How to update to the PACSDK library from the XPacSDK (CE6) library

Applies to:

Platform	*OS Version*	*XPAC Utility Version*
XPAC series	*All versions (WinCE6)*	*All versions*

Before updating to the PACSDK library from the XPacSDK_CE library, ensure that the latest version of the XPAC platform SDK installed on Windows PC first. Please note that the version number of SDK installation package includes the PACSDK library should be later than or equal to 1.4.0, e.g., XPacSDK_CE_1.4.0_VS2008.msi or XPacSDK_CE_1.4.0_VS2005.msi

The latest version of the XPAC (CE6) platform SDK installation package can be found on the shippment CD and from the ICPDAS FTP site, as shown below.

CD:

CD: \XP-8000-CE6\SDK\PlatformSDK

CD: \XPAC-ATOM-CE6\SDK\PlatformSDK

FTP:

http://ftp.icpdas.com/pub/cd/xp-8000-ce6/sdk/platformsdk/

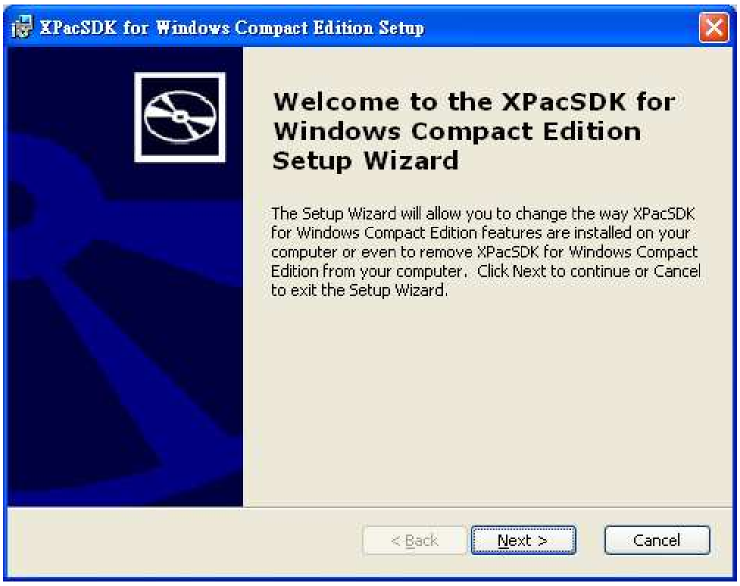
http://ftp.icpdas.com/pub/cd/xpac-atom-ce6/sdk/platformsdk/

File name: xpacsdk_ce_**n.n.n**_vs**xxxx**.msi, where **n.n.n** is the platform SDK version number, **xxxx**: 2005 indicates VS2005, 2008 indicates VS2008)

Follow the procedure described below to install SDK:

Step 1: Run the “XPacSDK_CE.msi

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



XPacSDK for Windows Compact Edition Setup

Welcome to the XPacSDK for Windows Compact Edition Setup Wizard

The Setup Wizard will allow you to change the way XPacSDK for Windows Compact Edition features are installed on your computer or even to remove XPacSDK for Windows Compact Edition from your computer. Click Next to continue or Cancel to exit the Setup Wizard.

We provide two methods that allow users to update their project files to use the PACSDK library from XPacSDK_CE. 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

