
2021 한국폴리텍대학 (다기능과정)
IT융합전자회로 설계 및 제작 경진대회

제 3 과 제

Project Name : MICOM PROGRAMMING

제한 시간 : 5시간



후원 : 학교법인 한국폴리텍대학

협찬 : 한국폴리텍대학 창원캠퍼스,나인플러스아이티(주)

IT융합전자회로 설계 및 제작 경진대회 과제

과 제 명	Micom Programming	경기시간	5시간
비번호		감독위원확인	(인)

1. 요구사항

가. 지급된 프로그램 설계 회로기관, 회로도, 동작 설명 사항을 참조하여 제작하시오.
(단, DC MOTOR는 모렉스와 케이블을 이용하여 기관 밖으로 배치하시오.)

나. 배포된 소스는 프로그램 요구조건 가) ~ 바)까지 동작되게 한 것이다. 이 소스를 컴파일, 라이팅하여 프로그램 요구조건 가) ~ 바)까지 동작 시키시오.

※ 단, 자신의 컴파일러와 맞지 않는 부분은 수정하여 동작시키시오.

다. 배포된 소스를 참조하여 프로그램 요구조건 문제1]~문제3]을 프로그램하고, 컴파일, 라이팅하여 동작시킨 후 자신의 학번을 적고, 저장하시오.

C:\ ElevatorXXX.C (XXX : 자신의 학번)

라. 이 작품은 엘리베이터를 축소하여 구현한 것으로 다음과 같은 동작을 한다.

- (1) 엘리베이터 내부 / 외부 구분
- (2) 층수 표시 기능
- (3) DOOR에 물체 감지 기능
- (4) 무게 초과 확인 기능
- (5) 층별 안내 기능

마. 동작 요구사항

※ 유의사항

(가) PCB 기관은 엘리베이터 내부와 외부로 구분되어 있음을 유의하시오.

(나) FND1, FND2는 같은 숫자를 표시한다.

(다) MOTOR의 정회전은 엘리베이터의 ‘문 열림’이고 역회전은 ‘문 닫힘’이다.

(라) MOTOR의 정회전은 MOTOR의 축을 중심으로 시계방향, 역회전은 반시계 방향이다.

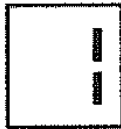
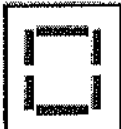

(마) ‘외부 층’ SW는 ‘UP’ SW와 ‘DOWN’ SW로 누르는 층을 설정하는 스위치이다.

(바) ‘외부 층’ SW를 누른 후 ‘UP’ SW와 ‘DOWN’ SW를 누르면 그 층에 맞는 LED가 점등한다.

(1) 다음 상황에서 발생하는 소리를 주파수를 참고하여 프로그램 하시오.

경보음		‘문 열림 / 닫힘’ 음	
삐~	493.88[Hz]	딩	440.0[Hz]
		동	349.23[Hz]

(2) 전원을 인가하거나 RESET SW를 누르면 다음과 같이 동작되게 하시오.

시간	LCD	FND	MOTOR																														
0초 후	<table><tr><td>E</td><td>l</td><td>e</td><td>v</td><td>a</td><td>t</td><td>o</td><td>r</td><td></td><td>S</td><td>y</td><td>s</td><td>t</td><td>e</td><td>m</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>L</td><td>O</td><td>A</td><td>D</td><td>I</td><td>N</td><td>G</td><td></td></tr></table>	E	l	e	v	a	t	o	r		S	y	s	t	e	m								L	O	A	D	I	N	G			정회전
E	l	e	v	a	t	o	r		S	y	s	t	e	m																			
							L	O	A	D	I	N	G																				
1초 후	<table><tr><td>E</td><td>l</td><td>e</td><td>v</td><td>a</td><td>t</td><td>o</td><td>r</td><td></td><td>S</td><td>y</td><td>s</td><td>t</td><td>e</td><td>m</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>L</td><td>O</td><td>A</td><td>D</td><td>I</td><td>N</td><td>G</td><td>!</td></tr></table>	E	l	e	v	a	t	o	r		S	y	s	t	e	m								L	O	A	D	I	N	G	!		정지
E	l	e	v	a	t	o	r		S	y	s	t	e	m																			
							L	O	A	D	I	N	G	!																			
2초 후	<table><tr><td>E</td><td>l</td><td>e</td><td>v</td><td>a</td><td>t</td><td>o</td><td>r</td><td></td><td>S</td><td>y</td><td>s</td><td>t</td><td>e</td><td>m</td></tr><tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>L</td><td>O</td><td>A</td><td>D</td><td>I</td><td>N</td><td>G</td><td>!!</td></tr></table>	E	l	e	v	a	t	o	r		S	y	s	t	e	m								L	O	A	D	I	N	G	!!		역회전
E	l	e	v	a	t	o	r		S	y	s	t	e	m																			
							L	O	A	D	I	N	G	!!																			

(3) 다음의 엘리베이터의 기본 동작 요구사항이 동작되게 프로그램 하시오.

(가) 엘리베이터가 1층을 이동하는데 소요되는 시간은 2초가 되게 하시오.

(나) 엘리베이터의 현재 층수를 표시하는 FND1, FND2는 같은 숫자가 표시되게 하시오.

(다) “요구사항 (1)”에서 ‘문닫힘, 문열림’ 소리가 상황에 맞게 출력되게 하시오.

(라) 엘리베이터의 내부 SW(‘내부 1층’~‘내부 5층’)를 눌러 원하는 층에 도착하면 1초 후에 MOTOR가 3초간 정회전하고, 3초간 정지 후 역회전을 3초간 하게 하시오.

(마) 엘리베이터의 문이 열린 상태(MOTOR의 3초간 정회전)에서 ‘열림’ SW를 누르면 MOTOR가 정지되게 하고 ‘열림’ SW에서 손을 떼면 3초 후에 문 닫힘(역회전) 되게 하시오.

단, 닫히는 중에 ‘열림’ SW를 누를 시 MOTOR가 1~2초간 문 닫힘(역회전)을 하였다면, 다시 문 열림(정회전)을 1~2초하게 하시오.

(바) 문이 완전히 열린 후 ‘닫힘’ SW를 누르면 즉시 MOTOR가 3초간 역회전하게 하시오.

(사) 문이 완전히 닫히지 않으면 (MOTOR 3초간 역회전) 엘리베이터의 이동이 되지 않게 하시오.

- (아) 문 열림과 문 닫힘 후 층수 표시 LED는 OFF 하시오.
 (자) 문 열림과 문 닫힘 시 LCD 두 번째줄에 다음과 표시되게 하시오.

1	F	:													
W	:		4	7	k	g			C			2		4	5

(문 닫힘인 경우)

1	F	:													
W	:		4	7	k	g			O			2		4	5

(문 열림인 경우)

- (4) 엘리베이터 ‘외부 2층’ SW를 누른 후 ‘UP’ SW 또는 ‘DOWN’ SW를 누르면 다음과 같은 동작이 되게 하시오.
- (가) 엘리베이터가 2층으로 이동되게 하시오.
 - (나) 2층 도착 후 1초 후에 MOTOR는 3초간 정회전을 하게 하시오.
 - (다) MOTOR의 정회전 3초 후에 3초간 정지한 다음 3초간 역회전하게 하시오.
 - (라) 역회전 도중에 포토인터럽트에 물체가 감지되면 문이 다시 열리게 하고 (정회전), 물체가 감지되지 않으면 3초 후에 다시 문이 닫히게(역회전) 하시오.
- (예) 문 닫힘(역회전)을 1초간 하였을 때 포토인터럽트에 물체가 감지되면 1초간 문 열림(정회전)하게 하고, 정지하였다가 물체가 감지되지 않으면 3초 후에 문 닫힘(역회전)을 한다.
- (5) “요구사항 (4)” 실행 후 엘리베이터 내부의 ‘내부 1층’~‘내부 5층’중 ‘내부 5층’ SW를 누른 후 엘리베이터 ‘외부 1층’ SW를 누르고 ‘UP’ SW를 누르면 다음과 같이 동작되게 하시오.
- (가) 엘리베이터가 5층에 도착(FND1 = FND2 = ‘5’)하면 “요구사항 (3)”과 같이 되게 하시오.
 - (나) 엘리베이터 외부 1층의 ‘UP’ SW를 눌렀으므로, 5층에서 문이 닫힌 후 1층으로 이동하게 하시오.
 - (다) 1층 도착 후 1초 후에 문 열림(정회전)을 하고 3초 정지 후 문 닫힘(역회전)을 하게 하시오.
- (6) 현재 엘리베이터가 위치하고 있는 층을 LCD 첫 번째 줄에 내부에서 ‘내부 1층’~‘내부 5층’ 스위치를 누를 경우 선택된 층을 LCD 두 번째 줄에 다음과 같이 표시하시오.

(예) 내부 2층, 내부 4층, 내부 5층 버튼을 눌렀을 경우

1	F	:													
W	:		4	7	k	g			O			2		4	5

2	F	:													
W	:		4	7	k	g			O					4	5

• •

5	F	:													
W	:		4	7	k	g			O						

[문제 1] 무게 표시

(7) 가변저항(VR1)으로 무게를 입력받는다. 무게가 99kg을 초과할 시 99kg이하로 무게가 내려갈 때까지 “요구사항 (1)”에서 설정한 경보음이 1초 간격으로 울리고, 모든 스위치가 HOLD 되고, 문은 열린 상태를 유지, LCD 두 번째 줄에는 다음과 같이 ‘OVER WEIGHT!!!’가 출력된다.

