

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA THÀNH PHỐ HỒ CHÍ MINH



BÁO CÁO BÀI TẬP LỚN
VI ĐIỀU KHIỂN VÀ HỆ THỐNG NHÚNG

Đề tài

ĐIỀU KHIỂN VỊ TRÍ ĐỘNG CƠ BẰNG BỘ
ĐIỀU KHIỂN PID, STR VÀ LQR

GVHD: Nguyễn Vĩnh Hảo

Nhóm 5: Nguyễn Quốc Khánh Lân 2010371

Nguyễn Tiến Đạt 2011073

TP. HỒ CHÍ MINH, 14 THÁNG 12 NĂM 2024

MUC LUC

1. Phần cứng	3
2. Chương trình	8
2.1. Chương trình Matlab	8
2.2. Firmware	15
2.2.1. Đọc Encoder và xuất xung điều khiển	16
2.2.2. Xử lý dữ liệu nhận qua UART từ GUI	17
2.3. Software	18
2.3.1 Giao tiếp giữa GUI và vi điều khiển	18
2.3.2. Giao diện	22
3. Kết quả thực hiện:	25

1. Phần cứng

Mô hình phần cứng bao gồm motor DC có encoder, mạch điều khiển động cơ, vi điều khiển STM32F407 Discovery, đồng thời giao tiếp với PC thông qua giao tiếp UART.

1.1. Động cơ DC

Nhóm sử dụng motor GA37 – 520 các các thông số như sau:

Thông số	Giá trị
Điện áp sử dụng	12VDC
Đường kính động cơ	37mm
Đường kính trục động cơ	6mm (Tương thích với khớp nối trục 6mm)
Encoder	Cảm biến từ trường Hall, 2 kênh AB, 11 xung/1 kênh/1 vòng
Điện áp cấp cho Encoder	3.3~5VDC
Tỉ số truyền	270:1 (Động cơ quay 270 vòng, trục chính hộp giảm tốc quay 1 vòng)
Dòng không tải	120mA
Dòng chịu đựng tối đa khi có tải	1A
Tốc độ không tải	37RPM (37 vòng/phút)
Tốc độ chịu đựng tối đa khi có tải	28RPM (28 vòng/phút)
Lực kéo Moment định mức	30KG.CM
Lực kéo Moment tối đa	35KG.CM
Chiều dài hộp số L	26.5mm
Số xung Encoder mỗi kênh trên 1 vòng quay trục chính	2970 xung



Hình 1.1 Motor DC GA37 – 520

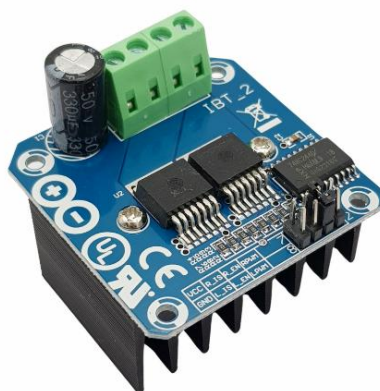


Hình 1.2 Sơ đồ chân của động cơ

1.2. Mạch điều khiển động cơ:

Nhóm sử dụng mạch điều khiển động cơ DC BTS7960. Thông số kỹ thuật của mạch cụ thể như sau:

Thông số	Giá trị
Nguồn cấp	6 ~ 27VDC
Dòng điện tải mạch	43A (Tải trở) hoặc 15A (Tải cảm)
Tín hiệu logic điều khiển	3.3 ~ 5VDC
Tần số điều khiển tối đa	25KHz
Tự động shutdown khi điện áp thấp	Tự shutdown khi điện áp < 5.5VDC, mở lại khi > 5.5VDC
Bảo vệ quá nhiệt	BTS7960 có cảm biến nhiệt tích hợp, ngắt khi quá nhiệt
Kích thước	40 x 50 x 12mm



Hình 1.3 Mạch điều khiển động cơ BTS7960

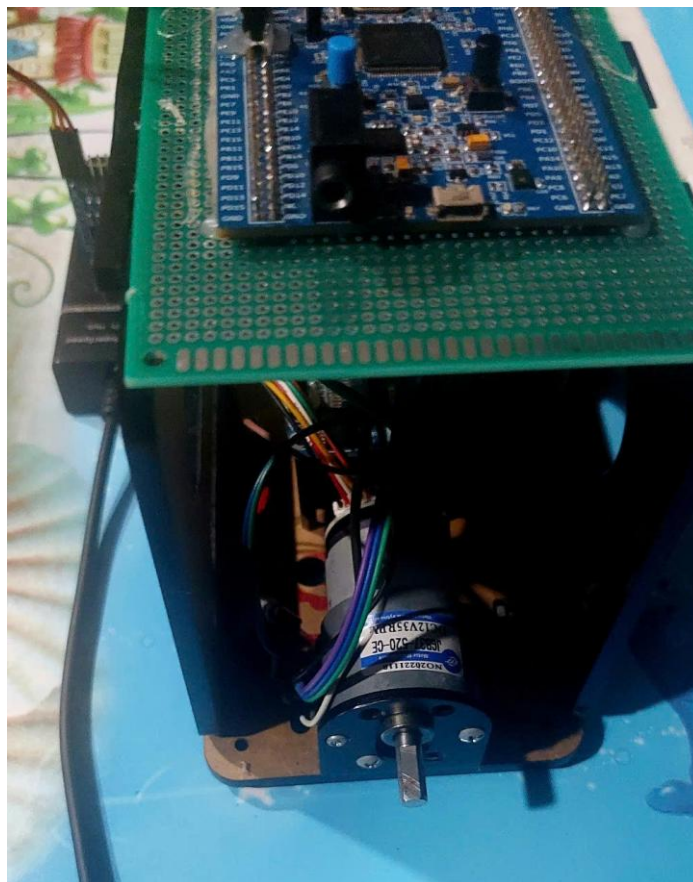
Chân	Mô tả
VCC	Nguồn tạo mức logic điều khiển (3.3~5VDC)
GND	Chân đất
R_EN	R_EN = 0: Disable nửa cầu H phải. R_EN = 1: Enable nửa cầu H phải
L_EN	L_EN = 0: Disable nửa cầu H trái. L_EN = 1: Enable nửa cầu H trái
RPWM và LPWM	Chân điều khiển đảo chiều và tốc độ động cơ

RPWM = 1 và LPWM = 0	Mô tơ quay thuận
RPWM = 0 và LPWM = 1	Mô tơ quay nghịch
RPWM = 1 và LPWM = 1 hoặc RPWM = 0 và LPWM = 0	Dừng
R_IS và L_IS	Kết hợp với điện trở để giới hạn dòng qua cầu H

1.3. Sơ đồ kết nối

Cấp nguồn 12VDC cho mạch cầu, ngõ ra Output kết nối với động cơ DC. Sử dụng 2 chân Enable để cấp xung điều khiển cho động cơ, 2 chân RPWM và LPWM để xác định chiều quay động cơ.

Tên chân	Chức năng	Ghi chú
GPIOB_PIN_13	Forward Direction	FORWARD_Pin - Điều khiển hướng tiến của động cơ. Nối vào RPWM của cầu H.
GPIOB_PIN_14	Backward Direction	BACKWARD_Pin - Điều khiển hướng lùi của động cơ. Nối vào LPWM của cầu H.
UART4_TX (GPIOA_PIN_0)	UART Transmission	UART giao tiếp ở chế độ truyền dữ liệu. Nối vào chân RX mạch giao tiếp UART to USB.
UART4_RX (GPIOA_PIN_1)	UART Reception	UART giao tiếp ở chế độ nhận dữ liệu. Nối vào chân TX mạch giao tiếp UART to USB.
TIM1 CH1 và TIM1 CH2 (GPIOE_PIN_9 và GPIOE_PIN_11)	Encoder Interface	Đếm xung encoder thông qua chế độ Encoder 2 kênh. Nối vào 2 kênh A và B của Encoder
TIM3 CH1 (GPIOA_PIN_6)	PWM Output	Điều khiển tín hiệu PWM cho động cơ. Nối chung với 2 chân Enable.



