2019 한국폴리텍대학 (다기능과정)
IT융합전자회로 설계 및 제작 경진대회

제 3 과제

Project Name : MICOM PROGRAMMING

제한 시간 : 5시간

평생기술로 행복일터를 KOREA POLYTECHNICS 한국폴리텍대학

후원 : 학교법인 한국폴리텍대학

협찬 : 한국폴리텍대학 광주캠퍼스,나인플러스아이티(주)

IT융합전자회로 설계 및 제작 경진대회 과제

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

1. 요구사항

가. 지급 된 프로그램설계 회로기판, 회로도, 동작 개요를 참조하여 제작하시오.

나. 배포되는 소스는 프로그램 요구조건 (1)~(8)까지 동작되게 한 것이다.
이 소스를 컴파일, 라이팅하여 프로그램 요구조건 (1)~(8)까지 동작시키시오.
※ 단, 자신의 컴파일과 맞지 않는 부분은 수정하여 동작시키시오.

다. 배포된 소스 를 참조하여 프로그램 요구조건 [문제1]~[문제3]을 프로그램하고, 컴파일, 라이팅하여 동작시킨 후 자신의 비번호를 적고, 저장하시오.
C:\ BAPOXX.C , C:\ BAPOXX.asm (XX : 자신의 비번호)

라. 본 과제는 전자통장의 기능을 구현한 것으로 아래의 기능을 한다.
(1) 사용자 설정 모드 기능
(2) 전자통장 기능 등

마. 동작요구사항
※ 아래 요구사항 (1)~ (8)까지는 배포되는 소스를 이용하여 동작시키시오.
(1) RESET 또는 전원 인가시 FND는 OFF, LCD는 다음과 같이 동작되게 하시오

	D	i	g	i	t		B	a	n	k	b	o	o	k	
	I	n	i	t	.	.	.								

↓ (3초후)

1	.	C	a	s	h		S	e	r	v	i	c	e		
2	.	U	s	e	r		S	e	t	t	i	n	g		

(2) "요구사항 (1)"에서 2번키를 누르면 USER SETTING MODE가 실행되게 하시오.
(가) 0~9번 키를 이용하여 4자리의 User Number를 설정하시오. 4자리 입력

후, 다시 숫자키를 누르면 처음부터 입력되게 하고(LCD화면은 User Number 입력 초기 상태), Cancel 키를 누르면 숫자 하나가 지워지게 하시오. FND에는 입력한 숫자가 표시되도록 하시오. 4자리 입력 후 ENTER 키를 눌러 다음으로 넘어가게 하시오.

		U	s	e	r		N	u	m	b	e	r		
						_	_	_	_					

(나) 0~9번 키를 이용하여 4자리의 Password를 설정하시오. 4자리 입력 후, 다시 숫자키를 누르면 처음부터 입력되게 하고(LCD화면은 Password 초기 입력 상태), Cancel키를 누르면 숫자 하나가 지워지게 하시오. FND는 OFF되며, 4자리 입력 후 ENTER 키를 눌러 다음으로 넘어가게 하시오.

				P	a	s	s	w	o	r	d			
						_	_	_	_					

(다) FND는 OFF되며, Password가 맞는지 다시 한 번 입력하여 확인하시오.

	C	h	e	c	k		P	a	s	s	w	o	r	d
						_	_	_	_					

(예) 입력한 Password가 맞을 경우,

		U	s	e	r		N	u	m	b	e	r		
						X	X	X	X					

↓ (3초후)

1	.	C	a	s	h		S	e	r	v	i	c	e	
2	.	U	s	e	r		S	e	t	t	i	n	g	

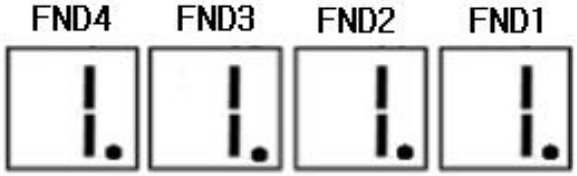
(예) 입력한 Password가 틀릴 경우,

	E	r	r	o	r	!	!	!	!					
	P	a	s	s	w	o	r	d		E	r	r	o	r

↓ LCD 첫 줄의 "Error!!!!" 가 1초 간격으로 3번 깜빡인 후

	C	h	e	c	k		P	a	s	s	w	o	r	d
						_	_	_	_					

(3) “요구사항 (2)”에서 Password, Check Password를 입력할 때, 키를 누를 때 마다 LCD에는 “-” 표시가 나타나고 FND에는 DOT가 표시되게 하시오. FND에는 User Number가 계속 표시되게 하시오. Cancel키를 누르면 LCD는 숫자가 지워지고, FND에는 DOT가 꺼지게 하시오.



(User Number가 1111 일 때, Password 4자리 입력시)

[문제 1] Cash Service Mode의 ID Check

(4) “요구사항 (1)”에서 1번키를 누르면 Cash Service Mode가 실행되게 하시오. 설정한 User Number와 Password를 입력하시오.

	U	s	e	r		N	u	m	:		_	_	_	_
	P	a	s	s	w	o	r	d	:		_	_	_	_

User Number를 입력할 때는, FND에 입력한 숫자가 표시되게 하고, Password를 입력할 때에는 입력한 User Number가 계속 표시되게 하시오.

(가) User Number와 Password가 틀릴 경우.

	E	r	r	o	r	!	!	!	!					
	I	D		C	h	e	c	k		E	r	r	o	r

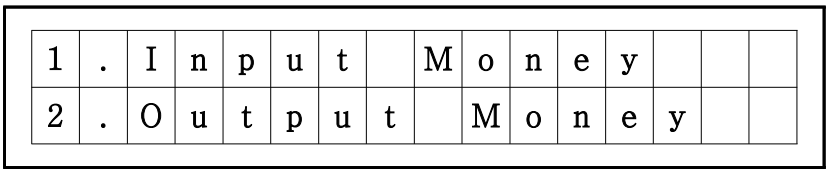
↓ LCD 첫 줄의 "Error!!!!" 가 1초 간격으로 3번 깜박인 후

	U	s	e	r		N	u	m	:		_	_	_	_
	P	a	s	s	w	o	r	d	:		_	_	_	_

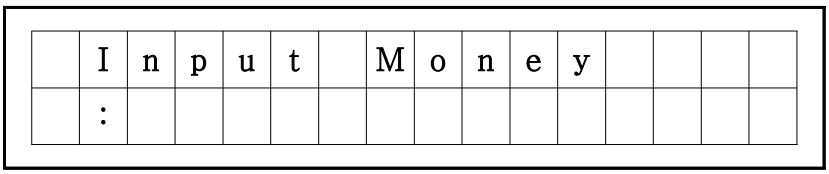
(나) User Number와 Password가 맞을 경우.

1	.	I	n	p	u	t	/	O	u	t	p	u	t	
2	.	A	c	c	o	u	n	t		C	h	e	c	k

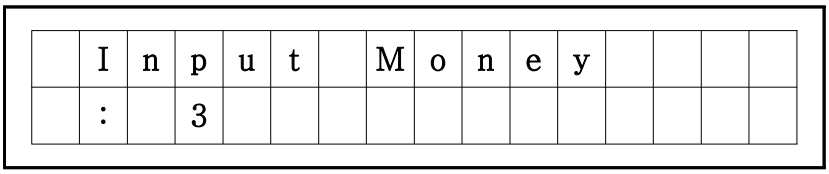
(5) “요구사항 (3)-(나)”에서 1번키를 누르면 입금/출금 모드가 실행되게 하시오.
입력은 ENTER키가 아닌 ‘만’, ‘원’ 키를 눌러서 입력하게 하시오.
※ 입금/출금은 한 번에 최대 99만원까지, 최대 잔고는 999만원까지 되게 하시오.



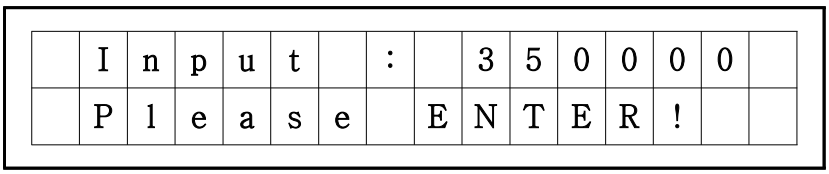
(가) 1번 키를 누르면 입금모드가 실행되게 하시오.



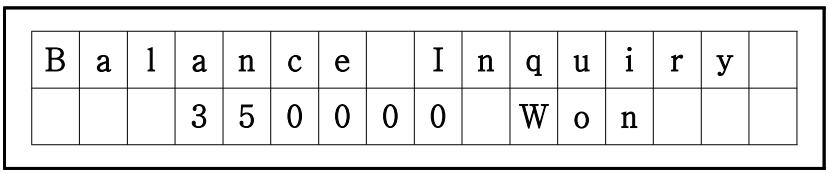
↓ 3번 키를 누르면,



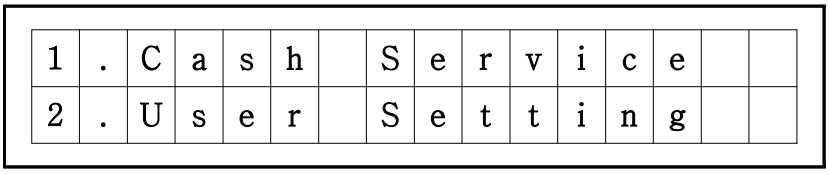
↓ 35를 입력하고 만 키를 누르면,



↓ ENTER 키를 누르면, 현재 잔고가 표시되게 하시오.



