

# DJYOS 开发工具手册

编写： 罗侍田 2015 年 02 月 12 日

Review： \_年\_月\_日

审阅：

## 贡献者列表：

向为本文档做出重要贡献的人致以诚挚的感谢，DJYOS 开源项目的发展，有你们一份功劳。

- 1、DJYOS 开发团队。
- 2、网友 刘建平，QQ：99961098

以开发 cortex-m3 为例，说明开发环境配置。

## 1. 安装 Eclipse 环境

从主页 [www.djyos.com](http://www.djyos.com) 下载 `eclipse.luna.rar` 文件并解压缩即可，已经安装好了 `gnuarm` 插件和 `cdt` 插件的。

## 2. 安装编译工具

从主页 [www.djyos.com](http://www.djyos.com) 下载 `yagarto-gcc4.9.rar` 文件并解压缩即可，然后配置 Windows 的 `path` 指向“`yagarto-gcc4.9\bin`”目录，注意不是 `yagarto-gcc4.9\arm-none-eabi\bin` 这个啊。如果你安装了多个可执行文件名是 `arm-none-eabi-gcc.exe` 的 `gcc` 版本，特别注意一定要在 `path` 列表中把“`yagarto-gcc4.9\bin`”放在第一个。

## 3. 安装 jvm

从主页 [www.djyos.com](http://www.djyos.com) 下载 `chromeinstall.exe.7z` 文件并解压缩运行，它会从网上下载 `jvm` 虚拟机并安装。心急的，可下载 `jre-7u10-windows-i586.rar` 直接安装，只是版本有点老，安装后最好 `update` 一下。

安装完成之后，打开命令提示窗口输入 `java -version`，如果得到类似图 3-1 的输出则表示安装 JRE 正确。

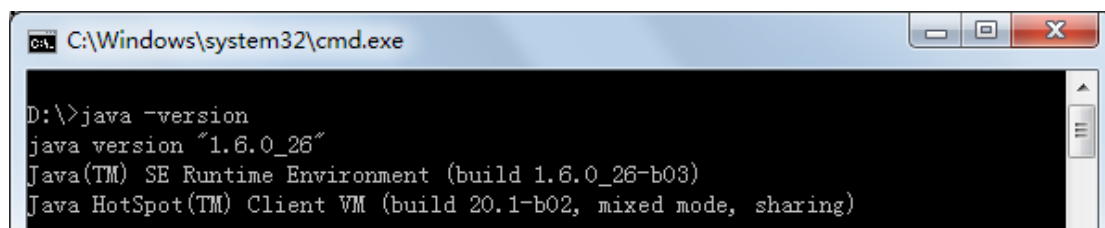


图 3-1 java 输出

## 4. 安装源码

从主页 [www.djyos.com](http://www.djyos.com) 下载 `djysdk.rar`，解压缩即可。它包括了 DJYOS 的所有源码，以及 `example` 工程，`eclipse` 环境可以直接导入这些工程。

## 5. 导入 example 工程

### 5.1. 初次使用 eclipse

首次打开 Eclipse 时都会弹出 Workspace Launcher 窗口，在这个窗口中您可以选择自己工程的主目录，这个目录将会作为在此 Eclipse 中新建工程的默认存放目录。勾选左下角的 *Use this as the default and do not ask again* 选项可使得下一次打开 Eclipse 时不再弹出这个窗口。

