VxWorks

程序员速查手册

⊙ 周启平　张 杨　编著

机械工业出版社
CHINA MACHINE PRESS

VxWorks 程序员速查手册

周启平　张　杨　编著

机 械 工 业 出 版 社

本书介绍了 VxWorks 操作系统的常用函数库，并对库中各函数进行了详细描述。全书共 12 章，主要内容包括：任务管理与调度函数、任务同步与通信函数、时钟管理函数、中断管理函数、内存管理函数、I/O 系统函数、文件系统函数、系统内核函数、用户函数、浮点函数、网络系统函数和错误状态函数等。

本书语言通畅、条理清晰、内容全面且深入浅出，以精选实例加文字说明的方式结合编者多年的实际开发经验编写而成。它是 VxWorks 程序员的案头参考书，无论是 VxWorks 初学者还是资深程序员都能从中受益匪浅。

图书在版编目（CIP）数据

Vx Works 程序员速查手册 / 周启平，张杨编著. —北京：机械工业出版社，2005.2

ISBN 7-111-16123-8

Ⅰ. V… Ⅱ. ①周…②张… Ⅲ. 实时操作系统，Vx Works—技术手册 Ⅳ. TP316.2-62

中国版本图书馆 CIP 数据核字（2005）第 009276 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）

责任编辑：吉 玲（E-mail：jiling@mail.machineinfo.gov.cn）

责任印制：石 冉

三河市宏达印刷有限公司印刷 · 新华书店北京发行所发行

2005 年 2 月第 1 版第 1 次印刷

787mm × 1092mm 1/16 · 20.25 印张 · 652 千字

0001—4000 册

定价：36.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

本社购书热线电话（010）68993821、88379646

68326294、68320718

Http://www.machineinfo.gov.cn/book/

前　言

随着 VxWorks 操作系统在嵌入式领域越来越广泛的应用，一种新的书籍需求正在形成。即编程者需要多种相关的速查手册。该书的出版正是为了适应这样的需求，为广大程序员提供最大的便利。为了能做到给程序员提供正确的帮助，书中的函数说明力争可以完全地符合 WindRiver 公司所提供的说明文档，而所有的例子则尽量使用公开的程序片断。

本书适合于使用 VxWorks 操作系统的人，不论是入门级的或是资深级的程序员都可以将本书作为函数速查手册使用。书中提供了一些例子和函数库的详细说明，使得该书也可以作为 VxWorks 的参考手册来使用。在书中，很多函数库的说明较为详细，可以对其他的资料作一定程度的补充。

本书包括了任务的调度和管理、终端管理、系统时钟的管理、内存管理、IO 系统、文件系统、网络系统、用户接口、异常处理等诸多分类的函数。在函数的选材上，我们希望能包含所有的常用的函数，但有很多函数库和函数的使用程度是我们无法预料的，希望读者能将意见及时地反馈给我们。

该书由周启平和张杨合作完成，其中第 1～4、9、11 章由周启平完成，第 5～8、10、12 章由张杨完成。例子的添加与测试，章节的修改、校对等工作，是由周启平和张杨共同完成的。在此，向所有支持本书编写的人员表示感谢。

同时还要感谢出版社的编辑们、我们的亲友和同事在该书出版过程中对我们的帮助和鼓励。尽管抱着认真的态度，但在书中还是难免会出现各种各样的错误，希望各位读者能不吝赐教。

我们的联系方式：
周启平：qpzhou@sina.com
张杨：hwybird@sohu.com

编者

目　录

第1章　任务管理与调度

1.1　任务管理函数

1.1.1　函数库描述

1．库命名

任务管理函数库名称为 taskLib。

2．函数

表 1-1 中列出了任务管理函数。

表 1-1　任务管理函数

函　数	描　述	函　数	描　述
taskSpawn()	创建并激活任务	taskPriorityGet()	获得任务优先级
taskInit()	初始化任务（指定堆栈地址）	taskLock()	禁止任务调度
taskActivate()	激活已经被初始化的任务	taskUnlock()	使能任务调度
exit()	结束任务并释放内存	taskSafe()	保护任务，防止其被非法删除
taskDelete()	删除任务	taskUnsafe()	解除任务的安全保护
taskDeleteForce()	无条件删除任务	taskDelay()	任务延时（参数为时间片）
taskSuspend()	挂起任务	taskIdSelf()	获得当前正在运行任务的任务 ID
taskResume()	恢复任务	taskIdVerify()	核实任务的存在
taskRestart()	重启任务	taskTcb()	获得任务控制块
taskPrioritySet()	改变任务优先级		

