

引用格式:孙哲,李涛,邢立冬,等. 面向机器学习的高性能 SIMT 处理器存储系统设计与实现[J]. 微电子学与计算机,2019, 36(8):72-76. [SUN Z, LI T, XING L D, et al. Design and implementation of high performance SIMT processor storage system for machine learning [J]. Microeletronics & Computer, 2019, 36(8): 72-76.]

面向机器学习的高性能 SIMT 处理器存储系统设计与实现

孙 哲,李 涛,邢立冬,许晓燕
(西安邮电大学 电子工程学院,陕西 西安 710121)

摘 要: 针对自主研发的高性能 SIMT 处理器中多线程运算时并行数据的快速存取问题,设计了一种适用于 SIMT 架构的存储系统,其主要包含存储控制器、数据缓存(cache)和指令缓存等设备.该设计使用可综合的 Verilog HDL 语言实现其硬件电路,同时搭建基于 FPGA 的验证平台对存储系统进行功能验证.在 Xilinx 公司的 FPGA 芯片 xcvu440-flga-2892-2-e 上综合最大时钟频率可达到 285 MHz.通过各方面验证,表明所设计的存储系统满足系统要求.

关键词: SIMT;多线程;cache;FPGA

中图分类号: TP399 **文献标识码:** A **文章编号:** 1000—7180(2019)08—0072—05

Design and implementation of high performance SIMT processor storage system for machine learning

SUN Zhe, LI Tao, XING Li-dong, XU Xiao-yan

(School of Electornic Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710121, China)

Abstract: Aiming at the problem of fast access of parallel data in multi-thread operation in self-developed high performance SIMT processor, a storage system suitable for SIMT architecture is designed, which mainly includes storage controller, data cache and instruction cache. This design uses Verilog HDL language which can be integrated to realize its hardware circuit, and meanwhile builds a verification platform based on FPGA to perform functional verification of the storage system. The integrated maximum clock frequency can reach 285MHz on Xilinx's FPGA chip xcvu440-flga-2892-2-e. The verification of all aspects shows that the designed storage system meets the system requirements.

Key words: SIMT; multi-thread; cache; FPGA

1 引言

随着科学技术的不断进步,人工智能技术已经与人们的生活紧密结合在了一起,而机器学习是实现人工智能的一种关键技术.机器学习是计算机通过对现有数据的不断学习并且自我完善的一个过程,其中会涉及到对大量数据的采集、运算,普通的处理器已经不能很好的满足机器学习算法^[1].在传

统的 CPU 中通过单指令多数据^[2] (Single Instruction Multiple Data, SIMD)来处理矢量数据, SIMD 是一种采用一个控制器来控制多个处理器同时对每一个数据向量分别执行相同的操作从而实现空间上的并行性的技术. SIMD 架构处理器特点是在编程方面简单易懂,执行效率高,但其无法并行的处理有条件跳转函数.然而这个问题在单指令多线程^[3] (Single Instruction Multiple Threads, SIMT)体系

结构中可以得到很好的处理。

SIMT 是相对于 CPU 中 SIMD 的概念来的. 其优点在于开发者无需将数据凑成合适的矢量长度, 并且允许每个线程可以有不同的分支, 进行不同的操作, 这一点可以很好地解决在 SIMD 中无法并行处理的条件跳转指令. SIMD 与 SIMT 最大不同点是, 就寄存器而言, SIMD 的各个向量数据在物理上位于同一空间可以自由通信, 而 SIMT 则无法做到. 因为 SIMT 允许单个指令的多数据分开寻址, 进而导致每个线程的寄存器仅对本线程开放, 线程间无法相互自由通信, 但可以通过共享存储实现. 在 SIMT 处理器中, 实现多线程^[4]运算时并行数据的快速存取问题尤为重要, 其性能的优劣直接制约了后续模块获取结果的速度。

本文讨论设计的是应用于机器学习的自主研发 SIMT 处理器, 它采用了与一般的 CPU 和 GPU^[5]不同的体系结构, 本处理器的高性能来自于其高效的信号流处理管线, 以及类似于 GPU 的存储体系。

本文重点讨论的是自主研发 SIMT 处理器中的存储系统的硬件设计与验证。

2 高性能 SIMT 处理器结构

高性能 SIMT 处理器由 1 个向量处理单元 (Vector processing unit, VPU), 8 个运算处理单元 (Processing element, PE), 1 个存储管理单元 (Memory management unit, MMU) 等组成. 其系统结构如图 1 所示, VPU 负责对整个处理器的取指、译码、执行、线程调度进行控制, PE 主要负责对指令集中的数据运算操作进行支持. MMU 负责对处理器运行所需的指令和数据进行存取。

