



Protouch2 Command Line Interface User Manual


<div>  <div> MICROCHIP Microchip Technology, Inc. </div> </div> <div> Microchip Technology, Incorporated 2355 W. Chandler Boulevard Chandler, Arizona 85224 480/792-7416 </div>			
REV	DATE	ORIGINATOR	DESCRIPTION OF CHANGE
0.2	9/2/2014	Vishnu P	Initial version
0.4	10/14/2014	Vishnu P	Changes from version 0.2
0.9	12/23/2014	Vishnu P	Changes from version 0.3 to 0.9
1.1	9/3/2015	Vishnu P	Changes from version 0.9 to 1.1
1.2	21/4/2015	Riyas K	Changes from version 1.1 to 1.2
1.4	12/05/2015	Riyas K	Changes from version 1.2 to 1.4
1.5	04/08/2015	Vishnu P	Changes from version 1.4 to 1.5
1.51	09/04/2015	Vishnu P	Changes from version 1.5 to 1.51
1.54	09/25/2015	PrasannaV	Changes from version 1.51 to 1.54
1.55	10/01/2015	Karpagam A	Changes from version 1.54 to 1.55
1.6	3/30/2016	Prasanna V	Changes from version 1.55 to 1.6
1.7	5/30/2016	Prasanna V	Changes from version 1.6 to 1.7
1.8	8/18/2016	Vignesh N	Changes from version 1.7 to 1.8

Table of Contents

1	Introduction.....	6
1.1	Abbreviations	6
2	Legal Information	6
3	INI File Description	7
3.1	HUB_VID_LIST section (Hub Vendor ID)	7
3.2	HUB_RESET_DELAY Section:	7
3.2.1	RESTART_DELAY	7
3.3	HCE_DEV_INFO Section (Hub Controller Information)	7
4	Logging.....	9
5	Version.....	9
6	Help.....	9
7	Known Limitations	9
8	Programming time	10
8.1	USB Devices	10
8.2	LAN78xx Devices.....	10
9	USB Devices.....	11
9.1.1	Prerequisites.....	11
9.2	Programming SPI Flash firmware.....	11
9.2.1	Hub filter is installed as hub class filter.....	11
9.2.2	Device specific hub filter approach	12
9.2.3	Time optimised Production line programming method	13
9.2.4	Erasing SPI Flash.....	14
9.3	Single Device OTP Programming.....	15
9.3.1	OTP Programming.....	15
9.3.2	Program OTP and Lock OTP – USB57x4 Family.....	15
9.3.3	Program OTP and USB serial string.....	15
9.3.4	Program OTP and Verify Programmed Configuration Item.....	16
9.4	Batch Programming (Automated Execution).....	17
9.4.1	Mass OTP Programming (Single MCHP device is connected).....	17
9.4.2	Mass OTP Programming (Multiple MCHP devices are connected).....	18
9.4.3	Mass program OTP and USB serial string.....	18
9.4.4	Mass program OTP and Verify Programmed Configuration Item	20
9.5	Verification of Hub Configuration Items	21
9.5.1	Supported Parameters	21
9.5.1.1	vid	21
9.5.1.2	pid	21
9.5.1.3	did	21
9.5.1.4	usbvcd.....	21
9.5.1.5	languageid.....	21
9.5.1.6	manufacturer	22
9.5.1.7	product	22
9.5.1.8	serial.....	22
9.5.1.9	usb3vid.....	22

9.5.1.10	usb3pid.....	22
9.5.1.11	usb3did.....	22
9.5.1.12	usb3vcd.....	22
9.5.1.13	usb3languageid	22
9.5.1.14	usb3manufacturer	22
9.5.1.15	usb3product	22
9.5.1.16	usb3serial	22
9.5.2	Get value of parameter before and after programming.....	22
9.5.3	Get value of any parameter	23
9.5.4	Compare value of a parameter with known value	23
9.6	Dump Memory	24
9.6.1	OTP Memory	24
9.6.1.1	Hub class filter approach,	24
9.6.1.2	Device specific filter approach,	24
9.6.2	SPI Memory	24
9.6.2.1	SPI Flash Firmware dump	24
9.6.2.2	SPI flash firmware and configuration dump.....	24
9.7	Changing Vendor ID/Product ID of the Hub	25
9.8	Changing Vendor ID/Product ID of the Hub Controller.....	25
10	LAN78XX Devices.....	26
10.1	Single Device EEPROM Programming	26
10.1.1	EEPROM Programming	26
10.1.2	Program EEPROM and USB Serial/MAC address	26
10.1.2.1	Program EEPROM with serial number (Override Serial number)	26
10.1.2.2	Program EEPROM with MAC address (Override Mac address) ..	26
10.1.2.3	Program EEPROM with MAC address (Override Mac address) and serial number (Override serial number).....	27
10.1.3	Program EEPROM and Verify Programmed Configuration Item	27
10.2	Single Device OTP Programming	28
10.2.1	OTP Programming	28
10.2.2	Program OTP and USB Serial/MAC address	28
10.2.2.1	Program OTP with serial number (Override Serial number)	28
10.2.2.2	Program OTP with MAC address (Override Mac address).....	28
10.2.2.3	Program OTP with MAC address (Override Mac address) and serial number (Override serial number).....	29
10.2.3	Program OTP and Verify Programmed Configuration Item.....	29
10.3	Batch Programming (Automated Execution)	30
10.3.1	Mass EEPROM/OTP Programming (Single device is connected)	30
10.3.2	Mass program EEPROM/OTP and USB serial string	30
10.3.3	Mass program EEPROM/OTP and Mac Address.....	31
10.3.4	Mass program EEPROM/OTP with Mac Address and serial number..	32
10.3.5	Mass program OTP and Verify Programmed Configuration Item	33
10.4	Verification of LAN Configuration Items	34
10.4.1	Get value of parameter before and after programming.....	38
10.4.2	Get value of parameter.....	39
10.4.3	Compare value of a parameter with known value	39

10.5	Dump Memory	40
10.5.1	OTP Memory	40
10.5.2	EEPROM Memory.....	40
10.6	Changing Vendor ID/Product ID of the LAN78xx	40
11	Appendix I - Serial Number Suppression	41
11.1	Why is it required?	41
11.2	When is it needed?.....	41
11.3	How to execute SerialNumSuppression.bat file?	41
12	Appendix II - Changing filename extension	42
12.1	To show or hide file name extensions	42
13	Appendix III – Find Hub Index or path - USB Hubs.....	44
13.1	Index Method.....	44
13.1.1	Usage of index in “/id” command.....	44
13.1.2	To List the hubs or to find the index of a hub.....	45
13.2	Port Chain Method.....	46
13.2.1	Usage of port chain in “/devpath” command.....	46
13.2.2	To find the port chain of a hub.....	48
14	Appendix IV.....	49
14.1	Troubleshooting.....	49
14.2	Error codes.....	49
15	Appendix V	51
15.1	Internal HFC device enabled by Default	51
15.1.1	SKUS	51
15.1.2	HFC Driver Installation	52
15.1.2.1	Automatic HFC Driver installation	52
15.1.2.2	Manual HFC Driver installation	53
15.1.3	HFC Driver Uninstallation.....	53
15.2	Internal HFC device disabled by Default	54
15.2.1	SKUS	54
15.2.2	Hub class filter driver.....	54
15.2.2.1	Hub class filter driver installation	54
15.2.2.2	Hub class filter driver uninstallation	55
15.2.2.3	Advantage.....	55
15.2.2.4	Disadvantage	55
15.2.3	Hub device specific filter driver	56
15.2.3.1	Advantage.....	56
15.2.3.2	Disadvantage	56
16	Appendix VI - LAN78xx Driver Installation.....	57
17	Appendix VII – Find index of LAN device	58
17.1	Usage of index /id for LAN commands.....	58
17.2	To List the LAN78xx devices or to find the index of LAN78xx	58

1 Introduction

Protouch2 CLI application is a programming tool for USB253x/USB4604, USB57X4, USB58XX, USB59XX, LAN78XX and other families of Microchip USB products.

1.1 Abbreviations

HFC – Hub Feature controller (Internal USB Device) – USB device that appears on the (n+1th) (nth port of the Hub where n is the no of downstream physical ports in the hub

PT2 – Protouch2 application

2 Legal Information

Software License Agreement

(c) 2004 - 2016 Microchip Technology Inc.

Microchip licenses this software to you solely for use with Microchip products. The software is owned by Microchip and its

licensors, and is protected under applicable copyright laws. All rights reserved.

SOFTWARE IS PROVIDED "AS IS" MICROCHIP EXPRESSLY DISCLAIMS ANY WARRANTY OF ANY KIND, WHETHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL MICROCHIP BE LIABLE FOR ANY INCIDENTAL, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, HARM TO YOUR EQUIPMENT, COST OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), ANY CLAIMS FOR INDEMNITY OR CONTRIBUTION, OR OTHER SIMILAR COSTS.

To the fullest extent allowed by law, Microchip and its licensors liability shall not exceed the amount of fees, if any, that you have paid directly to Microchip to use this software.

MICROCHIP PROVIDES THIS SOFTWARE CONDITIONALLY UPON YOUR ACCEPTANCE OF THESE TERMS.

Trademark Information

The Microchip name and logo, the Microchip logo, MPLAB, and PIC are registered trademarks of Microchip Technology

Incorporated in the U.S.A. and other countries.

PICDEM and PICtail are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Microsoft, Windows, Windows Vista, and Authenticode are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

SD is a trademark of the SD Association in the U.S.A and other countries

3 INI File Description

This section is only applicable for USB hub products.

3.1 HUB_VID_LIST section (Hub Vendor ID)

The application will populate list of hubs connected to the computer with unique index numbers. By default, microchip hubs will be moved to lower index's based on the Vendor ID (VID) during the initial process of application. To move other hubs to lower index position when listing the hubs, Vendor ID needs to be specified in HUB_VID_LIST section of INI file.

INI File Name: Protouch2.ini

Example:

```
[HUB_VID_LIST]
; Microchip VID
HUB_VID1=0x0424
; Other VID's Section
; Add as HUB_VID'n+1' = VID in "0x" format
; "HUB_VID2=0x8085"
; Maximum five VID's can be added here.
; So the maximum is HUB_VID5= 0xFFFF;
```

3.2 HUB_RESET_DELAY Section:

3.2.1 RESTART_DELAY

RESTART_DELAY is the time delay for the device enumeration once the device is restarted. Device will be restarted in the application with RESTART_DELAY timeout for specific commands. Application will wait for the device enumeration up to value specified in the RESTART_DELAY variable.

Example:

RESTART_DELAY = 10000; In Milliseconds

3.3 HCE_DEV_INFO Section (Hub Controller Information)

HCE is nothing but the internal HFC device. The internal HFC device is basically a winusb class USB device. By default, the tool will support with microchip default Vendor ID and Product ID of the HFC device (Not the Vendor ID and Product ID of HUB).

If the Vendor ID and Product ID (PID) of HFC Device need to be changed, then add HFC Product ID to the HCE_DEV_INFO section in the INI file.

To get the winusb handle from the HFC, PID needs to be added to the following list. Vendor ID of the HFC will be taken from "HUB_VID_LIST" section (Refer [section 3.1](#)). Following listed Product ID's are default one and if any of the HFC device has different PID's except below listed values, those need to be added into the list.

