

ARM® Cortex®-M 32-bit 微控制器

NuEclipse 用户手册

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

内容目录

1	序言	5
2	系统需求和安装使用说明	6
2.1	系统需求	6
2.2	支持芯片	7
2.3	安装	7
2.3.1	执行 NuEclipse 安装软件在微软 Windows 上	7
2.3.2	解压 NuEclipse Tar 文件在 GNU/Linux 上	8
2.3.3	检验 Eclipse Preferences	9
2.4	执行 Eclipse	11
3	工程开发教学	12
3.1	选择 Workspace	12
3.2	新工程向导	13
3.3	汇入现有工程	15
3.4	建置设定	16
3.5	调试配置	17
3.5.1	Debugger 标签页	18
3.5.2	Startup 标签页	19
3.6	调试视图	20
3.6.1	寄存器视图	20
3.6.2	内存视图	21
3.6.3	反汇编视图	22
3.6.4	外围寄存器视图	23
3.7	检测点	29
3.8	在 RAM 中调试	32
4	Q&A	36
5	修订历史	40

图片目录

图 2-1 NuEclipse 安装向导	7
图 2-2 Install.sh 脚本	8
图 2-3 Preferences 设置 Global Tools Paths 的对话框	9
图 2-4 Preferences 设置 OpenOCD Nu-Link 的对话框	10
图 2-5 Eclipse.exe 和相关文件夹	11
图 3-1 选择 Workspace	12
图 3-2 新工程向导	13
图 3-3 目标处理器的设定	14
图 3-4 汇入工程	15
图 3-5 建置设定	16
图 3-6 调试配置	17
图 3-7 设置 Debugger 标签页	18
图 3-8 设置 Startup 标签页	19
图 3-9 寄存器视图	20
图 3-10 内存视图	21
图 3-11 点选 Instruction Stepping Mode 按键	22
图 3-12 反汇编视图	22
图 3-13 开启 Packs 观点	23
图 3-14 如何下载套件	24
图 3-15 数据库位置	25
图 3-16 安装新唐 SFR 文件	26
图 3-17 选择设备	27
图 3-18 外围寄存器视图	28
图 3-19 切换检测点	29
图 3-20 C/C++ 检测点属性对话框	30
图 3-21 在断点视图中新增的检测点	31
图 3-22 内存配置	32
图 3-23 修改 Mem.ld 脚本	33
图 3-24 设定调试配置	34

图 3-25 在 RAM 中调试 35

图 4-1 新增 Udev 规则 37

图 4-2 Preferences 设置 String Substitution 的对话框 38

1 序言

NuEclipse 是个跨平台 ARM 嵌入式系统的软件开发环境。它包含一系列的 **Eclipse** 插件和工具。该插件可让使用者创建、建置和调试基于 ARM 且在 **Eclipse** 开发环境下的工程。它拥有下面这些特色。

- **藉由新工程向导来创建工程:** 新工程向导为不同的目标芯片提供一些模板工程。
- **藉由 GNU ARM 工具键来建置工程:** 该工具键包含 ARM 的 GCC 编译程序。使用者可以不受限制地利用它来建置工程。
- **藉由 GDB 来调试工程:** 使用者可暂停、单步、运行和监视目标芯片。存取内存及闪存是允许的。支持设置硬件断点及检测点。此外，用户可以抹除目标芯片和编程用户配置。

通过 **NuEclipse**，使用者可在 **Eclipse** 开发环境下开发新唐微控制器的工程。

2 系统需求和安装使用说明

2.1 系统需求

若用户想要在他们的计算机上执行 **NuEclipse**，这里列出系统需求。

	最低需求	建议规格
操作系统	Windows®7 且更新至最新套件或 GNU/Linux	Windows®10 且更新至最新套件或 Ubuntu 16.04 LTS
GNU ARM Embedded 工具键	6-2016-q4-major	目前最新版本
Java	JDK 1.7 or JRE 1.7	目前最新版本的 JDK
Eclipse IDE	Eclipse 4.4 Luna SR2	Eclipse 4.5 Mars SR2
Eclipse CDT	Eclipse CDT 8.6	目前最新版本的 Eclipse CDT

注意:为了在 Linux 上得到完整可用且舒适的使用环境，推荐 Linux 发行版为 Ubuntu 16.04 LTS (64-bit)。

2.2 支持芯片

请参照 user manual 文件夹下的 **Supported_chips.htm**。

2.3 安装

为安装 **NuEclipse**，请根据你的操作系统执行下面步骤：

1. 执行 NuEclipse 安装软件在微软 Windows 上。
2. 解压 NuEclipse tar 文件在 GNU/Linux 上。

2.3.1 执行 NuEclipse 安装软件在微软 Windows 上

在 Windows 上，只要执行 NuEclipse 安装软件便能很轻松地安装好 NuEclipse。安装过程中，向导会问使用者是否要安装 **GNU ARM Eclipse Windows Build Tools** 和 **GNU ARM Embedded Toolchain**。对 NuEclipse 而言，它们的存在是必需的。

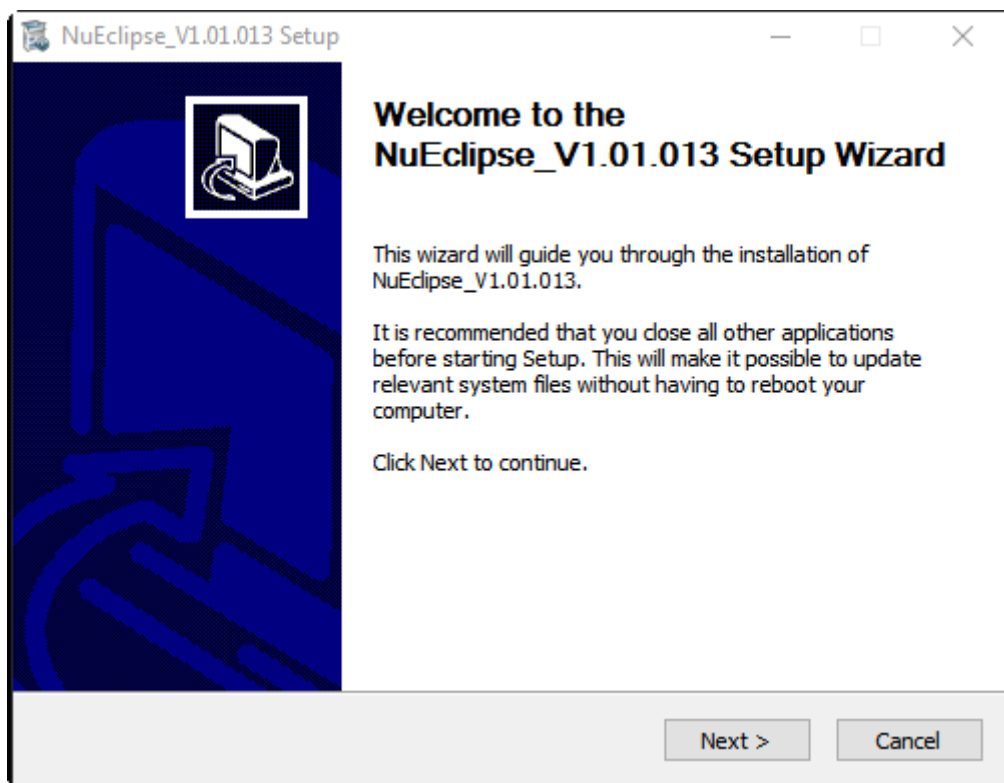


图 2-1 NuEclipse 安装向导

2.3.2 解压 NuEclipse Tar 文件在 GNU/Linux 上

在 GNU/Linux 上，只要解压 NuEclipse tar 文件便能很轻松地安装好 NuEclipse。随后，执行 install.sh 脚本以完成安装程序。请不要使用 **sudo** 命令来执行脚本。

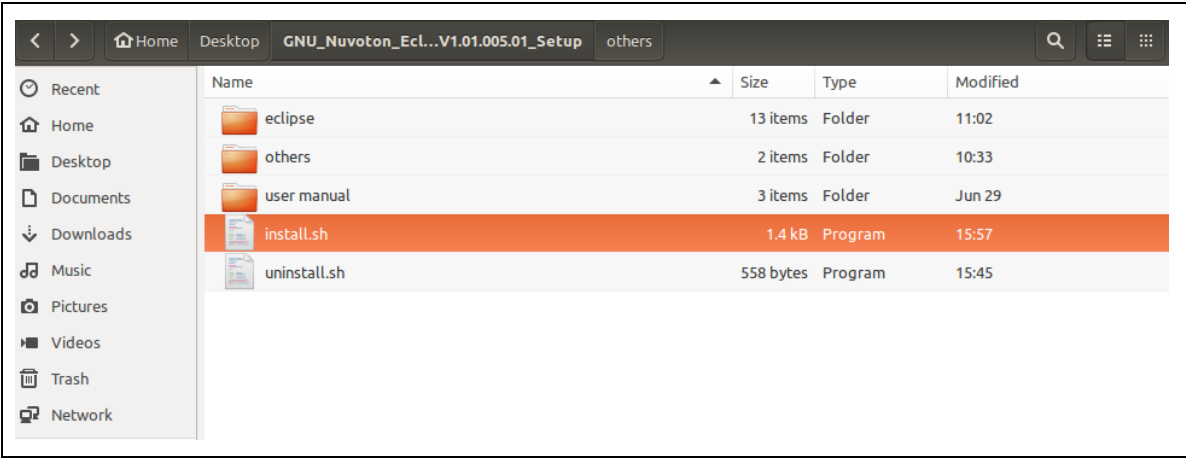


图 2-2 Install.sh 脚本

2.3.3 检验 Eclipse Preferences

在 Windows 上安装结束后, Eclipse preferences 会自动写入对应的值。为检验它们, 单击 **Window > Preferences**。Preferences 对话框会出现。前往 **C/C++ > Build > Global Tools Paths** 并确认 Build tools 和 Toolchain 的文件夹路径是否正确地指向先前安装软件所装的位置。确认完毕, 点选 **Apply** 按键来生效。在 GNU/Linux 上, Build tools 活页夹路径并不需要。该路径应为空白。

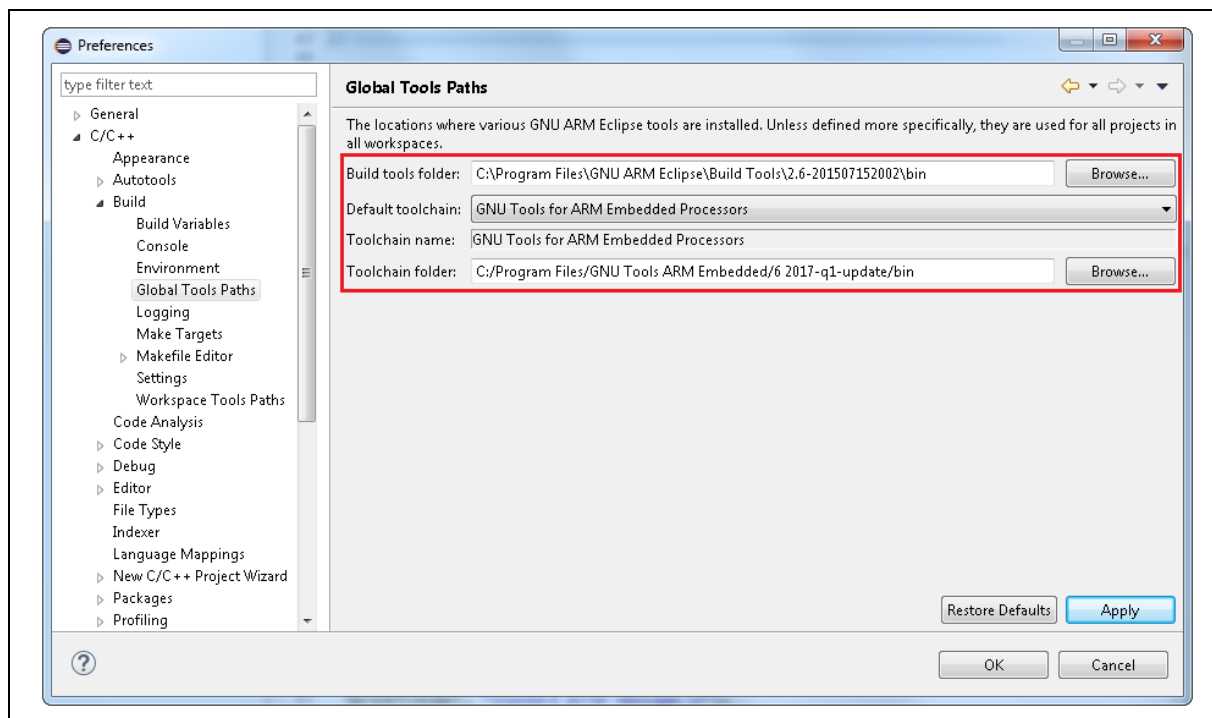


图 2-3 Preferences 设置 Global Tools Paths 的对话框

接下来，前往 **Run/Debug > OpenOCD Nu-Link** 并确认 OpenOCD 文件夹路径是否正确地指向安装软件所存放 OpenOCD 执行文件。例如，在微软 Windows 上，OpenOCD 活页夹的路径可能是 C:/Program Files/Nuvoton Tools/OpenOCD。同样地，在 GNU/Linux 上它则可能是 /usr/local/OpenOCD。该 OpenOCD 执行档是由新唐提供且为 Nu-Link 客制化。若使用者用别的 OpenOCD 执行档，OpenOCD 和 Nu-Link 可能不能正常一起工作。确认完毕，点选 **Apply** 按键来生效。

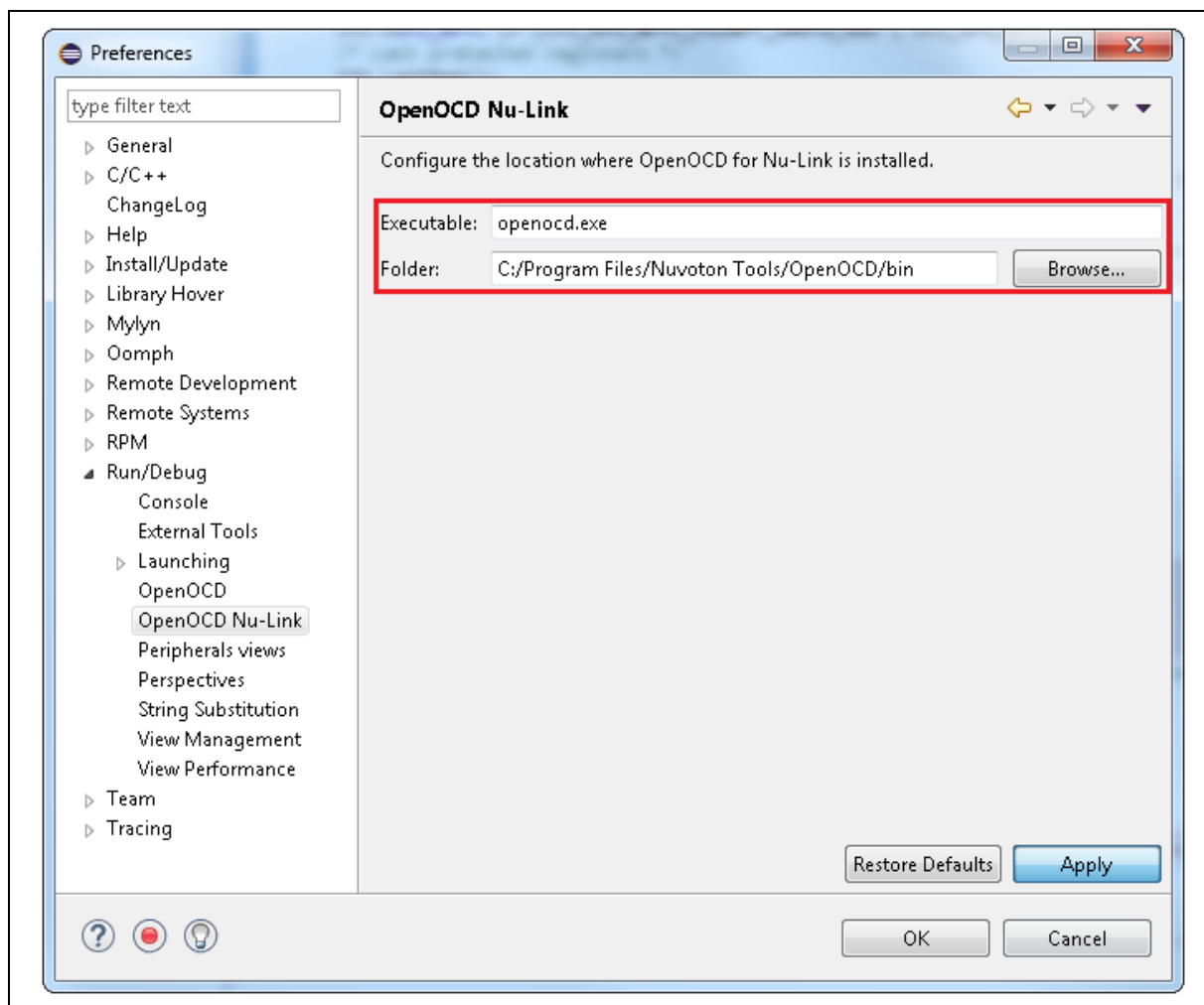


图 2-4 Preferences 设置 OpenOCD Nu-Link 的对话框

2.4 执行 Eclipse

为执行 **NuEclipse**，双击 **Eclipse.exe**。请注意执行档和相关的资料夹(例如 OpenOCD 资料夹)应该待在同一目录底下。否则，软件将不能正常运作。

