ĐẠI HỌC BÁCH KHOA THÀNH PHỐ HỒ CHÍ MINH
KHOA ĐIỆN – ĐIỆN TỬ, BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG
PHÒNG THÍ NGHIỆM TỰ ĐỘNG HÓA CÔNG NGHIỆP

HƯỚNG DẪN THÍ NGHIỆM

Bài 5

SỬ DỤNG CARD PCL818L ĐIỀU KHIỂN NHIỆT ĐỘ

I. MỤC ĐÍCH

Bài thí nghiệm cung cấp cho sinh viên kiến thức về cách thức lập trình điều khiển sử dụng Card PCL818L và ứng dụng giải thuật điều khiển PID vào hệ thống điều khiển nhiệt độ. Sinh viên sẽ được hướng dẫn lập trình các thí nghiệm sau: + Điều khiển các ngõ Digital.

+ Đọc dữ liệu ngõ vào Analog.

+ Điều khiển nhiệt độ theo giải thuật PID số.

II. CHUẨN BỊ THÍ NGHIỆM

Để thực hiện được thí nghiệm sinh viên cần phải biết trước lập trình Visual Basic cơ bản và đọc các mục theo thứ tự sau đây:

+ *Nội dung thí nghiệm*: Đây là các yêu cầu phải thực hiện trong suốt buổi thí nghiệm. Sinh viên cần đọc trước mục này để nắm được nội dung thí nghiệm.

+ *Giới thiệu Card PCL818L*: Phần này trình bày cấu trúc phần cứng, cách thức cài đặt và sử dụng Driver kèm theo để điều khiển cho Card.

+ *Giải thuật điều khiển PID số.*

+ *Thiết bị thí nghiệm*

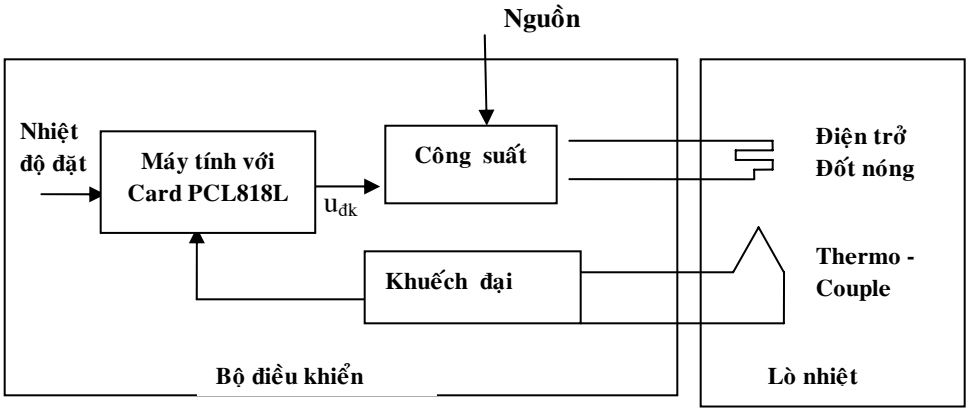
+ *Hướng dẫn thí nghiệm:* Phần này hướng dẫn chi tiết các bước để thực hiện được những yêu cầu trong *Nội dung thí nghiệm*.

Để đạt được kết quả thí nghiệm tốt và để hiểu rõ hơn sinh viên có thể xem các sơ đồ mạch điện phục vụ cho bài thí nghiệm ở phần phụ lục, các tài liệu về Card PCL818L cũng như các vấn đề liên quan trong môn học Đo Lường Điều Khiển Bằng Máy Tính và Điều Khiển Tự Động 1.

Sinh viên cần viết chương trình ở nhà trước khi tiến hành thí nghiệm.

III. THIẾT BỊ THÍ NGHIỆM

Thiết bị thí nghiệm được xây dựng theo sơ đồ khối sau:



Bộ điều khiển bao gồm Card PCL 818L gắn trên rãnh cắm ISA của máy tính và chương trình điều khiển. Nhiệt độ của lò được nhận biết thông qua cảm biến ThermoCouple được đọc về máy tính sau khi qua mạch khuếch đại. Chương trình máy tính xuất tín hiệu điều khiển theo giải thuật PID dựa vào sai lệch giữa nhiệt độ đặt và nhiệt độ đo. Tín hiệu điều khiển được đưa đến mạch công suất dùng triac để đóng ngắt nguồn điện cung cấp cho lò (lò điện là loại lò nướng dân dụng 220V đã được loại bỏ phần điều khiển nhiệt độ).

Máy tính sử dụng trong thí nghiệm đã được cài Driver dùng để điều khiển cho Card.

Trước khi tiến hành thí nghiệm, sinh viên cắm dây nối vào hai Connector trên hộp điều khiển để nối với máy tính như hình sau:

IV. NỘI DUNG THÍ NGHIỆM

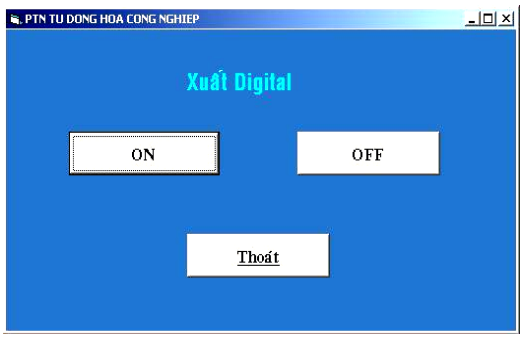
Dùng Visual Basic tạo một Form chính bao gồm ba nút chức năng : Xuất Digital, Đọc AD, Điều khiển nhiệt độ vòng kín. Khi nhấn các nút này thì hiện ra Form chức năng tương ứng. Có thể tạo Form chính như sau:

(Không cần tạo thêm nút nhấn "Khảo sát vòng hở" như Form ở trên)

Lưu ý: Tất cả các Form trình bày trong bài thí nghiệm này chỉ mang tính tham khảo. Sinh viên có thể thiết kế giao diện tuỳ ý sao cho đảm bảo các yêu cầu của thí nghiệm.

Thí nghiệm 1: Xuất dữ liệu ra ngõ Digital

Viết chương trình khi nhấn nút On thì xuất mức 1, nhấn nút OFF thì xuất mức 0 ra bit thứ 1 hoặc bit thứ 2 của Port 0.



Có thể quan sát kết quả hiển thị của hai đèn Led trên hộp điều khiển (đã được đấu nối tương ứng với hai bit này).

Nhấn nút Thoát thì thoát khỏi Form này và trở về cửa sổ của Form chính.

