2024

Simple embedded elevator

임베디드시스템 기말평가

발표일 2024.12.10

1조 20192531 장영민, 20203023 우상욱

CONTENTS

Simple embedded elevator

Simple embedded elevator?

STM32 nucleo board를 이용해 구현한 간단한 엘리베이터

설계 목표

01 / elevator 기본 동작

Elevator가 수직방향의 상승, 하강 동작을 수행한다

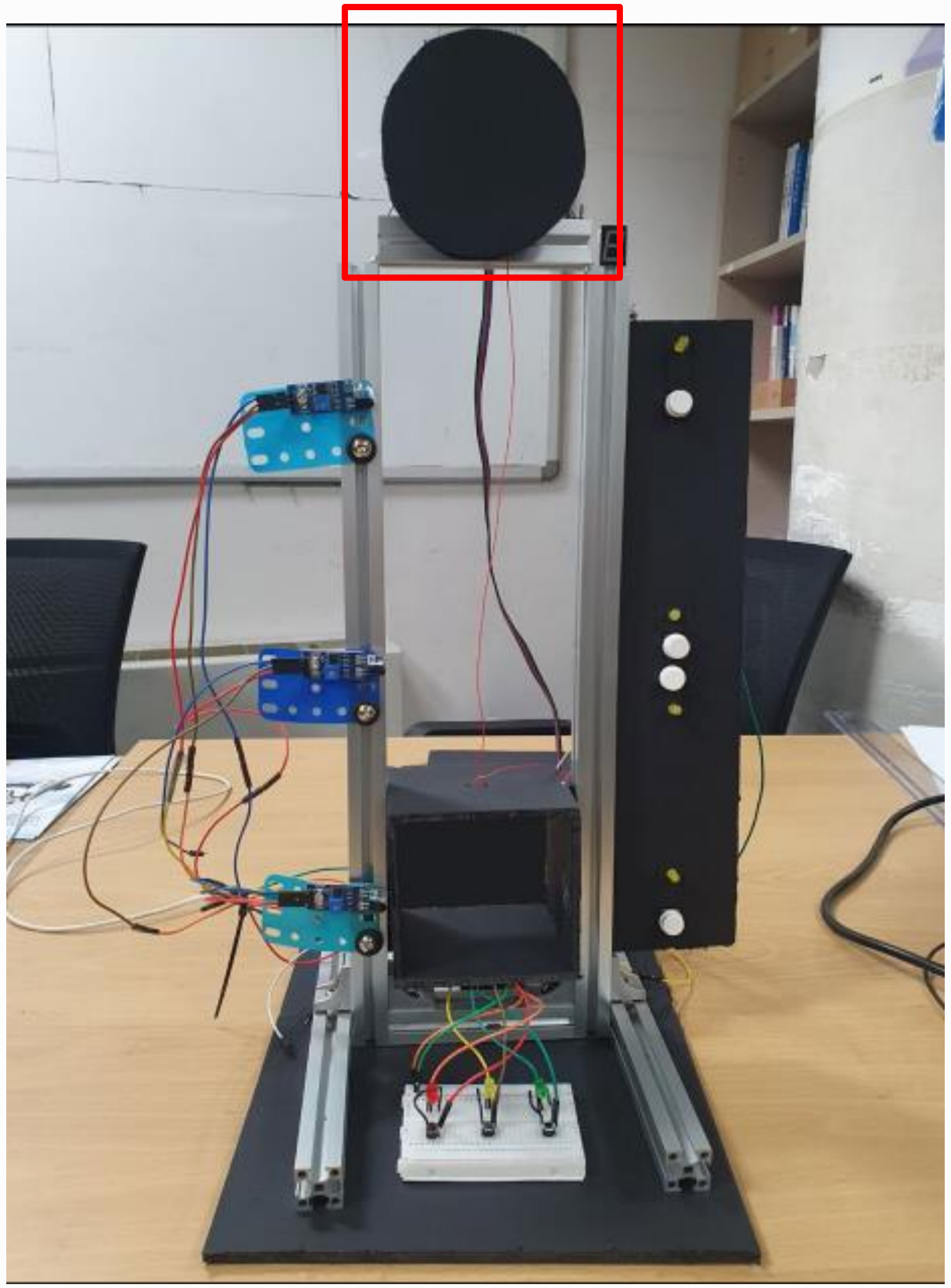
02 / Toggle switch

Switch를 **toggle**형식으로 설계해 실제 엘리베이터 버튼처럼 취소동작을 구현한다

03 / 우선순위가 있는 동작

Elevator가 **switch**에 의한 다양한 상황에서 우선순위를 가지고 동작할 수 있도록한다

Simple embedded elevator

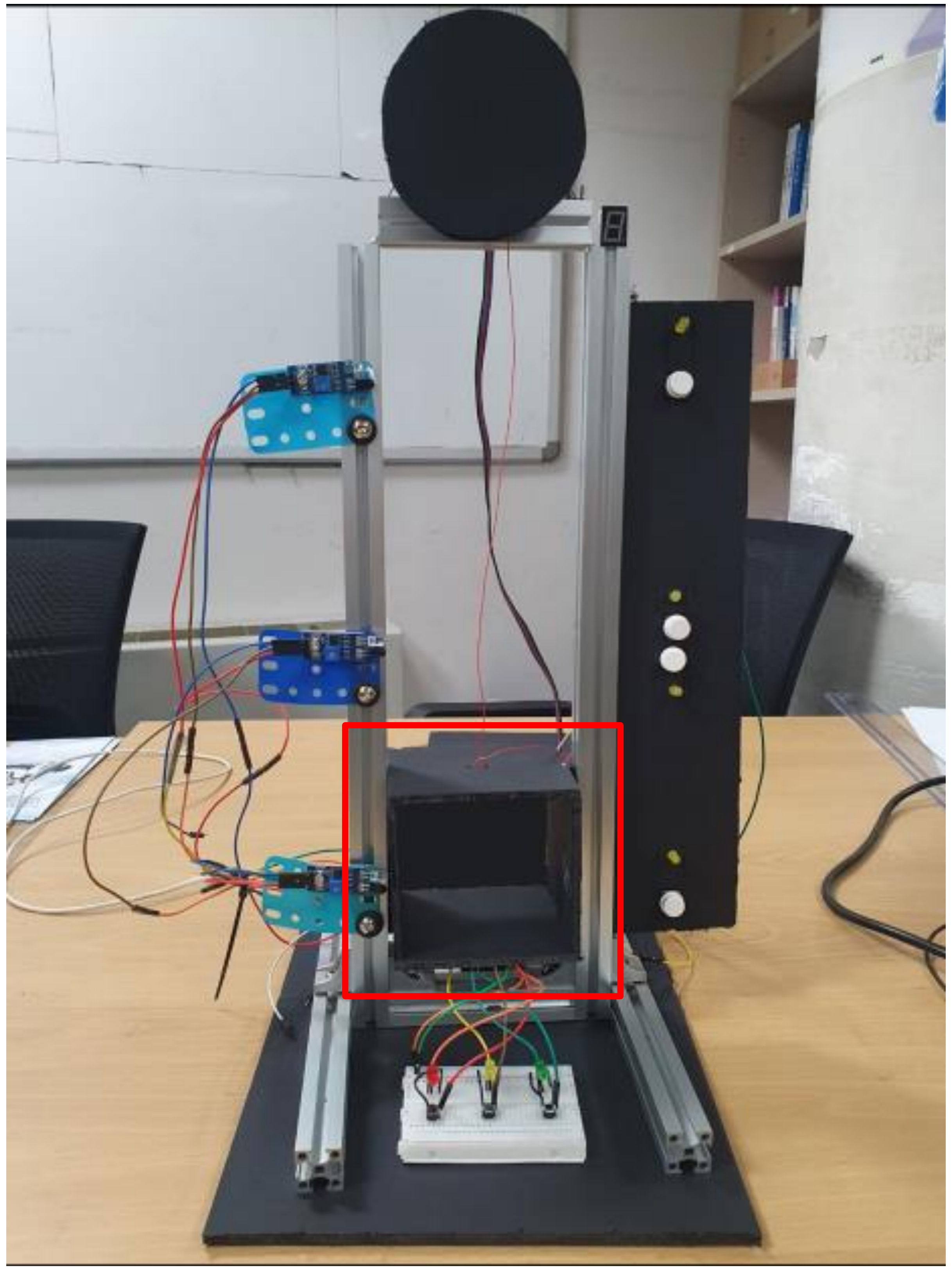


- **Step Motor**

한 **step**당 **0.8**도 돌아가는 **step motor**로 **step** 수를 계산하여 층 수를 구별한다

1

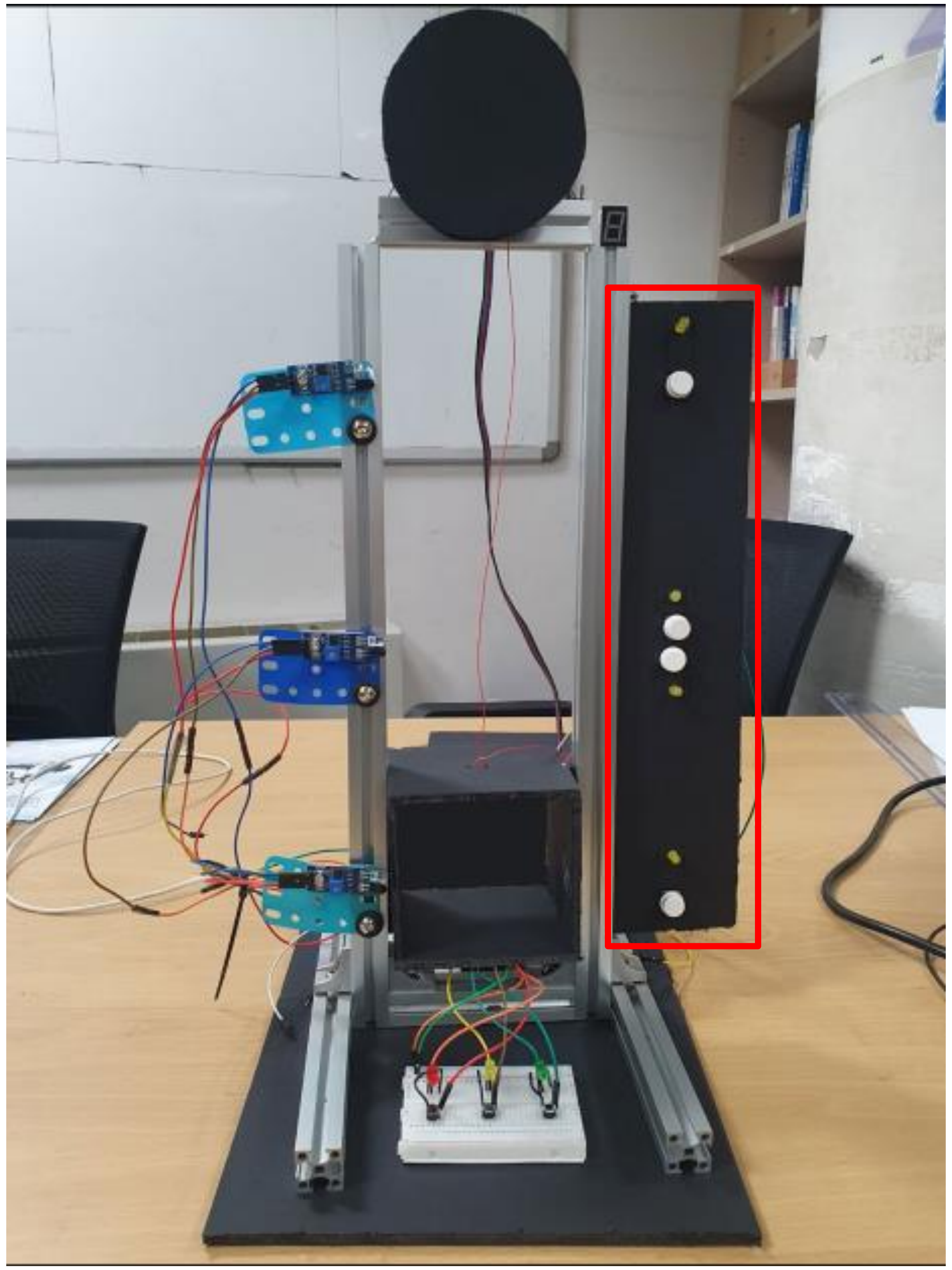
Simple embedded elevator



- **Elevator**

실제 엘리베이터 구조를 모사한 본체. **step motor**를 이용하여 수직 방향의 상승 및 하강 동작을 수행한다.

