MICROCHIP

AN2775

使用 SAM L10 MCU 的 DMAC 演示 CPU 的使用率

简介

Microchip | SMART SAM L10/SAM L11 中的直接存储器访问控制器（Direct Memory Access Controller，DMAC）支持存储器和外设之间的传输，从而减轻 CPU 处理这些任务的负载。DMAC 可在最少 CPU 干预的情况下实现高数据传输速率，并释放 CPU 时间。它包含直接存储器访问引擎和循环冗余校验（Cyclic Redundancy Check，CRC）引擎。通过访问所有外设，DMAC 可以处理通信模块之间的数据自动传输。

本文档演示了使用或不使用 DMA 执行应用程序时的 CPU 使用率。使用 ADC 对来自光传感器的模拟数据进行采样，并将数据发送到 USART。在本文档中，计算 CPU 使用率时考虑了使用和不使用 DMA 进行数据传输这两种情况。

特性

本应用涵盖以下外设特性：

- DMA 数据传输方向：
  - 外设（ADC）到外设（USART）
  - 外设（ADC）到存储器（SRAM）
  - 存储器（SRAM）到存储器（SRAM）
  - 存储器（SRAM）到外设（USART）
- 传输触发源
  - 软件
  - 外设（ADC 结果就绪，USART 数据寄存器为空）
- 通过链接多个描述符来实现多缓冲区传输模式
- 支持三个独立通道，每个通道均带有自动描述符
- 每个优先级内采用固定优先级机制
- 单个块传输支持传输 1 KB-256 KB 的数据
- 多个寻址模式
  - 静态
  - 可编程递增机制
- 事务完成中断生成
- DMA 事件输出
- 用于外设到外设直接通信信号传输的事件系统
- 用于实现精确定时的事件触发 ADC 转换
- 转换结果的 DMA 传输
- 使用系统定时器（SYSTICK）计算 CPU 使用率

先决条件

本文档中讨论的解决方案需要您对以下工具有基本的了解：

- 带 USB 线的 SAM L10/SAM L11 Xplained Pro 评估工具包

本文档包含以下外设的概述：

- DMAC
- SERCOM – USART
- EVSYS
- ADC
- SYSTICK

有关每个外设的详细信息，请参见产品数据手册。

目录

1. 设置

本应用程序是针对 SAM L10 Xplained Pro 板开发的。本章介绍了测试本应用程序所需的硬件和软件设置。

1.1 硬件设置

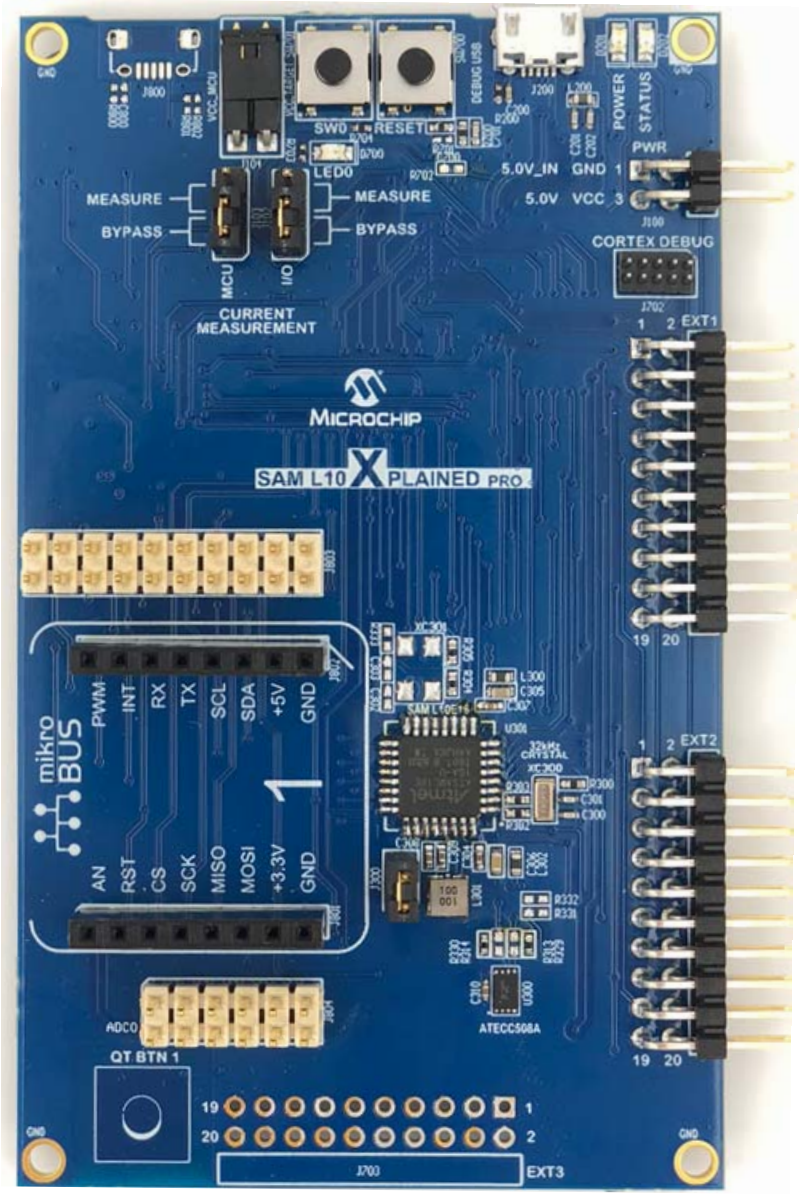
1.1.1 SAM L10 Xplained Pro

Microchip SAM L10 Xplained Pro 评估工具包是用于评估 ATSAML10E16A-AU 单片机的硬件平台。

评估工具包提供一系列功能，能够让 ATSAML10E16A 用户立即开始使用单片机外设，并了解将器件集成到定制设计中的各种步骤。

评估工具包通过 Xplained Pro 扩展插座支持各种 Xplained Pro 扩展板。SAM L10 Xplained Pro 有三个这样的 Xplained Pro 扩展插座，本应用中使用 EXT1。

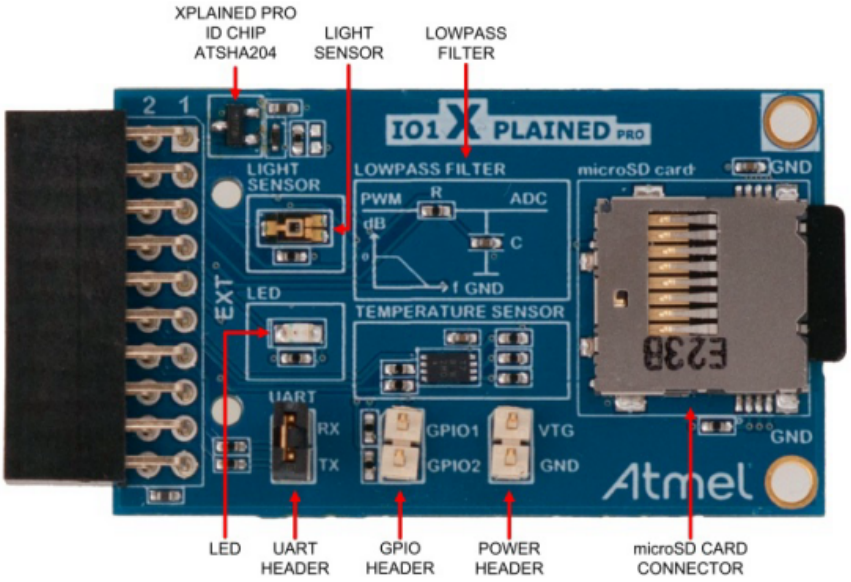
图 1-1. SAM L10 Xplained Pro 板



1.1.2 IO1 Xplained Pro 扩展板

Microchip IO1 Xplained Pro 扩展板是 Xplained Pro 平台的通用扩展板。它连接至任一 Xplained Pro MCU 板的任一 Xplained Pro 标准扩展插座。扩展板使用标准 Xplained Pro 扩展插座上的所有可用功能。

图 1-2. IO1 Xplained Pro 扩展板



Microchip IO1 Xplained Pro 设计用于连接标有 EXT1 或 EXT2 的 Xplained Pro 插座。但是，它与 Xplained Pro 板上提供的所有 Xplained Pro EXT 插座兼容。要确定可以使用哪些 Xplained Pro EXT 插座，需要了解相应 Xplained Pro 评估工具包的引脚排列。本文档介绍了使用 EXT1 插座的演示。

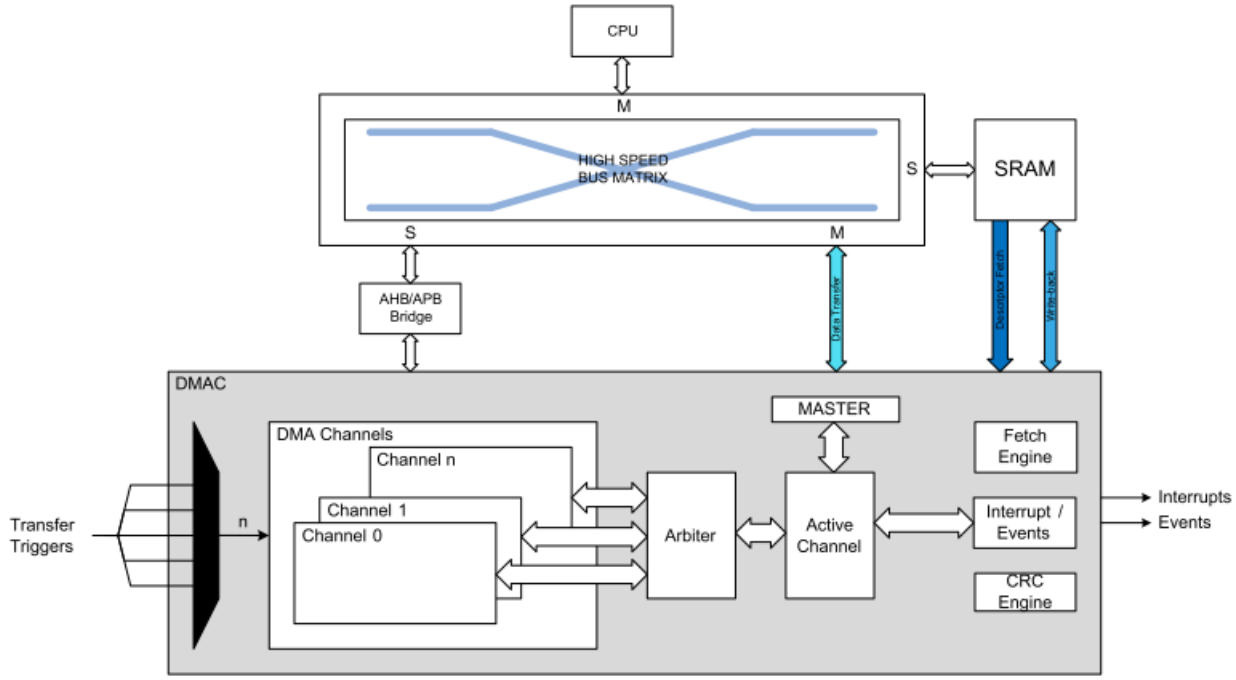
IO1 Xplained Pro 配有光传感器。扩展板的引脚 3 可用于此传感器。传感器数据可由 Xplained Pro MCU 板上的 ADC 引脚收集。在 SAM L10 Xplained Pro 工具包中，它连接到 EXT1 插座。本应用将 IO1 Xplained Pro 板上的光传感器作为 ADC 的模拟输入。

