

基于 MCU 的嵌入式系统的 Bootloader 设计

温 暖, 杨维明, 彭菊红, 王旭光

(湖北大学 计算机与信息工程学院, 湖北 武汉 430062)

摘 要: 在基于 MCU 的嵌入式应用中, 应用程序一般通过编程器烧写到存储器中, 该方法的缺点是应用程序的下载和更新必须依赖仿真器, 不但操作麻烦而且成本高效率低, 对比设计的 Bootloader 首先会对 MCU 及相关外设做初始化, 然后仅通过 UART 口就可以实现嵌入式应用程序下载, 并自动跳转并运行该应用程序, 为嵌入式应用程序的升级与维护带来便利。

关键词: 自编程系统; 嵌入式应用; MCU; Bootloader

中图分类号: TN311

文献标识码: A

文章编号: 1000-7180(2018)03-0079-04

Design of Embedded System Bootloader Based on MCU

WEN Nuan, YANG Wei-ming, PENG Ju-hong, WANG Xu-guang

(Faculty of Computer and Information Engineering, Hubei University, Wuhan 430062, China)

Abstract: In the embedded application system based on MCU, the application program is commonly written to internal memory by emulator. In which the download and update of the application program must rely on emulator. It is not only troublesome, but also cost efficient. The design of the Bootloader will be the first to do the initialization of MCU and peripherals, then you can download the embedded application only through the UART port, and automatically jump and run the application, it brings great convenience for the upgrade and maintenance of embedded applications.

Key words: self-programming system; embedded application; MCU; Bootloader

1 引言

Bootloader 是固化在嵌入式系统特定存储区域的一段代码, 在系统上电或者复位后, 处理器会执行该代码区域的第一条指令. 通过 Bootloader 进行代码的烧写和维护要比编程器更加灵活, 通常可利用 MCU 的 UART 接口连到计算机的 RS232 口, 通过专门设计的固件程序来编程内部存储器实现升级和^[1-3]. 本文应用的 MCU 在存储单元上, 有 32 kB 的 FLASH(0x0000~0x7FFF), 以及 2 kB 的 ROM 区(0xF800~0xFFFF), ROM 掉电不丢失可以存放 Bootloader 代码, 在硬件上符合 Bootloader 的设计要求. 本文设计的 Bootloader 不但完成了处理器及外设初始化, 同时实现了嵌入式应用程序的下载与调试.

2 Bootloader 指令设计

MCU 通过 UART 口与计算机进行数据交互, 所以需要制定一套规范的数据格式, 由计算机发送不同的指令对 MCU 执行不同的数据操作, 以及 MCU 执行完不同的操作后对计算机做出不同的响应. 该套指令是由 8 字节的执行指令串以及响应组成, MCU 接收完一组 8 个字节的指令串, 并执行完该指令对应的操作后, 向计算机返回一个 0xE0 作为结束标志.

MCU 系统的指令不仅支持对 FLASH 的读写操作, 还支持对寄存器以及 RAM 的读写功能, 而这些指令既可以使 MCU 更好地工作, 也能破坏 MCU 的正常运行以及代码的保护^[4]. 所以本文设计了 MCU 两种工作权限, 当工作在普通权限的模式下,

只开放对 Flash 的读写操作的指令,当工作在超级权限的模式下,支持所有的操作指令.这样做的优点可以使 MCU 程序运行的更稳定,增强了安全性.

3 Bootloader 设计步骤

MCU 有两个独立的存储区域,ROM 和 FLASH 区域,Bootloader 代码固化在 ROM 区中^[5].本文所用的 MCU,在硬件上实现了上电复位后自动跳到 0xF800 处取指,ROM 的地址范围在 (0xF800 ~ 0xFFFF),所以 MCU 上电复位后运行 Bootloader 程序,下载完应用程序后,修改寄存器的值,在下次复位后会跳转到 0x0000 处运行应用程序.

3.1 Bootloader 主函数设计

本文使用的 MCU 共有 128 个特殊功能寄存器,首先在头文件中定义的拓展寄存器.Bootloader 主函数设计流程图如图 1 所示,主函数主要做两个工作,一是相关寄存器的初始化,二是确定 MCU 工作的模式.Bootloader 的一些初始化,如图 2 所示.

