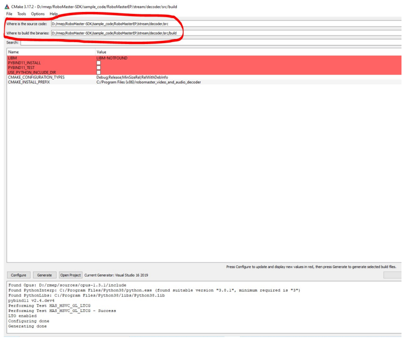
Compiling H264 under windows

1. Install Visual Studio 2019 (with only C++ development options)
2. download and install cmake and python https://www.python.org/ftp/python/3.8.3/python-3.8.3.exe
3. download https://archive.mozilla.org/pub/opus/opus-1.3.1.tar.gz
4. unzip opus and go into the folder where you unzop opus and go into the subdirectory win32/vc2015 in that folder and double click on opus.sln
5. make sure x64 and then build opus. "Build->Build Solution"
6. after build, make sure that is opus.lib in win32/vc2015/x65/Debug/
7. download boost https://dl.bintray.com/boostorg/release/1.73.0/source/boost_1_73_0.zip
8. build boost.
   a) bootstrap
   b) ./b2
   take note of :

The following directory should be added to compiler include paths:
    D:\rmep\sources\boost_1_73_0
The following directory should be added to linker library paths:
    D:\rmep\sources\boost_1_73_0\stage\lib

9. download the following and unzip:
    a) pybind11 https://github.com/pybind/pybind11/archive/v2.5.0.zip
     b) ffmpeg https://ffmpeg.zeranoe.com/builds/
        ffpmeg dev : E.g. Win10 64bit https://ffmpeg.zeranoe.com/builds/win64/dev/ (Select the one is for your OS and architecture) (Win64 - https://ffmpeg.zeranoe.com/builds/win64/dev/ffmpeg-latest-win64-dev.zip)
        ffmpeg shared : E.g. Win10 64bit https://ffmpeg.zeranoe.com/builds/win64/shared/ https://ffmpeg.zeranoe.com/builds/win64/shared/ffmpeg-latest-win64-shared.zip
     c) opus-v0.9-win32 libraries https://archive.mozilla.org/pub/opus/win32/opusfile-v0.9-win32.zip

10. Use git to clone ("download") https://github.com/dji-sdk/RoboMaster-SDK.git (git clone https://github.com/dji-sdk/RoboMaster-SDK.git)
11. run cmake.
12. Make sure "Where is your source code?" is pointed to the folder where you download the Robomaster-SDK and the sample_code/RoboMasterEP/stream/decoder/src.
13. Make sure "Where to build the binaries" is pointed to the folder where you download the Robomaster-SDK and the same_code/RobomasterEP/stream/decoder/src/build


CMake 3.17.2 - D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/src/build
File
Tools
Options
Help
Configure
Generate
Open Project

Where is the source code: D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/src

Where to build the binaries: D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/src/build

Search:

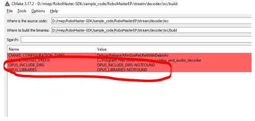
Name	Value
LIBM	LIBM-NOTFOUND
PYBIND11_INSTALL	□
PYBIND11_TEST	□
USE_PYTHON_INCLUDE_DIR	□
CMAKE_CONFIGURATION_TYPES	Debug;Release;MinSizeRel;RelWithDebInfo
CMAKE_INSTALL_PREFIX	C:/Program Files (x86)/robomaster_video_and_audio_decoder

Press Configure to update and display new values in red, then press Generate to generate selected build files.

Current Generator: Visual Studio 16 2019

```
Found Opus: D:/rmep/sources/opus-1.3.1/include
Found PythonInterp: C:/Program Files/Python38/python.exe (found suitable version "3.8.1", minimum required is "3")
Found PythonLibs: C:/Program Files/Python38/libs/Python38.lib
pybind11 v2.4.dev4
Performing Test HAS_MSVC_GL_LTCG
Performing Test HAS_MSVC_GL_LTCG - Success
LTO enabled
Configuring done
Generating done
```

14. click on configure.
15. When you encountered error on opus libraries such as the below:


CMake 3.17.2 - D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/src/build
File
Tools
Options
Help

Where is the source code: D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/src

Where to build the binaries: D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/src/build

Search:

Name	Value
CMAKE_CONFIGURATION_TYPES	Debug;Release;MinSizeRel;RelWithDebInfo
CMAKE_INSTALL_PREFIX	C:/Program Files (x86)/robomaster_video_and_audio_decoder
OPUS_INCLUDE_DIRS	OPUS_INCLUDE_DIRS-NOTFOUND
OPUS_LIBRARIES	OPUS_LIBRARIES-NOTFOUND

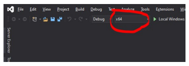
Edit the value of OPUS_INCLUDE_DIRS to the folder where you unzip opus-1.3.1 and the subdirectory include in that folder and edit value of OPUS_LIBRARIES to point to the location where VC compile and stored the opus.lib (step 5)

Once edited, clicked configure and generate again.