Example:

```
[HCE_DEV_INFO]
```

```
HCE_PID1 = 0x2530;
```

```
HCE_PID2 = 0x2740;
```

```
HCE_PID3 = 0x274E;
```

```
HCE_PID4 = 0x274F;
```

```
; VID of the HFC will be taken from "HUB_VID_LIST" section
```

Maximum entry is 10. (Max is HFC_PID10 = 0x1234 and HFC_PID11 = 0x1234 is not valid one)

4 Logging

A detailed log file named “PT2.log” is automatically created in the same path as where the application is running.

Logging levels can be selected using following commands,

/s0 (or) /s – Suppress the command window and no log messages will be updated in “PT2.log” file for that operation

/s1 – Suppress the command window and short description for operation performed will be updated in log file

/s2 – Suppress the command window and detailed description for operation performed will be updated in log file

By default, detailed description will be logged if any one of these options was not given.

5 Version

The version number of the tool can be found using the following command;

>pt2main.exe /version

6 Help

Help regarding the command line arguments can be obtained by using the following command;

For USB and LAN help

>pt2main.exe /help or >pt2main.exe /?

For USB help

>pt2main.exe /hu

For LAN help

>pt2main.exe /hl

7 Known Limitations

Please refer to the release notes for more information on supported operating systems, SKUs supported, USB controllers supported and known limitations.

8 Programming time

8.1 *USB Devices*

If the internal HFC device is enabled by default, depending on the OS (Win 7 or 8.1) and architecture (32 or 64 bit), it takes about 0.5 to 2 seconds for programming configuration file and 4 to 8 seconds for programming firmware file.

If the internal HFC device is disabled by default, depending on the OS (Win 7 or 8.1) and architecture (32 or 64 bit), it takes about 5 to 15 seconds for programming configuration file and 8 to 18 seconds for programming firmware file.

8.2 *LAN78xx Devices*

To program 256 bytes in LAN78xx devices EEPROM, it takes about 2 to 3 seconds with EHCI controller and takes about 7 to 8 seconds with xHCI controller.

To program 256 bytes in LAN78xx devices OTP, it takes about 2 to 3 seconds with EHCI controller and takes about 7 to 8 seconds with xHCI controller.

9 USB Devices

9.1.1 Prerequisites

1. Refer [Appendix V](#) to install HFC driver. One time installation is required per system. This step can be skipped if the HFC driver is already installed.
2. Hub filter driver installation is required if the internal hub feature controller is disabled by default in the hub. This step can be skipped either if the hub feature controller is enabled by default or hub class filter driver is already installed. Refer [Section 15.2](#) for more details.
3. Find out the hub index by executing the command “*pt2main.exe /l*”. Refer [Section 13.1](#) for more details.

As an option, port chain method also can be used to identify a device. Execute the command “*pt2main.exe /lp*” to get port chain of the hub. Refer [Section 13.2](#) for more details.

9.2 Programming SPI Flash firmware

9.2.1 Hub filter is installed as hub class filter

1. One of the following methods can be used for SPI Flash firmware programming if either HFC is enabled by default or hub filter is installed as hub class filter.

- **Program SPI flash firmware; Do not erase existing Pseudo-OTP configurations**

```
>pt2main.exe /pspi <firmware_filename.bin> /id <index>
```

- /id <index> is the index of the hub to be programmed.

- **Program SPI flash firmware; Erase existing Pseudo-OTP configurations**

```
>pt2main.exe /pspi <firmware_filename.bin> /id <index> /e
```

- /e is the option to erase existing Pseudo-OTP configurations.

- **Program SPI flash firmware with new configuration file; Do not erase existing Pseudo-OTP configurations**

```
>pt2main.exe /pspi <firmware_filename.bin> /p <config_file.cfg> /id <index>
```

- /p <config_file.cfg> is the configuration file to be programmed in to Pseudo OTP.
- /id <index> is the index of the hub to be programmed.
- **Program SPI flash firmware with new configuration file ; Erase existing Pseudo-OTP configurations**

```
>pt2main.exe /pspi <firmware_filename.bin> /p <config_file.cfg> /id <index> /e
```

- /e is the option to erase existing Pseudo-OTP configurations.
- **Program SPI flash firmware with new configuration file; Program USB serial string**

```
>pt2main.exe /pspi <firmware_filename.bin> /p <config_file.cfg> /pser <serial_string> /id <index>
```

- /pser <serial_string> is the USB2.0/USB3.0 serial string to be programmed.

9.2.2 Device specific hub filter approach

1. For the device specific hub filter approach, append the command “/iv” with all generic commands mentioned in Section 9.2.1

Eg : >pt2main.exe /pspi <firmware_filename.bin> /id <index> /iv

2. If the firmware location is SPI Flash ROM before SPI flash programming then the Protouch2 tool will try to boot the device from internal ROM where it may take time for USB enumeration (The driver loading time based on PC performance). The default timeout value is 15 seconds. After 15 seconds time out, if the device is not enumerated then the tool will come out of the programming with the error code.

The default time out value can be overridden using following command,

```
>pt2main.exe /pspi <firmware_filename.bin> /id <index> /d <Time_in_Milliseconds>
```

- /d <Time_in_Milliseconds> is time out value
- 3. After programing, reset will be done automatically for the device to boot from SPI flash ROM.

9.2.3 Time optimised Production line programming method

The programming time will be more for USB253X/USB4604 family hubs to program SPI firmware and configuration file in one shot since it does a re-enumeration after programming the firmware to program the configuration file. To avoid more time for programming, do the following

1. The hub filter driver should be installed as class filter.
2. Program SPI flash firmware and erase existing Pseudo-OTP configurations.

```
>pt2main.exe /pspi <firmware_filename.bin> /id <index> /e
```

- /e is the option to erase existing Pseudo-OTP configurations.

3. Program the configuration file generated by Protouch2 GUI tool.

```
>pt2main.exe /p <config_file.cfg> /id <index>
```

4. Read the entire SPI flash firmware memory and Pseudo-OTP configuration memory. This will generate **Read_SPI_MM_DD_YYYY_HH_MM_SS.bin** file

```
>pt2main.exe /rspi /cfg /id <index>
```

5. **Read_SPI_MM_DD_YYYY_HH_MM_SS.bin** will be created in the same running directory based on current date and time. The newly read bin file will have the binary data's of firmware and configuration memory. This bin file can be used to program the SPI flash firmware along with configuration file in production line.

```
>pt2main.exe /pspi < Read_SPI_MM_DD_YYYY_HH_MM_SS.bin > /id <index>
```

9.2.4 Erasing SPI Flash

SPI flash firmware contents can be erased by using the following command

```
>pt2main.exe /pspi DisableSPIFlash.bin /id <index>
```

DisableSPIFlash.bin binary file can be found in the released package.

If the SPI Flash configuration memory should also be erased along with firmware memory, please extend the “DisableSPIFlash.bin” file for another 64K with the same contents as “DisableSPIFlash.bin” to make it 128K and then program using the command given above.

9.3 Single Device OTP Programming

Please follow Steps 1, 2, 3 in Section 9.1.1

9.3.1 OTP Programming

1. Following method can be used to program OTP if either internal HFC is enabled by default or hub filter is installed as class filter.

```
>pt2main.exe /p < config_filename.cfg> [/id <index>]
```

- /p <config_filename.cfg> is to program configuration file into OTP memory.
- /id <index> is index of the hub to be programmed.

For device specific filter approach

```
>pt2main.exe /p < config_filename.cfg> [/id <index>] /iv
```

2. After programming, reset will be done automatically for the device.

9.3.2 Program OTP and Lock OTP – USB57x4 Family

Once the OTP programming is successful, the OTP memory can be locked using the command (/pl) to avoid further programming. This support is only available in USB57x4 Family. This is not applicable if ROM is running from SPI flash.

E.g.:

```
>pt2main.exe /p < config_filename.cfg> [/id <index>] /pl
```

9.3.3 Program OTP and USB serial string

1. Following method can be used to program OTP along with serial number if either internal HFC is enabled by default or hub filter is installed as class filter.

```
>pt2main.exe /p < config_filename.cfg> [/id <index>] /pser <serial_string>
```

- /p <config_filename.cfg> is to program configuration file into OTP memory.
- /id <index> is index of the hub to be programmed.
- /pser <serial_string> is the USB serial number in string descriptor.

For device specific filter approach,

```
>pt2main.exe /p < config_filename.cfg> [/id <index>] /pser <serial_string> /iv
```

Note: For USB57X4 family, same serial number will be programmed for USB 2.0 and USB 3.1 Gen1.

2. After programing, reset will be done automatically for the device.

9.3.4 Program OTP and Verify Programmed Configuration Item

1. Verification will be done once the OTP is programmed and reset is done for the device. Sometimes the device enumeration may take time after reset due to the time taken of driver loading for the device. The default timeout value is 15 seconds. After 15 seconds time out, if the device is not enumerated then the tool will come out of the programming with the error code. The default timeout time can be over ridden using the command /d.

Program OTP and Verify configuration items like Vendor ID, Product ID etc.

>pt2main.exe /p < config_filename.cfg> [/id <index>] [/cv <configuration item names>] [/d <Delay_In_Milliseconds>]

- /p <config_filename.cfg> is to program configuration file into OTP memory.
- /id <index> is index of the hub to be programmed.
- /cv <configuration item names> is the command to verify the mentioned configuration items. Refer [Section 9.6](#) for more details.

9.4 Batch Programming (Automated Execution)

This option is used for programming devices one after the other in batch mode. This is used for mass programming.

Please follow Steps 1, 2, 3 in Section 9.1.1

9.4.1 Mass OTP Programming (Single MCHP device is connected)

1. Following method can be used to program OTP if either internal HFC is enabled by default or hub filter is installed as class filter.

```
>pt2main.exe /bp < config_filename.cfg> [/id <index>]
```

- /bp <config_filename.cfg> is to program configuration file into OTP memory in continuous mode.
- /id <index> is index of the hub to be programmed.

For device specific filter approach

```
>pt2main.exe /bp < config_filename.cfg> [/id <index>] /iv
```

```
Administrator: C:\Windows\System32\cmd.exe
D:\Perforce\depot_dsg\Apps Group\Release\PT2\PT2_v1.5\PT2_CLI>pt2main.exe /bp otp_image_1234.cfg /id 0 /iv
Waiting for device arrival...
New device detected - "UID": "0424", "PID": "4504", "DID": "1234"
Programming in progress...
Device programmed successfully
Please remove the device and press any key after inserting new one...
Waiting for device arrival...
New device detected - "UID": "0424", "PID": "4504", "DID": "1234"
Programming in progress...
Device programmed successfully
Please remove the device and press any key after inserting new one...
D:\Perforce\depot_dsg\Apps Group\Release\PT2\PT2_v1.5\PT2_CLI>
```

2. Press “CTRL+C” keys to abort the programming.
3. After each OTP programming, coloured status will be displayed in command line.
GREEN – OTP Program success
RED - OTP program failed
4. After each OTP programming, device reset will be done automatically.

