TMS320DM64x/C64x ™ 上的视频编码优化

Cheng Peng
DSP 视频/影像解决方案

摘要

数字视频编码在许多应用领域中扮演重要角色，如数字视频监视系统和视频会议系统。 本应用报告描述了 TI TMS320DM64x/C64x DSP 上的通用视频编码器的优化技术。 TMS320DM64x/C64x 是一种高性能的数字媒体处理器，具有两级存储器/高速缓存体系和超长指令字 (VLIW) 架构。 DM64x 上的视频编码优化组合了多种技术，包括算法/系统优化、存储器缓冲优化、增强直接存储器存取 (EDMA) 和高速缓存利用优化。

内容

附图目录

附表目录

1 DM64x/TMS320C64x 介绍

TMS320DM64x/C64x™ 设备基于德州仪器 (TI) 开发的第二代高性能超长指令字 (VLIW) 架构 VelociTI.2™。 此器件的关键特性（如 VLIW 架构、两级存储器/高速缓存体系和 EDMA 引擎）使它成为计算密集型视频/图像应用领域（如视频编码和分析）的理想选择。 当在 DM64x 上开发应用程序时，为了获得优化的性能，透彻了解它的特性和存储器结构非常重要。

以 DM642 为例，以下列出了所有 DSP 内核和必需的特性：

- 用于视频/图像应用领域的增强功能单元
  - DM64x 的八个功能单元中的 VelociTI.2™ 扩展包括新的加速视频和影像应用性能的指令。

- L1/L2 存储器体系
  - 具有 32 字节高速缓存行的 16K 字节直接映射 L1P 程序高速缓存。 （8 周期 L1P 高速缓存缺失损失）。
  - 具有 64 字节高速缓存行的 16K 字节 2 路设置关联的 L1D 数据高速缓存。 （6 周期 L1D 高速缓存缺失损失）。
  - 256K 字节 L2 统一映射 RAM / 高速缓存（灵活的 RAM / 高速缓存分配、8 周期 L2 高速缓存缺失损失）
  - L2 4 路设置关联的高速缓存具有 128 字节的高速缓存行
- 字节顺序：低位地址存低位字节、低位地址存高位字节
- 64 位外部存储器接口 (EMIF)
  - 同步和异步存储器的无缝接口
- 总共 1024M 字节的可寻址外部存储器空间
- EDMA 控制器（64 个独立通道）

片上外设集包括：三个可配置视频端口；一个 10/100Mb/s 以太网 MAC (EMAC)；一个管理数据输入/输出 (MDIO) 模块和一个 VCXO 内插控制端口 (VIC)。 视频端口外设提供到常用视频解码器和编码器器件的无缝接口。 它们支持多种视频解决方案和标准（例如， ITU-BT.656、BT.1120、SMPTE 125M、260M、274M 和 296M）。

上述的存储器存取的特性和开销对包括视频编码在内的所有算法实现非常重要。 两级存储器/高速缓存体系和 EDMA 引擎从本质上确定了视频编码器实现的架构。

对于算法实现，需要考虑与存储器/高速缓存体系和 EDMA 引擎有关的一些基本概念。 当代码大小超过 L1P 的大小时，可能出现 L1P 高速缓存缺失，CPU 会拖延直到取到所需的代码。 类似地，当 L1D 容不下数据时，会出现 L1D 高速缓存缺失和 CPU 拖延。 L2 高速缓存 / SRAM 将解决所有 L1P 和 L1D 的缺失。 如果代码和数据超过 L2 高速缓存的大小，则会出现 L2 高速缓存缺失。

高速缓存友好的程序分块和数据传输处理（例如，减少 L1/L2 缺失）是保证视频编码器的最佳性能的两个关键因素。 EDMA 更适用于在 L2 SRAM 与片外存储器之间传输代码/数据。

2 视频编码器系统/算法优化

常用视频编码算法的结构图如图 1 中所示。 可以从此算法图派生出许多视频编码标准，如 MPEG2、H.263 和 MPEG4。 在图 1 中，DCT 和量化 (Q) 减少了视频的空间冗余。 运动估值 (ME) 减少了视频的时间冗余，VLC 是有效封装数据的熵编码方法。

