Classification	WinPAC SDK FAQ					No.	6-010-00
Author	Sean	Version	1.0.1	Date	2012/10/18	Page	1/16

How to update to the PACSDK library from the WinPacSDK library

Applies to:

Platform	*OS Version*	*XPAC Utility Version*
WP-8x3x	*All versions (WinCE5)*	*All versions*
WP-8x4x	*All versions (WinCE5)*	*All versions*
VP-25W1	*All versions (WinCE5)*	*All versions*
VP-23W1	*All versions (WinCE5)*	*All versions*
WP-5000	*All versions (WinCE5)*	*All versions*

Before updating to the PACSDK library from the WinPacSDK library, ensure that the latest version of the WinPAC platform SDK installed on Windows PC first. Please note that the release date of the SDK installation package that includes the PACSDK library should be later than or equal to 2012/10/15, e.g., PAC270_SDK_20121015.msi

The latest version of the platform SDK installation package can be found on the shippment CD and from the ICPDAS FTP site, as shown below

CD:

CD:\Napdos\wp-8x4x_ce50\SDK\

FTP:

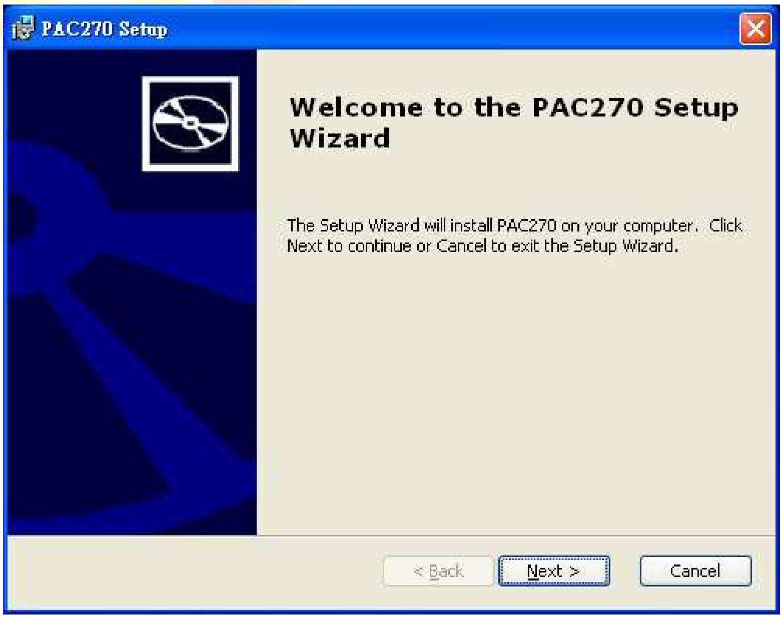
http://ftp.icpdas.com/pub/cd/winpac/napdos/wp-8x4x_ce50/sdk/

File name: pac270_sdk_**yyyymmdd**.msi, where **yyyymmdd** is the platform SDK release date.

Follow the procedure described below to install SDK:

Step 1: Execute the “PAC270_SDK_YYYYMMDD.msi” file

Step 2: Follow the prompts until the PAC270_SDK_YYYYMMDD.msi installation process is complete.



Welcome to the PAC270 Setup Wizard

The Setup Wizard will install PAC270 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

We provide two methods that allow users their project files to update the PACSDK library from WinPacSDK. The first is to provide detailed steps for users to update their programs to use the PACSDK library by themselves, and the other is to provide an update tool that can be used to update the PACSDK automatically.

Updating the SDK manually

eVC programs

Follow the steps below to update your program so that it uses the PACSDK library.

Step 1: Change the WinPacSDK.h #include file reference to PACSDK.h

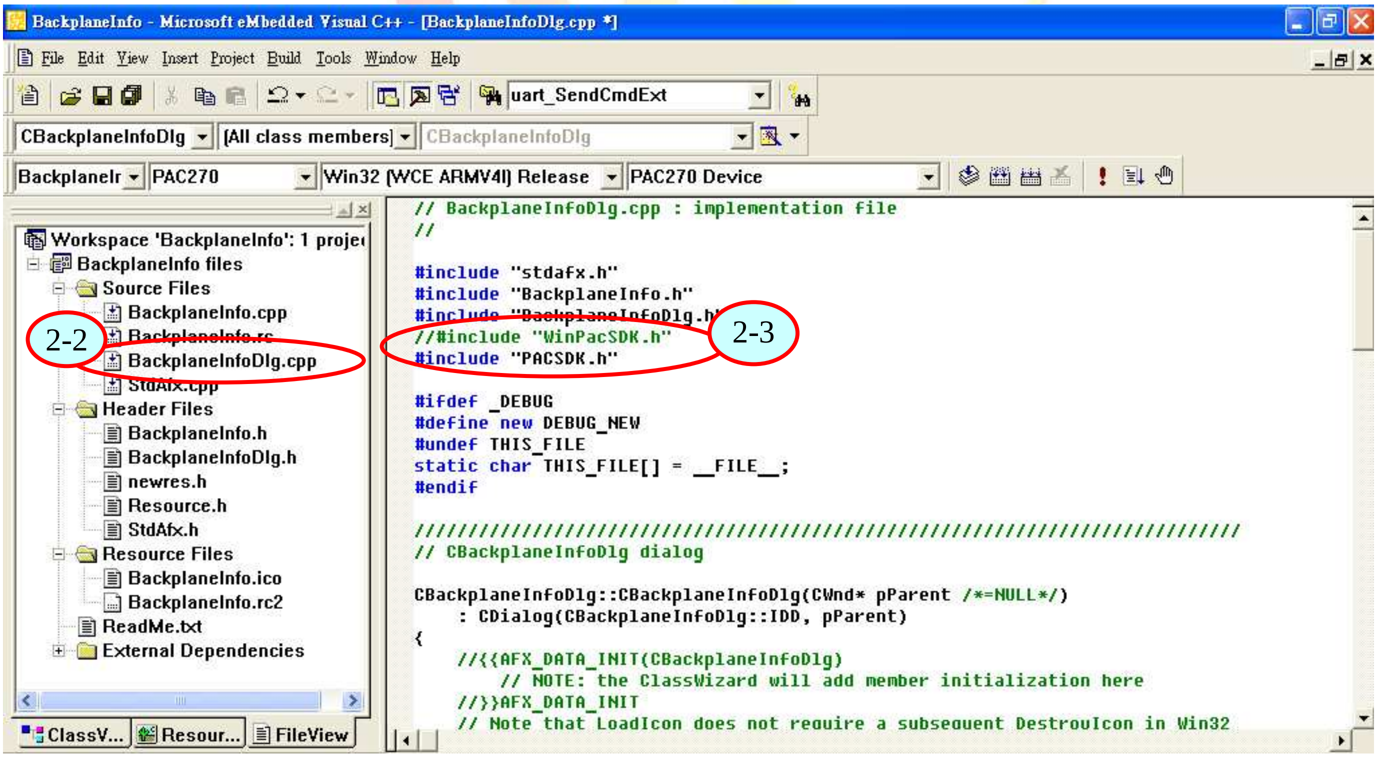
1-1 Open your **Microsoft Embedded Visual C++** project (*.vcw) file (that is used for WinPAC series modules.)

