

基于 Nios 的 IP 网络电话终端设计

杨玉峰, 黄炜

(电子科技大学 通信与信息工程学院, 四川 成都 610054)

摘要: 本文介绍了如何在 Altera 开发平台上应用 Quartus II 软件平台的 SOPC Builder 设计工具, 把 Nios 软核处理器、以太网芯片控制电路、其它外围器件的控制电路及用户逻辑电路都集成在了一片 FPGA 可编程逻辑芯片上。在这个系统硬件平台上运行 uc/OS 嵌入式操作系统、TCP/IP 网络通信协议和应用软件, 实现了基于 Nios 的 IP 网络电话功能。

关键词: FPGA IP 电话 TCP/IP 协议 可编程片上系统 Nios

Design of LAN IP Telephone Based on Nios

Yang yufeng, Huang wei

(College of Communication & Information Engineering, University of Electronic Science & Technology of China, Chengdu Sichuan 610054, China)

Abstract: This paper introduced how to integrate Nios soft MCU core, Ethernet MAC/PHY IC controller and other peripheral equipment's controllers in FPGA by using Quartus II and SOPC Builder. Based on the hardware platform, we can run uc/OS embedded system, TCP/IP network protocol and some applications. Thereby, we realized LAN IP Telephone based on Nios.

Key words: FPGA IP telephon TCP/IP protocol SOPC Nios

一. 系统概述

随着因特网技术的快速发展, IP 网络电话已经得到广泛使用。所谓 IP 网络电话是指利用因特网以分组数据包形式传输语音/传真等的新型电信业务, 简称 VoIP (Voice over IP: IP 网络承载语音业务)。它以低廉的价格, 灵活的应用使传统电信业务受到极大的挑战, 截止目前, 已经超过长途业务量的 50%。当然, 这包括各大电信运营商推广 IP 电话卡形式的 IP 电话长途业务, 其特点是接入部分使用传统电信终端 (如电话机、手机), 以特殊号码形式识别 (179XX), 在主干线上以分组数据包的形式在因特网上传输语音。目前专家普遍认为 VOIP 将代表电话未来的发展方向, 甚至有这样一句话: every thing is over IP (一切都基于 IP)。许多人相信 IP 电话时代即将到来, Frost&Sullivan 调研公司的报告中预计, 到 2007 年 VOIP 的通话量将占全部通话量的 75%, 甚至更高一些。因此, 在未来的几年内, IP 电话必将在电信市场占到举足轻重的地位, 对以太网电话终端的研究是符合市场需求并具有巨大的市场潜力和发展前景。

Nios 是 Altera 公司开发的一种采用流水线技术软核处理器, 专门针对可编程逻辑器件进行了优化, 因此是一种可配置的通用 RISC 微处理器, 可以与用户自定义逻辑结合。构成完整、功能强大的 SOC 系统, 在嵌入式系统设计中已成为趋势。应用 SOPC 技术把 Nios 配置进 FPGA 芯片后, 网络电话终端设备不再需要外部处理器, 用单一 FPGA 芯片就可以实现以前需要 FPGA + MCU 才能完成的功能。

嵌入式操作系统是嵌入式系统的灵魂, 可以显著提高软件操作效率并进行软件复用。针对当前日益复杂的系统开发, 应用操作系统是保证产品上市时间的关键。嵌入式 uc/OS 操作系统是一种性能优良、源码公开的免费操作系统, 具有高度灵活性。为此我们自行移植了 uc/OS 操作系统的 Nios CPU 版本, 继而在操作系统上完成了 IP 电话的设计。

二. 系统功能说明

基于 Nios 的 IP 电话分为接受方和呼叫方, 发起通话请求的一端称为呼叫方, 另一端为接受方。根据不同的应用场景, 呼叫方和接受方可以调换角色, 也就是说任意一方都可以发起通话请求。

