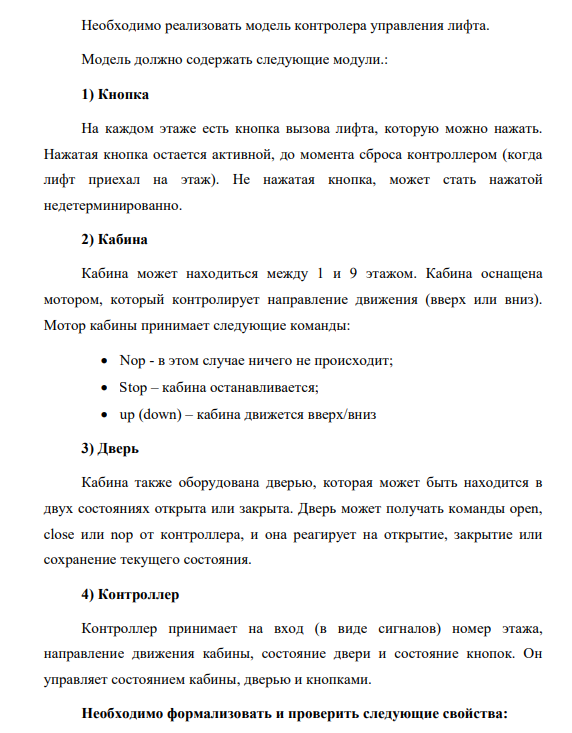
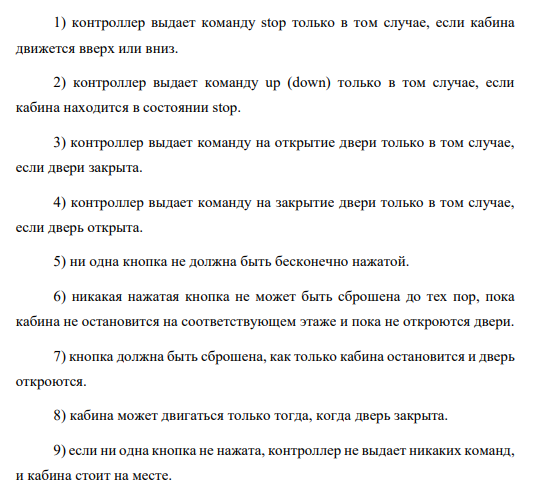
**Постановка задачи**





**Ход работы**

1. Контроллер выдает команду stop только в том случае, если кабина движется вверх или вниз

***AG (controller.cabinCommand = stop -> cabin.state in { movingUp, movingDown })***

Свойство выполняется

1. Контроллер выдает команду up (down) только в том случае, если кабина находится в состоянии stop

***AG (controller.cabinCommand in { moveUp, moveDown } -> cabin.state = standing)*** ******

Свойство выполняется

1. Контроллер выдает команду на открытие двери только в том случае, если двери закрыта

***AG (controller.doorCommand = open -> door.status = closed)***



1. контроллер выдает команду на закрытие двери только в том случае, если дверь открыта

***AG (controller.doorCommand = close -> door.status = open)*** ******

1. ни одна кнопка не должна быть бесконечно нажатой

***AG (AF (!b1.pressed) & AF (!b2.pressed) & AF (!b3.pressed) & AF (!b4.pressed) & AF (!b5.pressed) & AF (!b6.pressed) & AF (!b7.pressed) & AF (!b8.pressed) & AF (!b9.pressed))*** ******

1. никакая нажатая кнопка не может быть сброшена до тех пор, пока кабина не остановится на соответствующем этаже и пока не откроются двери

