

Technical Note

TN_153

Instructions on Including the D2XX Driver in a VS Express Project

Version 1.1

Issue Date: 2018-06-06

The purpose of this technical note is to provide instructions on how to create a project using Microsoft Visual Studio Express and how to include the D2XX Driver in that project.

Use of FTDI devices in life support and/or safety applications is entirely at the user's risk, and the user agrees to defend, indemnify and hold FTDI harmless from any and all damages, claims, suits or expense resulting from such use.

Future Technology Devices International Limited (FTDI) Unit 1, 2 Seaward Place, Glasgow G41 1HH, United Kingdom Tel.: +44 (0) 141 429 2777 Fax: + 44 (0) 141 429 2758 Web Site: <u>http://ftdichip.com</u> Copyright © Future Technology Devices International Limited



Table of Contents

1 In	troduction 2
1.1	Software Required2
1.2	Hardware Required2
2 Cr	eating the Project
2.1	Initial Setup3
2.2	Building a Win32 Application which uses the FTD2XX.dll7
2.3	Building an x64 application which uses the FTD2XX.dll10
2.4	Building an application which uses the static library11
2.5	Building a solution for Windows XP with VS2013 Express.12
3 Co	ontact Information 13
Арре	endix A – References 14
Doc	ument References14
Acro	onyms and Abbreviations14
Арре	endix B – List of Tables & Figures 15
List	of Tables15
List	of Figures15
Арре	endix C – Revision History 16



1 Introduction

This document provides instructions on the flow required to create a project using Microsoft Visual Studio Express, which uses the FTDI D2XX driver. The Windows CDM driver package contains the following files used by the D2XX interface:

- ftd2xx.h C/C++ header file
- Dynamic library:
 - o ftd2xx.lib for 32-bit systems in folder i386
 - o ftd2xx.dll for 32-bit systems in folder i386
 - ftd2xx.lib for 64-bit systems in folder amd64
 - o ftd2xx.dll for 64-bit systems in folder amd64
- Static library:
 - ftd2xx.lib for 32-bit systems in folder Static\i386
 - o ftd2xx.lib for 64-bit systems in folder Static\amd64

The following pages will show how to create a project which uses the dynamic library or the static library.

An application which uses the dynamic library only accesses the dll code needed by the program when the program starts, or is running. This makes the executable file smaller and also offers other advantages such as better memory utilisation, since the code is only loaded once for all processes using the library. Applications can also make use of any updates to the dynamic library (as long as these maintain calling and return values) without the need for recompiling or relinking.

An application which is built using the static library includes the necessary code from the library in the executable file. This means that there is no dependence on another file (the dll in the dynamic case) for the application to run. The executable file will be larger as a result, but its self-contained character may be an advantage in some situations. If it is desired to include any changes made to the static library, then the application would need to be recompiled.

The example project performs a simple loopback test on an FTDI USB to serial converter IC, for example the FT232R or FT230X. The only hardware connection required is that the RXD pin of the serial converter IC is connected to the TXD pin. The application sets the baud rate (9600), the data characteristics (8 data bits, 1 stop bit and no parity) and selects no flow control. The message 'Hello World' is sent and received by the serial converter IC.

1.1 Software Required

FTDI D2XX driver (2.12.26 or later) which can be downloaded from: <u>http://www.ftdichip.com/Drivers/D2XX.htm</u>

Microsoft Visual Studio Express for Windows Desktop. 2015 version was used in this Technical Note which can be downloaded from:

https://www.visualstudio.com/en-us/products/visual-studio-express-vs.aspx

1.2 Hardware Required

PC with Windows OS installed. Visual Studio Express 2015 supports Windows 7 Service Pack 1, Windows 8, Windows 8.1, Windows 10, Windows Server 2008 R2 SP1, Windows Server 2012 & Windows Server 2012 R2.

An FTDI USB to serial converter, e.g. FT230X or FT232R, in <u>module</u> or <u>cable</u> form. For the loopback test TXD needs to be connected to RXD.



2 Creating the Project

2.1 Initial Setup

The following pages assume that a customer has connected an FTDI USB to serial device, e.g. an FT232R or FT230X, and that the driver has been loaded. The RXD pin of the serial device needs to be connected to the TXD pin.