Thí nghiệm 2: Đọc dữ liệu ngõ vào Analog

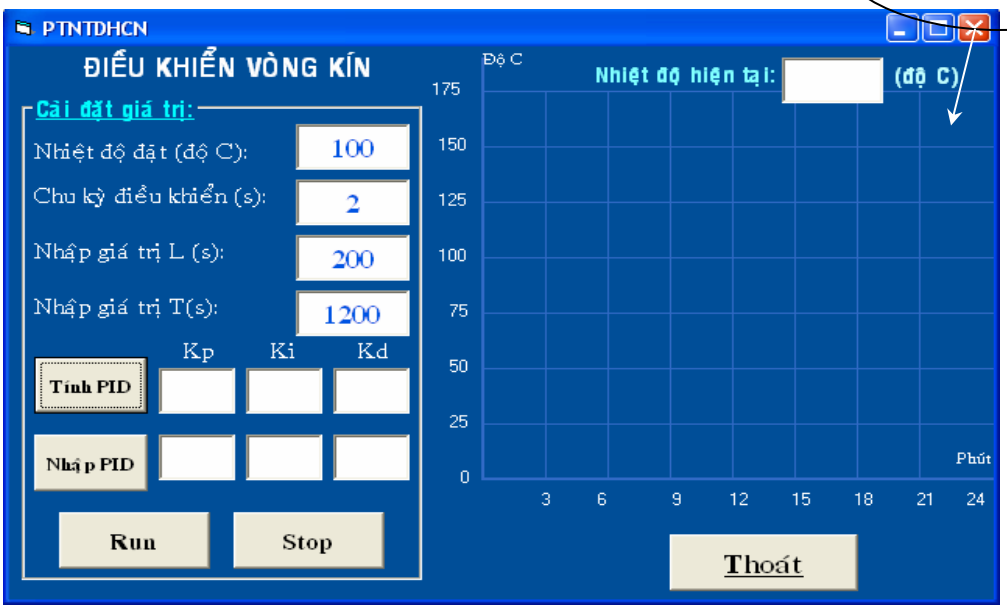
Viết chương trình khi nhấn nút Đọc AD thì cứ sau 200ms đọc giá trị điện áp ở Kênh 0, cập nhật và hiển thị lên ô TextBox . Giá trị điện áp ở Kênh 0 chính là giá trị điện áp đọc về của ThermoCouple sau khi qua mạch khuếch đại (phần này đã được đấu nối sẵn trong hộp thí nghiệm).

Khi nhấn nút Dừng đọc AD thì ngưng việc đọc và cập nhật giá trị AD. Nhấn nút Thoát thì thoát khỏi Form này và trở về cửa sổ của Form chính.

Thí nghiệm 3: Điều khiển nhiệt độ theo giải thuật PID số

Phần vẽ đồ thị chỉ thực hiện khi báo cáo thí nghiệm

Tạo các nút chức năng như Form sau:



Với các thông số lò nhiệt theo mô hình Ziegler-Nichols có được từ khảo sát hệ hở: + Thời hằng T=1200s

+ Khâu trễ thời gian L=200s

Viết chương trình theo các yêu cầu sau:

- Nhấn nút Tính PID thì tính và hiển thị lên ô TextBox các giá trị Kp, Ki, Kd theo công thức sau:

$$Kp = \frac{T}{2L}\ ; \qquad Ki = \frac{T}{4L^2}\ ; \qquad Kd = \frac{T}{4}$$

Nhập lại giá trị này vào các ô bên dưới tuỳ theo bộ điều khiển sử dụng:

+ Nếu là bộ điều khiển PID, nhập cả ba giá trị

+ Nếu là bộ điều khiển P chỉ cần nhập Kp (nhập Ki = Kd =0)

+ Nếu là bộ điều khiển PI , nhập Kp, Ki (Kd nhập là 0)

- Nhấn nút Nhập PID để xác nhận các giá trị này.
- Nhấn nút Run thì chạy chương trình điều khiển lò nhiệt theo giải thuật PID với luật điều khiển:

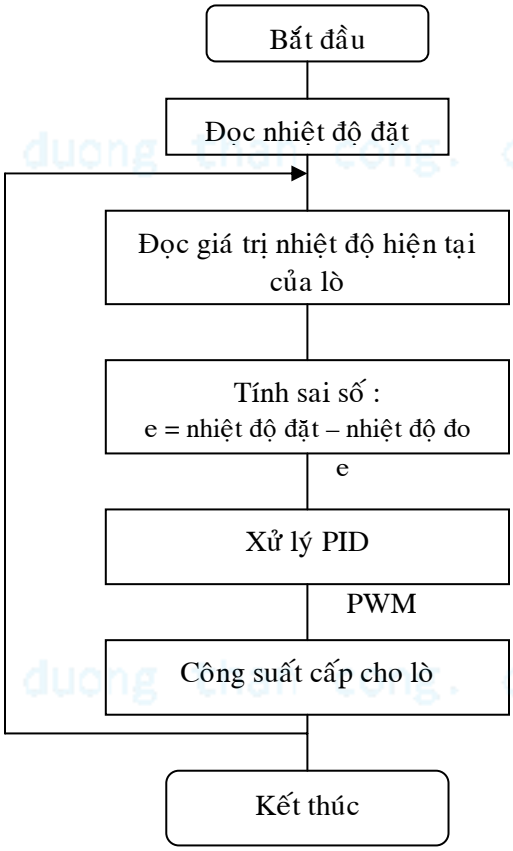
$$u(kT) = e(kT)\left(Kp + \frac{Kd}{T} + \frac{KiT}{2}\right) - e[(k-1)T]\left(Kp + \frac{2Kd}{T} - \frac{KiT}{2}\right) + \frac{Kd}{T}e[(k-2)T] + u[(K-1)T]$$

Trong đó:

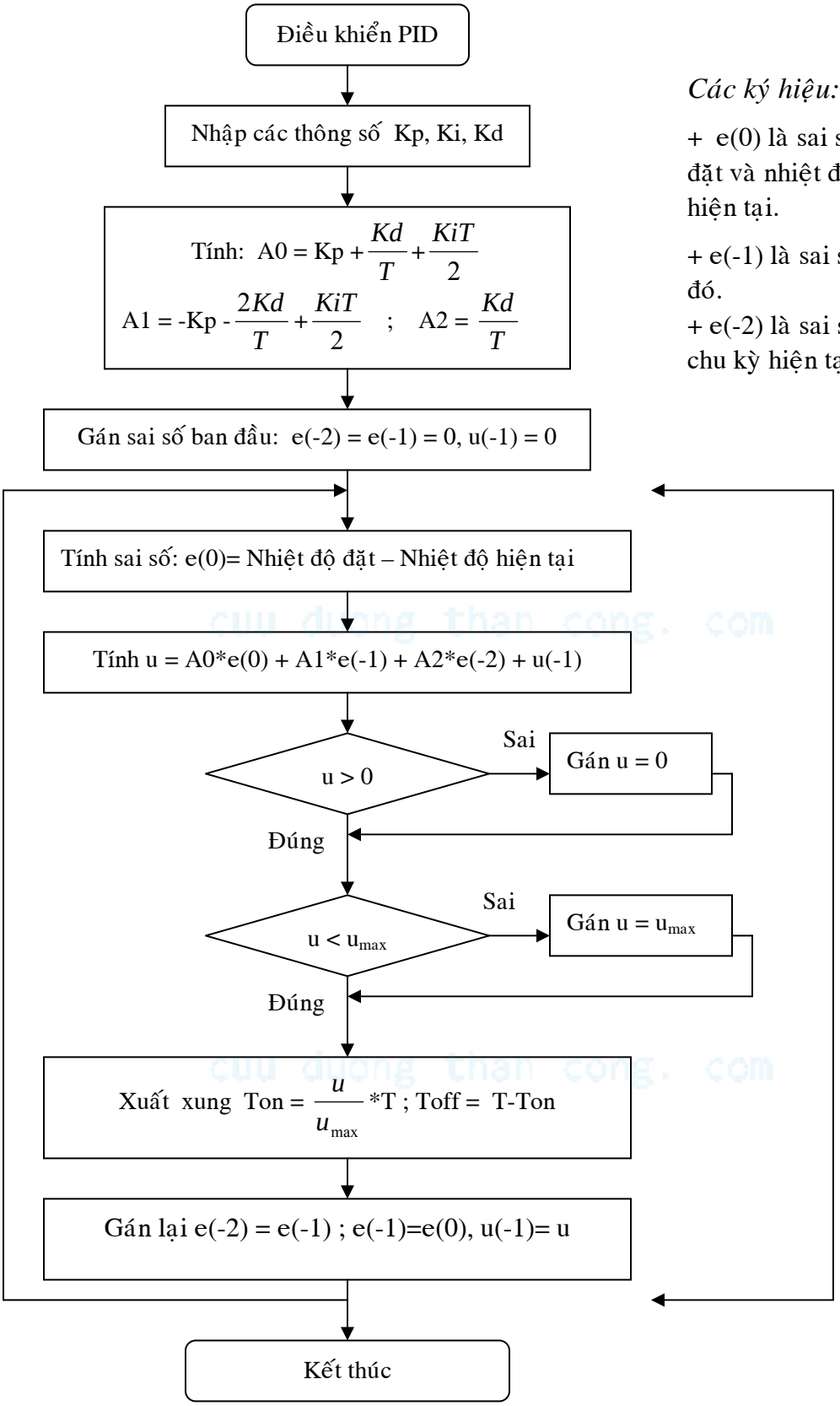
+ T là chu kỳ điều khiển chọn trước, ở đây là 2s.
+ u(kT) là tín hiệu điều khiển ở chu kỳ thứ điều khiển thứ k
+ u[(k-1)T] là tín hiệu điều khiển ở chu kỳ điều khiển thứ k-1
+ e(kT) là sai số giữa nhiệt độ đặt và nhiệt độ đo ở chu kỳ điều khiển thứ k
+ e(k-1) là sai số ở chu kỳ thứ k-1
+ e(k-2) là sai số ở chu kỳ điều khiển thứ k-2

- Nhấn nút Stop thì dừng việc điều khiển lại.

Lưu đồgiải thuật chương trình:



Lưu đồ giải thuật của chương trình xử lý PID như sau:



Các ký hiệu:

+ e(0) là sai số giữa nhiệt độ đặt và nhiệt độ đo ở chu kỳ hiện tại.

+ e(-1) là sai số ở chu kỳ trước đó.

+ e(-2) là sai số ở chu kỳ trước chu kỳ hiện tại hai chu kỳ.

Một chu kỳ điều khiển

Lưu ý: U_{max} có thể được tính như trong phần Hướng dẫn thí nghiệm.

Trong quá trình điều khiển , nhiệt độ hiện tại của lò phải luôn được hiển thị lên ô TextBox trên màn hình.

V. HƯỚNG DẪN THÍ NGHIỆM

- Chạy chương trình Visual Basic, chọn New Project→ Standard EXE→ Nhấn Open.
- Để sử dụng được Driver có sẵn của Card, vào menu Project → chọn Add, tìm đường dẫn đến File Driver.bas như sau: Add File…\ Programs Files\ Advantech\ Adsapi\ include\ Driver.bas (máy tính đã cài sẵn Driver ở đường dẫn này).
- Thiết kế các nút chức năng như trong yêu cầu của phần *Nội dung thí nghiệm,* nhấp đôi chuột vào từng nút nhấn để viết mã lệnh cho nút đó.
- Một số hàm thường dùng trong thiết kế giao diện:

  Form x.Show : Hiển thị Form x khi nhấp chuột vào một nút nhấn.

  Unload Me : Thoát khỏi Form hiện tại.

  End : Kết thúc chương trình.

  ...

- Sử dụng hàm DRV_DeviceOpen(DeviceNum, DriverHandle) để mở thiết bị (Card) khi điều khiển và hàm DRV_DeviceClose(DriverHandle) để đóng thiết bị khikhông cần điều khiển nữa. Cách thức thực hiện như sau:

  Mở thiết bị

  Thiết bị được mở khi bắt đầu chạy chương trình

  Khai báo các biến:

```
Option Explicit
    Dim status As Long              'Biến này lưu trữ trạng thái thực hiện lệnh.
    Dim Loi As String * 80          ' Lưu lỗi
    Dim DriverHandle As Long'DriverHandle làtên biến dùng để điều khiển Card
    Dim Response As String      'Báolỗi
```


  Nhấp đôi chuột vào Form cần điều khiển Card, viết đoạn mã sau:

```
Private Sub Form_Load()
    status = DRV_DeviceOpen(0, DriverHandle)    'Mở thiết bị số 0 là card PCL đã cài đặt bằng chương trình Devices Installation
    If (status <> 0) Then
```

```
    DRV_GetErrorMessage status, Loi
    Response = MsgBox(Loi, vbOKOnly, "Error")
  End If
End Sub
```

Đóng thiết bị

Thiết bị được đóng khi kết thúc chương trình

Nhấp đôi chuột vào nút:

để viết mãchương trình như sau:

```
Private Sub CommandExit_Click()
  status = DRV_DeviceClose(DriverHandle)  'Đóng thiết bị
  If (status <> 0) Then
    DRV_GetErrorMessage status, Loi
    Response = MsgBox(Loi, vbOKOnly,"Error") 'Thực hiện việc báo lỗi
  End if
  End                    'Kết thúc chương trình chính
End Sub
```

Lưu ý: Có thể đóng, mở thiết bị khi tạo và thoát ra khỏi một form điều khiển.