2. 直接存储器访问控制器（DMA）

本章介绍 DMAC 特性及其与本文档相关的操作。有关其操作和配置的详细说明，请参见产品数据手册。

2.1 框图

图 2-1. DMAC 框图



2.2 功能说明

2.2.1 DMAC 基本操作

直接存储器访问控制器（DMAC）包含直接存储器访问引擎和循环冗余校验（CRC）引擎。DMAC 可实现存储器和外设之间的数据传输，从而为 CPU 减轻任务负担。DMAC 可在最少的 CPU 干预下实现高数据传输速率，并释放 CPU 时间。通过访问所有外设，DMAC 可以处理通信模块之间的数据自动传输。这样，CPU 便可休眠更长时间，从而降低功耗。

存储器与外设之间的完整 DMA 读写操作称为 DMA 事务。DMA 在写入目标地址之前从源地址读取数据。完成上一个写操作时，将读取新数据。事务由触发信号启动并使用 DMA 通道。DMA 触发源可以是应用程序软件、外设或事件系统（EVSYS）中的事件。每个读写操作都是以块为单位完成的。传输大小由块传输大小控制，并在软件中进行配置。块的大小可以是 1 至 64K 个节拍。节拍可以是字节、半字或字。

2.2.2 DMAC 通道

DMA 实现了 8 个通道，可实现 8 个独立传输。每个 DMA 通道都有一个单独的传输控制描述符设置，存储在 SRAM 中。

传输控制描述符定义源地址和目标地址、源地址和目标地址递增设置、块传输计数以及可选事件输出条件选择。

源寻址和目标寻址可以是静态的或递增的。

每个通道都有专用的 I/O 寄存器，用于控制触发模式（外设/软件）、外设触发源类型、事件输入操作和通道优先级设置。

每个活动通道都有专用的回写存储段，用于保持当前的传输设置和状态。

使能多个通道时，支持 4 级通道优先级，每个优先级都支持固定或循环机制。

2.2.3 DMAC 传输操作

可以执行单个事务（仅使用一个描述符），也可以执行多个事务（使用链接描述符）。可以使用相同的 DMA 通道使能单个或多个块传输。

当使能 DMA 外设和相应通道时，将在收到触发请求时进行传输。传输类型可以是节拍、块（一组节拍构成块）或事务（一组块构成事务）。

DMA 传输完成后，通道自动禁止。如果为某通道定义了单个描述符，则在块传输完成时，该通道会自动禁止。在链接描述符的情况下，执行最后一个描述符后，将禁止相应通道。

2.2.4 其他特性

通道暂停和恢复

可以通过软件随时暂停或恢复通道操作，也可以在完成可选块传输时暂停通道操作。

中断请求

以下情形下可以生成中断请求：

- 事务完成
- 可选块传输完成
- DMA 控制器检测到总线错误
- 通道操作暂停

事件输入

事件输入操作在最不重要的 DMA 通道上可用。事件可以编程为触发以下操作：

- 传输
- 周期性传输
- 条件传输
- 暂停或恢复通道操作

事件输出

事件输出选择可用于最不重要的 DMA 通道。以下情形下可以生成事件：

- 每个 AHB 数据传输完成
- 可选块传输完成
- 整个事务完成

3. 外设概述

本章概述与本文档相关的其他外设。有关操作和配置的详细说明，请参见产品数据手册中的相应章节。

3.1 事件系统

凭借事件系统（EVSYS），外设之间可实现自主、低延时且可配置的通信。多个外设可配置为发出和响应称为事件的信号。

生成事件的确切条件或接收事件时采取的操作是每个模块特定的。响应事件的外设称为事件用户。发出事件的外设称为事件生成器。外设可以有多个事件生成器和多个事件用户。

无需 CPU 干预即可进行通信，而且无需消耗总线或 RAM 带宽等系统资源。与传统基于中断的系统相比，这减轻了 CPU 和其他系统资源的负载。

在本文档中，EVSYS 配置为使用“DMA 通道 0 传输完成”（DMAC CH0）作为事件生成器，并将 ADC 启动转换（ADC START）作为事件用户。在 DMA 传输完成时，DMA 将触发输出事件，并且事件系统在接收到该事件时会触发 ADC 启动转换。

3.2 模数转换器

模数转换器（Analog-to-Digital Converter，ADC）将模拟信号转换为数字值。ADC 的分辨率最高为 12 位，最高可转换 1 Msps。输入选择灵活，可以执行差分和单端测量。此外，还提供多个内部信号输入。

ADC 测量可以通过应用软件启动，也可以通过器件中其他外设的传入事件启动。可以使用内部和外部参考电压。

ADC 可配置为生成 8 位、10 位或 12 位结果，从而缩短转换时间。ADC 转换结果以左对齐或右对齐的形式提供，当结果表示为有符号值时，可以简化计算。转换完成后，可以使用 DMA 将 ADC 结果直接移至存储器或外设。

在本示例中，ADC 配置为 8 位分辨率，并使用 DMA 将 ADC 结果传输到配置的目标地址（可以是外设或存储器）。来自 DMA 的事件输入用于触发下一次 ADC 转换。如果不使用 DMA 实现该过程，则使用软件触发。

3.3 SERCOM - USART

SERCOM 串行引擎包括发送器和接收器、波特率发生器和地址匹配功能。发送器由一个写缓冲器和一个移位寄存器组成。接收器由一个两级接收缓冲器和一个移位寄存器组成。波特率发生器能够在 GCLK_SERCOMx_CORE 时钟或外部时钟的驱动下运行。

串行通信接口（SERCOM）可配置为支持多种模式：I^2C、SPI 和 USART。配置并使能后，所有 SERCOM 资源都专用于所选模式。

通用同步和异步收发器（USART）是串行通信接口（SERCOM）中的可用模式之一。

通过向数据寄存器装入要发送的数据来启动数据传输。当移位寄存器为空并准备好发送新帧时，TxDATA 中的数据将移至移位寄存器。当移位寄存器装入数据时，将传输一个完整的帧。

当整个帧加停止位移出并且没有新数据写入 DATA 寄存器时，中断标志状态和清除寄存器中的发送完成中断标志（INTFLAG.TXC）置 1，并产生可选中断。

只有当中断标志状态和清零寄存器中的数据寄存器空标志（INTFLAG.DRE）置 1（表示寄存器为空并准备好接收新数据）时，才应写入数据寄存器。

当发送缓冲区（TX 数据）为空时，USART 可以生成 DMA 请求。写入数据时，将清除请求。

在本应用中，EDGB CDC（SERCOM0）用于将 ADC 结果数据传输到终端。

3.4 系统定时器（SYSTICK）

系统定时器是一个 24 位定时器，可扩展处理器和 NVIC 的功能。有关其他信息，请参见 Arm® Technical Reference Manual，可从以下位置下载：www.arm.com。

定时器包含以下寄存器：

- 控制和状态寄存器（SYST_CSR）。该寄存器可配置 SYSTICK 时钟、使能计数器、允许 SYSTICK 中断，并指示计数器状态。
- 计数器重载值寄存器（SYST_RVR）。该寄存器为计数器提供绕回值。
- 计数器当前值寄存器（SYST_CVR）。

使能时，定时器从 SYST_CVR 寄存器中的值开始递减计数。当计数器达到零时，它会在下一个时钟沿重载 SYST_RVR 寄存器中的值，然后在后续时钟递减。计数器达到零时的重载称为绕回。可以允许中断，每次计数器绕回时都会触发该中断。

在本应用中，计数器装入了最大计数值，用于在计算 CPU 利用率时获取时间戳。SYSTICK 以处理器时钟作为时钟源运行。

4. 示例实现

本章详细介绍应用的实现。

本文档的目的是演示各种功能及其配置。除此之外，还会计算使用或不使用 DMA 时的 CPU 利用率。将重点介绍 DMAC 在减轻 CPU 负载方面的用途。

在示例实现中，ADC 将输入模拟信号转换为数字值，并将结果传输到 USART。IO1 Xplained Pro 中的光传感器通过 EXT1 插座用作 ADC 的输入。

本文档介绍了可实现目标的三种不同情形（即使用和不使用 DMAC）。每种情形都是通过单独的 Atmel Start 示例项目实现的，用户需要选择适当的示例项目。以下各节详细介绍了每个示例项目。

4.1 使用 DMAC 进行外设到外设的传输（ADC 到 USART）

此传输类型的 Atmel Start 示例项目为 *ADC DMAC USART*。本例中，ADC 结果被写入 USART DATA 寄存器，以说明外设到外设的 DMA 传输类型。

