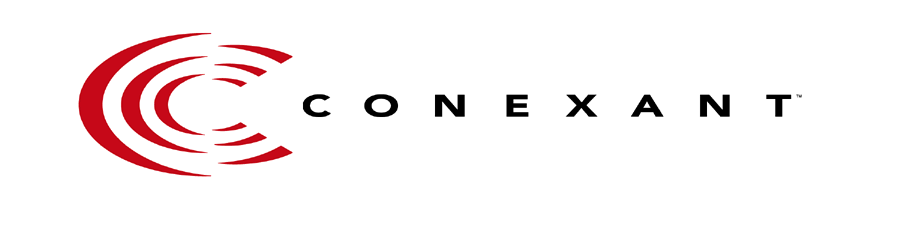
Conexant Audio Framework

CxAudioHidLib Overview



This document contains Conexant Secret, Proprietary, and Confidential information. Unauthorized use or disclosure of this information could impact Conexant competitive advantages as well as those of other companies. This document is not to be disclosed outside of Conexant.

Information in this document is subject to change. No patent liability is assumed with respect to the information contained herein. No liability is assumed for damages resulting from the use of information contained herein.

**July,17, 2017 Document Revision 1.2**

**Administrative Information**

Only the forms fields’ variables should be changed. Note that some fields are used to fill other parts of the document e.g. the title page and the header fields.

|  |  |  |
| --- | --- | --- |
| Form Field Variable | Notes | |
| **(double click to change)** | Bookmark *variable* *name* | Description |
| 1.2 | MyCurrRev | *Current Revision* |
| CxAudioHidLib Overview | MyProjectName | *Project or Product Name* |
| July,17, 2017 | MyCurrRevDate | *Current Revision Date* |
| Chunyu | MyCurrRevisor | *Name of Current Revisor(s)* |
| D001 | MyCurrDocNumber | *Present on the bottom of all pages* |
|  |  |  |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Devisor | Date | Comments |
| 1.0 | Chunyu | 2,17, 2017 | Initial version |
| 1.1 | Chunyu | 4, 5, 2017 | Add information for Linux |
| 1.2 | Chunyu | 7,17,2017 | Add information for Android |

**TABLE OF CONTENTS**

1. Scope 4

2. Target Audience 4

3. References 4

4. Definitions, Acronyms, and Abbreviations 4

5. Introduction 5

6. Supported devices 5

7. Supported operating system 5

8. Basic Functionality 5

9. Architecture 5

10. Build Options 7

11. How to Integrate Library into Application 7

11.1. For Mac operating system 8

11.2. For Windows operating system 8

11.3. For Linux operating system 11

11.4. For Android system 11

12. How to use Library to download the firmware patch 12

12.1. How to use the library on Mac, Windows and Linux? 12

12.2. How to use the library on Android device? 12

13. How to run the Demo 13

14. Upgrading the Library 14

15. API guide 14

15.1. Android Library 14

15.2. Mac, Linux and windows Library 15

# Scope

The purpose of this document is to explain how to use Conexant’s firmware update library CxAudioHidLib on Windows, Mac, Linux and Android.

# Target Audience

It is intended for users to have a simple understanding of CxAudioHidLib, help to develop tools for updating firmware patch.

# References

Table 3‑1: References

|  |  |
| --- | --- |
| Name of Document | Description |
|  |  |
|  |  |
|  |  |

# Definitions, Acronyms, and Abbreviations

Table 4‑1

|  |  |
| --- | --- |
| Word, Acronym, or Abbreviation | Description |
|  |  |
|  |  |
|  |  |
|  |  |

# Introduction

This document gives the high level overview of the LibCxAudioHidLib library: the basic functionality, how to build the project, how to use the library, how to run the demo and sample code.

# Supported devices

The firmware patch upgrade procedure applies to the following devices:

* CX3198x

# Supported operating system

* Windows XP/Vista/7/8/10
* Mac OS X 10.4+
* Linux (Ubuntu 64bit 14.04+)
* Android 6.0/7.11

# Basic Functionality

The basic functionality of the CxAudioHidLib library is:

* find and open connection to USB Audio device based on the Conexant chipsets using HID
* read/write device registers (if supported by device/firmware)
* read/write NVM (if supported by device/firmware)
* upgrade firmware (if supported by device/firmware)
* program serial number (if supported by device/firmware)
* perform I2C tunneling (if supported by device/firmware)
* etc.

