应用笔记

在 KF32 IDE 集成开发环境中使用外设库与例程

前言

本应用笔记将介绍如何在 KF32 IDE 集成开发环境中使用 ChipON 提供的外设库与例程开发应用代码，使用户快速熟悉如何在 KF32 IDE 中新建项目，运用外设库资源，导入参考例程，进行编译调试。

KF32 IDE 集成开发环境是 ChipON 基于 Eclipse 平台，自主研发，专门针对 KungFu32 系列单片机开发设计的新一代集成开发环境，具有文本编辑功能丰富，编译效率高等特点。

本应用笔记使用的 KF32 IDE 与 KF32Fxxx 外设固件库可以从 ChipON 官方网站 www.chipon-ic.com 下载。

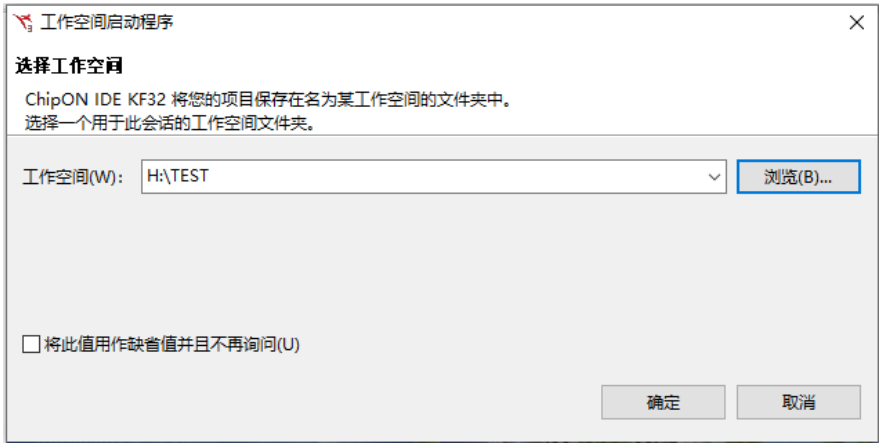
目录

1. KF32 IDE 的“项目”与“工作空间”

KF32 IDE 继承了 Eclipse 强大的项目管理能力，通过“项目”与“工作空间”的概念来组织和管理源代码，为种类繁多的项目提供了良好的管理解决方案。

工作空间（Workspace）是项目（Project）的合集，项目是源代码文件的合集。将相同类型的项目或者有相互关联的项目放在同一工作空间中，非常便于管理与历史遍历。

KF32 IDE 集成开发环境中，项目必须放在工作空间目录下才可以被使用与编译。在首次启动 KF32 IDE 时，它将提示即将打开的工作空间位置。



通过“浏览”选项可以切换到其他工作空间或者建立新的工作空间。

工作空间目录下的.metadata 目录存储了该工作空间中项目和插件的配置信息，此目录的存在将告诉 KF32 IDE 当前目录是有效的工作空间。.metadata 目录中还包含了以.log 命名的文件。此文件将包含在运行 KF32 IDE 时可能抛出的所有错误或异常。如果 KF32 IDE 在某一刻意外崩溃，该文件将对错误诊断十分有用。（注意：工作空间目录下只能包含项目文件，不能包含其他工作空间，否则可能会导致 IDE 崩溃）

单击“确定”选项后 KF32 IDE 进入项目开发界面。

“项目资源管理器”显示当前工作空间下管理的项目。如果是新建的工作空间，“项目资源管理器”中没有任何项目信息，需要用户新建或者导入项目到该工作空间下，才可对项目进行编辑与编译。

