基于 ARM 和 FPGA 的 NoC 资源网络接口驱动设计与实现

许川佩，孙义军，吴玉龙
（桂林电子科技大学 电子工程与自动化学院，广西 桂林 541004）

摘 要：将 ARM 处理器作为 NoC 系统中的一个资源节点，设计了资源网路接口，基于 Linux 操作系统的基础上，编写了 FPGA 设备的驱动程序。在典型的 3×3 2D Mesh 结构的 NoC 系统中进行了测试，结果表明该设计实现了 ARM 处理器资源节点和 NoC 系统中其他 IP 核数据的高速、可靠传输。

关键词：NoC；ARM；Linux；设备驱动；资源网络接口

中图分类号：TP316 **文献标识码**：A **文章编号**：1674-7720(2013)13-0083-04

Design and implementation of NoC's resource network interface driver based on ARM and FPGA

Xu Chuanpei, Sun Yijun, Wu Yulong
(School of Electronic Engineering and Automatic, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: In the design, ARM processor is set as one of resource nodes of NoC, the resource-network interface was designed, and the driver was programmed for FPGA device based on Linux operating system. This driver has been verified in NoC system which uses 3×3 2D Mesh structure. This design achieves the high speed and reliable data transmission between ARM Processor and other resource nodes of NoC.

Key words: NoC; ARM; Linux; device driver; RNI

在半导体工艺进入深亚微米时代后，由于 SoC（System-on Chip）大多采用类似计算机系统的总线结构，使其存在着通信效率低下、全局同步时钟设计困难等问题，这些问题使得 SoC 体系结构以及其相应的设计方法在多核的复杂系统中遇到了无法逾越的障碍。为了解决 SoC 面临的上述问题，提出了全新的 NoC（Network-on-chip）体系结构[1]。NoC 技术的核心是将计算机网络通信的思想移植到芯片设计中来，它采用路由和分组交换技术替代传统的总线通信方式，从体系结构上彻底解决了片上系统的通信瓶颈和时钟问题。

目前各国的研究人员正积极从事 NoC 设计研究，但缺少成熟技术和产品。本设计在开展 NoC 设计技术研究的基础上，将 ARM 处理器作为 NoC 的其中一个资源节点，利用 ARM 处理器功能强大等特点，拓展 NoC 的应用。

本文通过向 ARM 处理器中移植 Linux 操作系统，在此基础上进行了资源网络接口和 FPGA 的设备驱动设计，并对多核系统之间的大量数据高速传输通信进行了探索和验证。

1 NoC 系统模型及硬件平台

NoC 是由通信节点网络和资源节点组成，通信资源网络包括路由节点和资源网络接口 RNI（Resource-Network-Interface），通信节点负责资源节点之间的数据通信，资源节点完成广义上的计算任务，资源节点可以是嵌入式微处理器和 DSP 核、可重构器件、输入输出设备等，它通过资源网络接口连接到片上网络中。资源网络接口是资源节点与路由节点之间进行通信的桥梁，主要由发送模块和接收模块组成。其功能是将资源节点的数据按照传输协议进行打包处理后发送到片上网络中，并从将网络中接收提取有用数据传递给资源节点。NoC 系统模型如图 1 所示。

本硬件平台选用 Altera Cyclone IV 系列的 EP4CE115F29 FPGA 芯片作为 NoC 系统的核心部件[2]，在此 FPGA 中以

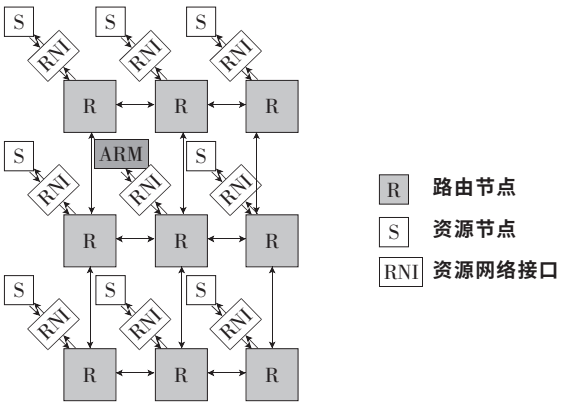


图 1 基于 2D-Mesh 的 3×3 NoC 系统结构模型

规则的 3×3 2D-Mesh 拓扑结构[3]，虚通道技术的虫洞数据交换方式以及无锁死的确定性 XY 维序路由算法作为理论模型，完成 NoC 通信框架的构建。本设计将 ARM 处理器作为 NoC 系统其中一个资源节点，通过资源网络接口和 FPGA 设备驱动实现 ARM 与 NoC 系统其他资源节点之间数据的交互。