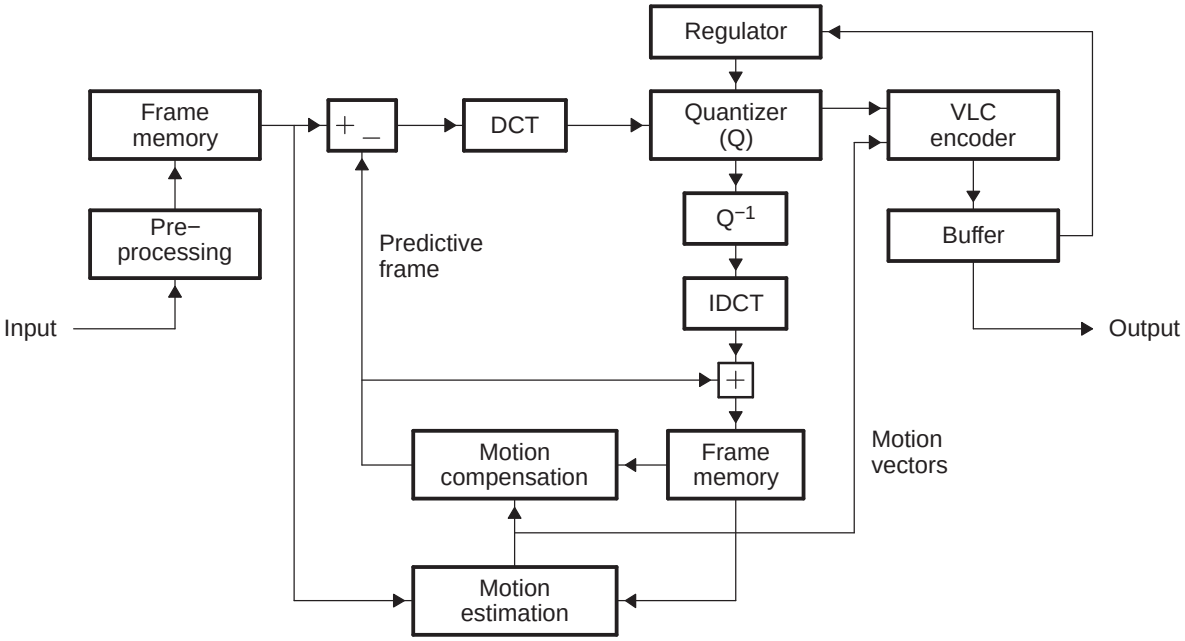

图 1. 视频编码结构图 [1]

视频编码器的传统实现基于宏块层处理。 视频编码器只在当前宏块 (MB) 经过所有处理步骤之后才去取新的宏块。 这种直观的方法有以下两个缺点：

- 视频编码器的整个代码大小通常大于 L1P。 在每次获取宏块期间，代码需要在 L1P 和 L2P 之间切换。 这会导致显著的高速缓存缺失损失。
- 通过 EDMA 将较小的数据块（如单个宏块）从外部视频帧存储器传输到内部存储器的效率不高。

为了避免大量的高速缓存缺失损失和 CPU 拖延，可以将算法分解为三个循环/模块，每一个都能装进 L1P。 每个循环每次处理 M 个宏块（宏块条）而不是一个宏块。 M 是宏块条的大小，只受可用 L1D 大小的限制。 M 越大，数据吞吐量就拥有更好的 EDMA 性能。

三个循环是：

- 宏块编码循环
- 运动估值循环
- 宏块重构循环

正如上面强调的那样，获取 M 个宏块并一起通过三个处理循环之一。 例如，在宏块编码循环中，当 M 个宏块被取入内部存储器时，它们被 DCT 转换、量化和熵编码。 在宏块编码循环结束之前，这组宏块不会被清理出 L1D。 相应的程序（包括 DCT、量化和 VLC 内核）也会保留在 L1P 内，直到在此循环内完成处理所有 M 个宏块。 EDMA 引擎驱动的交替存储器缓冲方案有助于减少对一组宏块执行这些循环所需的初始设置时间。 它还确保 CPU 拖延周期最少，因为传输与处理重叠进行。

2.1 宏块编码循环

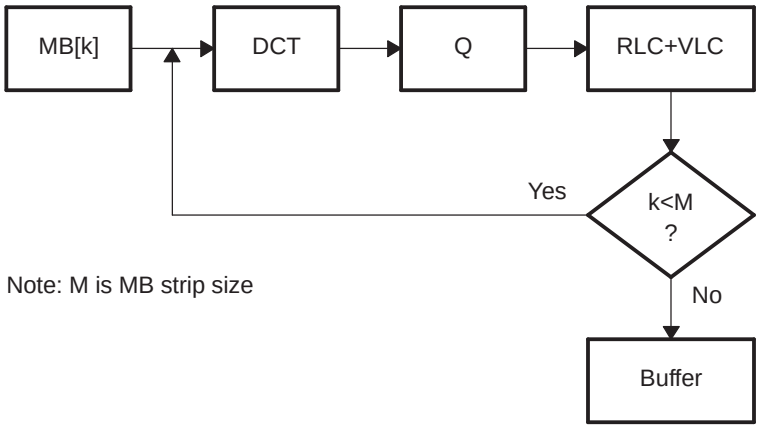

图 2. 宏块编码循环

MB[k] 是 I 帧中 M 个宏块（宏块条）中的一个，或者是需要经过 DCT、量化、运行长度编码 (RLC) 和 VLC 的 P/B 帧中的预测剩余块。 在 DCT 之后，DC 组件和 AC 组件分别通过差分编码处理。 量化主要启用压缩，也会引入量化错误。 运行长度编码按 Z 字形顺序扫描块。 它后面跟随变量长度代码指定，其中每对运行级别被作为符号处理并转换为代码。 由表查找和位操作组成的 VLC 是此循环中计算繁重的部分。 查找表被格式化以便代码字和相应的长度存储为封装的 32 位数。 VLC 编码直接将编码后的位写入以 32 位数表示的比特流。 此循环结束时的缓冲器取决于一次要处理的宏块数。 它受 L1D 高速缓存大小影响。 设计思想是让循环中要处理的所有数据进入以 CPU 速度运行的 L1D 高速缓存，同时操作此数据的代码的总大小应适合 L1P。 I 帧中的 M 个宏块或 P/B 帧中的预测剩余块需要一起通过图 2 中所示的处理。

