HZ BOOKS
华章科技

深入剖析RT-Thread内核实现，详解各个组件如何使用。
由浅入深，基于野火STM32全系列开发板，提供完整的源代码，极具可操作性。

野火
野火嵌入式系列

RT-Thread
中国自主物联网操作系统

RTOS

RT-Thread内核实现与应用开发实战指南

基于STM32

刘火良 杨森 编著

机械工业出版社
China Machine Press

野火

野火嵌入式系列

RT-Thread

中国自主物联网操作系统

RTOS

RT-Thread内核实现与应用开发实战指南

基于STM32

刘火良 杨森 编著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

RT-Thread 内核实现与应用开发实战指南：基于 STM32/ 刘火良，杨森编著．—北京：机械工业出版社，2019.1
（电子与嵌入式系统设计丛书）

ISBN 978-7-111-61366-4

I. R… II. ① 刘… ② 杨… III. 微控制器 - 系统开发 - 指南 IV. TP332.3-62

中国版本图书馆 CIP 数据核字（2018）第 263077 号

RT-Thread 内核实现与应用开发实战指南：基于 STM32

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码：100037）
责任编辑：赵亮宇　　责任校对：殷　虹
印　　刷：北京市兆成印刷有限责任公司　　版　　次：2019 年 1 月第 1 版第 1 次印刷
开　　本：186mm × 240mm　1/16　　印　　张：26
书　　号：ISBN 978-7-111-61366-4　　定　　价：99.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88379426　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzit@hzbook.com

推 荐 序

从2006年走来，RT-Thread已经走过了十多个年头。早年就和野火相识于网络，记得那个时候野火电子才开始做开发板，RT-Thread也只是国内一个小众的RTOS。直至2018年年中，野火来上海我俩才有缘相见，互道年轻。

2017年年底，欣闻野火有写作RT-Thread相关书籍的计划，非常欣喜、感谢。这将是介绍RT-Thread的首本书籍，在RT-Thread的基础生态推广方面有着举足轻重的作用！不仅如此，作为RT-Thread的官方合作伙伴，野火电子也在其全系列的STM32开发板上适配了RT-Thread例程并配套了手把手的教程。

野火的这本书，第一部分由0到1，从底层的汇编开始，一步一步构造出一个RT-Thread操作系统内核，向大家揭示了任务如何定义、如何切换，也讲解了任务的延时如何实现、如何支持多优先级、如何实现定时器以及如何实现时间片等RT-Thread操作系统的核心知识点；第二部分则讲解了RT-Thread内核设施的应用，使得大家学习和使用RT-Thread不再困难。

整本书由浅入深，层层叠加，与初学者的入门路径完全吻合，是学习RT-Thread物联网操作系统的不二之选。同时，整本书也兼顾深度，对想要了解操作系统内核原理的读者来说，也非常值得一读。

RT-Thread 创始人

熊谱翔

RT-Thread

前　言

如何学习本书

本书是首本系统讲解 RT-Thread 的中文书籍，共分为两个部分。第一部分重点讲解 RT-Thread 的原理实现，从 0 开始，不断迭代，教你把 RT-Thread 的内核写出来，让你彻底学会线程是如何定义的、系统是如何调度的（包括底层的汇编代码讲解）、多优先级是如何实现的等操作系统的最深层次的知识。当你拿到本书开始学习的时候，你一定会惊讶，原来 RTOS 的学习并没有那么复杂，反而是那么有趣；原来自己也可以写 RTOS，成就感立马爆棚。

当彻底掌握第一部分的知识之后，再学习其他 RTOS，可以说十分轻松。纵观现在市面上流行的几种 RTOS，它们的内核实现差异不大，只需要深入研究其中一种即可，没有必要对每一种 RTOS 都深入地研究源码，但如果时间允许，看一看也并无坏处。第二部分重点讲解 RT-Thread 的移植、内核中每个组件的应用，比起第一部分，这部分内容掌握起来应该比较容易。

