AN97060

PSoC® 4 BLE 和 PRoC™ BLE：无线传输（Over The Air - OTA）器件固件升级（DFU）指南

作者： Deepak John

相关器件系列：所有 PSoC 4 BLE 和 PRoC BLE

相关代码示例：CE95351

相关应用笔记：要想获取完整列表，请点击此处。

想要获取本应用笔记的最新版本或相关项目文件，请访问 http://www.cypress.com/go/AN97060。

AN97060 为基于 PSoC® 4 和 PRoC™ BLE 器件的应用提供了实现无线传输（OTA）固件升级功能的指导信息。

目录

1 简介 1
2 PSoC 和 PRoC 资源 2
3 PSoC Creator 3
4 BLE OTA Bootloader 4
4.1 外部存储器 OTA Bootloader 5
4.2 固定堆栈 OTA Bootloader 6
4.3 可升级堆栈 OTA Bootloader 7
5 为目标项目添加固件 OTA Bootloader 支持 9
5.1 创建基本的示例目标项目 9
5.2 添加外部存储器 OTA Bootloader 12
5.3 添加固定堆栈 OTA Bootloader 15
5.4 添加可升级堆栈 OTA Bootloader 22
6 执行 OTA 升级 29
6.1 使用 Bootloader 主机工具进行升级 30
6.2 使用 CySmart PC 工具进行升级 31
6.3 使用 CySmart 移动应用进行升级 32
7 测试 OTA 功能 32
8 其他注意事项 32
8.1 连接/配对信息 32
8.2 调试 34
8.3 数据长度扩展（DLE） 35
8.4 数据持久性 36
9 总结 38
10 相关应用笔记 38
A 附录 A 39
A.1 创建一个示例项目工作区 39
A.2 为现有工作区添加一个示例项目 40
A.3 选择其它器件 42
A.4 添加 Bootloader 服务 44
A.5 为其它赛普拉斯 BLE 器件配置固定堆栈 OTA 项目 46

1 简介

无线传输（OTA）器件固件升级基本上是一种引导加载机制，它使用无线链接进行更新目标器件上的固件。OTA 可通过任意无线链接执行，但是本文档中所描述的 OTA 是指通过低功耗蓝牙（Bluetooth® Low Energy — BLE）链接执行的。通过 BLE 器件中的 OTA 功能，可以升级器件功能或修正现场使用器件中的固件问题。

本应用笔记对各种 OTA 升级选项以及如何根据所使用的产品选择正确选项进行了简要描述。它还提供了有关测试和故障排除的详细信息，有助于终端产品的功能集成和使用。

阅读本文档前，请阅读 AN73854 应用笔记，它简单介绍了 Bootloader 的原理和技术，并且说明了如何使用 PSoC Creator™在 PSoC 3、PSoC 4 和 PSoC 5LP 器件中轻松快速实现 Bootloader。

除了本文档中描述的 OTA 功能外，您还可以使用 PSoC Bootloader 通过其它接口（如 UART、I²C 和 SPI）对 PSoC 4 BLE 和 PRoC BLE 器件（更多信息，请查阅 PSoC 和 PRoC 资源）进行固件更新；请查阅相关应用笔记内容。在本文档的背景下，您将通过使用 BLE 来执行 OTA。

通过依次选择菜单选项 **File > Code Example...**，您可以从 PSoC Creator 中访问与 Bootloader 相关的示例项目。然后在弹出的窗口中输入“bootloader”。

2 PSoC 和 PRoC 资源

赛普拉斯的网站 www.cypress.com 上提供了大量资料，有助于选择符合您设计的 PSoC（可编程片上系统）和 PRoC（可编程片上射频）器件，并能够快速和有效地将器件集成到设计中。有关资源的完整列表，请参考 KBA86521 — 如何使用 PSoC 3、PSoC 4 和 PSoC 5LP 进行设计。下面提供的是 PSoC 4 BLE 和 PRoC BLE 的简要列表：

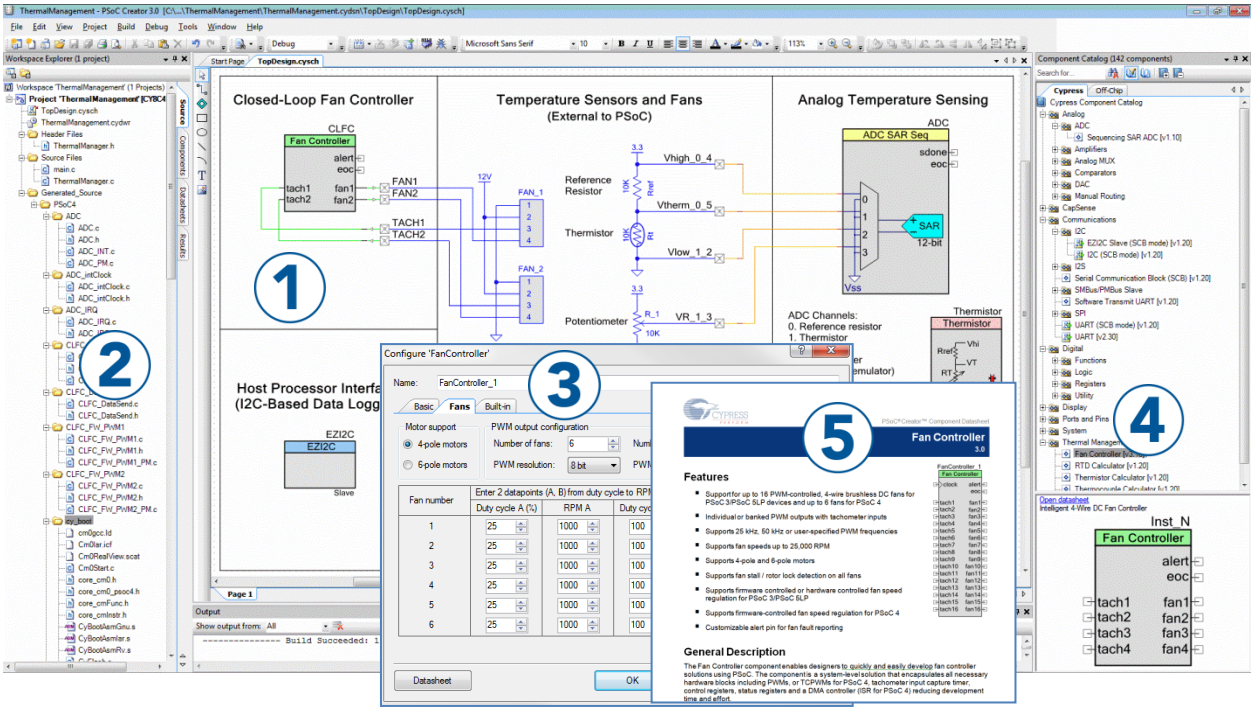
- **概况：**低功耗蓝牙产品系列 — 赛普拉斯物联网无线方案产品路线图
- **产品选型器：**PSoC 4 BLE 或 PRoC BLE。此外，PSoC Creator 还包含了一个器件选择工具。
- **数据手册**描述并提供了适用于 PSoC 4 BLE 和 PRoC BLE 器件系列的电气规范。
- **CapSense®设计指南：**了解如何在 PSoC 4、PSoC 4 BLE 和 PRoC BLE 器件系列中设计电容式触摸感应。
- **应用笔记和代码示例**包括了从基本到高级的广泛主题。许多应用笔记包括了代码示例。
- **技术参考手册（TRM）**对每个 PSoC 4 BLE 和 PRoC BLE 器件系列中所使用的架构和寄存器均进行了详细的说明。
- **开发套件：**
  - CY8CKIT-042-BLE 和 CY8CKIT-042-BLE-A BLE Pioneer 套件：通过这些套件，用户能够使用 PSoC 4 BLE 和 PRoC BLE 器件来评估和开发 BLE 应用。
  - CY5682 PRoC BLE 触控鼠标参考设计套件（RDK）提供了低功耗蓝牙（BLE）或智能蓝牙触控鼠标的量产实施方案。
  - CY5672：PRoC BLE 遥控器 RDK 提供了 BLE 或智能蓝牙遥控器的量产实施方案。
  - CY8CKIT-042 和 CY8CKIT-040 等 PSoC 4 Pioneer 套件均提供了易用且价廉的开发平台。这些套件包括用于连接 Arduino™兼容扩展板/Digilent® Pmod™子卡的连接器。
  - CY8CKIT-049 是一种低成本的原型平台，适用于对 PSoC 4 器件进行初步评估。
  - CY8CKIT-001 是所有 PSoC 器件系列经常使用的开发平台。
- MiniProg3 器件提供了一个用于进行闪存编程和调试的接口。
- **CySmart™**是一个适用于 Windows PC 的 BLE 主机仿真工具。该工具提供了一个易用的 GUI，能够测试和调试您的 BLE 外设应用。
- **CySmart 移动应用**是由赛普拉斯开发的一款 BLE 或智能蓝牙工具。CySmart 可用于连接到多种 BLE 产品，也可以同赛普拉斯的 BLE 开发套件（包括 CY8CKIT-042-BLE PSoC 4 BLE Pioneer 套件、CY5672 PRoC BLE 遥控器 RDK 和 CY5682 PRoC BLE 触控鼠标 RDK）一起使用。
- **赛普拉斯的自定义 BLE 配置文件和服务：**赛普拉斯的 BLE 组件（作为 PSoC Creator 的一部分）会定期更新，以包含蓝牙技术联盟（Bluetooth SIG）定义的基于 GATT BLE 配置文件和服务。除了蓝牙技术联盟定义的配置文件和服务外，赛普拉斯还提供了一些自定义 BLE 配置文件和服务。利用这些非蓝牙 SIG 支持的配置文件和服务的特性，用户可以方便地通过 BLE 进行数据传输。其它器件与赛普拉斯 BLE 器件间进行通信或者各赛普拉斯 BLE 器件之间进行通信时，可以使用这些配置文件和服务。

3 PSoC Creator

PSoC Creator 是一个基于 Windows 的免费集成开发环境（IDE）。通过它可以同时对 PSoC 3、PSoC 4、PSoC 4 BLE、PRoC BLE 和 PSoC 5LP 器件进行硬件和固件设计。请参考图 1。通过使用 PSoC Creator，可以执行以下操作：

1. 将组件图标拖放到主要设计工作区中，以进行您的硬件系统设计
2. 同时设计应用固件和 PSoC 硬件
3. 使用配置工具配置各组件
4. 了解包含一百多个组件的库。
5. 查看组件数据手册

图 1. PSoC Creator 特性



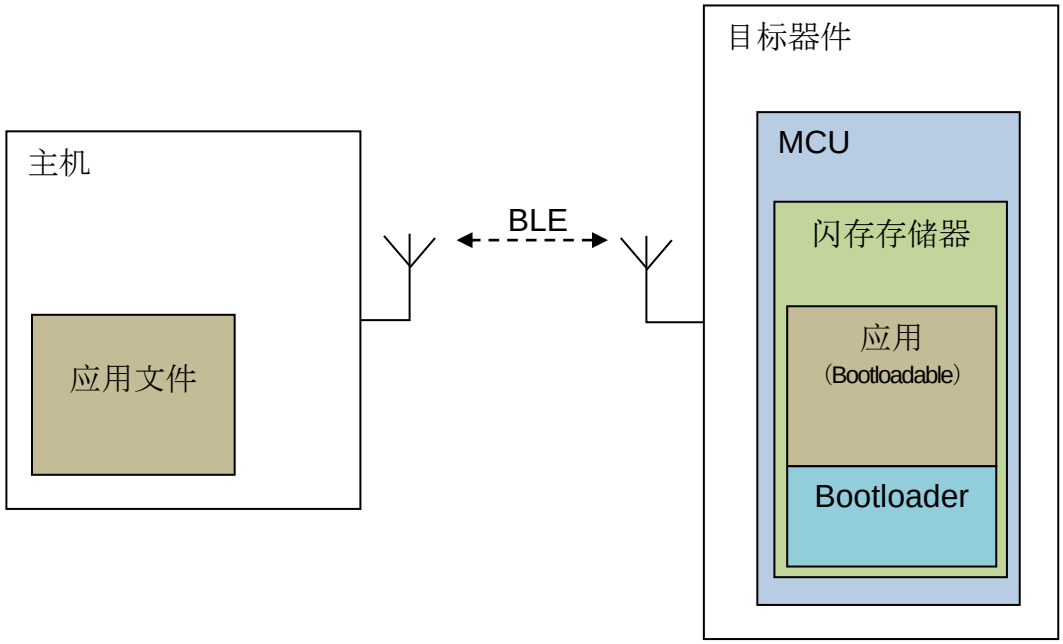
4 BLE OTA Bootloader

以下各术语在本文档中被频繁使用。要想了解本应用笔记，先要掌握它们的定义。

- **Bootloader：**指的是固件的一部分，它明确更新闪存存储器的机制，并负责执行更新操作
- **Bootloadable：**指的是固件的一部分，它包含无线接收并在目标器件的闪存存储器中更新的应用
- **启动程序：**指的是只包含 Bootloader 组件却不包含通信组件的 Bootloader。一个启动程序也可以配置为启动与复制程序（Launcher + Copier）。复制程序是一个附加功能（内置于启动程序），用于将先前已保存的堆栈应用（即 PSoC Creator 项目中包含的 BLE 堆栈）镜像从临时地址复制到堆栈应用闪存空间内（覆盖旧的堆栈应用）。

图 2 目标器件 MCU 中的闪存存储器分两部分：应用（Bootloadable 镜像）和 Bootloader，如图 2 所示。它有多种结构，适用于不同的服务要求。欲了解有关 Bootloader 及其工作原理的详细信息，请查阅 AN73854 和 Bootloader 和 Bootloadable 数据手册。

图 2. BLE Bootloader 系统



将数据（Bootladable 部分）从主机（一个 PC 或一个智能手机）传输到目标器件闪存的过程被称为“引导加载过程”（或“引导加载操作”，或简称为“引导加载”）、“固件升级”或者“器件固件升级（DFU）”。引导加载的另一个常见术语是“系统内编程（ISP）”。

赛普拉斯提供了三种 BLE Bootloader，可将它们添加到任何 BLE 项目中，以实现 OTA 升级：

- 外部存储器 OTA Bootloader
- 固定堆栈 OTA Bootloader
- 可升级堆栈 OTA Bootloader

每种 bootloader 都有其优点和缺点。本部分简要描述了它们的结构和其它设计注意事项。根据本部分内容，您可以选择最适合您设计的 Bootloader。

4.1 外部存储器 OTA Bootloader

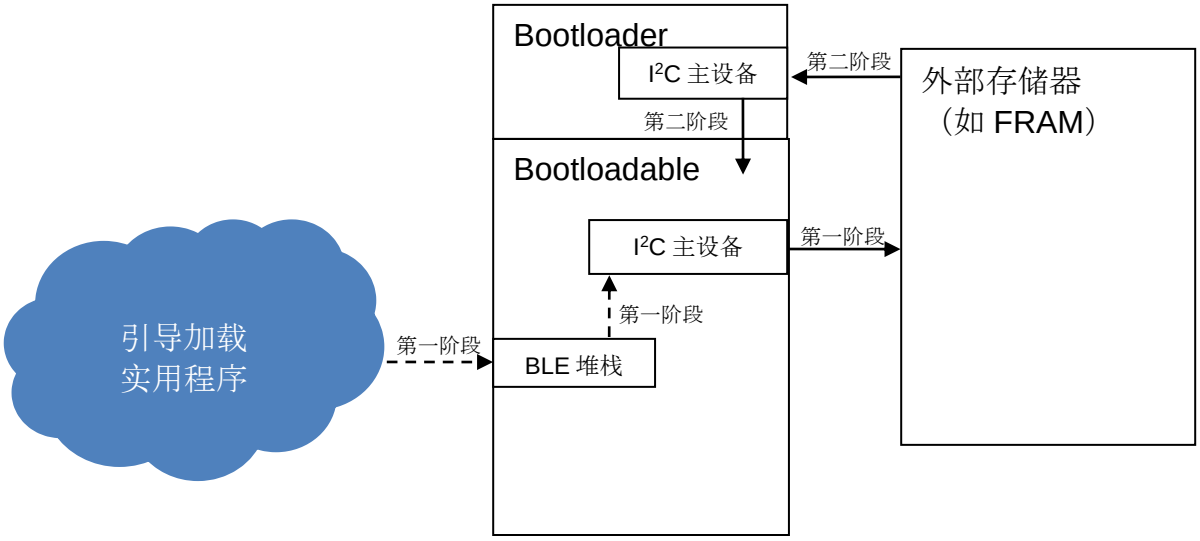
外部存储器 OTA Bootloader 使用了 PSoC Creator 中的 Bootloader 和单 Bootloadable 镜像结构。该 Bootloader 使用外部存储器暂时存储 Bootloadable 镜像。BLE 堆栈位于固件中的 Bootloadable 部分，如图 3 所示。因此，通过固件升级操作可以同时升级应用和 BLE 堆栈。

例如，通过 BLE Pioneer 套件中的 I²C 总线将外部存储器连接到 PSoC/PRoC BLE 器件上。PSoC/PRoC BLE 器件作为 I²C 主设备，外部存储器（FRAM™）作为 I²C 从设备。

执行外部存储器 OTA Bootloader 时，引导加载操作分两个阶段。在第一阶段中，通过 BLE，闪存中现有的 Bootloadable 应用将接收新的 Bootloadable 镜像。接收完应用镜像中每一部分后，它将得到验证，然后通过 I²C 被写入外部存储器。成功接收 Bootloadable 镜像时，固件会将控制权转给 Bootloader。

在第二阶段中，Bootloader 使用第一阶段中接收到的新应用镜像重新编程闪存。整个流程如图 3 所示。箭头表示数据流的方向。

图 3. 外部存储器 OTA 引导加载流程



优点

- 可以同时升级 BLE 堆栈和应用。
- 可以轻松实现固件，因为 Bootloader 和 Bootloadable 是两个独立的项目，并没有共享存储器。

缺点

- 需要一个外部存储器：如果外部存储器只用于实现 OTA，那么便要增加 BOM。
- 与其它 OTA Bootloader 相比，这种 Bootloader 的固件升级时间更长。这是因为需要通过 I²C 将镜像保存到外部存储器内，然后从外部存储器检索镜像。
- 与 I²C 相关的代码同时被存储在 Bootloader 和 Bootloadable 区域内，因此占用了更大的闪存空间。

外部存储器 OTA Bootloader 适用于所有赛普拉斯可编程 BLE 器件。

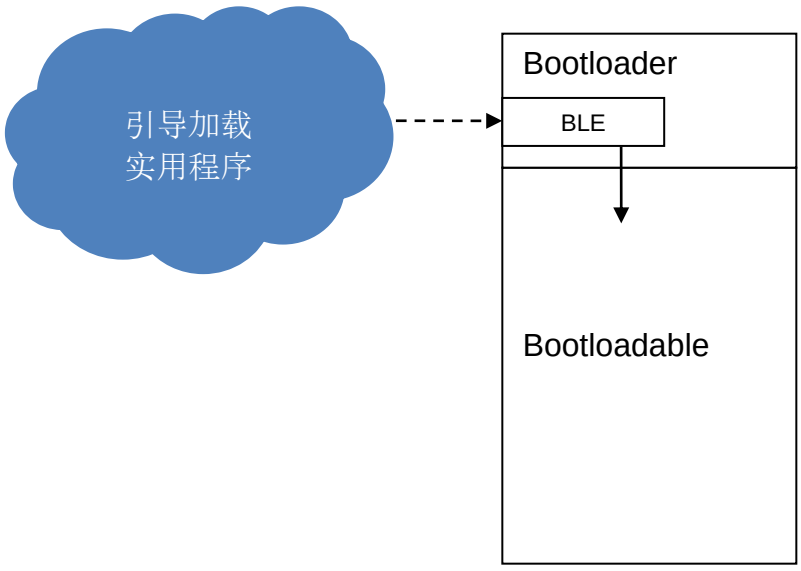
4.2 固定堆栈 OTA Bootloader

类似于外部存储器 OTA Bootloader，固定堆栈 OTA Bootloader 也使用了 PSoC Creator 中的 Bootloader 和单 Bootloadable 镜像结构。然而，因为 BLE 堆栈位于 Bootloader 存储器内，所以不能通过 OTA 升级操作实现 BLE 堆栈升级。Bootloadable 应用必须链接到 Bootloader 存储器中的 BLE API（请参考 BLE 组件数据手册）。

理想情况下，Bootloader 和 Bootloadable 镜像（在两个 PSoC Creator 项目中实现）应有自己的 BLE 堆栈副本；但这样会占用更大的存储器空间。实际上，通过分享堆栈，Bootloader 和 Bootloadable 项目可以重复使用与 BLE 组件相关的代码。这样可以节省相当大的闪存容量，以供固件的应用部分使用。该结构还简化了固件升级流程，如下所示。

使用固定堆栈 OTA Bootloader 结构时，引导加载操作只需要一个阶段。固件直接进入 Bootloader 模式，并等待通过 BLE 链接接收到应用镜像。验证成功后，收到的应用镜像被直接写入到 Bootloadable 区域内。图 4 显示的是固定堆栈 OTA 的引导加载流程，箭头表示数据流的方向。

图 4. 固定堆栈 OTA 引导加载流程



优点

- 由于 Bootloader 和 Bootloadable 复用了 BLE 堆栈，因此节省了闪存存储器空间。
- 这样使升级时间变得更短，这是因为所收到的应用镜像被直接写入到闪存存储器内。
- 不需要使用外部存储器。
- 即使当前的镜像无效，仍可以升级应用镜像。

缺点

- BLE 堆栈是 Bootloader 的一部分，不能通过 OTA 升级（BLE 配置文件也无法升级）。

BLE 堆栈位于闪存存储器的 Bootloader 区域内，不能在 OTA 固件升级期间修改它。因此，即使没有有效的应用镜像，Bootloader 仍被连续重载，并能够进行 OTA 升级。固定堆栈 OTA Bootloader 适用于赛普拉斯可编程 BLE 器件。

4.3 可升级堆栈 OTA Bootloader

可升级堆栈 OTA Bootloader 使用双应用镜像结构。在该结构中，可用的闪存分成三部分：启动程序与复制程序镜像（简称为启动程序）、堆栈应用镜像和用户应用镜像（请参考图 5）。

启动程序镜像将启动堆栈应用或用户应用，具体取决于镜像的标志和有效性。更新堆栈应用时，启动程序也将最终的堆栈应用镜像（通过 BLE 链接下载并保存在临时位置（用户应用）中）复制到闪存的堆栈应用位置。

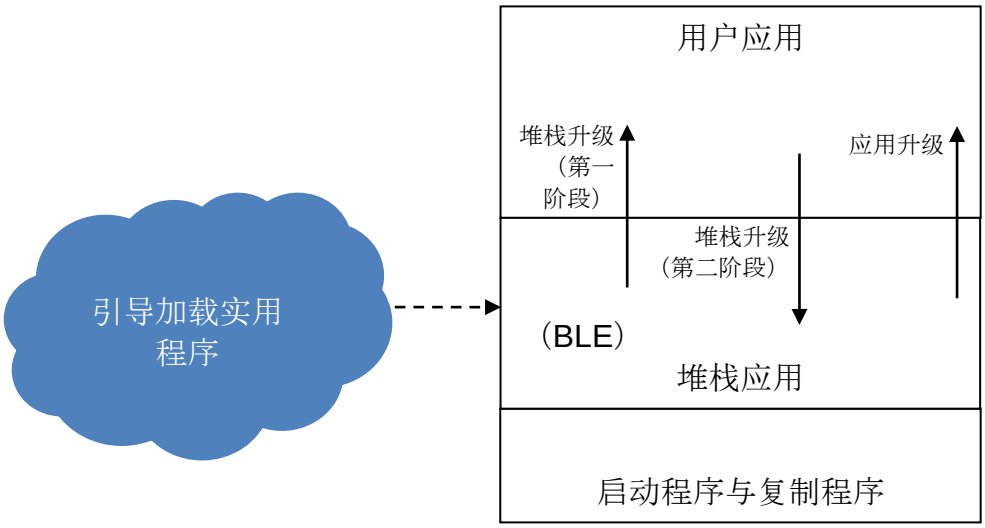
堆栈应用镜像包含 BLE 堆栈或所有堆栈应用镜像和用户应用镜像共享的其它代码。堆栈应用负责执行用户应用镜像的升级操作。它还下载了新版堆栈应用镜像，然后将其暂时存储在闪存的用户应用区域内。

