



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: <http://ftdichip.com>

Copyright © Future Technology Devices International Limited

Table of Contents

1	Introduction	2
1.1	Software Required.....	2
1.2	Hardware Required	2
2	Creating the Project	3
2.1	Initial Setup	3
2.2	Building a Win32 Application which uses the FTD2XX.dll	7
2.3	Building an x64 application which uses the FTD2XX.dll.....	10
2.4	Building an application which uses the static library	11
2.5	Building a solution for Windows XP with VS2013 Express.	12
3	Contact Information	13
	Appendix A – References	14
	Document References	14
	Acronyms and Abbreviations.....	14
	Appendix B – List of Tables & Figures	15
	List of Tables.....	15
	List of Figures	15
	Appendix 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:
 - ftd2xx.lib for 32-bit systems in folder i386
 - ftd2xx.dll for 32-bit systems in folder i386
 - ftd2xx.lib for 64-bit systems in folder amd64
 - ftd2xx.dll for 64-bit systems in folder amd64
- Static library:
 - ftd2xx.lib for 32-bit systems in folder Static\i386
 - 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:
<http://www.ftdichip.com/Drivers/D2XX.htm>

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 [module](#) or [cable](#) 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.

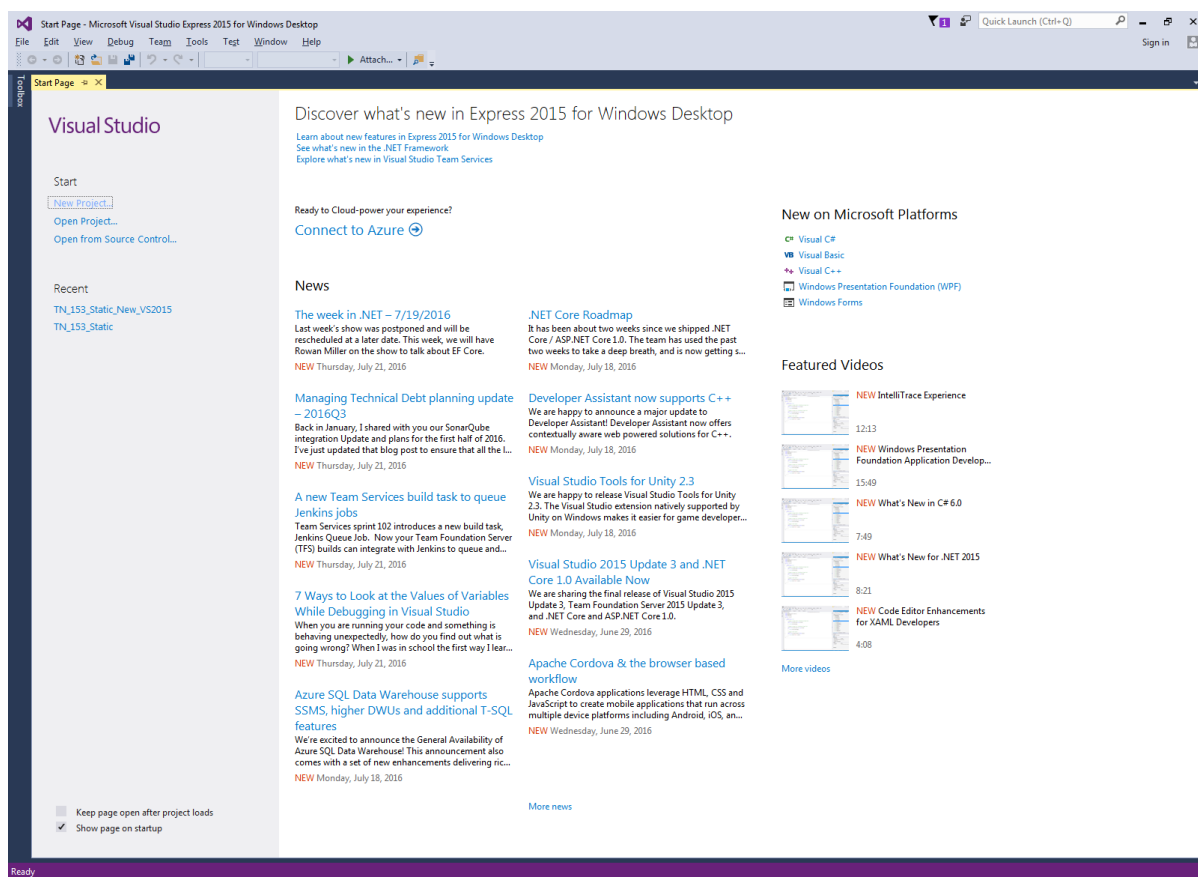


Figure 2.1 - The opening screen for VS Express 2015 for Windows Desktop

To create a new project from this screen, select File → New → 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.

