Freescale Semiconductor
应用笔记

Document Number: AN4745
Rev 1, 06/2014

优化 Kinetis K 系列 MCU 的性能

作者: Melissa Hunter

内容

1 简介

在嵌入式系统中，资源通常有限，因而如何利用这些资源获得最佳性能显得至关重要。虽然高性能和低功耗看起来是相互对立的概念，但是若能快速地执行任务然后进入低功耗模式，则可从整体上降低系统的功耗。因此，几乎任何系统都可以从改善性能的工作中受益。

提高嵌入式系统的性能可能是一项复杂的任务。架构和系统功能的内部运行方式往往存在一些细微差别，而这些差别会对系统产生影响。此外，每一个系统可能有不同的性能目标。例如，有些系统可能仅侧重于 CPU 性能，而有的系统则可能需要优化某个通信端口（如以太网或 USB）的吞吐量。

本应用笔记将阐述在 Kinetis K 系列器件上发现的可能影响系统性能的功能特性。本文档并非关于如何优化应用的进阶指南，因为没有任何一套硬性规则可适用于所有情况。本文的主要目的是为了阐释某些能够优化应用的主要架构和系统模块特性，便于设计者在设计系统的软件和硬件时能在了解情况的前提下做出决策。

2 Kinetis K 系列架构概述

系统架构是系统整体性能的最大要素之一。不同数据块之间如何结合也会影响某些模块级的功能。因此，若要了解如何优化系统性能，首先需从一个较高的层面来了解架构。

下图显示了 Kinetis K70 系列器件的简化结构框图。选择此系列是因为这个系列涵盖了下面将讨论的所有性能特点。其他 Kinetis 系列器件不一定具备所有这些特点，但整体架构通常大致相同。