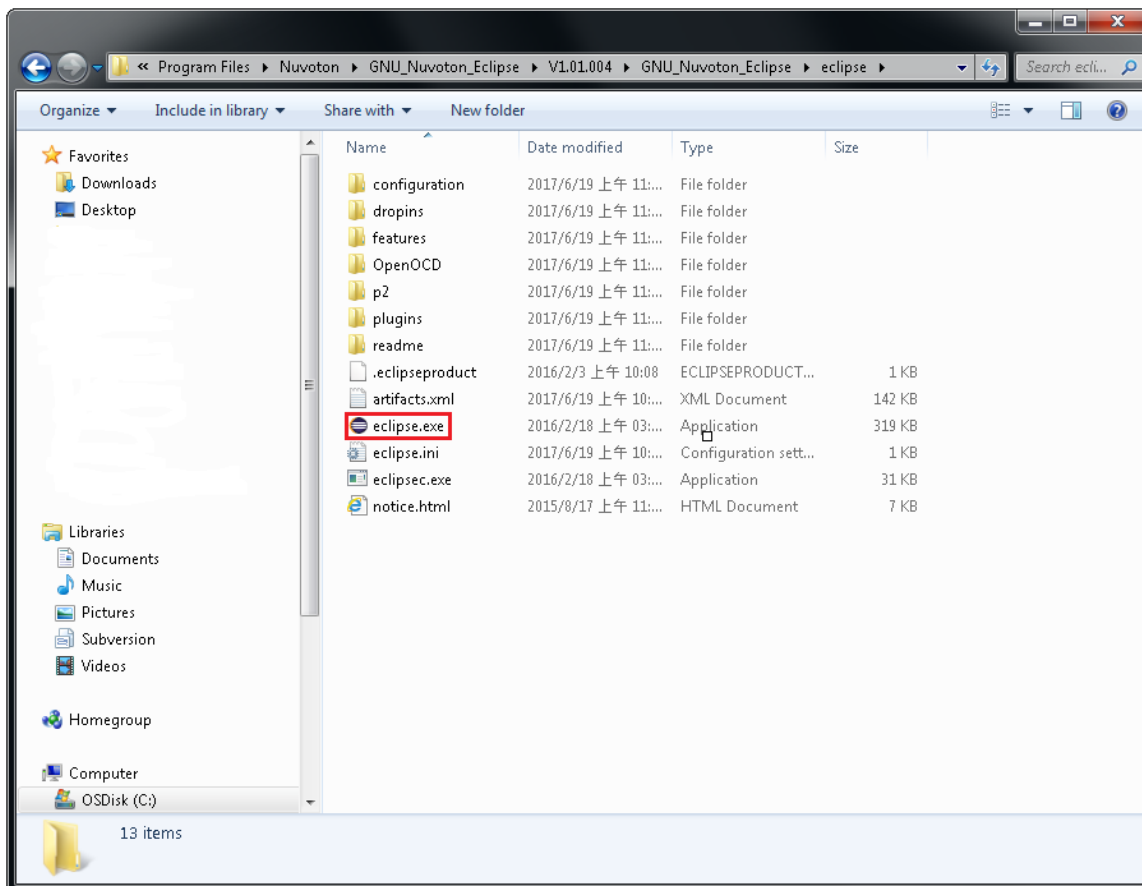


图 2-5 Eclipse.exe 和相关文件夹

3 工程开发教学

3.1 选择 Workspace

当Eclipse启动时，我们必须选择一个Workspace，其包含相关工程并构成一个应用程序。此外，Eclipse和工程们的一些配置设置也存储在这里。对于不同的计算机，配置设置可能会有所不同。我们应该创建自己的Workspace，而不是复制其他用户的Workspace。一次只能有一个工作空间处于活动状态。若要切换Workspace，点选**File->Switch Workspace**。

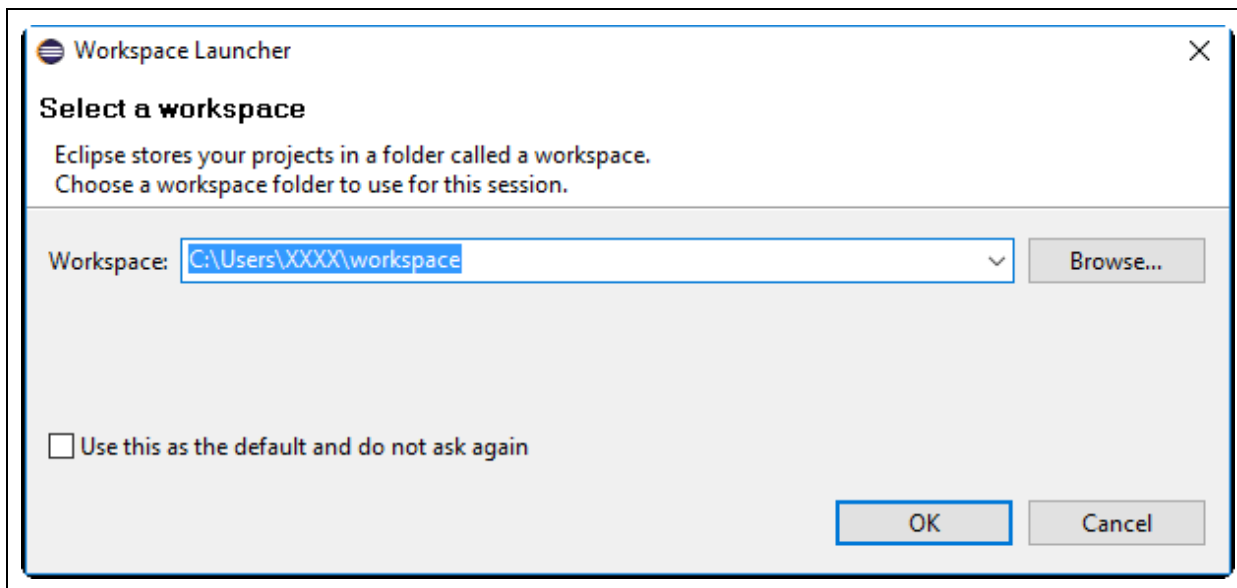


图 3-1 选择 Workspace

3.2 新工程向导

作为一个新手，快速创建 C/C++工程的方式为使用新工程向导。以创建 C 工程为例，点选 **File > New > C Project**。新工程向导的对话框会出现。在对话框内，点选 **Hello World Nuvoton Cortex-M C Project**，然后输入工程名称，最后点选 **Next >** 按键以继续下一步。

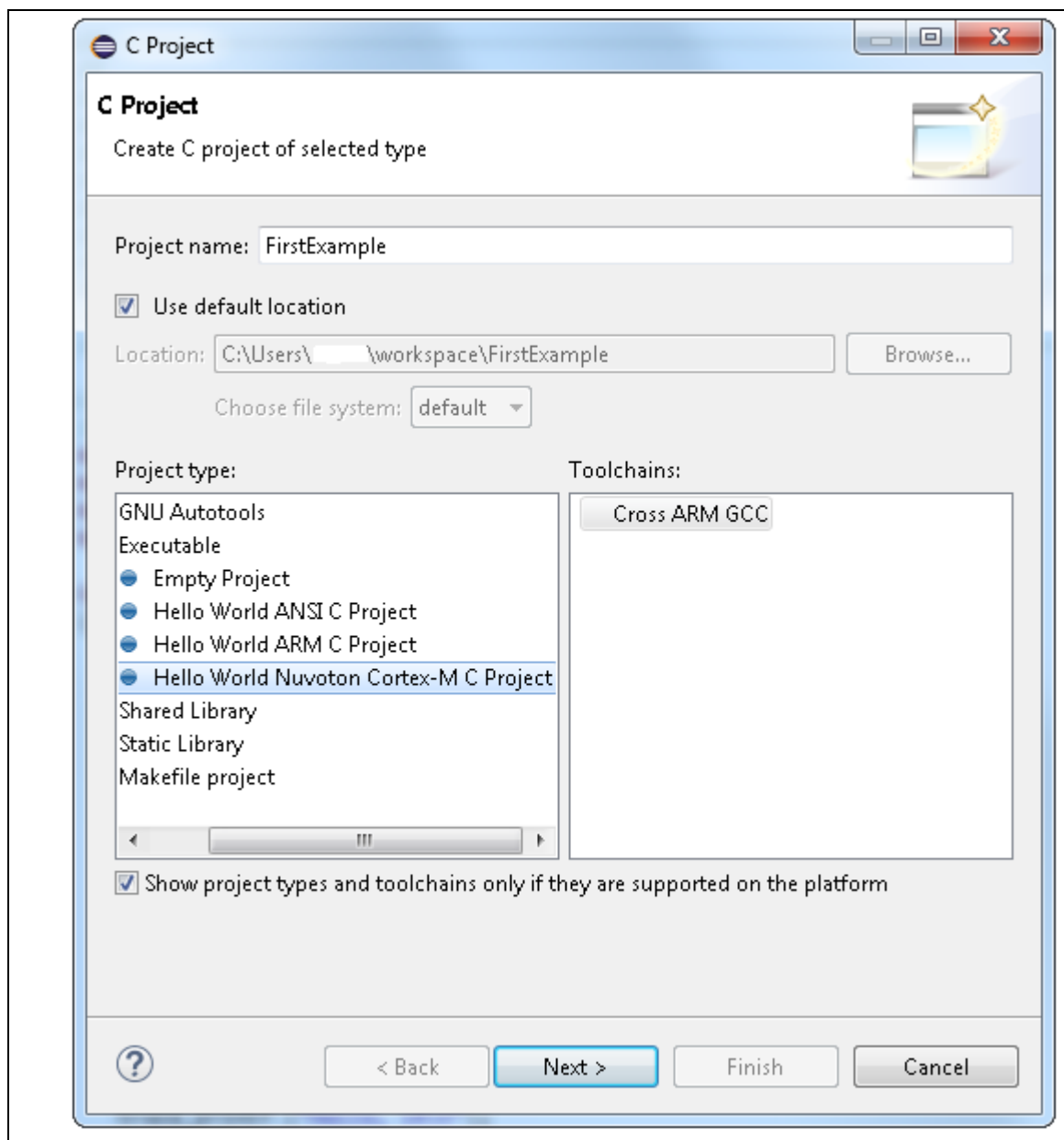


图 3-2 新工程向导

依据实际的目标芯片，我们选择相对应的芯片系列。对于某些芯片，例如 M2351_NonSecure，我们需要输入额外的库路径。否则，构置过程可能会失败。此外，输入实际闪存和内存的值。否则，默认值将会被采用。当所有设定都完成后，点选 **Next >** 按键直到点选 **Finish** 按键为止。

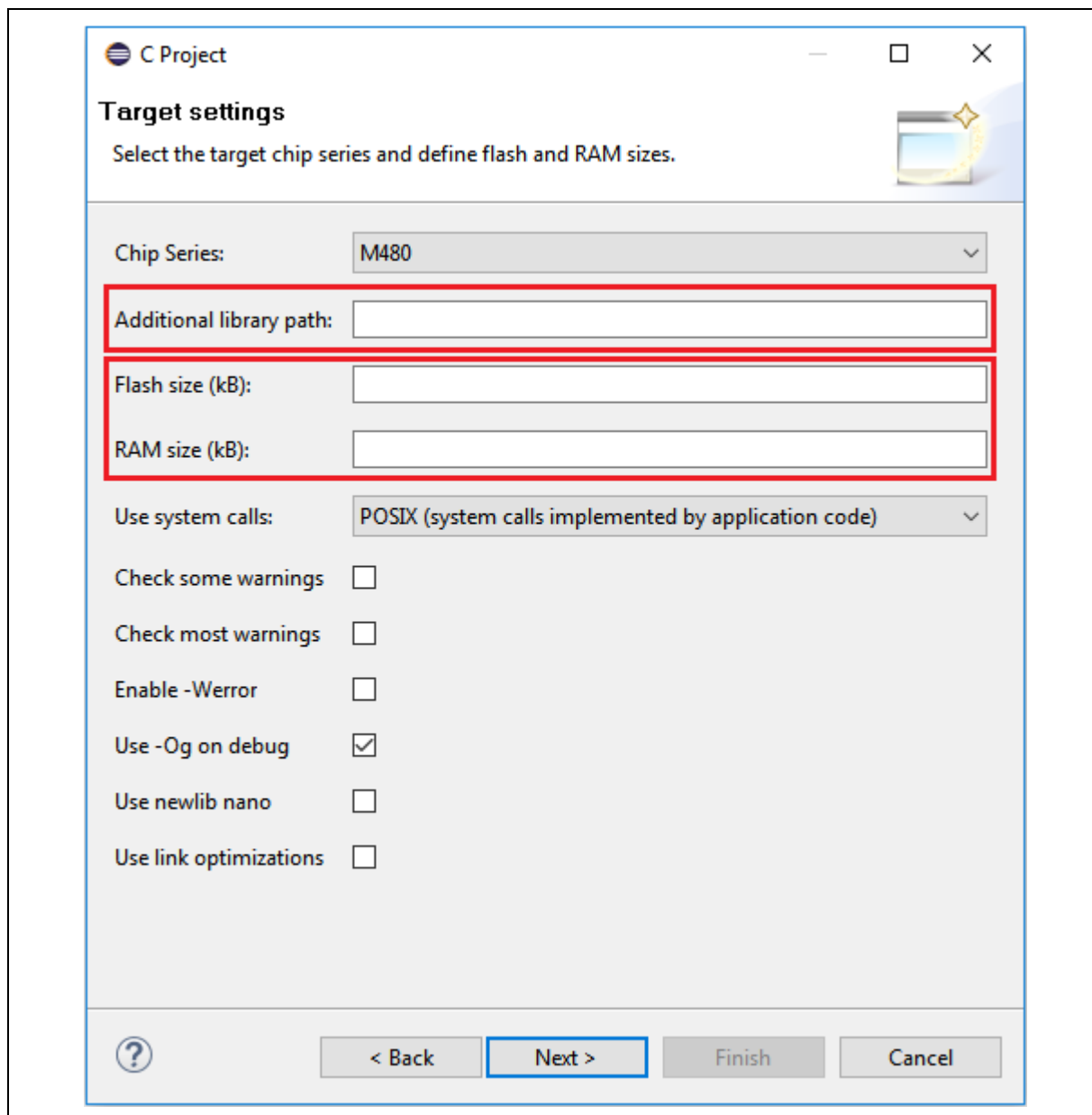


图 3-3 目标处理器的设定

3.3 汇入现有工程

当 BSP 可用时，我们可以使用以下步骤将它们汇入到 Workspace 中：

1. 从 menu 中，选择 **File > Import**。汇入向导出现。
2. 选择 **General > Existing Project into Workspace** 并点选 **Next**。
3. 选择 **Select root directory** 或 **Select archive file**，然后点选 **Browse** 找到包含工程的目录或文件。在 Nuvoton BSP 中，Eclipse 工程放在 GCC 文件夹中（请参阅下图）。
4. 在 **Projects** 下选择您想要汇入的工程，然后点选 **Finish**。

