Application Note

从 STM32® 到基于 Arm® 的 MSPM0 的迁移指南

摘要

本应用手册可协助您从 STMicroelectronics STM32® 平台迁移至德州仪器 (TI) MSPM0 MCU 生态系统。本指南介绍了 MSPM0 开发和工具生态系统、内核架构、外设注意事项以及软件开发套件。目的是突出两个系列之间的差异，并利用现有的 STM32 生态系统知识快速提升 MSPM0 系列 MCU 的性能。

内容

商标

1 MSPM0 产品系列概述

1.1 引言

MSP430™ MCU 作为 TI 的经典微控制器拥有近 30 年的历史。最新一代推出了 MSPM0 系列。MSPM0 微控制器 (MCU) 是 MSP 高度集成的超低功耗 32 位 MCU 系列的一部分，基于增强型 Arm® Cortex®-M0+ 32 位内核平台。这些成本优化型 MCU 提供高性能模拟外设集成，支持扩展的工作温度范围并提供小尺寸封装。TI MSPM0 系列低功耗 MCU 包含具有不同模拟和数字集成度的器件，使工程师能够找到满足其工程需求的 MCU。MSPM0 MCU 系列将 Arm Cortex-M0+ 平台与超低功耗系统架构相结合，使系统设计人员能够在降低能耗的同时提高性能。

MSPM0 MCU 是 STM32 MCU 的替代产品，具有很强的竞争力。本应用手册通过比较器件功能和生态系统来帮助从 STM32 MCU 迁移到 MSPM0 MCU。

1.2 STM32 MCU 与 MSPM0 MCU 的产品系列比较

表 1-1. TI MSPM0Gx/LX 和 STM32G0/F0 系列的比较

	ST Micro STM32G0 系列	ST Micro STM32F0 系列	TI MSPM0 MSPM0Gx 系列	TI MSPM0 MSPM0Lx 系列
内核/频率	CM0+/64MHz	CM0/48MHz	CM0+/80MHz	CM0+/32MHz
电源电压	1.7 V 至 3.6 V	2 V 至 3.6V	1.62V 至 3.6V	1.62 V 至 3.6 V
温度	-40°C 至 125°C	-40°C 至 105°C	-40°C 至 125°C	-40°C 至 125°C
存储器	512KB 至 16KB	256KB 至 16KB	128KB 至 32KB	64KB 至 8KB
RAM	高达 144KB	高达 32KB	高达 32KB	高达 4KB
GPIO (最大)	90	88	60	28
模拟	1 个 2.5Msps 12 位 ADC 1 个 12 位 DAC 3 个比较器	1 个 1Msps 12 位 ADC 1 个 12 位 DAC 2 个比较器	2 个 4Msps 12 位 ADC 1 个 12 位 DAC 3 个高速比较器 2 个运算放大器	1 个 1Msps 12 位 ADC 1 个高速比较器 2 个运算放大器
通信 (最大)	3 个 SPI 3 个 I²C 超快速 6 个 UART (LIN) 2x CAN-FD 1 个 USB	2x SPI 2 个 I²C 超快速 8 个 UART (LIN) 1x CAN	2x SPI 2 个 I²C 超快速 4 个 UART (LIN) 1 个 CAN-FD	1x SPI 2 个 I²C 超快速 2 个 UART (LIN)
计时器	8	4	7	4
高级计时器	是 (1 个)	是 (1 个)	是 (3 个)	否
硬件加速器	不适用	不适用	可选	不适用
安全性	CRC、TRNG、AES256	CRC	CRC、TRNG、AES256	CRC
低功耗	有效：100µA/MHz 待机 (RTC)：1.5µA	有效：281µA/MHz 待机 (RTC)：2.5µA	有效：85µA/MHz 待机 (RTC)：1.5µA	有效：85µA/MHz 待机：1.5µA

2 生态系统和迁移

MSPM0 MCU 由广泛的硬件和软件生态系统提供支持，随附参考设计和代码示例，便于您快速开始设计。MSPM0 MCU 还具有在线资源、MSP Academy 培训支持和 TI E2E™ 支持论坛提供的在线支持。

2.1 软件生态系统比较

表 2-1. 与 STM32 软件工具等效的 MSPM0 产品

	STM32	MSPM0
IDE	CubeIDE	Code Composer Studio™ IDE (CCS)
软件配置	CubeMX	SysConfig
独立编程	CubeProgrammer	UniFlash
显示/演示 GUI 编辑器	CubeMonitor	GuiComposer

2.1.1 MSPM0 软件开发套件 (MSPM0 SDK)

MSPM0 SDK 提供软件 API、示例、文档和库，可帮助工程师在德州仪器 (TI) MSPM0+ 微控制器器件上快速开发应用程序。提供了各种示例来展示如何在每个受支持的器件上使用各功能区，这些示例可用作您开发自己的工程的起点。此外，MSPM0 SDK 中还包含交互式 MSP Academy 培训，以提供引导式学习路径。

示例文件夹分为 RTOS 和非 RTOS 子文件夹（目前仅支持非 RTOS）。这些文件夹包含每个 LaunchPad™ 开发套件的示例，并且按类别有序分类，例如较低级别的 DriverLib 示例、较高级别的 TI 驱动程序示例以及 GUI Composer、LIN、IQMath 等中间件示例等等。有关详细信息，请参阅 *MSPM0 SDK 用户指南*。

2.1.2 CubeIDE 与 Code Composer Studio IDE (CCS)

Code Composer Studio IDE (CCS) 是与 STM32 的 CubeIDE 等效的 TI 产品。CCS 是一款基于 Eclipse 的免费 IDE，支持 TI 的微控制器 (MCU) 和嵌入式处理器产品系列。CCS 包含一整套用于开发和调试嵌入式应用程序的工具，其中包含优化的 C/C++ 编译器、源代码编辑器、工程构建环境、调试器、性能评测工具和许多其他功能。CCS 可作为桌面或基于云的 IDE 提供。

CCS 集成了 SysConfig 的 MSPM0 器件配置和自动代码生成功能，并在集成式 TI Resource explorer 中集成了 MSPM0 代码示例和 Academy 培训。CCS 提供一体式开发工具体验。

除 CCS 之外，下表中列出的业界通用 IDE 也支持 MSPM0 器件。

表 2-2. MSPM0 支持的 IDE

IDE	MSPM0
CCS	✓
IAR	✓
Keil	✓

2.1.3 CubeMX 与 SysConfig

SysConfig 是一个直观而全面的图形实用程序集合，用于配置引脚、外设、无线电、子系统和其他组件。它是与 STM32 CubeMX 等效的 TI 产品。SysConfig 可帮助您直观地管理、发现和解决冲突，以便您有更多时间创建差异化应用程序。该工具的输出包括 C 头文件和代码文件，这些文件可与 MSPM0 SDK 示例配合使用，或用于配置定制软件。SysConfig 集成在 CCS 中，但也可以用作独立的程序。

有关详细信息，请参阅 MSPM0 SysConfig 指南。

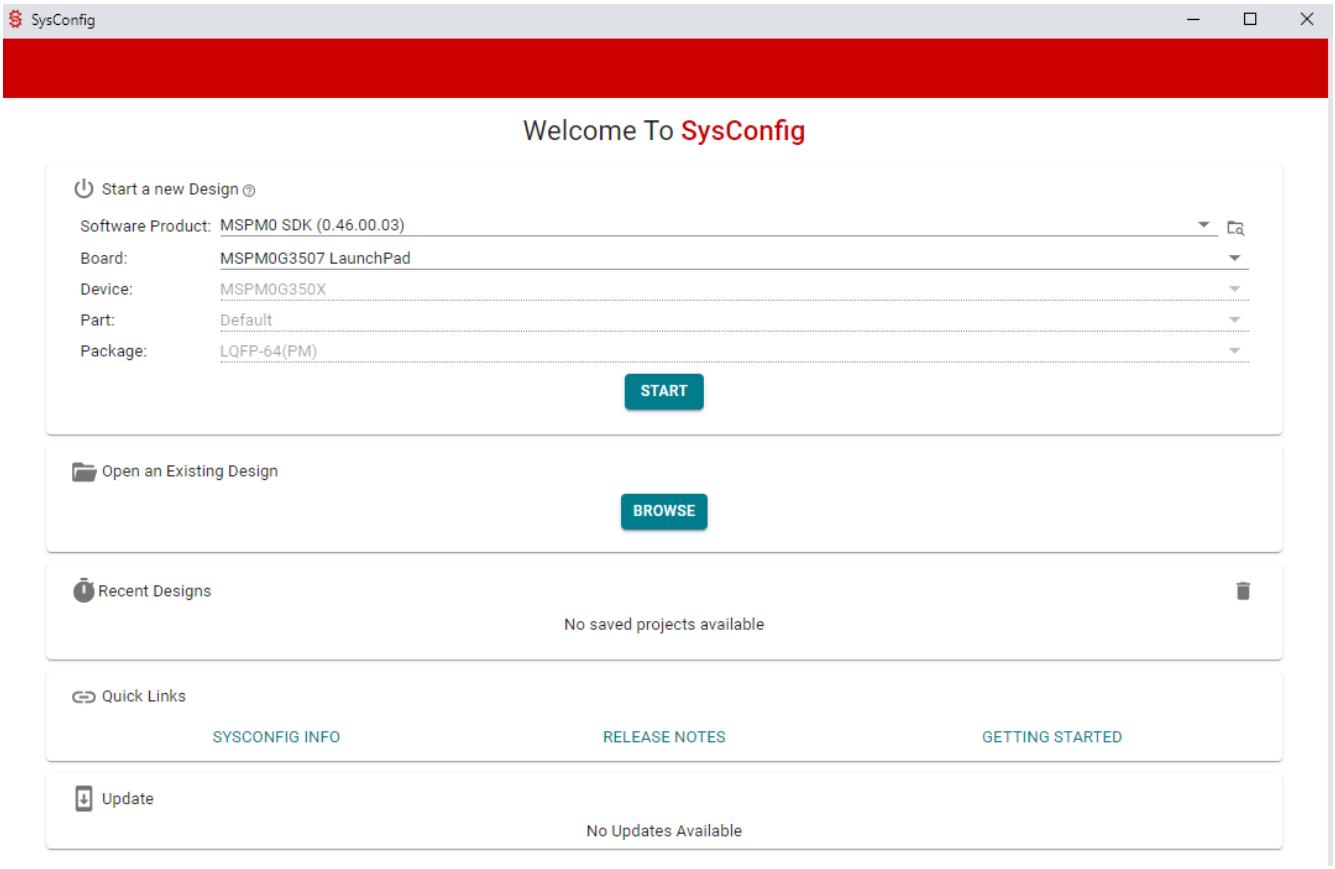

图 2-1. MSPM0 SysConfig

2.2 硬件生态系统

LaunchPad 开发套件是适用于 MSPM0 的唯一评估模块。LaunchPad 套件是易于使用的 EVM，其中包含在 MSPM0 上开始开发所需的一切内容。这包括一个板载调试探针，用于使用 EnergyTrace™ 技术进行编程、调试和测量功耗。MSPM0 LaunchPad 套件还具有板载按钮、LED 和温度传感器以及其他电路。40 引脚 BoosterPack™ 插件模块接头简化了快速原型设计，支持市面上的多种 BoosterPack 插件模块。您可快速添加无线连接、图形显示、环境检测等功能。

- LP-MSPM0G3507 LaunchPad 开发套件
- LP-MSPM0L1306 LaunchPad 开发套件

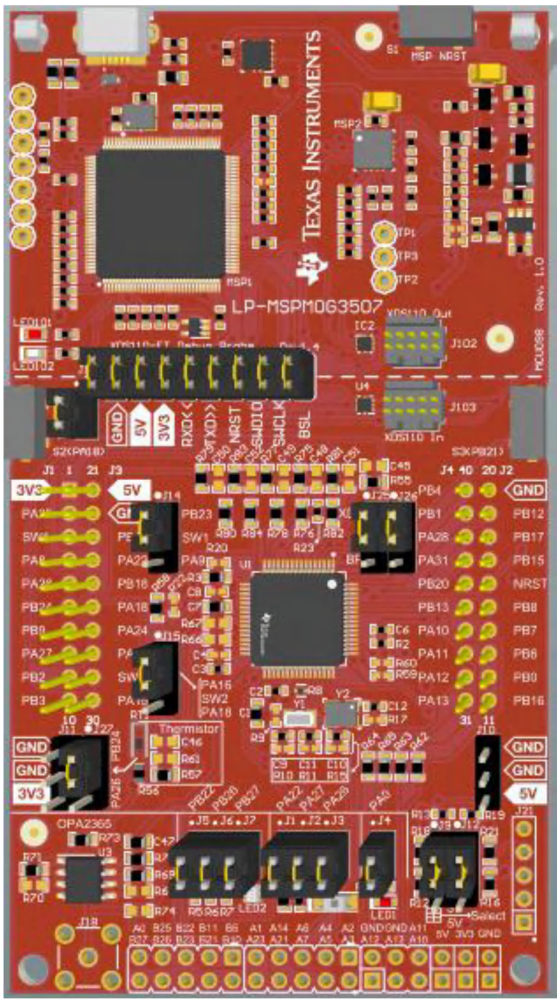

图 2-2. LP-MSPM0G3507 LaunchPad 开发套件

2.3 调试工具

调试子系统 (DEBUGSS) 将串行线调试 (SWD) 两线制物理接口连接到器件内的多个调试功能。MSPM0 器件支持调试处理器执行情况、器件状态和电源状态（使用 EnergyTrace 技术）。图 2-3 显示了调试器的连接。

