**编译licurl**

**关于libcurl**

libcurl是一个免费易用的客户端URL传输库，支持DICT、FILE、FTP、FTPS、GOPHER、GOPHERS、HTTP、HTTPS、IMAP、IMAPS、LDAP、LDAPS、MQTT、POP3、POP3S、RTMP、 RTMPS、RTSP、SCP、SFTP、SMB、SMBS、SMTP、SMTPS、TELNET 和 TFTP。 libcurl 支持 SSL 证书、HTTP POST、HTTP PUT、FTP 上传、基于 HTTP 表单的上传、代理、HTTP/2、HTTP/3、cookies、用户+密码身份验证（Basic、Digest、NTLM、Negotiate、Kerberos）、文件传输恢复 、http 代理隧道等等！

libcurl 是高度可移植的，它在许多平台上构建和工作相同，包括 Solaris、NetBSD、FreeBSD、OpenBSD、Darwin、HPUX、IRIX、AIX、Tru64、Linux、UnixWare、HURD、Windows、Amiga、OS/2、BeOs、Mac OS X、Ultrix、QNX、OpenVMS、RISC OS、Novell NetWare、DOS 等等...

https://github.com/curl/curl

1. 使用VS2022 nmake 编译libcurl静态库
2. 同时将openssl、libssh2、zlib编译成静态库，然后连同libcurl一起进行编译。

