Cơ sở Lý thuyết

1. Thuật điều khiển PID và việc rời rạc hóa nó:

Trong miền thời gian, bộ điều khiển PID được mô tả bằng mô hình vào ra:

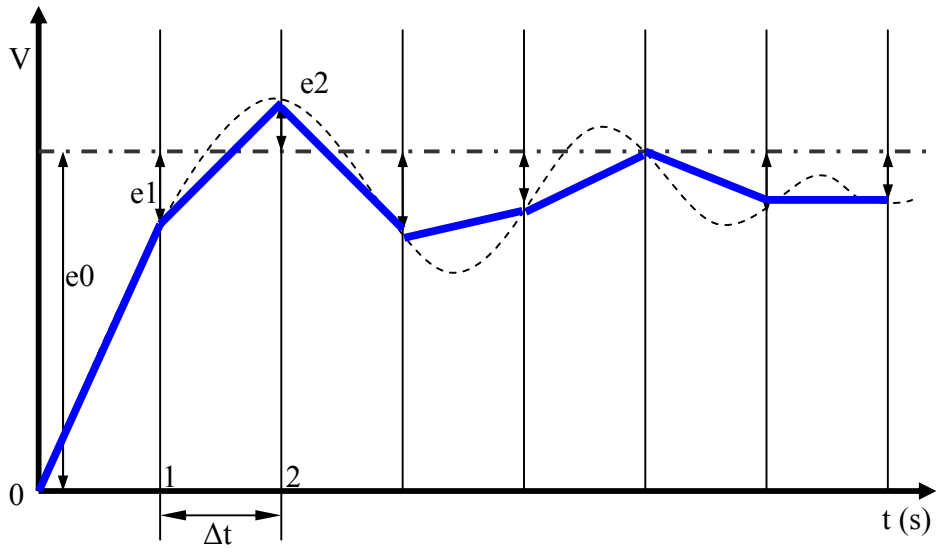
$$U(t) = K_P e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt}$$

trong đó e(t) là tín hiệu ngõ vào, u(t) là tín hiệu ngõ ra của bộ điều khiển.

Tuy nhiên, đối với Vi Điều khiển nói chung, việc tính toán các thành phần P,I,D- nói cách khác là tính các tích phân hay đạo hàm trong công thức trên là không thực hiện được. Lý do: CPU không thể tính toán chính xác tới mức $\Delta t = 0$, nghĩa là không liên tục.

Do đó, ta chỉ có thể tính toán gần đúng bằng cách ta cho $\Delta t = \varepsilon$ rất nhỏ nhưng lớn hơn 0.

Để tìm hệ thức PID rời rạc ,ta xét đồ thị sau đây:



Chú thích: - đường chấm gạch biểu diễn vận tốc cần thiết v_set.
- đường gạch gạch biểu diễn vận tốc thực tế của động cơ.

- đường gạch đậm là đồ thị rời rạc hóa của vận tốc động cơ.

- Δt là thời gian lấy mẫu.

Thành phần tích phân:

$\int_0^t e(t)\,dt = \lim_{\Delta t \to 0} (\Sigma\, e(t)\, \Delta t)$. Do đó khi lấy gần đúng $\Delta t = \varepsilon > 0$, ta có:

$$\int_0^t e(t)\,dt \cong \Sigma\, e(i)\, \Delta t \text{ , } i=0,1,2,3\ldots$$

Thành phần vi phân:

$de(t)/dt = \lim_{\Delta t \to 0} \{ [\, e(t_2) - e(t_1)\,] / \Delta t \}$. Do đó khi lấy gần đúng $\Delta t = \varepsilon > 0$, ta có:

$$de(t)/dt = [e(i+1) - e(i)]/ \Delta t \text{ , } i= 0,1,2,3\ldots$$

Tóm lại, ta có:

$$u(i)= Kp *e + Kd* [e(i+1) - e(i)]/ \Delta t + Ki* \Sigma\, e(i)\, \Delta t$$

Đặt e_delta (i+1)= e(i +1) - e(i)

e_sum (i+1) = Σ e(i) = e_sum (i) + e (i+1)

Trong công thức trên, thời gian sampling time Δt là rất nhỏ, ta bỏ qua Δt. Sau này, khi tìm các hệ số Kd, Ki bằng thực nghiệm, Kd và Ki lúc đó đã bao gồm cả Δt.

