提供源码 讲述原理

# 嵌入式操作系统内核调度

## ——底层开发者手册

王 奇 谷志茹 姜日凡 编著


从无到有，和你一起编写
实时嵌入式操作系统内核
——基于STM32F103处理器

- 操作系统内核
  也许并没你想象的那么神秘

北京航空航天大学出版社
BEIHANG UNIVERSITY PRESS

# 嵌入式操作系统内核调度
# ——底层开发者手册

王　奇　谷志茹　姜日凡　编著



北京航空航天大学出版社

## 内 容 简 介

《嵌入式操作系统内核调度——底层开发者手册》从程序员的角度设计、编写嵌入式操作系统，实现了内核调度功能。作者按照介绍原理、设计编码、举例验证的顺序逐一介绍各功能的编写方法，为读者理解和应用嵌入式操作系统提供了一种全新的思路。

本手册共分7章，第1章概述操作系统的基本概念；第2章介绍编写操作系统任务调度程序所需具备的基本知识；第3章讲解如何编写非抢占式嵌入式操作系统 Wanlix；第4章和第5章讲解编写实时抢占式嵌入式操作系统 Mindows 的方法；第6章讲解在4种操作系统下分别编写相同结构的任务调度程序；第7章简述进程机制，并使用线程模拟多进程。

本手册可供从事嵌入式开发工作的程序员、高等院校本科生及研究生参考，适合具有一定 C 语言基础的读者阅读。

**图书在版编目(CIP)数据**

嵌入式操作系统内核调度：底层开发者手册 / 王奇，谷志茹，姜日凡编著. -- 北京：北京航空航天大学出版社，2015.1

ISBN 978-7-5124-1611-6

Ⅰ. ①嵌… Ⅱ. ①王… ②谷… ③姜… Ⅲ. ①实时操作系统 Ⅳ. ①TP316.2

中国版本图书馆 CIP 数据核字(2014)第 241355 号



嵌入式操作系统内核调度——底层开发者手册

王 奇 谷志茹 姜日凡 编著

责任编辑 梅栾芳

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) http://www.buaapress.com.cn

发行部电话：(010)82317024 传真：(010)82328026

读者信箱：emsbook@gmail.com 邮购电话：(010)82316936

涿州市新华印刷有限公司印装 各地书店经销

*

开本：710×1 000 1/16 印张：29.25 字数：623 千字

2015 年 1 月第 1 版 2015 年 1 月第 1 次印刷 印数：3 000 册

ISBN 978-7-5124-1611-6 定价：69.00 元

# 序

本手册介绍实时嵌入式操作系统内核任务调度的实现方法及相关知识。

部分介绍操作系统的书籍只是从应用的角度讲解操作系统，有些可能连译者自己都没搞清楚，就更不要说让读者明白了，有些甚至是直接翻译操作系统的用户手册就出书了，这样的书籍对于学习操作系统原理来说意义不大。本手册最大的特点是从操作系统结构设计、编写操作系统的角度讲述操作系统内核的调度原理，并结合多个例子演示一个一边设计、一边编码、一边写书的过程，记录了操作系统从无到有的过程，讲解了实现操作系统的基本原理，读者只要了解 C 语言，再对汇编语言和处理器结构稍微有所了解便能看懂本手册。

2005 年 4 月，经历了漫长的学生时代，我终于参加工作了。进入公司后，我选择了做软件，直至今天。刚入公司时软件基础太差，学校里学的知识也仅使我知道一点 C 语言的概念，几乎从来没编写过代码。好在当时所做的项目已过了编码阶段，我的工作就是学习别人的代码并帮助测试、修改问题。现在回想起来，在这平淡的工作过程中有三点对我至关重要：① 正是在这段时间培养了我比较扎实的 C 语言基础，虽不能说学到了很多，但是让我明白了很多最基本的概念，知道了学习的方法；② 正是在这段时间我接触了项目开发的一些流程，参与到历时几年几百人协作的项目开发中，经历了大项目的开发过程，接触到了很多在学校里永远不会接触到的事物，这些经验对我至关重要，虽然只是冰山一角；③ 正是在这段时间让我有机会接触到嵌入式操作系统——VxWorks，尽管只是嵌入式操作系统的一些应用层用法。