### 5.2. 使用 example 工程

由于新建工程要添加许多 OS 的，特别是跟硬件相关的文件，比较繁琐，建议用户不要新建工程，而是导入源码提供的 example，在此基础上增加自己的代码。

在 DJYOS 源码中，提供了一些 example 示例工程。

1. 在 *Project Explorer* 窗口中，单击右键后选择 *Import...*，如 图 5-1 所示；

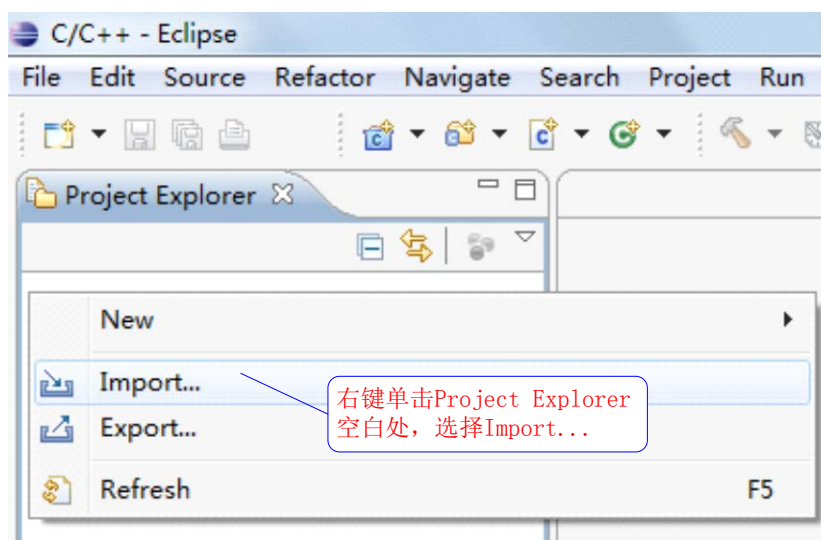


图 5-1 Project Explorer

2. 在弹出的 *Import/Select* 窗口中，依次打开 *General -> Existing Projects into Workspace*，点击 *Next >*。如 图 5-2 所示；

