使用 OTAP 客户端软件重新编程 KW36 设备

原文：https://community.nxp.com/docs/DOC-342493

介绍

本文指导用户通过 OTAP bootloader 给 KW36 加载新的软件映像（software image），重点介绍了 KW36 中的各种 memory。

（译者注：OTAP = Over The Air Programming，也叫做 OTA，即空中下载技术，指通过无线的方式为设备升级新的程序。）

软件需求

1. IAR Embedded Workbench IDE 或 MCUXpresso IDE
2. FRDM-KW36 1.3.4 SDK 或更高版本

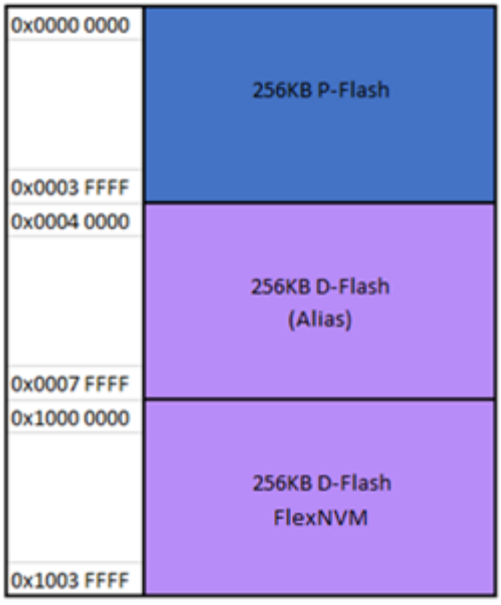
硬件需求

1. FRDM-KW36 开发板

KW36 Flash Memory 在 OTAP 软件升级中的使用

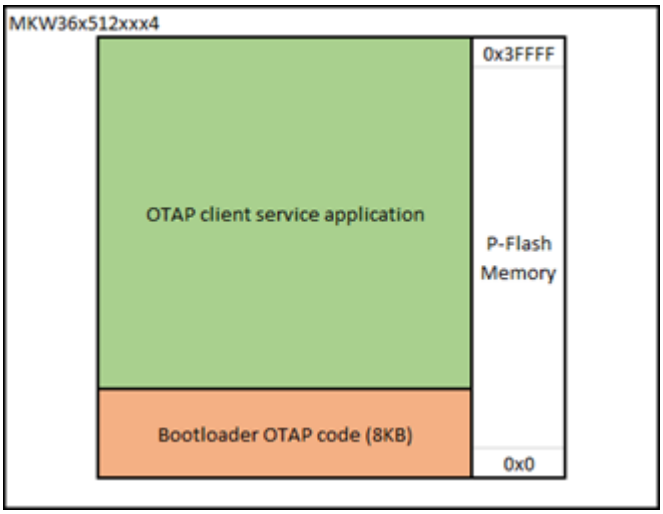
KW36 的 Flash 可划分为如下区域：

- 一个 256 KB 的 Program Flash（P-Flash），每 2 KB 一个扇区（sector），地址范围为 0x0000_0000 至 0x0003_FFFF。
- 一个 256 KB 的 FlexNVM，每 2 KB 一个扇区，地址范围为 0x1000_0000 至 0x1003_FFFF。这个存储空间也可以通过别名（Alias）访问，使用地址 0x0004_0000 至 0x0007_FFFF。

