

LẬP TRÌNH CHO MÁY TÍNH ĐIỀU KHIỂN

4.1 LẬP TRÌNH XUẤT NHẬP NGOẠI VI

Khi dùng máy tính để đo lường, điều khiển ta phải giải quyết vấn đề là xuất một dữ liệu 8 bit ra một thanh ghi hay đọc dữ liệu 8 bit từ thanh ghi vào một biến.

Vấn đề này được giải quyết dễ dàng bởi các ngôn ngữ lập trình trong môi trường DOS như hợp ngữ, Qbasic, Pascal, C. Với hệ điều hành Windows 98, 2000XP thì công việc trở nên phức tạp hơn..

4.1.1 Lập trình xuất nhập trong môi trường DOS

1- Lập trình hợp ngữ

Xuất dữ liệu từ biến *data* ra cổng có địa chỉ *address*:

```
mov    dx, address
mov    ax, data
out    dx, ax
```

Ví dụ:

```
mov    dx, 300H
mov    ax, F0H
out    dx, ax
```

Nhập số liệu từ cổng địa chỉ *address* vào biến *data*

```
mov    dx, address
in     ax, dx
mov    data, ax
```

data, address là số nhị phân 16 bit.

2- Qbasic

Dùng lệnh INP (*port%*) và OUT *port%, data%*

Nguyễn Đức Thành

port là số nguyên từ 0 đến 65535 chỉ địa chỉ cổng

data là số nguyên từ 0 ÷ 255,

&H ký hiệu số Hex phía sau , % ký hiệu biến là số nguyên

Ví dụ: a% = INP (&H3FC) ‘đọc số liệu ở cổng 3FCH, thanh ghi modem

OUT &H3FC, (a% XOR 1) ‘đảo bit cuối, xuất ra cổng 3FCH

OUT 768, 160 ‘ xuất A0H ra cổng 300H

3- Pascal

Dùng lệnh port: xuất nhập một byte, portw: xuất nhập một từ

Xuất data ra port địa chỉ reg

port[reg]: = data;

Đọc port địa chỉ reg vào biến value

value: = port [reg];

Ví dụ: data: = port [\$378];

port [\$37A]:= data out;

4- C, C Builder, Visual C

Dùng lệnh inp và outp (hay inportb, outportb, inport, outport)

Control = inp (0X61);

outp (0X43, 0Xb6);

Visual C++ 6.0 có các hàm và thủ tục:

<i>_inp:</i>	đọc byte	<i>_outp:</i>	xuất byte
<i>_inpw:</i>	đọc một từ	<i>_outpw:</i>	xuất từ
<i>_inpd:</i>	đọc từ kép	<i>_outpd:</i>	xuất từ kép

Ví dụ: lập trình C

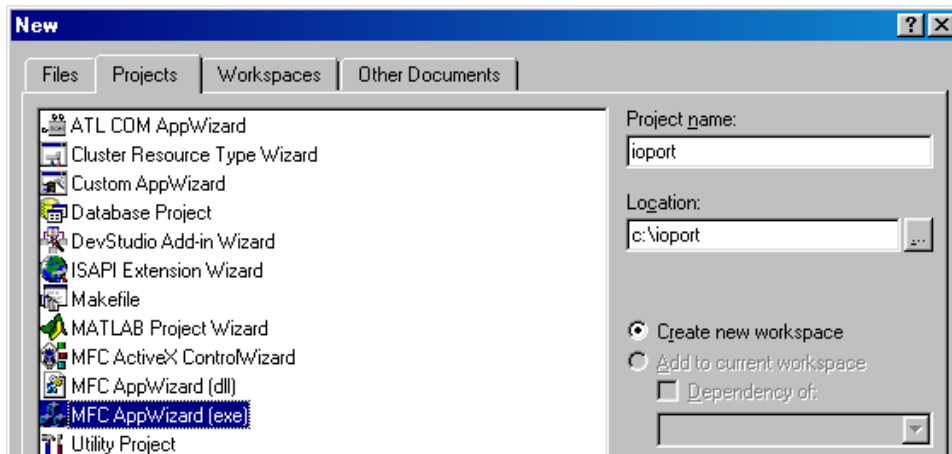
```
# include <conio.h>
# define dataport 0x378
# define statusport 0x379

int Dummy;

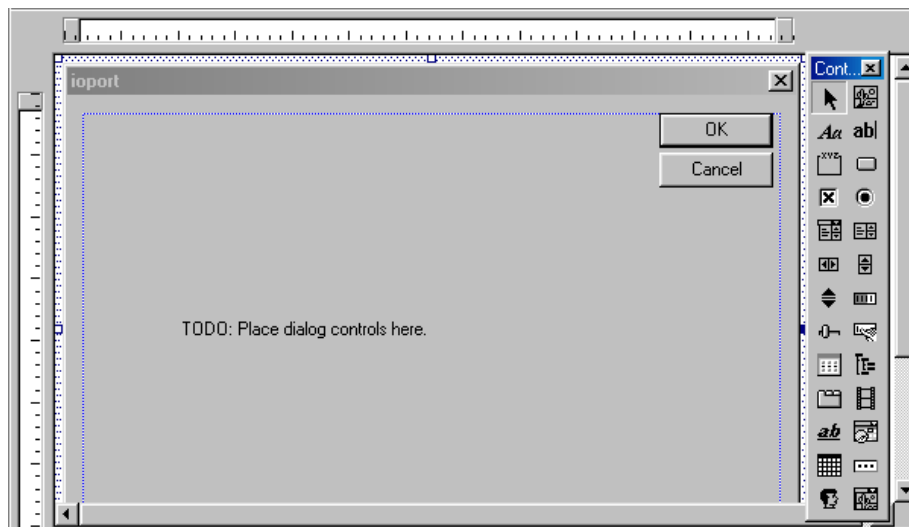
Dummy = _outp(dataport, 0);
Dummy = _inp(statusport);
...
```

4.1.2 Lập trình xuất nhập dùng Visual C++6.0 (Win 98)

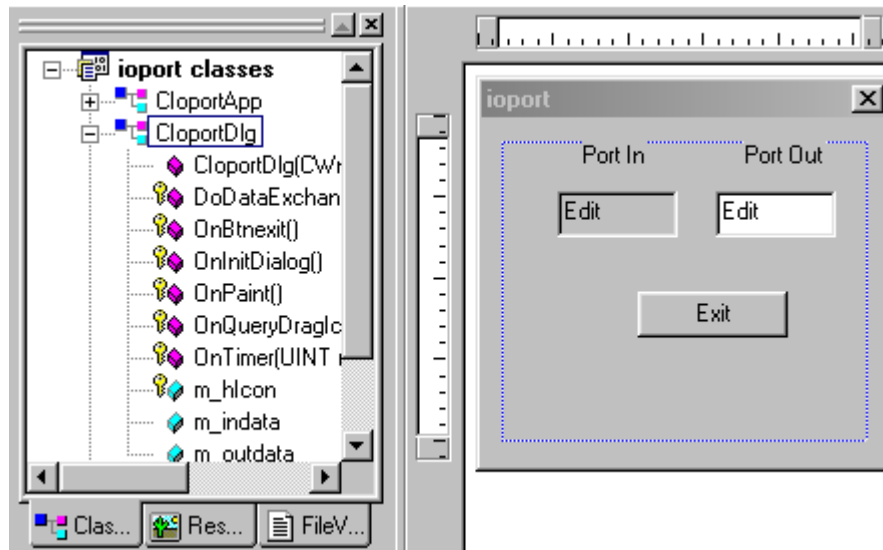
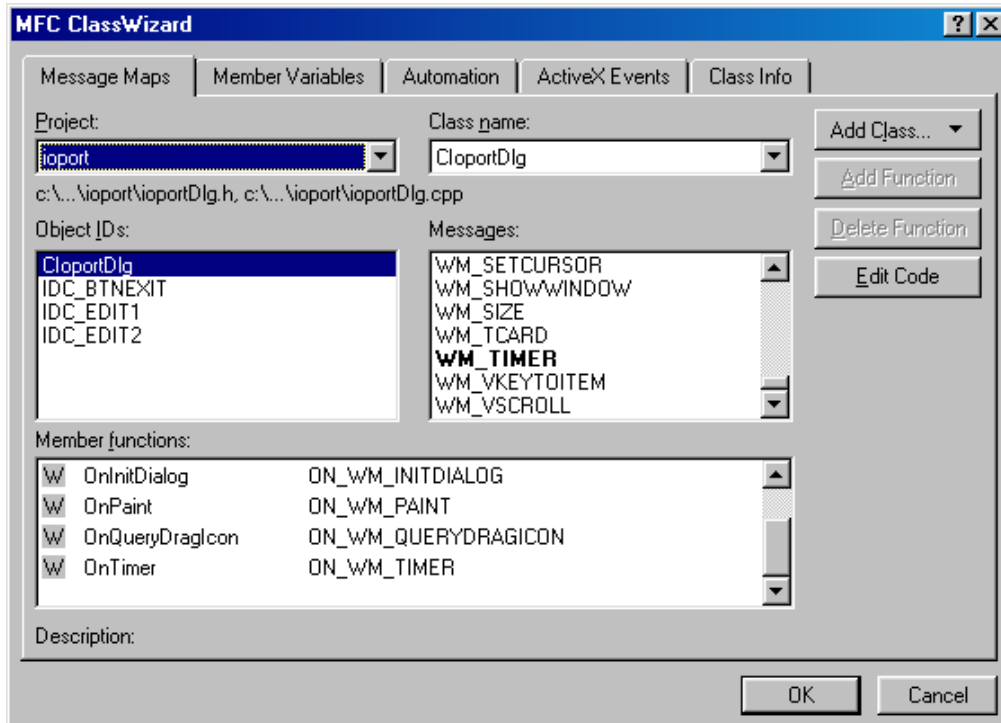
Sau khi mở VC, bạn vào menu File – New – Projects chọn mục MFCAppWizard (exe), sau đó đặt tên cho project và địa chỉ lưu file rồi bấm OK.



