基于 SoC 和嵌入式 Linux 的数据采集系统设计

叶 琴,谢捷如
(南京航空航天大学 自动化学院,江苏 南京 210016)

摘 要:随着信息处理技术的不断发展,信息容量、处理速度和数据精度的要求越来越高。文中主要针对典型发动机管理系统参数,采用软硬件协同设计方法,实现了一套功能完备的数据采集系统。航空发动机气路及机械性能参数的采集工作已然成为飞行器状态监控与故障诊断至关重要的环节之一。为了实时了解发动机的运行状态,准确掌握运行参数的变化规律并实现一定程度的人机交互,系统硬件部分采用 SoC 作为核心,软件部分基于 FPGA、嵌入式 Linux 以及 CGI 相关技术,使整个系统更具实用性、人性化。系统的开发主要借助于 Xilinx ISE 开发套件以及 Linux 交叉编译环境,设计了相关 FIR 数字滤波模块并编写模拟 ADC 驱动程序,最终通过 CGI 接口实现数据采集系统与上位机之间的通信,为状态监控与故障诊断系统设计提供了思路。

关键词:SoC;CGI;嵌入式 Linux;数据采集系统;软硬件协同设计
中图分类号:TP302.1 **文献标识码**:A **文章编号**:1673-629X(2015)08-0203-05
doi:10.3969/j.issn.1673-629X.2015.08.043

Design of Data Acquisition System Based on SoC and Embedded Linux

YE Shen, XIE Jie-ru
(College of Automation, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract: With the continuous development of information processing technology, the requirement of information capacity, processing speed and data accuracy has become higher and higher. In connection with the typical parameter of engine management system analyzing, hardware/software collaborative design methodology is used to realize a fully functioning data acquisition system. Collecting performance parameters of aeroengine gas circuit and mechanical part has been a vital segment of aerocraft condition monitoring and fault diagnosis. In order to know the real-time running state of the engine, to grasp the changing law of operating parameters accurately and to achieve a certain degree of human-computer interaction, SoC is used as the core of hardware. With the help of embedded Linux and CGI technology, the whole system has become more practical and more user-friendly. The system development mainly depends on the Xilinx ISE development kit and the Linux cross compiler environment and the relevant FIR digital filter module and the preparation of simulated ADC driver are designed. By achieving the communication between data acquisition system and host computer through CGI interface, the system provides a way for condition monitoring and fault diagnosis system design.

Key words: SoC; CGI; embedded Linux; data acquisition system; hardware/software collaborative design

0 引 言

航空发动机故障往往是导致飞行事故的主要原因,航空发动机性能的稳定便成为了飞行器安全的重要保证。考虑到其高昂的设计制造和故障维护成本,对发动机关键部件实行实时状态监控与故障诊断能够迅速并准确地确定故障的具体部位及严重程度,从而减少维护费用并确保飞行器安全。针对典型发动机监控系统参数的数据采集系统便是实现该策略不可或缺的一部分。作为新一代嵌入式处理器的代表,SoC 拥有着更小的体积、更低的功耗以及更高的性价比,其功能也更加丰富,应用范围更广。

采用 SoC 作为硬件部分的核心不仅可以解决产品性能与体积之间的矛盾,而且可以使信号处理更加高速化与功能化。相对于其他种类的嵌入式操作系统,Linux 的移植更加容易,性能也更为优异。作为 GNU 计划的开源项目,其良好的生态环境也使得产品

收稿日期:2014-09-15 **修回日期**:2014-12-16 **网络出版时间**:2015-07-21
基金项目:国家自然科学基金资助项目(61275199)
作者简介:叶 琴(1989-),男,硕士,研究方向为嵌入式系统;谢捷如,副教授,研究方向为电工理论与新技术。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20150721.1433.020.html

的开发周期大为缩短,稳定性和安全性得到提高。

1 Zynq 简介

Zynq 是赛灵思公司(Xilinx)推出的全可编程片上系统解决方案,该系统基于可扩展处理平台(EPP)结构,将双精度浮点引擎的 ARM Cortex-A9 双核处理器与低功耗可编程逻辑紧密结合,提供了强大的灵活性、可配置性和系统性能[1]。