# Architecture

The CxAudioHidLib mainly include Audio Hid Device Factory layer, Conexant Audio Hid Device Layer, Hid Device API Layer.

The Audio Hid Factory layer connects to hid device, get information from the device. If it matches with predefined Conexant hid device information, instantiate the device.

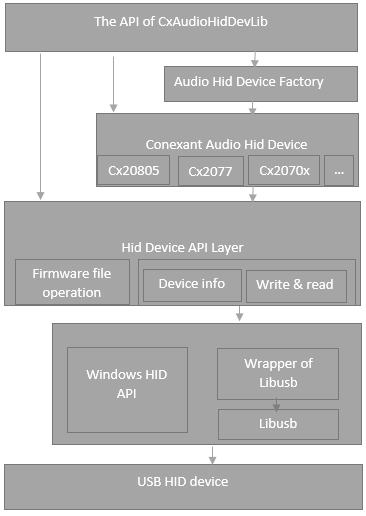
The Conexant Audio Hid device layer include different Conexant audio device. Such as CX20805, CX2077, CX2070x … Every device should provide API about firmware information, update and verify firmware. If we have a new Conexant device to support, we should develop this part.

The Hid device API layer provides different read and write interfaces, device information query interfaces, firmware file information query interfaces.

The block diagram as follow may help you to understand the architecture of the library.

* For Windows CxAudioHidLib, the Hid device API layer has two ways to access the USB HID device.

One is by windows HID API, the other one is by libusb. We can configure it by modifing the Macros in config.h.

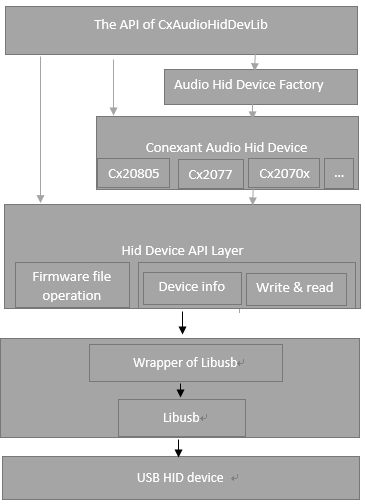


* For Mac, Linux, the difference is that it has to use libusb to access the USB HID device. So ,run the CxAudioHidLib on Mac and Linux devices, please make sure if it has installed libusb; One thing needs pointing out, because we found bugs on the windows and Android libusb,we have to use the modified libusb to fix the issues. Conexant has committed the modifications to libusb github, please download the source code and compile the libusb dynamic library.

For Android, we addcheck the modifications by accessing the link: <https://github.com/libusb/libusb/pull/242>

For Windows, check the modifications by accesing the link: <https://github.com/libusb/libusb/pull/280>

<https://github.com/libusb/libusb/pull/281>



# Build Options

* The Mac CxAudioHidLib only support 64bits. It supports Debug and Release configurations.
* The Linux CxAudioHidLib has no especial build options.
* For Android device, provides the dynamic library for arm64-v8a and armeabi-v7a.
* The Windows CxAudioHidLib supports Debug and Release configurations and 32-bit (Win32) and 64-bit (x64) platforms. The following table summarizes the library file name depending on the configuration/platform combination:

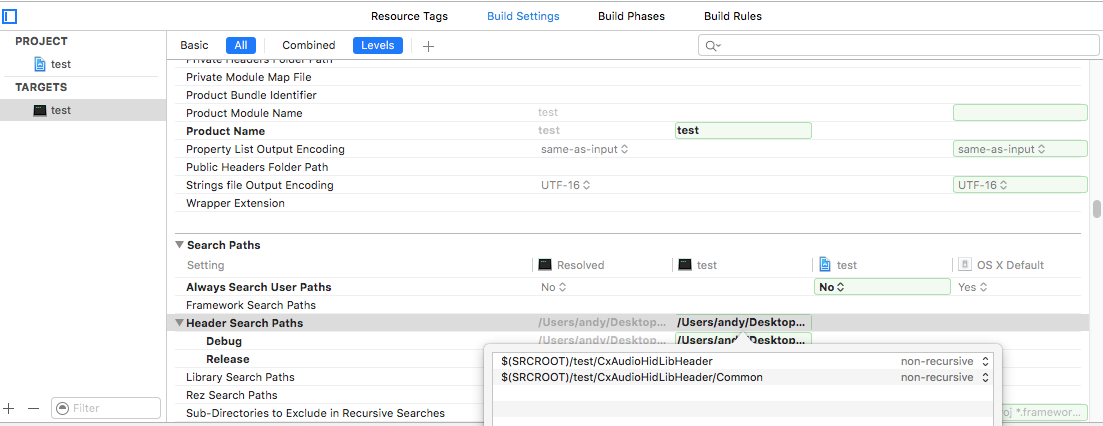
|  |  |  |
| --- | --- | --- |
|  | **Win32** | **x64** |
| **Debug** | CxAudioHidLib32d.lib | CxAudioHidLib64d.lib |
| **Release** | CxAudioHidLib32.lib | CxAudioHidLib64.lib |

# How to Integrate Library into Application

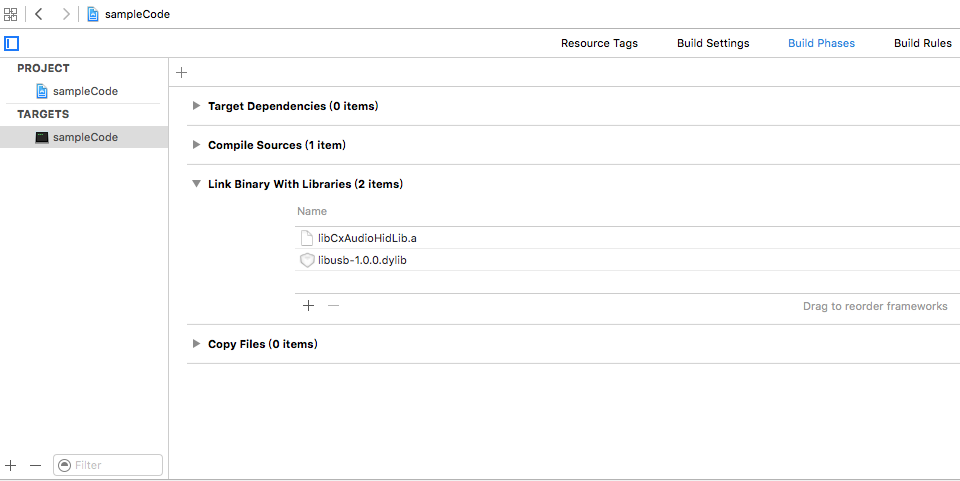
* For convenience, all files (header and binary) needed for integration are located under the same folder (Export). Copy Export folder to the root of the application project, rename it to CxAudioHidLibHeader or what else.