工作一年半后我换到了一个小部门，从原有部门做大系统的冰山一角到做麻雀虽小五脏俱全的整个小型嵌入式系统，各有各的难处，但也各有各的好处，这也为我编写本手册提供了必要条件。

在做小系统时有一个问题一直困扰着我：我所做的下位机设备需要与上位机设备通信，上位机下发的消息需要下位机实时回应，并且下位机可能会同时处理多条消息，这样看来，如果有一个嵌入式操作系统解决任务调度问题，那么下位机的设计就会比较方便。但由于我们的小型嵌入式系统硬件资源受到一些限制，主频低、存储空间少，很难找到一款合适的嵌入式操作系统。当时的一些符合硬件资源条件的操作系统要么需要收费，要么不提供源码又没有可靠的服务保障，最不能接受的是资料不全，使用起来非常不方便，因此我在做这些小系统时一直是“裸奔”。“裸奔”是可以搞定一切，但对于系统设计、维护来说代价也是不小的。

后来在一个项目中我放弃了原有的51单片机，使用了一款ARM7处理器。随着反复查看ARM处理器手册并在项目开发过程中对ARM7处理器的逐步了解，我意识到实现一个简单的操作系统任务调度功能并没有想象的那么困难，原以为实现操作系统任务调度功能需要深入了解编译器的知识，但我发现只需要使用标准的C语言、一些汇编语言和处理器知识就可以实现。

整理一下我当时所处的境况：

(1) 迫切需要一个适合小系统的嵌入式操作系统，但没有合适的。

(2) 了解了嵌入式操作系统的一些应用层概念。

(3) 掌握了ARM7处理器的结构、C语言和汇编语言知识。

(4) 找不到一本可以较好地介绍操作系统内核调度的书籍，希望能让更多的人了解嵌入式操作系统内核调度的基本原理，而不仅仅是如何使用。

既然如此，那么我们就开始一起编写具有任务调度功能的两个嵌入式操作系统内核——Wanlix和Mindows。

Wanlix是一个内核非常小的嵌入式操作系统，只有1 KB左右(大小与编译器、编译选项等因素有关)，功能也非常少，只提供任务切换功能，而且需要主动调用函数切换任务。但是，它确实可以实现任务调度功能，最难能可贵的是它的小巧，非常适合资源特别少但又需要任务切换的小项目。在这个源码开放的时代，Linux、Unix遍地生根，考虑到我姓王，因此我将它叫作Wanlix。



Windows是一种大型PC机操作系统，它是分时操作系统，是PC机通用操作系统，而我们将要编写的Mindows则是一种小型操作系统，是实时的，是用在嵌入式设备上的实时嵌入式操作系统，一切都是与Windows相反的，因此这个操作系统就叫作Mindows！Mindows具有较多的功能，支持多优先级任务抢占，可以实现任务实时切换，具有信号量、队列等机制，并且可以裁剪掉不需要的功能。

本手册只讲解Wanlix和Mindows操作系统的内核调度，至于其他的文件系统、内存管理等，由于内容过于庞杂，本人没有能力也没有精力实现，只是在某些章节会作一些浅显的介绍。这两个操作系统提供了源码，有兴趣的读者可以在此基础上自己试着实现其他功能，与他人讨论、交流，共同提高，在此我为大家提供3个网址：

主力网站：www.ifreecoding.com

网站论坛：bbs.ifreecoding.com

新浪博客：blog.sina.com.cn/ifreecoding

大家可以登录这些网站下载本手册的源代码、演示录像及参考资料、开发工具等，并可在论坛进行交流。

本手册主要由王奇负责编写，湖南大学谷志茹博士、大连海事大学姜日凡博士参与了部分章节的编写工作。感谢在华为工作时的同事郑朝晖、时峰、马继彬、潘玉园、赵峰、贾国昌、何斌、王继松、赵虎、李佳、张婷婷等对本手册提出了许多宝贵的意见，

