Application Report
SLLA170F–August 2004–Revised June 2011

USB/Serial Applications Using TUSB3410 VCP Software

Dan Harmon Connectivity Interface Solutions

ABSTRACT

TI's TUSB3410 device can be used to implement flexible full-speed USB/serial bridging applications, including USB/RS232, USB/RS485, USB/IrDA, and others. There are several decisions that need to be made by a designer using any of TI's TUSBxxxx family of USB controllers, and these are covered in the application report *VIDs, PIDs, and Firmware: Design Decisions When Using TI USB Device Controllers* (SLLA154). The USB/serial application is somewhat different than the general case because of the turnkey driver/firmware provided by TI. The steps required to convert TI's materials into a productized solution are summarized in this document.

1 Introduction

TI provides a driver/firmware solution for USB/serial applications using TUSB3410 and the VCP software. Although this solution is designed to be turnkey, there are a few steps that need to be taken by the designer to customize the solution. These steps are similar to those necessary for any TUSBxxxx application, which are covered in *VIDs, PIDs, and Firmware: Design Decisions When Using TI USB Device Controllers* (SLLA154), but differ somewhat as a result of the provided software. This document is intended to serve as a supplement to SLLA154 in USB/serial applications.

2 VCP Driver/Firmware Overview

TI’s Virtual COM Port (VCP) Driver/Firmware, in conjunction with the TUSB3410 device, provides a simple turnkey software solution for USB/serial bridge applications. Possibilities include:

- USB/RS232
- USB/RS485
- USB/UART (for interface to microcontroller)
- USB/IrDA

The initial version of the driver was for Windows platforms only (Windows 98, Windows 2000, and Windows XP).

The current Windows version supports 32-bit and 64-bit versions of Windows XP and Windows Vista. There is a BETA version for MAC OSX . The Windows and Mac versions are available under TI's no-cost license agreement. In addition, a Linux version is available under GNU General Public License for the following kernels:

- Linux kernel 2.4
- Linux kernel 2.6.

Most of the contents of this document apply to implementation with any operating system (OS), but some sections are dedicated to Windows only.

In addition to development for other OSs, TI is dedicated toward continual improvements on the Windows drivers.

NOTE: The Windows version of this software was formerly known as the UMP (USB MultiPort) driver/firmware, and this term can be found in the source code and earlier documentation. The function of this software has not changed, but the name has been changed to more clearly communicate its function. Also, please note that as of January 2008 Windows 98, Windows 2000, and Windows ME are no longer supported for this software.

3 Design Steps

When designing a TUSB3410-based USB/serial solution, these steps need to be taken:

1. Obtain vendor ID (VID) and decide on a product ID (PID).
2. Determine the firmware location (on PC host or in EEPROM on the device).
3. Decide whether to serialize the device, and if so, how.
4. Configure and program EEPROM.
5. Configure the driver’s INF file to match the VID/PID on the device (Windows implementations).
6. Configure the installer.

3.1 Determine VID/PID

For USB compliance, any USB product must have a unique vendor ID and product ID. The OS uses the VID/PID combination to determine what driver to load. The VID and PID are reported to the USB host in the USB device descriptor when this descriptor is requested by the host.

A unique VID can be obtained from the USB Implementers Forum (www.usb.org). A product ID can be whatever a vendor chooses, but since the VID/PID pair determines what driver is loaded, the same PID should not be given to two different products.

If you do not want to obtain your own VID, you have the option of requesting a unique PID from TI to be used with TI's VID (0x0451). Contact TI for more information at usb@ti.com.

3.2 Determine Firmware Location

Firmware in TUSBxxxx applications can either be stored on the USB host PC or in an EEPROM on the I2C port of the TUSB3410. Storing it on the host has the advantage of simplifying the process of field firmware updates, since all that is required for an update is replacing the firmware file on the host. Updating of firmware in the EEPROM requires use of a TI USB EEPROM Burner Utility for the TUSB3410.

