

Using WinUSB in a Visual Studio Project with Freescale USB device controller

by: Paolo Alcantara
Microcontroller Solutions Group

Contents

1 Introduction

Freescale has different MCUs and MPUs with a USB device controller for 8/16/32-bit architectures. Windows provides default USB drivers for standard USB classes like human interface device (HID) class, mass storage device (MSD) or communication device class (CDC). However Windows requires the development of new USB drivers when using a customized USB class for a specific use or a Windows non-supported USB class. This application note explains how to skip the development of a Windows USB device driver by using a USB DLL named FSLwinusb_v2.dll. The dynamic link library (DLL) is based on a generic USB driver provided by Microsoft named WinUSB. The MCF51JM128 is the device used to test the USB device controller compatibility with WinUSB.

1.1 Scope

The following document presents information about FSLwinusb_v2.dll and how to integrate with a Microsoft Visual Studio project. Details for Visual Basic (VB) or C# project are explained in this document.

1	Introduction.....	1
1.1	Scope.....	1
1.2	Audience description.....	1
2	WinUSB Overview.....	2
3	Using FSLwinusb_v2.....	2
3.1	Running the example software and firmware.....	2
4	Adding FSLwinusb_v2.dll.....	9
4.1	FSLwinusb_v2.dll inside a C# project.....	9
4.2	FSLwinusb_v2.dll inside a VB project.....	12
5	FSLwinusb_v2.dll API.....	13
6	Conclusion.....	13
6.1	Problem reporting instructions.....	13
6.2	Considerations and References.....	14

1.2 Audience description

This document is intended to be used by all software development engineers, test engineers, and anyone else who is integrating a USB device driver in a PC application.

2 WinUSB Overview

WinUSB is a generic USB driver provided by Microsoft. The application programming interface (API) is intuitive in providing access to: bulk, interrupt and control endpoints. Currently, isochronous endpoints are not supported. Control endpoint's data are exposed to the user, after that any USB setup packet can be built. WinUSB is supported with the following Windows OSes:

- Windows XP with SP2 or later
- Windows Vista
- Windows 7 and later

WinUSB works with 32 and 64-bit Windows editions by using the same USB setup information file (INF file). The following architectures are supported:

- x86
- amd64
- Itanium (ia64)

FSLwinusb_v2.dll contains WinUSB functionality in the form of a Windows DLL. FSLwinusb_v2.dll hides the WinUSB initialization to the user, and then the DLL can be used by the Visual Studio suite. This application note is tested with the following:

- Microsoft Visual C# 2008 Express Edition
- Microsoft Visual Basic 2008 Express Edition

3 Using FSLwinusb_v2

The following documents are suggested to understand the details mentioned in the next sections:

- Microsoft WinUSB reference document (WinUsb_howto.docx) available from <https://www.microsoft.com>
- Demo JM User Manual (DEMOJMUM.pdf) available from <https://www.freescale.com>
- Demo JM Quick Start Guide (DEMOJMQSG.pdf) available from <https://www.freescale.com>
- Demo JM Schematics (DEMOJM_Schematics.pdf) available from <https://www.freescale.com>

3.1 Running the example software and firmware

The following steps explain how to test the FSLwinusb_v2.dll test included with this application note.

1. Unzip file AN4378SW.zip.
2. Open {INSTALLATION DIR}\WinUSBExample in C#\WinUSBExample\bin\Release\WinUSBExample.exe. The following GUI is shown in Figure 1. FSLwinusb_v2.dll file must be in the same folder of the executable file.

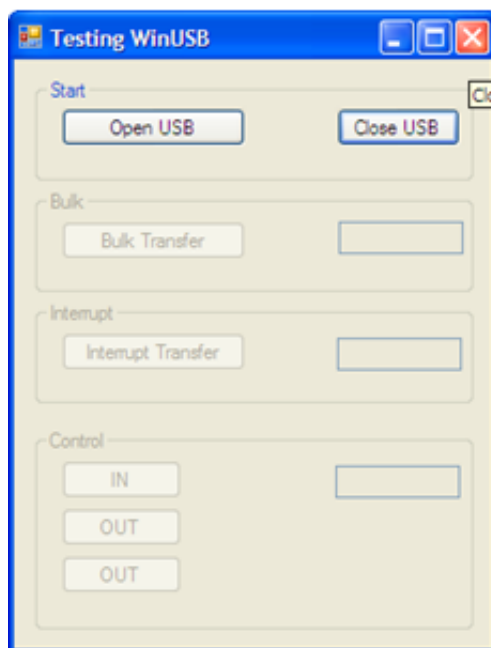


Figure 1. C# application

3. Download and install CMX USB V1 stack from JM128 web page. Installer unzips CMX USB V1 SW stack in C:\
4. Copy and paste "WinUSB Test 2.0" directory from AN4378SW.zip inside C:\CMXUSB_LITE_V1 directory.
5. Back up usb_config.h file contained in C:\CMXUSB_LITE_V1\usb-peripheral\src\mcf51xx\usb-driv\.. Then copy the file C:\CMXUSB_LITE_V1\WinUSB Test 2.0\Sources\drivers\usb\usb_config.h and use this file to replace backup file in C:\CMXUSB_LITE_V1\usb-peripheral\src\mcf51xx\usb-driv\.
6. Open MCP file using CodeWarrior v6.3 (Classic Edition) located in C:\CMXUSB_LITE_V1\WinUSB Test 2.0\WinUSB Test 2.0.mcp as shown in Figure 2.