全书内容循序渐进，不断迭代，尤其在第一部分，前一章是后一章的基础，必须从头开始阅读，不能进行跳跃式的阅读。在学习时务必做到两点：一是不能一味地看书，要把代码和书本结合起来学习，一边看书，一边调试代码。如何调试代码呢？即单步执行每一条程序，看程序的执行流程和执行的效果与自己所想的是否一致。二是在每学完一章之后，必须将配套的例程重写一遍（切记不要复制，即使是一个分号，但可以照书录入），做到举一反三，确保真正理解。在自己写的时候难免错误百出，要珍惜这些错误，好好调试，这是你提高编程能力的最好机会。记住，程序不是一气呵成写出来的，而是一步一步调试出来的。

本书的编写风格

本书第一部分主要以 RT-Thread Nano 3.0.3 官方源码为蓝本，抽丝剥茧，不断迭代，教你如何从 0 开始把 RT-Thread 内核写出来。书中涉及的数据类型、变量名称、函数名称、文件名称、文件存放的位置都完全按照 RT-Thread 官方的方式来实现。学完这本书之后，你可

以无缝地切换到原版的 RT-Thread 中使用。要注意的是，在实现的过程中，某些函数中会去掉一些形参和冗余的代码，只保留核心的功能，但这并不会影响我们学习。

本书第二部分主要介绍 RT-Thread 的移植和内核组件的使用，不会再去深入讲解源码，而是着重讲解如何应用，如果对第一部分不感兴趣，也可以跳过第一部分，直接进入第二部分的学习。

本书还有姊妹篇——《FreeRTOS 内核实现与应用开发实战指南：基于 STM32》，两本书的编写风格、内容框架和章节命名与排序基本一致，语言阐述类似，且涉及 RTOS 抽象层的理论部分也相同，不同之处在于 RTOS 的实现原理、内核源码的讲解和上层 API 的使用，这些内容才是重点部分，是读者学习的核心。例如，虽然两本书的第一部分的章节名称基本类似，但内容不同，因为针对的 RTOS 不一样。其中，关于新建 RT-Thread 工程和裸机系统与多线程（任务）系统的描述属于 RTOS 抽象层的理论部分，不具体针对某个 RTOS，所以基本一样。第二部分中，对于什么是线程（任务）、阻塞延时和信号量的应用等 RTOS 抽象层的理论讲解也基本类似，但是具体涉及这两个 RTOS 的原理实现和代码讲解时则完全不同。

如果读者已经学习了其中一本书，再学习另外一本的话，那么涉及 RTOS 抽象层的理论部分可跳过，只需要把精力放在 RTOS 内核的实现和源码 API 的应用方面。因为现有的 RTOS 在理论层基本都是相通的，但在具体的代码实现上各有特点，所以可以用这两本书进行互补学习，掌握了其中一本书的知识，再学习另外一本书定会得心应手，事半功倍。

本书的参考资料和配套硬件

关于本书的参考资料和配套硬件的信息，请参考本书附录部分。

本书的技术论坛

如果在学习过程中遇到问题，可以到野火电子论坛 www.firebbs.cn 发帖交流，开源共享，共同进步。

鉴于水平有限，书中难免有错漏之处，热心的读者也可把勘误发送到论坛上以便改进。祝你学习愉快，RT-Thread 的世界，野火与你同行。

引　言

为什么学习 RTOS

当我们进入嵌入式系统这个领域时，首先接触的往往是单片机编程，单片机编程又首选 51 单片机来入门。这里说的单片机编程通常都是指裸机编程，即不加入任何 RTOS（Real Time Operation System，实时操作系统）的程序。常用的 RTOS 有国外的 FreeRTOS、μC/OS、RTX 和国内的 RT-Thread、Huawei LiteOS、AliOS-Things 等，其中，开源且免费的 FreeRTOS 的市场占有率最高。如今国产的 RT-Thread 经过 10 余年的迅猛发展，在国产 RTOS 中占据鳌头。

在裸机系统中，所有的程序基本都是用户自己写的，所有的操作都是在一个无限的大循环里面实现。现实生活的很多中小型电子产品中用的都是裸机系统，而且能够满足需求。但是为什么还要学习 RTOS 编程，并会涉及一个操作系统呢？一是因为项目需求，随着产品要实现的功能越来越多，单纯的裸机系统已经不能完美地解决问题，反而会使编程变得更加复杂，如果想降低编程的难度，可以考虑引入 RTOS 实现多线程管理，这是使用 RTOS 的最大优势；二是出于学习的需要，必须学习更高级的技术，实现更好的职业规划，为将来能有更好的职业发展做准备，而不是一味拘泥于裸机编程。作为一个合格的嵌入式软件工程师，学习是永远不能停歇的，时刻都得为将来做准备。书到用时方恨少，希望当机会来临时，读者不要有这种感觉。