Simple embedded elevator

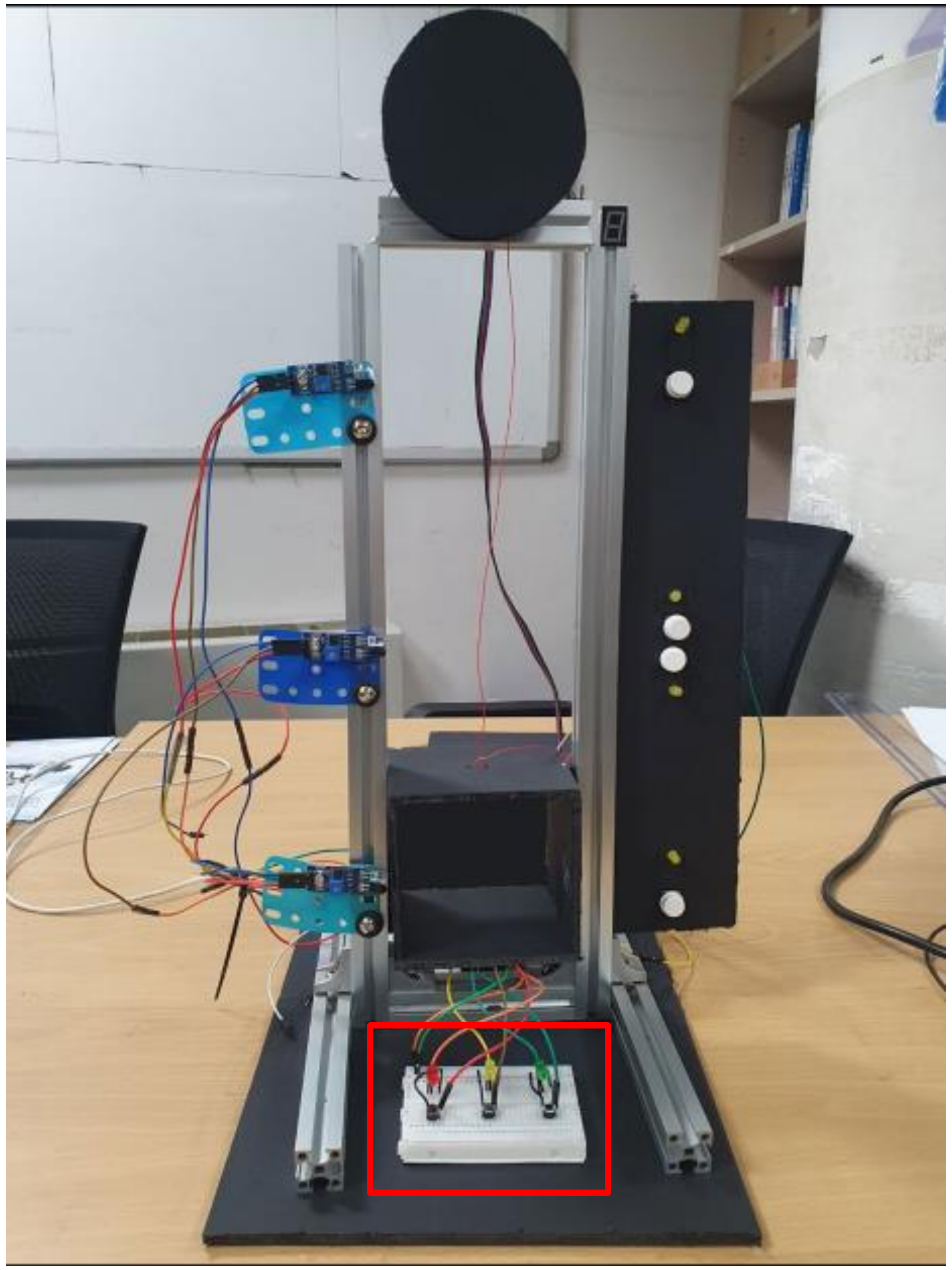


- **Elevator 외부 버튼**

Elevator 프레임 옆에 버튼을 **STM32Nucleo**에 연결하여 외부 버튼을 구현했다

1

Simple embedded elevator

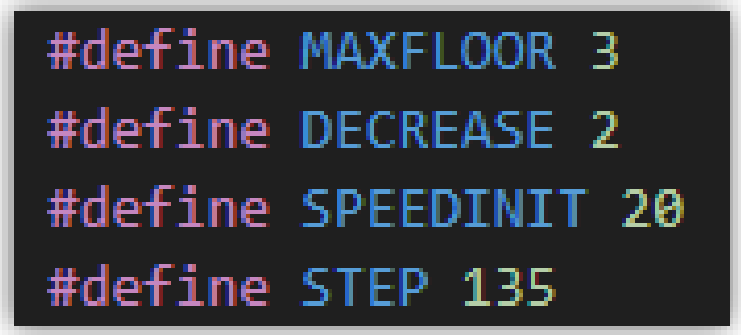


- **Elevator 내부 버튼**

Bread board를 통해 버튼을 **STM32Nucleo**에 연결하여 내부 버튼을 구현했다

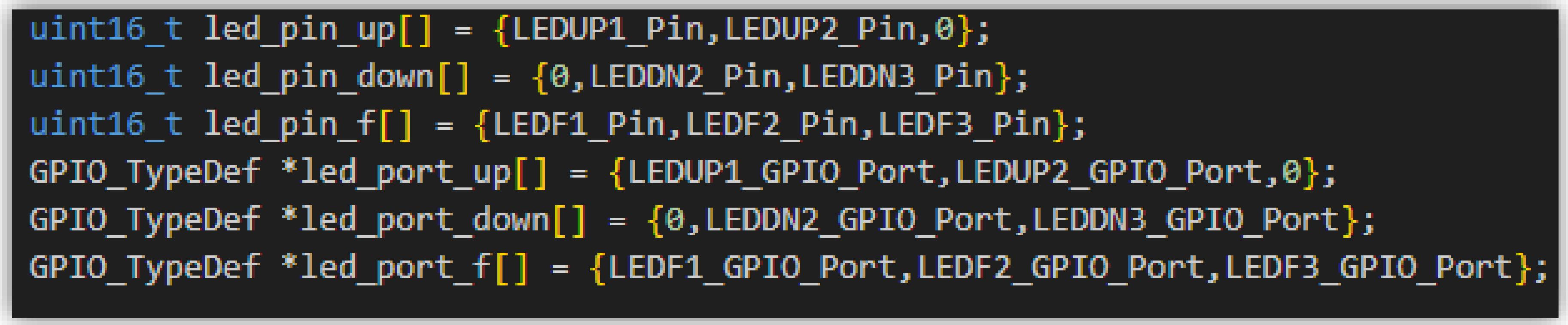
2

Code



```
#define MAXFLOOR 3
#define DECREASE 2
#define SPEEDINIT 20
#define STEP 135
```

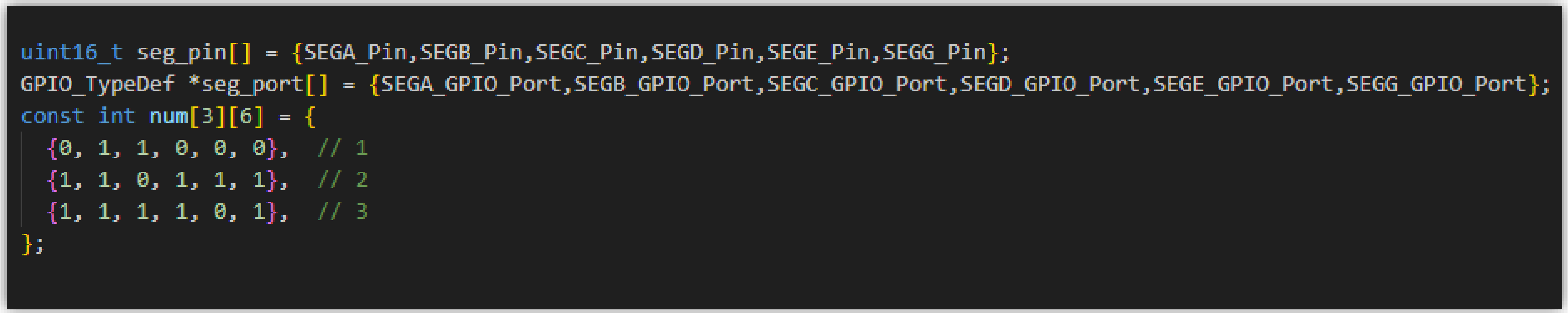
최대 층 및 모터 제어 상수 설정



```
uint16_t led_pin_up[] = {LEDUP1_Pin,LEDUP2_Pin,0};
uint16_t led_pin_down[] = {0,LEDDN2_Pin,LEDDN3_Pin};
uint16_t led_pin_f[] = {LEDF1_Pin,LEDF2_Pin,LEDF3_Pin};
GPIO_TypeDef *led_port_up[] = {LEDUP1_GPIO_Port,LEDUP2_GPIO_Port,0};
GPIO_TypeDef *led_port_down[] = {0,LEDDN2_GPIO_Port,LEDDN3_GPIO_Port};
GPIO_TypeDef *led_port_f[] = {LEDF1_GPIO_Port,LEDF2_GPIO_Port,LEDF3_GPIO_Port};
```