呼叫方发起通话请求, 输入接受方 IP 地址, 两方通过网络连接成功后, 利用麦克风和听筒, 就可以进行谈话了。

本设计的实现分为硬件和软件两个部分。

硬件部分由带Nios 软核CPU 的FPGA芯片、外部数据存储Flash RAM 和SRAM、带AD 和DA 变换的PCM 语音编解码芯片电路、RS232 串行通信电路、以太网网络模块及RJ45接口、JTAG配置接口电路组成，此外还有电源电路等。

软件实现主要包含以下几个步骤：

1. 将嵌入式操作系统uc/OS移植到NIO5处理器上；
2. 开发语音采集编解码芯片的接口驱动程序；
3. 开发操作系统下的应用程序，实现IP电话功能。分别为呼叫方和接受方的应用程序，提供IP电话的电话呼叫和等待通话功能。

三. 性能参数

支持频率范围在300到3400HZ的单声道模拟语音信号通话；

系统将20ms的PCM语音数据直接打成语音数据包封装传输到因特网；

工作电压低，具有低功耗的特性。

四. 系统构成

系统主要以内嵌Nios软核处理器的FPGA为核心处理单元，辅以太网MAC/PHY控制器和PCM语音编解码芯片完成整个系统中的信号处理工作。其主要工作是完成模拟语音信号到PCM语音信号的编解码，以太网数据的接收和发送，协议处理，谈话的呼叫建立和释放等。

本系统的硬件结构框图如图4.1所示：

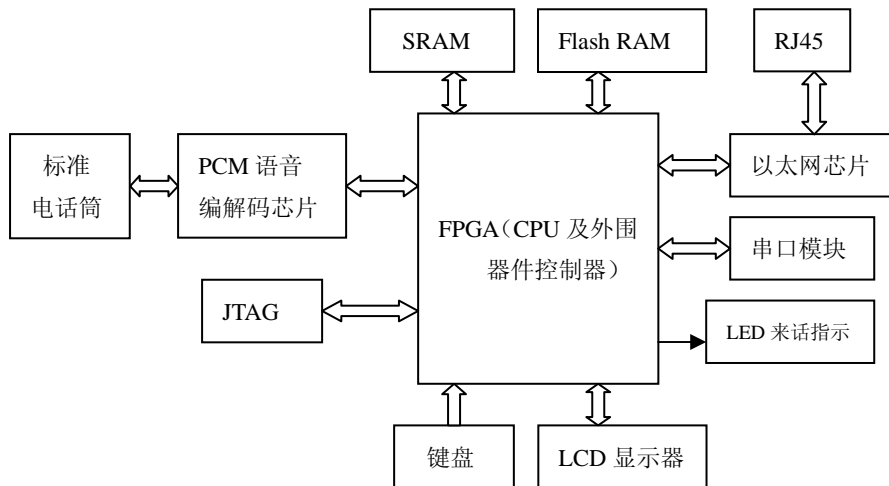


图 4.1 系统结构框图

系统软件体系结构如图4.2所示：

呼叫方应用程序	被叫方应用程序
SOPC Build自动生成驱动程序库	PCM语音编解码芯片驱动程序
uc/OS操作系统内核	
Nios CPU、PCM语音编解码芯片接口逻辑、网络模块、键盘显示模块等硬件电路	

图4.2 系统软件体系结构图

五. 设计描述

本系统的设计分为硬件和软件两个部分。硬件设计由于借助Altera的Stratix开发板，所以主要任务是扩展一块语音板，用来实现语音信号的数模变换和PCM语音编解码功能；软件部分移植uc/OS操作系统，编

写驱动程序和上层应用程序的开发。

1. 硬件部分

1.1 语音板电路设计

该部分采用OKI公司的A律PCM语音信号编/解码芯片MSM7702-3完成模拟语音到PCM信号的转换。该芯片支持频率范围在300到3400Hz的单声道模拟语音信号到PCM语音信号相互转换。该芯片具有如下特点：

1. 工作电压低(2.7-3.8V)，低功耗的特性。
2. 支持64/128/256/512/1024/2048 kHz等频率的串行数据输出。
3. 内置PLL电路，低电压运算放大器，具有可调的输出增益。
4. 芯片模拟信号输出引脚能直接驱动1.2k的负载。