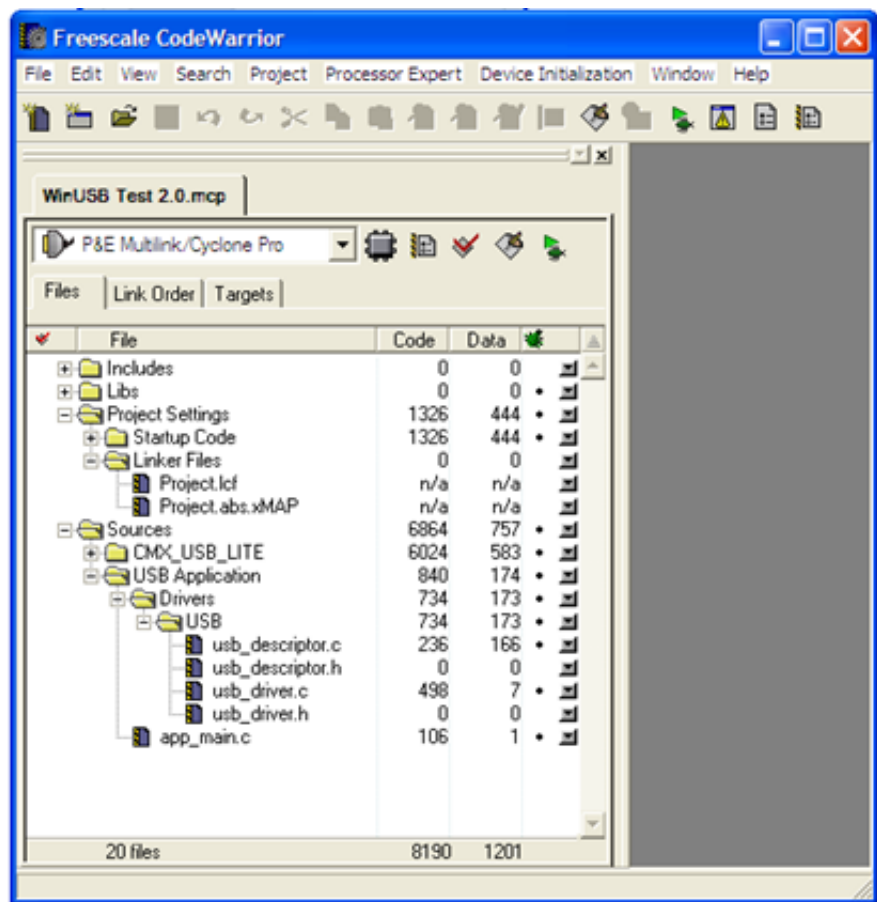


Figure 2. WinUSB Test using CodeWarrior

7. Set jumpers of JM128 according to DEMOJMUM.pdf default jumper position. This file also contains details on how to download an application using PEmicro debugger.
8. Compile and download to Freescale JM128 board.
9. Use miniUSB cable to connect between PC and JM128 board. The following figure must be shown. The DEMOJMUM.pdf contains details on how to switch between PEmicro and MiniUSB power source.