***AG (***

***(b1.pressed -> (A [b1.pressed U (cabin.floor = 1 & door.status = open)])) &***

***(b2.pressed -> (A [b2.pressed U (cabin.floor = 2 & door.status = open)])) &***

***(b3.pressed -> (A [b3.pressed U (cabin.floor = 3 & door.status = open)])) &***

***(b4.pressed -> (A [b4.pressed U (cabin.floor = 4 & door.status = open)])) &***

***(b5.pressed -> (A [b5.pressed U (cabin.floor = 5 & door.status = open)])) &***

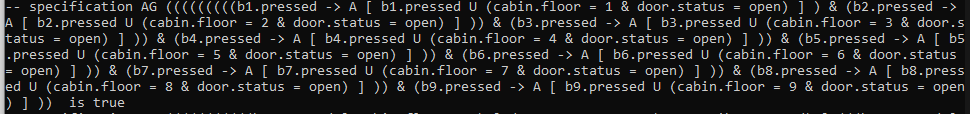
***(b6.pressed -> (A [b6.pressed U (cabin.floor = 6 & door.status = open)])) &***

***(b7.pressed -> (A [b7.pressed U (cabin.floor = 7 & door.status = open)])) &***

***(b8.pressed -> (A [b8.pressed U (cabin.floor = 8 & door.status = open)])) &***

***(b9.pressed -> (A [b9.pressed U (cabin.floor = 9 & door.status = open)]))***

***)***



1. кнопка должна быть сброшена, как только кабина остановится и дверь откроются

***AG (***

***((b1.pressed & cabin.floor = 1 & door.status = open) -> AX !b1.pressed) &***

***((b2.pressed & cabin.floor = 2 & door.status = open) -> AX !b2.pressed) &***

***((b3.pressed & cabin.floor = 3 & door.status = open) -> AX !b3.pressed) &***

***((b4.pressed & cabin.floor = 4 & door.status = open) -> AX !b4.pressed) &***

***((b5.pressed & cabin.floor = 5 & door.status = open) -> AX !b5.pressed) &***

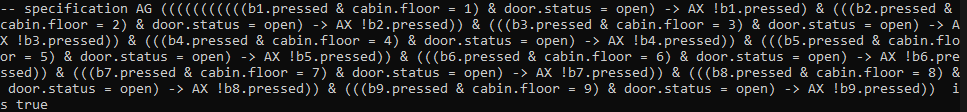
***((b6.pressed & cabin.floor = 6 & door.status = open) -> AX !b6.pressed) &***

***((b7.pressed & cabin.floor = 7 & door.status = open) -> AX !b7.pressed) &***

***((b8.pressed & cabin.floor = 8 & door.status = open) -> AX !b8.pressed) &***

***((b9.pressed & cabin.floor = 9 & door.status = open) -> AX !b9.pressed)***

***)***



1. кабина может двигаться только тогда, когда дверь закрыта

***AG (((cabin.state = moveUp) | (cabin.state = moveDown)) -> door.status = closed)*** ******

1. если ни одна кнопка не нажата, контроллер не выдает никаких команд, и кабина стоит на месте

***AG ((!b1.pressed & !b2.pressed & !b3.pressed & !b4.pressed & !b5.pressed & !b6.pressed & !b7.pressed & !b8.pressed & !b9.pressed) -> (controller.doorCommand = nop & controller.cabinCommand = nop & cabin.state = standing))*** ******

**Программа**

MODULE Button(reset)

VAR

pressed : boolean;

ASSIGN

init(pressed) := FALSE;

next(pressed) :=

case

pressed & reset : FALSE;

pressed & !reset : TRUE;

!pressed : { FALSE, TRUE };

esac;

MODULE Cabin(cabinCommand)

VAR

floor : { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

state : { standing, movingUp, movingDown };

ASSIGN

init(state) := standing;

next(state) :=

case

cabinCommand = stop : standing;

cabinCommand = moveUp : movingUp;

cabinCommand = moveDown : movingDown;

cabinCommand = nop : state;

esac;

next(floor) :=

case

next(state) = standing : floor;

next(state) = movingUp :

case

floor = 9 : 9;

TRUE : floor + 1;

esac;

next(state) = movingDown :

case

floor = 1 : 1;

TRUE : floor - 1;

esac;

esac;

MODULE Door(doorCommand)

VAR

status : { open, closed };

ASSIGN

next(status) :=

case

doorCommand = open : open;

doorCommand = close : closed;

doorCommand = nop : status;

esac;

MODULE Controller(floor, cabinState, doorState, pressed1, pressed2, pressed3, pressed4, pressed5, pressed6, pressed7, pressed8, pressed9)