3．描述

任务管理库为 VxWorks 的任务管理机制提供了接口。任务控制服务是由 VxWorks 内核提供，其内核包括 kernelLib 库、taskLib 库、semLib 库、tickLib 库、msgQLib 库和 wdLib 库。taskInfo 库提供了访问和调试任务信息的函数。至于更高级别的任务信息显示函数则由 taskShow 库来提供。

4．任务创建

任务创建通常是由 taskSpawn()函数来完成的。其创建过程包括：为任务的堆栈和任务控制块（WIND_TCB）分配内存、初始化任务控制块和启动任务控制块。如果用户有特殊需求，可以使用更低级别函数 taskInit()和 taskActivate()，并且它们就是函数 taskSpawn()的基本构成。

VxWorks 系统采用优先级抢占调度方式，任务执行在体系结构所决定的特权状态中。

对于 VxWorks 系统而言，只要有足够的可用内存能够满足系统分配需求，其创建任务的个数是没有限制的。

usrLib 库提供的 sp()函数是创建任务的一个简单缩写，它使用缺省参数调用 taskSpawn()函数。

5．任务删除

在任务创建期间，如果任务退出它的“主”程序，内核隐含调用 exit()函数删除这个任务。事实上用户可以通过调用 taskDelete()或 exit()函数删除任务。

由于资源回收困难，用户需要小心对待任务删除操作。删除一个任务时，其拥有的临界资源可能会破坏

系统，因为这个资源可能再也不可用。因为系统在删除任务的时候并不知道资源所处的状态，所以在此时简单地释放资源不是可行的解决方案。

采用删除保护方式解决任务删除问题胜于采用其他过度复杂的删除机制。对于未预料到的删除操作，用户可以通过使用 taskSafe()函数来保护任务。当任务处于删除保护的时候，删除者将会被阻塞直到解除任务的删除保护。同样，任务可以通过获得一个互斥信号量来保护自己免于被删除，而这个互斥信号量是在任务创建时选中 SEM_DELETE_SAFE 选项来创建的。无论系统调用 taskSafe()函数还是调用 semTake()函数，系统都需要调用与之对应的 taskUnsafe()和 semGive()函数(有关详细信息请参考 2.1.3 节)。许多 VxWorks 系统资源都采用这种方式来保护。应用程序设计者在可能发生动态的任务删除操作之处也应该考虑使用这种方法。

系统也会使用 sigLib 模块，允许任务在实际终止前执行清除代码。

6．任务控制

在任务创建的时候，系统会给任务分配一个 ID，其后系统对任务的操作是依靠这个 ID 进行的。在任务控制函数中，VxWorks 通过使用指定的任务 ID 控制任务。

下面这些函数可以控制任务状态：taskResume()、taskSuspend()、taskDelay()、taskRestart()、taskPrioritySet()和 taskRegsSet()。

7．任务调度

VxWorks 系统基于优先级抢占方式调度任务。任务可以拥有的优先级范围为 0～255，其中 0 为最高优先级，255 为最低优先级。VxWorks 中任务的优先级是可以动态改变的，用户可以通过调用 taskPrioritySet()函数来改变任务现有的优先级。

8．头文件

任务管理函数声明在 taskLib.h 中。

9．参考

相关信息请参考 taskInfo、taskShow、taskHookLib、taskVarLib、semLib、semMLib、kernelLib 库描述，以及“VxWorks Programmer's Guide”书中的基本操作系统章节。

1.1.2 任务管理函数详细描述

taskSpawn()

函数原型：

```
int taskSpawn
(
char * name,            /* 新任务名称(存储在 pStackBase 中) */
int priority,           /* 新任务优先级 */
int options,            /* 任务选项字 */
int stackSize,          /* 任务所需堆栈大小(字节) */
FUNCPTR entryPt,        /* 新任务入口点 */
int arg1,               /* 以下是传递给任务函数的 10 个参数 */
int arg2,
int arg3,
int arg4,
int arg5,
int arg6,
int arg7,
int arg8,
int arg9,
int arg10
```