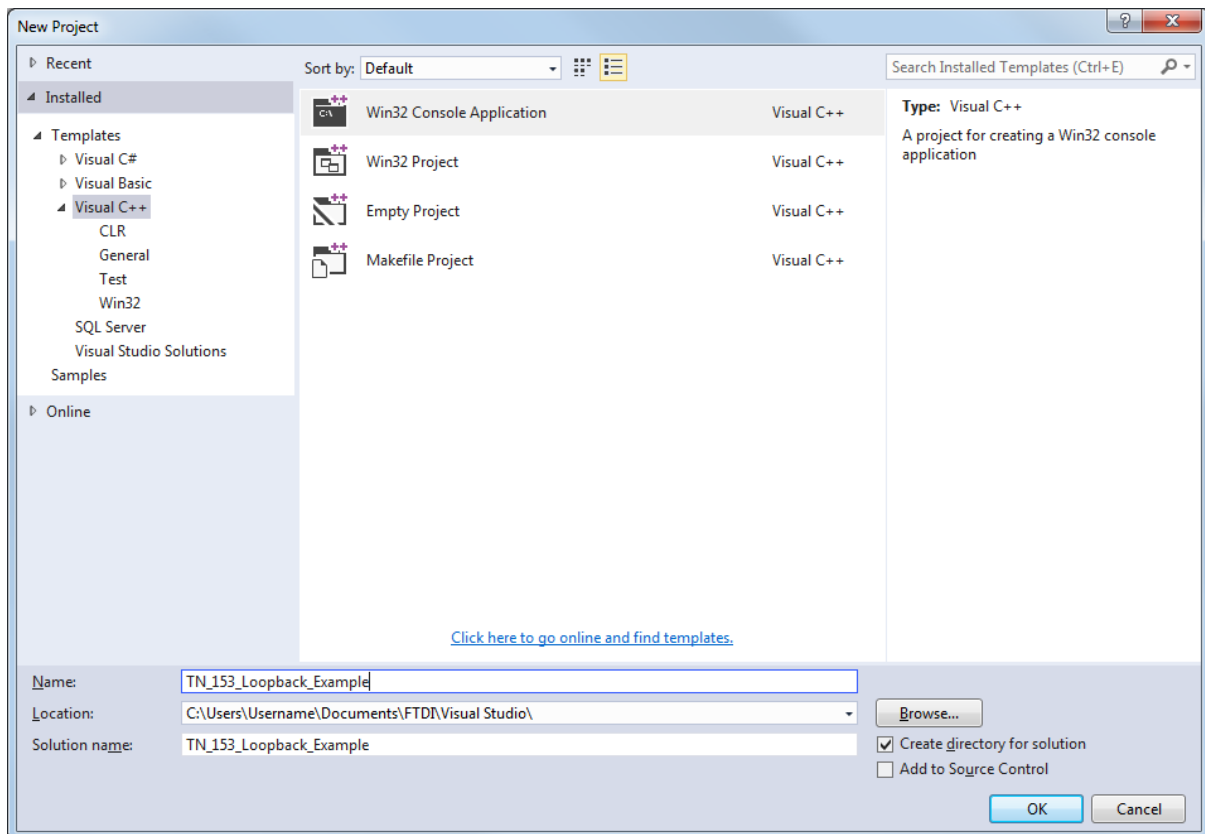


Figure 2.2 - New Project Window

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.

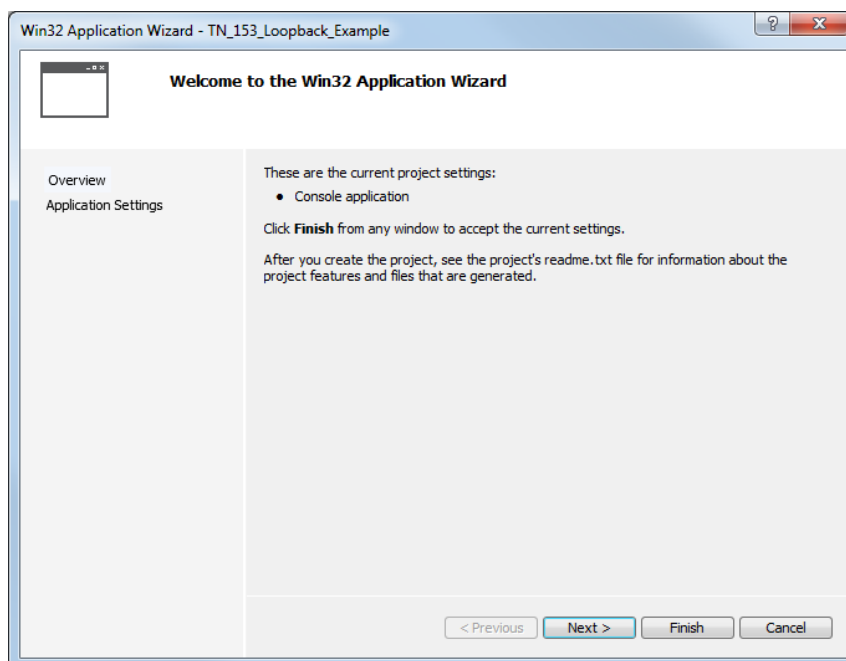


Figure 2.3 – New Project Window 2

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**.

