基于龙芯 1A 平台的 PMON 源码编译和启动分析

吴昌昊，官琴
（中国兵器工业第五八研究所特种电子技术部，四川 绵阳 621000）

摘要：为降低开发人员理解龙芯(MIPS32)和 PMON 源码的难度，对基于龙芯 1A 平台的 PMON 源码编译和启动进行分析。采用流程图和文字相结合的形式，对 PMON 的目录结构、编译方法和执行流程进行研究。结果表明：该方法能清晰地展示 PMON 的编译和启动时期的工作机理，可为相关人员了解 PMON 工作方式和修改编译其源代码提供帮助。

关键词：龙芯；MIPS；PMON；编译；启动
中图分类号：TP316.8　　**文献标志码**：A

PMON Source Code Compiling and Starting Analysis Based on Loongson 1A Platform

Wu Changhao, Guan Qin
(*Department of Special Electronic Technology, No. 58 Research Institute of China Ordnance Industries, Mianyang* 621000, *China*)

Abstract: For decreasing difficulties of researcher understanding Loongson(MIPS32) and PMON source codes, analyze the compiling and starting of PMON source code based on Loongson 1A platform. Combine flow charts with character, research content structure, compiling method and execution process of PMON. The results show that the method can give a clear illustration about PMON's operating principle in compile-time and start-time. This article may help new developers and anyone interested in PMON learn and modify the source code.

Keywords: Loongson; MIPS; PMON; compile; start

0　引言

基本输入输出系统(Basic Input/Output System，BIOS)，一般是指 x86 平台一种业界标准的固件接口，用于计算机开机时执行系统各部分的自检，并启动引导程序或装载在内存的操作系统[1]。

龙芯平台使用了名为 PMON(Prom Monitor)的程序作为 BIOS。PMON 具有强大的功能，包括硬件初始化、操作系统引导、硬件测试、程序调试等功能。同时它提供多种操作系统加载方式及多种硬件基础测试工具[2]。

然而，现阶段无论是在嵌入式、家用机、服务器或者是大型机的市场上，龙芯都属于小众，甚至可以说 MIPS 架构都很冷门，直接导致了龙芯平台相关硬件或软件项目参与的人员数量少，也就导致了相关文档资料很少。

为了能让更多人能更快地理解 PMON 源码，笔者针对龙芯 1A 平台的 PMON 代码的编译和启动过程进行分析。文中所使用的 PMON 是指龙芯中科针对龙芯 1A 发布的版本。

1　PMON 历史简介

PMON 原本是由 Phil Bunce 专为 LSI Logic MIPS R3000 评估板设计的一个免费 PROM 调试监控程序。从发布之日起，它就成为了许多 MIPS 平台开发系统的首选固件，但是原作者在 1999 年就停止更新了[3]。

后来出现的 PMON 2000 是由商业公司 Opsycon 开发的，改动巨大。该版本曾经也是最受欢迎的嵌入式系统固件之一[4]。从其官网可以发现，2008 年后 PMON 2000 也不再更新。

现在龙芯平台所用的 PMON 版本，是龙芯中科技术有限公司基于 PMON 2000 修改而来。

2　PMON 目录结构

PMON 的目录结构非常清晰，如表 1 所示。

表 1 中 tools 目录中包含多个工具，但实际上只需要用到里面的 pmoncfg，该程序是用来根据配置文件为 PMON 主程序的生成做准备工作。

最后一行的“zloader.xxx”是指存在于源码根目录中的软链接，比如“zloader.ls1a”。由于这些软链接都是指向同一个目录 zloader，所以使用起来并无差异，只是会影响 zloader 目录中文件的绝对路径值。创建这些软链接的目的是在编译时根据路径名使用不同的 Makefile。

收稿日期：2013-11-25；**修回日期**：2014-01-20
作者简介：吴昌昊(1990—)，男，四川人，本科，助理工程师，从事 Linux、VxWorks 软件开发和多核并行运算研究。

