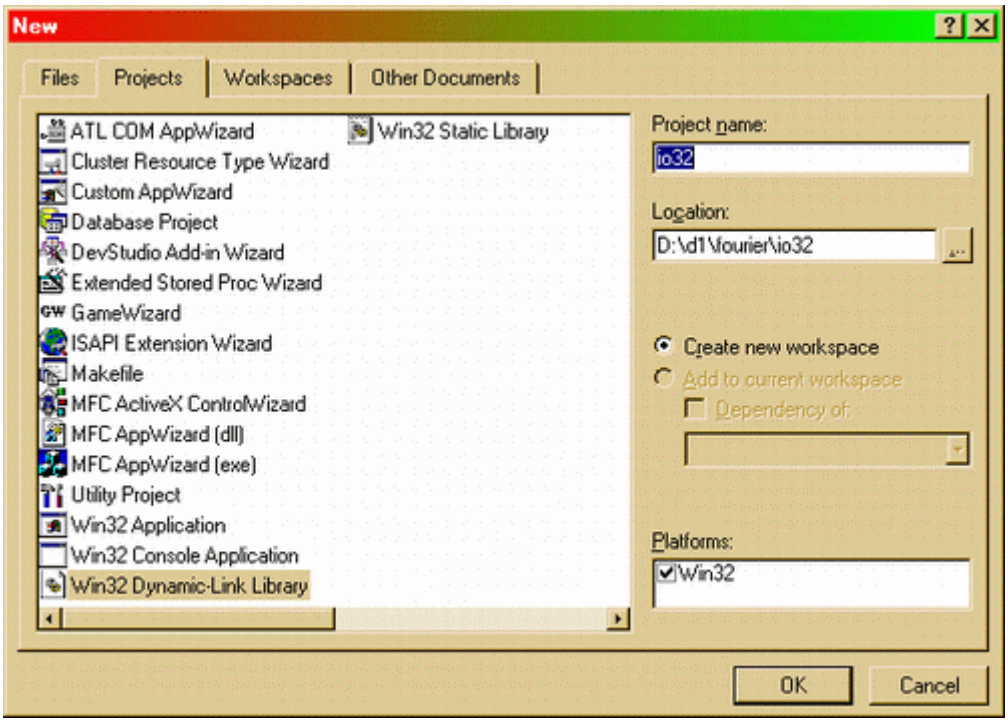
استفاده از پورت ها در ویژوال بیسیک

متاسفانه ویژوال بیسیک ابزاری را برای کار با پورت (درگاه)های کامپیوتر ارائه – نداده است. اما با استفاده از سایر کامپایلرهایی که روی سی دی ویژوال استودیو یافت می شود می توان ابزاری را برای این منظور ایجاد نمود و سپس از آن در **VB** استفاده کرد. ما در اینجا از **VC++ 6.0** برای ساختن **DLL** ایی جهت ارتباط برقرار کردن با پورت ها استفاده خواهیم نمود.

مراحل انجام این کار :

۱– **VC++ 6.0** را از روی سی دی ویژوال استودیو نصب کنید. برای انجام این کار به حدود **400 MB** فضای خالی روی سخت دیسک نیاز خواهد بود !

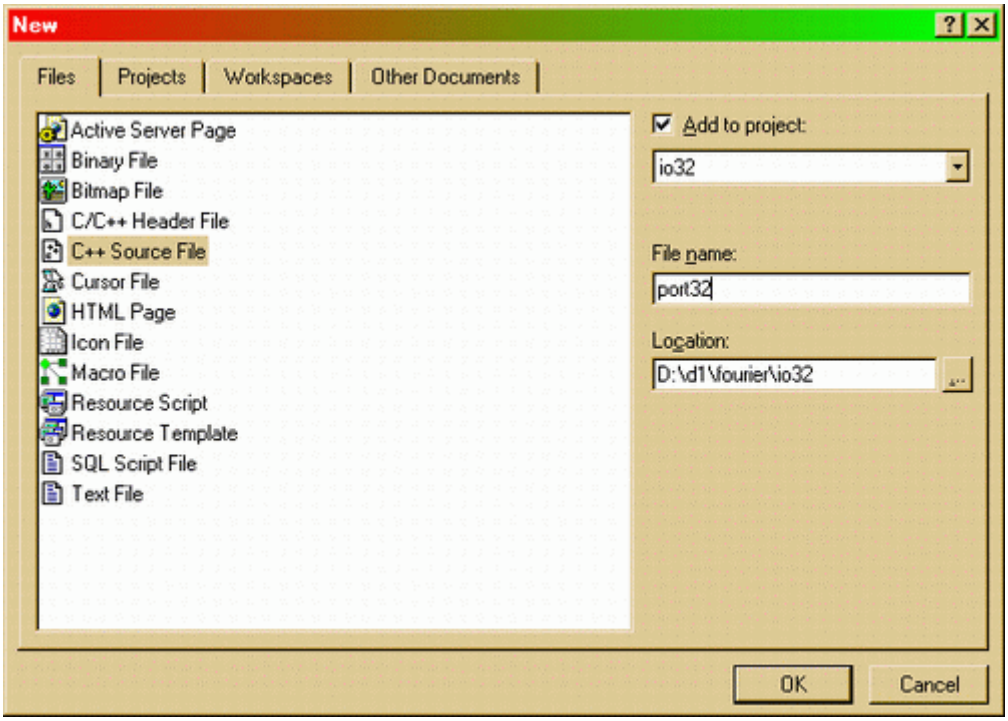
۲– پس از اجرای **VC++ 6.0** از منوی فایل گزینه **New** را انتخاب کنید. در برگه پروژه ها آیتم **Win32 Dynamic-Link Library** را انتخاب کنید و در قسمت نام پروژه ، نامی دلخواه مانند **IO32** را بنویسید و روی دکمه **Ok** کلیک کنید .