Zynq 主要由 PS(Processing System)和 PL(Programmable Logic)两大功能模块构成。其中,PS 部分主要包括应用处理单元(APU)、存储器接口、I/O 外设以及内部互联;PL 部分则采用了 FPGA 技术,用于扩展功能,以满足特定的功能要求[2]。

2 总体设计

数据采集系统大多是借助传感器将被测非电量参数转换为连续的模拟信号,经由一系列采样量化手段转换为离散的数字信号,最终通过存储传输等方式实现与用户之间的交互[3]。

一个完整的嵌入式系统通常需要由硬件和软件共同组成,本系统主要从硬件设计与软件设计两部分着手,逐步完成数据采集系统的设计工作。其中,硬件设计部分主要包括参数计算、芯片选型以及 PCB 绘制;软件设计部分主要包括 FPGA 程序设计、操作系统配置、驱动程序设计以及 CGI 程序设计[4]。

3 硬件设计

硬件设计的正确性与稳定性是确保整个系统正常运行的基础,根据实际性能指标、系统需求及生产成本等要素进行总体规划,得出已有条件下的最佳方案是硬件设计的最终目标[5]。

本系统的主要任务是采集航空发动机气路及机械部分的相关运行参数,在线实时数字滤波并以网页形式反馈当前航空发动机的运行状态。除了嵌入式处理器,通信模块与存储模块也是本系统必不可少的组成部分,具体系统硬件结构如图 1 所示。

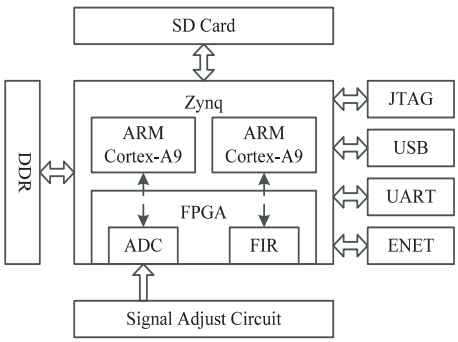

图 1 系统硬件结构图

3.1 参数计算

信号调理电路主要由幅值调理、抗混叠滤波以及差分驱动三大部分组成,这三部分模拟电路的相应参数均需经过准确计算获得。

幅值调理电路分为运算放大电路与电压跟随电路两部分,根据发动机气路与机械部分传感器的输出端幅值以及差分驱动电路输入端幅值要求设计电路参数。在信号传输过程中,不可避免地会引入一些干扰或噪声,为了提高数据采集的精度,本系统采用五阶巴特沃斯滤波器对干扰信号进行滤除。鉴于温度对输入信号的影响,ADC 前端需加入相应的差分放大驱动电路。差分驱动电路参数主要由幅值调理电路输出端幅值以及 ADC 模块输入端幅值决定,通过调整线性工作状态下的共模参考电压即可改变输入信号的幅值范围。

3.2 芯片选型

嵌入式处理器的选择往往决定着系统的性能以及功能的实现。本系统选用 Zynq 作为核心处理单元,其异构多核架构使得其具有更高的能量效率与并行效益。通过软硬件协同设计可最大程度地发挥该处理器的优势,使其具有更多维的优化空间以及更好的灵活性。

通信模块的功能是实现终端系统与上位机之间的互联,其主要由 USB 接口、网络接口以及串行接口等子模块组成。通用串行总线是一种用于规范系统与外设之间连接的总线标准,本系统选用 TI 公司的 TUSB1210 独立物理层收发器作为 USB 模块的控制核心,该芯片支持 OTG(On-The-Go)补充标准可实现便携设备间的数据交换。网络接口通常特指以太网接口,是当今应用最为普遍的局域网技术,本系统选用 MARVELL 公司的 88E3018 单端口快速以太网收发器作为以太网接口的控制核心,其遵循千兆以太网协议标准并支持 JTAG 在线调试,配以 RJ-45 接口可实现系统间的网络访问。串口通信是一种可将并行数据按位进行传输的通信方式,其传输速度较低但可以节约远距离通信的成本,本系统选用 MAXIM 公司的 MAX232 多通道 RS-232 驱动芯片,将 TTL 电平转换为 232 电平以用于硬件平台的调试工作。

