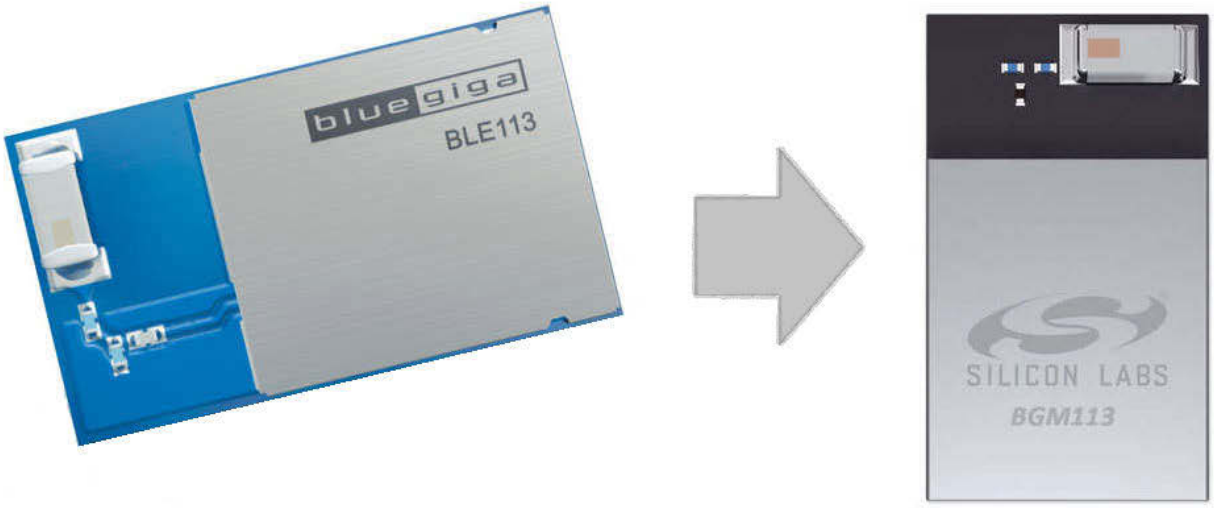
AN1036：BLE113 至 BGM113 迁移指南

本文档介绍了如何从 BLE113 Bluetooth® Module 迁移至 Blue Gecko BGM113 Bluetooth® Module。本文旨在确保尽可能简化迁移流程，重点介绍了在高级功能、模块配置、硬件设计要求和软件 API 方面的主要差异。

内容要点

- SDK 和工具
- BLE113 与 BGM113 的差异
  - 电源
  - PCB 布局
  - 配置
  - API
- 简化迁移



1. 介绍

本文档介绍了如何从 BLE113 Bluetooth Module 迁移至 Blue Gecko BGM113 Bluetooth Module。第 2. SDK 和工具 节介绍了搭配 BGM113 使用的 SDK 和工具。第 3. 功能差异 节概述了 BLE113 和 BGM113 的功能差异。第 4. 硬件相关差异 节介绍了硬件相关的差异。

BGM113 在封装上与 BLE113 兼容，但是存在着一些差异，例如 GPIO 分配可防止 BGM113 被用作 BLE113 的直接简易替换件。

第 5. 软件相关差异 节解释了 BLE113 和 BGM113 在项目配置选项上的差异。第 6. 将应用代码从 BLE113 移植到 BGM113 节比较了 BLE113 和 BGM113 的应用编程接口（BGAPI）。

2. SDK 和工具

截至 2016 年 9 月，BLE113 的最新 SDK 版本为 1.4.2.-130。

对于基于 Blue Gecko 的产品（包括 BGM113），最新的 SDK 版本为 2.0.0 (build 1391)。该 SDK 已整合到 Simplicity Studio v4 中。安装最新版本的 Simplicity Studio v4 后，您可以在 Studio 中运行*更新软件*功能，进而添加 Bluetooth Smart SDK。

有关 Blue Gecko 模块开发工具的详细信息载于文档《QSG108：Blue Gecko Bluetooth® Smart 软件快速入门指南》。

BLE113 SDK 内置 BLE GUI 测试程序，可用于测试 BLExxx 模块的不同功能。BGM113 开发工具为 **BGTool**。

Note: BLEGUI 无法用于 BGM113。

3. 功能差异

本节介绍了 BLE113 和 BGM113 之间的主要功能差异。

Table 3.1. Feature Differences between BLE113 and BGM113

Feature	BLE113	BGM113
Bluetooth features	Bluetooth 4.0	Bluetooth 4.2[1]
Max TX power	0 dBm	+3 dBm
Sensitivity	-93 dBm	-93 dBm
TX current (at 0 dBm)	18.2 mA (14.3 mA with DC-DC)	8.8 mA
RX current	14.3 mA	8.7 mA
Sleep current	0.4 µA (Sleep mode 3)	1.4 µA (EM2), 1.1 µA (EM3 stop current)
MCU	Single-cycle 8051-compatible core	ARM®Cortex®-M4, 38.4 MHz
RAM	8 kB	32 kB
FLASH	128 kB / 256 kB	256 kB
Number of GPIO	19	14
DC-DC	No DC-DC	Integrated DC-DC converter
Supply voltage range	2V to 3.6V	1.85 V to 3.8 V
Operating temperature range	-40 ° C to 85 ° C	-40 ° C to 85 ° C
Dimensions	9.15 × 15.75 × 2.1 mm	9.15 × 15.73 × 1.9 mm

Note:
1. Software upgradable to Bluetooth 4.2.