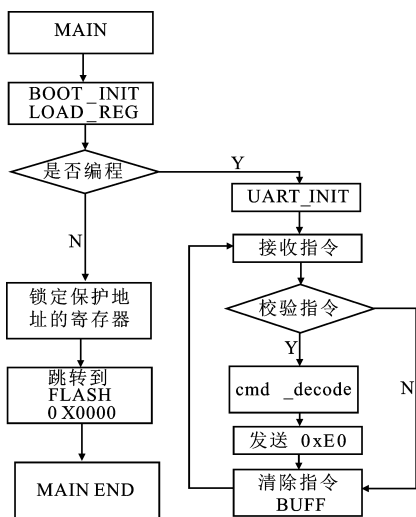


图 1 Main 函数流程图

3.2 指令的解码函数设计

在 MAIN 中存放的指令 BUFF,也就是指令数组会传入 cmd_decode 函数中,进入该函数首先校验指令数组最后两位是否是 0x73 和 0x00,若不是,则清除指令数组中的数据并退出该函数;若是,MCU 则进行下一步的解码和执行操作.

因为 Flash 的读写指令工作在两种权限的 MCU 中都支持,所以不需要标志位判断,若该指令是 Flash 读写指令就执行;对于超级权限里的指令,会先判断 MCU 当前模式,若是超级权限模式就执行,不是就跳过执行.

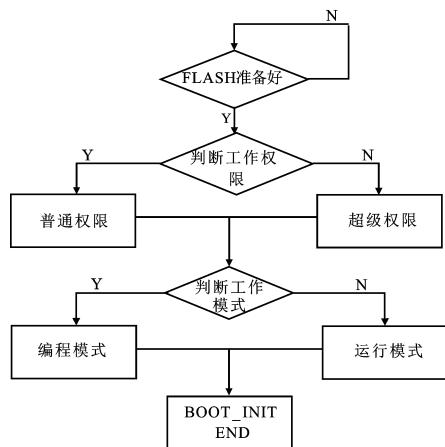


图 2 Boot 初始化流程图

3.3 片外 RAM 实现 SFR 寻址

Bootloader 读写 SFR(特殊功能寄存器),只有当指令组传进来时,寄存器地址才会确定,而寄存器间接寻址只能访问到内部 RAM 区,而不能访问 SFR 区域的数据^[6].可以用 switch-case 语句,将 128 个 SFR 的读写操作全部写入,再用 case 语句匹配地址值,执行相应的操作.这种方案可实现 SFR 读写,但对于 2 kB 的 ROM 来说,消耗了很多的存储空间.

本文提出利用片外 RAM 实现 SFR 寻址,可节省内部存储空间.由于程序是以机器码的形式存放在存储区域的,不同的汇编语言对应不同的机器码.当 MCU 接收到读写 SFR 的指令时,同时也接收到了操作的地址和数据,此时获取地址和数据,将其翻译成直接寻址对应的机器码,并写到片外 RAM 的指定的地址区域中,然后再保存当前运行地址到 RAM 中,当 RAM 指令执行完再跳转回 ROM 中,执行 Bootloader 下面的指令.

4 Bootloader 功能仿真与分析

Bootloader 功能仿真的结构框图如图 3 所示,计算机通过串口向 MCU 传输指令,Bootloader 接收到并执行相应的操作,首先要在原先的 Teast-beach 中例化一个 UART 控制器,并且连接到 MCU 模块的 UART 接口上^[7].向串口输入指令,MCU 执行对应的操作,查看相关寄存器值,验证 Bootloader 设计.本文仿真了 SFR 读写、RAM 读写和 Flash 读写过程.

