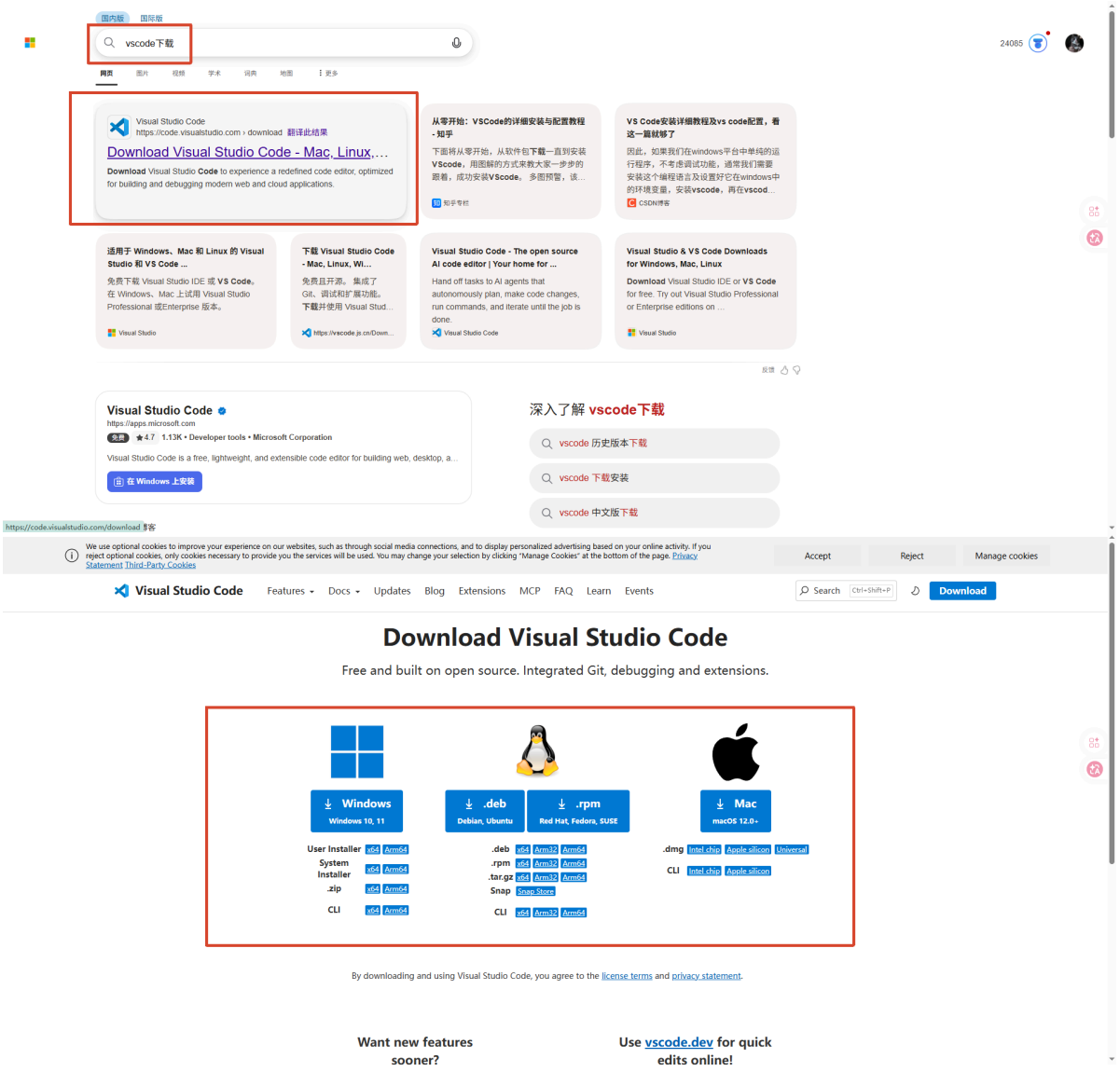
2-使用 VScode + keil 实现(代码提示)编辑+编译程序

第一步 下载并安装VS code

由于我已经安装好了，并且网络上也有大量的教程，因此这一步便不过多叙述。



第二步 VScode 的初步设置

1、安装中文插件 进行汉化

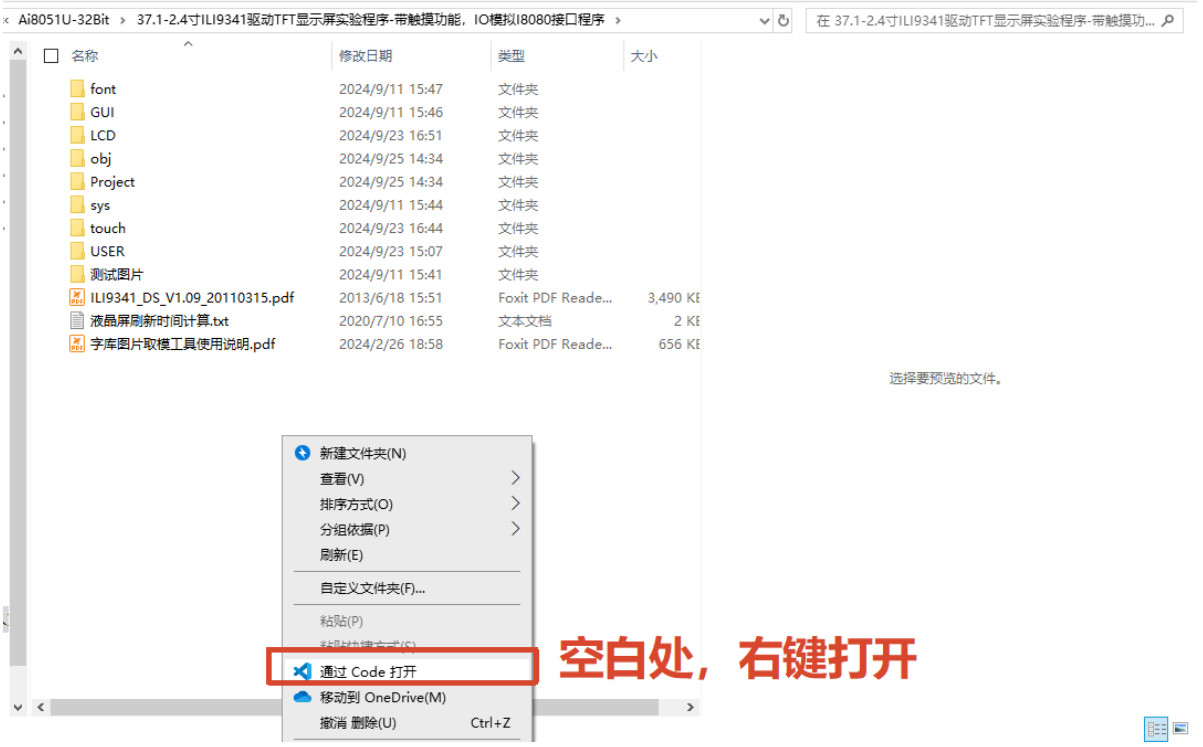
大家第一次打开 VScode 时，应该是英文界面，需要在插件商店搜索 Chinese 插件进行汉化。
先切换到扩展界面，再搜索汉化插件【Chinese】，点击install安装。
安装完成之后，右下角会提示需要切换语言并重启后生效【Change Language and Restart】，点击重启即可。
注：右侧工具栏的顺序可能与我的不一致（我可能切换了排序顺序），大家看这个方块的图标即可，或者按快捷键ctrl+shift+X 切换到扩展界面