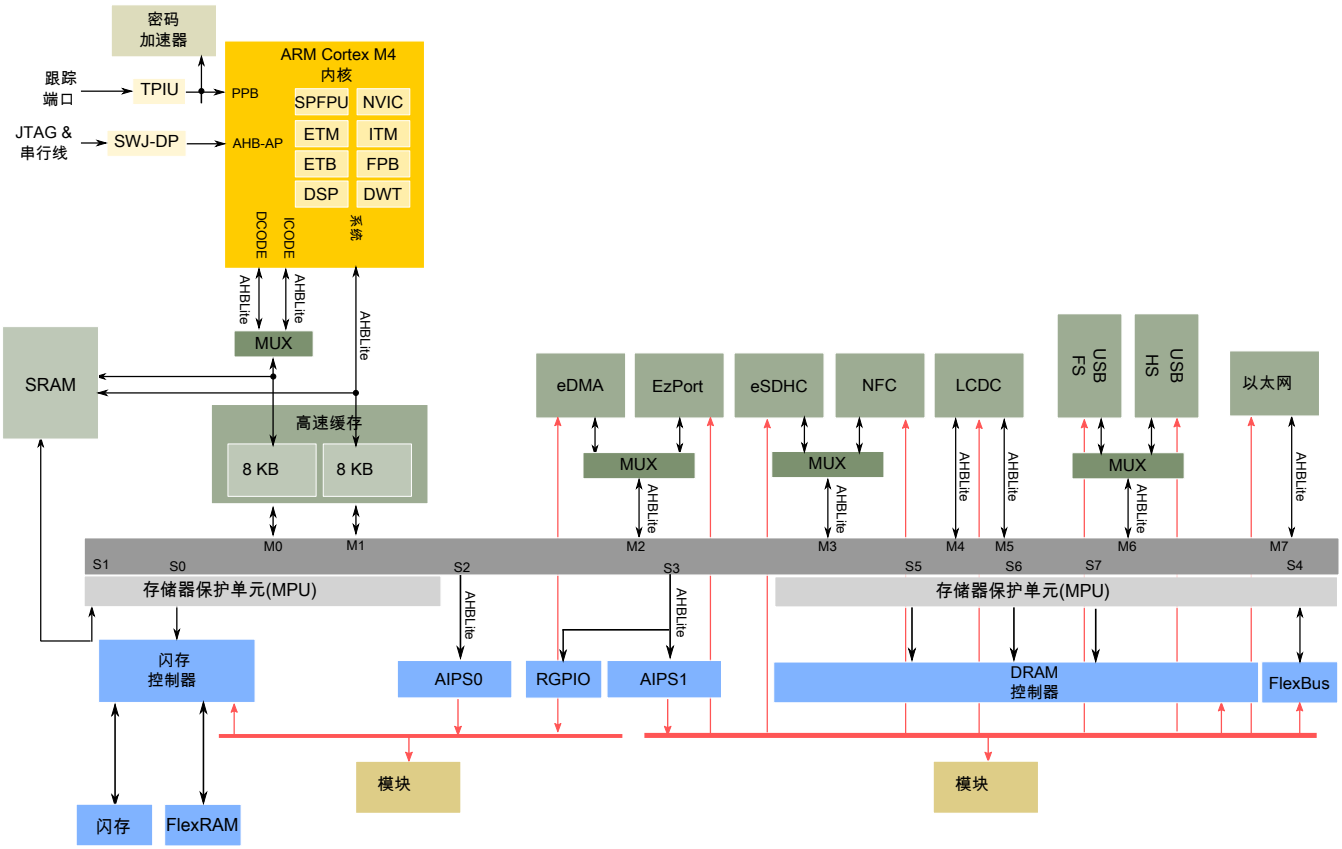

图 1. K70 结构框图

2.1 Kinetis 的内核总线

ARM Cortex M4 内核采用哈佛架构，带有几种内存映射总线:

- ICODE- ICODE 总线用来访问存储在 0x0000_0000-0x1FFF_FFFF 地址段的任何指令。
- DCODE- DCODE 总线用来访问存储在 0x0000_0000-0x1FFF_FFFF 地址段的任何数据。
- 系统 - 系统总线用来访问 0x2000_0000-0xDFFF_FFFF 和 0xE010_0000-0xFFFF_FFFF 之间的任何地址。
- 专用外设总线 - 专用外设总线(PPB)映射至 0xE004_0000-0xE00FFFFF 地址段。

在 Kinetis 器件上，ICODE 和 DCODE 总线进行多路复用，合为一个 CODE 总线。0x2000_0000 以下的任何内核访问将通过 CODE 总线进行，而 0x2000_0000 或以上的地址大多通过系统总线进行访问（PPB 访问例外）。

从应用方面来说，CODE 和系统总线并无多大区别，但是这两种总线的执行方式有点差别。CODE 总线周期在内核无附加延时。系统总线周期的时序取决于访问类型。系统总线的数据访问在内核无附加延时，而指令访问则会在内核增加了一个等待状态。

2.2 Kinetis K 系列存储器映像

为使应用能尽可能地使用 CODE 总线，Kinetis 系统的存储器映像汇总了地址在 0x2000_0000 以下的主要存储器区域。

下表中显示了 CODE 总线的存储器映像区。这些存储器区域涵盖了用于 DRAM 控制器和 FlexBus 的别名区。一般情况下，DRAM 和 FlexBus 区位于存储器映像中的系统总线部分。加上别名区后，存储器也可供 CODE 总线使用。这样的话，从外部存储器执行代码时可以发挥最大性能。任何情况下，应尽可能使用 CODE 总线区来存储代码。

表 1. CODE 总线存储器映像

系统 32 位地址范围	目标从机	访问
0x0000_0000-0x07FF_FFFF	程序闪存和只读数据	所有主机
0x0800_0000-0x0FFF_FFFF	DRAM 控制器（别名区）（可选）	仅限 Cortex-M4 内核
0x1000_0000-0x13FF_FFFF	FlexNVM（可选）	所有主机
0x1400_0000-0x17FF_FFFF	FlexRAM/可编程加速 RAM	所有主机
0x1800_0000-0x1BFF_FFFF	FlexBus（别名区）（可选）	仅限 Cortex-M4 内核
0x1C00_0000-0x1FFF_FFFF	SRAM_L：低位 SRAM	所有主机

3 Kinetis SRAM

所有 Kinetis K 系列器件都含有两个片上 SRAM 块。第一个块(SRAM_L)映射于 CODE 总线，第二个块(SRAM_U)则映射于系统总线。访问存储器本身仅需一个周期，但由于指令访问系统总线时在内核延时一个时钟周期，所以 SRAM_U 的指令访问至少需要两个时钟周期。

