LỜI NÓI ĐẦU

Trong nhiều lĩnh vực sản xuất công nghiệp hiện nay, nhất là ngành công nghiệp luyện kim, chề biến thực phẩm… vấn đề điều khiển nhiệt độ đặc biệt được chú trọng đến vì nó là một yếu tố quyết định chất lượng sản phẩm. Nắm được tầm quan trọng của vấn đề trên em tiến hành nghiên cứu và thiết kế một hệ thống điều khiển số nhiệt độ, với mong muốn là giải quyết những yêu cầu trên, và lấy đó làm đề tài tốt nghiệp cho mình.

Những kiến thức năng lực đạt được trong quá trình học tập ở trường sẽ được đánh giá qua đợt bảo đồ án cuối khóa. Vì vậy em cố gắng tận dụng tất cả những kiến thức đã học ở trường cùng với sự tìm tòi nghiên cứu, để có thể hoàn thành tốt đồ án này. Những sản phẫm những kết quả đạt được ngày hôm nay tuy không có gì lớn lao. Nhưng đó là những thành quả của những năm học tập. Là thành công đầu tiên của em trước khi ra trường .

Mặt dù em rất cố gắng để hoàn thành tập đồ án này đúng thời hạn, nên không tránh khỏi những thiếu sót mong quí thầy cô thông cảm. Em mong được đón nhận những ý kiến đóng góp. Cuối cùng xin chân thành cảm ơn quí thầy cô và các bạn sinh viên.

Sinh viên thực hiện

Lê Thanh Tùng

BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC DÂN LẬP HẢI PH ÒNG
KHOA ĐIỆN _ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
ĐỘC LẬP_ TỰ DO _HẠNH PHÚC

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ và tên sinh viên thực hiện : Lê Thanh Tùng
Lớp : ĐC 1001
Ngành : Điện công nghiệp và dân dụng

1.Tên đề tài : Nghiên cứu và thiết kế hệ thống điều khiển số nhiệt độ
2. Các số liệu ban đầu :
Công suất lò: 5 (KW).
Dải nhiệt độ đo: 300-1200 (độ C).
3.Các yêu cầu thiết kế :
+Thiết kế bằng 8051.
+Xây dựng sơ đồ cấu trúc có bộ điều khiển mềm bên trong VXL:
-Bộ PID.
-Tổng hợp bằng thiết bị bù nối tiếp và bù hồi tiếp.
-Thực hiện hồi tiếp trạng thái và tính toán các hệ số hồi tiếp trạng thái.
+Mô hình hoá hệ thống bằng Matlab hoặc Simulink.
+Viết chương trình tổng hợp hệ thống bằng C++.
+Thông báo kết quả đạt được trên mô hình.

4.Giáo viên hướng dẫn : Ths.Nguyễn Trọng Thắng
5. Ngày giao nhiệm vụ :
6.Ngày hoàn thành nhiệm vụ :

Thông qua bộ môn.
Ngày___tháng___năm___
Giáo viên hướng dẫn
Chủ nhiệm bộ môn

MỤC LỤC

CHƯƠNG 1:
GIỚI THIỆU CHUNG

1.1. Giới thiệu về hệ thống điều khiển

Ngày nay các hệ thống điều khiển tự động được sử dụng trong nhiều lĩnh vực và có rất nhiều ứng dụng khác nhau. Hệ thống điều khiển của nhiều nhà máy hiện đại có chứa nhiều mạch điều khiển, nhiều trong số chúng có tác động qua lại với nhau. Trong những nhà máy và hệ thống điều khiển hiện đại như vậy, việc truyền và xử lý số liệu trong các khâu của hệ thống giữ vai trò quan trọng. Công việc này được thực hiện rất tiện lợi và hiệu quả trong các hệ thống điều khiển số, đặc biệt là các hệ thống điều khiển số có sự trợ giúp của máy tính.

1.1.1. Hệ liên tục:

Là hệ thống trong đó các tín hiệu tác động trong hệ là các hàm liên tục theo thời gian.

1.1.2. Hệ gián đoạn:

Là hệ thống mà trong đó có một hoặc nhiều phần tử nhận thông tin vào hay xuất thông tin ra là một hàm rời rạc theo thời gian.

1.1.3.Các phần tử cơ bản của hệ thống điều khiển tự động:

Hệ thống điều khiển tự động là hệ thống được xây dựng từ ba bộ phận chính sau đây:

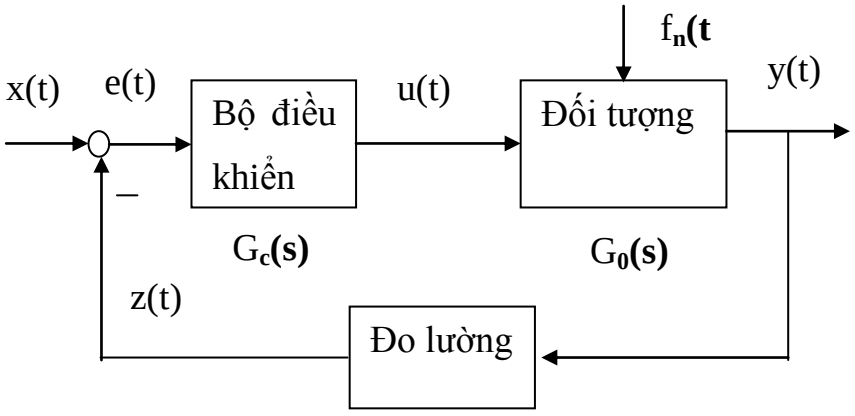
- Thiết bị điều khiển.
- Đối tượng điều khiển.
- Thiết bị đo lường.

Các tín hiệu tác động lên hệ thống:

x(t) tín hiệu đầu vào của hệ thống (tín hiệu chủ đạo).

y(t) tín hiệu đầu ra của hệ thống.

u(t) tín hiệu điều khiển tác động lên đối tượng.



Sơ đồ hệ thống điều khiển tự động.

e(t) sai lệch điều khiển,

z(t) tín hiệu phản hồi,

$f_n(t)$ nhiễu tác động lên đối tượng.

Thiết bị điều khiển là bộ phận quan trọng, nó tạo ra tín hiệu điều khiển u(t) tác động vào đối tượng điều khiển nhằm điều khiển được đầu ra của đối tượng- y(t) thoả mãn yêu cầu, mục đích của bài toán nghĩa là nó phải đáp ứng được các chỉ tiêu kinh tế, kỹ thuật đề ra.

Bộ điều khiển là thiết bị kỹ thuật mà trong đó khi xây dựng người ta xây dựng cấu trúc hoặc là cài đặt theo những luật điều khiển nào đó (P, PI, PID) nhằm mục đích tạo ra tín hiệu điều khiển u(t) là tín hiệu thoả mãn những yêu cầu và chỉ tiêu kỹ thuật. Trong thực tế các bộ điều khiển thường được xây dựng trên các luật điều khiển là :

- Luật điều khiển tỉ lệ P :

$u(t) = K_P .e(t)$; với K_P là hệ số khuếch đại tỉ lệ.

- Luật điều khiển tích phân I :

$u(t) = \frac{1}{T_i}\int_0^t e(t).dt$; với T_i là hằng số thời gian tích phân.

- Luật điều khiển vi phân D :

$u(t) = T_d.\frac{de(t)}{dt}$; với T_d là hằng số thời gian vi phân.

- Luật điều khiển tỷ lệ - tích phân PI :

$$u(t) = K_p \{e(t) + \frac{1}{Ti}\int_0^t e(t).dt \}$$

- Luật điều khiển tỷ lệ - vi phân PD :

$$u(t) = K_p\{e(t) + T_d.\frac{de(t)}{dt}\}$$

- Luật điều khiển tỉ lệ - vi phân - tích phân PID :

$$u(t) = Kp\{e(t) + T_d.\frac{de(t)}{dt} + \frac{1}{T_i}\int_0^t e(t).dt \}$$

Ngày nay do sự phát triển mạnh mẽ của kỹ thuật máy tính và công nghệ phần mềm đã tạo nền tảng cho sự phát triển của các bộ điều khiển số. Các bộ điều khiển số cũng được xây dựng trên cơ sở các luật điều khiển trên:
Bộ điều khiển số PID có cấu trúc như sau :

$$u(k) = K_p \{e(k) + \frac{T_s}{T_i}\sum_{k=1}^{i} e(k) + \frac{T_d}{T_s}[e(k) - e(k-1)]\}$$

Trong đó :

Các tín hiệu u(k), e(k) là những đại lượng rời rạc hoặc số,

T_d là thời gian lấy mẫu.

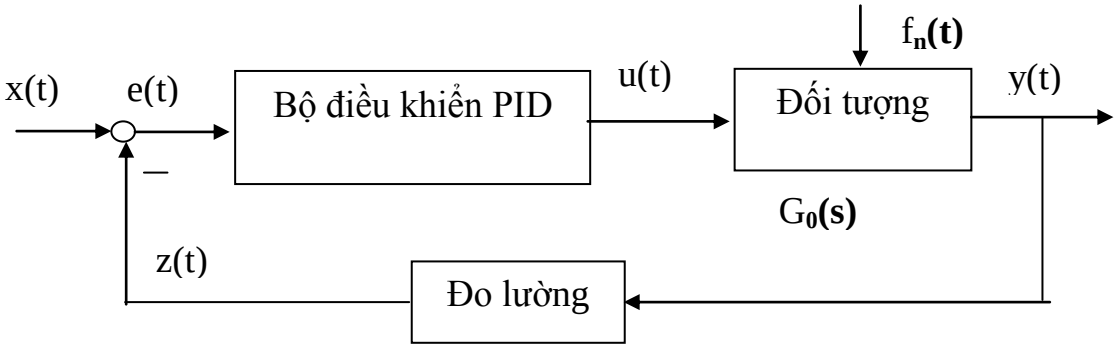
1.1.4. Hệ thống điều khiển PID:

Mặc dù các bộ điều khiển tỉ lệ - tích phân PI, tỉ lệ vi phân PD đã đáp ứng được tương đối đầy đủ các yêu cầu kỹ thuật, chỉ tiêu chất lượng trong nhiều trường hợp nhưng còn có những nhược điểm của nó. Để thoả mãn các yêu cầu về chất

lượng điều khiển trong thực tế người ta thường sử dụng tổ hợp bộ điều khiển tỉ lệ - tích phân - vi phân (PID). Bộ điều khiển PID được áp dụng rộng rãi trong các hệ thống điều khiển tự động vì nó có hàm trễ lớn đồng thời lại có được tất cả các ưu điểm của các bộ điều khiển P, PI, PD.

Hàm truyền đạt dạng:

$$G_c(s)=\frac{U(s)}{E(s)} = K_P(1+ \frac{1}{T_i s} + T_d.s)$$



Sơ đồ cấu trúc của hệ thống với bộ điều khiển PID:

Trong hoạt động của bộ điều khiển PID, bộ phận điều khiển tích phân có tác dụng loại trừ sự truyền tín hiệu tăng theo tỉ lệ đặc biệt là sự truyền tăng theo tỉ lệ của nhiễu lớn bằng cách hiệu chỉnh liên tục hoặc hồi tiếp đầu ra của thiết bị điều khiển.

Tác động điều khiển vi phân có khuynh hướng dự phòng trước các thay đổi trong tín hiệu sai lệch và do đó làm giảm khuynh hướng dao động. Tác động vi phân đáp ứng với tốc độ thay đổi của đầu ra nên còn gọi là tác động tốc độ, thời gian vi phân T_d được gọi là tốc độ xuất còn hệ số khuếch đại k_d là khoảng thời gian T_d mà trong đó có tác động vi phân hình thành bởi điều khiển tỉ lệ sớm hơn.

Các dạng cấu trúc bộ điều khiển PID:

- Mắc nối tiếp hai bộ điều khiển PI và PD:

$$G_c(s)=\frac{U(s)}{E(s)} = K_P(1 + \frac{1}{T_i s})(1 + T_d.s)$$

- Bộ điều khiển PID thực:

$$G_c(s)=\frac{U(s)}{E(s)} = K_P(1 + \frac{1}{T_i s} + \frac{T_d.s}{1+T_d.s})$$

- Nếu đặc tính tần số PID là đặc tính tiệm cận thì hàm truyền có dạng:

$$G_c(s)=\frac{U(s)}{E(s)} = K_P(1 + \frac{1}{T_i s})(\frac{T_d.s}{1+T_d.s})$$

- Bộ điều khiển PID số:

$$G_c(z) = K_p.(\frac{T_d}{T_s}.\frac{z-1}{z} + 1 + \frac{T_s}{T}.\frac{z}{z-1})$$

- Phương trình sai phân có dạng:

$$u(k.T_s) = K_p\left\{\frac{Td}{Ts}se(k-1)Ts + e(k.Ts) + \frac{Ts}{Ti}\sum_{k=0}^{n} e(k.Ts)\right.$$

1.2. Hệ thống điều khiển số

Do sự phát triển của các hệ thống điều khiển công nghiệp phức tạp, ngày càng lớn và yêu cầu các chỉ tiêu kỹ thuật, chất lượng ngày càng cao nên việc ứng dụng các bộ điều khiển số là cần thiết. Các hệ thống điều khiển số ra đời đã đem lại hiệu quả kỹ thật và kinh tế vô cùng lớn. Kỹ thuật điều khiển số vừa là công cụ vừa là nền tảng để phân tích và tổng hợp các hệ thống điều khiển. Các hệ thống điều khiển số được ứng dụng phổ biến trong các quá trình sản xuất như: điều khiển robot, máy công cụ, điều khiển các hệ thống thông tin liên lạc, các hệ thống thu thập dữ liệu, điều khiển và giám sát các quá trình công nghệ.

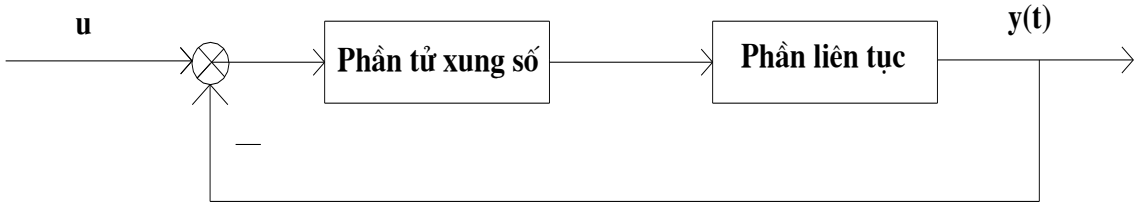
Hệ thống điều khiển số (DCS) là hệ thống điều khiển mà ở một khâu nào đó có một tín hiệu tác động dạng tín hiệu số. Hệ thống điều khiển số có những ưu điểm so với hệ thống điều khiển tương tự truyền thống là:

- Làm dễ dàng, thuận lợi cho việc thu thập và quản lý số liệu, trao đổi thông tin với hệ ngoài.
- Có tính mềm dẻo khi thay đổi cấu trúc và tham số của các bộ điều khiển nói riêng và hệ thống điều khiển nói chung. Đặc biệt do chúng được thực hiện bằng phần mềm nên các hệ thống điều khiển số cho phép thực hiện các luật điều khiển phức tạp. Kết quả là nâng cao được chất lượng điều khiển của toàn hệ thống.
- Dữ liệu trong các hệ thống điều khiển số được mã hoá nên việc trao đổi và truyền tin ít bị mất mát, có tính chống nhiễu cao.

Hệ thống điều khiển số cũng có những nhược điểm:

- Chỉ trao đổi thông tin ở thời điểm gián đoạn nên về nguyên tắc cho độ chính xác kém hơn so với hệ thống điều khiển tương tự.
- Phải có bộ biến đổi ADC , DAC làm phức tạp hệ thống.
- Có trễ ít nhất là một chu kỳ lấy mẫu.

Thực tế các DCS, đối tượng điều khiển của ta thường là một quá trình kỹ thuật có tính chất liên tục, còn phần điều khiển xử lý tín hiệu là phần tử xung số. Do đó sơ đồ khối của một DCS chung nhất như sau:



Do trong một DCS có cả phần liên tục và số, nên phương pháp nghiên cứu của ta là số hoá phần liên tục trong hệ thống để được một hệ thống số thuần tuý (chỉ số hoá phần liên tục để nghiên cứu, hệ thống thực vẫn gồm hai phần: liên tục và số).

1.2.1. Số hoá các tín hiệu:

Số hoá các tín hiệu là bước đầu tiên cần phải thực hiện trong một hệ thống điều khiển số. Việc số hoá tín hiệu được thực hiện trước hết bởi việc lấy mẫu trong khoảng thời gian T, sau đó tín hiệu lấy mẫu này được mã hoá thành dạng số nhờ các mạch chuyển đổi ADC.

Để có thể số hoá phần liên tục, ta phải qua hai bước là: rời rạc hoá và biến đổi sang dạng số. Để không mắc phải sai số trong quá trình số hoá ta phải tuân theo định lý lấy mẫu của Shannon.

* Định lý lấy mẫu Shannon:

Giả sử có tín hiệu liên tục x(t), qua quá trình rời rạc hoá ta được tín hiệu x(kT), với T là chu kỳ lấy mẫu:

x(kT) = x(t)*s(t); với

$$s(t) = \begin{cases} \sum_{k=-\infty}^{+\infty} \delta(t - kT) & \text{, khi } t > 0 \\ 0 & \text{, khi } t < 0 \end{cases}$$

Tín hiệu x(t) có biến đổi Fourier là X(jω)

Tín hiệu x(kT) có biến đổi Fourier là $X_a(j\omega)$

Để có thể tính được X(jω) từ $X_a(j\omega)$, có nghĩa là các mẫu x(kT) đặc trưng hoàn toàn cho tín hiệu x(t) ta phải tuân theo định lý lấy mẫu Shannon:

Định lý: Nếu phổ X(jω) của tín hiệu x(t) đồng nhất bằng không ngoài miền giới nội $|\omega| \geq \frac{2\Pi}{T_a}$ thì với chu kỳ lấy mẫu $T_a \leq \frac{2\Pi}{\omega}$ ảnh X(jω) của x(t) sẽ được suy ra từ $X_a(j\omega)$ của x(kT) bằng cách lấy phổ $X_a(j\omega)$ trong một chu kỳ.

1.2.2. Biến đổi Z:

Trong các hệ tuyến tính liên tục, phép biến đổi Laplace giữ vai trò quan trọng. Trong hệ thống số phép biến đổi Z đóng vai trò cũng có chức năng tương tự.

Nếu ta có tín hiệu liên tục x(t) thì tín hiệu rời rạc x(iT), với T là chu kỳ lấy mẫu, sẽ là:

$$x(iT) = \sum_{i=1}^{n} x(t) * \delta(i - iT);$$

Biến đổi Laplace của tín hiệu liên tục x(t) là:

$$X(p) = \int_0^{\infty} x(t) e^{-pt} dt \text{ ; với } p = \alpha + j\omega$$

Biến đổi Z của tín hiệu rời rạc rạc x(iT) là:

$$X(Z) = \sum_{i=0}^{\infty} x(iT) * Z^{-i};$$

Như vậy X(Z) chỉ phụ thuộc vào các giá trị rời rạc x(iT), tức là các giá trị x(t) tại các thời điểm t=iT. Chu kỳ lấy mẫu T phải đảm bảo thông tin nhận được là tin cậy và phải đảm bảo cho quá trình khôi phục tín hiệu.

1.2.3. Máy tính trong hệ thống điều khiển số:

Trong các hệ thống điều khiển số máy tính - hệ vi xử lý là một bộ phận không thể thiếu được. Nó đóng vai trò vô cùng to lớn trong việc thu thập, xử lý thông tin, lưu trữ, hiển thị và điều khiển đồng thời còn có chức năng giám sát, cảnh báo, báo động khi hệ thống có sự cố.

Việc ứng dụng kỹ thuật điều khiển số làm tăng độ bền vững, độ tin cậy, độ mềm dẻo và tốc độ điều khiển cao đồng thời chống nhiễu tốt. Nó còn cho phép xây dựng các phương án mềm, linh hoạt môdul hoá các khối tạo điều kiện thuận lợi trong thiết kế hệ thống, dễ dàng thay thế thiết bị và chương trình khi cần thiết do đó việc trao đổi thông tin giữa người và máy là đơn giản và có thể điều khiển linh hoạt hơn.

Các thiết bị và quá trình công nghệ có thể sử dụng vi xử lý (micro-processor) và vi điều khiển (micro-controller) để thực hiện quá trình điều khiển nhỏ, gọn, đơn giản cho người theo dõi hoạt động của hệ thống.

Tóm lại việc ứng dụng máy vi tính trong hệ thống điều khiển đã đáp ứng và giải quyết các vấn đề phức tạp trong lĩnh vực điều khiển.

1.3. Lò điện

1.3.1.Cấu tạo lò điện:

Lò điện là thiết bị dùng để biến đổi điện năng thành nhiệt năng trong quá trình gia nhiệt. Lò điện có nguồn nhiệt là những điện trở nung hoặc thanh nung toả nhiệt theo định luật Jun - lenxơ. Dòng chạy qua sẽ làm nóng thanh đốt và phát ra nhiệt lượng:

$$Q = C.R.I^2.t \text{ (kJ)}$$

Trong đó:

C : là hệ số tỉ lệ.

I : là cường độ dòng điện chạy qua thanh đốt (A)

R : là điện trở của dây nung (Ω)

Q : là nhệt lượng toả ra (J)

t : là thời gian dòng điện chạy qua dây nung (s)

1.3.2. Một số đặc điểm của lò điện:

- Quán tính nhiệt của lò lớn, sự thay đổi nhiệt độ của lò xảy ra chậm. Lò có hệ số dung lượng lớn thì độ trễ càng lớn.
- Nhiệt độ của buồng lò không hoàn toàn đồng đều và cặp nhiệt cũng có quán tính nhất định nên việc xác định nhiệt độ còn phụ thuộc vào vị trí đặt bộ cảm biến nhiệt độ.
- Biến thiên nhiệt độ lò có tính chất tự cân bằng. Nhờ tính chất này khi mất cân bằng giữa lượng nhiệt cung cấp và lượng nhiệt tiêu thụ thì nhiệt độ lò có thể tiến tới một giá trị xác lập mới mà không cần có sự tham gia của máy điều chỉnh.
- Các thanh nung cần thoả mãn một số yêu cầu sau: chịu được nhiệt độ cao, độ bền cơ học lớn, có điện trở suất nhỏ.

1.3.3. Các phương pháp điều khiển nhiệt độ lò điện:

Lò điện và vật cần nung là đối tượng điều khiển của hệ thống với đại lượng cần điều chỉnh là nhiệt độ vật cần nung. Việc điều chỉnh nhiệt độ của vật cần nung

cũng chính là điều khiển nhiệt độ trong buồng lò hay điều khiển công suất đặt vào lò.

$$P = I^2.R.t$$

Có hai phương án để xây dựng công suất này là:

- Điều chỉnh về phía tiêu thụ tức là thay đổi điện trở của lò. Phương pháp này ít được sử dụng bởi tính không liên tục và hạn chế về phạm vi điều khiển.
- Điều chỉnh về phía cung cấp tức là thay đổi cường độ dòng điện chạy qua thanh nung. Điều này có thể thực hiện được bằng biến áp, rơle hoặc thiristor.

(a). Phương pháp dùng biến áp:

Đây là phương pháp điều chỉnh điện áp theo cấp, nó đòi hỏi điện áp phải có công suất lớn. Phương pháp này dùng điện áp để thay đổi điện áp cung cấp cho lò.

(b). Phương pháp dùng rơle:

Phương pháp này có đặc điểm là có thể khống chế nhiệt độ của lò ở những mức điện áp khác nhau nhưng do rơle chỉ có tác động điều chỉnh ở các thời điểm ngưỡng nhất định nên việc điều chỉnh mang tính chất không liên tục. Mặt khác quá trình điều khiển luôn bị dao động, biên độ dao động phụ thuộc vào các điểm đặt khác nhau, vì thế độ chính xác điều chỉnh không cao, rơle phải đóng ngắt nhiều lần nên độ tin cậy kém. Tuy nhiên, phương pháp này có ưu điểm là đơn giản, dễ ghép nối, phù hợp với các yêu cầu công nghệ đòi hỏi độ chính xác không cao.

(c). Phương pháp dùng rơle kết hợp với thyristor:

Khi sử dụng phương pháp này thì khả năng điều chỉnh với các phạm vi khác nhau là tương đối tốt. Tuy nhiên phương pháp này không thực hiện điều chỉnh liên tục được bởi vì khi tiếp điểm của rơle đóng ta luôn có cả chu kỳ cung cấp cho tải, khi mở nguồn cung cấp phía diod bị ngắt do đó việc cung cấp cho lò là do thyristor như vậy công suất đưa vào lò chỉ điều khiển được 1/2 chu kỳ.