2、安装C语言相关插件

我们常使用C语言进行单片机编程，因此，想要轻松阅读编辑代码，需要下载相关的插件。

1.未安装插件的界面

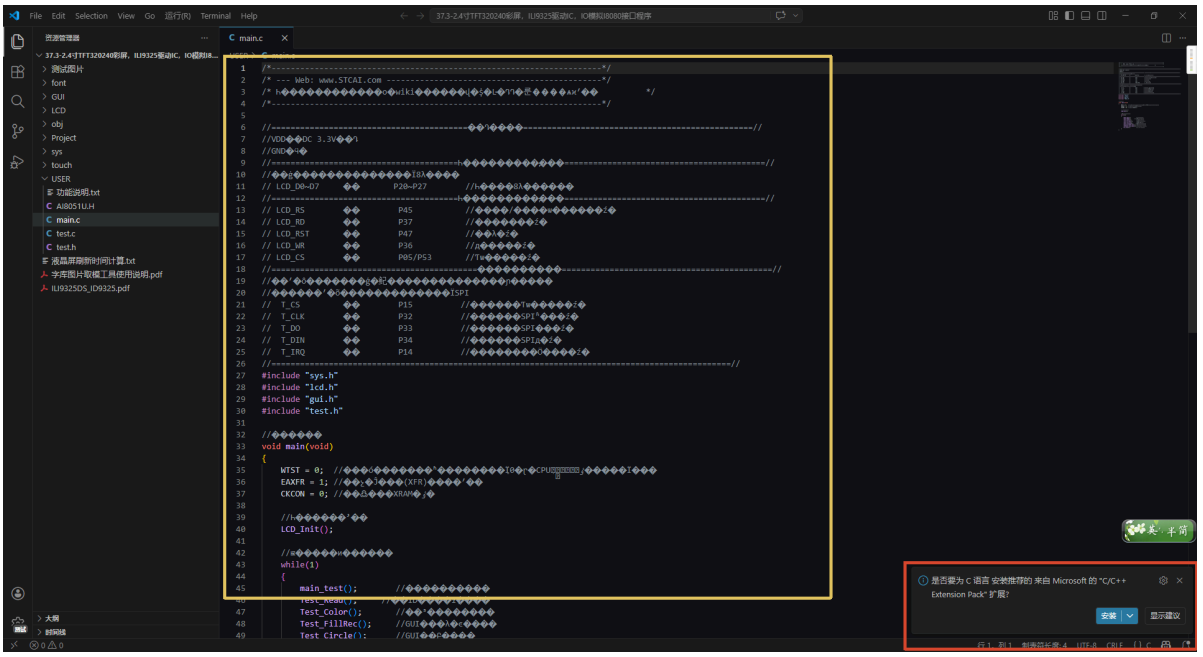
先打开一个官方的例程试试看是什么样子的



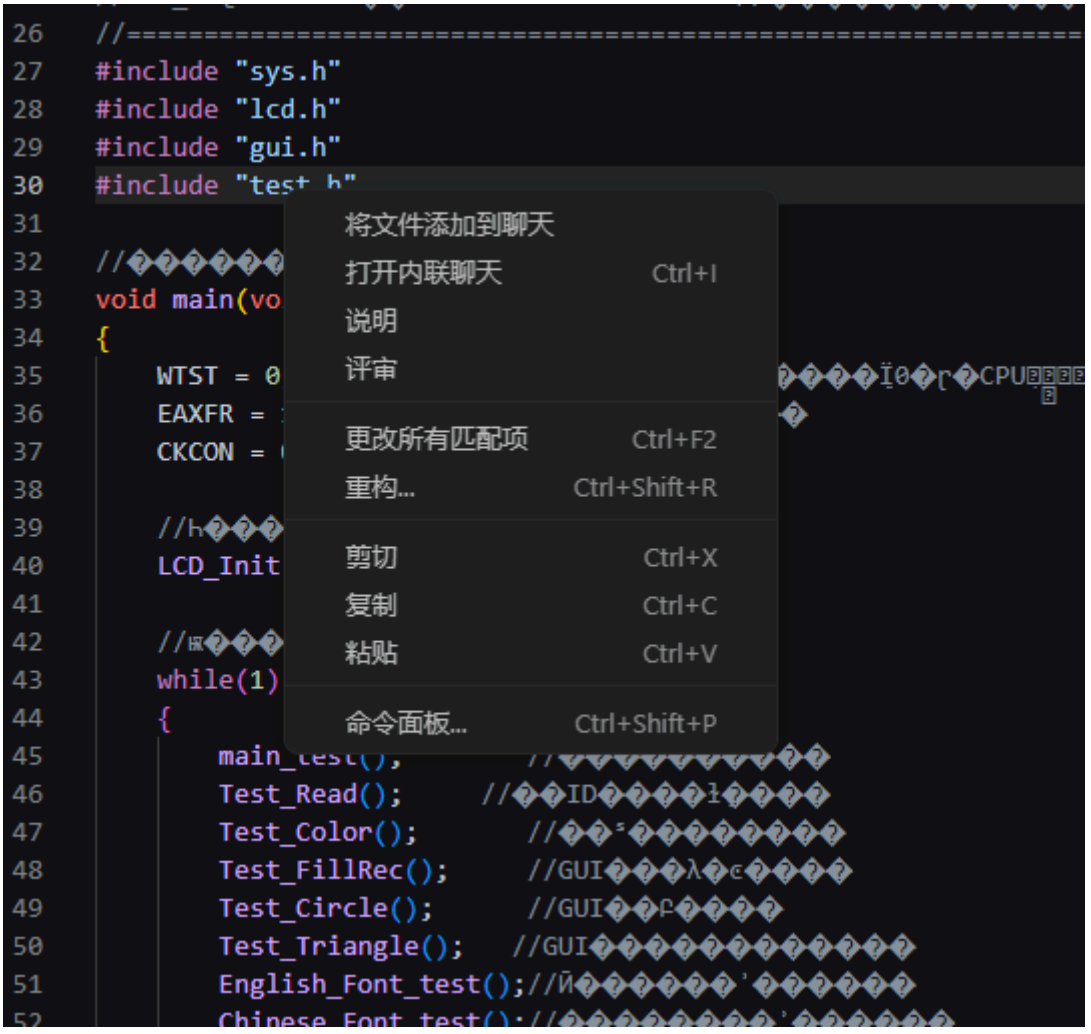
也可以，点击文件【File】，打开文件夹。

打开文件夹之后，再打开main.c文件，在右下角可以看见编辑器已经识别出C语言了并提示我们安装对应的插件，

同时还可以看见注释都变成了乱码，这是因为官方例程是使用GBK编码保存的，而系统默认都是UTF-8编码，编码的问题稍后再介绍。

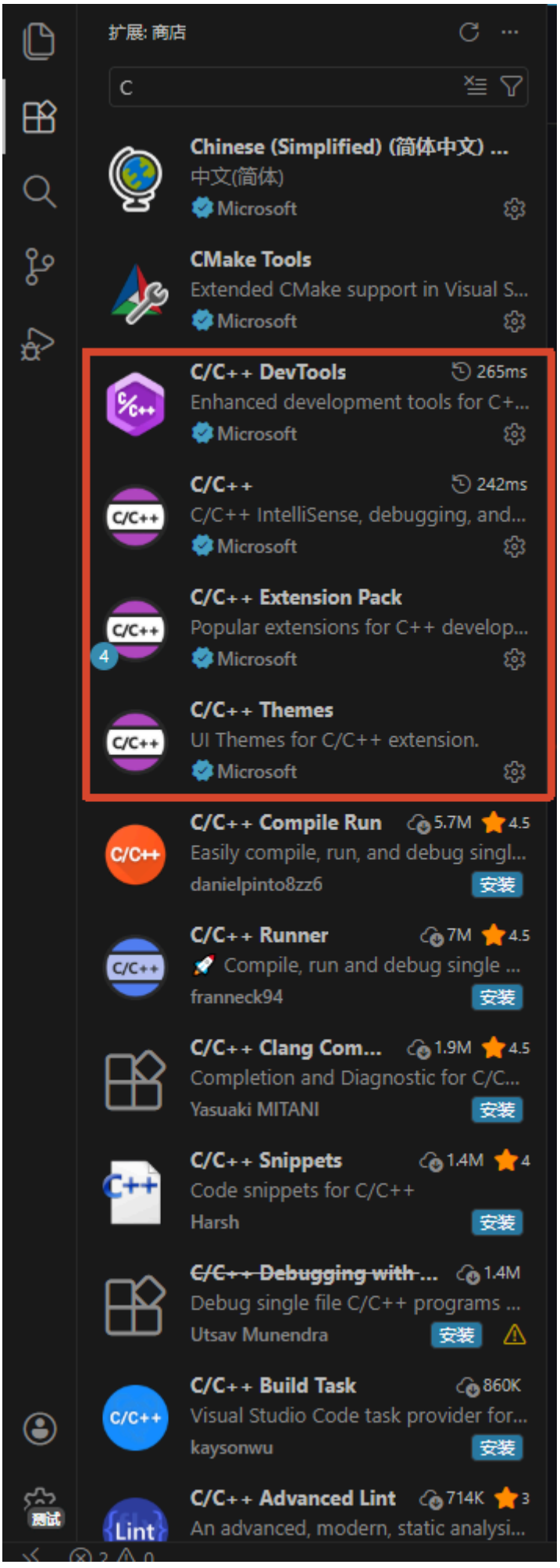


没有安装插件时，我们无法通过长按Ctrl键，单击跳转。并且右键也没有跳转选项。

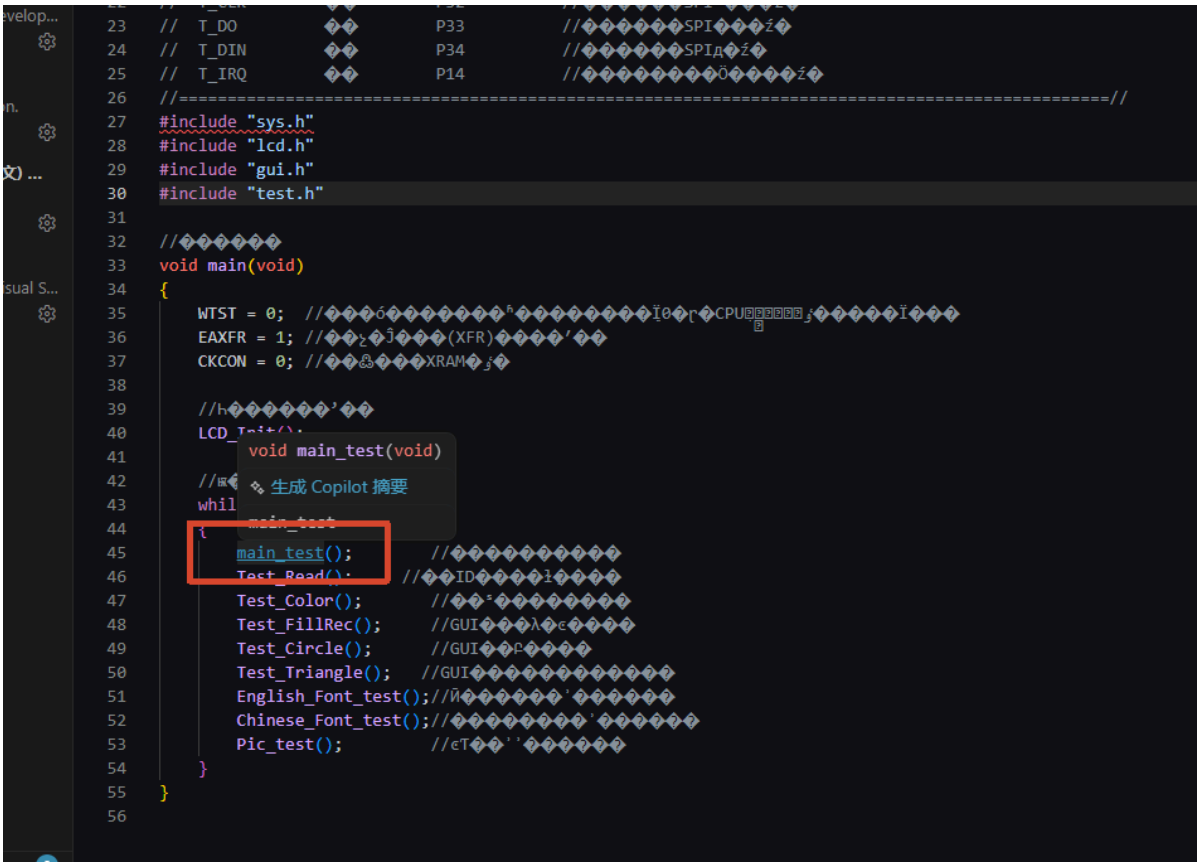


2.安装C语言插件

还是回到扩展界面，搜索【C】，安装C/C++插件，把这4个都安装上，或者直接点击之前弹出的提示框一键安装。

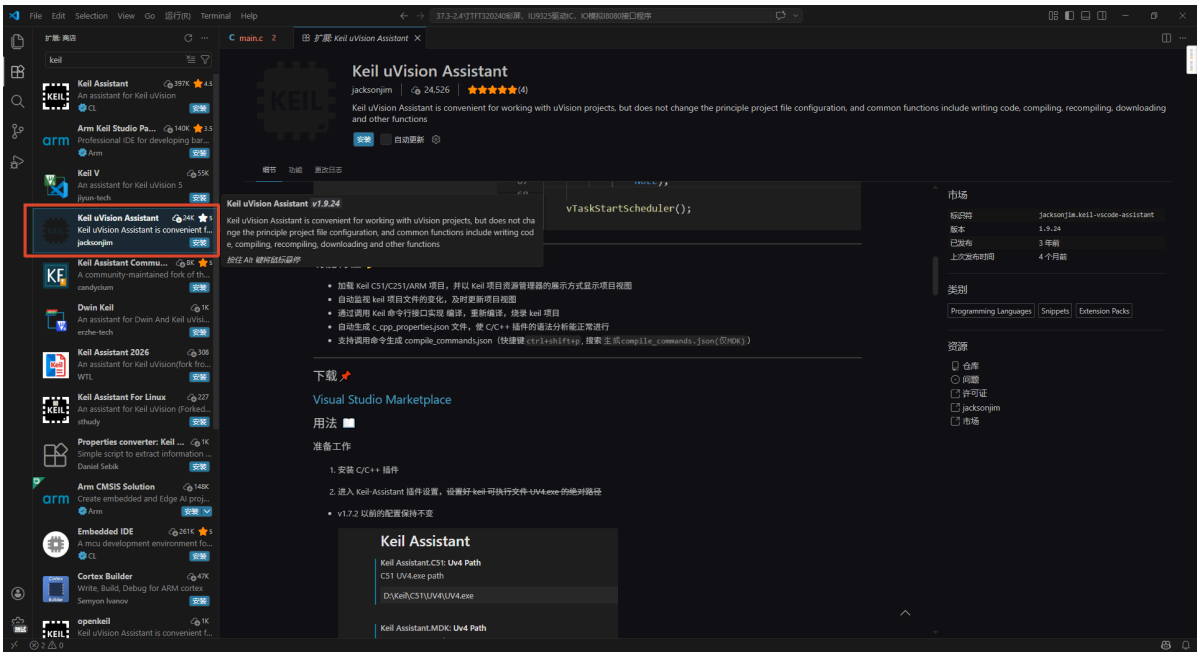


安装完插件后，可以进行快捷跳转了，右键也多了很多功能。

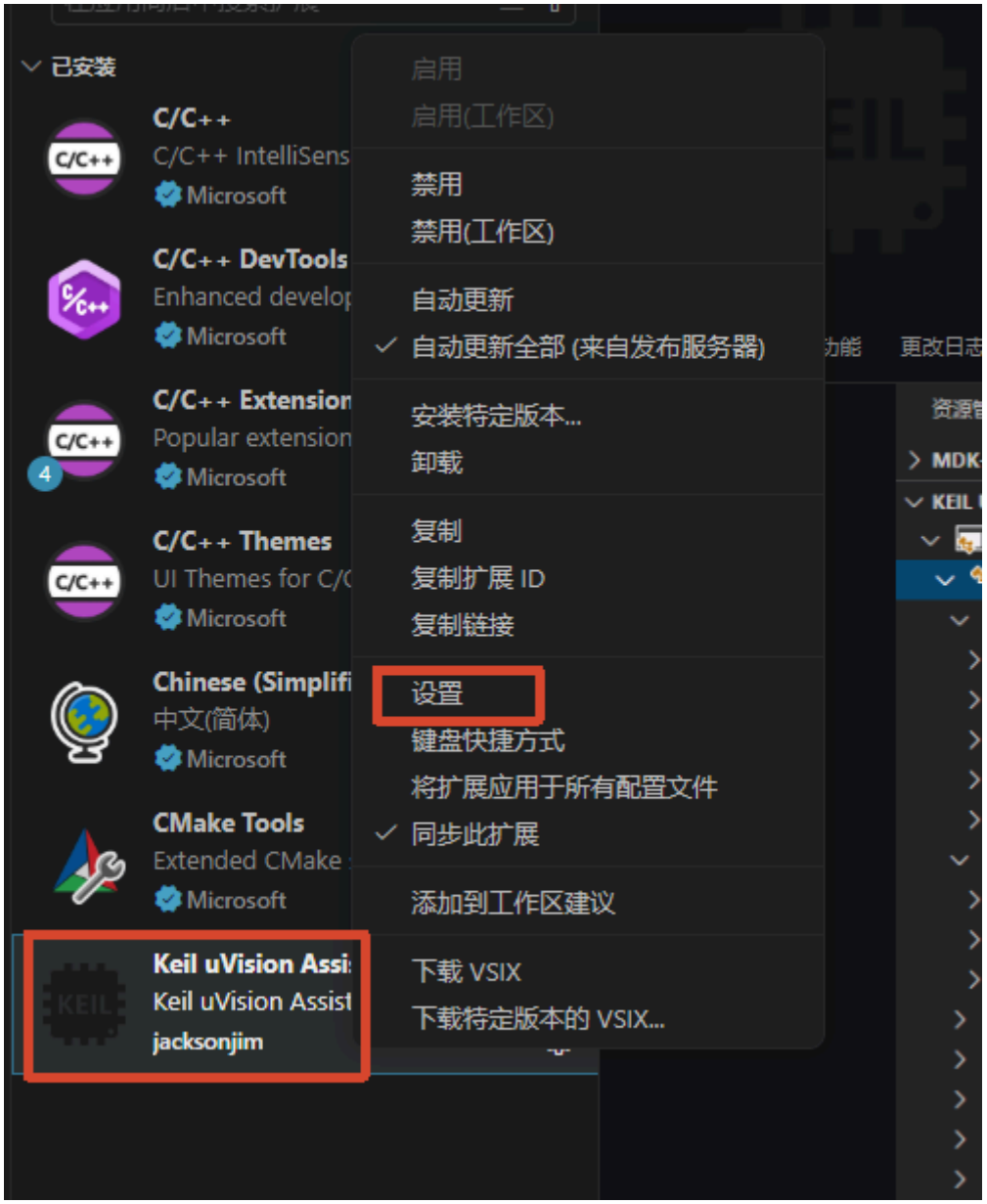

23 // T_DO ?? P33 //???????SPI???é?
24 // T_DIN ?? P34 //???????SPIд?é?
25 // T_IRQ ?? P14 //?????????Ö????é?
26 //==========
27 #include "sys.h"
28 #include "lcd.h"
29 #include "gui.h"
30 #include "test.h"
31
32 //???????
33 void main(void)
34 {
35 WTST = 0; //???ó?????????ʱ??????????Ï0?ר?CPU?????¡?????Ï???
36 EAXFR = 1; //??¿?Ĵ???(XFR)????'??
37 CKCON = 0; //??&???XRAM?¡?
38
39 //h???????'??
40 LCD_Init();
41
42 //
43 whil
44 {
45 main_test(); //???????????
46 Test_Read(); //??ID????ì????
47 Test_Color(); //??ˢ??????????
48 Test_FillRec(); //GUI???λ?ε????
49 Test_Circle(); //GUI??Բ????
50 Test_Triangle(); //GUI????????????
51 English_Font_test();//Ӣ???????'???????
52 Chinese_Font_test();//??????????'???????
53 Pic_test(); //єT??''???????
54 }
55 }
56
void main_test(void)
生成 Copilot 摘要
main_test