۳– در صفحه ای که سپس ظاهر می شود فقط روی دکمه **Finish** کلیک کنید .

۴– سپس از منوی فایل گزینه **New** رادوباره انتخاب کنید و از برگه فایلها **C++ Source File**

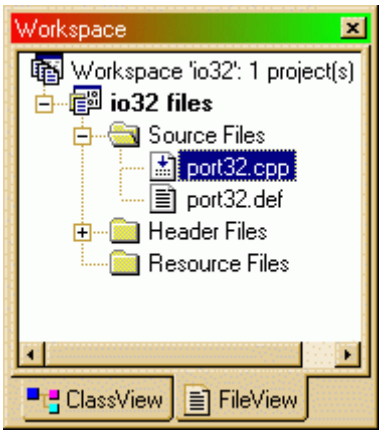
را برگزیده و در قسمت نام فایل ، **Port32** را بنویسید. سپس روی دکمه **Ok** کلیک کنید .



۵- مرحله قبل را دوباره تکرار کنید و این بار **C/C++ Header File** را انتخاب و نام آنرا به **Port32** تغییر دهید. سپس روی دکمه **Ok** کلیک کنید .

۶- مرحله ۴ را دوباره تکرار کنید و این بار **Text File** را انتخاب و نام آنرا به **Port32.def** تغییر تغییر دهید. سپس روی دکمه **Ok** کلیک کنید .

۷- در برگه **Workspace** که در سمت چپ صفحه قرار دارد روی **Port32.cpp** دوبار کلیک کنید و سپس در صفحه خالی که ظاهر می شود کد زیر را بنویسید.



```
            // ---------------------------------------- //
            //                Port32 v2.2               //
            //    I/O Port Access Under Windows 95/98   //
            // ---------------------------------------- //

#include <windows.h>
#include "Port32.h"

// These are our ring 0 functions responsible for tinkering
// with the hardware ports.
// They have a similar privilege to a Windows VxD and are
// therefore free to access
// protected system resources (such as the page tables) and
// even place calls to
// exported VxD services.

__declspec(naked) void Ring0GetPortByte()
{
  _asm In AL, DX
  _asm Mov [EBX], AL
  _asm Retf
}

__declspec(naked) void Ring0SetPortByte()
{
  _asm Mov AL, [EBX]
  _asm Out DX, AL
  _asm Retf
}

// This function makes it possible to call ring 0 code from
//a ring 3 application.

bool CallRing0(PVOID pvRing0FuncAddr, WORD wPortAddr,
               PBYTE pbPortVal)
{

  GDT_DESCRIPTOR *pGDTDescriptor;
  GDTR gdtr;

  _asm Sgdt [gdtr]

  // Skip the null descriptor

  pGDTDescriptor = (GDT_DESCRIPTOR *)(gdtr.dwGDTBase + 8);

  // Search for a free GDT descriptor

  for (WORD wGDTIndex = 1; wGDTIndex < (gdtr.wGDTLimit / 8);
         wGDTIndex++)
  {
    if (pGDTDescriptor->Type == 0     &&
        pGDTDescriptor->System == 0   &&
        pGDTDescriptor->DPL == 0      &&
        pGDTDescriptor->Present == 0)
    {
      // Found one !
      // Now we need to transform this descriptor into a callgate.
      // Note that we're using selector 0x28 since it corresponds
```

```
      // to a ring 0 segment which spans the entire linear address
      // space of the processor (0-4GB).

      CALLGATE_DESCRIPTOR *pCallgate;

      pCallgate = (CALLGATE_DESCRIPTOR *) pGDTDescriptor;
      pCallgate->Offset_0_15 = LOWORD(pvRing0FuncAddr);
      pCallgate->Selector = 0x28;
      pCallgate->ParamCount = 0;
      pCallgate->Unused = 0;
      pCallgate->Type = 0xc;
      pCallgate->System = 0;
      pCallgate->DPL = 3;
      pCallgate->Present = 1;
      pCallgate->Offset_16_31 = HIWORD(pvRing0FuncAddr);

      // Prepare the far call parameters

      WORD CallgateAddr[3];

      CallgateAddr[0] = 0x0;
      CallgateAddr[1] = 0x0;
      CallgateAddr[2] = (wGDTIndex << 3) | 3;

      // Please fasten your seat belts!
      // We're about to make a hyperspace jump into RING 0.

      _asm Mov DX, [wPortAddr]
      _asm Mov EBX, [pbPortVal]
      _asm Call FWORD PTR [CallgateAddr]

      // We have made it !
      // Now free the GDT descriptor

      memset(pGDTDescriptor, 0, 8);

      // Our journey was successful. Seeya.

      return true;
    }

    // Advance to the next GDT descriptor

    pGDTDescriptor++;
  }

  // Whoops, the GDT is full

  return false;
}


short _stdcall GetPortByte(WORD wPortAddr)
{
  BYTE bPortVal;
  bool Result;

  Result = CallRing0((PVOID)Ring0GetPortByte, wPortAddr, &bPortVal);

  if (Result == false)
  {
```

```
    return -1;
  }
  else
  {
    return bPortVal;
  }
}


bool _stdcall SetPortByte(WORD wPortAddr, BYTE bPortVal)
{
  return CallRing0((PVOID)Ring0SetPortByte, wPortAddr, &bPortVal);
}

          // ---------------------------------------- //
```