为了帮大家厘清 RTOS 编程的思路，本书会在第 2 章简单地分析这两种编程方式的区别，这个区别被笔者称为“学习 RTOS 的‘命门’”，只要掌握这一关键内容，以后的 RTOS 学习可以说是易如反掌。在讲解这两种编程方式的区别时，我们主要讲解方法，不会涉及具体的代码编程，即主要还是通过伪代码来讲解。

如何学习 RTOS

裸机编程和 RTOS 编程的风格有些不一样，而且有很多人说学习 RTOS 很难，这就导致

想要学习 RTOS 的人一听到 RTOS 编程就在心里忌惮三分，结果就是“出师未捷身先死”。

那么到底如何学习 RTOS 呢？最简单的方法就是在别人移植好的系统上，先看看 RTOS 中 API 的使用说明，然后调用这些 API 实现自己想要的功能即可，完全不用关心底层的移植，这是最简单、快速的入门方法。这种方法有利有弊，如果是做产品，好处是可以快速地实现功能，将产品推向市场，赢得先机；弊端是当程序出现问题时，因对 RTOS 不够了解，会导致调试困难。如果想系统地学习 RTOS，那么只会简单地调用 API 是不可取的，我们应该深入学习其中一款 RTOS。

目前市场上现有的 RTOS，其内核实现方式差异不大，我们只需要深入学习其中一款即可。万变不离其宗，即使以后换到其他型号的 RTOS，使用起来自然也是得心应手。那么，如何深入地学习一款 RTOS 呢？这里有一个非常有效但也十分难的方法，就是阅读 RTOS 的源码，深入研究内核和每个组件的实现方式。这个过程枯燥且痛苦，但为了能够学到 RTOS 的精华，还是很值得一试的。

市面上虽然有一些讲解 RTOS 源码的图书，但如果基础知识掌握得不够，且先前没有使用过该款 RTOS，那么只看源码还是会非常枯燥，并且不能从全局掌握整个 RTOS 的构成和实现。

现在，我们采用一种全新的方法来教大家学习一款 RTOS，既不是单纯地介绍其中的 API 如何使用，也不是单纯拿里面的源码一句句地讲解，而是从 0 开始，层层叠加，不断完善，教大家如何把一个 RTOS 从 0 到 1 写出来，让你在每一个阶段都能享受到成功的喜悦。在这个 RTOS 实现的过程中，只需要具备 C 语言基础即可，然后就是跟着本书笃定前行，最后定有所成。

选择什么 RTOS

用来教学的 RTOS，我们不会完全从头写一个，而是选取目前国内很流行的 RT-Thread 为蓝本，将其抽丝剥茧，从 0 到 1 写出来。在实现的过程中，数据类型、变量名、函数名称、文件类型等都完全按照 RT-Thread 里面的写法，不会再重新命名。这样学完本书之后，就可以无缝地过渡到 RT-Thread 了。

RT-Thread 简介

RT-Thread 的版权属于上海睿赛德电子科技有限公司，它于 2006 年 1 月首次发布，初始版本号为 0.1.0，经过 10 余年的发展，如今主版本号已经升级到 3.0，累计开发者人数达数

百万，在各行各业产品中装机量达到了 2000 多万，占据国产 RTOS 的鳌头。

RT-Thread 是一款开源免费的实时操作系统，遵循的是 GPLv2+ 许可协议。这里所说的开源，指的是你可以免费获取 RT-Thread 的源代码，而且当你的产品使用了 RT-Thread 且没有修改 RT-Thread 内核源码时，你的产品的全部代码都可以闭源，但是当你修改了 RT-Thread 内核源码时，就必须将修改的这部分开源，反馈给社区，其他应用部分不用开源。无论是个人还是公司，都可以免费使用 RT-Thread。

