

嵌入式操作系统任务调度

司新生

(鹤壁职业技术学院 河南 鹤壁 458030)

【摘要】 $\mu\text{C}/\text{OS-2}$ 操作系统是一个多任务占先式的实时操作系统,其任务切换是整个操作系统的核心,本文就其任务的切换进行系统的分析,以便更多的人掌握 $\mu\text{C}/\text{OS-2}$ 操作系统,使嵌入式系统的开发更快更方便。

【关键词】 $\mu\text{C}/\text{OS-2}$; 操作系统; 任务切换

随着电子技术的发展,出现了越来越多的针对嵌入式的操作系统,其中 $\mu\text{C}/\text{OS-2}$ 操作系统是专门为嵌入式设计的实时操作系统,该系统完全免费,且源码公开,其内核很小,可以移植到 32 位的 arm 芯片上或 16 位的单片机,甚至可以移植到 8 位的 51 单片机上。由于嵌入式系统的多样性,不可能象桌面系统一样有统一的操作系统。

1 $\mu\text{C}/\text{OS-2}$ 的任务

$\mu\text{C}/\text{OS-2}$ 操作系统是一个多任务系统,它最多可以管理 64 个任务,但两个优先级最低的任务已被系统占用,一个是统计任务,一个是空闲任务。空闲任务的作用时当操作系统没有其它任务执行时,就转入空闲任务而不使系统没事可做。另外系统还保留了其它 6 个任务以备将来操作系统的升级等用,用户实际可以控制的用户任务多达 56 个。

1.1 任务的结构。每一个任务都有如下结构。它由任务控制块 TCB,任务代码,任务堆栈组成,多个任务控制块形成一个任务控制块链表,在一起形成一个整体。

1.2 任务优先级。每一个任务在创建时都被分配有一个任务优先级,优先级序号从 0 到 63,优先级数值越大则表示优先级越低,最高的优先级是优先级序号为 0 的任务,最低的优先级是优先级为 63 的任务,操作系统可以设定管理的任务数,在 OS_CFG.H 文件中,可以定义 OS_LOWEST_PRIO 值,该值最大为 63。每一个任务都有唯一的任务优先级, $\mu\text{C}/\text{OS-2}$ 操作系统任务切换的关键就是该任务的优先级,操作系统总是运行处于就绪状态的最高优先级的任务。

1.3 用户任务代码。任务代码一般具有如下结构,整体上它是一个无限循环,完成特定的任务。它由三部分组成,一是任务的初始化,为任务的运行设置相应的外部环境;二是完成任务的循环体中可被中断的用户代码片断,这部分代码运行时可被中断源中断,转而去执行其它实时性要求更高的任务;三是不可被中断的用户代码片断,一旦被中断就可能产生不可预期的后果。任务代码结构上象一个 C 语言的子函数,但任务代码不能被其它函数或 main 函数调用,只能被 main 函数或已激活的任务创建,只有进行任务切换时,任务代码才能被执行。

1.4 任务的创建。任务的创建主要完成四项任务,一是指出任务代码存放的地址,二是指明任务参数指针,三是指明任务堆栈栈顶指针,在进行任务切换时保存或恢复与任务相关的寄存器的值,四是确定任务的优先级,优先级的高低决定了任务的紧迫性和重要性。

1.5 任务堆栈。任务堆栈是为了任务切换时保存上下文而设置的一个空间,每一个任务都有一个私有的堆栈空间,保存 CPU 寄存器现场 (R0~R12, LR, SPSR 等)和本任务的私有数据。也就是我们所说的用来保存任务切换时的上下文或环境值。任务进行切换前,把当前的任务相关的寄存器保存在该任务的私有堆栈中,然后把待运行的任务的堆栈中的内容复制到相关寄存器中,这样就现实了任务的上下文切换。

1.6 任务控制块。任务控制块是 $\mu\text{C}/\text{OS-2}$ 操作系统任务管理与切换的关键。任务控制块是一个 C 语言的自定义结构体,主要包含有任务堆栈栈顶地址,任务的优先级,上下任务控制块的地址,任务的状态等相关信息。任务控制块由上下控制块地址指针与上下任务控制块之间形成一个双向链表。