感谢我的主管张键、袁震等给予工作上的支持，感谢姜英华、曹昌平、喻妍等参与了本书部分文字录入及校对工作。

最后，向那些无偿提供知识的兄弟姐妹们表示敬意！在编写操作系统过程中，确实遇到了一些问题，正是你们贡献出的宝贵经验才让我得以完成此手册的编写工作。

作　者

2015 年 1 月

# 前　言

现在有很多介绍操作系统的书籍，介绍操作系统的概念、原理以及用法等，这些书籍对读者学习操作系统有一定的帮助，但也会有不足之处。我一直认为，介绍一种技术的书籍，首先要以最简单的方式让读者明白原理，“哦，原来是这么回事。”然后，再结合例子加以演示，最好可以让读者亲自操作，让读者明白，“哦，原来这么用就可以了。”

一些介绍操作系统的书籍介绍的内容非常多，其中每一个功能都可以写成一本书来介绍，因此，读这样的书只能大概了解，无法深入本质；或者这些书会假设你具备了非常多的知识，否则你根本就不知道这书是在讲什么，如果你是希望入门的读者，那么这个要求就太高了。

本手册正好相反，介绍的内容不多，你看不到操作系统多种功能的介绍，当然，阅读本手册你也会学习到从其他书籍学习不到的知识。本手册只介绍嵌入式操作系统最核心的功能——任务调度功能，非常适合入门学习。这不需要你具备很多有关操作系统的知识，你只要具有程序员的基本功就可以了，只需要你会用 C 语言，如果对汇编语言和处理器也有那么一点了解就更好了，剩下的事情就交给我，我将和你一起将 C 语言和一小部分汇编语言组装成操作系统程序，我们一起运行这段程序，一步步实现与操作系统任务调度有关的多个功能。本手册的目的不是推广手册中的操作系统商用，而是让读者能更好地了解操作系统的基本原理，能够灵活应用。

本手册将按照下面的顺序编写：

- 第 1 章举例说明不使用操作系统编程会遇到的困难，然后介绍操作系统的分类、功能。
- 第 2 章介绍编写操作系统任务调度所需要具备的基本知识，介绍本手册所使用的几条汇编指令以及所使用的处理器结构。
- 第 3 章开始编写操作系统代码，介绍任务调度原理，并使用 C 语言和汇编语言编程，实现 2 个固定任务间的切换，之后再扩展到任意多个任务间的切换，最后再增加一些基本功能，完成 Wanlix 操作系统的编写。作为本章的结束，将使用 Wanlix 操作系统编写一个交通红绿灯控制系统。
- 第 4 章开始编写实时嵌入式操作系统 Mindows，介绍实时嵌入式操作系统的调度原理，编写任务钩子、任务删除、信号量、队列等最基本的功能程序。作

为本章的结束，将使用 Mindows 操作系统编写一个俄罗斯方块游戏。

- ◆ 第 5 章继续完善 Mindows 操作系统的功能，增加任务优先级继承、任务轮转调度、栈异常打印、栈统计、CPU 占有率统计等功能，这些功能都是可裁剪的。作为本章的结束，将使用 Mindows 操作系统编写一个嵌入式软件平台。
- ◆ 第 6 章将分别在 Mindows、μCos、Windows 和 Linux 操作系统下编写任务调度程序，采用相同的程序结构。
- ◆ 第 7 章简单介绍进程机制，并使用线程模拟多进程。

本手册每增加一个功能，会先对该功能的原理作一番介绍，然后进行结构设计并编写代码实现该功能，最后使用一个例子演示该功能。例子可以在开发板上运行，通过串口和 LCD 显示屏观察该功能的运行效果。在一些章节的最后，还会编写几个嵌入式应用程序，应用编写的操作系统。

通过本手册，不但可以了解嵌入式操作系统的原理并一步步编码实现它，还可以通过本手册中丰富的例子学习操作系统的使用方法，学习在一个项目中如何设计和应用它。

另外，需要说明，我在编写本手册时参考的是其他操作系统的应用层功能，然后自己再反过来设计并实现操作系统内核层功能，因此本手册所实现的操作系统任务调度功能与其他操作系统在细节上可能会有些差异。



