Ki, và Kd là các thông số hoặc hệ số điều chỉnh cho các khâu tỉ lệ, tích phân và vi phân tương ứng.

2.3.1 Khâu tỉ lệ (P)

Khâu tỉ lệ (P) đáp ứng tỷ lệ thuận với sai số hiện tại (error), tức là sai số giữa giá trị thiết lập mong muốn (setpoint - SP) và giá trị đã đo lường (process variable - PV) tại thời điểm hiện tại. Khi sai số tăng lên, khâu tỉ lệ sẽ có sự sửa đổi ngay lập tức đối với đầu ra của bộ điều khiển để làm giảm sai số này.

$$u(t) = K_P \,.\, e(t) \tag{2.3}$$

Hệ số K_P là yếu tố quyết định độ mạnh mẽ của khâu tỉ lệ. Giá trị K_P lớn có thể làm tăng độ nhạy của hệ thống, nhưng nếu quá lớn, nó có thể dẫn đến không ổn định và dao động.

Tác động của khâu tỉ lệ đơn giản là tín hiệu điều khiển tỉ lệ tuyến tính với sai lệch điều khiển. Thành phần P thiết lập một mối quan hệ tuyến tính, tức là nếu sai số gấp đôi, tín hiệu điều khiển P cũng tăng gấp đôi. Điều này tạo ra một phản ứng tỷ lệ với độ lớn của sai số, đẩy hệ thống đến gần giá trị mong muốn. Ban đầu, khi sai lệch lớn thì tín hiệu điều khiển lớn. Sai lệch giảm dần thì tín hiệu điều khiển cũng giảm dần. Khi sai lệch e(t) = 0 thì u(t) = 0. Một vấn đề là khi sai lệch đổi dấu thì tín hiệu điều khiển cũng đổi dấu. Thành phần P có ưu điểm là tác động nhanh và đơn giản. Hệ số tỉ lệ K_P càng lớn

thì tốc độ đáp ứng càng nhanh, do đó thành phần P có vai trò lớn trong giai đoạn đầu của quá trình quá độ.

Tuy nhiên, khi hệ số tỉ lệ K_P càng lớn thì sự thay đổi của tín hiệu điều khiển càng mạnh dẫn đến dao động lớn, đồng thời làm hệ thống nhạy cảm hơn với nhiễu đo. Hơn nữa, đối với đối tượng không có đặc tính tích phân thì sử dụng bộ P vẫn tồn tại sai lệch tĩnh.

2.3.2 Khâu tích phân (I)

Khâu tích phân tính toán tổng lũy của sai số (error) theo thời gian và áp dụng một sự điều chỉnh dựa trên tổng lũy này. Nó được sử dụng để loại bỏ sai số tồn đọng (steady-state error) trong hệ thống, là sai số duy trì trong trạng thái ổn định sau một thời gian dài.

$$u(t) = \int_0^t e(\tau)d\tau \tag{2.4}$$

Hệ số K_I quyết định tốc độ. Nếu K_I lớn, Hệ thống sẽ nhanh chóng tích phân lỗi, có thể làm thăng độ nhạy. Với Khâu tích phân, khi tồn tại một sai lệch điều khiển dương, luôn làm tăng tín hiệu điều khiển, và khi sai lệch là âm thì luôn làm giảm tín hiệu điều khiển, bất kể sai lệch đó là nhỏ hay lớn. Do đó, ở trạng thái xác lập, sai lệch bị triệt tiêu e(t) = 0. Đây cũng là ưu điểm của Khâu tích phân.

Nhược điểm của Khâu tích phân là do phải mất một khoảng thời gian để đợi e(t) về 0 nên đặc tính tác động của bộ điều khiển sẽ chậm hơn. Ngoài ra, Khâu tích phân đôi khi còn làm xấu đi đặc tính động học của hệ thống, thậm chí có thể làm mất ổn định. Người ta thuònng sử dụng bộ PI hoặc PID thay vì bộ I đơn thuần vừa để cải thiện tốc độ đáp ứng, vừa đảm bảo yêu cầu động học của hệ thống.

2.3.3 Khâu vi phân (D)

Khâu vi phân (D) đáp ứng dựa trên tốc độ biến đổi của sai số theo thời gian. Nó được sử dụng để dự đoán và ngăn chặn sự dao động hoặc overshoot (sự vượt quá giá trị thiết lập) trong hệ thống. Thành phần này giúp làm giảm tốc độ tăng của sai số và giúp hệ thống tiếp cận giá trị thiết lập một cách ổn định.

$$u(t) = \frac{de(t)}{dt} \tag{2.5}$$

Mục đích của Khâu vi phân là cải thiện sử ổn định của hệ kín. Do tính động học của quá trình, nên sẽ tồn tại khoảng thời gian trễ làm bộ điều khiển chậm so với sự thay đổi của sai lệch e(t) và đầu ra của quá trình. Khâu vi phân đóng vai trò dự đoán đầu ra của quá trình và đưa ra phản ứng thích hợp dựa trên chiều hướng và tốc độ thay đổi của sai lệch e(t), làm tăng tốc độ đáp ứng e(t).Khâu vi phân giúp làm giảm tốc độ tăng của sai số, ngăn chặn sự dao động và overshoot trong hệ thống.Nó đóng góp vào việc đảm bảo rằng hệ thống tiếp cận giá trị thiết lập một cách ổn định.

Để điều chỉnh khâu vi phân D, chúng ta cần điều chỉnh hệ số tỷ lệ vi phân (K_D) của bộ điều khiển PID.Tính chất chống đối với điều khiển ôn định của thành phần D giúp làm giảm nguy cơ dao động do phản ứng quá mạnh của thành phần P. Việc điều chỉnh hệ số K_D sẽ phụ thuộc vào đặc điểm cụ thể của hệ thống và ứng dụng. Thông thường, việc điều chỉnh Kd cần phải được thực hiện cùng với việc điều chỉnh K_P và K_I để đảm bảo hiệu suất điều khiển tối ưu và tính ổn định trong hệ thống điều khiển. Thành phần D giúp kiểm soát tốt hơn trong môi trường có nhiễu, đặc biệt là khi nhiễu có tần suất cao.

2.3.4 Hiệu chỉnh PID

Hiệu chỉnh tỉ lệ P (Proportional):

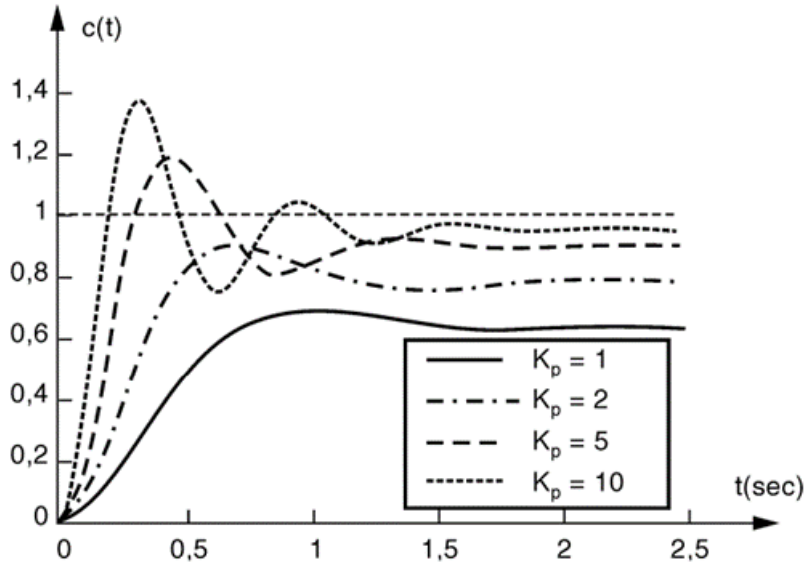
Hàm truyền:

$$G_c(s) = K_P \tag{2.6}$$

Nếu hệ số khuếch đại K_P càng lớn thì sai số xác lập càng nhỏ, tuy nhiên khi K_P tăng thì các cực của hệ thống nói chung có xu hướng di chuyển ra xa trục thực, điều đó có nghĩa là đáp ứng của hệ thống càng dao động, độ vọt lố (overshoot) càng cao. Nếu K_P tăng quá giá trị hệ số khuếch đại giới hạn thì hệ thống sẽ trở nên mất ổn định. Do đó nếu không thể có sai số của hệ thống bằng 0 thì cũng không thể tăng hệ số khuếch đại lên vô cùng.

Ví dụ 2.1. *Khảo sát ảnh hưởng của bộ điều khiển tỉ lệ.*

Xét hàm truyền của đối tượng là: $G(s) = \frac{10}{(s+2)(s+3)}$. Bộ điều khiển được sử dụng là bộ điều khiển tỉ lệ. Đường liền nét trong **hình 2.7** là đáp ứng của hệ thống khi chưa hiệu chỉnh K_P = 1. Theo hình về ta thấy khi tăng K_P thì sai số xác lập giảm, đồng thời độ vọt lố cũng tăng lên (các đường đứt nét).



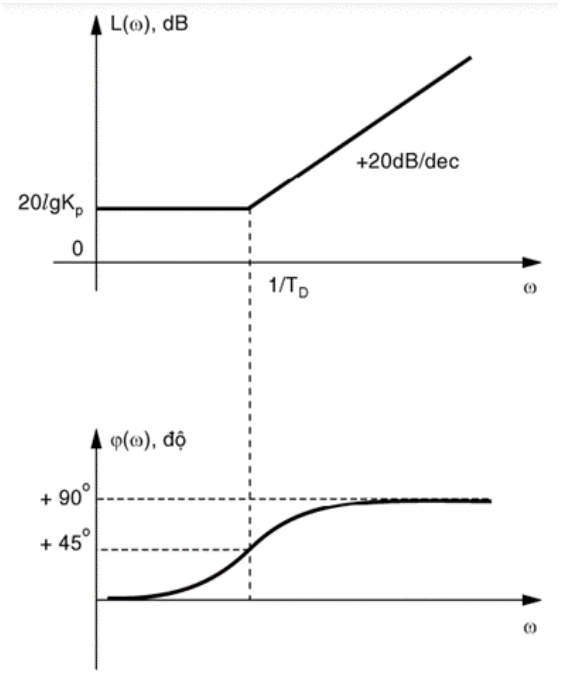
Hình 2.7: Đáp ứng nấc của hệ thống kín

Hiệu chỉnh vi phân tỉ lệ PD (Proportiona Derivativel):

Hàm truyền:
$$G_c(s) = K_P + K_D \,.\, s = K_P(1 + T_D \,.\, s) \tag{2.7}$$

Trong đó $K_d = K_P \,.\, T_D$, T_D được gọi là hằng vi phân của bộ điều khiển PD.

Đặc tính tần số:
$$G_c(j\omega) = K_P + K_D \,.\, j\omega = K_P(1 + T_D \,.\, j\omega) \tag{2.8}$$



Hình 2.8: Biểu đồ Bode của khâu hiệu chỉnh PD

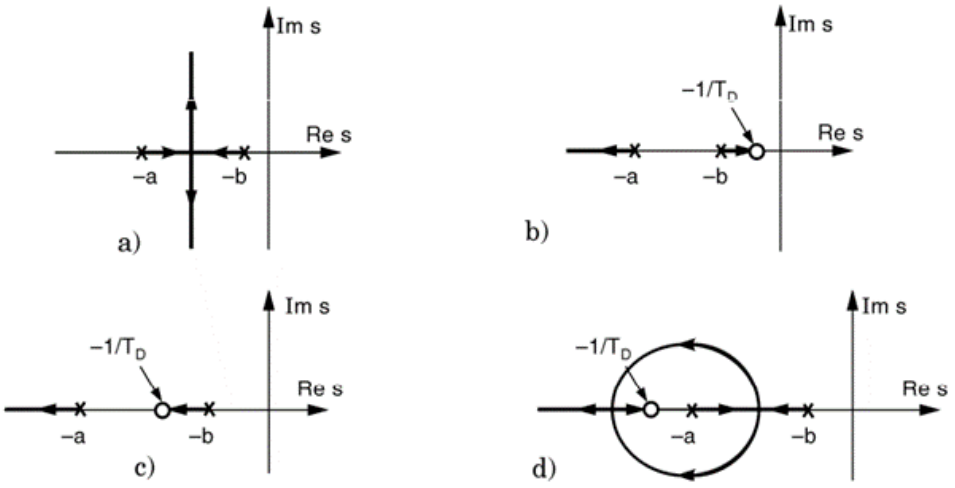
Dựa vào biểu đồ Bode của khâu hiệu chỉnh PD ta thấy khâu hiệu chỉnh PD là một trường hợp riêng của khâu hiệu chỉnh sớm pha, trong đó độ lệch pha cực đại giữa tín hiệu ra và tín hiệu vào là $\varphi_{max} = 90^\circ$, tương ứng với tần số $\omega_{max} = +\infty$. Khâu hiệu chỉnh PD có đặc điểm của khẩu hiệu chỉnh sớm pha, nghĩa là làm nhanh đáp ứng của hệ thống, giảm thời gian quá độ. Tuy nhiên do hệ số khuếch đại ở tần số cao của khâu hiệu chỉnh PD là vô cùng lớn nên khâu hiệu chỉnh PD làm cho hệ thống rất nhạy với nhiễu tần số cao.

Ví dụ 2.2. Khảo sát ảnh hưởng của bộ điều khiển vi phân tỉ lệ.

Xét hệ thống hiệu chỉnh nối tiếp có sơ đồ khối như hình 6.1, trong đó hàm truyền của đối tượng là: $G(s) = \frac{K}{(s+a)(s+b)}$ (a > b > 0). Bộ điều khiển được sử dụng là bộ điều khiển vi phân tỉ lệ. Phương trình đặc tính của hệ thống sau khi hiệu chỉnh là:

$$1 + K_P(1 + T_D s)\frac{K}{(s+a)(s+b)} = 0 \tag{2.9}$$

Ảnh hưởng đặc trưng của khâu PD quyết định bởi thời hằng vi phân T_D (cũng chính là vị trí zero $-1/T_D$ trên độ lợi hay tần số gãy $1/T_D$ trên đặc tính tần số). Tùy theo giá trị của T_D mà độ lợi của hệ thống sau khi hiệu chỉnh có thể có các dạng như **hình 2.9**.

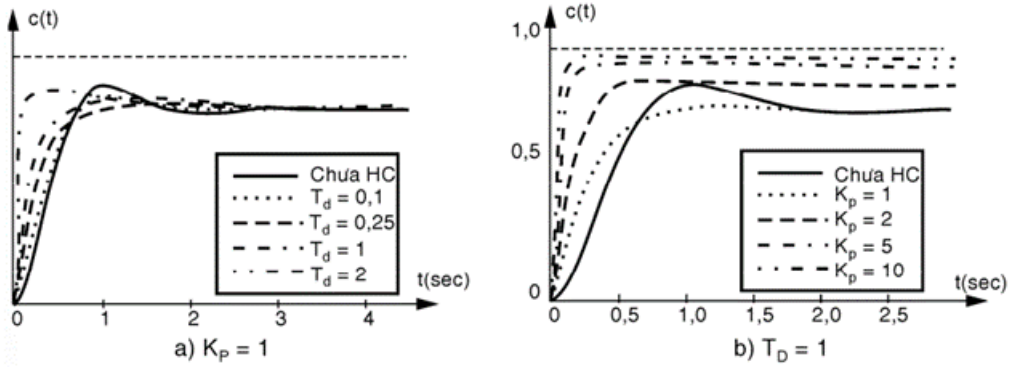


Hình 2.9: Sự thay đổi dạng độ lợi khi thêm khẩu hiệu chỉnh PD vào hệ thống
a)Chưa hiệu chỉnh; b) Đã hiệu chỉnh (0 < 1 / T_D < b);
c) Đã hiệu chỉnh (b < 1 / T_D < a) ;d) Đã hiệu chỉnh (1 / T_D > a)

Ta thấy nếu 0 < 1 / T_D < a thì độ lợi của hệ thống sau khi hiệu chỉnh nằm hoàn toàn trên trục thực (**hình 2.9b** và **2.9c**), do đó đáp ứng của hệ thống hoàn toàn không có dao động. Nếu 1 / T_D > a thì tùy giá trị của K_P mà hệ thống có thể có nghiệm phức, tuy nhiên

nghiệm phức này gần trục thực hơn so với trục ảo (nghĩa là ξ >0 707,), do đó độ vọt lố của hệ thống thấp hơn so với chưa hiệu chỉnh.

Hình 2.9a trình bày đáp ứng quá độ của hệ thống khi thay đổi giá trị T_D và giữ hệ số K_P bằng hằng số. Ta thấy T_D càng lớn thì đáp ứng càng nhanh, thời gian lên càng ngắn. Tuy nhiên nếu thời gian lên nhanh quá thì sẽ dẫn đến vọt lố mặc dù đáp ứng không có dao động. Khi đã xác định được T_D thì ảnh hưởng của K_P tương tự như ảnh hưởng của khâu khuếch đại, nghĩa là nếu K_P càng tăng (nhưng phải nhỏ hơn K_{gh}) thì sai số xác lập càng giảm, tuy nhiên sai số xác lập lúc nào cũng khác 0. Mặt khác trong trường hợp hệ thống đang khảo sát, khi K_P càng tăng thì độ lợi càng rời xa trục ảo nên thời gian đáp ứng cũng nhanh lên. Tuy nhiên ảnh hưởng này không phải là ảnh hưởng đặc trưng của khâu PD.



