基于 WinCE 的 BootLoader 研究

吴杉杉，周东升

（安徽理工大学 计算机学院，安徽 淮南 232001）

摘　要：随着计算机技术的快速发展，嵌入式微控制器技术和嵌入式系统已成为当前的一大热点，而 BootLoader 则是嵌入式系统中重要的一部分。对 WinCE 下的 BootLoader 做了详细的研究。

关键词：WinCE；BootLoader

中图分类号：TP301　　**文献标识码**：A　　**文章编号**：1672-7800(2011)02-0044-03

1　BootLoader 的角色与功能

BootLoader 是在操作系统内核运行之前运行的一段小程序，它存放于目标平台的非易失存储介质中，如 ROM 或 Flash。它可以初始化硬件设备，建立内存空间的映射图，从而将系统的软硬件环境带到 1 个合适的状态，然后加载操作系统映像到内存，跳转到操作系统代码去执行。在开发 CE 的过程中，它主要用于启动硬件和下载 nk. bin 到目标板上，并有一定的监控作用。

下图描述了 WinCE 的 BSP 基本结构以及 BootLoader 所处的位置。一般来说，对于 BootLoader 的功能要求并不是严格定义的，不同的场合区别很大。比如，在 PC 的硬件平台上，由于硬件启动根本就不是通过 BootLoader（而是通过 BIOS），所以 BootLoader 就不需要对 CPU 加电后的初始化做任何工作. 通常，BootLoader 必须包含下载 CE 映像文件的功能。

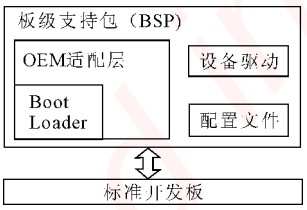

图 1　WinCE BSP 的基本结构

总体上 BootLoader 需要完成以下工作：

（1）初始化 CPU 速度。

（2）初始化内存，包括启用内存库，初始化内存配置寄存器等。

（3）初始化中断控制器，在系统启动时，关闭中断，关闭看门狗（Watchdog）。

（4）初始化串行端口（如果在目标上有的话）。

（5）启用指令/数据高速缓存。

（6）设置堆栈指针。

（7）设置参数区域并构造参数结构和标记。

（8）执行 POST（加电自检）来标识存在的设备并报告有何问题。

（9）为电源管理提供挂起/恢复支持。

（10）传输操作系统内核镜像文件到目标机。也可以将操作系统内核镜像文件事先在 FLASH 中，这样就不需要 BootLoader 和主机传输操作系统内核镜像文件，这通常是在做成品的情况下使用。而一般在开发过程中，为了调试内核的方便，不将操作系统内核镜像文件固化在 FLASH 中，这就需要主机和目标主机进行文件传输。

（11）跳转到内核的开始，在此又分为 ROM 启动和 RAM 启动。所谓 ROM 启动就是用 XIP 技术直接在 FLASH 中执行操作系统镜像文件；所谓 RAM 启动就是指把内核镜像从 FLASH 复制到 RAM 中，然后再将 PC 指针跳转到 RAM 中的操作系统启动地址。

下图展示了 Windows CE 的 BootLoader 的执行顺序。从系统加电自检开始，到 Windows CE 操作系统开始执行结束。

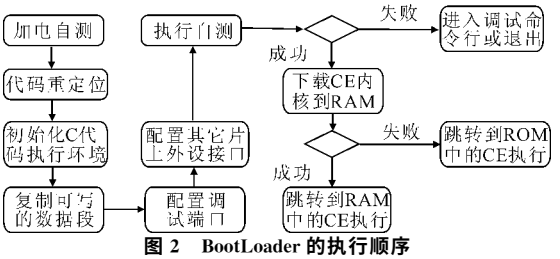

图 2　BootLoader 的执行顺序

2　BootLoader 的架构

WinCE 5. 0 BSP 主要由 4 个部分组成：OEM 抽象

作者简介：吴杉杉（1986－），女，安徽安庆人，安徽理工大学计算机学院硕士研究生，研究方向为嵌入式系统；周东升（1987－），男，安徽庐江人，安徽理工大学计算机学院硕士研究生，研究方向为嵌入式系统。