2.2 运动估值循环

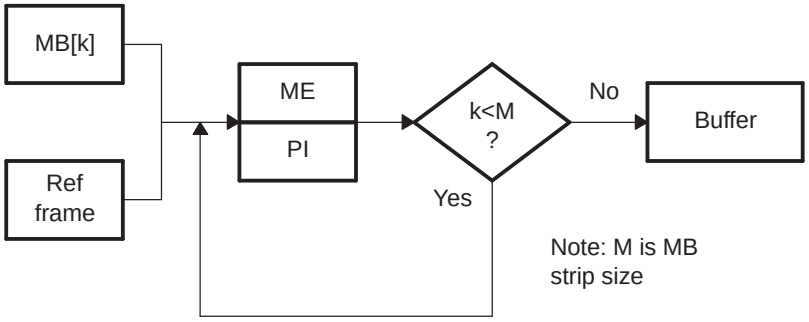

图 3. 运动估值循环

DM64x 提供丰富广泛的视频/图像指令集，可以有效地实现任何运动估值方案。 无边界调整双字加载 (LDNDW) 可能读取具有任何字节边界的 64 位值。 此指令对在当前帧中加速从宏块获取数据非常重要，特别是在参考帧中搜索窗口的时候。 它可以轻松地从当前宏块中获取 8 个对齐的像素或从搜索窗口中获取非对齐像素。 这种非对齐加载比对齐加载效率更高，因为后者需要移位操作。 减去绝对值 (SUBABS4) 指令计算源寄存器中所包含封装的 8 位数据之间差异的绝对值。 DOTPU4 是一个重要的视频/图像指令，返回四对封装的 8 位值之间的点乘。 由于两个 DOPTPU4 可以在单个周期中并行运行，所以此指令使绝对差异求和 (SAD) 处理显著加速。 这是运动估值的核心。 SAD 内核的思想可以归纳为以下步骤：

1. 两个 *LDNDW* 从当前帧和参考帧获取 8 个像素
2. 两个 *SUBABS4* 计算 8 个加法
3. 两个 *DOTPU4* 累计 8 个加法

C64x IMGLIB 的 SAD_16x16 和 MAD_16x16 函数说明如何实现上述思想，此思想在*图像/视频处理库程序参考* (SPRU023) 中进行了详细介绍。

像素内插 (PI) 也是此循环中的计算密集型块。 已使用 C64x 视频处理指令集高效地实现了它。 AVG4 指令对封装的 8 位数据执行 4 个求平均值操作，结果以无符号数的形式写入。 AVG2 指令对封装的 16 位数据执行 2 个求平均值操作，结果以无符号整数值的形式写入。 右移并合并字节 (SHRMB) 将第二个寄存器右移一个字节，然后将第一个寄存器的最低位字节合并进最高位字节位置。 在像素内插中，AVG4 计算平均值，SHRMB 更紧密地封装结果。 有关 PI 实现的详细信息，请参阅图 4。

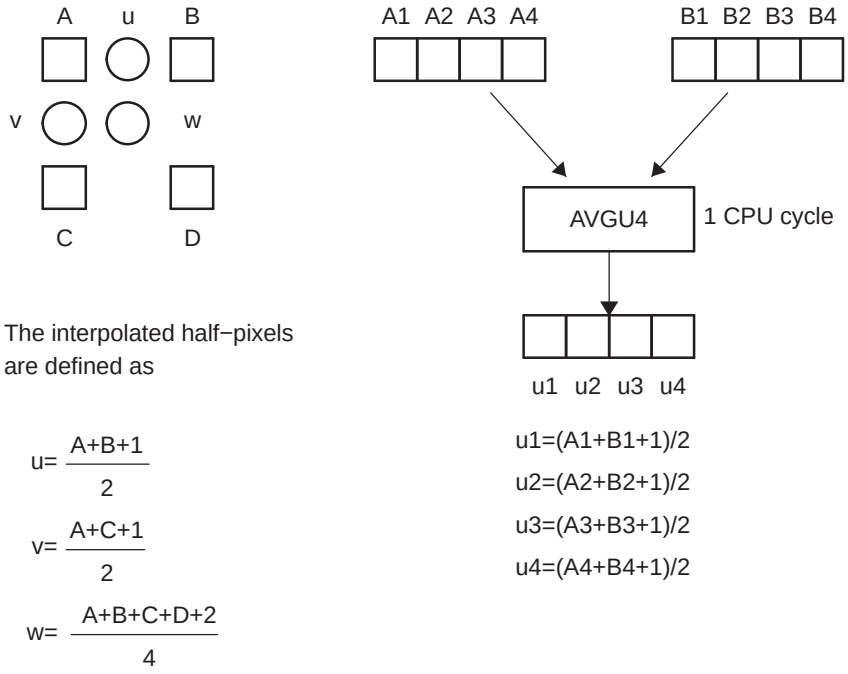

图 4. DM64x 上的像素内插

2.3 宏块重构循环

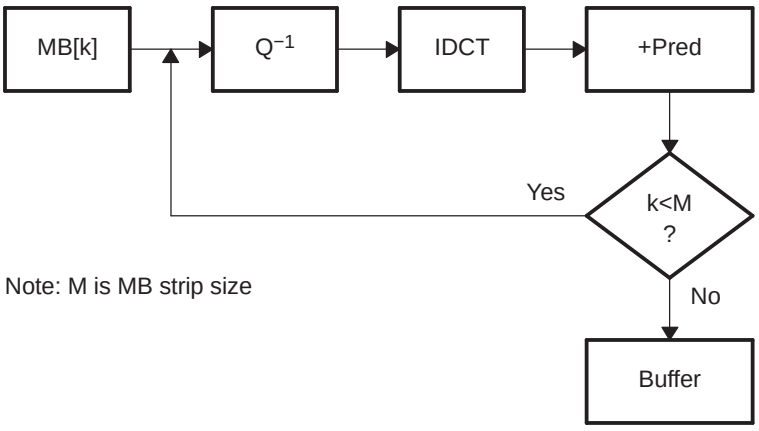

图 5. 宏块重构循环

I 帧中的宏块条或 P 帧中的预测剩余宏块条需要一起通过图 5 中所示的处理。 顾名思义，反量化是图 2 中显示的量化处理的完全相反的处理。 在 CPU 速度为 600 MHz 时，TMS320C64x 器件一秒钟可以执行 24 亿次双 16 位 MAC。 MPY2 指令执行两对有符号封装的 16 位值之间的两个 16 位乘以 16 位的乘法。 SPACK2 指令获取两个有符号 32 位数并将它们填充到有符号 16 位中。 要消除有符号值的有符号算术和舍入问题，开始时从系数中抽取了符号位，并对系数的绝对值执行了反量化操作。 符号位直接应用到最终结果上。

