

Calling DLLs

The User Manual of Calling DLLs

[For Windows 95/98/NT]
(Version 1.0)

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assumes no liability for damage consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, ICP DAS assumes no responsibility for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright 1997-1999 by ICP DAS. All rights are reserved.

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

The user can use, modify and backup this software **on a single machine**. The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Table of Contents

1. INTRODUCTION.....	3
2. DLLS AND FUNCTION CALL	6
2.1 USING VISUAL C++	6
2.2 USING MFC	7
2.3 USING VISUAL BASIC	9
2.4 USING DELPHI.....	10
2.5 USING BORLAND C++ BUILDER.....	11
3. PROBLEMS REPORT	13

1. Introduction

The driver for the products of [PCI/ISA DAQ Card](#) and [I-7000 Series Modules](#), it provides one or more [DLL](#) files (and [Vxd/Sys](#) files) to be used by higher-level computer languages.

The DLL files is written in Visual C++ and provides lots of functions to perform a variety of [Analog input/output](#), [Digital input/output](#), [Counter/Timer](#) and [RS-232/RS-485 Communication](#) operations with the hardware of PCI/ISA DAQ Card and I-7000 Series Modules. The DLL files are [standard Win32 DLL](#) format, and it can be used [under Windows 95/98/NT](#). With these functions of DLL files, the user doesn't need to process the lower-level hardware controls. The DLL files can be used by higher-level computer language easily. For example, it provides lots of demo programs that are written in [Visual C++](#), [Delphi](#), [Borland C++ Builder](#) and [Visual Basic](#).

The subroutines in DLL files are easy understanding as its name standing for. It provides powerful, easy-to-use subroutine for developing your data acquisition, Analog input/output, Digital input/output and RS-232/RS-485 Communication applications.

In this manual, it describes how to call the DLL with Visual C++ 5.0, Visual Basic 5.0, Delphi 3.0 and Borland C++ Builder 3.0. It uses the [PCI-TMC12 DAQ Card](#) to be the example, and all of the DLL files of our products have the similar steps to be called by the higher-level computer languages.

Before the user calling the function that provided by DLL files, the user must install the software/driver firstly. Please remember the folder that the user installs the software into. And the folder will contains all the drivers, demo programs and manuals after the user installed the software/driver.

In additional, the DLL, Vxd and Sys files will be copied into the following folder automatically when the user installs the software/driver.

DLL files	→ C:\Windows\System\	(for Windows 95/98 user)
Vxd files	→ C:\Windows\System\	(for Windows 95/98 user)
DLL files	→ C:\WinNT\System32\	(for Windows NT user)
Sys files	→ C:\WinNT\System32\Drivers\	(for Windows NT user)

The Sys files need to be registered under Windows NT, thus the user must refer to the file README.TXT of the software/driver to create the registry value if the user copy these files manually.

The DLL functions needs the declarations for higher-level computer languages to uses. And the software/driver for our products also provides the completely declaration files for Visual C++ 5.0, Visual Basic 5.0, Delphi 3.0 and Borland C++ Builder 3.0. The user can find these declaration files under the folder “[driver](#)” that the user installs the software/driver into. [The DLL and declaration files are different between Windows 95/98 and Windows NT.](#)