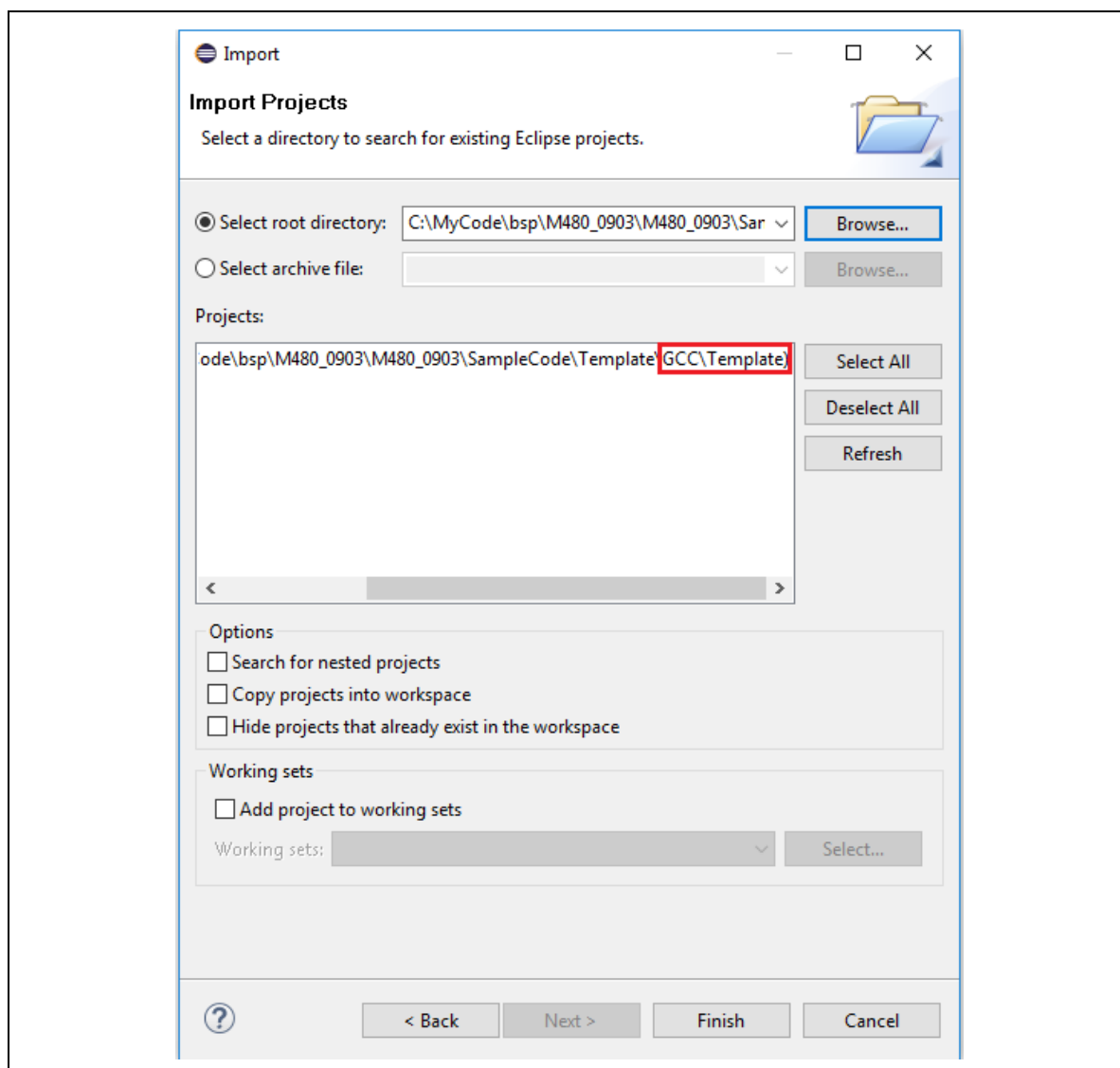


图 3-4 汇入工程

3.4 建置设定

工程创建完成后，我们仍有机会来更改建置设定，只要点选 **Project > Properties** 即可。当 **Properties** 对话框出现，前往 **C/C++ Build > Settings**。在那里，我们可以根据现况来更改建置设定。点选 **Apply** 按键来生效。设定完成后，我们应该能够顺利地建置工程了。

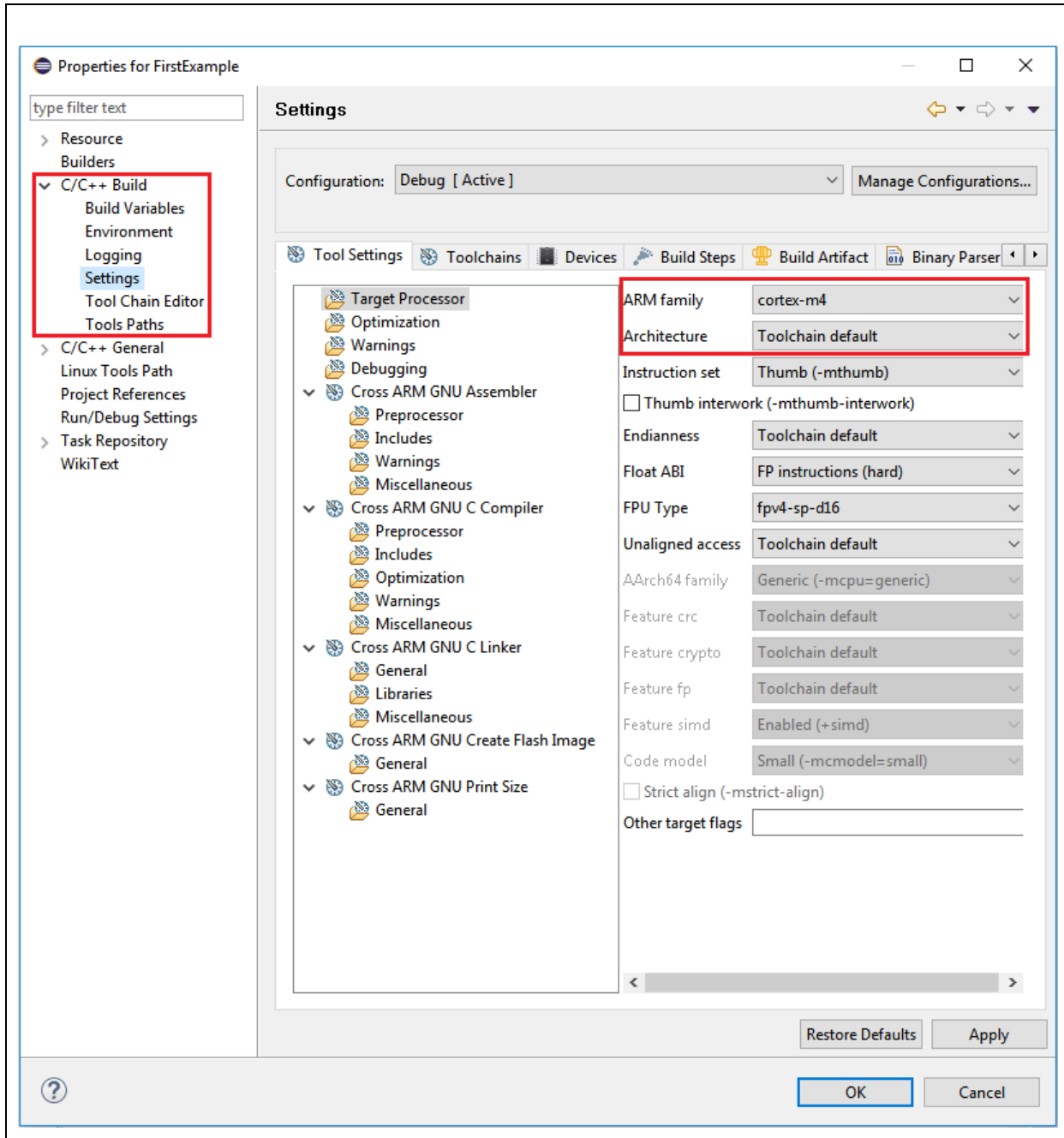


图 3-5 建置设定

3.5 调试配置

在启动应用程序进入调试模式前，我们必须先准备好调试配置，其包含所有关于调试所需的信息。点选 **Run > Debug Configuration...** 来呼叫调试配置对话框。双击 **GDB Nuvoton Nu-Link Debugging** 群组。新唐 Nu-Link 调试配置会出现在右边边在 Main 标签页里，Project 名称应与工程名称一致。C/C++ 应用程序应指向建置产生出来的 .elf 应用程序。倘若工程名称或 C/C++ 应用程序不正确，请至工程视图选取想要调试的工程并建置它以产生执行档，然后展开树以确保生成的可执行文件存在。然后再重复先前的操作一遍。

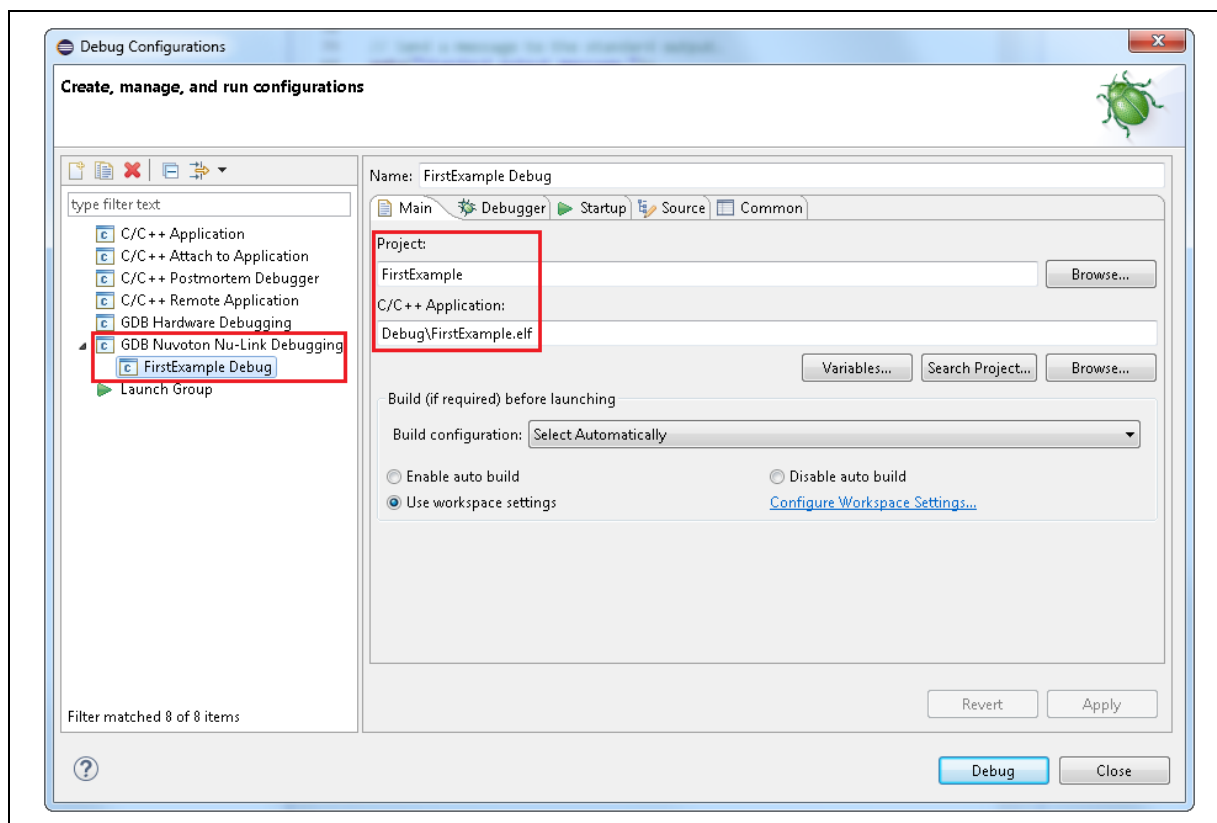


图 3-6 调试配置

3.5.1 Debugger 标签页

Debugger 标签页用来提供 OpenOCD 和 GDB 客户设置。OpenOCD 需要正确的配置文件才能知道如何和适配器及目标芯片工作。配置文件定义在 **Config options** 字段。新唐适配器为 Nu-Link，其接口配置文件为 **nulink.cfg**。新唐有三个不同的 ARM 家族，即 M0、M4 和 M23。相对应的目标配置文件为 **numicroM0.cfg**、**numicroM4.cfg** 和 **numicroM23.cfg**。就 M23 二次开发而言，目标配置文件会是 **numicroM23_NS.cfg**。

