

NAND Flash 仿真模块的一种验证方法

王 洋¹, 杨国武², 郑德生², 李晓瑜²

(1 电子科技大学 示范性软件学院, 四川 成都 611731; 2 电子科技大学 计算机科学与工程学院, 四川 成都 611731)

摘 要: 根据 NAND Flash 硬件设备的工作原理和过程, 利用线性时态逻辑建模语言描述该硬件设备读写状态的性质, 分析了 NAND Flash 软件仿真模块的设计与实现, 并通过移植 u-boot 及 ARM Linux 内核到仿真器中, 验证上述性质是否满足, 以保证软件仿真模块设计的正确性。

关键词: NAND Flash; 仿真; 线性时态逻辑; 验证

中图分类号: TP319

文献标识码: A

文章编号: 1000-7180(2011)05-0032-04

A Verification Method of NAND Flash Simulation Module

WANG Yang¹, YANG Guo-wu², ZHENG De-sheng², LI Xiao-yu²

(1 School of Software, University of Electronic Science & Technology of China, Chengdu 611731, China;

2 School of Computer Science & Engineering,

University of Electronic Science & Technology of China, Chengdu 611731, China)

Abstract: According to the working theory and process of NAND Flash device, this paper describes the properties of the reading or writing state of the hardware device by using LTL language, and analyzes the design and implementation of NAND Flash device simulation module. Then transplants u-boot and Linux kernel to simulator and verify the properties above, which proves that the design of the simulation module is correct.

Key words: NAND Flash; simulation; Linear Temporal Logic (LTL); verification

1 引言

嵌入式技术的迅速发展,使之成为当今的一门热门学科. 嵌入式开发板作为学习嵌入式软件开发的必需品,由于其高昂的价格及复杂的调试过程,使得开发板的应用范围有限,于是软件仿真的开发平台在嵌入式开发中也有着较多的应用. 目前国内外都出现了相关的嵌入式开发板仿真的产品,其中的代表为 Skyeeye^[1] 和 ARMulator^[2]. 然而软件模拟的开发板毕竟与真实的硬件存在较大的差异,在投入使用前必须经过严格的测试和验证. 文中即针对仿真器的一个模块: NAND Flash 的 K9F1208U0B 型号的仿真实现与规格进行简单描述,并提出一种验证方法,即通过抽取验证公式和模型并移植 u-boot、Linux 内核进行验证.

2 NAND 存储结构与 NAND Flash

NAND 存储结构是在 1989 年由东芝公司提出的. 由于 NAND 结构存储空间利用率高,存储量大,并且写入和擦除的速度也很快,量产后得到了迅猛的发展,成为嵌入式产品中极其重要的组成部分. NAND Flash 是采用 NAND 结构技术的非易失存储器,比起同类产品 NOR flash 具有容量大,写入和擦除速度快等优点. 然而 NAND Flash 都存在位反转的问题,即存储空间内部某个比特值由 0 变为 1 或由 1 变为 0,可能会导致严重后果,故安全性不及 NOR flash.

K9F1208U0B 是 Samsung 公司生产的采用 NAND 技术的大容量、高可靠 flash 存储器. 存储容量为 64M×8 位,每页容量为 512 B,此外芯片给每

收稿日期: 2010-07-12; 修回日期: 2010-10-09

基金项目: 2009 年国家“核高基”重大专项(2009ZX01039)

一页还提供 16 B 大小的附加区,用于 NAND Flash 命令执行完后设置状态用^[3]. Flash 提供 8 个 I/O 复用引脚,用于传输命令,地址或数据,由地址锁存端 ALE 和命令锁存端 CLE 引脚来控制当前传输的是哪一项,这样可以降低接口电路的复杂度^[4-6].

验证软件仿真的 NAND Flash 设备模块的过程就是通过运行 NAND Flash 模块中存储的操作系统或应用程序来检测是否可以正确读写 NAND Flash 的寄存器,设置控制引脚状态,以及对存储区或附加区进行读、写、擦除操作时是否得到的正确的结果.

3 设计仿真结构

NAND Flash 模块被设计为对外接口、芯片接口、内部实现及 DUMP 文件四部分,如图 1 所示.