↓ 3초 후



(나) 입금시 최대 잔고가 999만원을 초과할 경우 다음과 같이 되게 하시오.

	I	n	p	u	t		:		3	5	0	0	0	
	P	l	e	a	s	e		E	n	t	e	r	!	

↓ 최대 잔고를 초과할 경우

	E	r	r	o	r	!	!	!	!					
	B	a	l	a	n	c	e		E	x	c	e	s	s

↓ LCD 첫 줄의 "Error!!!!" 가 1초 간격으로 3번 깜박인 후

	I	n	p	u	t		M	o	n	e	y			
	:													

입금모드가 다시 실행되게 하시오.

(다) 2번 키를 누르면 출금모드가 실행되게 하시오.

	O	u	t	p	u	t		M	o	n	e	y		
	:													

↓ 3번 키를 누르면,

	O	u	t	p	u	t		M	o	n	e	y		
	:		3											

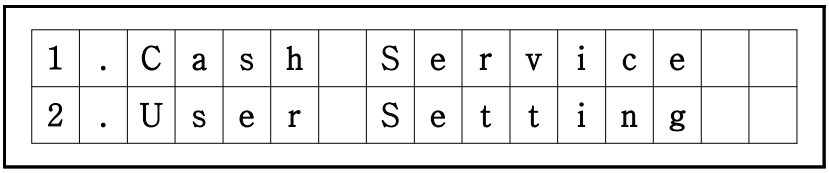
↓ 35를 입력하고 만 키를 누른 후,

	O	u	t	p	u	t		:		3	5	0	0	0
	P	l	e	a	s	e		E	n	t	e	r	!	

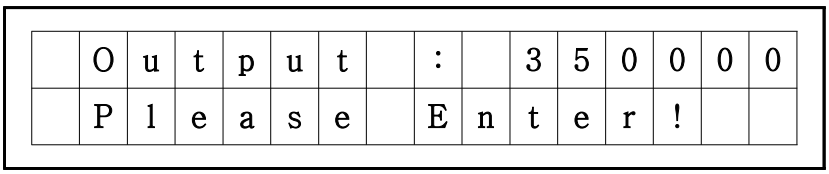
ENTER 키를 누르면, 현재 잔고가 표시되게 하시오.

B	a	l	a	n	c	e		I	n	q	u	i	r	y
			3	5	0	0	0	0		W	o	n		

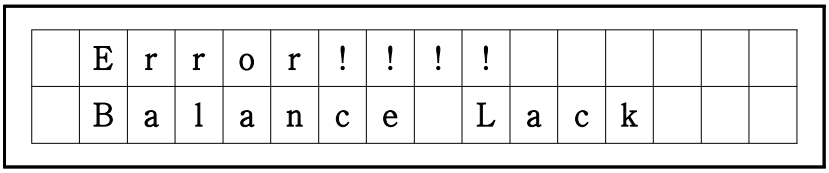
↓ 3초 후



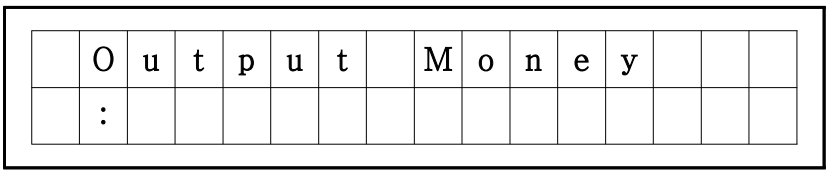
(라) 출금 금액이 현재 잔액보다 많을 경우 다음과 같이 실행되게 하시오.



↓ 잔액이 부족할 경우



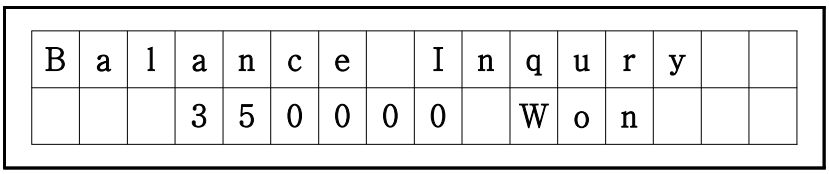
↓ LCD 첫 줄의 "Error!!!!" 가 1초 간격으로 3번 깜박인 후



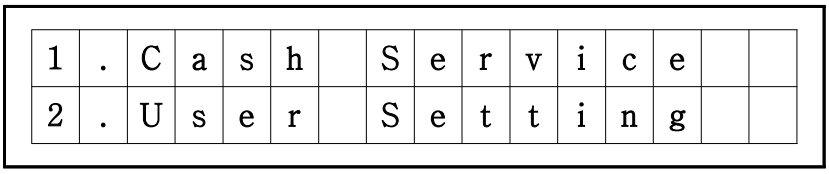
출금모드가 다시 실행되게 하시오.

(마) 입금/출금 금액 입력시 Cancel 키를 누르면 처음부터 금액을 새로이 입력하게 하시오.

(6) “요구사항 (3)-(나)”에서 2번키를 누르면 계좌조회 모드가 실행되게 하시오.

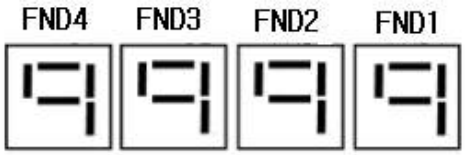


↓ 3초 후

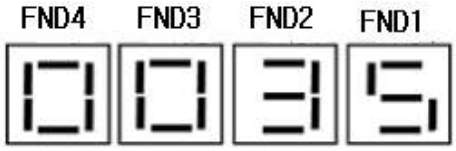


[문제 2] FND에 현재 잔고 표시

(7) “요구사항 (5)”실행 시, FND는 OFF 되어 있다가 현재 잔고가 표시될 때 1만원 미만이면 그대로 표시하고 1만원 이상 일 때는 만원 단위로 표시되게 하시오.



(9999원 일 때)



(35만원 일 때)

[문제 3] 새로운 User Number 설정

(8) "요구사항 (2)"에서 User Setting을 실행 완료 후, 다시 실행할 경우 다음과 같이 새로운 USER NUMBER로 설정되게 하시오.

(가) B.I(Balance Inquiry)가 표시되고, Cancel 키를 누르면 "요구사항 (1)"로 돌아가게 하시오. Enter 키를 누르면 다음으로 동작이 실행되게 하시오.

B	.	I	:		x	x	x	x	x	x	x	W	o	n
N	e	w		U	s	e	r	S	e	t	t	i	n	g	?

↓ ENTER 키를 눌렀을 때

		U	s	e	r		N	u	m	b	e	r			
						_	_	_	_						

(나) User Number와 Password는 바뀌어도 Balance Inquiry(잔액)은 기존의 금액을 유지하게 하시오.

바. 배포된 PCB에 오류가 있으면 수정하여 동작시키시오.

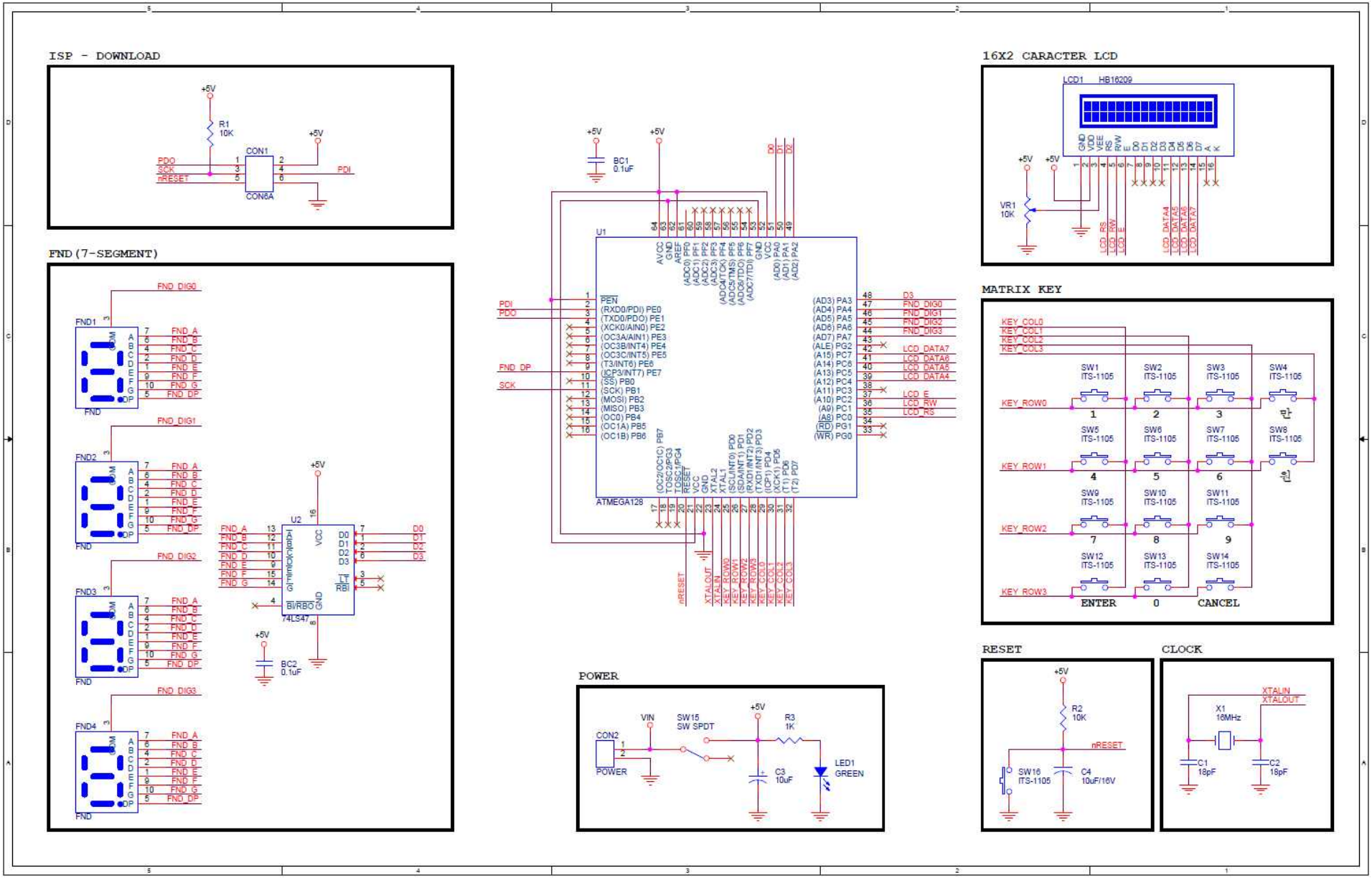
사. 작업이 완료되면 심사 위원에게 동작 검사를 받으시오.

