

# **Protouch2 USB57X4 MAC SDK User Guide**

Version 0.9

## Table of Contents

<b>1 Introduction</b>	<b>1-1</b>
<b>2 Legal Information</b>	<b>2-2</b>
<b>3 Package Content</b>	<b>3-3</b>
<b>4 List of APIs</b>	<b>4-4</b>
4.1 Device Open / Close APIs	4-5
4.2 GPIO Bridging APIs	4-6
4.3 XDATA Bridging APIs	4-7
4.4 I2C Bridging APIs	4-8
4.5 SPI Bridging APIs	4-9
4.6 Flexconnect API	4-10
4.7 Programming APIs	4-11
4.8 Miscellaneous APIs	4-12
<b>5 Demo</b>	<b>5-13</b>
5.1 I2C Bridging Demo	5-14
5.1.1 Board Setup	5-14
5.1.2 Executable file	5-14
<b>6 Appendix A</b>	<b>6-15</b>
6.1 libusb Installation	6-16
<b>7 Appendix B</b>	<b>7-17</b>
7.1 Build the static library and the applications	7-18
<b>8 Appendix C</b>	<b>8-20</b>
8.1 I2C supported configuration Modes	8-21

# 1 Introduction

Protouch2 MAC SDK allows the user to access the USB57X4 family of Microchip hubs. It can also be used for exercising unique features like FlexConnect, GPIO, SPI and I2C bridging on top of the hub functionality.

## 2 Legal Information

### Software License Agreement

(c) 2004 - 2015 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 Package Content

**mac:** This Folder contains the pt2lib xcode project

- pt2lib.xcodeproj

**src:** This Folder contains source code for library(lib\_src) and sample application (app\_src)

### **lib\_src**

- MchpUSBInterface.cpp
- MchpUSBInterface.h
- typedef.h
- USB2530\_SpiFlash.cpp
- USB2530\_SpiFlash.h
- USBHubAbstraction.cpp
- USBHubAbstraction.h

### **app\_src**

- FlexConnect.cpp
- gpio.cpp
- i2c\_bridging.cpp
- OTP\_Programmer.cpp
- register\_rw.cpp
- spi\_bridging.cpp

## 4 List of APIs

---

## 4.1 Device Open / Close APIs

This section covers APIs which enable open / close of the device. Before issuing any command to the device, the handle needs to be opened first.

---

## 4.2 GPIO Bridging APIs

This APIs are used for low level control of GPIO pins in Microchip USB hubs. User can configure the direction, pull up / down, read data & write data to any GPIO.

---

## 4.3 XDATA Bridging APIs

This section lists the APIs that enable read / write of register space in Microchip USB hubs.

---

## 4.4 I2C Bridging APIs

Microchip USB hubs facilitate USB-I2C bridging through USB control point of the embedded USB device (5th port).

---

## 4.5 SPI Bridging APIs

This section lists all the USB-SPI bridging APIs.

---

## 4.6 Flexconnect API

Flexconnect refers to the feature in Microchip USB hubs, wherein the upstream port swaps its role with downstream port 1 and also vice versa at run time



---

## 4.7 Programming APIs

This section lists all high level APIs which can be used for programming.

---

## 4.8 Miscellaneous APIs

This section lists all miscellaneous APIs which contains various additional features.

## 5 Demo

## 5.1 I2C Bridging Demo

This demo performs Hub Feature Controller-I2C bridging (USB - I2C)

