

# 使用手册

最后修改时间 2019-05-23

# 目录

<b>1 快速开始</b>	<b>6</b>
1.1 下载、安装	6
1.2 使用IDE	6
1.2.1 IDE的主要界面元素	6
1.2.2 下载pwm demo项目，编译、烧写	8
1.2.3 创建新的可执行项目	9
1.2.4 创建新的库项目	9
1.2.5 使用包管理器	9
1.2.6 编写代码时常见的操作	10
1.2.7 常用选项设置	10
1.3 注意事项	10
<b>2 维护操作</b>	<b>11</b>
2.1 下载、安装	11
2.1.1 下载正确的版本	11
2.1.2 解压缩与首次启动	11
2.1.3 添加开始菜单快捷方式	13
2.1.4 Mac OS允许未认证的开发者	14
2.1.5 脱机使用	15
2.2 备份	15
2.3 卸载	15
2.4 错误报告	15
<b>3 主界面</b>	<b>16</b>
3.1 串口监视器	16
3.2 工具、状态栏	16
3.3 文件管理	17
3.4 问题	17
3.5 日志	17
3.6 命令面板	17
<b>4 特色页面</b>	<b>18</b>
4.1 IDE设置页面	18
4.1.1 操作说明	18

4.2 项目配置	20
4.2.1 打开项目配置	20
4.2.2 配置界面	20
4.2.3 部分配置选项介绍	20
4.3 包管理器	22
4.4 flash编辑工具 (kflash)	23
4.4.1 配置界面	23
4.4.2 引用文件	24
<b>5 特色功能</b>	<b>25</b>
5.1 工作区	25
5.1.1 编译	25
<b>6 生成 (编译) 系统</b>	<b>26</b>
6.1 CMake	26
6.2 CMakeLists.txt生成	26
<b>7 调试功能简介</b>	<b>28</b>
7.1 说明：	28
7.2 调试界面	28
7.3 常见操作：	29
7.3.1 重置软、硬件状态	29
7.3.2 启动调试	29
7.3.3 查看调试信息	29
7.3.4 查看变量值	30
7.4 在Linux和Mac中安装系统依赖	30
7.5 在Windows上安装JLink驱动	30
<b>8 常用操作</b>	<b>31</b>
8.1 一、界面、操作、交互	31
8.1.1 打开/隐藏底部面板	31
8.1.2 打开系统终端软件	31
8.1.3 快速执行命令	31
8.2 二、开发、编辑	31
8.3 三、编译、调试、运行	31
8.4 四、维护	32

8.4.1 备份IDE设置	32
8.4.2 查看日志	32
<b>9 常见问题 (FAQ)</b>	<b>33</b>
9.1 常见问题	33



# 1 快速开始

本教程的目的是让任何人快速上手使用IDE， 默认需要有嵌入式开发的经验。如果看完有不清楚的地方， 欢迎发送反馈。

## 1.1 下载、安装

Beta：<http://kendryte-ide.s3-website.cn-northwest-1.amazonaws.com.cn/>

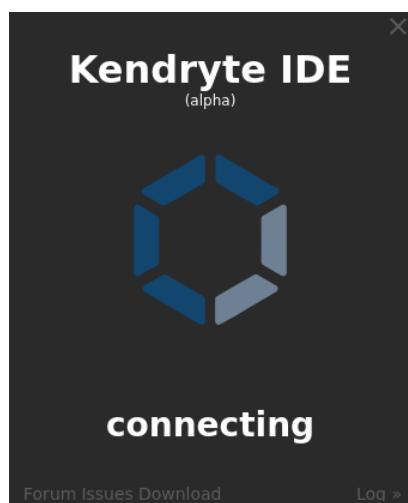
Alpha：<http://kendryte-ide.s3-website.cn-northwest-1.amazonaws.com.cn/alpha.html>

下载后使用解压缩软件解压， 放到任意目录中。

**Mac 下请勿使用 iZip 解压压缩包， 可能会无法打开程序。建议使用 Keka 来解压软件包。**

打开解压出的KendryteIDE文件夹， 双击“KendryteIDE”运行。 (MAC还需要点击允许未知开发者， 见另外章节)

此时会弹出更新器窗口：



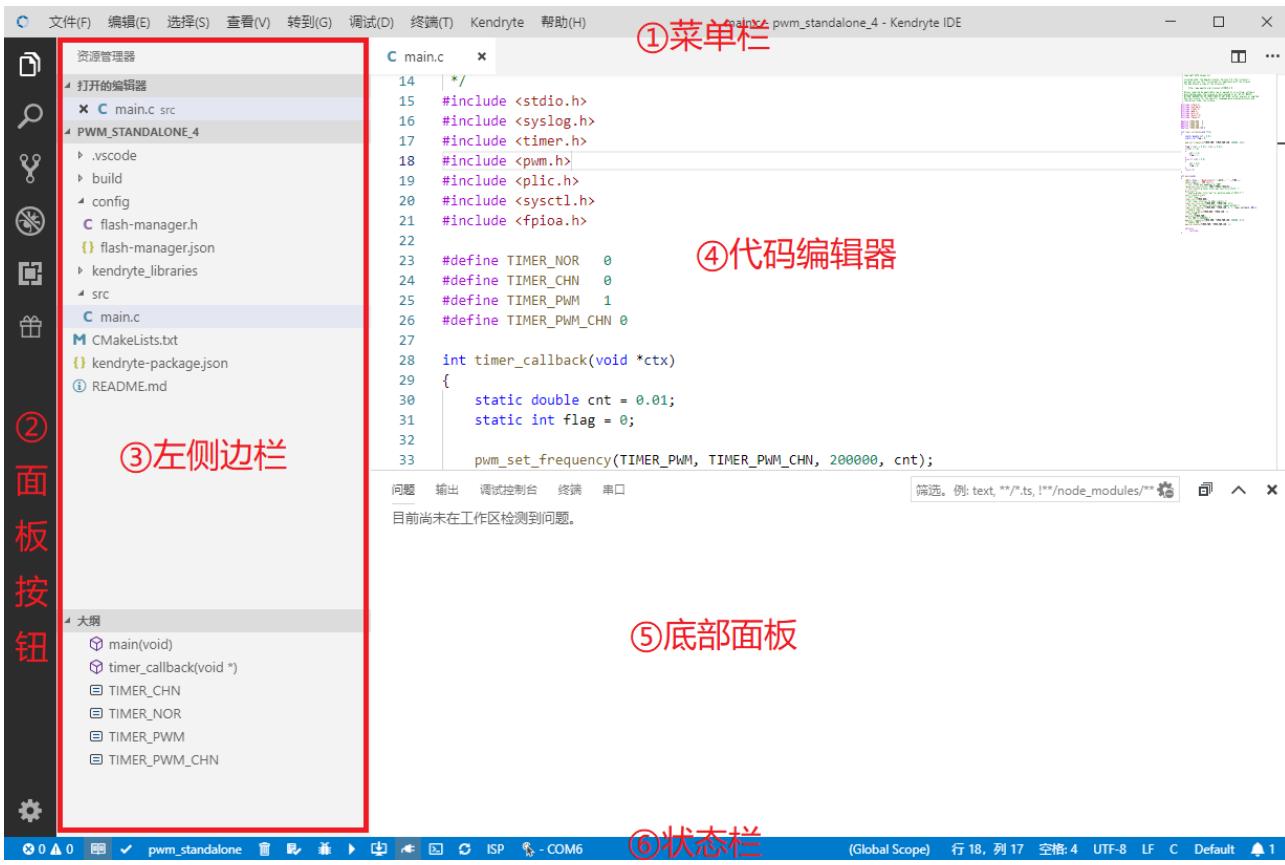
等待更新完成， IDE将启动（见下图）。

启动后，界面右下角会弹出插件下载、安装进度条，需要等待它们结束。这个过程中IDE将重启1~2次。

## 1.2 使用IDE

### 1.2.1 IDE的主要界面元素

IDE各个界面、面板的详细介绍请查看后续章节

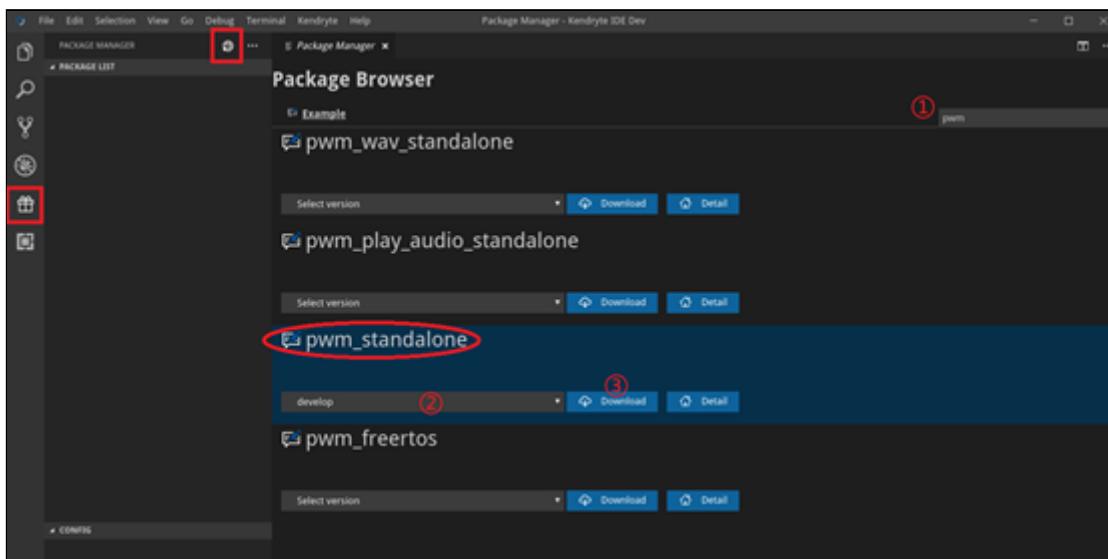


(IDE界面截图)