(d). Phương pháp dùng hai thyristor mắc xung đối:

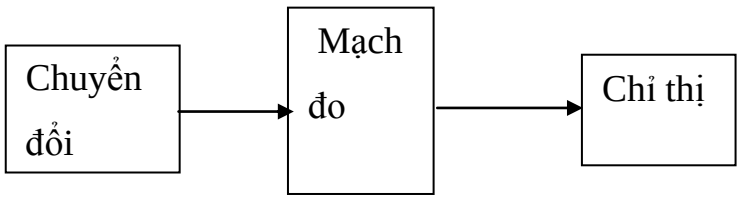
Phương pháp này cho phép điều chỉnh trong phạm vi rộng, độ chính xác điều khiển tương đối cao, độ nhạy điều chỉnh tương đối lớn, có khả năng điều chỉnh tương đối liên tục và đều đặn. Tuy nhiên, hệ thống tương đối phức tạp và giá thành cao.

1.3.4. Đo nhiệt độ

1.3.4.1. Giới thiệu

Để thực hiện phép đo của một đại lượng nào đó thì tuỳ thuộc vào đặc tính của đại lượng cần đo,điều kiện đo,cũng như độ chính xác theo yêu cầu của một phép đo mà ta có thể thực hiện đo bằng nhiều cách khác nhau trên cơ sỡ của các hệ thống đo lường khác nhau.

Sơ đồ khối của một hệ thống đo lường tổng quát



_ Khối chuyển đổi: làm nhiệm vụ nhận trực tiếp các đại lượng vật lý đặc trưng cho đối tượng cần đo biến đổi các đại lượng thành các đại lượng vật lý thống nhất(dòng điện hay điện áp) để thuận lợi cho việc tính toán.

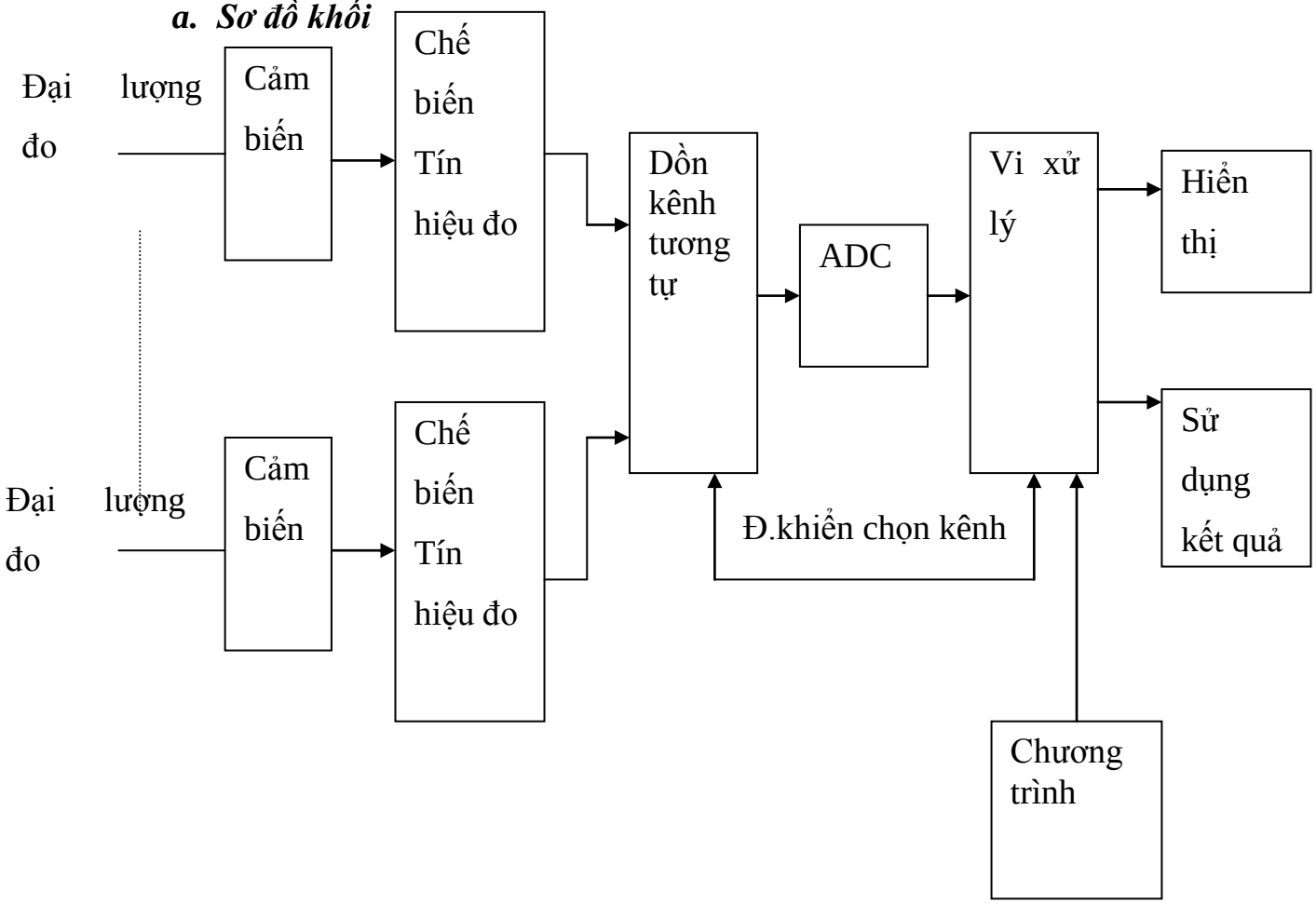
_ Mạch đo: có nhiệm vụ tính toán biến đổi tín hiệu nhận được từ bộ chuyển đổi sao cho phù hợp với yêu cầu thể hiện kết quả đo của bộ chỉ thị.

_ Khối chỉ thị:làm nhiệm vụ biến đổi tín hiệu điện nhận được từ mạch đo để thể hiện kết quả đo.

1.3.4.2. Hệ thống đo lường số

Hệ thống đo lường số được em áp dụng để thực hiện bản đồ án này vì có các ưu điểm:các tín hiệu tương tự qua biến đổi thành các tín hiệu số có các xung rỏ ràng ở trạng thái 0,1 sẽ giới hạn được nhiều mức tín hiệu gây sai số .Mặt khác ,hệ thống này tương thích với dữ liệu của máy tính,qua giao tiếp với máy tính ứng dụng rộng rãi trong kỹ thuật.

a. Sơ đồ khối



Sơ đồ khối của hệ thống đo lường số

b. Nguyên lý hoạt động

Đối tượng cần đo là đại lượng vật lý,dựa vào các đặc tính của đối tượng cần đo mà ta chọn một loại cảm biến phù hợp để biến đổi thông số đại lượng vật lý cần đo thành đại lượng điện ,đưa vào mạch chế biến tín hiệu(gồm:bộ cảm biến,hệ thống khuếch đại,xử lý tín hiệu).

Bộ chuyển đổi tín hiệu sang số ADC(Analog Digital Converter) làm nhiệm vụ chuyển đổi tín hiệu tương tự sang tín hiệu số và kết nối với vi xử lý.

Bộ vi xử lý có nhiệm vụ thực hiện những phép tính và xuất ra những lệnh trên cơ sở trình tự những lệnh chấp hành đã thực hiện trước đó.

Bộ dồn kênh tương tự (multiplexers) và bộ chuyển ADC được dùng chung tất cả các kênh . Dữ liệu nhập vào vi xử lý sẽ có tín hiệu chọn đúng kênh cần xử lý đê đưa vào bộ chuyển đổi ADC và đọc đúng giá trị đặc trưng của nó qua tính toán để có kết quả của đại lượng cần đo.

1.3.4.3. Các phương pháp đo nhiệt độ

Đo nhiệt độ là một phương thức đo lường không điện,đo nhiệt độ được chia thành nhiều dải:

+ Đo nhiệt độ thấp
+ Đo nhiệt độ trung bình
+ Đo nhiệt độ cao.

Việc đo nhiệt độ được tiến hành nhờ các dụng cụ hổ trợ chuyên biệt như:

+ Cặp nhiệt điện
+ Nhiệt kế điện kế kim loại
+ Nhiệt điện trở kim loại
+ Nhiệt điện trở bán dẫn
+ Cảm biến thạch anh.

Việc sử dụng các IC cảm biến nhiệt để đo nhiệt độ là một phương pháp thông dụng được em sử dụng trong bản đồ án này,nên ở đây chỉ giới thiệu về IC cảm biến nhiệt.

 Nguyên lý hoạt động chung của IC đo nhiệt độ

IC đo nhiệt độ là một mạch tích hợp nhận tín hiệu nhiệt độ chuyển thành tín hiệu điện dưới dạng dòng điện hay điện áp.Dựa vào đặc tính rất nhạy của các bán dẫn với nhiệt độ,tạo ra điện áp hoặc dòng điện,tỉ lệ thuận với nhiệt độ tuyệt đối.Đo tín hiệu điện ta biết được giá trị của nhiệt độ cần đo.Sự tác động của nhiệt độ tạo ra điện tích tự do và các lổ trống trong chất bán dẫn . Bằng sự phá vỡ các phân tư , bứt các electron thành dạng tự do di chuyển qua vùng cấu trúc mạng tinh thể tạo sự xuất hiện các lỗ trống . Làm cho tỉ lệ điện tử tự do và lổ trống tăng lên theo qui luật hàm mũ với nhiệt độ .

 Đặc tính của một số IC đo nhiệt độ thông dụng

+AD590

Ngõ ra là dòng điện.

Độ nhạy $1A/^0K$.

Độ chính xác $+4^0C$.

Nguồn cung cấp $V_{cc} = 4 – 30V$.

Phạm vi sử dụng –55^{o}c đến 150^{o}c

+ LX5700

Ngõ ra là điện áp.

Độ nhạy –10mv/^{0}K.

Phạm vi sử dụng –55^{0}C – 150^{0}C.

+ LM135,LM335

Ngõ ra là điện áp.

Độ nhạy 10mv/^{0}C.

Sai số cực đại 1,5^{0}C khi nhiệt độ lớn hơn 100^{0}C.

Phạm vi sử dụng –55^{0}C – 150^{0}C.

CHƯƠNG 2:
THIẾT KẾ PHẦN CỨNG HỆ THỐNG ĐIỀU KHIỂN SỐ

2.1. Thiết kế mạch ghép nối với PC

2.1.1. Sơ đồ khối ghép nối

Thiết kế mạch ghép nối bao gồm cả việc thiết kế tổ chức phần cứng và viết phần mềm cho nền phần cứng mà ta thiết kế. Việc xem xét giữa tổ chức phần cứng và chương trình phần mềm cho một thiết kế là một vấn đề cần phải cân nhắc. Vì khi tổ chức phần cứng càng phức tạp, càng có nhiều chức năng hỗ trợ cho yêu cầu thiết kế thì phần mềm càng được giảm bớt và dễ dàng thực hiện nhưng lại đẩy cao giá thành chi phí cho phần cứng, cũng như chi phí bảo trì. Ngược lại với một phần cứng tối thiểu lại yêu cầu một chương trình phần mềm phức tạp hơn, hoàn thiện hơn; nhưng lại cho phép bảo trì hệ thống dễ dàng hơn cũng như việc phát triển tính năng của hệ thống từ đó có thể đưa ra giá cạnh tranh được.

Từ nhiệm vụ bài toán đặt ra ta xây dựng sơ đồ khối ghép nối:

(***a). Bộ điều khiển mềm:***

Bộ điều khiển mềm được đặt trên máy tính cá nhân. Việc giao tiếp được tiến hành qua cổng nối tiếp RS 232 bằng vi điều khiển 8051 và chíp giao tiếp MAX 232.

Ta dùng chip vi điều khiển 8051 vì những lý do sau:

- Thứ nhất 8051 thuộc họ MCS - 51, là chip vi điều khiển. Đặc điểm của các chip vi điều khiển nói chung là nó được tích hợp với đầy đủ chức năng của một hệ VXL nhỏ, rất thích hợp với những thiết kế hướng điều khiển. Tức là trong nó bao gồm: mạch VXL, bộ nhớ chương trình và dữ liệu, bộ đếm, bộ tạo xung, các cổng vào/ra nối tiếp và song song, mạch điều khiển ngắt…

- Thứ hai là, vi điều khiển 8051 cùng với các họ vi điều khiển khác nói chung trong những năm gần đây được phát triển theo các hướng sau:

+ Giảm nhỏ dòng tiêu thụ.

+ Tăng tốc độ làm việc hay tần số xung nhịp của CPU .

+ Giảm điệp áp nguồn nuôi.

+ Có thể mở rộng nhiều chức năng trên chip, mở rộng cho các thiết kế lớn.

Những đặc điểm đó dẫn đến đạt được hai tính năng quan trọng là: giảm công suất tiêu thụ và cho phép điều khiển thời gian thực nên về mặt ứng dụng nó rất thích hợp với các thiết kế hướng điều khiển.

- Thứ ba là, vi điều khiển thuộc họ MCS - 51 được hỗ trợ một tập lệnh phong phú nên cho phép nhiều khả năng mềm dẻo trong vấn đề viết chương trình phần mềm điều khiển.

- Cuối cùng là, các chip thuộc họ MCS - 51 hiện được sử dụng phổ biến và được coi là chuẩn công nghiệp cho các thiết kế khả dụng. Mặt khác, qua việc khảo sát thị trường linh kiện việc có được chip 8051 là dễ dàng nên mở ra khả năng thiết kế thực tế.

Vì những lý do trên mà việc lựa chọn vi điều khiển 8051 là một giải pháp hoàn toàn phù hợp cho thiết kế.

(b). Tổ chức ngoại vi:

Xử lý tín hiệu vào ta dùng thiết bị chuyển đổi tương tự - số (ADC) có 8 kênh vào tương tự kết nối với tín hiệu đo nhiệt độ từ 0 ÷ 5V tương ứng với nhiệt độ từ $0^o \div 1300^oC$.

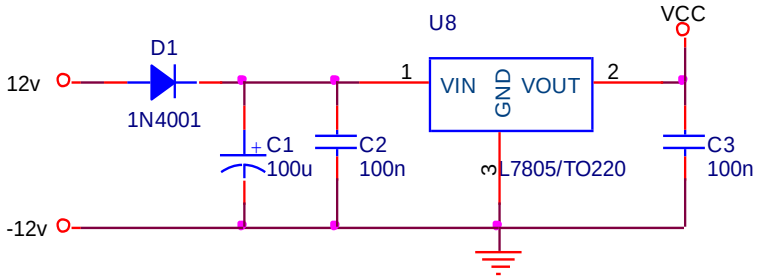
Để phát tín hiệu điều khiển ra ta dùng DAC 0808 đưa đến một bộ khuyếch đại công suất để đưa dòng điện đến thanh trở của lò, điều khiển thông qua thyristor.

Để giao tiếp với PC ta sử dụng IC Max232 cùng cổng DB9 làm giao diện.

Tất cả các thiết bị phải được kết nối với nhau thông qua các bus cần thiết gồm bus dữ liệu, bus địa chỉ và bus điều khiển.

(c). ***Các khối chức năng:***

***.** Khối nguồn:



Khối nguồn gồm có một IC 7805, tụ C1=100 uF và C2=100 nF mắc ở ngõ vào và một tụ C3=100 nF mắc ở ngõ ra nhằm mục đích ổn định

***.** Khối vi điều khiển 8051:

VXL8051 đóng vai trò trung tâm trong hệ thống. Nó có nhiệm vụ tương tác với các khối như việc nhận, xử lý và chuyển dữ liệu tới các khối. 8051 đảm nhận tất cả các nhiệm vụ từ việc nhận dữ liệu vào dưới dạng tín hiệu nhị phân ở cổng vào, xử lý dữ liệu, chuyển dữ liệu thành mã Hexa, lưu giữ dữ liệu và điều khiển hoạt động của khối hiển thị và tạo tín hiệu điều khiển.

***.** Khối các thiết bị giao tiếp/ghép nối.

Cổng vào ra tương tự/số dùng ADC 0809. Số liệu vào tương tự từ cảm biến nhiệt độ và từ biến trở để tạo tín hiệu setpoint sẽ được kết nối vào cổng vào của ADC, ADC được điều khiển bởi VDDK 8051 thực hiện việc chuyển đổi số liệu sang dạng số và lưu trữ vào một vùng nào đó trong RAM trong. DAC 0808 sẽ dùng để phát tín hiệu điều khiển từ VXL ra lò

2.1.2 Giới thiệu các thiết bị và tổ chức phối ghép

2.1.2.1. Bộ vi điều khiển 8051:

Những tính chất đặc trưng của họ vi điều khiển MCS - 51:

* Đơn vị xử lý trung tâm (CPU) 8 bit đã được tối ưu hoá để đáp ứng các chức năng điều khiển.

* Khối lôgic (ALU) xử lý theo bit nên thuận tiện cho các phép toán logic Boolean.

* Bộ tạo dao động giữ nhịp được tích hợp bên trong với tần số 12 MHz.

* Giao diện nối tiếp có khả năng hoạt động song song, đồng bộ.

* Các cổng vào - ra hai hướng và từng đường dẫn có thể được định địa chỉ một cách tách biệt.

* Có năm hay sáu nguồn ngắt với hai mức ưu tiên .

* Hai hoặc ba bộ đếm định thời 16 bit.

* Bus và khối định thời tương thích với các khối ngoại vi của bộ vi xử lý 8085/8088.

* Dung lượng của bộ nhớ chương trình (ROM) bên ngoài có thể lên tới 64 KByte.

* Dung lượng của bộ nhớ dữ liệu (RAM) bên ngoài có thể lên tới 64 kbyte.

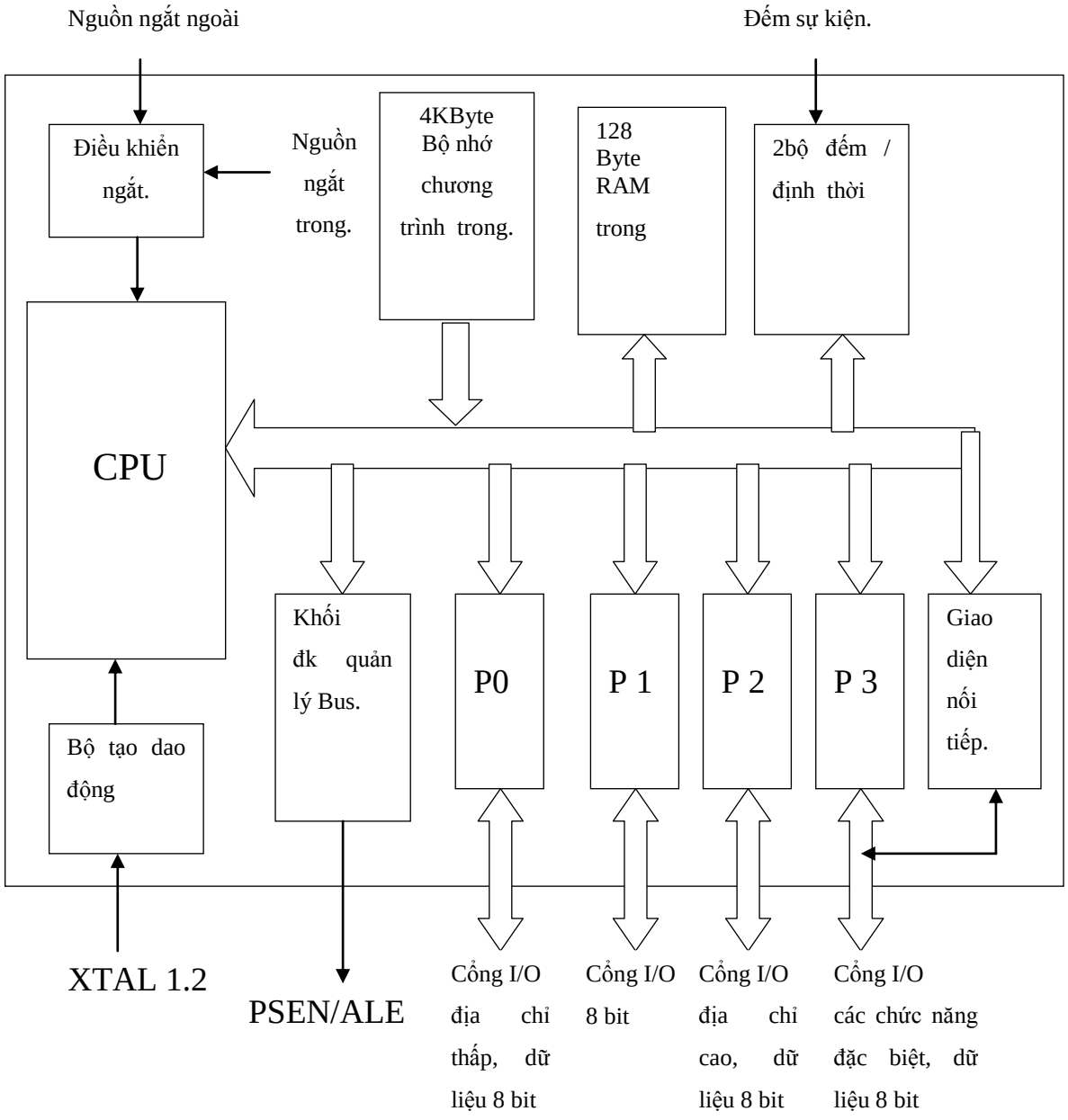
* Dung lượng của bộ nhớ ROM bên trong có thể lên đến 8 KByte.

* Dung lượng bộ nhớ RAM bên trong có thể đạt đến 256 byte.

* Tập lệnh phong phú.

(a). Sơ đồ khối:

Sơ đồ khối tổng quát của một vi điều khiển 8051 có thể được mô tả như sau:



Cấu trúc của vi điều khiển 8051.

Chức năng của từng khối :

* Khối xử lý trung tâm CPU: Phần chính của bộ vi xử lý là khối xử lý trung tâm (CPU = Central Processing Unit), khối này có chứa các thành phần chính:

- Thanh ghi tích luỹ (ký hiệu là A)

- Thanh ghi tích luỹ phụ (ký hiệu là B) thường được dùng cho phép nhân và phép chia

- Khối logic số học (ALU = Arithmetic Logical Unit)

- Từ trạng thái chương trình (PSW = Program Status Word)

- 4 bank thanh ghi

- Con trỏ ngăn xếp (SP = Stack Point) cũng như con trỏ dữ liệu để định địa chỉ cho bộ nhớ dữ liệu ở bên ngoài

Ngoài ra, khối xử lý trung tâm còn chứa:

- Thanh ghi đếm chương trình (PC = Progam Counter)

- Bộ giải mã lệnh

- Bộ điều khiển thời gian và logic

Sau khi được Reset, CPU bắt đầu làm việc tại địa chỉ 0000h, là địa chỉ đầu được ghi trong thanh ghi chứa chương trình (PC) và sau đó, thanh ghi này sẽ tăng lên 1 đơn vị và chỉ đến các lệnh tiếp theo của chương trình.

* Bộ tạo dao động:

Khối xử lý trung tâm nhận trực tiếp xung nhịp từ bộ tạo dao động được lắp thêm vào, linh kiện phụ trợ có thể là một khung dao động làm bằng tụ gốm hoặc thạch anh. Ngoài ra, còn có thể đưa một tín hiệu giữ nhịp từ bên ngoài vào.

* Khối điều khiển ngắt:

Chương trình đang chạy có thể cho dừng lại nhờ một khối logic ngắt ở bên trong. Các nguồn ngắt có thể là: các biến cố ở bên ngoài, sự tràn bộ đếm/bộ định thời hay có thể là giao diện nối tiếp. Tất cả các ngắt đều có thể được thiết lập chế độ làm việc thông qua hai thanh ghi IE (Interrupt Enable) và IP (Interrupt Priority).

* Khối điều khiển và quản lý Bus:

Các khối trong vi điều khiển liên lạc với nhau thông qua hệ thống Bus nội bộ được điều khiển bởi khối điều khiển quản lý Bus.

* Các bộ đếm/định thời:

Vi điều khiển 8051 có chứa hai bộ đếm tiến 16 bit có thể hoạt động như là bộ định thời hay bộ đếm sự kiện bên ngoài hoặc như bộ phát tốc độ Baud dùng cho

giao diện nối tiếp. Trạng thái tràn bộ đếm có thể được kiểm tra trực tiếp hoặc được xoá đi bằng một ngắt.