LED 제어를 위한 변수 선언

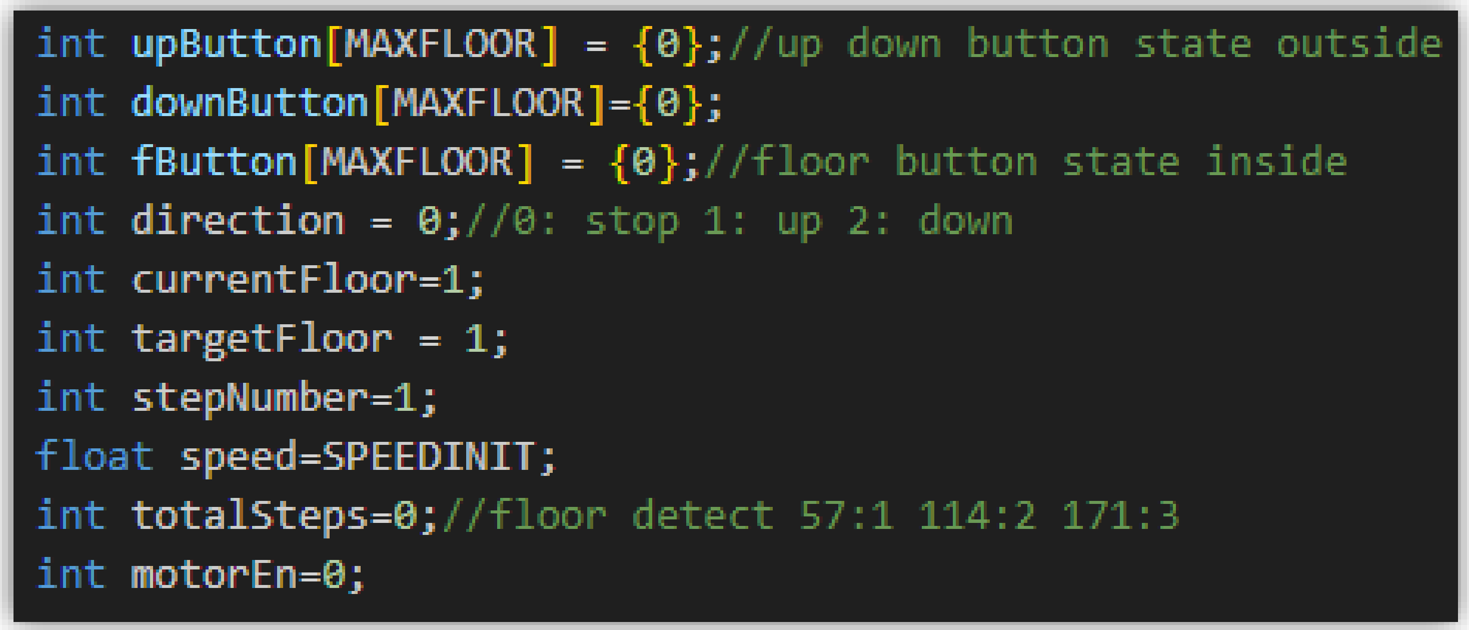
Code



```c
uint16_t seg_pin[] = {SEGA_Pin,SEGB_Pin,SEGC_Pin,SEGD_Pin,SEGE_Pin,SEGG_Pin};
GPIO_TypeDef *seg_port[] = {SEGA_GPIO_Port,SEGB_GPIO_Port,SEGC_GPIO_Port,SEGD_GPIO_Port,SEGE_GPIO_Port,SEGG_GPIO_Port};
const int num[3][6] = {
  {0, 1, 1, 0, 0, 0},  // 1
  {1, 1, 0, 1, 1, 1},  // 2
  {1, 1, 1, 1, 0, 1},  // 3
};
```

Segment 제어를 위한 변수 선언

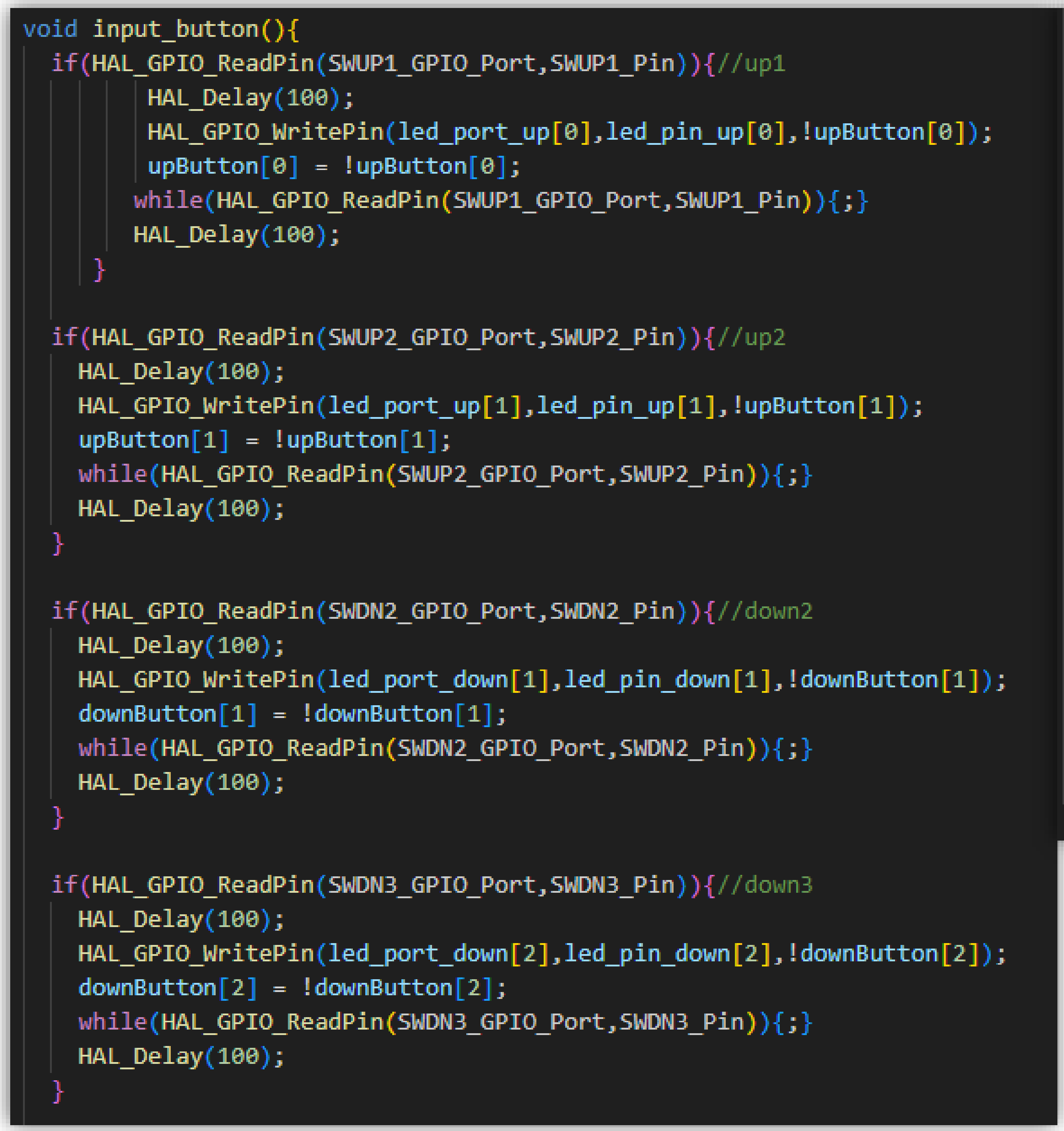
Code



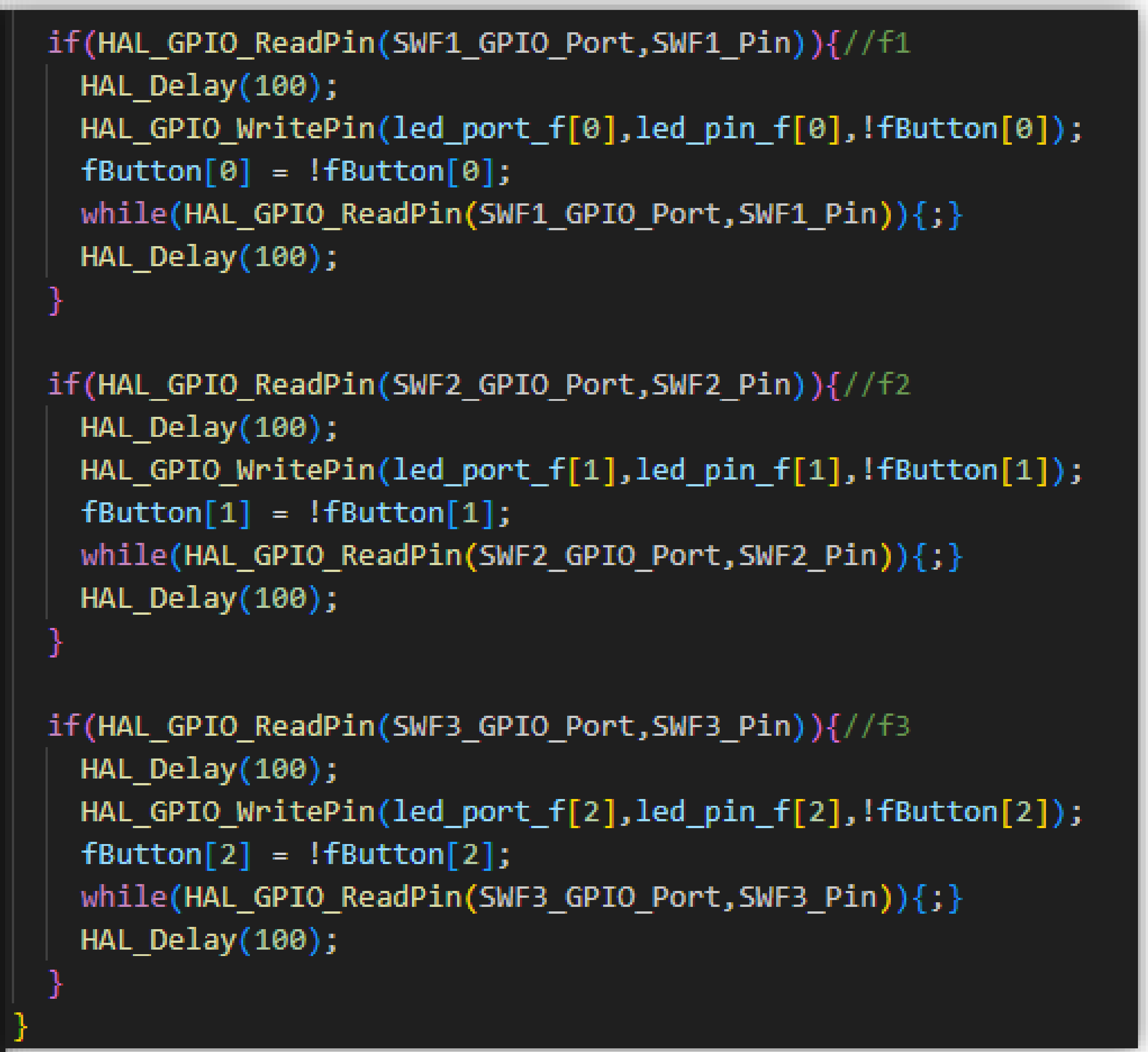
```
int upButton[MAXFLOOR] = {0};//up down button state outside
int downButton[MAXFLOOR]={0};
int fButton[MAXFLOOR] = {0};//floor button state inside
int direction = 0;//0: stop 1: up 2: down
int currentFloor=1;
int targetFloor = 1;
int stepNumber=1;
float speed=SPEEDINIT;
int totalSteps=0;//floor detect 57:1 114:2 171:3
int motorEn=0;
```

흐름 제어를 위한 변수 선언

Code



```c
void input_button(){
  if(HAL_GPIO_ReadPin(SWUP1_GPIO_Port,SWUP1_Pin)){//up1
        HAL_Delay(100);
        HAL_GPIO_WritePin(led_port_up[0],led_pin_up[0],!upButton[0]);
        upButton[0] = !upButton[0];
       while(HAL_GPIO_ReadPin(SWUP1_GPIO_Port,SWUP1_Pin)){;}
       HAL_Delay(100);
    }

  if(HAL_GPIO_ReadPin(SWUP2_GPIO_Port,SWUP2_Pin)){//up2
    HAL_Delay(100);
    HAL_GPIO_WritePin(led_port_up[1],led_pin_up[1],!upButton[1]);
    upButton[1] = !upButton[1];
    while(HAL_GPIO_ReadPin(SWUP2_GPIO_Port,SWUP2_Pin)){;}
    HAL_Delay(100);
  }

  if(HAL_GPIO_ReadPin(SWDN2_GPIO_Port,SWDN2_Pin)){//down2
    HAL_Delay(100);
    HAL_GPIO_WritePin(led_port_down[1],led_pin_down[1],!downButton[1]);
    downButton[1] = !downButton[1];
    while(HAL_GPIO_ReadPin(SWDN2_GPIO_Port,SWDN2_Pin)){;}
    HAL_Delay(100);
  }

  if(HAL_GPIO_ReadPin(SWDN3_GPIO_Port,SWDN3_Pin)){//down3
    HAL_Delay(100);
    HAL_GPIO_WritePin(led_port_down[2],led_pin_down[2],!downButton[2]);
    downButton[2] = !downButton[2];
    while(HAL_GPIO_ReadPin(SWDN3_GPIO_Port,SWDN3_Pin)){;}
    HAL_Delay(100);
  }
```

