浙江科技学院学报,第 24 卷第 1 期,2012 年 2 月
Journal of Zhejiang University of Science and Technology
Vol. 24 No. 1, Feb. 2012

DOI: 10.3969/j.issn.1671-8798.2012.01.008

基于 Windows CE 的 BootLoader 架构设计与移植

陈　才,马连伟
(浙江科技学院 自动化与电气工程学院,杭州 310023)

摘　要: BootLoader 作为嵌入式系统设计关键性的步骤,在嵌入式系统设计中起着举足轻重的作用。针对微软嵌入式操作系统 Windows CE 的 Bootloader 设计,介绍了 Eboot 的软件整体架构和各模块的功能,分析了 Eboot 源代码级的启动运行流程;在此基础上,分析将 Eboot 成功移植到嵌入式系统所要做的工作和详细步骤;总结了移植过程中的难点和要注意的几个问题。

关键词: BootLoader;Windows CE;移植
中图分类号: TP316.7　　**文献标志码**: A　　**文章编号**: 1671-8798(2012)01-0035-04

Architecture design and transplantation of BootLoader based on Windows CE

CHEN Cai, MA Lian-wei
(School of Automation and Electrical Engineering, Zhejiang Universiy of Science and Technology, Hangzhou 310023, China)

Abstract: BootLoader is very important in the system design as the key step during the embeded system design. In order to design the Bootloader for the Windows CE, the framework as a whole and module function as depart for the Eboot software are presented. The running flow of the Eboot based on the source code is analyzed. The main works to do and steps to transplant the Eboot framework are explained in details. Also, some difficulties and problems are discussed during the transplant.

Key words: BootLoader; Windows CE; transplant

BootLoader 是嵌入式系统上电后启动运行的第一段代码,用于对嵌入式系统硬件设备的最底层操作,不但包括嵌入式系统中的 CPU、存储器、通信外设的初始化设置、嵌入式系统参数配置和人机接口实现,而且还包括对嵌入式操作系统的引导启动和下载固化等操作。因此,BootLoader 在嵌入式系统设计中起极其重要的作用,是整个嵌入式系统应用的基础。现代嵌入式设备如手机 Androd 系统、平板电脑 MacOS 系统等底层软件开发过程中,BootLoader 是其中不可缺少的环节,BootLoader 的设计是其中的难

收稿日期: 2011-07-18
基金项目: 浙江省自然科学基金资助项目(Y1110508);浙江省教育厅科研计划项目(Y201119421)
作者简介: 陈　才(1981—　),男,湖北省武汉人,讲师,硕士,主要从事嵌入式系统设计研究。

点和核心点。采用 Windows 操作系统的嵌入式设备，用于启动引导和加载操作系统的程序代码 BootLoader称为 Eboot。Eboot 程序框架代码的设计主要分为 BLCommon 代码、OEM 代码、Eboot 代码、存储管理代码和 EDBG 调试工具代码 5 个部分[1]。

BLCommon 代码实现了通用的 BootLoader 框架，它提供了与 Windows CE 内核访问的接口。BLCommon 实现的功能包括：将 BootLoader 自身的代码从 NAND 拷贝到 SDRAM 中，以加快程序运行速度；将 Windows CE 内核镜像文件 NK.bin 解码成可执行的指令码，并进行代码校验、代码下载工作；提供 OS 启动或加载进度条的图形界面显示；调用 OEM 代码实现硬件初始化等。BLCommon 代码被编译成 BLCommon.lib 静态库，供用户调用。

Eboot 代码实现了对 DHCP、TFTP、UDP 等网络通信协议的支持。Eboot 代码被编译为 Eboot.lib 静态库，是 Eboot 框架代码的数据传输的主要组成部分。

OEM 代码是 BootLoader 中直接对硬件进行操作的函数，这些函数通常与硬件密切相关，在 Window CE5.0 中这些函数一般以 OEM 开头。比如：对 Flash 进行读、写、编程、擦除操作的 OEM 函数，对以太网卡 CS9000 或 DM9000 进行操作的 OEM 函数等。

存储管理代码主要是完成操作系统启动过程中对 SDRAM、Cache、MMU 等硬件的初始化和对各种数据结构、堆、栈等进行内存分配。

EDBG 调试工具代码是方便调试跟踪 BootLoader 的代码，设计者可以通过这些代码输出 Bootloader 运行过程中的调试信息。

1 Windows CE BootLoader 的软件架构分析

Windows CE BootLoader 的设计又叫 BootLoader 的移值。一个功能齐全、结构清晰的 BootLoader 框架运行主要包括 2 个相对独立的阶段[2]，每个阶段都操作特定的硬件并完成特定的功能。Windows CE BootLoader 启动的前一个阶段叫作 Nboot，第二个阶段叫作 Eboot。