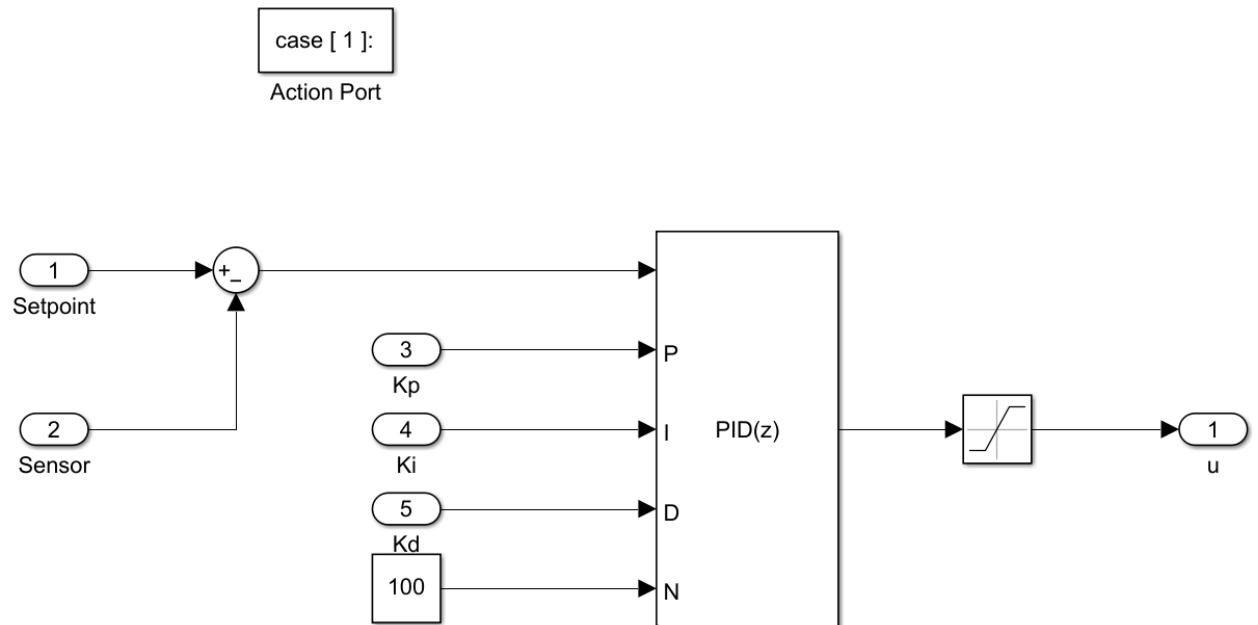
Hình 1.2 Mô hình thực tế

2. Chương trình

2.1. Chương trình Matlab

2.1.1. Điều khiển PID

Sử dụng bộ điều khiển PID có sẵn của Matlab, ta có sơ đồ simulink của bộ điều khiển như sau:



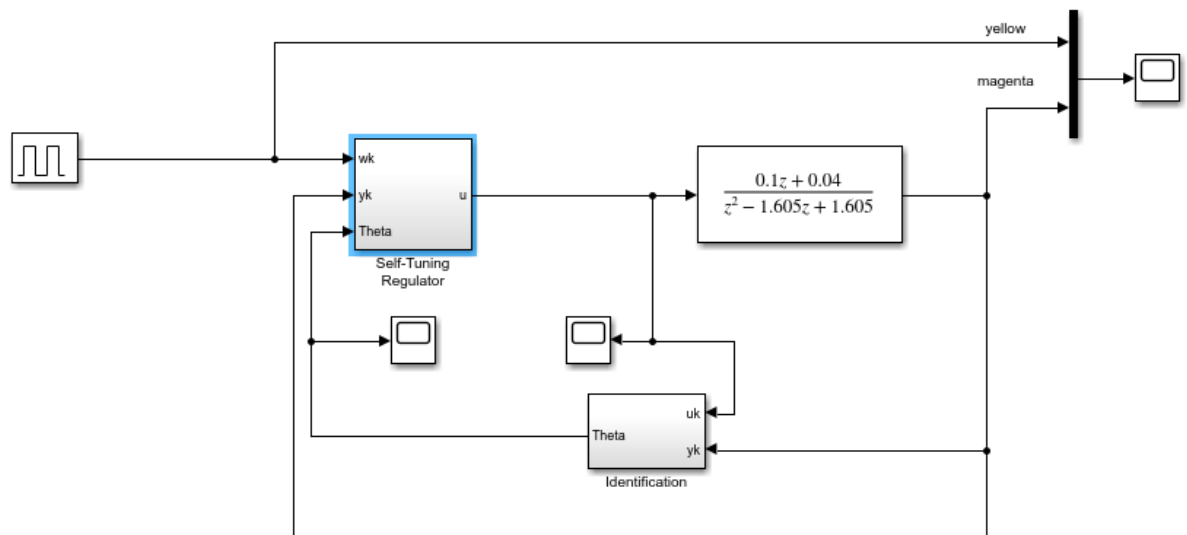
Hình 2.1 Sơ đồ bộ điều khiển PID

2.1.2. Điều khiển STR

a. Mô phỏng Simulink

Kiểm tra hoạt động bộ điều khiển STR thông qua mô hình motor DC cho trước

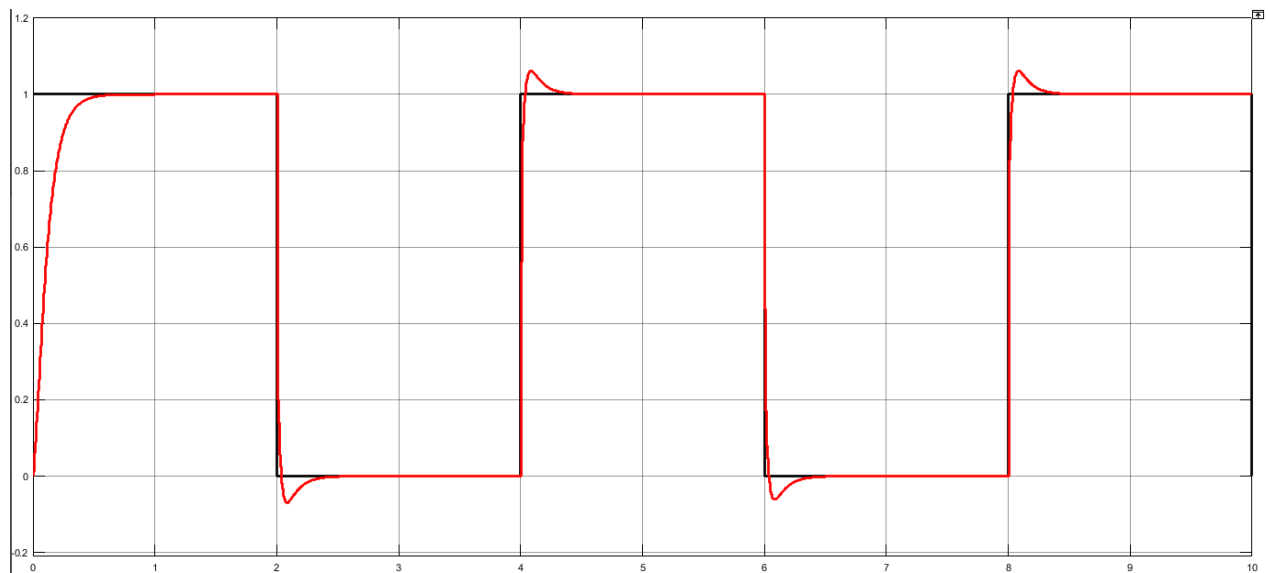
$$\frac{0.1z + 0.04}{z^2 - 1.605z + 0.605}$$



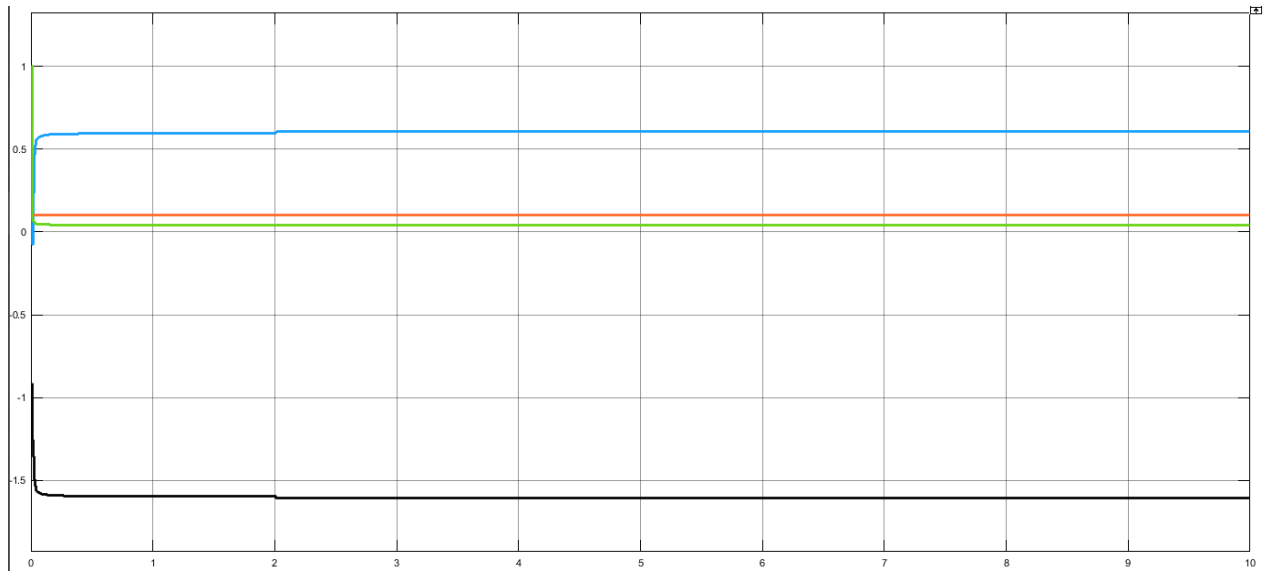
Hình 2.2 Sơ đồ bộ điều khiển STR mô phỏng

Kết quả mô phỏng:

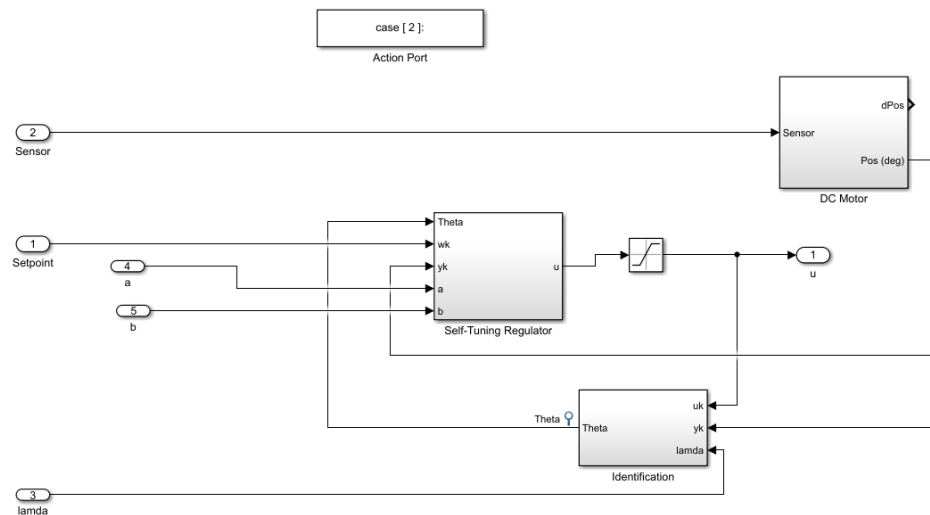
Tiến hành mô phỏng, thu được kết quả mô phỏng của tín hiệu ra so với tín hiệu đặt như hình bên dưới



Tín hiệu của giá trị theta, được nhận dạng thông qua khối Identification sẽ hội tụ về 1 giá trị, chứng tỏ mô hình đã được nhận dạng.

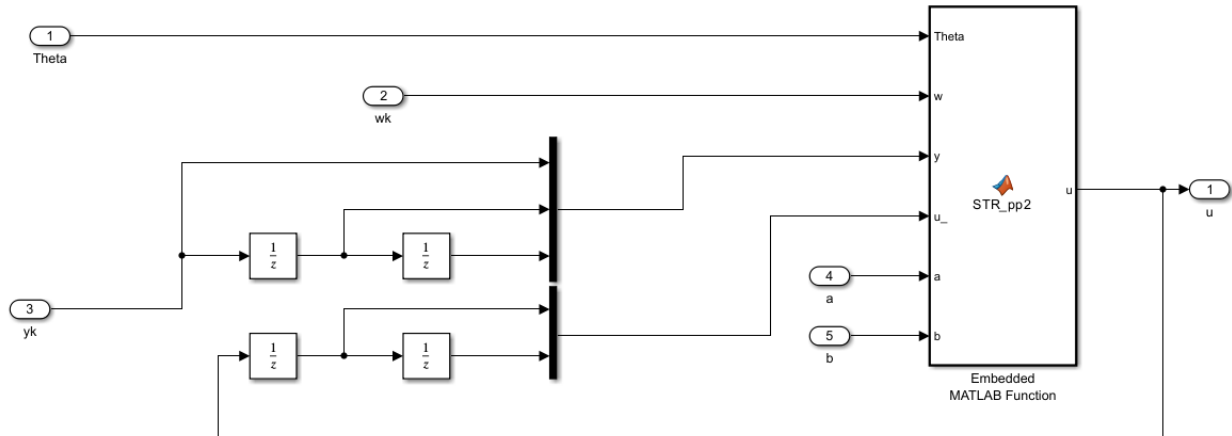


Mô hình gồm 2 khối chính là khối nhận dạng và khối tự chỉnh. Khối nhận dạng dùng để nhận dạng thông số mô hình thông qua các tín hiệu đầu vào và đầu ra. Khối điều khiển tự chỉnh dùng để tính toán và xuất tín hiệu điều khiển u .



Hình 2.3 Sơ đồ bộ điều khiển STR thực tế

Khối điều khiển tự chỉnh có sơ đồ như sau:



Hình 2.4 Khối Self Tuning Regulator

Với khối Embedded MATLAB Function dùng để viết code Matlab tính toán tín

`function u = STR_pp2(Theta,w,y,u_,a,b)`

```
% Cài đặt các tham số mong muốn
% beta = 0.1;
% alfa = 0.6;
beta = b;
alfa = a;
```

```
% Gán lại cho dễ sử dụng
a1 = Theta(1);
a2 = Theta(2);
b1 = Theta(3);
b2 = Theta(4);
```

```
% Sinh viên tham khảo hướng dẫn Mục 2.4.2 để viết bổ sung code
```

```
% Calculate intermediate variables x1, x2, x3, and x4
x1 = -4 * alfa + 1 - a1;
x2 = 6 * alfa^2 + beta^2 + a1 - a2;
x3 = 2 * alfa * (2 * alfa + beta^2) - a2;
```

```

x4 = alfa^2 * (alfa^2 + beta^2);

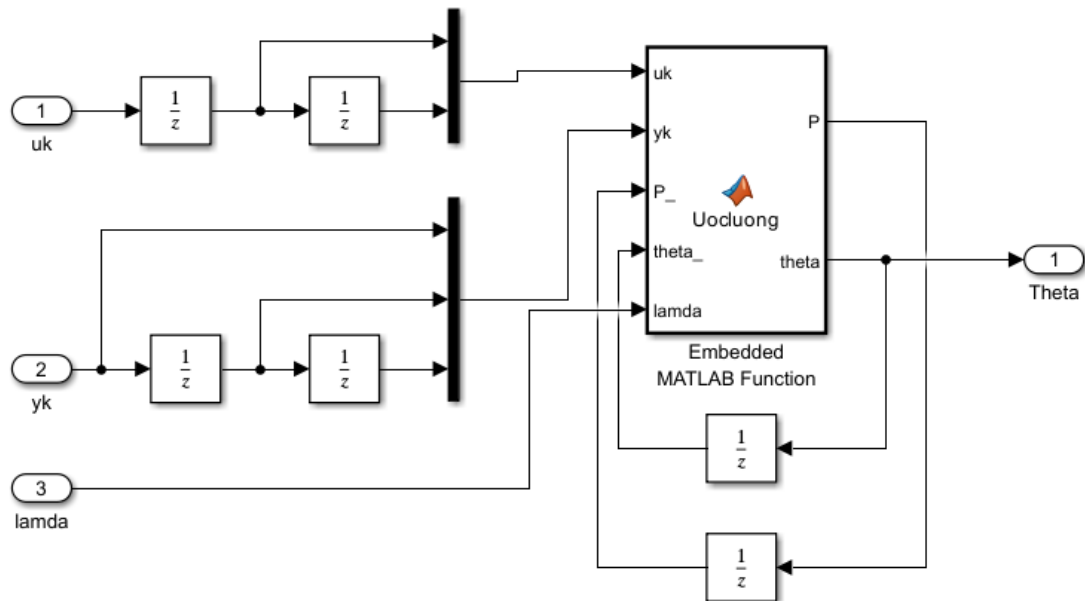
% Calculate r1 through r7
r1 = (b1 + b2) * (a1 * b2 - a2 * b1^2 - b2^2);
r2 = a1 * b2 * (b1 * (x2 - x3 + x4) - b2 * x1);
r3 = a2 * b1 * (b2 * x1 - b1 * (x2 - x3 + x4));
r4 = (b1 + b2) * (b1 * x4 + b2 * (x3 - x4));
r5 = b1 * (b1^2 * x4 + b1 * b2 * x3 + b2^2 * x2) - b2^3 * x1;
r6 = b2 * (b2 * (a2 * x3 + a1 * x4 - a2 * x4));
r7 = b2 * (b1 * (a1 * x4 + a2 * x2 - x4) - b2 * (a2 * x1 + x4));

% Calculate r0, q0, q1, q2, p1, and p2
r0 = (x1 + x2 - x3 + x4) / (b1 + b2);
q0 = r0 - (r2 + r3 + r4) / r1;
q1 = (r2 + r3 + r4 - r6 - r7) / r1;
q2 = (r6 + r7) / r1;
p1 = (r5 - r1) / r1;
p2 = -r5 / r1;

% Luật điều khiển
u = r0*w - q0*y(1) - q1*y(2) - q2*y(3) - p1*u_(1) - p2*u_(2);

```

Khối nhận dạng hệ thống có sơ đồ như sau:



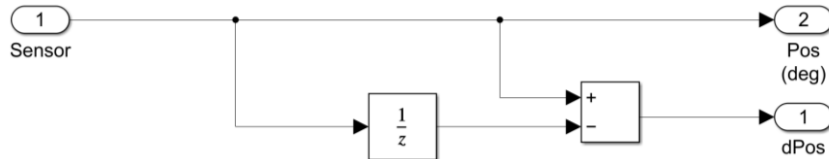
Hình 2.5 Khối nhận dạng

```

function [P,theta] = Uocluong(uk,yk,P_,theta_,lamda)
%lamda = 0.99;
PHI = [-yk(2); -yk(3); uk(1); uk(2)];
e = yk(1) - PHI'*theta_;
L = P_*PHI / (lamda + PHI'*P_*PHI);
P = 1/lamda*(P_ - P_*PHI*PHI'*P_/(lamda + PHI'*P_*PHI));
theta = theta_ + L*e;

