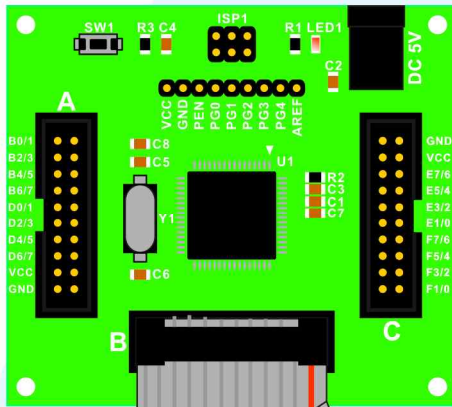
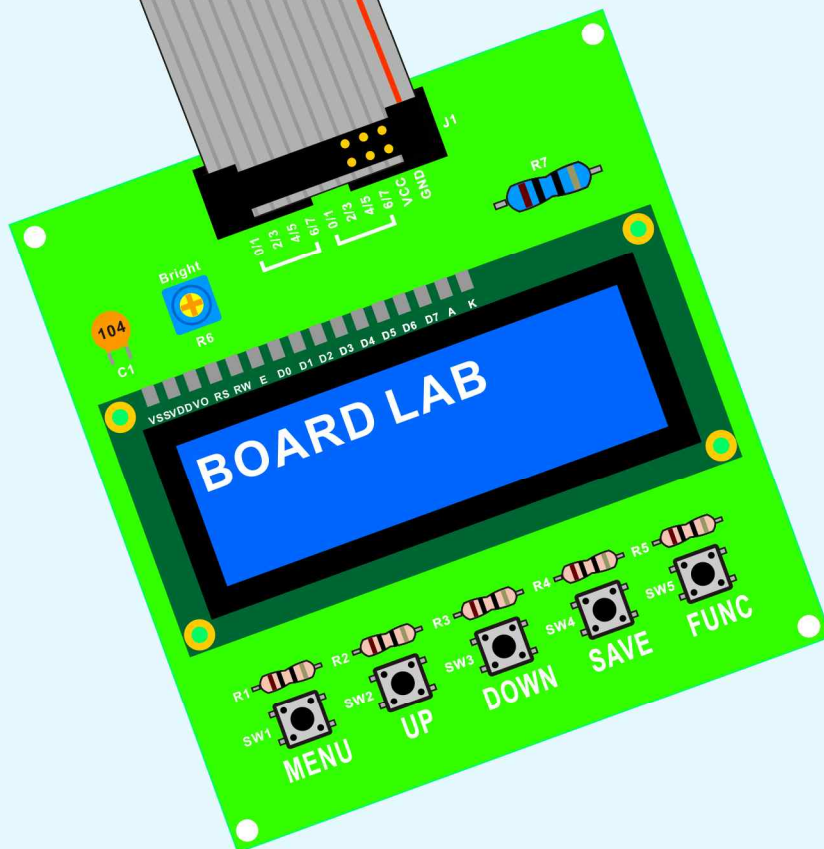


선배가 취업나와 보니까 꼭 필요하더라

전자분야 취준생을 위한 실무 프로젝트



- 실무 프로젝트를 활용한 교육
- 스스로 생각하는힘 형성
- 다양한 예제를 통한 실력향상
- 실무 경험 꿀팁 전달



www.BoardFree.kr

저자:김명수.이소리 공저 / 감수 : 김용필

차례

01. MAIN BOARD

ATmega128 개요
컴파일러
ISP
메인보드
보드 to 보드 연결법

02. LED MATRIX

기본설명, 예제
프로젝트 1 : 미니 전광판
프로젝트 2 : 통신 전광판

03. FND

기본설명, 예제
프로젝트 1 : 전자시계
프로젝트 2 : 모터제어기
프로젝트 3 : 스텝모터제어기
프로젝트 4 : 전자온도계
프로젝트 5 : 온도감지 후드
프로젝트 6 : 온도감지 벨브

04. LCD

기본설명, 예제
프로젝트 1 : 전자시계
프로젝트 2 : 모터제어기
프로젝트 3 : 스텝모터제어기
프로젝트 4 : 전자온도계
프로젝트 5 : 온도감지 후드
프로젝트 6 : 온도감지 벨브

05. 만능기판 사용법

기본설명, 예제
제어 루프 프로그램
Buck타입을 이용한 전압제어
Boost타입을 이용한 전압제어

06. 자료 정리

출처모음

04. LCD

기본설명, 예제

프로젝트 1 : 전자시계

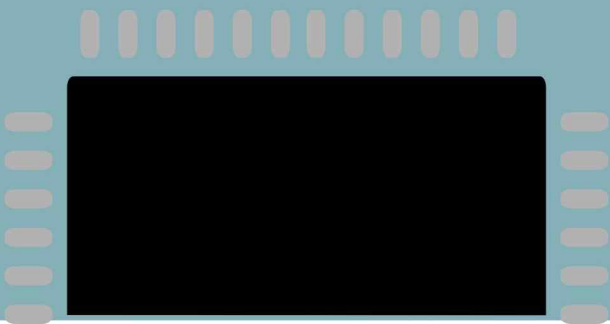
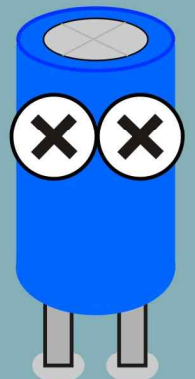
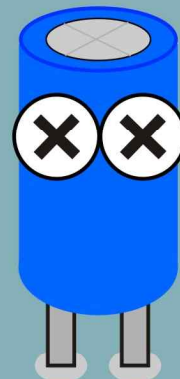
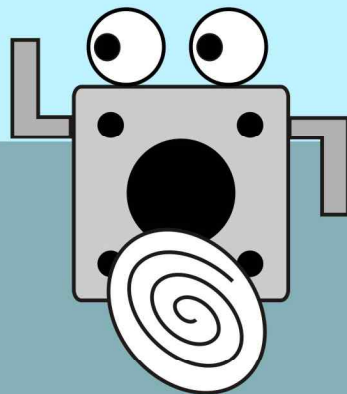
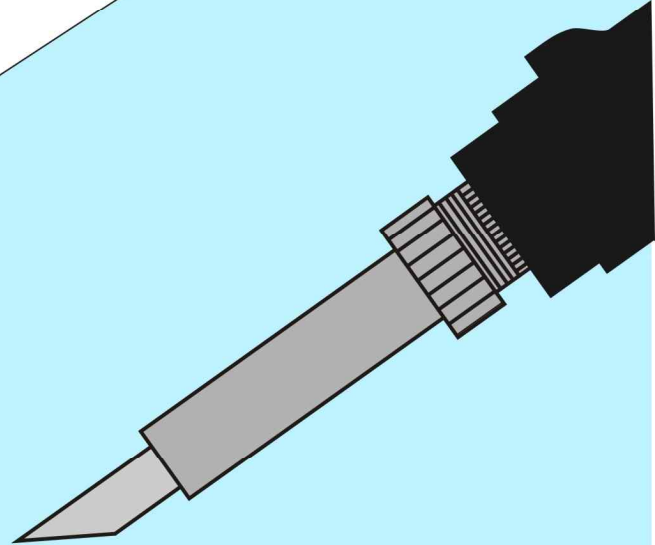
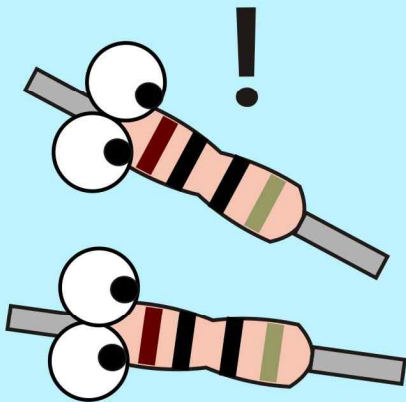
프로젝트 2 : 모터제어기

프로젝트 3 : 스텝모터제어기

프로젝트 4 : 전자온도계

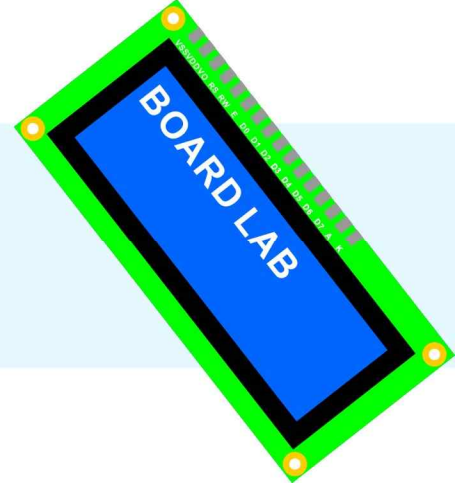
프로젝트 5 : 온도감지 후드

프로젝트 6 : 온도감지 벨브



1

기본설명 및 예제



1.1 LCD가 뭔지 아니?

갑자기 LCD에 대한 기본지식을 설명하니까 어려울 수 있지만, 지금부터 설명하는 내용은 LCD의 기본 중의 기본이야. 이런 기초지식을 알고 있으면 할 얘기가 많아지니까 많은 도움이 될 거야. 실제로 엔지니어가 이런 지식을 가지고 있지 않다면 실력이 부족하다는 인식을 하게 될 수 있어!