3.安装链接 keil 的插件

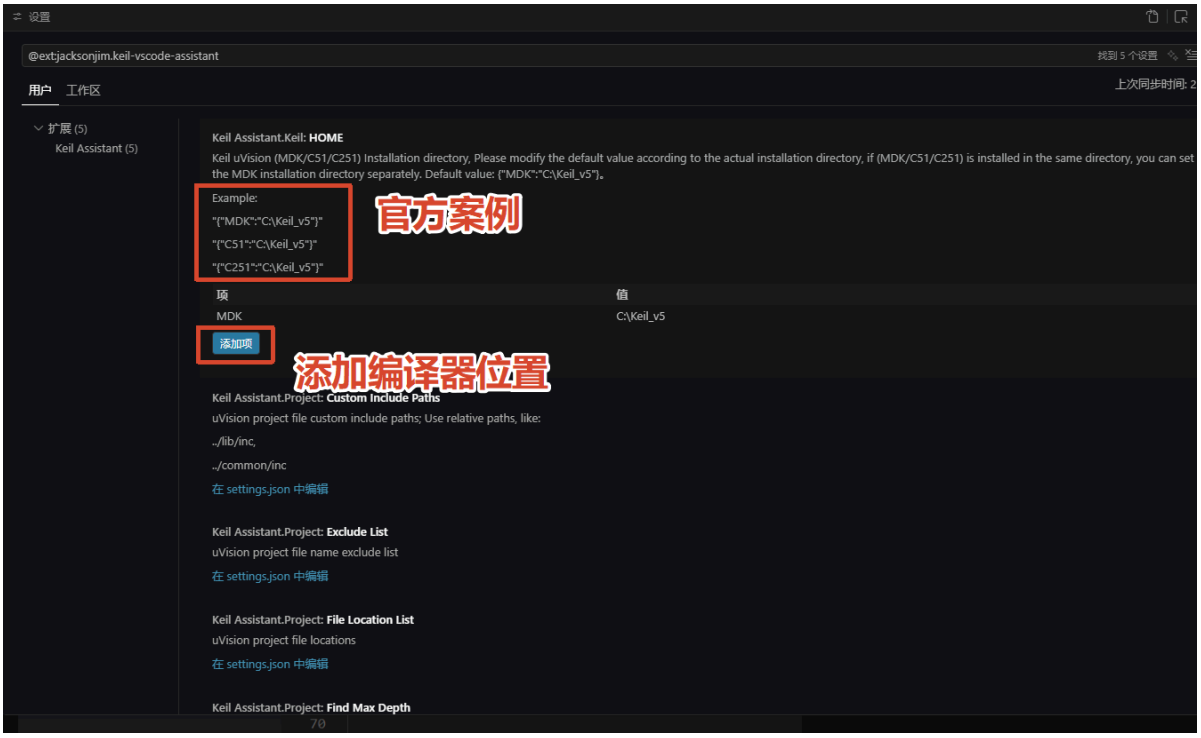
继续在插件商城里，搜索【Keil uVision Assistant】，切记不要安装错了，因为我了解到的，目前只有这个插件支持C51/C251/ARM，其他的一般不支持C251。



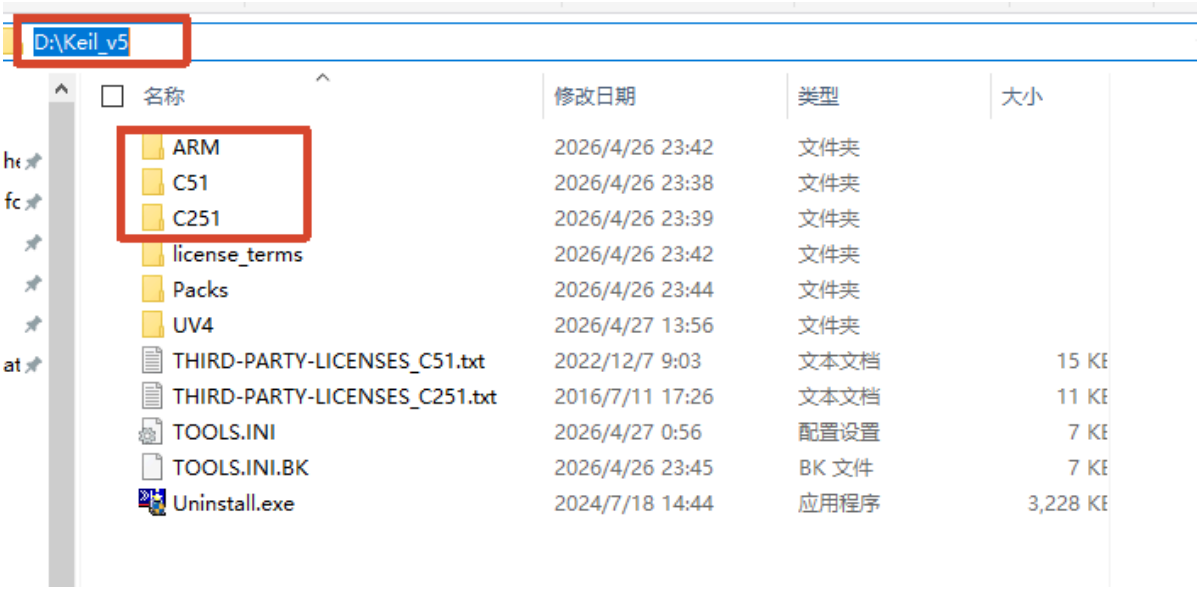
安装完成后，右键设置一下参数。



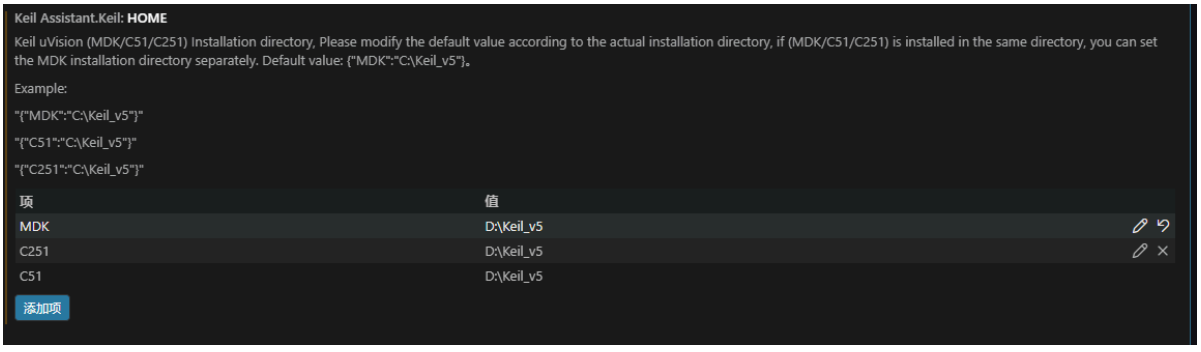
可以参考一下官方案例，进行配置。
项分别填入MDK、C51、C251。值填入keil5的安装目录。
注：如果你是参考【萌新指南】序列00--keil5安装指南（官方渠道方法下载并安装C251, C51），安装的keil5，那么3种编译器的路径都是一样。如果不是，请按照你的安装目录为准。