RT-Thread 的意义

不知你是否发现，在 RTOS 领域，我们能接触到的实时操作系统基本都来自国外，很少见到有国内厂家的产品。从早年很流行的 μC/OS，到如今市场占有率很高的 FreeRTOS，到获得安全验证非常多的 RTX（KEIL 自家的），再到盈利能力领先的 ThreadX，均来自国外。可在十几年前，在中国，出现了一个天赋异禀、倔强不屈的极客——熊谱翔。他编写了 RT-Thread 初代内核，并联合中国开源社区的极客不断完善，推陈出新，经过十几年的发展，如今占据国产 RTOS 的鳌头，且每年能递增数十万名开发者，加上如今 AI 和物联网等技术发展的机遇，让 RT-Thread 有“一统江湖”之势，从 2018 年完成 A 轮数百万美元的融资就可以看出，在未来不出 5 年，RT-Thread 将是你学习和做产品的不二之选。

那么 RT-Thread 的意义究竟是什么？ RT-Thread 来自中国，让我们看到了国内的技术开发者也能写出如此优秀的 RTOS，技术方面并不逊色于其他国家。以我们 10 多年从事电子行业的经验来看，RT-Thread 无疑增强了我们在这一领域的自信，这是我们认为的 RT-Thread 的最大意义。当然，作为一款国产的物联网操作系统，RT-Thread 简单易用、低功耗设计、组件丰富等特性也将让其大放异彩。野火，作为一个国内嵌入式领域的教育品牌，能为国产 RTOS 出一份力也是我们的荣幸，希望本书能够帮助大家快速地入门并掌握 RT-Thread。

目　录

第一部分

从 0 到 1 教你写 RT-Thread 内核

本部分以 RT-Thread Nano 为蓝本，抽丝剥茧，不断迭代，教大家如何从 0 开始写 RT-thread。这一部分的重点在于 RTOS 的实现过程，当学完这部分内容之后，再来重新使用 RT-Thread 或者其他 RTOS 将会得心应手，不仅知其然，而且知其所以然。在源码实现的过程中，涉及的数据类型、变量名称、函数名称、文件名称以及文件的存放目录都会完全按照 RT-Thread 的来实现，一些不必要的代码会被剔除，但这并不会影响我们理解整个操作系统的功能。

本部分中每一章都以前一章为基础，环环相扣，逐渐揭开操作系统的神秘面纱，读起来会有一种令人豁然开朗的感觉。如果你把代码都自己输入一遍，得到的仿真效果如果与书中的一样，那从心里油然而生的成就感简直就要爆棚，恨不得一下子把这本书读完，真是让人看了还想看，读了还想读。

第 1 章
新建 RT-Thread 工程——软件仿真

在开始编写 RT-Thread 内核之前，我们先新建一个 RT-Thread 工程，Device 选择 Cortex-M3（Cortex-M4 或 Cortex-M7）内核的处理器，调试方式选择软件仿真，然后我们开始一步一步地教大家把 RT-Thread 内核写出来，让大家彻底明白 RT-Thread 的内部实现和设计思想，最后把 RT-Thread 移植到野火 STM32 开发板上。最后的移植其实已经非常简单，只需要换一下启动文件并添加 bsp 驱动即可。

1.1 新建本地工程文件夹

在开始新建工程之前，我们先在本地计算机中新建一个文件夹用于存放工程。可将文件夹命名为“新建 RT-Thread 工程——软件仿真”（名字可以随意设置），然后在该文件夹下新建各个文件夹和文件，有关这些文件夹的包含关系和作用如表 1-1 所示。

表 1-1 工程文件夹根目录下的文件夹及作用

文件夹名称	文 件 夹	文件夹作用
Doc	—	用于存放整个工程的说明文件，如 readme.txt。通常情况下，要对整个工程实现的功能、如何编译、如何使用等做一个简要的说明
Project	—	用于存放新建的工程文件
rtthread/3.0.3	bsp	存放板级支持包，暂时为空
	components\finsh	存放 RT-Thread 组件，暂时为空
	include	存放头文件，暂时为空
	include\libc	
	libcpu\arm\cortex-m0	存放与处理器相关的接口文件，暂时为空
	libcpu\arm\cortex-m3	
	libcpu\arm\cortex-m4	
	libcpu\arm\cortex-m7	
	src	存放 RT-Thread 内核源码，暂时为空
User		存放 main.c 和其他用户编写的程序，第一次使用 main.c 时需要用户自行创建