调研报告

小组成员

- 陆子睦
- 黄与进
- 刘津畅
- 唐星
- 杨涛

项目简介

随着智能硬件、物联网行业的迅猛发展, 嵌入式系统在各个领域都得到了广泛的应用。嵌入式操作系统可以帮助嵌入式设备更好地完成任务的调度，从而更高效地完成任务。然而，一开始由于嵌入式设备内存较少，一些主流的嵌入式操作系统如FreeRTOS并没有内存管理单元。随着时代的发展，内存的成本越来越低，嵌入式设备也在向着内存增加的方向发展。所以，按照当前的趋势，嵌入式操作系统也应该有自己的内存管理单元。嵌入式系统内存配置较小,不能采用一般桌面系统的内存管理方式,选取合适的内存管理策略在嵌入式系统设计中起着重要的作用。于是，我们小组决定为FreeRTOS编写一个内存管理单元，从而让其有更强大的任务调度能力。由于Rust是一门高效而且安全的语句，我们将使用Rust来进行编写。

而有了内存管理单元之后，嵌入式内核就有可能去以模块化的形式去调用Linux内核，从而实现微内核调用宏内核。这种实现方式可以有效地降低功耗，从而实现性能的提升。因此，我们小组计划在实现了FreeRTOS的内存管理单元之后，再实现FreeRTOS模块化方式调度Linux的功能。

然后，我们将把FreeRTOS移植到树莓派上，并进行性能的测试。

项目背景

FreeRTOS

简介：

FreeRTOS是一个迷你的实时操作系统内核。作为一个轻量级的操作系统，功能包括：任务管理、时间管理、信号量、消息队列、内存管理、记录功能、软件定时器、协程等，可基本满足较小系统的需要。

由于RTOS需占用一定的系统资源(尤其是RAM资源)，只有μC/OS-II、embOS、salvo、FreeRTOS等少数实时操作系统能在小RAM单片机上运行。相对μC/OS-II、embOS等商业操作系统，FreeRTOS操作系统是完全免费的操作系统，具有源码公开、可移植、可裁减、调度策略灵活的特点，可以方便地移植到各种单片机上运行。

