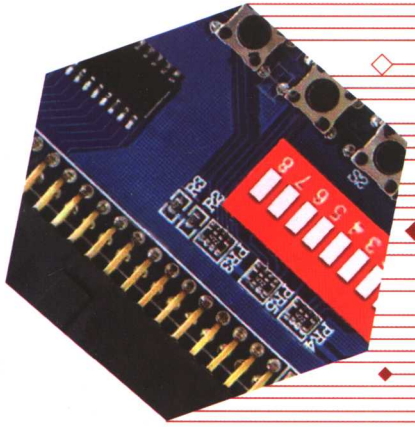
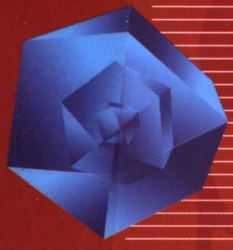
中国电子学会嵌入式专家委员会推荐丛书



开源软核处理器
OpenRisc的SOPC设计



徐 敏 孙 恺 潘 峰 编著

北京航空航天大学出版社

中国电子学会嵌入式专家委员会推荐

开源软核处理器 OpenRisc 的 SOPC 设计

徐 敏　孙 恺　潘 峰　编著

北京航空航天大学出版社

内 容 简 介

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。本书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。本书分为两部分,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 控制片内存储器和 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。

本书适合对 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。

图书在版编目(CIP)数据

开源软核处理器 OpenRisc 的 SOPC 设计/徐敏,孙恺,
潘峰编著. —北京 :北京航空航天大学出版社, 2008.3
ISBN 978-7-81124-195-2

Ⅰ. 开… Ⅱ. ①徐…②孙…③潘… Ⅲ. 微处理器—系统设计 Ⅳ. TP332

中国版本图书馆 CIP 数据核字(2008)第 007972 号

开源软核处理器 OpenRisc 的 SOPC 设计

徐 敏 孙 恺 潘 峰 编著

责任编辑 李春凤

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(100083) 发行部电话:010-82317024 传真:010-82328026

http://www.buaapress.com.cn E-mail:bhpress@263.net

涿州市新华印刷有限公司印装 各地书店经销

*

开本:787 mm×960 mm 1/16 印张:16.25 字数:364 千字

2008 年 3 月第 1 版 2008 年 3 月第 1 次印刷 印数:5 000 册

ISBN 978-7-81124-195-2 定价:28.00 元

前　言

随着集成电路工艺和电子技术的不断发展，数字集成电路经历了从电子管、晶体管、中小规模集成电路、超大规模集成电路（VLSIC）到今天的专用集成电路（ASIC）的发展过程。ASIC 虽然有着低成本、高可靠性和芯片面积小的优点；但是 ASIC 也存在着设计周期长、改版投资大和灵活性差的缺点，这些缺点制约着 ASIC 的应用范围。本着缩短设计周期、灵活地修改大规模数字逻辑的基本思想，可编程逻辑器件应运而生，目前最有代表性的可编程逻辑器件就是现场可编程逻辑门阵列（FPGA）。随着更多的数字逻辑模块被集成到 FPGA 芯片中，2000 年，FPGA 芯片生产商之一的 Altera 公司提出了 SOPC 的概念。

最近几年，FPGA 芯片得到了飞速发展，生产工艺从 90 nm 到65 nm，更大容量、更多片上资源的 FPGA 芯片相继诞生，为 SOPC 的发展提供了条件和强大的硬件支持。SOPC 发展的另外一个因素就是 IP（Intellectual Property）核，IP 核也就是已经设计并验证好的功能模块，重新设计系统时不用修改或做很少的修改就可以完成。目前 SOPC 发展最重要的部分就是设计出大量可重复使用的 IP 核，其中嵌入式软核的设计被认为是最关键的。嵌入式软核通常是用 Verilog HDL 或者 VHDL 等硬件描述语言来编写的、具有 CPU 功能的 IP 核。目前最常用的嵌入式软核有 3 个：NiosII、OpenRisc 系列和 LEON 系列。本书重点介绍 OpenRisc 系列中的 OR1200 软核处理器，它是一种源代码开放的软核，其最大的优点是免费；因此使用开源的软硬件就可以大大降低系统的成本，所以源代码开放的设计越来越得到工程师们的青睐。与 Linux 一样，开源的 OR1200 软核也必然会得到同样的重视。