)

功能描述：

创建并激活任务。该函数创建并激活一个带有指定优先级和选项字的新任务，同时返回一个系统分配的 ID 号。相关信息参考该函数的组成部分：taskInit()和 taskActivate()。

在任务创建时可以分配一个名称给任务，便于调试。任务名将会出现在各种系统信息函数所产生的显示中，例如 i()函数。任务名的长度和内容可以是任意的，不过当前 VxWorks 系统约定任务名的长度不超过 10 个字符而且任务名第一个字母为“t”。如果 name 参数为 NULL，系统将为该任务分配一个 ASCII 名称，其形式为“t*n*”，在这里“*n*”是一个随着新任务创建而增加的整数。

分配给新任务的堆栈资源大小由 stackSize 参数指定，而资源是从系统内存区中分配得到。堆栈大小应该是一个偶数。任务控制块（TCB）存放在堆栈中，像任务名一样也需要一些内存。剩余的任务堆栈通过 checkStack()函数用变量 0xEE 来填充堆栈的每个字节。相关信息请参考堆栈检验函数 checkStack()。

入口地址 entryPt 是任务主函数的地址。一旦建立了 C 环境，系统将调用该函数。该函数最多可以带有 10 个给定的参数。在主函数将要返回时系统自动地调用 exit()。

值得注意的是 10 个（最多 10 个）参数必须传递给创建函数。

选项字参数中的位可以设置任务运行在如下模式：

VX_FP_TASK (0x0008)：浮点协处理器支持；

VX_PRIVATE_ENV (0x0080)：包含私有环境支持(参考 envLib)；

VX_NO_STACK_FILL (0x0100)：不使用 checkStack()填充堆栈；

VX_UNBREAKABLE (0x0002)：禁止断点调试。

有关选项字定义请参考头文件 taskLib.h。

返回值：

成功创建任务则返回任务 ID，如果内存不足或不能创建任务则返回 ERROR。

错误码：

S_intLib_NOT_ISR_CALLABLE、S_objLib_OBJ_ID_ERROR、S_memLib_BLOCK_ERROR、S_smObjLib_NOT_INITIALIZED、S_memLib_NOT_ENOUGH_MEMORY。

参考：

相关信息请参考 taskLib 库描述、taskInit()、taskActivate()、sp()函数等。

例：

```
/* 数据定义 */
#define     TASK_PRI150         /* 任务优先级 */
int         taskId;             /* 任务 ID */
void  taskDemo (void);          /* 任务入口函数声明 */
...
void mainDemo(void)
{
/* 创建并启动任务 */
taskId = taskSpawn ("tDemo", TASK_PRI, VX_FP_TASK, 4000,
        (FUNCPTR) taskDemo, 0,0,0,0,0,0,0,0,0,0);
if(taskId == ERROR)
printf("spawn taskDemo failed!\n");
}
/* 演示任务入口函数 */
void taskDemo (void)
```

```
    {
...
    for (;;)
{
printf("taskDemo is running!\n");
...
}
    }
```

taskInit()

函数原型：

```
    STATUS taskInit
    (
    WIND_TCB * pTcb,    /* 新任务的 TCB 地址 */
    char * name,        /* 新任务名称 (存储在 pStackBase 中) */
    int priority,       /* 新任务优先级 */
    int options,        /* 任务选项字 */
    char * pStackBase,  /* 新任务的堆栈基地址 */
    int stackSize,      /* 任务所需堆栈大小(字节) */
    FUNCPTR entryPt,    /* 新任务入口点 */
    int arg1,           /* 以下是传递给任务函数的 10 个参数 */
    int arg2,
    int arg3,
    int arg4,
    int arg5,
    int arg6,
    int arg7,
    int arg8,
    int arg9,
    int arg10
    )
```

功能描述：

初始化任务。该函数初始化用户分配给任务堆栈和控制块的内存区域，代替从系统内存中自动分配的空间（taskSpawn()自动分配内存空间）。该函数将使用指向 WIND_TCB 和堆栈的指针作为任务的组成部分，这样允许一个静态 WIND_TCB 变量的初始化，也允许对特殊的堆栈进行配置用于辅助调试。

