参考如下

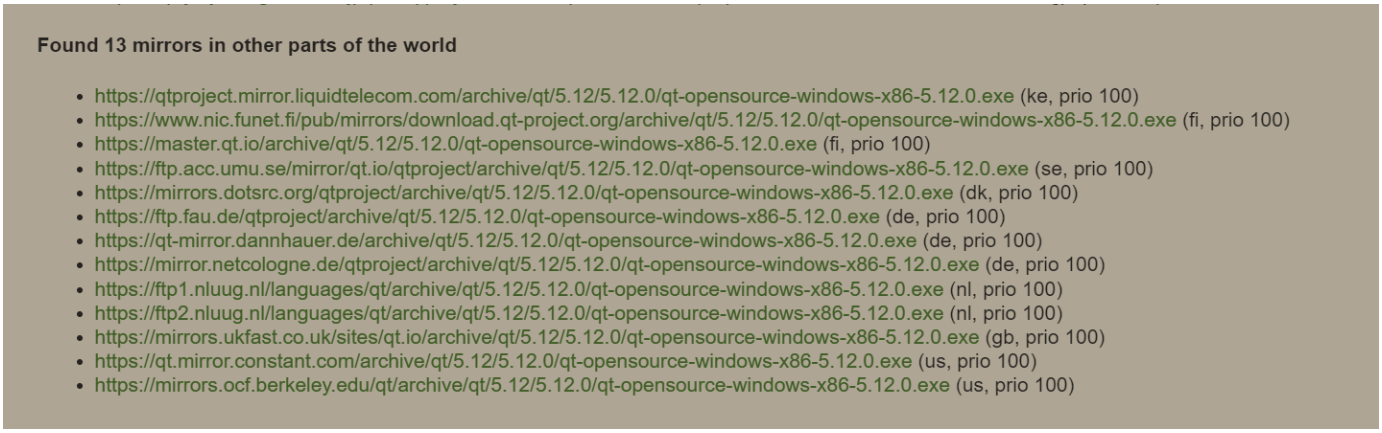
https://blog.csdn.net/qq_35905572/article/details/95797171

1.安装Visual Studio 2022

2.下载安装qt

https://download.qt.io/archive/qt/5.12/5.12.0/

Details



我用的最后一个下载

https://mirrors.ocf.berkeley.edu/qt/archive/qt/5.12/5.12.0/qt-opensource-windows-x86-5.12.0.exe