VAR

cabinCommand : { moveUp, moveDown, stop, nop };

doorCommand : { open, close, nop };

reset1 : boolean;

reset2 : boolean;

reset3 : boolean;

reset4 : boolean;

reset5 : boolean;

reset6 : boolean;

reset7 : boolean;

reset8 : boolean;

reset9 : boolean;

ASSIGN

reset1 := (pressed1 & floor = 1 & doorState = open);

reset2 := (pressed2 & floor = 2 & doorState = open);

reset3 := (pressed3 & floor = 3 & doorState = open);

reset4 := (pressed4 & floor = 4 & doorState = open);

reset5 := (pressed5 & floor = 5 & doorState = open);

reset6 := (pressed6 & floor = 6 & doorState = open);

reset7 := (pressed7 & floor = 7 & doorState = open);

reset8 := (pressed8 & floor = 8 & doorState = open);

reset9 := (pressed9 & floor = 9 & doorState = open);

VAR

prevState : { movingUp, movingDown };

ASSIGN

next(prevState) :=

case

cabinState = standing : prevState;

TRUE : cabinState;

esac;

DEFINE

stay := (floor = 1 & pressed1) | (floor = 2 & pressed2) |

(floor = 3 & pressed3) | (floor = 4 & pressed4) |

(floor = 5 & pressed5) | (floor = 6 & pressed6) |

(floor = 7 & pressed7) | (floor = 8 & pressed8) |

(floor = 9 & pressed9);

goUp := (floor = 1 & (pressed2 | pressed3 | pressed4 | pressed5 | pressed6 | pressed7 | pressed8 | pressed9)) |

(floor = 2 & (pressed3 | pressed4 | pressed5 | pressed6 | pressed7 | pressed8 | pressed9)) |

(floor = 3 & (pressed4 | pressed5 | pressed6 | pressed7 | pressed8 | pressed9)) |

(floor = 4 & (pressed5 | pressed6 | pressed7 | pressed8 | pressed9)) |

(floor = 5 & (pressed6 | pressed7 | pressed8 | pressed9)) |

(floor = 6 & (pressed7 | pressed8 | pressed9)) |

(floor = 7 & (pressed8 | pressed9)) |

(floor = 8 & pressed9);

goDown := (floor = 9 & (pressed8 | pressed6 | pressed7 | pressed5 | pressed4 | pressed3 | pressed2 | pressed1)) |

(floor = 8 & (pressed7 | pressed6 | pressed5 | pressed4 | pressed3 | pressed2 | pressed1)) |

(floor = 7 & (pressed6 | pressed5 | pressed4 | pressed3 | pressed2 | pressed1)) |

(floor = 6 & (pressed5 | pressed4 | pressed3 | pressed2 | pressed1)) |

(floor = 5 & (pressed4 | pressed3 | pressed2 | pressed1)) |

(floor = 4 & (pressed3 | pressed2 | pressed1)) |

(floor = 3 & (pressed2 | pressed1)) |

(floor = 2 & pressed1);

ASSIGN

doorCommand :=

case

cabinState != standing : nop;

stay & doorState = closed : open;

goUp & doorState = open : close;

goDown & doorState = open : close;

TRUE : nop;

esac;

ASSIGN

cabinCommand :=

case

doorState = open : nop;

stay :

case

cabinState != standing : stop;

TRUE : nop;

esac;

goDown & goUp :

case

cabinState != standing : nop;

prevState = movingUp : moveUp;

prevState = movingDown : moveDown;

esac;

goUp :

case

cabinState != standing : nop;

TRUE : moveUp;

esac;

goDown :

case

cabinState != standing : nop;

TRUE : moveDown;

esac;

TRUE : nop;

esac;

MODULE main

VAR

cabin : Cabin(controller.cabinCommand);

door : Door(controller.doorCommand);

b1 : Button(controller.reset1);

b2 : Button(controller.reset2);

b3 : Button(controller.reset3);

b4 : Button(controller.reset4);

b5 : Button(controller.reset5);

b6 : Button(controller.reset6);

b7 : Button(controller.reset7);

b8 : Button(controller.reset8);

b9 : Button(controller.reset9);

controller : Controller(cabin.floor, cabin.state, door.status, b1.pressed, b2.pressed, b3.pressed, b4.pressed, b5.pressed, b6.pressed, b7.pressed, b8.pressed, b9.pressed);