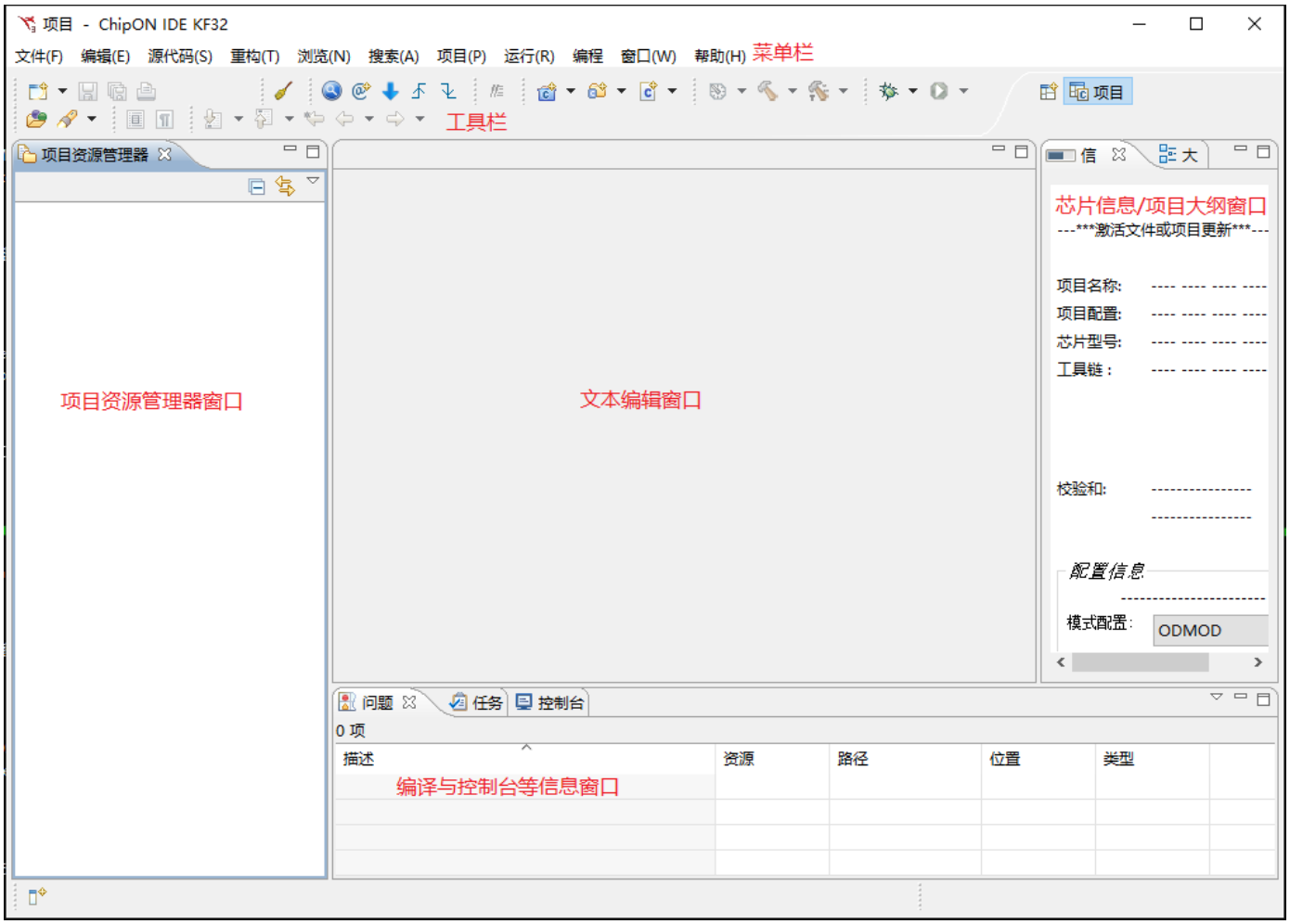
“菜单栏”提供了 KF32 IDE 功能配置等选项。

“工具栏”配合 ChipON 提供的编程器，设置了寻找编程设备，编译，运行上下电，

调试等用户常用功能。

“编译与控制台信息窗口”提供了编程过程信息，便于用户查错。

“芯片信息/项目大纲窗口”提供了当前项目的芯片型号，代码内存占用，芯片&编程器配置，项目内容大纲（文件包含的变量/函数）等信息。

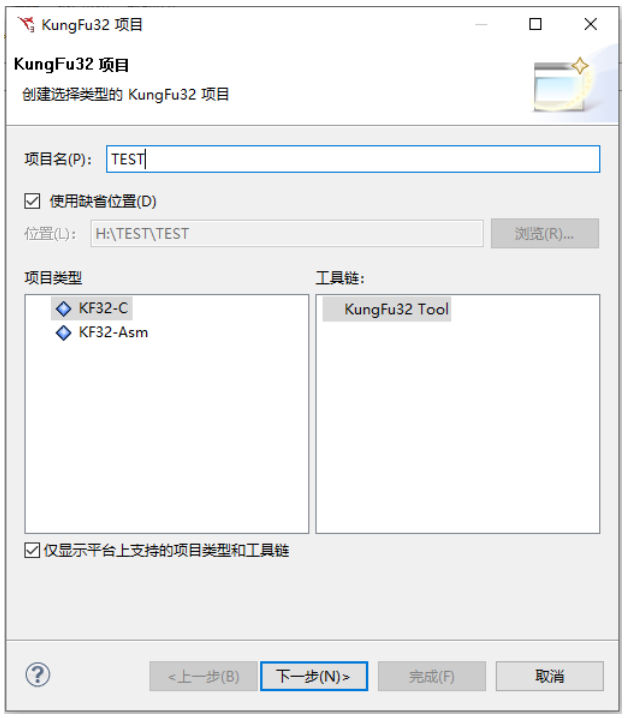


2. 在 KF32 IDE 中新建项目并引用标准外设库

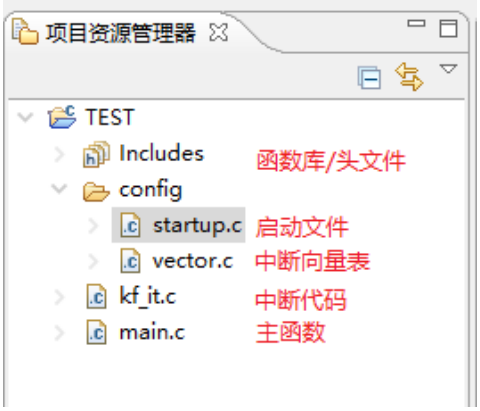
2.1 新建项目

在“菜单栏”的“文件”选项中，选择“新建”-->“KungFu32 项目”

在弹出的对话框中，对新建项目命名，C 语言编程项目类型选择“KF32-C”，单击“下一步”，弹出界面的设置为默认即可，继续单击“下一步”，出现“芯片型号选择窗口”，选择对应的芯片型号，点击“完成”，即可在“项目资源管理器”中看到新生成的项目文件。

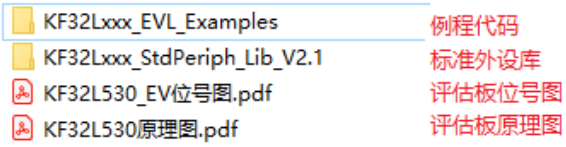


新生成的项目文件如下图，Include 中包括了编译器支持的头文件集与编译器支持的如 math、stdio 等函数头文件；startup.c 为启动代码，进行 RAM 初始化；vector.c 为中断向量定义；kf_it.c 中断入口代码；main.c 主函数。