4. 硬件相关差异

本节介绍了 BLE113 和 BGM113 之间的硬件相关差异。

4.1 引出线兼容性

BGM113 模块的引出线如下图所示。BGM113 与 BLE113 具有**封装兼容性**。

接地和电源引脚、复位、编程和 GPIO 引脚映射至相同的引脚。

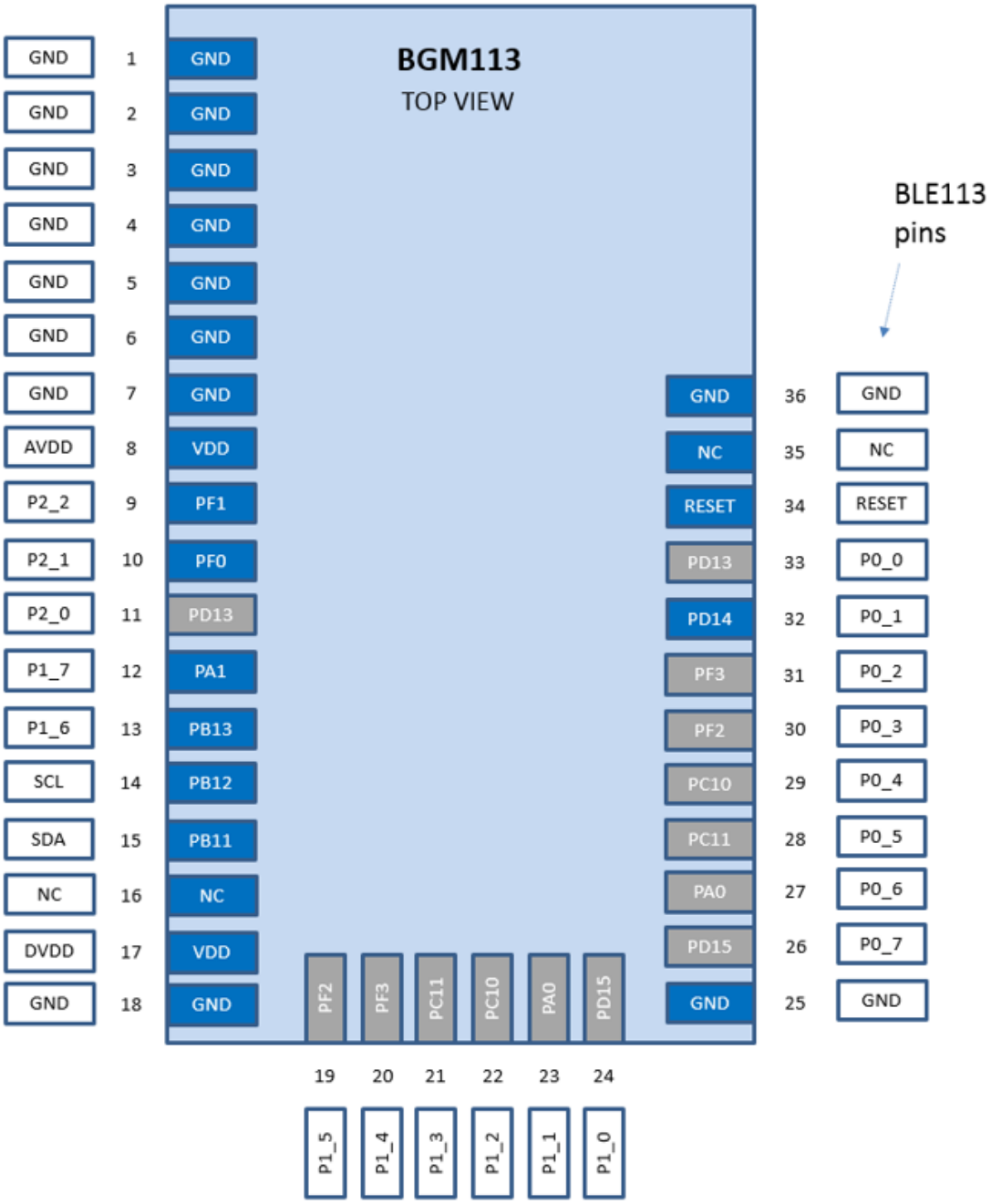


Figure 4.1. BLE113 和 BGM113 引出线

Note: BGM113 中的某些 GPIO 信号连接至多个引脚。这意味着，BGM113 的独立 GPIO 线路数量上限小于 BLE113。在上图中，这些引脚显示为灰色。