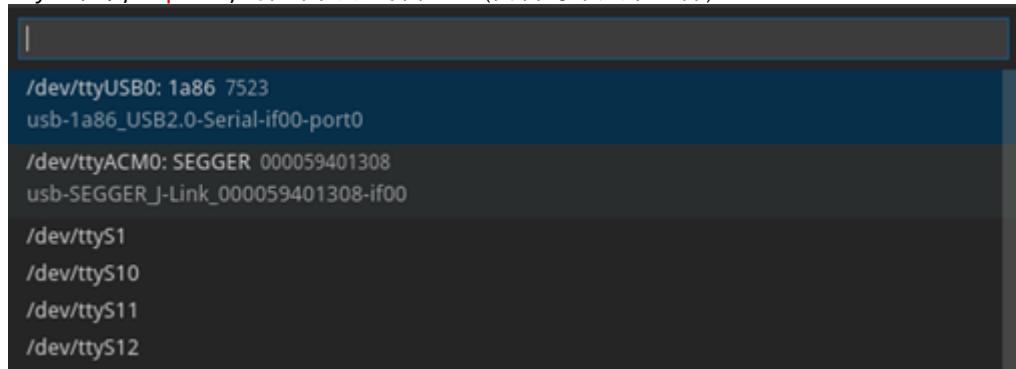
其中：

- ① 菜单栏
- ② 左侧面板按钮，点击切换不同面板。包括：文件、搜索、版本控制、调试、包管理器、扩展
- ③ 左侧边栏（此处展示的是文件管理器）
- ④ 文件编辑器
- ⑤ 底部面板（此处是“问题”选项卡）
- ⑥ 状态栏，其中包括几个重要的快捷按钮

### 1.2.2 下载pwm demo项目，编译、烧写



1. 点击IDE左侧按钮即可打开包管理器。  
此处显示当前安装的包（由于还没有开始，这里是空的）
2. 点击上方的浏览按钮，打开浏览器。
3. 在①处搜索框输入“pwm”
4. 从列表中找到pwm\_standalone
5. 在②处选择develop
6. 点击③按钮下载
7. 弹出目标选择窗口，选择任意一个路径（但路径中不要出现空格），demo项目将被解压到此处。  
例如demo名为test，选择的目录是D:\code，则最终目录是D:\code\test。如果这个目录已经存在，则为D:\code\test\_2，依此类推。
8. IDE会自动打开这个新项目
9. 点击Kendryte菜单，**Install Dependency**，等待提示成功（通常只需要几秒钟）
10. 点击Kendryte菜单，**Build**，项目就会开始编译，底部出现进度条
11. 等待编译完成，进度条消失
12. 点击Kendryte菜单，**Upload**，弹出串口选择框：（内容每个人不一样）



13. 点击连接了开发板的串口设备（如上图中的第一个），烧写将会开始
14. 等待右下角的滚动条消失，程序烧写完毕
15. 按下开发板上的reset按钮，可以看到led呼吸灯效果

### 1.2.3 创建新的可执行项目

1. 点击文件 → 打开文件夹
2. 在弹出的窗口中选择或者新建一个空白文件夹，路径中不要使用空格
3. 点击窗口左下角的“叹号”图标
4. 文件管理器中打开src/main.c，编写你的代码

**创建新的源文件：** (假设要创建src/my.c)

1. 点击文件管理器，使其获得焦点（周围出现蓝边）
2. 然后点击新文件所在的目录“src”，使其所在行高亮
3. 点击顶部的加号按钮，目录中出现一个空白的文本框
4. 输入文件名“my.c”
5. 双击打开kendryte-package.json文件，找到“Source files”
6. 在其文本框中另起一行，输入新建的文件相对路径，src/my.c

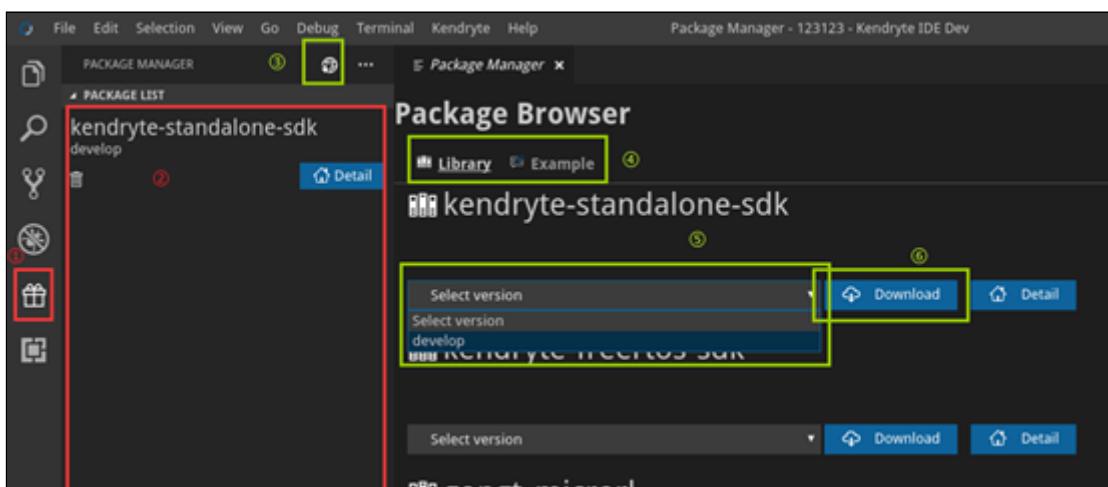
**创建新的头文件：** (假设要创建inc/my.h)

1. 点击文件管理器，使其获得焦点
2. 点击顶部的“新文件夹”按钮，命名为“lib”
3. 点击新创建的lib文件夹，点击顶部的加号按钮，命名为my.h
4. 双击打开kendryte-package.json文件，找到“Headers direcotry”
5. 在其文本框中另起一行，输入“lib”（注意与源文件不同，不要写my.h）
6. 在源文件中添加#include <my.h>

### 1.2.4 创建新的库项目

1. 与可执行项目相同，不过应该删除自动创建的main.c（因为不能有main函数）
2. 编写这个库的源文件、头文件（导出的头文件请单独放在一个文件夹中）
3. 双击打开kendryte-package.json文件，在Project type选择框中选中“Library”
4. 与可执行项目相同，填写Source files和Headers directory
5. 找到“Include root”，在其文本框中填写导出的头文件所在目录。这些头文件可以被库的用户#include。（Headers directory中的不可以）

### 1.2.5 使用包管理器



### (包管理器界面)

- ① 打开包管理器按钮
- ② 当前已下载的库的列表，点击垃圾桶图标可以删除这个库
- ③ 打开浏览器
- ④ 根据类型和名称筛选库
- ⑤ 选择要下载的版本
- ⑥ 下载并解压所选的版本

每个项目都有自己的依赖，不和其他项目互相影响，重新下载就可以更新。

当前项目的依赖被记录在kendryte-package.json中，通常随代码一起发布。

### 1.2.6 编写代码时常见的操作

- 在函数调用上按Ctrl+左键，可以跳转到声明
- 在头文件上按Ctrl+左键，可以跳转到这个文件
  - 如果有多个同名文件，将显示选择框，可在右侧选择实际想要打开的文件
- Ctrl+T可以全局搜索符号（例如函数和变量）
- 如果当前选择的行有错误，最左侧会显示灯泡按钮，点击可以获得修复选项（包括忽略错误）
- 在打开的文件名上按下鼠标中键可以关闭它
- 鼠标在变量上悬停几秒钟，会弹出提示（比如变量的类型）