### 5.1.1 Board Setup

#### USB57X4-EVB

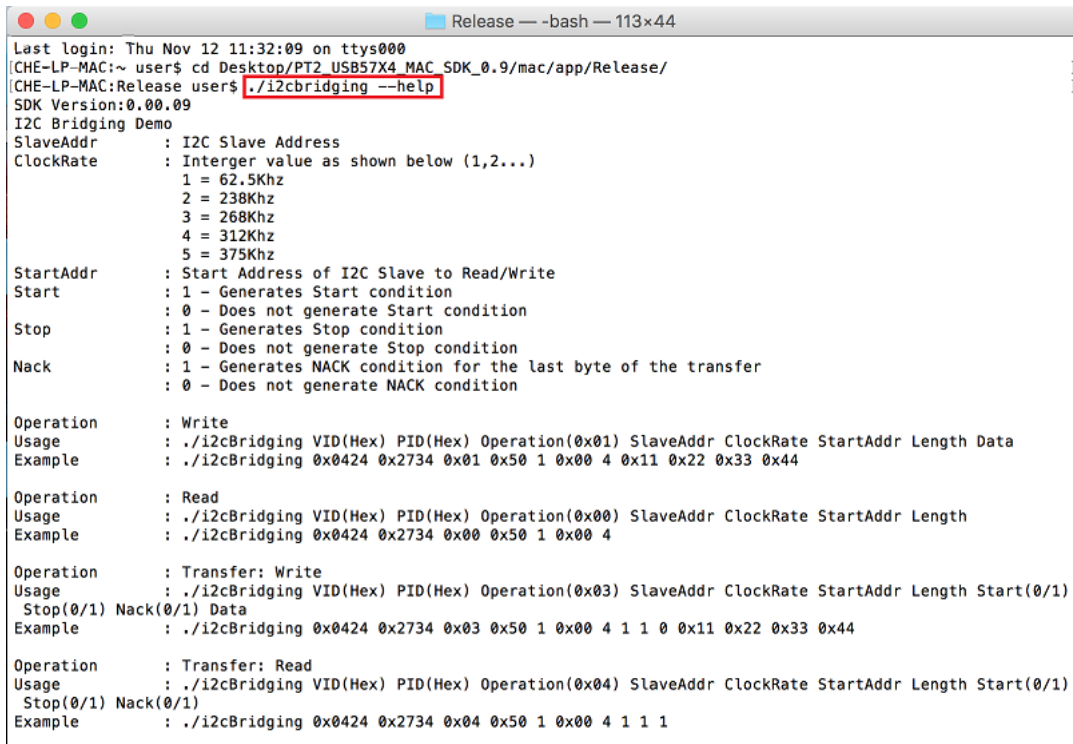
For schematics refer to the link : <http://ww1.microchip.com/downloads/en/DeviceDoc/50002350A.pdf>

Follow the below steps to configure the I2C interface in the EVB

- Connect IO Daughter card to IO Mode Daughter card Interface J7 in EVB USB5734. Refer APPENDIX C to know about I2C supported configuration modes
- Connect AT24C04 EEPROM (SCL,SDA,GND pins) to IO Daughter card