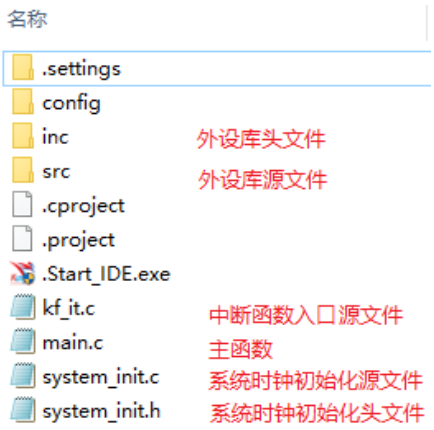


2．2 引用标准外设库

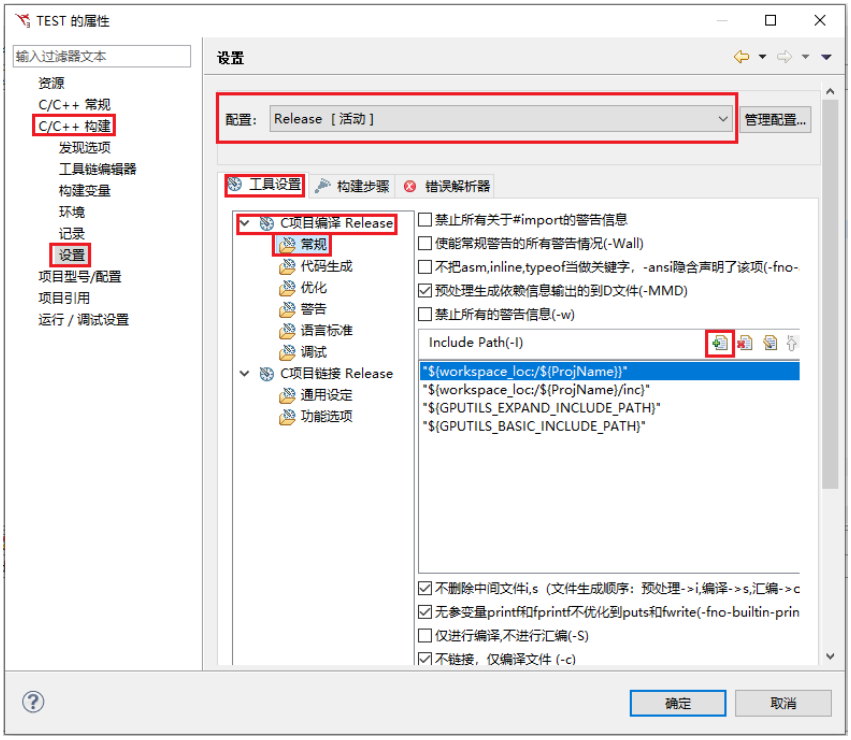
ChipON 针对 KF32F、KF32L、KF32LS、KF32A 等系列产品，提供标准外设库与例程代码。相关资源可在官网下载。本文以 KF32L 系列为例，说明引用 L 系列的标准外设库方法。在官网下载“KF32Lxxx 开发资源”压缩包,开发资源一般提供如下资料：