1	F	:													
W	:		4	7	k	g			O						

(평상시)

1	F	:												
	O	V	E	R			W	E	I	G	H	T	!	!

(무게 99kg 초과시)

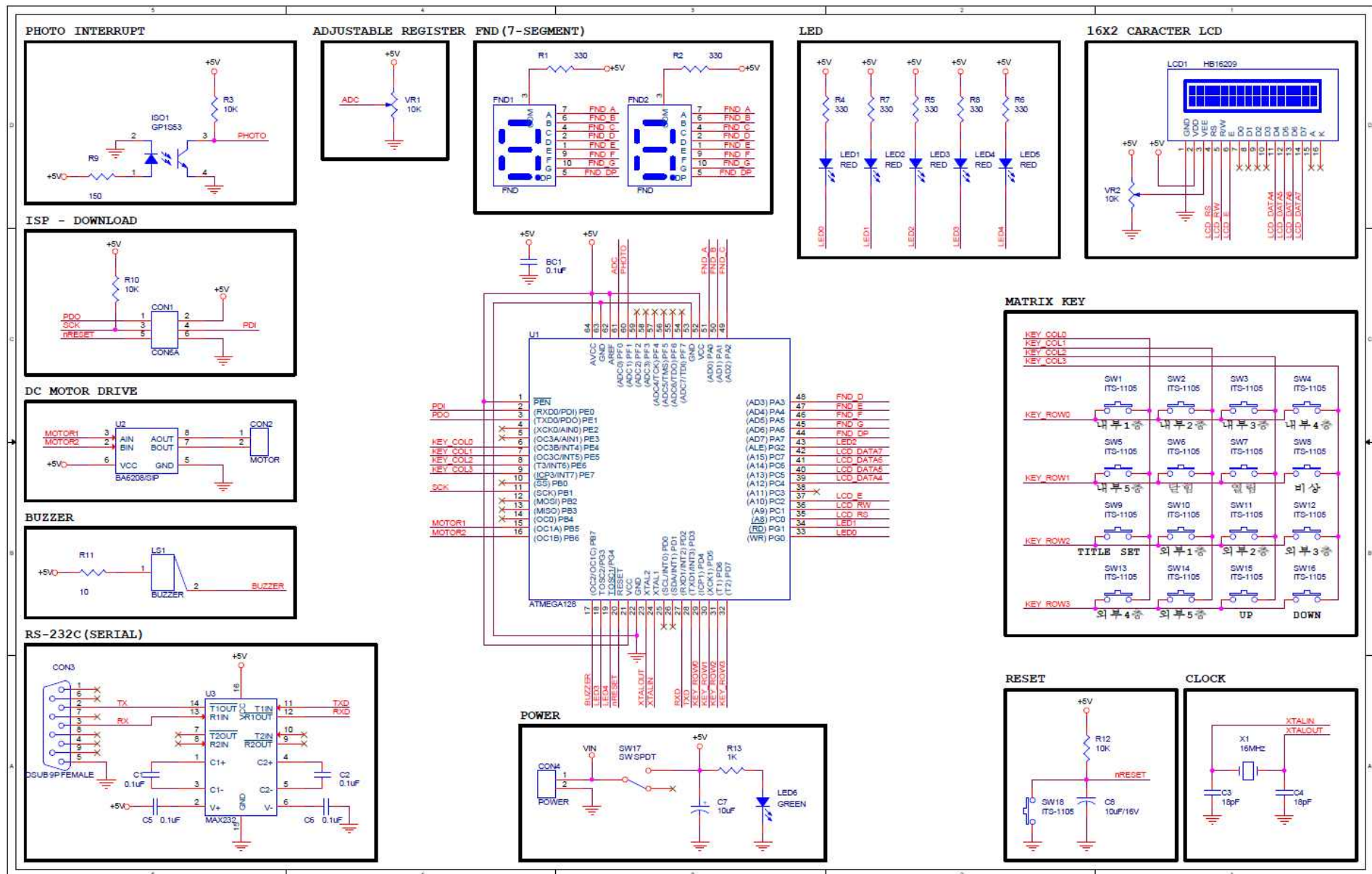
(단, 가변저항의 최소값을 0kg, 최대값을 130kg으로 정의한다.)

(단, 완전히 문이 열린 상태에서만 무게를 입력받는다.)

[문제 2] 비상스위치 동작

(8) 엘리베이터가 이동하는 중에 ‘비상’ SW를 누르면 PC에서 ‘ESC’ 키를 누르기 전 까지 모든 스위치가 HOLD 되고 “요구사항 (1)”에서 설정한 경보음이 1초 간격으로 울리고 LCD 두 번째 줄에 다음과 같이 출력한다.

3. 회로도



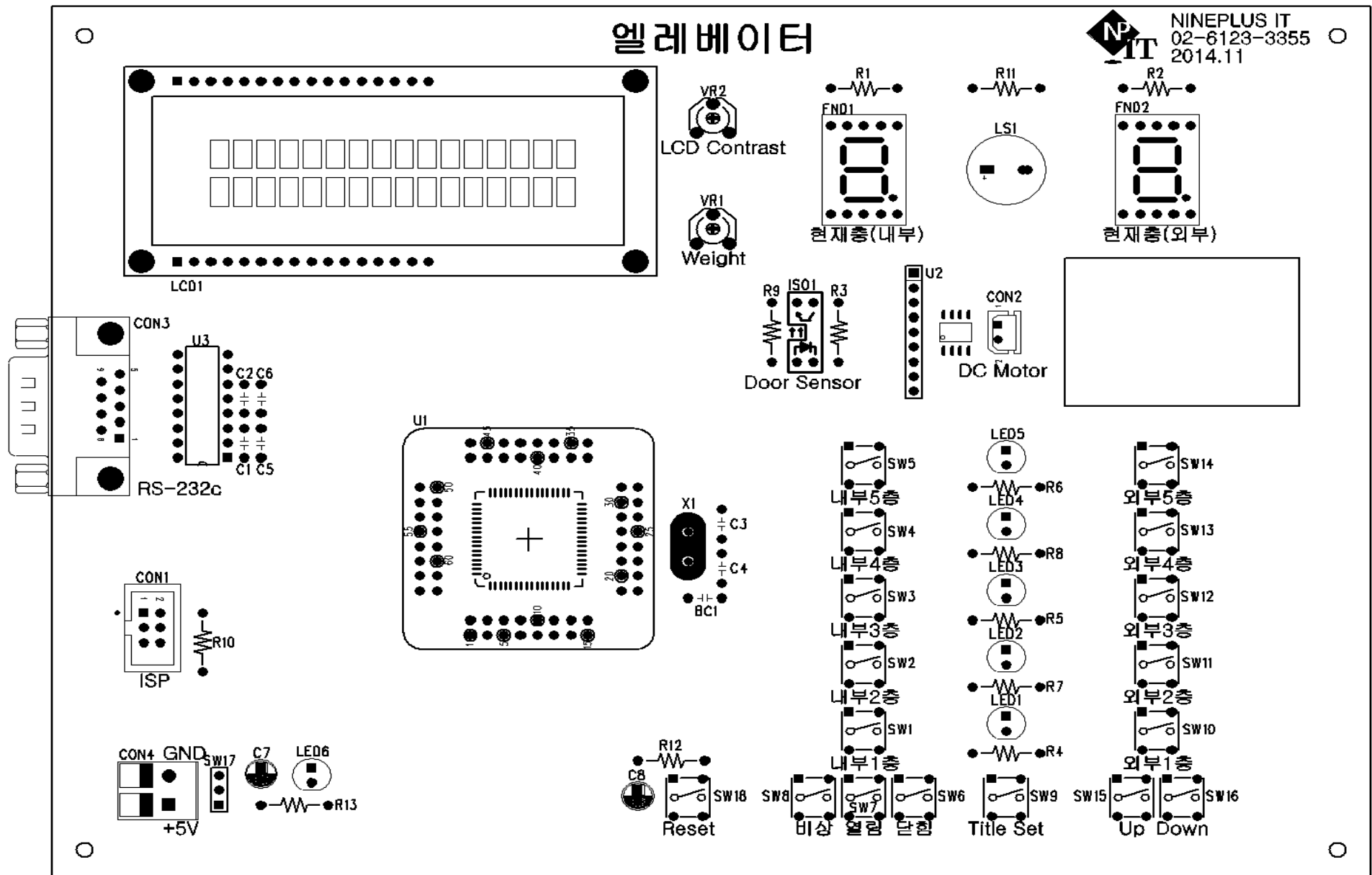
4-1. 지급 재료 목록

일련 번호	재 료 명	규 격(치수)	단위	수량	비 고
1	IC	ATMEGA128 (MCU)	개	1	
2	IC	LB1630 (MOTOR DRIVE)	개	1	
3	IC SOCKET	8PIN(DIP)	개	1	
4	IC	MAX232	개	1	
5	IC SOCKET	16PIN(DIP)	개	1	
6	LCD	HB16209	개	1	
7	LCD CNECTOR	1열 HEADER PIN (14PIN 암)	개	1	
8	LCD CONNECTOR	1열 HEADER PIN (14PIN 수)	개	1	
9	CONNECTOR	2열 HEADER PIN (8x2 PIN 암)	개	4	
10	CONNECTOR	2열 HEADER PIN (8x2 PIN 수)	개	4	
11	MOTOR	DC MOTOR(5V)	개	1	
12	MOTOR CONNECTOR	1열 CONNECTOR (2PIN 암)	개	1	
13	MOTOR CONNECTOR	1열 CONNECTOR (2PIN 수)	개	1	
14	BOX HEADER	2열 BOXHEADER(5x2 PIN 수)	개	1	
15	녹색단자	2P 5mm	개	1	
16	세라믹콘덴서	0.1uF	개	7	
17	세라믹콘덴서	20pF	개	2	
18	전해 콘덴서	10uF/16V	개	2	
19	전해 콘덴서	100uF/16V	개	1	
20	LED	RED	개	6	
21	저항	4.7KΩ	개	7	