### 5.1.2 Executable file

1. Refer Appendix A for libusb installation
2. Refer Appendix B to build static library and executable file
3. Set 'i2cbridging' as an active Scheme



```

Last login: Thu Nov 12 11:32:09 on ttys000
CHE-LP-MAC:~ user$ cd Desktop/PT2_USB57X4_MAC_SDK_0.9/mac/app/Release/
CHE-LP-MAC:Release user$ ./i2cbridging --help
SDK Version:0.00.09
I2C Bridging Demo
SlaveAddr      : I2C Slave Address
ClockRate      : Integer value as shown below (1,2...)
                  1 = 62.5Khz
                  2 = 238Khz
                  3 = 268Khz
                  4 = 312Khz
                  5 = 375Khz
StartAddr      : Start Address of I2C Slave to Read/Write
Start          : 1 - Generates Start condition
                : 0 - Does not generate Start condition
Stop          : 1 - Generates Stop condition
                : 0 - Does not generate Stop condition
Nack          : 1 - Generates NACK condition for the last byte of the transfer
                : 0 - Does not generate NACK condition

Operation      : Write
Usage          : ./i2cBridging VID(Hex) PID(Hex) Operation(0x01) SlaveAddr ClockRate StartAddr Length Data
Example        : ./i2cBridging 0x0424 0x2734 0x01 0x50 1 0x00 4 0x11 0x22 0x33 0x44

Operation      : Read
Usage          : ./i2cBridging VID(Hex) PID(Hex) Operation(0x00) SlaveAddr ClockRate StartAddr Length
Example        : ./i2cBridging 0x0424 0x2734 0x00 0x50 1 0x00 4

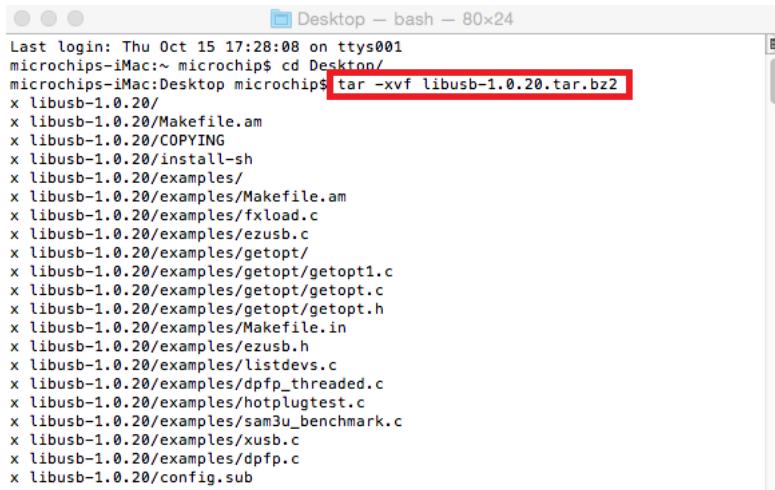
Operation      : Transfer: Write
Usage          : ./i2cBridging VID(Hex) PID(Hex) Operation(0x03) SlaveAddr ClockRate StartAddr Length Start(0/1)
                Stop(0/1) Nack(0/1) Data
Example        : ./i2cBridging 0x0424 0x2734 0x03 0x50 1 0x00 4 1 1 0 0x11 0x22 0x33 0x44

Operation      : Transfer: Read
Usage          : ./i2cBridging VID(Hex) PID(Hex) Operation(0x04) SlaveAddr ClockRate StartAddr Length Start(0/1)
                Stop(0/1) Nack(0/1)
Example        : ./i2cBridging 0x0424 0x2734 0x04 0x50 1 0x00 4 1 1 1
  
```

# 6 Appendix A

## 6.1 libusb Installation

1. Download the latest libusb from <http://sourceforge.net/projects/libusb/files/latest/download?source=files>
2. Use command 'tar -xvf <libusb folder>' to untar the package



```
Desktop — bash — 80x24
Last login: Thu Oct 15 17:28:08 on ttys001
microchips-iMac:~ microchip$ cd Desktop/
microchips-iMac:Desktop microchip$ tar -xvf libusb-1.0.20.tar.bz2
x libusb-1.0.20/
x libusb-1.0.20/Makefile.am
x libusb-1.0.20/COPYING
x libusb-1.0.20/install-sh
x libusb-1.0.20/examples/
x libusb-1.0.20/examples/Makefile.am
x libusb-1.0.20/examples/fxload.c
x libusb-1.0.20/examples/ezusb.c
x libusb-1.0.20/examples/getopt/
x libusb-1.0.20/examples/getopt/getopt1.c
x libusb-1.0.20/examples/getopt/getopt.c
x libusb-1.0.20/examples/getopt/getopt.h
x libusb-1.0.20/examples/Makefile.in
x libusb-1.0.20/examples/ezusb.h
x libusb-1.0.20/examples/listdevs.c
x libusb-1.0.20/examples/dpfp_threaded.c
x libusb-1.0.20/examples/hotplugtest.c
x libusb-1.0.20/examples/sam3u_benchmark.c
x libusb-1.0.20/examples/xusb.c
x libusb-1.0.20/examples/dpfp.c
x libusb-1.0.20/config.sub
```