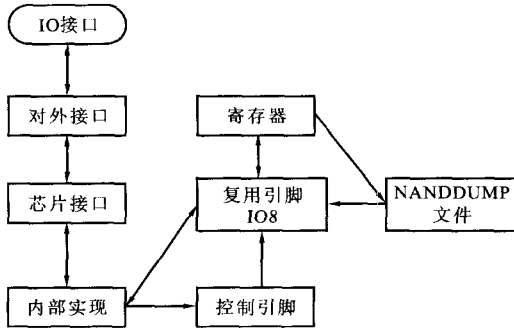


图1 NAND Flash 设备模块结构

① 模块对外接口:将 NAND Flash 设备模块组织成一个设备描述表由核心模块统一管理,负责安装、卸载 NAND Flash 模块,设置存储空间的大小和分页分块情况,将来自 CPU 的读写命令或需要写入的数据格式化并调用具体芯片接口的读写函数;

② 模块芯片接口:负责设置 NAND Flash 模块的寄存器,实现对寄存器的读写操作;

③ 模块内部实现:负责改变模拟的引脚状态,命令、地址的解析及 NANDDUMP 文件的内存映射和读写等操作;

④ NANDDUMP 文件:负责保存烧写到 NAND Flash 中的数据,文件大小与 NAND Flash 的存储空间一致.工作时将文件映射到内存中的一段区域中去.

NAND Flash 模块的模拟程度达到了指令级,故其工作流程与真实的 NAND Flash 设备一致.

4 规格描述

文中选择用线性时态逻辑对 NAND Flash 仿

真模块的性质和工作时的行为进行规格描述,抽取验证公式和模型用于接下来的工作.

4.1 线性时态逻辑

线性时态逻辑(LTL)规定了用来指示未来状态的连接词,可以用来对有限状态序列进行描述和建模^[6].下面是用BNF范式给出的语法:

$$\Phi ::= p | (\neg \Phi) | (\Phi \vee \Phi) | (\Phi \wedge \Phi) | (\Phi \rightarrow \Phi) | (X\Phi) | (F\Phi) | (G\Phi) | (\Phi U \Phi) | (\Phi R \Phi) | (\Phi W \Phi)$$

其中 p 是取自某原子集的任何命题原子; $\neg, \vee, \wedge$ 表示逻辑非、逻辑或、逻辑与; X, F, G, U, R, W 是时态连接词,表示与时间有关的一些特性. X, F, G 表示“下一个状态”、“某未来状态”、“所以未来状态”, U, R, W 表示“直到”、“释放”、“弱一直到”^[7].

4.2 仿真模块的规格描述

在对 NAND Flash 模块进行验证之前,需要根据 NAND Flash 模块的组织结构和引脚功能,将 NAND Flash 的功能抽取出一个模型来,如图 2 所示.

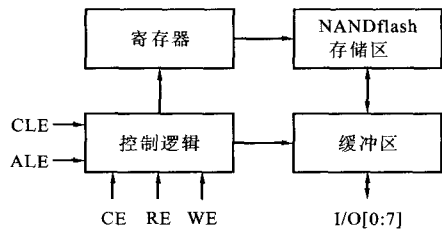


图2 NAND Flash 引脚结构

NAND Flash 作为一个存储设备,其主要功能就是保存数据并提供读写操作(擦除操作也可视为写操作).由于仿真的 NAND Flash 创建一个 NANDDUMP 文件用来存储数据,对 NAND Flash 的读写即为对该文件的读写.如果每次在地址 A 处写入数据 D 之后,再通过读指令从地址 A 读出来的数据亦为 D,则可以证明模拟的 NAND Flash 功能的正确性.该模型的性质可用 LTL 公式描述如下:

① 写模式:

$$RE \wedge \neg WE \wedge X(WE) \wedge \neg CE \wedge ALE \wedge \neg CLE \wedge I/O[7:0] * 4 = A_1 \quad (1)$$

$$RE \wedge \neg WE \wedge X(WE) \wedge \neg CE \wedge \neg ALE \wedge CLE \wedge I/O[7:0] = D_1 \quad (2)$$

$$RE \wedge \neg WE \wedge X(WE) \rightarrow NAND[A_1] = D_1 \quad (3)$$

② 读模式:

$$RE \wedge \neg WE \wedge X(WE) \wedge \neg CE \wedge ALE \wedge \neg CLE \wedge I/O[7:0] * 4 = A_2 \quad (4)$$

$$RE \wedge X(\neg RE) \wedge WE \rightarrow D_2 = NAND[A_2] \quad (5)$$

其中 RE 和 WE 为读写使能端,分别用来控制

NAND Flash 的读写动作,当 WE 由低电平跳到高电平时,执行一个周期的写寄存器或连续的写入存储区的动作,当 RE 由高电平跳到低电平时,执行一个周期的读寄存器或存储区内容的动作;CE 为芯片使能端,为低电平时芯片才可工作;ALE 和 CLE 即为命令锁存端和地址锁存端,ALE 为低电平且 CLE 为高电平时,从 I/O[7:0]读命令并放到命令寄存器中;ALE 为高电平且 CLE 为低电平时,从 I/O 读地址并放到地址寄存器中,注意读地址需要四个周期;ALE 为低电平且 CLE 也为低电平时,从 I/O 读数据并存到 NAND Flash 的存储区中. NAND $[A_1]$ 表示地址 NAND Flash 存储区中地址 A_1 存放的数据. 如果对任意 $A_1 = A_2$, 都有 $D_1 = D_2$, 则 NAND Flash 模块满足硬件的性质.