### 1.2.7 常用选项设置

**语言设置**：在命令面板中，输入“language”，选择 Configure Display Language。

这会打开一个文件，内容类似：{"locale":"en"}，改成{"locale":"zh-cn"}，然后重启IDE即可切换为中文。

#### 程序设置：

要打开程序设置，在左下角齿轮图标菜单中，点击“设置”，或者在命令面板中输入open settings。

- Window: Title Bar Style – 这个选项通常无需修改。但在部分linux桌面上，如果拖动标题栏无法移动窗口，需要切换为native，然后重启IDE
- Workbench: Color Theme – 这个选项控制IDE整体颜色，选择“light”可以切换为浅色
- Workbench > List: Open Mode – 控制IDE中各种组件使用单击还是双击来触发操作（比如文件管理器如何打开文件），默认用双击。
- Editor: Font Size – 这个选项控制编辑器内字体大小
- Debug > Openocd: Core – 指定openocd调试哪个核心，重启openocd后生效
- Debug > JLink: Speed – 指定JLink速度。过高的速度可能导致传输错误（openocd报错、无法调试等），过低影响加载速度。
- Debug > Custom – 指定一个cfg文件的路径，启动openocd时将跳过IDE内置配置文件。

## 1.3 注意事项

1. 目录中不要用空格、中文等，容易导致各种问题。特别要注意的是Program Files目录，它本身自带空格，所以不要使用。
2. IDE是绿色软件，最好不要放在系统文件附近（例如C盘、/usr、/Application）
3. Windows上输入路径时，需要用反斜线“/”，而不是Windows默认的“\”
4. 调试前建议烧入空白程序，或者进入ISP模式再调试。以防止老程序的初始化操作影响新程序的结果

## 2 维护操作

### 2.1 下载、安装

#### 2.1.1 下载正确的版本

IDE有2个版本：Alpha和Beta。支持3种操作系统：Windows、Linux、Mac，均只支持64位版本。为避免问题，请使用最新版本的操作系统。

下载地址为：

Beta：<http://kendryte-ide.s3-website.cn-northwest-1.amazonaws.com.cn/>

Alpha：<http://kendryte-ide.s3-website.cn-northwest-1.amazonaws.com.cn/alpha.html>

其中Alpha版主要用于测试，不推荐一般使用。如果你使用其他版本遇到了问题并报告给我们，我们可能立刻修复并更新Alpha版，此时你可以暂时使用Alpha版。

通常打开网页后，最大的按钮就是当前操作系统的下载链接。如果要给其他系统下载，需要翻到下方进行选择。

离线依赖包：绝大多数情况下，你完全不需要它，如果你没有充足的理由，请不要下载。

#### 2.1.2 解压缩与首次启动

下载的是一个7zip压缩包，几乎全部解压缩软件都支持这种格式。

- Windows：使用WinRAR、7zip（下载：<https://www.7-zip.org/download.html>）等解压缩软件。
- Linux：桌面环境双击即可。命令行需要p7zip软件包，在解压目标文件夹里运行命令：7za x /path/to/Kendryte-ide.7z
- Mac：在AppStore搜索7zip然后选择任意一个软件安装，然后就可以双击解压缩。

**为防止出现各种问题，解压缩目标应该不包含空格，例如：D:\kendryte。**

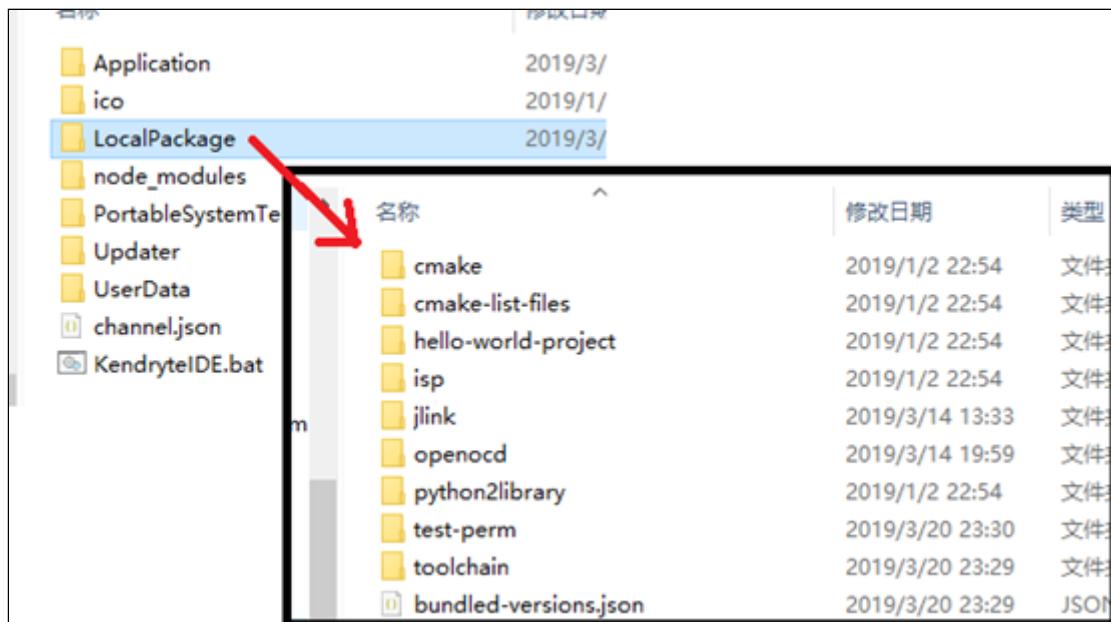
如果你同时使用Alpha、Beta版，它们不能解压到同一目录中，否则会产生冲突。

解压后，打开KendryteIDE文件夹：

- Windows：直接双击Kendryte文件（或Kendryte.bat）
- Linux：通常直接双击Kendryte.sh文件即可，有些桌面环境可能需要打开终端运行它
- Mac：双击Kendryte.command。如果提示禁止运行未知应用，请按照弹出的提示，去系统设置中添加例外条件（操作方法见下方章节）。

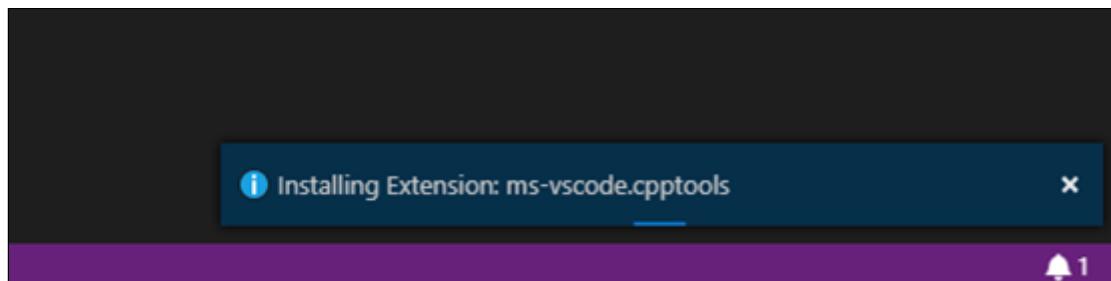
首次启动时，IDE会自行更新，此过程需要下载数百MB的软件包，请耐心等待。

如果下载速度太慢（比如迅雷可以达到10MB/s，此处却只有几百KB），可以下载离线依赖包，解压到相同目录中。解压后，IDE目录中应该有LocalPackage文件夹，如图所示。



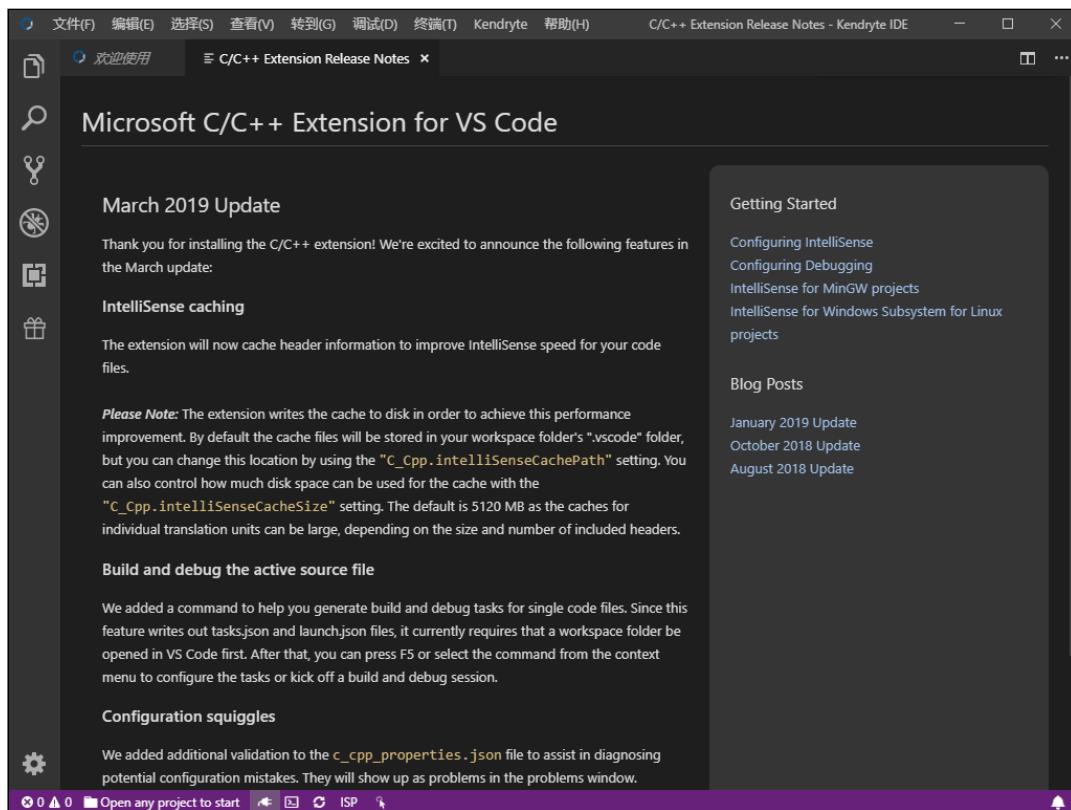
(此为正确安装后应有的目录结构)