9.4.2 Mass OTP Programming (Multiple MCHP devices are connected)

This method must be used if more than one microchip hub is connected in the machine.

1. Following method can be used to program OTP if either internal HFC is enabled by default or hub filter is installed as class filter.

```
>pt2main.exe /bp < config_filename.cfg> [/devpath  
"VVVV/PPPP/portchain"]
```

- /bp <config_filename.cfg> is to program configuration file into OTP memory in continuous mode.
- /devpath "VVVV/PPPP/portchain" is the port chain of the hub to be programmed.

For device specific filter approach

```
>pt2main.exe /bp < config_filename.cfg> [/id <index>] /iv
```

2. Press "CTRL+C" keys to abort the programming.
3. After each OTP programing, coloured status will be displayed in command line.
GREEN – OTP Program success
RED - OTP program failed

4. After each OTP programming, device reset will be done automatically.

9.4.3 Mass program OTP and USB serial string

1. Following method can be used to program OTP along with different serial number if either internal HFC is enabled by default or hub filter is installed as class filter.

```
>pt2main.exe /bp < config_filename.cfg> [/id <index>] /pser [/alpha_prefix  
<string>] [/start_val <decimal>] [/inc_val <decimal>] [/max_val  
<decimal>]
```

- /bp <config_filename.cfg> is to program configuration file into OTP memory in continuous mode /id <index> is index of the hub to be programmed.
- /alpha_prefix <string> is the string to be prepended with all serial numbers. E.g., MCHP , TESTDEVICE
- /start_val <decimal> is the suffix start value of the serial string.
- /inc_val <decimal> is the value to be incremented for each serial string,

- max_val <decimal> is the maximum value of the serial string to be programmed. After the specified maximum value, the tool will stop the programming.

E.g.: >pt2main.exe /bp config_filename.cfg /id 0 /pser /alpha_prefix Mchp /start_val 0 /inc_val 2 /max_val 6

For device specific filter approach,

>pt2main.exe /bp < config_filename.cfg> [/id <index>] /pser
<serial_string> /iv

Note: For USB57X4 family, same serial number will be programmed for USB 2.0 and USB 3.1 Gen1.

2. Press “CTRL+C” keys to abort the programming.
3. After each OTP programing, coloured status will be displayed in command line.
GREEN – OTP Program success
RED - OTP program failed
4. After each OTP programming, device reset will be done automatically.

9.4.4 Mass program OTP and Verify Programmed Configuration Item

1. Verification will be done once the OTP is programmed and reset is done for the device. Sometimes the device enumeration may take time after reset due to the time taken of driver loading for the device. The default timeout value is 15 seconds. After 15 seconds time out, if the device is not enumerated then the tool will come out of the programming with the error code. The default timeout time can be over ridden using the command /d.

Program OTP and Verify configuration items like Vendor ID, Product ID etc.

>pt2main.exe /bp < config_filename.cfg> [/id <index>] [/cv <configuration item names>] [/d <Delay_In_Milliseconds>]

- /bp <config_filename.cfg> is to program configuration file into OTP memory.
- /id <index> is index of the hub to be programmed.
- /cv <configuration item names> is the command to verify the mentioned configuration items. Refer [Section 9.6](#) for more details.

Press “CTRL+C” keys to abort the programming.

9.5 Verification of Hub Configuration Items

The following configuration parameters can be read from a device to compare and verify correct operation (/cv):

The configuration items supported for USB hubs are

1. vid
2. pid
3. did
4. usbvcd
5. languageid
6. manufacturer
7. product
8. serial

The configuration items supported by USB57X4;

1. usb3vid
2. usb3pid
3. usb3did
4. usb3vcd
5. usb3languageid
6. usb3manufacturer
7. usb3product
8. usb3serial

9.5.1 Supported Parameters

9.5.1.1 vid

This is a 16-bit value that uniquely identifies the Vendor of the USB2 user device (idVendor: assigned by USB-Interface)

9.5.1.2 pid

This is a 16-bit value that the Vendor can assign that uniquely identifies this particular product for USB2 user device (idProduct)

9.5.1.3 did

This is a 16-bit device release number for USB2 user device in BCD format (bcdDevice)

9.5.1.4 usbvcd

USB2 Specification Release Number in BCD format (bcdUSB)

9.5.1.5 languageid

Language ID of the USB2 HUB.

9.5.1.6 manufacturer

Manufacturer String of the USB2 HUB.

9.5.1.7 product

Product String of the USB2 HUB.

9.5.1.8 serial

Serial String of the USB2 HUB.

9.5.1.9 usb3vid

This is a 16-bit value that uniquely identifies the Vendor of the USB3 user device (idVendor: assigned by USB-Interface)

9.5.1.10 usb3pid

This is a 16-bit value that the Vendor can assign that uniquely identifies this particular product for USB3 user device (idProduct)

9.5.1.11 usb3did

This is a 16-bit device release number for USB3 user device in BCD format (bcdDevice)

9.5.1.12 usb3vcd

USB3 Specification Release Number in BCD format (bcdUSB)

9.5.1.13 usb3languageid

Language ID of the USB3 HUB.

9.5.1.14 usb3manufacturer

Manufacturer String of the USB3 HUB.

9.5.1.15 usb3product

Product string of the USB3 HUB.

9.5.1.16 usb3serial

Serial string of the USB3 HUB.

Only the above configuration items are supported when using /cv command.

9.5.2 Get value of parameter before and after programming

To display the configuration items before and after programming

```
>pt2main.exe /p < config_filename.cfg> /cv "vid,pid" /id <index>  
>pt2main.exe /bp < config_filename.cfg> /cv "vid,pid" /id <index>
```

9.5.3 Get value of any parameter

/cv command can also be used to verify the configuration parameters without programming,

```
>pt2main.exe /cv "vid,pid,did,usbvcd,languageid,manufacturer,product,serial"  
/id <index>  
>pt2main.exe /cv "vid,pid" /id <index>
```

9.5.4 Compare value of a parameter with known value

To compare and verify the configuration items, following format should be used
/cv "<ConfigItem1> : <Value>, <ConfigItem2> : <Value>" /id <index>

E.g.

```
>pt2main.exe /p < config_filename.cfg> /cv "vid:0x424,pid:0x2534" /id 0  
  
>pt2main.exe /cv "vid:0x424,pid:0x2534,serial:123456" /id 0
```

9.6 Dump Memory

9.6.1 OTP Memory

This option is used to dump OTP memory of the hub.

9.6.1.1 Hub class filter approach,

>pt2main.exe /rotp /id <index>

9.6.1.2 Device specific filter approach,

>pt2main.exe /rotp /id <index> /iv

E.g.:

pt2main.exe /rotp /id 0

9.6.2 SPI Memory

9.6.2.1 SPI Flash Firmware dump

This option is used to dump SPI Flash Firmware Memory of the hub.

9.6.2.1.1 Hub class filter approach

>pt2main.exe /rspi /id <index>

9.6.2.1.2 Device specific filter approach

>pt2main.exe /rspi /id <index> /iv

9.6.2.2 SPI flash firmware and configuration dump

Below option is used to dump pseudo OTP with SPI Flash Firmware Memory.

9.6.2.2.1 Hub class filter approach

>pt2main.exe /rspi /cfg /id <index>

9.6.2.2.2 Device specific filter approach

>pt2main.exe /rspi /cfg /id <index> /iv

E.g. 1:

pt2main.exe /rspi /id 0

E.g. 2:

pt2main.exe /rspi /cfg /id 0

9.7 Changing Vendor ID/Product ID of the Hub

When programming new Vendor ID/Product ID to the hub, driver loading for the hub may take time for unique Vendor ID/Product ID once programming is completed.

If the same Vendor ID/Product ID is programmed to the different hub, hub driver loading will take time only for the first hub per computer.

If **/cv** and **/p** option is used together, delay (**/d**) also should be increased to wait for the hub driver loading.

```
>pt2main.exe /p < config_filename.cfg> /cv "vid:0x424,pid:0x2534" /id 0 /d  
30000
```

9.8 Changing Vendor ID/Product ID of the Hub Controller

If the default vendor ID/Product ID is changed for the hub controller, Protouch2 hub controller driver will not be loaded for the hub controller.

New WinUSB driver package should be generated for the hub controller. Then the new Product ID of the hub controller should be added to the Protouch2.ini file.

Add new hub controller product ID to the HFC_DEV_INFO section in the Protouch2.ini file.

Example:

```
[HFC_DEV_INFO]  
HFC_PID1 = 0x2530;  
HFC_PID2 = 0x2740;  
HFC_PID3 = 0x274E;  
HFC_PID4 = 0x274F;  
HFC_PID5 = 0x1234;
```

(Note: The automatic & manual WinUSB driver installation will not work, if the hub controller Vendor ID & Product ID is changed)

10 LAN78XX Devices

10.1 Single Device EEPROM Programming

1. Refer [Appendix VI](#) to install LAN78xx driver for windows. One time installation is required per system. This step can be skipped if the LAN78xx driver is already installed.
2. Find out the LAN adapter index by executing the command “*pt2main.exe /le*”. Refer [Section 17.2](#) for more details.
Note: Application should have administrator rights.

10.1.1 EEPROM Programming

1. Following method can be used to program EEPROM.

```
>pt2main.exe /pl < myfile.ini or .bin> [/id <index>] [/eel]
```

- /p < myfile.ini//bin > is to program configuration file into OTP memory.
 - /id <index> is index of the LAN adapter to be programmed.
 - /eel is to erase EEPROM content
2. After programing, reset will be done automatically for the device.

10.1.2 Program EEPROM and USB Serial/MAC address

10.1.2.1 Program EEPROM with serial number (Override Serial number)

```
>pt2main.exe /pl < myfile.ini or .bin> [/id <index>] /pser <serial_string>
```

- /p < myfile.ini//bin> is to program configuration file into OTP memory.
- /id <index> is index of the LAN adapter to be programmed.
- /pser <serial_string> is the USB serial number in USB string descriptor.

10.1.2.2 Program EEPROM with MAC address (Override Mac address)

```
>pt2main.exe /pl < myfile.ini or .bin> [/id <index>] /pmac <mac addr>
```

- /p < myfile.ini//bin> is to program configuration file into OTP memory.
- /id <index> is index of the LAN adapter to be programmed.

- /pmac <mac addr> is the Ethernet MAC address for LAN78xx with colon separated (XX:XX:XX:XX:XX).

10.1.2.3 Program EEPROM with MAC address (Override Mac address) and serial number (Override serial number)

```
>pt2main.exe /pl < myfile.ini or .bin> [/id <index>] /pmac <mac addr>
/pser <serial_num>
```

1. After programming, reset will be done automatically for the device.

10.1.3 Program EEPROM and Verify Programmed Configuration Item

1. Verification will be done once the EEPROM is programmed and reset is done for the device. Sometimes the device enumeration may take time after reset due to the time taken of driver loading for the device. The default timeout value is 15 seconds. After 15 seconds time out, if the device is not enumerated then the tool will come out of the programming with the error code. The default timeout time can be over ridden using the command /d.