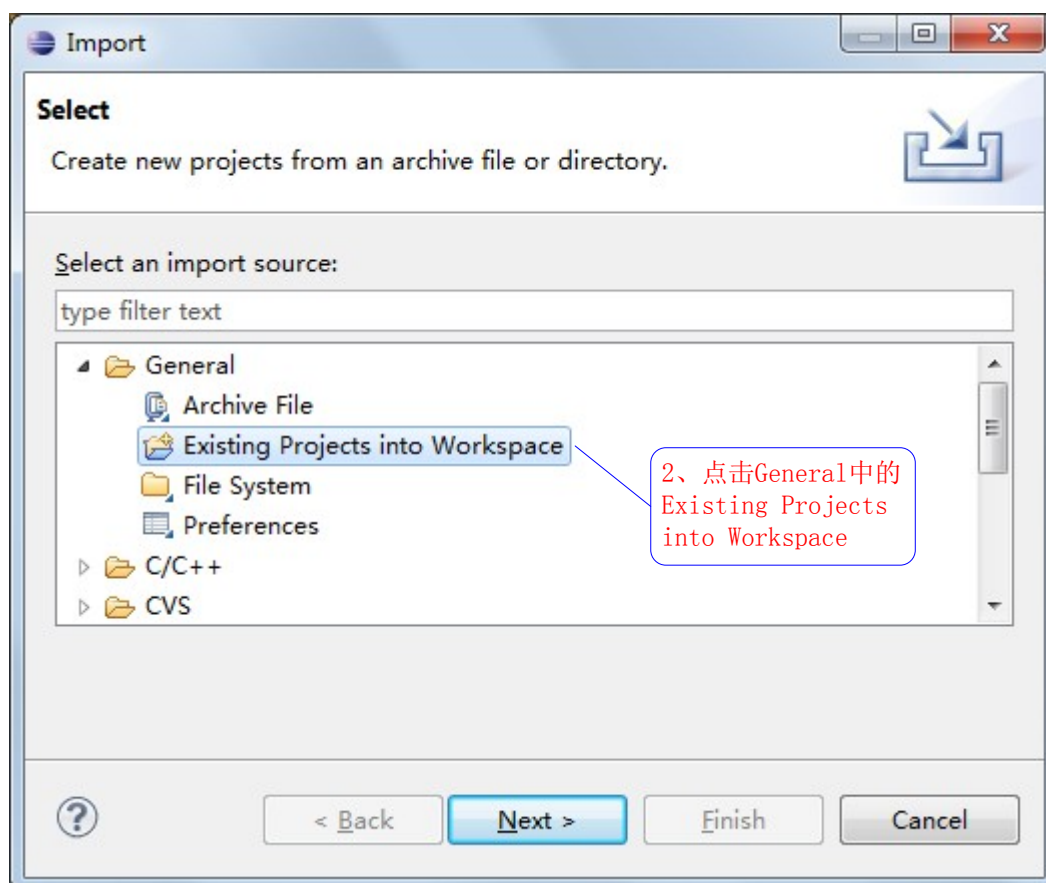
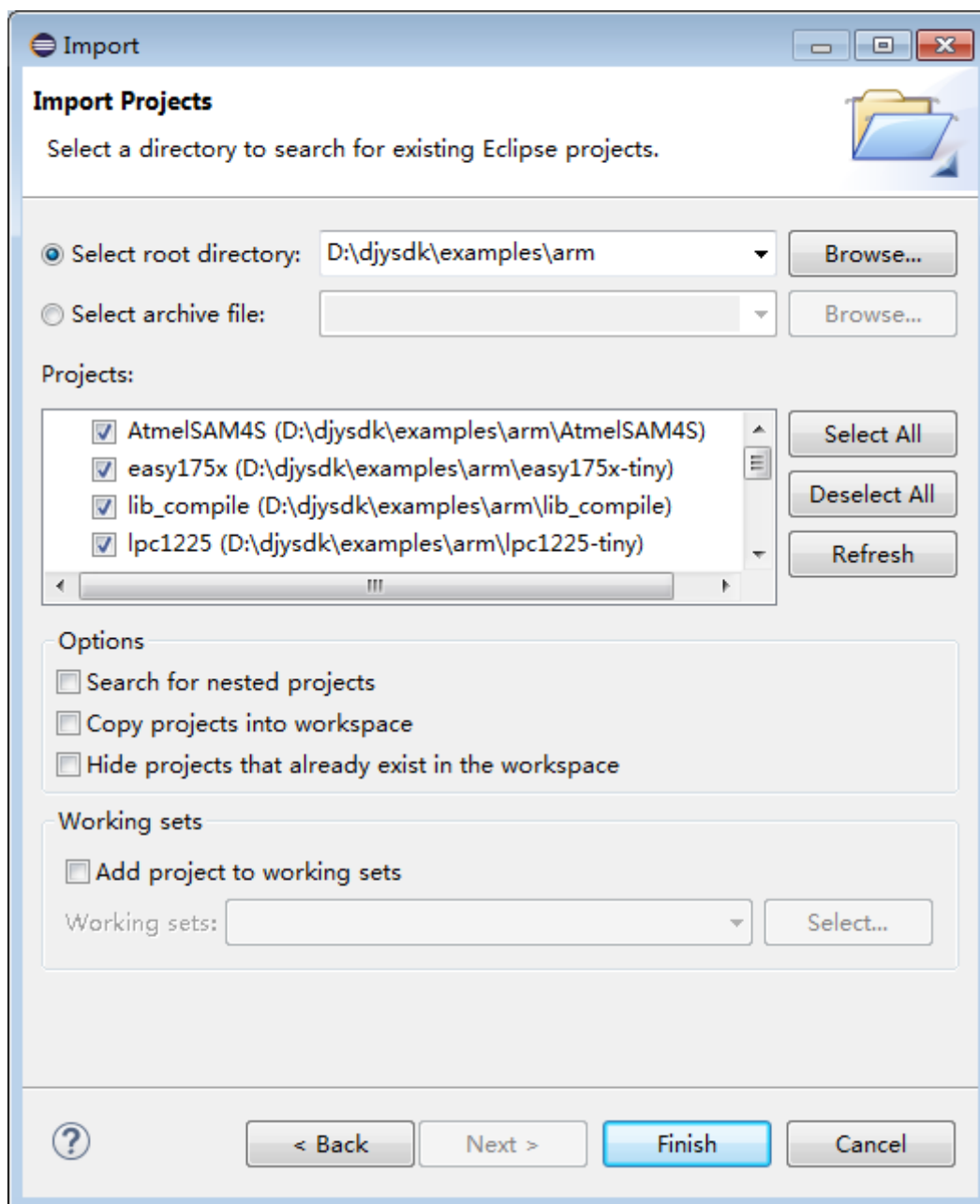


图 5-2 选择已经存在工程

接着会弹出 Import | Import Projects 窗口，如



3. 图 5-3 所示。在 *Select root directory:* 中填入包含您的工程的文件夹，在 *Projects* 中选择您想要导入的工程，之后便可以点击 *Finish* 完成了。