更新完毕后IDE窗口会打开，此时安装并未彻底完成，请等待右下角的多个滚动条全部到100%。这个过程中IDE会自己重启1~2次。



(右下角的滚动条)

所有滚动条到100%后相关提示框会消失，等它们全部消失后，安装就结束了。



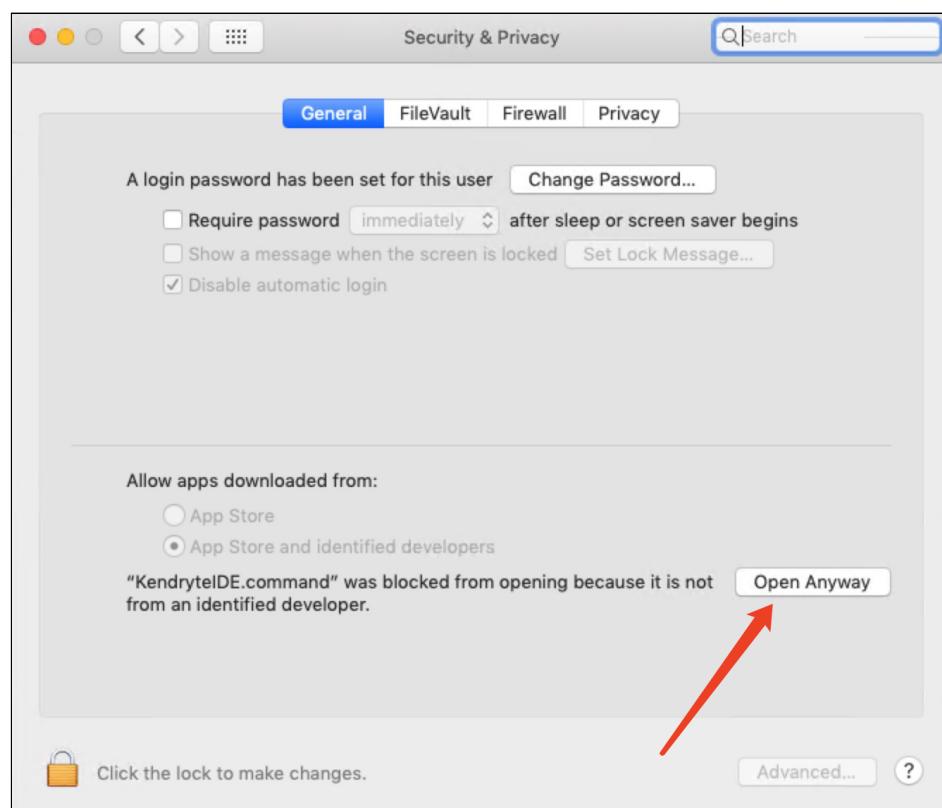
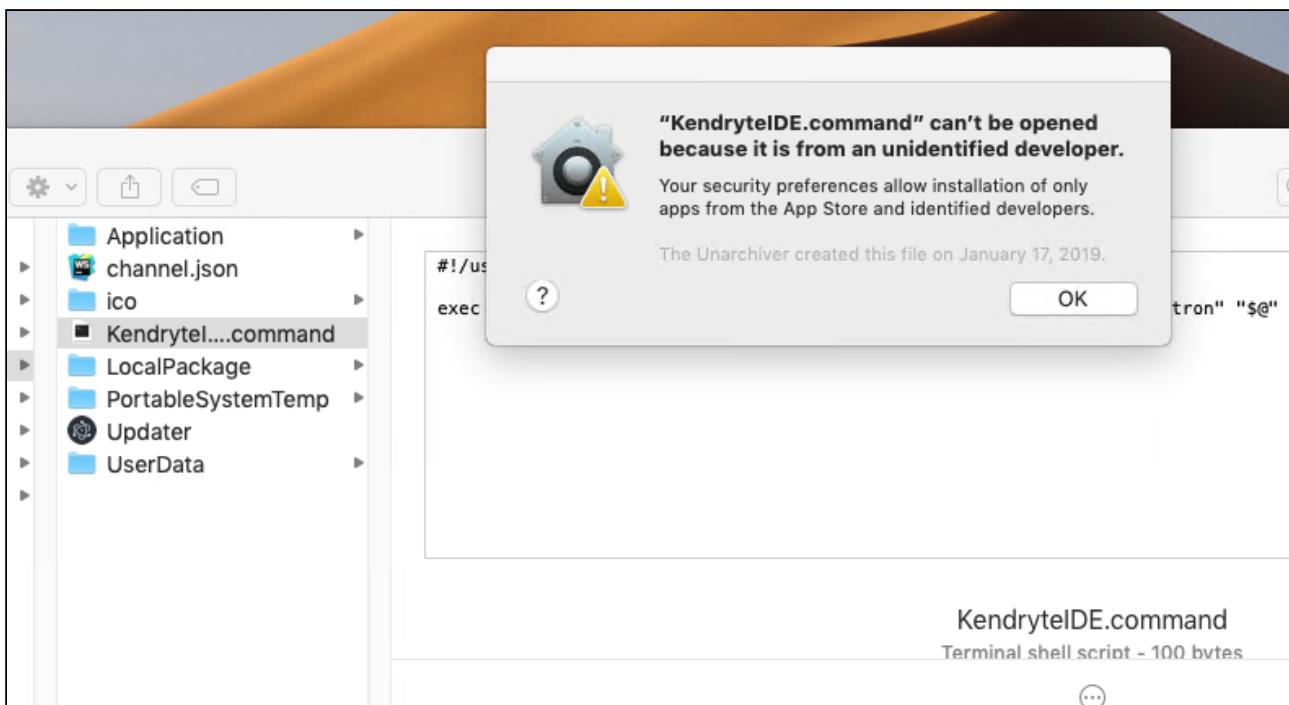
(完成安装)

安装完成后，显示欢迎页面（可能还有C++插件的更新日志，如上图），可以随时关闭。

### 2.1.3 添加开始菜单快捷方式

首次启动时，会提示是否创建快捷方式。如果需要手动创建，可以点击“Kendryte 菜单 > IDE调试工具 > 创建快捷方式”按钮。

## 2.1.4 Mac OS允许未认证的开发者



## 2.1.5 脱机使用

IDE支持离线使用。启动时跳过更新，但会导致部分功能不可用。具体过程为：

1. 首次启动时必须联网下载组件，此时无法离线
2. 当全部安装完毕后，复制（或压缩）KendryteIDE目录到需要使用的机器上（操作系统必须相同）
3. 双击运行：
  - a. Windows : KendryteIDE/Application/xxxxxxxx/Kendryte IDE.exe
  - b. Linux : KendryteIDE/Application/xxxxxxxx/kendryte-ide
  - c. Mac : KendryteIDE/Application/xxxxxxxx

## 2.2 备份

通常情况下，需要备份的只有设置、快捷键。

在界面左下角齿轮处找到这两项，分别点击页面右上角的  按钮，然后将它的内容复制到安全位置。还原时覆盖它并重新启动IDE即可。

IDE使用过程中的所有数据都在安装目录中“UserData”内。

## 2.3 卸载

IDE是“绿色软件”，直接删除KendryteIDE文件夹即可，其中UserData\latest\user-data\User目录中包含配置文件。

不过，如果你创建过快捷方式，需要手动删除它：

- Windows : 在开始菜单找到“Kendryte IDE”图标，右键点击“打开文件位置”，然后删除它。
- Linux : 删除\$HOME/.local/share/applications/kendryte-ide.desktop
- Mac : 打开/Applications, 找到Kendryte IDE图标，删除它

IDE还会使用系统临时文件夹，这些文件会被系统自动清理，无需专门删除。

## 2.4 错误报告

## 3 主界面

KendryteIDE修改自微软的VS Code编辑器，它的全部功能均可使用，文档在：<https://code.visualstudio.com/docs>

### 3.1 串口监视器

### 3.2 工具、状态栏



(状态栏)