When Visual Studio Express is first opened then a screen similar to Figure 2.1 is displayed.



Figure 2.1 - The opening screen for VS Express 2015 for Windows Desktop

To create a new project from this screen, select File \rightarrow New \rightarrow Project from main Visual Studio Express toolbar or select New Project from the Start Page. A window will pop up as per Figure 2.2 which allows selection of the project language, type and name.



New Project					? ×
▷ Recent	:	Sort by:	Default 👻 🏭		Search Installed Templates (Ctrl+E)
▲ Installed			Win32 Console Application	Visual C++	Type: Visual C++
▲ Templates ▷ Visual C# ▷ Visual Basic		5	Win32 Project	Visual C++	A project for creating a Win32 console application
✓ Visual C++ CLR			Empty Project	Visual C++	
General Test Win32 SQL Server Visual Studio So Samples	olutions	5	Makefile Project	Visual C++	
P Online			Click here to go online and find templates.		
<u>N</u> ame:	TN_153_Loopback	_Examp	le		
Location:	Location: C:\Users\Username\Documents\FTDI\Visual Studio\				Browse
Solution name: TN_153_Loopba		_Examp	le		Create directory for solution
					Add to Source Control OK Cancel

Figure 2.2 - New Project Window

This document provides instructions on creating a Visual C++ Win32 Console Application, therefore select this option. Fill in the 'Name', 'Location' and 'Solution Name' fields, tick 'Create directory for solution' and then select OK. The pop-up window shown in Figure 2.3 will be shown – select Next to continue.

Win32 Application Wizard - TN_1	53_Loopback_Example
Welcome	to the Win32 Application Wizard
Overview Application Settings	These are the current project settings: • Console application Click Finish from any window to accept the current settings. After you create the project, see the project's readme.txt file for information about the project features and files that are generated. Vertice Next > Finish Cancel

Figure 2.3 – New Project Window 2



Figure 2.4 shows the final setup window before the coding screen. Select 'Console Application', 'Precompiled header' and 'Security Development checks' which are the defaults.

Then select Finish.

Win32 Application Wizard - TN_15	Win32 Application Wizard - TN_153_Loopback_Example				
Applicatio	on Settings				
Overview Application Settings	Application type: Windows application Console application DLL Static library Additional options: Empty project Export symbols Precompiled header Security Development Lifecyde (SDL) checks	Add common header files for:			
	< Previous	Next > Finish	Cancel		

Figure 2.4 - Final setup window before coding screen

A new project is now displayed with the C++ template as per Figure 2.5.







```
Remove the pre-populated code and insert the sample code below - see Figure 2.6.
// TN_153_Loopback_Example
#include "stdafx.h"
#include <windows.h>
#include "ftd2xx.h"
int main()
{
       FT_HANDLE ftHandle;
       FT_STATUS ftStatus;
       ftStatus = FT_Open(0, &ftHandle);
       ftStatus |= FT_SetUSBParameters(ftHandle, 4096, 4096); // Set USB transfer sizes
       ftStatus |= FT_SetChars(ftHandle, false, 0, false, 0); // Disable event characters
       ftStatus |= FT_SetTimeouts(ftHandle, 5000, 5000); // Set read/write timeouts to 5 sec
       ftStatus |= FT_SetLatencyTimer(ftHandle, 16); // Latency timer at default 16ms
       ftStatus |= FT_SetFlowControl(ftHandle, FT_FLOW_NONE, 0x11, 0x13); // No flow control
       ftStatus |= FT_SetBaudRate(ftHandle, 9600); // Baud rate = 9600
                        FT_SetDataCharacteristics(ftHandle,
                                                               FT_BITS_8,
       ftStatus
                  |=
                                                                              FT_STOP_BITS_1,
FT_PARITY_NONE);
       if (ftStatus != FT_OK) printf("ftStatus not ok %d\n", ftStatus); //check for error
       else
       {
       char data_out[12] = "Hello World";
       DWORD w_data_len = 12;
       DWORD data_written;
       ftStatus = FT Write(ftHandle, data out, w data len, &data written);
       char data_in[12];
       DWORD r_data_len = 12;
       DWORD data_read;
       ftStatus = FT_Read(ftHandle, data_in, r_data_len, &data_read);
       if (ftStatus != FT OK)
               printf("ftStatus not ok %d\n", ftStatus);
       else
               printf("Data Read: %s\n", data_in);
       }
       ftStatus = FT_Close(ftHandle);
       printf("Press Return To End Program");
       getchar();
       return 0;
```