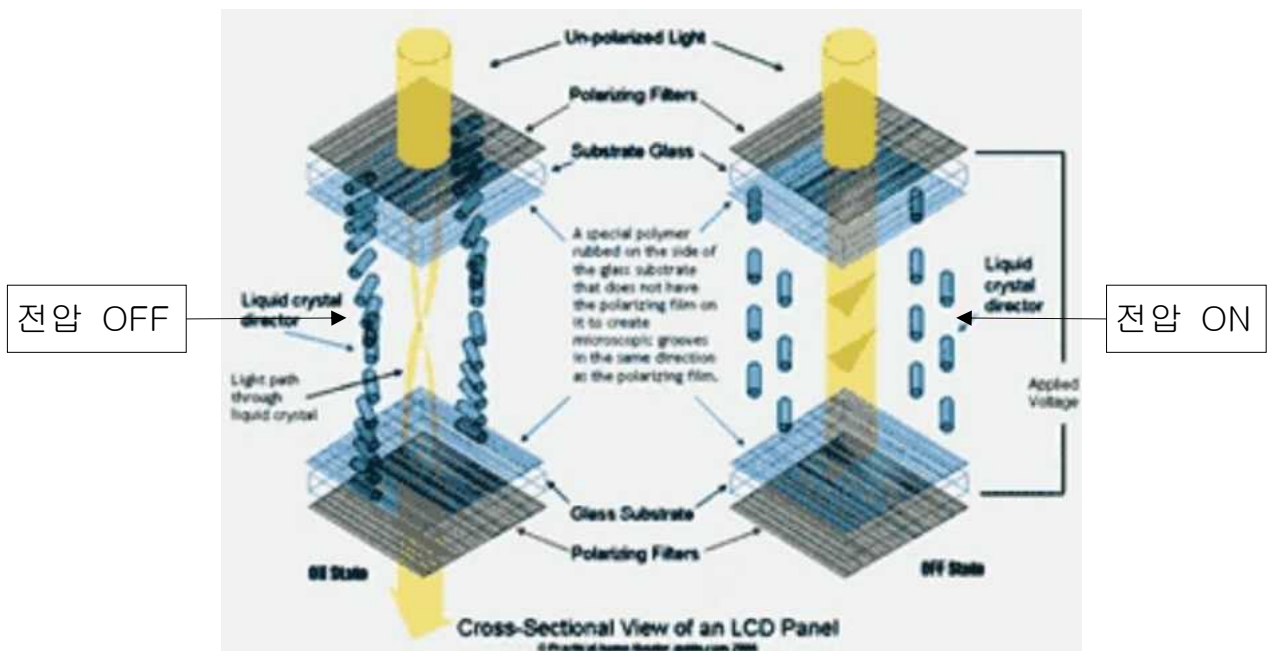


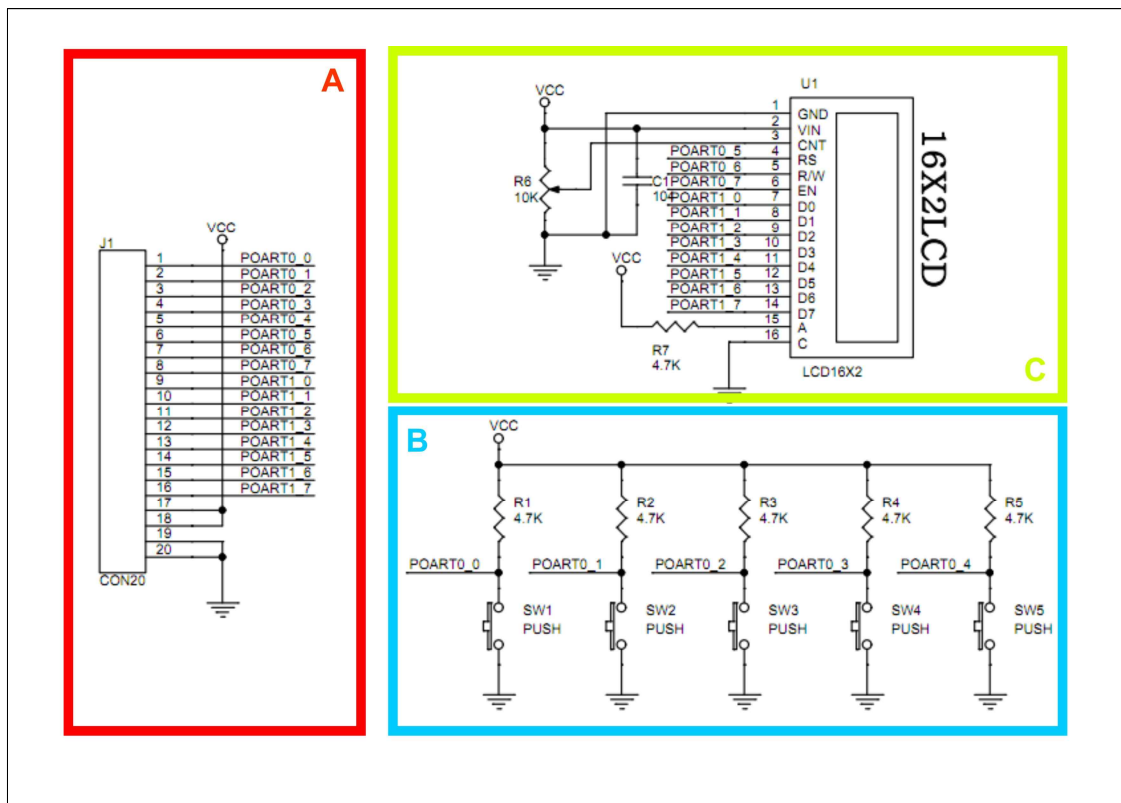
그림4-1. 액정 패널의 구동원리

LCD는 액정이라는 물질을 이용한 디스플레이 장치인데 여기서 액정은 고체와 액체 사이의 물질이야. 액정이란 가열하면 액체가 되고 식히면 고체가 되는 상태를 말해. 액정을 이용하는 이유는 전기적인 성질이 특이하기 때문이야. 전기적인 성질이라 말하면 조금 어려울 수 있지만, 전압을 가해주면 액정분자의 배열이 바뀌는 간단한 특성이야.

액정분자의 배열이라니까 상상이 안 가지? 그림4-1의 그림을 봐봐. 액정분자 배열을 보면 왼쪽 그림은 전압이 OFF 상태이기 때문에 파란 분자가 회전하는 배열인 거고, 오른쪽 그림은 전압이 ON 상태라 배열이 일자로 정렬되는 그림인 거야. 쉽지?

마찬가지로 액정분자의 배열에 따라 빛의 방향도 똑같이 바뀌어. 이걸 이용해서 특정 방향의 빛만을 통과시키는 편광판을 이용해서 출력으로 나가는 빛의 양을 조절하는 거야. 쉽게 말해서 액정이 빛의 방향을 바꿔주면서 편광판을 통과하는 양을 조절한다는 거지. 정리하자면 처음에 백라이트가 빛을 제공해주고 액정을 통해 조절된 빛이 모여서 글씨처럼 보이게 되는 거야.

1.2 회로도



A. 박스 헤더

LCD보드의 박스헤더는 LCD를 구동시키기 위한 11개의 핀과 스위치에서 신호를 받기 위한 5개의 핀으로 구성되어 있어. 일반적인 IO 핀

만 있으면 되니까 어떤 박스헤더에 연결해도 상관없어.

B. SWITCH

5개의 스위치로 각각 MENU, UP, DOWN, SAVE, FUNCTION의 기능을 가지고 있어. MENU스위치는 FND에는 존재하지 않았던 스위치라 어떻게 써야 좋을지 잘 모를 거라 생각해. LCD는 화면을 옮겨가면서 다양한 표현하기 좋은 특징을 가지고 있지? 이 특징을 살리기 위해 MENU 스위치가 있다고 생각하면 될 것 같아.

C. LCD

LCD는 11개의 데이터 핀과 5개의 전원 핀으로 구성되어 있어. 전원 핀이 왜 이리 많지? 하고 생각하고 있을 거야. 그 이유는 LCD 전원과 백라이트용 전원을 따로 사용하기 때문이야.