```

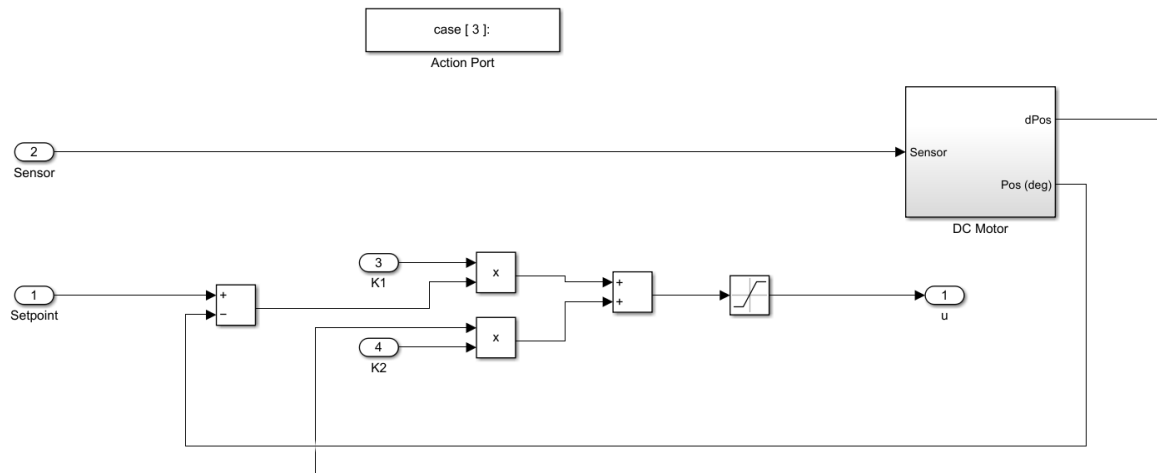
Khối motor DC



Hình 2.6 Khối DC Motor

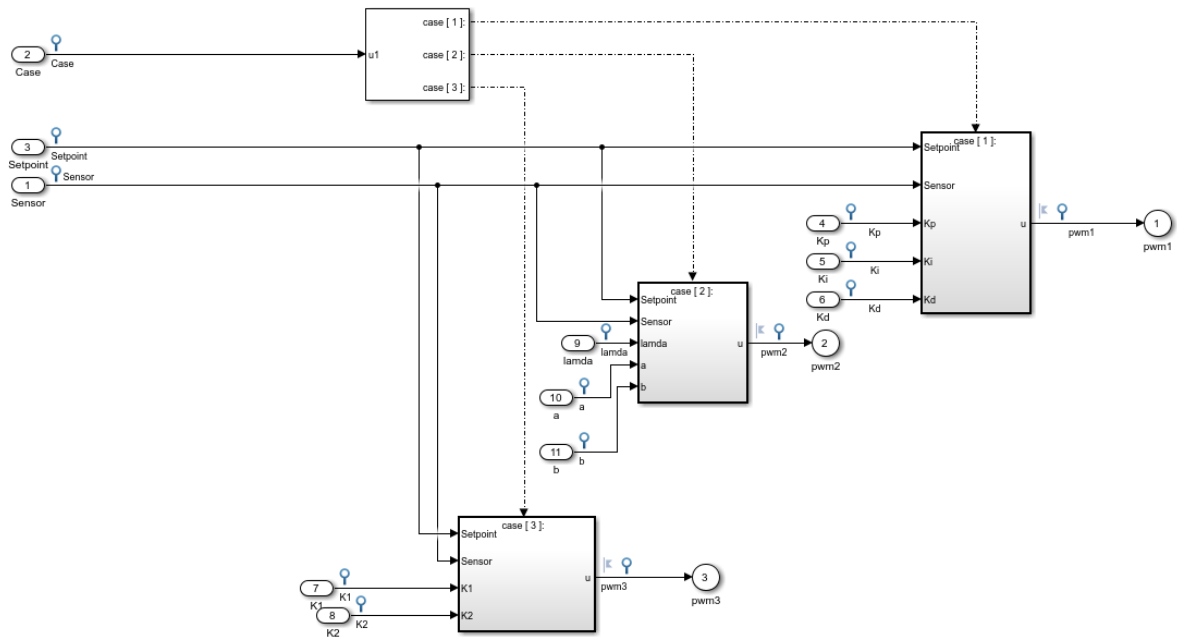
2.1.3. Điều khiển LQR

Trong bộ điều khiển LQR, cần tính 2 giá trị K1 và K2 là hệ số ứng với sai số giữa ngõ ra và tín hiệu đặt và tốc độ của động cơ.



Hình 2.7 Sơ đồ bộ điều khiển LQR
















2.1.4. Kết hợp 3 bộ điều khiển



Hình 2.8 Sơ đồ Simulink

Trong mô hình simulink dùng để build code C, các giá trị Case, Setpoint, sensor, K1, K2, Kp, Ki, Kd, a, b, lamda sẽ là ngõ vào lấy được từ chương trình Firmware qua tính toán, nhập thông số từ PC và đọc encoder của motor.

Giá trị Case sẽ chọn bộ điều khiển, giá trị đặt setpoint, sensor là giá trị encoder đọc được. Từ các giá trị trên sẽ tính được giá trị pwm thông qua đó lập trình để xuất xung điều khiển motor.

	Name	BlockType
	Sensor	Inport
	Case	Inport
	Setpoint	Inport
	Kp	Inport
	Ki	Inport
	Kd	Inport
	K1	Inport
	K2	Inport
	lamda	Inport
	a	Inport
	b	Inport
	Switch Case	SwitchCase
	pwm1	Outport
	pwm2	Outport
	pwm3	Outport

Hình 2.9 Các biến Inport và Outport

2.2. Firmware

Luồng logic:

1. Trường hợp 1: Điều khiển PID

- Gán điểm đặt từ dữ liệu nhận được.
- Lấy phản hồi từ cảm biến là giá trị hiện tại của vị trí.
- Cài đặt các hệ số PID từ dữ liệu nhận được.
- Thực hiện thuật toán PID.
- Tính toán giá trị điều khiển và truyền vào hàm để điều khiển tốc độ/hướng của động cơ.
- Xử lý dữ liệu phản hồi thành chuỗi và truyền qua UART.

2. Trường hợp 2: Điều khiển STR

- Tương tự như Trường hợp 1 nhưng không cài đặt hệ số.
- Sử dụng thuật toán điều khiển khác để tính toán giá trị điều khiển.
- Áp dụng giá trị điều khiển cho động cơ.
- Truyền dữ liệu phản hồi qua UART.

3. Trường hợp 3: Điều khiển LQR

- Lấy các tham số đặc trưng của LQR từ dữ liệu nhận được.
- Thực hiện thuật toán điều khiển để tính toán giá trị điều khiển.
- Áp dụng giá trị điều khiển cho động cơ.

- Truyền dữ liệu phản hồi qua UART giống như các trường hợp khác.

2.2.1. Đọc Encoder và xuất xung điều khiển

Điều khiển động cơ:

1. Kiểm tra giá trị đầu vào:

◦ Nếu giá trị không âm:

- Đặt giá trị làm độ lớn tín hiệu điều khiển.
- Kích hoạt tín hiệu điều khiển hướng tiến.
- Vô hiệu hóa tín hiệu điều khiển hướng lùi.

◦ Nếu giá trị âm:

- Lấy giá trị tuyệt đối và đặt làm độ lớn tín hiệu điều khiển.
- Kích hoạt tín hiệu điều khiển hướng lùi.
- Vô hiệu hóa tín hiệu điều khiển hướng tiến.

2. Điều chỉnh chân điều khiển:

- Bật/tắt các chân tín hiệu tương ứng với hướng quay của động cơ.

Đọc vị trí động cơ:

1. Đọc tín hiệu:

- Lấy giá trị hiện tại từ bộ đếm xung.
- Thực hiện ép kiểu để xác định chiều qua (Nếu giá trị là âm thì quay chiều ngược, nếu dương là thuận).

2. Tính toán vị trí:

- Chuyển đổi số xung nhận được thành góc quay dựa trên số xung trên mỗi vòng và số độ trên một vòng.

3. Cập nhật vị trí:

- Thêm giá trị vừa tính được vào giá trị hiện tại.

4. Thiết lập lại bộ đếm:

- Reset bộ đếm xung để chuẩn bị cho lần đọc tiếp theo

2.2.2. Xử lý dữ liệu nhận qua UART từ GUI

Các bước xử lý chuỗi dữ liệu

1. Phân tích chuỗi UART:

- Tách chuỗi UART nhận được thành các phần
- Lưu trữ loại lệnh vào dữ liệu điều khiển.