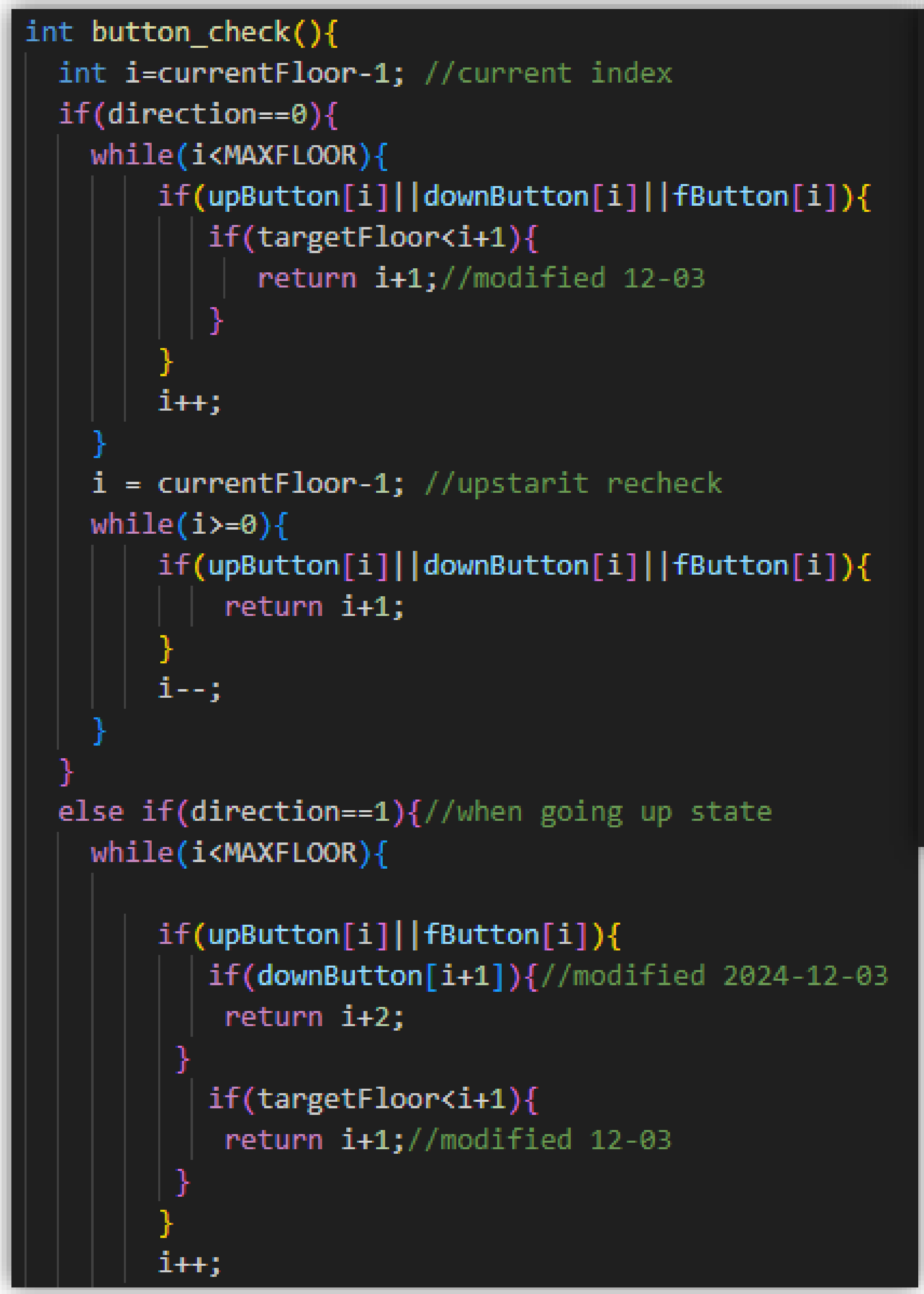


```c
  if(HAL_GPIO_ReadPin(SWF1_GPIO_Port,SWF1_Pin)){//f1
    HAL_Delay(100);
    HAL_GPIO_WritePin(led_port_f[0],led_pin_f[0],!fButton[0]);
    fButton[0] = !fButton[0];
    while(HAL_GPIO_ReadPin(SWF1_GPIO_Port,SWF1_Pin)){;}
    HAL_Delay(100);
  }

  if(HAL_GPIO_ReadPin(SWF2_GPIO_Port,SWF2_Pin)){//f2
    HAL_Delay(100);
    HAL_GPIO_WritePin(led_port_f[1],led_pin_f[1],!fButton[1]);
    fButton[1] = !fButton[1];
    while(HAL_GPIO_ReadPin(SWF2_GPIO_Port,SWF2_Pin)){;}
    HAL_Delay(100);
  }

  if(HAL_GPIO_ReadPin(SWF3_GPIO_Port,SWF3_Pin)){//f3
    HAL_Delay(100);
    HAL_GPIO_WritePin(led_port_f[2],led_pin_f[2],!fButton[2]);
    fButton[2] = !fButton[2];
    while(HAL_GPIO_ReadPin(SWF3_GPIO_Port,SWF3_Pin)){;}
    HAL_Delay(100);
  }
}
```