表 1　PMON 目录名称及作用

目录名	作用
conf	通用配置文件
doc	文档
examples	程序例子
fb	SiS 显卡驱动、图形 logo 代码
include	通用头文件
lib	库源码，包含 libc、libm、libz
pmon	PMON 主体源码
sys	系统内核基础代码、部分驱动代码
Targets	目标板级相关的配置文件、头文件、源文件、目标文件
tools	编译辅助工具
x86emu	x86 模拟程序源码
zloader	解压缩程序
zloader.xxx	指向 zloader 目录的软链接

3　PMON 编译方法

PMON 的编译较为简单，以笔者的 Fedora 18 i686 系统为例，流程如图 1。

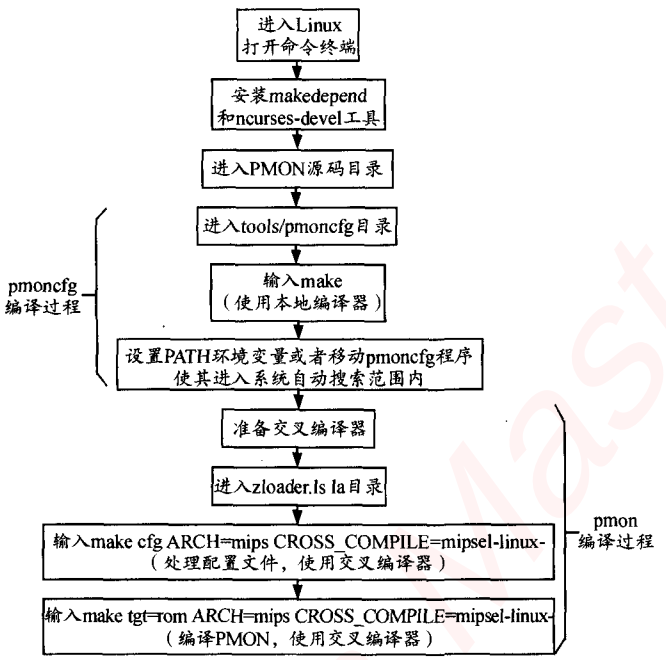

图 1　PMON 编译方法

由图可见，PMON 编译的操作很简单，最终得到 gzrom.bin 二进制文件，该文件可以通过网络、烧写器、EJATG 的方式写入板上 norFlash，然后开机上电后 CPU 就开始执行新编译的 PMON 代码。另外一种 gzram.bin 是可以直接放入内存执行的，暂不讨论。

4　PMON 编译执行流程

在 PMON 编译开始之前，首先得生成一个名为 pmoncfg 的工具，其编译执行流程非常简单，与一般的 C 语言小程序的编译过程很相似，即将多个.c 源文件编译生成.o 目标文件，再链接成可执行程序。但不同的是它利用了词法分析器 flex 和语法分析器 bison，其流程见图 2。