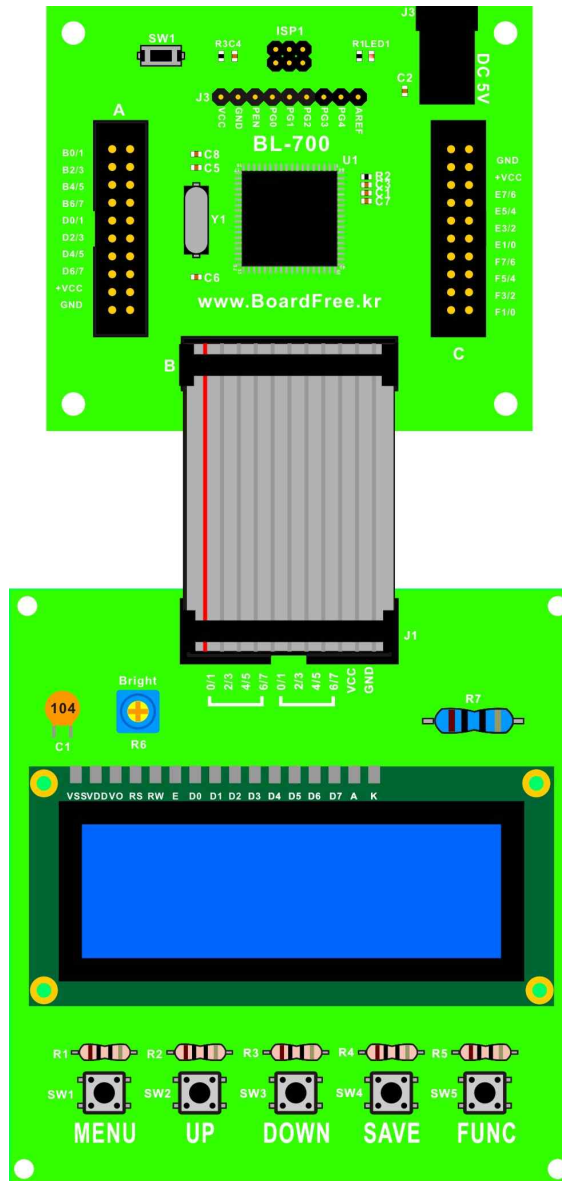
그런데 LCD, 백라이트에 VCC와 GND가 하나씩 들어가도 VCC가 하나 남지? 남아있는 VCC는 글씨의 선명도를 결정해줘. 회로도를 보면 가변저항이 하나 있을 거야. 이 가변저항을 통해 글씨의 선명도를 조절할 수 있도록 하는 거야.

이제 데이터 핀에 관해서 설명해줄게. 실제로 데이터 핀은 RS, RW, EN, D4~D7만 있어도 충분히 조작할 수 있어. 그런데 너희가 다양하게 사용할 수 있도록 D0~D3도 연결했으니까 여러 가지 방법으로 코드를 작성해볼 수 있을 거야.

1.3 LCD의 기본적인 사용법 알기 1

현재 보드에는 RS, R/W, EN핀과 8개의 데이터 핀이 있어서 설정에 알맞은 값을 넣어주는 것으로 LCD에 쓰고 지우는 게 가능해. 하지만, CodeVision에서는 기본적으로 alcd.h를 통해서 LCD를 사용하기 쉽게 되어있어. 지금부터 그 사용방법을 알려줄게.

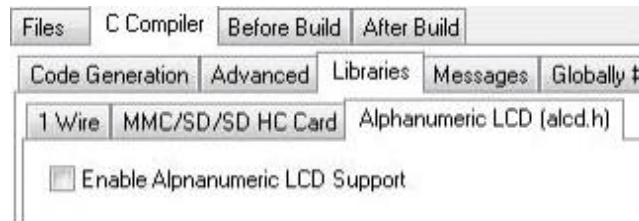
참고로 보드간의 연결은 아래 그림과 같이 했어!



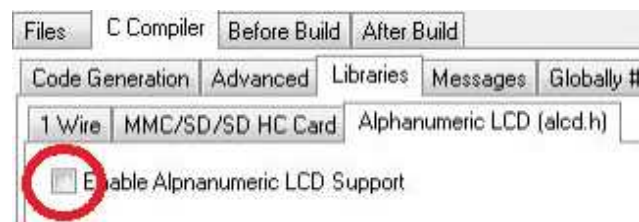
1. Project의 Configure에 들어가봐.



2. Compiler의 Libraries의 LCD부분에 들어가면 Enable이 보일거야.



3. Enable에 체크를 해줘.



4. MCU에서 LCD와 연결된 핀을 설정해줘. 내 경우에는 PORTA, PORTC쪽 박스헤더에 연결했으니까 아래사진과 같이 설정했어.



5. 아래에 간단한 LCD코드가 있으니까 작성해봐.

```
#include <alcd.h>           //선언
void main(void)
{
    lcd_init(16);           //LCD초기화
    lcd_gotoxy(0, 0);       //LCD의 첫 번째 칸, 줄로 이동
    lcd_puts("ABC123#!@");  //다양한 문자 출력
}
```


alcd.h에는 위에 사용한 함수처럼 LCD를 사용하기 위한 함수가 다수 들어있어. LCD에 간단하게 글씨를 쓴다면 위에 있는 함수만으로도 충분하지만 다양하게 사용하는 게 좋으니까 시간이 날 때 한 번씩 해봐.

6. 코드를 컴파일해서 보드에 넣은 뒤에 LCD를 확인하면 끝!

1.4 LCD의 기본적인 사용법 알기 2

위에 있는 코드는 LCD에 무언가를 작성할 수는 있지만, 그 이상의 응용은 불가능해. 그래서 변수를 출력하도록 하고 그 변수 값을 바꿔 주는 것이 평범한 방법이야.

```
lcd_puts(lcd_1); //lcd_1이라는 변수에 담긴 문자를 출력
```

물론 변수 lcd_1은 최대16개의 문자를 포함해야하기 때문에 선언에서도 16개의 배열로 해줘야해.

```
int lcd_1[16]; //lcd_1의 선언
```

이후 lcd_1에 자신이 원하는 글자만 담으면 시간에 따라 계속 변화하는 글자도 쉽게 표현할 수 있어. 예를 들면 시계나 온도계 같은 값이 변화하는 것 말이야. 자신이 원하는 글자를 담기 위해서 사용하는 간단한 함수가 있어.

```
sprintf(lcd_1, "Time:%d", time_data)
```

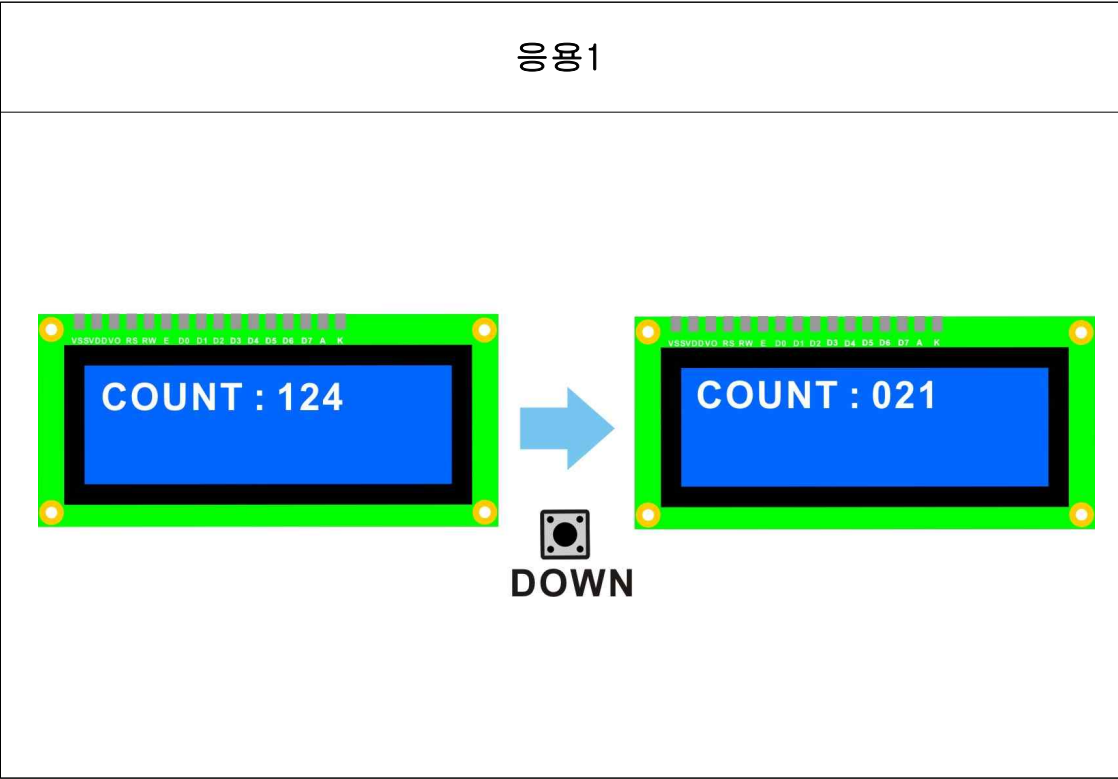
위에 함수는 time_data라는 변수에 1초마다 1씩 증가하도록 하면 그 값이 %d 칸에 반영되고 그렇게 완성된 글자를 lcd_1이라는 변수에 넣