在编译libcurl前，需要先将openssl、libssh2、zlib先进行编译。

1. 如何编译openssl
2. 如何编译libssh2
3. 如何编译zlib
4. 如何编译libcurl

**准备工作**

1. **安装Strawberry Perl**

Microsoft Windows的perl环境（构建OpenSSL所必需的）

官方网站: https://strawberryperl.com/

1. **Netwide Assembler(NASM)**

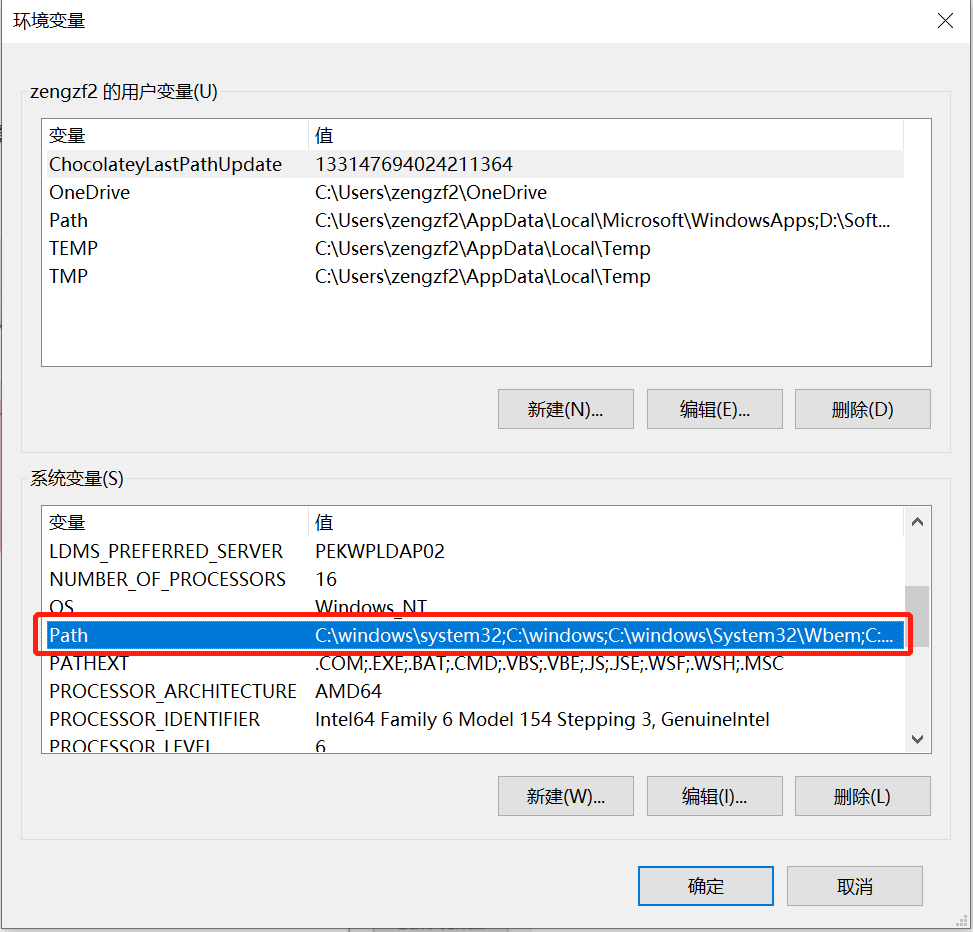
x86 CPU架构的汇编程序（构建OpenSSL所必需的）

官方网站: https://www.nasm.us/

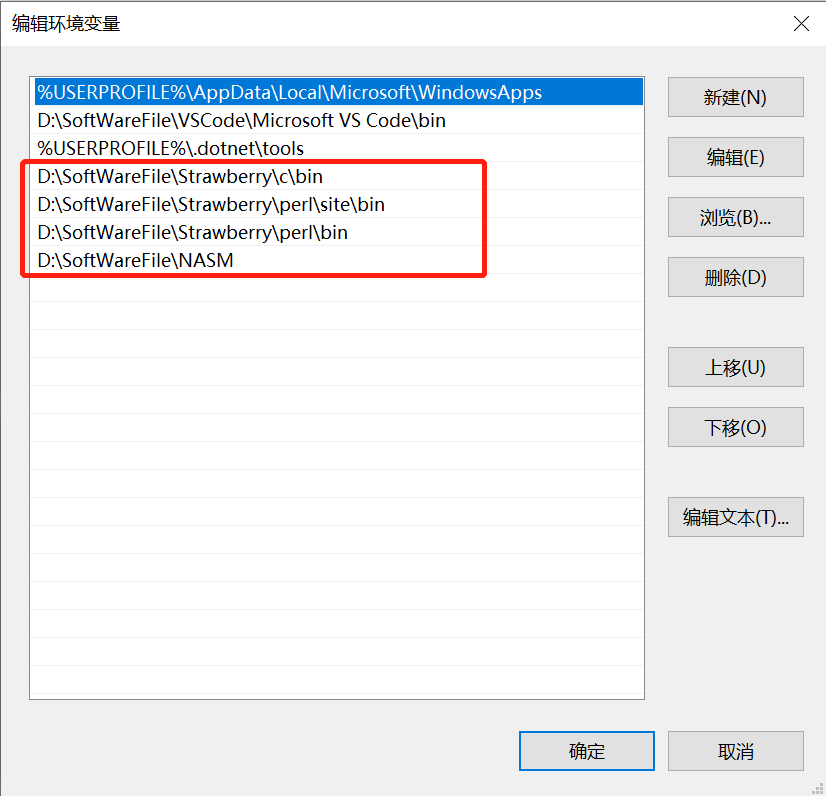
1. **配置环境变量**
2. 搜索“编辑系统环境变量”
3. 点击环境变量



1. 选中系统变量---》编辑



1. 加入如下环境变量



1. **测试是否安装成功Perl&NASM**
2. 打开命令行工具(cmd/powershell)
3. 输入如下指令



**开始进行编译**

1. **编译openssl**
2. Openssl官方网站: <https://www.openssl.org/>
3. 编译32位openssl静态库
4. 打开VS x86 Native Tools Command Prompt for VS2022
5. cd 到openssl源码路径
6. 使用命令创建32位目标文件夹(mkdir openssl32)
7. 使用perl配置环境(perl Configure VC-WIN32 no-asm --prefix=d:/openssl路径/openssl32)
8. 输入指令 nmake
9. 输入指令 nmake install
10. 编译64位openssl静态库

1. 打开VS x64 Native Tools Command Prompt for VS2022

2. cd 到openssl源码路径

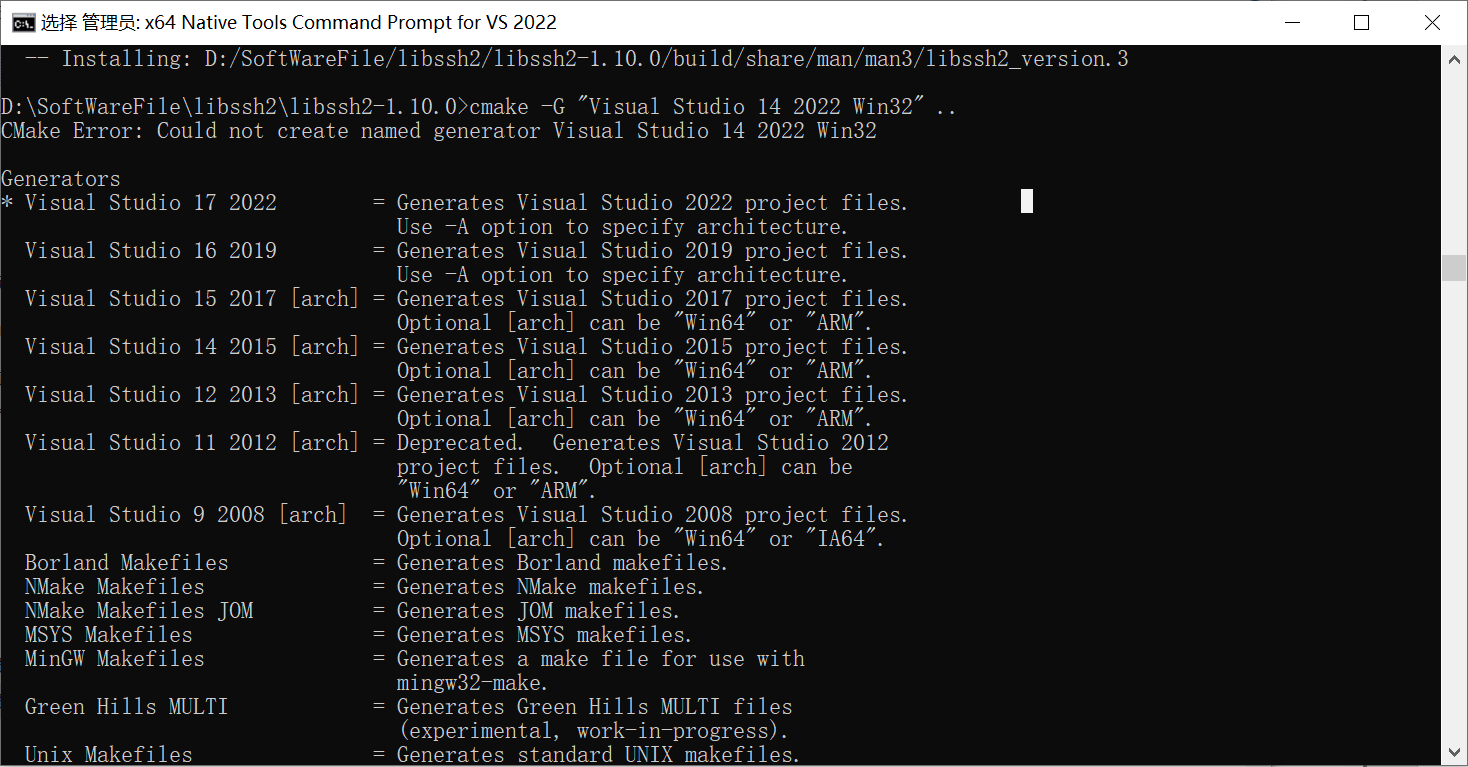
3. 使用命令创建64位目标文件夹(mkdir openssl64)

4. 使用perl配置环境(perl Configure VC-WIN64A no-asm --prefix=d:/openssl路径/openssl64)