由于本人能力有限，工作之余写书，其中疏忽、错误在所难免。编码不易，写书不易，如有问题请读者反馈给我，我将尽力修正，还请大家多多支持！

声明：本人提供 Wanlix 和 Mindows 操作系统的源码，可以免费使用，可以到 www. ifreecoding. com、bbs. ifreecoding. com、blog. sina. com. cn/ifreecoding 网站获取相关的资料，但如果因使用本手册中的代码而带来损失，本人将不承担责任。

# 目　录

# 第1章

# 操作系统基础知识

有很多嵌入式设备的资源非常少，几十 K 的 ROM，几 K 的 RAM，实现的功能非常简单，其上运行的程序也非常简单，只需要在一个死循环里按照固定的顺序周而复始地运行就可以了。这种小型系统设备不需要操作系统，也几乎没有合适的操作系统可以运行在资源如此之少的设备上。

当处理器资源越来越丰富，要实现的功能越来越多的时候，我们就会发现软件所做的工作不再只是简单地重复一件事情了，它需要及时地响应外部的输入信号，及时协调自己内部的运行状态，不能只是自顾自地完成自己的工作。它需要不断地与外界交互，及时满足外界的需求，并根据这些需求及时调整自己的状态。

本章将从几个例子开始，说明在没有操作系统的情况下软件编程的不便之处，以帮助读者理解为何要使用操作系统的任务调度功能，并通过介绍操作系统的相关概念使读者对操作系统有一个基本了解，并基于这些知识编写一个非常简单、小巧的非抢占式操作系统内核——Wanlix。

## 1.1 为什么要使用操作系统

计算机是由一大堆硬件组成的，程序员们为它开发出了各式各样的程序，在这些硬件上面运行这些程序就能实现各种丰富多彩的功能。但这些程序是依赖于操作系统才能运行的，如果没有操作系统的支持，这些程序将一无是处，计算机也将是一堆废铜烂铁。

以 PC 机为例，它所做的任何一件事都与操作系统密不可分。计算机启动时，操作系统会对 CPU、主板、内存、显卡等设备进行初始化，只有经过这些处理，计算机才具有使用的可能性。我们从显示界面与计算机进行互动，这是操作系统将计算机内部的各种信息以图形这种容易理解的方式显示给我们，与我们进行沟通；在计算机上插上 USB 设备时，操作系统会帮助我们安装它的驱动程序并管理这个设备；在计算机上存储文件时，操作系统会自动在硬盘上帮我们寻找一个合适的位置并写入文件数据；上网所看到的信息都需要操作系统经过复杂的协议栈传送过来。操作系统还有很多很多功能，有很多是隐藏在后台运行的，我们甚至都感觉不到它的存在，总之，操作系统用于管理计算机的软硬件，保证计算机能正常工作。

计算机如果没有操作系统的支持，那么一个希望在计算机上运行的应用程序就需要自己来管理计算机，这些工作都需要由编写该程序的程序员来完成，那么程序员的工作将不只是编写一个应用程序这么简单，更多的工作将是编写与应用程序无关的管理计算机的程序。任何一个应用程序都需要重复着这样的工作，如果没有操作系统对计算机各种资源的统一管理，这些程序在运行时就会产生冲突，无法在计算机上共同运行。

对于小型嵌入式设备来说，它的功能有限，不会使用到很多功能，它很可能没有显示界面，没有内存管理，没有文件管理，没有设备管理，没有协议栈，没有……也许只有一个程序员为之开发程序，也只有一个程序在这个设备上运行。这样的嵌入式设备除了需要为外设编写驱动程序外，剩下的工作主要就是编写应用程序实现产品的功能。这么看来，这种小型嵌入式设备似乎没有使用操作系统的必要。但不要忘了，操作系统除了上述功能外，其最基本的功能是软件调度功能，这在小型嵌入式设备中仍是非常重要的。

对于功能特别简单的小型嵌入式设备来说，一般不需要使用操作系统，只需要设计一个 while 死循环就可以实现所有的功能。这种小系统一般没有复杂的外部输入，例如电子表，外部输入只有调节时间的按钮，软件的主要功能也只是读取定时器的数值并显示出来。以伪码的形式描述一个这样的软件结构：

```
int main(void)
{
    while(1)
    {
        1.判断按键输入并执行相关操作
        2.读取定时器数值
        3.刷新液晶屏显示时间
    }
}
```

