

基于 OR1200 的批处理协处理器设计

金 鹏, 杨 刚, 胡 耀

(北京遥感设备研究所, 北京 100854)

摘 要: 基于开源 OpenRISC 软核处理器 OR1200, 以其自主设计的指令集作为基础, 通过增改处理器内核内部设计, 设定了一个新建指令 l. cp. OR1200 通过指令 l. cp 完成对协处理器的调用, 将操作数和执行码发送至协处理器, 暂停 CPU 的指令流水线. 协处理器自行完成对向量数据的批处理后, 停止对总线的占用, 并使 OR1200 的处理器内核恢复正常的指令处理流程. 协处理器设定的功能为完成单边选大恒虚警检测算法. 仿真结果验证了设计的正确性.

关键词: 片上系统; OR1200; 协处理器; 批处理; 恒虚警检测

中图分类号: TN492

文献标识码: A

文章编号: 1000-7180(2014)10-0039-04

Design of Sequential Processing Co-processor Based on OR1200

JIN Peng, YANG Gang, HU Yao

(Beijing Institute of Remote Sensing Equipment, Beijing 100854, China)

Abstract: This design is based on the open source soft-core processor OR1200 with its unique instruction set. With the modification to the interior core module of the CPU, a new instruction is built as l. cp. Through calling the instruction, CPU triggers a co-processor, meanwhile sends data and op-code to it. By the same time, CPU suspends its own instruction pipeline. After finishing sequential data processing, the co-processor stops the BUS occupation, which resumes the suspended pipeline. The co-processor realizes GO-CFAR algorithm. And the corresponding simulation result verified the design.

Key words: System-on-Chip; OR1200; co-processor; sequential processing; CFAR

1 引言

高端的嵌入式系统在高层次进行了集成, 将一部分功能元件纳入到固定的芯片中. 这种设计方法固然有很多优势, 但一个显著的缺点就是有太多的架构被预设, 这对于特定目标的应用来说并非最佳选择^[1]. 基于此, 自定义的片上系统(System-on-a-Chip, SoC)被越来越多的设计师所采纳.

通过对典型的五级整数流水线 RISC 处理器的分析可得出, 循环逻辑引起的指令跳转会导致指令流水线长时间的停滞, 造成指令执行时间过长的现象^[2]. 批处理是这个问题的一个典型情况. 在通常的设计中, 批处理的执行方式是通过 CPU 寻址, 将 RAM 中的向量数据读取至 CPU 寄存器, 通过循环逻辑进行累加累乘等算术运算. 之后将结果储存至

RAM 的目标地址空间, 完成针对向量数据的批处理. 虽然设计师可以通过软件程序优化(如将循环逻辑代码有限的展开, 减少循环次数)提升效率, 但无法本质上解决频繁使用跳转指令的问题^[3-4].

具有批处理功能的协处理器提供了这种问题的解决方法, 通过数字电路“面积换时间”的博弈, 为成熟且固定的算法实现提供了硬件解决方案. 自定义指令提供的软件接口, 增加了算法的可配置性和调用的即时性. 并且对于可设计为流水线执行的算法(如恒虚警检测)时, 有更好的效率提升.

2 SoC 系统设计

包含协处理器(co-processor, CP)的 SoC, 基于软核处理器 OR1200 设计, 由增改后的 OR1200 模块、CP 模块、RAM 和交互式总线 WB_CONMAX

组成,其系统框架图如图 1 所示. CP 接收 OR1200 传输的 4 组信号(. opcode 执行码、. src_addr 源向量首地址、. dst_addr 目的向量首地址,以及. num 批处理向量长度). 这些信息在自定义指令 l. cp 的译码阶段会传递至模块 OR1200 与 CP 间的接口.. cp_stall 是 CP 输出至 OR1200 的信号,其置高时 OR1200 的 CPU 指令流水会处于暂停状态. OR1200 采用了 Silicore 公司提供的 Wishbone (WB) 开放性总线标准,包括一个数据 WB 接口 m_0 和一个指令 WB 接口 m_1 . CP 的数据总线连接到 WB_CONMAX 的 m_2 接口,作为第三个主设备. RAM 作为指令代码和数据向量的存储器,连接至 WB_CONMAX 的从设备接口 s_0 .