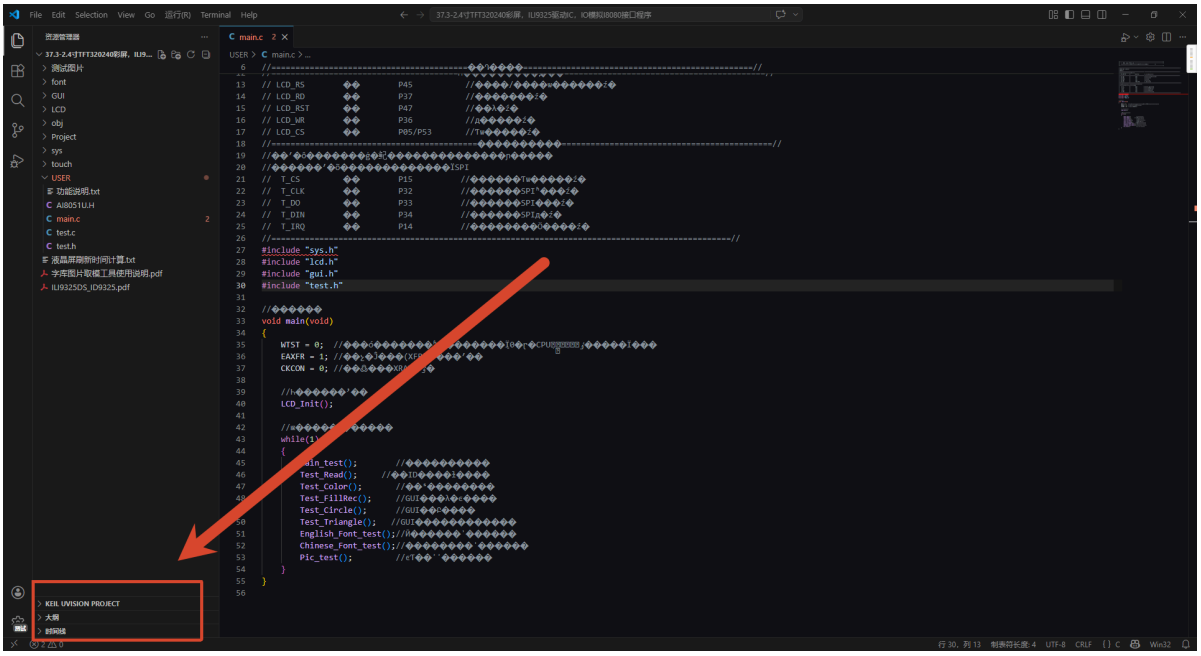
我的MDK、C51、C251安装目录都是在 D:\Keil_v5



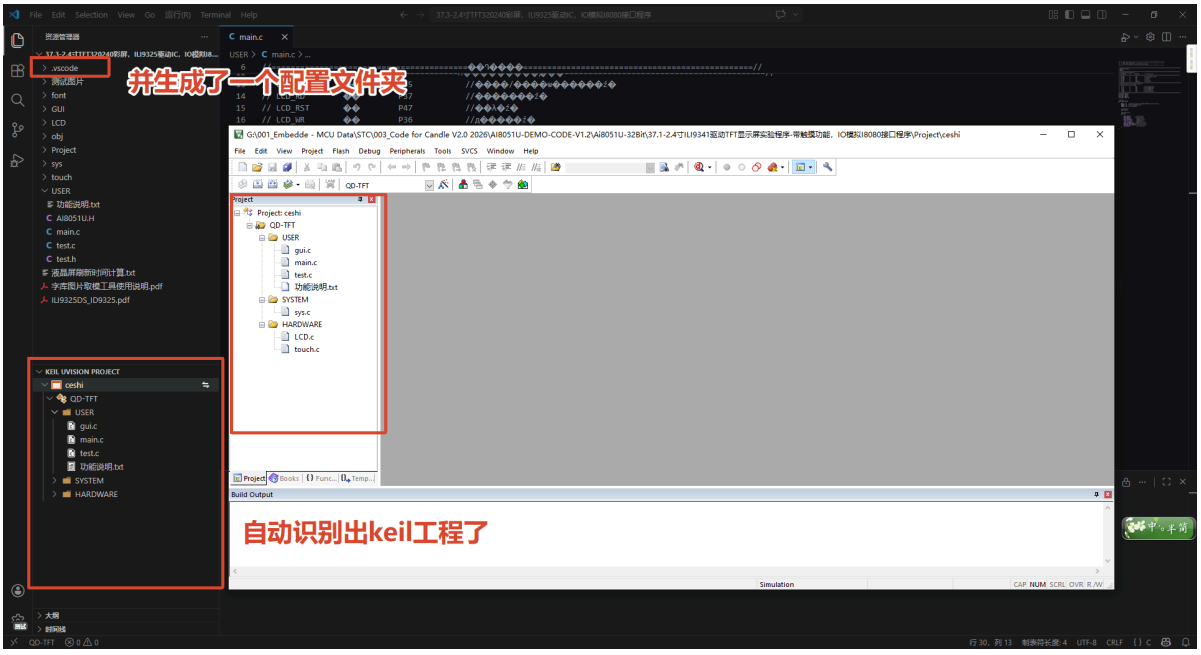
因此我配置成这样就好了



这个插件配置完成后，回到资源管理器（侧边工具栏中文件的图标）。就可以看到有一个keil uvison project目录



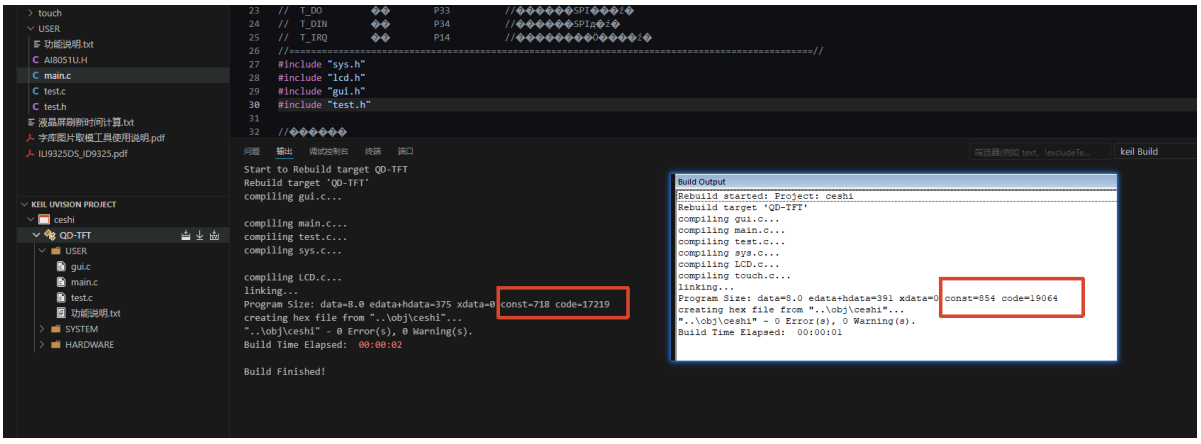
点开这个目录，就会发现Vscode已经自动识别出工程了，并生成了一个配置文件夹。



这3个按键与keil中的是一样的效果。

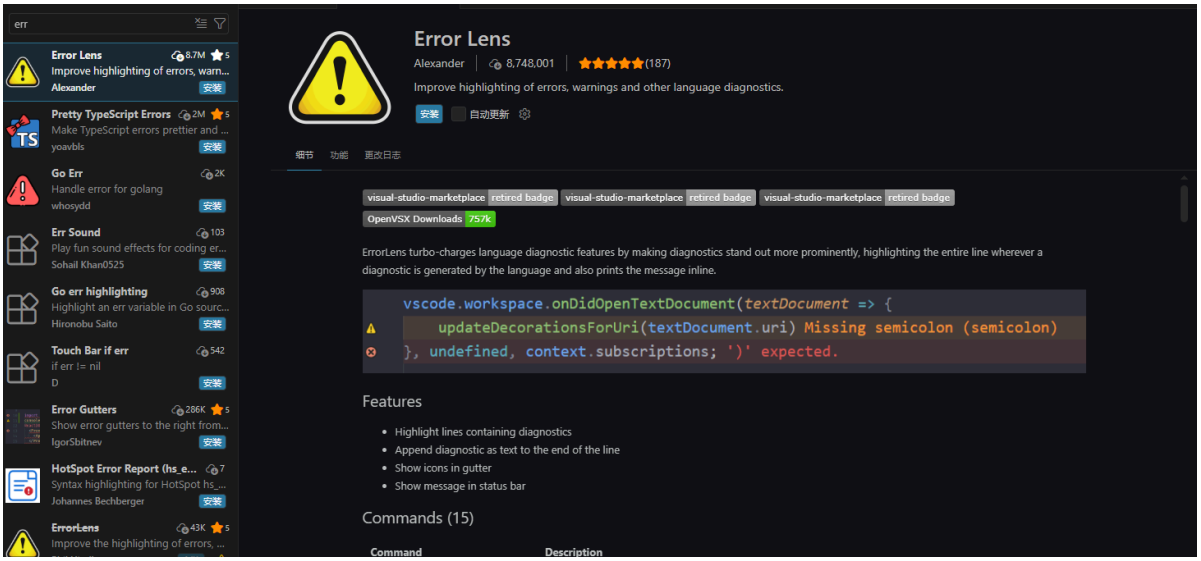
使用全部编译后，也是成功编译了。但是编译结果与keil中的有点差异，而且编译速度也远远不如keil。虽然能用，但十分不建议用这个插件。