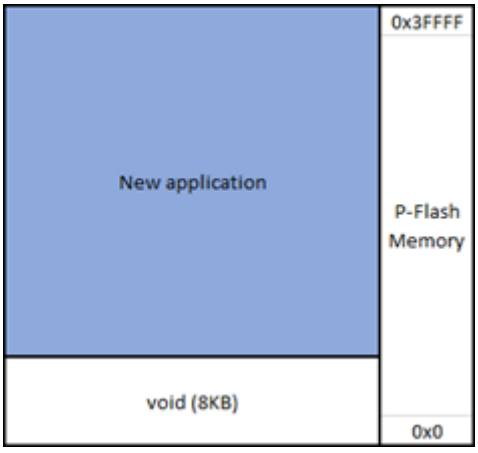


下面的描述用来帮助理解 OTAP 客户端软件（OTAP Client software）是如何工作的：

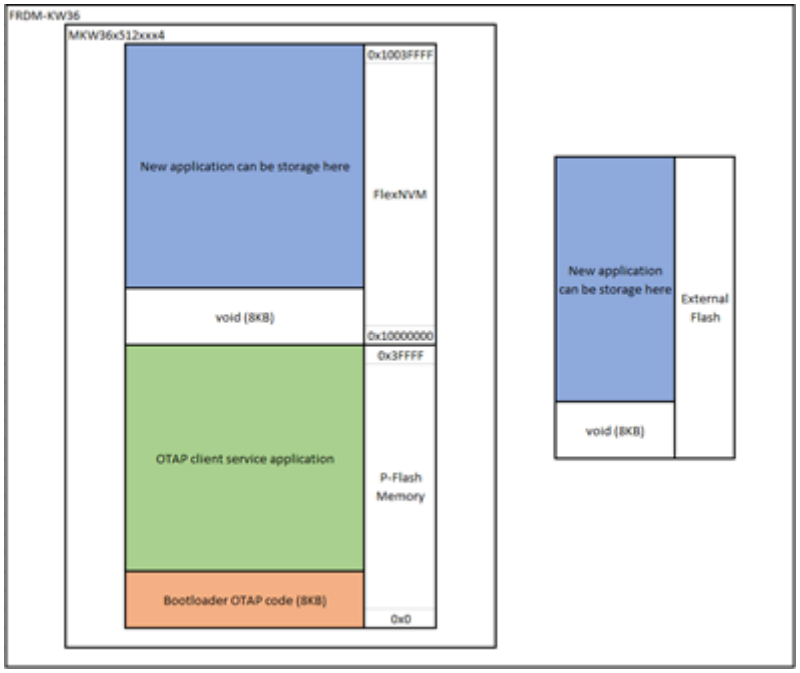
1. OTAP 客户端软件分为两部分：OTAP bootloader 和 OTAP 客户端服务（OTAP client service）。OTAP bootloader 检查是否有新的软件映像可以用来对设备重新编程。OTAP 客户端服务包含了蓝牙 BLE 自定义服务（custom service），用来和提供新的软件映像的服务器互相通信。所以，在测试 OTAP 之前，我们要先分两次把 OTAP 烧写到设备里，一次是烧写 OTAP bootloader 工程，一次是烧写 OTAP 客户端服务工程。能让两个软件工程在同一个设备中共存，其中的机制是把每个软件工程保存到一个独立的 memory 区域中去，而这是通过 linker 文件中的配置实现的。在 KW36 中，bootloader 程序占据了 8 KB 的存储空间，从地址 0x0 到 0x1FFF；剩下的空间（0x2000 到 0x3FFFF）保留给 OTAP 客户端服务程序。



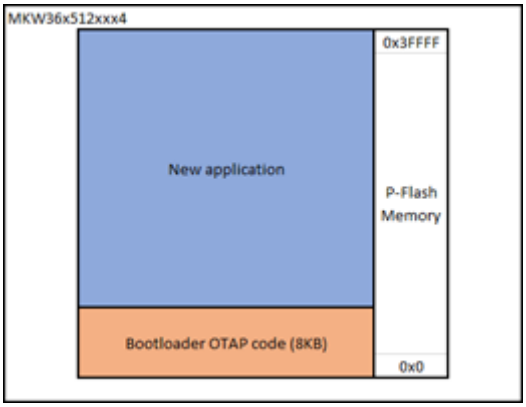
2. 在创建新软件映像时，开发人员需要通过 linker 文件为自己的代码设置一个 8 KB 的地址偏置，因为前面的 8 KB 是为了 bootloader 所保留的。



