windows webrtc编译

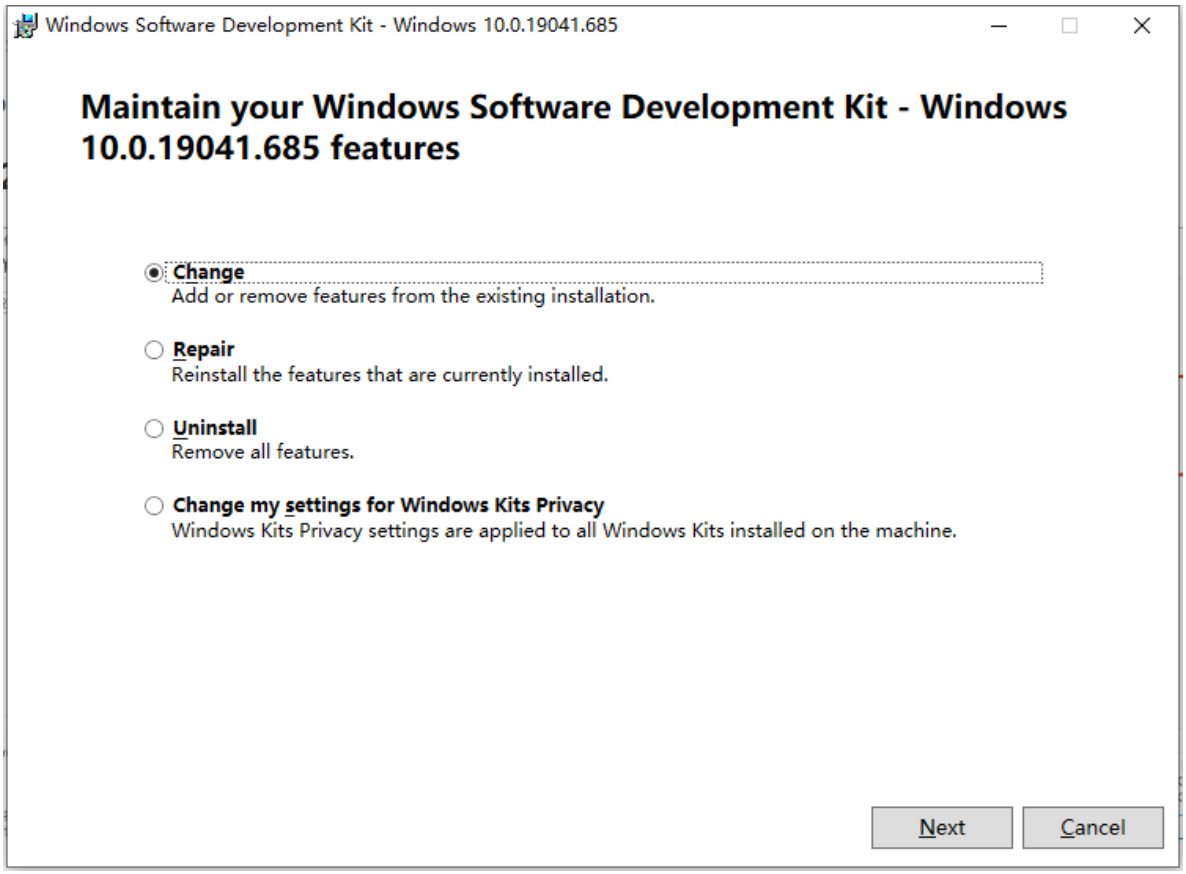
编译工具

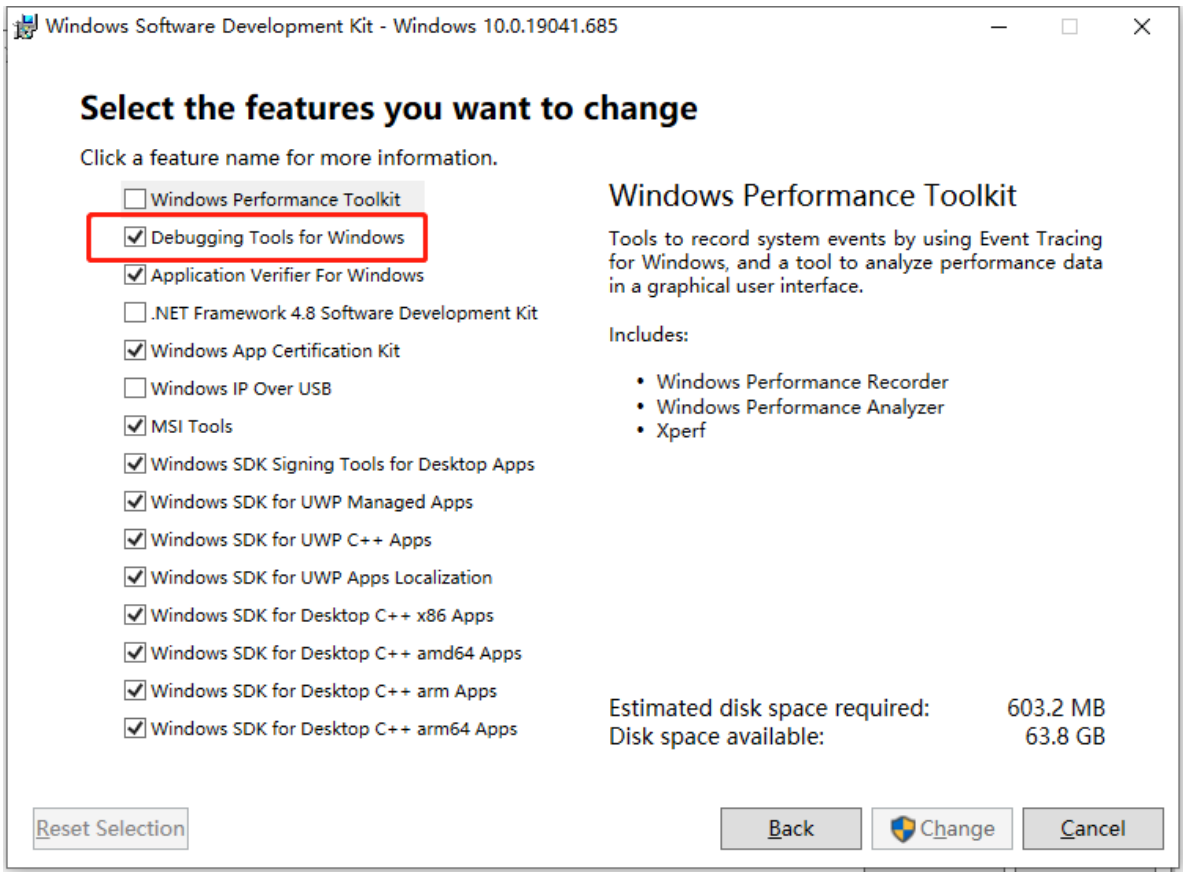
win10　vs2019　winsdk: 10.0.19041.0

webrtc版本m99

vs2019 WinSDK配置

WinSDK安装调试工具：打开控制面板->程序与功能，找到安装的Windows Software Development Kit，鼠标右键->更改

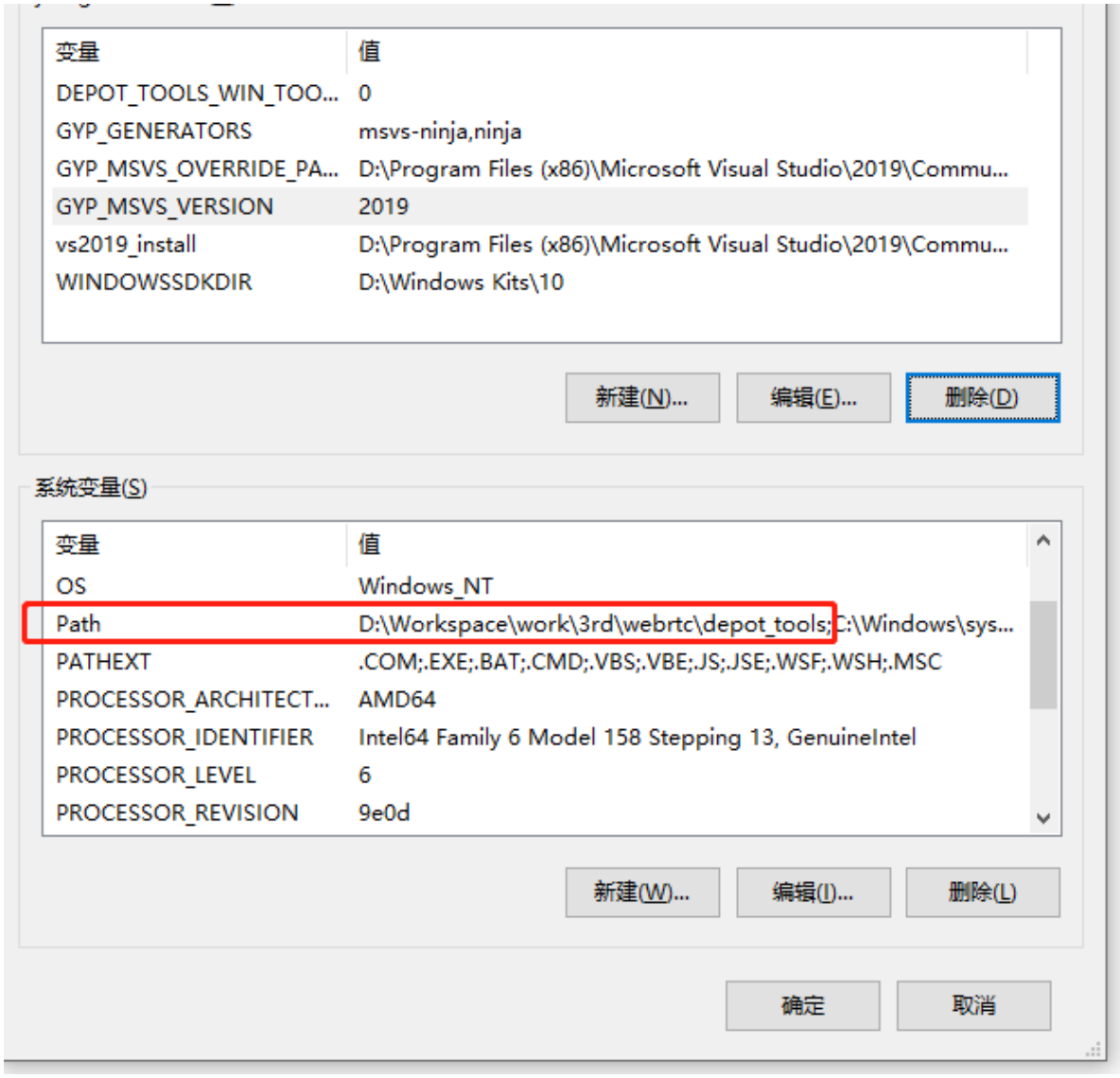




depot_tools安装

[depot_tools下载地址]

安装完之后，将depot_tools配置到系统环境变量



配置环境变量

```
DEPOT_TOOLS_WIN_TOOLCHAIN 0
GYP_GENERATORS msvs-ninja,ninja
GYP_MSVS_OVERRIDE_PATH D:\Program Files (x86)\Microsoft Visual
Studio\2019\Community