V.1. Xuất dữ liệu ra ngõ Digital

+ Khai báo biến

```
Option Explicit
Dim lpDioWriteBit As PT_DioWriteBit  'Khai báo biến kiểu PT_DioWriteBit.
```

(Đây là struct đã được định nghĩa sẵn trong Driver, bao gồm các thông tin : Port, Bit, giá trị của Bit cần xuất).

+ Tạo nút nhấn

với mã lệnh như sau:

```
Private Sub Command1_Click()
```

lpDioWriteBit.Port = 0 'Port 0

lpDioWriteBit.Bit = 2 'Bit 2

lpDioWriteBit.state = 1 'Xuất mức 1

status = DRV_DioWriteBit(DriverHandle, lpDioWriteBit)' xuất dữ liệu

End Sub

+ Tạo nút nhấn

với mã lệnh tương tự như trên nhưng xuất mức 0 ra Bit2 của Port 0.

Chạy thử chương trình để kiểm tra kết quả

(Có thể tạo File chạy .exe bằng cách vào menu File , chọn Make Project.exe)

V.2 Đọc dữ liệu ngõ vào Analog

Dùng Timer định thời sau mỗi 200ms đọc giá trị điện áp về một lần.

+ Có thể khai báo các biến như sau:

Option Explicit

Dim Cauhinh As PT_AIConfig

Dim ReadAD As PT_AIVoltageIn

Dim Voltage As Single

(*PT_AIConfig, PT_AIVoltageIn* là những struct đã được định nghĩa sẵn trong Driver, sinh viên cần xem các thành phần của struct ở Phụ lục.)

+ Kéo biểu tượng Timer vào Form đang thiết kế, khai báo thuộc tính Interval là 200ms, nhấp đôi chuột vào biểu tượng Timer  để viết mã lệnh như sau:

Private Sub Timer1_Timer()

Cauhinh.DasChan = 1 ' Chọn kênh 1

Cauhinh.DasGain=0'Chọn gain code là 0:Không nhân hệ số, tầm điện áp vào là ±5V

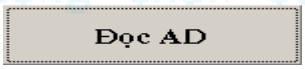
status = DRV_AIConfig(DriverHandle, Cauhinh)'Đặt cấu hình cho ngõ vào analog

If (status <> 0) Then

DRV_GetErrorMessage status, Loi

Response = MsgBox(Loi, vbOKOnly, "Error")

```
    Exit Sub
  End If
  ReadAD.Chan =Cauhinh.DasChan  'Đọc kênh 1
  ReadAD.Gain =Cauhinh.DasGain
  ReadAD.TrigMode = 0                ' Kích bằng phần mềm
  ReadAD.Voltage=DRV_GetAddress(voltage)'Đọc giá trị điện áp lưu vào biến voltage
  status = DRV_AIVoltageIn(DriverHandle,ReadAD)' Lệnh đọc AD
  If (status <> 0) Then
     DRV_GetErrorMessage status, Loi
    Response = MsgBox(Loi, vbOKOnly, "Error")
  Exit Sub
  End If
  Text1.Text = Format(Voltage, "##0.0##") 'Cập nhật giá trị điện áp lên ô TextBox
End Sub
```


+ Khi nhấn nút 

thì cho phép Timer hoạt động bằng lệnh: *Timer1.Enabled = True*

+ Khi nhấn nút 

thì ngừng Timer bằng lệnh: *Timer1.Enabled = False*

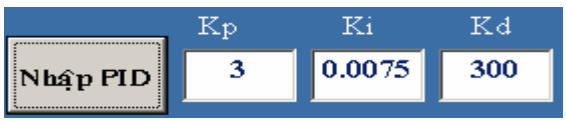
V.3 Điều khiển nhiệt độ theo giải thuật PID số

Sinh viên có thực hiện theo các bước sau:

Bước 1: Tính các giá trị Kp, Ki, Kd với thông số L và T và công thức đã liệt kê ở phần *Nội dung thí nghiệm*.

Tuỳ theo bộ điều khiển sử dụng P, PI hay PID mà khi chạy chương trình người sử dụng có thể nhập lại các giá trị Kp, Ki, Kd.

Cần khai báo biến để lưu các giá trị này khi nhấn nút Nhập PID:



Có thể sử dụng hàm : *CDbl* để chuyển từ dạng Text hiển thị trên ô TextBox sang biến kiểu Double, cú pháp như sau:

Tên biến = CDbl (Tên ô TextBox.Text)

Bước 2: Tạo một Timer có chu kỳ là 2s làm chu kỳ điều khiển lò nhiệt.

Khi nhấn nút 

thì cho phép Timer này hoạt động: *Timer.Enabled = True*

Khi nhấn nút Thoát thì dừng Timer này lại: *Timer1.Enabled = False*

Bước 3: Nhấp đôi vào biểu tượng Timer để viết chương trình điều khiển theo giải thuật PID đã trình bày trong phần *Nội dung thí nghiệm*, với các lưu ý sau:

+ Nhiệt độ hiện tại của lò được xác định bằng cách đọc giá trị điện áp (đọc AD) như đã trình bày ở mục V.2 và tính theo công thức :

Nhiệt độ đo = 78.125*Điện áp vào

Giá trị 78.125 là hệ số quy đổi từ điện áp AD đọc về ra nhiệt độ được tính từ thông số của mạch đã trình bày ở mục III của Phụ lục. Cần hiển thị nhiệt độ này lên một ô TextBox

+ Giá trị Umax trong giải thuật đã trình bày ở ***Thí nghiệm 3*** có thể được xác định từ công thức tính luật điều khiển PID tại thời điểm ban đầu khi cho e[(k-1)T],e[(k-T], u[(k-1)T] bằng 0:

$$u_{max} = e(0)\left(Kp + \frac{Kd}{T} + \frac{KiT}{2}\right)$$

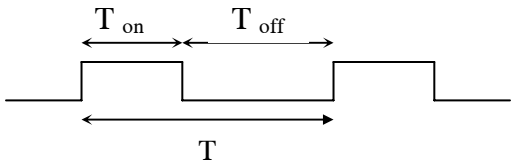
Trong đó:

+ e(0) là sai số giữa nhiệt độ đặt và nhiệt độ hiện tại của lò ngay tại thời điểm ban đầu.

+ T là chu kỳ điều khiển, ở đây là 2s.

+ Để điều khiển được lò nhiệt theo giải thuật đã thiết kế, ta sử dụng phương pháp điều rộng xung như sau:

Tạo một xung điều rộng ở **Bit 3 của Port 0** như ở hình bên



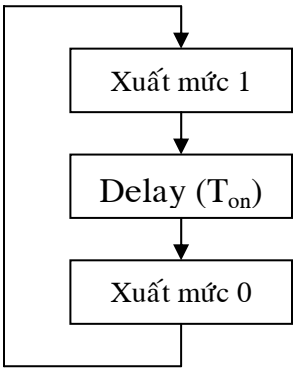
Với T là chu kỳ điều khiển, T_{on} là thời gian cấp điện, T_{off} là thời gian không cấp điện cho lò bằng cách xuất mức 1 (ứng với T_{on}) hay mức 0 (ứng với T_{off}) ra **Bit 3 của Port 0** để kích dẫn hoặc không dẫn Triac cấp điện cho lò.

