

基于 SOPC 技术的视频点播系统设计

王建国, 洪胜峰, 慕声波, 吴书铭

(中国海洋大学工程学院, 青岛 266100)

摘要: 提出一种以 Altera 公司高性能的 FPGA 芯片为平台, 利用 SOPC 技术和双 Nios II 处理器设计实现 16 终端视频点播系统的方法。阐述了在可编程逻辑设计中使用“乒乓操作”实现高速数据传输的设计思想与实现方法。介绍多 CPU 系统中各 CPU 之间同步通信技术的设计方法。

关键词: 可编程片上系统技术; 视频点播; NiosII 软核处理器; 双处理器系统

Design of Video-on-demand System Based on SOPC

WANG Jian-guo, HONG Sheng-feng, QI Sheng-bo, WU Shu-ming

(Engineering College, Ocean University of China, Qingdao 266100)

【Abstract】 This paper introduces a new method based on system on programmable chip(SOPC) technology and dual NiosII processor to design VoD system with 16 terminals. It provides a technique about how to apply the “Ping-pong” theory to achieve high speed data transfer, and how to communicate synchronously between two CPUs.

【Key words】 system on programmable chip(SOPC); video-on-demand(VoD); NiosII IP core; dual processor system

可编程片上系统(system on programmable chip, SOPC)是一种特殊的嵌入式系统, 因为: (1)SOPC是片上系统(SoC), 即由单个芯片完成整个系统的主要逻辑功能; (2)SOPC是可编程系统, 具有灵活的设计方式, 可剪裁、可扩充、可升级, 并具备软硬件在系统可编程的功能; (3)SOPC结合了SoC和FPGA的优点, 涵盖了嵌入式系统设计技术的全部内容; (4)SOPC结合了以处理器和实时多任务操作系统(RTOS)为中心的软件设计技术, 以PCB和信号完整性分析为基础的高速电路设计技术; (5)SOPC涉及目前已引起普遍关注的软硬件协同设计技术^[1-2]。

本文将 VOD 视频点播系统为例, 介绍了采用 SOPC 技术, 以及多 CPU 及其同步通信技术实现的视频点播系统主机部分的设计思想和工作原理。

1 不同终端所需数据的读取与发送

1.1 系统整体结构

以乒乓操^[2]作为核心设计思想的多终端视频点播系统的主机原理框图如图 1 所示。

CPU1 与 CPU2 为高性能的 NiosII 软核处理器。其中, CPU1 为系统主控制器, 负责对整个系统进行统一管理。其任务包括: 接收各终端的点播信息并进行处理, 初始化硬盘, 控制“DMA1”模块将硬盘数据送到任意一片双口 RAM 中及对 CPU2 工作节拍的调节。CPU2 为从控制器, 其主要功能是: 控制“DMA2”模块, 从“双口 RAM1”或“双口 RAM2”中读出各个终端所需数据, 并送到 AM7968 的数据口上进行发送。

“DMA1”模块是由 CPU1 管理的 DMA 控制器, 主要负责将点播数据从硬盘读出, 并写入到任一双口 RAM 中去。CPU1 对该模块的初始化主要有: 当前所写入双口 RAM 的选择、写入数据的目标地址以及所写入连续数据的字(16bit/word)数。当 CPU1 完成对“DMA1”和硬盘的初始化

后, “DMA1”便等待由硬盘发起的数据传输, 并在传输启动后将硬盘数据口上的数据写入双口 RAM 中去。

“DMA2”模块是由 CPU2 管理的 DMA 控制器, 主要负责将双口 RAM 中对应于各个终端的点播数据分块读出并打包, 然后通过 AM7968 芯片发送到各个终端上。