Hình 2.10: Ảnh hưởng của khâu hiệu chỉnh PD đến hệ thống

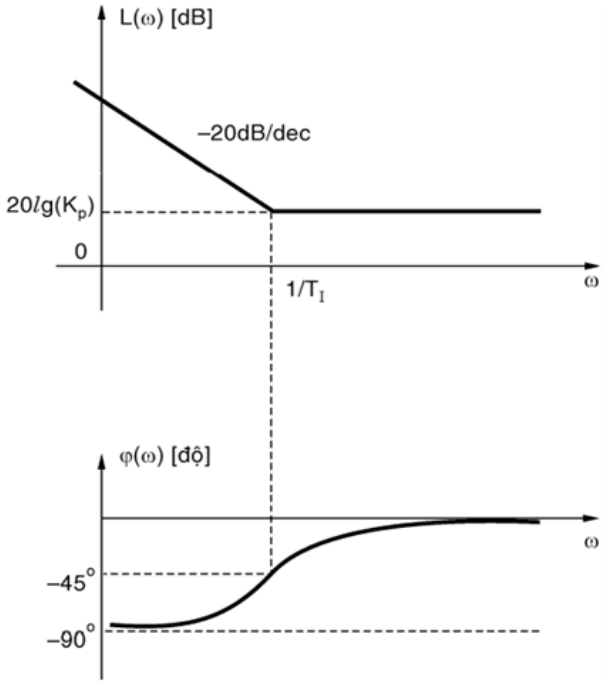
Hiệu chỉnh tích phân tỉ lệ PI (Proportional Integral):

Hàm truyền: $$G_c(s) = K_P + \frac{K_I}{s} = K_P(1 + \frac{1}{T_I s}) \quad (2.10)$$

trong đó $K_I = \frac{K_P}{T_I}$, T_I được gọi là thời hằng tích phân của bộ điều khiển PI.

Đặc tính tần số: $$G_c j\omega) = K_P(1 + \frac{1}{T_I j\omega} \quad (2.11)$$

Mắc nối tiếp khâu hiệu chỉnh PI với hàm truyền của đối tượng tương đương với việc thêm vào hệ thống một zero tại vị trí $-1/T_I$ và một cực tại góc tọa độ, điều này làm cho độ lợi của hệ thống sau khi hiệu chỉnh bị đẩy về phía phải mặt phẳng phức, nên hệ thống kém ổn định hơn.

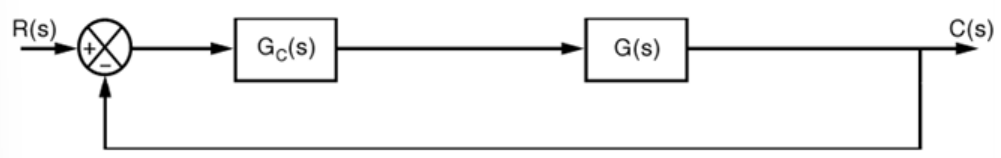


Hình 2.11: Biểu đồ Bode của khẩu hiệu chỉnh PI

Hình 2.11 là biểu đồ Bode của khẩu hiệu chỉnh PI. Dựa vào biểu đồ Bode của khâu hiệu chỉnh PI ta thấy khâu hiệu chỉnh PI là một trường hợp riêng của khâu hiệu chỉnh trễ pha, trong đó độ lệch pha cực tiểu giữa tín hiệu ra và tín hiệu vào là φ_{min} = -90° , tương ứng với tần số $\omega_{min} = 0$. Khâu hiệu chỉnh PI có đặc điểm của khâu hiệu chỉnh trễ pha, nghĩa là làm chậm đáp ứng quá độ, tăng độ vọt lố, giảm sai số xác lập. Do hệ số khuếch đại của khâu PI bằng vô cùng tại tần số bằng 0 nên khâu hiệu chỉnh PI làm cho sai số đối với tín hiệu vào là hàm nấc của hệ thống không có khâu vi phân lý tưởng bằng 0 (hệ vô sai bậc một). Ngoài ra do khẩu PI là một bộ lọc thông thấp nên nó còn có tác dụng triệt tiêu nhiễu tần số cao tác động vào hệ thống.

Ví dụ 2.3. Khảo sát ảnh hưởng của bộ điều khiển tích phân tỉ lệ.

Xét hệ thống hiệu chỉnh nối tiếp có sơ đồ khối như hình dưới:

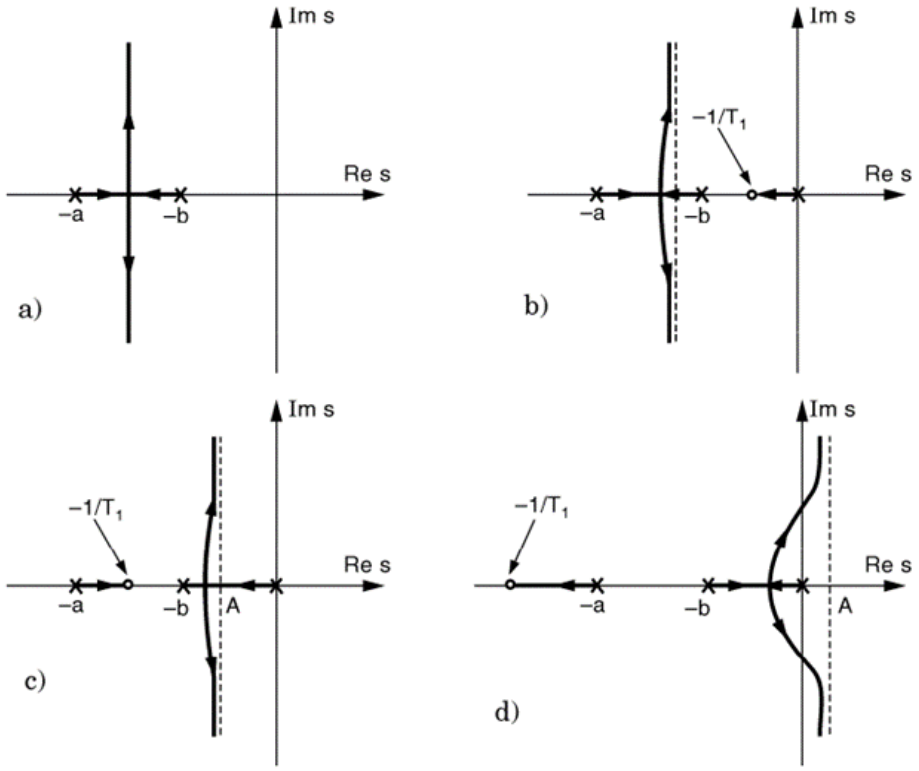


Hình 2.12: Hệ thống hiệu chỉnh nối tiếp

Trong đó hàm truyền của đối tượng là: $G(s) = \frac{K}{(s+a)(s+b)}$(a>b>0). Bộ điều khiển được sử dụng là bộ điều khiển tích phân tỉ lệ. Phương trình đặc tính của hệ thống sau khi hiệu chỉnh là:

$$1 + K_P . (\frac{T_I s + z}{T_I s} \frac{K}{(s+a)(s+b)}) \tag{2.12}$$

Ảnh hưởng đặc trưng của khâu PI quyết định bởi thời hằng tích phân T_I (cũng chính là vị trí zero $-1/T_I$ trên độ lợi hay tần số gãy $1/T_I$ trên đặc tính tần số). Tùy theo giá trị của T_I mà độ lợi của hệ thống sau khi hiệu chỉnh có thể có các dạng như **hình 2.12**.

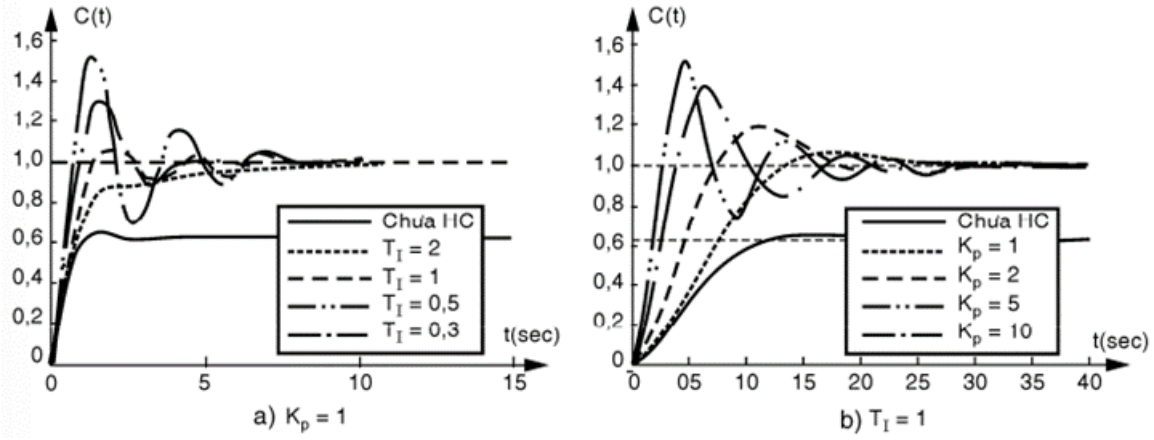


Hình 2.13: Sự thay đổi dạng độ lợi khi thêm khẩu hiệu chỉnh PI vào hệ thống
a) Chưa hiệu chỉnh; b) Đã hiệu chỉnh (0 <1/T_I<b)
c) Đã hiệu chỉnh (b<1/T_I<a); d) Đã hiệu chỉnh (1/T_I>2)

Hình 2.13 minh họa đáp ứng quá độ của hệ thống khi thay đổi thông số của bộ điều khiển PI. Ở **hình 2.13a** ta thấy khi càng giảm thời hằng tích phân T_I thì độ vọt lố của hệ thống càng cao, hệ thống càng chậm xác lập. Từ đây ta rút ra kết luận khi thiết kế khâu hiệu chỉnh PI nên chọn zero $-1/T_I$ nằm gần gốc tọa độ để thời hằng tích phân T_I có giá trị lớn nhằm hạn chế độ vọt lố. Khi giữ T_I bằng hằng số thì ảnh hưởng của K_P đến chất lượng của hệ thống chính là ảnh hưởng của khâu khuếch đại, K_P càng tăng thì độ vọt lố càng tăng, tuy nhiên thời gian quá độ gần như không đổi (**H.2.13b**). Nếu K_P vượt quá

giá trị hệ số khuếch đại giới hạn thì hệ thống trở nên mất ổn định.

Ta thấy khâu hiệu chỉnh PI làm cho sai số xác lập của hệ thống đối với tín hiệu vào là hàm nấc bằng 0. Tuy nhiên khâu hiệu chỉnh PI làm cho hệ thống kém ổn định. Ta có thể kiểm chứng được điều này bằng cách phân tích sự thay đổi dạng độ lợi của hệ thống sau khi hiệu chỉnh. Theo công khi $1/T_I$ càng tăng thì độ lợi của hệ thống càng di chuyển về phía phải mặt phẳng phức(**H.2.12b,c**), hệ thống càng kém ổn định. Khi $1/T_I$ đủ lớn thỏa điều kiện TabI $1/T_I$ > a+b thì độ lợi có đoạn nằm bên phải mặt phẳng phức **(H.2.12d**), hệ thống không ổn định nếu hệ số khuếch đại của hệ thống lớn hơn giá trị K_{gh}.



Hình 2.14: Ảnh hưởng của khâu hiệu chỉnh PI đến hệ thống

Hiệu chỉnh vi tích phân tỉ lệ PID (Proportional Integral Derivative):

Hàm truyền:
$$G_c(s) = K_P + \frac{K_I}{s} + K_D\, s \tag{2.13}$$

Có thể xem khâu hiệu chỉnh PID gồm một khẩu PI mắc nối tiếp với một khâu PD.

$$G_c(s) = K_{P1}\,.\,(1 + \frac{1}{(T_{I1}\, s)(1 + T_{D2}\, s)}) \tag{2.14}$$

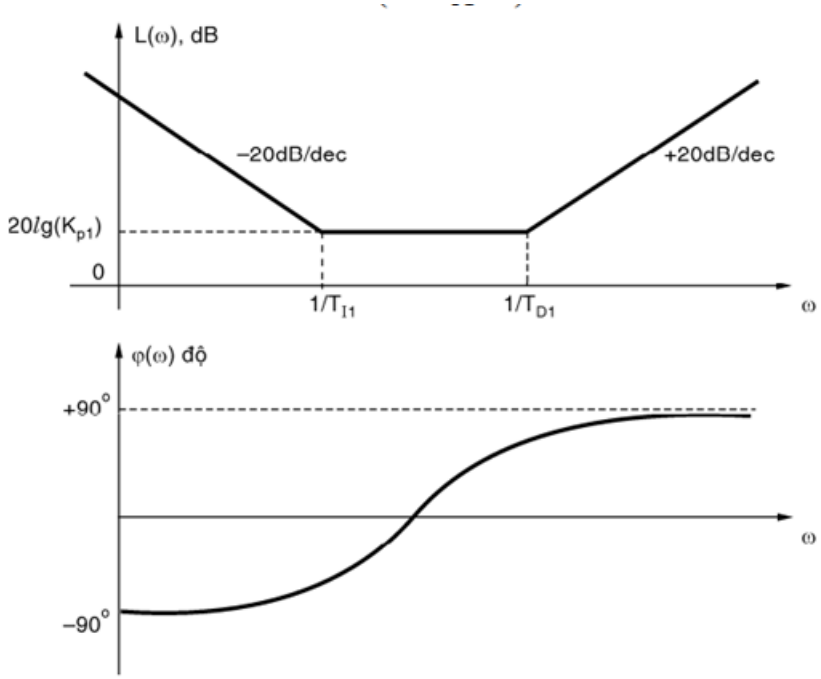
trong đó $T_I1 > T_D2$. Dễ dàng suy ra được mối quan hệ giữa các hệ số trong hai cách biểu diễn (2.13) và (2.14) như sau:

$$K_P = K_{P1}\,.\,(1 + \frac{T_{D2}}{T_{I1}}) \tag{2.15}$$

$$K_I = \frac{K_{P1}}{T_{I1}} \tag{2.16}$$

$$K_D = K_{P1} \, . \, T_{D2} \tag{2.17}$$

Đặc tính tần số : $$G_c(j\omega) = K_{P1}(1 + \frac{1}{T_{I1}j\omega})(1 + T_{D2}j\omega) \tag{2.18}$$



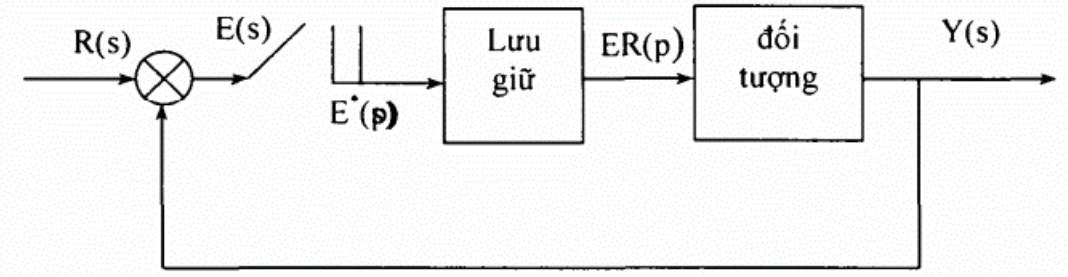
Hình 2.15: Biểu đồ Bode của khâu hiệu chỉnh PID

Khâu hiệu chỉnh PID là một trường hợp riêng của hiệu chỉnh sớm trễ pha, trong đó độ lệch pha cực tiểu giữa tín hiệu ra và tín hiệu vào là $\varphi_{min} = -90°$, tương ứng với tần số $\omega_{min} = 0°$; độ lệch pha cực đại giữa tín hiệu ra và tín hiệu vào là $\varphi_m in = 90°$, tương ứng với tần số $\omega_{max} = \infty$.

Do khâu hiệu chỉnh PID có thể xem là khâu PI mắc nối tiếp với khẩu PD nên nó có các ưu điểm của khâu PI và PD. Nghĩa là khâu hiệu chỉnh PID cải thiện đáp ứng quá độ (giảm vọt lố, giảm thời gian quá độ) và giảm sai số xác lập (nếu đối tượng không có khâu vi phân lý tưởng thì sai số xác lập đối với tín hiệu vào là hàm nấc bằng 0). Chúng ta vừa khảo sát xong ảnh hưởng của các khẩu hiệu chỉnh nối tiếp thường dùng đến chất lượng của hệ thống, mỗi khâu hiệu chỉnh có những ưu điểm cũng như khuyết điểm riêng. Do vậy cần phải hiểu rõ đặc điểm của từng khâu hiệu chỉnh chúng ta mới có thể sử dụng linh hoạt và hiệu quả được. Tùy theo đặc điểm của từng đối tượng điều khiển cụ thể và yêu cầu chất lượng mong muốn mà chúng ta phải sử dụng khâu hiệu chỉnh thích hợp. Khi đã

xác định được khâu hiệu chỉnh cần dùng thì vấn đề còn lại là xác định thông số của nó.

2.4 Khâu lưu giữ dữ liệu



Hình 2.16: Sơ đồ khâu lưu giữ dữ liệu

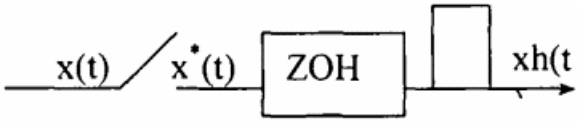
Nhiệm vụ của khâu lưu giữ là xây dựng lại hàm đã được lấy mẫu thành một tín hiệu liên tục dựa vào các hàm lấy mẫu trước đó. Trong hệ thống điều khiển số, khâu lưu giữ thường được thực hiện ngay sau bộ lấy mẫu. Khâu lưu giữ được chia thành hai loại:

- **Lưu giữ bậc không (Zero Hold Order - ZOH):** với ZOH tín hiệu được phục hồi chỉ phụ thuộc vào hàm đã được lấy mẫu tại thời điểm bắt đầu của chu kỳ lấy mẫu. Lưu giữ ZOH có thể coi tương tự như máy khoá điện tử, nó duy trì mức điện áp đầu ra bằng biên độ xung đầu vào và sau đó tự lặp lại khi có xung mới đặt vào.
- **Lưu giữ bậc một (First Hold Order - FOH):** tín hiệu được khôi phục lại phụ thuộc vào mẫu trước đó.

Thông thường trong điều khiển số thực tế người ta không sử dụng khâu ngoại suy dữ liệu bậc một, vì chúng tạo ra sự quá chậm pha trong hệ thống điều khiển có hồi tiếp. Mặt khác làm tăng ảnh hưởng của nhiễu tăng độ phức tạp và giá thành sản phẩm.