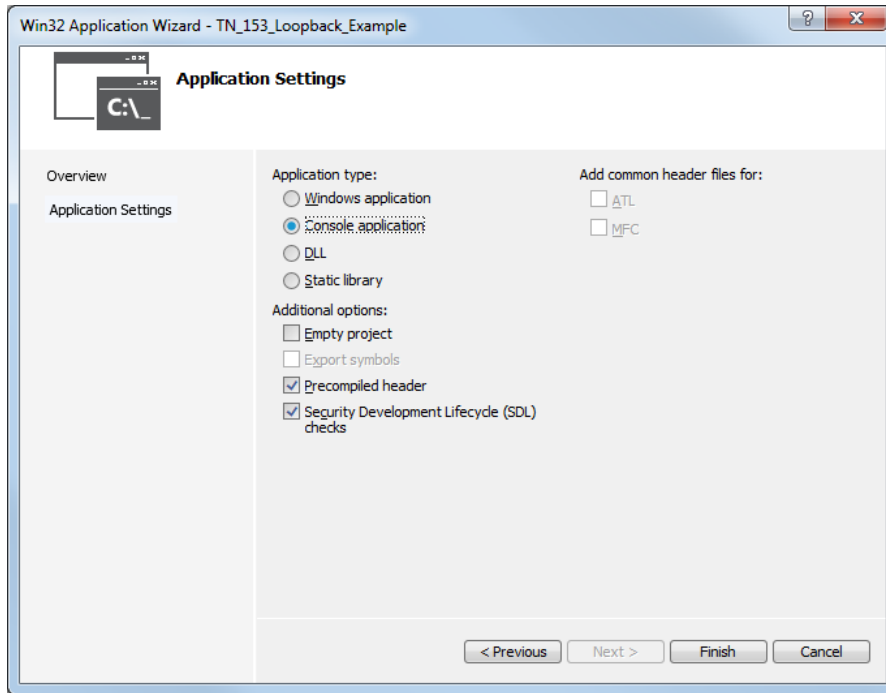


Figure 2.4 - Final setup window before coding screen

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

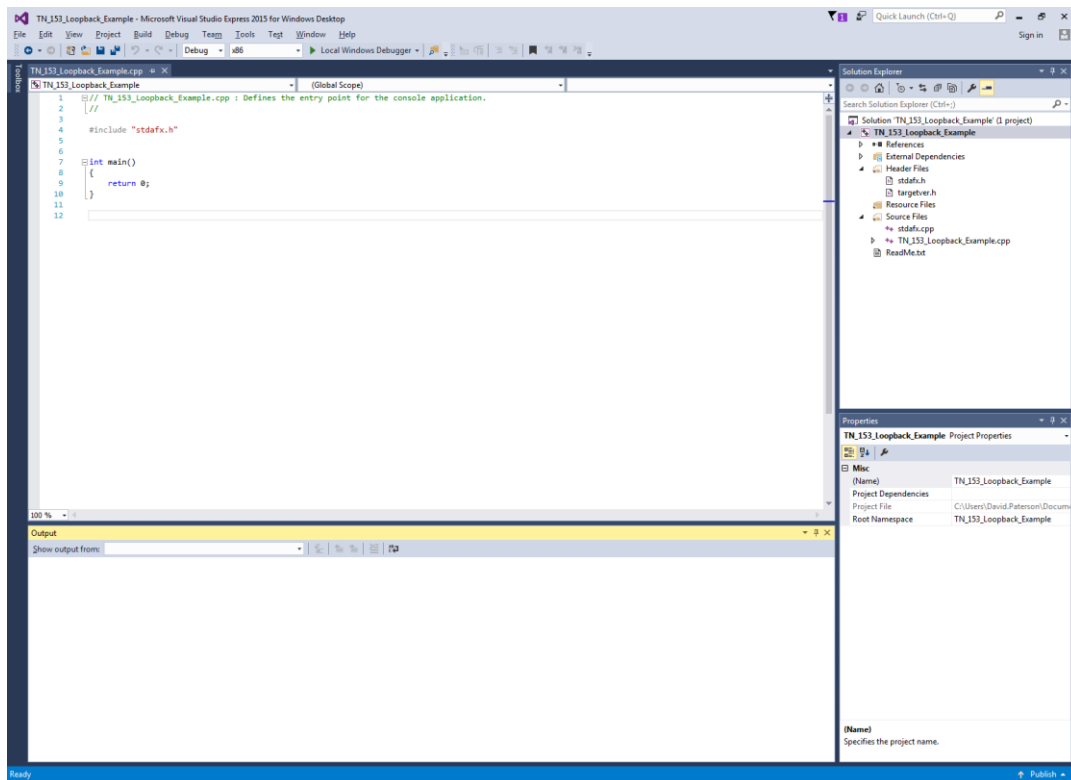
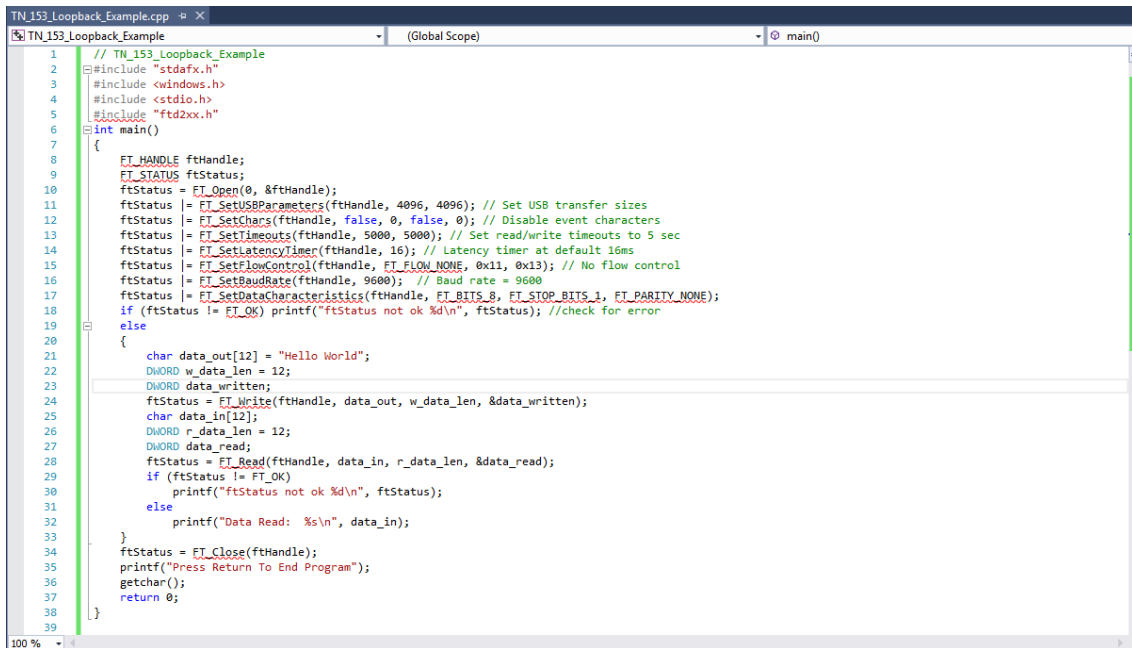