Storing firmware in EEPROM in TUSB3410 applications has the advantage of automatically using the 3410's unique device ID as the USB serial number. If storing on the host, the only way to achieve a unique serial number is to serialize the EEPROM. See Section 3.4.

NOTE: TI does NOT recommend putting firmware in EEPROM when the device will receive its power from the USB. Although this scenario functions well in ordinary use, it can cause a problem with the USBCV application used during the USB certification and WHQL processes.

In VCP driver versions dated July 2004 or later, there are two firmware files. There is a version used for download and a version used for programming the EEPROM.

Table 1. Firmware Binary Filenames

	Firmware Download	Firmware in EEPROM
TUSB3410	umpf3410.i51	umpe3410.i51

The difference between umpfxxxx.i51 and umpexxxx.i51 is in the way they handle serial number reporting (see Section 3.5).

3.3 EEPROM Selection

EEPROM choice is largely determined by the outcome of the decisions in Section 3.2. If firmware is being placed in the EEPROM, a 128K-bit EEPROM should be enough for just about any application. The firmware is approximately 14K, leaving approximately 2K for header descriptor information

If firmware is being stored on the host, the EEPROM need only be big enough to store descriptor information. This usually depends on how long the string descriptors are. 1K bit is probably enough for most applications. If the strings are long, a 2K bit device may be required.

Note that due to an errata, TUSB3410 does not work with 16K bit EEPROMs.

3.4 Configure and Program EEPROM

To store the VID/PID, it is necessary to use an EEPROM on the I^{2}C bus. Without this, the device will report the default descriptors within TUSB3410, which will cause non-compliance with the USB specification and result in potential conflicts with other devices.

The VID/PID pair reported to the host is used to identify which driver to load. In Windows applications, the INF file performs this function, and therefore the designer must modify the file with the proper VID/PID (see Section 3.2).

TI provides a simple DOS utility called the TUSB3410 USB I^{2}C Header Generator Utility for VCP Applications, which can be downloaded from TI's website (SLLC297A). This utility inputs a text file that describes the EEPROM contents and outputs a binary file that can be downloaded to the EEPROM. The utility contains a README file with instructions, but they are summarized in the following sections.

3.4.1 Process the Configuration File with “Header Generator”

Follow the instructions in the Header Generator file to process your configuration file. If firmware is being placed in EEPROM, make sure the firmware file is in the same directory as “header.exe”.

3.4.2 Program the EEPROM

At this time, the recommended way to program the EEPROM is with a standalone EEPROM programmer.

There are service providers that can program EEPROMs in bulk for production purposes. If filters can be selected for the EEPROM programmer, note that the proper filter is “absolute binary”.

3.5 EEPROM Serialization

The OS uses the VID/PID combination to determine what type of device is attached and therefore what driver to load. But without a unique serial number, it cannot differentiate between different devices of the same type. This is a problem in USB/serial applications because it is desirable for the OS to keep a consistent association between a given device and a COM port number.

TUSB3410 applications have the option to serialize themselves so that the OS can differentiate them. If it is not acceptable for the OS to confuse the COM port assignment for a device in a multi-device system, serialization is recommended. If it is known that only one device will be attached to a host in most cases, the designer may choose not to serialize.

NOTE: A side effect of serialization is that Windows will see every instance of your device as a new device, and therefore it will present the "Found New Hardware" dialog box and guide the user through the installation sequence. Windows does this every time it sees a new combination of VID, PID, and serial number values.

Serialization can be implemented with one of two methods:

- Serialize the EEPROM. To do this, include a serial number string descriptor in the EEPROM; then, for each EEPROM created, increment the value of this string. This incrementing procedure is provided by EEPROM programming equipment.
- Store the firmware in EEPROM. The firmware is capable of extracting the serial number from the TUSB3410 device ID, a unique code in the chip. If firmware is in EEPROM, and if the serial number in EEPROM is the same as the default ("TUSB3410") this device ID value will be reported to the host. This does not occur if firmware is downloaded from the host, since descriptors are already reported before the firmware is downloaded. (Note that the string "TUSB3410 " must be identical to the default, which means there are exactly are eight trailing spaces. Since this string is already part of the firmware-in-EEPROM Header Generator utility script provided by TI, it should not be necessary to make any changes.)

