# 基于SOPC的网络视频平台的构建

何 宾 何 军

(北京化工大学信息科学与技术学院 北京 100029)

**摘 要** 片上可编程系统(System on a Programmable Chip,SOPC)是可编程逻辑器件在嵌入式领域的重要应用。基于SOPC构建了一个网络视频传输及显示平台,该平台接收来自以太网的图像,经过处理后,通过数字视频接口传输到显示设备进行显示。详细描述了硬件平台和软件平台的创建,以及测试该系统的方法,在该平台上可以扩展视频监控、远端视频显示等应用。

**关键词** 以太网,FPGA,SOPC,LwIP协议,网络视频传输

## Design and Realization of the Network Video Platform on SOPC

HE Bin HE Jun

(School of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, China)

**Abstract** SOPC is an important application of programmable device in the embedded system. Based on SOPC, a general network video platform was introduced, which can receive images from the Ethernet, and process them, then transfer them to terminal equipment for display via digital video interface. In this paper, creation of hardware and software platform and a method to test this system were described in detail. On this platform, a variety of network video application can be achieved, such as video surveillance, remote video display.

**Keywords** Ethernet, FPGA, SOPC, LwIP, Network video transmission

SOPC是一种特殊的嵌入式系统,即在单芯片上完成整个嵌入式系统的绝大多数功能,是一种可裁减、可扩充、可升级、可编程的系统,具有很好的扩展性和灵活性。作为世界上知名的可编程器件厂商,美国Xilinx公司提供了完整的SOPC解决方案,可以选择软核和硬核处理器以满足不同的性能需求。

在SOPC系统中,本地存储器、处理器总线、内部外设、外设控制器和存储器由FPGA的通用逻辑实现,在单芯片上实现整个系统,减少了电路板占用面积,降低了系统功耗,也使得基于嵌入式处理器的串行处理和可编程逻辑的并行处理完美结合,让整个系统设计变得更加灵活、更加高效,这大大提高了嵌入式系统的整体性能。

在SOPC系统设计阶段,已经从传统的以硬件描述语言HDL为中心的硬件设计,转移到了以C语言进行功能描述为中心的软件设计,这使得软硬件的协同设计得以发展,软硬件的开发可以同时进行。SOPC系统设计越来越依赖于各种知识产权核(Intellectual Property core,IP),通过IP核复用技术极大地提高了系统的设计效率[1,2]。

SOPC平台的产生,为FPGA提供了更加广阔的舞台,典型的是基于网络的应用,即通过网络实现数据传输、远程控制、数据共享等功能。为了在内存资源有限的嵌入式系统中使用TCP/IP网络协议,需要对标准的TCP/IP协议进行优化,减少内存资源占用和数据复制等,轻型网络协议(Light Weight Internet Protocol,LwIP)就是由瑞典计算机科学院开发,适合在嵌入式领域使用的一种精简的、开源的TCP/IP协议[3-5]。

嵌入式系统需要用显示接口将信息直观地展现出来,本设计中通过使用先进的数字视频接口(Digital Visual Interface,DVI),使得图像的色彩更纯净、更逼真,进一步提升了人们的视觉感受。

## 1 系统平台的硬件设计

### 1.1 系统结构

图1给出了网络视频传输的结构,设计中使用了Xilinx公司提供的32位MicroBlaze软核处理器,该处理器是采用功能强大的32位流水线、精简指令集(Reduced Instruction Set Computer,RISC)结构,包含32个32位通用寄存器和一个可选的32位移位寄存器、32位指令和数据总线的哈佛结构,可以全速执行片上存储器和外部存储器中存储的程序,MicroBlaze处理器提供了包括本地存储器总线(Local Memory Bus,LMB)、处理器本地总线(Processor Local Bus,PLB)、快速单向链路总线(Fast Simplex Link,FSL)在内的各种互联结构,用来实现与其他外设资源的连接。

本设计的所有外设都连接到PLB总线上,PLB总线主要由总线控制单元、看门狗时钟、单独的寻址和读写数据路径单元组成。PLB总线包含32位地址总线,32/64/128位数据总线,采用轮询或者优先级的仲裁方式,且支持4级动态请求级别,仲裁时间小于3个时钟周期,通过使用该总线直接连接外设,减少了延迟时间,加快了数据的处理速度,充分发挥了系统的整体性能[6]。

