第 35 卷 第 7 期 计 算 机 工 程 2009 年 4 月
Vol.35 No.7 Computer Engineering April 2009
·工程应用技术与实现· 文章编号：1000—3428(2009)07—0232—03 文献标识码：A 中图分类号：TP3

WinCE 5.0 Bootloader 的设计与实现

张 飞，白瑞林，陆 林
(江南大学智能控制研究所，无锡 214122)

摘 要：提出一种基于 ARM9 的 WinCE 5.0 引导程序(Bootloader)设计与实现方法。通过分析 Bootloader 架构，在研究触摸屏控制器的 WinCE 5.0 BSP 开发基础上，设计并实现一种基于 AT91SAM9261 的 WinCE 5.0 Bootloader。阐述其启动代码和主代码的开发过程，并给出主代码开发中硬件初始化的步骤。测试结果表明，该 Bootloader 达到了设计目标并已应用于触摸屏控制器。

关键词：触摸屏控制器；WinCE 5.0 操作系统；AT91SAM9261 处理器；引导程序

Design and Implementation of WinCE 5.0 Bootloader

ZHANG Fei, BAI Rui-lin, LU Lin
(Institute of Intelligent Control, Jiangnan University, Wuxi 214122)

【Abstract】This paper proposes a design and implementation of WinCE 5.0 Bootloader based on ARM9. By analyzing the structure of Bootloader, based on the development of WinCE 5.0 BSP of touch-screen controller, it designs and implements WinCE 5.0 Bootloader based on AT91SAM9261. It describes the development process of boot code and main code, and gives the detailed hardware initialization steps of main code. Test result indicates that the Bootloader meets the design objectives and is used in touch-screen controller.

【Key words】touch-screen controller; WinCE 5.0 operation system; AT91SAM9261 processor; Bootloader

在目前的智能终端市场，将 ARM9 和 WinCE 结合正成为一种趋势。ARM9 处理器因其卓越的性能和显著的优点，已经占据了绝大部分 32 位高端嵌入式处理器市场。WinCE 5. 0 是一个开放的、可裁剪的 32 位嵌入式实时操作系统。它能提供优秀的画面效果和类似于桌面 Windows 的交互界面，方便用户的操作和使用[1]。另外它还提供了完善的设备驱动程序开发环境和软件开发包，能很快地开发设备驱动程序，缩短开发周期，加快产品的上市时间。

1 系统原理

本文的研究源于触摸屏控制器，其硬件结构如图 1 所示。该触摸屏控制器采用 Atmel AT91SAM9261 微控理器(ARM9)。硬件包括：64 MB Nand Flash，64 MB SDRAM，4 MB 的 Nor Flash，LCD 接口，USB Host 和 Device 接口，100 Mb/s 网络接口，串口等。

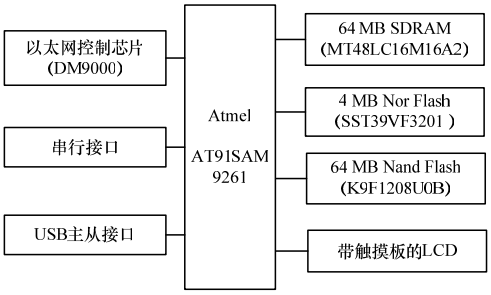

图 1 基于 AT91SAM9261 的触摸屏控制器硬件结构

Atmel 公司对 AT91SAM9261 没有提供 WinCE 5.0 的 BSP 支持，所以需自主开发。其中，Bootloader 在 BSP 的开发中至关重要。Nor Flash 用来存放 Bootloader，加载操作系统映像至内存。Nand Flash 和 SDRAM 提供了足够大的空间来存放和运行操作系统映像。

2 Bootloader的架构

WinCE 5.0 BSP 主要由 4 个部分组成：OEM 抽象层，Bootloader，设备驱动程序和配置文件。开发 Bootloader 是进行 BSP 开发的关键一步。Bootloader 作为嵌入式系统软件的最底层，是硬件上电后运行的第 1 个程序，主要功能是初始化硬件，加载操作系统映像到内存，然后跳转到操作系统代码去执行[2]。Bootloader 由 BLCOMMON、OEM 代码、Eboot 以及网络驱动组成，相应的 Bootloader 架构如图 2 所示。