存储模块是运行嵌入式 Linux 系统必不可少的硬件基础,操作系统数据的存取通常需要借助于动态随机存储器以提高访问速度,本系统选用两片 MICRON 公司的 MT14J128M DDR3 SDRAM 芯片作为动态存储单元。鉴于 SDRAM 容量大小有限并且其数据掉电易挥发,本系统还选用了 TI 公司的 TXS02612 SDIO 多路传输扩展芯片作为 SD 卡的驱动核心。嵌入式 Linux 内核、文件系统以及采集到的数据均需存放在 SD 卡

中,以确保系统软件部分的正常运行。

4　软件设计

相对于硬件设计,软件设计则是完善系统功能、实现人机交互的重要环节。本系统通过操作系统配置与驱动设计实现对 FPGA 模拟所得 ADC 模块与 FIR 模块及其他外设的底层访问,并借助 CGI 程序完成人机交互界面的顶层设计。

4.1　FPGA 程序设计

传统的 FPGA 开发通常需要借助 HDL(Hardware Description Language),完成电路设计、输入设计、功能仿真、综合优化、综合仿真、时序仿真以及板级调试等一系列步骤。Xilinx 公司提出的高级综合 HLS(High Level Synthesis)工具使得传统的硬件级设计也可以采用软件级的设计模式,极大地提高了 FPGA 的设计效率,降低了 FPGA 的设计成本。

基于 HLS 的 FIR 数字滤波器设计流程为:首先由 Matlab 仿真计算出所有满足设计指标的脉冲响应常数,然后根据 FIR 数字滤波器设计方法编写 C 源码,最终通过 HLS 工具创建 RTL(Register Transfer Level)提取控制和数据流,从而完成 IP 核设计。

Matlab 仿真设计 FIR 数字滤波器的途径主要有三种:程序设计法、FDATool 设计法和 SPTool 设计法。其中,FDATool 设计法需要在 Simulink 组件中建立 FIR 动态系统模型并配置相关滤波器参数,如采样频率 Fs、截止频率 Fc、最小阻带衰减 As 等。FIR 数字滤波器设计方法是以直接逼近所需离散系统的频率响应为基础,包括窗函数法、最小二乘法、等纹波法和频率抽样法等。本系统采用窗函数法,尝试不同滤波器阶数和窗函数直至获得理想的幅频特性以及相频特性。将仿真所得 FIR 数字滤波器的脉冲响应常数输出生成 dat 数据文件,由于单位冲激响应序列与其差分方程的对应系数相等,只要确定了 FIR 数字滤波器单位冲激响应序列,就能够得到相应的输入输出关系[6]。

$N-1$ 阶 FIR 数字滤波器单位冲激响应与传递函数之间的频率响应函数关系为:

$$H(\mathrm{e}^{\mathrm{j}\omega}) = H(z)\mid_{z=\mathrm{e}^{\mathrm{j}\omega}} = \sum_{n=0}^{N-1} h(n)\mathrm{e}^{-\mathrm{j}\omega n}$$

根据给定的理想频率响应,利用傅里叶反变换,求出单位冲激响应,再乘以窗函数,即可求得所要求的 FIR 数字滤波器的滤波器系数:

$$h(n) = w(n)h_d(n) = \frac{w(n)}{2\pi}\int_{-\pi}^{\pi} H_d(\mathrm{e}^{\mathrm{j}\omega})\mathrm{e}^{\mathrm{j}n\omega}\mathrm{d}\omega$$

