



随书DVD含本书源码、学习环境及资料

Broadview
www.broadview.com.cn



Professional Embedded Software Development

专业嵌入式软件开发

全面走向高质高效编程

李云 著



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

http://www.phei.com.cn

Professional Embedded Software Development

专业嵌入式软件开发

全面走向高质高效编程

李云◎著

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书分为 6 篇。硬件篇就嵌入式软件开发所需掌握的处理器概念进行了介绍。工具篇对 make、gcc 编译器、bintuils 工具集、ld 链接器和 gdb 调试器进行了讲解，其中对 make 这一嵌入式开发环境的全能管家进行了精辟的介绍，致力于帮助读者成为 Makefile 方面的专家。编程语言篇致力于让读者更深入地理解 C 编程语言。操作系统篇通过循序渐进的方式介绍 ClearRTOS 的设计与实现，使得读者能透彻地理解操作系统的关键概念和实现原理。设计篇和质量保证篇通过实践的方式逐步展开讲解，以帮助读者获得一些实用的设计原则、最佳实践和一套有效的质量保证方法论。

本书适合嵌入式软件开发领域的新手和在工作中碰到瓶颈的老手阅读。阅读本书要求读者已掌握 C 编程语言和基本的 UML 知识。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有，侵权必究。

图书在版编目 (CIP) 数据

专业嵌入式软件开发：全面走向高质高效编程 / 李云著. —北京：电子工业出版社，2012.1
ISBN 978-7-121-14783-8

I. ①专… II. ①李… III. ①程序设计 IV. ①TP311.1

中国版本图书馆 CIP 数据核字 (2011) 第 206049 号

策划编辑：张春雨

责任编辑：葛 娜

印 刷：北京东光印刷厂

装 订：三河市皇庄路通装订厂

出版发行：电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本：787×1092 1/16 印张：40 字数：1020 千字

印 次：2012 年 1 月第 1 次印刷

印 数：4000 册 定价：108.00 元 (含 DVD 光盘 1 张)

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：(010) 88258888。

前言

我于 2000 年第一次接触嵌入式软件开发工作，那时和很多入门者一样，因为找不到全面、易懂、深入的读物，也没有人指导，因而遭遇了极大的自学痛苦。即使在今天，学习嵌入式软件开发似乎仍困难重重，这从我的博客空间不时有网友发私信询问如何学习可以看出。

我也曾被网友要求推荐学习嵌入式软件开发的好书。但当我以“嵌入式”关键字在网上书店进行搜索时，所获得的书大部分与 Linux、Windows CE、Android 和 ARM 有关。在我看来，网友并不是让我帮助他选择 Linux 还是 Windows CE，ARM 还是 x86，而认为他希望获得一本学习通用原理和方法的书，因此不敢贸然推荐。基于这种现状，我萌发了写一本既能指导新手入门，又能帮助老手获得突破的书。读者手上拿的正是这本书！本书的创作始于 2009 年 6 月，历时 2 年后于 2011 年下半年面市。

在本书的创作之初，我问自己：这本书应当包含哪些内容呢？或许可以根据自己过去十多年所经历并克服的成长痛苦进行编排！

嵌入式软件开发是一种软硬件结合非常紧密的职业，对工程师的能力要求自然也就高了。刚开始学习嵌入式软件开发时，最困难的莫过于学习操作系统原理和处理器方面的知识，所以本书必须包含这两方面的内容。讲解操作系统原理如果以 Linux、Windows CE 等成熟的操作系统为素材并不好，因为它们太大，很容易让人“只见森林不见树木”，也容易让人望而生畏而失去学习的兴趣和信心。从软件开发的角度来看，操作系统的概念和实现原理一旦掌握，不论基于哪一个操作系统做开发都只是调用不同的函数而已。为了让读者获得最好的学习体验，我为本书设计了一个实现简洁、完整的“实时”^①操作系统——ClearRTOS，通过渐进式的方式细致地讲解操作系统的概念和实现原理。至于处理器方面的知识，本书没有针对某一具体处理器，而是就编程方面所需的通用知识进行了介绍。对这些通用知识的掌握，将使得处理器对于读者不再那么神秘。

学习嵌入式软件开发的另一大困难是实践问题，本书必须帮助读者解决这一问题。对于很多初学者来说，为了实践而购买一块开发板的学习成本偏高。值得欣喜的是，读者学习本书并不需要购买开发板，而只需要有一台安装于 x86 或 x86-64（包括 Intel 64^②和 AMD 64）处理器