Khi cửa sổ MFCAppWizard xuất hiện, chọn option Dialog Based sau đó tiếp tục bấm Next, ở bước 4 bạn bấm Finish rồi OK



Sau đó bạn tạo các đối tượng Static Text, Edit Box, Button và Timer, các biến liên kết với Edit Box Port In và Port Out lần lượt là m_indata và m_outdata. Cứ mỗi 1s do Timer chương trình đọc dữ liệu ở Port In 379, xử lý rồi xuất ra Port Out 378



Sau đây là một phần của chương trình xuất nhập cổng, các dòng in đậm là các dòng bổ sung thêm

// ioportDlg.cpp : implementation file

```

#include "stdafx.h"
#include "ioport.h"
#include "ioportDlg.h"
# include "conio.h" //definition _inp and _outp
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[ ] = __FILE__;
#endif
BOOL CloportDlg::OnInitDialog()
{
    CDialog::OnInitDialog();
    SetIcon(m_hIcon, TRUE);           // Set big icon
    SetIcon(m_hIcon, FALSE);        // Set small icon
    SetTimer(ID_CLOCK_TIMER,1000, NULL);
    return TRUE;
}
void CloportDlg::OnTimer(UINT nIDEvent)
{
    UpdateData(TRUE);
    m_indata=_inp(0x379);
    m_outdata=m_indata ^0x80;
    _outp(0x378,m_outdata);
    UpdateData(FALSE);
    CDialog::OnTimer(nIDEvent);
}
void CloportDlg::OnBtnexit()
{
    OnOK();
}

```

4.1.3 Lập trình dùng Delphi

Không có lệnh xuất nhập mà ta phải dùng hợp ngữ tạo hàm xuất nhập.

```

Function Inport (address: word): byte;
    Var data: word;
    begin
        asm
            push dx
            mov dx, address
            in ax, dx
            mov data, ax
            pop dx
        end;
        inport:= data; end;
Procedure Outport (address: word; data: word);
    begin
        asm
            push dx
            mov dx, address
            mov ax, data

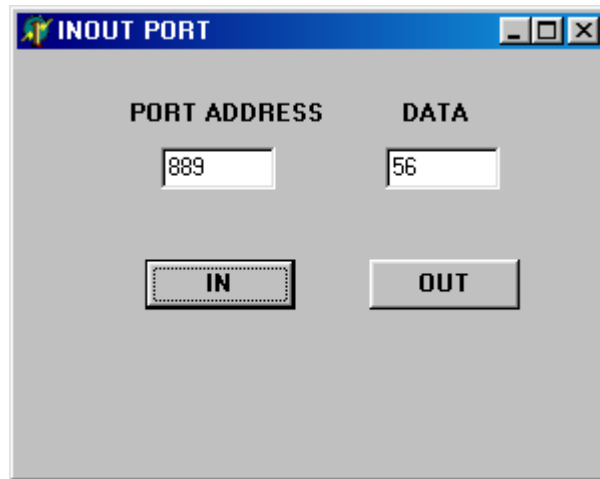
```

```

        out dx, ax
    pop dx
end;
end;

```

Ví dụ: Chương trình Delphi xuất nhập Port



```

unit inoutunit;
interface
uses
    Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls;
type
    TForm1 = class(TForm)
        Edit1: TEdit;
        Edit2: TEdit;
        Label1: TLabel;
        Label2: TLabel;
        btnIN: TButton;
        btnOUT: TButton;
    public
        function inport (address:word):byte;
        procedure outport (address:word; data:byte);
        procedure btnINClick (Sender: TObject);
        procedure btnOUTClick (Sender: TObject);
    private
        { Private declarations }
    public
        { Public declarations }
    end;
var
    Form1: TForm1;
implementation
{$R *.DFM}
function TForm1.inport (address:word):byte;
var data:byte;
begin
    asm

```

```

        push dx
        mov dx,address
        in al,dx
        mov data, al
        pop dx
    end;
    inport:= data;
end;
procedure TForm1.outport (address:word; data:byte);
begin
    asm
        push dx
        mov dx,address
        mov al,data
        out dx,al
        pop dx
    end;
end;
procedure TForm1.btnINClick (Sender: TObject);
begin
    edit2.text:= inttostr (inport (strtoint (edit1.text)));
end;
procedure TForm1.btnOUTClick (Sender: TObject);
begin
    outport (strtoint (edit1.text), strtoint (edit2.text));
end;
end.

```

4.1.4 Visual Basic

Ngôn ngữ này không hỗ trợ xuất nhập port, ta phải dùng thư viện liên kết động (*dynamic link library*)

4.2 VIẾT TẬP TIN LIÊN KẾT ĐỘNG DLL

File thư viện liên kết động có đuôi .dll chứa các hàm và thường trình mà các chương trình chạy trong môi trường Windows như Delphi, Visual Basic, Visual C có thể sử dụng.

Các ngôn ngữ lập trình Visual C, Delphi, Borland C++ Builder đều cho phép viết tập tin .dll. Hệ điều hành Windows có sẵn một số hàm thư viện liên kết động gọi chung dưới tên Win API (*Applications Programming Interface*) tuy nhiên việc sử dụng chúng đòi hỏi kinh nghiệm lập trình.

Có thể lấy các tập tin dll nhập xuất dữ liệu các port máy tính từ mạng Internet và cài vào máy tính, hoặc có thể tự viết.

Sau đây trình bày hai ví dụ dùng Visual C và Delphi để viết tập tin.dll mà Visual Basic sẽ sử dụng.

4.2.1 Dùng Visual C++4.0 (Win 16 bit)

Ta viết tập tin thư viện đặt tên là INOUT.dll gồm thường trình xuất ra cổng đặt tên OUTPORT và thường trình nhập dữ liệu INPORT.

Đầu tiên dùng phần mềm soạn văn bản viết hai tập tin INOUT.DEF (define) và tập tin INOUT.CPP (chương trình C).

```
// Inout.def listing
-----
Library inout
Description dll for I/O card
Exports
    Outport @1
    Inport @2
-----
```

Hàng đầu là tên file INOUT, hàng thứ hai là chú thích, các hàng sau liệt kê tên các thường trình trong thư viện. Sau tên thường trình là dấu @ và số thứ tự.

```
// INOUT.CPP Listing
-----
// file nguồn. CPP cho dll INOUT.dll
# include <stdio.h>
# include <conio.h>
// chứa hàm INP và OUTP
Short_stdcall OUTPORT (int PortAddress, int PortData)
{
    Short Dummy;
    Dummy = (Short) (_outp (PortAddress, PortData));
    Return (Dummy);
};
Short_stdcall INPORT (int PortAddress)
{
    Short PortData;
    PortData = (Short) (_inp (PortAddress));
    Return (PortData);
};
-----
```

Sau khi đã soạn xong hai tập tin INOUT.def và INOUT.cpp ta cất vào một thư mục, ví dụ, C:\port. Vào Visual C chọn File - New - Projects - Win32DLL đánh tên INOUT và địa chỉ C:\port\INOUT rồi bấm OK. Màn hình INOUT - Microsoft Developer Studio sẽ xuất hiện và project INOUT classes được tạo

ra, ta sẽ cộng thêm hai file INOUT.def và INOUT.cpp vào project INOUT rồi bấm FILE - SAVE ALL.

Sau khi đã tạo project INOUT bấm Build và chọn Build INOUT.dll, nếu không có gì sai sót file INOUT.dll sẽ được tạo ra và cất trong C:\port\inout\debug. Sau đó cất file vào thư mục hệ thống của Windows, ví dụ, C:\Windows. Với Visual C 5.0 và 6.0 cách viết sẽ khác đi.

Các chương trình viết trên ngôn ngữ khác như Delphi, Visual Basic có thể gọi hàm INPORT và OUTPORT khi đã khai báo các hàm này và INOUT.dll

Ví dụ, trong Visual Basic ta thêm các dòng này trong phần khai báo chương trình sau phát biểu Option Explicit:

```
-----
Option Explicit
Private Declare Function OUTPORT LIB "INOUT.dll" (ByVal PortAddress As Integer,
By Val Port Data as Integer) As Integer.
Private Declare Function INPORT Lib "INOUT.dll" (ByVal PortAddress As Integer) As
Integer
-----
```

Private cho biết các hàm và thường trình sau Declare là cục bộ. Sau Declare ta viết Function nếu có trả về giá trị, còn không thì dùng SUB, sau đó là tên. Từ LIB và kèm theo cho biết nơi tìm tập tin dll, thường phải kèm đường dẫn.

Sau khi đã khai báo có thể sử dụng các hàm và chương trình con trong chương trình..

Ví dụ, muốn xuất 0 ra địa chỉ 300H ta viết:

```
Dummy = OUTPORT (768, 0)
```

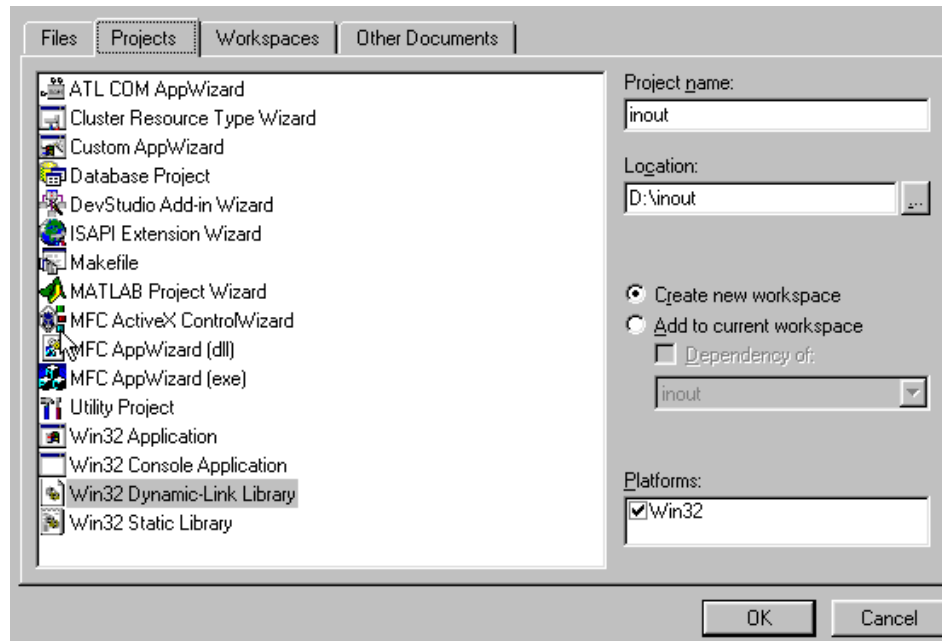
Dummy là biến nguyên đã khai báo (Dim Dummy as Integer;)