MSPM0 支持用于标准串行线调试的 XDS110 和 J-Link 调试器。

德州仪器 (TI) XDS110 专为 TI 嵌入式处理器而设计。XDS110 通过 TI 20 引脚连接器（带有用于 TI 14 引脚以及 Arm 10 引脚和 Arm 20 引脚的多个适配器）连接到目标板，并通过 USB2.0 高速 (480Mbps) 连接到主机 PC。它在单个仓体中支持更广泛的标准（IEEE1149.1、IEEE1149.7、SWD）。所有 XDS 调试探针都支持所有具有嵌入式跟踪缓冲器 (ETB) 的 Arm 和 DSP 处理器中的内核和系统跟踪。有关详细信息，请参阅 XDS110 调试探针。

J-Link 调试探针是优化调试和闪存编程体验的最常见的选择。这得益于其创纪录的闪存加载程序、高达 3MiB/s 的 RAM 下载速度以及在 MCU 闪存中设置无限数量断点的能力。J-Link 还支持各种 CPU 和架构，包括 CortexM0+。有关详细信息，请访问 Segger J-Link 调试探针页面。

图 2-3 显示了连接到 MSPM0 目标的 XDS110 探针的主要功能区域和接口的简要示意图。

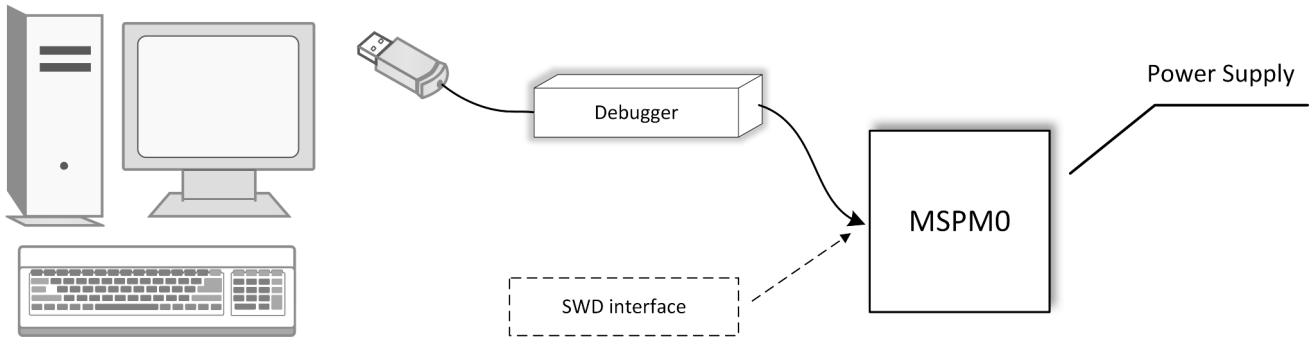

图 2-3. MSPM0 调试

2.4 迁移过程

迁移的第一步是查看产品系列并选择最佳的 MSPM0 MCU。选择 MSPM0 MCU 后，选择开发套件。开发套件包括可供购买的 LaunchPad 套件和适用于目标插座板的设计文件。TI 还提供免费的 MSPM0 软件开发套件 (SDK)，该套件在 TI Resource Explorer 中作为 Code Composer Studio IDE 桌面版和云版组件提供。可使用本应用手册的外设部分来帮助将软件从 STM32 移植到 MSPM0。最后，软件移植后，使用我们的调试工具下载并调试应用程序。

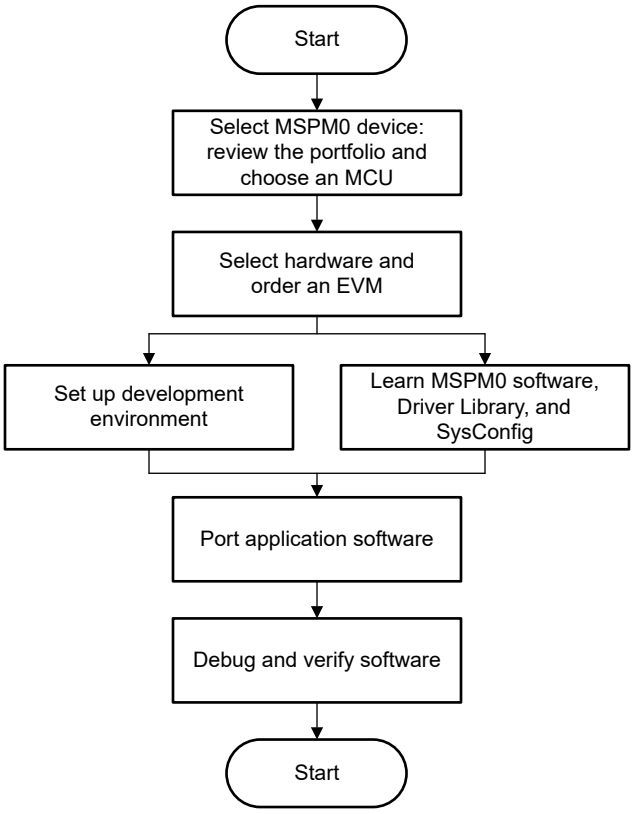

图 2-4. MSPM0 迁移流程图

2.5 迁移和移植示例

为了让您更加熟悉 TI 生态系统并更好地开始使用 MSPM0，本节说明了基本应用程序的逐步迁移过程。

为了演示从 STM32 移植到 MSPM0 的过程，本说明使用现有 ST UART 示例为起点，来介绍将基本低功耗 UART 监控器应用程序从 STM32G0x 移植到 MSPM0 器件的步骤。

步骤 1.选择合适的 MSPM0 MCU

迁移的第一步是为应用程序选择正确的 MSPM0 器件。为此，可使用本指南的产品系列部分选择 MSPM0 系列。可以使用产品选择工具将范围缩小至特定的器件。STM32G0 和 MSPM0 都具有 M0+ 内核，但还必须考虑内存大小、功耗和关键外设等功能。MSPM0 还提供许多引脚对引脚可扩展选项，从而能够轻松扩展到存储器更大或更小的器件，而无需更改系统中的任何其他内容。

在本示例中，我们选择了 MSPM0G3507 作为该应用的最佳器件。

步骤 2.选择硬件并订购 EVM

使用评估模块 (EVM) 可以加快迁移过程。对于 MSPM0 MCU，LaunchPad 套件是最容易上手的硬件。LaunchPad 套件易于使用，因为它们附带内置编程器，旨在实现快速开发。

MSPM0G3507 具有可用于移植软件的 LaunchPad 开发套件 (LP-MSPM0G3507)。

步骤 3.设置软件 IDE 和 SDK

在移植软件之前，必须选择并设置软件开发环境。节 2.1 显示了 MSPM0 支持的所有 IDE。对于所选的任何 IDE，迁移和移植过程都是类似的。应使用最新版本的 MSPM0 SDK。

在本示例中，TI 的 CCS 是所选的 IDE。

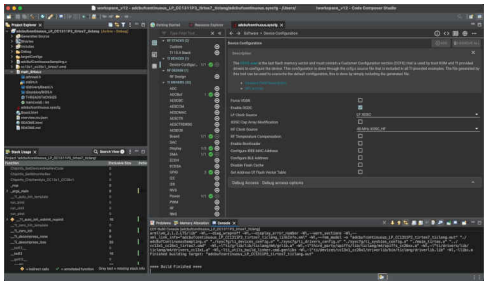

图 2-5. Code Composer Studio IDE

步骤 4.软件移植

准备好环境后，开始使用 MSPM0 SDK。如前所述，MSPM0 SDK 与 STM32Cube 软件包类似。MSPM0 SDK 为软件开发提供了不同的层。MSPM0 TI 驱动程序的运行级别与 STM32Cube HAL 类似，而 MSPM0 DriverLib 与 STM32Cube 低级别驱动程序相当。大多数 MSPM0 用户发现 DriverLib 级别的软件最适合其应用程序，因此大多数 MSPM0 软件示例也基于 DriverLib。本示例使用 DriverLib。

移植工程时的一个选项是尝试使用等效的 MSPM0 DriverLib API 替换代码的每个部分，但这通常不是最简单的方法。通常，最好先了解被移植的应用程序代码。然后，从最接近的 MSPM0 示例工程开始，对其进行修改以匹配原始代码功能。下面将使用 STM32CubeG0 中的低功耗 UART 示例展示该过程。对于使用许多外设的更复杂的工程，通常会针对每个外设重复该过程。

步骤 4a：了解应用程序

以下说明来自 STM32CubeG0 中名为“LPUART_WakeUpFromStop_Init”的示例工程。

```
@par Example Description

Configuration of GPIO and LPUART peripherals to allow characters received on LPUART_RX pin to wake
up the MCU from low-power mode.This example is based on the LPUART LL API.The peripheral
initialization uses LL initialization function to demonstrate LL init usage.

LPUART Peripheral is configured in asynchronous mode (9600 bauds, 8 data bit, 1 start bit, 1 stop
bit, no parity).
No HW flow control is used.
LPUART Clock is based on HSI.

Example execution:
After startup from reset and system configuration, LED3 is blinking quickly during 3 sec, then MCU
enters "Stop 0" mode (LED3 off).On first character reception by the LPUART from PC Com port (ex:
using HyperTerminal) after "Stop 0" Mode period, MCU wakes up from "Stop 0" Mode.

Received character value is checked :
- On a specific value ('S' or 's'), LED3 is turned On and program ends.
- If different from 'S' or 's', program performs a quick LED3 blinks during 3   sec and enters
again "Stop 0" mode, waiting for next character to wake up.
```

第一步是了解 MCU 的主要设置。这通常是时钟速度和电源策略。在本示例中，未指定通用时钟频率，因为唯一重要的设置是 UART 在低功耗停止 0 模式下工作。它指出低功耗 UART 时钟基于“HIS”或高速内部振荡器，这意味着没有使用外部晶体。UART 以 9600 波特运行，具有 8 个数据位、1 个起始位和停止位，无奇偶校验。没有使用硬件流控制。应用程序侧检查是否收到“S”或“s”并使 LED 闪烁。

步骤 4b：找到最接近的 MSPM0 示例

下一步是了解 STM32G0 和 MSPM0 的 UART 模块之间的任何差异，然后在 MSPM0 SDK 中找到最接近的示例。这可以通过参考节 4 中的 UART 部分轻松完成。该部分重点介绍 UART 模块之间的差异以及指向与 UART 相关的 MSPM0 SDK 代码示例的链接。对于本示例，SDK 中最接近的示例可能是 uart_echo_interrupts_standby，其中“当器件处于待机模式时 UART RX/TX 使用中断回显”。

该 MSPM0 示例与要移植的工程类似，但不完全相同。该示例将进入待机模式，这是一种功耗比停止模式更低的模式。必须检查 UART 通信设置以及正在使用哪些 GPIO。最后，必须添加监视特定字符的应用程序层。

步骤 4c：导入并修改示例

找到类似的示例后，打开 CCS 并导入代码示例，方法是转到 **Project > Import CCS Projects...** 并导航至 MSPM0 SDK 的示例文件夹。导入示例。以下是导入的 *uart_echo_interrupts_standby* 示例。这是一个 SysConfig 工程，因此主 C 文件很简单。它首先调用 SysConfig driverlib 初始化，这是 SysConfig 自动生成的用于配置器件的函数。然后它启用 UART 中断。最后，它进入睡眠状态，等待任何 UART 事务。如果它接收到 UART 事务，它会立即回显数据并唤醒。