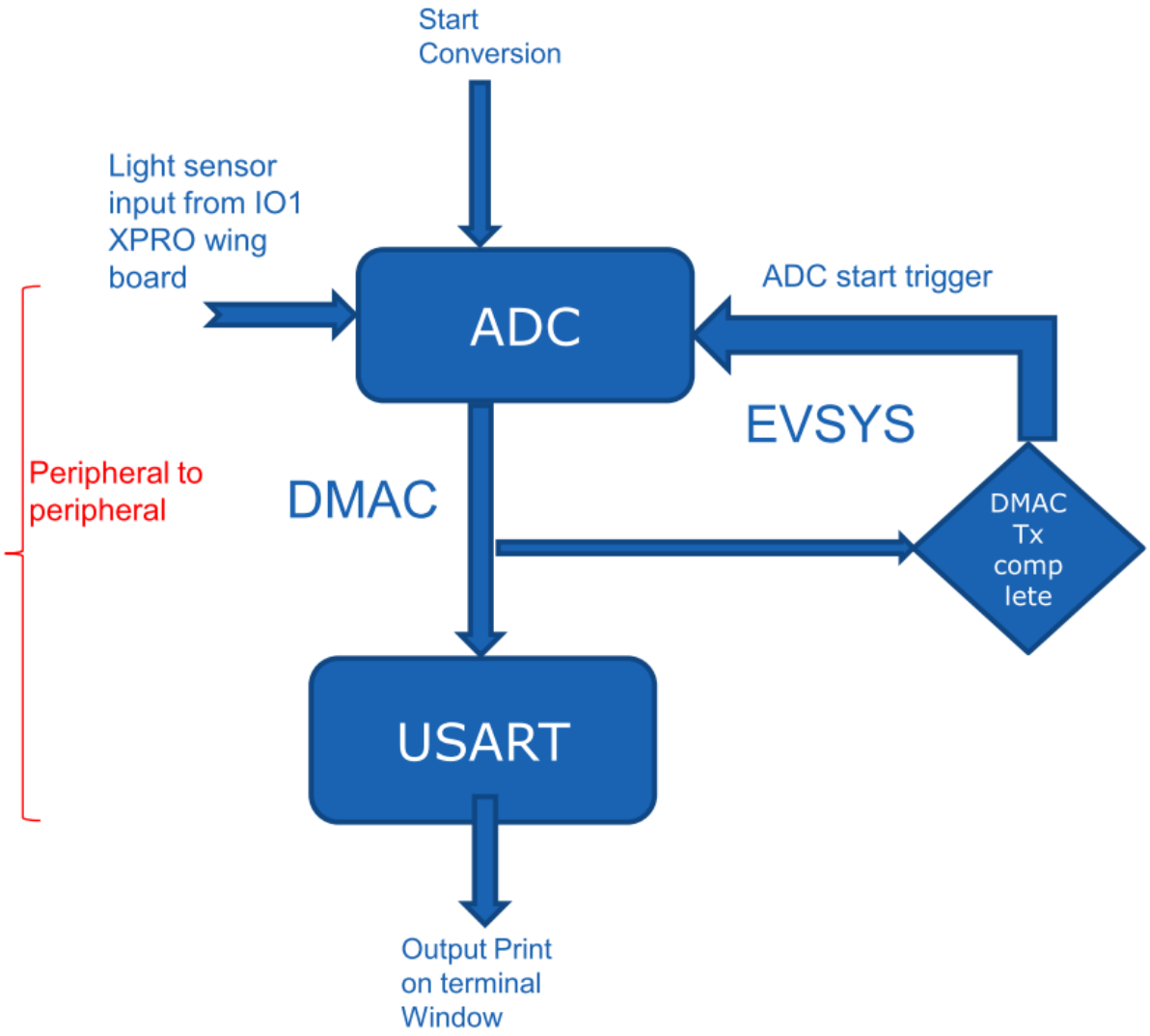
4.1.1 应用配置和实现

DMAC 配置为在 ADC 结果就绪（外设触发源）时触发数据传输到配置的目标地址。配置的目标地址是 USART 数据寄存器地址，源地址是 ADC 结果寄存器地址。本例中，DMA 源地址和目标地址是静态的，因为两个寄存器地址都是固定的。

ADC 配置为事件用户，它将在通过事件系统（EVSYS）从 DMAC 接收到事件信号时启动转换。EVSYS 通过 DMAC 配置为事件生成器，ADC 为事件用户。

第一次 ADC 转换由软件触发信号触发。一旦输入被采样并且结果准备就绪，就会触发从 ADC 结果寄存器到 USART 数据寄存器的 DMA 传输。完成到 USART 数据寄存器的传输后，来自 DMA 的事件信号会触发下一个 ADC 转换，如此循环继续。整个操作过程如下图所示。

图 4-1. 使用 DMAC 时的外设到外设传输



使用 DMAC 和 EVSYS 执行整个操作，不会中断 CPU。DMAC 块传输大小配置为 1024 字节（BLOCK_COUNT），配置为在块传输完成时产生中断。块传输完成后，DMAC 通道将自动禁止。

振荡器控制器配置：

选择 16 MHz 内部振荡器，频率设置为 16 MHz。

- 通过将 OSC16MCTRL.FSEL[1:0]设置为 0x11，将 16 MHz 内部振荡器频率配置为 16 MHz
- 通过将 OSC16MCTRL.ENABLE 设置为 1 来使能 16 MHz 内部振荡器

GCLK 配置：

GCLK0 配置为使用 16 MHz 内部振荡器（OSC16M）作为时钟源。

- 通过将 GENCTRL0.SRC[4:0]设置为 0x05，将 GCLK0 时钟源配置为 16 MHz 振荡器输出（OSC16M 振荡器输出）
- 通过将 GENCTRL0.GENEN 设置为 1 来使能 GCLK0
- 使能 GCLK0 输出以测量 CPU 时钟频率。I/O 引脚 PA27 配置为输出 GCLK0
- 通过将 PMUX13.PMUXO[3:0]设置为 0x07（选择外设功能 H），将 GCLK_IO[0]外设功能分配给 PA27。
- 将 GENCTRL0.OE 设置为 1 以在 GCLK_IO 上输出发生器时钟

MCLK 配置：

GCLK0 始终是 GCLK_MAIN 的直接源，由主时钟模块使用。

ADC 配置：

ADC 配置为使用 GCLK0（16 MHz）作为时钟源。ADC 转换分辨率设置为 8 位，ADC 参考设置为½ VDDANA。将 ADC 时钟 64 分频（ADC 时钟= 16 MHz/64 = 250 kHz）。选择 PA02 作为 ADC 的正输入，GND 作为负输入。

- 通过将 PCHCTRL20.GEN[2:0]配置为 0x0（通用时钟发生器 0），选择 GCLK0 作为 ADC 模块的时钟源
- 通过将 CTRLB.PRESCALER[2:0]设置为 0x05，将 ADC 外设的输入时钟 64 分频
- 通过将 CTRLC.RESSEL[1:0]设置为 0x3，将 ADC 分辨率设置为 8 位
- 通过将 REFCTRL.REFSEL[3:0]设置为 0x02，将 ADC 参考电压设置为½ VDDANA
- 通过将 INPUTCTRL.MUXPOS[4:0]设置为 0x00，选择 AIN0 作为 ADC 的正输入（ADC AIN0 引脚）
- 通过将 INPUTCTRL.MUXNEG[4:0]设置为 0x18，选择 GND 作为 ADC 的负输入（内部地）
- 通过将 PMUX1.PMUXE[3:0]设置为 0x01，将 AIN[0]外设功能分配给 PA02（选择外设功能 B）
- ADC 配置为出现事件输入时触发事件。DMA 传输完成事件将用作事件输入。
- 通过设置 EVCTRL.STARTEI = 1，在出现事件输入时触发 ADC 转换

DMAC 配置：

DMA 通道 0 配置为在出现 ADC 结果就绪触发信号时执行节拍（8 位）传输。每次节拍传输都会生成一个事件。由于源地址和目标地址是静态的，因此它们不会递增。一旦事务中的最后一个块传输完成，DMA 通道就会被禁止。

- 通过将 CHID.ID[3:0]设置为 0x0，选择要配置的 DMA 通道（本例中为 DMA 通道 0）
- 通过将 CHCTRLB.TRIGSRC[4:0]设置为 0x13（ADC 结果就绪触发），选择 ADC 结果就绪作为 DMA 触发源
- 要在每次触发时执行节拍（8 位）传输，需要设置触发操作，即将 CHCTRLB.TRIGACT[1:0]设置为 0x2（每次节拍传输需要一次触发）

- 通过设置 CHCTRLB.EVOE = 1 使能通道事件输出。ADC 将（通过事件系统）使用此事件输出来启动 ADC 转换。
- DMA 通道传输描述符寄存器配置如下：
  - 源地址和目标地址是静态的。BTCTRL.SRCINC =0 且 BTCTRL.DSTINC = 0。
  - 节拍大小配置为 1 字节。BTCTRL.BEATSIZE[1:0] = 0x0（8 位总线传输）。
  - 要在节拍（1 字节）传输完成时生成事件输出，需要将 BTCTRL.EVOSEL[1:0]设置为 0x3（节拍传输完成时的事件选通）
  - 将 DMA 通道 0 源地址（SRCADDR[31:0]）配置为 ADC 结果寄存器
  - 将 DMA 通道 0 目标地址（DSTADDR[31:0]）配置为 USART 发送寄存器
  - 通过将 BTCNT[15:0]设置为 1024，将块传输计数设置为 1024
  - 必须通过将 DESCADDR 设置为 0，将通道 0 的下一个描述符地址设置为 NULL
- 通过将 CHINTENSET.TCMPL 设置为 1 来允许 DMA 通道 0 传输完成中断

注： 所有传输描述符必须保留在 SRAM 中。存储在描述符存储段基址（BASEADDR）和回写存储段基址（WRBADDR）寄存器中的地址将告知 DMAC 描述符存储段和回写存储段的位置。由于 BASEADDR 仅指向通道 0 的第一个传输描述符，因此所有第一个传输描述符必须存储在连续的存储段中，其中传输描述符必须根据其通道编号进行排序。更多信息，请参见 SAM L10/L11 数据手册。

USART 配置：

SERCOM0 配置为 USART 操作模式，GCLK0（16 MHz）作为 SERCOM0 模块的时钟源。USART RX 和 TX 线分别映射到 PA25（SERCOM0 PAD[3]）和 PA24（SERCOM0 PAD[2]）。

波特率设置为 460800。有关为何选择此波特率的详细信息，请参见 USART 波特率和 ADC 采样频率。

事件系统配置：

事件系统的通道 0 配置为 DMA 通道 0 作为事件生成器，ADC 启动转换作为事件用户。

- 通过将 CHANNEL0.EVGEN[5:0]设置为 0x26（ADC 结果就绪），选择 DMA 通道 0 作为事件系统通道 0 的事件生成器
- 通过设置 CHANNEL0.PATH[1:0] = 0x2（异步路径），为事件系统的通道 0 选择异步路径
- 通过将 USER14.CHANNEL[3:0]设置为 0x1（此字段中的值 x 选择通道 n = x-1），选择 ADC 启动转换作为事件系统通道 0 的用户