Muốn nhập một trị ở cổng Control Port vào biến Value ta viết

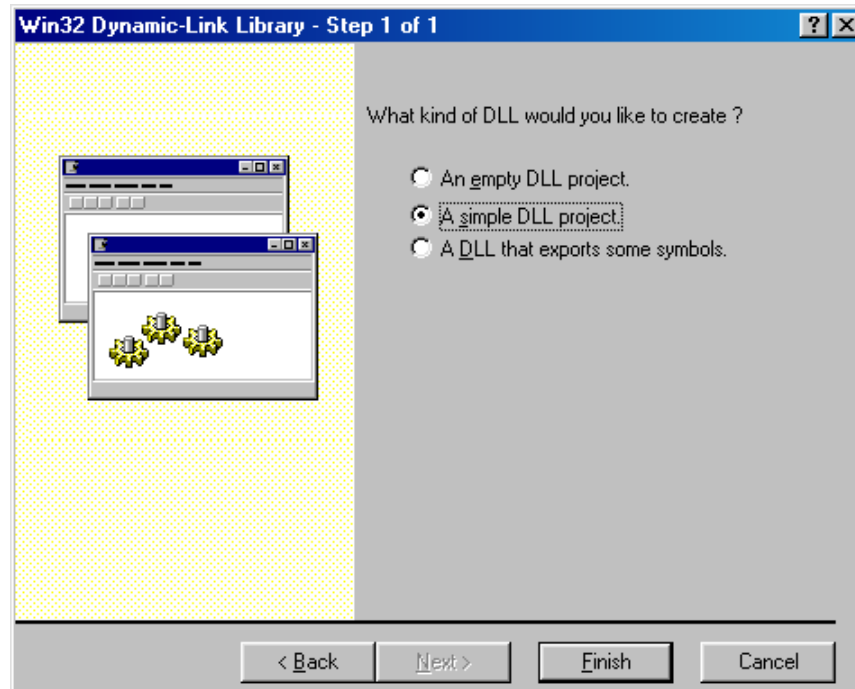
```
Value = INPORT (Control Port)
```

4.2.2 Tạo dll dùng VC++ 6.0 (Win32 bit)

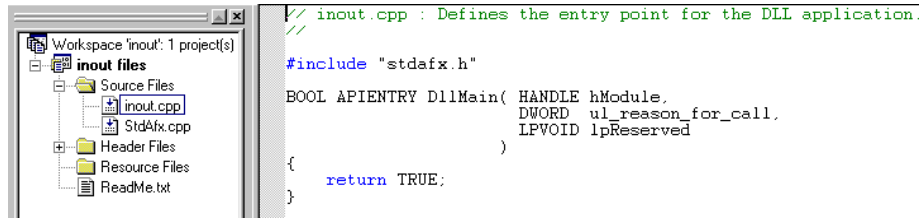
Vào File- New- Projects chọn Win32 Dynamic- Link Library, đặt tên cho project là inout



Sau đó tiếp tục chọn option A Simple DLL Project



Khung soạn thảo chương trình xuất hiện



Ta bổ sung thêm các dòng lệnh vào inout.cpp

```
// inout.cpp : Defines the entry point for the DLL application.
```

```
#include "stdafx.h"
```

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
BOOL APIENTRY DllMain( HANDLE hModule,
                      DWORD ul_reason_for_call,
                      LPVOID lpReserved
                      )
```

```
{
    return TRUE;
}
```

```
int APIENTRY inport(WORD Port)
```

```
{
    return(_inp(Port));
}
```

```
int APIENTRY outport(WORD Port, int ByteData)
```

```
{
    return(_outp(Port, ByteData));
}
```

Sau đó vào File – New -Files, chọn Text File, bổ sung vào Project inout file inout.def

```
//inout.def
```

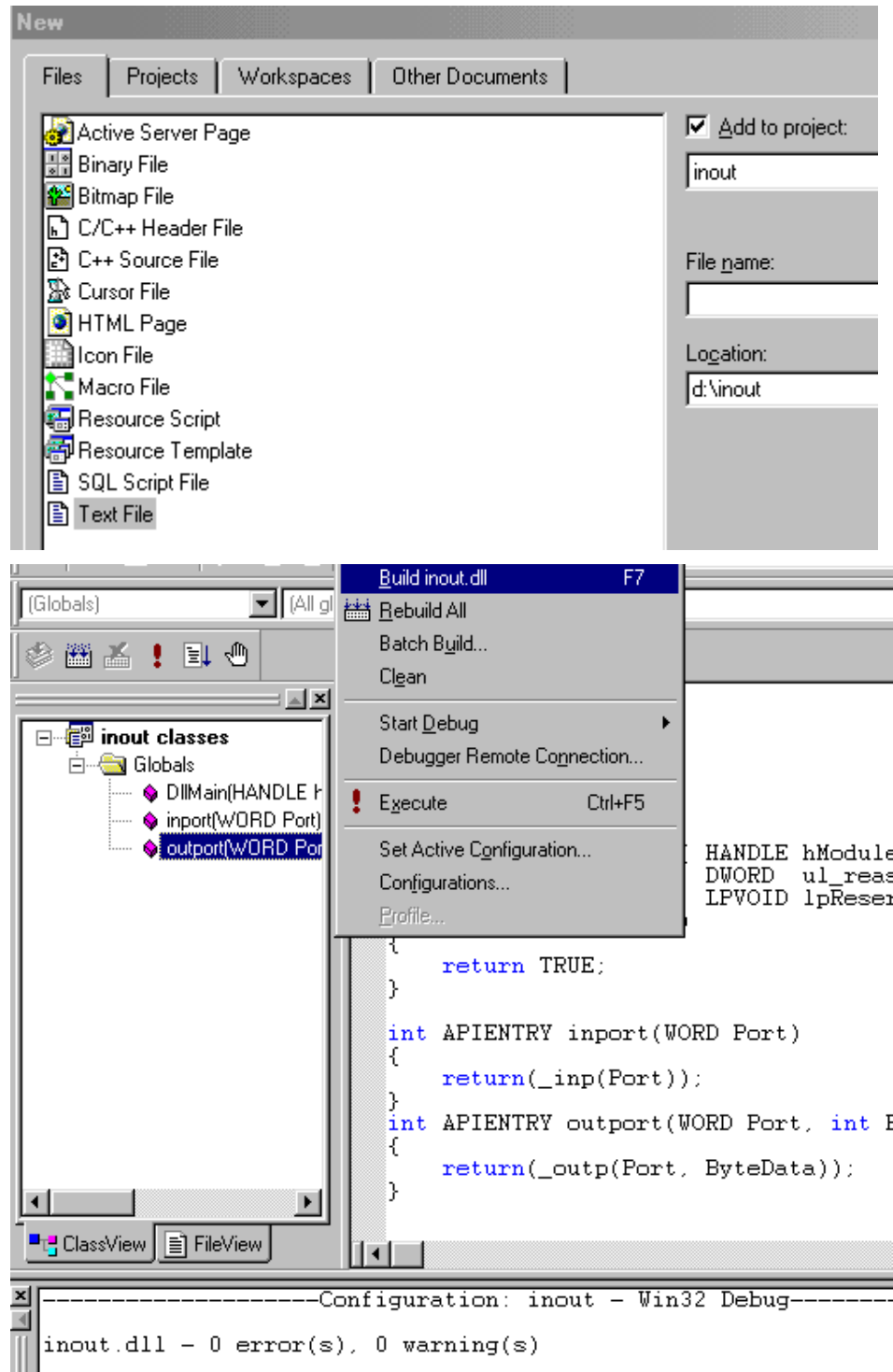
```
LIBRARY inout
```

```
EXPORTS
```

```
inport @1
```

```
outport @2
```

Vào menu Chọn Build- Build inout.dll, file inout được lưu vào thư mục d:\inout\Debug, ta chép vào thư mục c:\Windows

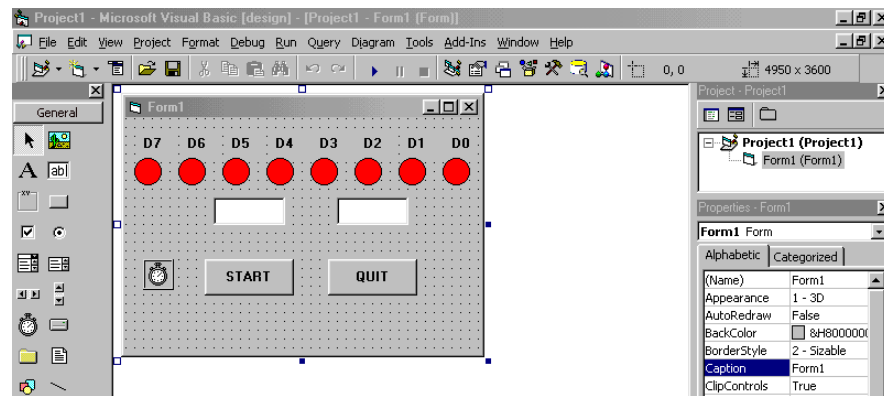


Chương trình sử dụng file inout. dll trong Visual Basic 6.0:

```

Private Declare Function inport Lib "inout.dll" (ByVal portaddress As Integer) As Long
Private Declare Function outport Lib "inout.dll" (ByVal portaddress As Integer, ByVal_
value As Byte) As Long
Private Sub Command1_Click()
    Timer1.Enabled = True
End Sub
Private Sub Command2_Click()
    Timer1.Enabled = False
    Unload Form1
End Sub
Private Sub Form_Load()
    Timer1.Interval = 1000
    Timer1.Enabled = False
End Sub
Private Sub Timer1_Timer()
    Dim i, port, giatri, dummy
    port = 768
    dummy = outport(port + 1, Val(Text1.Text))
    giatri = inport(port)
    Text2.text=giatri
    For i = 0 To 7
        If (giatri And (2 ^ i))=2^i Then
            Lamp(i).FillColor = QBColor(12)
        Else
            Lamp(i).FillColor = QBColor(10)
        End If
    Next
End Sub

```

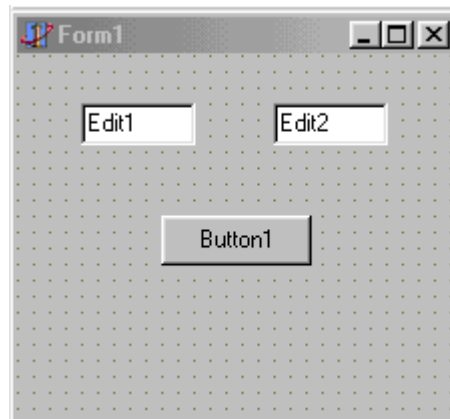


Chương trình Delphi dùng inout.dll:

```

unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;
type
  TForm1 = class(TForm)
    Edit1: TEdit;
    Button1: TButton;
    Edit2: TEdit;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
Function inport (adr: word): byte; stdcall; external 'inout.dll'
Function outport (adr: word; data:byte): byte; stdcall; external'inout.dll'
var
  Form1: TForm1;
implementation
{$R *.DFM}
procedure TForm1.Button1Click(Sender: TObject);
var data: byte;
begin
  data := outport ($378, strtoint ( edit1.text));
  edit2.text := inttostr (inport($379))
end;
end.

```



Ghi chú: Khoảng thời gian trong timer có độ chính xác vài chục ms, do đó đặt thời gian trong timer cần lớn hơn 50ms

4.2.3 Tạo DLL dùng Delphi

Chạy Delphi, vào menu File- New, chọn DLL - OK, trên cửa sổ soạn thảo Project1.dpr xuất hiện khung chương trình sau:

```
library Project1;
uses
  SysUtils,
  Classes;
{$R *.RES}
begin
end.
```

Ta sẽ đổi tên Project1 thành inout và thêm các dòng lệnh trước begin end. Ví dụ sau tạo inout.dll gồm chương trình con output và hàm inport