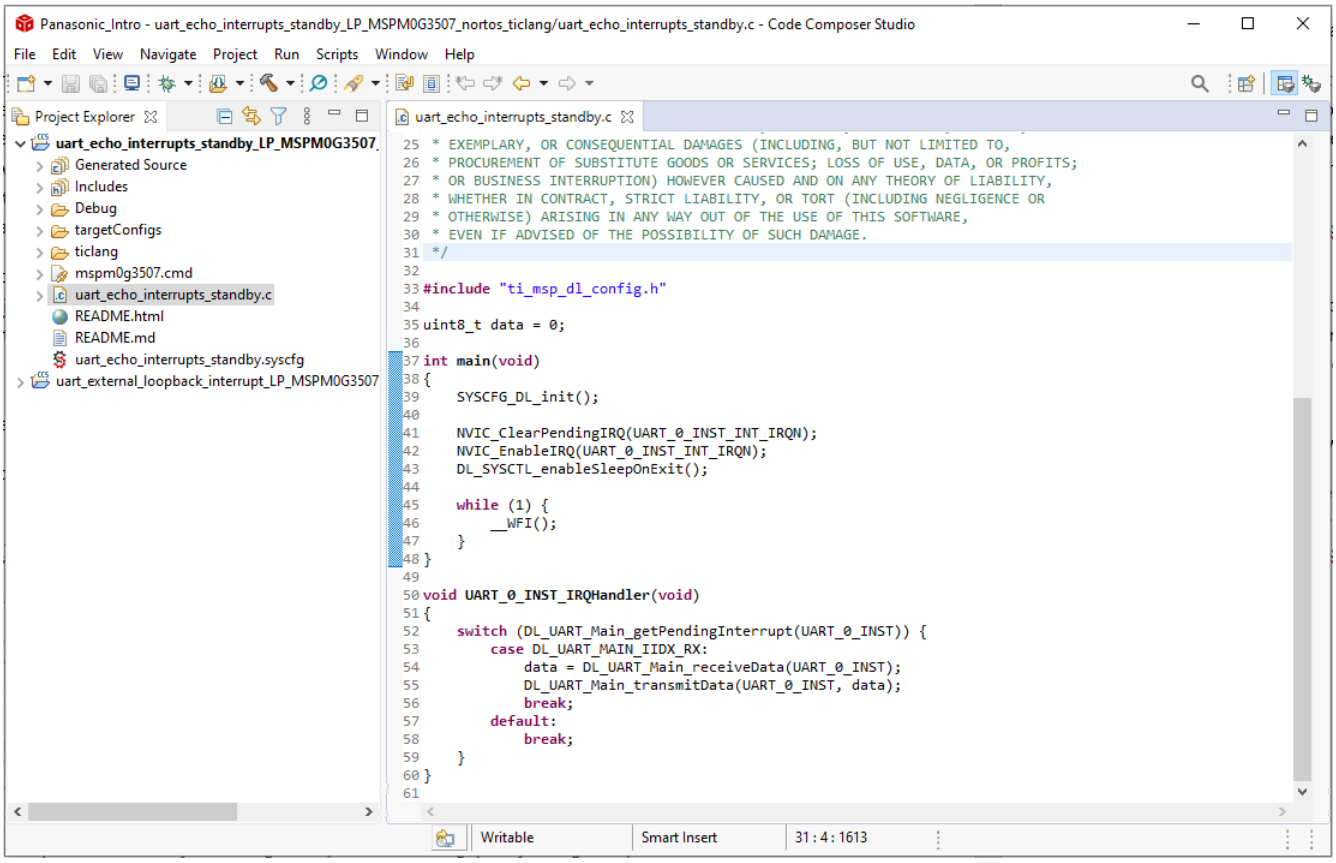

图 2-6. uart_echo_interrupts_standby 示例

要查看 SysConfig 配置，请打开.syscfg 文件，该文件默认在“SYSCTL”选项卡上打开。有关使用 SysConfig 的详细指南，请参阅 MSPM0 SDK 中的 SysConfig 指南。

首先需要注意的是电源策略。该 MSPM0 示例使用待机 0 模式，但目标是使用停止 0 模式。通过点击下拉列表，可以选择正确的低功耗模式。也可以在该选项卡上配置所有时钟和振荡器，但它们现在没问题，无需配置。

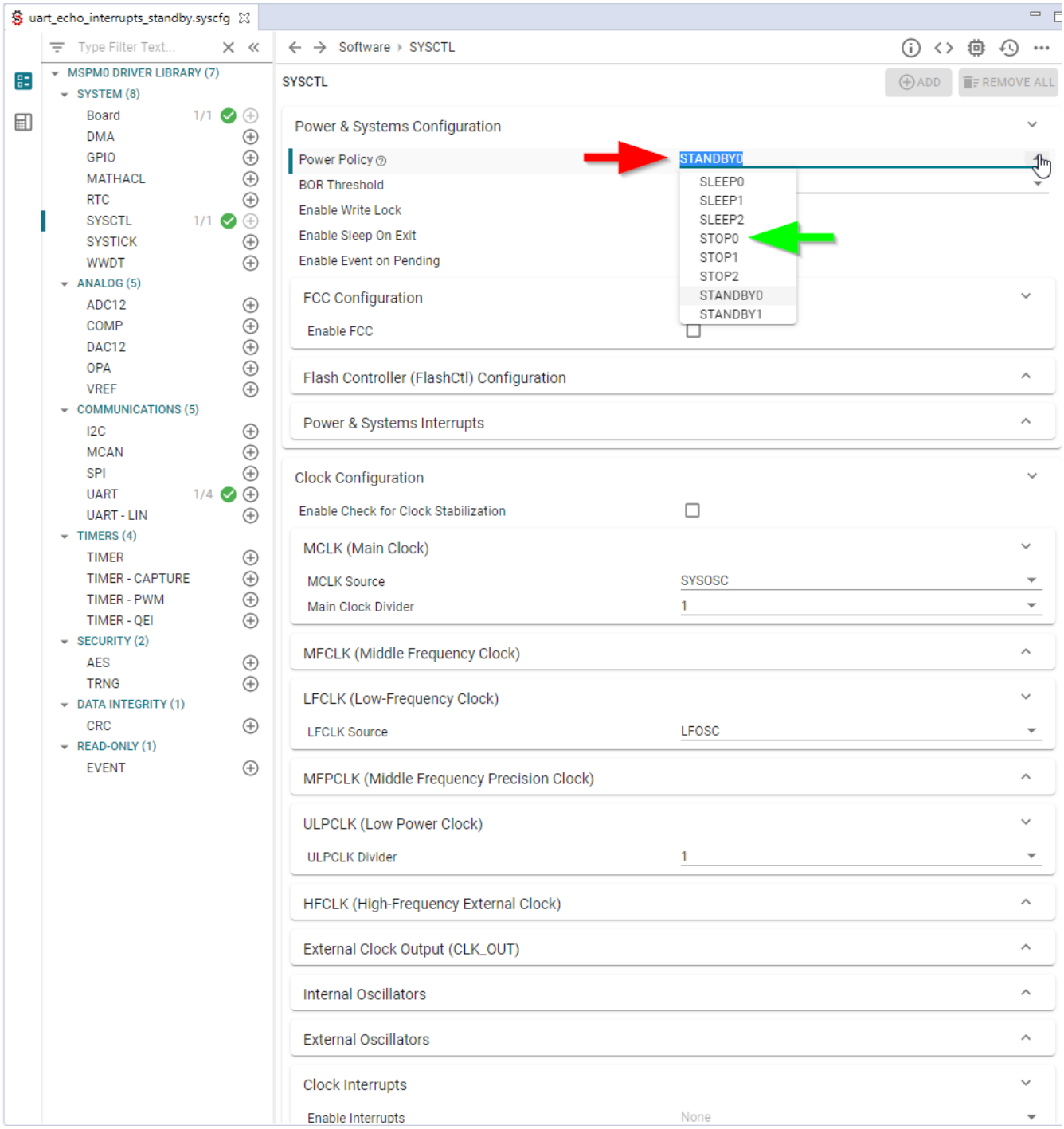

图 2-7. 电源模式配置

接下来，检查“UART”选项卡上的 UART 通信设置（请参阅图 2-8）。在本例中，波特率已设置为 9600，其余的通信设置是正确的。接收中断已经启用并在主程序中使用。此外，通过点击右上角的芯片图标并检查 UART 的突出显示引脚来检查正在使用的 UART 模块和引脚。此处不需要更改任何内容，因为它们已经连接到 MSPM0G3507 LaunchPad 套件的反向通道 UART。

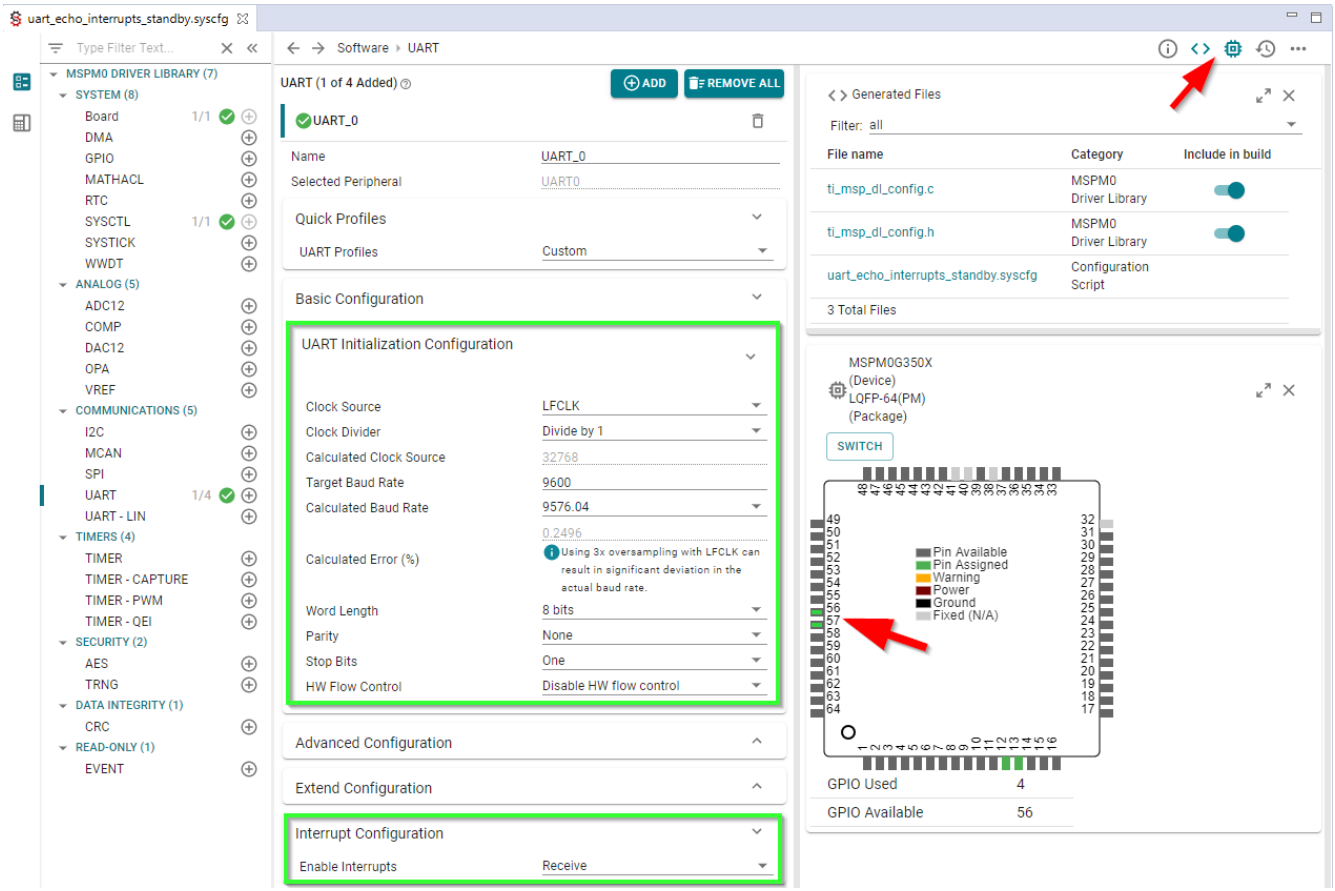

图 2-8. UART 配置

该示例目前没有配置用于驱动 LED 的 GPIO，但可以轻松添加配置（请参阅图 2-9）。可以使用页面顶部的 **+ADD** 按钮添加 GPIO。可以为 GPIO 端口和引脚命名，在本例中分别为“LED”和“RED”。该 GPIO 设置为输出，然后置于端口 A 引脚 0 (PA0) 上。在 LaunchPad 套件上，该 GPIO 连接到一个简单的红色 LED。