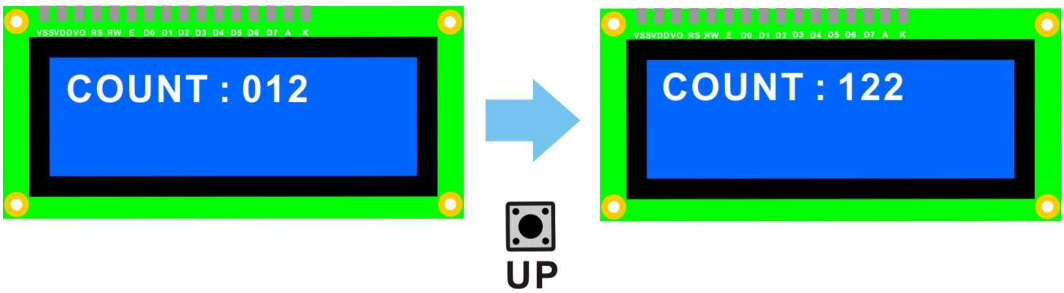
는 거야. 그리고 위에 있는 lcd_puts를 해주면 간단한 초시계가 되는 거지. 이후에도 어떤 값을 담느냐에 따라 다양한 응용이 가능하지. 이후의 응용을 위해 기본적인 %의 종류들을 나열해 놔어.

%c	단일문자	%o	8진수형
%d	정수형	%p	포인트형
%e	지수형	%s	문자열형
%f	실수형	%x	16진수형

1.5 응용해보기

이제 위에서 배운 내용을 바탕으로 응용해보자!





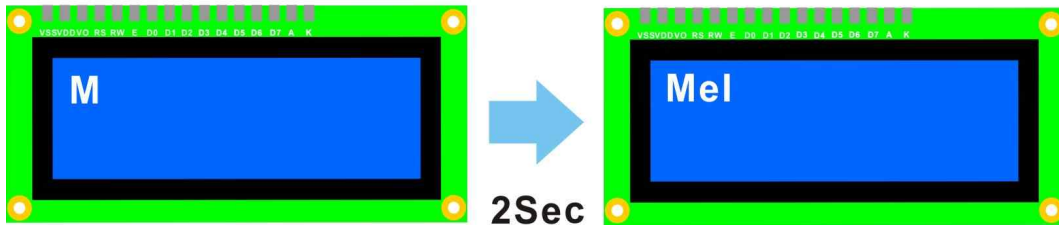
스위치를 통해 0~255를 표현하는게 좋을 것 같아. Up버튼을 누르면 1씩 증가하고 Down버튼을 누르면 1씩 감소하는 코드를 짜보자. 물론 아래 그림처럼 12면 앞에 0이 붙어서 3자리로 표현되도록 해야되!

응용2



타이머 카운트를 이용해서 1초마다 1씩 0~255까지 증가하도록 코드를 작성해보자. 이것도 위에처럼 항상 3자리를 차지하도록 작성해야되.

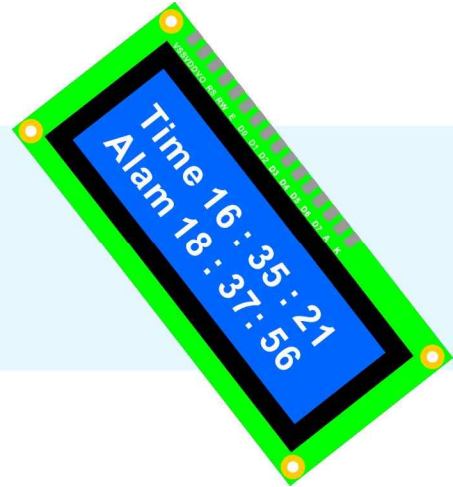
응용3



위의 응용을 이용해서 1초마다 1글자씩 LCD에 글씨가 써지는 코드를 작성해보자. 글씨가 다 작성되면 초기화한 후 다시 글씨가 작성되도록 해야 되.

2

프로젝트1:전자시계



2.1 전자시계가 뭘까?

전자시계를 만드는 종류는 크게 2가지가 있어. 첫 번째는 설정한 시간부터 시간이 흘러가는 시계야. 이 경우에는 전원을 끌 경우 끈 시간 동안은 시간이 흘러가지 않으니 오차가 발생할 수 있어. 두 번째는 인터넷에서 현재 시간을 가져오는 시계로 통신이 유지되는 한 정확한 시간을 전달해주지만, 통신이 끊기면 시계가 멈춘다는 단점이 있지. 지금부터 우리가 만들 건 첫 번째 시계야.

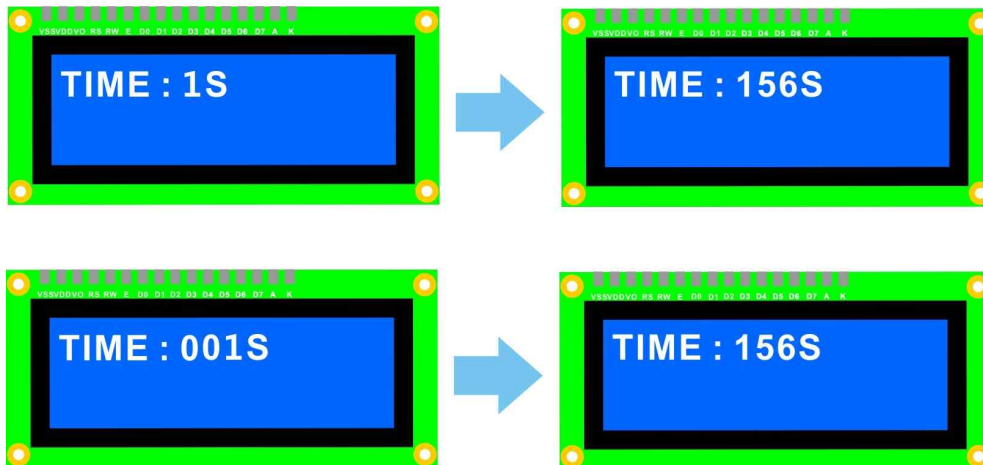
2.2 전자시계 기본적인 코드

전자시계에서 가장 중요한 부분은 시간을 세는 부분이라고 할 수 있어. 하지만 이 부분은 FND에서 이미 만들었을 거야. 그래서 타이머 인터럽트 부분은 생략할게. 먼저 타이머 인터럽트를 이용해 만든 시간을 LCD에 표시하는 방법을 설명할 거야. 위에 있는 응용에서 간단한 초시계를 만들었을 거야. 3자리 수를 유지하는 부분에서 다양한 방법이 나왔을 거로 생각해. 그중 효율이 좋은 방법은 ‘%03d’를 사용하는 방법이야.

```
sprintf(lcd_1, "Time:%03d", time_data)
```

이렇게 하면 항상 3자리 수를 표현하면서 코드는 간결하거든 시계에서는 ‘%02d’를 사용해서 시, 분, 초를 항상 2자리 수로 사용하면 돼. 혹시 코드를 짜면서 왜 자릿수를 맞춰가면서 LCD에 표시해야 되나 생각하고 있니? 1초면 01초가 아니라 1로 표현해도 의미는 똑같이 전달되니까 그런 생각을 할 수 있다고 생각해. 하지만 그렇게 표현하면 누가 봐도 예쁘다는 생각이 들지 않는 형태로 글자를 표현하게 되는 거야.

아래 그림을 보면 첫 번째 그림은 시간 뒤에 나오는 S의 위치가 계속 변하지만 두 번째 그림은 항상 같은 위치에 S가 쓰일 거야. 아래 그림을 보았을 때 아래쪽이 조금 더 안정감이 있다고 생각이 들 거야. 하지만 사람에 따라서 이런 느낌이 느껴지지 않을 수 있어.

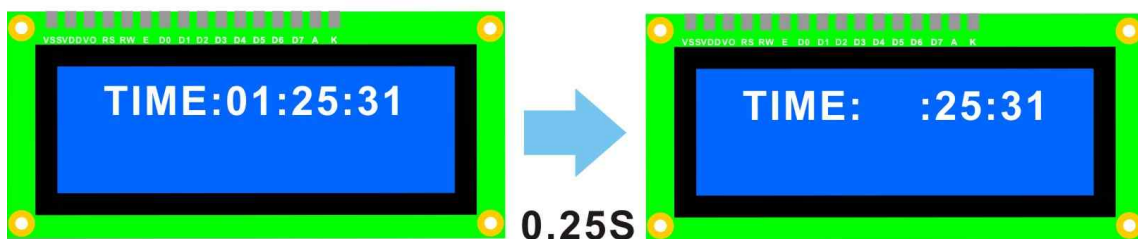


그림에서 느껴진 게 있다면 다음은 시계에 관한 코드를 작성하는 단계야. 시, 분, 초를 각각 표현하는 건 쉬워

```
printf(lcd_1, " Time:%02d:%02d:%02d ",time_24h,time_1h,time_1m)
```

time변수들에는 각각 24시간, 1시간, 1분을 카운트하도록 하면 간단한 시계가 완성되는 거야. 물론 이 시계는 지금 스위치를 통해 시간을 제어하지 못하는 상태지. 우선 스위치 채터링은 이미 앞에서 끝냈을 거니까 생략할게.

