

一种基于串口通信的 DSP 调试平台设计

薛志远^{1,2}, 朱 浩², 张铁军², 王东辉², 侯朝焕²

(1 中国科学院大学, 北京 100049; 2 中国科学院 声学研究所, 北京 100190)

摘 要: 给出了一种基于串口的 DSP 调试平台设计方法. 通过该方法, DSP 调试平台实现了对应用程序的加载、上传、执行与校验. 同时, 通过对 SuperV 处理器中执行信息的实时反馈, 调试平台与模拟器完成协同调试.

关键词: DSP; 串口; 协同调试; SuperV 处理器

中图分类号: TN7

文献标识码: A

文章编号: 1000-7180(2013)12-0057-04

Design of a UART-based DSP Debugging Platform

XUE Zhi-yuan^{1,2}, ZHU Hao², ZHANG Tie-jun², WANG Dong-hui², HOU Chao-huan²

(1 University of Chinese Academy of Sciences, Beijing 100049, China;

2 Institute of Acoustics, Chinese Academy of Sciences, Beijing 100190, China)

Abstract: In this paper, a method of UART-based DSP debugging platform design is presented. Through this method, the downloading, uploading, executing and checking process of application program are completed in the DSP debugging platform. Meanwhile, by the real-time feedback of SuperV executing information, the co-debug process is implemented with the help of simulator.

Key words: DSP; UART; co-debug; SuperV processor

1 引言

随着数字信号处理器(DSP)运行速度不断提高, DSP 芯片在高端应用领域有着越来越广泛的应用^[1]. 然而, 随着应用程序规模地扩大与复杂度地提高, 应用程序在 DSP 中运行时出现故障后, 如何对故障点的快速定位变成一个急需解决的问题. 因此, 迫切地需要设计一种合适的 DSP 应用程序调试平台, 通过采用合适的通信机制来完成对应用程序的加载以及对应用程序执行过程中信息的实时反馈. 针对所反馈的信息, 通过与 Gold Model 进行比较, 达到快速定位的目的.

针对以上问题, 本文以 FPGA 为平台, 提出了一种基于串口通信的 DSP 调试平台设计方法, 并以一款自主研发的 SuperV 处理器为研究对象, 完成了对大型应用程序的调试.

2 常用的调试方法

现有的调试方法中比较常用的是基于硬件的调试与基于软件的调试. 基于硬件的调试主要是 JTAG 方法. 基于 JTAG 的方法在芯片中集成硬件调试模块, 通过扫描移位寄存器链将测试

指令从测试数据端口移入, 下位机根据指令的要求来实现相应的数据输入, 数据输出以及调试操作^[2]. 但是这种方法缺乏下位机对上位机的反馈机制, 当上位机向下位机发送操作命令后, 下位机处于何种状态以及传送的数据是否正确是未知的. 基于软件的调试方法是在芯片中以软件的方式建立一种软件调试环境, 以此环境为依托, 上位机与下位机建立了一种良好的接收与反馈机制, 上位机与下位机之间的通信也因此更加透明.

本文通过对 SuperV 处理器自身的结构特点进行分析, 采用了基于软件的调试方法. 通过在下位机

建立软件调试平台,以及在上位机建立相应的配套环境,与模拟器协同进行调试。

3 总体框架

本文以基于软件的调试为基本方法,设计了基于串口的调试平台,结构框图如图 1 所示。

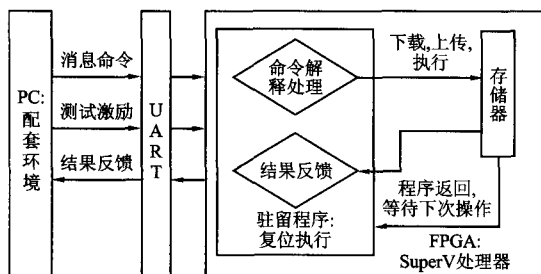


图 1 调试平台结构框图

由图 1 可看出,本设计通过初始化 Memory 的方式在 FPGA 中固化了一段驻留程序,其主要负责上位机(PC)与下位机(FPGA)之间的通信。通过 UART 以及下位机中的驻留程序,下位机可以对上位机发送的消息命令进行处理,并引导 SuperV 处理器根据所接收到的命令做出相应的行为。在 SuperV 处理器操作完成后,驻留程序可以将结果对上位机进行反馈。程序执行完毕后返回驻留程序,继续等待下次操作命令。同时,本文在上位机中设计了配套环境,以此实现对消息命令与应用程序的发送,方便了调试过程。