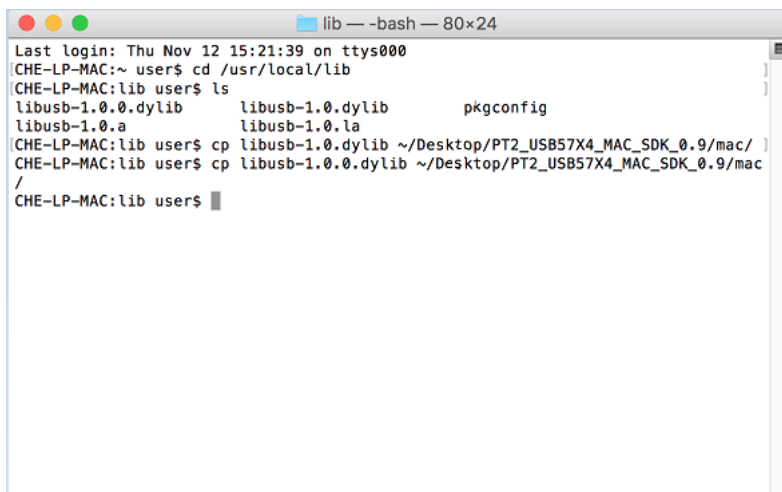
3. Refer to 'INSTALL' file in the directory for installation procedure

