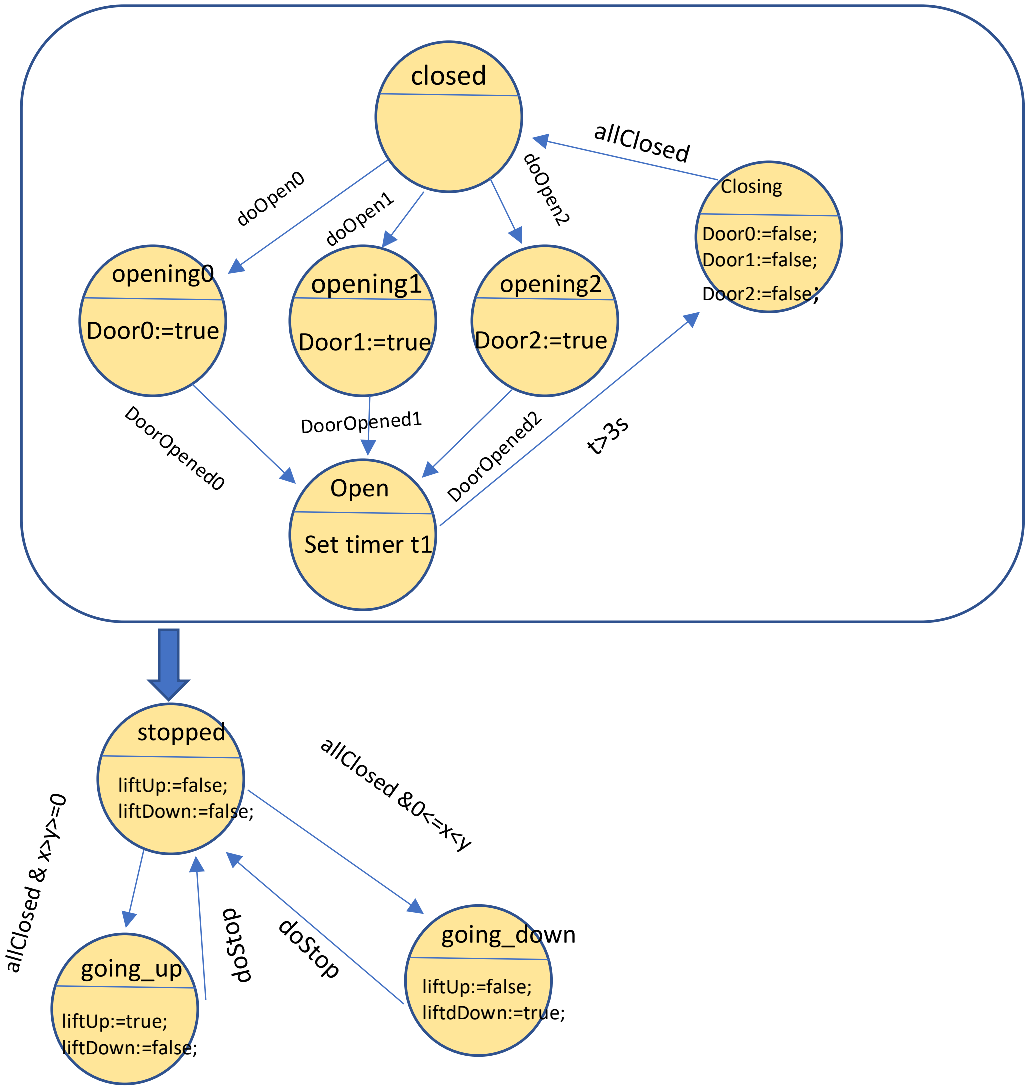
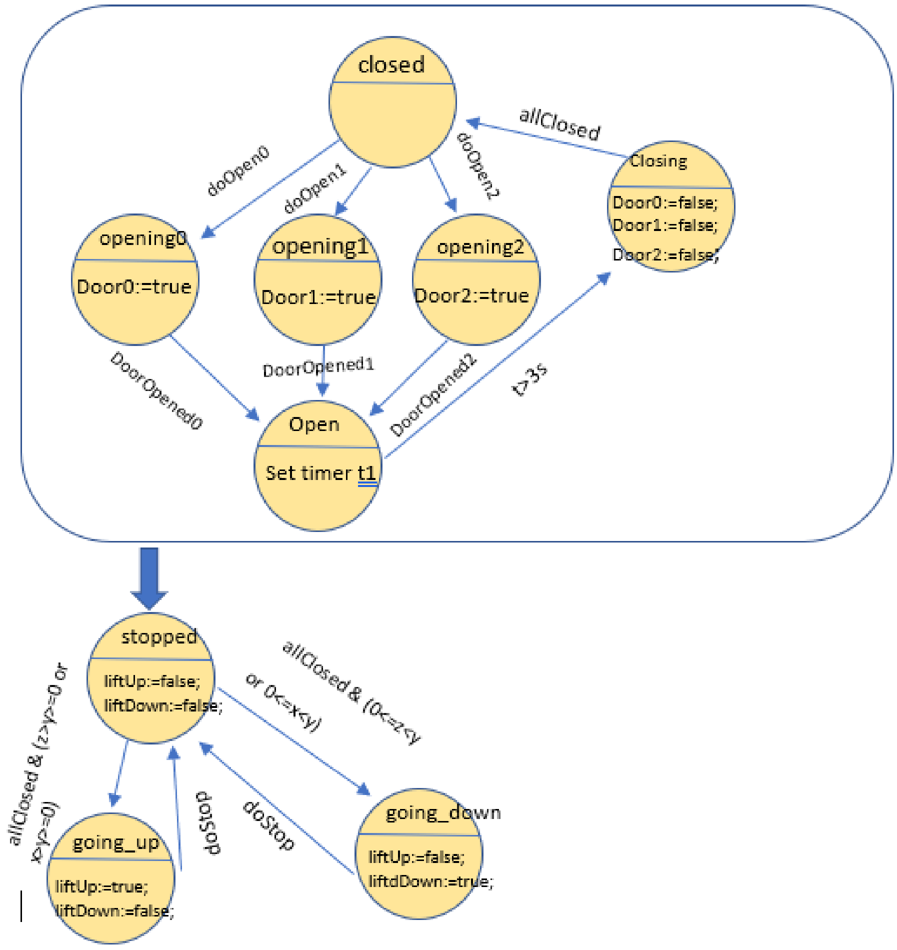
Assignment 1. State-based PLC Programming with Timers **GROUP NUMBER 2**



