

基于 GDB 的多核 DSP 平台调试器的设计与实现

李 琮, 王竹平, 费晓琪

(西安微电子技术研究所, 陕西 西安 710054)

摘 要: 随着科技发展, 多核处理器正以其高性能、低功耗等优势逐步取代传统的单核处理器成为主流。如何为多核平台开发搭建可靠高效的调试环境亦成为我们所面临的问题之一。基于 GDB 的移植, 针对多核 DSP 平台提出了一种调试环境的构建方法, 能够满足各种主流的调试要求, 并具有实现简单、可靠性高的优点。

关键词: 多核 DSP; 多核调试; GDB 移植

中图分类号: TN402

文献标识码: A

文章编号: 1000-7180(2015)01-0072-04

Design and Implementation of Debugger of Multi-core DSP Platform Based on GDB

LI Cong, WANG Zhu-ping, FEI Xiao-qi

(Xi'an Microelectronic Technology Institute, Xi'an 710054, China)

Abstract: With the development of science and technology, multicore processors are gradually replacing the traditional single-core processors with its advantages of high performance, low power consumption and is becoming mainstream. How to build a reliable and efficient debugging environment for multicore platform has become one of the problems we face. In this paper, we proposed a method of constructing the debugging environment for the multicore DSP platform based on porting GDB. The method can meet a variety of mainstream debugging requirements, and has a simple, high reliability advantages achieved.

Key words: multicore DSP; multicore debugging; porting GDB

1 引言

多核处理器将多个完全功能的核心集成在同一个芯片内, 整个芯片作为一个统一的结构对外提供服务。相对于单核处理器, 多核处理器拥有控制逻辑简单、高主频、低通信延迟、低功耗、设计和验证周期短等优势^[1]。然而, 在多核处理器带来丰厚回报的同时, 也必须认识到其作为一种新的架构, 必然面临诸多新的技术问题的挑战。如何针对多核平台开发设计调试环境便是问题之一。本文针对西安微电子技术研究所自主研发的一款多核 DSP 芯片, 提出了一种调试环境的设计方案, 能够满足各种主流的调试要求。

2 硬件平台

LCSA3201A 是一款由西安微电子技术研究所自主研发的适用于多通道控制信号处理应用的高性能定点 DSP 处理器。如图 1 所示, 它基于片上总线, 一共集成了 8 个高性能的 DSP 处理器核。它们可以根据不同通道的处理特点分别加载不同的应用程序, 也可以加载同样的程序, 在中心处理器核控制下协同处理同一任务。

LCSA3201A 上集成了调试支持单元 (Debug Support Unit, DSU), 其用来实现调试用计算机或其他监控设备对本芯片所属全局地址空间的访问, 如对处理器核程序的加载、处理器运行参数的配置、对处理器核数据存储器资源的访问、对处理器核程序运行的控制、接口模块和程序运行的调试等。

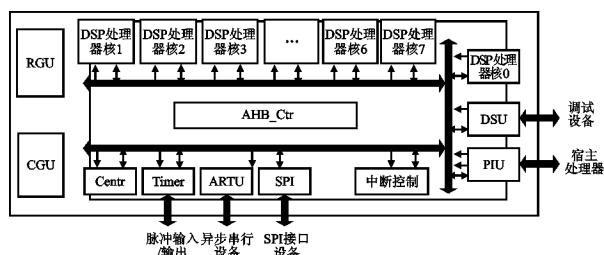


图1 LCSA3201A 结构框图

基于一个通用串行收发器和协议控制器,调试用计算机或其他监控设备可以得到程序运行时处理器核内各个寄存器的内容并显示给使用者,也可以得到片上存储器的内容。当对某段程序运行有疑问时,可以控制特定寄存器让程序暂停,以便使用者对当前信息进行分析。

3 GDB 调试

GDB(GNU Debugger)是一个功能强大且全面的开源的 GNU 源码级调试器,支持 C、C++、Java、Pascal、Fortran 等多种程序语言的调试。GDB 根据编译配制参数的不同,可支持多种计算机架构平台以及目标文件格式,并且为移植到新的硬件平台留下了对应接口。