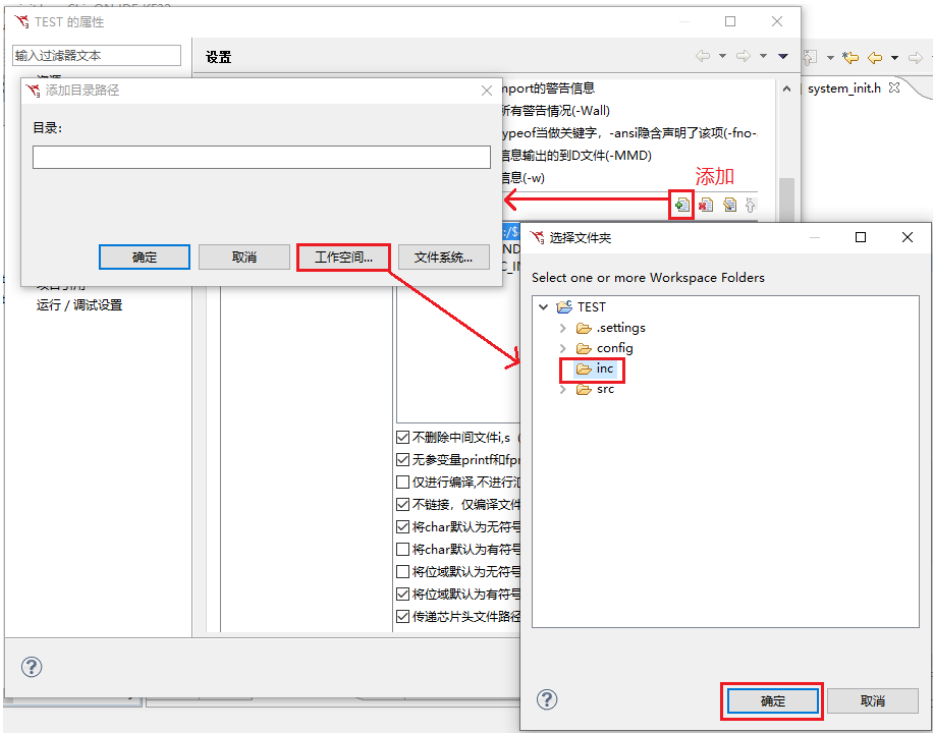
将标准外设库中的内容：外设库头文件，外设库源文件，中断函数入口源文件，主函数，系统时钟初始化源文件，系统时钟初始化头文件，选中复制。在 IDE 的“项目资源管理器”的项目上，左键粘贴，将资源复制到项目空间中。



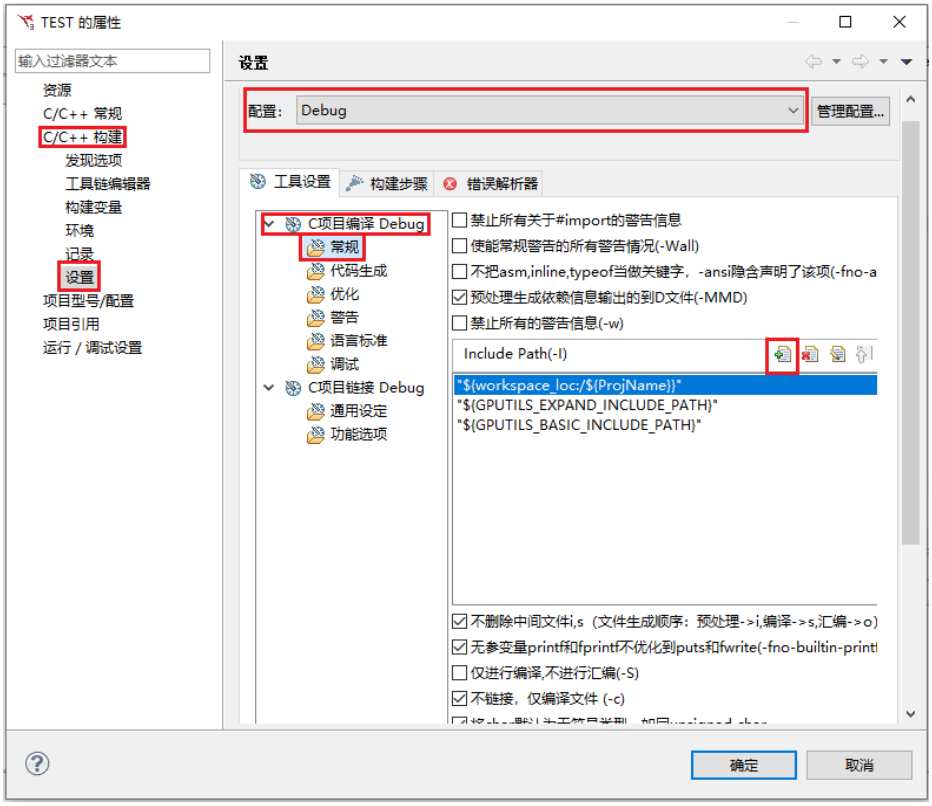
在项目名上单击左键，选则“属性”菜单，弹出的对话框中，“C/C++构建”的下拉菜单中，选择“设置”，新建项目的“配置”选项默认为“Release[活动]”，然后在“工具设置”-->“C项目编译 Release”-->“常规”菜单中，添加 inc 文件路径，参考下图：