① “实时”打上引号是因为 ClearRTOS 在 Cygwin 环境和 Linux 操作系统中无法直接接管处理器的中断，所以无法实现在中断返回的过程中完成任务切换的功能。这是与真正的嵌入式操作系统唯一的区别。

② IA-64 不同于 Intel 64，后者是指 x86-64，详情参见 <http://en.wikipedia.org/wiki/x86-64>。

上的 Windows 或 Linux 操作系统的计算机，对于大多数读者来说这样的学习环境就在身边。另外，软件开发工程师有一个特点，对于自己能修改和调试的代码更具学习兴趣，通过这种方式学习的效果也更佳。本书的所有代码（包括 ClearRTOS）都被设计成能在 Cygwin 环境^③和 Linux 操作系统上编译、调试和运行，所以本书完全迎合工程师的这一学习偏好。总的说来，实践性强是本书很突出的一个特色。

掌握开发所需的工具是学习嵌入式软件开发的又一大挑战，本书在这方面也花费了大量笔墨。与非嵌入式软件开发采用集成开发环境不同，嵌入式软件开发大多是基于命令行的。软件开发工程师除了进行编码工作，还需要能驾驭自己的编译环境并运用其他的开发工具辅助开发工作。本书的工具篇以来自 GNU 的工具为例帮助读者战胜这一挑战。值得强调的是，其中花了很大的篇幅帮助读者成为 Makefile 方面的专家。

如果读者只想入门，那么掌握操作系统、处理器和必要的工具就足够了。但如果想获得突破，以实现高质高效地从事软件开发工作显然不够，还必须理解软件设计的重要性，并借助一定的质量保证方法论来提高工作质量和效率。软件设计和质量保证方法论是业内比较抽象和高级的话题，为此本书在设计篇和质量保证篇通过实践的方式逐步展开讲解，以帮助读者获得一些实用的设计原则、最佳实践和一套有效的质量保证方法论。

总而言之，本书从知识、工具、方法和思想这四大方面全面讲解如何专业地从事嵌入式软件开发，致力于帮助读者全面走向高质高效编程。

读者阅读本书之前，需要掌握 C 编程语言和基本的 UML 知识^④。如果有使用 Linux 操作系统的基础经验，对学习本书也会有小小的帮助^⑤。尽管本书是针对嵌入式领域的，但书中的很多思想和方法适用于整个软件行业。

本书结构

全书分为 6 大篇共 33 章，读者可以通过浏览书的目录以进一步了解各篇所涵盖的内容。

硬件篇就嵌入式软件开发所需掌握的处理器概念进行了介绍，并通过介绍电路信号的完整性问题告诉读者，嵌入式产品的质量不是软件质量单方面能保证的。

工具篇介绍了提高嵌入式软件开发效率所需掌握的工具。make 作为嵌入式开发环境的全能管家，在本篇中花了较大的篇幅对其进行精辟的介绍。此外，gcc 编译器、binutils 工具集、

③ Cygwin 是一个开源项目，实现了在 Windows 操作系统上虚拟 Linux 操作系统的环境。

④ UML 是“Unified Modeling Language”的缩写，即统一建模语言。如果需要，读者可以以“跟我学 UML”为关键字在我的博客中查找所需的学习资料。

⑤ 本书并不需要读者有丰富的 Linux 操作系统使用经验，只需掌握十几个命令就行了。即使读者第一次接触 Linux，对于学习本书也不会有困难。

ld 链接器和 gdb 调试器都在本篇中涵盖了。对于工具的介绍是基于实用的角度展开的，而不是“大全”。

编程语言篇致力于让读者更深入地理解 C 编程语言。其中对程序的结构、ABI/EABI、volatile 关键字进行了讲解，这几方面的知识在非嵌入式软件开发中并不需要深入了解，但在嵌入式软件开发中却是必须掌握的。本篇还通过分析一个因混淆指针和数组所导致的问题，指出开发活动中容易忽视的一个认识盲点，并提出了预防这类问题的终极方法。

设计篇解释了为什么设计是软件产品的质量之本，还介绍了作者常用的设计原则及所倡导的软件设计思想和一些最佳实践。设计思想包括：平台与框架开发、可查错性设计、可开发性设计；最佳实践则覆盖模块管理和错误管理。

