ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG

----------o0o----------

ĐỒ ÁN MÔN HỌC

ĐIỀU KHIỂN ĐỘNG CƠ BẰNG THUẬT TOÁN PID THÔNG QUA GIAO DIỆN NGƯỜI DÙNG

GVHD: Bùi Thanh Huyền

Sinh viên thực hiện	MSSV
Trần Hữu Tôn Hoàng Phi Long	2013671
Nguyễn Quang Đại	2012905

TP. HỒ CHÍ MINH, NGÀY 07 THÁNG 08 NĂM 2023

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn chân thành và sâu sắc đến Cô vì sự hướng dẫn, hỗ trợ trong quá trình thực hiện đồ án của chúng em. Dưới đây là lời cảm ơn chúng em muốn truyền đạt:

Trước tiên, chúng em xin bày tỏ lòng biết ơn sâu sắc với Cô vì sự hướng dẫn từ lúc chúng em bắt đầu đồ án cho đến khi hoàn thành nộp báo cáo cuối cùng.

Chúng em cũng muốn thể hiện lòng biết ơn với Cô vì dành thời gian để đọc và đánh giá báo cáo của chúng em. Những góp ý và phản hồi chân thành từ Cô đã giúp chúng em hiểu rõ hơn về những khía cạnh cần cải thiện và mở rộng kiến thức của mình.

Cuối cùng, chúng em xin cam kết tiếp tục nỗ lực hết mình và áp dụng những kiến thức đã học được từ đồ án này vào công việc và học tập của mình. Chúng em sẽ luôn ghi nhớ những lời khuyên và chỉ dẫn quý báu từ Cô để trưởng thành và phát triển trong tương lai.

Xin chân thành cảm ơn và kính chúc Cô sức khỏe, hạnh phúc và thành công trong công việc và cuộc sống.

TÓM TẮT ĐỒ ÁN

Đồ án tập trung vào việc điều khiển vị trí và vận tốc của động cơ DC Servo bằng vi điều khiển STM32, sử dụng thuật toán điều khiển PID và cầu H. Vi điều khiển STM32 được lựa chọn làm nền tảng điều khiển chính, vì đây là vi điều khiển có nhiều chân và đầy đủ các tính năng cần thiết trong quá trình làm đồ án và giúp làm quen với kiến trúc ARM đang phổ biến.

Thuật toán điều khiển PID (Proportional-Integral-Derivative) được sử dụng để điều khiển vị trí và vận tốc của động cơ DC Servo một cách chính xác và ổn định. PID điều chỉnh điện áp đầu vào vào động cơ để điều chỉnh vị trí và vận tốc của nó dựa trên sự sai khác giữa giá trị đặt trước và giá trị đo được. Sự kết hợp giữa các thành phần P, I và D trong PID giúp đạt được hiệu suất tốt và ổn định trong quá trình điều khiển động cơ.

Giao diện Windows Form được sử dụng để tạo môi trường thân thiện cho người dùng để điều khiển động cơ DC Servo. Giao diện này cung cấp các điều khiển và các trường nhập liệu cho phép người dùng thay đổi vị trí và vận tốc mong muốn của động cơ. Khi người dùng thay đổi giá trị, thông tin này được gửi qua kết nối UART đến vi điều khiển STM32 để thực hiện điều khiển động cơ theo yêu cầu.

Cầu H (H-Bridge) được sử dụng để kiểm soát động cơ DC Servo, cho phép đảo chiều dòng điện và kiểm soát chiều quay của động cơ. Cầu H giúp đảm bảo động cơ có thể quay cả chiều thuận và chiều nghịch một cách linh hoạt.

Kết quả của đồ án là việc xây dựng thành công một hệ thống điều khiển vị trí và vận tốc động cơ DC Servo bằng vi điều khiển STM32, sử dụng thuật toán điều khiển PID và cầu H, thông qua giao diện Windows Form qua giao tiếp UART. Hệ thống này có thể được ứng dụng trong nhiều lĩnh vực như tự động hóa, robot, hoặc các ứng dụng thực tế khác đòi hỏi điều khiển chính xác và linh hoạt của động cơ DC Servo.

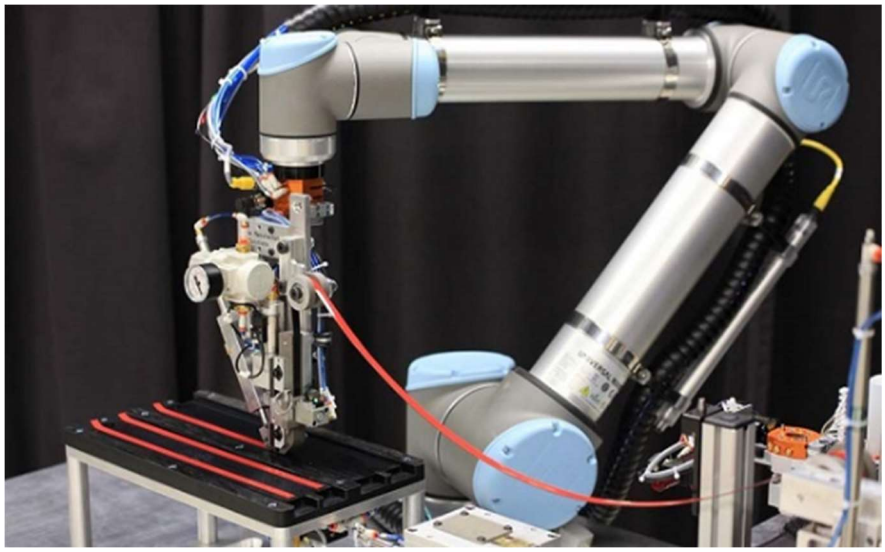
MỤC LỤC

DANH SÁCH HÌNH MINH HỌA

I. GIỚI THIỆU TỔNG QUAN ĐỀ TÀI

1.1 Đặt vấn đề

Hiện nay, động cơ là một thành phần cốt lõi không thể thiếu trong hầu hết các hệ thống điện tử và cơ khí hiện đại. Điều khiển động cơ theo cách chính xác và hiệu quả là một thách thức không nhỏ đối với các kỹ sư và nhà nghiên cứu.



Trong quá trình điều khiển động cơ, có nhiều yếu tố phức tạp cần được xem xét. Đầu tiên là yếu tố độ chính xác, tức là khả năng điều khiển động cơ giữ cho nó hoạt động ở mức độ chính xác cao và theo đúng yêu cầu. Khả năng đáp ứng nhanh chóng và chính xác đòi hỏi hệ thống điều khiển phải có khả năng đo lường và điều chỉnh liên tục, từ đó đảm bảo độ chính xác và độ tin cậy của hệ thống.

Thứ hai, tốc độ phản hồi là một yếu tố quan trọng khác trong việc điều khiển động cơ. Tốc độ phản hồi đo lường khả năng của hệ thống điều khiển để đáp ứng và điều chỉnh một cách nhanh chóng theo sự thay đổi của tải hay điều kiện hoạt động. Một hệ thống điều khiển động cơ tốt sẽ có khả năng điều chỉnh tốc độ phản hồi một cách linh hoạt và ổn định, từ đó giúp đảm bảo hoạt động trơn tru và chính xác.

Cuối cùng, yếu tố độ ổn định cũng rất quan trọng trong điều khiển động cơ. Độ ổn định của một hệ thống đo lường khả năng của nó để duy trì hoạt động ổn định và chống lại các sai số và nhiễu. Đối với một hệ thống điều khiển động cơ, độ ổn định là yếu tố quyết định để đảm bảo hoạt động an toàn và hiệu quả, đồng thời giảm thiểu mức độ lỗi và nhiễu gây ảnh hưởng đến chất lượng sản phẩm hoặc tiến trình sản xuất.

Chính vì những yếu tố phức tạp và quan trọng như vậy, việc áp dụng thuật toán là điều cần thiết và quan trọng, hiện nay có nhiều loại thuật toán mới như điều khiển thích nghi, điều khiển mờ, thuật toán di truyền,… .Nhưng việc dùng thuật toán PID (Proportional-Integral-Derivative) vẫn rất phổ biến và quan trọng trong các ứng dụng công nghiệp và tự động hóa. Thuật toán PID cung cấp khả năng điều khiển linh hoạt và chính xác, tăng cường khả năng đáp ứng nhanh chóng và ổn định, giúp đạt được mức độ chính xác cao và khả năng điều khiển tốt hơn cho các hệ thống động cơ.

1.2 Hướng giải quyết

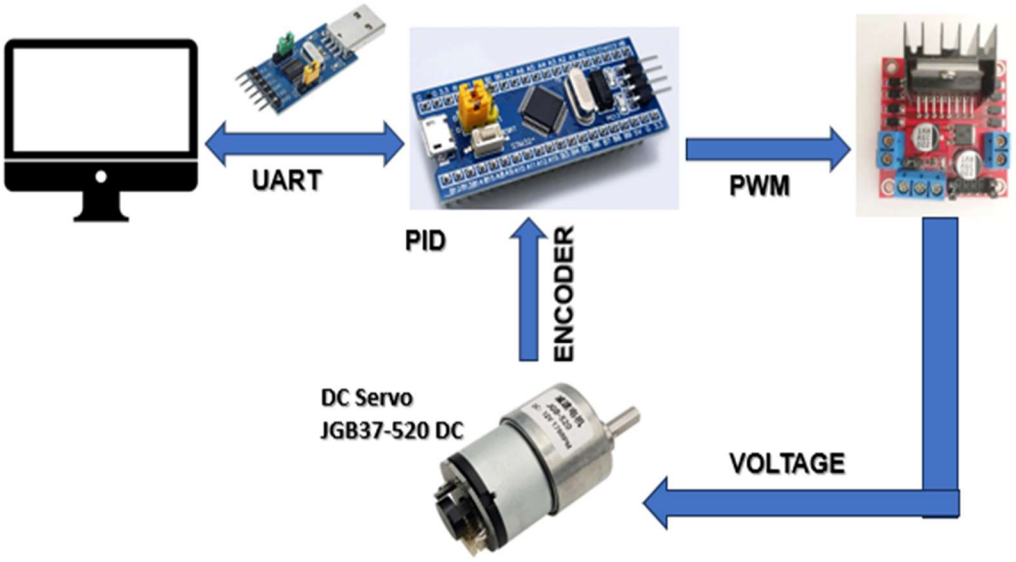


Figure 1: Sơ đồ toàn bộ hệ thống điều khiển

Để giải quyết bài toán điều khiển chúng ta sẽ sử dụng vi điều khiển STM32F103C8T6 để điều khiển động cơ DC servo thông qua mạch cầu H, hỗ trợ chuyển đổi hướng quay và kiểm soát tốc độ. Thuật toán PID sẽ được triển khai để tính toán lệnh điều khiển dựa trên sai số giữa vị trí hiện tại và vị trí mục tiêu, cũng như sai số giữa vận tốc hiện tại và vận tốc mục tiêu.

Giao tiếp giữa vi điều khiển và máy tính sẽ được thực hiện qua cổng UART, giúp truyền và nhận dữ liệu giữa hai thiết bị. Trên máy tính, chúng ta sẽ thiết kế giao diện trực quan trên Windows Form của Visual Studio, cho phép người dùng đặt vị trí và vận tốc mục tiêu của động cơ DC servo và theo dõi các thông số điều khiển như vị trí và tốc độ.

II. GIỚI THIỆU VÀ PHƯƠNG PHÁP ĐIỀU KHIỂN ĐỘNG CƠ

2.1 Động cơ DC:

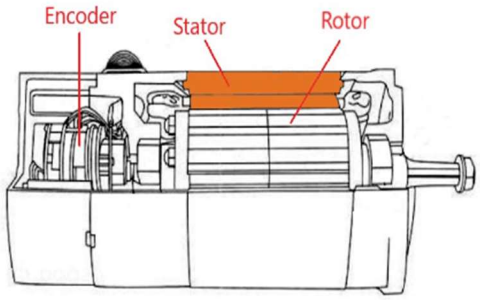


Figure 2: Cấu tạo động cơ DC

2.1.1 Cấu tạo của động cơ DC servo JGB37-520:

Rotor: Bộ phận quay trong động cơ, được nối trực tiếp với trục quay và có nam châm tích hợp.

Stator (bộ từ): Bộ phận tĩnh trong động cơ, bao gồm các cuộn dây dẫn điện được xếp chồng lên nhau và đặt xung quanh rotor.

Encoder (bộ mã hóa): Đây là một thiết bị cảm biến được gắn trên trục của động cơ để theo dõi vị trí và tốc độ quay của rotor. Thông tin từ encoder được sử dụng để truyền dữ liệu về cho vi điều khiển cho biết số xung đọc được khi động cơ hoạt động.