스위치는 시, 분, 초를 늘리거나 줄이는 데 사용할 거야. 그런데 보통 LCD를 보면 설정값을 변경할 때 그 부분이 점멸하지? 여기서도 그 기능을 넣어줄 건데 그렇게 어렵지 않아. 지금 아래 그림은 0.25초 동안 시간 부분을 공백으로 두고 0.25초 동안 시간을 표시하는 걸 반복하는 모습이야.



이렇게 선택한 부분이 점멸되게 하고 싶다면 아래 코드처럼 0.5초를 만들고 난 후 절반의 시간은 공백으로 표시하고 절반의 시간은 실제 글씨를 표시하면 점멸하는 글씨를 쓰는 게 가능해. 아래 있는 코드처럼 func을 카운트해서 그 부분에 맞는 부분을 0.25s동안 공백으로 만든 후 나머지 시간은 제대로 표시하는 거야.

```
if(func==1&&count_500ms>=250)
    sprintf(lcd_1," TIME   :%02d:%02d",time_1h,time_1m);
else if(func==2&&count_500ms>=250)
    sprintf(lcd_1," TIME %02d:  :%02d",time_24h,time_1m);
else if(func==3&&count_500ms>=250)
    sprintf(lcd_1," TIME %02d:%02d:  ",time_24h,time_1h);
else
    sprintf(lcd_1," TIME %02d:%02d:%02d",time_24h,time_1h,time_1m);
```

이렇게 바꿀 부분이 점멸하도록 만들었으면 이제 해야 할 부분은 Up, Down스위치를 통해 실제로 값이 변하도록 만들어야 해. func스위치가 1이면 1시간이 올라가고 2면 1분이 올라가고 3이면 1초가 올라가도록 말이지. 물론 내려가는 동작도 똑같은 방식으로 만들어야 해. 단, 스위치로 시간을 바꾸는 동안 시계는 멈춰있어야 해! 스위치로 시간을 조절하는 기능을 완성했다면 거의 완성이야.

완성했다고? 그럼 이제 SAVE버튼을 통해 현재 시간을 저장해두는 기능을 만들 거야. 이 기능은 EEPROM에 저장되기 때문에 전원을 종료해도 데이터를 저장해줄 거야. EEPROM은 전원이 꺼져도 내용은 남아서 시계로 동작하도록 데이터를 저장해 줄 거야.

```
eeprom unsigned int save_24h, save_1h, save_1m;
```


위의 선언을 해주면 이제 전원이 꺼져도 데이터를 저장하는 변수가 완성되는 거야. 이제 SAVE버튼을 눌렀을 때 현재 시, 분, 초를 EEPROM 변수들에 넣어주고 코드의 시작 부분에 EEPROM 데이터가 범위 안에 있는 데이터라면 현재 값에 넣어주는 코드를 작성하면 돼. 예를 들면 아래처럼 말이지.

```
if(save_button_press_flag==1)
{
    save_button_press_flag=0;
    func=0;
    save_24h = time_24h;
    save_1h = time_1h;
    save_1m = time_1m;
}

if(save_24h <= 23) time_24h = save_24h;
if(save_1h <= 59) time_1h = save_1h;
if(save_1m <= 59) time_1m = save_1m;
```

위 방법이 틀린 방법은 아니지만 별로 좋지 않은 방법이야. 코드에도 ‘가독성’이 있거든. 위에 코드와 아래 코드를 비교해보면 위에 코드는 한번 보면 이해할 수 있지만 아래 코드는 한번 보고 이해가 불가능해. 또, 어떤 의미로 이런 코드를 사용하는 건지 알 수 없지.

```
if(save_button_press_flag==1)
{
    save_button_press_flag=0;
    func=0;
    save_time = (time_24h * 3600) +
    (time_1h * 60) + time_1m;
}

if(save_time<=86400)
```

```

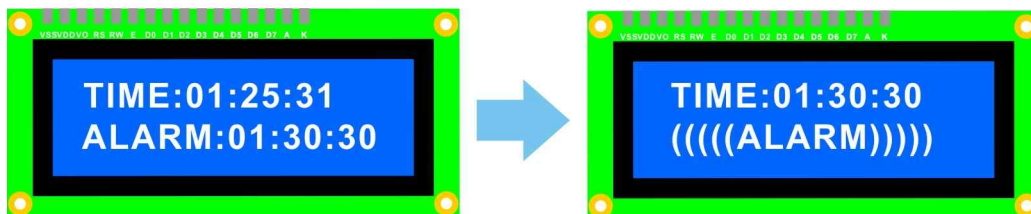
{
    time_24h = save_time / 3600;
    time_1h = (save_time%3600)/60;
    time_1m = (save_time%3600)%60;
}

```

그래서 항상 코드를 작성할 때는 보는 사람이 이해하기 쉽도록 변수는 확실히 의미를 담고 식은 최대한 간단하게 만들어야해.

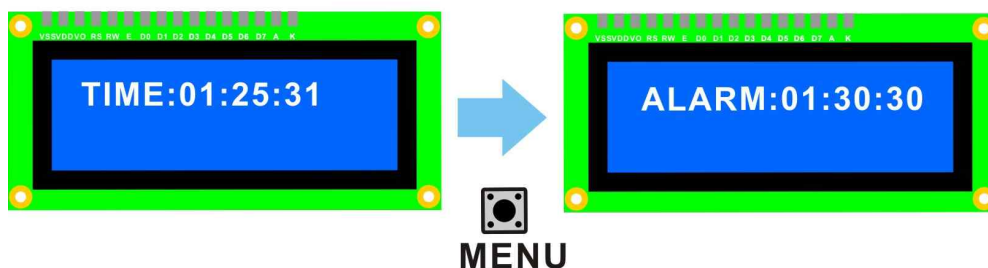
2.3 응용해보기

응용1



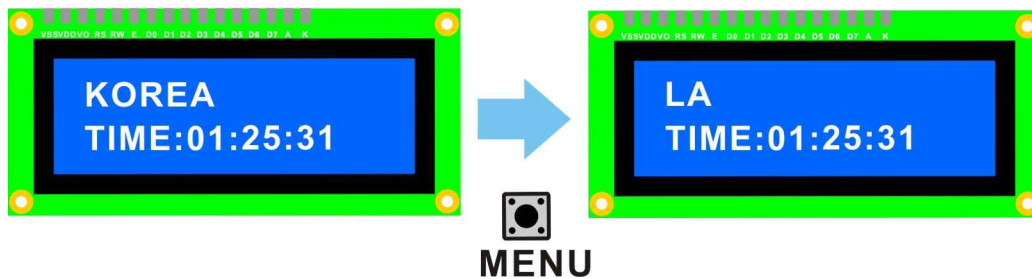
지금까진 전자시계의 첫 번째 줄에 시간을 표시하고 있었을 거야.이제 그 아래에 알람을 넣어서 시계와 같은 시간이 되면 1분 동안 알람표시 칸이 특정 문자로 바뀌도록 만들면 간단한 알람시계가 완성해봐.

응용2



물론 func버튼으로 알람시계의 시, 분, 초를 선택하고 Up, Down 할 수 있도록 만들어야해. 하지만 이렇게 하면 알람을 설정할 때 마다 시계가 멈춰버리지? 그래서 이번엔 두 번째 페이지에 만드는 거야. 첫 번째 페이지는 시계만 출력하고 두 번째 페이지에 알람이 나오도록 만드는 거야. 이렇게 하면 알람을 설정하더라도 시계가 멈추거나 하는 일은 없어. 페이지의 변경은 Menu버튼으로 하게 해봐!

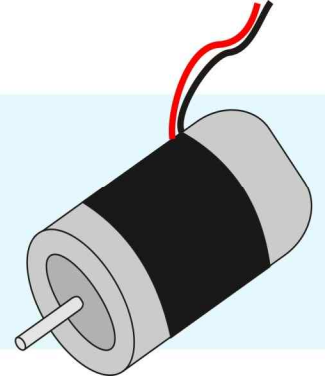
응용3



알람시계의 응용이 끝났다면 이번엔 세계시간을 표시하는 거야. Menu버튼을 통해 화면을 옮겨가면서 각각 다른 국가의 시간을 표시하는 거지. 아래 그림처럼 윗줄에 국가명을 적고 아랫줄에 그 국가의 시간을 적는 거야. 이렇게 5개정도의 국가를 한 시계에 표현할 수 있도록 만들면 완성이야.