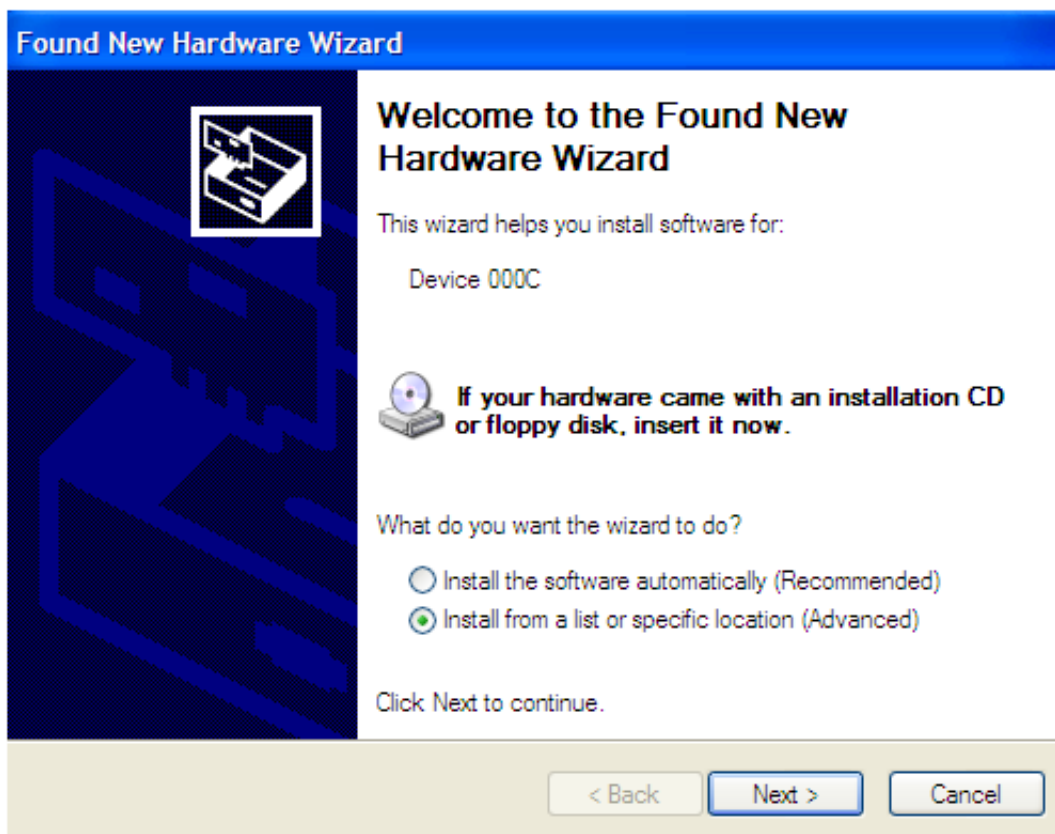


Figure 3. New USB device detected by Windows

10. Use a customized Windows USB driver by selecting “JM128 WinUSB drivers” folder from AN4378SW.zip. Click next as shown in Figure 4. Then USB device is installed and recognized by Windows as shown in Figures 5 and 6.

