使用 Cortex-M7 紧密耦合内存快如闪电地运行算法

作者：
Atmel 产品营销高级总监 Jacko Wilbrink
Atmel 产品营销经理 Lionel Perdigon

ARM® Cortex®-M 系列在要求具有稳定性能、成本效率、低功率的嵌入式系统中无疑是占据主导的处理器产品系列。最早的 Cortex-M 处理器主要关注产品的某些领域：Cortex-M0 成本最低，Cortex-M0+ 能效最出色，Cortex-M3 功率/性能最平衡，Cortex-M4 则适合需要数字信号处理 (DSP) 能力的应用。

现在，最高性能成员 — Cortex-M7 — 的首个实现正在成形。这款处理器具备了以更高速度运行更复杂算法所需要的性能，而且仍然可以维持最低的能耗和总体成本。在一个 65 nm 的嵌入式闪存处理设备中，Cortex-M7 在以 300 MHz 运行时可以达到 1500 分的 CoreMark 评分，而其 DSP 性能将比使用 Cortex-M4 翻一倍。双精度浮点单元和双指令流水线进一步提高了速度，打开了 Cortex-M7 的销路。

得益于这些特性，在低功率、低成本应用中的丰富音频和视频效果需要重要音频和视频算法时，Cortex-M7 就成为了自然而然的选择。Cortex-M7 无比适合遥控装置的图像处理、汽车与更高端的物联网 (IoT) 传感中的音频、声音控制、对象识别和复杂的传感器融合。

用于确定的代码迅速执行的紧密耦合内存

通常情况下，FIR、FFT 或双二阶等算法需要尽快且尽量具有决定性地运行，以便实时响应或提供无缝的音频和视频性能。要满足系统性能和延迟目标，此类例程应以完整时钟频率运行，不受缓存缺失、中断、线程交换和其他不利于确定时间的执行意外影响。

这意味着您不能简单地在标准内存外运行此类代码，而且您也不能依赖标准内存保存处理的数据。一般的内存，例如闪存，速度过慢，无法跟上 Cortex-M7 内核的速度，而且它们要求进行缓存，这会在缓存缺失发生时导致较长的延迟。所以 Cortex-M7 架构通过“紧密耦合内存”或称 TCM，提供了一种绕过标准执行机制的方法。

ARM 内核提供了 TCM 接口，其专为以最高处理器速度运行的单周期随机 TCM 访问而设计。64 位指令内存端口 (ITCM) 支持双指令处理器架构，而且因为有两个数据端口 (DTCM)，就可以实现两个并行 32 位数据的同时访问。然而，该架构并未指定应提供什么类型的内存或需要多大内存。这将留给在微控制器 (MCU) 中使用 Cortex-M7 内核的设计师决定，这也是产品创新和脱颖而出的机遇。