Nguyên lý hoạt động của động cơ DC servo JGB37-520:

- Động cơ DC servo hoạt động dựa trên nguyên tắc phản hồi điều khiển (feedback control) để đạt được mục tiêu hoạt động như vị trí, tốc độ...
- Đầu tiên, điện áp DC được cấp vào cuộn dây của stator, tạo ra từ trường từ stator. Khi dòng điện chạy qua cuộn dây này, nó tạo ra lực tác động lên rotor do sự tương tác giữa từ trường từ stator và từ trường từ rotor
- Encoder liên tục theo dõi vị trí của rotor và gửi tín hiệu phản hồi về hệ thống điều khiển.
- Hệ thống điều khiển sẽ so sánh tín hiệu từ encoder với mục tiêu (ví dụ: vị trí hay tốc độ) đã đặt trước đó. Nếu có sự chênh lệch giữa vị trí hiện tại và mục tiêu, hệ thống sẽ điều chỉnh điện áp cấp vào cuộn dây stator để tạo ra lực điều khiển phù hợp.
- Quá trình này tiếp tục lặp lại, giúp giữ cho động cơ servo đạt được độ chính xác và ổn định trong việc giữ vị trí hoặc điều khiển tốc độ theo yêu cầu.

Điều này cho phép động cơ DC servo JGB37-520 có khả năng điều khiển chính xác vị trí và tốc độ quay của nó, làm cho nó phù hợp cho các ứng dụng cần độ chính xác cao như trong các máy móc tự động hóa, robot, các hệ thống giám sát vị trí, và các ứng dụng cần điều khiển vị trí chính xác.

2.1.2 Mô hình hóa động cơ

Mô hình hóa động cơ DC servo có thể được thực hiện bằng cách sử dụng các phương trình và tham số để miêu tả các tương tác giữa các thành phần và đặc tính hoạt động của động cơ.

Mô hình toán học cho động cơ DC servo:

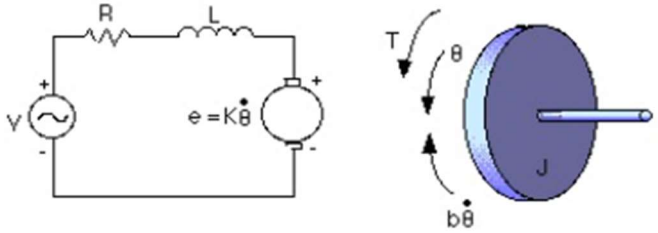


Figure 3: Mô hình động cơ DC

R: Điện trở phần ứng

L: Điện cảm phần ứng

J: Momen quán tính trên trục động cơ

b: Hệ số ma sát

Kt: Hằng số momen xoắn

Ke: Hằng số sức phản điện

Ngõ vào điện áp nguồn U. Ngõ ra vị trí trục quay θ và tốc độ động cơ $\dot{\theta}$

Momen phát sinh trên trục động cơ: $T_m = K_t * I$ (I là dòng điện nguồn cấp)

Sức điện động trên động cơ: $E = K_e * I$

Theo định luật Newton II ta có:

$$J\frac{d^2\theta}{dt^2} + b\frac{d\theta}{dt} = T_m$$

Theo định luật Kirchhoff ta có:

$$L\frac{di}{dt} + Ri = U - K_e * \dot{\theta}$$

Biến đổi Laplace các phương trình trên ta có:

$$s(Js + b) * \theta(s) = Kt * I(s)$$

$$(Ls + R) * I(s) = U(s) - Ke * s\theta(s)$$

Từ đó, hàm truyền vị trí và tốc độ động cơ theo điện áp nguồn được tính như sau:

$$\frac{\theta(s)}{U(s)} = \frac{1}{s} * \frac{Kt}{(Js+b)(Ls+R)+Ke*Kt}$$

$$\frac{\dot{\theta}(s)}{U(s)} = \frac{Kt}{(Js+b)(Ls+R)+Ke*Kt}$$

Ở tần số thấp, điện cảm động cơ rất nhỏ nên có thể bỏ qua, khi đó ta sẽ được phương trình rút gọn như sau

$$\frac{\theta(s)}{U(s)} = \frac{1}{s} * \frac{Kt}{RJs+(bR+KeKt)} = \frac{1}{s} * \frac{K}{Ts+1}$$

$$\frac{\dot{\theta}(s)}{U(s)} = \frac{Kt}{RJs+(bR+KeKt)} = \frac{K}{Ts+1}$$

Với độ lợi và thời hằng được tính như sau:

$$K = \frac{Kt}{bR+KeKt} \qquad T = \frac{JR}{bR+KeKt}$$

2.2 Phương pháp điều khiển động cơ

Động cơ DC servo sử dụng PWM và cầu H để kiểm soát chính xác và linh hoạt cho động cơ. Điều khiển PWM là một kỹ thuật dựa trên độ rộng xung, thông qua đó tốc độ quay của động cơ có thể được điều chỉnh. Tín hiệu PWM được tạo ra từ bộ điều khiển hoặc vi xử lý dựa trên phản hồi từ cảm biến (encoder) để xác định

vị trí hoặc tốc độ quay hiện tại của động cơ. Sau đó, tín hiệu PWM này được chuyển qua cầu H để điều chỉnh dòng điện đầu vào vào động cơ.

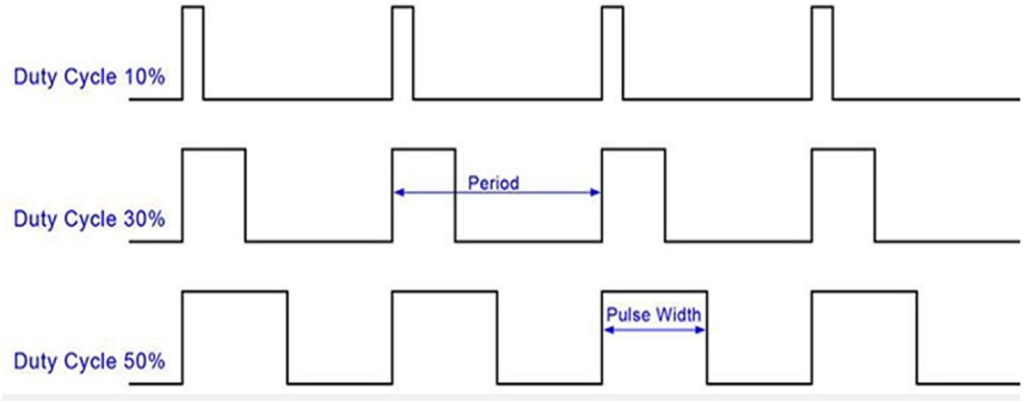


Figure 4: Nguyên lí điều chế độ rộng xung PWM

Cầu H là một mạch điện tử đặc biệt, cho phép đảo ngược hướng dòng điện trong động cơ. Nó gồm bốn chân kết nối độc lập, có thể kết nối với nguồn điện dương và âm, tạo điều kiện để động cơ có thể quay cả hai hướng. Tóm lại, trong phương pháp này, tín hiệu PWM từ bộ điều khiển được chuyển qua cầu H để điều chỉnh điện áp vào và chiều của động cơ.

III. GIỚI THIỆU VỀ THUẬT TOÁN PID

3.1 Các thành phần của thuật toán PID

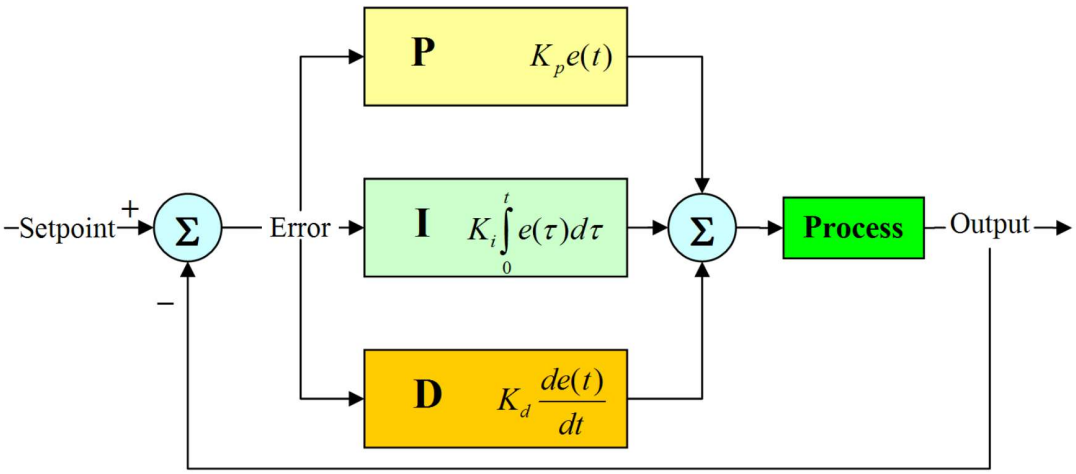


Figure 5: Sơ đồ khối bộ điều khiển PID

Luật điều khiển bộ PID (Proportional-Integral-Derivative) là một phương pháp điều khiển phản hồi phổ biến trong hệ thống tự động. Nó được sử dụng để điều chỉnh và điều khiển các biến số trong hệ thống để đạt được mục tiêu hoạt động mong muốn.

Bộ điều khiển PID sử dụng ba thành phần chính: tỷ lệ (P), tích phân (I) và vi phân (D). Công thức chung của bộ điều khiển PID được biểu diễn như sau:

$$u(t) = Kp * \left(e(t) + \frac{1}{Ti} \int_0^t e(t)dt + Td \frac{de(t)}{dt} \right)$$

Trong đó:

- ***Khâu Tỉ lệ (P):***

$$Pout = K_p * e(t)$$

Khâu này tác động làm thay đổi giá trị đầu ra, tỉ lệ với giá trị sai số hiện tại. Nó quyết định mức độ tác động của sai số lên đầu ra của bộ điều khiển. Khi KP càng lớn, đầu ra sẽ tăng nhanh hơn khi sai số càng lớn. Tuy nhiên, quá mức KP lớn có thể gây dao động hoặc ổn định không tốt. Do đó, việc lựa chọn KP phải cân nhắc giữa độ nhạy và ổn định của hệ thống.

- ***Khâu Tích phân (I):***

$$Iout = K_i * \int_0^t e(t)dt$$

Thành phần tích phân trong điều khiển có ưu điểm là tăng tín hiệu điều khiển khi có sai lệch dương và giảm tín hiệu điều khiển khi có sai lệch âm. Điều này đảm bảo rằng sai lệch sẽ được tiêu giảm và đạt đến giá trị ổn định ($e(t) = 0$) trong trạng thái cân bằng.

Tuy nhiên, nhược điểm của thành phần tích phân là phải chờ một khoảng thời gian để sai lệch e(t) giảm về 0, dẫn đến tác động của bộ điều khiển chậm hơn và

đáp ứng không linh hoạt. Ngoài ra, nếu hệ số Ki (hệ số tỷ lệ tích phân) quá lớn, thành phần tích phân có thể làm mất ổn định hệ thống, gây dao động đảo chiều liên tục trong thời gian ngắn.

- ***Khâu Vi phân (D):***

$$Dout = Kd*\frac{de(t)}{dt}$$

Khâu vi phân (D) trong bộ điều khiển PID được sử dụng để giảm đáp ứng chậm và ổn định hơn của hệ thống. Nó dựa trên việc tính toán đạo hàm của sai lệch e(t) (hiệu của giá trị đặt trước và giá trị đo được). Khâu vi phân cho phép bộ điều khiển dự đoán tốc độ thay đổi của e(t) và điều chỉnh đáp ứng của hệ thống dựa trên tốc độ thay đổi này. Ưu điểm là đáp ứng mạnh mẽ hơn đối với sai lệch e(t), giúp hệ thống đạt được giá trị đặt mục tiêu nhanh hơn và giảm thời gian ổn định

Nhược điểm là nhạy cảm đối với nhiễu: Kd có thể làm tăng đáng kể phản ứng của hệ thống đối với nhiễu. Nếu giá trị Kd được đặt quá cao, các biến đổi nhỏ trong tín hiệu đầu vào có thể dẫn đến phản ứng không mong muốn hoặc dao động không ổn định.

3.2 Hiệu chỉnh bộ thông số PID

Hiệu chỉnh bộ điều khiển PID là quá trình điều chỉnh các tham số của bộ điều khiển PID để đạt được hiệu suất tối ưu của hệ thống điều khiển. Có nhiều phương pháp và kỹ thuật khác nhau để hiệu chỉnh PID, nhưng phương pháp phổ biến nhất bao gồm:

Phương pháp thủ công (tự điều chỉnh): Đây là phương pháp đơn giản nhưng thường tốn thời gian buộc đòi hỏi người kĩ sư phải có chuyên môn cao, ước tính gần đúng nhất các thông số thích hợp thì quá trình tìm ra thông số tốt nhất sẽ nhanh chóng. Điều chỉnh từng tham số PID (Kp, Ki, Kd) một cách tuần tự và đánh giá

hiệu quả của hệ thống. Dựa trên kết quả, tham số được điều chỉnh lại cho đến khi đạt được hiệu suất tốt nhất.