何 宾(1975-),男,博士,讲师,主要研究方向为嵌入式系统、高速实时信号处理,E-mail:hebinhello@263.net;何 军(1986-),男,硕士,主要研究方向为嵌入式系统、计算机应用技术。

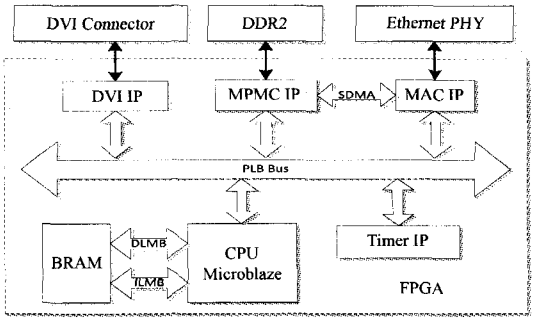


图 1　系统结构图

通过 PLB 总线将该设计中的定时器模块(Time IP)、以太网控制器模块(MAC IP)、存储器管理模块(MPMC IP)、数字视频接口控制器模块(DVI IP)连接到处理器上,并通过 FPGA 的管脚分配分别连接到外部物理转换芯片上,由片上处理器对这些 IP 核进行配置及操作,来实现对外部模块控制。

### 1.2　系统主要模块的功能

处于核心地位的 MicroBlaze 软核处理器用于处理来自各个外设的信息,并对这些外设进行控制。处理器主要用来控制以太网控制器实现 TCP/IP 数据包的接收、对接收的图像数据进行解码和控制数字视频接口控制器实现图像数据的显示[7]。

SOPC 系统运行程序有两种方式,一是使用 FPGA 的 BRAM 资源,其通过 LMB 总线与 MicroBlaze 软核处理器相连;二是使用外部存储器,其通过 PLB 总线与 MicroBlaze 软核处理器相连;使用 FPGA 内部的 BRAM 资源,这些 BRAM 运行速度快,这就进一步提升了指令的执行速度,从而提升系统的整体性能,如果使用外部存储器,在取址执行时需要通过总线仲裁来获取总线的使用权,因为外部存储器也作为其中的一个外设,会降低系统的性能,所以把程序存储在片内 BRAM 中是本系统设计的最佳选择。

设计中的各种 IP 核提供了丰富的功能,其中内存控制器实现对内存的访问;以太网控制器完成网络数据的收发;数字视频接口实现图像数据的显示,实现行同步、场同步时钟的产生、数据转换、显示配置等功能。在该设计中需要传输大量数据,使用 DMA 技术,通过 SDMA 接口把以太网接收到的图像数据存储传输到 DDR2 内存中,等待处理器处理完成以后交给 DVI 控制器进行图像显示,从而提高了系统的性能。

### 1.3　硬件平台的创建

硬件平台的创建,用于实现各外设 IP 的功能配置及各 IP 在系统中的存储空间分配、总线连接方式、端口引脚配置。通过使用 Xilinx 公司提供的嵌入式开发套件(Embedded Development Kit,EDK)完成所有的功能配置。图 2 给出了硬件平台的创建流程。

1)通过 Xilinx 公司的 EDK 工具,使用基本系统向导生成器(Base System Builder,BSB)创建基本的硬件平台,根据实际使用的硬件资源选择板描述文件,完成 SOPC 的最基本功能的配置。设置 Microblaze 软核处理器的运行时钟为 125MHz,指令和数据内存大小为 256k。

2)外设 IP 的添加与配置,添加内存控制 IP 核、以太网控制 IP 核、串口 IP 核、时钟 IP 核、DVI 控制器 IP 核,完成添加后将各 IP 连接到 PLB 总线上,并给各 IP 核分配内存资源,将系统的外部信号设置为外部端口,并在 UCF 文件中添加管脚约束,然后保存以上工程。