2.4 快速运动估值算法优化

图 1 中显示的基于块的快速运动估值模块已被每个常用的编码标准采用，包括 H.261、H.263、MPEG2、MPEG4 和 H.264。 在各种标准中，运动估值的块大小可能不同。 例如，H.264 中的 ME 最多可以支持 7 种块大小（16x16、16x8、8x16、8x8、8x4、4x8 和 4x4）。 众所周知，视频编码从运动估值获得它的大部分编码效率，因为运动估值在时间领域显著地消除了巨大的视频冗余。 另一方面，在整个视频编码中，运动估值带来最繁重的计算负荷。 好的视频编码器算法实现需要在计算强度和编码效率之间保持良好的平衡。 独占搜索（全搜索）保证在运动估值中获得全局最佳匹配结果。 不幸的是，计算成本非常大。 例如，+/-63 英寸水平和 +/-63 英寸垂直的全搜索窗口将需要处理器每秒计算 500G 次 SAD [1]。 因此，对于嵌入式解决方案，纯独占搜索 ME 是不现实的。 现实世界的运动估值将许多快速搜索技术组合在一起。 本节的余下部分介绍广泛使用的 4 步快速搜索算法。

假设块大小为 16x16，搜索窗口大小为 48x48。 4 步搜索的运动估值可以用以下等式描述：

```
Setup:
                d={8,4,2,1} // d[i] is the distance between current MB and
                            //immediately succeeding MB (See Figure 6)

 Process:
                   for(i=0; i<4; i++)
                  {
                      Compute three upper SAD for d[i].
                      Compute three central SADs for d[i].
                      Compute three lower SADs for d[i].

                  Compute the minimum value of the 9–SAD table (see Figure 6)
                 Start above process around the minimum location for the new
                  distance d[i+1].
                    }
                    // The overall algorithm is shown in Figure 7.
```

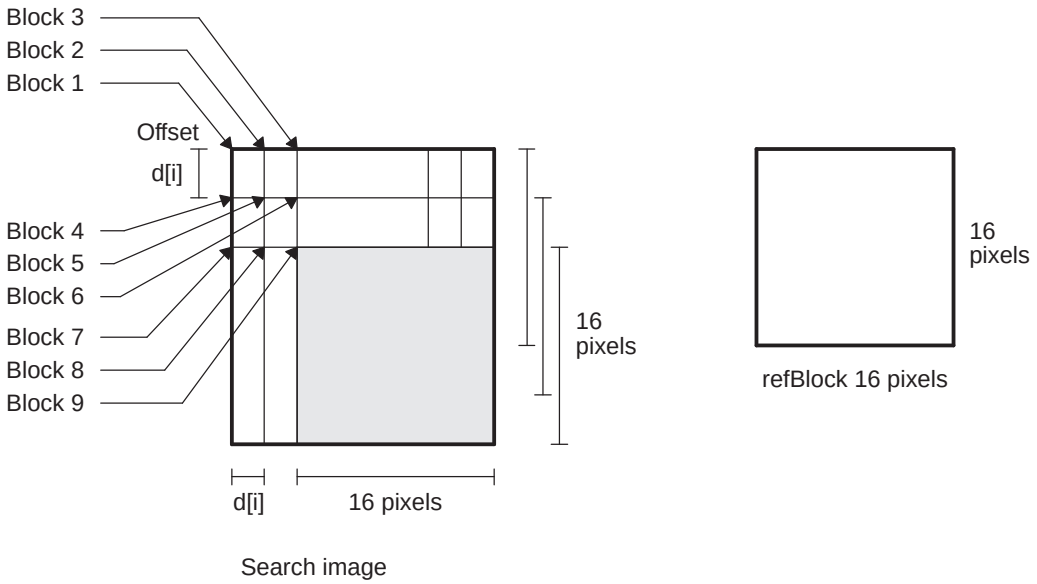

图 6. 9-SAD 表

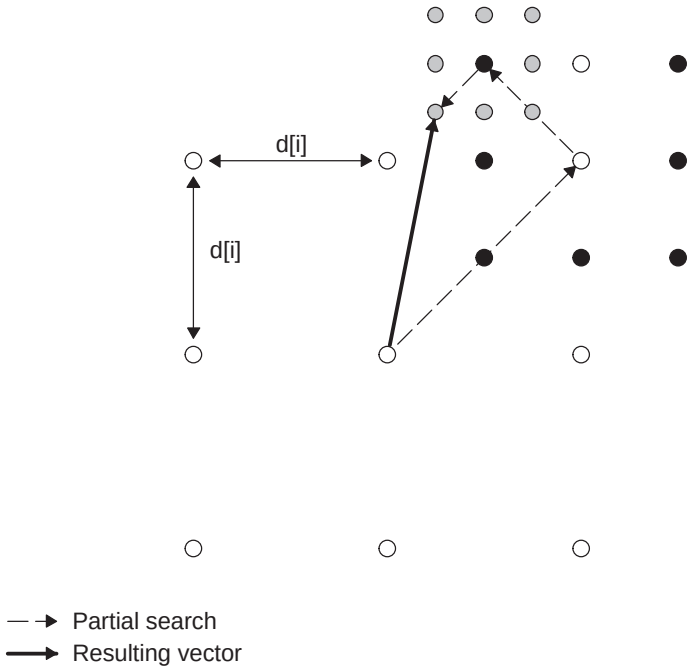

图 7. ME 的 4 步快速搜索方案

上述 4 步快速搜索技术可以显著减少计算成本。 但是，这种基本的快速 ME 技术独自可能不能提供满意的性能。 必须将许多技术组合在一起以获得优化的性能。 下面列出了几种潜在的快速运动估值优化技术 [1]：