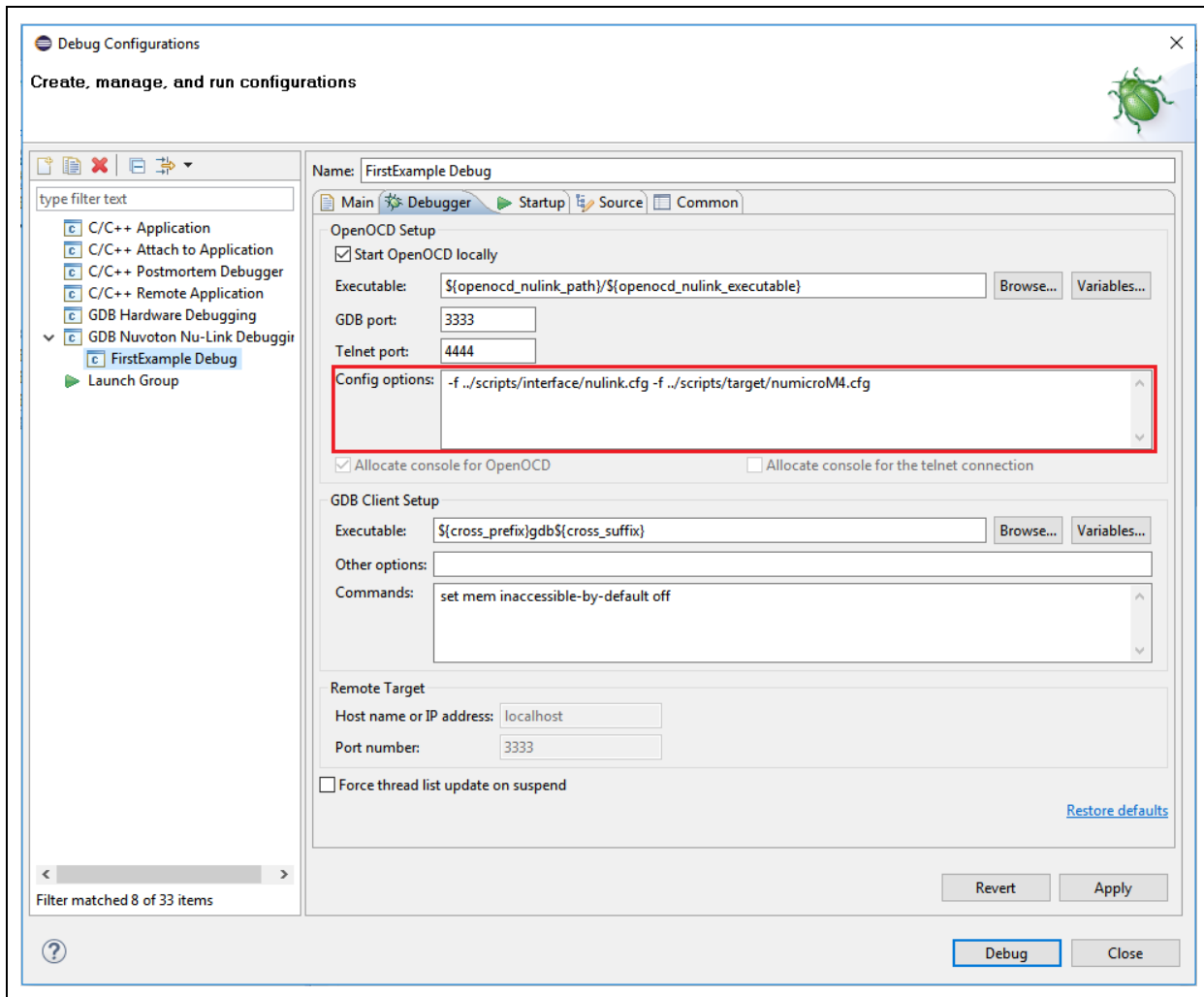


图 3-7 设置 Debugger 标签页

3.5.2 Startup 标签页

作为开发的第一步，我们应到 Startup 标签页选择对的 **Chip Series**。选完，相对应的目标配置文件将会自动地写入 Debugger 标签页的 **Config options** 字段。为加载执行档至闪存，我们需勾选 **Load executable to flash** 复选框。为加载执行档至内存，我们需勾选 **Load executable to SRAM** 复选框。当所有的设定完成，点选 **Apply** 按键来生效。为启动应用程序进入调试模式，请点选 **Debug** 按键。

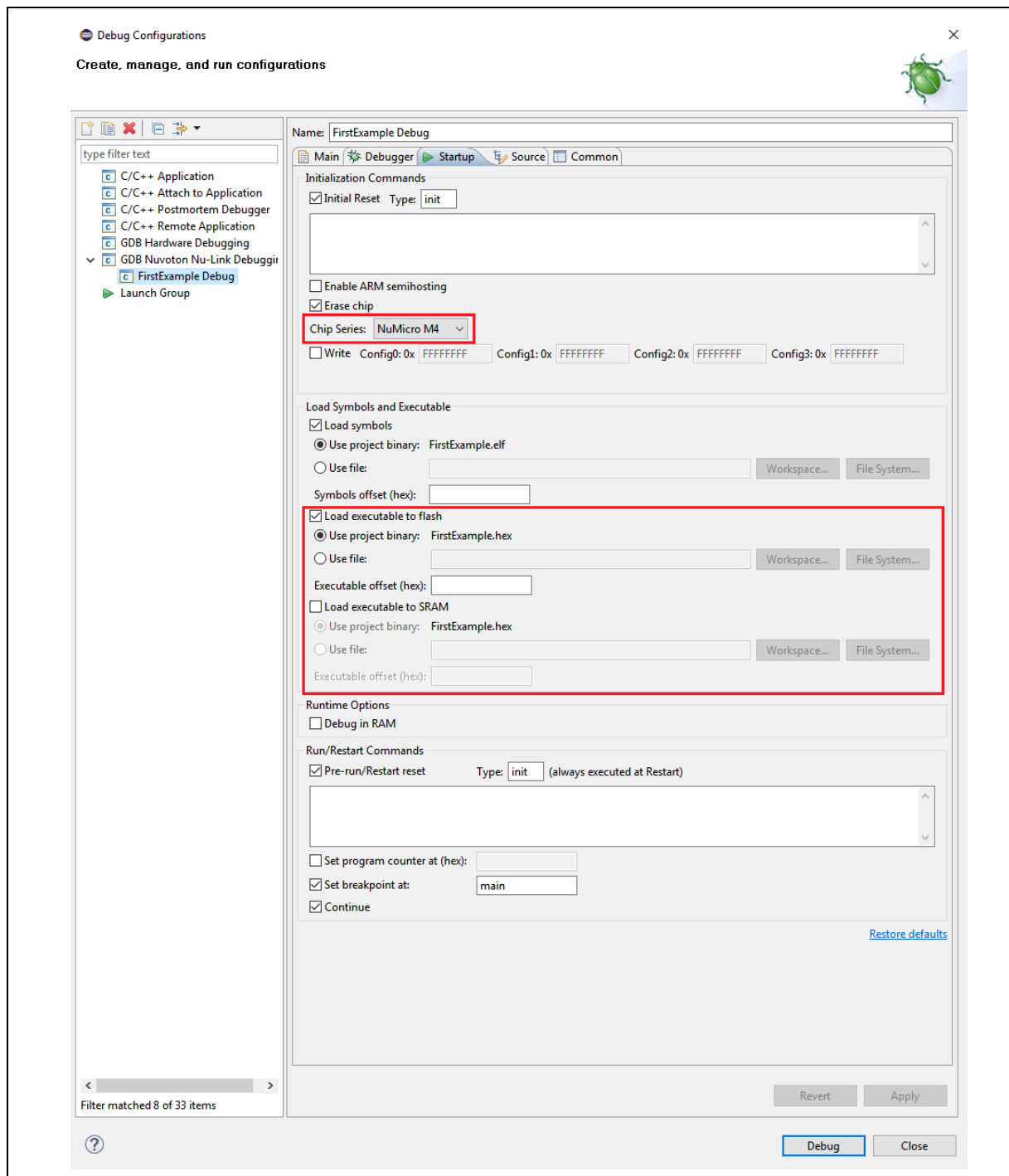


图 3-8 设置 Startup 标签页

3.6 调试视图

Eclipse 提供多个调试视图。当中每一个都有特定的调试信息。

3.6.1 寄存器视图

当进入调试模式时，我们可以打开位于调试观点中右上角的**寄存器视图**。寄存器视图显示目前栈上的寄存器信息。

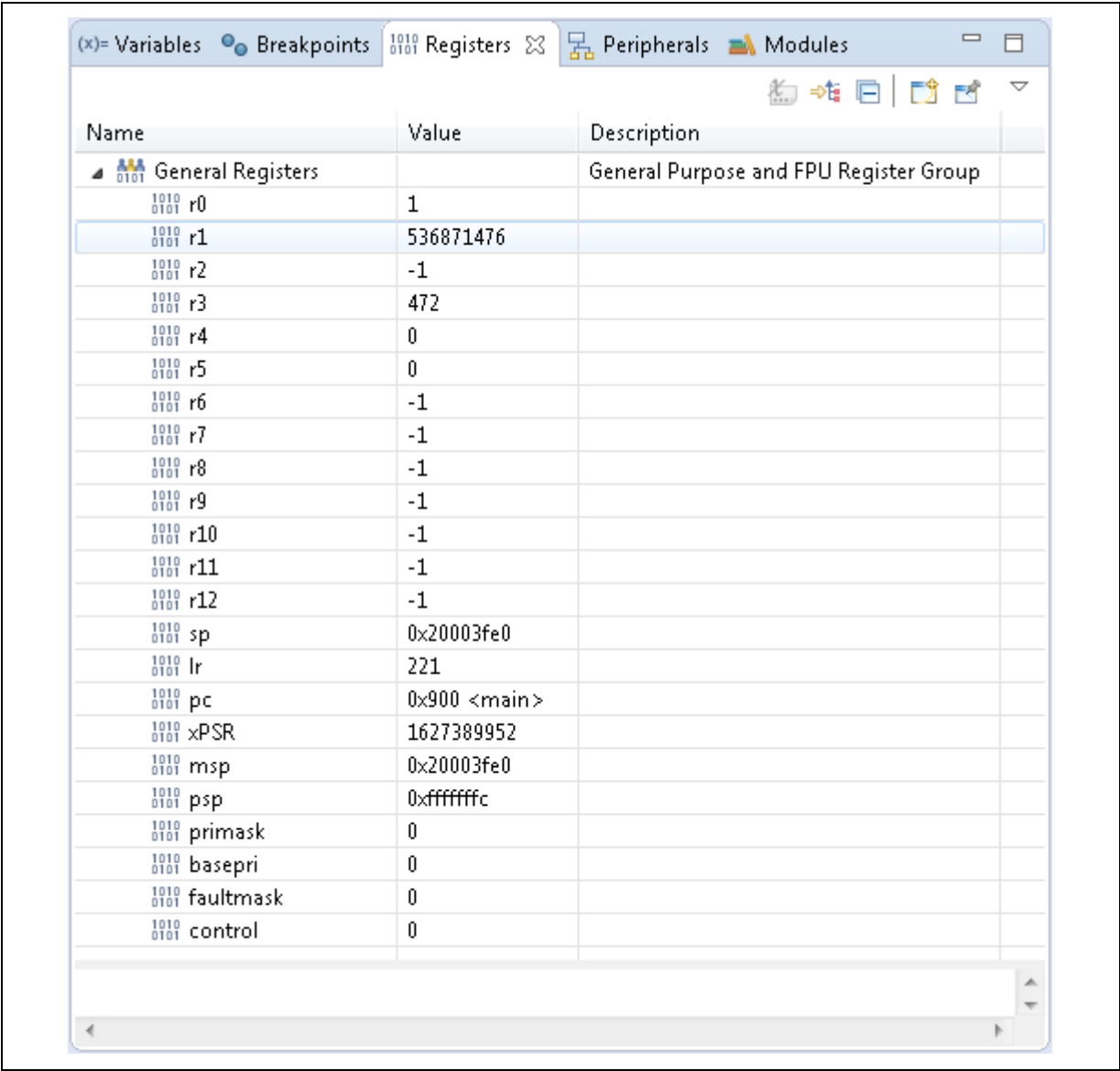


图 3-9 寄存器视图

3.6.2 内存视图

调试观点中的**内存视图**可用作监视和修改进程内存。进程内存被表示成一系列的**内存监测器**。每一个监测器代表一小段内存，由它的**初始地址**来描述。为打开视图，点选调试观点下半部的内存标签页。

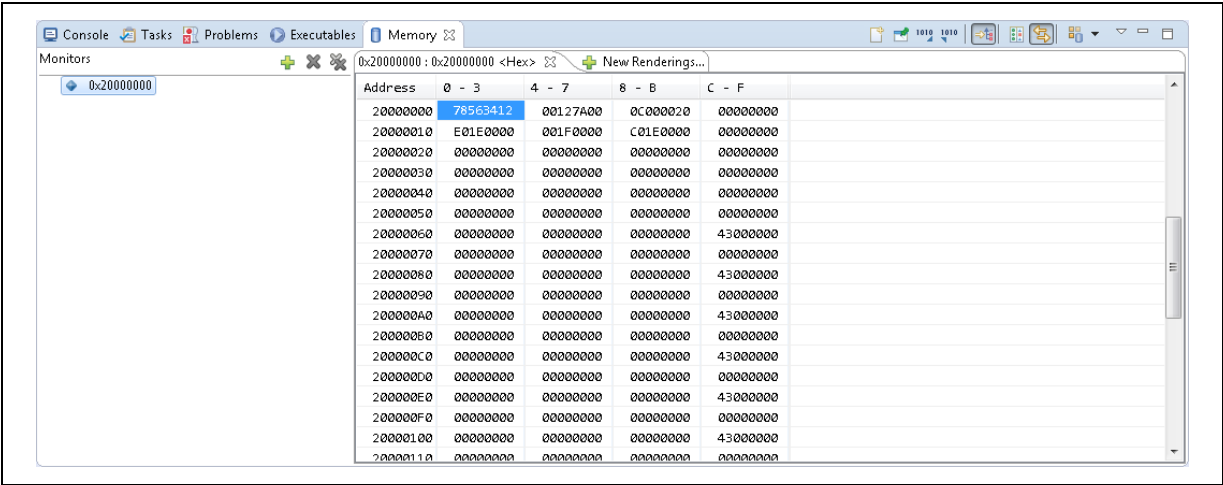


图 3-10 内存视图

3.6.3 反汇编视图

反汇编视图将加载程序显示为汇编指令并混合源代码作为比对。我们可在反汇编视图上作下列操作：

1. 设置断点在任何汇编指令的起点。
2. 启用或关闭断点。
3. 单步汇编指令。
4. 跳至特定的指令。

为打开视图，我们需在上方工具栏点选 **Instruction Stepping Mode** 按键，如下图。



图 3-11 点选 Instruction Stepping Mode 按键

然后反汇编视图会出现在右边窗口。

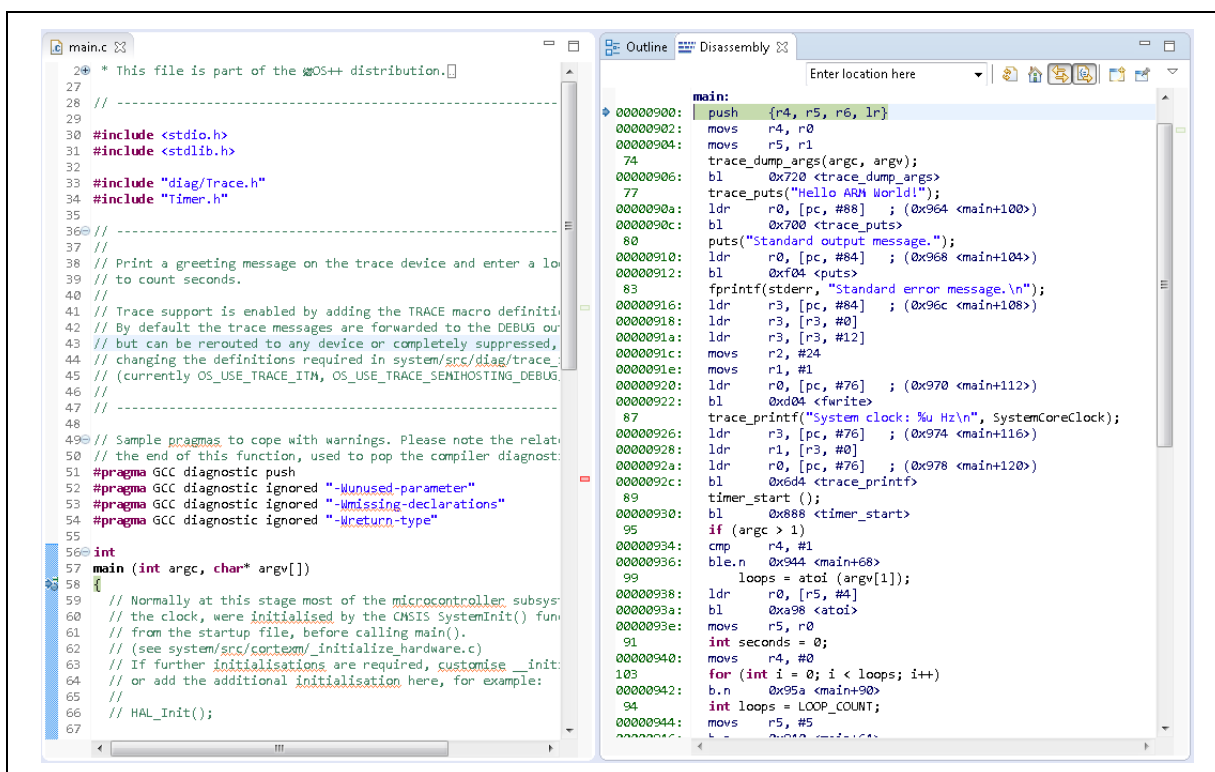


图 3-12 反汇编视图

3.6.4 外围寄存器视图

为显示外围寄存器视图，我们需使用 **Packs** 机制。该机制帮助用户从 Keil 数据库下载 SFR(特殊函数寄存器)文件。首先，我们从 **Open Perspective** 对话框中开启 **Packs** 观点。

