优先级反序算法在汽车电子控制平台设计中的应用*

冷 静[1],耿剑锋[1],宋雪桦[1],沈廷根[1,2]

(1.江苏大学 计算机科学与通信工程学院,江苏 镇江 212013;
2.江苏大学 应用物理研究所,江苏 镇江 212003)

摘 要:为了实现汽车应用软件在不同开发平台上的无缝连接,将 μC/OS-II 操作系统移植到飞思卡尔芯片 MC9S12DP256B 微控制器上,构建了一个开放的微控制器硬件控制平台;同时与汽车业界提出的 OSEK/VDX 标准相匹配,采用优先级反序算法设计了基于 μC/OS-II 微内核结构的汽车专用嵌入式操作系统。通过测试表明,系统为应用程序提供了稳定的运行平台,优化了汽车控制性能。

关键词:μC/OS-II 微内核;OSEK/VDX 标准;优先级;嵌入式

中图分类号:TP316.2 **文献标识码**:B

An anti-priority method for designing the vehicle control platform

LENG Jing[1], GENG Jian Feng[1], SONG Xue Hua[1], SHEN Ting Gen[1,2]

(1.Department of Computer Science and Telecommunications Engineering, Jiangsu University, Zhenjiang, 212013, China;
2.Department of Physics, Jiangsu University, Zhenjiang 212003, China)

Abstract: In order to make the auto software can adapt to different electronic control unit, first, replant the μC/OS-II microkernel to the chip named MC9S12DP256B, then construct a exoteric and manipulative hardware platform. Second, in order to match the OSEK/VDX standard, design the automotive embedded operating system by adopting the method of antitone priority based on μC/OS-II microkernel. Third, test the OS, then the test result shows the operating system can enhance the automotive sensitivity and optimize the control performance.

Key words: μC/OS-II microkernel; OSEK/VDX standard; priority; embedded

随着科技的进步,人们对汽车的性能指标要求越来越高,由此汽车电子化程度也在不断提高,使得当代汽车电控单元从硬件到软件变得更加复杂,迫切需要汽车嵌入式操作系统开发平台协助完成各种电控单元的设计及开发。μC/OS-II 微内核代码简洁,实时性和专用性强,作为汽车专用嵌入式操作系统的微内核,移植到芯片 MC9S12DP256B[1]微控制器上,构建一个开放的汽车电子微控制器硬件控制平台,在此硬件基础上通过优先级反序算法设计符合汽车业界 OSEK/VDX 规范标准的汽车电子嵌入式操作系统,使得该操作系统在可移植性、可靠性以及扩展性方面有很大的提高。

本文针对 μC/OS-II[2]微内核设计符合 OSEK/VDX 标准的汽车电子嵌入式操作系统,并进行了功能测试,系统运行稳定。

1 OSEK/VDX 规范的研究

为了屏蔽不同电控单元(ECU)的接口特性,减少开发费用和时间,实现软件的可移植性、可扩展性而提出的 OSEK/VDX 开放式系统及接口规范[3],主要包括 4 部分:(1)操作系统(OS)规范,该规范定义操作系统内核的实现机制和应用编程接口(API);(2)通信(COM)规范。实现各个电控单元间和某个电控单元内的数据信息交换,即外部与内部通信;(3)网络管理(NM)规范。电控单元通过串行数据通信链连接成网络,网络管理规范为保证通信网的安全性与可靠性,提供了确保网络功能的接口函数;(4)实现语言(OIL)。根据应用软件的实际需要配置操作系统及通信机制,以缩减最终生成可执行文件的体积。操作系统、通信管理和网络管理是 3 个可以独立存在的模块,三者之间关系如图 1 所示。

* 基金项目:国家 863 项目(2006AA11A128)

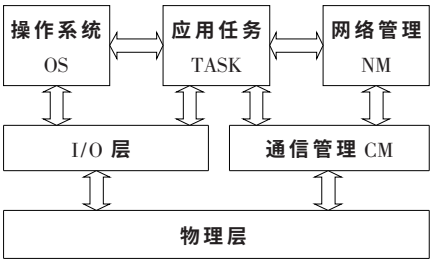

图 1 OSEK 结构关系图

OSEK OS 是针对汽车应用特点而专门制定的一个小型 RTOS 规范，具有以下特点：

(1)可移植性，所有 API 都是标准化的并且在功能上都有明确的定义。

(2)可扩展性，OSEK OS 要求通用于各种类型的 ECU，因此一方面系统要高度地模块化，同时能进行灵活的配置。

(3)汽车应用需要的可靠性、实用性等。由于采用微内核结构，其微内核部分代码数量小，各个部分之间关系不是很复杂，可以保证代码的正确性。当增加操作系统功能时，只需在核外调试和运行，不会危及到微核，整个系统的安全系数也比较高。