GYP_MSVS_VERSION 2019
vs2019_install D:\Program Files (x86)\Microsoft Visual Studio\2019\Community
WINDOWSSDKDIR D:\Windows Kits\10
```

获取webrtc源码

```
配置代理
set http_proxy=127.0.0.1:10809
set https_proxy=127.0.0.1:10809

拉取代码
mkdir webrtc_win
cd webrtc_win
fetch --nohooks webrtc
gclient sync

代码下载完成之后，切换到对应的版本，如果本地没有，可以通过git获取远程对应版本的仓库，如：
git fetch origin refs/branch-heads/4844:m99
```

编译(默认使用clang编译)

```
(1) gn生成编译配置
gn gen out/Default --args="is_debug=true target_os=\"win\" current_os=\"win\"
target_cpu=\"x64\" is_component_build=false rtc_include_tests=false
enable_iterator_debugging=true use_custom_libcxx=false rtc_use_h264=true
proprietary_codecs=true ffmpeg_branding=\"Chrome\"" --ide=vs

参数说明
enable_iterator_debugging: debug版本为true，release版本为false
use_custom_libcxx: 是否使用webrtc自带的libcxx库，默认为true，需要修改为false(避免冲突)

开启H264支持
rtc_use_h264=true
proprietary_codecs=true
ffmpeg_branding=\"Chrome\"

其他参数可通过命令查阅
gn args --list out/Default

(2) ninja编译
ninja -C out/Default

最后在out/Default文件夹中生成all.sln解决方案文件
```