## For Mac operating system

1. Add CxAudioHidLibHeader folder file path to the Header Search path.

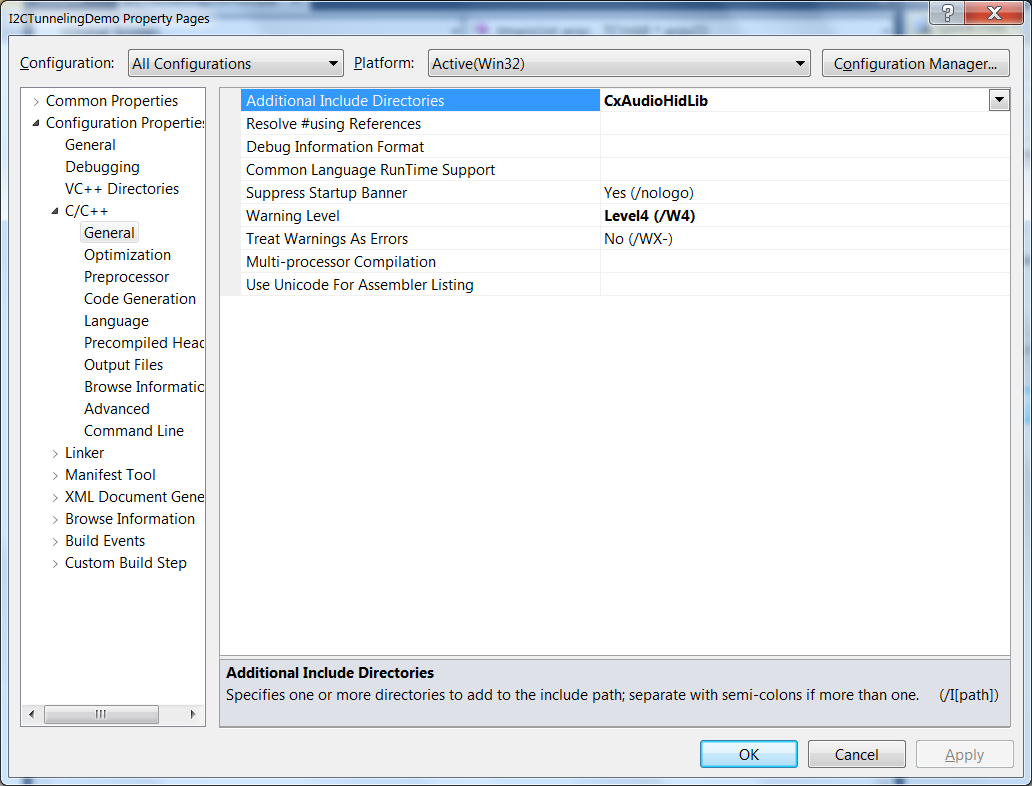


1. Copy the static library to the project folder, add static library libCxAudioHidLib.a to the Link Binary with Libraries in the application project. The static library is based on libusb, please make sure that libusb dynamic library has been added into the project.