这个小系统的运行过程在大部分时间里是不需要依赖外界输入的，只需要按照软件设定好的顺序周而复始地运行就可以实现所有功能。但如果系统功能再复杂一些，使用上述的软件结构就会显得力不从心。比如说我以前做过一个控制步进电机的小系统，这个系统需要实时接收、处理、返回上位机的命令，软件设计时只使用了一个 while 循环结构，主要分为 3 个部分，分别是消息接收功能、消息处理功能和消息发送功能，先由消息接收功能接收消息，再由消息处理功能处理消息，最后由消息发送功能发送返回的消息，软件结构如下所示：

```
00001   int main(void)
00002   {
00003       while(1)
00004       {
```

```
00005        接收消息
00006        {
00007            if(接收到一条消息数据)
00008            {
00009                对数据进行校验
00010
00011                if(数据正确)
00012                {
00013                    置处理消息标志
00014                }
00015            }
00016        }
00017
00018        处理消息
00019        {
00020            if(有需要处理的消息)
00021            {
00022                执行消息
00023
00024                生成返回消息
00025
00026                置返回消息标志
00027            }
00028        }
00029
00030        发送消息
00031        {
00032            if(有需要发送的消息)
00033            {
00034                发送消息
00035            }
00036        }
00037    }
00038 }
```

这个软件系统由中断收发消息中的每个字节数据，当接收到一条完整消息时中断就会将接收数据标志置为已收到消息状态，软件在 while 循环里只需要判断该标志就可以了。如果不是已收到消息状态，那么消息接收功能、消息处理功能、消息发送功能就不会执行；如果是已接收到消息状态，那么消息接收功能就开始执行，然后触发消息处理功能执行，再触发消息发送功能执行，完成对一条消息的处理。

这其中有一条消息需要控制步进电机连续转动几分钟，在处理这条消息时，软件

启动 PWM 驱动电机转动，软件会停留在 00022 行不断查询电机转动的状态，直到几分钟后电机转动完成，软件生成返回消息，再将返回消息由消息发送功能发送给上位机，完成对该条消息的处理过程。但这样一来，软件在处理该消息过程中就无法处理其他消息了，如果希望还能同时处理其他消息，就需要更改此软件结构，要么在 00022 行增加接收、处理、发送其他消息的过程，要么在 00022 行增加标志，临时退出该消息的处理过程，然后重新进入 while 循环里接收、处理、发送其他消息，并根据标志对控制步进电机的消息另作处理，或者使用其他方法。但不管怎么改，都会使软件结构支离破碎，不利于编码和维护。

一个理想的解决方法是有多个消息处理单元，并且程序既可以在消息处理单元运行时临时退出，又可以回到消息处理单元接着退出的地方继续运行，这样就可以在控制电机与处理其他消息之间切换而又不破坏上述程序结构，如图 1.1 所示。