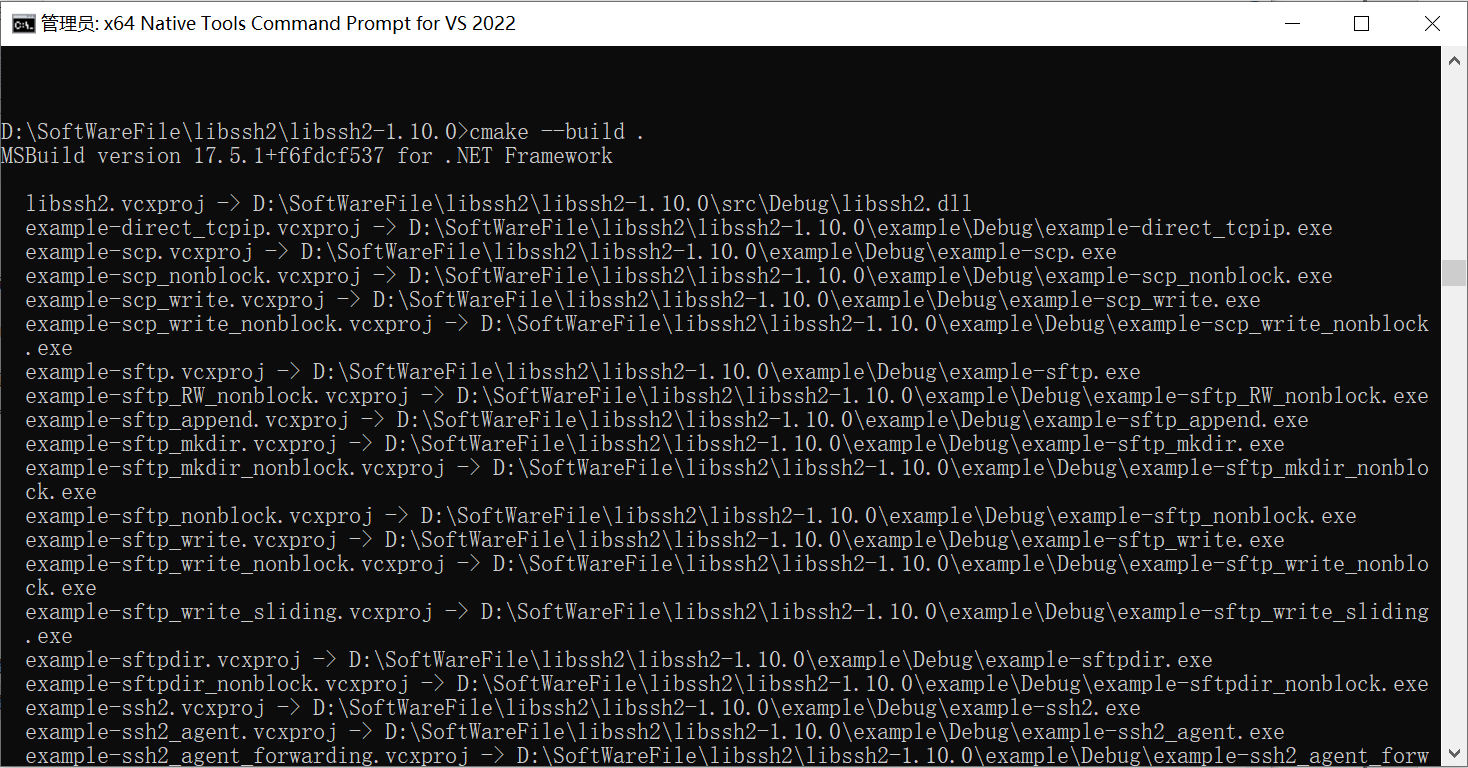
5. 输入指令 nmake

6. 输入指令 nmake install

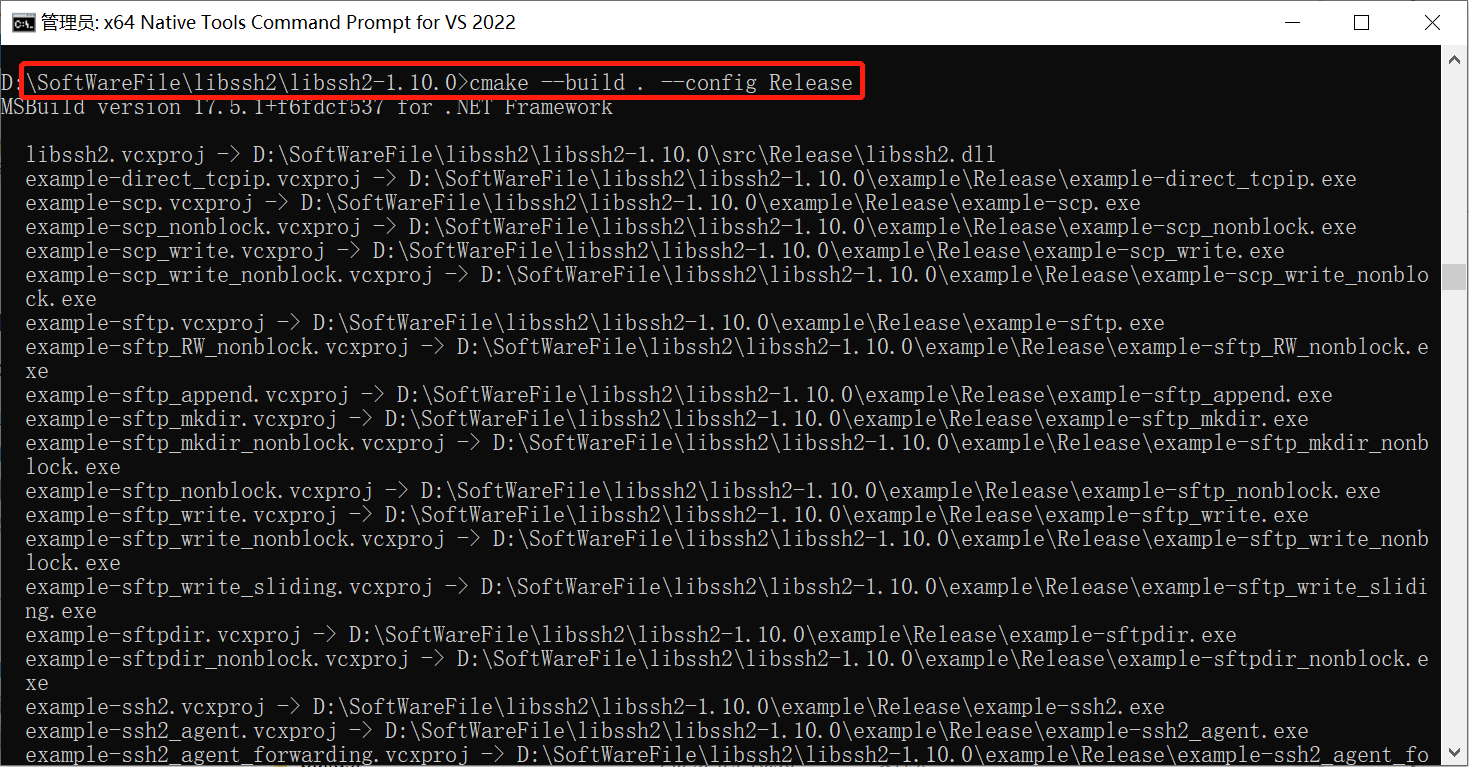
1. 检查编译结果
2. **编译libssh2**
3. libssh2官方网站: <https://www.libssh2.org/>
4. 编译32位libssh2静态库
5. 打开VS x86 Native Tools Command Prompt for VS2022
6. cmake -G "Visual Studio 14 2022 Win32" ..