断网跳过账号验证，重新安装

3.安装vs对应插件

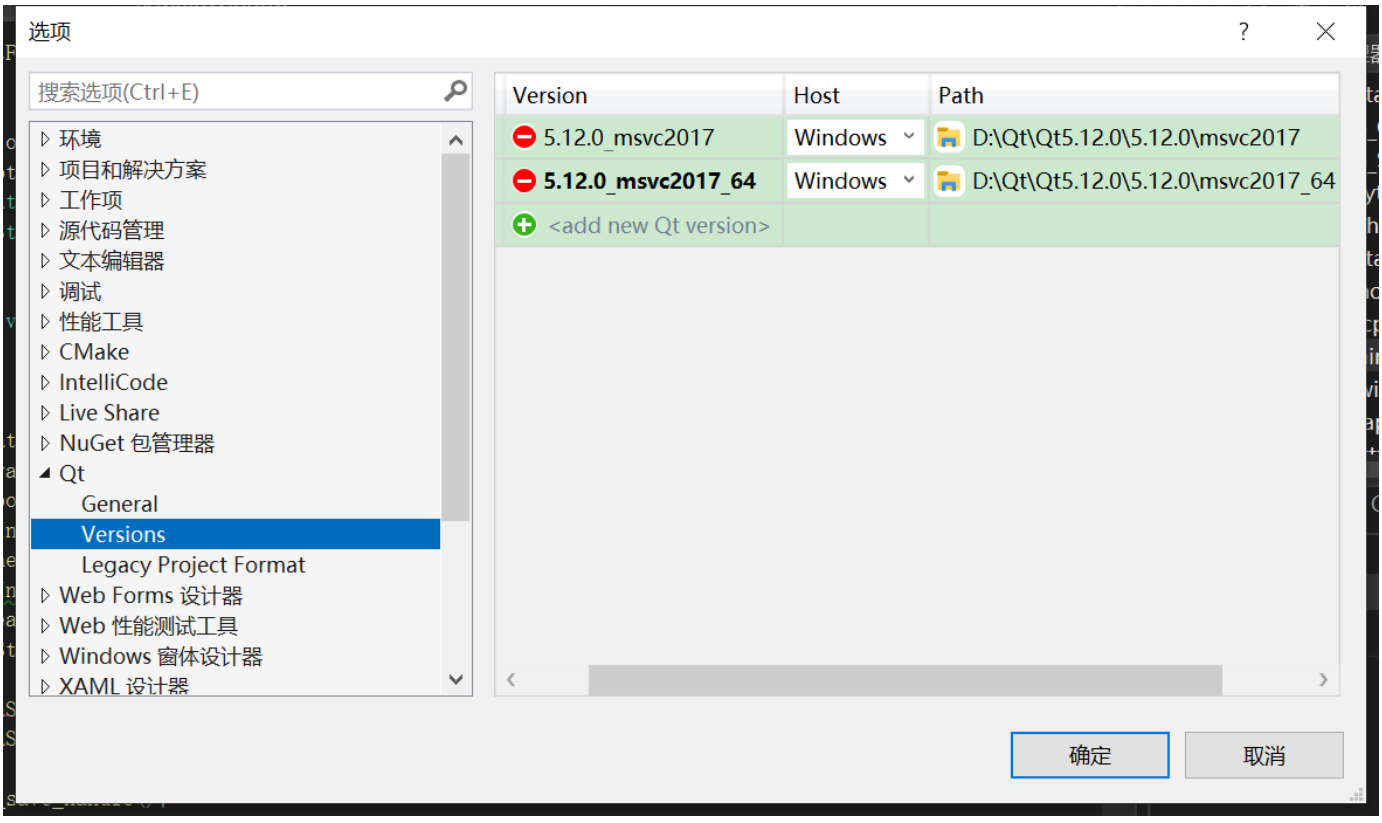
qt-vsaddin-msvc2022-2.8.1-rev.06.vsix

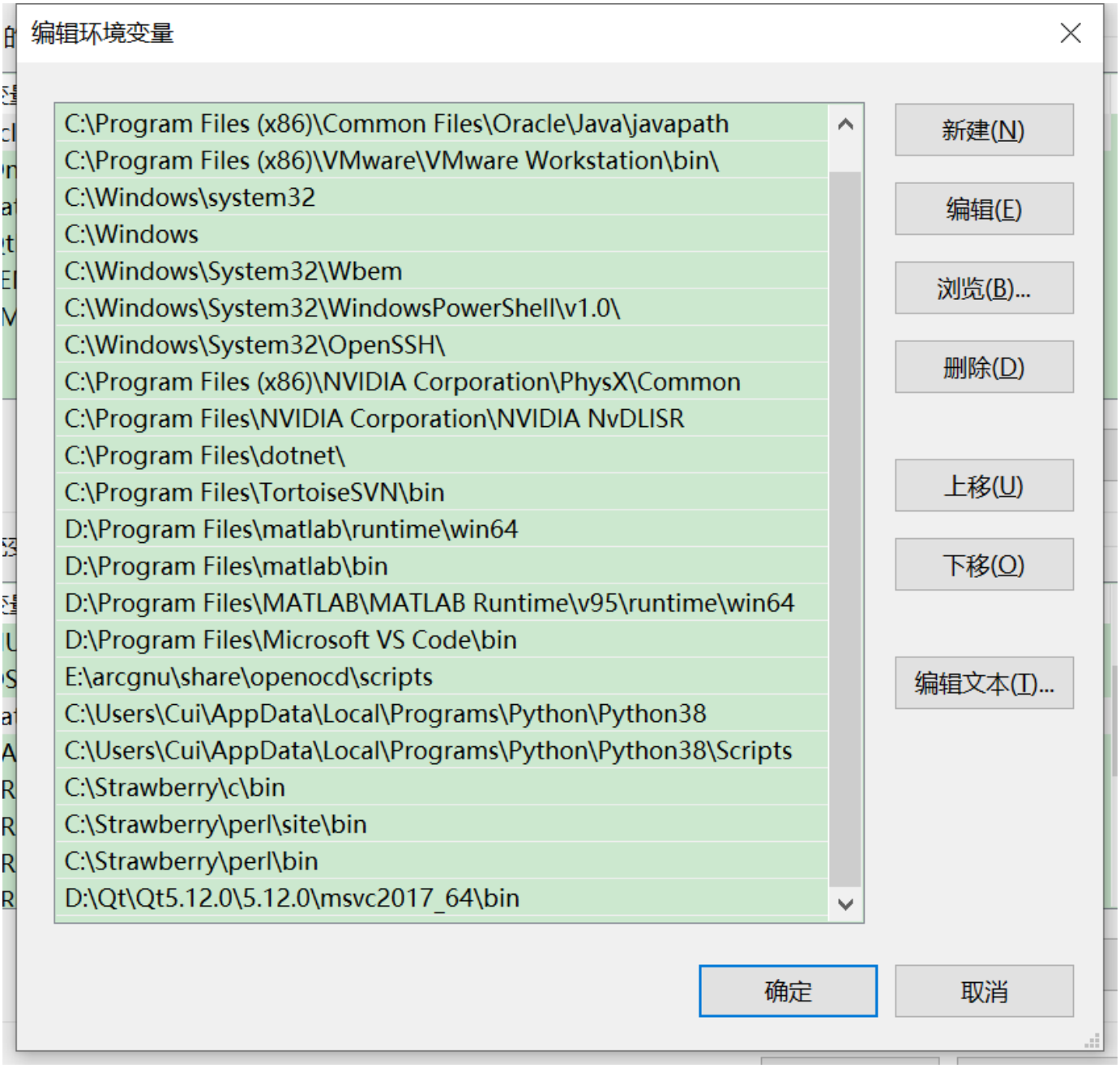
https://blog.csdn.net/zaozao_/article/details/122685557

