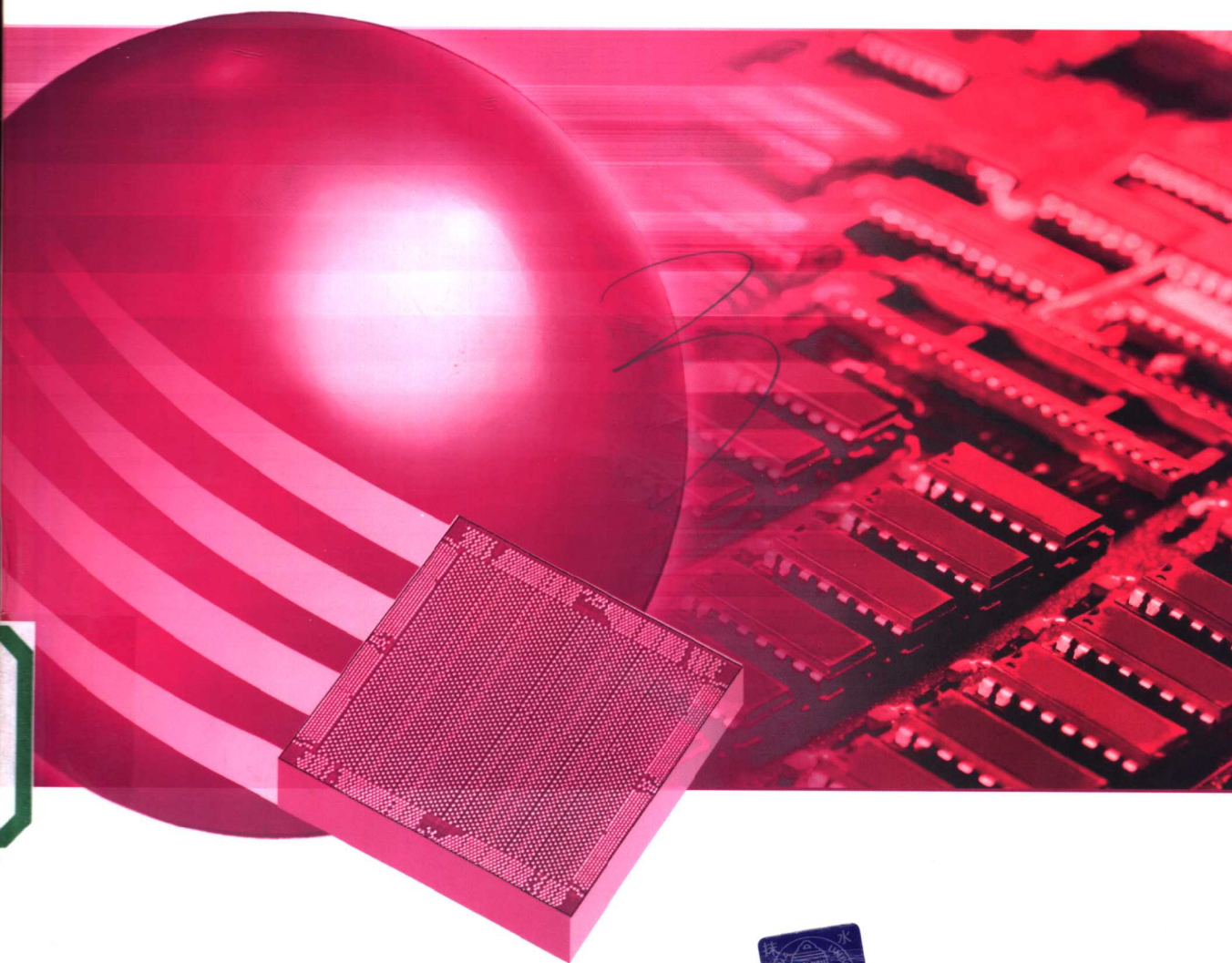


郭书军 王玉花 葛纫秋 编著

# 嵌入式处理器 原理及应用

——Nios 系统设计和 C 语言编程



清华大学出版社

# 嵌入式处理器原理及应用

——Nios 系统设计和 C 语言编程

郭书军 王玉花 葛纫秋 编著

清华大学出版社

北 京

## 内 容 简 介

本书以 Altera 公司开发的 Nios 嵌入式处理器软核为例,介绍了嵌入式处理器的组成原理和开发应用。全书分为 6 章:第 1、2 章介绍嵌入式处理器的组成原理,主要介绍嵌入式处理器的系统组件;第 3、4 章介绍嵌入式处理器的开发环境,包括硬件开发环境和软件开发工具 SOPC Builder、Quartus II 以及嵌入式软件开发工具 GNUPro 的使用;第 5、6 章结合实例介绍嵌入式处理器的应用,主要介绍 Nios 系统设计和 C 程序编程与调试。

本书将 Nios 嵌入式处理器和现场可编程门阵列(FPGA)紧密结合起来,将系统设计和软件编程融为一体,综合了数字逻辑电路、微机原理和接口技术以及 C 语言程序设计,内容新颖,实用性强,特别适合于研究和开发各种嵌入式系统的工程师和研究人员阅读,同时也可以作为计算机应用系统课程的教材以供相关专业的本科生和研究生学习和参考。