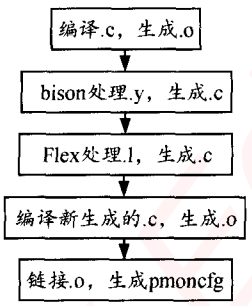

图 2　pmoncfg 工具编译执行流程

PMON 编译期的执行流程是本节的重点，它的编译流程复杂程度很高，涉及到多个 Makefile。这些 Makefile 之间的具体关系见图 3。

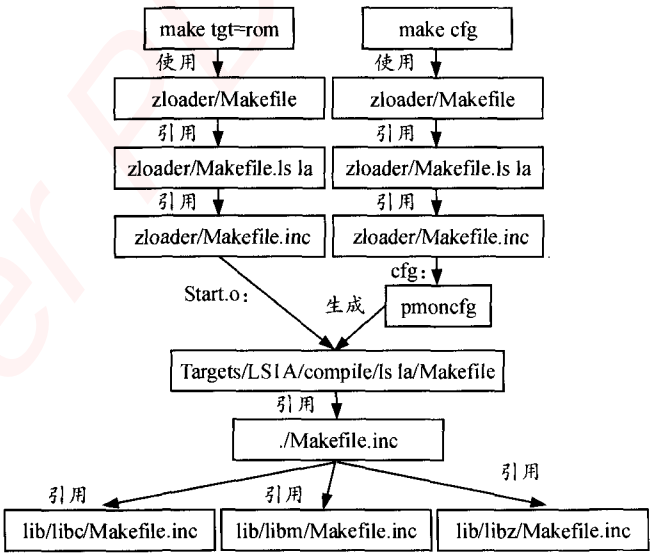

图 3　PMON 编译期各个 Makefile 关系

图中的路径都是对于源码根目录的相对路径。图中“引用”字样代表 makefile 中的“include”指令，即将某个文件原封不动地放进 include 指令所在位置。

在文中第 2 节中提到过“zloader.xxx”字样的软链接会影响使用的 Makefile，被影响的就是图 3 中从上往下数的第 3 层。本例中，使用 zloader.ls1a 进入目录，那么就会使用 Makefile.ls1a。

编译 PMON 最主要的是执行 make cfg(处理配置文件)和 make tgt=rom(生成可执行文件)这 2 步。从图 3 也可知道，make cfg 指令一定要先完成。

make cfg(处理配置文件)步骤的流程最关键的就是执行 pmoncfg 工具根据配置文件生成一个几千行的 Makefile，其余的流程都比较简单而且不影响理解源码，所以文中将其忽略。

