

第一讲、VScode 编辑器软件安装

实验概况

安装 VScode 软件及各个支持库，完成对 Verilog 代码自动补全和语法错误检查。

实验目的

熟悉软件安装方法，为开发做好充足准备

软件

VScode、以及各个插件

硬件

电脑

目录

第一讲、VScode 编辑器软件安装.....	1
1. 软件基本介绍:	2
2. VSCode 软件安装	2
2.1 VSCode 软件安装	2
2.2 VSCode 编辑器设置中文环境.....	5
3. 插件安装.....	8
3.1 verilog 语法高亮和自动例化功能.....	8
3.2 iverilog 语法检查功能.....	13
3.3 生成 testbench 模板功能.....	18
3.4 使用代码片段生成文件头.....	23
3.5 绘制波形插件.....	26
3.6 绘图组件.....	27
3.7 颜色主题更改.....	28
4. 关联 ISE 软件.....	29
5. 关联 vivado 软件.....	30



1. 软件基本介绍:


VSCode 是一款免费开源的现代化轻量级代码编辑器，在 IT 开发领域有广泛应用，在语法高亮、代码补全等功能表现很出色。

作为工程师通常在使用开发环境的时候都会选择一个适合自己习惯和审美的编辑器。Vs Code 作为微软的编辑器曾被称为良心，它是一款免费开源的代码编辑器，支持现今主流的开发语言，支持插件扩展，可以在应用商店中直接找到对应的插件工具进行安装，并且卸载也十分方便，如果大家有兴趣可以自行设计喜欢的插件来改变 Vs Code 的界面和功能。使自己的开发环境与众不同且灵活高效

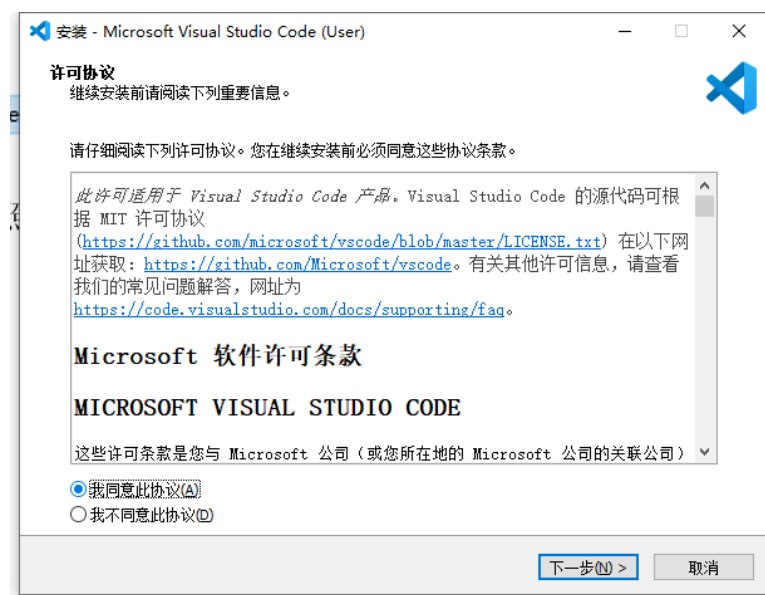
2. VSCode 软件安装

2.1 VSCode 软件安装

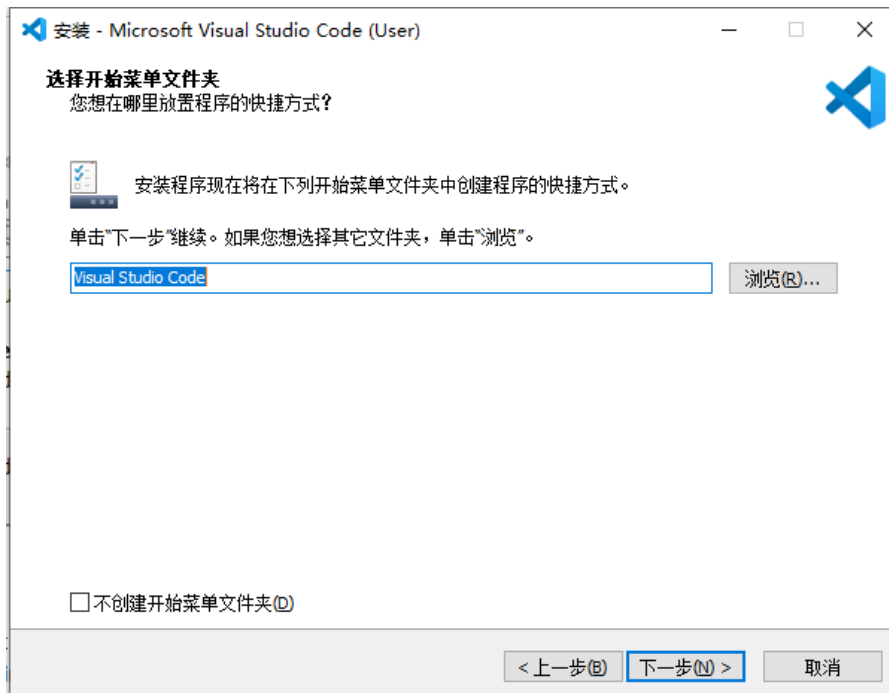
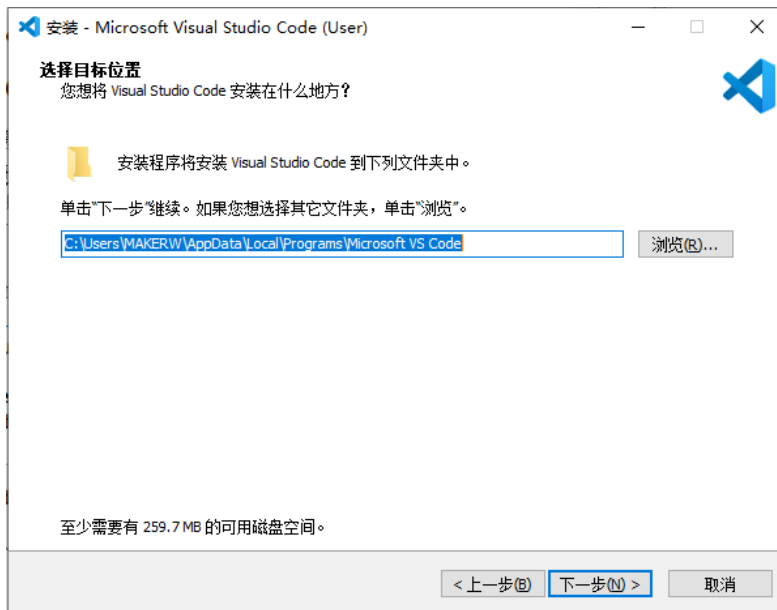
第一步双击运行如下安装程序（Tools 目录下提供了）