不同的窗函数性能也各不相同,窗函数的选择往往决定着滤波器是否能够满足设计指标。对于阻带最小衰减大于 60 dB 的 FIR 数字滤波器,通常选用布莱克曼(Blackman)窗或凯瑟(Kaiser)窗作为窗函数。凯瑟窗函数表达式如下:

$$w(n) = \frac{I_0(\beta\sqrt{1-(1-2n/(N-1))^2})}{I_0(\beta)}$$

其中,I_0 为第一类零阶修正贝塞尔函数;β 为控制参数。

因已知单位冲激响应 $h_d(n)$ 和凯瑟窗函数表达式,只需将滤波器系数求解过程编写成 C 源码并添加到 HLS 工程中进行相应的模型仿真和验证,经过综合优化和时序仿真即可生成相应 FIR 数字滤波器的 IP 核。时序仿真结果如图 2 所示。

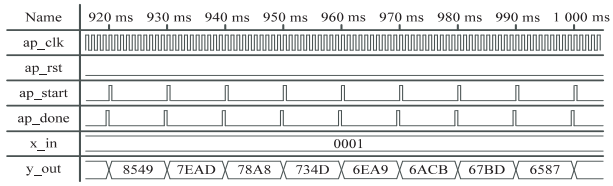

图 2　Modelsim 协同仿真时序图

在建立系统工程时,需将定制的 FIR 数字滤波器模块复制到 XPS(Xilinx Platform Studio)项目对应的 pcores 目录下,例化连接 IP 核并添加用户约束,最后导入 SDK(Xilinx Software Development Kit)进行软硬件协同设计。

4.2　操作系统配置

嵌入式 Linux 是将 Linux 内核进行剪裁和修改,使之能够运行在嵌入式系统上的一种小型操作系统。相较于其他种类的嵌入式操作系统,嵌入式 Linux 凭借其灵活的配置、小巧的体积以及丰富的功能,广泛应用于消费电子、医疗设备、工业控制、航空航天等领域[7]。

嵌入式 Linux 的开发环境较为特殊,驱动程序以及应用程序的开发往往需要在特定的交叉编译环境下进行。考虑到大多数调试软件都必须运行在 Windows 环境下,为了避免独立系统之间的反复切换,开发者通常会在 Windows 环境下安装 Linux 虚拟机,本系统选用 VirtualBox 虚拟机软件以及 Ubuntu 操作系统作为开发平台。除了配置虚拟的 Linux 系统环境,还需为主机端安装与目标系统相对应的编译软件。通常,桌面级 Linux 都会自带 gcc 编译套件,然而嵌入式系统的处理器架构与桌面级系统之间往往存在着差异[8-9]。在以 Zynq 为核心的硬件系统中,Bootloader、Linux 内核、驱动程序以及应用程序的编译工作都需借助 arm-xilinx-linux-gnueabi-gcc 完成。该套件可将程序源码进行预处理、编译、汇编和链接,最终生成能够运行在 Zynq 体系结构中的二进制可执行文件。

由图 3 可以看出,经过硬件引导固件以及第一阶段引导程序的引导,系统初步完成硬件初始化并将跳转执行第二阶段引导程序。其中,第一阶段的比特流

文件可由 XPS 配置 PL 获得,ARM 初始化程序可由 SDK 例化生成。在第二阶段中,系统需要分别加载 Bootloader、Linux 内核、根文件系统以及设备树文件,以完成嵌入式 Linux 的正常启动。因此,对这些部件的编译和移植就成为了嵌入式 Linux 系统配置不可或缺的重要环节。