版权所有,翻印必究。举报电话:010-62782989 13901104297 13801310933

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

### 图书在版编目(CIP)数据

嵌入式处理器原理及应用——Nios 系统设计和 C 语言编程/郭书军,王玉花,葛勿秋编著. —北京:清华大学出版社,2004.10

ISBN 7-302-09547-7

I. 嵌… II. ①郭… ②王… ③葛… III. ①微处理器-系统设计 ②C 语言-程序设计 IV. TP332

中国版本图书馆 CIP 数据核字(2004)第 095671 号

出 版 者:清华大学出版社 地 址:北京清华大学学研大厦

<http://www.tup.com.cn> 邮 编:100084

社 总 机:010-62770175 客户服务:010-62776969

组稿编辑:曾 刚

文稿编辑:钟志芳

封面设计:秦 铭

版式设计:张红英

印 刷 者:北京国马印刷厂

装 订 者:三河市李旗庄少明装订厂

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:18.5 字数:268 千字

版 次:2004 年 10 月第 1 版 2004 年 10 月第 1 次印刷

书 号:ISBN 7-302-09547-7/TP·6637

印 数:1~5000

定 价:26.00 元

---

本书如存在文字不清、漏印以及缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770175-3103 或(010)62795704

# 前 言

嵌入式系统是当今非常热门的研究领域，早期多以单片机为核心，应用领域非常广泛。但单片机系统功能比较简单，速度较慢，难以适应现代技术的快速发展。

随着现场可编程逻辑阵列（FPGA）技术的日益成熟，基于片上可编程系统（SOPC）的嵌入式处理器越来越多地受到人们的关注。特别是 Altera 公司推出的 Nios 嵌入式处理器软核，通过软件编程的方法可灵活地实现嵌入式处理器的功能，并且针对 FPGA 进行性能优化，可大大提高系统性能。它还具有片上调试功能，便于系统的设计和调试。

Nios 嵌入处理器软核在国外已广泛应用于通信、家电和控制等众多领域，国内正在推广应用，但相关中文资料的严重缺乏给 Nios 嵌入处理器的推广和用户的开发应用带来困难。在 Altera 公司中国代表处和清华大学出版社的大力支持下，作者历经一年多的时间，经过认真钻研和精心编写，终于完成了这部著作，希望能给读者带来帮助。

全书分 4 个层次从硬件设计和软件设计两个方面全面系统地介绍嵌入式处理器的原理和应用，其中前两个层次是入门，后两个层次是提高。

第 1、2 章是第 1 层次。第 1 章简单介绍 Nios 嵌入处理器的概况，包括系统组件和开发工具，使初学者对 Nios 嵌入式处理器有一个初步的认识。

第 2 章是 Nios 嵌入式处理器基础，介绍 Nios CPU、Avalon 总线和外围设备等，重点介绍外围设备的原理和应用基础，初学者可以暂时跳过本章。

第 3 章是第 2 层次，以 Altera 公司的 Nios 开发套件为例介绍典型 Nios 系统的硬件设计过程，其中绝大部分工作是在个人计算机上完成，并不需要 Nios 开发套件。通过本章的介绍，使初学者对 Nios 系统的硬件开发有一定的感性认识，有经验者可以跳过本章。

第 4、5 章是第 3 层次。第 4 章介绍 Nios 系统的软件开发环境，包括 SOPC Builder、Quartus II 和嵌入式软件开发工具 GNUPro，其中 SOPC Builder 和 Quartus II 主要用于硬件开发，GNUPro 用于嵌入式软件的开发和调试。

第 5 章是 C 语言编程基础，介绍 C 语言编程的基本知识和与 Nios 系统软件开发有关的编程基础，供读者在用 C 语言进行软件开发时参考。

第 6 章是第 4 层次。本章以简易 Nios 开发板作为开发平台，介绍 Nios 系统的基本设计、模块设计和综合设计。它是本书的核心和精华，结合实例详细介绍 Nios 系统的硬件设计和软件设计与调试。通过本章的介绍，使读者能够熟练地进行 Nios 系统开发。

在本书的编写过程中，得到了 Altera 公司中国代表处徐平波先生以及清华大学出版社曾刚先生和吕春龙先生的大力支持，作者在此表示衷心的感谢！

本书内容新颖，可参考的相关资料少，难免出现不妥之处，敬请广大读者批评指正。  
E-mail: nios2@126.com。

作 者  
2004 年 6 月

# 目 录

|                               |    |
|-------------------------------|----|
| 第 1 章 概述 .....                | 1  |
| 1.1 嵌入式系统简介 .....             | 1  |
| 1.2 Nios 嵌入式处理器 .....         | 3  |
| 1.3 Nios 系统组件 .....           | 5  |
| 1.3.1 Nios CPU 结构 .....       | 5  |
| 1.3.2 Avalon 交换结构总线 .....     | 7  |
| 1.3.3 外围设备 .....              | 9  |
| 1.3.4 片内调试模块 .....            | 10 |
| 1.4 Nios 开发环境 .....           | 11 |
| 1.4.1 SOPC Builder 开发工具 ..... | 12 |
| 1.4.2 Quartus II 开发工具 .....   | 13 |
| 1.4.3 GNUPro 嵌入式软件开发工具 .....  | 14 |
| 1.4.4 Nios 开发套件 .....         | 14 |
| 第 2 章 系统组件 .....              | 16 |
| 2.1 Nios CPU .....            | 16 |
| 2.1.1 内部寄存器 .....             | 16 |
| 2.1.2 中断处理 .....              | 20 |
| 2.2 Avalon 总线 .....           | 24 |
| 2.3 外围设备 .....                | 25 |
| 2.3.1 定时器 .....               | 25 |
| 2.3.2 并行输入输出接口 PIO .....      | 31 |
| 2.3.3 通用异步串行接口 UART .....     | 34 |
| 2.3.4 3 线主/从串行接口 SPI .....    | 44 |
| 2.3.5 DMA 控制器 .....           | 51 |
| 2.3.6 存储器接口 .....             | 58 |
| 第 3 章 硬件开发环境 .....            | 64 |
| 3.1 开发套件 .....                | 64 |
| 3.1.1 开发板简介 .....             | 65 |
| 3.1.2 入门指南 .....              | 73 |
| 3.2 硬件开发指南 .....              | 79 |
| 3.2.1 设计输入 .....              | 80 |
| 3.2.2 编译 .....                | 96 |