编写本书主要有以下 3 个目的：

① 降低学习的门槛：2002 年 SOPC 概念进入中国，许多专家学者都认为 SOPC 具有极大的发展潜力，但是与之相关的书籍则少之又少，关于 OpenRisc 系列的更是如此。广大读者苦于没有一本合适的入门级的书籍。本书定位于 OpenRisc 初学者，降低学习的门槛。

② 使读者少走弯路：相对于付费的资源，开放源码的资源有价值的技术支持很少，所以给学习开源的读者带来很多困惑。本书以实践的角度来说明 OR1200 软核的用途，以具体的

应用实例使读者达到掌握 OR1200 的目的。使初学者少走弯路。

③ 普及 SOPC 技术：美国的 Altera 和 Xilinx 公司都已经推出了自己的软核处理器，分别是 NiosII 和 MicroBlaze。虽然它们的功能强大，开发环境和配套的 IP 完善，使得设计周期缩短，加快产品的上市，是工程应用的首选；但同时也封装了一些内部的控制机制和工作原理，这对 SOPC 的学习者和中国的 SOPC 发展很不利。编写本书的主旨是在中国普及 SOPC 技术，缩短与发达国家之间的差距。

全书共 13 章。

第 1 章主要介绍嵌入式技术从 SoC 到 SOPC 的发展，3 种常用软核的简介和比较。第 2 章分析 OR1200 的架构，并说明如何配置 OR1200 软核。第 3 章介绍 Wishbone 总线，包括信号、时序、接口、主设备和从设备模型。第 4 章介绍在 Linux 环境下的编译工具及 make、Makefile 的使用。第 5 章介绍 FPGA 芯片上 RAM 和 ROM 的使用及最小系统的组成和编译。第 6 章介绍 JTAG 的标准，如何建立 OR1200 与 JTAG 的连接。第 7 章介绍 UART16550 的结构和使用，以及用 OR1200 控制 UART。第 8 章介绍 SDRAM 的时序，以及用 OR1200 控制 SDRAM。第 9 章介绍用 OR1200 控制外部异步总线。第 10 章介绍 ORPMon 的基本功能，以及如何运行 ORPMon。第 11 章介绍以太网控制器的基本知识，Linux 的配置编译和下载，以及使用 ORPMon 启动 Linux。第 12 章介绍用 OR1200 控制 LCD。第 13 章介绍适用于 Wishbone 总线的 SBSRAM 控制器。

本书设计环境的硬件平台为：博创科技 UP－SOPC2000 开发板（详细内容见附录）。设计软件为：QuartusII 5.0 SP2、ModelSim SE 6.0、Synplify Pro8.1。

本书由厦门理工学院徐敏副教授以及孙恺、潘峰主编。徐敏编写第 5～13 章，孙恺编写第 3 章和第 4 章，潘峰编写第 1 章和第 2 章，徐敏负责全书的统稿、定稿及其他事宜。在本书的编写过程中，得到了北京航空航天大学出版社的大力支持，在此表示深深的感谢。同时北京博创科技研发部工程师黄伦学、乾正光，赵宁、刘英杰、欧阳鑫、申昆、王君、张小川、王松柏等也参与了本书部分实验的调试工作并提出了很多建议，在此一并表示感谢。

由于 SOPC 技术在中国发展时间尚短，作者的水平有限，书中难免有错误和不合适的地方，希望广大读者和嵌入式爱好者给予批评指正，我们将不胜感激。

E－mail：pf uptech@126.com

广大读者可以到 http://www.up－tech.com 的下载专区下载 OR1200 源代码及其相关内容。

作　者

2007.6

目　录

第 1 章　SOPC 及常用软核处理器概述

1.1　从 SoC 到 SOPC …… 1
1.2　常用软核处理器概述 …… 2
1.2.1　LEON 系列 …… 2
1.2.2　Altera 公司的 NiosII …… 3
1.2.3　OpenCores 组织的 OpenRisc 系列 …… 4