* Các cổng vào/ra:

Vi điều khiển 8051 có bốn cổng vào/ra (P0 ÷ P3), mỗi cổng chứa 8 bit, độc lập với nhau. Các cổng này có thể được sử dụng cho những mục đích điều khiển rất đa dạng. Ngoài chức năng chung, một số cổng còn đảm nhận thêm một số chức năng đặc biệt khác.

* Giao diện nối tiếp:

Giao diện nối tiếp có chứa một bộ truyền và một bộ nhận không đồng bộ làm việc độc lập với nhau. Bằng cách đấu nối các bộ đệm thích hợp, ta có thể hình thành một cổng nối tiếp RS 232 đơn giản. Tốc độ truyền qua cổng nối tiếp có thể đặt được trong một vùng rộng phụ thuộc vào một bộ định thời và tần số dao động riêng của thạch anh.

* Bộ nhớ chương trình:

Bộ nhớ chương trình thường là bộ nhớ ROM (Read Only Memory), bộ nhớ chương trình được sử dụng để cất giữ chương trình điều khiển hoạt động của vi điều khiển.

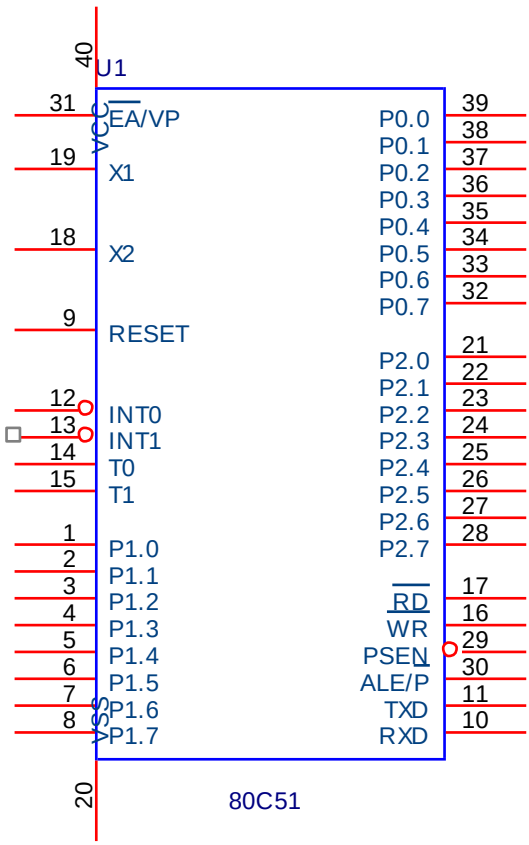
* Bộ nhớ số liệu:

Bộ nhớ số liệu thường là bộ nhớ RAM (Random Access Memory), bộ nhớ số liệu dùng để cất giữ các thông tin tạm thời trong quá trình vi điều khiển làm việc.

(b). Sự sắp xếp chân ra của vi điều khiển 8051:

Phần lớn các bộ vi điều khiển 8051 được đóng vào vỏ theo kiểu hai hàng DIL (Dual In Line) với tổng số là 40 chân ra, một số ít còn lại được đóng vỏ theo kiểu hình vuông PLCC (Plastic Leaded Chip Carrier) với 44 chân và loại này thường được dùng cho những hệ thống cần thiết phải tiết kiệm diện tích.

Sơ đồ chân bộ vi điều khiển 8051:



Sơ đồ chân của IC 80C51

Bảng chức năng các chân của vi điều khiển 8051.

Chân	Ký hiệu	Chức năng
1 ÷ 8	P1.0 ÷ P1.7	Cổng giả hai hướng P1, có thể tự do sử dụng
9	Reset	Lối vào Reset, khi hoạt động ở mức High(1)
10 ÷ 17	P3.0 ÷ P3.7	Cổng giả hai hướng P3, sắp xếp tất cả các đường dẫn với chức năng đặc biệt
18	XTAL2	Lối ra của bộ dao động thạch anh bên trong
19	XTAL1	Lối vào của bộ dao động thạch anh bên trong
20	Vss	Nối mát (0V)
21 ÷ 28	P2.0 ÷ P2.7	Cổng giả hai hướng P2, chức năng đặc biệt là các đường dẫn địa chỉ A8 ÷ A15
29	PSEN	Progam Strobe Enable, xuất ra các xung đọc dùng cho bộ nhớ chương trình bên ngoài
30	ALE	Address Latch Enable, xuất ra các xung điều khiển để lưu trữ trung gian các địa chỉ
31	EA	External Access, khi được nối với mát là để

		làm việc với ROM ngoại vi
32 ÷ 39	P1.0 ÷ P1.7	Cổng hai hướng cực máng hở P0 hay Bus dữ liệu hai hướng dùng cho ROM, RAM và thiết bị ngoại vi đồng thời cũng chuyển giao 8 bit địa chỉ thấp
40	Vdd	Nguồn nuôi dương (+5V)

* *Chức năng các chân vi điều khiển:*

Port 0: là Port có 2 chức năng ở trên chân từ 32 đến 39 trong các thiết kế cỡ nhỏ (không dùng bộ nhớ mở rộng), có hai chức năng như các đường I/O. Đối với các thiết kế cỡ lớn (với bộ nhớ mở rộng) nó được kết hợp kênh giữa các bus.

Port 1: là một port I/O trên các chân 1-8. Các chân được ký hiệu P1.0, P1.1, P1.2,..,P1.8 có thể dùng cho các thiết bị ngoài nếu cần. Port1 không có chức năng khác, vì vậy chúng ta chỉ được dùng trong giao tiếp với các thiết bị ngoài.

Port 2: là một Port công dụng kép trên các chân 21 - 28 được dùng như các đường xuất nhập hoặc là Byte cao của bus địa chỉ đối với các thiết kế dùng bộ nhớ mở rộng.

Port 3: là một Port công dụng kép trên các chân 10 -17.

PSEN (*Program Store Enable*): 8051 / 8031 có 1 tín hiệu điều khiển PSEN là tín hiệu ra trên chân 29. Nó là tín hiệu điều khiển để cho phép bộ nhớ chương trình mở rộng và thường được nối đến chân OE (*Output Enable*) của một EPROM để cho phép đọc các Byte mã lệnh. PSEN sẽ ở mức thấp trong thời gian lấy lệnh. Các mã nhị phân của chương trình được đọc từ EPROM qua bus và được chốt vào thanh ghi lệnh của 8051 để giải mã lệnh. Khi thi hành chương trình trong ROM nội (8051) PSEN sẽ ở mức thụ động (mức cao).

ALE (*Address Latch Enable*): Tín hiệu ra ALE trên chân 30 tương hợp với các thiết bị làm việc với các xử lý 8585, 8088, 8086, 8051 dùng ALE một cách tương tự cho làm việc giải các kênh các bus địa chỉ và dữ liệu khi Port 0 được dùng trong chế độ chuyển đổi của nó: vừa là bus dữ liệu vừa là bit thấp của địa chỉ, ALE là tín hiệu để chốt địa chỉ vào một thanh ghi bên ngoài trong nữa đầu của chu kỳ bộ nhớ. Sau đó, các đường Port 0 dùng để xuất hoặc nhập dữ liệu trong nữa sau chu kỳ của bộ nhớ. Các xung tín hiệu ALE có tốc độ bằng 1/6 lần tần số dao động trên chip và có thể được dùng là nguồn xung nhịp cho các hệ thống. Nếu xung trên 8051 là 12 MHz thì ALE có tần số 2 MHz. Chỉ ngoại trừ khi thi hành lệnh MOVX, một xung ALE sẽ bị mất. Chân này cũng được làm ngõ vào cho xung lập trình cho EPROM trong 8051.

EA (*External Access*): Tín hiệu vào EA trên chân 31 thường được mắc lên mức cao (+5V) hoặc mức thấp (GND). Nếu ở mức cao, 8051 thi hành chương trình từ ROM nội trong khoảng địa chỉ thấp (4K). Nếu ở mức thấp, chương trình chỉ được thi hành từ bộ nhớ mở rộng. Khi dùng 8031, EA luôn được nối mức thấp vì không có bộ nhớ chương trình trên chip. Nếu EA được nối mức thấp bộ nhớ bên trong chương trình 8051 sẽ bị cấm và chương trình thi hành từ EPROM mở rộng.

Người ta còn dùng chân EA làm chân cấp điện áp 21V khi lập trình cho EPROM trong 8051.

RST (*Reset*): Ngõ vào RST trên chân 9 là ngõ reset của 8051. Khi tín hiệu này được đưa lên múc cao (trong ít nhất 2 chu kỳ máy), các thanh ghi trong 8051 được tải những giá trị thích hợp để khởi động hệ thống.

Các ngõ vào bộ dao động trên chip: Như đã thấy trong các hình trên, 8051 có một bộ dao động trên chip. Nó thường được nối với thạch anh giữa hai chân 18 và 19. Các tụ giữa cũng cần thiết như đã vẽ. Tần số thạch anh thông thường là 12 MHz.

Các chân nguồn: 8051 vận hành với nguồn đơn +5V. Vcc được nối vào chân 40 và Vss (GND) được nối vào chân 20.

Các chân đối thoại với bộ nhớ ngoài :

WR (*Write*) - đối thoại để viết vào bộ nhớ dữ liệu ở ngoài.

RD (*Read*) - đối thoại để đọc từ bộ nhớ dữ liệu ở ngoài.

EA (*External Address*) - tín hiệu chọn 4K đầu tiên của ROM :

EA = 0 : chọn 4K ROM ngoài.

EA = 1 : chọn 4K ROM trong.

ALE (*Address Latch Enable*) - cho phép chốt địa chỉ trên cổng đa hợp P0

PSEN (*Program Store Enable*) - cho phép đọc bộ nhớ chương trình ngoài.

* *Các thanh ghi đặc biệt bên trong chip:*

A (Accumulator): Thanh ghi đa chức năng.

B : Như thanh ghi A, ngoài ra còn dùng trong các lệnh Mul, Div.

PSW (*Program Status Word*): Thanh ghi từ điều khiển

SP (*Stack Pointer*): Con trỏ Stack. Sau khi Reset có giá trị là 07h. SP được tăng trước khi đữ liệu được cất vào Stack.

DPTR (*Data Pointer* (*DPH, DPL*)): Con trỏ chứa địa chỉ 16-bit dùng trong một số lệnh truy nhập bộ nhớ.

P0, P1, P2, P3 (*Port Latches*): Các bộ chốt cho 4 cổng vào ra tương ứng.

SBUF (*Serial Data Buffer*): Thanh ghi để đọc và viết cho cổng nối tiếp.

SCON (*Serial Port Control*): Thanh ghi điều khiển cổng nối tiếp.

TMOD (*Timer Mode*): Thanh ghi chế độ cho các Timer.

TCON (*Timer Control*): Thanh ghi điều khiển cho các Timer.

T2CON (*8052 Timer 2 Control*): Thanh ghi điều khiển cho Timer 2 của 8052.

PCON (*Power Control*): Thanh ghi điều khiển nguồn (chỉ sử dụng cho 89C51).

IE (*Interrupt Enable*): Thanh ghi cho phép ngắt (1=cho phép; 0=không cho phép).

IP (*Interrupt Priority*): Thanh ghi ưu tiên các ngắt.

(c). Tổ chức phần cứng Vi điều khiển 8051:

c.1. Tổ chức bộ nhớ (*Memory Map*):

Từ cấu trúc của vi điều khiển 8051 đã giới thiệu và yêu cầu thiết kế ta tiến hành phân bổ các vùng nhớ như sau:

- Bộ nhớ chương trình 8K ROM chia làm hai vùng:

ROM trong (*On-chip*) có địa chỉ vật lý: 0000H ÷ 0FFFH.

ROM ngoài (2764) có địa chỉ vật lý: 1000H ÷ 2FFFH.

- Bộ nhớ dữ liệu được mở rộng thêm 8K RAM ngoài, với địa chỉ vật lý: 2000H ÷ 3FFFH.

- Mạch ghép nối vào/ ra sử dụng IC8255 với địa chỉ của từng cấu hình như sau:

Địa chỉ cổng PA: 4000H

Địa chỉ cổng PB: 4001H

Địa chỉ cổng PC: 4002H

Địa chỉ của từ điều khiển PSW: 4003H

- Địa chỉ của ADC 0809 8 kênh vào tương tự: 6000H ÷ 6007H.

- Địa chỉ của DAC 0808 là :8000H

c.2. Thiết kế bộ nhớ:

Xem xét cấu trúc của 8051 và để tạo khả năng mở rộng phạm vi điều khiển cho hệ điều khiển nếu có nhu cầu về sau này ta thiết kế thêm vùng bộ nhớ chương trình dùng thêm 8 KB ROM đặt ở ngoài. VĐK 8051 đã có 128 Byte cho bộ nhớ dữ liệu tuy nhiên đối với yêu cầu mở rộng cho nhớ dữ liệu đối với các ứng dụng sau này ta sử dụng thêm 8 KB Ram dữ liệu.

Nguyên tắc phối ghép bộ nhớ với VĐK:

- Nhóm tín hiệu địa chỉ phối ghép với Bus địa chỉ của hệ thống để chọn ra một ô nhớ cụ thể để đọc/ghi.

- Nhóm tín hiệu dữ liệu được phối ghép với Bus dữ liệu của hệ thống nhằm thực hiện được việc trao đổi dữ liệu trong hệ thống với bộ nhớ.

- Nhóm tín hiệu chọn vi mạch (*Chip Select*): được phối ghép với đầu ra của giải mã địa chỉ để có thể thực hiện được việc chọn ra một vùng nhớ làm việc.

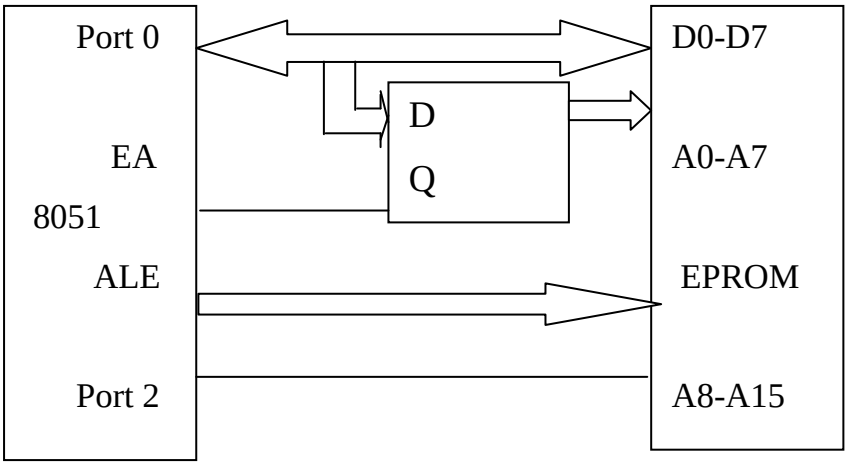
- Nhóm tín hiệu điều khiển: Kết nối với các Bus điều khiển của hệ VXL. Đối với ROM thì đầu vào điều khiển OE (*Output Enable*) để cho phép dữ liệu được đưa ra Bus thì được kết nối với dây tín hiệu RD của VXL. Đối với RAM có hai tín hiệu điều khiển thì tín hiệu điều khiển ghi WE (*Write Enable*) được nối với chân

tín hiệu WR của VXL, còn tín hiệu điều hiển đọc OE thì được nối với chân tín hiệu RD của VXL.

*** Bộ nhớ ROM ngoài:**

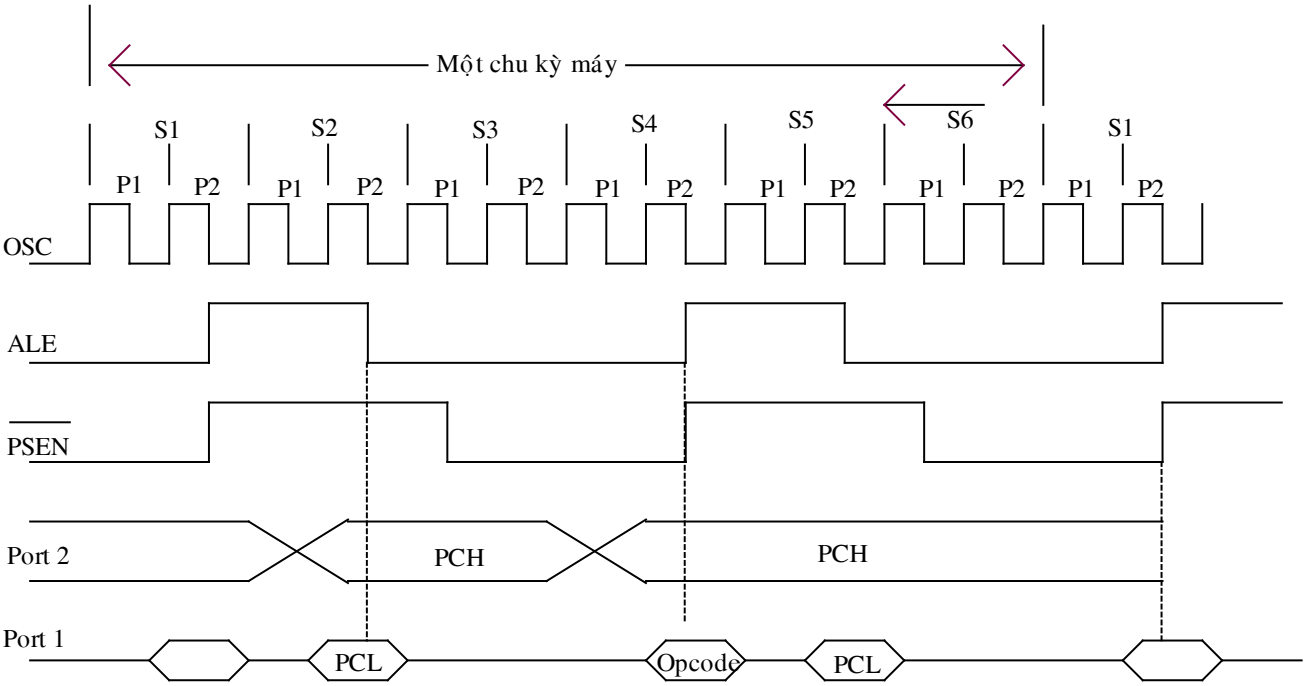
Đối với ROM ngoài ta dùng vi mạch nhớ chỉ đọc loại EPROM 2764. Đây là bộ nhớ lập trình xoá bằng tia cực tím, nó có tốc độ truy nhập rất nhanh. Với dung lượng 8K Byte như vậy nó có 13 đường chọn địa chỉ và có 8 đường ra dữ liệu. EPROM chỉ được hoạt động khi chân OE ở mức tích cực thấp, nó được vi điều khiển chọn làm việc khi chân CE cũng được tích cực thấp. EPROM được nuôi với mức điện áp 5V, điện áp này được đưa vào bộ nhớ thông qua chân Vpp. Địa chỉ của EPROM trong hệ thống là 1000H ÷ 2FFFH nên nó được chọn bởi tín hiệu chọn chip Y0 của giải mã địa chỉ.

Bộ nhớ chương trình ngoài là mộ IC ROM được phép bởi tín hiệu PSen. Hình sau mô tả cách nối một EPROM vào 8051/8031:



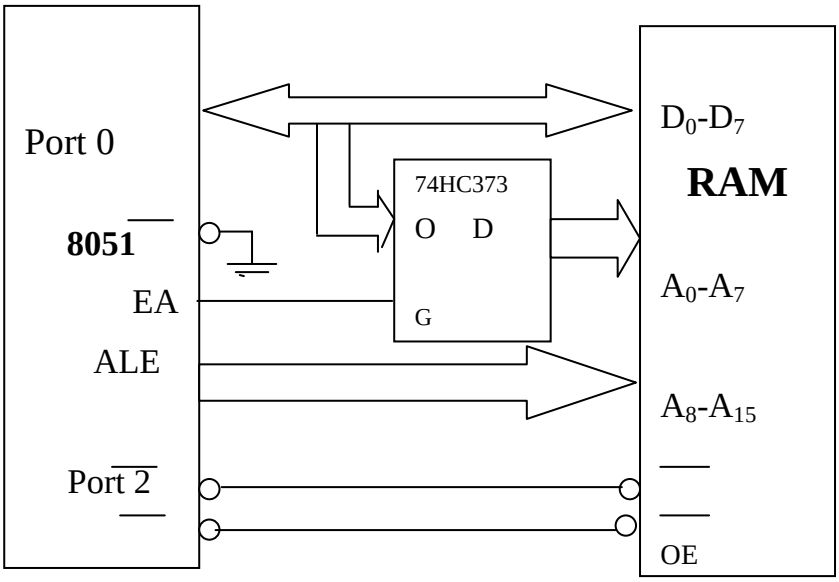
Giao tiếp giữa 8051/8031 và EPROM

Một chu kỳ máy của 8051/8031 có 12 chu kỳ xung nhịp. Nếu bộ dao động trên chip được lái bởi một thạch anh 12MHz thì chu kỳ máy kéo dài 1μs. Trong một chu kỳ máy sẽ có 2 xung ALE và 2 byte được đọc từ bộ nhớ chương trình (nếu lệnh hiện hành là một byte thì byte thứ hai sẽ được loại bỏ). Giản đồ thời gian của một lần lấy lệnh được vẽ ở hình sau:



Giản đồ thời gian đọc bộ nhớ chương trình ngoài.

Truy xuất bộ nhớ dữ liệu ngoài:

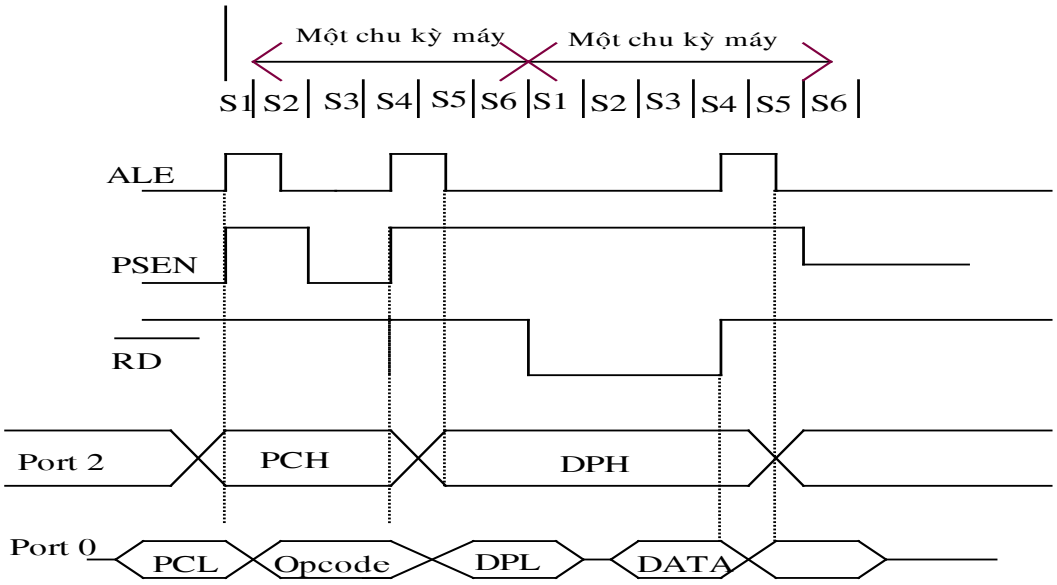


Giao tiếp giữa 8051/8031 và RAM

Bộ nhớ dữ liệu ngoài là một bộ nhớ RAM được cho phép ghi/đọc bằng các tín hệu WR và RD (các chân P3.6 và P3.7 thay đổi chức năng). chỉ có một cách truy xuất bộ nhớ dữ liệu ngoài là với lệnh MOVX dùng con trỏ dữ liệu (DPTR) 16 bit hoặc R0 và R1 xem như thanh ghi địa chỉ.

Kết nối bus địa chỉ và bus dữ liệu giữa RAM và 8051/8031 cũng giống EPROM và do đó cũng có thể lên đến 64 byte bộ nhớ RAM. Ngoài ra, chân RD của 8051/8031 được nối tới chân cho phép xuất (OE) của RAM và chân WR được nối tới chân ghi (WR) của RAM.

Giản đồ thời gian cho lệnh đọc bộ nhớ dữ liệu ngoài được vẽ trên hình sau đối với lệnh MOVX A, @DPTR:



Giản đồ thời gian của lệnh MOVX

Giản đồ thời gian cho lệnh ghi (MOVX @DPTR, A) cũng tương tự chỉ khác đường WR sẽ thay vào đường RD và dữ liệu được xuất ra trên port 0 (RD vẫn giữ mức cao).