4.1.1 调试和编程引脚

调试端口在 BGM113 和 BLE113 上映射至相同的引脚。借此，可对两个模块使用相同的调试连接器。但是，应注意编程接口完全不同。BGM113 使用 ARM SWD 接口，而 BLE113 使用的是 TI CC 调试程序接口。

下表中标出了 BLE113 和 BGM113 的调试程序引脚。

Table 4.1. BLE113 and BGM113 Debugger Pins

Pin Number	BLE113 Pin Name / Function	BGM113 Pin Name / Function
9	P2_2 / PROG_DC	PF1 / SWDIO
10	P2_1 / PROG_DD	PF0 / SWCLK

4.1.2 电源和复位引脚

电源电压连接至引脚 8 和引脚 17。BLE113 设有模拟和数字电源（AVDD、DVDD）的单独引脚，而 BGM113 仅有一个通用于模拟和数字域的电源电压 VDD。

复位引脚为低电平有效，在 BGM113 和 BLE113 中均有内部上拉电阻。

4.1.3 GPIO 引脚

GPIO 信号在 BGM113 和 BLE113 中映射至相同的引脚。但是如 Figure 4.1 BLE113 和 BGM113 引出线 on page 5 所示，某些 GPIO 信号在 BGM113 中映射至两个模块引脚。在用 BGM113 替换 BLE113 时，这有可能会引起某些冲突。

第 7. 附录 — BLE113 和 BGM113 引脚对比 节列出了 BLE113 和 BGM113 中的所有 GPIO 引脚。请注意在 BLE113 中，引脚 14 和 15 为 I^2C 硬连线，但是在 BGM113 中，这些引脚可用于 I^2C，也可以用作 GPIO 信号。

4.1.4 不使用的引脚

不使用的引脚不得作悬空处理。在 BLE113 中，内部上拉电阻或下拉电阻会影响整个端口。例如，端口 P2 中的所有引脚可拉高或拉低。在 BGM113 中，提拉电阻可针对每个引脚单独编程。

4.1.5 带有高电流驱动功能的引脚

BLE113 有两个 20 mA 驱动强度的 I/O 引脚，分别为引脚 **P1_0** 和 **P1_1**。BGM113 没有更高驱动强度的专用引脚。

4.1.6 专用功能引脚

在 BLE113 中，I^2C SCL 和 SDA 信号固定在引脚 14 和引脚 15。在 BGM113 中，I^2C 信号可自由映射至任何可用的 GPIO。它们可像在 BLE113 中一样映射至相同的引脚，实现兼容性。如果不需要 I^2C，应用可将这些引脚用作通用 I/O。

4.2 外围设备

本节介绍了 BLE113 和 BGM113 的外围设备差异。

4.2.1 UART/SPI

BLE113 和 BGM113 有两个 USART 块，可配置为 UART 或 SPI 模式。BLE113 中有两个 USART 引脚的备用引脚映射。信号以一组进行映射，例如所有 UART0 信号都使用 **alt1** 或 **alt2** 映射。

在 BGM113 中，USART 信号可分别传输至任何可用的 GPIO，实现更加灵活的引脚分配。

4.2.2 ADC

BLE113 中有八个可能的引脚用作 ADC 输入，即 **P0_0** – **P0_7**。在 BGM113 中，任何可用的 GPIO 输入可输入到 ADC 中。

ADC 参考

BLE113 中可能的 ADC 参考为：

- 内部参考 (1.24 V)
- AIN7 引脚上的外部参考
- AVDD 引脚
- AIN6 – AIN7 差分输入上的外部参考

在 BGM113 中，可能的 ADC 参考为：

- 内部 1.25 V 参考
- 内部 2.5 V 参考
- 缓冲 VDD
- 内部差分 5 V 参考
- 来自引脚 6 的单端外部参考
- 差分外部参考，2x（引脚 6 至引脚 7）
- 未缓冲 2xVDD

有关详情，请参见 *EFR32xG1 无线 Gecko 参考手册*。

测量电源电压

BLE113 中有一个特殊的 ADC 频道 VDD/3，可用于测量 ADC 的电源电压大小。

在 BGM113 中，无法将下变换 VDD 连接到 ADC 输入。但是，仍然可以设置内部 5 V 差分的 ADC 参考并对 AVDD 频道进行取样，进而测量 VDD。有关详情，请参见 *Bluetooth® Smart Software API 参考手册*中的命令 cmd_hardware_read_adc_channel。

4.2.3 定时器

在 BLE113 中，用户应用可访问三个定时器，分别为定时器 1、定时器 3 和定时器 4。这些定时器具有定时器/计数器/PWM 功能。定时器 1 为 16 位，而定时器 3 和 4 为 8 位。

BLE113 数据表列出了从定时器频道到输出引脚的可能映射。每个定时器有两个备用映射，分别为 alt1 和 alt2。可映射至模块引脚的定时器频道总数为 5+2+2 = 9。

