# Arm64操作系统实现原理札记

# 前言

作者目前就职于国际知名通信厂商，从事ARM嵌入式领域约10年。硬件平台从C51、MSP430、STM8、STM32到ARM Cortex-M0,M3M4，再到Cortex-A53,A57的功能开发。也曾在Altera的FPGA上开发过图像缓存帧调度算法，基于X86开发过视频图像驱动芯片。软件系统从裸机代码到ucosii，再到zephyer，到vxworks，到qnx，到linux，到fuchsia。最终深耕在ARM v8的架构的芯片，微内核操作系统领域、嵌入式虚拟化领域的软件开发。

写这本书的目的一是给自己这么多年的工作做个总结；二是想将自己多年来在ARM v8架构操作系统领域的经验分享给大家。希望对大家有所帮助。

感谢自己投身于自己热爱的行业，也感谢自己坚持了下来。

# 基础篇

本篇主要讲解ARM架构操作系统的一些基本的概念，包括软件、硬件方面的概念。描述每个模块的功能职责，以及为什么需要这些模块。

## 启动原理

## 中断

## 异常

## 物理内存

## 虚拟内存

## 地址空间

## 进程管理

操作系统中，进程比线程拥有更大的资源范围。通常一个进程包含，数个线程、当前线程运行的地址空间、已映射的物理地址、当前进程的环境变量、已开启的文件信息等等。所以进程切换将会比线程切换的开销大得多。

## 线程管理

操纵系统中，线程是最小的调度单位。一个线程的正常运行至少需要使用通用寄存器、栈指针寄存器、状态寄存器。因此不同线程之间的切换，需要切换通用寄存器、栈指针寄存器和状态寄存器。由于在进行线程切换时，使用中的通用寄存器会被按序压入栈中，切换回来的时候，这些通用寄存器会被弹栈恢复，因此未使用的通用寄存器就可以不必压栈保存。

## 调度器

## 同步机制

## 驱动

# 进阶篇

主要将结合实际代码来操作理解操作系统中的概念。

Armv8相关寄存器

主要包含了31个通用寄存器和几个特殊寄存器。如SP，LR等。

cache介绍

MMU介绍

MMU全称，Memory manage unit内存管理单元，管理物理内存映射到虚拟地址空间的地址、权限等。通过MMU可以实现多进程任务之间的虚拟地址空间隔离，使得每个进程任务都满足当前虚拟地址空间中，只有它一个进程任务。

用户态与内核态

调试异常