特点：

用户可配置内核功能

多平台的支持

提供一个高层次的信任代码的完整性

目标代码小，简单易用

遵循MISRA-C标准的编程规范

强大的执行跟踪功能

堆栈溢出检测

没有限制的任务数量

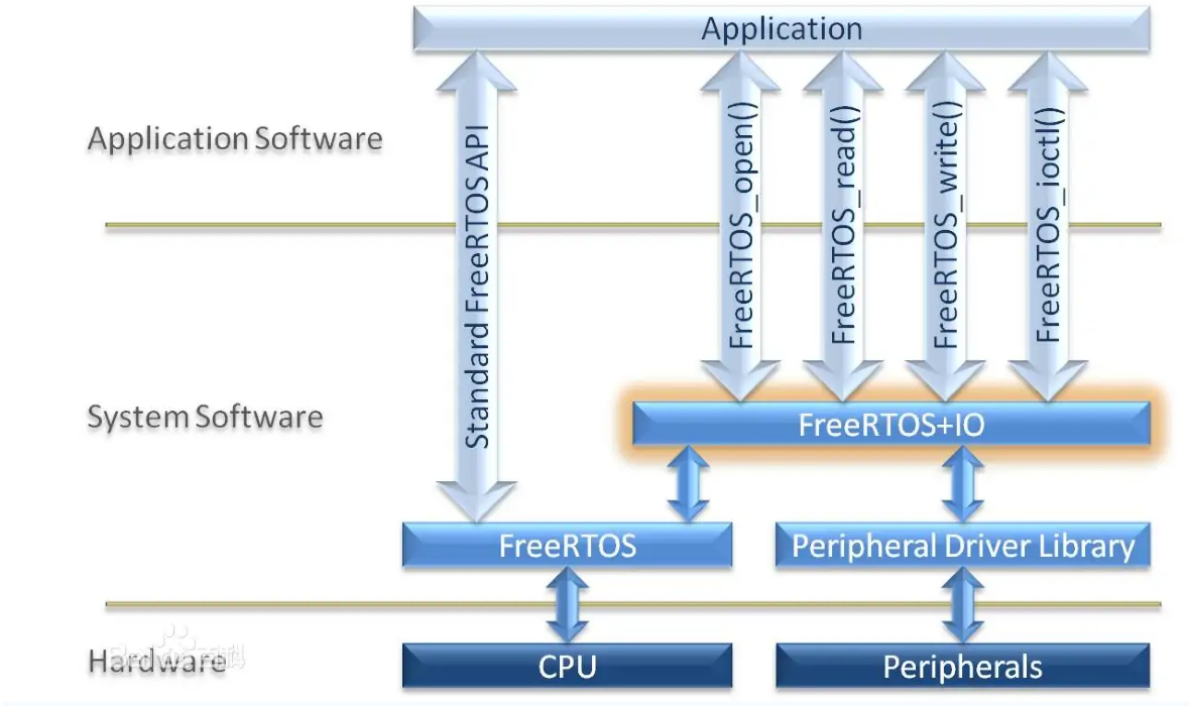
没有限制的任务优先级

多个任务可以分配相同的优先权

队列，二进制信号量，计数信号灯和递归通信和同步的任务

优先级继承

免费开源的源代码



系统功能：

作为一个轻量级的操作系统，FreeRTOS提供的功能包括：任务管理、时间管理、信号量、消息队列、内存管理、记录功能等，可基本满足较小系统的需要。FreeRTOS内核支持优先级调度算法，每个任务可根据重要程度的不同被赋予一定的优先级，CPU总是让处于就绪态的、优先级最高的任务先运行。
FreeRTOS内核同时支持轮换调度算法，系统允许不同的任务使用相同的优先级，在没有更高优先级任务就绪的情况下，同一优先级的任务共享CPU的使用时间。

FreeRTOS的内核可根据用户需要设置为可剥夺型内核或不可剥夺型内核。当FreeRTOS被设置为可剥夺型内核时，处于就绪态的高优先级任务能剥夺低优先级任务的CPU使用权，这样可保证系统满足实时性的要求；当FreeRTOS被设置为不可剥夺型内核时，处于就绪态的高优先级任务只有等当前运行任务主动释放CPU的使用权后才能获得运行，这样可提高CPU的运行效率。

前景：

在嵌入式领域，FreeRTOS是不多的同时具有实时性，开源性，可靠性，易用性，多平台支持等特点的嵌入式操作系统。目前，FreeRTOS已经发展到支持包含X86，Xilinx，Altera等多达30种的硬件平台，其广阔的应用前景已经越来越受到业内人士的瞩目。

内存管理单元（MMU）

桌面系统的微处理器大多带有存储管理单元（MMU）， 所以桌面操作系统大都使用虚拟存储器，实际存储器和程序都被分成大小相同的页面，程序运行时，只将要运行的部分页面载 入内存即可。 MMU的作用是将虚地址映射为物理地址，保护地址越界。 大多数嵌入式系统的处理器没有MMU，即使系统中含有这些硬件也没采用，因此不能使用虚拟存管理技术，只能采用实存管理，直接访问实际的物理地址。 每个任务运行前，必须为它分配足够的连续地址空间，运行时全部载入嵌入式操作系统没有内存保护，所有任务共享一个运行空间，任何 一个任务都可能破坏其它任务的代码、数据或堆栈，甚至破坏内核代码或数据结构，导致整个系统工作异常，或使系统崩溃。 由此可见，开发嵌入式系统时， 内存管理非常重要。内存如何分配和释放，才能保证内存碎片少，且不会导致内存丢失 ，每个任务的堆栈如何安排，如何保证不侵犯其它程序包括系统程序和数据的地址空间，才能保证程序不会破坏系统或其它程序的正常工作，这些都是内存管理所要考虑的问题