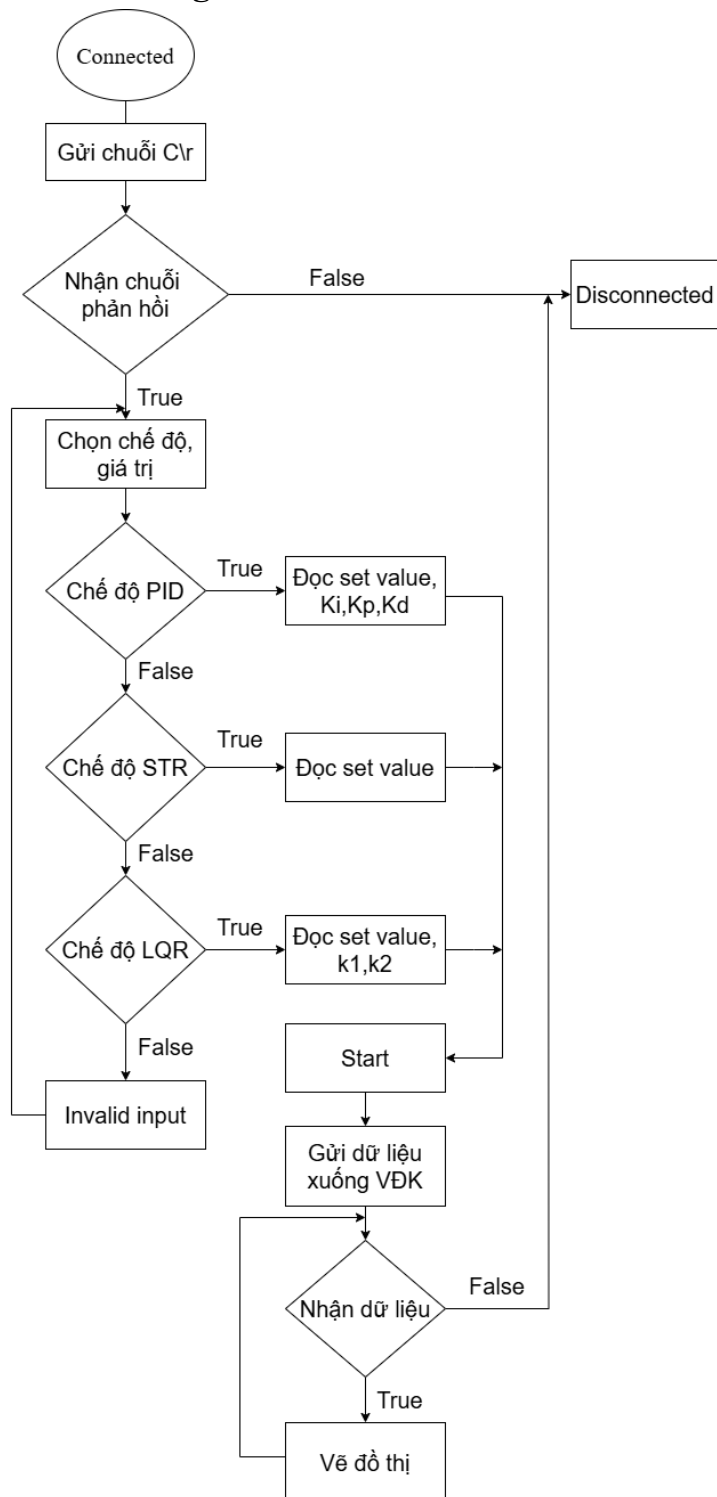
2. Xử lý loại lệnh:

- Lệnh "PID":
 - Chuyển sang chế độ PID.
 - Đọc các tham số điều khiển từ chuỗi: giá trị đặt, hệ số điều khiển (K_p, K_i, K_d).
- Lệnh "STR":
 - Chuyển sang chế độ điều khiển STR.
 - Đọc tham số điều khiển từ chuỗi: giá trị đặt.
- Lệnh "LQR":
 - Chuyển sang chế độ điều khiển LQR.
 - Đọc các tham số điều khiển cần thiết từ chuỗi (giá trị đặt, K_1, K_2).
- Lệnh "C":
 - Gọi hàm xử lý dữ liệu để tạo chuỗi kết quả mặc định.
 - Truyền chuỗi qua UART để GUI nhận được tín hiệu nhằm xác định kết nối giữa GUI và VDK không bị gián đoạn
- Lệnh "STOP":
 - Dừng hệ thống điều khiển:
 - Đặt giá trị ban đầu cho trạng thái và các biến liên quan.
 - Ngừng động cơ.

2.3. Software

2.3.1 Giao tiếp giữa GUI và vi điều khiển

2.3.1.1. Cấu trúc chương trình:

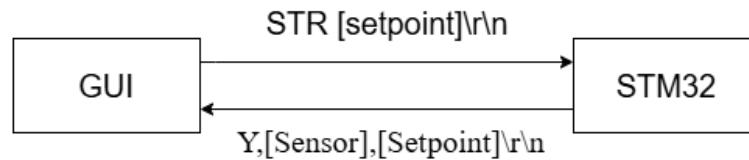


Hình 2.9 Lưu đồ giải thuật của GUI

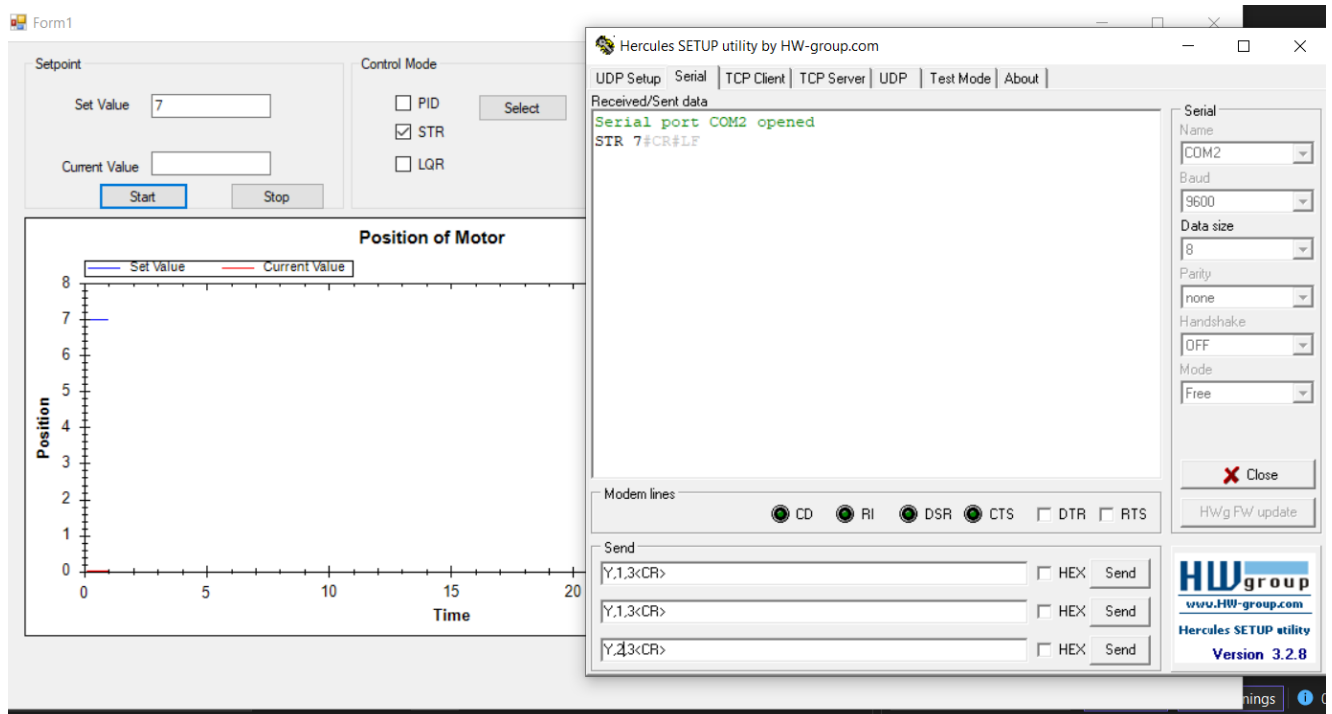
2.3.1.2. Chương trình đọc và gửi các thông số qua UART

Ta quy định chuỗi truyền các thông số từ GUI về VĐK qua UART như sau:

- PID: PID [setpoint] [Kp] [Ki] [Kd]\r\n
- STR: STR [setpoint]\r\n
- LQR: LQR [setpoint] [K1] [K2]\r\n