I/O 引脚配置：

I/O 引脚 PA04 配置为数字输出。PA04 用于测量 CPU 使空闲循环计数器递增所花费的时间。随后，将使用该时间计算 CPU 空闲时间。

I/O 引脚 PA06 配置为数字输出。PA06 将在中断服务处理程序中翻转，并且还用于计算 CPU 空闲时间。

I/O 引脚 PA27 配置为输出 GCLK0 频率（如前文的 GCLK 配置中所述），用于测量 CPU 时钟频率。CPU 时钟频率将用于计算总执行时间。

4.1.2 CPU 利用率计算

MCU 和外设初始化完成后，应用程序通过读取 SYSTICK 当前值寄存器来获取时间戳并将其保存到变量（即 time_stamp1）中。然后，它使用软件触发（SWTRIG.START = 1）启动第一次 ADC 转换。当前一个 ADC 结果通过 DMA 传输到目标（USART DATA 寄存器）时，DMA 生成的事件会触发后续转换。

当转换正在进行时，应用程序进入 while (1)循环，使空闲循环计数器（即 idle_loop_counter）递增，并翻转 I/O 引脚 PA04。使用示波器探测 PA04，计算使一个空闲循环计数器递增所需的时间，从而得出 CPU 空闲的总持续时间。

全部 1024 次 ADC 转换完成后，DMA 中断将采用另一个时间戳（即 time_stamp2），并且标志将置 1 以指示传输完成。传输完成后，应用程序通过获取转换开始之前的时间戳（即 time_stamp1）和 1024 次传输完成时所用的时间戳（即 time_stamp2）的差异来计算完成传输所需的总周期数。空闲循环计数器值和完成传输所需的总周期数的值将在主计算机上运行的终端应用程序上输出。

注： 有关基于观察到的结果计算 CPU 利用率的详细说明，请参见不同情形之间的 CPU 利用率分析。

4.2 使用 DMAC 时的外设到存储器和存储器到外设传输（ADC 到 SRAM 和 SRAM 到 USART）

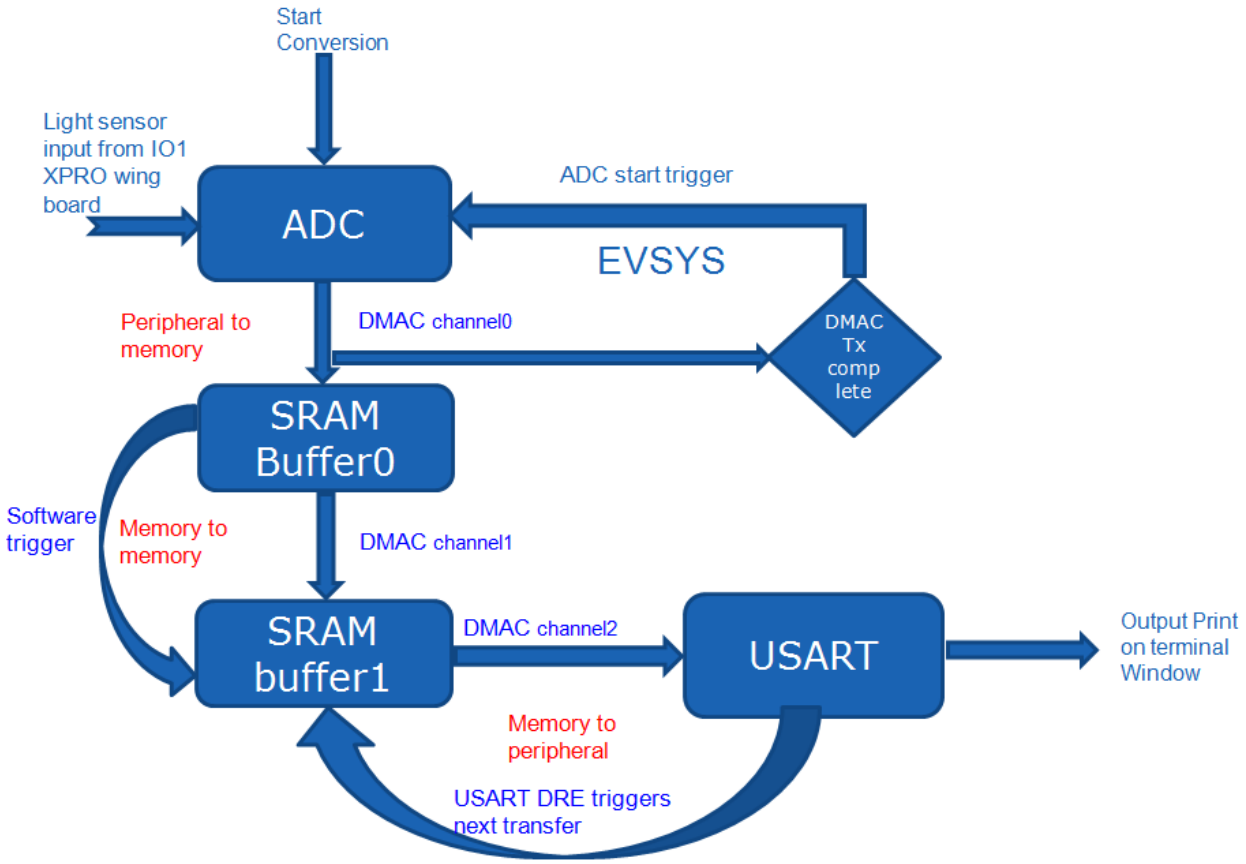
这种情形使用了三个 DMAC 通道来演示每种传输类型。存储器到存储器类型的 DMA 传输仅用于演示目的，应用程序不需要该类型也可正常工作。

注： 请参见 Atmel Start 示例文档 *ADC DMAC MEM MEM USART*。

4.2.1 应用配置和实现

本例中使用三个 DMA 通道。DMA 通道 0 配置为使用 ADC 结果就绪作为触发源，将 1024 个节拍（1 个节拍= 1 个字节）从 ADC 结果寄存器（外设）传输到 SRAM 缓冲区（存储器）。DMA 通道 1 配置为使用软件触发将 1024 个字节从 SRAM 缓冲区（存储器）传输到另一个 SRAM 缓冲区（存储器）。DMA 通道 2 配置为使用 USART DATA 寄存器为空（DRE）作为触发源，将 1024 个字节从第二个 SRAM 缓冲区（存储器）传输到 USART 数据寄存器（外设）。整个操作过程如下图所示。

图 4-2. 使用 DMAC 时的外设到存储器和存储器到外设传输



DMA 通道 0 配置（外设到存储器）：

DMAC 通道 0 配置为通过 ADC 结果就绪实现外设触发。源地址是静态的，因为它是 ADC 结果寄存器。由于 SRAM 缓冲区需要存储来自 ADC 的 1024 字节采样，因此目标地址会递增。在完成每次节拍传输时使能事件输出，该事件输出用于触发下一个 ADC 转换。

- 通过将 CHID.ID[3:0]设置为 0x0，选择要配置的 DMA 通道（本例中为 DMA 通道 0）
- 通过将 CHCTRLB.TRIGSRC[4:0]设置为 0x13（ADC 结果就绪触发），选择 ADC 结果就绪作为 DMA 触发源
- 要在每次触发时执行节拍（8 位）传输，需要设置触发操作，即将 CHCTRLB.TRIGACT[1:0]设置为 0x2（每次节拍传输需要一次触发）
- 通过设置 CHCTRLB.EVOE = 1 使能通道事件输出。ADC 将（通过事件系统）使用此事件输出来启动 ADC 转换。
- DMA 通道传输描述符寄存器配置如下：
  - 源地址是静态的，目标地址是递增的。BTCTRL.SRCINC = 0 且 BTCTRL.DSTINC = 1
  - 节拍大小配置为 1 字节。BTCTRL.BEATSIZE[1:0] = 0x0（8 位总线传输）。
  - 要在节拍（1 字节）传输完成时生成事件输出，需要将 BTCTRL.EVOSEL[1:0]设置为 0x3（节拍传输完成时的事件选通）
  - 将 DMA 通道 0 源地址（SRCADDR[31:0]）配置为 ADC 结果寄存器
  - 将 DMA 通道 0 目标地址（DSTADDR[31:0]）配置为 SRAM 缓冲区
  - 通过将 BTCNT[15:0]设置为 1024，将块传输计数设置为 1024

  - 通过将 CHINTENSET.TCMPL 设置为 1 来允许 DMA 通道 0 传输完成中断
  - 必须通过将 DESCADDR 设置为 0，将通道 0 的下一个描述符地址设置为 NULL
- 通过将 CHINTENSET.TCMPL 设置为 1 来允许 DMA 通道 0 传输完成中断

1024 字节样本从 ADC 传输到 SRAM 缓冲区后，就会执行 DMA 块传输完成中断处理程序。中断处理程序触发 DMA 通道 1 传输。

DMA 通道 1 配置（存储器到存储器）：

通道 1 配置有软件触发，传输类型为事务，即，一旦软件触发传输，存储在一个 SRAM 缓冲区中的所有 ADC 结果都将传输到存储在 SRAM 中的另一个缓冲区。由于源地址和目标地址都是 SRAM 缓冲区，因此源地址和目标地址都会递增。

- 通过将 CHID.ID[3:0]设置为 0x1，选择要配置的 DMA 通道（本例中为 DMA 通道 1）
- 通过将 CHCTRLB.TRIGSRC[4:0]设置为 0x00（仅软件/事件触发），选择软件触发作为 DMA 触发源
- 要通过单次触发将全部 1024 个字节从一个 SRAM 缓冲区传输到另一个 SRAM 缓冲区，需设置触发操作，即将 CHCTRLB.TRIGACT[1:0]设置为 0x3（每个事务需要一次触发）
- DMA 通道传输描述符寄存器配置如下：
  - 源地址和目标地址递增。BTCTRL.SRCINC = 1 且 BTCTRL.DSTINC = 1
  - 通过将 BTCTRL.STEPSIZE[2:0]设置为 0x0，将地址增量步长设置为 1 节拍
  - 节拍大小配置为 1 字节。BTCTRL.BEATSIZE[1:0] = 0x0（8 位总线传输）
  - 将 DMA 通道 1 源地址（SRCADDR[31:0]）配置为源 SRAM 缓冲区的地址
  - 将 DMA 通道 0 目标地址（DSTADDR[31:0]）配置为目标 SRAM 缓冲区的地址
  - 通过将 BTCNT[15:0]设置为 1024，将块传输计数设置为 1024
  - 必须通过将 DESCADDR 设置为 0，将通道 1 的下一个描述符地址设置为 NULL
- 通过将 CHINTENSET.TCMPL 设置为 1 来允许 DMA 通道 1 传输完成中断

