基于 FPGA 的八位 RISC CPU 的设计

张杰

摘　要：从 CPU 的总体结构到局部功能的实现采用了自顶向下的设计方法和模块化的设计思想，利用 Xilinx 公司的 Spartan II 系列 FPGA，设计实现了八位 CPU 软核。在 FPGA 内部不仅实现了 CPU 必需的算术逻辑器、寄存器堆、指令缓冲、跳转计数、指令集，而且针对 FPGA 内部的结构特点对设计进行了地址和数据的优化。

关键词： Verilog, RISC CPU, FPGA，可编程逻辑

1 引　言

随着数字通信和工业控制领域的高速发展，要求专用集成电路（ASIC）的功能越来越强，功耗越来越低，生产周期越来越短，这些都对芯片设计提出了巨大的挑战，传统的芯片设计方法已经不能适应复杂的应用需求了。SoC（System on a Chip）以其高集成度，低功耗等优点越来越受欢迎。开发人员不必从单个逻辑门开始去设计 ASIC，而是应用已有 IC 芯片的功能模块，称为核（core），或知识产权（IP）宏单元进行快速设计，效率大为提高。CPU 的 IP 核是 SoC 技术的核心，开发出具有自主知识产权的 CPU IP 核对我国在电子技术方面跟上世界先进的步伐，提高信息产业在世界上的核心竞争力有重大意义。

精简指令集计算机 RISC（Reduced Instruction Set Computer）是针对复杂指令集计算机 CISC（Complex Instruction Set Computer）提出的，具备如下特征 1）一个有限的简单的指令集； 2）强调寄存器的使用或 CPU 配备大量的能用的寄存器；3）强调对

指令流水线的使用。

2 CPU IP核的组成

尽管各种CPU的性能指标和结构细节不同，但所要完成的基本功能相同，从整体上可分为八个基本的部件：时钟发生器、指令寄存器、累加器、RISC CPU算术逻辑运算单元、数据控制器、状态控制器、程序控制器、程序计数器、地址多路器。状态控制器负责控制每一个部件之间的相互操作关系，具体的结构和逻辑关系如图1所示。

时钟发生器利用外部时钟信号，经过分频生成一系列时钟信号给CPU中的各个部件使用。为了保证分频后信号的跳变性能，在设计中采用了同步状态机的方法。

指令寄存器在触发时钟clk1的正跳变触发下，将数据总线送来的指令存入寄存器中。数据总线分时复用传递数据和指令，由状态控制器的load_ir信号负责判别。load_ir信号通过使能信号ena口线输入到指令寄存器。复位后，指令寄存器被清为零。每条指令为两个字节16位，高3位是操作码，低13位是地址线。CPU的地址总线为是13位，位寻址空间为8K 字节。本设计的数据总线是8位，每条指令取两次，每次由变量state控制。

累加器用于存放当前的运算结果，是双目运算中的一个数据来源。复位后，累加器的值为零。当累加器通过使能信号ena 口线收到来自CPU状态控制器load_acc 信号后，在clk1时钟正跳沿时就接收来自数据总线的数据。