用户应用将执行心率传感器、遥控器或鼠标等终端应用功能。它会链接到堆栈应用镜像中的 BLE API（请参考 BLE 组件数据手册）。这样，堆栈应用和用户应用可以共享堆栈（和/或该区域中的所有其它代码）。

理想情况下，堆栈应用和用户应用镜像（在两个 PSoC Creator 项目中实现）应有各自的 BLE 堆栈副本；但是，这样会占用更大的存储器空间。由于共享堆栈（和/或所有其它代码），堆栈应用和用户应用可以复用与 BLE 堆栈相关的代码。这样可以节省相当大的闪存容量，以供应用镜像使用。该结构提供了两个固件升级选项，如下所述。

- **应用升级：**只对用户应用镜像进行升级。应用升级操作只需一个阶段。要想进入 OTA 升级模式，固件需要将控制权转给堆栈应用，它会接收新的用户应用镜像。然后，堆栈应用将新接收的用户应用镜像直接写入到闪存的相应区域内（如图 5 所示）。
- **堆栈升级：**对堆栈应用和用户应用都被升级。
  - 第一阶段：固件将控制权转给堆栈应用，它会接收新的堆栈应用镜像，然后将其写入到闪存存储器中的临时位置（用户应用区域）。在该过程中，用户应用受到损坏（新接收到的堆栈应用镜像覆盖掉了现有的用户应用镜像）。
  - 第二阶段 — 下载完成后，堆栈应用将启动软件复位，并将控制权转给启动程序镜像。它会检测临时位置（用户应用区域）中的镜像，然后将其复制到堆栈应用区域中（请参考图 5）。此时，用户应用镜像受到损坏，因此需要执行应用升级。

图 5. 可升级堆栈 OTA 引导加载流程



优点

- 可以灵活更新 BLE 堆栈和应用镜像
- BLE 堆栈被复用，减少了所占用的闪存存储器空间。
- 升级时间变得更短，这是因为所收到的 BLE 堆栈/应用镜像被直接写入到闪存存储器内
- 不需要使用外部存储器。

缺点

- 需要使用容量更大（256 KB）的闪存，以便在 BLE 堆栈升级期间存储新的 BLE 堆栈副本。

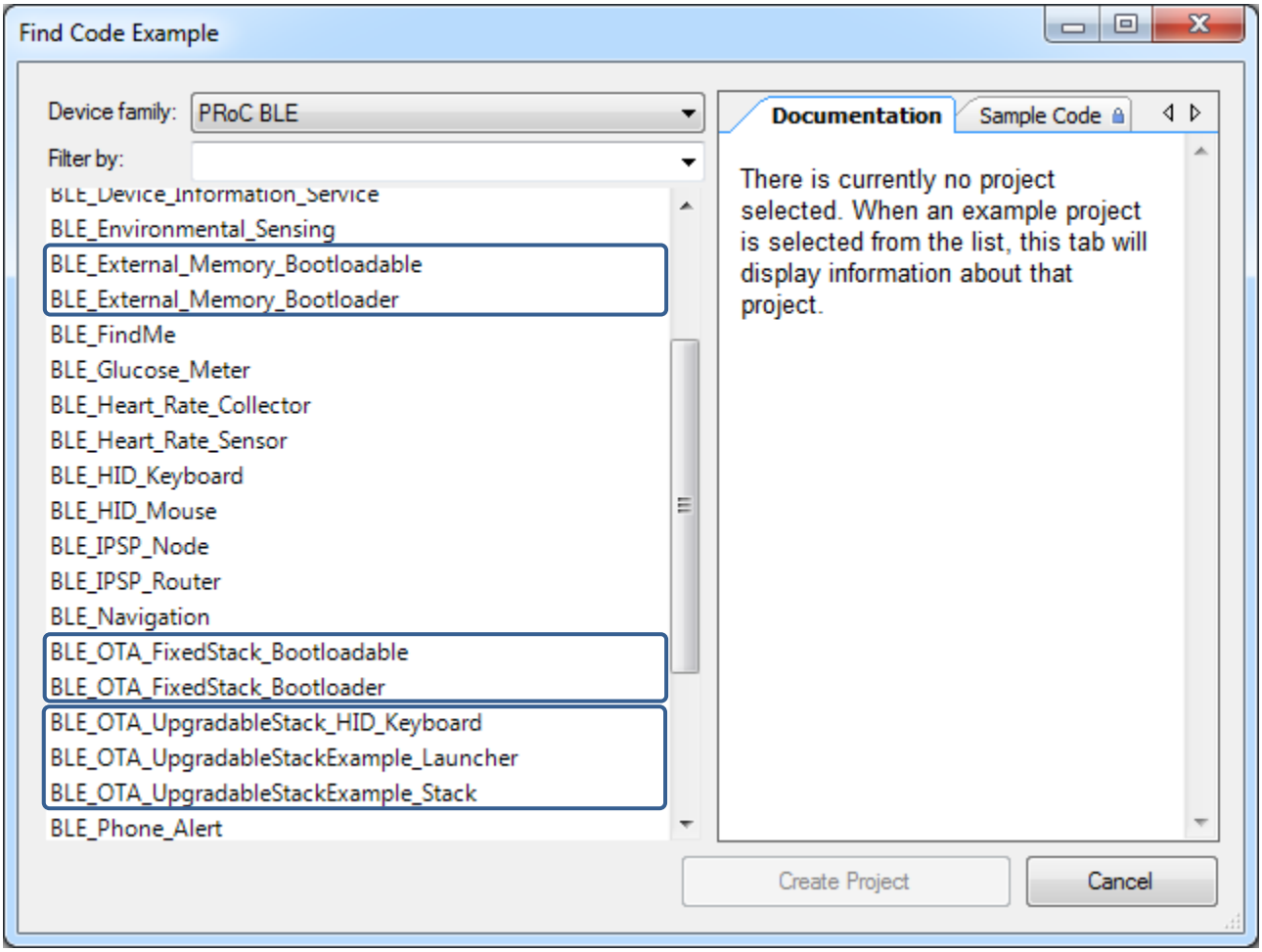
通过可升级堆栈 OTA，可以实现 BLE 堆栈和 Bootloadable/应用镜像的升级操作。另外，要暂时将 BLE 堆栈镜像（在下载期间）存储在闪存中，这样便要求容量更大的闪存。因此，可升级堆栈 OTA Bootloader 的方法仅适用于赛普拉斯的 256 KB BLE 器件。

5 为目标项目添加固件 OTA Bootloader 支持

本部分介绍了如何将一个 OTA Bootloader 添加到目标项目内。要想实现该操作，要创建一个示例目标项目，然后向该目标项目中添加一个 OTA Bootloader（将逐步介绍添加三种 OTA Bootloader 的各个步骤）。

PSoC Creator 提供了使能 OTA 功能的示例项目；通过转到 **File** > **Example Project...**，然后选择合适的 **Device Family** 和 **Filter by** 关键词，可以访问这些示例项目，如图 6 所示。每个示例项目都有所附的文档。请查阅文档，了解项目特定的执行信息。本部分描述了如何向非 OTA 应用固件添加 OTA 功能。请查阅 BLE OTA Bootloader，了解有关各种 Bootloader 的详细信息。

图 6. BLE 示例项目



5.1 创建基本的示例目标项目

添加 OTA Bootloader 前，先要创建一个基本的示例项目。为了创建示例项目，首先要使用‘PWMExample’项目。该项目使用 PRoC BLE/PSoC 4 BLE 中的 PWM 组件对 LED 亮度进行控制。请按照以下步骤为 PSoC 4 BLE/PRoC BLE 器件创建 ‘PWMExample’项目。如果您已经有了需要添加 OTA Bootloader 的项目，请跳过本部分。

1. 在 PSoC Creator 中，依次选择 **File** > **Code Example ...**。这样会启动 **Find Example Project** 对话框，如图 7 所示。
2. 在 **Find Example Project** 对话框中，将 **Device Family** 滤波器设置为 **PSoC 4200 BLE**，并将 **Filter by** 关键词设置为 **PWMExample**，如图 7 所示。

3. 从列表中选择 **PWMExample** 项目，然后点击 **Create New Project**。
4. 在 **Workspace** 字段中，选择 **Create New Workspace** 项。指定新示例项目工作区的位置（请参考图 8），然后点击 **Finish**。

图 7. Find Example Project 对话框

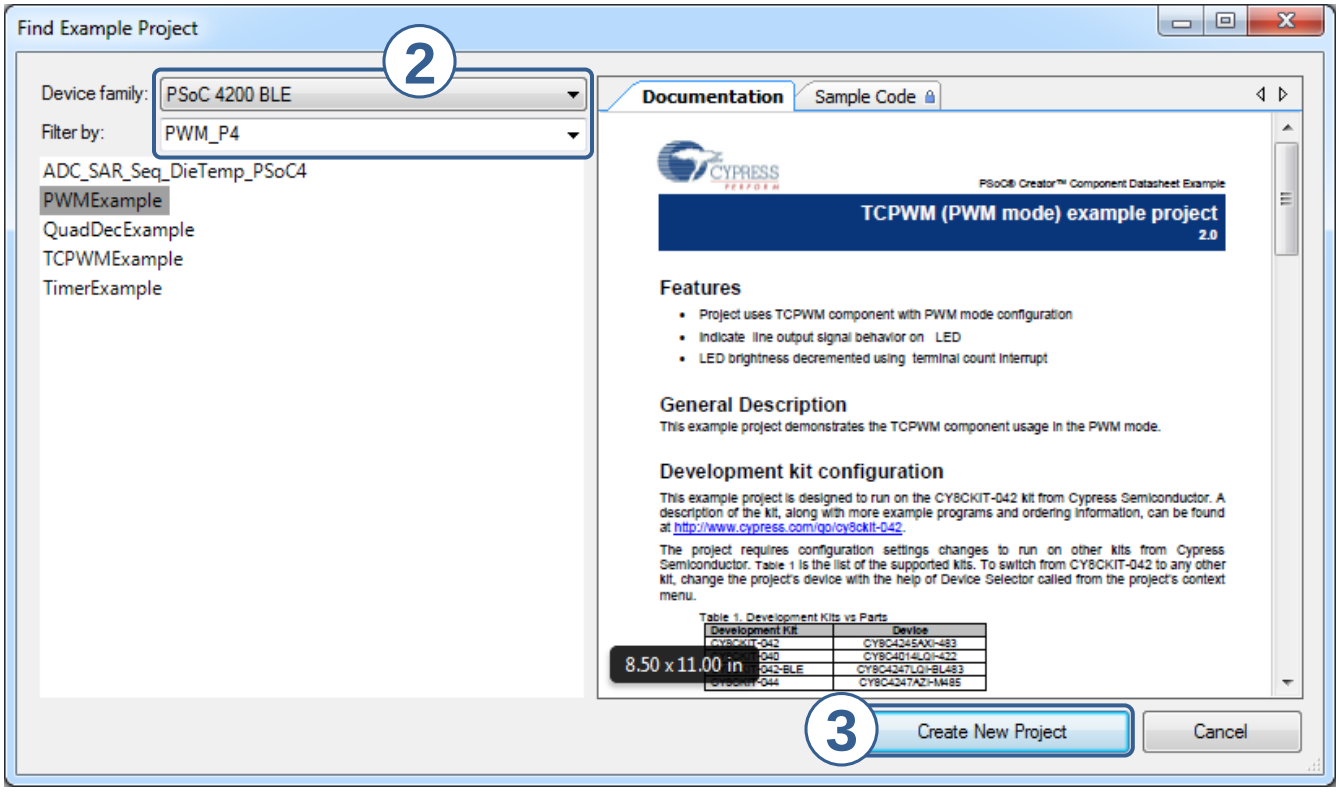
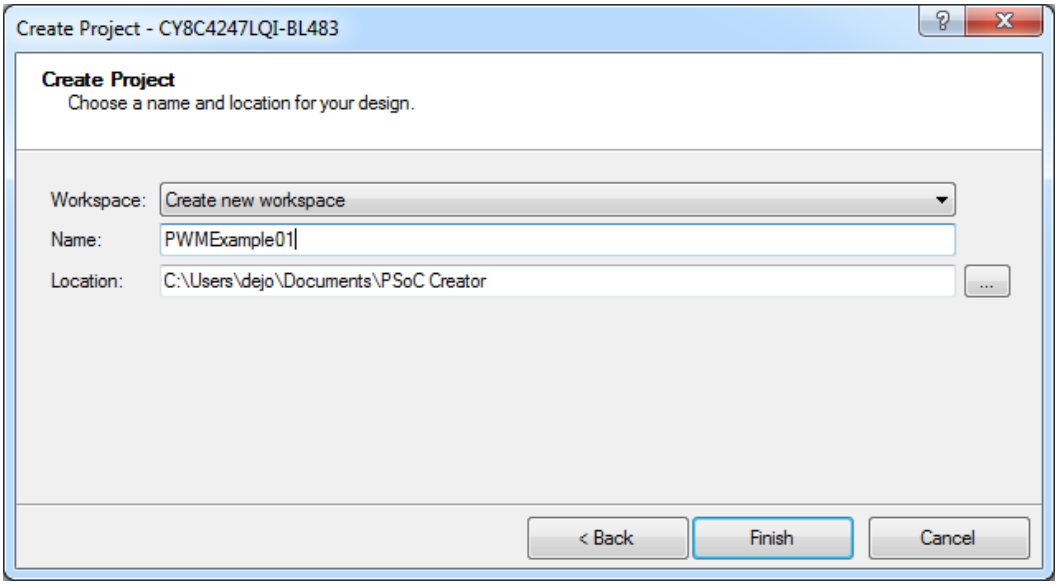


图 8. Create Project 对话框



5. 双击 **Workspace Explorer** 中的文件，以打开 **PWMExample01.cydwr** 窗口。将 LED_GREEN 的分配管脚从默认更改为 P3[6]（请参考表 1）。需要更改引脚分配，使之适用于该项目以及 CY8CKIT-042-BLE Pioneer 套件。端口 P3[6]连接着套件上的绿色 LED。

   如果需要更改/选择正确的目标应用器件，请按照选择其它器件部分介绍的指导进行操作。

6. 将 **LFCLK** 时钟源设置为 **WCO（32.768 kHz）**。

   a. 请打开 **PWMExample01.cydwr** 窗口，转到 **Clocks** 选项卡，然后点击 **Edit Clock…**，以打开 **Configure System Clocks** 对话框。

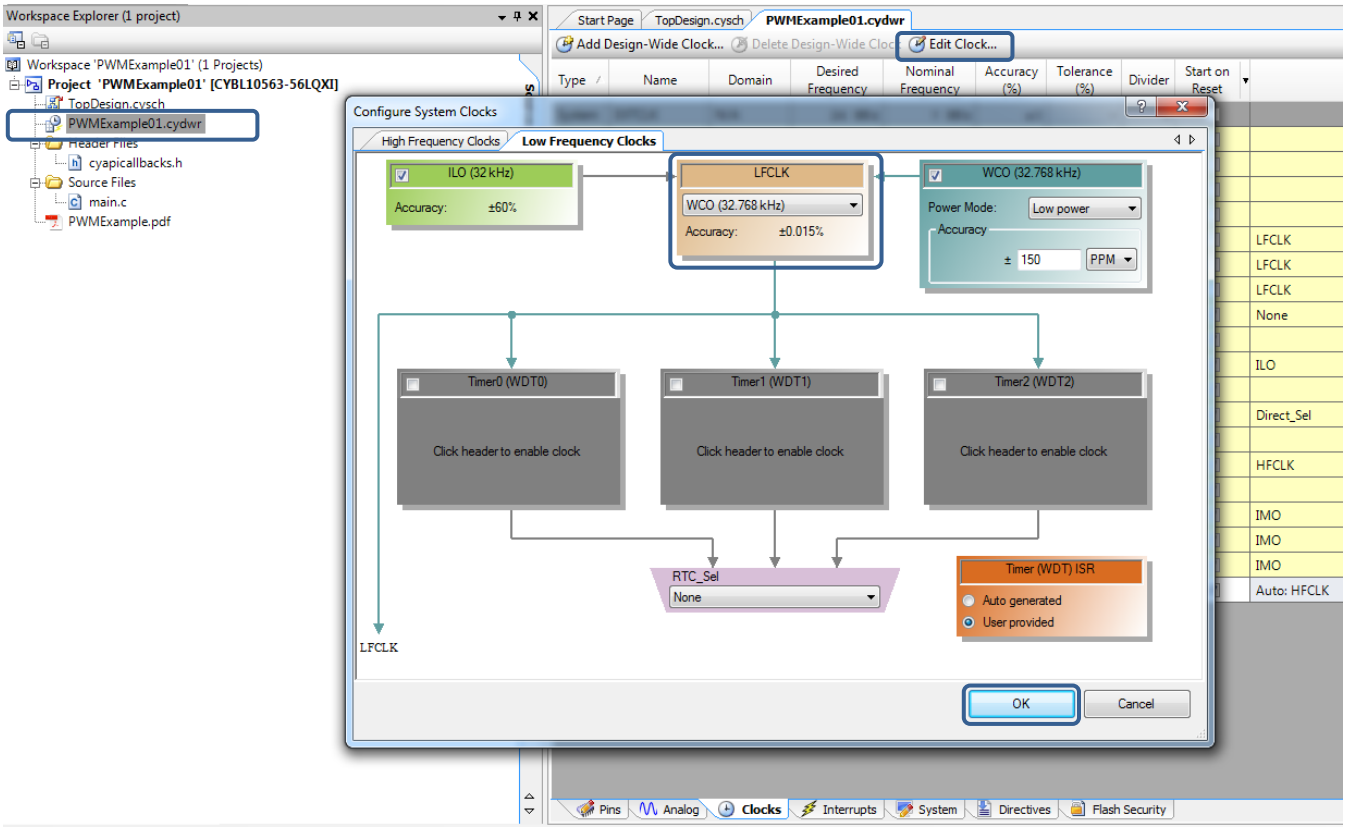
   b. 在 **Configure System Clocks** 对话框中，转到 **Low Frequency Clocks** 选项卡以修改 LFCLK 时钟源（请参考图 9）。

7. 保存项目。

表 1. PWMExample01 的引脚映射

引脚名称	端口分配
LED_GREEN	P3[6]

图 9. LFCLK 选择



这样您便设置好了一个基本的 PWM 示例项目，它可以在目标 BLE 器件上运行。您可以编译项目，并将生成的镜像文件编程目标器件（选择 **Debug** > **Program** 以同时进行这两项操作），了解该示例项目的工作原理。要想进一步了解有关 PSoC Creator 和编程 CY8CKIT-042-BLE Pioneer 套件的信息，请查阅 AN91267 和 AN94020。

通过更改 *main.c* 中 BRIGHTNESS_DECREASE 宏的值，可以控制 LED 的状态。该宏值的范围为 0 到 63000。该值越小，亮度调光周期速率越慢；相反，该值越大，亮度调光周期速率越快。

5.2 添加外部存储器 OTA Bootloader

本部分描述了如何为上面的创建基本的示例目标项目一节中所创建好的 PWMExample 项目添加一个外部存储器 OTA Bootloader。您也需要参考 BLE 外部存储器 Bootloader 和 Bootloadable 示例项目的数据手册和源代码。

在 CY8CKIT-042-BLE Pioneer 套件上，将一个基于 I²C 的 F-RAM 作为外部存储器使用。然而，外部存储器 OTA Bootloader 可适用于其它类别的存储器，如闪存（这些存储器使用 I²C 或 SPI 接口）。可以在此处查看基于 SPI 的外部存储器 OTA Bootloader 的示例。

按照以下步骤，为 PSoC 4 BLE Pioneer 套件设置 PWMExample 项目中基于 I²C 的外部存储器 Bootloader。

1. 在 PSoC Creator 中打开创建的 PWMExample01 项目（请查看创建基本的示例目标项目一节）。
2. 向 PWMExample01 工作区内添加 BLE 外部存储器 Bootloader 示例项目（请参考为现有工作区添加一个示例项目）。
3. 右键点击 **Workspace Explorer** 中的项目，然后选择 **Set As Active Project** 项，将 BLE_External_Memory_Bootloader01 项目设置为有效项目。
4. 如果需要更改/选择正确的目标应用器件，请按照选择其它器件部分介绍的指导进行操作。
5. 通过依次选择 **Build** > **Build BLE_External_Memory_Bootloader01**，编译 BLE_External_Memory_Bootloader01 项目。该项目应在无错误的情况下完成编译流程。
6. 打开 PSoC Creator 的其它实例，并且为 BLE 外部存储器 Bootloadable 示例项目创建新的工作区（查看创建一个示例项目工作区）。您需要正确配置该项目中的文件和代码段来使能外部存储器 Bootloader。
7. 将 PWMExample01 项目设置为有效项目。
8. 将以下组件从 BLE 外部存储器 Bootloadable 项目（在 6 中所创建）的 *TopDesign.cysch* 中复制到 PWMExample01 项目（在创建基本的示例目标项目一节中所创建）的 *TopDesign.cysch* 中。BLE 外部存储器 Bootloadable 示例项目的数据手册中介绍了这些组件的配置情况。

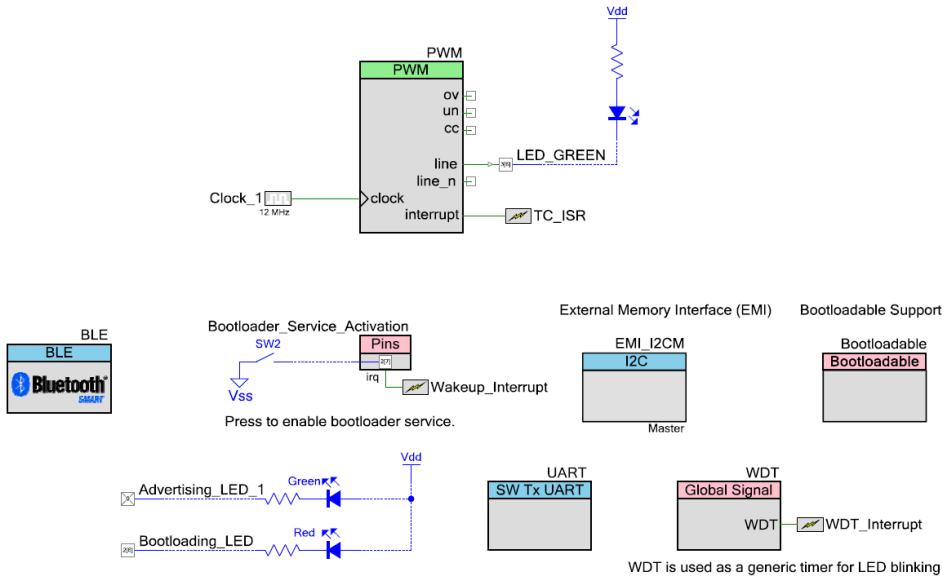
   - BLE
   - Bootloader_Service_Activation
   - Bootloading_LED
   - Advertising_LED_1
   - Bootloadable
   - EMI_I2CM
   - UART
   - WDT
   - WDT_Interrupt
   - Wake_Interrupt

   如果您选择的项目已经包含了 BLE 组件，那么请忽略该步骤中有关 BLE 组件的操作。而是按照添加 Bootloader 服务一节中所述的指导为现有 BLE 组件添加一个 Bootloader 服务，然后对其进行配置。

   要想实现外部存储器 OTA，需要使用 BLE、EMI_I2CM 和 Bootloadable 组件。Bootloader_Service_Activation 和 Wakeup_Interrupt 组件用于触发进入 Bootloader 模式操作。如果您的项目拥有其它进入 Bootloader 模式的机制，那么请避免复制 Bootloader_Service_Activation 和 Wakeup_Interrupt 组件。UART 组件用于打印调试信息。所有其他组件并不重要，在该步骤可忽略它们。然而，需要使用这些组件来防止编译时错误。

