

<<开源软核处理器OpenRisc的SO>>

图书基本信息

书名：<<开源软核处理器OpenRisc的SOPC设计>>

13位ISBN编号：9787811241952

10位ISBN编号：7811241951

出版时间：2008-3

出版时间：北京航空航天大学出版社

作者：徐敏等著

页数：246

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<开源软核处理器OpenRisc的SO>>

### 内容概要

片上可编程系统（System On Programmable Chip，SOPC）已经成为嵌入式系统的发展方向。

《开源软核处理器OpenRisc的SOPC设计》介绍基于源代码开放的OpenRisc1200（以下简称OR1200）软核处理器的SOPC设计方法。

《开源软核处理器OpenRisc的SOPC设计》分为两部分，第一部分介绍OR1200软核处理器的架构和配置、Wishbone总线的标准及OR1200软核处理器软硬件开发环境的建立；第二部分以具体实例说明如何使用OR1200软核处理器完成嵌入式设计，其中包括：调试接口的实现、OR1200控制片内存储器及I/O、串口、SDRAM、外部总线、以太网、LCD及SRAM；另外还介绍如何在OR1200上运行嵌入式Linux，并针对第二部分给出部分源代码。

《开源软核处理器OpenRisc的SOPC设计》适合对SOPC或OR1200软核处理器感兴趣的初学者使用，也可作为嵌入式系统设计人员的自学用书，或作为相关专业研究生的教材和教师的教学参考书。

## 书籍目录

第1章 SOPC及常用软核处理器概述1.1 从SoC到SOPC1.3 常用软核处理器概述1.2.1 LEON系列1.2.2 Altera公司的NiosII1.2.3 OpenCores组织的OpenRisc系列第2章 OR1200软核的配置2.1 OR1200软核的架构2.2 OR1200软核的组成2.3 OR1200软核的配置第3章 Wishbone片上总线3.1 Wishbone总线概述3.2 Wishbone总线信号和时序3.2.1 Wishbone总线信号3.2.2 Wishbone总线循环3.2.3 Wishbone互连接口、结构及工作原理3.2.4 Wishbone主设备和从设备模型第4章 软件开发工具的安装和使用4.1 GNU交叉编译环境的组成和建立4.1.1 交叉编译4.1.2 binutils4.1.3 GCC4.1.4 GDB4.1.5链接描述文件4.2 make和Makefile的使用4.2.1 Makefile的基本结构4.2.2 Makefile的变量4.2.3 隐含规则4.2.4 make的命令行选项4.3 加深对Makefile的理解4.3.1 汇编语言4.3.2 C语言4.4 OR1k系列CPU的体系结构模拟器orlksim第5章 片内存储器及I/O控制器的设计5.1 FPGA内部的RAM块资源5.1.1 RAM块的使用5.1.2 Cylonell的RAM块5.1.3 单口RAM块的描述方法5.1.4 简单双口RAM块的描述方法5.1.5 单口ROM块的描述方法5.2 I/O控制器的结构和功能5.2.1 通用I/O控制器5.2.2 最简I/O控制器5.3 ORP概念及其定义5.4 设计与Wishbone兼容的RAM和ROM模块5.4.1 RAM模块5.4.2 ROM模块5.5 最简I/O控制器及综合结果分析5.5.1 最简I/O控制器5.5.2 综合结果分析5.6 最小系统的建立、编译和仿真5.6.1 最小系统的建立5.6.2 编写程序5.6.3 仿真第6章 Debug接口的实现6.1 JTAG原理和标准6.1.1 JTAG简介6.1.2 基本单元6.1.3 总体结构6.1.4 TAP状态机6.1.5 应用6.2 调试模块的结构及其与OR1200的连接方法6.2.1 DBGI简介6.2.2 DBGI结构6.2.3 I/O端口6.2.4 内部寄存器6.2.5 链结构6.2.6 未来发展6.3 DBGI的集成和板级功能仿真6.3.1 DBGI的集成6.3.2 板级功能仿真6.4 GDB、JTAG、GDBServer、orlksim的工作原理6.4.1 GDB6.4.2 GDB和JTAG Server6.4.3 GDB和GDBServer6.4.4 GDB和orlksim6.4.5 JTAG协议6.5 使用GDB和JTAG Server进行Debug接口的调试6.6 使用DDD进行可视化调试第7章 UART16550内核的结构和使用7.1 UART的概念、功能和发展7.2 UART的通信模式、数据格式和流控制7.2.1 通信模式7.2.2 数据格式7.2.3 流控制7.3 工业标准UART 165507.3.1 特性7.3.2 接口和结构7.3.3 寄存器7.4 兼容16550的UART IP Core7.5 OR1200的异常和外部中断处理7.6 集成带有UART的系统7.6.1 集成7.6.2 编程7.7 仿真带有UART的系统7.8 验证带有UART的系统第8章 SDRAM的时序和控制器8.1 SRAM与DRAM8.1.1 SRAM8.1.2 IS61LV256168.1.3 DRAM8.1.4 SRAM和DRAM比较8.2 SDRAM的内部结构和控制时序8.2.1 结构8.2.2 命令和初始化8.2.3 模式寄存器8.2.4 Bank行激活8.2.5 读/写时序8.2.6 自动刷新8.3 SDRAM控制器wb\_sdram8.4 集成和仿真存储系统8.4.1 存储器模型8.4.2 system\_sdram.v8.4.3 ar2000\_sdram.v8.4.4 ar2000\_sdram\_bench.v8.4.5 结构8.4.6 仿真8.5 验证存储系统第9章 外部异步总线控制器的设计9.1 异步总线控制器的结构和功能9.1.1 异步总线的组成9.1.2 异步总线的读/写时序9.2 编写异步总线控制器9.2.1 编写代码9.2.2 I/O端口9.3 异步总线控制器的仿真9.4 集成和仿真存储系统9.4.1 存储器模型9.4.2 system\_eabus.v9.4.3 ar2000\_eabus.v9.4.4 ar2000\_eabus\_bench.v9.4.5 结构9.4.6 编程9.4.7 仿真第10章 ORPMon的功能和实现10.1 C语言函数接口10.1.1 寄存器使用10.1.2 堆栈帧10.1.3 参数传递和返回值10.2 ORPMon的基本功能及其实现方法10.2.1 ORPMon10.2.2 ORPMon基本工作原理10.2.3 特殊功能寄存器操作10.3 ORPMon的移植10.3.1 源代码10.3.2 链接文件10.4 ORPMon的仿真10.5 ORPMon的运行10.6 使用Flash运行ORPMon第11章 以太网控制器的结构和Linux驱动11.1 以太网的CSMA/CD原理和MII接口11.1.1 CSMA/CD11.1.2 MII接口11.1.3 CSMA/CD的帧接收和发送过程11.2 OpenCores的以太网控制器11.2.1 以太网控制器简介11.2.2 以太网控制器的接口11.2.3 以太网控制器的寄存器11.2.4 缓冲描述符11.3 以太网控制器的内部结构11.3.1 控制器总体结构11.3.2 MII管理模块11.3.3 接收模块11.3.4 发送模块11.3.5 控制模块11.3.6 状态模块11.3.7 寄存器模块11.3.8 Wishbone接口模块11.4 嵌入式Linux简介11.5 对Linux进行配置、修改、编译、下载和运行11.6 使用ORPMon启动Linux11.6.1 设计可以启动Linux的ORPMon11.6.2 固化Linux11.7 集成以太网控制器11.7.1 system\_eth.v11.7.2 ar2000\_eth.v11.7.3验证以太网控制器第12章 LCD控制器的使用12.1 OpenCores的VGA/LCD控制器12.2 VGA/LCD控制器的接口与寄存器12.2.1 VGA/LCD控制器的接口12.2.2 VGA/LCD控制器的寄存器12.3 VGA/LCD控制器的使用方法12.3.1 视频时序12.3.2 像素色彩12.3.3 带宽需求12.4 集成和仿真VGA/LCD控制器12.5 验证VGA/LCD控制器第13章 SBSRAM的时序和控制器设计13.1 SBSRAM控制器的结构和功能13.1.1 SBSRAM的概念13.1.2 SBSRAM控制器的读/写操作和时序13.2 编写SBSRAM控制器13.3 SBSRAM控制器的仿真13.4 集成SSRAM控制器13.4.1 system\_ssram.v13.4.2 ar2000\_ssram.v13.5 验证SSRAM控制器附录 UP-SOPC2000教