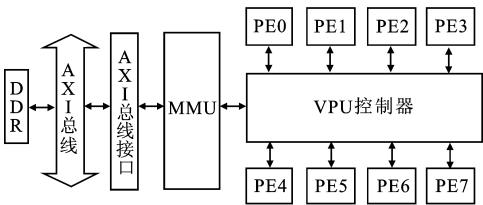


图 1 SIMT 处理器结构框图

在该 SIMT 处理器中, 包含 8 个计算内核, 每个核都可以执行一个线程, 也就是说处理器可以同步执行 8 个线程, 其中每个线程都会使用各自的数据执行相同的指令, 并且每个线程都有真假, 真线程执行, 假线程不执行. 在线程调度过程中, 线程块中的线程重新组合形成一组 warp, warp 就是 SIMT 处

理器在硬件中调度执行线程的基本单位^[6]. 在本处理器中, 最多支持 8 个 warp 的调度执行, 而每个 warp 最多包含 8 个线程. 当 warp 中的线程等待存储器数据访问时, 该 warp 将进入等待区等候, 同时切换下一个 warp 继续执行, 上一个 warp 请求的数据成功取回时才会被再次激活. 这样的调度处理很好的掩盖了处理器的访存延时。

3 存储管理单元的硬件设计

3.1 硬件设计框图

存储管理单元主要由 1 个指令缓存 (ICACHE), 1 个数据缓存 (DCACHE), 1 个共享存储 (S_MEM) 和 1 个存储控制器 (MEM_CTRL) 组成. 其结构如图 2 所示。

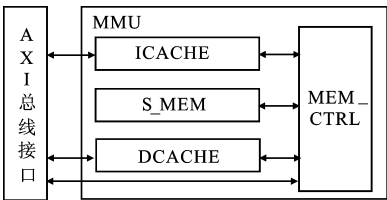


图 2 存储管理单元结构框图

3.2 模块功能描述

3.2.1 存储控制器

存储控制器是存储控制器系统的核心部件. 它记录最多 64 个未执行完的取数指令 (load) 或存数指令 (store) 操作. 虽然同时存在多个 load 和 store 操作, 但在存储控制队列中按照它们在处理器中的调度顺序排列. 由于同时多线程产生的多个读写请求有真有假, 并且针对某些特定的程序, 多个线程可能请求访问相同的地址. 为了避免大量重复的请求相同的地址, 因此需要根据线程真假情况、存取地址的相关性, 进行分类缓存, 根据分类的情况, 将请求按指定的顺序送往不同的地方. 其整体结构如图 3 所示。

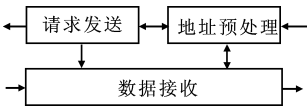


图 3 存储控制器结构框图

地址预处理中包含 8 个相同的缓存模块, 每个缓存模块对应一个 warp, 负责接收并缓存来自处理器的读写请求, 对于同一个 warp 中的相同地址的请求进行合并, 最后根据 warp 号码选择将请求缓存至对应的缓存模块中。

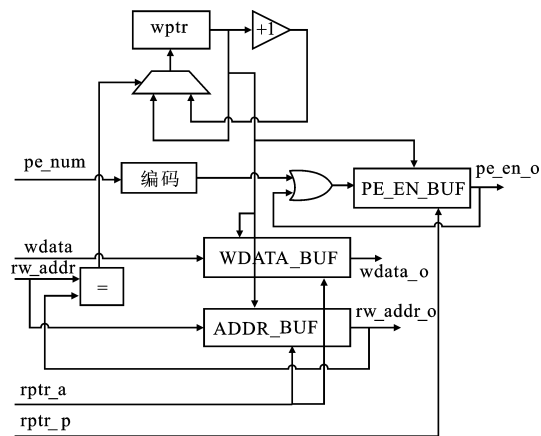


图 4 地址缓存模块结构

地址缓存模块如图 4 所示,其中包含三个深度为 8 的缓存,分别为 ADDR_BUF(地址缓存)、WDATA_BUF(写数据缓存)和 PE_EN_BUF(使能缓存).地址缓存和写数据缓存都为 32 位,用来缓存读写请求的地址以及写请求对应的写数据.使能缓存为 10 位,其中高两位用来标记对应地址的范围,寻址范围在 0~16 k 时,其值为 2'b10,属于 SRAM,寻址范围在 16 k~256 M 时,其值为 2'b01,属于 DCACHE;寻址范围大于 256 M 时,其值为 2'b11,属于 DDR.低 8 位用来标记当前地址属于哪些 pe,比如当 pe0 和 pe3 请求的地址相同时,低 8 位设置为 8'b00001001.使能缓存的初始值根据 pe 号码分别为 8'b00000001, 8'b00000010, …… , 8'b10000000.所有缓存共用同一个写指针 wptr,通过比较读写请求的地址与已缓存的所有地址,都不相等时写指针加 1,否则不变,同时更新相应的使能缓存状态,新的使能值为之前值与当前值的按位或.写数据缓存和地址缓存共用一个读指针 rptra_a,使能缓存用读指针 rptra_p. rptra_a 和 rptra_p 由请求发送模块产生.