Program EEPROM and Verify configuration items like Vendor ID, Product ID etc.

```
> pt2main.exe /pl < myfile.ini or .bin> [/id <index>] / [cvl <configuration
item names>] [/d <Delay_In_Milliseconds>]
```

- /p < myfile.ini/bin> is to program configuration file into OTP memory.
- /id <index> is index of the LAN adapter to be programmed.
- /cvl <configuration item names> is the command to verify the mentioned configuration items. Refer [Section 12](#) for more details.
- /d <Delay_In_Milliseconds> is the time out for device reset. Device will be restarted once the programming is completed. Application would start to scan the LAN devices again after the time out specified in this argument.

10.2 Single Device OTP Programming

10.2.1 OTP Programming

1. Refer [Appendix VI](#) to install LAN78xx driver for windows. One time installation is required per system. This step can be skipped if the LAN78xx driver is already installed.

2. Find out the LAN adapter index by executing the command “*pt2main.exe /le*”. Refer [Section 17.2](#) for more details.

Note: Application should have administrator rights.

3. Following method can be used to program OTP.

```
>pt2main.exe /pl < myfile.ini or .bin> [/id <index>] /o
```

- /p < myfile.ini//bin > is to program configuration file into OTP memory.
- /id <index> is index of the LAN adapter to be programmed.
- /o – Memory selects to OTP

4. After programing, reset will be done automatically for the device.

10.2.2 Program OTP and USB Serial/MAC address

Replace Step 3 in Section 10.2.1 with the following

10.2.2.1 Program OTP with serial number (Override Serial number)

```
>pt2main.exe /pl < myfile.ini or .bin> [/id <index>] /o /pser <serial_string>
```

- /p < myfile.ini//bin> is to program configuration file into OTP memory.
- /id <index> is index of the LAN adapter to be programmed.
- /o – Memory selects to OTP
- /pser <serial_string> is the USB serial number in string descriptor.

10.2.2.2 Program OTP with MAC address (Override Mac address)

```
>pt2main.exe /pl < myfile.ini or .bin> [/id <index>] /o /pmac <mac addr>
```

- /p < myfile.ini//bin> is to program configuration file into OTP memory.
- /id <index> is index of the LAN adapter to be programmed.
- /o – Memory selects to OTP

- /pmac <mac addr> is the Ethernet MAC address for LAN78xx with colon separated (XX:XX:XX:XX:XX).

10.2.2.3 Program OTP with MAC address (Override Mac address) and serial number (Override serial number)

```
>pt2main.exe /pl < myfile.ini or .bin> [/id <index>] /o /pmac <mac addr>
/pser <serial_num>
```

1. After programing, reset will be done automatically for the device.

10.2.3 Program OTP and Verify Programmed Configuration Item

1. Refer [Appendix VI](#) to install LAN78xx driver for windows. One time installation is required per system. This step can be skipped if the LAN78xx driver is already installed.
2. Find out the LAN adapter index by executing the command “*pt2main.exe /le*”. Refer [Section 17.2](#) for more details.
Note: Application should have administrator rights.
3. Verification will be done once the OTP is programmed and reset is done for the device. Sometimes the device enumeration may take time after reset due to the time taken of driver loading for the device. The default timeout value is 15 seconds. After 15 seconds time out, if the device is not enumerated then the tool will come out of the programming with the error code. The default timeout time can be over ridden using the command /d.

Program OTP and Verify configuration items like Vendor ID, Product ID etc.

```
>pt2main.exe /pl < myfile.ini or .bin> /o [/id <index>] / [cvl <configuration
item names>] [/d <Delay_In_Milliseconds>]
```

- /p < myfile.ini//bin> is to program configuration file into OTP memory.
- /id <index> is index of the LAN adapter to be programmed.
- /o – Memory selects to OTP
- /cvl <configuration item names> is the command to verify the mentioned configuration items. Refer [Section 12](#) for more details.

10.3 Batch Programming (Automated Execution)

This option is used for programming devices one after the other in batch mode. This is used for mass programming.

10.3.1 Mass EEPROM/OTP Programming (Single device is connected)

1. Refer [Appendix VI](#) to install LAN78xx driver for windows. One time installation is required per system. This step can be skipped if the LAN78xx driver is already installed.
2. Find out the LAN adapter index by executing the command “**pt2main.exe /le**”. Refer [Section 17.2](#) for more details.
Note: Application should have administrator rights.
3. Following method can be used to program EEPROM/OTP.

>pt2main.exe /bpl < myfile.ini or .bin> [/id <index>] /o

- /p < myfile.ini//bin > is to program configuration file into OTP memory.
 - /id <index> is index of the LAN adapter to be programmed.
 - /o – Memory selects to OTP, if this argument is not given default memory selects to EEPROM
4. After programing, reset will be done automatically for the device.
 5. Press “CTRL+C” keys to abort the programming.
 6. After each OTP programing, coloured status will be displayed in command line.
GREEN – OTP Program success
RED - OTP program failed
 7. After each OTP programming, device reset will be done automatically.

10.3.2 Mass program EEPROM/OTP and USB serial string

1. Refer [Appendix VI](#) to install LAN78xx driver for windows. One time installation is required per system. This step can be skipped if the LAN78xx driver is already installed.
2. Find out the LAN adapter index by executing the command “**pt2main.exe /le**”. Refer [Section 17.2](#) for more details.
Note: Application should have administrator rights.

- Following method can be used to program EEPROM/OTP along with different serial number.

```
> pt2main.exe /bpl < myfile.ini or .bin> [/id <index>] /o /pser [/alpha_prefix
<string>] [/start_val <decimal>] [/inc_val <decimal>] [/max_val <decimal>]
```

- /bpl < myfile.ini//bin > is to program configuration file into OTP memory in continuous mode
- /id <index> is index of the LAN adapter to be programmed.
- /o – Memory selects to OTP, if this argument is not given default memory selects to EEPROM
- /pser – Selects serial number mode.
- /alpha_prefix <string> is the string to be prepended with all serial numbers. E.g., MCHP, TESTDEVICE. If alpha prefix do not require, then NULL string should be given. E.g., /alpha_prefix ""
- /start_val <decimal> is the suffix start value of the serial string.
- /inc_val <decimal> is the value to be incremented for each serial string.
- max_val <decimal> is the maximum value of the serial string to be programmed. After the specified maximum value, the tool will stop the programming.

E.g.: >pt2main.exe /bpl config_filename.bin /id 0 /o /pser /alpha_prefix Mchp /start_val 0 /inc_val 2 /max_val 6

- Press “CTRL+C” keys to abort the programming.
- After each OTP programing, coloured status will be displayed in command line.
GREEN – OTP Program success
RED - OTP program failed
- After each OTP programming, device reset will be done automatically.

10.3.3 Mass program EEPROM/OTP and Mac Address

- Refer [Appendix VI](#) to install LAN78xx driver for windows. One time installation is required per system. This step can be skipped if the LAN78xx driver is already installed.
- Find out the LAN adapter index by executing the command “pt2main.exe /le”. Refer [Section 17.2](#) for more details.
Note: Application should have administrator rights.
- Following method can be used to program OTP along with different mac address.

```
> pt2main.exe /bpl < myfile.ini or .bin> [/id <index>] /o [/pmac
XX:XX:XX:XX:XX:XX] [ /inc_mac < decimal>] [max_mac <hex>]
```

- /bpl < myfile.ini//bin > is to program configuration file into OTP memory in continuous mode
- /id <index> is index of the LAN adapter to be programmed.
- /o – Memory selects to OTP, if this argument is not given default memory selects to EEPROM
- /pmac XX:XX:XX:XX:XX:XX – Mac address to be programmed.
- /inc_mac <decimal> is the value to be incremented for each mac address.
- /max_mac <hex> is the maximum value of the mac address to be programmed. After the specified maximum value, the tool will stop the programming.

E.g.: >pt2main.exe /bpl config_filename.bin /id 0 /pmac
“00:11:22:33:44:55” /o /inc_mac 1 /max_mac 0x60

4. Press “CTRL+C” keys to abort the programming.
5. After each OTP programing, coloured status will be displayed in command line.
GREEN – OTP Program success
RED - OTP program failed
6. After each OTP programming, device reset will be done automatically.

10.3.4 Mass program EEPROM/OTP with Mac Address and serial number

1. Refer [Appendix VI](#) to install LAN78xx driver for windows. One time installation is required per system. This step can be skipped if the LAN78xx driver is already installed.
2. Find out the LAN adapter index by executing the command “*pt2main.exe /le*”. Refer [Section 17.2](#) for more details.
Note: Application should have administrator rights.
3. Following method can be used to program OTP along with different mac address.

```
> pt2main.exe /bpl < myfile.ini or .bin> [/id <index>] /o [/pmac
XX:XX:XX:XX:XX:XX] [ /inc_mac < decimal>] [max_mac <hex>]
/pser [/alpha_prefix <string>] [/start_val <decimal>] [/inc_val <decimal>]
[/max_val <decimal>]
```


- /bpl < myfile.ini//bin > is to program configuration file into OTP memory in continuous mode
- /id <index> is index of the LAN adapter to be programmed.
- /o – Memory selects to OTP, if this argument is not given default memory selects to EEPROM.
- /pmac XX:XX:XX:XX:XX:XX – Mac address to be programmed.
- /inc_mac <decimal> is the value to be incremented for each mac address.
- max_mac <hex> is the maximum value of the mac address to be programmed. After the specified maximum value, the tool will stop the programming.
- /pser – Selects serial number mode.
- /alpha_prefix <string> is the string to be prepended with all serial numbers. E.g., MCHP, TESTDEVICE. If alpha prefix do not require, then NULL string should be given. E.g., /alpha_prefix ""
- /start_val <decimal> is the suffix start value of the serial string.
- /inc_val <decimal> is the value to be incremented for each serial string.
- max_val <decimal> is the maximum value of the serial string to be programmed. After the specified maximum value, the tool will stop the programming.

E.g.: >pt2main.exe /bpl config_filename.bin /id 0 /pmac
 “00:11:22:33:44:55” /o /inc_val 1 /max_val 0x60 /pser /alpha_prefix Mchp
 /start_val 0 /inc_val 2 /max_val 6

4. Press “CTRL+C” keys to abort the programming.
5. After each OTP programing, coloured status will be displayed in command line.
 GREEN – OTP Program success
 RED - OTP program failed
6. After each OTP programming, device reset will be done automatically.

10.3.5 Mass program OTP and Verify Programmed Configuration Item

1. Refer [Appendix VI](#) to install LAN78xx driver for windows. One time installation is required per system. This step can be skipped if the LAN78xx driver is already installed.
2. Find out the LAN adapter index by executing the command “*pt2main.exe /le*”. Refer [Section 17.2](#) for more details.

Note: Application should have administrator rights.

3. Verification will be done once the OTP is programmed and reset is done for the device. Sometimes the device enumeration may take time after reset due to the time taken of driver loading for the device. The default timeout value is 15 seconds. After 15 seconds time out, if the device is not enumerated then the tool will come out of the programming with the error code. The default timeout time can be over ridden using the command /d.