4.vs搭建环境

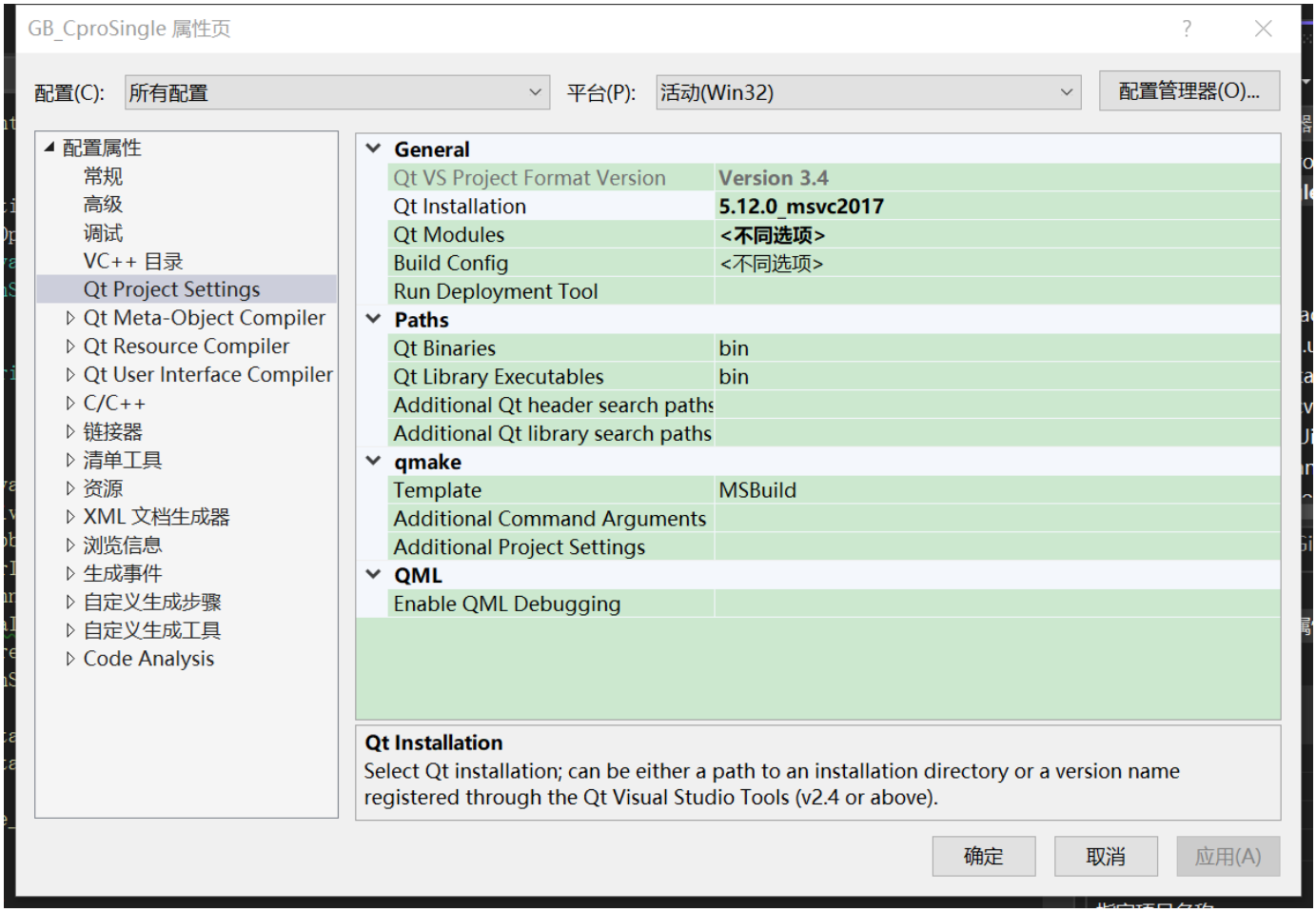
打开sln,点击扩展->Qt VS Tools菜单->qt versions（俩个都试一下）