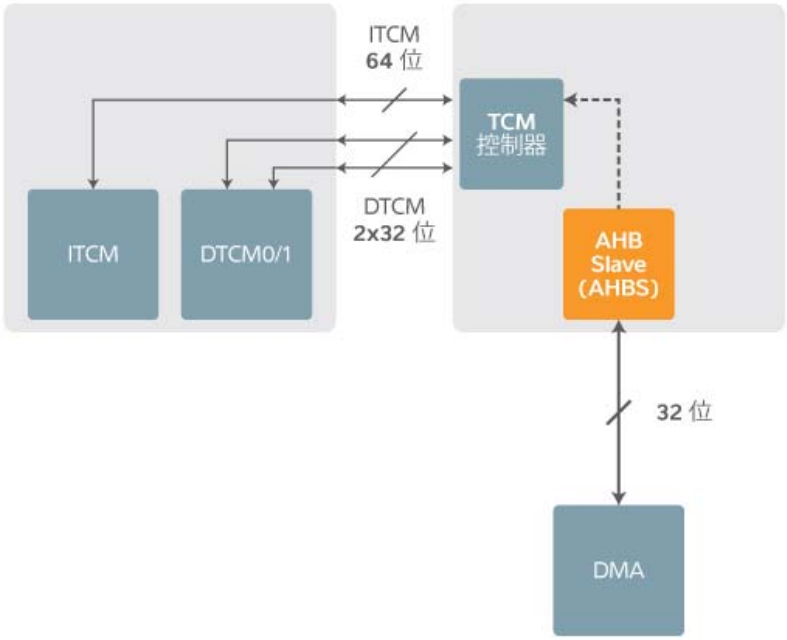

图 1。TCM 接口提供了一个 64 位指令端口和两个 32 位数据端口。

所以，举例来说，可以将嵌入式闪存装入 TCM 接口中。但是闪存无法以处理器时钟频率运行，所以就需要缓存 — 以免威胁 TCM 本应提供的决定性。DRAM 中消隐的代码在理论上可行，但成本高昂。唯一一种足够快、能够直接进行无缓存访问的内存是 SRAM。

SRAM 具有许多优益之处：SRAM 锁可以直接连接至 TCM 接口，SRAM 技术也可以轻松嵌入芯片，SRAM 可以实现以处理器速度随机访问。SRAM 的唯一缺点就是单位比特成本，其高于闪存和 DRAM。所以，限制 TCM 大小至关重要，即使 65 或 40 纳米技术能够实现以可接受成本进行大量 SRAM 集成。

TCM 可以从众多来源加载，架构也并未指定这些来源。MCU 实现有可能在他的存储器加载方式上发生变化.。直接内存存取 (DMA) 引擎当然要使用，但其是单一的 DMA 还是若干来自视频或 USB 等各类流的加载数据，则留给 MCU 设计师决定。

系统程序员将识别在 TCM 外执行的代码。在准备进行软件构建时，程序员将需要确定哪些代码段和数据块应分配给 TCM。一般通过将杂注嵌入软件和应用链接器设置，从而在构建中恰当放置代码和内存，完成这些分配。

多个端口，实现更快 SRAM 访问

虽然 TCM 提供了可以迅速严格执行关键例程的简单机制，但通常借助 DMA 将一定数量的系统 SRAM 用作处理器和外设的通用高速内存也有帮助。虽然此类内存将从逻辑上与 TCM 分隔，但单一 SRAM 块也可以共享 TCM，承担通用责任，还提供了在每个用例中调和 TCM 和系统 RAM 之间分割的可能性。

在通用系统 SRAM 中实现的外设数据缓冲一般使用从系统外设传输的 DMA 加载。但从大量来源加载的能力有可能带来不必要的延迟和冲突，因为多个 DMA 可能试图同时访问内存。

在典型示例中，我们可能有三个不同实体竞争 DMA 对 SRAM 的访问：处理器（在此示例中 64 位访问，请求 128 位）和两个单独的外设 DMA 请求（DMA0 和 DMA1，每个都是 32 位访问）。我们假设处理器优先于 DMA，DMA0 优先于 DMA1。

