一，状态机图及其分解描述

Eve，nts：

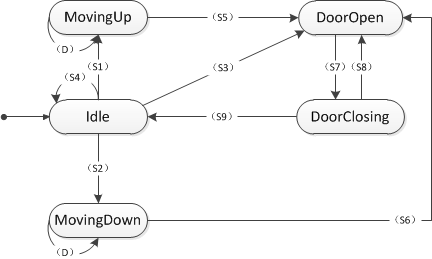
E1: 门内开门按钮(OpenDoorLight)

E2: 门内关门按钮(CloseDoorLight)

E3: 门内楼层按钮(PanelFloorLight)

E4: 门外up呼叫按钮(Call Light)

E5: 门外down呼叫按钮(Call Light)



**Idle状态，电梯停止在某楼层，门是关闭的，处于静止状态，等待相关事件的发生，从而转换到下一个状态。**

**Idle 🡪 MovingUp**

**(S1) 检查E3、E4、E5事件，已经封装在下列函数中。**

**静态检测(检测将要到的目标楼层),**

**int floor; bool up;**

**目标楼层floor=IdleWhatFloorToGoTo(&up); // Event**

**if (floor > 0 && up) {**

**上升SetMotorPower(1)。进入MovingUp状态。// Transition**

**}**

**Idle 🡪 MovingDown**

**(S2)检查E3、E4、E5事件，已经封装在下列函数中。**

**静态检测(检测将要到的目标楼层),**

**int floor; bool up;**

**目标楼层floor=IdleWhatFloorToGoTo(&up); // Event**

**if (floor > 0 && !up) {**

**上升SetMotorPower(-1)。进入MovingDown状态。// Transition**

**}**

**Idle 🡪 DoorOpen**

**(S3)**

1. **检查E1事件GetOpenDoorLight(), 开门SetDoor(), 消费开门按钮SetOpenDoorLight(); 进入DoorOpen状态。**
2. **检查E4/E5事件GetCallLight(), 开门SetDoor(),消费门外up/down按钮SetCallLight(); 进入DoorOpen状态。**

**Moving Up状态：MovingUp 🡪 DoorOpen**

**(S5) 检查E3、E4、E5事件，已封装在下列函数中。**

**动态检测，目标楼层floor=GoingUpToFloor();**

**if(fabs(GetFloor() - floor) < Lib\_FloorTolerance) 到达目标楼层，停止SetMotorPower(0)，开门SetDoor();进入DoorOpen状态。**

**消费门外up按钮SetCallLight(); 到了最高层Lib\_FloorNum, 消费门外down按钮。**

**消费门内楼层按钮为当前楼层的按钮SetPanelFloorLight()。**

**GetNearestFloor(); // 获取当前楼层**

**Moving Up状态: 安全设置，无动作。**

**(D) 运动状态，开关门失效，检查E1、E2事件GetOpenDoorLight()/GetCloseDoorLight()，无动作，消费开Moving Down状态：MovingDown 🡪 DoorOpen**

**(S6) 检查E3、E4、E5事件，已封装在下列函数中。**

**动态检测，目标楼层floor=GoingDownToFloor();**

**if(fabs(GetFloor() - floor) < Lib\_FloorTolerance) 到达目标楼层，停止SetMotorPower(0)，开门SetDoor();进入DoorOpen状态。**

**消费门外down按钮SetCallLight(); 到了1层, 消费门外up按钮SetUpLight()。**

**消费门内楼层按钮为当前楼层的按钮SetPanelFloorLight()。**

**GetNearestFloor(); // 获取当前楼层**

**/关Moving Down状态: 安全设置，无动作。**

**(D) 运动状态，开关门失效，检查E1、E2事件GetOpenDoorLight()/GetCloseDoorLight()，无动作，消费开/关门按钮SetOpenDoorLight()/SetCloseDoorLight()。**

**门按钮SetOpenDoorLight()/SetCloseDoorLight()**

**Door Closing状态: DoorClosing 🡪 DoorOpen**

**(S8) GetNearestFloor()获取当前楼层**

1. **检查E1事件GetOpenDoorLight(); 正在关门，按了开门灯，转而开门SetDoor(); 消费开门按钮SetOpenDoorLight()。进入DoorOpen状态。**
2. **检查E2事件GetCloseDoorLight()，正在关门，按了关门灯，无动作，消费关门按钮SetCloseDoorLight()。**
3. **如果红外探测到遮挡 IsBeamBroken()，转而开门SetDoor()，进入DoorOpen状态。**

**DoorClosing状态: DoorClosing 🡪 Idle**

**(S9) 关门结束后IsDoorClosed(); 进入Idle状态。**

**二，状态机代码**

#include "stdafx.h"

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include "ElevatorLib.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Idle状态，电梯停止在某楼层，门是关闭的，处于静止状态，等待相关事件的发生，从而转换到下一个状态。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void StateIdle(int \*state)

{

int floor;

bool up;

floor = WhichFloorToGoTo(&up);

if(floor>=1)

{

if(down)

{

\*state=MovingDown;

SetCallLight(floor, false, false);

}

else

{

\*state=MovingUp;

SetCallLight(floor, true, false);

}

if (GetOpenDoorlight()= true)

{

SetDoor(GetNearnestFloor(), true);

SetOpenDoorLight(false);

\*state = DoorOpen;

}

if (GetCloseDoorLight())

{

SetCloseDoorLight(false);

return;