BGM113 带有两个 16 位通用定时器，具备 7 (3+4) 个对比/捕捉频道用于输入捕获和对比/脉冲宽度调制 (PWM)。

在 BGM113 中，定时器频道可映射至任何可用的 GPIO 引脚。它们无需以组的形式映射（如 BLE113 中的 alt1/alt2 映射）。

通常来说，BGM113 中的定时器配置至少与 BLE113 中一样灵活。定时器频道可自由映射至可用引脚，因此可将设计从 BLE113 移植到 BGM113，前提是假设 GPIO 引脚分配无冲突（因为在 BGM113 中，某些信号会像之前讨论的一样传输至多个引脚）。

除了通用的 16 位定时器外，BGM113 还带有两个特殊定时器外围设备 CRYOTIMER 和 LETIMER。

CRYOTIMER 是一个 32 位计数器，在低频振荡器上工作，能够在所有能源模式下运行，并且可从睡眠、甚至最深睡眠模式 EM4 唤醒模块。

LETIMER 是一个 16 位逐减计数器，在 32.768 Hz 时钟上运行。它能够跟踪时间和输出可配置波形。LETIMER 可下至睡眠模式 EM2。

Note: 在写入时，BGM113 中的定时器函数（基础定时器和 LETIMER、CRYOTIMER）仅可在基于 C 的应用中访问，使用 EMLIB 驱动程序直接访问 MCU 外围设备。定时器函数（例如 PWM 生成）无法通过 BGAPI 或 BGScript 访问。

5. 软件相关差异

本节介绍了 BLE113 和 BGM113 之间的软件相关差异。

5.1 转换项目配置文件

本节介绍了如何将 BGScript™ 项目配置文件从 BLE113 转换至 BGM113 格式。

BLE113 和 BGM113 的这套项目结构相同：

- *Project_name.bgproj*（定义设备类型和所含 XML 文件列表）
  - *hardware.xml*（hw 配置）
  - *gatt.xml*（GATT 数据库定义）
  - *script.bgs*（BGScript 脚本）

下表显示了两个示例项目文件（**.bgproj*），左侧为 BLE113，右侧为 BGM113。

Table 5.1. BLE113 and BGM113 Debugger Pins

BLE113 Project File	BGM113 Project File
<?xml version="1.0" encoding="UTF-8" ?> <project> <gatt in="gatt.xml" /> <hardware in="hardware.xml" /> <script in="keyboard.bgs" /> <image out="out-ble113.hex" /> <device type="ble113" /> <boot fw="bootuart" /> </project>	<?xml version="1.0" encoding="UTF-8" ?> <project device="bgm113"> <!-- GATT service database --> <gatt in="gatt.xml" /> <!-- Local hardware configuration file --> <hardware in="hardware.xml" /> <!-- BGScript source code --> <script in="keyboard.bgs" /> <!-- Firmware output file --> <image out="bgm113demo.bin" /> </project>

在将项目文件从 BLE113 移植到 BGM113 时，需要考虑某些差异：

- 在 BGM113 中，目标设备类型在 <project> 标记中定义
- BGScript 源文件位于 BGM113 项目的 <scripting> 标记中

Note: 对于 BGM113，<boot> 选项不可用。

5.2 硬件配置文件

下面的子章节介绍了 BLE113 和 BGM113 在硬件配置方面的差异。

5.2.1 GPIO 配置

在 BLE113 项目中，<port> 标记用于配置仅对 GPIO 输入有效的 GPIO 设置。下例来源于 BLE113 项目：

示例：使用 <port> 标记配置 GPIO 设置

```
<port index="0" tristatemask="0" pull="down" />
```

上一行将库 0 中所有引脚的拉动方向配置为“向下”。请注意，拉动方向针对整个库定义，无法在同一个库中同时使用上拉和下拉。Tristatemask 可用于禁用库中所选引脚的提拉电阻。

在 BGM113 中，GPIO 配置完全不同。下例展示了如何配置两个输出引脚（PA6，PA3）和一个输入引脚（PF6）：

示例：配置两个输出引脚（PA6，PA3）和一个输入引脚（PF6）

```
<gpio port="A" pin="6" mode="pushpull" out="1" />
<gpio port="A" pin="3" mode="pushpull" out="0" />
<gpio port="F" pin="6" mode="input" out="0" interrupt="both" />
```

在 BGM113 项目中，可在 *hardware.xml* 文件中配置输入和输出引脚设置。上拉电阻针对每个引脚单独定义，对于 GPIO 输出，还可以定义初始状态。

输入和输出的基本 GPIO 设置还可在运行时使用 BGScript 命令或 BGAPI 命令进行配置。函数为 hardware_configure_gpio，请参见 *Bluetooth® Smart Software API 参考手册*了解详情。

可用 GPIO 引脚配置选项的详细说明载于 *UG119：Blue Gecko Bluetooth® Smart Device 配置指南*。

5.2.2 UART 配置

BGAPI 命令或应用特定输入/输出的 UART 基本用法在 BLE113 和 BGM113 中类似。但在 UART 配置中，将 BLE113 项目移植到 BGM113 时需要考虑几个不同点。

以下为 UART 配置的几个例子，第一个例子针对 BLE113，第二个例子针对 BGM113：

示例：BLE113 的 UART 配置

```
<usart channel="1" alternate="1" baud="115200" endpoint="none" />
```