|                            |            |
|----------------------------|------------|
| 3.2.3 编程.....              | 101        |
| 3.3 软件开发指南.....            | 106        |
| <b>第 4 章 软件开发环境.....</b>   | <b>119</b> |
| 4.1 SOPC Builder 开发工具..... | 119        |
| 4.1.1 SOPC Builder 功能..... | 119        |
| 4.1.2 SOPC Builder 组成..... | 120        |
| 4.1.3 SOPC Builder 使用..... | 126        |
| 4.2 Quartus II 开发工具.....   | 137        |
| 4.2.1 设计流程.....            | 137        |
| 4.2.2 设计输入.....            | 140        |
| 4.2.3 综合.....              | 145        |
| 4.2.4 布局布线.....            | 147        |
| 4.2.5 定时分析.....            | 151        |
| 4.2.6 仿真.....              | 154        |
| 4.2.7 编程配置.....            | 155        |
| 4.2.8 调试.....              | 158        |
| 4.2.9 系统级设计.....           | 162        |
| 4.2.10 软件开发.....           | 164        |
| 4.3 GNUPro 嵌入式软件开发工具.....  | 165        |
| 4.3.1 软件开发工具包 SDK.....     | 165        |
| 4.3.2 软件开发实用程序.....        | 174        |
| <b>第 5 章 C 语言编程基础.....</b> | <b>184</b> |
| 5.1 数据及其运算.....            | 184        |
| 5.1.1 数据和数据类型.....         | 184        |
| 5.1.2 运算符和表达式.....         | 187        |
| 5.2 程序设计基础.....            | 189        |
| 5.2.1 函数及其使用.....          | 189        |
| 5.2.2 输入输出函数.....          | 191        |
| 5.2.3 流程控制语句.....          | 192        |
| 5.2.4 编译预处理.....           | 195        |
| 5.3 构造和派生.....             | 197        |
| 5.3.1 构造数据类型.....          | 197        |
| 5.3.2 指针及其使用.....          | 201        |
| <b>第 6 章 系统设计与调试.....</b>  | <b>208</b> |
| 6.1 简易 Nios 开发板简介.....     | 208        |
| 6.2 基本设计.....              | 212        |

---

|                           |     |
|---------------------------|-----|
| 6.2.1 I 型开发板基本设计 .....    | 212 |
| 6.2.2 II 型开发板基本设计 .....   | 222 |
| 6.3 模块设计 .....            | 228 |
| 6.3.1 矩阵键盘接口设计 .....      | 228 |
| 6.3.2 液晶显示屏接口设计 .....     | 234 |
| 6.3.3 GPS 模块接口设计 .....    | 238 |
| 6.3.4 GSM 模块接口设计 .....    | 243 |
| 6.4 综合设计 .....            | 252 |
| 6.4.1 硬件设计 .....          | 252 |
| 6.4.2 软件设计 .....          | 254 |
| 附录 A ASCII 码表 .....       | 261 |
| 附录 B 32 位 Nios 指令系统 ..... | 262 |
| 附录 C Cyclone 器件简介 .....   | 266 |
| 参考文献 .....                | 286 |

# 第 1 章 概 述

嵌入式系统是嵌入到对象体系中的专用计算机系统，包括硬件和软件两大部分。硬件包括处理器、存储器、输入输出接口和外部设备等，软件包括系统软件和应用软件，嵌入式系统的系统软件和应用软件紧密结合。

嵌入式处理器是嵌入式系统的核心，有硬核和软核之分。常用的嵌入式处理器硬核有 ARM、MIPS、PowerPC、Intel x86 和 Motorola 68000 等；Altera 公司开发的 Nios 是 16/32 位嵌入式处理器软核。和硬核相比，软核的使用灵活方便。

Nios 嵌入式处理器是可配置的通用 RISC 处理器，可以很容易地与用户逻辑相结合，集成到 Altera FPGA 器件中。Nios 具有 16 位指令系统，用户可选择 16 位或 32 位数据宽度和可灵活配置的标准外围设备及软件库。Nios 系统将处理器、存储器、输入输出接口和其他 IP 核灵活地集成到 SOPC 设计中。

## 1.1 嵌入式系统简介

嵌入式系统 (Embedded System) 是当今最热门的概念之一，但这个概念并非新近才出现。从 20 世纪 70 年代单片机的出现到今天各式各样嵌入式处理器的大规模应用，嵌入式系统已经有了近 30 年的发展历史。

嵌入式系统最初的应用是单片机系统。20 世纪 70 年代出现的单片机使汽车、家电、工业机器、通信装置及其他成千上万种产品可以通过内嵌电子装置来获得更佳的使用性能、更容易使用，更快和更便宜，这些装置已经初步具备了嵌入式的应用特点。