3)综合和实现,对构建的硬件平台进行综合,生成可下载到 FPGA 运行的比特流文件,这就完成了硬件平台的构建。

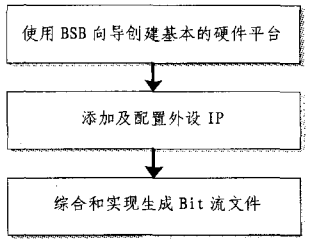


图 2　硬件平台的创建

## 2　系统平台的软件设计

该系统的软件由 Xilkernel 操作系统、LwIP 支持库、驱动程序、应用程序 4 部分组成,在 Xilinx 公司提供的 SDK 平台下完成所有软件的设计。操作系统完成系统资源管理及调度,驱动程序为物理层与应用层提供数据通道,并为应用层提供函数接口,从而为应用软件的开发带来便利。通过使用驱动程序,屏蔽了底层的硬件操作,应用程序通过调用 LwIP 库函数接收来自网络的图像数据,图像经过处理后,由 DVI 控制器来控制图像在显示器上的显示。

### 2.1　Xilkernel 嵌入式操作系统

Xilkernel 是 Xilinx 公司用于 EDK 系统的小型化、模块化的嵌入式操作系统,支持可移植操作系统接口(Portable Operating System Interface of Unix,POSIX),可以根据应用需求进行裁剪,该系统占用 CPU 资源少,运行速度快,并具备嵌入式操作系统的基本特征,即多任务处理、进程间通信、基于优先级的调度方法、同步以及中断处理等[8]。

Xilkernel 本身并不带 TCP/IP 协议栈,但是提供与 LwIP 库的良好接口,可以根据实际的应用,通过加载和卸载不同的功能模块来实现内核的高扩展性。EDK 工具集成了内核功能,通过 EDK 工具对内核功能进行配置,使整个开发过程变得简单、高效。

Xilkernel 的基本执行单元为任务处理,且调度也在任务处理器级别上完成,它没有线程组的概念,即没有进程的概念,所有的线程性质都一样,可以完全占用相应 CPU 资源。其支持基于优先级调度和轮询调度,这两种都是全局调度准则,在生成内核的时候必须确定,在基于先后顺序的轮询模式中,所有任务都具有相同的优先级,只有一个单独的等待队列,CPU 按顺序执行队列中的任务。

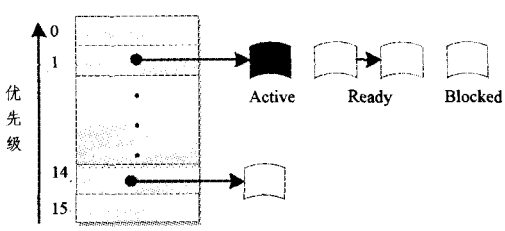


图 3　优先级调度模式原理图

在基于优先级调度的模式中,不同优先级的任务有不同的等待队列,优先级系数为 0 的线程具有最高优先级,优先级系数越高意味着优先级越低,CPU 先执行高优先级队列,然后执行低优先级队列,队列长度和优先级可以通过软件进行

设置，图 3 清晰地显示了基于优先级调度的调度策略。

在本设计中，所创建的线程都使用系统默认的优先级，即应用程序的线程都在同一优先级运行，系统线程的优先级高于应用程序优先级，这使得系统线程能优先运行。

### 2.2 LwIP 协议栈

LwIP 是 TCP/IP 协议栈的一个实现，在保持 TCP 协议主要功能的基础上减少对存储器资源的占用，简化了处理过程和对内存的要求，并且对 API 进行了裁剪，减少了复制数据所需的时间和内存空间占用。一般情况下，它只需要几百字节的 RAM 和 40k 左右的 ROM 就可以运行，这使 LwIP 协议栈非常适合在嵌入式系统中使用[9,10]。

LwIP 把所有的协议封装到一个独立的过程中，从而与操作系统内核分开，应用程序可能驻留在 LwIP 中，也可能在单独的过程中，TCP/IP 协议栈和应用程序之间的通信可以通过函数调用实现，把 LwIP 作为一个过程的主要优点是便于在不同操作系统上移植，图 4 显示了数据的接收过程，该过程是通过调用 LwIP 库函数来实现的。