- 分层搜索使用比例重采样的图像 [2]。 上述的 N 步 (N=2,3,4...) 搜索是分层搜索方案的一个好示例。
- 使用同一帧中相邻 MB 的运动向量作为预测向量。 （从空间相关中受益）
- 使用时间上相邻帧中并存的 MB 的运动向量作为预测向量。 （时间预测） [3]
- 望远镜式搜索 [4]

例如，组合 N 步搜索和图像比例重采样技术可以用较少的计算成本提供良好的结果。 使用这种组合作为示例，我们稍后将通过第 4 部分中的 ME 的增强直接存储器存取 (EDMA) 活动。

3 视频编码器的存储器缓冲方案

为了获得最佳性能，许多与关键内核关联的查找表、状态变量和数据缓冲器必须位于内部存储器中。 不同 C64x/DM64x 器件的内部 SRAM 不同。 除了上述基本数据消耗的存储器之外，某些 TMS320DM64x 还有足够的内部存储器来保存整个视频帧供编码器处理，而其它器件可能没有。 某些应用只需要运行视频编码器；其它应用可能运行包括视频编码在内的其它算法。 为了为大多数应用方案中的所有 DM64x 器件提供通用视频缓冲方案，全部视频帧位于外部存储器而不是内部存储器。 每次 EDMA 将 M 个宏块（宏块条）从外部缓冲器传输到内部缓冲器。 如前一节所述，M 只受 L1D 的大小限制。

图像组 (GOP) 是视频编码中的重要概念，因为它定义了每个视频帧的编码方案。 通常，在所有视频编码标准的 GOP 中定义了 I 帧和 P 帧。 B 帧只包括在高级视频编码配置文件中，如 MPEG4 高级简单配置文件和 MPEG2 主配置文件。 对于 I 帧，实现了类似 JPEG 的编码方案以消除空间冗余。 对于 P 帧，实现了前向运动估值且将前一个 I/P 帧用作参考。 对于 B 帧，不仅需要前一个 I/P 帧用于前向 ME，还需要后面的 I/P 帧用于后向 ME。 由于 B 帧的双向 ME 策略，所以视频编码器必须缓冲两个 P 帧之间的所有 B 帧。 视频帧的数据依赖关系如图 8 所示。

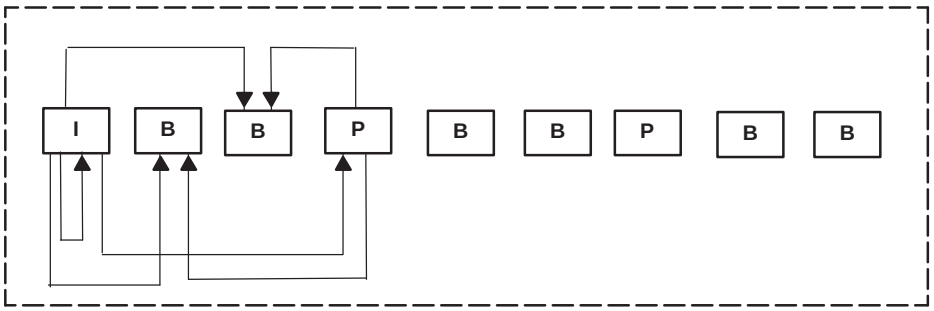

图 8. 视频编码器中的视频帧的数据依赖关系

众所周知，对于具有 B 帧的 GOP，视频捕捉/显示顺序与视频编码顺序不同。 例如，GOP=IBB PBB PBB PBB PBB 是捕捉/显示的顺序，而此 GOP 的编码顺序为 GOP ′ =IPBB PBB PBB PBB...。下表显示如何高效地在时间上对视频编码算法分区。 视频编码算法分区的关键是使 CPU 负荷在时间上尽可能恒定。

假设 vp 为要处理的视频帧，

- T(vp)：vp（I、B、P 帧）的 DCT 转换和量化。
- F(vp)：vp（B、P 帧）的前向运动估值。
- B(vp)：vp（B 帧）的后向运动估值
- C(vp)：vp（I、B、P 帧）的 VLC 编码。

表 1. 视频处理步骤

捕捉/显示顺序的视频帧	视频处理步骤
...	...
...	...
F(P4) + C(P4)	
B(B2)+B(B3)+C(B2)	
C(B3)+F(B5)+F(B6)	
F(P7) + C(P7)	
B(B5)+B(B6)+C(B5)	
C(B6)+ F(B8)+F(B9)	