make tgt=rom(生成可执行文件)步骤的流程要稍微复杂些，见图 4 所示的简化流程。

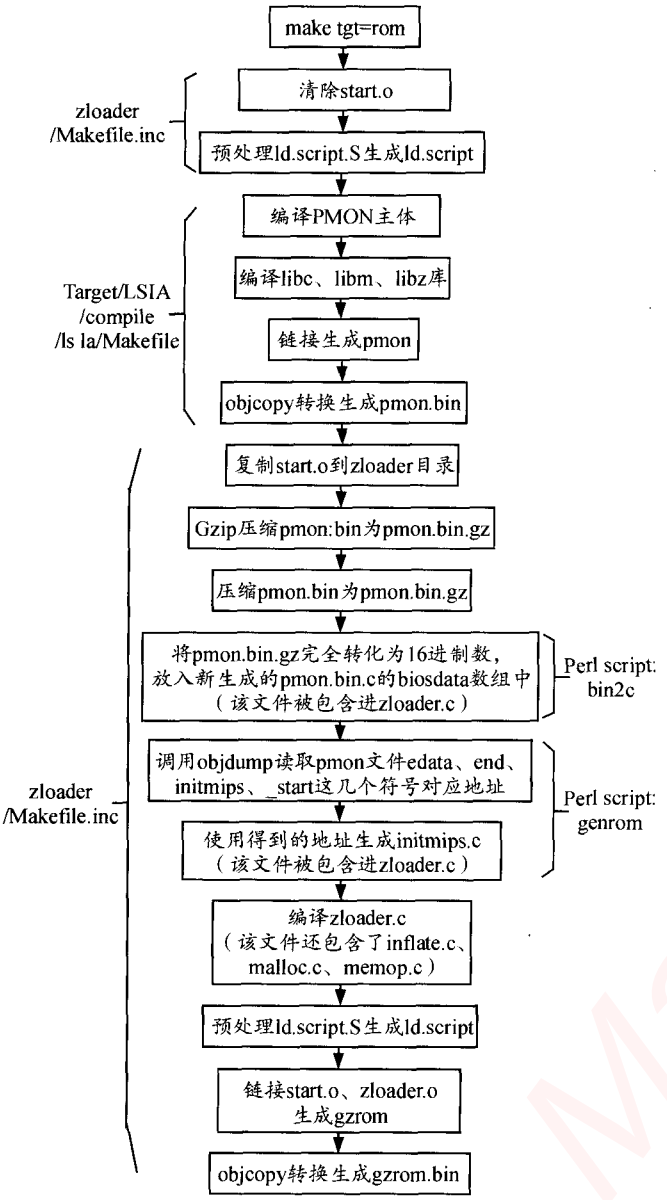

图 4　PMON 二进制文件编译执行流程

从图中可以看到，生成了 2 次 ld.script，实际上这 2 次生成的链接脚本的地址并不相同，也就是说生成的二进制文件起始地址不同。链接过程同样出现了 2 次，第一次是组装 PMON，第二次是组装 zloader。其实从这部分也就可以看出，zloader 并不是 pmon 的一部分，反而是 PMON 成为了 zloader 的一部分，因为最终链接 zloader 时指令较短，其简化后相当于：

mipsel-linux-ld -T ld.script -o gzrom start.o zloader.o

虽然 pmon.bin 已经包含了 start.o，但在 zloader.o 前面同样也包含了 1 次，因为该文件用于 CPU 上电后的关键硬件的初始化，必须保证它最先执行。

该过程还涉及到 2 个 perl 语言编写的脚本，它们的主要功能就是将 zloader 代码和 PMON 二进制内容组装在一起，以便 PMON 启动过程中能正确地在这两者间跳转。

5　PMON 执行流程

龙芯 1A 符合 MIPS32 标准，按照其架构要求，0xA0000000～0xBFFFFFFF（kseg1 段）是唯一在系统重启时能正常工作的内存映射空间[5]，并且地址 0xBFC00000 为重新启动时的入口向量，即 CPU 启动后第一条执行语句所在地址。

对于 kseg1 段的地址，通过将最高 3 位清零的方法，把这些地址映射为相应的物理地址，然后再映射到物理内存中 512 MB 大小的低位[5]。所以启动地址对应的物理地址为 0x1fc00000，且对应于 norFlash 的 0 地址。也就是说，刚开始的 PMON 是直接在 Flash 上执行的。

启动地址的映射关系，这完全是靠硬件设计来完成。在这之后，就要靠 start.S 所包含的源代码来处理接下来的初始化工作，然后跳转到 C 代码的入口继续执行。总的来说，启动过程见图 5。