2. 유의사항

가. 안전사고에 유의하시오.

나. 심사위원의 지시에 따라 작업을 진행하시오.

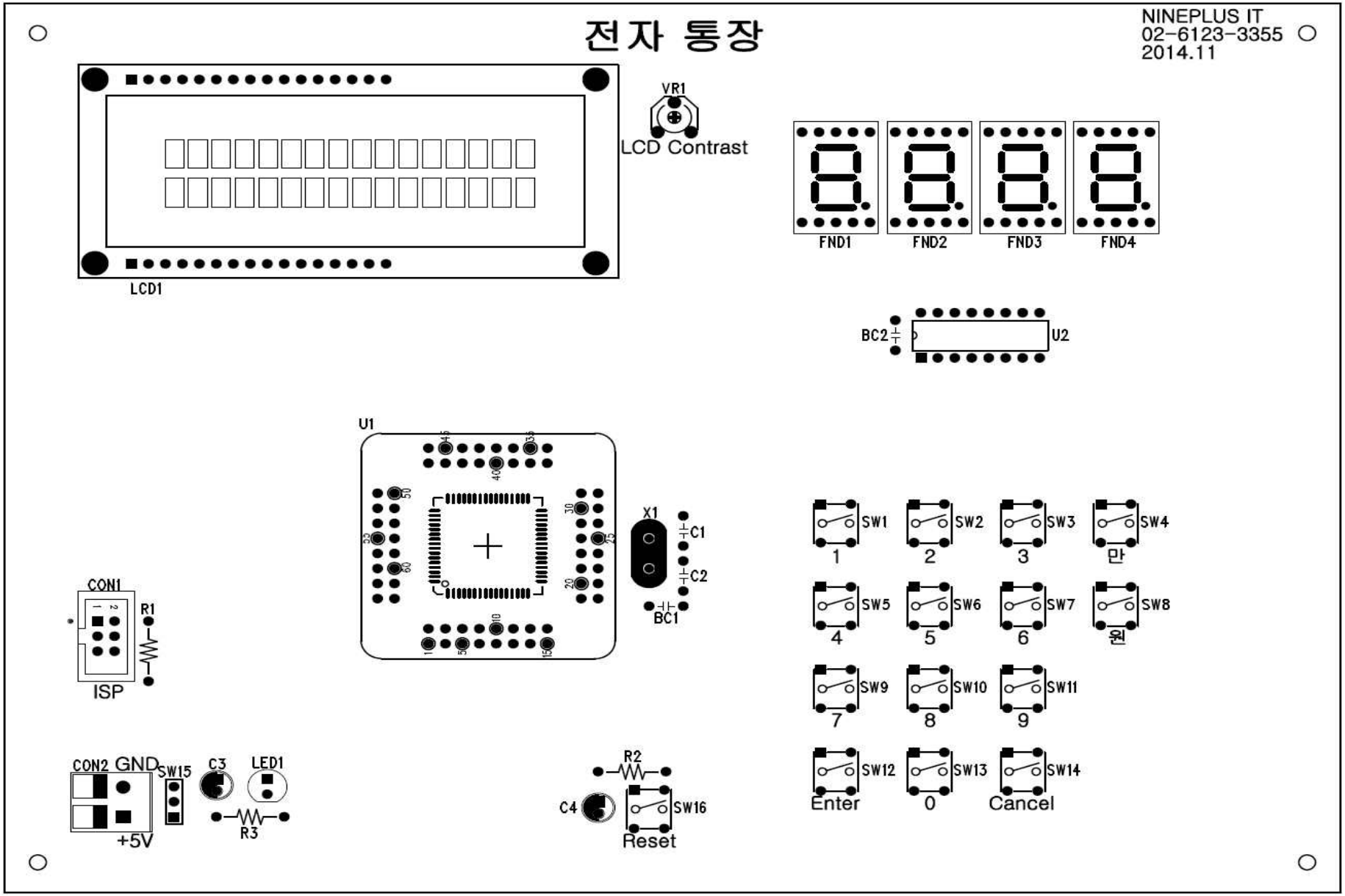
3. 회로도


ISP - DOWNLOAD
FND (7-SEGMENT)
16X2 CARACTER LCD
MATRIX KEY
RESET
CLOCK
POWER
ATMEGA128
74LS47
ENTER
CANCEL

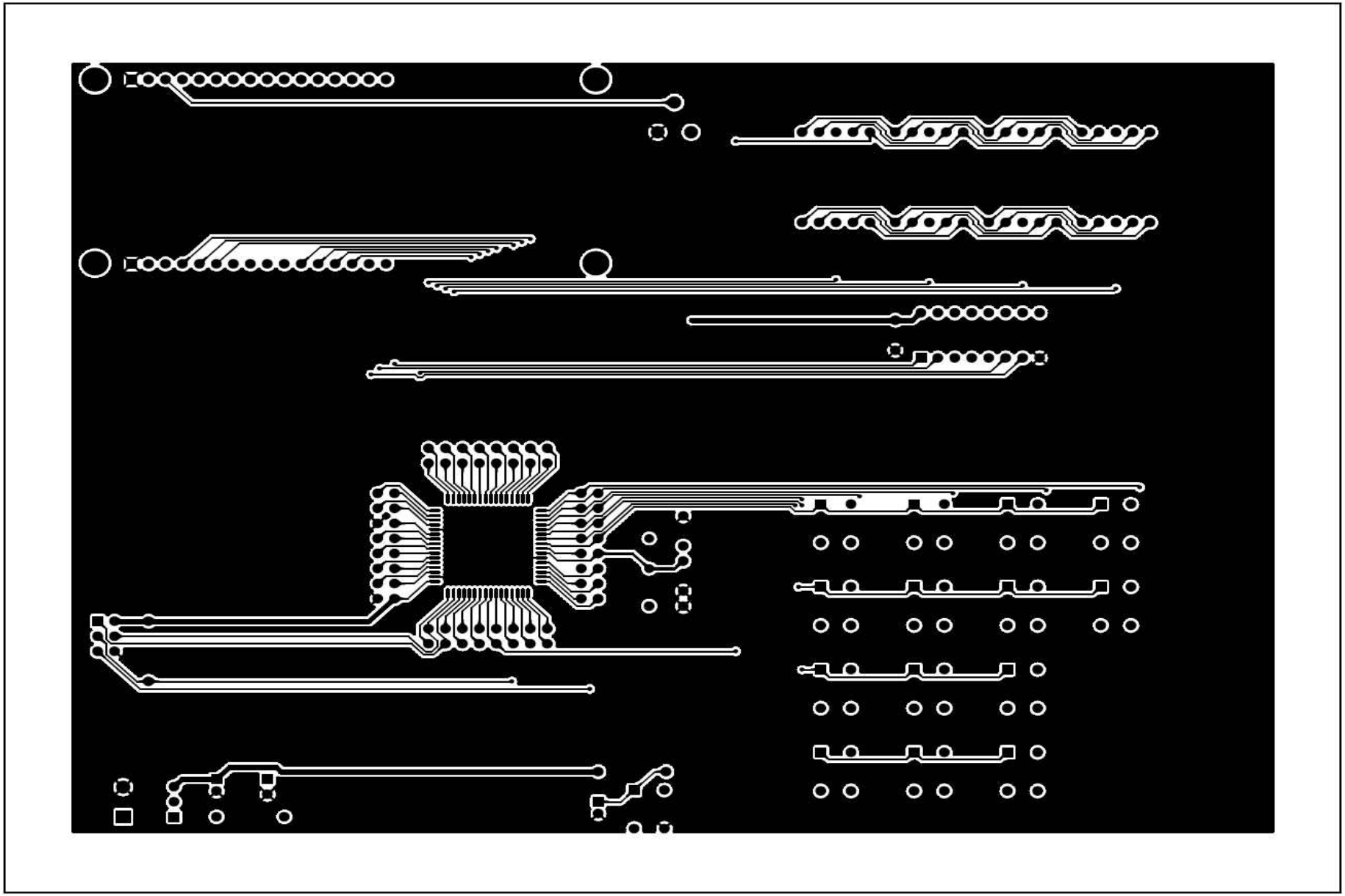
4. 지급 재료 목록

일련번호	재 료 명	규 격(치수)	단위	수량	비 고
1	IC	ATMEGA128	개	1	
2	CPU Module Connector	Dual Connector(8x2, 수)	개	4	
3	CPU Module Connector	Dual Connector(8x2, 암)	개	4	
4	ISP Connector	Dual Header 3x2, 수	개	1	
5	Power Connector	녹색단자 2pin, 5.0mm	개	1	
6	Slide Switch	MSL-1C2P	개	1	
7	Crystal	16MHz	개	1	
8	세라믹 Capacitor	0.1uF	개	2	
9	세라믹 Capacitor	18pF	개	2	
10	전해 Capacitor	10uF/16V	개	2	
11	저항	1KΩ	개	1	
12	저항	10KΩ	개	2	
13	가변저항	10KΩ	개	1	