3.1 工作原理

GDB 的工作原理图如图 2 所示。

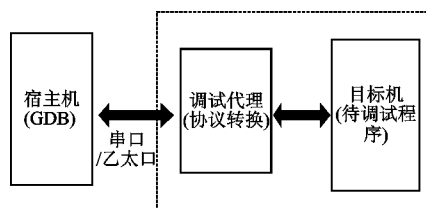


图2 GDB 工作原理

一般而言,宿主机及 GDB 称为主机端,调试代理、目标机及待调试程序共称为目标端。调试时,主机端通过串口、网口等通信通道与目标端连接,利用远程串行通信协议(RSP)对目标端发送各种调试命令,如:设置断点、查看寄存器、查看内存、控制运行等。目标端调试代理接收并解析这些命令,控制目标机完成相应的调试操作,再将结果送回给主机端。

通常调试代理作为一段代码嵌入到目标系统中,被称为调试桩(gdbstub)。然而,这种侵入式的调试方式有如下缺点^[2]:插入到目标系统中的调试桩代码可能对被调试的程序产生各种影响,如占用通信端口、改变被调试程序运行时序等;而在对嵌入式系统进行底层调试时,如面临裸机环境开发 boot-

loader 或者是在开发操作系统或者驱动程序,需要进入操作系统内部调试时,调试桩就很难满足这样的需求。此外,对 GDB 进行多核控制扩展时,要求调试代理增加新的功能,如:切换不同的内核上下文等,此时对调试桩进行扩展也很困难。所以,在硬件 DSU 的支持下,本文采用非侵入的方式实现调试代理,使其作为与 GDB 同层次的应用运行在宿主主机上。

2.2 RSP 协议

GDB 远程串行通信协议 RSP (GDB Remote Serial Protocol) 是基于消息的 ASCII 码字符流协议,宿主机和目标机双方通过这个协议进行数据的接收和发送^[3]。与其他的通信协议一样,RSP 协议包含了起始位,结束位和校验和。其一般格式如下:

\$ <data> # <checksum>

其中,\$ 是起始位,# 是结束位,<data> 是数据段,checksum 是校验和,它将数据段所有字符的 ASCII 码相加然后模 256 求得。

当一个消息发送了之后,接受方应该对收到的消息进行数据的校验,并根据校验结果向发送端返回相应的信息:

+:表示接收校验和正确,并且接收方已准备好接收下一个数据包;

-:表示接收校验和不正确,要求重新发送消息。

目标机端对来自 gdb 的消息的响应方式有两种:一种是使用 data 或 OK(依赖于消息);一种是目标机端定义的错误代码。当 gdb 接受的错误代码时,它会通过 gdb 控制台向用户报告错误号。

调试的命令是通过第一个字母进行区别的。以下是比较常用的调试命令、参数:

g:查看 CPU 寄存器的值;

GXX:设置 CPU 寄存器的值为 XX;

maddr,lenth:从 addr 位置开始读 lenth 字节的数据;

Maddr,lenth;XX:从 addr 位置开始写 lenth 字节的数据 XX;

c/caddr:在当前位置上继续执行程序或从地址 addr 开始执行程序;

s/saddr:单步执行当前的指令,或者从指定的 addr 位置单步执行;

k:杀掉 target 进程;

?打印出最近的信号(signal)值,即指出目标机暂停的原因。

3.3 GDB 的移植

GDB 由前端和后端组成.

GDB 前端主要包括 GDB 中与具体芯片无关的代码,如各种文件的支持代码,读取调试信息的代码,设置断点、观察点等相关的代码,还包括 GDB 的机器语言接口(Machine Interface,MI)等.

GDB 后端是指与计算机体系结构密切相关的那部分代码. GDB 依赖这部分代码来实现对 CPU 行为的理解和操控,所以这部分代码被称为 target dependent code,包含这些代码的文件也被称为 target dependent file (tdep)^[4]. 一般而言,移植 GDB 其实就是为 GDB 写一个新的后端.

4 多核 DSP 平台调试环境的设计实现

针对 LCSA3201A 硬件平台,多核调试系统至少应该满足以下几点功能.