VC programs

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

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

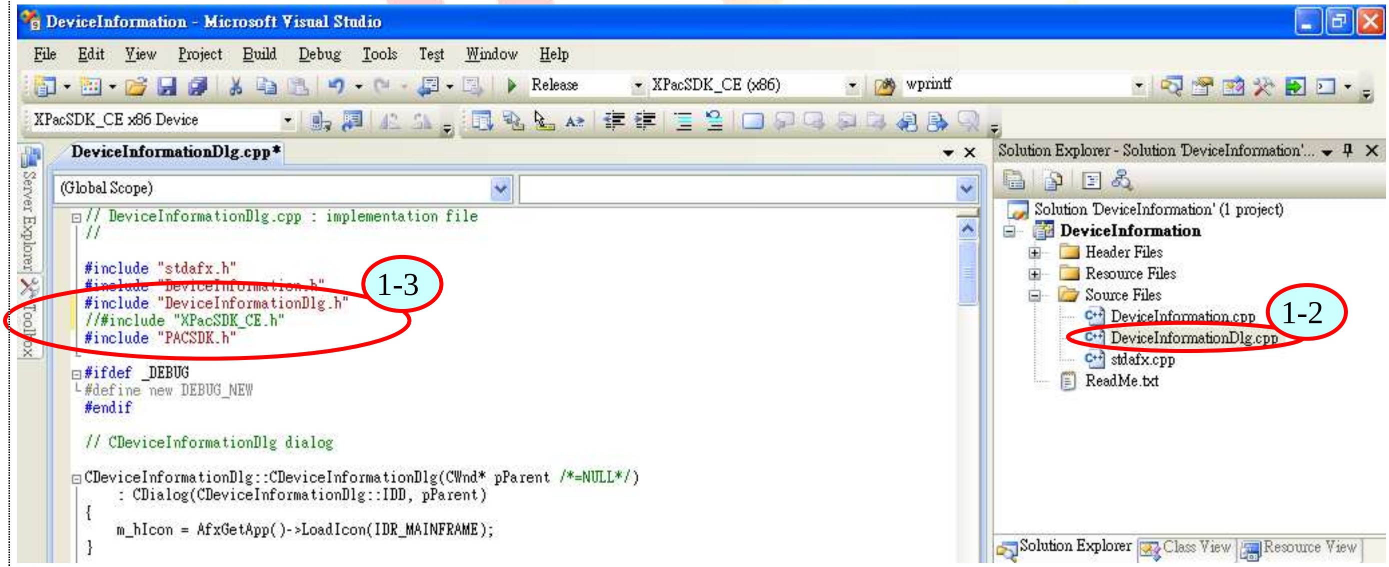
1-1 Open your **Visual Studio 2005** or **2008** project (*.sln) file (that is used for XPAC series modules.)

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

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

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

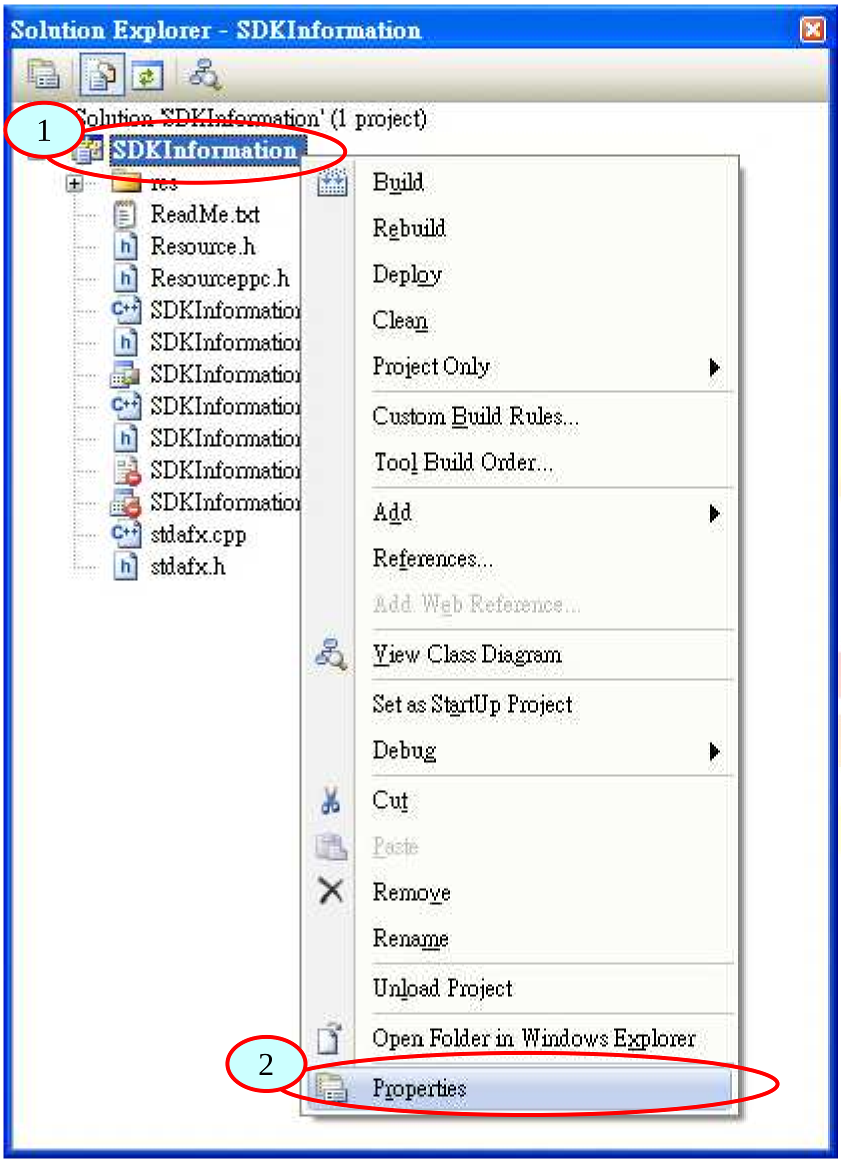
(Replace each line that contains #include “XPacSDK_CE.h” with #include “PACSDK.h”)