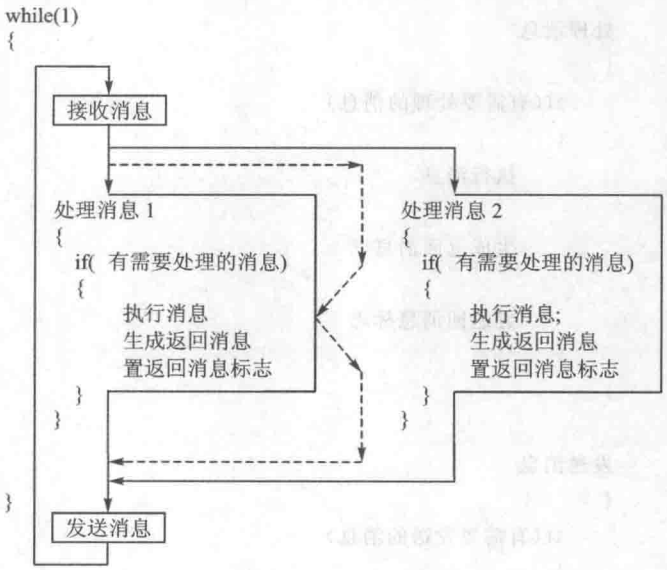


**图 1.1　多个消息处理单元执行过程**

当接收到控制电机的消息时，软件在“处理消息 1”中触发 PWM 开始驱动电机转动，在电机转动期间为了能接收、处理、发送其他消息，软件可以临时退出等待电机转动完成的过程。如图 1.1 中虚线在“执行消息”环节临时退出，从消息处理环节直接跳转到消息发送环节，但由于没有可发送的消息，便经由 while 循环通过“接收消息”、“处理消息 2”和“发送消息”处理其他消息，在下个 while 循环则可以在“接收消息”之后沿虚线直接回到“处理消息 1”中的“执行消息”环节，接着上次退出的地方继续运行。软件就在 while 循环中不断在 2 个消息处理环节中切换，实现同时处理多个消息的功能，这样做不但没有破坏整个程序的结构，而且能够同时处理多个消息。

但这样做存在着一个明显的问题，那就是 C 语言的运行方式决定了其无法临时退出一个函数然后又从退出点进入继续运行。C 语言是以函数为单位实现功能的，

一个完整的功能会由多个函数共同完成，这些函数只能是一个调用一个串行地执行。在函数中间临时退出而后又能进入的运行方式属于操作系统任务调度的基本概念，这个功能可以使用操作系统实现。使用操作系统的任务调度功能对函数加以管理，可以使得使用C语言方便地编写出同时运行多个函数的程序。

从操作系统中函数运行的方式来说，操作系统是对函数运行进行管理的系统，它可以在一个函数还没有运行完时就转而去执行另外一个函数，并且还可以恢复到原来的函数继续运行，这样就可以根据需要及时调整到需要运行的函数来实现其功能。下面以大家熟悉的Windows为例来说明任务调度功能。Windows上运行了很多软件，有办公的、看电影的、玩游戏的等等，太多了。想过没有，它们是如何做到同时运行的？这些应用程序从宏观上看是在同一台计算机上同时运行的，但从微观上看它们仍是串行运行的。计算机的CPU(不考虑多核CPU)每一时刻只能执行一条指令(不考虑流水线)，执行很短一段时间之后，操作系统会进行任务调度，CPU又去运行另外一个应用程序的指令，周而复始地运行。由于CPU的速度特别快，因此每个应用程序在很短的时间就可以运行很多次，以人的感觉根本就感觉不到CPU在各个应用程序之间切换运行，只能感觉到每个程序都是在连续运行，因此就觉得计算机上的多个应用程序是在同时运行。这就像看电影一样，由于影片的刷新频率快过了人眼的可分辨频率，因此就觉得电影是在连续播放。这就是操作系统的一个非常重要的功能——任务调度功能。

需要说明一点，上述描述中的“任务”其实是“进程”，而本手册中实现的“任务”对应于“线程”。本手册主要讲述线程的概念，在本手册最后部分会对“进程”作一个简单的介绍。

对于一个完整的操作系统来说，它会有很多功能，但对于小型嵌入式设备来说，它也许并不需要操作系统具有这么多的功能，多余的功能对它来说是一种负担，会占用并不富裕的系统资源。但无论如何，操作系统的任务调度功能是必备的，它是操作系统赖以生存最核心的功能，本手册将和你一起从零开始，一步步编写操作系统的任务调度功能。

## 1.2 操作系统的嵌入性和实时性

从不同的角度来看，操作系统可以有很多种划分，比如按与用户对话的界面划分，可分为命令行界面操作系统和图形界面操作系统；按支持用户数的多少，可以分为单用户和多用户操作系统；按功能可以分为嵌入式操作系统和PC机通用操作系统；按调度的方式可分为分时操作系统和实时操作系统。操作系统种类繁多，很难用单一标准统一分类，这里不再详细介绍各种操作系统。

本手册所编写的操作系统是应用于嵌入式设备之上的，突出的特性在于嵌入性和实时性，这里将着重介绍一下“嵌入式”和“实时”等概念。