(1)能对所有处理器核能进行统一的控制管理,如使其继续运行至断点或停止运行;统一的控制管理使得各处理器核之间的同步得到保证,减少程序调试态与运行态差别,提供更准确的调试结果.

(2)能对每个处理器核上传独立的程序,并在不影响其他核调试或运行的情况下,完成单核的基本调试功能:如插入断点、删除断点、单步运行、查看寄存器、查看内存、查看变量等.

(3)调试控制权可以在不同的处理器核之间进行切换.

目前版本的 GDB 并不支持多核调试,可以尝试在移植 GDB 的基础上,整理修改 GDB 代码结构,使其以多线程的方式对多核调试进行支持.这样要求开发者对 GDB 代码整体包括细节都非常清楚,工作量大,难度很高.而且对于 GDB 前端代码结构大量修改并不符合 GDB 移植的一般准则,将开发者从前端代码解放使其专注于后端代码实现的优势不复存在,同时会引发很多的不确定性,本来已经测试稳定的基本调试功能也可能因此受到影响.在调试时,每个处理器核大部分情况下还是独立于其他核单独进行调试的,并且 GDB 需要通过调试代理与 DSU 通信来控制目标完成相关调试操作.基于此,我们可以使每个处理器核对应一个 GDB,并对 GDB 做简单的多核调试扩充,使它通过调试代理对其他处理器核及 GDB 有一定的访问控制权利.然后由调试代理与开发环境对所有的 GDB 进行统一的控制调度,这样,不仅实现了多核调试功能,而且保证了 GDB 基本调试功能不受多核调试影响,能够正常运行,实

现简单可靠.方案设计如图 3 所示.

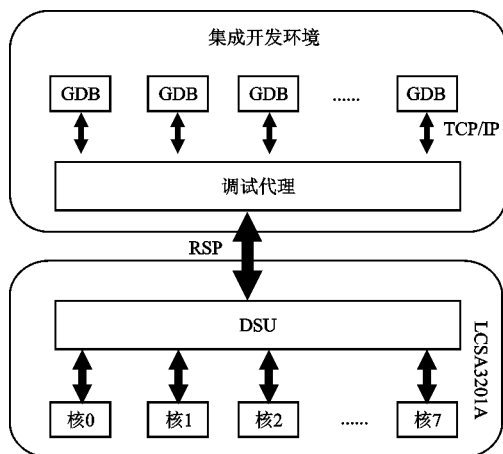


图 3 多核平台调试环境设计方案

调试环境主要由调试器(GDB)与调试代理构成,它们通过 TCP/IP 协议进行数据传输.调试时,用户选择需要的 CPU 核(GDB),输入调试命令,如:在被调试程序中进行断点的插入/删除,通过单步、继续执行等操作命令来控制程序的执行,或者在程序暂停时对寄存器、程序变量等进行查询等,调试代理通过 RSP 协议解析调试命令,分析命令由哪个 CPU 核(GDB)发出,然后控制硬件完成调试要求,再将结果返回给用户.

4.1 调试器的设计与实现

调试器的设计基于 GDB 的移植.然而,它除了要满足 GDB 所提供的一般调试功能以外,针对多核调试的特性,还应该做以下的功能扩展:

- (1)能够控制调试环境中除自己以外的一个或多个 GDB 继续运行;
- (2)能够查看调试环境中自己对应以外其他的处理器核的内存以及寄存器的内容;
- (3)能够主动交出控制权,切换调试环境中其他的 GDB 到前台工作;

因此,为 LCSA3201A 设计调试器的具体步骤如下^[5]所示.

(1) 确保在 bfd 目录中存在一个为目标体系结构的执行而存在的 BFD(binary file descriptor 二进制文件描述器).因为目前并不存在,我们通过修改已存在的相似者来创建一个.

(2) 在 opcodes 目录中,为目标体系结构实现一个反汇编器.

(3) 为 GDB 添加新的后端文件,定义目标体系结构.并修改相应的配制文件,使其在编译时可以生成支持当前目标结构的 GDB.