第 2 章　OR1200 软核的配置

2.1　OR1200 软核的架构 …… 6
2.2　OR1200 软核的组成 …… 7
2.3　OR1200 软核的配置 …… 10

第 3 章　Wishbone 片上总线

3.1　Wishbone 总线概述 …… 15
3.2　Wishbone 总线信号和时序 …… 17
3.2.1　Wishbone 总线信号 …… 17
3.2.2　Wishbone 总线循环 …… 20
3.2.3　Wishbone 互连接口、结构及工作原理 …… 28
3.2.4　Wishbone 主设备和从设备模型 …… 30

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 …… 31
4.1.1　交叉编译 …… 31
4.1.2　binutils …… 31
4.1.3　GCC …… 32
4.1.4　GDB …… 33
4.1.5　链接描述文件 …… 35
4.2　make 和 Makefile 的使用 …… 37
4.2.1　Makefile 的基本结构 …… 37

4.2.2 Makefile 的变量 …… 38
4.2.3 隐含规则 …… 39
4.2.4 make 的命令行选项 …… 40
4.3 加深对 Makefile 的理解 …… 41
4.3.1 汇编语言 …… 41
4.3.2 C 语言 …… 43
4.4 OR1k 系列 CPU 的体系结构模拟器 or1ksim …… 46

第 5 章 片内存储器和 I/O 控制器的设计

5.1 FPGA 内部的 RAM 块资源 …… 47
5.1.1 RAM 块的使用 …… 47
5.1.2 CycloneII 的 RAM 块 …… 48
5.1.3 单口 RAM 块的描述方法 …… 49
5.1.4 简单双口 RAM 块的描述方法 …… 51
5.1.5 单口 ROM 块的描述方法 …… 53
5.2 I/O 控制器的结构和功能 …… 55
5.2.1 通用 I/O 控制器 …… 55
5.2.2 最简 I/O 控制器 …… 56
5.3 ORP 概念及其定义 …… 57
5.4 设计与 Wishbone 兼容的 RAM 和 ROM 模块 …… 58
5.4.1 RAM 模块 …… 58
5.4.2 ROM 模块 …… 61
5.5 最简 I/O 控制器及综合结果分析 …… 62
5.5.1 最简 I/O 控制器 …… 62
5.5.2 综合结果分析 …… 63
5.6 最小系统的建立、编译和仿真 …… 65
5.6.1 最小系统的建立 …… 65
5.6.2 编写程序 …… 66
5.6.3 仿 真 …… 66

第 6 章 Debug 接口的实现

6.1 JTAG 原理和标准 …… 69
6.1.1 JTAG 简介 …… 69
6.1.2 基本单元 …… 69
6.1.3 总体结构 …… 70
6.1.4 TAP 状态机 …… 72
6.1.5 应 用 …… 73

6.2 调试模块的结构及其与 OR1200 的连接方法 …… 73
6.2.1 DBGI 简介 …… 73
6.2.2 DBGI 结构 …… 74
6.2.3 I/O 端口 …… 76
6.2.4 内部寄存器 …… 77
6.2.5 链结构 …… 77
6.2.6 未来发展 …… 78
6.3 DBGI 的集成和板级功能仿真 …… 80
6.3.1 DBGI 的集成 …… 80
6.3.2 板级功能仿真 …… 81
6.4 GDB、JTAG、GDBServer、orlksim 的工作原理 …… 83
6.4.1 GDB …… 83
6.4.2 GDB 和 JTAG Server …… 84
6.4.3 GDB 和 GDBServer …… 85
6.4.4 GDB 和 orlksim …… 86
6.4.5 JTAG 协议 …… 86
6.5 使用 GDB 和 JTAG Server 进行 Debug 接口的调试 …… 92
6.6 使用 DDD 进行可视化调试 …… 93