14	LED	Green(5φ)	개	1	
15	FND	FND 507	개	4	
16	LCD	LCD1602 5V	개	1	
17	LCD Connector	Single Connector(14pin, 수)	개	1	
18	LCD Connector	Single Connector(14pin, 암)	개	1	
19	IC	74LS47	개	1	
20	Tact Switch	Tact Switch	개	15	
21	서포트	3φ x 40mm F	개	4	
22	서포트	3φ x 5mm M	개	4	
23	PCB	Bare PCB	장	1	
24	PCB2	CPU Module PCB	장	1	
25					

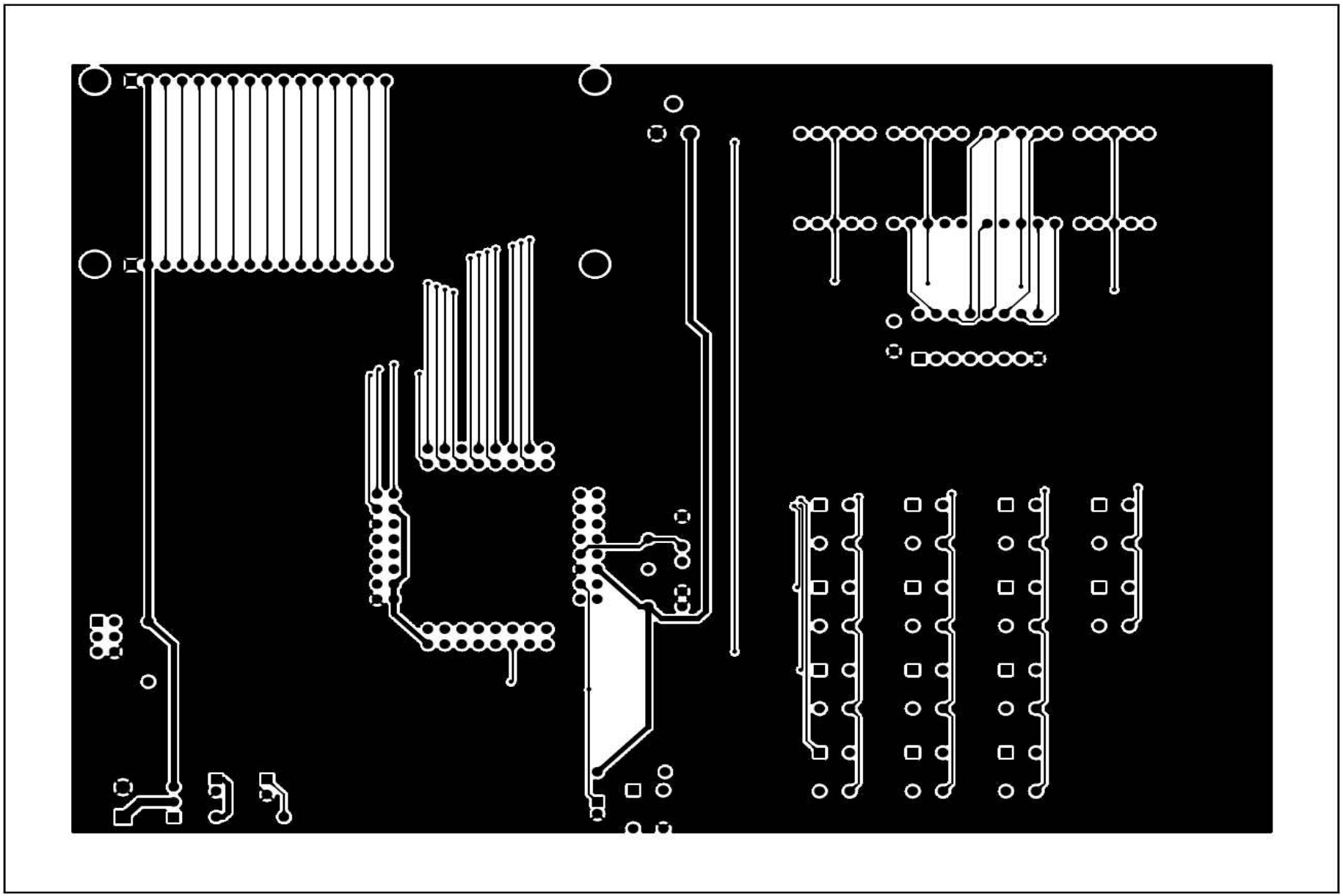
5-1. PCB(부품면)


전자 통장
NINEPLUS IT
02-6123-3355
2014.11
VR1
LCD Contrast
LCD1
FND1
FND2
FND3
FND4
BC2
U2
U1
X1
C1
C2
BC1
CON1
R1
ISP
SW1
SW2
SW3
SW4
1
2
3
만
SW5
SW6
SW7
SW8
4
5
6
원
SW9
SW10
SW11
7
8
9
SW12
SW13
SW14
Enter
0
Cancel
CON2
GND
SW15
C3
LED1
R3
+5V
R2
C4
SW16
Reset

5-2. PCB(TOP면)



5-3. PCB(BOTTOM면)