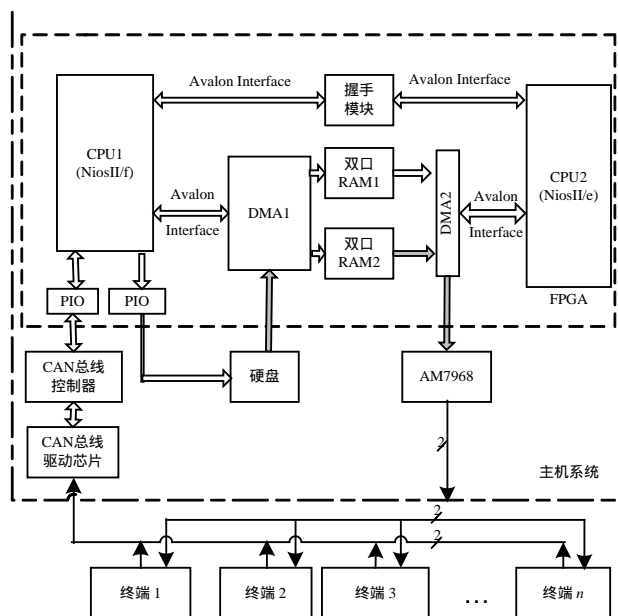


图 1 多终端视频点播系统主机设计框图

各个终端与主机之间有两个独立的信息通道分别负责点播请求和视频数据的传输。其中, 点播请求和各终端的状态

作者简介: 王建国(1954 -), 男, 教授, 主研方向: 智能仪器仪表, 嵌入式技术; 洪胜峰, 硕士研究生; 慕声波, 副教授; 吴书铭, 硕士研究生

收稿日期: 2006-12-19 **E-mail:** hongshfeng27@126.com

信息的传送由 CAN 总线实现。CAN 总线系统为时间同步的多点接收系统，支持多主机，具有“非破坏性逐位”仲裁，以及总线接口简单、成本低廉等优点，因此，选用 CAN 总线作为终端向主机发送点播请求和工作状态等信息的传输通道。

主机向终端发送视频数据是通过由高速串行化器件 AM7968/7969 组成发送/接收模块实现的。使用该串行化器件可使视频数据的传输速率达到 175Mb/s，从而满足 16 个终端同时工作的需要。

1.2 系统工作原理

系统进入正常工作后，CPU1 通过 CAN 总线接收各个终端的点播请求，并根据各个终端不同的需要，从硬盘内取出各终端点播的视频数据，按照固定格式存放到双口 RAM 中。当其中一片双口 RAM 被写满后，CPU1 通过“握手模块”向 CPU2 发出视频数据准备就绪信号。当 CPU2 完成对当前双口 RAM 中数据的发送后，若查询到该就绪信号，便转向另一片双口 RAM 执行数据读取和发送任务。

CPU2 开始执行数据发送程序，同时将自身工作状态通过“握手模块”回传给 CPU1。CPU1 接收到 CPU2 的回传信息后，随即开始从硬盘中读取各终端的下一部分数据写入另一片空闲的双口 RAM 中，操作完成后向 CPU2 发出新的数据就绪信号。

按照上述方式，CPU1 不断地从硬盘中读取各终端的点播数据，并交替写入两片双口 RAM 中；CPU2 也与 CPU1 同步从另一片双口 RAM 中读取数据并发送到各个终端，如图 1 中灰色箭头所示。各个终端所点播的视频数据就在两个 CPU 的协同乒乓操作下实现了无损缓冲和发送，从而实现乒乓操作。

1.3 双 CPU 的交互式同步控制——“握手模块”设计

“握手模块”是为了使 CPU1 与 CPU2 同步协调工作，以满足实现乒乓操作的要求而设计的。该模块设计如图 2 所示。

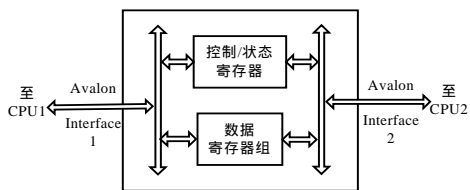


图 2 “握手模块”设计框图

该模块以 CPU 外扩寄存器访问方式为设计模型，将模块内部的 8 个寄存器分别定义为 1 个控制/状态寄存器、3 个数据寄存器和 4 个保留寄存器。模块中所用寄存器都为双口，分别接在各自独立的两套 Avalon 总线接口上。其中，控制/状态寄存器用于实现 CPU1 向 CPU2 发送启动数据传输命令以及两片双口 RAM 的片选信号。数据寄存器组用来实现两个 CPU 之间的参数传递，包括 CPU2 发送的数据包大小、每片双口 RAM 中分配给每个点播终端的数据空间大小、每个节点是否需要发送点播数据以及发送每个数据包之间的定时周期等。寄存器具体定义如表 1 所示。

该模块使用 VHDL 语言完全遵循 Altera 公司的 Avalon 接口规范^[3-4]编写。该模块中 8 个寄存器都能被两个 CPU 所访问，但访问的优先级却有所不同。其中，CPU1 写寄存器操作的优先级最高，接下来依次是：CPU1 读寄存器，CPU2 写寄存器，最后是 CPU2 读寄存器。当 CPU1 或 CPU2 对该模块