2 符合 OSEK/VDX 规范的优先级反序算法

OSEK OS 与 μC/OS-II 内核在优先级顺序上采取不同使用方法。在 μC/OS-II 内核的 64 个优先级中，数字越小表明级别越高。而 OSEK 标准相反，要求数字越大级别越高。为了实现符合 OSEK 汽车电子国际标准的汽车嵌入式系统，本文针对 μC/OS-II 内核，设计了符合 OSEK OS 标准中关于优先级的要求。

运行最高优先级任务的实现步骤[4]：首先使已经创建的任务进入就绪表，然后查询就绪表中优先级最高的任务，最后通过调度函数运行该最高优先级任务。在 μC/OS-II 操作系统中，定义了 2 个数组 OSMapTbl[ ]和 OSUnMapTbl[ ]以及 1 张就绪表。就绪表中有 2 个变量：OSRdyTbl[ ]和 OSRdyGrp。在 OSRdyGrp 中，任务按优先级分组，由于 μC/OS-II 操作系统的优先级定义为 64 级，所以 8 个任务定为一组。OSRdyGrp 中的每一位表示 8 组任务中每一组是否有进入就绪态的任务。任务进入就绪态时，OSRdyTbl[ ]中的相应元素的相应位置为 1。OSRdyGrp 和 OSRdyTbl [ ] 之间的关系为：当 OSRdyTbl[i] ($0 \leqslant i \leqslant 7$)中的任何 1 位是 1 时，OSRdyGrp 的第 i 位置 1。任务优先级 prio 的低 3 位确定任务在就绪表 OSRdyTbl[ ]中的位置，紧接着的 3 位表示任务占据 OSRdyGrp 的第几位。

在 μC/OS-II 中，使任务进入就绪态的方法是：

```
OSRdyGrp|=OSMapTbl[prio>>3];
OSRdyTbl[prio>>3]|=OSMapTbl[prio&0 x 07]
```

其中，(prio>>3) 表示优先级在变量 OSRdyGrp 中的位置；(prio&0x07)表示优先级在变量 OSRdyTbl[ ]中的位置。|=表示将 OSMapTbl[ ]的值与 OSRdyGrp 按位“或”，该方法中，第 1 句的含义是使就绪表变量 OSRdyGrp 置位，第 2 句则使就绪表变量 OSRdyTbl[ ]置位。

任务进入就绪态以后，操作系统根据应用程序，通过调度函数调用已经就绪的任务。由于 CPU 的使用权只能被一个任务占有，此时需要找出就绪表中优先级最高的任务投入运行，这一过程由如下代码表示：

```
y=OSUnMapTbl[OSRdyGrp];
x=OSUnMapTbl[OSRdyTbl[y]];
prio=(y<<3)+x;
```

OSUnMapTbl[ ]表用来查找已经就绪的优先级最高的任务。利用此表，可以不用遍历所有就绪的任务而直接找到就绪了的最高优先级任务，节省系统调度时间，提高 CPU 利用率，增强内核实时性。在该代码段中，第 1 句表示找出就绪任务中最高优先级所处变量 OSRdyGrp 中的位置码 y，第 2 句则找出了该优先级在变量 OSRdyTbl[ ]中的位置码 x，第 3 句根据 y 与 x 求出就绪表中任务的最高优先级。

当任务运行完毕(即任务进入挂起、等待状态时)或者任务被删除，需要将任务从就绪表中删除。具体实现方法：

```
if((OSRdyTbl[prio>>3]&~OSMapTbl[prio&0x07])==0)
    OSRdyGrp &=~OSMapTbl[prio>>3]
```