第 7 章 UART16550 内核的结构和使用

7.1 UART 的概念、功能和发展 …… 95
7.2 UART 的通信模式、数据格式和流控制 …… 96
7.2.1 通信模式 …… 96
7.2.2 数据格式 …… 97
7.2.3 流控制 …… 97
7.3 工业标准 UART 16550 …… 99
7.3.1 特 性 …… 99
7.3.2 接口和结构 …… 99
7.3.3 寄存器 …… 101
7.4 兼容 16550 的 UART IP Core …… 105
7.5 OR1200 的异常和外部中断处理 …… 106
7.6 集成带有 UART 的系统 …… 109
7.6.1 集 成 …… 109
7.6.2 编 程 …… 109
7.7 仿真带有 UART 的系统 …… 111
7.8 验证带有 UART 的系统 …… 113

第 8 章 SDRAM 的时序和控制器

8.1 SRAM 与 DRAM …… 114
8.1.1 SRAM …… 114
8.1.2 IS61LV25616 …… 115
8.1.3 DRAM …… 116
8.1.4 SRAM 和 DRAM 比较 …… 117
8.2 SDRAM 的内部结构和控制时序 …… 117
8.2.1 结 构 …… 117
8.2.2 命令和初始化 …… 121
8.2.3 模式寄存器 …… 122
8.2.4 Bank 行激活 …… 124
8.2.5 读/写时序 …… 125
8.2.6 自动刷新 …… 128
8.3 SDRAM 控制器 wb_sdram …… 129
8.4 集成和仿真存储系统 …… 130
8.4.1 存储器模型 …… 130
8.4.2 system_sdram.v …… 131
8.4.3 ar2000_sdram.v …… 132
8.4.4 ar2000_sdram_bench.v …… 133
8.4.5 结 构 …… 135
8.4.6 仿 真 …… 135
8.5 验证存储系统 …… 137

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 …… 140
9.1.1 异步总线的组成 …… 140
9.1.2 异步总线的读/写时序 …… 140
9.2 编写异步总线控制器 …… 142
9.2.1 编写代码 …… 142
9.2.2 I/O 端口 …… 144
9.3 异步总线控制器的仿真 …… 145
9.4 集成和仿真存储系统 …… 148
9.4.1 存储器模型 …… 148
9.4.2 system_eabus.v …… 148
9.4.3 ar2000_eabus.v …… 149
9.4.4 ar2000_eabus_bench.v …… 150

9.4.5 结 构 …… 153
9.4.6 编 程 …… 154
9.4.7 仿 真 …… 154

第 10 章 ORPMon 的功能和实现

10.1 C 语言函数接口 …… 156
10.1.1 寄存器使用 …… 156
10.1.2 堆栈帧 …… 157
10.1.3 参数传递和返回值 …… 158
10.2 ORPMon 的基本功能及其实现方法 …… 158
10.2.1 ORPMon …… 158
10.2.2 ORPMon 基本工作原理 …… 159
10.2.3 特殊功能寄存器操作 …… 161
10.3 ORPMon 的移植 …… 162
10.3.1 源代码 …… 162
10.3.2 链接文件 …… 167
10.4 ORPMon 的仿真 …… 171
10.5 ORPMon 的运行 …… 172
10.6 使用 Flash 运行 ORPMon …… 174

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 …… 175
11.1.1 CSMA/CD …… 175
11.1.2 MII 接口 …… 175
11.1.3 CSMA/CD 的帧接收和发送过程 …… 177
11.2 OpenCores 的以太网控制器 …… 179
11.2.1 以太网控制器简介 …… 179
11.2.2 以太网控制器的接口 …… 180
11.2.3 以太网控制器的寄存器 …… 181
11.2.4 缓冲描述符 …… 189
11.3 以太网控制器的内部结构 …… 191
11.3.1 控制器总体结构 …… 191
11.3.2 MII 管理模块 …… 191
11.3.3 接收模块 …… 192
11.3.4 发送模块 …… 194
11.3.5 控制模块 …… 196
11.3.6 状态模块 …… 196

11.3.7 寄存器模块 …… 197
11.3.8 Wishbone 接口模块 …… 198
11.4 嵌入式 Linux 简介 …… 199
11.5 对 Linux 进行配置、修改、编译、下载和运行 …… 200
11.6 使用 ORPMon 启动 Linux …… 205
11.6.1 设计可以启动 Linux 的 ORPMon …… 205
11.6.2 固化 Linux …… 206
11.7 集成以太网控制器 …… 206
11.7.1 system_eth. v …… 207
11.7.2 ar2000_eth. v …… 208
11.7.3 验证以太网控制器 …… 210