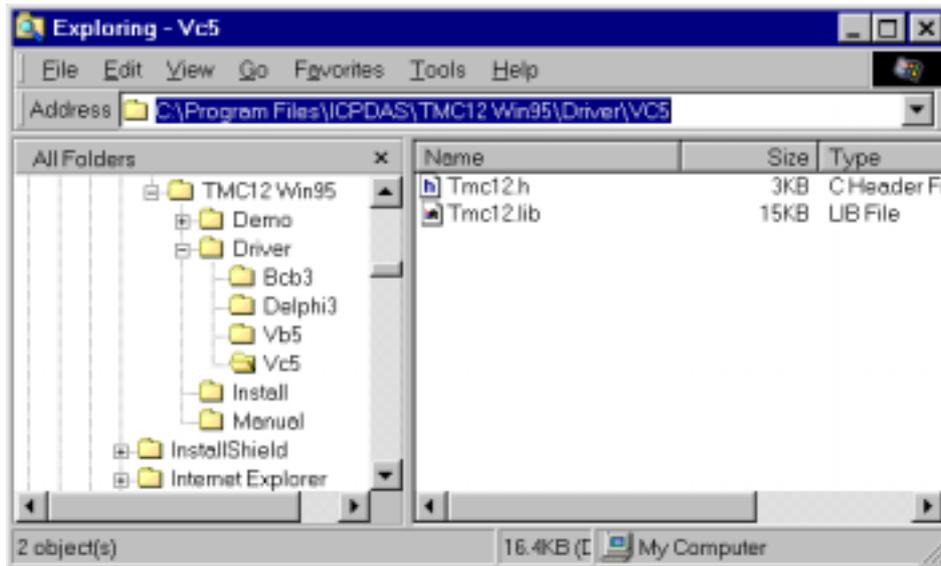


Figure 1-1. The declaration file for Visual C++ 5.0.

For example, the user can find the declaration files “TMC12.h” and “TMC12.Lib” for Visual C++ 5.0 under the folder:

“C:\Program Files\ICPDAS\TMC12 Win95\Driver\VC5”

(The user must remember the folder that the user installs the software into.)

Note: The .Lib files are different for VC++ and BCB.

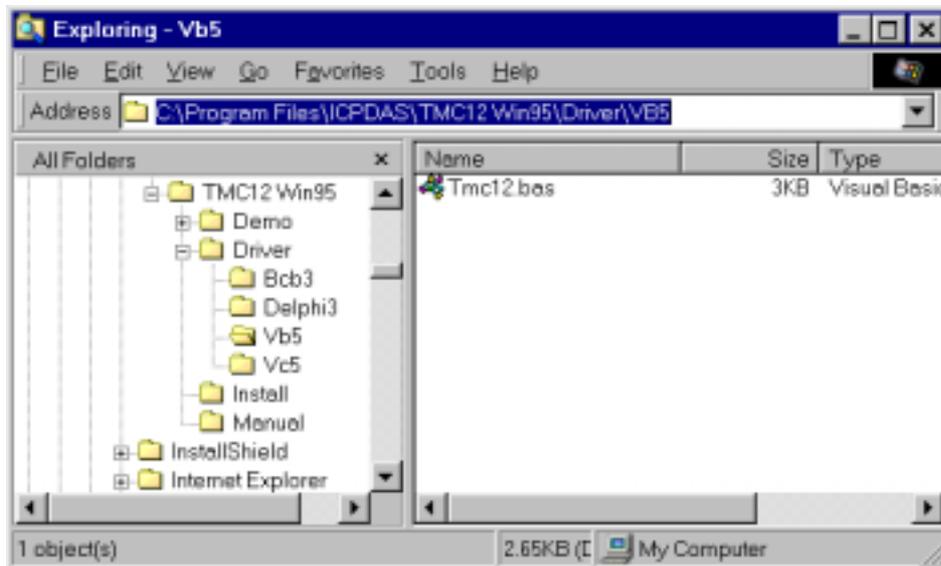


Figure 1-2. The declaration file for Visual Basic 5.0.

For example, the user can find the declaration file “TMC12.BAS” for Visual Basic 5.0 under the folder:

“C:\Program Files\ICPDAS\TMC12 Win95\Driver\VB5”

(The user must remember the folder that the user installs the software into.)