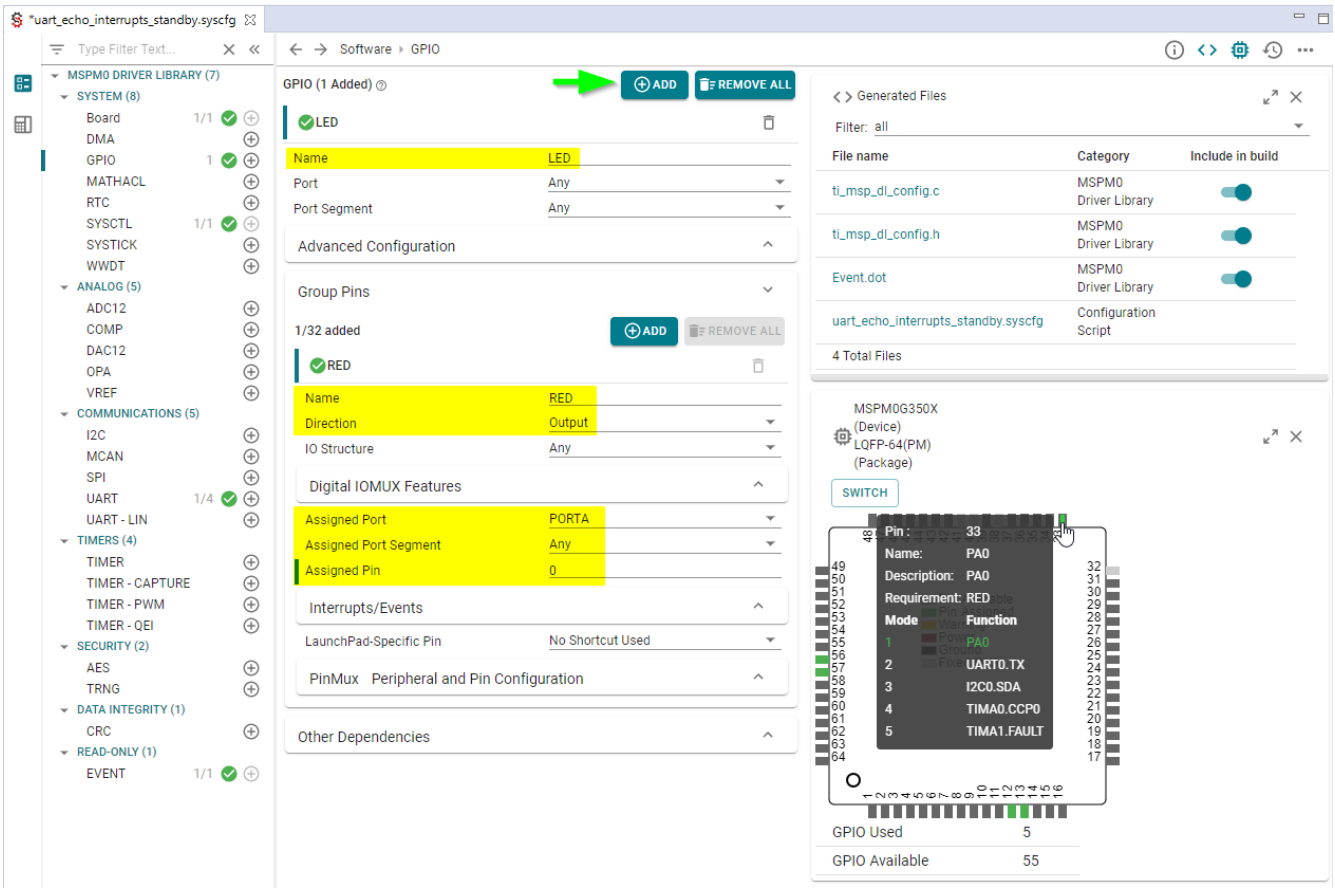

图 2-9. GPIO 配置

保存并重建工程后，SysConfig 会更新该示例的 ti_msp_dl_config.c 和 ti_msp_dl_config.h 文件。此时，已修改示例硬件配置，以匹配正在移植的原始软件的全部功能。唯一剩下的工作是应用程序级软件检查传入的 UART 字节并切换 LED。这可以通过将少量代码移到主 C 文件中来实现。

*uart_echo_interrupts_standby.c uart_echo_interrupts_standby.syscfg

```
*/

#include "ti_msp_dl_config.h"

uint8_t data = 0;

int main(void)
{
    SYSCFG_DL_init();

    DL_GPIO_clearPins(LED_PORT, LED_RED_PIN);

    NVIC_ClearPendingIRQ(UART_0_INST_INT_IRQN);
    NVIC_EnableIRQ(UART_0_INST_INT_IRQN);
//    DL_SYSCTL_enableSleepOnExit();
    DL_SYSCTL_disableSleepOnExit();

    while (1) {
        __WFI();

        if((data == 'S')||( data == 's')){
            DL_GPIO_setPins(LED_PORT, LED_RED_PIN);

        }else{
            DL_GPIO_clearPins(LED_PORT, LED_RED_PIN);
        }
    }
}

void UART_0_INST_IRQHandler(void)
{
    switch (DL_UART_Main_getPendingInterrupt(UART_0_INST)) {
        case DL_UART_MAIN_IIDX_RX:
            data = DL_UART_Main_receiveData(UART_0_INST);
            DL_UART_Main_transmitData(UART_0_INST, data);
            break;
        default:
            break;
    }
}
```

图 2-10. 对应用程序代码的更改

对应用程序代码进行了两处更改。首先，使用 DL_SYSCTL_disablesSleepOnExit()，以使 MSPM0 在每个 UART RX 上短暂唤醒。接下来，添加对 UART RX 数据的简单检查，如果接收到“S”或“s”，则红色 LED 亮起。如果接收到的是其他任何数据，它会熄灭。

步骤 5 ：调试和验证

以下各图是逻辑分析仪的屏幕截图，其中显示了 9600 波特下的 UART 通信以及正确开启和关闭的红色 LED。代码会回显每个 UART 字符，但仅在接收到正确的字符时才开启 LED。

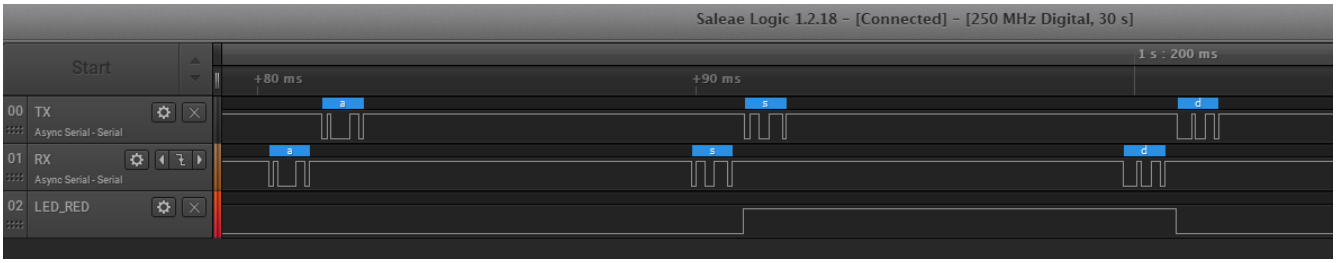

图 2-11.

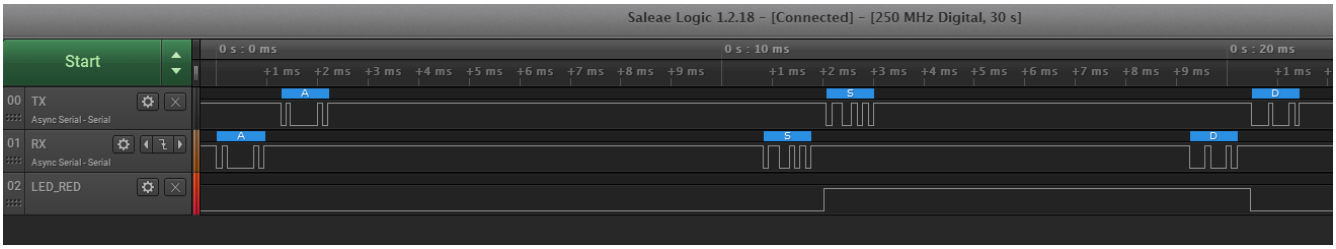

图 2-12.

软件移植成功！如果这只是许多外设的第一个外设，请继续重复该过程并使用 SysConfig 组合每个块。

3 内核架构比较

3.1 CPU

STM32G0 和 MSPM0 系列器件均基于 Arm Cortex® M0+ CPU 内核架构和指令集。下表简要概述了 MSPM0G 和 MSPM0L 系列与 STM32G0 CPU 之间的一般功能比较。中断和异常针对比较了中断和异常以及它们在每款器件 M0 架构中包含的嵌套矢量中断控制器 (NVIC) 外设中的映射方式。

表 3-1. CPU 功能集比较

功能	STM32G0	MSPM0G	MSPM0L
架构	Arm Cortex-M0+	Arm Cortex-M0+	Arm Cortex-M0+
最大 MCLK	64MHz	80MHz	32MHz
CPU 指令缓存	2x64 位行（16 字节）	4x64 位行（32 字节）	2x64 位行（16 字节）
处理器跟踪功能	否	是，集成微跟踪缓冲器	否
存储器保护单元 (MPU)	是	是	否
系统计时器 (SYSTICK)	是	是 - 24 位	是 - 24 位
NVM 预取	是	是	是
硬件乘法	是	是	是
硬件断点/观察点	4/2	4/2	4/2
引导例程存储器	闪存（系统存储器）	ROM	ROM
引导加载程序存储器	闪存（系统存储器）	ROM	ROM
引导加载程序接口支持(1) (2)	UART、I2C、SPI、USB、FDCAN	UART、I2C、用户可扩展	UART、I2C、用户可扩展
DMA	是	是	是

(1) 有关可用性，请参阅特定于器件的数据表。
(2) 其他接口将在以后的器件版本中提供。

3.2 嵌入式存储器比较

3.2.1 闪存功能

MSPM0 和 STM32G0 系列 MCU 具有非易失性闪存，用于存储可执行程序代码和应用数据。

表 3-2. 闪存功能比较

功能	STM32G0	MSPM0
闪存	STM32G0B1xx、G0C1xx（最大 512KB） STM32G071xx、G081xx（最大 128KB） STM32G031xx、G041xx、G051xx、G061xx（最大 64KB）	MSPM0Gxx 范围为 128KB 至 32KB MSPM0Lxx 范围为 64KB 至 8KB
存储器组织	1 组 – 器件存储器最大为 128KB 2 组 – 器件存储器大于 128KB	1 组 – 器件存储器最大为 256KB 2 组 – 器件存储器大于 256KB
闪存等待状态	0 (HCLK ≤ 24MHz) 1 (HCLK ≤ 48MHz) 2 (HCLK ≤ 64MHz)	0（MCLK、CPUCLK ≤ 24MHz） 1（MCLK、CPUCLK ≤ 48MHz） 2（MCLK、CPUCLK ≤ 80MHz）
闪存字大小	64 位加 8 个 ECC 位	相同
编程分辨率	单字大小	单字、32 位、16 位或 8 位（字节）
多字编程	32 个字（256 字节）	2、4 或 8 个字（最多 64 个字节）
擦除	页面大小 = 2KB 组擦除（单组） 批量擦除（所有组）	扇区大小 = 1KB 组擦除（最大 256KB）
写保护	是（每组 2 个写保护区域）	是，静态和动态
读保护	是	是

表 3-2. 闪存功能比较 (continued)

功能	STM32G0	MSPM0
闪存读取操作	64 位闪存字大小加 8 个 ECC 位	相同 – 如果存在可选的 ECC
闪存写入操作	64 位闪存字大小加 8 个 ECC 位	相同 – 如果存在可选的 ECC
纠错码 (ECC)	对于 64 位为 8 位	相同
安全存储器区域	是，主存储器	否
信息存储器	是	是 (NONMAIN)
OTP 数据区域	1KB	否
预取	是	是
CPU 指令缓存	两个 64 位缓存行（16 字节） 4 条 32 位指令或 8 条 16 位指令	四个 64 位缓存行（32 字节）8 条 32 位指令 或 16 条 16 位指令