1024 字节从 SRAM 中的源缓冲区传输到 SRAM 中的目标缓冲区后，就会执行 DMA 块传输完成中断处理程序。中断处理程序触发 DMA 通道 2 传输，以将数据从 SRAM 中的目标缓冲区传输到 USART。

DMA 通道 2 配置（存储器到外设）：

通道 2 配置为采用外设触发和节拍传输类型。只要 USART 数据寄存器为空，就应将 SRAM 缓冲区中的一个字节写入其中，即只要 USART 数据寄存器为空（DRE）并准备好写入新数据，就会触发从源到目标的 DMA 传输（通过通道 2 实现）。

- 通过将 CHID.ID[3:0]设置为 0x2，选择要配置的 DMA 通道（本例中为 DMA 通道 2）
- 通过将 CHCTRLB.TRIGSRC[4:0]设置为 0x05，选择 SERCOM0 TX 触发作为 DMA 触发源
- 要在每次触发时执行节拍（8 位）传输，需要设置触发操作，即将 CHCTRLB.TRIGACT[1:0]设置为 0x2（每次节拍传输需要一次触发）
- DMA 通道传输描述符寄存器配置如下：
  - 源地址是递增的，目标地址是静态的。BTCTRL.SRCINC =1 且 BTCTRL.DSTINC = 0。
  - 通过将 BTCTRL.STEPSIZE[2:0]设置为 0x0，将地址增量步长设置为 1 节拍
  - 节拍大小配置为 1 字节。BTCTRL.BEATSIZE[1:0] = 0x0（8 位总线传输）。
  - 将 DMA 通道 2 源地址（SRCADDR[31:0]）配置为 SRAM 缓冲区的地址
  - 将 DMA 通道 2 目标地址（DSTADDR[31:0]）配置为 USART 发送寄存器的地址
  - 通过将 BTCNT[15:0]设置为 1024，将块传输计数设置为 1024
  - 必须通过将 DESCADDR 设置为 0，将通道 2 的下一个描述符地址设置为 NULL

- 通过将 CHINTENSET.TCMPL 设置为 1 来允许 DMA 通道 2 传输完成中断

与其他通道不同，此通道应在通道 1 传输完成时使能。原因是由于之前未进行任何通信，USART DRE 始终置 1。因此，如果在初始化期间使能此通道，则由于 USART DRE 已置 1，DMA 传输将立即在通道 2 上启动，从而导致错误操作。

在 DMA 通道 2 完成传输之后，标志置 1，指示传输结束，将采用相应时间戳（即，time_stamp2）计算 CPU 利用率。

其他外设配置（如时钟、ADC、事件系统、USART 和 I/O）仍与使用 DMAC 进行外设到外设传输时相同。

4.2.2 CPU 利用率计算

应用程序通过读取 SYSTICK 当前值寄存器来获取初始时间戳，并将其保存到变量（即 time_stamp1）中。然后，它使用软件触发（SWTRIG.START = 1）启动第一次 ADC 转换。当前一个 ADC 结果通过 DMA 传输到目标（USART DATA 寄存器）时，DMA 生成的事件会触发后续转换。

与使用 DMAC 进行外设到外设传输时相似，当传输正在进行时，应用程序进入 while (1)循环，使空闲循环计数器递增，并翻转 I/O 引脚 PA04。全部 1024 次 ADC 转换完成后，DMA 通道 2 中断处理程序将采用另一个时间戳（即 time_stamp2），并且标志将置 1 以指示传输完成。

注： 有关更多信息，请参见不同情形之间的 CPU 利用率分析。

4.3 不使用 DMAC 时的外设到存储器和存储器到外设传输（ADC 到 SRAM 和 SRAM 到 USART）

在这种情况下，应用程序通过中断处理实现，而不使用 DMA。这样做是为了演示 DMAC 在减轻 CPU 负载方面的用途。

外设到存储器：ADC 结果存储在 ADC 驱动程序缓冲区（即 Buffer1）中。

存储器到存储器：一旦所有 ADC 结果都可用，数据就会从 ADC 驱动程序缓冲区（Buffer1）复制到应用程序缓冲区（即 Buffer2）。

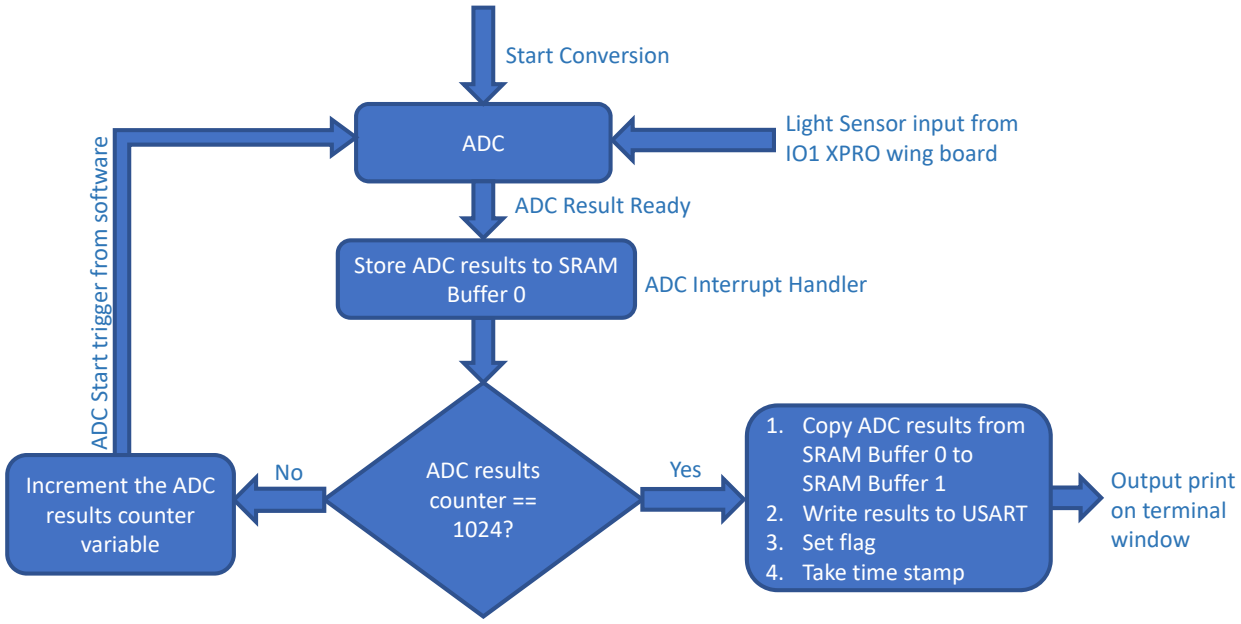
存储器到外设：应用程序缓冲区（Buffer2）传输到 USART。

注： 请参见 Atmel Start 示例 *ADC NO DMAC MEM MEM USART*。

4.3.1 应用配置和实现

时钟、USART 和 I/O 配置仍与使用 DMAC 进行外设到外设传输时相同。本例不使用 DMA 和事件系统。ADC 配置与使用 DMAC 进行外设到外设传输时相同，不同之处在于 ADC 未配置为在出现事件输出（将 EVCTRL.STARTEI 设置为 0）时启动转换。

MCU 和外设初始化完成后，主循环中的软件触发启动第一次 ADC 转换。转换完成后，将调用 ADC 中断处理程序。在中断处理程序中，ADC 采样保存在缓冲区（即 SRAM 缓冲区 0）中，并且指示所完成 ADC 采样数量的计数变量递增。中断处理程序触发下一次 ADC 转换，一直持续到全部 1024 个采样都可用。计数达到 1024 后，将禁止 ADC 并停止进一步转换。随后从 SRAM 缓冲区 0 读取 ADC 结果并将其复制到另一个缓冲区（即 SRAM 缓冲区 1）。之后将 ADC 结果发送到 USART，传输完成时标志将置 1。现在，还会采用时间戳来计算 CPU 利用率。整个应用流程如下图所示。



4.3.2 CPU 利用率

本例使用相同的逻辑来计算 CPU 利用率，不同之处在于允许中断且不使用 DMA。通过递增计数器来计算转换次数。计数器达到 1024 后，ADC 结果将从 SRAM 缓冲区 0 复制到 SRAM 缓冲区 1，然后传输到 USART。采用时间戳并记下 `idle_loop_count` 来计算 CPU 利用率，如用于计算 CPU 利用率的逻辑实现中所示。

4.4 用于计算 CPU 利用率的逻辑实现

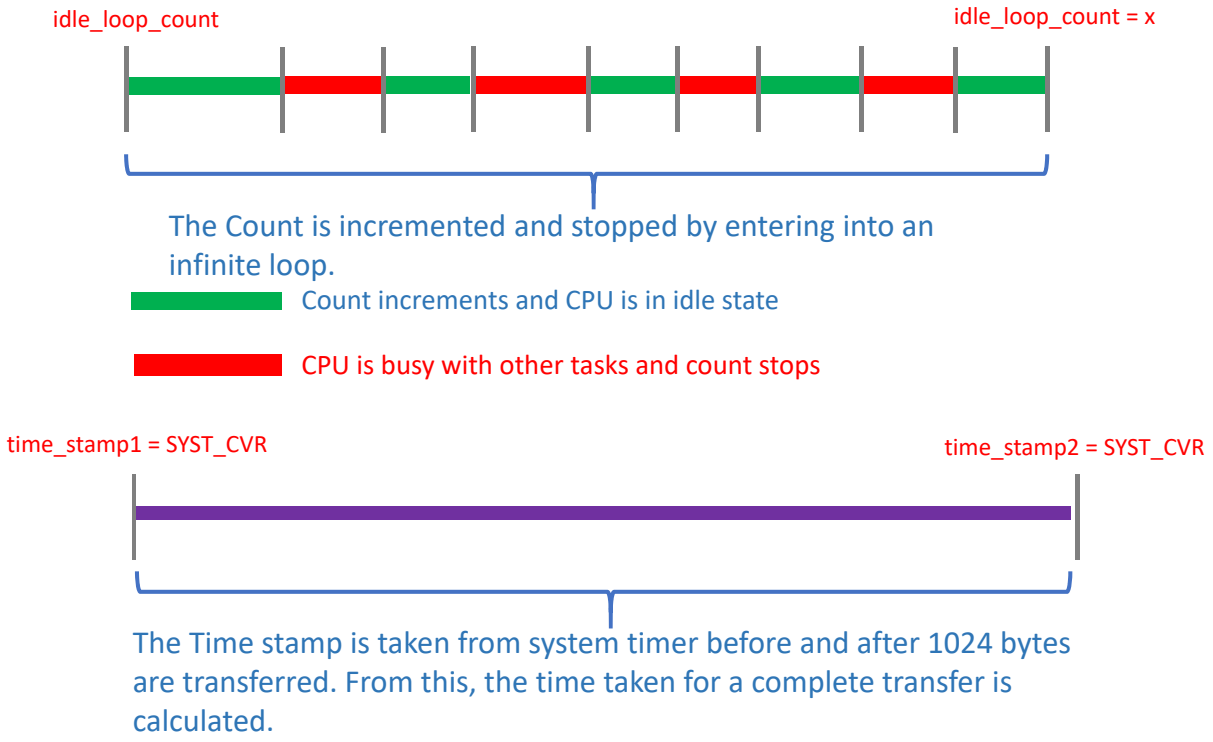
本节介绍本应用程序中计算 CPU 利用率所采用的逻辑。

要计算 CPU 利用率，请测量执行数据传输程序所需的总时间。该时间使用 SYSTICK 定时器测量。

测量执行数据传输程序时的 CPU 空闲时间。每当 CPU 空闲时，便通过递增变量（`idle_loop_count`）来测量空闲时间。通过将计数值乘以递增一次所花费的时间，将空闲计数器值转换为时间标度。