操作系统篇通过循序渐进的方式介绍 ClearRTOS 的设计与实现，使得读者能透彻地理解操作系统的关键概念和实现原理。读者掌握这篇的内容，有助于轻松地在实时 Linux、VxWorks、Windows CE 等各种实时操作系统上从事软件开发工作。

质量保证篇关注于如何通过质量保证方法论来获得高质量的软件产品，也探讨了工程师的编程习惯对软件质量的影响。本篇中强调了单元测试这一被忽视的质量保证方法的价值，并通过设计实用的单元测试框架展示如何在项目中实施它。本篇中还展示了如何将代码覆盖、静态分析、动态分析和性能分析无缝地整合到开发环境中，以及阐述了“以单元测试为中心”和“要素有形化”质量保证方法论设计思想的具体含义。

致谢

本书是我的处女作，能与读者见面离不开很多人的支持和帮助。首先，感谢我的妻子和女儿。正是在妻子的提议下，我从写博客开始扬起了本书的写作之帆。女儿则是我的开心果，给我的写作之路带来了很多的乐趣，让我得到更多的放松机会。

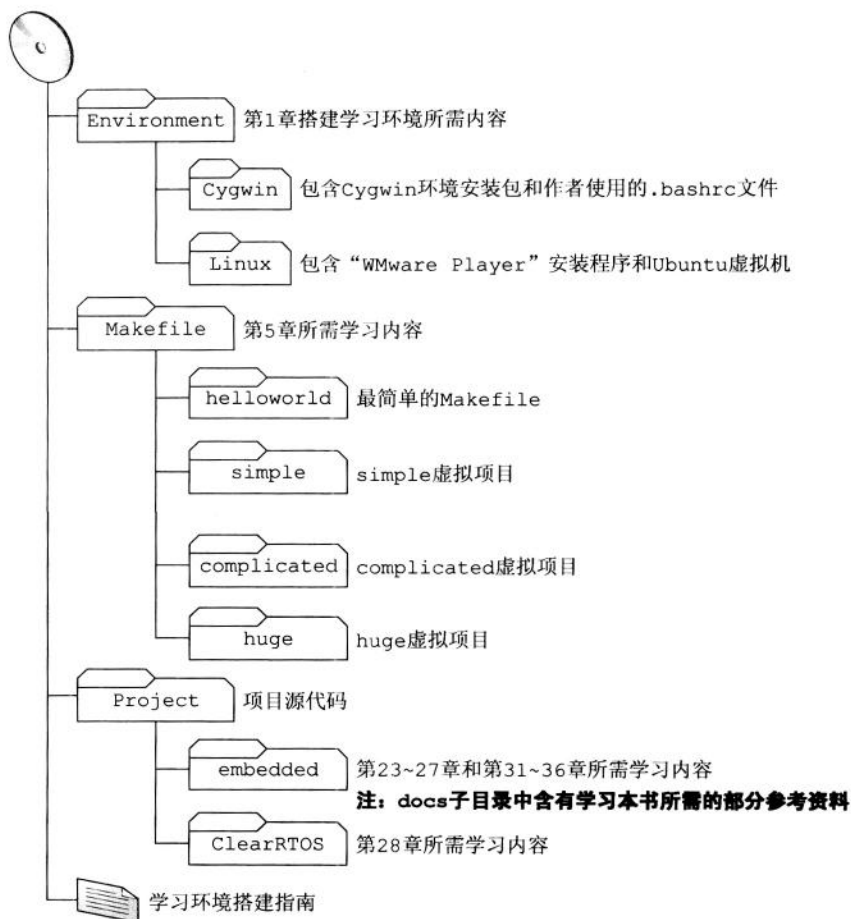
感谢我的朋友及职业生涯中的上司和同事，正是他们给我机会，或鼓励，或帮助，让我一路积累，才有可能完成本书的创作。他们包括但不限于：庞惠民、章佳欢、刘伟民、夏青、于善成、范鹏、罗延庭。

感谢 51CTO 博客的同仁，他们的幕后支持让我坚持了下来。广大 51CTO 博友期待也激励着我努力地写好本书。

感谢电子工业出版社的策划编辑张春雨，他的出现加速了本书的面市，也给我吃了一颗将书写到底的“定心丸”。与他交流写作方面的话题让我感受到了什么是隔行如隔山，他对出版行业的专家意见和追求满分的精神让本书增色不少。