1-2 Left-click the *.cpp file to open it in the FileView window.

1-3 Modify the following code in the Editor Window (Refer to the figure below)

```
//#include "WinPacSDK.h"
#include "PACSDK.h"
```

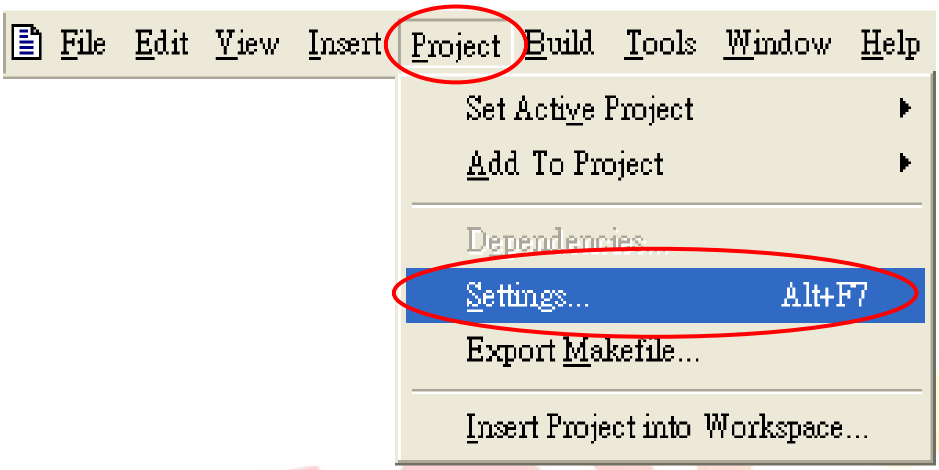
(Replace eahc line that contains #include “WinPacSDK.h” with #include “PACSDK.h”)



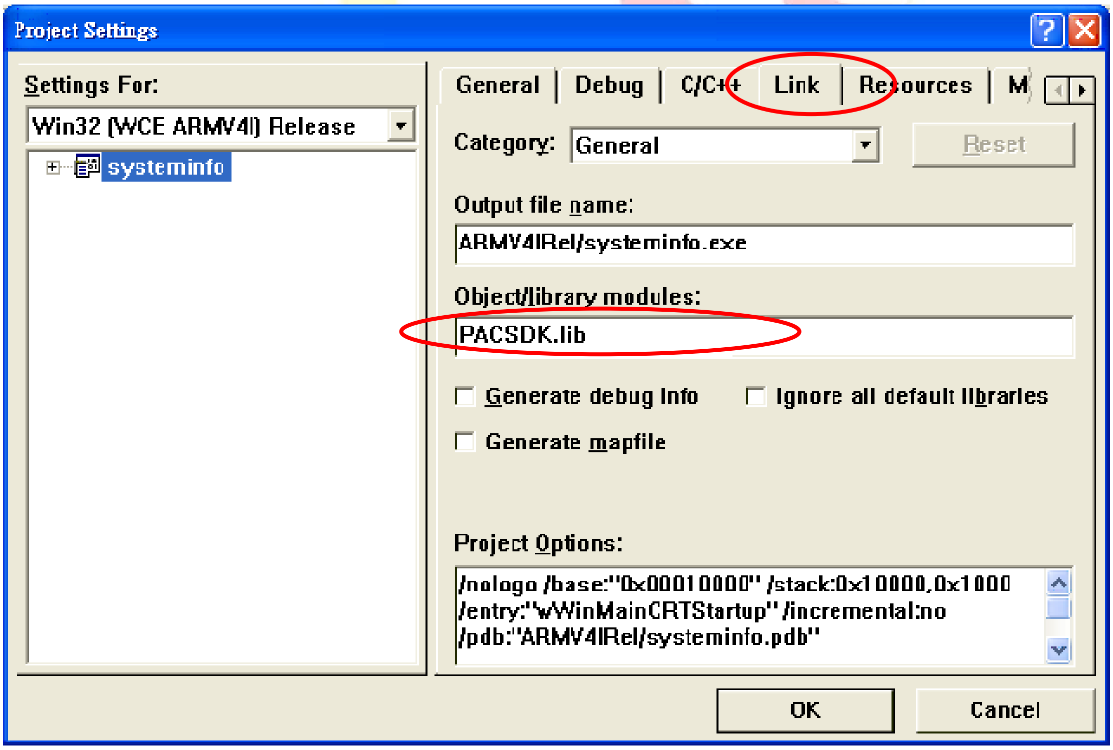
```
// BackplaneInfoDlg.cpp : implementation file
//

#include "stdafx.h"
#include "BackplaneInfo.h"
#include "BackplaneInfoDlg.h"
//#include "WinPacSDK.h"
#include "PACSDK.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CBackplaneInfoDlg dialog

CBackplaneInfoDlg::CBackplaneInfoDlg(CWnd* pParent /*=NULL*/)
    : CDialog(CBackplaneInfoDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CBackplaneInfoDlg)
        // NOTE: the ClassWizard will add member initialization here
    //}}AFX_DATA_INIT
    // Note that LoadIcon does not require a subsequent DestroyIcon in Win32
```

Step 2: Change the WinPacSDK.lib reference to the PACSDK.lib

2-1 Click the “Settings…” option from the “Project” menu, or press Alt+F7.



2-2 In the “Project Settings” dialog box, Click the Link tab and type PACSDK.lib in the Object/library modules text box then click the “OK” button.