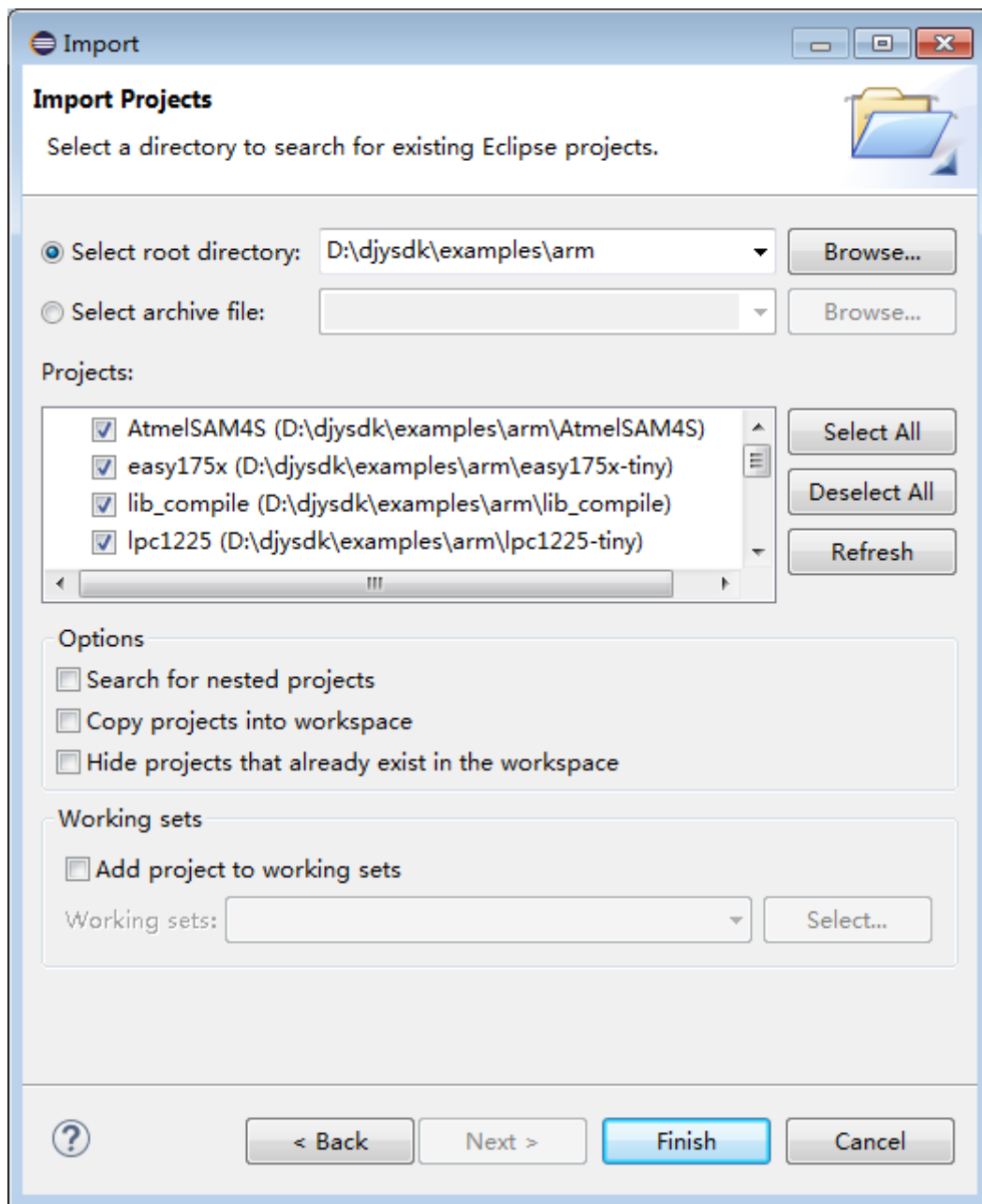


图 5-3 选择并拷贝工程

## 6. 创建新工程

设置 gcc 工程是个繁琐的过程，告诉你一个偷懒的方法：直接 copy 现有的工程文件夹，改名即可，然后在 eclipse 中导入，再次修改工程名，即完成了创建工程的绝大部分工作。你只需要加入自己的代码，再设置少量跟自己具体工程相关的部分就可以了。

## 7. 库说明

DJYOS 的 C 库，称作 libC 库，是移植自 newlib2.10，替换掉了其中跟操作系统相关和硬件系统相关的部分，也不再需要实现编译 newlib 所需要的桩函数。