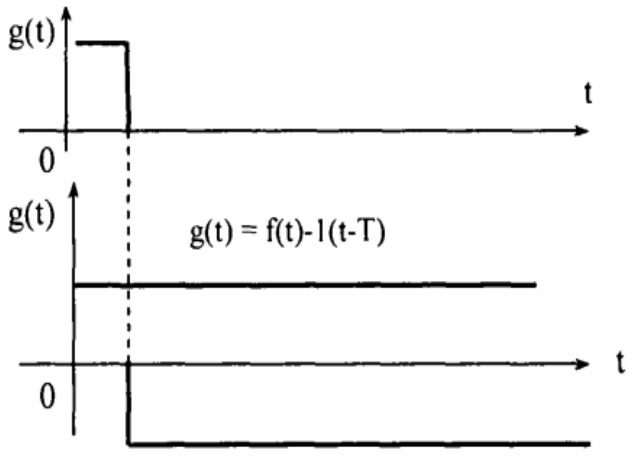
2.4.1 Khâu lưu giữ bậc không (Zero Order Hold - ZOH)

Đầu và của khâu ZOH là xung Dirac, đầu ra là tương tự



Hình 2.17: Sơ đồ khâu lưu giữ bậc không

Để xây dựng hàm truyền của khâu ZOH ta dựa vào các ứng dụng. Đáp ứng xung của ZOH là xếp chồng của hai hàm nhảy, một hàm dương tác động tại t = 0 và hàm âm tác động tại t = T (T là chu kỳ lấy mẫu).



Hình 2.18: Sơ đồ đáp ứng xung của khâu ZOH

Để thấy được ảnh hưởng của ZOH trong hệ thống điều khiển có hồi tiếp, ta hãy vẽ đáp ứng tần số của nó.:

$$G(s) = \frac{1}{s}\left(1 - e^{-TS}\right) \tag{2.19}$$

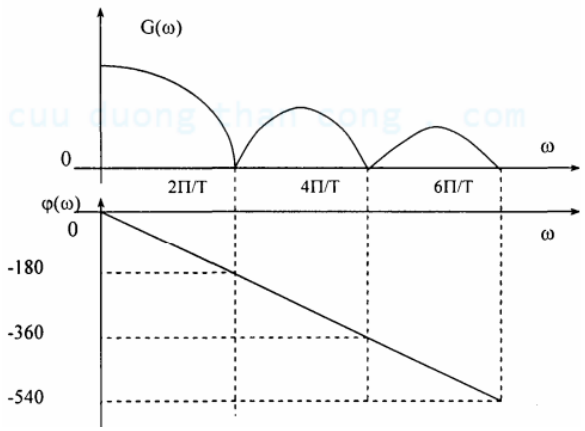
Thay $s = j\omega$, ta có:

$$\begin{aligned} G(\omega j) &= \tfrac{1}{j\omega}\left(1 - e^{-j\omega T}\right) \\ &= \tfrac{e^{-j\omega T/2}\left(e^{j\omega T/2} - e^{-j\omega T/2}\right)}{j\omega} \\ &= \tfrac{2}{\omega}\, e^{-j\omega T/2}\, \tfrac{e^{j\omega T/2} - e^{-j\omega T/2}}{2j} \\ &= \tfrac{2}{\omega}\, e^{-j\omega T/2}\, sin(\tfrac{\omega T}{2}) \end{aligned}$$

Nhân tử số với mẫu với T, ta được:

$$G(j\omega) = \frac{T\, sin(\omega\, T/2)}{\omega T/2}\, e^{-j\omega\, T/2} \tag{2.20}$$

Từ đó ta vẽ được đặc tính biên và pha của khâu ZOH:



Hình 2.19: Sơ đồ đặc tính biên và pha của khâu ZOH

Hàm truyền của khâu quán tính bậc không tương tự đặc tính bộ lọc thông thấp với tần số cắt $\frac{2\pi}{T}$.

Khi thêm một khâu ZOH thì hệ thống bị chậm pha điều này có thể làm cho hệ thống hồi tiếp ổn định ở dạng liên tục trở thành một ổn định sau khi lấy được mẫu.

2.5 Động cơ DC

Một động cơ DC là một động cơ điện sử dụng dòng điện một chiều (DC) để tạo ra lực cơ học. Các loại phổ biến nhất dựa trên lực từ sản xuất bởi dòng điện trong các cuộn dây. Hầu hết tất cả các loại động cơ DC đều có một cơ chế nội tại, có thể là cơ điện hoặc điện tử, để định kỳ thay đổi hướng dòng điện trong một phần của động cơ.

Động cơ DC là loại động cơ đầu tiên được sử dụng rộng rãi, vì chúng có thể được cung cấp điện từ các hệ thống phân phối điện ánh sáng hiện tại dùng để cung cấp điện một chiều. Tốc độ của động cơ DC có thể được kiểm soát trên một phạm vi rộng, sử dụng hoặc một nguồn điện áp biến đổi hoặc bằng cách thay đổi độ mạnh của dòng điện trong các cuộn dây trường của nó.

Ưu điểm của DC:

Động cơ DC có cấu tạo và hoạt động đơn giản nhưng đóng vai trò rất quan trọng trong công nghệ sản xuất và chế tạo một số đồ vật. Dù đơn giản và nhỏ bé nhưng động cơ DC đem lại rất nhiều lợi ích và ưu điểm nổi bật:

• Động cơ DC có thiết kế đơn giản và có sử dụng nam châm vĩnh cửu nên sử hoạt

động bền bỉ hơn, gia tăng được tuổi thọ (trung bình 15 năm).

- Động cơ DC có khả năng biến đổi dòng điện liên tục nên phù hợp với nhiều dòng điện khác nhau mà vẫn đảm bảo hoạt động hiệu quả.
- Trọng lượng khá nhẹ nên có thể dễ dàng lắp đặt dù ở nơi cao và có độ an toàn cao.
- Động cơ điện một chiều DC có chổi than được đánh giá cao với khả năng khởi động và điều chỉnh đạt hiệu suất tốt.

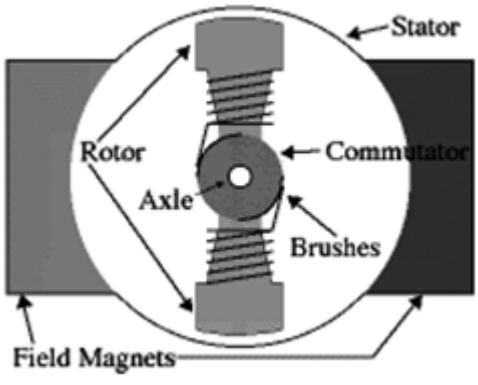
Nhược điểm của DC:

Đi đôi với những ưu điểm vượt trội, động cơ DC cũng không tránh khỏi những nhược điểm trong quá trình hoạt động như:

- Giá thành cao hơn so với những loại động cơ dòng điện một chiều khác như AC.
- Động cơ điện một chiều DC có cấu tạo đơn giản nhưng có tiếp điểm giữa chổi than và cổ góp, có khả năng tạo ra các tia điện trong quá trình hoạt động và mài mòn cơ học. Do đó, làm tăng nhiệt độ của động cơ DC và có thể hư hỏng, cháy chập khi hoạt động quá mức.

2.5.1 Cấu tạo chung

Cấu trúc của động cơ DC được mô tả dưới đây. Việc hiểu về thiết kế của nó rất quan trọng trước khi tìm hiểu về cách nó hoạt động.



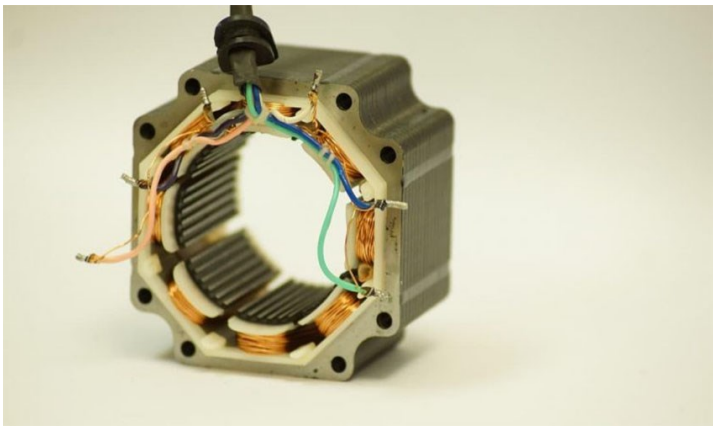
Hình 2.20: Cấu tạo của động cơ DC

Động cơ DC, một thành tựu quan trọng trong lĩnh vực công nghiệp và kỹ thuật điện, kết hợp những thành phần động và tĩnh để chuyển đổi năng lượng điện thành năng lượng

cơ học. Cấu tạo chung của động cơ này đặc trưng bởi hai phần quan trọng: Stator (phần tĩnh) và Rotor (phần quay).

Stator

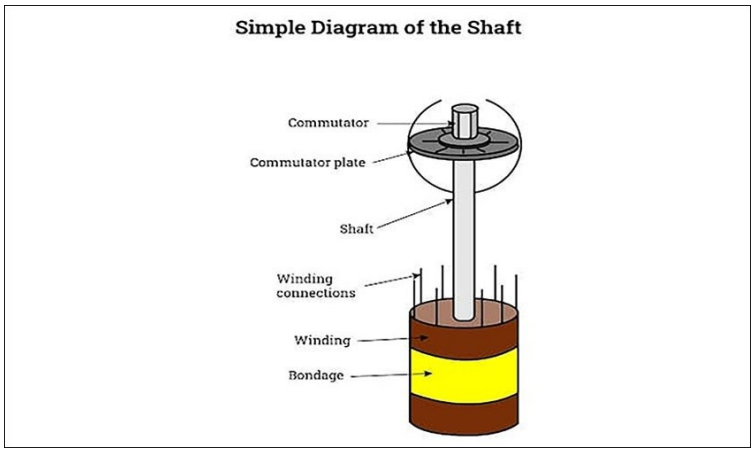
Stator (stato) là một bộ phận đứng yên trên động cơ với một hay nhiều cặp nam châm vĩnh cửu/ nam châm điện với chức năng bảo vệ và hỗ trợ cho động cơ. Stato cung cấp một từ trường quay điều khiển Roto hoặc điều khiển ứng dụng và nhận nguồn cung cấp điện thông qua các thiết bị đầu cuối của nó.



Hình 2.21: Stator của động cơ DC

Trục

Trục của động cơ DC nằm chính giữa trung tâm động cơ và được các bộ phận khác bao quanh. Trục của động cơ DC thường được làm bằng các loại kim loại cứng và sẽ được các bộ phận khác bao quanh, gắn chặt.



Hình 2.22: Trục của động cơ DC

Thiết bị đầu cuối

Thiết bị đầu cuối sẽ có cổ góp vỏ động cơ. Các bộ phận này có thể tháo lắp dễ dàng và gắn chặt để bảo vệ, cách ly động cơ DC với môi trường bên ngoài. Một động cơ DC sẽ có hai thiết bị đầu cuối là tích cực và tiêu cực.

Khi dây dương (+) được kết nối với thiết bị đầu cuối dương (+) và dây âm (-) kết nối với thiết bị đầu cuối âm (-) thì động cơ sẽ quay theo chiều kim đồng hồ. Ngược lại, khi gắn trái dấu với nhau, động cơ sẽ quay ngược chiều kim đồng hồ.

Các thiết bị đầu cuối cung cấp nguồn điện cho động cơ và được kết nối với bàn chải chổi than) và cánh tay bàn chải bên trong nắp lưng để động cơ có thể hoạt động bình thường và liên tục.

Hình 2.23: Thiết bị đầu cuối của động cơ DC

Nam châm

Nam châm sẽ bao quanh thành hình tròn với các miếng đều nhau, được Roto bao bọc xung quanh và cuộn dây quấn đều để có thể cố định. Các nam châm sử dụng trong động cơ DC là nam châm vĩnh cửu.

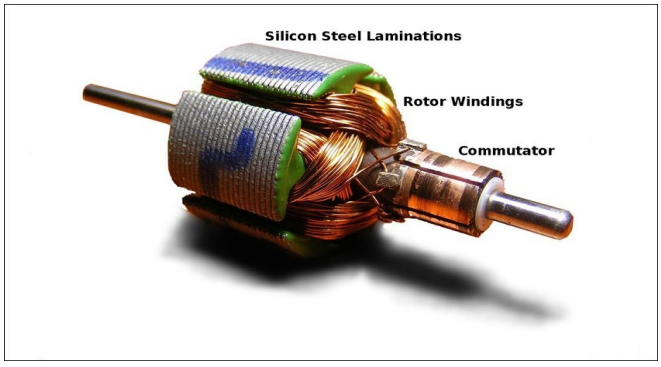
Khi đó, từ trường của động cơ luôn hoạt động. Hai nam châm sẽ tạo ra được mổ từ trường mạnh. Vì vậy, hai nam châm sẽ bao gồm trong động cơ DC xung quanh Roto sao cho từ trường mạnh đi qua Roto.

Hình 2.24: Nam châm vĩnh cửu của động cơ DC

Rotor

Rotor (roto) là phần lõi chuyển động và quay liên tục được làm từ các cuộn dây và quấn lại tạo thành nam châm điện. Rotor được làm bằng nhiều đĩa, cách nhiệt với nhau bằng các tấm nhiều lớp và được sử dụng để tạo ra các cuộc cách mạng cơ học, ứng dụng rộng rãi.

Nhiều đĩa ngăn chặn việc tạo ra một dòng điện xoáy lớn. Rotor là một bộ phận rất cần thiết trong hoạt động của động cơ DC. Các đĩa của Rotor càng nhỏ thì hiệu quả động cơ sẽ càng cao.



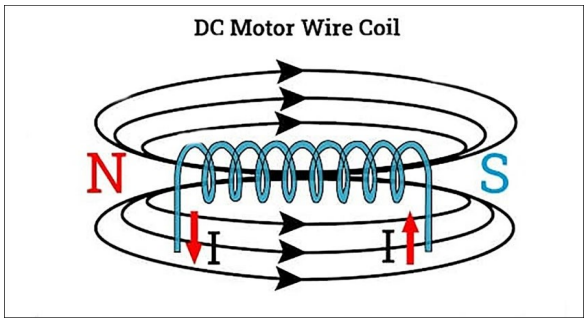
Hình 2.25: Rotor của động cơ DC

Cuộn dây

Cuộn đây sẽ được bao quanh, quấn lấy Roto và tạo ra một từ trường mạnh mẽ. Mỗi loại dây tạo ra một từ trường yếu khi điện đi qua nó nhưng khi kết hợp với tất cả các dây cuộn khác nhau thì sẽ tạo ra một từ trường mạnh.

Thông thường, các động cơ DC sẽ có tối thiểu ba cuộn dây để đảm bảo sự hoạt động

hiệu quả và nhanh nhạy vì khi động cơ có 2 cuộn dây thì dễ xảy ra kẹt và dừng động cơ. Khi nhiều cuộn dây được thêm vào Rotor, vòng quay của nó trở nên mượt mà hơn vì nhiều vòng quấn hơn.

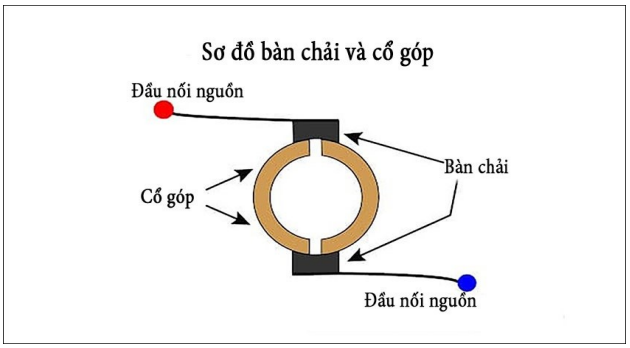


Hình 2.26: Cuộn dây của động cơ DC

Bàn chải

Tùy vào từng mục tiêu và nhu cầu khác nhau, có loại động cơ DC có chổi than và động cơ DC không có chổi than. Hai loại này sẽ có cách thức hoạt động khác nhau nhưng vẫn cùng một mục tiêu là tạo ra từ trường Roto và làm cho Roto quay.

Bàn chải của động cơ DC sẽ cung cấp cho các cuộn dây nguồn điện và là những mảnh kim loại hoạt động như lò xo bởi bàn chải có một vật liệu dẫn điện làm bằng carbon. Các bàn chải được kết nối trực tiếp với các thiết bị đầu cuối hoặc nguồn cung cấp điện của động cơ.



Hình 2.27: Bàn Chải của động cơ DC

Bộ chuyển đổi

Thông thường, bộ chuyển đổi của một động cơ DC được làm bằng các tấm đồng nhỏ gắn trên trục và xoay khi trục quay. Khi Rotor quay, các cực của nguồn điện cung cấp cho các cuộn dây sẽ bị thay đổi.

Mỗi cuộn dây sẽ được kết nối với hai bộ chuyển đổi, được kết nối với nhau bởi các cuộn dây và cách ly điện với nhau. Với các thiết bị có đầu cuối dương và âm được kết nối với hai tấm chuyển đổi, dòng điện dễ dàng chảy và một trường điện từ được tạo ra.

Hình 2.28: Bộ chuyển đổi của động cơ DC

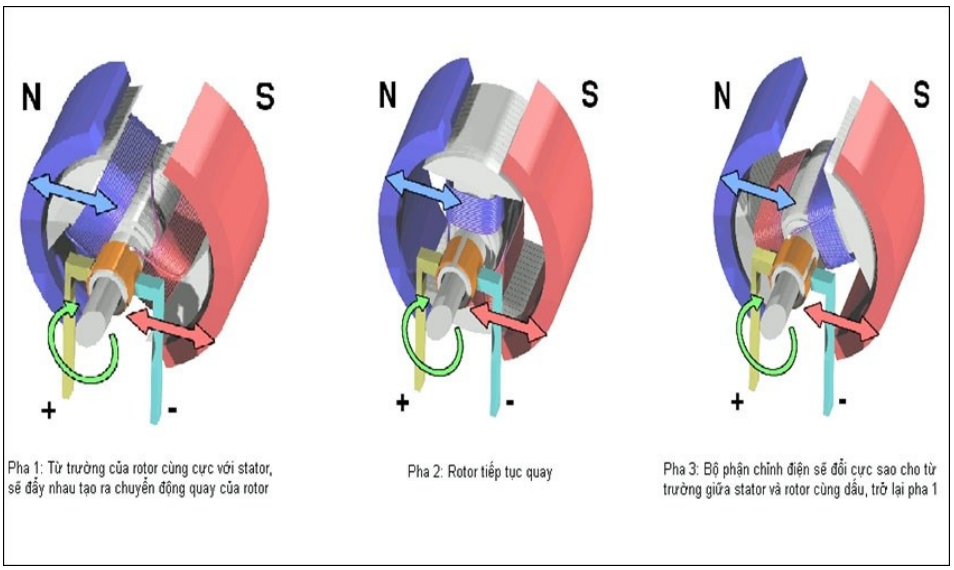
2.5.2 Nguyên lý hoạt động

Động cơ DC lấy năng lượng điện thông qua dòng điện trực tiếp và chuyển đổi năng lượng điện này thành vòng quay cơ học. Khi động cơ DC lấy năng lượng điện sẽ tạo ra một từ trường trong Stator. Từ trường này sẽ thu hút và đẩy lùi nam châm trên Roto, làm cho Roto quay.