Từ giá trị của luật điều khiển *u* tính được theo công thức PID, ta tính các thời gian T_{on}, T_{off} theo công thức:

$$T_{on} = \frac{u}{u_{max}} * T \quad , \quad T_{off} = T - T_{on}$$

Vì chu kỳ điều khiển T = 2s là khá lớn so với độ phân giải của Timer trong máy tính, do đó để đơn giản ta có thể tạo xung điều rộng bằng cách viết chương trình xuất mức 1 ra **Bit 3 của Port 0**, sau đó làm trễ (delay) một khoảng thời gian bằng T_{on} rồi xuất mức 0 trong khoảng thời gian còn lại của chu kỳ điều khiển.

Một chu kỳ



- Hàm Delay có thể viết đơn giản bằng cách dùng một Timer khác, giả sử là Timer2, và viết mã lệnh như sau:

```
Public Sub Delay(x AsInteger)
    Timer2.Interval = x
    Timer2.Enabled = True
  Do
    DoEvents   ' Trong khi delay thì máy tính có thể thực hiện việc khác.
  Loop Until Timer2.Enabled = False
End Sub
```

- Nhấp đôi vào biểu tượng của Timer2 và gõ vào dòng lệnh: *Timer2.Enabled = False*

- Khi cần gọi hàm Delay thì đưa đối số vào là một số nguyên, đơn vị tính là ms.

<u>*Lưu ý*</u>: Chỉ sử dụng hàm Delay ở trên khi $T_{on} \neq T$ và $T_{on} \neq 0$. Khi $T_{on}=T$ hay $T_{on}=0$ thì không sử dụng hàm này mà chỉ cấn xuất mức 1 hay mức 0 ra tương ứng. Sinh viên có thể dùng thêm câu lệnh If hay Case để xét hai trường hợp này.

+ Cách thức xuất dữ liệu ra Bit 3 của Port 0, sinh viên thực hiện như mục V.1 *(Xuất dữ liệu ra ngõ Digital)*

VI. BÁO CÁO THÍ NGHIỆM

1. Các chương trình ở Thí nghiệm 1, 2, 3 bao gồm các Form thiết kế và mã lệnh phải được chú thích cẩn thận.

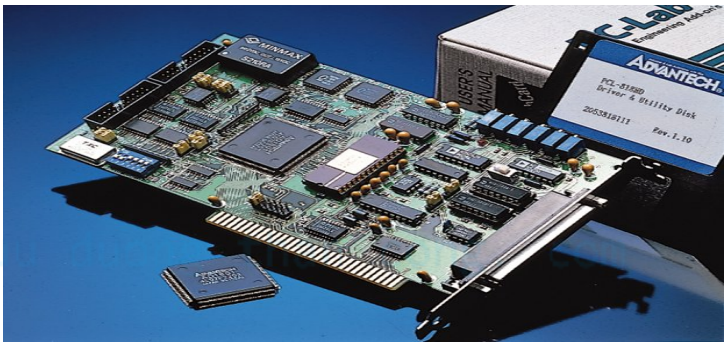
2. Dùng hàm Line của Visual Basic để vẽ đồ thị hiển thị nhiệt độ của lò theo thời gian.

(Sinh viên có thể xem các trợ giúp lập trình trong bộ đĩa MSDN)

Phụ lục

I. GIỚI THIỆU CARD PCL818L

PCL818L là Card cắm vào rãnh ISA của máy tính của hãng Advantech được dùng trong các ứng dụng đo lường và điều khiển bằng máy tính như : chuyển đổi D/A, A/D 12bits nhiều kênh, Digital Input, Digital Output, Timer, Counter.



I.1. Các đặc điểm chính

Chuyển đổi A/D

+ Số kênh :16 kênh đơn cực hoặc 8 kênh vi sai, chọn lựa bằng jumper JP6 trên Card

+ Độ phân giải 12 bit

+ Tầm điện áp ngõ vào có thể chọn được trên Card bằng jumper JP7 (± 5V hay ± 10V) hoặc bằng phần mềm. Ngõ vào vi sai hay đơn cực được chọn bằng JP6

+ Giới hạn áp vào cho phép : +/- 30 Vmax

+ Tốc độ biến đổi tối đa: 40 KHz

+ Có thể Trigger bằng phần mềm hoặc bên ngoài

Chuyển đổi D/A

+ Số kênh : 1 kênh

+ Độ phân giải : 12 bit

+ Tầm điện áp ngõra:từ 0 đến +5V /+10V với nguồn chuẩn –5V/-10 V trên board

+ Thời gian đáp ứng : 5µs.

Ngõ vào số

+ Số kênh :16 bit

+ Mức logic: tương thích TTL

+ Áp vào : mức cao = 2 Vmin

mức thấp = 0.8 Vmax

+ Dòng ngõ vào : mức thấp = 0.4 mA max tại 0.5V

mức cao = 0.05 mAmax tại 2.7 V

Ngõ ra số

+ Số kênh :16 bit

+Mức logic: tương thích TTL.

+Áp ra : mức thấp = 0.5 V

mức cao = 2.4

Bộ định thời và bộ đếm

+ Thiết bị : Intel 8254

+ Bộ đếm : 3 bộ 16 bit: 2 bộ được nối nối tiếp nhau để phục vụ cho việc tạo xung Trigger để chuyển đổi A/D, bộ còn lại có thể dùng được.

+Tần số ngõ vào:

Kênh 1: 10 MHz hay 1 MHz được chọn bởi switch

Kênh 2: lấy từ ngõ ra kênh 1

Kênh 0: từ 100KHz đến 10MHz

+ Xung clock đưa ra ngoài: 0.00023 Hz đến 2.5 MHz

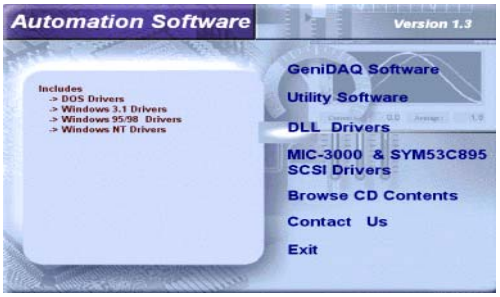
I.2. Cài đặt và sử dụng DLL Driver kèm theo của Card PCL818L

Trước khi gắn Card PCL818L vào rãnh ISA của máy tính ta cần cài đặt địa chỉ trên Switch1 và các Jumper trên Card thích hợp (Xem thêm trong User Manual).