第 12 章 LCD 控制器的使用

12.1 OpenCores 的 VGA/LCD 控制器 …… 213
12.2 VGA/LCD 控制器的接口与寄存器 …… 215
12.2.1 VGA/LCD 控制器的接口 …… 215
12.2.2 VGA/LCD 控制器的寄存器 …… 217
12.3 VGA/LCD 控制器的使用方法 …… 222
12.3.1 视频时序 …… 222
12.3.2 像素色彩 …… 223
12.3.3 带宽需求 …… 224
12.4 集成和仿真 VGA/LCD 控制器 …… 225
12.5 验证 VGA/LCD 控制器 …… 230

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 …… 231
13.1.1 SBSRAM 的概念 …… 231
13.1.2 SBSRAM 控制器的读/写操作和时序 …… 231
13.2 编写 SBSRAM 控制器 …… 234
13.3 SBSRAM 控制器的仿真 …… 237
13.4 集成 SSRAM 控制器 …… 240
13.4.1 system_ssram. v …… 240
13.4.2 ar2000_ssram. v …… 242
13.5 验证 SSRAM 控制器 …… 243

附录 UP－SOPC2000 教学科研平台 …… 244

参考文献 …… 247

第1章

SOPC及常用软核处理器概述

1.1 从SoC到SOPC

SoC(System on Chip)称为片上系统，它是指将一个完整的产品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。集成电路和系统达到什么程度才算是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、Fresscale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。

1.2 常用软核处理器概述

IP 核重用同样适用 SOPC，而目前开放性源码也已经从软件（Linux、GCC 等）扩展到硬件。对于嵌入式软核处理器来说，出现了像 OpenCores 这样专门发布免费的 IP 核源代码的组织。目前，免费的 32 位嵌入式软核处理器有：GaislResearch 公司的 LEON2/LEON3、OpenCores 组织公布的 OR1200 和 Altera 公司的 NiosII。这 3 种开放性处理器凭借其高性能、低成本，良好的可配置性和完善的开发环境，受到了学术界和工业界的普遍重视，下面对这 3 种软核进行比较。

1.2.1 LEON 系列

LEON 系列处理器主要包括 3 款处理器：LEON1、LEON2 和 LEON3，它们都是由 Jiri Gaisler 设计，并且指令集与 SPARC（Scalable Processor ARChitecture）兼容，其中 LEON1 与 SPARC V7 兼容，LEON2 和 LEON3 与 SPARC V8 兼容。说起 LEON 系列处理器的发展历史，不得不提到 ERC32 和欧洲航天局。

1992 年，Cypress 公司推出了与 SPARC V7 体系兼容的整数处理器 CY7C601 和浮点处理器 CY7C602，运行速度 25 MHz。1996 年（或更早），欧洲航天局（European Space Agency，ESA）以 CY7C601 和 CY7C602 为基础，设计了具备应对 SEU（Single Event Upsets，单粒子翻转）能力的 ERC32（Embedded Real time 32 - bit Computer，嵌入式实时 32 位计算机）模型，作为未来航天器的核心处理器，该项目的技术核心人物正是 Jiri Gaisler。

SEU 是航天器上的电子器件面对的一个特殊问题。太空中的各种高能粒子（包括高能质子、中子、重离子等）具有很高的动能，当这些粒子穿过航天器的电子器件时可能会影响半导体电路的逻辑状态，甚至对半导体材料造成永久损害。单个高能粒子对电子器件功能产生的影响，称之为单粒子效应。其中，导致存储内容在 0 和 1 之间发生变化的现象，称为 SEU。由于这个特殊而严重的问题，导致所有以地面环境为设计要求的商业、甚至军用电子器件都不能很好地满足太空环境的使用要求。