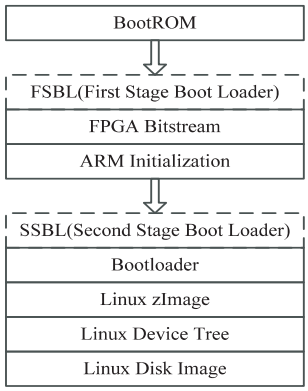

图 3 Zynq 启动流程图

Bootloader 是操作系统运行前的一段引导程序,它的主要功能包括了系统参数配置、硬件模块初始化、功能在线更新、操作系统装载等。本系统选用 U-Boot 作为第二阶段的启动装载程序,U-Boot 是众多 Bootloader 中的一种,对目前主流处理器和嵌入式操作系统都有着稳定的支持。为了使设备树文件的相关信息能够顺利地传递给 Linux 内核,首先需要对相关头文件作相应配置。通过 Git 下载获得 u-boot-xlnx 源码并 make zynq_zed_config,完成配置后经过交叉编译即可生成相应的 u-boot. elf 可执行文件。该文件与比特流文件、ARM 初始化程序一起可由 SDK 整合生成相应的 BOOT. BIN 文件,用于引导嵌入式 Linux 系统[10]。

Linux 内核的主要任务是对计算机硬件进行编程控制和调度访问,并为上层用户程序提供相应的虚拟接口和执行环境。其主要由进程调度、内存管理、文件系统、进程通信、网络接口五大模块构成[11]。与 U-Boot 相同,在对 Linux 内核进行交叉编译之前,首先需要对其进行可视化配置。通过修改相应的 Kconfig 以及 Makefile 文件,可以将定制的驱动程序加载到 Linux 内核中,以实现对底层硬件的控制和访问。通过 Git 下载获得 linux-3. 3-digilent 并 make zImage ARCH = arm CROSS_COMPILE = arm-xilinx-linux-gnueabi-即可生成 zImage 镜像文件。

为了实现用户与操作系统之间的交互,Linux 在启动时需要从存储设备上挂载相应的根文件系统。根文件系统的种类繁多,分别适用于不同的存储设备,本系统搭载了 256 M DDR,因此选用 Ramdisk 作为根文件系统。Ramdisk 基于虚拟磁盘技术,极大地提高了文件的访问速度,然而其固有的数据易失性使得开发者不得不事先将压缩镜像存放在诸如 SD 卡的非易失性存储介质中。Ramdisk 的制作需要相当的准备工作,首先在主机端 Ubuntu 系统中配置安装 BusyBox 生成相应的根文件系统目录,编译安装 Dropbear 并在该目录中建立链接使系统支持 SSH 功能,然后创建 Ramdisk 镜像并挂载根文件系统目录,最后卸载目录并压缩镜像生成 Ramdisk. image. gz 文件[12]。

设备树源自 OpenFirmware,是一种硬件描述数据结构,由一系列节点和属性组成。引入设备树的目的在于减少 Linux 内核板级硬件编码的冗余,使板级细节更具可配置性。设备树的编译需要借助于 DTC (Device Tree Compiler),DTC 可由相应的 Linux 内核编译生成。通过指令将设备树源文件(. dts)编译生成扁平设备树文件(. dtb),经由 Bootloader 将硬件信息传递给 Linux 内核。

DTC 具体指令如下:

```
dtc [-I dts] [-O dtb] [-o output] [-V version] input
```

设备树源文件部分代码如下:

```
xadc@ f8007100 {
compatible = "xlnx,ps7-xadc-1.00.a";
reg = <0xf8007100 0x20>;
interrupts = <0 7 0>;
interrupt-parent = <&gic>;
};
```

4.3 驱动程序设计

设备驱动是建立硬件与软件之间互联的一种接口程序,Linux 系统将存储器和外设大致分为三类,分别为字符设备、块设备和网络设备。Zynq 的 PL 部分集成了一个双 12 位最高分辨率、1MSPS 最大采样率的 ADC(Analog-to-Digital Convertor)模块。与该模块交互的方式主要有三种:通过 PS 部分的专用串行接口直接访问;作为 AXI 总线外设供 PS 部分或 MicroBlaze 虚拟内核访问;作为 IP 核虚拟成独立的逻辑器件单独访问。本系统主要采用了第一种访问方式,其串行接口结构如图 4 所示。