在 taskSpawn()中，系统会分配一个名称给任务。taskSpawn()自动为未命名的任务命名，而 taskInit()允许未命名任务的存在。对于其他任务函数而言，任务 ID 是必需的，它实际上是参数 pTcb 的地址。

值得注意的是任务堆栈的基地址由参数 pStackBase 决定，根据目标机体系结构向上或向下增长。

该函数的其他参数与 taskSpawn()中的参数一样。与 taskSpawn()不同的是 taskInit()并不激活任务。任务的激活操作必须是在调用 taskInit()之后调用 taskActivate()函数来完成。

通常，用户使用 taskSpawn()而非 taskInit()启动任务，除非当任务内存分配需要额外控制或希望单独激活任务。

返回值：

成功初始化任务则返回 OK，如果初始化任务失败则返回 ERROR。

错误码：

S_intLib_NOT_ISR_CALLABLE、S_objLib_OBJ_ID_ERROR。

参考：

相关信息请参考 taskLib 库描述、taskActivate()和 taskSpawn()函数。

例：

```
/* 数据定义 */
#define     TASK_PRI100         /* 任务优先级 */
WIND_TCB    * pTcb;             /* 任务的 TCB 地址 */
int         status;             /* 函数返回状态值 */
void        taskDemo (void);    /* 任务入口函数声明 */
int         pStackBase          /* 堆栈基地址 */
...
void mainDemo(void)
{
pStackBase = 0x2000;

/* 创建并启动任务 */
status = taskInit (pTcb, "tDemo", TASK_PRI, VX_FP_TASK,(char *)pStackBase, 500,
                   (FUNCPTR) taskDemo, 0,0,0,0,0,0,0,0,0,0);
if(status == ERROR)
printf("initialize taskDemo failed!\n");
}
/* 演示任务入口函数 */
void taskDemo (void)
    {
...
    for (;;)
{
printf("taskDemo is running!\n");
...
}
    }
```

taskActivate()

函数原型：

```
    STATUS taskActivate
    (
    int tid  /* 任务 ID */
    )
```

功能描述：

激活已经初始化的任务。该函数激活用 taskInit()函数创建的任务。如果任务没有激活，则任务没有资格让调度程序分配 CPU 给它。参数 tid（任务 ID）是任务的 WIND_TCB 地址，且为一个整数：

```
tid = (int) pTcb;
```

taskSpawn()函数是由 taskActivate()和 taskInit()函数组成。通过 taskSpawn()函数创建的任务并不需要激活

任务操作。

返回值：

成功激活任务则返回 OK，如果任务激活失败则返回 ERROR。

参考：

相关信息请参考 taskLib 库描述和 taskInit()函数。

exit()

函数原型：

```
void exit
(
int code            /* 传递给删除钩子函数的代码，存储在 TCB 中 */
)
```

功能描述：

任务退出。这是一个 ANSI C 函数，任务通过调用该函数退出任务。当创建任务的主程序退出时，系统隐含地调用该函数。参数 code 将存储在 WIND_TCB 中，删除钩子函数或后继的调试可能会用到它。

返回值：

无。

参考：

相关信息请参考 taskLib 库描述、taskDelete()函数和“VxWorks Programmer's Guide”中的基本操作系统章节。

taskDelete()

函数原型：

```
STATUS taskDelete
(
int tid         /* 任务 ID */
)
```

功能描述：

删除任务。该函数停止并删除指定的任务，同时重新分配相关的堆栈和 WIND_TCB 内存资源。任务删除前，在被删除任务的上下文中将调用所有通过 taskDeleteHookAdd()函数添加的钩子函数。另外，该函数是 taskSpawn()的对应函数。

返回值：

成功删除任务则返回 OK,任务删除失败则返回 ERROR。

错误码：

S_intLib_NOT_ISR_CALLABLE、S_objLib_OBJ_DELETED、S_objLib_OBJ_UNAVAILABLE、S_objLib_OBJ_ID_ERROR。

参考：

相关信息请参考 taskLib、excLib 库描述，taskDeleteHookAdd()和 taskSpawn()函数，以及“VxWorks Programmer's Guide”中的基本操作系统章节。

例：

```
int         taskId;             /* 任务 ID */
...
/* 查找任务名为“tDemo”的任务 */
taskId = taskNameToId("tDemo");
```