(4) 根据目标体系结构硬件特性,对 GDB 前端进行相应的修改以使得 GDB 的某些功能可以正常运行,如读写内存、变量等。

(5) 添加新的用户命令及机器命令,为用户及机器调用留下多核调试功能接口。

(6) 扩展 RSP 协议,以满足上面提到的多核调试新的功能要求。

4.2 调试代理的设计实现

调试代理作为调试器与目标板的中介,需要负责连接多个调试器,处理它们并发的调试请求,通过 RSP 协议解析调试命令,控制硬件完成调试操作,再将正确的结果回传。具体应该实现的功能如下所示。

(1) 调试代理需要作为一个并发服务器实现。系统以进程的形式逐个启动调试器,并向调试代理发起连接。调试代理必须能正确处理它们的并发连接以及调试请求。此外,调试代理对目标 DSU 的控制接口作为唯一资源,为多个调试器所共享,其间的互斥工作也必须处理妥当。

(2) 调试代理需要正确的解析 RSP 协议中相关命令以及扩展命令,并控制硬件完成相应的操作,将结果返回给用户。能让 GDB 运行的最小 RSP 命令集合为 g、G、m、M、c,其具体含义可见上文对 RSP 协议的讨论。

(3) 调试代理需要准确的识别出调试请求由哪一个调试器发出,并控制相应的处理器核完成相应的调试操作再将结果回传。硬件方面,由于各处理器核的内存结构完全相同,仅由地址空间区分。因此,在接受调试器连接时,可为该进程编号,这样,便很容易得出调试器与相应处理器核的映射关系,从而进行正确的操作并返回结果。

(4) 调试代理需要提供全局统一的控制接口,以使所有的处理器核同时运行或停止。多核调试,各核之间的同步非常重要,能对其进行统一的控制,便可以使调试态与运行态保持一致,从而得到更为准确的调试结果。调试代理可以通过依次控制各个调试器操作相应

的处理器核来实现这样的功能。然而,这样的方法必然有调用延迟,实时性差。如果硬件支持的话,可以采用先通过硬件手段同时停止所有处理器核,再将调试器所需要的调试信息回传给各个调试器的方法。

5 结束语

本文针对多核 DSP 平台 LCSA3201A,结合 GDB,设计实现了一种能满足主流调试要求的源码级多核调试系统。该系统实现简单、可靠性好,但同时占用系统资源较多。不过,多核调试技术依然在不断的完善和发展中,只要对调试代理进行相关的完善和升级,必然能实现更好更强的调试环境。

参考文献:

- [1] 贺松建. 多核处理器技术浅析[J]. 科技促进发展, 2009(2):60-63.
- [2] 刘洋,张毓森,甄奇. 基于 GDB 的 ARM 嵌入式调试系统的设计与实现[J]. 军事通信技术, 2007, 28(Sup): 22-28.
- [3] Bill Gatliff. Embedding with GNU: the gdb remote serial protocol[J]. Embedded Systems Programming, 1999(9):109-116.
- [4] 杨文博. GNU Debugger 的移植及原理分析[D]. 南京: 南京大学, 2007.
- [5] Jeremy Bennett. Howto: porting the GNU Debugger—Practical Experience with the OpenRISC 1000 Architecture[J]. Embedded Software Development, 2008, 2(3):1-74.

作者简介:

李琮 男, (1989-), 硕士研究生. 研究方向为计算机应用技术. E-mail: lc19890709@126.com.

王竹平 男, (1963-), 研究员, 总工程师. 研究方向为嵌入式计算机软硬件设计.

费晓琪 男, (1972-), 高级工程师. 研究方向为嵌入式计算机软件设计.

(上接第 71 页)

- [14] 孙劲光, 汪洁, 姜文涛, 等. 改进的分块算法在矩形图像加密中的应用[J]. 计算机应用研究, 2013, 30(1): 282-284.
- [15] Jing Wang, Guoping Jiang. A self-adaptive parallel encryption algorithm based on discrete 2D-logistic map [J]. International Journal of Modern Nonlinear Theory and Application, 2013(2): 89-96.

作者简介:

刘祝华 男, (1977-), 硕士, 讲师. 研究方向为信息安全、数字图像处理等. E-mail: happy_lzh@126.com.

袁文 男, (1972-), 博士, 教授. 研究方向为信号检测与处理.