对于固定次数的数据传输，测量数据传输程序和空闲计数器所花费的总时间如下图所示。本测试中为一次传输 1024 字节。

图 4-4. CPU 利用率计算



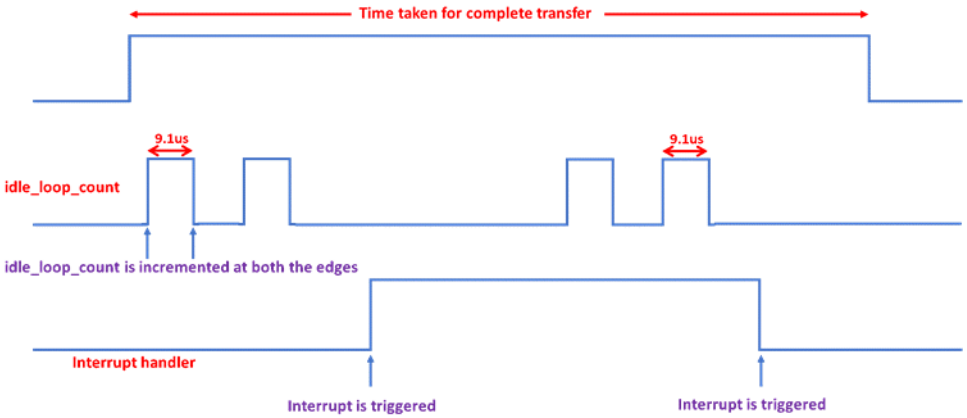
完成事务所花费的周期数（cycles_taken）可以使用 SYSTICK 通过所采用的时间戳计算得到。由于 SYSTICK 运行处理器时钟，因此可以根据所花费的周期计算全部事务所花费的时间，应用程序的 CPU 时钟频率如下所示。

完成事务所花费的时间 = (cycles_taken/CPU 时钟频率)

idle_loop_count 表示代码进入空闲任务的次数，可用于推导 CPU 在完成事务期间空闲的时间。要将此计数值转换为时间标度，应知道每次计数递增所花费的时间。

为此，在应用程序代码中，在空闲循环和中断处理程序中翻转两个单独的端口引脚。每当代码进入中断处理程序时，空闲循环计数停止，并且在空闲循环内翻转的引脚保持相同的电平。当代码退出处理程序时，处理程序内部翻转的引脚保持相同电平，空闲循环引脚启动翻转。在除去处理程序执行所花费的时间之后计算单次翻转所花费的时间。如下图所示。

图 4-5. 计算单次 idle_loop_count 递增所花费的时间

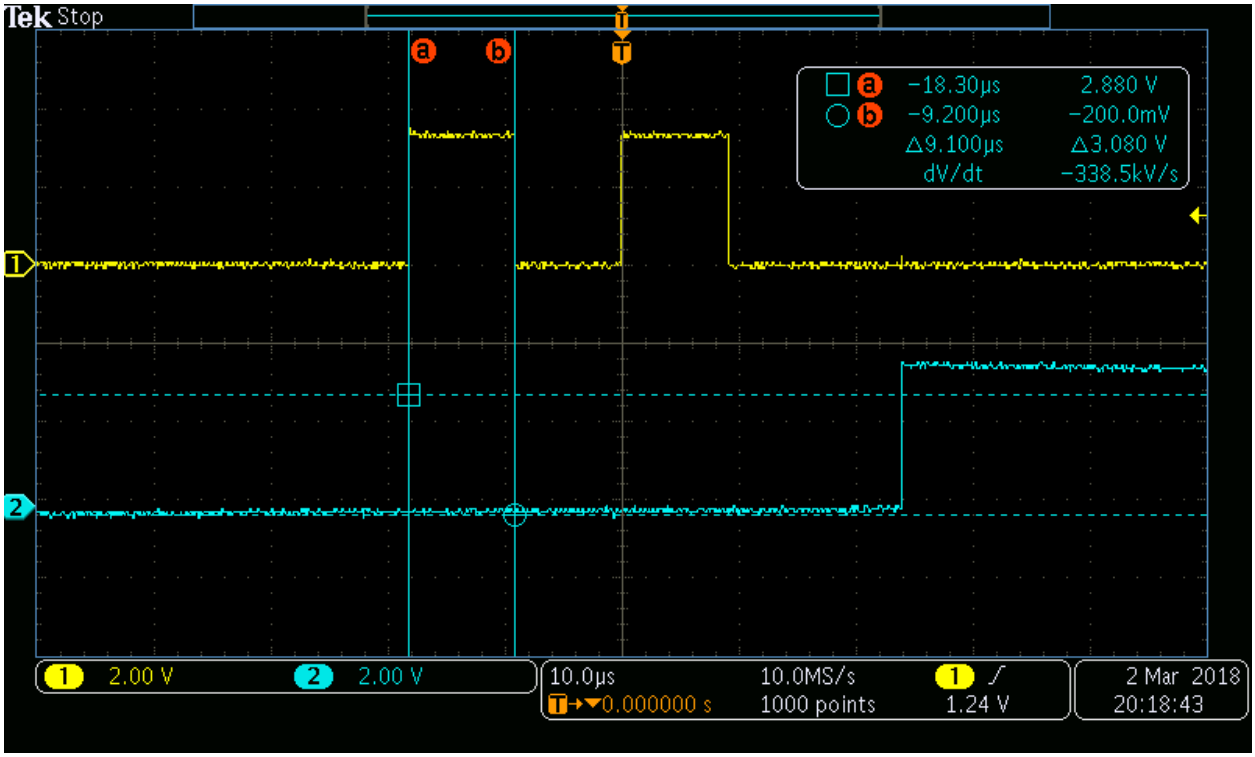


由示波器中的值计算可得出，每次计数递增所花费的时间为~9.1 µs，如下图所示。计数值与脉冲宽度（即 9.1 µs）相乘可得出 CPU 在空闲任务中花费的时间。

注：

1. `idle_loop_count` 脉冲的宽度是在没有触发中断时使一个空闲计数值递增所花费的时间。`idle_loop_count` 脉冲在 PA04 上输出，PA06 在中断处理程序中翻转。
2. CPU 空闲时间可通过将 `idle_loop_count` 乘以 `idle_loop_count` 脉冲宽度来计算。

图 4-6. 空闲任务的示波器截图



不同情形之间的 CPU 利用率分析给出了每种情形的 CPU 利用率分析。

注： 理想预期是使用 DMA 时的空闲循环计数应该比不使用 DMA 时更多。使用 DMA 时，CPU 不会中断，并且可以并行执行空闲任务。但在实践中，情况并非如此。原因是，DMA 将花费较少的时间来完成传输。在使用中断方法的情况下，完成事务需要更多时间。因此，对于 DMAC 情形而言，代码可以花费在理想任务上的时间会更少，`idle_loop_count` 值比不使用 DMA 时更小。为避免这种混淆，还使用系统定时器来获取完成传输所需的时间，并使用二者的比值计算 CPU 利用率。

5. 应用程序限制

5.1 USART 波特率和 ADC 采样频率

在 DMAC 用例中，ADC 结果的内容直接写入 USART 数据寄存器。数据写入 USART 后，DMA 立即触发下一次 ADC 转换。如果 USART 波特率低于 ADC 转换速率，则会导致终端窗口上的数据丢失。为避免这种情况，ADC 配置了尽可能最低的频率，而 USART 则配置了更高的波特率。

对于 ADC：

ADC 时钟的转换速率取决于 GCLK_ADC（即 16 MHz）及其预分频比，本例中为 64。

因此，ADC 时钟频率 = 16 MHz/64 = 250 kHz ~= 4 µs。

转换时间 = 8 个周期（8 位分辨率）+ 1 个周期（采样时间）= 9 * 4 µs ~= 36 µs

对于 USART：

在产品数据手册的*波特率公式*中，应满足 fbaud ≤ fref/S。

对于异步算术模式，每位的采样数为(S) = 16。

fref = 16 MHz

因此，最大可能的波特率 = 16 MHz/16 = 1000000。

配置的波特率 = 460800（即 1s 内发送 460800 位）。

对于 10 位，需要（10/460800）～=21.7 µs。

因此，建议将波特率设置为 460800，因为 ADC 采样每 36 µs 就绪一次。USART 会在 21.7µs 内发送先前的数据，并等待下一个 ADC 结果，而不会丢失任何数据。

注：

1. USART 数据帧的 10 位 = 起始位（1）+ 数据位（8）+ 停止位（1）。
2. 有关 ADC 和 USART 时序计算的详细信息，请参见器件数据手册。

5.2 SRAM 到 SRAM 传输类型

对于 ADC DMAC MEM MEM USART 和 ADC NO DMAC MEM MEM USART 示例，存储器到存储器传输类型（即将 ADC 结果从一个 SRAM 缓冲区复制到另一个 SRAM 缓冲区）用于演示目的。本应用程序不需要该类型也可正常工作。

6. 不同情形之间的 CPU 利用率分析

编程固件后，可以在终端窗口中看到结果。结果包含 ADC 结果数据、花费的周期数和“十六进制”格式的空闲循环计数。本章介绍如何从观察到的结果中推导出每种情形的 CPU 利用率。用于计算 CPU 利用率的逻辑实现给出了具体的计算过程。