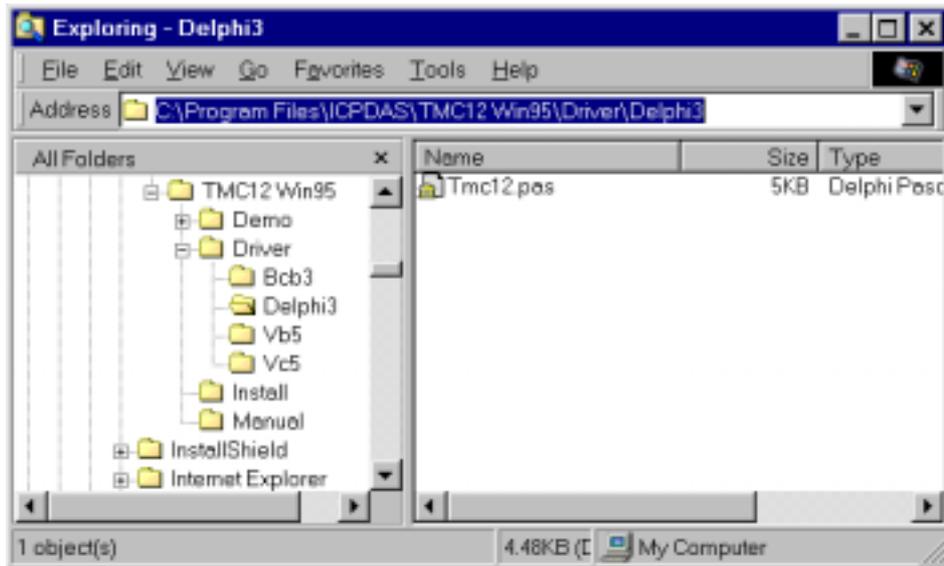


Figure 1-3. The declaration file for Delphi 3.0.

For example, the user can find the declaration file “TMC12.PAS” for Delphi 3.0 under the folder:

“C:\Program Files\ICPDAS\TMC12 Win95\Driver\Delphi3”

(The user must remember the folder that the user installs the software into.)

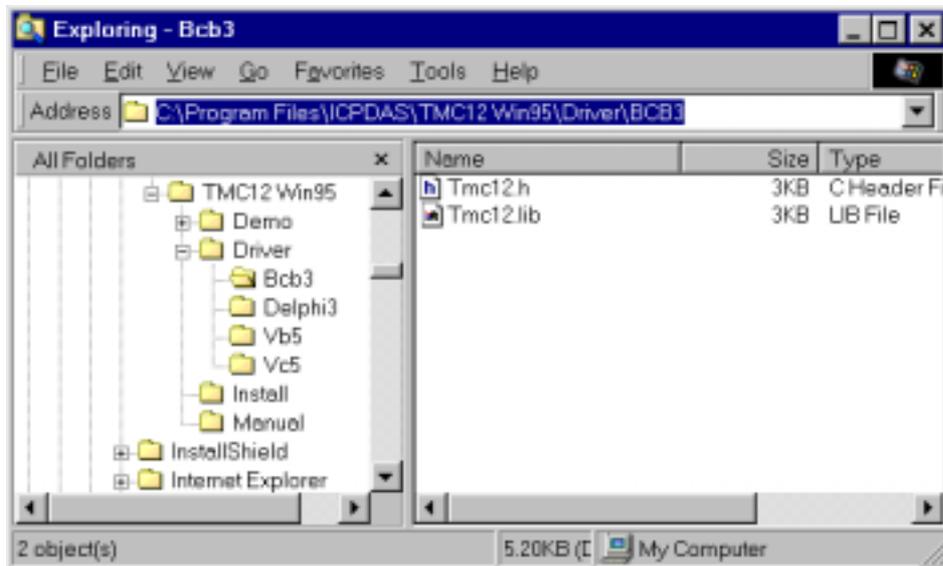


Figure 1-3. The declaration file for Borland C++ Builder 3.0.

For example, the user can find the declaration file “TMC12.H” and “TMC12.Lib” for Borland C++ Builder 3.0 under the folder:

“C:\Program Files\ICPDAS\TMC12 Win95\Driver\BCB3”

(The user must remember the folder that the user installs the software into.)

Note: The .Lib files are different for VC++ and BCB.

2. DLLs and Function Call

2.1 Using Visual C++

Step 1: Execute the `\MSDEV\BIN\VCVARS32.BAT` to setup the environment.

Step 2: Copy the declaration files into the user's project folder.

For example, declaration files: "TMC12.H" and "TMC12.Lib".

Note: The .H and .Lib files are different between VC++ and BCB.

Step 3: The source program must include the declaration file. For example:

```
#include "TMC12.H"
```

Step 4: Edit the source program. (Refer to demo programs.)

Step 5: Edit the MAKE file. (Refer to file `XXX.mak` for demo programs.)