Program OTP and Verify configuration items like Vendor ID, Product ID etc.

```
> pt2main.exe /bpl < myfile.ini or .bin> /o [/id <index>] /[/cv  
<configuration item names>] [/d <Delay_In_Milliseconds>]
```

- /bpl < myfile.ini//bin> is to program configuration file into OTP memory.
- /id <index> is index of the LAN adapter to be programmed.
- /o – Memory selects to OTP, if this argument is not given default memory selects to EEPROM.
- /cv <configuration item names> is the command to verify the mentioned configuration items. Refer [Section 12](#) for more details.

4. Press “CTRL+C” keys to abort the programming.

10.4 Verification of LAN Configuration Items

The following configuration parameters can be read from a device to compare and verify correct operation (/cvl).

The supported configuration items are listed below.

1. **macaddr**

This is the 6-byte Mac address. Bytes are separated by a colon.

2. **usb2vid**

This is a 16-bit value that uniquely identifies the Vendor of the USB2 user device (idVendor: assigned by USB-Interface)

3. **usb2pid**

This is a 16-bit value that the Vendor can assign that uniquely identifies this particular product for USB2 user device (idProduct)

4. **usb2did**

This is a 16-bit device release number for USB2 user device in BCD format (bcdDevice)

5. usb3vid

This is a 16-bit value that uniquely identifies the Vendor of the USB3 user device (idVendor: assigned by USB-Interface)

6. usb3pid

This is a 16-bit value that the Vendor can assign that uniquely identifies this particular product for USB3 user device (idProduct)

7. usb3did

This is a 16-bit device release number for USB3 user device in BCD format (bcdDevice)

8. langid

Language ID of the device.

9. manufacturer

Manufacturer string of the device.

10. product

Product string of the device.

11. serial

Serial string of the device.

12. configstring

Configuration string of the device

13. interfacestring

Interface string of the device

14. enremwakeup

Remote wake up feature of the device.

0 - Device does not support the remote wakeup

1 - Device support the remote wakeup

15. enselfpower

The power method of the device

0 - Bus power

1 - Self power

16. FSmaxpower

17. HSmaxpower

It affects the values of the USB configuration descriptor. Maximum Power Consumption in mA for USB 2.0 operation (between 2mA and 500mA for BusPower; between 2mA and 100mA for SelfPower)

18. SSmaxpower

It affects the values of the USB configuration descriptor. Maximum Power Consumption in mA for USB 3.0 operation (between 8mA and 896mA for BusPower; between 8mA and 100mA for SelfPower)

19. FSInterruptEPInterval

Full speed interrupt polling interval. The values are in decimal and in the range 0 - 255. The default value is 1.

20. HSInterruptEPInterval

High speed interrupt polling interval. The values are in decimal and in the range 0 - 16. The default value is 4.

21. SSInterruptEPInterval

Super speed interrupt polling interval. The values are in decimal and in the range 0 - 16. The default value is 16.

22. LED3Enable

The LED3 status of phy.

23. LED2Enable

The LED3 status of phy.

24. LED1Enable

The LED3 status of phy.

25. LED0Enable

The LED3 status of phy.

26. LED1Function

The LED mode for LED 1

27. LED0Function

The LED mode for LED 0

28. LED3Function

The LED mode for LED 3

29. LED2Function

The LED mode for LED 2

30. LED1Behavior

The hex value of LED Link/pulse behaviour.

31. LED2Behavior

The hex value of LED Link/pulse behaviour.

32. GPIOEnable

Hex values of GPIO groups (0x0FFF)

1 - Disable GPIO

0 - Enable GPIO

33. GPIOBuffer

Hex values of GPIO groups (0x0FFF)

0 - Open-drain

1 - push/pull

34. GPIODirection

Hex values of GPIO groups (0x0FFF)

0 - Output

1 - input

35. GPIOData

Hex values of GPIO groups (0x0FFF)

0 - low

1 – high

36. GPIOWake

Hex values of GPIO groups (0x0FFF)

0 = The GPIO cannot wakeup the device.

1 = The GPIO can trigger a wakeup event.

37. GPIOWakePolarity

Hex values of GPIO groups (0x0FFF)

0 = Wakeup/interrupt is triggered when GPIO is driven low

1 = Wakeup/interrupt is triggered when GPIO is driven high

38. GPIOPMEEEnable

This bit shows the status of the assertion of the GPIO5 pin, as a result of a Wakeup (GPIO) pin, Magic Packet, or PHY Link Up. The host processor may use the GPIO5 pin to asynchronously wake up, in a manner analogous to a PCI PME pin.

0 = the device does not support GPIO PME signalling.

1 = the device supports GPIO PME signalling.

Note: When this bit is 0, the remaining GPIO PME parameters are ignored.

39. GPIOPMEConfiguration

This bit shows the status of whether the GPIO PME is signalled on the GPIO pin as a level or a pulse.

0 = GPIO PME is signalled via a level.

1 = GPIO PME is signalled via a pulse

40. GPIOPMELength

When GPIOPMEConfiguration is set to 1 (pulse), this bit determines the duration of the pulse.

0 = GPIO PME pulse length is 1.5 ms.

1 = GPIO PME pulse length is 150 ms

41. GPIOPMEPolarity

The bit shows the level of the signal or the polarity of the pulse used for GPIO PME signalling.

0 = GPIO PME signalling polarity is low.

1 = GPIO PME signalling polarity is high.

42. GPIOBufferType

This bit shows the output buffer type for GPIO.

0 = Open drain driver / open source

1 = Push-Pull driver

Note: Buffer Type = 0, Polarity = 0 implies Open Drain.

Buffer Type = 0, Polarity = 1 implies Open Source.

43. GPIOPMEWOLSelect

This bit shows whether WOL wakeup events or Link up wakeup events.

0 = WOL event wakeup supported.

1 = PHY linkup wakeup supported.

Note: If WOL is selected, the PME Magic Packet Enable and PME Perfect DA Enable bits determine the WOL event(s) that will cause a wakeup.

44. PMEMagicPacketEnable

When GPIOPMEWOLSelect set to 0 (WOL), this flag enables/disables Magic Packet detection and wakeup.

0 = Magic Packet event wakeup disabled.

1 = Magic Packet event wakeup enabled.

45. PMEPerfectDAEnable

When GPIOPMEWOLSelect set to 0 (WOL), this flag enables/disables Perfect DA detection and wakeup.

0 = Perfect DA event wakeup disabled.

1 = Perfect DA event wakeup enabled.

Only the above configuration items are supported when using **/cwl** command.

10.4.1 Get value of parameter before and after programming

To display the configuration items before and after programming.

```
>pt2main.exe /pl < config_filename.bin> /cvl "macaddr" /id <index>  
>pt2main.exe /bpl < config_filename.bin> /cvl "macaddr" /id <index>
```

10.4.2 Get value of parameter

/cv command can also be used to verify the configuration parameters without programming,

```
>pt2main.exe /cvl "macaddr,languageid,manufacturer,product,serial" /id  
<index>  
>pt2main.exe /cvl "macaddr,enselfpower" /id <index>
```

10.4.3 Compare value of a parameter with known value

To compare and verify the configuration items, following format should be used
/cvl "<ConfigItem1> : <Value>, <ConfigItem2> : <Value>" /id <index>

E.g.

```
>pt2main.exe /pl < config_filename.bin> /cvl "usb2vid:0x424,usb2pid:0x7800"  
/id 0  
>pt2main.exe /cvl "usb2vid:0x424,usb2pid:0x7800,serial:123456" /id 0
```

10.5 Dump Memory

10.5.1 OTP Memory

This option is used to dump OTP memory of the LAN78xx.

```
>pt2main.exe /rl /id <index> /o
```

E.g.:

```
pt2main.exe /rl /id /o
```

10.5.2 EEPROM Memory

This option is used to dump EEPROM Flash Memory of the LAN78xx.

```
>pt2main.exe /rl /id <index>
```

E.g.:

```
pt2main.exe /rl /id 0
```

10.6 Changing Vendor ID/Product ID of the LAN78xx

The Microchip provided LAN78xx driver is supported only for Microchip vendor id and product id.

The user should create their own driver corresponding to the desired Vendor ID and Product ID. The inf files in the directory “\Drivers\LAN78xxDriver\x64\Driver” and “\Drivers\LAN78xxDriver\x86\Driver” should be modified to support the new Vendor ID and Product ID.

Once the inf is changed, the Microsoft regular driver generation and certification must be followed.

When programming new Vendor ID/Product ID to the LAN78xx, driver loading for the LAN78xx may take time for unique Vendor ID/Product ID once programming is completed.

If the same Vendor ID/Product ID is programmed to the different LAN78xx, LAN78xx driver loading will take time only for the first hub per computer.

If **/cvl** and **/pl** option is used together, delay (**/d**) also should be increased to wait for the hub driver loading.

```
>pt2main.exe /pl < config_filename.bin> /cvl “usb2vid:0x424,usb2pid:0x7800”  
/id 0 /d 30000
```


11 Appendix I - Serial Number Suppression

11.1 *Why is it required?*

When programming the serial number to the hub, driver loading may take time for unique serial numbers. If serial number suppression is enabled, it forces the USB driver stack to ignore the serial number of the device. Therefore driver loading will not take long time to load the driver for the device.

11.2 *When is it needed?*

In CLI application, suppression is required for the following arguments

1. **"/cv"** or **"/cvi"** – if compare and verify option is present
2. **"/pser"** - if programming serial number option is present.

11.3 *How to execute SerialNumSuppression.bat file?*

1. Open command prompt as "Run as administrator"
2. Run batch file as "SerialNumSuppression.bat <VVVV> <PPPP> <DDDD>"

Here **VVVV** - Vendor ID of the Hub,
 PPPP - Product ID of the Hub,
 DDDD - Device ID of the Hub,
 ** Four digits must be there, if not there pad zeros.
 ** Give without "0x" format (**Ex:** 0424, 0x0424 is invalid)

E.g.

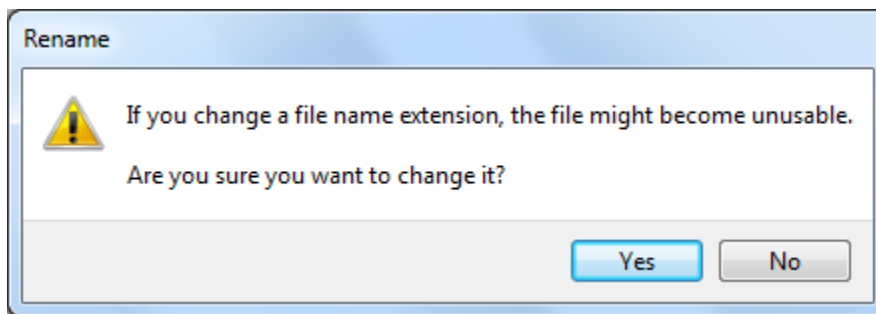
>SerialNumSuppression.bat 0424 2534 1234

12 Appendix II - Changing filename extension

The Protouch2 tool will support only the configuration files which have the extension of “*.cfg”.