3. 通过蓝牙连接，服务器将所有的小代码块发送到客户端。客户端软件可以把新的代码保存到外部 Flash AT45DB041E 中或者内部的 FlexNVM 中。保存在哪里可以通过 OTAP 客户端软件来选择。



4. 当广播结束，所有的小代码块发送完成后，OTAP bootloader 会检测到这种情况，并发出命令开始重新编程。由于新的程序已经预留了 8 KB 的偏置，OTAP bootloader 可以直接从地址 0x1FFF 开始烧写，用新程序覆盖原 OTAP 客户端服务程序（OTAP client service）。最后，bootloader 触发新的程序开始运行。（译者注：原文有误，应该是从 0x2000 地址开始烧写。）



用 IAR Embedded Workbench 在 KW36 上准备 OTAP 测试

本节提供了在 FRDM-KW36 开发板上测试 OTAP 所需的步骤。

1. 烧写 OTAP bootloader 到 FRDM-KW36 开发板。
   1.1. 在下面路径中，将 bootloader 预编译的 bin 文件拖拽到 FRDM-KW36 板上。
   <SDK_download_root>\tools\wireless\binaries\bootloader_otap_frdmkw36.bin

2. 烧写 OTAP 客户端服务程序到 FRDM-KW36 开发板。

2.1. 打开下面路径中的 OTAP 客户端项目。
<SDK_download_root>\boards\frdmkw36\wireless_examples\bluetooth\otac_att\freertos\iar\otac_att_freertos.eww

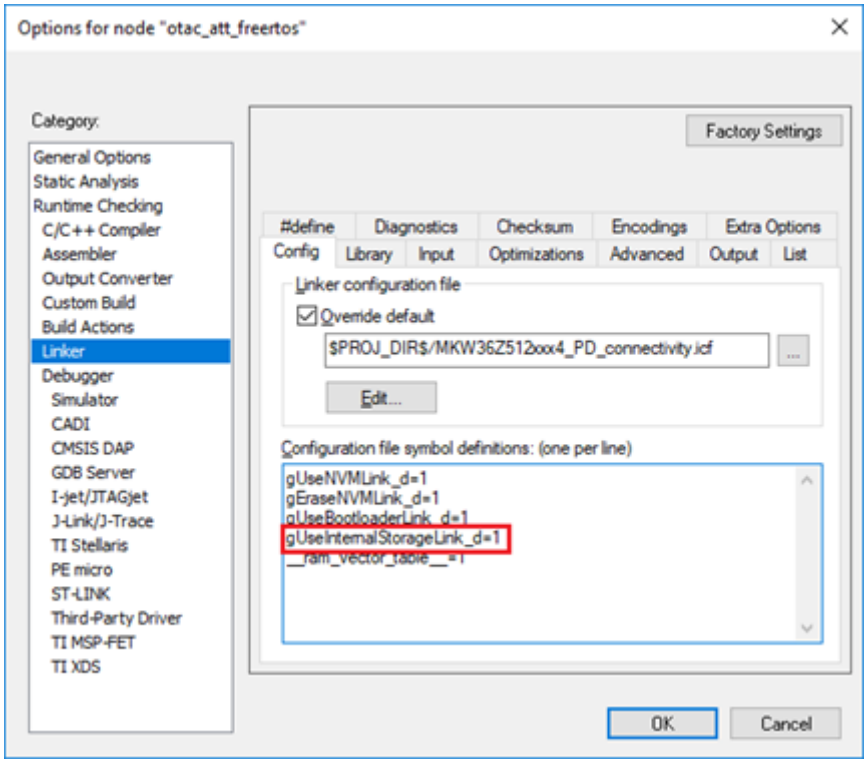
2.2. 设置存储方式。

a. 在 workspace 中，找到 app_preinclude.h 文件。

- 如果使用外部 Flash 存储，将 gEepromType_d 设置为“gEepromDevice_AT45DB041E_c”
- 如果使用内部 FlexNVM 存储，将 gEepromType_d 设置为“gEepromDevice_InternalFlash_c”