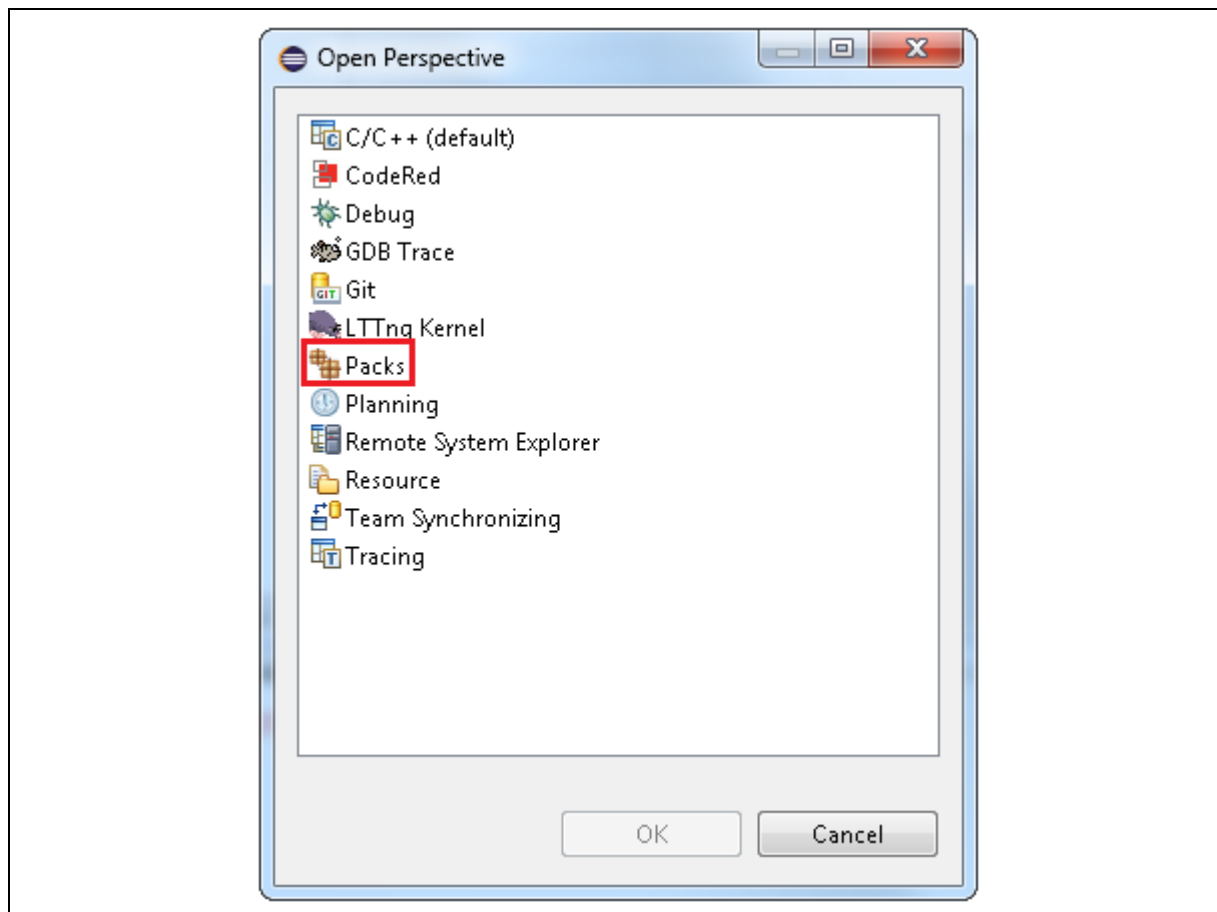


图 3-13 开启 Packs 观点

我们第一次看到 Packs 观点时，其内容是由 NuEclipse 安装软件所提供。倘若默认内容不见了，请切换到一个全新的 Workspace 然后再试一次。若想获得最新版本，点选右上角的 **Update the packages definitions from all repositories** 按键。之后，Eclipse 会从数据库开始下载所有的 SFR 文件。

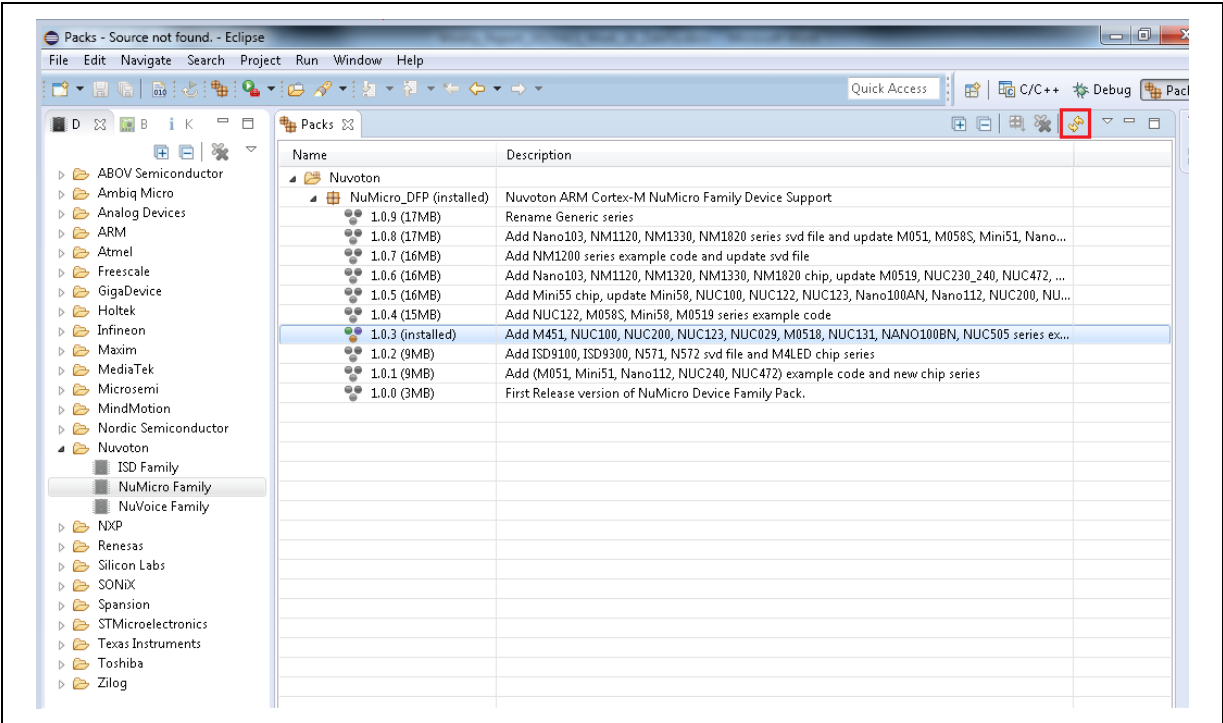


图 3-144 如何下载套件

数据库所在位置定义在 **Window > Preferences > C/C++ > Packages > Repositories**。初始库是来自 Keil's CMSIS pack。

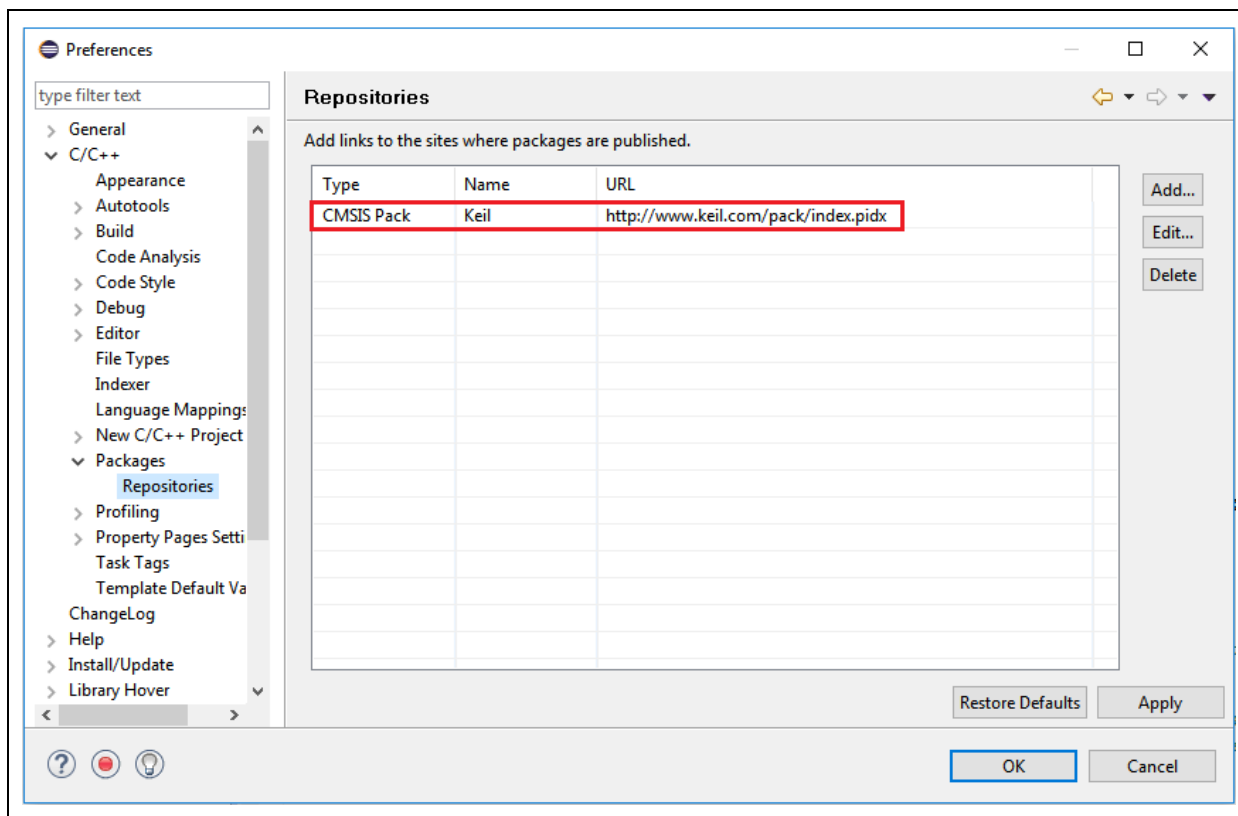


图 3-155 数据库位置

当下载完成时，我们可以找到新唐的 SFR 文件并在 Eclipse 安装它们，如果需要的话。

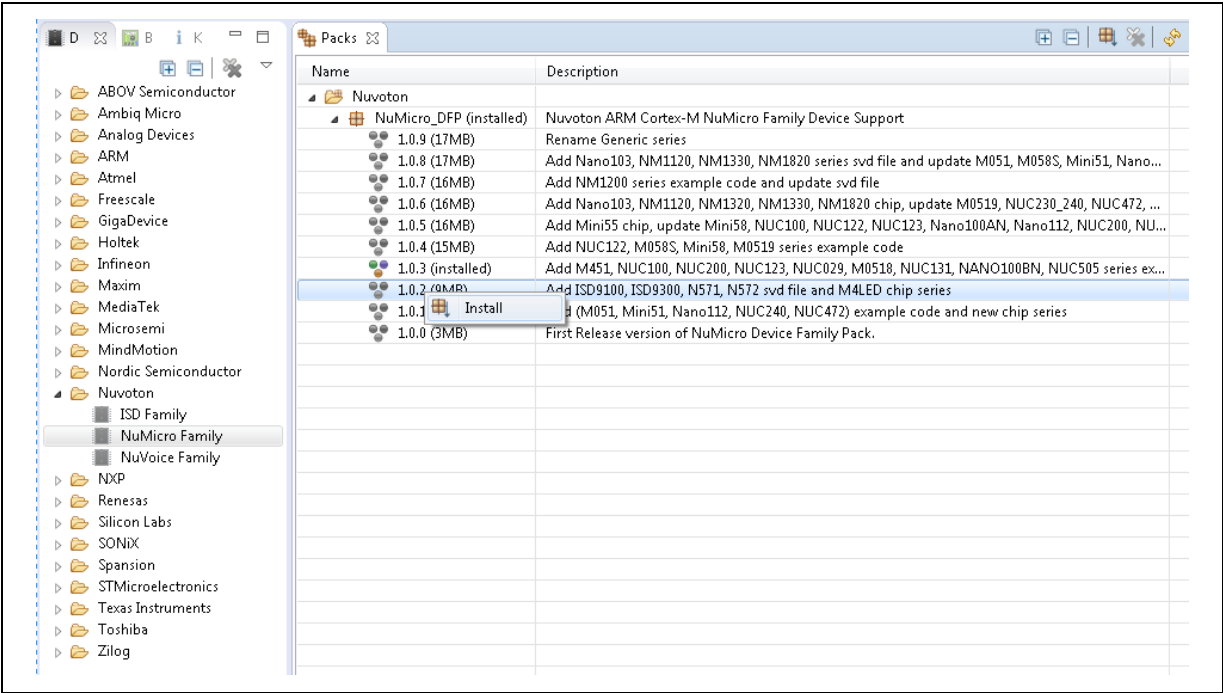


图 3-166 安装新唐 SFR 文件

为使用特定的 SFR 文件，打开工程的属性对话框，前往 **C/C++ Build > Settings**。从那里，我们应该选择特定的设备以符合现实的芯片。在这个范例里，设备是 M487JIDAE。点选 **Apply** 按键来生效。