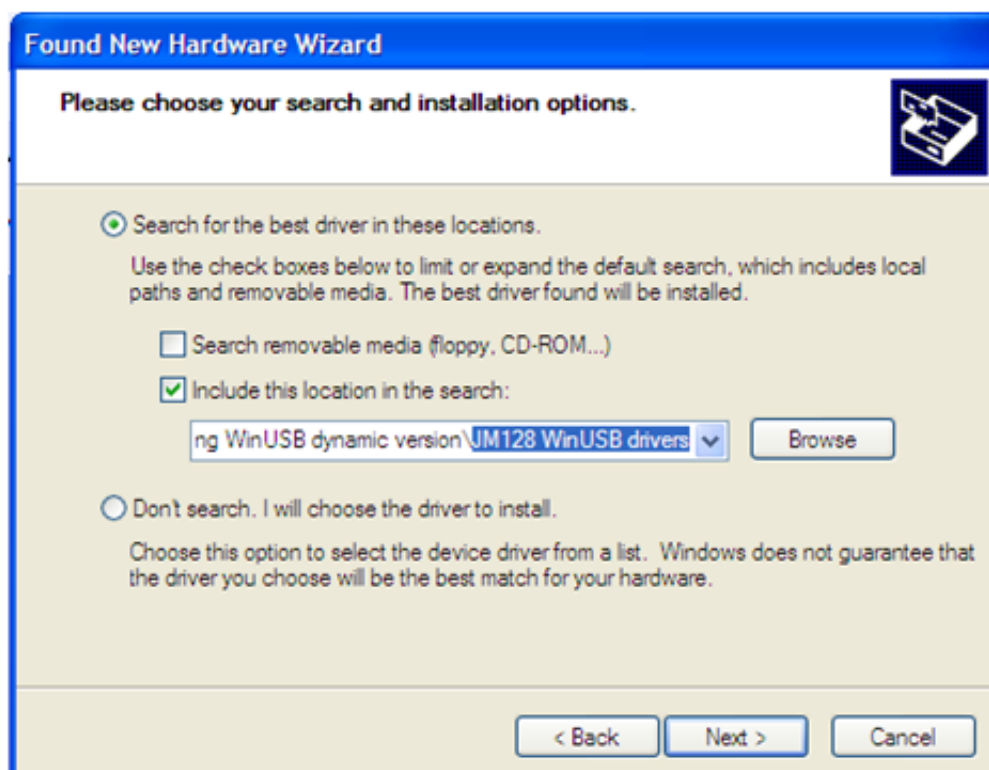


Figure 4. JM128 USB drivers are selected as the drivers for the USB device

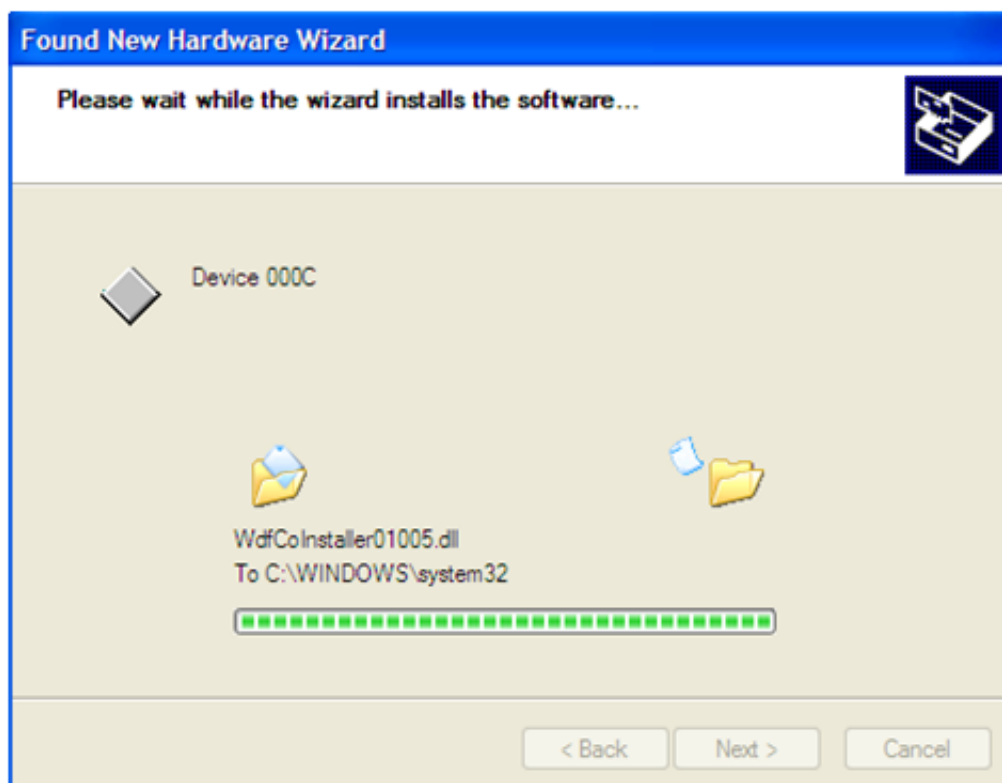


Figure 5. USB device is installed

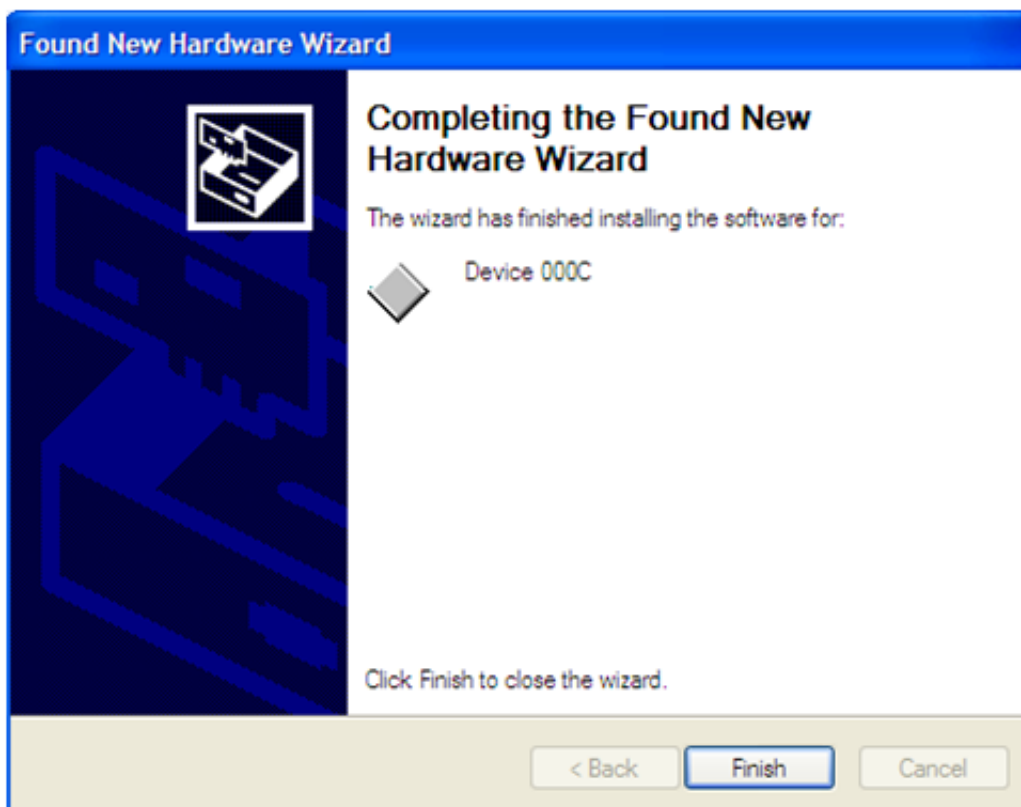


Figure 6. USB device recognized by Windows

11. Using C# application, press “Open USB” button. Then a number appears inside the upper text field of the GUI as shown in Figure 7. GUI and USB communication can be stopped by pressing “Close USB”. Basic connection and communication is completed.