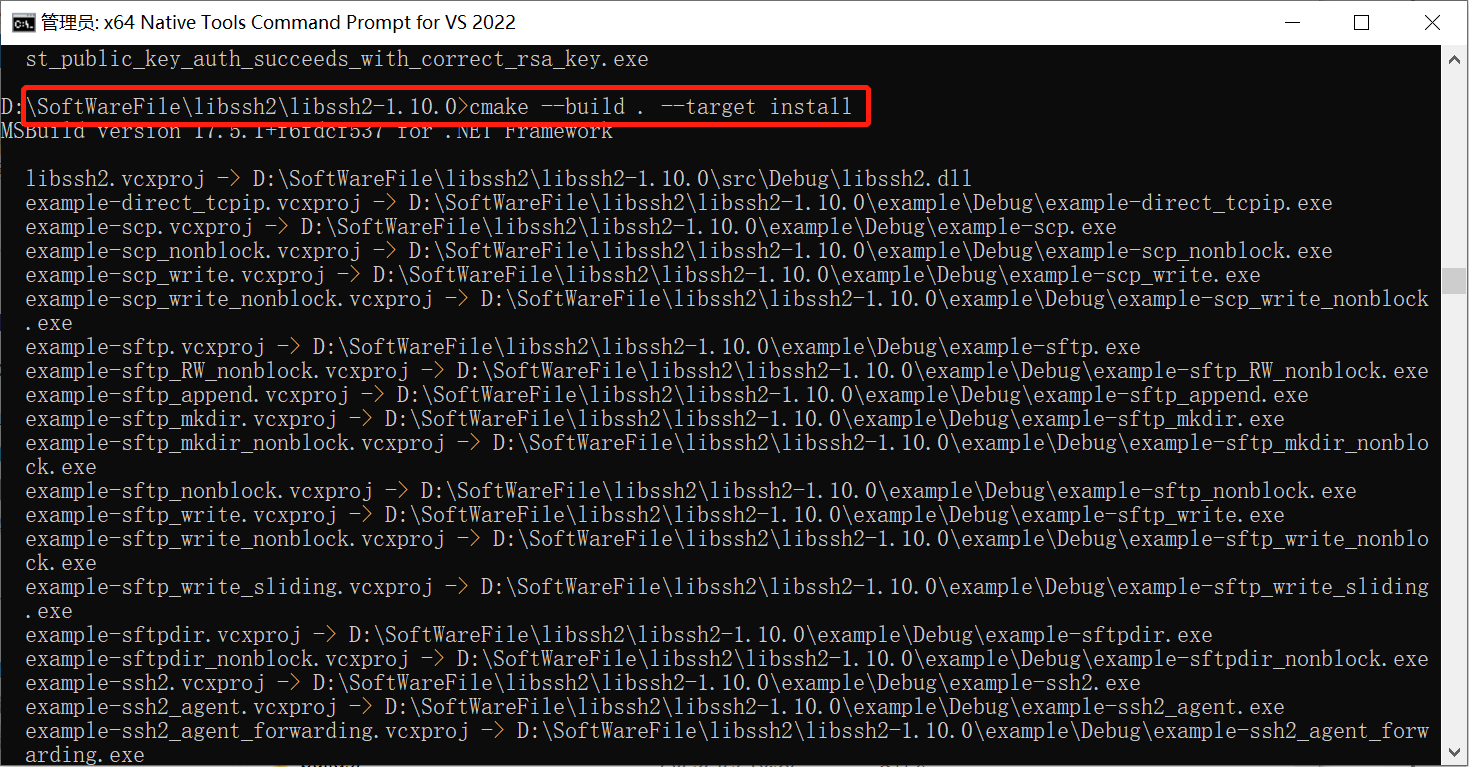
1. cmake --build .