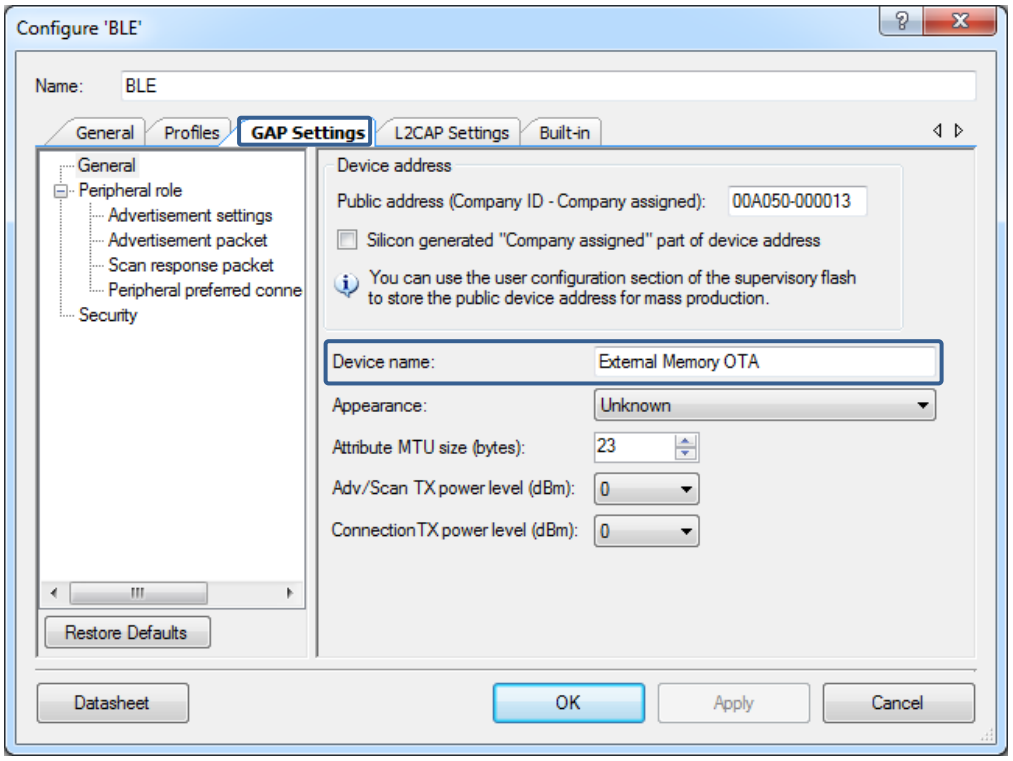
   完成该步骤时，您的原理图如图 10 所示。

图 10. 添加所需组件后 TopDesign.cysch 的视图



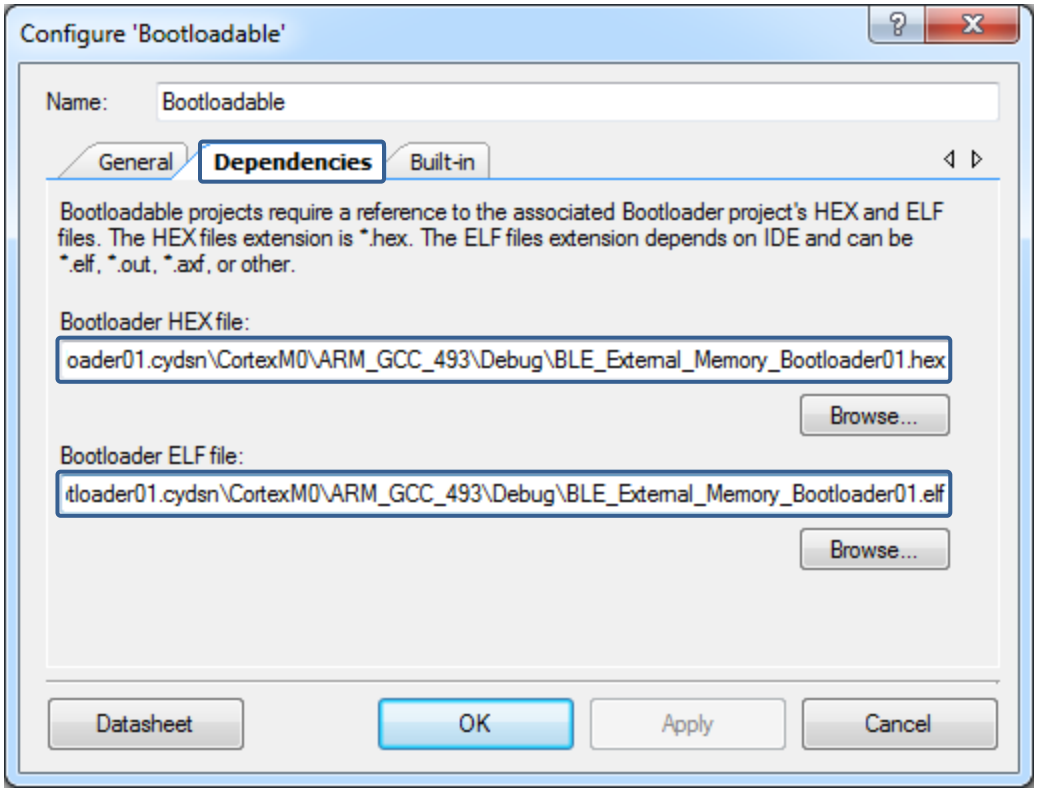
9. 在 BLE 组件配置窗口的 **GAP Settings** 选项卡中，将 **Device Name** 修改为“External Memory OTA”。请参考图 11。

图 11. BLE 组件配置窗口 — GAP Settings 选项卡



10. 指定 Bootloader 项目 HEX 和 ELF 文件的路径：
    a. 双击 Bootloadable 组件。
    b. 转到 **Dependencies** 选项卡，并将 **Bootloader HEX file** 链接到 *BLE_External_Memory_Bootloader01.hex* 文件（位于…*\BLE_External_Memory_Bootloader01.cydsn \CortexM0\<compiler version>\<build configuration>*\中），如图 12 所示。
    选定 HEX 文件后，会自动选择相应的 ELF 文件。
    c. 点击 **OK**，关闭 Bootloadable 组件配置对话框。

图 12. Bootloadable 组件配置



11. 如果需要为第 8 步骤中所添加的组件分配正确的引脚，请打开 *PWMExample01.cydwr* 窗口并转到 **Pins** 选项卡。按照表 2 配置各引脚。

表 2. PWMExample01 的引脚映射情况

引脚名称	端口分配
EMI_I2CM:scl	P5[1]
EMI_I2CM:sda	P5[0]
UART:tx	P1[5]
Advertising_LED_1	P3[7]
Bootloader_Service_Activation	P2[7]
Bootloading_LED	P2[6]
LED_GREEN	P3[6]

12. 将以下文件从 Bootloadable 项目工作区目录复制到 PWMExample01 项目工作区目录中，并且将其添加到 PWMExample01 项目中。这些文件用于执行部分 OTA 和调试功能。

a. 要想添加头文件，请右键点击 **Workspace Explorer** 中的 **Header Files** 文件夹，然后依次选择 **Add** > **Existing Item…**项。浏览并选择所需文件，然后点击 **Open**。

b. 要想添加源文件，请右键点击 **Workspace Explorer** 中的 **Source Files** 文件夹，然后依次选择 **Add** > **Existing Item…**项。浏览并选择所需文件，然后点击 **Open**。

- *Common.h*
- *debug.h*
- *main.h*
- *Options.h*
- *OTAMandatory.h*
- *OTAOptional.h*
- *Common.c*
- *debug.c*
- *OTAMandatory.c*
- *OTAOptional.c*

OTAMandatory.c/h 文件包含所有必需功能，从而使能外部存储器 OTA。所有其他文件都不必要的，可以复制它们，以便编译时不会发生错误。*debug.h/c* 文件用于打印基于 UART 的调试信息。*Common.c/h* 文件用于实现看门狗（WDT）、LED、调试信息打印和设置 Bootloader 服务可视性功能。*main.h* 文件包含 LED 状态和使能/禁用 Bootloader 服务的定义。*Options.h* 文件包含了用于使能/禁用调试和加密功能的定义。*OTAOptional.c/h* 文件用于对外部存储器内的信息实现加密和解密。通过在 *Options.h* 中将 `ENCRYPT_ENABLED` 宏设置为 `YES`，可以使能这些操作。

13. 必须将 BLE 外部存储器 Bootloadable 项目中的其它代码添加到 PWMExample01 项目中的 *main.c* 内，这样才能使能引导加载或 OTA 功能。更改的内容太多，不再逐一列出。但您可以下载并使用已修改文件。将 PWMExample01 项目的 *main.c* 文件替换为*…\Code\External Memory OTA*路径下的 *main.c* 文件。
14. 添加 Bootloader/Bootloadable 组件后，PSoC Creator 不再支持调试功能。然而，当固件开始执行时，通过连接至目标器件（依次选择 **Debug** > **Attach to Running Target…**）或使用 UART 信息，您也可以调试该固件。请确保 UART 被使能时，设置合适的堆（0x400 字节）和栈（0x800 字节）的大小，使项目能够正确操作。否则，固件会发生不可预知的行为。更多详细信息，请参考调试一节。
15. 依次选择 **Build** > **Build PWMExample01**，编译 PWMExample01 项目。PWMExample01 项目应在无错误的情况下完成编译流程。
16. 依次选择 **Debug** > **Program**，以编程 PSoC BLE Pioneer 套件。

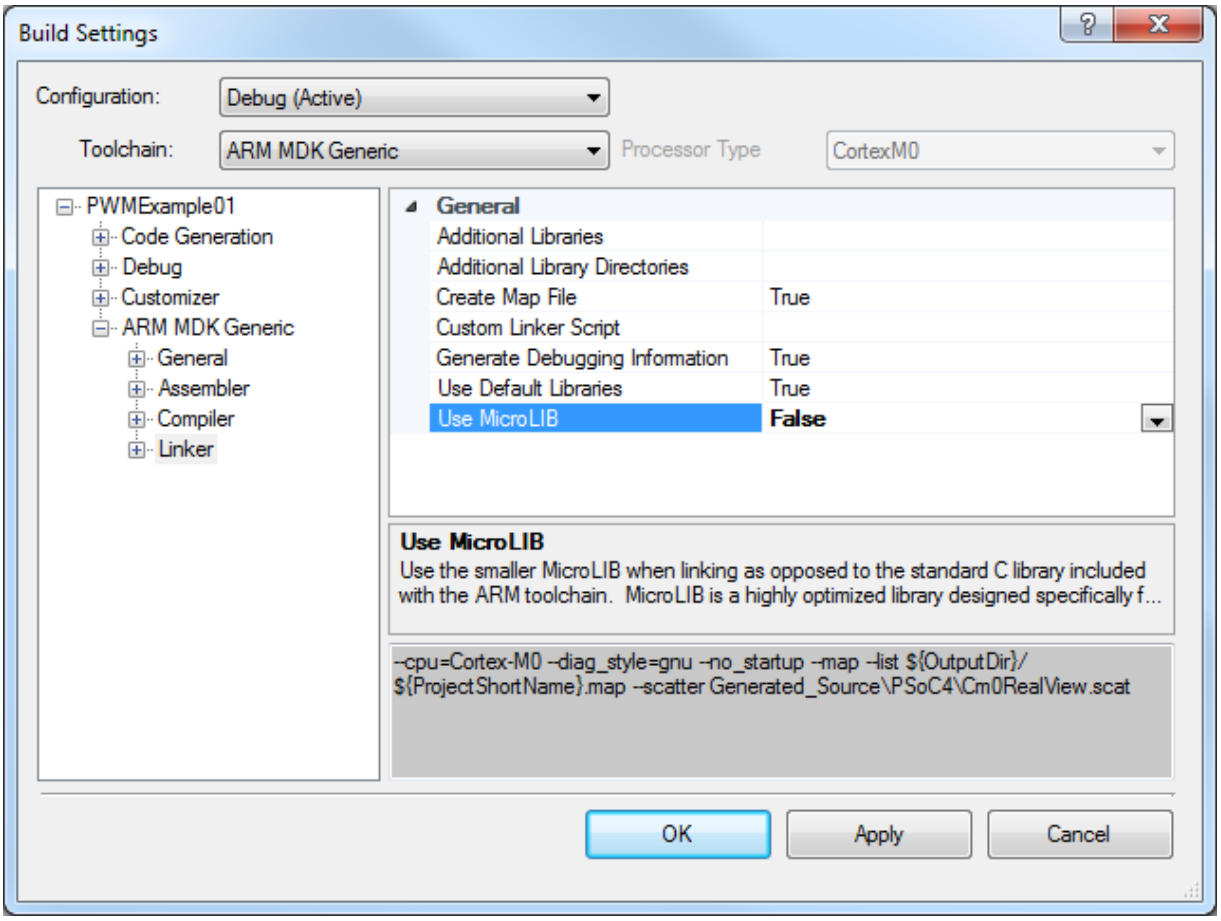
编程完成后，绿色 LED 将会从最高亮度等级到最低亮度等级循环变化。此时，您可以按照执行 OTA 升级一节所述的各种方法中的一种进行操作。另外，还可以按照测试 OTA 功能一节中所描述的各个步骤进行测试 OTA 功能。

5.3 添加固定堆栈 OTA Bootloader

本部分描述了如何为上述创建基本的示例目标项目一节中所创建的 PWMExample 项目添加一个固定堆栈 OTA Bootloader。您也需要参考 BLE 固定堆栈 Bootloader 和 Bootloadable 示例项目的数据手册和源代码。按照以下步骤设置 PWMExample 项目中的固定堆栈 Bootloader。

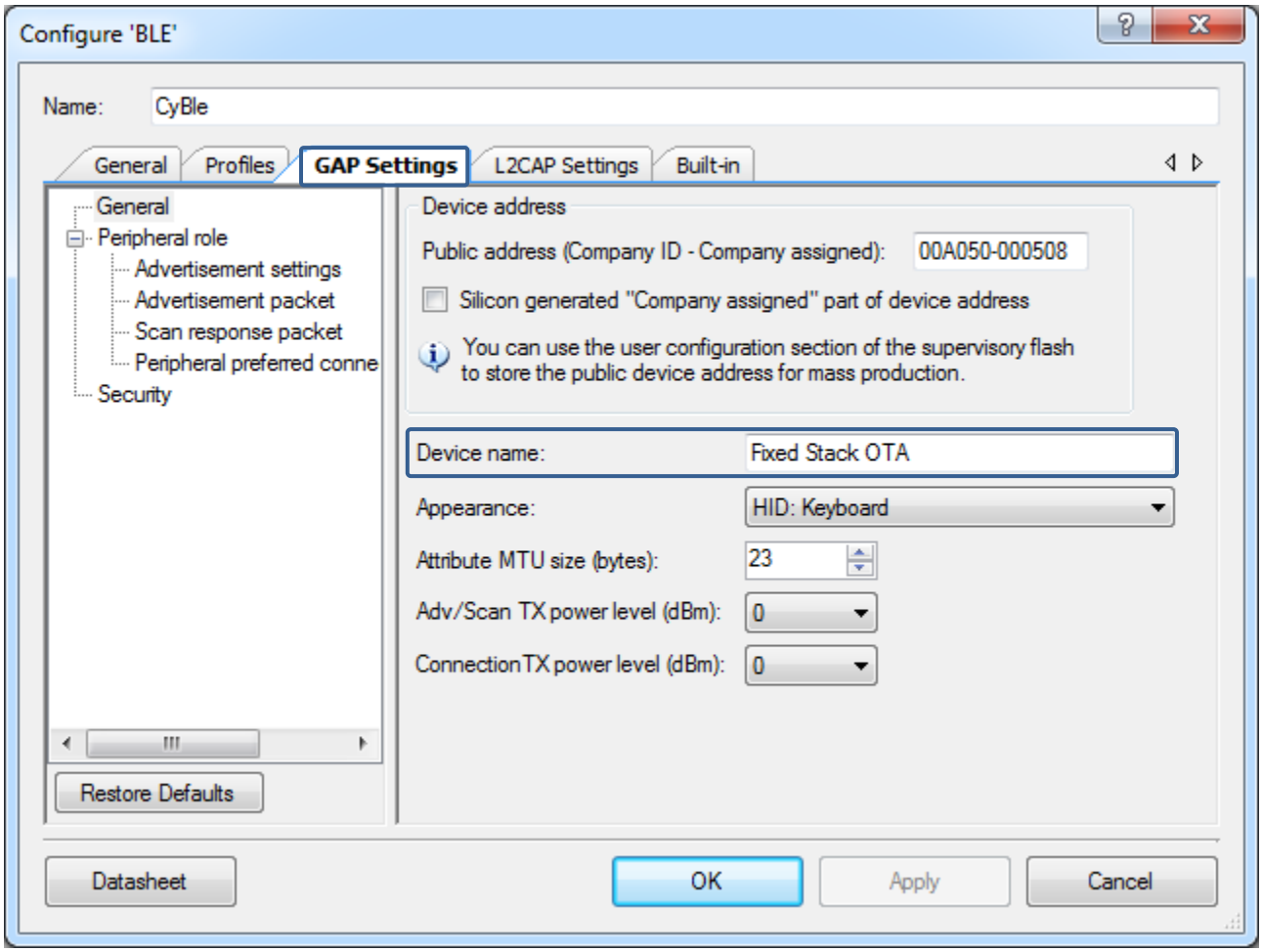
注意：当勾选 MDK MicroLIB linker 选项时，固定堆栈 OTA Bootloader 的代码共享性能不受支持。因此，在 **Project** > **Build Settings** 中，必须将 **Use MicroLIB** 设置为 **False**（请参考图 13）。

图 13. MDK 链接器的编译设置



1. 打开在 PSoC Creator 中创建的 PWMExample01 项目（请查看创建基本的示例目标项目一节）。
2. 为 PWMExample01 工作区添加 BLE_OTA_FixedStack_Bootloader 示例项目（请参考为现有工作区添加一个示例项目的内容）。
3. 通过右键点击 **Workspace Explorer** 中的项目，并选择 **Set As Active Project** 项，可将 BLE_OTA_FixedStack_Bootloader01 项目设置为有效项目。
4. 如果需要更改/选择正确的目标应用器件，请按照选择其它器件部分介绍的指导进行操作。
5. 在 BLE 组件配置窗口的 **GAP Settings** 选项卡中，将 **Device Name** 修改为“Fixed Stack OTA”（请参考图 14）。双击 BLE 组件，可以打开 BLE 组件配置对话框。

图 14. BLE 组件配置窗口 — GAP Settings 选项卡



如果您选定的 Bootloadable 项目已经包含了 BLE 组件，那么：

a. 将 BLE_OTA_FixedStack_Bootloader01 项目中现有的 BLE 组件替换为您 Bootloadable 项目中的 BLE 组件。

b. 为现有的 BLE 组件添加 Bootloader 服务（请查阅添加 Bootloader 服务），然后对其进行配置。

c. 从 Bootloadable 项目原理图中移除 BLE 组件。

通过替换 BLE 组件，根据新的 BLE 组件配置，必须移除或添加某些与服务/配置文件相关的代码，从而防止编译时错误。

6. BLE OTA 固定堆栈示例项目使用自定义链接脚本，并且必须根据选定器件对其进行配置。按照为其它赛普拉斯 BLE 器件配置固定堆栈 OTA 项目一节介绍的内容进行配置。
7. 编译 BLE_OTA_FixedStack_Bootloader01 项目。该项目应在无错误的情况下完成编译流程。
8. 在 *PWMExample01.cydsn* 项目文件夹中创建名称为“LinkerScripts”的新文件夹。
9. 在 PSoC Creator 中，双击 *mk.bat* 文件（位于 **Workspace Explorer** > **BLE_OTA_FixedStack_Bootloader01** > **Scripts** > **mk.bat** 路径下），以打开它。
10. 按表 3 的内容进行编辑，然后保存该文件。必须将 `LOADABLE_PRJ_NAME` 设置为 Bootloadable 的应用名称（这里为“PWMExample01”）。

表 3. *mk.bat* 文件中的更改内容

行编号	变量	数值
28	LOADER_PRJ_NAME	BLE_OTA_FixedStack_Bootloader01
30	LOADABLE_PRJ_NAME	PWMExample01

11. 从 **Windows Explorer** 中运行 *mk.bat* 文件。该文件位于*…\PWMExample01\BLE_OTA_FixedStack_Bootloader01.cydsn\Scripts* 路径下的位置。该批处理文件应该运行无误。批处理文件运行完成后，按下任意按键以关闭窗口。该步骤生成了 *BootloaderSymbolsGcc.ld* 文件，位于*…\PWMExample01\PWMExample01.cydsn\LinkerScripts* 路径下。
12. 打开 PSoC Creator 的其它实例，并且为 BLE_OTA_FixedStack_Bootloadable 示例项目创建新的工作区（查看创建一个示例项目工作区内容）。您需要使用该项目中的文件和代码段来使能固定堆栈 Bootloader。
13. 将 PWMExample01 项目设置为有效项目。
14. 在 PSoC Creator 的 **Workspace Explorer** 中，为 PWMExample01 项目创建名为“LinkerScripts”的新文件夹。要想创建新的文件夹，右键点击 **Workspace Explorer** 中的项目名称，并依次选择 **Add** > **New Folder**。
15. 第 12 步骤中所创建的 Bootloadable 示例项目具有一个名称为 LinkerScripts 的文件夹。该文件夹包含全部三种受 PSoC Creator 支持的编译器的链接脚本（*cm0gcc.ld*、*Cm0Iar.icf* 和 *Cm0Mdk.scat*）。将这三个文件复制到 PWMExample01 项目中第 8 步骤所创建的 LinkerScripts 文件夹内。
16. 将以下文件添加到 PSoC Creator PWMExample01 项目中位于 **Workspace Explorer** 内的 LinkerScripts 文件夹（在第 14 步骤中创建）。在文件系统中，这些文件位于在第 8 步骤中所创建的 LinkerScripts 文件夹内。
    - *BootloaderSymbolsGcc.ld*
    - *Cm0Iar.icf*
    - *cm0gcc.ld*
    - *Cm0Mdk.scat*
17. BLE OTA 固定堆栈示例项目使用自定义链接脚本，并且必须根据选定器件对其进行配置。按照为其它赛普拉斯 BLE 器件配置固定堆栈 OTA 项目一节介绍的内容进行配置。
18. 将以下组件从 BLE_OTA_FixedStack_Bootloadable 示例项目（第 12 步创建）的 *TopDesign.cysch* 中复制到 PWMExample01 项目（在创建基本的示例目标项目一节创建）的 *TopDesign.cysch* 中。这些组件的配置如 BLE 固定堆栈 Bootloadable 示例项目的数据手册中所述。这些组件的原理图可能包含几页。
    - Bootloadable
    - WDT
    - UART_DEB
    - WDT_Interrupt

    请确保最终的 Bootloadable 项目原理图不包含 BLE 组件。如果 Bootloadable 项目已经包含了 BLE 组件，那么：

    a. 将 BLE 组件移至 BLE_OTA_FixedStack_Bootloader01 项目内。

    b. 为现有的 BLE 组件添加 Bootloader 服务（请查阅添加 Bootloader 服务），然后对其进行配置。

    完成该步骤后，从第 5 步骤重复进行各个操作，以更新 Bootloader 并生成新的链接脚本。

    要想实现固定堆栈 OTA，需要使用 Bootloadable 组件。所有其他组件并不重要，在该步骤可忽略它们。然而，需要使用这些组件来防止编译时错误。UART 组件用于打印调试信息。如果需要，可使用 WDT 和 WDT_Interrupt 来实现项目中的时序。

完成该步骤后，您的原理图如图 15 所示。指定 Bootloader 项目 HEX 和 ELF 文件的路径。

a. 双击 Bootloadable 组件。

b. 转到 **Dependencies** 选项卡，并且将 **Bootloader HEX file** 链接到 *BLE_OTA_FixedStack_Bootloader01.hex* 文件（位于…*\BLE_OTA_FixedStack_Bootloader01.cydsn \CortexM0\<compiler version>\<build configuration>*），如图 16 所示。

c. 点击 **OK**，关闭 Bootloadable 组件配置对话框。

选定 HEX 文件后，会自动选择相应的 ELF 文件。

19. 如果需要为第 17 步骤中所添加的组件分配正确的引脚，请打开 *PWMExample01.cydwr* 窗口并转到 **Pins** 选项卡。按照表 4 配置各引脚。