Phương pháp Ziegler-Nichols: Đây là một phương pháp tiêu chuẩn và phổ biến hơn. Phương pháp này bắt đầu bằng việc tăng dần hệ số Kp cho đến khi hệ thống bắt đầu dao động với biên độ không đổi. Sau đó, các giá trị Kp, Ki và Kd được tính toán dựa trên các quy tắc cụ thể, tùy thuộc vào loại điều khiển PID (P, PI, PD hay PID).

Việc lựa chọn tham số cho bộ điều khiển PID phụ thuộc vào đối tượng điều khiển và phương pháp xác định thông số. Tuy nhiên, kinh nghiệm cũng đóng vai trò quan trọng trong quá trình này:

Đáp ứng vòng kín	**Thời gian lên**	**Vọt lố**	**Thời gian xác lập**	**Sai số xác lập**
Kp	Giảm	Tăng	Thay đổi nhỏ	Giảm
Ki	Giảm	Tăng	Tăng	Loại bỏ
Kd	Thay đổi nhỏ	Giảm	Giảm	Thay đổi nhỏ

Figure 6: Ảnh hưởng của mỗi bộ điều khiển Kp, Ki và Kd

3.3 Ưu điểm, khuyết điểm của bộ điều khiển PID

Điều khiển động cơ bằng PID là một phương pháp phổ biến được sử dụng trong hệ thống điều khiển tự động. Dưới đây là một số ưu điểm và khuyết điểm của việc sử dụng điều khiển động cơ bằng PID:

❖ **Ưu điểm:**

- Đơn giản và dễ triển khai: PID là một thuật toán điều khiển đơn giản, dễ hiểu và dễ triển khai trong nhiều ứng dụng điều khiển động cơ.

- Độ ổn định tốt: PID có khả năng điều chỉnh độ ổn định của hệ thống động cơ. Phương pháp này sử dụng các thông số như tỷ lệ, tích phân và vi phân để điều chỉnh đáp ứng của hệ thống và đạt được sự ổn định mong muốn.

- Khả năng tương thích và tích hợp: PID có thể dễ dàng tích hợp với các hệ thống điều khiển khác, như các cảm biến và bộ điều khiển. Nó có khả năng tương thích với nhiều loại động cơ và ứng dụng khác nhau.

❖ **Khuyết điểm:**

- Phụ thuộc vào điều chỉnh thủ công: Để đạt được hiệu suất tối ưu, PID yêu cầu việc điều chỉnh thủ công các tham số điều khiển. Điều này đòi hỏi kiến thức và kinh nghiệm để tìm ra các giá trị phù hợp, và quá trình điều chỉnh có thể tốn nhiều thời gian và công sức.

- Không linh hoạt: PID có hạn chế trong việc xử lý các tác động và môi trường biến đổi. Khi các điều kiện hoạt động thay đổi đáng kể, PID có thể không đáp ứng một cách linh hoạt và cần điều chỉnh lại các tham số điều khiển.

IV. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

4.1 Động cơ DC JGB37-520

Động cơ DC JGB37-520 là một loại động cơ điện tử trực tiếp (DC) có kích thước nhỏ, thích hợp cho các ứng dụng có không gian hạn chế. Nó có tốc độ quay tối đa 333 vòng/phút khi không có tải đè lên động cơ và 250 vòng/phút khi có tải. Động cơ này có khả năng tạo ra mômen xoắn để xoay các bộ phận cơ khí như bánh răng, vòng bi, ổ đĩa, và các thiết bị khác. Thường được sử dụng trong robot, máy in, thiết bị tự động hóa, máy cắt cỏ, và nhiều ứng dụng khác.

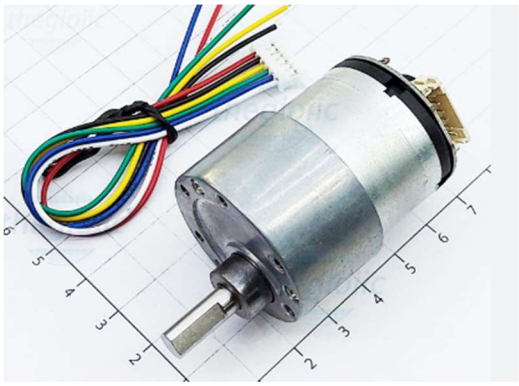


Figure 7: Động cơ DC JGB37-520

Thông số kĩ thuật:

Điện áp định mức	6 -24V/ 12VDC
Dòng tối đa	3A
Công suất	36W
Tốc độ chịu đựng tối đa khi có tải	250 RPM
Tốc độ không tải (sau giảm tốc)	333 RPM
Tỉ lệ Hộp số	1:30
Điện áp nguồn	3.3 - 5VDC
Số xung/vòng (trước giảm tốc)	11
Số xung/vòng (sau giảm tốc)	330

Pha A và pha B trong encoder của động cơ DC servo đóng vai trò quan trọng trong việc cung cấp phản hồi về vị trí và hướng quay của trục quay. Tín hiệu pha A thay đổi từ mức logic thấp lên mức cao hoặc ngược lại mỗi vòng quay hoàn chỉnh, trong khi tín hiệu pha B có pha trễ so với pha A. Kết hợp tín hiệu pha A và pha B giúp bộ điều khiển PID tính toán và điều chỉnh tín hiệu điện áp để duy trì vị trí và tốc độ ổn định của động cơ, đảm bảo hoạt động chính xác và đáng tin cậy của hệ thống.

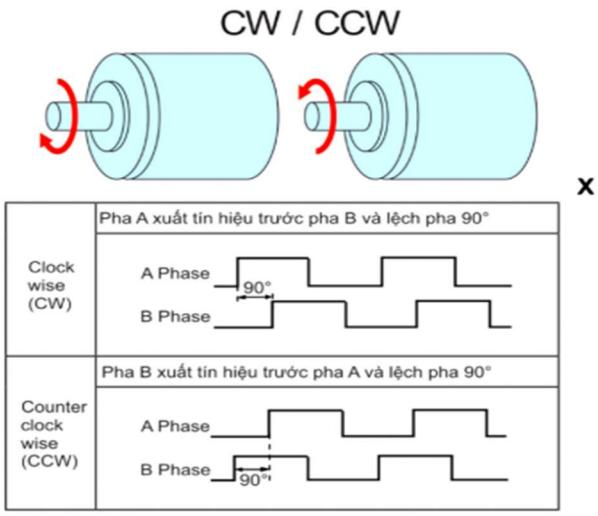


Figure 8: Xác định chiều quay động cơ

Khi ta quay trục Encoder theo chiều kim đồng hồ thì xung A sẽ lệch pha với xung B một góc 90 độ. Ngược lại, nếu ta quay trục Encoder ngược chiều kim đồng hồ thì xung B sẽ xuất tín hiệu trước xung A. Lúc này, xung B sẽ lệch pha xung A góc 90 độ.

4.2 STM32F103C8

Vi điều khiển STM32F103C8T6, là một trong những vi điều khiển phổ biến thuộc dòng sản phẩm STM32 của STMicroelectronics. Đây là một loạt vi điều khiển 32-bit ARM Cortex-M3, phổ biến trong cộng đồng Maker và DIY do giá thành thấp, khả năng mạnh mẽ và tích hợp các tính năng đa dạng.

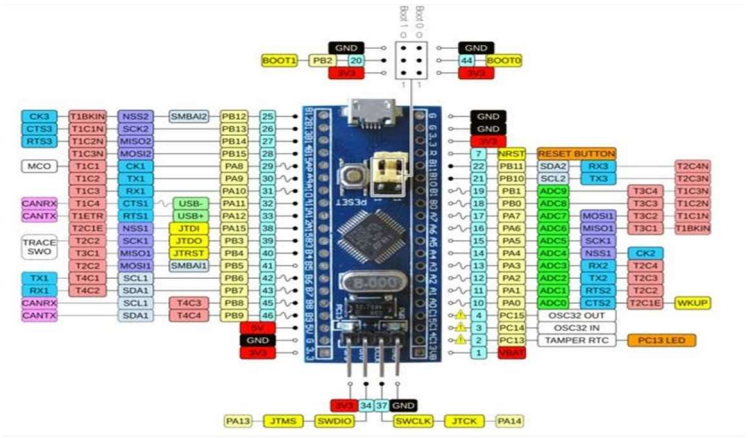


Figure 9: STM32F103C8

Thông số kỹ thuật	Giá trị
Kiến trúc	ARM Cortex-M3
Tốc độ xung nhịp tối đa	72 MHz
Bộ nhớ Flash	64 Kbytes
Bộ nhớ RAM	20 Kbytes
Số lượng GPIO	37
Số lượng USART	3
Giao tiếp USB	1 (Full-Speed)
Giao tiếp PWM	16 (16-bit)
Nguồn cấp	2.0V đến 3.6V
Chế độ tiết kiệm	Hỗ trợ chế độ tiết kiệm năng lượng

Vi điều khiển STM32F103C8T6 được hỗ trợ bởi môi trường phát triển tích hợp (IDE) như STM32CubeIDE, Keil, và Arduino IDE, giúp đơn giản hóa quá trình phát triển phần mềm cho các ứng dụng nhúng.

4.3 Mạch cầu H

Mạch cầu H (H-Bridge) là một bộ điều khiển cơ bản được sử dụng trong nhiều ứng dụng để thay đổi hướng quay của động cơ DC và điều khiển điện áp qua động cơ. Đặc biệt, mạch cầu H thường được sử dụng để điều khiển động cơ servo và động cơ định vị trong các ứng dụng tự động hóa.

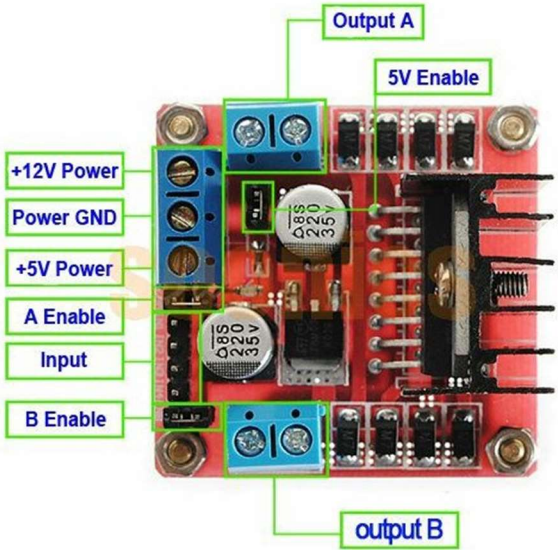


Figure 10: Mạch cầu H

Các bước cơ bản để điều khiển động cơ bằng mạch cầu H:

- Điều khiển hướng quay: Kích hoạt các cầu dao tương ứng để thay đổi hướng dòng điện qua động cơ, điều này sẽ thay đổi hướng quay của động cơ.
- Điều khiển tốc độ: Tăng hoặc giảm cường độ xung PWM qua động cơ.

Mạch cầu H cung cấp sự linh hoạt và kiểm soát tốt cho việc điều khiển động cơ, đồng thời đảm bảo an toàn trong quá trình điều khiển. Điều này làm cho mạch cầu H trở thành một thành phần quan trọng trong các hệ thống điều khiển động cơ DC như động cơ servo.

4.4 UART

UART (Universal Asynchronous Receiver/Transmitter) là giao thức truyền thông phổ biến dùng để truyền và nhận dữ liệu không đồng bộ giữa các thiết bị điện tử.

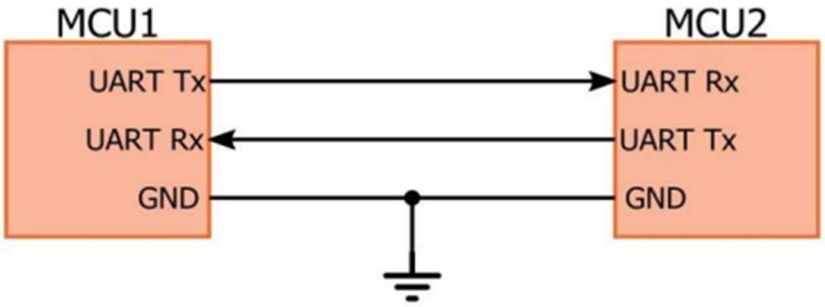


Figure 11: Sơ đồ kết nối UART

Đặc điểm nổi bật của UART bao gồm:

- Không đồng bộ: UART truyền dữ liệu không đồng bộ, không dựa vào tín hiệu đồng hồ chung, mà sử dụng các bit Start và Stop để đồng bộ dữ liệu.