示例：BGM113 的 UART 配置

```
<uart index="1" baud="115200" flowcontrol="false" bgapi="false" />
```

BLE113 和 BGM113 之间的 UART 配置差异如下：

- 标记名称在 BLE113 中为 <usart>，而在 BGM113 中则为 <uart>
- BLE113 有参数“频道”，BGM113 有参数“索引”，用于选择 UART 实例
- BLE113 中的“Flow”参数在 BGM113 中为“Flowcontrol”

如需 UART 配置选项的详细列表，请参见 CONFIG_GUIDE。

请注意，上文中的 BLE113 UART 示例有“替代”参数，用于在两个替代性引脚映射之间选择。BGM113 中没有此参数，因为 UART 引脚可单独映射至任何可用的 GPIO。

以下是一个 BGM113 的 UART 配置示例，其明确指定了 **RTS** 和 **CTS** 引脚的位置：

示例：明确定义 RTS 和 CTS 引脚的 BGM113 UART 配置

```
<uart index="1" baud="115200" flowcontrol="true" rts_pin="PA3" cts_pin="PA2" bgapi="true"/>
```

5.2.3 唤醒引脚

BLE113 和 BGM113 中的唤醒引脚功能相似。在如何定义唤醒引脚的句法方面略有不同。

示例：BLE113 的唤醒引脚定义（引脚 P0_0）

```
<wakeup_pin enable="true" port="0" pin="0" state="up" />
```

示例：BGM113 的唤醒引脚定义（引脚 PD13）

```
<wake_up port="d" pin="13" state="up" />
```

在功能上，上面两个例子相同：其将唤醒引脚配置为 P0_0 / PD13，并将极性设置为正高。只有句法略有不同。有关完整详情，请参见 *UG119：Blue Gecko Bluetooth® Smart Device 配置指南*。

5.2.4 主机唤醒引脚

主机唤醒引脚在 BLE113 和 BGM113 中的功能类似。配置此引脚的句法略有不同，参见下例。有关完整详情，请参见 *UG119：Blue Gecko Bluetooth® Smart Device 配置指南*。

示例：在 BLE113 中将 P1_7 配置为主机唤醒引脚

```
<host_wakeup_pin enable="true" port="1" pin="7" state="up" />
```

示例：在 BGM113 中将 PA1 配置为主机唤醒引脚：

```
<host_wake_up port="a" pin="1" state="up" />
```

5.2.5 睡眠启用

在 BLE113 和 BGM113 中配置睡眠启用的方法几乎完全一致。差别在于，BGM113 不使用属性 max_mode。下例显示了如何分别启用 BLE113 和 BGM113 的睡眠。

示例：BLE113 启用睡眠（最大模式 2，电源模式 3 不启用）

```
<sleep enable="true" max_mode="2" />
```

示例：BGM113 启用睡眠（最大模式暗指 EM2）

```
<sleep enable="true" />
```

5.2.6 TX 功率设置

在 BLE113 项目中，可使用 <txpower> 标记在硬件配置文件中设置 TX 功率。另外，在运行时调用函数 hardware_set_txpower 可动态设置 BLE113 TX 功率。

示例：运行时在 BLE113 中设置最大功率 (0dBm)

```
call hardware_set_txpower(14)
```

在 BGM113 中，配置文件**无选项用于静态配置 TX 功率**。您需要使用函数 system_set_tx_power 动态设置 TX 功率。有关详情，请参见 *Bluetooth® Smart Software API 参考手册*。

示例：在运行时在 BGM113 中设置最大功率 (+3 dBm)

```
call system_set_tx_power(30)
```

调整 TX 功率时请注意单位差异。BLE113 使用 0..14 值，其中 0 为最低值（约 -24 dBm），14 为最高设置，约等于 0 dBm。

在 BGM113 中，参数步幅为 0.1 dBm。例如，设置 15 对应 1.5 dBm。函数 system_set_tx_power 将返回协议栈使用的实际功率设置值。请注意，功率无法以 0.1 dBm 的步幅调整，协议栈将使用最接近用户请求值的预定义功率。

5.2.7 过时的 HW 相关设置

以下 HW 相关设置不可用于 BGM113：

- <sleeposc>
- <slow_clock>
- <lock_debug>
- <pmux>

6. 将应用代码从 BLE113 移植到 BGM113

本章介绍了 BLE113 和 BGM113 的 BGAPI 命令之间的差异。BLE113 和 BGM113 协议栈的某些 BGAPI 命令几乎完全相同。另一方面，BGM113 中的某些部分（例如 GATT 服务发现）与 BLE113 有显著差异。

本章将介绍下列主题：

- System_boot 事件
- 启用广播和更改广播数据/参数
- 打开/关闭连接
- 发送通知和指示
- 软定时器
- 启用发现
- 连接至外围设备
- GATT 服务发现
- 订阅通知

6.1 系统启动

在 BLE113 和 BGM113 中，“system_boot”是在模块重启后出现的第一个事件。BGScript 原型设计显示如下：

```
# BLE113:
event system_boot(major, minor, patch, build, ll_version, protocol_version, hw)
# BGM113:
event system_boot(major, minor, patch, build, bootloader, hw)
```

