

数字信号处理器片上流预取存储系统的设计

刘华杰, 谢 憬, 毛志刚

(上海交通大学 微电子学院, 上海 200240)

摘 要: 针对高性能 VLIW 数字信号流处理器运算速度与存储系统片外访存请求速度不能匹配的问题, 提出了一种基于流运算的片上预取存储系统, 包括数据的预取以及读写功能, 流预取控制以及流预取存储单元的状态转换等. 评估了所提出的预取存储系统的面积以及相关应用性能, 结果证明在增加少许面积的情况下, 有预取系统的处理器性能比没有预取的情况下提高了 14.6% 左右.

关键词: 流处理器; 流预取策略; 流预取存储系统

中图分类号: TN402

文献标识码: A

文章编号: 1000-7180(2014)06-0001-04

The On-chip Prefetch Memory System for Digital Signal Stream Processor

LIU Hua-jie, XIE Jing, MAO Zhi-gang

(School of Microelectronics, Shanghai Jiaotong University, Shanghai 200240, China)

Abstract: The article perform a on-chip prefetch memory system for stream processor to solve the problem that memory access can not match the processor in the high-performance VLIW DSP. The prefetch memory system can prefetch the data and read or write correctly. It contains the stream prefetch control and memory's state transformation. The article assess the area and performance of the design, the result proves that using the prefetch memory system, it only increases little areas and the performance of the stream processor can improve about 14.6% than without it.

Key words: stream processor; stream prefetch strategy; stream prefetch memory system

1 引言

高性能数字信号流处理器^[1]相比传统处理器有着数据分批处理以及并行处理的优点, 由于流的大小是任意的, 数据可以被分成不同的流以配合不同应用的特别需求. 而由于流应用具有计算密集性以及数据并行性等特点^[2], 因此流处理器被广泛用于媒体处理、科学运算等领域.

无论是传统处理器还是流处理器, 片外存储器对于数据的读写速度都不能匹配处理器的运算速度^[3], 由此导致处理器常常会停滞下来等待片外存储单元进行读写数据, 从而延长了整个处理器的执行时间. 因此人们开始研究基于片外存储单元的数据预取技术. 文献[4]中采用了一种步长自适应和信

心系统的预取方式^[5], 以此来保证提高预取的性能, 该机制使用访存失效地地址作为查询预取表的索引并简化了实现的硬件结构, 而信心系统机制的引入能够有效地降低预取失效率. 文献[6]中提出了一种基于值预测的预取技术, 集指令预取和数据预取为一体, 利用步长反馈机制, 提高预取的有效性.

然而, 无论哪种预取方式对于流处理器的数据预取作用都十分有限, 因为流处理器中往往包括多个运算簇, 这些运算簇可能会同时需求完全无关的批量数据, 如果仅仅是使用预取单个数据或者某些数据的策略, 不能符合数据并行的流应用对于批量数据的需求. 因此需要一种可以配合流应用的流预取机制.

本文通过研究基于片上流存储系统的特性, 提出了一种适合片外 SDRAM^[6]的存储预取系统, 能

收稿日期: 2013-08-01; 修回日期: 2013-09-22

基金项目: 国家“八六三”重大专项子课题(2012AA012702); 国家自然科学基金面上项目(61176037)

够有效地提高存储单元的工作效率, 缩减流处理器等待存储单元工作的时间, 从而提高整个系统的性能。

2 流处理器的预取机制设计

2.1 流处理器

本文的基本结构是自主研发的高性能流处理器, 采用通过超长指令字来执行发射指令的结构。

如图 1(a) 所示, 处理器的内部有 8 个运算簇单元。VLIW 指令经过译码单元后广播给 8 个运算簇进行运算, 之后根据不同的应用开启不同的流, 运算簇会根据需求发出请求给片上的存储系统, 然后片上存储系统根据自身情况再向片外存储存取数据。由于不含预取机制, 因此一旦需求的数据不在片上, 整个处理器就需要等待片外取数。