Khi đó, công thức trên được viết lại như sau:

$$u(i)= Kp * e + Ki * e\_sum + Kd* delta\_e$$

điều kiện biên: u(0) = duty >0.

2. <u>Đối tượng điều khiển:</u>

Đối tượng điều khiển là vận tốc động cơ DC (đc). Ta biết vận tốc động cơ DC phụ thuộc dòng điện hay điện áp mà ta cấp cho nó (dĩ nhiên phải nằm trong khoảng cho phép của đc). Cụ thể ở đây sử dụng đc 24VDC, do đó điện áp cấp không được quá 24V. Tuy nhiên, việc cấp áp cho đc trong một khoảng rộng từ 0 đến 24V là khó khăn.

Do đó ta điều khiển vận tốc theo xung PWM (Pulse Width Modulation), cụ thể là duty cycle.

Vậy các ngõ vào và ra của bộ điều khiển PID như sau:

Ngõ vào: e= vận tốc hiện tại (v_cur) - vận tốc thiết lập (v_set)

Ngõ ra: u = %duty cycle

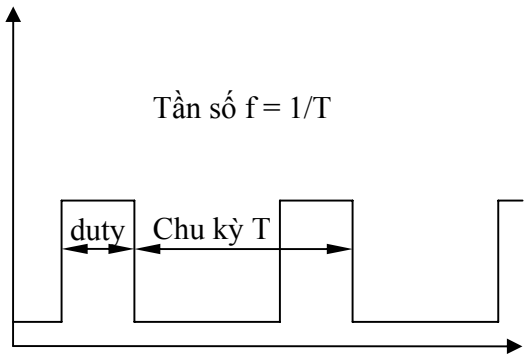
Phụ thuộc giữa %duty và vận tốc đc gần như tuyến tính nên để đơn giản, ta giả sử nó hoàn toàn tuyến tính. Vậy, ta có thể điều khiển vận tốc đc thông qua %duty.

3.Sơ đồ mạch: Giới thiệu các thành phần trong mạch và chức năng:

- *Vi điều khiển chính* của mạch điều khiển là PIC 16F88 của Microchip. PIC 16F88 có 7 kênh ADC 10bit, 1 PWM để điều khiển đc.

Do đó, ta sử dụng 4 kênh ADC dùng để nhập dữ liệu (hệ số PID chỉnh bằng tay và vận tốc mong muốn). Chân ADC nối với một cầu phân áp bằng biến trở. Giá trị biến trở làm thay đổi điện áp dạng analog trên chân ADC, và PIC lấy điện áp đó chuyển thành giá trị digital. Việc tính toán cụ thể xin xem phần 5 của báo cáo này.

PWM (Pulse Width Modulation) là một module mở rộng của PIC 16F88. Nó có chức năng tạo ra một dãy xung có %duty và tần số xác định. Các giá trị %duty và tần số của xung hoàn toàn có thể hiệu chỉnh bằng phần mềm.



Chân T0CLKI của PIC16F88 là chân vào của xung clock bộ định thời 0. Chân này được nối với dây tín hiệu của encoder. Timer0 lúc này có chức năng counter đếm số xung phát từ encoder. Timer1 sẽ định thời trong 1 khoảng thời gian. Khi Timer1 tràn, ta lấy giá trị của Timer0. Dựa vào giá trị

này, có thể tính được vận tốc đc ở thời điểm này. Phần tính toán xem mục 5.

- *Driver điều khiển đc* là chip L293 của Texas Instruments. Chip này có các đặc điểm sau:

Dòng tối đa 1A, quá dòng 1.2 A.

Điện áp tối đa là 33V.