名称	修改日期	类型	大小
 VSCodeUserSetup-x64-1.53.2.exe	2021/2/20 9:15	应用程序	68,907 KB

第二步按照截图设置安装选项

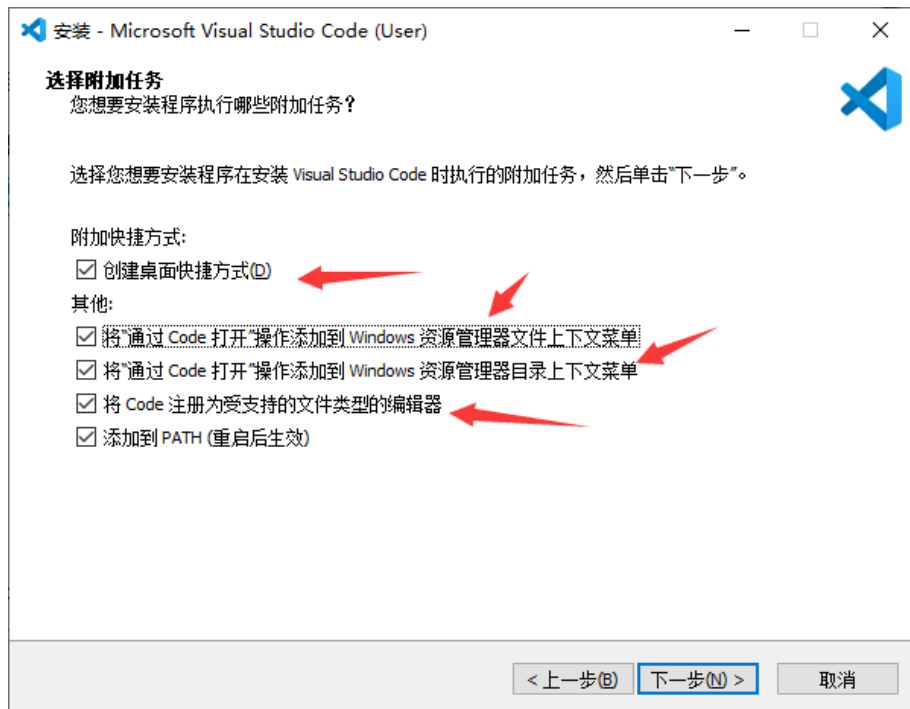


默认路径即可设置安装路径，不要中文目录

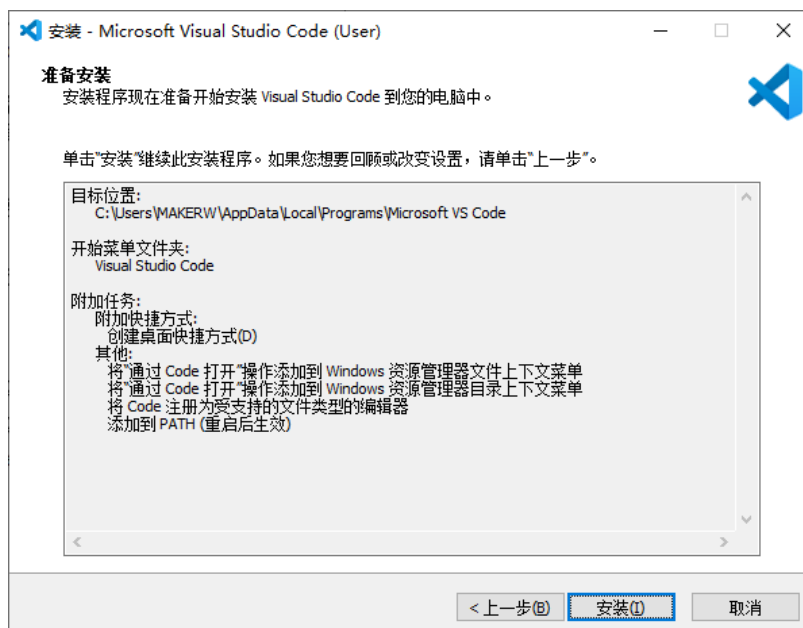


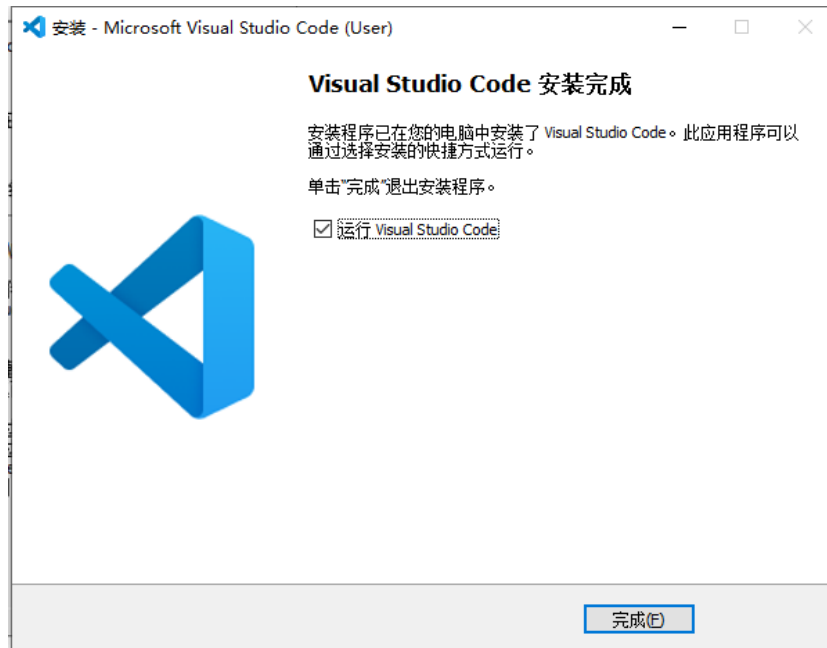
勾选如图的这些选项





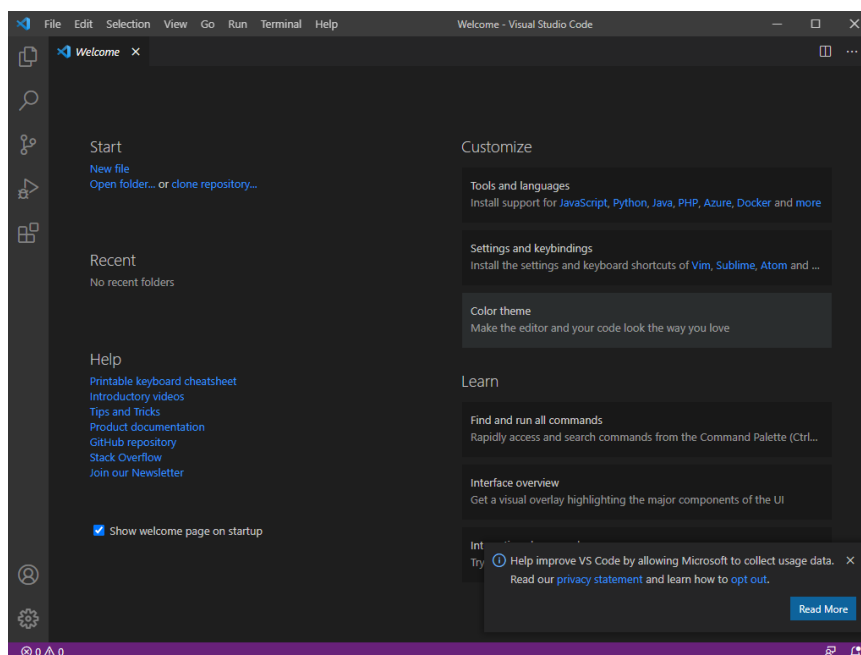
选择安装





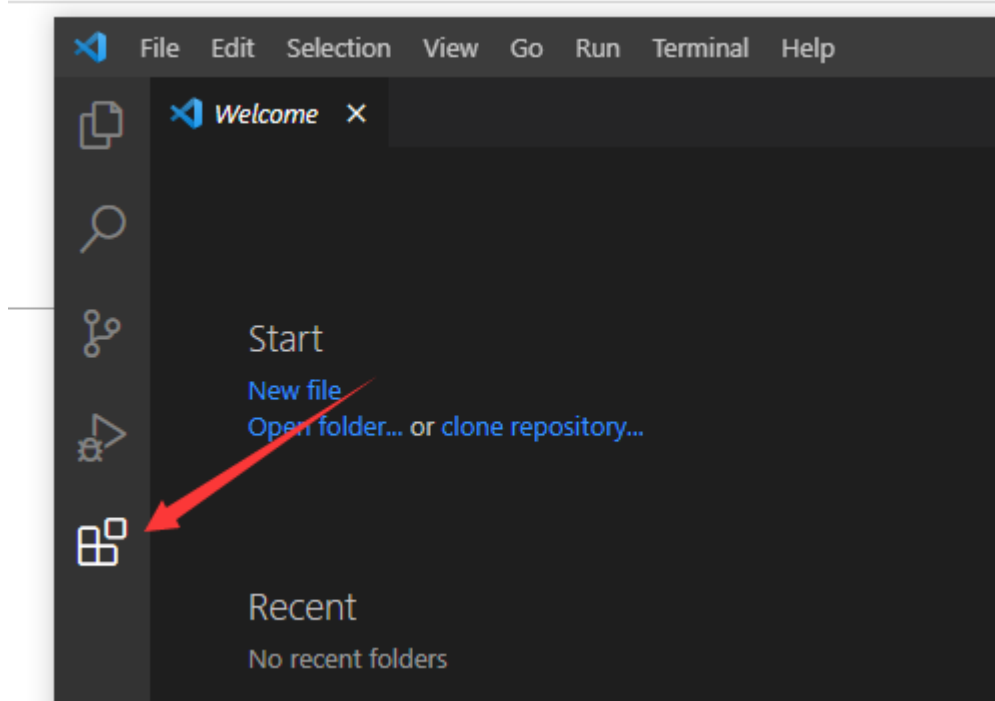
2.2 VScode 编辑器设置中文环境

打开 vscode 编辑器

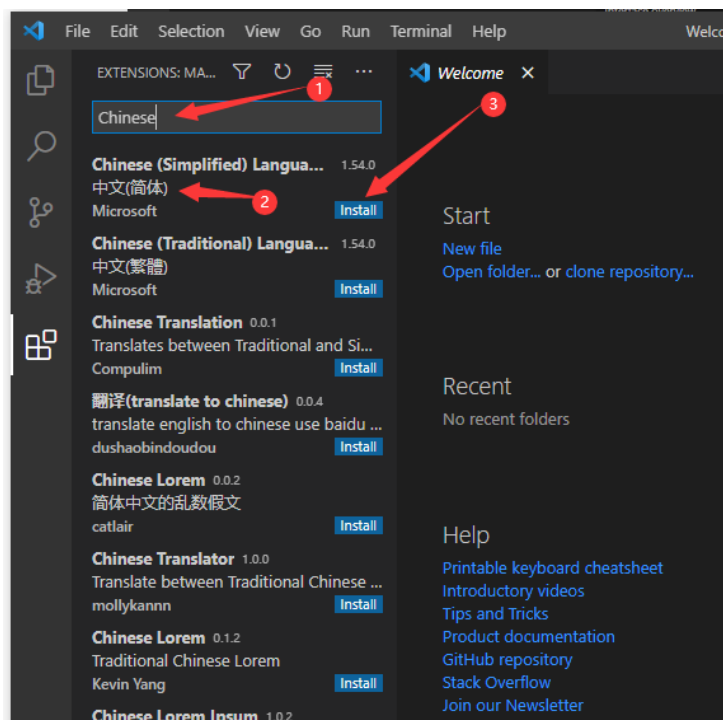


点击左侧工具栏的 extensions 或者使用快捷键【Ctrl+Shift+X】, 输入 chinese, 点击 Install 安装中文简体



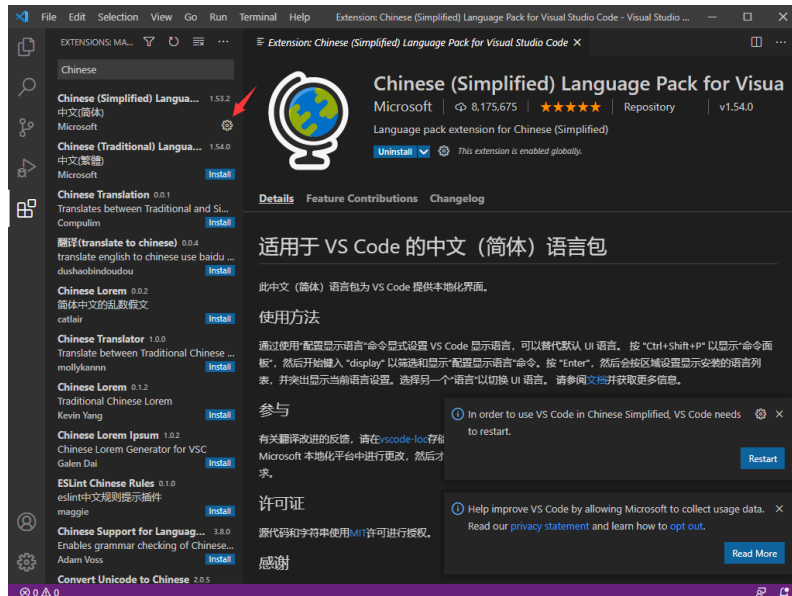


安装中文简体语言环境



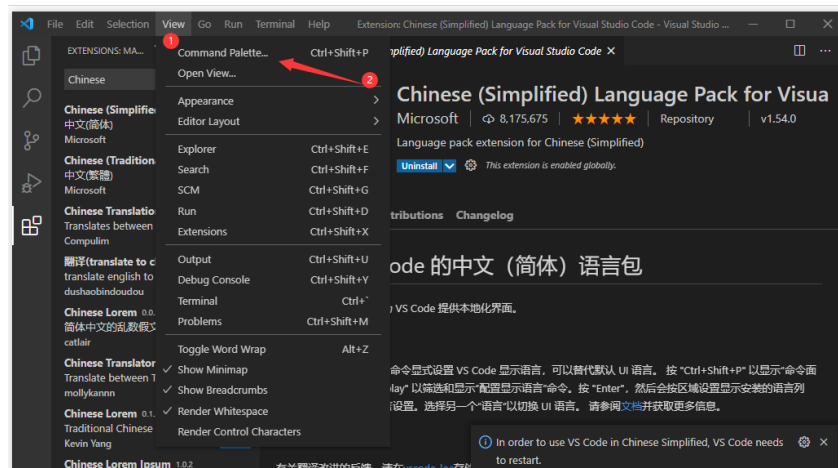
显示如下界面安装完成



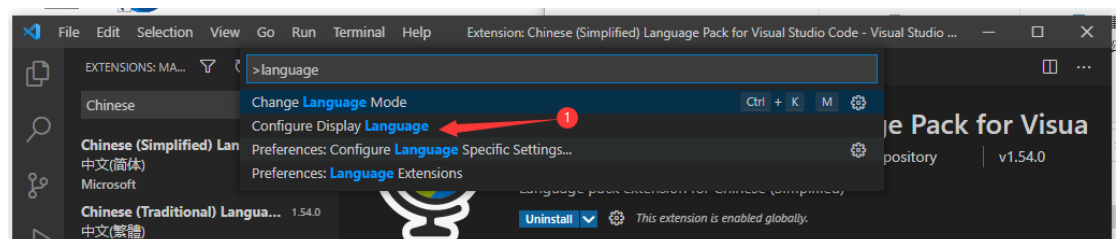


配置语言环境

使用快捷键【Ctrl+Shift+P】弹出查找命令框，输入 language，找到 Configure Display Language，点击，选择 locale 属性为"zh-CN"，如下图所示：