这部分的原理图如图5.1所示，该部分的输入增益可调是通过调节图中的电阻R4和R3的值来实现的。其增益的具体运算公式如式(5.1)所示。该增益的最大的增益值不能超过10。

增益:Gain=1+R4/R3 式(5.1)

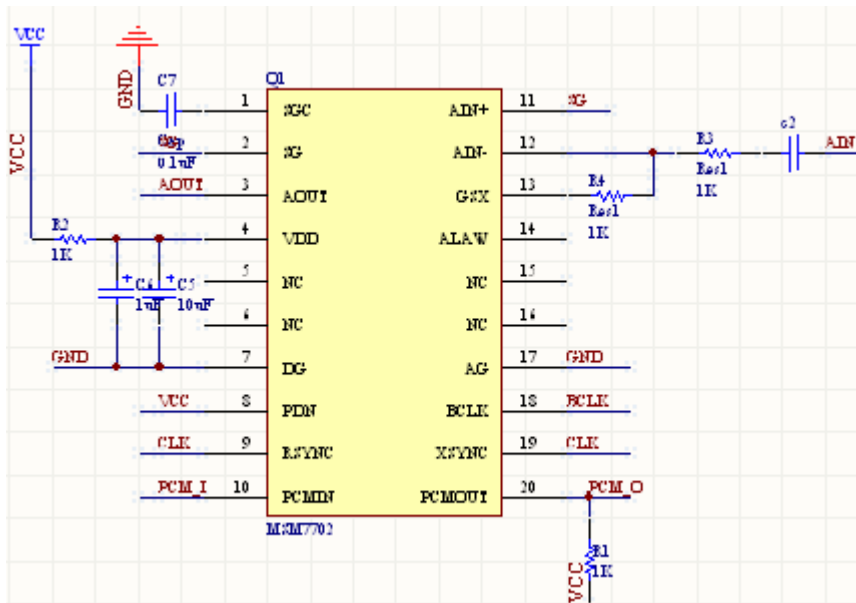


图5.1 MSM7702-3原理图

FPGA接收和发送PCM语音数据时，按照MSM7702-3的接口时序进行。MSM7702-3提供了发送PCM信号接口(BCLK, XSYNC和PCMOOUT引脚)和接收PCM信号接口(BCLK, RSYNC和PCMIN引脚)，其时序如图5.2和图5.3所示。