5 仿真模块的验证

验证工作在开发板仿真器上进行. 修改仿真器配置,模拟基于 S3C2410 的开发板,并将实验用不到的外设去掉,只保留 K9F1208U0B 型号 NAND Flash,term 类型串口即可,并为接下来的验证工作做准备.

移植 u-boot、Linux 内核和根文件系统到模拟的开发板中,经过安装交叉编译环境、修改源码、编译等工作后得到 u-boot. bin, uImage, root. cramfs 三个文件. 根据 Linux 内核将整个 NAND Flash 存储分区的划分情况,利用仿真器附加的 NANDDUMP 文件制作工具 DUMPMAKER,将 u-boot. bin 文件烧写到从地址 0x0 位置开始的 NANDDUMP 文件区域中去,uImage 从 0x100000 位置开始,root. cramfs 从 0x400000 位置开始. 该过程与使用 JTAG 烧写硬件开发板中的 NAND Flash 类似,相当于执行了模型中写模式的式(1)~(3). NAND Flash 的硬件烧写和仿真模块的烧写过程如图 3 和图 4 所示.

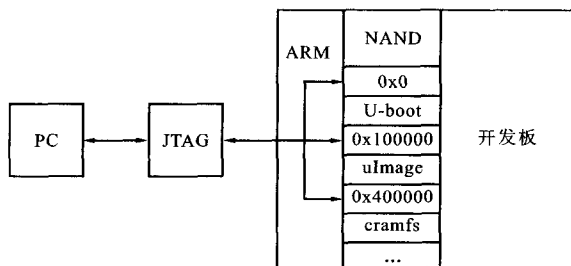


图 3 NAND Flash 硬件的烧写

将 u-boot. bin 与 NANDDUMP 文件拷贝到仿真器目录下,启动仿真器运行目标型号开发板并

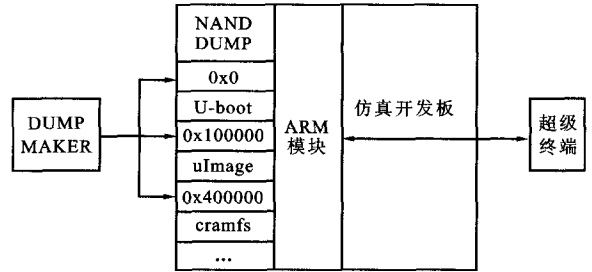


图 4 NAND Flash 仿真模块的烧写

开始运行,u-boot 运行结果见图 5.

```
/uart_console.exe
U-Boot 1.2.0(May 11 2010 - 22:01:44)

DRAM: 64 MB
Flash: 16 MB
NAND: 64 MB
***Warning -bad CRC,using default environment

In: serial
Out: serial
Err: serial
SMDK2410 # nand read 0x30008000 0x 100000 0xee2c0

NAND read:device 0 offset 1048576,size975552...
975552bytes read:OK
```

图 5 u-boot 运行结果

5.1 规格描述的指令验证

进入 u-boot 后,可输入命令 help 查看 u-boot 所支持的命令. 其中的 nand 命令集是在制作 u-boot 时加入的. 目前的有效命令只有 nand read. 但这对于 u-boot 的功能来说已经足够,执行读入命令:

nand read 0x30008000 0x100000 0xee2c0

nand read 的三个参数:读到 RAM 中的起始地址,dump 文件中的偏移地址,读取内容的长度(即 uImage 文件大小).

此时 NAND Flash 模块执行一系列的修改引脚电平值变量的指令,相当于执行了模型中读模式的式(4)、(5). 同时在 SetRE 函数中调用读取 NANDDUMP 文件内容的函数,这样 Linux 内核镜像被读入到 RAM,可以使用仿真器的调试功能的 md 命令查看指定内存地址的内容:

该命令按字读取,每次同时显示 4 个地址里的 4 字节的内容,测试 uImage 存在 RAM 的起始位置的两组数据结果见表 1.