嵌入式系统应用广泛，嵌入式处理器的使用数量已经远远超过了通用微处理器。个人计算机的外部设备中就有多个嵌入式处理器。制造、控制、通信、测量、汽车、船舶、航空、航天、军事设备和消费类产品等都是嵌入式系统发挥重要作用的领域。

嵌入式系统是以应用为中心，以计算机技术为基础，硬件和软件可裁剪，适应应用系统对功能、可靠性、成本、体积和功耗等严格要求的专用计算机系统。

可以从以下几个方面来理解嵌入式系统的含义：

- ❖ 嵌入式系统面向用户、面向产品和面向应用，必须与具体应用相结合才会具有生命力，才更具有优势。嵌入式系统与应用紧密结合，具有很强的专用性。
- ❖ 嵌入式系统将先进的半导体技术、计算机技术和电子技术以及各个行业的具体应用相结合，是一个技术密集、资金密集、高度分散和不断创新的知识集成系统。
- ❖ 嵌入式系统必须根据应用需求对硬件和软件进行裁剪，以满足应用系统的功能、可靠性、成本、体积和功耗等要求。比较好的发展模式是：首先建立相对通用的硬件和软件基础，然后开发出适应各种需要的嵌入式系统。



同时还应该看到, 嵌入式系统是一个外延极广的概念。凡是与产品结合在一起的具有嵌入式特点的系统都可以称为嵌入式系统。

一般而言, 嵌入式系统和通用计算机系统类似, 由处理器、存储器、输入输出接口和设备以及软件等部分组成。但作为专用计算机系统的嵌入式系统与通用计算机系统相比, 具有以下几个重要特征。

- ❖ 系统简练。由于嵌入式系统一般应用于小型电子装置, 系统资源相对有限, 所以系统内核与传统的系统相比要小得多。嵌入式系统一般没有系统软件和应用软件的明显区分, 不要求功能设计和实现的过于复杂, 这既有利于控制系统成本, 同时也有利于实现系统安全。
- ❖ 专用性强。嵌入式系统的个性化很强, 软件和硬件的结合非常紧密, 一般要针对硬件进行软件的设计, 即使在同一品牌、同一系列的产品中也需要根据硬件的变化和增减对软件不断进行修改。同时针对不同的任务, 往往需要对系统进行较大更改, 程序的编译下载也要同系统相结合。
- ❖ 实时操作系统支持。嵌入式系统的应用程序可以不需要操作系统的支持直接运行, 但为了合理地调度多任务, 充分利用系统资源, 用户必须自行选配实时操作系统 (Real-Time Operating System, RTOS) 开发平台, 这样才能保证程序执行的实时性和可靠性, 减少开发时间, 保障软件质量。
- ❖ 专门开发工具支持。嵌入式系统本身不具备自主开发能力, 即使在设计完成以后用户通常也不能对程序功能进行修改, 必须有一套开发工具和环境才能进行开发。开发工具和环境一般基于通用计算机的软硬件设备、逻辑分析仪和信号示波器等。

在嵌入式系统的软件开发过程中, 采用 C 语言将是最佳的选择。由于汇编语言是非结构化的语言, 不能胜任大型的结构化程序设计, 必须采用更高级的 C 语言进行设计。

随着半导体技术的不断发展, 片上系统 (System on a Chip, SOC) 成为嵌入式应用领域的热门方向之一。SOC 最大的特点是成功实现了软硬件的无缝结合, 直接在处理器芯片内嵌入操作系统的代码模块。此外, SOC 有极高的综合性, 在一个芯片内部运用 VHDL 等硬件描述语言可以实现复杂的系统。

用户使用 SOC, 不需要再像传统的系统设计一样绘制庞大复杂的电路板, 而只需要使用相应的开发工具, 将处理器、存储器和接口逻辑集成在一起, 并开发相应的软件, 编译仿真之后就可以直接交付芯片厂商进行生产。

SOC 通常是专用集成电路 (ASIC), 所以不为用户所熟知, 而且其开发周期长, 生产成本低, 产品不能进行修改。随着可编程逻辑器件 (Programmable Logic Device, PLD) 的广泛应用, 可编程片上系统 (System on a Programmable Chip, SOPC) 越来越多地受到人们的关注。

SOPC 是在 PLD 上实现 SOC, PLD 的可编程性使 SOPC 的设计和实现非常方便。用户可以灵活地进行系统硬件和软件设计, 还可以在现场进行系统修改。PLD 性能的不断提, 也使 SOPC 的性能越来越高。

Altera 是 PLD 的大生产商, 生产的 PLD 有 CPLD (Complex Programmable Logic Device, 复杂可编程逻辑器件) 和 FPGA (Field Programmable Gate Array, 现场可编程门阵列) 两大系列。CPLD 和 FPGA 的结构有所不同, 但功能差别不大, 作为新产品的 FPGA 要比早

期的 CPLD 性能强大。

Altera 的 SOPC 嵌入式处理器 (Embedded Processor) 解决方案有两种: 嵌入 ARM922T 硬核的 Excalibur 器件和用于 FPGA 的可配置 Nios 嵌入式处理器软核。

## 1.2 Nios 嵌入式处理器

Nios 嵌入式处理器是用户可配置的通用 RISC 嵌入式处理器, 它是一个非常灵活和强大的处理器。Nios 处理器的易用和灵活已经使它成为世界上最流行的嵌入式处理器。