整个视频编码缓冲方案如图 9 中所示。

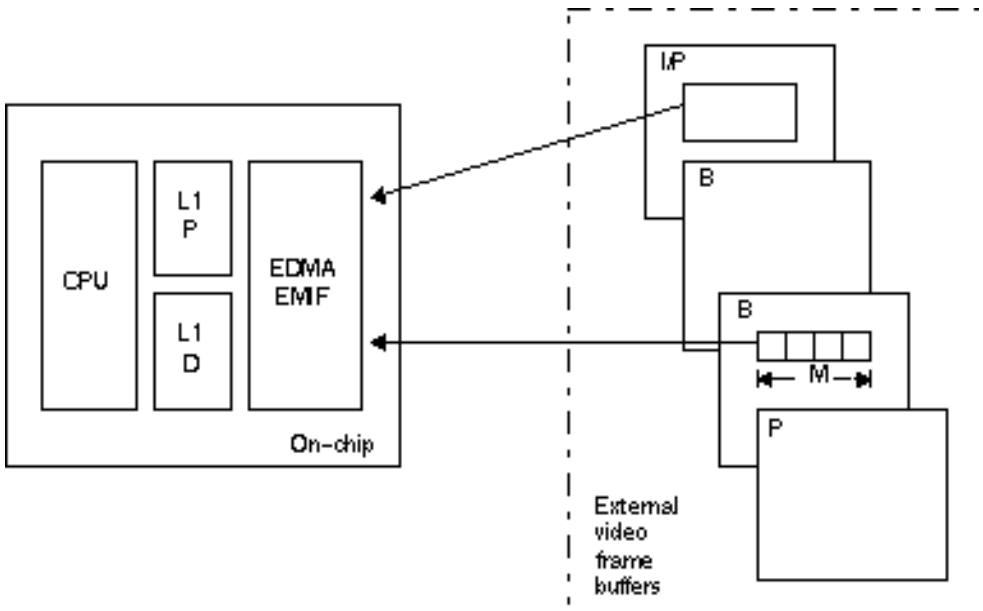

图 9. 视频编码缓冲方案

基于帧的运动估值循环中的像素内插需要来自参考帧缓冲器的一个 18x18 清晰块。 为确保最佳性能，所有 EDMA 数据传输为 32 位对齐。 即起始地址和传输数据大小必须为 4 字节的倍数。

因此，必须为所需的 18x18 块传输 18x24 清晰块。 对于彩色块，实现了类似的偏移技术。 在运动估值循环中需要考虑这些块的附加偏移。 清晰块的偏移如图 10 中所示。

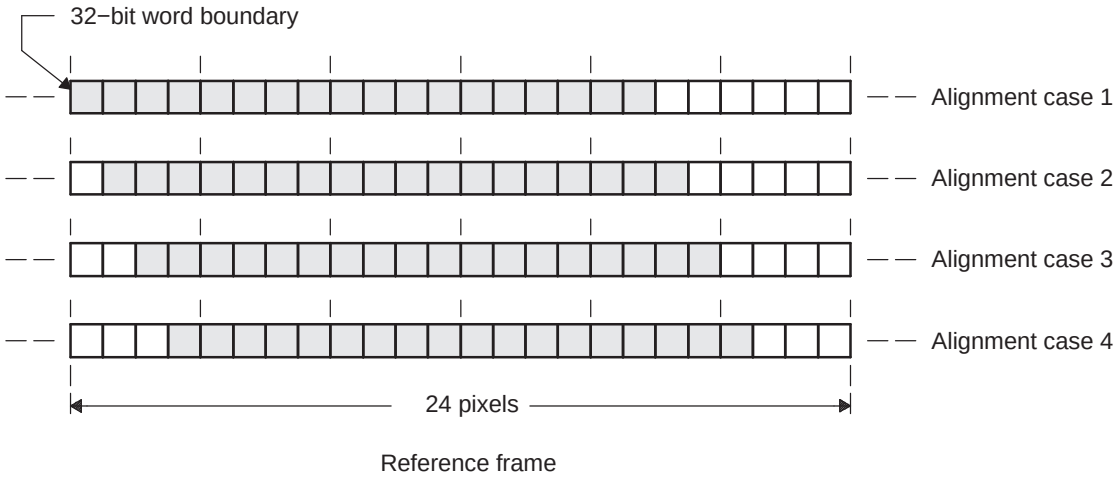

图 10. 视频编码中 PI 的存储器缓冲偏移

4 增强直接存储器存取 (EDMA) 用法

DM64x 器件的增强直接存储器存取 (EDMA) 控制器是一种高效数据传输引擎，每个 EDMA 周期最多可处理 8 个字节，这样在 CPU 速度为 600MHz 时可以产生 2.4GB/秒的总数据吞吐量。 为了使我们的视频编码器应用充分利用传输引擎的带宽，最好尽可能使用 32 位的元素大小。

为了有效地利用 EDMA，需要很好地理解 EDMA 传输进程。 在 EDMA 通信中，每次数据传输由传输请求 (TR) 启动，传输请求包含执行传输所需的所有信息：源地址、目标地址、传输属性、元素数量等。所有提交的 TR 基于优先级分成队列。 当 TR 被移进其中一个传输请求队列等待处理时，传输优先级确定它分类到哪个队列。 有四个队列：Q0（急）、Q1（高）、Q2（中）和 Q3（低）对应四种优先级别，每个队列可容纳 16 项。每个传输请求程序是可编程的，以便它可以提交任何优先级别的 TR。 一次只能按地址生成/传输逻辑为每个优先级队列中的一个 TR 提供服务。 当 TR 到达队列的头部时，它被移入 EDMA 传输控制器队列寄存器，寄存器执行 TR 定义的实际数据移动。

EDMA 具有通过使用 CPU 的 QDMA 请求执行非同步传输的能力。 即，QDMA 传输由 CPU 进行同步。 在视频编码器中，EDMA 传输由算法的数据流进行同步，而不是由外部事件进行同步。 QDMA 更擅长发出单个独立的传输命令来快速移动数据，而不是执行类似其它 EDMA 通道的周期性或重复传输。 每个 QDMA 提交一个传输请求给 EDMA 处理。 根据优先级给请求排队，优先级较高的请求先得到服务。 由于 EDMA 结构，使用帧同步提交所有 QDMA 传输。 因此，QDMA 始终请求传输一个完整的数据帧。 只为任何 QDMA 提交发送一个请求。 好的视频编码器应并行使用所有三个优先级队列（低、中和高）以在外部存储器和内部片上缓冲器之间传输数据。 ME 中的整个 QDMA 实现如图 11 中所示。 在图 11 中，只显示了两个缓冲器中的一个。 图 11 中显示的快速运动估值是 N 步搜索和图像次采样技术的组合。