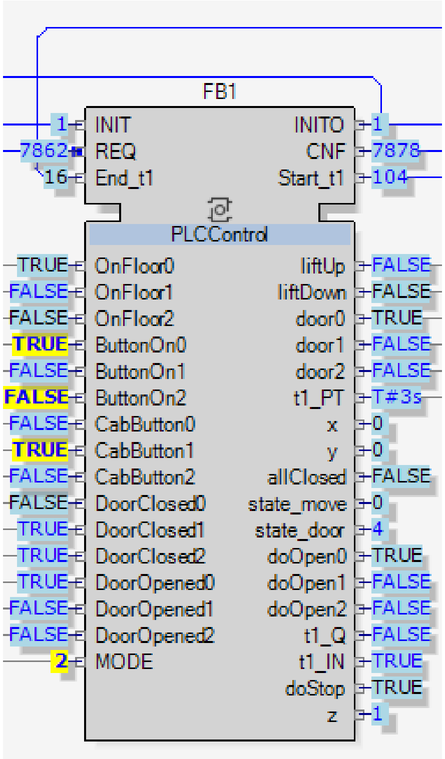
First controller works by checking if buttons are pressed. If they are pressed then the state is set to move to up or down, depending on if the button pressed was lower or higher. After this doors are opened for 3 seconds.



First diagram is for mode1 and second one for mode2.

All of the code for REQ1 and REQ2 is below.