ERC32 使用 VHDL 语言描述，结构上分为 3 个部分：指令单元（IU）、浮点单元（FPU）和存储器控制器（MEC）。位于法国南特的 TEMIC 公司以 ERC32 模型和 MHS（1992 成为 TEMIC 的一部分）的 SPARC V7 整数处理器 90C601、浮点处理器 90C602 为基础，制造了 ERC32 芯片组 TSC690E，该芯片组包括 TSC691E、TSC692E 和 TSC693E，分别对应 ERC32 模型的 IU、FPU 和 MEC。Atmel 公司 1998 年合并了 TEMIC 公司以后，以 ERC32 为基础，制造了 TSC695。TSC695 采用 SoC 技术，把 ERC32 的 3 个分离的部分集成在一块芯片上，主频 25 MHz，虽然结构组织上与 ERC 模型有些差别，但一般认为还是属于 ERC32 体系。现在

TSC691E、TSC692E 和 TSC693E 已经停产，能够买到的只有 TSC695。总的来说，ERC32 系列是一款很成功的产品，欧洲航天局、SAA ERICSSON SPACE 公司、Astrium Space 公司以及著名的伽利略系统都采用了 ERC32 系列的芯片。

1999 年，针对 ERC32 存在的一些问题，由欧洲航天局出资，Jiri Gaisler 对 ERC32 进行了重新设计，使用 SPARC V8 指令集，总体功能与 TSC695 类似，把 ERC32 的 IU、FPU、MEC 集成在一起，同时增加了对处理器 Cache、Meiko 浮点处理器、PCI、AMBA(AMBA 2.0)总线的支持，以及对更多实现工艺库的支持，命名为 LEON。在 LEON 的发展过程中出现过几次大的结构变化，从 2.1 版开始，增加对 PCI 的支持；从 2.2 版开始，内部总线采用了 AMBA 总线；从 2.3 版开始，分为了 2 个版本：标准版和容错版；2001 年底的 2.4 版标志着开发完毕。标准版遵从 GNU 的 LGPL 规定，开放所有源代码，主要用在开发和商用领域，不具备类似 ERC32 的容错功能。容错版则与 ERC32 类似，具有容错功能，但是与 ERC32 不同是，这个版本不是依靠采用特殊的抗辐射工艺而是依靠逻辑结构来实现高可靠性。容错版针对的是空间应用，但是采用的是商业授权模式，不但需要授权费用，而且无法得到源代码。

2001 年，Jiri Gaisler 建立了 Gaisler Resear 公司，专门从事 SPARC 体系处理器的设计和开发。2002 年开始以 LEON 为基础设计 LEON2 处理器，为了区别欧洲航天局的 LEON，LEON 也被称为 LEON1。2005 年底，1.0.32 版完成，标志着 LEON2 开发过程的完毕，此时，LEON2 除了集成了处理器核心及 Cache、Memory 控制器、PCI 控制器、中断控制、DMA、实时时钟、UART 等以外，还集成了处理器 MMU 和以太网控制器。LEON2 也有 2 个版本：标准版和容错版。由于在商业环境下，LEON2 标准版提供的性能不亚于 MIPS、ARM 等嵌入式处理器，而且采用了嵌入式处理器实际上的工业标准——AMBA 总线，使得 LEON2 得到了比 ERC32 更广泛的应用，目前已经有多家公司成功进行了流片，而且其 FPGA 原型也在多个研究项目中得到应用。

在老本行——航天应用上，2005 年底，Atmel 公司基于 LEON2 的容错版制造了 AT697 处理器，主频达到 100 MHz，预计可以得到广泛的使用。

2005 年，Gaisler 开始开发 LEON3。LEON3 也是基于 SPAERC V8 体系和 AMBA 总线的，但是增加了流水线级数，由 LEON2 的 5 级变为 7 级，同时增加了对多处理器的支持，最多同时可支持 4 个处理器。LEON3 的授权模式与 LEON2 基本相同，同样也有容错版。目前，LEON3 还在开发中，并且有多家公司正在进行流片尝试。

1.2.2 Altera 公司的 NiosII

Nios 系列处理器是 Altera 公司推出的基于 Risc 体系结构的通用嵌入式处理器软核，它是 Altera 的可编程逻辑和可编程片上系统(SOPC)设计综合解决方案的核心部分。Altera 前后推出了两代 Nios 系列处理器：Nios 和 NiosII。Nios 是第一代产品，具有 16 位指令集和