For example:

```
demo1.exe : demo1.obj TMC12.lib
link -SUBSYSTEM:windows -OUT:demo1.exe demo1.obj TMC12.lib
-DEFAULTLIB:user32.lib gdi32.lib winmm.lib comdlg32.lib
comctl32.lib
```

```
Demo1.obj : demo1.c TMC12.h
cl -c -DSTRICT -G3 -Ow -W3 -Zp -Tp demo1.c
```

Step 6: Uses NMAKE to make the user's project. For example:

```
NMAKE /f demo1.mak
```

Step 7: End.

NOTE: The Lib file is used in linking time and the DLL and Vxd is used in run time for Windows 95/98. (The DLL and Sys files for Windows NT.)

2.2 Using MFC

The usage for MFC user is very similar to that for C user. It tests OK under Windows 95/98/NT and Visual C++ 5.0. The key points are given as below:

Step 1: Use MFC wizard to create source code.

Step 2: Copy the declaration files into the user's project folder.

For example, declaration files: "TMC12.H" and "TMC12.Lib".

Note: The .H and .Lib files are different between VC++ and BCB.

Step 3: The source program must include the declaration file. For example:

#include "TMC12.H"

Step 4: Select the menu items "Project" / "Add To Project" / "Files...".

Refer to Figure 2-2-1.

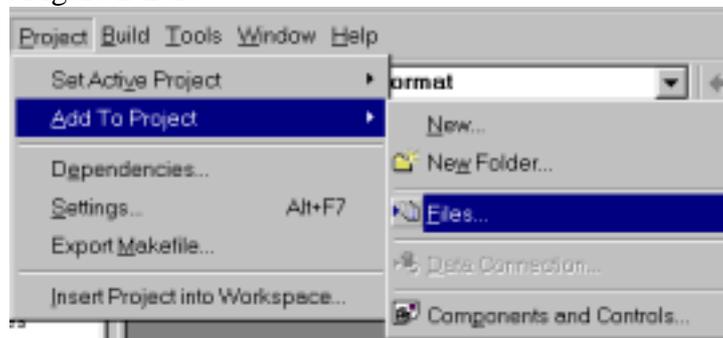


Figure 2-2-1. Select the menu items "Project" / "Add To Project" / "Files...".

Step 5: Select the declaration file to include.

For example, "TMC12.h". Refer to Figure 2-2-2.

Step 6: Click the button "OK". Refer to Figure 2-2-2.

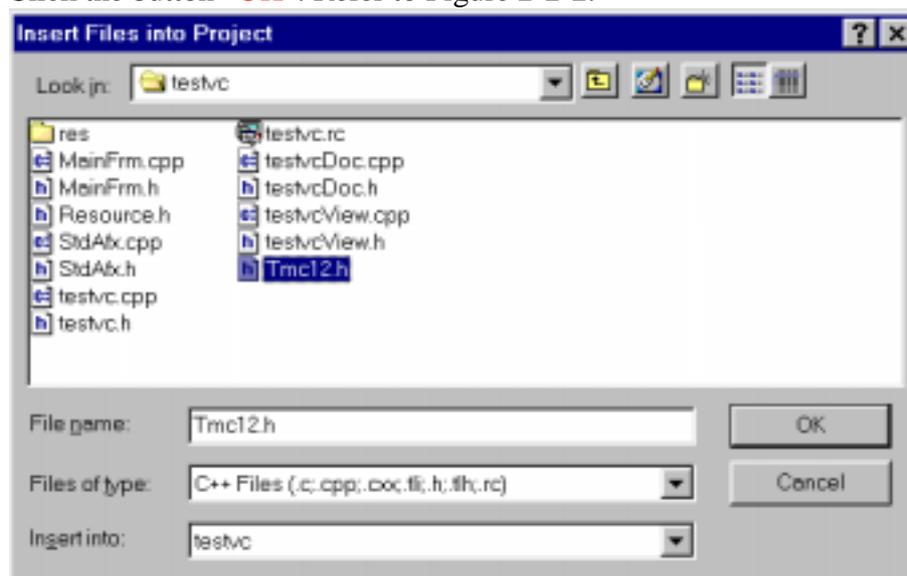


Figure 2-2-2. Select the declaration file to include.

Step 7: Select the menu items “Project” / “Add To Project” / “Files...” again.
Refer to Figure 2-2-3.