任务控制块的管理主要有以下几个变量,一是 OSTCBTbl[]—任务控制块数组;二是 OSTCBPrioTbl[]—任务控制块优先级数组,以 Prio 为下标存放已使能的 TCB 指针,用于加速 TCB 的访问。OSTCBCur—全局系统变量,指向当前正在运行的任务的 TCB。

2 任务就绪表及任务调度

2.1 任务就绪表的结构

为了能够使系统清楚地知道,系统中哪些任务已经就绪,哪些还没有就绪, $\mu\text{C}/\text{OS-2}$ 在 RAM 中设立了一个记录表,系统中的每个任务都在这个表中占据一 Bit 的位置,并用这个位置的状态 (1 或者 0) 来表示任务是否处于就绪状态,这个表就叫做任务就绪状态表,简称任务就绪表。由一个 8 位的数组 OSRdyTbl[] 充当,该数组最多 8 个元素,可以标示 64 个任务。

任务就绪表就是一个二维数组 OSRdyTbl[],如果某一位为 1,则表明该优先级的任务已就绪,可以进行调度,否则该任务还没有处于就绪状态,系统还不能直接运行该项任务。例如 OSRdyTbl[2]的 D3 为 1 则表示任务优先级为 19 的任务已处于就绪状态,可以被系统运行。为加快访问任务就绪表的速度,系统定义了一个变量 OSRdyGrp 来表明就绪表每行中是否存在就绪任务。

根据任务的优先级 prio、优先级就绪数组 OSRdyTbl[]、任务就绪组 OSRdyGrp。系统就可以对任务进行管理与控制。

$\mu\text{C}/\text{OS-2}$ 可以最多管理 64 个任务,其优先级为 0~63, prio 的值为 000000~111111,根据任务就绪表的结构,由任务的优先级就可以找到该任务在任务就绪表中所对应的位,优先级 prio 除以 8 取整,也就相当于优先级 prio 右移三位,代表该任务在任务就绪表中的组号 x,也即 OSRdyTbl[]数组的下标等于 prio.[5:3]。优先级 prio 的低三位 prio.[2:0]代表在任务优先级组中的位 y。为了加快任务的调度及保证任务切换时间一致性, $\mu\text{C}/\text{OS-2}$ 操作系统预先定义了一个数组 OSMMapTbl[]。

通过查表法,操作系统就可以知道该优先级在任务就绪表中哪一组,对应 OSRdyGrp 的哪一位,在该组中处于哪一位。例如:如果任务优先级 prio 为 21 的任务处于就绪状态,十进制 21 转换为二进制则为 10101,该数值右移三位为 010,010 表示十进制的 2,查表 OSMMapTbl [2]=00000100B 知道 OSRdyTbl[2]有任务进行就绪状态。10101 的低三位为 101,101 表示十进制 5,查表 OSMMapTbl [5]= 00100000B 可确定 OSRdyTbl[2]的第五位为 1。

2.2 任务进入就绪状态

只有进入就绪状态的任务才能被调度,从而被 CPU 执行。一个任务如何才能处于就绪状态呢?只要在对应的任务就绪表中登记该任务,就使该任务处于就绪状态。如果该任务的优先级为 prio,首先使该优先级对应的就绪组置位,

```
OSRdyGrp|= OSMMapTbl[prio>>3]
```

然后使该组对应的位置位,

```
OSRdyTbl[prio>>3]= OSMMapTbl[prio&0x07]
```

2.3 任务脱离就绪状态

与任务进入就绪状态相反,使该优先级的任务对应的位复位,如果该就绪组没有任务处于就绪状态,就使该任务就绪组对应的位复位。首先使该任务对应的位复位,

```
OSRdyTbl[prio>>3]&= ~OSMapTbl[prio&0x07]
```

如果该就绪组没有任务处于就绪状态,则该就绪组对应的位复位,

```
if(OSRdyTbl[prio>>3]==0)
```

```
{OSRdyGrp&=~ OSMMapTbl[prio>>3]}
```