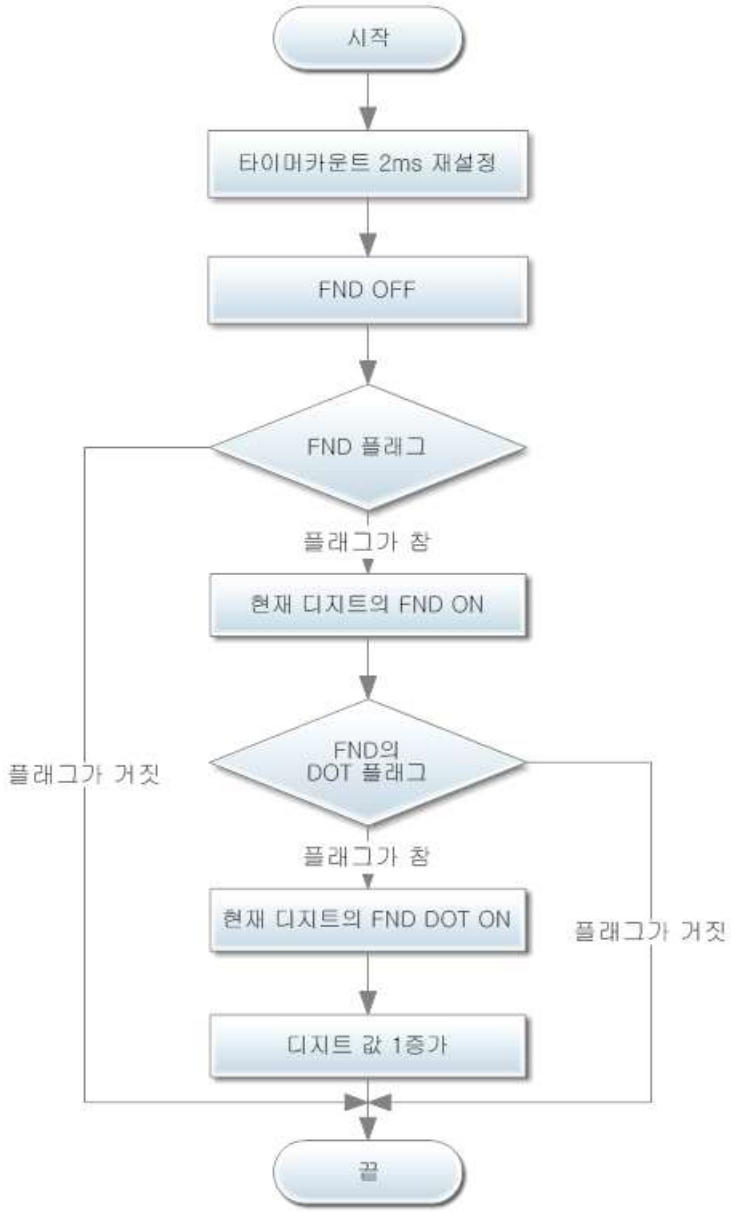
6. 배포용 소스

6-1. 주변장치 동작 소스

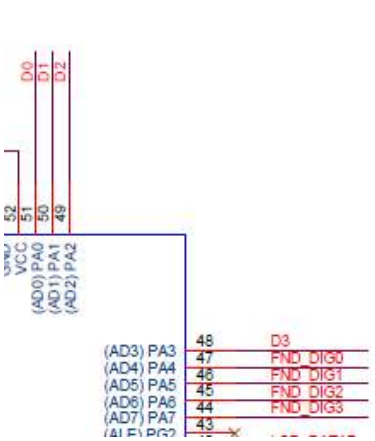
가. FND

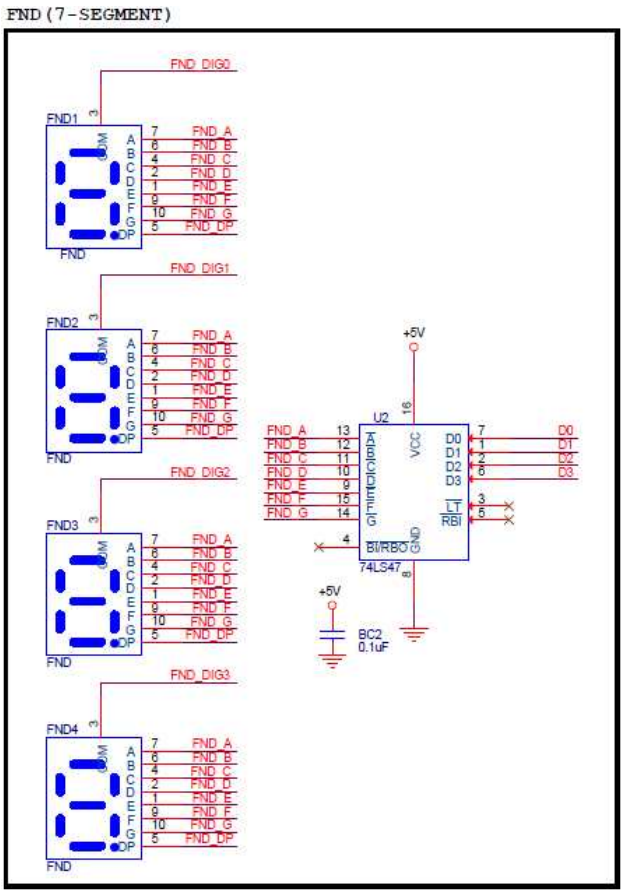
(1) 동작흐름도

타이머 인터럽트를 이용한 4digit FND의 다이나믹 구동 흐름도



(2) 회로도 및 사진





(3) C Source : fnd.c

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

#include "fnd.h"

#define FND_OUT    PORTA
#define FND_DDR    DDRA
#define DOT_OUT    PORTE
#define DOT_DDR    DDRE
#define DOT        0x80

volatile unsigned char fnd_buf[4], dot_buf[4], digit;
volatile char fnd_flag, dot_flag;


ISR(TIMER2_OVF_vect)
{
    TCNT2 = 256 - 125;

    FND_OUT = 0x00;
    DOT_OUT |= 0x80;

    if( fnd_flag ) {
        FND_OUT = fnd_buf[digit];
        FND_OUT |= 0x10 << digit;
    }
```

```
        if( dot_flag ) {
            if( dot_buf[digit] )   DOT_OUT &= ~DOT;
        }

        digit = ++digit % 4;
    }

    void timer2_init(void)
    {
        TCCR2 = 0x04;
        TCNT2 = 256 - 125;
        TIMSK |= 0x40;
    }

    void fnd_init(void)
    {
        FND_DDR = 0xFF;
        DOT_DDR |= 0x80;

        for( int i = 0; i < 4; i++ ) {
            fnd_buf[i] = 0x0F;
            dot_buf[i] = 0;
        }
        fnd_flag = 0;
        dot_flag = 0;
        digit = 0;

        timer2_init();
    }
```

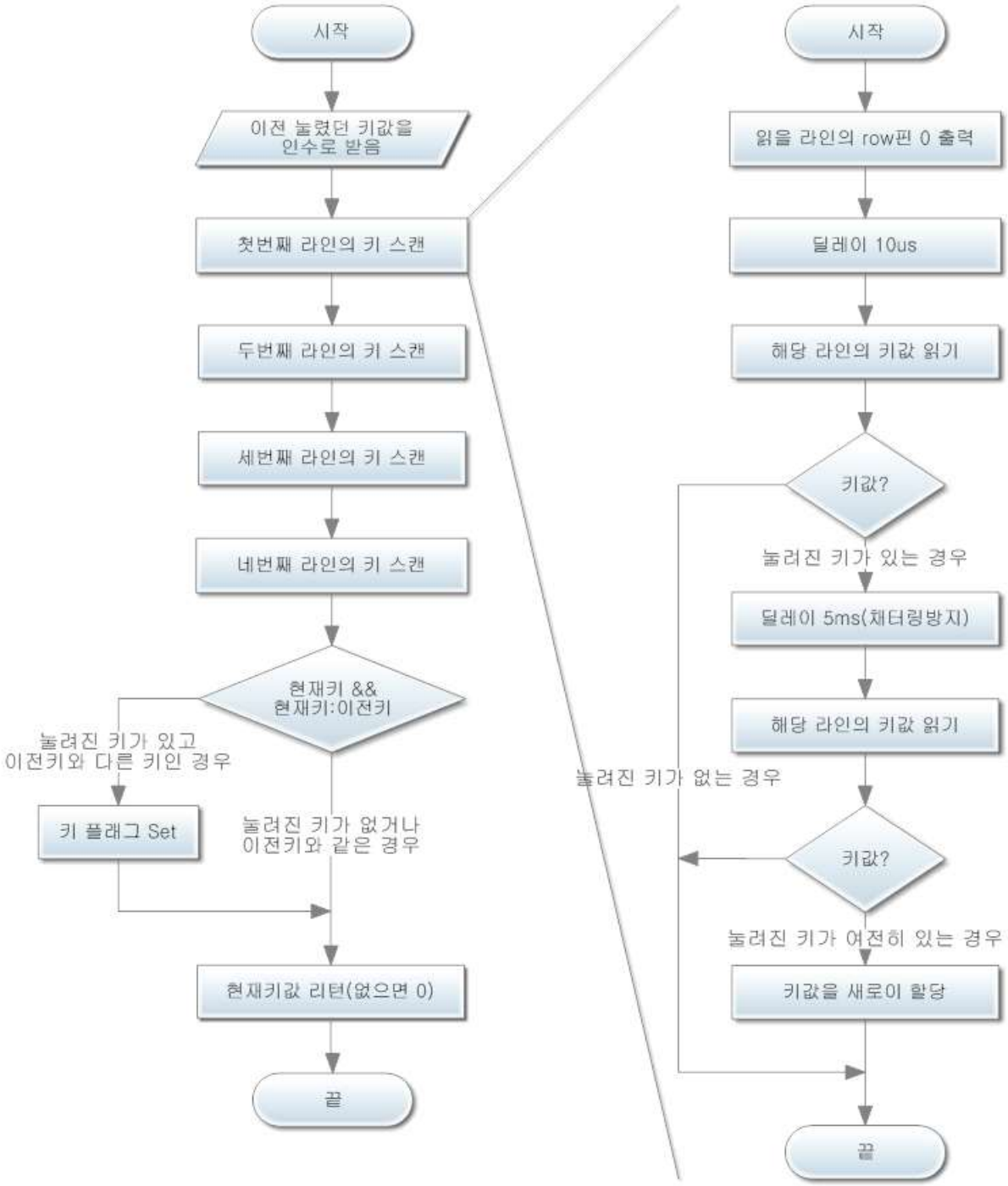
(4) Header : fnd.h

```
    #ifndef  __FND_H
    #define  __FND_H

    void timer2_init(void);
    void fnd_init(void);