添加如下（要添加对应的环境变量）

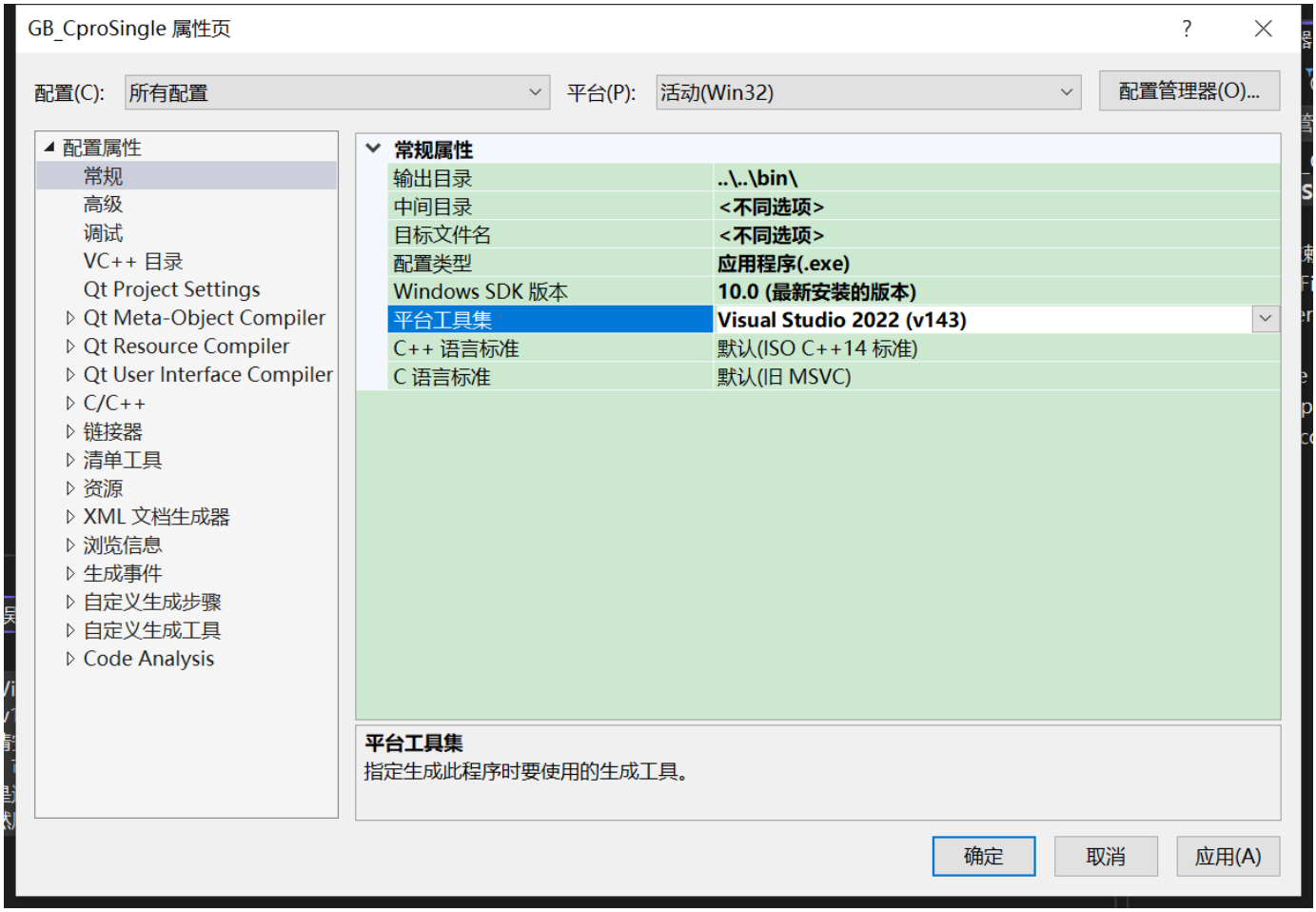




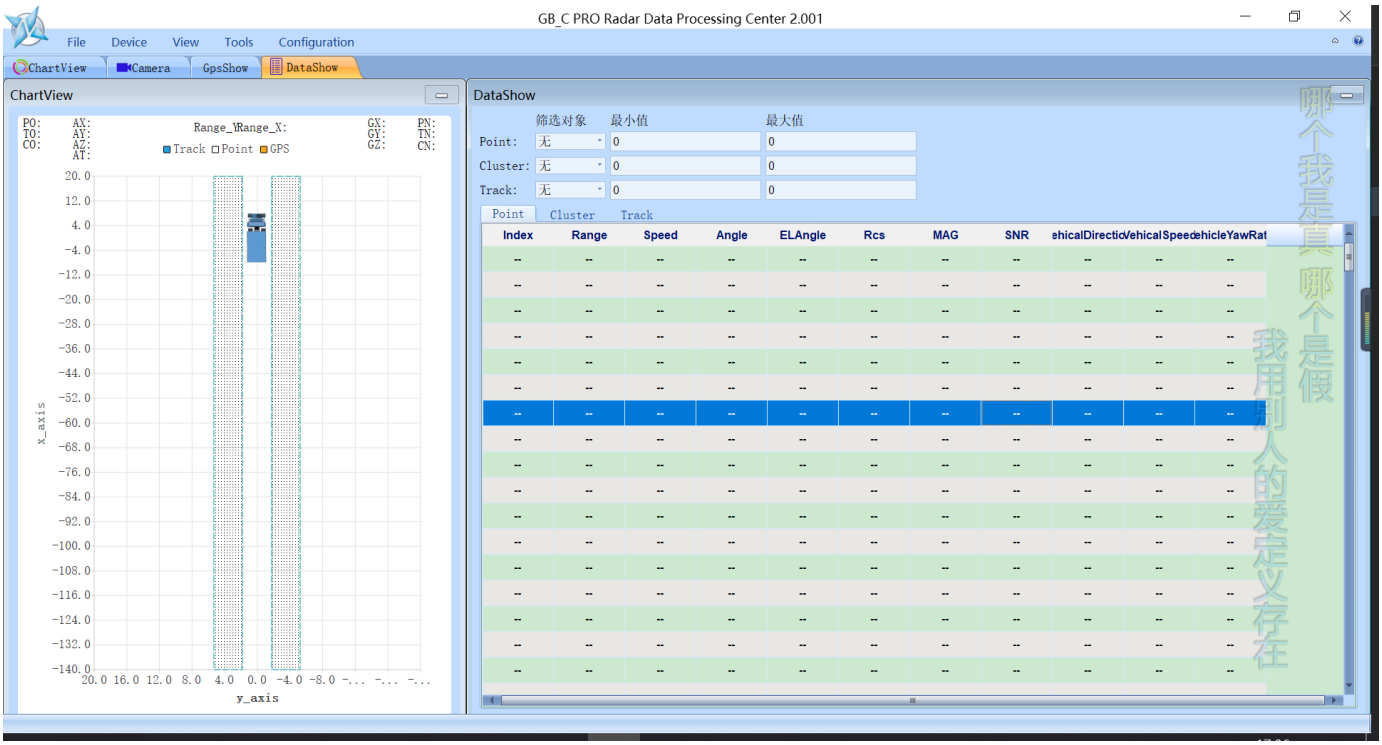
打开项目的属性，配置QT installation


GB_CproSingle 属性页
配置(C): 所有配置
平台(P): 活动(Win32)
配置管理器(O)...
配置属性
常规
高级
调试
VC++ 目录
Qt Project Settings
Qt Meta-Object Compiler
Qt Resource Compiler
Qt User Interface Compiler
C/C++
链接器
清单工具
资源
XML 文档生成器
浏览信息
生成事件
自定义生成步骤
自定义生成工具
Code Analysis
General
Qt VS Project Format Version
Version 3.4
Qt Installation
5.12.0_msvc2017
Qt Modules
<不同选项>
Build Config
<不同选项>
Run Deployment Tool
Paths
Qt Binaries
bin
Qt Library Executables
bin
Additional Qt header search paths