图 15. 添加所需组件后 TopDesign.cysch 的视图

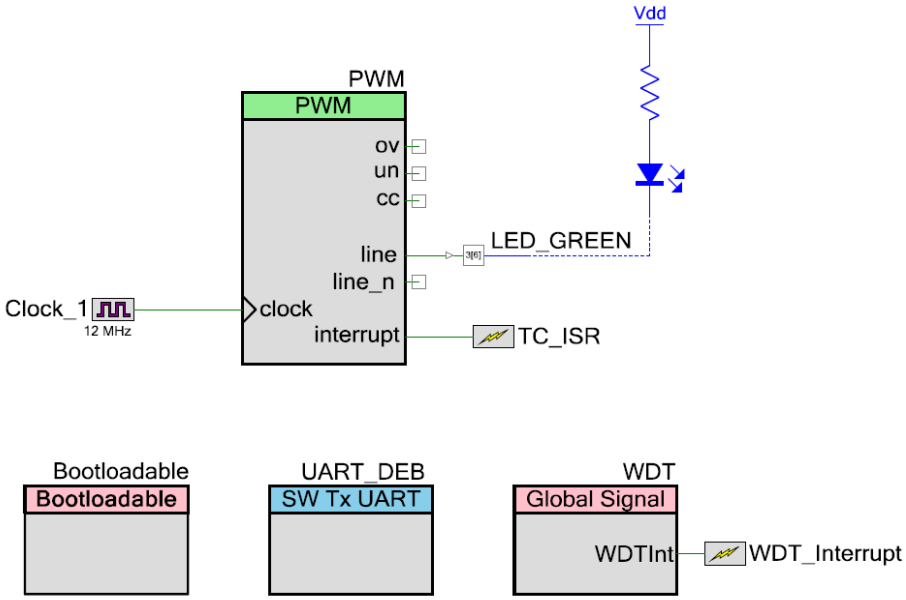


图 16. Bootloadable 组件配置

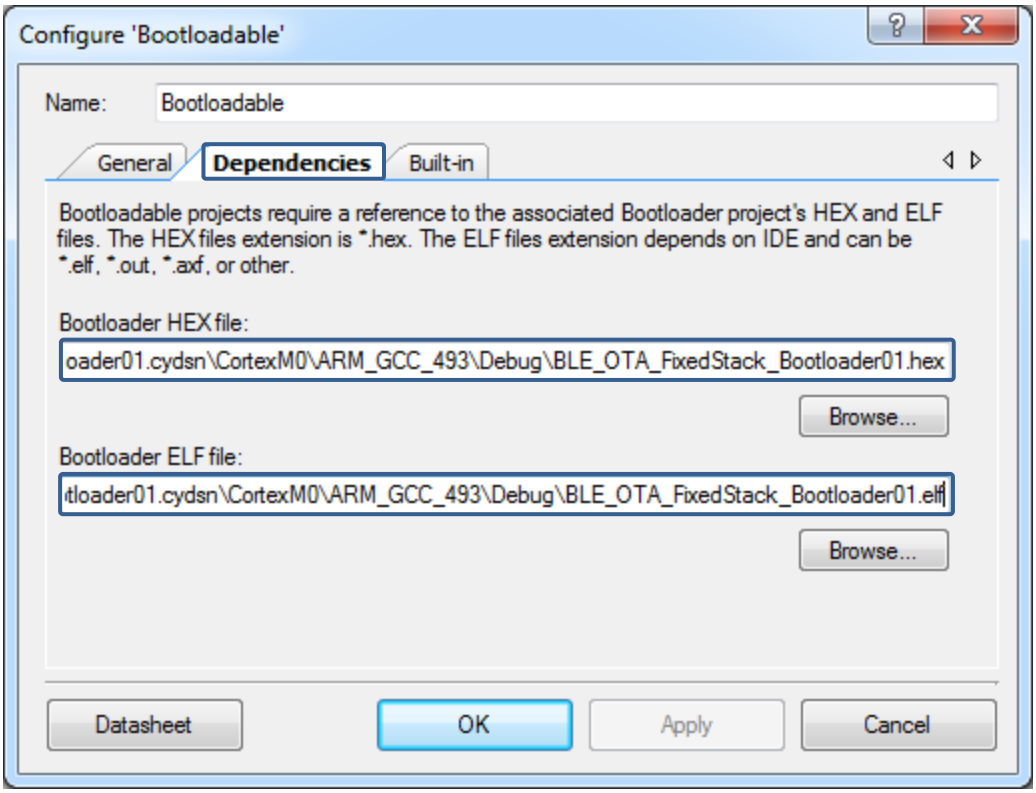


表 4. PWMExample01 的引脚映射情况

引脚名称	端口分配
UART_DEB:tx	P1[5]
LED_GREEN	P3[6]

20. 将 BLE 固定堆栈 Bootloadable 示例项目工作区目录中的以下文件复制到 PWMExample01 项目工作区目录中，并且将其添加到 PWMExample01 项目中。这些文件用于执行部分 OTA 和调试功能。

    a. 要想添加头文件，请右键点击 **Workspace Explorer** 中的 **Header Files** 文件夹，然后依次选择 **Add** > **Existing Item…**项。浏览并选择所需文件，然后点击 **Open**。

    b. 要想添加源文件，请右键点击 **Workspace Explorer** 中的 **Source Files** 文件夹，然后依次选择 **Add** > **Existing Item…**项。浏览并选择所需文件，然后点击 **Open**。

    - *common.h*
    - *main.h*
    - *OTAMandatory.h*
    - *OTAOptional.h*
    - *debug.h*
    - *Options.h*
    - *OTAMandatory.c*
    - *OTAOptional.c*
    - *debug.c*

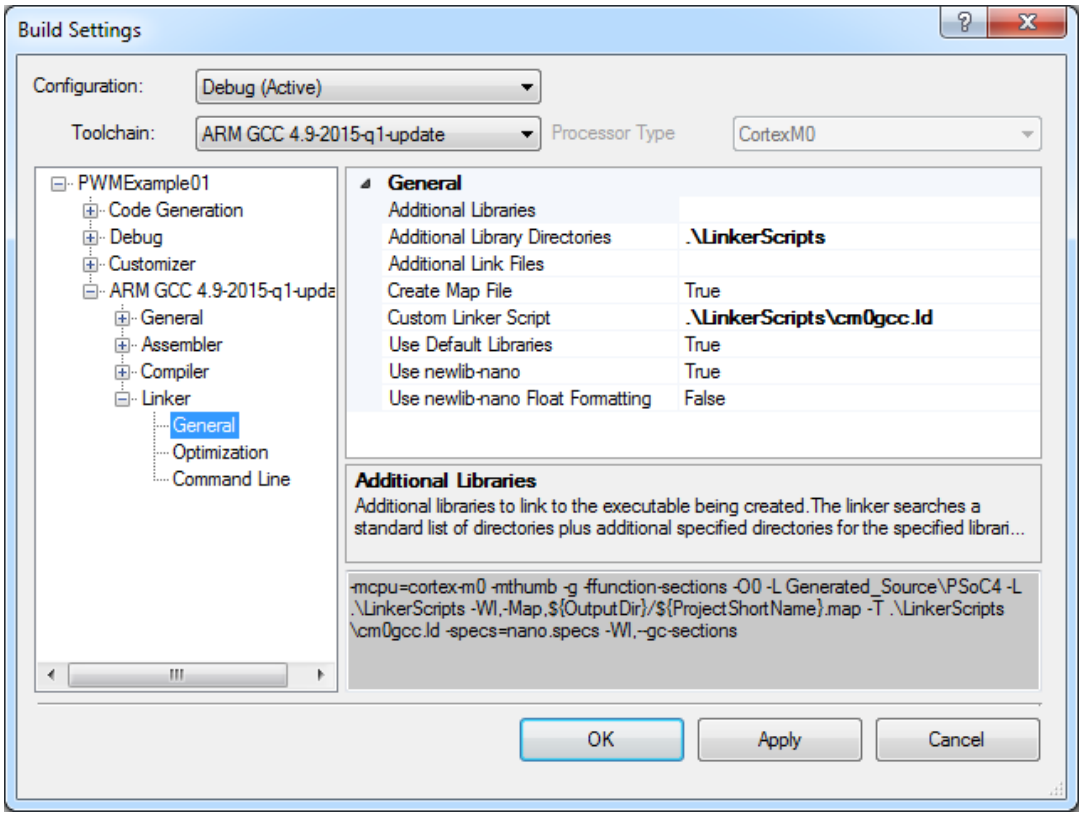
    OTAMandatory.c/h 文件执行所有所需功能，从而使能外部存储器 OTA。所有其他文件都不必要的，可以复制它们，以便编译时不会发生错误。*debug.h/c* 文件用于打印基于 UART 的调试信息。*common.h* 文件包含了 LED 状态和 WDT 选项的定义。*Options.h* 文件用于使能/禁用调试功能。*OTAOptional.c/h* 文件有助于实现 WDT、LED 和调试信息打印。

21. 必须将 BLE_OTA_FixedStack_Bootloadable 项目中的其它代码添加到 PWMExample01 项目中的 *main.c* 内才能使能引导加载或 OTA 功能。更改的内容太多，不再逐一列出。但您可以下载并使用已修改文件。将 PWMExample01 项目的 *main.c* 文件替换为*…\Code\Fixed Stack OTA*路径下的 *main.c* 文件。
22. 按照表 5 所述的内容配置 PWMExample01 项目的编译设置。这些更改允许链接器使用新的自定义链接脚本。要想修改编译设置，依次选择 **Project** > **Build Settings…**，然后在树形视图中依次选择 **PWMExample01** > **ARM GCC 4.9-2015-q1-update** > **Linker** > **General**，如图 17 所示。

表 5. 编译设置更改

字段	数值
附加库目录	.\LinkerScripts
自定义链接脚本	.\LinkerScripts\cm0gcc.ld

图 17. Build Settings 对话框 — Linker 设置



23. 添加 Bootloader/Bootloadable 组件后，PSoC Creator 不再支持调试功能。然而，当固件开始执行时，通过连接至目标器件（**Debug** > **Attach to Running Target…**）或使用 UART 信息，您也可以调试该固件。如果需要使能或禁用 UART 调试功能，请查阅调试一节。请确保 UART 被使能时，设置合适的堆（0x400 字节）和栈（0x800 字节）的大小，使项目能够正确操作。否则，固件会发生不可预知的行为。更多详细信息，请参考调试一节。
24. 编译 PWMExample01 项目。项目应在无错误的情况下完成编译流程。
25. 依次选择 **Debug** > **Program**，以编程 PSoC BLE Pioneer 套件。

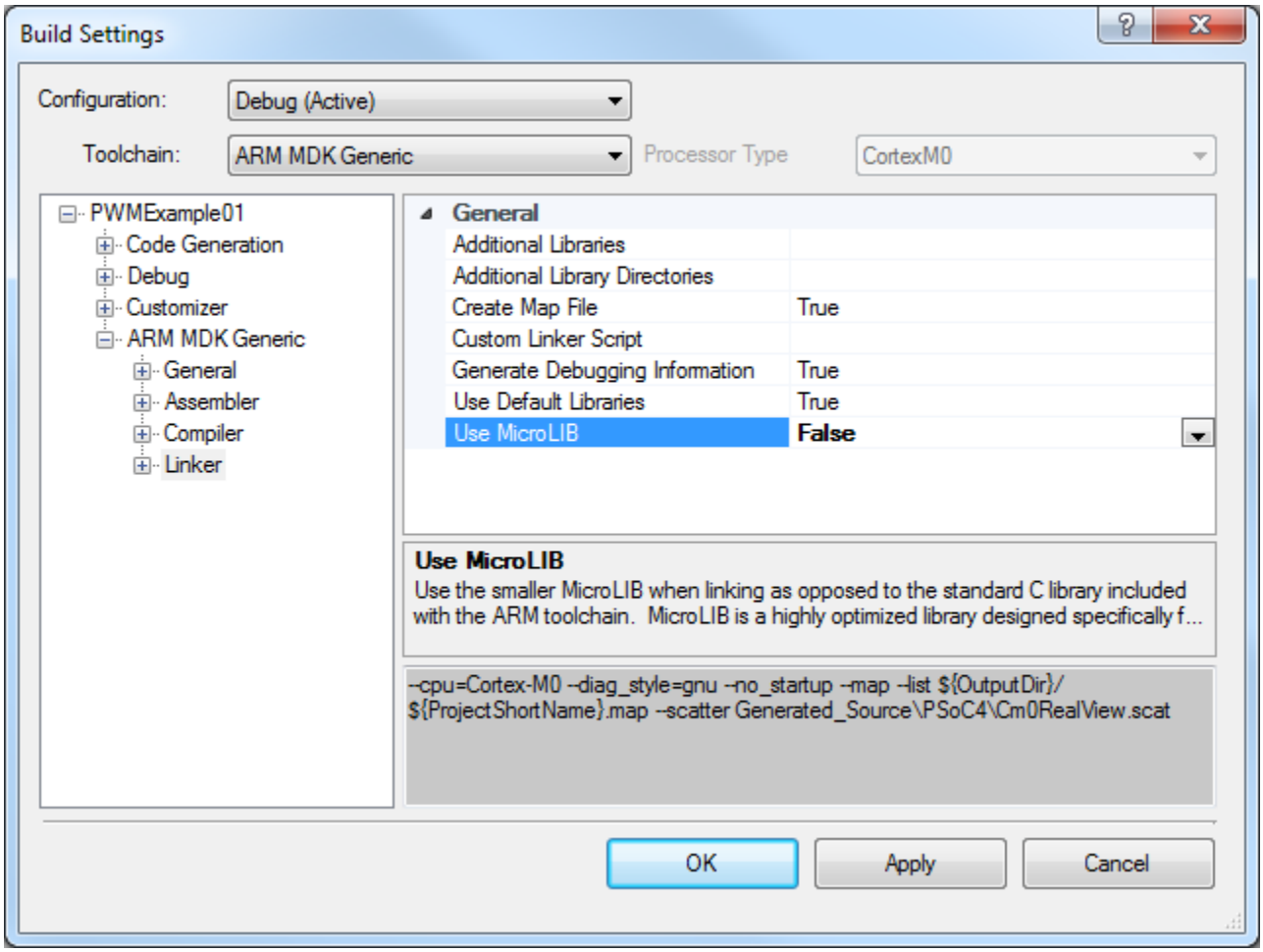
编程完成后，绿色 LED 将会从最高亮度等级到最低亮度等级循环变化。此时，您可以按照执行 OTA 升级一节所述的各种方法中的一种进行操作。另外，您还可以按照测试 OTA 功能一节中所描述的各个步骤进行测试 OTA 功能。

5.4 添加可升级堆栈 OTA Bootloader

本部分描述了如何为上述的创建基本的示例目标项目一节所创建的 PWMExample 项目添加一个可升级堆栈 OTA Bootloader。您也需要检查 BLE OTA 可升级堆栈启动程序、堆栈和键盘示例项目数据手册和源代码。按照以下步骤对 PWMExample 项目中的可升级堆栈存储器 Bootloader 进行设置。

注意：当勾选 MDK MicroLIB linker 选项时，可升级堆栈 OTA Bootloader 的代码共享性能不受支持。因此，在 **Project** > **Build Settings** 中，必须将 **Use MicroLIB** 设置为 **False**（请参考图 18）。

图 18. MDK 链接器的编译设置

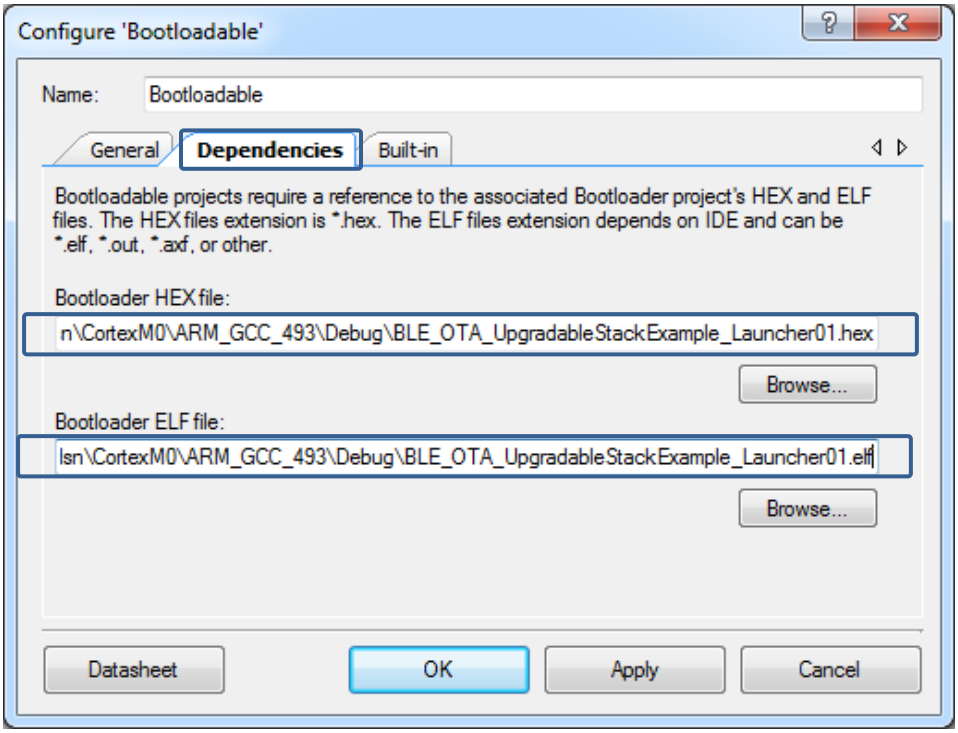


1. 打开在 PSoC Creator 中所创建的 PWMExample01 项目（请查看创建基本的示例目标项目一节）。
2. 为 PWMExample01 工作区添加 BLE_OTA_UpgradableStackExample_Launcher 示例项目（请参考为现有工作区添加一个示例项目中的内容）。
3. 将 BLE_OTA_UpgradableStackExample_Launcher01 项目设置为有效项目。要想实现该操作，请右击 **Workspace Explorer** 中的项目，并选择 **Set As Active Project**。
4. 如果需要更改/选择正确的目标应用器件，请按照选择其它器件部分中介绍的指导进行操作。
5. 对 BLE_OTA_UpgradableStackExample_Launcher01 项目进行编译。项目应在无错误的情况下完成编译流程。
6. 为 PWMExample01 工作区添加 BLE_OTA_UpgradableStackExample_Stack 示例项目（请参考为现有工作区添加一个示例项目中的内容）。
7. 将 BLE_OTA_UpgradableStackExample_Stack01 项目设置为有效项目。

8. 如果需要更改/选择正确的目标应用器件，请按照选择其它器件部分中介绍的指导进行操作。
9. 指定启动程序项目 HEX 和 ELF 文件的路径。
   a. 双击 Bootloadable 组件。
   b. 转到 **Dependencies**（依赖属性）选项卡，并且将 **Bootloader HEX file** 链接到 *BLE_OTA_UpgradableStackExample_Launcher01.hex* 文件（位于…*\BLE_OTA_UpgradableStackExample_Launcher01.cydsn \CortexM0\<compiler version>\<build configuration>*），如图 19 所示。
   c. 点击 **OK**，关闭 Bootloadable 组件配置对话框。

   选定 HEX 文件后，会自动选择相应的 ELF 文件。

图 19. Bootloadable 组件配置

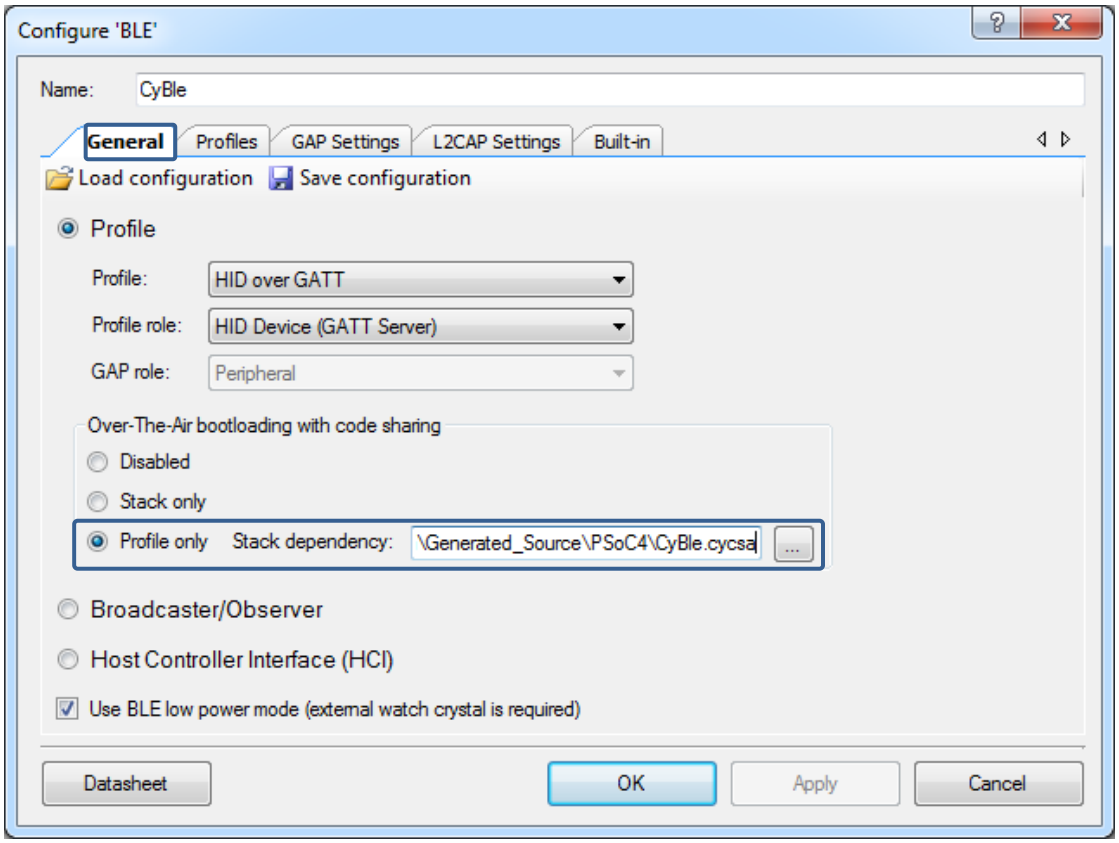


10. 对 BLE_OTA_UpgradableStackExample_Stack01 项目进行编译操作。项目应在无错误的情况下完成编译流程。
11. 打开 PSoC Creator 的其它实例，并且为 BLE_OTA_UpgradableStack_HID_Keyboard 示例项目创建新的工作区（查看创建一个示例项目工作区中的内容）。您需要使用该项目中的文件和代码段来使能可升级堆栈 OTA Bootloader。
12. 将 PWMExample01 项目设置为有效项目。
13. 将以下组件从 BLE_OTA_UpgradableStack_HID_Keyboard 示例项目（在第 23 步中所创建）的 TopDesign.cysch 文件中复制到 PWMExample01 项目（在*创建基本的示例目标项目*一节创建）的 TopDesign.cysch 文件中创建基本的示例目标项目这些组件的配置如 BLE_OTA_UpgradableStack_HID_Keyboard 示例项目的数据手册所述。

    - CyBle
    - Bootloadable
    - UART
    - SW2
    - Wakeup_Interrupt

    如果您选定的项目已经包含了一个 BLE 组件，请确保这是最新版本，并且将其配置为 **Profile only** 模式（请参考图 20）。

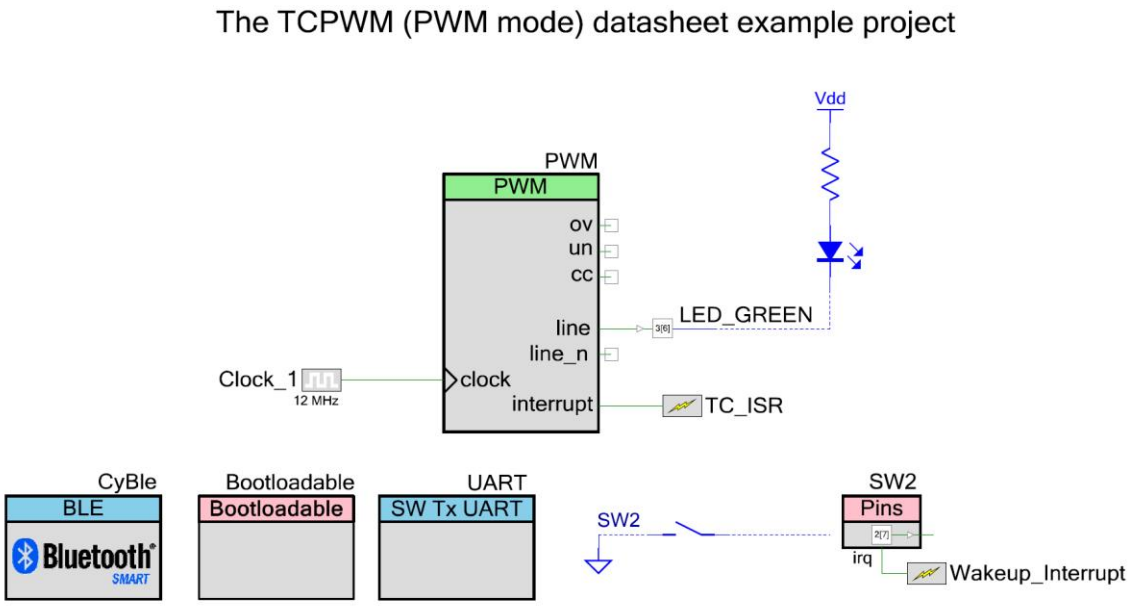
图 20. BLE 组件配置对话框 — 选择‘Profile Only’项



要想实现可升级堆栈 OTA，需要使用 BLE 和 Bootloadable 组件。所有其他组件并不重要，在该步骤可忽略它们。然而，需要使用这些组件来防止编译时错误。UART 组件用于打印调试信息。SW2 和 Wakeup_Interrupt 组件用于触发进入 Bootloader 模式操作。如果您的项目拥有进入 Bootloader 模式的其他机制，请勿复制 SW2 和 Wakeup_Interrupt 组件。