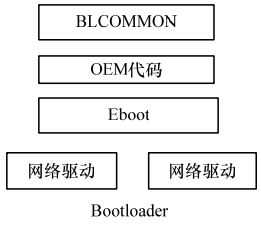

图 2 Bootloader 的架构

BLCOMMON 是通用的 Bootloader 框架，提供了丰富的 Bootloader 基本功能，包括复制 Bootloader 到 RAM 以便高速运行、解析.bin 文件、控制系统加载过程等。

OEM 代码是与目标硬件相关的初始化和扩展程序。

Eboot 是以太网功能程序，如 UDP, DHCP, TFTP 程序等。

网络驱动是调试以太网的驱动程序，提供了初始化和控制网络设备的功能。

BLCOMMON, Eboot 和网络驱动是可以被重用和可移植的代码库，这部分代码与 OEM 代码进行联编构成最终的

作者简介：张 飞(1984－)，男，硕士研究生，主研方向：嵌入式系统；白瑞林，教授；陆 林，硕士研究生

收稿日期：2008-11-01 **E-mail：**zhangfei236600@yahoo.com.cn

Bootloader 二进制映像。

3 Bootloader的设计

为一个硬件平台设计 Bootloader 时，首先要考虑需实现的功能。本文设计的 Bootloader 主要实现以下功能：

(1)Bootloader 驻留在非易失存储设备中(Nor Flash)；

(2)提供一种消息反馈机制，在输出信息中，明确显示 Bootloader 的版本、开发者及构建日期等信息；

(3)利用以太网口下载操作系统映像；

(4)设计一个序列化的功能选择菜单，允许开发者或用户对使用动态 IP 还是静态 IP 等进行选择和设置，将这些信息永久保存在 Flash 中，避免每次 Bootloader 运行时重新设置；

(5)在设定时间内没有选择任何选项时，Bootloader 执行自动下载功能；

(6)能够加载 Bootloader 映像和操作系统运行时映像；

(7)Bootloader 能对下载的数据进行校验，在保证下载的数据准确无误后，再将其写入 Flash 存储器[3]；

(8)提供从 Platform Builder 下载自身,并将自身写入 Flash 存储器，实现自我更新的机制。

4 Bootloader的实现

Bootloader 包括启动代码和主代码。启动代码 Startup 是 Bootloader 的入口函数，在 CPU 启动后将立即运行。该函数使用汇编语言编写，负责初始化 CPU 和一些核心的逻辑部件，然后跳到主代码中执行。主代码用 C 语言编写，实现 OEM 平台的初始化、映像下载、调试串口以及写 FLASH 代码的实现。Bootloader 的工作流程和函数调用如图 3 所示。

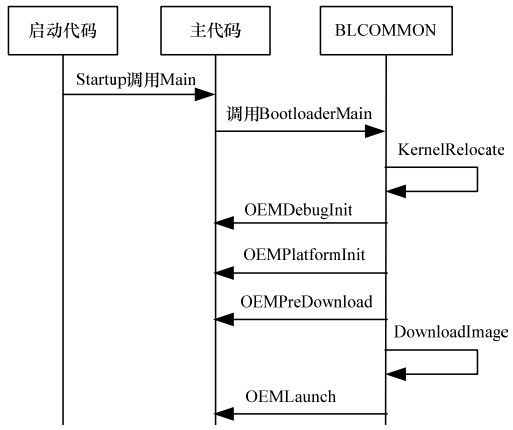

图 3 Bootloader 的工作流程和函数调用

4.1 启动代码的实现

代码的执行流程如图 4 所示。

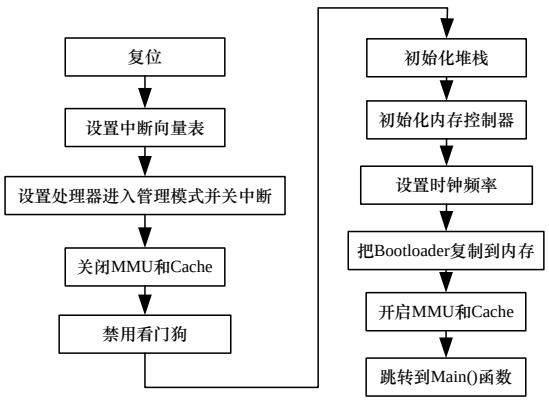

图 4 启动代码流程

启动代码主要完成的任务包括：使 CPU 进入正确的运行模式；在 CPU 级别关闭所有中断；关闭 MMU 和 Cache；初始化内存控制器；初始化其他硬件设备，如时钟等；设置栈指针；设置并打开 MMU 进行逻辑和物理地址映射；打开 Cache；将 Bootloader 本身复制到内存中；跳转到 C 语言的 Main()函数。