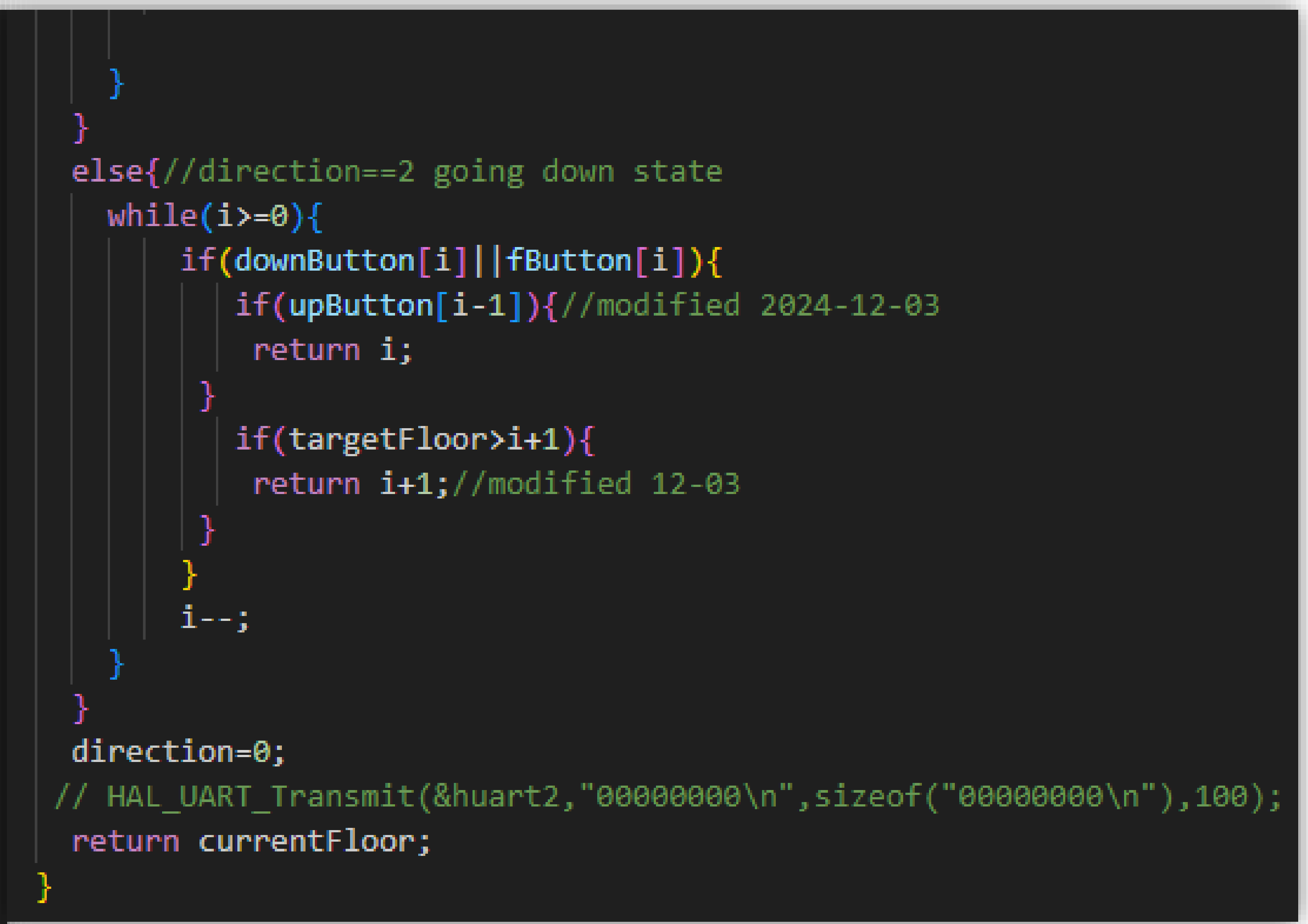
Switch 입력 시 LED를
toggle 하기 위한 함수

2

Code



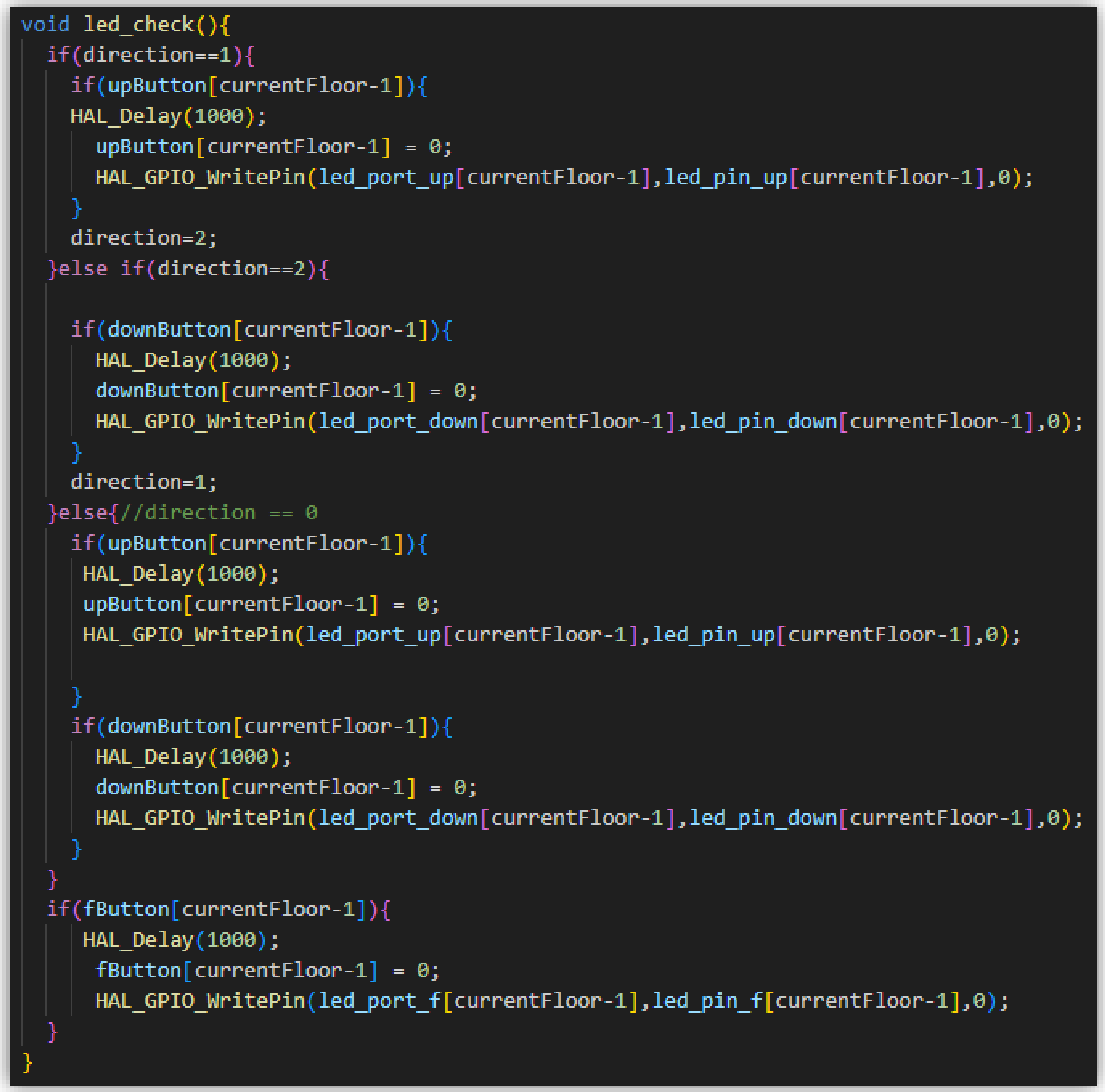
```
int button_check(){
  int i=currentFloor-1; //current index
  if(direction==0){
    while(i<MAXFLOOR){
        if(upButton[i]||downButton[i]||fButton[i]){
           if(targetFloor<i+1){
              return i+1;//modified 12-03
           }
        }
        i++;
    }
    i = currentFloor-1; //upstarit recheck
    while(i>=0){
        if(upButton[i]||downButton[i]||fButton[i]){
          return i+1;
        }
        i--;
    }
  }
  else if(direction==1){//when going up state
    while(i<MAXFLOOR){

        if(upButton[i]||fButton[i]){
           if(downButton[i+1]){//modified 2024-12-03
          return i+2;
         }
           if(targetFloor<i+1){
          return i+1;//modified 12-03
         }
        }
        i++;
```



```
      }
    }
    else{//direction==2 going down state
      while(i>=0){
          if(downButton[i]||fButton[i]){
            if(upButton[i-1]){//modified 2024-12-03
             return i;
            }
            if(targetFloor>i+1){
             return i+1;//modified 12-03
            }
          }
          i--;
      }
    }
    direction=0;
   // HAL_UART_Transmit(&huart2,"00000000\n",sizeof("00000000\n"),100);
    return currentFloor;
  }
```

몇 층에서 버튼이 눌렸는지
확인하고 우선순위에 맞게
targetFloor를 설정하는 함수

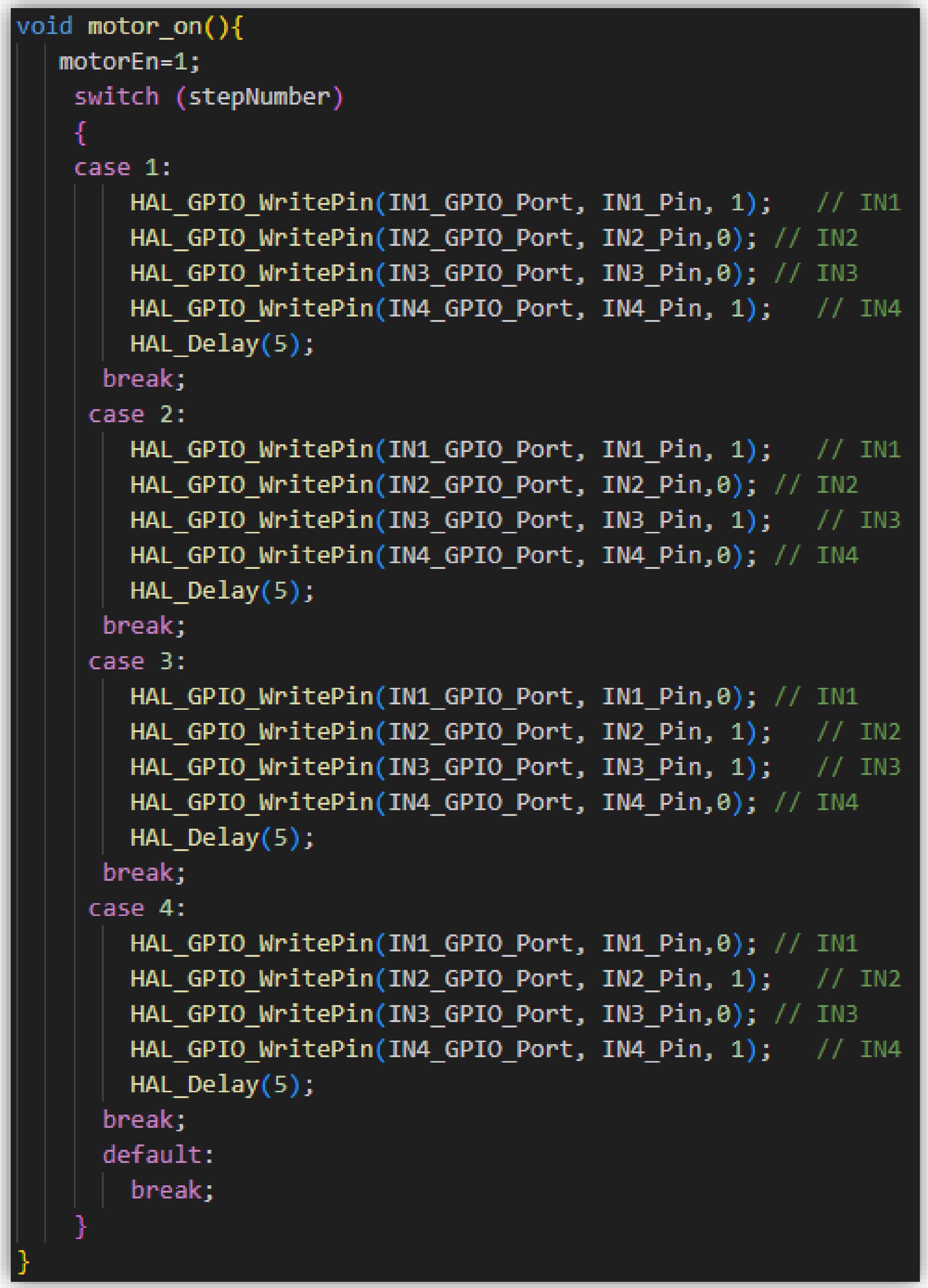
Code



```
void led_check(){
  if(direction==1){
    if(upButton[currentFloor-1]){
    HAL_Delay(1000);
      upButton[currentFloor-1] = 0;
      HAL_GPIO_WritePin(led_port_up[currentFloor-1],led_pin_up[currentFloor-1],0);
    }
    direction=2;
  }else if(direction==2){

    if(downButton[currentFloor-1]){
      HAL_Delay(1000);
      downButton[currentFloor-1] = 0;
      HAL_GPIO_WritePin(led_port_down[currentFloor-1],led_pin_down[currentFloor-1],0);
    }
    direction=1;
  }else{//direction == 0
    if(upButton[currentFloor-1]){
     HAL_Delay(1000);
     upButton[currentFloor-1] = 0;
     HAL_GPIO_WritePin(led_port_up[currentFloor-1],led_pin_up[currentFloor-1],0);

    }
    if(downButton[currentFloor-1]){
      HAL_Delay(1000);
      downButton[currentFloor-1] = 0;
      HAL_GPIO_WritePin(led_port_down[currentFloor-1],led_pin_down[currentFloor-1],0);
    }
  }
  if(fButton[currentFloor-1]){
   HAL_Delay(1000);
    fButton[currentFloor-1] = 0;
    HAL_GPIO_WritePin(led_port_f[currentFloor-1],led_pin_f[currentFloor-1],0);
  }
}
```

엘리베이터 현재 위치와 같은 층 버튼이
눌리면 **LED**를 **off** 시키는 함수

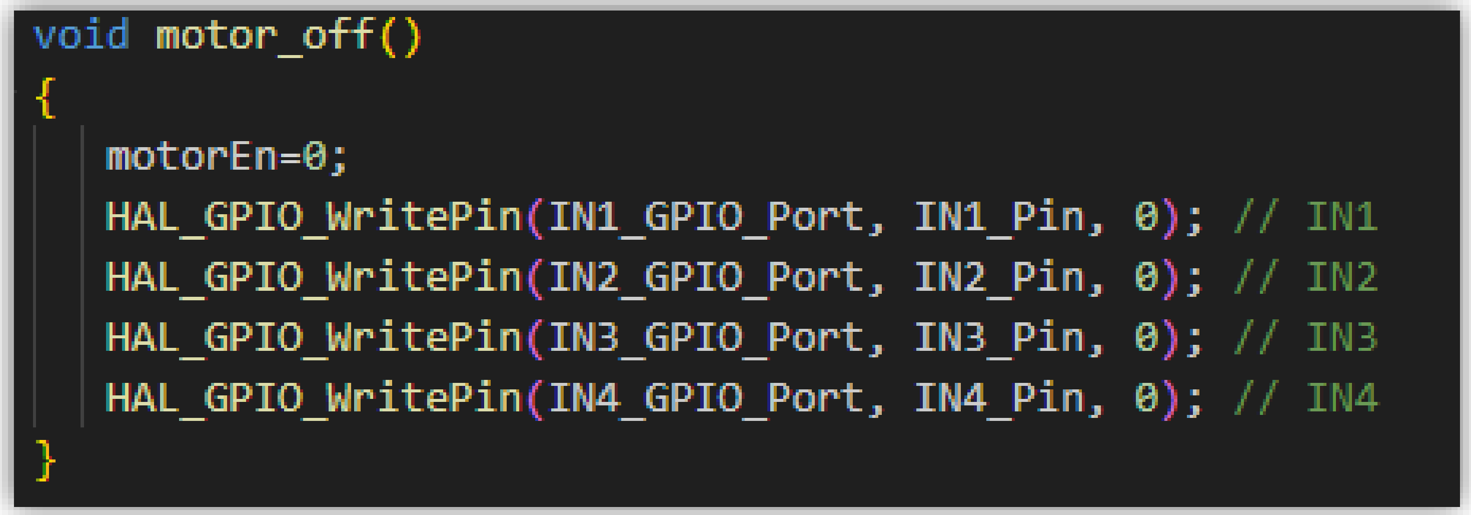
Code



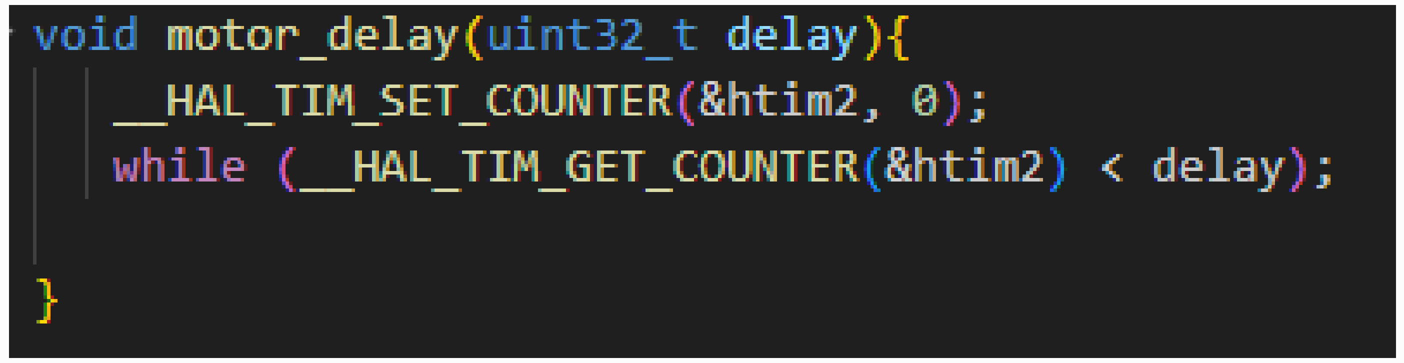
```
void motor_on(){
   motorEn=1;
    switch (stepNumber)
    {
    case 1:
        HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 1);   // IN1
        HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin,0); // IN2
        HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin,0); // IN3
        HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 1);   // IN4
        HAL_Delay(5);
      break;
     case 2:
        HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 1);   // IN1
        HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin,0); // IN2
        HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 1);   // IN3
        HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin,0); // IN4
        HAL_Delay(5);
      break;
     case 3:
        HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin,0); // IN1
        HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 1);   // IN2
        HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 1);   // IN3
        HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin,0); // IN4
        HAL_Delay(5);
      break;
     case 4:
        HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin,0); // IN1
        HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 1);   // IN2
        HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin,0); // IN3
        HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 1);   // IN4
        HAL_Delay(5);
      break;
      default:
        break;
    }
}
```