完成该步骤后，您的原理图如图 21 所示。

图 21. 添加所需组件后 TopDesign.cysch 的视图

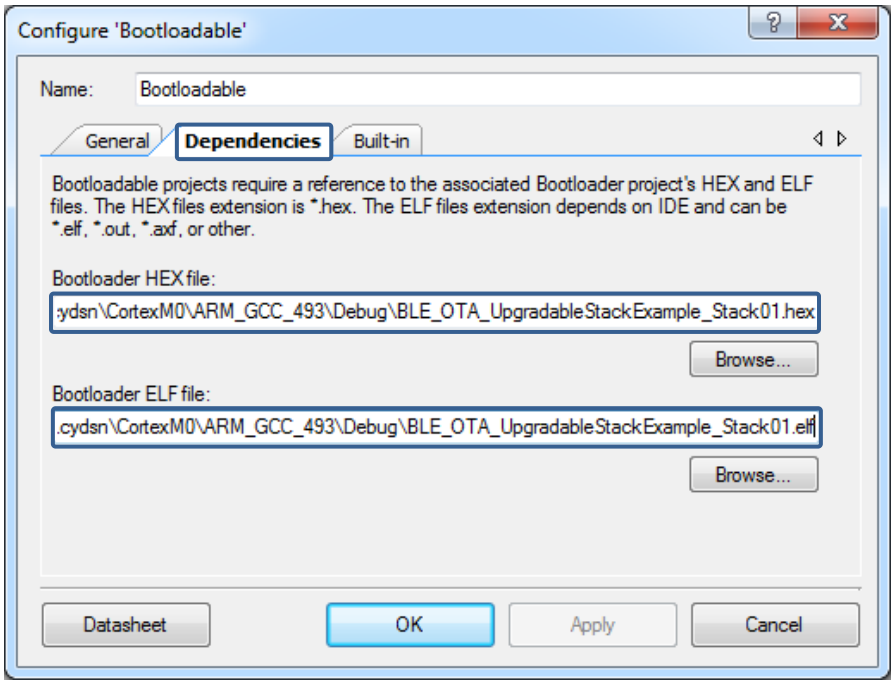


14. 指定 BLE 组件 BLE_OTA_UpgradableStackExample_Stack01 项目中 *CyBle.cycsa* 文件的路径。
    a. 双击 BLE 组件。
    b. 转到 **General** 选项卡；在 **Over-The-Air bootloading with code sharing**（共享代码的无线传输引导加载）部分中，选择 **Profile only** 项。
    c. 在 **Stack dependency**（堆栈依赖属性）字段中选择 *CyBle.cycsa* 文件，如图 20 所示。

    CyBle.cycsa 文件位于 BLE_OTA_UpgradableStackExample_Stack01 项目目录中的…*Generated_Source**PSoC 4*\路径下。
15. 指定启动程序项目的 HEX 和 ELF 文件的路径。
    a. 双击 Bootloadable 组件。
    b. 转到 **Dependencies** 选项卡，并且将 **Bootloader HEX file** 链接到 *BLE_OTA_UpgradableStackExample_Stack01.hex* 文件（位于…*BLE_OTA_UpgradableStackExample_Stack01.cydsn* *CortexM0*\<*compiler version*>\<*build configuration*>\），如图 22 所示。
    c. 点击 **OK**，关闭 Bootloadable 组件配置对话框。

    选定 HEX 文件后，会自动选择相应的 ELF 文件。

图 22. Bootloadable 组件配置



16. 如果需要为第 17 步骤中所添加的组件分配正确的引脚，请打开 *PWMExample01.cydwr* 窗口并转到 **Pins** 选项卡。按照表 6 配置各引脚。

表 6. PWMExample01 的引脚映射情况

引脚名称	端口分配
UART:tx	P1[5]
LED_GREEN	P3[6]
SW2	P2[7]

17. 将以下文件从 BLE_OTA_UpgradableStack_HID_Keyboard 示例项目工作区目录中复制到 PWMExample01 项目工作区目录中，并且将其添加到 PWMExample01 项目中。这些文件用于执行部分 OTA 和调试功能。

    a. 要想添加头文件，请右键点击 **Workspace Explorer** 中的 **Header Files** 文件夹，然后依次点击 **Add** > **Existing Item…**项。浏览并选择所需文件，然后点击 **Open**。

    b. 要想添加源文件，请右键点击 **Workspace Explorer** 中的 **Source Files** 文件夹，然后依次点击 **Add** > **Existing Item…**项。浏览并选择所需文件，然后点击 **Open**。

    - *OTAMandatory.h*
    - *debug.h*
    - *common.h*
    - *options.h*
    - *OTAMandatory.c*
    - *debug.c*

    OTAMandatory.c/h 文件包含所有必需功能，从而使能可升级堆栈 OTA。所有其他文件都是不必要的，可以复制它们，以防止编译时错误。*debug.h/c* 文件用于打印基于 UART 的调试信息。*common.h* 文件包含了 LED 状态和 WDT 选项的定义。*options.h* 文件用于使能/禁用调试功能。

18. 必须将 BLE_OTA_UpgradableStack_HID_Keyboard 项目中的其它代码添加到 PWMExample01 项目中的 *main.c* 内才能使能引导加载或 OTA 功能。可以下载修改文件。将 PWMExample01 项目的 *main.c* 文件替换为 *…\Code\Upgradable Stack OTA*路径下的 *main.c* 文件。下面显示的是代码相关部分（针对 OTA）的说明。

AfterImageUpdate()函数检查是否更新并首次运行了应用镜像。如果该镜像首次运行，并且在 BLE 组件中 Bonding Requirement（连接要求）选项被设置为 Bonding，那么该函数将验证连接数据并擦除它（如果它无效）。另外，该函数还设置了未使用元数据区中的更新检测标志。

InitializeBootloaderSRAM()函数用于初始化代码共享功能所需的 BLE 堆栈 SRAM。主函数一开始，就会调用该函数。否则，固件将发生不可预知的行为。

要想从应用镜像切换到堆栈应用，首先要使用输入参数 0 调用 Bootloadable_SetActiveApplication()，以便将堆栈设置为有效的应用。接下来要调用 Bootloadable_Load()函数，再使用 CySoftwareReset()进行软件复位。整个序列显示在代码 1 中。在该示例中，可通过按下并释放 SW2 开关来启动引导加载程序。

代码 1. 编写应用级的连接信息

```
/* For GCC compiler use separate API to initialize BLE Stack SRAM.
 * This is needed for code sharing.
 */
#if !defined(__ARMCC_VERSION)
    InitializeBootloaderSRAM();
#endif


/* Checks if Self Project Image is updated and Runs for the First time */
AfterImageUpdate();

/* Start CYBLE component and register generic event handler */
CyBle_Start(AppCallBack);

while (1)
{
    /* If key press event was detected - debounce it and switch to
    bootloader emulator mode */
    if (SW2_Read() == 0u)
    {
        CyDelay(500u);
        if (SW2_Read() == 0u)
        {
            CyDelay(500u);
            while (SW2_Read() == 0u)
            {
                /* Wait for button to be released */
            }

            //Switch to the Stack project, which enables OTA service
            Bootloadable_SetActiveApplication(0);
            Bootloadable_Load();
            CySoftwareReset();
        }
    }
}
```

19. 在 *PWMExample01.cydsn* 项目中创建名称为“LinkerScripts”的新文件夹。
20. 第 11 步骤中所创建的 Bootloadable 示例项目具有一个名称为 LinkerScripts 的文件夹。该文件夹包含受 PSoC Creator 支持的所有三种编译器的链接脚本（*cm0gcc.ld* 和 *Cm0Mdk.scat*）。将这三个文件复制到 PWMExample01 项目中第 19 步骤所创建的 LinkerScripts 文件夹内。
21. 在 PSoC Creator 的 **Workspace Explorer** 中，为 PWMExample01 项目创建名为“LinkerScripts”的新文件夹。

22. 将以下文件添加到 PSoC Creator PWMExample01 项目中位于 **Workspace Explorer** 内的 LinkerScripts 文件夹（在第 21 步骤中创建）。在文件系统中，这些文件位于在第 19 步骤中所创建的 LinkerScripts 文件夹内。

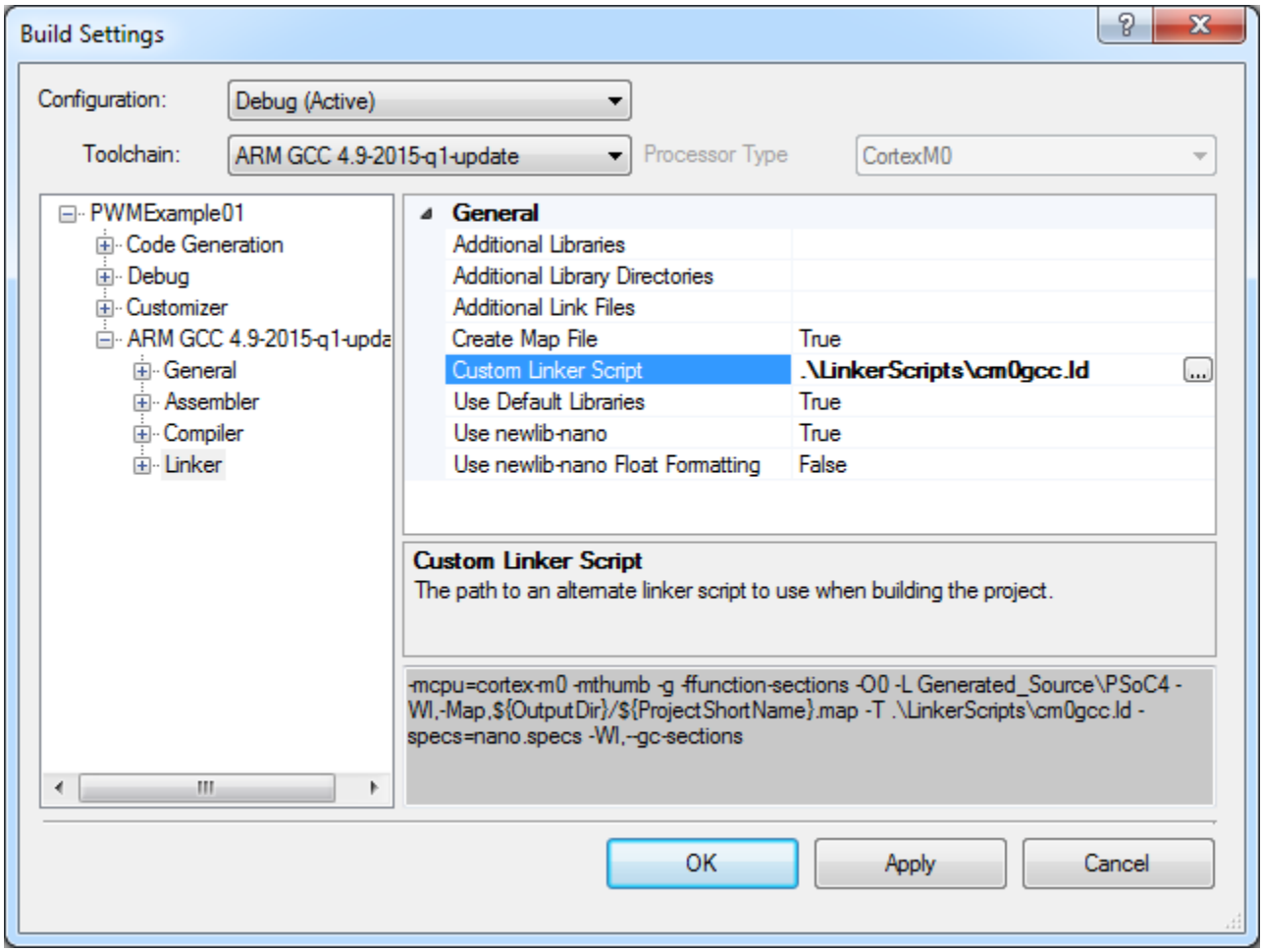
    - *cm0gcc.ld*
    - *Cm0Mdk.scat*

23. 请按照表 7 所述的内容配置 PWMExample01 项目的编译设置。这些更改允许链接器使用新的自定义链接脚本。要想修改编译设置，依次选择 **Project** > **Build Settings...**，然后在树形视图中依次选择 **PWMExample01** > **ARM GCC 4.9-2015-q1-update** > **Linker** > **General**，如图 23 所示。

表 7. 编译设置更改

字段	值
自定义链接脚本	.\LinkerScripts\cm0gcc.ld

图 23. Build Settings 对话框 — Linker 设置



24. 添加 Bootloader/Bootloadable 组件后，PSoC Creator 不再支持调试功能。然而，当固件开始执行时，通过连接至目标器件（**Debug** > **Attach to Running Target...**）或使用 UART 信息，您也可以调试该固件。如果需要使能或禁用 UART 调试功能，请查阅调试一节。请确保 UART 被使能时，设置合适的堆（0x400 字节）和栈（0x800 字节）的大小，使项目能够正确操作。否则，固件会发生不可预知的行为。更多详细信息，请参考调试一节。

25. 编译 PWMExample01 项目。项目应在无错误的情况下完成编译流程。

26. 依次选择 **Debug** > **Program**，以编程 PSoC BLE Pioneer 套件。

编程完成后，绿色 LED 将会从最高亮度等级到最低亮度等级循环。此时，您可以按照执行 OTA 升级一节所述的各种方法中的一种进行操作。另外，您还可以按照测试 OTA 功能一节中所描述的各个步骤进行测试 OTA 功能。

共有两个可用的 Bootloadable 镜像： *BLE_OTA_UpgradableStackExample_Stack01.cyacd*（栈镜像）和 *PWMExample01.cyacd*（应用镜像）。要想升级应用，需要使用应用镜像文件进行升级操作。要想升级堆栈，需要使用堆栈镜像文件进行升级操作。重新连接至目标器件，然后执行应用升级。

6 执行 OTA 升级

赛普拉斯提供了三种主机应用，用于升级目标器件的固件。无论目标器件上采用哪种 OTA 方法，都能互换使用三种主机应用进行 OTA 升级。

要想执行 OTA 升级，需要使用添加外部存储器 OTA Bootloader、添加固定堆栈 OTA Bootloader 或添加可升级堆栈 OTA Bootloader 的流程结束后所生成的 HEX 文件对目标平台进行预编程。除了 HEX 文件外，编译流程还会生成一个作为 Bootloadable/应用镜像的.*cyacd* 文件。这两个文件都位于 Bootloadable 项目输出目录内。

不能将一种特定 OTA Bootloader 的项目所生成的.*cyacd* 文件用于升级使用其它 OTA Bootloader 进行编程的器件。例如，使用外部存储器 OTA Bootloader 结构的项目所生成的.*cyacd* 文件不适用于具有固定堆栈 OTA Bootloader 结构的器件。

基于 PC 的固件升级方法都需要使用 CySmart USB 收发器（请参考图 24）才能正常运行。它带有 PSoC 4 BLE Pioneer 套件。

图 24. CySmart USB 收发器



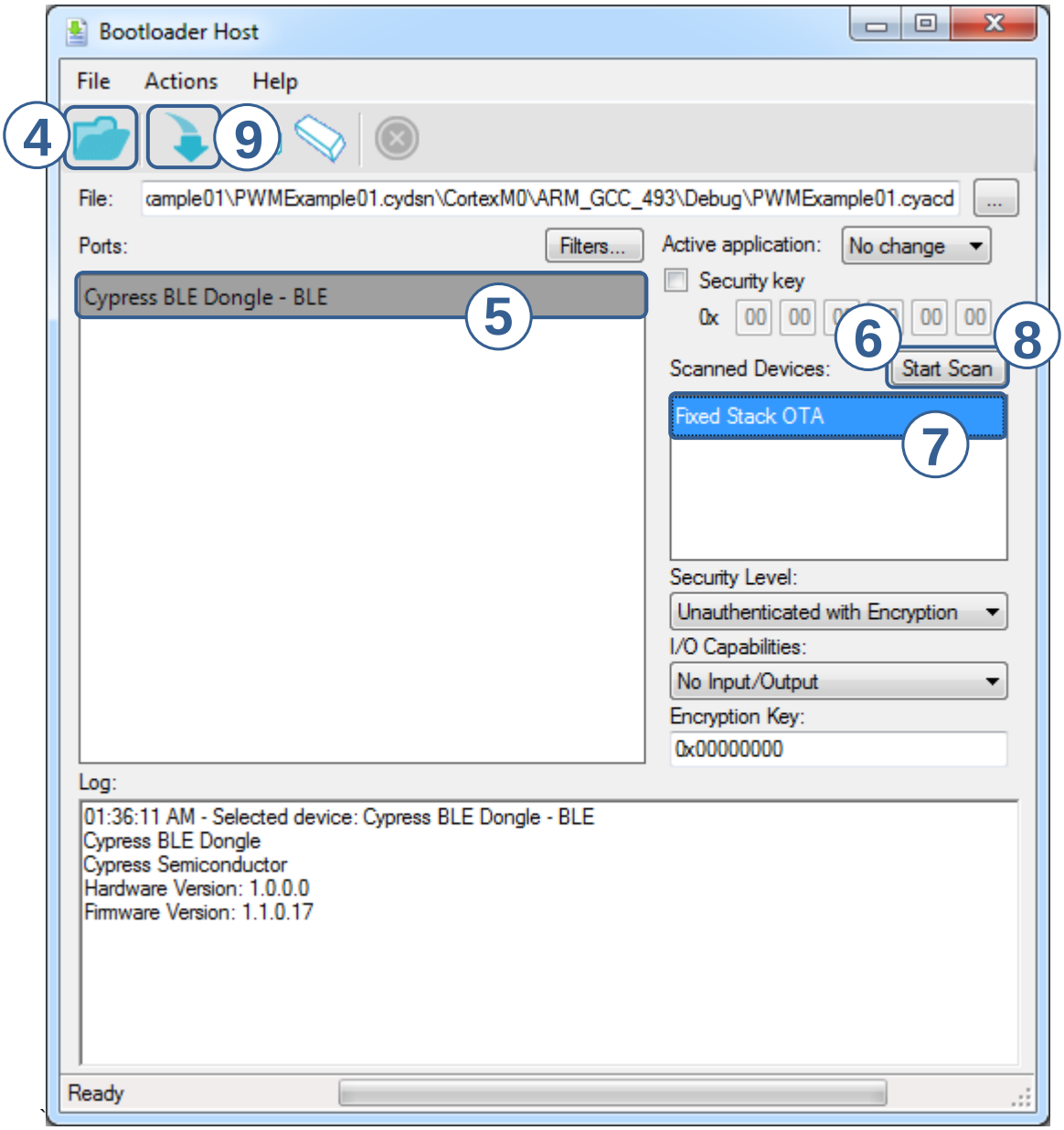
6.1 使用 Bootloader 主机工具进行升级

Bootloader 主机工具（请参考图 25）附带着 PSoC Creator，并可用于多种器件固件升级操作，包括 BLE OTA 升级操作。请按照以下各步骤使用 Bootloader 主机工具执行基于 OTA 的器件固件升级操作。执行以下各步骤前，请确保 CySmart USB 收发器已经被连接到 PC 上。

注意：支持 OTA 升级操作的 Bootloader 主机工具不适用于 CySmart 1.2 收发器固件。

1. 将 CySmart USB 收发器（如图 24 所示）插入到 PC 上的 USB 端口。

图 25. Bootloader 主机工具



2. 按下 PSoC 4 BLE Pioneer 套件上的 **SW2** 开关，使器件进入 Bootloader 模式（用红色 LED 表示）。可以根据特定要求重新实现进入 Bootloader 模式的方法（例如，通过一个特定 BLE 特性写入命令来启动引导加载过程）。

使用可升级堆栈 OTA 方法时，如果经过 40 秒 OTA 流程尚未开始，则引导加载模式会发生超时。而使用外部存储器 OTA 和固定堆栈 OTA 方法时，并不存在超时情况；固件将在引导加载模式下无限期等待。

3. 在 PSoC Creator 中，依次选择 **Tools >Bootloader Host**，打开 Bootloader 主机工具。
4. 点击 **Open** 图标（请参考图 25），然后指向**.cyacd* 文件所在的路径。该文件位于项目文件夹内（*[project folder]\CortexM0\[compiler name]*）。
5. 在 Bootloader 主机工具中，选择 **Ports** 下所列出的 **Cypress BLE Dongle**。请参考图 25。
6. 点击 **Scanned Devices** 字段旁边的 **Start Scan** 按键，如图 25 所示。
7. 等到所需器件（外部存储器 OTA/固定堆栈 OTA）出现在 **Scanned Devices** 字段中，然后选择它（请参考图 25）。
8. 点击 **Stop Scan** 按键（请参考图 25）。
9. 点击 Bootloader 主机工具中的 **Program** 按键（参考图 25），并等待完成新应用镜像上载过程。

固件升级完成后，器件将自动复位，您会看到绿色 LED 闪烁发光。更详细的信息，请查阅 PWMExample 项目数据手册。

6.2 使用 CySmart PC 工具进行升级

CySmart 是一个适用于 Windows PC 的 BLE 主机仿真工具。通过易用的 GUI，您可以测试和调试您的 BLE 外设应用。可以下载 CySmart PC 工具。欲了解更多有关 CySmart 的信息，请查阅此网页上随附的用户指南。

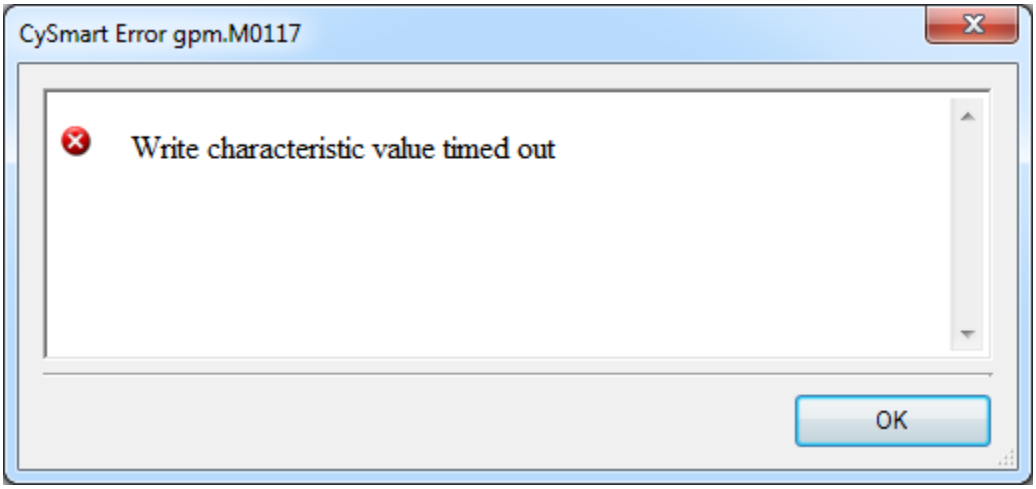
请按照以下各步骤使用 CySmart PC 工具执行一个基于 OTA 的器件固件升级操作。执行以下各步骤前，请确保 CySmart USB 收发器已经被连接到 PC 上。

1. 将 CySmart USB 收发器（如图 24 所示）插入到 PC 上的 USB 端口。
2. 按下 PSoC 4 BLE Pioneer 套件上的 **SW2** 开关，使器件进入 Bootloader 模式（用红色 LED 表示）。可以根据特定要求重新实现进入 Bootloader 模式的方法（例如，通过一个特定 BLE 特性写入命令来启动引导加载过程）。
3. 打开 CySmart 工具，并且按照 CySmart 用户指南中 2.7 更新外设固件一节的指导内容下载镜像。