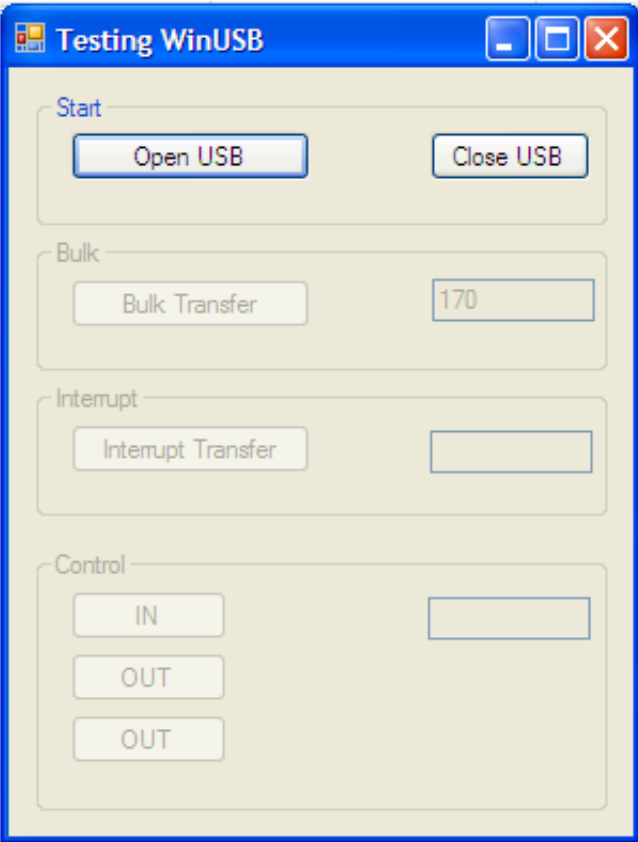


Figure 7. C# application communicating with the USB device

12. Communication can be observed by using a USB packet analyzer as shown in Figure 8. USB analyzer used is Ellisys Visual USB viewer. The log file is inside “USB Log Files” folder in the AN4378SW.zip.

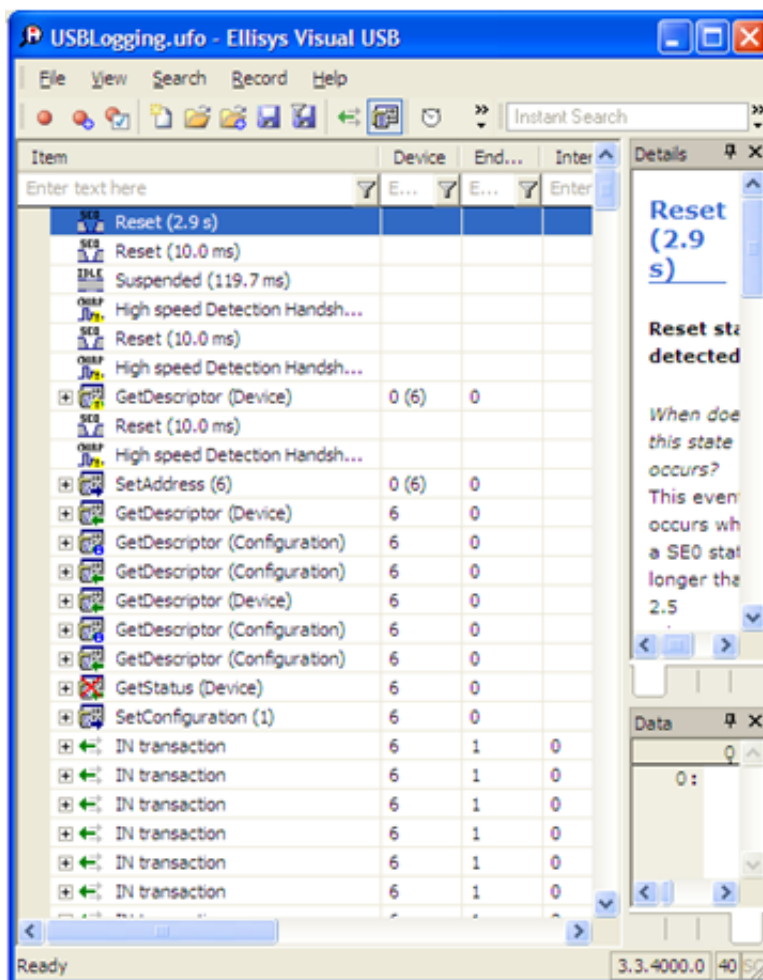


Figure 8. USB log showing USB transactions

NOTE

Direct connection of the USB cable to the PC's USB port is strongly advised. The use of a USB hub or docking station for this application note is not recommended.