SRAM_L 是唯一的能存放代码也能存放数据的存储器，并且始终保证内核仅需一个周期。因此，建议尽可能地多使用 SRAM_L 块。这是存储关键代码的理想区域。

3.1 SRAM 访问

除了内核端口可通过 CODE 总线和系统总线访问，片上的非内核主机也可通过后门端口访问 SRAM。后门端口是交叉开关(XBS)上的一个从端口。有三个端口通向 SRAM 控制器（CODE 总线、系统总线和后门端口），映射到对其中一个 SRAM 块的访问。双 SRAM 块结构使 SRAM 控制器可处理对不同 SRAM 块的同时访问。允许对 SRAM 的同时访问包括:

- 内核 CODE 总线(SRAM_L)和内核系统总线(SRAM_U)的访问
- 内核 CODE 总线(SRAM_L)和非内核主机对 SRAM_U 的访问
- 内核系统总线(SRAM_U)和非内核主机对 SRAM_L 的访问

对每个 SRAM 块中的代码和数据进行策略性布局有助于提高并行性和整体性能。对于典型的应用，将关键代码置于 SRAM_L 块中，数据和堆栈置于 SRAM_U 块中将产生最佳的性能。

3.2 SRAM 仲裁

由于 SRAM 控制器处理的访问端口比 SRAM 块要多，因此 SRAM 控制器具有内部的仲裁逻辑。仲裁过程通过 MCM_CR 中的字段进行控制，给每个 SRAM 块预留了可编程的仲裁模式。

可用的 SRAM 仲裁模式有:

- 固定 CPU 优先权 - 在这种模式下, CPU 访问对应的 SRAM 块时总是具有最高优先级。如果要求单周期访问(或两周期访问 SRAM_U)，建议使用这种模式。
- 固定后门优先权 - 在这种模式下，后门访问对应的 SRAM 块时总是具有最高优先级。为非内核主机扩大带宽时可采用这种模式；但在某些情况下，特殊循环模式可能是最佳方式。更多信息，请参见关于特殊循环模式的描述。

- 循环模式 (Round Robin) - 在这种模式下，优先权在内核与后门端口之间转换，力求在两种端口之间平均分配优先权。这种模式建议在需要平衡 CPU 性能和非内核主机带宽的情况下采用。两个 SRAM 块均默认配置为循环模式仲裁方式。
- 特殊循环模式 - 在这种模式下，优先权在内核与后门端口之间转换，但是算法倾向于后门端口。在许多情况下，这种模式是扩大非内核主机总吞吐量的一种理想设置。例如，如果你正在运行以太网堆栈，缓冲区存储在一个 SRAM 块中，ENET 模块将需要访问 SRAM，但 CPU 还需要处理 ENET 模块正在发送或接收的数据包。将优先权赋给 ENET 看起来可能是最佳设置，但由于 CPU 没有机会处理数据包，因此可能会导致 ENET 必须等待。对于这种情况，最好是先对特殊循环模式和固定后门优先权模式进行试验，以确定哪个设置可以提供最佳的整体性能。

4 系统缓存

某些 Kinetis 器件中含有可显著提高性能的系统缓存，这种优势在从外部存储器执行代码时尤为明显。目前仅 120/150 MHz Kinetis 器件上可以使用系统缓存。

4.1 高速缓存的组织结构和功能

系统缓存实际上包含两个独立的 8 KB 高速缓存块。第一个 8 KB 缓存用于 CODE 总线访问，第二个 8 KB 缓存用于系统总线访问。

缓存控制器的特性包括:

- 两个独立的 8 KB 缓存阵列（缓存总容量为 16 KB）
- 两路组相联高速缓存结构
- 16 字节缓存线
- 支持回写（回拷贝）、透写和不可缓存模式
- 16 个带有独立缓存模式的缓存区

4.2 高速缓存区和缓存模式配置

高速缓存的设置通过使用预定义的地址区域进行配置。高速缓存控制器支持 16 个区域，但 Kinetis K 系列器件目前仅使用十个区域。地址区域定义了将使用的地址范围，以及该区域默认采用的缓存配置。