```
library inout;
uses
  SysUtils,
  Classes;
{$R *.RES}
Procedure output(adr:word;data:word);export; stdcall;
begin
  asm
    push dx
    mov dx,adr
    mov ax,data
    out dx,ax
    pop dx
  end;
end;
Function inport(adr:word):word;export; stdcall;
var data:word;
begin
  asm
    push dx
    mov dx,adr
    in ax,dx
    mov data, ax
    pop dx
```

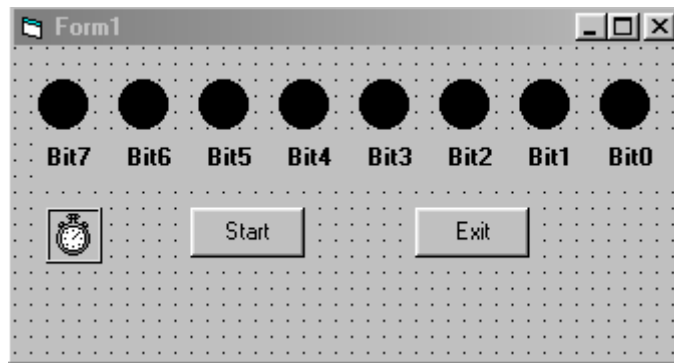
```

    end;
    inport:=data;
    end;
    exports
    outport index 1,
    inport index 2;
    begin
    end.

```

Sau đó vào menu Project- Build inout. File inout.dll được tạo ra trong thư mục \Delphi5\Projects cùng với các file inout.dpr, inout.res, inout.dof, inout.cfg. Ta chép file inout.dll vào thư mục Windows\System.

Ví dụ: Viết chương trình Visual Basic dùng dll inout do Delphi tạo ra



```

Private Declare Function inport Lib "inout.dll" (ByVal portaddress As Integer) As Byte
Private Declare Sub outport Lib "inout.dll" (ByVal portaddress As Integer, ByVal_
value As Byte)
Private Sub Command1_Click()
    Timer1.Enabled = True
End Sub
Private Sub Command2_Click()
    Timer1.Enabled = False
    Unload Form1
End Sub
Private Sub Timer1_Timer()
    Dim i, port, giatri
    port = 768

```

```

    giatri = inport (port+1)
    For i = 0 To 7
    If (giatri And (2 ^ i)) and 2^i Then Lamp(i).FillColor = vbBlue Else_
    Lamp(i).FillColor = vbRed
    Next
End Sub

```

Ví dụ: viết chương trình Delphi dùng inout.dll do Delphi tạo ra

```

unit Unit1;
interface
uses
    Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
    StdCtrls;
type
    TForm1 = class(TForm)
        Edit1: TEdit;
        Button1: TButton;
        Edit2: TEdit;
        procedure Button1Click(Sender: TObject);
    private
        { Private declarations }
    public
        { Public declarations }
    end;
Function inport (adr: word): word; stdcall; external 'inout.dll'
Procedure output (adr: word; data: word); stdcall; external 'inout.dll'
var
    Form1: TForm1;
implementation
{$R *.DFM}
procedure TForm1.Button1Click (Sender: TObject);
begin
    output ($378, strtoint ( edit1.text));
    edit2.text:= inttostr((inport($379)) and $ff)
end;
end.

```

4.3 Xuất nhập với Win 2000 và Win NT

Việc xuất nhập trong DOS và Win 98 thực hiện dễ dàng như

đã trình bày ở các mục trên, tuy nhiên các hệ điều hành Win 2000 và Win NT, Win XP ngăn cản việc thực hiện các lệnh truy cập ngoại vi trực tiếp trong mode người dùng. Muốn vượt qua rào cản này ta phải viết các driver truy cập ngoại vi trong mode kernel sử dụng các hàm WinAPI. Viết các file xuất nhập dll dạng này khá phức tạp và đòi hỏi trình độ lập trình cao, bạn có thể vào các trang web để download về sử dụng, ví dụ một địa chỉ là www.logix4u.net. Một địa chỉ khác www.jungo.com hỗ trợ viết driver cho card ISA, PCI, USB....

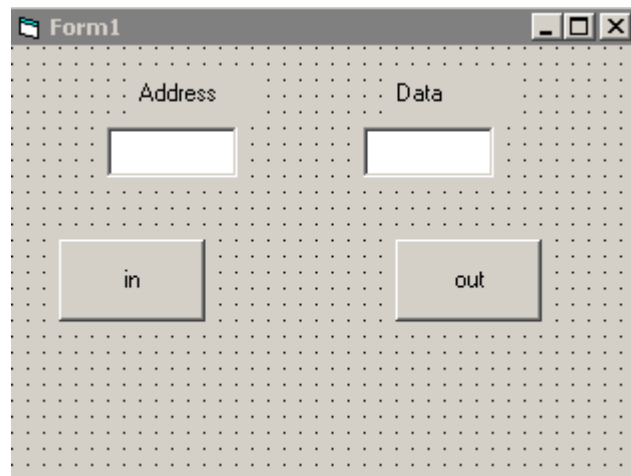
Trang web www.logix4u.net cung cấp thư viện inpout32.dll với hai hàm Inp32 và Out32 khai báo như sau

```
Public Declare Function Inp32 Lib "inpout32.dll" (ByVal PortAddress As Integer) As Integer
```

```
Public Declare Sub Out32 Lib "inpout32.dll" (ByVal PortAddress As Integer, ByVal Value As Integer)
```

Sau đây là ba ví dụ sử dụng thư viện này

Ví dụ: lập trình Visual Basic đọc và xuất port



```
Private Declare Function Inp32 Lib "inpout32.dll" _
    (ByVal PortAddress As Integer) As Integer

Public Declare Sub Out32 Lib "inpout32.dll" _
    (ByVal PortAddress As Integer, ByVal Value As Integer)

Private Sub Command1_Click()
    Text2.Text = Inp32(Val("&H" + Text1.Text))
End Sub
```

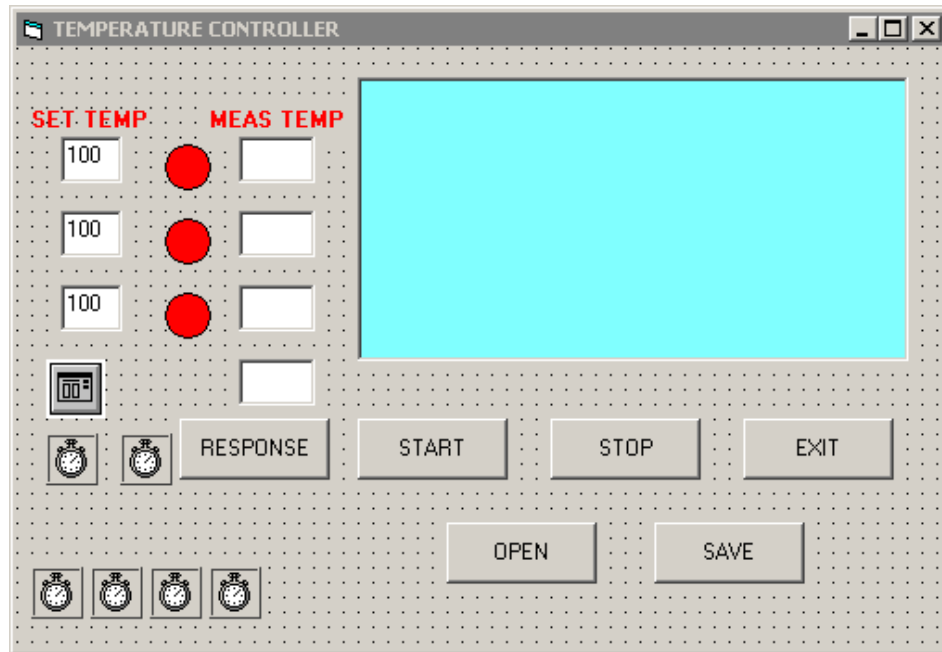
End Sub

Private Sub Command2_Click()

Out32 Val("&H" + Text1.Text), Val(Text2.Text)

End Sub

Ví dụ: lập trình Visual Basic điều khiển nhiệt độ, vẽ đồ thị nhiệt độ



```
Private Declare Function Inp32 Lib "inpout32.dll" (ByVal portadr As Integer) As Integer
Private Declare Sub Out32 Lib "inpout32.dll" (ByVal portadr As Integer, ByVal data As Integer)
Private Declare Sub Sleep Lib "kernel32.dll" (ByVal delay As Long)
Const portADC = &H300
Const portOut = &H308
Dim valout, sh, sw, dem, k
Dim data(1000) As String
Private Sub Form_load()
    CommonDialog1.Filter = "text file (*.dat)|*.dat"
    valout = 0
    dem = 0
    k = 1
    Out32 portOut, valout
    Timer2.Interval = 1000
    Timer2.Enabled = False
    Picture1.ScaleMode = 6
```

```

Picture1.AutoRedraw = True
Picture1.DrawWidth = 2
Picture1.ForeColor = QBColor(12)
sh = Int(Picture1.ScaleHeight)
sw = Int(Picture1.ScaleWidth)
Picture1.Line (0, sh)-(sw, sh)
Picture1.Line (0, sh)-(0, 0)
Picture1.CurrentX = 0
Picture1.CurrentY = sh
End Sub

Private Sub CmdStart_Click()
    CmdStart.Enabled = False
    Timer2.Enabled = True
End Sub

Private Sub Timer2_Timer()
    Dim i, nd, ss, cx
    For i = 0 To 2
        Out32 portADC + i, 0
        Sleep (1)
        nd = Round((Inp32(portADC + i)) * 60 / 51, 2)
        Tdo(i).Text = Str(nd)
        ss = Val(Tdat(i).Text) - nd
        If 20 <= ss Then
            Shape1(i).FillColor = QBColor(10)
            valout = valout Or (2 ^ i)
            Out32 portOut, valout
        ElseIf ((0 < ss) And (ss < 20)) Then
            Timer1(i).Interval = ss * 50
            Timer1(i).Enabled = True
            Shape1(i).FillColor = QBColor(10)
            valout = valout Or (2 ^ i)
            Out32 portOut, valout
        Else
            Shape1(i).FillColor = QBColor(12)
            valout = valout And (Not (2 ^ i))
            Out32 portOut, valout
        End If
    Next
    data(dem) = Tdo(0).Text
    cx = Picture1.CurrentX
    Picture1.Line -(cx + 1, sh - data(dem) * sh / 300)
    dem = dem + 1
End Sub

Private Sub CmdOpen_Click()
    Dim datadraw
    Dim i, j
    i = 0
    j = 8.5
    CommonDialog1.ShowOpen

```

```

If CommonDialog1.FileName <> "" Then
    Open CommonDialog1.FileName For Input As #1
    Form2.Show
    Form2.CurrentX = j
    Form2.CurrentY = 60

    Do While Not EOF(1) ' Loop until end of file.

