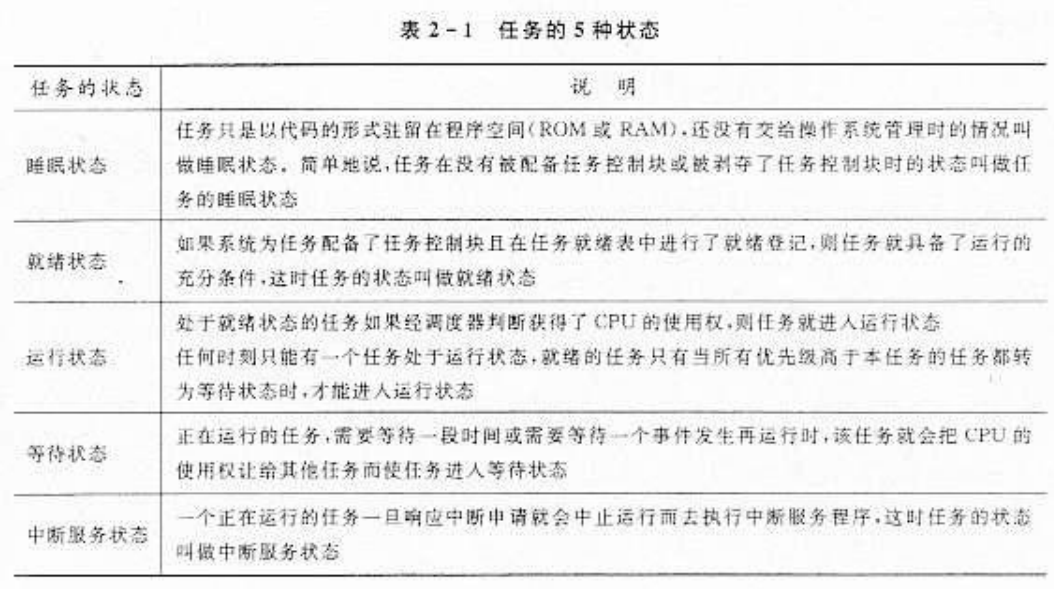
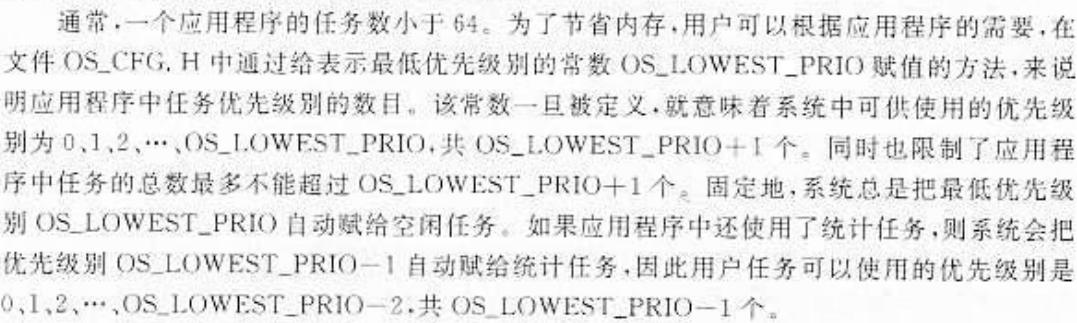
# 任务