모터를 작동시키는 함수

Code



```
void motor_off()
{
    motorEn=0;
    HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 0); // IN1
    HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 0); // IN2
    HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 0); // IN3
    HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 0); // IN4
}
```

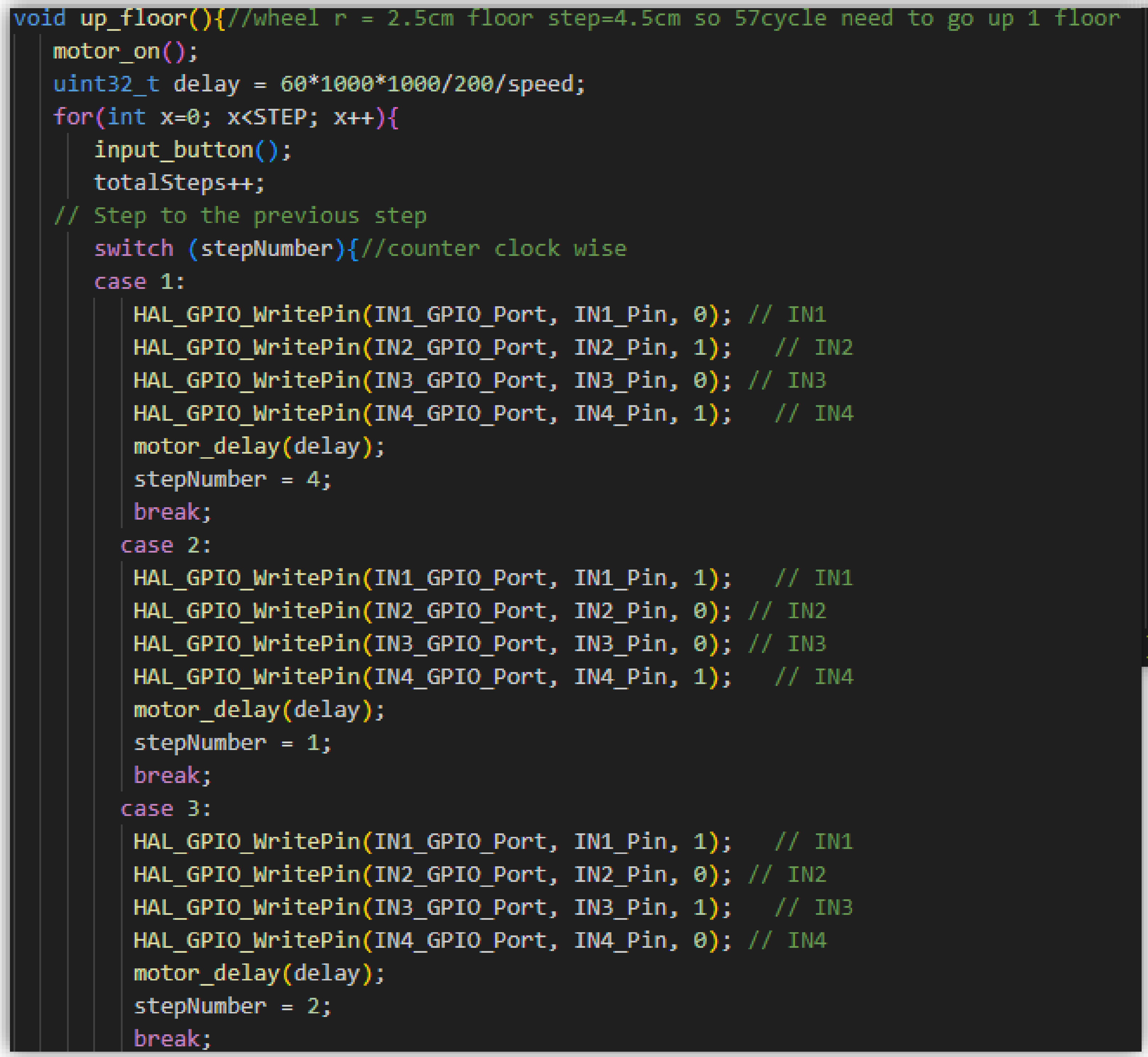


```
void motor_delay(uint32_t delay){
    __HAL_TIM_SET_COUNTER(&htim2, 0);
    while (__HAL_TIM_GET_COUNTER(&htim2) < delay);

}
```

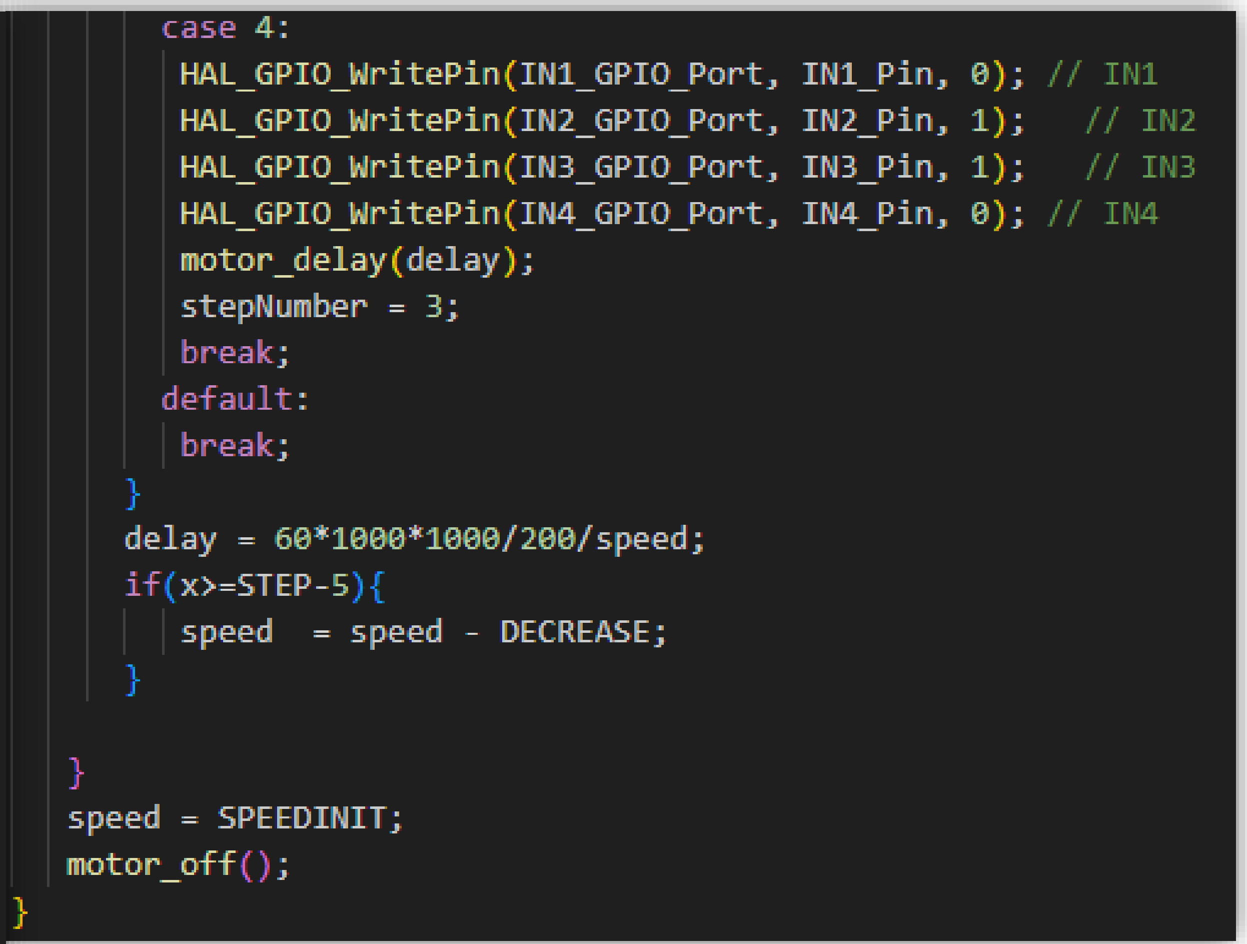
모터를 작동 중지시키는 함수

Step motor의 속도를 위한 **dalay** 함수

Code



```c
void up_floor(){//wheel r = 2.5cm floor step=4.5cm so 57cycle need to go up 1 floor
    motor_on();
    uint32_t delay = 60*1000*1000/200/speed;
    for(int x=0; x<STEP; x++){
        input_button();
        totalSteps++;
    // Step to the previous step
        switch (stepNumber){//counter clock wise
        case 1:
          HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 0); // IN1
          HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 1);   // IN2
          HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 0); // IN3
          HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 1);   // IN4
          motor_delay(delay);
          stepNumber = 4;
          break;
         case 2:
          HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 1);   // IN1
          HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 0); // IN2
          HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 0); // IN3
          HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 1);   // IN4
          motor_delay(delay);
          stepNumber = 1;
          break;
         case 3:
          HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 1);   // IN1
          HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 0); // IN2
          HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 1);   // IN3
          HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 0); // IN4
          motor_delay(delay);
          stepNumber = 2;
          break;
```