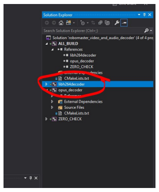
18. Go into the folder that you extract opus-1.3.1 and the include folder in it. Create a new folder opus in that include folder and copy all the files the include into the new fodler opus.

19. Go into Robomaster-SDK/sample_code/RoboMasterEP/stream/decoder/src/build directory and double click on ALL_BUILD.vcxproj. Visual Studio will be launched.

21. Ensure that the architecture is as per the platform on which you are going to use the libraries.

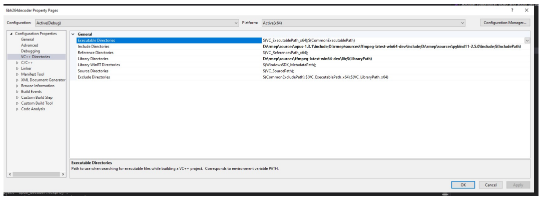


21. click on the lib264decoder in the solution explorer pane and then enter "ALT+ENTER" to go into properties of lib264decoder



22. Make sure the "Include Directories" under VC++ Directories contains the path to:
    A) The include folder in the folder containing opus-1.3.1
    B) the include folder in the folder containing ffpeg-latest-win64-dev
    c) the include folder in the folder containing pybind11-2.5.0

23. Make sure the "Library Directories" under VC++ Directories contains the directory path to:
    A) The lib folder in the folder containing ffmpeg-latest-win64


libh264decoder Property Pages

Configuration: Active(Debug) Platform: Active(x64) Configuration Manager...

Configuration Properties > VC++ Directories

General
- Executable Directories: $(VC_ExecutablePath_x64);$(CommonExecutablePath)
- Include Directories: **D:\rmep\sources\opus-1.3.1\include;D:\rmep\sources\ffmpeg-latest-win64-dev\include;D:\rmep\sources\pybind11-2.5.0\include;$(IncludePath)**
- Reference Directories: $(VC_ReferencesPath_x64);
- Library Directories: **D:\rmep\sources\ffmpeg-latest-win64-dev\lib;$(LibraryPath)**
- Library WinRT Directories: $(WindowsSDK_MetadataPath);
- Source Directories: $(VC_SourcePath);
- Exclude Directories: $(CommonExcludePath);$(VC_ExecutablePath_x64);$(VC_LibraryPath_x64)

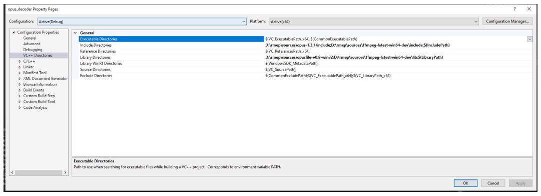
Executable Directories
Path to use when searching for executable files while building a VC++ project. Corresponds to environment variable PATH.

24. click ok.

25. Click on opus_decoder in the solution explorer and enter "ALT+ENTER" to go into properties of opus_decoder.

26. Make sure the "Include Directories" in VC++ Directories contains the following path. If not, edit to add these path.
    a) the include directories in the folder containing opus-1.3.1
    b) the include directories in the folder containing ffmpeg-latest-win64-dev
    c) the include folder in the folder containing pybind11-2.5.0

27. Make sure the "Library Directories" contains the following path. If not, edit to add these path.
    a) the folder containing opusfile-v0.9-win32
    b) the lib directory in the folder containing ffmpeg-latest-win64-dev


opus_decoder Property Pages

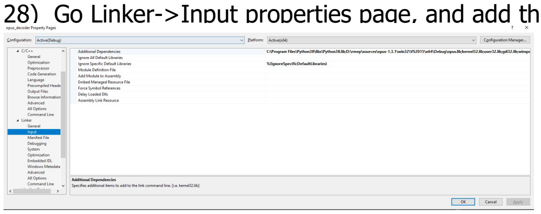
Configuration: Active(Debug) Platform: Active(x64) Configuration Manager...