4 Adding FSLwinusb_v2.dll

The following sections explain how to add FSLwinusb_v2.dll inside a C# and VB projects.

4.1 FSLwinusb_v2.dll inside a C# project

There are two versions of FSLwinusb_v2.dll:

- Static version
- Dynamic version

The static version can create a single instance of WinUSB. The dynamic version can create more than one instance. For the C# example code, the dynamic version is used. The following steps show the procedure needed to add FSLwinusb_v2.dll dynamic version:

1. Go to “WinUSBExample in C#\WinUSBExample\WinUSBExample.sln” from AN4378SW.zip. Open it with VSC#2008. WinUSBExample project is shown in Figure 9.

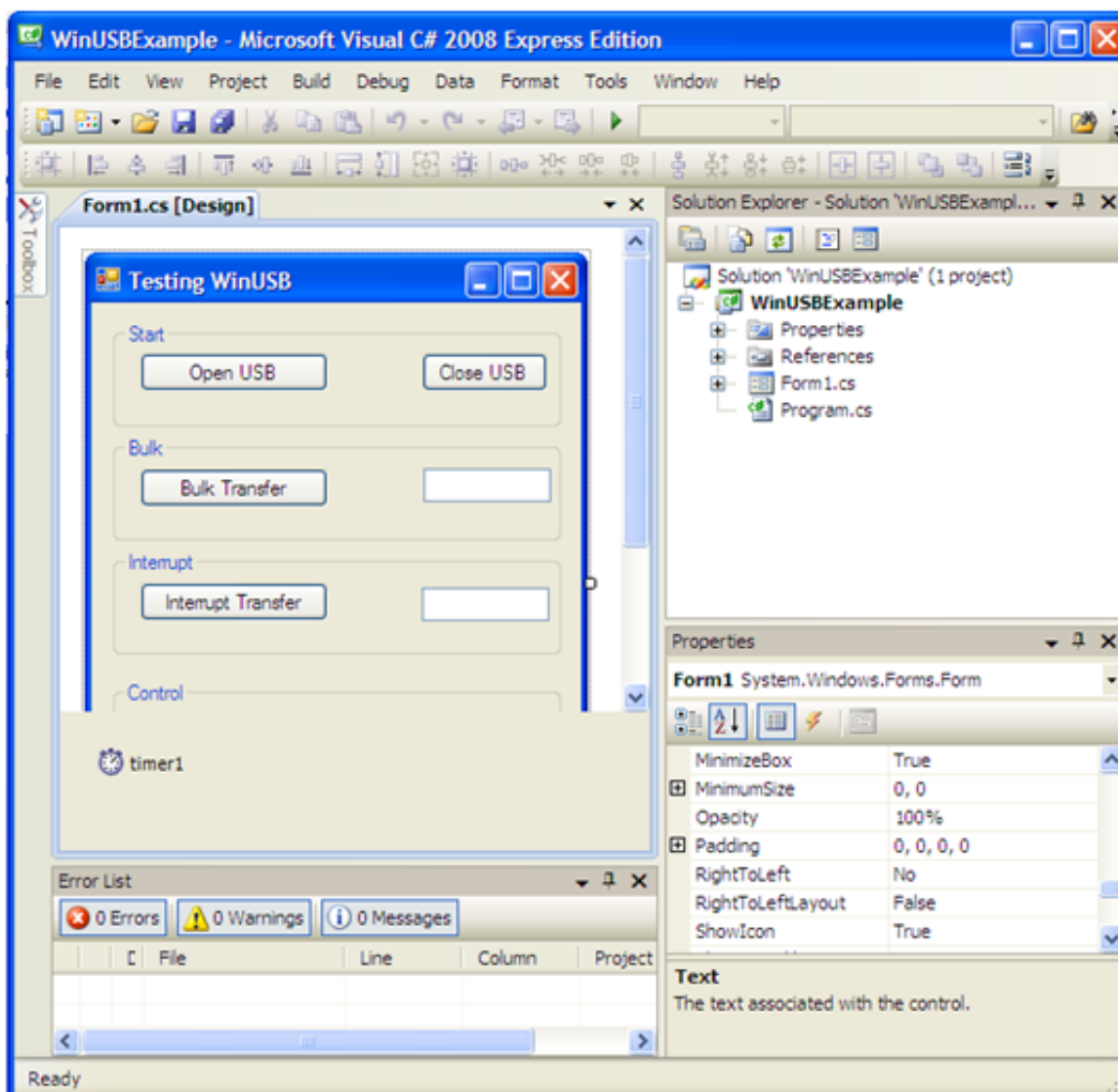