4.1 SFR 读写操作仿真

SFR 读写操作的仿真结果如图 4 所示,“0x80”“0x30”“0x00”“0x00”“0xA0”“0xAA”等是串口接收

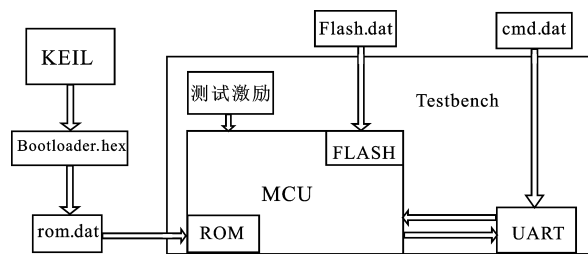


图3 Bootloader功能仿真结构框图

到的指令数据串,对应的操作应该是向 0xA0 特殊功能寄存器写入 0xAA,且 0xA0 特殊功能寄存器对应的是 P2 口,所以 P2 端口也会输出 0xAA。

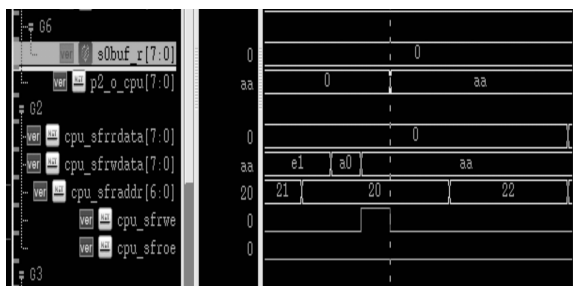


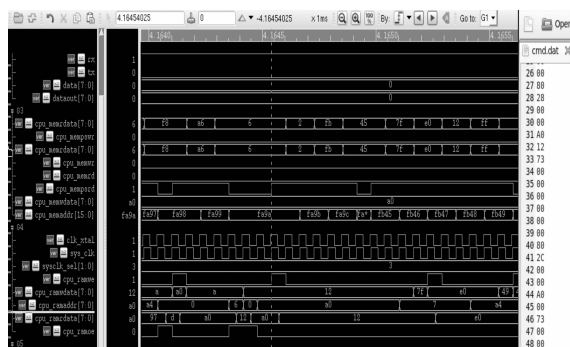
图4 SFR读写操作仿真图

当MCU接受完指令串后,SFR写信号拉高时,SFR地址总线的低7位地址为0x20,最高一位默认是“1”,所以实际SFR操作的地址是0xA0,SFR数据总线的数据是0xAA,执行的操作就是向0xA0地址处的特殊功能寄存器写入0xAA,此时MCU的P2口由0x00变为了0xAA,和预期输入的指令对应的操作一致,SFR读写过程符合要求。

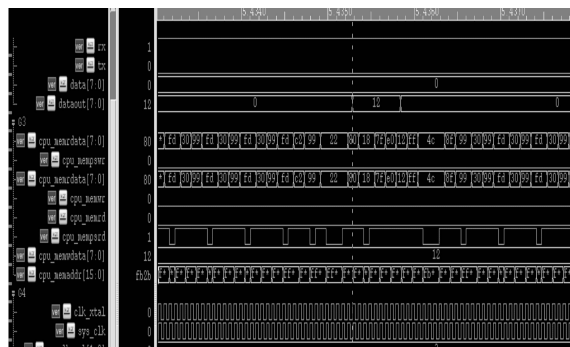
4.2 RAM读写操作仿真

RAM写操作的仿真结果如图5(a)所示,“0x80”“0x28”“0x00”“0x00”“0xA0”“0x12”等是cmd.dat文件中的指令数据串,通过串口MCU正确无误的接收到该指令数据串,且对应的操作是向RAM的0xA0地址处写入0x12,可以看出图中虚线处的片内RAM写标志位cpu_ramwr为“1”,此时片内RAM地址线cpu_ramaddr[7:0]的值为“0xA0”,数据线cpu_ramdata[7:0]的值为“0x12”,故写片内RAM符合设计要求。

当MCU接收到读操作指令,MCU读取片内RAM的“0xA0”地址处,在返回响应命令0xE0前,把读到的数据经串口送出。从图5(b)可看出data_out[7:0]串口输出BUFF的数据由“0”变成了“0x12”,读到的数据和之前写入的一致,故读RAM设计也符合要求。



(a) RAM写仿真图



(b) RAM读仿真图

图5 RAM读写仿真图

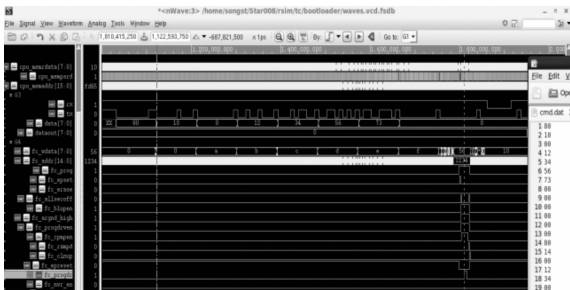
4.3 Flash读写操作仿真

Flash的编程擦除仿真重点是验证Bootloader通过串口向Flash写入的数据与读出的数据是否一致,写入操作的仿真图如图6(a)所示,仿真指令是向Flash的地址“0X1234”处写入“0x56”,如图中虚线所示,Flash的编程信号fc_progdr有效时,地址线fc_addr是“0x1234”,数据线是“0x56”,通过串口发出读Flash该地址的指令。