1.1 第一阶段 Nboot 主要完成的工作

1）初始化硬件设备，主要包括：PLL 设置、关闭中断、关闭 MMU、清空 TLB 等。

2）配置 SDRAM 控制的参数。

3）为第二阶段代码运行初始化一段 SDRAM 内存，内存大小 1M 左右。

4）拷贝第二阶段代码到 SDRAM 中。

5）设置好高级语言运行环境。

6）跳转到第二阶段程序入口地址运行。

Nboot 运行流程图如图 1 所示。

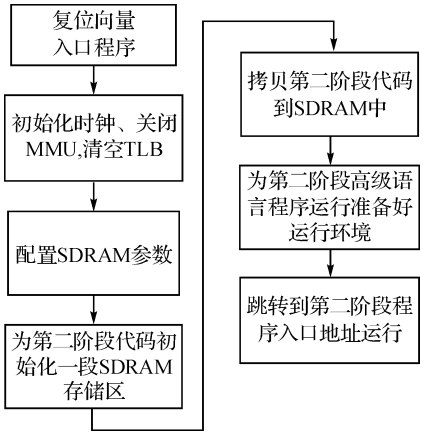

图 1 Nboot 阶段运行流程图

Fig. 1 Nboot stage run flow diagram

Nboot 完成的功能主要采用汇编语言编写和 C 语言混合编写。相关源代码文件及功能说明如下：

2440init.s	:	Nboot 阶段所用到的硬件初始化化	汇编语言编写
2440slib.s	:	Nboot 阶段所用到汇编库文件	汇编语言编写
2440Loader.c	:	Nboot 阶段的 main 函数文件	高级语言编写
2440.lib	:	Nboot 阶段的高级语言库文件	二级制目标代码
Nand.c	:	对 Nand flash 读写的高层函数	高级语言编写
Nand.s	:	对 Nand flash 读写的底层函数	汇编语言编写

Nboot 阶段的代码经过编译、连接后形成可执行文件为 Nboot.nb0。

1.2 第二阶段 Eboot 主要完成的工作

Eboot 又称为 Ethernet boot，是一种高级的 BootLoader，整个 Windows CE BootLoader 的大部分功能在这一阶段完成，Eboot 的运行流程图如图 2 所示。这一阶段完成的主要功能有：

1) 初始化 Eboot 阶段所要用到的硬件设备,主要包括:串口驱动、网卡驱动、液晶屏驱动等[3]。

2) 检测系统的内存映射。

3) 将 OS 映像从 FLASH 拷贝到 SDRAM 中。

4) 设置内核启动参数,并启动内核。

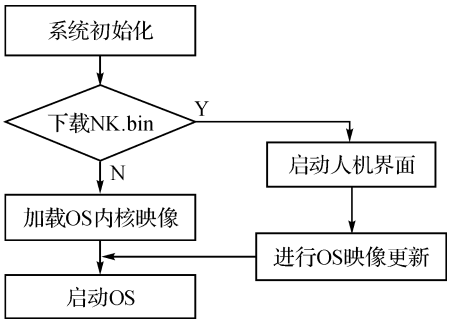

图 2　Eboot 运行流程图

Fig. 2　Eboot run flow diagram

2　EBOOT 部分源代码及相关函数的架构分析

EBOOT 是由 Main()函数进入的,其代码如下:

```
Main()
{
        BootloaderMain();
}
```

由代码可以看到 Eboot 阶段的 Main 函数的主要任务是调用系统的 BootLoaderMain()函数,该函数是微软提供的专门用来加载启动 Wince 的 OS 镜像的函数,其定义位于%WINCE500%\PUBLIC\COMMON\OAK\DRIVERS\ETHDBG\Blcommon.c 文件中。Eboot 的代码框架如图 3 所示。

在 BootLoaderMain()函数中,主要调用如下几个函数:

```
BootLoaderMain()
{
        KernelRelocated();
        OEMDebugInit();
        OEMPlatformInit();
        OEMPreDownLoad();
        DownLoadImage();
        OEMLaunch();
}
```

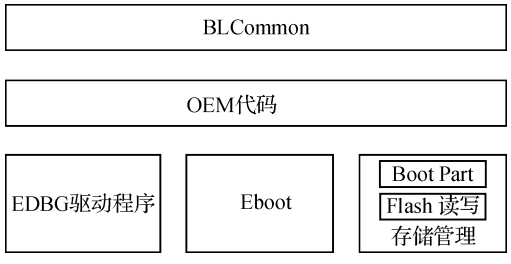

图 3　Eboot 的软件框架图

Fig. 3　Eboot software framework

各函数模块的主要功能:

1) KernelRelocated()　将 Eboot 中的全局变量全部拷贝到 SDRAM 中,确保 Eboot 中的全局变量能在 RAM 中读写。

2) OEMDebugInit()　初始化调试硬件接口,一般为串口,该函数将调用 OEMInitDebugSerial()函数来初始化调试串口,用于在调试过程中打印相关调试信息到终端。

3) OEMPlatformInit()　该函数全面初始化目标板上的硬件设备。需要根据实际电路板的硬件资源来进行配置,一般主要包括 LCD\RTC\以太网\串口(除调试串口外)等的配置工作。

4) OEMPreDownLoad()　完成以太网拷贝 OS 映像前的准备工作,主要包括获取和设定 IP 地址,初始化 DHTP\TFTP\UDP 服务等[4-5]。