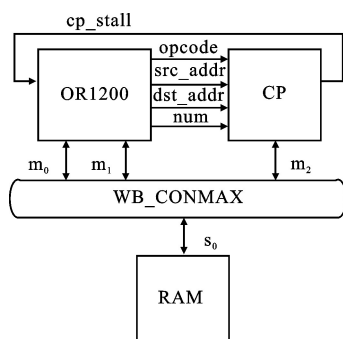


图 1 系统架构框图

3 l. cp 自定义指令

OR1200 采用 OpenRISC-1000 体系结构和 ORBIS32 指令集. 其指令宽度均为 32 位,格式分为双操作数、单操作数和无操作数三类. 指令的前两位用来产生执行单元操作所需的寄存器选择控制信号,即 11 表示两个操作数均来自寄存器,10 和 01 表示有一个来自寄存器另一个来自立即数或其他,00 表示均不来自寄存器.

OR1200 提供了 8 个自定义指令,为 l. cust1 至 l. cust8. 由于 CP 并不包含在 OR1200 的 CPU 算术运算器 ALU 内,并且需要双操作数,所以选择其他自定义指令作为 CP 的软件接口;因为 CP 通过执行码识别指令 l. cp,并将其作为运行的使能,所以需确保指令本身有唯一的执行码. 基于此,本设计的协处理器工作指令 l. cp r3, r1, r2 设定如表 1 所示.

表 1 l. cp 指令结构

31-26	25-21	20-16	15-11	1-0	9-8	7-4	3-0
Opcode	rD	rA	rB	Reserved	Opcode	Reserved	Opcode
0x3D	0x3	0x1	0x2	0	0x3	0	0x6

上述指令 l. cp r3, r1, r2 的十六进制表达为 0xf4611306. 指令前两位为 11,允许 CPU 读取寄存器文件(RF)中的两个寄存器作为操作数. 指令中设立目的寄存器 rD,但由于实际设计中将 l. cp 的回写使能抑制,所以并未实际使用,本设计中只作为预留.

批处理 CP 的软件接口需要有如下三项数据:源向量数据首地址、目的向量数据首地址和向量长度,因此需对源寄存器 rA 设计一定程度的复用. 高 16 位为目的数据向量首地址的[15:8]位,低 16 位为源数据向量首地址的[15:8]位. 由于指令最高可设定向量地址的第 16 位,所以可批处理的数据最大长度为 65 536 个字. rB 存储的数据设置为批处理操作数据长度. 举例说明,如表 2 所示.

表 2 rA、rB 设置对应关系

寄存器	存储数据	数据意义
rA	0x1008	源数据向量首地址为 0xXXXX1000
		源数据向量首地址为 0xXXXX0800
rB	0x200	批处理数据长度为 512 点

4 针对 OR1200 CPU 的增改

本设计针对 CPU 增加了新建指令并修改了相关逻辑,但并没有对体系结构做出修改. 对 OR1200 CPU 的外环境而言,只增加 CPU 与 CP 交互的数据信号,连接至 OR1200 与 CP 的接口处,如图 1 所示.

OR1200 的 CPU 以 CTRL(控制器)为逻辑控制主单元,控制指令流水进程. CPU 按五级指令流水设计,模块清晰. 各模块在指令各个的流水阶段有着明确的分工. 各模块及其定义如图 2 所示^[5].

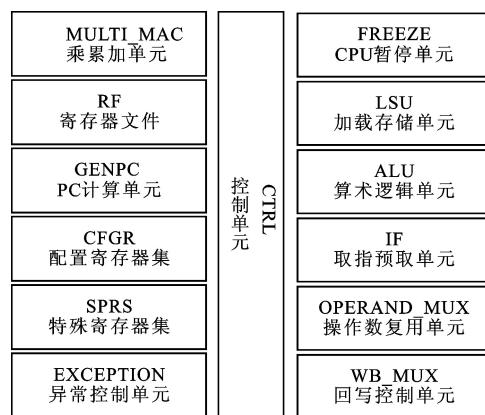


图 2 OR1200 CPU 内部模块

针对 CPU 的改动需要达到如下目的:首先, CPU 需要在译码阶段将 rA 和 rB 的数据读出,并通

过 OPERAND_MUX(操作数复用单元)选择,由后者将两个源寄存器的数据输出,作为 CP 的操作数^[6];其次,执行阶段,CP 在判定由 CPU 发出的执行码后,进入数据批处理运行状态.最后,由于 OR1200 为五级指令流水结构,所以当 CP 进入长时间的数据批处理运行阶段时,需要解决后续指令如何处理的问题.对应的方案,诸如可设置流水线照常进行,当 CP 完成数据批处理运行后,发送信号致 CPU 中断,CPU 被触发处理中断事件;或者可设置将 CPU 指令流水线暂停,等待 CP 完成数据批处理,之后由 CP 恢复 CPU 的指令流水.本设计选择后一种方法.基于此,本设计需要针对控制模块 CTRL 进行围绕新建指令和流水暂停的逻辑增改,主要的内容如表 3 所示.