在通信过程中,上位机每发起一次通信请求,下位机便会通过驻留程序向上位机进行反馈,通过这种反馈,既可以确保所传输数据的稳定性,上位机又可以获取 SuperV 内部执行信息,方便调试。

4 调试平台设计

4.1 串口 API 与配套环境设计

本设计平台采用串口进行通信。为此,本文在上位机中设计了基于串口的 API 程序。

API 是附带在 Windows 内部的一个重要组成部分,可以看作是 Windows 系统为在其下运行各种开发系统提供的开放式通用功能增强接口^[3]。API 可以对串口的基本属性进行设置,并通过内置读写缓冲完成对串口读写操作。为了方便调试,本文在串口 API 程序中增加了比较模块,可以通过模拟器与硬件执行信息的比较来实现对故障点的定位。针对本平台的特点,我们采用了异步操作的串口操作方式,将波特率设为 115200,无奇偶校验,使用 1 位停止位。

为提高硬件调试效率,降低调试代价,本文为调试平台开发了配套环境。此配套环境为用户配备了命令行界面输入方式,以及独特的脚本编程语言编写脚本,用户通过此环境可以很方便地实现上位机对下位机行为的控制,以及获得下位机的反馈信息。

通过 API 程序与此配套环境的配合,建立了一种较佳的通信机制,用一种自动化的方式实现了辅助驻留程序对应用程序的下载、执行与结果的上传。

4.2 驻留程序的设计

固化的驻留程序主要建立了一种上位机与下位机之间的通信机制。这种通信机制用于约定上位机如何发送握手信号或数据流,以及下位机如何接收数据流和解释握手信号并执行命令。

如图 2 所示,驻留程序采用了两层 Client/Server 的通信结构^[4]。其中上位机为客户端,负责生成操作命令并通过串口 API 程序向下位机的服务器发出请求;服务器接收到操作请求后,做出相应的操作,然后向客户端发送反馈数据,客户端接收反馈数据并对反馈数据进行处理。

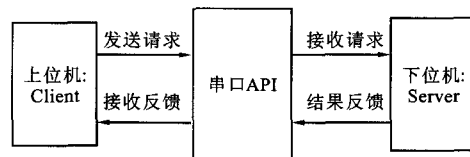


图 2 通信机制

下位机的行为模式可分为数据下载,数据上传,命令执行,数据校验等。下位机根据上位机所发送的命令来进行相应的行为操作。为了简化下位机中驻留程序对消息命令的处理,上位机与下位机之间约定了一种消息命令的格式,命令按功能可分为通信行为命令和用于指导下位机处理执行的指导命令。如表 1 所示,通信行为命令包括 COMM 和 comm,其中 COMM 命令表示一次通信的开始,comm 表示此次通信的终止,两者之间可以包含一次有效的操作即 EXEC 等指导命令。

驻留程序是由两层嵌套循环实现的,基本框架如图 3 所示。外层循环用于查询接收到的命令是否为通信行为命令,内层循环用于查询接收到的命令是否为指导命令。当外层循环查询到通信行为之后便进行内层循环,内层循环向上位机发送反馈并等待接收指导命令,接收到指导命令后返回外层循环继续执行,根据所接收到的指导命令进行相应的操作,操作完成后等待进行下一次操作。下位机在每一次通信行为发生变化时都会向上位机进行反馈,因

此,下位机的行为对上位机来说是透明的,上位机可以针对反馈消息对下位机的行为进行灵活控制。

表 1 消息命令功能与格式

消息命令	功能描述	发送模式
通信行为命令	通信开始。	“COMM”
	通信结束。	‘comm’
指导命令	下载,上位机向下位机传输文件。	‘DNLD’+ADDR+SIZE
	程序执行。	‘EXEC’+ADDR
	上传,下位机向上位机写内存数据。	‘UPLD’+ADDR+SIZE
	下位机校验接收的数据文件是否正确。	‘CHCK’