图 1(b) 是添加了预取机制之后的流处理器结构, 粗线内标识的就是添加的流预取存储系统, 用以希望在处理器运算的时候预取数据, 提高性能。

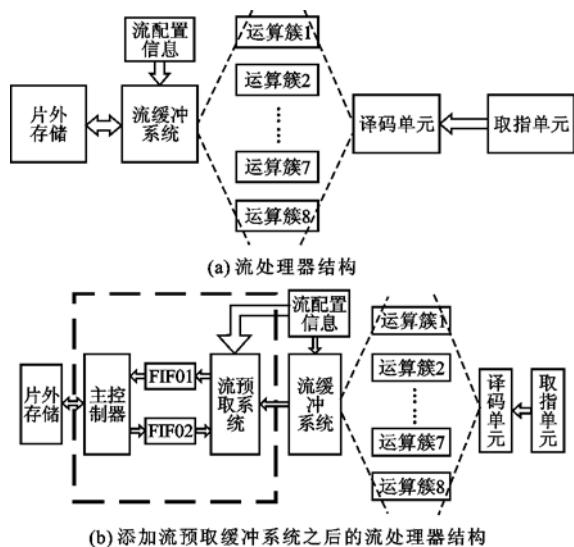


图 1 流处理器结构图

2.2 流预取流程

在没有预取的情况下, 流配置信息生成的地址会发送给流缓冲系统, 之后发送到片外存储, 这样每开启一个新的流都需要等待片外取数, 这些时间开销十分巨大。

加入预取之后, 整个流预取的过程如图 2 所示: 首先通过预取策略得到预取地址, 之后把预取地址和正常请求地址一起进行仲裁。具体仲裁的原则是正常访存优先, 即有正常请求是会把预取请求延后, 同时还会判断正常请求地址是否已经预取过或者正准备预取, 如符合之前两种情况则优先预取请求。而

预取的地址也会根据正常访存的请求进行不停的更新计算从而得到的新的地址, 此新地址再进行下一步的仲裁工作。仲裁完成之后把结果输出到一个请求队列中, 通过这个队列去访问控制器, 然后控制器根据控制逻辑来进行相应的访存工作。

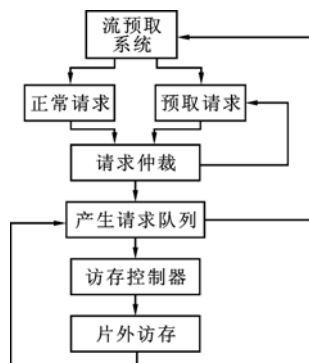


图 2 流预取流程图

由于片外存储的访存接口只有一个, 即无论进行预取与否, 同时只能进行一种读写操作。因此, 在进行预取仲裁的时候要结合流处理器的特性需求: 即如果设定的流大小是簇个数 (现有 8 个) 的多数倍的时候, 对于同一个流中的连续请求, 虽然各个请求之间有一定的间隔时间, 但是这间隔时间远小于一般预取的访存速度, 仲裁单元会判断到这种情况, 并且是不进行预取的, 否则会占据访存接口而导致正常请求的阻塞, 从而得不偿失。即预取只有在不影响正常请求的情况下才允许发生。

3 流预取存储系统的实现

3.1 流预取存储系统

流预取存储系统如图 1(b) 粗线内标识所示, 流配置信息寄存器单元会同时发送配置信息给流缓冲系统和流预取系统。此时流缓冲系统根据信息进行片上存取工作, 与此同时, 流存储预取系统也会根据配置单元给出的配置信息进行进一步的预取工作, 然后将所需要的预取地址和控制信号通过一个异步 FIFO 发送给存储单元主控制器, 此异步 FIFO 的作用还有同步两边的快慢时钟信号, 之后主控制器通过自身状态机的转化以及操作译码单元所产生的译码结果来进行片外存储的存取工作。

为了配合流处理器的大量连续数据需求, 同时结合自身的相关流应用, 主控制器配置为突发读写的模式, 一次设定为 32 个连续数据, 这个大小和预取缓冲系统中的块大小相同, 即预取系统发送一个首地址给片外存储, 片外存储就通过另一个异步 FIFO 返回 32 个连续地址的数据, 然后再由预取系

统进行下一步的相关工作。

3.2 流预取控制单元

本文的流预取是基于整个流处理器的特性所提出的. 流预取控制单元的结构如图3所示:主要由预取地址计算单元,预取判断单元,仲裁单元和状态转化单元4个部分组成. 预取地址计算单元会存储由流配置信息寄存器单元发出的配置信息,然后当系统处于预取状态的时候根据此配置信息计算得到下一个流的首地址和步长,将它们相加得到可能的可能预取地址1:

$$P_addr_1 = (H_addr + \sum_{n=1}^k \text{Stride} \ll 2) \times V \quad (1)$$

由于步长是随时可变的,因此将当前地址与下一个步长也进行相加,再次得到一个新的可能预取地址2:

$$P_addr_2 = (N_addr + \sum_{n=k-1}^k \text{Stride} \ll 2) \times V \quad (2)$$

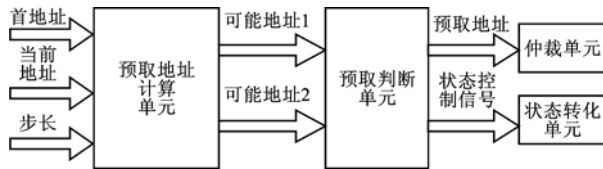


图3 流预取单元结构图

理想情况下,这些预取地址都是有效的,这些地址进入预取判断单元。

将它们和当前的地址相比较判断,比较这些计算得出的预取地址是否已经访存或者预取过,以此来决定流预取系统的下一个状态时候进行预取. 由于为了配合预取单元的块大小,片外已经设定成32位的突发读写操作模式,因此预取的时候也是32个数同时预取来配合满足流计算和片外控制器的要求. 由于现有流处理器结构中有8个运算簇单元,因此当流大小设定成8的多数倍的时候,计算预取地址的时候可能会出现遗漏地址或者重复地址,这时需要根据流大小和跨步的情况来调整预取的地址。

经过仲裁单元仲裁产生正确的预取地址和状态转化单元之后,才能真正进入预取状态. 仲裁单元的作用主要是判断是否是有效请求以及和正常请求做相关仲裁并存下所有请求。

由于流寄存器配置信息单元对于整个片上系统是可见,因此当运算簇在计算而存储系统空闲时,进入预取状态能够有效地节约后续对于该地址访存的时间,从而提升了整个处理器的性能。

3.3 预取策略实现

整个流预取系统的状态转换如图4所示,流预

取系统在正常工作的时候会进入读写状态,在空闲的时候并且预取地址已经产生而且有效的时候会进入预取状态,而无论读写还是预取状态都会进入访存状态. 由此可见,加入预取单元之后对于状态转换仅仅是增加了一个预取状态,但是却能有效地节约片外访存等待的时间。

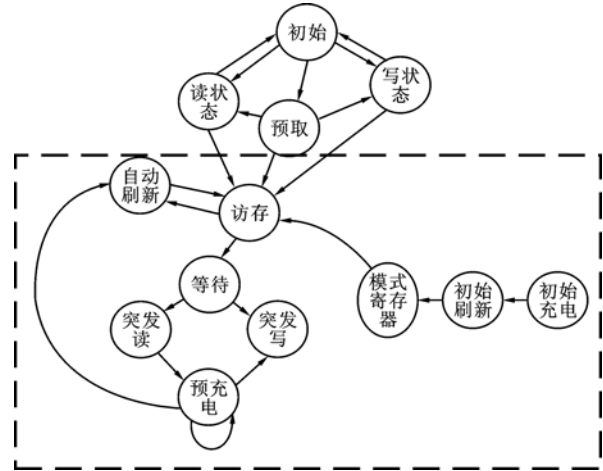


图4 流存储单元状态转换图

本文所设计的预取系统具有较强的适应性,可以配合常用片外存储如SDRAM或者DDR等使用. 其主要通过2个输入输出FIFO来同步片上和片外的不同时钟以及传送数据和控制信号,由于不同片外存储的频率不同,所使用的异步FIFO时钟频率也不同. 结合实际需求,文章的片外存储设备采用SDRAM,因此整个流预取存储系统中包含了一个基于预取优化的SDRAM控制器^[7-8],其状态转换是图4粗线内标识:开始会有初始化充电,刷新等状态,为了配合片上的预取,主要的工作状态是突发读或者写状态,一次可以访存预取需求的批量数据. 在收到访存请求之后,会根据FIFO的空信号来判断是否进入等待状态,如果非空则进入等待状态. 之后根据控制器得到的控制信号来判断进入突发读或者写状态,每次突发读或者写完成之后都会进行预充电和自刷新工作。

4 性能评估

对于采用流预取存储系统版本的流处理器,文章采用了TSMC 65 nm工艺库文件对其进行了逻辑综合,整体综合的硬件结构中基本组成部分包含了2个深度为32的FIFO以及512 kB大小的片上缓存以及基于SDRAM的流预取存储系统. 而有预取功能的版本则增加了流预取单元及其他相关控制. 其结果如表1所示。