- Dữ liệu tuần tự: Dữ liệu được truyền và nhận một bit một lần theo trình tự từ trái sang phải. Mỗi byte dữ liệu thường có bit Start và Stop đi kèm.
- Tốc độ baud: Tốc độ baud đo lường số lần truyền bit trong một giây. Cả thiết bị gửi và nhận dữ liệu phải được cấu hình với cùng một tốc độ baud.
- Đơn giản và linh hoạt: UART rất dễ triển khai và linh hoạt trong việc truyền thông giữa các thiết bị.
- Không kiểm tra lỗi: UART không cung cấp kiểm tra lỗi hoặc khả năng chống sai số trong quá trình truyền dữ liệu, mà chỉ dựa vào các bit Start và Stop để phát hiện lỗi.

Giao thức UART được ứng dụng rộng rãi trong việc kết nối vi điều khiển với cảm biến, máy tính thông qua cổng COM.

4.5 Nguồn tổ ong 12V

Nguồn tổ ong 12V chuyển đổi từ nguồn AC (điện xoay chiều) có điện áp đầu vào là 220V sang nguồn DC (điện một chiều) với điện áp đầu ra là 12V. Quá trình này được thực hiện thông qua một bộ biến đổi và mạch điện tử trong nguồn tổ ong.

Figure 12: Nguồn tổ ong 12V

Bên trong nguồn tổ ong, có các linh kiện như biến trở, điốt, tụ điện, IC điều khiển và bộ truyền đổi DC-DC để chuyển đổi từ nguồn AC sang DC. Điều này cho

phép nguồn tổ ong cung cấp điện áp DC ổn định và phù hợp với các thiết bị điện tử và được dùng để cấp nguồn ổn định cho mạch cầu H L298 của đồ án.

4.6 Tải

Động cơ DC servo JGB37-520 (12V/333RPM) có các thông số như sau:

Lực kéo Moment định mức: 3.5KG.CM - Đây là mô-men lực tối đa mà động cơ servo có thể tạo ra khi hoạt động với dòng điện định mức. Mô-men lực là sức kéo tối đa mà động cơ có thể cung cấp tại khoảng cách từ trục quay đến điểm áp dụng lực.

Lực kéo Moment tối đa: 5KG.CM - Đây là mô-men lực tối đa mà động cơ servo có thể tạo ra khi hoạt động với dòng điện tối đa hoặc trong tình huống đặc biệt. Nó cho biết khả năng động cơ đối phó với tải nặng nhất mà nó có thể đạt được.

Công thức lực kéo moments:

Lực kéo Moment = R* m * Trọng lượng gia tốc (g) (đơn vị: N.m)

Trong đó:

R là khoảng cách từ trục động cơ đến tải (bán kính bánh xe) (đơn vị: mét)

M là khối lượng tải (đơn vị là KG)

Trọng lượng gia tốc (**g**) (đơn vị m/s^2)

-> Với Moment định mức: 3.5KG.CM thì động cơ có lực kéo là:

1/100 * 3.5 * 9.8 = 0.343 (N.m)

Trong bài này tụi em dùng tải với hình ảnh như sau:

Figure 13: Tải động cơ

Thông số tải:

Bán kính: 4cm

Khối lượng: 0.25 kg

->Lực kéo moment = 0.25 * 4/100 * 9.8 = 0.098(N.m) ($\approx \frac{\text{Lực kéo moment định mức}}{3}$)

Lực kéo moment tải cung cấp (0.098N.m) <= Lực kéo moment định mức (0.343N.m)

=> Thỏa

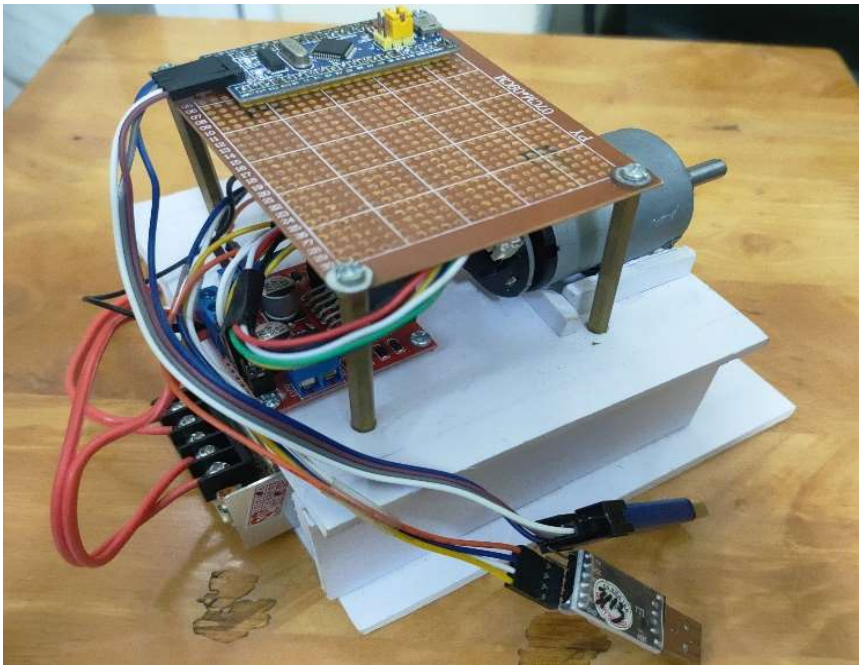


Figure 14: Sản phẩm hoàn thiện

V. LẬP TRÌNH VI ĐIỀU KHIỂN VÀ GIAO DIỆN NGƯỜI DÙNG

❖ STM32CubeMx

Trong quá trình thực hiện đồ án chúng em đã dùng STM32CubeMX là công cụ phần mềm miễn phí của STMicroelectronics, dùng để cấu hình và tạo mã cho vi điều khiển STM32 thông qua giao diện đồ họa. Nó tự động tạo mã C chuẩn sau khi người dùng hoàn tất cấu hình vi điều khiển. STM32CubeMX hỗ trợ nhiều dòng vi điều khiển STM32 và cung cấp các Middleware để tích hợp vào dự án. Điều này giúp tiết kiệm thời gian và công sức trong phát triển phần mềm cho vi điều khiển STM32, đồng thời đảm bảo tính ổn định và đáng tin cậy của hệ thống.

❖ Visual Studio_Window Form (Giao diện C#)

Window_Form cung cấp môi trường lập trình đồ họa người dùng (GUI) dễ sử dụng và trực quan. Được tích hợp với các công cụ và thư viện của Visual Studio, giúp tạo ra các ứng dụng Windows chất lượng cao với hiệu suất tốt.

Điều khiển động cơ: Ứng dụng Window Form sẽ giúp bạn gửi các tín hiệu điều khiển (như dữ liệu PID, thông số tốc độ và hướng quay) từ máy tính tới vi điều khiển STM32F103 thông qua kết nối UART. Điều này cho phép bạn thực hiện điều khiển động cơ DC servo từ máy tính một cách linh hoạt và dễ dàng.

Ghi nhận và hiển thị dữ liệu: Visual Studio Window Form cũng có thể giúp bạn ghi nhận và hiển thị dữ liệu từ động cơ DC servo, chẳng hạn như tốc độ quay hiện tại và các giá trị đo đạc từ cảm biến phản hồi (nếu có). Điều này giúp bạn theo dõi hiệu suất và điều chỉnh điều khiển động cơ một cách chính xác.Vì vậy Visual Studio_Window Form được nhóm chúng em lựa chọn để thiết kế giao diện cho đồ án điều khiển này ạ.

5.1 FlowChart của hệ thống điều khiển động cơ bằng PID

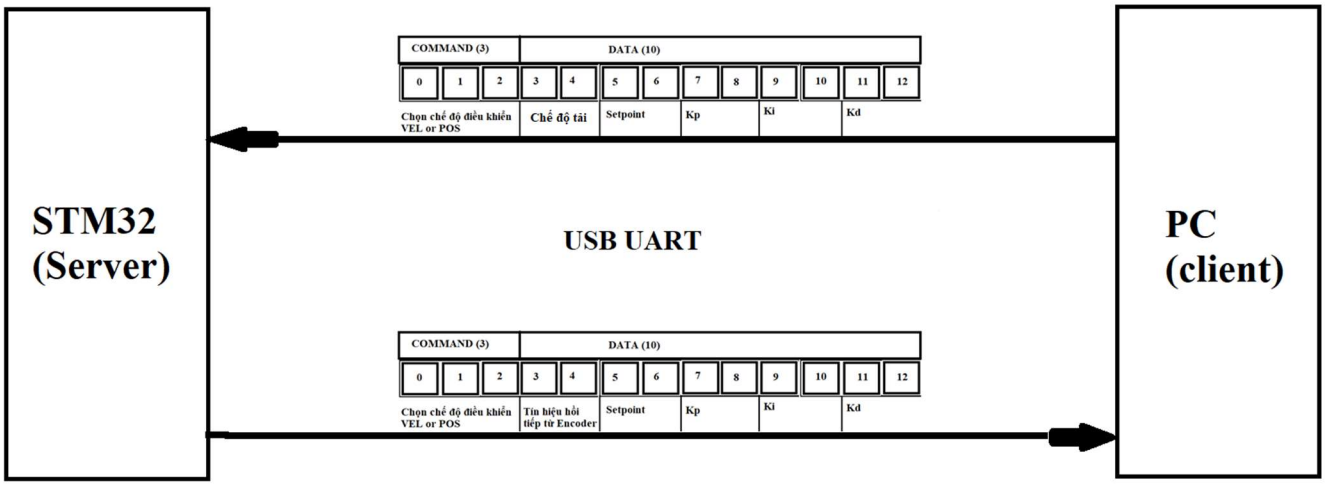


Figure 15: Giao thức bắt tay

Chương trình cho phần cứng STM32

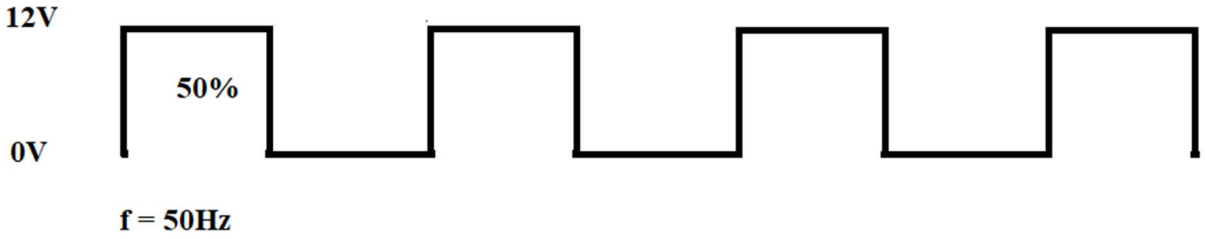
Như đã biết, vi điều khiển đóng vai trò trung tâm để điều khiển hệ thống. Để có thể nạp chương trình vào vi điều khiển ta cần phải cấu hình trên phần mềm STM32CubeMx và Generate Code ra trên Keli C V5 và nạp code xuống vi điều khiển qua ST Link V2. Chương trình của vi điều khiển gồm các khối chức năng sau: Khối map, khối Set Pwm Duty, khối Read Rpm Encoder, khối Count Encoder, khối PID Control, khối Transmit Data, khối Set Motor.

Khối map:

Khối này có nhiệm vụ chuyển đổi tốc độ của động cơ thành phần trăm duty tương ứng để điều chế xung PWM. Ví dụ, nếu tín hiệu điều khiển tương ứng với tốc độ 165 rpm (với giới hạn tối đa là 333 rpm), thì khối Map sẽ chuyển đổi thành 50% duty cycle của xung PWM.

Khối Set Pwm Duty:

Khi duty cycle được định vị là 50%, khối Set Pwm Duty sẽ điều khiển xuất ra một xung PWM với duty cycle là 50%.



Có thể hiểu, với một giá trị duty cycle được set là 50%, khối Set Pwm Duty sẽ tạo ra một xung PWM có thời gian đúng bằng 50% của chu kỳ xung.

Sẽ xuất ra 12*50% = 6V để điều khiển động cơ.

Khối Read Rpm Encoder:

Khối này có chức năng đọc tốc độ vòng/phút (rpm) từ Encoder được tích hợp sẵn trong động cơ. Khối sử dụng một thư viện đặc biệt để đọc xung từ Encoder và tính toán tốc độ dựa trên thông số cấu hình của Encoder.

Trong khối ***Read Rpm Encoder***, xung từ Encoder được đọc và đếm để xác định số lượng xung trong một khoảng thời gian nhất định. Sau đó, thông qua tính toán dựa trên số lượng xung và thông số cấu hình của Encoder, tốc độ vòng/phút (rpm) của động cơ được tính toán.

Để hạn chế nhiễu do sai số của Encoder, khối Encoder Reader được trang bị một bộ lọc thông thấp. Bộ lọc này hoạt động để giảm nhiễu và nhiễu môi trường, đảm bảo rằng tín hiệu đọc từ Encoder là ổn định và chính xác.

Khối Count Encoder:

Có chức năng đọc và đếm xung từ Encoder cho việc điều khiển vị trí động cơ

Khối PID Control:

Khối PID nhận tín hiệu đặt (Setpoint) và các thông số bộ điều khiển từ trên giao diện máy tính gửi xuống, đồng thời nhận tín hiệu hồi tiếp từ Encoder tùy theo chế độ tốc độ hay vị trí, sau đó áp dụng giải thuật PID tính toán ra tính hiệu điều khiển.