嵌入式设计者利用 SOPC Builder 系统开发工具能够很容易地创建自己的处理器系统。SOPC Builder 可用于集成一个或多个可配置的带有许多标准外围设备的 Nios CPU, 并利用自动形成的 Avalon 交换结构总线将这些系统连接在一起。

可配置 Nios CPU 是 Nios 处理器系统的核心, 它能够被灵活配置而适用于各种各样的应用。例如一个 16 位 Nios CPU, 在片内 ROM 中运行一个小程序, 可以制作成一个实际的序列发生器或控制器, 并且能够代替固定编码的状态机。又如一个 32 位 Nios CPU, 与外围设备、硬件加速单位和自定义指令一起, 构成一个功能强大的 32 位嵌入式处理器系统。

Nios 嵌入式处理器的独特性 (例如自定义指令和并行的多控制器 Avalon 交换结构总线) 使它不同于市场上其他的处理器软核。这些特性允许 Nios 用户通过用简单的而非传统的方法加速和优化自己的设计。

32 位和 16 位 Nios 嵌入式处理器典型配置的比较如表 1-1 所示。

表 1-1 Nios 嵌入式处理器典型配置比较

| 特 性                            | 32 位 Nios CPU | 16 位 Nios CPU |
|--------------------------------|---------------|---------------|
| 数据总线宽度/bit                     | 32            | 16            |
| 算术逻辑单元 (ALU) 宽度/bit            | 32            | 16            |
| 内部寄存器宽度/bit                    | 32            | 16            |
| 地址总线宽度/bit                     | 32            | 16            |
| 指令长度/bit                       | 16            | 16            |
| 逻辑单元数 (LEs) (典型值) <sup>①</sup> | <1500         | <1000         |
| $f_{\text{MAX}}$ <sup>①</sup>  | >125MHz       | >125MHz       |

注: ① 其具体数值与器件结构有关。

Nios 嵌入式处理器指令系统结构的设计具有以下特性:

- ❖ 在 Altera FPGA 中有效实现。
  - ✧ 使用最少的逻辑单元。
  - ✧ 使用最少的存储单元。
  - ✧ 最大的时钟速度。
- ❖ 用 SOPC Builder 容易进行系统集成。
  - ✧ 简单的存储器接口。

- ✧ 标准的可配置外围设备库。
- ✧ 在 CPU、外围设备和存储器之间自动形成 Avalon 交换结构接口逻辑电路。
- ❖ 为编译嵌入式软件优化指令系统结构。
  - ✧ 灵活的寻址方式。
  - ✧ 大容量内部寄存器组的有效利用。
  - ✧ 快速的中断处理。
- ❖ 硬件加速模块。
  - ✧ 有效算法实现。
  - ✧ MSTEP 指令：单步乘法单元。
  - ✧ MUL 指令：快速整数乘法单元。
- ❖ 自定义指令

Nios 嵌入式处理器支持 Altera 主流 FPGA 的全部系列，器件支持如表 1-2 所示。

表 1-2 Nios 嵌入式处理器器件支持

| 器 件            | 说 明                         |
|----------------|-----------------------------|
| Stratix II     | 最高的性能，最高的密度，大量的存储资源，特性丰富的平台 |
| Stratix GX     | 最高的性能结构，高速串行收发器             |
| Cyclone        | 低成本，替代 ASIC，适用于价格敏感的应用      |
| APEX II        | 高密度，高性能，支持高速差分 I/O 标准       |
| Mercury        | 高性能，高带宽，中密度，包括时钟数据恢复（CDR）支持 |
| Excalibur      | 高性能，处理器硬核解决方案               |
| APEX 20K/KE/KC | 高性能，中到高密度                   |
| FLEX 10K/KE    | 低成本，低到中密度                   |
| ACEX 1K        | 低成本，低到中密度                   |
| HardCopy       | 高密度，大批量 ASIC 替换器件           |

随着超过 1 万个 Nios 开发套件的交付使用，Nios 嵌入式处理器已经成为嵌入式处理器软核的标准。3.0 版本的 Nios 嵌入式处理器具有更高的性能，包括：

- ❖ 更快地存取低成本的 SDRAM 器件。
- ❖ 片内指令和数据高速缓存。
- ❖ 支持实时调试的 JTAG 调试器。
- ❖ 增强的 Avalon 交换结构总线。

Nios 嵌入式处理器的系统组件、开发工具和开发平台如表 1-3 所示。

表 1-3 Nios 嵌入式处理器系统组件、开发工具和开发平台

| 系 统 组 件       | 开 发 工 具          | 开 发 平 台 |
|---------------|------------------|---------|
| CPU           | SOPC Builder     | 开发套件    |
| Avalon 交换结构总线 | Quartus II 设计软件  | 软件授权    |
| 外围设备          | GNUPro 嵌入式软件开发工具 |         |
| 片内调试模块        | 第三方工具            |         |

## 1.3 Nios 系统组件

Nios 嵌入式处理器系统包括一个或多个 Nios CPU、Avalon 交换结构总线和其他组件。Altera 的 SOPC Builder 系统开发工具可以自动生成这些组件以及连接它们的总线。