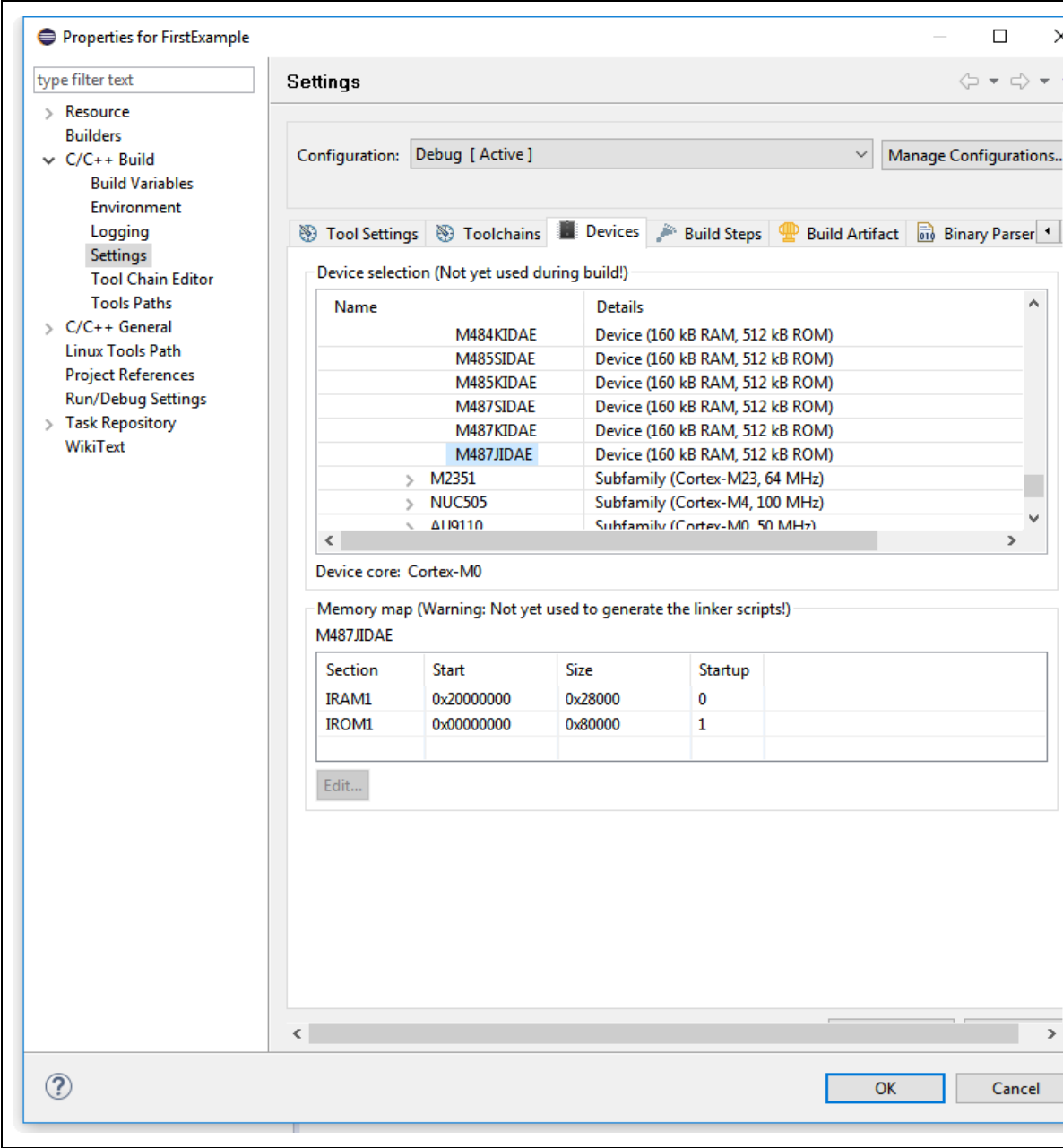


图 3-17 选择设备

最后结果，当我们在调试时可以监看外围寄存器。

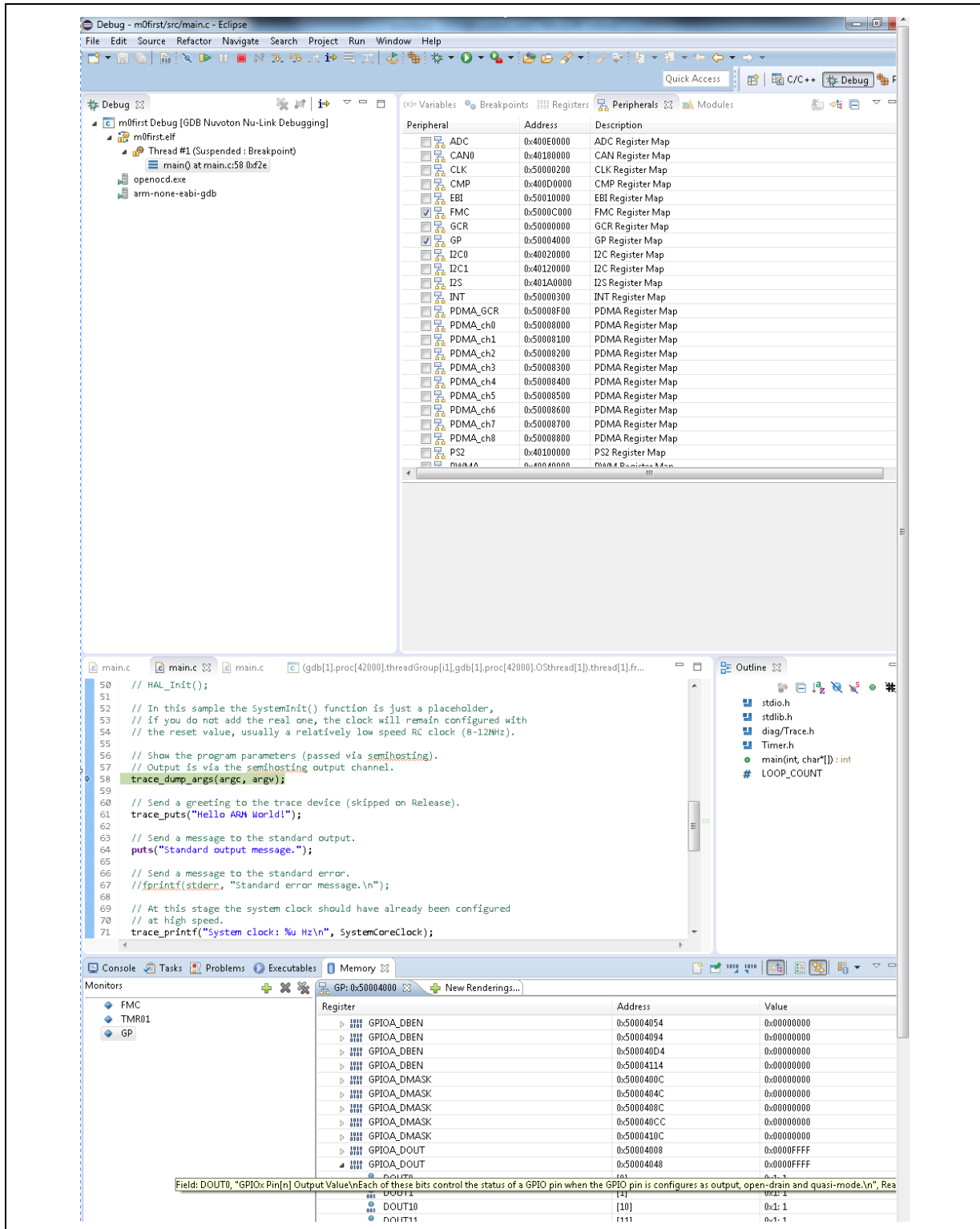


图 3-18 外围寄存器视图

3.7 检测点

为在 Eclipse 新增检测点，我们需作下面的步骤。

1. 在 Outline 视图中，选取一个全局变量，例如: `g_seconds`。
2. 鼠标右键单击它并选取 **Toggle Watchpoint**。

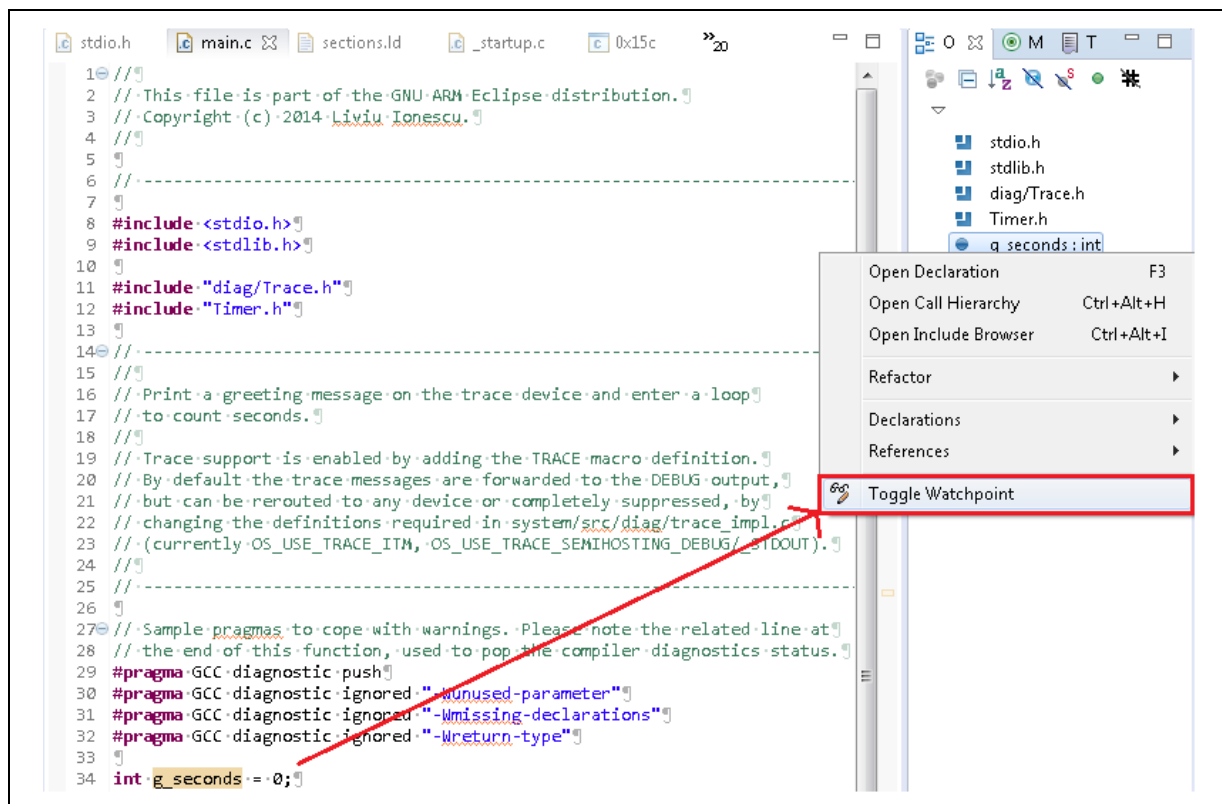


图 3-19 切换检测点

3. 设置检测点。若要检测表达式被读取时让程序停止运行的话，选取 **Read** 复选框。若要检测表达式被写入时让程序停止运行的话，选取 **Write** 复选框。