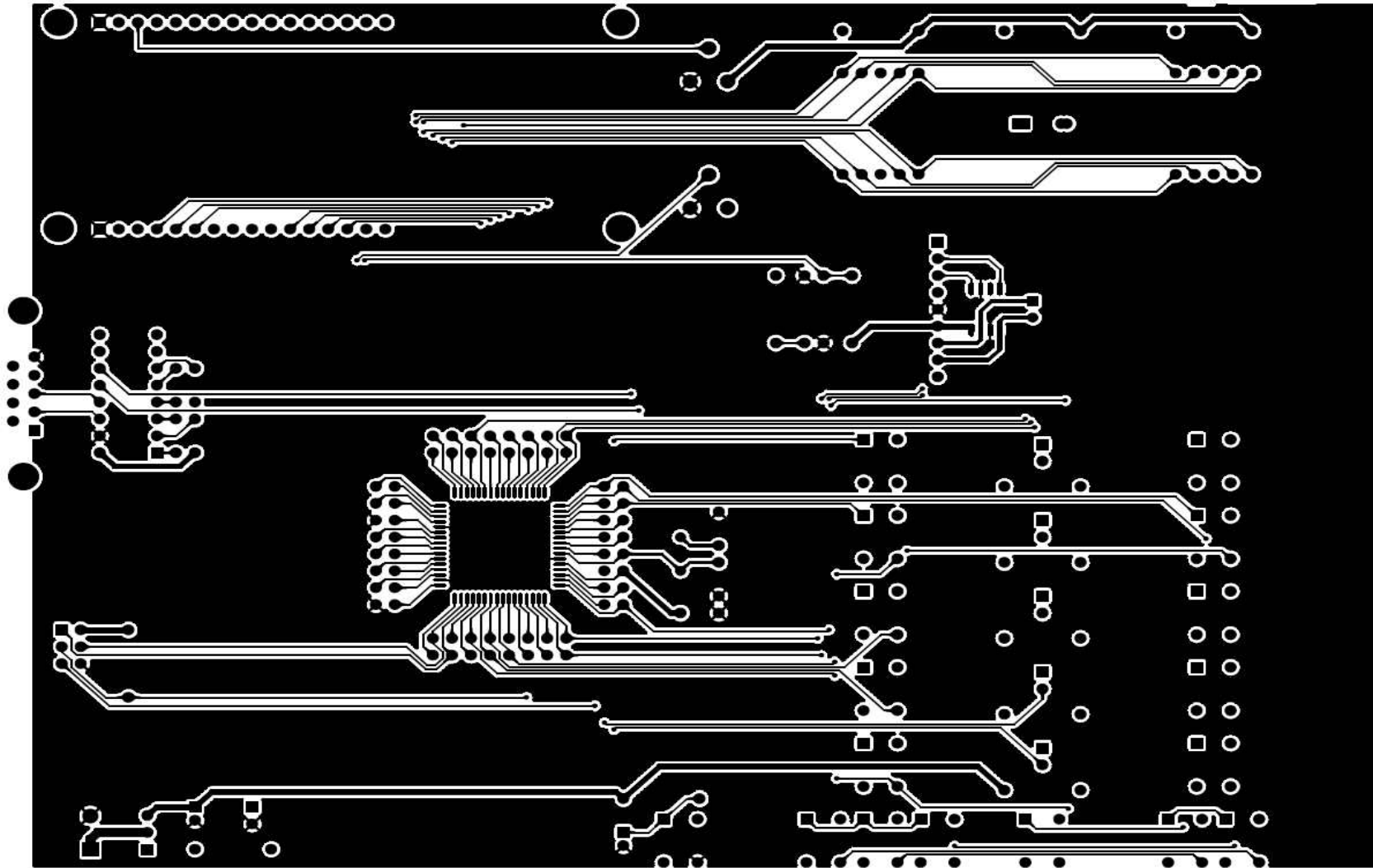
4-2. 지급 재료 목록

일련 번호	재 료 명	규 격(치수)	단위	수량	비 고
22	저항	330Ω	개	9	
23	가변저항	10KΩ	개	2	
24	SWITCH	TACT SWITCH (ITS-1105)	개	17	
27	CRYSTAL	16MHz	개	1	
28	부저	DC 5V	개	1	
29	통신커넥터	RS-232(DSUB 9P 수)	개	1	
30	포토인터럽트	GP1S53	개	1	
31	CONNECTOR	RS232(DSUB 9P 암)	개	2	
32	통신케이블	4P	m	2	
33	서포트	3∅x5mm	개	4	
34	너트	3∅	개	4	
35	PCB	BARE PCB	개	1	
36	납	SN60%	m	3	
37	단선	2색	m	0.3	

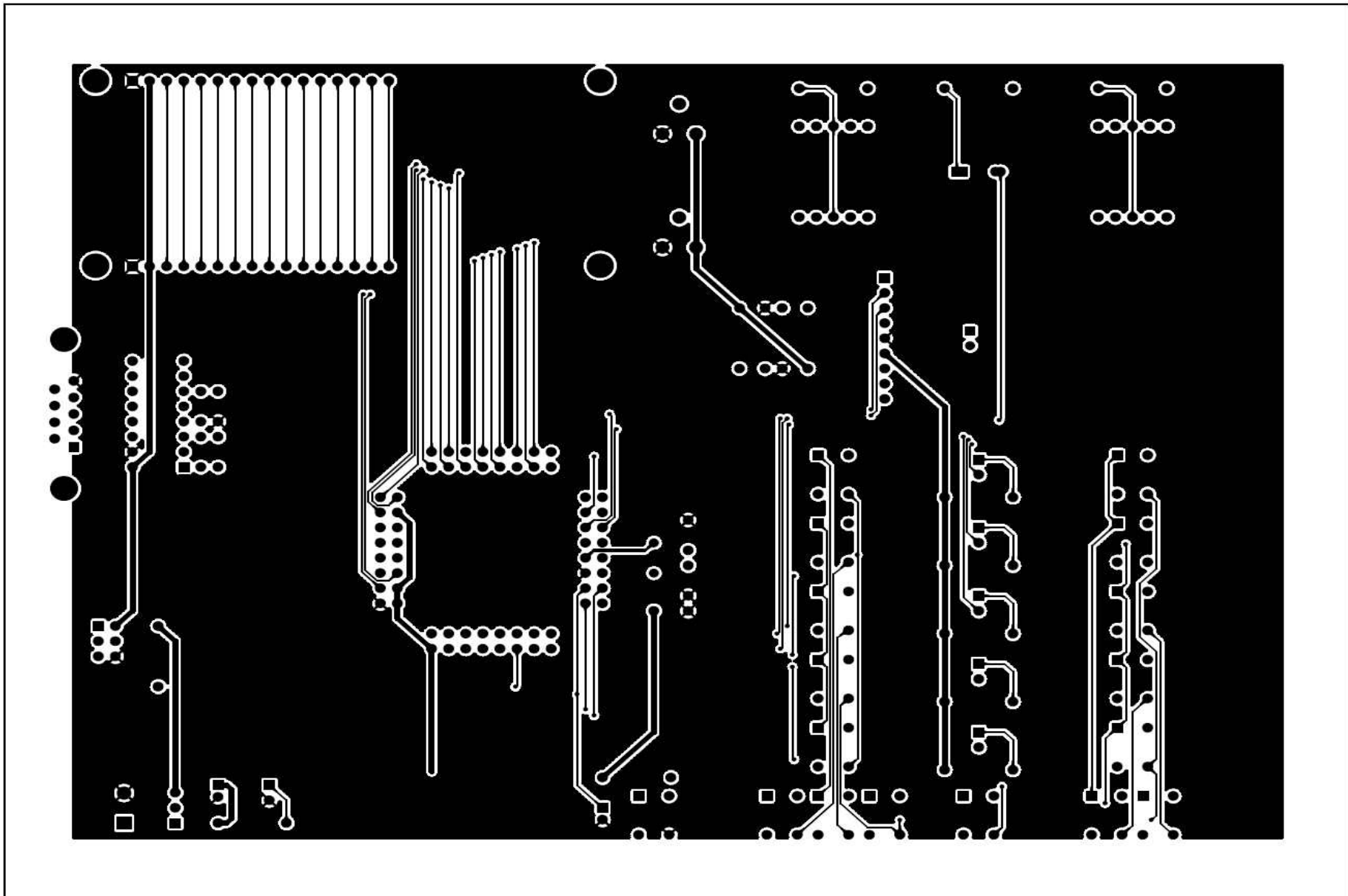
5-1. PCB(부품면)



5-2. PCB(TOP면)



5-3. PCB(BOTTOM면)



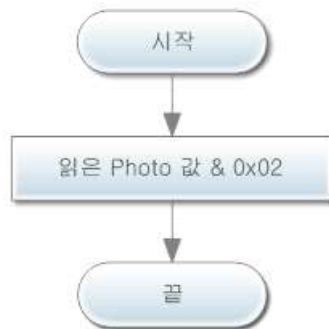
6. 배포용 소스

6-1. 주변장치 동작 소스

가. Photo Interrupt

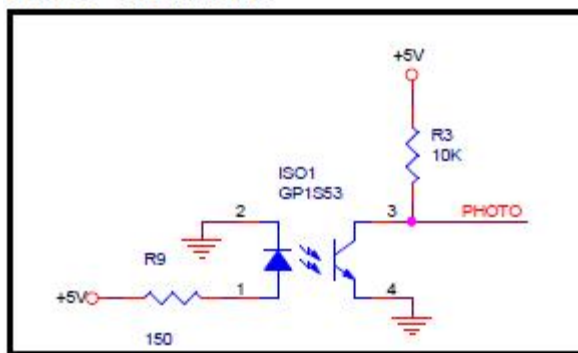
(1) 동작 흐름도

unsigned char photo_check(void) 함수의 흐름도



(2) 회로도 및 사진

PHOTO INTERRUPT



ADC	61		ADC
(ADC0) PF0	60		PHOTO
(ADC1) PF1	59		
(ADC2) PF2	58		



(3) C Source : photo.c

```
#include <avr/io.h>
#include <util/delay.h>

#include "photo.h"

#define PHOTO_IN    PINF
#define PHOTO_DDR    DDRF

unsigned char photo_check(void)
{
    return (PHOTO_IN & 0x02);
}

void photo_init(void)
{
    PHOTO_DDR &= ~0x02;
}
```

(4) Header : photo.h

```
#ifndef __PHOTO_H
#define __PHOTO_H

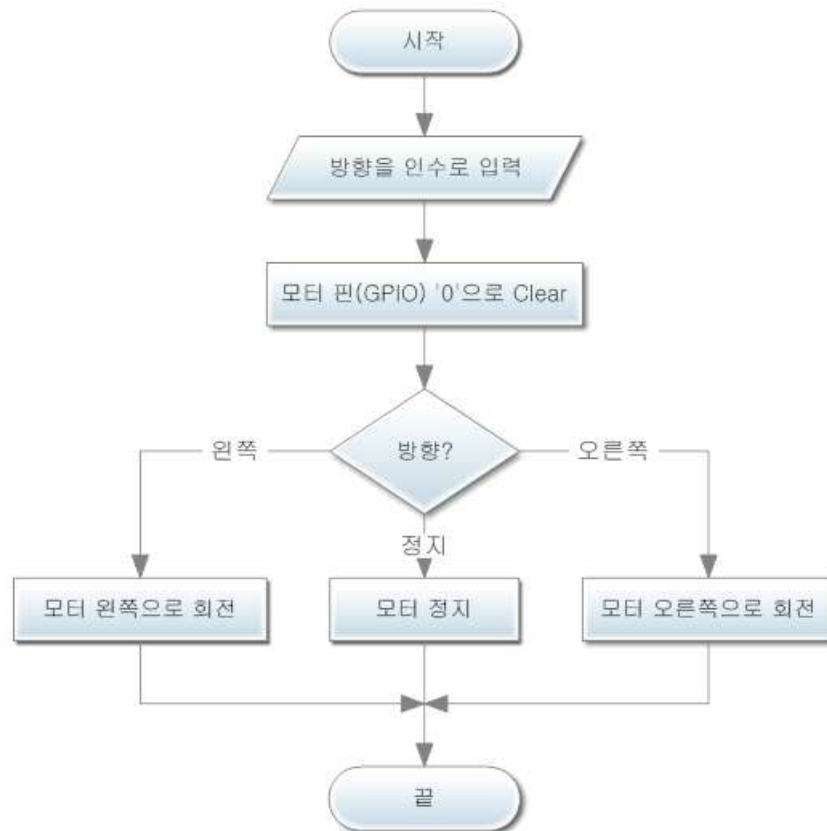
unsigned char photo_check(void);
void photo_init(void);

#endif // __PHOTO_H
```