## 任务的5种状态



## 任务优先权和优先级别

总共64个优先级别（ucosⅡ2.86版本之前支持64个任务，之后的版本支持256个任务）。数字越大优先级越小。每个任务拥有唯一的优先级别。



OSPrioCur：存放当前任务的优先级别。

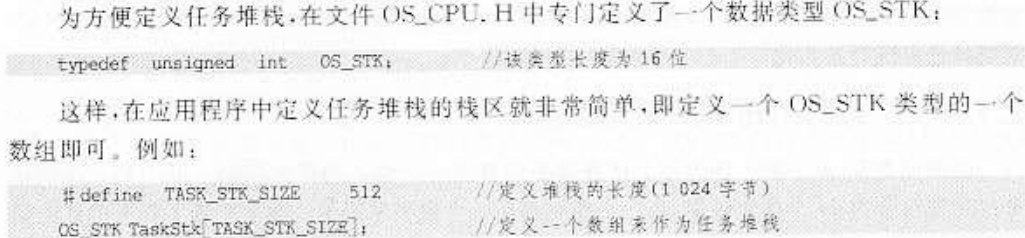
## 系统任务

两个。

空闲任务OSTaskIdle()和统计任务OSTaskStat()。

## 任务堆栈STK

### 堆栈的创建



### 不同处理器堆栈的增长方向

堆栈增长方向分为向上和向下。在使用OSTaskCreate()创建任务时，一定要注意第三个参数“OS\_STK \*ptos”，必须为栈顶的地址。例如，处理器堆栈向下增长，则栈顶地址为TaskStk[TASK\_STK\_SIZE-1]；向上，则栈顶地址为TaskStk0[0]。

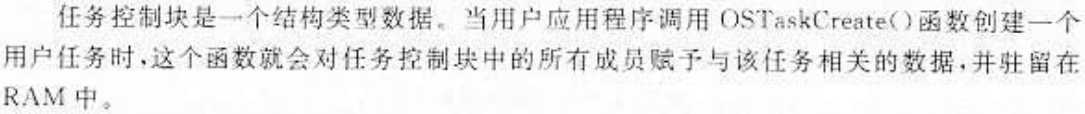
OS\_CFG.H中的常数OS\_STK\_GROWTH为选择开关，用户可自定义。

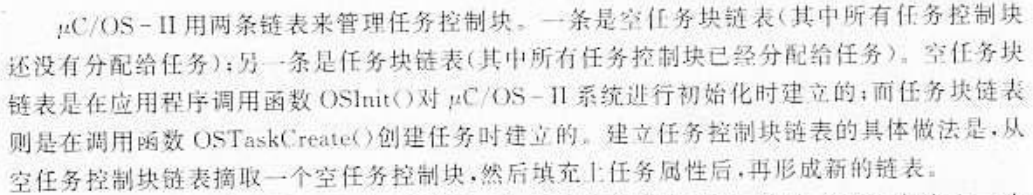
### 堆栈初始化

Uc/os-ii在创建任务函数OSTa-skCreate()中通过调用任务堆栈初始化函数OSTaskStkInit()来完成任务堆栈初始化工作。

## 任务控制块TCB

### 定义





### 相关知识

1. 任务控制块结构体：

Typedef struct os\_tcb

{

…..

}OS\_TCB

1. 系统在调用OSIint()对ucos-ii系统进行初始化时，先在RAM中建立一个OS\_TCB结构类型的数据OSTCBTbl[]，每个数组元素都是一个任务控制块，形成一个链表。空任务链表的元素一共有OS\_MAX\_TASKS（用户任务最大数目）+OS\_N\_SYS\_TASKS（系统任务数目）个。
2. 指针OSTCBFreeList：空任务控制块链表头指针

指针OSTCBList:任务控制块链表头指针

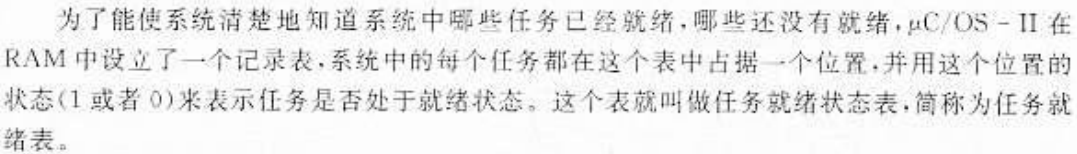
1. OS\_TCB \* OSTCBTbl[]：专门用来以任务的优先级别为顺序在各个数组元素里存放指向各个任务控制块的指针。
2. OS\_TCB \* OSTCBCur：只存放当前正在运行的任务的任务控制块指针。