Figure 9. C# project source code

2. Add FSLwinusb_v2.dll as a reference file. See Figure 10, 11 and 12.

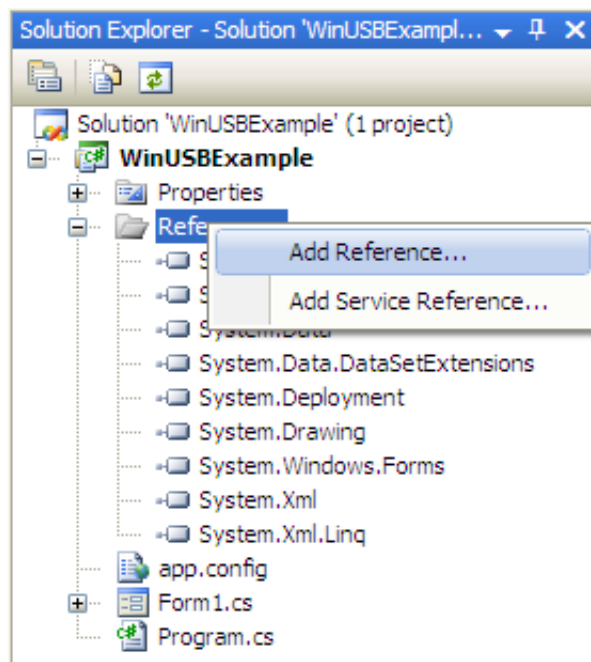


Figure 10. adding FSLwinusb_v2.dll as a reference

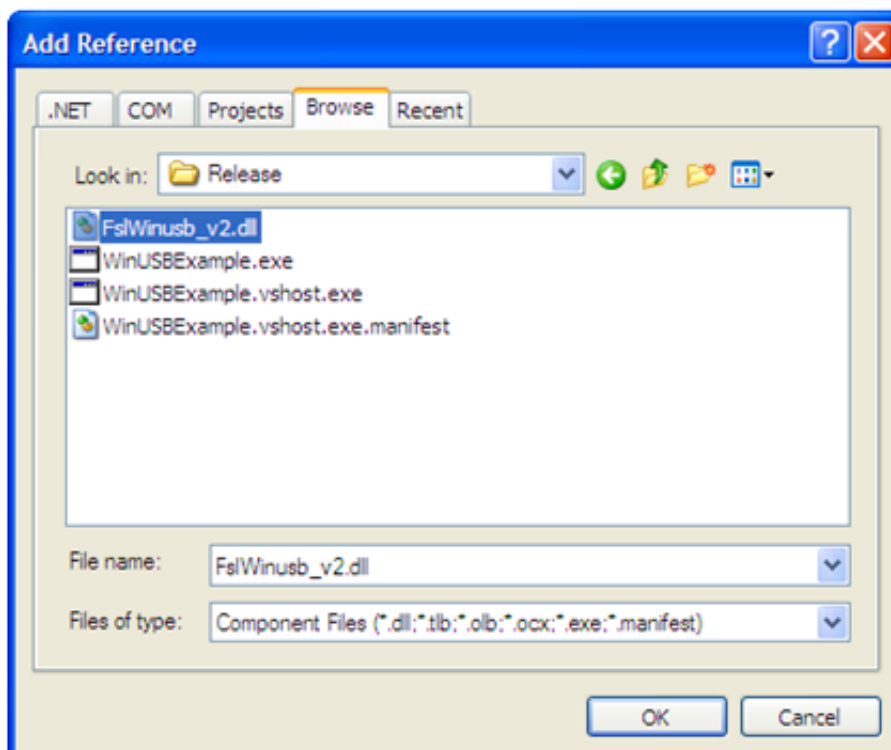


Figure 11. FSLwinusb_v2.dll added as a reference

3. Figure 13 shows some lines included in the Form1.cs from the C# project. The important parts of the source code needed by FSLwinusb_v2.dll are highlighted in red rectangles. Note that the GUID passed to the usb_open_device() function is the same as the one listed in the INF file from the USB device.

NOTE

FSLwinusb_v2.dll dynamic version requires the use of the reserved word “new” during USB init.