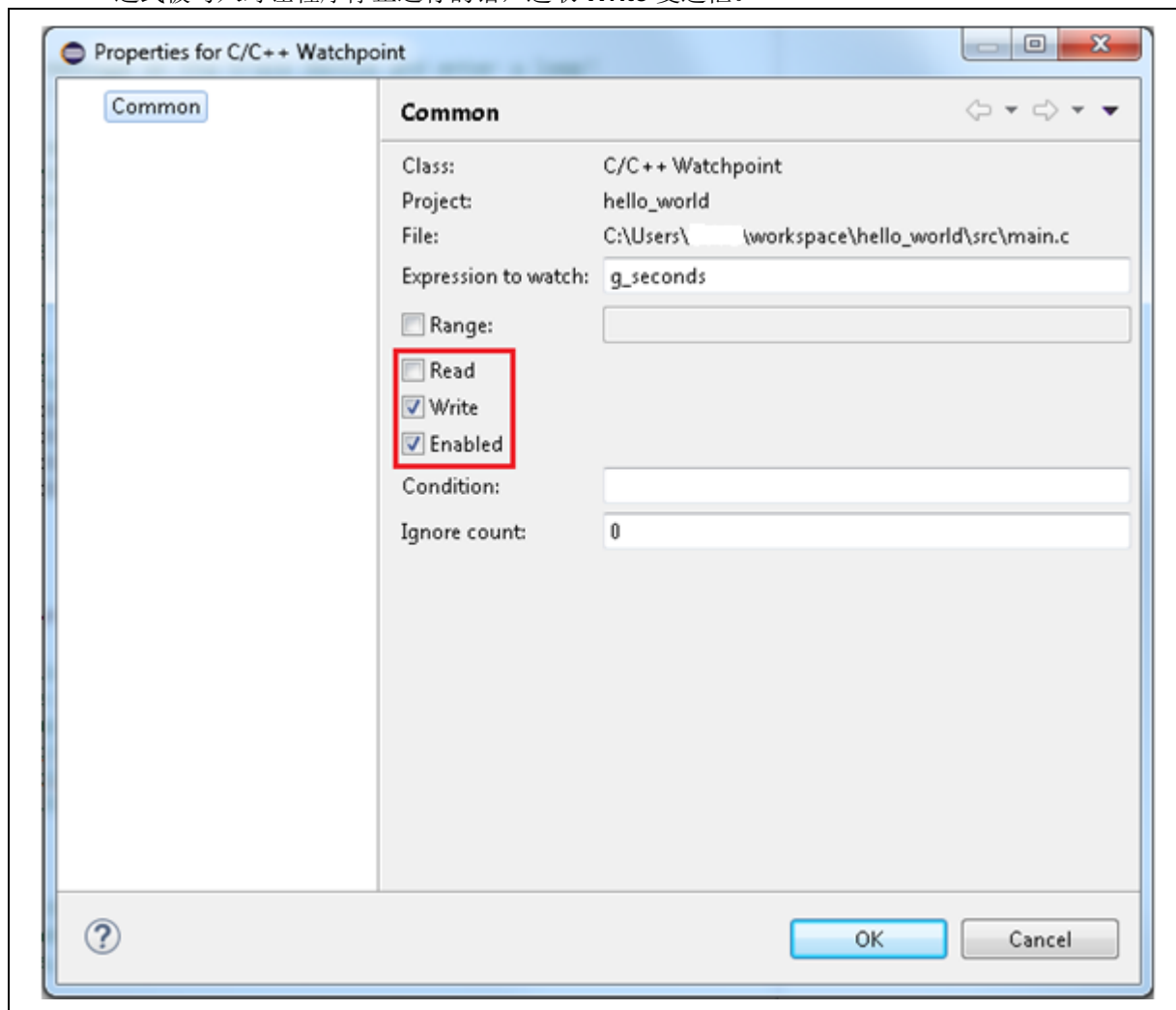


图 3-20 C/C++ 检测点属性对话框

当检测点被新增时，它会出现在断点视图里面。

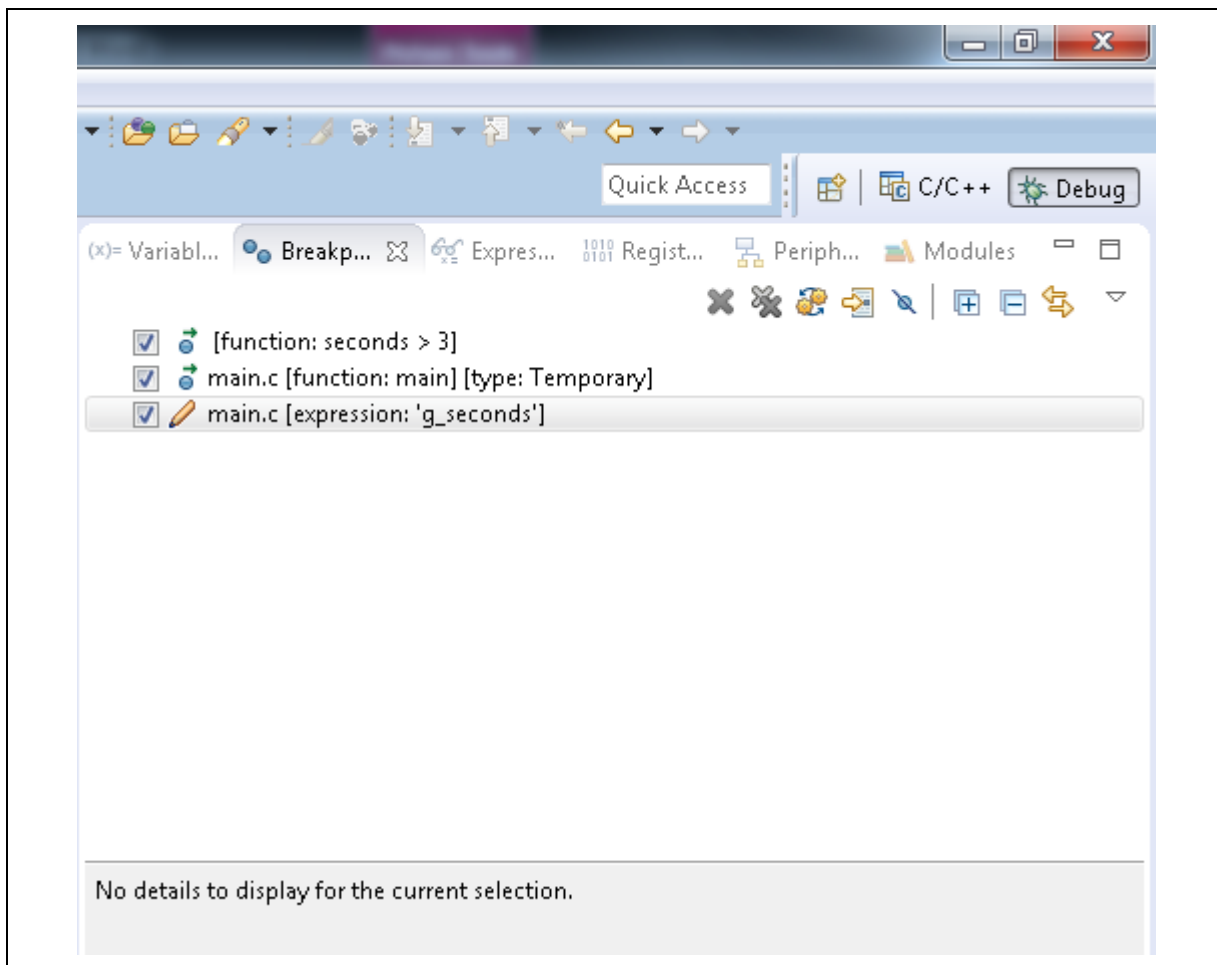


图 3-21 在断点视图中新增的检测点

3.8 在 RAM 中调试

为在 RAM 上调试，有几个步骤需要完成：

1. 修改 Id 脚本。
2. 设定 PC 值至特定 RAM 地址。
3. 设定 SP 值至特定 RAM 地址。
4. 下载二进制文件到 RAM。

Id 脚本负责告诉链接器编译完成的执行文件如何配置。举例来说，一个内存配置如下：

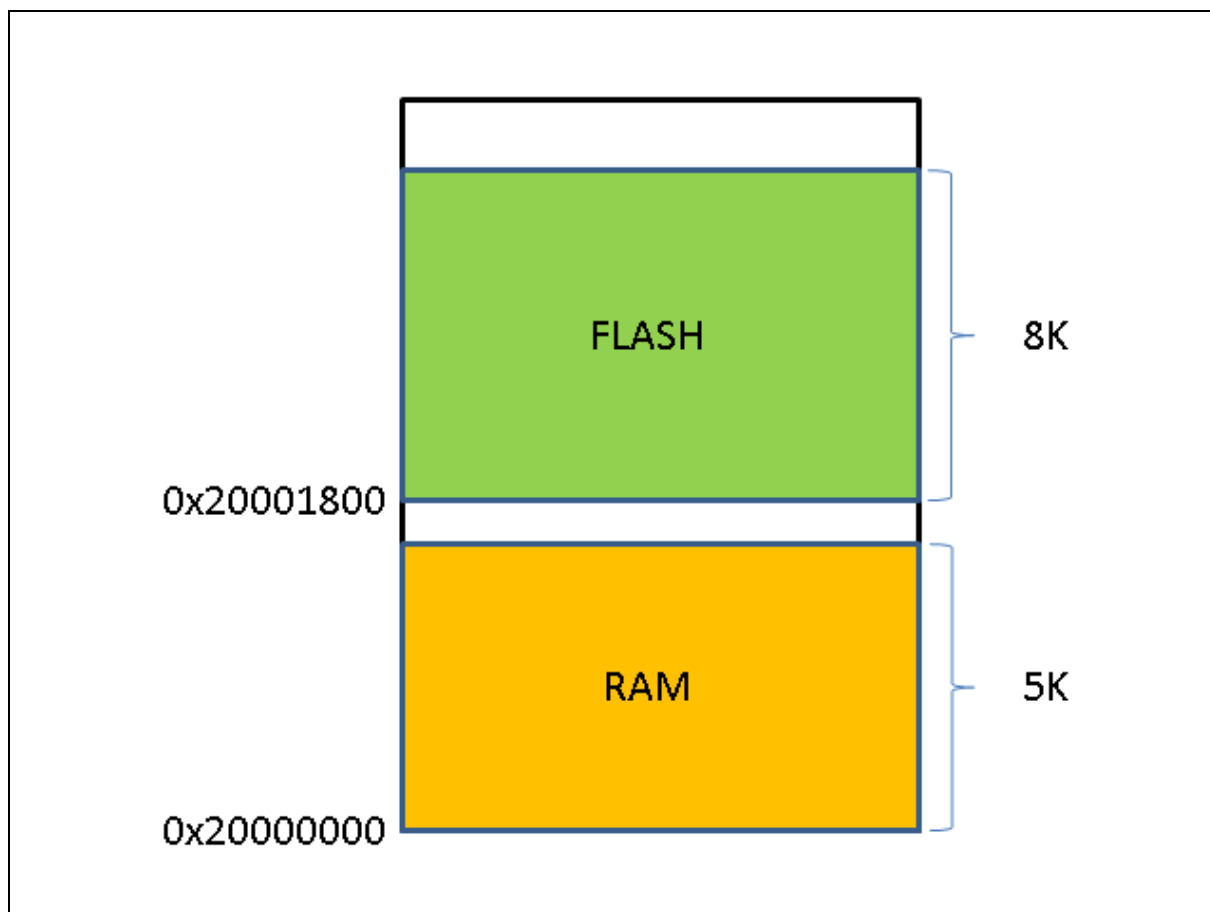
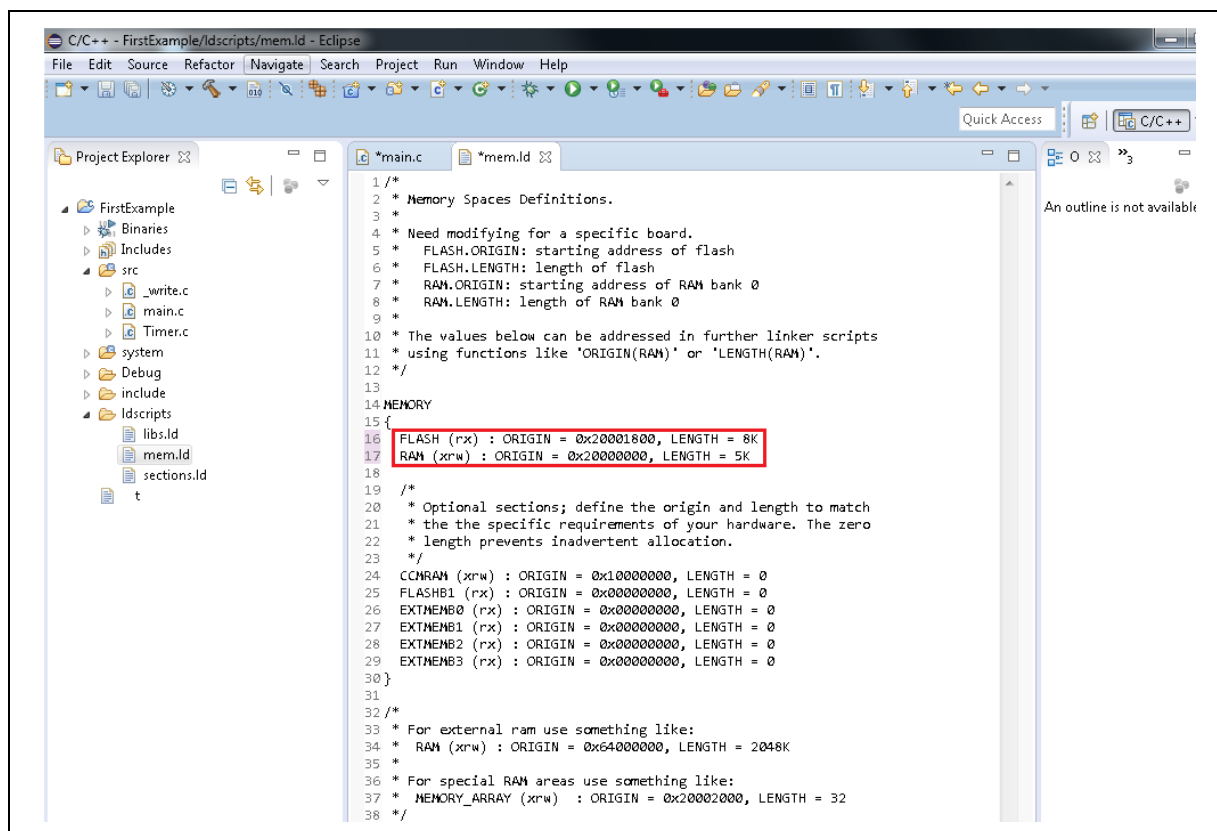


图 3-22 内存配置

修改后的 mem.ld 脚本应该符合内存配置的设计。



```

1 /*
2  * Memory Spaces Definitions.
3  *
4  * Need modifying for a specific board.
5  * FLASH.ORIGIN: starting address of flash
6  * FLASH.LENGTH: length of flash
7  * RAM.ORIGIN: starting address of RAM bank 0
8  * RAM.LENGTH: length of RAM bank 0
9  *
10 * The values below can be addressed in further linker scripts
11 * using functions like 'ORIGIN(RAM)' or 'LENGTH(RAM)'.
12 */
13
14 MEMORY
15 {
16     FLASH (rx) : ORIGIN = 0x20001800, LENGTH = 8K
17     RAM (xrw) : ORIGIN = 0x20000000, LENGTH = 5K
18 }
19
20 /*
21 * Optional sections; define the origin and length to match
22 * the the specific requirements of your hardware. The zero
23 * length prevents inadvertent allocation.
24 */
25 CCMRAM (xrw) : ORIGIN = 0x10000000, LENGTH = 0
26 FLASHB1 (rx) : ORIGIN = 0x00000000, LENGTH = 0
27 EXTMEMB0 (rx) : ORIGIN = 0x00000000, LENGTH = 0
28 EXTMEMB1 (rx) : ORIGIN = 0x00000000, LENGTH = 0
29 EXTMEMB2 (rx) : ORIGIN = 0x00000000, LENGTH = 0
30 EXTMEMB3 (rx) : ORIGIN = 0x00000000, LENGTH = 0
31 }
32
33 /*
34 * For external ram use something like:
35 * RAM (xrw) : ORIGIN = 0x64000000, LENGTH = 2048K
36 *
37 * For special RAM areas use something like:
38 * MEMORY_ARRAY (xrw) : ORIGIN = 0x20002000, LENGTH = 32
39 */
    
```

图 3-23 修改 Mem.ld 脚本

为设定 PC 和 SP 值至特定地址，我们需要输入它们至调试配置，如下。基于前面的内存配置，PC 和 SP 的地址应分别为 Reset_Handler 和 0x20001400。此外，设定 Vector Table Offset Register (0x 0xE000ED08)为 0x20000000 并且取消 **Pre-run/Restart reset** 按键。为下载二进制文件到 RAM，勾选 **Load executable to SRAM** 按键和取消 **Load executable to flash** 按键。点选 **Debug** 按键来开始调试阶段。