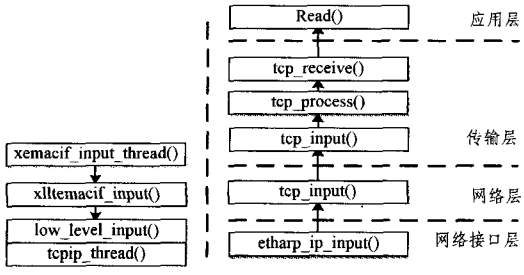


图 4 数据的接收过程

底层数据包的接收线程 xemacif_input_thread 在应用程序启动的时候被创建，当有以太网数据包到来时，通过调用 xlltemacif_input 函数接收数据包，一次只能处理一个数据包，把接收的数据包交给 tcpip_thread 线程处理，线程中通过函数 ip_input 进行 ip 头的校验、并检查包的大小是否合法，然后去掉 ip 头，把数据交给函数 tcp_input 处理，该函数完成 tcp 头的校验，然后让函数 tcp_process 处理，完成 tcp 层的超时、窗口大小的验证等操作，最后把接收的数据放到 pbuf 队列里，由应用程序的线程调用函数 read 进行读取，然后把数据交给应用层的数据处理函数进行处理。

### 2.3 应用程序的实现

在以上设计完成后，就可以进行应用程序的设计，该系统设计的软件部分包括两个部分：一方面进行图像数据的接收；另一方面进行数据图像的显示。这部分程序运行在 Microblaze 软核处理器上，由 Xilkernel 操作系统进行调度，图 5 给出了应用程序线程的创建过程。

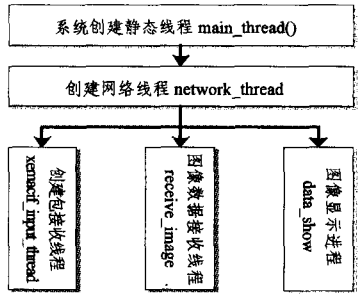


图 5 应用程序线程的创建

首先由操作系统提供的线程创建函数 sys_thread_new() 创建静态启动线程 main_thread，该线程首先对 Lwip 进行初始化，然后创建网络线程 network_thread；在网络线程中首先完成 IP 地址、网关、掩码的设置，然后调用系统函数 xemac_add 把新建的网络接口添加到网络接口队列里。

以上操作完成以后，创建数据包接收线程 xemacif_input_thread、应用程序图像数据接收和处理进程 receive_image、图像显示线程 data_show。数据包接收线程是用来接收来自网络接口的数据包，在整个程序的运行过程中一直存在，当有网络数据到来时通过中断调用进行处理。图像数据接收线程接收完后进行处理，然后交给图像显示进程进行显示。

接收图像数据包并处理线程首先接收来自 tcpip_thread 进程提交的数据，把接收的数据放到内存中，数据存放的首地址由用户设定，可以在应用程序中通过宏定义进行设置，当接收到一个处理段数据后，对整段的数据进行处理，然后按照显示的格式放入内存中，记录该存放的首地址和数据长度，地址长度信息存储在全局变量中，把该地址交给显示进程，以便图像显示线程读取数据进行显示。

图像显示线程 data_show 通过读取接收和处理线程所设置的全局变量来判断是否有数据显示，当有数据显示时，按照 DVI 的显示规范将数据传输到显示器上进行显示。其数据是 24 位，这需要在时钟的两个沿发送，根据实际需要的传输模式来配置 CH7301C 的寄存器。

整个系统的运行中包含 3 个线程，接收线程 xemacif_input_thread、图像数据接收和处理进程 receive_image、显示线程 data_show。应用程序运行时创建以上 3 个线程，并由操作系统进行调度。当没有数据显示时，显示进程 data_show 进入等待队列，当有数据显示，且当前 CPU 处于空闲状态时，则执行显示线程，使用的操作系统的线程操作有 7 个状态：线程创建、线程准备好、线程运行、线程等待、线程超时、线程等待资源超时、线程结束。系统根据调度准则进行线程的调度，其线程的状态在这 7 个状态中进行转换。

## 3 发送端软件设计

为了对以上的设计进行验证，在 PC 机开发了数据发送软件，为本系统提供数据来源，该软件主要实现了打开用户指定的图像数据文件，通过网络发送到用户指定的 IP 地址、给出发送的状态信息功能。图 6 给出了图像通过网络发送的过程。