2.1.2.2. Vi mạch ADC 0809:

Bộ ADC 0809 là một thiết bị CMOS tích hợp với một bộ chuyển đổi tương sang số 8 bit, bộ chọn kênh và một bộ logic điều khiển tương thích. Bộ chuyển đổi tương tự số này sử dụng phương pháp chuyển đổi xấp xỉ. Bộ chọn kênh có thể chọn ra kênh cần chuyển đổi bằng 3 chân chọn địa chỉ. Thiết bị này loại trừ khả năng cần thiết điều chỉnh điểm zero bên ngoài và khả năng điều chỉnh tỉ số làm cho ADC đễ dàng giao tiếp với các bộ vi xử lý.

** Các đặc điểm cơ bản của ADC 0809:*

- Nguồn nuôi đơn ± 5 V, hiệu suất cao.

- Dải tín hiệu lối vào tương tự 5V khi nguồn nuôi là +5V. Có thể mở rộng thang đo bằng các giải pháp kỹ thuật cho từng mạch cụ thể.

- Dễ dàng giao tiếp với vi xử lý vì đầu ra có bộ đệm 3 trạng thái nên có thể ghép trực tiếp vào kênh dữ liệu của hệ VXL.

- Tổng sai số chưa chỉnh ±1/2 LSB.

- Thời gian chuyển đổi 100 μs .

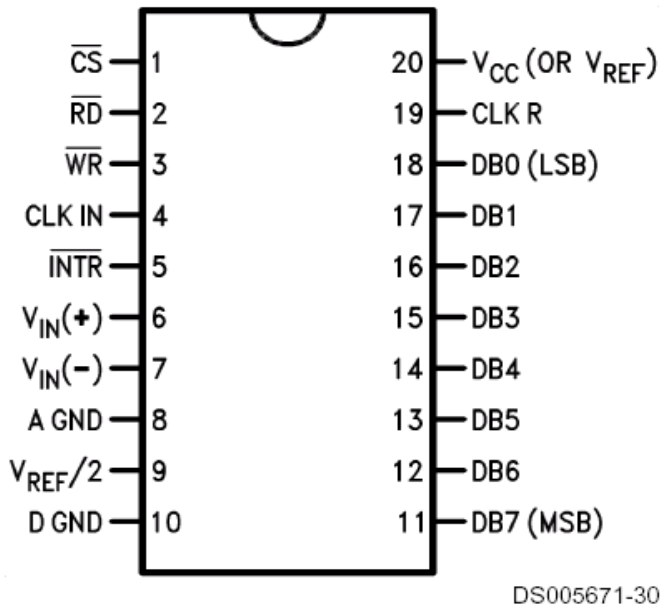
- Tần số xung clock 10 KHz - 1028 KHz.

- Đảm bảo sai số tuyến tính trong dải nhiệt độ từ $40^0C \div 85^OC$.

(a). Bảng chân lý và sơ đồ chân của vi mạch ADC 0809:

Bảng chân lý:

A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7
X	X	X	(?)



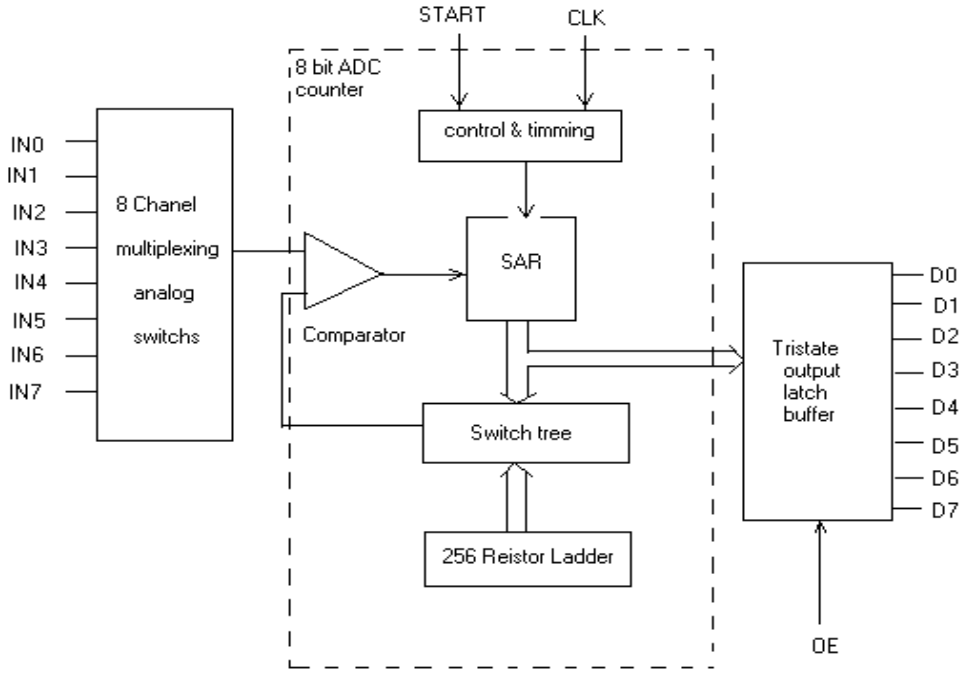
Sơ đồ chân của vi mạch ADC 0809

** ý nghĩa các chân:*

- Các chân 11 đến 17 (DB7 - DB1) : là các đầu ra số.
- /CS cho phép lựa chọn IC hoạt động.
- /RD chân tác động từ bên ngoài để IC thực hiện quá trình chuyển đổi.
- CLK IN: đầu vào xung Clock.
- Ref(+): điện áp vào chuẩn +5V.
- Ref(-): điện áp vào chuẩn 0.
- Vcc: nguồn cung cấp.
- AGND, DGND: chân nối đất.
- Vref/2: 1/2 điện áp chuẩn

(b). Cấu trúc bên trong của ADC 0809:

Cấu trúc bên trong của ADC 0809 được thể hiện ở hình vẽ dưới:



Hoạt động chuyển đổi:

Quá trình biến đổi được bắt đầu bằng một xung Low ngắn hạn ở lối vào /WR. Muốn thế điều kiện cần có là một mức Low của tín hiệu /CS. Sau thời gian biến đổi 100às, lối ra /INTR chuyển sang Low và báo hiệu việc kết thúc quá trình biến đổi. Sau đó qua một mức Low ở lối vào /RD có thể đọc ra các bit số liệu. Sự truy nhập để đọc sẽ dẫn đến hậu quả là tín hiệu /INTR sẽ chuyển trở lại mức cao.

Khi mà lối vào /RD được chuyển hẳn sang mức Low, thì lối ra /INTR chuyển sang Low sau quá trình biến đổi kéo dài 8 chu kì giữ nhịp của bộ giữ nhịp bên trong. Ở tần số giữ nhịp là 640 KHz, chu kì này là 12.5s.

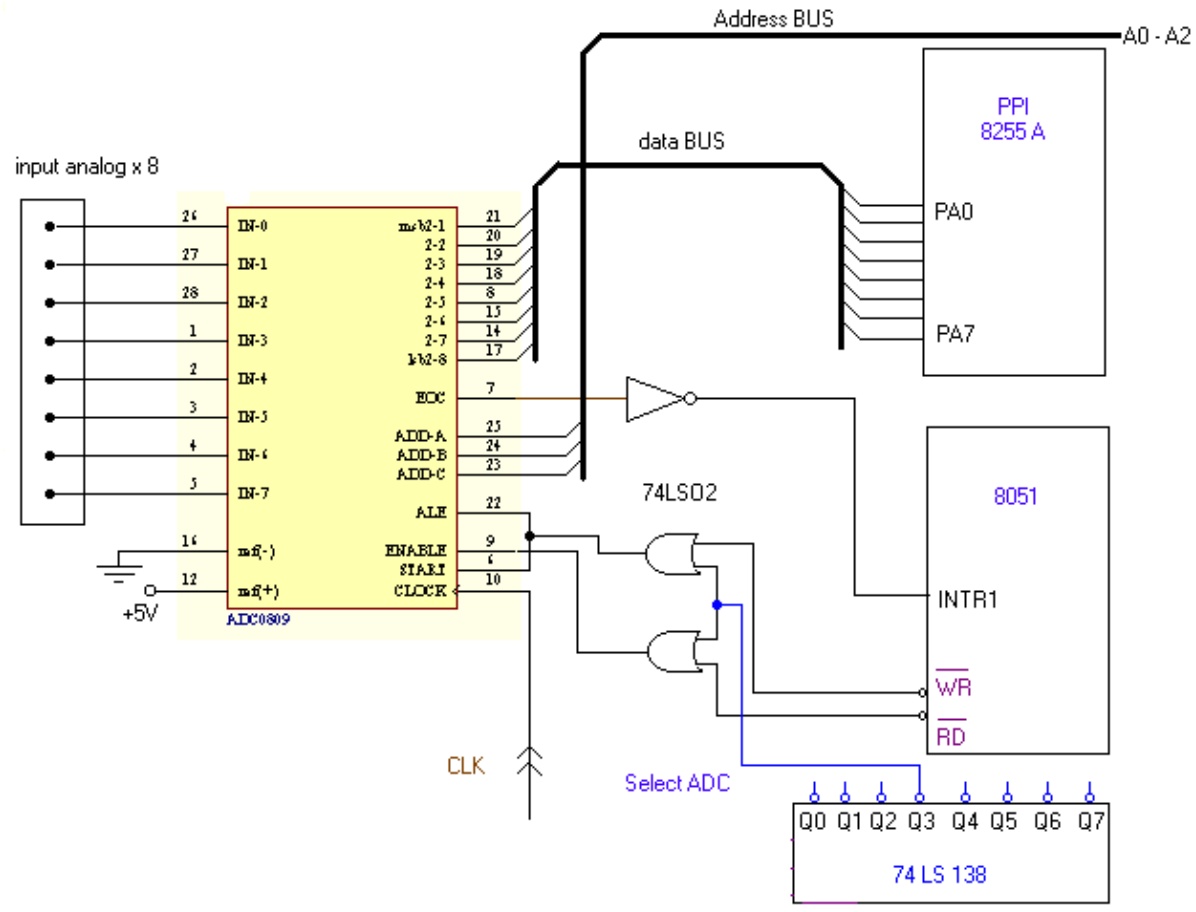
(c). Ghép ADC 0809 với VĐK 8051:

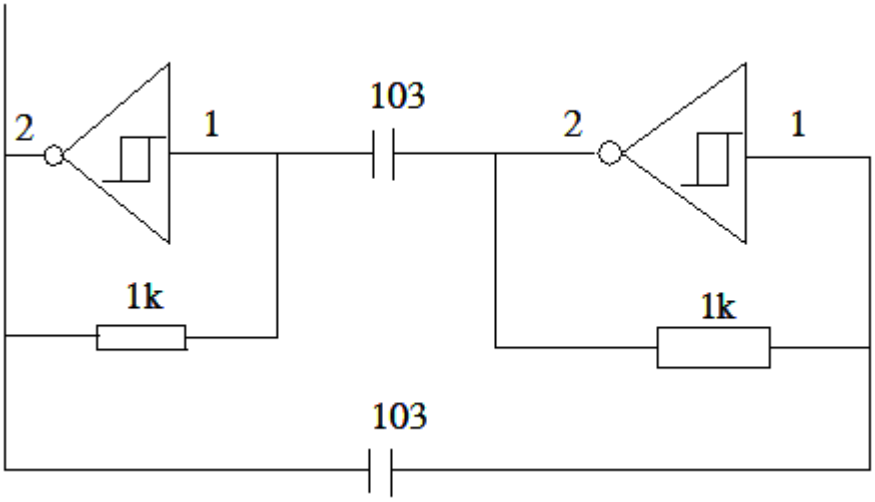
- Các kênh vào Analog được nối vào các đầu vào tương ứng của ADC. Mỗi kênh đó có địa chỉ riêng do tổ hợp 3 bit địa chỉ A, B, C quy định. Các đầu vào địa chỉ này kết nối với đường địa chỉ A_0, A_1, A_2 của Bus địa chỉ của hệ thống. Các đường địa chỉ cao của hệ thống được dùng để tạo tín hiệu chọn chip (/CS) cho ADC0809.

- Tín hiệu /CS được đưa tới đầu vào của mạch OR để khởi động ADC (Start) khi có tín hiệu /WR đồng thời chốt địa chỉ (ALE) của kênh hiện hành có giá trị là giá trị 3 bit A, B, C. Tín hiệu /CS cũng được đưa tới đầu vào của mạch OR thứ hai để tạo tín hiệu OE cùng với /RD nhằm chốt dữ liệu đã biến đổi xong ở đầu ra.

- Vì khi biến đổi xong, ACD 0809 dùng tín hiệu ra chân EOC để báo cho VĐK biết mã nhị phân tương ứng với mức cao của tín hiệu đầu vào đã được tạo ra. Vì vậy ta kết nối EOC với đầu vào ngắt ngoài /INT1 của 8051.

- 8 bit dữ liệu thường được ghép trực tiếp với Bus dữ liệu hệ thống vì bản thân bộ đệm ra là 3 trạng thái, cũng có thể ghép qua 8255.

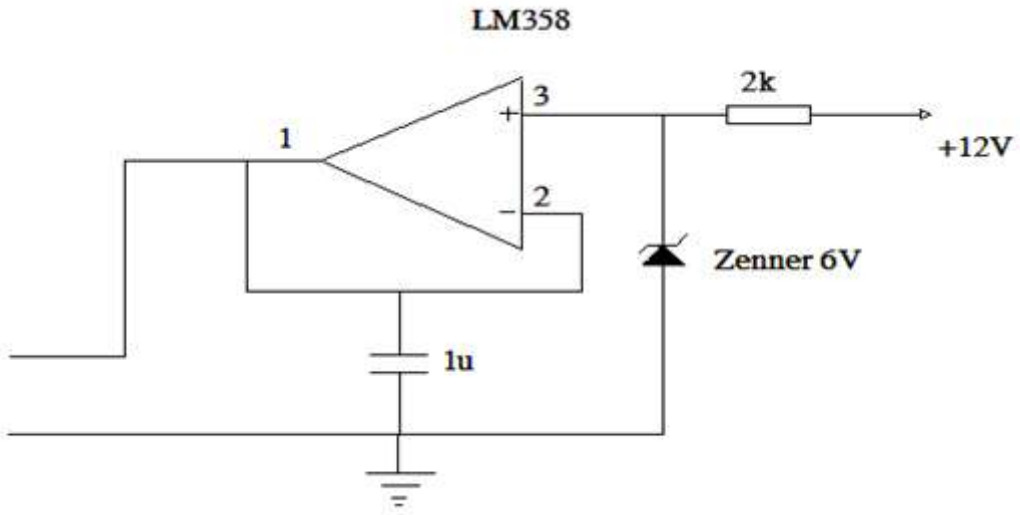




Mạch tạo xung nhịp cho ADC

*Mạch tạo điện áp chuẩn:

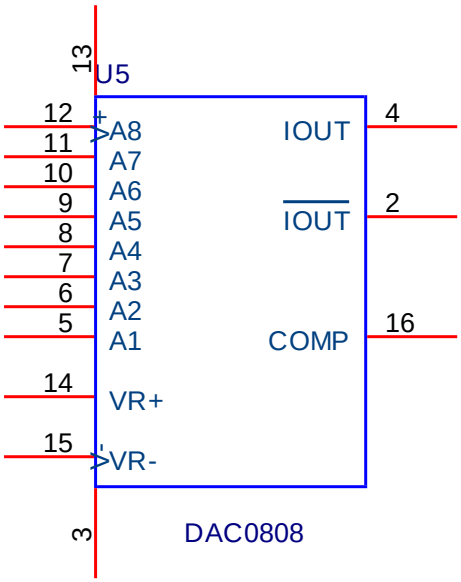
Do ADCkhông cần điều chỉnh điểm 0 nên ta dùng mạch tạo điện áp chuẩn như sau :



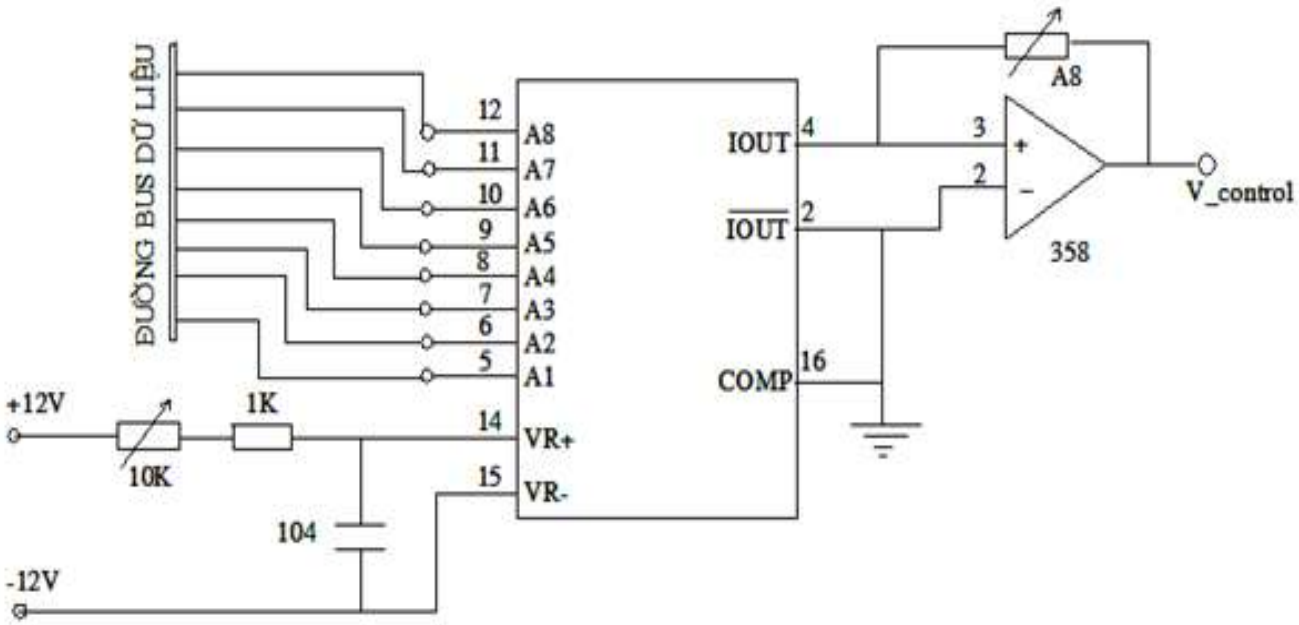
Mạch tạo điện áp chuẩn cho ADC

2.1.2.3. Vi mạch DAC 0808:

Đây là vi mạch thực hiện việc chuyển đổi dữ liệu từ số ra tương tự, điện áp ra được lấy từ một điện áp so sánh xác định. Vi mạch này có độ phân giải là 1/256 giá trị với điện áp ra 10V thì có bước nhảy điện áp là 39,1 mV



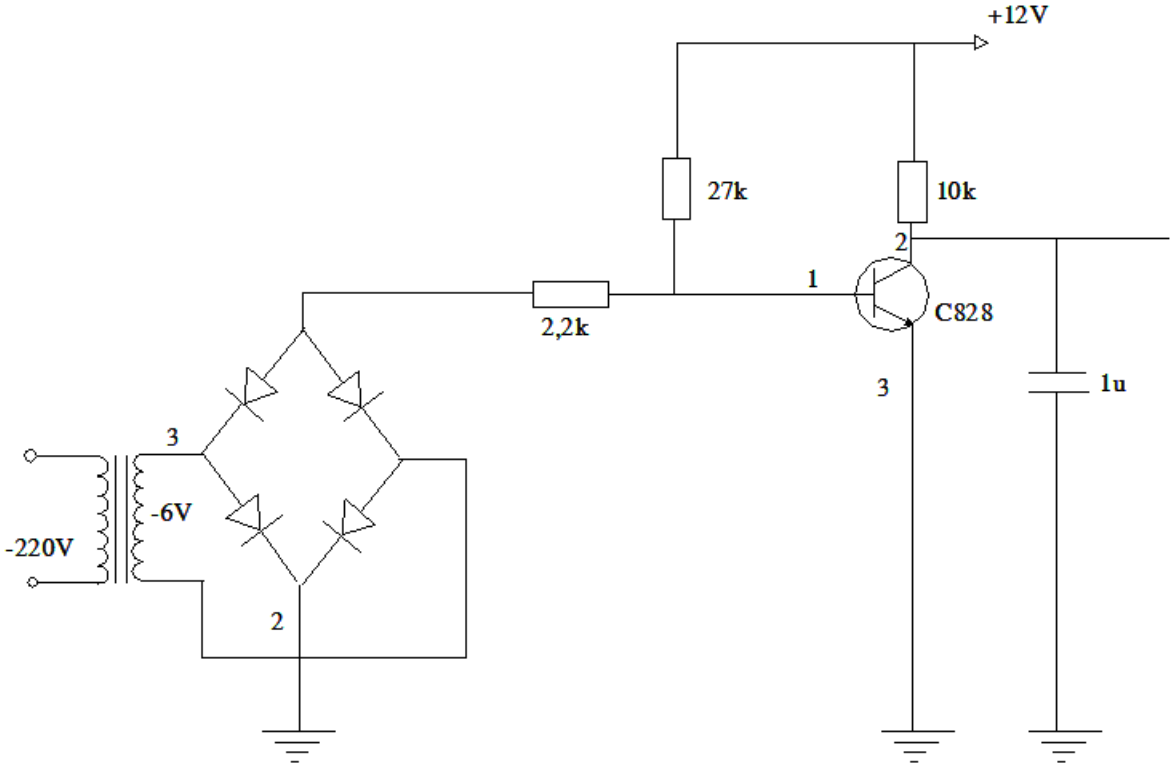
Sơ đồ chân của DAC 0808



Bộ biến đổi DAC0808

2.1.2.4. Mạch tạo tín hiệu mở thyristor:

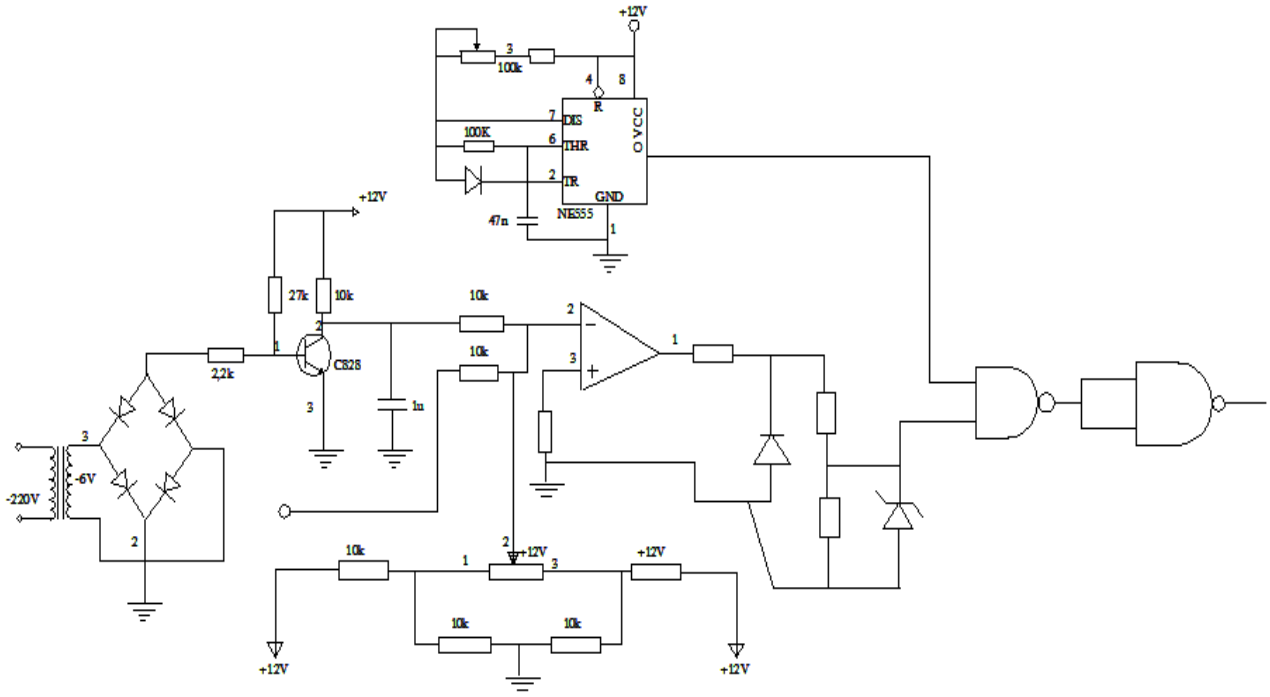
Để đảm bảo tín hiệu mở thyristor đồng bộ với điện áp nguồn , ta dùng mạch đồng pha theo nguyên lý tạo xung tam giác có cùng chu kì vói điện áp nguồn. Sơ đồ mạch được cho dưới đây:



Mạch tạo xung răng cưa đồng pha với điện áp nguồn

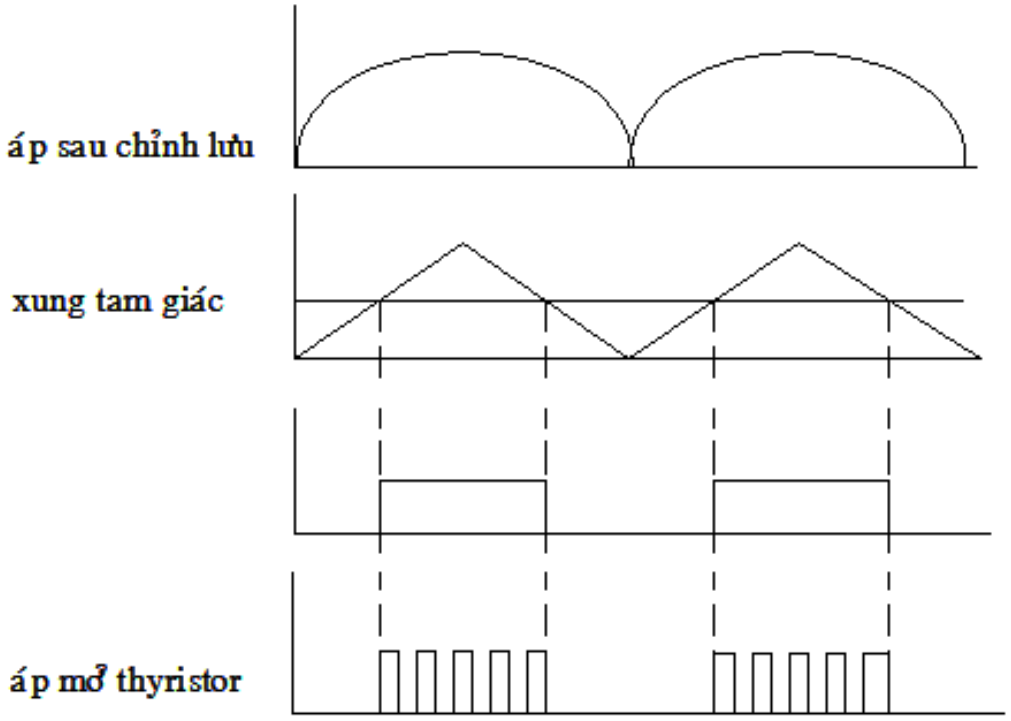
Trong mạch này , điện áp nguồn dược chỉnh lưu về áp 6V,sau đó được cho phóng nạp qua tụ để tạo xung răng cưa. Điện áp xung răng cưa này tiếp tục được so sánh với điện áp điều khiển được đưa ra từ bộ điều khiển và tạo thành xung vuông có bề rộng dùng để mở thyristor.