注： 本文档中显示的结果在以下条件下得出。得出的 `idle_loop_count` 和 `cycles_taken` 将随优化、频率或应用程序代码而变化。

- 优化设置为零（-O0）
- 配置：调试
- 使能端口切换功能（`ENABLE_PORT_TOGGLE`）
- 使用 OSC16M 内部振荡器，频率选择为 16 MHz

6.1 CPU 频率计算

要确定所花费的时间，需要知道 CPU 频率。该应用程序以 16 MHz（OSC16M）运行。可以从产品数据手册的“电气特性”部分获得内部 RC 振荡器的精度。经计算得出，测试用电路板中的 RC 的精度为 15.98 MHz。这是通过将主时钟 GCLK0（以 16 MHz 运行）输出分配给 I/O 引脚来完成的。

注：

1. 这里使用的 I/O 引脚是 PA27。
2. 在所有三个示例项目中都使能了 GCLK0 输出。

6.2 根据观察到的结果计算 CPU 空闲时间

下图给出了应用程序中使用的各种情形的结果。

注： ADC 值以十六进制格式打印。因此，必须将终端仿真器程序配置为显示十六进制值。使用以下步骤将 Tera Term 配置为显示十六进制值：

1. 转到 Tera Term 的安装文件夹并搜索该文件

   ```
   TERATERM.INI
   ```

   .
2. 打开

   ```
   TERATERM.INI
   ```

   文件并将 Debug 变量设置为 ON。
3. 打开 Tera Term 并设置连接。将 Shift + Esc 键按下两次，将模式更改为十六进制格式。

图 6-1. ADC DMAC USART 终端输出

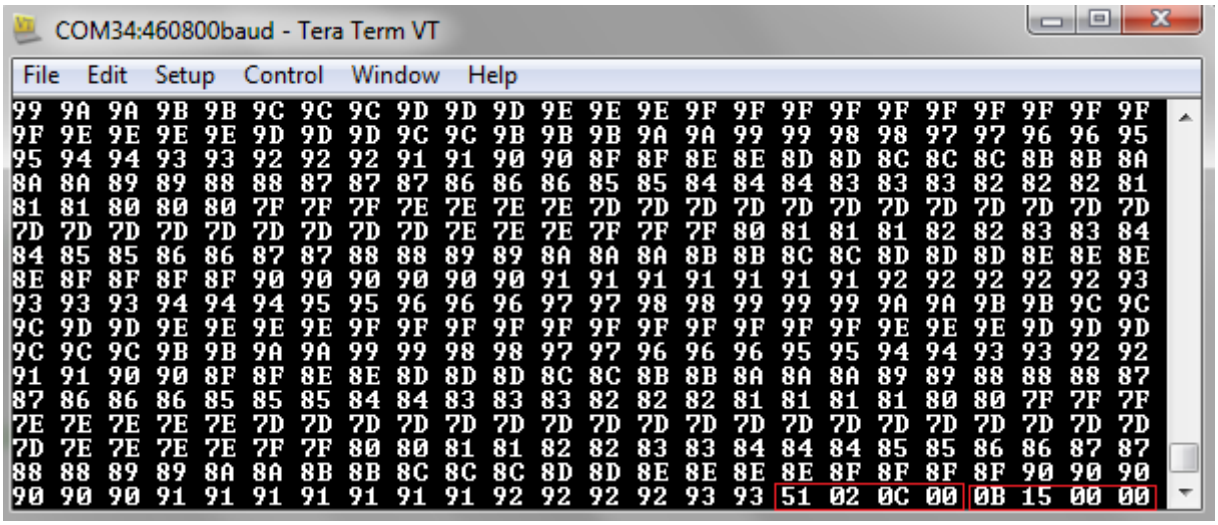


图 6-2. ADC DMAC MEM MEM USART 终端输出

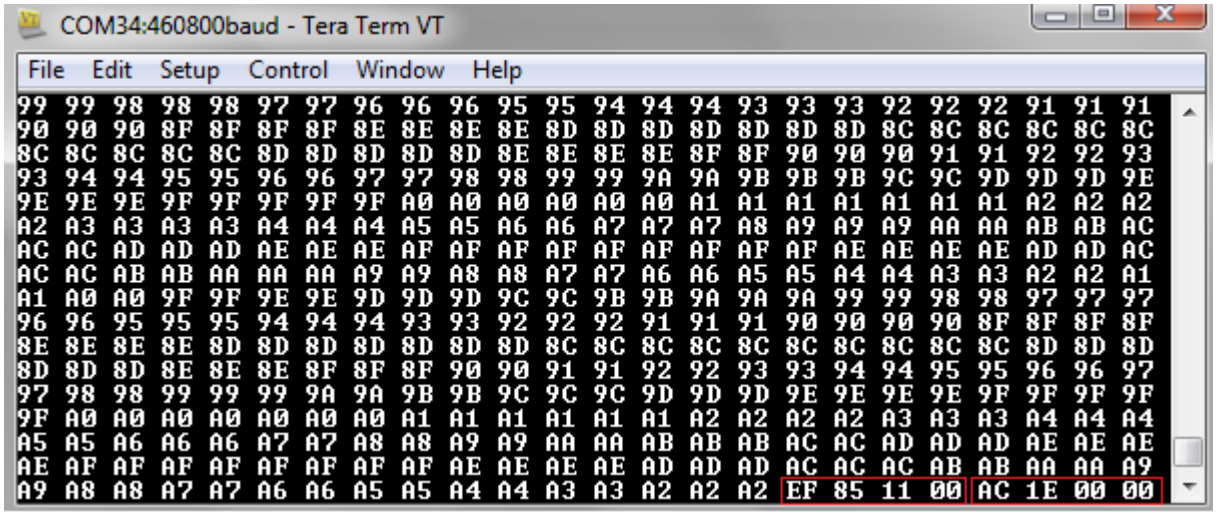
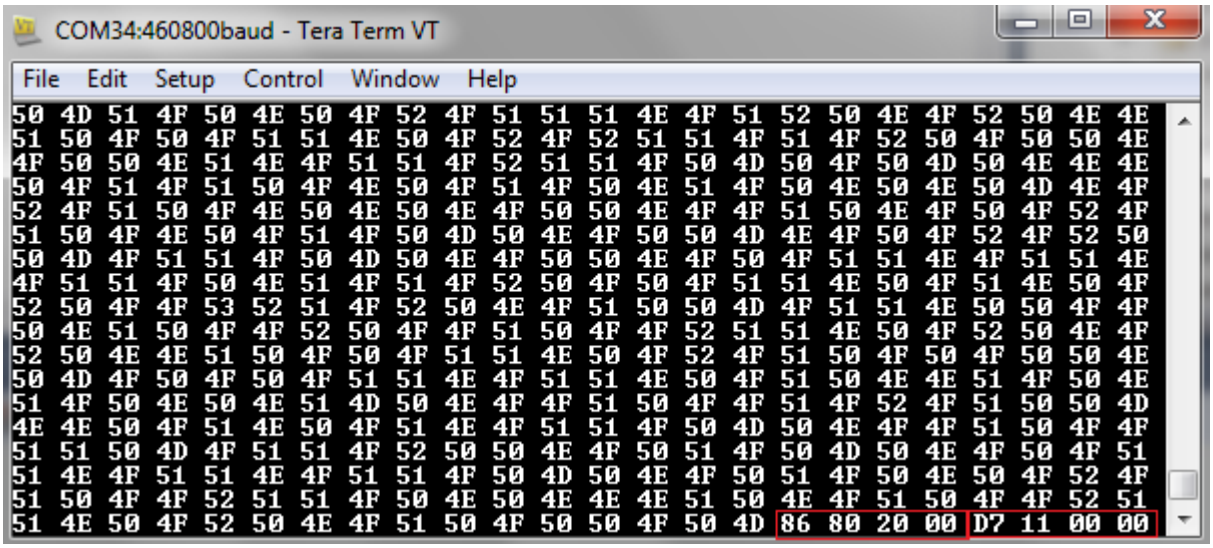


图 6-3. ADC NO DMAC MEM MEM USART 终端输出



ADC DMAC USART：

最后八个字节的数据代表大尾数格式的 `idle_loop_count` 和 `cycles_taken`，如下式所示。最后四个字节的结果是 `idle_loop_count`，接下来四个字节是 `cycles_taken`。

`idle_loop_count` = 0x0000150B = 5387d

`cycles_taken` = 0x000C0251 = 787025d

完成事务所花费的时间 = (cycles_taken/CPU 时钟频率)

完成事务所花费的时间 =(787025/15.98) µs = 49.250 ms。

经计算，每个空闲计数所花费的时间为 3.391 µs，如用于计算 CPU 利用率的逻辑实现中所述。

总 CPU 空闲时间 = `idle_loop_count` * 9.1 µs = 5387 * 9.1 µs = 49.021 ms

因此，在 49.250 ms 的传输周期内，`ADC_DMAC_USART` 情形的 CPU 空闲时间约为 49.021 ms。

同样，可以对其他情形进行计算，如下表所示。

表 6-1. CPU 利用率计算

情形	Idle_Loop_Count	Cycles_Taken	总传输时间（ms）	CPU 空闲时间（ms）	CPU 空闲时间（%）
ADC DMAC USART	0x0000150B	0x000C0251	49.250	49.021	99.53
ADC DMAC MEM MEM USART	0x00001EAC	0x001185EF	71.864	71.453	99.42
ADC NO DMAC MEM MEM USART	0x000011D7	0x00208086	133.294	41.559	31.178

使用 DMAC 时，CPU 在数据传输期间主要处于空闲状态。但是如果不使用 DMAC，CPU 只能在数据传输期间的一小段时间内处于空闲模式，整体传输时间也很长。

注： 该表用于提供可通过使用 DMA 改进性能的指示信息，而非用于比较不同的用例场景。

7. 应用程序执行

可从以下位置下载与本文档对应的示例应用程序：start.atmel.com。请按照以下步骤下载示例项目。

1. 转到 start.atmel.com，单击 *Browse Examples*（浏览示例）选项卡。
2. *搜索*所需项目：
   - `ADC DMAC USART`
   - `ADC DMAC MEM MEM USART`
   - `ADC NO DMAC MEM MEM USART`
3. 单击 **Open Selected Example**（打开所选示例），查看项目中的软件组件，或单击 **Download Selected Example**（下载所选示例），下载示例项目。