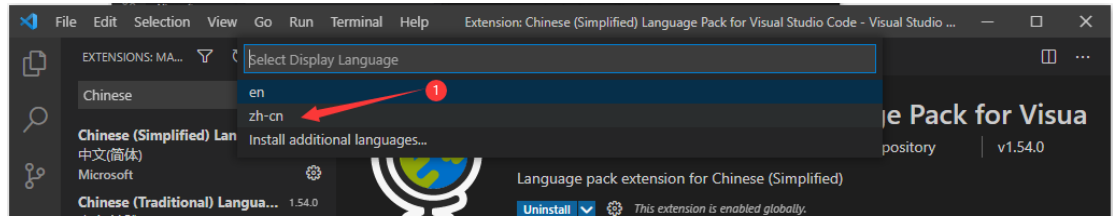


输入 language 搜索配置选项然后选中如图所示 Configure Display Language

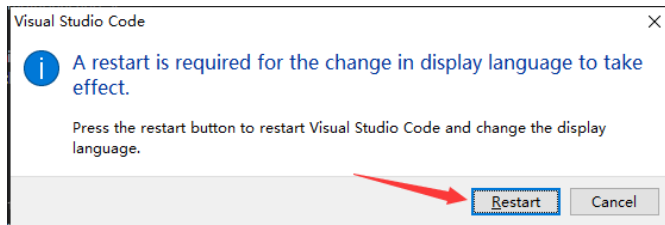


选择 ZH-CN

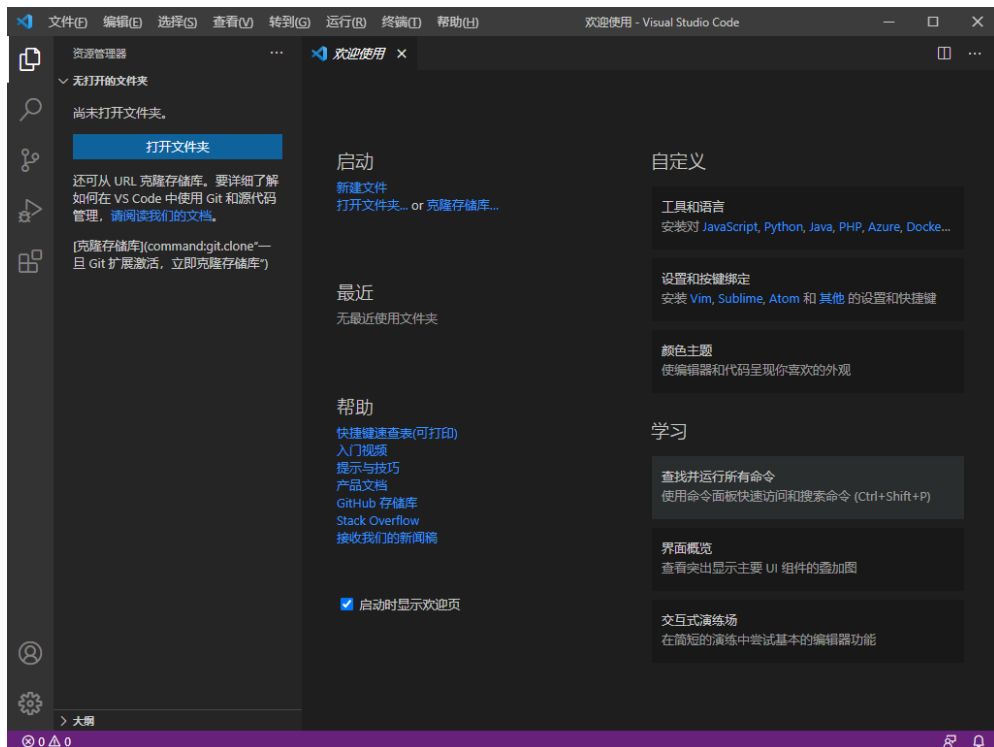




重启软件显示中文语言包



重新打开后显示中文



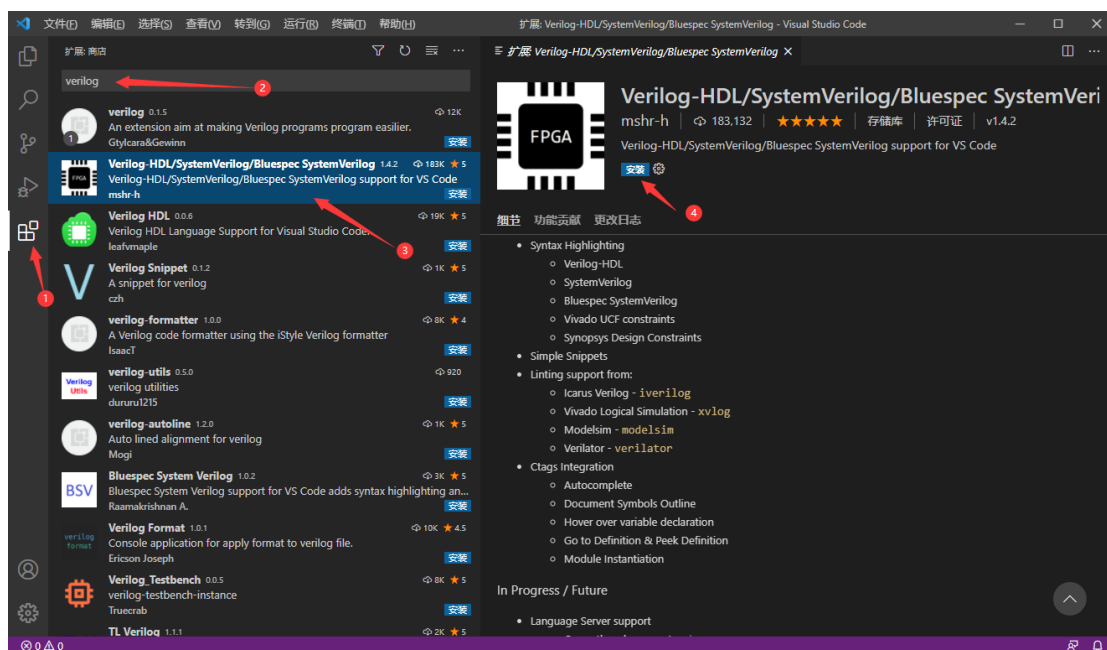
到此 VSCODE 安装成功，接下来安装插件！

3. 插件安装

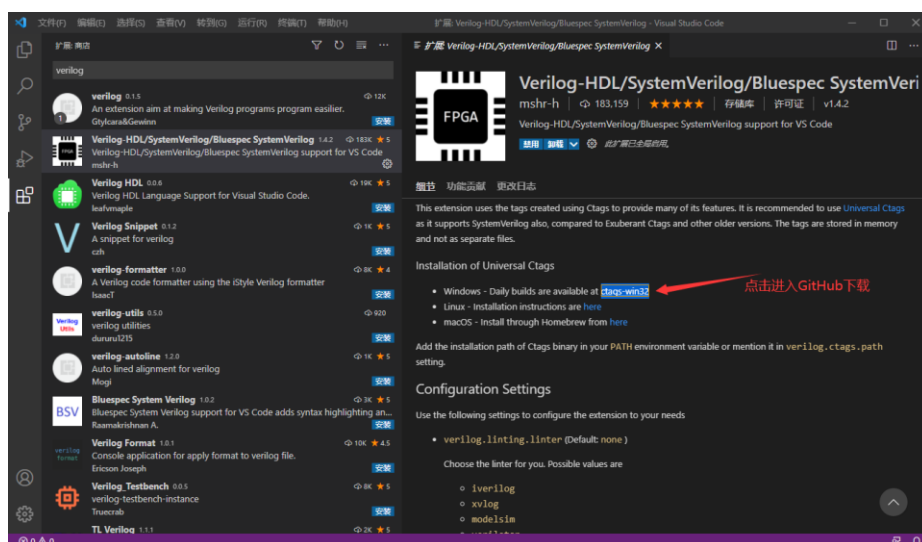
3.1 verilog 语法高亮和自动例化功能

插件名称 **Verilog-HDL/SystemVerilog/Bluespec SystemVerilog**
支持语法高亮、例化模板、UCF 和 XDC 的语法、verilog 语法检查等等。