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

#include "stdafx.h"
#include "DeviceInformation.h"
#include "DeviceInformationDlg.h"
//#include "XPacSDK_CE.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);
}
```

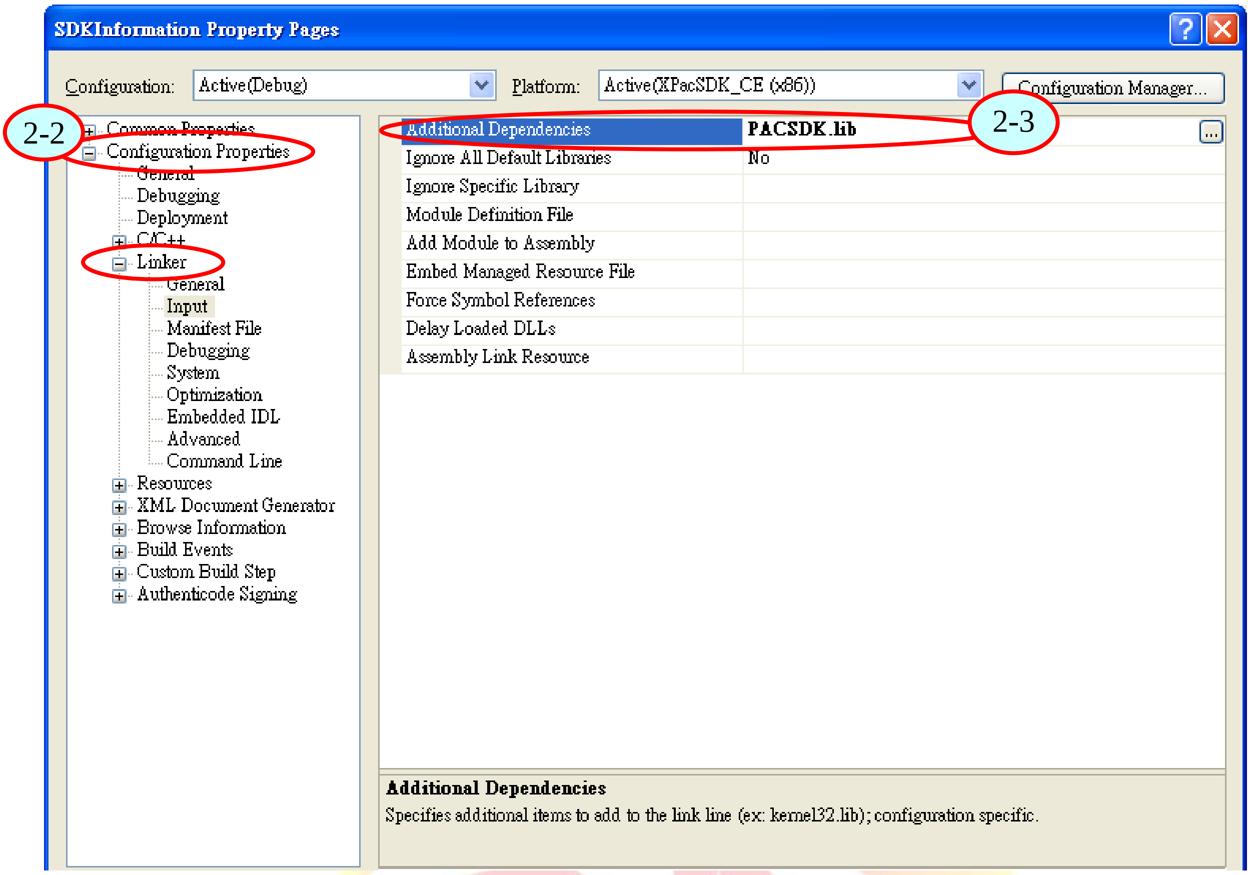
Step 2: Change the XPacSDK_CE.lib reference to PACSDK.lib

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



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

2-3 In the right pane, type “PACSDK.lib” in the Additional Dependencies textbox.



Configuration: Active(Debug)   Platform: Active(XPacSDK_CE (x86))

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.

1. **Error code modification**