8. 参考资料

Arm Cortex®-M23 内核文档

- Cortex-M23 处理器技术参考手册版本 r1p0

器件数据手册

本数据手册包含外设框图和有关实现器件固件的详细信息，还包含器件的电气规范和预期特性。

可从 https://www.microchip.com 下载本数据手册。按照以下路径下载文档：*Products > Microcontrollers & Microprocessors > 32-bit MCUs > SAM 32-bit MCUs > SAM L MCUs*（产品 > 单片机和微处理器 > 32 位 MCU > SAM 32 位 MCU > SAM L MCU）。

硬件工具用户指南

- 可在以下位置获得 SAM L10 Xplained Pro 用户指南和原理图： http://www.microchip.com/DevelopmentTools/ProductDetails/dm320204
- 可在以下位置获得 SAM L11 Xplained Pro 用户指南和原理图：http://www.microchip.com/DevelopmentTools/ProductDetails/dm320205

Microchip 网站

Microchip 网站 http://www.microchip.com/为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的互联网浏览器即可访问，网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及归档软件
- **一般技术支持**——常见问题（FAQ）、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

变更通知客户服务

Microchip 的变更通知客户服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 http://www.microchip.com/。在“支持”（Support）下，点击“变更通知客户”（Customer Change Notification）服务后按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师（FAE）
- 技术支持

客户应联系其代理商、代表或应用工程师（FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过以下网站获得技术支持： http://www.microchip.com/support

Microchip 器件代码保护功能

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下，Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿意与关心代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是“牢不可破”的。

代码保护功能处于持续发展中。Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了《数字器件千年版权法案（Digital Millennium Copyright Act）》。如

果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

法律声明

本出版物中所述的器件应用信息及其他类似内容仅为您提供便利，它们可能由更新之信息所替代。确保应用符合技术规范，是您自身应负的责任。Microchip 对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对其使用情况、质量、性能、适销性或特定用途的适用性的声明或担保。Microchip 对因这些信息及使用这些信息而引起的后果不承担任何责任。如果将 Microchip 器件用于生命维持和/或生命安全应用，一切风险由买方自负。买方同意在由此引发任何一切伤害、索赔、诉讼或费用时，会维护和保障 Microchip 免于承担法律责任，并加以赔偿。除非另外声明，否则在 Microchip 知识产权保护下，不得暗中或以其他方式转让任何许可证。

商标

Microchip 的名称和徽标组合、Microchip 徽标、AnyRate、AVR、AVR 徽标、AVR Freaks、BitCloud、chipKIT、chipKIT 徽标、CryptoMemory、CryptoRF、dsPIC、FlashFlex、flexPWR、Heldo、JukeBlox、KeeLoq、Kleer、LANCheck、LINK MD、maXStylus、maXTouch、MediaLB、megaAVR、MOST、MOST 徽标、MPLAB、OptoLyzer、PIC、picoPower、PICSTART、PIC32 徽标、Prochip Designer、QTouch、SAM-BA、SpyNIC、SST、SST 徽标、SuperFlash、tinyAVR、UNI/O 和 XMEGA 是 Microchip Technology Incorporated 在美国和其他国家或地区的注册商标。

ClockWorks、The Embedded Control Solutions Company、EtherSynch、Hyper Speed Control、HyperLight Load、IntelliMOS、mTouch、Precision Edge 和 Quiet-Wire 为 Microchip Technology Incorporated 在美国的注册商标。

Adjacent Key Suppression、AKS、Analog-for-the-Digital Age、Any Capacitor、AnyIn、AnyOut、BodyCom、CodeGuard、CryptoAuthentication、CryptoAutomotive、CryptoCompanion、CryptoController、dsPICDEM、dsPICDEM.net、Dynamic Average Matching、DAM、ECAN、EtherGREEN、In-Circuit Serial Programming、ICSP、INICnet、Inter-Chip Connectivity、JitterBlocker、KleerNet、KleerNet 徽标、memBrain、Mindi、MiWi、motorBench、MPASM、MPF、MPLAB Certified 徽标、MPLIB、MPLINK、MultiTRAK、NetDetach、Omniscient Code Generation、PICDEM、PICDEM.net、PICkit、PICtail、PowerSmart、PureSilicon、QMatrix、REAL ICE、Ripple Blocker、SAM-ICE、Serial Quad I/O、SMART-I.S.、SQI、SuperSwitcher、SuperSwitcher II、Total Endurance、TSHARC、USBCheck、VariSense、ViewSpan、WiperLock、Wireless DNA 和 ZENA 为 Microchip Technology Incorporated 在美国和其他国家或地区的商标。

SQTP 为 Microchip Technology Inc.在美国的服务标记。

Silicon Storage Technology 为 Microchip Technology Inc.在除美国外的国家或地区的注册商标。

GestIC 是 Microchip Technology Inc.的子公司 Microchip Technology Germany II GmbH & Co. KG 在除美国外的国家或地区的注册商标。

在此提及的所有其他商标均为各持有公司所有。

ISBN： 978-1-5224-4014-7

DNV 认证的质量管理体系

ISO/TS 16949

Microchip 位于美国亚利桑那州 Chandler 和 Tempe 与位于俄勒冈州 Gresham 的全球总部、设计和晶圆生产厂及位于美国加利福尼亚州和印度的设计中心均通过了 ISO/TS-16949:2009 认证。Microchip 的 PIC® MCU 和 dsPIC® DSC、KEELOQ®跳码器件、串行 EEPROM、单片机外设、非易失性存储器及模拟产品严格遵守公司的质量体系流程。此外，Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。

全球销售及服务网点

美洲	亚太地区	亚太地区	欧洲
公司总部 2355 West Chandler Blvd. Chandler, AZ 85224-6199 电话：1-480-792-7200 传真：1-480-792-7277 技术支持： http://www.microchip.com/support 网址： www.microchip.com **亚特兰大** 德卢斯，乔治亚州 电话：1-678-957-9614 传真：1-678-957-1455 **奥斯汀，德克萨斯州** 电话：1-512-257-3370 **波士顿** 韦斯特伯鲁，马萨诸塞州 电话：1-774-760-0087 传真：1-774-760-0088 **芝加哥** 艾塔斯卡，伊利诺伊州 电话：1-630-285-0071 传真：1-630-285-0075 **达拉斯** 艾迪生，德克萨斯州 电话：1-972-818-7423 传真：1-972-818-2924 **底特律** 诺维，密歇根州 电话：1-248-848-4000 **休斯敦，德克萨斯州** 电话：1-281-894-5983 **印第安纳波利斯** 诺布尔斯维尔，印第安纳州 电话：1-317-773-8323 传真：1-317-773-5453 电话：1-317-536-2380 **洛杉矶** 米申维耶霍，加利福尼亚州 电话：1-949-462-9523 传真：1-949-462-9608 电话：1-951-273-7800 **罗利，北卡罗来纳州** 电话：1-919-844-7510 **纽约，纽约州** 电话：1-631-435-6000 **圣何塞，加利福尼亚州** 电话：1-408-735-9110 电话：1-408-436-4270 **加拿大 - 多伦多** 电话：1-905-695-1980 传真：1-905-695-2078	**中国 - 北京** 电话：86-10-8569-7000 **中国 - 成都** 电话：86-28-8665-5511 **中国 - 重庆** 电话：86-23-8980-9588 **中国 - 东莞** 电话：86-769-8702-9880 **中国 - 广州** 电话：86-20-8755-8029 **中国 - 杭州** 电话：86-571-8792-8115 **中国 - 南京** 电话：86-25-8473-2460 **中国 - 青岛** 电话：86-532-8502-7355 **中国 - 上海** 电话：86-21-3326-8000 **中国 - 沈阳** 电话：86-24-2334-2829 **中国 - 深圳** 电话：86-755-8864-2200 **中国 - 苏州** 电话：86-186-6233-1526 **中国 - 武汉** 电话：86-27-5980-5300 **中国 - 西安** 电话：86-29-8833-7252 **中国 - 厦门** 电话：86-592-2388138 **中国 - 香港特别行政区** 电话：852-2943-5100 **中国 - 珠海** 电话：86-756-3210040 **台湾地区 - 高雄** 电话：886-7-213-7830 **台湾地区 - 台北** 电话：886-2-2508-8600 **台湾地区 - 新竹** 电话：886-3-577-8366	**澳大利亚 - 悉尼** 电话：61-2-9868-6733 **印度 - 班加罗尔** 电话：91-80-3090-4444 **印度 - 新德里** 电话：91-11-4160-8631 **印度 - 浦那** 电话：91-20-4121-0141 **日本 - 大阪** 电话：81-6-6152-7160 **日本 - 东京** 电话：81-3-6880- 3770 **韩国 - 大邱** 电话：82-53-744-4301 **韩国 - 首尔** 电话：82-2-554-7200 **马来西亚 - 吉隆坡** 电话：60-3-7651-7906 **马来西亚 - 槟榔屿** 电话：60-4-227-8870 **菲律宾 - 马尼拉** 电话：63-2-634-9065 **新加坡** 电话：65-6334-8870 **泰国 - 曼谷** 电话：66-2-694-1351 **越南 - 胡志明市** 电话：84-28-5448-2100	**奥地利 - 韦尔斯** 电话：43-7242-2244-39 传真：43-7242-2244-393 **丹麦 - 哥本哈根** 电话：45-4450-2828 传真：45-4485-2829 **芬兰 - 埃斯波** 电话：358-9-4520-820 **法国 - 巴黎** 电话：33-1-69-53-63-20 传真：33-1-69-30-90-79 **德国 - 加兴** 电话：49-8931-9700 **德国 - 哈恩** 电话：49-2129-3766400 **德国 - 海尔布隆** 电话：49-7131-67-3636 **德国 - 卡尔斯鲁厄** 电话：49-721-625370 **德国 - 慕尼黑** 电话：49-89-627-144-0 传真：49-89-627-144-44 **德国 - 罗森海姆** 电话：49-8031-354-560 **以色列 - 赖阿南纳** 电话：972-9-744-7705 **意大利 - 米兰** 电话：39-0331-742611 传真：39-0331-466781 **意大利 - 帕多瓦** 电话：39-049-7625286 **荷兰 - 德卢内市** 电话：31-416-690399 传真：31-416-690340 **挪威 - 特隆赫姆** 电话：47-7288-4388 **波兰 - 华沙** 电话：48-22-3325737 **罗马尼亚 - 布加勒斯特** 电话：40-21-407-87-50 **西班牙 - 马德里** 电话：34-91-708-08-90 传真：34-91-708-08-91 **瑞典 - 哥德堡** 电话：46-31-704-60-40 **瑞典 - 斯德哥尔摩** 电话：46-8-5090-4654 **英国 - 沃金厄姆** 电话：44-118-921-5800 传真：44-118-921-5820