Additional Qt library search paths
qmake
Template
MSBuild
Additional Command Arguments
Additional Project Settings
QML
Enable QML Debugging
Qt Installation
Select Qt installation; can be either a path to an installation directory or a version name registered through the Qt Visual Studio Tools (v2.4 or above).
确定
取消
应用(A)

配置平台工作集


GB_CproSingle 属性页
配置(C): 所有配置
平台(P): 活动(Win32)
配置管理器(O)...
配置属性
常规
高级
调试
VC++ 目录
Qt Project Settings
Qt Meta-Object Compiler
Qt Resource Compiler
Qt User Interface Compiler
C/C++
链接器
清单工具
资源
XML 文档生成器
浏览信息
生成事件
自定义生成步骤
自定义生成工具
Code Analysis
常规属性
输出目录
..\..\bin\
中间目录
<不同选项>
目标文件名
<不同选项>
配置类型
应用程序(.exe)
Windows SDK 版本
10.0 (最新安装的版本)
平台工具集
Visual Studio 2022 (v143)
C++ 语言标准
默认(ISO C++14 标准)
C 语言标准
默认(旧 MSVC)
平台工具集
指定生成此程序时要使用的生成工具。
确定
取消
应用(A)

点击本地调试器，生成上位机



上下位机同步

下位机：

在自己加的功能的总的.h文件定义

//#define OFFLINE_ADAS_APP_

//条件编译，离线回灌使用，下位机注释掉，上位机打开

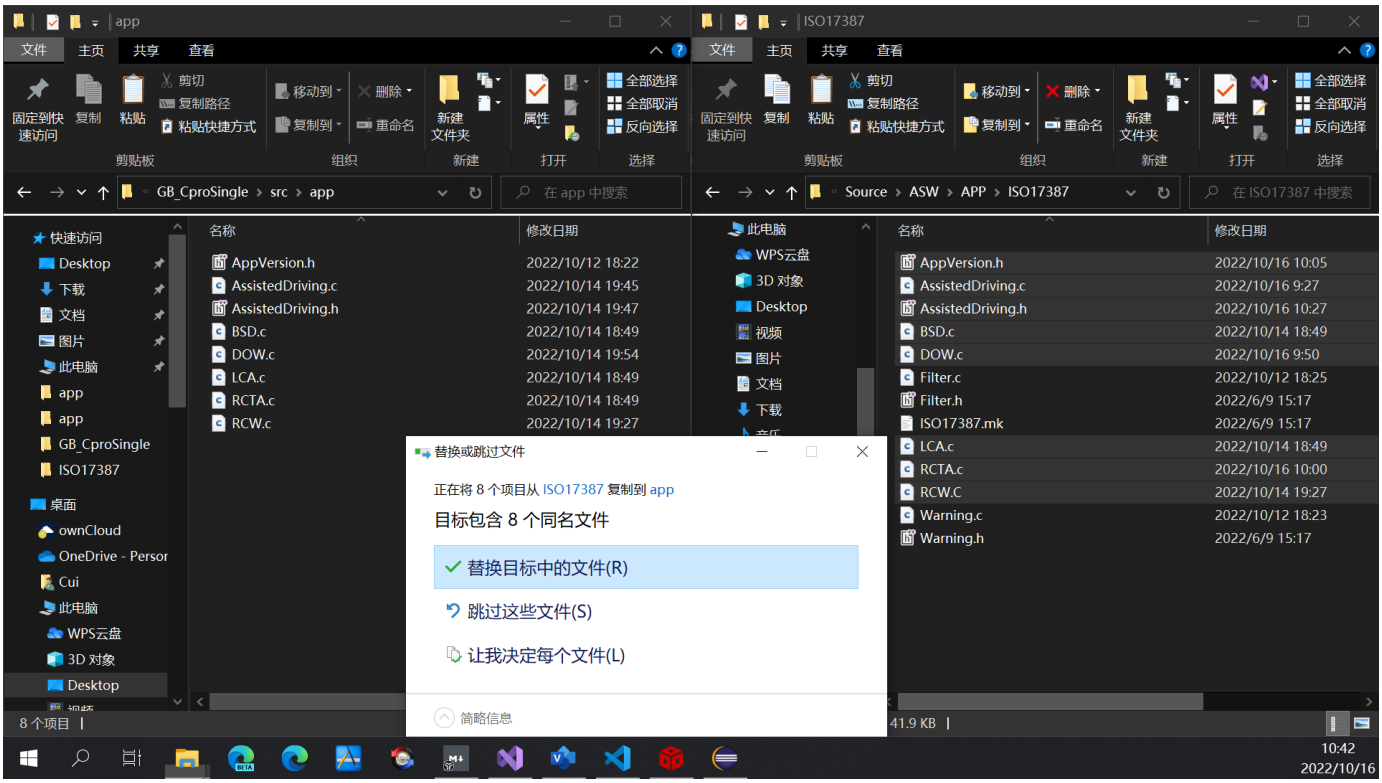
e.g.

```
#ifndef OFFLINE_ADAS_APP_   //下位机模式
#include "stdint.h"
#include "DF_ProExternalFile.h"
#include "../../APP/Com/Vehicle_Info.h"

#else   //上位机模式
#include "stdint.h"
#include "../DPM/DF_ProExternalFile.h"
#include "AppVersion.h"

#endif // !OFFLINE_ADAS_APP_
```