```
//if(pac_GetLastError() == PAC_ERR_INTR_CRATE_EVENT_FAILURE)
if(pac_GetLastError() == PAC_ERR_INTR_CREATE_EVENT_FAILURE)
{
    //...
}
```

   The error code,PAC_ERR_INTR_CRATE_EVENT_FAILURE, as defined in XPacSDK_CE.h is misspelled, and it is corrected in PACSDK.h as PAC_ERR_INTR_CREATE_EVENT_FAILURE

2. **API function modification**

```
//pac_EnableLED(pin, bFlag);
pac_EnableLEDs(pin, bFlag);
```

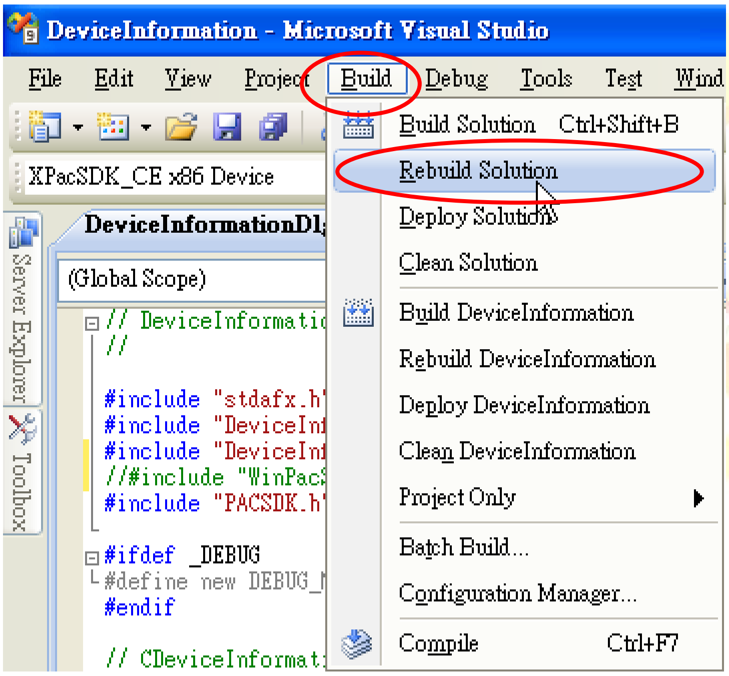
   pac_EnableLED(pin, bFlag) function, as defined in XPacSDK_CE.lib has been renamed as pac_EnableLEDs(pin, bFlag) in PACSDK.lib.

3. **Add the reserved memory section**

In order to reserve some memory sections of EEPROM and SRAM for the use by the system, the reserved section of the pac_ReadMemory and pac_WriteMemory function must be changed.
The definition of the items included in the reserved section is
EEPROM
0 ~0x1FFF (8KB) for users
0x2000~0x3FFF (8KB) is reserved for the system
SRAM
The size of the input range for the SRAM is only 0 ~0x6FFFF (448KB), with another 64KB of SRAM is reserved for use by the system.

Check whether the reserved memory space using in pac_ReadMemory and pac_WriteMemoryour for the previous source code. If yes, change the address to the memory spaces that are used for users.

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



```
// DeviceInformatic
//

#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 PACNET library.

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

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

The PACNET.dll can be obtained from the link below that has been provided on the CD or by downloading the latest version from ICP DAS web site.