固件升级完成后，器件将自动复位，您会看到绿色 LED 闪烁发光。更详细的信息，请查阅 PWMExample 项目数据手册。

如果使用 CySmart PC 工具进行 OTA 升级，引导加载过程结束时，您可能看到 **Write characteristic value timed out** 出错信息（如图 26 所示）。请注意，引导加载过程并非失败，您可以关闭该出错对话框。

图 26. 引导加载流程结束时出现“Write Characteristic Value Timed Out”错误对话框



6.3 使用 CySmart 移动应用进行升级

CySmart 移动应用是由赛普拉斯半导体公司开发的一款 BLE 或智能蓝牙工具。CySmart 可用于连接到多种 BLE 产品，也可以同赛普拉斯的 BLE 开发套件（包括 CY8CKIT-042-BLE PSoC 4 BLE Pioneer 套件、CY5672 PRoC BLE 遥控器 RDK 和 CY5682 PRoC BLE 触控鼠标 RDK）一起使用。CySmart 移动应用可用于 iOS®和 Android™。可以通过 www.cypress.com/cysmartmobile 网页下载它们。更多有关信息，请查阅此网页上随附的用户指南。

6.3.1 在 iOS 平台上进行升级

请按照以下步骤使用 CySmart iOS 应用执行基于 OTA 的器件固件升级操作。执行以下各步骤前，请确保您的手机/平板电脑已经安装好了 CySmart iOS 应用。另外，请确保您的移动设备具有新的应用镜像（*PWMExample01.cyacd*）。

1. 按下 PSoC 4 BLE Pioneer 套件上的 **SW2** 开关，使器件进入 Bootloader 模式（用红色 LED 表示）。
2. 在 iOS 设备上启动 CySmart 应用，并且按照 CySmart iOS 应用用户指南中 2.1.2.3 赛普拉斯 Bootloader 服务一节的指导内容下载镜像。

6.3.2 在 Android 平台上进行升级

请按照以下步骤使用 CySmart Android 应用执行基于 OTA 的器件固件升级操作。执行以下各步骤前，请确保您的手机/平板电脑已经安装好了 CySmart Android 应用。另外，请确保您的移动设备具有新的应用镜像（*PWMExample01.cyacd*）。

1. 按下 PSoC 4 BLE Pioneer 套件上的 **SW2** 开关，使器件进入 Bootloader 模式（用红色 LED 表示）。
2. 在 Android 设备上启动 CySmart 应用，并且按照 CySmart Android 应用用户指南中 2.1.2.3 赛普拉斯 Bootloader 服务一节的指导内容下载镜像。

固件升级完成后，器件将自动复位，您会看到绿色 LED 闪烁发光。更详细的信息，请查阅 PWMExample 项目数据手册。

7 测试 OTA 功能

请按照以下步骤测试 OTA 功能：

1. 在 PWMExample 项目的 *main.c* 中，将 `BRIGHTNESS_DECREASE` 宏的数值设置为 1000。更详细的信息，请查阅 PWMExample 数据手册。
2. 重新编译 PWMExample 项目（依次选择 **Build** > **Build PWMExample01**）。这样将生成新的.*cyacd* 文件。
3. 按照执行 OTA 升级一节所介绍的一种方法进行操作。请确保选择第 2 步骤所生成的 *PWMExample.cyacd* 输出文件。

固件升级完成后，器件将自动复位，您会看到绿色 LED 亮度的调光周期速率比较快。

将 `BRIGHTNESS_DECREASE` 宏的数值更改为 0 到 63000 范围内的任意值，并且按照上述各个步骤进行操作，以观察不同的结果。该值越小，亮度调光周期性变化越慢；该值越大，亮度调光周期性变化越快。

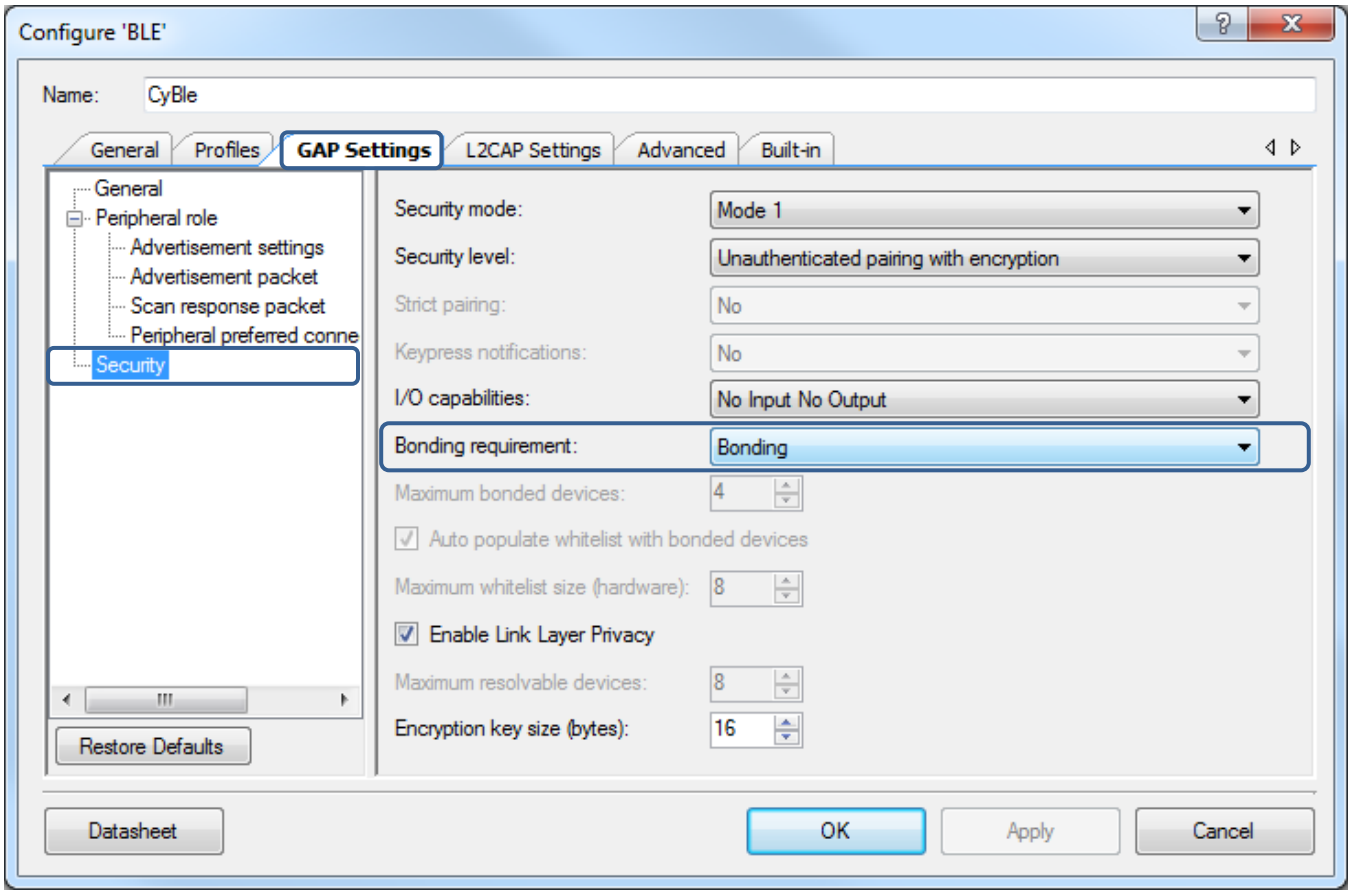
8 其他注意事项

8.1 连接/配对信息

本节介绍的是在 BLE 组件中使能连接功能的方式。另外，它还介绍了三种 OTA 升级方法对连接/配对信息的影响。

要想使能连接功能，请在 **Configure ‘BLE’** 对话框中 **GAP Settings** 选项卡下的 **Security** 内将 **Bonding requirement** 选项设置为 **Bonding**（请参考图 27）。

图 27. 配置 BLE 组件以保留连接信息



在执行闪存写操作过程中，中断不被处理（被禁止），因此应用必须在适当的时候处理闪存写操作。闪存写事件挂起时，堆栈将设置 cyBle_pendingFlashWrite 变量。应用程序通过检查该标志的状态以及 CyBle_StoreBondingData API 来编写连接信息。请参考代码 2。

代码 2. 编写应用级的连接信息

```
if((cyBle_pendingFlashWrite != 0u))
{
    #if (DEBUG_UART_ENABLED == YES)
        CYBLE_API_RESULT_T apiResult;
        apiResult = CyBle_StoreBondingData(0u);
        DBG_PRINTF("Store bonding data, status: %x \r\n", apiResult);
    #else
         (void)CyBle_StoreBondingData(0u);
   #endif /* (DEBUG_UART_ENABLED == YES) */
}
```

下面介绍的是特定于每一种 OTA Bootloader 的连接/配对信息。

8.1.1 外部存储器 OTA Bootloader

使用外部存储器 OTA Bootloader 结构进行升级时，连接信息由应用（Bootloadable）项目处理。因此，进行升级后，所有连接信息将被丢失。

8.1.2 固定堆栈 OTA Bootloader

使用固定堆栈 OTA Bootloader 进行升级时，连接信息被保存在 Bootloader 项目区域内。因此，升级 Bootloadable 项目后，连接信息不受任何损害；仅在使用串行线调试（SWD）编程器重新编程器件时，该信息才被擦除。

8.1.3 可升级堆栈 OTA Bootloader

使用可升级堆栈 OTA Bootloader 时，连接信息被保存在堆栈（可选）和应用镜像中。默认情况下，堆栈不保存连接信息。只要客户端特性配置描述符（CCCD）保持不变，应用升级就不会对应用的连接信息产生任何影响。更新堆栈时，应用的连接信息被丢失。

8.2 调试

为某个项目添加 OTA Bootloader 后，不能通过 PSoC Creator 调试该项目。然而，当固件开始执行时，您可以连接至目标器件（**Debug > Attach to Running Target**）。使用该方法时，一次只能调试一个项目。如果将某个错误的项目连接到正在运行的目标器件，该器件将被重新编程。另外，必须使能 SWD 接口，从而将调试器连接至目标器件。通过在您项目中的*.cydwr* 文件内（例如，*PWMExample01.cydwr*）将 **Debug Select** 更改为 **SWD**，可以使能项目的 SWD 接口（请参考图 28）。可以从该项目的 **Workspace Explorer** 访问*.cydwr* 文件。

如果 SWD 接口不能用于调试，则使用 GPIO 翻转功能或任何串行通信接口或协议（如 UART）。可以使用软件 UART（TX）组件调试 PSoC Creator 中的 OTA 示例项目。由于您已经将 PSoC Creator 示例项目中的 *debug.c* 和 *debug.h* 文件来添加您的 Bootloadable 项目（请参考为目标项目添加固件 OTA Bootloader 支持一节），因此您可以使用该功能来调试这些项目。

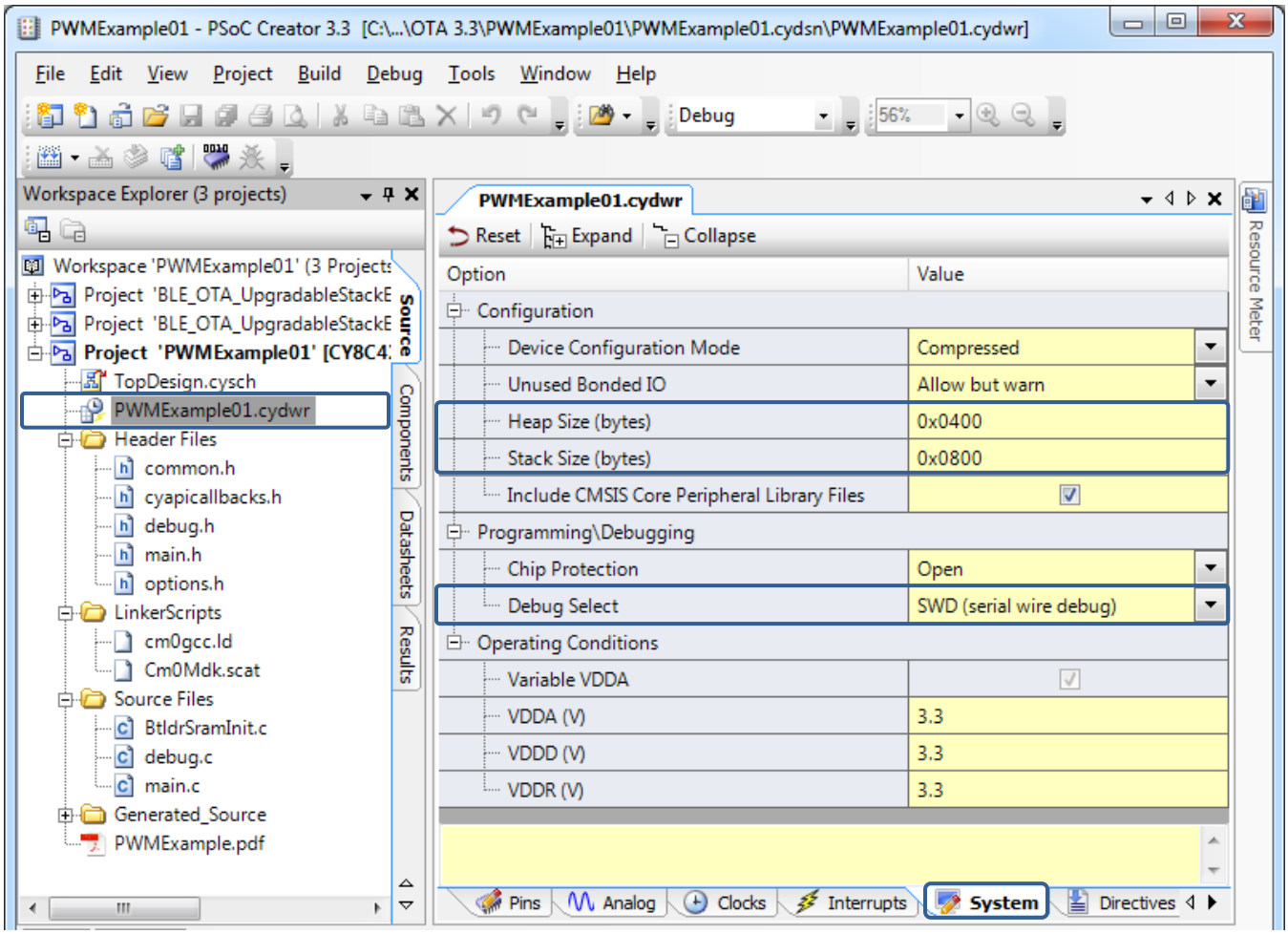
通过将 *Options.h* 头文件中的 DEBUG_UART_ENABLED 宏设置为 YES（如代码 3 所示），可以使用 UART 对任意 OTA 项目进行调试。

代码 3. Options.h 文件中的 DEBUG_UART_ENABLED 宏

```
#define DEBUG_UART_ENABLED      (YES)
```

OTA 示例项目使用 printf 语句（直接或在宏中定义的语句）通过 UART 发送信息。printf 需要大量的堆和栈空间来正常执行。因此，您需要分配足够的堆和栈存储器空间，使应用在 DEBUG_UART_ENABLED 宏被使能时能够正常运行。要想实现该操作，请双击打开 **Workspace Explorer** 中的项目的*.cydwr* 文件（例如，*PWMExample01.cydwr*），然后转到 **System** 选项卡（如图 28 所示）。将 **Heap Size** 和 **Stack Size** 设置为合适的数值（对于示例项目，应该将 **Heap Size** 和 **Stack Size** 分别设置为 **0x400** 和 **0x800** 字节）。

图 28. 在 PSoC Creator 中更改堆和栈的大小



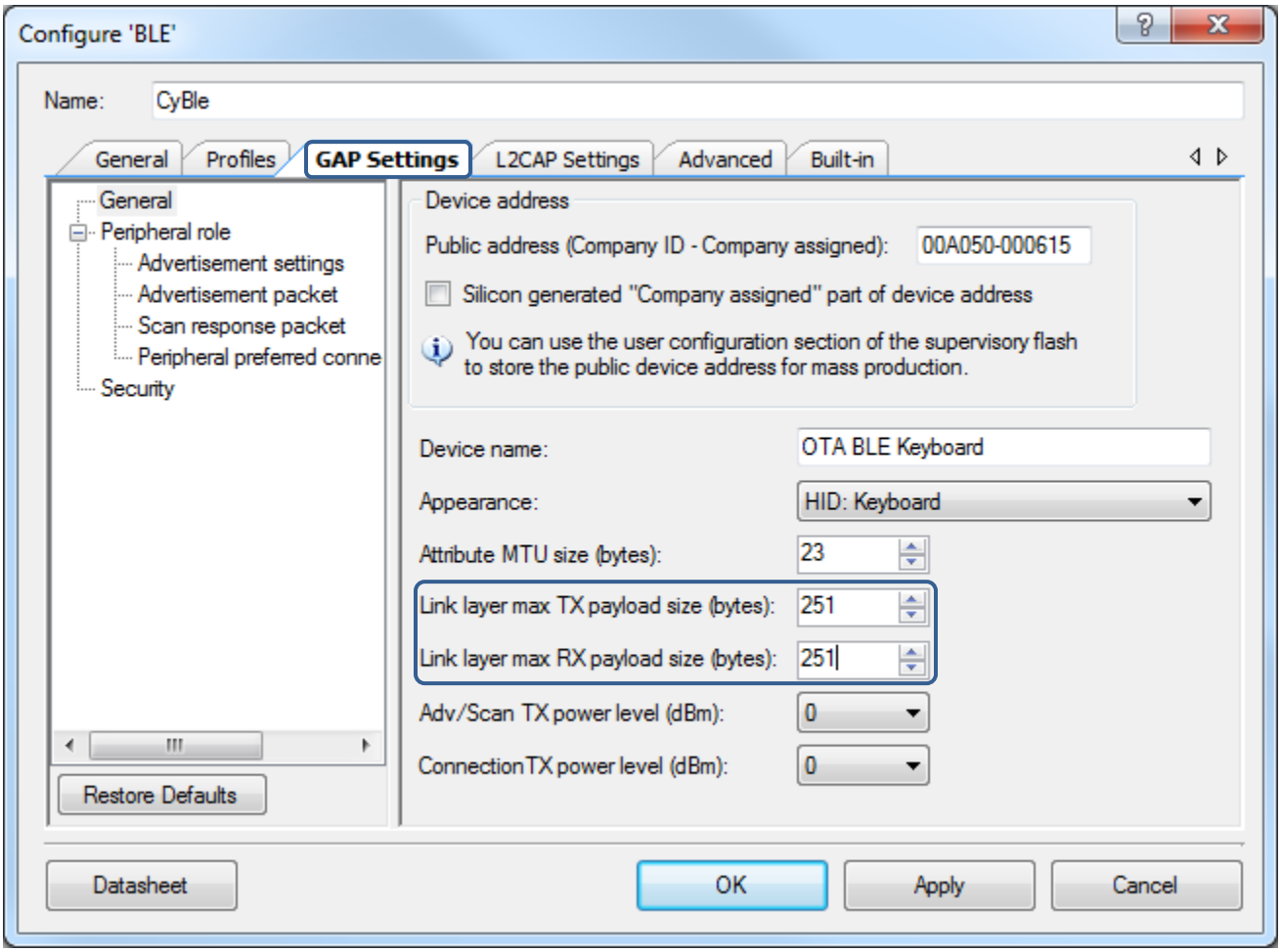
8.3 数据长度扩展（DLE）

蓝牙 SIG 在蓝牙内核规范 4.2 中引入了 LE 数据包长度扩展或数据长度扩展（DLE）。凭借该特性，链路层中最大数据通道的负载长度可从 27 个字节（在蓝牙内核规范 4.1 或更早版本中）递增到 251 个字节。这样会使链路层数据包的容量大约递增 10 倍，且该链路层的吞吐量大约递增 2.6 倍。更多有关赛普拉斯器件支持的蓝牙内核规范 4.2 的 DLE 和其他性能，请参考 AN99209 中的内容。

DLE 性能在 BLE 组件版本 3.0 或更高版本中提供，并受符合蓝牙内核规范 4.2 标准的所有赛普拉斯器件的支持。通过以下步骤，可以为任何 BLE 项目（包括 OTA 项目）使能 DLE 性能。

1. 双击 BLE 组件，可以打开 BLE 组件配置对话框。
2. 在 **GAP Settings** 选项卡中，将 **Link layer max TX payload size (bytes)**和 **Link layer max RX payload size (bytes)**的数值设置为 **27** 到 **251** 范围内的某个值（如图 29 所示）。

图 29. BLE 组件的 GAP Settings 选项卡



8.4 数据持久性

8.4.1 使用 SFlash

PSoC 和 PRoC BLE 器件都有用户 SFlash 区域。SFlash 用于存储闪存保护设置、调整设置等信息（用户不能访问这些信息）。另外，SFlash 拥有用户可配置的四行，用于存储蓝牙或产品特定的信息，如设备地址、制造编号/序列号、传感器校准数据，等等。在编程周期或 OTA 升级期间，未擦除这些行，因此可以保持数据。此外，在独立于固件编程的生产周期中，可以对 SFlash 进行写操作。通过固件或 SWD 编程接口可以访问 SFlash 的可配置行。

用户读/写 SFlash 的示例项目显示在此处。更多有关在您的项目中实现该性能的信息，请参考项目用户指南。该示例项目使用 128 KB BLE 器件。表 8 显示的是所有 BLE 器件的参数信息。

表 8. SFlash 示例项目的器件特定参数

参数	128 KB 器件	256 KB 器件
USER_SFLASH_ROW_SIZE	128	256
USER_SFLASH_ROWS	4	4
USER_SFLASH_BASE_ADDRESS	0x0FFFF200u	0x0FFFF400u

8.4.2 使用校验和排除功能

在计算 Bootloadable 应用校验和时，可以不包括部分闪存。该闪存区可用于存储用户数据或组件数据（例如，BLE 使用该方法来存储配对数据）。更多有关校验和排除功能的信息，请参考 Bootloadable 组件数据手册。要想使能该功能，请按照下列步骤进行操作：

1. 将目标数据迁移到 cy_checksum_exclude 区域内。请参考以下代码示例。

代码 4.GCC 编译器的 cy_checksum_exclude 示例

```
const uint8 byteArray[10] CY_SECTION(".cy_checksum_exclude") =
            {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

代码 5.MDK 编译器的 cy_checksum_exclude 示例

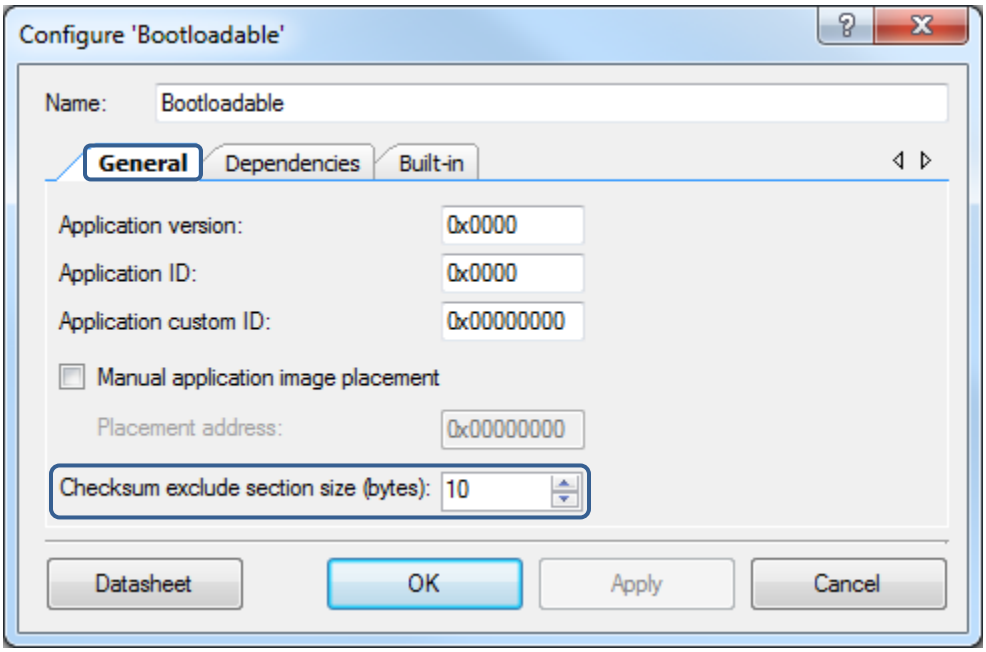
```
const uint8 byteArray[10] CY_SECTION(".cy_checksum_exclude") =
            {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