나. DC Motor

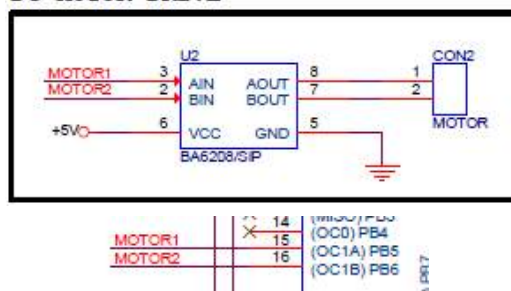
(1) 동작 흐름도

void dcmotor_spin(char direction) 함수의 흐름도



(2) 회로도 및 사진

DC MOTOR DRIVE



(3) C Source : dc_motor.c

```
#include <avr/io.h>
#include <util/delay.h>
```

```
#include "dc_motor.h"
```

```
#define DCMOTOR_OUT PORTB
#define DCMOTOR_DDR DDRB
```

```
void dcmotor_spin(char direction)
{
```

```
    DCMOTOR_OUT &= ~0x03;
```

```
    if( direction == LEFT )
```

```
        DCMOTOR_OUT |= LEFT;
```

```
    else if( direction == RIGHT )
```

```
        DCMOTOR_OUT |= RIGHT;
```

```
}

void dcmotor_init(void)
{
    DCMOTOR_DDR |= 0x60;
}
```

(4) Header : dc_motor.h

```
#ifndef __DC_MOTOR_H
#define __DC_MOTOR_H

#define STOP 0
#define LEFT 0x20
#define RIGHT 0x40

void dcmotor_spin(char direction);
void dcmotor_init(void);

#endif // __DC_MOTOR_H
```


다. Buzzer

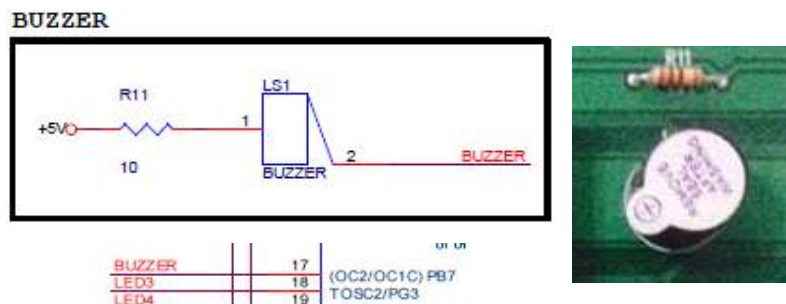
(1) 동작 흐름도

타이머1의 PCM을 이용한

void buzzer_out(unsigned char buz) 함수의 흐름도



(2) 회로도 및 사진



(3) C Source : buzzer.c

```
#include <avr/io.h>
#include <util/delay.h>

#include "buzzer.h"

#define BUZZER_OUT PORTB
#define BUZZER_DDR DDRB
#define BUZZER PB7

void timer1_init(void)
{
    TCCR1A = 0x82;
    TCCR1B = 0x1B;
    TCCR1C = 0x00;
    TCNT1 = 0;
    ICR1 = 0;
    OCR1C = 0;
}
```

```

void buzzer_out(unsigned char buz)
{
    unsigned int frequency_table[4] = { 0, 506, 568, 716 };

    ICR1 = frequency_table[buz];
    OCR1C = ICR1 / 2;
}

void buzzer_init(void)
{
    BUZZER_DDR = 1 << BUZZER;
    BUZZER_OUT |= 1 << BUZZER;

    timer1_init();
}

```

(4) Header : buzzer.h

```

#ifndef __BUZZER_H
#define __BUZZER_H

enum {
    NONE,
    BEEP,
    DING,
    DONG
};

void timer1_init(void);
void buzzer_out(unsigned char buz);
void buzzer_init(void);

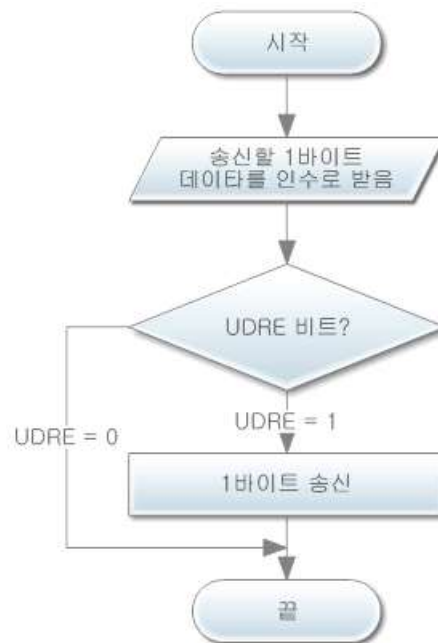
#endif // __BUZZER_H

```

라. 시리얼 통신

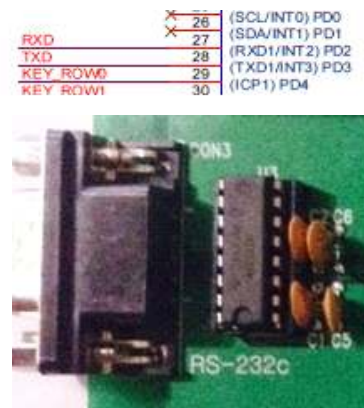
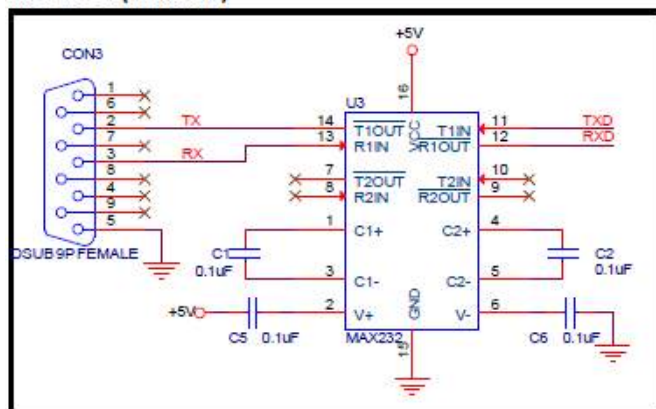
USART port를 이용한 시리얼 데이터 송신 함수인

void serial_transmit(unsigned char data) 함수의 흐름도



(2) 회로도 및 사진

RS-232C (SERIAL)



(3) C Source : Serial.c

```
#include <avr/io.h>
#include <util/delay.h>

#include "serial.h"

void serial_transmit(unsigned char data)
{
    while( !( UCSR1A & (1 << UDRE1)) );

    UDR1 = data;
}

void serial_string(char *str)
{

```

```

        while( *str )    serial_transmit(*str++);
    }

    unsigned char serial_receive(void)
    {
        while ( !(UCSR1A & (1 << RXC)) );

        return UDR1;
    }

    void serial_init(unsigned int baudrate)
    {
        UCSR1A = 0x00;
        UCSR1B = 0x08;
        UCSR1C = 0x06;
        UBRR1H = baudrate >> 8;
        UBRR1L = baudrate & 0x0FF;
    }

```

(4) Header : Serial.h

```

#ifndef __SERIAL_H
#define __SERIAL_H

#define B2400      416
#define B4800      207
#define B9600      103
#define B19200     51
#define B38400     25
#define B76800     12

void serial_transmit(unsigned char data);
void serial_string(char *str);
unsigned char serial_receive(void);
void serial_init(unsigned int baudrate);

#endif // __SERIAL_H

```

마. 가변저항

(1) 동작 흐름도

ADC 인터럽트를 이용한 가변저항 읽기 흐름도



(2) 회로도 및 사진



(3) C Source : var.c

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

#include "var.h"

volatile int weight;

ISR(ADC_vect)
{
    weight = (int)(ADC / 7.85);
    ADCSRA |= 0x40;
}

int var_read(void)
{
    ADCSRA |= 0x40;
    while( !(ADCSRA & 0x10) );

    return ADC;
}
```

```

void var_start(void)
{
    ADCSRA |= 0x40;
}

void var_init(void)
{
    ADCSRA = 0x8F;
    ADMUX = 0x00;
}

```

(4) Header : var.h

```

#ifndef __VAR_H
#define __VAR_H

int var_read(void);
void var_start(void);
void var_init(void);

#endif // __VAR_H

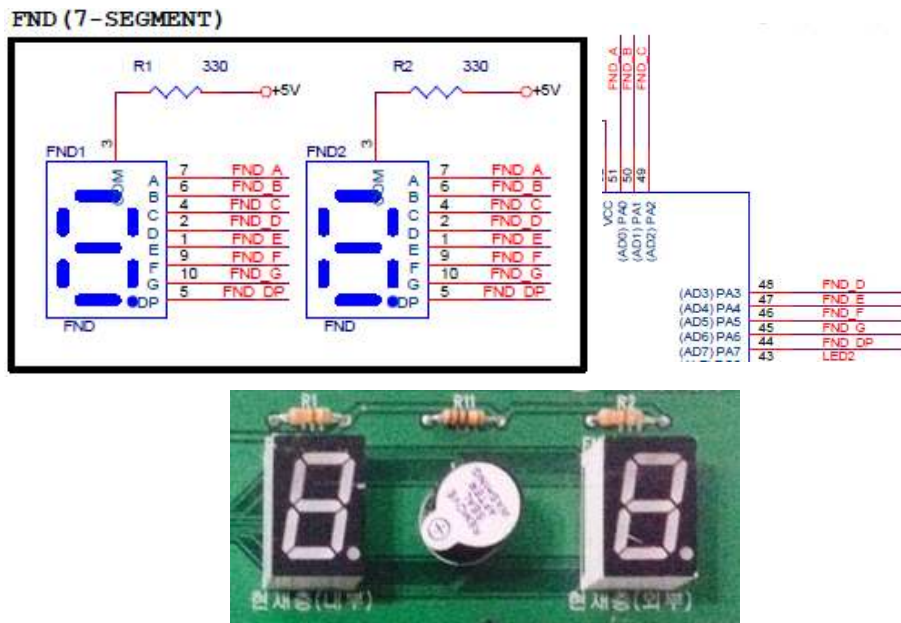
```

바. FND

(1) 동작 흐름도



(2) 회로도 및 사진



(3) C Source : fnd.c

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdio.h>

#include "Fnd.h"

#define SEG_OUT PORTA
#define SEG_DDR DDRA

void fnd_out(unsigned char num)
{
    char segment[11] = {
        0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xD8, 0x80, 0x90, 0xFF
    };

    SEG_OUT = segment[num];
}
```

```
}  
  
void fnd_init(void)  
{  
    SEG_DDR = 0xFF;  
    fnd_out(10);  
}
```

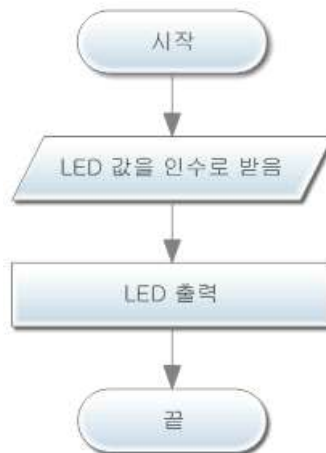
(4) Header : fnd.h

```
#ifndef __FND_H  
#define __FND_H  
  
void fnd_out(unsigned char num);  
void fnd_init(void);  
  
#endif // FND_H
```