Khối Set Motor:

Khối này chứa khối Set Pwm Duty và điều khiển động cơ dựa trên các tham số đầu vào như duty cycle (duty) và hướng quay (dir).

Trong khối ***Set Motor***, khối Set Pwm Duty được sử dụng để điều khiển động cơ thông qua xung PWM. Tham số duty cycle (duty) được truyền vào khối Set Pwm Duty để đặt mức duty cycle của xung PWM, điều chỉnh công suất đầu ra đến động cơ.

Ngoài ra, khối Set Motor cũng nhận tham số hướng quay (dir) để xác định chiều quay thuận, nghịch của động cơ hoặc dừng động cơ. Thông qua tham số dir, khối Set Motor điều khiển động cơ theo hướng quay mong muốn.

Khối Transmit Data:

Đây là một khối quan trọng trong việc truyền dữ liệu từ động cơ lên máy tính thông qua frame truyền gồm 13 byte.

Chức năng chính của khối Transmit Data là gửi dữ liệu từ động cơ lên máy tính để được sử dụng cho mục đích vẽ đồ thị đáp ứng của động cơ theo thời gian. Dữ liệu này được lấy từ tín hiệu hồi tiếp từ Encoder, đại diện cho các thông số về vị trí, tốc độ.

Bằng cách truyền dữ liệu từ động cơ lên máy tính, khối ***Transmit Data*** cho phép người dùng theo dõi và giám sát quá trình hoạt động của động cơ. Đồng thời, dữ liệu này cũng cung cấp thông tin quan trọng để đánh giá chất lượng bộ điều khiển PID hoặc các thông số điều chỉnh khác liên quan đến động cơ.

(Lưu ý frame truyền lên PC khác frame truyền xuống STM32)

COMMAND (3)			DATA (10)									
0	1	2	3	4	5	6	7	8	9	10	11	12
Chọn chế độ điều khiển VEL or POS			Tín hiệu hồi tiếp từ Encoder		Setpoint		Kp		Ki		Kd	

Figure 16: Định dạng frame truyền lên PC

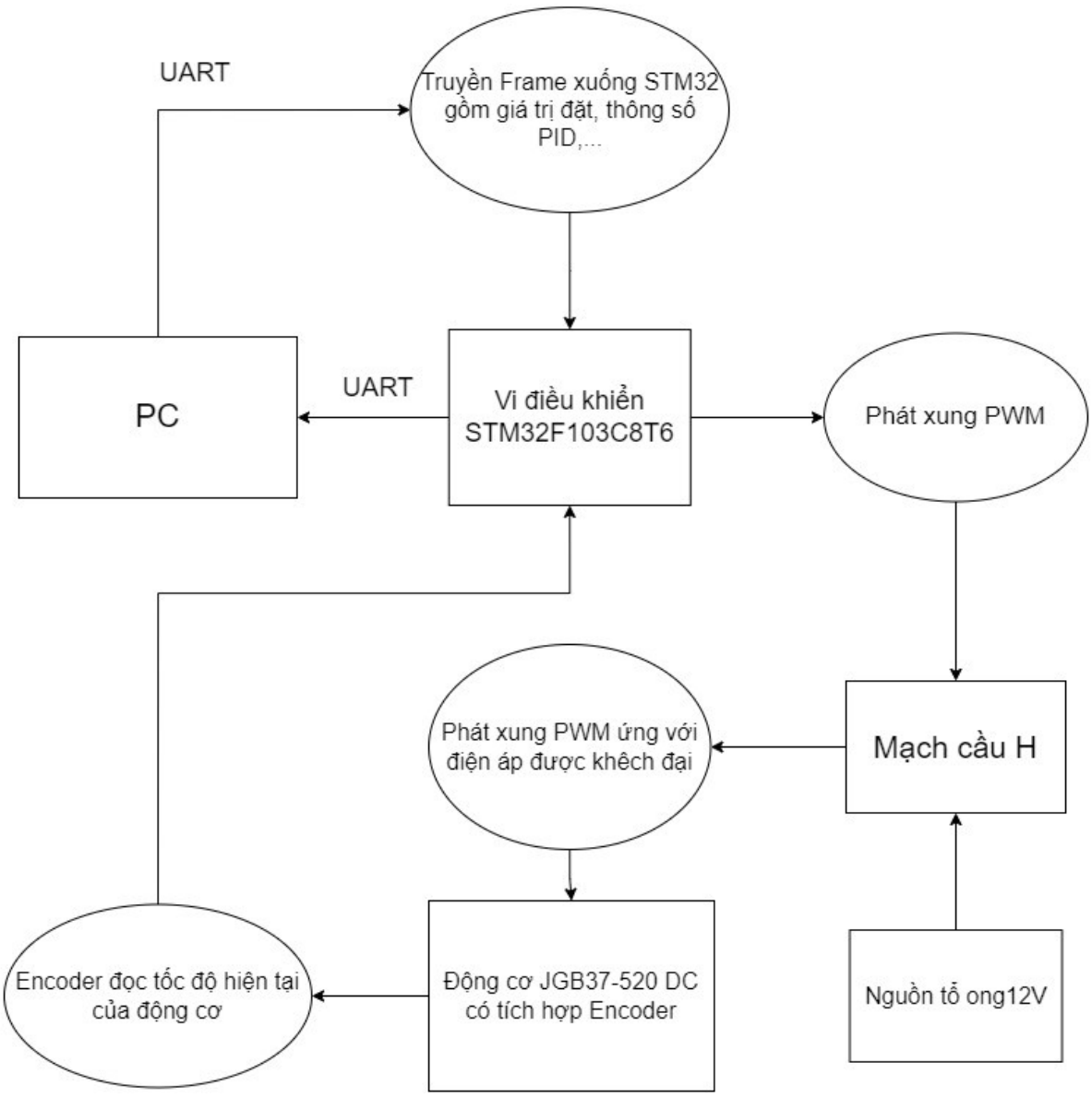


Figure 17: Sơ đồ giải thuật điều khiển hệ thống

5.2 FlowChart của hệ thống giao diện cho việc truyền, nhận

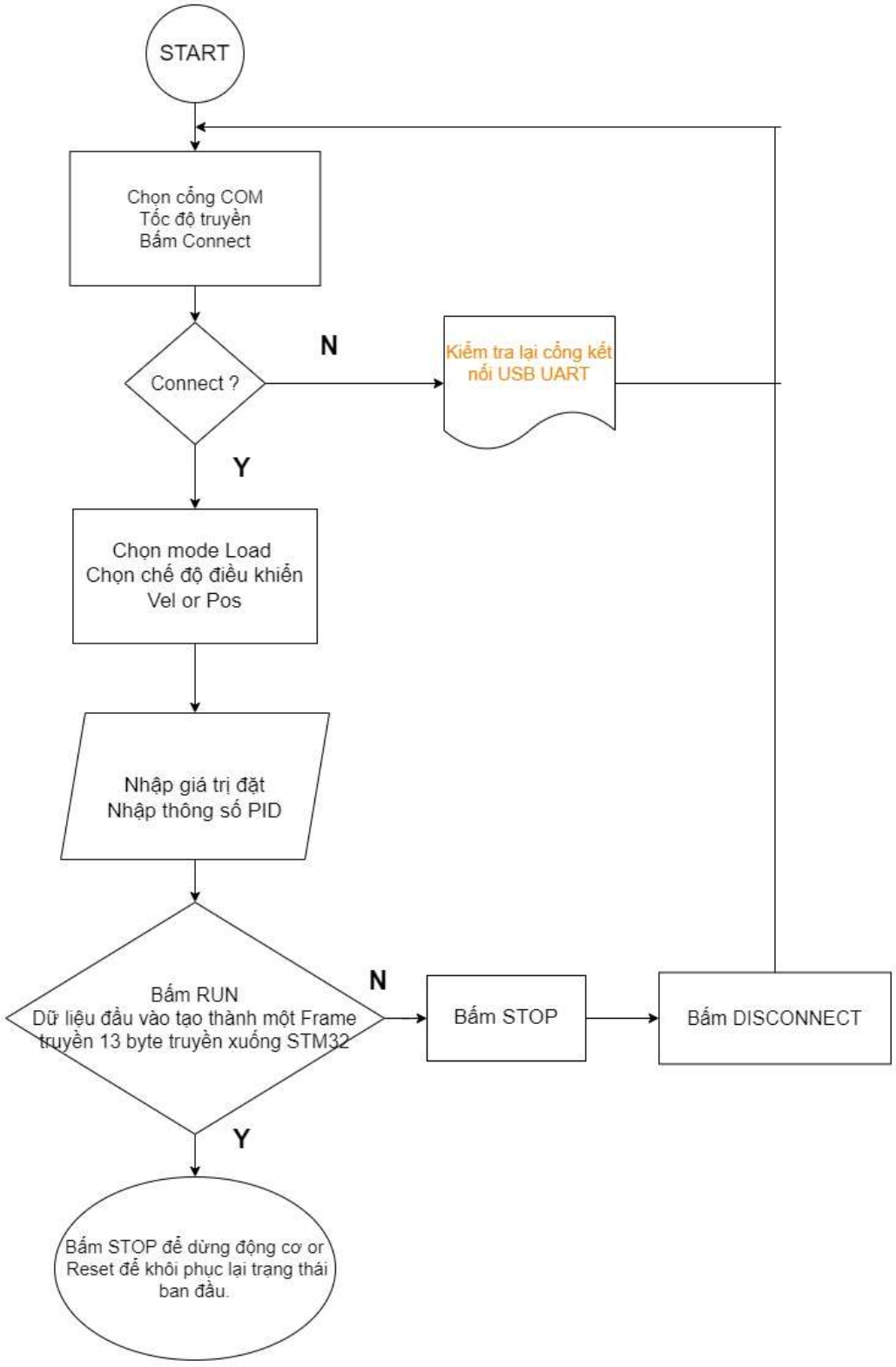


Figure 18: Sơ đồ giải thuật hệ thống giao diện

5.3 Thiết kế giao diện trên người dùng của Visual Studio C++

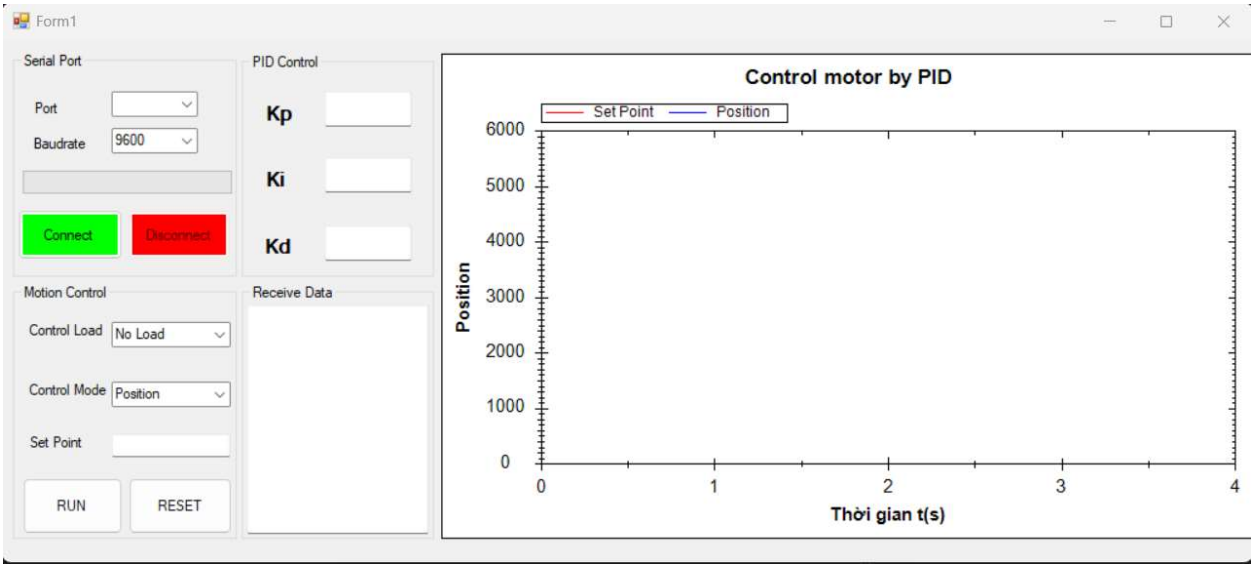


Figure 19: Giao diện người dùng

Giao diện tương tác với người dùng có các khối **Serial Port** dùng để Connect hoặc Disconnect với cổng Serial Port , khối **PID** Control để nhập bộ thông số điều khiển , khối **Motion Control** gồm việc chọn chế độ tải, chế độ điều khiển, khối **Receive Data** nhận dữ liệu từ động cơ sau khi thực hiện các chức năng điều khiển và cuối cùng là đồ thị đáp ứng hệ thống **Graph** để hiện thị dữ liệu nhận được từ động cơ dưới dạng biểu đồ đáp ứng hệ thống.