除了上表中列出的闪存功能外，MSPM0 闪存还具有以下功能：

- 在整个电源电压范围内支持电路内编程和擦除操作
- 内部编程电压生成
- 支持 EEPROM 仿真，在低 32KB 闪存上支持多达 100 000 个编程/擦除周期，在剩余闪存上支持多达 10 000 个编程/擦除周期（32KB 的器件在整个闪存上支持 100 000 个周期）

3.2.2 闪存组织

闪存用于存储应用程序代码和数据、器件引导配置以及 TI 在出厂时预编程的参数。闪存被组织为一个或多个组，每个组中的存储器进一步映射到一个或多个逻辑存储区域，并分配有系统地址空间以供应用程序使用。

存储器组

大多数 MSPM0 器件会实现单个闪存组 (BANK0)。在具有单个闪存组的器件上，一个正在进行的编程/擦除操作会暂停对闪存的所有读取请求，直到操作完成并且闪存控制器已经释放了对组的控制。在具有多个闪存组的器件上，一个组上的编程/擦除操作也会暂停向正在执行编程/擦除操作的组发出的读取请求，但不会暂停向另一个组发出的读取请求。因此，存在多个组可实现以下应用案例：

- 双映像固件更新（应用程序可以从一个闪存组中执行代码，同时将第二个映像编程到第二个对称的闪存组，而不会暂停应用程序的执行）
- EEPROM 仿真（应用程序可以从一个闪存组中执行代码，第二个闪存组用于写入数据，而不会暂停应用程序的执行）

闪存区域

根据每个组中存储器所支持的功能，每个组中的存储器映射到一个或多个逻辑区域。有四个区域：

- FACTORY – 器件 ID 和其他参数
- NONMAIN – 器件引导配置（BCR 和 BSL）
- MAIN – 应用程序代码和数据
- DATA – 数据或 EEPROM 仿真

具有一个组的器件在 BANK0（唯一存在的组）上实现 FACTORY、NONMAIN 和 MAIN 区域，而 DATA 区域不可用。具有多个组的器件也在 BANK0 上实现 FACTORY、NONMAIN 和 MAIN 区域，但包括可实现 MAIN 或 DATA 区域的其他组（BANK1 至 BANK4）。

NONMAIN 存储器

NONMAIN 是闪存的专用区域，用于存储 BCR 和 BSL 用于引导器件的配置数据。该区域不用于任何其他目的。BCR 和 BSL 都具有配置策略，这些策略可以保留为默认值（在开发和评估期间是典型值），也可以通过更改编程到 NONMAIN 闪存区域中的值来针对特定用途进行修改（在生产编程期间是典型值）。

3.2.3 嵌入式 SRAM

MSPM0 和 STM32G0 系列 MCU 具有用于存储应用程序数据的 SRAM。

表 3-3. SRAM 功能比较

功能	STM32G0	MSPM0
SRAM 存储器	STM32G0B1xx、G0C1xx：144KB（启用 SRAM 奇偶校验时为 128KB） STM32G071xx、G081xx：36KB（启用 SRAM 奇偶校验时为 32KB） STM32G051xx、G061xx：18KB（启用 SRAM 奇偶校验时为 16KB） STM32G031xx、G041xx：8KB（启用 SRAM 奇偶校验时为 8KB） 零等待状态	MSPM0Gxx：32KB 至 16KB MSPM0Lxx：4KB 至 2KB 零等待状态 部分器件包括 SRAM 奇偶校验和 ECC。有关详细信息，请参阅器件数据表
最大 CPU 时钟频率下的零等待状态	是	是
访问分辨率	字节、半字（16 位）或全字（32 位）	字节、半字（16 位）或全字（32 位）
奇偶效验检查	是	是

MSPM0 MCU 包含低功耗高性能 SRAM，可在器件支持的 CPU 频率范围内实现零等待状态访问。除代码之外，SRAM 存储器还可用于存储易失性信息，例如调用栈、堆和全局数据。SRAM 内容在运行、睡眠、停止和待机工作模式下完全保留，但在关断模式下会丢失。提供了一种写保护机制，允许应用程序以 1KB 的分辨率对低 32KB SRAM 进行动态写保护。在 SRAM 小于 32KB 的器件上，器件为整个 SRAM 提供了写保护。在将可执行代码放入 SRAM 时写保护很有用，因为它可以针对 CPU 或 DMA 无意覆盖代码提供一定程度的保护。将代码放置在 SRAM 中可以通过实现零等待状态操作和降低功耗来提高关键循环的性能。

3.3 上电和复位总结和比较

与 STM32G0 器件类似，MSPM0 器件具有最低工作电压，并具有相应的模块，可通过将器件或器件的某些部分保持在复位状态来确保器件正常启动。表 3-4 比较了这两个系列的实现方式以及哪些模块控制整个系列的上电过程和复位。

表 3-4. 上电比较

STM32G0 器件		MSPM0 器件	
控制上电和复位的模块	PWR（电源）和 RCC（复位和时钟控制）模块	控制加电和复位的模块	PMCU（电源管理和时钟单元）
基于电压电平的复位			
POR（上电复位）	完整的器件复位。上电时发生第一级电压释放。断电时具有最低电压电平。	POR（上电复位）	完整的器件复位。上电时发生第一级电压释放。断电时具有最低电压电平。
电平可配置的 BOR（欠压复位）	有时可编程。设置在上电时释放复位状态或在断电时复位器件的电压电平。	可配置 BOR（欠压复位）	可配置为复位或中断，具有不同的电压阈值，同时结合了 STM32G0 BOR 和 PVD 功能。
PVD（可编程电压检测器）	可提供中断的可配置电压监视器。		

STM32G0 定义了不同的复位域，而 MSPM0 器件具有不同级别的复位状态。对于 MSPM0 器件，复位级别具有设定的顺序，当一个级别被触发时，所有后续级别都会被复位，直到器件被释放至运行模式。表 3-5 简要说明和比较了 STM32G0 复位域和 MSPM0 复位状态。图 3-1 显示了所有 MSPM0 复位状态之间的关系。

表 3-5. 复位域比较

STM32G0 复位域		MSPM0 复位状态(1)	
电源复位域	典型的触发器为 POR、BOR 以及退出待机或关断模式。所有寄存器都复位，VCORE 域之外的寄存器除外。	POR	典型的触发器：POR 电压电平、SW 触发、NRST 保持低电平的时长大于 1s。复位会关断存储器，重新启用 NRST 和 SWD，触发 BOR
		BOR	典型的触发器：POR 或 BOR 电压电平、退出关断模式。复位 PMU、VCORE 和相关逻辑。触发 BOOTRST。
没有确切的等效功能。复位后在 SYSCLK 的第四个时钟周期读取引导配置。		引导复位 (BOOTRST)	典型的触发器：BOR 或软件触发、致命时钟故障、NRST 保持低电平的时长小于 1s。执行引导配置例程。复位大多数内核逻辑和寄存器，包括 RTC、时钟和 IO 配置。(2)SRAM 下电上电，并且内容丢失。触发 SYSRST。
系统复位域	系统复位将所有寄存器设置为其复位值，但时钟控制和状态寄存器 (RCC_CSR) 中的复位标志以及 RTC 域中的寄存器除外。	系统复位 (SYSRST)	典型的触发器：BOOTRST、BSL 进入或退出、看门狗计时器、软件触发、调试子系统。复位 CPU 状态和除 RTC、LFCLK、LFXT 和 SYSOSC 频率校正环路之外的所有外设。器件在退出时进入运行模式。
无等效功能		仅 CPU 复位 (CPURST)	仅限软件和调试子系统触发器。仅复位 CPU 逻辑。外设状态不受影响。
RTC 域	由软件或 VDD 或 VBAT 上电触发，前提是两个电源之前都已断电。仅复位 LSE 振荡器、RTC、备份寄存器和 RCC RTC 域控制寄存器。	RTC 和相关时钟通过 BOOTRST、BOR 或 POR 复位。(2)	

(1) 并未说明所有复位触发器。有关所有可用的复位触发器，请参阅器件 TRM 的 PMCU 章节。
(2) 如果 BOOTRST 是 NRST 或软件触发导致的，则 RTC、LFCLK 和 LFXT/LFLCK_IN 配置和 IOMUX 设置不会复位，从而允许 RTC 在外部复位时保持运行。

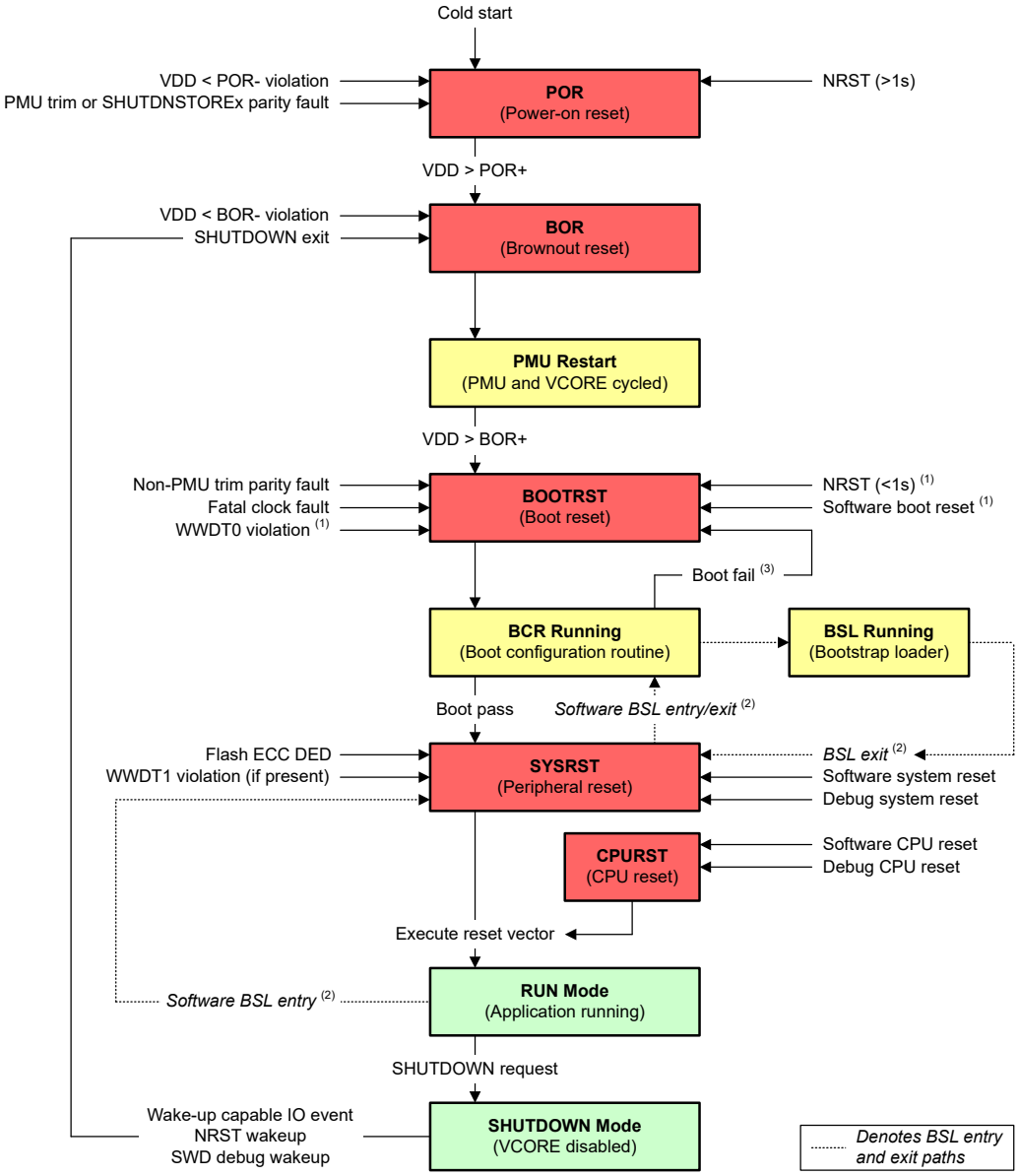

图 3-1. MSPM0 复位级别

3.4 时钟总结和比较

STM32G 和 MSPM0 包含为主时钟提供源的内部振荡器。这些时钟可被分频，从而为其他时钟提供源并被分配到多个外设上。

表 3-6. 振荡器比较

STM32G0 振荡器	MSPM0 振荡器
HSI16RC 16MHz	SYSOSC(1)
HSI48RC 48MHz	SYSOSC
LSI RC 32kHz	LFOSC
HSE OSC 4-48MHz	HFXT
LSE OSC 32kHz	LFXT
I2S_CLKIN	HFCLK_IN (数字时钟)