3

프로젝트2:모터제어기

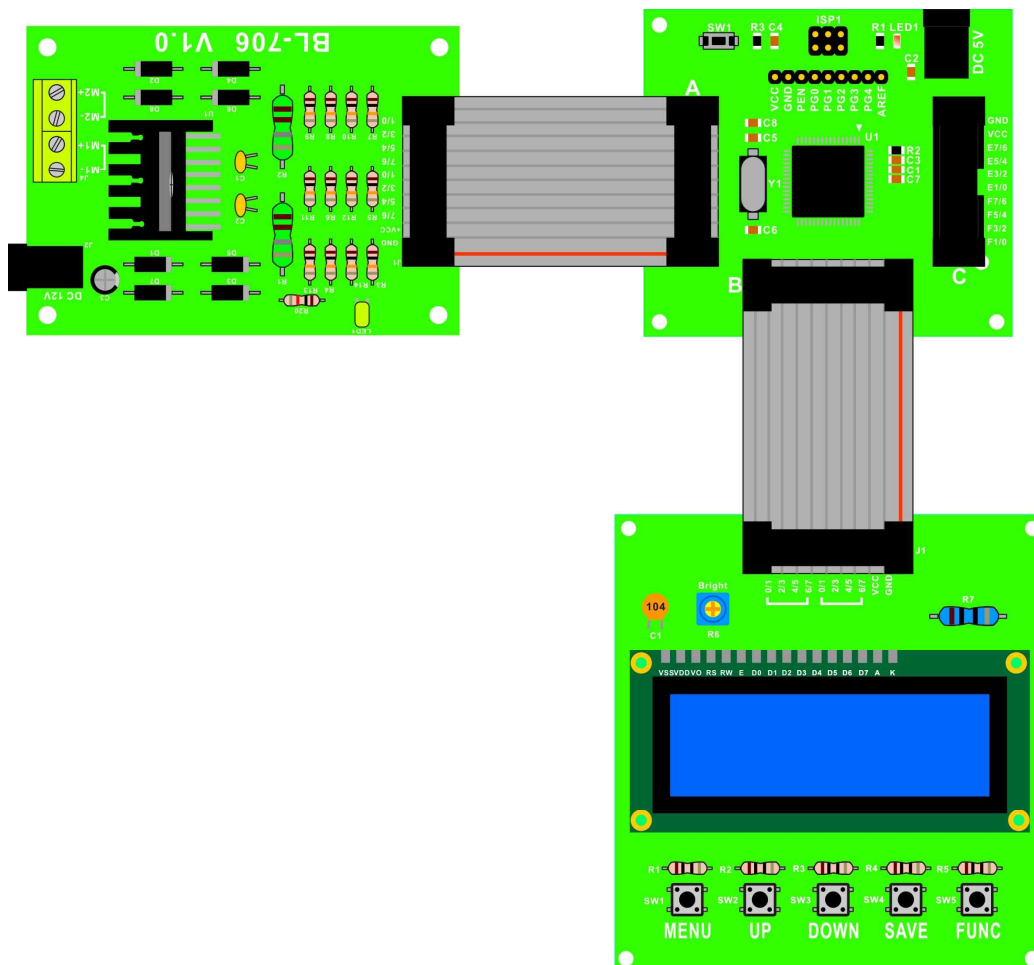


3.1 모터제어기란?

모터제어기는 쉽게 말해서 스위치만으로 모터를 원하는 대로 제어하도록 만드는 거야. 너가 모터를 움직여서 해야 하는 일이 있다고 생각해봐. 그런데 A는 가변저항으로 감으로 속도를 맞춰야 되고 B는 스위치 한번 누르는 걸로 자동으로 속도를 조절해간다면 어떤 쪽이 편할까? 당연히 B쪽이 편리할거야. 그래서 우리는 B같은 모터제어기를 만들어 보는 거야.

3.2 모터의 제어하는 코드작성하기

보드간의 연결은 아래 그림과 같이 했어.



지금부터 할 내용은 LCD를 통해 모터를 제어하는 거야. 간단하게 정회전 속도를 결정하는 부분부터 시작하자.



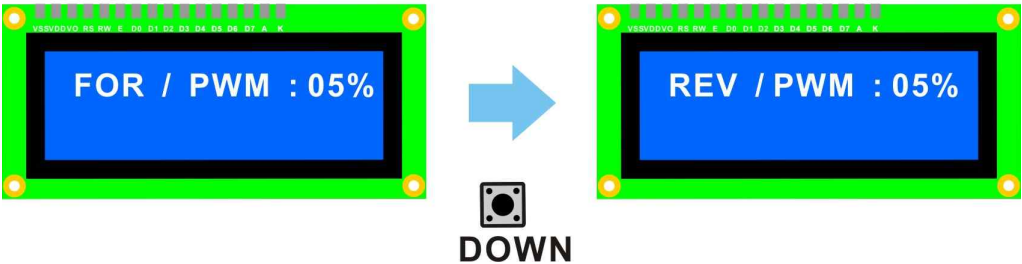
```
void main()
{
    TCCR1A = 0x83;
    TCCR1B = 0x0B;
    TCNT1 = 0;

    DDRB = 0xff;
    for(i=0; i<9; i++)
    {
        data[i/3][i%3]=data_save[i/3][i%3]%100;
    }
    ASSR=0x00;
    TCCR0=0x0C;
    TCNT0=0x00;
    OCR0=0xFA;
    TIMSK=0x02;
    lcd_init(16); // lcd초기화
    #asm("sei")
    while(1)
    {
        pwm_value = 250;
        OCR1A = pwm_value;

        sprintf(lcd_1,"PWM:%d",pwm_value);
        lcd_gotoxy(0, 0); lcd_puts(lcd_1); // 첫째줄 표시
    }
}
```

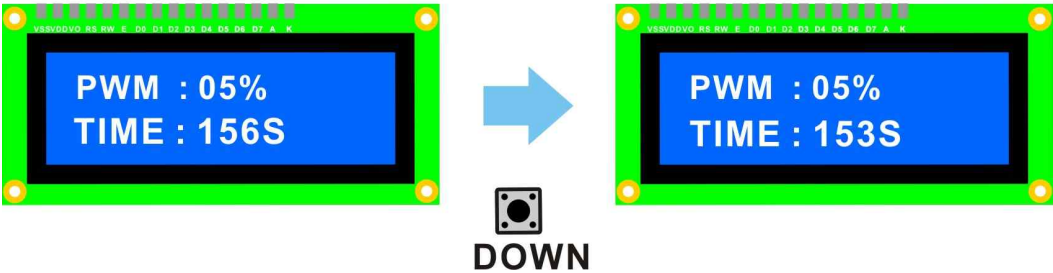
3.3 모터 응용하기

응용1

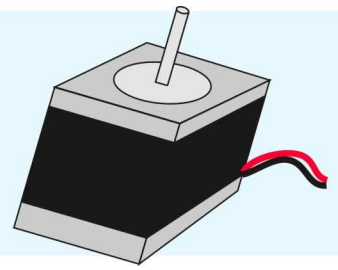


모터응용의 첫 번째는 모터의 방향을 조절해주는 부분이야. 아래 그림처럼 Down버튼으로 역방향 회전하고 Up버튼으로 정방향 회전하도록 결정하는 거지. 아래 그림처럼 말이야.

응용2



모터응용의 두 번째는 회전할 시간을 아랫줄에 표시하는 기능이야. 정해진 시간만큼 회전한뒤 정지하도록 만드는 거지.