        Line Input #1, datadraw ' Read line into variable.
        Form2.Line -(j + 1, 60 - Val(datadraw) * 50 / 300)
        j = Form2.CurrentX
        i = i + 1
    Loop
    Close #1 ' Close file.
End If
End Sub

Private Sub CmdSave_Click()
Dim i
    CommonDialog1.Filter = "text file (*.dat)*.dat"
    CommonDialog1.ShowSave
    If CommonDialog1.FileName <> "" Then
        Open CommonDialog1.FileName For Append As #1
        For i = 0 To dem
            Print #1, , data(i)
        Next
        Close #1
    End If
End Sub

End Sub

Private Sub CmdStop_Click()
    CmdStart.Enabled = True
    Timer2.Enabled = False
End Sub

Private Sub Command1_Click()
    Timer2.Enabled = False
    Timer3.Enabled = True
End Sub

Private Sub Timer1_Timer(Index As Integer)
    Shape1(Index).FillColor = QBColor(12)
    Timer1(Index).Enabled = False
    valout = valout And (Not (2 ^ Index))
    Out32 portOut, valout
End Sub

Private Sub Timer3_Timer()
Dim y
    y = 1 - Exp(-k / 10)

```

```

        Text1.Text = Str(Round(y, 2))
        cx = Picture1.CurrentX
        Picture1.Line -(cx + 1, sh - y * 30)
        k = k + 1
    End Sub

    Private Sub Timer4_Timer()
        Form1.Caption = "TEMPERATURE CONTROLLER " & Now
    End Sub

    Private Sub CmdExit_Click()
        End
    End Sub

```

Ví dụ; lập trình điều khiển PID tám lò nhiệt

```

Private Declare Function Inp32 Lib "inout32.dll" (ByVal add As Integer) As Integer
Private Declare Sub Out32 Lib "inout32.dll" (ByVal add As Integer, ByVal data As Integer)
Private Declare Sub Sleep Lib "Kernel32.dll" (ByVal delay As Long)
Const n0 = 30
Const n300 = 250

Const umax = 10
Const Ts = 1
Const KD = 5
Const KP = 10
Const KI = 0.5
Const a2 = KD / Ts
Const a0 = KP + KI * Ts + a2
Const a1 = -KP - 2 * a2

Dim control
Dim ek1(8), ek2(8), uk1(8)
Private Sub Command1_Click()
    Timer1.Enabled = True
End Sub

Private Sub Command2_Click()
    Timer1.Enabled = False
    control = 0
    Out32 &H318, control
End Sub

End Sub

Private Sub Form_Load()
    control = 0
    Out32 &H318, control
    Timer1.Interval = 1000
    Timer1.Enabled = False
    For i = 0 To 7
        ek1(i) = 0
    
```

```

ek2(i) = 0
uk1(i) = 0
Next

```

```
End Sub
```

```

Private Sub Timer1_Timer()
    Dim ek, t, a
    For i = 0 To 7
        Out32 (&H310 + i), 0
        Sleep (1)
        a = Inp32(&H310 + i)
        a = Round(((a And &HFF) - n0) * 300 / (n300 - n0), 1)
        Text1(i).Text = a
        ek = Val(Text2(i).Text) - a
        u = uk1(i) + a0 * ek + a1 * ek1(i) + a2 * ek2(i)
        If i = 0 Then
            Text3.Text = u
        End If
        ek2(i) = ek1(i)
        ek1(i) = ek
        If (u > 10) Or (u = 10) Then
            control = control And 2 ^ i
            Out32 &H318, control
            Shape1(i).FillColor = vbBlue
            u = 10
        End If
        If (u < 0) Or (u = 0) Then
            Shape1(i).FillColor = vbRed
            control = control And Not (2 ^ i)
            Out32 &H318, control
            u = 0
        End If
        If (u < 10) And (u > 0) Then
            t = Round(100 * u)
            control = control And 2 ^ i
            Out32 &H318, control
            Shape1(i).FillColor = vbBlue
            Timer2(i).Interval = t
            Timer2(i).Enabled = True
        End If
        uk1(i) = u
        'dk onoff
        'If (data < Val(Text2(i).Text)) Then
        ' control = control Or 2 ^ i
        ' Shape1(i).FillColor = vbBlue
        ' Out32 &H318, control
        'Else
        ' control = control And (Not 2 ^ i)
        ' Shape1(i).FillColor = vbRed
        ' Out32 &H318, control
        'End If
    Next

```

End Sub

```
Private Sub Timer2_Timer(i As Integer)
    Timer2(i).Enabled = False
    Shape1(i).FillColor = vbRed
    control = control And Not (2 ^ i)
    Out32 &H318, control
End Sub
```

Ví dụ: lập trình Visual C đọc và xuất port

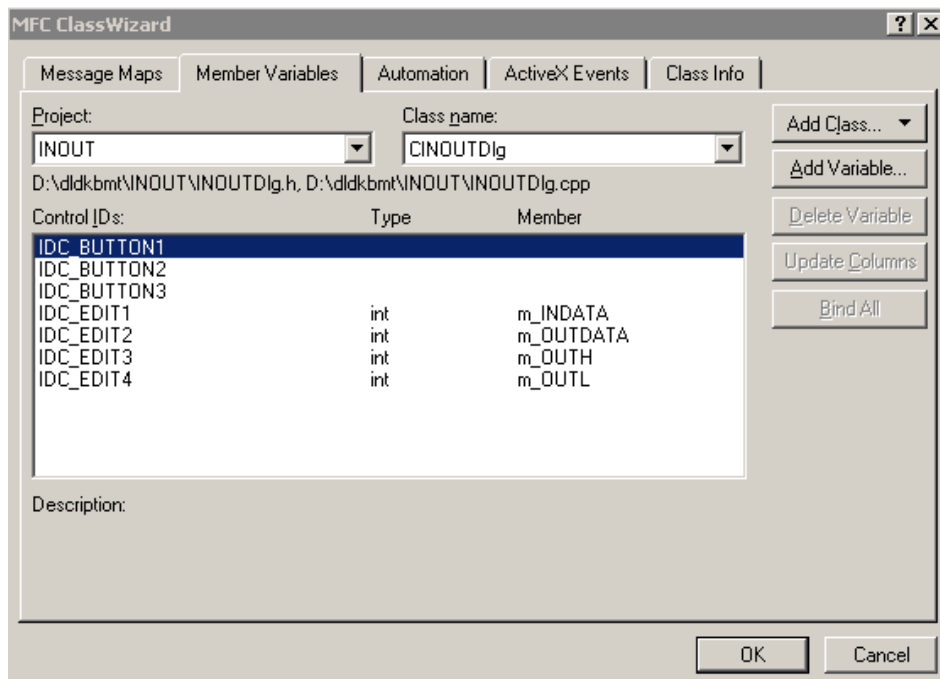
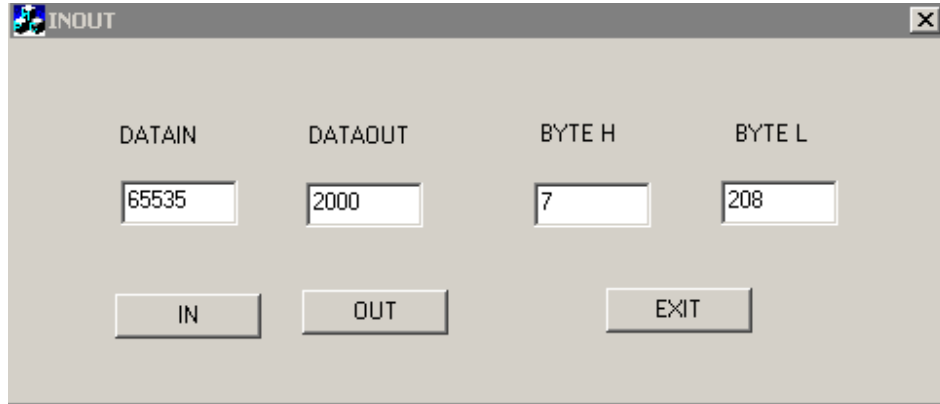
```
#include "stdafx.h"
#include "stdio.h"
#include "string.h"
#include "stdlib.h"
/* ----Prototypes of Inp and Outp--- */
short _stdcall Inp32(short PortAddress);
void _stdcall Out32(short PortAddress, short data);

int main(int argc, char* argv[])
{
    int data;
    if(argc<3)
    {
        printf("Error : too few arguments\n\n***** Usage *****\n\nInpoutTest read <ADDRESS> \nor \nInpoutTest write <ADDRESS> <DATA>\n\n\n\n");
    }
    else if(!strcmp(argv[1],"read"))
    {
        data = Inp32(atoi(argv[2]));
        printf("Data read from address %s is %d \n\n\n",argv[2],data);
    }
    else if(!strcmp(argv[1],"write"))
    {
        if(argc<4)
        {
            printf("Error in arguments supplied");
            printf("\n***** Usage *****\n\nInpoutTest read <ADDRESS> \nor \nInpoutTest write <ADDRESS> <DATA>\n\n\n\n");
        }
        else
        {
            Out32(atoi(argv[2]),atoi(argv[3])); printf("data written to %s\n\n",argv[2]);
        }
    }
    return 0;
}
```

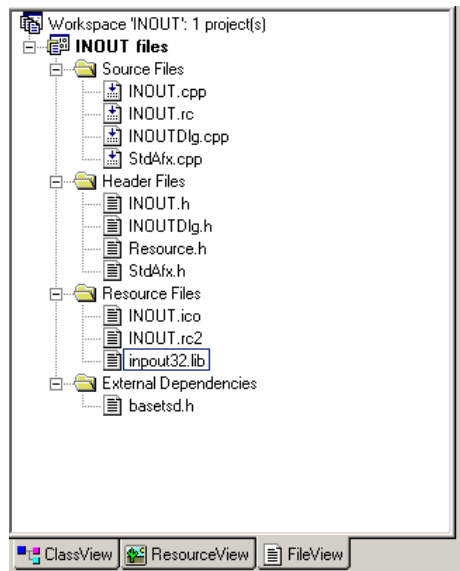
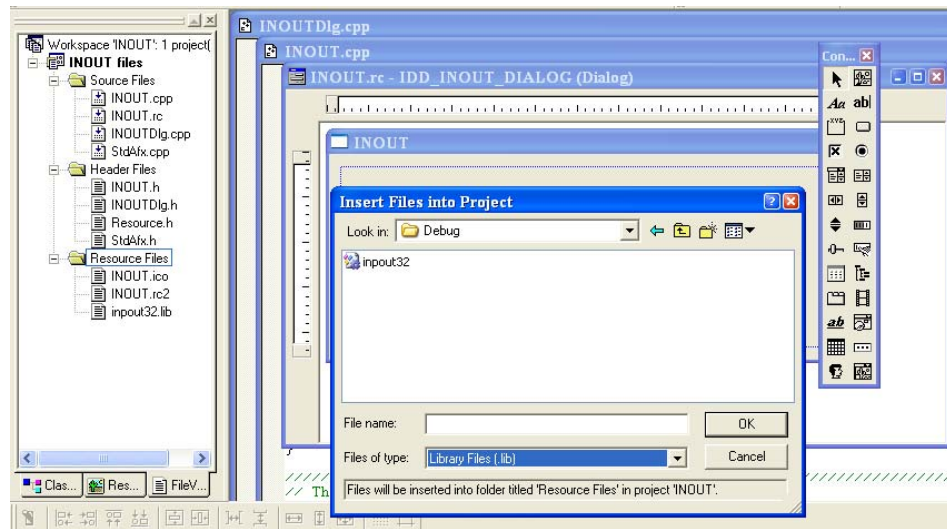
Chương trình chạy trong cửa sổ command với dòng lệnh
inpoutTest read <ADDRESS>

inputTest write <ADDRESS> <DATA>

Ví dụ lập trình VC trong windows, xuất dữ liệu ra port 302, 303 và đọc dữ liệu port 300, 301



Chú ý là trong tab File View/Resource Files ta phải add thêm file inpout32.lib bằng cách bấm chuột phải vào Resource Files, chọn Add Files to Folder, tìm file inpout32.lib