(1) SYSOSC 可编程为 32MHz、24MHz、16MHz 或 4MHz。

表 3-7. 时钟比较

STM32G 时钟	MSPM0 时钟
HSISYS	不适用
PLLPCLK	SYSPLLCLK1
PLLQCLK	SYSPLLCLK1
PLLRCLK	SYSPLLCLK0
不适用	SYSPLLCLK2x(1)
SYSCLK	BUSCLK(2)
HCLK	MCLK
HCLK8	CPUCLK
来获取 VOUT PCLK。	BUSCLK
TIMPCLK	BUSCLK
LPTIMx_IN	LFCLK_IN

(1) SYSPLLCLK2x 的速度是 PLL 模块输出速度的两倍，可进行分频。
(2) BUSCLK 取决于电源域。对于电源域 0，BUSCLK 为 ULPCLK。对于电源域 1，BUSCLK 为 MCLK。

表 3-8. 外设时钟源

外设	STM32G 时钟源	MSPM0 时钟源
RTC	LSI、LSE、HSE/32	LFCLK（LFOSC、LFXT）
UART	PCLK、LSE、HSI16、SYSCLK	BUSCLK、MFCLK、LFCLK
SPI	需要查找	BUSCLK、MFCLK、LFCLK
I2C	PCLK、HSI16、SYSCLK	BUSCLK、MFCLK
ADC	HSI16、SYSCLK、PLLPCLK	ULPCLK、HFCLK、SYSOSC
CAN	PCLK、HSE、PLLQCLK	PLLCLK1、HFCLK
定时器	PCLK、TIMPCLK、PLLQCLK	BUSCLK、MFCLK、LFCLK
LPTIM 1/2 (TIMG0/1)	PCLK、LSI、LSE、HSI16、LPTIMX_IN	LFCLK、ULPCLK、LFCLK_IN
RNG	HSI48、PLLQCLK、HSI16/8、SYSCLK	MCLK

每个器件系列的 TRM 都有一个时钟树，可帮助使时钟系统可视化。Sysconfig 可以帮助您选择时钟分频以及为外设提供源。

3.5 MSPM0 工作模式总结和比较

MSPM0L MCU 提供五种主要工作模式（电源模式），可根据应用要求优化器件功耗。这些模式按照功耗从高到低排列如下：运行、睡眠、停止、待机和关断。CPU 会在运行模式中执行代码。外设中断事件可将器件从睡眠、停止或待机模式唤醒至运行模式。关断模式会完全禁用内部内核稳压器，以更大限度地降低功耗，并且只能通过 NRST、SWD 或某些 IO 上的逻辑电平匹配来实现唤醒。运行、睡眠、停止和待机模式还包括多个可配置的策略选项（例如，RUN.x），用于平衡性能与功耗。

为了进一步平衡性能和功耗，MSPM0L 器件实现了两个电源域：PD1（用于 CPU、存储器和高性能外设）和 PD0（用于低速、低功耗外设）。在运行和睡眠模式下，PD1 始终通电，但在所有其他模式下会禁用。PD0 在运行、睡眠、停止和待机模式下始终通电。PD1 和 PD0 在关断模式下都会禁用。

工作模式比较

STM32G0 器件具有类似的工作模式。下表对 STM32G0 器件和 MSPM0 器件进行了简要比较。

表 3-9. STM32G0 器件和 MSPM0 器件的工作模式比较

STM32G0			MSPM0		
模式		说明	模式	说明	
运行		提供完整的时钟和外设	运行	0	提供完整的时钟和外设
LP 运行		CPU 限制为 2MHz		1	SYSOSC 处于设定的频率；CPUCLK 和 MCLK 限制为 32kHz
				2	SYSOSC 被禁用；CPUCLK 和 MCLK 限制为 32kHz
睡眠		不对 CPU 计时	睡眠	0	不对 CPU 计时
LP 睡眠		与 LP 运行相同；但不对 CPU 计时		1	与运行 1 相同，但不对 CPU 计时
				2	与运行 2 相同，但不对 CPU 计时
停止	0	VCORE 域时钟被禁用	停止	0	睡眠 0 + PD1 被禁用
				1	睡眠 1 + SYSOSC 档位切换至 4MHz
	1	停止 0 + 主电源稳压器关闭		2	睡眠 2 + ULPCLK 限制为 32kHz
待机		具有 BOR 功能的最低功耗；RTC 可用；寄存器设置丢失。	待机	0	具有 BOR 功能的最低功耗；所有 PD0 外设都可以接收 32kHz 的 ULPCLK 和 LFCLK；RTC 可用于 RTCCLK
				1	只有 TIMG0 和 TIMG1 可以接收 32kHz 的 ULPCLK 或 LFCLK；RTC 可用于 RTCCLK
关断		没有时钟或 BOR。内核稳压关闭。RTC 域仍可处于活动状态。退出触发复位。	关断	没有时钟、BOR 或 RTC。内核稳压关闭。PD1 和 PD0 被禁用。退出触发复位级别 BOR。	

低功耗模式下的 MSPM0 功能

如表 3-9 所示，在低功耗工作模式下，MSPM0 外设或外设模式的可用性或运行速度可能会受到限制。有关具体详细信息，请参阅特定于 MSPM0 器件的数据表中的“不同工作模式下支持的功能”表，例如：

MSPM0G350x 混合信号微控制器数据表

MSPM0L134x、MSPM0L130x 混合信号微控制器数据表

MSPM0 器件的另一项功能是某些外设能够执行异步快速时钟请求。这使 MSPM0 器件能够处于低功耗模式，在该模式下外设未处于活动状态，但仍然能够触发或激活外设。当异步快速时钟请求发生时，MSPM0 器件能够快速将内部振荡器提升至更高的速度和/或暂时使其进入更高的工作模式以处理即将发生的操作。这允许通过计时器、比较器、GPIO 和 RTC 快速唤醒 CPU；接收 SPI、UART 和 I2C；或触发 DMA 传输和 ADC 转换，同时在最低功耗模式下睡眠。有关异步时钟请求实现以及外设支持和用途的具体详细信息，请参阅 MSPM0 TRM 中相应的章节。

MSPM0 G 系列 80MHz 微控制器技术参考手册

MSPM0 L 系列 32MHz 微控制器技术参考手册

进入低功耗模式

与 STM32G0 器件类似，MSPM0 器件在执行等待事件 `__WFE();` 或等待中断 `__WFI();` 指令时会进入低功耗模式。低功耗模式由当前电源策略设置决定。器件电源策略由驱动程序库函数设置。以下函数调用将该电源策略设置为待机 0。

```
DL_SYSCTL_setPowerPolicySTANDBY0();
```

`STANDBY0` 可替换为选择的工作模式。有关管理电源策略的 driverlib API 的完整列表，请参阅 MSPM0 SDK DriverLib API 指南的这一部分。另请参阅以下代码示例，这些示例演示了如何进入不同的工作模式。每个 MSPM0 器件都有类似的示例。

低功耗模式代码示例

导航至 SDK 安装目录，在 examples > nortos > LP name > driverlib 中找到低功耗模式代码示例

3.6 中断和事件比较

中断和异常

MSPM0 和 STM32G0 都根据器件的可用外设来注册和映射中断和异常矢量。表 3-10 中包含每个器件系列的中断矢量的总结和比较。中断或异常的优先级值越低，优先级就越高。对于其中的一些矢量，优先级是用户可选的，而对于其他矢量，优先级是固定的。

在 MSPM0 和 STM32G0 中，NMI、复位和硬故障处理程序等异常被赋予负优先级值，以指示它们始终比外设中断具有更高的优先级。对于具有可选中断优先级的外设，两个器件系列均提供多达 4 个可编程优先级。

表 3-10. 中断比较

NVIC 编号	STM32G0		MSPM0x	
	中断/异常	优先级	中断/异常	优先级
-	复位	固定：-3	复位	固定：-3
-	NMI 处理程序	固定：-2	NMI 处理程序	固定：-2
-	硬故障处理程序	固定：-1	硬故障处理程序	固定：-1
-	SVCall 处理程序	可选	SVCall 处理程序	可选
-	PendSV	可选	PendSV	可选
-	SysTick	可选	SysTick	可选
0	窗口看门狗中断	可选	INT_GROUP0：WWDT0、DEBUGSS、FLASHCTL、WUC FSUBx 和 SYSCTL	可选
1	电源电压检测器中断	可选	INT_GROUP1：GPIO0 和 COMP0	可选
2	RTC 和时间戳	可选	计时器 G1 (TIMG1)	可选
3	闪存全局中断	可选	UART3(1)	可选
4	RCC 全局中断	可选	ADC0	可选
5	EXTI0 和 EXTI1 中断	可选	ADC1(1)	可选
6	EXTI2 和 EXTI3 中断	可选	CANFD0(1)	可选
7	EXTI4-EXTI15 中断	可选	DAC0(1)	可选
8	UCPD1/UCPD2/USB	可选	保留	可选
9	DMA1 通道 1	可选	SPI0	可选
10	DMA1 通道 2 和 3	可选	SPI1(1)	可选
11	DMA1 通道 4-6 和 DMA2 通道 1-5	可选	保留	可选
12	ADC 和比较器	可选	保留	可选
13	计时器 1 (TIM1)、中断、更新、触发和换向	可选	UART1	可选
14	TIM1 捕捉比较	可选	UART2(1)	可选
15	TIM2 全局中断	可选	UART0	可选
16	TIM3 和 TIM4 全局中断	可选	TIMG0	可选
17	TIM6、LPTIM1 和 DAC 中断	可选	TIMG10(1)	可选
18	TIM6 和 LPTIM2 全局中断	可选	TIMA0(1)	可选
19	TIM14 全局中断	可选	TIMA1	可选
20	TIM15 全局中断	可选	TIMA2(2)	可选
21	TIM16 和 FDCAN0 全局中断	可选	TIMH0(1)	可选
22	TIM17 和 FDCAN1 全局中断	可选	保留	可选
23	12C1 全局中断	可选	保留	可选

表 3-10. 中断比较 (continued)

NVIC 编号	STM32G0		MSPM0x	
	中断/异常	优先级	中断/异常	优先级
24	I2C2 和 I2C3 全局中断	可选	I2C0	可选
25	SPI1 全局中断	可选	I2C1	可选
26	SPI2 和 SPI3 全局中断	可选	保留	可选
27	USART1 全局中断	可选	保留	可选
28	USART2 和 LPUART2 全局中断	可选	AES(1)	可选
29	USART 3-6 和 LPUART1 全局中断	可选	保留	可选
30	CEC 全局中断	可选	RTC(1)	可选
31	AES 和 RNG 全局中断	可选	DMA	可选

(1) 仅适用于 MSPM0G 系列器件。
(2) 在 MSPM0L 系列器件上为 TIMG4

事件处理程序和 EXTI（扩展中断和事件控制器）

MSPM0 器件包含一个专用事件管理器外设，它扩展了 NVIC 的概念，允许将来自外设的数字事件作为中断传输到 CPU，作为触发器传输到 DMA，或传输到另一个外设以触发硬件操作。事件管理器还可以与电源管理和时钟单元 (PMCU) 进行握手，以确保存在必要的时钟和电源域，从而执行触发事件操作。