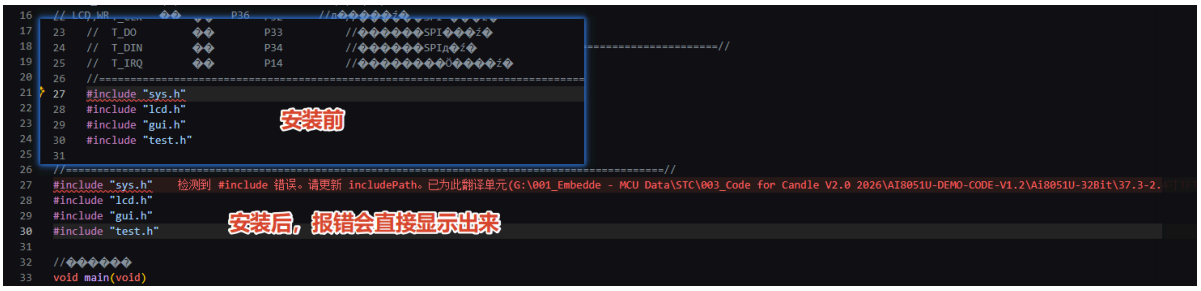
建议使用VScode编辑程序代码，使用keil5进行编译调试。



4.推荐的插件

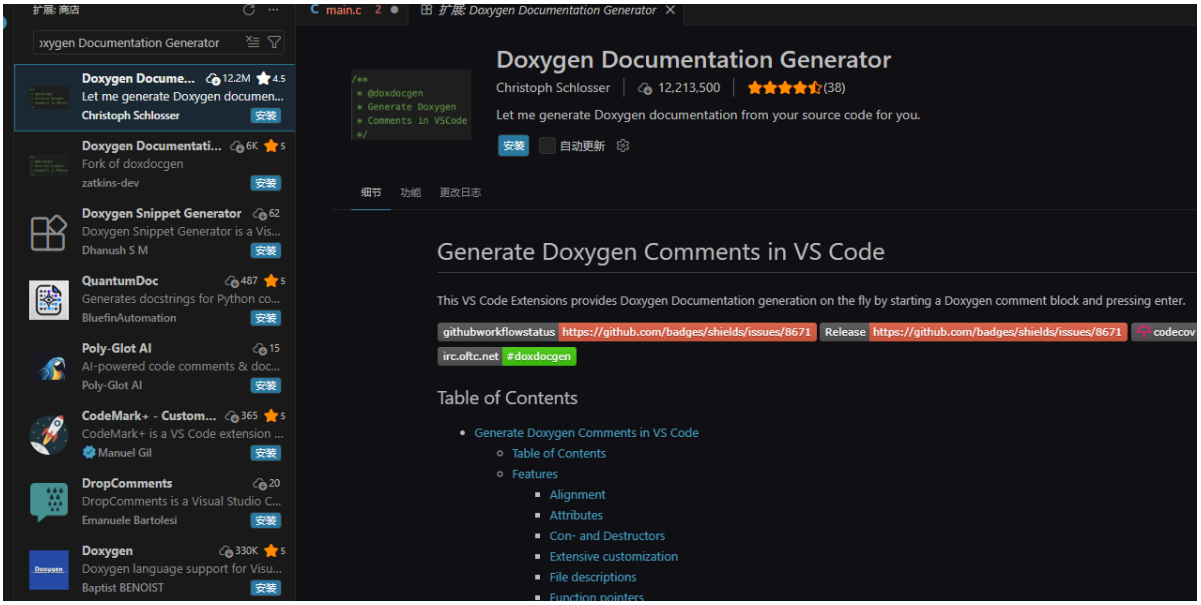
a.Error Lens 更直观的报错提示



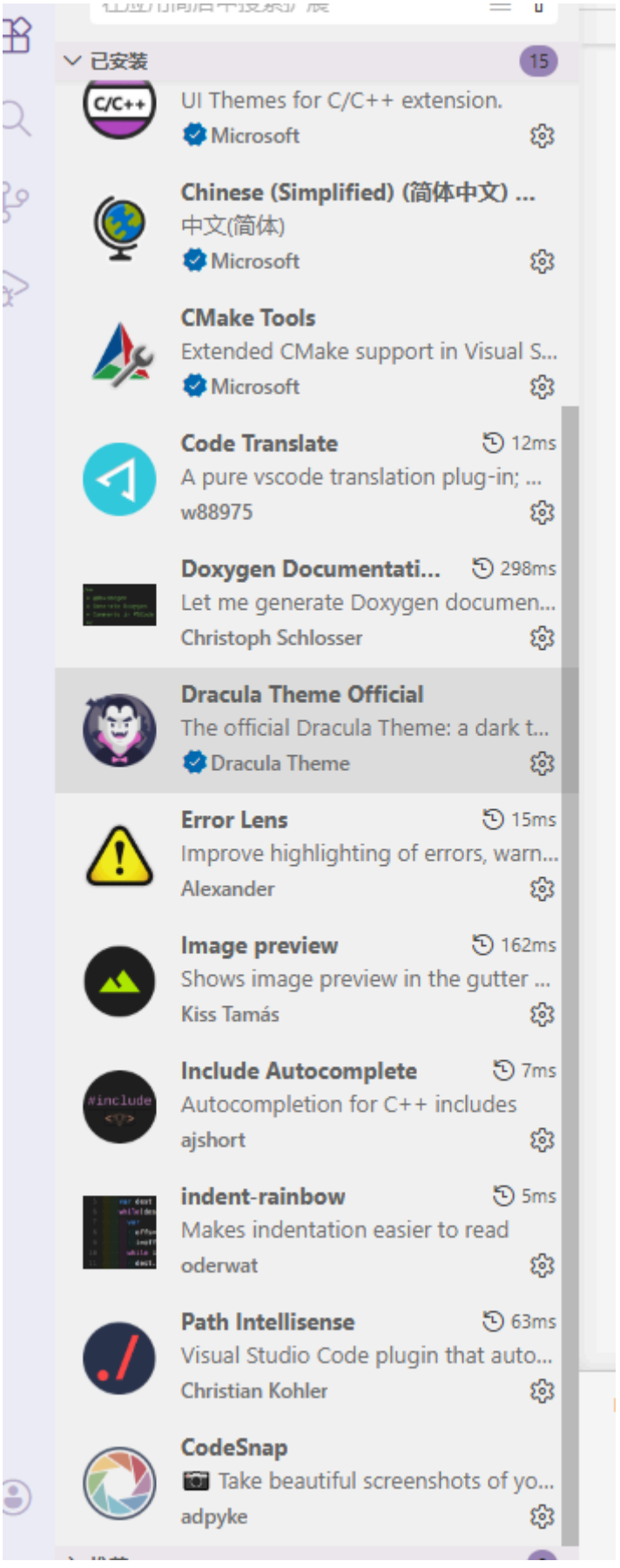


b.Code Translate 鼠标悬停在英文上，可以翻译单词（长句好像不行）

c.Doxygen Documentation Generator 更直观的注释

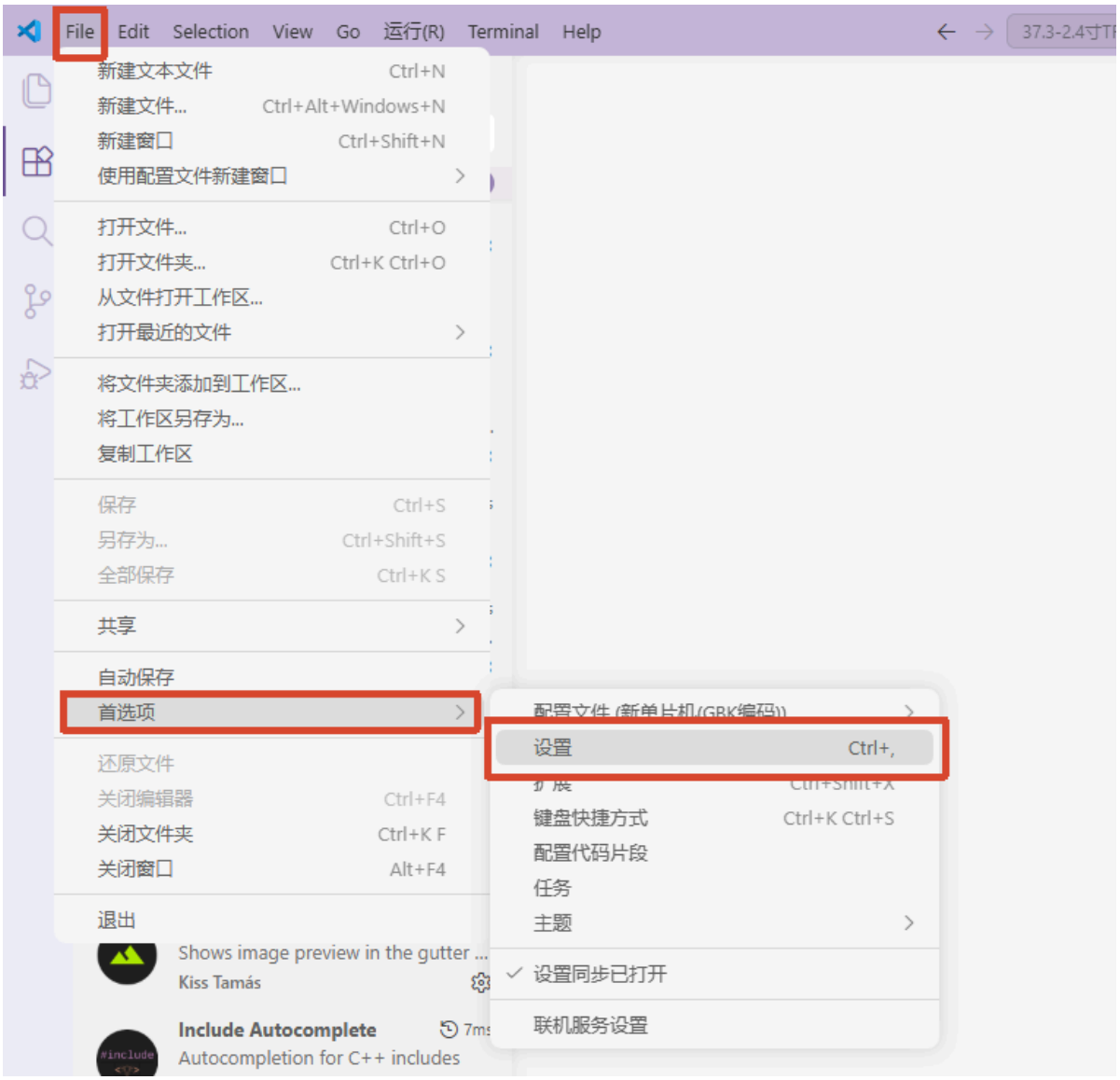


不过多缀叙了，本人在单片机中常用的插件如下


已安装
15
C/C++
UI Themes for C/C++ extension.
Microsoft
Chinese (Simplified) (简体中文) ...
中文(简体)
Microsoft
CMake Tools
Extended CMake support in Visual S...
Microsoft
Code Translate
12ms
A pure vscode translation plug-in; ...
w88975
Doxygen Documentati...
298ms
Let me generate Doxygen documen...
Christoph Schlosser
Dracula Theme Official
The official Dracula Theme: a dark t...
Dracula Theme
Error Lens
15ms
Improve highlighting of errors, warn...
Alexander
Image preview
162ms
Shows image preview in the gutter ...
Kiss Tamás
Include Autocomplete
7ms
Autocompletion for C++ includes
ajshort
indent-rainbow
5ms
Makes indentation easier to read
oderwat
Path Intellisense
63ms
Visual Studio Code plugin that auto...
Christian Kohler
CodeSnap
Take beautiful screenshots of yo...
adpyke

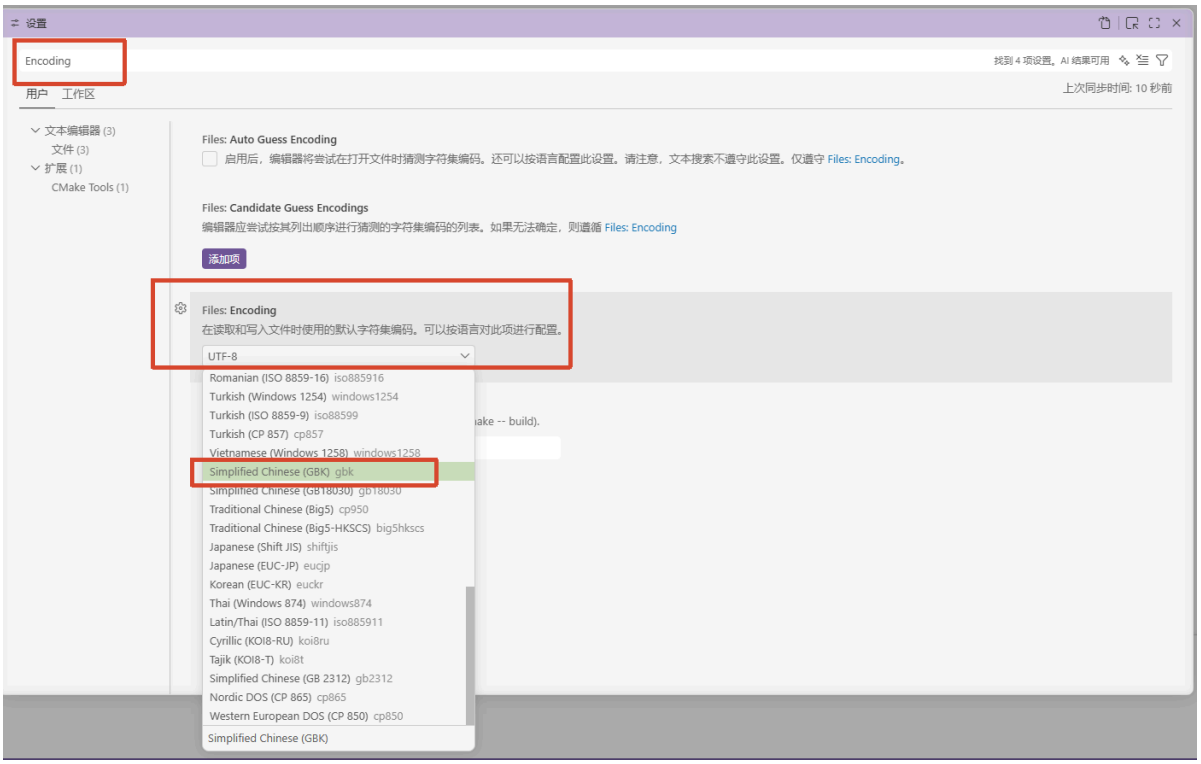
3、VScode 的基础设置

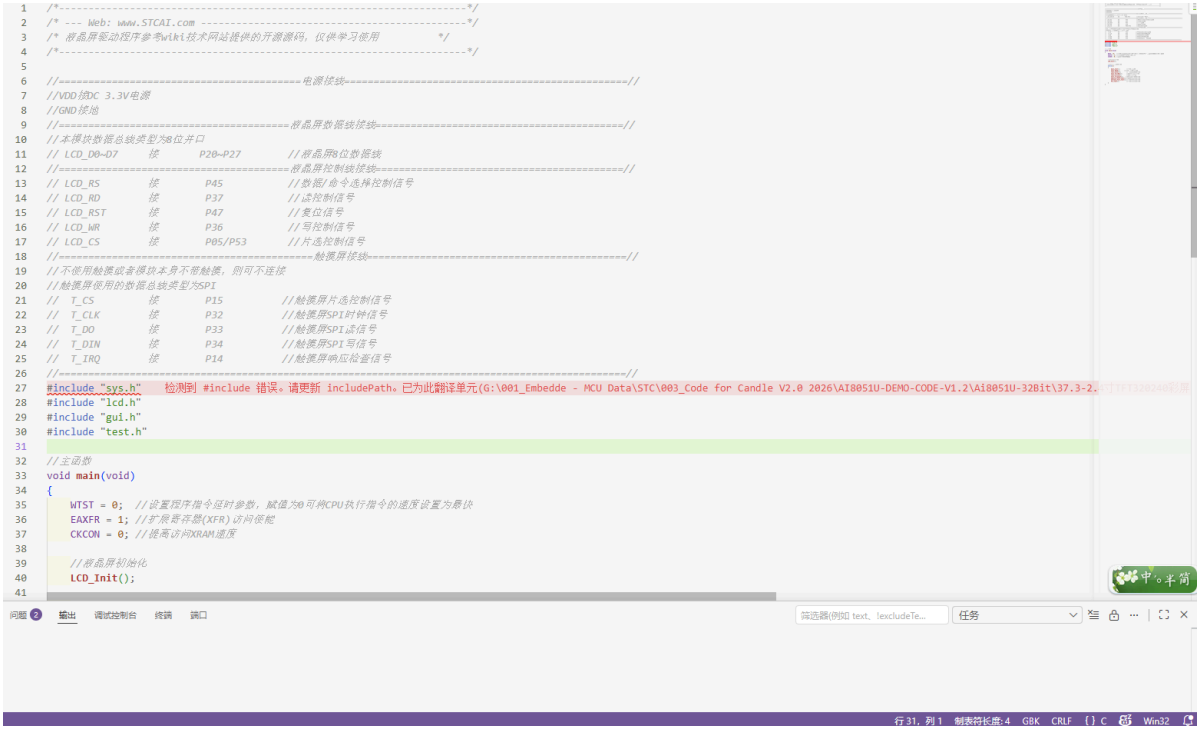
打开右上角File中的首选项，设置



1.乱码问题

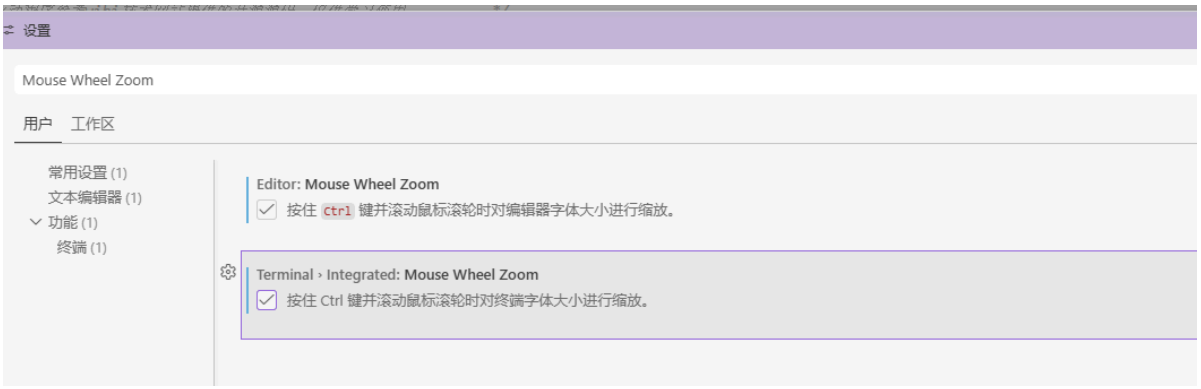
搜索Encoding，将UTF-8修改为GBK编码，如果你默认使用的时UTF-8编码，则不需要修改。