Nguyên lý hoạt động này chúng ta còn được biết với cái tên "Quy tắc bàn tay trái" được dạy trong môn Vật Lý. Tuy nhiên, để cỗ máy có thể hoạt động thì Roto cần quay liên tục không ngừng nghỉ, người ta sẽ gắn bộ chuyển đổi vào bàn chải đã kết nối với dòng điện để cung cấp nguồn điện cho cuộn dây động cơ.

Ngoài ra, tốc độ quay của mỗi loại động cơ DC sẽ thay đổi và khác nhau theo từng chu kỳ thời gian, có thể là vòng/phút hoặc nghìn vòng/phút tùy thuộc vào việc ứng dụng đối với từng thiết bị khác nhau.

Có thể nói động cơ DC là loại động cơ đơn giản và phổ biến nhất, được áp dụng trong nhiều ngành công nghiệp sản xuất và chế tạo như trong thiết bị gia dụng, lắp ráp quạt trần, dao cạo thông minh, cửa sổ điện ô tô,...

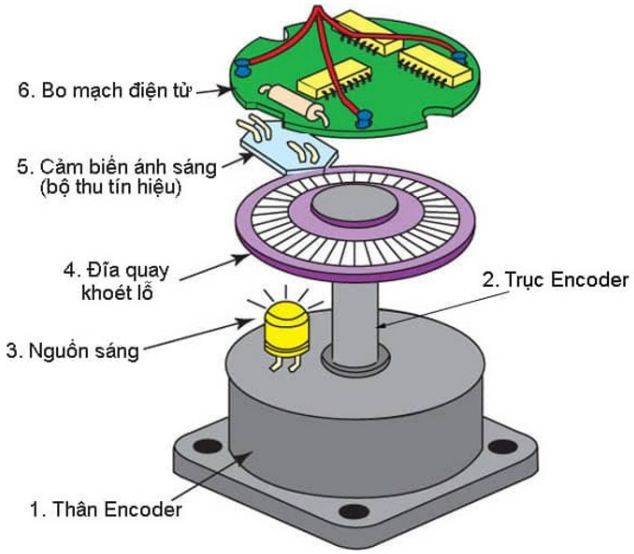


Hình 2.29: Nguyên lý hoạt động của động cơ DC

2.5.3 Encoder của động cơ DC

Encoder là một bộ cảm biến chuyển động cơ học tạo ra tín hiệu analog hoặc tín hiệu kỹ thuật số (digital) đáp ứng với chuyển động. Loại thiết bị cơ điện này có khả năng biến đổi chuyển động (chuyển động tịnh tiến, chuyển động quay của trục, ...) thành tín hiệu đầu ra số hoặc xung. Encoder hay Bộ mã hóa cung cấp một cơ chế đo lường tốc độ của rotor và cung cấp phản hồi vòng đóng đến bộ điều khiển để kiểm soát tốc độ chính xác.

Cấu tạo của Encoder



Hình 2.30: Cấu tạo của encoder

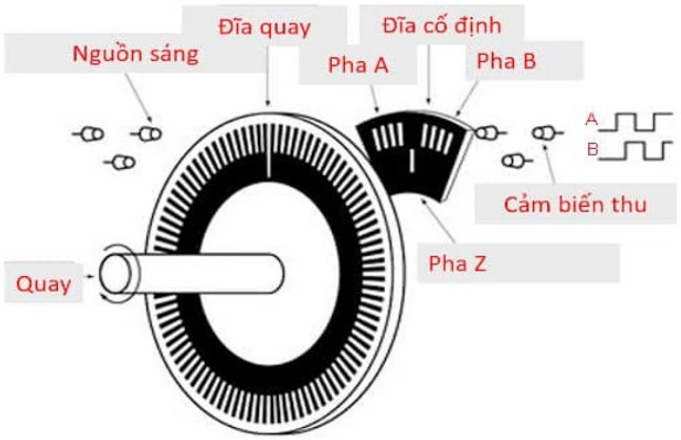
Một bộ mã hóa Encoder gồm có các bộ phận chính:

- **Thân và trục**
- **Nguồn phát sáng (lightsource)**: là 1 đèn LED
- **Đĩa mã hóa (code disk)**: có rãnh nhỏ quay quanh trục, khi đĩa này quay và chiếu đèn LED lên trên mặt đĩa thì sẽ có sự ngắt quãng xảy ra. Các rãnh trên đĩa chia vòng tròn $360°$ thành các góc bằng nhau. Một đĩa có thể có nhiều dãy rãnh tính từ tâm tròn.
- **Bộ cảm biến ánh sáng thu tín hiệu (photosensor)**: là một con mắt thu quang điện để nhận tín hiệu từ đĩa quay.

Nguyên lý hoạt động của Encoder:

Encoder hoạt động theo nguyên lý đĩa quay quanh trục. Trên đĩa mã hóa có các rãnh nhỏ để nguồn phát sáng chiếu tín hiệu quang qua đĩa. Chỗ có rãnh thì ánh sáng xuyên qua được, chỗ không có rãnh ánh sáng không xuyên qua được. Với các tín hiệu có, hoặc không có ánh sáng chiếu qua, người ta ghi nhận được đèn led có chiếu qua lỗ hay không. Số xung đếm được và tăng lên được tính bằng số lần ánh sáng bị cắt.

Cảm biến thu ánh sáng sẽ bật tắt liên tục để tạo ra các xung vuông. Việc sử dụng các bộ mã hóa sẽ ghi nhận lại số xung và tốc độ xung. Tín hiệu dạng xung sẽ được truyền về bộ xử lý trung tâm (vi xử lý, PLC,…) và từ đó chúng ta sẽ biết được vị trí và tốc độ của động cơ.



Hình 2.31: Cấu tạo của encoder

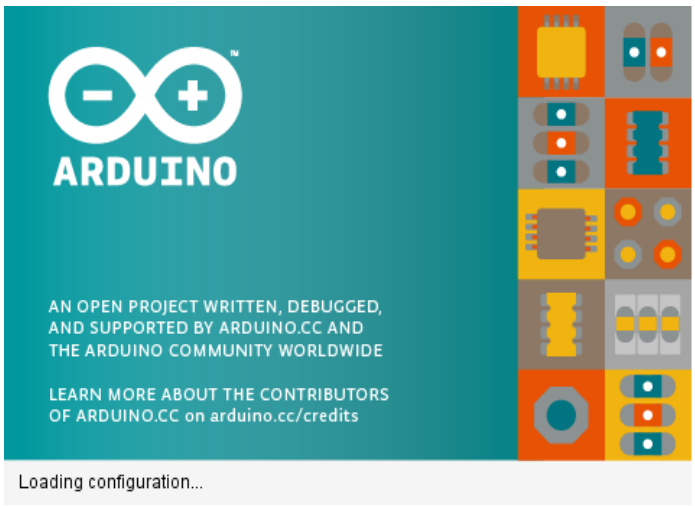
Ví dụ 2.4. Ở trên đĩa có 1 lỗ duy nhất, khi mỗi lần con mắt thu nhận được 1 tín hiệu đèn Led thì cũng có nghĩa là đĩa đã quay được 1 vòng.

Đây chính là nguyên lý hoạt động của Encoder cơ bản. Còn đối với nhiều chủng loại encoder khác thì khi đĩa quay có nhiều lỗ hơn thì khi đó tín hiệu thu nhận sẽ khác hơn.

2.6 Phần mềm Arduino IDE

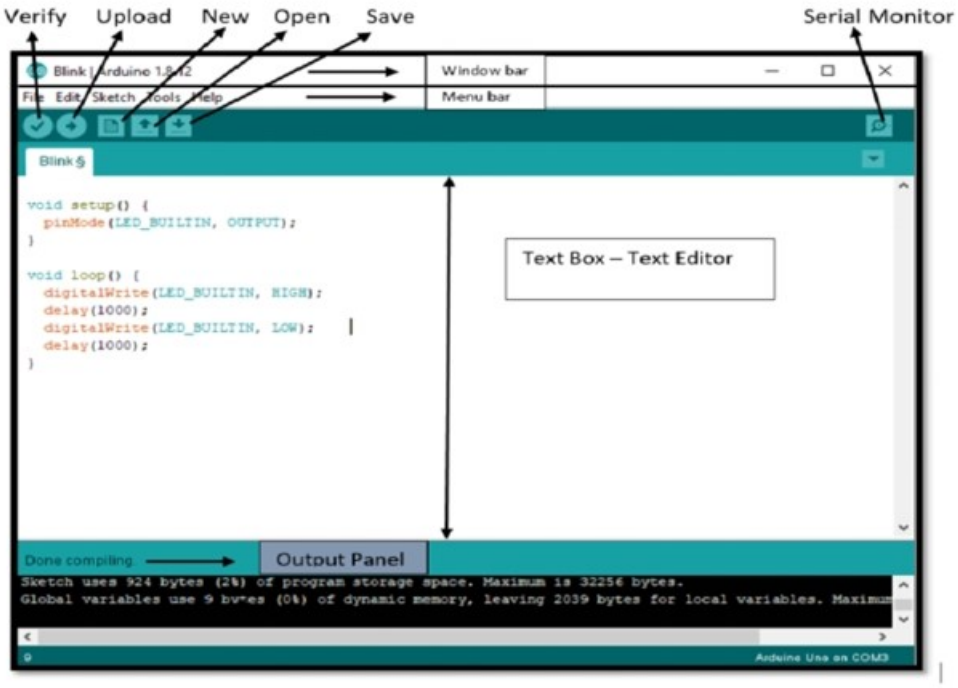
Arduino IDE là một phần mềm với một mã nguồn mở, được sử dụng chủ yếu để viết và biên dịch mã vào module Arduino. Nó bao gồm phần cứng và phần mềm. Phần cứng chứa đến 300,000 board mạch được thiết kế sẵn với các cảm biến, linh kiện. Phần mềm giúp có thể sử dụng các cảm biến, linh kiện ấy của Arduino một cách linh hoạt phù hợp với mục đích sử dụng.

Môi trường IDE chủ yếu chứa hai phần cơ bản: Trình chỉnh sửa và Trình biên dịch, phần đầu sử dụng để viết mã được yêu cầu và phần sau được sử dụng để biên dịch và tải mã lên module Arduino. Mã chính, còn được gọi là sketch, được tạo trên nền tảng IDE sẽ tạo ra một file Hex, sau đó được chuyển và tải lên trong bộ điều khiển trên bo. Môi trường này hỗ trợ cả ngôn ngữ C và C ++.



Hình 2.32: Phần mềm Arduino IDE

Arduino IDE có một giao diện đơn giản, dễ sử dụng giúp người dùng thuận tiện hơn trong thao tác. Dưới đây là một số tính năng chúng ta thường sử dụng:



Hình 2.33: Một số nút chức năng và giao diện của Arduino IDE

Nút kiểm tra chương trình (Verify): giúp dò lỗi phần code định truyền xuống bo mạch Arduino. Menu "Sketchbook" (Sổ ghi chép) trong Arduino IDE chứa các tùy chọn và chức năng liên quan đến quản lý chương trình và thư viện Arduino . Điều này cho phép quản lý các chương trình và thư viện đã tạo và lưu trữ trong sketchbook.

Chức năng quan trọng trong menu "Sketchbook":

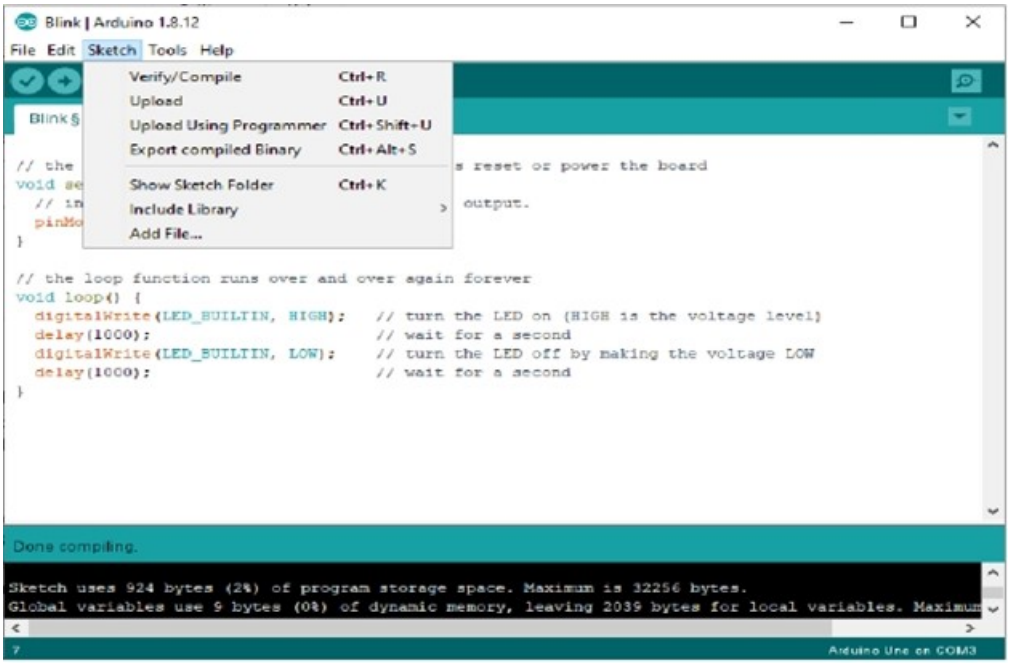
Show Sketch Folder (Hiển thị thư mục chương trình): Khi chọn tùy chọn này, IDE sẽ mở thư mục chứa chương trình Arduino mà đang làm việc. Thông qua cửa sổ thư mục, có thể dễ dàng quản lý các tệp liên quan đến dự án của mình, bao gồm cả mã nguồn chương trình và các tệp hỗ trợ khác.

Sketchbook Manager (Quản lý sổ ghi chép): Tùy chọn này cho phép quản lý chương trình và thư viện đã tạo trong sketchbook. Người dùng có thể xem danh sách các chương trình đã tạo, cài đặt, sao chép, và xóa chúng từ sketchbook manager.

Import Library (Nhập thư viện): Khi sử dụng các thư viện bên ngoài, người dùng có thể sử dụng tùy chọn này để nhập các thư viện đó vào sketchbook của mình. Sau khi nhập, có thể sử dụng các thư viện này trong chương trình của mình.

Import File (Nhập tệp): Tùy chọn này cho phép người dùng nhập các tệp không phải là chương trình Arduino vào sketchbook. Điều này hữu ích khi muốn lưu trữ các tài liệu hướng dẫn, tệp dữ liệu hoặc hình ảnh liên quan đến dự án của mình.

Menu Sketchbook trong Arduino IDE giúp người dùng quản lý các chương trình và thư viện một cách dễ dàng và tiện lợi. Khi làm việc với nhiều dự án hoặc muốn sử dụng lại các thư viện trong các dự án khác nhau, các chức năng trong menu này sẽ giúp tổ chức và quản lý mã nguồn của mình một cách hiệu quả.



Hình 2.34: Menu Sketch của phần mềm Arduino IDE

Menu "Tool" (Công cụ) trong Arduino IDE chứa các tùy chọn và chức năng liên quan đến cấu hình và điều chỉnh các công cụ, thiết lập và tùy chọn liên quan đến việc lập trình và nạp chương trình vào board Arduino.

Chức năng quan trọng trong menu "Tool":

Board (Bo mạch): Tùy chọn này cho phép người dùng chọn loại board Arduino đang sử dụng. Arduino IDE hỗ trợ nhiều loại board khác nhau, chẳng hạn như Arduino Uno, Arduino Nano, Arduino Mega, và nhiều loại khác. Bằng cách chọn board thích hợp, IDE sẽ cấu hình các thông số cần thiết để biên dịch và nạp chương trình vào board.

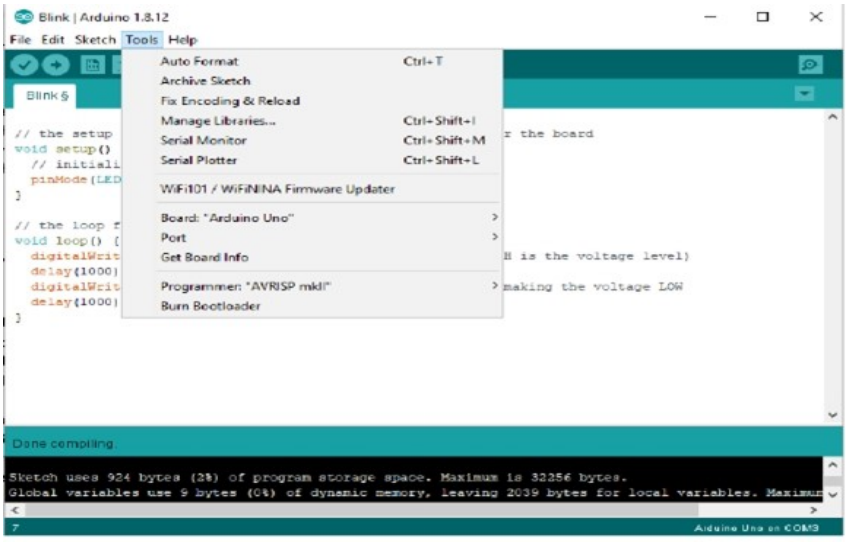
Port (Cổng): Khi kết nối board Arduino với máy tính thông qua cáp USB, tùy chọn này cho phép người dùng chọn cổng nối tiếp mà board đang được kết nối. Điều này cho

phép IDE giao tiếp với board và nạp chương trình vào board thông qua cổng đã chọn Programmer (Chương trình): Đối với một số board Arduino, chẳng hạn như Arduino Uno, người dùng có thể sử dụng chương trình AVRISP hoặc Arduino as ISP để nạp chương trình. Tùy chọn này cho phép chọn chương trình nạp phù hợp với board đang sử dụng.