b. 打开工程选项窗口（ALT+F7），在 Linker/Config 窗口中，更新 gUseInternalStorageLink_d 配置。

- 如果使用外部 Flash 存储，将“gUseInternalStorageLink_d=1”从 Configuration file symbol definitions 窗口中移除。
- 如果使用内部 FlexNVM 存储，将“gUseInternalStorageLink_d=1”加入到 Configuration file symbol definitions 窗口。



2.3. 将工程下载到 FRDM-KW36 开发板（Ctrl+D），然后退出 debug 模式（Ctrl+Shift+D）。用工程中默认的 linker 配置就可以把 OTAP 客户端服务程序下载到适当的地址位置。

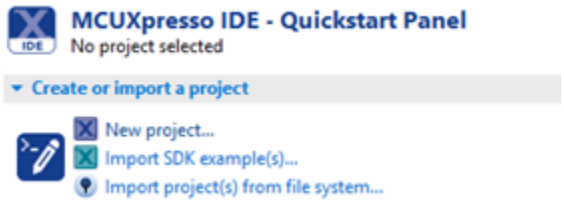
用 MCUXpresso IDE 在 KW36 上准备 OTAP 测试

本节提供了在 FRDM-KW36 开发板上测试 OTAP 所需的步骤。

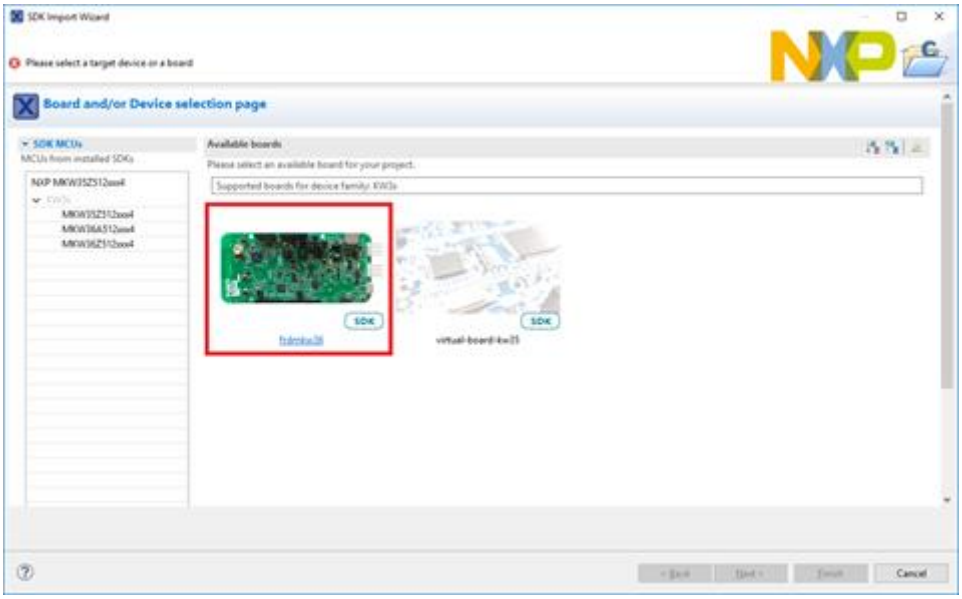
1. 烧写 OTAP bootloader 到 FRDM-KW36 开发板。

1.1. 在下面路径中，将 bootloader 预编译的 bin 文件拖拽到 FRDM-KW36 板上。
<SDK_download_root>\tools\wireless\binaries\bootloader_otap_frdmkw36.bin