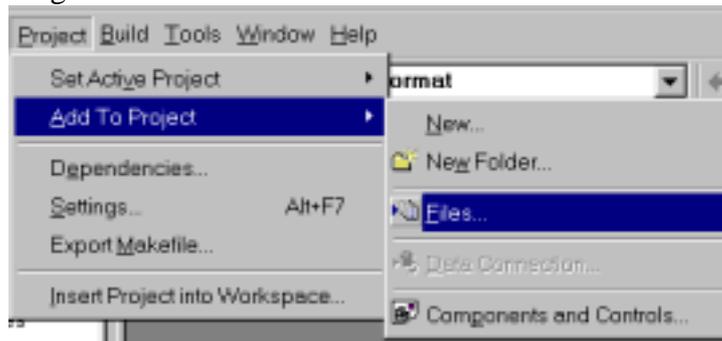


Figure 2-2-3. Select the menu items “Project” / “Add To Project” / “Files...”.

Step 8: Change the field “Files of type:” to “Library Files (.lib)”.

Refer to Figure 2-2-4.

Step 9: Select the library file to include.

For example, “TMC12.lib”. Refer to Figure 2-2-4.

Step 10: Click the button “OK”. Refer to Figure 2-2-4.

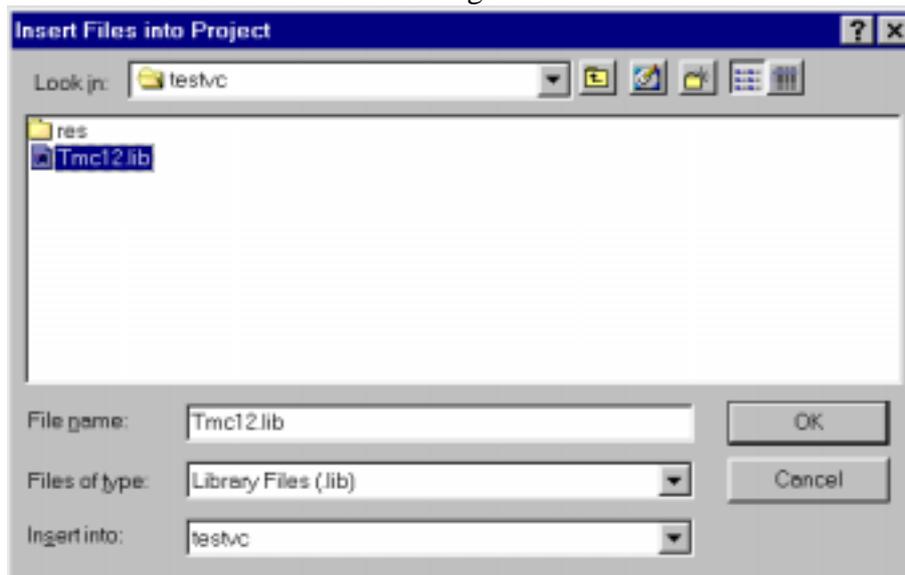


Figure 2-2-4. Change the field “Files of type:” to “Library Files (.lib)”.

Step 11: End.

NOTE: The Lib file is used in linking time and the DLL and Vxd is used in run time for Windows 95/98. (The DLL and Sys files for Windows NT.)

2.3 Using Visual Basic

Step 1: Copy the declaration file into the user's project folder.
For example, "TMC12.BAS".

Step 2: Select the menu items "Project" / "Add Module". Refer to Figure 2-3-1.

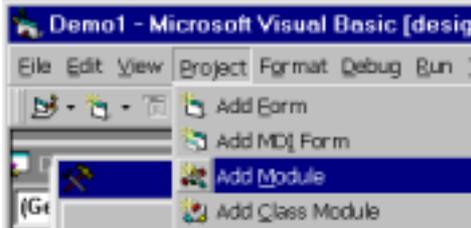


Figure 2-3-1. Select the menu items "Project" / "Add Module".

Step 3: Select the page "Existing". Refer to Figure 2-3-2.

Step 4: Select the declaration file to include.

For example, the declaration file: "TMC12.BAS". Refer to Figure 2-3-2.

Step 5: Click the button "Open". Refer to Figure 2-3-2.