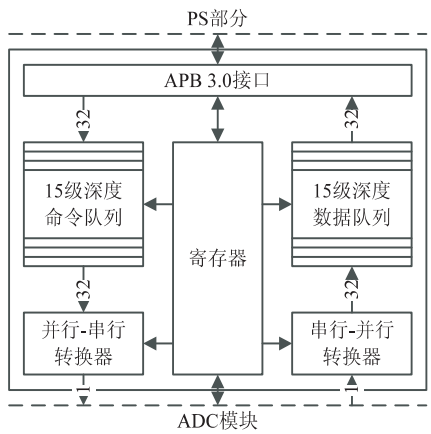

图 4 ADC 模块串行访问接口结构图

ADC 模块本身属于字符设备,但其并非如其他外设一样通过特定的总线协议实现与 SoC 之间的交互,而是挂接在特定的内存空间中[13]。

Linux 系统中的 platform 虚拟总线机制可以轻松地实现此类设备的驱动设计,platform_device 并非与字符设备、块设备和网络设备并列的概念,而是 Linux 系统提供的一种附加手段,与之对应的 platform_driver 结构便是实现具体硬件操作的设计对象。该机制的优势在于可将设备资源交由内核统一管理,并在使用时通过标准接口进行访问,提高了驱动与资源管理的独立性。

除了使用 platform 虚拟总线作为驱动管理手段,本系统还借助 sysfs 文件系统将驱动数据由内核导出至用户空间。sysfs 文件系统的提出解决了设备驱动模型不统一的局面,并为驱动程序提供了丰富的辅助性函数,大大提高了开发效率。基于 sysfs 开发的驱动程序可为上层应用程序提供独立的设备属性接口,使用户可以像访问普通文件一样对设备进行操作。

ADC 模块驱动工作流程如图 5 所示。

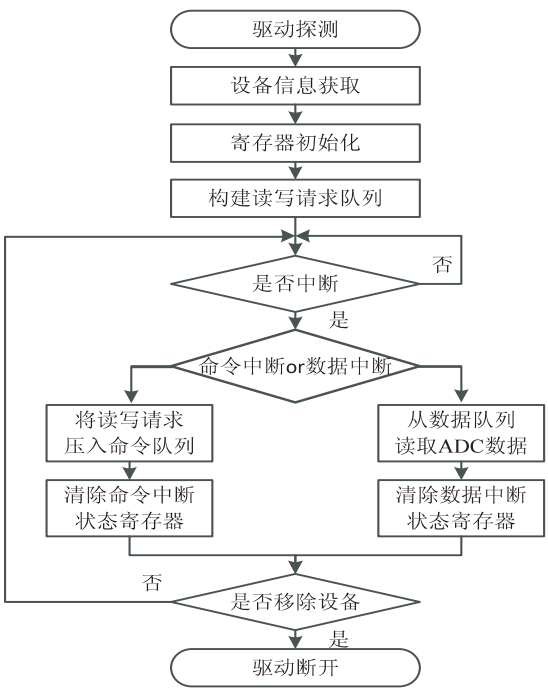

图 5　ADC 驱动程序工作流程图

其具体设计步骤如下:

(1)根据访问接口结构构建 ADC 结构体;

(2)基于 platform 虚拟总线设计设备初始化和析构函数,并关联相应设备树资源;

(3)构建请求队列并设计对应压入函数;

(4)设计中断函数完成队列读写操作;

(5)根据 sysfs 文件系统结构设计设备属性读写函数,创建属性组并注册相应设备属性;

(6)利用 gdb 完成调试并加入 Linux 内核。

为了实现 Linux 系统对 FIR 数字滤波器的访问,同样需要为其开发相应的驱动程序。FIR 模块驱动的实现较为简单,同样基于 platform 虚拟总线与 sysfs 文件系统。与 ADC 模块不同的是,对 FIR 模块的操作并不需要访问相关寄存器和队列,只需将 FPGA 程序设计部分得到的 IP 核接口物理地址映射到 Linux 内核相应内存空间,并基于系统内联信号即可实现实时 FIR 数字滤波。FIR 模块驱动具体设计步骤与 ADC 模块近似,在此不予赘述。