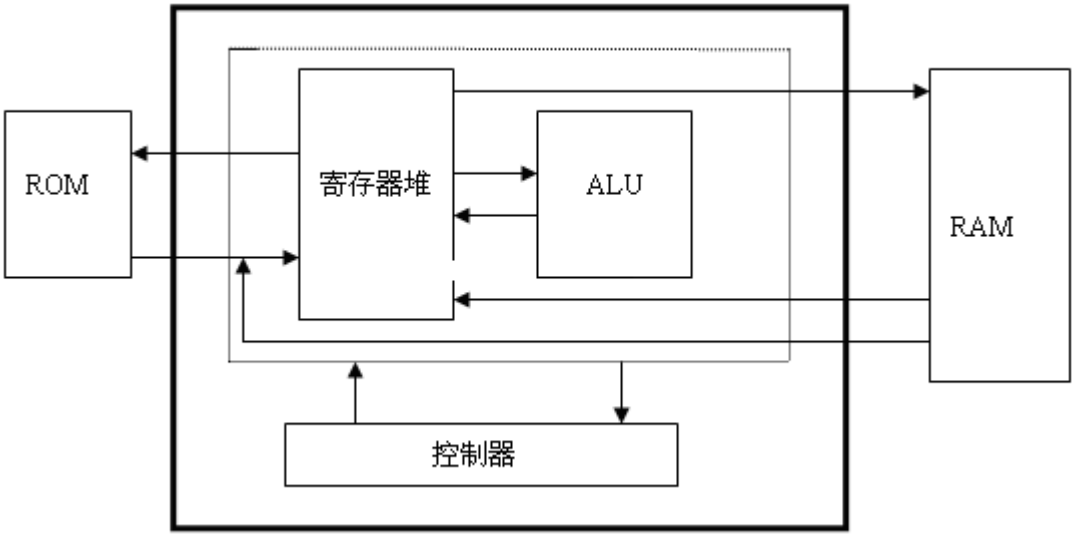

图 1 CPU 结构图

算术逻辑运算单元根据输入的不同的操作码分别实现相应的加、与、异或、跳转等基本运算。

数据控制器其作用是控制累加器的数据输出，由于数据总线是各种操作传送数据的公共通道，分时复用，有时传输指令，有时要传送数据。其余时候，数据总线应呈高阻态，以允许其他部件使用。所以，任何部件向总线上输出数据时，都需要一个控制信号的，而此控制信号的启、停则由 CPU 状态控制器输出的各信号控制决定。控制信号 data ctl_ena 决定何时输出累加器中的数据。

地址多路器用于输出的地址是 PC（程序计数器）地址还是数据/端口地址。每个指令周期的前 4 个时钟周期用于从 ROM 中读取指令，输出的应是 PC 地址，后 4 个时钟周期用于对 RAM 或端口的读写，该地址由指令给出，地址的选择输出信号由时钟信号的 8 分频信号 fecth 提供。

程序计数器用于提供指令地址，以便读取指令，指令按地址顺序存放在存储器中，有两种途径可形成指令地址，一是顺序执行程序的情况，二是执行 JMP 指令后，获得新的指令地址。

状态机控制器接受复位信号 RST，当 RST 有效时，能通过信号 ena 使其为 0 ，输入到状态机中以停止状态机的工作。状态机是 CPU 的控制核心，用于产生一系列的控制信号，启动或停止某些部件，CPU 何时进行读指令来读写 I/O 端口及 RAM 区等操作，都是由状态机来控制的。状态机的当前状态，由变量 state 记录，state 的值就是当前这个指令周期中已经过的时钟数。指令周期是由 8 个时钟组成，每个时钟都要完成固定的操作。

3 系统时序

RISC CPU 的复位和启动操作是通过 rst 引脚的信号触发执行的，当 rst 信号一进入高电平，RISC CPU 就会结束现行操作，并且只要 rst 停留在高电平状态，CPU 就维持在复位状态，CPU 各状态寄存器都设为无效状态。当信号 rst 回到低电平，接着到来的第一个 fetch 上升沿将启动 RISC CPU 开始工作，从 ROM 的 000 处的开始读取指令并执行相应的操作。

读指令时序，每个指令的前 3 个时钟周期用于读指令，4~6 周期读信号 rd 有效，第 7 个周期读信号无效，第 8 个周期地址总线输出 PC 地址，为下一个指令作准备。

写指令时序，每个指令的第 3.5 个时钟周期建立写地址，第四个周期输出数据，第 5 个时钟周期输出写信号，第 6 个时钟结束，第 7.5 个时钟周期输出为 PC 地址，为下个指令做准备。

如图 2 所示，这是 ModelSim SE6.0 进行波形仿真的结果。

4 微处理器指令

数据处理指令：数据处理指令完成寄存器中数据的算术和逻辑操作，其他指令只是传送数据和控制程序执行的顺序．因此，数据处理指令是唯一可以修改数据值的指令，数据处理指令一般需两个源操作数，产生单个结果．所有的操作数都是 8 位宽，或者来自寄存器，或者来自指令中定义的立即数．每一个源操作数寄存器和结果寄存器都在指令中独立的指定。