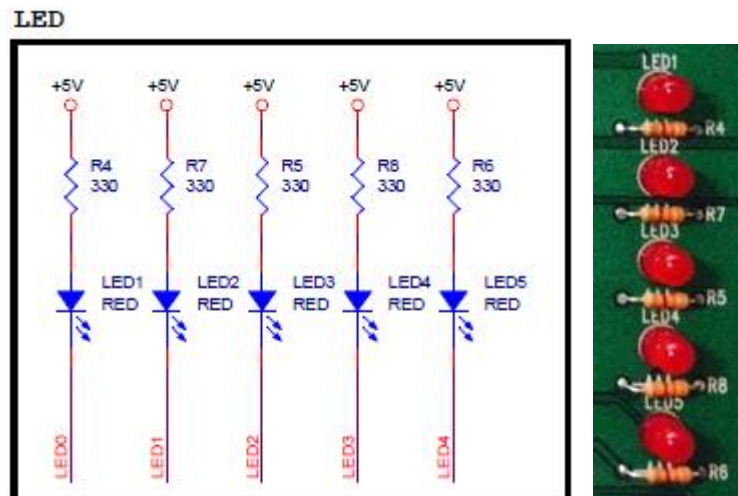

사. LED

(1) 동작 흐름도

void led_light(unsigned char led) 함수의 흐름도



(2) 회로도 및 사진



(3) C Source : led.c

```
#include <avr/io.h>
#include <util/delay.h>

#include "led.h"

#define LED_OUT  PORTG
#define LED_DDR  DDRG

void led_light(unsigned char led)
{
    LED_OUT &= ~led;
}

void led_init(void)
{
    LED_DDR = 0x1F;
    LED_OUT = 0x1F;
}
```

(4) Header : led.h

```
#ifndef __LED_H
#define __LED_H

#define LED1      0x01
#define LED2      0x02
#define LED3      0x04
#define LED4      0x08
#define LED5      0x10
#define LED_OFF   0x1F

void led_light(unsigned char led);
void led_init(void);

#endif // __LED_H
```


(3) C Source : lcd.c

```
#include <avr/io.h>
#include <util/delay.h>

#include "lcd.h"

#define LCD_OUT    PORTC
#define LCD_IN     PINC
#define LCD_DDR    DDRC

#define LCD_RS     0x01
#define LCD_RW     0x02
#define LCD_E      0x04

#define DDRAM      0x80

void lcd_bussycheck(void)
{
    LCD_DDR = 0x0F;

    LCD_OUT = 0x00;
    LCD_OUT |= LCD_RW;
    LCD_OUT |= LCD_E;
    while( LCD_IN & 0x80 );
    LCD_OUT &= ~LCD_E;

    LCD_DDR = 0xFF;
}

unsigned char lcd_command_read(void)
{
    char cmd;

    LCD_DDR = 0x0F;
    LCD_OUT = 0x00;

    LCD_OUT |= LCD_RW;

    LCD_OUT |= LCD_E;
    asm("nop");
    cmd = LCD_IN & 0xF0;
    LCD_OUT &= ~LCD_E;

    LCD_OUT |= LCD_E;
    asm("nop");
    cmd |= LCD_IN >> 4;
    LCD_OUT &= ~LCD_E;

    return cmd;
}

void lcd_command_write(unsigned char cmd)
{
    lcd_bussycheck();

    LCD_OUT = 0x00;
```

```

    LCD_OUT |= cmd & 0xF0;
    LCD_OUT |= LCD_E;
    LCD_OUT &= ~LCD_E;

    LCD_OUT &= 0x0F;
    LCD_OUT |= cmd << 4;
    LCD_OUT |= LCD_E;
    LCD_OUT &= ~LCD_E;

    _delay_ms(2);
}

unsigned char lcd_data_read(void)
{
    char data;

    LCD_DDR = 0x0F;
    LCD_OUT = 0x00;

    LCD_OUT |= LCD_RS;
    LCD_OUT |= LCD_RW;

    LCD_OUT |= LCD_E;
    asm("nop");
    data = LCD_IN & 0xF0;
    LCD_OUT &= ~LCD_E;

    LCD_OUT |= LCD_E;
    asm("nop");
    data |= LCD_IN >> 4;
    LCD_OUT &= ~LCD_E;

    return data;
}

void lcd_data_write(unsigned char data)
{
    lcd_bussycheck();

    LCD_OUT = 0x00;

    LCD_OUT |= LCD_RS;

    LCD_OUT |= data & 0xF0;
    LCD_OUT |= LCD_E;
    LCD_OUT &= ~LCD_E;

    LCD_OUT &= 0x0F;
    LCD_OUT |= data << 4;
    LCD_OUT |= LCD_E;
    LCD_OUT &= ~LCD_E;

    _delay_us(50);
}

void lcd_string(char *str)

```

```

{
    while( *str )lcd_data_write(*str++);
}

void lcd_gotoxy(char x, char y)
{
    lcd_command_write(DDRAM | (0x40 * y) | x);
}

void lcd_init(void)
{
    LCD_DDR = 0xFF;

    lcd_command_write(0x20);
    _delay_ms(10);
    lcd_command_write(0x20);
    _delay_ms(10);
    lcd_command_write(0x20);
    _delay_ms(10);
    lcd_command_write(0x28);
    lcd_command_write(0x08);
    lcd_command_write(0x01);
    lcd_command_write(0x06);
    _delay_ms(10);
    lcd_command_write(0x0C);
}

```

(4) Header : lcd.h

```

#ifndef __LCD_H
#define __LCD_H

#define LCD_ON      0x0C
#define LCD_OFF     0x08
#define LCD_CLEAR   0x01

void lcd_bussycheck(void);
unsigned char lcd_command_read(void);
void lcd_command_write(unsigned char cmd);
unsigned char lcd_data_read(void);
void lcd_data_write(unsigned char data);
void lcd_string(char *str);
void lcd_gotoxy(char x, char y);
void lcd_init(void);

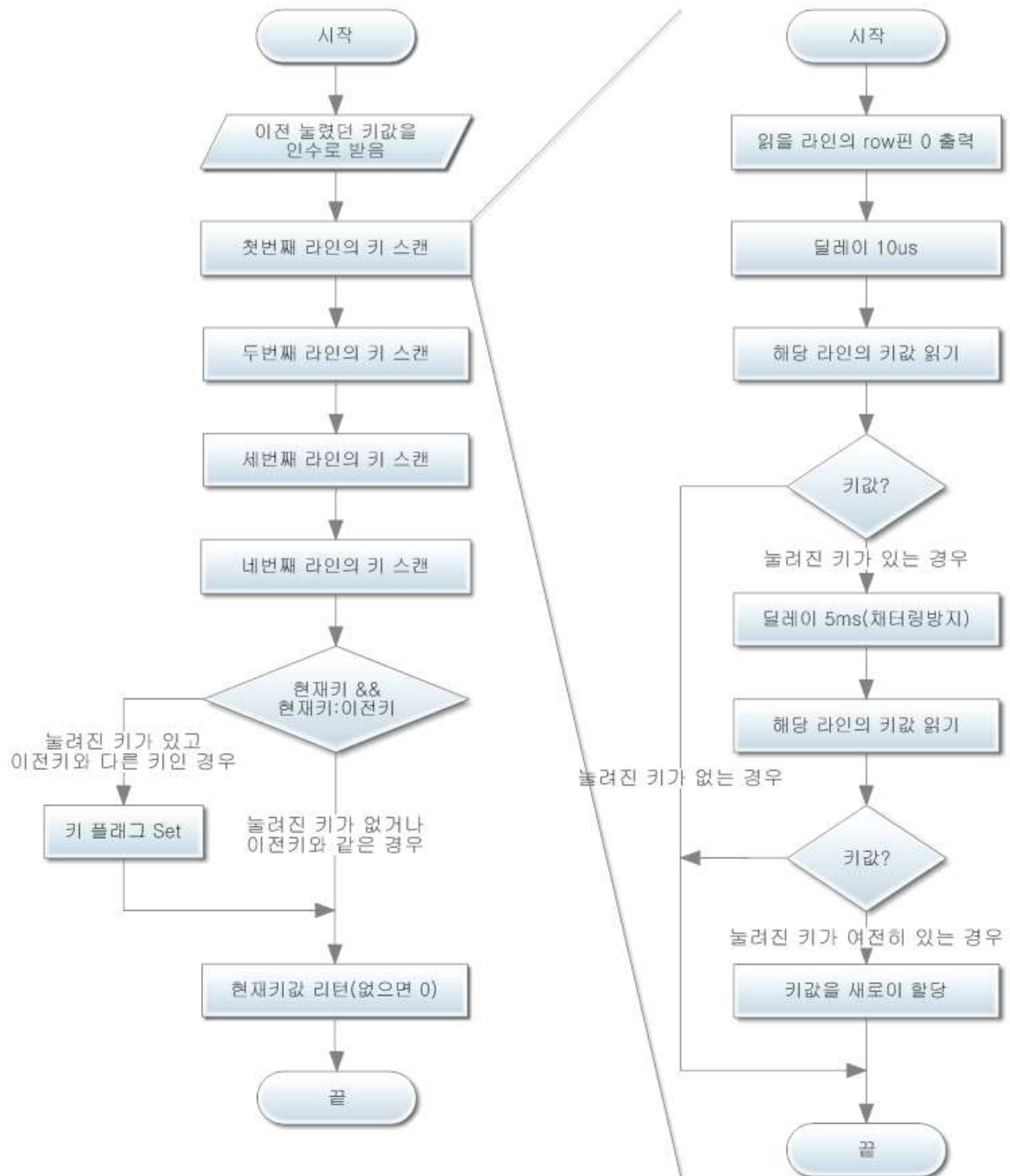
#endif // __LCD_H