在两种情况下，识别协议栈版本的前四个参数（例如 1.0.2-755）类似。与 HW 版本等相关的最后几个参数略有不同。

6.2 启用广播

BLE113 和 BGM113 中启用和停用广播的方法基本相同，都是使用命令 gap_set_mode：

```
# BLE113:
call gap_set_mode(discover, connect)(result)
# BGM113:
call le_gap_set_mode(discover,connect)(result)
```

唯一的不同点在于 BGM113 命令的名称中有前缀“le_”（“le”代表低能耗）。大部分 BGM113 命令和事件均采用这种惯例。

在 BLE113 和 BGM113 中，设置广播参数和广播数据的方法几乎相同。函数和参数相同（BGM113 的“le_”前缀除外）：

```
# BLE113:
call gap_set_adv_parameters(adv_interval_min, adv_interval_max, adv_channels)(result)
call gap_set_adv_data(set_scanrsp, adv_data_len, adv_data_data)(result)
# BGM113:
call le_gap_set_adv_parameters(interval_min, interval_max, channel_map)(result)
call le_gap_set_adv_data(scan_rsp,adv_data_len, adv_data_data)(result)
```

6.3 连接状态更改

表示连接状态改变的 BGAPI 事件在 BLE113 和 BGM113 中完全不同。这可能需要对接收传入连接的应用进行一些移植操作。

在 BLE113 中，连接状态改变以带有以下原型设计的一个事件“connection_status”表示：

```
# BLE113:
event connection_status(connection, flags, address, address_type, conn_interval, timeout, latency, bonding)
```

根据 Flags 参数，应用可解码出现该事件的原因。

BGM113 中没有 connection_status 事件。连接状态改变通过用途更加具体的几个不同事件来表示（例如，处理传入连接）。BGM113 协议栈中的一些相关事件如下：

```
# BGM113:
event le_connection_opened(address,address_type,master,connection,bonding)
event le_connection_closed(reason,connection)
event le_connection_parameters(connection,interval,latency,timeout,security_mode)
```

在将应用从 BLE113 移植到 BGM113 时，connection_status 事件处理程序中的功能需要移至 BGM113 API 提供的合适事件中。

6.4 发送通知或指示

很多应用使用通知或指示向客户端推送数据。BLE113 和 BGM113 中用于发送通知的 API 函数如下。

```
# BLE113:
call attributes_write(handle, offset, value_len, value_data)(result)

# BGM113:
call gatt_server_send_characteristic_notification(connection,characteristic,value_len, value_data)(result)
# Optional:
call gatt_server_write_attribute_value(attribute, offset, value_len, value_data)(result)
```

BLE113 和 BGM113 中实施通知的方式存在着基本差别。在 BLE113 中，应用只会使用 attributes_write 命令写入本地 GATT 数据库。如果某个已连接的客户端订阅了该属性的通知，协议栈则会自动生成通知。

BGM113 中没有自动触发通知。应用可在本地数据中使用命令 gatt_server_write_attribute_value 更新该值。要触发通知，应用必须明确使用命令 gatt_server_send_characteristic_notification。

在很多情况下，如果应用需要发送通知，且客户端无需能够从数据库中读取值，那么只需用 gatt_server_send_characteristic_notification 替代 **attribute_write** 即可。

上述说明解释了 BGM113 如何生成通知，请注意这同样适用于*指示*。

6.5 软定时器

配置软定时器的 API 调用在 BLE113 和 BGM113 中相同。

```
call hardware_set_soft_timer(time, handle, single_shot)(result)
```

对于这两个模块，定时器间隔已在硬件时钟节拍中定义，1 秒等于 32768 节拍。

在 BLE113 中，一次只能有一个重复定时器。如果您启动新的重复定时器，现有的重复定时器将被删除（如果存在此类定时器）。在 BGM113 中，重复软定时器的数量无上限。

6.6 启用发现

BLE113 和 BGM113 用于设置扫描参数、开始发现和结束发现的 API 调用如下。原型设计完全相同，但是 BGM1113 会使用“le_”前缀。

```
# BLE113:
call gap_set_scan_parameters(scan_interval, scan_window, active)(result)
call gap_discover(mode)(result)
call gap_end_procedure()(result)

# BGM113:
call le_gap_set_scan_parameters(scan_interval,scan_window,active)(result)
call le_gap_discover(mode)(result)
call le_gap_end_procedure()(result)
```

BLE113 和 BGM113 的默认扫描参数不同。如果应用不明确调用 gap_set_scan_parameters，而是使用协议栈默认值，则必须将这一点考虑在内。在 BLE113 中，扫描间隔和窗口的默认值为 (75, 50) ms，在 BGM113 中，默认值为 (10,10) ms。

针对每个扫描响应出现的事件在两个模块中类似：

```
# BLE113：事件 gap_scan_response(rssi, packet_type, sender, address_type, bond, data_len, data_data) # BGM113：事件 le_gap_scan_respo
nse(rssi, packet_type, address, address_type, bonding, data_len, data_data)
```