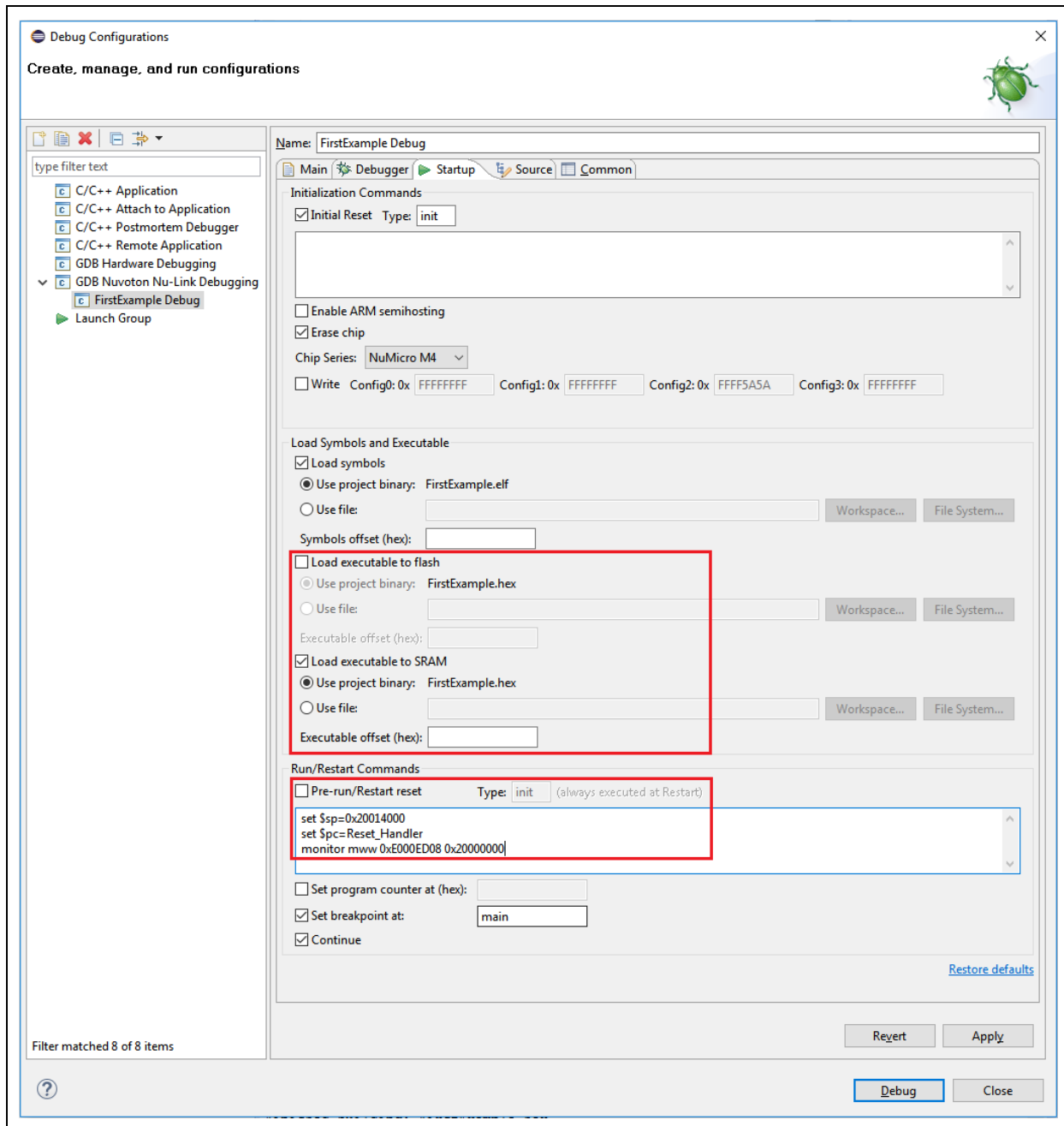


图 3-24 设定调试配置

当程序停在主函数时，我们打开内存视图。从那可以确认到二进制文件已成功地加载至 RAM。第一个字代表 SP 地址，接下来的字代表处理函数的地址。

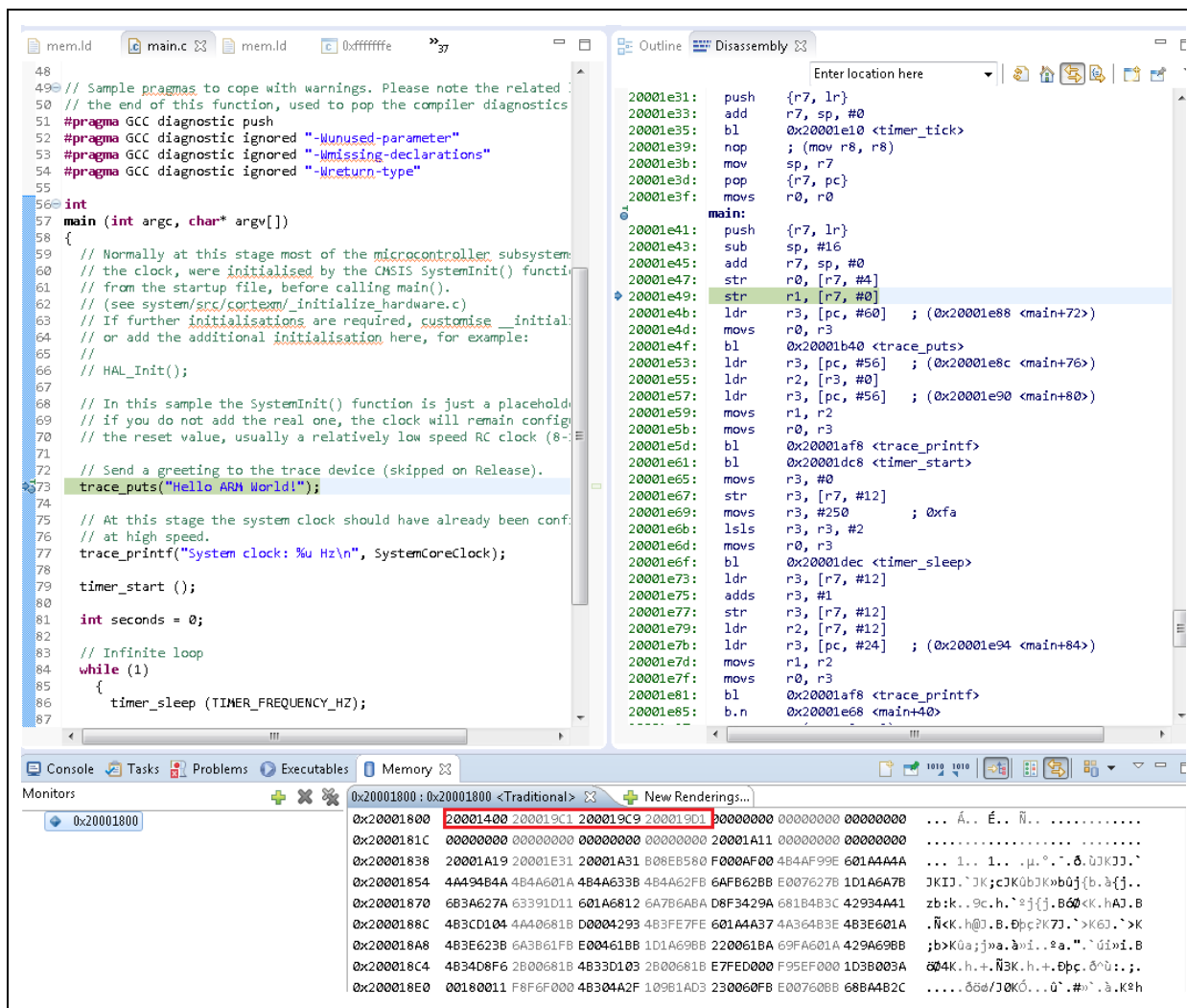


图 3-25 在 RAM 中调试

4 Q&A

1. Q: 我们能同时在 Eclipse、Keil 和 Iar 调试吗?

A: 不行, 我们必须停下 Eclipse 的调试模式, 才能在别的 IDE 上调试。

2. Q: 我们能在 Eclipse 上调试时同时使用新唐的开发软件(例如 ICP Programming Tool)吗?

A: 不行, 我们必须停下 Eclipse 的调试模式, 才能使用它们, 反之亦然。

3. Q: 总共支持几个断点和检测点?

A: 视硬件而定。M0 有 4 个断点, 2 个检测点。M4 有 6 个断点, 4 个检测点。M23 有 4 个断点, 4 个检测点。目前, 我们不支持闪存断点。

4. Q: 如何更新 Nu-Link 的固件?

A: 请使用 ICP Programming Tool 或 Keil 来更新固件。

5. Q: 工程创建后要如何修改闪存和内存定义的大小?

A: 请至 Idscripts 文件夹找到并打开 Id 文件, 那里可以修改它们。

6. Q: 为何应用程序进不去调试模式?

A: 首先, 我们需安装所有必要的软件, 请参照 2.3 节的说明。接着检查下面的列表:

I. 如果正在调试的话, 离开上一次的调试模式。

II. 闪存和内存定义的大小要正确。

III. 进入调试前, OpenOCD 不该处在运行中。为检查它有没有运行, 请打开微软任务管理器或系统监视器。如果看到 OpenOCD 进程正在跑, 请关掉该进程。

IV. 目标芯片不应该被别的 IDE 和工具占据住。

V. Debugger 标签页的 **Config options** 字段要正确。

VI. Startup 标签页的 **Initial Reset type** 应为 **init**。**Pre-run/Restart reset type** 应为 **init**。

VII. Eclipse Preferences 的内容要正确, 请参阅 2.3.3 节。

VIII. 将 Nu-Link 固件和驱动程序升级到最新版本。

IX. 检查执行文件是否已正确地被写入目标芯片。

X. 检查 SP。如果它是错的, 请指定它至正确的位置。

XI. 写入正确的 Config 值至目标芯片。

XII. 如果操作系统是 Windows 7, 请使用 USB 2.0 端口, 而不是 USB 3.0 端口。

XIII. 工程路径不能存在任何特殊字符或空白, 例如#\$&`.{}。

XIV. 重开机计算机。

7. Q: 在 GNU/Linux 上, 如何为 Nu-Link 新增 udev 规则?

A: 当透过 Nu-Link 存取目标芯片时, GNU/Linux 需要 USB 权限。我们可为 Nu-Link 新增 udev 规则以获得该权限。这里列出步骤:

I. 新增 User 至 plugdev 群组。在 Terminal 输入下面指令:

sudo useradd -G plugdev \$USER

II. 新增 Nu-Link 至 udev。在 Terminal 输入下面指令:

cd /etc/udev/rules.d 和 **sudo gedit 10-openocd-nulink.rule**

III. 新增下面字符串至该文件:

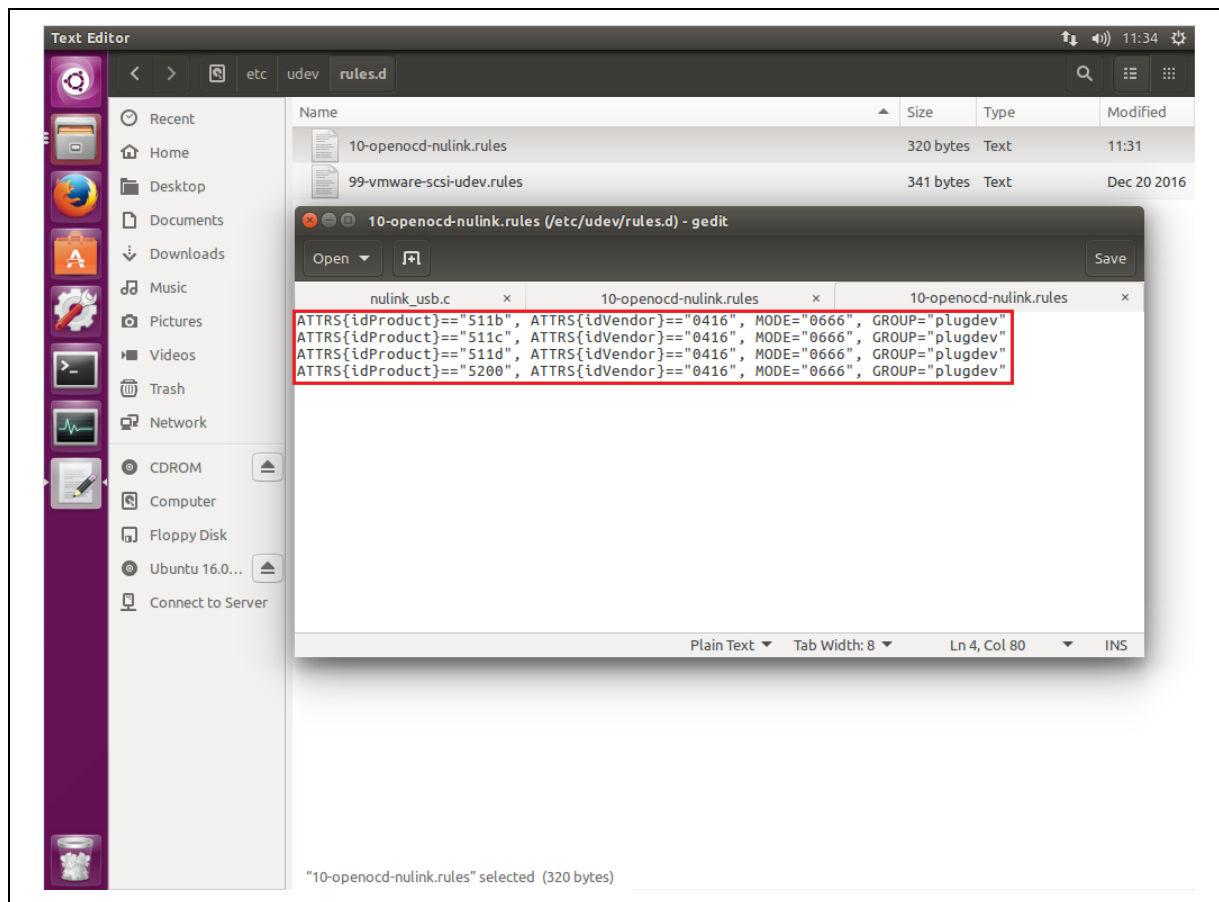


图 4-1 新增 Udev 规则

IV. 重载新的 udev 规则。在 Terminal 输入下面指令:

sudo udevadm trigger

8. Q: 如何编辑 openocd_nulink_path 字符串?

A: **openocd_nulink_path** 字符串存放 OpenOCD 执行文件的所在位置。升级 NuEclipse 后, 该字符串可能会维持在上一版 OpenOCD 的路径。为修正它, 单击 **Window > Preferences**。

Preferences 对话框会出现。前往 **Run/Debug > String Substitution**。找到并编辑 **openocd_nulink_path** 字符串, 使其指向新的 OpenOCD 路径。

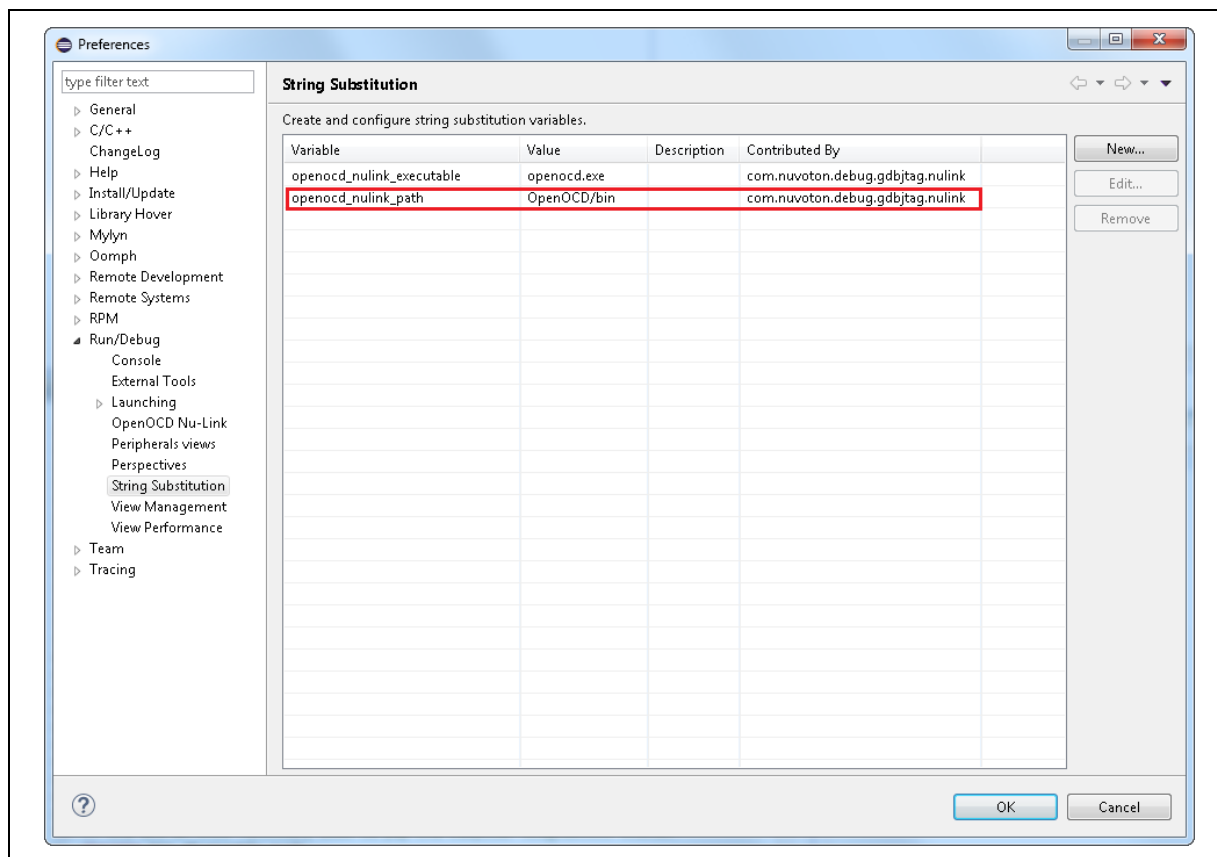


图 4-2 Preferences 设置 String Substitution 的对话框

9. Q:在 Windows 上, 为何更新或安装软件 Packs 会失败呢?

A: 可能的原因之一是 Windows 资料夹的写入权限被拒绝。我们需要找到正确的资料夹并允许写入权限。在 64 位 Windows 上, 我们放置软件 Packs 的位置是 C: \ Program Files (x86) \ Nuvoton Tools \ Packages。同样地, 在 32 位 Windows 上, 我们放置软件 Packs 的位置是 C: \ Program Files \ Nuvoton Tools \ Packages。

5 修订历史

Date	Revision	Description
2017.03.31	0.01.000	Alpha 版发布。
2017.06.30	1.01.000	Beta 版发布
2018.09.15	1.01.013	官方版发布
2018.11.30	1.01.014	1. 支持 NUC505。 2. 更新新工程向导
2019.08.09	1.01.015	支持 M031、M261 和 M480LD。
2020.03.06	1.01.016	支持 M031BT、NUC1311、M2354 和 M479。
2020.09.30	1.01.017	支持 M030G、M071、M0A21、M251 和 M471。

使用本软体表示您接受已下之免责声明(声明以英文为准)。

免责声明：本公司所提供之本软体係以现况提供，本公司不作任何明示或默示之担保，包括但不限于商业适售性及特定目的适用性之保证，均无提供。您应自行判断抉择或评估本软体之适用性及正确性，对于任何因本软体而生之直接、间接、附带、特别、惩罚性或衍生性损害，本公司一概不负任何法律责任。

Notice: Using this software indicates your acceptance of the disclaimer hereunder:

THIS SOFTWARE IS FOR YOUR REFERENCE ONLY AND PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. YOUR USING THIS SOFTWARE/FIRMWARE IS BASED ON YOUR OWN DISCRETION, IN NO EVENT SHALL THE COPYRIGHT OWNER OR PROVIDER BE LIABLE TO ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.