此示例在图 2 中说明。左侧显示的是 64 位单 Bank 内存；右侧显示有四个 32 位 Bank 的相同内存中进行的相同事务。

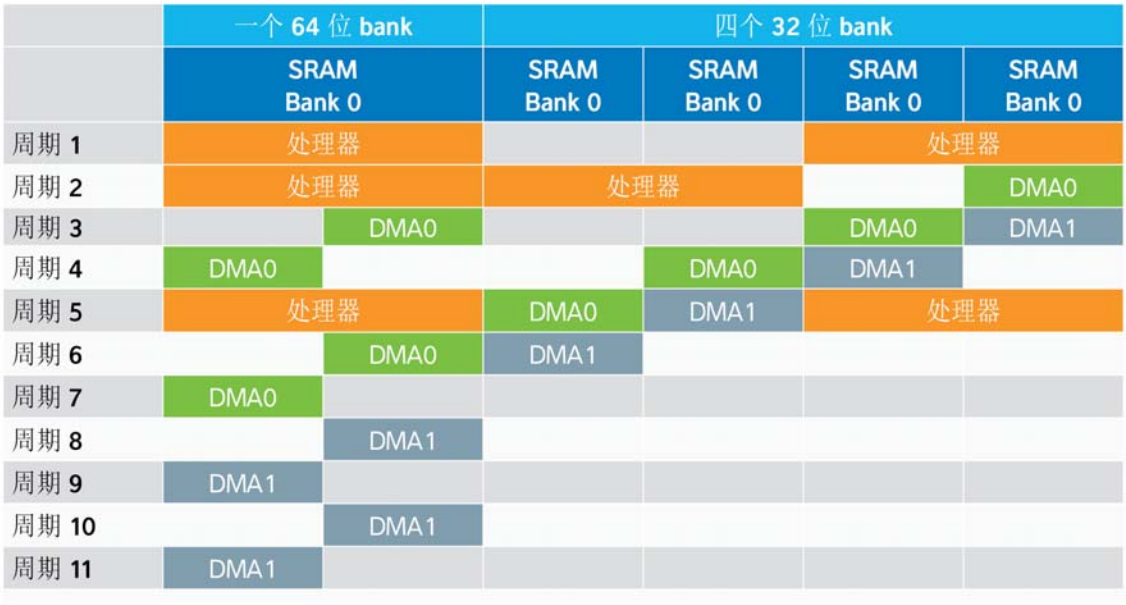

图 2。通过将 SRAM 组织成 Bank，多个 DMA 突发可以在延迟最低的情况下同时进行。

通过单 Bank 内存，处理器将在两个周期内完成其访问，然后在下一个周期中 DMA0 突发开始。DMA1 将被锁定，直至 DMA0 完成。任何更高优先级的处理器请求都将中断 DMA 加载，增加更多延迟。如上所示，周期 5 中的指令取出为两个 DMA 运行都又增加了一个延迟周期。

更多有效操作可以通过将内存组织为有交错地址（可以独立访问其中每一个）的 Bank 实现。这样一来，DMA0 突发将在处理器完成首个 Bank 后开始 — 与处理器访问更高 Bank 同时。DMA1 不再需要等整个 DMA0 传输完成；它可以在 DMA0 完成首个 Bank 后，在 DMA0 之后开始一个周期。

因为有更高优先级，处理器不会再遇到两个排列之间的延迟差异。但 DMA0 的延迟在改善后从两个周期变为一个周期，而且 DMA1 的延迟从七个周期降到两个周期。DMA 运行期间的指令取出甚至可能会与 DMA 运行并行发生，不会带来额外的延迟。延迟降低改善了性能，但也意味着您可以将外设 FIFO 变得更小。

考虑到这些内存排列可能有用，就留给 MCU 设计师探索如何将闪存、系统 SRAM 和 TCM 组合在一起。它们是否都分隔开来？它们是否占据类似空间？是否在芯片上？这些决策将分化彼此竞争的 Cortex-M7 实现。

Cortex- M7 TCM 具体示例

Cortex-M7 MCU 的一个示例就是在 Atmel® | SMART SAM S70、SAM E70 和 SAM V70/1 系列产品中实现。这些 MCU 将 SRAM 组织为四个 Bank，这些 Bank 可以用作通用 SRAM（Atmel 将其称为系统 SRAM），供 TCM 使用。在配置为 TCM 的部分，指令也能够在以 300 MHz 核心速度全速运行决定性单周期访问时运行，而且取出或缓存也并无延迟。与此类似，位于 TCM 内的数据可以在没有缓存损失的情况下访问。