}





Figure 2.6 - Insertion of D2XX loopback sample code

Note: Errors will be highlighted until required files are included into the project. See the next section.

2.2 Building a Win32 Application which uses the FTD2XX.dll

Copy the ftd2xx.h header file and the 32-bit dynamic ftd2xx.lib (from driver folder i386) to the project folder (TN_153_Loopback_Example\TN_153_Loopback_Example) where the other header files (stdafx.h and targetver.h) are located – see Figure 2.7.

Visual Studio 🔸 TN_153_Loopback_Example 🔸 TN_153_Loopl	oack_Example 🗸	Search TN_153_Loopba
Burn New folder		8-
Documents library TN_153_Loopback_Example		Arrange
Name	Date modified	Туре
ftd2xx.h	16/03/2016 10:07	H File
ftd2xx.lib	16/03/2016 10:07	Object File Library
ReadMe.txt	26/07/2016 14:41	Text Document
📄 stdafx.cpp	26/07/2016 14:41	CPP File
📄 stdafx.h	26/07/2016 14:41	H File
🛅 targetver.h	26/07/2016 14:41	H File
TN_153_Loopback_Example.cpp	26/07/2016 14:43	CPP File
💁 TN_153_Loopback_Example.vcxproj	26/07/2016 14:41	VC++ Project
TN_153_Loopback_Example.vcxproj.filters	26/07/2016 14:41	VC++ Project Filters File

Figure 2.7 - Copy ftd2xx.h & ftd2xx.lib to the project folder



Next, press Alt-F7 in Visual Studio Express to display the project's Property Pages – see **Figure 2.8**.

TN_153_Loopback_Example Property	Pages			? ×	
Configuration: All Configurations		Platform: All Platforms	•	Configuration Manager	
 Configuration Properties 	⊿	General			
General		Target Platform	Windows		
Debugging		Target Platform Version	8.1		
VC++ Directories		Output Directory	<different options=""></different>		
▷ C/C++		Intermediate Directory	<different options=""></different>		
▶ Linker		Target Name	\$(ProjectName)		
Manifest Tool		Target Extension	.exe		
XIVIL Document Generator		Extensions to Delete on Clean	*.cdf;*.cache;*.obj;*.obj.enc;*.ilk;*.ipdb;*.iobj;*.resources;*.tlb;*.tli;*.t		
browse information browse information		Build Log File	\$(IntDir)\$(MSBuildProjectName).log		
Custom Build Sten		Platform Toolset	Visual Studio 2015 (v140)		
Code Analysis		Enable Managed Incremental Build	No		
,,	4	Project Defaults			
		Configuration Type	Application (.exe)		
		Use of MFC	Use Standard Windows Libraries		
		Character Set	Use Unicode Character Set		
		Common Language Runtime Support	No Common Language Runtime Sup	oport	
		.NET Target Framework Version			
		Whole Program Optimization	<different options=""></different>		
		Windows Store App Support	No		
	Targ The	r get Platform e current target platform of the project.			
			ОК	Cancel Apply	

Figure 2.8 - Project property pages

Expand Configuration Properties->Linker->Input, and ensure all 'All Configurations' and 'All Platforms' are selected – see **Figure 2.9**. Now add ftd2xx.lib to the Additional Dependencies field – see **Figure 2.10**.

Click OK to finish.