DJYOS 本身，连同 BSP 在内，也被编译成了库，称作 libOS 库。

libC 和 libOS 共同构成了 DJYOS 的 c runtime library，暂时（DJYOS V1.10）还没有实现 C++库。

一般来说，C runtime library 需要与编译器结合，但 DJYOS 暂未实现自己的编译器，故目前与

应用程序工程做在一起了，这样做有以下好处：

- 1、库中的代码，可以做源码级调试，单步、观测变量、设置断点等都可以，完全与工程中的代码一样。
- 2、作为开源项目，更加方便爱好者阅读源码和参与开发。

不好的地方也有，就是编译 C 库比较费时间，如果修改了编译选项，还可能需要重新编译库。

## 8. 编译说明

小伙伴们，配置工程时，这个选项是万万不能勾选的，否则 linker 就不听你的 lds 文件指示了。

### ☐ Link-time optimizer (-flto)

libC 库、libOS 库，与应用程序放在一个工程中，第一次下载使用 DJYOS 时，**须先编译所需要的库，再编译应用程序**。导入工程后，至少会有 6 个编译配置，分别是 debug、release、libc-debug、libOS-debug、libc-release、libOS-release，分别是用于编译 debug 和 release 模式的可执行程序 and 库，如果还使用了厂家的固件库或者第三方库（例如 splite3），可能会更多配置。图 8-1 显示了一个 STM32F103 的板件的配置：

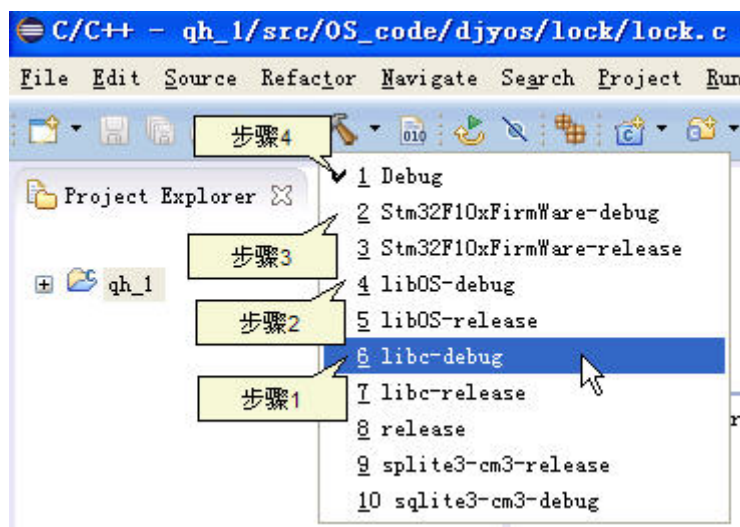




图 8-1

编译库的方法：

Step1: 点击  这个三角符号，选中需要编译的库，例如选择 libOS。

Step2: 再次点击小榔头 ，就会开始编译。

编译顺序：

debug 版本编译顺序

1.编译 libc-debug; 2.libOS-debug; 3.厂商固件-debug; 4.其他库（如果需要，例如 sqlite3）; 5.debug

release 版本编译顺序

1. 编译 libc-release; 2.libOS-release; 3. 厂商固件 -release; 4. 其他库（如果需要，例如 sqlite3）; 5.release

特别提示，可能是 eclipse gnuarm 插件的 bug，clean 命令有时候会执行不成功，可靠的方法，是在 eclipse 环境下点击“右键点击 libOS——delete”删掉 libOS 目录，然后在工程名上点右键，选择

clean。

## 8.1. 重编译库

如果修改了编译和链接选项或命令行参数，须把相应的库也改成相同的配置并重新编译。

库是二进制形式存在的可执行代码，是按照一定的编译规则编译 c 代码得到的。有许多编译规则是可以通过编译选项配置的，编译选项是执行编译命令时从命令行输入的，对于使用 IDE 的开发环境，也有部分编译规则是在 IDE 中用图形化的方式配置。

既然可执行程序是用库和用户程序文件“组装”而成的，那编译库和编译用户程序文件的“编译规则”必须是一致的，一个道路上开车，交通规则总该一致才好吧。

最安全的做法是，只要修改了编译规则，无论是直接配命令行参数，还是在 IDE 中勾选，都重新编译所有库，笨办法是最可靠的。反正编译选项也不会经常变，推荐大家用笨办法。