一个区域的缓存模式只能采用低于初始值的设置:

```
write-back > write-through > non-cacheable
```

这意味着区域的默认缓存配置决定了可使用何种缓存模式。如果一个区域被默认定义为不可缓存，则将无法使用两种可缓存设置。

下表中显示了 Kinetis 器件上使用的高速缓存区，以及默认缓存模式和可用缓存模式。

表 2. Kinetis K 系列的高速缓存区

区号	地址范围	目标从机	默认缓存模式	可用缓存模式
R0	0x0000_0000 – 0x07FF_FFFF	程序闪存和只读数据	透写	透写和不可缓存
R1	0x0800_0000 – 0x0FFF_FFFF	DRAM 控制器（别名区）	透写	透写和不可缓存

下一页继续介绍此表...

表 2. Kinetis K 系列的高速缓存区 (继续)

区号	地址范围	目标从机	默认缓存模式	可用缓存模式
R2	0x1000_0000 – 0x17FF_FFFF	FlexNVM	透写	透写和不可缓存
R3	0x1800_0000 – 0x1BFF_FFFF	FlexBus（别名区）	透写	透写和不可缓存
R4	0x1C00_0000 – 0x1FFF_FFFF	SRAM_L：低位 SRAM (ICODE/ DCODE)	不可缓存	不可缓存
R5	0x2000_0000 – 0x200F_FFFF	SRAM_U：高位 SRAM	不可缓存	不可缓存
R6	0x6000_0000 – 0x6FFF_FFFF	Flexbus（外部存储器 - 回写 ）	回写	回写、透写和不可缓存
R7	0x7000_0000 – 0x7FFF_FFFF	DRAM 控制器	回写	回写、透写和不可缓存
R8	0x8000_0000 – 0x8FFF_FFFF	DRAM 控制器 - 透写	透写	透写和不可缓存
R9	0x9000_0000 – 0x9FFF_FFFF	FlexBus（外部存储器 - 透写）	透写	透写和不可缓存

闪存区的默认选项为透写模式，这是可采用的最低可缓存模式。对闪存的写操作不使用内存映射的写操作总线周期，因此即使闪存区可透写，写操作也不会实际改变闪存的内容，校正码不应尝试通过 CODE 总线对闪存进行写访问。

SRAM 区不可缓存。这是因为高速缓存不会提高 SRAM 存取的速度。对 SRAM 数据块的读操作与对高速缓存的读操作耗时相同（假定缓存命中）。对 SRAM（SRAM_L 和 CODE 缓存命中均需要一个时钟周期，而 SRAM_U 和系统总线缓存命中均需要两个时钟周期）进行高速缓存毫无优势，因此总是默认不可缓存模式。

4.3 缓存初始化

系统缓存在复位时禁用。以下是用于初始化缓存的建议步骤:

1. 在 LMEM_PCCRMR 中根据需要修改缓存区配置的默认值
2. 将 LMEM_PCCCR[INVW1 和 INVW0]置 1，使控制器配置为 CODE 总线缓存的两路无效。
3. 将 LMEM_PCCCR[GO]置 1 以启动无效。
4. 等待 LMEM_PCCCR[GO]位清零，清零表示已完成命令。
5. 将 LMEM_PCCR[ENCACHE]置 1，以启用 CODE 总线缓存。
6. 将 LMEM_PSCCR[INVW1 和 INVW0]置 1，使控制器配置为系统总线缓存的两路均无效。
7. 将 LMEM_PCCCR[GO]置 1 以启动无效。
8. 等待 LMEM_PSCCR[GO]位清零，清零表示已完成命令。
9. 将 LMEM_PSCR[ENCACHE]置 1，以启用系统总线缓存。

4.4 缓存一致性

缓存仅用于内核发起的访问。因为不具有监听能力，因此无法缓存非内核主机的访问。如果非内核主机可以访问高速缓冲存储器，则需将缓存一致性作为整体系统设计的组成部分来考虑。

应考虑缓存一致性的常见案例如下:

表 3. 缓存一致性建议

案例	建议措施
固件将用于对闪存空间进行擦除和编程（这将影响系统缓存、FMC 缓存和预取缓冲器）	使总线缓存闪存上的缓存线无效，同时使待擦除/编程区域对应闪存组的 FMC 缓存路径和预取缓冲器无效。
非内核主机（DMA、USB、ENET、SDHC 或 NFC）将读/写一个存储器区	• 如果在一个已知时帧内，非内核主机将访问存储器，而内核不会对相同区域或该区域附近进行任何访问，那么，在非内核主机开始读/写存储器之前，可使相关的缓存线清零。 • 如果在一个已知时帧内，非内核主机将访问存储器，而内核将需要对相同区域或该区域附近进行访问，那么，在非内核主机开始读/写存储器之前，可使相关缓存线清零然后禁用缓存。因为缓存区域的设置无法从“不可缓存”回退至“已缓存”模式，整个缓存（无论 CODE 总线或系统总线）将必须暂时禁用。 • 如果非内核主机将访问存储器的时间并不确定，则存储器区域应配置为不可缓存模式。这应该是一种罕见的情况。当内核和非内核主机共享数据时，通常存在相关的握手层允许使用前两种方式。例如，内核为 ENET 接收数据配置一个缓冲器。内核将为缓冲器位置清空缓存线，然后告知 ENET 缓冲已经准备好接收。当缓冲器已接收到数据并且准备好被读取时，ENET 将向内核发出通知（通过中断或轮询状态位）。在此情况下，内核可读入缓冲器的内容，同时将内容载入缓存。
内核正在修改将被非内核主机读取的数据(LCDC)	• 如果采用回写模式，缓存线必须先清零才可用于非内核主机的读取。 • 存储器区域使用透写模式。因为非内核主机仅读取数据，所以内核读取的缓存内容永不过期。透写模式还意味着非内核主机将可访问最新数据。

5 闪存控制器(FMC)

对于很多系统来说，主存储器就是片上闪存。闪存控制器(FMC)是闪存块和系统的接口。在典型配置中，内核总线和系统总线的时钟速度比闪存时钟的速度要快得多。FMC 具有的一些特性可用来加速闪存访问 。

5.1 FMC 特性

FMC 有两个关键特征可促使闪存访问在单个时钟周期内完成:

- FMC 缓存 - FMC 内有一个小缓存可用来存放近期访问的闪存信息。对于不同的器件，FMC 缓存的正确配置可能不同，但所有器件上都有 FMC 缓存。注意：某些 Kinetis 器件还含有系统缓存，系统缓存与 FMC 缓存是完全独立的。虽然两个缓存独立运行，但可以一起使用，以促进闪存读取的加速。
- 预取推理缓冲器 - 由于存储器访问通常是连续的，因此，当 FMC 接收到已知闪存位置的请求时，将预取下一个闪存数据块。预取信息被存放在推理缓冲器中，直到接收到访问其他数据块的请求为止。

在许多情况下，FMC 缓存和预取推理缓冲器可使 FMC 响应闪存访问时无附加等待状态。只要请求的信息在缓存和预取缓冲器中可用，FMC 的响应无附加等待状态。

5.2 FMC 配置

默认情况下，FMC 的缓存和预取缓冲器均为使能状态。大多数应用无需重新配置 FMC 即可达到最佳性能。

但也有一些可编程选项可供更改:

- 指令与数据缓存 - 默认情况下，指令和数据的访问均为缓存方式。可更改此选项，使整个 FMC 缓存仅用于指令或数据。也可通过关闭指令和数据缓存来禁用整个 FMC 缓存，但不建议在尝试提高性能时采用这种设置。
- 指令与数据预取 - 默认情况下，指令和数据的访问均能触发一个推理预取周期。可更改此选项，使推理预取仅由指令访问或数据访问发起。如果随机数据访问和基本连续的指令访问均指向同一闪存组，则可能需要仅允许指令预取。
- 缓存锁定 - FMC 缓存中的四路均可锁定，以强制缓存保持某些特定值。由于 FMC 缓存的容量较小，与锁定 FMC 缓存相比，将关键代码或数据转移至一个 SRAM 块（首选 SRAM_L）通常是更佳的方案。若采用这种方法，关键信息取用时无需等待，并且还可将整个 FMC 缓存用于加速闪存访问。
- 缓存替换控制 - 若指令和数据都采用相同的处理方式，可修改 FMC 缓存替换算法的默认设置， 使 0-1 路或 0-2 路为指令专用，其他通路为数据专用。