•	TN_153_Loopback_Example Property Pages		8 23
	Configuration: All Configurations	Platform: All Platforms	✓ Configuration Manager

Figure 2.9 - Select 'All Configurations'



_133_LOODDack	_Example Property F	rages					
onfiguration:	All Configurations		m: All Platforms			•	Configuration Manager
 Configurati 	on Properties	Additional Depe	endencies	ftd2xx	.lib;kernel32.lib;user	32.lib;gdi32	.lib;winspool.lib;comdlg32.
General		Ignore All Defau	ılt Libraries				
Debugg	ing	Ignore Specific I	Default Libraries				
VC++ D	irectories	Module Definiti	on File				
▷ C/C++		Add Module to	Assembly				
 Linker 		Embed Manage	d Resource File				
Gen	eral	Force Symbol R	eferences				
Inpu	ıt	Delay Loaded D	lls				
Mar	ifest File	Assembly Link R	Resource				
Deb	ugging	Assembly Ellik I	(cooline)				
Syst	em						
Opti	imization						
Emb	edded IDL						
Win	dows Metadata						
Adv	anced						
All C	Options						
Con	nmand Line						
Manifes	t Tool						
XML Do	cument Generator						
Browse	Information						
Build Ev	ents						
Custom	Build Step						
Code A	nalysis						
		Additional Depend	encies				
		Specifies additional	items to add to the I	link command	line. [i.e. kernel32.lil	b]	
•	4						
						OK	
						UK	Cancel <u>Apply</u>

Figure 2.10 - ftd2xx.lib is added to Additional Dependencies

Select either Debug or Release mode, and x86 (compatible with the ftd2xx.lib file copied from the i386 folder), from the drop down boxes on the main Visual Studio Express toolbar. Debug mode will allow breakpoints to be set and the code to be stepped through, whereas release mode will generate an executable file.

Now Select Build Solution (F7) from the main Visual Studio Window. The output window should show that the build has succeeded – see Figure 2.11.



Figure 2.11 - Output window after successful build

Connect the FTDI USB to serial device (with TXD connected to RXD) and hit F5 to run the application with debugging, or Ctrl+F5 without debugging. The program will execute and the output screen shown in Figure 2.12 will be displayed.





Figure 2.12 - Program output screen

2.3 Building an x64 application which uses the FTD2XX.dll

Select x64 as shown in Figure 2.13 - Selecting x64

×	TN_15	3_Loopba	ack_Examp	le - Micro	osoft Visua	l Studio E	xpress 20	15 for W	indows Desk	ttop
File	Edit	View	Project	Build	Debug	Team	Tools	Test	Window	Help
	• ©	智 當	12 d ²	9-0	🗧 👻 🛛 De	bug 🔸	хб4		- 🕨 Lo	cal Windows Debugger 👻

Figure 2.13 – Selecting x64

Replace the 32-bit ftd2xx.lib file previously copied to the project directory (**Figure 2.7**) with the 64-bit ftd2xx.lib file from the amd64 driver directory.

Select either Debug or Release mode from the drop down box on the main Visual Studio Express toolbar. Debug mode will allow breakpoints to be set and the code to be stepped through, whereas release mode will generate an executable file.

Now Select Build Solution (F7) from the main Visual Studio Window. The output window should show that the build has succeeded – see **Figure 2.14** – Output window after successful build



Figure 2.14 – Output window after successful build

Connect the FTDI USB to serial device (with TXD connected to RXD) and hit F5 to run the application. The program will execute and the output screen shown in Figure 2.12 will be displayed.



2.4 Building an application which uses the static library

The steps for building an application which uses the static library are the same as the dynamic build except for the following:

At the stage shown in **Figure 2.7**, copy the required ftd2xx.lib (either 32 or 64 bit) from the driver package Static folder, instead of the dynamic ftd2xx.lib, to the project directory where the header files are located.