下载 ctags-win32 插件，不想下载可以在我给出 tools 目录下找到

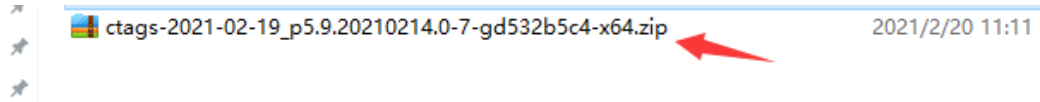


我再 C 盘建立如下 ctags 文件夹

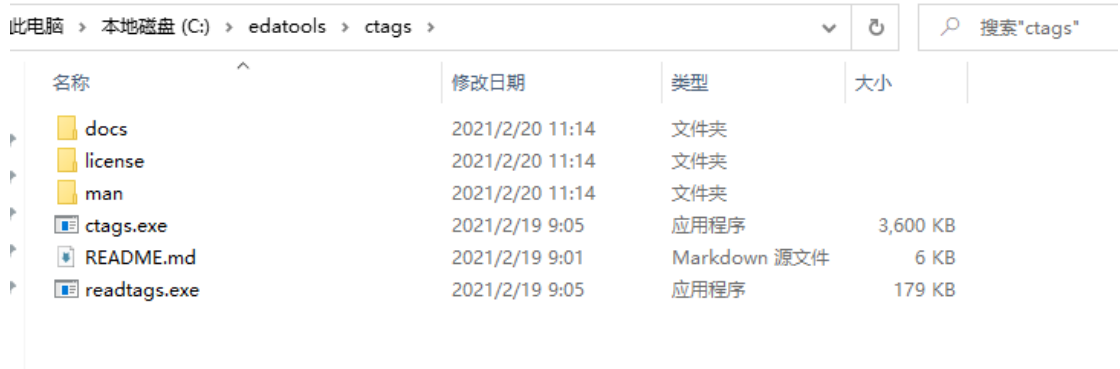


将下载的 ctags 压缩包解压到上边的 C 盘/edatools/ctags 目录

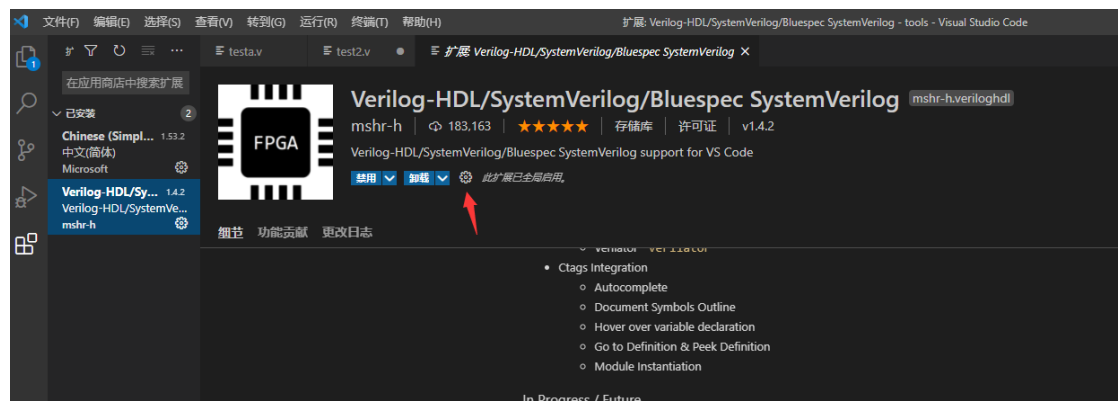




解压完成后

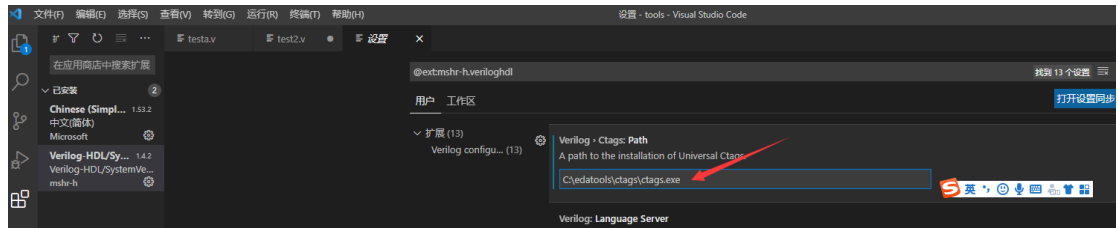


设置 ctags 的目录

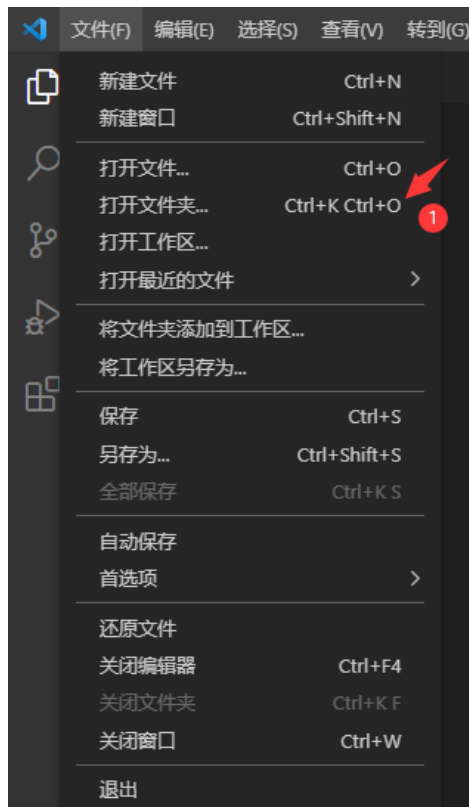


设置 ctags.exe 的解压路径，你解压到什么路径设置什么路径，否则无法支持自动例化功能

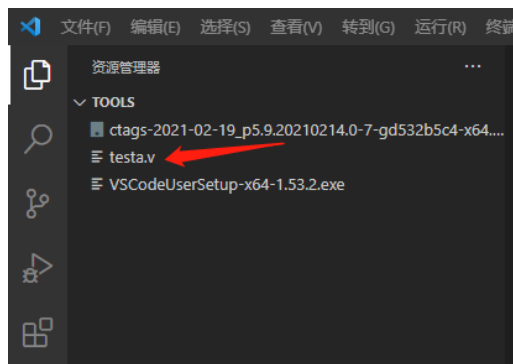


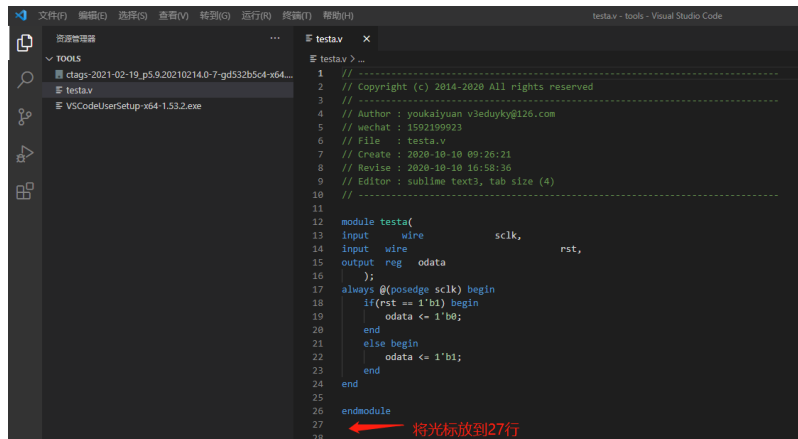


点击 file 打开文件夹

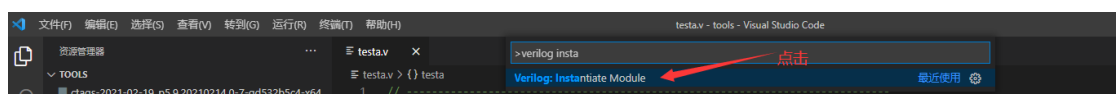


打开 tools 文件夹，双击打开 testa.v

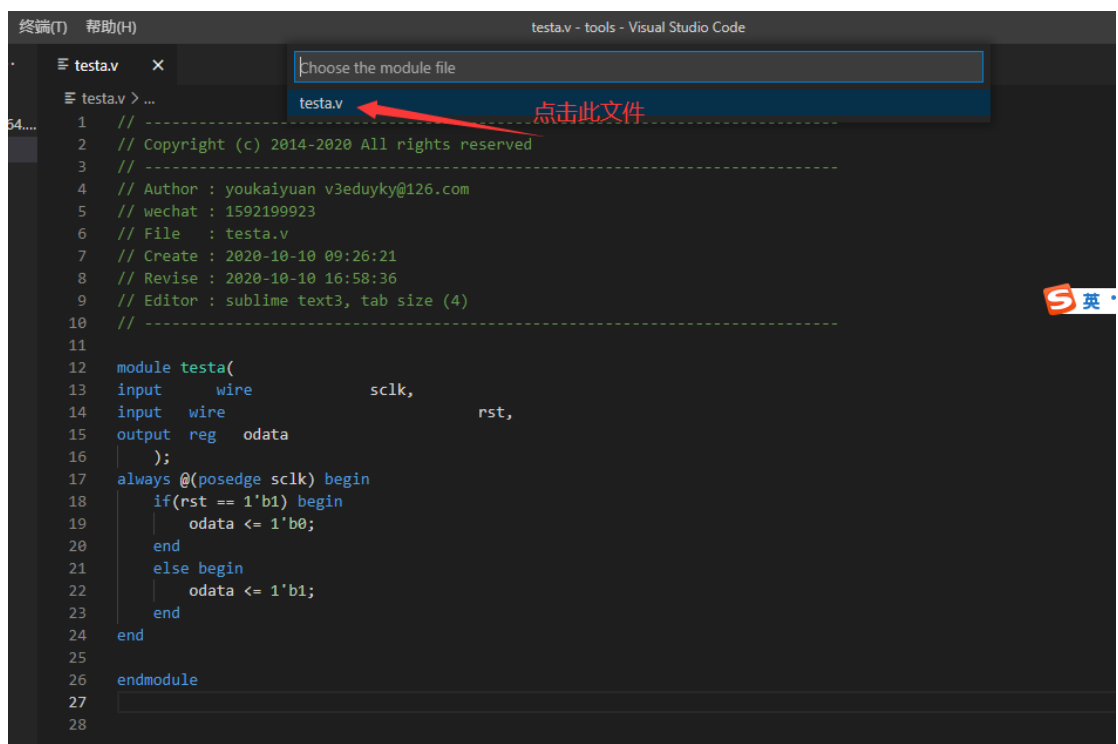




Ctrl+shift+p 调出命令行窗口输入 verilog insta 匹配出如下命令



选择要例化的模块. v



在第 27 行出现例化的代码



```

testa.v
testa.v > ...
1 // -----
2 // Copyright (c) 2014-2020 All rights reserved
3 // -----
4 // Author : youkaiyuan v3eduyky@126.com
5 // wechat : 1592199923
6 // File : testa.v
7 // Create : 2020-10-10 09:26:21
8 // Revise : 2020-10-10 16:58:36
9 // Editor : sublime text3, tab size (4)
10 // -----
11
12 module testa(
13     input wire          sclk,
14     input wire          rst,
15     output reg          odata
16 );
17 always @(posedge sclk) begin
18     if(rst == 1'b1) begin
19         odata <= 1'b0;
20     end
21     else begin
22         odata <= 1'b1;
23     end
24 end
25
26 endmodule
27
28 test u_testa(
29     .sclk (sclk ),
30     .rst  (rst  ),
31     .odata (odata )
32 );
33

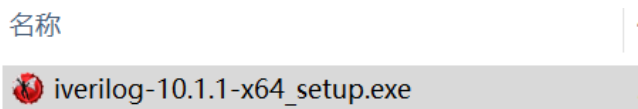
```

自动添加的例化并且选中例化名称
方便更改名称

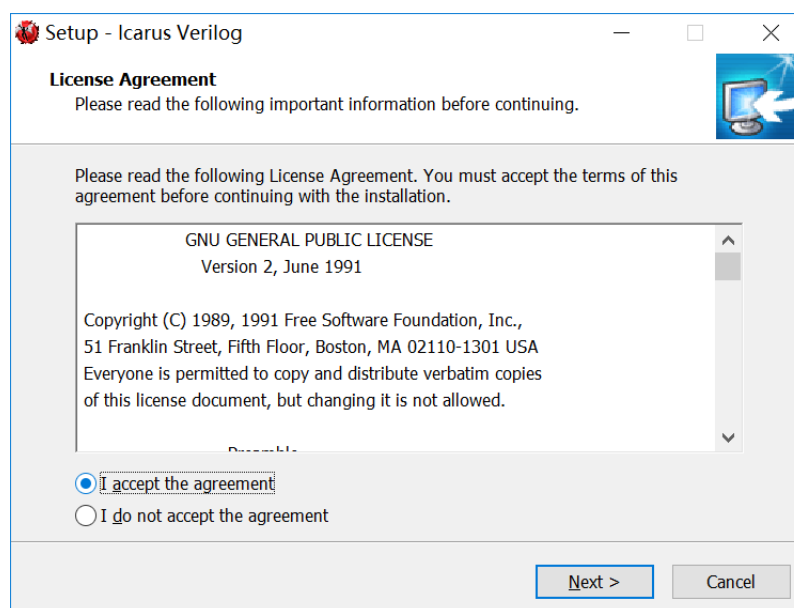
到此为止完成自动例化功能配置。

3.2 iverilog 语法检查功能

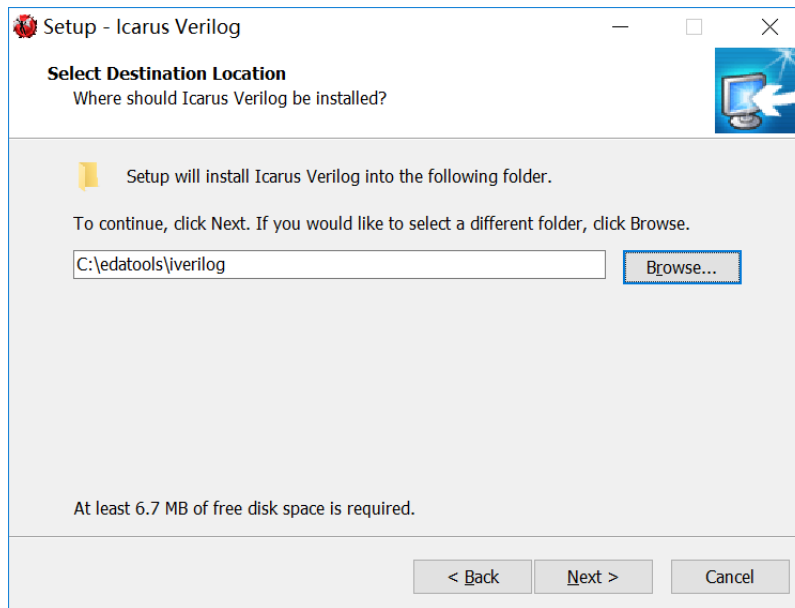
第一步以管理员身份运行如下安装程序



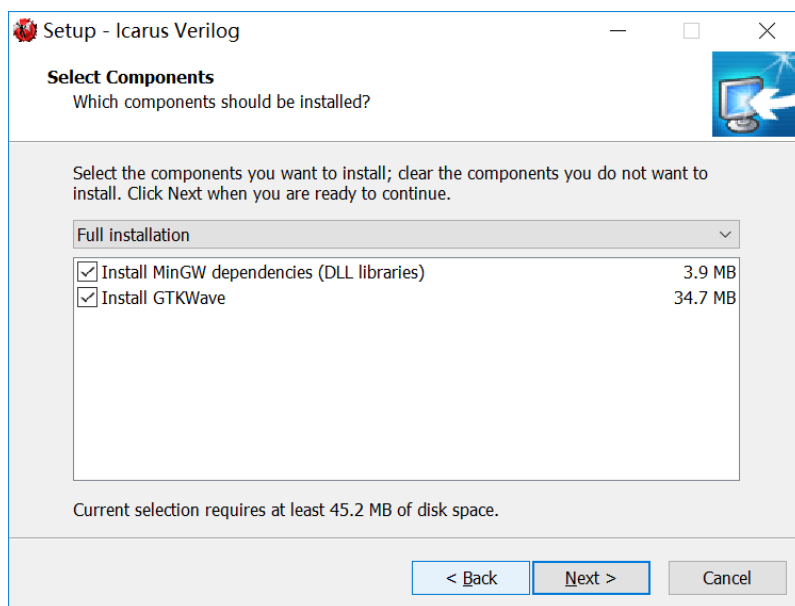
第二步按照截图设置安装选项

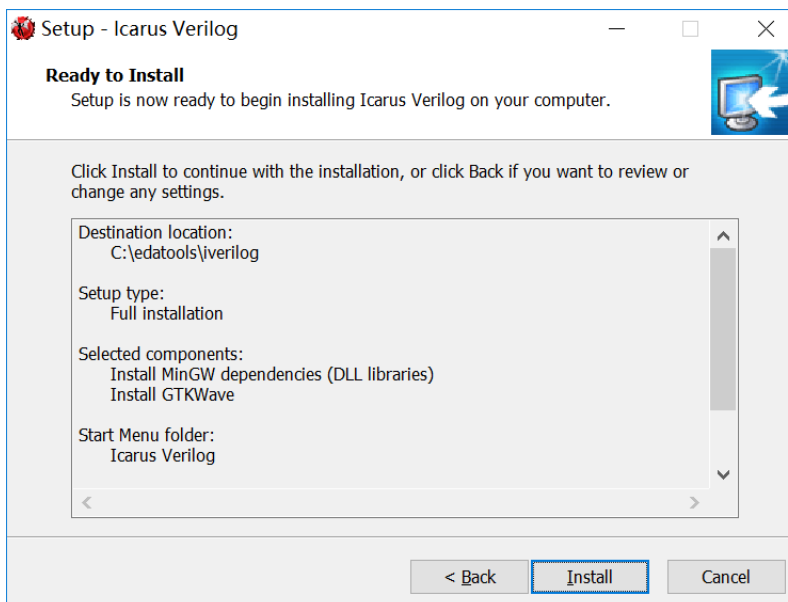
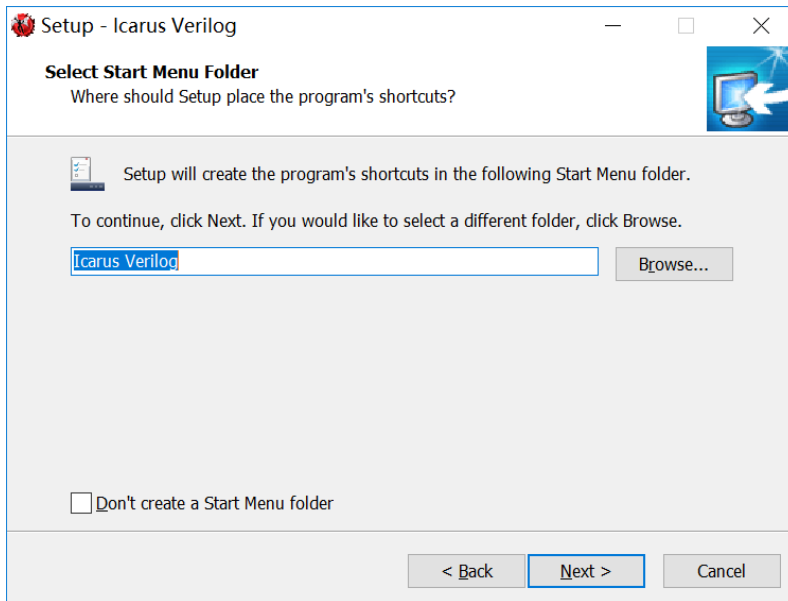