往上位机挪了一个专属的"AppVersion.h"，这个.h在下位机是app文件夹之外的文件



上位机：

下位机编译过后，用到的.c和.h覆盖上位机对应的文件，打开上位机，取消注释//#define OFFLINE_ADAS_APP_

调通上位机接口，用到的参数往总的.h加，完成后用到的.c和.h覆盖下位机对应的文件，app文件夹内仅上位机用到的文件可不挪到下位机或者推荐离线回灌不使用

e.g.

```
/*
 * AppVersion.h
 *
 *  Created on: 2021年8月26日
 *      Author: Eric
 */

#ifndef SOURCE_CONFIG_APPVERSION_H_
#define SOURCE_CONFIG_APPVERSION_H_
#ifndef OFFLINE_ADAS_APP_    //下位机模式

#else    //上位机模式
#include "stdint.h"

#define CALIBRATION_RCS_ADDR     (0x1D0000)


#define RADAR_LEFT      (0)
```

```
#define RADAR_RIGHT     (1)
#define FRONT_RADAR     (0)
#define CORNER_RADAR    (1)

typedef enum
{
    RADAR_FRONT_1 = 0,
    RADAR_FRONT_2,
    RADAR_FRONT_LEFT,
    RADAR_FRONT_RIGHT,
    RADAR_REAR_LEFT,
    RADAR_REAR_RIGHT,
    RADAR_REAR_1,
    RADAR_REAR_2,
    RADAR_LATERAL_LEFT,
    RADAR_LATERAL_RIGHT,
    RADAR_TYPE_MAX,
}Radar_Type_t;

typedef struct
{
    float x_min;
    float x_max;
    float y_min;
    float y_max;
}Radar_FunArea_Type;

typedef struct
{
    uint8_t saveflag;
    uint8_t obj_num;
    Radar_FunArea_Type Area;
    Radar_FunArea_Type BSD1;
    Radar_FunArea_Type BSD2;
    Radar_FunArea_Type RCTA;
}Radar_Cfg_Type;

extern char SOFTWARE_VER[];
extern uint8_t CURRENT_RADAR;
extern uint8_t OBJ_COMBINE;
extern uint8_t RADAR_COORDINATE;
extern Radar_Cfg_Type Radar_Cfg;



//void Radar_Parameter_Initialize(void);
void Radar_Propeties_Init(void);
uint8_t Radar_Read_HW_Version(void);

#endif // !OFFLINE_ADAS_APP_
#endif /* SOURCE_CONFIG_APPVERSION_H_ */
```