针对以上 μC/OS-II 操作系统对优先级使用的描述，本文提出了一种将优先级反序的方法，用来与 OSEK/VDX 的操作系统规范相对应。

首先，将就绪表中的内容反序，OSMapTbl[ ]的值保持不变，结果如图 2 所示。

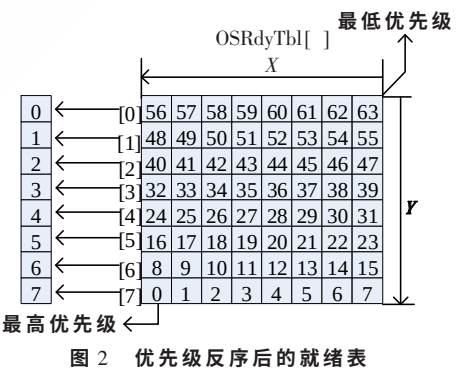

图 2 优先级反序后的就绪表

同时把任务进入就绪态的方法修改为：

```
OSRdyGrp|=OSMapTbl[(~prio)&0 x 3 F>>3];
OSRdyTbl[(OSMapTbl)>>3]|=OSMapTbl[((~prio)&0 x 3 F)& 0 x 07];
```

其次，在 OSUnMapTbl[ ]的值维持原状的基础上，把找出就绪态中优先级最高任务的方法做如下改动：

```
y=OSUnMapTbl[OSRdyGrp];
x=OSUnMapTbl[OSRdy[y]];
prio=((7-y)<<3)+x;
```

最后，任务退出就绪状态所用的方法也需要改变：

```
if ((OSRdyTbl [((~prio)&0 x 3 F)>>3]&=~OSMapTbl
[((~prio)& 0 x 3 F)& 0 x 0 7])==0)
OSRdyGrp&=~OSMapTbl[((~prio)& 0 x 3 F)>>3]
```

3 测试

为了验证系统的可靠性，使用以 MC9S12DP256B 芯片为主的电控单元开发板、背景调试模式的 BDM 调试器[5]和示波器对上面算法的修改做了测试，利用 Codewarrior 编译器将操作系统下载到开发板上。测试实验环境如图 3 所示。

3.1 测试方案

首先在主函数中创建一个优先级为 60 的 MainTask 任务，然后调用 OSStart()函数使操作系统开始运行。在 MainTask 里，先对时钟进行初始化，并创建 2 个优先级分别为 55 和 50 的任务 Taska 和 Taskb，在 Taska 中设置 PORTB_BIT0=1、PORTB_BIT0=0，并持续 50 万个计数值。Taskb 中设置 PORTB_BIT1=1。在 MainTask 任务循环里，先设置 PORTB_BIT2=1，再通过 OSTimeGet()函数为自己设置计数器，当计数器达到预定值时，将自己挂起。

3.2 测试结果

通过观察开发板上相对于 PORTB 口的指示灯，发现 2 灯(对应 PORTB_BIT2 位)先灭(开发板上端口置 1 表示灯灭，因为锁存处于一直灭掉状态)，之后 0 灯(对应 PORTB_BIT0 位)灭亮交替，而 1 灯(对应 PORTB_BIT1 位)一直没有变化(没有执行到该任务，优先级最低)。测试结果如图 4 所示。图中示波器 1 通道表示 Taska 对应的输出口 PORTB_BIT0，保持高电平的 2μs 是执行 PORTB_BIT0=1 这一语句所用的时间，保持低电平的 3.6 μs 是执行持续 50 万个计数值得 PORTB_BIT0=0 语句；示波器 2 通道表示 MainTask 对应的输出口 PORTB_BIT2。测试结果的波形显示与端口灯的显示情况完全符合，说明优先级反序算法是正确的。

图 3 测试实验环境

图 4 优先级测试实验图

本文实现了优先级符合 OSEK/VDX 标准的汽车电子嵌入式操作系统设计，实际测试表明，该系统运行稳定，能够为应用程序提供良好的运行环境，适应于要求反应可靠的汽车控制系统，具有良好的应用前景。

参考文献

[1] 杨国田，白焰.Motorola 68HC12 系列微控制器原理、应用与开发技术[M].北京：中国电力出版社，2003.

[2] LABROSSE J J. 嵌入式实时操作系统 μC/OS-Ⅱ (第 2 版)[M].邵贝贝，译.北京：北京航空航天大学出版社，2003.

[3] 国际标准化组织汽车委员会.OSEK/VDX specifications. ISO 17356-2-2005，汽车电子开放式系统及接口规范[S]. 2005.

[4] 任哲. 嵌入式实时操作系统 μC/OS-II 原理及应用[M]. 北京：北京航空航天大学出版社，2005.

[5] 姚冉中，潘宏侠. μC/OS-II 在 TMS320F2812 上的移植和研究[J].计算机工程与设计，2007，140(2)：162-163.

(收稿日期：2009-02-05)