内存管理模块管理系统的内存资源，它是操作系统的核心模块之一。主要包括内存的初始化、分配以及释放。

嵌入式系统不同于一般的桌面系统，对内存分配有如下要求：

①快速性： 嵌入式系统对实时性的保证，要求简单、快速地分配内存。在嵌入式系统中，不可能采用通用操作系统中复杂而完善的内存分配策略。

②可靠性：内存分配的请求必须得到满足，如果分配失败可能会带来灾难性的后果。

③高效性：嵌入式系统中内存是有限、昂贵的资源，内存分配要尽可能地少浪费。

Rust编程语言

Rust是由Mozilla主导开发的通用、编译型编程语言。设计准则为“安全、并发、实用”，支持函数式、并发式、过程式以及面向对象的程序设计风格。

rust优点：

高性能

Rust 速度惊人且内存利用率极高。由于没有运行时和垃圾回收，它能够胜任对性能要求特别高的服务，可以在嵌入式设备上运行，还能轻松和其他语言集成。

可靠性

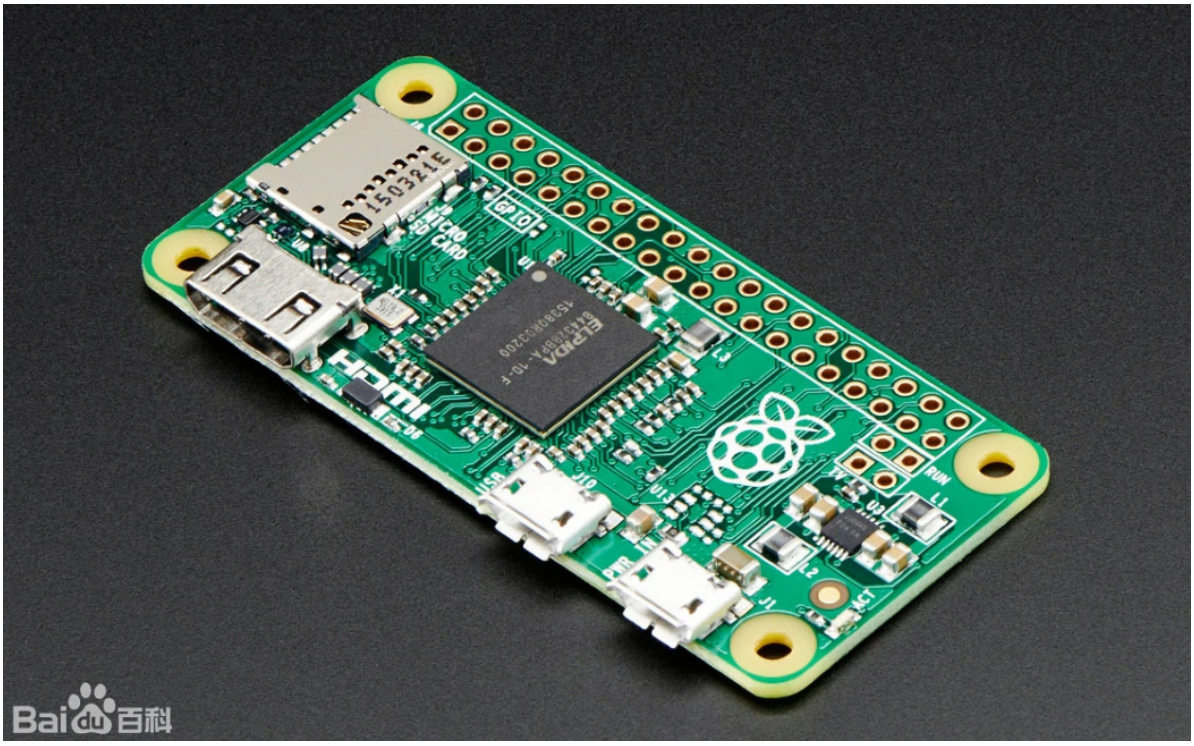
Rust 丰富的类型系统和所有权模型保证了内存安全和线程安全，让程序员在编译期就能够消除各种各样的错误。