Để giảm tiêu hao công suất mà đảm bảo mở van tốt,ta dùng chùm xung tạo ra do mạch timer 555. Toàn bộ phần mạch tạo tín hiệu mở thyristor được cho dưới đây:



Mạch tạo xung điều khiển

Giản đồ thời gian trong mộtchu kì của điện áp dùng để mở thyristor được cho trong hình vẽ sau:

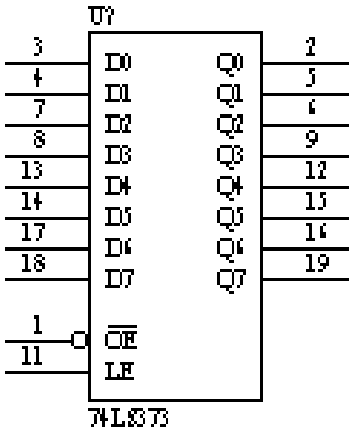


Giản đồ thời gian của điện áp điều khiển

Điện áp sau cùng sẽ được đưa tới mạch khuyếch đại công suất gồm hai tranzitor mắc lặp, đưa qua một biến áp xung rồi đưa đến chân gate của các thyristor.

2.1.*2.5. Các vi mạch phụ trợ khác:*

(a). Vi mạch chốt 74LS373:



Đây là mạch có tác dụng chốt lại số liệu ở đầu vào khi có tín hiệu tích cực, đầu ra sẽ không bị biến đổi khi tín hiệu đầu vào đã mất. Nó chỉ thay đổi khi tín hiệu chốt tích cực trở lại. Bên ngoài vỏ cũng có tín hiệu /OE cho phép hoạt động. Khi có yêu cầu chốt chân LE sẽ được tích cực.

Trong ghép nối với 8051:

- Chân /OE (số 1) của 74LS373 được nối đất.
- Chân LE(số 11) của 74LS373 được nối với chân ALE (số 30) của 8051.

(b). Vi mạch MAX 232 (của hãng Maxim):

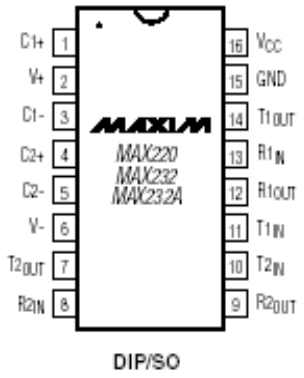
Khi thực hiện giao tiếp giữa VĐK với máy tính thì ta cần có vi mạch MAX 232 chuyển đổi.

MAX cho phép đọc vào cũng như đưa ra 8 tín hiệu TTL qua giao diện nối tiếp của máy tính PC.

Thích ứng với mức tín hiệu (+12V, -12V) trên giao diện RS 232.

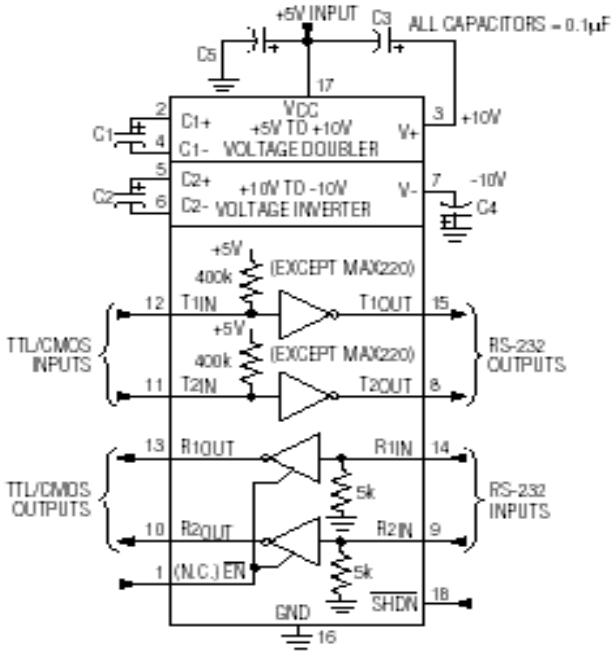
Vi mạch này nhận mức RS 232 đã được gửi từ máy PC và biến đổi tín hiệu này thành tín hiệu TTL, để rồi sau đó dẫn đến VĐK. Tín hiệu từ VĐK được biến đổi thành tín hiệu mức +12V/ -12V và gửi tới máy tính PC.

Vi mạch có khả năng thiết lập tốc độ Baud.



Sơ đồ chân vi mạch MAX 232

MAX 232 được nối với cổng truyền thông nối tiếp của 8051 qua 2 chân: RxD và TxD. Nhờ đó mà luồng dữ liệu có thể dịch chuyển 2 chiều từ máy tính xuống vi điều khiển và ngược lại.

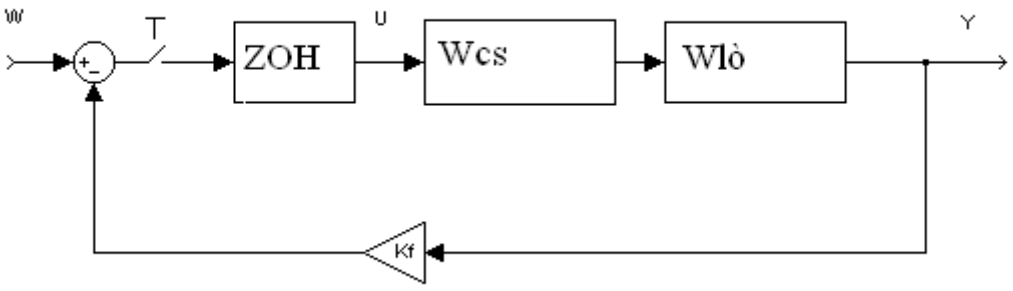


Cấu trúc bên trong của MAX 232

Qua cấu trúc bên trong của MAX 232 ta thấy vi mạch này có thể kết nối một lúc với cả 2 đầu RS 232.

2.2. Phân tích hệ thống điều khiển số

Khi chưa có bộ điều khiển trong hệ thống , sơ đồ khối của hệ thống như sau :



Trong đó $W_{kđcs}$ là hàm truyền của mạch công suất:

$$W_{kđcs}(p) = \frac{K_1}{T_1 * p + 1}$$

Trong bài tập này lấy $K_1=120$

$T_1=0.02$ sec

$$=> W_{kđcs}(p) = \frac{120}{0.02 * p + 1}$$

$W_{lò}$ là hàm truyền của lò:

$$W_{lò}(p) = \frac{K_{lo}}{T_2 * p + 1} e^{-pt} = \frac{K_{lo}}{(T_2 * p + 1)(\tau * p + 1)}$$

Với τ=150 sec

T_2= 500 sec

K_{lo} = 4

Trong bài tập này lấy K_{lo} = 4 ; T_2= 500 sec

=> Hàm truyền của lò là: $\frac{4}{500p + 1} e^{-p.150}$

Khai triển gần đúng ta được :

$$W_{lo} = \frac{4}{(150 * p + 1)(500 * p + 1)}$$

K_f là hàm truyền của mạch phản hồi .

K_f=0.5

Ta thấy rằng hằng số thời gian của mạch khuếch đại công suất rất nhỏ so với hằng số thời gian của lò nên ta có thể bỏ qua. Như vậy đối tượng điều khiển ở đây gồm lò và mạch công suất có hàm truyền như sau :

$$Wđt = \frac{120 * 4}{(150 * p + 1)(500 * p + 1)} = \frac{480}{(150 * p + 1)(500 * p + 1)}$$

$$= \frac{480}{75000 * p * p + 650 * p + 1}$$

Căn cứ vào khả năng của vi điều khiển và đặc điểm của đối tượng ta có thể chọn chu kì trích mẫu của hệ thống là 10 giây.

Với chu kì trích mẫu này chuyển sang miền số ta có hàm truyền rời rạc của đối tượng sử dụng Matlab :

```
>> Wdtz=c2d(Wdt,10,'ZOH')

Transfer function:
  0.3109 z + 0.3021
---------------------
z^2 - 1.916 z + 0.917

Sampling time: 10
```

Chuyển sang phương trình trạng thái :

$$\begin{cases} x(k+1) = \begin{bmatrix} 1.916 & -0.917 \\ 1 & 0 \end{bmatrix} * x(k) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} * u(k) \\ y(k) = \begin{bmatrix} 0.3109 & 0.3021 \end{bmatrix} * x(k) \end{cases}$$

2.2.1.Kiểm tra tính điều khiển được và tính quan sát được của hệ thống

Ta đã xác định được phương trình trạng thái của đối tượng điều khiển (công thức trên) .Để kiểm tra tính điều khiển được của đốu tượng cần ma trân điều khiển được :

P_d=[Bd Ad*Bd]=$\begin{bmatrix} 1 & 1.916 \\ 0 & 1 \end{bmatrix}$

Có det(P_d)=1 ≠0 suy ra rank(p_d) = 2 do đó đối tượng là điều khiển được.
Để kiểm tra tính quan sát được của đối tượng ta cần tính ma trận quan sát của hệ thống :

N_d= [Cd' Ad'*Cd']=$\begin{bmatrix} 0.3109 & 0.8977 \\ 0.3021 & -0.2851 \end{bmatrix}$

Suy ra : det(N_d) = -0.3598 ≠0

=> Hệ thống là quan sát được.

2.2.2.Xét ổn định của đối tượng :

Phần này sẽ xét ổn định của đối tượng , nghĩa là xét ổn định của một hệ thống hở trong đó không có bộ điều khiển .Công thức (3.1) đã cho biết hàm truyền đạt rời rạc của đối tượng . Phương trình đặc tính của đối tượng là :

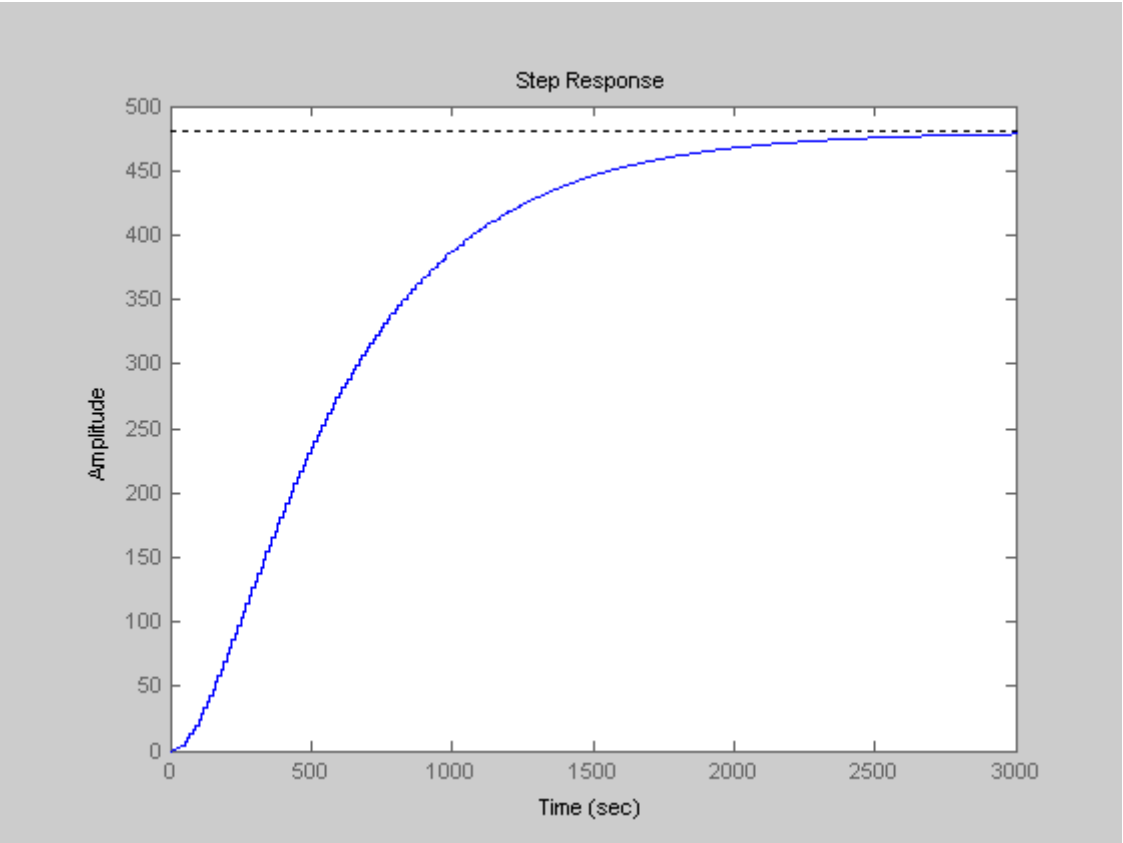
z^2 - 1.916 z + 0.917 = 0

Giải phương trình của phương trình này ta có các điểm cực của dối tượng là :

Z1= 0.98564

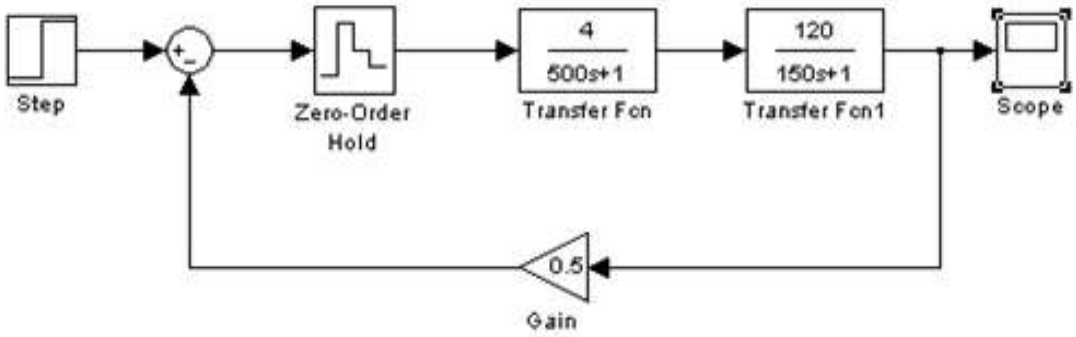
Z2= 0.93035

các điểm cực này đều nằm bên trong đường tròn đơn vị nên đối tượng là ổn định , tức là hệ thống ổn định . Ta có đặc tính quá độ của hệ thống như hình vẽ dưới:



2.2.3.Xét ổn định của hệ thống kín khi chưa có bộ điều khiển

Xét đối tượng trong một hệ thống kín nhưng chưa có bộ điều khiển. Cần xét ổn định của hệ thống này :
Mô hình của hệ thống:



Hàm truyền rời rạc của đối tượng được cho bởi công thức :

Transfer function:
  0.3109 z + 0.3021
---
z^2 - 1.916 z + 0.917

Sampling time: 10

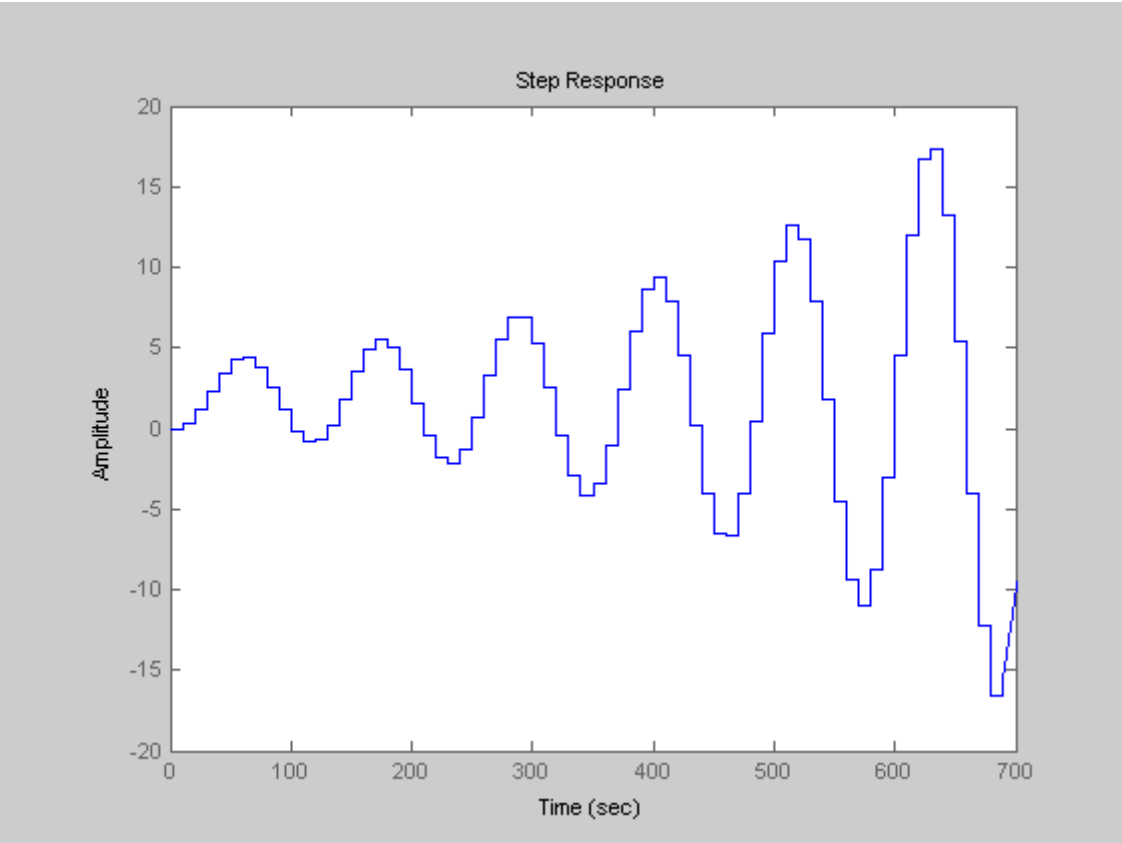
Hàm truyền đạt của hệ thống có hồi tiếp âm là :
Transfer function:
 0.3109 z + 0.3021
---
z^2 - 1.76 z + 1.068

Sampling time: 10
Phương trình đặc tính của hệ thống là : z^2 - 1.76 z + 1.068

Giải phương trình trên ta có các điểm cực của hệ thống là :
  z1= 0.88 + 0.5418i
  z2=   0.88 - 0.5418i

Căn cứ vào Modul của 2 điểm cực ta thấy hệ thống kín không ổn định vì các điểm cực có Modul lớn hơn 1 .

Đặc tính quá độ của hệ thống là:



2.3. Tổng hợp hệ thống

2.3.1. Tổng hợp hệ thống dùng bộ điều khiển PID:

2.3.1.1. Bộ điều khiển PID và tìm các thông số cho bộ điều khiển PID:

Bộ điều khiển PID (Proportional - Integral - Derivative) là bộ điều khiển kinh điển, được sử dụng rất nhiều khi tổng hợp hệ thống. Mặc dù hiện nay đã có các phương pháp tổng hợp hệ thống khác tốt hơn (như phương pháp dùng hồi tiếp trạng thái sẽ được xét ở phần sau) nhưng bộ điều khiển PID vẫn tiếp tục được sử dụng rộng rãi. Bộ điều khiển PID gồm ba thành phần: thành phần tỉ lệ (P), thành phần tích phân (I) và thành phần vi phân (D). Mỗi thành phần có những ảnh hưởng nhất định đến chất lượng của hệ thống, và việc lựa chọn một bộ tham số phù hợp cho ba thành phần đó sẽ đem lại cho hệ thống chất lượng mong muốn.

Hàm truyền liên tục của bộ điều khiển PID có thể được viết dưới dạng sau:

$$W_{PID}(p) = K_P + \frac{K_I}{p} + K_d p$$

Để chuyển từ bộ PID liên tục sang bộ PID số có vài cách khác nhau. Một phương pháp là chuyển gần đúng từng thành phần của bộ PID từ liên tục sang dạng rời rạc như sau:

- Thành phần tỉ lệ được giữ nguyên.
- Thành phần tích phân được lấy gần đúng theo Tustin: $\frac{K_I}{p} = \frac{K_I T(z+1)}{2(z-1)}$

- Thành phần vi phân được lấy gần đúng: $K_D p = \frac{2K_D(z-1)}{T.z}$

Với T là chu kì trích mẫu.

Như vậy, hàm truyền rời rạc của bộ PID số là:

$$W_{PID}(z) = K_P + \frac{K_I T(z+1)}{2(z-1)} + \frac{K_D(z-1)}{T.z}$$

$$= \frac{2.T.K_P.z.(z-1) + K_I.T^2.z.(z+1) + 2K_D(z-1)^2}{2.T.z.(z-1)}$$

$$= \frac{(K_P + 0.5K_I T + \frac{K_D}{T})z^2 - (K_P - 0.5K_I T + 2\frac{K_D}{T})z + \frac{K_D}{T}}{z(z-1)}$$

Có thể thấy, bộ điều khiển PID số có 2 điểm cực (0 và 1) và tối đa 2 điểm Zero.