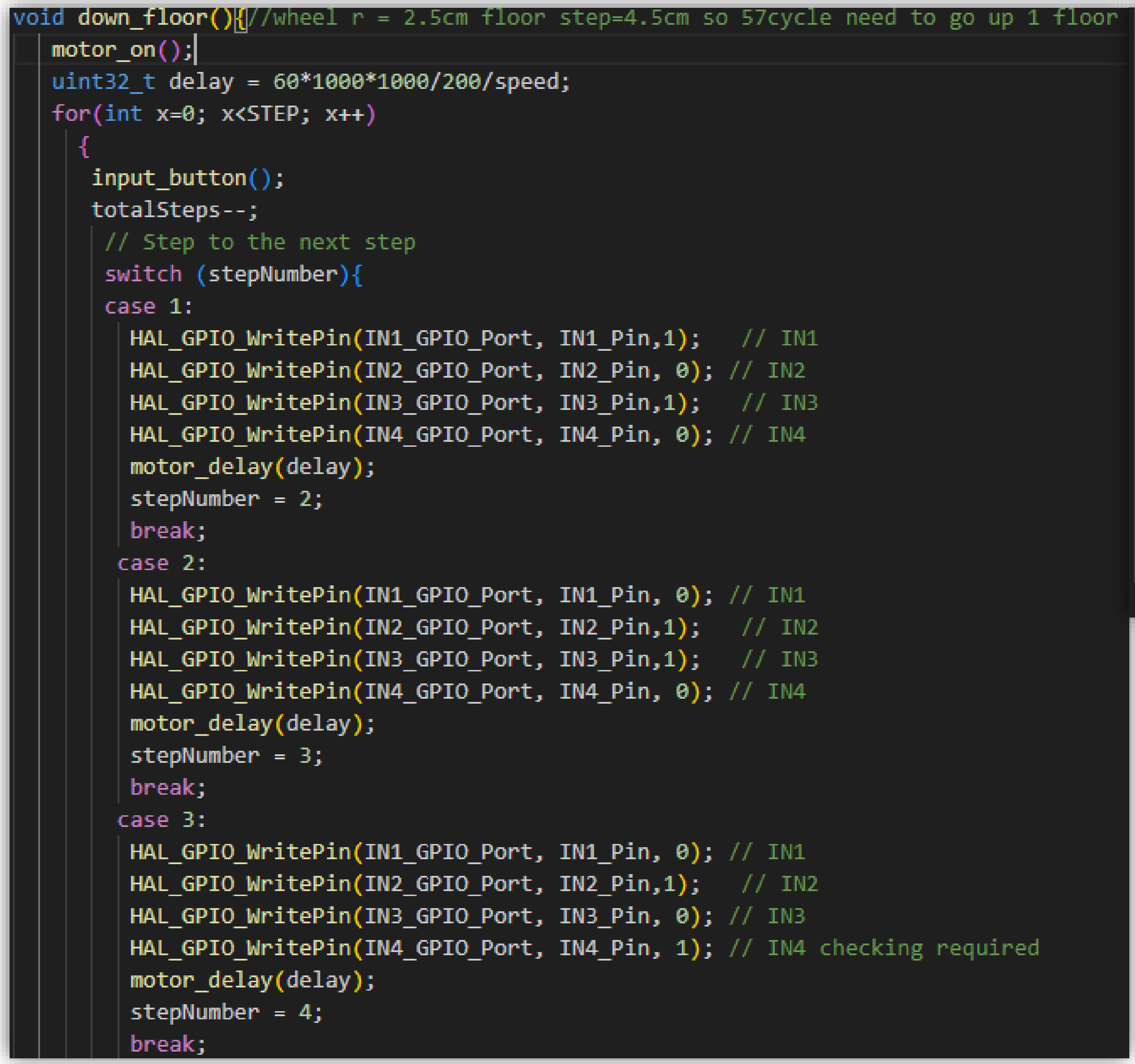


```c
         case 4:
          HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 0); // IN1
          HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 1);   // IN2
          HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 1);   // IN3
          HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 0); // IN4
          motor_delay(delay);
          stepNumber = 3;
          break;
         default:
          break;
       }
       delay = 60*1000*1000/200/speed;
       if(x>=STEP-5){
          speed  = speed - DECREASE;
       }

    }
    speed = SPEEDINIT;
    motor_off();
}
```

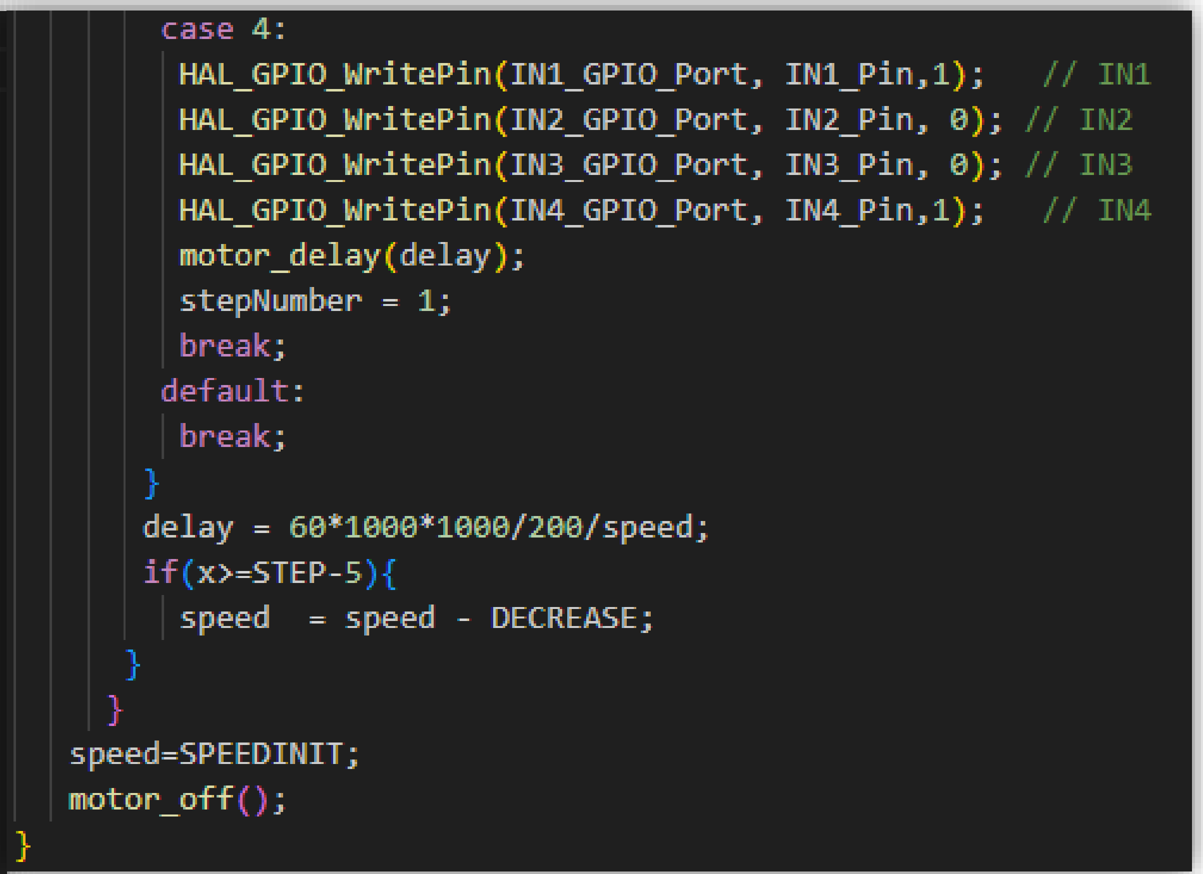
Elevator를 위로 보내기 위해 **step motor**를 정방향으로 회전시키는 함수

2

Code



```
void down_floor(){//wheel r = 2.5cm floor step=4.5cm so 57cycle need to go up 1 floor
    motor_on();
    uint32_t delay = 60*1000*1000/200/speed;
    for(int x=0; x<STEP; x++)
     {
       input_button();
       totalSteps--;
        // Step to the next step
        switch (stepNumber){
        case 1:
          HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin,1);   // IN1
          HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 0); // IN2
          HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin,1);   // IN3
          HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 0); // IN4
          motor_delay(delay);
          stepNumber = 2;
          break;
         case 2:
          HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 0); // IN1
          HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin,1);   // IN2
          HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin,1);   // IN3
          HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 0); // IN4
          motor_delay(delay);
          stepNumber = 3;
          break;
         case 3:
          HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin, 0); // IN1
          HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin,1);   // IN2
          HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 0); // IN3
          HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin, 1); // IN4 checking required
          motor_delay(delay);
          stepNumber = 4;
          break;
```

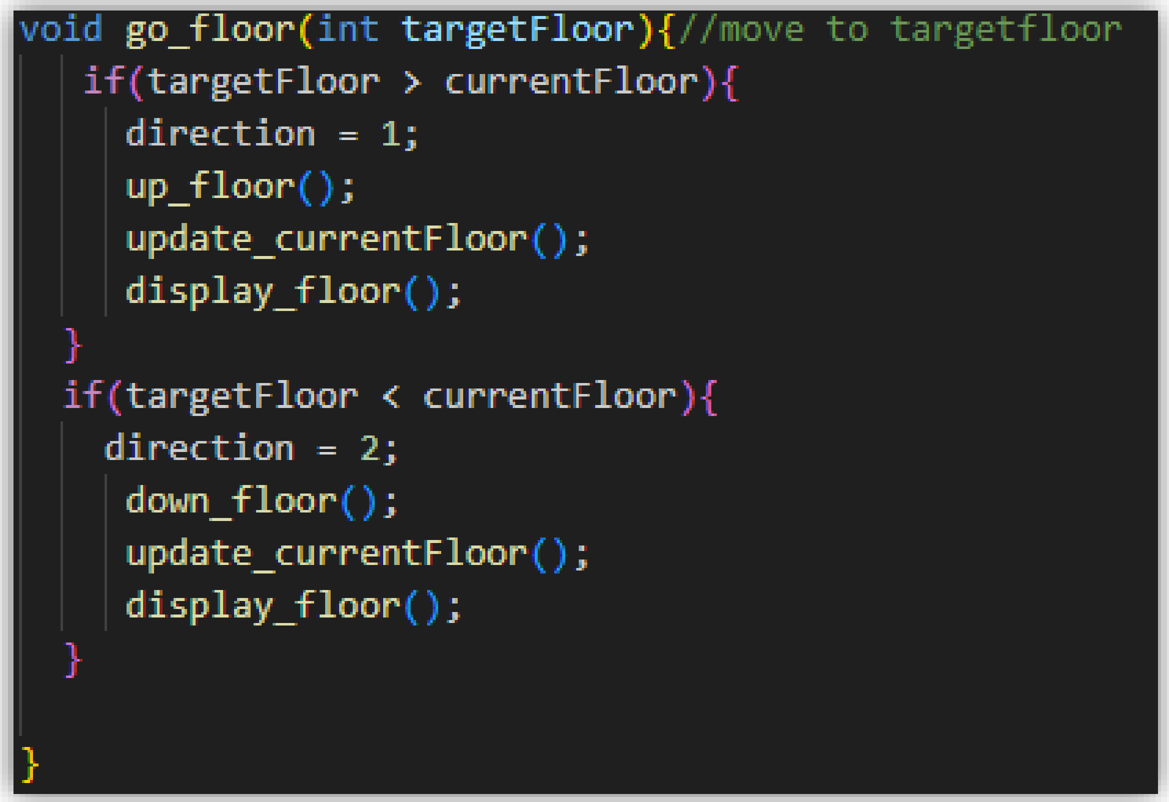


```
         case 4:
          HAL_GPIO_WritePin(IN1_GPIO_Port, IN1_Pin,1);   // IN1
          HAL_GPIO_WritePin(IN2_GPIO_Port, IN2_Pin, 0); // IN2
          HAL_GPIO_WritePin(IN3_GPIO_Port, IN3_Pin, 0); // IN3
          HAL_GPIO_WritePin(IN4_GPIO_Port, IN4_Pin,1);   // IN4
          motor_delay(delay);
          stepNumber = 1;
          break;
         default:
          break;
        }
        delay = 60*1000*1000/200/speed;
        if(x>=STEP-5){
          speed  = speed - DECREASE;
        }
      }
    speed=SPEEDINIT;
    motor_off();
}
```