BCLK:接收和发送数据的时钟，频率为64KHz。

XSYNC:用于同步PCMOOUT引脚的数据输出的同步信号。

PCMOOUT:为输出的串行PCM语音数据引脚。

RSYNC:用于同步PCMIN引脚的数据输入的同步信号。

PCMIN:为输入的串行PCM语音数据引脚。

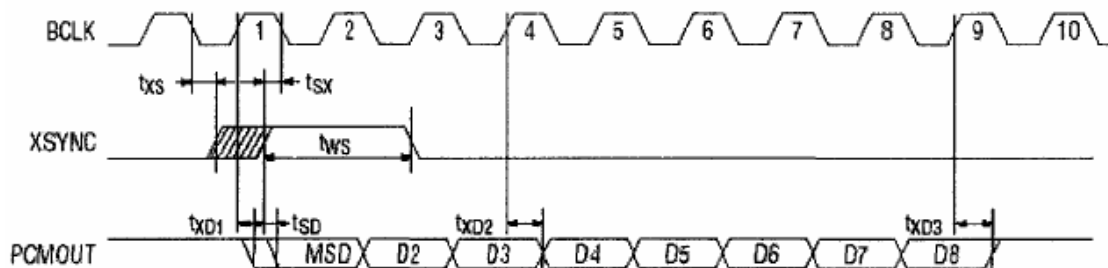


图5.2 发送PCM数据时序

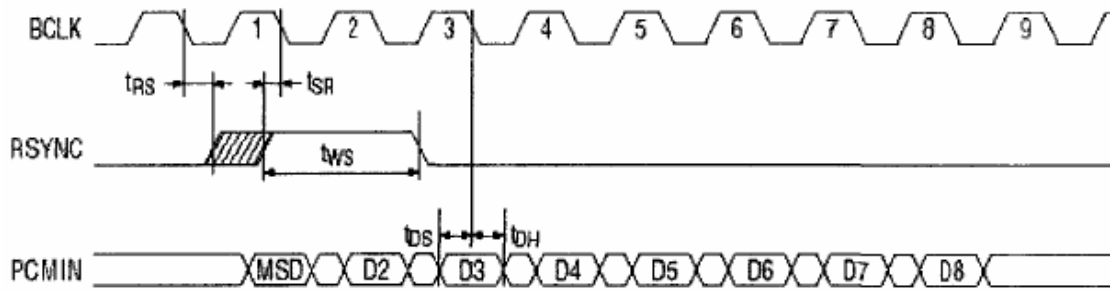


图5.3 接收PCM数据时序

根据MSM7702-3芯片数据收发时序，设计FPGA和MSM7702-3的接口电路结构如图5.4所示。

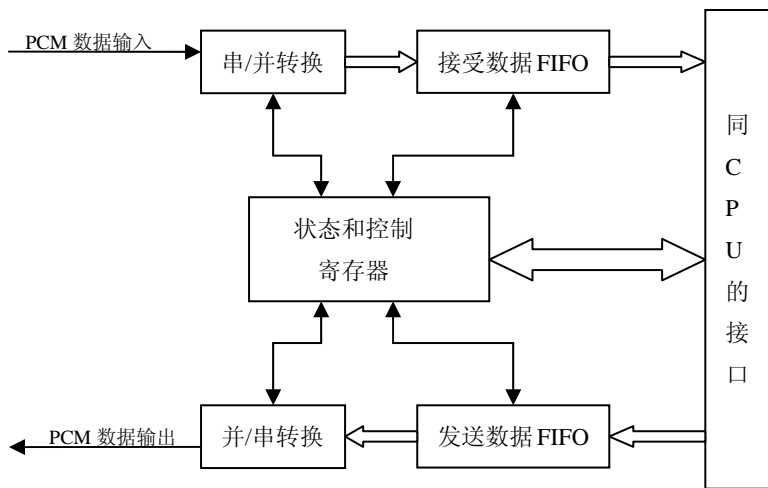


图5.4 MSM7702-3的接口电路框图

1.2 用SOPC Builder设计系统模块硬件

用SOPC Builder可以进行系统模块硬件设计和底层软件生成。进行硬件模块设计时，SOPC Builder提供图形化配置界面，备有一些常用外设的IP（Intellectual Property）模块，如SRAM、Flash RAM、UART、Ethernet Interface、Interval timer、Parallel I/O等。这些已开发好并已经引入到SOPC Builder环境中的功能模块被称为部件(Component)，打开SOPC Builder的图形界面时可以在左边模块池中看到这些功能部件。用户还可以用“Interface to User Logic”加入自己的外设设计文件，或直接加外设接口。可以采用的外设设计文件是用输入的电路原理图或HDL描述语言写的.vhd或.v文件。对于本设计来说，就是将在前面提到的MSM7702-3接口以用户自定义接口的方式添加进来，SOPC Builder可以进行系统配置以及生成，系统配置除了对外设设置外还包括启动程序、中断向量表、系统启动地址等的设置。具体如图5.5所示：

The screenshot shows the 'System Generation' tab in the Nios II IDE. It displays configuration options for the target board (Nios Development Board, Stratix EP1S10ES) and a clock frequency of 50.0 MHz. Below these settings is a table of hardware modules with their respective memory addresses and IRQs.