۸- سپس روی **Port32.def** دوبار کلیک کنید ودر صفحه خالی که ظاهر می شود کد زیر را بنویسید.

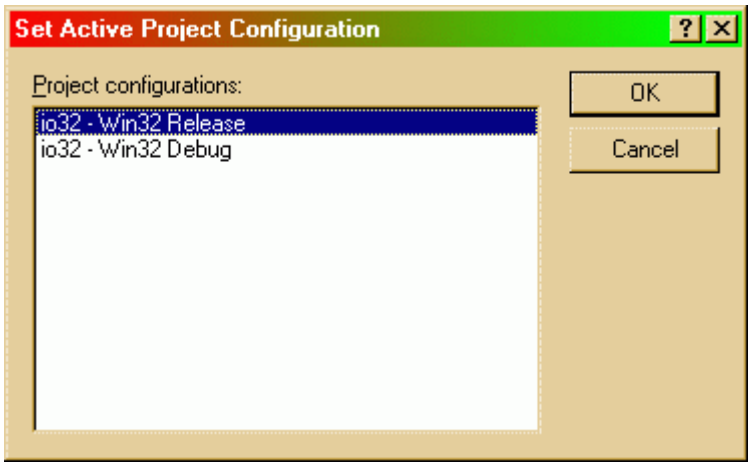
```
LIBRARY IO32

EXPORTS

GetPortByte
SetPortByte
```

۹- و در آخر روی **Port32.h** دوبار کلیک کنید ودر صفحه خالی که ظاهر می شود کد زیر را بنویسید.

```
#ifndef PORT32_H
#define PORT32_H

#ifdef PORT32_DLL
#define PORT32API _declspec(dllexport)
#else
#define PORT32API _declspec(dllimport)
#endif

#pragma pack(1)

struct GDT_DESCRIPTOR
{
  WORD Limit_0_15;
  WORD Base_0_15;
  BYTE Base_16_23;
  BYTE Type         : 4;
  BYTE System       : 1;
  BYTE DPL          : 2;
  BYTE Present      : 1;
  BYTE Limit_16_19  : 4;
  BYTE Available    : 1;
  BYTE Reserved     : 1;
  BYTE D_B          : 1;
  BYTE Granularity  : 1;
```

```
  BYTE Base_24_31;
};

struct CALLGATE_DESCRIPTOR
{
  WORD Offset_0_15;
  WORD Selector;
  WORD ParamCount   : 5;
  WORD Unused       : 3;
  WORD Type         : 4;
  WORD System       : 1;
  WORD DPL          : 2;
  WORD Present      : 1;
  WORD Offset_16_31;
};

struct GDTR
{
  WORD wGDTLimit;
  DWORD dwGDTBase;
};

#pragma pack()


extern "C"
{
  PORT32API short _stdcall GetPortByte(WORD wPortAddr);
  PORT32API bool _stdcall SetPortByte(WORD wPortAddr, BYTE bPortVal);
}

#endif

          // ---------------------------------------- //
```

۱۰- از منوی **Build** گزینه **Set Active Configuration** را انتخاب نموده و در صفحه ای که ظاهر می شود **Win32 Release** را برگزینید. و سپس پروژه را ذخیره کنید.