Category: General

Output file name: ARMV4IRel/systeminfo.exe

Object/library modules: PACSDK.lib

□ Generate debug info □ Ignore all default libraries

□ Generate mapfile

Project Options:

```
/nologo /base:"0x00010000" /stack:0x10000,0x1000
/entry:"wWinMainCRTStartup" /incremental:no
/pdb:"ARMV4IRel/systeminfo.pdb"
```

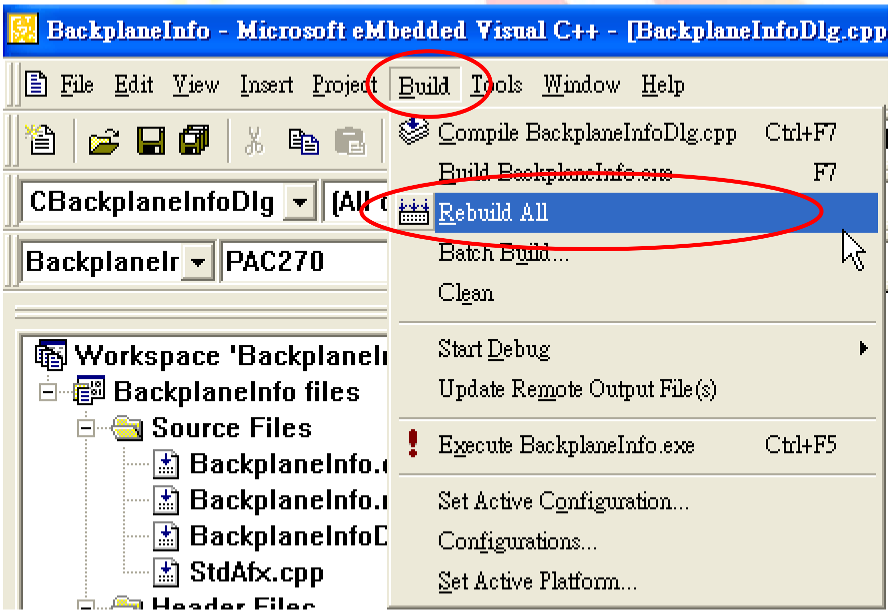
Step 3: Several of the error code messages have been clanged in the PACSDK library, so use the example code below to check whether you need to modify your program and amend it where necessary.

Error code modification

```
        //if(pac_GetLastError() == PAC_ERR_EEP_ACCESS_RESTRICTION)
    if(pac_GetLastError() == PAC_ERR_EEP_INVALID_ADDRESS)
    {
        //...
    }

    //if(pac_GetLastError() == PAC_ERR_SRAM_INVALID_TYPE)
    if(pac_GetLastError() == PAC_ERR_MEMORY_INVALID_TYPE)
    {
        //...
    }
```

The error codes, PAC_ERR_EEP_ACCESS_RESTRICTION and PAC_ERR_SRAM_INVALID_TYPE, as defined in WinPacSDK.h have been modified to PAC_ERR_EEP_INVALID_ADDRESS and PAC_ERR_MEMORY_INVALID_TYPE, as defined in PACSDK.h.

Step 4: From the "Build" menu, select the "Rebuild All" option to rebuild your program.



BackplaneInfo - Microsoft eMbedded Visual C++ - [BackplaneInfoDlg.cpp
File Edit View Insert Project Build Tools Window Help
CBackplaneInfoDlg
[All
BackplaneIr
PAC270
Workspace 'BackplaneI
BackplaneInfo files
Source Files
BackplaneInfo.
BackplaneInfo.
BackplaneInfoD
StdAfx.cpp
Header Files
Compile BackplaneInfoDlg.cpp Ctrl+F7
Build BackplaneInfo.exe F7
Rebuild All
Batch Build...
Clean
Start Debug
Update Remote Output File(s)
Execute BackplaneInfo.exe Ctrl+F5
Set Active Configuration...
Configurations...
Set Active Platform...

VC programs

Follow the steps below to update your program so that it uses the PACSDK library.

Step 1: Change the WinPacSDK.h #include file reference to PACSDK.h

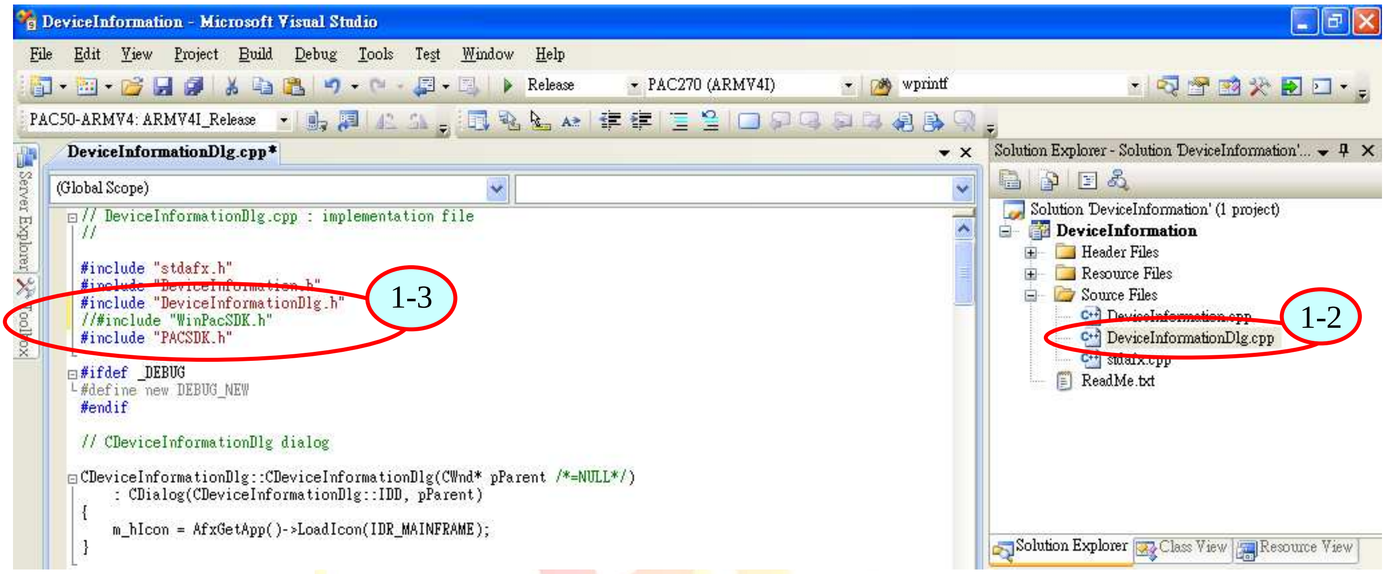
1-1 Open your **Visual Studio 2005** or **2008** project (*.sln) file (that is used for WinPAC series modules.)

1-2 Left-click the *.cpp file to open it in the Solution Explorer windows.