CD:\XPAC-ATOM-CE6\SDK\XPacNET

CD:\XP-8000-CE6\SDK\XPacNET

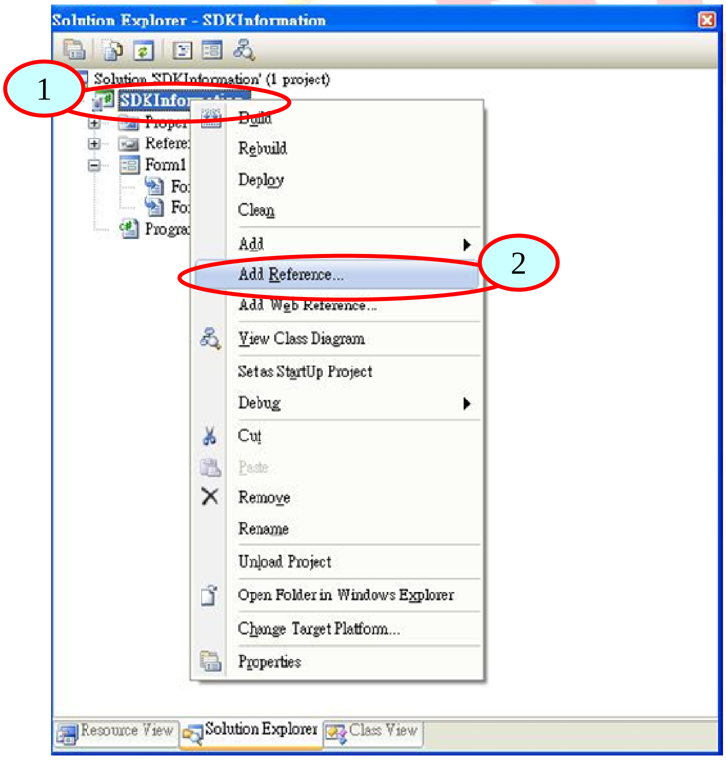
http://ftp.icpdas.com/pub/cd/xpac-atom-ce6/sdk/xpacnet/

http://ftp.icpdas.com/pub/cd/xp-8000-ce6/sdk/xpacnet/

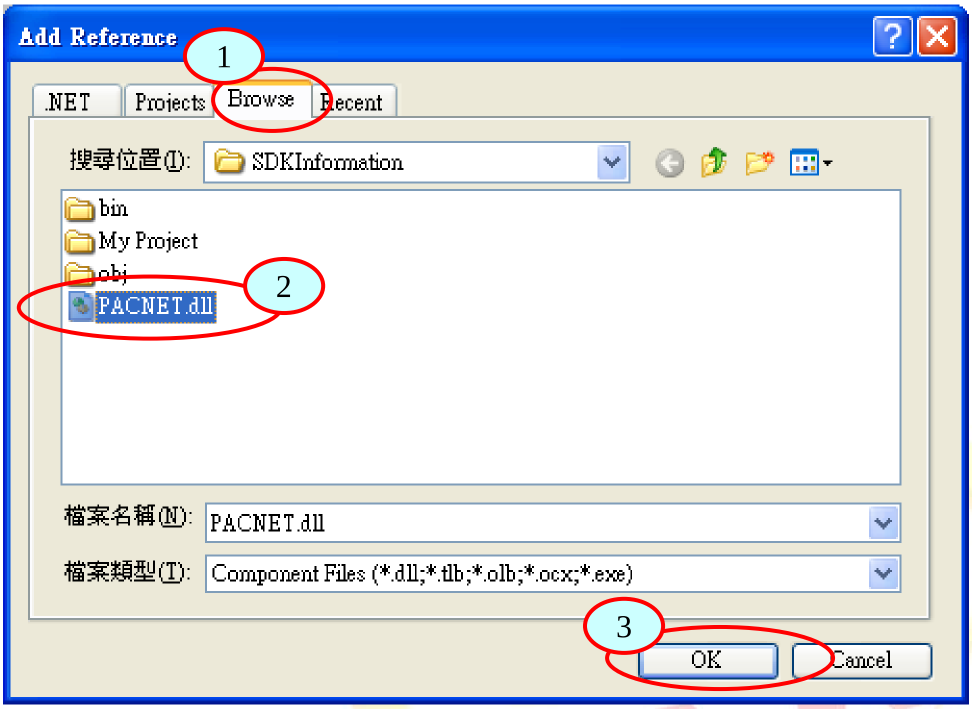
PACNET.dll
2.0.7.24491
PACNET

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

1-3 In the Solution Explorer windows, right-click the References node 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: Follow the example code below to check whether you need to modify your program and amend it where necessary.