    #endif   // __KEY_H
```

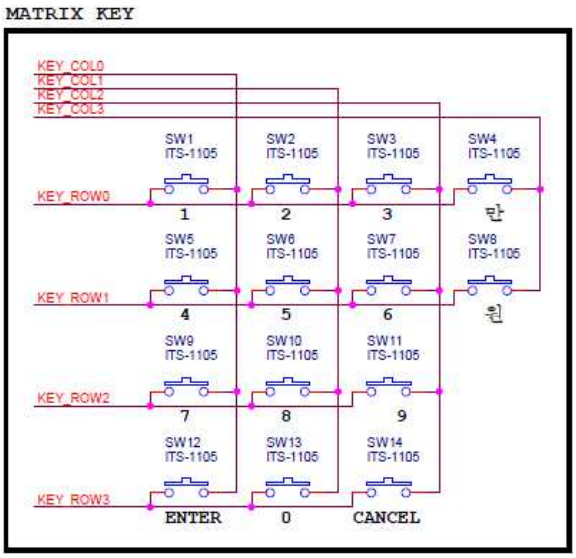
나. Matrix Key

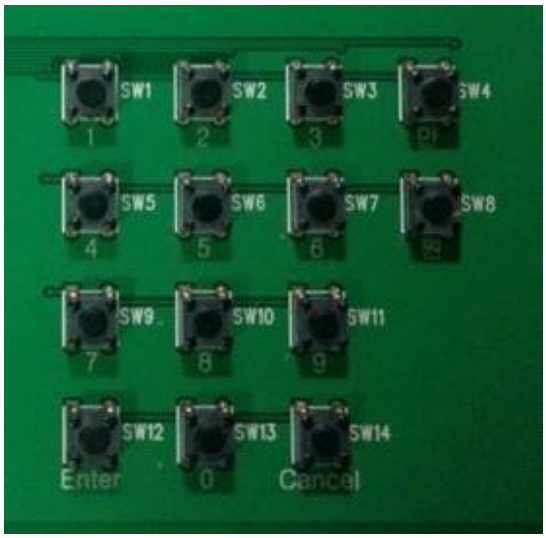
(1) 동작흐름도

unsigned char getkey(unsigned char keyin) 함수의 흐름도



(2) 회로도 및 사진





(3) C Source : key.c

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

#include "key.h"

#define KEY_OUT   PORTD
#define KEY_IN PIND
#define KEY_DDR   DDRD

volatile char key_flag;

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

    key_flag = 0;

    KEY_OUT = ~0x01;
    _delay_us(10);
    key = ~KEY_IN & 0xF0;
    if( key ) {
        _delay_ms(5);
        key = ~KEY_IN & 0xF0;
        if( key ) {
            if( key == 0x10 )       key = KEY_1;
            else if( key == 0x20 )  key = KEY_2;
            else if( key == 0x40 )  key = KEY_3;
            else if( key == 0x80 )  key = KEY_MAN;
```

```
        }
    }
    else {
        KEY_OUT = ~0x02;
        _delay_us(10);
        key = ~KEY_IN & 0xF0;
        if( key ) {
            _delay_ms(5);
            key = ~KEY_IN & 0xF0;
            if( key ) {
                if( key == 0x10 )       key = KEY_4;
                else if( key == 0x20 )  key = KEY_5;
                else if( key == 0x40 )  key = KEY_6;
                else if( key == 0x80 )  key = KEY_WON;
            }
        }
        else {
            KEY_OUT = ~0x04;
            _delay_us(10);
            key = ~KEY_IN & 0xF0;
            if( key ) {
                _delay_ms(5);
                key = ~KEY_IN & 0xF0;
                if( key ) {
                    if( key == 0x10 )       key = KEY_7;
                    else if( key == 0x20 )  key = KEY_8;
                    else if( key == 0x40 )  key = KEY_9;
                }
            }
            else {
                KEY_OUT = ~0x08;
                _delay_us(10);
                key = ~KEY_IN & 0xF0;
                if( key ) {
                    _delay_ms(5);
                    key = ~KEY_IN & 0xF0;
                    if( key ) {
                        if( key == 0x10 )       key = KEY_ENTER;
                        else if( key == 0x20 )  key = KEY_0;
                        else if( key == 0x40 )  key = KEY_CANCEL;
                    }
                }
            }
        }
    }

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

    return key;
}

void key_init(void)
{
    KEY_OUT = 0xF0;
    KEY_DDR = 0x0F;
}
```

(4) Header : key.h

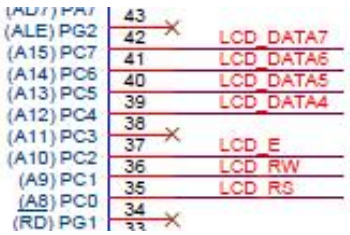
```
#ifndef  __KEY_H
#define  __KEY_H

#define KEY_1          '1'
#define KEY_2          '2'
#define KEY_3          '3'
#define KEY_4          '4'
#define KEY_5          '5'
#define KEY_6          '6'
#define KEY_7          '7'
#define KEY_8          '8'
#define KEY_9          '9'
#define KEY_0          '0'
#define KEY_MAN        1
#define KEY_WON        2
#define KEY_ENTER      3
#define KEY_CANCEL     4

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

#endif   // __KEY_H
```

다. Character LCD 16x2

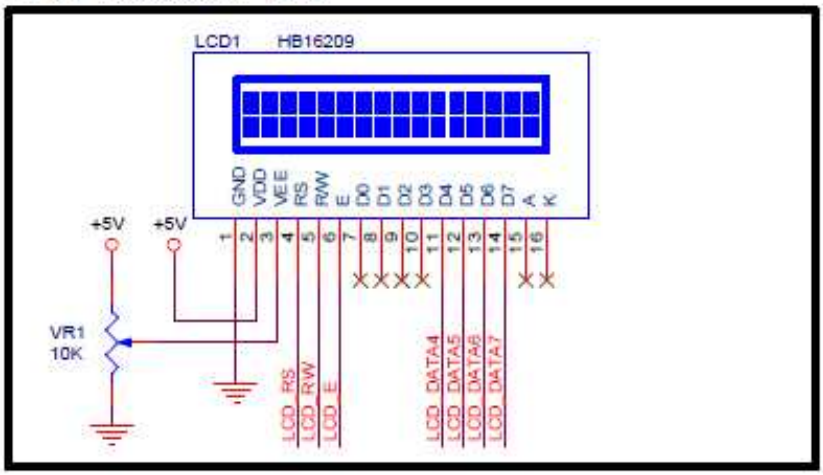
(1) 동작 흐름도

LCD Data 및 명령어를 4비트 인터페이스로 읽고 쓰는 함수의 흐름도

(2) 회로도 및 사진



16X2 CARACTER LCD



(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_busycheck(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_busycheck();

    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_busycheck();

    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

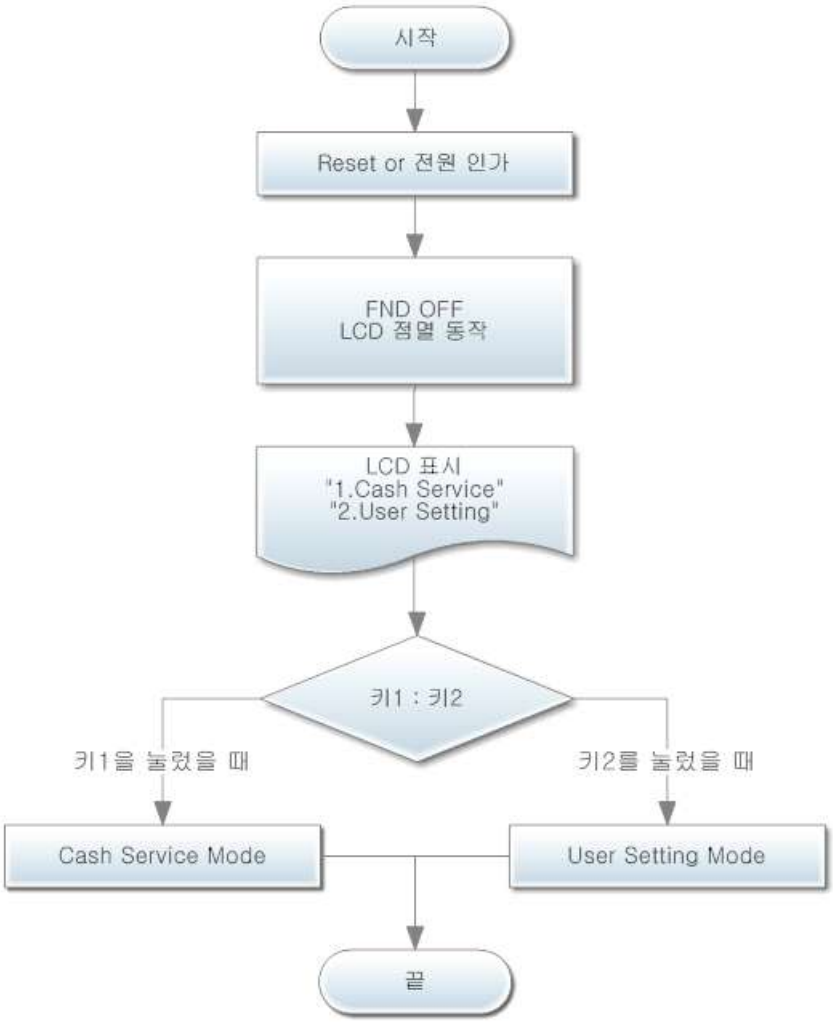
```
#ifndef  __LCD_H
```

```
#define  __LCD_H

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

void lcd_busycheck(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
```