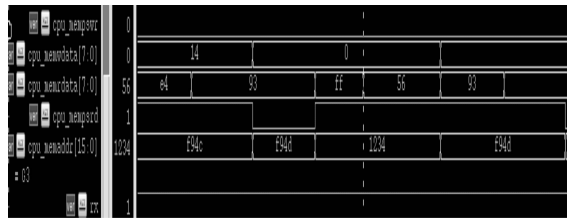
当MCU接受到读Flash指令,并开始执行,放大波形,如图6(b)所示。可看出,当读信号cpu_mempsrd有效时,地址线cpu_memaddr[15:0]地址是“0X1234”,数据线cpu_memrdata数据是“0x56”,和之前写入一致,故读写Flash符合设计要求。因为应用程序是存放在Flash的数据,所以可实现MCU应用程序的下载升级功能。

5 结束语

本文分析了基于MCU的Bootloader的设计方法,具体设计了Bootloader的指令集,其主函数包括MCU初始化,设置工作模式,以及接收指令,最后到解码函数以及响应的实现。通过UART向MCU发送指令的仿真实验,实现了MCU的引导启动以及应用程序下载与维护,该方案灵活度高,安全性强,较比传统编程器下载更加经济实用,MCU运



(a) Flash写操作仿真图



(b) Flash读操作仿真图

图 6 Flash 读写操作仿真图

行 Bootloader 引导启动代码,为网络化产品的应用和开发带来很大的便利,对实际的嵌入式应用也有一定参考意义。

参考文献:

[1] 邵新颜,蔡梅琳.在 Bootloader 中实现嵌入式系统自动升级[J].单片机与嵌入式系统应用,2006,6(11):33-35.

(上接第 78 页)

还可以增加浮点运算指令和电路,以及定点运算指令和电路等。

参考文献:

[1] Rick OConnor. Intro & foundation overview [C]//4th RISC-V Workshop Proceedings. IDT,2016.

[2] 雷思磊. RISC-V 架构的开源处理器及 SoC 研究综述[J]. 单片机与嵌入式系统应用,2017,17(2):56-60.

[3] Andrew Waterman, Lee Yunsup, David A Patterson, et al. The RISC-V instruction set manual, Volume I: User-level ISA version 2.1 [R]. Berkeley: University of California, 2016.

[4] Andrew Waterman, Lee Yunsup, Rimas Avizienis, et al. The RISC-V instruction set manual, Volume II: Privileged architecture version 1.9 [R]. Berkeley: University of California, 2016.

[2] 邢小亮,詹鑫,邹雪城,等.基于 NTRU 的双向认证安全芯片设计与实现[J].微电子学与计算机,2015,32(8):172-175.

[3] 杨珂瑶,项涛.一种通用 BOOTLOADER 架构研究[J].航空计算技术,2017,47(2):131-133.

[4] 吴磊,皮智,袁宗胜.一种基于 S3C6410 的 BootLoader 的设计与实现[J].计算机应用与软件,2016,33(9):238-244.

[5] NEC Electronics. Preliminary application note: 78K0 flash memory self programming [Z]. NEC electronics,2009.

[6] 袁磊,朱怡安,兰婧.嵌入式系统 Bootloader 设计与实现[J].计算机测量与控制,2009,17(2):389-391.

[7] 刘芳,臧威. TMS320C672x DSP 引导加载系统的设计与实现[J].微电子学与计算机,2013,30(10):46-49.

作者简介:

温暖 男,(1993-),硕士研究生.研究方向为数字集成电路.

杨维明 男,(1969-),教授.研究方向为数字电路与系统、射频微波电路.

彭菊红(通讯作者) 女,(1978-),讲师.研究方向射频微电子学. E-mail:juhongpeng@hubu.edu.cn.

王旭光 男,(1982-),讲师.研究方向为射频微波滤波器设计.

[5] Muralidaran, Vijayaraghavan. A memory model for RISC-V[C] // 5th RISC-V Workshop Proceedings. IDT,2016.

[6] Caroline Trippel. A memory consistency model for RISC-V[C] // 5th RISC-V Workshop Proceedings. IDT,2016.

[7] Krste Asanovic, Interrupts[C]// 4th RISC-V Workshop Proceedings. IDT,2016.

[8] 雷思磊. 步步惊“芯”:软核处理器内部设计分析[M].北京:电子工业出版社,2013.

作者简介:

张迅珍 女,(1992-),硕士研究生.研究方向为电路与系统. E-mail:1197174756@qq.com.

梁青 女,(1966-),教授.研究方向为电路与系统、无线传感器网络.

李涛 男,(1954-),博士,教授.研究方向为计算机体系结构、计算机图形学.