## 7 Appendix B

## 7.1 Build the static library and the applications

Follow the steps listed below to build the PT2 static library and applications

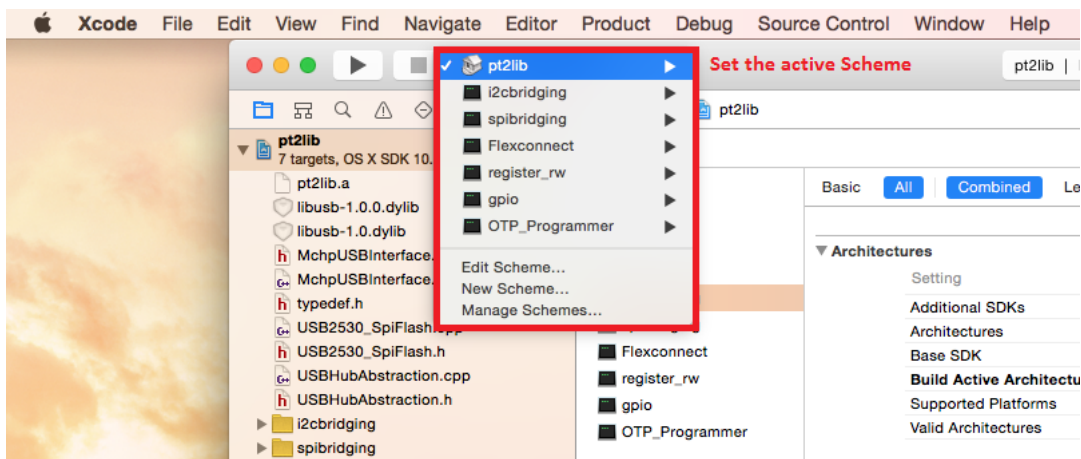
1. Download and install Xcode 6.3.2 or later on Mac from <https://developer.apple.com/support/xcode/> or download from Mac App store with OS X 10.6.6 or later
2. Open pt2lib.xcodeproj file from mac folder
3. Copy libusb dynamic library (libusb-1.0.0.dylib and libusb-1.0.dylib) from '/usr/local/lib/' to the mac folder '\$SDK\_DIR/mac/'



```

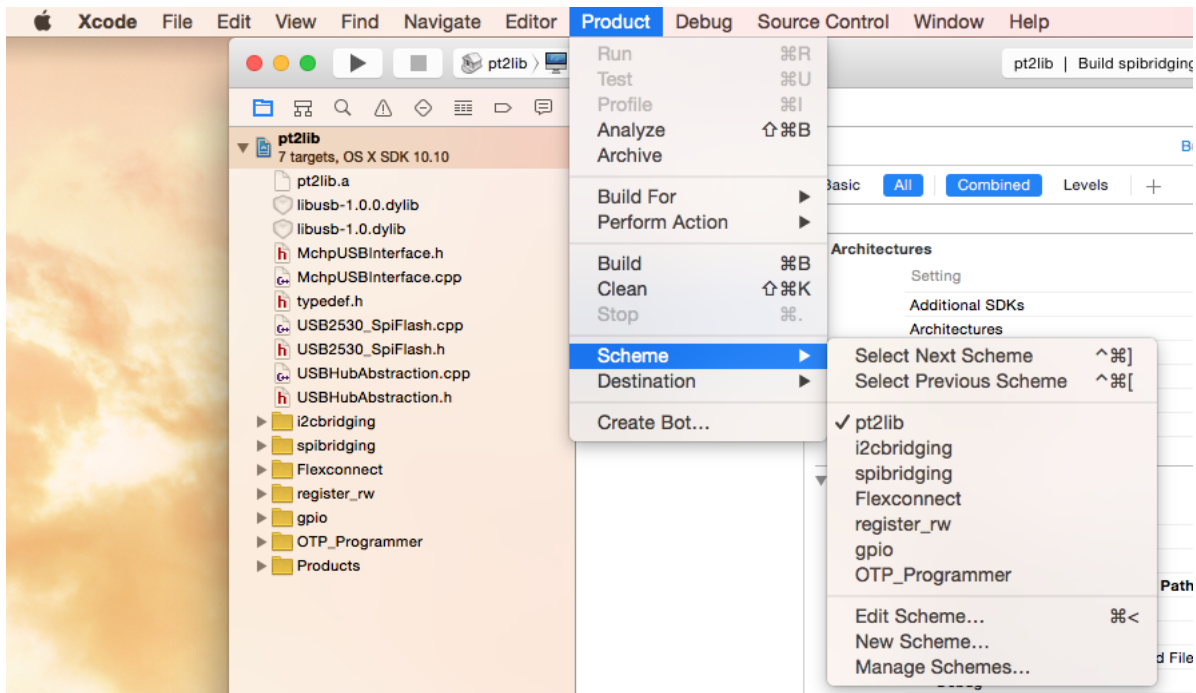
lib — -bash — 80x24
Last login: Thu Nov 12 15:21:39 on ttys000
CHE-LP-MAC:~ user$ cd /usr/local/lib
CHE-LP-MAC:lib user$ ls
libusb-1.0.0.dylib    libusb-1.0.dylib    pkgconfig
libusb-1.0.a          libusb-1.0.la
CHE-LP-MAC:lib user$ cp libusb-1.0.dylib ~/Desktop/PT2_USB57X4_MAC_SDK_0.9/mac/
CHE-LP-MAC:lib user$ cp libusb-1.0.0.dylib ~/Desktop/PT2_USB57X4_MAC_SDK_0.9/mac/
CHE-LP-MAC:lib user$
  
```

4. "Set the active Scheme" drop-down button at the top of the project window or Product >Scheme option is used to select active Scheme



(Or)





5. Build the project

a) Static library file pt2lib.a will be created in mac folder

'\$SDK\_DIR/mac/'

b) Application executable file will be created based on the application target selected (Based on scheme selected) as follows

'\$SDK\_DIR/mac/app/Debug' or '\$SDK\_DIR/mac/app/Release'

6. Navigate to the appropriate directory and run the executable file from terminal

7. For each sample application, "--help" option is provided to find the usage and example

## 8 Appendix C

## 8.1 I2C supported configuration Modes

Based on Configuration resistor encoding, there are 6 configuration modes available in USB5734, each mode sets to different functionalities as mentioned in below table.

Configuration mode	I2C Bridging
Mode 1 : LINX_BUS	Supported
Mode 2: FLEXMODE	Supported
Mode 3: CONNECT	Supported
Mode 4: Not Applicable	Not Applicable
Mode 5: BATTERY_CHARGING	Not Applicable
Mode 6: FULL_UART	Supported