Use	Module Name	Description	Clock	Base	End	IRQ
<input checked="" type="checkbox"/>	cpu	Nios II Processor - Altera Corporation	clk	0x00050800	0x00050FFF	
<input checked="" type="checkbox"/>	ext_men_bus	Avalon Tri-State Bridge	clk			
<input checked="" type="checkbox"/>	boot_rom	Legacy On-Chip Memory (RAM or ROM)	clk	0x00051000	0x000517FF	
<input checked="" type="checkbox"/>	timer1	Interval timer	clk	0x00050020	0x0005003F	1
<input checked="" type="checkbox"/>	timer2	Interval timer	clk	0x00050040	0x0005005F	2
<input checked="" type="checkbox"/>	uart	UART (RS-232 serial port)	clk	0x00050060	0x0005007F	4
<input checked="" type="checkbox"/>	ext_flash_rom	Flash Memory (Common Flash Interface)		0x00000000	0x0003FFFF	
<input checked="" type="checkbox"/>	ext_sram	IDT71V416 SRAM		0x00100000	0x001FFFFF	
<input checked="" type="checkbox"/>	lan91c111	LAN91c111 Interface (Ethernet)		0x00040000	0x0004FFFF	3
<input checked="" type="checkbox"/>	button_pio	PIO (Parallel I/O)	clk	0x00050010	0x0005001F	0
<input checked="" type="checkbox"/>	lcd	Character LCD (16x2, Optrex 16207)	clk	0x00050080	0x0005008F	
<input checked="" type="checkbox"/>	led_pio	PIO (Parallel I/O)	clk	0x00050090	0x0005009F	
<input checked="" type="checkbox"/>	user_PCM	user_PCM		0x00050000	0x00050003	5

图5.5 系统硬件模块

2. 软件部分

软件设计包括嵌入式操作系统的移植、网络协议栈、驱动设计及应用级代码编写等部分，下面就对这几部分作简单介绍。

2.1 uc/OS 操作系统移植

uc/OS 操作系统是美国人Jean Labrosse 于1992 年开始编写的，它是一种适合于小型、微控制器的可剥夺实时操作系统。uc/OS 的内核除了没有网络协议栈之外，也没有文件系统，但是本设计中并不需要对话音数据按文件方式保存，因而使用uc/OS 可以满足需要。

uc/OS 在设计时就充分考虑到在不同平台上移植的需求，将同平台相关的部分局限在一个很小的范围内，对于不同的平台只需对下面一些函数和宏进行重写。

◆ OS_ENTER_CRITICAL 和 OS_EXIT_CRITICAL：这是两个宏，用来进行临界段保护。在本设计中使用汇编代码开关中断来实现。

◆ OS_TASK_SW：这是一个用于进行任务切换的宏。本设计中利用CPU 的软件中断方式实现。也就是说调用此宏产生软件中断，然后由相应的中断处理程序来具体实现任务上下文保护和任务切换。

◆ OSIntCtxSw：实现中断级任务切换，用纯汇编实现。

◆ OSCtxSw：实现用户级上下文切换，用纯汇编实现。

◆ OSTickISR：为系统定时器中断的处理函数，用纯汇编实现。

◆ OSTaskStkInit：用来在创建任务时，对任务堆栈进行初始化。

OS_CFG.H 用来配置内核，用户根据需要对内核进行定制，设置系统的基本情况；INCLUDES.H 为系统头文件，包括了整个实时系统所需要的内核以及用户的头文件。

2.2 网络协议栈

本系统采用的是SOPC Builder 中包含针对Nios 的网络协议栈，该栈以名为Plugs 的函数库的形式提供给设计者，该库向软件编写者隐藏了底层硬件细节，以类似于unix 套接字的方式实现了一个精简的网络协议栈。它支持以下一些网络协议：IP、ARP、ICMP、UDP、TCP。

使用该库在硬件上有四个要求：必须使用Nios CPU，需要20KB 的代码存储空间，需要4KB 的数据存储空间，系统中需要有一个专供其使用的定时器。

2.3 驱动及应用级代码编写设计

语音板驱动程序设计主要由以下几个部分组成：读缓冲区内容、写入缓冲区数据、产生中断信号等。另外应用级程序中的呼叫连接处理部分是系统中非常重要的部分，它主要完成通话前的呼叫建立过程及通话结束后的释放过程。下面对其进行详细描述，图5.5是呼叫连接处理部分的状态转换图。