```

자. Matrix Key

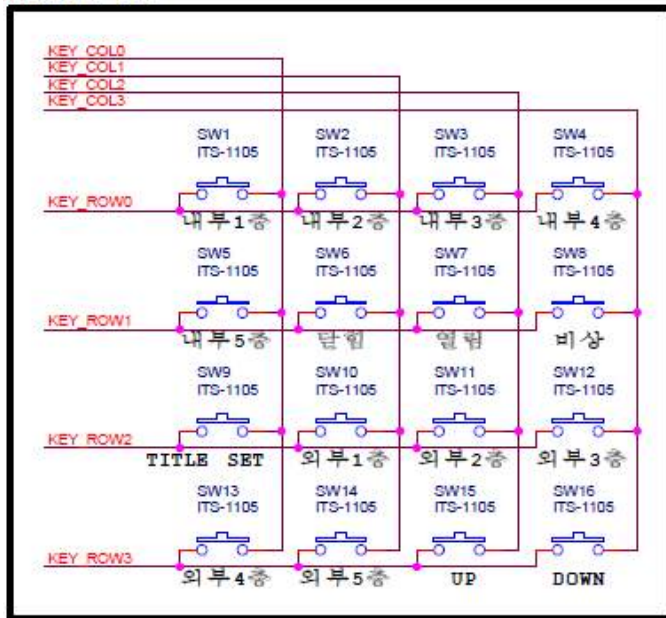
(1) 동작 흐름도

unsigned char getkey(unsigned char keyin) 함수의 흐름도

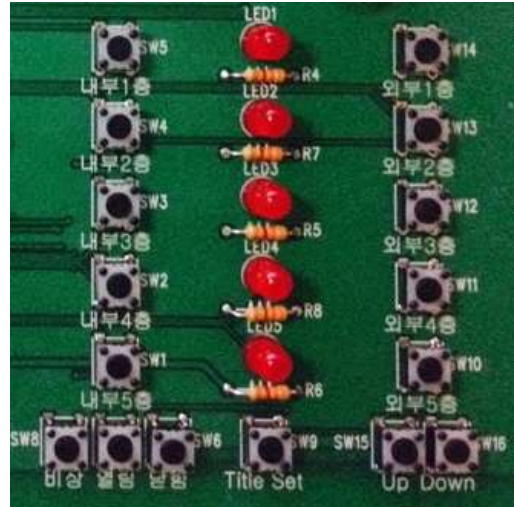


(2) 회로도 및 사진

MATRIX KEY



KEY_COL0	5	(XCK1/INT3) PE2	TXD	28	(RXD1/INT2) PD
KEY_COL1	6	(OC3A/AIN1) PE3	KEY_ROW0	29	(TXD1/INT3) PD
KEY_COL2	7	(OC3B/INT4) PE4	KEY_ROW1	30	(ICP1) PD4
KEY_COL3	8	(OC3C/INT5) PE5	KEY_ROW2	31	(XCK1) PD5
	9	(T3/INT6) PE6	KEY_ROW3	32	(T1) PD6
	10	(ICP3/INT7) PE7			(T2) PD7



(3) C Source : key.c

```
#include <avr/io.h>
#include <util/delay.h>

#include "key.h"

#define KEY_OUT    PORTD
#define KEY_IN     PINE
#define KEY_ODDR   DDRD
#define KEY_IDDR   DDRE

volatile char key_flag;

unsigned char getkey(unsigned char keyin)
{
    unsigned char key;

    key_flag = 0;

    KEY_OUT &= ~0x10;
    _delay_us(10);
    key = ~KEY_IN & 0xF0;
    if( key ) {
        _delay_ms(5);
        key = ~KEY_IN & 0xF0;
        if( key ) {
            if( key == 0x10 )    key = KEY_IN1;
            else if( key == 0x20 ) key = KEY_IN2;
            else if( key == 0x40 ) key = KEY_IN3;
            else if( key == 0x80 ) key = KEY_IN4;
        }
    }
    else {
```



```

KEY_OUT &= ~0x20;
_delay_us(10);
key = ~KEY_IN & 0xF0;
if( key ) {
    _delay_ms(5);
    key = ~KEY_IN & 0xF0;
    if( key ) {
        if( key == 0x10 )    key = KEY_IN5;
        else if( key == 0x20 ) key = KEY_CLOSE;
        else if( key == 0x40 ) key = KEY_OPEN;
        else if( key == 0x80 ) key = KEY_EMERGENCY;
    }
}
else {
    KEY_OUT &= ~0x40;
    _delay_us(10);
    key = ~KEY_IN & 0xF0;
    if( key ) {
        _delay_ms(5);
        key = ~KEY_IN & 0xF0;
        if( key ) {
            if( key == 0x10 )    key = KEY_TITLE;
            else if( key == 0x20 ) key = KEY_OUT1;
            else if( key == 0x40 ) key = KEY_OUT2;
            else if( key == 0x80 ) key = KEY_OUT3;
        }
    }
    else {
        KEY_OUT &= ~0x80;
        _delay_us(10);
        key = ~KEY_IN & 0xF0;
        if( key ) {
            _delay_ms(5);
            key = ~KEY_IN & 0xF0;
            if( key ) {
                if( key == 0x10 )    key = KEY_OUT4;
                else if( key == 0x20 ) key = KEY_OUT5;
                else if( key == 0x40 ) key = KEY_UP;
                else if( key == 0x80 ) key = KEY_DOWN;
            }
        }
    }
}

if( key && (key != keyin) )    key_flag = 1;

return key;
}

void key_init(void)
{
    KEY_ODDR |= 0xF0;
    KEY_OUT |= 0xF0;

    KEY_IDDR &= 0xF0;
    PORTE |= 0xF0;

```

```
}
```

(4) Header : key.h

```
#ifndef __KEY_H
#define __KEY_H

#define KEY_IN1      1
#define KEY_IN2      2
#define KEY_IN3      3
#define KEY_IN4      4
#define KEY_IN5      5

#define KEY_CLOSE     21
#define KEY_OPEN      22
#define KEY_EMERGENCY 23
#define KEY_TITLE     24

#define KEY_OUT1      11
#define KEY_OUT2      12
#define KEY_OUT3      13
#define KEY_OUT4      14
#define KEY_OUT5      15

#define KEY_UP        25
#define KEY_DOWN      26

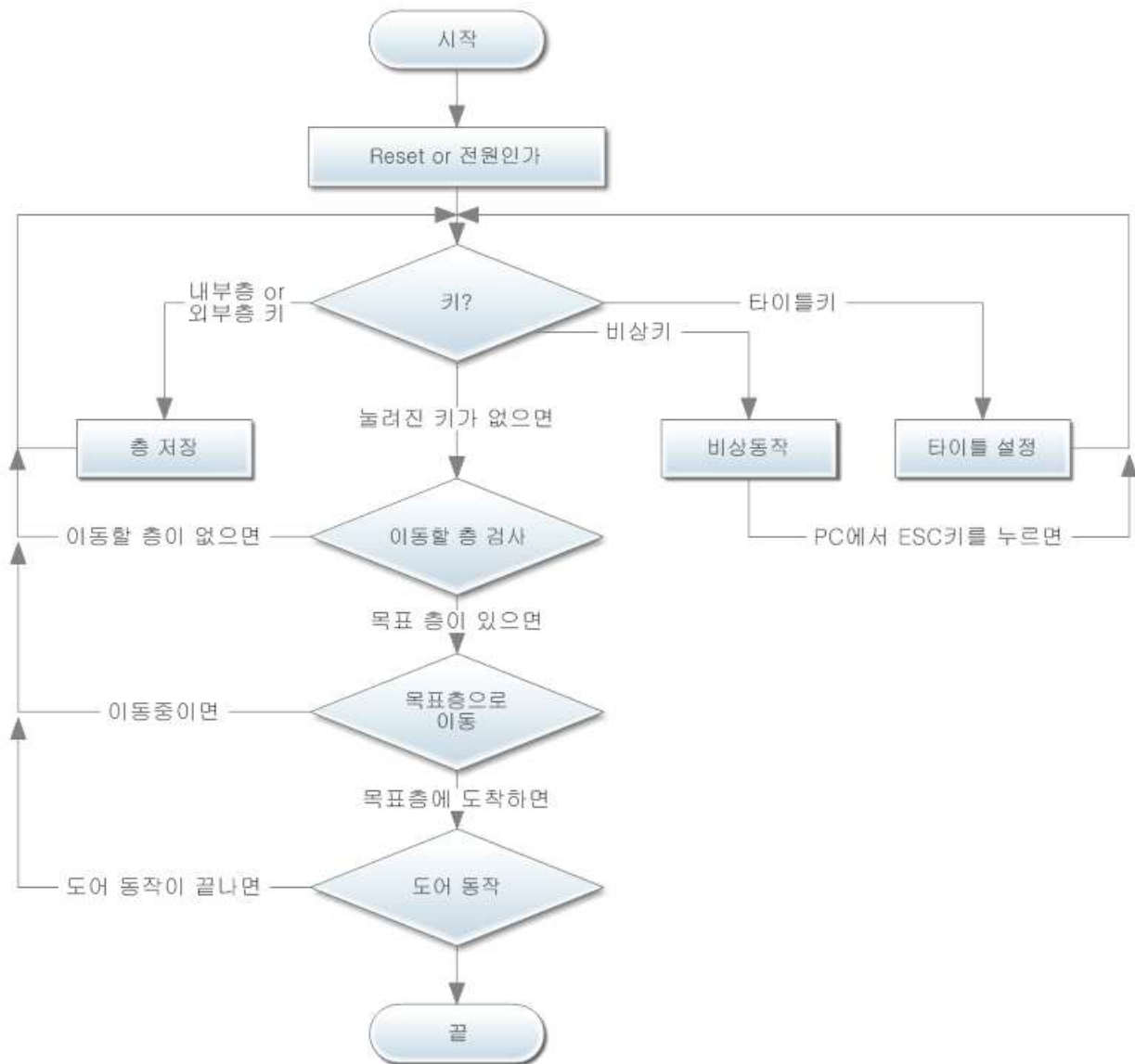
unsigned char getkey(unsigned char keyin);
void key_init(void);

#endif // __KEY_H
```