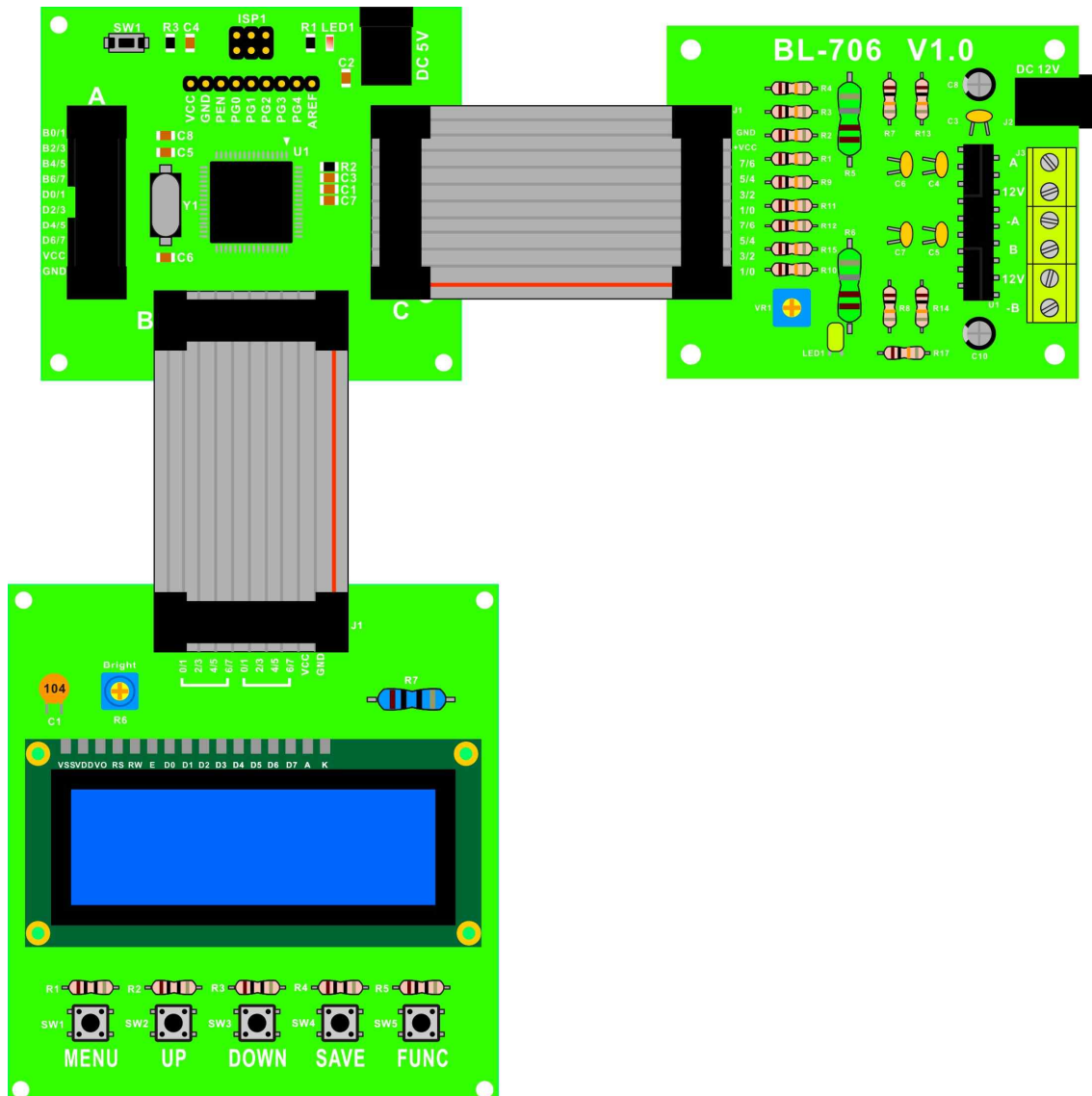


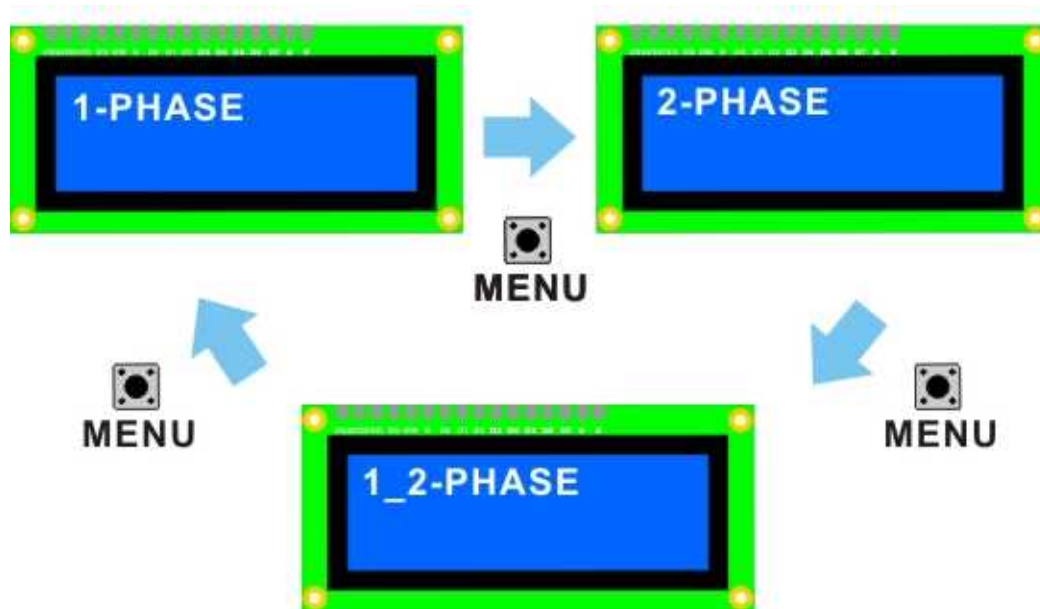
4.1 스텝모터 제어해보기

제어 방법에 대한 설명은 생략할게. LCD부분에서는 FND보다 다양한 기능을 넣을 수 있을 거야. 지금 가르쳐 줄 LCD의 사용법 이외에도 마음껏 응용하길 바래!

4.2 스텝모터 제어하는 코드 작성하기

보드간의 연결은 아래 그림과 같이했어.





```
#include <mega128.h>
#include <delay.h>
#include <alcd.h>
#include <stdio.h>

#define STEP_PORT    PORTB
#define MENU PINC.0
#define CHT_value 150

// Declare your global variables here
unsigned char one_ms_count=0;
unsigned char step_count=0;
unsigned char step_1[4]={0x10,0x40,0x20,0x80};
unsigned char step_2[4]={0x50,0x60,0xA0,0x90};
unsigned char step_3[8]={0x10,0x50,0x40,0x60,0x20,0xA0,0x80,0x90};
unsigned int MENU_CHT;
unsigned char motor_step_temp;
char lcd_1[16], lcd_2[16];

void step_up(void);
void step_down(void);
void step_task(void);
void cpu_init(void);
```

```

void display_lcd();

interrupt [TIM0_COMP] void timer0_comp_isr(void)
{
    one_ms_count=one_ms_count+1;
    if(one_ms_count>10)
    {
        one_ms_count=0;
        step_up();
    }
    if(MENU == 0)
    {
        if(MENU_CHT==CHT_value)
        {
            MENU_CHT=0;
            motor_step_temp = (motor_step_temp +1)%3;
        }
        else
        {
            MENU_CHT+=1;
        }
    }
    else MENU_CHT=0;
}

void main(void)
{
    cpu_init();

    while (1)
    {
        step_task();
        display_lcd();
    }
}

void step_up()
{
    step_count=(step_count+1);
}

```

```

}

void step_down()
{
    if(step_count==0)
    {
        if(motor_step_temp == 2) step_count=7;
        else step_count=3;
    }
    else
    {
        step_count=step_count-1;
    }
}

void step_task()
{
    if(motor_step_temp == 0) STEP_PORT=step_1[step_count%4];
    else if(motor_step_temp == 1) STEP_PORT=step_2[step_count%4];
    else if(motor_step_temp == 2) STEP_PORT=step_3[step_count%8];
}

void display_lcd()
{
    if(motor_step_temp == 0) sprintf(lcd_1,"motor menu1");
    else if(motor_step_temp == 1) sprintf(lcd_1,"motor menu2");
    else if(motor_step_temp == 2) sprintf(lcd_1,"motor menu3");

    lcd_gotoxy(0, 0); lcd_puts(lcd_1);
}

void cpu_init(void)
{
    PORTB=0x00;
    DDRB=0xF0;
    DDRC = 0x00;
    ASSR=0x00;
    TCCR0=0x0C;
    TCNT0=0x00;
}

```

```

OCR0=0xF9;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
lcd_init(16); // lcd초기화
TIMSK=0x02;
ETIMSK=0x00;
#asm("sei")
}

```

4.3 스텝모터 응용하기

응용1



스텝 모터와 DC 모터의 가장 큰 차이점은 각도의 유무야. 스텝 모터는 원하는 각도만큼 회전하고 멈추는 게 가능해. 그래서 지금 만들 기능은 0.5초마다 일정 각도만큼 회전하는 기능이야. 어렵다고 생각할 수 있지만, 생각보다 쉬울 거야.

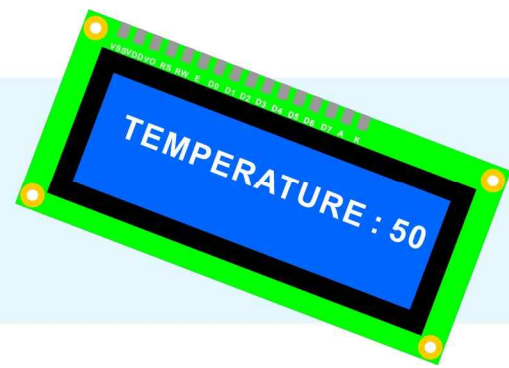
응용2



일정 각도를 회전하도록 코드를 작성했다면 이제 반복하는 시간을 0.1 ~ 1초로 조절할 수 있도록 만드는 거야. 하지만 이 부분은 각자 간단하게 만들 수 있을거라 생각해. 시간 조절정도는 다른 응용에서도 기본적인 부분이었으니까. 아래 그림의 LCD 그림처럼 설정할 수 있도록 만들면 끝이야.