直接导入的 example 工程，已经做好了正确设置，直接使用即可。

用户工程和库必须保持一致的编译选项一般是（根据 CPU 不同，可能会有更多）：

- 1、char 是有符号还是无符号。
- 2、CPU 家族和架构选择。
- 3、软浮点还是硬浮点，以及浮点格式和浮点精度等相关的选项。
- 4、跟指令集选择相关的选项，例如 arm 或 thumb。
- 5、跟存储器对齐相关的选项。
- 6、存储器大小端相关选项。
- 7、有全局影响的宏定义，例如 NODEBUG 宏。

可以不同的选项：

- 1、优化级别。
- 2、警告输出相关选项。
- 3、调试信息相关选项。

还有其他很多类型选项，具体分析吧，不放心就重新编译全部库。

## 9. 调试配置

Eclipse 中包含调试功能，结合 jlink 提供的 jlink\_gdbserver，调试功能还是很强大的，配置方法如下：

1. 安装 4.90 以上版本的 jlink 工具。

2. 在 eclipse 界面下，在“Project Explorer”框鼠标点击你要调试的工程名，然后点击小虫子旁边的倒三角形，会列出你曾经执行过的配置，需要修改配置，或者增加配置的话，可选择“Debug configurations”，

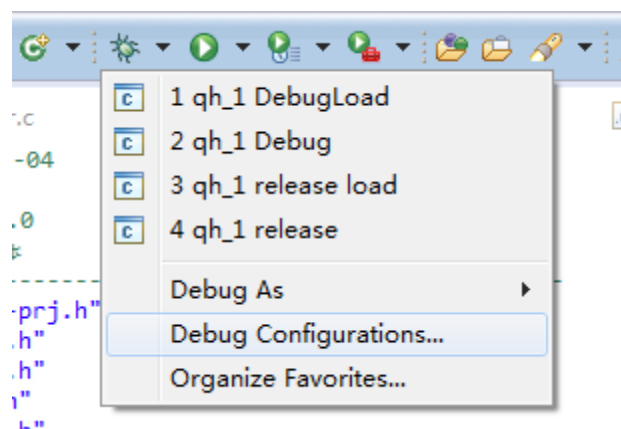


图 9-1

3. 在弹出对话框中，双击“GDB SEGGER J-Link Debugging”。或者右键点击相似的配置，选择“Duplicate”，将创建新配置。

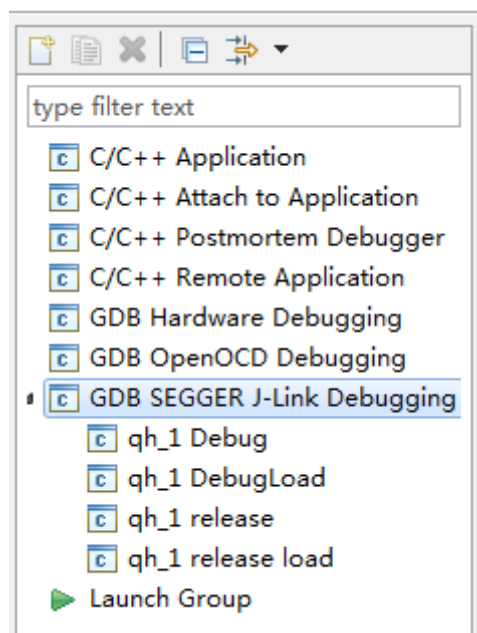


图 9-2

4. 在 main 选项卡中，在 Name 框填入你的配置名，在 c/c++ Application 框填要调试的文件。

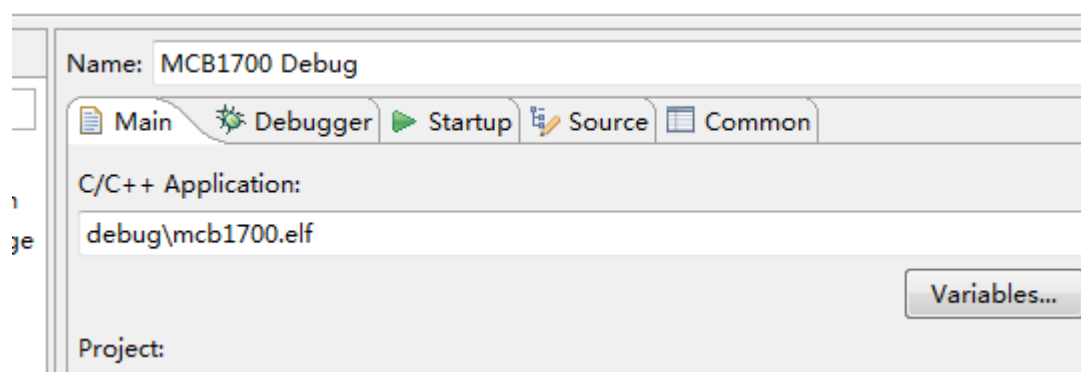


图 9-3



5. 在 debugger 选项卡中,除不用勾选 **Allocate console for semihosting and SWO** 外,其他默认即可。点击 Variables, 把 jlink\_path 变量的值改为你在 step1 安装的 jlink 可执行文件目录。器件名可点击 “Supported device names” 从 segger 公司网站选择。