Có nhiều phương pháp khác nhau để tổng hợp hệ thống dùng bộ điều khiển PID nói chung và bộ điều khiển PID số nói riêng. Tuy nhiên, hiện vẫn chưa có một phương pháp tổng quát và chính xác nào để tìm được bộ điều khiển PID tốt nhất cho một hệ thống. Các phương pháp cho đến nay vẫn chỉ cho phép xác định một cách tương đối các thông số của bộ PID (đáp ứng được phần nào chất lượng mong muốn). Sau đó, phải tiếp tục thay đổi các thông số (trong một lân cận xung quanh giá trị tìm được) và "mò" cho đến khi tìm được bộ thông số đáp ứng yêu cầu chất lượng đã đề ra. Việc dò tìm các thông số cho bộ điều khiển PID phải dựa trên các nguyên tắc về ảnh hưởng của từng thành phần trong bộ điều khiển PID đến chất lượng của hệ thống. Một cách chung nhất, có thể tóm tắt các nguyên tắc đó trong bảng sau:

	Rise Time	Overshoot	Settling Time	Steady State Error
K_P	Giảm	Tăng	Thay đổi ít	Giảm
K_I	Giảm	Tăng	Tăng	Triệt tiêu
K_D	Thay đổi ít	Giảm	Giảm	Thay đổi ít

Lấy một ví dụ, với sai lệch tĩnh (Steady State Error), khi tăng K_P sẽ làm giảm sai lệch tĩnh, tăng K_I sẽ có thể triệt tiêu được sai lệch tĩnh, còn K_D ít có ảnh hưởng. Tất nhiên, các nguyên tắc trên không đúng tuyệt đối bởi ba thông số trên có ảnh hưởng lẫn nhau và sự thay đổi của bất kì một thông số nào cũng có thể gây ảnh hưởng không nhỏ đến tác dụng của hai thông số còn lại.

Riêng đối với bộ PID số, có hai hướng chính để tổng hợp là:

- Hướng thứ nhất là tổng hợp bộ điều khiển PID liên tục trước, sau đó chuyển bộ điều khiển tìm được sang miền rời rạc bằng công thức gần đúng đã trình bày ở trên. Hướng này chỉ áp dụng được khi chu kì lấy mẫu của hệ số nhỏ hơn rất nhiều lần so với hằng số thời gian nhỏ nhất trong đối tượng.

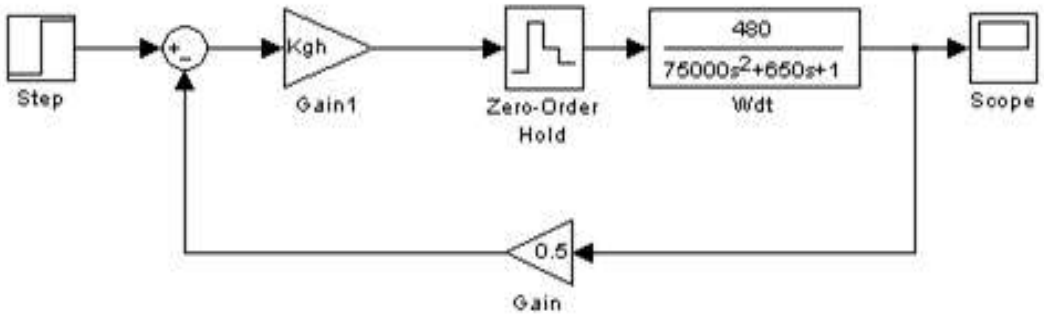
- Hướng thứ hai là tổng hợp trực tiếp trong miền rời rạc. Phương pháp này không bị hạn chế về chu kì lấy mẫu như phương pháp trên.

Dễ nhận thấy rằng việc lựa chọn theo hướng thứ hai là phù hợp hơn cả bởi không có sự phụ thuộc vào thời gian lấy mẫu, đồng thời có thể tính toán trực tiếp ra bộ PID số mà không cần qua khâu biến đổi ở trên.

2.3.1.2. Chọn thông số cho bộ điều khiển PID:

Có rất nhiều cách để lựa chọn thông số cho bộ PID, giả dụ như ta có thể mò bằng cách thay đổi các thông số để hệ ổn định, rồi sau đó chỉnh định lại các thông số để đạt được hệ ổn định với các chỉ tiêu thoả mãn. Tuy nhiên theo cách đó thì mất nhiều thời gian đồng thời không đem lại hiệu quả cao. Một trong những công cụ mạnh mà MATLAB đưa ra để giải quyết vấn đề trên là khảo sát tính ổn định của hệ thống bằng quĩ đạo nghiệm số. Ta có thể xác định ra ngay các thông số (K_P, T_I, T_D) đảm bảo cho hệ ổn định bằng **rltool** trong thư viện MATLAB hoặc ta có thể làm bằng phương pháp sau cũng có hiệu quả không kém :

- Trước tiên ta tìm K_{gh} của hệ theo mô hình sau:



Việc xác định Kgh đựoc tiến hành trên MATLAB bằng hai hàm **rlocus** và **rlocfind:**

```
>> Wdt=tf(num,den)
Transfer function:
        480
---------------------
75000 s^2 + 650 s + 1
 >> Wdtz=c2d(Wdt,10,'ZOH')
```

Transfer function:

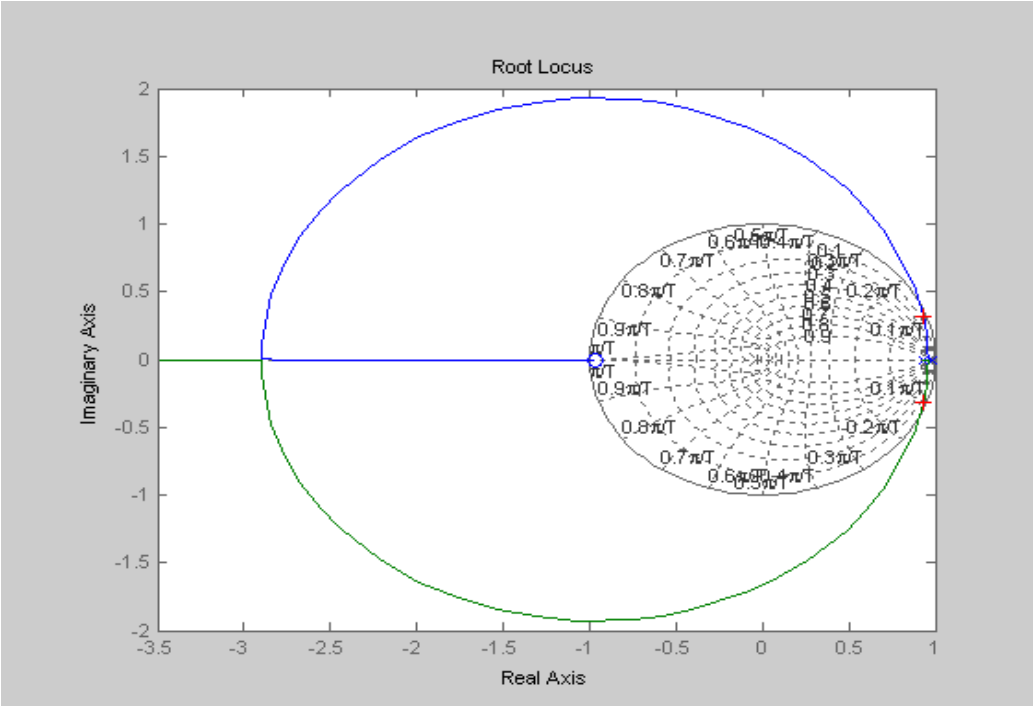
0.3109 z + 0.3021

---

z^2 - 1.916 z + 0.917

Sampling time: 10

```
>> rlocus(Wdtz)
>> hold on
>> k=rlocfind(Wdtz)
Select a point in the graphics window
selected_point =
   0.7814 + 0.2422i
k =
    0.1635
```



Quĩ đạo nghiệm số của đối tượng

Như vậy ta tìm được K_{gh}=0.1635

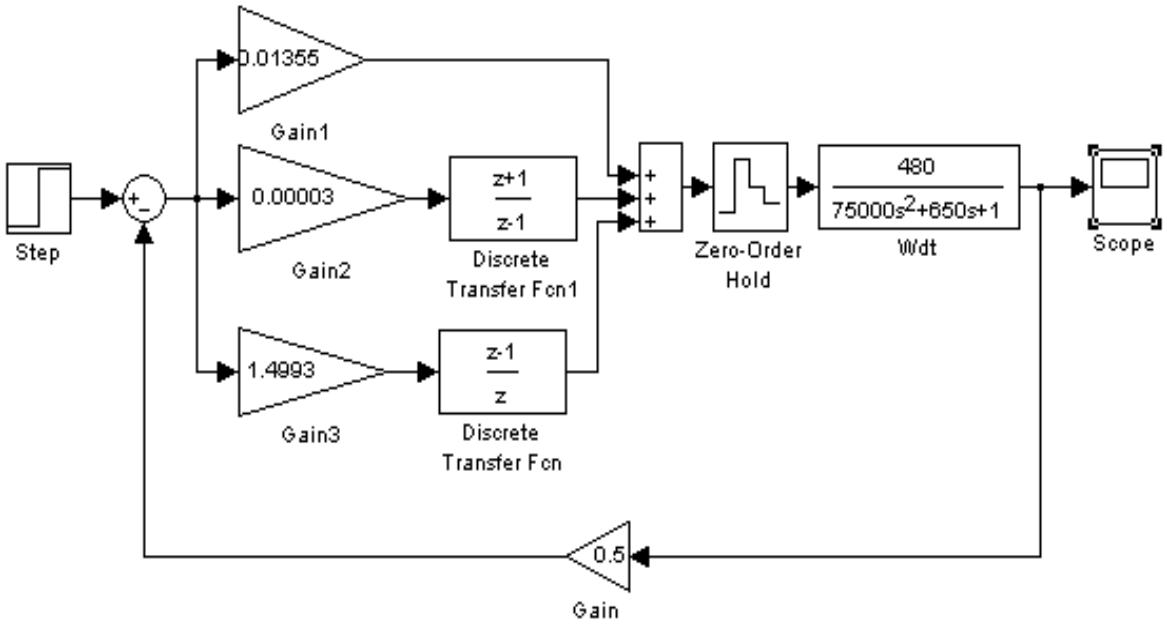
-Việc tiếp theo là đặt điểm zezo cho khâu PID để bù điểm cực của đối tượng . Đã biết đối tượng có hai điểm cực là z_1 = 0.98564 và z_2 = 0.93035 , do đó ta

chọn luôn hai điểm này làm điểm zezo cho khâu PID . Ta được các thông số cần tìm thoả mãn 3 phương trình sau:

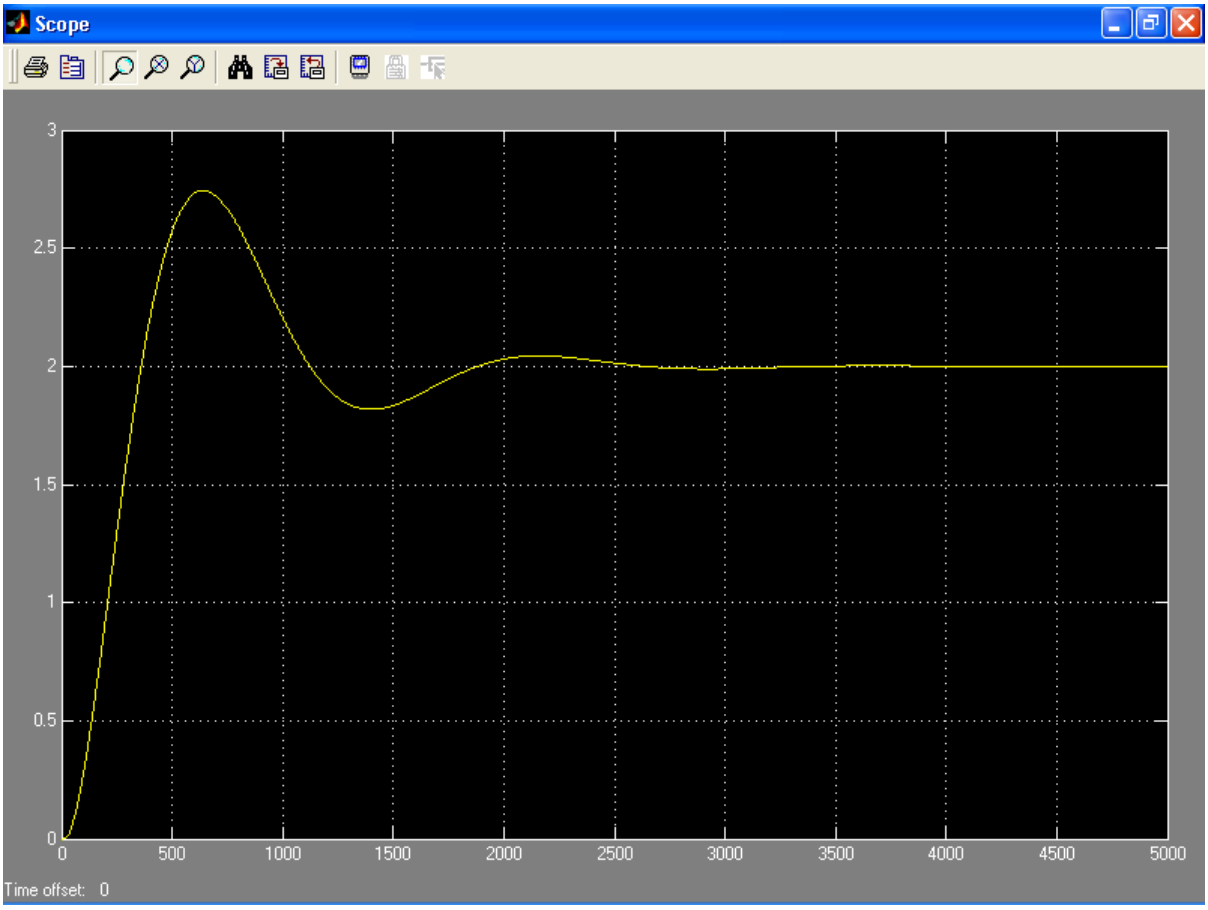
- $K_p+0.5K_i+K_đ/T=K_{gh}=0.1635$
- $K_p-0.5K_iT+2K_đ/T=K_{gh}(z_1+z_2)= 0.31326$
- $K_đ/T=K_{gh} . z_1z_2 = 0.14993$

Giải ra ta thu được $\begin{cases} K_p=0.01355 \\ Ki=0.0000309 \\ K_đ=1,4993 \end{cases}$

Thay thông số của bộ PID vào mô hình sau:

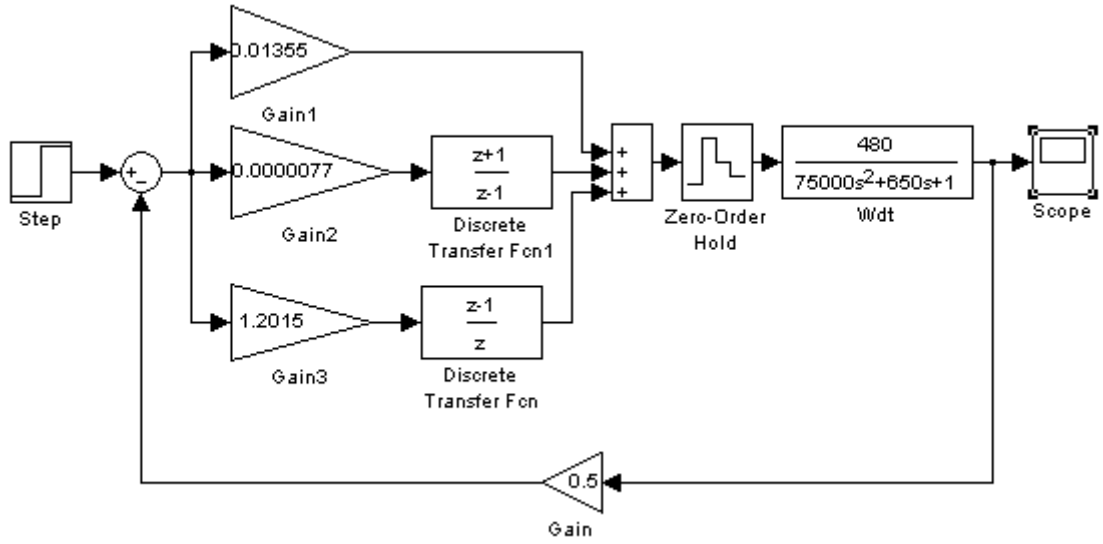


Thu được kết quả :

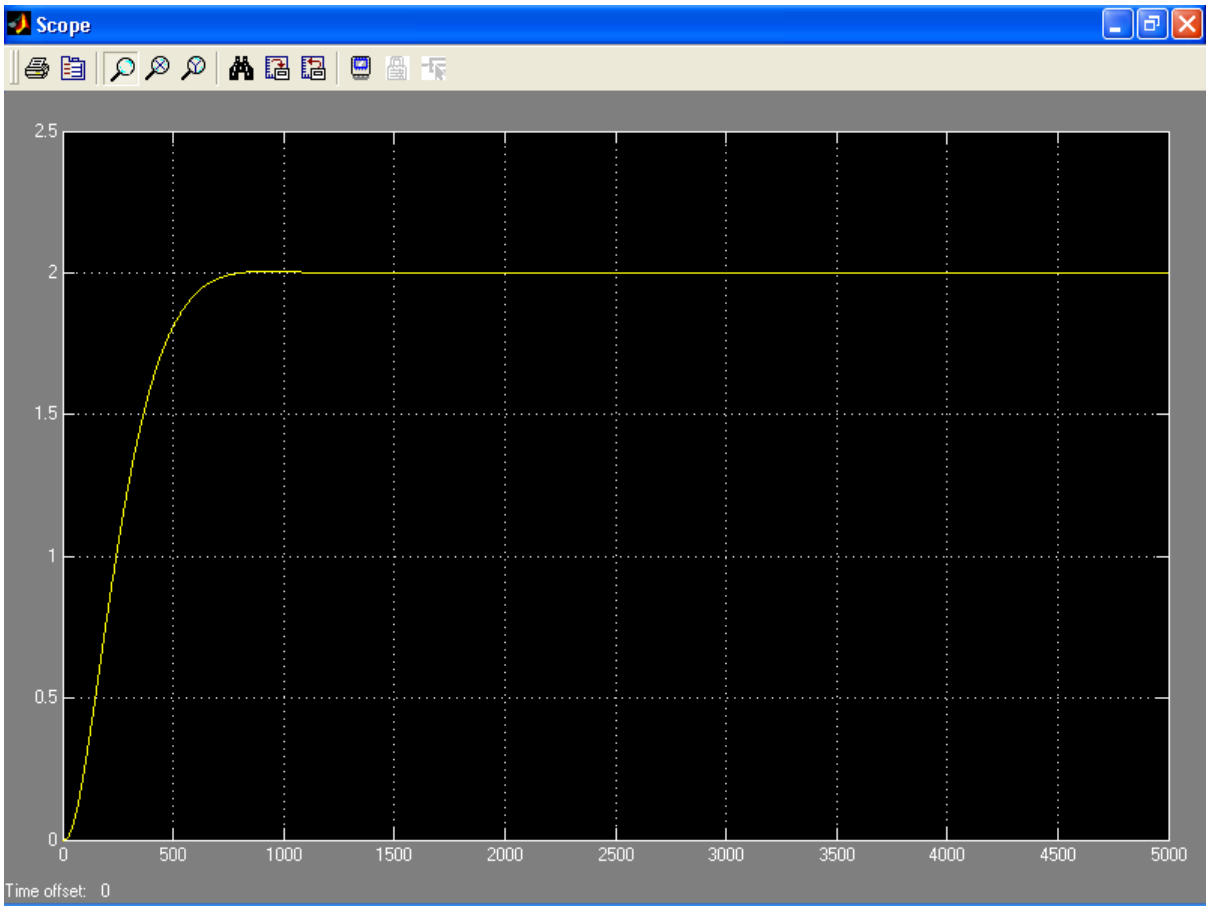


Mặc dù hệ ổn định nhưng độ quá điều chỉnh là tương đối lớn , điều này đặc biệt nguy hại bởi khi đối tượng làm việc gần giới hạn nhiệt độ cho phép sẽ gây ra mất an toàn cho các thiết bị điều chỉnh cũng như bản thân đối tượng.

Căn cứ vào những tác động của các thông số K_p,K_i,$K_đ$ tới đặc tính quá độ của đối tượng tiến hành tinh chỉnh lại lại các thông số ta thu được đặc tính quá độ của đối tượng sau khi chỉnh định lại như sau:



Các chỉ tiêu chất lượng : Độ quá điều chỉnh là 0.06% và thời gian xác lập là 730s . Như vậy chất lượng của hệ thống là rất tốt!



2.3.2. Tổng hợp hệ thống dùng hồi tiếp trạng thái

Mặc dù bộ điều khiển PID số đã được sử dụng rất rộng rãi và phổ biến nhưng nó thường có nhiều nhược điểm khó khắc phục như : Việc tổng hợp và thiết kế phức tạp , mang nặng tính mò mẫm, kém chính xác , chất lượng khó có thể cao , chỉ có áp dụng với từng trường hợp cụ thể mà không thể tổng quát hoá , chưa có một phương pháp tổng quát và chính xác để thiết kế .Hiện nay đã xuất hiện nhiều phương pháp tổng hợp hệ thốnghiện đại hơn tôt hơn và khắc phục được nhược điểmkể trên của bộ PID số. Một trong các phương pháp đó là phương pháp tổng hợp hệ thống dùng hồi tiếp trạng thái . Sử dụng phương pháp nàycó thể áp đặt khá chính xác các điểm cực cho hệ thống , mà ta đã biết các điểm cực quyết định đến tính chất , chất lượng hệ thống. Sau đây là phương pháp tổng hợp hệ thống dùng hồi tiếp trạng thái.

2.3.2.1.Nhắc lại về mô hình đối tượng

Như đã trình bày ở tr ên cho ta phương trình trạng thái của đối tượng :

$$\begin{cases} X(k+1) = \begin{bmatrix} 1.916 & -0.917 \\ 1 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(k) \\ y(k)=[0.3109 \quad 0.3021]\, x(k) \end{cases}$$

Trong ph ần trên ta cũng đã kiểm tra tính điều khiển được và tính quan sát được của đối tượng và đã khẳng định : Đối tượng là quan sát được và điều khiển được! Hai tính chất trên là điều kiện qua trọng để có thể tổng hợp hệ thống dùng hồi tiếp trạng thái.