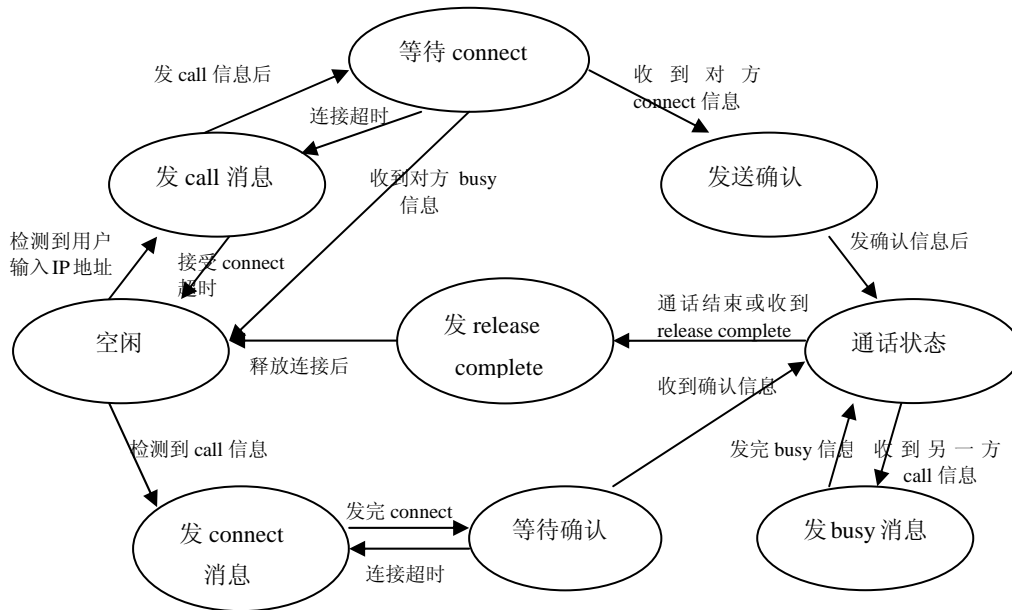


图 5.5 呼叫处理状态转换图

系统上电复位后，处于IDLE状态:如果检测到用户输入的IP地址，系统进入MASTER CALL状态;如果此时系统接收到一个CALL消息后，系统产生一个Informuse信号，使指示灯工作亮，通知用户有电话打进来，用户拿起话筒时，系统将置Informuse信号无效，指示灯熄灭停止工作，系统进入SLAVE CONNECT状态。

系统处于MASTER CALL状态:系统产生一个发送CALL消息使能信号xmitC.allMessageEn，控制发送部分发送一个CALL呼叫信号，当发送模块发送完该信号后，产生一个SendMessageOk信号，系统检测到该信号后，进入WAIT CONNECT状态。

系统处于WAIT CONNECT状态:系统等待接收CONNECT消息，同时设置一个定时器(最大值为4000)。定时接收CONNECT消息。该定时器是一个时钟频率8KHz计数器，当计数到最大值后，产生超时信号用于控制接收CONNECT消息是否超时。如果在规定时间内正确收到CONNECT消息，系统发送一确认消息ACK，然后进入TALK状态，开始通话。如果系统超时未接收到CONNECT消息，系统置重发呼叫CALL信号寄存器有效，返回MASTER CALL状态，重新发送呼叫请求CALL信号，同时将呼叫次数寄存器(calltimes)加1。一旦系统检测到呼叫次数寄存器(calltimes)值大于3，即系统连续3次发送呼叫请求，都没有完成，则本次呼叫连接过程失败，系统返回IDLE状态。如果系统没收到CONNECT消息，收到被呼叫方发来的BUSY消息(说明被呼叫方正在通话)，系统无法建立呼叫连接，系统返回IDLE状态。