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，是硬件上电台运行的第 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。BootLoader 由 BLCOMMON，OEM 代码，Eboot 以及网络驱动组成，相应的 BootLoader 架构如图 3 所示。

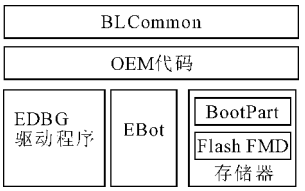

图 3　BootLoader 的架构

3　BootLoader 的设计

为一个硬件平台设计 BootLoader 时，首先要考虑实现的功能。本文的 BootLoader 主要实现以下功能：

(1)BootLoader 驻留在非易失存储设备中。

(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) 利用以太网下载操作系统映像。

(4)在设定时间内没有任何选项是，BootLoader 执行自动下载功能。

(5)BootLoader 能对下载的数据进行校验，在保证下载的数据准备无误后，再将其写入 FLASH 存储器。

(6)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 函数的移植

BootLoader 是由微软提供的库，Eboot 驱动和 OEM 函数构成。其中微软提供的库是不能修改的；Eboot 驱动可能需要结合硬件电路板的实际情况自己编写，如在一些板子上用 DM9000A 作为以太网控制芯片而有些则用 KSZ8841 替代；OEM 函数则必须由开发人员自己实现。接下来本文将具体阐述如何实现 OEM 函数。

对于 OEM 函数可分为两类：一类是必须实现的 OEM 函数，另外一类是可选的。

对于必须实现的 OEM 函数，BLCommon 库会直接地调用它们，因此如果不实现这些函数的话 Eboot 的链接就会失败。对于可选的 OEM 函数，BLCommon 库通常是通过指针去调用这些函数，在初始状态下，这些函数的指针为空，如果实现了这些函数，就可以给这些函数的指针赋值，在调用这些函数的时候，BLCommon 库会自动地检测这些函数指针；如果没有实现，则不会调用这些函数。

其中必须实现的 OEM 函数主要分为以下几类。

(1)初始化函数。

它们是在 Eboot 启动的时候执行的一些函数，主要负责初始化硬件平台。因此必须根据自己芯片的 datasheet 来改写 Stattup()函数中的相关操作。

(2)调试相关的函数。

这些函数主要负责调试端口输出的一些调试信息，以便跟踪代码的执行，同时可以用于人机交互。主要包括 OEMIinitDebugSerial( )，OEMReadDebugByte ( )，OEMWriteDebugByte( )和 OEMReadDebugString( )4 个函数。其中 OEMReadDebugByte( )和 OEMWriteDebugByte( )与硬件高度相关，但是实现并不困难。OEMReadDebugString( )通常可以由 OEMWriteDebugByte( )函数实现。

(3)时钟相关的函数。

在 Eboot 中时钟相关函数只有 OEMEthGetSecs( )，此函数通过时钟读取来返回某个时间过去的秒数。

(4)Flash 操作函数。

对于一般的 BootLoader 来说都会有一个功能，就是把映像烧写到 Flash 上，所以在移植 BootLoader 的时候必须实现 Flash 操作的相关函数。如读、写、擦除等一些基本操作。在实现这些函数的时候，具体的命令格式和读写操作流程需要参照与 Flash 型号配套的 Datasheet。

5　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 的初始化，映像下载，调试串口以及 FLASH 代码的实现。BootLoader 的工作流程和函数调用如下图所示：