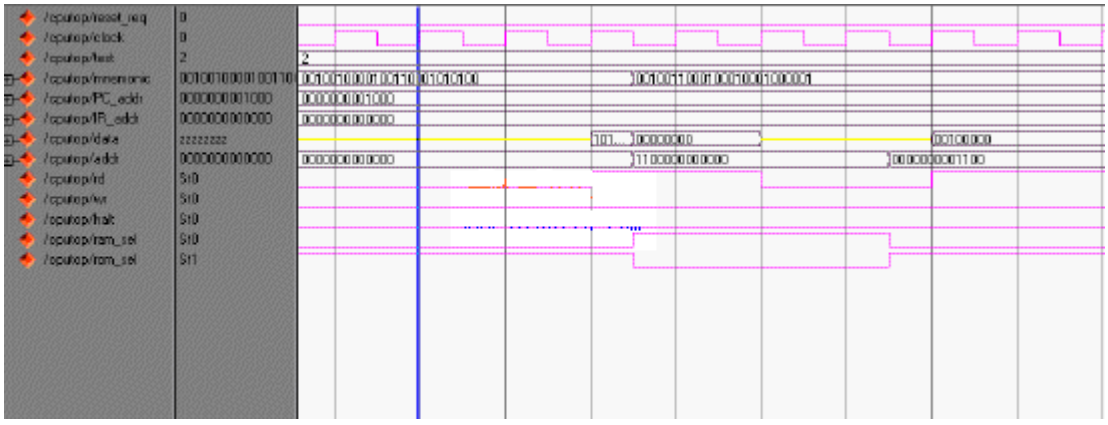

图 2 读写指令时序

数据传送和控制转移类指令：共有 17 条，不包括按布尔变量控制程序转移的指令。其中有全存储空间的长调用、长转移和按 2KB 分块的程序空间内的绝对调用和绝对转移；全空间的长度相对转移及一页范围内的短相对转移；还有条件转移指令。这类指令用到的助记符有 ACALL, AJMP, LCALL, LJMP, SJMP, M, JZ, JNZ, ONE,DJNZ。控制转移类指令主要用来修改 1x 指针从而达到对程序流的控制，所用到的寄存器主要有 sp, pc, ir 等寄存器。

指令由操作码和操作数组成，取指令电路的目的就是把指令码和操作数分开。组成电路由如图 3 所示。取指令电路由程序指针，程序指针解析模块、ROM, IR(指令寄存器)，控制器状态寄存器组成。取指令指令的过程如下:PC 指针的值经过 pc_mux 模块赋值，把 ROM 中的指令取出来，送到指令寄存器的数据输入口。指令寄存器受状态寄存器的控制，当取指令信号有效时，ROM 中的指令码被保存在指令寄存器中，然后经控制器译码，产生控制信号，对 PC 指针的增量加以控制取出下一条指令。

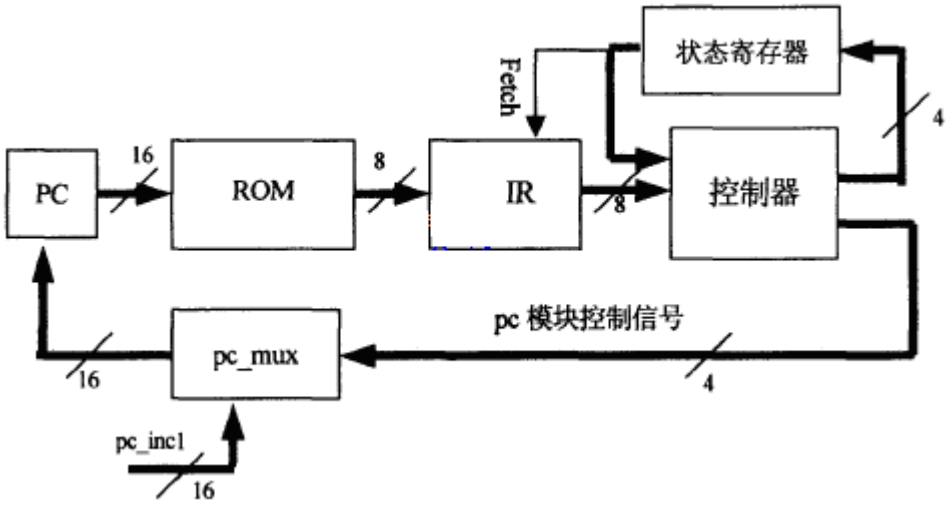

图 3 取指令电路

5 汇编

汇编程序是为了调试软核而开发的，手工编写机器码很容易出错并且工作量很大。在调试过程中修改指令集时，汇编程序也要作相应的修改。所以要求编译器的结构简单性能可靠，在程序中必要的地方可以用堆叠代码方法实现，不必考虑编程技巧和汇编器效率问题。汇编程序用于测试 RISC CPU 的基本指令集，如果 CPU 的各条指令执行正确，停止在 HLT 指令处。如果程序在其它地址暂停运行，则有一个指令出错。程序中，@符号后的十六进制表示存储器的地址，每行的//后表示注释。下面是一小段程序代码，编译好的汇编机器代码装入虚拟 ROM，要参加运算的数据装入虚拟 RAM 就可以开始进行仿真。

```
机器码 地址 汇编助记符 注释
@00 //地址声明
101_11000 //00 BEGIN: LDA DATA_2
0000_0001
011_11000 //02 AND DATA_3
0000_0010
100_11000 //04 XOR DATA_2
0000_0001
001_00000 //06 SKZ
0000_0000
000_00000 //08 HLT //AND doest work
```