4.4　CGI 程序设计

CGI(Common Gateway Interface)是 Web 服务器与外部应用程序之间的一种接口标准,该标准允许 Web 服务器收集 Web 页面中的表单信息,执行相应的外部应用程序并将输出结果反馈给 Web 浏览器[14]。

CGI 程序可由多种语言实现,考虑到 Linux 系统环境下多进程任务的优势以及嵌入式系统对于运行速度的要求,本系统采用 C 语言作为 CGI 程序的主要实现方式。CGI 程序设计的主要任务不仅限于 HTML 表单的制作,数据处理程序的开发以及图形界面的应用也是其重要的组成部分。

CGI 程序动态页面的实现很大程度上依赖于 SSI(Server Side Include)命令的使用,这虽然加重了 Web 服务器的负担,但能有效地减少客户端的代码量。SSI 命令不仅能够执行特定的系统指令或应用程序,还可以实现对 ODBC(Open DataBase Connect)数据库的查询与访问。

本系统 CGI 程序设计主要基于前期开发的 ADC 与 FIR 模块驱动,通过 GET 和 POST 表单传送方式执行相应的数据处理程序,并借助 SQLite 数据库实现数据的存储与读取。其具体设计步骤如下:

(1)设计 ADC 模块工作模式与采样频率配置函数;

(2)为 FIR 模块创建子进程,设计定时中断读取 ADC 数据存入数据库并向 FIR 子进程发送信号;

(3)设计 FIR 子进程信号中断,从数据库读取 ADC 数据并写入 FIR 模块,将滤波后的数据写入数据库并关联 ADC 数据;

(4)制作 HTML 表单,根据数据库相关数据生成动态图形界面。

5　结束语

文中从实际项目出发,介绍了嵌入式数据采集系统的具体设计过程,为其他嵌入式设计方案提供了基本思路。同时,文中主要对软硬件协同设计和嵌入式 Linux 下的系统配置、驱动开发以及 CGI 设计进行了深

(下转第 212 页)

GNS3 软件,运行真实的协议栈,在逻辑上模拟真实路由器,以提高其真实性,由 tc 和 NetEm 配合使用控制链路性能。通过实验验证了 INPTX 对链路、网络节点的控制能力,通过分析 Reno 和 Vegas 算法的公平性实验结果,与 NS2 软件仿真对比,证实所设计的准实验床具有高逼真度。

参考文献:

[1] Siaterlis C, Masera M. A survey of software tools for the creation of networked testbeds[J]. International Journal on Advances in Security, 2010, 3(1): 1-12.

[2] White B, Lepreau J, Stoller L, et al. An integrated experimental environment for distributed systems and networks[C]// Proceedings of the 5th symposium on operating systems design and implementation. [s. l.]: [s. n.], 2002: 255-270.

[3] 秦董洪,陈智勇,杨家海. 基于 Emulab 的网络仿真实验平台研究[J]. 实验室科学,2013,16(3):92-95.

[4] Mahrenholz D, Purushothaman I. The network simulator ns-2 [EB/OL]. (2011-11-04)[2014-04-15]. http://www.isi.edu/nsnam/ns.

[5] 马元飞. NS2 条件网络性能分析实践[J]. 电脑与电信,2013(1):39-41.

[6] Zhang Enming. Xen project, a linux foundation collaborative project[EB/OL]. (2014-05-20)[2014-06-24]. http://wiki.xensource.com/wiki.

[7] Grossmann J, Saraiva F J. GNS3, graphical network simulator [EB/OL]. (2014-03-25)[2014-07-02]. http://www.gns3.net.

[8] Hemminger S. Network emulation with netem[C]//Proceedings of the 6th Australia's national Linux conference. [s. l.]: [s. n.], 2005.

[9] 何秀淼. 虚拟化技术基本原理及应用[J]. 电信网技术,2014(6):16-18.

[10] Calarco G, Casoni M. On the effectiveness of Linux containers for network virtualization[J]. Simulation Modelling Practice and Theory, 2013, 31: 169-185.