表 3 对 CPU CTRL 模块的主要逻辑增改

序号	改动部分	作用阶段	改动目的
1	$alu_op \leq \{OR1200_ALUOP_CP\}$	译码阶段	在执行阶段将执行码发送至协处理器
2	$wait_on = \{OR1200_WAIT_ON_MULTMAC\}$	译码阶段	借用串行乘法的流水线暂停信号,在执行阶段的第一个时钟周期暂停流水线,后续的流水线暂停需要 CP 的相关信号支持
3	$sel_imm \leq 1'b0$	取指阶段	非立即数操作指令
4	$except_illegal \leq 1'b0$	译码阶段	不做非法处理
5	$rwb_op \leq \{OR1200_RWBOP_NOP\}$	译码阶段	不做回写处理

当指令 l. cp 进入执行阶段, FREEZE 模块因为 wait_on 的输入做暂停流水线处理,即将模块输出信号 if_freeze、id_freeze、ex_freeze 和 wb_freeze 全部置为有效,即令指令流水线各个环节(取指、译码、执行以及回写)均进入暂停状态.这种暂停状态会保持一个时钟周期.与此同时,CP 进入执行阶段,并且在下一个时钟周期开始,置流水线暂停信号 cp_stall 为有效,并保持其状态,这个信号会连接到 FREEZE 模块的输入端口. mac_stall,其作用是将流水暂停状态保持至 CP 批处理结束.当 CP 完成数据批处理后,将输出信号 cp_stall 置为无效, FREEZE 模块会因此在下一个时钟周期将四个上述 xx_freeze 信号置为无效,当 cp_stall 无效后的第二个周期,指令流水线恢复运行.其系统架构如图 1 所示,时序关系图 5、图 6 所示.

5 GO-CFAR 协处理器设计

本设计中的 CP 是完成 GO-CFAR 算法的数字电路.当 l. cp 的执行码输入 CP 后,CP 通过 WB_CONMAX^[7]交互总线对 RAM 中的向量数据进行读写批处理操作.在一个读写周期内,CP 将读出的数据串行输入 GO-CFAR 算法模块,并将后者流水输出的串行数据作为结果,写入 RAM 目标地址.每个读写周期结束后,CP 会依据指令提供的源向量首地址、目的向量首地址和操作长度,递增发送至 RAM 的读地址和写地址,完成数据的批处理.

CFAR 处理是雷达信号处理的重要组成部分之一,目的是在有杂波干扰的环境中,保持信号检测时的虚警率恒定,而使数据处理终端不致因虚警过多而超载. GO-CFAR 结构示意图见图 3.

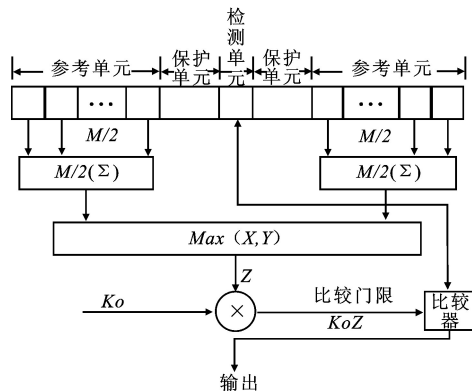


图 3 GO-CFAR 算法示意图

通过串行输入输出数据,将其算法实现进行流水优化,提高 CP 的工作效率.优化后的设计将待检测的数据向量串行输入至一个存储队列中,队列的长度为单边参考单元、保护单元以及检测单元的长度和.其中参考单元的队列相加,得出单边参考单元和,输入单边参考单元和队列中,后者队列的长度为双边保护单元、检测单元加二后的值^[8].参考单元和队列两端的数据选大,完成单边选大操作,其值为 Z.将后者与门限系数相乘,并与检测单元比较,得出恒虚警检测的结果,作为输出.上述实现方法如图 4 所示.