6 调试

最基本的调试手段是基于 FPGA 厂商提供的开发和仿真环境,用硬件描述语言编写 TESTBENCH，构成一个最小运行环境。TESTBENCH 产生对目标软核的激励，同时记录软核的输出，和预期值进行比对，可以确定核的设计错误。这种方法的好处是实现容易，结果准确，但硬件描述语言编码量较大。为了仿真结果的准确性，无论功能仿真还是时序仿真，仿真的步长都不能太小，结果导致整个系统仿真时间太长。本设计中先对 RISC CPU 的各个子模块进行了分别综合，检查正确性，如果发现错误可以在较小的范围内来检查并验证。子模块综合完毕后，把要综合的 RISC CPU 的模块与外围器件以及测试模块分离出来组成一个大模块，综合后的的 RISC CPU 模块如图 4 所示，这是 Xilinx ISE7.1 所综合生成的技术原理图。

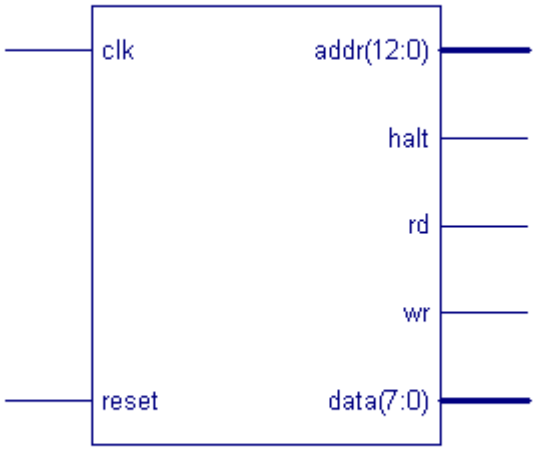

图 4 CPU 技术原理图

综合的结果只是通用的门级网表，只是一些与、或、非门的逻辑关系，和芯片实际的配置情况还有差距。此时应该使用 FPGA/CPLD 厂商提供的实现与布局布线工具，根据所选芯片的型号，进行芯片内部功能单元的实际连接与映射。这种实现与布局布线工具

一般要选用所选器件的生产商开发的工具，因为只有生产者最了解器件内部的结构，如在 ISE 的集成环境中完成实现与布局布线的工具是 Flow Engine。

STA（Static Timing Analysis）静态时序分析，完成 FPGA 设计时必须的一个步骤。在 FPGA 加约束、综合、布局布线后，在 ISE 中可以运行 Timing Analyzer 生成详细的时序报告，本设计中 Minimum period: 12.032ns (Maximum Frequency: 83.112MHz)，Minimum input arrival time before clock: 6.479ns，Maximum output required time after clock: 9.767ns。然后，设计人员检查时序报告，根据工具的提示找出不满足 Setup/Hold time 的路径，以及不符合约束的路径，进行修改保证数据能被正确的采样。在后仿真中将布局布线的时延反标到设计中去，使仿真既包含门延时，又包含线延时信息。这种后仿真是最准确的仿真，能真实地反映芯片的实际工作情况。

7 结 论

复杂的 RISC CPU 设计是一个从抽象到具体的过程，本文根据 FPGA 的结构特点，围绕在 FPGA 上设计实现八位微处理器软核设计方法进行探讨，研究了片上系统的设计方法和设计复用技术，并给出了指令集和其调试方法，提出了一种基于 FPGA 的微处理器的 IP 的设计方法。本文作者创新点是：根据 Spartan II 的内部结构，在编码阶段实现了地址和数据的优化，实现阶段对内部布局布线进行重新配置，设计实现的微处理器仅占用 78 个 slices，1 个 Block RAM，在 10 万门的芯片实现，占用 6%的资源。

参考文献：

[1] 夏宇闻.Verilog数字系统设计教程[M].北京：北京航空航天大学出版社，2003.

[2] 袁俊泉 孙敏琪 曹瑞 Verilog数字系统设计教程[M].西安：西安电子科技大学出版社，2002.

[3] J.Bhasjer 著 孙海平 等译 Verilog HDL综合实用教程[M].北京：清华大学出版社，2004.

[4] 杨厚俊 张公敬 张昆藏 编著 计算机系统结构---奔腾PC[M].北京：科学出版社，2004.

[5] 冯海涛 王永纲 石江涛 颜天信 王砚方. 基于FPGA的32位整数微处理器的设计与实现[J]. 小型微型计算机系统，2005，26（6）：1113-1117.

[6] 王喆.八位CPU IP核的研究与设计[D].大连：大连理工大学，2005

[7] http://www.xilinx-china.com/

[8] 袁本荣 刘万春 贾云得 朱玉文 用Verilog HDL进行FPGA设计的一些基本方法[J].微计算机信息，2004,20(6):93-95

来源：微计算机信息