Template scripts provided with the Header Generator utility simplify implementing either method.

3.6 Configure the INF

If Windows is the OS, the INF file (file of extension *.inf) is the mechanism used to associate devices with drivers. The information and driver file for Windows Vista/XP version of VCP driver would be:

- umpuart3410.inf
- umpusbvista.sys

After the VID/PID has been properly set in the device, the INF must be updated with the same VID/PID.

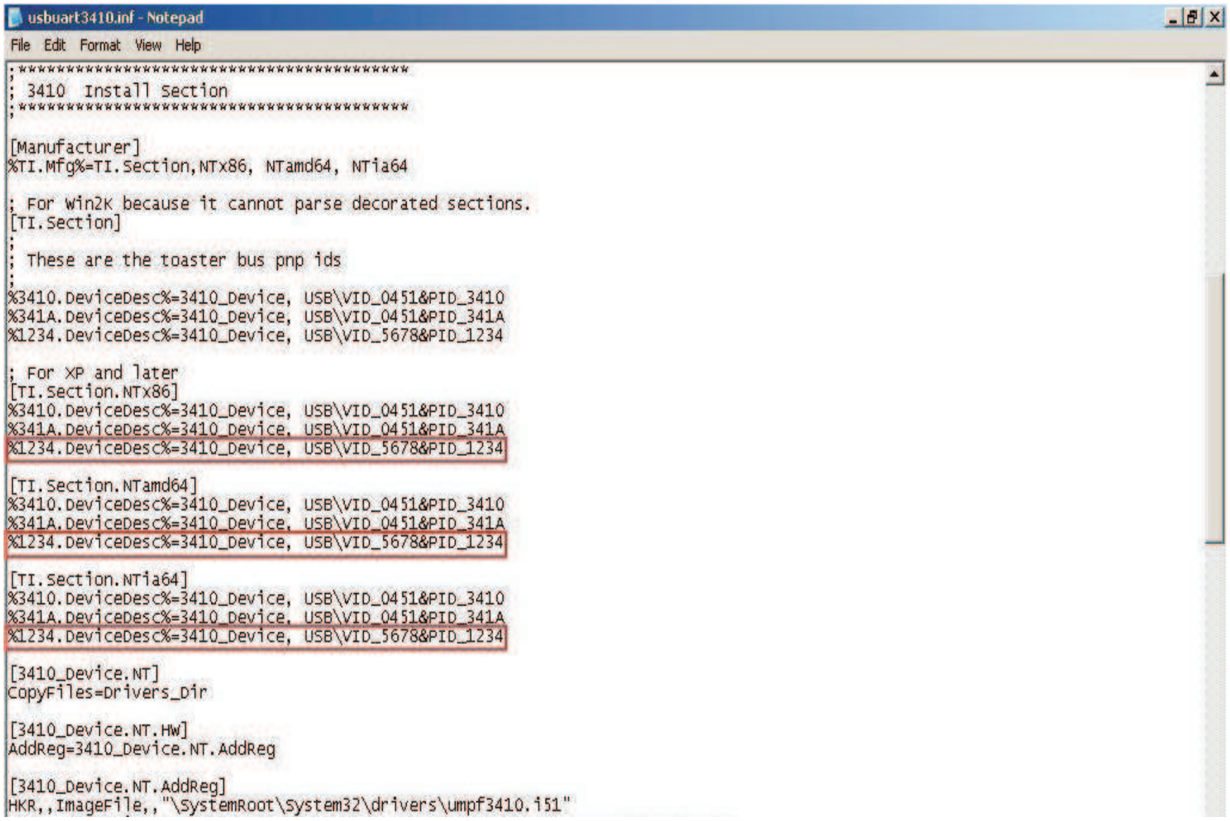
The INF is also used to store strings that will appear in Windows to describe the device to the user. The INF is a text file, and so it can be easily changed using any text editor, such as notepad.