根据项目状态不同，状态栏可能有非常大的不同，此处介绍标准状态下存在的各个按钮

浅色的图标为分隔符，其余按钮分别为：

① 状态提示：

1. 打开错误与警告面板按钮
2. 其他提示信息（如调试进程）

② 快速工具左半边：

1. 项目状态提示（如果有错误，显示为叹号）
2. 选择项目（只有打开多个项目时才显示）
3. 清理（删除）编译结果，下次将重新编译
4. 运行编译
5. 运行调试
6. 运行程序，但不调试
7. 刷写

③ 快速工具右半边：

1. 打开串口监视器
2. 重启开发板
3. 重启开发板到ISP模式
4. 选择串口（选择的串口影响刷写、重启功能，不影响调试、运行）

④ 最右侧按钮：

1. 打开消息中心按钮

### 3.3 文件管理

### 3.4 问题

### 3.5 日志

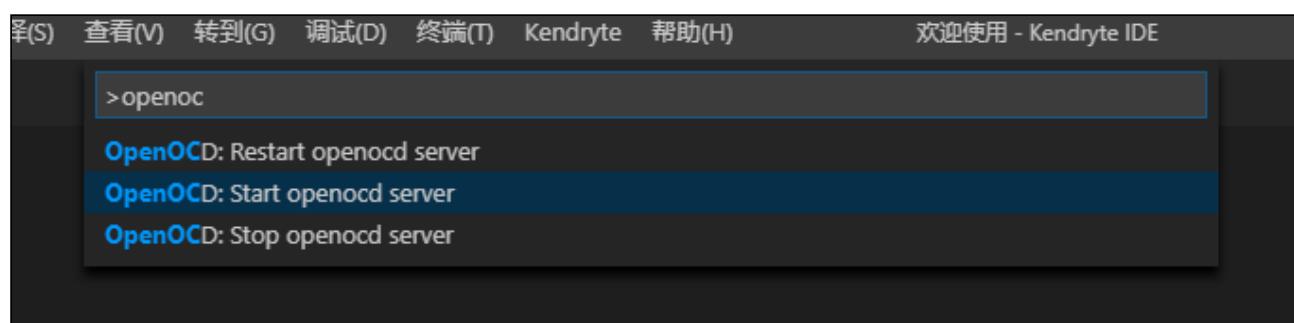
### 3.6 命令面板

可以点击左下角齿轮，在弹出菜单中选择第一个“命令面板... (Command Palette...)”按钮，或者按下快捷键：Ctrl+Shift+P。

界面中所有功能均可在这里执行。

在弹出的输入框中输入想要运行的功能（输入一部分即可），按回车即可执行对应的命令。

按下ESC可以取消。重新按下Ctrl+Shift+P可以清空输入，并重新开始搜索命令。



例如：想要运行“重启OpenOCD”，可以输入“OpenOCD”，然后在下面列表中选择“OpenOCD : Restart openocd server”

## 4 特色页面

### 4.1 IDE设置页面

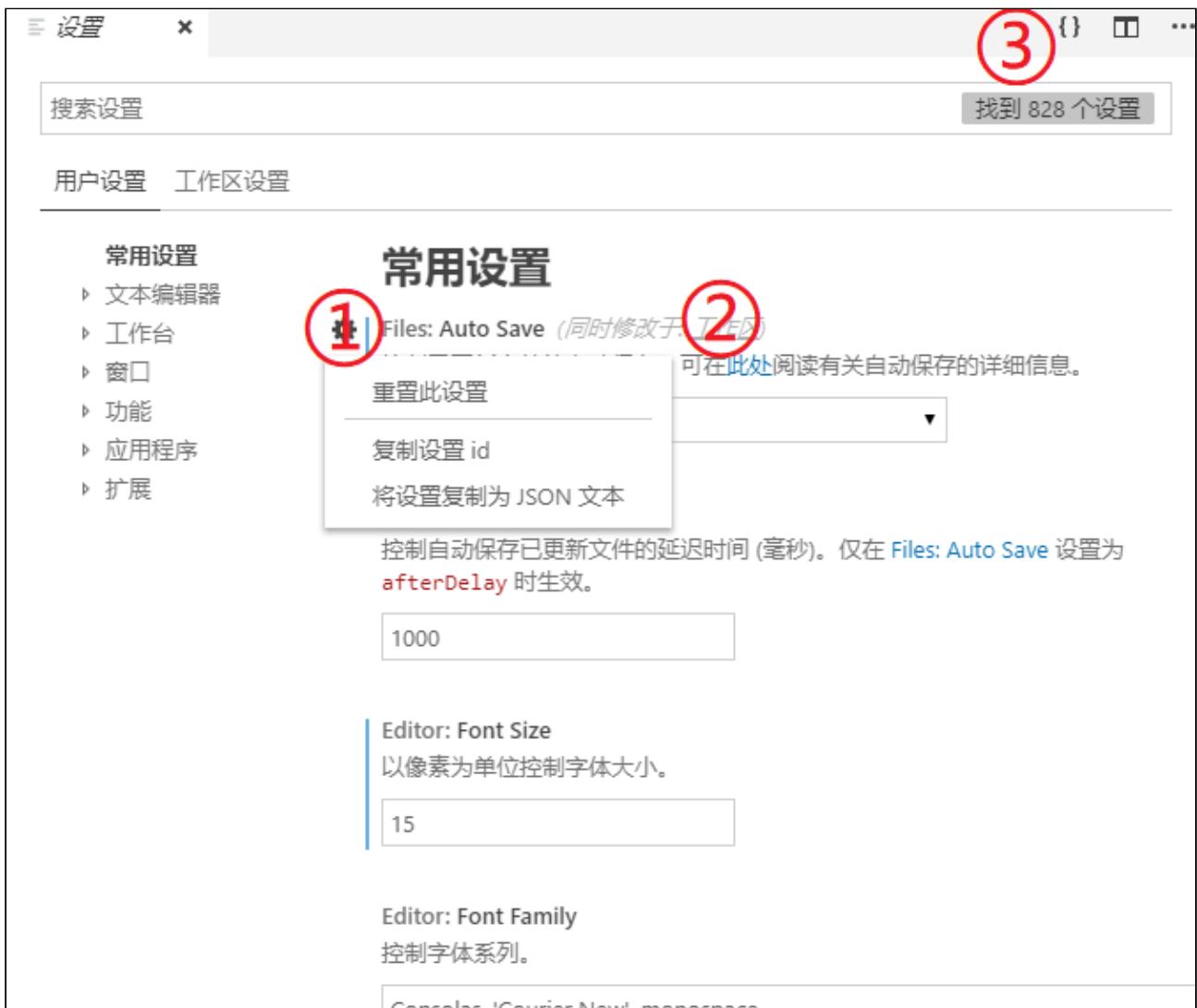
#### 4.1.1 操作说明

要打开IDE设置，点击左下角齿轮图标，选择第二项“设置”。

如果你打开了一个文件夹（见下），则在搜索框下方有两个选项卡“用户设置”和“工作区设置”。如果没有打开文件夹，则只有“用户设置”。

1. 用户设置：保存在IDE安装目录中，可以设置IDE的绝大部分功能
2. 工作区设置：只针对当前项目的配置，保存在.vscode文件夹中，可以和代码一起打包或提交，优先级高于用户设置。

提示：工作区设置主要用于同步代码格式（如：缩进方式）等，一般不应该将个性化设置保存在代码中（如：编辑器外观、字体）



要修改设置，直接在下拉框中选择或在文本框输入即可。所有设置立即保存，大部分立即生效（部分无法立即生效的，修改后会提示）。

如果不慎改错，想要还原默认值，可以点击每个设置项目前面的齿轮图标（图中①），选择“重置此设置”。已修改的设置，其左侧有蓝线作为提示。

修改“用户设置”时注意此设置是否在当前项目中指定过（图中②）。

对于高级用户，可以使用json编辑器修改配置，点击图中③处按钮即可。

## 4.2 项目配置

### 4.2.1 打开项目配置

两种方法：

1. 点击菜单“Kendryte > Project settings”
2. 找到项目根目录的kendryte-package.json并双击

由于还没建立的项目没有kendryte-package.json，所以只能用方法1。

### 4.2.2 配置界面



(项目配置界面)

每个字段都有相应说明。文本框以红色显示时，说明内容有误。有些字段的错误不影响编译（例如版本号），而其他的可能直接导致编译失败（例如源码文件）。

### 4.2.3 部分配置选项介绍

**项目类型：有三种**

1. executable：该项目编译后可直接烧写到芯片中运行

2. library：不能直接运行，编译后输出的是.a文件，可以被其他项目使用
3. prebuilt library：与library相似，但不包含源代码，只包含.a文件