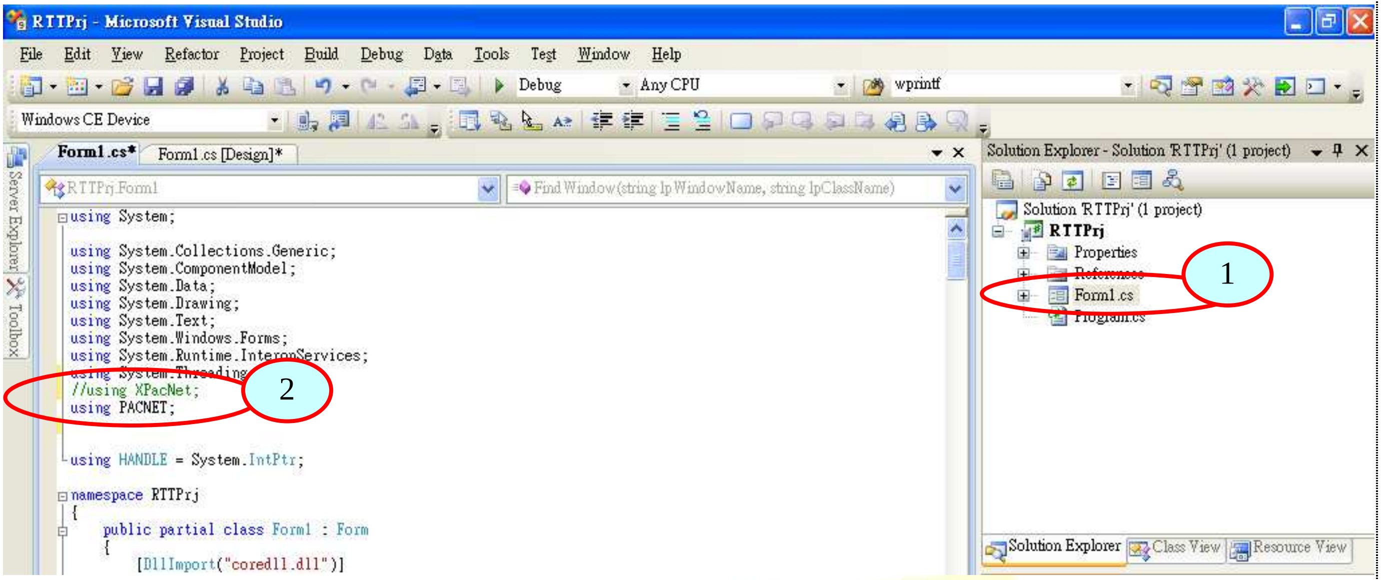
2-1 Modify the code, “using XPacNET” to “using PACNET”.

For a C# program

```
//using XPacNet;
using PACNET;
```

For a VB program

```
//Imports XPacNet
Imports PACNET
```



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
FindWindow(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

```
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")]
```

2-2 API function modification

//XPacNet.XPac.pac_EnableLED(pin, bFlag);
PACNET.SysInfo.pac_EnableLEDs(pin, bFlag);

XPacNet.XPac.pac_EnableLED(pin, bFlag) function defined in XPacNet.dll has been changed as PACNET.SysInfo.pac_EnableLEDs(pin, bFlag) PACNET.dll.

2-3 DllImport Modification

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

In a C# program

//[DllImport("XPacSDK_CE.dll")]
[DllImport("PACSDK.dll")]

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

In a VB program

//<DllImport("XPacSDK_CE.dll")>
<DllImport("PACSDK.dll")>

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

2-4 Class name modification

All API functions for the XPacNet.dll are placed in a single XPacNet.XPac.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

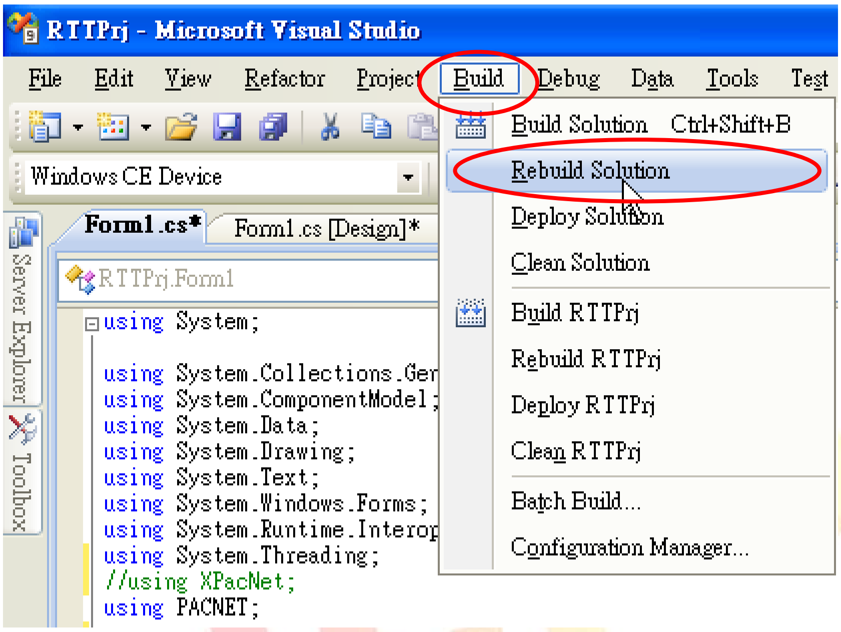
3 Add the reserved memory section

In order to reserve some memory sections of EEPROM and SRAM for the use by the system, the reserved section of the pac_ReadMemory and pac_WriteMemory function must be changed.
The definition of the items included in the reserved section is