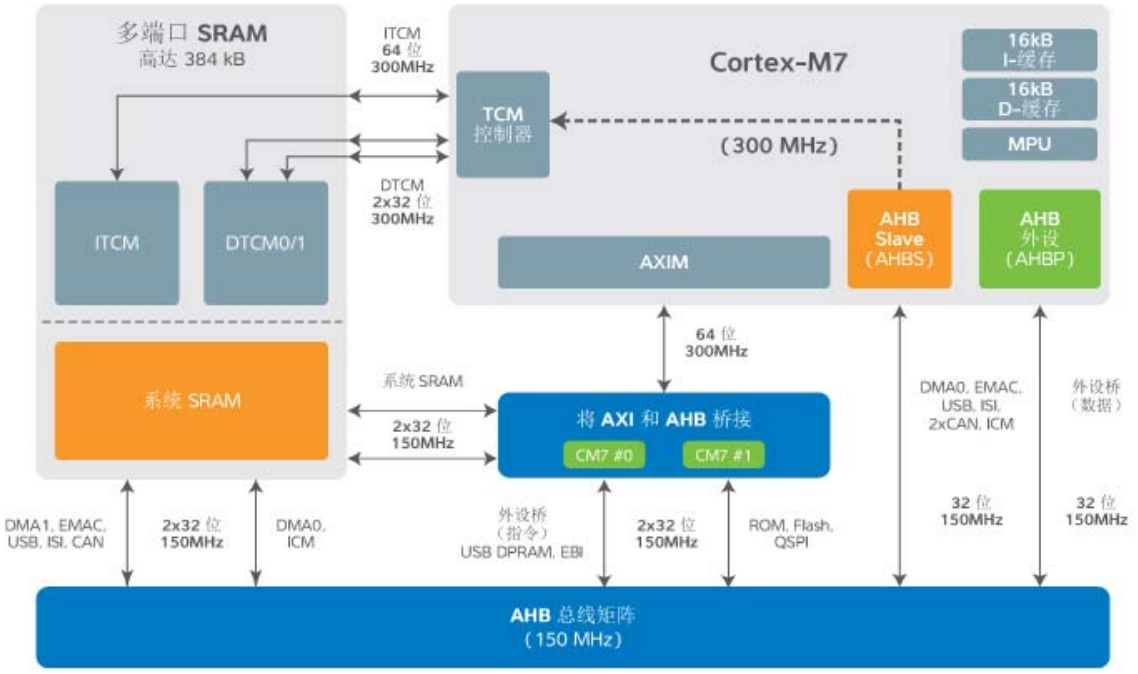

图 3。Atmel | SMART SAM S70/E70 系列产品使用可以用作 TCM 和/或系统 SRAM 的 SRAM，从而提高灵活性和利用率。

指令 TCM (ITCM) 和数据 TCM (DTCM) 必须调整到类似大小，每个都可以是 32、64 或 128 KB。大小调整是构建时的设置，不能匆忙更改。

整个 SRAM 块的大小最高可以达到 384 KB。这意味着系统 SRAM 和 TCM 可以同时使用。配置为 TCM 的内存大小降低了可以用作系统 RAM 的内存大小；例如，如果配置为 128 KB 的 ITCM 和 128 KB 的 DTCM，则系统 RAM 的大小将降到 128 KB。系统 SRAM 的运行最高达到 150 MHz，或者处理器速度的一半，并组织为四个 Bank，降低了 DMA 延迟。DMA 访问优先性可以固定或者轮询，与系统 SRAM 相同，因为它们是供内部总线矩阵使用。

寄存器位激活或取消激活 TCM，所以给定访问的目标将取决于 TCM 激活。设备始终在 ITCM 取消激活时启动，从而从启动内存加载代码。在启动后，TCM 可以酌情激活。

ITCM 地址空间和启动内存重叠。如果取消激活 TCM，则指令访问将转到启动内存。如果激活 TCM，则 TCM 地址空间内的访问将来自 TCM；TCM 空间以外的访问将来自闪存。

使用 MPU 可以保护各类空间，确保不会意外访问错误内存。

Cortex-M7 是下一波 ARM 浪潮

Cortex-M7 名副其实，它带来了更为强大的算法，但仍然保持了低功率和低成本。通过紧密耦合内存 — Cortex-M7 的新特色之一，实现了以最高处理器时钟频率进行确定的执行的同时进行严格快速的代码执行。

具体如何在给定 MCU 上实现此性能将取决于将 Cortex-M7 集成入 MCU 的设计师对系统的选择。最高速度将通过使用 SRAM 实现 TCM 支撑。此 SRAM 可以与更加综合的系统 SRAM 能力共享，而且通过将 SRAM 组织为 Bank，DMA 延迟可以最小化。

这些架构特点相对简单，大幅提升了在个人、工业和汽车设备中实现重要嵌入式功能时可以使用的性能。

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**

© 2015 Atmel Corporation. / Rev.: Atmel-45151A-Fast-Algorithms-with-Cortex-M7_Article_CS_072715

Atmel,® Atmel logo and combinations thereof, Enabling Unlimited Possibilities,® and others are registered trademarks or trademarks of Atmel Corporation or its sub-sidiaries. ARM,® ARM Connected® logo and others are the registered trademarks or trademarks of ARM Ltd. Other terms and product names may be the trademarks of others.

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.