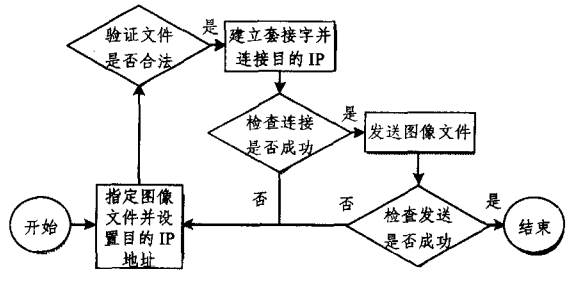


图 6 发送软件功能

本设计在 Xilinx 公司的 XUPV5-LX110T 开发版上进行验证，通过数据发送软件给该系统发送图像数据，经过接收处理以后，在显示器上能清晰地看到发送的图像。

**结束语** 在该设计中使用的是 Microblaze 软核处理器，该处理器的性能可以满足一般要求，如果使用复杂的算法或

（下转第 446 页）

4 个交换机，因此其消耗时延更大，这不适合于 GOOSE 网对实时性要求高的特点。

3.2.2 场景二：变电站进行正常操作

本仿真中，各个智能单元、测控装置以事件重发的时间间隔，即 0ms、1ms、2ms、4ms 的时间间隔发送 GOOSE 报文，且发送 GOOSE 报文的开始时间、持续时间、终止时间等采用的是 uniform 分布，具有一定的随机性。表 2 和表 3 分别是 GOOSE 网中智能单元至测控装置的时延和测控装置至智能单元的时延。

表 2 GOOSE 网智能单元至测控装置时延

| 拓扑 | 线路设备时延(us) | | | 母线设备时延(us) | | 母联设备时延(us) | |
|---|---|---|---|---|---|---|---|
| 星形 | 47.1 | 47.1 | 47.1 | 71.7 | 47.1 | 47.1 | |
| 环形 | 47.1 | 47.1 | 47.1 | 108.6 | 47.1 | 47.1 | |

表 3 GOOSE 网测控装置至智能单元时延

| 拓扑 | 线路设备时延(us) | | | 母线设备时延(us) | | | 母联设备时延(us) | |
|---|---|---|---|---|---|---|---|---|
| 星形 | 47.1 | 47.1 | 47.1 | 71.7 | 71.7 | 71.7 | 47.1 | 47.1 |
| 环形 | 47.1 | 47.1 | 47.1 | 96.3 | 102.9 | 96.3 | 47.1 | 47.1 |

从表 2 和表 3 可以看出，星形网络拓扑结构与环形网络拓扑结构中，同一个交换机内的智能单元与测控装置的双向通信的端到端时延值相同。但是母线智能单元与母线测控装置的端到端时延，星形网络拓扑的性能要好一些，这仍然是由于环形网络中的母线智能单元与母线测控装置通信需要跨越比星形网络拓扑更多的交换机，传播时延增加，从而导致了端到端时延的增加。另外，与场景一中的仿真结果相比可知，虽然减少发送的时间间隔，但是并没有导致结点之间的端到端时延大量的增加，这是由于 GOOSE 网的网络带宽是 100Mbit/s，而 GOOSE 报文较小，网络可用带宽充足，网络不会发生拥塞。因此，不会造成端到端时延值的增大，保证了 GOOSE 网的实时性。

3.2.3 场景三：变电站发生事故

本仿真中，各个 GOOSE 结点以事件重发的时间间隔，即 0ms、1ms、2ms、4ms 的时间间隔发送 GOOSE 报文。对仿真中各结点发送 GOOSE 报文的时间分布设置为 uniform 分布，对 GOOSE 网的星形网络拓扑结构和环形网络拓扑结构中各个结点的端到端时延的变化情况进行对比分析。

表 4 和表 5 分别是 GOOSE 网中保护单元至智能单元和智能单元至测控装置的端到端时延。

表 4 GOOSE 网保护装置至智能单元时延

| 拓扑 | 线路设备时延(us) | | | 母线设备时延(us) | | | 母联设备时延(us) | |
|---|---|---|---|---|---|---|---|---|
| 星形 | 56.6 | 57.6 | 57.1 | 72.0 | 72.0 | 72.0 | 47.3 | 47.4 |
| 环形 | 47.3 | 58.2 | 50.1 | 96.7 | 127.6 | 96.7 | 47.3 | 47.4 |