EDMA 通道和优先级利用如表 2 所示。

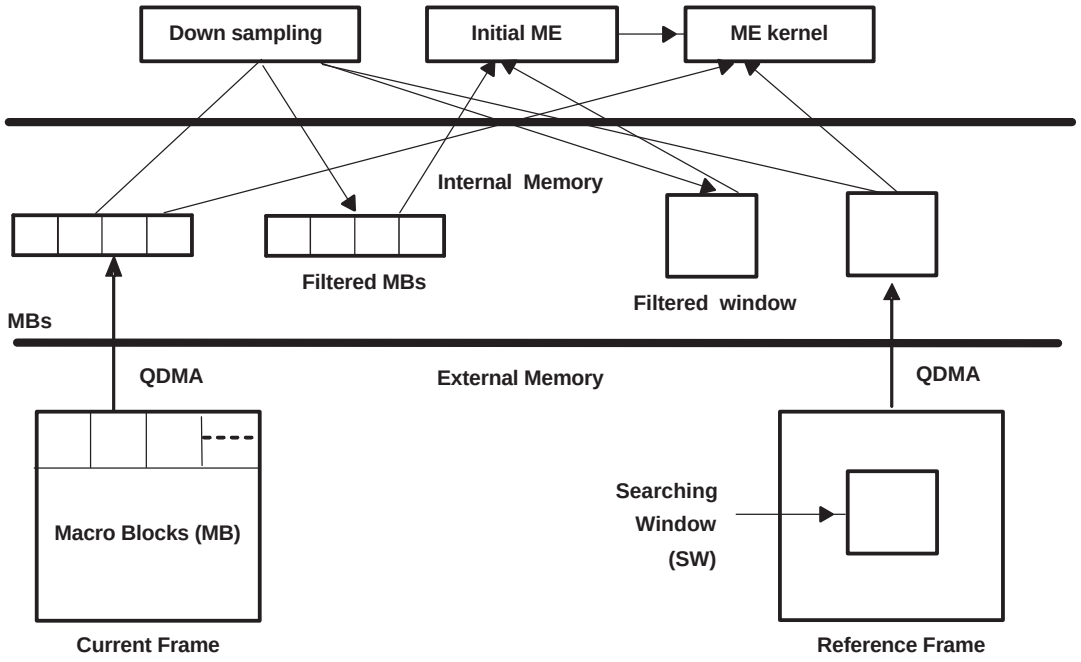

图 11. ME 的 QDMA 管理

表 2. 基于帧的 ME 的 QDMA 通道和优先级利用

EDMA 通道	优先级	传输中使用的缓冲器
I 帧情况		
CH3	低	RMB(Cr)(1)
CH4	低	RMB(Y, Cr)(1)
CH6	低	CMB(2) 的清晰和彩色信息和宏块辅助信息
CH7	低	(2)(CMB) 的彩色 (Cr) 信息
P、B 帧		
CH2	中	
CH5	中	帧参考区域获取
CH8	高	
CH3	低	RMB(Cr)(1)
CH4	低	RMB(Y, Cr)(1)
CH6	低	ME 数据、宏块信息、CMB(2)
CH7	低	CMB(Cr)(2)

(1) RMB：编码之后重构的宏块集
(2) CMB：来自正编码的帧的当前宏块集

5 高速缓存优化

最大化高速缓存效率是达到期望的全部视频编码性能的关键因素。 高速缓存效率通过减少由于存储器存活动导致的 CPU 拖延周期提高处理器吞吐量。 如第 1 节中所介绍，TM320C64x DSP（包括 DM64x）为片上程序和数据存取使用了高效的两级存储器架构。 在这种体系中，CPU 直接实现与专用一级程序 (L1 P) 和数据 (L1D) 高速缓存的接口。 这些 L1 高速缓存操作的速度与 CPU 相同，只可以读取直接映射的 L1P 高速缓存，可以读取和写入双向设置关联的高速缓存 L1D。 L1 存储器连接到称作 L2 的第二级片上存储器。 L2 是包含程序和数据的统一存储器块。 L2 高速缓存充当 L1 与片外存储器之间的桥梁。 有关这种高速缓存架构的详细文档，请参阅 *Code Composer Studio v2.3 用户指南的高速缓存分析* (SPRU575)。

高速缓存性能分析和优化贯穿于整个视频编码开发生命周期。 使用 TI 新的高速缓存分析工具，可以识别高速缓存效率问题区域并将它们可视化以便对用于视频编码的高速缓存性能进行快速改进。 ’ 视频编码开发的高速缓存优化可以分为两个不同的阶段：

- 在特定存储器背景下在视频编码算法内核级别执行高速缓存效率分析。 （请参阅图 12）
- 当将算法集成进应用程序框架之后测量 CPU 和存储器周期时，对整个应用程序级别执行高速缓存效率分析。 （请参阅图 13）