❖ **Chức năng các khối trong giao diện:**

Khối Serial Port: Cho phép người dùng kết nối cổng COM trên máy tính với Vi điều khiển thông qua giao diện USB UART. Người dùng có thể chọn cổng COM phù hợp và Baudrate, sau đó nhấn nút "CONNECT" để thực hiện kết nối.

Trong quá trình kết nối, thanh ProgressBar sẽ tải lên từ 0% đến 100% để hiển thị tiến trình kết nối. Ngược lại, nếu người dùng nhấn nút "DISCONNECT", việc ngắt kết nối sẽ được thực hiện và thanh ProgressBar sẽ trở về 0% để chỉ ra rằng kết nối đã được ngắt.

Khối PID Control: Khối này cho phép người dùng nhập thông số PID vào các ô Kp, Ki, Kd. Sau khi nhấn "RUN" dữ liệu trên 3 ô này sẽ được đưa vào frame truyền gửi xuống vi điều khiển. Vi điều khiển sẽ thực hiện xử lí và tính toán các thông số PID rồi gửi lại frame truyền lên máy tính và hiển thị trên khối **"Recceive Data".**

Khối Motion Control: Trong chế độ Control Load, chúng ta có hai tùy chọn quan trọng: "Không tải" và "Có tải". Lựa chọn chính xác giữa hai tùy chọn này rất quan trọng, vì nếu không chọn đúng, có thể gây hư hại cho động cơ.

Tiếp theo, chế độ Control Mode (chế độ điều khiển) cho phép chọn cách điều khiển động cơ dựa trên vận tốc hoặc vị trí, tùy thuộc vào nhu cầu của người sử dụng. Chúng ta có thể nhập giá trị mong muốn cho tốc độ hoặc vị trí vào ô "Setpoint" để động cơ đạt được giá trị đó.

Nhấn "**RUN**", máy tính sẽ gửi một frame truyền xuống vi điều khiển. Sau khi nhận được frame truyền ,vi điều khiển sẽ thực hiện xử lí chức năng tương ứng với frame truyền đó, bao gồm phát xung PWM để chạy động cơ. Sau đó, vi điều khiển sẽ gửi frame truyền đó lên máy tính chứa thông tin về tốc độ hiện tại của động cơ. (Lưu ý trước khi nhấn RUN phải nhập đầy đủ bộ thông số PID)

Nhấn "**STOP**", máy tính sẽ gửi xuống vi điều khiển một Frame truyền. Sau khi nhận được Frame truyền, vi điều khiển sẽ thực hiện Set Counter Encoder về 0, dừng phát xung PWM, và đưa toàn bộ biến trong chương trình trả về 0. Khi dừng động cơ dữ liệu trên Grap đồng thời cũng sẽ dừng truyền.

Nhấn "**RESET**", Đồ thị hiện thị dữ liệu Grap sẽ được reset về trạng thái ban đầu và đồng thời làm mới các biến chứa dữ liệu động cơ.

Khối Recceive Data: Khối này có nhiệm vụ hiển thị frame truyền dạng byte từ vi điều khiển gửi lên máy tính và hiển thị giá trị hiện tại của động cơ.

Khối Graph: Khối này gồm 2 trang cho 2 chế độ điều khiển động cơ. Việc lựa chọn chế độ điều khiển sẽ làm thay đổi trục đồ thị phù hợp với việc chạy động cơ để cho dễ nhìn thấy đáp ứng của hệ thống. Khi nhận được giá trị của động cơ gửi lên từ vi điều khiển khối này sẽ cập nhật liên tục đồ thị đáp ứng.

VI. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 Thực hiện các thử nghiệm và phân tích kết quả

Với mỗi Setpoint ta sẽ chạy thử nghiệm để tìm thông số PID phù với từng chế độ điều khiển để có thể làm đáp ứng tối ưu nhất về thời gian xác lập, độ vọt lố, sai số xác lập.

6.1.1 Đối với chế độ điều khiển tốc độ động cơ:

❖ ***Không tải:***

Ta sẽ chạy 5 trường hợp: cung một bộ thông số PID

Kp = 1, Ki = 0.4894, Kd = 0.1996

Trường hợp 1: Tốc độ đặt 165 rpm với

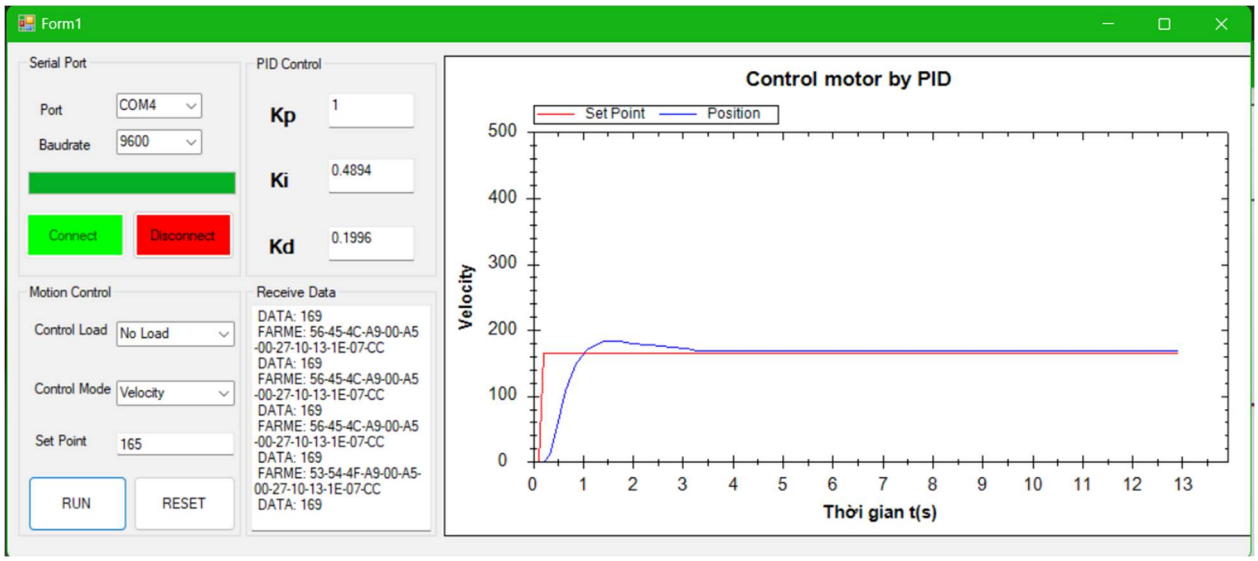


Figure 20: Đáp ứng đồ thị động cơ chạy với tốc độ 165 rpm (Không tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: $\frac{180-1}{165} \times 100 = 9\%$
- Sai số xác lập: 1
- Thời gian xác lập: 3s

Trường hợp 2: Tốc độ đặt 200 rpm

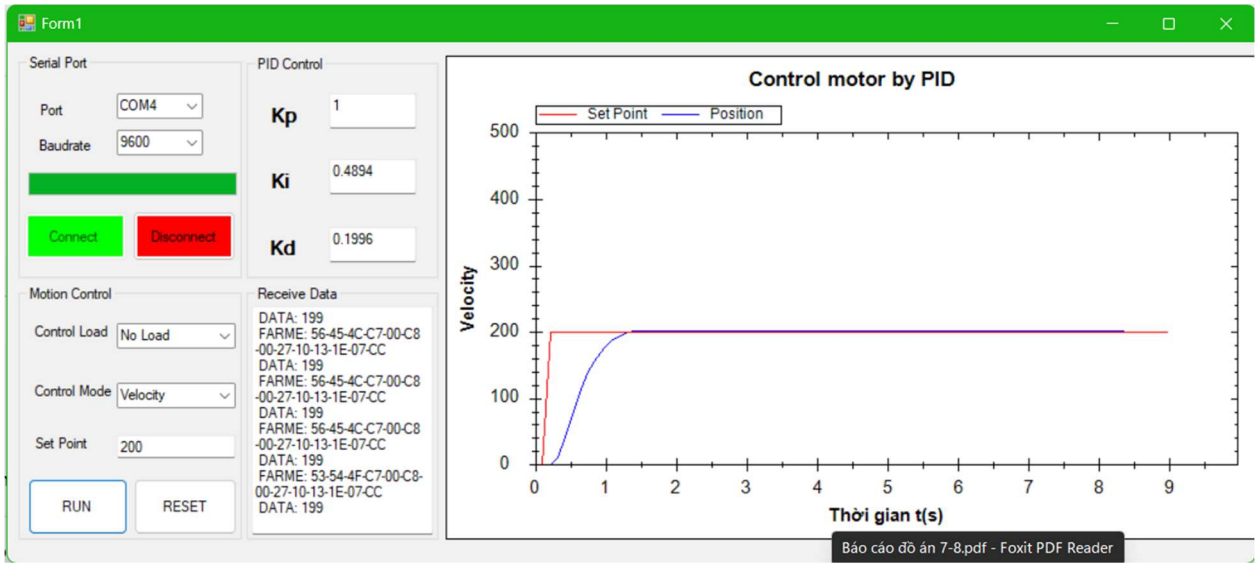


Figure 21: Đáp ứng đồ thị động cơ chạy với tốc độ 200 rpm (Không tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: 0%
- Sai số xác lập: 1
- Thời gian xác lập: 1.25 s

Trường hợp 3: Tốc độ đặt 250 rpm

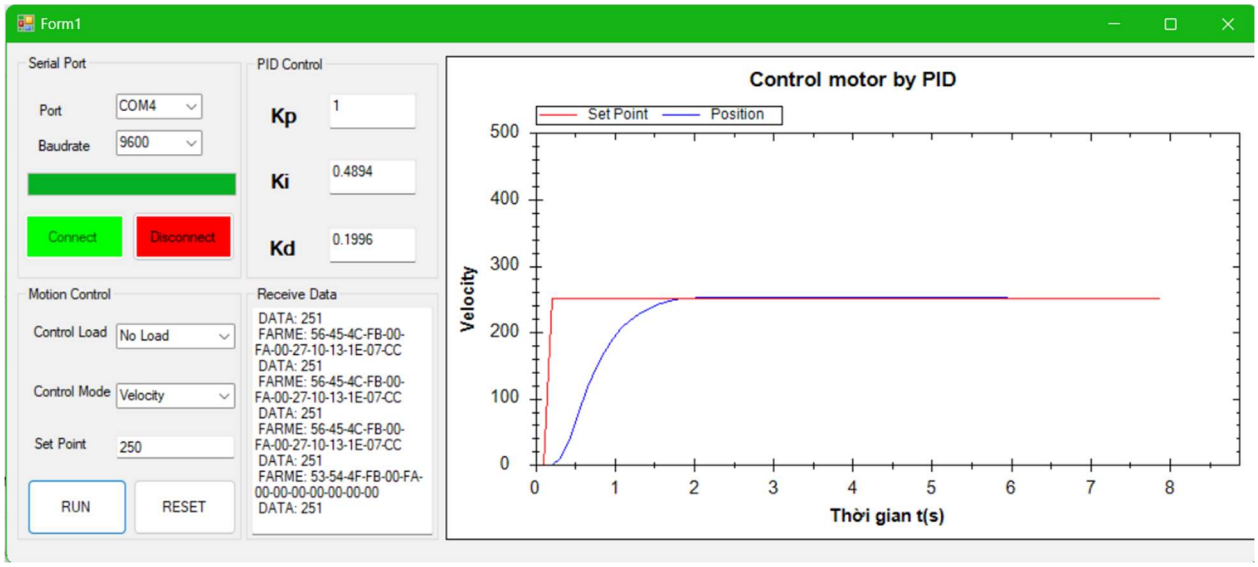


Figure 22: Đáp ứng đồ thị động cơ chạy với tốc độ 250 rpm (Không tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: 0%
- Sai số xác lập: 1
- Thời gian xác lập: 1.75 s