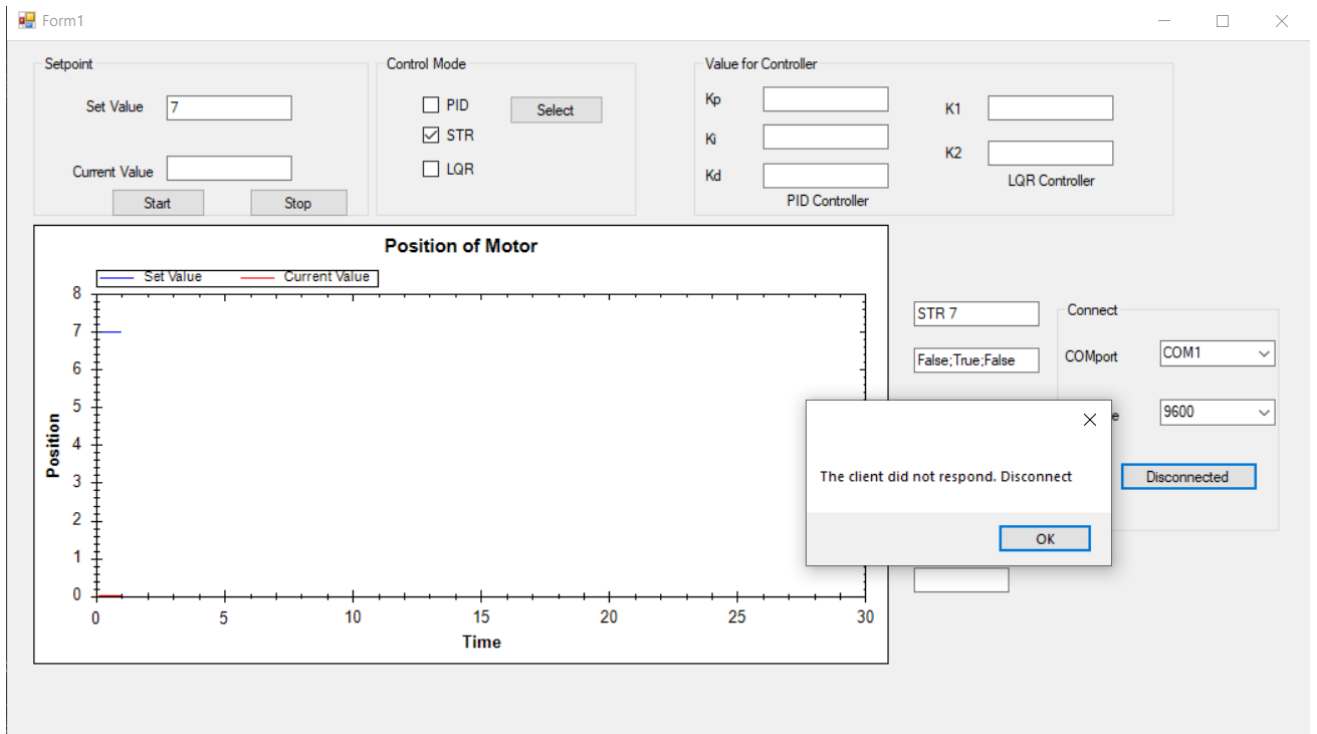


Hình 2.10 Frame truyền của GUI và vi điều khiển trong chế độ STR

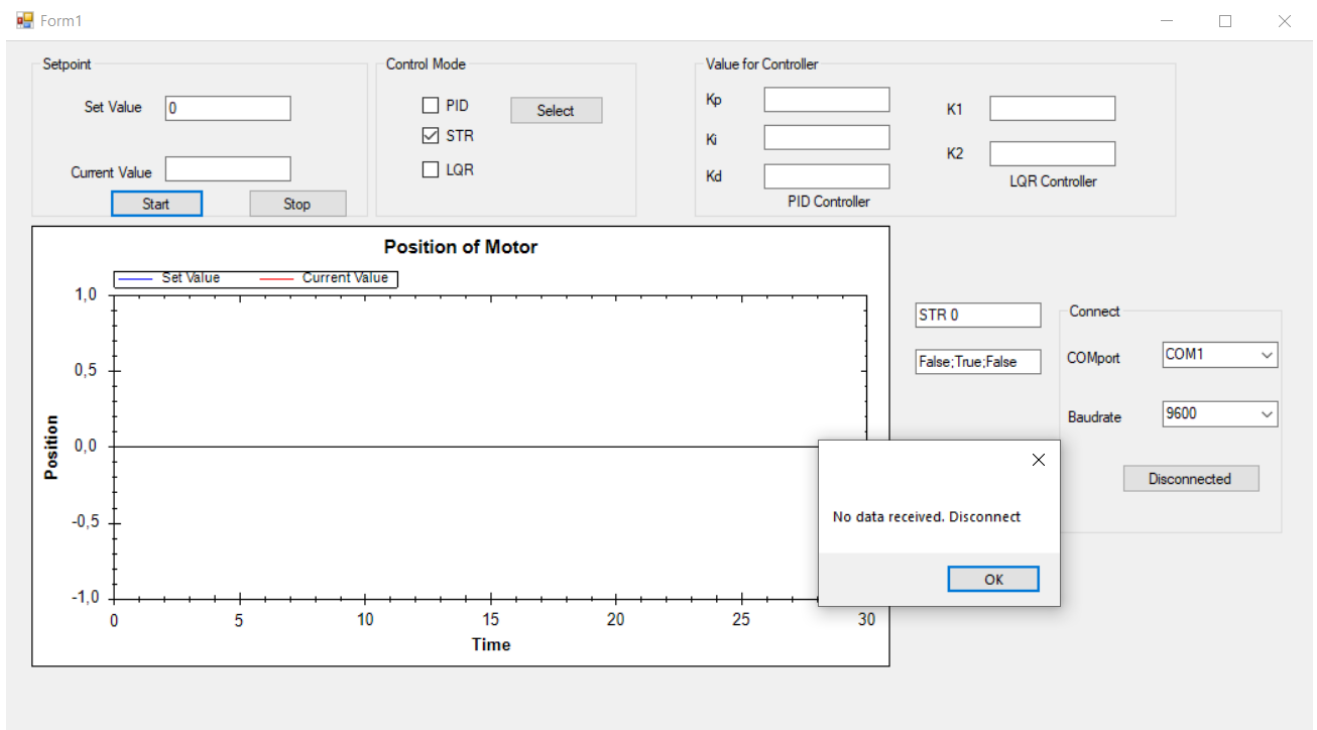


Hình 2.11 Demo truyền dữ liệu trong chế độ STR

Nếu không nhận tín hiệu phản hồi sau 1s, chương trình ngắt tự động kết nối và hiện thông báo.



Hình 2.12 Tự động ngắt kết nối nếu không nhận tín hiệu phản hồi sau 1s



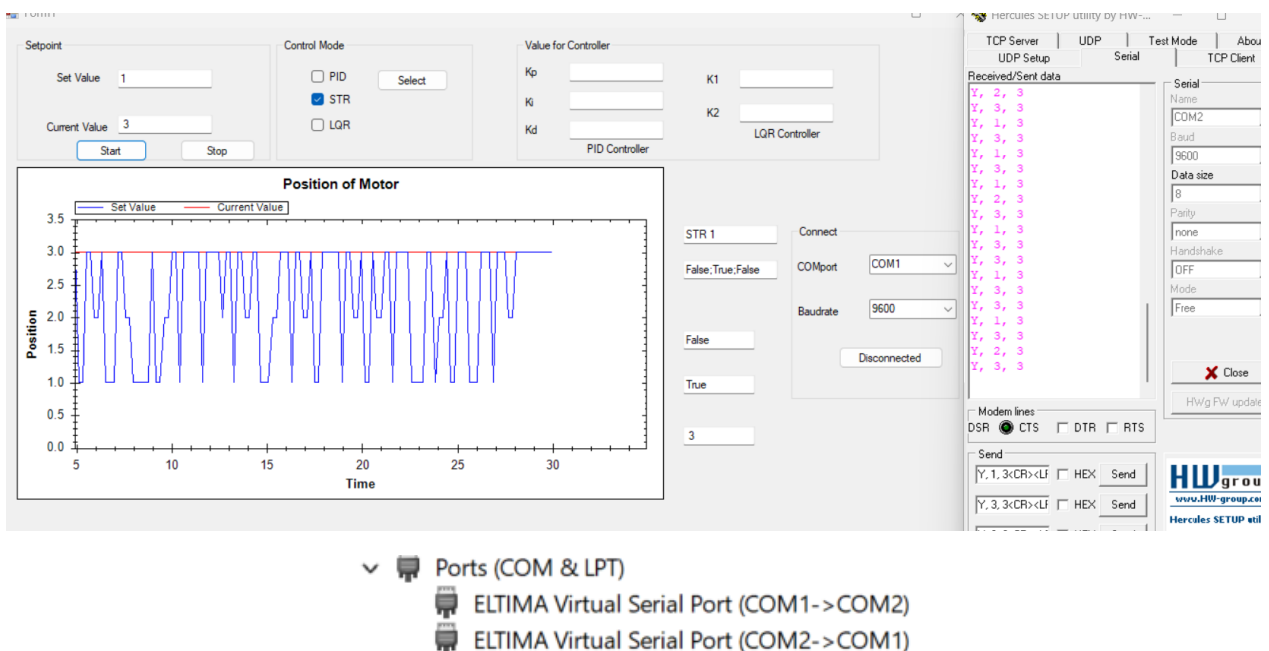
Hình 2.13 Tự động ngắt kết nối nếu dữ liệu đang truyền nhận bị ngắt

2.3.1.3. Thiết lập chuỗi gửi vào buffer gửi giá trị sensor cho GUI

- Ta thiết lập chuỗi truyền cho UART từ vi điều như sau: Y, [Sensor], [Setpoint]\r\n

Ở trường hợp mặc định 2 giá trị Sensor và Setpoint sẽ bằng 0. GUI chỉ nhận tín hiệu để xác định kết nối vẫn giữa GUI và VĐK không bị gián đoạn

Khi điều khiển GUI sẽ lấy giá trị setpoint để đối chiếu với giá trị đã gửi xuống và giá trị sensor để thực hiện vẽ đồ thị.