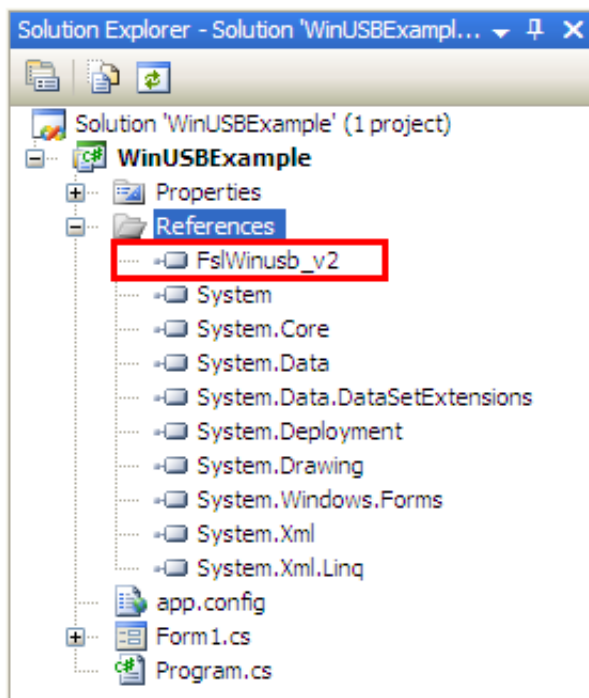


Figure 12. C# source code using FSLwinusb_v2.dll

4.2 FSLwinusb_v2.dll inside a VB project

The next figure shows how to call FSLwinusb_v2.dll static version from a VB project. Visual Basic does not support the “new” reserved word, and then only the static version can be used. FSLwinusb_v2.dll is added in the same way as stated in Section 3.2 for C#.

NOTE

FSLwinUSB is called directly without using the “new” reserved word.

A different import reserved word is used with the static version.

```
Imports FslWinusb.FslWinusb

...

my_guid.DataB = &H8F51
my_guid.DataC = &H4C8E
my_guid.Data0 = &H87
my_guid.Data1 = &H2B
my_guid.Data2 = &H3E
my_guid.Data3 = &H7F
my_guid.Data4 = &H53
my_guid.Data5 = &HCC
my_guid.Data6 = &HF1
my_guid.Data7 = &H82

'opening USB device with new GUID
On Error GoTo wrong_label
bGood = usb_open_device(my_guid) 'usb open device() is in FslWinUSB.DLL
If bGood = True Then
    bGood = fnWinusb_QueryPipe(0, 0, sWinusb_ep0)
    bGood = fnWinusb_QueryPipe(0, 1, sWinusb_ep1)
    bGood = fnWinusb_QueryPipe(0, 2, sWinusb_ep2)
    bGood = fnWinusb_QueryPipe(0, 3, sWinusb_ep3) 'EP4

    bGood = usb_SetPipePolicy(sWinusb_ep0.PipeId, PIPE_TRANSFER_TIMEOUT, 4, 1000)
    bGood = usb_SetPipePolicy(sWinusb_ep1.PipeId, PIPE_TRANSFER_TIMEOUT, 4, 10)
    bGood = usb_SetPipePolicy(sWinusb_ep2.PipeId, PIPE_TRANSFER_TIMEOUT, 4, 10)
    bGood = usb_SetPipePolicy(sWinusb_ep3.PipeId, PIPE_TRANSFER_TIMEOUT, 4, 10)

    fnWinusb_OpenConfig_DeviceL = True
End If
wrong_label:
    On Error GoTo 0
End Function
```

Figure 13. VB project using FSLwinUSB_v2.dll static version

5 FSLwinusb_v2.dll API

The complete API is available at Microsoft website. `usb_open_device()` is the only function that is not available, which requires a GUID as the single parameter and returns a Boolean variable if USB device is correctly opened. Additionally, the first parameter of all functions is handled by the DLL. The name of the suppressed parameter is “WINUSB_INTERFACE_HANDLE InterfaceHandle”. The FSLwinUSB_v2.dll handles this parameter internally, and then this parameter is omitted from the whole FSLwinUSB_v2.dll API.

6 Conclusion

This Application note explains how to integrate WinUSB in a PC application to use with a Freescale embedded device. The following solution can be used with any Freescale 8/16/32-bit MCU or MPU by adapting the firmware.

6.1 Problem reporting instructions

Issues and suggestions about this document and drivers must be provided through the support web page at <https://www.freescale.com/support>. Please reference this application note number in the support ticket.

6.2 Considerations and References

The latest software updates and information about Using FSLwinusb_v2.dll in a Visual Studio Project is available on the Freescale Semiconductor home page <https://www.freescale.com>.

- More details about WinUSB is available at <https://www.microsoft.com>.
- The AN4378SW contains all the necessary SW to run a customized USB class in the Freescale MCF51JM128 device and a PC running Windows OS.
- Download the source files for AN4378SW from <https://www.freescale.com>.

How to Reach Us:

Home Page:

www.freescale.com

Web Support:

<http://www.freescale.com/support>

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductors products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claims alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-complaint and/or non-Pb-free counterparts. For further information, see <http://www.freescale.com> or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© 2011 Freescale Semiconductor, Inc.