系统处于TALK状态:系统内部的一个指示通话进行的状态寄存器(StartTalk)的值为1表示系统处于通话。系统检测用户是否挂机，一旦检测到用户挂机，表示通话结束，系统将通话状态寄存器((StartTalk)置为0，系统进入RELEAS COMPLETE状态。系统如果检测到呼叫方发来的release complete消息后，寄存器(StartTalk)置为0，同时系统进入IDLE状态。如果系统收到另一呼叫方所发的CALL消息，系统由于已经在和一个用户在通话，不能接收另一个用户的呼叫请求，系统进入BUSY状态。

系统处于RELEASE COMPLETE状态:系统发送一个release complete消息，该消息发送完毕后，系统进入IDLE状态。

系统处于SLAVE CONNECT状态:系统发送一个CONNECT消息，该消息发送完毕后，系统进入WAIT ACK状态。

系统处于WAIT ACK状态:系统等待接收ACK消息，同时启动定时器工作(最大值为4000)，当计数到最大值后，产生超时信号用于控制接收ACK消息是否超时。如果系统超时未收到ACK消息，系统则重发呼叫CALL信号寄存器有效，返回SLAVE CONNECT状态，重新发送CONNECT信号，同时将呼叫次数寄存器(calltimes)加1。一旦系统检测到呼叫次数寄存器(calltimes)值大于3，即系统连续3次发送呼叫连接请求，都没有收到确认ACK消息，则本次呼叫连接过程失败，系统返回IDLE状态。如果系统正确收到确认ACK消息，系统进

入TALK状态，可以开始通话。

系统处于BUSY状态:该BUSY状态是指，当前系统正在通话阶段，同时接收到另一呼叫方发来的呼叫请求CALL消息时所处的状态。系统在此状态时，发送一个BUSY消息给该呼叫方，拒绝本次呼叫请求。完毕后，系统返回TALK状态。

2.4 数据的协议处理

需要说明的是，在本系统中呼叫信息和语音信息采用不同的协议处理方式。呼叫消息用TCP报文封装传送，以确保呼叫信息传输的可靠性；语音消息用UDP报文封装传送。呼叫前后的协议处理流程如图5.6所示。