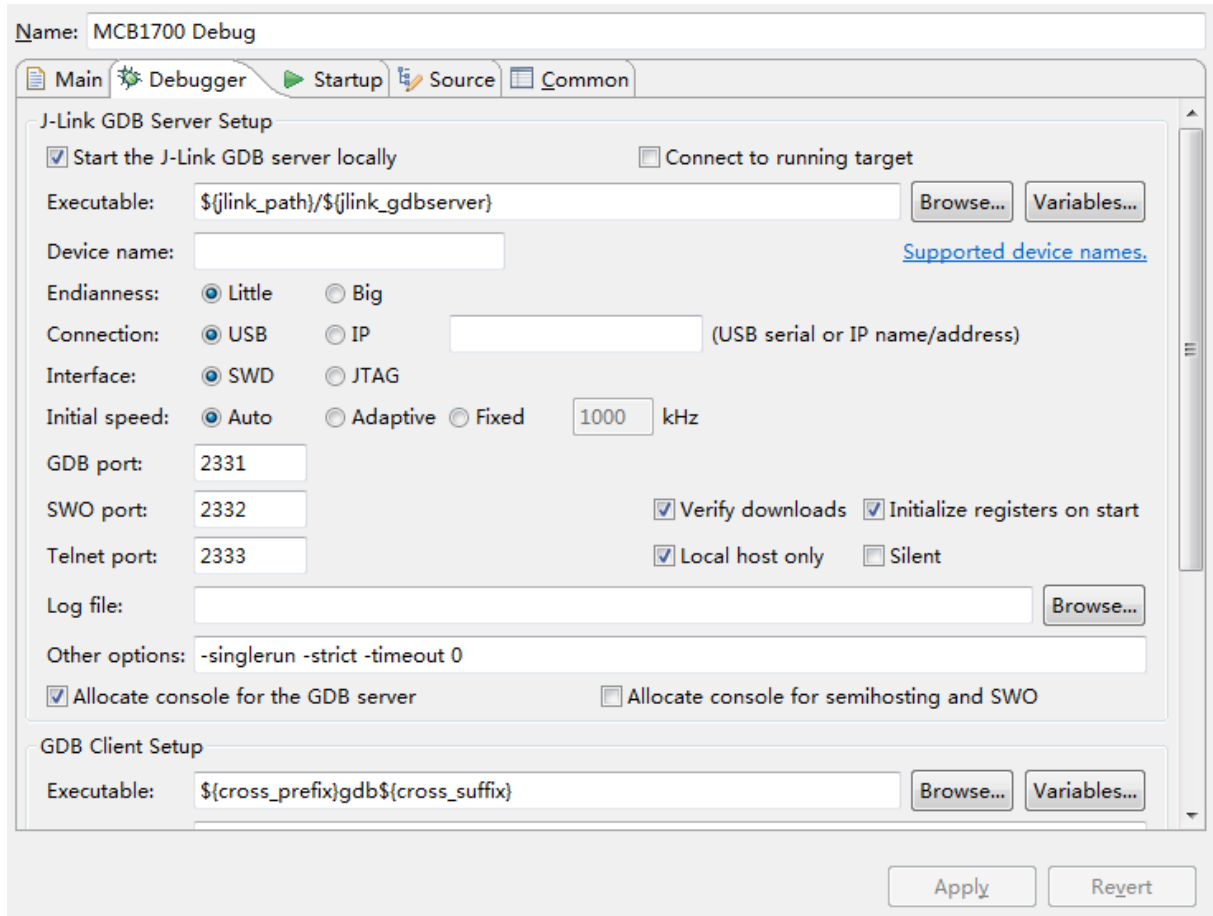
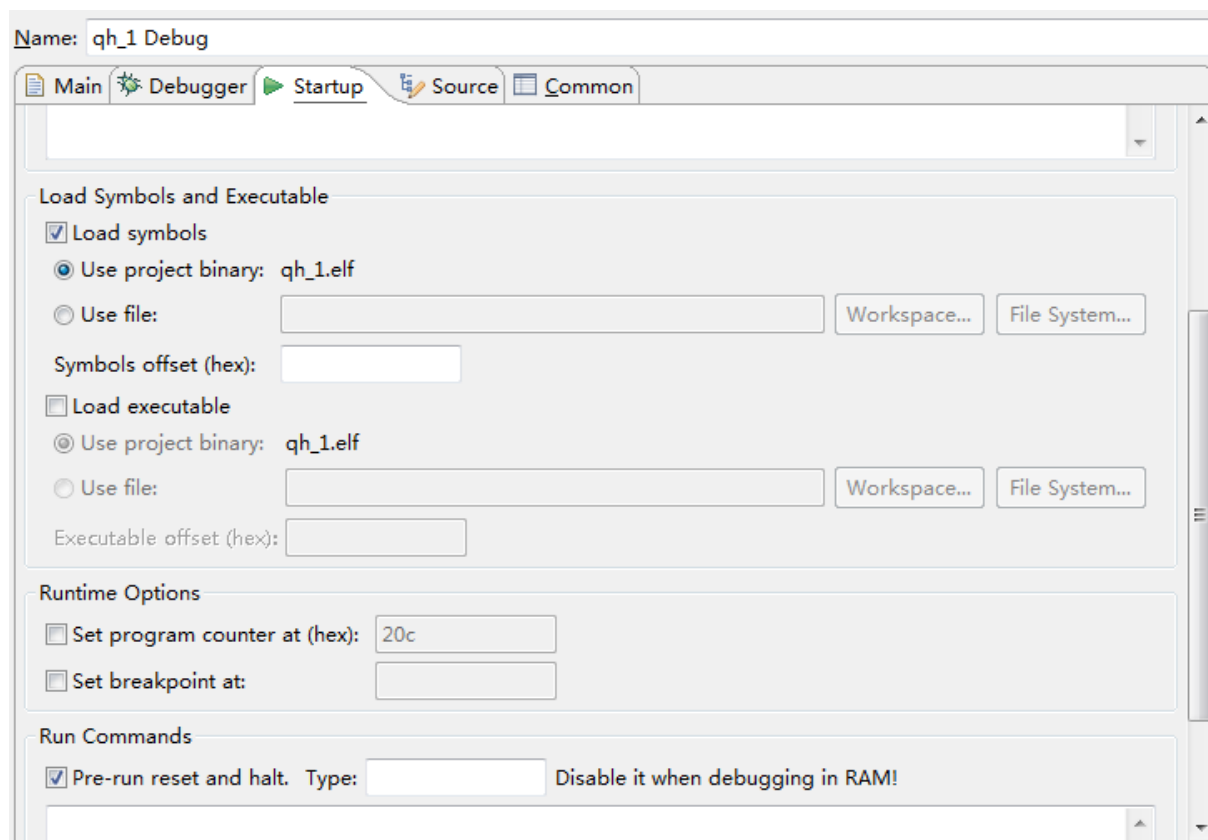


图 9-4

6. 在 Startup 选项卡中,无须勾选: **Enable semihosting.** **Set breakpoint at:** **Continue** , 其他按照默认设置即可。其中 **Load executable** 框需要特别注意,勾上的话,调试前就会烧写代码到 flash, 适合于修改了代码后首次调试。如果没有修改代码,重新开始调试的话,就不要勾。



7. 在次点击图 9-1，点击你刚才添加的配置，即可调试。