如果是library项目，或者一个公开发布的executable项目，项目名称建议使用类似 **用户名\_项目名** 的风格命名，以防止和其他人冲突。

**源码文件**：指定本项目需要编译哪些文件。

可以直接编辑内容，每行一个文件名（或一个匹配规则）。也可以使用文本框下方的快速添加按钮。

推荐直接填写文件名，这样可以控制编译顺序，最大程度的提升编译过程的确定性。

如果需要一次添加大量文件，可以点击文本框下方的“添加目录”和“目录树”按钮。

如果一定要使用匹配规则，可以使用两种匹配方式：

1. 匹配文件夹里的所有文件，不递归。例如：src/\*.cpp
2. 递归匹配整个目录。例如：src/\*\*/\*.cpp

Tips：

- 由于操作系统通常都会生成一些文件，因此匹配时一定要加文件类型，例如cpp、c等，防止误将系统文件当成源文件编译
- 匹配不包括以英文句号“.”开头的文件和文件夹（也不递归它的内容）
- 自动忽略根目录中的“config”、“build”、“.vscode”文件夹

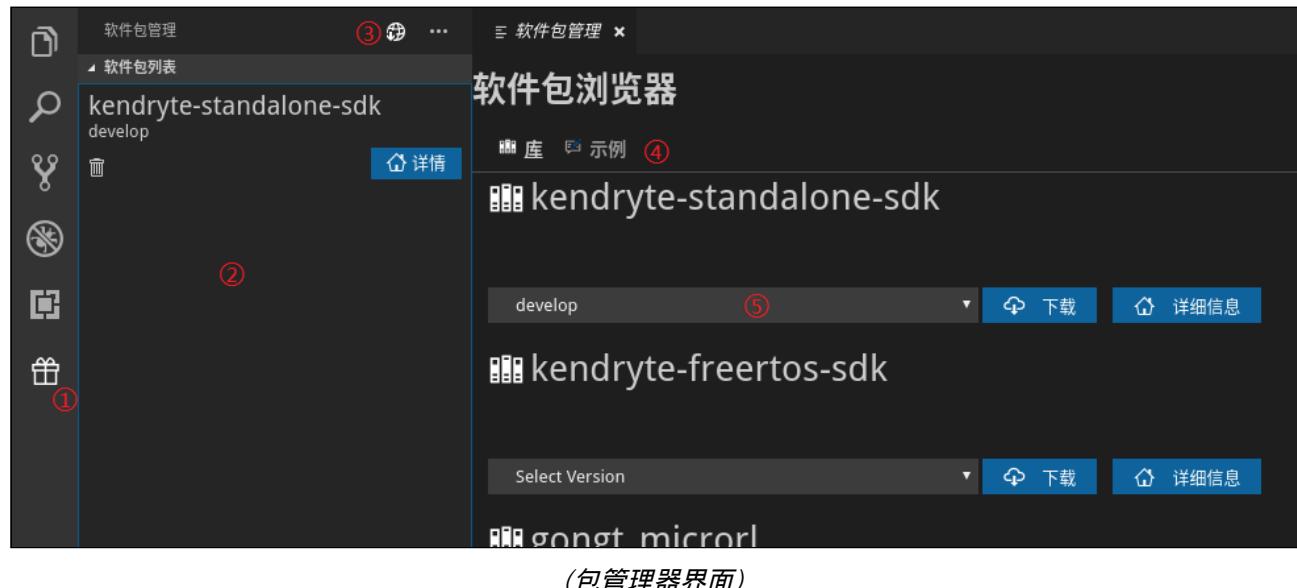
**项目依赖**：指定依赖的软件包，以及它们的版本

每行一个软件包，格式为“软件包名称=版本号”。其中，版本号可以为url，内容是这个软件包的zip压缩包地址。

所有此处列出的包将被编译。

要特别注意的是：如果此处没有列出，它是不会被编译的，即使它就在“kendryte\_libraries”文件夹中也是如此。

## 4.3 包管理器



(包管理器界面)

包管理器作为Kendryte IDE的主要功能之一，它的按钮在界面最左侧（①处）。点击即可打开**本地软件包列表**面板（②）。其中列出当前已经安装的软件包列表。

**要安装软件包：**

1. 点击图中③处浏览器按钮，打开软件包浏览器
2. ④处选择要下载的软件包种类
3. 然后再⑤处选择要下载的版本
4. 点击下载按钮
5. 等待右下角弹出成功提示

“库”类型的包下载后，保存在当前项目的“kendryte\_libraries”文件夹中，由于每次点击下载按钮都会覆盖，因此最好不要修改其中的内容。

包安装后，会自动修改项目配置（见 [项目配置\(see page 20\)](#) 界面），编译时将被包括。

“示例”类型的包下载后，会弹出选择文件夹，然后保存到指定的文件夹中。每个示例有自己的文件夹，不需要手动创建。

将示例重复下载到同一个位置，将会产生带序号结尾的新文件夹，已有文件不会被覆盖。

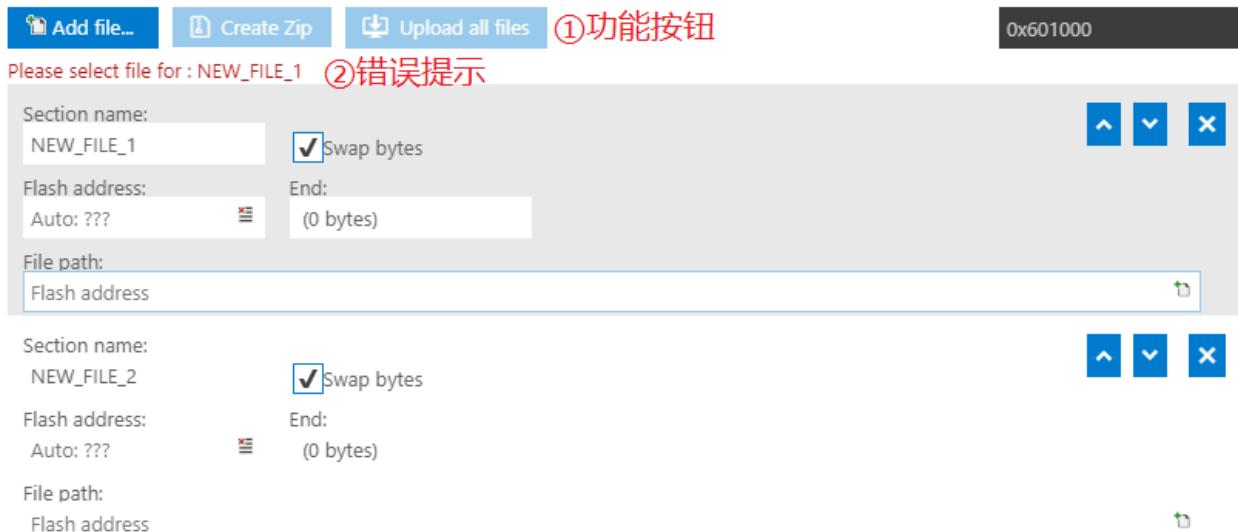
## 4.4 flash编辑工具 (kflash)

这个工具与kflash类似，提供兼容的kf pkg文件，而且包含一个图形界面。

要打开flash编辑工具，请点击“Kendryte菜单 > 闪存编辑器”

### 4.4.1 配置界面

#### Flash manager



(flash编辑工具)

当有错误时，①处按钮将会禁用（添加文件除外），修复错误后：

1. 点击创建压缩包按钮创建kf pkg文件，生成的文件保存在build目录中，文件名是项目名称.kf pkg
2. 刷写页面中的文件到flash中

文件列表中可以编辑要烧写的文件信息。对每个文件可以进行修改名称、修改烧写地址、调换位置、删除等操作。

1. 文件名：定义如何在代码中引用该文件，它必须是合法的C语言标识符（见下方如何实用）
2. flash地址：可输入一个十六进制地址，要求不能小于基址、按要求对齐。点击文本框右侧的按钮，可以使用自动分配模式。如果没有问题，文件结尾的地址将自动显示（不可编辑）。
3. 文件路径：输入或选择（文本框右侧按钮）本地一个文件，将其内容写入flash，文件必须位于当前项目内，相对于项目根目录。
4. 交换字节：自动转换文件字节顺序，以符合K210平台要求。（即翻转每4字节）

#### 4.4.2 引用文件

如果有文件命名为：TEST\_FILE\_1

则编译前将生成常量定义文件，其内容大致是：

```
1 #define TEST_FILE_1_START 0x601000  
2 #define TEST_FILE_1_END 0x602000  
3 #define TEST_FILE_1_SIZE 48
```

在代码中可以使用**TEST\_FILE\_1\_START**和**TEST\_FILE\_1\_END**两个常量读出对应文件内容。

flash编辑工具不包括闪存驱动，你需要通过包管理器下载对应驱动程序。

例如kendryte官方为KD233开发板提供的驱动程序是**w25qxx**，下载安装后，可以在代码中这样使用它：

```
1 char *file = malloc(TEST_FILE_1_SIZE + 1);  
2 w25qxx_read_data(TEST_FILE_1_START, file, TEST_FILE_1_SIZE);  
3 // printf(file);
```

## 5 特色功能

### 5.1 工作区

IDE包含工作区（Workspace）支持。工作区是多个项目的集合。

要使用工作区，有两种方式开始：

- 打开一个目录后，将另一个目录拖动到“资源管理器”面板中
- 在目录树空白处点击右键，选择“将文件夹添加到工作区...”