代码 6.IAR 编译器的 cy_checksum_exclude 示例

```
#pragma location=".cy_checksum_exclude"
const uint8 byteArray[10] =
            {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

2. 在 Bootloadable 组件中的 **General** 选项卡下，将 **Checksum exclude section size (bytes)**设置为所需要的值（对于代码 4、5 和 6，该值应该为 10 个字节）。请参考图 30。

图 30. Bootloadable 组件对话框



3. 在固定堆栈 OTA Bootloader 中，使用自定义链接脚本（显示在 LinkerScripts 文件夹内的 *cm0gcc.ld*），必须手动添加校验和排除信息。要实现该操作，请按照下面各个步骤进行操作：

a. 在 Bootloadable 组件中设置 **Checksum exclude section size (bytes)** 后，清除（**Build > Clean PWMExample01**）并生成项目（**Build > Generate Application**）。这样将在 *….\PWMExample01.cydsn\Generated_Source\PSoC4* 中创建新的链接脚本文件。另外，在 PSoC Creator **Workspace Explorer** 中依次选择 **Generated Source > PSoC 4 > cy_boot** 也可以找到这些文件。

b. 打开自定义链接脚本文件。在固定堆栈 OTA 示例项目中，该文件位于*…\PWMExample01.cydsn\LinkerScripts* 路径内。另外，在 PSoC Creator **Workspace Explorer** 中依次选择 **PWMExample01 > LinkerScripts** 也可以找到该文件。

c. 在 PSoC Creator 所生成的链接脚本中查找 `CY_CHECKSUM_EXCLUDE_SIZE` 链接脚本变量。对自定义的链接脚本进行相同的更改。例如，在这种情况下，您需要清除 10 个字节，具体如下所示。

代码 7. GCC *cm0gcc.ld* 链接脚本中的 `CY_CHECKSUM_EXCLUDE_SIZE` 变量

```
CY_CHECKSUM_EXCLUDE_SIZE          = ALIGN(10, CY_FLASH_ROW_SIZE);
```

代码 8. IAR *Cm0Iar.icf* 链接脚本中的 `CY_CHECKSUM_EXCLUDE_SIZE` 变量

```
define symbol CY_CHECKSUM_EXCLUDE_SIZE = 10;
```

代码 9. *`Cm0RealView.scat`* 链接脚本中的 `CY_CHECKSUM_EXCLUDE_SIZE` 变量

```
#define CY_CHECKSUM_EXCLUDE_SIZE          AlignExpr(10, CY_FLASH_ROW_SIZE)
```

d. 进行这些更改后，保存链接脚本文件并编译该项目（**Build > PWMExample01**）。

9 总结

本应用笔记描述了各种 OTA 以及在某个应用中实现 OTA 的方式。另外，它还介绍了如何使用 OTA 升级功能和赛普拉斯所提供的工具来更新目标器件上的固件。

10 相关应用笔记

AN86526 — PSoC 4 I²C Bootloader

AN73854 — PSoC 3、PSoC 4 和 PSoC 5LP 中 Bootloader 的简介

AN68272 — PSoC 3、PSoC 4 和 PSoC 5LP UART Bootloader

AN91267 — PSoC 4 BLE 入门手册

AN94020 — PRoC BLE 入门手册

关于作者

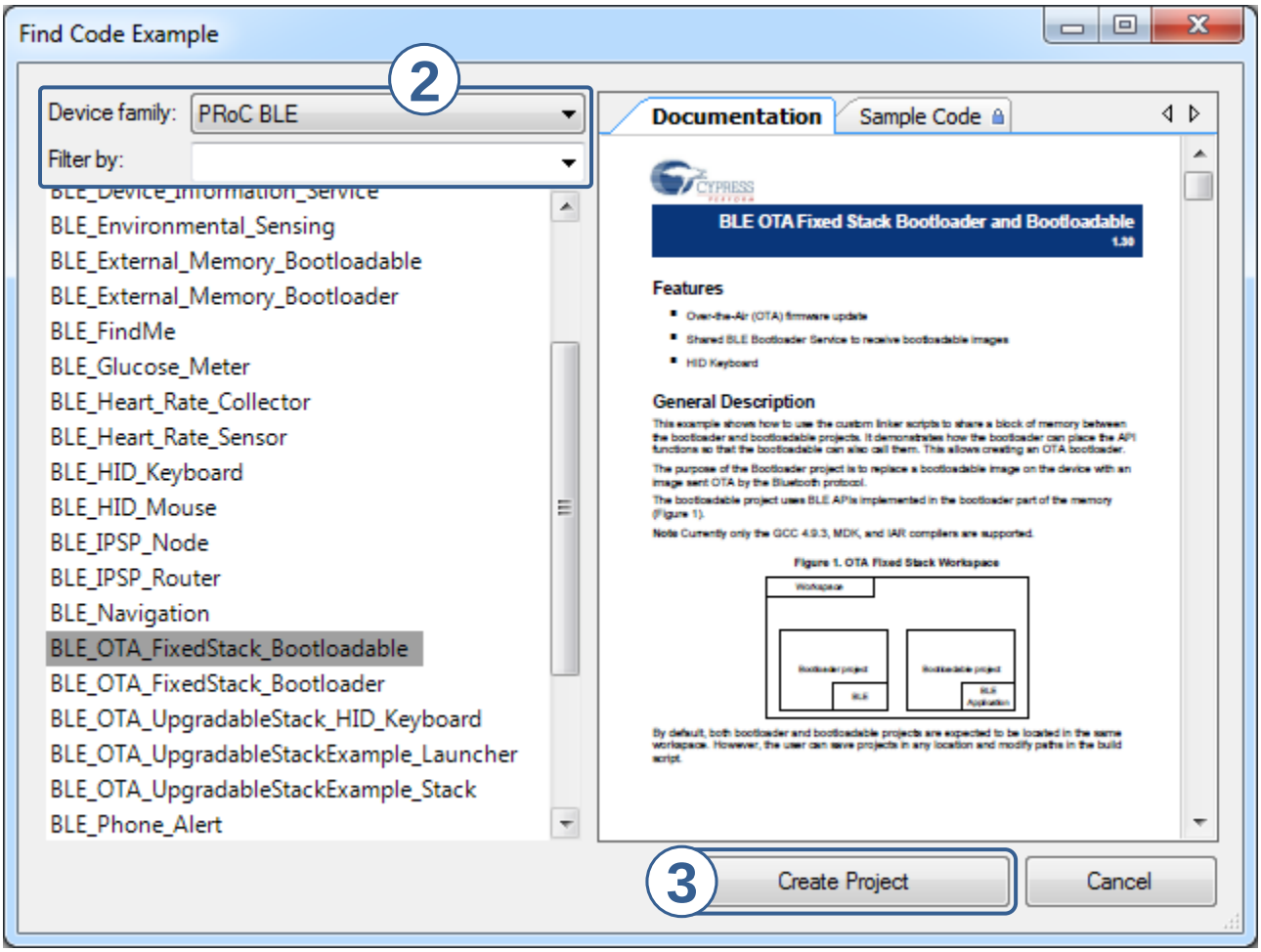
姓名：	Deepak John
职务：	应用工程师
背景信息：	Deepak John 获得了 Cochin 科技大学电子与通信工程方面的技术学士学位以及 Bridgeport 大学的电子工程硕士学位，并且在设计和编译嵌入式系统方面有多年经验。

A 附录 A

A.1 创建一个示例项目工作区

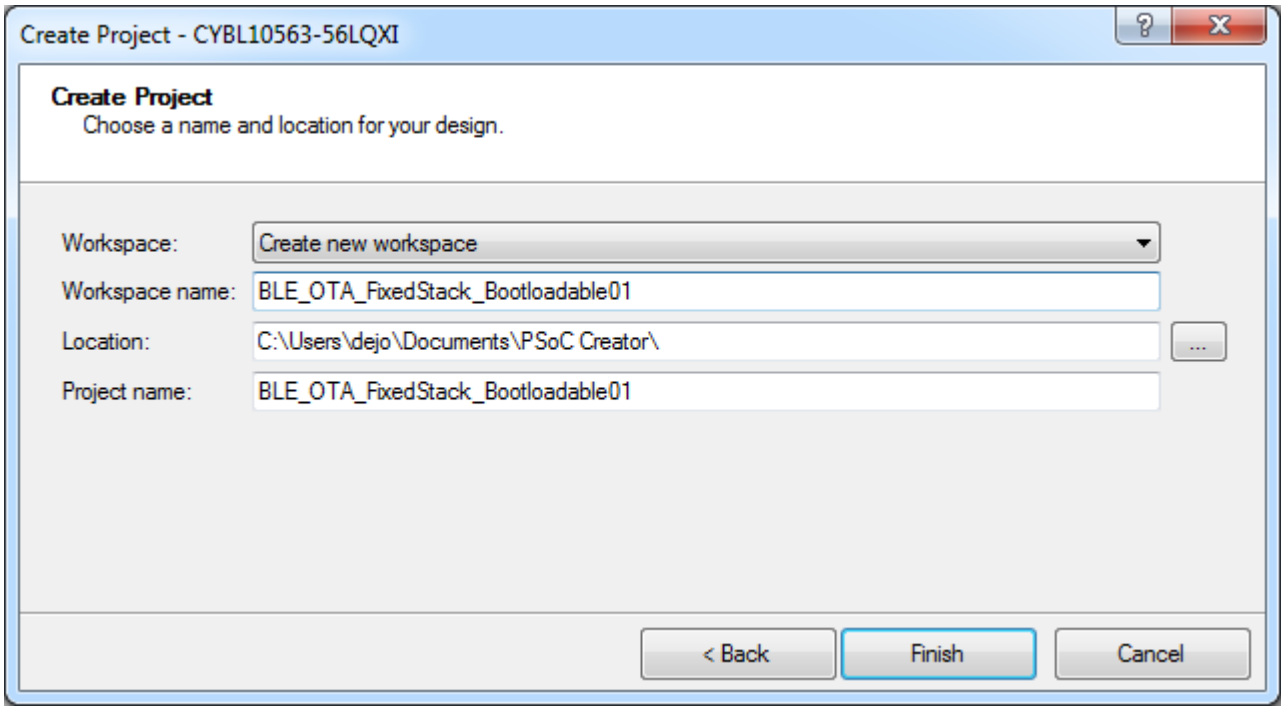
1. 在 PSoC Creator 中，依次点击 **File** > **Code Example…**以打开 **Find Example Project** 对话框，如图 31 所示。
2. 在 **Device Family** 和 **Filter by** 字段中，输入所需关键词来缩小搜索范围，如图 31 所示。
3. 选择所需要的示例项目，并点击 **Create Project** 按键。

图 31. Find Code Example Project 对话框 — 创建示例项目工作区



4. 将 **Workspace** 选项设置为 **Create New Workspace**。指定新示例项目工作区的 **Location**（位置）（请参考图 32），然后点击 **Finish**。
5. 选择所需路径，并点击 **Finish**。

图 32. Create Project 对话框 — 创建示例项目工作区



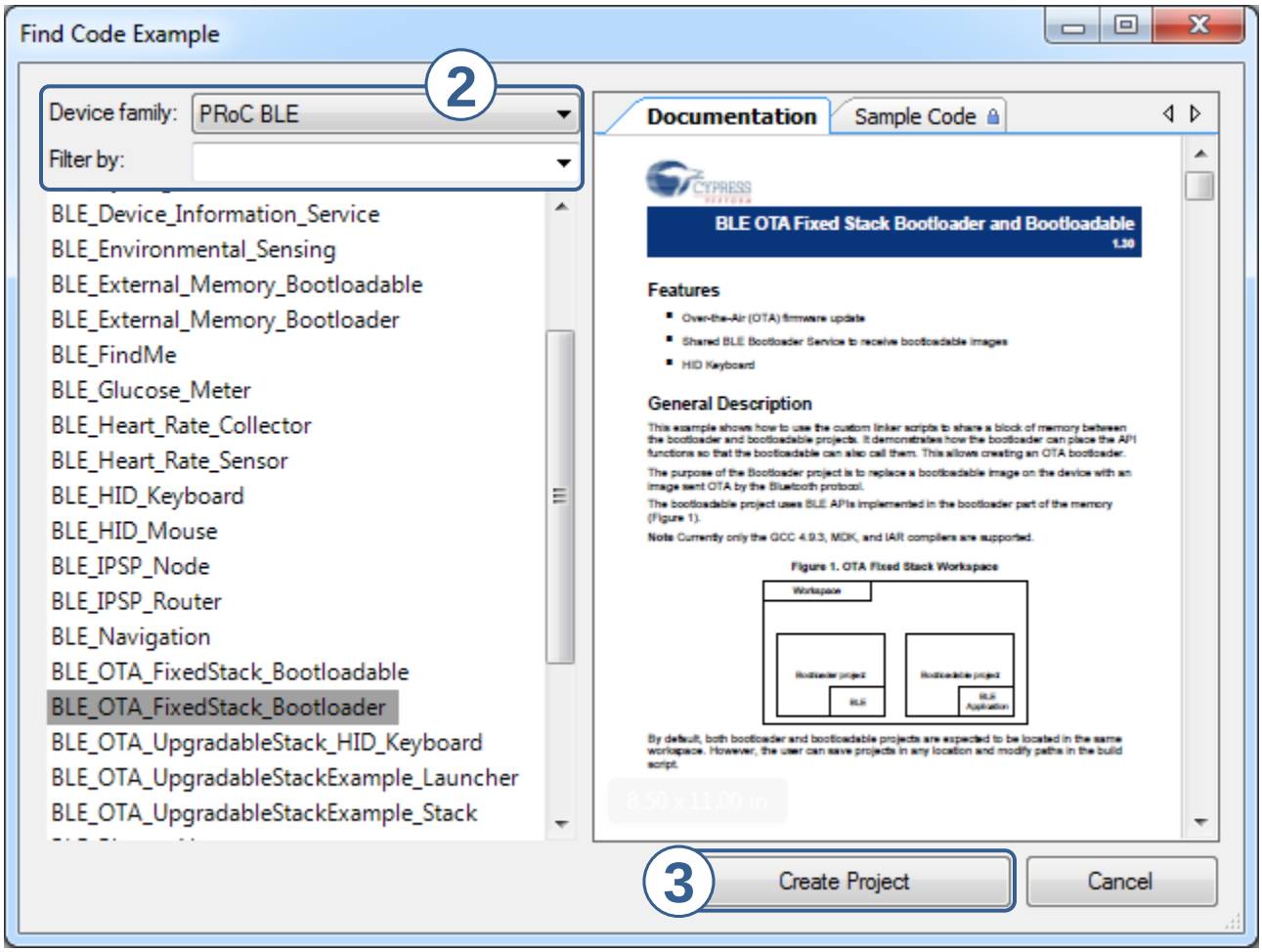
完成这些步骤后，将在指定的位置中出现新创建的工作区。向该工作区添加所选的示例项目。新创建的项目中存在一个数据手册（*<project_name>.pdf*）。您可以查阅该文档，了解有关示例项目的信息。

A.2 为现有工作区添加一个示例项目

在执行本节中的各步骤前，请确保 PSoC Creator 实例中已经打开了项目/工作区。要想打开现有的项目/工作区，请依次选择 **File** > **Open** > **Project/Workspace**。

1. 依次选择 **File** > **Code Example…**以打开 **Find Example Project** 对话框，如图 33 所示。

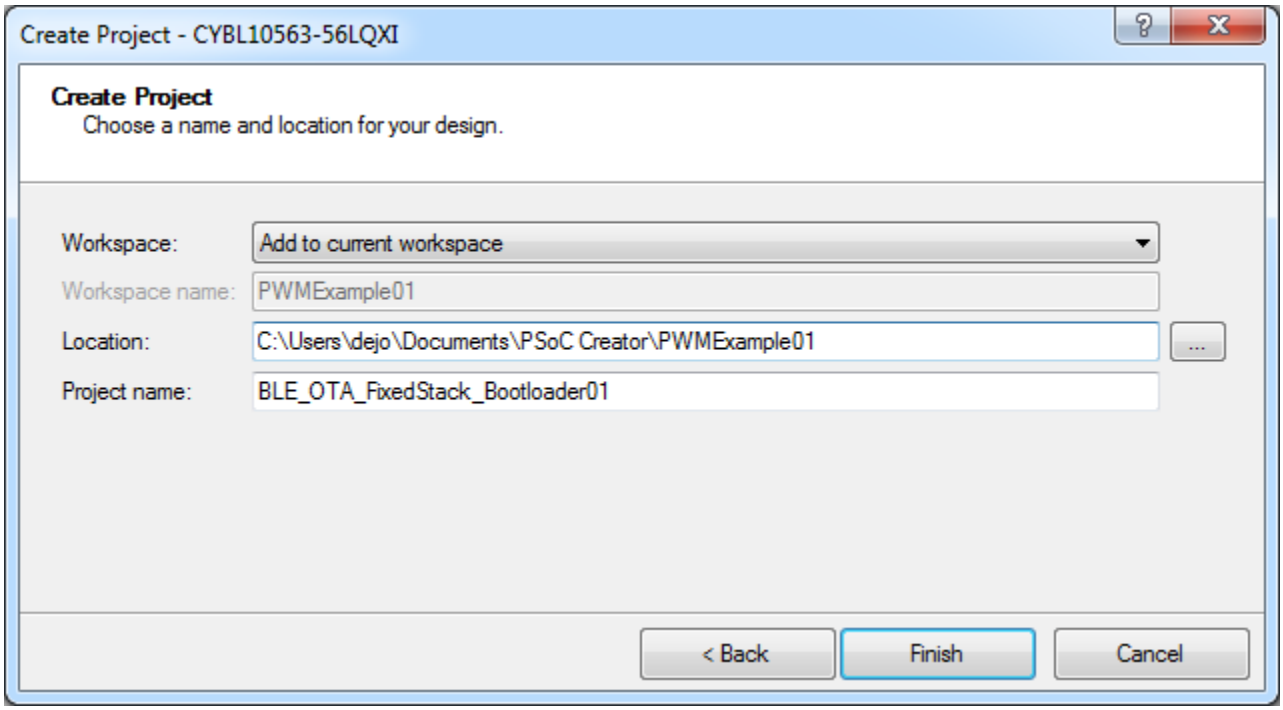
图 33. Find Example Project 对话框 — 将示例项目添加到现有的工作区



2. 在 **Device Family** 和 **Filter by** 字段中，输入所需关键词来缩小搜索范围，如图 33 所示。
3. 选择所需要的示例项目，并点击 **Create Project** 按键。
4. 将 **Workspace** 选项设置为 **Add to current workspace**。指定新示例项目工作区的 **Location**（位置）（请参考图 34），然后点击 **Finish**。
5. 选择所需路径，并点击 **Finish**。

完成这些步骤后，选定的示例项目将被添加到 PSoC Creator 示例中已打开的工作区内。在新创建的项目中存在一个数据手册（*<project_name>.pdf*）。您可以查阅该文档，了解有关示例项目的信息。

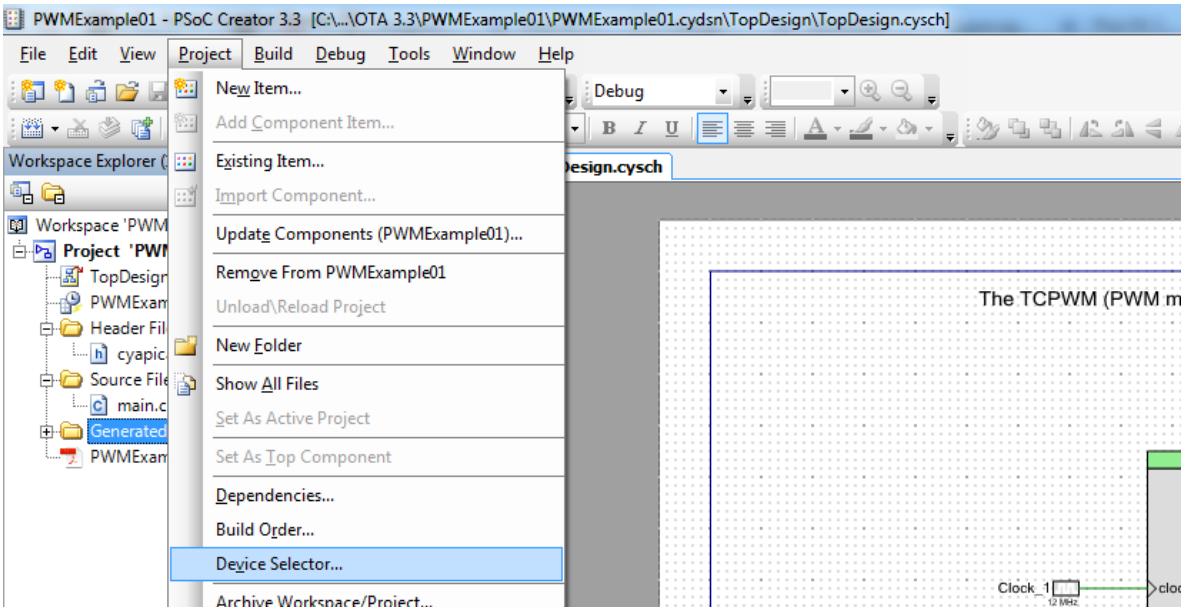
图 34. Create Project 对话框 — 将示例项目添加到现有的工作区



A.3 选择其它器件

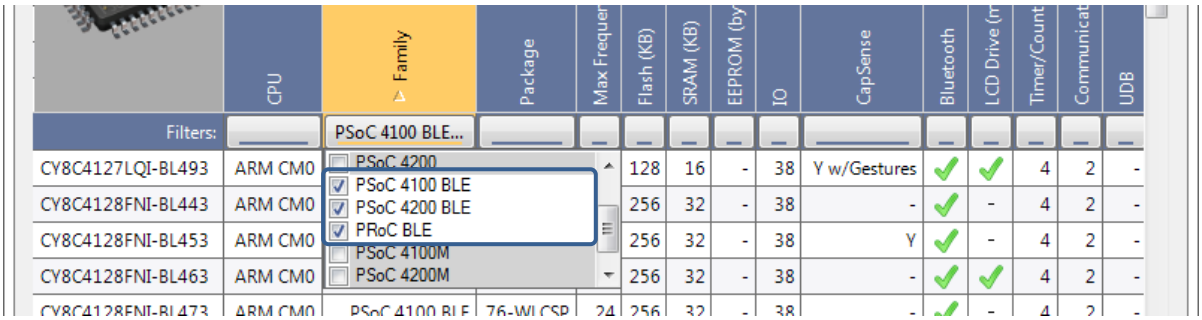
1. 在 PSoC Creator 中，依次选择 **Project** > **Device Selector…**（请参考图 35）。

图 35. 启动器件选型器



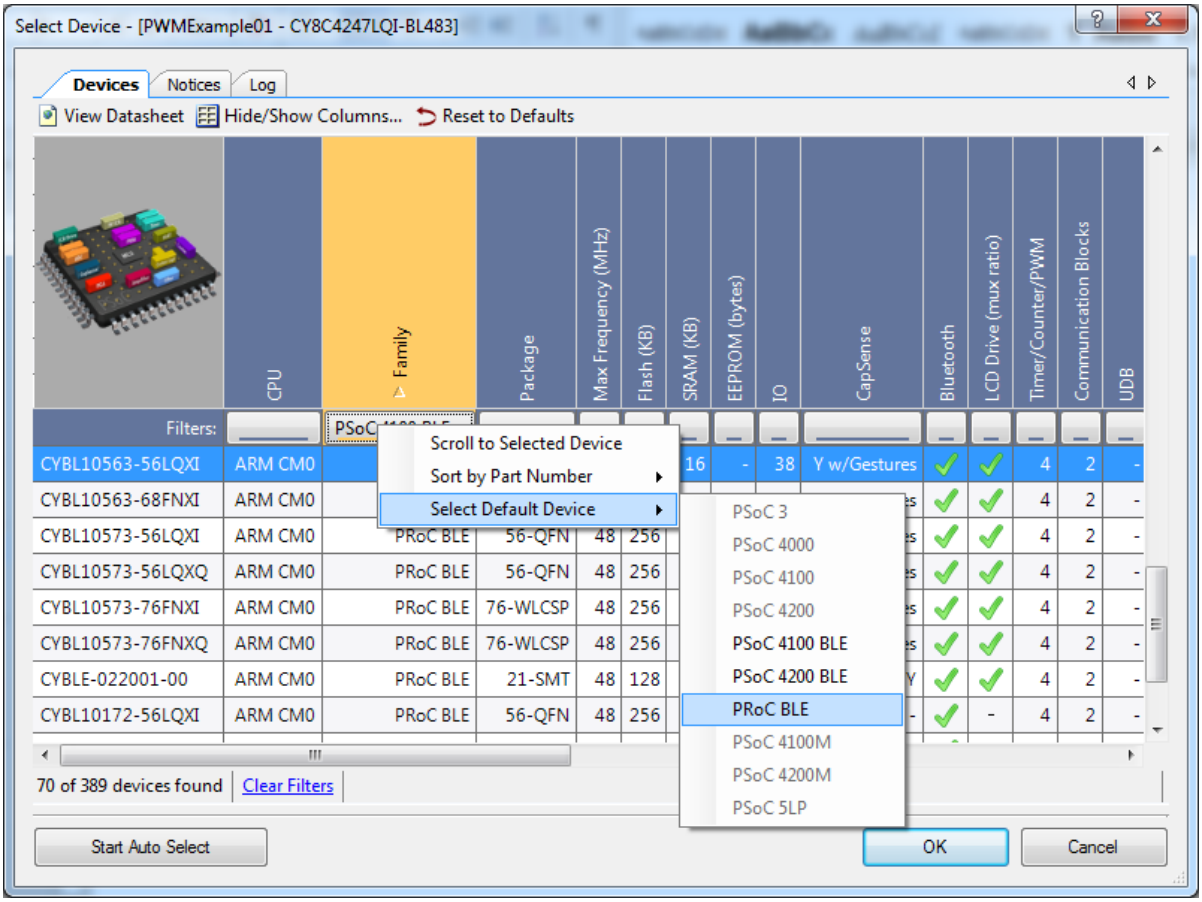
2. 使能 **Family** 筛选类别中的 **PSoC 4200 BLE**、**PSoC 4100 BLE** 和 **PRoC BLE** 项，如图 36 所示。

图 36. 选择器件系列



3. 从列表中选择符合应用的 BLE 器件。如果您的目标器件是 CY8CKIT-042-BLE Pioneer 套件附带的 PRoC BLE/PSoC 4 BLE 模块，请右键点击 **Family** 筛选器，然后选择 **Select Default Device** 项，再选择合适的器件系列，如图 37 所示。

图 37. 选择 CY8CKIT-042-BLE Pioneer 套件中的默认 PRoC BLE 器件



4. 点击 **OK**，以关闭器件选型器。

A.4 添加 Bootloader 服务

请按照以下步骤，为现有的 BLE 组件添加一个 Bootloader 服务。请查阅 BLE 组件数据手册，了解更多有关 Bootloader 服务的信息。

1. 双击 BLE 组件，打开配置对话框（请参考图 38）。
2. 转到 **Profiles** 选项卡，并选择一个服务/配置文件（如图 38 中的 **HID Device**）。
3. 点击 **Add Service**（如图 38 所示）以显示可用的服务，然后选择 **Bootloader**（请参考图 39）。
4. 依次选择 **Bootloader** > **Command** > **Fields**，将出现的 **Data** 字段设置为 137（请参考图 40）。
5. 依次选择 **GAP Settings** > **Security** 选项卡，然后将 **Security level** 设置为 **Unauthenticated pairing with encryption**（请参考图 41）。PSoC Creator 所提供的示例项目使用该设置；本应用笔记中所描述的流程需要使用该设置以正常工作。或者，执行更新操作时选择更新工具（CySmart）中的安全选项（组件选定的）。该步骤是不必要的。

图 38. BLE 组件配置对话框

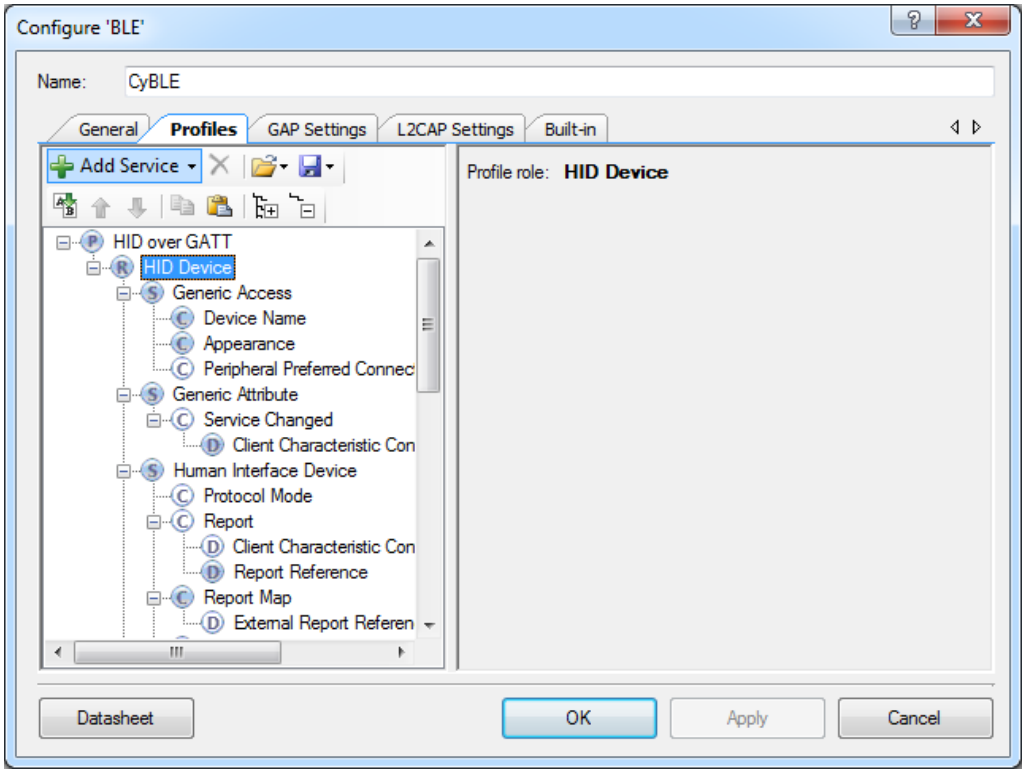


图 39. 选择 Bootloader 服务

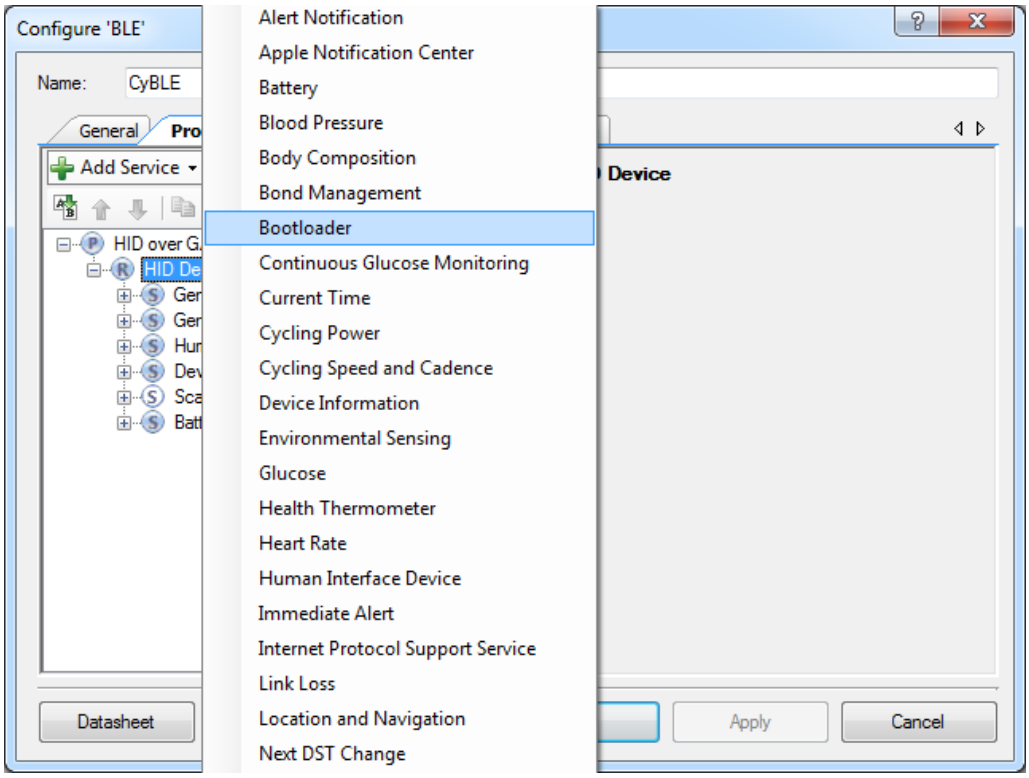


图 40. 将 Bootloader 服务数据字段设置为 137

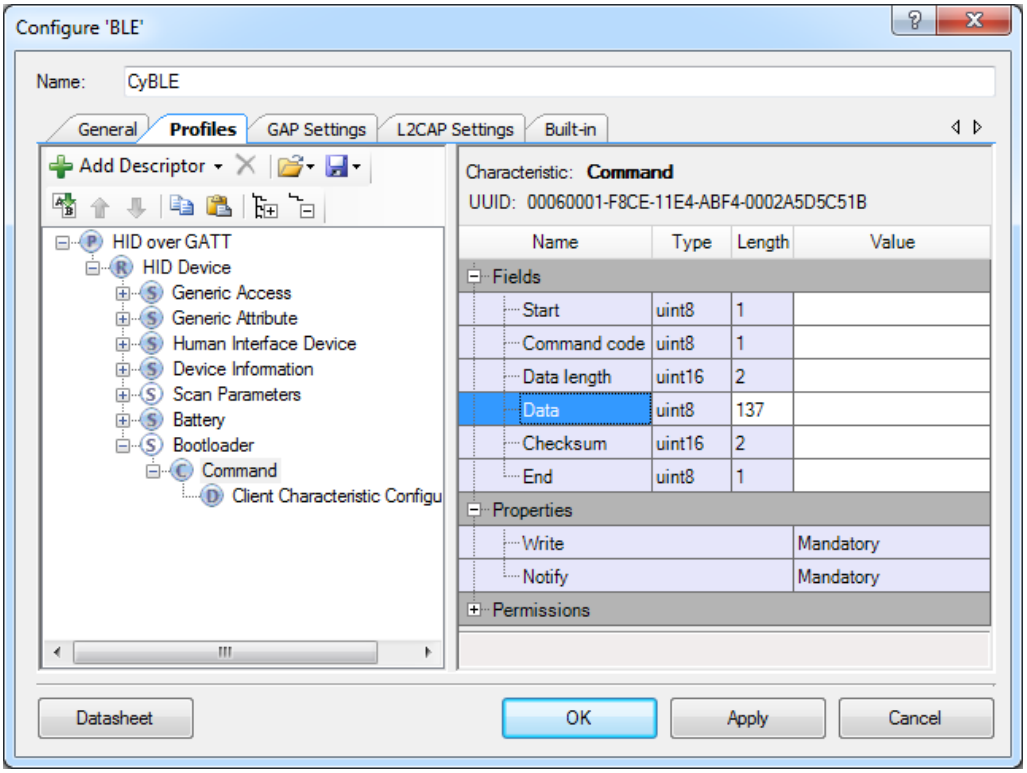
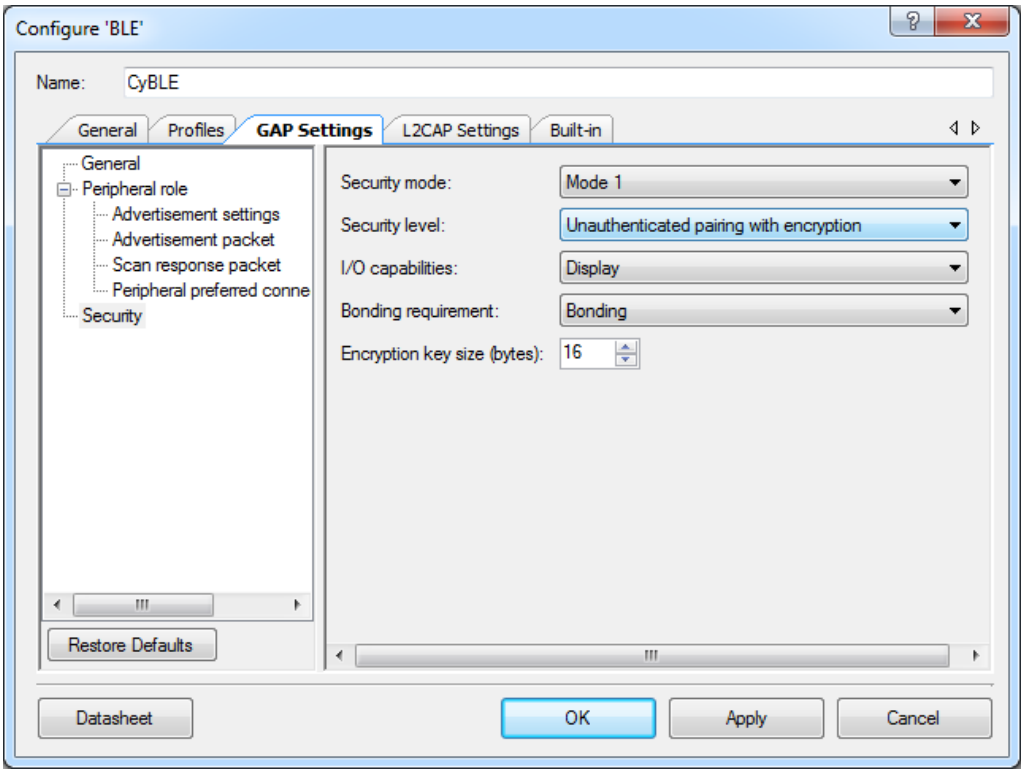


图 41. 更改安全选项



A.5 为其它赛普拉斯 BLE 器件配置固定堆栈 OTA 项目

BLE OTA 固定堆栈示例项目使用自定义链接脚本，因此 PSoC Creator 不能通过配置链接脚本提供正确的链接和位置。本部分描述的是在不使用 CY8C4247LQI-BL483 或 CYBL10563-56LQXI 器件或者使用具有不同 RAM 或 ROM 存储器大小的器件时，需要遵守的各个步骤。默认情况下，链接脚本的设置适用于 128 KB 器件。

以下部分描述了在 PSoC Creator 中更改器件后需要对链接脚本进行的修改。

A.5.1 GCC 编译器

对于 GCC 编译器，需要对 Bootloader 和 Bootloadable 链接脚本进行更改，各步骤如下。

1. 将 256 KB 器件的闪存存储器大小设置为 262144（请参考图 42）。
2. 将 256 KB 器件的 RAM 大小设置为 32768（请参考图 42）。
3. 将 256 KB 器件的行大小设置为 256（请参考图 42）。

图 42. cm0gcc.ld 文件内容 — 适用于 128 KB 器件的配置

