마이크로프로세서 설계 프로젝트 결과 보고서

과제	이름	한글 시계
제출	날짜	14. 12. 19(금)
성	명	컴퓨터과학부 2010920023 김희중

1주제 선정

- 감성디자인, 시계

자유 주제의 프로젝트를 하게 되어 평소 크로스 컴파일을 공부하던 중 만들어보고 싶었던 주제를 선정하게 되었다. 컴퓨터 옆에 조그마한 시계가 있으면 좋겠다는 생각을 항상 했었다. 하지만 그 때문에 탁장 시계를 놓는 것은 과하다는 생각이 든다. 분명 윈도우의 우측 하단에 항상 시계가 나오고 있지만 그 흔한 알람기능 조차 없으며 시계라는 인식 자체가 들지 않는다. 컴퓨터 화면을 보면서도 시간이 궁금할 때는 손목시계나 휴대폰을 찾아보는 것이 일반적이다.

하지만 USB 전원만으로도 작동하고 시간을 읽는 가독성이 뛰어나며 감성을 자극하는 디자인까지 탑재된 시계가 컴퓨터 옆에 있다면 손목을 돌려본다거나 마우스대신 스마트 폰을 들었다 놓는 수고를 하지 않게 될 것이다. '시계'라는 주제를 처음 마주했을 때 코딩의 기술적인 소양이 부족하여 쉽게 가는 길을 선택했다고 생각 할 수도 있다. 하지만 그냥 숫자만 화면에 뿌려주는 시계가 아닌 Dot Matrix를 완벽하게 제어할 수 있는 코딩을 해야 하기 때문에 처음 배우는 분야의 학기 말 프로젝트로 적당한 난이도를 갖는다고 생각한다. 그냥 시계가 아닌 디자인적인 요소에 조금 더 신경 쓴 시계이고 그 디자인을 만족하기 위해 숫자를 직접 출력하는 시계와 다른 코딩 기법이 필요하다.

2기능

- 시간 출력

Dot Matrix에서 LED불빛이 나오는 위치들의 조합으로 시간을 나타낸다. 시계를 나타내기 위해서 Dot Matrix위에 씌울 한글판 제작이 필요하다.

- 시간 설정

시간을 수정하여 현재 시간과 맞는 시간을 출력하게 한다.

- 알람 설정

특정 시간이 되면 부저를 울리는 알람 기능이 있으며 알람 시간을 설정할 수 있고 알람을 On/Off 할 수 있다.

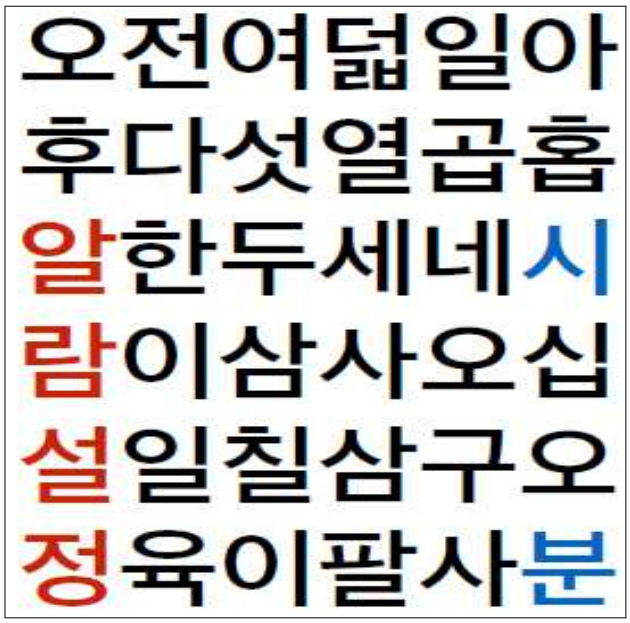
- 스탑 워치

직관적인 기호로 스탑 워치 즉, 초시계 기능을 나타낼 수 있다.

3 내용

- 시간의 표시

기본적으로 시계이기 때문에 시간을 표기한다. 시간을 표시하는 방법은 8X8 Dot Matrix를 이용하는데 6X6의 LED만 있으면 오전, 오후 모든 시간을 표시할 수 있다. 시간 표시 뿐 아니라 알람과 시간설정 메뉴를 나타내는 것 까지도 가능하다. 아래에 6x6의 LED로 시간을 표시하기 위한 한글판 그림이 있다.