۱۱- دکمه **F7** را فشار دهید. ویا از منوی **Build** گزینه **Build io32.dll** را انتخاب کنید.

اگر در تایپ کد دچار اشتباه نشده باشید با این کار **io32.dll** تولید خواهد شد. دقت داشته باشید که زبان **C/C++** برخلاف **VB** بین حروف کوچک وبزرگ فرق می گذارد.

۱۲- برای استفاده از این فایل کتابخانه ای در **VB** کافی است یک ماژول جدید باز کنید و تعاریف زیر را به آن اضافه کنید:

```
Option Explicit

Declare Function GetPortByte Lib "io32.dll" ( _
                        ByVal PortAddr As Integer) As Integer
```

اگر خطایی رخ دهد خروجی تابع فوق **1-** خواهد بود . '

```
Declare Function SetPortByte Lib "io32.dll" ( _
  ByVal PortAddr As Integer, ByVal PortVal As Byte) As Boolean
```

و اگر خطایی رخ دهد خروجی تابع فوق **FALSE** خواهد بود . '

مثالی از استفاده این dll :

ابتدا یک پروژه جدید در ویژوال بیسیک باز کنید و یک ماژول جدید به آن اضافه نمایید. سپس کد زیر را در آن بنویسید :

```
Option Explicit

Declare Function GetPortByte Lib "io32.dll" _
 (ByVal PortAddr As Integer) As Integer

Declare Function SetPortByte Lib "io32.dll" _
(ByVal PortAddr As Integer, ByVal portVal As Byte) _
 As Boolean


' PC Speaker sample
' This sample illustrates how to use IO32
' Library to control PC speaker.
Public Sub Speaker(bOn As Boolean)
    Dim portVal As Integer

    portVal = GetPortByte(&H61)
    If bOn Then
        portVal = portVal Or 3
        portVal = &H2F
    Else
        portVal = portVal And (Not 3)
    End If
    SetPortByte &H61, portVal
End Sub

' Speaker freq is set by setting divider in Intel
```

```
' 8253/8254 timer chip at port addresses $42 through $43
Public Sub SetFreq(ByVal hertz As Integer)
    Dim s As String

    s = Space(255)

    If hertz Then
        Dim divisor As Long
        divisor = 1193180 / hertz

        SetPortByte &H43, &HB6

        SetPortByte &H42, divisor Mod 256
        SetPortByte &H42, divisor \ 256

        Speaker True
    Else
        Speaker False
    End If
End Sub

Public Sub PlayFrequency(ByVal Frequency As Long)
 Dim clockTicks
 Dim loopCount
 clockTicks = 1193280 \ Frequency
 'prepare for data
 SetPortByte 67, 182
 'send data
 SetPortByte 66, clockTicks And &HFF
 SetPortByte 66, clockTicks \ 256
 'turn speaker on
 SetPortByte 97, GetPortByte(97) Or &H3
 For loopCount = 1 To 200
   DoEvents
 Next
 'turn speaker off
 SetPortByte 97, GetPortByte(97) And &HFC
End Sub
```

بله همانطور که حدس زده اید می خواهیم فرکانس خاصی را از بلندگوی کامپیوتر پخش کنیم. برای این منظور سه دکمه و یک جعبه متنی به فرم برنامه اضافه کنید.

نام دکمه ها را مطابق شکل به cmdSound, cmdNoSound, cmdPlayFrequency تغییر دهیدو نام جعبه متنی را به txtF .

روی هرکدام از دکمه ها دو بار کلیک کنید و در روال رخداد هرکدام کدهای زیر را

بنویسید.

```
Private Sub cmdNoSound_Click()
 Speaker False
End Sub

Private Sub cmdPlayFrequency_Click()
 PlayFrequency txtF.Text
End Sub

Private Sub cmdSound_Click()
 SetFreq txtF.Text
End Sub
```

البته بدیهی است که فایل **dll** فوق باید در دایرکتوری جاری فایل **اجرایی** برنامه شما قرار داشته باشد و یا در دایرکتوری **Windows*system** .

راه دیگری نیز برای دستیابی مستقیم به پورت ها وجود دارد. لطفا به مثال زیر توجه کنید :

```
'Input
  dim s as string
  open "LPT1" for input as #1
  s = input(NumberOfBytesToRead, 1)
  close #1

'Output
   open "LPT1" for output as #1
   print #1, "TheStuffToSend"
   close #1
```

و برای دستیابی به پورت موازی می توان از کنترل **Comm** Microsoft استفاده کرد.

```
Project->Components->Microsoft Comm Control XX.0
```

منابع و مآخذ:

http://personal.vsnl.com/ar/
http://members.tripod.com/~zealsoft/vbio/
http://www.netease.com/~zealsoft/vbio/
http://WWW.INTERNALS.COM/