Hình 2.14 Kiểm tra chức năng vẽ đồ thị với phần mềm Hercules qua COM ảo

2.3.2. Giao diện

Ta tạo giao diện C# như sau:

The screenshot shows a Windows application window titled "Form1". It contains several control panels. On the top left, the "Setpoint" panel has "Set Value" and "Current Value" text boxes, and "Start" and "Stop" buttons. Next to it is the "Control Mode" panel with checkboxes for "PID", "STR", and "LQR", and a "Select" button. To the right is the "Value for Controller" panel with input fields for "Kp", "Ki", "Kd", "K1", and "K2", and labels for "PID Controller" and "LQR Controller". Below these is a large plot area titled "Position of Motor" with "Position" on the y-axis (0.0 to 1.2) and "Time" on the x-axis (0 to 30). The plot shows a blue line for "Set Value" and a red line for "Current Value". On the far right is a "Connect" panel with "COMport" (set to "COM1") and "Baudrate" (set to "9600") dropdown menus, "False" and "True" buttons, and a "Connect" button.

Kết nối COMport:

This is a close-up of the "Connect" section. It shows the "COMport" dropdown menu set to "COM1" and the "Baudrate" dropdown menu set to "9600". Below these is a button labeled "Connected".

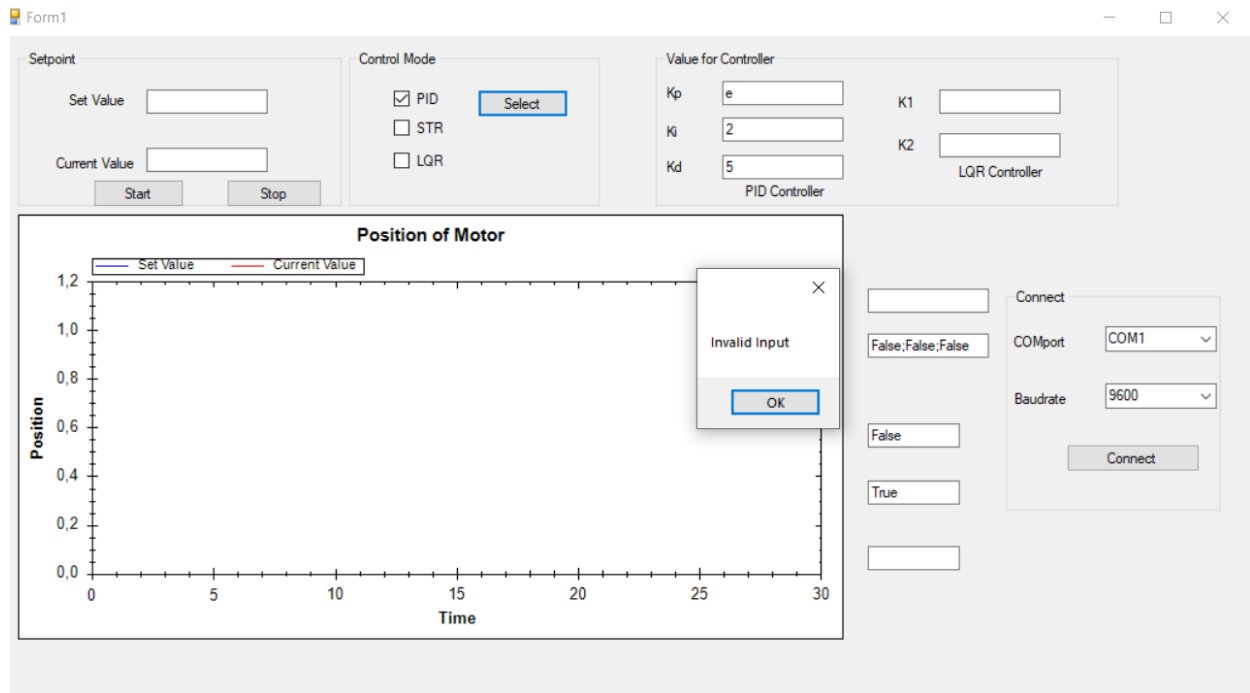
- Chương trình liên tục kiểm tra danh sách các cổng COM được kết nối và so sánh danh sách COMport mới với danh sách cũ. Nếu danh sách mới khác thì cập nhật.

- Nút “Connect” sau khi bấm lần đầu sẽ hiện thị trạng thái kết nối hoặc không kết nối của GUI. Bấm 1 lần nữa để thay đổi trạng thái

Chọn chế độ điều khiển:

This is a close-up of the "Control Mode" and "Value for Controller" sections. The "Control Mode" panel has checkboxes for "PID", "STR", and "LQR", and a "Select" button. The "Value for Controller" panel has input fields for "Kp", "Ki", "Kd", "K1", and "K2", and labels for "PID Controller" and "LQR Controller".

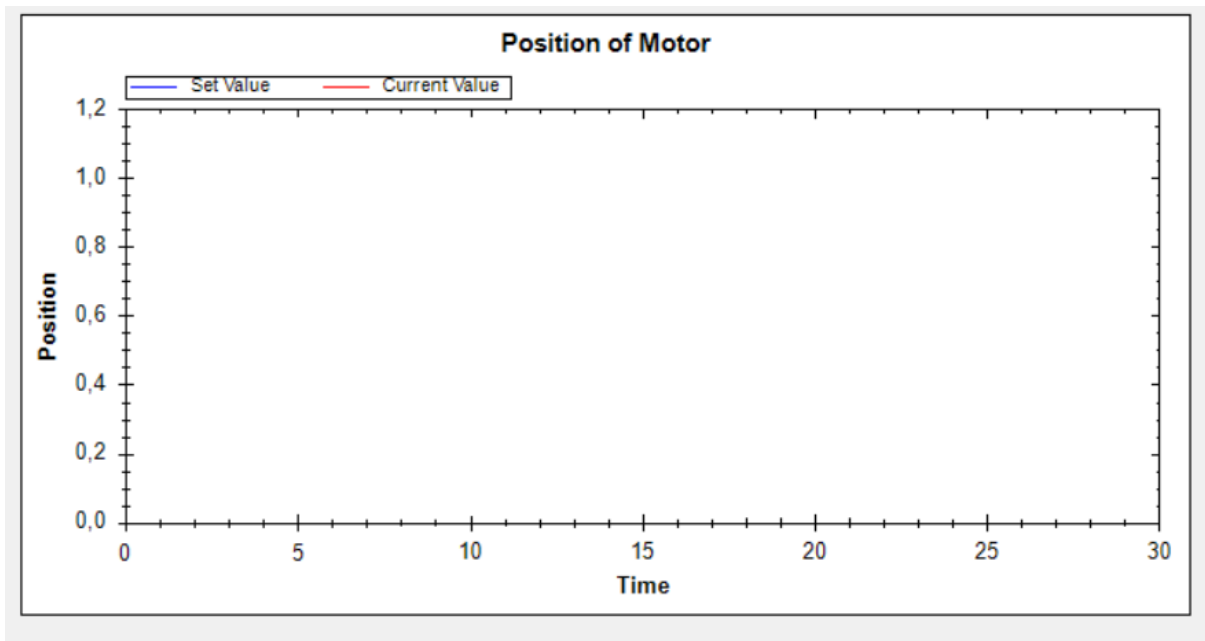
- Tích vào ô để chọn chế độ điều khiển, nhập dữ liệu K_i , K_p , K_d , K_1 , K_2 tương ứng rồi bấm “Select” để ghi nhận lựa chọn
- Nếu chọn sai sẽ báo lỗi



Truyền nhận dữ liệu điều khiển:

- “Set Value”: nhập giá trị cài đặt
- “Current Value”: hiển thị giá trị hiện tại nhận được từ vi điều khiển
- Nút “Start” bắt đầu truyền dữ liệu điều khiển theo frame truyền
- Nút “Stop” dừng truyền nhận và reset các Timer

Vẽ đồ thị:



- Sử dụng thư viện ZedGraph
- Trục tung là vị trí nhận được từ vi điều khiển, trục hoành là thời gian đếm bằng Timer1