设置安装路径，不要中文目录



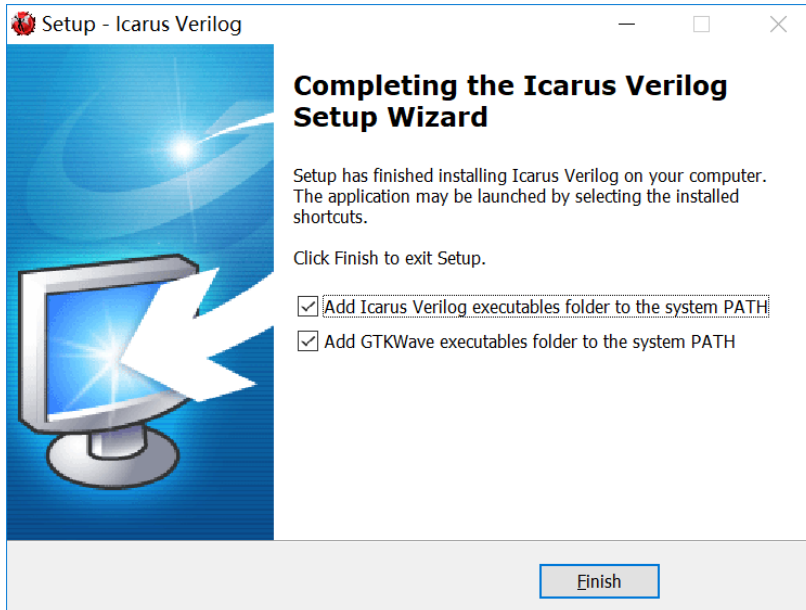
选择安装软件库





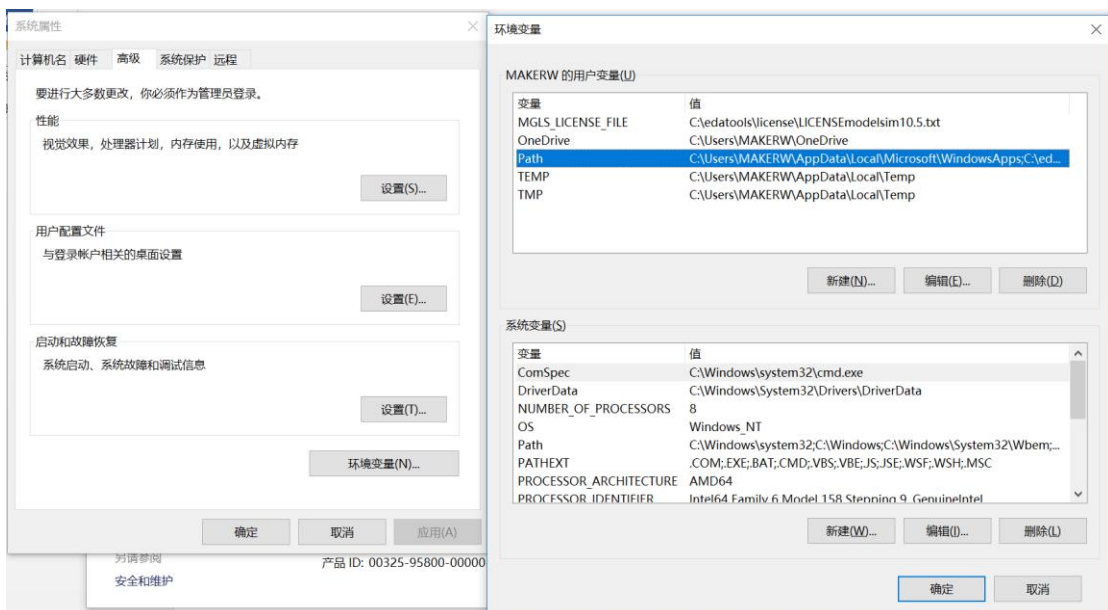
设置环境变量：（注意关闭杀毒软件否则添加不成功还需要手动添加）





手动添加环境变量方法:

右键点我的电脑->属性->高级->环境变量



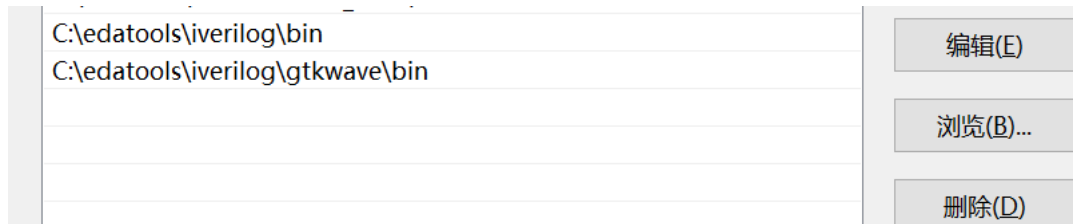
在 path 变量下加入如下属性

C:\xxxxx\iverilog\bin (xxxxx 根据自己目录选择)

C:\xxxxx\iverilog\gtkwave\bin (xxxxx 根据自己目录选择)

如下图我电脑设置





测试软件安装成功否

打开计算机开始->运行->CMD（回车）

打开 CMD 窗口后输入如下命令 iverilog

如果提示信息和如下一致表明安装成功，如果提示未找到 iverilog 命令，请检查环境变量是否设置成功。

```
iverilog: no source files.

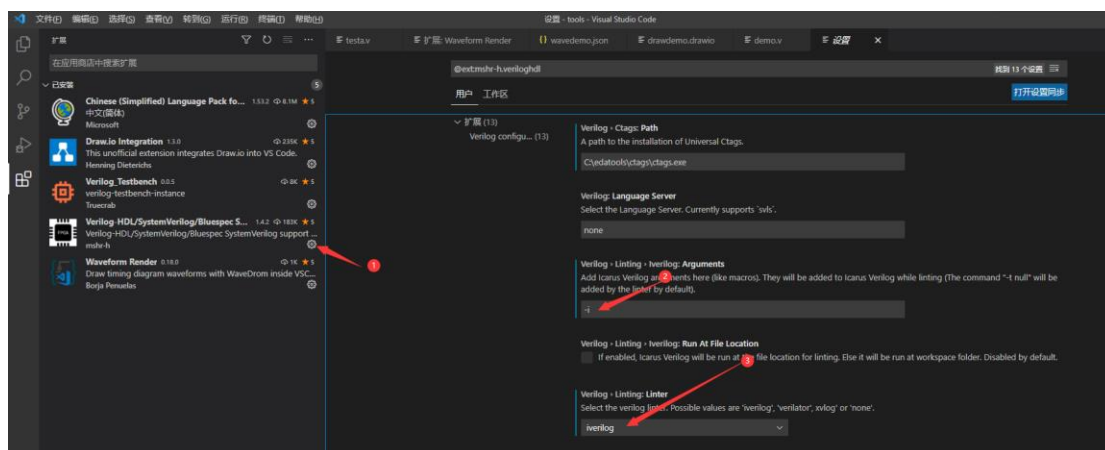
Usage: iverilog [-ESvV] [-B base] [-c cmdfile|-f cmdfile]
               [-g1995|-g2001|-g2005|-g2005-sv|-g2009|-g2012] [-g<feature>]
               [-D macro[=defn]] [-I includedir]
               [-M [mode=]depfile] [-m module]
               [-N file] [-o filename] [-p flag=value]
               [-s topmodule] [-t target] [-T min|typ|max]
               [-W class] [-y dir] [-Y suf] source_file(s)

See the man page for details.
```

打开已安装的插件设置 vscode

Arguments 输入-i

Linters 选择 iverilog



打开 testa.v

删除一个分号，CTRL+S 保存文件提示语法错误



```

testa.v x 设置
testa.v > {} testa
1 // -----
2 // Copyright (c) 2014-2020 All rights reserved
3 // -----
4 // Author : youkaiyuan v3eduyky@126.com
5 // wechat : 1592199923
6 // File : testa.v
7 // Create : 2020-10-10 09:26:21
8 // Revise : 2020-10-10 16:58:36
9 // Editor : sublime text3, tab size (4)
10 // -----
11
12 module testa(
13     input wire sclk,
14     input wire rst,
15     output reg odata
16 );
17 always @(posedge sclk) begin
18     if(rst == 1'b1) begin
19         odata <= 1'b0;
20     end
21     else begin
22         odata <= 1'b1
23     end
24 end
25 endmodule
26
27

```


到此为止语法检查配置完成。

3.3 生成 testbench 模板功能

安装 python 3 以上的环境

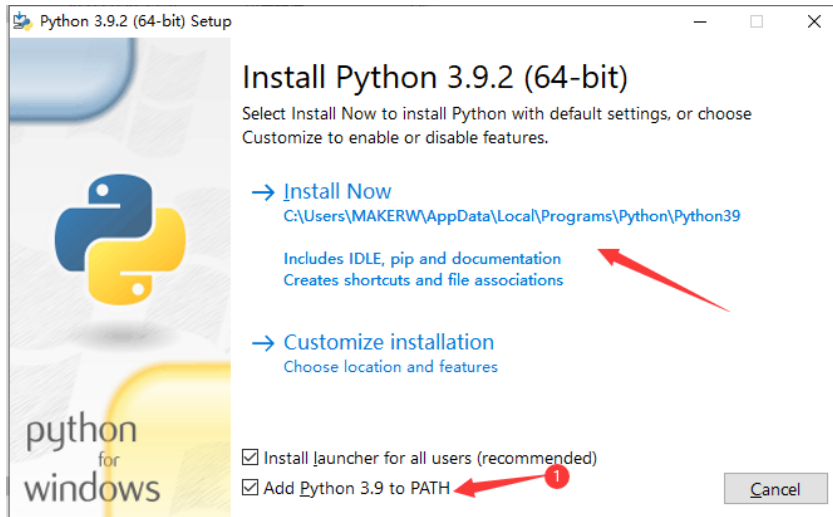
<https://www.python.org/downloads/release/python-373/>

tools 目录下点击安装

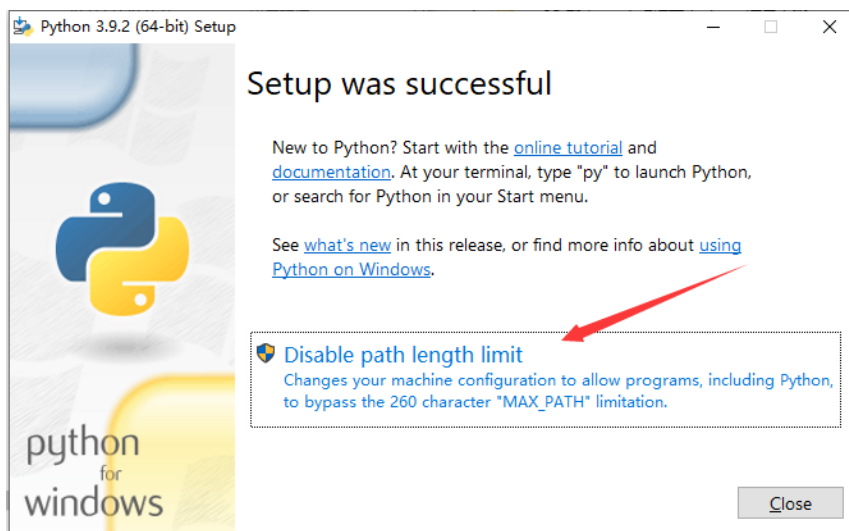
 python-3.9.2-amd64.exe

2021/2/20 17:36





取消路径长度字符限制



然后 close 掉就可以了！

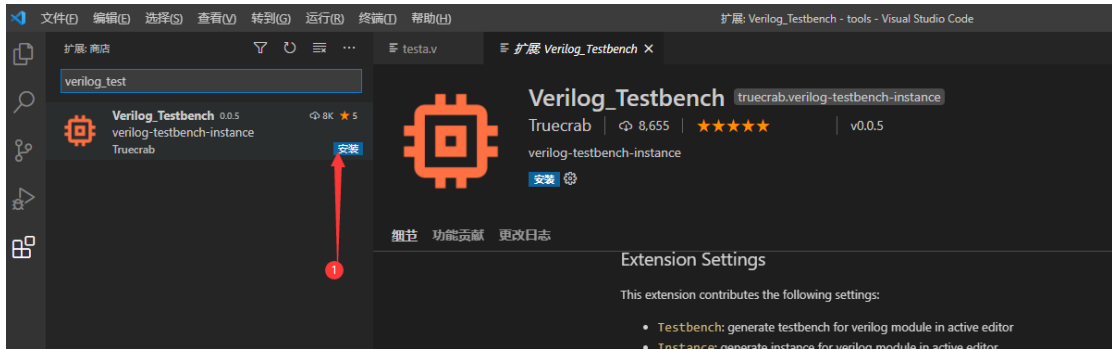
打开 cmd 窗口

在 cmd 窗口输入 python 即可验证是否安装成功！



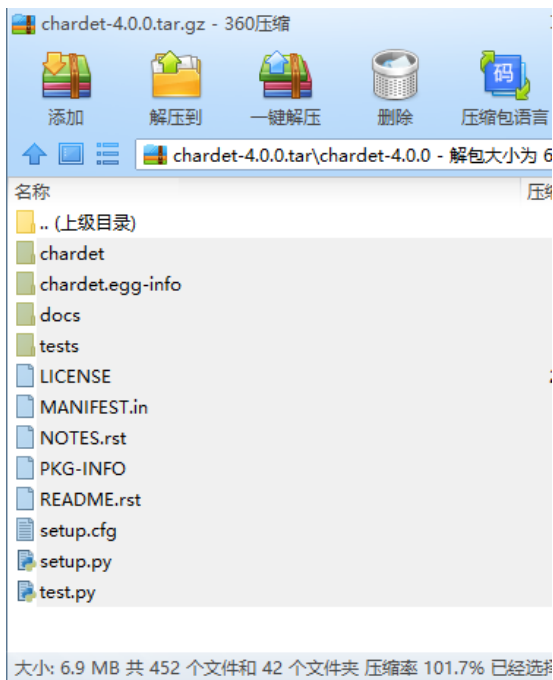
安装 verilog_testbench 插件，打开 vscode 的插件上电搜索