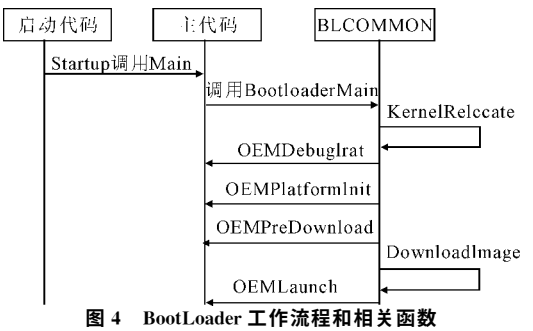

图 4　BootLoader 工作流程和相关函数

6　编写 BootLoader 源程序

前面已经提到，由于硬件的不同，BootLoader 的功能可能有多有少，此处笔者以自己开发 BootLoader 的过程进行叙述。图 5 是工作流程：

6.1　启动代码的实现

代码的执行流程如图 6 所示。

6.2　主代码的实现

主代码主要调用 BLCOMMON 中的 BootloaderMain

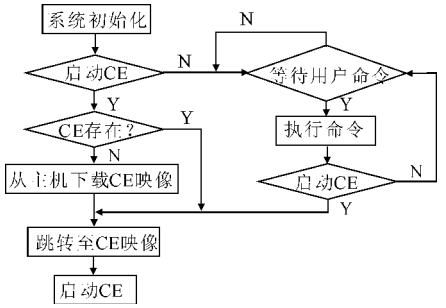

图 5　BootLoader 的工作流程

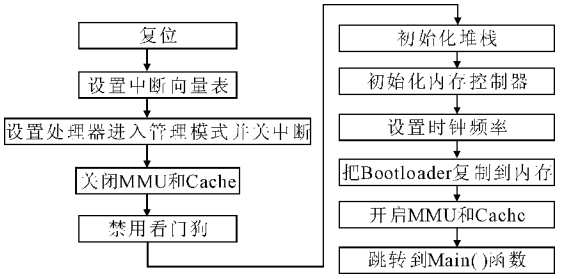

图 6　BootLoader 代码执行流程

()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面板，如 LCD，Flash，SDRAM，DM9000M 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，并定稿 Nor Flash，然后跳转到操作系统映像。

7　BootLoader 的编译，链接和下载

BootLoader 程序可以通过 PB 的集成编译环境编译链接，控制文件为。Bib 文件。下面是一个简单的 BootLoader 的。bib 文件。

```
MEMORY
CLI 9fc00000 00050000 RAMMIMAGE
    RAM 80080000 00070000 RAM
    CONFIG
      COMPRESSION=ON
      SRE=ON
      ROMSTART=9fc00000
      ROMSIZE=00020000
      ROMWIDTH=32
      ROMOFFSET=000000
    MODULES
      nk.exe $(_FALTRELEASEDIR)\cli.exe CLI
```

MEMORY 部分定义了生成的映像文件的目标地址，以及程序运行可以使用的内存空间。

CONFIG 部分：COMPRESSION 是否对目标代码进行压缩；SRE 是否生成格式为 sre 的目标；ROMSTART 与 ROMSIZE，ROMWIDTH，ROMOFFSET 共同定义了开发平台上存放 BootLoader 物理介质的起始地址、大小、宽度和偏移量。

MODULES 部分定义了 BootLoader 所包含的文件，一般就只有一个文件：cli.exe。

编译过程中，首先用命令 build－c 编译生成文件 cli.exe，然后用 romimage cli.bib 命令产生最后的映像文件 cli.sre。

对于 BootLoader 文件的下载；有很多种方法：可以通过仿真器下载；也可以通过其它调试程序下载；还可以直接烧写到 Flash 中。需要说明的一点是，这些方法可能会要求不同的映像格式。在 PB 环境下，可以生成的有.sre 格式，纯二进制格式以及和 CE 映像一样的。bin 格式。

8　结束语

本文详细讨论了 WinCE 下 BootLoader 的功能、架构及主要实现代码。经实验证明，此方法效果良好，对于学习 WinCE 下 BootLoader 有很好的指导作用。

参考文献：

[1] 张冬泉. Windows CE 实用开发技术[M]. 北京：电子工业出版社，2006.

[2] 田泽. 嵌入式系统开发与应用[M]. 北京：北京航空航天大学出版社，2005.

[3] 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(7).

[4] 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.