2.3.2.2.Các phương pháp tìm bộ hồi tiếp trạng thái

Cho một đối tượng mô tả bởi phương trình trạng thái sau:

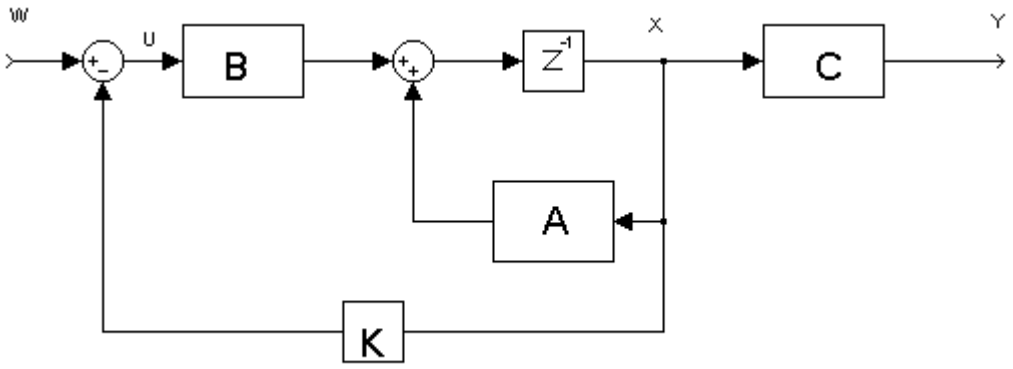
$$X(k+1)=A.x(k)+B.u(k)$$
$$Y(k)=C.x(k)$$

Các điểm cực của đối tượng là nghiệm của phương trình đặc tính :

$$Det(z.I-A)=0$$

Giải phương trình đặc tính ta có được các điểm cực của đối tượng : $pc_1,pc_2...pc_n$

Vị trí của các điểm cực sẽ ảnh hưởng đến chất lượng của hệ thống.Giả sử ta cần tìm một bộ hồi tiếp trạng thái cho đối tượng trên sao cho hệ thỗng đã hồi tiếp có các điểm cực mong muốn $pc_1,pc_2...pc_n$ .Mô hình của hệ thống có hồi tiếp trạng thái được cho bởi hình sau :



Cho đầu vào w=0 ta có :

$$X(k+1)=A.x(k)+B.u(k)$$
$$u(k)=-K.x(k)$$
$$\Rightarrow x(k+1)=A.x(k)-B.K.x(k)=(A-B.K).x(k)$$

Như vậy hệ mới sẽ có phương trình đặc tính là : det(z.I-A+B.K)=0 và theo yêu cầu đã đề ra , phương trình đặc tính này phải có các nghiệm $pm_1,pm_2...pm_n$ .Có một số cách để xác định K thoả mãn yêu cầu trên như sau:

a). Cách 1

Vì phương trình đặc tính mới det (ZI-A+BK)=0 phải có các nghiệm pm_i nên nó phải có dạng sau :

$Det(ZI-A+BK)=(z-pm_1)(z-pm_2)...(z-pm_n)=z^n+c_1z^{n-1}+...c_n$

Trong đó K chỉ là một vector các hệ số : $K=[K_1,K_2,...K_n]$.Khai triển đẳng thức trên và cân bằng hệ số hai vế ta tìm được $K=[K_1,K_2,...K_n]$.

b.) Cách 2

-Gọi p là ma trận điều khiển được của đối tượng:$[B, AB,...,A^{n-1}B]$

-Định nghĩa một ma trân w như sau :

$$w=\begin{bmatrix} a_{n-1} & a_{n-2} & ... & a_1 & 1 \\ a_{n-2} & a_{n-3} & ... & 1 & 0 \\ ... & ... & ... & ... & ... \\ a_1 & 1 & ... & 0 & 0 \\ 1 & 0 & ... & 0 & 0 \end{bmatrix}$$

Trong đó $det(ZI-A) =z^n+a_1z^{n-1}+...a_n$

Tính ma trận T=p.w

Tính K theo công thức sau :

$K=[c_n-a_{n,c}n-1_{-a}n-1_{,...}c_1-a_1].T^{-1}$

c.) Cách 3

- Tính $\phi(A)=A^n+c_1A^{n-1}...c_n.I$
- Tính K theo công thức :

$K=[0,0,...1][B,AB,...,A^{n-1}B].\phi(A)$

Ba cách trên tuy phương pháp khác nhau nhưng hoàn toàn tương đương và đều cho các kết quả giống nhau.

d.) Cách 4

- Sử dụng hàm Acker trong thư viện của Matlab, cú pháp dạng
  Acker(A,B,p)
  Trong đó p là vector cột các điểm cực cần gán ,hàm này sẽ trả về ma trận hàng hồi tiếp K...

2.3.2.3.Phương pháp chọn điểm cực của Bessel

Khi đã có phương pháp tìmh bộ hồi tiếp trạng thái thì vấn đề còn lại bây giờ là tìm điểm cực mới của hệ thống thế nào để hệ thống đạt chất lượng như mong muốn.Bessel đã xác định các điểm cực chuẩn cho các hệ thống (liên tục) bậc k sao cho với các điểm cực đó hệ thống đạt chất lượng như sau : không có quá điều chỉnh và thời gian quá độ là T=1s. Bessel đã tổng kết các điểm cực chuẩn đó trong bảng sau :

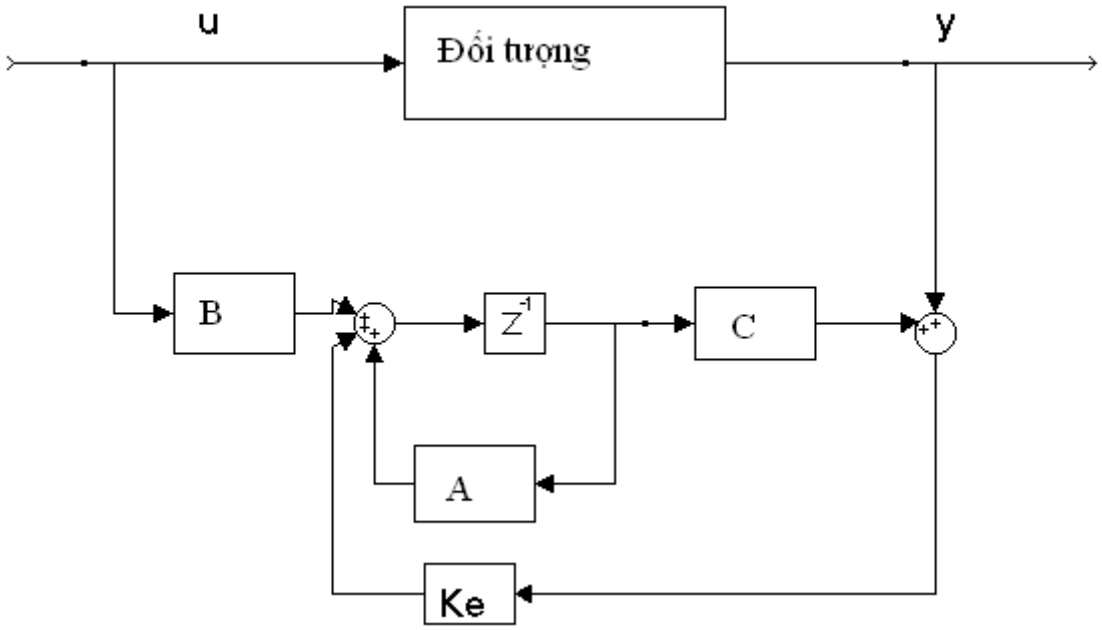
Bậc	Điểm cực chuẩn
1	-4.6200
2	-4.0530±2.3400i
3	-5.0093, -3.9668±3.7845i

Nếu muốn hệ có quá trình quá độ không có quá hiệu chỉnh và thời gian quá độ là T bất kỳ thì hệ thống cần có các điểm cực được xác định bằng cách lấy các điểm cực trong bảng trên chia cho T.

Với hệ thống xung số , để xác định các điểm cực cần thiết, trước hết ta xác địng các điểm cực trong miền liên tục theo quy tắc trên sau đó chuyển các điểm cực này sang miền Z theo công thức $z_k=e^{T.pk}$.

2.3.2.4. Xây dựng bộ ước lượng trạng thái (Bộ quan sát trạng thái)

Một vấn đề nảy sinh khi ta tổng hợp hệ thống dùng hồi tiếp trạng thái là trong thực tế hiếm khi có thể đo được trực tiếp các biến trạng thái (x) của đối tượng . Vậy ta phải làm cách nào để xác định được các biến trạng thái của đối tượng chỉ dựa trên các đại lượng đo được hoặc biết được (như tín hiệu điều khiển,đầu ra của đối tượng ...). Xây dựng được một hệ thống như thế chính là xây dựng được bộ ước lượng trạng thái .Bộ ước lượng trạng thái dựa vào các đại lượng có thể biết được là đầu vào của đối tượng (chính là tín hiệu điều khiển) và đầu ra của đối tượng để xác định trạng thái của đối tượng .Chỉ làm được điều này khi đối tượng là quan sát được ,và ta thấy rõ là đối tượng trong bài tập này thoả mãn điều kiện đó, do đó có thể xây dựng được bộ ước lượng trạng thái cho đối tượng. Bộ ước lượng trạng thái cho đối tượng được cho bởi mô hình sau :



Mục đích của bộ ước lượng trạng thái là phải làm sao cho e(k) tiến đến 0.Nhận thấy hệ phương trình trên có dạng giống với ph ần ta đã trình bầy. Vì vậy ta có thể áp dụng phương pháp đặt điểm cực để tìm Ke.
Với đối tượng cụ thể ta chọn điểm cực theo Bessel cho hệ bâc hai sao cho thời gian quá độ là T=500s , suy ra điểm cực cần thiết là:

$P_{1,2}$=(- 0.0081 ± 0.0047i)
Và trong miền Z thì điểm cực là :
$Zp_{1,2}$=exp(10$p_{1,2}$) =0.9211 ± 0.0431i
Suy ra phương trình đặc tính mới là:
Det(zI-A+KeC)=(z-zp_1)(z-zp_2)=(z-0.9211+0.0431i)(z-0.9211-0.0431i)
=z^2-1.8422z + 0.8503

Ta lại có:

$$\text{Det(zI-A+KeC)=det(z.}\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\text{-}\begin{bmatrix}1.916 & -0.917\\ 1 & 0\end{bmatrix}+\begin{bmatrix}Ke1\\ Ke2\end{bmatrix}\text{ 0.3109 0.3021)}$$

$$=\det\left(\begin{bmatrix}z-1.916+0.3109.Ke1 & 0.917+0.3021Ke1\\ -1+0.3109Ke2 & z+0.3021Ke2\end{bmatrix}\right)$$

=z^2+(0.3109.Ke_1+0.3021Ke_2-1.916)z+0.3021Ke_1+0.917-0.8639Ke_2

Cân bằng hai vế của phương trình đặc tính ta có hệ phương trình sau :

$$\begin{cases} 0.3109*Ke1+0.3021*Ke2-1.916=-1.8422 \\ 0.3021*Ke1-0.8639*Ke2+0.917=0.8503 \end{cases}$$

Giải ra ta có nghiệm :

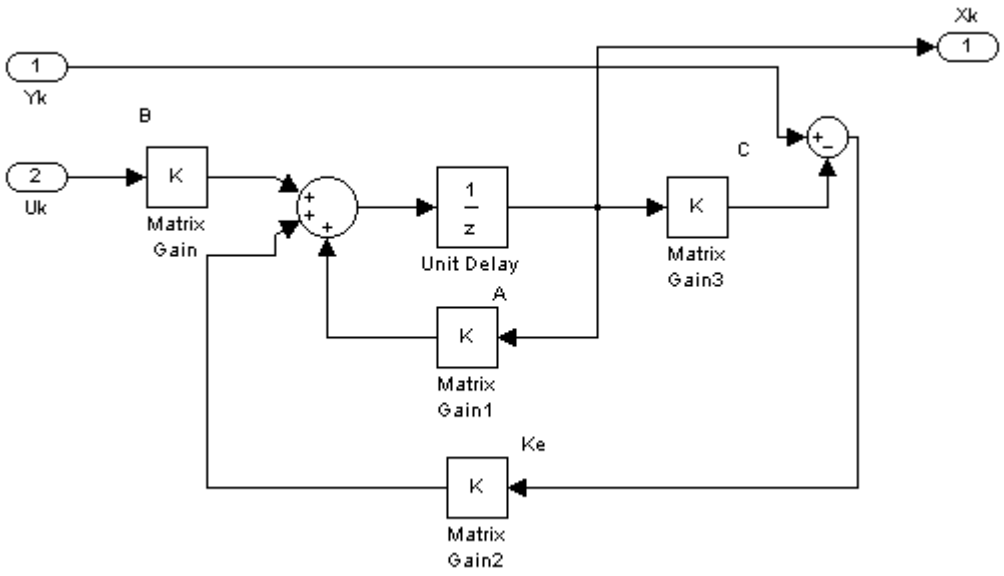
Ke1= 0.12117

Ke2= 0.11958

Đó chính là dạng tổng quát của Ke áp dụng cho bộ trạng thái.

$$Ke=\begin{bmatrix} 0.12117 \\ 0.11958 \end{bmatrix}$$

Để tiện lợi cho việc khảo sát và tổng hợp hệ thống sau này , ta sẽ xây dựng một khối con trên simulink chứa bộ ước lượng trạng thái và đặt mặt nạ cho nó rồi đưa vào thư viện. Sơ đồ cấu trúc của bộ ước lượng trạng thái trên simulink như hình vẽ sau :



2.3.2.5.Tổng hợp hệ thống dùng hồi tiếp trạng thái

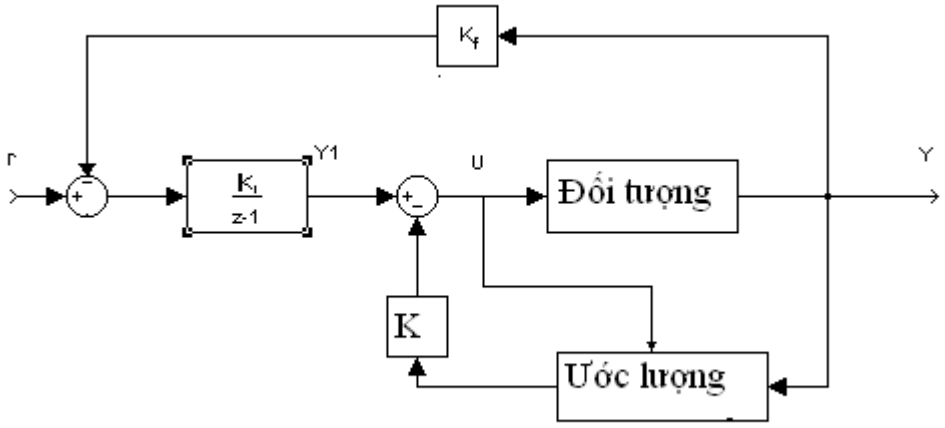
Sơ đồ khối của hệ thống có hồi tiếp trạng thái đã được trình bày trong phần 2 . Một thành phần được thêm vào sơ đồ trên là bộ ước lượng trạng thái.Tuy nhiên nếu thuần thuý chỉ hồi tiếp trạng thái qua bộ hồi tiếp K như vậy thì hệ thống sẽ có sai lệch tĩnh rất lớn.

Điều này được giải thích do trong đối tượng và trong cả hệ kín đều có khâu tích phân, ngoài ra đối tượng có hệ số khuyếch đại khá lớn.Không những thế khi có nhiễu tác động vào hệ thống (chủ yếu là nhiễu đối tượng) thì hệ thống sẽ bị ảnh hưởng rất lớn và sai lệch tĩnh sẽ có thể rất lớn.Để khắc phục được vấn đề này có hai giải pháp có thể áp dụng :

Một là: Dùng các khâu bù đầu vào và bù song song để giảm sai lệch tĩnh của hệ thống và hạn chế ảnh hưởng của nhiễu như hình dưới.Thực tế phương pháp này không thể triệt tiêu hẳn được sai lệch tĩnh và vẫn chịu ảnh hưởng không nhỏ của nhiễu.Sở dĩ như vậy là do khi thiết kế bộ bù ta không thể tính chính xác được các thông số và luôn có sự làm tròn và bản thân các thông số của đối tượng cũng có thể thay đổi trong quá trình hoạt động .
Hai là: Đưa thêm khâu tích phân vào vị trí thích hợp trong hệ thống.Khâu tích phân có khả năng triệt tiêu hoàn tòan được sai lệch tĩnh và giảm ảnh hưởng của nhiễu đến hệ thống.Với hệ thống xung –số khâu tích phân được lựa chọn có hàm truyền đạt như sau :

$$W1(z)=\frac{Ki}{z-1}$$

Sơ đồ khối của hệ thống đã được bù dùng khâu tích phân như sau :



Sau khi điểm qua hai pương pháp trên, ta thấy phương pháp thêm khâu tích phân có nhiều ưu điểm hơn, bởi vậy trong bài này ta sẽ dùng khâu tích phân .
Trước hết cần tìm phương trình trạng thái của khâu tích phân :

$$\frac{y1(z)}{u1(z)}=\frac{Ki}{z-1}$$

$$\Rightarrow\begin{cases} x1(k+1)=x1(k)+u1(k) \\ y1(k)=Ki*x1(k) \end{cases}$$

Ta thấy khâu tích phân này chỉ có một biến trạng thái duy nhất và đầu ra của khâu tích phân tỉ lệ với biến trạng thái .
Đặt $X(k)=\begin{bmatrix} x(k) \\ x1(k) \end{bmatrix}$ ta có:

u(k)=K1.x1(k)-K.x(k)=- $k \quad -ki$ .x(k)
u1(k)=- C.x(k)

$$\begin{cases} x(k+1) = A.x(k) + B.u(k) \\ x1(k+1) = x1(k) + u1(k) = -C.x(k) + x1(k) \end{cases}$$

$$x(k+1)=\begin{bmatrix} A & 0 \\ -C & 1 \end{bmatrix} x(k)+\begin{bmatrix} B \\ 0 \end{bmatrix} u(k)$$

$$u(k)=- k \quad -ki \; x(k)$$

$$\Rightarrow x(k+1)=\left(\begin{bmatrix} A & 0 \\ -C & 1 \end{bmatrix}-\begin{bmatrix} B \\ 0 \end{bmatrix} k \quad -ki \right) x(k)$$

Đặt e(k)=x(k)-x(∞) là sai lệch tĩnh ta có :

$$e(k+1)= \left(\begin{bmatrix} A & 0 \\ -C & 1 \end{bmatrix}-\begin{bmatrix} B \\ 0 \end{bmatrix} k \quad -ki \right) e(k)$$

Mục đích của ta là phải làm cho e(k) tiến đến 0 càng nhanh càng tốt để triệt tiêu được sai tĩnh .Đến đây ta lại thấy dạng quen thuộc của (4.22) và do đó có thể áp dụng phương pháp tìm bộ hồi tiếp trạng thái để tìm $k \quad -ki$.

Hệ mới bây giờ có bậc là 3, dùng bảng các điểm cực chuẩn của Bessel cho hệ bậc 3 và chọn thời gian quá độ T=500 (sec) ta có các điểm cực mới là:

$$p_{1}=\frac{-5.0093}{500}=-0.0100; p_{2,3}=\frac{-3.9668\pm 3.7845i}{500}=-0.0079\pm 0.0076i$$

$\Rightarrow zp_1=e^{10*1}= 0.9047;$

$zp_{2,3} =e^{10*p2,3}= 0.9211 \pm 0.0699i$

Phương trình đặc tính mới là :

$$\det\left(z.I-\begin{bmatrix} A & 0 \\ -C & 1 \end{bmatrix}+\begin{bmatrix} B \\ 0 \end{bmatrix} k \quad -ki \right)$$

$$=\det\left(\begin{bmatrix} z & 0 & 0 \\ 0 & z & 0 \\ 0 & 0 & z \end{bmatrix}-\begin{bmatrix} 1.916 & -0.917 & 0 \\ 1 & 0 & 0 \\ -0.3109 & -0.3021 & 1 \end{bmatrix}+\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} k1 \quad k2 \quad -ki \right)$$

$$=\det\begin{pmatrix} z-1.916+k1 & 0.917+k2 & -Ki \\ -1 & z & 0 \\ 0.3109 & 0.3021 & z-1 \end{pmatrix}$$

$=z^3 + z^2(k1-2.916) +z(0.3021*ki - k1 + k2 +2.833) +(0.3021*ki -0.917-k2)$

Mặt khác ta lại có :

$$\det\left(z.i - \begin{bmatrix} A & 0 \\ -C & 1 \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix}. K - Ki \right)$$

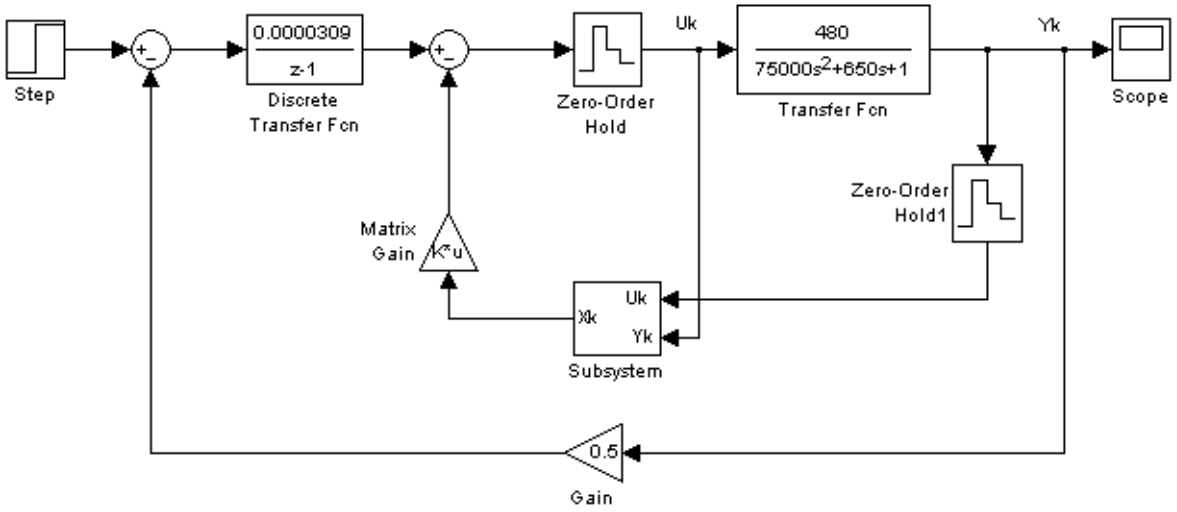
= (z-zp1).(z-zp2).(z-zp3)
= (z - 0.9047)*(z - 0.9211 + 0.0699i)*(z - 0.9211- 0.0699i)
$=z^3 - 2.747*z^2 + 2.52*z - 0.772$

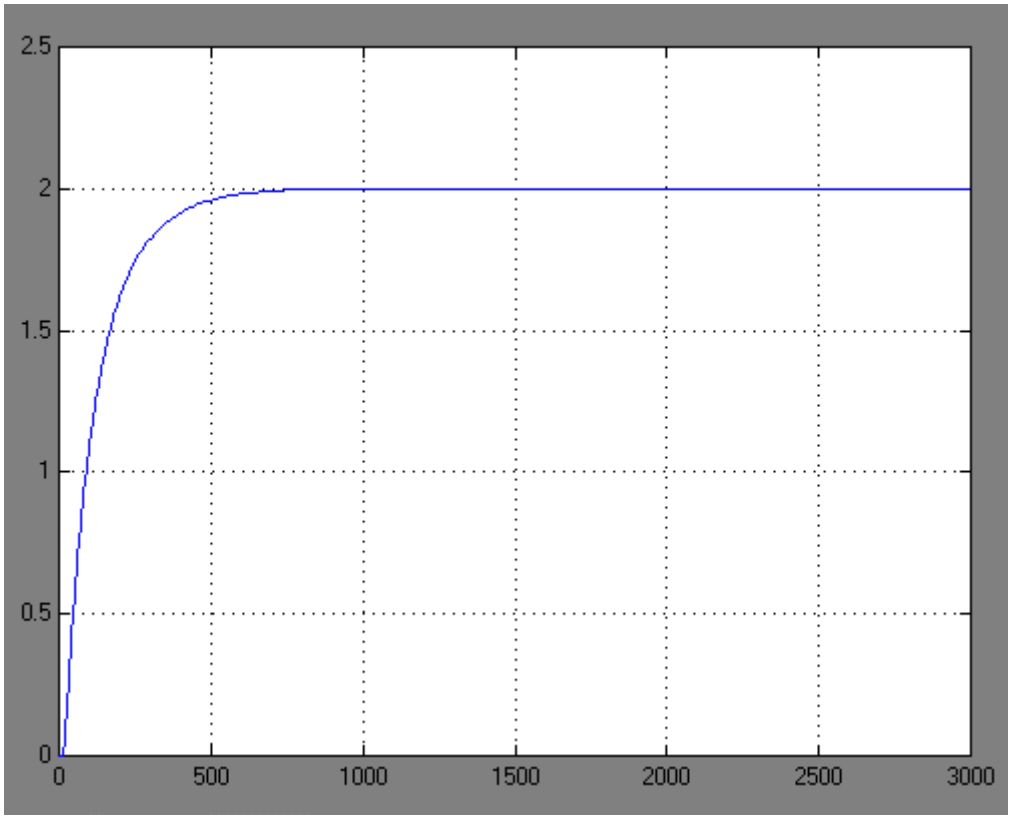
Cân bằng hai vế của phương trình trên ta được :

$$\begin{cases} k1 - 2.916 = -2.747 \\ 0.3021*ki - k1 + k2 + 2.833 = 2.52 \\ 0.3021*ki - 0.917 - k2 = -0.772 \end{cases} \Rightarrow \begin{cases} k1 = 0.169 \\ k2 = -0.14451 \\ ki = 0.00165 \end{cases}$$

vậy ta đã tìm được các hệ số cần thiết một cách tổng quát.Thực hiện trên sơ đồ simulink ta có sơ đồ sau :

Ta thu được qúa trình của hệ thống như sau:





Ta nhận thấy rằng sử dụng phương pháp phản hồi trạng thái chất lượng của hệ thống cũng rất tốt; thời gian quá độ là 700 (sec) độ quá điều chỉnh là 0,05% và không có sai lệch tĩnh.

2.3.2.6.So sánh hai bộ điều khiển tìm được

Qua so sánh kết quả mô phỏng với hai bộ điều khiển : bộ điều khiển PID và bộ phản hồi trạng thái ta có nhận xét sau :

-Cả hai phương pháp đều đem lại hệ thống có chất lượng rất tốt .