EEPROM
0 ~0x1FFF (8KB) for users
0x2000~0x3FFF (8KB) is reserved for the system
SRAM
The size of the input range for the SRAM is only 0 ~0x6FFFF (448KB), with another 64KB of SRAM is reserved for use by the system.

Check whether the reserved memory space using in PACNET.pac_ReadMemory and PACNET.pac_WriteMemoryour for the previous source code. If yes, change the address to the memory spaces that are used for users.

Step 4: From the “Build”menu, select “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 XPacSDK_CE library so that it uses the PACSDK library, and to update the C#/VB.net program that uses the XPacNet 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/xp-8000-ce6/pc_tools/EasyUpgradeSDK
http://ftp.icpdas.com/pub/cd/xpac-atom-ce6/pc_tools/EasyUpgradeSDK

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 steps below to update the SDK using EasyUpgradeSDK.exe

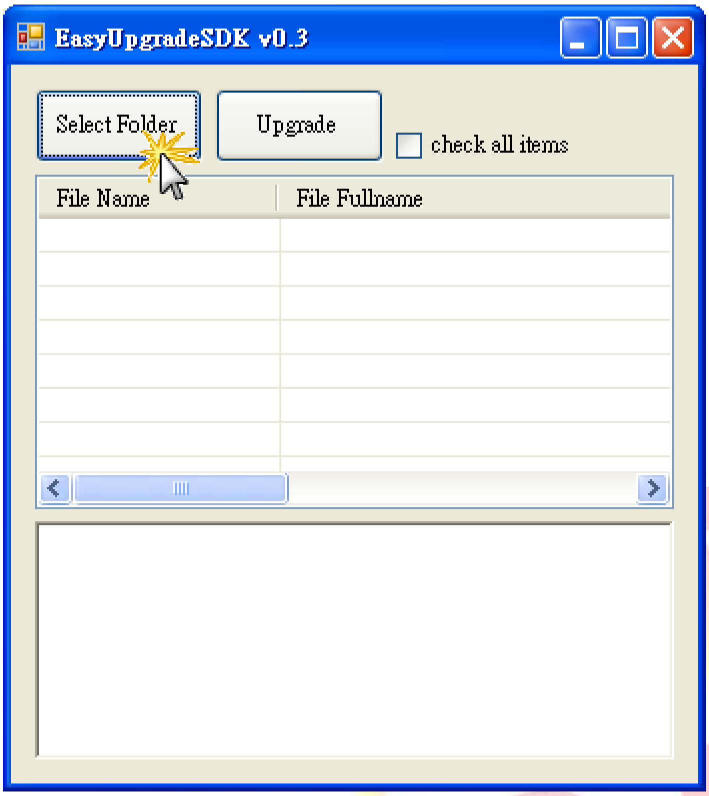
Setp 1: Confirm that the file structure is shown as following, and copy them to a Windows PC.