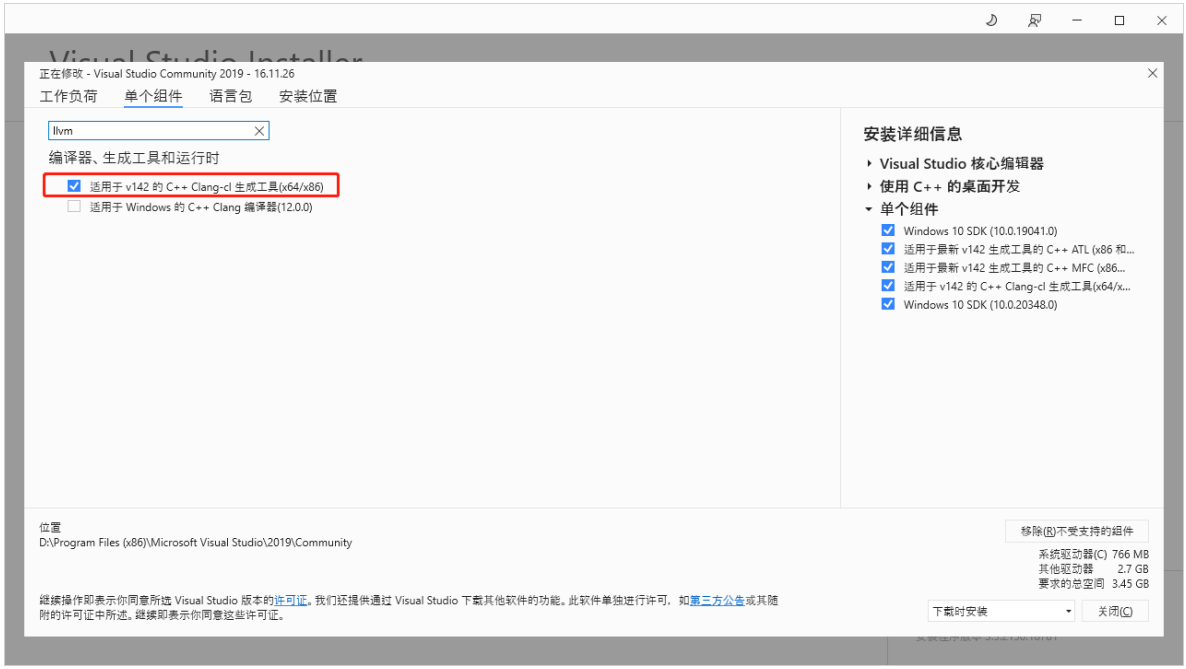
VS2019中引用webrtc.lib库

LLVM工具链安装(使用vs工具链编译有问题)