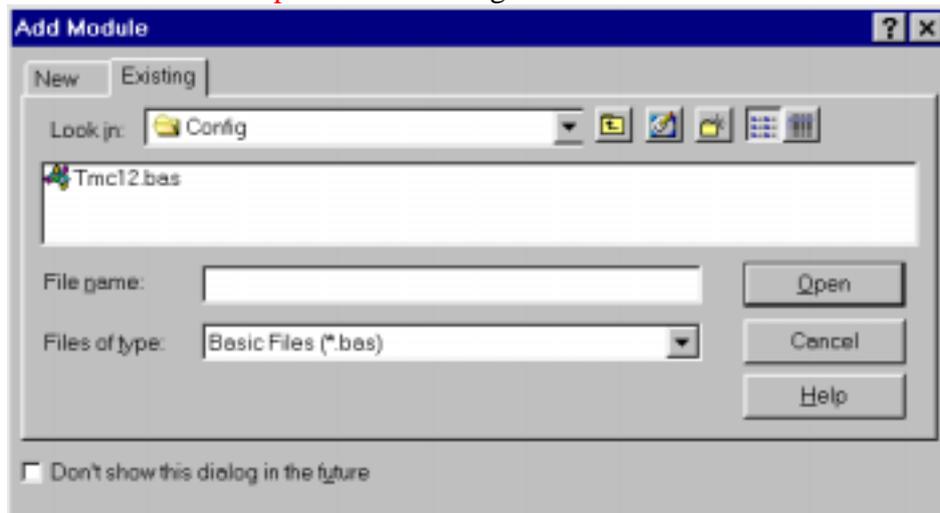


Figure 2-3-2. Select the declaration file to include.

Step 6: Check the project manager if the module had added successfully?
Refer to Figure 2-3-3.

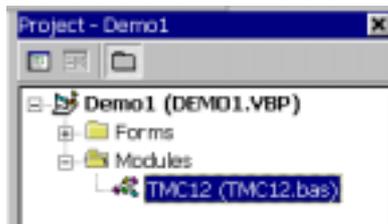


Figure 2-3-3. Check if the module had added into the project?

Step 7: End.

2.4 Using Delphi

Step 1: Copy the declaration file into the user's project folder.
For example, "TMC12.PAS".

Step 2: In the user's source program to uses the declaration file.
For example:
Uses TMC12;

Refer to Figure 2-4-1.