OS\_TCB \* OSTCBHighRdy：只存放待运行任务的任务控制块指针。由任务调度决定。

1. OSTCBIint()：当使用OSTaskCreate()创建一个任务时，会调用此函数来为任务控制块进行初始化。

## 任务就绪表OSRdyTbl

### 定义



### OSRdyTbl和OSRdyGrp

* OSRdyTbl[]：类型为INT8U的数组。充当任务就绪表。

在此表中，以任务的优先级别高低为顺序，为每个任务安排了一个二进制位，并规定该位的值为1表示任务处于就绪状态，0为非就绪状态。

如果支持64个任务，则该数组有8个元素。

* OSRdyGrp：类型为INT8U的数。每一位对应OSRdyTbl[]的一个任务组。

如果某任务组中有任务就绪，则在变量OSRdyGrp里把该任务组所对应的位置为1；否则为0。

## 任务调度

### 任务调度器

主要工作有两项：

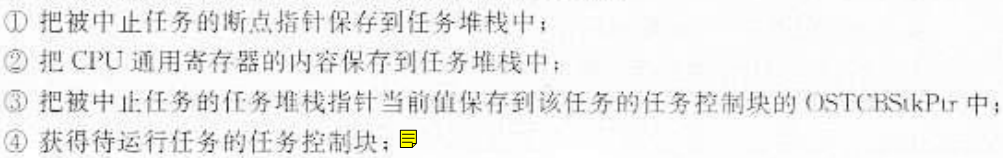
1. 在任务就绪表中查找具有最高优先级别的就绪任务。略。
2. 实现任务切换。有两步：

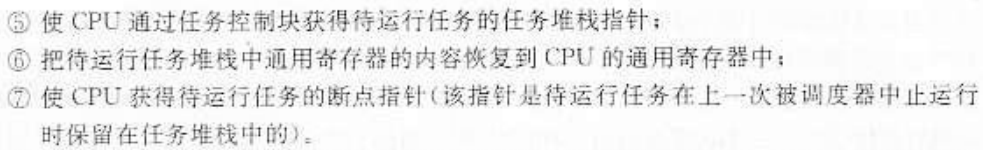
（1）获得待运行任务的TCB指针。由函数OSSched()实现，存放在OSTCBHighRdy中。

（2）断点数据切换。由宏OS\_TASK\_SW()和函数OSCtxSw完成。详见1.7.2。

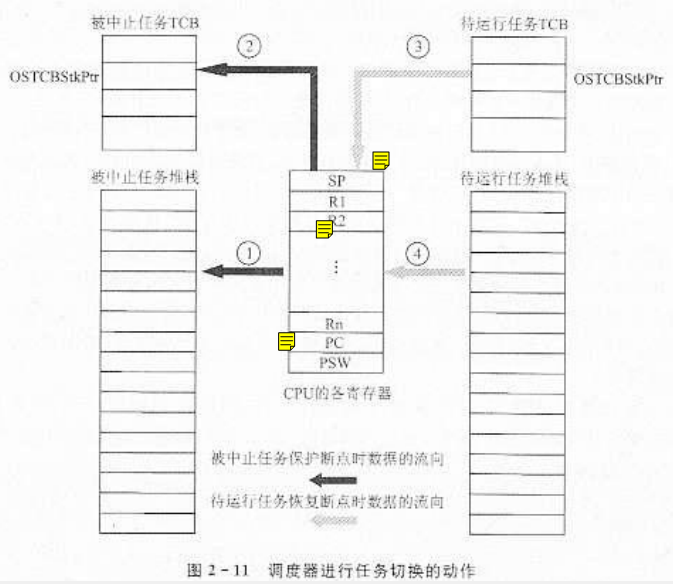
### 相关知识

* 宏OS\_TASK\_SW()用来引发中断。如果使用的微处理器具有软中断指令，那么在这个宏中封装一个软中断指令即可；若无，则可以试一试在宏中封装其它可使PC等相关寄存器压栈的指令（如调用指令）。
* OSCtxSw其实是宏OS\_TASK\_SW()的中断服务程序。工作：





如此即可实现任务切换！图示如下：



* 调度器可以通过OSSchedLock()和OSSchedUnlock()上锁和解锁。并通过变量OSLockNesting记录上锁的嵌套次数，即上锁就+1，解锁就-1。

# 其它

## OS\_ENTER\_CRITICAL()和OS\_EXIT\_CRITICAL():为定义的宏，用来关闭、打开CPU的中断。也称临界段。

参数

无。

返回值

无。

注意/警告

OS\_ENTER\_CRITICAL（）和OS\_EXIT\_CRITICAL（）必须成对使用。