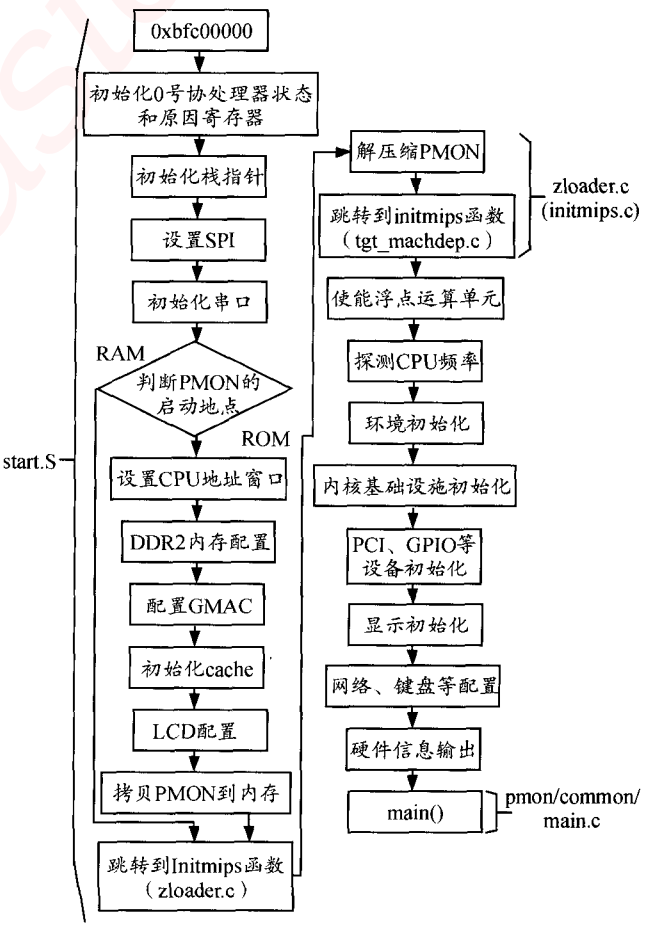

图 5　PMON 启动流程

从图中首先可以看出：一般情况下，PMON 会直接在 Flash 中运行，一段时间后才会将自身拷贝到内存中，并且还需要通过 zloader 将压缩过的 PMON 主程序解压[6]。

而且，PMON 还存在另一种启动流程，该流程是指以 tgt=ram 为编译目标时所得到的，完全运行在内存中，一般用作调试，笔者暂不讨论该情况。

在串口初始化之后，就会看到 PMON 输出的第一行字符串“PMON2000 MIPS Initializing. Standby...”。从这之后，就能不断地在串口终端上看到 PMON 的启动情况。

图中出现了 2 个 initmips 函数，第 1 个存在于自动生成的 initmips.c 中，而且该文件又被包含进 zloader.c，该函数作为 zloader 的入口，同时也是第一个执行的 C 代码程序；第 2 个存在于 tgt_machdep.c 文件，作为 PMON 主体的入口，直接从 zloader 的 realinitmips()函数中跳转过来，同时也对应于串口输出信息“OK,Booting Bios”之后。

initmips()函数(tgt_machdep.c)大部分的工作都是配置相关硬件、系统环境，在最后通过调用 main.c 提供的 main()函数，正式进入 PMON 的世界。

6 结束语

笔者用流程图和文字相结合的方式，较为清晰地描述了 PMON 编译时期和启动时期的执行流程，可为相关人员提供一定的帮助。

参考文献：

[1] Wikipedia. BIOS[EB/OL]. Wikipedia, 2004(2013-11-12). http://zh.wikipedia.org/zh/BIOS.

[2] 龙芯开源社区. PMON[EB/OL]. 龙芯开源百科(2012-6-20). http://www.loongson.cn/dev/wiki/pmon.

[3] LinuxMIPS. PMON[EB/OL]. LinuxMIPS(2010-2-8). http://www.linux-mips.org/wiki/pmon.

[4] LinuxMIPS. PMON 2000[EB/OL]. LinuxMIPS(2013-1-24). http://www.linux-mips.org/wiki/pmon_2000.

[5] Dominic Sweetman. MIPS 体系结构透视[M]. 2 版. 北京：机械工业出版社, 2008: 33-35.

[6] 吴昌昊. 基于 Zentyal 的数据过滤程序[J]. 兵工自动化, 2013, 32(7): 89-92.

(上接第 92 页)

5.2 实验数据

为了验证优化后的效果，笔者将改进前全程直线运输策略比赛与改进后策略的进球数进行比较。由于 2 个策略的进球方法不一样，笔者将目的区域设定为易进球区，5 min 2 个策略进球的数量作为实验数据。如表 1～表 3 所示，进球数明显提高。

表 1 第 1 次测试数据

时间/min	改进前策略进球数	改进后策略进球数
1	0	0
2	0	2
3	0	4
4	1	5
5	2	7

表 2 第 2 次测试数据

时间/min	改进前策略进球数	改进后策略进球数
1	0	0
2	0	2
3	0	4
4	1	6
5	1	8

表 3 第 3 次测试数据

时间/min	改进前策略进球数	改进后策略进球数
1	0	0
2	0	2
3	0	3
4	1	5
5	2	7

5.3 实验结果分析

经过实验数据分析，优化后的策略比之前策略的进球数提高了 3 倍，并且不会出现运球过程中尾部碰触边界弹鱼的情况。能流畅地运输水球，在上半场将 7～8 个水球顶入己方球门，运输效率提高了 90%。实验证明，本策略是行之有效的。

6 结论

优化后的抢球大作战策略充分考虑了比赛场地的地形以及机器鱼和水球的实时信息，可以将整个地形运用起来，在不同的区域执行不同的策略，加强了机器鱼的运输和绕障能力。这只是机器人仿真研究中的冰山一角，笔者将通过深入研究，将机器人协作研究运用到实际生活中去。

参考文献：

[1] 谢广明. 机器人水球比赛项目推介书[M]. 北京：北京大学工学院, 2009: 1-5.