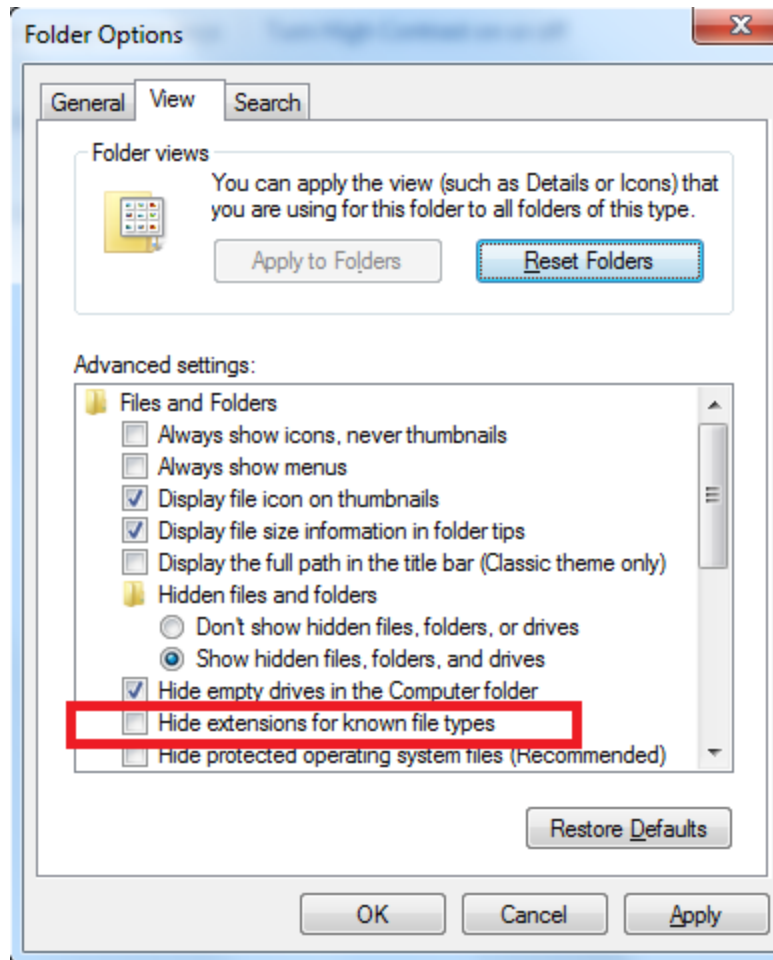
The following method can be used to change the filename extension to “*.cfg” if it was “*.bin”

1. Make sure that filename extensions (.bin) are visible. To know how to do this, see [section 19.2](#).
2. Right-click the file you want to change, and then click Rename.
3. Delete the file name extension (.bin), type the .cfg extension, and then press Enter.
4. Windows will warn you that changing the file name extension might cause the file to stop working properly. If you are certain that the extension you typed will work with the program you're using, click Yes to confirm the change.



12.1 To show or hide file name extensions

1. Open Folder Options by clicking the Start button, clicking Control Panel, clicking Appearance and Personalization, and then clicking Folder Options.
2. Click the View tab, and then, under advanced settings, do one of the following:
 1. To show file name extensions, clear the Hide extensions for known file types check box, and then click OK.
 2. To hide file name extensions, select the Hide extensions for known file types check box, and then click OK.



13 Appendix III – Find Hub Index or path - USB Hubs

The following one of the method can be selected to find the hub location

1. Index method
2. Port chain method

Note: Both index and port chain method cannot be used together.

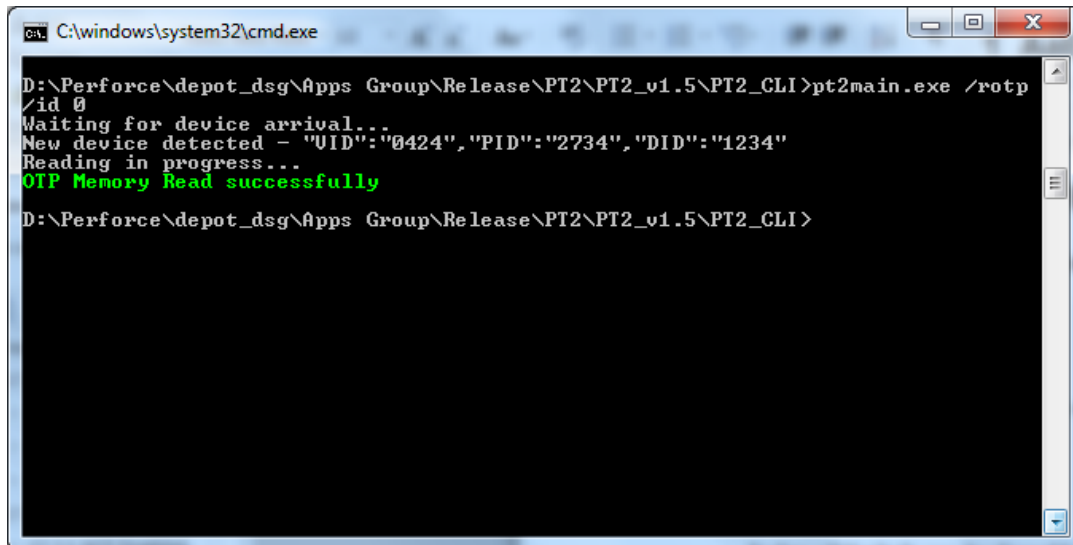
13.1 Index Method

13.1.1 Usage of index in “/id” command

The index of the hub must be mentioned along with “/id” command, to program a specific hub/device. Refer [Section 18.1.2](#) to find an index of the hub.

Example 1:

`>pt2main.exe /rotp /id <index>`



```

C:\windows\system32\cmd.exe
D:\Perforce\depot_dsg\Apps Group\Release\PT2\PT2_v1.5\PT2_CLI>pt2main.exe /rotp
/id 0
Waiting for device arrival...
New device detected - "UID":"0424", "PID":"2734", "DID":"1234"
Reading in progress...
OTP Memory Read successfully
D:\Perforce\depot_dsg\Apps Group\Release\PT2\PT2_v1.5\PT2_CLI>
    
```

Example 2:

`>pt2main.exe /p < config_filename.cfg> /id <index>`

```

C:\windows\system32\cmd.exe
D:\Perforce\depot_dsg\Apps Group\Release\PT2\PT2_v1.5\PT2_CLI>pt2main.exe /p otp_image_1234.cfg /id 0
Waiting for device arrival...
New device detected - "UID":"0424", "PID":"2734", "DID":"1234"
Programming in progress...
Device programmed successfully
D:\Perforce\depot_dsg\Apps Group\Release\PT2\PT2_v1.5\PT2_CLI>

```

13.1.2 To List the hubs or to find the index of a hub

Index of the hub can be found by using the below command

>pt2main.exe /l

Application will populate list of hubs connected to the computer with unique index numbers. By default, microchip hubs will be moved to lower index's based on the Vendor ID (VID) during the initial process of application. This is similar to a tree view listing of the USB devices under the root controller for easy identification of the hub of interest.

```

C:\windows\system32\cmd.exe
D:\Perforce\depot_dsg\Apps Group\Release\PT2\PT2_v1.5\PT2_CLI>pt2main.exe /l
MyComputer
- Mobile 5th Generation Intel(R) Core(TM) USB EHCI Controller - 9CA6
  ->Port:1 : Generic USB Hub <Index:2 ,VID:8087,PID:8001>
- Intel(R) USB 3.0 eXtensible Host Controller
  ->Port:1 : USB 2.0 MTT Hub <Index:0 ,VID:0424,PID:2734>
  ->Port:5 : Microchip Hub Controller
  ->Port:4 : USB Input Device
  ->Port:6 : USB 2.0 Hub <Index:1 ,VID:05e3,PID:0608>
  ->Port:1 : Synaptics FP Sensors (WBF) <PID=0011>
  ->Port:7 : Realtek Bluetooth 4.0 + High Speed Chip
  ->Port:8 : USB Composite Device
  ->Port:12 : USB 3.0 Hub
D:\Perforce\depot_dsg\Apps Group\Release\PT2\PT2_v1.5\PT2_CLI>

```

13.2 Port Chain Method

The port chain is the format which is used to select the device for the programming. The port chain format is generated by the PT2 based on the USB device tree in the computer. This unique identification for the particular hub will remain the same even though any external hubs are connected /disconnected from the device tree. This method can be used for all commands wherever the index method is used.

13.2.1 Usage of port chain in “/devpath” command

The port chain of the hub must be mentioned along with “/devpath” command, to program a specific hub/device.

Format of the device path:

VVVV/PPPP/X-Y-Z

VVVV – Vendor ID of the USB 2.0 Hub in Hexadecimal (without ‘0x’)
 PPPP – Product ID of the USB 2.0 Hub in Hexadecimal (without ‘0x’)
 X-Y-Z – Port chain format from the command “pt2main.exe /lp”. It must be the same format how it displays in the terminal for the /lp command

Example

1. Give /lp command “pt2main.exe /lp”

E.g.

In this example four USB programmable hubs are connected to PC (Computer) directly.

```

6_BETA3\PT2_CLI>pt2main.exe /lp
MyComputer
- Intel(R) 8 Series/C220 Series USB EHCI #1 - 8C26
!->Port:1 : Generic USB Hub <PortChain:1-1 ,VID:8087,PID:8000>
- Intel(R) 8 Series/C220 Series USB EHCI #2 - 8C2D
!->Port:1 : Generic USB Hub <PortChain:2-1 ,VID:8087,PID:8008>
- Intel(R) USB 3.0 extensible Host Controller
!->Port:1 : USB 2.0 MTT Hub <PortChain:3-1 ,VID:0424,PID:2734>
! !->Port:5 : Microchip Hub Controller
!->Port:2 : USB 2.0 MTT Hub <PortChain:3-2 ,VID:0424,PID:2734>
! !->Port:5 : Microchip Hub Controller
!->Port:3 : USB 2.0 MTT Hub <PortChain:3-3 ,VID:0424,PID:2734>
! !->Port:5 : Microchip Hub Controller
!->Port:6 : USB 2.0 MTT Hub <PortChain:3-6 ,VID:0424,PID:2734>
! !->Port:5 : Microchip Hub Controller
!->Port:7 : Synaptics FP Sensors (WBF) <PID=0011>
!->Port:12 : USB Composite Device
!->Port:16 : USB 3.0 Hub
    
```

2. Find out the port chain of programmable hubs.

As per step 1, all USB programmable hubs have the vendor ID/Product ID of 0x0424/0x2734 and the respective port chains are printed in the terminal along with hub vendor ID and Product ID.

E.g.:

The port chain would be “3-6” (PortChain: 3-6 ,VID:0424,PID:2734) for the red highlighted hub as shown below.

```
.6_BETA3\PT2_CLI>pt2main.exe /lp
MyComputer
- Intel(R) 8 Series/C220 Series USB EHCI #1 - 8C26
!->Port:1 : Generic USB Hub <PortChain:1-1 ,VID:8087,PID:8000>
- Intel(R) 8 Series/C220 Series USB EHCI #2 - 8C2D
!->Port:1 : Generic USB Hub <PortChain:2-1 ,VID:8087,PID:8008>
- Intel(R) USB 3.0 eXtensible Host Controller
!->Port:1 : USB 2.0 MTT Hub <PortChain:3-1 ,VID:0424,PID:2734>
!->Port:5 : Microchip Hub Controller
!->Port:2 : USB 2.0 MTT Hub <PortChain:3-2 ,VID:0424,PID:2734>
!->Port:5 : Microchip Hub Controller
!->Port:3 : USB 2.0 MTT Hub <PortChain:3-3 ,VID:0424,PID:2734>
!->Port:5 : Microchip Hub Controller
!->Port:6 : USB 2.0 MTT Hub <PortChain:3-6 ,VID:0424,PID:2734>
!->Port:5 : Microchip Hub Controller
!->Port:7 : Synaptics FP Sensors <WBF> <PID=0011>
!->Port:12 : USB Composite Device
!->Port:16 : USB 3.0 Hub
```

3. Create the device path using port chain.

E.g.