Figure 2.5 - Initial coding screen

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
    ftStatus |= FT_SetDataCharacteristics(ftHandle, FT_BITS_8, 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;
}
```



```

1 // TN_153_Loopback_Example
2 #include "stdafx.h"
3 #include <windows.h>
4 #include <stdio.h>
5 #include "ftd2xx.h"
6 int main()
7 {
8     FT_HANDLE ftHandle;
9     FT_STATUS ftStatus;
10    ftStatus = FT_Open(0, &ftHandle);
11    ftStatus = FT_SetUSBParameters(ftHandle, 4096, 4096); // Set USB transfer sizes
12    ftStatus = FT_SetChars(ftHandle, false, 0, false, 0); // Disable event characters
13    ftStatus = FT_SetTimeouts(ftHandle, 5000, 5000); // Set read/write timeouts to 5 sec
14    ftStatus = FT_SetLatencyTimer(ftHandle, 16); // Latency timer at default 16ms
15    ftStatus = FT_SetFlowControl(ftHandle, FT_FLOW_NONE, 0x11, 0x13); // No flow control
16    ftStatus = FT_SetBaudRate(ftHandle, 9600); // Baud rate = 9600
17    ftStatus = FT_SetDataCharacteristics(ftHandle, FT_BITS_8, FT_STOP_BITS_1, FT_PARITY_NONE);
18    if (ftStatus != FT_OK) printf("ftStatus not ok %d\n", ftStatus); //check for error
19    else
20    {
21        char data_out[12] = "Hello World";
22        DWORD w_data_len = 12;
23        DWORD data_written;
24        ftStatus = FT_Write(ftHandle, data_out, w_data_len, &data_written);
25        char data_in[12];
26        DWORD r_data_len = 12;
27        DWORD data_read;
28        ftStatus = FT_Read(ftHandle, data_in, r_data_len, &data_read);
29        if (ftStatus != FT_OK)
30            printf("ftStatus not ok %d\n", ftStatus);
31        else
32            printf("Data Read: %s\n", data_in);
33    }
34    ftStatus = FT_Close(ftHandle);