1-3 Modify the following code in the Editor Window (Refer to the figure below)

```
//#include "WinPacSDK.h"
#include " PACSDK.h"
```

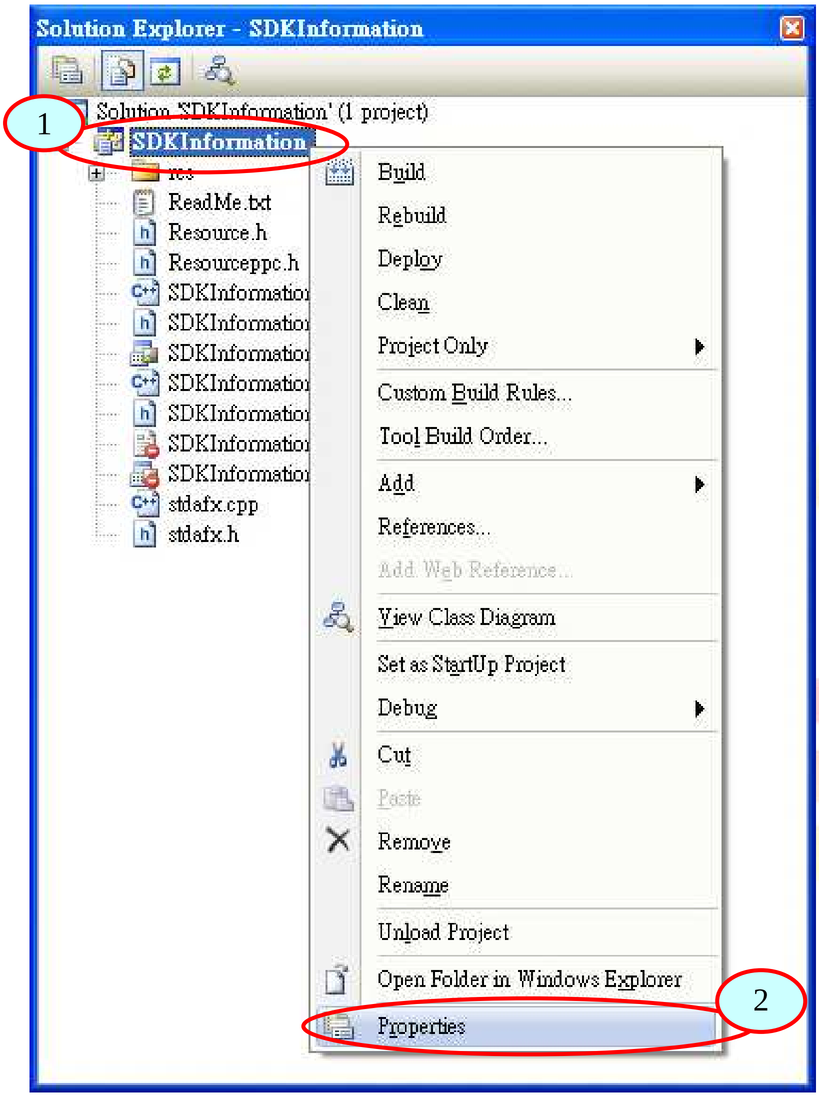
(Replace each line that contains #include “WinPacSDK.h” with #include “PACSDK.h”)



```
// DeviceInformationDlg.cpp : implementation file
//

#include "stdafx.h"
#include "DeviceInformation.h"
#include "DeviceInformationDlg.h"
//#include "WinPacSDK.h"
#include "PACSDK.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

// CDeviceInformationDlg dialog

CDeviceInformationDlg::CDeviceInformationDlg(CWnd* pParent /*=NULL*/)
    : CDialog(CDeviceInformationDlg::IDD, pParent)
{
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}
```

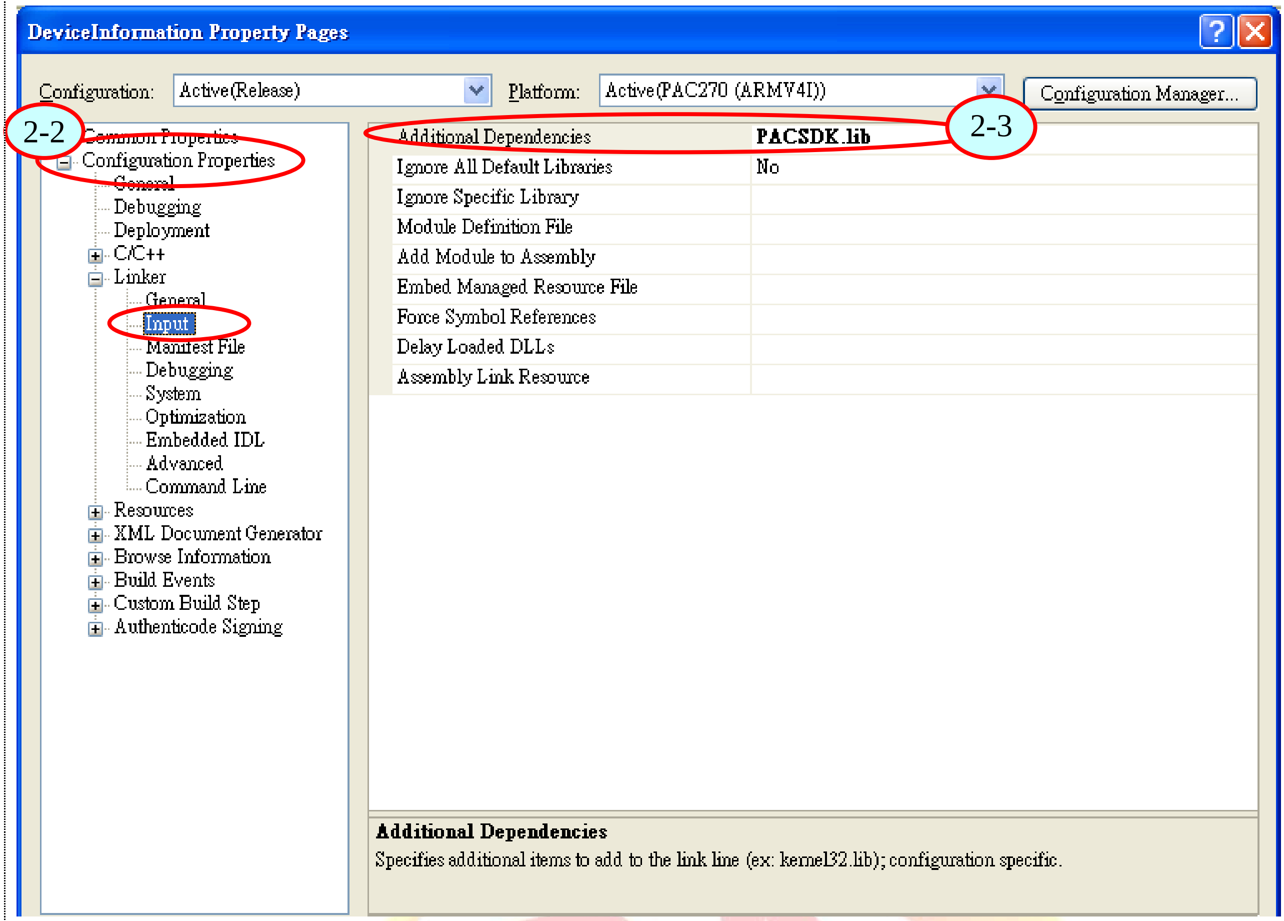
Step 2: Change the WinPacSDK.lib reference to the PACSDK.lib

2-1 Right-click the project name in Solution Explorer window, and then select the "Properties" option.