2 ARM 处理器资源网络接口设计

资源网络接口负责将 ARM 资源节点的输出数据进行组包并发送至路由节点，完成接收处理片上网络传递的数据包，并通过中断方式通知 ARM 资源节点接收数据。因此设计分为资源网络接口发送和接收两个部分。

2.1 ARM 资源节点和路由节点接口结构

ARM 资源节点和路由节点之间交换数据是异步时钟域通信，因此涉及到数据接口的同步问题，对于随机到达的数据，需要建立数据同步机制，通过 RAM 或者 FIFO 的缓存实现数据同步，可将前级模块提供的时钟作为写时钟，使用后级的基本时钟产生读信号，完成数据读出。通过这种方式实现全局异步局部同步（GALS）的设计。

本文在 FPGA 中构建异步 FIFO 来完成 ARM 资源节点和路由节点异步时钟域之间的数据传送。根据设计的 NoC 系统中通信节点整体架构，在 RNI 中构建两个异步 FIFO，FIFO 的存储深度设置为 8，宽度设置为 34 位，ARM 和路由节点通信时可以根据实际的数据位宽和 FIFO 进行连接。ARM 资源节点和路由节点的连接如图 2 所示。

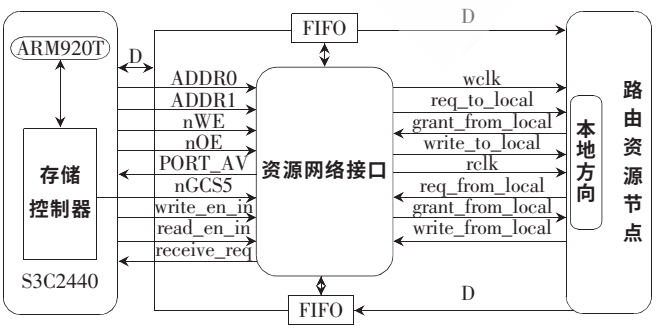


图 2 ARM 资源节点通信接口连接图

2.2 资源网络接口发送模块设计

ARM 资源节点发出的数据在片上网络中传输需要经过 RNI 对数据进行相应的打包处理，然后将数据送到路由节点中，经过路由传输后送达目的路由，由目的路由 RNI 对数据解包后发送给目的资源节点。为了能使数据包正确到达目的节点，一个完整的数据包在经过 RNI 后被分为若干个微片（flit），flit 分为 3 种类型，即头微片、数据微片和尾微片。头 flit 携带数据包源地址、目的地址、数据包长度等信息，尾微片代表着数据包的终结，数据微片表示传递的有效数据。路由节点根据数据携带的地址信息将资源节点发送的数据进行转发。发送模块主要由三个部分组成：输入缓存器（FIFO A）、组包器和控制器。

ARM 资源节点在向路由发送数据之前首先检测 PORT_AV 是否有效（高电平有效），若有效则将数据缓存到 FIFO A 中，在 S1 状态控制器检测到该 FIFO 中有数据就会向路由的本地方向发送数据传输的请求信号 req_to_local，并且进入 S2 状态。S2 状态判断本地方向是否给出应答信号 grant_from_local，若没有则返回 S1 同时清除请求信号。如果有应答信号则进入 S3 状态。S3 状态将输入缓存器中的数据送到本地方向，完成了数据从 ARM 资源节点到片上网络的传递。模块运行的状态转移图如图 3 所示。