Burn Bootloader (Nạp bootloader): Tùy chọn này cho phép nạp bootloader vào board Arduino. Bootloader là một chương trình nhỏ được lưu trữ trong bộ nhớ Flash của board và cho phép nạp chương trình vào board qua cổng nối tiếp. Nạp bootloader thường chỉ cần thực hiện khi sử dụng một board mới hoặc board đã bị xóa bootloader. Serial Monitor (Bộ giám sát nối tiếp): Tùy chọn này mở một cửa sổ giám sát nối tiếp, cho phép theo dõi dữ liệu gửi và nhận thông qua cổng nối tiếp trên board Arduino. Điều này hữu ích khi muốn xem các dữ liệu đầu ra hoặc thông báo từ chương trình Arduino.

Auto Format (Định dạng tự động): Tùy chọn này giúp định dạng mã nguồn một cách tự động và dễ đọc hơn. Nó tự động sắp xếp các dòng mã và các cấu trúc điều khiển để mã của người dùng dễ đọc và dễ theo dõi hơn.

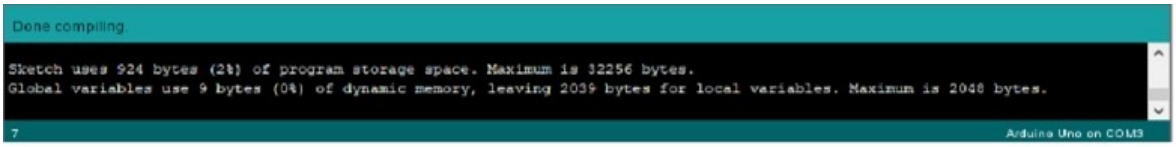
Menu Tools trong Arduino IDE cung cấp các tùy chọn quan trọng để cấu hình và điều chỉnh môi trường lập trình và nạp chương trình. Bằng cách sử dụng các chức năng trong menu này, có thể đảm bảo rằng người dùng đã cấu hình đúng board, cổng và các tùy chọn liên quan.



Hình 2.35: Menu tools của phần mềm Arduino IDE

Output Panel (Cửa sổ đầu ra) trong Arduino IDE là một khu vực hiển thị các thông báo, lỗi, cảnh báo và kết quả từ quá trình biên dịch (compile) và nạp chương trình (upload) vào board Arduino. Nó giúp người dùng kiểm tra lỗi cú pháp, thông báo từ trình biên dịchvà trạng thái quá trình nạp chương trình, cũng như cung cấp thông tin hữu ích để gỡ lỗi chương trình.

Khi thực hiện quá trình biên dịch chương trình bằng cách chọn Verify/Compile từ menu Sketch hoặc nhấn tổ hợp phím Ctrl + R, Arduino IDE sẽ kiểm tra mã nguồn để tìm lỗi cú pháp và trình biên dịch sẽ tạo ra các thông báo và cảnh báo nếu có lỗi hoặc cần chú ý. Các thông báo này sẽ được hiển thị trong cửa sổ Output Panel.



Hình 2.36: Bảng thông báo ngõ ra của phần mềm Arduino IDE

Chương 3:

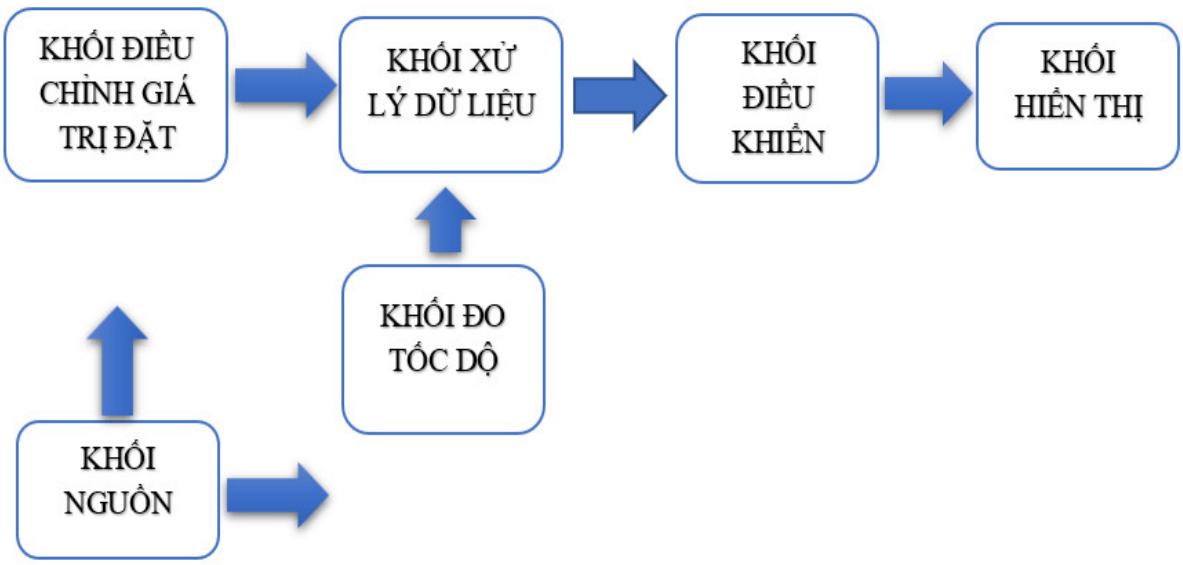
THIẾT KẾ

3.1 Giới thiệu

Chương 3 của đồ án này xoay quanh việc thiết kế và tính toán các khối chức năng cũng như kế hoạch tổng thể cho đồ án của nhóm. Trong phần này, chúng ta sẽ tìm hiểu cụ thể về các khối chức năng trong đồ án và cách chúng tương tác với nhau để đạt được mục tiêu cuối cùng.

3.2 Tính toán vả thiết kế hệ thống

3.2.1 Thiết kế sơ đồ khối



Hình 3.37: Sơ đồ khối toàn mạch

Chức năng:

- **Khối Điều Chỉnh Giá Trị Đặt:** Cho phép người sử dụng thiết lập giá trị tốc độ mong muốn.
- **Khối Xử Lý Dữ Liệu:** Tính toán và xử lý tín hiệu từ encoder và giá trị đặt để tính ra tín hiệu điều khiển cho motor theo thuật toán PID (Proportional-Integral-Derivative).
- **Khối Điều Khiển:** Điều khiển motor DC thông qua tín hiệu điều khiển được tạo

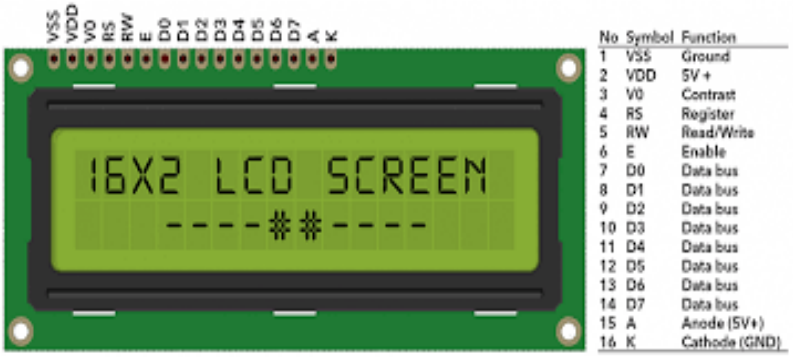
ra bởi khối Xử Lý Dữ Liệu dựa trên tín hiệu từ khối Encoder và giá trị đặt từ Khối Điều Chỉnh Giá Trị Đặt.

- **Khối Hiển Thị** Hiển thị giá trị tốc độ đặt (Setpoint) và giá trị tốc độ hiện tại (Process Variable) lên màn hình LCD để người sử dụng theo dõi.
- **Khối Đo Tốc Độ:** Đo lường tốc độ quay của motor bằng cách theo dõi sự thay đổi của vòng quay.
- **Khối Nguồn:** Cung cấp nguồn điện cho toàn bộ hệ thống, bao gồm nguồn 12V và 5V cho các linh kiện khác.

3.2.2 Thiết kế sơ đồ nguyên lý

a. Khối hiển thị:

Màn hình LCD 16x2 là một loại màn hình hiển thị sử dụng công nghệ LCD (Liquid Crystal Display) với kích thước 16 ký tự trên mỗi dòng và 2 dòng hiển thị. Màn hình này thường được sử dụng trong các ứng dụng như điều khiển và hiển thị thông tin trong các thiết bị điện tử như máy tính, máy tính công nghiệp, máy in, máy tính cá nhân, và nhiều ứng dụng khác.



Hình 3.38: Màn hình LCD 16x2

Bảng 3.1: Bảng chức năng của các chân LCD

Chân	Tên	Chức năng
1	VSS	GND (chân nối đất)
2	VDD	Nguồn cấp cho LCD
3	V0	Điều chỉnh độ tương phản (Bias Voltage)

4	RS	Chọn thanh ghi (Lệnh/Instruction hoặc Dữ liệu/Data)
5	R/W	Chọn thanh ghi đọc/viết dữ liệu (Chọn chế độ đọc hoặc ghi)
6	E	Tín hiệu cho phép
7–14	D0–D7	Dữ liệu tín hiệu (8 chân dữ liệu)
15	LED+	Nguồn cung cấp cho đèn nền (Anode)
16	LED-	Nguồn đất cho đèn nền (Cathode)

Bảng 3.2: Thông số kỹ thuật của màn hình LCD 16x2

Tham số	Ký hiệu	Min	Typ	Max	Đơn vị
Nhiệt độ hoạt động	T_{op}	0		+50	°C
Nhiệt độ lưu trử	T_{st}	-20		+70	°C
Nhiệt độ đầu vào	V_i	V_{ss}		V_{dd}	V
Điện áp cung cấp cho Logic	V_{ss}-V_{dd}	2.7	5.0	5.5	V
Điện áp cung cấp cho LCD	V_{ss}-V_{dd}	3.0		13.0	V

LCD 16×2 có thể sử dụng ở chế độ 4 bit hoặc 8 bit tùy theo ứng dụng ta đang làm nhưng LCD có quá nhiều nhiều chân gây khó khăn trong quá trình đấu nối và chiếm dụng nhiều chân trên vi điều khiển, do vậy nên chúng ta có thể dùng giao thức I2C. I2C, viết tắt của ”Inter-Integrated Circuit,” là một giao thức liên kết dành cho việc truyền thông giữa các thiết bị điện tử như vi xử lý, cảm biến, bộ nhớ, và nhiều loại linh kiện khác. Giao thức I2C giúp các thiết bị giao tiếp với nhau thông qua dây dẫn dữ liệu chung.

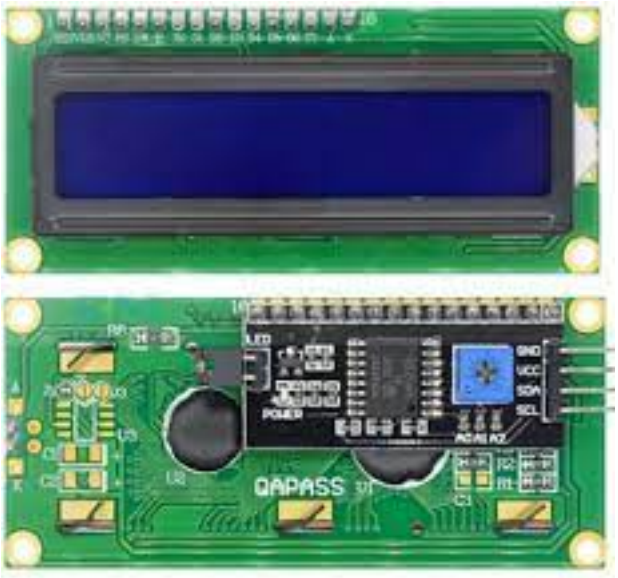
Hình 3.39: Mạch Chuyển Đổi Giao Tiếp I2C Cho LCD

Thay vì phải mất 6 chân vi điều khiển để kết nối với LCD 16x2 (RS, EN, D7, D6, D5 và D4) thì module IC2 chỉ cần tốn 2 chân (SCL, SDA) để kết nối. Module I2C hỗ trợ

các loại LCD sử dụng driver HD44780(LCD 16x2, LCD 20x4, ...) và tương thích với hầu hết các vi điều khiển hiện nay.

Thông số kĩ thuật của module I2C:

- Điện áp hoạt động: 2.5-6V DC.
- Hỗ trợ màn hình: LCD1602,1604,2004 (driver HD44780).
- Giao tiếp: I2C.
- Địa chỉ mặc định: 0X27 (có thể điều chỉnh bằng ngắn mạch chân A0/A1/A2).
- Tích hợp Jump chốt để cung cấp đèn cho LCD hoặc ngắt.
- Tích hợp biến trở xoay điều chỉnh độ tương phản cho LCD.



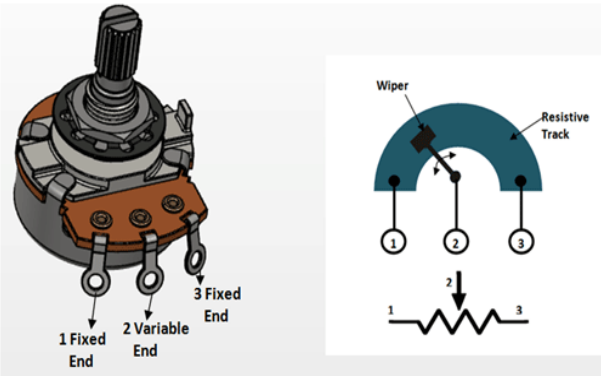
Hình 3.40: LCD 16x2 tích hợp module chuyển đổi I2C

Ưu điểm:

- Giảm dây cáp.
- Hiệu suất và tiết kiệm năng lượng.
- Tốc độ truyền dữ liệu lên đến 400Kbps.

b. Khối điều chỉnh giá trị đặt

Để có thể chỉnh giá trị đặt (setpoint) cho tốc độ quay mong muốn của động cơ. Nhóm đã chọn biến trở với giá trị 10(kohm). Bằng cách xoay biến trở, bạn có thể thay đổi giá trị đặt để đạt được tốc độ quay mong muốn.



Hình 3.41: Biến trở 10k

Bảng 3.3: Bảng chức năng của các chân biến trở 10 kohm

Chân	Tên	Chức năng
1	Chân cố định (Fixed End)	Cung cấp một điểm cố định về giá trị điện trở hoặc điện áp
2	Chân Biến Đổi (Variable End)	Tạo ra một điểm điều chỉnh giữa hai điểm cuối (Chân 1 và Chân 3)
3	Chân cố định (Fixed End)	Cung cấp một điểm cố định về giá trị điện trở hoặc điện áp

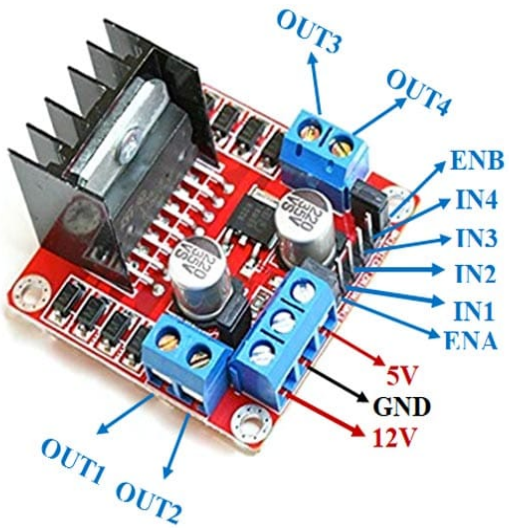
Tín hiệu từ biến trở được đọc bởi Arduino để xác định giá trị setpoint. Giá trị đọc từ biến trở sẽ được chuyển đổi thành RPM (vòng quay trên mỗi phút) để sử dụng trong quá trình điều khiển tốc độ.

Thông số kĩ thuật của biến trở 10k:

- Loại: Chân biến trở xoay (Radio POT)
- Công suất định mức: 0.3W
- Tuổi thọ quay: 2000K vòng quay

c. Khối điều khiển

Để điều khiển và đảm bảo hoạt động của động cơ DC nhóm lựa chọn dùng driver L298N. Driver L298N là một driver cầu H hoặc còn gọi là driver motor.



Hình 3.42: Driver L298N

Bảng 3.4: Bảng chức năng các chân driver L298N

Chân	Tên	Chức năng
1	IN1 và IN2	Các chân đầu vào điều khiển hướng quay động cơ A
2	IN3 và IN4	Các chân đầu vào điều khiển hướng quay động cơ B
3	ENA	Kích hoạt tín hiệu PWM cho Động cơ A
4	ENB	Kích hoạt tín hiệu PWM cho Động cơ B
5	OUT1 và OUT2	Chân đầu ra cho Động cơ A
6	OUT3 và OUT4	Chân đầu ra cho Động cơ B
7	12V	Đầu vào cấp nguồn 12V
8	5V	Cấp nguồn cho mạch logic bên trong IC L298N
9	GND	Chân nối đất

Thông số kĩ thuật của L298N:

- Điện áp cấp cho động cơ (Tối đa): 46V
- Dòng điện cấp động cơ (tối đa): 2A
- Điện áp logic: 5V

- Điện áp hoạt động của IC: 5-35V
- Dòng điện hoạt động IC: 2A
- Dòng logic: 0-36mA
- Công suất tối đa (W): 25W

Giải thích cách hoạt động:

Module này sử dụng hai kỹ thuật để kiểm soát tốc độ và hướng quay của động cơ DC. Đó là mạch cầu H (để kiểm soát hướng quay) và PWM (để kiểm soát tốc độ).