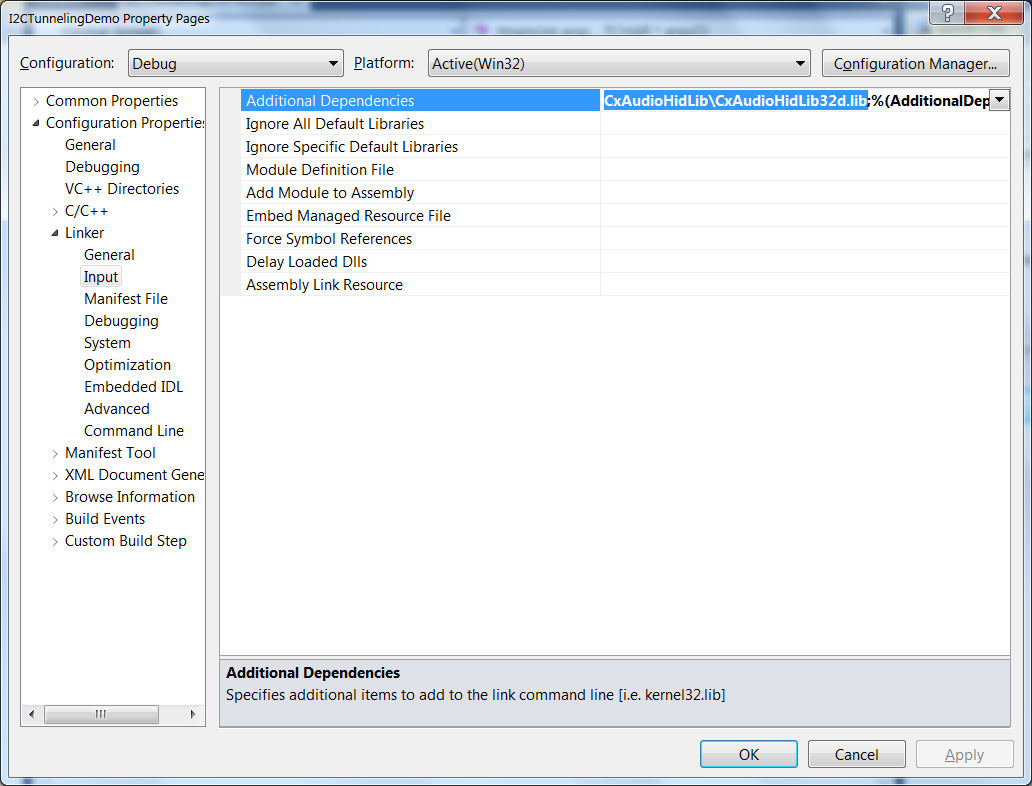


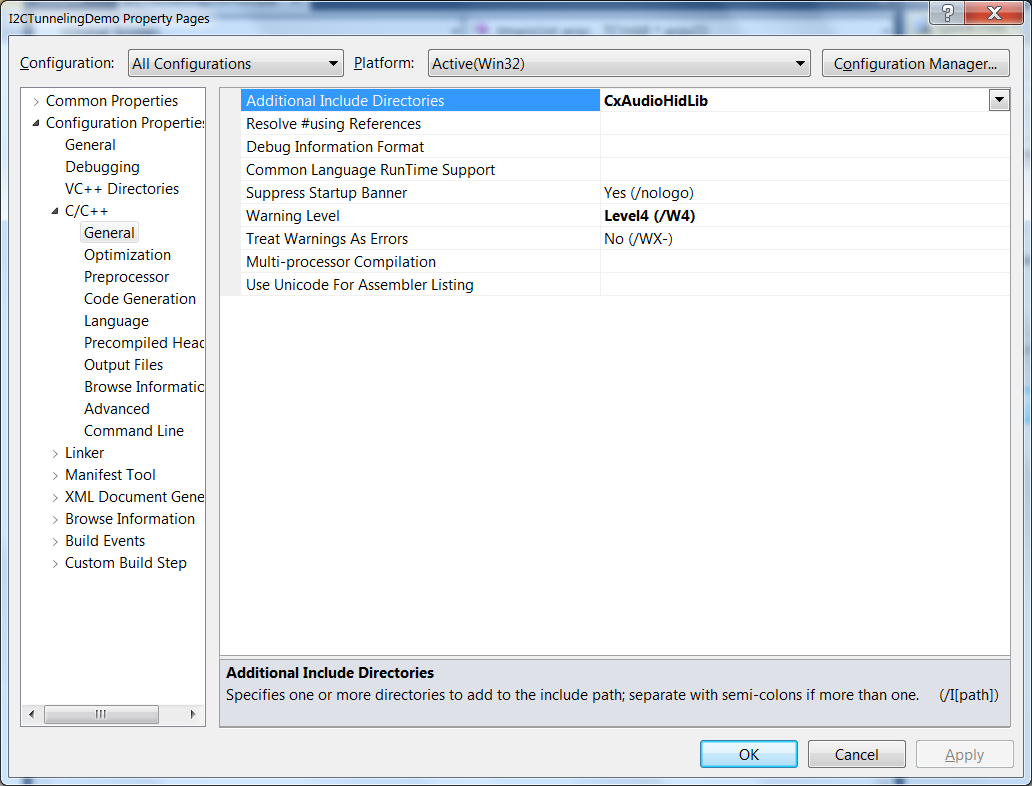
## For Windows operating system

1. Add CxAudioHidLib folder to the Additional Include Directories to find header files for **all** Configurations defined for application project:



1. Add CxAudioHidLib\CxAudioHidLib32**d**.lib library to the Additional Dependencies of the linker for Debug configuration (and CxAudioHidLib\CxAudioHidLib32.lib for Release configuration)





## For Linux operating system

For Linux, one point need to note is that don’t forget to link the libusb when use the libCxAudioHidLib.a. The version of libusb should be libusb-1.0+. On Ubuntu, run the command as follows to install libusb.

sudo apt-get install libusb-dev

sudo apt-get install libusb-1.0.0-dev

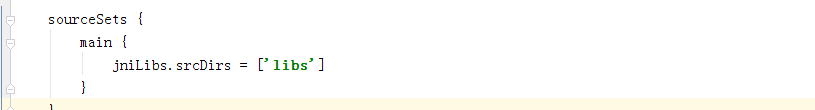
When compile the sample code, compile it like these:

gcc -I../Header/Export -I../Header/linux -I../Header/Export/Common -lstdc++ -c SampleCode.cpp

gcc -o sampleCode SampleCode.o -L../library -lCxAudioHidLib -lusb-1.0 -lstdc++

## For Android system

On Android, integrate the CxAudioHidLib library by Android Studio.

* 1. Under the library directory, it includes the jar package and JNI library for arm64(arm64-v8a), arm(armeabi-v7a). Copy all of them to “libs” folder of your project.
  2. Open your project by Android Studio. Add the jar package to project; Modify the module “build.gradle” to add the JNI library to the project as follows: 
  3. Import the “com.conexant.FirmwareParam ” and “com.conexant.FirmwareUpdate” , then you can use API what the library provide to download the firmware file.