[11] Citrix CloudPlatform. XCP overview[EB/OL]. (2013-01-20)[2014-01-27]. http://wiki.xensource.com/wiki/XCP_Overview.

[12] Hubert B, Graf T, Maxwell G, et al. Linux advanced routing & traffic control[EB/OL]. (2012-05-20)[2013-01-20]. http://lartc.org.

[13] 秦 军,袁翰林,陈 迪. 异构无线网络中 TCP Vegas 算法的研究与改进[J]. 计算机技术与发展,2012,22(4):88-92.

[14] 牛 磊,王 峰,刘冬冬,等. TCP Reno 拥塞控制的改进算法[J]. 福建电脑,2014,30(2):7-9.

[15] Feng W, Vanichpum S. Enabling compatibility between TCP Reno and TCP Vegas[C]//Proceedings of IEEE symposium on applications and the Internet. [s. l.]: IEEE, 2003: 301-308.

[16] 肖 锴,章国安,杨 云. 基于 NS2 的 TCP Vegas 算法的仿真与研究[J]. 实验技术与管理,2012,29(8):92-95.

[17] 刘国柱,高文娟. 基于 TCP Reno 和 TCP Vegas 拥塞控制性能研究[J]. 计算机工程与设计,2011,32(2):434-437.

(上接第 207 页)

入研究,基于设计所得硬件搭建实验平台,对数据采集功能作了反复测试。文中给出的方案能够有效减小数据采集系统的体积并极大地扩展其系统功能,对于飞行器状态监控与故障诊断系统的进一步设计具有一定的参考价值。

参考文献:

[1] 何 宾. Xilinx All Programmable Zynq-7000 SoC 设计指南[M]. 北京:清华大学出版社,2013.

[2] 陆佳华,江 舟,马 岷. 嵌入式系统软硬件协同设计实战指南[M]. 北京:机械工业出版社,2013.

[3] 王 冰. 基于 STM32 和 FPGA 的多通道多功能数据采集器[D]. 成都:西南交通大学,2012.

[4] 王正浩. 嵌入式 8 通道高速数据采集器的设计[D]. 哈尔滨:哈尔滨理工大学,2010.

[5] Wang Jiannong, Wang Wei. The common data acquisition system based on Arm9[C]//Proc of international conference on electronic measurement & instruments. [s. l.]: [s. n.], 2011: 324-327.

[6] 李文刚. 基于 FPGA 的高速高阶 FIR 滤波器设计[D]. 成都:电子科技大学,2005.

[7] 刘文峰,李程远,李善平. 嵌入式 Linux 操作系统的研究[J]. 浙江大学学报:工学版,2004,38(4):447-452.

[8] Stevens W R, Rago S A. Advanced programming in the UNIX environment[M]. [s. l.]: Addison-Wesley, 2014.

[9] Jones M T. GNU/LINUX application programming[M]. [s. l.]: Charles River Media, 2008.

[10] 刘 磊,张凤荔,秦志光. 基于 U-boot 构建嵌入式 Linux 的 Bootloader[J]. 计算机应用研究,2007,24(12):238-240.

[11] Bovet D P, Cesati M. Understanding the Linux kernel[M]. [s. l.]: O'Reilly Media, 2002.

[12] 邵长彬,李洪亮. 用 Busybox 制作嵌入式 Linux 根文件系统[J]. 微计算机信息,2007,23(10-2):48-50.

[13] Corbet J, Kroah-Hartman G, Rubini A. Linux device drivers [M]. [s. l.]: O'Reilly Media, 2005.

[14] 张曦煌,柴志雷. 嵌入式 Web 服务器中 CGI 的特点及实现[J]. 小型微型计算机系统,2003,24(11):2046-2048.

基于SoC和嵌入式Linux的数据采集系统设计

作者： 叶梦， 谢捷如， YE Shen， XIE Jie-ru
作者单位： 南京航空航天大学 自动化学院,江苏 南京,210016
刊名： 计算机技术与发展 ISTIC
英文刊名： Computer Technology and Development
年，卷(期)： 2015(8)