2-2 In left pane of the Properties page, expand the Configuration Properties menu group, and then click the Linker option.

2-3 In the right pane, type "PACSDK.lib" in the Additional Dependencies item



Additional Dependencies	PACSDK.lib
Ignore All Default Libraries	No
Ignore Specific Library	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded DLLs	
Assembly Link Resource	

Additional Dependencies
Specifies additional items to add to the link line (ex: kernel32.lib); configuration specific.

Step 3: Several of the error code messages have been clanged in the PACSDK library, so use the example code below to check whether you need to modify your program and amend it where necessary.

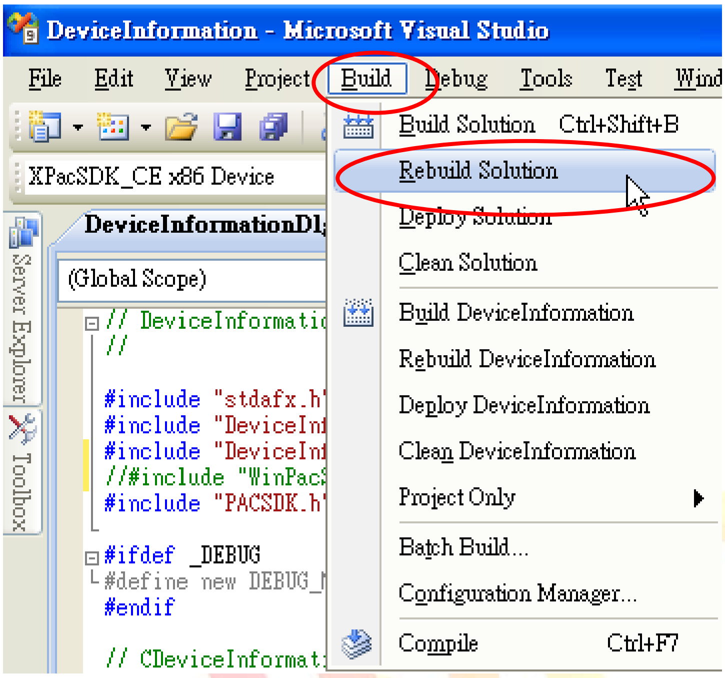
.

Error code modification

```
    //if(pac_GetLastError() == PAC_ERR_EEP_ACCESS_RESTRICTION)
if(pac_GetLastError() == PAC_ERR_EEP_INVALID_ADDRESS)
{
      //...
}

//if(pac_GetLastError() == PAC_ERR_SRAM_INVALID_TYPE)
if(pac_GetLastError() == PAC_ERR_MEMORY_INVALID_TYPE)
{
      //...
}
```

The error codes, PAC_ERR_EEP_ACCESS_RESTRICTION and PAC_ERR_SRAM_INVALID_TYPE, as defined in WinPacSDK.h have been modified as PAC_ERR_EEP_INVALID_ADDRESS and PAC_ERR_MEMORY_INVALID_TYPE, as defined in PACSDK.h.

Step 4: From the “Build” menu, select the “Rebuild Solution” option to rebuild your program.



```
// DeviceInformati
//

#include "stdafx.h"
#include "DeviceInf
#include "DeviceInf
//#include "WinPacS
#include "PACSDK.h"

#ifdef _DEBUG
#define new DEBUG_N
#endif

// CDeviceInformati
```

.NET Compact Framework programs (C#, VB.net)

Follow the steps below to update your program so that it uses the PACNET library.

Step 1: Change the WinPacNet.dll to the PACSDK.dll

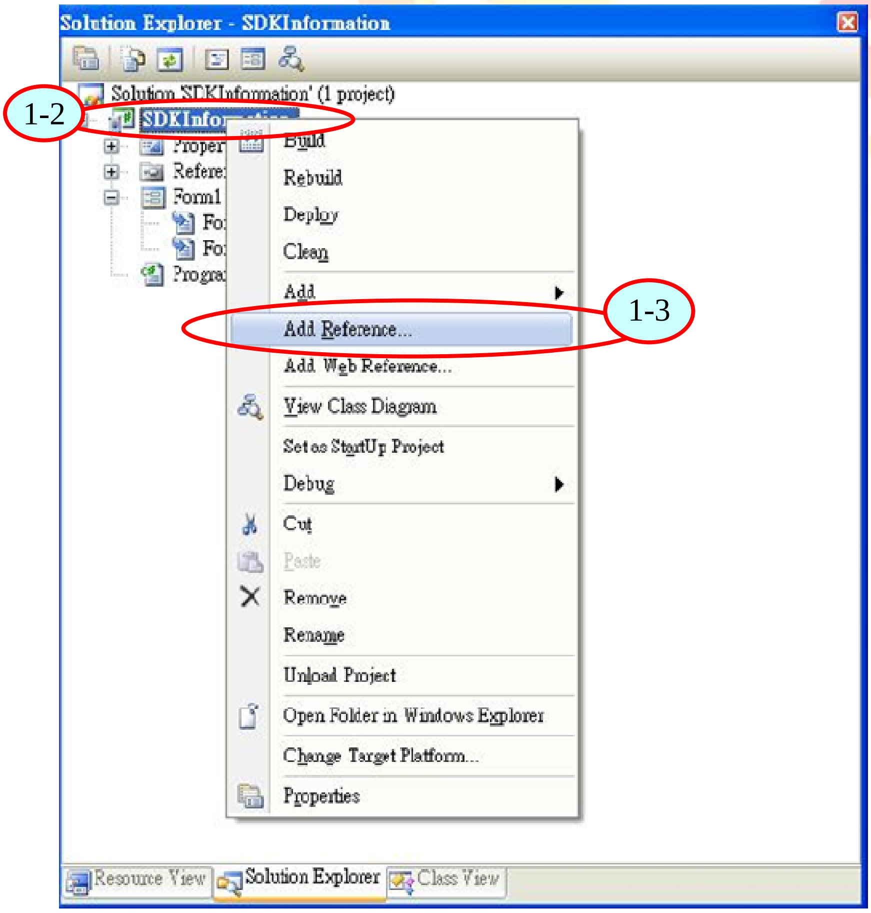
1-1 Locate the PACNET.dll and copy it to the .NET CF project folder

The PACNET.dll can be obtained using the link below that has been provided on the CD or by downloading the latest version from ICP DAS web site.
CD:\WinPAC\napdos\wp-8x4x_ce50\sdk\WinPacNet