你可以把工作区保存到文件（文件 > 将工作区另存为...）。

该文件记录了所有打开的文件夹的路径，下次只需打开它（文件 > 打开工作区...），即可一次打开所有文件夹。

当打开超过一个目录时，状态栏将出现当前活动项目的名称：



当点击编译、烧写、调试等按钮时，针对的是这个项目。点击它，可以修改活动项目。

#### 5.1.1 编译

当对活动项目进行编译时，IDE会检查其他打开的文件夹，并试图找到依赖项目，并覆盖当前项目的依赖。

例如：

- 同时打开pwm\_standalone\_demo与kendryte-standalone-sdk两个项目
- 当前项目为pwm\_standalone\_demo
- pwm\_standalone\_demo依赖kendryte-standalone-sdk
- kendryte-standalone-sdk项目的类型是库（Library、PrebuiltLibrary）

当点击编译按钮时，先编译打开的kendryte-standalone-sdk，然后编译pwm\_standalone\_demo，并把它们链接。

注意：

- 打开的其他Executable项目不参与依赖查找，即使名字相同
- 如果项目的kendryte\_libraries中也有相同依赖，则kendryte\_libraries中的将被忽略（如同没有安装它一样），打开的项目总是优先生效
- 其他依赖项目仍然使用kendryte\_libraries中的
- 在Windows平台有限制：当前和所有依赖项目必须在相同盘符上

## 6 生成（编译）系统

这部分手册主要与编译过程相关，目的是解释IDE编译时发生了什么，如果编译时没有发现问题，可以跳过。

### 6.1 CMake

Kendryte IDE集成CMake工具，版本非常新，可在主菜单“帮助 > 关于”中查看。

每次编译启动一个新的CMake，参数是：

`| cmake --build /path/to/project -j ${CPU数量}`  
工作目录也是当前项目路径。

CPU数量自动探测，如果探测失败则固定为2。不过，这个选项在开启CMake详尽输出时无效。

除编译外的所有功能，例如configure，均由一个cmake实例完成，每个IDE窗口都会启动一个这样的CMake实例。它的参数是：

`| cmake -E server --experimental --pipe=一个临时管道文件`  
工作目录是内部的cmake安装路径。

如果你需要给cmake添加环境变量，可以在当前项目.vscode文件夹中创建文件“cmake-env.json”，内容是一个键值对object。

这些环境变量同时也是configuration的参数

IDE还会添加这些环境变量：（优先级低于cmake-env.json）

- PATH：
  - windows：toolchain/bin加上C:\Windows\system32等关键路径，不包含自定义路径
  - 其他：在当前环境变量前加上toolchain/bin
- PYTHONHOME：（仅windows）指向一个内部python library
- PYTHONPATH：在系统PYTHONPATH后添加toolchain/share/gdb/python
- PYTHONDONTWRITEBYTECODE：yes

### 6.2 CMakeLists.txt生成

1. 首先检查CMakeLists.txt是否已存在，检查其中是否有IDE的警告注释。找到就删除这个文件，如果没有则报错，防止错误覆盖。
2. IDE中有一些GUI工具，例如闪存编辑。他们此时会在config文件夹中生成C语言源码。
3. 读取kendryte-package.json，生成一些基础脚本
4. 通常项目有依赖，对列表中每个依赖项目递归这个生成过程，然后对每个项目调用add\_subdirectory()
5. 匹配源码文件、收集头文件目录等工作
6. 生成最终的CMakeLists.txt



## 7 调试功能简介

调试程序是一个庞大的话题，本节只说明“在IDE中调试勘智芯片上的程序”是怎样操作，不包括如何找到bug等内容。

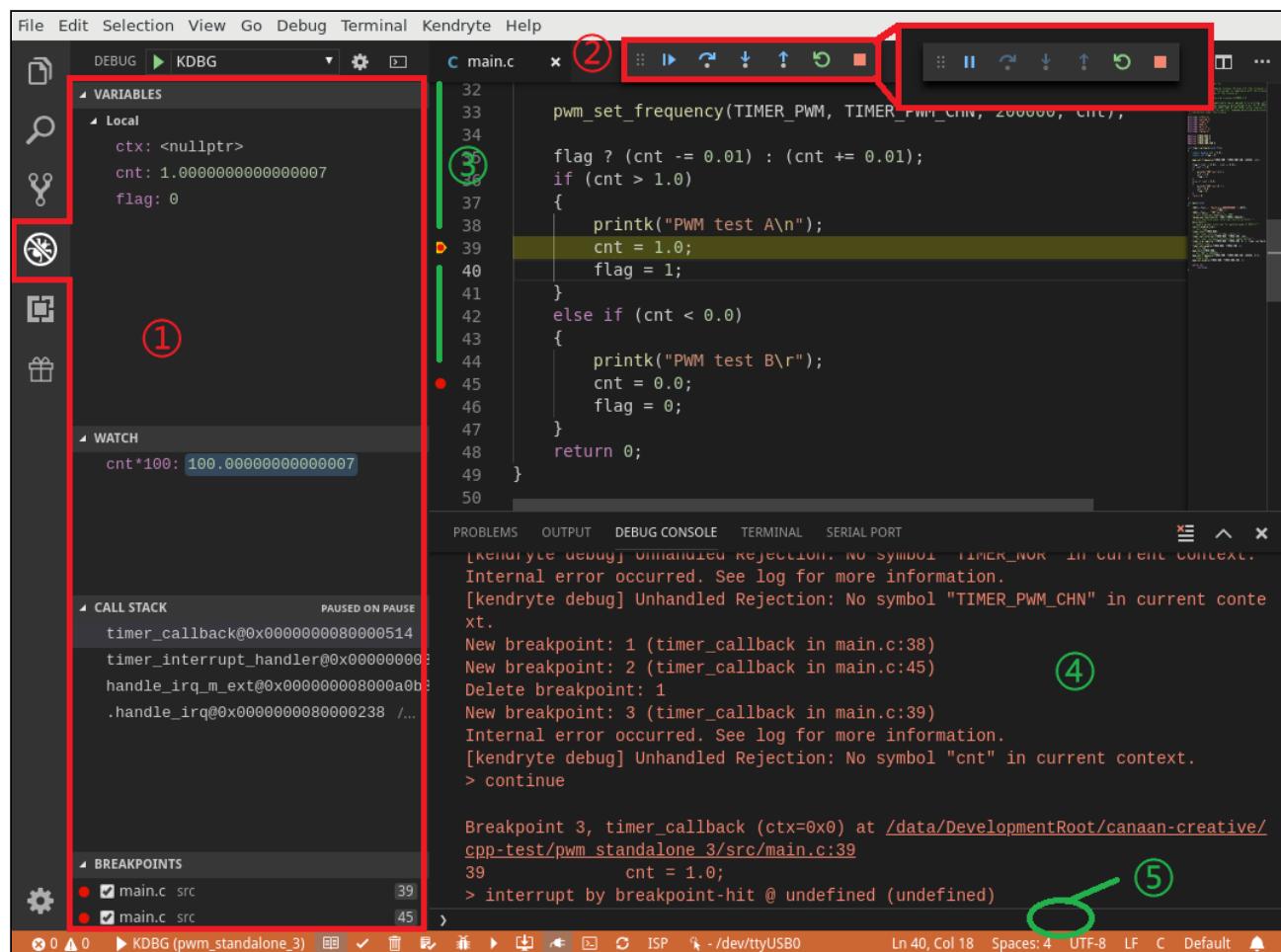
### 7.1 说明：

IDE集成图形化调试界面、gdb、openocd，不需要单独下载。

使用调试功能的条件：

1. 正确链接JLink和开发板
2. 安装JLink驱动（Windows）
3. 安装libusb等依赖（Linux、Mac）

### 7.2 调试界面



(调试界面)

图中：

①. 调试信息：从上到下分别是：

1. 变量列表：显示当前上下文中的变量和值
2. 监视列表：可以添加（点击右键）表达式，暂停时自动运行并显示结果
3. 调用栈：双击可查看对应位置的信息
4. 断点列表：支持 **文件名:行号** 和 **函数名** 两种断点

②. 调试控制（右边的是运行状态）。按钮分别是：

1. 继续或暂停
2. 单步运行
3. 单步运行，跟随函数调用
4. 运行到当前函数返回
5. 软重置：建议在暂停状态使用，点击后立即运行load指令
6. 结束调试

③. 在行号标尺左侧点击鼠标即可快速切换断点

④. gdb输出信息

⑤. gdb表达式输入框

## 7.3 常见操作：

### 7.3.1 重置软、硬件状态

由于硬件调试有一定不确定性，建议调试前重置开发板和JLink设备。步骤是：

1. 停止OpenOCD：点击菜单“Kendryte > OpenOCD > 停止OpenOCD服务”
2. 断开JLink的USB线、关闭开发板的电源
3. 打开开发板电源
4. 连接JLink