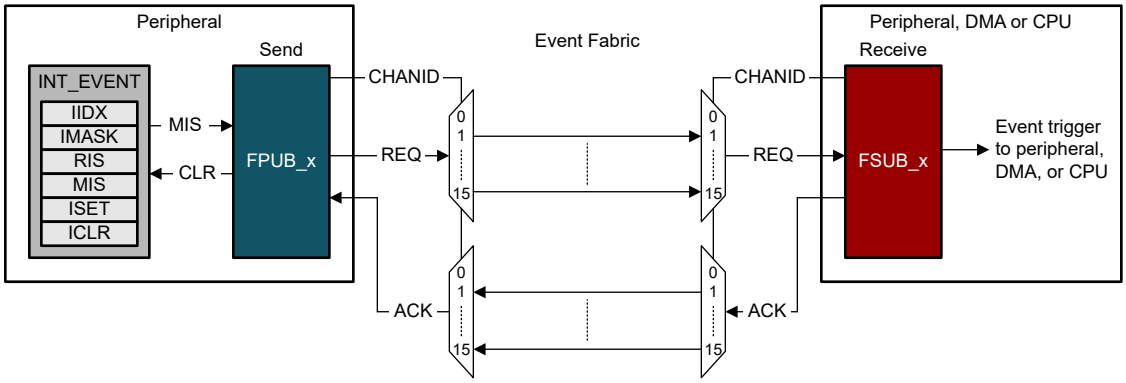

图 3-2. 通用事件路由

在 MSPM0 事件管理器中，生成事件的外设称为发布者，而基于发布者执行操作的外设、DMA 或 CPU 称为订阅者。可用发布者和订阅者的潜在组合非常灵活，可以在移植软件时使用，以替代以前由中断矢量和 CPU 处理的功能，从而完全绕过 CPU。例如，I^2C 转 UART 桥接器可能先前在接收到 I^2C 停止时触发了 UART 传输，使用 ISR 设置标志，或直接加载 UART TX 缓冲区。利用 MSPM0 事件处理程序，I^2C 事务完成事件可以触发 DMA 直接加载 UART TX 缓冲区，因此无需 CPU 采取任何操作。

有关如何使用 MSPM0 中的事件处理程序的更多详细信息，请参阅 MSPM0G 技术参考手册或 MSPM0L 技术参考手册的“事件”部分。

不要与 MSPM0 事件处理程序混淆，STM32G0 系列器件实现了扩展中断和事件控制器 (EXTI)，它允许通过来自 IO 或外设的可配置事件将系统从停止模式唤醒。使用 IO 唤醒功能（请参阅 MSPM0 技术参考手册的 IOMUX 部分）和 GPIO FastWake（请参阅 MSPM0 技术参考手册的 GPIO 部分），可以在 MSPM0 中最好地复制 STM32G0 EXTI 的唤醒功能。如果唤醒是针对单个操作，则事件处理程序外设能够为外设操作发生请求必要的 PMCU 资源，并在之后返回适用的低功耗模式。

3.7 调试和编程比较

Arm SWD 2 线 JTAG 端口是 MSPM0 和 STM32G0 器件的主要调试和编程接口。该接口通常在应用程序开发期间和生产编程期间使用。表 3-11 比较了两个器件系列的功能。有关 MSPM0 调试接口安全功能的其他信息，请参阅 *MSPM0 MCU 中的网络安全机制* 应用手册。

表 3-11. Arm SWD JTAG 功能比较

	STM32G0	MSPM0
调试端口	Arm SWD 端口（2 线）	Arm SWD 端口（2 线）
断点单元 (BPU)	4 个硬件断点	4 个硬件断点
数据观察单元 (DWT)	2 个观察点	2 个观察点
微跟踪缓冲器 (MTB)	否	4 个跟踪数据包支持 MTB(1)
低功耗调试支持	是	是
EnergyTrace 支持	否	EnergyTrace+ 支持（CPU 状态与功耗性能评测）
调试期间的外设运行支持	是	是
调试接口锁定	可以暂时阻止调试读取访问	可以永久禁用调试功能，或者可以使用密码锁定

(1) 仅限 MSPM0Gxxxx 器件

引导加载程序 (BSL) 编程选项

引导加载程序 (BSL) 编程接口是 Arm SWD 的替代编程接口。该接口仅提供编程功能，通常通过标准嵌入式通信接口使用。这允许通过与系统中用于连接其他嵌入式器件的现有接头或外部端口进行固件更新。尽管编程更新是该接口的主要用途，但它也可用于初始生产编程。表 3-12 比较了 MSPM0 和 STM32G0 器件系列的不同选项和功能。

表 3-12. BSL 功能比较

BSL 功能	STM32G0	MSPM0
BSL 在空白器件上启动	是	是
自动检测编程接口	是	是
安全性	存储器安全和访问限制选项	安全启动选项；CRC 保护
可定制	否	是，可配置调用引脚和插件功能
调用方法	模式(1) 在发生 RESET、SW 进入时最多涉及 2 个引脚和器件寄存器设置	在发生 BOOTRST、SW 进入时 1 个引脚处于高电平
支持的接口		
UART	是	是
I2C	是	是
SPI	是(2)	需要自定义插件
CAN	是(2)	已计划插件(2)
USB	是(2)	目前没有具有 USB 功能的 MSPM0 器件。

(1) 模式选项可用性取决于器件。
(2) 仅适用于部分器件

4 数字外设比较

4.1 通用 I/O（GPIO、IOMUX）

MSPM0 GPIO 功能几乎涵盖了 STM32G0 GPIO 提供的所有功能。STM32G0 使用术语 GPIO 来指代负责管理器件引脚的所有功能。不过，MSPM0 使用稍微不同的命名法，即：

- MSPM0 GPIO 指能够读取和写入 IO、生成中断等的硬件。
- MSPM0 IOMUX 指负责将不同内部数字外设连接到引脚的硬件。IOMUX 为许多不同的数字外设提供服务，包括但不限于 GPIO。

MSPM0 GPIO 和 IOMUX 共同涵盖了与 STM32G0 GPIO 相同的功能。此外，MSPM0 提供了 STM32G0 器件所不具备的功能，例如 DMA 连接、可控输入滤波和事件功能。

表 4-1. GPIO 功能比较

功能	STM32G0	MSPM0G 和 MSPM0L
输出模式	推挽 具有上拉或下拉的开漏	等效
GPIO 速度选择	针对每个 I/O 进行速度选择	类似 MSPM0 在所有 IO 引脚上提供标准 IO (SDIO)。SDIO 与 STM GPIO speed=01 相当或更好。 MSPM0 高速 IO (HSIO) 在选定引脚上可用。HSIO 相当于 STM GPIO speed=10。
高驱动 GPIO	大约 20mA	等效，称为高驱动 IO (HDIO)
输入模式	浮动 上拉或下拉 模拟	等效
原子位设置和复位	是	等效
GPIO 锁定	寄存器锁定机制	*无 MSPM0 等效功能*
替代功能	选择寄存器	等效 MSPM0 使用 IOMUX
快速切换	每两个时钟周期可以更改一次	等效，MSPM0 可以每个时钟周期切换一次引脚
唤醒	GPIO 引脚状态更改	等效
DMA 控制的 GPIO	否	*仅在 MSPM0 上可用*
用户控制的输入滤波，可抑制小于 1、3 或 8 个 ULPCLK 周期的干扰	否	*仅在 MSPM0 上可用*
用户可控制的输入迟滞	否	*仅在 MSPM0 上可用*

GPIO 代码示例

有关 GPIO 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

4.2 通用异步接收器/发送器 (UART)

STM32G0 和 MSPM0 都提供用于执行异步（无时钟）通信的外设。这些 UART 外设有两种型号，一种具有标准功能，另一种具有高级功能。表 4-2 显示了命名差异。

表 4-2. STM32G0 和 MSPM0 之间的 UART 命名差异

	STM32G0 命名	MSPM0 命名
标准功能	基础型	主要热点
高级特性	全双工	扩展

表 4-3. UART 高级功能集比较

功能	STM32G0 USART 全套功能	MSPM0L 和 MSPM0G UART 扩展功能集
硬件流控制	是	是
使用 DMA 的连续通信	是	是
多处理器	是	是
同步模式	是	否
智能卡模式 (ISO7816)	是	是
单线半双工通信	是	是(1)
IrDA 硬件支持	是	是
LIN 硬件支持	是	是
DALI 硬件支持	否	是
曼彻斯特编码硬件支持	否	是
从低功耗模式唤醒	是	是
自动波特率检测	是	否
驱动器使能	是	是
数据长度	7、8、9	5、6、7、8
Tx/Rx FIFO 深度	8	4

(1) 需要在发送和接收之间重新配置外设

表 4-4. UART 标准功能集比较

功能	STM32G0 USART 基本功能集	MSPM0 UART 主要功能集
硬件流控制	是	是
使用 DMA 的连续通信	是	是
多处理器	是	是
同步模式	是	否
单线半双工通信	是	是(1)
从低功耗模式唤醒	否	是
驱动器使能	是	是
数据长度	7、8、9	5、6、7、8
Tx/Rx FIFO 深度	无	4

(1) 需要在发送和接收之间重新配置外设

UART 代码示例

有关 UART 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

4.3 串行外设接口 (SPI)

MSPM0 和 STM32G0 都支持串行外设接口 (SPI)。总的来说，MSPM0 和 STM32G0 SPI 支持是相当的，表 4-5 列出了两者之间的差异。

表 4-5. SPI 功能比较

功能	STM32G0x	MSPM0L 和 MSPM0G
控制器或外设操作	是	是
数据位宽（控制器模式）	4 至 16 位	4 至 16 位
数据位宽（外设模式）	4 至 16 位	7 至 16 位
最大速度	32MHz	MSPM0L：16MHz
		MSPM0G：32MHz
全双工传输	是	是
半双工传输（双向数据线）	是	否
单工传输（单向数据线）	是	是
多控制器功能	是	否
硬件芯片选择管理	是（1 个外设）	是（4 个外设）
可编程时钟极性和相位	是	是
具有 MSB 优先或 LSB 优先移位的可编程数据顺序	是	是
SPI 格式支持	Motorola、TI	Motorola、TI、MICROWIRE
硬件 CRC	是	否，MSPM0 提供 SPI 奇偶校验模式
TX FIFO 深度	取决于数据大小	4
RX FIFO 深度	取决于数据大小	4

SPI 代码示例

有关 SPI 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

4.4 I²C

MSPM0 和 STM32G0 都支持 I²C。总的来说，MSPM0 和 STM32G0 I²C 支持是相当的，下表列出了显著差异。

表 4-6. I²C 功能比较

功能	STM32G0	MSPM0L 和 MSPM0G
控制器和目标模式	是	是
多控制器功能	是	是
标准模式（最高 100kHz）	是	是
快速模式（最高 400kHz）	是	是
超快速模式（最高 1MHz）	是	是
寻址模式	7、10 位	7 位
外设地址	2 个地址和 1 个可配置掩码	2 个地址
常规调用	是	是
可编程设置时间和保持时间	是	否
事件管理	是	是
时钟延展	是	是
软件复位	是	是
FIFO/缓冲器	1 字节	TX：8 字节
		RX：8 字节
DMA	是	是
可编程模拟和数字噪声滤波器	是	是

I²C 代码示例

有关 I2C 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

4.5 计时器（TIMGx、TIMAx）

STM32G0 和 MSPM0 都提供各种计时器。MSPM0 提供具有不同功能的计时器，支持从低功耗监控到高级电机控制的各种用例。

表 4-7. 计时器命名

STM32G0		MSPM0	
计时器名称	缩写名称	计时器名称	缩写名称
高级控制	TIM1	高级控制	TIMA0
通用	TIM2-4、TIM14/-17	通用	TIMG0-11
		高分辨率	TIMG12
基础型	TIM6/7		
低功耗	LPTIM		

表 4-8. 计时器功能比较

功能	STM32G0 计时器	MSPM0G 计时器	MSPM0L 计时器
分辨率	16 位、32 位	16 位、32 位	16 位
PWM	是	是	是
捕获	是	是	是
比较	是	是	是
单次触发	是	是	是
向上/向下计数功能	是	是	是
电源模式	是	是	是
QEI 支持	是	是	否
可编程预分频器	是	是	是
影子寄存器模式	是	是	是
事件/中断	是	是	是
故障事件机制	是	是	否
自动重新加载功能	是	是	是

表 4-9. 计时器模块替换

STM32G0 计时器	MSPM0 等效功能	理由
TIM1	TIMA、TIMG8-12	高级控制，均为 16 位分辨率，支持 QEI
TIM2	TIMG12	32 位分辨率
TIM3/4	TIMG0-7	通用，16 位分辨率
TIM6/7	不限	基本计时器
TIM14	不限	与 TIM3/4 功能相同
TIM15/16/17	不限	通用
LPTIM	PD0 中的任何计时器	LPTIM 为 LFCLK、PD0 提供源 – MSPM0 中的低功耗模式

表 4-10. 计时器用例比较

功能	STM32G0 计时器	MSPM0 计时器
PWM	TIM1-4 具有边沿和中心对齐选项，TIM6-7 没有 PWM 功能。TIM15-17 仅具有边沿对齐选项。	所有计时器都具有边沿对齐或中心对齐选项
捕获	无重大差异	无重大差异
比较	无重大差异	无重大差异
单次触发	无重大差异	无重大差异
预分频器	除了 LPTIM（3 位预分频器），还有 16 位预分频器	8 位预分频器