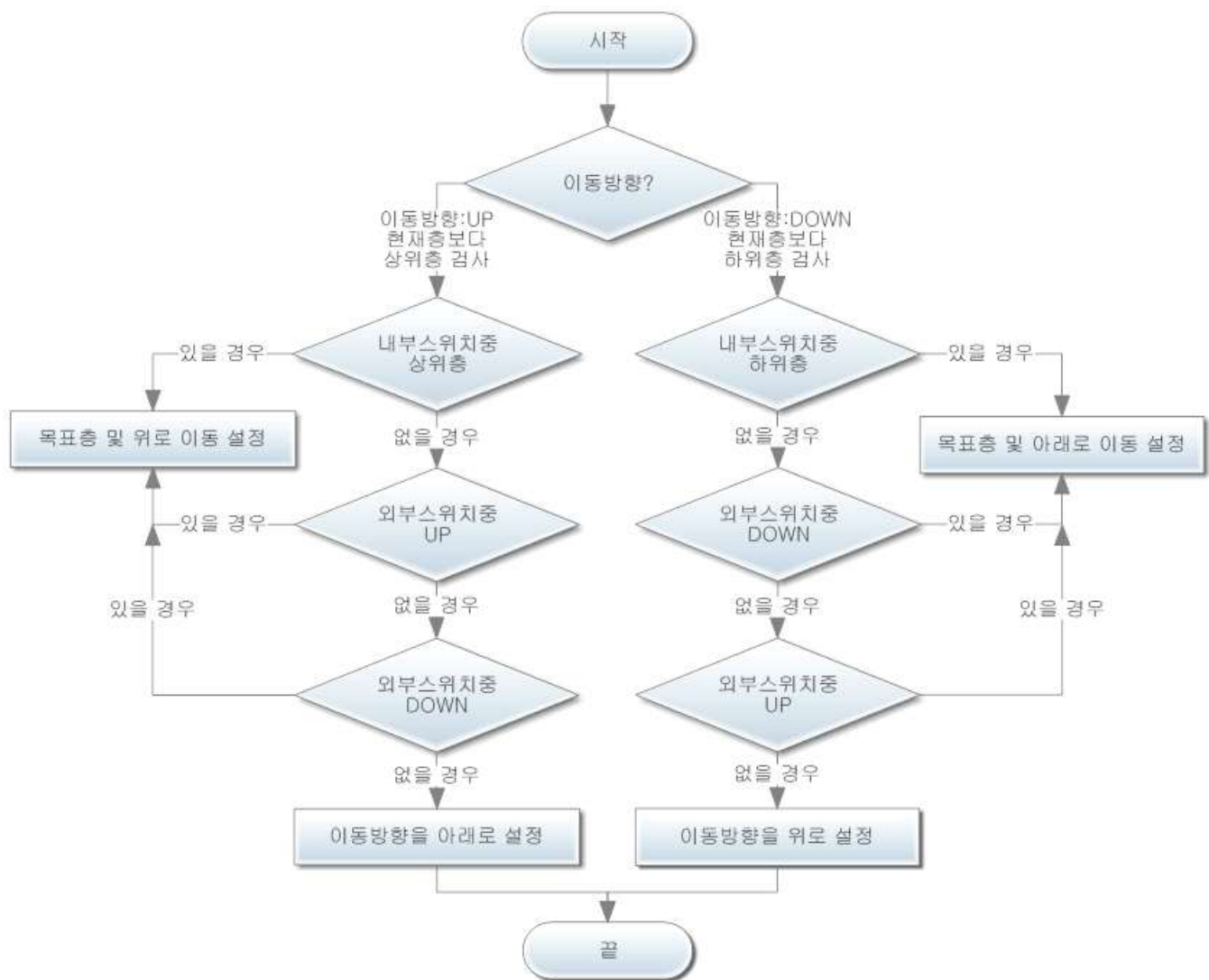
6-2. 엘리베이터 전체 동작(Main) 소스

가. 동작 흐름도

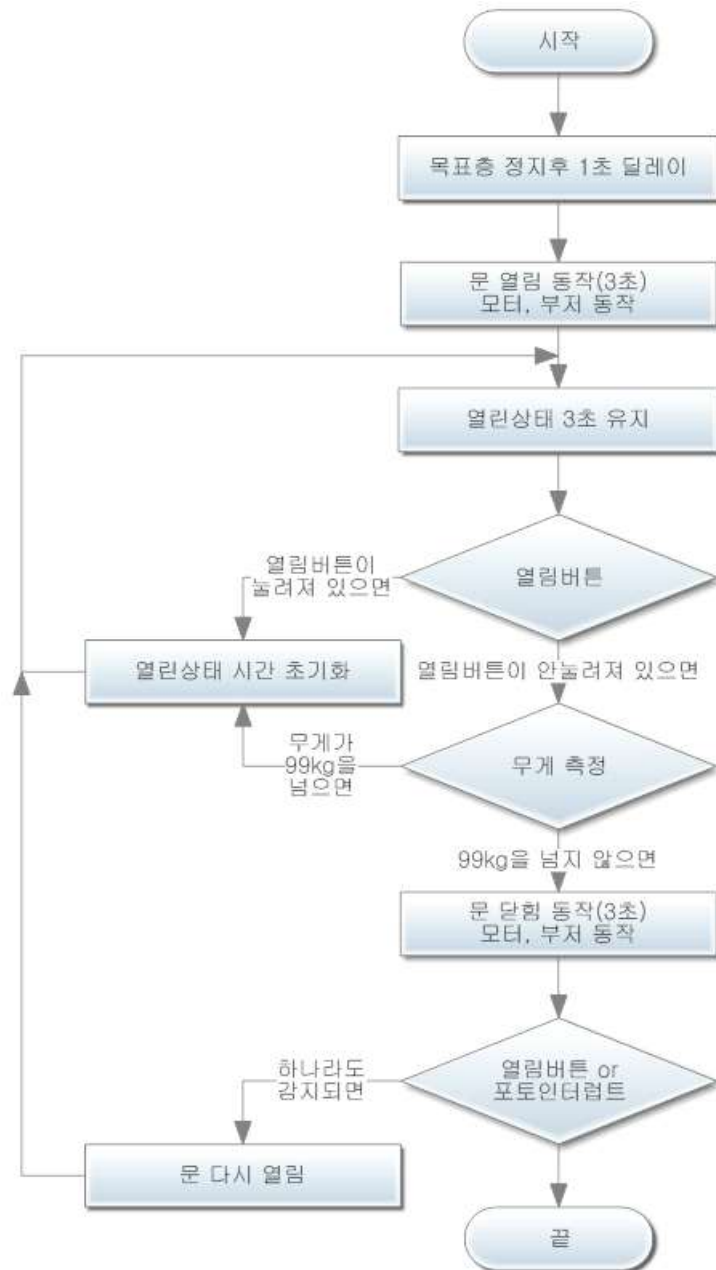
(1) 전체 흐름도



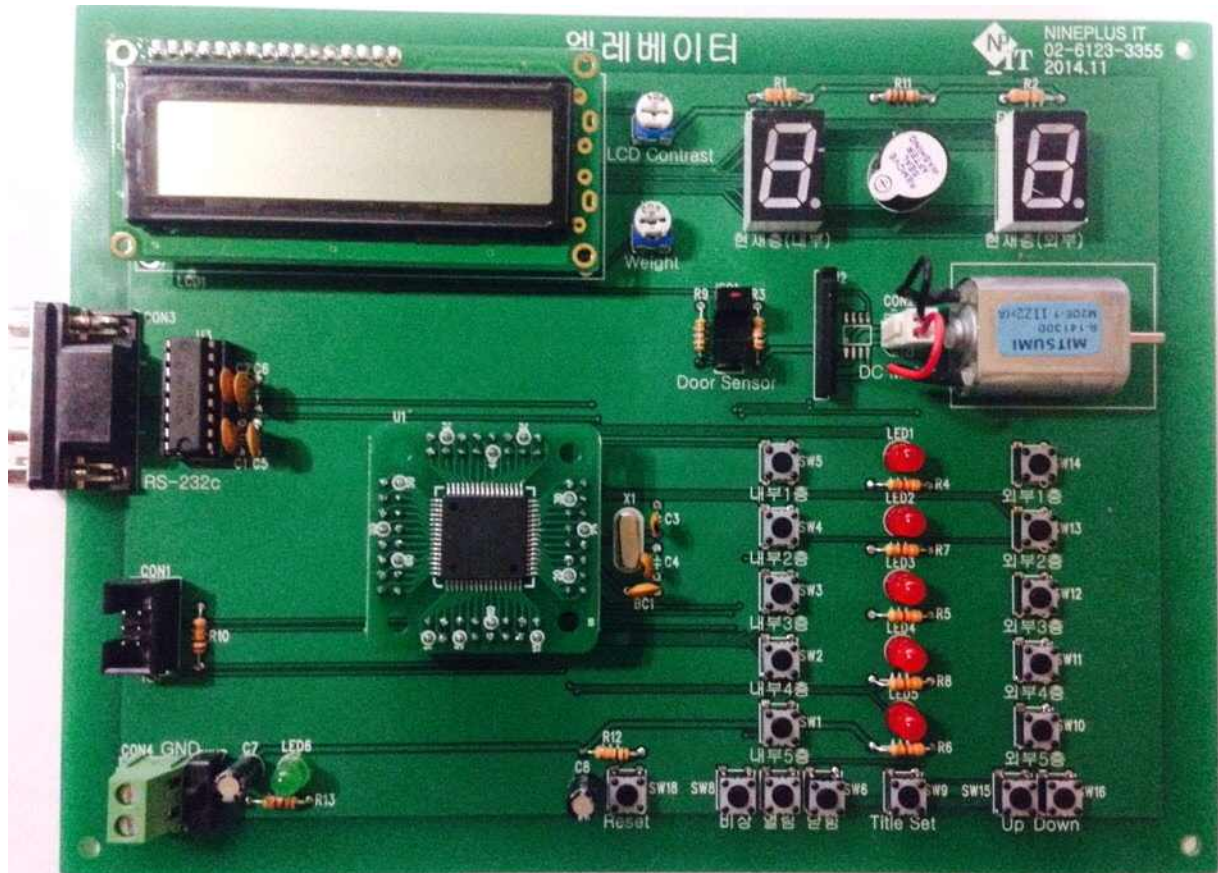
(2) 이동할 층 검사하는 흐름도



(3) 도어 동작의 흐름도



나. 전체 사진



다. C Source : main.c

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/eeprom.h>
#include <util/delay.h>
```

```
#include <stdio.h>
#include <string.h>
```

```
#include "photo.h"
#include "dc_motor.h"
#include "buzzer.h"
#include "serial.h"
#include "var.h"
#include "fnd.h"
#include "led.h"
#include "lcd.h"
#include "key.h"
```

```
#define MAX_FLOOR 5
```

```
#define READY 'R'
#define CLOSE 'C'
#define OPEN 'O'
```

```
struct floor {
    unsigned char in;
    unsigned char out_up;
```

```

    unsigned char out_down;
};

extern volatile char key_flag;
extern volatile int weight;
extern volatile unsigned char serial_flag, serial_buf[20];

volatile unsigned char cur_floor, out_floor, elevator_dir, target_floor, updn;
volatile unsigned char door_dir, door_state, door_flag;
volatile unsigned char move_flag, move_tick, move_sec;
volatile unsigned char warning_flag, warning_tick, emergency_flag, emergency_tick;
volatile struct floor elevator[MAX_FLOOR];
volatile char floor_title[5][15];

enum {
    UP = 1,
    DOWN
};

enum {
    FLOOR1 = 1,
    FLOOR2,
    FLOOR3,
    FLOOR4,
    FLOOR5
};

ISR(TIMER0_OVF_vect)
{
    TCNT1 = 256 - 250;

    if( move_flag ) {
        move_tick++;
        if( move_tick == 250 ) {
            move_tick = 0;
            move_sec++;
        }
    }

    if( warning_flag ) {
        if( ++warning_tick == 250 ) warning_tick = 0;
    }

    if( emergency_flag ) {
        if( ++emergency_tick == 250 ) emergency_tick = 0;
    }
}

void timer0_init(void)
{
    TCCR0 = 0x06;
    TCNT0 = 256 - 250;
    TIMSK |= 0x01;
}

void mcu_init(void)
{

```

```

    photo_init();
    dcmotor_init();
    buzzer_init();
    serial_init(B9600);
    var_init();
    fnd_init();
    led_init();
    lcd_init();
    key_init();

    timer0_init();
}

void variable_init(void)
{
    int i, j;

    cur_floor = FLOOR1;
    out_floor = 0;
    updn = 0;

    door_state = CLOSE;
    door_dir = 0;
    door_flag = 0;

    elevator_dir = STOP;
    target_floor = cur_floor;

    move_flag = 0;

    warning_flag = 0;
    emergency_flag = 0;

    memset(floor_title, 0x00, sizeof(floor_title));
    if( eeprom_read_byte(0) == 107 ) {
        for( i = 0; i < 5; i++ ) {
            for( j = 0; j < 13; j++ ) floor_title[i][j] = eeprom_read_byte((0x10 * (i + 1)) + j);
        }
    }

    memset((char *)elevator, 0, sizeof(elevator));
}

void start_elevator(void)
{
    led_light(LED_OFF);

    lcd_gotoxy(0, 0);
    lcd_string("Elevator System ");
    lcd_gotoxy(0, 1);
    lcd_string("      Loading");
    dcmotor_spin(LEFT);
    fnd_out(1);