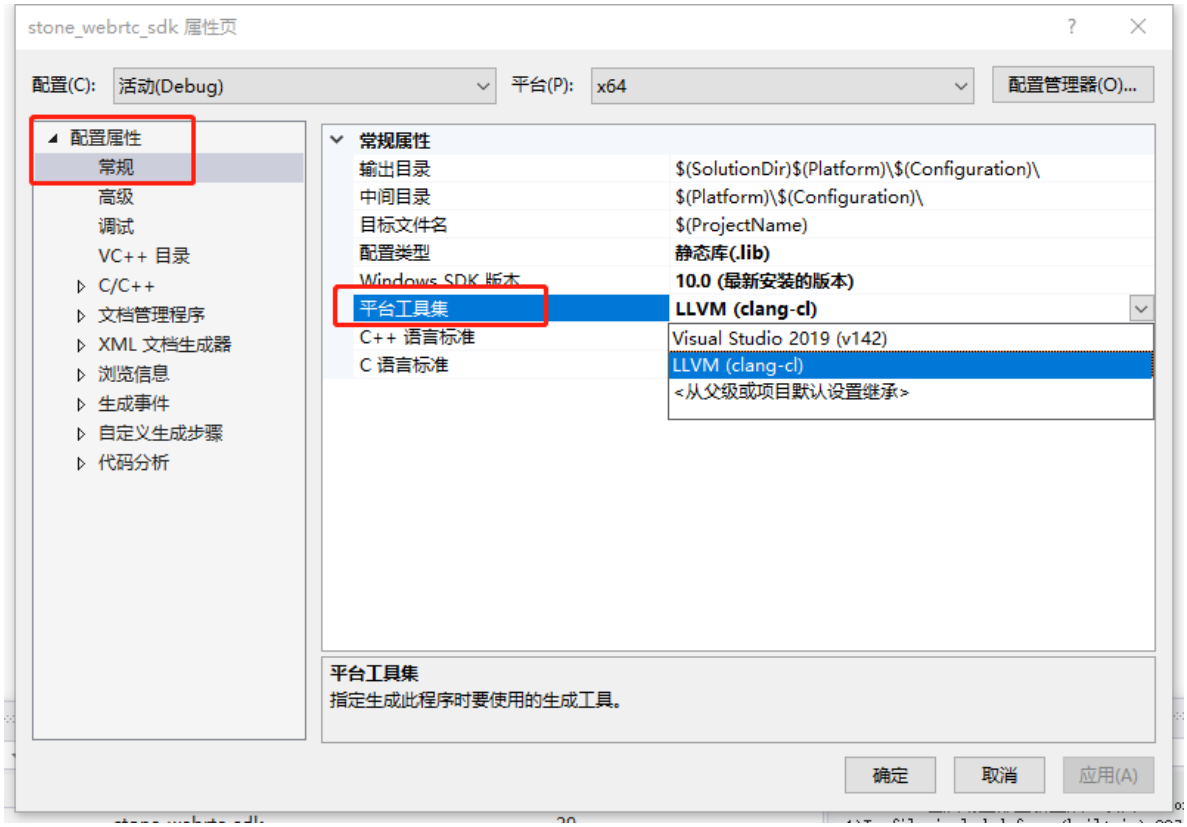
不使用vs提供的clang版本，需要自己安装llvm工具链，对应的版本最好跟webrtc源码中的llvm版本一致，webrtc自带的llvm缺少部分命令工具。WebRTC附带的llvm版本查看：src\third_party\llvm-build\Release+Asserts