```
26
27  MEMORY
28  {
29      rom (rx) : ORIGIN = 0x0, LENGTH = 131072
30      ram (rwx) : ORIGIN = 0x20000000, LENGTH = 16384
31  }
32
33
34  CY_APPL_ORIGIN                  = 0;
35  CY_FLASH_ROW_SIZE               = 128;
36  CY_APPL_NUM                     = 1;
37  CY_APPL_MAX                     = 1;
38  CY_METADATA_SIZE                = 64;
39  CY_APPL_LOADABLE                = 0;
40  CY_CHECKSUM_EXCLUDE_SIZE        = ALIGN(0, CY_FLASH_ROW_SIZE);
41  CY_APP_FOR_STACK_AND_COPIER     = 0;
42
```

Bootloader 和 Bootloadable 项目的链接脚本分别位于*…\cm0gcc.ld* 和*…\LinkerScripts\cm0gcc.ld* 路径下。

如果不能确定您的器件所需的数值，可以参考 PSoC Creator 生成的链接脚本中的数值（即使不使用该链接脚本）。该链接脚本位于*%PROJECT_DIR%\Generated_Source\PSoC4\cy_boot* 文件夹内，它的名称为 *cm0gcc.ld*。

A.5.2 MDK 编译器

对于 MDK 编译器，只要更改 Bootloadable 项目的链接脚本。该链接脚本位于*…\LinkerScripts\Cm0Mdk.scat* 文件夹内。

1. 将 256 KB 器件的闪存存储器大小设置为 262144（请参考图 43）。
2. 将 256 KB 器件的行大小设置为 256（请参考图 43）。

图 43. Cm0Mdk.scat 文件内容 — 适用于 128 KB 器件的存储器和行大小配置

```
36
37  #define CY_FLASH_SIZE           131072
38  #define CY_APPL_ORIGIN          0
39  #define CY_FLASH_ROW_SIZE       128
40  #define CY_METADATA_SIZE        64
41
```

3. 将 256 KB 器件的 RAM 大小（堆和栈的大小）设置为 32768（请参考图 44）。

图 44. Cm0Mdk.scat 文件内容 — 适用于 128 KB 器件的 RAM 大小

```
125     ARM_LIB_HEAP (0x20000000 + 16384 - 0x400 - 0x0800) EMPTY 0x400
126     {
127     }
128
129     ARM_LIB_STACK (0x20000000 + 16384) EMPTY -0x0800
130     {
131     }
```

如果不能确定您器件所需的数值，可以参考 PSoC Creator 生成的链接脚本中的数值（即使重新编译项目后不使用该链接脚本）。该链接脚本位于 *%PROJECT_DIR%\Generated_Source\PSoC4\cy_boot* 文件夹内，它的名称为 *Cm0Mdk.scat*。

A.5.3 IAR 编译器

如果您使用 IAR 编译器，对于 Bootloader 项目的链接脚本，在 PSoC Creator 中修改器件后可以安全使用 PSoC Creator 所生成的链接脚本。但是，必须修改 Bootloadable 项目的链接脚本。链接脚本位于…*\LinkerScripts\Cm0Iar.icf* 文件夹内。

1. 将 256 KB 器件的闪存存储器大小设置为 262144（请参考图 45）。
2. 将 256 KB 器件的 RAM 大小设置为 32768（请参考图 45）。
3. 将 256 KB 器件的行大小设置为 256（请参考图 45）。

图 45. Cm0Iar.icf 文件内容 — 适用于 128 KB 器件的配置

```
7  define symbol __ICFEDIT_region_ROM_start__   = 0x0;
8  define symbol __ICFEDIT_region_ROM_end__     = 131072 - 1;
9  define symbol __ICFEDIT_region_RAM_start__   = 0x20000000;
10 define symbol __ICFEDIT_region_RAM_end__     = 0x20000000 + 16384 - 1;
11 /*-Sizes-*/
12 define symbol __ICFEDIT_size_cstack__ = 0x0800;
13 define symbol __ICFEDIT_size_heap__   = 0x400;
14 /**** End of ICF editor section. ###ICF###*/
15
16
17 /******** Definitions ********/
18 define symbol CY_FLASH_SIZE     = 131072;
19 define symbol CY_APPL_ORIGIN    = 0;
20 define symbol CY_FLASH_ROW_SIZE = 128;
21 define symbol CY_APPL_LOADABLE  = 1;
22 define symbol CY_APPL_LOADER    = 0;
23 define symbol CY_APPL_NUM       = 1;
24 define symbol CY_METADATA_SIZE  = 64;
25 define symbol CY_APPL_MAX       = 1;
26 define symbol CY_CHECKSUM_EXCLUDE_SIZE = 0;
27 define symbol CY_APPL_FOR_STACK_AND_COPIER  = 0;
28 define symbol CY_FIRST_AVAILABLE_META_ROW   = 1;
```

如果不能确定您器件所需的数值，可以参考 PSoC Creator 生成的链接脚本中的数值（即使重新编译项目后不使用该链接脚本）。该链接脚本位于*%PROJECT_DIR%\Generated_Source\PSoC4\cy_boot* 路径下，它的名称为 *Cm0Iar.icf*。

文档修订记录

文档编号：AN97060 — PSoC® 4 BLE 和 PRoC™ BLE：无线传输（Over The Air - OTA）器件固件升级（DFU）指南

文档编号： 002-15740

版本	ECN	提交日期	变更说明
**	5403974	08/17/2016	本文档版本号为 Rev**，译自英文版 001-97060 Rev*B。
*A	5818240	07/14/2017	更新徽标和版权。
*B	6651889	03/31/2020	本文档版本号为 Rev*B，译自英文版 001-97060 Rev*C。

全球销售和设计支持

赛普拉斯公司拥有一个由办事处、解决方案中心、厂商代表和经销商组成的全球性网络。要想查找离您最近的办事处，请访问赛普拉斯所在地。

产品

Arm® Cortex®微控制器	cypress.com/arm
汽车级产品	cypress.com/automotive
时钟与缓冲器	cypress.com/clocks
接口	cypress.com/interface
物联网	cypress.com/iot
存储器	cypress.com/memory
微控制器	cypress.com/mcu
PSoC	cypress.com/psoc
电源管理 IC	cypress.com/pmic
触摸感应	cypress.com/touch
USB 控制器	cypress.com/usb
无线连接	cypress.com/wireless

PSoC®解决方案

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

赛普拉斯开发者社区

社区 | 代码示例 | 项目 | 视频 | 博客 | 培训 | 组件

技术支持

cypress.com/support

此处引用的所有其它商标或注册商标都归其各自所有者所有。

赛普拉斯半导体公司
198 Champion Court
San Jose，CA 95134-1709

© 赛普拉斯半导体公司，2015-2020 年。本文件是赛普拉斯半导体公司及其子公司，包括 Spansion LLC（“赛普拉斯”）的财产。本文件，包括其包含或引用的任何软件或固件（“软件”），根据全球范围内的知识产权法律以及美国与其他国家签署条约归赛普拉斯所有。除非在本款中另有明确规定，赛普拉斯保留在该等法律和条约下的所有权利，且未就其专利、版权、商标或其他知识产权授予任何许可。如果软件没有附带许可协议且贵方未以其他方式与赛普拉斯签署关于使用软件的书面协议，赛普拉斯特此授予贵方适用于个人的、非独占性、不可转让的许可（无转授许可权）（1）在版权保护下的软件（a）以源代码形式提供的软件,只能是在组织内部为了使用赛普拉斯的硬件去修改和复制。（b）以二进制代码形式从外部发到终端用户（直接或间接通过经销商和分销商）,仅用于赛普拉斯硬件产品单元。（2）在软件（由赛普拉斯公司提供，且未经修改）侵犯赛普拉斯专利的权利主张下，仅许可在赛普拉斯硬件产品上制造、使用、提供和导入软件。禁止对软件的任何其他使用、复制、修改、翻译或编译。

赛普拉斯不对此材料提供任何类型的明示或暗示保证，包括但不限于针对特定用途的适销性和适用性的暗示保证。没有任何电子设备是绝对安全的。因此，尽管赛普拉斯在其硬件和软件产品中采取了必要的安全措施，但是赛普拉斯并不承担任何由于使用赛普拉斯产品而引起的安全问题及安全漏洞的责任，例如未经授权的访问或使用赛普拉斯产品。此外，本材料中所介绍的赛普拉斯产品有可能存在设计缺陷或设计错误，从而导致产品的性能与公布的规 格不一致。（如果发现此类问题，赛普拉斯会提供勘误表）赛普拉斯保留更改本文件的权利，届时将不另行通知。在适用法律允许的范围内，赛普拉斯不对因应用或使用本文件所述任何产品或电路引起的任何后果负责。本文件，包括任何样本设计信息或程序代码信息，仅为供参考之目的提供。文件使用人应负责正确设计、计划和测试信息应用和由此生产的任何产品的功能和安全性。赛普拉斯产品不应被设计为、设定为或授权用作武器操作、武器系统、核设施、生命支持设备或系统、其他医疗设备或系统（包括急救设备和手术植入物）、污染控制或有害物质管理系统中的关键部件，或产品植入之设备或系统故障可能导致人身伤害、死亡或财产损失其他用途（“非预期用途”）。关键部件指，若该部件发生故障，经合理预期会导致设备或系统故障或会影响设备或系统安全性和有效性的部件。针对由赛普拉斯产品非预期用途产生或相关的任何主张、费用、损失和其他责任，赛普拉斯不承担全部或部分责任且贵方不应追究赛普拉斯之责任。贵方应赔偿并保护赛普拉斯免受所有索赔的损害，包括因人身伤害或死亡引起的索赔、费用、损失和其它责任。

赛普拉斯、赛普拉斯徽标、Spansion、Spansion 徽标，及上述项目的组合，WICED，及 PSoC、CapSense、EZ-USB、F-RAM 和 Traveo 应视为赛普拉斯在美国和其他国家的商标或注册商标。请访问 cypress.com 获取赛普拉斯商标的完整列表。其他名称和品牌可能由其各自所有者主张为该方财产。