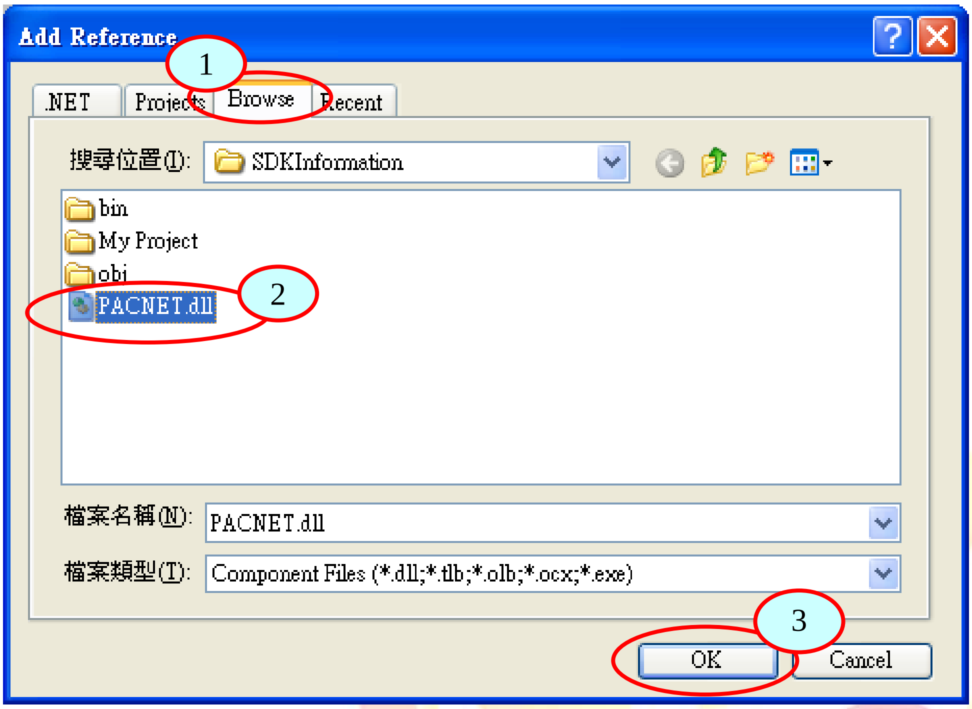
http://ftp.icpdas.com/pub/cd/winpac/napdos/wp-8x4x_ce50/sdk/winpacnet/

1-2 Open your **Visual Studio 2005** or **2008** project (*.sln) file (that is used for WinPAC series modules.)

1-3 Right-click the References node in the Solution Explorer, and then click Add Reference…



1-4 In the “Add Reference” windows, click the “Browse” tab and select PACNET.dll file, and then click the “OK” button.



搜尋位置(I): SDKInformation

檔案名稱(N): PACNET.dll

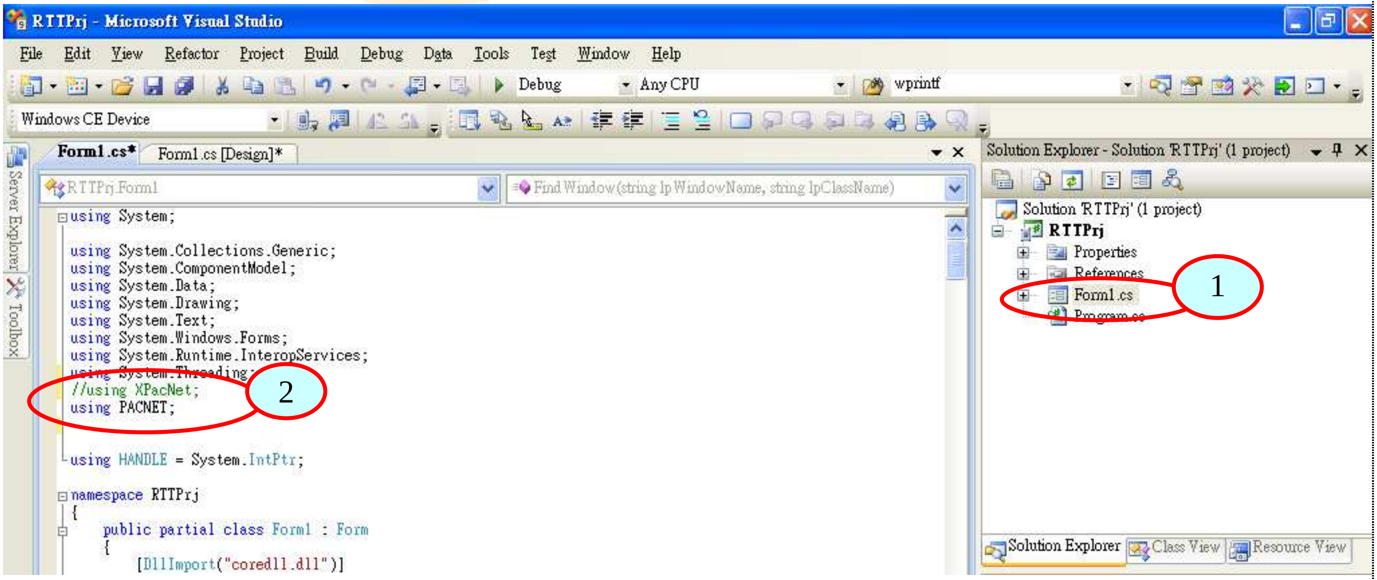
檔案類型(T): Component Files (*.dll;*.tlb;*.olb;*.ocx;*.exe)

Step 2: Use the example code below to check whether you need to modify your program and amend it where necessary.

2-1 Modify the code, “using WinPacNet” to “using PACNET”.

For a C# program

```
// using WinPacNet;
using PACNET;
```



```
using System;

using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Runtime.InteropServices;
using System.Threading;
//using XPacNet;
using PACNET;

using HANDLE = System.IntPtr;

namespace RTTPrj
{
    public partial class Form1 : Form
    {
        [DllImport("coredll.dll")]
```

RTTPrj - Microsoft Visual Studio
File Edit View Refactor Project Build Debug Data Tools Test Window Help
Debug
Any CPU
wprintf
Windows CE Device
Form1.cs*
Form1.cs [Design]*
RTTPrj.Form1
Find Window(string lpWindowName, string lpClassName)
Server Explorer
Toolbox
Solution Explorer - Solution 'RTTPrj' (1 project)
Solution 'RTTPrj' (1 project)
RTTPrj
Properties
References
Form1.cs
Program.cs
1
2
Solution Explorer
Class View
Resource View

For a VB program

```
//Imports WinPacNet
 Imports PACNET
```

2-2 DllImport Modification

In order to call the API function of WinPAC SDK in a C# or VB program, you need to use DllImport.

In a C# program

```
//[DllImport("WinPacSDK.dll")]
[DllImport("PACSDK.dll")]
```

In a C# program, use the code Dllimport(“PACSDK.dll”) to replace Dllimport(“WinPacSDK.dll”).