生产力

Rust 拥有出色的文档、友好的编译器和清晰的错误提示信息， 还集成了一流的工具——包管理器和构建工具， 智能地自动补全和类型检验的多编辑器支持， 以及自动格式化代码等等。

树莓派

树莓派（英语：Raspberry Pi）英国树莓派基金会开发的微型单板计算机，目的是以低价硬件及自由软件促进学校的基本计算机科学教育。



树莓派系列计算机每一代均使用博通（Broadcom）出产的ARM架构处理器，如今生产的机型（树莓派4B）内存在2GB和8GB之间，主要TF卡作为系统存储媒体（初代使用SD卡），配备USB接口和HDMI的视频输出（支持声音输出），内置Ethernet/WLAN/Bluetooth网络链接的方式（依据型号决定），并且可使用多种操作系统。产品线型号分为A型、B型、Zero型和ComputeModule计算卡。

立项依据

随着智能硬件、物联网行业的迅猛发展, 嵌入式系统在各个领域都得到了广泛的应用, 内存管理算法作为决定系统性能的重要因素之一, 也成为了嵌入式领域一个重要的研究课题。随处嵌入式设备内存的逐渐扩大，嵌入式操作系统具有完善的内存管理单元已经成为一个发展趋势。嵌入式系统内存配置较小,不能采用一般桌面系统的内存管理方式,选取合适的内存管理策略在嵌入式系统设计中起着重要的作用。因此，我们组的题目是十分符合现实趋势的。

微内核的模块化使用也是一个十分高效的内核实现方式，它可以大大降低功耗，而这在操作系统的应用中无疑是非常理想的。而内核的模块化挂载也是一种完全可行的实现方式。所以，我们组使用微内核调用宏内核的路线是实用而且可行的。

重要性分析

为FreeRTOS添加MMU：

现代操作系统普遍采用虚拟内存机制，这需要处理器中的MMU（Memory Management Unit，内存管理单元）提供支持，下面简要介绍MMU的作用。

首先引入两个概念，虚拟地址和物理地址。如果处理器没有MMU，或者有MMU但没有启用，CPU执行单元发出的内存地址将直接传到芯片引脚上，被内存芯片（以下称为物理内存，以便与虚拟内存区分）接收，这称为物理地址（Physical Address，以下简称PA）。如果处理器启用了MMU，CPU执行单元发出的内存地址将被MMU截获，从CPU到MMU的地址称为虚拟地址（Virtual Address，以下简称VA），而MMU将这个地址翻译成另一个地址发到CPU芯片的外部地址引脚上，也就是将VA映射成PA。

如果是32位处理器，则内地址总线是32位的，与CPU执行单元相连（图中只是示意性地画了4条地址线），而经过MMU转换之后的外地址总线则不一定是32 位的。也就是说，虚拟地址空间和物理地址空间是独立的，32位处理器的虚拟地址空间是4GB，而物理地址空间既可以大于也可以小于4GB。

MMU除了做地址转换之外，还提供内存保护机制。各种体系结构都有用户模式（User Mode）和特权模式（Privileged Mode）之分，操作系统可以在页表中设置每个内存页面的访问权限，有些页面不允许访问，有些页面只有在CPU处于特权模式时才允许访问，有些页面在用户模式和特权模式都可以访问，访问权限又分为可读、可写和可执行三种。这样设定好之后，当CPU要访问一个VA时，MMU会检查CPU当前处于用户模式还是特权模式，访问内存的目的是读数据、写数据还是取指令，如果和操作系统设定的页面权限相符，就允许访问，把它转换成PA，否则不允许访问，产生一个异常（Exception）。异常的处理过程和中断类似，不同的是中断由外部设备产生而异常由CPU内部产生，中断产生的原因和CPU当前执行的指令无关，而异常的产生就是由于CPU当前执行的指令出了问题，例如访问内存的指令被MMU检查出权限错误，除法指令的除数为0等都会产生异常。