The device path would be “0424/2734/3-6” (VendorID/ProductID/Portchain) for the highlighted as shown below.

```
.6_BETA3\PT2_CLI>pt2main.exe /lp
MyComputer
- Intel(R) 8 Series/C220 Series USB EHCI #1 - 8C26
!->Port:1 : Generic USB Hub <PortChain:1-1 ,VID:8087,PID:8000>
- Intel(R) 8 Series/C220 Series USB EHCI #2 - 8C2D
!->Port:1 : Generic USB Hub <PortChain:2-1 ,VID:8087,PID:8008>
- Intel(R) USB 3.0 eXtensible Host Controller
!->Port:1 : USB 2.0 MTT Hub <PortChain:3-1 ,VID:0424,PID:2734>
!->Port:5 : Microchip Hub Controller
!->Port:2 : USB 2.0 MTT Hub <PortChain:3-2 ,VID:0424,PID:2734>
!->Port:5 : Microchip Hub Controller
!->Port:3 : USB 2.0 MTT Hub <PortChain:3-3 ,VID:0424,PID:2734>
!->Port:5 : Microchip Hub Controller
!->Port:6 : USB 2.0 MTT Hub <PortChain:3-6 ,VID:0424,PID:2734>
!->Port:5 : Microchip Hub Controller
!->Port:7 : Synaptics FP Sensors <WBF> <PID=0011>
!->Port:12 : USB Composite Device
!->Port:16 : USB 3.0 Hub
```

4. Mention the device path as CLI arguments along with the keyword “/devpath”

E.g.

>>pt2main.exe /p myconfig.cfg /devpath 0424/2734/3-1

Following CLI commands were created based on port chain command (/lp) as specified in step 1.

To program SPI firmware and erase existing configuration data

>>pt2main.exe /pspi myfirmware.bin /devpath 0424/2734/3-1 /e /d 15000

To program configuration file

```
>> pt2main.exe /p mycfg.cfg /devpath 0424/2734/3-2
```

To read OTP

```
>> pt2main.exe /rotp /devpath 0424/2734/3-3
```

To read SPI flash with configuration area

```
>> pt2main.exe /rspi /cfg /devpath 0424/2734/3-6
```

13.2.2 To find the port chain of a hub

Port chain of the hub can be found by using following command

```
>> pt2main.exe /lp
```

```
F:\Prasanna BKP\rkattukandan_indcontract\depot_dsg\Apps Group\src\PI2\main\Debug
>pt2main.exe /lp
MyComputer
- Intel(R) 8 Series/C220 Series USB EHCI #1 - 8C26
  !->Port:1 : Generic USB Hub <PortChain:1-1 ,VID:8087,PID:8000>
- Intel(R) 8 Series/C220 Series USB EHCI #2 - 8C2D
  !->Port:1 : Generic USB Hub <PortChain:2-1 ,VID:8087,PID:8008>
- Intel(R) USB 3.0 eXtensible Host Controller
  !->Port:1 : USB 2.0 MTT Hub <PortChain:3-1 ,VID:0424,PID:2734>
  ! ! ! !->Port:1 : USB 2.0 MTT Hub <PortChain:3-1-1 ,VID:0424,PID:4504>
  ! ! ! !->Port:1 : USB 2.0 MTT Hub <PortChain:3-1-1-1 ,VID:0424,PID:4504>
  ! ! ! !->Port:2 : USB Mass Storage Device
  ! ! ! !->Port:5 : Microchip Hub Controller
  ! ! ! !->Port:5 : Microchip Hub Controller
  !->Port:7 : Synaptics FP Sensors <WBF> <PID=0011>
  !->Port:12 : USB Composite Device
  !->Port:16 : USB 3.0 Hub
```

From the above example if user wants to program the hub which is highlighted in red, Then the command should be

```
>>pt2main.exe /p myfile.cfg /devpath 0424/4504/3-1-1-1
```


14 Appendix IV

14.1 Troubleshooting

Review the log file for debugging failures. The error code described as shown in [section 19.2](#) below. If debug assistance is required, save the log file and provide it to Microchip support.

14.2 Error codes

General descriptions of the error code are shown in the table below.

Error code	Description
0x0000	Success ; No Errors
0x0001	The specified device was not found
0x0002	Argument passed to the API is invalid
0x0003	Device handle passed to the API is not valid
0x0004	API of the WinUSB library failed
0x0005	System reboot is required
0x0006	Error in installing VSM filter driver
0x0007	Operation successful but requires reboot
0x0008	Bin file size is invalid
0x0009	Error while reading cfg/bin file
0x0010	Code is running from SPI
0x0011	Error in installing WinUSB driver
0x0012	Invalid argument
0x0013	Error when VSM Class Filter is not installed
0x0014	Error when application does not have administrator rights
0x0015	Error if VSM command is failed due to power state of the device
0x0016	Error not supported
0x0017	Error if Memory of the device reached maximum size
0x1000	Could not load the binary file
0x1001	Reading from SPI flash failed
0x1002	File size did not match
0x1003	SPI pass through write command failed
0x1004	SPI pass through Enter command failed 1
0x1005	SPI flash could not be detected or was not present
0x1006	SPI Cancel Download
0x1007	SPI flash programming failed
0x1008	SPI pass through Enter command failed 2
0x1009	SPI pass through read command failed
0x100A	Unsupported SPI flash detected
0x100B	SPI flash read back and compare failed with programmed binary
0x100C	Open Erase signature bin file failed
0x100D	Read Erase signature bin file failed

0x100E	SRAM programming failed
0x100F	SPI Flash Erase Signature Failed
0x1010	SPI Flash Chip Erase Failed
0x1011	Could not load the Json file
0x1012	Could not load the ini file
0x1013	Flex Reg Field was not programmed
0x1014	SPI Flash access is not supported for the device
0x2000	Cannot enable I2C Passthrough interface
0x2001	Cannot disable I2C Passthrough interface
0x2002	I2C Transfer failed
0x2003	I2C Max Size Reached
0x3000	OTP max number of configurations exceeded
0x4000	Communication at the specified baud rate will be error prone
0x4001	Cannot set USB2534 UART registers, probably command failure
0x4002	Transmit failed without transmitting any data
0x4003	Transmit failed after transmitting some data
0x4004	Receive failed due to buffer overrun, reduce baud rate
0x4005	Receive failed due to unexpected Rx FIFO status
0x4006	Receive failed since worker thread creation failed
0x4007	UART Rx is pending due to asynchronous mode
0x4008	UART receive aborted as per user request
0x4009	UART Receive command failed by the firmware
0x400A	Receive failed without receiving any data
0x400B	UART Receive Timeout
0x5000	Invalid GPIO PIN Number
0x6000	Access Denied , Application does not have admin rights
0x6001	LAN78xx adapter is busy
0x6002	General failure in LAN78xx commands
0x6003	Physical eeprom is absent. OTP blank
0x6004	EEPROM absent and OTP has free space
0x6005	EEPROM absent and no free space in OTP
0x6006	EEPROM present and no free space in OTP
0x6007	EEPROM present and OTP is blank
0x6008	EEPROM present and OTP has invalid signature
0x6009	EEPROM absent and OTP has invalid signature
0x600A	EEPROM absent
0x600B	EEPROM is present and highest priority goes to EEPROM
0xFFFF	Unknown error occurred

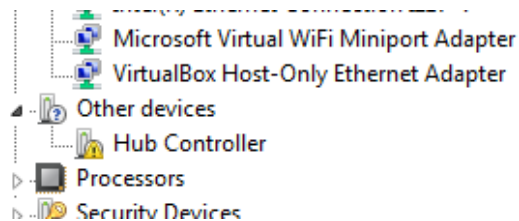
15 Appendix V

15.1 Internal HFC device enabled by Default

Internal HFC device is enabled by default for USB57x4 family hubs and USB4604 hub. To find internal HFC device enumeration, connect the hub to the computer and open device manager and verify as follows.

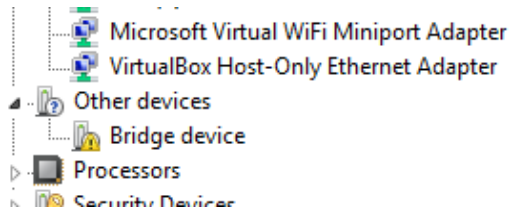
a) USB57x4 Family hub with internal HFC enabled before installing HFC driver:

The following image shows that the internal HFC is enumerated without the HFC windows driver in device manager for USB57x4 family.



b) USB4604 Family hub with internal HFC enabled before installing HFC driver:

The following image shows that the internal HFC is enumerated without the HFC windows driver in device manager for USB4604 hub.



Note: If the internal HFC is enabled for USB253x hubs, then the internal HFC device will appear as above without windows driver.