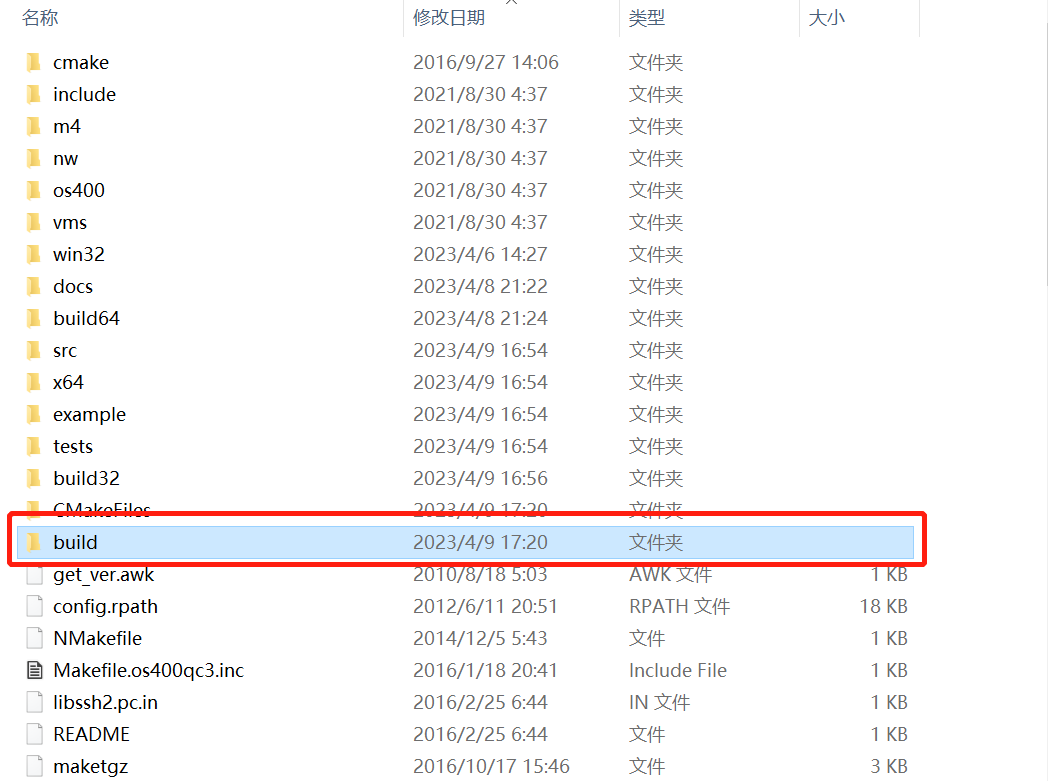
1. cmake --build . --config Release

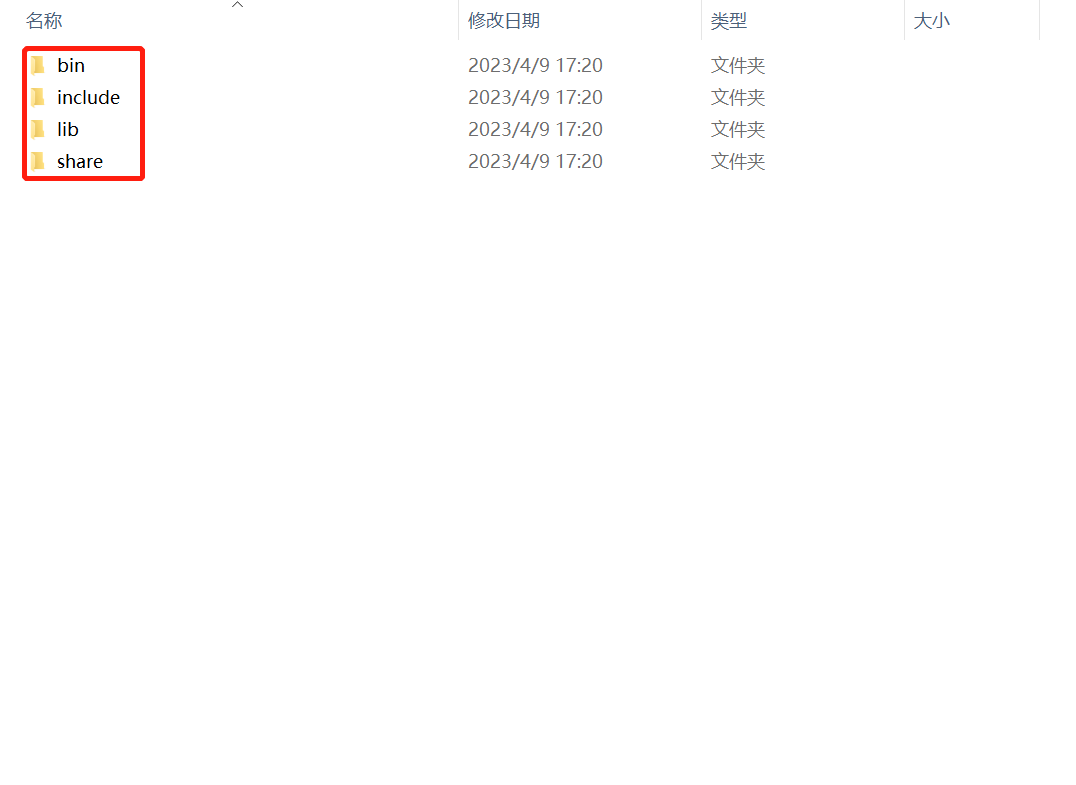


1. cmake --build . --target install



1. 检查是否生成build文件夹和相关文件





Remark:

检查是否在bin文件夹下生成libssh2.dll，lib文件夹下生成libssh2.lib。

1. 编译64位libssh2静态库

1. 打开VS x64 Native Tools Command Prompt for VS2022

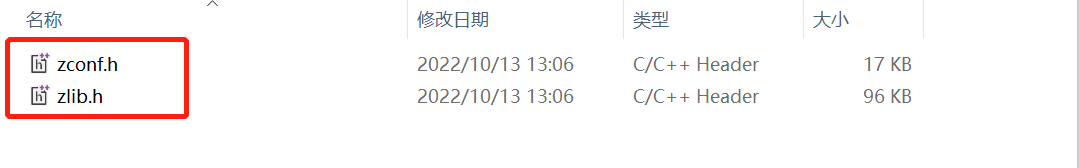
2. cmake -G "Visual Studio 14 2022 Win64" ..

3. cmake --build .