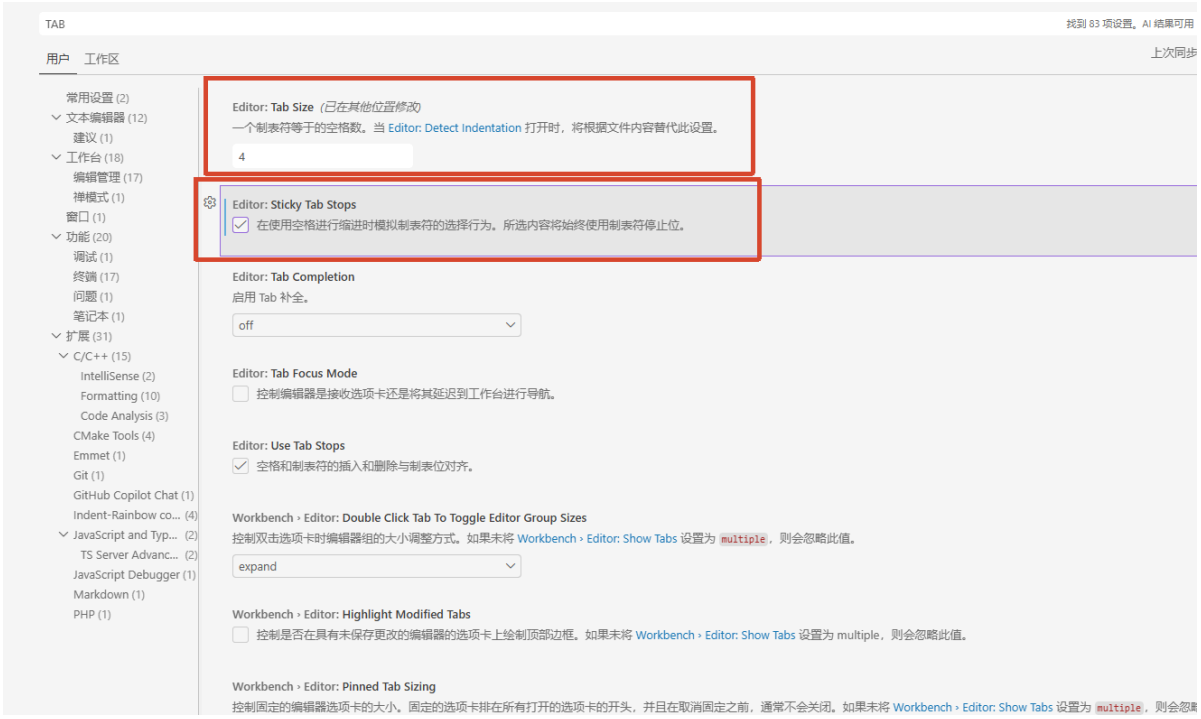
2. Ctrl + 鼠标滚轮：缩放代码字体大小

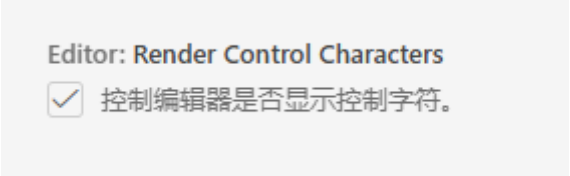
搜索 Mouse Wheel Zoom，根据需求勾选

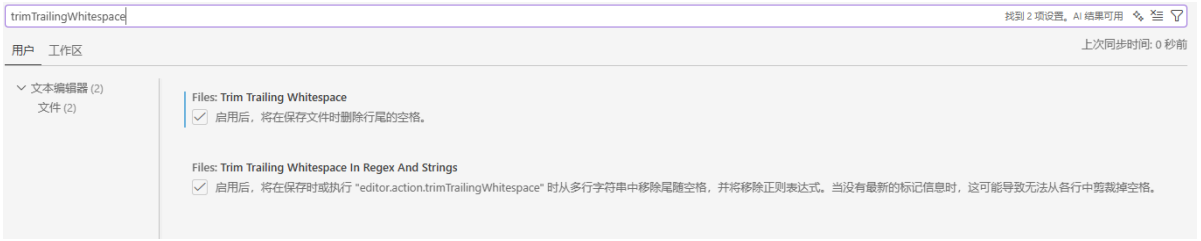


3.TAB键的设置(曾经打开过的文件不会采用修改后的配置)

搜索 TAB ， 可以根据个人习惯设置



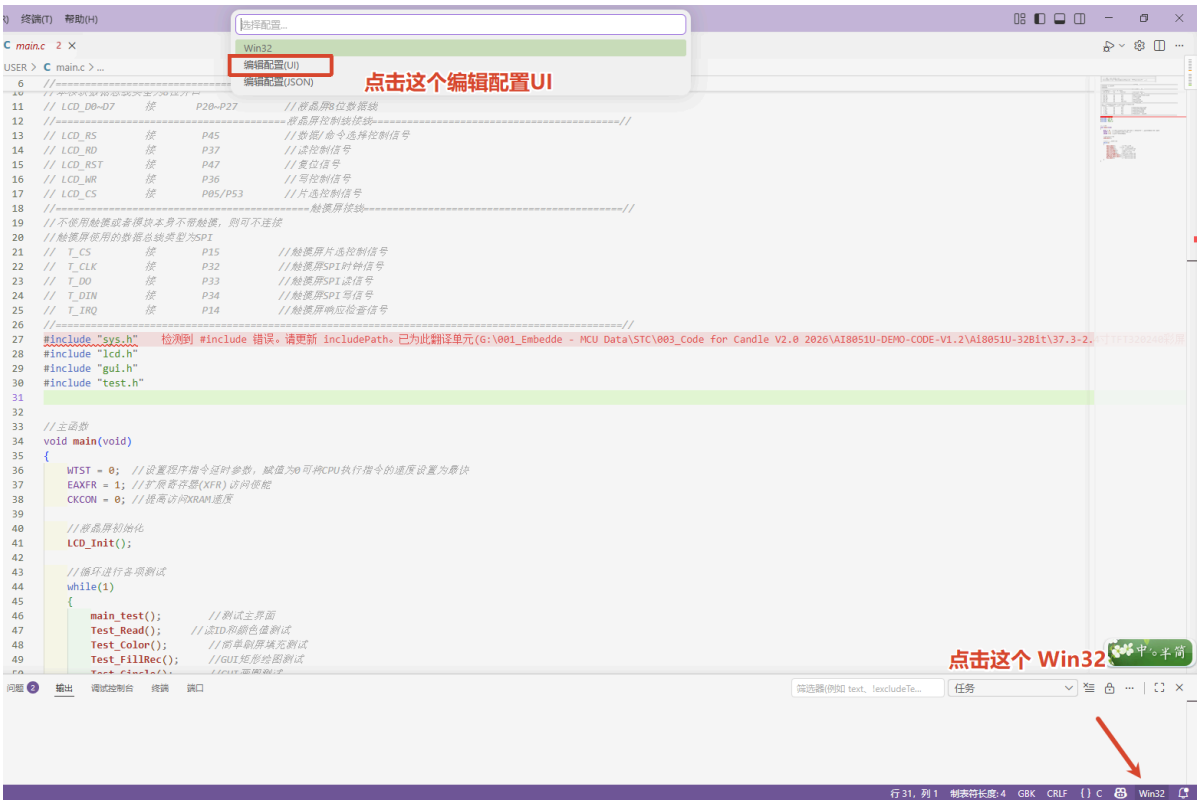




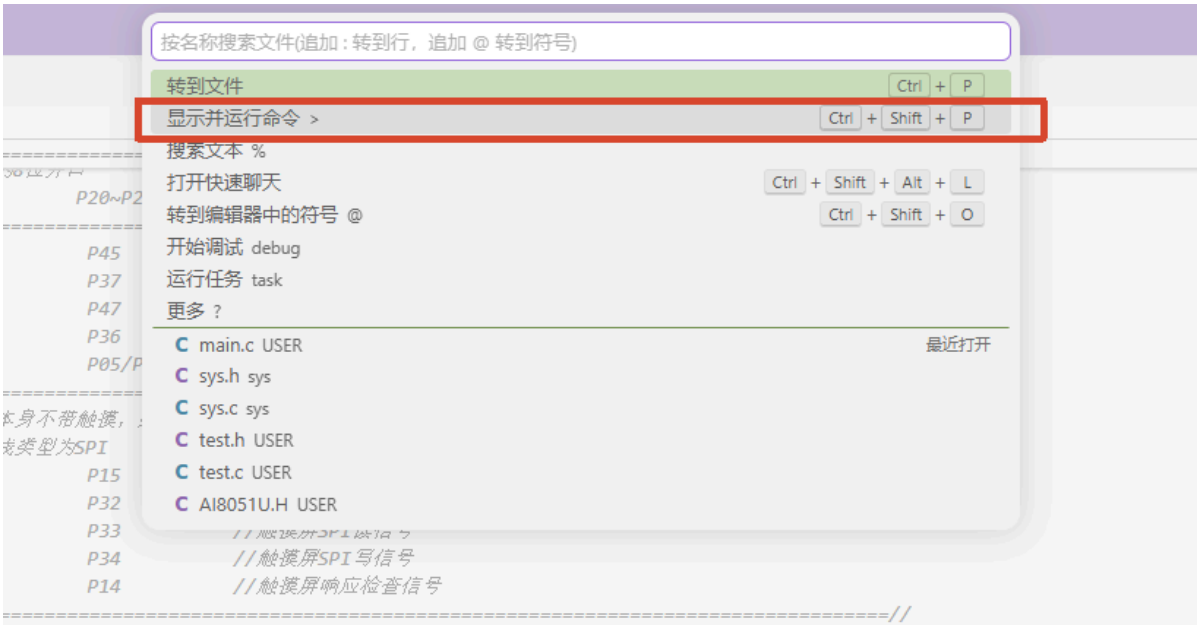
4.无法找到头文件的报错（未安装keil插件的情况，麻烦！每次打开新工程都要重新配置。安装了请看本条目的第5点）

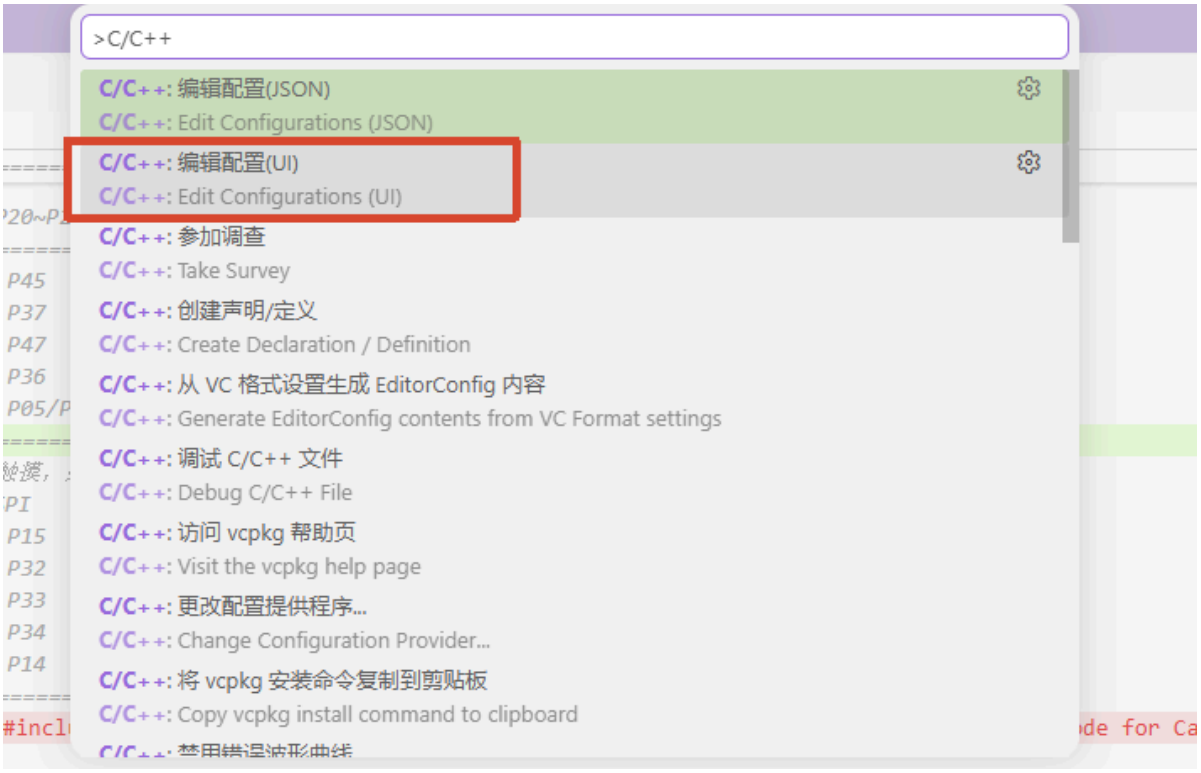
这个报错是因为没有告诉VScode头文件的位置在哪里，所以会报错，但实际不会印象keil5编译。