下列组件可用于生成基于 Nios 处理器的嵌入式系统：

- ❖ Nios CPU。
- ❖ Avalon 总线。
- ❖ 外围设备和存储器接口。
- ❖ 片内调试模块。

设计者能够使用 SOPC Builder 设计 Nios 处理器系统，如图 1-1 所示。

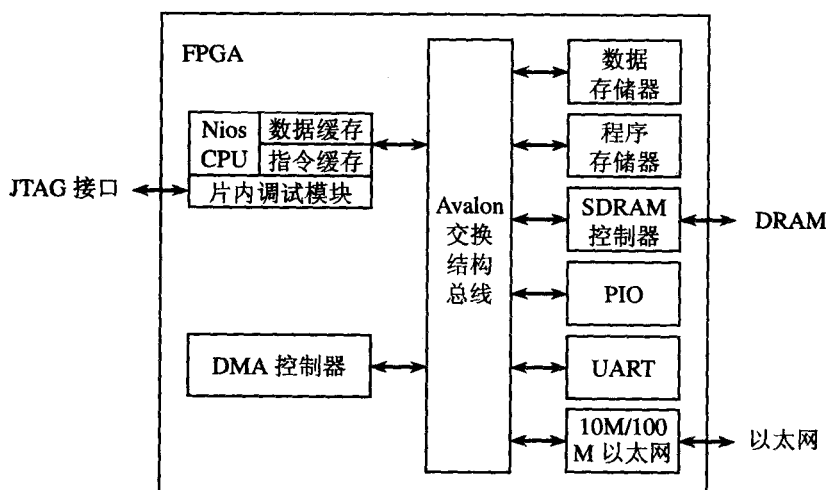


图 1-1 Nios 处理器系统

Nios 处理器系统包含带指令和数据高速缓存的 Nios CPU、片内调试模块、直接存储器存取（DMA）控制器、常用外围设备（PIO、UART、以太网端口和存储器接口等）和并行多控制器 Avalon 交换结构总线。

### 1.3.1 Nios CPU 结构

Nios CPU 是 16 位和 32 位结构可配置并包含五级流水线的通用 RISC 微处理器。16 位和 32 位 Nios CPU 都使用 16 位指令格式以减少程序代码长度和指令存储宽度。Nios CPU 指令系统针对 SOPC 和编译嵌入式应用进行优化。

Nios CPU 结构包括：

- ❖ 指令系统。
- ❖ 寄存器组。

- ❖ 高速缓存。
- ❖ 中断处理。
- ❖ 硬件加速。

Nios 嵌入式处理器采用改进的哈佛存储器结构, CPU 带有分离的数据和程序存储器总线控制。SOPC Builder 系统开发工具允许用户容易地指定系统中 Avalon 控制器和从属设备之间的连接, 这些从属设备可以是存储器或外围设备。

Nios 指令总线是 16 位, 用于从存储器中读取指令。Nios 数据总线宽度是 16 位或 32 位, 分别用于 Nios CPU 的 16 位或 32 位配置。

#### (1) 指令系统

Nios 指令系统支持 C 和 C++ 程序编译, 包括算术和逻辑运算、位操作、字节读、数据传送、流程控制和条件转移等指令。指令系统包含丰富的寻址方式以减少代码长度和提高处理器性能。

#### (2) 寄存器组

Nios CPU 有一个大容量的窗口化的通用寄存器组、8 个控制寄存器、一个程序计数器和一个用于指令前缀的 K 寄存器。

通用寄存器在 16 位 Nios CPU 中是 16 位, 在 32 位 Nios CPU 中是 32 位。寄存器组可配置为包含 128、256 或 512 个寄存器。软件可以通过包含 32 个寄存器的滑动窗口存取这些寄存器, 滑动窗口的移动间隔是 16 个寄存器。滑动窗口允许快速地进行寄存器切换, 加速子程序的调用和返回。

#### (3) 高速缓存

可配置的 Nios CPU 可以有选择地包含指令和数据高速缓存。高速缓存通常通过提供局部存储系统提高 CPU 的性能, 这个局部存储系统可以快速响应 CPU 产生的总线事件。Nios 高速缓存的实现是采用简单的直接映射的连续写入结构, 这种结构设计能够用最少的器件资源消耗获得最大的性能。

#### (4) 中断处理

Nios 处理器允许多达 64 个矢量中断。中断源有三类: 外部硬件中断、内部中断和软件中断。Nios 中断处理模式能够准确地处理所有内部中断。

用户可以有选择地禁止 TRAP 指令软件中断、硬件中断和内部中断。这项选择能够减少 Nios 系统的大小, 但只用于处理器不运行复杂软件的系统。

#### (5) 硬件加速

Nios 指令系统可以利用硬件提高系统性能。特殊的周期密集型软件操作可以用硬件显著地提高系统性能。这项特性通过修改指令系统提供。

Nios 处理器有两种指令系统修改方法: 自定义指令和标准 CPU 选项。

##### ① 自定义指令

开发者可以通过向 Nios 处理器指令系统中添加自定义指令加快时间要求严格的软件算法。开发者也可以用自定义指令在单周期和多周期操作中执行复杂的处理任务。另外, 用户添加的自定义指令逻辑电路可以访问 Nios 系统外的存储器和逻辑电路。

复杂的操作序列可以在硬件中简化为单指令的执行。这项特性允许开发者为数字信号

处理 (DSP)、分组标题处理和计算密集操作优化自己的软件。

Altera 的 SOPC Builder 软件提供一个图形用户界面 (GUI)，开发者利用这个图形用户界面可以向 Nios 嵌入式处理器中添加多达 5 个自定义指令。