4. cmake --build . --config Release

5. cmake --build . --target install

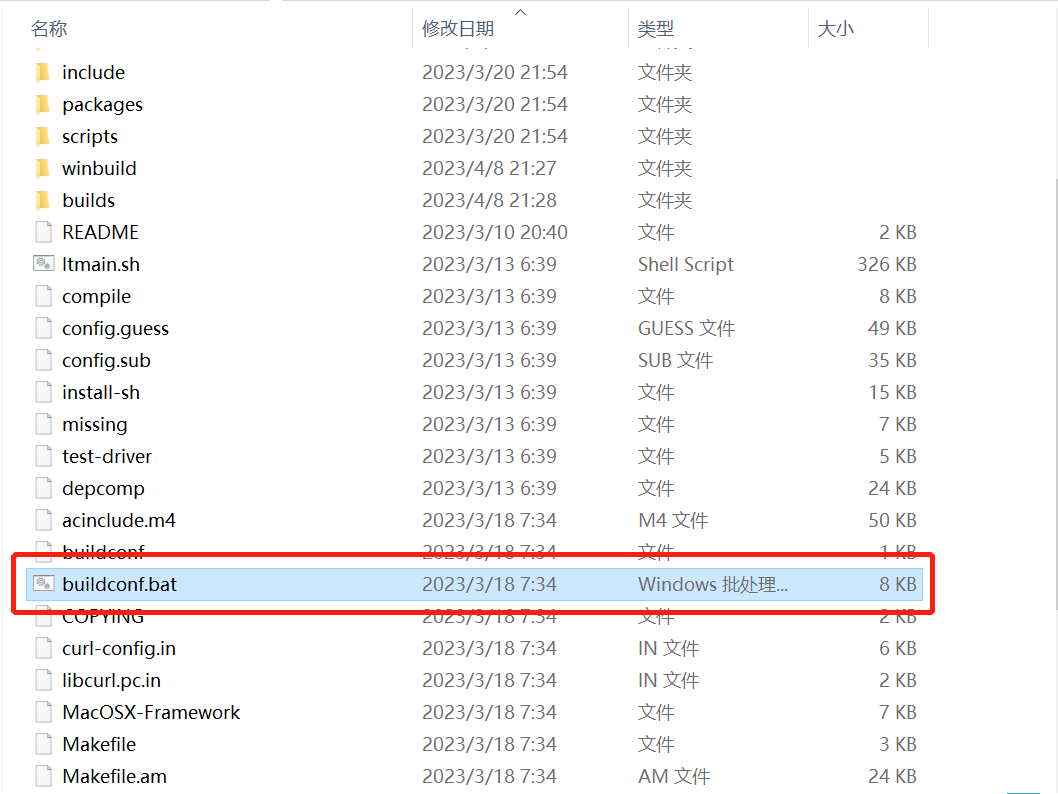
1. **编译zlib**
2. zlib官方网站：<https://zlib.net/>
3. 打开VS x64 Native Tools Command Prompt for VS2022
4. nmake /f win32/Makefile.msc
5. 创建文件夹build
6. 在build中创建include&lib文件夹
7. 将如下头文件拷贝到include文件夹中



1. 将如下文件拷贝到lib文件夹中



1. **编译libcurl**
2. 先运行buildconf.bat



1. 打开VS x64 Native Tools Command Prompt for VS2022
2. 进入到winbuild文件夹目录下
3. nmake /f Makefile.vc MODE=static RTLIBCFG=static vc=14 SSH2\_PATH=D:\SoftWareFile\libssh2\libssh2-1.10.0 SSL\_PATH=D:\SoftWareFile\OpenSSL\openssl-3.1.0\OpenSSLBuild ZLIB\_PATH=D:\SoftWareFile\zlib\zlib-master\ZlibBuild ENABLE\_IDN=no DEBUG=no WITH\_SSL=static WITH\_ZLIB=static WITH\_SSH2=static

Remark:

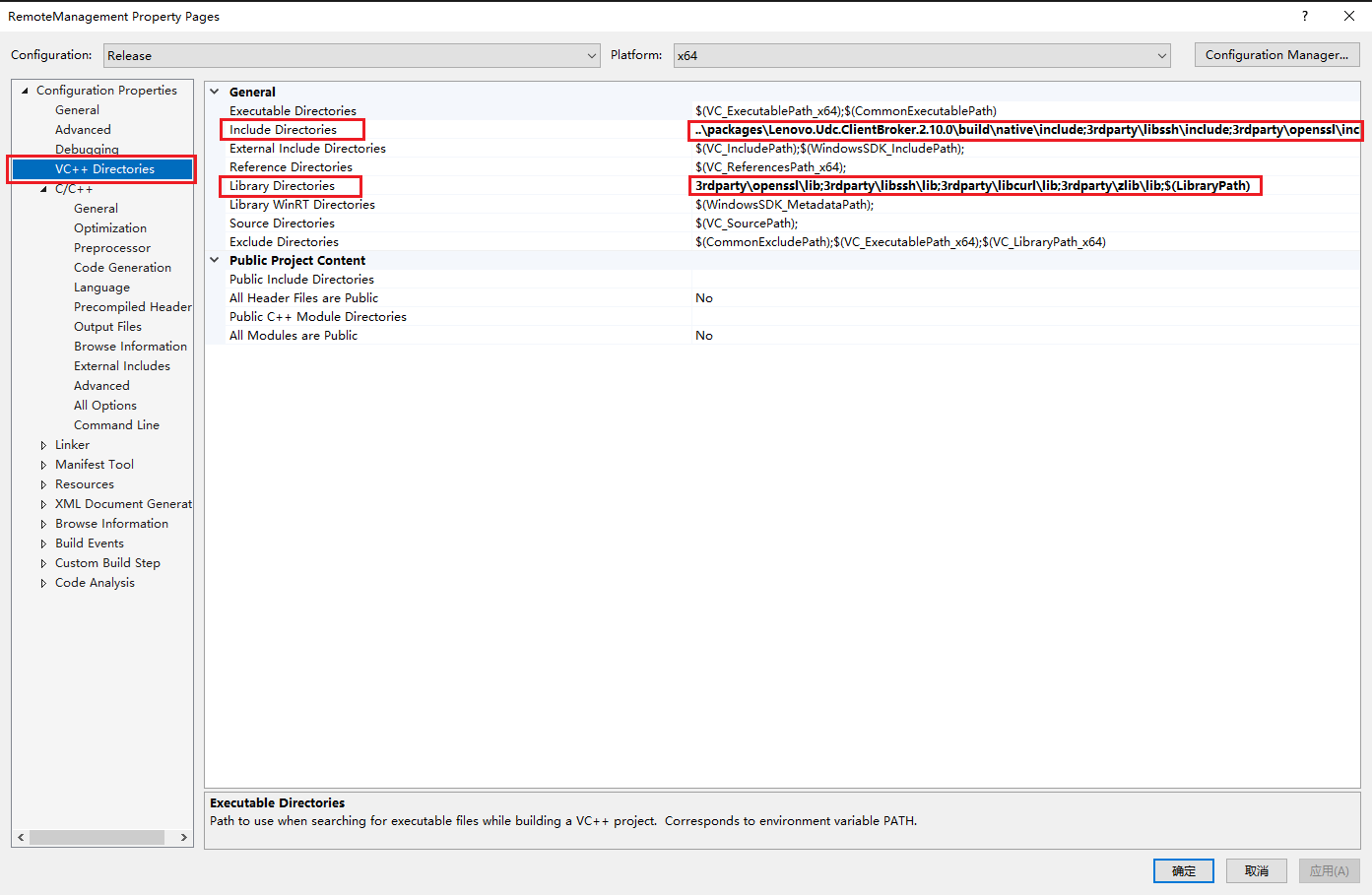
MODE:可选静态库/动态库

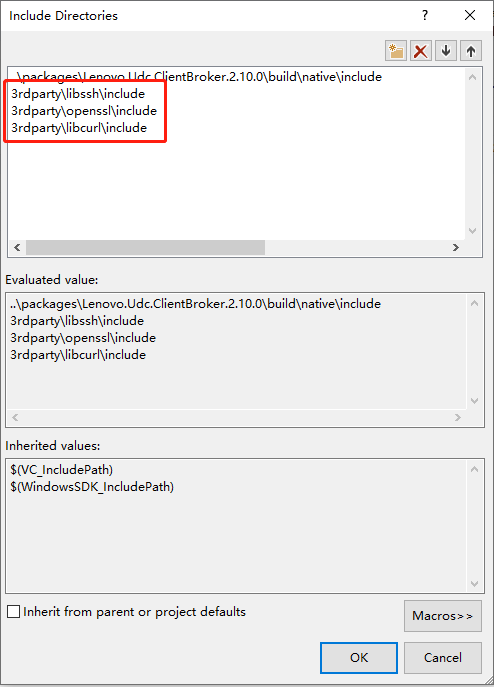
VC:代表编译VC++语言的版本，VC=14 ---->代表VC++14

SSH2\_PATH/ZLIB\_PATH/SSL\_PATH:表示编译结果路径

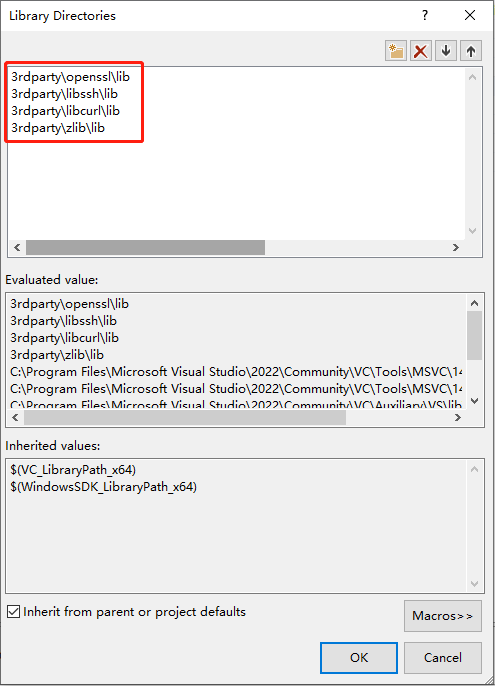
DEBUG:NO表示Release版本，YES表示DEBUG版本

1. **测试libcurl**
2. **设置属性页 --> VC++Directories**
3. **设置Include Directoties(包含头文件文件夹)**