5

프로젝트4:전자온도계



5.1 전자온도계가 뭘까?

전자 온도계의 관한 설명은 FND쪽에서 이미 끝났으니까 생략할게 이
번에 사용하는 온도 센서도 NTC 써미스터야.

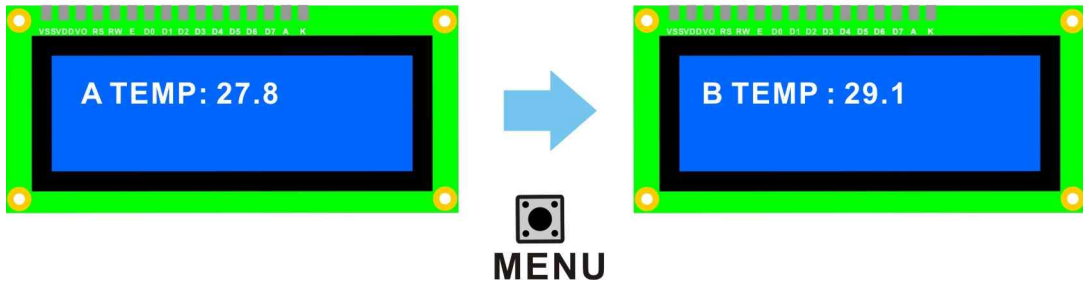
5.2 전자온도계 응용

응용1



지금 사용하는 온도센서를 보면 0.3이나 0.2씩 올라가면서 부자연스럽게 보이지? 그 이유는 계산식을 거치니까 항상 0.1씩 올라가지 못하는 거야. 그래서 지금부터 응용은 0.1씩 올라가도록 코드를 작성하는 거야. 다양한 방법이 나올거라 생각해. 우선 목표는 아래 그림처럼 0.1씩 증가하도록 LCD에 표시하면 돼. 테스트는 손으로 잡아보는 것만으로 충분해.

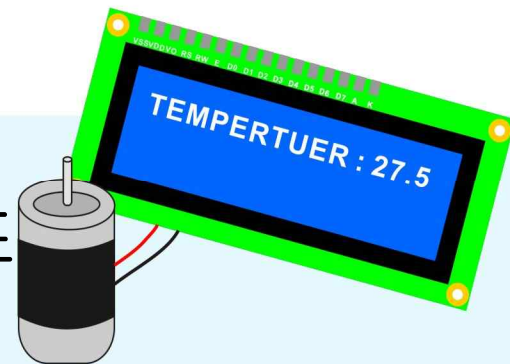
응용2



온도센서 3개를 동시에 사용해서 3지점의 온도를 각각 표시하는 거야. Menu버튼을 통해 각 지점의 온도를 돌아가면서 볼 수 있도록 코드를 작성하는 거지. 실제로 다양한 지점의 온도를 측정하는 일도 있었으니까 말이지. 아래 그림처럼 LCD에 3지점의 온도를 표시하도록 작성하도록 해.

6

프로젝트5:온도감지 후드

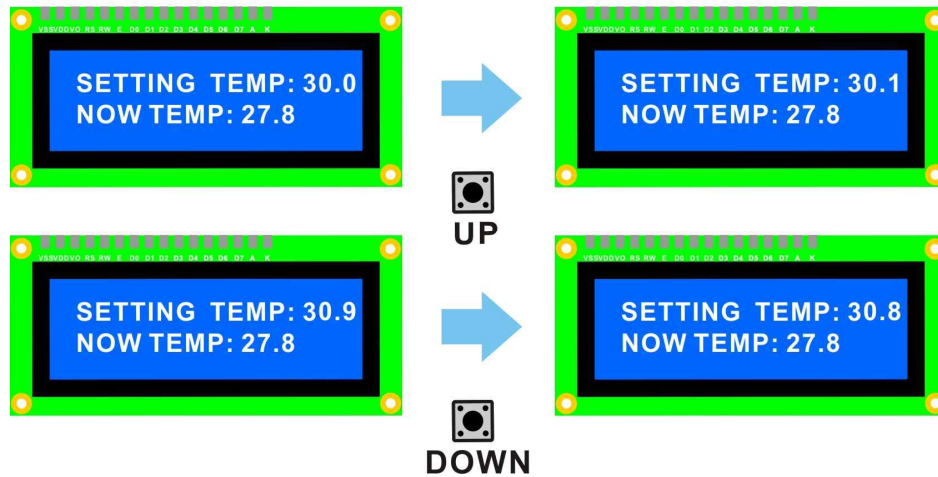


6.1 온도감지 후드란?

지금까지 만들어온 프로젝트들이 각 주요 부품을 다루고 응용해 왔다면 지금부터는 2개의 부품을 합쳐서 프로젝트다운 프로젝트를 만드는 거야. 온도감지 후드는 말 그대로 온도를 감지하고 일정량을 넘어가면 후드가 작동하면서 기체를 외부로 방출하는 기능을 가졌어. 모터보드 + 온도계보드라고 생각하면 좋아.

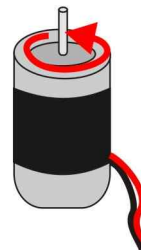
6.2 온도감지 후드 응용

응용1



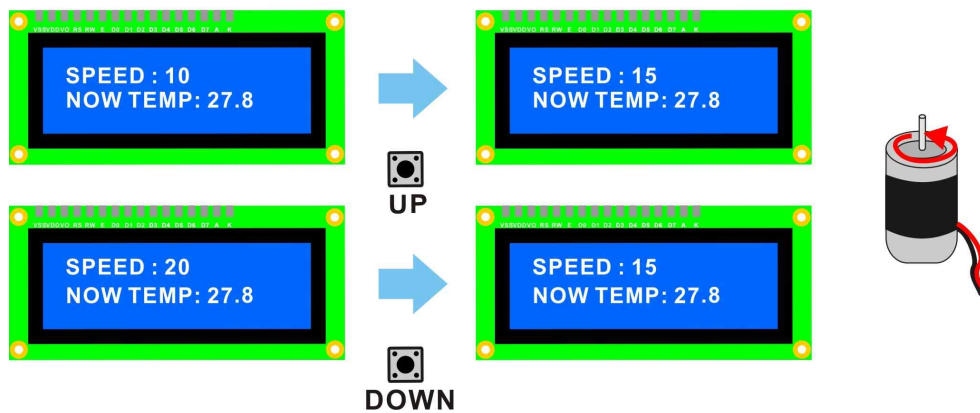
온도감지 후드의 응용은 온도의 임계점을 스위치로 결정할 수 있도록 만드는 거야. 위에 그림처럼 온도를 표시하면서 온도의 임계점을 스위치로 조절할 수 있도록 만드는 거야. UP버튼을 누르면 높은 온도에서 후드가 작동하도록 조작하고 DOWN버튼을 누르면 낮은 온도에서부터 후드가 작동하도록 말이지.

응용3



세 번째 응용은 범위를 넘은 양이 크면 클수록 모터가 빠르게 회전하도록 하는 기능이야. 실제로 온도에 따라서 필요한 후드 속도가 다르니까 자동으로 조절하도록 말이지.

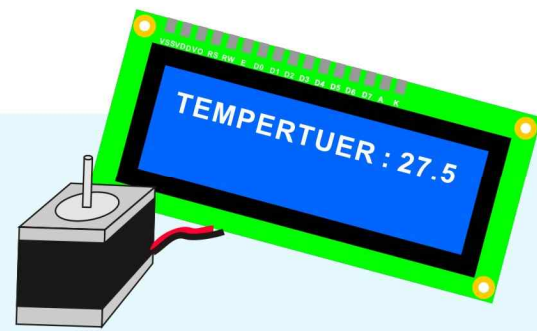
응용2



버튼을 통해 후드가 회전하는 속도를 스위치로 조절할 수 있어야 해.

7

프로젝트6:온도감지 밸브



7.1 온도감지 밸브란?

온도감지 밸브는 온도를 감지하면 밸브를 닫음으로써 2차 피해를 미연에 방지할 수 있도록 만들어주는 기능이야. 온도가 높으면 밸브를 닫아서 가스를 차단하고 온도가 낮으면 열어두는 기능을 가지게 돼.

7.2 온도감지 기본 사용법

온도감지 밸브는 온도센서 + 스텝모터라고 생각하면 돼. 기본적인 코드는 온도가 높으면 밸브가 90도 움직이고 온도가 내려가면 다시 반대 방향으로 90도 움직여서 밸브를 열어 두는 거야. 위에 있는 코드들을 이해했다면 충분히 작성할 수 있을 거야.

7.3 온도감지 밸브 응용