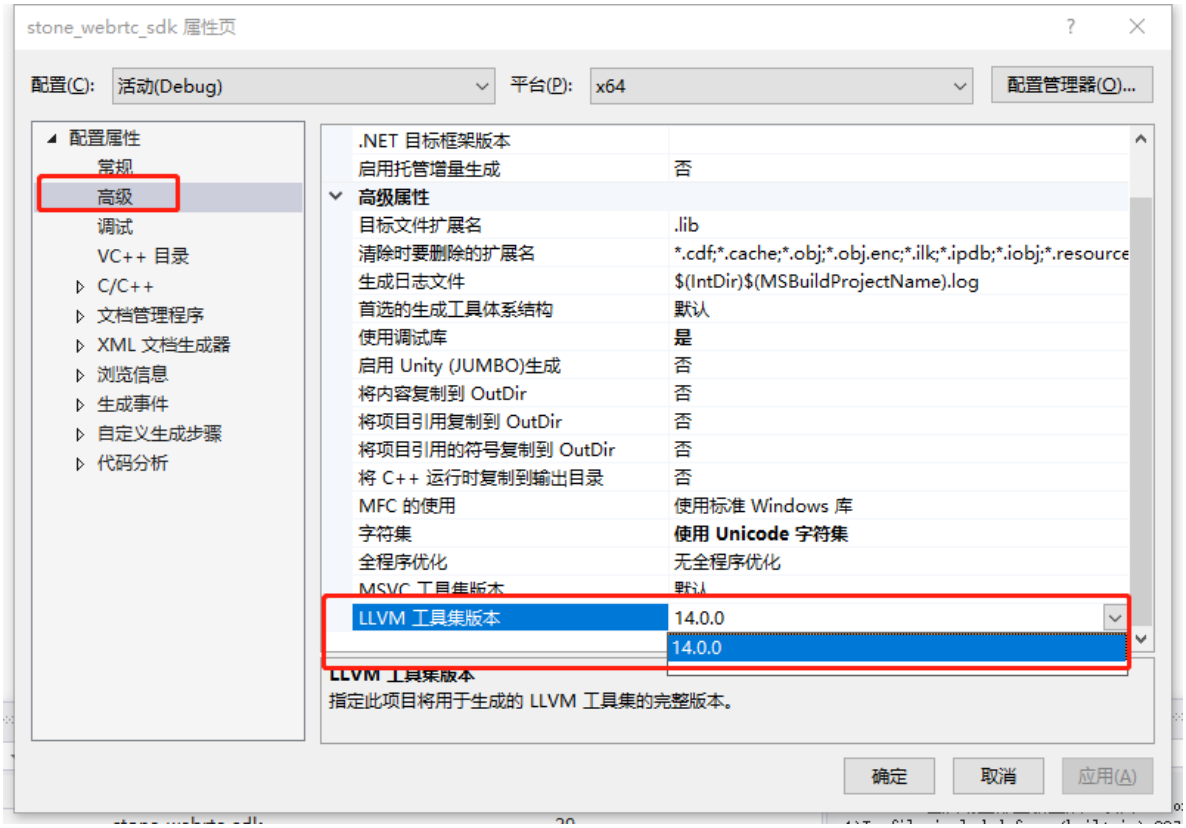
llvm下载地址

vs启用llvm工具链



vs项目配置llvm工具链





如果没有可配置的LLVM版本，在解决方案根目录创建Directory.build.props文件

```
<Project>
  <PropertyGroup>
    <LLVMInstallDir>D:\Workspace\work\3rd\webrtc\win\src\third_party\llvm-
build\Release+Asserts</LLVMInstallDir>
    <LLVMToolsVersion>14.0.0</LLVMToolsVersion>
  </PropertyGroup>
</Project>
```

重新生成WebRTC库

WebRTC默认编译出来的webrtc.lib没有将部分库一块链接进来，可以通过VS将webrtc.lib和未链接的库重新生成，这样使用的时候只用一个库就可以了。未连接的库：

```
third_party\abseil-cpp\absl\flags\marshalling\marshalling.obj
third_party\abseil-cpp\absl\flags\program_name\program_name.obj
third_party\abseil-cpp\absl\flags\flag\flag.obj
third_party\abseil-cpp\absl\flags\flag_internal\flag.obj
third_party\abseil-cpp\absl\flags\commandlineflag\commandlineflag.obj
third_party\abseil-cpp\absl\flags\commandlineflag_internal\commandlineflag.obj
third_party\abseil-
cpp\absl\flags\private_handle_accessor\private_handle_accessor.obj
third_party\abseil-cpp\absl\flags\reflection\reflection.obj
third_party\abseil-cpp\absl\flags\parse\parse.obj
third_party\abseil-cpp\absl\flags\usage\usage.obj
third_party\abseil-cpp\absl\flags\usage_internal\usage.obj
third_party\abseil-cpp\absl\flags\config\usage_config.obj
third_party\jsoncpp\jsoncpp\json_reader.obj
third_party\jsoncpp\jsoncpp\json_value.obj
third_party\jsoncpp\jsoncpp\json_writer.obj
test\field_trial\field_trial.obj
test\video_test_common\test_video_capturer.obj
```

```
test\platform_video_capturer\vcm_capturer.obj
rtc_base\rtc_json\json.obj
```

WebRTC依赖头文件:

```
D:\Workspace\work\3rd\webrtc\win\src
D:\Workspace\work\3rd\webrtc\win\src\out\Default\gen
D:\Workspace\work\3rd\webrtc\win\src\third_party\abseil-cpp
D:\Workspace\work\3rd\webrtc\win\src\third_party\jsoncpp\source\include
D:\Workspace\work\3rd\webrtc\win\src\third_party\jsoncpp\generated
D:\Workspace\work\3rd\webrtc\win\src\third_party\libyuv\include
```

WebRTC提取所有头文件的bat脚本

```
echo off

::该脚本必须放在webrtc源码src目录的上一级目录

:: 创建目标目录
mkdir include

::定义源目录
set srcpath=.\src
::定义目标路径
set dstpath=.\include

:: /s 复制目录和子目录，除了空的。
:: /e 复制目录和子目录，包括空的。
:: /y 禁止提示以确认改写一个现存目标文件。
:: /c 即使有错误也继续执行
:: /h 也复杂隐藏文件和系统文件
:: /r 覆盖只读文件
xcopy %srcpath%\*.h %dstpath%\ /s /e /c /y /h /r

pause
```