In a VB program

```
//<DllImport("WinPacSDK.dll")>
<DllImport("PACSDK.dll")>
```

In a VB program, use the code Dllimport(“PACSDK.dll”) to replace Dllimport(“WinPacSDK.dll”).

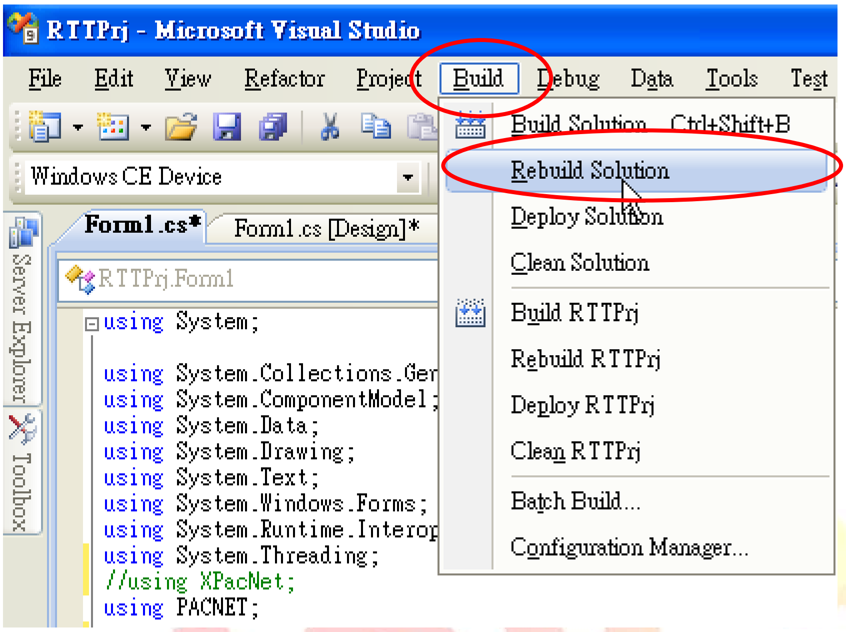
2-3 Class name modification

All API functions for the WinPacNet.dll are placed in a single WinPacNet.WinPAC.xxx class, but the API functions for the PACNET.dll are classified as PACNET.sys, PACNET.Memory, and PACNET.Interrupt, etc.

The classifications applied to the API functions for the PACNET.dll, as defined in the API user manual, are as follows.

Classification in the API Manual	Class Name in PACNET.dll
2.1 System Information API	Sys
2.1 Backplane API	Sys
2.1 Buzzer API	Sys.Buzzer
2.2 Interrupt API	Interrupt
2.3 Memory Access API	Memory
2.4 Watchdog API	Sys.WDT
2.5 Registry API	PAC_Reg
2.6 UART API	UART
2.7 PAC_IO API	PAC_IO
2.8. PWM API	PWM
2.9. Backplane Timer API	BPTimer
2.10. Error Handling API	ErrHandling
2.11 MISC API	MISC

Step 4: From the “Build” menu, select the “Rebuild Solution” option to rebuild your program.



```
using System;

using System.Collections.Ger
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Runtime.Interop
using System.Threading;
//using XPacNet;
using PACNET;
```

Updating the SDK using the update tool

EasyUpgradeSDK.exe
EasyUpgradeSDK is a program transformation tool that can be used to easily update a VC program that use the WinPacSDK library so that it uses the PACSDK library, and to update the C#/VB.net program that uses the WinPacNet library so that it uses PACNET library.
This program needs to be run on a Windows PC.

The latest version of EasyUpgradeSDK.exe can be downloaded from:
http://ftp.icpdas.com/pub/cd/winpac/napdos/wp-8x4x_ce50/pc_tools/UpgradeSDK

The EasyUpgradeSDK software can be used to convert the following file extension types
*.cpp
*.cs
*.vb
*.vcproj
*.csproj
*.vbproj
*.vcp

The benefits of using the transformation tool incluse:

- The EasyUpgradeSDK tool will automatically replace the keyword codes that exists on the content of the files for the file extension types that are listed above.
- The tool will automatically copy the PACSDK.h, PACSDK.lib or PACNET.dll files to the appropriate folder, which the user can select once the tool determinest that Winpacsdk.h/Xpacsdk.h, Winpacsdk.lib/Xpacsdk.lib or WinPacNet.dll/XPacNet.dll files exist in the folder.

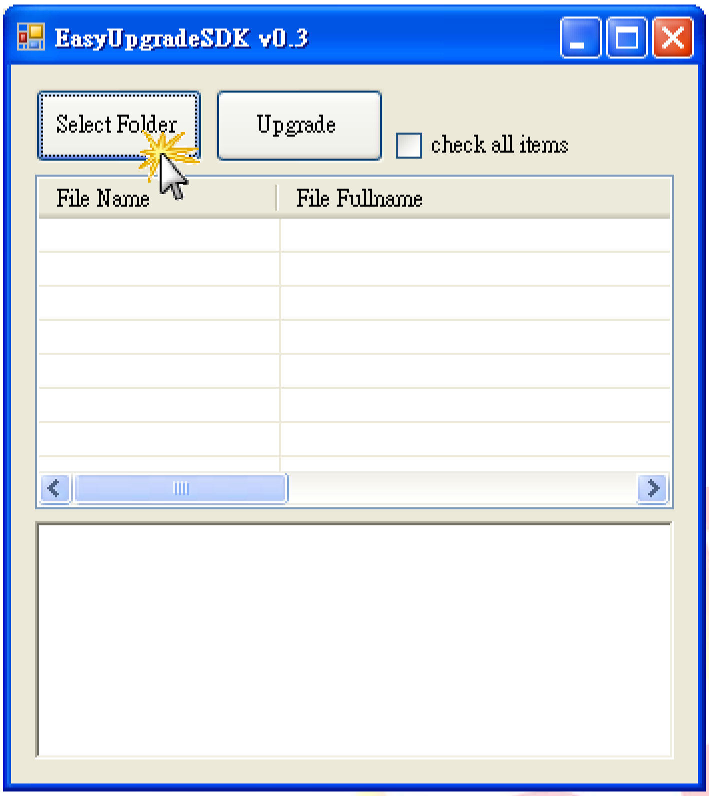
Follow the procedure below to update the SDK using the EasyUpgradeSDK tool.
Step 1: Confirm that the file structure is the same as that shown below, and then copy the files to a Windows PC.

```
EasyUpgradeSDK
 │    EasyUpgradeSDK.exe
 │    PACNET.dll
 │
 ├──release_arm
 │        PACSDK.H
 │        PACSDK.lib
 │
 └──release_x86
           PACSDK.H
           PACSDK.lib
```