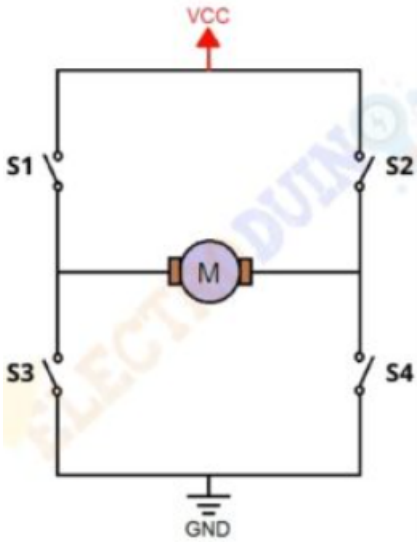
Mạch cầu H

Mạch cầu H kiểm soát hướng quay của động cơ DC bằng cách thay đổi cực tính của điện áp đầu vào. Mạch cầu H chứa bốn thành phần như là bốn transitor (BJT hoặc MOSFET), với động cơ ở giữa tạo thành cấu hình giống chữ H. Các chân đầu vào IN1, IN2, IN3 và IN4 thực sự kiểm soát các công tắc của mạch H-Bridge bên trong vi mạch L298N.

Chúng ta có thể thay đổi hướng dòng chảy bằng cách kích hoạt hai công tắc cụ thể cùng một lúc, từ đó thay đổi hướng quay của động cơ.

- *Trường hợp 1:*

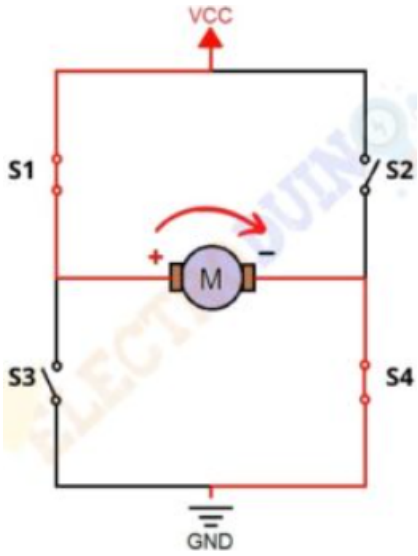
Khi cả bốn công tắc S1, S2, S3 và S4 đều đóng cửa không có dòng chảy đến cực của động cơ. Do đó, trong trạng thái này, động cơ bị ngừng (không hoạt động).



Hình 3.43: Trường hợp 1 cách hoạt động của mạch cầu H

• *Trường hợp 2:*

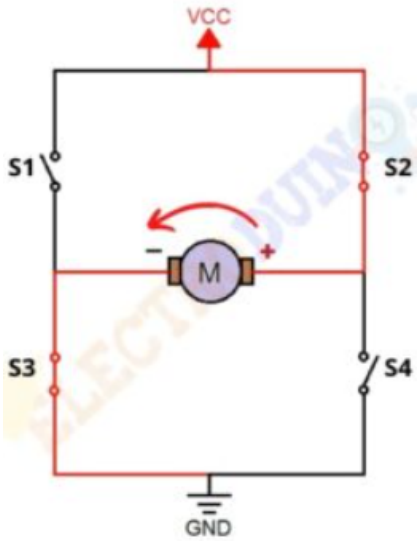
Khi công tắc S1 và S4 được đóng, thì cực trái của động cơ nhận được điện áp dương (+) và cực phải của động cơ nhận được điện áp âm (-). Do đó, trong trạng thái này, động cơ bắt đầu quay theo một hướng cụ thể (theo chiều kim đồng hồ).



Hình 3.44: Trường hợp 2 cách hoạt động của mạch cầu H

• *Trường hợp 3:*

Khi công tắc S2 và S3 được đóng , thì cực phải của động cơ nhận điện áp dương (+) và cực trái của động cơ nhận điện áp âm (-). Do đó, trong trạng thái này, động cơ bắt đầu quay theo một hướng cụ thể (ngược chiều kim đồng hồ).

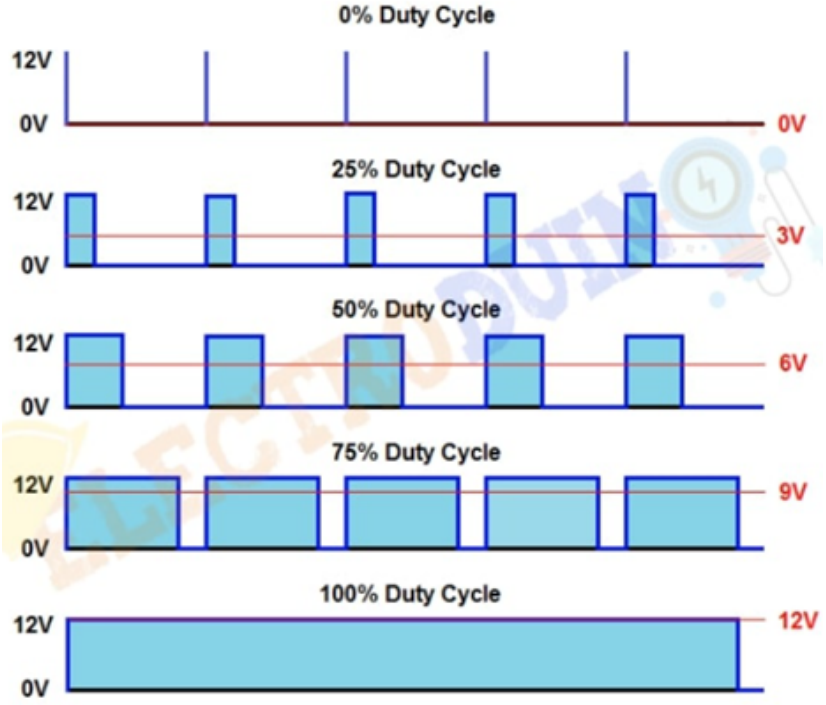


Hình 3.45: Trường hợp 3 cách hoạt động của mạch cầu H

PWM

Module điều khiển động cơ L298n sử dụng kỹ thuật PWM để kiểm soát tốc độ quay của động cơ DC. Trong kỹ thuật này, tốc độ của động cơ DC có thể được điều khiển bằng cách thay đổi điện áp đầu vào của nó.

Pulse Width Modulation (PWM) là một kỹ thuật trong đó giá trị trung bình của điện áp đầu vào được điều chỉnh bằng cách gửi một chuỗi các xung BẬT-TẮT. Giá trị trung bình này tỷ lệ với độ rộng của các xung, các xung này được gọi là Duty Cycle. Nếu Duty Cycle cao, thì điện áp trung bình được áp dụng cho động cơ DC (Tốc độ cao), và nếu Duty Cycle thấp, thì điện áp trung bình thấp hơn sẽ được áp dụng cho động cơ DC (Tốc độ thấp).



Hình 3.46: Kỹ thuật PWM của module điều khiển động cơ L298N

Động cơ DC JGA25-37:

Nhóm lựa chọn động cơ DC JGA25-371 để ổn định và điều khiển dựa trên tín hiệu điều khiển được tính toán bằng hệ thống điều khiển PID.

Hình 3.47: DC motor JGA25-371 280RPM

Thông số kỹ thuật:

- Điện áp của động cơ: 12V.
- Tốc độ của động cơ:

  Ở điện áp 12V: 280 vòng/phút.

  Ở điện áp 9V: 210 vòng/phút.

  Ở điện áp 14V: 325 vòng/phút.
- Điện áp định mức: 12V.
- Màu Phạm vi điện áp: Từ 6V đến 24V.
- Tốc độ không tải: 201 vòng/phút.
- Tốc độ tải: 168 vòng/phút.
- Dòng không tải: 46mA.
- Dòng tải: 300mA.
- Dòng tối đa (stall current): 1A.
- Trọng lượng tịnh: Khoảng 100g.
- tỷ lệ giảm tốc: 21.3

d. Khối đo tốc độ:

Encoder của động cơ JGA25-371 là một thành phần quan trọng, giúp nâng cao hiệu suất và linh hoạt của hệ thống động cơ, đặc biệt là trong các ứng dụng đòi hỏi độ chính xác và độ ổn định.

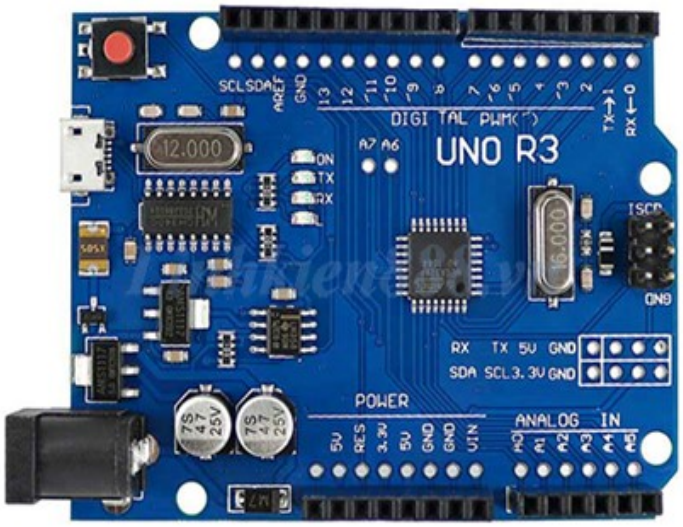
Hình 3.48: Encoder của động cơ DC

Phương thức kết nối:

- Màu đỏ: Nguồn dương của motor.
- Màu đen: Nguồn âm của encoder (kết nối ngược có thể điều khiển motor theo chiều ngược lạ)i.
- Màu vàng: Encoder A.
- Màu lục: EncoderB.
- Màu lam: Nguồn dương của encoder (điện áp nên là 3.3 – 6V).
- Màu trằng: Nguồn âm của encoder.
- Độ Phân Giải Encoder: 11 lines.

e. Khối xử lý dữ liệu:

Để xử lý dữ liệu đầu vào từ encoder và biến trở, sau đó thực hiện điều khiển tốc độ của động cơ DC bằng cách tính toán tín hiệu điều khiển PID và gửi nó thông qua chân PWM để điều chỉnh tốc độ quay của động cơ. Nhóm sẽ sử dụng bo mạch điện tử Arduino.



Hình 3.49: Arduino Uno R3

Bảng 3.5: Bảng chức năng các chân Arduino UNO R3

Chân	Tên	Chức năng
1	NC	Không kết nối (Not connected)
2	IOREF	Chân đọc điện áp của vi điều khiển trên Arduino UNO
3	Reset	Chân đặt lại bo mạch Arduino.
4	+3.3V	Nguồn điện áp 3.3V
5	+5V	Nguồn điện áp 5V
6	GND	Chân Ground
7	V_in	Chân dùng để cấp nguồn ngoài (điện áp cấp từ 7-12V_dc)
8	A0-A3	Các chân đầu vào analog
9	A4/SDAL-A5/SCL	Hỗ trợ giao tiếp I2C/TWI với các thiết bị khác.
10	AREF	Chân đặt điện áp đầu vào tương tự bên ngoài
11	D0-D9	Các chân đầu vào Digital
12	D10-D13	Hỗ trợ giao tiếp SPI bằng thư viện SPI

Bảng 3.6: Thông số kỹ thuật của màn hình Arduino UNO R3

Tham số	Ký hiệu	Min	Typ	Max	Đơn vị
Nhiệt độ hoạt động	T_{op}	-40		+85	°C

Điện áp đầu vào tối đa từ chân VIN	Vinmax	6		+20	V
Điện áp đầu vào tối đa từ cổng US	Vusbmax			+5.5	V

Lưu ý:

Ở nhiệt độ quá cao, EEPROM, bộ điều tiết điện áp và bộ dao động thạch anh thể có thể không hoạt động như mong đợi.

Dòng điện DC trên mỗi chân I/O: 20 mA

Dòng điện DC trên chân 3.3V: 50 mA

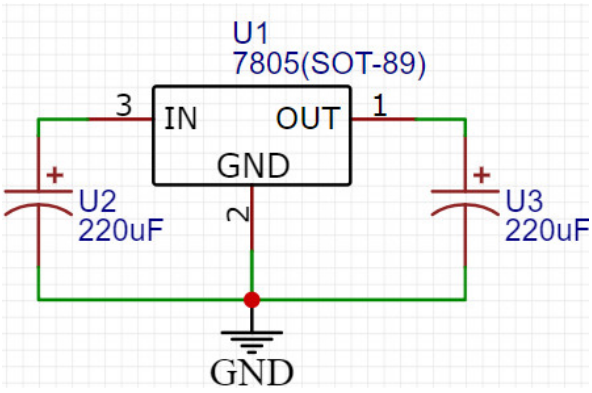
Tốc độ thạch anh: 16 MHz

f. Khối nguồn:

Phần Khối Nguồn trong đồ án là một phần quan trọng để đảm bảo rằng tất cả các linh kiện trong hệ thống như màn hình LCD, Arduino, L298N, biến trở, và motor DC được cung cấp nguồn điện đủ và ổn định để hoạt động một cách đúng đắn.

Để cung cấp năng lượng cho Arduino, nguồn điện cần phải là 5V. Tuy nhiên, động cơ DC mà tôi đang sử dụng yêu cầu một nguồn điện 12V. Để giải quyết vấn đề này, thì phải thêm một mạch hạ áp vào hệ thống. Mạch này giúp giảm áp nguồn từ 12V xuống 5V, làm cho nó phù hợp với yêu cầu của Arduino. Điều này đảm bảo cả động cơ và Arduino có thể hoạt động một cách ổn định với nguồn điện chung, tạo điều kiện cho việc tích hợp chúng trong dự án.

Mạch hạ áp từ 12V xuống 5V:

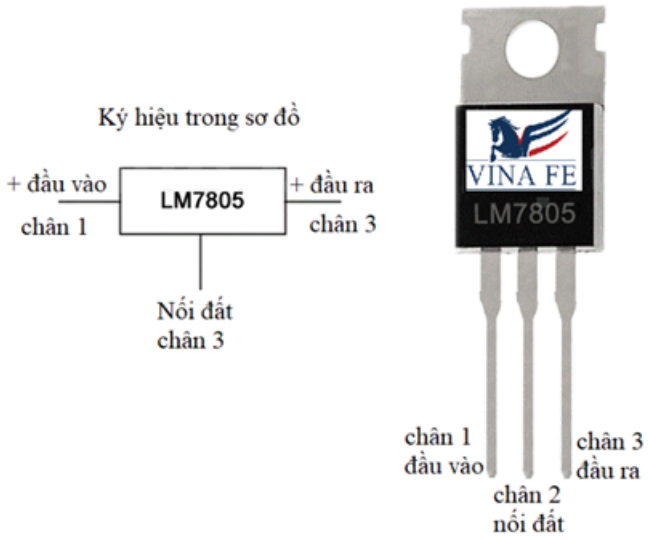


Hình 3.50: Mạch hạ áp từ 12V xuống 5V

Mạch hạ áp thường sử dụng một linh kiện gọi là ”voltage regulator” để giảm áp đầu vào từ mức cao (ví dụ: 12V) xuống mức thấp hơn (ví dụ: 5V) để cung cấp cho mạch điện tử, như Arduino, một nguồn điện ổn định.

IC 7805:

IC LM7805 là một linh kiện ổn áp (voltage regulator) có khả năng giảm áp đầu vào xuống 5V.



Hình 3.51: IC LM7805

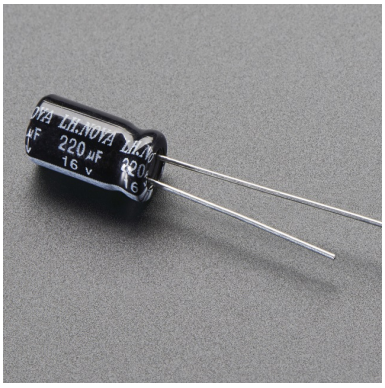
- **Chân 1** : Chân đầu vào.
- **Chân 2** ; Chân nối đất.
- **Chân 3** : Chân đầu ra.

Thông số kỹ thuật:

- Bộ điều chỉnh điện áp dương 5V
- Điện áp đầu vào tối thiểu là 7V
- Điện áp đầu vào tối đa là 25V
- Dòng hoạt động (IQ) là 5mA
- Có sẵn bảo vệ giới hạn quá tải nhiệt và ngắn mạch bên trong.
- Nhiệt độ mối nối tối đa 125 độ C

IC LM7805 đảm bảo rằng áp đầu ra được giữ ổn định ở mức 5V dù áp đầu vào có thể thay đổi. Nếu áp đầu vào tăng lên, 7805 tự động giảm áp đầu ra để duy trì giá trị mong muốn.

Tụ hóa 220 μF:



Hình 3.52: Mạch hạ áp từ 12V xuống 5V

Tụ hóa có khả năng lọc nhiễu từ nguồn điện, giảm các dao động hay nhiễu gây ra bởi các yếu tố khác trong mạch hoặc từ nguồn điện ngoại vi.

Đảm bảo rằng tụ được chọn có điện áp tuân thủ (voltage rating) đủ cao để chịu được điện áp trong mạch mà nó sẽ được sử dụng. Điều này giúp tránh hiện tượng hỏng hóc hoặc hỏng tụ do điện áp quá mức.

Tính toán tổng dòng toàn mạch:

- **Màn hình LCD I2C:** Màn hình LCD kết hợp với module I2C tiêu thụ từ 20mA đến 50mA tùy theo hoạt động. Điều này đòi hỏi một nguồn điện 5V ổn định.
- **L298N**: L298N là driver motor mạnh mẽ, dòng tiêu thụ của module là từ 6-36mA.
- Arduino Uno R3: Arduino Uno R3 tiêu thụ từ 50mA đến 100mA và yêu cầu một nguồn điện 5V.
- **Motor DC tích hợp encoder JGA27-371**: Đây là linh kiện yêu cầu dòng khá lớn, khoảng 300mA khi tải. Chúng ta cần cung cấp cho nó nguồn điện 12V.