<<开源软核处理器OpenRisc的SO>>

学科研平台参考文献

章节摘录

第1章 SOPC及常用软核处理器概述 SoC (System on Chip) 称为片上系统, 它是指将一个完整的功能集成在一个芯片上, SoC中包括微处理器、DSP、存储器 (ROM、RAM、Flash等)、总线以及I/O, 甚至可以包括AD/DA、锁相环等。

集成电路和系统到什么程度才算是SoC, 并没有严格的规定。

片上使用IP (Intellectual Property) 核是构建SoC的重要步骤, IP核即知识产权核或者知识产权模块, IP核在功能上已经设计并得到验证, 而且可以重复使用。

当要推出新产品时, SoC开发人员可以将原来使用过的IP核移植到新的系统中, 或者只修改一小部分电路就可以满足新的设计要求; 利用IP核的重复使用可以缩短新产品的开发周期, 降低开发难度。

例如, ARM公司的Risc架构的ARM、IBM公司的PowerPC、MIPS公司的MIPS核、Freescale公司的MCore等, 这些需要交付一定的授权费; 还有一些IP核是开源的, 可免费使用。

对于经过验证又可批量应用的系统芯片, 可以做成专用集成电路 (ASIC) 而大量生产。

而对于小批量应用就面临着高投资、高风险, 这样无法被中小企业、研究所以及大专高校等采用。

在这样的情况下, Altera公司于2000年首先提出了SOPC的概念。

SOPC是基于FPGA的可重构的SoC, 嵌入在FPGA芯片上的系统组件, 如微处理器、ROM/RAM、总线、I/O等模块, 都可以根据设计需要进行灵活的修改, 因此, SOPC是灵活的、高效的SoC解决方案

。工程师们可以自由地发挥想象力, 开发出更具特色的嵌入式产品。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>