5) DownLoadImage()　从远程开发主机端下载操作系统映像到嵌入式目标板的 FLASH 或 SDRAM 中。这个函数是一个比较大的函数。也是 Eboot 功能的主要实现模块。

6) OEMLaunch()　将 PC 指针直接指向 SDRAM 中 OS 映像的执行地址,启动 OS 映像的执行。它是 EBOOT 执行的最后一个函数,同时也是 OS 中 OAL.exe 模块运行的起点。

3　Windows CE BootLoader 的移植

在理解 BootLoder 的基本原理和 Windows CE BootLoader 架构分析的基础上,可以进行 Windows CE BootLoader 的移植工作。所谓“移植”就是将一定的软件框架运行到具体的硬件环境中。Windows CE BootLoader 移植过程实际是实现 BootLoader 中与目标系统硬件操作的相关函数的编写,以适应不同

的硬件环境。微软将这些直接操作硬件的函数称为 OEM 函数，函数名一般以 OEM 开头[5]。由于嵌入式系统硬件设备多种多样，每种设备的工作模式、配置、地址及功能选择都不相同，因此微软只是为这些操作起了一个函数名，而没有提供各种外设的 OEM 函数的源代码。这些函数的具体实现需要嵌入式设计人员根据具体的芯片选型和硬件电路设计来进行编写。即：微软只规定对哪些硬件需要做哪些操作，但没有提供具体操作方法和代码，这些由嵌入式设计人员根据实际硬件电路来具体完成。这种软件架构的主要思想是将软件中与硬件无关的部分和硬件相关的部分隔离开来。微软已经做好与硬件无关部分的软件设计，而与硬件相关部分的代码只留出了接口函数，用户使用时可根据具体的硬件来设计，这样的好处使得软件的重用性得到极大的提高。用户仅仅需实现 Eboot 中操作硬件有关的 OEM 代码。Eboot 中对硬件操作 OEM 函数分类为：控制程序运行的 OEM 函数、与系统调试相关的 OEM 函数、与下载 OS 相关的 OEM 函数、与 FLASH 烧写相关的 OEM 函数、与以太网相关的 OEM 函数及与时钟相关的 OEM 函数，见表 1。

表 1 OEM 函数分类

Table 1 Classfication of OEM function

OEM 函数分类	OEM 函数名	函数功能
控制程序运行 OEM 函数	OEMDebugInit()	调试初始化相关的 OEM 函数
	OEMPlatformInit()	平台初始化化相关的 OEM 函数
	OEMPreDownLoad()	下载前准备工作相关的 OEM 函数
	OEMLaunch()	OS 引导相关的 OEM 函数
与系统调试相关的 OEM 函数	OEMInitDebugSerial()	调试信息输出所使用串口初始化化
	OEMReadDebugByte()	调试时从串口读人数据，接收调试命令
	OEMWriteDebugByte()	调试时将数据串口输出，打印调试信息
	OEMWriteDebugString()	调试时写字符串到串口，打印调试信息
与下载 OS 相关的 OEM 函数	OEMMapMemAddr()	为 OS 映像开辟临时 RAM 缓冲区
	OEMReadData()	下载 OS 映像到临时 RAM 缓冲区
	OEMShowProcess()	在终端显示下载进度条信息
与 FLASH 烧写相关的 OEM 函数	OEMCheckSignature()	对 NK. BIN 文件进行校验
	OEMVerifyMemory()	检测下载内存地址是否合法
与以太网相关的 OEM 函数	OEMEthGetFrame ()	由以太网获取一帧数据
	OEMEthSendFrame()	用以太网上传一帧数据
与时钟相关的 OEM 函数	OEMEthGetSec ()	获得一段时间的长短(秒数)

在移植过程中，应根据具体采用的 FLASH、以太网、调试接口芯片和电路原理图来确定各个外设的物理地址和控制方式，然后依据相关的数据手册来实现以上的各个 OEM 函数。

4 结 语

在 Windows CE BootLoader 的移植过程中，要把握 2 个重点：一是需要对 ARM 体系结构和指令系统有比较深入的理解，熟练掌握 ARM 的硬件资源的使用；二是要掌握 Eboot 的软件架构，对一个软件体系的掌握和使用，关键是看对软件架构的理解。只有对这两方面都有比较深入的理解才能写出结构清晰、功能齐全、界面良好的 BootLoader 程序，为后续的 WindowsCE 操作系统移植打下一个良好的基础。

参考文献：

[1] 刘林真，刘大茂. 基于 WinCE. NET 的 BootLoader 原理与启动分析[J]. 计算机与数字工程，2008，36(10)：111-128.
[2] 何宗健. Windows CE 嵌入式系统[M]. 北京：北京航空航天大学出版社，2006：214-220.
[3] 王黎明，陈双桥，闫晓玲，等. ARM9 嵌入式系统开发与实践[M]. 北京：北京航空航天大学出版社，2008：407-453.
[4] 吴细宝，高梅国. 基于 WinCE 的嵌入式远程监控系统的设计与实现[J]. 仪器仪表学报，2008，129(4)：547-550.
[5] 王浩. Windows CE 嵌入式应用开发实训教程[M]. 北京：中国水利水电出版社，2010.