TUSB3410 VCP driver has default TI VID and PID in the INF files, which have been certified by Microsoft and are digitally signed. When designing a USB device customers need an official USB vendor ID (VID) which is unique world-wide. Vendor IDs will be assigned by the USB Implementers Forum (www.usb.org). The Product ID can be chosen by the customers.

NOTE: When the vendor ID and Product IDs are changed in the INF files of the driver, Windows detects that it is not digitally signed and you may receive a warning message and the option to cancel or continue the installation. This message is displayed only if your computer is configured to display a warning message whenever an installation program tries to install a device driver without a digital signature. In Windows XP and Windows Vista 32-bit version this is just a warning and the installer will still be able to install the driver when chosen to continue the installation. However, in case of Windows Vista 64-bit version when the operating system detects that the VID/PID has changed, it invalidates the digital signature and the installer will not be able to install the unsigned driver. Customers will have to submit their custom vendor ID and product ID to Microsoft for certification and get the digital signature for their product.

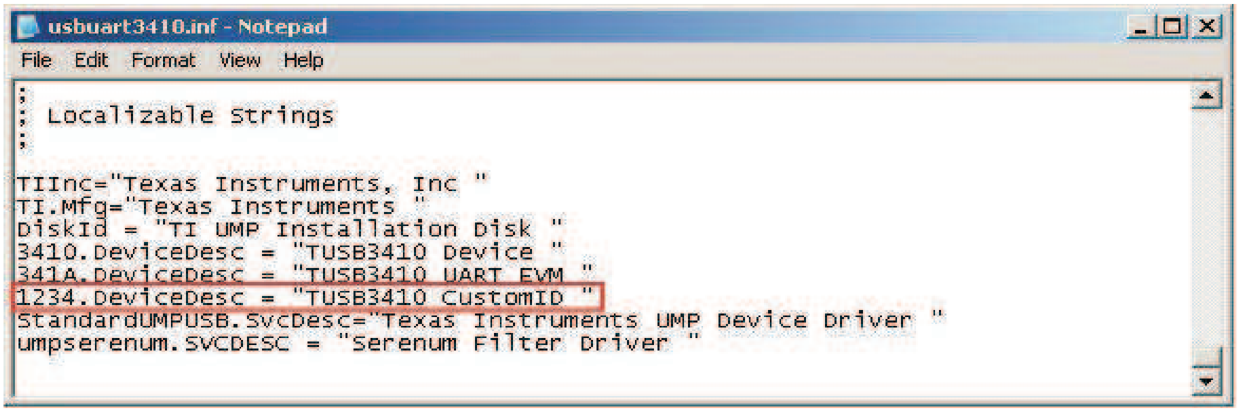
If customers choose to test the VCP driver on Windows Vista 64-bit version before getting it digitally signed for their VID/PID through Microsoft, then they can do so by booting their OS in safe mode and disabling the digital signature during boot up. Although customization of the INF generally requires knowledge of the Windows Driver Model, TI has simplified this process for the VCP application by giving specific instructions within the INF files. All you need are your new VID and PID, as well as new strings. In the INF, lines to be modified are marked in the red border in Figure 1 and Figure 2.