通常操作系统把虚拟地址空间划分为用户空间和内核空间，例如x86平台的Linux系统虚拟地址空间是0x00000000~0xffffffff，前 3GB（0x00000000~0xbfffffff）是用户空间，后1GB（0xc0000000~0xffffffff）是内核空间。用户程序加载到用户空间，在用户模式下执行，不能访问内核中的数据，也不能跳转到内核代码中执行。这样可以保护内核，如果一个进程访问了非法地址，顶多这一个进程崩溃，而不会影响到内核和整个系统的稳定性。CPU在产生中断或异常时不仅会跳转到中断或异常服务程序，还会自动切换模式，从用户模式切换到特权模式，因此从中断或异常服务程序可以跳转到内核代码中执行。事实上，整个内核就是由各种中断和异常处理程序组成的。总结一下：在正常情况下处理器在用户模式执行用户程序，在中断或异常情况下处理器切换到特权模式执行内核程序，处理完中断或异常之后再返回用户模式继续执行用户程序。

综上，MMU是十分重要的，它可以为操作系统内存调度提供很大方便，而且可以让内存访问更加安全。

下面来说说FreeRTOS的优点。

FreeRTOS的设计小巧且简易，整个核心代码只有3到4个C文件，为了让代码容易阅读、移植和维护，大部分的代码都是以C语言编写，只有一些函数（多数是架构特定排班副程序）采用汇编语言编写。

FreeRTOS提供许多方法以实现多线程（threads）、多作业（task）、互斥锁（mutex）、信号量（semaphore）和软件计时器（software timer），有个为低耗电应用程序提供的无嘀嗒（tick-less）模式，线程的优先权管理也有支持，此外，FreeRTOS提供了四种存储器配置的模式：

FreeRTOS中没有一些像Linux、Microsoft Windows等典型操作系统具有的先进特征，例如设备驱动程序、先进存储器管理机制、用户管理和网络管理，FreeRTOS着重在执行的简洁与速度，FreeRTOS有时会被视为是一个‘线程库’而非‘操作系统’，尽管可以找到命令行接口和类似POSIX I/O 接口的插件。

FreeRTOS实现了多线程，主程序会在规律的短时间区间内调用一个线程时计方法，这个方法会以循环制依照任务的优先级进行任务切换，一般来说，这个短时间区间介于 1/1000 秒与 1/100 秒之间，透过一个硬件时计中断来计时，但这个区间经常随着特定的应用而改变。

从FreeRTOS官网（FreeRTOS.org（页面存档备份，存于互联网档案馆））所下载到的代码包含准备用来移植或编译的配置文件和演示代码，让用户可以快速地进行应用程序设计。

综上，FreeRTOS是一个发展前景很好的嵌入式操作系统，我们对它的改进也是十分重要的。

相关工作和应用文献出处

《嵌入式实时系统内存管理策略》

《嵌入式实时系统中动态内存管理算法的设计与实现》

《嵌入式系统新型动态内存管理机制的研究》

《可生存嵌入式 OS 内存管理设计与实现》

《工业物联网中的缓冲内存管理设计与实现》

《基于TLSF算法改进的动态内存管理算法研究》

《基于线段树的高效内存管理算法及其空间优化》

VxWin White Paper JN 23058 - Kuka (yumpu.com)

https://zh.wikipedia.org/zh-cn/FreeRTOS

https://baike.baidu.com/item/FreeRTOS

树莓派 - 维基百科，自由的百科全书 (wikipedia.org)

https://www.rust-lang.org/zh-CN/

https://baike.baidu.com/item/Rust%E8%AF%AD%E8%A8%80/9502634

操作系统内存管理(思维导图详解)*hguisu*的博客-*CSDN*博客内存管理

操作系统的内存管理算法_strongerHuang的博客-CSDN博客

嵌入式学习记录：内存管理单元（MMU）介绍*Linux*编程Linux公社-Linux系统门户网站 (linuxidc.com)