4.2 主代码的实现

主代码主要调用 BLCOMMON 中的 BootloaderMain()函数，主要任务是调用以下函数：

(1)重定位全局变量函数 KernelRelocate()，它将 Bootloader 中的全局变量重定位到 RAM 中。

(2)初始化调试端口函数 OEMDebugInit()，主要任务是初始化调试输出用的串口，方便输出调试信息，包括串口端口的初始化、波特率的设定、数据的发送和接收。串口调试是嵌入式平台调试的主要手段。通过串口和超级终端，开发人员和目标平台建立交互，得到平台的调试信息。

(3)初始化平台函数 OEMPlatformInit()，其作用是初始化目标板上的设备，如 LCD、Flash、SDRAM、DM9000 网卡、实时时钟等，并初始化 Bootloader 的配置信息。接收用户输入和显示功能选择菜单也是在该函数中实现。

(4)预下载函数 OEMPreDownload()，主要任务是完成以太网下载前的一些准备工作,包括通过 DHCP 获得 IP 地址及初始化 TFTP 服务等。

(5)下载映像函数 DownloadImage()，该函数完成从远程开发机上下载操作系统映像。

(6)启动映像函数 OEMLaunch()，把下载的映像写入 NAND Flash，调用函数获得 Platform Builder 的配置信息，并写入 Nor Flash，然后跳转到操作系统映像。

其中，OEMDebugInit(), OEMPlatformInit(), OEMPreDownload(), EMLaunch()等 OEM 函数是开发者必须实现的。这些函数与硬件密切相关,BLCOMMON 库会直接调用它们，如果不实现这些函数，Eboot 的链接就会失败。本文所开发的 Bootloader 的 OEMPlatformInit()如下：

```
BOOL OEMPlatformInit(void)
{…
SST39VF3201_Init();
AT91F_InitSDRAM();
InitDisplay();
InitRealTimeClock();
InitDM9000();
…
  if (!FMD_Init(NULL, &NANDInfo, NULL))
    {
        OALMSG(OAL_WARN, (TEXT("WARNING: OEMPlatformInit: Failed to initialize Smart Media.\r\n")));
        g_bSmartMediaExist = FALSE;
    }
    else
    {
        g_bSmartMediaExist = TRUE;
}
…
}
```

以上一段程序完成了对本触摸屏控制器的 LCD、Nor

Flash(SST39VF3201)、Nand Flash(K9F1208U0B)、SDRAM (MT48LC16M16A2)、DM9000、实时时钟的初始化。要实现这些函数，必须对硬件有充分的了解。

4.3 共享缓存区的建立

共享缓存区存放了 Bootloader 和操作系统共享的信息，它包括 Driver globals 和 Boot args 两部分[4]。Driver globals 定义的是一些设备配置。Boot args 定义的是与启动过程相关的信息，比如目标设备的 MAC 地址、IP 地址以及其他系统或用户设置等信息。其中 Driver globals 必须要在一个 RAM 页面里。另外，开发者还必须在.bib 文件中为 Driver globals 预留一定的内存。

4.4 SOURCES文件的编写

SOURCES 文件定义了代码的编译方法。编译程序通过 SOURCES 文件决定如何对代码进行编译和链接。SOURCES 文件主要包括如下几个宏:

(1)INCLUDES，包含头文件的查找目录列表。

(2)TARGETNAME，编译代码最终生成的文件名。

(3)TARGETTYPE，编译代码最终生成的文件类型，如生成 LIB 静态库文件、EXE 可执行文件等。

(4)TARGETLIBS，要动态链接到目标文件中的库文件列表。

(5)SOURCELIBS，要静态链接到目标文件中的库文件列表。

(6)SOURCES，要被编译和链接的源文件列表。

(7)EXEENTRY，指定生成 EXE 文件的入口函数。

Source.cmn 是一个特殊的 SOURCES 文件,它是对应 BSP 通用的 SOURCES。相应 BSP 里的所有 SOURCES 都应用了 Source.cmn 的内容，开发者需要在这个文件里声明全局宏。