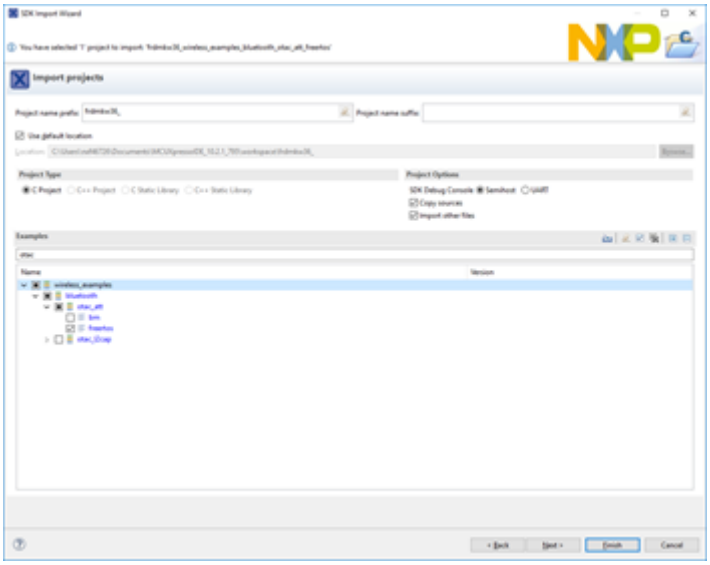
2. 烧写 OTAP 客户端服务程序到 FRDM-KW36 开发板。
2.1. 打开 MCUXpresso IDE，在 Quickstart Panel 中点击 Import SDK example(s)选项。



2.2. 点击两次 frdmkw36 图标。



2.3. 在 examples 项目上面的文本框输入“otac_att”，选择 wireless_examples\bluetooth\otac_att\freertos，最后单击 Finish 按钮。