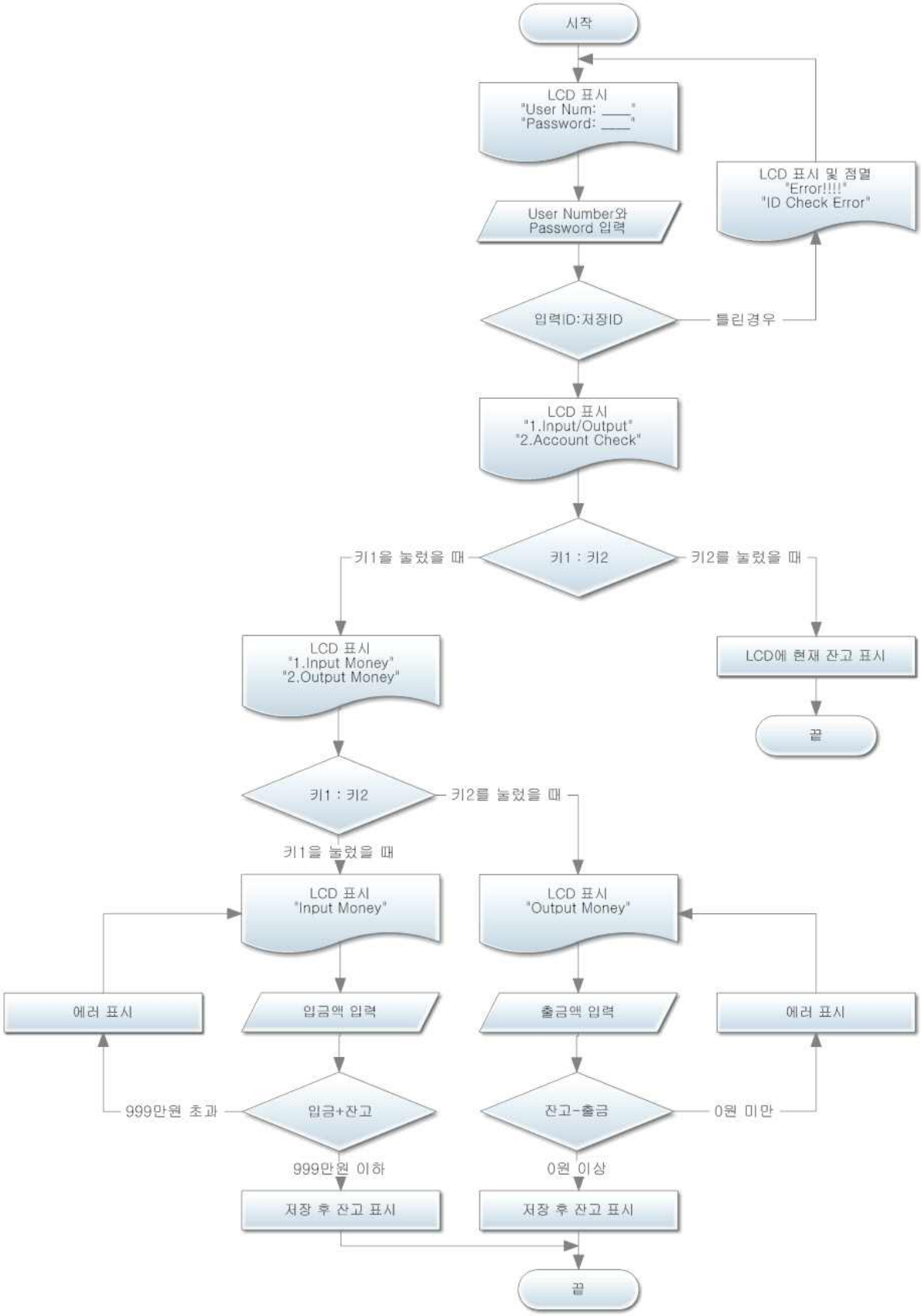
6-2. 전체 동작(Main) 소스

가. 동작 흐름도

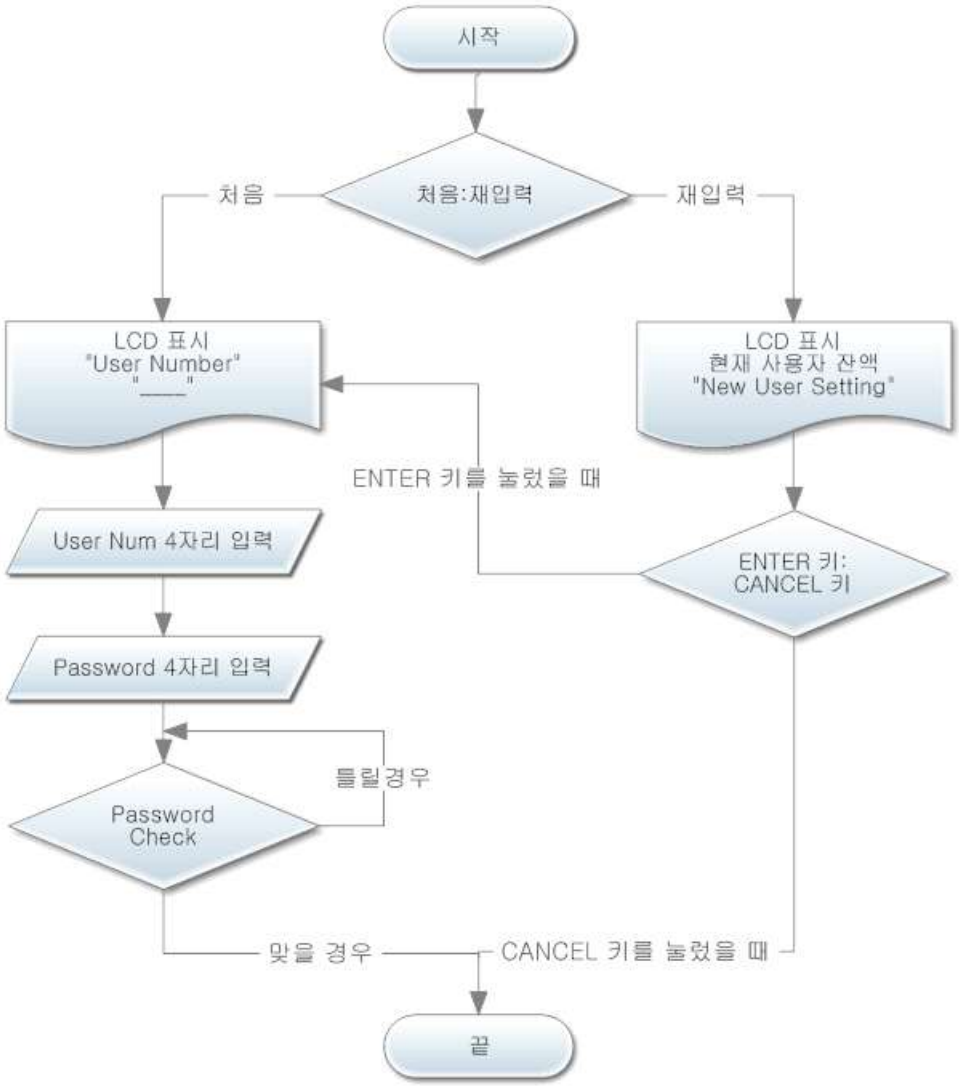
(1) 전체 흐름도



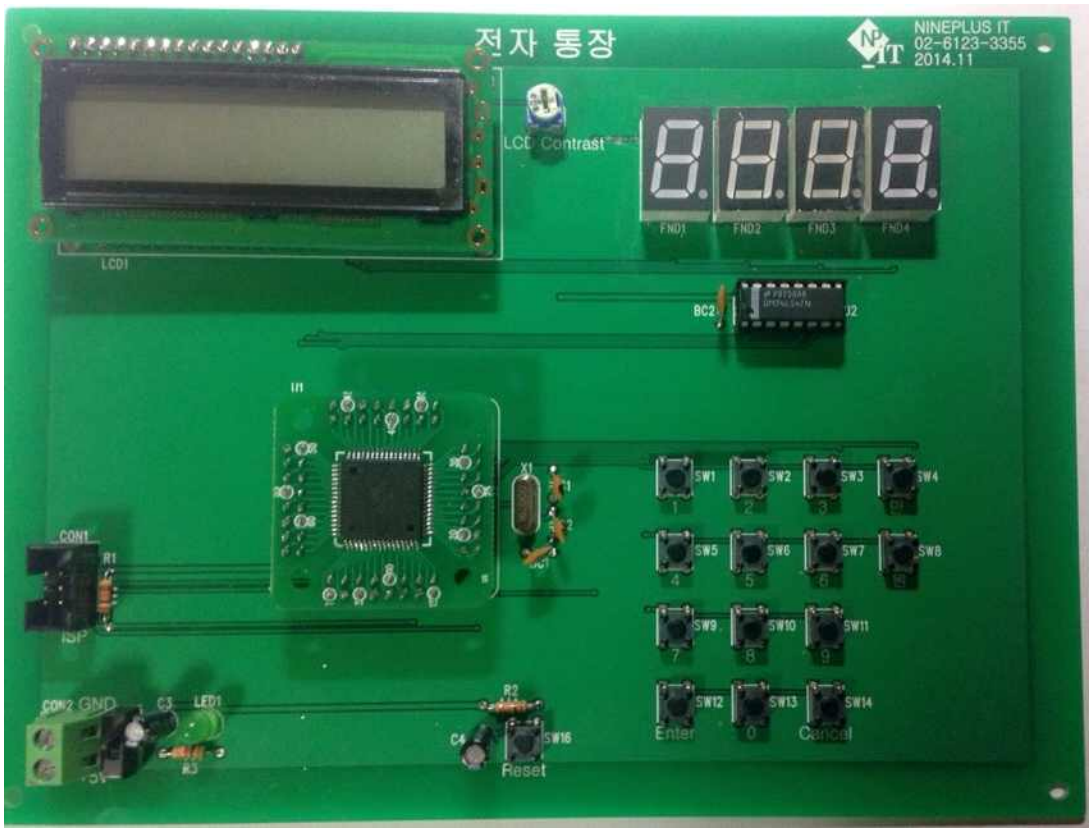
(2) Cash Service Mode 흐름도



(3) User Setting Mode 흐름도



나. 전체사진



다. Source : main.c

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

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

#include "lcd.h"
#include "key.h"
#include "fnd.h"

extern volatile char key_flag;
extern volatile unsigned char fnd_buf[4], dot_buf[4];
extern volatile char fnd_flag, dot_flag;

char user_number[4], password[4];
volatile unsigned char key = 0, user_set;
volatile long save_money;

void mcu_init(void)
{
    lcd_init();
    key_init();
    fnd_init();

    sei();
}
```

```
void start_bankbook(void)
{
    fnd_flag = dot_flag = 0;

    lcd_gotoxy(0, 0);
    lcd_string(" Digit Bankbook ");
    lcd_gotoxy(0, 1);
    lcd_string(" Init...          ");