请求发送模块负责将地址预处理模块缓存的读写请求按顺序下发至 DCACHE、SRAM 或 DDR.如图 5 所示为发送请求结构图.其中 WARP_BUF 是按照 warp 请求的先后对 warp 号码进行排队,深度为 8,宽度为 4.其中低三位为 warp 号码,最高位表示该 warp 对应的读操作或写操作.当接收到 finish 信号时,写指针 wptr_w 加 1,表示一个 warp 中的所有请求接收完成.根据 warp_num,判断对应的地址预处理模块中的缓存是否为空,如果不空,则将对应的读写请求按照其地址范围发往 DCACHE、SRAM 或者 DDR.当前 warp 中的所有请求处理完成时,

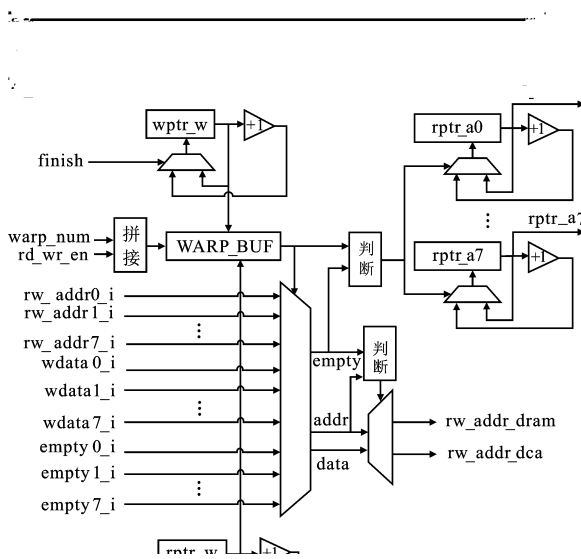


图 5 请求发送模块结构图

rw_done 信号拉高,表示该 warp 的所有请求处理完成,WARP_BUF 对应的读指针加 1. rptra_a0~rptra_a7 分别为地址预处理模块中 8 个地址缓存的读指针,在根据 warp_num 成功选择并发送一个请求后,判断对应模块中的缓存是否为空,如果不空,则对应读指针加 1.

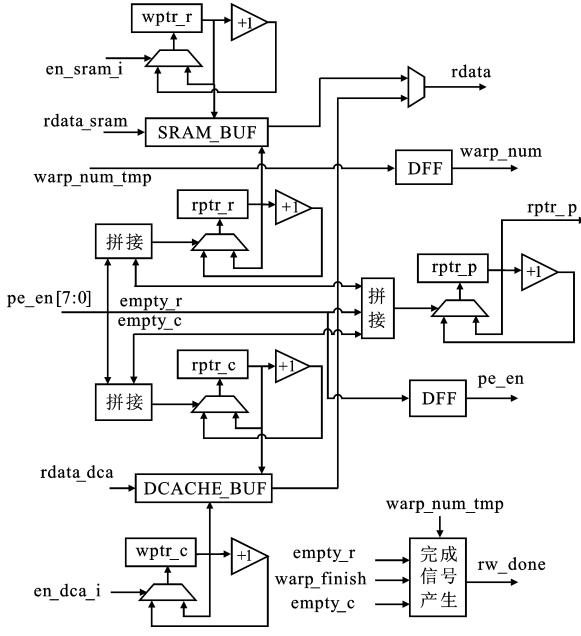


图 6 数据接收模块结构图

数据接收模块主要负责将处理器从上级存储中读取的数据进行缓存,并将数据依次发送至处理器运算单元中.如图 6 所示为接收数据结构图. SRAM_BUF、DCACHE_BUF、DDR_BUF 为数据缓存,分别缓存从 SRAM、DCACHE、DDR 中读取的数据.

在数据读取成功时,判断此数据属于 DCACHE、SRAM 还是 DDR,并将数据缓存到对应的 BUF 中.当前 warp 的所有请求发送完成后,将数据按照请求的顺序发往处理器运算单元处.并输出对应的 warp 号码,pe 使能,以及读写完成信号.