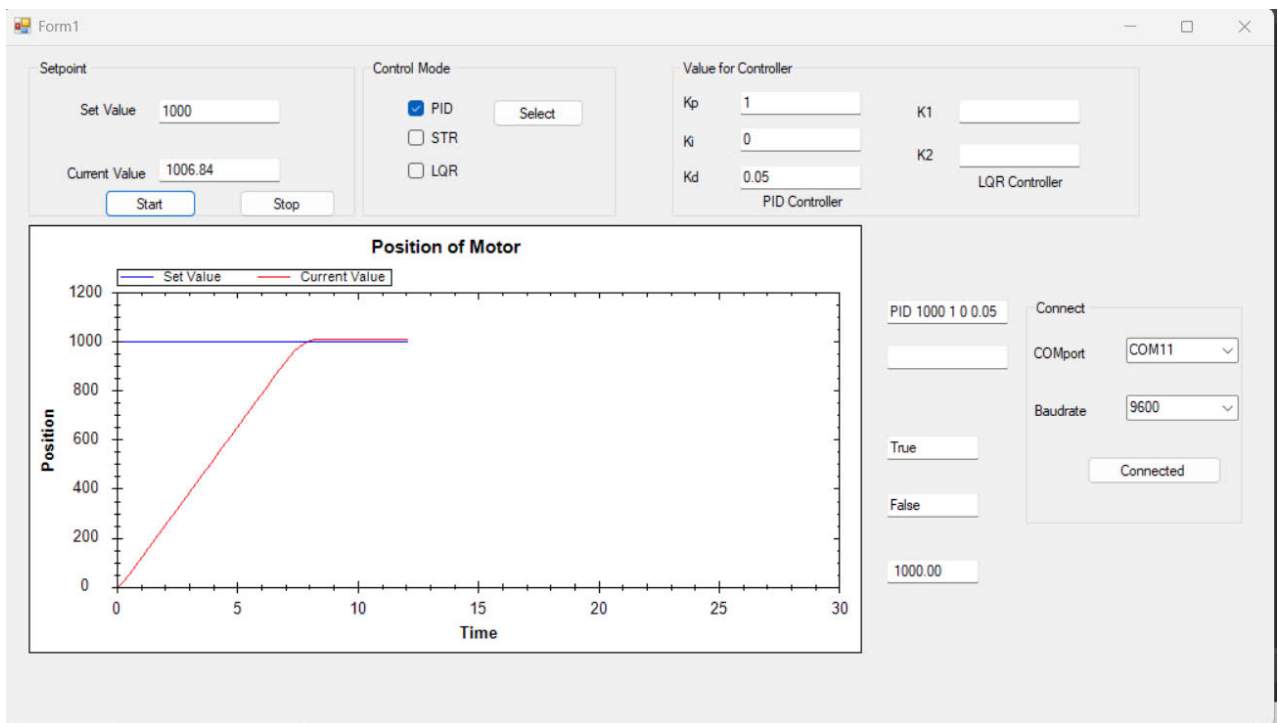
Hiện thị một vài thông số phụ:

Xét hình trên, chương trình điều khiển động cơ bằng chế độ PID

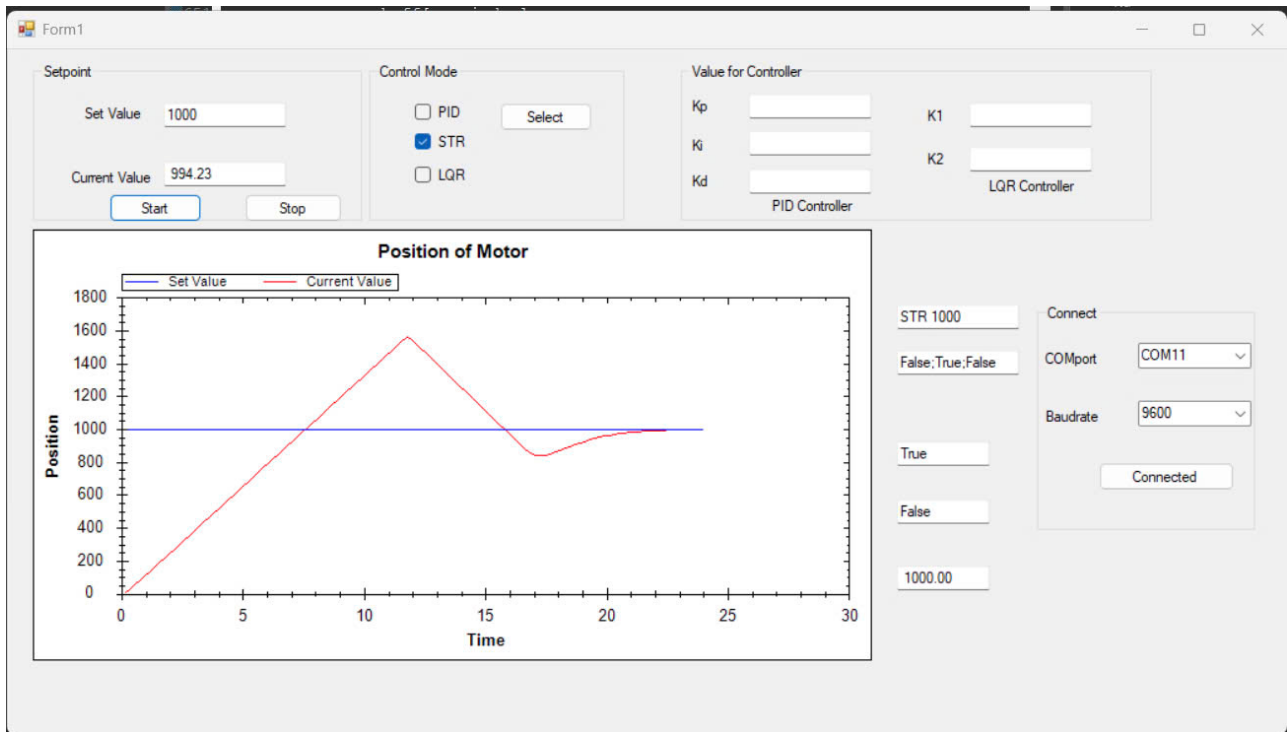
- Ô đầu hiển thị dữ liệu truyền theo frame PID: PID [setpoint] [Kp] [Ki] [Kd]\r\n
- Ô thứ hai hiển thị chế độ điều khiển đã chọn: True, False, False // PID, STR, LQR
- Ô thứ ba hiển thị trạng thái của đồ thị. Nếu False tức là đồ thị không vẽ hoặc đã dừng
- Ô thứ tư hiển thị trạng thái mất kết nối của chương trình. Nếu True tức là chương trình đang mất kết nối với vi điều khiển

3. Kết quả thực hiện:

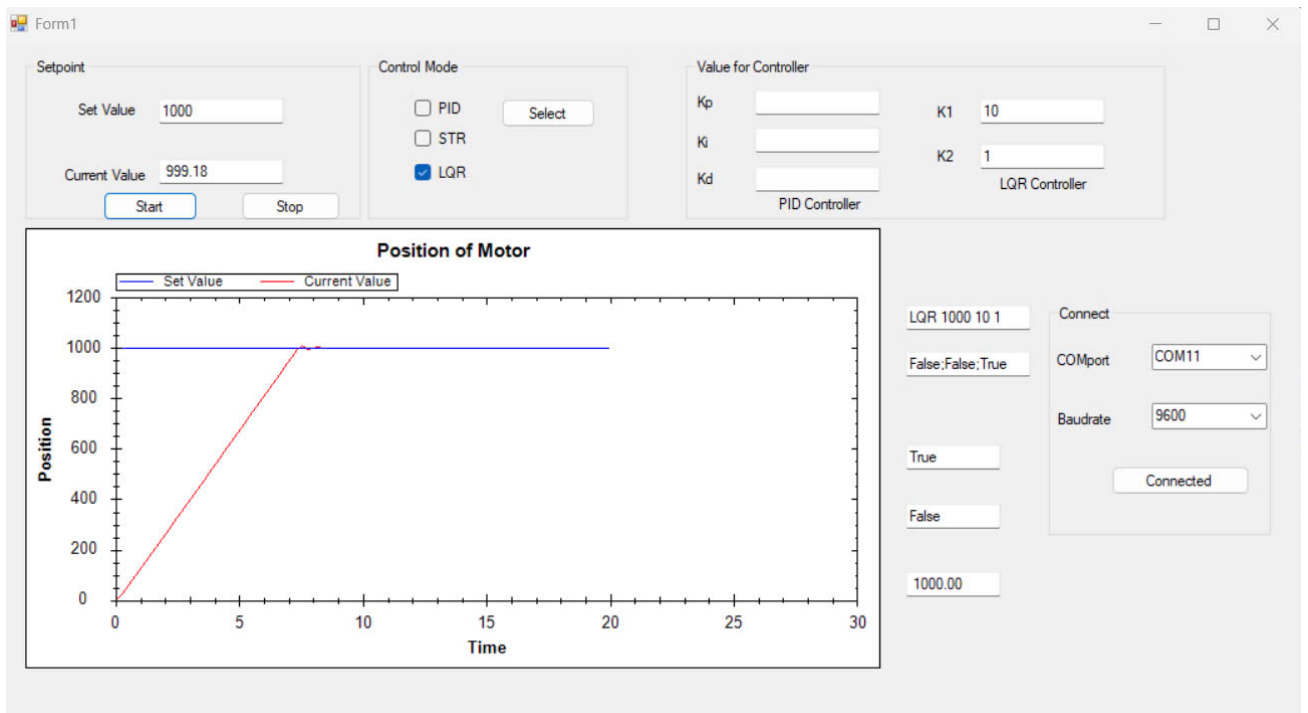
Thực hành điều khiển động cơ bằng các chế độ khác nhau với Set value = 1000



Hình 3.1 Kết quả thực nghiệm chế độ PID



Hình 3.2 Kết quả thực nghiệm chế độ STR



Hình 3.3 Kết quả thực nghiệm chế độ LQR