4.5 Boot.bib文件的编写

BIB 文件的作用是指示构建系统如何构建二进制映像。BIB 文件会指示哪些文件被打包到运行时映像中，同时还会指示如何往目标板上加载模块和文件。Boot.bib 文件定义了 Bootloader 的最终结构，它主要包含如下几部分内容:

(1)MEM0RY，定义可用的物理内存，包括起始地址、大小和内存类型。

(2)C0NFIG，定义了一些附加选项来定制输出。

(3)FILES，指定放在运行时映像中的文件列表。

(4)MODULES，指定放在运行时映像中的模块列表。

4.6 Makefile和Dirs文件的编写

Makefile 文件只需包含 WinCE 的通用 Makefile。Dirs 文件定义了对应文件夹的子目录，编译器通过 Dirs 文件来确定要编译的文件位置。

5 Bootloader的编译与测试

5.1 Bootloader的编译

本文采用命令行方式编译 Bootloader。用命令行编译 Bootloader 可以分为 3 部分:

(1)设置命令行参数。这一步要通过运行 Wince.bat 来确定目标板的 CPU 和 Bootloader 对应的 BSP，为特定的操作系统设计配置好所有的命令行环境变量。

(2)运行 sysgen。这一步将创建 Bootloader 需要的静态库。

(3)编译 Bootloader 代码。编译最终将得到 eboot.nb0 和 eboot.bin 这 2 个 Bootloader 映像文件。

5.2 Bootloader的测试

Bootloader 是系统启动后第一个运行的程序，因此它必须放在 CPU 上电和复位后立即运行的地址处。AT91SAM9261 在上电和复位后是从物理地址 0x00000000(片选 0，Nor Flash 的起始地址)开始运行的。用 FLASH 下载工具将 eboot.nb0 下载到 Nor Flash，通过串口连接开发板和上位机，打开超级终端，上电复位后，上面会显示 Bootloader 的版本号等信息和功能选择菜单，可以选择设置配置参数、下载或启动已经存在的操作系统映像等功能。在上位机上用 WinCE 操作系统定制工具 Platform Builder 定制一个操作系统映像，设置连接参数，即可通过网线将操作系统映像下载至目标板。从超级终端上显示的调试信息可以看出 Bootloader 的工作状况，如图 5 所示。

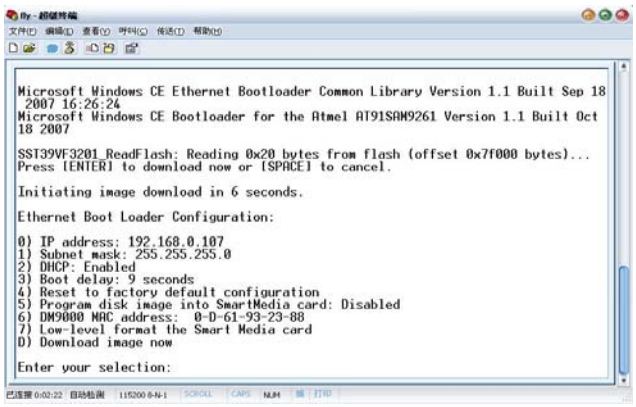

图 5 Bootloader 的功能选择菜单

6 结束语

本文致力于研究 WinCE 5.0 在基于 AT91SAM9261 的触摸屏控制器上的移植。BSP 的开发在 WinCE 的移植中必不可少，而 Bootloader 的开发是其中关键的环节。目前，基于 AT91SAM9261 的 WinCE 5.0 Bootloader 已开发成功并通过调试，在此基础上开发的 WinCE 5.0 BSP 也已应用于触摸屏控制器。本文的研究成果对于基于不同硬件的 WinCE Bootloader 开发也具有一定的参考价值。

参考文献

[1] 何宗键. Windows CE 嵌入式系统[M]. 北京: 北京航空航天大学出版社, 2006.

[2] 马学文, 朱名日, 程小辉. 嵌入式系统中 Bootloader 的设计与实现[J]. 计算机工程, 2005, 31(7): 96-98.

[3] 邹 莹, 魏洪兴, 王田苗. 基于NAND Flash的PXA270 Eboot启动技术[J]. 计算机工程, 2007, 33(12): 277-279.

[4] 张冬泉, 谭南林, 王雪梅, 等. Windows CE 实用开发技术[M]. 北京: 电子工业出版社, 2006.

编辑 任吉慧