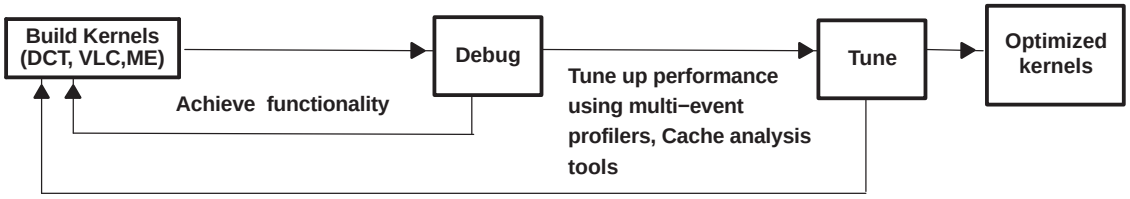

图 12. 算法内核的高速缓存效率分析

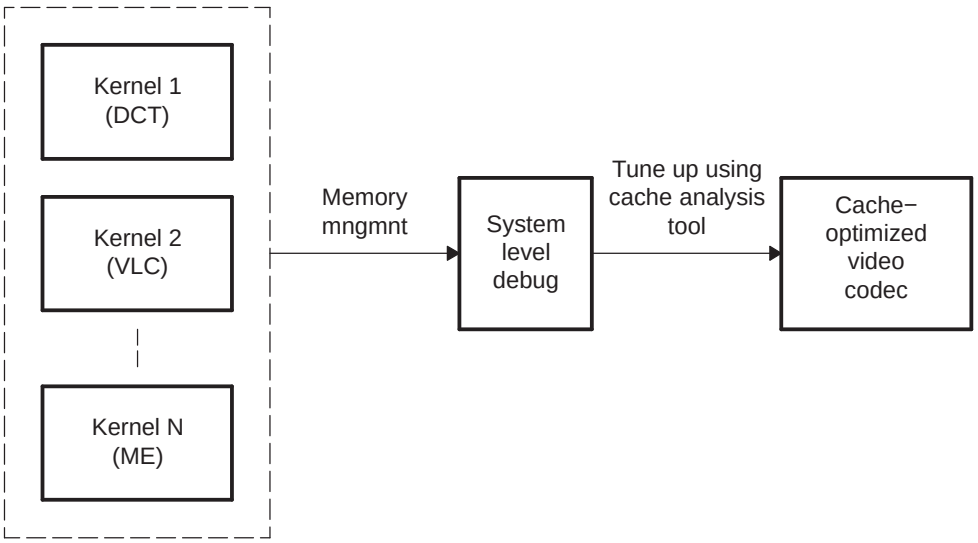

图 13. 高速缓存效率系统级别分析

使用上述高速缓存优化技术和先前各节中介绍的算法优化，可以为 DM642 上的 MPEG2 编码器获得良好的高速缓存性能（请参阅表 3）的图像所示。 未显示 L2 高速缓存命中/缺失数，因为它取决于系统中使用的外部存储器类型。

表 3. DM642 上的 MPEG2 编码器 L1 高速缓存性能

事件	计数/状态	总数	类型
I1d			
缺失			
读取	845312	845312	计数
写入	3233156	3233156	计数
命中			
读取	102447507	102447507	计数
写入	16918505	16918505	计数
I1p			
缺失	2148568	2148568	计数
命中	102964973	102964973	计数

上表告诉我们适当的高速缓存优化将使其缺损处在视频编码器基准的边缘。

6 参考

1. Anurag Jain、Ratna Reddy、Jeremiah Golston、Jagadeesh Sankaran，Programmable Real-time MPEG-2 Encoding，GSPx 2002
2. Kyoung Won Lim、Jong Beom Ra。 Improved hierarchical search block matching algorithm by using multiple motion vector candidates. Electronic Letters, Volume: 33 Issue:21, 9 Oct. 1997 Page(s): 1771-1772.
3. Ismaeil, I.R.、Docef, A.、Kossentini, F.、Ward, R。 Motion estimation using long-term motion vector prediction. Data Compression Conference, Proc. 1999 Page(s): 531.
4. J. Lee and B.W. Dickinson。 Temporally adaptive motion interpolation exploiting temporal asking in visual perception. IEEE Trans. Image Proc., 3(5): 513-526, Sep 1994.

重要声明

德州仪器 (TI) 及其下属子公司有权在不事先通知的情况下，随时对所提供的产品和服务进行更正、修改、增强、改进或其它更改，并有权随时中止提供任何产品和服务。 客户在下订单前应获取最新的相关信息，并验证这些信息是否完整且是最新的。 所有产品的销售都遵循在订单确认时所提供的 TI 销售条款与条件。

TI 保证其所销售的硬件产品的性能符合 TI 标准保修的适用规范。 仅在 TI 保修的范围内，且 TI 认为有必要时才会使用测试或其它质量控制技术。 除非政府做出了硬性规定，否则没有必要对每种产品的所有参数进行测试。

TI 对应用帮助或客户产品设计不承担任何义务。 客户应对其使用 TI 组件的产品和应用自行负责。 为尽量减小与客户产品和应用相关的风险，客户应提供充分的设计与操作安全措施。

TI 不对任何 TI 专利权、版权、屏蔽作品权或其它与使用了 TI 产品或服务的组合设备、机器、流程相关的 TI 知识产权中授予的直接或隐含权限作出任何保证或解释。 TI 所发布的与第三方产品或服务有关的信息，不能构成从 TI 获得使用这些产品或服务的许可、授权、或认可。 使用此类信息可能需要获得第三方的专利权或其它知识产权方面的许可，或是 TI 的专利权或其它知识产权方面的许可。

对于 TI 的数据手册或数据表，仅在没有对内容进行任何篡改且带有相关授权、条件、限制和声明的情况下才允许进行复制。 在复制信息的过程中对内容的篡改属于非法的、欺诈性商业行为。 TI 对此类篡改过的文件不承担任何责任。

在转售 TI 产品或服务时，如果存在对产品或服务参数的虚假陈述，则会失去相关 TI 产品或服务的明示或暗示授权，且这是非法的、欺诈性商业行为。 TI 对此类虚假陈述不承担任何责任。

可访问以下 URL 地址以获取有关其它 TI 产品和应用解决方案的信息：

产品	
放大器	http://www.ti.com.cn/amplifiers
数据转换器	http://www.ti.com.cn/dataconverters
DSP	http://www.ti.com.cn/dsp
接口	http://www.ti.com.cn/interface
逻辑	http://www.ti.com.cn/logic
电源管理	http://www.ti.com.cn/power
微控制器	http://www.ti.com.cn/microcontrollers
应用	
音频	http://www.ti.com.cn/audio
汽车	http://www.ti.com.cn/automotive
宽带	http://www.ti.com.cn/broadband
数字控制	http://www.ti.com.cn/control
光纤网络	http://www.ti.com.cn/opticalnetwork
安全	http://www.ti.com.cn/security
电话	http://www.ti.com.cn/telecom
视频与成像	http://www.ti.com.cn/video
无线	http://www.ti.com.cn/wireless

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2006, Texas Instruments Incorporated