Trong Visual Basic để điều khiển Card ta có thể viết file thư viện liên kết động DLL truy xuất trực tiếp các địa chỉ của Card hoặc sử dụng Driver kèm theo.

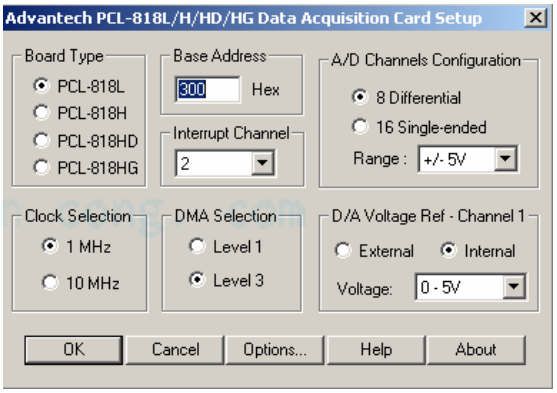
Driver của Card PCL818L bao gồm các hàm đã viết sẵn, khi sử dụng ta chỉ cần gọi hàm và đưa vào các thông số thích hợp.

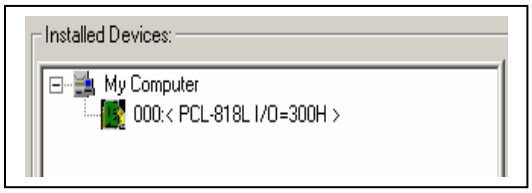
Chọn DDL Driver từ đĩa Driver kèm theo để cài Driver của Card vào máy tính.



Sau khi cài đặt xong Driver, chạy chương trình Device Installation để cài đặt cho Card sử dụng:

Chọn Advantech PCL-818L/H/HD/HG ở mục List of Devices → Chọn Add→Khai báo các thông số cho Card → Nhấn OK để xác nhận và thoát khỏi cài đặt. Lúc này trên mục Installed Devives sẽ có tên, số thứ tự (chỉ số thiết bị) và địa chỉ của Card sử dụng.





Số thứ tự của Card cài đặt đầu tiên là 0, đây là chỉ số tương ứng với tham số DeviceNumber dùng trong Driver

Khi viết chương trình chỉ cần gọi hàm có sẵn trong Driver và khai báo con trỏ với cấu trúc dữ liệu tương ứng với hàm đó.

I.3. Một số hàm được sử dụng trong bài thí nghiệm

DRV_DeviceOpen

Status = DRV_DeviceOpen(DeviceNum, DriverHandle)

Ứng dụng: Được dùng để mở thiết bị.

Tham số:

Tên	Hướng	Kiểu	Mô tả
DeviceNum	Input	Unsigned long	Chỉ số thiết bị (có được khi cài đặt bằng chương trình Device Installation)
DriverHandle	Output	Long pointer	Trỏ đến thiết bị sử dụng (đây là một "thẻ" chỉ đến Card đang sử dụng)

DRV_DeviceClose(DriverHandle)

Status = DRV_DeviceClose(DriverHandle)

Ứng dụng: Dùng để đóng thiết bị đã được mở bởi hàm DRV_DeviceOpen

Tham số :

Tên	Hướng	Kiểu	Mô tả
DriverHandle	Input/Output	Long pointer	Được gán bởi hàm DRV_DeviceOpen

DRV_DioWriteBit

Status = DRV_DioWriteBit(DriverHandle, lpDioWriteBit)

Ứng dụng: Xuất dữ liệu ra một địa chỉ bit

Tham số :

Tên	Hướng	Kiểu	Mô tả
DriverHandle	Input	Long	Được gán bởi hàm DRV_DeviceOpen
lpDioWriteBit	Input/Output	Trỏ đến con trỏ PT_DioWriteBit	Lưu địa chỉ cho port, bit và state

PT_DioWriteBit

Con trỏ PT_DioWriteBit được sử dụng bởi hàm DRV_DioWriteBit có cấu trúc dữ liệu như sau:

Tên	Hướng	Kiểu	Phạm vi	Môtả
Port	input	Unsigned short	0-n (n phụ thuộc vào phần cứng)	Chỉ số port

Bit	input	Unsigned short	0-7	Chỉ số bit
State	input	Unsigned short	0 hay1	Trang thái của bit (mức 0 hay 1)

DRV_ AIConfig

Status = DRV_AIConfig(DriverHandle, lpAIConfig)

Ứng dụng: Cài đặt độ lợi (gain) cho ngõ vào analog

Tham số:

Tên	Hướng	Kiểu	Mô tả
DriverHandle	Input	Long	Được gán bởi DRV_DeviceOpen
lpAIConfig	Output	Trỏ đến con trỏ PT_AIConfig	Lưu địa chỉ của kênh biến đổi A/D và độ lợi Gain

PT_AIConfig

Con trỏ PT_AIConfig được sử dụng bởi hàm DRV_AIConfig có cấu trúc dữ liệu như sau:

Tên	Hướng	Kiểu	Phạm vi	Mô tả
DasChan	Input	Unsigned short	0-n (n phụ thuộc vào phần cứng)	Chỉ số kênh ngõ vào analog
DasGain	Input	Unsigned short	Xem trong quyển User Manual	Mã độ lợi

(Tuỳ theo tầm điện áp ngõ vào mà mã số độ lợi Gain *khác nhau, xem thêm trong User Manual, trang 117)*

DRV_AIVoltageIn

Status = DRV_AIVoltageIn(DriverHandle, lpAIVoltageIn)

Ứng dụng: Đọc 1 kênh vào analog và trả về kết quả là giá trị điện áp. (Đơn vị: Volt)

Tham số:

Tên	Tính chất	Kiểu	Mô tả
DriverHandle	Input	Long	Được gán bởi DRV_DeviceOpen
lpAIVoltageIn	Input/Output	Trỏ đến con trỏ PT_AIVoltageIn	Lưu địa chỉ chỉ kênh biến đổi, gain, TrigMode và điện áp

PT_AIVoltageIn

Con trỏ PT_AIVoltageIn được sử dụng bởi hàm DRV_AIVoltageIn có cấu trúc dữ liệu như sau:

Tên	Hướng	Kiểu	Phạm vi	Mô tả
chan	Input	Unsigned short	0-n (n phụ thuộc phần cứng)	Chỉ số kênh ngõ vào
gain	Input	Unsigned short	Xem trong quyển User Manual	Mã độ lợi
TrigMode	Input	Unsigned short	0-normal (phần mềm) 1-bên ngoài	Mã kích
Voltage	Output	Trỏ đến dữ liệu kiểu floating-point	Phụ thuộc vào tầm điện áp ngõ vào	Giá trị điện áp ngõ vào đo được

DRV_GetErrorMessage

Status = DRV_GetErrorMessage(ErrorCode,ErrorMsg)

Ứng dụng: Gọi thông báo lỗi tương ứng với mã lỗi xác định và trả về trong message buffer.