2.4 任务调度

任务调度,也就是找到优先级最高的任务,然后转到该任务的代码段并执行该项段代码。

2.4.1 找到优先级最高的任务。根据任务就绪表的结构,任务就绪组中最低为 1 的位所对应的组中最低为 1 的位所对应的 (下转第 575 页)

力作用下做变速到加速的过程,并具有实时将油滴运动的加速度的瞬时值显示出来。下面是的设计的源程序:

```

a=a-0.01;
cst=Number(cst) +0.2;
if(Number(a)<0){css=Number(css) +Number(csyu*0.2);
csta=0;}
else
{css=Number(css)+Number(csu*0.2)+Number(0.5*a*0.2*0.2);
csy=Number(cst) ;
csu=Number(csu+a*0.2);csyu=csu;
csat=a;}
stproperty("/youdc",_y,css);
if(Number(css)<335){prevFrame;play();}

```

密立根油滴在无电场的作用下做变速——匀速的运动,利用在做匀速运动的过程测出速度 V,然后计算油滴的半径和体积。有电场时,带电油滴的运动情况和测量方法读者可以自己模拟。

3.2 静电场实验数据的计算机处理方法 在静电场物理实验中,学生要测量电位相等的方向大致对称的 8 个点,然后依据这 8 个点手工绘制成圆环,并估定圆心。进而测量等位圆环的半径值。这样处理,一是描绘圆环比较困难,二是圆心是估画出来的,误差较大。

若利用计算机,采用数学中的三点定位法,可以科学地找到圆心及半径,并画出正规的圆环。采用编程来处理数据及绘图,能够避免繁琐的数学计算和手工绘制图形带来的误差,有一定的实用价值。

以下是实验数据处理及绘图的主程序和相关函数:

```

#include "stdio.h" #include "math.h" #include "graphics"
struct slope {int infintdeMax;float m_slope;}struct coordinate {float x;float y;}struct coordinate CurrentCenter *next;
main()
{struct slope m_slope;struct coordinate m_Current;struct coordinate CurrentCenter,m_Cente;structcoordinate*head;float CurrentRadius,m_radius;int i;flag =0;i =0;CurrentRadius =0;m_radius =0;m_Current.y =0;head =malloc (sizeof (sturct coordinate));InputParameter (&head);m_Current =head;while (flag <1){SeekCircleCenter (*m_Current,* (m_Current->next),*(m_Current->next->next),&CurrentCenter);seekRadius (*m_Current,CurrentCenter,&CurrentRadius);m_Current.x +=CurrentCenter.x;m_Current.y +=CurrentCenter.y;m_radius +=CurrintRadius;m_Current = m_Current->next;if(m_Current==head)flag++;i++;}m-Center.x=m_Cen-ter.x/i;m_radius=m_radius/i;getch();Draw(head,m_Cener,m-radius);}
InputParameter(&head); 录入实验数据,并形成一个环形链表;

```

(上接第 461 页) 优先级就是处于就绪状态的最高优先级的任务。

如果 OSRdyGrp=01011010, 则说明四个组中有任务处于就绪状态, 而 OSRdyTbl [1] 这一组优先级最高。如果 OSRdyTbl [1]=10010010, 则说明该组有三个任务处于就绪状态, 从右数第二个 1 所代表的优先级是所有就绪的任务中优先级最高的。为了快速得到优先级最高的任务, μC\OS- 仍然采用查表的方式, μC\OS- 操作系统预先定义了一个数组 OSUnMapTbl[256]

设最高优先级组为 y, 该组内优先级最高的位为 x, 则 y=OSUnMapTbl[OSRdyGrp],x= OSUnMapTbl[OSRdyTbl[y]], 与由优先级找到就绪组和该组内的位的方法相反, 则最高优先任务的优先级 prio=y*3+x。当 OSRdyGrp=01011010b=0x5a, OSRdyTbl[1]=10010010=0x92 时, y= OSUnMapTbl[0x5a]=1 x= OSUnMapTbl[OSRdyTbl[1]]= OSUnMapTbl[0x92]=1 prio=1*3+1=4