    _delay_ms(3000);

    user_set = 0;
    save_money = 0;
}

void init_screen(void)
{
    lcd_gotoxy(0, 0);
    lcd_string("1.Cash Service  ");
    lcd_gotoxy(0, 1);
    lcd_string("2.User Setting  ");
}

enum {
    USER_MODE = 1,
    PASS_MODE,
    CHECK_MODE,
    CASH_USER,
    CASH_PASS
};

void input_number(char x, char y, char *str, char mode)
{
    unsigned char pos;

    for( int i = 0; i < 4; i++ ) {
        if( dot_flag )    dot_buf[i] = 0;
        if( (mode == USER_MODE) || (mode == CASH_USER) )fnd_buf[i] = 0x0F;
        str[i] = 0x00;
    }

    pos = 0;

    while( 1 ) {
        key = getkey(key);
        if( key_flag ) {
            switch( key ) {
                case KEY_0 :
                case KEY_1 :
                case KEY_2 :
                case KEY_3 :
                case KEY_4 :
                case KEY_5 :
                case KEY_6 :
                case KEY_7 :
                case KEY_8 :
```

```
                case KEY_9 :
                    if( pos == 4 ) {
                        for( int i = 0; i < 4; i++ ) {
                            if( (mode == USER_MODE) || (mode == CASH_USER) )
                                fnd_buf[i] = 0x0F;
                            else      dot_buf[i] = 0;
                            str[i] = 0x0F;
                        }

                        pos = 0;

                        lcd_gotoxy(pos + x, y);
                        lcd_string("____");
                    }

                    if( pos < 4 ) {
                        if( (mode == USER_MODE) || (mode == CASH_USER) )
                                fnd_buf[pos] = key - '0';
                        else      dot_buf[pos] = 1;
                        str[pos] = key;

                        lcd_gotoxy(pos + x, y);
                        if( (mode == PASS_MODE) || (mode == CHECK_MODE) )
                                lcd_data_write('-');
                        else      lcd_data_write(key);

                        pos++;

                        if( (pos == 4) && (mode >= CHECK_MODE) ) return;
                    }
                    break;

                case KEY_ENTER :
                    return;

                case KEY_CANCEL :
                    pos--;
                    if( (mode == USER_MODE) || (mode == CASH_USER) )
                            fnd_buf[pos] = 0x0F;
                    else      dot_buf[pos] = 0;
                    str[pos] = 0x00;

                    lcd_gotoxy(pos + x, y);
                    lcd_data_write('_');
                    break;
            }
        }
    }
}

void user_setting(void)
{
    char tbuf[4], STATE;
    int i;

    fnd_flag = 1;
```

```
    lcd_gotoxy(0, 0);
    lcd_string("  User Number    ");
    lcd_gotoxy(0, 1);
    lcd_string("        ____        ");
    input_number(6, 1, user_number, USER_MODE);

    dot_flag = 1;
    lcd_gotoxy(0, 0);
    lcd_string("     Password     ");
    lcd_gotoxy(0, 1);
    lcd_string("        ____        ");
    input_number(6, 1, password, PASS_MODE);

    STATE = 1;
    do {
        lcd_gotoxy(0, 0);
        lcd_string(" Check Password ");
        lcd_gotoxy(0, 1);
        lcd_string("        ____        ");
        input_number(6, 1, tbuf, CHECK_MODE);

        for( i = 0; i < 4; i++ ) {
            if( tbuf[i] != password[i] ) {
                lcd_gotoxy(0, 1);
                lcd_string(" Password Error ");

                for( int j = 0; j < 3; j++ ) {
                    lcd_gotoxy(0, 0);
                    lcd_string(" Error!!!!        ");
                    _delay_ms(500);
                    lcd_gotoxy(0, 0);
                    lcd_string("                  ");
                    _delay_ms(500);
                }

                break;
            }
        }

        if( i == 4 ) {
            lcd_gotoxy(0, 0);
            lcd_string("  User Number    ");
            lcd_gotoxy(0, 1);
            lcd_string("        XXXX        ");
            user_set = 1;

            _delay_ms(3000);

            STATE = 0;
        }
    } while( STATE );

    fnd_flag = dot_flag = 0;
}

void new_user_setting(void)
```

```
{
////////////////////////////////////////////////////////////////////////////
// [문제 3] 새로운 User Number 설정
////////////////////////////////////////////////////////////////////////////









}

void id_check(void)
{
    char user_buf[4], pass_buf[4], STATE;
    int i;

    fnd_flag = 1;

    STATE = 1;
    do {
////////////////////////////////////////////////////////////////////////////
// [문제 1] Cash Service Mode의 ID Check
////////////////////////////////////////////////////////////////////////////
```

```
    } while( STATE );

    fnd_flag = 0;
}

long input_money(void)
{
    long money = 0, num;
    int pos = 0;
    char buf[10];

    while( 1 ) {
        key = getkey(key);
        if( key_flag ) {
            switch( key ) {
                case KEY_0 :
                case KEY_1 :
                case KEY_2 :
                case KEY_3 :
                case KEY_4 :
                case KEY_5 :
                case KEY_6 :
                case KEY_7 :
                case KEY_8 :
                case KEY_9 :
                    if( pos < 6 ) {
                        if( (pos == 0) && (key == KEY_0) );
                        else {
                            lcd_gotoxy(pos + 3, 1);
                            lcd_data_write(key);

                            buf[pos++] = key - '0';
                        }
                    }
                    break;

                case KEY_MAN :
                    if( pos <= 2 ) {
                        if( pos == 1 )   money = buf[0];
                        else             money = buf[0] * 10 + buf[1];
                        money *= 10000;

                        return money;
                    }
                    break;

                case KEY_WON :
                    if( pos ) {
                        money = 0;
                        for( int i = 0; i < pos; i++ ) {
                            num = 1;
```

```
                        for( int j = 0; j < i; j++ )   num *= 10;
                        money += buf[pos - i - 1] * num;
                    }

                    return money;
                }
                break;

            case KEY_CANCEL :
                lcd_gotoxy(0, 1);
                lcd_string(" :                ");
                pos = 0;
                break;
        }
    }
}

#define IN_MONEY      1
#define OUT_MONEY     2

void inout_mode(void)
{
    long money;
    char STATE, error, mode;

    lcd_gotoxy(0, 0);
    lcd_string("1.Input Money    ");
    lcd_gotoxy(0, 1);
    lcd_string("2.Output Money   ");

    while( 1 ) {
        key = getkey(key);
        if( key_flag ) {
            if( (key == KEY_1) || (key == KEY_2) ) {
                if( key == KEY_1 ) mode = IN_MONEY;
                else               mode = OUT_MONEY;

                error = 0;
                STATE = 1;
                do {
                    lcd_gotoxy(0, 0);
                    if( mode == IN_MONEY )   lcd_string(" Input Money     ");
                    else                lcd_string(" Output Money     ");
                    lcd_gotoxy(0, 1);
                    lcd_string(" :                ");

                    money = input_money();

                    lcd_gotoxy(0, 0);
                    if( mode == IN_MONEY )   printf(" Input : %6ld ", money);
                    else                printf(" Output : %6ld", money);
                    lcd_gotoxy(0, 1);
                    lcd_string(" Please ENTER!   ");

                    while( (key = getkey(key)) != KEY_ENTER );
```

```
            if( mode == IN_MONEY ) {
                if( (save_money + money) > 9990000 )  error = 1;
                else save_money += money;
            }
            else {
                if( (save_money - money) < 0 )      error = 1;
                else save_money -= money;
            }

            if( error ) {
                lcd_gotoxy(0, 1);
                if( mode == IN_MONEY )  lcd_string(" Balance Excess ");
                else                    lcd_string(" Balance Lack    ");

                for( int i = 0; i < 3; i++ ) {
                    lcd_gotoxy(0, 0);
                    lcd_string(" Error!!!!        ");
                    _delay_ms(500);
                    lcd_gotoxy(0, 0);
                    lcd_string("                  ");
                    _delay_ms(500);
                }
            }
            else STATE = 0;
        } while( STATE );

        lcd_gotoxy(0, 0);
        lcd_string("Balance Inquiry ");
        lcd_gotoxy(0, 1);
        printf("  %7ld Won    ", save_money);

////////////////////////////////////////////////////////////////////////////
// [문제 2] FND에 현재 잔고 표시
////////////////////////////////////////////////////////////////////////////









        _delay_ms(3000);

        fnd_flag = 0;

        return;
```

```
            }
        }
    }
}

void account_mode(void)
{
    lcd_gotoxy(0, 0);
    lcd_string("Balance Inquiry ");
    lcd_gotoxy(0, 1);
    printf("  %7ld Won    ", save_money);

    _delay_ms(3000);
}

void cash_service(void)
{
    id_check();

    lcd_gotoxy(0, 0);
    lcd_string("1.Input/Output  ");
    lcd_gotoxy(0, 1);
    lcd_string("2.Account Check ");

    while( 1 ) {
        key = getkey(key);
        if( key_flag ) {
            if( key == KEY_1 ) {
                inout_mode();
                return;
            }
            else if( key == KEY_2 ) {
                account_mode();
                return;
            }
        }
    }
}

int main(void)
{
    mcu_init();
    fdevopen((void *)lcd_data_write, 0);

    start_bankbook();
    init_screen();

    while( 1 ) {
        key = getkey(key);
        if( key_flag ) {
            switch( key ) {
                case KEY_1 :
                    if( user_set )    cash_service();
                    init_screen();
                    break;
```

```
                case KEY_2 :
                    if( user_set )     new_user_setting();
                    else               user_setting();
                    init_screen();
                    break;
            }
        }
    }

    return 0;
}
```