35    printf("Press Return To End Program");
36    getchar();
37    return 0;
38 }
  
```

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.

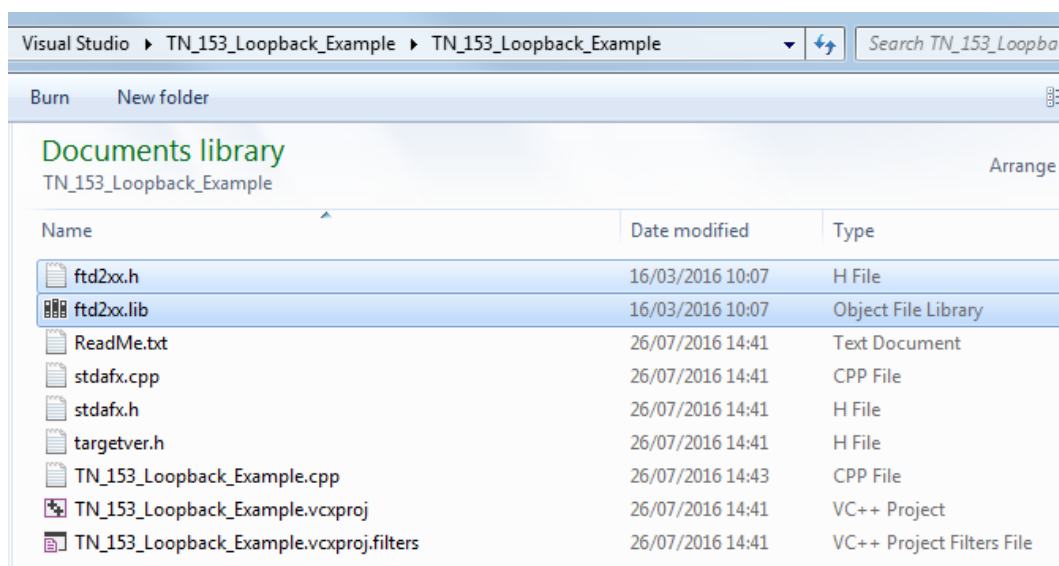


Figure 2.7 - Copy ftd2xx.h & ftd2xx.lib to the project folder

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

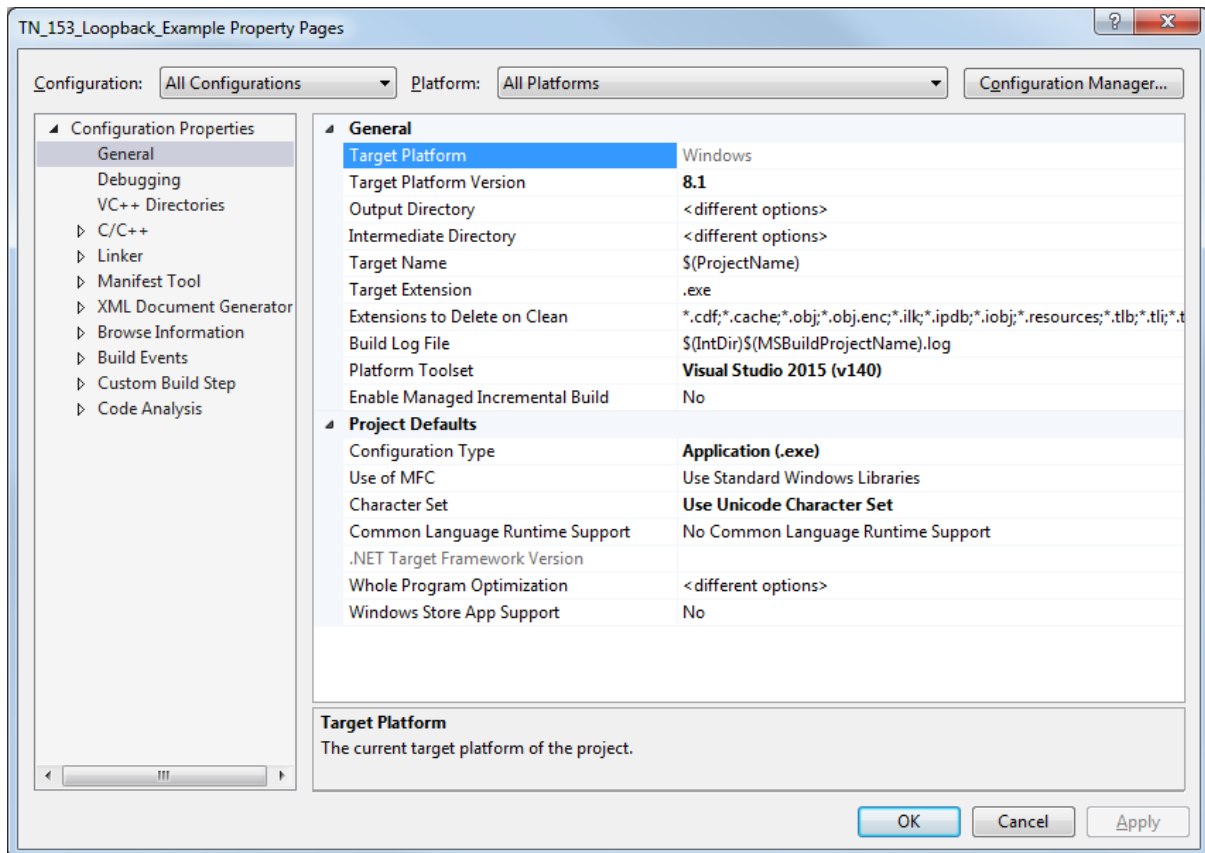


Figure 2.8 - Project property pages

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.

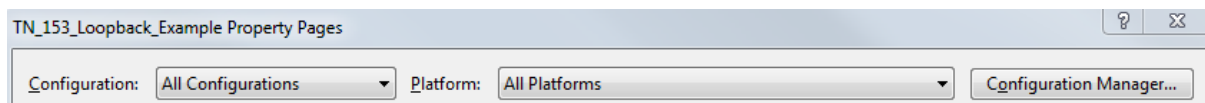


Figure 2.9 - Select 'All Configurations'

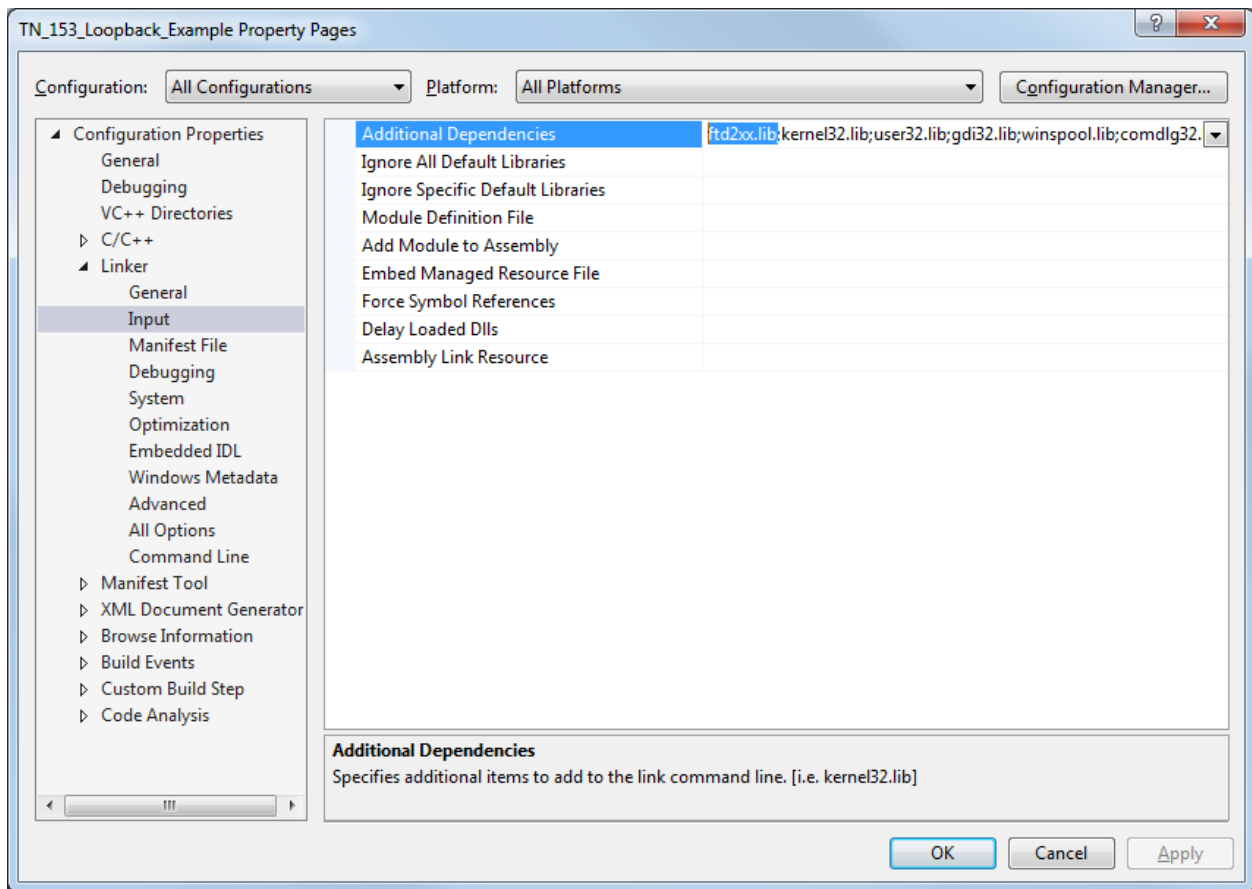


Figure 2.10 - ftd2xx.lib is added to Additional Dependencies

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.

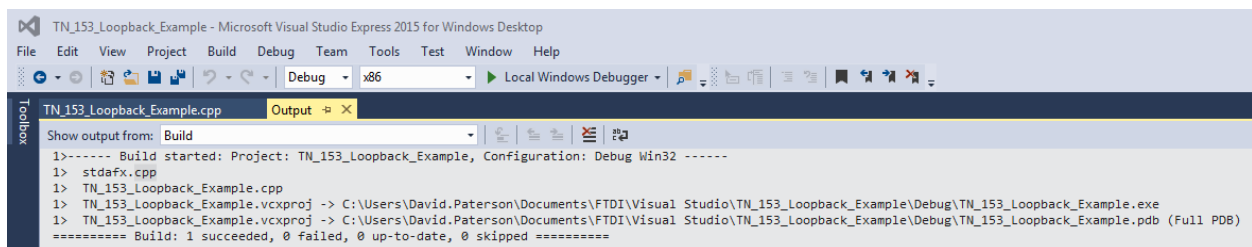


Figure 2.11 - Output window after successful build

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.