也就是说优先级为 4 的任务是当前优先级最高的任务。知道了最高优先级的任务, 由 OSTCBPrioTbl[prio]就可以得到该任务的任務控制块的地址。

2.4.2 切换任务上下文。这里有四个工作要做, 第一五保存当前任务的 CPU 的格寄存器内容到当前任务的私有堆栈中, 第二保存当前任务的堆栈的栈底地址到当前任务的任務 TCB 中, 第三由最高任务的控制块 TCB 得到该任务的私有堆栈地址 sp, 第四根据该任务的堆栈地址 sp, 装入该任务的中断的数据。

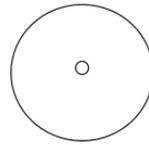
SeekCircleCenter; 计算 3 点所确定的圆的圆心坐标;

SeekRadius; 计算 3 点所确定的圆的半径坐标;

Draw: 图形工作模式下输出实验数据, 圆环的半径, 模拟静电场图。

计算机处理实验数据后的输出结果见下图:

The parameter list:



```

##:dot1:9.8999,59.5999
##:dot2:23.7999,23.8999
##:dot3:60.999,9.8999
##:dot4:95.8000,24.10000
##:dot5:110.40000,59.59999
##:dot6:96.90000,96.90000
##:dot7:59.97999,110.12000
##:dot8:24.0,96.0
##:Radius:50.95086
##:CircleCenter:60.19884,60.13910

```

4 结束语

计算机引入实验教学会给传统实验教学带来新的生机和活力, 计算机辅助实验不仅能激发学生对实验的兴趣, 使学生从被动做实验转为主动探索实验规律, 同时计算机为学生又提供了深入研究实验的数据的手段, 学生在实验中手脑并用, 其解决问题的能力必然会提高。通过编写程序也培养了学生的逻辑思维能力。当然, 我们也应注意计算机的引入不应使实验过于自动化, 一按回车键实验就结束了, 结果就出来了, 若这样将会使学生一无所获。

【参考文献】

[1] 李灵杰. 物理实验教学中学生消极心理定势分析[J]. 物理实验, 2000, (07).
[2] 董涛. 中学物理创新实验的开发与研究[D]. 华中师范大学, 2003.
[3] 廖先碧. 利用计算机辅助实验改革大学物理实验[J]. 四川职业技术学院学报, 2004, (04).
[4] 沈元华. 在基础物理实验教学中努力培养创新人才[J]. 大学物理, 2000, (01)
[5] 蔡青. 仿真实验的应用与大学物理实验的教学改革[J]. 成都信息工程学院学报, 2006, (01).

作者简介: 钟带生 (1978.10.12—), 男, 赣南师范学院教育科学学院教师, 研究方向为高校物理教育。

[责任编辑: 翟成梁]

2.4.3 运行最高优先级的任务代码。经任务调度后, PC 已指向最高优先级的代码段的位置, 该任务获得了 CPU 的使用权, 从而实现了任务的切换工作。

3 结束语

μC\OS- 操作系统是一个多任务占先式的实时操作系统, 其任务的管理依靠任务控制块 TCB, 任务控制块 TCB 保存有任务的私有堆栈地址、任务的代码段地址、上下任务控制块 TCB 的地址及任务的优先级等信息。所有就绪的任务在任务就绪表中都有登记, 根据任务就绪表就可以找到最高优先级的任务, 任务的切换就是保存原来任务的数据到该任务的私有堆栈中, 从最高优先级的任务堆栈中获得其被中断的数据, PC 指向该任务的代码段并运行该段代码。

【参考文献】

[1] 任哲. 嵌入式实时操作系统 μC\OS- II 原理及应用. 第二版[M]. 北京: 北京航空航天大学出版社, 2009.
[2] 王田苗. 嵌入式系统设计与实例开发[M]. 北京: 清华大学出版社, 2003 年 10 月.
[3] [美] Jean J. Labrosse. 嵌入式实时操作系统 μC\OS- [M]. 邵贝贝, 等, 译. 北京: 北京航空航天大学出版社, 2003.

[责任编辑: 曹明明]