表 4-10. 计时器用例比较 (continued)

功能	STM32G0 计时器	MSPM0 计时器
同步	TIM1-4、TIM15	所有计时器都具有该功能

计时器代码示例

有关计时器代码示例的信息，请参阅 MSPM0 SDK 示例指南。

4.6 窗口化看门狗计时器 (WWDT)

STM32G0 和 MSPM0 都提供窗口看门狗计时器。当应用程序未能在指定的时间窗口内签入时，窗口看门狗计时器 (WWDT) 会启动系统复位。

表 4-11. WWDT 命名

密钥	STM32G0	MSPM0
名称	独立看门狗计时器、窗口化看门狗计时器	窗口化看门狗计时器
缩写名称 (相同的顺序)	IWDG、WWDG	WWDT

表 4-12. WDT 功能比较

功能	STM32G0	MSPM0G	MSPM0L
窗口模式	是	是	是
间隔定时器模式	是	是	是
LFCLK 源	是	是	是
中断	是	是	是
计数器分辨率	7 位	25 位	25 位
时钟分频器	WWDG 否，IWDG 是	是	是

WWDT 代码示例

有关 WWDT 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

4.7 实时时钟 (RTC)

STM32G0 和 MSPM0[1] 都提供实时时钟 (RTC)。实时时钟 (RTC) 模块为应用程序提供时间跟踪，并以可选的二进制或二进制编码十进制数格式提供秒、分钟、小时、星期几、一月中的第几日和年的计数器。

表 4-13. RTC 功能比较

功能	STM32G0	MSPM0G
电源模式	是	是
二进制编码格式	是	是
闰年校正	是	是
可定制警报的数量	2	2
内部和外部晶体	是	是
晶体偏移校准	是	是
预分频器块	是	是
中断	是	是

RTC 代码示例

有关 RTC 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

[1] 仅 MSPM0G 器件支持 RTC。

5 模拟外设比较

5.1 模数转换器 (ADC)

STM32G0 和 MSPM0 都提供 ADC 外设来将模拟信号转换为数字等效信号。两个器件系列都具有 12 位 ADC。下表比较了 ADC 的不同功能和模式。

表 5-1. 功能集比较

功能	STM32G0	MSPM0G	MSPM0L
分辨率 (位)	12	12	12
转换速率 (Msps)	2.5	4	1.4
过采样 (位)	16	14	不适用
硬件过采样	256x	128x	不适用
FIFO	否	是	是
ADC 基准 (V)	内部：2.048、2.5	内部：1.4、2.5、VDD	内部：1.4、2.5、VDD
	当 V_{DD} < 2 时 外部：$V_{REF} = V_{DD}$	外部： $1.4 \leqslant V_{REF} \leqslant V_{DD}$	外部： $1.4 \leqslant V_{REF} \leqslant V_{DD}$
	当 $V_{DD} \geqslant 2$ 时 外部：$2 \leqslant V_{REF} \leqslant V_{DD}$		
工作电源模式	运行、睡眠	运行、睡眠、停止、待机(1)	运行、睡眠、停止、待机(1)
自动断电	是	是	是
外部输入通道(2)	高达 16	高达 16	高达 16
内部输入通道	温度传感器、VREF、VBAT	温度传感器、电源监控、模拟信号链	温度传感器、电源监控、模拟信号链
DMA 支持	是	是	是
ADC 窗口比较器单元	否	是	是
同时采样	否	是	否
ADC 数量(3)	高达 1	高达 2	高达 1

(1) ADC 可以在待机模式下触发，从而改变工作模式。
(2) 外部输入通道的数量因器件而异。
(3) ADC 的数量因器件而异。

表 5-2. 转换模式

STM32G0	MSPM0	说明
单次转换模式	单通道单次转换	ADC 对单个通道进行一次采样和转换
扫描一个通道序列	序列通道单次转换	ADC 对序列通道进行采样并转换一次。
连续转换模式	单通道重复转换	重复单通道连续采样，转换一个通道
	序列通道重复转换	对序列通道进行采样和转换，然后重复相同的序列
不连续模式	序列通道重复转换	对一组不连续的通道进行采样和转换。通过将 MEMCTRLx 映射到不同的通道，可在 MSPM0 上完成该操作。

ADC 代码示例

有关 ADC 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

5.2 比较器 (COMP)

STM32G0 和 MSPM0 系列器件都在某些器件上提供集成比较器作为可选外设。在这两个器件系列中，这些都表示为 COMPx，其中最后一个字符“x”指所考虑的特定比较器模块。在 STM32G0 系列中，这些编号为 1-3，在 MSPM0 系列中，这些编号为 0-2。这两种比较器模块都可以在器件中使用 1 个以上的比较器提供窗口比较器功能，可以从各种内部和外部源获取输入，并且可以用于触发电源模式变化或截断/控制 PWM 信号。表 5-3 汇总了 MSPM0 和 STM32G0 比较器模块逐功能比较结果。

表 5-3. COMP 功能集比较

功能	SMT32G0	MSPM0G	MSPM0L
可用的比较器	高达 3	高达 3	高达 1
输出路由	多路复用 I/O 引脚	多路复用 I/O 引脚	多路复用 I/O 引脚
	EXTI 中断	中断/事件接口	中断/事件接口
同相输入源	多路复用 I/O 引脚	多路复用 I/O 引脚	多路复用 I/O 引脚
		DAC12 输出(1)	DAC8 输出
		DAC8 输出	OPA1 输出(2)
		内部 V_{REF}：1.4 V 和 2.5 V	
		OPA1 输出(2)	
反相输入源	多路复用 I/O 引脚	多路复用 I/O 引脚	多路复用 I/O 引脚
	DAC 通道 1 和 2	内部温度传感器	内部温度传感器
	内部 V_{REF}：2.048 V 和 2.5 V	内部 V_{REF}：1.4 V 和 2.5 V	DAC8 输出
	缓冲 V_{REF} 分压器，包括：¼ V_{REF}、½V_{REF} 和 ¾V_{REF}	DAC8 输出	OPA0(3) 输出
		OPA0 输出 (3)	
可编程迟滞	无、10mV、20mV、30mV	无、10mV、20mV、30mV	无、10mV、20mV、30mV
		从 0V 到 V_{REF}/V_{DD} 的其他值（使用 DAC8）	从 0V 到 V_{DD} 的其他值（使用 DAC8）
寄存器锁	是，所有 COMP 寄存器（在器件复位时禁用）	是，某些 COMP 寄存器（写入时需要密钥）	是，某些 COMP 寄存器（写入时需要密钥）
窗口比较器配置	是	是	否（单个 COMP）
输入短路模式	否	是	是
工作模式	高速、中速	高速，低功耗	高速，低功耗
快速 PWM 关断	是	是（通过 TIMA 故障处理程序）	否
输出滤波	消隐滤波器	消隐滤波器	消隐滤波器
		可调节模拟滤波器	可调节模拟滤波器
输出极性控制	是	是	是
中断	上升沿	上升沿	上升沿
	下降沿	下降沿	下降沿
	双边沿	输出就绪	输出就绪
交换输入模式	否	是	是

(1) 仅适用于具有 DAC12 外设的器件
(2) 仅适用于具有 OPA1 外设的器件
(3) 仅适用于具有 OPA0 外设的器件

COMP 代码示例

有关 COMP 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

5.3 模数转换器 (DAC)

STM32G0 和 MSPM0 系列器件都提供 12 位 DAC 外设，用于为各种应用执行数模转换。在 STM32G0 文档中，该外设称为 DAC。在 MSPM0 技术参考手册、MSPM0 系列数据表和 MSPM0 SDK 中，12 位 DAC 外设称为 DAC12。这将 DAC12 与 8 位 DAC 区分开来，后者可用于给定 MSPM0 器件中包含的每个比较器外设。本文档的比较器部分介绍了这些额外的 8 位 DAC。该 DAC12 外设仅在 MSPM0G 系列器件上可用。

表 5-4 总结了 STM32G0 和 MSPM0G 的 12 位 DAC 外设的功能。

表 5-4. DAC 功能集比较

功能	STM32G0	MSPM0
分辨率	12 位（11.4 至 11.5 ENOB）	12 位 (11 ENOB)
输出速率	1MSPS	1MSPS
输出通道	2 (1)	1 (2)
数据格式	8 位右对齐，12 位右对齐，12 位左对齐	8 位右对齐，12 位右对齐，二进制补码或直接二进制
DMA 集成	是	是
输出路由	外部引脚	外部引脚
	内部外设连接：COMP IN-、ADC	内部外设连接：OPA IN+、COMP IN+、ADC0
内部基准电压	是，2.5V 或 2.048V	是，2.5V 或 1.4V
外部基准电压	是	是
FIFO	否	是
输出缓冲器	是	是
可配置输出失调电压	是	是
自校准模式	是	是
自动波形生成	噪声波、三角波	否
采样保持模式	是	否
触发源	外部引脚、内部计时器信号、DAC 保持时钟、DMA 欠运转	内部专用采样时间发生器、DMA 中断/事件、FIFO 阈值中断/事件、2 个硬件触发器（可从事件结构获得）

(1) 仅在某些器件上可用
(2) 双 DAC 通道计划用于未来的 MSPM0G 器件。

DAC12 代码示例

有关 DAC12 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

5.4 运算放大器 (OPA)

STM32G0 系列器件不提供集成运算放大器 (OPA) 外设，但在从 STM32G0 系列迁移到 MSPM0 系列时，您可以使用 MSPM0 内部 OPA 替换外部分立器件或根据需要缓冲内部信号。MSPM0 OPA 模块非常灵活，可以单独或组合替换检测或控制应用中的许多分立式放大器。表 5-5 包含了 MSPM0 OPA 模块的主要功能，OPA 代码示例包含您可以重新创建的常见 OPA 配置示例。

表 5-5. MSPM0 OPA 功能集

功能	MSPM0 实现
输入类型	轨至轨（可以启用或禁用）
增益带宽	1MHz（低功耗模式）
	6MHz（标准模式）

表 5-5. MSPM0 OPA 功能集 (continued)

功能	MSPM0 实现
放大器配置	通用模式
	缓冲模式
	PGA 模式 (反相或同相)
	差分放大器模式
	级联放大器模式
输入/输出路由	外部引脚布线
	ADC 和 COMP 模块的内部连接
故障检测	烧毁电流源 (BCS)
斩波稳定	标准 (可选斩波频率)
	ADC 辅助斩波
	禁用
基准电压	内部 VREF (仅限 MSPM0G 器件)
	DAC12 (仅限 MSPM0G 器件)
	DAC8 (仅限具有 COMP 模块的器件)

OPA 代码示例

有关 OPA 代码示例的信息，请参阅 MSPM0 SDK 示例指南。

5.5 电压基准 (VREF)

STM32G0x 和 MSPM0 都具有内部基准，可用于为内部外设提供基准电压并输出到外部外设。

表 5-6. 功能集比较

功能	STM32G0	MSPM0G	MSPM0L
内部基准 (V)	2.048、2.5	1.4、2.5	1.4、2.5
外部基准 (V)	当 V_{DD} < 2 时，V_{REF} = V_{DD} 当 V_{DD} ≥ 2 时，2 ≤ V_{REF} ≤ V_{DD}	外部：1.4 ≤ V_{REF} ≤ V_{DD}	外部：1.4 ≤ V_{REF} ≤ V_{DD}
输出内部基准	是	是	是
在内部连接到 ADC	是	是	是
在内部连接到 DAC	是	是	否
在内部连接到 COMP	否	是	否
在内部连接到 OPA	不适用	是	否

表 5-7. 控制位比较

STM32G0x VREFBUF 位	MSPM0 等效功能
VREFBUF 位 3 (VRR)	CTL1 位 0 (READY)
VREFBUF 位 2 (VRS)	CTL0 位 7 (BUFCONFIG)
VREFBUF 位 1 (HIZ)	不适用
VREFBUF 位 0 (ENVR)	CTL0 位 0 (ENABLE)
	对于采样保持模式：CTL0 位 8 (SHMODE)

对于 MSPM0 VREF，您必须启用电源位 PWREN 位 0 (ENABLE)。

VREF 代码示例

有关使用 VREF 的代码示例，请参阅 MSPM0 SDK 示例指南。

6 修订历史记录

注：以前版本的页码可能与当前版本的页码不同

日期	修订版本	说明
2023 年 3 月	A	首次公开发布

重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265