[2] 黄永安, 马路, 刘惠敏. Matlab7.0/simulink 6.0 建模仿真开发与高级工程应用[M]. 北京：清华大学出版社, 2007: 1-75.

[3] 龙海楠, 李淑琴, 安永跃. 仿真机器鱼抢球大作战策略的研究[J]. 计算机仿真, 2012, 30(7): 312-315.

[4] 安永跃, 李淑琴, 龙海楠, 等. 机器鱼仿真水球斯诺克比赛策略[J]. 兵工自动化, 2012, 31(11): 52-54.

[5] 陈晓, 李淑琴, 谢广明. 基于启发式路径评估的仿真机器鱼策略研究[J]. 北京信息科技大学学报, 2012, 28(1): 79-82.

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
16. 基于 CPCI 总线的 PMC 载板设计
17. 基于 VPX 总线的工件台运动控制系统研究与开发
18. PCI Express 流控机制的研究与实现
19. UART16C554 的设计
20. 基于 VPX 的高性能计算机设计
21. 基于 CAN 总线技术的嵌入式网关设计
22. Visual C 串行通讯控件使用方法与技巧的研究
23. IEEE1588 精密时钟同步关键技术研究
24. GPS 信号发生器射频模块的一种实现方案
25. 基于 CPCI 接口的视频采集卡的设计
26. 基于 VPX 的 3U 信号处理平台的设计
27. 基于 PCI Express 总线 1394b 网络传输系统 WDM 驱动设计
28. AT89C52 单片机与 ARINC429 航空总线接口设计
29. 基于 CPCI 总线多 DSP 系统的高速主机接口设计
30. 总线协议中的 CRC 及其在 SATA 通信技术中的应用
31. 基于 FPGA 的 SATA 硬盘加解密控制器设计
32. Modbus 协议在串口通讯中的研究及应用
33. 高可用的磁盘阵列 Cache 的设计和实现
34. RAID 阵列中高速 Cache 管理的优化

35. 一种新的基于 RAID 的 CACHE 技术研究与实现
36. 基于 PCIE-104 总线的高速数据接口设计
37. 基于 VPX 标准的 RapidIO 交换和 Flash 存储模块设计
38. 北斗卫星系统在海洋工程中的应用
39. 北斗卫星系统在远洋船舶上应用的研究
40. 基于 CPCI 总线的红外实时信号处理系统
41. 硬件实现 RAID 与软件实现 RAID 的比较
42. 基于 PCI Express 总线系统的热插拔设计
43. 基于 RAID5 的磁盘阵列 Cache 的研究与实现
44. 基于 PCI 总线的 MPEG2 码流播放卡驱动程序开发
45. 基于磁盘异或引擎的 RAID5 小写性能优化
46. 基于 IEEE1588 的时钟同步技术研究
47. 基于 Davinci 平台的 SD 卡读写优化
48. 基于 PCI 总线的图像处理及传输系统的设计
49. 串口和以太网通信技术在油液在线监测系统中的应用
50. USB30 数据传输协议分析及实现
51. IEEE 1588 协议在工业以太网中的实现
52. 基于 USB30 的设备自定义请求实现方法
53. IEEE1588 协议在网络测控系统中的应用
54. USB30 物理层中弹性缓冲的设计与实现
55. USB30 的高速信息传输瓶颈研究
56. 基于 IPv6 的 UDP 通信的实现