Tổng dòng cần thiết cho toàn mạch:

$$I = 50mA + 36mA + 100mA + 300mA = 486mA \tag{3.21}$$

Để đảm bảo rằng hệ thống hoạt động ổn định và an toàn, chúng ta cần cung cấp một nguồn điện có thể cung cấp ít nhất 500mA (hoặc hơn) với điện áp phù hợp (5V hoặc 12V) cho toàn bộ đồ án. Do đó nhóm chọn Adapter 12V với dòng 2A để cấp nguồn cho toàn mạch.

3.2.3 Lưu đồ và chương trình

a. Giời thiệu yêu cầu điều khiển.

Các yêu cầu điều khiển trong đồ án này liên quan đến cách bạn điều khiển và quản lý các linh kiện khác nhau trong hệ thống:

Điều Khiển Motor DC Tích Hợp Encoder:

- Đảm bảo rằng motor DC được điều khiển theo tốc độ mong muốn.
- Giám sát tốc độ của motor DC bằng cảm biến encoder.
- Cung cấp khả năng thay đổi tốc độ của motor DC từ Arduino.

Điều Khiển Màn Hình LCD:

- Hiển thị thông tin cần thiết lên màn hình LCD.
- Cung cấp giao diện người dùng hoặc hiển thị các thông số quan trọng.

Điều Khiển L298N:

- Đảm bảo rằng L298N cung cấp nguồn điện ổn định cho motor DC và được điều khiển đúng cách để quyết định tốc độ của motor.

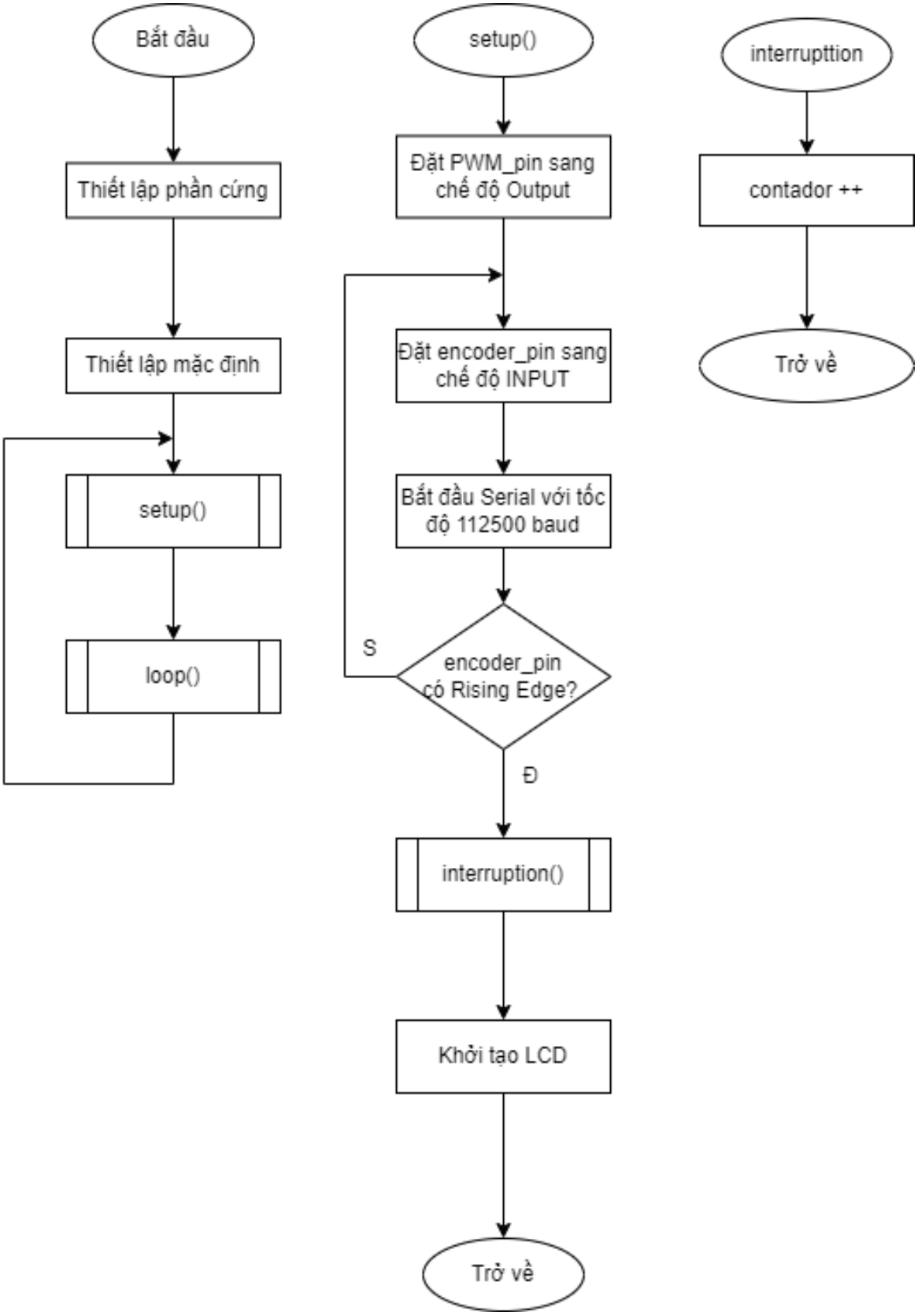
Điều Khiển Arduino Uno R3:

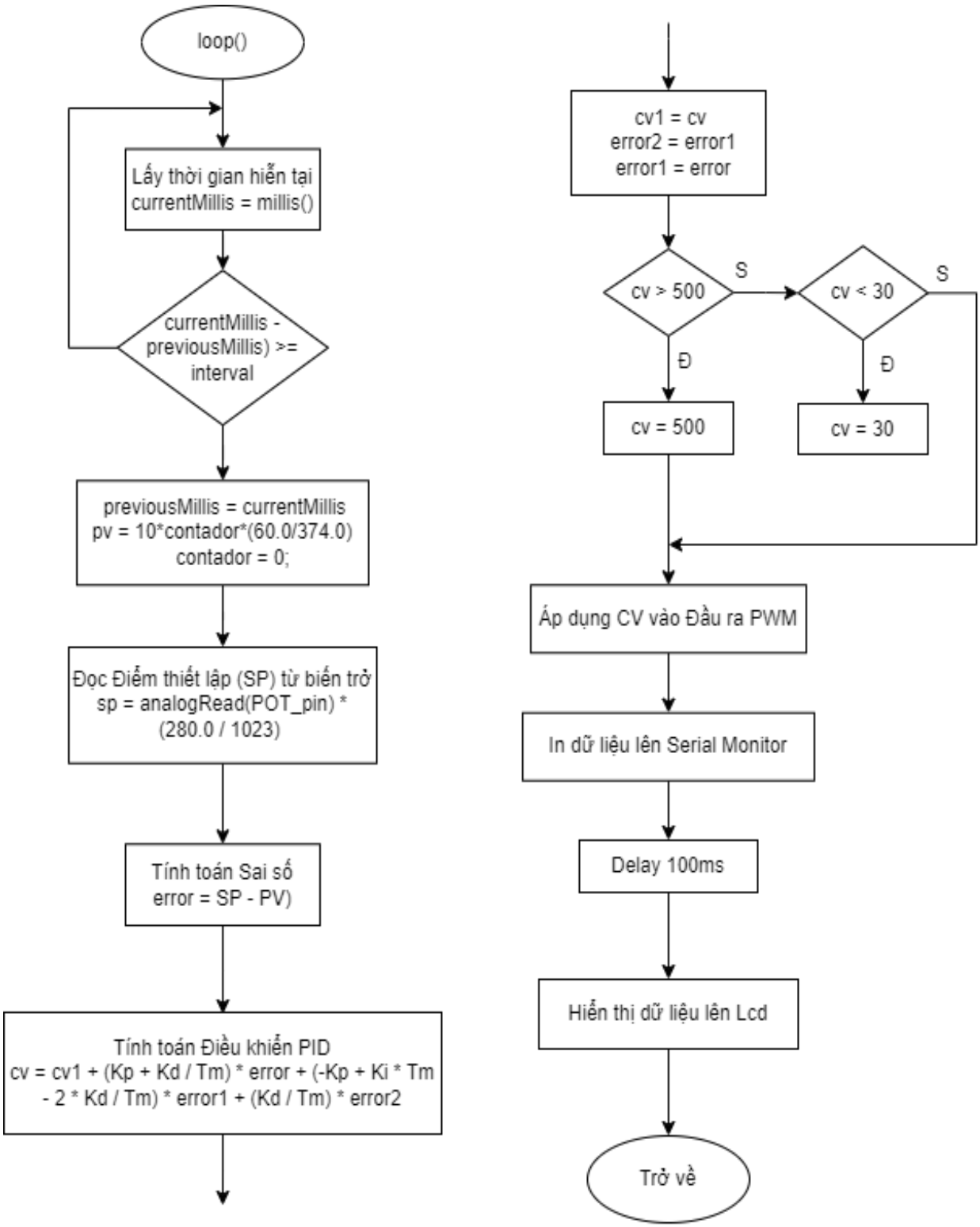
- Lập trình Arduino để quản lý các tác vụ điều khiển, bao gồm điều khiển motor DC, hiển thị trên màn hình LCD và tương tác với người dùng nếu cần.

Kiểm Soát Dòng và Điện Áp:

- Đảm bảo rằng các linh kiện như motor DC và màn hình LCD nhận đủ nguồn điện với điện áp và dòng phù hợp để hoạt động ổn định.
- Những yêu cầu này liên quan đến cách lập trình và kết nối các linh kiện để hệ thống hoạt động theo cách mong muốn.

b. Lưu đồ: Cho biết trình tự diều khiển.





Hình 3.53: Lưu đồ hoạt động

c. Chương trình.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

```
3
4  // Constants for pins and intervals
5  const int POT_pin = A1;
6  const int PWM_pin = 6;
7  const int encoder_pin = 3;
8  const long interval = 100;
9
10 // Constants for PID control
11 const float Kp = 0.5; // Adjust this value for optimal performance
12 const float Ki = 5; // Adjust this value for optimal performance
13 const float Kd = 0.001; // Adjust this value for optimal performance
14 const float Tm = 0.1;
15
16 LiquidCrystal_I2C lcd(0x27, 16, 2);
17
18 // Variables
19 volatile int contador = 0;
20 unsigned long previousMillis = 0;
21
22 float sp = 0;
23 float pv = 0;
24 float cv = 0;
25 float cv1 = 0;
26 float error = 0;
27 float error1 = 0;
28 float error2 = 0;
29
30 void setup() {
31   pinMode(PWM_pin, OUTPUT);
32   pinMode(encoder_pin, INPUT);
33   Serial.begin(112500);
34   attachInterrupt(digitalPinToInterrupt(encoder_pin), interruption,
      RISING);
35
36   lcd.init();
37   //lcd.backlight();
38   lcd.begin(16, 2);
```

```
}

void loop() {
  unsigned long currentMillis = millis();

  if ((currentMillis - previousMillis) >= interval) {
    previousMillis = currentMillis;
    pv = 10*contador*(60.0/374.0);; // Calculate in RPM
    contador = 0;
  }

  // Read the setpoint from the potentiometer
  sp = analogRead(POT_pin) * (280.0 / 1023); // Map potentiometer
   value to RPM range

  error = sp - pv;

  // PID Control Calculation
  cv = cv1 + (Kp + Kd / Tm) * error + (-Kp + Ki * Tm - 2 * Kd / Tm) *
   error1 + (Kd / Tm) * error2;
  cv1 = cv;
  error2 = error1;
  error1 = error;

  // Limit the output of the PID controller
  if (cv > 500) {cv = 500.0;}

  if(cv<30.0) {cv = 30.0;}

  // Apply the control signal to the PWM output
  analogWrite(PWM_pin,cv*(255.0/500.0));//0-255

  // Print data to Serial monitor
  Serial.print("Setpoint (RPM): ");
  Serial.print(sp);
  Serial.print(", Process Variable (RPM): ");
  Serial.println(pv);
```

```
  delay(100);

  // Display data on the LCD
  lcd.setCursor(0, 0);
  lcd.print("SP (RPM): ");
  lcd.print(sp);
  lcd.setCursor(0, 1);
  lcd.print("PV (RPM): ");
  lcd.print(pv);
}

void interruption() {contador++;}
```

d. Giải thích các lệnh sử dụng trong chương trình

Khối Import Thư Viện:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

Thư viện **Wire.h** cần được sử dụng để giao tiếp I2C, còn **LiquidCrystal_I2C.h** giúp quản lý LCD thông qua giao tiếp I2C.

Khối Định Nghĩa Hằng Số và Biến:

```
const int POT_pin = A1;
const int PWM_pin = 6;
const int encoder_pin = 3;
const long interval = 100;

const float Kp = 0.5; // ệH ố s Proportional
const float Ki = 5;   // ệH ố s Integral
const float Kd = 0.001; // ệH ố s Derivative
const float Tm = 0.1;  // ờThi gian ãmu

LiquidCrystal_I2C lcd(0x27, 16, 2);
```

POT_pin: Chân nối potentiometer.

PWM_pin: Chân nối PWM (đầu ra điều khiển).

encoder_pin: Chân nối encoder.

interval: Khoảng thời gian giữa các lần lấy mẫu.

Kp, Ki, Kd: Hệ số của PID.

Tm: Thời gian mẫu cho PID.

LiquidCrystal_I2C lcd(0x27, 16, 2): Khởi tạo đối tượng LCD I2C với địa chỉ 0x27, màn hình 16x2.

Hàm Setup:

```
void setup() {
  pinMode(PWM_pin, OUTPUT);
  pinMode(encoder_pin, INPUT);
  Serial.begin(112500);
  attachInterrupt(digitalPinToInterrupt(encoder_pin), interruption,
   RISING);

  lcd.init();
  lcd.begin(16, 2);
}
```

pinMode(PWM_pin, OUTPUT): Chế độ đầu ra cho chân PWM.

pinMode(encoder_pin, INPUT): Chế độ đầu vào cho chân encoder.

Serial.begin(112500): Bắt đầu giao tiếp Serial với tốc độ 112500 bps.

attachInterrupt(digitalPinToInterrupt(encoder_pin), interruption, RISING): Kích hoạt ngắt trên chân encoder, gọi hàm interruption khi có sự kiện RISING.

lcd.init(), lcd.begin(16, 2): Khởi tạo và thiết lập LCD.

Hàm Loop:

```
void loop() {
  unsigned long currentMillis = millis();

  if ((currentMillis - previousMillis) >= interval) {
    previousMillis = currentMillis;
    pv = 10*contador*(60.0/234.3); // Calculate in RPM
```

```
    contador = 0;
  }
  // ...
}
```

Kiểm tra xem đã đến thời điểm lấy mẫu chưa, nếu đúng, tính toán giá trị process variable (pv) từ số xung đếm của encoder. Giá trị của **process variabl**e được tính theo công thức:

$$V_{truc\,tai}(rpm) = (Tan\,suat\,encoder) \times \frac{1}{IMP\,.\,Ty\,le\,giam\,toc} \times \frac{1}{60} \tag{3.22}$$

Trong công thức này:

- $V_{truc\,tai}$ là vận tốc của trục tải, tính bằng vòng quay trên trục mỗi giây.
- Tần suất encoder là số xung mà encoder tạo ra mỗi giây.
- IMP là số xung (pulsos) mỗi vòng quay của encoder.
- Tỷ lệ giảm tốc là tỉ lệ giảm tốc của hộp số.
- 1/60 là để chuyển đơn vị thời gian từ giây sang phút.

Công thức này có thể được hiểu như sau:

Lấy tần suất của encoder (tức là số xung mà encoder tạo ra mỗi giây), chia cho số xung mỗi vòng quay của encoder (IMP) nhằm chuyển đổi sang số vòng quay mỗi giây.

Chia cho tỉ lệ giảm tốc của hộp số để tính vận tốc của trục tải theo vòng quay mỗi giây. Nhân với 1/60 để chuyển đổi vận tốc từ vòng quay mỗi giây sang vòng quay mỗi phút (rpm). Công thức này cung cấp vận tốc của trục tải dựa trên tần suất của encoder và các thông số khác liên quan đến encoder và hộp số.

Đọc Setpoint và Process Variable:

```
sp = analogRead(POT_pin) * (280.0 / 1023);
error = sp - pv;
```

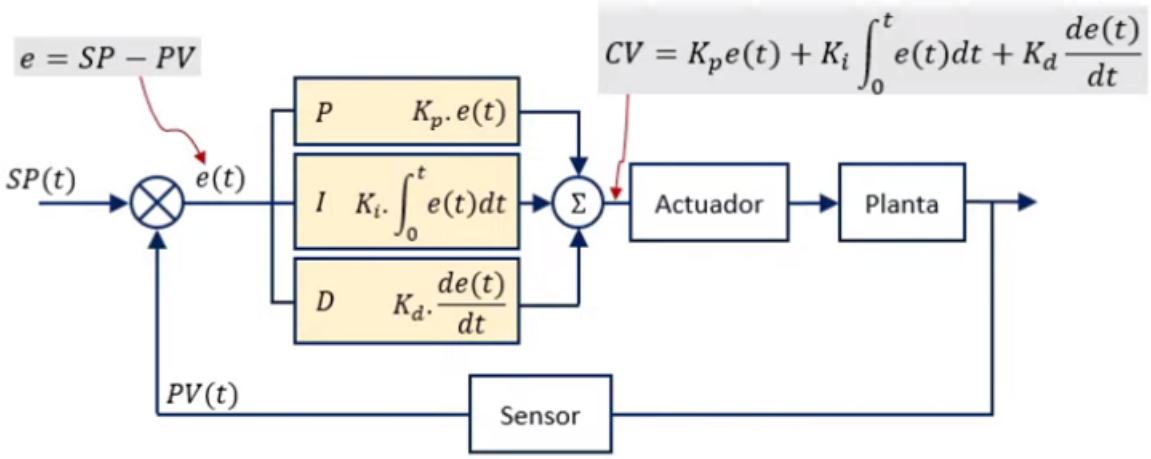
analogRead(POT_pin): Đọc giá trị từ potentiometer, chuyển đổi giá trị sang RPM (vòng/phút).

error = sp - pv: Tính sai số giữa setpoint và process variable.