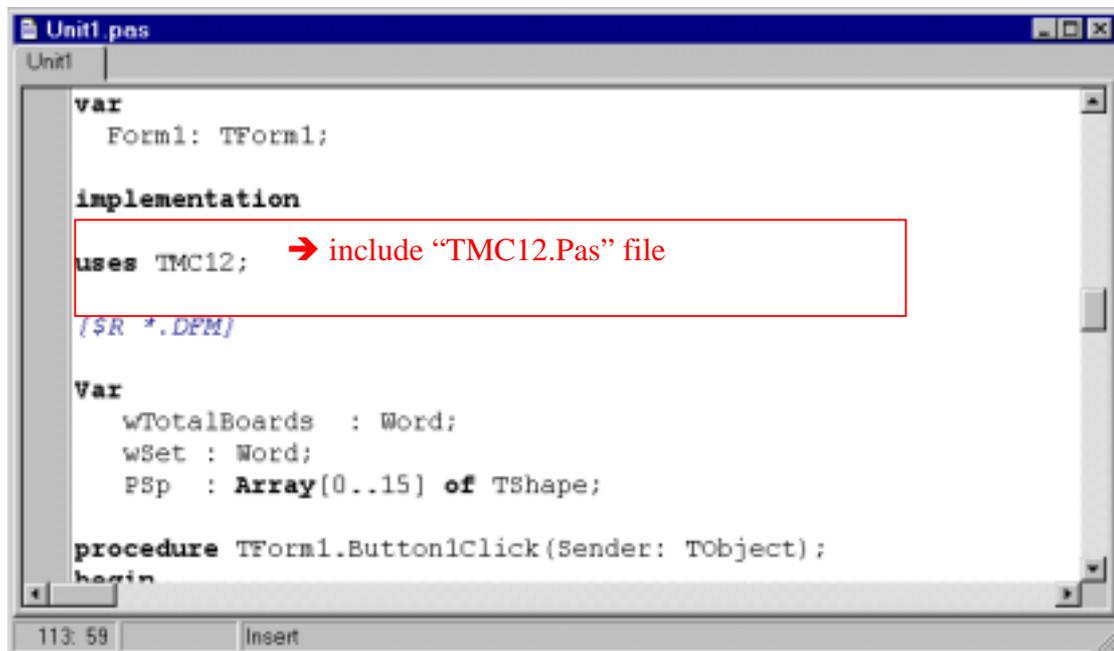


Figure 2-4-1. To uses the declaration file.

Step 3: End.

2.5 Using Borland C++ Builder

Step 1: Copy the declaration files into the user's project folder.

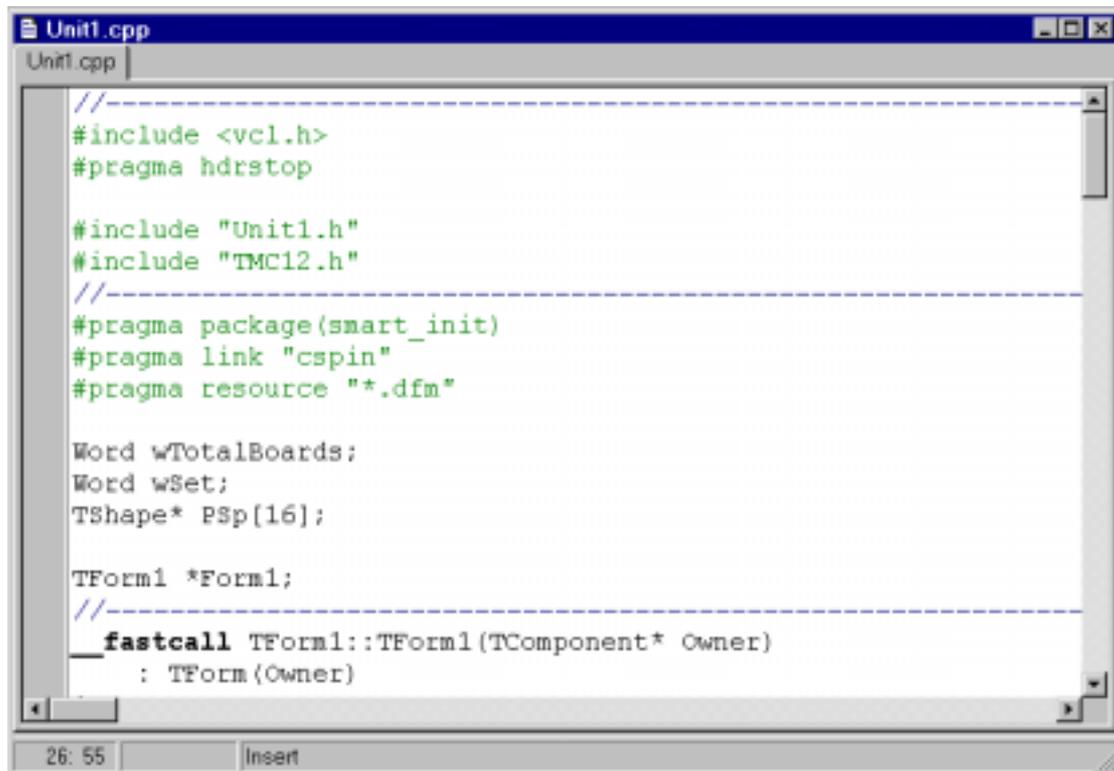
For example, declaration files: "TMC12.H" and "TMC12.Lib".

Note: The .H and .Lib files are different between VC++ and BCB.

Step 2: The source program must include the declaration file. For example:

#include "TMC12.H"

Refer to Figure 2-5-1.



```
Unit1.cpp
Unit1.cpp
//-----
#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "TMC12.h"
//-----
#pragma package(smart_init)
#pragma link "cspin"
#pragma resource "*.dfm"

Word wTotalBoards;
Word wSet;
TShape* Psp[16];

TForm1 *Form1;
//-----
fastcall TForm1::TForm1(TComponent* Owner)
: TForm(Owner)
```

Figure 2-5-1. Include the declaration file.