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

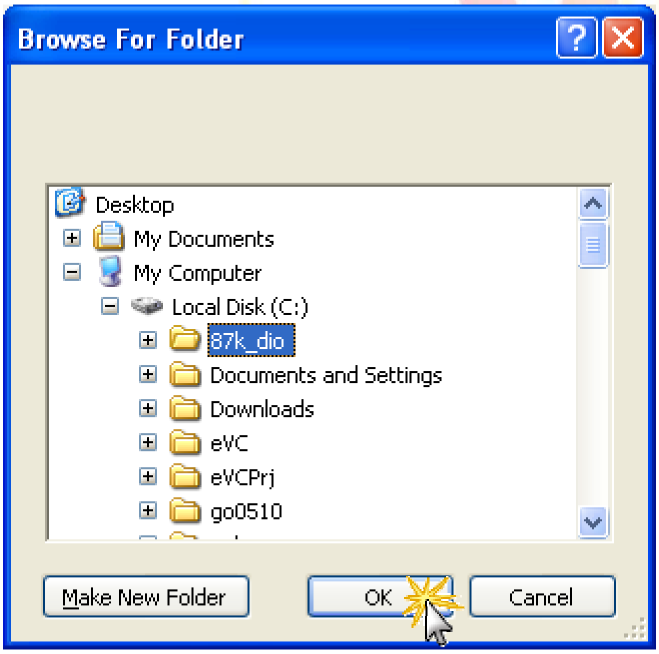
Setp 2: Run the EasyUpgradeSDK.exe

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

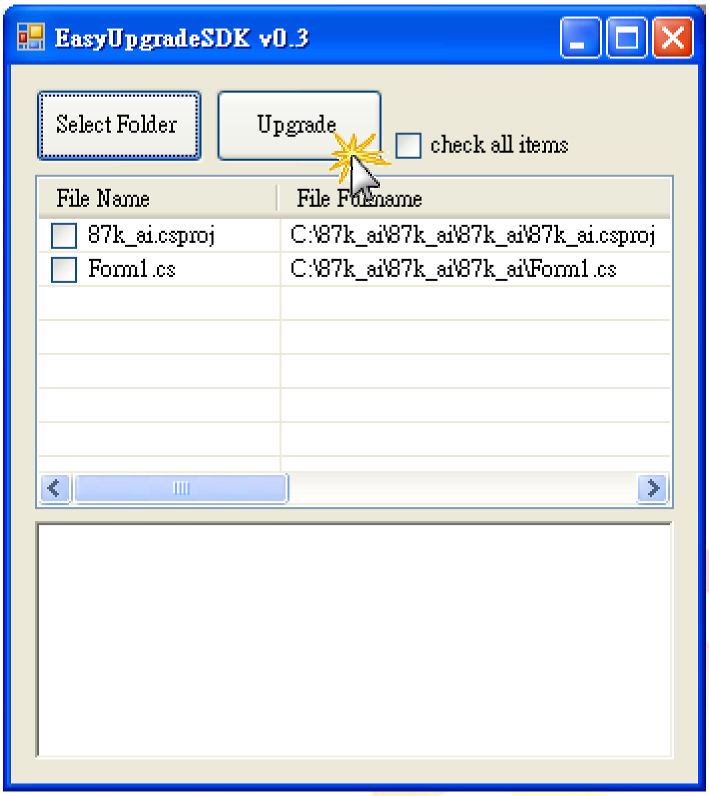
3-1: Click “Select Folder” button



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

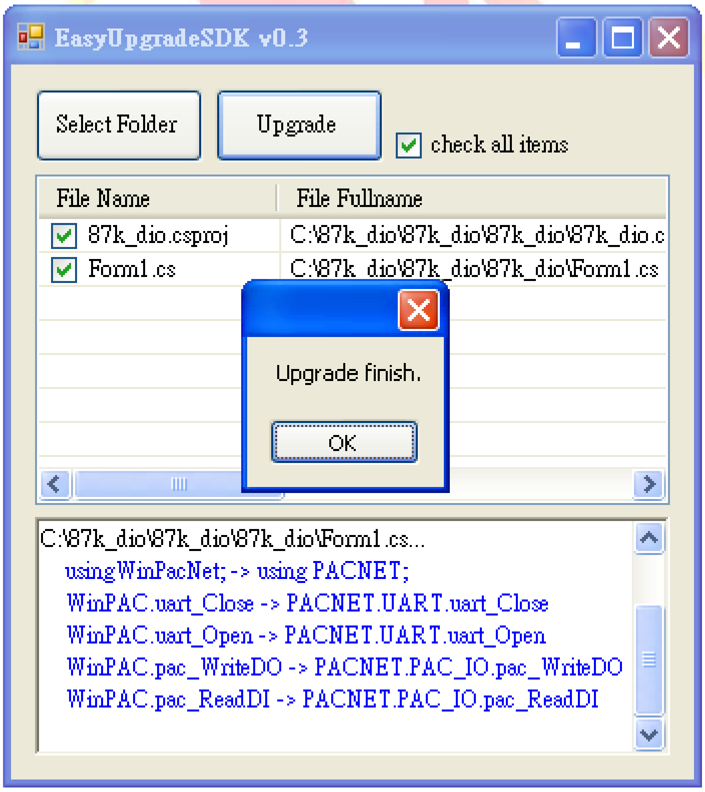


3-3: Click “check all items” checkbox and then click “Upgrade” button



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...
    using WinPacNet; -> 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: After successfully 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.