请注意：BLE113 和 BGM113 的数据包类型枚举略有不同。

BLE113：

- **0**：可连接的广播数据包
- **2**：不可连接的广播数据包
- **4**：扫描响应数据包
- **6**：可发现的广播数据包

BGM113：

- **0**：可连接的无向广播
- **2**：可扫描的无向广播
- **3**：不可连接的无向广播
- **4**：扫描响应

6.7 打开连接

BLE113 和 BGM113 中用于打开广播设备连接的 API 调用有显著差异，函数原型设计如下：

```
# BLE113:
call gap_connect_direct(address, addr_type, conn_interval_min, conn_interval_max, timeout, latency)(result, connection_handle)

# BGM113:
call le_gap_open(address,address_type)(result,connection)
```

在 BLE113 中，连接参数作为 gap_connect_direct 的参数通过，而在 BGM113 中，函数 le_gap_open 仅采用目标地址和地址类型作为参数。

在 BGM113 中，如果您要更改连接参数，则可使用下列 API 调用：

```
# BGM113:
call le_gap_set_conn_parameters(min_interval,max_interval,latency,timeout)(result)
call le_connection_set_parameters(connection,min_interval,max_interval,latency,timeout)(result)
```

这些函数用于设置您在 BLE113 的 gap_connect_direct 命令中定义的参数。上述两个 API 调用的不同之处在于，le_gap_set_conn_parameters 设置用于所有后续连接的默认参数，而 le_connection_set_parameters 则用于修改已打开的单个连接的参数。

连接打开后生成的事件如下：

```
# BLE113:
event connection_status(connection, flags, address, address_type, conn_interval, timeout, latency, bonding)

# BGM113:
event le_connection_opened(address,address_type,master,connection,bonding)
event le_connection_parameters(connection,interval,latency,timeout,security_mode)
```

6.8 关闭连接

应用可能会意外关闭（例如由于监视超时）或故意关闭连接。

如果发生意外连接关闭，BLE113 和 BGM113 中则会出现下列事件：

```
# BLE113:
event connection_disconnected(connection, reason)

#BGM113:
event le_connection_closed(reason,connection)
```

上述两个事件包含相同的信息，仅有句法不同。

如果应用需要关闭激活的连接，BLE113 和 BGM113 则会使用下列 API 调用：

```
# BLE113:
call connection_disconnect(connection)(connection, result)

# BGM113:
call endpoint_close(endpoint)(result,endpoint)
```

同样，基本用法相同，仅有句法不同。

6.9 GATT 服务发现

BLE113 和 BGM113 完成服务发现的方式有很多不同之处。下文概述了与两个模块的服务发现相关的典型 API 调用和事件。我们无法在这里解释所有差别，该列表旨在向读者指出 BGAPI 文档中的相关部分，帮助将其应用从 BLE113 移植到 BGM113。

在 BLE113 中查找服务：

1. 调用 attclient_read_by_group_type。
2. 这会生成类型 attclient_group_found 的事件（每个主要或次要服务一个）。
3. 最终事件后，协议栈将产生事件 attclient_procedure_completed。

在 BLE113 中查找特定服务的属性：

1. 调用 attclient_find_information（句柄范围作为参数通过）。
2. 这会生成类型 attclient_find_information_found 的事件。
3. 最终事件后，协议栈将产生事件 attclient_procedure_completed。

在 BGM113 中查找服务：

1. 调用 gatt_discover_primary_services（或 gatt_discover_primary_services_by_uuid）。
2. 这会生成类型 gatt_service 的事件（每项服务一个）。
3. 最终事件后，协议栈将产生事件 gatt_procedure_completed。

在 BGM113 中查找特定服务的属性：

1. 调用 gatt_discover_characteristics（服务句柄作为参数通过）。
2. 这会生成类型 gatt_characteristic 的事件。
3. 最终事件后，协议栈将产生事件 gatt_procedure_completed。

6.10 订阅通知或指示

要订阅通知（或指示），GATT 客户端需要将值 0x0001（或 0x0002）写入某个特性的客户端特性配置 (CCC)。一般来说，CCC 句柄值相对于特性本身步进 +1 或 +2。（请注意，客户端不得假设此情况）

在 BLE113 中，调用 attclient_attribute_write 直接更改 CCC 值便可完成这一点。客户端需要知道与 CCC 相关的句柄，这可在服务发现过程中检测（见上文）。

在 BGM113 中，客户端不需要知道 CCC 句柄。它会调用 gatt_set_characteristic_notification 并将特性值（非 CCC）传递至函数。协议栈将自动发现特性客户端配置。

函数 gatt_set_characteristic_notification 用于启用/禁用两个通知和指示，通过传递至函数的“Flags”参数进行选择。有关详情，请参见 *Bluetooth® Smart Software API 参考手册*。