接下来打开VScode的配置文件.。

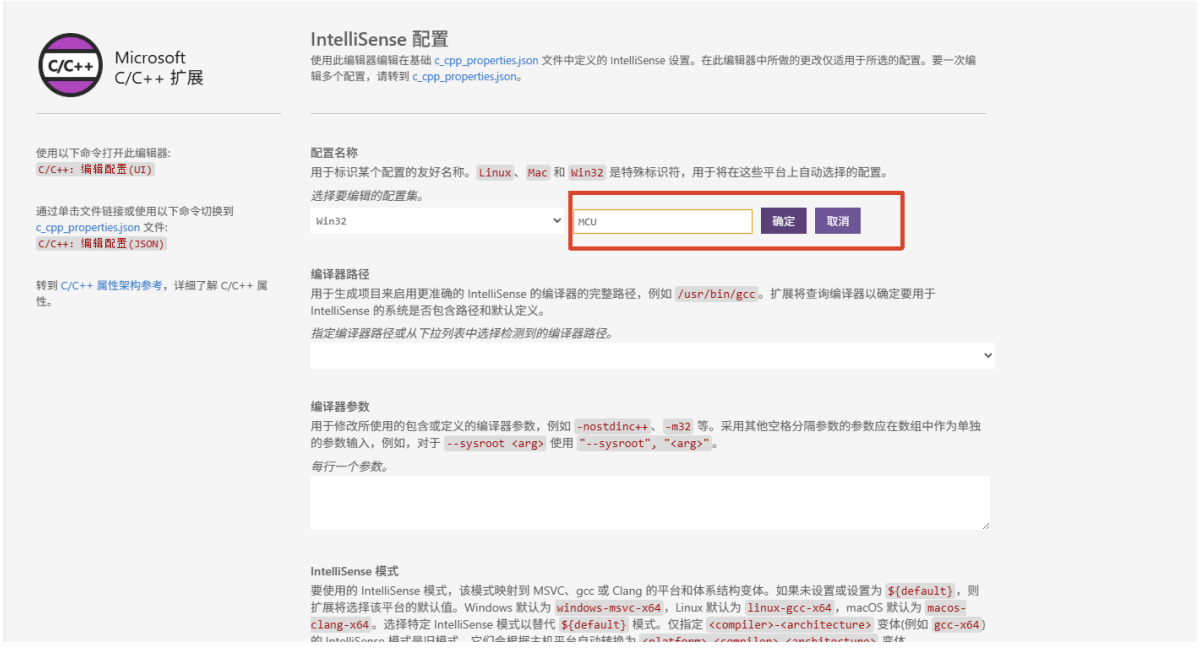


或者点击上方搜索栏，点击显示并运行命令，然后搜索C/C++
或者使用快捷键 Ctrl+shift+P，然后搜索C/C++
或者直接在上方搜索栏，搜索>C/C++

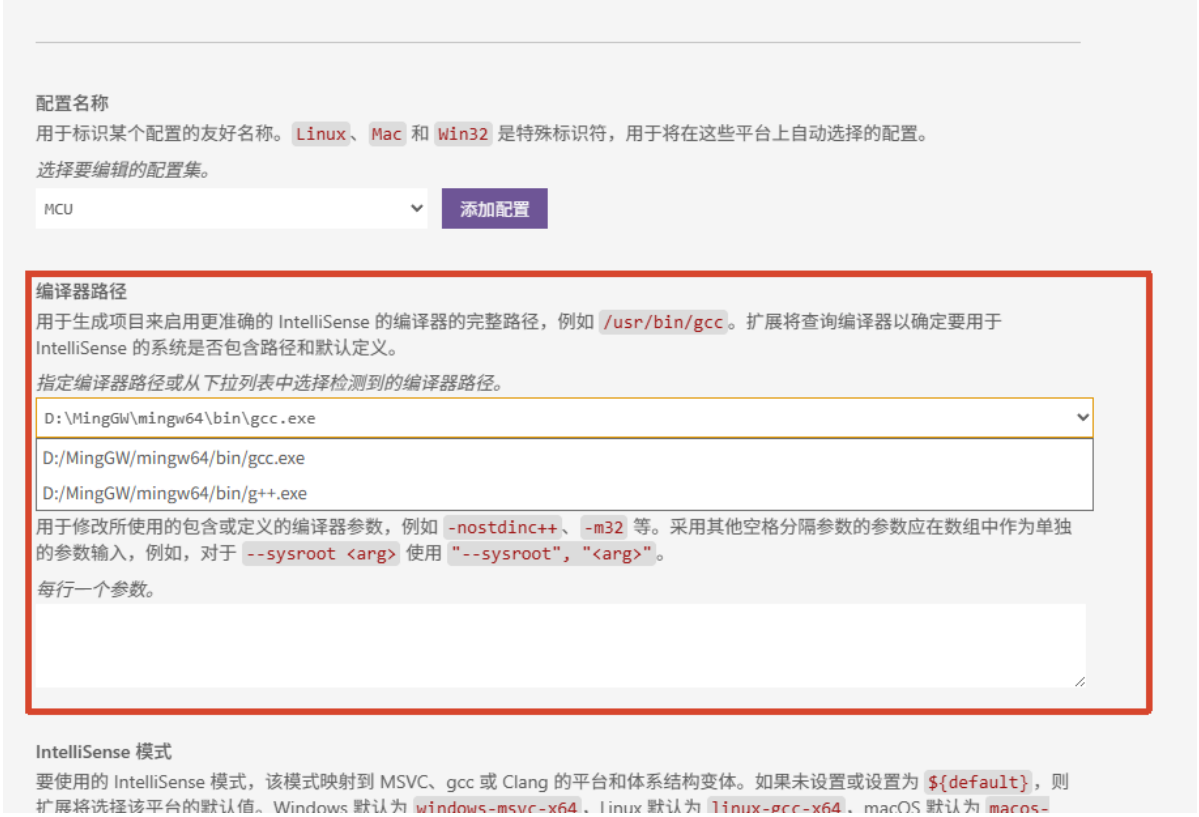




可以添加一个新的配置，名为 MCU。也可以直接修改自动生成的win32配置，因为每次打开一个新的工程都要重新配置。



如果你的电脑中安装了GCC编译器（MingGW），这里会自动识别出来。
如果没有安装，空着也没事，毕竟我们是使用keil5编译的，这一步的目的是为了消除头文件报错



然后在 包含路径 中加入keil5的头文件路径，修改成自己的，在后面补上 /**

```
D:/Keil_v5/C51/INC/**
D:/Keil_v5/C251/INC/**
```

然后在 定义 中加入keil5的常用定义，可以消除 xdata 等在VScode中的报错

```
__C251__
__VSCODE_C251__
reentrant=
compact=
small=
large=
data=
idata=
pdata=
bdata=
edata=
xdata=
```

```
code=
bit=char
sbit=char
sfr=char
sfr16=int
sfr32=int
interrupt=
using=
far=
_at_=
_priority_=
_task_=
```

定义

分析文件时 IntelliSense 引擎要使用的预处理器定义的列表。(可选)使用 `=` 设置值，例如 `VERSION=1`。

每行一个定义。

```
__C251__
__VSCODE_C251__
reentrant=
compact=
small=
large=
data=
idata=
pdata=
bdata=
edata=
xdata=
code=
bit=char
sbit=char
sfr=char
sfr16=int
sfr32=int
interrupt=
using=
far=
_at_=
_priority_=
_task_=
```

最终效果如下

IntelliSense 配置

使用此编辑器编辑在基础 c_cpp_properties.json 文件中定义的 IntelliSense 设置。在此编辑器中所做的更改仅适用于所选的配置。要一次编辑多个配置，请转到 c_cpp_properties.json。

配置名称

用于标识某个配置的友好名称。`Linux`、`Mac` 和 `Win32` 是特殊标识符，用于将在这些平台上自动选择的配置。

选择要编辑的配置集。

MCU 添加配置

编译器路径

用于生成项目来启用更准确的 IntelliSense 的编译器的完整路径，例如 `/usr/bin/gcc`。扩展将查询编译器以确定要用于 IntelliSense 的系统是否包含路径和默认定义。

指定编译器路径或从下拉列表中选择检测到的编译器路径。

编译器参数

用于修改所使用的包含或定义的编译器参数，例如 `-nostdinc++`、`-m32` 等。采用其他空格分隔参数的参数应在数组中作为单独的参数输入，例如，对于 `--sysroot <arg>` 使用 `"--sysroot", "<arg>"`。

每行一个参数。

IntelliSense 模式

要使用的 IntelliSense 模式，该模式映射到 MSVC、gcc 或 Clang 的平台和体系结构变体。如果未设置或设置为 `${default}`，则扩展将选择该平台的默认值。Windows 默认为 `windows-msvc-x64`，Linux 默认为 `linux-gcc-x64`，macOS 默认为 `macos-clang-x64`。选择特定 IntelliSense 模式以替代 `${default}` 模式。仅指定 `<compiler>-<architecture>` 变体(例如 `gcc-x64`)的 IntelliSense 模式是旧模式，它们会根据主机平台自动转换为 `<platform>-<compiler>-<architecture>` 变体。

windows-gcc-x64

包含路径

include 路径是包括源文件中随附的头文件(如 `#include "myHeaderFile.h"`)的文件夹。指定 IntelliSense 引擎在搜索包含的头文件时要使用的列表路径。对这些路径进行的搜索不是递归搜索。指定 `**` 可指示递归搜索。例如，`${workspaceFolder}/**` 将搜索所有子目录，而 `${workspaceFolder}` 则不会。如果在安装了 Visual Studio 的 Windows 上，或者在 `compilerPath` 设置中指定了编译器，则无需在此列表中列出系统 include 路径。

每行一个包含路径。

```
${workspaceFolder}/**
D:/Keil_v5/C51/INC/**
D:/Keil_v5/C251/INC/**
```

定义

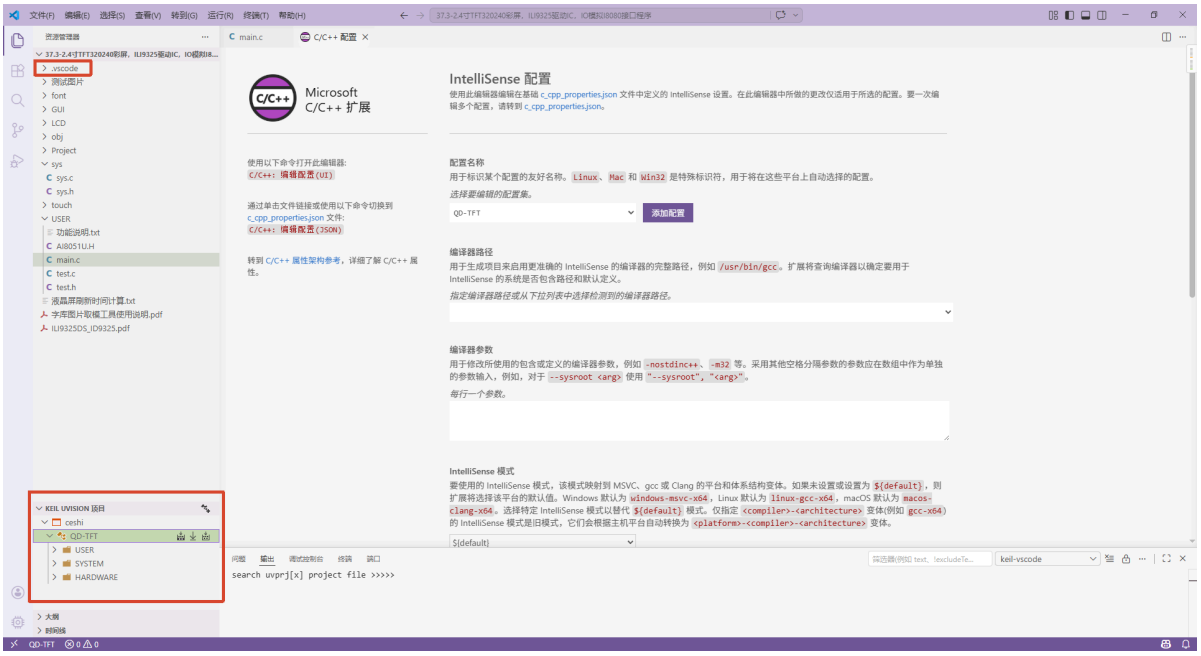
分析文件时 IntelliSense 引擎要使用的预处理器定义的列表。(可选)使用 `=` 设置值，例如 `VERSION=1`。

每行一个定义。

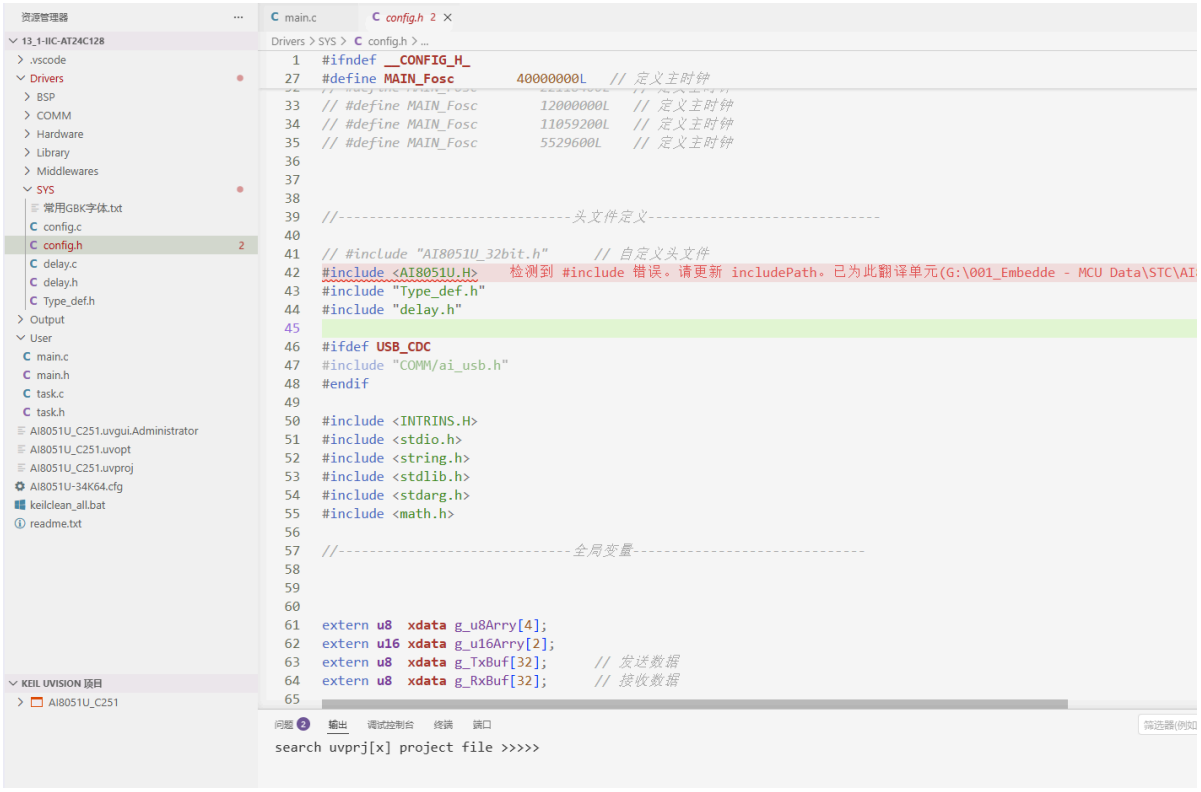
```
__C251__
__VSCODE_C251__
reentrant=
compact=
small=
large=
data=
idata=
pdata=
bdata=
edata=
xdata=
code=
bit=char
sbit=char
sfr=char
sfr16=int
sfr32=int
interrupt=
using=
far=
_at_=
_priority_=
_task_=
```