最后，再一次感谢我的妻子和好友于善成，两位预读了本书并提出了自己的真知灼见，使得本书更简练、严谨和更具可读性。

附书光盘内容介绍



书中如果出现错误,请先接受作者的致歉,如能来信告知那将不胜感激。错误一旦发现会通过我的博客第一时间通知其他读者。读者在学习过程中如需帮助,可以通过技术圈发帖讨论。

联系方式

邮箱: yunli.book@gmail.com
微博: weibo.com/UltraEmbedded
博客: yunli.blog.51CTO.com
技术圈: g.51CTO.com/UltraEmbedded

李 云

2011-07-04

目 录

硬 件 篇

第 1 章 处理器的基本概念	2
1.1 区分微处理器与微控制器	2
1.2 寄存器	2
1.3 处理器是如何启动的	4
1.4 输入与输出	4
1.5 指令与数据	5
1.6 中断	6
1.7 字节序	8
1.8 边界对齐	10
1.9 程序断点和数据断点	15
1.10 内存管理单元	16
1.11 缓存	17
1.12 小结	18
第 2 章 开发活动中的硬件问题	19
2.1 两个案例	19
2.2 案例的背后——信号完整性	19
2.3 应对方法	21
2.4 小结	21

工 具 篇

第 3 章 make, 开发环境全能管家	24
3.1 从最简单的 Makefile 中了解规则	24
3.2 创建基本的编译环境	29
3.2.1 将规则运用于程序编译	30
3.2.2 让 Makefile 更专业	34
3.3 提高编译环境的实用性	48

3.3.1	让编译环境更加有序	48
3.3.2	提升依赖关系管理	51
3.4	打造更专业的编译环境	67
3.4.1	规划项目目录结构	68
3.4.2	增进复用性	72
3.4.3	支持头文件目录的指定	75
3.4.4	实现库链接	77
3.4.5	增强可使用性	82
3.4.6	管理对库的依赖关系	84
3.4.7	改善编译效率	87
3.4.8	恰当地书写注释	89
3.5	理解 make 的解析行为	90
3.6	Makefile 的调试	91
3.7	make 的常用选项	92
3.8	活用 make	92
3.9	小结	94
第 4 章	gcc, C 语言编译器	96
4.1	什么是交叉编译器	96
4.2	gcc 幕后工作揭示	97
4.3	实用的 gcc 选项	99
4.3.1	解决宏错误的好帮手	99
4.3.2	辅助编写汇编程序的好方法	100
4.3.3	获取系统头文件路径	101
4.3.4	产生映射文件	102
4.3.5	通过选项定义宏	102
4.3.6	生成依赖关系	103
4.3.7	指定链接库	104
第 5 章	binutils 工具集, 软件开发利器	107
5.1	addr2line, 指令地址翻译器	108
5.2	ar, 静态库生成器	111
5.3	nm, 符号显示器	113
5.4	objdump, 信息查看器	115
5.5	objcopy, 段编辑器	119
5.6	ranlib, 库索引生成器	120
5.7	size, 段大小观察器	121
5.8	strings, 字符串窥视器	122

5.9 strip, 程序文件瘦身器	124
第 6 章 ld, 链接器	125
6.1 重定位的概念	125
6.2 链接脚本	126
6.2.1 段	128
6.2.2 符号	129
6.2.3 存储区域	130
6.2.4 常用命令	131
6.3 常用选项	137
6.3.1 指定程序的入口点	137
6.3.2 生成可重定位的中间文件	137
6.3.3 指定链接脚本	138
练习与思考	138
第 7 章 gdb, 程序调试助手	139
7.1 启动和退出 gdb	139
7.2 获取帮助	140
7.3 调试程序	142
7.3.1 断点设置	142
7.3.2 控制程序运行	144
7.3.3 检查程序	147
7.3.4 提高调试效率	151
7.4 查看符号表	152
7.5 控制 gdb 的行为	153

编程语言篇

第 8 章 掌握必要的汇编知识	156
8.1 as 的语法	156
8.1.1 宏	157
8.1.2 汇编命令	157
8.1.3 符号和标签	157
8.1.4 汇编指令	158
8.2 嵌入汇编的语法	158
第 9 章 深入理解程序的结构	161
9.1 段	161
9.1.1 指令段	161