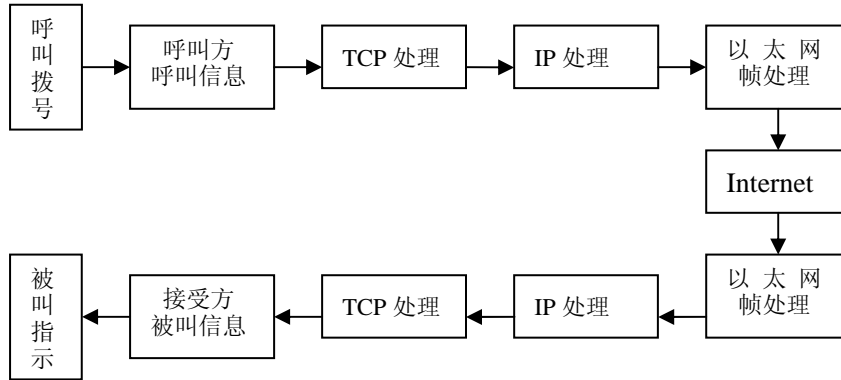


图 5.6 呼叫信息协议处理流程

通话状态时语音数据处理流程如图5.7所示

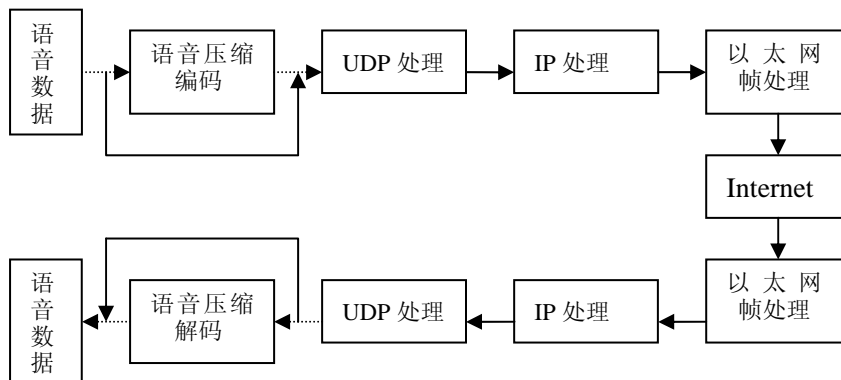


图 5.7 语音数据协议处理流程

在本设计中没有采用语音数据包没有进行压缩后传输，而是直接对G. 711标准的PCM语音信号直接打包，故图中语音压缩编/解码未用到。

六. 系统特点

1. 在目前数字系统开发中，传统的MCU+FPGA的方式成本较高。本系统采用嵌入Nios软核的FPGA芯片，并通过Avalon总线接口，将系统最大限度地集成在一块芯片中实现，提高了稳定性，同时简化了系统软硬件设计。
2. uc/OS 操作系统的移植。uc/OS 操作系统源代码公开，便于自行开发各种应用软件程序，因此，进行Nios CPU的移植具有极其重要的意义和价值。
3. 编写操作系统下的语音板的驱动程序。在嵌入式系统设计中，常需要根据数据或处理任务进行外设定制，从而提高整个系统性能，这也是SOPC Build的长处所在。只有为这些自定义的外设进行驱动程序开发，才能完成操作系统下的应用程序开发。
4. 完成uc/OS 下的应用程序。分别为呼叫方和接收方应用程序提供IP电话的呼叫和等待通话功能。
5. 由于SOPC Builder 开发环境的完备功能，可以把注意力集中在系统整体构架和功能上来，而无需用过多的考虑细节性的电路设计，缩短了产品面市周期；同时由于FPGA和Nios的灵活性，为系统

的功能升级带来方便。

七. 总结

基于Nios 进行嵌入式IP电话系统设计有两个突出优点。首先，硬件设计快速灵活，可扩展性强。SOPC Builder 降低了硬件设计的难度，缩短了硬件开发周期，提高了设计的可靠性。CPU 本身是以软核的方式实现，其功能可根据需要进行定制。其次，软件开发容易。SOPC Builder 不但在硬件设计上提供了支持，在软件上也为设计者提供了较好的支持，方便驱动程序和上层应用程序的编写，并提供了现成的网络协议栈。

由于本系统中是直接对PCM语音信号打包发送，因而要求的带宽较大。系统的功能也相对简单，下一步的工作可以将语音数据压缩编/解码引入到FPGA中，减小系统要求的带宽。同时还可以将基于H. 263实时图象传输系统引入FPGA中，构建局域网IP可视电话。

参考文献

- [1] William E. Witowsky. IP Telephone Design and Implementation Issue, 1998. 7.
- [2] Nios II Software Developer's Handbook (ver 1.2, Dec 2004)
- [3] 潘松、黄继业等著. SOPC技术实用教程. 清华大学出版社 2004. 6
- [4] 谢兵森. 基于嵌入式系统的指挥调度终端试验研究. 上海船舶运输科学研究所硕士论文, 2004.
- [5] Jean J. Labrosse 著, 邵贝贝译. μ C/ OS II—源码公开的实时嵌入式操作系统. 中国电力出版社, 2003.
- [6] 夏宇闻著. Verilog数字系统设计教程. 北京航空航天大学出版社. 2004. 7

原创性声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果。

杨玉峰

2006. 03. 18

作者简介: 杨玉峰 (1983—)，男，江苏盐城人，电子科技大学在读硕士研究生，主要研究方向为嵌入式系统开发应用；黄炜 (1952—)，男，湖南宁乡人，电子科技大学副教授，硕士生导师，主要研究方向为现代通信中的信号处理、音频及视频数据通信、嵌入式系统开发应用。

作者联系方式:

杨玉峰: 地址: 电子科技大学通信与信息工程学院 2004201050 班
邮编: 610054

E-mail: happyangyufeng@163.com

电话: (028) 83200335 (028) 83208880