## ② 标准 CPU 选项

Altera 提供单独的预定义指令来提高软件性能。MUL 和 MSTEP 指令就是两个与其他的硬件一起实现的预定义指令。当用户在 SOPC Builder 中选择这些 CPU 选项时，相关逻辑被增加到算术逻辑运算单元 (ALU)。例如，如果用户选择执行 MUL 指令，整数乘法器被自动地添加到 CPU 的 ALU 中，并在两个时钟周期内完成 16 位与 16 位的乘法操作。相同的操作循环的软件程序实现需要 80 个时钟周期。

硬件加速乘法器的资源利用和时钟周期如表 1-4 所示。

表 1-4 硬件加速乘法器的资源使用和时钟周期

| 乘法器选项  | 逻辑单元             | 时钟周期 $16 \times 16 \Rightarrow 32^{\text{①}}$ | 时钟周期 $32 \times 32 \Rightarrow 32^{\text{②}}$ |
|--------|------------------|-----------------------------------------------|-----------------------------------------------|
| 无 (软件) | 0                | 80                                            | 250                                           |
| MSTEP  | 125              | 18                                            | 80                                            |
| MUL    | 370 <sup>③</sup> | 2                                             | 20                                            |

注：① 两个无符号的 16 位数整乘产生一个无符号的 32 位结果。两个有符号的 16 位数整乘产生一个有符号的 32 位结果。

② 两个无符号的 32 位数整乘产生一个无符号的 32 位结果。两个有符号的 32 位数整乘产生一个有符号的 32 位结果。

③ 当使用 Stratix 器件时，MUL 在 DSP 模块中实现，不使用额外的逻辑单元。

另外，Nios CPU 包括一个内部移位单元用于实现逻辑和算术移位指令。CPU 使用固定的桶状移位逻辑在两个时钟周期内完成全部的移位操作。

硬件加速操作对可编程逻辑器件 (PLD) 中的处理器软核非常有利。Nios 开发者可以在速度和面积间选择，增加了 SOPC 设计的灵活性。

## 1.3.2 Avalon 交换结构总线

Avalon 交换结构总线是 Altera 开发的用于 Nios 嵌入式处理器的参数化接口总线，由一组预定义的信号组成，用户用这些信号可以连接一个或多个 IP 模块。Altera 的 SOPC Builder 系统开发工具自动地产生 Avalon 交换结构总线逻辑。

Avalon 交换结构总线需要极小的 FPGA 资源，提供完全的同步操作，它的重要特性有：

- ❖ 简单的基于向导的配置。
- ❖ 并行的多控制器。
- ❖ 多达 4GB 的地址空间。
- ❖ 同步接口。
- ❖ 嵌入的地址译码。
- ❖ 带延迟的读写操作。
- ❖ 数据流处理。
- ❖ 动态的外围设备接口大小。

### (1) 简单的基于向导的配置

Altera 的 SOPC Builder 系统开发工具中易于使用的图形用户界面引导用户进行增加外围设备、指定控制器和从属设备关系以及定义存储地址等操作。Avalon 交换结构总线按照用户从向导界面的输入自动形成。

### (2) 并行的多控制器

Nios 开发者可以按照自己的特殊操作需要创建自定义的系统总线结构, 优化自己的系统数据流。Avalon 交换结构总线支持所有总线控制器的并行事务处理, 并自动地为共享外围设备和存储器接口进行仲裁。另外, 直接存储器存取 (DMA) 设备能被用于与其从属设备一起提供总线控制能力。

在传统的总线中, 单个仲裁器控制一个或多个总线控制器跟总线从属设备进行通信, 由于每次只能有一个控制器可以存取系统总线而形成带宽瓶颈, 如图 1-2 所示。

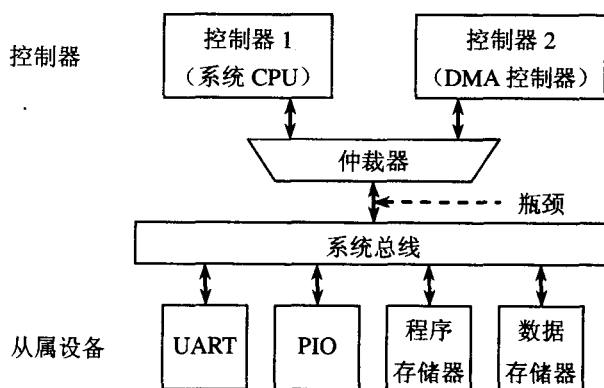


图 1-2 传统总线结构

Avalon 交换结构总线使用从属设备侧仲裁技术, 使并行多控制器操作最大限度地提高系统性能。如果多个控制器同时存取从属设备, 则由从属设备侧仲裁决定哪一个控制器得到从属设备的存取权, 如图 1-3 所示。