表 5 GOOSE 网智能单元至测控装置时延

| 拓扑 | 线路设备时延(us) | | | 母线设备时延(us) | | 母联设备时延(us) | |
|---|---|---|---|---|---|---|---|
| 星形 | 47.3us | 47.2us | 49.1us | 72.0us | 53.2us | 47.4us | |
| 环形 | 47.1us | 47.1us | 49.4us | 105.0us | 47.3us | 47.4us | |

从表 4 和表 5 可以看出，在 GOOSE 网星形网络拓扑与环形网络拓扑中，同一个交换机内的保护装置至智能单元与智能单元与测控装置通信的端到端时延值很接近。同时，环形网络中的母差保护装置至母线智能单元与母线智能单元至母线测控装置的通信需要跨越比星形网络拓扑更多的交换机，传播时延与星形网络拓扑相比较大。

**结束语** 本文分析了 GOOSE 网的传输机制和在智能变电站中的工作方式。采用 OPNET 仿真软件对 GOOSE 网的星形网络拓扑和环形网络拓扑中各结点通信的端到端时延进行了对比分析，仿真结果表明，GOOSE 星形网络拓扑较环形网络拓扑实时性更好，且两者的网络时延都在 150us 内，具有非常好的实时性。

## 参 考 文 献


[1] 高翔，周健，周红，等. IEC61850 标准在南桥变电站监控系统中应用[J]. 电力系统自动化，2006，30(16)：105-107

[2] GB/T 13729-2002 远动终端设备[S]

[3] 孙军平，盛万兴，王孙安. 基于以太网的实时发布者/订阅者模型研究与实现[J]. 西安交通大学学报，2002，36(12)：1299-1302

[4] 王松，黄晓明. GOOSE 报文过滤方法研究[J]. 电力系统自动化，2008，32(19)：54-57

[5] 李小滨，韩明峰. GOOSE 实时通信的分析与实现[J]. 电力系统保护与控制，2009，37(10)：59-62

[6] 柯善文，刘曙光，何能，等. 关于变电站 GOOSE 报文传输的研究[J]. 继电器，2007，35：308-310

[7] 徐成斌，孙一民. 数字化变电站过程层 GOOSE 通信方案[J]. 电力系统自动化，2007，31(19)：91-94

[8] 陈敏. OPNET 网络仿真[S]


(上接第 439 页)

者需要更高的处理性能，可以使用 Xilinx 公司提供的硬核处理器。设计中采用 IP 复用技术，而不像传统的设计那样，在更改其中某部分设计时需要更改整体设计，这使得修改和维护更加方便、更加高效、节约成本和开发时间。

该平台可以应用在许多领域，如远程监控、远端视频显示，可以根据实际的项目需求进行修改、升级、设计和开发特定的应用。

## 参 考 文 献


[1] 何宾. EDA 原理及应用[M]. 北京：清华大学出版社，2009

[2] 何宾. 片上可编程系统原理及应用[M]. 北京：清华大学出版社，2010

[3] 胡龙腾，田雨. 基于 LwIP 的嵌入式以太网系统的设计与实现[J]. 数字技术与应用，2010，24(8)：59-61

[4] 章智慧，白瑞林，沈宪明. LwIP 协议栈在 SOPC 系统中的实现[J]. 计算机工程与设计，2007，28(6)：1378-1380

[5] 江晋剑. 基于 SOPC 构建的嵌入式 Web 服务器设计[J]. 自动化与仪器仪表，2009，3：22-26

[6] XILINX. Processer Local Bus(PLB v4. 6)(v1. 05. a) [P/OL]. http://www. xilinx. com，2010

[7] XILINX. MicroBlaze Processer Reference Guide [P/OL]. http://www. xilinx. com，2009

[8] XILINX. OS and Liberaries Document Collection [P/OL]. http://www. xilinx. com，2009

[9] XILINX. LwIP Library(v3. 00. a) [P/OL]. http://www. xilinx. com，2008

[10] Tanenbaum A S. 计算机网络(第四版)[M]. 潘爱民，译. 徐明伟审. 北京：清华大学出版社，2004