表 1 逻辑综合结果

有无预取	时序/ns	面积/ μm^2
有预取	2.2	4 185 666
无预取	2.2	4 097 795

从表 1 中可以看出, 无论是否带有预取的存储系统都能达到 450 MHz 的频率要求, 同时预取功能单元的添加增加了少许的面积, 有预取功能的流存储系统比没有预取功能的加了 2% 左右的面积。

文中还针对 6 种不同的流应用进行了仿真验证, 所有应用的数据都是先存放在片外存储中的, 因此耗费周期数较多。例如使用了 8 个运算簇的 512 点的 FFT 和 IFFT, 1024 点 FFT, 32×32 矩阵相乘, 64×64 矩阵相乘, 32×32 CSA 算法等, 这些应用的数据量都比较大, 对于存储单元的访问也比较的频繁, 但同时访存请求间隔也比较接近, 因此留给可以预取的空间并不多, 比较了这些应用的指令周期数和执行周期数, 结果如图 5 所示。

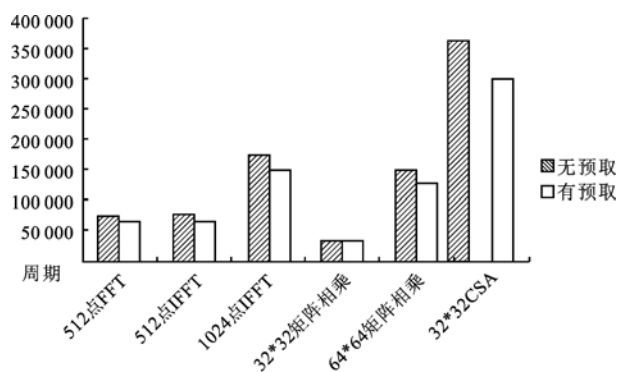


图 5 不同应用验证结果图

从图 5 中可以看出, 针对这 6 种应用, 即使在留有预取空间不多的情况下, 带有预取存储系统的流处理器比没有预取的性能分别高 12.8%, 14.1%, 15.0%, 13.0%, 14.9%, 18.0% 左右, 6 种应用平均提升 14.6% 左右。其中应用 CSA 的数据量最大, 因此提升也最大。

5 结束语

本文提出了一种基于流处理器的预取存储系统,

用以控制流处理器的访存读写工作, 同时还带有适合流处理器需求的预取功能。主要采用基于流首地址和跨步等流信息计算出可能预取地址继而进行仲裁再预取的方式, 保证在计算单元运算而访存请求空闲的时候能够进行有效的预取工作。从实验结果中可以看出, 采用该预取存储系统之后, 在面积开销少许增加的情况下, 整体性能有了 14.6% 左右的提升。

参考文献:

- [1] Khailany B, Dally W J. Imagine: media processing with streams[J]. IEEE micro, 2001, 21(2): 35-46.
- [2] Kapasi U J. programmable stream processor[J]. IEEE computer, 2003, 36(8): 54-52.
- [3] Hur I, Lin C. Memory prefetching using adaptive stream detection[C] // Proc 39th Ann. IEEE/ACM Int'l Symp. Orlando, Microarchitecture, 2006.
- [4] 勒强, 郭阳, 鲁健壮. 一种步长自适应二级 cache 预取机制[J]. 计算机工程与应用, 2011, 47(29): 56-59.
- [5] Akihiro Musa, Yoshiei Sato. Effects of MSHR and prefetch mechanisms on an on-chip cache of the vector architecture[C] // Parallel and Distributed Processing with Applications. Australia: Sydney, 2008: 335.
- [6] 肖勇. 值预测技术研究[D]. 长沙: 国防科学技术大学, 2005.
- [7] Chang Joo Lee, Onur Mutlu, Veynu Narasiman, et al. Prefetch aware memory controllers[J]. IEEE Transactions On computers, 2011, 60(10): 1406-1430.
- [8] Matthew Binuk. Design of a parallel vector access unit for SDRAM memory systems[C] // Proceedings of the Sixth International Symposium on High-Performance Computer Architecture. France: Toulouse, 2000: 39-48.

作者简介:

刘华杰 男, (1989-), 硕士研究生. 研究方向为数字集成电路设计。

谢 憬 男, (1989-), 助理研究员. 研究方向为大规模集成电路设计、片上处理器体系结构、先进集成电路设计方法。

毛志刚 男, (1962-), 教授. 研究方向为大规模集成电路设计、可重构体系结构设计、集成电路与元器件可靠性。