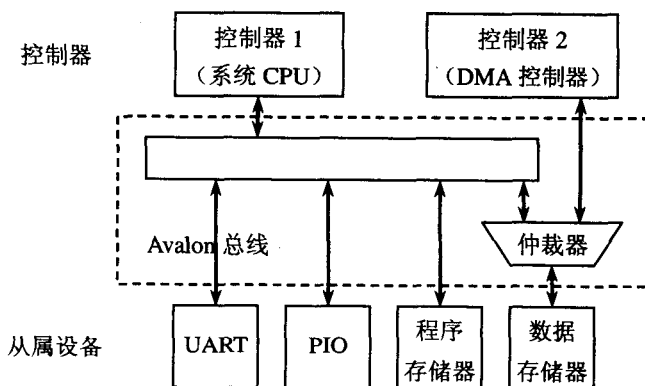


图 1-3 并行多控制器 Avalon 交换总线结构

在这样的系统中,快速以太网等高速外围设备可以在不暂停 CPU 的情况下直接存取数据存储器。

通过允许独立于 CPU 的存储器存取, Avalon 交换结构总线优化了数据处理过程,提高了系统吞吐量。

### (3) 多达 4GB 的地址空间

存储器和外围设备可以映射到 32 位地址空间的任何地方。换句话说, CPU (或其他的总线控制器) 有多达 4GB 的可寻址存储器范围。

### (4) 同步接口

所有的 Avalon 信号和 Avalon 总线时钟同步, 这使相应的 Avalon 交换结构总线时间性能简单化, 便于高速外围设备的集成。

### (5) 嵌入的地址译码

SOPC Builder 创建的 Avalon 交换结构总线自动地为所有外围设备 (甚至用户自定义外围设备) 形成片选信号, 这可极大地简化基于 Nios 处理器系统的设计。

分离的地址和数据通路为片内用户逻辑提供了一个极其容易的连接, 用户自定义外围设备不需要数据和地址总线周期译码。

### (6) 带延迟的读写传输

Avalon 交换结构总线可以完成带延迟的读写操作, 这种延迟传输是很有用的。这是因为, 控制器可以先发出读写请求, 在执行一个无关的任务后接收数据。这项特性对发送多个读写请求到一个已知延迟的从属设备也非常有用。

例如, 这对在连续的地址内同时进行取指令操作和 DMA 传输非常有好处。在这种情况下, CPU 或 DMA 控制器可以预取预定的数据, 以减少同步存储器的平均存取延迟。

### (7) 数据流处理

带 Avalon 交换矩阵的数据流处理在数据流控制器和数据流从属设备之间建立一个开放的通道, 以完成连续的数据传送。这些通道允许数据在控制器和从属设备对之间流动。控制器不必连续地读取从属设备中的状态寄存器来决定从属设备是否可以发送或接收数据。

数据流处理在控制器和从属设备对之间获得最大的数据吞吐量, 并避免在从属设备上出现数据溢出。这对 DMA 传输尤其有用。

### (8) 动态的外围设备接口大小

动态的总线大小允许 Nios 开发者使用低成本的窄的存储器件, 这些存储器件可以和 Nios CPU 的总线大小不匹配。例如, 32 位数据总线的系统可以容易地集成 8 位闪速存储器件。在这样的系统中, 如果必要的话, 动态的总线大小调整逻辑自动地执行多总线周期, 以便从窄的外围设备中取出宽的数据值。SOPC Builder 自动地添加完成大小调整和定位调整所需要的专用逻辑。

## 1.3.3 外围设备

Nios 嵌入式处理器包括可以在 Altera 可编程逻辑中实现的标准外围设备库, Altera 提



供给用户这些外围设备的 Verilog HDL 或 VHDL 源代码以及所有系统综合需要的软件程序。

设计者能够使用 SOPC Builder 系统开发工具配置自己的 Nios 处理器系统。利用 SOPC Builder 直观的向导用户界面,设计者可以配置系统组件、映像地址、主/从关系和中断优先顺序等。

Nios 外围设备库包括的外围设备如表 1-5 所示。

表 1-5 Nios 外围设备库

| 外 围 设 备 | 说 明                                         |
|---------|---------------------------------------------|
| 定时器     | 32 位定时器, 能被用作周期性脉冲发生器或系统监视定时器               |
| PIO     | 1~32 位并行 I/O 接口 (输入、输出和边沿捕捉)                |
| UART    | 通用串行接口, 波特率、数据位、校验位和停止位可调, 流量控制信号可选         |
| SPI     | 3 线主/从串行外设接口                                |
| DMA 控制器 | 在外围设备和存储器之间有效地进行批量数据传送                      |
| 存储器接口   | 连接片内 ROM 和 RAM, 片外 SRAM、Flash、SDRAM 和串行配置器件 |
| 以太网接口   | 与可选的 Nios 以太网开发套件一起使用                       |
| 用户逻辑接口  | 连接片内用户逻辑或片外器件                               |

另外,设计者也可以使用 SOPC Builder 很容易地像使用标准外围设备一样集成用户的自定义模块。

### 1.3.4 片内调试模块

Altera 的合作伙伴 First Silicon Solutions (FS2) 和 Accelerated Technology (Mentor Graphics 的嵌入式系统分公司) 提供给 Nios 嵌入式处理器系统开发者世界级的调试工具。

Nios 开发套件支持的不同级别的调试工具如图 1-4 所示。

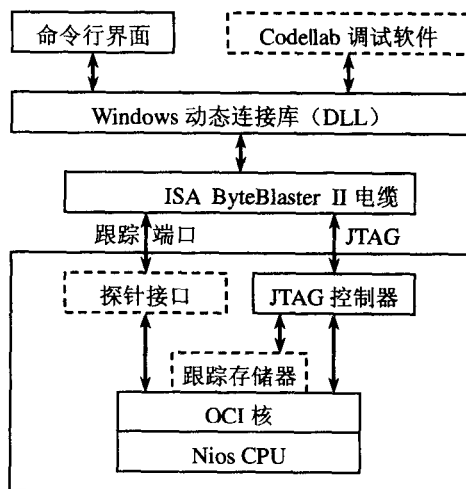


图 1-4 Nios 开发套件支持的调试工具