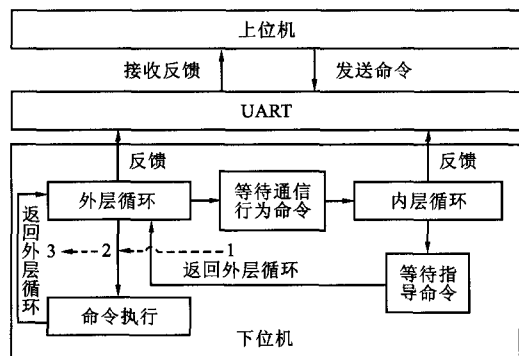


图 3 驻留程序基本框架

为了存储所要发送的消息命令、反馈信息、地址、大小等信息,本文在驻留程序中定义了 8 个虚拟寄存器,每个虚拟寄存器 32 位。下位机这 8 个虚拟寄存器并不驻留在任何存储部件中,而是动态地存在于上下位机之间消息发送或接收的数据流中。

驻留程序采用两种数据发送模式,模式 A 为除下载命令和上传命令外的传输模式,下位机只需要解析 8 个虚拟寄存器中的有效数据;模式 B 分前缀和后缀两部分,前缀与模式 A 一致,后缀为所上传或下载的数据流,当下位机反馈错误信息时,上位机将中止后缀数据流的写 UART 操作。

4.3 工作流程

图 4 中以下载命令为例说明了一下驻留程序的工作流程:

驻留程序从上电到操作结束共经过引发通信、命令下载、命令执行与数据校验四个过程。

(1)引发通信. 每进行一次操作之前,需要先引发一次通信,目的是建立一次握手,确保开始通信. 系统上电后,驻留程序对 UART 共享缓存进行查询,等待上位机发起一次操作请求. 当上位机需要引发一次通信时,便将相应的操作命令写入共享缓存,之后上位机便处于等待反馈状态. 下位机接收到上位机发送的引发通信命令后便向上位机发送反馈. 上位机接收到反馈后上位机与下位机便完成了一次握手。

(2)命令下载. 类似地,随后上位机与下位机完

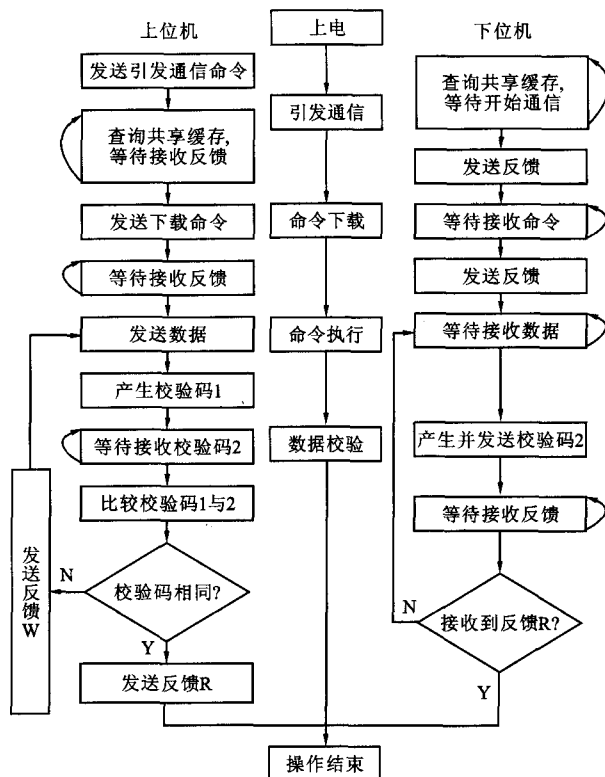


图 4 工作流程

成一次下载命令的握手,上位机生成下载地址与大小等信息,与下载数据一同向下位机发送。

(3)命令执行与数据校验. 下位机将下载数据下载到相应的地址中,同时下位机根据所接收到的数据生成奇偶校验码,并将校验码发送给上位机,上位机接收到校验码后,与自己产生的校验码进行比较,如果校验码相同,则操作结束,如果不相同,则重新进行一次下载数据操作。

