基于 LPC2134 和 UCOSII 的自动售货机状态机的研究与实现①

罗 信 金 瓯 (中南大学 信息科学与技术学院 湖南 长沙 410085)

摘 要: 由于售货机的功能不断增多，售货机控制系统也相应得不断变得庞大，这就使原来的面向过程的开发方法变得越来难以扩展和维护，根据自动状态机的理论，提出一种在售货机上的状态机模型，该模型对系统状态进行抽象和分离，建立了系统状态表和状态转换表，是一种网状的状态机模型。根据系统事件和当前状态来查找系统状态转换表，如果有匹配项，则进行状态转换。经过实验表明，该模型能解决系统的代码庞大问题，并且使系统开发变得易于维护。

关键词: 售货机控制; 状态机; 状态转换

FSM of Vending Machine Controller Based on Lpc2134 and UcosII

LUO Xin, JIN Ou

(Information and Technology College, Central South University, Changsha 410085, China)

Abstract: Due to the increasing number of functions of the Vending Machine, the VMC(Vending Machine Controller) is becoming more and more complex, so it is becoming more and more difficul to extend and maintain the developing method used before. According to the theory of FSM, this article suggests one FSM Model on the Vending Machine. The abstract model separates and divides the system into ones to establish a list of system states and transformations. If an event occurred, it would search the system transformation list by the system event and the current state. If there is a record that matches the state and event, then the state is transformed to the NextState defined in the record. Based on the experiment, the model can solve the big problem of the system and make the system development easier to maintain.

Keywords: VMC; FSM; state transform

1 概述

自动售货机是一种全新的商业零售形式，20 世纪 70 年代自日本和欧美发展起来。现在，自动售货机产业正在走向信息化并进一步实现合理化。例如实行联机方式，通过 GPRS 将自动售货机内的库存信息及时地传送各营业点的电脑中，从而确保 了商品的发送、补充以及商品选定的顺利进行。并且，为防止地球暖化，自动售货机的开发致力于能源的节省，节能型清凉饮料自动售货机成为该行业的主流。在夏季电力消费高峰时，这种机型的自动售货机即使在关掉冷却器的状况下也能保持低温，与以往的自动售货机相比，它能够节约 10－15％的电力。进入 21 世纪 时，自动售货机也将进一步向节省资源和能源以及高功能化的方向发展。这就导致了售货机的控制系统(Vencling Machine Controller，VMC)的复杂度大大增加[1].先前的基于汇编程序的 VMC 控制系统虽然效率高，但

① 基金项目:国家科技攻关计划(2003ba104c)
收稿时间:2009-12-28;收到修改稿时间:2010-02-08

是存在着可扩展性差，程序结构性差，不易维护等缺点。因此本文提出了一种基于 ARM7 的新型 VMC 主板的有限状态机的结构，对 VMC 进行改进.本文所用的硬件环境为使能 LPC2134 的 ARM7 主板，有一个 RS232 串口接 GPRS 模块，一个 MDB 总线模块，一个电机驱动模块，外接一个显示和按键模块。

软件方面采用的操作系统是 ucosII 操作系统，这是一种开源的实时多任务操作系统，在数据同步和互斥上提供了信号量，消息队列，消息邮箱等功能，本文主要采用的是信号量和消息队列。

总的来说，有限状态机系统，是指在不同阶段会呈现出不同的运行状态的系统，这些状态是有限的、不重叠的。这样的系统在某一时刻一定会处于其所有状态中的一个状态，此时它接收一部分允许的输入，产生一部分可能的响应，并且迁移到一部分可能的状态. 一个有限状态机(FSM)是一个五元组，M=(S，G，E，A，T)[2]。

State(状态)，就是一个系统在其生命周期中某一时刻的运行情况，此时，系统会执行一些动作，或者等待一些外部输入[3]。

Guard(条件),状态机对外部消息进行响应的时候，除了需要判断当前的状态，还要判断跟这个状态相关的一些条件是否成立。这种判断称为 guard（“条件”）。guard 通过允许或者禁止某些操作来影响状态机的行为。

Event（事件），就是在一定的时间和空间上发生的对系统有意义的事情.