```
// INOUTDlg.cpp : implementation file
#include "stdafx.h"
#include "INOUT.h"
#include "INOUTDlg.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

short _stdcall Inp32(short PortAddress);
void _stdcall Out32(short PortAddress, short data);
```

```

void CINOUTDlg::OnButton1()
{
    int dh;
    int dl;
    dh= Inp32(0x300);
    dl= Inp32(0x301);
    UpdateData(TRUE);
    m_INDATA= dh*256 +dl;
    UpdateData(FALSE);
}

void CINOUTDlg::OnButton2()
{
    int dh;
    int dl;
    UpdateData(TRUE);
    dh= m_OUTDATA /256;
    dl= m_OUTDATA % 256;
    Out32(0x302, dh);
    Out32(0x303, dl);

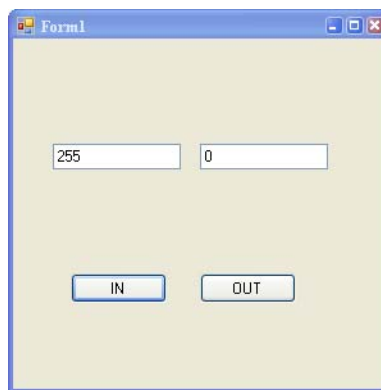
    m_OUTH=dh;
    m_OUTL=dl;
    UpdateData(FALSE);
}

void CINOUTDlg::OnButton3()
{
    OnOK();
}

```

LẬP TRÌNH TRONG VB.NET

Cách lập trình tương tự VB6.0



```

Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
        Handles MyBase.Load
            TextBox2.Text = 0
        End Sub
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
        Handles Button1.Click
            TextBox1.Text = Inp(&H378S)
        End Sub
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
        Handles Button2.Click
            Dim dataout As Short
            dataout = TextBox2.Text
            Out(&H378S, dataout)
        End Sub
End Class

```

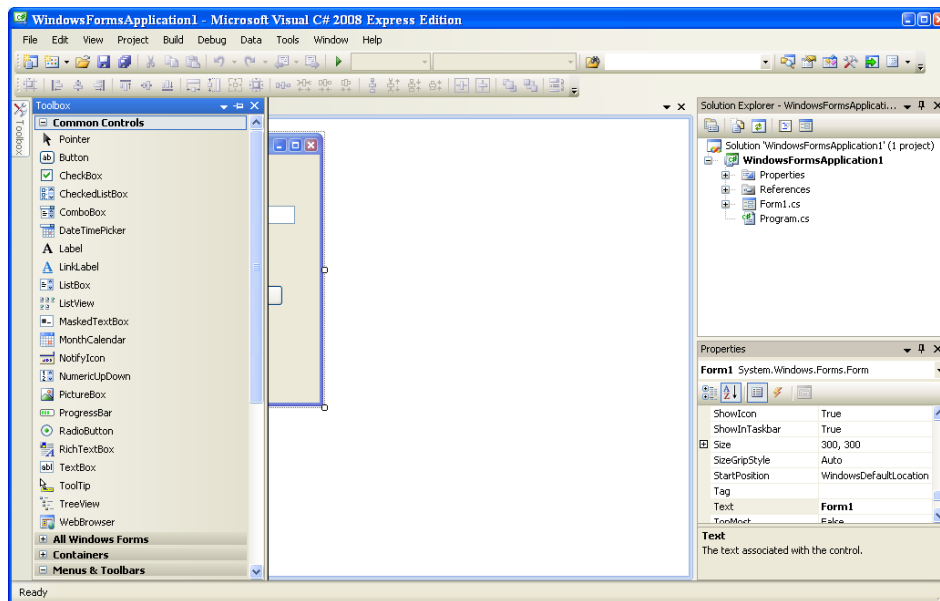
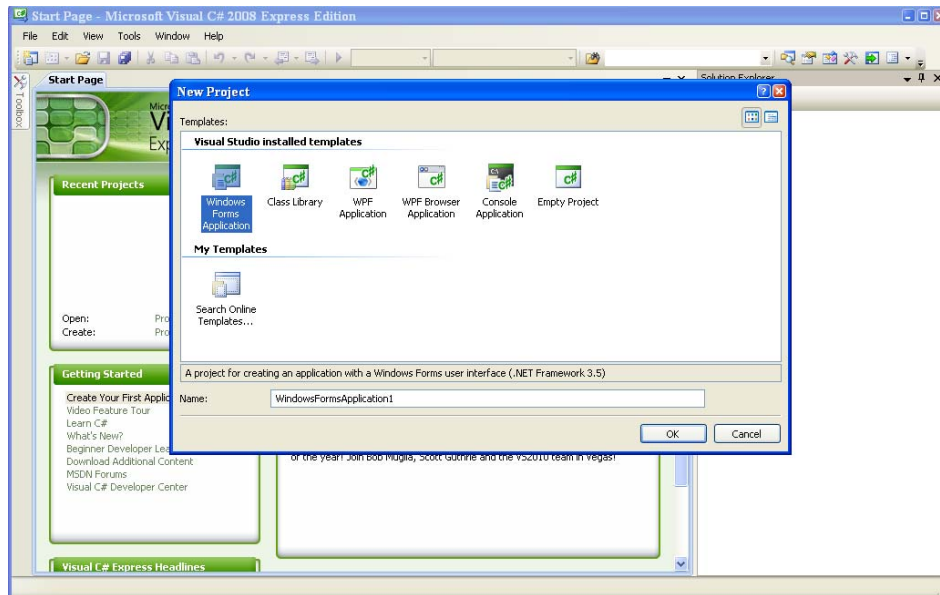
Trước Form Load ta thêm các dòng sau:

```

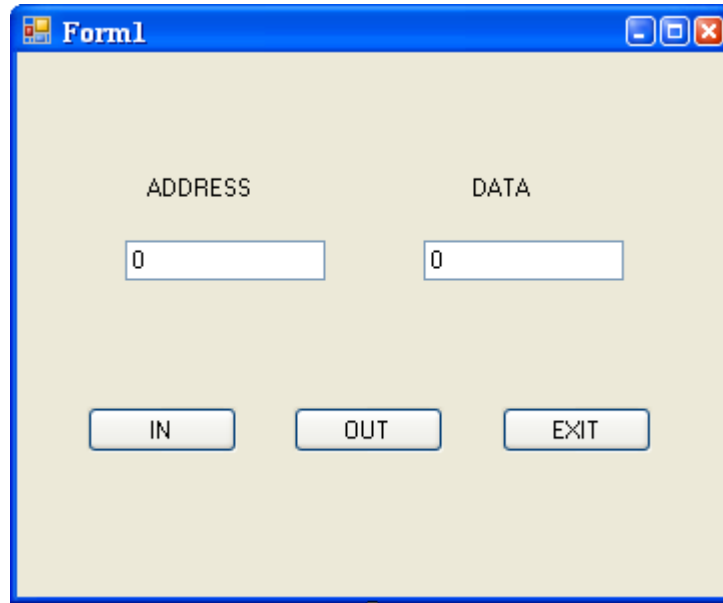
Module InpOut32_Declarations
    'Inp and Out declarations for port I/O using inpout32.dll.
    Public Declare Function Inp Lib "inpout32.dll" Alias "Inp32" (ByVal PortAddress As Short) As Short
    Public Declare Sub Out Lib "inpout32.dll" Alias "Out32" (ByVal PortAddress As Short, ByVal Value As Short)
End Module

```

LẬP TRÌNH TRONG C#



Tạo ba Button và hai Textbox



Khai báo class: add new class

```
using System.Runtime.InteropServices;
public class PortAccess
{
    [DllImport("inpout32.dll", EntryPoint="Out32")]
    public static extern void Output(int address, int
value);

    [DllImport("inpout32.dll", EntryPoint="Inp32")]
    public static extern int Input( int adress);
}
```

Code trong form.cs:

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

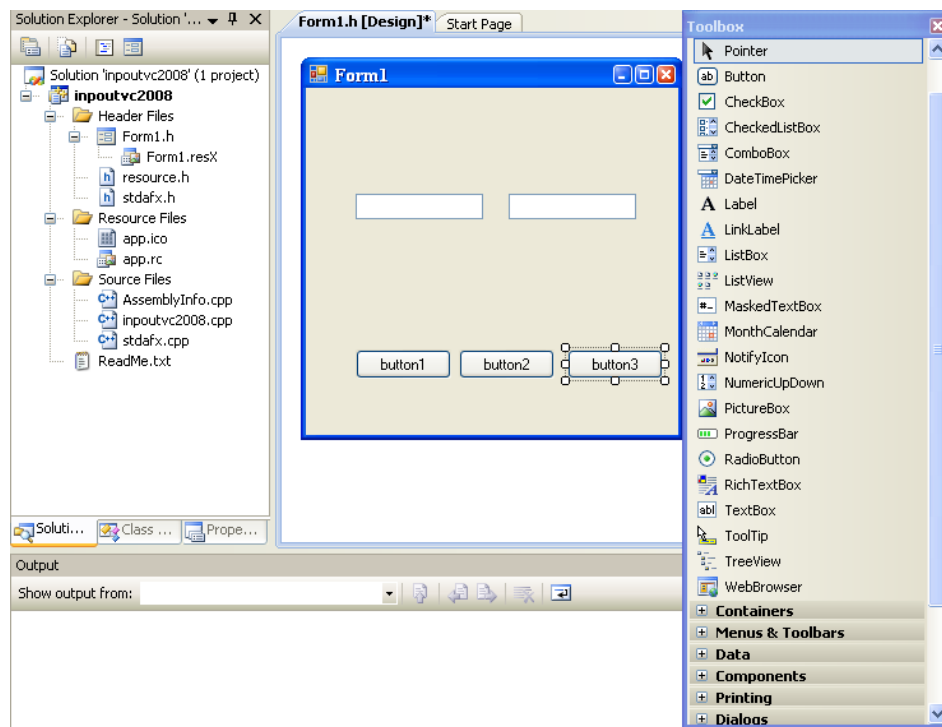
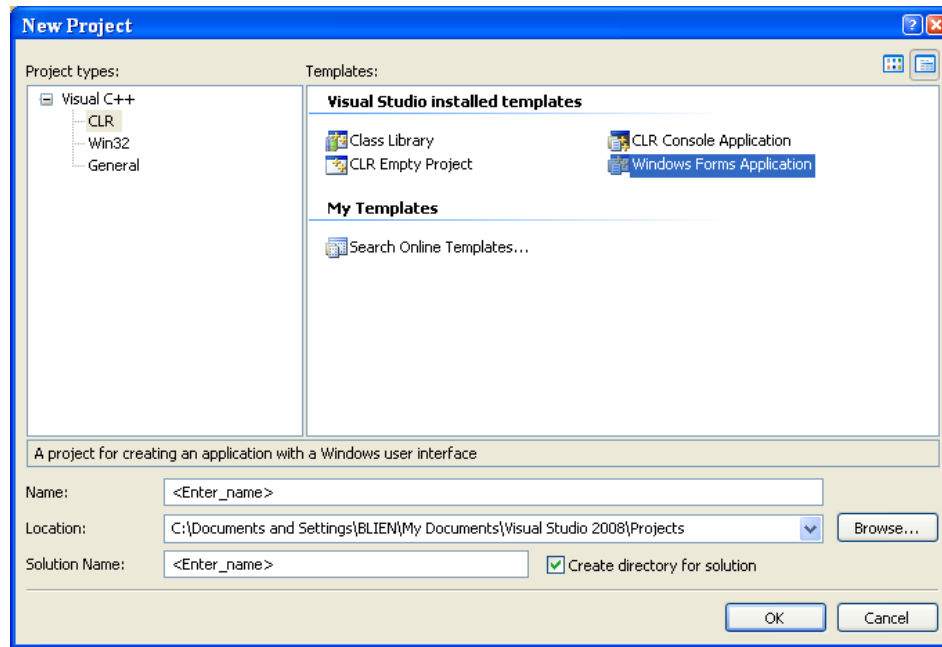
namespace WindowsFormsApplication1
{
```