16 位×32 位数据通路。NiosII 是第二代完全 32 位 Risc 处理器，具有 32 位指令集、数据通路和地址空间。NiosII 处理器是 5 级流水线设计，采用数据和指令分离的 Harvard 结构。NiosII 拥有自己专用的体系结构与指令集，支持 32 位的硬件乘除法指令，有 32 个通用寄存器。用户还可以根据自己的需要自定义最多 256 条指令。NiosII 采用了 Altera 公司自己的 Avalon 片内总线标准，用于连接定时器、UART 接口、LCD 接口、内存控制器和以太网接口等片内模块。NiosII 同时还提供了一个 Debug 模块，支持 JTAG 在线调试。Altera 公司为 NiosII 提供了极为完善的软硬件开发环境。NiosII 处理器方案是基于 HDL 源码构建的，提供了 3 种性能和资源消耗不同的基本软核：NiosII/f（快速型）、NiosII/s（标准型）和 NiosII/e（经济型）。通过 QuartusII 开发软件中的 SOPCBuilder 系统开发工具，使用者可以在任何一种软核的基础上方便地配置符合自己需要的 NiosII 内核。

Altera 公司同时为 NiosII 提供了基于 GNU C/C++ toolchain 和 Eclipse IDE 的软件开发环境。用户可以在这个开发环境下方便地完成编码、仿真和调试等工作。NiosII 的开发套件内免费提供了一个 Microc/OSII 的实时操作系统支持，同时 NiosII 还支持 μClinux、Nucleus Plus、KROS 等第三方操作系统。NiosII 内核属于半开放的内核。用户可以免费获得 NiosII 的开发平台，不过 NiosII 只支持 Altera 的 Stratix 和 Cyclone 器件。用户只能在 Altera 的 FPGA 芯片上免费使用 NiosII，而且无法获得 NiosII 的 HDL 源代码；另外，设计者若要在 ASIC 设计中使用 NiosII 内核，则需要向 Altera 公司支付一定的授权费。

NiosII 软核是 Altera 公司在其自己生产的 FPGA 上所做的专门的优化，所以在 Altera 的 FPGA 上表现出色，不仅性能强而且占用的逻辑资源少，并且 Altera 提供良好的技术支持；但是 NiosII 只能在 Stratix 和 Cyclone 器件上免费使用，使得 NiosII 的应用范围受到限制。

1.2.3 OpenCores 组织的 OpenRisc 系列

OR1200 是 OpenRisc 系列 Risc 处理器内核的一员。OpenRisc 由 OpenCores 组织负责开发和维护，是免费、开源的 Risc 处理器内核家族。OpenRisc 包括 OpenRisc1000 和 OpenRisc2000，OpenRisc2000 目前还只处于计划阶段。

OpenRisc1000（以下简称 OR1k）的指令系统包括 3 大部分：OpenRisc 基本指令集（ORBIS32/64）、OpenRisc 向量/DSP 扩展指令集（ORVDX64）和 OpenRisc 浮点扩展指令集（ORFPX32/64）。ORBIS32 指令集包括：32 位整数指令、基本 DSP 指令、32 位 Load 和 Store 指令、程序流程控制指令和特殊指令。ORBIS64 指令集包括 64 位整数指令、64 位 Load 和 Store 指令。ORFPX32 指令集包括单精度浮点指令。ORFPX64 指令集包括双精度浮点指令和 64 位 Load 和 Store 指令。ORVDX64 指令集包括向量指令和 DSP 指令。此外，OR1k 还支持自定义指令。

OR1k 的开发工作于 1999 年开始，第一个发行版本是 OpenRisc1001，发行于 2000 年 4

月，由 VHDL 语言描述。支持 ORBIS32 指令集和基本的 DSP 指令的 OR1200 出现于 2001 年 7 月，2002 年 8 月基本成熟，改进和维护一直持续到现在。

OR1200 是一种 32 位、标量、哈佛体系结构、5 级整数流水线 Risc，支持虚拟存储器和基本 DSP 功能。默认的缓存是单通道直接映射的 8 KB 数据缓存和单通道直接映射的 8 KB 指令缓存，每个缓存的分组尺寸为 16 字节。默认的存储器管理单元由基于 64 个散列入口的单通道直接映射的数据后备式转换缓冲区和基于 64 个散列入口的单通道直接映射的指令后备式转换缓冲区组成。辅助功能包括用于实时调试的调试单元、高分辨率滴答计数器、可编程中断控制器和电源管理。其数据和地址总线接口符合 Wishbone 标准。