表 1 md 命令得到的 RAM 地址中的内容

RAM 地址	DUMP 地址	数据
0x30008000	0x100000	0x56190527
0x30008004	0x100004	0x294e4674

这些数据的来源是 uImage 文件,对比用 Ultra-

Edit 文本查看器查看 uImage 文件中 0x00—0x07 的内容如表 2 所示.

表 2 用 UltraEdit 读取的 uImage 文件的内容

首地址	0	1	2	3
0x0	27	05	19	56
0x4	74	46	4E	29

由于仿真器模拟的开发板为小端存储模式,故低址存放低位的数据,所以两者结果一致,仿真模型的性质公式成立.

5.2 规格描述的系统验证

为了进一步验证 NAND Flash 模块是否可以模拟真实的 NAND Flash 硬件工作,接下来加载 ARM Linux 内核,查看其是否可以正常启动. 在超级终端中输入启动 Linux 内核命令:

```
bootm 0x30008000
```

从超级终端显示的启动信息可知 Linux 内核在仿真器中启动了. 由于 bootm 启动时有解压缩的过程,而且还需要读取 NANDDUMP 文件中根文件系统所在位置的内容,所以消耗时间较长. 进入文件系统后,可以使用 cd,ls 等 Linux 命令来进行操作. Linux 内核的运行结果如图 6 所示.

```
/uart_console.exe
Please press Enter to activate this console.
Starting pid 18.console/dev/tts/0: 'bin/sh'

BusyBox v1.1.3(2010.05.10-07:56+0000)Built-in shell(ash)
Enter 'help' for a list of built-in commands.

-sh:echo:not found
-sh:echo:not found
#ls
bin      etc      lib      linuxrc- proc    sbin     trnp
dev      home    linuxrc  mnt      root    sys      usr
#cd bin
#ls
addgroup busybox delgroup ls      mv      sh      touch
adduser  cat     deluser  mkdir  pwd     su      umount
ash      cp      login   mount  rm      tinyloginvi
#
```

图 6 Linux 内核运行结果

这样就实现了从仿真的 NAND Flash 中启动 u—boot 并成功引导挂载 cramfs 根文件系统的 Linux 启动,并且与使用真实的基于 S3C2410 的硬件开发板进行本次实验时得到的结果一致,说明仿真的 NAND Flash 模块正确执行了所有读写操作,因此满足模型的性质公式,其正确性得到了验证.

6 结束语

文中通过抽象仿真的 NAND Flash 的模型和移植运行 u—boot 与 Linux 内核,证明了软件仿真的 NAND Flash 设备模块可以正常运行并支持其他设备模块工作,由 NAND Flash 读取的应用程序的运行过程与结果也和真实的硬件设备没有区别. 软件仿真的 NAND Flash 的读取、烧写等工作是用文件的读写来实现,故速度极快,且不存在位反转问题等诸多优势,大大增加了嵌入式开发的方便性与安全性,同时也降低了成本,提高开发速度,在以后嵌入式领域中会有更为广泛的应用.

参考文献:

[1] Kang M. Skyeye user's manual[EB/OL]. (2005—03—08)[2010—07—02]. <http://skyeye.sourceforge.net/wiki/UserManual>.
[2] uClinux in the GDB/ARMulator [EB/OL]. (2007)[2010—07—02]. <http://www.uclinux.org/pub/uClinux/utilities/armulator/>
[3] 梁宏明,胡东伟,陈杰. SOC 中 NAND Flash 控制器的一种新型结构和实现方法研究[J]. 微电子学与计算机, 2009,26(6):89—92.
[4] Eames B, Neema S, Bapty T, et al. Interfacing a simulation engine to an embedded runtime environment[J]. IEEE International Conference, 2000(1):625—629.
[5] SAMSUNG Electronics Corporation. K9F1208U0B Product Manual[EB/OL]. (2003)[2010—07—02]. <http://www.samsung.Com/Products/Semiconductor>.
[6] Huth M, Ryan M. 面向计算机科学的数理逻辑系统建模与推理[M]. 北京:机械工业出版社,2007.
[7] 刘建元. 基于符号模型检验的硬件验证[J]. 微电子学与计算机,2002,19(5):62—64.

作者简介:

王 洋 男,(1985—),硕士研究生. 研究方向为嵌入式开发.
杨国武 男,(1966—),博士生导师. 研究方向为逻辑综合、形式化验证.
郑德生 男,(1983—),博士研究生. 研究方向为形式化验证、模型检测.
李晓瑜 女,(1984—),博士研究生. 研究方向为 DSP 技术、量子计算.