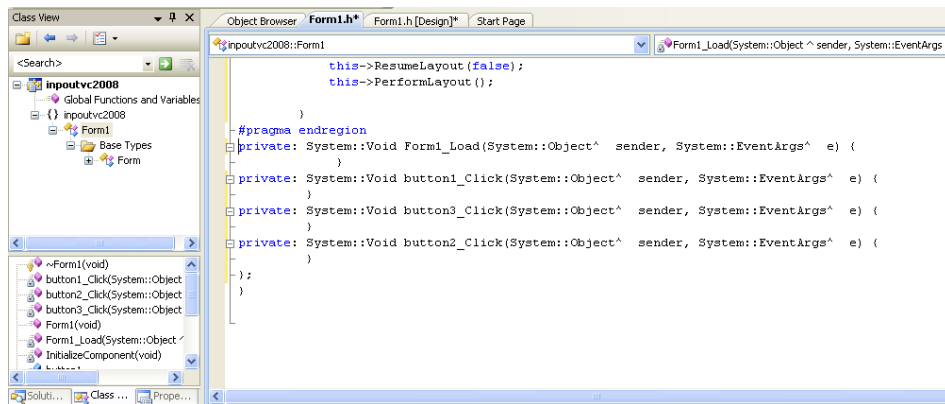
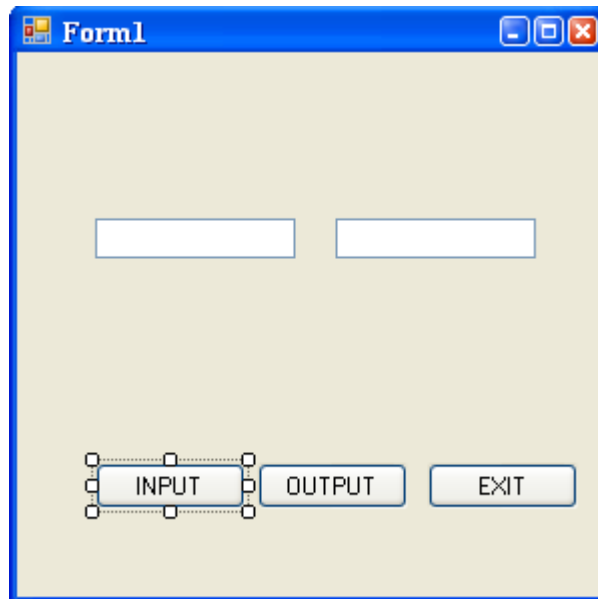
```
public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();
    }

    private void button2_Click(object sender, EventArgs e)
    {
PortAccess.Output(Convert.ToInt16(textBox1.Text),Convert.ToInt16(text
Box2.Text));
    }

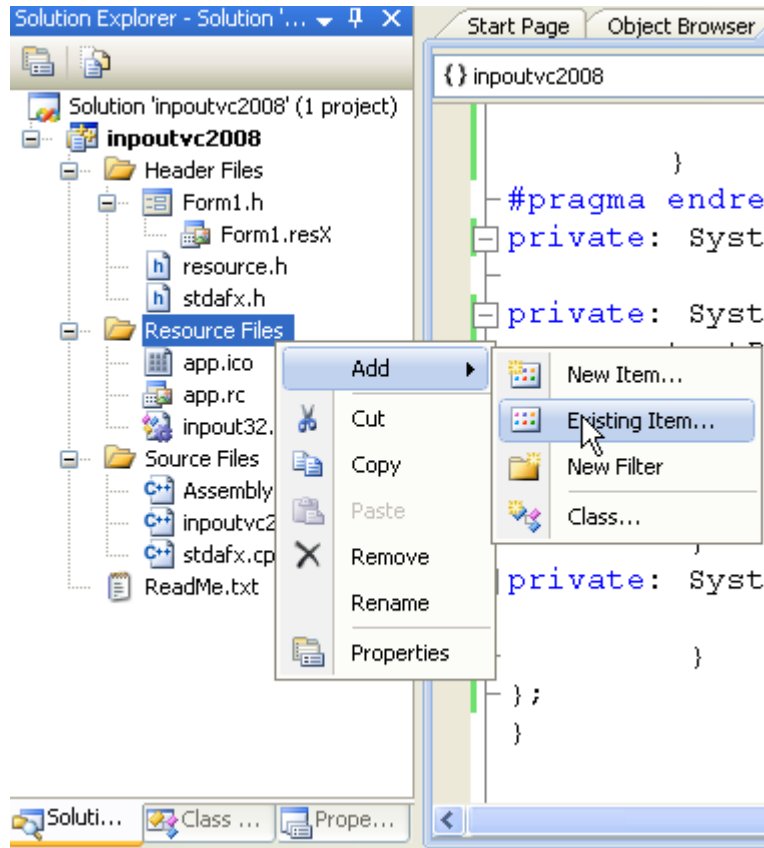
    private void button1_Click(object sender, EventArgs e)
    {
        textBox2.Text =
Convert.ToString(PortAccess.Input(Convert.ToInt16(textBox1.Text)));
    }
    private void button3_Click(object sender, EventArgs e)
    {
        this.Close();
    }
}
}
```

Lập trình trong VC2008
Tạo Project





Vào mục Resource Fles thêm file inpout32.lib



Trên đầu của file form1.h thêm các dòng khai báo

```
short _stdcall inp32(short portaddress);
```

```
void _stdcall out32(short portaddress, short data);
```

Các đoạn mã ứng với các button như sau:

```
private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e) {
    textBox2->Text= Convert::ToString(Inp32(0x300));
}
private: System::Void button3_Click(System::Object^ sender, System::EventArgs^ e) {
    this->Close();
}
private: System::Void button2_Click(System::Object^ sender, System::EventArgs^ e) {
    Out32(Convert::ToUInt16(textBox1-> Text),Convert:: ToUInt16(textBox2->Text));
}
```

NGẮT TRONG ĐIỀU KHIỂN

Khi đọc dữ liệu từ cổng ngoại vi thường dùng hai phương pháp. Phương pháp thứ nhất là thăm dò bằng cách đọc trạng thái của cổng theo chu kỳ để xem có gì thay đổi không, nếu có sẽ chuyển sang một chương trình phục vụ. Phương pháp này tốn thời gian và làm chậm các quá trình khác.

Phương pháp thứ hai là dùng ngắt, vi xử lý thực hiện công việc bình thường, khi ngoại vi có thông tin cần gửi đến sẽ gửi yêu cầu ngắt IRQ (*Interrupt Request*) đến vi xử lý, lúc đó VXL chấm dứt lệnh đang thực hiện, cất một số thông tin vào ngăn xếp và thực hiện thường trình phục vụ ngắt ISR (*Interrupt Service Routine*) đã định sẵn. Khi đã thực hiện xong ISR, VXL quay trở lại nơi đã rời khỏi.

Có tất cả 256 ngắt cho họ VXL 8086, một số ngắt (đa số) không liên quan gì đến ngoại vi, số ít còn lại là ngắt cứng phục vụ cho việc giao tiếp VXL với ngoại vi như ngắt thời gian, bàn phím, con chuột, cổng song song, cổng nối tiếp, đĩa cứng, đĩa mềm, card âm thanh...

VXL quản lý ngắt qua vector ngắt, vector ngắt cho biết địa chỉ chứa ISR cho mỗi loại ngắt. Bảng các địa chỉ gọi là bảng vector ngắt (bảng 4.1). Mỗi địa chỉ chiếm bốn byte

Các ngắt ký hiệu INT N có địa chỉ ISR ở trong các ô nhớ: gồm địa chỉ đoạn CS và địa chỉ tương đối IP, địa chỉ vật lý CS x 16 + IP

N x 4	}	IP
N x 4 + 1		
N x 4 + 2	}	CS
N x 4 + 3		

Các ngắt được xếp ưu tiên từ cao đến thấp như sau:

INT0; INTn; NMI; IRQ0; IRQ1; IRQ8 ÷ IRQ15;

IRQ3; IRQ4, IRQ5; IRQ6; IRQ7; INT1

INTn là các ngắt mềm và ngắt ngoại lệ.