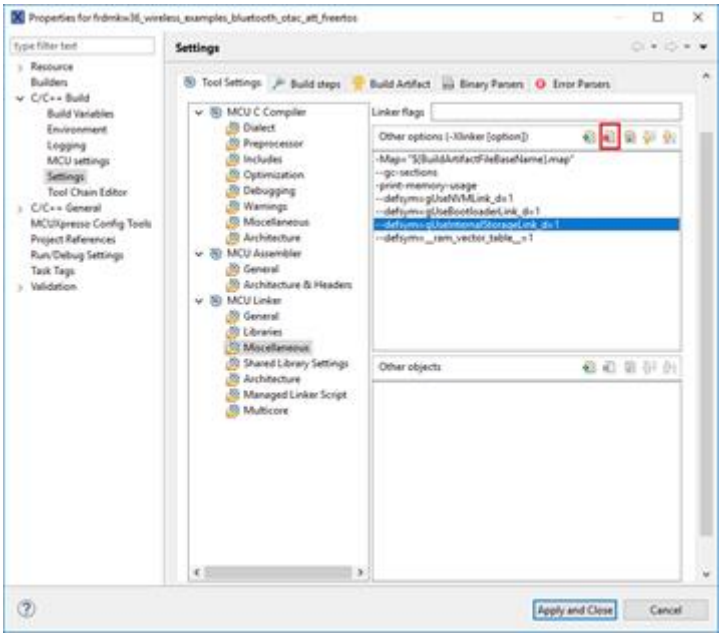
2.4. 设置存储方式。

a. 在 workspace 中，找到 app_preinclude.h 文件。

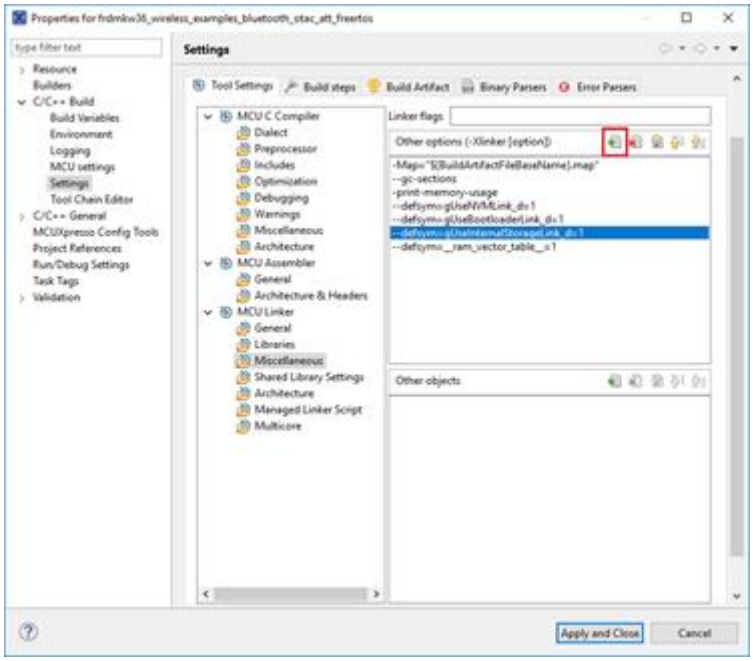
- 如果使用外部 Flash 存储，将 gEepromType_d 设置为“gEepromDevice_AT45DB041E_c”
- 如果使用内部 FlexNVM 存储，将 gEepromType_d 设置为“gEepromDevice_InternalFlash_c”

b. 打开工程属性窗口 Project/Properties，然后转到 C/C++ Build/Settings/Linker/Miscellaneous。

- 如果使用外部 Flash 存储，点击下面的图标，将“--defsym=gUseInternalStorageLink_d=1”移除。



- 如果使用内部 FlexNVM 存储，点击下面的图标，将“--defsym=gUseInternalStorageLink_d=1”加入。



2.5. 在 Quickstart Panel 中选择“Debug”选项。当项目加载好后，推出 debug 模式。

使用 IoT Toolbox App 运行 OTAP Demo

1. 把在附录 A 或附录 B 中生成的 S-Record 文件保存到手机上的一个已知位置。
2. 打开 IoT Toolbox App，选择 OTAP Demo。点击 SCAN，开始扫描 BLE advertiser。
3. 按下 FRDM-KW36 开发板上的 SW2 按钮开始广播。
4. 和扫描到的蓝牙设备建立连接。

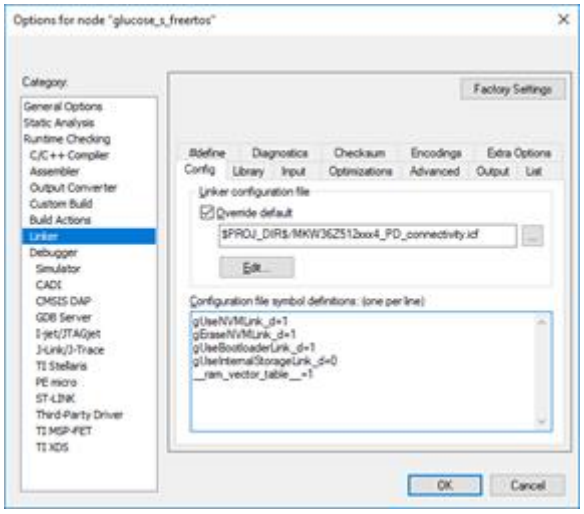
5. 点击 Open，然后找到 S-Record 文件。
6. 点击 Upload，开始文件传输。

7. 当文件传输完成，等待几秒钟让 bootloader 完成新程序的烧写。之后新程序会自动启动。

附录 A，使用 IAR Embedded Workbench 为 FRDM-KW36 建立一个 S-Record 文件

1. 从 SDK 中打开一个想要通过 OTAP bootloader 去烧写的 Connectivity 工程。在这个例子中，我们使用 glucose sensor 工程，在下面的路径。
   <SDK_download_root>\boards\frdmkw36\wireless_examples\bluetooth\glucose_s\freertos\iar\glucose_s_freertos.eww