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

15. Bootrom 功能改进经验谈
16. 基于 VxWorks 嵌入式系统的中文平台研究与实现
17. VxBus 的 A429 接口驱动
18. 基于 VxBus 和 MPC8569E 千兆网驱动开发和实现
19. 一种基于 vxBus 的 PPC 与 FPGA 高速互联的驱动设计方法
20. 基于 VxBus 的设备驱动开发
21. 基于 VxBus 的驱动程序架构分析
22. 基于 VxBus 的高速数据采集卡驱动程序开发
23. Vxworks 下的冗余 CAN 通讯模块设计
24. WindML 工业平台下开发 S1d13506 驱动及显示功能的实现
25. WindML 中 Mesa 的应用
26. VxWorks 下图形用户界面开发中双缓冲技术应用
27. VxWorks 上的一种 GUI 系统的设计与实现
28. VxWorks 环境下 socket 的实现
29. VxWorks 的 WindML 图形界面程序的框架分析
30. VxWorks 实时操作系统及其在 PC104 下以太网编程的应用
31. 实时操作系统任务调度策略的研究与设计
32. 军事指挥系统中 VxWorks 下汉字显示技术
33. 基于 VxWorks 实时控制系统中文交互界面开发平台
34. 基于 VxWorks 操作系统的 WindML 图形操控界面实现方法
35. 基于 GPU FPGA 芯片原型的 VxWorks 下驱动软件开发
36. VxWorks 下的多串口卡设计

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
8. 基于 Yocto Project 的嵌入式应用设计
9. Android 应用的反编译
10. 基于 Android 行为的加密应用系统研究
11. 嵌入式 Linux 系统移植步步通
12. 嵌入式 CC++语言精华文章集锦
13. 基于 Linux 的高性能服务器端的设计与研究
14. S3C6410 移植 Android 内核

15. Android 开发指南中文版
16. 图解 Linux 操作系统架构设计与实现原理（第二版）
17. 如何在 Ubuntu 和 Linux Mint 下轻松升级 Linux 内核
18. Android 简单 mp3 播放器源码
19. 嵌入式 Linux 系统实时性的研究
20. Android 嵌入式系统架构及内核浅析
21. 基于嵌入式 Linux 操作系统内核实时性的改进方法研究
22. Linux TCP IP 协议详解
23. Linux 桌面环境下内存去重技术的研究与实现
24. 掌握 Android 7.0 新增特性 Quick Settings
25. Android 应用逆向分析方法研究
26. Android 操作系统的课程教学
27. Android 智能手机操作系统的研究
28. Android 英文朗读功能的实现
29. 基于 Yocto 订制嵌入式 Linux 发行版
30. 基于嵌入式 Linux 的网络设备驱动设计与实现
31. 如何高效学习嵌入式
32. 基于 Android 平台的 GPS 定位系统的设计与实现
33. LINUX ARM 下的 USB 驱动开发
34. Linux 下基于 I2C 协议的 RTC 驱动开发
35. 嵌入式下 Linux 系统设备驱动程序的开发
36. 基于嵌入式 Linux 的 SD 卡驱动程序的设计与实现
37. Linux 系统中进程调度策略
38. 嵌入式 Linux 实时性方法
39. 基于实时 Linux 计算机联锁系统实时性分析与改进
40. 基于嵌入式 Linux 下的 USB30 驱动程序开发方法研究
41. Android 手机应用开发之音乐资源播放器
42. Linux 下以太网的 IPv6 隧道技术的实现
43. Research and design of mobile learning platform based on Android

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]
16. 基于 WinCE 的 BootLoader 研究
17. Windows CE 环境下无线网卡的自动安装
18. 基于 Windows CE 的可视电话的研究与实现
19. 基于 WinCE 的嵌入式图像采集系统设计
20. 基于 ARM 与 WinCE 的掌纹鉴别系统
21. DCOM 协议在网络冗余环境下的应用
22. Windows XP Embedded 在变电站通信管理机中的应用