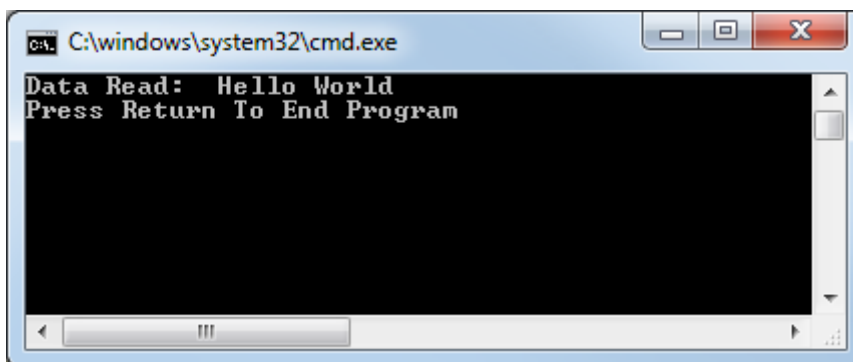


Figure 2.12 - Program output screen

2.3 Building an x64 application which uses the FTD2XX.dll

Select x64 as shown in **Figure 2.13** – Selecting x64

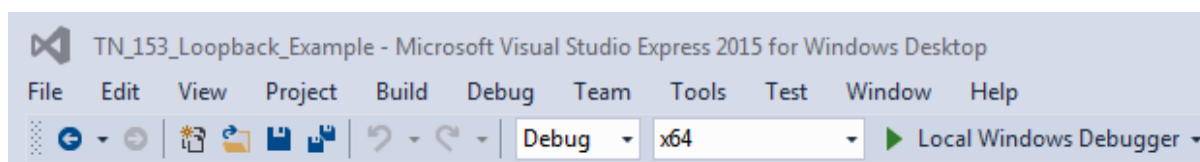


Figure 2.13 – Selecting x64

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

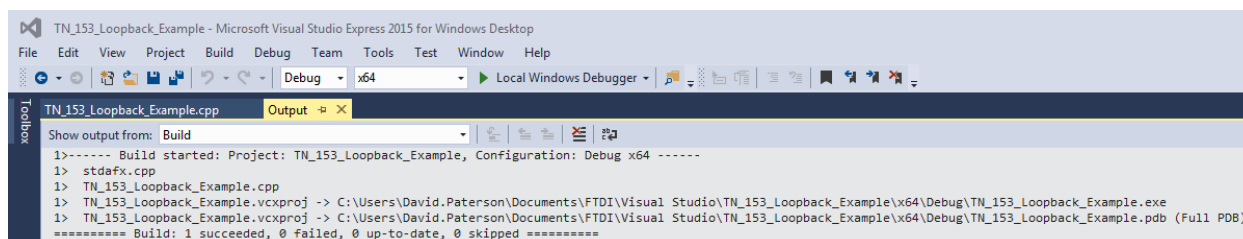


Figure 2.14 – Output window after successful build

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 → C/C++ → 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**

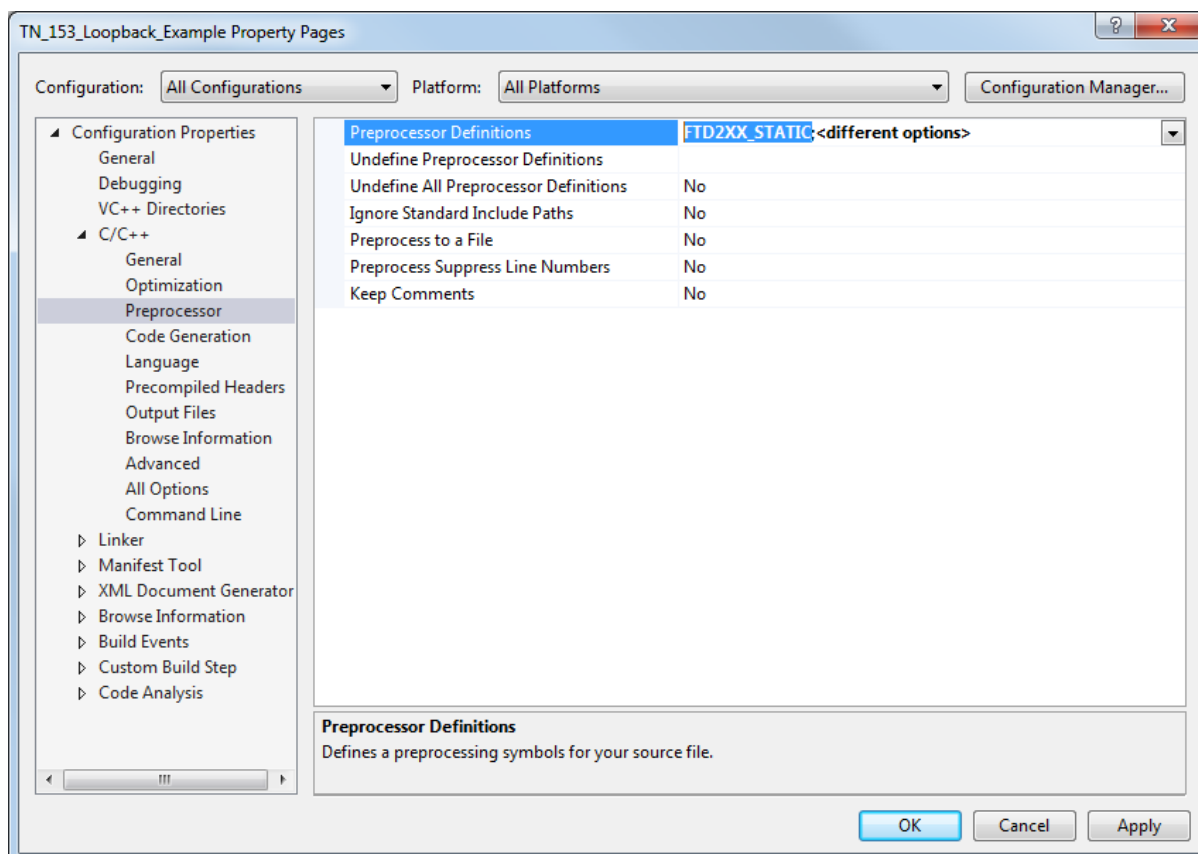


Figure 2.15 – Preprocessor definition for static build

In Configuration Properties → C/C++ → Code Generation → Runtime Library, select the Multi-threaded (/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 → Linker → 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**