Action(动作)，当一个 Event 被状态机系统分发的时候，状态机用 Action(“动作”)来进行响应，比如修改一下变量的值、进行输入输出、产生另外一个 Event 或者迁移到另外一个状态等等。

Transition(迁移)，从一个状态切换到另一个状态被称为 Transition（“迁移”）。引起状态迁移的事件被称为 triggering event （“触发事件”），或者被简称为 trigger(触发）。

有限状态机一般有 2 种表示方式：状态转移表和状态转移图[4]。通常用有向图来表示有限状态机，其节点代表状态。如图 1 所示，售货机售货流程一共分 5 个状态，每个状态都是根据消息的不同来进行转换。

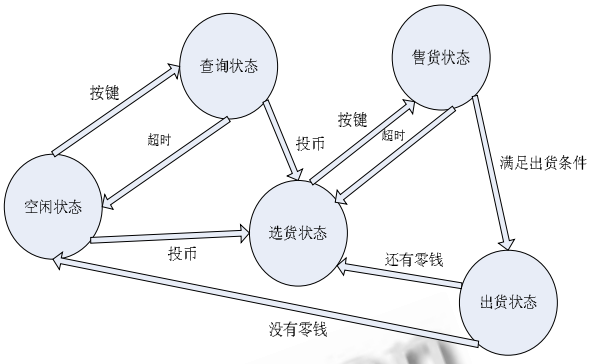

图 1　自动售货机中的状态转换图:

2　实现方式

2.1 嵌套 Switch 方式

```
Enum Signal{KEY_DOWN,COIN_IN,BILL_IN}
Enum State {Origin, Query, Sale, Select, Change}
Switch(state)
{
Case Origin:
    Switch(KEY_DOWN):
    {
      Case KEY_DOWN：{…}...
    }
}
```

即先定义一个状态和消息的枚举，再通过嵌套的 2 层 switch-case 结构实现，外层 switch-case 结构判断状态,虽然这种方法结构简单，便于理解，但是代码冗长，不便维护，因此不建议在复杂杂的状态机下运行.

2.2 状态表方式:

```
Struct Tran{
    Action action;
    Unsigned nextState;
Methods
}
Enum State{ Origin, Query, Sale, Select, Change }
Static Tran const loctable[MAX_STATE][MAX_SIG]=
{ {StateTableDoNothing,Origin},
{(Action*)Action1,Query},
{(Action*)Action2,Sale}},
```

```
    ...
}
```

这种实现方法对第一种方法进行了改良，根据 State,Event 做成二维表格，表格中的项表示 Action 和 Transmission,状态采用枚举量.这种方法结构简单，便于理解，代码比较简练，效率最高，但是代码结构不是太好.

2.3 用函数指针作为状态

```
Typedef void (*State)(Fsm* me,unsigned const sig);
Struct Fsm{State state_};
#define FsmDispatch(me_,sig_) ((*(me_)->state)((me_),sig))
#define TRAN(target_) (((Fsm*)me)-> state_=(State)(target_))
```

如果支持 entry/exit 方法,则可改为:

```
#define TRAN(target_) do{\
(*((Fsm*)me)->state_)((me),EXIT_SIG);\
((Fsm*)me)->state_=(State)(target_);\
(*((Fsm*)me)->state_)((me),ENTRY_SIG);\
}while(0)
```

这种方法用函数地址代替 state 值，比较直观，可以方便地增加 entry/exit 操作，并且效率较高[5]。

3 具体实现

综合以上几种方式，本文提出一种结合第二种和第三种方式的状态机实现方式，即创立一个状态表和一个状态转换表,其中状态表中有 exit,entry,default 处理方法，当进入该状态时,先执行 entry 方法，然后在状态运行时执行 default 方法,退出状态时执行 exit 方法，状态之间的切换是通过状态转换表实现的，即首先状态等待一个系统消息，如果在状态转换表中对应当前状态和当前消息的项，则进行消息转换，具体实现如下

3.1 数据结构

首先定义一个大小为 10 的系统消息队列用来接收外设和系统消息:

```
OS_EVENT *msgqueen;          //系统的消息队列
void *msg[10];
msgqueen=OSQCreate(msg,10);
```