Vi xử lý xử lý ngắt ngoài thông qua vi mạch điều khiển ưu tiên ngắt PIC (*Priority Interrupt Controller*), các mainboard đời cũ dùng hai PIC 8259A còn các mainboard đời mới tích hợp trong một chip đa năng theo công nghệ ASIC (*chip set*)

PIC có các thanh ghi phục vụ xử lý ngắt

PIC 1: ngắt IRQ7, 6, 5, 4, 3, 2, 1, 0 có địa chỉ 20H và 21H

Bảng 4.1: Bảng vector ngắt

INT (Hex)	IRQ	Sử dụng
00 - 01	Exception Handlers	Chia cho zero
02	IRQ không che (NMI)	Sai parity
03 - 07	Exception Handlers	Xử lý các ngoại lệ
08	Ngắt cứng IRQ0	Timer Hệ thống
09	Ngắt cứng IRQ1	Bàn phím
0A	Ngắt cứng IRQ2	Chuyển hướng sang IRQ9
0B	Ngắt cứng IRQ3	Cổng nối tiếp 2, 4
0C	Ngắt cứng IRQ4	Cổng nối tiếp 1, 3
0D	Ngắt cứng IRQ5	Card âm thanh
0E	Ngắt cứng IRQ6	Đĩa mềm
0F	Ngắt cứng IRQ7	Cổng song song
10 ÷ 6F	Ngắt mềm	Ngắt do phần mềm
70	Ngắt cứng IRQ8	Đồng hồ thời gian thực
71	Ngắt cứng IRQ9	Chuyển hướng từ IRQ2
72	Ngắt cứng IRQ10	Dự trữ
73	Ngắt cứng IRQ11	Dự trữ
74	Ngắt cứng IRQ12	Con chuột PS/2
75	Ngắt cứng IRQ13	Đồng xử lý
76	Ngắt cứng IRQ14	Đĩa cứng
77	Ngắt cứng IRQ15	Dự trữ
78 - FF	Ngắt mềm	Ngắt do phần mềm

Bảng 4.2 Các thanh ghi của PIC

Địa chỉ	Đọc/Viết	Chức năng
20H	Viết	Từ điều khiển khởi động 1 ICW1
	Viết	Từ điều khiển hoạt động 2 OCW2
	Viết	Từ điều khiển hoạt động 3 OCW3
	Đọc	Yêu cầu ngắt IRR
	Đọc	Phục vụ ngắt ISR
21H	Viết	Từ điều khiển khởi động 2 ICW2
	Viết	Từ điều khiển khởi động 3 ICW3
	Viết	Từ điều khiển khởi động 4 ICW4
	Đọc/Viết	Mặt ra ngắt IMR Từ điều khiển hoạt động 1 OCW1

PIC2: ngắt IRQ15, 14, 13, 12, 11, 10, 9, 8 có địa chỉ tương ứng là A0H và A1H

- Muốn che một ngắt nào đó ta cho bit tương ứng của nó trong OCW1 bằng 1. Ví dụ, muốn cho phép IRQ3 ta gởi byte F7 đến địa chỉ 21H lúc này các IRQ 0 ÷ 7 trừ IRQ3 sẽ bị cấm.

- Muốn không ảnh hưởng đến các IRQ khác ta dùng lệnh: Outportb (OX21, (inportb (0X21) & 0XF7)); (trong C)

- Muốn che IRQ3 ta gởi lệnh:

Outportb (OX21, (inportb (OX21) | 0X08));

- Muốn sử dụng ngắt ta phải viết chương trình phục vụ ngắt ISR, đặt địa chỉ của chương trình này vào vị trí phù hợp trên bảng vector ngắt, trước đó cần phải cất địa chỉ đã có sẵn để sau đó phục hồi trở lại. Khi có ngắt xảy ra và ISR thực hiện xong phải báo trở lại cho PIC bằng cách gởi EOI (*end of interrupt*) đến OCW2, thông thường là byte 20H (cho ngắt thường). Việc khởi động PIC do ROM BIOS đảm nhiệm ta không cần quan tâm đến.

Ví dụ : Lập trình ngắt dùng C

```
#include <process.h>
#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <conio.h>

void main(void)
{
    No_irq = 5;
    init_isr(No_irq);
    { /*Thêm mã*/
    }
    close_isr(No_irq);
}

/*  INIT INTERRUPT SERVICE ROUTINE */
void init_isr (int irq_num)
{
    disable();
    if ( irq_num < 8 )
        old_handler1 = getvect(irq_num+8);
```

```

else
    old_handler1 = getvect(irq_num-8+0x70);
if ( irq_num < 8 )
    setvect(irq_num+8, isr);
else
    setvect(irq_num-8+0x70, isr);
if ( irq_num < 8 )
    {
        int_mask = inportb(0x21) & ~(0x01<<(irq_num));
        outportb(0x21, int_mask);
    }
else
    {
        int_mask = inportb(0xa1) & ~(0x01<<(irq_num-8));
        outportb(0xa1, int_mask);
    }
enable( );
}

/*  CLOSE  INTERRUPT SERVICE ROUTINE      */
void close_isr(int irq_num)
{
    int int_mask;
    disable();
    if ( irq_num < 8 )
    {
        int_mask = inportb(0x21) | (0x01<<irq_num);
        outportb(0x21,int_mask);
        setvect(irq_num+8,old_handler1);
    }
    else
    {
        int_mask = inportb(0xa0) | (0x01<<(irq_num-8));
        outportb(0xa1,int_mask);
        setvect(irq_num-8+0x70,old_handler1);
    }
    enable();
}

/* INTERRUPT SERVICE ROUTINE      */
void interrupt_isr(void)

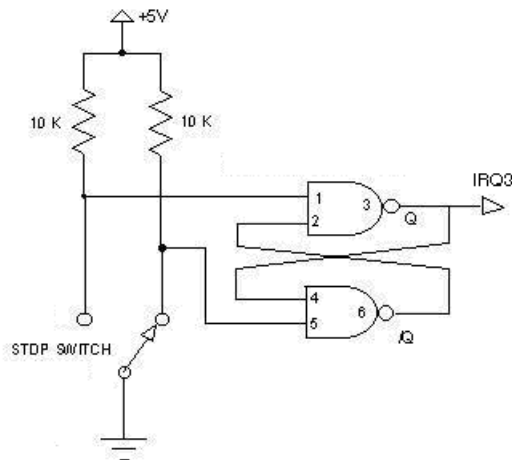
```

```

{
    disable();
    /* thêm mã */
}
outportb(0x20,0x20);
if ( DDA_irq > 7 )
    { outportb(0xa0,0x20); }
enable();
}

```

Ví dụ: đếm số lần nút nhấn tác động dùng ngắt IRQ3



```

#include< stdio.h >
#include< stdlib.h >
#include< dos.h >
#include< conio.h >

void interrupt (*oldIrq3)(void);
void interrupt countToggle(void);
int i = 0;
long j = 0;
#define IRQ3 0x0B /* IRQ3 address */
int main(void)
{
    window(5,5,50,75);
    clrscr();
    gotoxy(1,3);
    printf("Do-while loop iteration # ");
    oldIrq3 = getvect(IRQ3); /* lưu vectơ ngắt cũ */
    setvect(IRQ3, countToggle); /* cài vectơ ngắt mới */
    outportb(0x21, ( inportb(0x21) & 0xF7 )); /* cho phép IRQ3 */
    /*Vòng lặp, khi bấm phím thì thoát khỏi chương trình, khi nhấn nút thì vào ISR*/
}

```

```

do {
    j++;
    gotoxy(27,3); cprintf("%ld\n", j);
} while(!kbhit());
/* Có phím bấm, thoát khỏi chương trình chính */
setvect(IRQ3, oldIrq3);
outportb(0x21, (inportb(0x21) | 0x08) ); /* cấm IRQ3 */
/* Số lần bấm phím*/
printf("\nswitch presses i = %d\n", i);
printf("j = %ld\n", j);
return 0;
} /* end of main */
/* Chương trình phục vụ khi IRQ3 lên mức cao */
void interrupt countToggle(void)
{
    disable();
    i++;
    outportb(0x20, 0x20); /* send EOI signal */
    enable();
}

```

Ví dụ: Chương trình đọc kết quả chuyển đổi của card ADC 8 bit, mỗi khi đổi xong vi mạch cho tín hiệu EOC tác động lên IRQ3, kết quả chuyển đổi lưu vào file văn bản

```

#include < stdio.h >
#include < stdlib.h >
#include < dos.h >
#include < conio.h >
#define IRQ3      0x0b    /* IRQ3 */
#define BASEADDRESS 608    /*Địa chỉ gốc */
#define TRUE      1
#define FALSE     0
#define MAXSIZE   5000    /* Lấy 5000 mẫu */
/* globals */
int  EOC;                /* Đổi xong */
int  readData;           /* Kết quả đổi thập phân */
float i;                 /* Số lần lặp */
float data[MAXSIZE][2];  /* Mảng chứa các kết quả chuyển đổi */
FILE* fp;               /*Con trỏ đến file ASCII */
int  j;                 /* Biến đếm */

/* prototypes */
void interrupt (*oldIrq3)(void);
void interrupt eocTrue(void); /*

int main(void)
{
    clrscr();
    fp = fopen("data.txt", "wt"); /* file chứa kết quả */
    /* khởi động mảng về 0.0 */
    for(j= 0; j < MAXSIZE; j++) {

```

```

        data[j][0] = 0.0;
        data[j][1] = 0.0;
    };
    printf("data array initialized\n");
    i = 0.0;
    oldIrq3 = getvect(IRQ3);
    setvect(IRQ3, eocTrue);
    outportb(0x21, ( inportb(0x21) & 0xF7 ) );
    EOC = FALSE;
    do {
        readData = inportb(BASEADDRESS); /*Kích đổi */
        while(EOC == FALSE) { /* Chờ EOC ON */ };
        readData = inportb(BASEADDRESS); /* Đọc kết quả*/
        EOC = FALSE; /* Xóa EOC */
        data[i][0] = i;
        data[i][1] = (float)(readData * 5.0/255.0); /* Đổi ra volt */
        i++;
    } while(i < MAXSIZE);
    printf("Writing to file...\n");
    for(j=0; j < MAXSIZE; j++) {
        fprintf(fp, "%f\t%f\n", data[j][0], data[j][1]);
    };
    printf("done\n");
    fclose(fp);
    setvect(IRQ3, oldIrq3);
    outportb(0x21, (inportb(0x21) | 0x08) );
    printf("Bye!\n");
    return 0;
} /* end of main */

/* Gọi ISR cho EOC ON mỗi khi IRQ3 mức cao*/
void interrupt eocTrue(void)
{
    #pragma asm pushf;
    #pragma asm cli;
    EOC = TRUE;
    outportb(0x20, 0x20);
    #pragma asm popf;
    return;
} /* end of eofTrue */

```

Ví dụ: lập trình ngắt trong Turbo Pascal

```

unit ngat;
interface
uses dos,crt;
{$I+}
var
Interrupt9 : procedure ;
Procedure Int9; interrupt;

```

```

Procedure SetInt9;
Procedure ResetInt9;
IMPLEMENTATION
{ Interrupt subroutine }
Procedure Int9;
begin
    { call original interrupt }
    Interrupt9;
    {insert code here}
end;
Procedure SetInt9;
var R : registers;
begin
    { save current IRQ9 }
    getintvec($71,@Interrupt9); {$71: interrupt of IRQ9}
    { load new IRQ9 }
    setintvec($71,@nt9);
end;
Procedure ResetInt9;
var R : registers;
begin
    { restore IRQ9 }
    setintvec($71,@Interrupt9);
end;
Begin
End.

```

Viết chương trình ngắt trong Windows tương đối khó, đòi hỏi trình độ lập trình cao, các bạn có thể đọc thêm trên mạng ở địa chỉ www.jungo.com

Bài tập gợi ý

1/Download chương trình windriver và nghiên cứu cách viết driver và ngắt cho card ISA, PCI

2/ Viết chương trình Visual Basic đo và điều khiển nhiệt độ dùng card ISA có ADC0908 và 8255

3/ Lập lại câu 2 dùng Visual C và Delphi