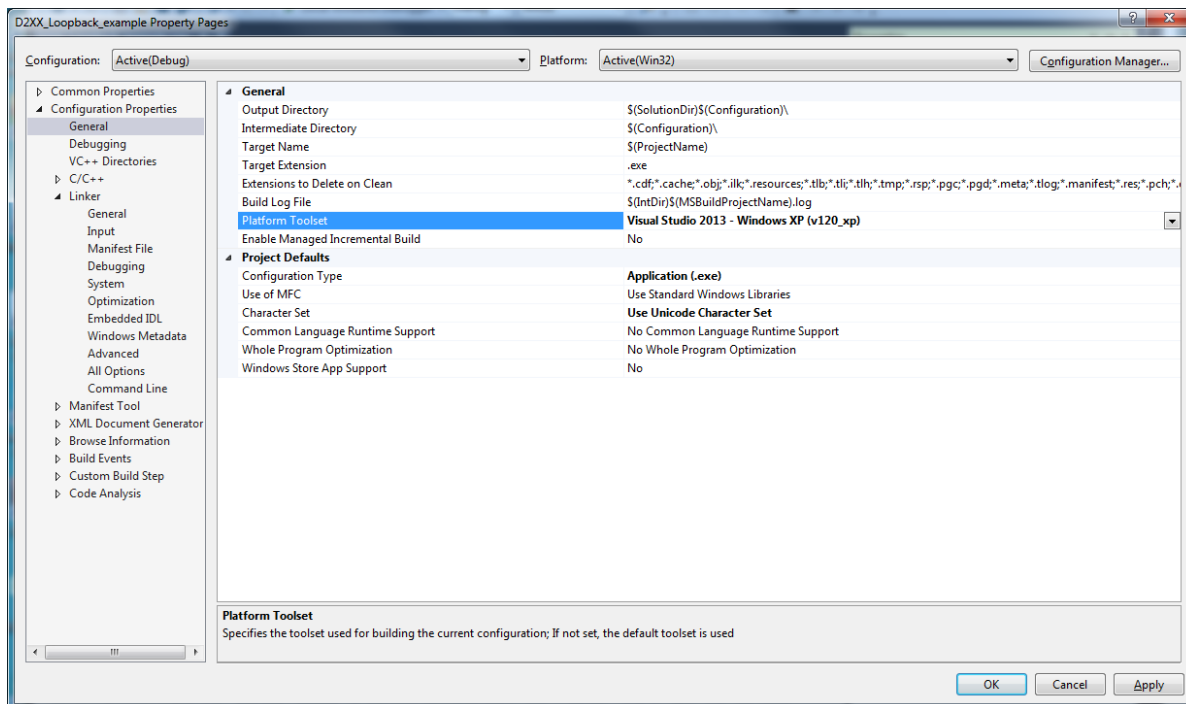


Figure 2.16 – Platform toolset selection for Windows XP

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 – 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) us.sales@ftdichip.com
E-Mail (Support) us.support@ftdichip.com
E-Mail (General Enquiries) us.admin@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) tw.sales1@ftdichip.com
E-mail (Support) tw.support1@ftdichip.com
E-mail (General Enquiries) tw.admin1@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) cn.sales@ftdichip.com
E-mail (Support) cn.support@ftdichip.com
E-mail (General Enquiries) cn.admin@ftdichip.com

Web Site

<http://ftdichip.com>

Distributor and Sales Representatives

Please visit the [Sales Network](#) page of the [FTDI Web site](#) 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