(责任编辑：余　晓)

嵌入式资源免费下载

总线协议：

1. 基于 PCIe 驱动程序的数据传输卡 DMA 传输
2. 基于 PCIe 总线协议的设备驱动开发
3. CANopen 协议介绍
4. 基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计
5. FPGA 实现 PCIe 总线 DMA 设计
6. PCI Express 协议实现与验证
7. VPX 总线技术及其实现
8. 基于 Xilinx FPGA 的 PCIE 接口实现
9. 基于 PCI 总线的 GPS 授时卡设计
10. 基于 CPCI 标准的 6U 信号处理平台的设计
11. USB30 电路保护
12. USB30 协议分析与框架设计
13. USB 30 中的 CRC 校验原理及实现
14. 基于 CPLD 的 UART 设计
15. IPMI 在 VPX 系统中的应用与设计

VxWorks：

1. 基于 VxWorks 的多任务程序设计
2. 基于 VxWorks 的数据采集存储装置设计
3. Flash 文件系统分析及其在 VxWorks 中的实现
4. VxWorks 多任务编程中的异常研究
5. VxWorks 应用技巧两例
6. 一种基于 VxWorks 的飞行仿真实时管理系统
7. 在 VxWorks 系统中使用 TrueType 字库
8. 基于 FreeType 的 VxWorks 中文显示方案
9. 基于 Tilcon 的 VxWorks 简单动画开发
10. 基于 Tilcon 的某武器显控系统界面设计
11. 基于 Tilcon 的综合导航信息处理装置界面设计
12. VxWorks 的内存配置和管理
13. 基于 VxWorks 系统的 PCI 配置与应用

14. 基于 MPC8270 的 VxWorks BSP 的移植

Linux：

1. Linux 程序设计第三版及源代码
2. NAND FLASH 文件系统的设计与实现
3. 多通道串行通信设备的 Linux 驱动程序实现
4. Zsh 开发指南-数组
5. 常用 GDB 命令中文速览
6. 嵌入式 C 进阶之道
7. Linux 串口编程实例

Windows CE：

1. Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计
2. Windows CE 的 CAN 总线驱动程序设计
3. 基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究
4. 基于 Windows CE.NET 平台的串行通信实现
5. 基于 Windows CE.NET 下的 GPRS 模块的研究与开发
6. win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码
7. Windows 下的 USB 设备驱动程序开发
8. WinCE 的大容量程控数据传输解决方案设计
9. WinCE6.0 安装开发详解
10. DOS 下仿 Windows 的自带计算器程序 C 源码
11. G726 局域网语音通话程序和源代码
12. WinCE 主板加载第三方驱动程序的方法
13. WinCE 下的注册表编辑程序和源代码
14. WinCE 串口通信源代码
15. WINCE 的 SD 卡程序[可实现读写的源码]

PowerPC：

1. Freescale MPC8536 开发板原理图

2. 基于 MPC8548E 的固件设计
3. 基于 MPC8548E 的嵌入式数据处理系统设计
4. 基于 PowerPC 嵌入式网络通信平台的实现
5. PowerPC 在车辆显控系统中的应用
6. 基于 PowerPC 的单板计算机的设计
7. 用 PowerPC860 实现 FPGA 配置

ARM：

1. 基于 DiskOnChip 2000 的驱动程序设计及应用
2. 基于 ARM 体系的 PC-104 总线设计
3. 基于 ARM 的嵌入式系统中断处理机制研究
4. 设计 ARM 的中断处理
5. 基于 ARM 的数据采集系统并行总线的驱动设计
6. S3C2410 下的 TFT LCD 驱动源码
7. STM32 SD 卡移植 FATFS 文件系统源码
8. STM32 ADC 多通道源码
9. ARM Linux 在 EP7312 上的移植

Hardware：

1. DSP 电源的典型设计
2. 高频脉冲电源设计
3. 电源的综合保护设计
4. 任意波形电源的设计
5. 高速 PCB 信号完整性分析及应用
6. DM642 高速图像采集系统的电磁干扰设计
7. 使用 COMExpress Nano 工控板实现 IP 调度设备