In the project's Property Pages, expand Configuration Properties \rightarrow C/C++ \rightarrow Preprocessor, select 'All Configurations' and 'All Platforms' for Configuration and add FTD2XX_STATIC to the Preprocessor Definitions – see **Figure 2.15 – Preprocessor definition for static build**

TN_153_Loop	pback_Example Property P	ages			? ×
Configurat	tion: All Configurations	▼ Platform:	All Platforms	•	Configuration Manager
▲ Config	guration Properties	Preprocessor Definit	tions	FTD2XX_STATIC; <different options<="" td=""><td>> •</td></different>	> •
Ge	eneral	Undefine Preprocess	sor Definitions		
De	ebugging	Undefine All Preproc	cessor Definitions	No	
VC	C++ Directories	Ignore Standard Incl	ude Paths	No	
▲ C/	/C++	Preprocess to a File		No	
	General	Preprocess Suppress	Line Numbers	No	
	Optimization	Keep Comments		No	
	Preprocessor				
	Code Generation				
	Language				
	Precompiled Headers				
	Output Files				
	Browse Information				
	Advanced				
	All Options				
	Command Line				
D Lii	nker				
	anifest I ool				
	VIL Document Generator				
D Br	uld Events				
	ustom Ruild Sten				
	nde Analysis				
, v ci	oue Analysis				
		Preprocessor Definitio	ns		
		Defines a preprocessing	symbols for your sour	ce file.	
	4 III				
				ОК	Cancel Apply

Figure 2.15 – Preprocessor definition for static build

In Configuration Properties \rightarrow C/C++ \rightarrow Code Generation \rightarrow Runtime Library, select the Multithreaded (/MT) option for a release build, or Multi-threaded Debug (/MTd) if it is a debug build.

In addition, for a release build set 'Generate Debug Info' to No at Configuration Properties \rightarrow Linker \rightarrow Debugging.

Click **Apply** and **OK** to close the window.

Select either Debug or Release mode from the drop down box on the main Visual Studio Express toolbar and then select Build Solution (F7). The output screen should show that the build has succeeded. Connect the FTDI USB to serial device (with TXD connected to RXD) and hit F5 to run the application. The program will execute and the output screen in **Figure 2.12** will be displayed.



2.5 Building a solution for Windows XP with VS2013 Express

To build a solution which will run on Windows XP, then from the Property Pages -> Configuration Properties -> General, select the Platform Toolset as: Visual Studio 2012 – Windows XP (v110_xp) - see **Figure 2.16 – Platform toolset selection for Windows XP**

D2XX_Loopback_example Property Pa	iges 🤉 🤋 🛃
Configuration: Active(Debug)	Platform: Active(Win32) Configuration Manager
Common Properties Configuration Properties General Debugging VC++ Directories Cort+ Linker General Input Manifest File Debugging System Optimization Embedded IDL Windows Metadata Advanced All Options Command Line Manifest Tol SMarkest File Debugging System Optimization Embedded IDL Windows Metadata Advanced All Options Command Line Manifest Tol SMarkest File Debugging System Optimization Embedded IDL Windows Metadata Advanced All Options Command Line Manifest Tool SMarkest Coutom Build Step Code Analysis	• General Output Directory \$(SolutionDir)\$(Configuration)\ Intermediate Directory \$(Configuration)\ Target Name \$(ProjectName) Target Extension .exe Extensions to Delete on Clean *.cdf*.ccache*.obj*.il(e*.resources;*.tlb*.tlf*.tlf*.tlf*.tlf*.tlf*.tlf*.tlf*.tlf
	Platform Toolset Specifies the toolset used for building the current configuration; If not set, the default toolset is used
	OK Cancel Apply

Figure 2.16 – Platform toolset selection for Windows XP

Note also that the XP machine must have the MSVCR110.dll (Microsoft Visual C++ Redistributable) loaded – please refer to the following Microsoft link:

http://www.microsoft.com/en-us/download/details.aspx?id=30679#



3 Contact Information

Head Office - Glasgow, UK

Future Technology Devices International Limited Unit 1, 2 Seaward Place, Centurion Business Park Glasgow G41 1HH United Kingdom Tel: +44 (0) 141 429 2777 Fax: +44 (0) 141 429 2758

E-mail (Sales) sales1@ftdichip.com E-mail (Support) support1@ftdichip.com E-mail (General Enquiries) admin1@ftdichip.com

Branch Office - Taipei, Taiwan

Future Technology Devices International Limited (Taiwan) 2F, No. 516, Sec. 1, NeiHu Road Taipei 114 Taiwan, R.O.C. Tel: +886 (0) 2 8797 1330 Fax: +886 (0) 2 8751 9737