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
8. 基于 MPC8247 嵌入式电力交换系统的设计与实现
9. 基于设备树的 MPC8247 嵌入式 Linux 系统开发
10. 基于 MPC8313E 嵌入式系统 UBoot 的移植
11. 基于 PowerPC 处理器 SMP 系统的 UBoot 移植
12. 基于 PowerPC 双核处理器嵌入式系统 UBoot 移植
13. 基于 PowerPC 的雷达通用处理机设计
14. PowerPC 平台引导加载程序的移植
15. 基于 PowerPC 嵌入式内核的多串口通信扩展设计
16. 基于 PowerPC 的多网口系统抗干扰设计
17. 基于 MPC860T 与 VxWorks 的图形界面设计
18. 基于 MPC8260 处理器的 PPMC 系统
19. 基于 PowerPC 的控制器研究与设计
20. 基于 PowerPC 的模拟量输入接口扩展

21. 基于 PowerPC 的车载通信系统设计

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
10. ARM 经典 300 问
11. 基于 S5PV210 的频谱监测设备嵌入式系统设计与实现
12. Uboot 中 start.S 源码的指令级的详尽解析
13. 基于 ARM9 的嵌入式 Zigbee 网关设计与实现
14. 基于 S3C6410 处理器的嵌入式 Linux 系统移植
15. CortexA8 平台的μC-OSⅡ及 LwIP 协议栈的移植与实现
16. 基于 ARM 的嵌入式 Linux 无线网卡设备驱动设计
17. ARM S3C2440 Linux ADC 驱动
18. ARM S3C2440 Linux 触摸屏驱动
19. Linux 和 Cortex-A8 的视频处理及数字微波传输系统设计
20. Nand Flash 启动模式下的 Uboot 移植
21. 基于 ARM 处理器的 UART 设计
22. ARM CortexM3 处理器故障的分析与处理
23. ARM 微处理器启动和调试浅析
24. 基于 ARM 系统下映像文件的执行与中断运行机制的实现
25. 中断调用方式的 ARM 二次开发接口设计
26. ARM11 嵌入式系统 Linux 下 LCD 的驱动设计
27. Uboot 在 S3C2440 上的移植
28. 基于 ARM11 的嵌入式无线视频终端的设计
29. 基于 S3C6410 的 Uboot 分析与移植
30. 基于 ARM 嵌入式系统的高保真无损音乐播放器设计

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
8. 基于 COM Express 架构的数据记录仪的设计与实现
9. 基于 COM Express 的信号系统逻辑运算单元设计
10. 基于 COM Express 的回波预处理模块设计
11. 基于 X86 平台的简单多任务内核的分析与实现
12. 基于 UEFI Shell 的 PreOS Application 的开发与研究
13. 基于 UEFI 固件的恶意代码防范技术研究
14. MIPS 架构计算机平台的支持固件研究
15. 基于 UEFI 固件的攻击验证技术研究
16. 基于 UEFI 的 Application 和 Driver 的分析与开发
17. 基于 UEFI 的可信 BIOS 研究与实现
18. 基于 UEFI 的国产计算机平台 BIOS 研究
19. 基于 UEFI 的安全模块设计分析
20. 基于 FPGA NiosⅡ的等精度频率计设计
21. 基于 FPGA 的 SOPC 设计
22. 基于 SOPC 基本信号产生器的设计与实现
23. 基于龙芯平台的 PMON 研究与开发
24. 基于 X86 平台的嵌入式 BIOS 可配置设计
25. 基于龙芯 2F 架构的 PMON 分析与优化
26. CPU 与 GPU 之间接口电路的设计与实现

Programming:

1. 计算机软件基础数据结构 - 算法
2. 高级数据结构对算法的优化
3. 零基础学算法
4. Linux 环境下基于 TCP 的 Socket 编程浅析
5. Linux 环境下基于 UDP 的 socket 编程浅析
6. 基于 Socket 的网络编程技术及其实现