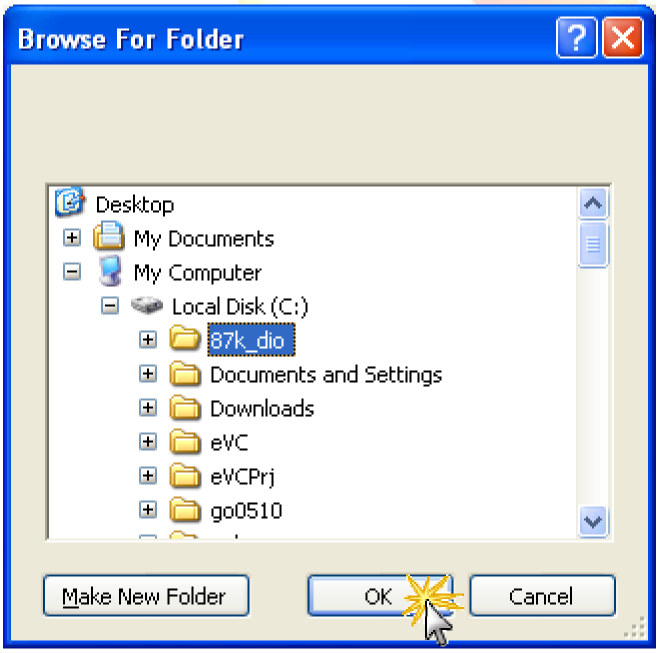
Step 2: Run the EasyUpgradeSDK.exe file.

Step 3: Select the folder where the project files are located on. (The project files for WinPAC or XPAC series modules must be implemented using C#, VB.net, VC or eVC)

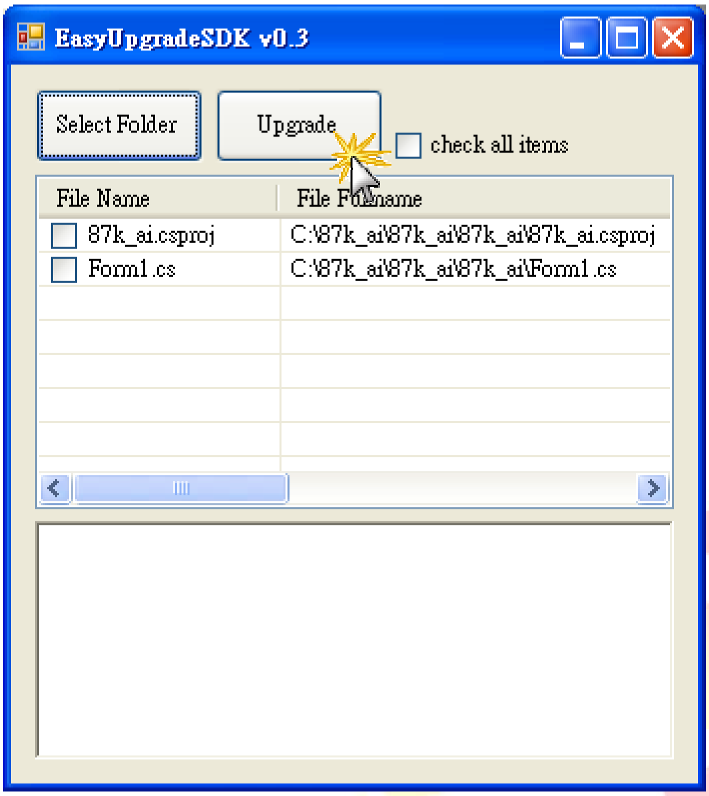
3-1: Click the “Select Folder” button



3-2: Select the appropriate folder, and then Click the “OK” button
(The C# demoe file, 87k_dio is used as the example here.)

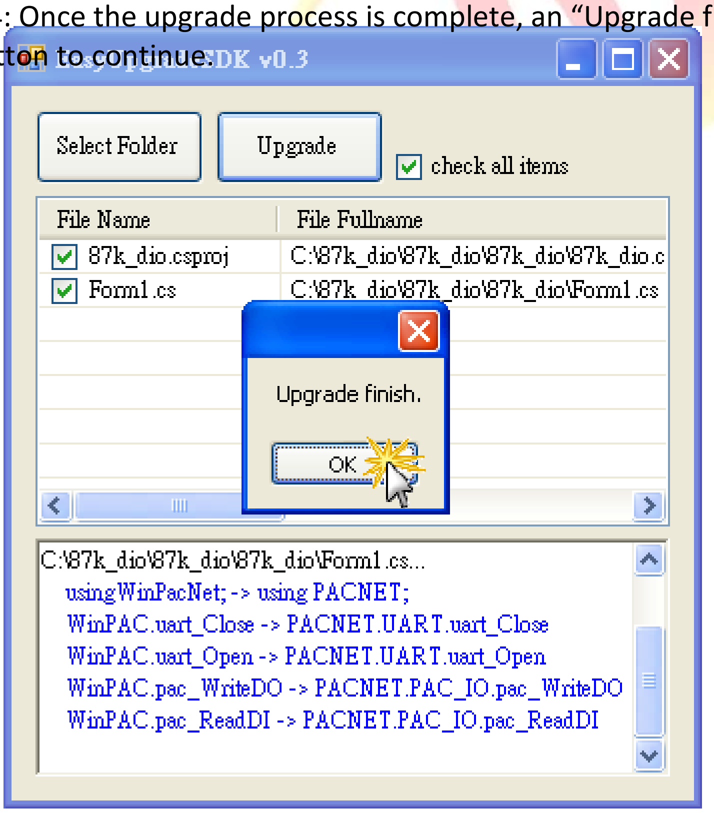


3-3: Click the “check all items” checkbox and then click the “Upgrade” button to begin the upgrade process.



File Name	File Fullname
□ 87k_ai.csproj	C:\87k_ai\87k_ai\87k_ai\87k_ai.csproj
□ Form1.cs	C:\87k_ai\87k_ai\87k_ai\Form1.cs

3-4: Once the upgrade process is complete, an “Upgrade finished” alert will be displayed. Click the “OK” button to continue.



File Name	File Fullname
☑ 87k_dio.csproj	C:\87k_dio\87k_dio\87k_dio\87k_dio.c
☑ Form1.cs	C:\87k_dio\87k_dio\87k_dio\Form1.cs

Upgrade finish.

```
C:\87k_dio\87k_dio\87k_dio\Form1.cs...
   usingWinPacNet; -> using PACNET;
   WinPAC.uart_Close -> PACNET.UART.uart_Close
   WinPAC.uart_Open -> PACNET.UART.uart_Open
   WinPAC.pac_WriteDO -> PACNET.PAC_IO.pac_WriteDO
   WinPAC.pac_ReadDI -> PACNET.PAC_IO.pac_ReadDI
```

Step 4: Aftersuccessfully upgrading the program, you will need to open the project and rebuild it before it can be used. (Follow the instructions given in previous sections for details of how to perform this for your specific project.