-- "Контроллер выдает команду stop только в том случае, если кабина движется вверх или вниз"

CTLSPEC AG (controller.cabinCommand = stop -> cabin.state in { movingUp, movingDown })

-- "Контроллер выдает команду up (down) только в том случае, если кабина находится в состоянии stop"

CTLSPEC AG (controller.cabinCommand in { moveUp, moveDown } -> cabin.state = standing)

-- "Контроллер выдает команду на открытие двери только в том случае, если дверь закрыта"

CTLSPEC AG (controller.doorCommand = open -> door.status = closed)

-- "Контроллер выдает команду на закрытие двери только в том случае, если дверь открыта"

CTLSPEC AG (controller.doorCommand = close -> door.status = open)

-- "Ни одна кнопка не должна быть бесконечно нажатой"

CTLSPEC AG (AF (!b1.pressed) & AF (!b2.pressed) & AF (!b3.pressed) & AF (!b4.pressed) & AF (!b5.pressed) & AF (!b6.pressed) & AF (!b7.pressed) & AF (!b8.pressed) & AF (!b9.pressed))

-- "Никакая нажатая кнопка не может быть сброшена до тех пор, пока кабина не остановится на соответствующем этаже и пока не откроются двери"

CTLSPEC AG (

(b1.pressed -> (A [b1.pressed U (cabin.floor = 1 & door.status = open)])) &

(b2.pressed -> (A [b2.pressed U (cabin.floor = 2 & door.status = open)])) &

(b3.pressed -> (A [b3.pressed U (cabin.floor = 3 & door.status = open)])) &

(b4.pressed -> (A [b4.pressed U (cabin.floor = 4 & door.status = open)])) &

(b5.pressed -> (A [b5.pressed U (cabin.floor = 5 & door.status = open)])) &

(b6.pressed -> (A [b6.pressed U (cabin.floor = 6 & door.status = open)])) &

(b7.pressed -> (A [b7.pressed U (cabin.floor = 7 & door.status = open)])) &

(b8.pressed -> (A [b8.pressed U (cabin.floor = 8 & door.status = open)])) &

(b9.pressed -> (A [b9.pressed U (cabin.floor = 9 & door.status = open)]))

)

-- "Кнопка должна быть сброшена, как только кабина остановится и дверь откроется"

CTLSPEC AG (

((b1.pressed & cabin.floor = 1 & door.status = open) -> AX !b1.pressed) &

((b2.pressed & cabin.floor = 2 & door.status = open) -> AX !b2.pressed) &

((b3.pressed & cabin.floor = 3 & door.status = open) -> AX !b3.pressed) &

((b4.pressed & cabin.floor = 4 & door.status = open) -> AX !b4.pressed) &

((b5.pressed & cabin.floor = 5 & door.status = open) -> AX !b5.pressed) &

((b6.pressed & cabin.floor = 6 & door.status = open) -> AX !b6.pressed) &

((b7.pressed & cabin.floor = 7 & door.status = open) -> AX !b7.pressed) &

((b8.pressed & cabin.floor = 8 & door.status = open) -> AX !b8.pressed) &

((b9.pressed & cabin.floor = 9 & door.status = open) -> AX !b9.pressed)

)

-- "Кабина может двигаться только тогда, когда дверь закрыта"

CTLSPEC AG (((cabin.state = moveUp) | (cabin.state = moveDown)) -> door.status = closed)

-- "Если ни одна кнопка не нажата, контроллер не выдает никаких команд, и кабина стоит на месте"

CTLSPEC AG ((!b1.pressed & !b2.pressed & !b3.pressed & !b4.pressed & !b5.pressed & !b6.pressed & !b7.pressed & !b8.pressed & !b9.pressed) -> (controller.doorCommand = nop & controller.cabinCommand = nop & cabin.state = standing))