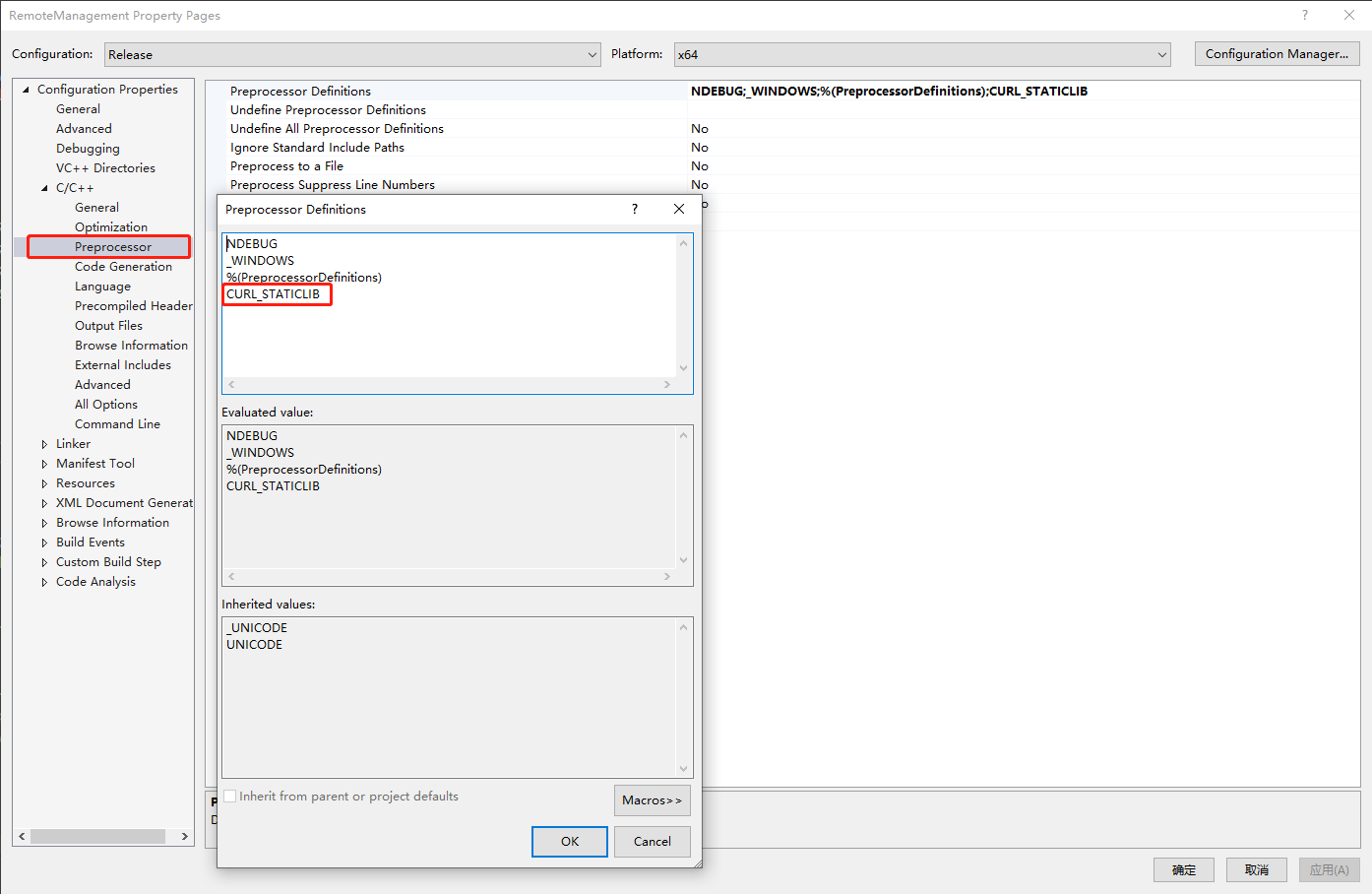
****



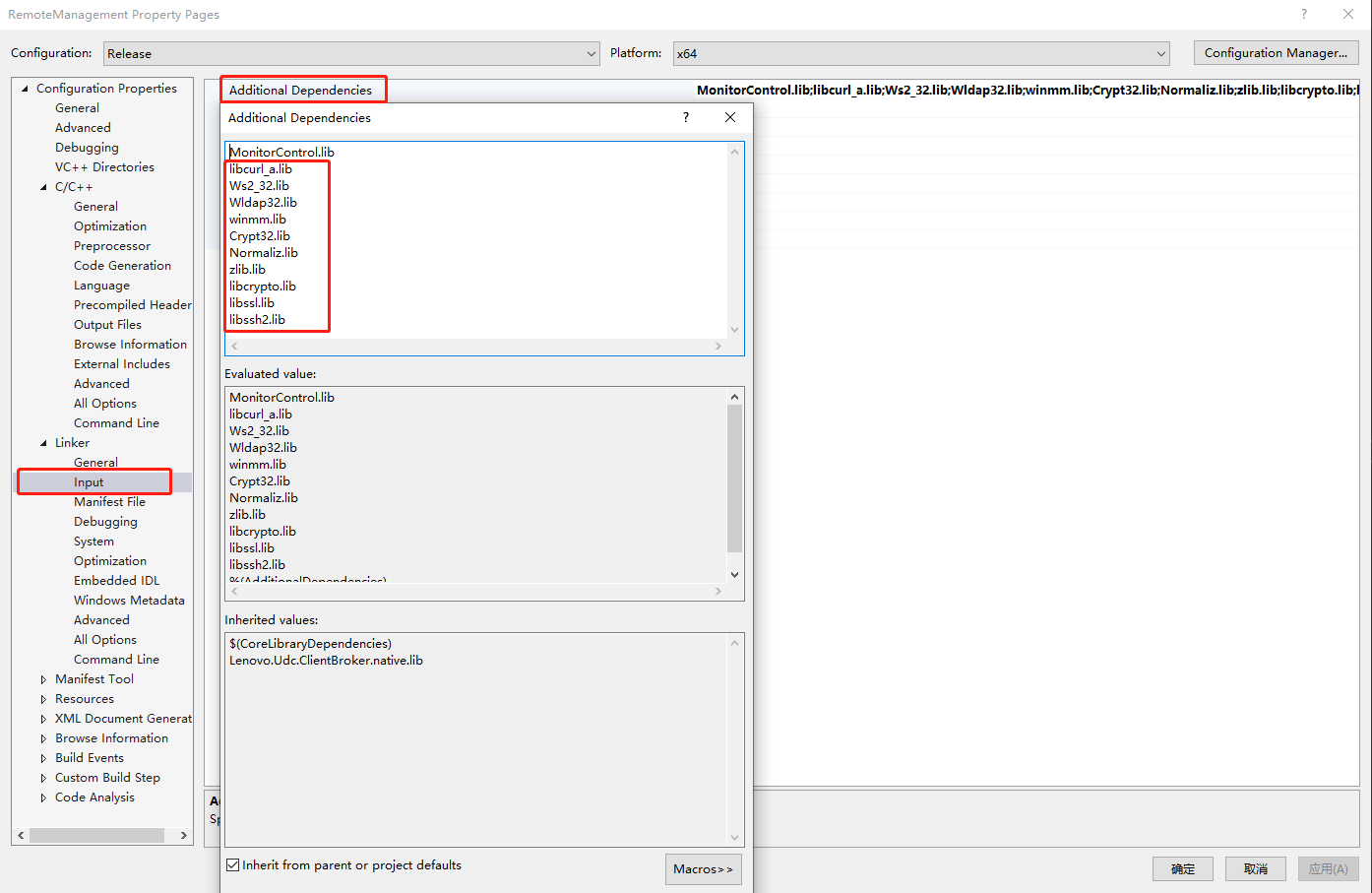
1. **设置Library Directories(包含静态库文件夹)**



1. **C/C++ --> Preprocessor(设置预编译头)**



1. **Linker —> Input (添加静态库列表)**



1. **测试代码**

#include <curl/curl.h>

int main()

{

CURL \*curl;

CURLcode res;

curl = curl\_easy\_init();

if(curl)

{

curl\_easy\_setopt(curl, CURLOPT\_URL, "http://example.com");

res = curl\_easy\_perform(curl);

if(res != CURLE\_OK)

fprintf(stderr, "curl\_easy\_perform() failed: %s\n", curl\_easy\_strerror(res)); curl\_easy\_cleanup(curl);

}

return 0;

}