### 7.3.2 启动调试



点击IDE下方状态栏的DEBUG按钮：

此按钮将编译当前项目，然后进行调试。如果编译不通过，则弹出错误，不启动调试。

这个按钮与顶部菜单“Kendryte > 部署和调试 > 编译并调试”功能相同

注：如果程序较大，调试器需要更久来启动，可以查看调试信息了解进度。程序写入过程不支持取消，如果想要放弃调试，请直接重置OpenOCD和硬件。

### 7.3.3 查看调试信息

点击菜单“查看 > 调试控制台”。也可以按键盘 Ctrl+Shift+Y。

如果IDE提供的调试功能不能完全满足你的需求，可以使用**表达式输入框**（在“调试控制台”的底部），可以输入gdb指令，结果会显示在上方显示，效果与命令行方式调用gdb类似。

### 7.3.4 查看变量值

调试时，**暂停状态下**，鼠标指向变量，可以看到它在**当前上下文**的值。

例如：

```
void sub() {  
    int a = 2;  
    return; //命中此行断点  
}  
void main() {  
    int a = 1;  
    sub();  
}
```

在这种状态下，不论鼠标指向sub中的a，还是main中的a，**显示的都是sub中a的值**。

如果想要了解main函数中a的值，需要在调试信息“调用栈”中双击main函数对应的行。

注：在非调试或运行状态下，看到的是变量的声明，而不是值。

## 7.4 在Linux和Mac中安装系统依赖

## 7.5 在Windows上安装JLink驱动

## 8 常用操作

### 8.1 一、界面、操作、交互

#### 8.1.1 打开/隐藏底部面板

1. 按键盘Ctrl+J可显示/隐藏底部面板
2. 也可以点击面板图标，直接打开对应面板。例如左下角的错误列表即可打开错误列表面板。

#### 8.1.2 打开系统终端软件

1. 按下键盘Ctrl+Shift+C
2. 点击菜单“Kendryte > 打开系统终端”

IDE会尝试找到系统终端程序。例如windows上是cmd.exe。

如果没有找到，或者找到的程序不是你想要启动的，则在IDE设置中搜索“terminal exec”，可手动指定需要打开的终端类型。

此时终端内可以直接使用kendryte toolchain等工具（如：riscv64-unknown-elf-gdb）

#### 8.1.3 快速执行命令

在IDE任何地方按下键盘Ctrl+Shift+P，即可快速执行指令。可以是编译、单步运行、打开文件等等一切IDE支持的动作。

根据编辑器状态不同，指令列表有一些区别。例如没有打开任何文件时，“复制文字”命令不显示。

指令搜索框可以使用中文或英文，匹配不区分大小写，支持缩写。重新按下Ctrl+Shift+P可以清除搜索框。

例：要复制选中的一段代码：

1. 按下Ctrl+C
2. 点击菜单“编辑 > 复制”
3. 按键盘Ctrl+Shift+P，输入“copy”，点击列表中“文件：复制”命令

### 8.2 二、开发、编辑

### 8.3 三、编译、调试、运行

## 8.4 四、维护

### 8.4.1 备份IDE设置

两种方式：

1. 打开IDE安装目录，进入UserData/latest/user-data/User，备份 settings.json 文件
2. 打开设置编辑器，点击右上角JSON编辑模式按钮，然后复制其中内容到笔记本工具

### 8.4.2 查看日志

1. 如果更新器报错，点击界面右下角”Show Log“按钮，在右侧文本框中按右键，可以复制内容。
2. 到IDE安装目录中的UserData/latest/user-data/logs文件夹中查看日志，日志文件按照启动时间归档到各个文件夹中，通常只需要看最新的一个。（如果IDE无法正常启动，只能在此处查看日志）
3. 打开底部面板（Ctrl+J），点击①”输出“选项卡，即可查看日志，日志内容与2.的日志文件相同。

多个日志文件可以在②处切换。③处按钮可以清空日志窗口、停止（继续）滚动输出。

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL SERIAL PORT
 ipcChannelMain.ts:93:22) at REMOTESERVICECHANNEL.<anonymous> (./src/vs/kendryte/extension/serialport/serialport.js:11:1)
 ipcChannelMain.ts:93:22) [38;5;9m[0m at Generator.next (<anonymous>)
 [38;5;9m[0m at new Promise (<anonymous>)
 [38;5;9m[0m at _awaiter (../out/vs/kendryte/services/ipc/electron-main/ipcChannelMain.ts:93:22)
 [38;5;9m[0m at RemoteServiceRunner.call (../out/vs/kendryte/services/ipc/electron-main/ipcChannelMain.ts:93:22)
 [38;5;9m[0m at ChannelServer.onPromise (.src/vs/base/parts/ipc/common/ipcChannelServer.ts:11:1)
 [38;5;9m[0m at ChannelServer.onRawMessage (.src/vs/base/parts/ipc/common/ipcChannelServer.ts:11:1)
 [38;5;9m[0m at ChannelServer.protocolListener.protocol.onMessage.msg (.src/vs/base/parts/ipc/common/ipcChannelServer.ts:11:1)
 ipc.ts:256:63) [38;5;9m[0m at Emitter.fire (.src/vs/base/common/event.ts:584:16)
 [38;5;9m[0m at i (.src/vs/base/common/event.ts:56:84) [38;5;9m[0m at Emitter.fire (.src/vs/base/common/event.ts:584:16)
 [38;5;9m[0m at e (.src/vs/base/common/event.ts:74:97) [38;5;9m[0m at Emitter.fire (.src/vs/base/common/event.ts:584:16)
 [38;5;9m[0m at EventEmitter.fn (.src/vs/base/common/event.ts:349:41) [38;5;9m[0m at EventEmitter.emit (events.js:182:13)
 [38;5;9m[0m at WebContents.<anonymous> (./build/electron/resources/electron.asar/browser/api/web-contents.js:285:13) [38;5;9m[0m at WebContents.emit (events.js:182:13)

```

## 9 常见问题 (FAQ)

### 9.1 常见问题

**安装后几分钟提示“扩展已在磁盘上更改， 请重新加载”**

点击重新加载， 也可以直接忽略。它不影响IDE或插件的正确运行。

**我遇到了问题， 怎么办？**

1. 到论坛 ([forum.kendryte.com](http://forum.kendryte.com)<sup>1</sup>) 的IDE版块发帖
2. 通过QQ、微信联系我们
3. 技术人员还可以到github提issue

描述问题时记得附带相关日志

**我不想用IDE， 在哪里找到相关资料？**

<http://kendryte.com><sup>2</sup>

**按Ctrl点头文件跳不过去**

首先确定当前项目是可以通过编译的， 不能编译的情况下， 很多头文件不能跳转。

还有部分特殊文件（例如C运行时）确实无法跳转。

如果可以编译， 且是普通的头文件， 那么你可能发现了bug， 建议发帖询问。

**点击调试， 提示“launch.json”什么的， 调试没有开始**

这说明OpenOCD没有正确运行， 可以查看它的日志寻找原因。

可以点击“Kendryte菜单 → OpenOCD → 重新启动OpenOCD”重启。

有时还需要重置各种硬件。

**不能调试， OpenOCD日志中只显示了一个路径， 就没有其他输出了**

在kendryte菜单中打开系统终端， 输入openocd并回车运行， 看看是否缺少系统依赖。

在linux或mac， 终端可能提示“Missing xxx.so”， windows上将弹出窗口显示缺少xxx.dll。对于linux、 mac，请自行安装对应依赖。windows请提交issue告诉我们。

如果看到OpenOCD输出了“Kendryte”图案， 说明不缺少依赖。

**不能调试， 点了没反应**

和上一个问题雷同， 运行riscv64-unknown-elf-gdb， 看看是否缺少依赖。

**不能编译， 似乎没有输出**

运行riscv64-unknown-elf-gcc， 看看是否缺少依赖。

**烧写提示错误“greeting”**

---

<sup>1</sup> <http://forum.kendryte.com/>

<sup>2</sup> <http://kendryte.com/>

一般是因为2个串口跳线没有接上，自动重启不一定支持全部第三方开发板，对于不支持的开发板，根据其说明进入ISP模式后再点烧写即可成功。

**看完文档，我还是不会用，怎么办**

到论坛 ([forum.kendryte.com](http://forum.kendryte.com)<sup>3</sup>) 的IDE版块发帖询问，论坛上用中文和英文都可以

**编译失败，问题列表空白（或没有编译相关的错误）**

可能遇到了链接错误，链接错误通常无法定位到特定文件，因此无法在问题中显示。需要点击左下角的红色叹号查看CMake原始输出。

**编译失败，CreateProcess: No such file or directory**

可能是系统安装过mingw/msys2导致某种冲突，这个问题还没有解决

---

<sup>3</sup> <http://forum.kendryte.com/>