3. 2. 2 指令 CACHE

指令 cache^[7]主要作用是提供处理器操作所需要的指令.该 cache 选择 16 路组相联的方式,容量大小为 8k 字,行大小为 128 个字,采用 lru 替换算法.指令 cache 可以分为指令 cache 的存储体以及指令 cache 的控制器两部分,存储体用来存储可提供处理器运行的指令以及标志位,控制器则对整个 cache 进行控制.

Cache 存储矩阵包含 16 块指令存储体和 16 块指令标记(tag)存储体,每一块都具有相同的结构,分别对应 16 路中的每一路.其中指令存储体每块大小为 2KB(分成 4 行,每行有 512 个字节),总共为 32KB.指令 tag 存储体每块分成 4 行,每行有 22 位.访问地址为 32 位,其中低 9 位表示行内偏移,9~10 位表示索引位,选择对应行,高 21 位表示 tag 标志位.

Cache 控制器采用三级流水线设计,第一级流水线对地址进行解码,根据解码地址将 16 路中对应行的 tag 和指令取出.第二级流水线通过比较 tag 位判断是否命中,若命中,将命中路数传至第三级流水线;若缺失,向总线发送缺失请求,暂停流水线,等待 cache 填充.三级流水线主要根据命中情况选择将对应的指令输出至处理器.

3. 2. 3 数据 CACHE

数据 cache^[7]为一块只读的 cache,主要负责缓存处理器操作所需要的数据.该 cache 选择 4 路组相联的方式,容量大小为 32KB,行大小为 8 个字,采用 lru、伪 lru 和 lfu 三种可配置式替换算法.其结构同样分为存储体和控制器两部分.

数据 cache 的存储体与指令 cache 存储体类似,由于采用 4 路组相联,将存储矩阵设计为 4 块数据存储体和 4 块数据 tag 存储体,并且各自拥有相同的结构.其中,数据存储体每块大小为 8KB(分成 256 行,每行有 32 个字节),总共 32KB.数据 tag 存储体每块分成 256 行,每行有 19 位.访问地址为 32 位,其中低 5 位表示行内偏移,5~12 位表示索引位,选择对应行,高 19 位表示 tag 标志位.

控制器模块主要使用状态机完成,命中时读取数据只需一个时钟,该状态机有 3 个状态,具体如下

所示.

IDLE:初始态.在该状态中进行 cache 的读命中判断,如果命中,则将对应的命中数据下发,否则跳转至 MISS 状态.

MISS:读缺失态.在本状态中,更新替换算法,同时向上级存储发出访存请求,待总线应答后,跳转至 REPL 状态.

REPL:重装状态.该状态负责 cache 行的重新填充,将上级存储返回的数据写入 cache 数据存储体,并将处理器需要的数据下发,然后返回初始态.

3. 2. 4 共享存储

共享存储由普通可读可写 RAM 构成,空间大小为 4096 * 4Bytes.其主要负责 64 个线程的共享数据的保存和加载.由于本文所设计的处理器是基于 SIMT 架构的,其各个线程间无法直接交换数据,所以通过共享存储实现各个线程之间数据的交互,并且共享存储的实现还可减少公用数据的传输或下发.

4 仿真验证

4. 1 验证平台

将本设计中的存储系统、另外的处理器控制系统、运算处理单元整合为完整的 SIMT 处理器,然后使用基于 FPGA 的验证平台验证模块功能的正确性.FPGA 采用 Xilinx 的 Virtex UltraScale XCVU-440,互联总线采用 2v2 的 AXI4 总线,DW 为异步转换核,主要负责匹配 DDR 与处理器的速度.

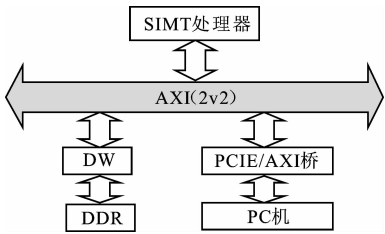


图 7 验证平台结构图

在该验证平台中,首先通过 PC 机将待测程序和待测数据写入 DDR 中,并将处理器所需的初始配置信息发送至处理器的配置寄存器中.然后通过 FPGA 开发板上的按键启动处理器,程序运行完成后将数据写回至 DDR 中,通过 PC 机读取 DDR 中的数据检验结果是否正确.

4. 2 结果分析

通过 Vivado 工具中的 chipscope 抓取测试程序验证过程中存储系统模块的部分波形,如图 8 所示,