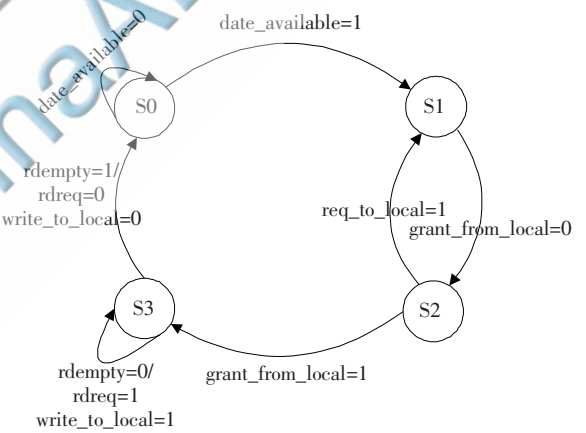


图 3 发送模块运行逻辑状态机

2.3 资源网络接口接收模块设计

ARM 处理器的资源网络接口接收模块对接收数据进行解包处理，提取有用的数据发给资源节点。接收模块主要由输出缓存器和应答器组成，异步 FIFO B 是输出缓存器，用它来存放从路由本地方向发送过来的数据，应答器是根据 FIFO B 存储状态对路由的请求给予对应的响应。接收过程具体的状态转移图如图 4 所示。

在 S0 状态中，数据传输到路由本地方会向 RNI 发送数据传送请求信号 req_from_local，在 S1 状态应答器根据输出缓冲器 FIFO B 的存储情况给出应答信号 grant_to_local。路由器在收到应答信号后将数据写到 RNI

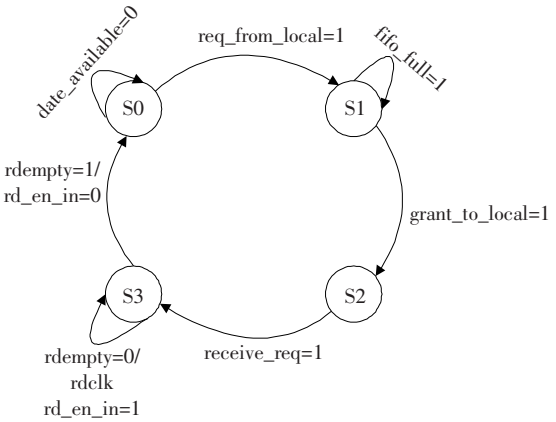


图 4 接收模块运行逻辑状态机

的 FIFO B 中。在 S2 状态中应答器检测到 FIFO B 中存在有效数据时就会向 ARM 资源节点发送读数据请求信号 receive_req。这个信号是直接连接到 ARM 的硬件中断上，ARM 资源节点捕获这个中断信号会在 S3 状态给 RNI 模块提供读数据时钟 rclk 和输出缓冲器的读使能信号 read_en_in，进而完成 ARM 资源节点接收片上网络传来的数据。

3 FPGA 设备的 Linux 驱动程序设计

ARM 处理器资源节点选用的是 SamSung 公司的 S3C24XX 系列处理器[4]，并向其中移植了嵌入式 Linux 操作系统。其内核功能强大，性能高效稳定且源代码开放，这使得设计者可以根据实际的需要对操作系统进行裁减以降低整个系统资源的开销和功耗。

为了使 FPGA 能够在 Linux 操作系统中工作，为其设计了相应的设备驱动程序。设备驱动程序是应用程序和实际设备之间的软件层。它为应用程序屏蔽了设备硬件工作的细节，在应用程序中只需要通过一组标准化调用完成对硬件设备的操作[5]。

本文 ARM 资源节点通过设备驱动实现和 FPGA 之间数据的通信。FPGA 设备驱动程序实现的主要功能是：(1) 初始化 FPGA 模块，注册 FPGA 设备，申请中断号等。(2) 通过 ioremap() 将资源网络接口中输入输出缓冲器的物理地址映射到内核虚拟空间。(3) 捕获资源网络接口发出的中断信号，并对中断事件进行处理。(4) 根据应用系统要求完成读写操作程序的设计。