的读写操作完成后，通过该模块向对方申请中断，以请求对方执行相应的操作。该模块程序流程如图 3 所示。

表 1 “握手模块”内部寄存器定义

偏移地址	寄存器定义	寄存器内容
0	数据包参数寄存器	31~16:终端数据包个数 P_max; 15~0 数据包长度 P_length
1	终端状态寄存器	31~16:保留; 15~0:各终端点播状态
2	定时周期寄存器	31~0:定时周期
3	控制/状态寄存器	31~3:保留; 2:RAM2_CS; 1:RAM1_CS; 0: Irq_starte
4~7	保留	31~0:保留

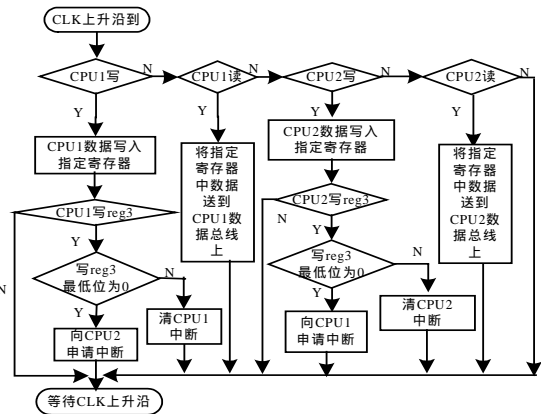


图 3 “握手模块”程序流程

1.4 数据的读取与分配

如图 4 所示，设两片双口 RAM 均为 16KB，将其分为 16 个连续数据空间，分别作为 16 个终端的数据缓冲区。为方便 CPU2 从双口 RAM 中读取数据并发送到对应的各个终端，可将每个数据区划分为若干个大小相同的数据包。

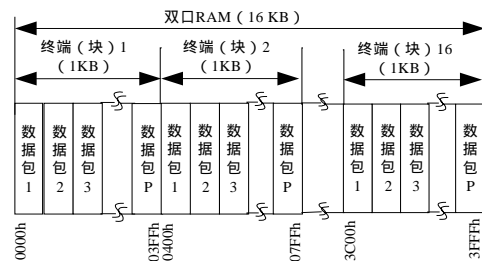


图 4 双口 RAM 的空间分配

在工作状态下，CPU1 根据终端发出的点播请求，控制 DMA1 模块将该终端的数据从硬盘读出，并写入为该终端分配的 RAM 空间中(共 1KB)，直至该部分空间被填满。当 CPU1 完成该终端数据块的读取后，将扫描下一个终端的工作状态，若申请点播，则重复上述步骤，从硬盘中读取相应数据送入缓存区；若无点播信息，则跳过该终端，继续扫描下一终端，直到 CPU1 完成对全部终端的扫描。

1.5 数据的发送与接收

当 CPU1 完成对任一片双口 RAM 的数据更新后，通过“握手模块”向 CPU2 发出数据发送请求。当 CPU2 进入空闲状态后，将该双口 RAM 中的数据读出并发送到相应终端。

在系统的高速视频压缩数据传输通道的设计上，本文采用总线式的拓扑结构，由主机以“广播”方式向所有终端发送数据包，各终端以“收听”方式对数据包进行选择接收，如图 1 所示。

由于所有终端播放的文件均为 MPEG-I 格式，要求播放的数据流速率为 1.416Mb/s，即 177KB/s，而在传输介质允许

的条件下, AM7968/7969 的最大传输速率可高达 175Mb/s。因此, 本设计利用 AM7968/7969 的高速传输特性, 采用时分多路复用(TDM)技术, 各终端的点播数据按一定规模分块打包排序, 并根据不同的接收终端对每个发送的数据包加入包头 ID, 然后定时向总线上广播出去。其数据传输格式如图 5 所示。

在图 5 中, 定时周期的长度是由数据包的大小唯一确定的。由于每个终端都需要满足 1.416Mb/s 的数据率要求, 因此, 定时周期可由如下公式计算得到:

$$T = P \div 177(\text{KB/s})$$

其中, T 为定时周期, 单位为 s; P 为每个数据包的大小, 单位是“KB”。

若 AM7968/7969 的传输频率设定为 80Mb/s, 当 16 个终端同时处于点播状态时, 连续发送 16 数据包的时间为

$$t = [(P \times 8) \div 80000(\text{Kb/s})] \times 16 = P \div 625(\text{KB/s})$$

其中, t 为连续发送 16 个数据包的时间, 单位为 s。

由上述两公式比较可知 $t < T$ 。因此, 在一个定时周期内, 主机可以通过 AM7968 实现将所有终端的相同序号的数据包依次进行“广播”发送。若工作过程中, 某终端无点播数据需要发送, 则该终端的发送时间段将被保留, 如图 5 中终端 3 的数据包 2 以及终端 1、终端 2 的数据包 3 所示。