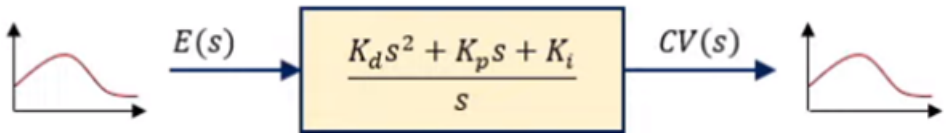
PID Control Calculation:

```
cv = cv1 + (Kp + Kd / Tm) * error + (-Kp + Ki * Tm - 2 * Kd / Tm) *
   error1 + (Kd / Tm) * error2;
cv1 = cv;
error2 = error1;
error1 = error;
```

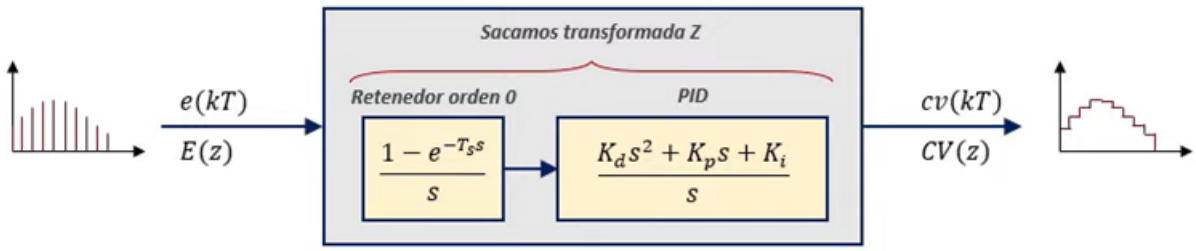
Tính toán giá trị điều khiển theo thuật toán PID:



Áp dụng biến đổi **Laplace**, chúng ta sẽ có:



Nếu chúng ta muốn rời rạc hóa bộ điều khiển này, thì chúng ta cần đặt một bộ giữ ở đầu vào, trong trường hợp này có thể là một bộ giữ cấp 0 (ZOH)



Chúng ta thực hiện biến đổi Z:

$$PID_d = \frac{CV(z)}{E(z)} = z[\frac{1-e^{-T_s s}}{s}(\frac{K_d s^2+K_p s+K_i}{s})]$$
$$= z[\frac{1-e^{-T_s s}}{s}(K_p + \frac{K_i}{s} + K_d s)]$$
$$= (1-z^{-1})z[\frac{K_p}{s} + \frac{K_i}{s^2} + K_d]$$
$$= (1-z^{-1})[\frac{K_p}{(1-z^{-1})} + \frac{K_i T_s z^{-1}}{(1-z^{-1})^2} + \frac{K_d}{T_s}]$$
$$\frac{CV(z)}{E(z)} = \frac{K_p(1-z^{-1}+K_i T_s z^{-1}+\frac{K_d}{T_s}(1-z^{-1})^2}{(1-z^{-1})}$$

Phương trình chuyển đổi Z của PID liên tục::

$$\frac{CV(z)}{E(z)} = K_p + K_i T_s \frac{1}{z-1} + \frac{K_d}{T_s}\frac{z-1}{z}$$

Lưu ý rằng các hằng số Kp, Fd và Ki được lấy từ một hệ thống liên tục.

$$CV(x) - CV(z)z^{-1} = K_p E(z) - K_p E(z)z^{-1} + K_i T_s E(z)z^{-1} + \frac{K_d}{T_s}E(z) + \frac{K_d}{T_s}E(z)z^{-2} - 2\frac{K_d}{T_s}E(z)z^{-1}$$

$$CV(x) - CV(z)z^{-1} = (K_p + \frac{K_d}{T_s})E(z) + (-K_p + K_i T_s - 2\frac{K_d}{T_s})E(z)z^{-1} + \frac{K_d}{T_s}E(z)z^{-2}$$

Phương trình cập nhật PID rời rạc:

$$cv(n) = cv(n-1) + \left(K_p + \frac{K_d}{T_s}\right)e(n) + \left(-K_p + K_i T_s - 2\frac{K_d}{T_s}\right)e(n-1) + \frac{K_d}{T_s}e(n-2)$$

Giới Hạn Đầu Ra PID:

```
if (cv > 500) {cv = 500.0;}

if (cv < 30.0) {cv = 30.0;}
```

Đảm bảo giá trị điều khiển nằm trong khoảng cho phép.

Áp Dụng Điều Khiển và In Dữ Liệu:

```
analogWrite(PWM_pin, cv * (255.0 / 500.0));
```

```
Serial.print("Setpoint (RPM): ");
Serial.print(sp);
Serial.print(", Process Variable (RPM): ");
Serial.println(pv);
delay(100);
```

Áp dụng giá trị điều khiển lên chân PWM.

In thông số setpoint và process variable lên Serial monitor.

Đợi 100ms để tránh in quá nhanh trên Serial.

Hiển Thị Dữ Liệu Lên LCD:

```
lcd.setCursor(0, 0);
lcd.print("SP (RPM): ");
lcd.print(sp);
lcd.setCursor(0, 1);
lcd.print("PV (RPM): ");
lcd.print(pv);
```

Hiển thị setpoint và process variable lên màn hình LCD.

Hàm Ngắt "Interruption":

```
void interruption() {contador++;}
```

Khi có sự kiện ngắt trên chân encoder (RISING), tăng biến contador.

Chương 4:

THI CÔNG MẠCH

4.1 Vẽ sơ đồ nguyên lý

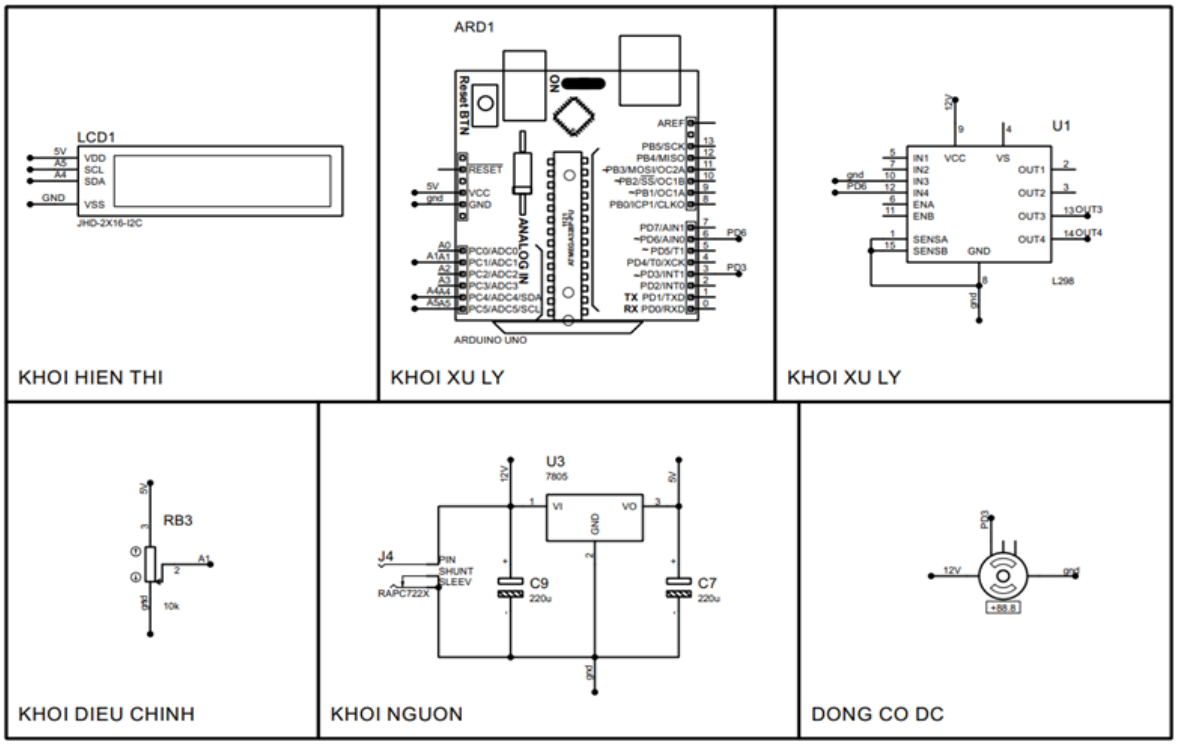
Phần mềm sử dụng để vẽ mạch:

Phần mềm Proteus là một môi trường mô phỏng vi mạch điện tử (Electronic Design Automation - EDA) và mô phỏng hệ thống nhúng (embedded systems) được phát triển bởi công ty Labcenter Electronics Ltd. Proteus được sử dụng rộng rãi trong việc thiết kế và mô phỏng các vi mạch điện tử, PCB (Printed Circuit Board), và hệ thống nhúng.

PROTEUS

Hình 4.1: Phần mềm proteus

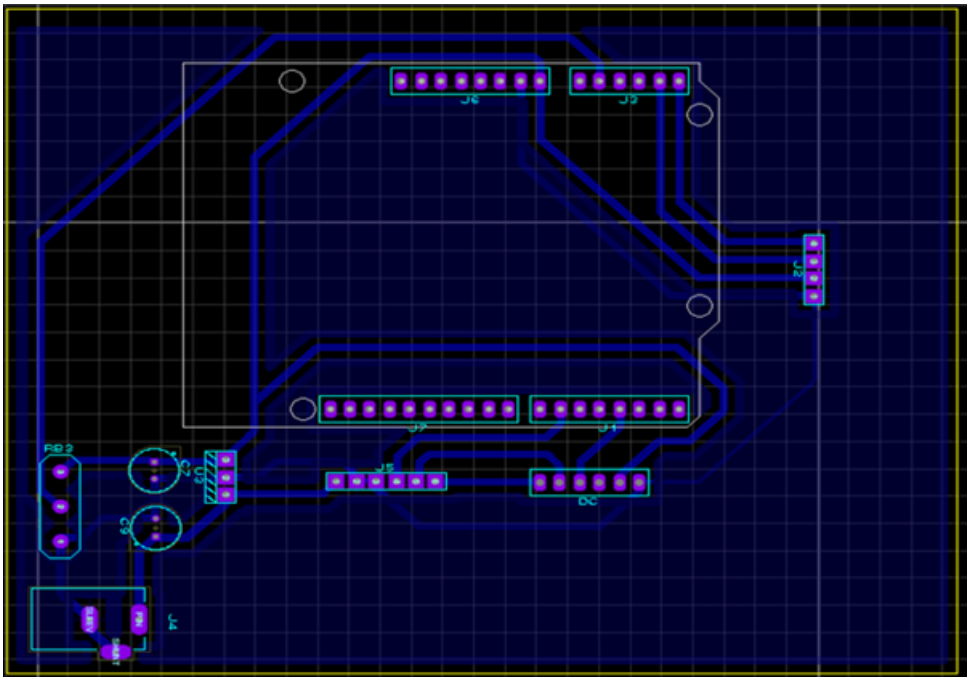
Sơ đồ nguyên lý:



Hình 4.2: Sơ đồ nguyên lý toàn mạch

4.2 Vẽ PCB

Đường nguồn chúng ta chọn T50 còn các đường còn có thể chon từ T15 đến T30.

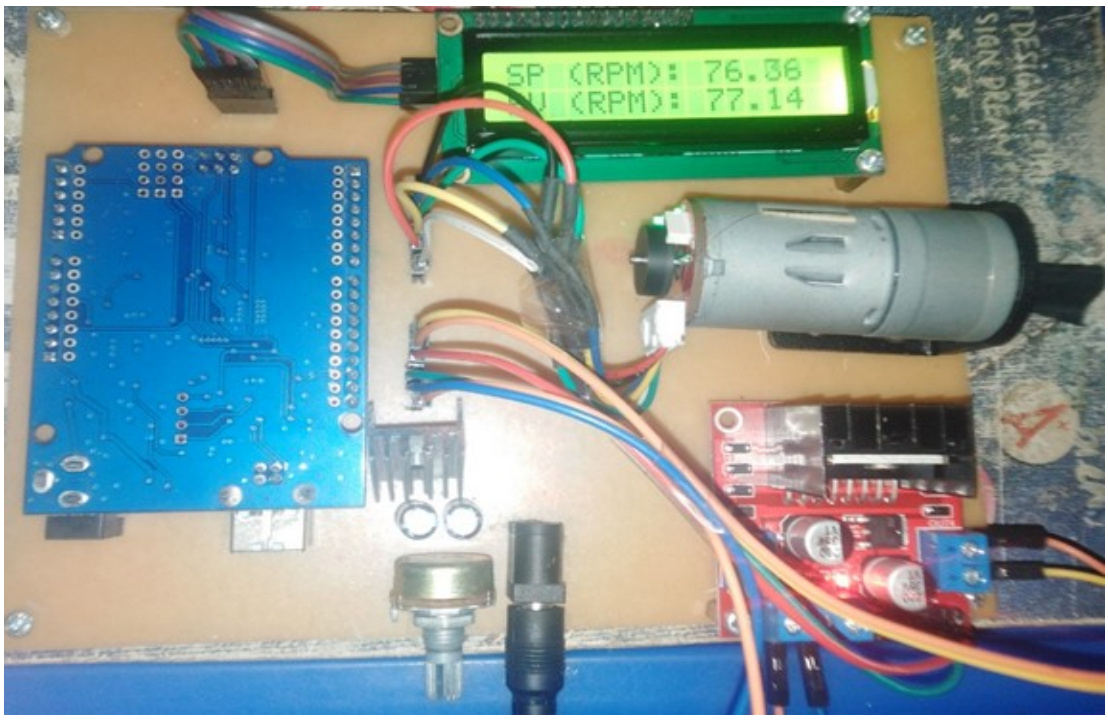


Hình 4.3: Sơ đồ PCB toàn mạch

4.3 Gia công mạch và lắp ráp kiểm tra mạch

Bảng 4.1: Bảng chức năng của các linh kiện

Linh kiện	Số lượng
Arfuino UNO R3	1
Module chuyển đổi I2C	1
Màn hình LCD	1
Motor DC tích hợp encoder	1
Module Drier L298N	1
Jack DC	1
IC ổn áp LM7805	1
Biến trở 10 kohm	1
Tụ hóa 220μF	2



Hình 4.4: Mạch thi công

Mạch đã được nhóm hàn và lắp ráp hoàn chỉnh, chắc chắn. Các dây nối và mối hàn khá ổn định và chắc chắn, cầm chắc tay.

4.3.1 Điều chỉnh thông số PID

Nhomq sử dụng phướng pháp thủ công và thực nghiệm để tìm ra thông số K_p, K_i, K_d cho hệ thống:

- **Thông số** K_p: Chọn K_p trước, điều chỉnh Kp sao cho thời gian đáp ứng đủ nhanh, chấp nhận độ vọt lố nhỏ.
- **Thông số** K_d: Thêm thành phần D để loại độ vọt lố, tăng K_d từ từ, thử nghiệm và chọn giá trị thích hợp. Steady state error có thể sẽ xuất hiện.
- **Thông số** K_i: Thêm thành phần I để giảm steady state error. Tăng K_i từ bé đến lớn để giảm steady state error đồng thời không để cho độ vọt lố xuất hiện trở lại.

Chương 5:

KẾT QUẢ THỰC HIỆN VÀ KẾT LUẬN

5.1 Kết quả thực hiện

Những kết quả mà nhóm đã đạt được:

- Tìm hiểu nguyên lý điều khiển, hoạt động của động cơ DC, encoder.
- Nghiên lý thuyết, cách hoạt động và hiệu chỉnh của bộ điều khiển PID.
- Thi công hoàn thiện và điều khiể ổn định động cơ DC.
- Lập trỉnh được board Arduino.

Ngoài ra thì nhóm cũng chưa hoàn thành được những nhiệm vụ sau:

- Dù đã cố gắng nhưng nhóm vẫn chưa hiểu hết tất cả về bộ điều khiển PID.
- Do sử dụng phương pháp hiệu chỉnh PID thủ công nên động cơ DC chưa thực sự ổn định.
- Khi chỉnh giá trị đặt giá trị bé hơn 10rpm thì động cơ không chạy.

5.2 Kết luận

Mạch chạy khá ổn định. Nhóm đã thực hiện việc lập trình và điều khiển thành công trên board Arduino. Dự án đã mang lại kiến thức và kinh nghiệm quý báu về điều khiển động cơ và sử dụng bộ điều khiển PID.

Nhưng nhóm còn gặp khó khăn trong việc hiểu đầy đủ về bộ điều khiển PID. DO nhóm sử dụng phương pháp hiệu chỉnh thủ công. nên chưa đạt được sự ổn định mong muốn khi giá trị đặt thấp.

Mặc dù còn gặp nhiều khó khăn nhưng đồ án này đã mang lại kiến thức và kinh nghiệm về điều khiển động cơ và sử dụng bộ điều khiển PID..

5.3 Hướng phát triển

Cần tiếp tục nghiên cứu sâu hơn về bộ điều khiển PID để cải thiện hiệu suất và ổn định hơn. Xem xét sử dụng phương pháp hiệu chỉnh PID tự động để tối ưu hóa tham số. Kiểm tra và giải quyết vấn đề khi động cơ không chạy ở các giá trị đặt thấp.

TÀI LIỆU THAM KHẢO

[1] PGS.TS Trần Thu Hà, Trương Thị Bích Ngà, Nguyễn Thị Lưỡng, Bùi Thị Tuyết Đan, Phù Thị Ngọc Hiếu, Dương Thị Cẩm Tú, "Điện tử cơ bản", Xuất bản ĐH Quốc Gia, Tp.HCM, 2013.

[2] Nguyễn Thị Phương Hà, Huỳnh Thái Hoàng, "Lý thuyết điều khiển tự động", Xuất bản ĐH Quốc Gia, Tp.HCM, 2003.

[3] Lại Khác Lãi, Nguyễn Như Hiền, "Điều khiển số", Xuất bản khoa học và kỹ thuật, Hà Nội, 2007.

[4] Lương Quốc Ân, Phan Ngọc Tường, "THIẾT KẾ VÀ THI CÔNG MÔ HÌNH MÁY BAY QUADCOPTER ", 2017.

[5] Đặng Đăc Công, Đặng Thái Giáp, "Thiết kế hệ thống điều khiển tốc độ động cơ DC sử dụng bộ điều khiển PID và PID mờ (Fuzzy-PID) ", 2013.