注

访问闪存时不应修改 FMC 寄存器。Freescale 建议从片上 SRAM 执行任何修改 FMC 设置的代码。

6 交叉开关(AXBS)

交叉开关是微控制器的主要总线互连结构。交叉开关负责处理总线主控器和从端口之间的连接，当多个主控器同时尝试访问同一从端口时，还负责处理主控器之间的仲裁。

注

50 MHz 的 Kinetis K 系列器件不支持下面小节中描述的特性。这些器件采用精简版交叉开关(AXBS-Lite)，具有简化的特性集。AXBS-Lite 的配置是固定的，不能根据给定系统的需求调整操作 。

6.1 AXBS 访问

AXBS 具有一个极重要的特性，即允许不同主机对不同从端口的同时访问。合理计划系统内主机对存储器的利用情况可以显著提高整体系统的性能。

例如，系统存储器配置可能为:

- 内核 - 闪存中的指令和 SRAM_L 中的内核数据和堆栈
- USB - SRAM_U 中的数据缓冲区
- LCD 控制器 - DDR 中的图形缓冲区

这样的存储器配置允许三个主机运行各自所需的总线周期，主机之间的相互干扰极小。有些情况下内核可能需要访问 USB 缓冲区和更新图形缓冲区，除这些访问之外主机可并行运行。

6.2 AXBS 仲裁

当两个或两个以上的主机试图访问同一从端口时，AXBS 将通过仲裁算法来决定哪个主机可以最先访问该端口。可通过可编程字段控制每个从端口的仲裁设置。

有两种不同的 AXBS 仲裁方案可供选择:

- 固定优先权 - 通过 AXBS_PRSn 寄存器决定主机访问相关从端口的优先级。如果某个主机在访问已知从端口时总是需要最高的优先级，则可使用这种设置。
- 循环模式 - 在此模式下，每个主机的优先级基于该端口与上一次访问从端口的主机端口之间的距离。例如，如果上次访问从端口的是主机 2，则在下一周期中主机 3 将具有最高优先级，而主机 2 的优先级最低。

应注意，仲裁仅发生在对同一从端口有多个访问请求的情况下。如果优先级较低的主机对空闲的从端口发出访问请求，该主机将可以启动总线周期。如果有一个优先级较高的主机在低优先级主机开始其总线周期之后发出请求，高优先级主机必须在低优先级主机的总线周期到达转换边界后才能介入。若为固定长度突发方式，传输边界在总线周期结束处。

6.3 不定长突发过程中的 AXBS 仲裁

AHBLite（正式名称为 AMBA AHB lite version 2.0）总线是访问 AXBS 的接口。总线上的许多事务是单次传输或固定长度突发。该总线支持不定长突发总线周期。ENET 和 USB-HS 控制器等主机可请求不定长突发传输。

AXBS_MGPCRn[AULB]字段定义了在发生仲裁时，不定长突发传输过程的哪个点作为总线边界。该设置允许优先级较高的主机在突发过程中获得从端口的支配权， 无需等待整个突发过程结束。AULB 可按如下配置:

- 不允许不定长突发过程中的仲裁
- 任何不定长突发节拍之间允许仲裁
- 4 个不定长突发节拍后允许仲裁
- 8 个不定长突发节拍后允许仲裁
- 16 个不定长突发节拍后允许仲裁

6.4 AXBS 默认连接

AXBS 不仅可提供仲裁方案，还可以将每个从端口默认连接至对应的主机。当从端口空闲时，AXBS 会将其默认连接至适当的主机。如果默认连接主机请求对从端口进行下一访问，则无需等待 AXBS 即可直接开始访问。如果一个空闲端口接收到非默认连接主机的请求，则当 AXBS 切换至该主机时需要延迟一个时钟周期。

与仲裁类似，默认连接设置由 AXBS_CRSn[PCTL]对每个从端口进行配置。默认连接选项有:

- 固定默认连接 - 在此模式下，从端口在空闲时始终默认连接于同一主机。默认连接于哪个主机由 AXBS_CRSn[PARK]字段控制。如果仅有一个主机将访问指定的从端口，或需保证特定主机的等待时间，则将使用上述设置。默认情况下，所有从端口通过固定默认连接方式默认连接于主机 0（Cortex M4 内核）。
- 默认连接于上次主机 - 在此模式下，从端口将默认连接于上次使用该端口的主机。如果对从端口的大多数访问将成批出现，或试图使各主机平均分享端口，则将使用上述设置。
- 无默认连接 - 在此模式下，从端口在空闲时不默认连接于任何主机。这意味着任何主机在试图访问该端口时将产生至少一个时钟周期的代价。一般情况下极少系统会使用无默认连接方式。这种模式可用于降低 AXBS 模块内的功耗，但在大多数情况下，其功耗节省量与整体功耗相比非常微小。

7 总结

- 了解应用和优先级。某些优化将有助于提高整体性能，而很多优化方案则是在各方面性能顾此失彼情况下的一种折衷方案。必须清楚优化目的。
- 预先对数据传送和代码位置进行计划。并非所有存储器地址都是同等的。注意，访问不同的存储器位置可能需要不同的等待时间。并且，不是所有主机都能访问所有地址，而默认缓存模式也不同（对于带有缓存的器件来说）。
- 使用 SRAM_L 和 SRAM_U 块存储关键代码和数据。这里也适合放置堆栈。SRAM 块是部件上速度最快的存储器，因此尽量充分利用它们。

- 利用闪存控制器(FMC)中固有的闪存加速特性。虽然以 25 MHz 频率运行整个芯片，使内核和闪存的时钟比达到 1:1 可以消除等待状态，但是，除非确实需要闪存无等待地执行程序 (为了满足确定性要求)，否则，内核以较高频率运行产生的整体性能将会更佳。如果内核运行的时钟速度比闪存时钟快，某些情况下可能会出现等待状态，但 FMC 的设计目的就是尽量减少等待状态。
- 如果器件上有系统缓存，应尽量使用。缓存命中与将代码/数据存入 SRAM 一样快。确保系统中有缓存管理软件，以避免一致性问题。
- 如果 FlexBus 或 DDR 控制器上的外部存储器将用于存放代码，应确保通过 CODE 总线上的存储器别名区来访问指令。这样比通过系统总线地址访问存储器要节省一个时钟周期。对于这些位置也建议使用系统缓存。
- 合理使用代码优化功能。编译器通常为优化速度或空间提供了一个选择。表面上看来，优化速度是优化性能的最佳方案，但事实并非总是如此。如果对空间的优化可以为 SRAM 块中的模式代码留有余地，或者更利于在缓存中容纳函数，那么优化空间可能更利于性能。实验可以设置的开关选项，以确定最佳的编译器设置。
- 并行化是提高系统整体性能的最佳方式。利用(AXBS)及其功能以实现并行无阻塞传输。两个 SRAM 块也可支持并行访问。
- 采用 DMA 传输大块数据。许多情况下，DMA 传输数据比内核的效率更高。使用 DMA 还可释放内核，便于内核执行其他任务 (更加并行化)。
- 注意 AXBS 仲裁和默认连接的设置。可能需要进行某些实验以找到最佳配置。

8 修订历史记录

表 4. 修订历史记录

修订版本号	日期	重要改动
0	2013/05	初始版本
1	2014/06	更新了系统总线等待状态信息

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

本文档中的信息仅供系统和软件实施方使用 Freescale 产品。 本文并未明示或者暗示授予利用本文档信息进行设计或者加工集成电路的版权许可。 Freescale 保留对此处任何产品进行更改的权利，恕不另行通知。

Freescale 对其产品在任何特定用途方面的适用性不做任何担保、表示或保证，也不承担因为应用程序或者使用产品或电路所产生的任何责任，明确拒绝承担包括但不局限于后果性的或附带性的损害在内的所有责任。 Freescale 的数据表和/或规格中所提供的“典型”参数在不同应用中可能并且确实不同，实际性能会随时间而有所变化。 所有运行参数，包括“经典值”在内，必须经由客户的技术专家对每个客户的应用程序进行验证。 Freescale 未转让与其专利权及其他权利相关的许可。 Freescale 销售产品时遵循以下网址中包含的标准销售条款和条件：freescale.com/SalesTermsandConditions.

Document Number AN4745
Revision 1, 06/2014

ARM® POWERED

freescale™