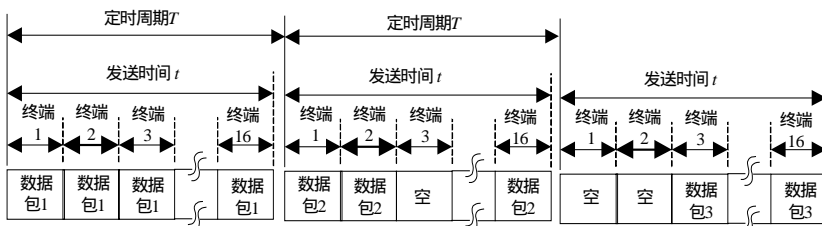


图 5 数据发送格式原理图

当终端通过 AM7969 接收到信道上的广播数据包后, 将首先判断数据包的包头 ID 号, 如果该数据包的 ID 号与本终端的 ID 编号相同, 则本终端启动接收该数据包, 否则将继续对数据传输通道进行监听。

2 软件设计

2.1 CPU1 程序流程

CPU1 主程序流程如图 6 所示。

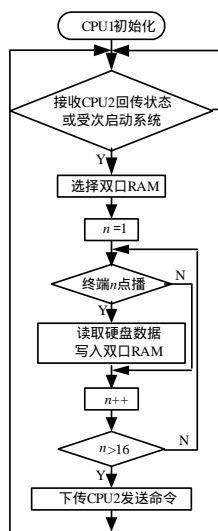


图 6 CPU1 主程序流程

该流程具体如下:

(1)CPU1 完成初始化后便进入等待状态;

(2)当 CPU1 接收到任一节点的点播请求, 系统进入点播运行状态。当 CPU1 进入运行状态后, 将依次扫描各个终端的状态, 若有点播请求, 则从硬盘中读取相应数据写入空闲的双口 RAM 中, 直至 16 个终端被依次操作完毕;

(3)CPU1 将点播终端编号、数据包大小、写入的双口 RAM 序号、定时周期时间长度等参数上传给 CPU2, 同时向 CPU2 发出启动发送命令;

(4)CPU1 将进入等待状态, 直至收到 CPU2 的回传状态, 并立即切换到另一片空闲的双口 RAM, 再次根据各终端点播状态, 从硬盘中读取下一部分数据, 写入该双口 RAM 中。

2.2 CPU2 程序流程

CPU2 主程序流程如图 7 所示。其中, P_{\max} 表示双口 RAM 中每个终端的数据缓存区所等分的数据包个数。

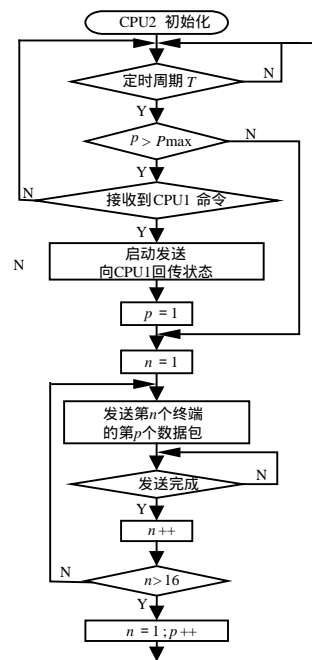


图 7 CPU2 主程序流程

CPU2 的主程序是以定时周期 T 为执行周期的。每次定时时间到, CPU2 都会判断当前发送的数据包序号。如果当前发送数据包序号大于 P_{\max} 值, 则表示当前使用的双口 RAM 中的数据已被全部发送, CPU2 将判断是否已经收到 CPU1 下传的新数据发送请求, 若未收到, CPU2 将停止发送, 等待下一次定时周期到再查询, 直至接收到新的发送请求; 若已收到 CPU1 下传新的发送请求, CPU2 将根据“握手模块”中的参数, 切换到更新的双口 RAM, 将其中数据读出并在数据信道上进行“广播”发送。

3 结束语

该视频点播系统现已进入整机调试阶段, 考虑到利用 Ultra DMA 模式对硬盘进行访问时, 16KB 的双口 RAM 空间极大地限制了硬盘数据的读取效率, 因此, 在实际使用时采用大容量的 SDRAM 作为数据缓冲区, 为每个终端分配的存储空间达 512KB, 可供播放 2.89s。

该视频点播系统将芯片级嵌入式技术应用到消费电子领域, 极大地减小了 VoD 视频点播系统的体积, 使得该系统安装更方便、使用更灵活, 拓宽了该系统的使用范围。

(下转第 282 页)