选择“添加”，在弹出的对话框中，选择“工作空间”，在下一级弹出对话框中，选择项目文件中的“inc”文件，点击“确认”返回上一级，直到配置完成。



完成上述对 inc 文件路径配置，项目可以进行 Release 模式编程。Debug 模式，需要在“C/C++构建”的“设置”中，“配置”选项选择为“Debug”，然后在的“工具设置”-->“C 项目编译 Debug”-->“常规”菜单中，添加 inc 文件路径：



2．3 编译

编译器的工具栏提供“构建”与“重构”选项：

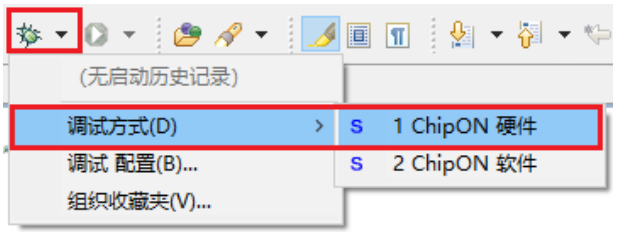
如项目是第一次编译，IDE 会遍历链接所有.c 文件，编译项目。

“构建”除非涉及头文件修改，否则 IDE 只编译发生了修改的.c 文件。在“项目资源管理器”中，选中项目名，点击鼠标右键，弹出的对话框中也能找到“构建项目”选项。

“重构”会重新遍历所有的.c 文件重新编程。

“重构”与“构建”默认为 Release 模式，编译会在项目内生成新的 Release 文件夹，项目编译成功，该文件夹会包含编译的过程文件.o 文件，变量的内存分布.map 文件，机器码与汇编注释文件.lst 和生产文件.hex 等。

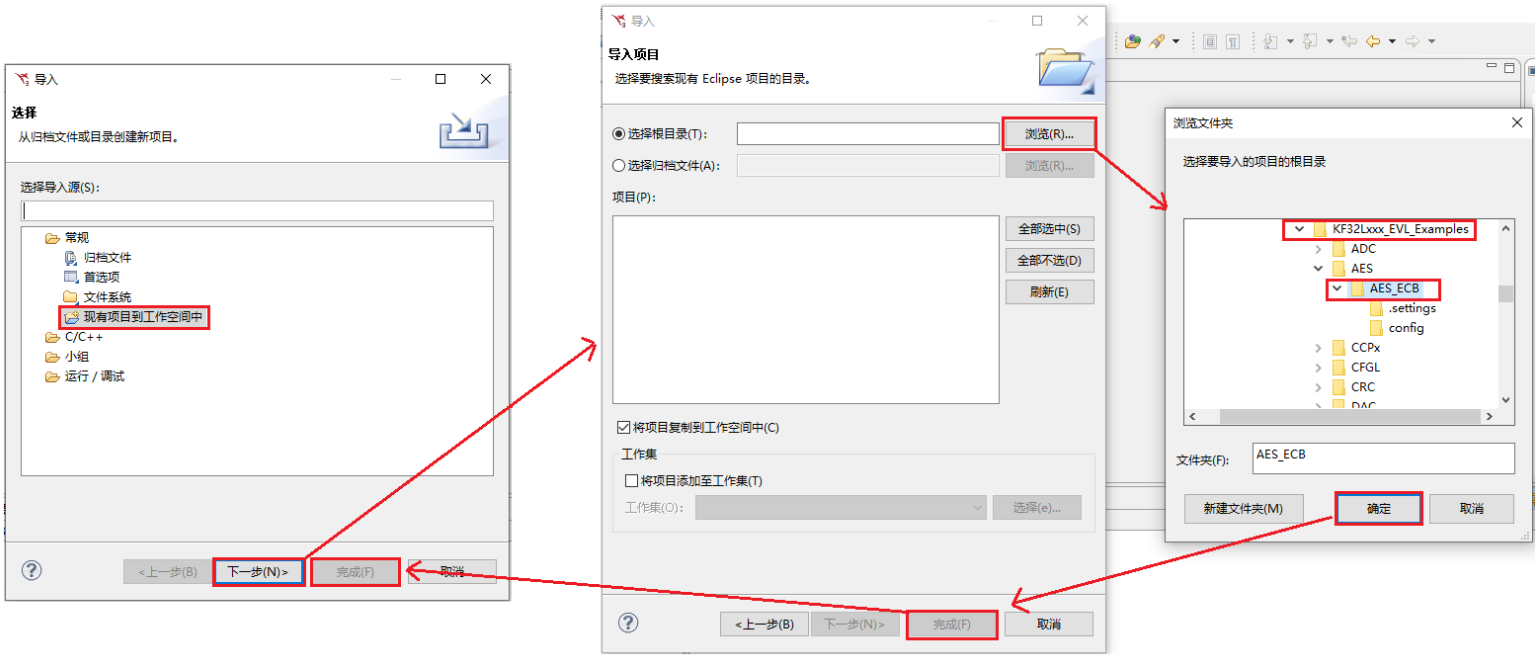
Debug 模式，需要在“构建”与“重构”工具旁的下拉菜单中选择“Debug”，与 Relaese 模式一样，第一次编译 Debug 会遍历所有的.c 文件。编译会在项目内生成新的 Debug 文件夹。编译成功，选择工具栏的“调试”即可链接芯片硬件进行调试。