}

}

void StateMovingUp(int \*state)

{

int floor;

floor=GoingUpToFloor();

if(fabs(GetFloor()-floor)<Lib\_FloorTolerance)

{

SetDoor(floor,true);

SetCallLight(floor,true,false);

SetPaneFloorLight(floor,false);

\*state=DooeOpen;

}

else

{

SetCallLight(Lib\_FloorNum,false);

SetPanelFLoorLight(Lib\_FloorNum, false);

}

if (GetOpenDoorLight())

SetOpenDoorLight(false);

if (GetCloseLight())

SetCloseDoorLight(false);

}

void StateMovingDown(int \*state)

{

int floor;

floor=GoingDownToFloor();

if(fabs(GetFloor()-floor)<Lib\_FloorTolerance)

{

SetDoor(floor,true);

SetCallLight(floor,false);

SetPaneFloorLight(floor,false);

\*state=DooeOpen;

}

if(fabs(GetFloor()-1)<Lib\_FloorTolerance)

{

SetCallLight(1,true,false);

SetPaneFloorLight(1,false);

}

if (GetOpenDoorLight())

SetOpenDoorLight(false);

if (GetCloseDoorLight())

SetCloseDoorLight(false);

}

void StateDoorOpen(int \*state)

{

int floor;

floor= GettheFloor();

if(GetCloseDoorLight()

{

SetDoor(GetNestFloor(),false);

SetCloseDoorLight(false);

}

if (GetOpenDoorLight())

SetOpenDoorLight(false;);

if(IsDoorOpen(floor))

{

SetDoor(floor,false);

\*state=DoorClosing;

}

}

void StateDoorClosing(int \*state)

{

if(GetOpenDoorLight())

{

SetDoor(GetNestFloor(),true);

SetOpenDoorLight(false);

\*state=DoorOpen;

}

if(GetCloseDoorLight())

{

SetCloseDoorLight(false);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 状态机，每隔一定时间(如，100ms)被调用一次，采集系统的运行状态

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void main\_control(int \*state)

{

if(IsElevatorRunning()) // 仿真正在运行

{

switch(\*state)

{

case Idle:

// Idle状态，一定时间无动作，自动到一楼

if(GetNearestFloor() !=1 ) {

AutoTo1Floor();

}

StateIdle(state);

break;

case MovingUp:

CancelTo1Floor(); // 其它状态，取消自动到一楼

StateMovingUp(state);

break;

case MovingDown:

CancelTo1Floor();

StateMovingDown(state);

break;

case DoorOpen:

CancelTo1Floor();

StateDoorOpen(state);

break;

case DoorClosing:

CancelTo1Floor();

StateDoorClosing(state);

break;

default:

printf("没有这种状态!!!\n");

}

}

}

三，运行测试

1. 电梯停于2F或3F时，按1F向上呼叫按钮；电梯下降到1F停止，开门/关门。电梯停于1F或2F时，按3F向下呼叫按钮；电梯上升到3F停止，开门/关门。
2. 电梯停于1F时，按2F向上呼叫按钮；电梯上升到2F停止，开门/关门。
3. 电梯停于3F时，按2F向下呼叫按钮；电梯下降到2F停止，开门/关门。
4. 电梯停于1F，2F和3F均有按钮呼叫；电梯先上升到2F，开门/关门，然后上升到3F停止，开门/关门。
5. 电梯停于3F，2F和1F均有按钮呼叫；电梯先下降到2F，开门/关门，然后下降到1F停止，开门/关门。
6. 电梯上升途中或下降途中，任何反方向按钮呼叫均无效。
7. 电梯停于1F，按3F向下呼叫按钮，然后立即按2F向下呼叫按钮；电梯上升到3F停止，开门/关门，然后下降到2F停止，开门/关门。
8. 电梯停于2F，按门内楼层按钮3，然后门内楼层按钮1；电梯上手到3F停止，开门/关门，然后下降到1F，开门/关门。
9. 电梯停于1F，按门内楼层按钮3，当电梯上升在1F到2F中间以下，按2F向上呼叫按钮；电梯先上升到2F，开门/关门；然后再到3F，开门/关门。
10. 电梯停于1F，按门内楼层按钮3，当电梯上升在1F到2F中间以上，按2F向上呼叫按钮；电梯先上升到3F，开门/关门；然后再到2F，开门/关门。
11. 电梯停于1F，按2F向下呼叫按钮和向上呼叫按钮以及3F的向下呼叫按钮；电梯上升到2F停止，开门/关门，2F的向上呼叫按钮灯关闭，开门/关门， 2F的向下呼叫按钮灯关闭，然后上升到3F停止，开门/关门，3F的向下呼叫按钮关闭。
12. 电梯在2F以上，10s无动作，自动降到1楼。
13. 所有停止，开门/关门后，对应楼层的同方向门外呼叫按钮灯（最高楼向下呼叫按钮，最底层向上呼叫按钮）和门内楼层按钮灯关闭。
14. 运动状态，开关门按钮失效。正在开门，开门按钮失效；正在关门，关门按钮失效。
15. 空闲状态，门是关闭的，因此按关门按钮失效。
16. 开关门结束，延时2秒用于乘客上下电梯【延时功能在库函数中实现，不用在状态函数中实现】，然后进入关门状态。
17. 开门结束前，按关门按钮，转而进入关门状态；关门结束前，按开门按钮，转而进入开门状态。