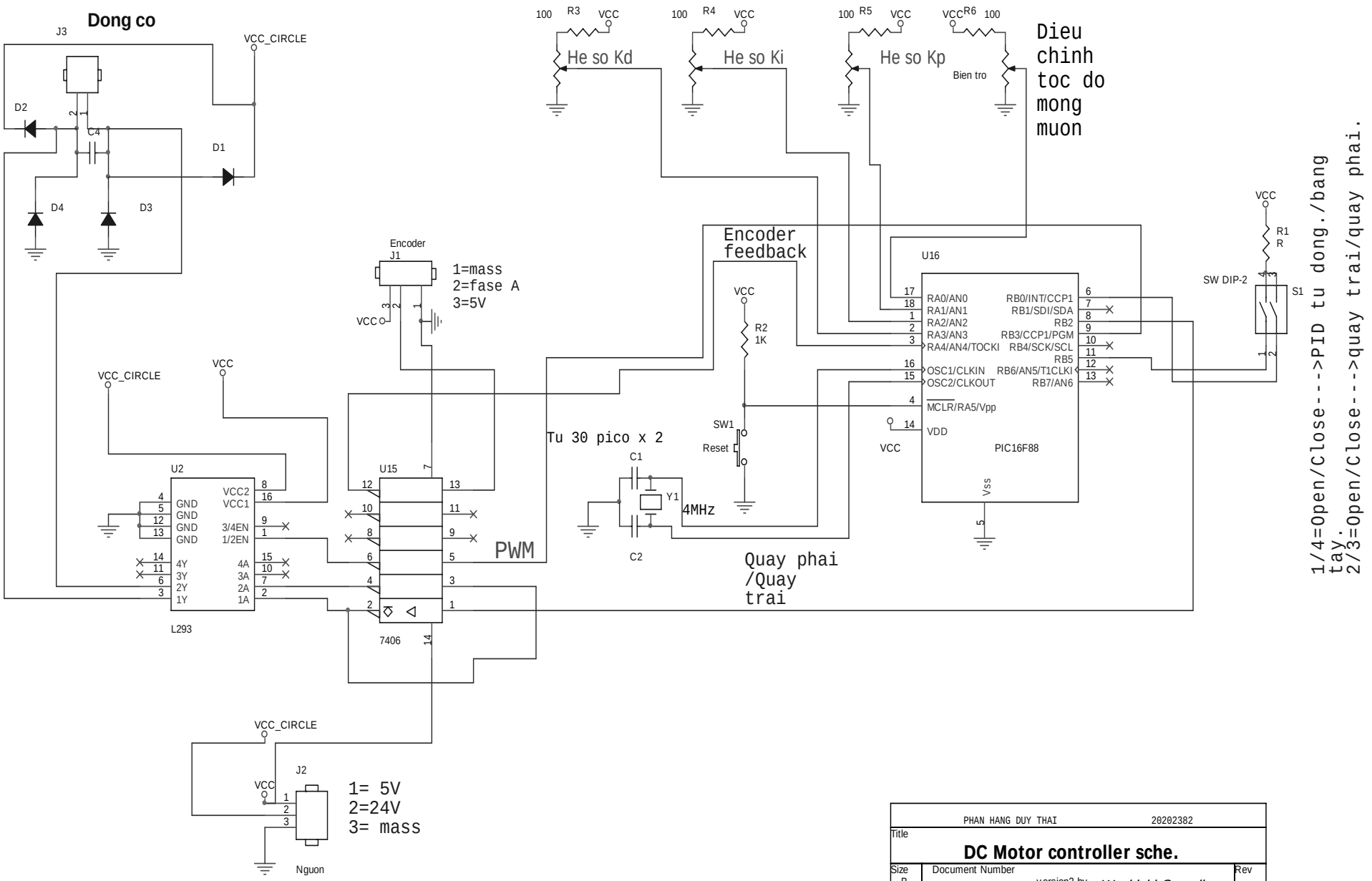
Có thể lái cùng lúc 2 đc DC. Tuy nhiên, ta chỉ sử dụng 1 cầu H để điều khiển 1 đc. Ta chọn cầu H1 có các chân ngõ vào là chân enable 1/2EN, chân chọn chiều quay 1A, 2A. Các chân ra là 1Y, 2Y. Chân 1/2EN được nối với chân cấp xung PWM của PIC 16F88.