9.1.2 数据段	162
9.2 栈	166
9.3 堆	168
9.4 小结	169
第 10 章 ABI/EABI 规范，缔造程序兼容合约	170
10.1 定义基本数据类型	171
10.2 规范字节对齐处理	171
10.3 分配寄存器的功能	173
10.4 规定栈帧结构	174
10.4.1 栈帧的含义和作用	175
10.4.2 函数参数的传递方法	182
10.4.3 函数返回值的返回方法	184
10.5 小结	187
练习与思考	187
第 11 章 混淆指针与数组所导致的问题	188
11.1 问题示例	188
11.2 问题分析	189
11.2.1 数组的内存模型	189
11.2.2 指针的内存模型	190
11.3 问题成因	191
11.4 预防措施	193
11.5 小结	194
第 12 章 volatile，让我保持原样	195

设计篇

第 13 章 设计，软件质量之本	200
13.1 软件设计是什么	200
13.2 软件质量的概念	201
13.3 阻碍改善设计的常见观念	203
13.3.1 测试是替罪羊或救命稻草	203
13.3.2 资源永远不足	204
13.3.3 不改变就可以规避风险	204
13.4 如何提高设计能力	205
13.5 设计模式、设计原则和设计思想	206
13.6 放之四海皆适用的设计原则	207

13.6.1	以人为本	207
13.6.2	追求简单性	210
13.6.3	让模块善始善终	211
13.6.4	重视收集统计信息	212
13.6.5	借助命名传达设计意图	213
13.6.6	消除“审美告警”	215
13.6.7	通过机制解决问题	215
13.6.8	防止他人犯错	218
13.6.9	考虑可查错性	220
13.7	小结	221
第 14 章 模块管理, 保障系统有序运行		222
14.1	管理参照系	222
14.2	设计思路	224
14.3	程序实现	226
14.3.1	引入模块标识	226
14.3.2	实现层与级的表达	226
14.3.3	系统状态和回调函数原型定义	228
14.3.4	模块注册	228
14.3.5	系统启动	230
14.3.6	系统关闭	232
14.4	module 示例程序	233
14.5	模块管理的一些思考	235
14.6	小结	235
	练习与思考	235
第 15 章 错误管理, 不可或缺的用户需求		236
15.1	表达错误的通用方法	236
15.1.1	错误码格式	237
15.1.2	定义方法	238
15.1.3	使用示例	239
15.1.4	提高可使用性	240
15.1.5	定义和使用错误码的准则	246
15.2	优化错误日志的输出	246
15.2.1	传统方法	246
15.2.2	更有效的方法	249
15.3	平台和框架层的错误处理	251
15.4	小结	251

第 16 章 目录结构管理，使项目进展更顺利	252
16.1 规划目录结构的意义	252
16.1.1 书架功能	252
16.1.2 意识引导	252
16.1.3 加速新手上手	253
16.2 出色目录结构的特点	253
16.3 一个示例	253
16.4 小结	254
第 17 章 平台与框架开发，高质量软件打造之路	255
17.1 区分系统库、平台和框架	255
17.1.1 系统库	255
17.1.2 平台	256
17.1.3 框架	256
17.2 本质和优点	257
17.3 确立架构模型	258
17.4 小结	259
第 18 章 可开发性设计，一种高效且经济的开发模式	260
18.1 可开发性问题一瞥	260
18.2 可开发性设计的内涵	261
18.3 引入设备抽象层	261
18.4 更复杂的设备抽象层	263
18.5 图形界面的可开发性设计	264
18.5.1 增强设备抽象层	264
18.5.2 提供可视化编辑环境	264
18.6 其他可开发性设计	264
18.7 小结	265

操作系统篇

第 19 章 引导加载器，系统启航者	268
19.1 功能	268
19.2 文件存储布局	269
19.3 程序加载原理	270
19.4 优点	274
19.5 小结	274
练习与思考	275

第 20 章 任务, 软件基本调度单元 276

20.1	任务情景	278
20.1.1	情景内容	278
20.1.2	情景保存	279
20.1.3	情景恢复	281
20.1.4	情景切换	282
20.2	任务调度	286
20.2.1	调度算法	286
20.2.2	调度器	290
20.3	任务的生命周期	293
20.4	任务控制	295
20.4.1	任务创建	297
20.4.2	任务启动	306
20.4.3	任务删除	307
20.4.4	任务挂起	309
20.4.5	任务恢复	310
20.4.6	任务睡眠	311
20.5	竞争问题与中断控制	313
20.5.1	竞争问题的产生	314
20.5.2	通过中断控制解决竞争问题	315
20.5.3	中断控制的嵌套问题	316
20.6	任务与中断状态	317
20.7	任务栈溢出检测	318
20.8	滴答与空闲任务	320
20.9	多任务环境控制	323
20.10	任务模块管理	324
20.11	taskv1 示例程序	326
20.12	任务钩子函数	330
20.13	任务变量	334
20.13.1	taskv2 示例程序	334
20.13.2	原理	336
20.13.3	实现	337
20.14	其他概念与思考	340
20.14.1	抢占式任务与实时系统的关系	340
20.14.2	影响任务切换效率的因素	341
20.14.3	避免直接删除任务	341
20.14.4	小心多任务设计被滥用	342