In Figure 1 and Figure 2, PID = 1234 and VID=5678 represent custom VID/PID that customers can modify to their own VID/PID.



```
;*******************************************
; 3410  Install Section
;*******************************************

[Manufacturer]
%TI.Mfg%=TI.Section,NTx86, NTamd64, NTia64

; For Win2K because it cannot parse decorated sections.
[TI.Section]
;
; These are the toaster bus pnp ids
;
%3410.DeviceDesc%=3410_Device, USB\VID_0451&PID_3410
%341A.DeviceDesc%=3410_Device, USB\VID_0451&PID_341A
%1234.DeviceDesc%=3410_Device, USB\VID_5678&PID_1234

; For XP and later
[TI.Section.NTx86]
%3410.DeviceDesc%=3410_Device, USB\VID_0451&PID_3410
%341A.DeviceDesc%=3410_Device, USB\VID_0451&PID_341A
%1234.DeviceDesc%=3410_Device, USB\VID_5678&PID_1234

[TI.Section.NTamd64]
%3410.DeviceDesc%=3410_Device, USB\VID_0451&PID_3410
%341A.DeviceDesc%=3410_Device, USB\VID_0451&PID_341A
%1234.DeviceDesc%=3410_Device, USB\VID_5678&PID_1234

[TI.Section.NTia64]
%3410.DeviceDesc%=3410_Device, USB\VID_0451&PID_3410
%341A.DeviceDesc%=3410_Device, USB\VID_0451&PID_341A
%1234.DeviceDesc%=3410_Device, USB\VID_5678&PID_1234

[3410_Device.NT]
CopyFiles=Drivers_Dir

[3410_Device.NT.HW]
AddReg=3410_Device.NT.AddReg

[3410_Device.NT.AddReg]
HKR,,ImageFile,,"\SystemRoot\System32\drivers\umpf3410.i51"
```

Figure 1. usbuart3410.inf Modification for Custom VID/PID



```
;
; Localizable Strings
;

TIInc="Texas Instruments, Inc "
TI.Mfg="Texas Instruments "
DiskId = "TI UMP Installation Disk "
3410.DeviceDesc = "TUSB3410 Device "
341A.DeviceDesc = "TUSB3410 UART EVM "
1234.DeviceDesc = "TUSB3410 CustomID "
StandardUMPUSB.SvcDesc="Texas Instruments UMP Device Driver "
umpserenum.SVCDESC = "Serenum Filter Driver "
```

Figure 2. usbuart3410.inf Modification for Custom Strings and ID

NOTE: The VID/PID in the INF file has to match the VID/PID on the EEPROM. In order to ensure this, customers will have to change the VID/PID in the .cfg file available in the header generator utility and generate the corresponding .bin file that should be burned on the EEPROM.

3.7 Installer Customization

In January 2005, TI released a new installer created with the InstallShield environment, in conjunction with co-installers written in Microsoft Visual C. This installer minimizes effort on the part of the user, reducing the need for user intervention.

The driver files are left visible so that the INF files can be modified per Section 3.6. The installer itself is designed to be used without the need for customization. However, TI will provide the Installshield X project, and the source for the co-installers, upon request.

3.7.1 Using Multiple TUSB3410 Devices on Single PC

If customers wish to enable the end-user to use two or more of the same TUSB3410-based end products (same PID) on a single PC, then the end product must include serialization in the USB Descriptor. When multiple TUSB3410-based end products with the same PID are serialized the operating system (OS) can differentiate them and assign different COM ports. If the end products are not serialized, it can confuse the OS and it may uninstall the driver when the second device is connected to the PC. For more details on how TUSB3410-based devices can be serialized please read section 3.5.

4 No-EEPROM Implementations

It is possible to design a TUSB3410-based device that does not use an EEPROM. This solution reports the default descriptors located in the bootcode, including the default VID/PID, and firmware is downloaded from the host. However, doing so has two consequences:

- The solution is not USB-compliant because it does not have the vendor’s unique VID.
- A host that sees two devices with the same VID/PID and same serial number may not allow both of them to function.

For this reason, TI strongly discourages this configuration, unless the system is known to be an isolated bus that will not encounter any unknown devices that may conflict with your device.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Audio	www.ti.com/audio	Communications and Telecom	www.ti.com/communications
Amplifiers	amplifier.ti.com	Computers and Peripherals	www.ti.com/computers
Data Converters	dataconverter.ti.com	Consumer Electronics	www.ti.com/consumer-apps
DLP® Products	www.dlp.com	Energy and Lighting	www.ti.com/energy
DSP	dsp.ti.com	Industrial	www.ti.com/industrial
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Security	www.ti.com/security
Logic	logic.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Power Mgmt	power.ti.com	Transportation and Automotive	www.ti.com/automotive
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com	Wireless	www.ti.com/wireless-apps
RF/IF and ZigBee® Solutions	www.ti.com/lprf		
		TI E2E Community Home Page	e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265