在使用 0.18μm 和 6 层金属工艺的 ASIC，它的主频达到 300 MHz 时，OR1200 可以提供 300 Dhrystone 2.1 MIPS、300 MHz、32×32 位的 DSP 乘加操作的能力。

OR1200 定位于嵌入式、移动和网络应用。它能够与同类的最新的 32 位标量处理器竞争，并且可以运行任何现代操作系统。竞争对象包括 ARM10、ARC 和 Tensilica 的 Risc 处理器。

2002 年 12 月，*EE Times* 杂志发表了一篇名为 *Building a custom embedded SoC platform for embedded Linux* 的文章，介绍了使用以 OR1200 为基础的 SoC 和嵌入式 Linux。自此，OpenRisc 开始得到广泛关注。OR1200 在 2003 年 3 月被瑞典一家公司采用，用于实现一个声音识别系统；在 2003 年 6 月被美国加州一家公司采用，用于实现一个定位系统；在 2002 年 9 月被 Flextronic 公司选中，用于集成在 Flextronic 的设计中，并提供商业的技术支持服务，如图 1-1 所示。2003 年 8 月，Flextronic 使用 ASIC 成功地实现了一个集成 OR1200、10/100 自适应 Ethernet MAC 控制器、32 位 33/66 MHz PCI 接口、UART16550 和存储器控制器的 SoC 芯片，并且成功地运行了 μClinux 和 Linux 操作系统。迄今为止，最新的一个 OR1200 是由 ViASIC 公司于 2004 年 8 月使用结构化 ASIC 实现的。

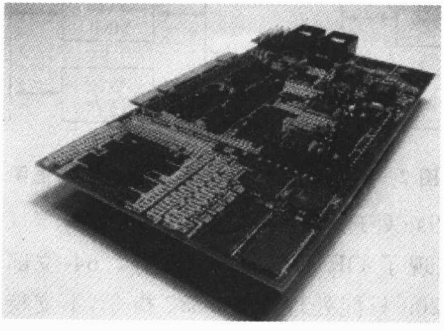


图 1-1　Flextronic 基于 OR1200 的 SoC 及基本系统

第 2 章

OR1200 软核的配置

2.1 OR1200 软核的架构

OR1200 的标准组成的结构框图如图 2－1 所示。

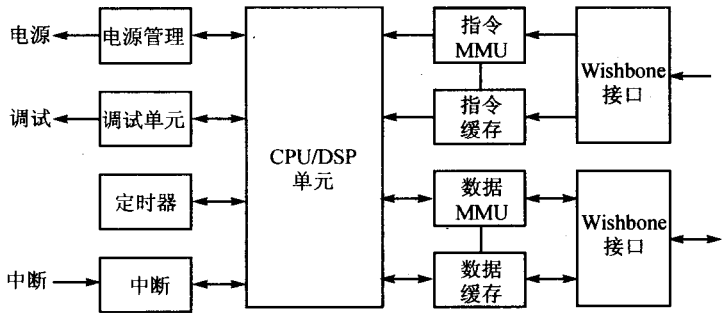


图 2－1　OR1200 的标准组成的结构框图

CPU/DSP 是 OR1200 Risc 处理器的中央部分，图 2－2 是 OR1200 CPU/DSP 的基本框图。OR1200 CPU/DSP 只实现了 OR1k 的 32 位部分。64 位部分、浮点和向量运算没有在 OR1200 中实现。因此，OR1200 只能处理 ORBIS32 指令，不支持 ORFPX32/64 和 ORVDX64 指令。OR1200 实现了 32 个 32 位的通用寄存器。OR1k 体系结构也支持寄存器的影子拷贝，以实现运行上下文的快速切换，但是这个特征还没有在 OR1200 中实现。

可编程中断控制器、滴答定时器、电源管理、调试单元大大增强了 CPU 独立工作的能力，