- *Chip 74LS06 là cổng NOT* . Mục đích để khuyếch đại xung clock của encoder trước khi đưa vào T0CLKI.

- Mạch sử dụng 2 nguồn: 5V cho các chip hoạt động và 24V cho động cơ.

Do cách đấu dây dưới đây, ta có chân RB2 của PIC16F88 =0/1 thì đc quay thuận/ nghịch chiều kim đồng hồ.

Sau đây là sơ đồ mạch được vẽ bằng OrCAD 9.2. Thứ tự chân IC, cách nối dây cụ thể, công dụng của các chân đặc biệt xin xem trên sơ đồ mạch này.



PHAN HANG DUY THAI		20202382
Title: **DC Motor controller sche.**		
Size: B	Document Number	Rev
	version2 by *Wonbinbk@gmail.com*	
Date: Wednesday, April 27, 2005	Sheet 1 of 1	

Mạch có 2 chế độ hoạt động: tự động và bằng tay.
Chế độ tự động: các hệ số Kp, Ki, Kd được lập trình sẵn. Các hệ số này có được sau quá trình thực nghiệm theo phương pháp thứ hai của Ziegler-Nichols. (thực tế có thể xem như đây là phương pháp giả Ziegler-Nichols. Lí do : việc xác định chu kì của dao động khi đối tượng bị dao động theo phương pháp Ziegler-nichols là rất khó lấy được một cách chính xác)

Chế độ bằng tay: các hệ số Kp, Ki, Kd có được từ các biến trở , do người dùng điều chỉnh tùy ý.
Trong cả hai chế độ, vận tốc được thiết lập thông qua biến trở.

4.Sơ lược về giải thuật lập trình :

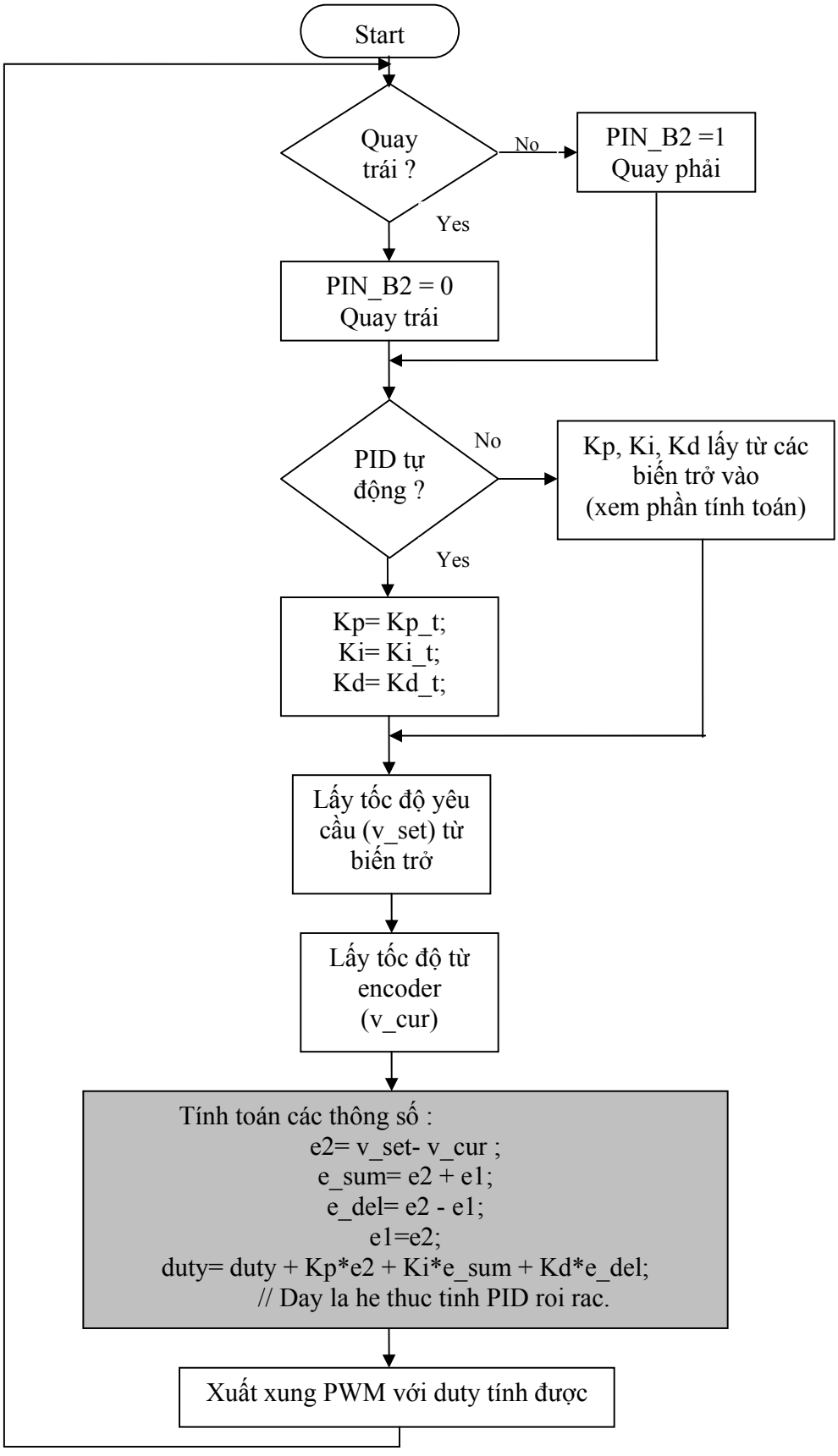
Các ký hiệu:

- Kp, Ki, Kd lần lượt là các hệ số Kp,Ki, Kd.
- Kp_t, Ki_t, Kd_t lần lượt là các giá trị tìm được từ thực nghiệm.
- e2 là sai lệch hiện tại (trong lúc đang xét).
- e1 là sai lệch ngay trước đó.
- e_sum là tổng của tất cả các sai lệch từ lúc bắt đầu đến thời điểm đang xét.
- e_del là hiệu số của hai sai lệch e2 và e1, hay nói cách khác, đó là độ biến thiên sai lệch.
- V_set là tốc độ được thiết lập qua biến trở.
- V_cur là tốc độ hiện tại đọc được từ encoder.
- Duty là số phần trăm duty cycle của xung PWM cần cung cấp.

Ban đầu: Duty phải có một giá trị khác 0, mục đích để động cơ thắng lực quán tính.

e1=e2=0
e_sum =0
e_del= 0

Theo đó ta có lưu đồ sau:


Start
Quay trái ?
No
PIN_B2 =1
Quay phải
Yes
PIN_B2 = 0
Quay trái
PID tự động ?
No
Kp, Ki, Kd lấy từ các biến trở vào
(xem phần tính toán)
Yes
Kp= Kp_t;
Ki= Ki_t;
Kd= Kd_t;
Lấy tốc độ yêu cầu (v_set) từ biến trở
Lấy tốc độ từ encoder
(v_cur)
Tính toán các thông số :
e2= v_set- v_cur ;
e_sum= e2 + e1;
e_del= e2 - e1;
e1=e2;
duty= duty + Kp*e2 + Ki*e_sum + Kd*e_del;
// Day la he thuc tinh PID roi rac.
Xuất xung PWM với duty tính được

5.Một số tính toán trong lập trình

a. Đọc và tính giá trị từ biến trở:

- Vi Điều khiển PIC16F88 của Microchip ™ có tất cả 7 kênh ADC 10 bit, nằm trên các chân AN0 đến AN6.
- Biến trở sử dụng trong mạch là loại Volume 10KΩ, các biến trở hạn dòng có trị số rất nhỏ (330Ω) nên có thể bỏ qua khi tính toán.
- Resolution của ADC là 10bit= 1024. Giá trị ADC đọc vào là từ 0 → 1023
- Với các biến trở điều chỉnh giá trị Kp, Ki, Kd, ta gán giá trị tương ứng là 0→100. Như vậy, bước thay đổi nhỏ nhất là 100/1024 = 0.0976. Điều này có nghĩa là Kx = giá trị đọc từ ADC * 0.0976.
- Tương tự như vậy, với biến trở thiết lập tốc độ, ta gán giá trị tốc độ tương ứng là 0→ 300 vòng / phút. Bước thay đổi nhỏ nhất là 300/ 1024= 0.293. Nghĩa là tốc độ v_set = giá trị đọc từ ADC * 0.293.

b. Đọc và tính vận tốc thực sự của động cơ từ encoder:

Ta sử dụng Timer0 ở chế độ counter với xung clock có được từ encoder. Timer1 ở chế độ timer 16bit.
Encoder 200 xung/vòng

Trước tiên đếm số xung trong 1000μs: timer1 sẽ tràn sau 1000 μs, trong thời gian đó, timer0 đếm số xung nhận được. Khi timer1 tràn, chương trình thực hiện ngắt. Giá trị từ Timer0 chính là số xung trong 1000 μs. Ta gọi giá trị này là T0.

Như vậy, số vòng/ phút= (số xung/200)/ (60* 1000) ms
= (số xung/ms) * 12.10^6
hay vận tốc đc v_cur = T0* 12.10^6 (vòng/ phút)

c. Hệ thức PID cuối cùng:

duty= duty + Kp*e2 + Ki*e_sum + Kd*e_del

6. Cách điều chỉnh các hệ số:

(Phần này được tham khảo trực tiếp từ trang web http://www.acroname.com/brainstem/ref/h/i/iA.html)

Đầu tiên ta chỉnh cho Kp =1 và Kd, Ki=0. Nghĩa là chỉ điều khiển P. Sau đó ta tăng Kp lên dần dần, và quan sát động cơ. Khi thấy động "dao động", nghĩa là nó lúc nhanh lúc chậm, ta lấy Kp tại đó nhân với 0.6 để tính toán. Nghĩa là Kp_t= 0.6 * Kp_dao động.

Sau đó ta tăng Kd lên dần dần (giữ nguyên Kp), nếu Kd quá lớn sẽ làm động cơ bị dao động mạnh. Giảm Kd lại cho đến khi động cơ hết dao động.

Điều chỉnh Ki là khó nhất, bắt đầu từ giá trị rất nhỏ, ví dụ lấy Ki= 1/ Kd chẳng hạn. Hệ số Ki không cần lớn vì động cơ tự nó đã chứa thành phần Ki (thể hiện ở moment quán tính, hay sức ì của động cơ). Do đó, thường với đối tượng điều khiển là nhiệt độ hay động cơ (các đối tượng có quán tính) thì chỉ cần bộ điều khiển PD là đủ.

Bảng sau chỉ rõ các ảnh hưởng của Kp,Ki,Kd đến các đặc tính của hệ thống:

Đáp ứng của hệ thống	**Thời gian tăng**	**Vọt lố**	**Thời gian ổn định**	**Sai lệch so với trạng thái bền**
K_P	**Giảm**	**Tăng**	**Ít thay đổi**	**Giảm**
K_I	**Giảm**	**Tăng**	**Tăng**	**Triệt tiêu**
K_D	**Ít thay đổi**	**Tăng**	**Tăng**	**Ít thay đổi**

Ta thấy phương pháp này có nhiều nhược điểm như: không có cơ sở toán học vững chắc, không tối ưu (vì tất cả đều chọn hết sức ngẫu nhiên trong một vùng giá trị nào đó). Tuy vậy , động cơ có thể có đáp ứng tốt, và độ quá điều chỉnh theo như website trên đưa ra là không quá 25% (trong khi phương pháp khác thì thường có độ quá điều chỉnh (hay vọt lố) chừng 40%). Đáp ứng của động cơ sẽ càng "tốt" nếu thời gian lấy mẫu PID càng nhỏ.