We can test algorithms 2’s functionality by forcing the states as HMI doesn’t accept more than one input.



```
                       FB1
     1  INIT                      INITO  1
  7862  REQ                         CNF  7878
    16  End_t1                  Start_t1  104
                   PLCControl
  TRUE  OnFloor0                 liftUp  FALSE
 FALSE  OnFloor1               liftDown  FALSE
 FALSE  OnFloor2                  door0  TRUE
  TRUE  ButtonOn0                 door1  FALSE
 FALSE  ButtonOn1                 door2  FALSE
 FALSE  ButtonOn2                 t1_PT  T#3s
 FALSE  CabButton0                    x  0
  TRUE  CabButton1                    y  0
 FALSE  CabButton2            allClosed  FALSE
 FALSE  DoorClosed0          state_move  0
  TRUE  DoorClosed1          state_door  4
  TRUE  DoorClosed2             doOpen0  TRUE
  TRUE  DoorOpened0             doOpen1  FALSE
 FALSE  DoorOpened1             doOpen2  FALSE
 FALSE  DoorOpened2                t1_Q  FALSE
     2  MODE                      t1_IN  TRUE
                                 doStop  TRUE
                                      z  1
```

```
IF CabButton0 THEN
   z:= 0;
ELSIF CabButton1 THEN
   z:= 1;
ELSIF CabButton2 THEN
   z:= 2;
ELSE
   z:=-2;
END_IF;
```

```
IF state_move = STATE_STOPPED THEN
   liftUp := FALSE; liftDown:= FALSE;
   IF allClosed AND z < y AND z >= 0 THEN
   state_move:= STATE_GOING_DOWN;
   ELSIF allClosed AND z > y AND z >= 0 THEN
   state_move:= STATE_GOING_UP;
   ELSIF allClosed AND x < y AND x >= 0 THEN
      state_move:= STATE_GOING_DOWN;
   ELSIF allClosed AND x > y AND x >= 0 THEN
      state_move:= STATE_GOING_UP;
   END_IF;
```

Priority is implemented by having Cabbutton variable z before floor button x, in the if statements.

While we are on floor 2 lets force button on from floor 0. Then while the elevator is moving, we can force Cabbutton1 on. We can observe that the elevator stops on floor 1 before continuing to floor 0 after we turn cabbutton1 off, thus Cabbuttons have priority over floor buttons.