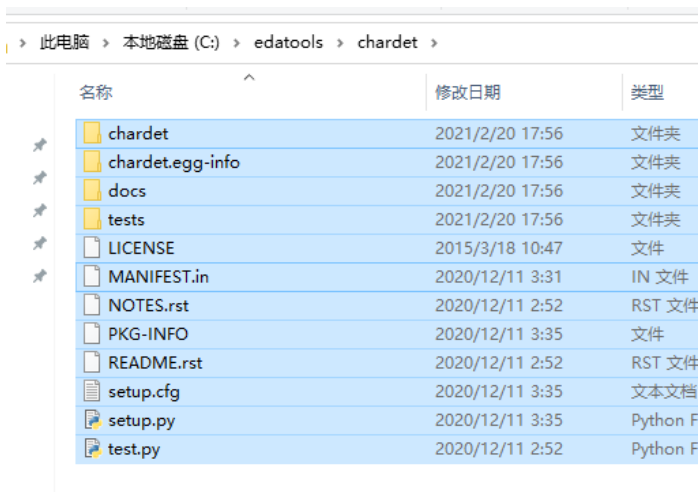


安装完毕后，打开 tools 目录，找到压缩包 chardet-4.0.0.tar.gz

我再 C:/edatools/目录下建立一个 chardet 的目录用来存储以上解压包的解压文件。



解压到新建的 chardet 目录



打开 cmd 窗口

```
命令提示符
Microsoft Windows [版本 10.0.19041.804]
(c) 2020 Microsoft Corporation. 保留所有权利。

C:\Users\MAKERW>
```

输入 `cd c:\` 然后回车

输入 `cd edatools/chardet` 回车这里是我的目录如果解压到其他目录请输入自己的目录地址

```
命令提示符
Microsoft Windows [版本 10.0.19041.804]
(c) 2020 Microsoft Corporation. 保留所有权利。

C:\Users\MAKERW>cd c:/
c:\>cd edatools/chardet
c:\edatools\chardet>
```

输入 `python setup.py install`

```
命令提示符
Microsoft Windows [版本 10.0.19041.804]
(c) 2020 Microsoft Corporation. 保留所有权利。

C:\Users\MAKERW>cd c:/
c:\>cd edatools/chardet
c:\edatools\chardet>python setup.py install
running install
running bdist_egg
running egg_info
writing chardet.egg-info\PKG-INFO
writing dependency_links to chardet.egg-info\dependency_links.txt
writing entry points to chardet.egg-info\entry_points.txt
writing top-level names to chardet.egg-info\top_level.txt
reading manifest file 'chardet.egg-info\SOURCES.txt'
reading manifest template 'MANIFEST.in'
warning: no files found matching 'requirements.txt'
warning: no previously-included files matching '*.pyc' found anywhere in distribution
warning: no previously-included files matching '__pycache__' found anywhere in distribution
writing manifest file 'chardet.egg-info\SOURCES.txt'
installing library code to build\bdist.win-amd64\egg
running install_lib
running build_py
creating build
creating build\lib
creating build\lib\chardet
copying chardet\big5freq.py -> build\lib\chardet
copying chardet\big5prober.py -> build\lib\chardet
copying chardet\chardistribution.py -> build\lib\chardet
```



```

C:\命令提示符
byte-compiling build\bdist.win-amd64\egg\chardet\sbccharsetprober.py to sbcharsetprober.cpython-39.pyc
byte-compiling build\bdist.win-amd64\egg\chardet\sbcsgroupprober.py to sbcsgruopprober.cpython-39.pyc
byte-compiling build\bdist.win-amd64\egg\chardet\sjisprober.py to sjisprober.cpython-39.pyc
byte-compiling build\bdist.win-amd64\egg\chardet\universaldetector.py to universaldetector.cpython-39.pyc
byte-compiling build\bdist.win-amd64\egg\chardet\utf8prober.py to utf8prober.cpython-39.pyc
byte-compiling build\bdist.win-amd64\egg\chardet\version.py to version.cpython-39.pyc
byte-compiling build\bdist.win-amd64\egg\chardet\__init__.py to __init__.cpython-39.pyc
creating build\bdist.win-amd64\egg\EGG-INFO
copying chardet.egg-info\PKG-INFO -> build\bdist.win-amd64\egg\EGG-INFO
copying chardet.egg-info\SOURCES.txt -> build\bdist.win-amd64\egg\EGG-INFO
copying chardet.egg-info\dependency_links.txt -> build\bdist.win-amd64\egg\EGG-INFO
copying chardet.egg-info\entry_points.txt -> build\bdist.win-amd64\egg\EGG-INFO
copying chardet.egg-info\top_level.txt -> build\bdist.win-amd64\egg\EGG-INFO
zip_safe flag not set; analyzing archive contents...
creating dist
creating 'dist\chardet-4.0.0-py3.9.egg' and adding 'build\bdist.win-amd64\egg' to it
removing 'build\bdist.win-amd64\egg' (and everything under it)
Processing chardet-4.0.0-py3.9.egg
Copying chardet-4.0.0-py3.9.egg to c:\users\makerw\appdata\local\programs\python\python39\lib\site-packages
Adding chardet 4.0.0 to easy-install.pth file
Installing chardetdetect-script.py script to C:\Users\MAKERW\AppData\Local\Programs\Python\Python39\Scripts
Installing chardet.exe script to C:\Users\MAKERW\AppData\Local\Programs\Python\Python39\Scripts

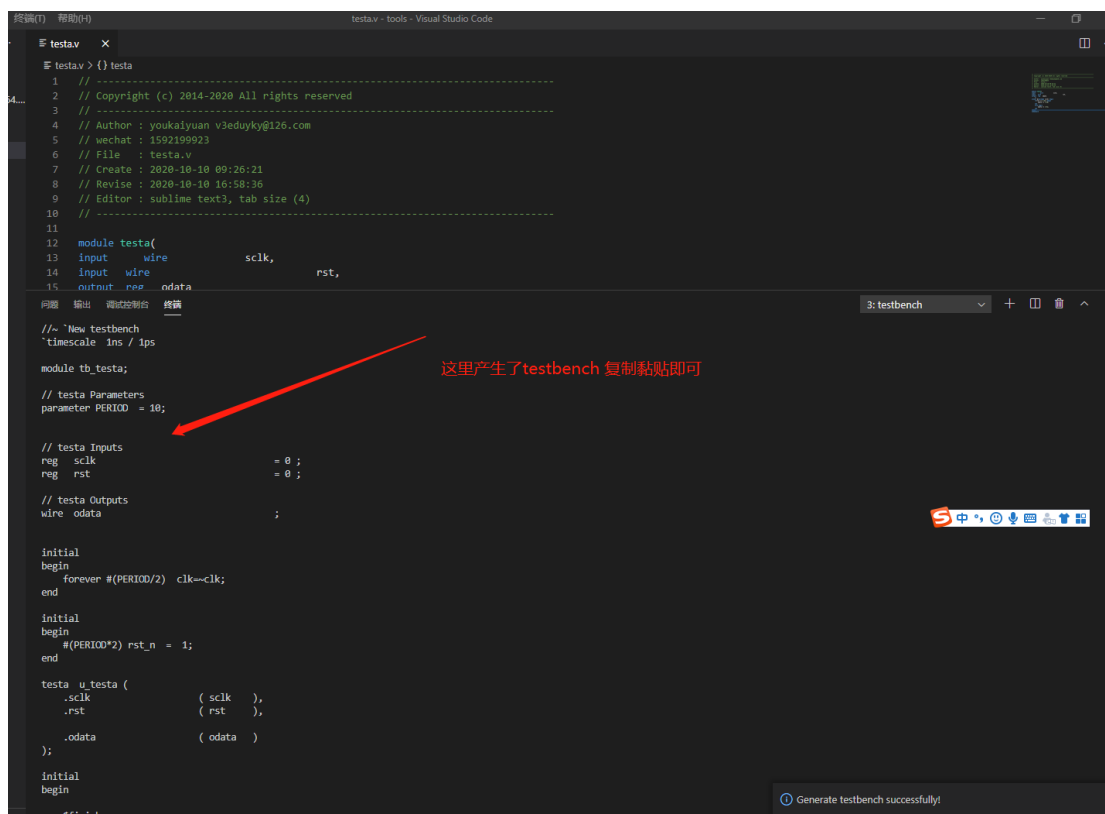
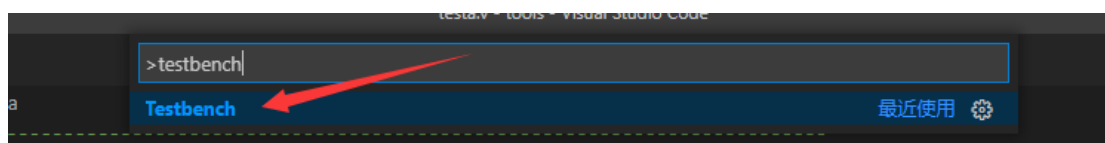
Installed c:\users\makerw\appdata\local\programs\python\python39\lib\site-packages\chardet-4.0.0-py3.9.egg
Processing dependencies for chardet==4.0.0
Finished processing dependencies for chardet==4.0.0

c:\edatools\chardet>

```

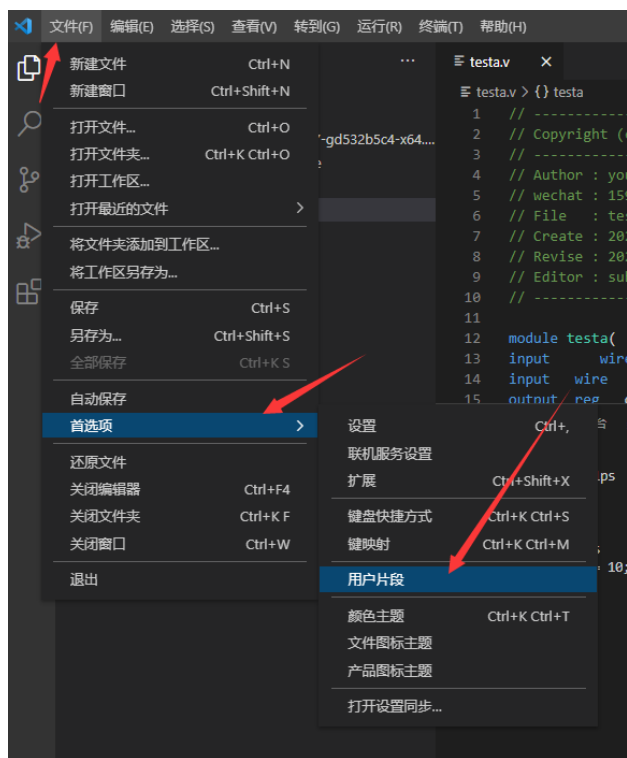
回到 vscode 编辑器界面，此时一定要打开 testa.v 当然可以是其他.v 我们验证生成 testbench 工程

Ctrl+shift+p 调出命令窗口输入 testbench

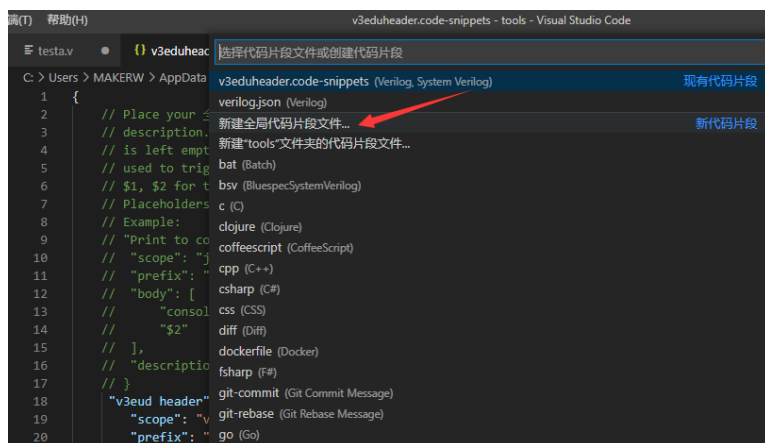


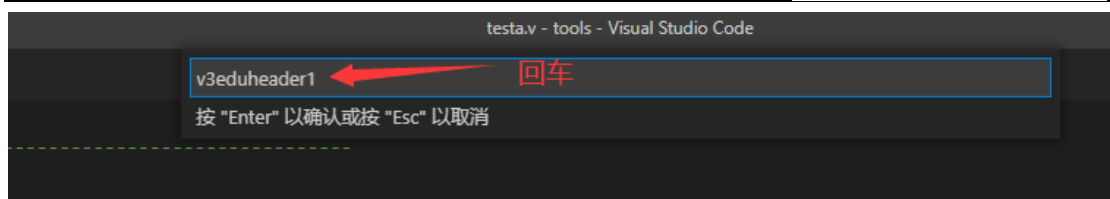
3.4 使用代码片段生成文件头

添加一个代码片段，代码片段可以选择支持的语言什么的。
scope 字段后面是支持的语法，没有这个字段，那就是使用 vscode 的时候，所有文件都可以使用这个代码片段。
prefix 字段是代码片段的快捷方式，输入对应的代码，就可以添加代码片段了。
body 里面就是代码片段的具体内容了。代码片段的内容需要在 “ ” 两个引号里面，每一行都需要。