-Trong quá trình tổng hợp thì sử dụng bộ PID số đơn giản hơn .

-Do đối tượng điều khiển ở đây là lò sấy không có yêu cầu cao về công nghệ cũng như đòi hỏi sự chính xác tuyệt đối nên em quyết định chọn bộ điều khiển PID để xây dựng hệ thống điều khiển số cho hệ thống.

CHƯƠNG 3 :
THIẾT KẾ PHẦN MỀM

3.1. Thuật toán điều khiển của hệ thống

Phương trình sai phân của bộ điều khiển:

Như đã trình bày ở chương 4, bộ điều khiển PID có hàm truyền đạt như sau:

$$W_{pid}(z) = \frac{Az^2 + Bz + C}{z(z-1)}$$

trong đó :

$$A = \frac{(K_i T^2 + 2K_d + 2K_p T)}{2T}$$

$$B = \frac{(K_i T^2 - 2K_p T - 4K_d)}{2T}$$

$$C = \frac{K_d}{T}$$

Từ hàm truyền đạt này, chuyển thành phương trình sai phân:

$$W_{pid}(z) = \frac{U(z)}{E(z)} = \frac{Az^2 + Bz + C}{z^2 - z} = \frac{A + B.z^{-1} + C.z^{-2}}{1 - z^{-1}}$$

$$\Rightarrow E(z).(A + B.z^{-1} + C.z^{-2}) = U(z)(1 - z^{-1})$$

$$\Rightarrow A.e(k) + B.e(k-1) + C.e(k-2) = u(k) - u(k-1)$$

$$\Rightarrow u(k) = u(k-1) + A.e(k) + B.e(k-1) + C.e(k-2)$$

Từ phương trình sai phân trên có được thuật toán xây dựng bộ PID số như sau:

- Khai báo 1 biến để lưu giá trị cũ của *u* và 1 mảng để lưu các giá trị cũ của *e* :

  float u_old;

  float e_old[2];

  Các giá trị này đều được khởi đầu bằng 0.

  Tính A,B,C theo Kp, Ti, Td đã chọn.
- Các bước tính cho bộ PID số:
  - Đọc giá trị hồi tiếp về *y* và giá trị đặt *w*. Tính sai lệch *e = w - y*.
  - Tính giá trị tín hiệu điều khiển hiện thời:

    *u = u_old + A*e + B*e[0] + C*e[1];*
  - Xuất tín hiệu điều khiển ra.
  - Cập nhập lại các biến lưu:

    u_old = u;

    e[1]=e[0];

    e[0]=e;
  - Lặp lại từ đầu.

3.2.Phương án xây dựng chương trình điều khiển và giao diện:

Chương trình điều khiển và giao diện bao gồm hai phần độc lập: phần giao diện người dùng và phần thực hiện thuật toán điều khiển.

- Phần giao diện người dùng:
  - Hiển thị kết quả quá trình điều khiển
  - Nhận nhiệt độ đặt mong muốn từ người dùng.
  - Nhận các thông số thiết lập cho hệ thống từ người dùng.
  - Yêu cầu của phần này là giao diện phải dễ dùng, thuận tiện cho người sử dụng. Giao diện phải hợp lý. Một phong cách chung trong giao diện điều khiển là mô phỏng các bàn điều khiển thiết bị (Instrument Panel).
- Phần thực hiện thuật toán điều khiển:
  - Thực hiện thuật toán điều khiển (như đã trình bày ở phần trên).
  - Giao tiếp với phần cứng để điều khiển đối tượng cũng như nhận các kết quả đo đạc từ các Sensor.
  - Nhận thông số hệ thống và giá trị đặt từ người dùng thông qua phần giao diện.
  - Cung cấp các số liệu cho phần giao diện (nhieọt ủoọ)
  - Yêu cầu của phần này laứ việc thực hiện thuật toán điều khiển phải chính xác

Căn cứ vào các yêu cầu trên, để thuận tiện cho phần giao diện, chương trình sẽ được viết trên môi trường Windows. Môi trường hệ điều hành Windows là một môi trường đồ hoạ hoàn thiện, cung cấp rất nhiều các công cụ phát triển cũng như các thành phần đồ hoạ cơ bản (menu, cửa sổ, hộp thoại, ...) giúp cho việc phát triển các ứng dụng được dễ dàng và nhanh chóng. Mặc dù môi trường Windows luôn hạn chế việc truy nhập cấp thấp với phần cứng, tuy nhiên trên Windows 9x thì việc thực hiện các vào ra cơ bản với phần cứng là được phép. Điều này hoàn toàn đáp ứng được yêu cầu trong bài tập này.

Chọn công cụ lập trình:

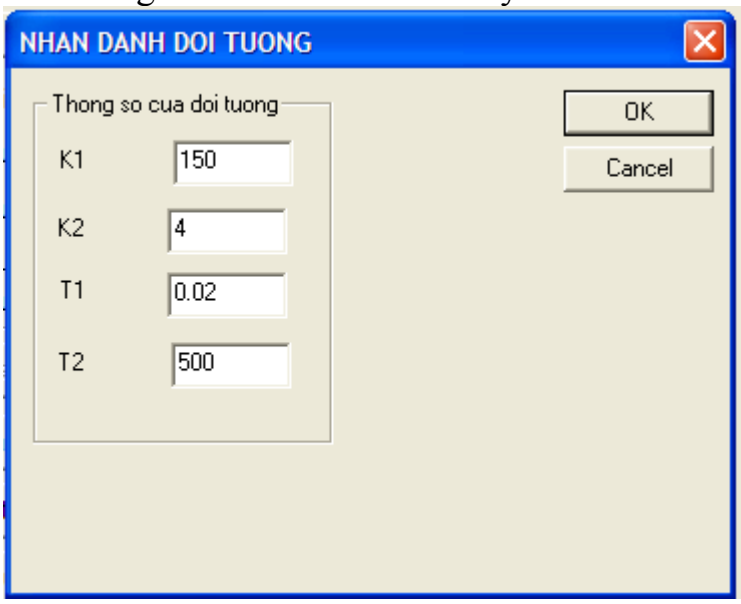
Với các bài toán điều khiển, ngôn ngữ C/C++ dường như là lựa chọn bắt buộc, bởi ngôn ngữ này cho phép viết các chương trình mạnh, nhanh, nhỏ gọn, truy nhập sâu vào phần cứng (về mặt này thì không bằng hợp ngữ). Tuy nhiên, để tối ưu phần mã điều khiển, một số đoạn trình hợp ngữ sẽ được sử dụng thêm vào đoạn mã C/C++.

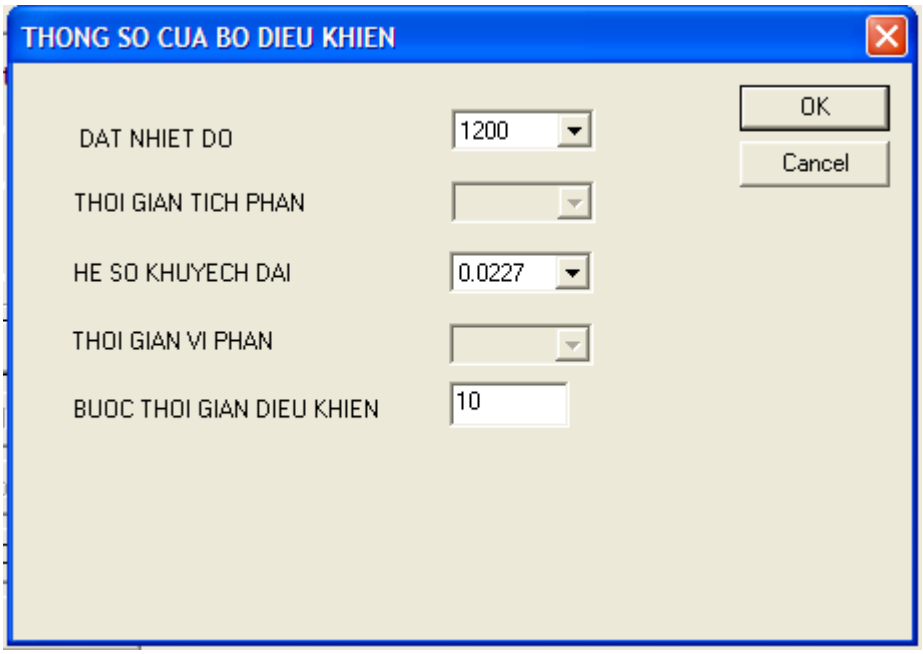
Về công cụ và mội trường phát triển: hiện nay có hai công cụ phát triển rất mạnh dùng ngôn ngữ C/C++, đó là Visual C++ của Microsoft và CBuilder của Inprise (tên mới của Borland). Trong khi Visual C++ ưu tiên khả năng can thiệp sâu vào hệ thống và chỉ đóng gói đơn giản các thành phần (đồ hoạ, file,...) của hệ thống thì CBuilder lại tận dụng tối đa khả năng hướng đối tượng trong C++ để đóng gói các thành phần hệ thống, giúp lập trình viên càng ít phải can thiệp chi tiết vào hệ thống càng tốt. Tất nhiên CBuilder sẽ có chút hạn chế khi lập trình viên muốn lập trình cấp thấp. Với mục đích nhanh chóng tạo ra chương trình với giao

diện phù hợp, dễ sử dụng mà vẫn đảm bảo yêu cầu về tốc độ, tính hiệu quả, công cụ CBuilder được chọn để thực hiện phần mềm cho bài tập này.

3.3.Kết quả chạy chương trình:

Chương trình được dịch thành công và chạy thử. Vì không có hệ thống điều khiển thực tế (card ghép nối, biến tần, động cơ,...) nên chỉ thử được giao diện. Giao diện của chương trình như hình dưới đây.






he thong dieu khien so
Time :
14:15:24
Mo phong nhiet do lo
luat dieu khien
Luat P
Luat PI
Luat PD
Luat PID
Bat dau
Nhan dang
Chon thong so
Dieu khien
Dung
Thoat


he thong dieu khien so
Mo phong nhiet do lo
luat dieu khien
Luat P
Luat PI
Luat PD
Luat PID
Bat dau
Nhan dang
Chon thong so
Dieu khien
Dung
Thoat

3.4.Mã nguồn chương trình:

```
// BTDDlg.cpp : implementation file
//
#include "stdafx.h"
#include "BTD.h"
#include "BTDDlg.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About
class CAboutDlg : public CDialog
{
public:
    CAboutDlg();
// Dialog Data
    //{{AFX_DATA(CAboutDlg)
    enum { IDD = IDD_ABOUTBOX };
    //}}AFX_DATA
    // ClassWizard generated virtual function overrides
    //{{AFX_VIRTUAL(CAboutDlg)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV
support
    //}}AFX_VIRTUAL
// Implementation
protected:
    //{{AFX_MSG(CAboutDlg)
    //}}AFX_MSG
    DECLARE_MESSAGE_MAP()
};
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
    //{{AFX_DATA_INIT(CAboutDlg)
    //}}AFX_DATA_INIT
}
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CAboutDlg)
    //}}AFX_DATA_MAP
```

```
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
    //{{AFX_MSG_MAP(CAboutDlg)
        // No message handlers
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
////////////////////////////////////////////////////////////////////////////
// CBTDDlg dialog
CBTDDlg::CBTDDlg(CWnd* pParent /*=NULL*/)
    : CDialog(CBTDDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CBTDDlg)
    m_TT = _T("");
    m_sTime = _T("");
    m_Radio = -1;
    //}}AFX_DATA_INIT
    // Note that LoadIcon does not require a subsequent DestroyIcon in Win32
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}
void CBTDDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CBTDDlg)
    DDX_Control(pDX, IDC_BTHONGSO, m_thongso);
    DDX_Control(pDX, IDC_BTHOAT, m_thoat);
    DDX_Control(pDX, IDC_BSTOP, m_dung);
    DDX_Control(pDX, IDC_BDIEUKHIEN, m_dieukhien);
    DDX_Control(pDX, IDC_BNHANDANG, m_nhandang);
    DDX_Control(pDX, IDC_BSTART, m_batdau);
    DDX_Text(pDX, IDC_TIME, m_TT);
    DDX_Text(pDX, IDC_STATICTIME, m_sTime);
    DDX_Radio(pDX, IDC_RALP, m_Radio);
    //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CBTDDlg, CDialog)
    //{{AFX_MSG_MAP(CBTDDlg)
    ON_WM_SYSCOMMAND()
    ON_WM_PAINT()
    ON_WM_QUERYDRAGICON()
    ON_WM_TIMER()
    ON_BN_CLICKED(IDC_BSTART, OnBstart)
    ON_BN_CLICKED(IDC_BNHANDANG, OnBnhandang)
```

```
    ON_BN_CLICKED(IDC_BTHONGSO, OnBthongso)
    ON_BN_CLICKED(IDC_BDIEUKHIEN, OnBdieukhien)
    ON_BN_CLICKED(IDC_BSTOP, OnBstop)
    ON_BN_CLICKED(IDC_BTHOAT, OnBthoat)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CBTDDlg message handlers
BOOL CBTDDlg::OnInitDialog()
{
    CDialog::OnInitDialog();
    // Add "About..." menu item to system menu.
    // IDM_ABOUTBOX must be in the system command range.
    ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
    ASSERT(IDM_ABOUTBOX < 0xF000);
    CMenu* pSysMenu = GetSystemMenu(FALSE);
    if (pSysMenu != NULL)
    {
        CString strAboutMenu;
        strAboutMenu.LoadString(IDS_ABOUTBOX);
        if (!strAboutMenu.IsEmpty())
        {
            pSysMenu->AppendMenu(MF_SEPARATOR);
            pSysMenu->AppendMenu(MF_STRING,  IDM_ABOUTBOX,
strAboutMenu);
        }
    }
    // Set the icon for this dialog.  The framework does this automatically
    //  when the application's main window is not a dialog
    SetIcon(m_hIcon, TRUE);                    // Set big icon
    SetIcon(m_hIcon, FALSE);            // Set small icon
        CRect rectWin;
    GetClientRect(rectWin);
    CClientDC dc(this);
    m_pdcMemory.CreateCompatibleDC(&dc);
    m_pBitmap.CreateCompatibleBitmap(&dc,rectWin.Width(),rectWin.Height
());
    m_t=FALSE;
    NhanDang[0]=m_DoiTuong.m_kdt;
    NhanDang[1]=m_DoiTuong.m_Tdt;
    NhanDang[2]=m_DoiTuong.m_Ttdt;
    NhanDang[7]=0.1;
    NhanDang[6]=50;
```

```
        NhanDang[5]=0;
        NhanDang[4]=0;
        NhanDang[3]=23;
        m_Radio=0;
        UpdateData(FALSE);
        for(int i=0;i<600;i++)
        {
                point[i].x=0;
                point[i].y=0;
        }
        // TODO: Add extra initialization here
        return TRUE;  // return TRUE  unless you set the focus to a control
    }
    void CBTDDlg::OnSysCommand(UINT nID, LPARAM lParam)
    {
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
        }
        else
        {
                CDialog::OnSysCommand(nID, lParam);
        }
    }
    // If you add a minimize button to your dialog, you will need the code below
    //  to draw the icon.  For MFC applications using the document/view model,
    //  this is automatically done for you by the framework.
    void CBTDDlg::OnPaint()
    {
        if (IsIconic())
        {
                CPaintDC dc(this); // device context for painting
                SendMessage(WM_ICONERASEBKGND,                (WPARAM)
dc.GetSafeHdc(), 0);
                // Center icon in client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;
                // Draw the icon
```

```
            dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
            CDialog::OnPaint();
        }
    }
    // The system calls this to obtain the cursor to display while the user drags
    //  the minimized window.
    HCURSOR CBTDDlg::OnQueryDragIcon()
    {
        return (HCURSOR) m_hIcon;
    }
    void CBTDDlg::OnTimer(UINT nIDEvent)
    {
        // TODO: Add your message handler code here and/or call default
            CTime curTime=CTime::GetCurrentTime();//de lam gi
        switch(nIDEvent)
        {
        case ID_CLOCK_TIME:
        m_sTime.Format("%d:%d:%d",curTime.GetHour(),curTime.GetMinute(),cu
rTime.GetSecond());
            UpdateData(FALSE);
            break;
        case ID_CLOCK_COUNT:
            double adl=(double)(inp_b1(1));
            ad=600*adl/225;
            point[m_count].x=(long)(270+5*m_count);
            point[m_count].y=(long)(400-ad*23/60);
            e[m_count]=(NhanDang[6]*225/500)-adl;
            unsigned short m_out;
            if(e[m_count]<0)
                e[m_count]=1;
            m_out=(unsigned short)(e[m_count]);
            _asm{
                push ax
                push dx
                mov ax,m_out
                mov dx,30Ah
                out dx,al
                pop dx
                pop ax
            }
```

```
            for(int i=0;i<40000;i++);
            InvalidateRect(CRect(245,120,605,440),FALSE);
            m_count ++;
            break;
        }
        CDialog::OnTimer(nIDEvent);
        CDialog::OnTimer(nIDEvent);
    }

    unsigned short CBTDDlg::inp_b1(unsigned short port)
    {
        _asm{
            push ax;
            push dx;
            mov ax,port;
            mov dx,3F8h;
            out dx,al;
            pop dx;
            pop ax;
        }
        for(int i=0;i<40000;i++);
        unsigned short c;
        _asm{
            push ax;
            push dx;
            xor ax,ax;
            mov dx,309h;
            in al,dx;
            mov c,ax;
            pop dx;
            pop ax;
        }
        return c;
    }
    void CBTDDlg::OnBstart()
    {
        // TODO: Add your control notification handler code here
        GetDlgItem(IDC_RALP)->EnableWindow(TRUE);
        GetDlgItem(IDC_RALPI)->EnableWindow(TRUE);
        GetDlgItem(IDC_RALPD)->EnableWindow(TRUE);
        GetDlgItem(IDC_RALPID)->EnableWindow(TRUE);
        m_batdau.EnableWindow(FALSE);
        m_dung.EnableWindow(FALSE);
```

```
        m_nhandang.EnableWindow(TRUE);
        m_thongso.EnableWindow(TRUE);
        m_dieukhien.EnableWindow(TRUE);
        m_thoat.EnableWindow(TRUE);
    }
    void CBTDDlg::OnBnhandang()
    {
        if(m_DoiTuong.DoModal()==IDOK)
        {
                NhanDang[0]=m_DoiTuong.m_kdt;
                NhanDang[1]=m_DoiTuong.m_Tdt;
                NhanDang[2]=m_DoiTuong.m_Ttdt;
        }
        //InvalidateRect(CRect(126,115,230,440),FALSE);
        m_nhandang.EnableWindow(FALSE);
        m_thongso.EnableWindow(TRUE);
        m_dung.EnableWindow(TRUE);
        m_dieukhien.EnableWindow(TRUE);
        m_thoat.EnableWindow(TRUE);// TODO: Add your control notification
handler code here
    }
    void CBTDDlg::OnBthongso()
    {
        // TODO: Add your control notification handler code here
        UpdateData(TRUE);
        m_thongSo.m_chot=m_Radio;
        m_thongso.EnableWindow(FALSE);
        m_nhandang.EnableWindow(FALSE);
        m_dieukhien.EnableWindow(TRUE);
        m_thoat.EnableWindow(TRUE);
        m_dung.EnableWindow(TRUE);
        if(m_thongSo.DoModal()==IDOK)
        {
                NhanDang[3]=m_thongSo.m_dt[0];
                NhanDang[4]=m_thongSo.m_dt[1];
                NhanDang[5]=m_thongSo.m_dt[2];
                NhanDang[6]=m_thongSo.m_dt[3];
                NhanDang[7]=m_thongSo.m_step;
        }
        InvalidateRect(CRect(126,115,230,440),FALSE);
    }
    void CBTDDlg::OnBdieukhien()
    {
```

```
    // TODO: Add your control notification handler code here
    SetTimer(ID_CLOCK_TIME,1000,NULL);
    SetTimer(ID_CLOCK_COUNT,3000,NULL);
    m_t=TRUE;
    e[0]=NhanDang[6];
    e[1]=NhanDang[6];
    m_count=2;
    m_TT="Time :";
    m_batdau.EnableWindow(FALSE);
    m_dieukhien.EnableWindow(FALSE);
    m_thongso.EnableWindow(FALSE);
    m_nhandang.EnableWindow(FALSE);
    m_dung.EnableWindow(TRUE);
    m_thoat.EnableWindow(TRUE);
    UpdateData(FALSE);
}
void CBTDDlg::OnBstop()
{
    // TODO: Add your control notification handler code here
    KillTimer(ID_CLOCK_TIME);
    KillTimer(ID_CLOCK_COUNT);
    m_t=FALSE;
    m_dung.EnableWindow(FALSE);
    m_thongso.EnableWindow(FALSE);
    m_dieukhien.EnableWindow(FALSE);
    m_nhandang.EnableWindow(FALSE);
    m_batdau.EnableWindow(TRUE);
    m_thoat.EnableWindow(TRUE);
    GetDlgItem(IDC_RALP)->EnableWindow(FALSE);
    GetDlgItem(IDC_RALPI)->EnableWindow(FALSE);
    GetDlgItem(IDC_RALPID)->EnableWindow(FALSE);
    InvalidateRect(CRect(240,120,605,440),FALSE);
    m_TT="";
    m_sTime="";
    UpdateData(FALSE);
}
void CBTDDlg::OnBthoat()
{
    // TODO: Add your control notification handler code here
        OnOK();
}
```

KẾT LUẬN

Sau một thời gian thực hiện đồ án tốt nghiệp với nhiều cố gắng và nổ lực cùng với sự tận tình hướng dẫn của thầy giáo**Th.s Nguyễn Trọng Thắng**, quyển đồ án này đã hoàn thành đúng thời gian qui định theo yêu cầu đặt ra là thiết kế va phân tích hệ thống điều khiển nhiệt độ cho lò đốt CN dùng họ vi điều khiển MSC-51,cụ thể là 80C51.

Để thực hiện được yêu cầu trên em đã nghiên cứu, tìm hiểu những vấn đế về vi điều khiển, vi xử lí, các phương pháp đo nhiệt độ, các phương pháp chuyển đổi từ tương tự sang số ,các phương tổng hợp hệ thống điều khiển và các vấn đề khác có liên quan đến đề tài.

Nội dung chính của đề tài này bao gồm những phần chính sau:

*Phần kiến thức.

-Khảo sát bộ vi điều khiển 8051/8031.

-Khảo sát IC giai tiếp ngoại vi 8255A.

-Khảo sát các bộ nhớ thông dụng .

-Các phương pháp chuyển đổi từ tương tự sang số.

-Các IC phụ trợ 74LS138,74LS373…

-Hệ thống đo nhiệt độ và các phương pháp đo nhiệt độ.

*Phần thiết kế thi công .

-Xây dựng sơ đồ khối toàn mạch .

-Xây dưng lưu đồ giải thuật .

-Viết chương trình.

Trên đây là những nội dung mà em đã thực hiện được trong đồ án này.

Theo nhận định chủ quan của em thi quyển đồ án này đã trình bày tương đối đầy đủ các nội dung, những kiến thức liên quan, giải quyết được những yêu cầu đặt ra.

Tuy nhiên do thời gian cũng như trình độ chuyên môn có hạn vẫn còn nhiều thiếu sót .

Để đề tài này thêm phong phú và tăng hiệu quả sử dụng thì cần đáp ứng được những yêu cầu sau:

+Phải hoàn thiện và nghiên cứu sâu về lập trình cho 80C51

+Dùng các phần mềm chuyên dụng như Orcard,Protel để vẽ mạch in,tiến tới lắp ráp thành card thật,tiến hành chạy thử.

Đó là những yêu cầu mà em chưa có điều kiện thực hiện do hạn chế về thời gian, cũng như gặp những khó khăn về kĩ thuật.

Xin chân thành cảm ơn các bạn sinh viên đã đóng góp những ý kiến quý báu để

đề tài này hoàn thành tốt đẹp, đúng thời hạn.

TÀI LIỆU THAM KHẢO

1.Lý thuyết điều khiển tự động-Phạm Công Ngô- NXB khoa học kỹ thuật
2.Kỹ thuật vi điều khiển-Lê văn Doanh-Phạm Khắc Chương-NXB khoa học kỹ thuật.
3. Đo lường và điều khiển bằng máy tính –Ngô Diên Tập-NXB khoa học kỹ thuật.
4.Họ vi điều khiển 8051-Tống Văn On-Hoàng Đức Hải-NXB Lao Động Xã Hội.
5.The 8051 Microcontroller –University of Guelph-I.SCOTT MACKENZIE.