Elevator를 아래로 보내기 위해 **step motor**를 역방향으로 회전시키는 함수

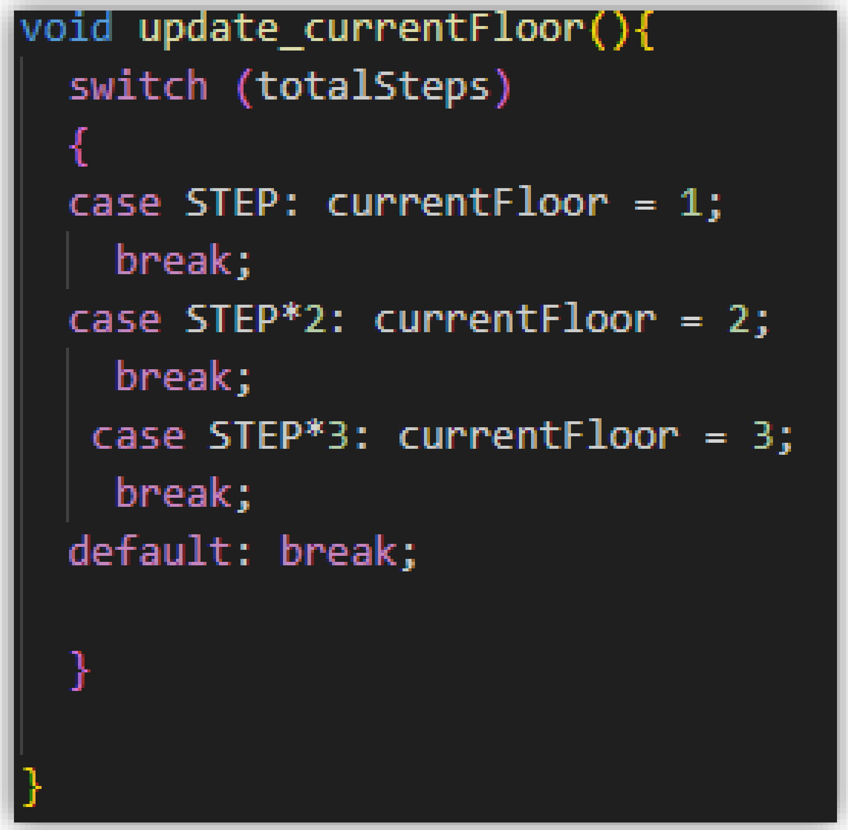
2

Code



```
void go_floor(int targetFloor){//move to targetfloor
    if(targetFloor > currentFloor){
      direction = 1;
      up_floor();
      update_currentFloor();
      display_floor();
   }
   if(targetFloor < currentFloor){
     direction = 2;
      down_floor();
      update_currentFloor();
      display_floor();
   }

}
```

Button_check 함수에서 설정된
targetFloor로 **elevator**를 이동시키는 함수

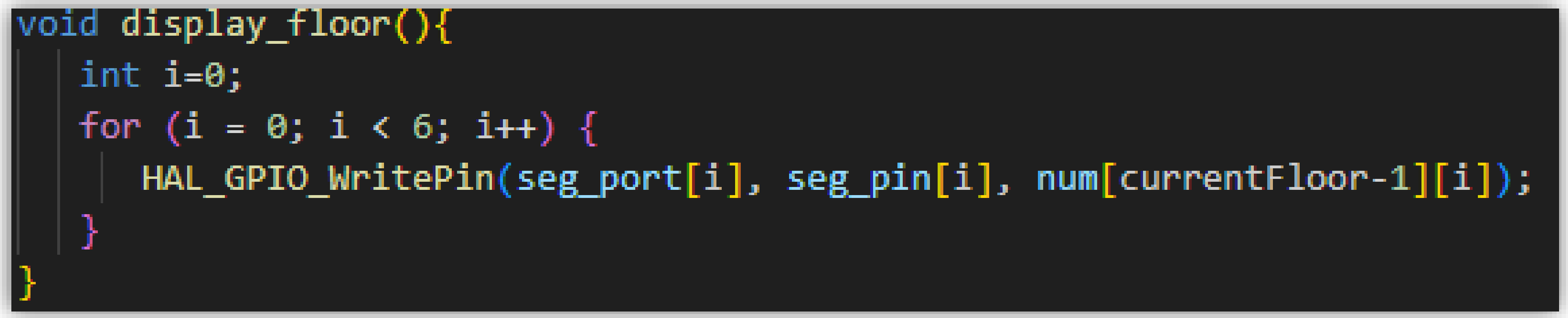


```
void update_currentFloor(){
  switch (totalSteps)
  {
  case STEP: currentFloor = 1;
    break;
  case STEP*2: currentFloor = 2;
    break;
   case STEP*3: currentFloor = 3;
    break;
  default: break;

  }

}
```

현재 층을 나타내는 변수를 **update**하는 함수

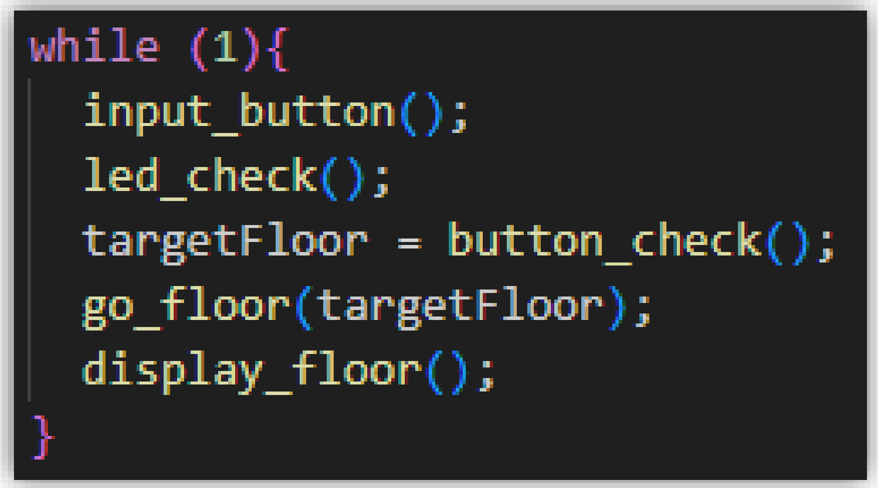
2

Code



```
void display_floor(){
    int i=0;
    for (i = 0; i < 6; i++) {
        HAL_GPIO_WritePin(seg_port[i], seg_pin[i], num[currentFloor-1][i]);
    }
}
```

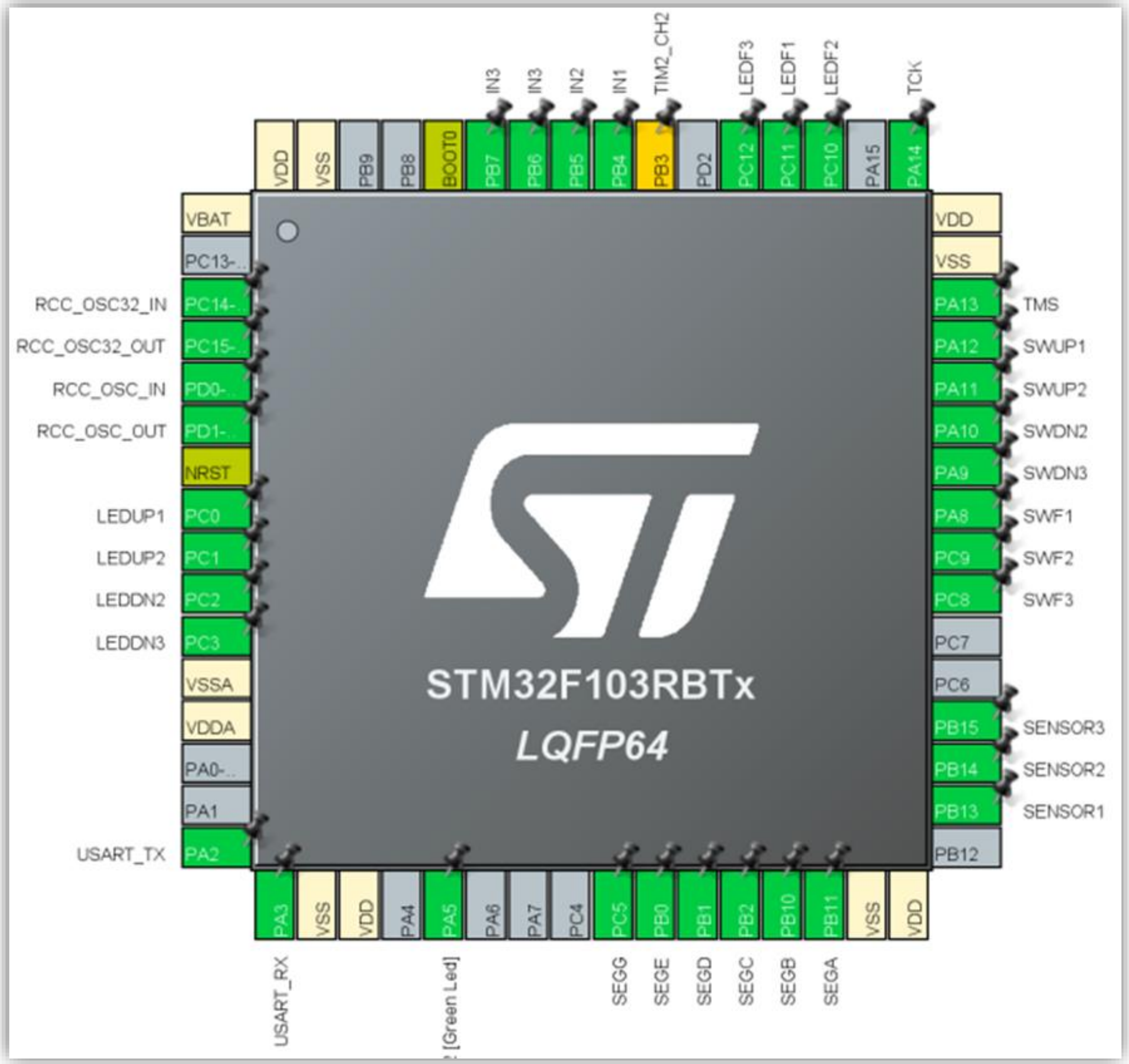
현재 **elevator**가 있는 층을 **7-segment**로 출력하기 위한 함수



```
while (1){
  input_button();
  led_check();
  targetFloor = button_check();
  go_floor(targetFloor);
  display_floor();
}
```

Main 함수

Pin Map


IN3
IN3
IN2
IN1
TIM2_CH2
LEDF3
LEDF1
LEDF2
TCK
VDD
VSS
PB9
PB8
BOOT0
PB7
PB6
PB5
PB4
PB3
PD2
PC12
PC11
PC10
PA15
PA14
VBAT
PC13-..
RCC_OSC32_IN
PC14-.
RCC_OSC32_OUT
PC15-.
RCC_OSC_IN
PD0-
RCC_OSC_OUT
PD1-
NRST
LEDUP1
PC0
LEDUP2
PC1
LEDDN2
PC2
LEDDN3
PC3
VSSA
VDDA
PA0-..
PA1
USART_TX
PA2
VDD
VSS
PA13
TMS
PA12
SWUP1
PA11
SWUP2
PA10
SWDN2
PA9
SWDN3
PA8
SWF1
PC9
SWF2
PC8
SWF3
PC7
PC6
PB15
SENSOR3
PB14
SENSOR2
PB13
SENSOR1
PB12
STM32F103RBTx
LQFP64
PA3
VSS
VDD
PA4
PA5
PA6
PA7
PC4
PC5
PB0
PB1
PB2
PB10
PB11
VSS
VDD
USART_RX
[Green Led]
SEGG
SEGE
SEGD
SEGC
SEGB
SEGA

4

작품 시연

THANK YOU