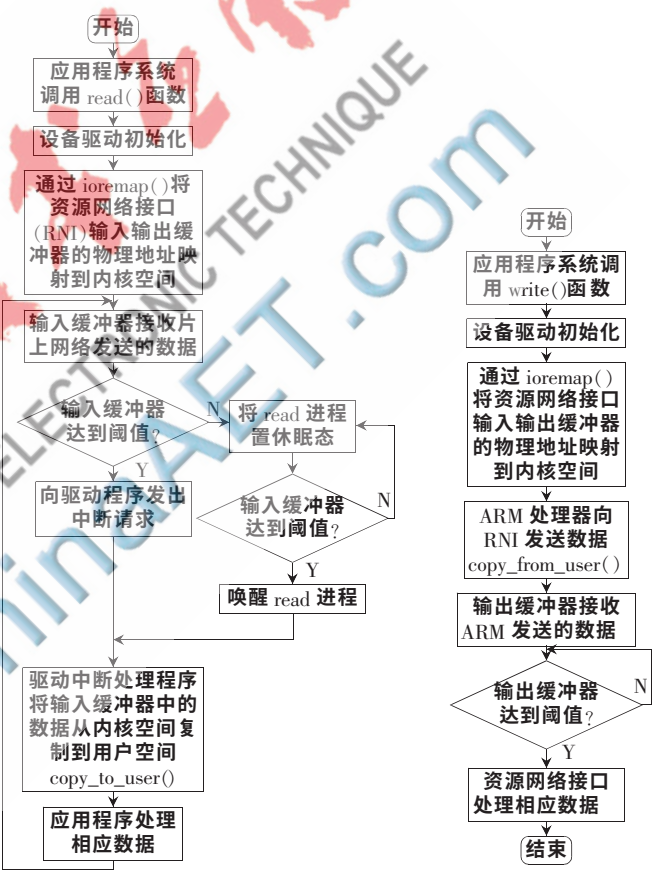
3.1 驱动硬件接口

FPGA 采用存储总线的方式直接连接在 S3C2440 的 AHB 总线上，将其作为 ARM 处理器的一个外部存储器进行读写操作，硬件连接如图 2 所示，其主要连接有 16 位宽的数据线，地址线以及读、写、中断和片选信号线参照 S3C2440 存储控制器的地址空间分布图，将 FPGA 设置到 bank5 的地址空间中，对应的片选信号线为 nGCS5，在 FPGA 内部构造了两个异步 FIFO (FIFO A、B) 作为资源网络接口的输入、输出缓冲器，ARM 通过访问异步 FIFO 完成和 FPGA 的数据通信。

3.2 驱动的软件设计

FPGA 设备驱动首先在初始化模块中向 Linux 操作系统申请设备号，申请成功后，该设备获得了系统分配的主设备号，并建立起与文件系统的关联。关联成功后，在应用层可以通过 read()、write()、ioctl() 等常规的文件操作对 FPGA 设备进行操作。

驱动程序为资源网络接口的输入、输出缓冲器分别分配物理地址，程序不能直接通过物理地址来访问 I/O 内存资源，必须通过内核函数 ioremap() 将缓冲器占用的物理地址映射到内核虚拟空间中。在此基础上结合系统读写网络资源接口的策略完成驱动程序设计，读写资源网络接口的程序流程图如图 5(a)、(b) 所示。



(a)设备驱动读操作流程图　　(b)设备驱动写操作流程图

图 5 设备驱动软件工作流程图

为了提高系统的效率，避免当设备资源不可用时，用户不停查询浪费 CPU 资源，在驱动程序中设计了阻塞操作，使用等待队列来实现阻塞进程的休眠和唤醒。应用程序进行 read() 函数的系统调用时，若 RNI 模块中的输入缓冲器中没有数据，驱动程序则将该读进程添加到等待队列头中，使该进程进入休眠状态，CPU 将资源让给其他进程。当输入缓冲器中的数据达到阈值时 RNI 就会向 ARM 资源节点发出读数据请求信号，ARM 资源节点通过中断来捕获这个通知，在驱动程序的中断处理函数唤醒休眠的读进程，将输入缓冲器中的数据中读取