ALGORITHM REQ1 IN ST:

```
allClosed := DoorClosed0 AND DoorClosed1 AND DoorClosed2;

IF ButtonOn0 OR CabButton0 THEN
   x:= 0;
ELSIF ButtonOn1 OR CabButton1 THEN
   x:= 1;
ELSIF ButtonOn2 OR CabButton2 THEN
   x:= 2;
ELSE
   x:=-2;
END_IF;

IF OnFloor0 THEN
   y:= 0;
ELSIF OnFloor1 THEN
   y:=1;
ELSIF OnFloor2 THEN
   y:=2;
ELSE
   y:=-2;
END_IF;
doStop := (y=0 AND (ButtonOn0 OR CabButton0)) OR (y=1 AND (ButtonOn1 OR CabButton1)) OR (y=2 AND (ButtonOn2 OR
CabButton2));
doOpen0 := y=0 AND (ButtonOn0 OR CabButton0);
doOpen1 := y=1 AND (ButtonOn1 OR CabButton1);
doOpen2 := y=2 AND (ButtonOn2 OR CabButton2);
```

```
IF state_move = STATE_STOPPED THEN
   liftUp := FALSE; liftDown:= FALSE;
   IF allClosed AND x < y AND x >= 0 THEN
      state_move:= STATE_GOING_DOWN;
   ELSIF allClosed AND x > y AND x >= 0 THEN
      state_move:= STATE_GOING_UP;
   END_IF;
ELSIF state_move = STATE_GOING_UP THEN
   liftUp:=TRUE; liftDown:= FALSE;
   IF doStop THEN
      state_move := STATE_STOPPED;
   END_IF;
ELSIF state_move = STATE_GOING_DOWN THEN
   liftDown:= TRUE; liftUp:= FALSE;
   IF doStop THEN
      state_move := STATE_STOPPED;
   END_IF;
END_IF;

IF state_door = STATE_CLOSED THEN
   IF doOpen0 THEN
      state_door := STATE_OPENING0;
   ELSIF doOpen1 THEN
      state_door := STATE_OPENING1;
   ELSIF doOpen2 THEN
      state_door := STATE_OPENING2;
   END_IF;
ELSIF state_door = STATE_OPENING0 THEN
   door0:= TRUE;
   IF DoorOpened0 THEN
      state_door := STATE_OPEN;
   END_IF;
ELSIF state_door = STATE_OPENING1 THEN
   door1:=TRUE;
   IF DoorOpened1 THEN
      state_door := STATE_OPEN;
   END_IF;
ELSIF state_door = STATE_OPENING2 THEN
   door2:=TRUE;
   IF DoorOpened2 THEN
      state_door := STATE_OPEN;
   END_IF;
ELSIF state_door = STATE_OPEN THEN
   t1_IN:=TRUE;
   t1_PT:=T#3S;
   Start_t1:=TRUE;
   IF t1_Q  THEN
      t1_IN:=FALSE;
      End_t1:=TRUE;
      t1_Q := FALSE;
      state_door := STATE_CLOSING;
   END_IF;
ELSIF state_door = STATE_CLOSING THEN
   door0:= FALSE;
   door1:= FALSE;
   door2:= FALSE;
   IF allClosed THEN
      state_door := STATE_CLOSED;
   END_IF;
END_IF;

END_ALGORITHM

ALGORITHM REQ2 IN ST:

allClosed := DoorClosed0 AND DoorClosed1 AND DoorClosed2;

IF CabButton0 THEN
   z:= 0;
ELSIF CabButton1 THEN
   z:= 1;
ELSIF CabButton2 THEN
   z:= 2;
ELSE
   z:=-2;
END_IF;

IF ButtonOn0 THEN
   x:= 0;
ELSIF ButtonOn1 THEN
   x:= 1;
ELSIF ButtonOn2 THEN
   x:= 2;
ELSE
   x:=-2;
END_IF;

IF OnFloor0 THEN
   y:= 0;
ELSIF OnFloor1 THEN
   y:=1;
ELSIF OnFloor2 THEN
   y:=2;
ELSE
   y:=-2;
END_IF;

doStop := (y=0 AND (ButtonOn0 OR CabButton0)) OR (y=1 AND (ButtonOn1 OR CabButton1)) OR (y=2 AND (ButtonOn2 OR
CabButton2));
doOpen0 := y=0 AND (ButtonOn0 OR CabButton0);
doOpen1 := y=1 AND (ButtonOn1 OR CabButton1);
doOpen2 := y=2 AND (ButtonOn2 OR CabButton2);

IF state_move = STATE_STOPPED THEN
   liftUp := FALSE; liftDown:= FALSE;
   IF allClosed AND z < y AND z >= 0 THEN
   state_move:= STATE_GOING_DOWN;
   ELSIF allClosed AND z > y AND z >= 0 THEN
   state_move:= STATE_GOING_UP;
   ELSIF allClosed AND x < y AND x >= 0 THEN
      state_move:= STATE_GOING_DOWN;
   ELSIF allClosed AND x > y AND x >= 0 THEN
      state_move:= STATE_GOING_UP;
   END_IF;
ELSIF state_move = STATE_GOING_UP THEN
   liftUp:=TRUE; liftDown:= FALSE;
   IF doStop THEN
      state_move := STATE_STOPPED;
   END_IF;
ELSIF state_move = STATE_GOING_DOWN THEN
   liftDown:= TRUE; liftUp:= FALSE;
   IF doStop THEN
      state_move := STATE_STOPPED;
   END_IF;
END_IF;

IF state_door = STATE_CLOSED THEN
   IF doOpen0 THEN
      state_door := STATE_OPENING0;
   ELSIF doOpen1 THEN
      state_door := STATE_OPENING1;
   ELSIF doOpen2 THEN
      state_door := STATE_OPENING2;
   END_IF;
ELSIF state_door = STATE_OPENING0 THEN
   door0:= TRUE;
   IF DoorOpened0 THEN
      state_door := STATE_OPEN;
   END_IF;
ELSIF state_door = STATE_OPENING1 THEN
   door1:=TRUE;
   IF DoorOpened1 THEN
      state_door := STATE_OPEN;
   END_IF;
ELSIF state_door = STATE_OPENING2 THEN
   door2:=TRUE;
   IF DoorOpened2 THEN
      state_door := STATE_OPEN;
   END_IF;
ELSIF state_door = STATE_OPEN THEN
   t1_IN:=TRUE;
   t1_PT:=T#3S;
   Start_t1:=TRUE;
   IF t1_Q  THEN
      t1_IN:=FALSE;
      End_t1:=TRUE;
      t1_Q := FALSE;
      state_door := STATE_CLOSING;
   END_IF;
ELSIF state_door = STATE_CLOSING THEN
   door0:= FALSE;
   door1:= FALSE;
   door2:= FALSE;
   IF allClosed THEN
      state_door := STATE_CLOSED;
   END_IF;
END_IF;

END_ALGORITHM
```