15.1.1 SKUS

- USB57x4
- USB (8)4604

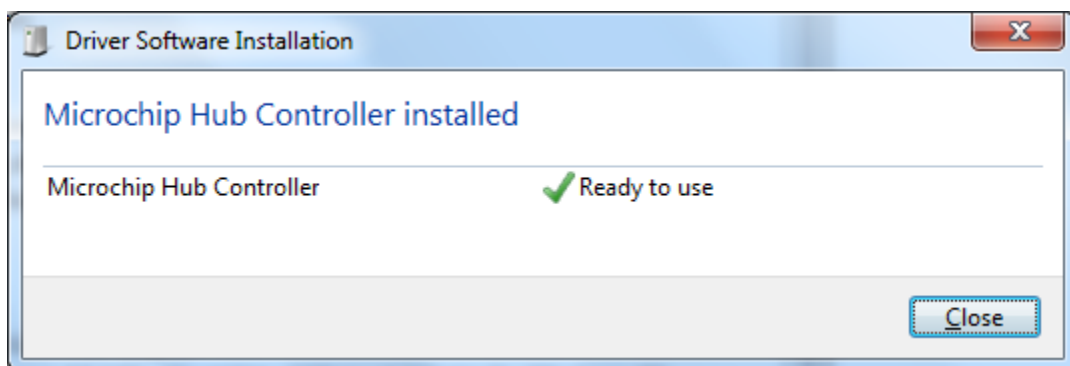
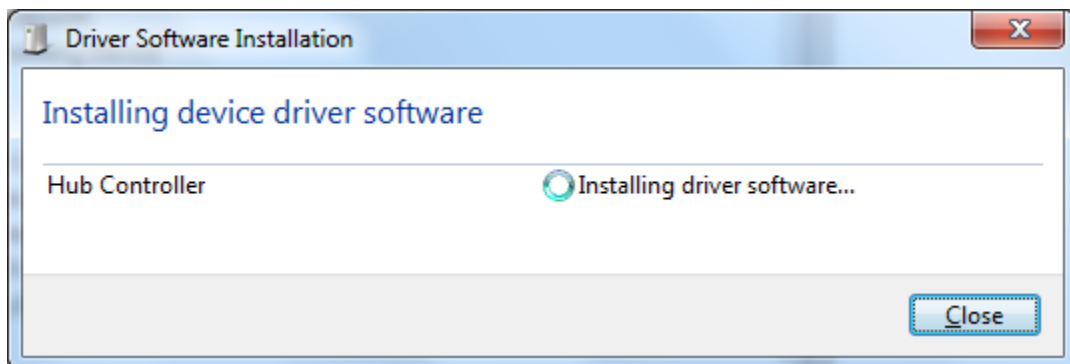
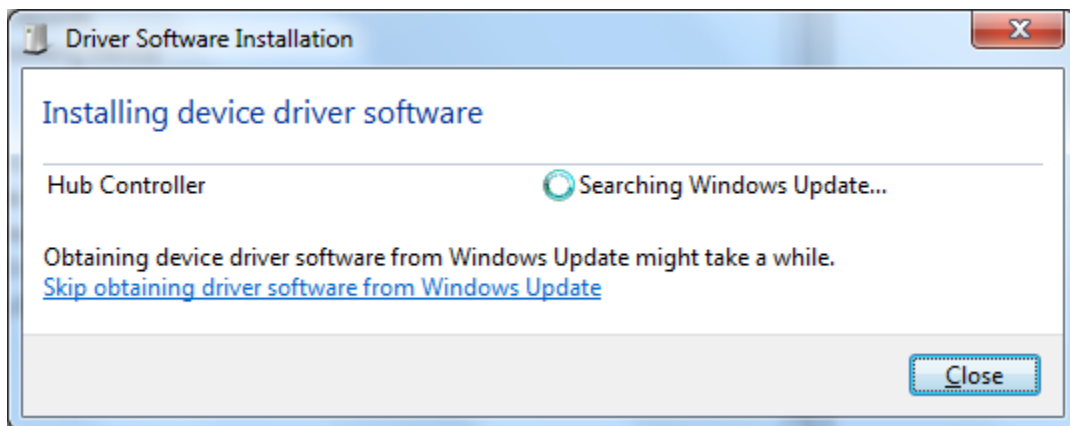
The HFC driver is nothing but the customized winusb driver. The HFC driver installation is required for the internal hub feature controller device before programming. The HFC driver is used for programming the OTP (One-Time Programmable memory) and SPI flash firmware by communicating with the HFC device.

15.1.2 HFC Driver Installation

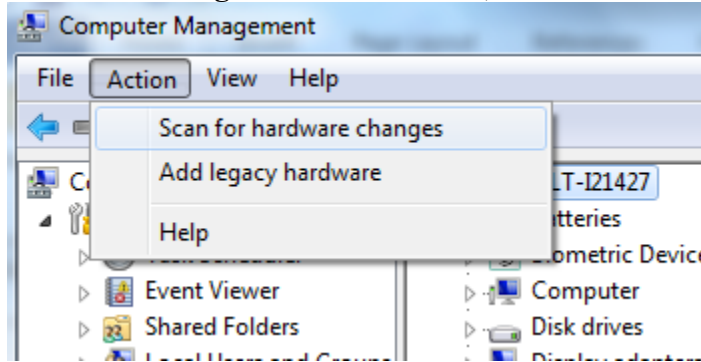
15.1.2.1 Automatic HFC Driver installation

The HFC driver will be installed automatically for the hub feature controller from windows update, once the hub is inserted to the host. The internet connection and windows update must be enabled for the automatic driver installation.

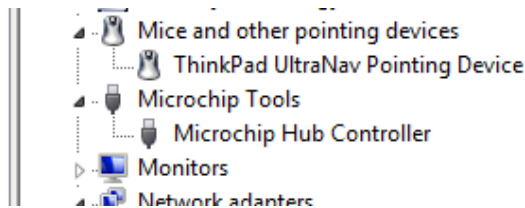
1. The “Driver Software Installation” wizard would appear as shown below if the automatic installation is started.



(Note: If the “Driver Software installation” wizard is not started, then click “**Scan for hardware changes**” as shown below)



2. If the driver installation is completed successfully then the device will appear in device manager as shown below.



15.1.2.2 Manual HFC Driver installation

The HFC driver can be installed manually as one time before programming (ONE TIME only per system). This does not require reboot of the system.

If the automatic HFC driver installation is successful, manual HFC driver installation is not required.

Install the HFC driver using the following command

```
>pt2main.exe /iw
```

15.1.3 HFC Driver Uninstallation

Uninstall the HFC driver using the following command, if the driver was installed manually.

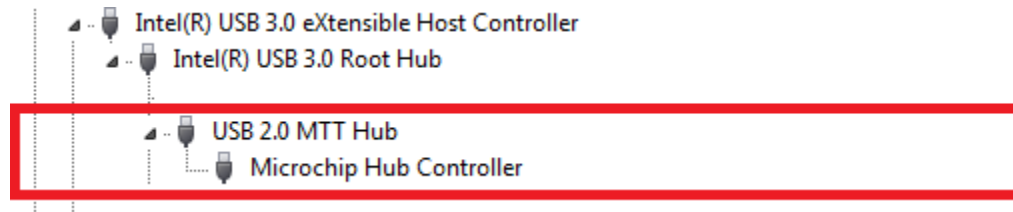
```
>pt2main.exe /uw
```

Note: If the HFC driver was installed manually using one of the version of PT2 CLI application, then the same PT2 CLI version must be used for uninstallation.

15.2 Internal HFC device disabled by Default

Connect the hub to the computer, after installing the hub feature controller driver (Either by manual or automatic from web) as mentioned in the [section 20.1.2](#). The internal hub feature controller is enabled by default in the USB Hub, if the “Microchip Hub Controller” is visible in the device manager.

“Device by Connection” view of Device Manager - The internal hub feature controller is enabled



“Device by Connection” view of Device Manager - The internal hub feature controller is enabled



15.2.1 SKUS

- USB2534

The VSM Driver is required for enabling the internal HFC device (HFC – Hub Feature Controller) and the WinUSB driver is used for programming the OTP (One-Time Programmable memory) by communicating with the HFC device.

The VSM hub filter driver installation can be done in two ways.

1. Hub Class filter
2. Device specific filter

15.2.2 Hub class filter driver

This command has to be executed once for installing both the WinUSB driver and the VSM hub class filter driver

15.2.2.1 Hub class filter driver installation

(One time only per system)

>pt2main.exe /i

Once installed, **system reboot is required**. After reboot, programming can be done any number of times and installation is not required for each new device insertion and removal.

If the Hub class filter is installed, then the Device Specific filter (**/iv** command while programming) method **should not** be used.

15.2.2.2 Hub class filter driver uninstallation

The following command can be used for uninstalling the VSM Hub class filter driver

- **>pt2main.exe /u**

Once uninstalled, **system reboot is required**.

This command can be used for uninstalling the WinUSB driver, if the driver was installed manually.

- **>pt2main.exe /uw**

15.2.2.3 Advantage

This option is useful for users who prefers using batch programming ([Section 11](#)) since it saves time by eliminating the need for installing the driver with each new device insertion

15.2.2.4 Disadvantage

System reboot is required one time.

15.2.3 Hub device specific filter driver

The WinUSB driver installation is required for the device. Refer [Section 20.1.2](#) for the details.

Device specific VSM filter driver should be installed using **/iv** command while programming.

If the VSM filter driver is not installed as Hub Class filter, then the device specific filter driver installation is required for each new device insertion and removal.

Example:

```
>pt2main.exe /p <config_filename.cfg> /id <index> /iv  
>pt2main.exe /pspi <firmware_filename.bin> /id <index> /iv
```

The Device specific VSM filter driver will be uninstalled when application exits from the operation.

System reboot is not required when the driver is installed or uninstalled.

15.2.3.1 Advantage

System reboot is not required when installing/uninstalling the drivers since it does not install the filter driver for all the hubs in the system

15.2.3.2 Disadvantage

Increased programming time for each device since driver installation occurs every time when hub is enumerated.

16 Appendix VI - LAN78xx Driver Installation

1. Run the “install.exe” application from the directory “\Drivers\LAN78xxDriver\”.
2. Accept EULA and proceed to the driver installation.
3. Once the driver installation is successful make sure the LAN78xx device is listed in the device manager.

17 Appendix VII – Find index of LAN device

17.1 Usage of index /id for LAN commands

The index of the LAN device must be mentioned along with “/id” command, to program a specific device. Refer [Section 17.2](#) to find an index of the hub.

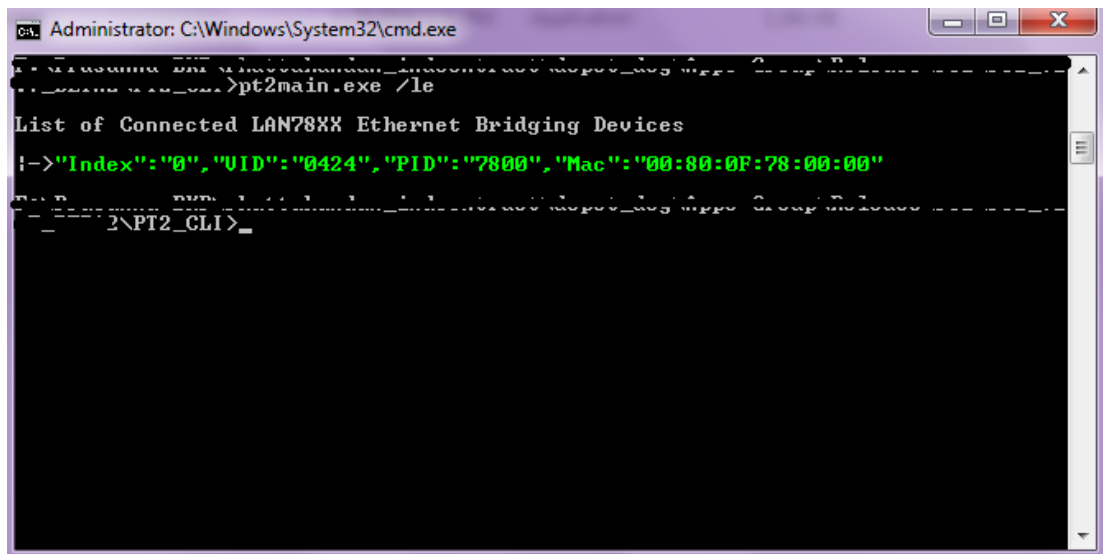
Example 1:

```
>pt2main.exe /pl /id <index>
```

17.2 To List the LAN78xx devices or to find the index of LAN78xx

Index of the LAN78xx can be found by using the below command with administrator rights.

```
>pt2main.exe /le
```



```
Administrator: C:\Windows\System32\cmd.exe
>pt2main.exe /le

List of Connected LAN78XX Ethernet Bridging Devices

Index: "0", "UID": "0424", "PID": "7800", "Mac": "00:80:0F:78:00:00"

2\PT2_CLI>
```

Application will populate list of connected LAN78xx devices to the computer with unique index numbers.