E-mail (Sales) E-mail (Support) E-mail (General Enquiries) tw.admin1@ftdichip.com

tw.sales1@ftdichip.com tw.support1@ftdichip.com

Branch Office – Tigard, Oregon, USA

Future Technology Devices International Limited (USA)7130 SW Fir Loop Tigard, OR 97223-8160 USA Tel: +1 (503) 547 0988 Fax: +1 (503) 547 0987

E-Mail (Sales) E-Mail (Support) E-Mail (General Enquiries)

us.sales@ftdichip.com us.support@ftdichip.com us.admin@ftdichip.com

Branch Office - Shanghai, China

Future Technology Devices International Limited (China) Room 1103, No. 666 West Huaihai Road, Shanghai, 200052 China Tel: +86 21 62351596 Fax: +86 21 62351595

E-mail (Sales) E-mail (Support) E-mail (General Enquiries) cn.sales@ftdichip.com cn.support@ftdichip.com cn.admin@ftdichip.com

13

Web Site

http://ftdichip.com

Distributor and Sales Representatives

Please visit the <u>Sales Network</u> page of the <u>FTDI Web site</u> for the contact details of our distributor(s) and sales representative(s) in your country

System and equipment manufacturers and designers are responsible to ensure that their systems, and any Future Technology Devices International Ltd (FTDI) devices incorporated in their systems, meet all applicable safety, regulatory and system-level performance requirements. All application-related information in this document (including application descriptions, suggested FTDI devices and other materials) is provided for reference only. While FTDI has taken care to assure it is accurate, this information is subject to customer confirmation, and FTDI disclaims all liability for system designs and for any applications assistance provided by FTDI. Use of FTDI devices in life support and/or safety applications is entirely at the user's risk, and the user agrees to defend, indemnify and hold harmless FTDI from any and all damages, claims, suits or expense resulting from such use. This document is subject to change without notice. No freedom to use patents or other intellectual property rights is implied by the publication of this document. Neither the whole nor any part of the information contained in, or the product described in this document, may be adapted or reproduced in any material or electronic form without the prior written consent of the copyright holder. Future Technology Devices International Ltd, Unit 1, 2 Seaward Place, Centurion Business Park, Glasgow G41 1HH, United Kingdom. Scotland Registered Company Number: SC136640



Appendix A – References

Document References

D2XX Programmers Guide

Latest FTDI Drivers

FTDI Cables

FTDI Development Modules

Acronyms and Abbreviations

Terms	Description
DLL	Dynamic-linked library
IC	Integrated Circuit
USB	Universal Serial Bus
VS	Visual Studio



Appendix B – List of Tables & Figures

List of Tables

NA

List of Figures

Figure 2.1 - The opening screen for VS Express 2015 for Windows Desktop	3
Figure 2.2 - New Project Window	4
Figure 2.3 – New Project Window 2	4
Figure 2.4 - Final setup window before coding screen	5
Figure 2.5 - Initial coding screen	5
Figure 2.6 - Insertion of D2XX loopback sample code	7
Figure 2.7 - Copy ftd2xx.h & ftd2xx.lib to the project folder	7
Figure 2.8 - Project property pages	8
Figure 2.9 - Select 'All Configurations'	8
Figure 2.10 - ftd2xx.lib is added to Additional Dependencies	9
Figure 2.11 - Output window after successful build	9
Figure 2.12 - Program output screen	10
Figure 2.13 – Selecting x64	10
Figure 2.14 – Output window after successful build	10
Figure 2.15 – Preprocessor definition for static build	. 11
Figure 2.16 – Platform toolset selection for Windows XP	. 12



Appendix C – Revision History

Document Title:	TN_153 Instructions on Including the D2XX Driver in a VS Express Project
Document Reference No.:	FT_000930
Clearance No.:	FTDI#530
Product Page:	http://www.ftdichip.com/FTProducts.htm
Document Feedback:	Send Feedback

Revision	Changes	Date
1.0	Initial Release	2017-06-08
1.1	Minor update to remove the 'Ignore Specific Default Libraries' field in Linker Input properties when building for static.	2018-06-06