3. 在 KF32 IDE 中导入官方例程

KF32L 系列例程为例，导入官方例程。在 IDE 的“菜单栏”-->“文件”选项中，选择“导入”：

弹出的“导入”对话框，选择“现有项目到工作空间中”，在“导入项目”对话框，选择“浏览”，找到官方固件“KF32Lxxx_EVL_Examples”文件夹，选择需要导入的例程目：



项目导入后，打开 system_init.h 文件，查看例程需求的外设库头文件。按照 2.2 章节，对项目添加相关的外设库头文件与源文件。也可以直接全部引用 inc 与 src 内容，全部引用因冗余的.c 文件过多，第一次编译时间会比较长，建议只引用例程需要的标准外设代码。

在 system_init.h 中，在对芯片型号的宏定义下，通过系统时钟的宏定义，可以对系统运行时钟进行配置：

```
/* 系统时钟选择 */
#ifdef KF32L530
//#define SYSCLK_FREQ_48MHz 48000000
#define SYSCLK_FREQ_72MHz   72000000
//#define SYSCLK_FREQ_96MHz 96000000
//#define SYSCLK_FREQ_120MHz    120000000
#endif
```

在 inc 文件夹的 KF32L_BASIC.h 中，通过宏定义 USE_CHECK_ASSERT 配置用户断点检查功能，开启后代码会对标准外设库的输入参数进行合法检查，但会增加代码运行时间，配合主函数的 void check_failed(uint8_t* file, uint32_t line)函数，可以对输入的错误参数进行定位。

```
/* 用户检查断点功能 */
#define USE_CHECK_ASSERT   1
#if USE_CHECK_ASSERT
   #define CHECK_RESTRICTION(expr) ((expr) ? (void)0 : check_failed((uint8_t *)__FILE__, __LINE__))
   /*  校验错误函数 ------------------------------------------------------- */
   void check_failed(uint8_t* file, uint32_t line);
#else
#define CHECK_RESTRICTION(expr) ((void)0)
#endif /* USE_CHECK_ASSERT */
```

4. 版本历史

文档版本历史

日期	版本	变更
2019 年 11 月 29 日	1	初始版本。