7. 附录 — BLE113 和 BGM113 引脚对比

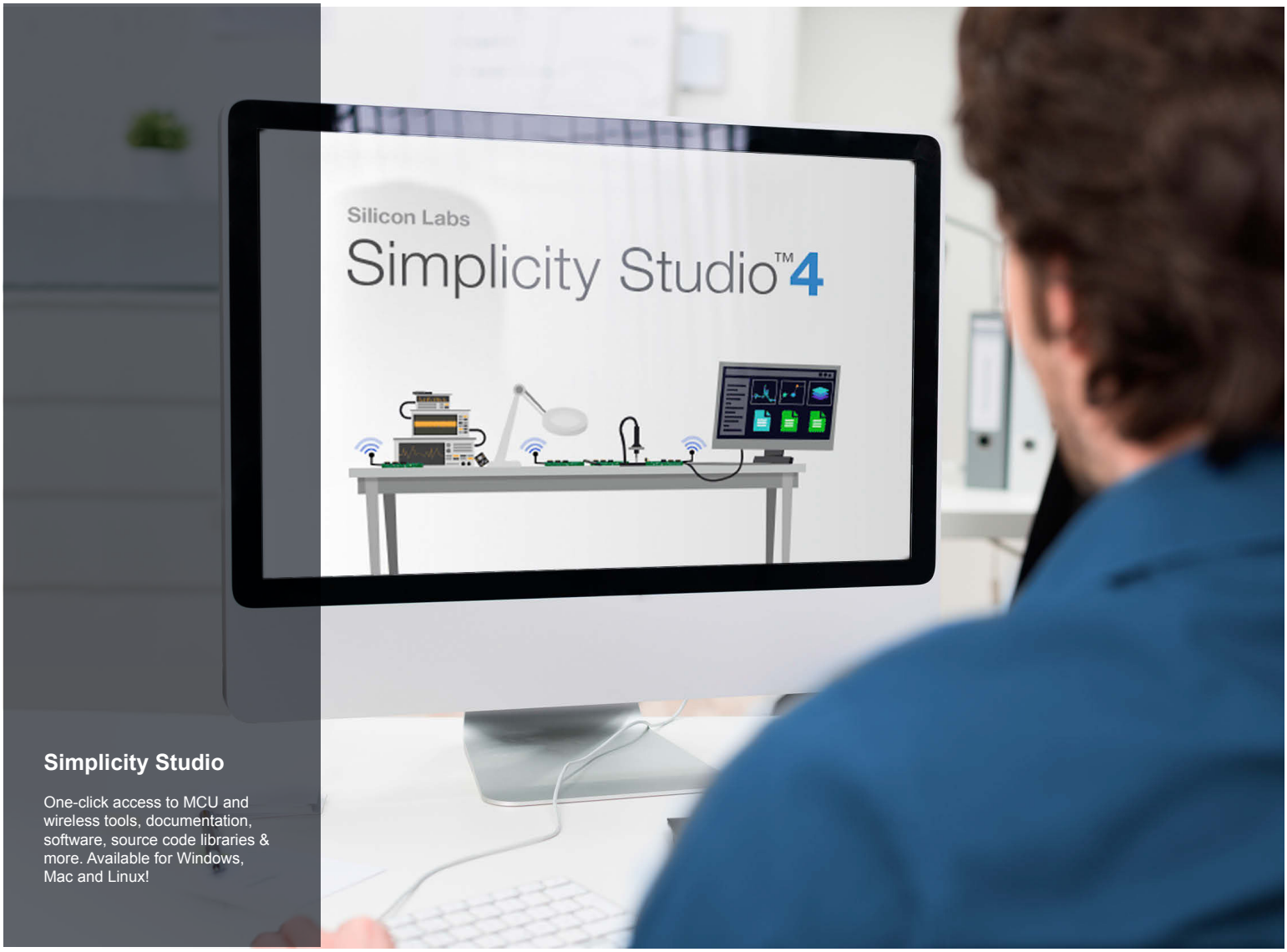
下表列出了 BLE113 和 BGM113 的所有引脚。

Table 7.1. BLE113 and BGM113 Pin Comparison

Pin Number	BLE113	BLE113 Notes	BGM113	BGM113 Notes
9	P2_2		PF1	
10	P2_1		PF0	
11	P2_0		PD13	Same as pin 33
12	P1_7		PA1	
13	P1_6		PB13	
14	SCL	I^2C only	PB12	
15	SDA	I^2C only	PB11	
19	P1_5		PF2	Same as pin 30
20	P1_4		PF3	Same as pin 31
21	P1_3		PC11	Same as pin 28
22	P1_2		PC10	Same as pin 29
23	P1_1	20 mA	PA0	Same as pin 27
24	P1_0	20 mA	PD15	Same as pin 26
26	P0_7		PD15	Same as pin 24
27	P0_6		PA0	Same as pin 23
28	P0_5		PC11	Same as pin 21
29	P0_4		PC10	Same as pin 22
30	P0_3		PF2	Same as pin 19
31	P0_2		PF3	Same as pin 20
32	P0_1		PD14	
33	P0_0		PD13	Same as pin 11

Notes:

1. BLE113 max number of GPIO lines is 19 (+ 2 dedicated pins for I2C).
2. BGM113 max number of GPIO lines is 14.



Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com