Step 3: Select the menu items "Project" / "Add to Project...".

Refer to Figure 2-5-2.

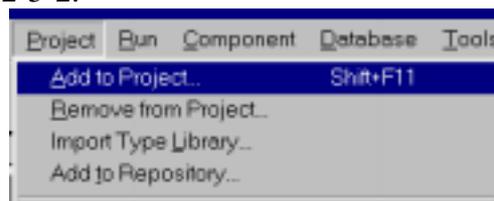


Figure 2-5-2. Select the menu items "Project" / "Add to Project...".

Step 4: Change the field “Files of type:” to “Library file (*.lib)”.
Refer to Figure 2-5-3.

Step 5: Select the library to include.
For example: “TMC12.Lib”. Refer to Figure 2-5-3.

Step 6: Click the button “Open”. Refer to Figure 2-5-3.

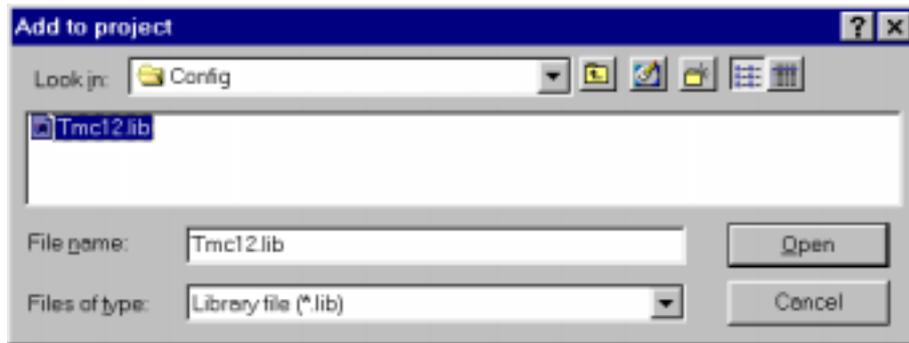


Figure 2-5-3. Select the library to include.

Step 7: Select the menu items “View” / “Project Manager”. Refer to Figure 2-5-4.

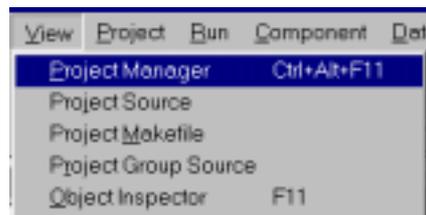


Figure 2-5-4. Select the menu items “View” / “Project Manager”.

Step 8: Check the Project Manager that if the library had added into this project?
Refer to Figure 2-5-5.

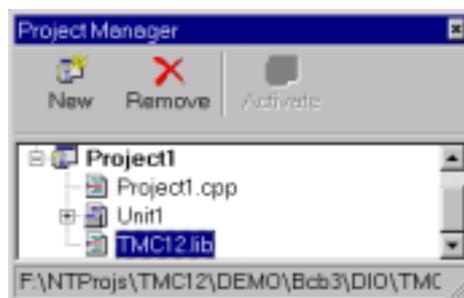


Figure 2-5-5. Check if the library had been added into this project?

Step 9: End.

3. Problems Report

Technical support is available at no charge as described below. The best way to report problems is send electronic mail to

icpdas@ms8.hinet.net

on the Internet.

When reporting problems, please include the following information:

- 1) Is the problem reproducible? If so, how?
- 2) What kind and version of Operation Systems that you running?
For example, Windows 3.1, Windows for Workgroups, Windows NT 4.0, etc.
- 3) What kinds of our products that you using? Please see the product's manual .
- 4) If a dialog box with an error message was displayed, please include the full text of the dialog box, including the text in the title bar.
- 5) If the problem involves other programs or hardware devices, what devices or version of the failing programs that you using?
- 6) Other comments relative to this problem or any suggestions will be welcomed.

After we had received your comments, we will take about two business days to test the problems that you said. And then reply as soon as possible to you. Please check that if we had received your comments? And please keeps contact with us.

E-mail: icpdas@ms8.hinet.net