5.无法找到头文件的报错（安装了keil插件的情况）

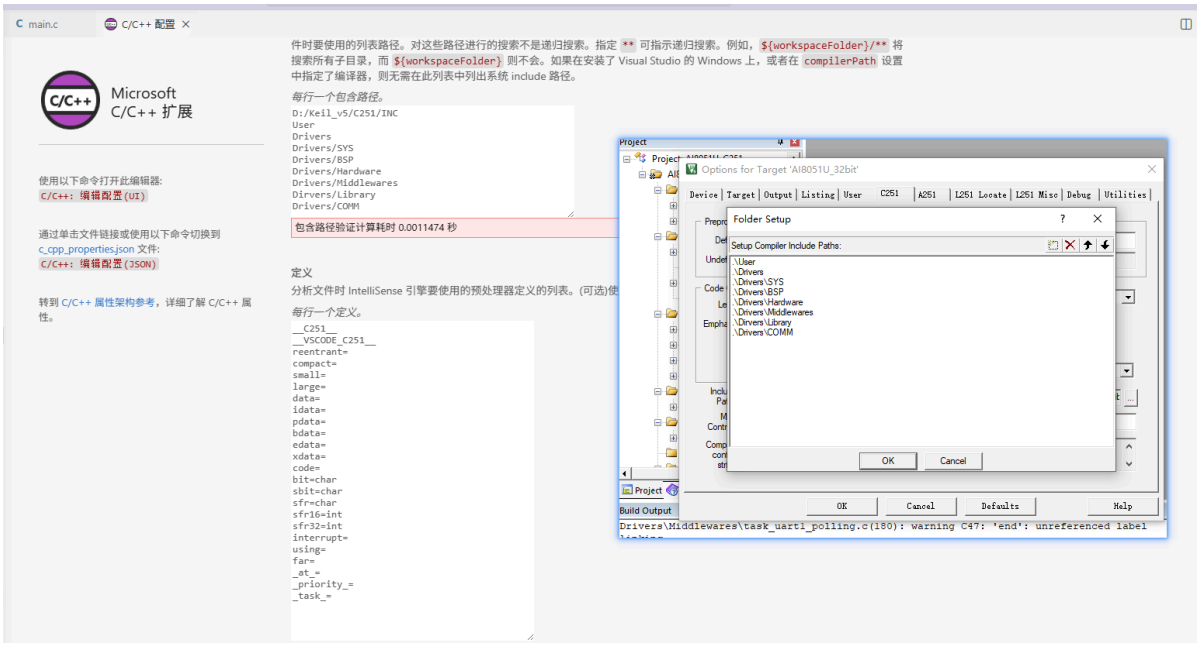
安装了keil插件，我们每次点开插件，都会自动生成第4点的配置，无需手动设置。



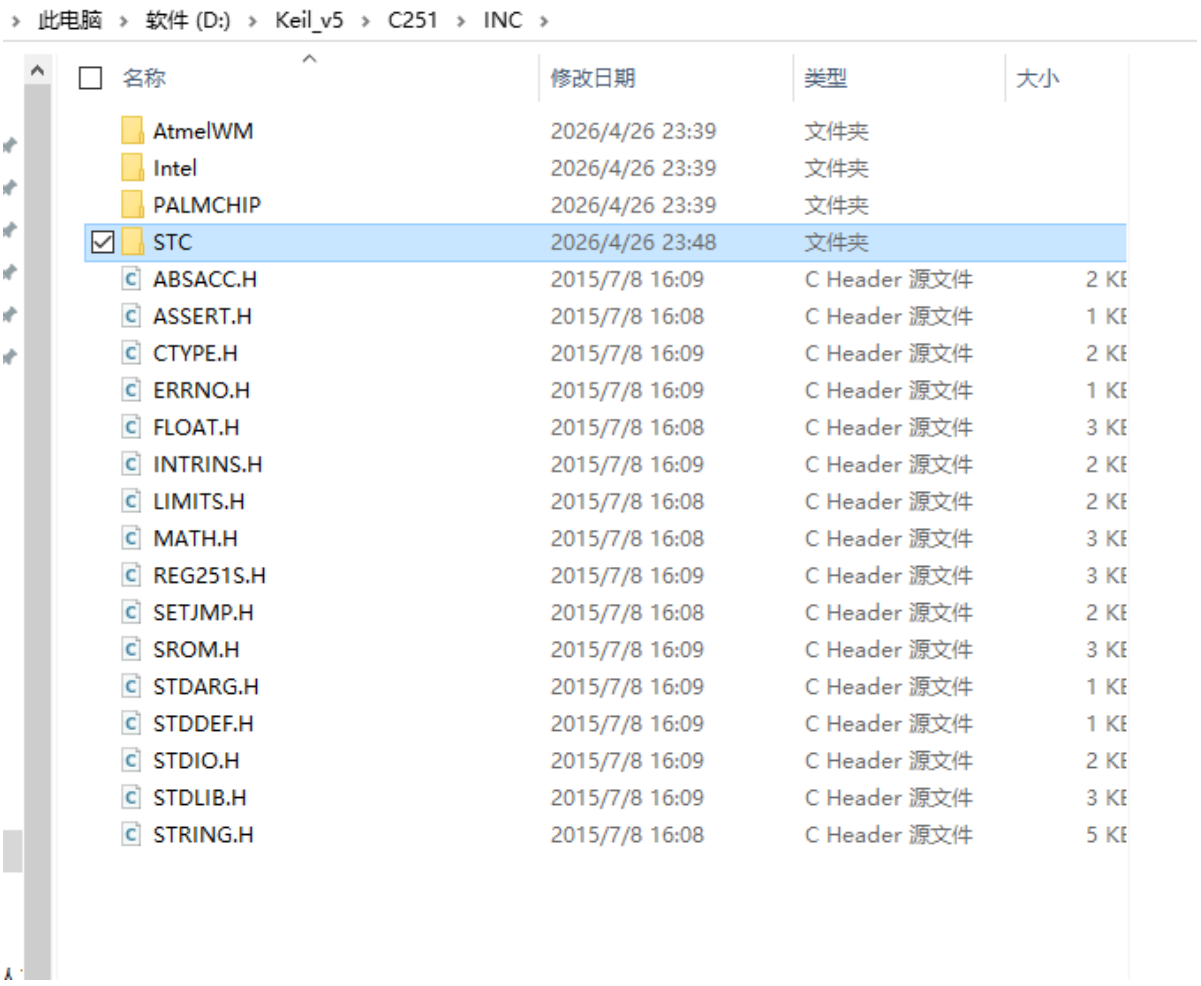
但有时候还是会出现这样的报错，这是怎么回事呢?



我们只需打开keil5的头文件目录便知晓。插件自动添加的头文件路径跟keil5中的一模一样，且多了一个C251的头文件路径

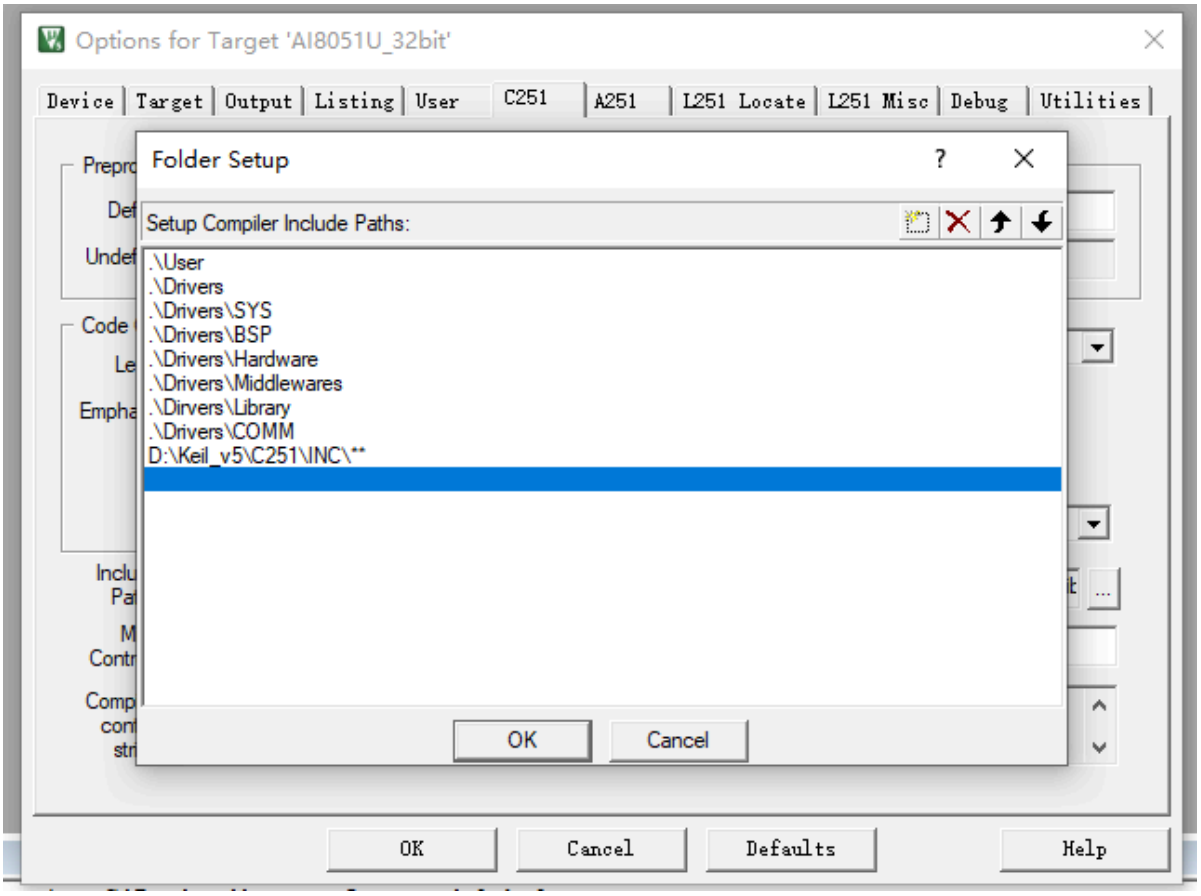


STC的头文件也在这个目录的子文件中，而想要调用子文件中的文件，只需在路径后补上 斜杠 星星 星星即可。



因此我们直接在创建keil5工程时，顺手在头文件路径中补上带斜杠 星星 星星的路径即可。

```
D:/Keil_v5/C51/INC/**
D:/Keil_v5/C251/INC/**
```



然后keil5点一下编译，vscode也会瞬间同步。

此时报错便消失了

唯一的遗憾是 Vscode 无法消除中断函数的报错

```
void Task_Marks_Handler_Callback(void);
void Timer0_Isr(void) interrupt TMR0_VECTOR     应输入“{”
{
    Task_Marks_Handler_Callback();  // 系统计时 1ms
}
```

总结

Vscode的玩法有很多，我这里介绍的也只是冰山一角。最后还是建议同志们使用VScode作为代码的编辑器，keil5作为程序的编译器使用。因为在编译调试方面而言，keil5都是完胜vscode的，当然不排除有大神能写出优于keil5的编译链方法。

其次Vscode中也有很多AI编程的插件，但现阶段而言，我认为Ai提供的程序还是在参考或部分可用的阶段，如果完全使用AI开发的程序做项目，后期出了问题都无从下手，让ai自己修复也可能会越改越多BUG。因此只推荐各位使用AI作为助手，辅助修改那些重复的工作，或咨询不明白的问题，本人在此不做Ai相关的插件推荐，各位请自行查找合适自己的AI助手进行安装使用。