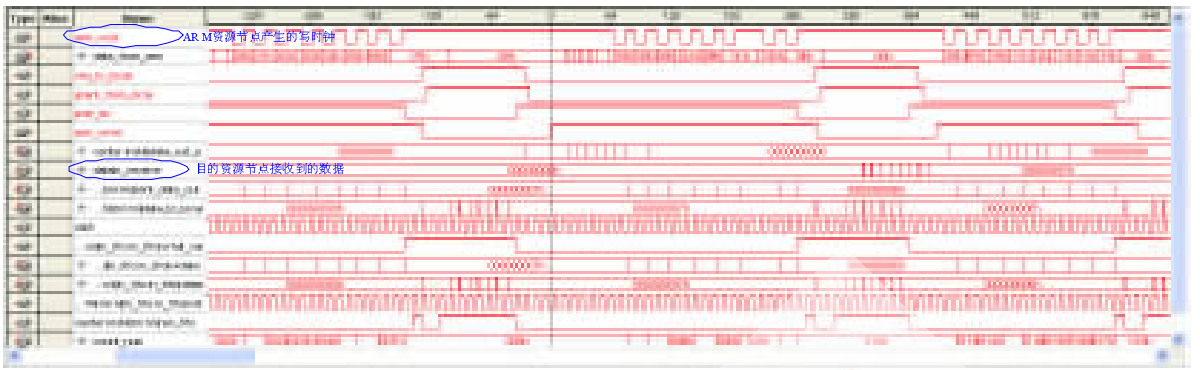


图6 Signal Tap II捕捉的接口发送时序图

到内核中，然后通过copy_to_user()将数据传递到用户空间进行相应的处理。类似的，应用程序中进行write()函数的系统调用时，ARM处理器通过copy_from_user ()将数据发送到RNI中。

4 资源网络接口和驱动功能验证

本设计在实验室自行开发的NoC硬件平台上进行了运行测试，对于运行结果使用Quartus II 11.0集成开发软件下的Signal Tap II嵌入式逻辑分析器进行测试。在测试程序中可以利用它捕捉通信接口相应的时序。

程序运行时，使用Signal Tap II观测ARM资源节点发送数据到路由节点的时序如图6所示。从图中观测可知数据data_to_local和datain_receive一致，说明数据传输正确性。

ARM资源节点通过接收NoC系统中其他资源节点发送来的数据，验证资源网络接口以及驱动通信接收功能的正确性。在程序运行时ARM资源节点将接收到的数据在终端打印出来，经观察终端显示的数据和该资源节点发送的数据是一致的。实际测试结果表明所设计资源网络接口和驱动功能的正确性。

本文给出了ARM处理器资源节点与NoC系统的网络资源接口设计，并且阐述在嵌入式Linux2.6.30内核下数据通信的驱动的设计和实现过程。系统在此设计的基础上充分利用ARM及其丰富的外设资源，完成了ARM处理器资源节点对NoC系统的其他资源节点进行控制以及数据处理等功能。ARM处理器有丰富的外设接口，能够稳定地运行移植到芯片中的Linux操作系统，以ARM处理器作为NoC片上多核系统的资源节点可以极大地拓展NoC系统应用空间。

参考文献

[1] BENINI L, MICHELI G D. Network on chip:A new SoC paradigm[J]. IEEE Computer, 2002, 3(1):70-78.

[2] Altera Inc. Cyclone IV device handbook [Z] . http://www.altera.com, 2010.

[3] 高明伦，杜高明.NoC：下一代集成电路主流设计技术[J].微电子学，2006，36(4).

[4] Samsung. S3C2440 32-bit Microcontroller User's manual[Z]. http://www.samsung.com,2004.

[5] CORBET J, RUBINI A, HARTMAN G K.Linux设备驱动程序(第3版)[M].魏永明，译.北京：中国电力出版社，2006.

(收稿日期：2013-03-05)

作者简介：

许川佩，女，1968年生，教授，硕士生导师，主要研究方向：集成电路测试及嵌入式系统应用。

孙义军，男，1986年生，硕士研究生，主要研究方向：计算机辅助测试。