上传命令与执行命令通信过程与上类似. 在运行执行命令时,驻留程序接收到执行信息后便将控制权交给应用程序,在应用程序执行完毕后返回驻留程序,等待进行下一次命令操作。

5 调试与实现结果

本文以串口为基本通信机制设计的平台实现了

对 DSP 的调试. 调试过程的关键在于对故障点的定位^[5]. 本文采用了模拟器作为 Gold Model 与硬件进行比较, 调试过程如图 5 所示. SuperV 调试平台对触发事件进行检测, 将此事件状态下的执行信息反馈给串口 API 中的比较模块, 与此同时, 模拟器亦将相应的信息发送给比较模块, 比较模块通过比对来判定两者信息是否一致, 如果一致, 则说明 SuperV 运行准确无误, 如果不一致, 则将故障信息通过串口反馈给 PC.

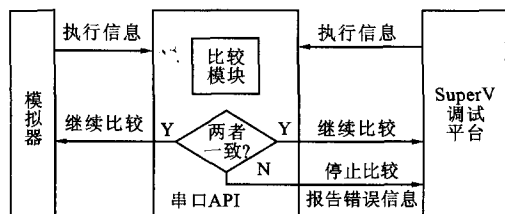


图 5 调试过程

在比较过程中, 可以将执行信息的比较划分为细粒度的指令级、中粒度的循环级与粗粒度的函数级. 在比较初期, 本文首先对函数级进行比较, 以一种粗粒度的方式来确定故障点的大致范围; 然后在此范围内依次对循环级与指令级进行中粒度与细粒度比较, 最终对故障点进行定位.

使用本平台, 在 mp3 解码程序、mpeg2 解码程序、语音识别程序等应用程序出现故障时, 本平台实现了对故障点的快速定位, 最终完成了对这些应用程序的调试, 使其能正确地运行在 SuperV 处理器上.

6 结束语

本文介绍了一种基于串口的 DSP 调试平台设

计方法. 通过以串口通信基础的驻留程序, 实现了对应用程序的加载, 以及上位机对应用程序执行的灵活控制. 在驻留程序中增加了校验机制, 确保了所传输数据的正确性. 本文设计的调试平台具有良好的反馈特性, 通过上、下位机的信息实时交互, 实现了协同调试.

参考文献:

- [1] 吴琼之, 南方, 张峰. 并行 DSP 系统软件调试器设计与实现[J]. 北京理工大学学报, 2011, 31(7): 855-858.
- [2] 高瑛珂, 王琪, 李泉泉, 等. 一种基于 JTAG 接口的片上调试与性能分析方法[J]. 微电子学与计算机, 2012, 29(7): 68-71.
- [3] 罗云虎. 一种利用 Win32 API 函数实现串口实时通讯的方法[J]. 现代电子技术, 2004, 27(11): 70-73.
- [4] 王彦臣, 凌伟, 黄廉卿. 基于 Client/Server 模式下实现大数据量文件的传输[J]. 光学精密工程, 2003, 11(4): 400-403.
- [5] Wang Qi, Zhu Hao, Gao Yingke, et al. Accelerating processor verification based on ESL model[C]// Recent Advances in Computer Science and Information Engineering. Berlin: Springer-Verlag, 2012: 745-751.

作者简介:



薛志远 男, (1986—), 博士研究生. 研究方向为集成电路验证.

(上接第 56 页)

- [3] 孙林, 罗大庸. 基于遗传算法的 Web 服务器集群负载均衡研究[J]. 计算机测量与控制, 2006, 14(10): 1364-1365.
- [4] 陕宣耀, 王芳. 一种改进的小生境遗传算法[J]. 重庆邮电学院学报, 2005, 17(6): 721-723.
- [5] 黄聪明, 陈湘秀. 小生境遗传算法的改进[J]. 北京理工大学学报, 2004, 24(8): 675-678.

- [6] 王强. 基于 LVS 集群的一种负载均衡改进算法的研究与实现[D]. 成都: 电子科技大学, 2006.
- [7] 章文嵩. 可伸缩网络服务的研究与实现[D]. 北京: 国防科技大学, 2000: 109.

作者简介:

杨越 女, (1987—), 硕士研究生. 研究方向为计算机网络通信.