新建全局片段





黏贴到里边去

```
"v3eud header": {
  "scope": "verilog,systemverilog",
  "prefix": "v3hdl",
  "body": [
    "// -----"
    "// Copyright (c) 2014-2020 All rights reserved"
    "// -----"
    "// Author      : v3eduyky@126.com"
    "// File        : $TM_FILENAME_BASE"
    "//wechat       : 15921999232"
    "// Create      : $CURRENT_YEAR-$CURRENT_MONTH-$CURRENT_DATE"
    "// Revise       : $CURRENT_YEAR-"
    "// Editor       : Vscode, tab size (4)"
    "// Functions    : "
    "//              "
    "// -----"
  ],
  "description": " file header"
}
```




```

1 {
2   // Place your 全局 snippets here. Each snippet is defined under a snippet name and has a scope, prefix, body and
3   // description. Add comma separated ids of the languages where the snippet is applicable in the scope field. If scope
4   // is left empty or omitted, the snippet gets applied to all languages. The prefix is what is
5   // used to trigger the snippet and the body will be expanded and inserted. Possible variables are:
6   // $1, $2 for tab stops, $0 for the final cursor position, and ${1:label}, ${2:another} for placeholders.
7   // Placeholders with the same ids are connected.
8   // Example:
9   // "Print to console": {
10    // "scope": "javascript,typescript",
11    // "prefix": "log",
12    // "body": [
13    //   "console.log('$1');",
14    //   "$2"
15    // ],
16    // "description": "Log output to console"
17  // }
18  "v3eud header": {
19    "scope": "verilog,systemverilog",
20    "prefix": "v3hd1",
21    "body": [
22      "-----",
23      "速览问题 (Alt+F8) 没有可用的快速修复",
24      "rights_reserved",
25      "-----",
26      "/// Author : v3eduyky@126.com",
27      "/// File : $TM_FILENAME_BASE",
28      "/// wechat : 15921999232",
29      "/// Create : $CURRENT_YEAR-$CURRENT_MONTH-$CURRENT_DATE",
30      "/// Revise : $CURRENT_YEAR-$CURRENT_MONTH-$CURRENT_DATE",
31      "/// Editor : Vscode, tab size (4)",
32      "/// Functions : ",
33      "-----",
34    ],
35    "description": " file header"
36  }
37 }

```

打开 testa.v 输入 v3hd1

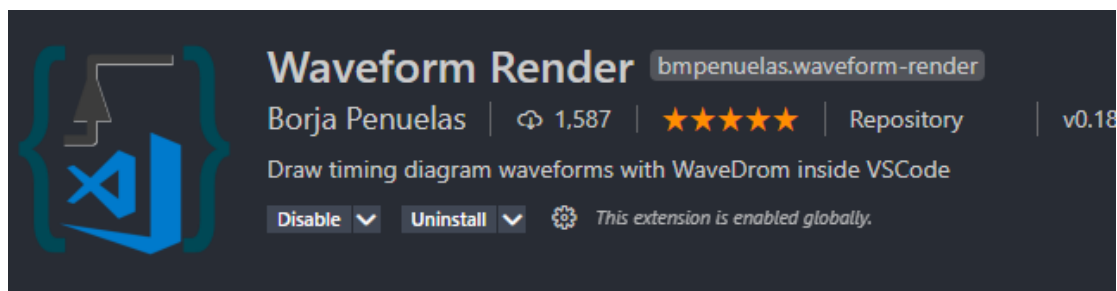
```

1 // -----
2 // Copyright (c) 2014-2020 All rights reserved
3 // -----
4 // Author : youkaiyuan v3eduyky@126.com
5 // wechat : 1592199923
6 // File : testa.v
7 // Create : 2020-10-10 09:26:21
8 // Revise : 2020-10-10 16:58:36
9 // Editor : sublime text3, tab size (4)
10 // -----
11 v3
12 mo v3hd1 v3eud header
13 input wire sclk,
14 input wire rst,
15 output reg odata
16 );
17 always @(posedge sclk) begin
18   if(rst == 1'b1) begin
19     odata <= 1'b0;
20   end
21   else begin
22     odata <= 1'b1;
23   end
24 end
25 endmodule
26
27

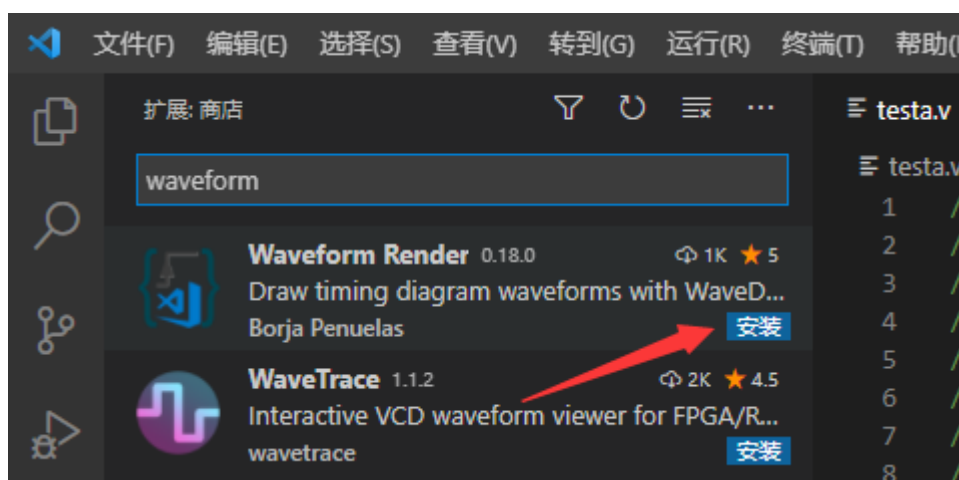
```



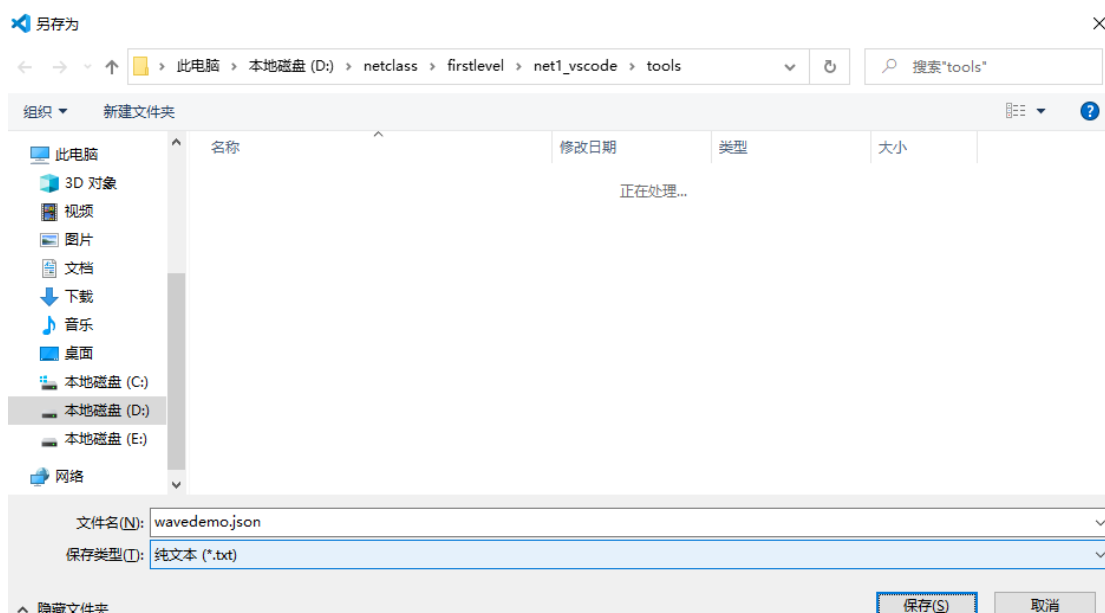
3.5 绘制波形插件



在插件中心搜索



使用这个绘图软件需要创建一个 json 脚本文件，新建文件保存为文件后缀为.json 即可



输入如下测试代码

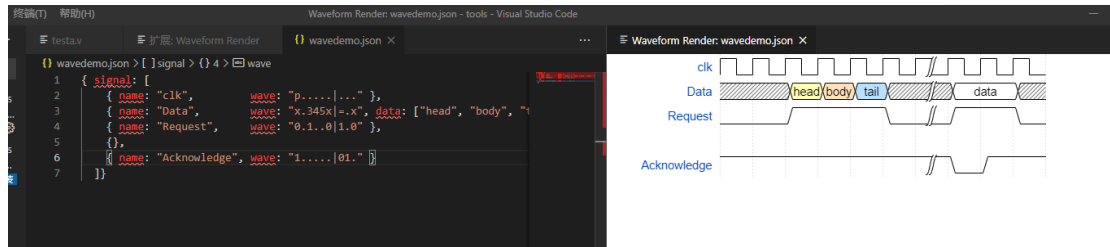
```
{ signal: [
  { name: "clk",          wave: "p.....|..." },
  { name: "Data",        wave: "x.345x|=.x", data: ["head", "body",
"tail", "data"] },
]
```



```
{ name: "Request",      wave: "0.1..0|1.0" },
{}},
{ name: "Acknowledge", wave: "1.....|01." }
}]}
```

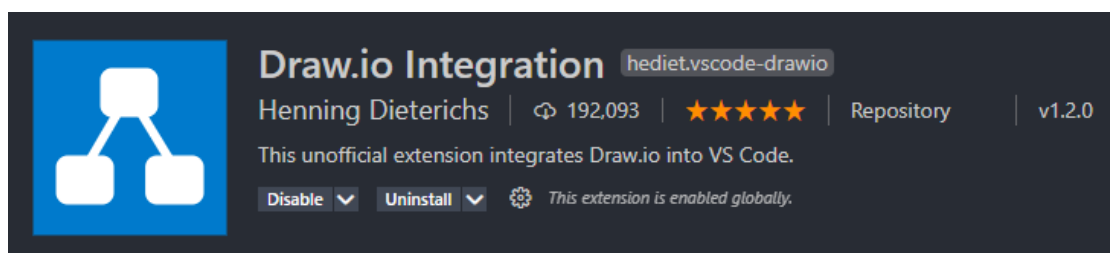
保存执行并按如下快捷键 **ctrl+k** 紧接着按 **ctrl+d** 左侧会显示波形