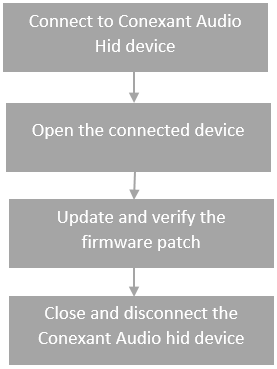
“

# How to use Library to download the firmware patch

## How to use the library on Mac, Windows and Linux?

After Integrated the head files and dynamic library, you should follow the flow of

CxAudioHidLib to use the API to download the firmware patch.

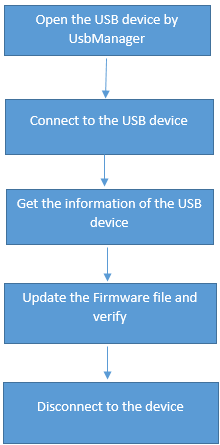


The sample code (check the file sampleCode.cpp) will show the details about how

to connect, open, get information, update and verify the firmware patch.

## How to use the library on Android device?

On Android, it is different from on Linux or Mac because of permissions. When find a USB device, open the device and pass the file descriptor to connect function. If connected successfully and the device is a Conexant device, get information of the device, then update the firmware file. The general flow as follows:



# How to run the Demo

* On Mac, open a terminal, run the Demo command line application. When run the demo, please pass the path and name of the firmware patch. You can run it as follow：

C:\Users\xiec2\Desktop\1483964146(1).png

* On Windows, it is same as on Mac.
* On Linux, run the demo is same as on Mac. The different is that it needs roots permission. It means when use the demo app to download the firmware files, should run it like as follow:

‘sudo ./sampleCode …’

* On Android, please install the demo app. Open the application, plug in Conexant USB device, give the authorization. When connected successfully, the demo app will show the information of Conexant USB device. Click the “browse” button to select the firmware files on Android device. Click the “Update” button, it will update the firmware to Conexant device and verify. If download successfully, it will pop up a toast to show the result.

# Upgrading the Library

In case a new version of CxAudioHidLib library is provided, remove all files from the Header folder under the root of your application and replace by the files from a new version of the library.

# API guide

## Android Library

* **public boolean connectSynaDevice** (**int** fileDescriptor)

Parameter: fileDescriptor the file descriptor of the plug in USB device

Return value: If connected the USB device successfully and the device is a Synaptics USB HID device, it will return true; If not, return false.

* **public FirmwareParam** **getSynaDeviceInfo** ()

FirmwareParam is a class for describing the Synaptics USB HID device. It includes:  
private String firmware\_version; //the version of the device’s firmware  
private int usb\_vendor\_id; // USB vendor ID  
private int usb\_product\_id; //USB product ID  
Parameter:

No

Return value:

If connected successful, return a FirmwareParam instance; If not, return null.

* **public boolean downloadFirmware** (String firmwareFileUrl,int option)

Download or verify the firmware.

parameter:

firmwareFileUrl a string, the URL of firmware that you will update to device.

Option the options for downloading and verifying the firmware.

option == 1, means download firmware;

option == 2, means verification;

option == 3, means download firstly, then verify.

Return value:

If the firmware file is suitable for the device. It burns the firmware file successful, verify the it is OK, return true; If not, return false.

* **public int getUpdateFirmwareProgress()**

It is used to get the download progress.

Parameter: No

Return value: It will return the current progress of downloading or verification. The return value is range from 0 to 100.

* **public void disconnectSynaDevice** ()

If the function connectSynaDevice has ever been called, whatever the result, call this function to release the resource.

Parameter: No

Return value: No

* **public boolean DetectNewFirmware(String FileName)**

This function is used for comparing the device’s firmware version and file’s firmware version.

Parameter: String, it indicates the file path and name of the firmware file

Return value: If firmware file is newer, return the true. Else, return false

* **public String fileGetFwVersion(String filePath)**

Parser the firmware file to get the firmware version string.

Parameter: String, it indicates the file path and name of the firmware file

Return value: The firmware version string.

## Mac, Linux and windows Library

No API guide for these platforms currently, please refer to the sample code.