    _delay_ms(1000);

    lcd_data_write('!');
}

```



```

    dcmotor_spin(STOP);
    fnd_out(0);

    _delay_ms(1000);

    lcd_data_write('!');
    dcmotor_spin(RIGHT);
    fnd_out(1);

    _delay_ms(1000);

    dcmotor_spin(STOP);
}

void init_screen(void)
{
    lcd_command_write(LCD_CLEAR);
    lcd_gotoxy(0, 0);
    printf("%1dF:%s", cur_floor, floor_title[cur_floor - 1]);
    lcd_gotoxy(0, 1);
    printf("W:%3dkg  %c ", weight, door_state);

    for( int i = 0; i < 5; i++ ) {
        if( elevator[i].in ) lcd_data_write(i + '1');
        else                 lcd_data_write(' ');
    }
}

void led_out(void)
{
    PORTG = 0xFF;
    for( int i = 0; i < 5; i++ ) {
        if( elevator[i].out_up || elevator[i].out_down ) led_light(0x01 << i);
    }
}

void floor_check(void)
{
    int i;

    if( cur_floor == FLOOR1 )    elevator_dir = UP;
    else if( cur_floor == FLOOR5 ) elevator_dir = DOWN;

    if( elevator_dir == UP ) {
        for( i = cur_floor; i < 5; i++ ) {
            if( elevator[i].in || elevator[i].out_up ) {
                target_floor = i + 1;
                updn = UP;
                break;
            }
        }

        if( i == 5 ) {
            for( i = 4; i > (cur_floor - 1); i-- ) {
                if( elevator[i].out_down ) {
                    target_floor = i + 1;
                    elevator_dir = UP;
                }
            }
        }
    }
}

```

```

        updn = DOWN;
        break;
    }
}

if( i == (cur_floor - 1) ) elevator_dir = DOWN;
}
}
else if( elevator_dir == DOWN ) {
    for( i = (cur_floor - 1); i >= 0; i-- ) {
        if( elevator[i].in || elevator[i].out_down ) {
            target_floor = i + 1;
            updn = DOWN;
            break;
        }
    }

    if( i == -1 ) {
        for( i = 0; i < (cur_floor - 1); i++ ) {
            if( elevator[i].out_up ) {
                target_floor = i + 1;
                elevator_dir = DOWN;
                updn = UP;
                break;
            }
        }

        if( i == (cur_floor - 1) ) elevator_dir = UP;
    }
}

void move_elevator(void)
{
    if( !move_flag && !door_flag ) {
        if( elevator_dir ) {
            move_tick = 0;
            move_sec = 0;
            move_flag = 1;
        }
    }
    else if( move_flag && !door_flag ) {
        if( move_sec == 2 ) {
            move_sec = 0;

            if( elevator_dir == UP )        cur_floor++;
            else if( elevator_dir == DOWN ) cur_floor--;

            if( cur_floor == target_floor ) {
                door_flag = 1;
                move_flag = 0;
            }

            init_screen();
            fnd_out(cur_floor);
        }
    }
}

```

```

}

void move_door(void)
{
    if( !move_flag ) {
        move_tick = 0;
        move_sec = 0;

        door_dir = READY;
        door_state = CLOSE;

        move_flag = 1;
    }
    else {
        if( move_sec == 1 ) {
            if( door_dir == READY ) {
                move_tick = 0;
                move_sec = 0;

                door_dir = OPEN;
                door_state = OPEN;
                dcmotor_spin(LEFT);

                buzzer_out(DING);
            }
            else if( (door_dir == OPEN) || (door_dir == CLOSE) ) {
                if( move_tick >= 125 ) buzzer_out(DONG);
            }
        }
        else if( move_sec == 3 ) {
            if( door_dir == OPEN ) {
                move_tick = 0;
                move_sec = 0;

                door_dir = STOP;
                dcmotor_spin(STOP);

                buzzer_out(0);
            }
            else if( door_dir == STOP ) {
                if( door_state == OPEN ) {
                    if( warning_flag ) move_sec = 0;
                    else {
                        move_tick = 0;
                        move_sec = 0;

                        door_dir = CLOSE;
                        dcmotor_spin(RIGHT);

                        buzzer_out(DING);
                    }
                }
            }
        }
        else if( door_dir == CLOSE ) {
            door_state = CLOSE;
            door_dir = STOP;
            dcmotor_spin(STOP);
        }
    }
}

```

```

        buzzer_out(NONE);

        elevator[cur_floor - 1].in = 0;
        if( (cur_floor == FLOOR1) || (cur_floor == FLOOR5) ) {
            elevator[cur_floor - 1].out_up = 0;
            elevator[cur_floor - 1].out_down = 0;
        }
        else {
            if( updn == UP )        elevator[cur_floor - 1].out_up = 0;
            else if( updn == DOWN ) elevator[cur_floor - 1].out_down = 0;
        }

        lcd_gotoxy(cur_floor + 10, 1);
        lcd_data_write(' ');
        led_out();

        door_flag = 0;
        move_flag = 0;
    }
}

if( !warning_flag ) {
    lcd_gotoxy(9, 1);
    lcd_data_write(door_state);
}
}

void weight_check(void)
{
    if( !warning_flag ) {
        //////////////////////////////////////
        // [문제 1] 무게 표시
        // (7) 가변저항(VR1)으로 무게를 입력받는다. 무게가 99kg을 초과할 시 99kg이하로
        // 무게가 내려갈 때까지 “요구사항 (1)”에서 설정한 경보음이 1초 간격으로
        // 울리고, 모든 스위치가 HOLD 되고, 문은 열린 상태를 유지, LCD 두 번째 줄에는
        // 다음과 같이 'OVER WEIGHT!!!'가 출력된다.
        //////////////////////////////////////
        lcd_gotoxy(2, 1);

        if( weight > 99 ) {
            lcd_gotoxy(0, 1);

            warning_flag = 1;
            warning_tick = 0;

        }
    }
    else {

        if( weight <= 99 ) {
            buzzer_out(NONE);

```

```

        init_screen();

        move_tick = 0;
        move_sec = 0;

        warning_flag = 0;
    }
}

/////////////////////////////////////////////////////////////////
// [문제 2] 비상스위치 동작
// (8) 엘리베이터가 이동하는 중에 '비상' SW를 누르면 PC에서 'ESC' 키를 누르기
// 전 까지 모든 스위치가 HOLD 되고 "요구사항 (1)"에서 설정한 경보음이 1초
// 간격으로 울리고 LCD 두 번째 줄에 다음과 같이 출력한다.
/////////////////////////////////////////////////////////////////
void emergency(void)
{

    {
        buzzer_out(NONE);
        init_screen();

        emergency_flag = 0;
    }
}

/////////////////////////////////////////////////////////////////
// [문제 3] Title Set 동작
// (9) 엘리베이터 문이 닫혀 있을 때 'TITLE SET' 스위치를 누르고, PC의 키보드를
// 이용하여 다음과 같이 실행되게 하시오. 단, 'TITLE SET' 스위치를 누르면
// 모든 스위치 입력은 받을 수 없게 되고 5층까지 TITLE 입력 후 다시 작동되게
// 하시오.
/////////////////////////////////////////////////////////////////
void title_set(void)
{
    int i, j;

    serial_transmit('\f');
    serial_string(">>> INPUT FLOOR TITLE\r\n\r\n");

    for( i = 0; i < 5; i++ ) {
        serial_transmit(i + 0x31);
        serial_string("F : ");

        while( 1 ) {
            if( serial_flag ) {

```

```

        }
    }
}

eeprom_write_byte(0, 107);

init_screen();
}

int main(void)
{
    unsigned char key = 0;

    mcu_init();
    variable_init();
    sei();

    fdevopen((void *)lcd_data_write, 0);

    var_start();
    start_elevator();

    init_screen();

    while( 1 ) {
        key = getkey(key);
        if( key_flag ) {
            switch( key ) {
                case KEY_IN1 :
                case KEY_IN2 :
                case KEY_IN3 :
                case KEY_IN4 :
                case KEY_IN5 :
                    if( key != cur_floor ) {
                        elevator[key - 1].in = 1;
                        lcd_gotoxy(key + 10, 1);
                        lcd_data_write(key + '0');
                    }
                    break;

                case KEY_OUT1 :
                case KEY_OUT2 :
                case KEY_OUT3 :
                case KEY_OUT4 :
                case KEY_OUT5 :
                    if( (key - 10) != cur_floor )        out_floor = key;
                    break;

                case KEY_CLOSE :
                    if( (door_state == OPEN) && (door_dir == STOP) ) {
                        move_tick = 0;
                        move_sec = 0;

                        door_dir = CLOSE;
                        dcmotor_spin(RIGHT);
                    }

```

```

case KEY_OPEN :
    if( (door_state == OPEN) && (door_dir == CLOSE) ) {
        dcmotor_spin(STOP);

        move_tick = 250 - move_tick;
        move_sec = 2 - move_sec;

        door_dir = OPEN;
        dcmotor_spin(LEFT);
    }
    break;

case KEY_EMERGENCY :
    lcd_gotoxy(0, 1);
    lcd_string(" EMERGENCY!!! ");

    serial_transmit('\f');
    serial_string(">>> ALERT EMERGENCY!!!!!\r\n");

    emergency_tick = 0;
    emergency_flag = 1;
    break;

case KEY_TITLE :
    if( (door_state == CLOSE) && (door_dir == STOP) ) title_set();
    break;

case KEY_UP :
    if( out_floor ) {
        if( out_floor != KEY_OUT5 )    elevator[out_floor - 11].out_up = 1;
        out_floor = 0;
    }
    led_out();
    break;

case KEY_DOWN :
    if( out_floor ) {
        if( out_floor != KEY_OUT1 )    elevator[out_floor - 11].out_down = 1;
        out_floor = 0;
    }
    led_out();
    break;
}
}
else {
    if( key == KEY_OPEN ) {
        if( (door_state == OPEN) && (door_dir == STOP) ) {
            move_tick = 0;
            move_sec = 0;
        }
    }
}
}

```

```

floor_check();
if( cur_floor != target_floor )    move_elevator();
if( door_flag )    move_door();
if( (door_state == OPEN) && (door_dir == STOP) )    weight_check();

```

```

if( (door_state == OPEN) && (door_dir == CLOSE) ) {
    if( photo_check() ) {
        dcmotor_spin(STOP);

        move_tick = 250 - move_tick;
        move_sec = 2 - move_sec;

        door_dir = OPEN;
        dcmotor_spin(LEFT);
    }
}
if( emergency_flag )emergency();

fnd_out(cur_floor);
}

return 0;
}

```