Trường hợp 4: Tốc độ đặt 280 rpm

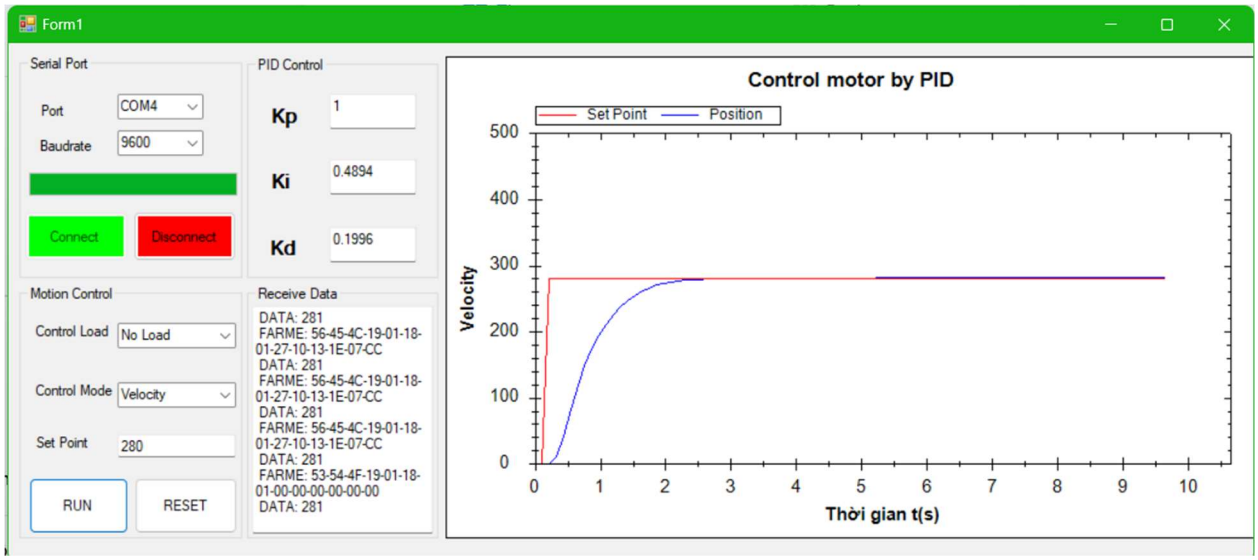


Figure 23: Đáp ứng đồ thị động cơ chạy với tốc độ 280 rpm (Không tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: 0%
- Sai số xác lập: 1
- Thời gian xác lập: 2.25 s

Trường hợp 5: Tốc độ đặt 300 rpm

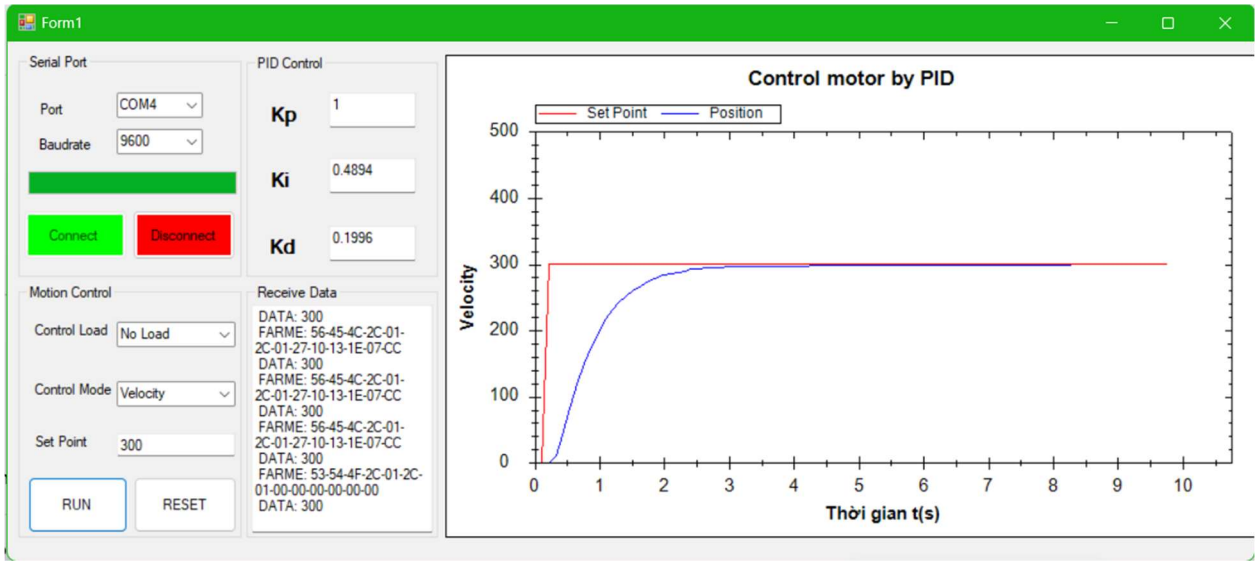


Figure 24: Đáp ứng đồ thị động cơ chạy với tốc độ 300 rpm (Không tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: 0%
- Sai số xác lập: 0
- Thời gian xác lập: 3 s

❖ ***Có tải***

Tốc độ đặt 250 rpm

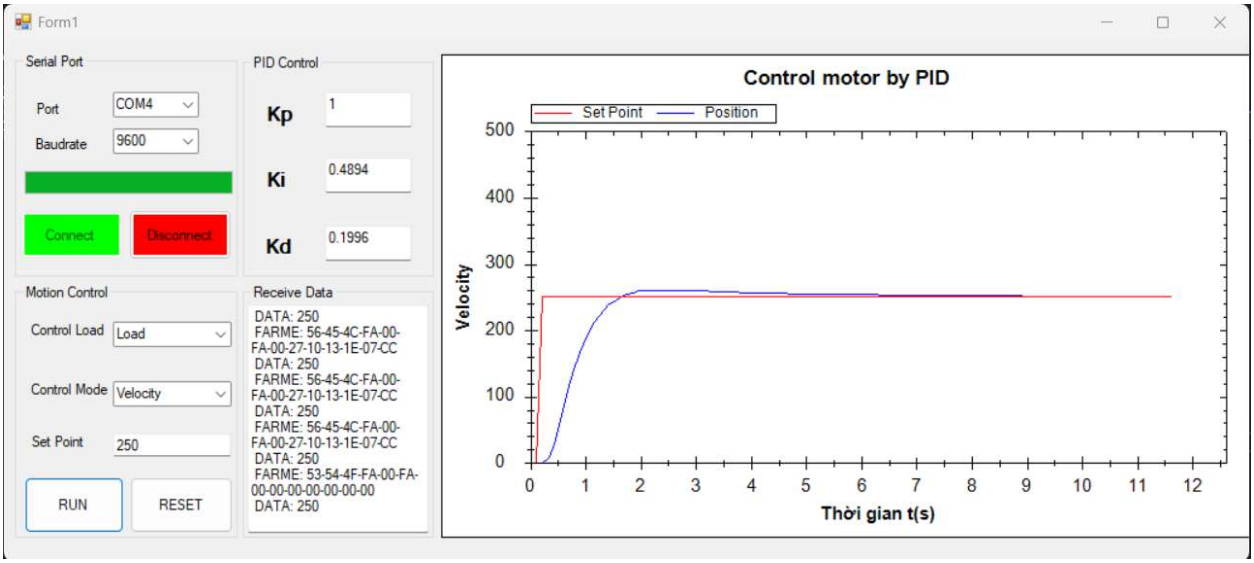


Figure 25: Đáp ứng đồ thị động cơ chạy với tốc độ 250 rpm (Tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: :$\frac{256-250}{250} \times 100 = 2.4\ \%$
- Sai số xác lập: 0
- Thời gian xác lập: 4 s

Kết luận:

Trường hợp 1, hệ thống có độ vọt lố nhỏ, sai số xác lập là 0 và thời gian xác lập nhanh. Trong trường hợp 2, hệ thống không có độ vọt lố, có một mức độ sai số xác lập nhỏ và thời gian xác lập tương đối nhanh. Nhận xét chung là cả hai trường hợp đều cho thấy hệ thống có khả năng đáp ứng tốt với các yêu cầu đặt ra.

Trường hợp có tải, hệ thống trong trường hợp này có độ vọt lố nhỏ, có mức độ sai số xác lập nhỏ và thời gian xác lập tương đối nhanh.

6.1.2 Đối với điều khiển vị trí động cơ

1320 xung của encoder = 1 vòng quay của động cơ (thông số động cơ)

❖ **Không tải**

Với bộ thông số PID : Kp = 0.0065, Ki = 0.0004, Kd= 0.0006

Trường hợp 1: Xét đến vị trí 2000 xung

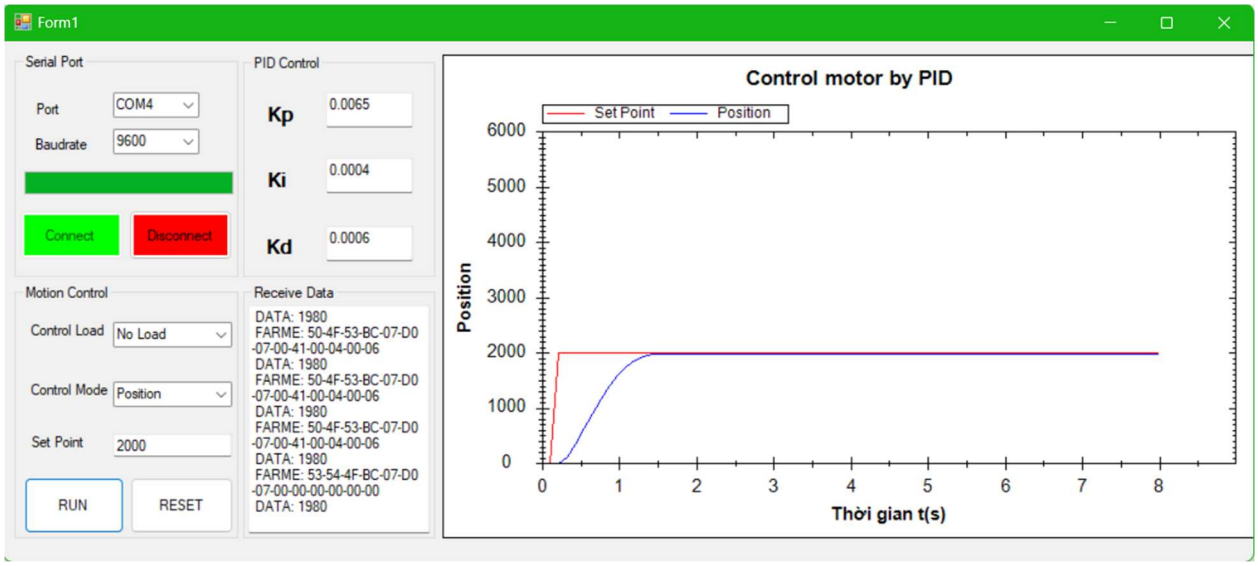


Figure 26: Đáp ứng đồ thị động cơ chạy đến vị trí 2000xung (Không tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: 0%
- Sai số xác lập: 20 xung
- Thời gian xác lập: 1.25s

Trường hợp 2: Xét vị trí 3000 xung

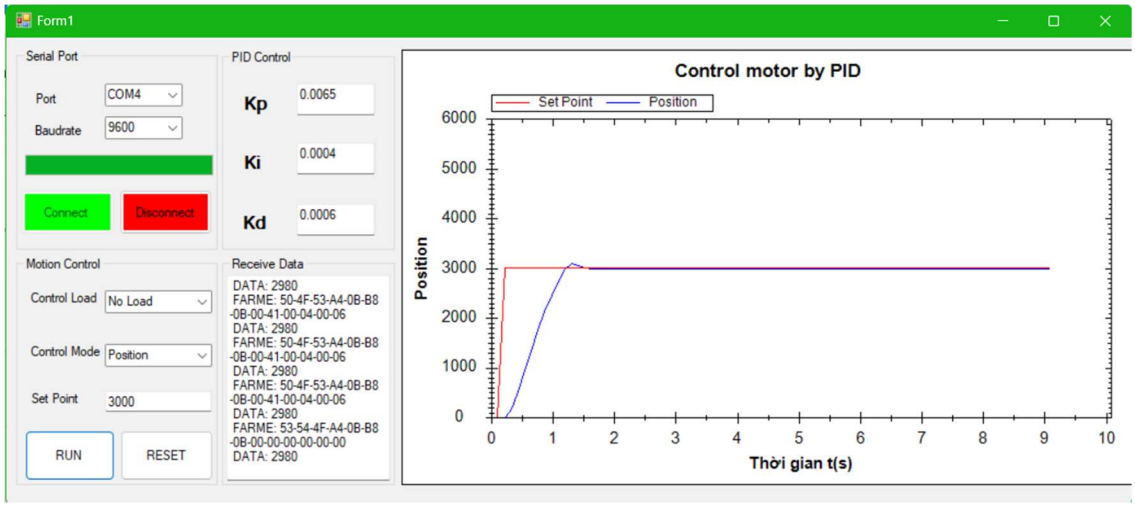


Figure 27: Đáp ứng đồ thị động cơ chạy đến vị trí 3000xung (Không tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: $\frac{3050-3000}{3000} \times 100 = 1.6\ \%$
- Sai số xác lập: 20 xung
- Thời gian xác lập: 1.5s