20.15 小结	343
练习与思考	343
第 21 章 任务同步与通信, 实现协同工作	345
21.1 信号量	345
21.1.1 应用场合	345
21.1.2 程序实现	347
21.1.3 semaphore 示例程序	358
21.2 互斥锁	360
21.2.1 应用场合	361
21.2.2 程序实现	361
21.2.3 mutex 示例程序	365
21.2.4 优先级反转与继承	367
21.2.5 递归锁	375
21.3 事件	379
21.3.1 应用场合	379
21.3.2 程序实现	379
21.3.3 event 示例程序	384
21.4 消息队列	386
21.4.1 应用场合	386
21.4.2 程序实现	387
21.4.3 实现消息队列	390
21.4.4 queue 示例程序	396
21.4.5 使用指南	398
21.5 死锁及预防	399
21.6 小结	399
练习与思考	400
第 22 章 内存管理, 协调动态内存的使用	401
22.1 堆管理	401
22.1.1 heapv1 示例程序	401
22.1.2 程序实现	406
22.1.3 设计改进	416
22.1.4 支持内存泄漏检测	421
22.1.5 实现内存溢出检测	431
22.1.6 内存碎片问题	431
22.2 内存池管理	432
22.2.1 mpool 示例程序	432

22.2.2	程序实现	436
22.2.3	缓冲区泄漏检测	444
22.3	小结	444
	练习与思考	444
第 23 章	设备管理，方便与外设交互	445
23.1	字符设备管理	445
23.2	中断管理	447
23.2.1	中断向量表	447
23.2.2	中断控制	448
23.2.3	中断状态管理	450
23.2.4	设备与中断	451
23.2.5	模块管理	451
23.3	实现设备管理	452
23.3.1	安装驱动程序	454
23.3.2	注册设备	455
23.3.3	打开设备	456
23.3.4	关闭设备	458
23.3.5	设备读写与控制	458
23.4	设备驱动程序实现	459
23.4.1	“滴答”设备	460
23.4.2	控制台设备	462
23.4.3	终止程序运行设备	464
23.5	驱动安装与设备注册	466
23.6	小结	468
	练习与思考	468
第 24 章	定时器，程序闹钟	469
24.1	软件定时器分类	469
24.2	设计思路	469
24.3	中断回调定时器	470
24.3.1	程序实现	470
24.3.2	timerv1 示例程序	481
24.4	定时误差	484
24.5	提高遍历效率	484
24.6	改善实时性	489
24.6.1	实时性分析	490
24.6.2	改进实时性	491

24.7	任务回调定时器	494
24.7.1	程序实现	494
24.7.2	timerv3 示例程序	497
24.8	小结	498
	练习与思考	498

第 25 章 ClearRTOS “实时”操作系统 499

25.1	设计原则	499
25.2	源程序目录管理	499
25.3	让 Makefile 体现概念	502
25.4	实现集中配置	503
25.5	改进与移植	504

质量保证篇

第 26 章 质量保证导言 508

26.1	软件开发的特点	508
26.1.1	脑力密集型工作	508
26.1.2	实现不具唯一性	508
26.1.3	隐性成本高	509
26.1.4	忽视的细节很容易被放大	509
26.1.5	质量难以评估	509
26.2	保证质量的关键要素	509
26.2.1	完备的需求分析	510
26.2.2	高质量的设计	510
26.2.3	编程好习惯	510
26.2.4	充分的验证	510
26.2.5	必要的流程	511
26.2.6	合适的工具	512
26.2.7	言简意赅的文档	512
26.3	质量保证需要系统性的方法论	512
26.3.1	方法论 = 流程 + 工具	513
26.3.2	构建有效方法论的核心手段	516
26.4	走出质量困境的指导性思想	518
26.4.1	从管理者的角度	518
26.4.2	从工程师的角度	519
26.4.3	从组织的角度	519
26.5	小结	520