Tham số:

Tên	Hướng	Kiểu	Mô tả
ErrorCode	Input	Unsigned short	Mã lỗi được trả về bởi Driver
ErrorMsg	Output	Trỏ đến dữ liệu kiểu string	Lưu thông báo lỗi

Ghi chú:

Nếu không sử dụng Driver có sẵn khi viết chương trình thì phải lập trình truy xuất trực tiếp địa chỉ của Card. Tuỳ theo ngôn ngữ sử dụng mà viết chương trình xuất nhập ngoại vi khác nhau:

+ C hoặc Visual C : Sử dụng các hàm INP và OUTP

+ Delphi : Tạo chương trình con xuất nhập viết bằng hợp ngữ chèn vào chương trình Delphi như sau:

```
Function  Inport (address:word):byte;
  var data:word;
  begin
     asm
        mov dx, address
        in ax, dx
        mov data, ax
     end;
     Inport:= data;
end;
```

```
Procedure Outport (address: word; data: word);
 begin
    asm
       mov dx, address
       mov ax, data
       out dx, ax
    end;
 end;
```


+ Visual Basic: Không hỗ trợ các hàm xuất nhập Port và chèn thêm hợp ngữ mà phải tạo một thư viện liên kết động (.dll) có thể bằng ngôn ngữ khác (Visual C hay Delphi,...) rồi khai báo sử dụng trong chương trình Visual Basic.

Chi tiết hơn về cách sử dụng các hàm này có thể tham khảo thêm trong *Chương 4* quyển sách Đo Lường Điều Khiển Bằng Máy Tính của tác giả TS.Nguyễn Đức Thành.

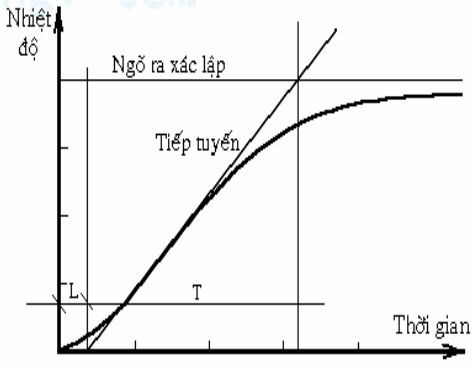
II. GIẢI THUẬT ĐIỀU KHIỂN PID SỐ

II.1 Mô hình lò nhiệt theo Ziegler-Nichols

Theo Ziegler-Nichol thì mô hình lò nhiệt có thể được biểu diễn dưới dạng hàm truyền sau :

$$H(s) = \frac{Ke^{-Ls}}{Ts+1}$$

Hàm truyền này bao gồm một khâu quán tính hệ số khuếch đại K và thời hằng T, và khâu trễ thời gian L, các thông số này có thể được lấy khi kẻ tiếp tuyến ở điểm uốn cho đồ thị quá độ hàm nấc như hình bên.



Hệ số khuếch đại K được tính như sau :

Xác định thông số của lò nhiệt theo
Ziegler – Nichols

Khi nhiệt độ đầu khác không, K được tính từ độ tăng nhiệt độ ngõ ra so với môi trường.

Để áp dụng cho hệ tuyến tính, ta lấy khai triển Talor của e^{-Ls} , hàm truyền trở nên:

$$H(s) = \frac{K}{(Ts+1)(Ls+1)}$$

Tóm lại, Ziegler-Nichols đã xấp xỉ hàm truyền lò với hệ bậc nhất có trễ hay hệ tuyến tính bậc hai, và cho phép tìm hàm truyền bằng thực nghiệm khi vẽ quá trình quá độ của hệ thống với ngõ vào hàm nấc.

II.2 Thuật toán hiệu chỉnh PID số

Hàm truyền liên tục PID có dạng: $H(s) = \frac{u(s)}{e(s)} = K_p + \frac{K_i}{S} + K_d s$

Trong đó u : ngõ ra , e : ngõ vào của bộ hiệu chỉnh

Thuật toán PID có thể nhận được khi sai phân hàm truyền trên, tương ứng phương trình vi tích phân sau:

$$K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt} = u(t)$$

Rời rạc hoá:

-Khâu vi phân (dùng định nghĩa sai phân): $K_d \frac{de(t)}{dt} = \frac{K_d}{T}(e[k] - e[k-1])$

-Khâu tích phân (theo nguyên tắc hình thang): $\int e(t)dt = \frac{T}{2}\sum_{i=1}^{k}(e[i] + e[i-1])$ với e[0]= 0

suy ra

$$u(k) = K_p e(k) + \frac{K_d}{T}(e[k] - e[k-1]) + \frac{T}{2}\sum_{i=1}^{k}(e[i] + e[i-1]) \text{ với e[0]= 0}$$

thay k bằng k-1 và trừ vào phương trình trên, nhận được công thức cho phép ta tính luật điều khiển u[k] từ u[k-1] và các giá trị liên tiếp của e[k] như sau:

$$u(k) - u(k-1) = A_0 * e(k) + A_1 * e(k-1) + A_2 * e(k-2)$$

với:

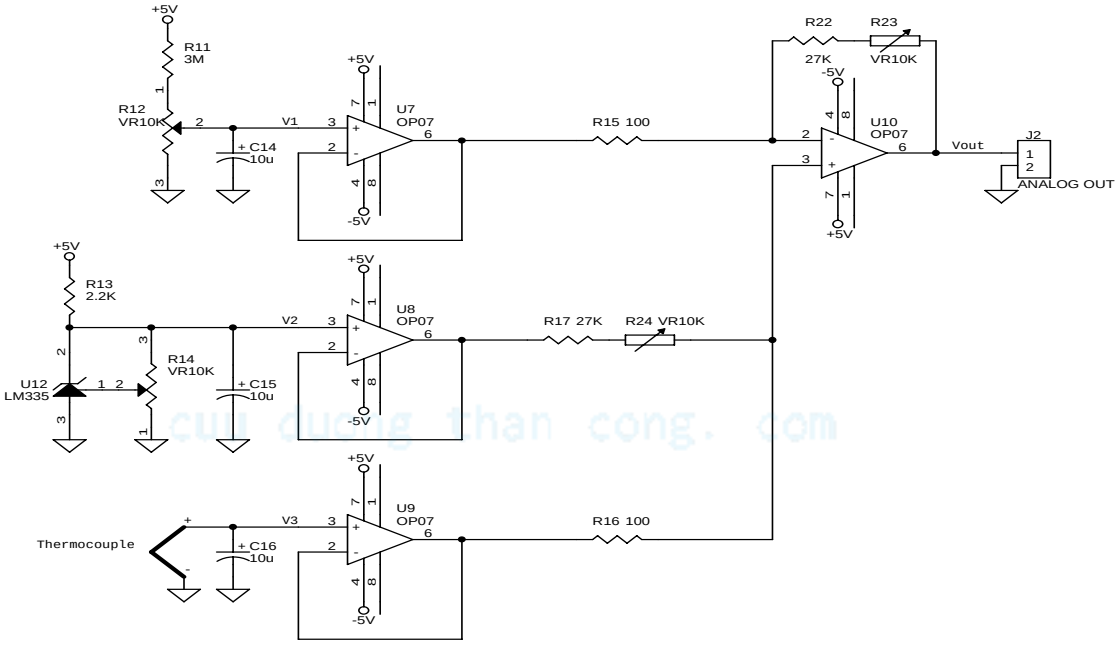
$$A_0 = K_p + K_i * T/2 + K_d/T$$
$$A_1 = K_i * T/2 - K_p - 2K_d/T$$
$$A_2 = K_d/T$$

