**Melon OS**

**用户手册**

Version 0.19

目录

[Overview 4](#_Toc364559247)

[系统框图 4](#_Toc364559248)

[文件结构 5](#_Toc364559249)

[支持的CPU类型 5](#_Toc364559250)

[Core & Task 6](#_Toc364559251)

[Task状态管理 6](#_Toc364559252)

[Task调度时机 7](#_Toc364559253)

[调度策略 7](#_Toc364559254)

[Task上下文切换 7](#_Toc364559255)

[Mutex 8](#_Toc364559256)

[功能列表 8](#_Toc364559257)

[实现原理 8](#_Toc364559258)

[注意事项 8](#_Toc364559259)

[Timer 9](#_Toc364559260)

[功能列表 9](#_Toc364559261)

[实现原理 9](#_Toc364559262)

[注意事项 9](#_Toc364559263)

[Mailbox 10](#_Toc364559264)

[功能列表 10](#_Toc364559265)

[实现原理 10](#_Toc364559266)

[注意事项 10](#_Toc364559267)

[Memory Management 11](#_Toc364559268)

[功能列表 11](#_Toc364559269)

[实现原理 11](#_Toc364559270)

[注意事项 11](#_Toc364559271)

[How to port 12](#_Toc364559272)

[代码获取 12](#_Toc364559273)

[数据类型定义 12](#_Toc364559274)

[函数实现 12](#_Toc364559275)

[特殊要求 12](#_Toc364559276)

[Set Up 13](#_Toc364559277)

[编译环境 13](#_Toc364559278)

[编译选项 13](#_Toc364559279)

[运行环境 13](#_Toc364559280)

[API List 14](#_Toc364559281)

# Overview

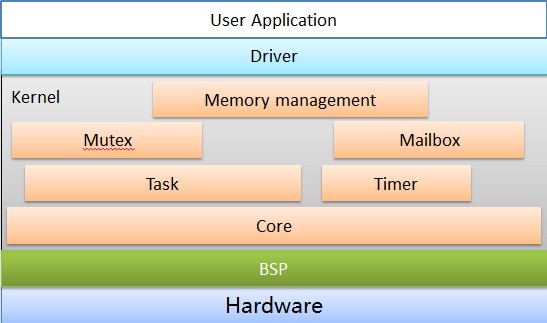
Melon OS 是一个基于优先级和时间片调度的嵌入式操作系统, 具有结构简单，易于移植，模块易裁剪等特点。支持的功能有：

* 多任务并发运行，支持任务的创建，查询，统计，删除，退出，睡眠等。
* 支持Timer的创建，暂停，停止。
* 支持Mutex的创建，锁定，解锁，支持优先级翻转。
* 支持Mailbox的创建，发送，接收，删除。
* 支持物理内存的申请，释放，管理的内存区域大小可配置。

## 系统框图

Melon OS划分为 bsp，kernel，driver，app等几个层次，其中kernel是系统的内核，由core, task, mutex, mailbox, timer, memory management等几个模块组成。

下图是Melon OS的系统框图:



BSP层即Board Support Package, 是对底层硬件的一个封装，为OS等上层代码提供了硬件的基本初始化，中断配置，中断处理函数注册，OS引导等功能。

Kernel层即Melon OS的核心代码部分，由以下子模块组成：

* Core模块负责系统的task调度，timer处理函数的的执行
* Task模块负责任务的创建，查询，删除等功能。
* Mutex模块提供二级制的临界资源锁。
* Mailbox模块提供对task之间的消息交互的支持。
* Memory Management提供对物理内存的申请，释放等管理。

Driver层负责对不同外部设备的配置及功能封装。

APP层即用户代码层。

## 文件结构

Melon OS的文件划分成以下的树状结构:

## 支持的CPU类型

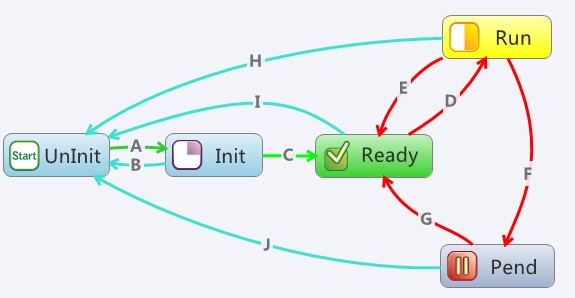
# Core & Task

## Task状态管理

Task的状态分为UnInit，Init，Ready，Run，Pend 等几种类型。

* UnInit表示Task尚未创建，用来标记TCB结构处在空闲的状态。
* Init表示Task正在被创建中。如果创建成功，则Task进度Ready状态，否则回到UnInit状态，并释放TCB结构。
* Ready状态表示一个Task处在可调度态，得到CPU即可运行。
* Run状态表示一个Task处在执行态，系统内同一时刻只有一个Task处在Run状态。
* Pend状态表示一个Task处在阻塞态，即该Task在等待某个系统事件的发生，在等待的事件发生前或等待超时前，该Task不参与调度，一旦该Task等待的事件到来，或等待超时，Task进入Ready状态。 可阻塞Task的系统事件包括：Task主动睡眠，Task 锁定mutex时被挂起，Task从mailbox接收消息时被挂起。

下面是一个TASK的状态迁移图:



1. 找到一个空闲的TCB结构，标记为占用。
2. Task创建失败，释放TCB结构。
3. Task创建成功，进入Ready状态，等待被调度。
4. Task得到CPU使用权，开始执行。
5. 一个调度轮回内，本Task的时间片已经用完，释放CPU使用权，等待下次调度。
6. Task从执行状态跳转为阻塞状态，可能是任务主动睡眠或锁定mutex时被阻塞或者是从mailbox读取消息时被阻塞。
7. 阻塞状态解除，回到Ready队列。由以下几种情况触发: 睡眠时间已经结束，锁定mutex成功/超时， 接收mailbox成功/超时。
8. Task 主动退出，释放TCB结构。
9. Task被删除，释放TCB结构。
10. Task被删除，释放TCB结构

## Task调度时机

## 调度策略

## Task上下文切换

# Mutex

## 功能列表

## 实现原理

## 注意事项

# Timer

## 功能列表

## 实现原理

## 注意事项

# Mailbox

## 功能列表

## 实现原理

## 注意事项

# Memory Management

## 功能列表

## 实现原理

## 注意事项

# How to port

## 代码获取

## 数据类型定义

## 函数实现

## 特殊要求

# Set Up

## 编译环境

## 编译选项

## 运行环境

# API List