2. 打开工程选项窗口（ALT+F7）。在 Linker/Config 窗口中，修改 Configuration file symbol definitions 中的 linker 设置，如下所示。要和 OTAP 客户端服务程序中的设置一致。
   gUseNVMLink_d=1
   gEraseNVMLink_d=1
   gUseBootloaderLink_d=1
   gUseInternalStorageLink_d=0
   __ram_vector_table__=1



3. 转到 Output Converter 窗口，取消 Override default 选项，然后在 Output format 下拉菜单中选择 Motorola S-records 格式。点击 OK 按钮。

4. 重新编译这个工程。

5. S-Record 文件（.srec）会在下面的路径中。
   <SDK_download_root>\boards\frdmkw36\wireless_examples\bluetooth\<Desired_project>\freertos\iar\debug

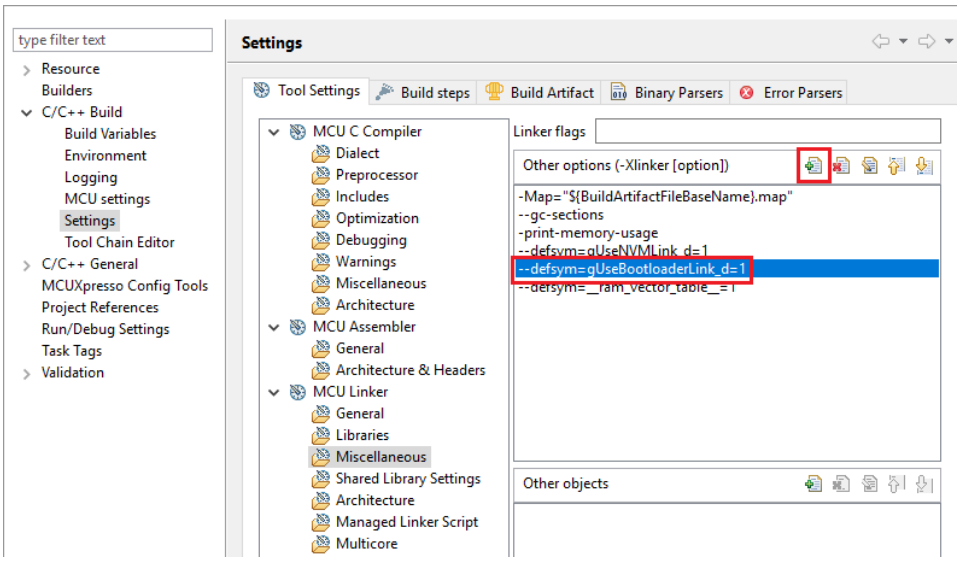
附录 B，使用 MCUXpresso IDE 为 FRDM-KW36 建立一个 S-Record 文件

1. 在 MCUXpresso IDE 中打开一个想要通过 OTAP bootloader 去烧写的 Connectivity 工程。在这个例子中，我们使用 glucose sensor 工程。

2. 打开 MKW36Z512xxx4_PD_connectivity.ld 这个 linker 文件，找到下面图片中的代码段，然后删掉 FILL 和 BYTE 两行。

```
/* Remove this section to keep the nvm section on writting the device */
.NVM :
{
  FILL(0xFFFFFFFF);
  . = ORIGIN(NVM_region) + LENGTH(NVM_region) - 1;
  BYTE(0xFF);
} > NVM_region
```

3. 打开 Project/Properties，然后打开 C/C++ Build/Settings/Linker/Miscellaneous 窗口。点击下面图片中的图标，然后在 Other options 中加上如下定义：
   --defsym=gUseBootloaderLink_d=1。点击 Apply and Close 按钮。



4. 重新编译工程。

5. 在 workspace 中展开 Binaries 图标，在“.axf”文件上单击鼠标右键。选择 Binary Utilities/Create S-Record 选项。S-Record 文件会以“.s19”的后缀名被保存在 workspace 的 Debug 文件夹中。