Configuration Properties > VC++ Directories

General
- Executable Directories: $(VC_ExecutablePath_x64);$(CommonExecutablePath)
- Include Directories: **D:\rmep\sources\opus-1.3.1\include;D:\rmep\sources\ffmpeg-latest-win64-dev\include;$(IncludePath)**
- Reference Directories: $(VC_ReferencesPath_x64);
- Library Directories: **D:\rmep\sources\opusfile-v0.9-win32;D:\rmep\sources\ffmpeg-latest-win64-dev\lib;$(LibraryPath)**
- Library WinRT Directories: $(WindowsSDK_MetadataPath);
- Source Directories: $(VC_SourcePath);
- Exclude Directories: $(CommonExcludePath);$(VC_ExecutablePath_x64);$(VC_LibraryPath_x64)

Executable Directories
Path to use when searching for executable files while building a VC++ project. Corresponds to environment variable PATH.

28) Go Linker->Input properties page, and add the path to the opus.lib file (created in step 5) into the "Additional Dependencies"


opus_decoder Property Pages

Configuration: Active(Debug) Platform: Active(x64) Configuration Manager...

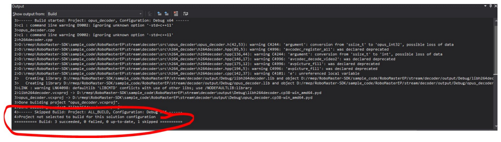
Linker > Input

- Additional Dependencies: **C:\Program Files\Python38\libs\Python38.lib;D:\rmep\sources\opus-1.3.1\win32\VS2015\x64\Debug\opus.lib;kernel32.lib;user32.lib;gdi32.lib;winspo**
- Ignore All Default Libraries
- Ignore Specific Default Libraries: **%(IgnoreSpecificDefaultLibraries)**
- Module Definition File
- Add Module to Assembly
- Embed Managed Resource File
- Force Symbol References
- Delay Loaded Dlls
- Assembly Link Resource

Additional Dependencies
Specifies additional items to add to the link command line. [i.e. kernel32.lib]

29. Click ok.

30. Click Build->Build Solution to build. When successful, you will have libh264decoder.cp38-win_amd64.pyd and opus_decoder.cp38-win_amd64.pyd in the RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\output\Debug directory.



```
3>------ Build started: Project: opus_decoder, Configuration: Debug x64 ------
3>cl : command line warning D9002: ignoring unknown option '-std=c++11'
3>opus_decoder.cpp
2>cl : command line warning D9002: ignoring unknown option '-std=c++11'
2>h264decoder.cpp
3>D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\src\opus_decoder\opus_decoder.h(42,53): warning C4244: 'argument': conversion from 'ssize_t' to 'opus_int32', possible loss of data
2>D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\src\h264_decoder\h264decoder.hpp(85,5): warning C4996: 'avcodec_register_all': was declared deprecated
2>D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\src\h264_decoder\h264decoder.hpp(136,44): warning C4244: 'argument': conversion from 'ssize_t' to 'int', possible loss of data
2>D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\src\h264_decoder\h264decoder.hpp(146,17): warning C4996: 'avcodec_decode_video2': was declared deprecated
2>D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\src\h264_decoder\h264decoder.hpp(175,12): warning C4996: 'avpicture_fill': was declared deprecated
2>D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\src\h264_decoder\h264decoder.hpp(194,5): warning C4996: 'avpicture_fill': was declared deprecated
2>D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\src\h264_decoder\h264decoder.cpp(142,37): warning C4101: 'e': unreferenced local variable
2>   Creating library D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/output/Debug/libh264decoder.lib and object D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/output/Debug/libh264dec
3>   Creating library D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/output/Debug/opus_decoder.lib and object D:/rmep/RoboMaster-SDK/sample_code/RoboMasterEP/stream/decoder/output/Debug/opus_decoder
3>LINK : warning LNK4098: defaultlib 'LIBCMTD' conflicts with use of other libs; use /NODEFAULTLIB:library
2>libh264decoder.vcxproj -> D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\output\Debug\libh264decoder.cp38-win_amd64.pyd
3>opus_decoder.vcxproj -> D:\rmep\RoboMaster-SDK\sample_code\RoboMasterEP\stream\decoder\output\Debug\opus_decoder.cp38-win_amd64.pyd
3>Done building project "opus_decoder.vcxproj".
2>Done building project "libh264decoder.vcxproj".
4>------ Skipped Build: Project: ALL_BUILD, Configuration: Debug x64 ------
4>Project not selected to build for this solution configuration
========== Build: 3 succeeded, 0 failed, 0 up-to-date, 1 skipped ==========
```