定义消息类型，状态结构，状态转换结构:

```
typedef struct{              //系统消息。所有的消息都由该结构体传递。
INT8U type;                  //消息类型
INT8U value;                 //传递值
}SYSMSG;
enum STATE_ID                //状态枚举
{Origin,Query,Sale,Select,Change,Verify,Set,END_STATE_ID}
typedef void (*FSM_FUNC)(SYSMSG *);
//函数指针
typedef struct               //状态机状态结构
{
enum STATE_ID id;            //状态编号
FSM_FUNC enter_func;         //进入操作
FSM_FUNC exit_func;          //退出操作
FSM_FUNC default_func;       //缺省操作
}FSM_STATE;
typedef struct
{
uint8 MsgType;               //转换的消息
enum STATE_ID NextStateID;   //转换后的状态
FSM_FUNC tran_func;//转换函数
FSM_STATE_TRAN *next;        //下一个转换信息
}FSM_STATE_TRAN;
```

3.2 状态转换算法:

先建立一个状态转换表，建立一个 FSM_STATE_TRAN 指针数组.即 FSM_STATE_TRAN *fsm_tran[7];将每个状态下的转换项都存在该状态下的链表中。

在有限状态机中，状态是通过系统消息和当前状态来查找状态转换表，如果有对应的项则进行转换，没有则继续当前状态：算法如下:

```
Void fsm_do_event(FSM_STATE *State, SYSMSG sysmsg)
{
FSM_STATE_TRAN *Temp= fsm_tran[State->id]->next;
if(State->default_func)      //执行当前状态的默认方法
```

```
State->default_func(&sysmsg);
sysmsg=* (WaitSYSMSG(5000,&err));  //等待 5S 超时
if(err!=OS_NO_ERR)
sysmsg.type=TIME_OUT;
//查找状态转换表一直到最后一条记录
while(Temp!=NULL )
{
//如果状态转换表中有对应项则进行状态转换
if(Temp->MsgType==sysmsg.type)
{
if(State->exit_func)
State->exit_func(&sysmsg);  //退出当前状态
if(Temp->tran_func)
Temp->tran_func(&sysmsg);//状态转换
State=Temp;
if(State->enter_func)
State->enter_func(&sysmsg); //进入新的状态
Break;
}
Temp=Temp->next;  //查找下一个
}
}
```

4　模型分析

4.1 扩展性

为状态机添加新状态，只需在状态表中添加新状态及其处理方法，以及在状态转换表中添加该状态的转换方式。

4.2 查找算法分析

设总共有 n 个状态，m 个事件，每个状态平均响应的事件为 k。

由于在这里每个状态都是互斥的，因此状态转换时的时间开销主要是花费在查找状态表上面，这里采用了基数排序查找的思想。因此主要查找开销是每个状态的状态转换个数。由于在售货机中，每个状态大概都只有 3-4 个状态转换表。因此查找的时间复杂度比 O(m)小.

5　结语

通过建立有限状态机模型，并应用改进的数据结构与状态转换算法，自动售货机控制器的程序结构更为清晰。原来存在于程序中的诸多标志变量，由状态机的各个状态所取代，使系统具有更好的扩展性；并且模型很好地利用了状态的相关性，缩短了查找所花费的时间。应用于别的嵌入式系统也有较高的意义。

参考文献

1 蒋卫星,金瓯. 基于 GPRS 售货机监控管理系统的研究与设计.计算机与数字工程, 2007,35(2):113－116.

2 唐勇,张欣,周明天.一种基于 FSM 的告警事件关联方法.计算机应用研究, 2006,23(9):243－246.

3 吴浩,彭鑫,赵文耕.一种基于分支条件的对象状态机自动提取方法.小型微型计算机系统, 2007,28(1):1－6.

4 魏先民.有限状态机在嵌入式软件中的应用.潍坊学院学报, 2006,6(4):24－25.

5 姚鑫骅,潘雪增,傅建中等.微制造数控系统的实时有限状态机建模研究.浙江大学学报(工学版), 2005,39(12):1965－1968.