Trường hợp 3: Xét vị trí 5000 xung

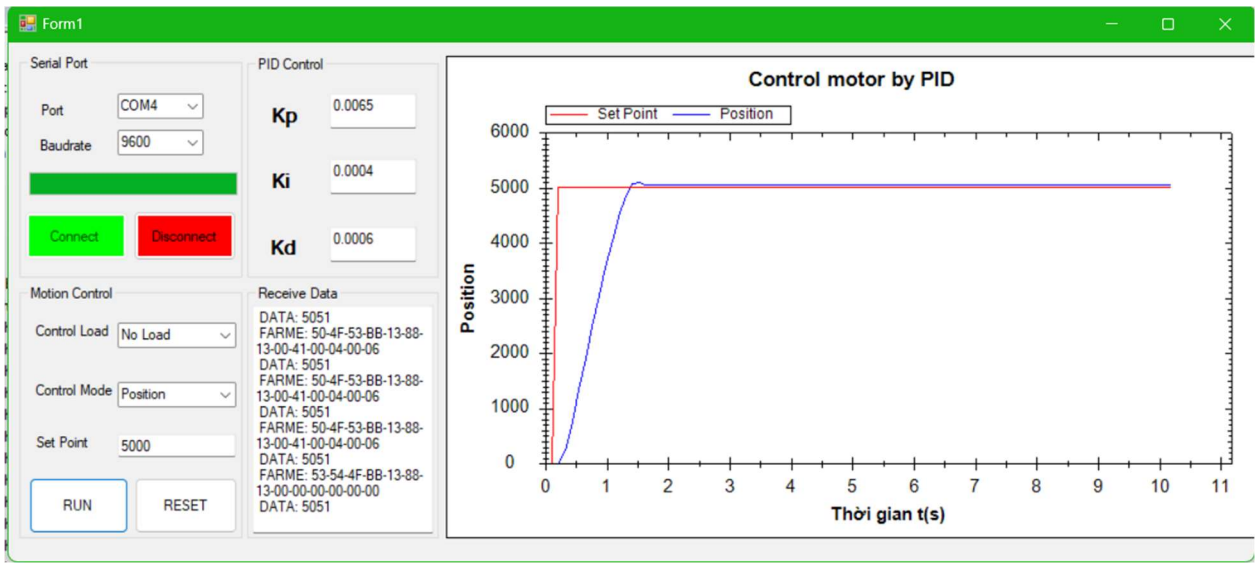


Figure 28: Đáp ứng đồ thị động cơ chạy đến vị trí 5000xung (Không tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: $\frac{3050-3000}{3000} \times 100 = 2\%$
- Sai số xác lập: 51 xung
- Thời gian xác lập: 1.5s

❖ **Có tải**

Đặt ví trí 3000 xung

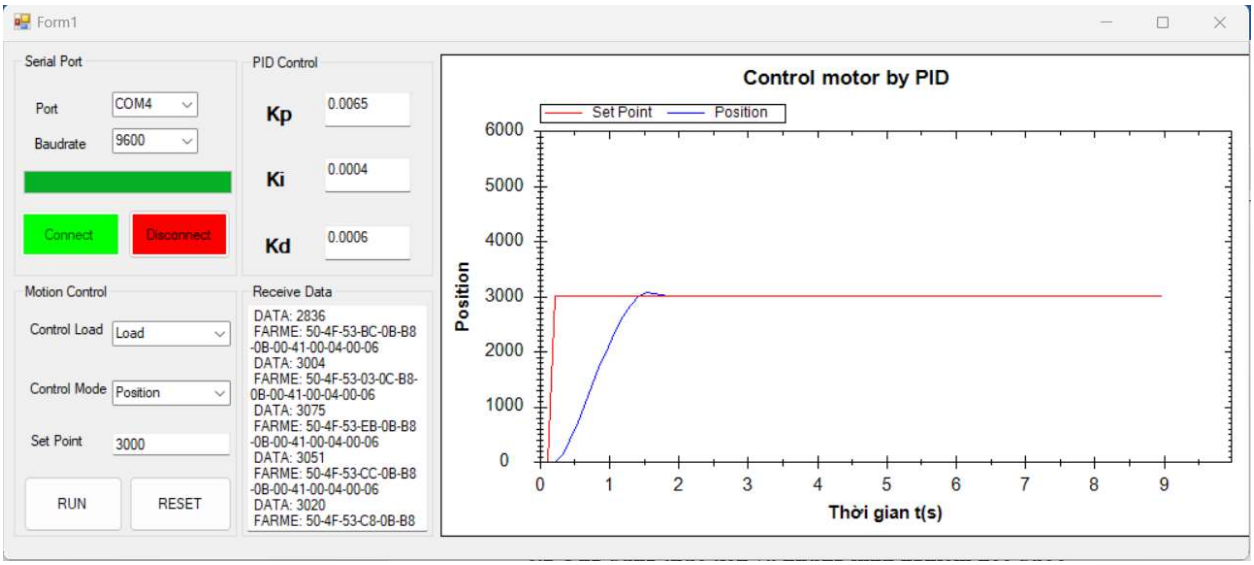


Figure 29: Đáp ứng đồ thị động cơ chạy đến vị trí 3000 xung (Tải)

Nhận xét đáp ứng hệ thống:

- Độ vọt lố: $\frac{3075-3000}{3000} \times 100 = 2.5\%$
- Sai số xác lập: 16 xung
- Thời gian xác lập: 1.75s

Kết luận: Nhìn chung trong tất cả các trường hợp, hệ thống đáp ứng tốt các yêu cầu đặt ra.

6.2 Ứng dụng thực tiễn và những kinh nghiệm học được

6.2.1 Ứng dụng thực tiễn

Một tiếp cận dễ thấy nhất được học trên trường là ứng dụng PID trong bộ điều lò nhiệt (nhiệt độ)

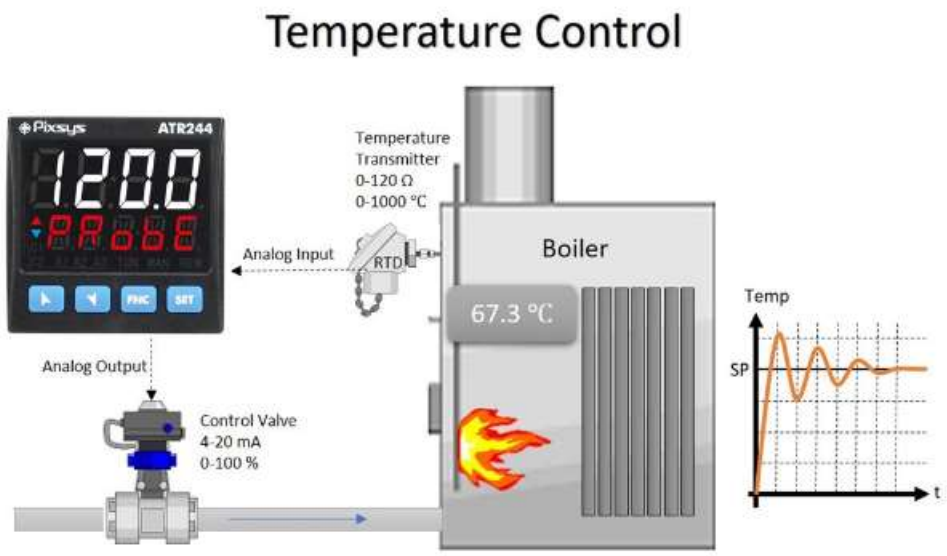


Figure 30: Ứng dụng PID trong điều khiển nhiệt độ

Trong bộ điều khiển lò nhiệt, PID có thể được sử dụng để điều chỉnh công tắc hoặc van để duy trì nhiệt độ mong muốn. PID sẽ so sánh giá trị đo nhiệt độ hiện tại với giá trị đặt trước và tính toán tín hiệu điều khiển dựa trên các thành phần P, I và D. Tín hiệu điều khiển này sẽ điều chỉnh nhiệt độ trong lò nhiệt.

Qua quá trình điều khiển PID, hệ thống sẽ tiếp tục kiểm tra và điều chỉnh tín hiệu điều khiển để đảm bảo rằng nhiệt độ được duy trì ở mức mong muốn một cách chính xác và ổn định.

Chính vì thế hiện nay có rất nhiều ứng dụng PID trong điều khiển nhiệt độ khác như sử dụng để điều chỉnh nhiệt độ trong các hệ thống sưởi, máy làm lạnh, hệ thống điều hòa không khí và các quy trình gia nhiệt khác.

Figure 31: Ứng dụng PID trong hệ thống điều hòa không khí

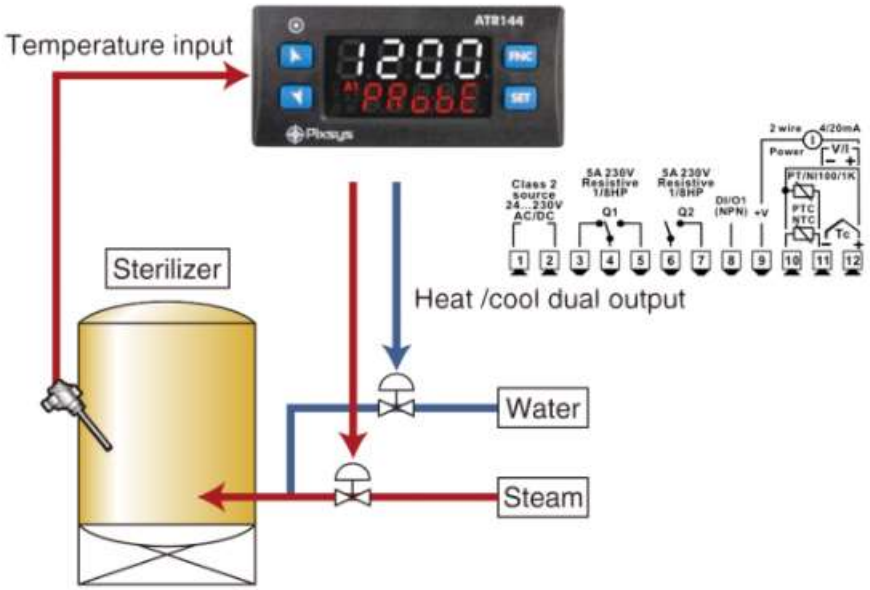


Figure 32: Ứng dụng PID trong hệ thống gia nhiệt

Ngoài ra còn có các ứng dụng khác như: Ứng dụng bộ điều khiển PID trong điều khiển mức nước, kiểm soát với lưu lượng nước qua đường ống, biến tần, điều khiển tự động hóa, kết nối với PLC nhằm giúp người dùng điều khiển nhiệt độ, lưu lượng và áp suất của các lưu chất, dung dịch… trong hệ thống sản xuất.

6.2.2 Kinh nghiệm học được:

Đọc thống số, hiểu rõ về nguyên lí hoạt động động cơ

Hiểu rõ về nguyên lí hoạt động của PID và các thành phần cơ bản như hệ số tỉ lệ (Kp,) hệ số tích phân (Ki), hệ số vi phân (Kd). Điều này giúp hiểu rõ về ảnh hưởng của mỗi hệ số đến quá trình điều khiển và tối ưu hiệu suất, đồng thời có thêm kinh nghiệm trong các vấn đề phát sinh lỗi.

Tinh chỉnh PID thông qua phương pháp thử và sai: Bằng cách điều chỉnh giá trị của các hệ số này và quan sát phản hồi trong quá trình thực hiện điều khiển, tôi có thể tìm ra các giá trị tối ưu cho hệ số PID để đạt được hiệu suất tốt nhất.

Biết được cách xử lí nhiễu bằng bộ lọc thông thấp trong quá trình hệ thống hoạt động. Để loại bỏ nhiễu tần số cao không mong muốn cho phép các tần số thấp hơn.

Điều quan trong nhất là phải đảm bảo tính ổn định và đáng t in cậy của hệ thống điều khiển. Đặc biệt, các thiết bị đo lường như Encoder có thể gây ra nhiễu dẫn đến hệ thống hoạt động sai sót.

VII. TÀI LIỆU THAM KHẢO

[1] Bộ điều khiển PID, Wikipedia,

https://vi.wikipedia.org/wiki/B%E1%BB%99_%C4%91i%E1%BB%81u_khi%E1%BB%83n_PID

[2] Mô hình hệ thống điều khiển

https://voer.edu.vn/c/mo-hinh-he-thong-dieu-khien-tu-dong/c949c256/ff8e453a

[3] Data sheet stm32f103c8t6

https://www.alldatasheet.com/view.jsp?Searchword=Stm32f103c8t6&gclid=Cj0KCQjwib2mBhDWARIsAPZUn_mlxKVAa7oxmB21UZmTLuBgtKG0mdYxF8KqijTiZYG7y0RM8mn8xGAaAgWKEALw_wcB

[4] Nguyên lý hoạt động cầu H L298

https://kysungheo.com/nguyen-ly-mach-cau-h-l298/

[5] Ứng dụng bộ điều khiển PID trong công nghiệp

https://cambiendoapsuat.vn/pid-la-gi/

[6] Điều khiển động cơ bằng bộ mã hóa, Curio Res,

https://www.youtube.com/watch?v=HRaZLCBFVDE&t=479s