6 仿真结果

将输入源数据向量的首地址设定为 0x800,目的向量的首地址设定为 0x1000,进行 512 点的 GO-CFAR 处理.通过 ISim 仿真软件生成的时序图,可以观测并检验指令 l. cp 暂停与恢复 OR1200 流水线的功能.通过将仿真工具 MATLAB 可以生成

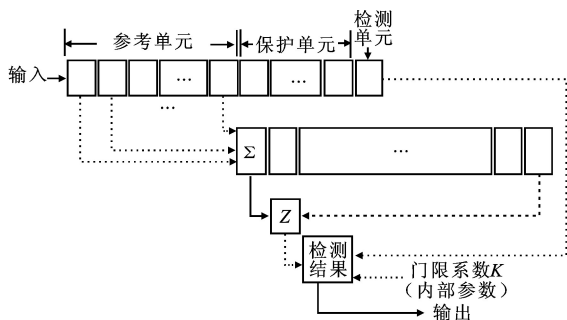


图4 优化后的 GO-CFAR 实现方法

待检测的数据向量,并通过其画图功能验证写入 RAM 的 CP 处理结果的正确性。l_cp 暂停流水线的仿真结果如图 5 所示。

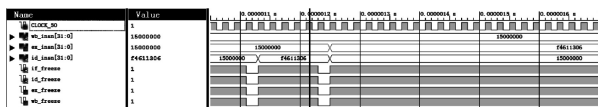


图5 流水线暂停的时序图

图 5 中指令 l_cp (即 0xf4611306) 进入执行阶段时,OR1200 内部模块 FREEZE 的输出信号 if_freeze、id_freeze、ex_freeze、wb_freeze 全部置高,此时 OR1200 的 CPU 处于指令流水线暂停状态。l_cp 恢复流水线的仿真结果如图 6 所示。

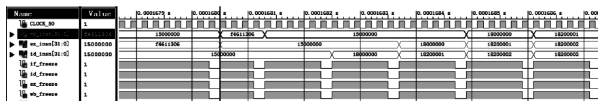


图6 流水线恢复的时序图

图 6 中指令 l_cp 在 xx_freeze 信号置低后进入回写阶段(尽管并不进行回写操作),执行阶段处理下一条指令 l_nop (0x15000000)。CPU 的指令流水线得到了恢复。CP 完成 GO-CFAR 的输入仿真数据与输出结果的比较如图 7 所示。

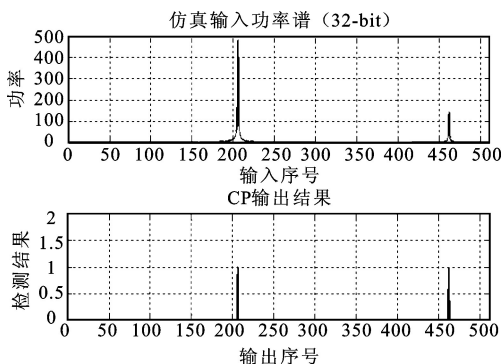


图7 CP 输入与输出信号仿真结果

7 结束语

本设计完成了对 OR1200 的改动,增加了支配协处理器的指令,并通过配置流水线相关信号,使新建指令具有暂停流水线的功能。协处理器在接到指定执行码之后,输出特定信号,继续暂停 OR1200 指令流水线,以待其完成批处理操作。协处理器实现了单边选大恒虚警检测处理,通过流水线优化提高了算法实现效率。在批处理完成后,恢复 OR1200 的指令流水线,进行后续指令代码的处理。设计通过 ISim 仿真软件完成综合前仿真,经验证,OR1200 指令流水线暂停、恢复均达到目的。经仿真软件 MATLAB 产生的待检测数据向量与写入 RAM 的检测结果输出向量相比较,协处理器处理结果正确。仿真结果佐证了设计的正确和可行性。

参考文献:

- [1] Ron Sass, Andrew G Schmidt. FPGA 嵌入式系统设计原理与实践[M]. 北京:清华大学出版社, 2012.
- [2] 何荣森,何希顺,张跃. 从 ARM 体系看嵌入式处理器的发展[J]. 微电子学与计算机, 2002, 19(5): 42-45.
- [3] 邵洋,刘鸿瑾. 专用指令集处理器系统级设计方法[J]. 微电子学与计算机, 2007, 24(7): 32-34.
- [4] 雷思磊. 步步惊“芯”: 软核处理器内部设计分析[M]. 北京:电子工业出版社, 2013.
- [5] Damjan Lampret, Julius Baxter. OpenRISC 1200 IP core specification (preliminary draft) v0.13[S]. 2012.
- [6] Damjan Lampret, Marko Mlinar, Jeanne Wiegmann, et al. OpenRISC 1000 architecture manual[S]. 2006.
- [7] Rudolf Usselman. WISHBONE interconnect matrix IP core (Rev:1.1)[S]. 2002.
- [8] 李洪涛,朱晓华,顾陈. Verilog HDL 与 FPGA 开发设计及应用[M]. 北京:国防工业出版社, 2013.

作者简介:



金鹏男, (1984-), 硕士, 工程师。研究方向为信号处理技术、嵌入式系统开发技术。