还可以打开实时预览波形，更改波形代码实时预览 **ctrl+k** 紧接着按 **ctrl+l**

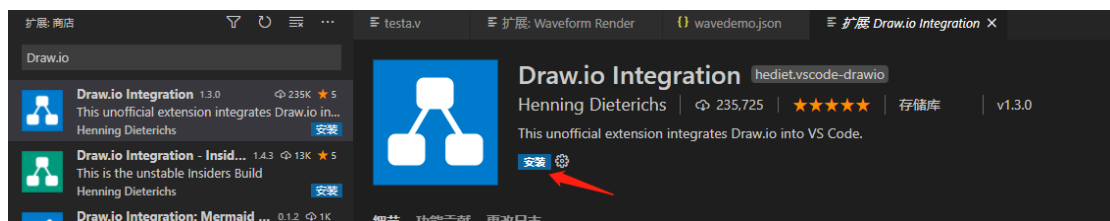


点击波形图片可以右键保存

3.6 绘图组件



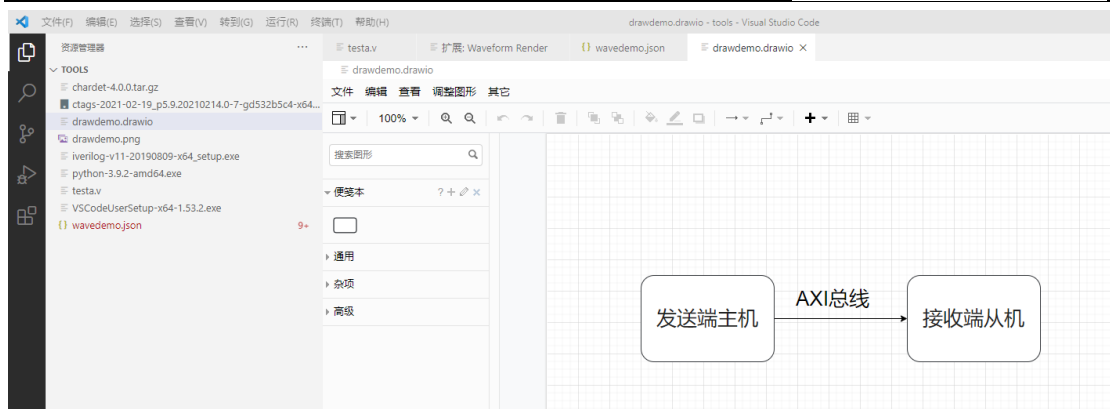
在插件中心搜索 Draw.io 插件



此软件可以替代 visio 绘制模块流程图

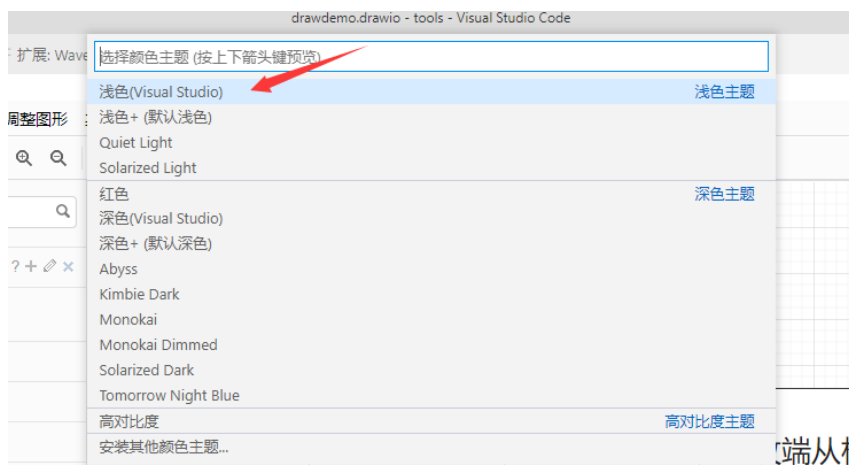
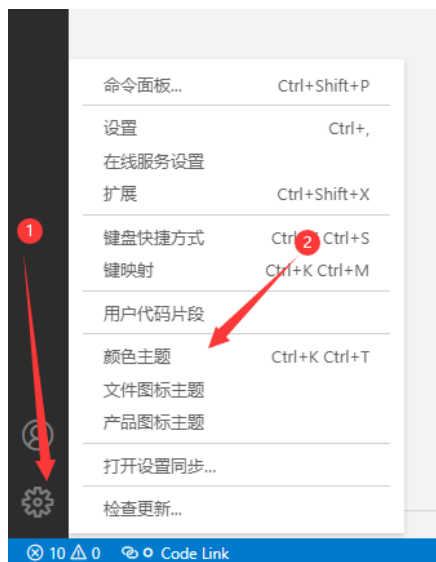
使用方法创建一个.drawio 后缀的文件打开就可以创建





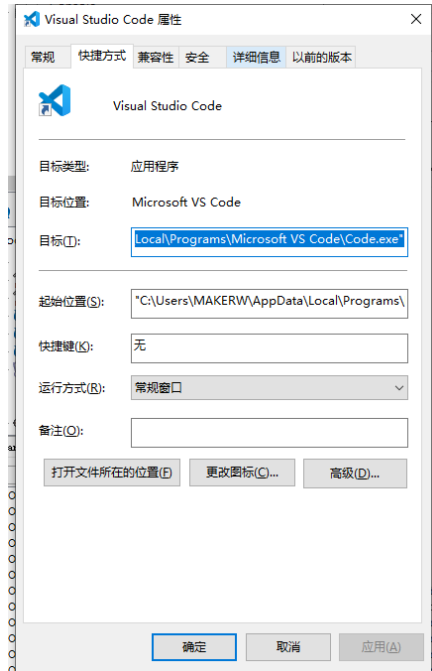
在文件中保存为 xxxx.png 即可导出图片。
我习惯使用浅色主题

3.7 颜色主题更改

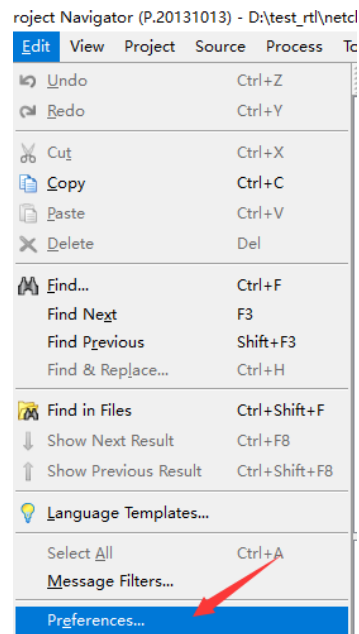


4. 关联 ISE 软件

打开 vscode 的快捷方式属性面板
复制目标中的路径

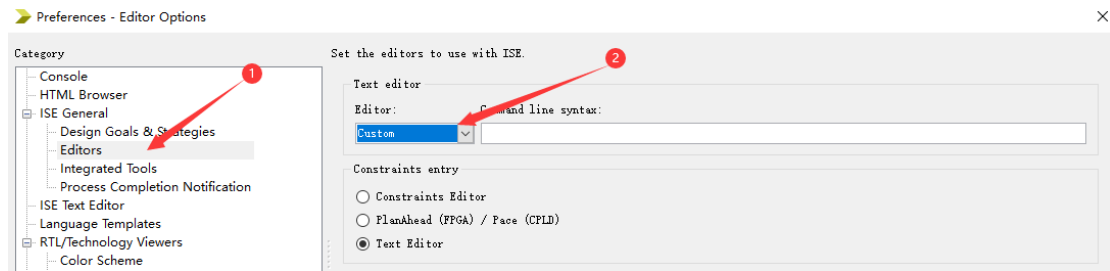


打开 ISE 软件，edit->preferences



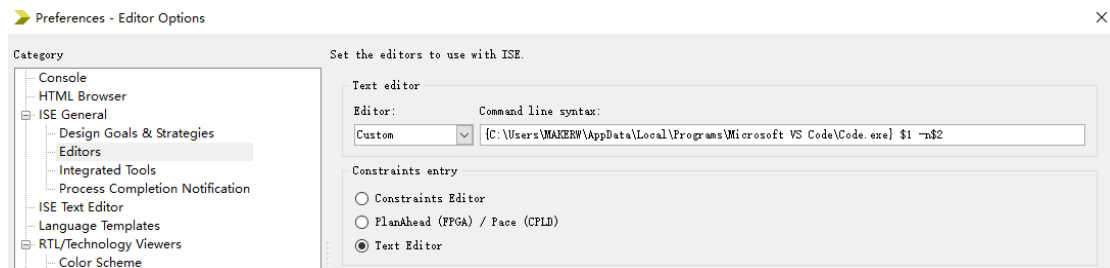
选择自定义的编辑器



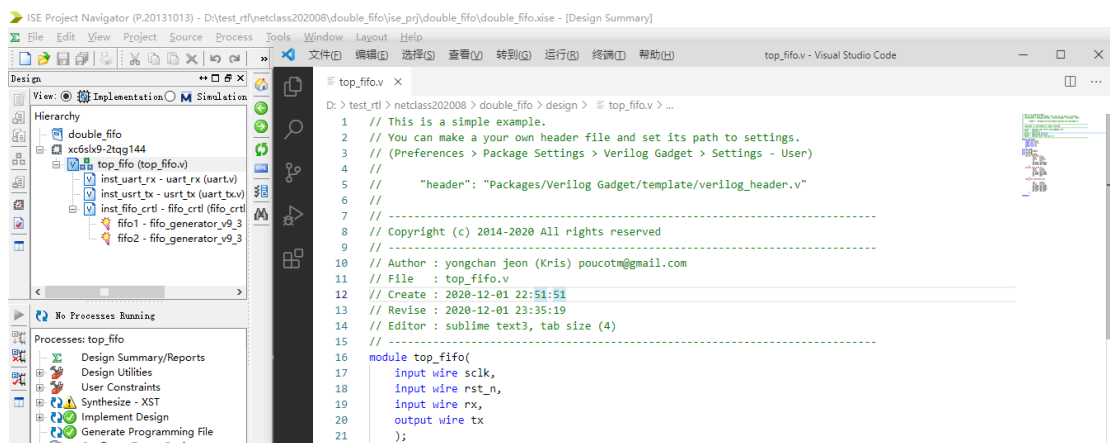


大括号中是 `vscode` 的启动文件目录替换为之前你复制的目录，大括号之外的 `$1 -n$2` 也必须复制进去

{C:\Users\MAKERW\AppData\Local\Programs\Microsoft VS Code\Code.exe} \$1 -n\$2



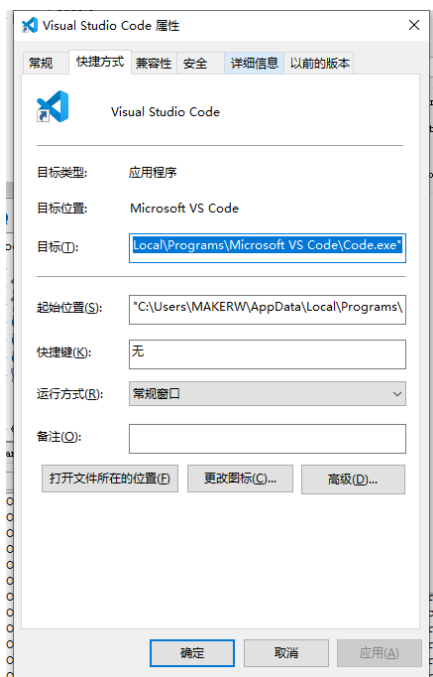
ISE 中双击.v 文件即可完成启动 `vscode` 编辑



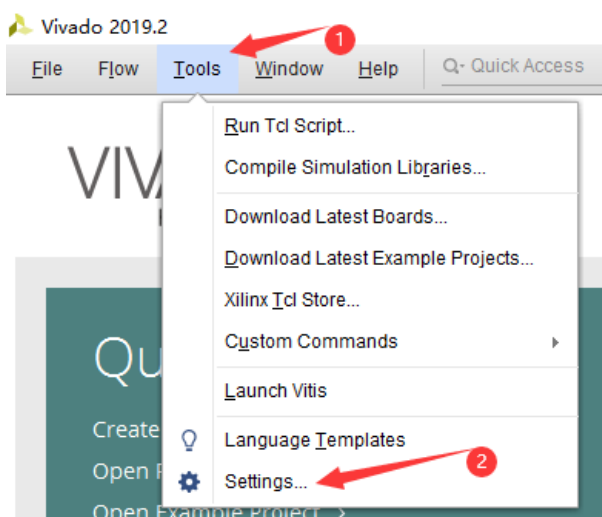
5. 关联 vivado 软件

打开 `vscode` 的快捷方式属性面板
复制目标中的路径





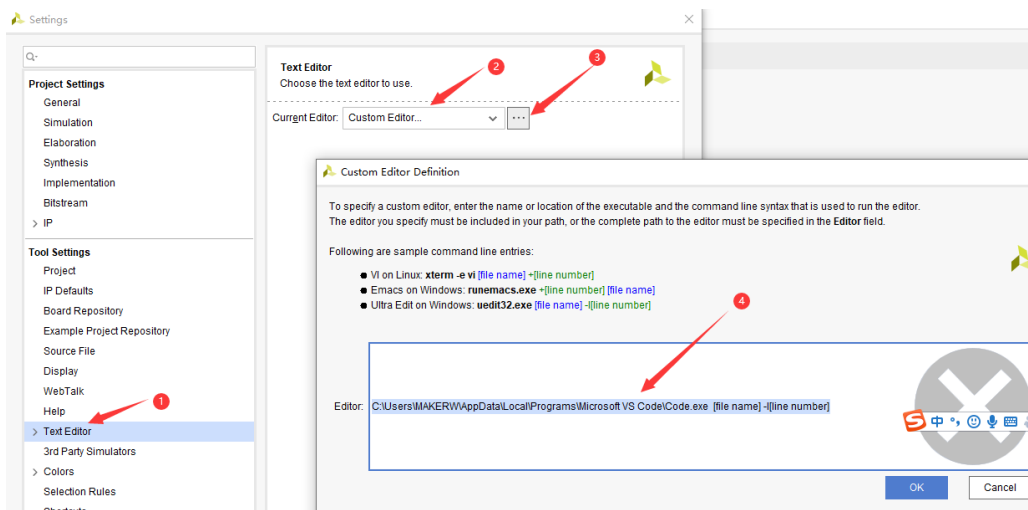
打开 vivado 工具



注意最后的[file name] -l[line number] 一定要加不然无法调用 vscode

C:\Users\MAKERW\AppData\Local\Programs\Microsoft VS Code\Code.exe [file name] -l[line number]





双击 vivado 中的 demo.v 在 vscode 中打开