trong đó T là chu kỳ lấy mẫu.

Bằng cách cho Kd bằng 0 hay Ki,Kd bằng 0 ta có được bộ điều khiển PI hay bộ điều khiển P.

III. CÁC SƠ ĐỒ MẠCH ĐIỆN DÙNG TRONG THÍ NGHIỆM

Mạch gia công tín hiệu:



Mạch này thực hiện 3 chức năng sau : bù nhiệt cho đầu tự do, khuếch đại, và tạo điện áp ra là 0V khi đo ở 0^0C. Hoạt động của mạch trên như sau:

- U7, U8, U9 (dùng OP07 cho offset thấp) làm bộ đệm điện áp.
- Điện áp ra trên thermocouple :

$$V3 = S(T_d - T_a) = S.T_d - S.T_a$$

Với : T_d là nhiệt độ cần đo.

T_a là nhiệt độ môi trường.

S là độ nhạy của thermocouple loại K (40µV/°C).

Để loại trừ ảnh hưởng nhiệt độ môi trường, dùng khối tạo ra điện áp theo nhiệt độ môi trường nhưng có dấu ngược lại sử dụng IC LM335A.

- IC LM335A là loại cảm biến nhiệt độ bán dẫn, có độ nhạy là 10mV/°K. Áp tạo ra do LM335A cảm biến được là :

$$V2 = K.T_a\ [°K] = K(273 + T_a)\ [°C]$$

$= K.273 + K.T_a = C + KT_a$ $(C = K.273)$

với K = 10mV/°K; C = 2,73V

Như vậy có thể triệt tiêu ảnh hưởng của T_a, tuy nhiên tạo ra một mức điện áp là 2,73V ở 0°C nên R12 là để trừ 2,73V nhằm tạo điện áp đầu ra là 0V ở 0°C..

- U10 (dùng OP07) đóng vai trò bộ cộng có khuếch đại, điện áp ra cuối cùng là:

$$V_{out} = \left(1 + \frac{R22 + R23}{R15}\right)\left[(R17 + R24) // R16\right]\left(\frac{V2}{R17 + R24} + \frac{V3}{R16}\right) - \frac{R22 + R23}{R15}V1$$

$$\Rightarrow V_{out} = \left(1 + \frac{R22 + R23}{R15}\right)\left[(R17 + R24) // R16\right]\left(\frac{C}{R17 + R24} + \frac{K.T_a}{R17 + R24} + \frac{S.T_d}{R16} - \frac{S.T_a}{R16}\right) - \frac{R22 + R23}{R15}V1$$

Để không bị ảnh hưởng của nhiệt độ môi trường :

$$\frac{K.T_a}{R17 + R24} - \frac{S.T_a}{R16} = 0 \Rightarrow \frac{R17 + R24}{R16} = \frac{K}{S} = \frac{10mV}{29,5\mu V} = 338,893$$

chọn R16 = 100Ω ⇒ R17 + R24 = 338,893KΩ ⇒ chọn R17 = 27KΩ và R24 là biến trở 10KΩ. Khi đó cần điều chỉnh R24 để triệt tiêu T_a.

Triệt tiêu điện áp tĩnh (2,73V) :

$$\left(1 + \frac{R22 + R23}{R15}\right)\left[(R17 + R24) // R16\right]\left(\frac{C}{R17 + R24}\right) - \frac{R22 + R23}{R15}V1 = 0$$

với (R17 + R24)//R16 = (338,893KΩ)//(100Ω) = 99,97Ω

$$\Rightarrow \left(1 + \frac{R22 + R23}{R15}\right)x99,97x\frac{2,73}{338893} = \frac{R22 + R23}{R15}xV1$$

$$hay: \frac{R22 + R23}{R15}\left(V1 - \frac{2,73}{3389,947}\right) = \frac{2,73}{3389,947} \qquad (1)$$

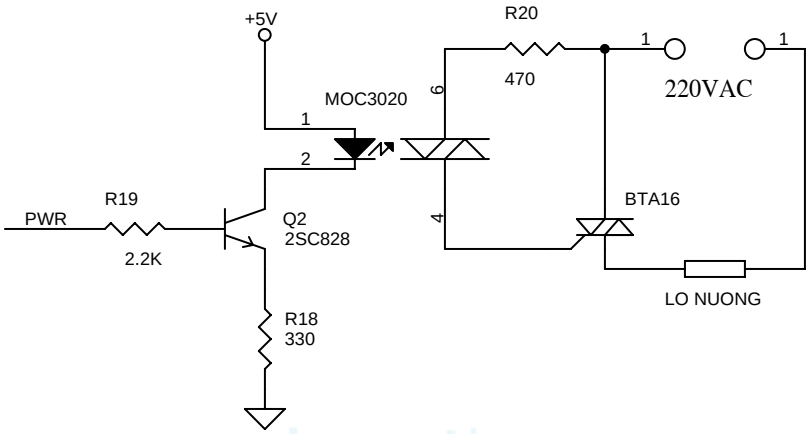
khi đó, điện áp ra là :

$$V_{out} = \left(1 + \frac{R22 + R23}{R15}\right)\left[(R17 + R24) // R16\right]\frac{S.T_d}{R16}$$

Thay giá trị của các linh kiện đã thiết kế , ta suy ra được quan hệ giữa nhiệt độ cần đo và điện áp của ngõ vào bộ biến đổi AD là: 78.125

- Các tụ C14, C15, C16 có giá trị 10μF là để chống nhiễu.

Mạch công suất:



Opto triac được lái qua transistor C828 gắn vào ngõ digital output D/01của card PCL-818L. Xuất mức 1 hay mức 0 ở ngõ ra này để điều khiển đóng hay mở điện cho lò.