이 한글판을 제작해서 Dot Matrix 위에 부착 시키면 LED가 빛나는 것으로 시간 표시를 할 수 있다. 왼쪽 상단부터 오른쪽 하단 방향으로 빛나는 글씨를 읽으면 자연스럽게 시간을 읽은 것이 된다.

- 알람

알람을 울릴 시간을 설정할 수 있고 알람을 On/Off할 수 있다. JKit판에 내장되어있는 부저를 이용해서 알람기능을 구현 하였다.

- 스탑 워치의 표현

8X8판중 6X6밖에 사용하지 않았다. 오른쪽과 하단에 2줄의 LED가 놀게 된다. 남는 LED를 사용하기 위해서 스탑 워치 기능을 만들도록 한다. 이 스탑 워치는 숫자나 한글을 통해서 읽는 것이 아니라 LED에 불이 몇 개가 들어왔는지를 세어서 알게 한다. 이는 PIMP라는 시계에서 고안했다. 아래 사진에서 보는 것처럼 어떤 위치의 LED등이 몇 개가 켜져 있는지를 통해서 시계를 읽는 방식이다. 한글시계에서도 마찬가지로 LED등이 켜지는 개수와 위치로 사용자가 직관적으로 시간을 알 수 있게 한다. 하지만 LED의 개수가 제한적인 관계로 10분 안쪽의 시간만 측정 할 수 있다. 긴 시간을 측정할 수는 없지만 책상에 둘 작은 시계의 역할로써 충분하다고 생각한다.

- 기능의 제어

시간을 재설정 할 수 있어야한다. 알람을 켜고 끄며 알람 시간을 설정할 수 있어야 한다. 스탑 워치를 켜고 멈추고 끌 수 있어야 한다. 간단하게 나열했을 때 생각보다 제어해야할 요소들이 많이 있다. 스위치가 많이 필요하다는 뜻이다. 8개의 스위치가 1열로 나열된 Switch Board를 이용해서 이 기능들을 제어하도록 한다.

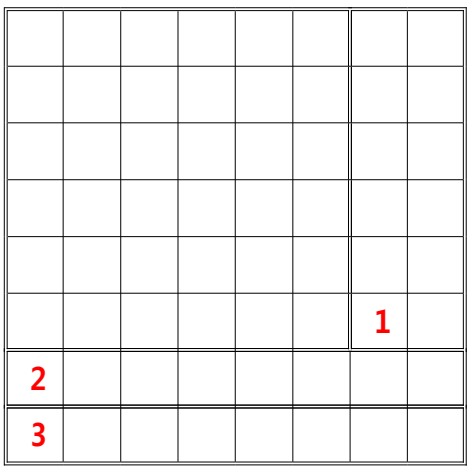
④예시

- 시간 읽는 법

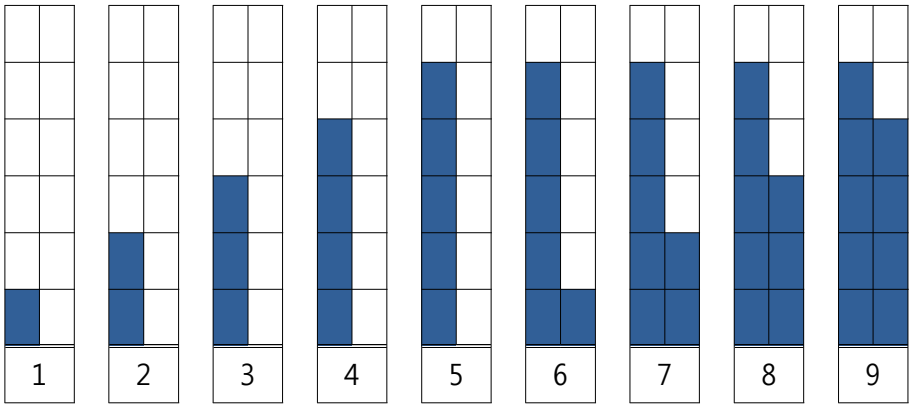
아래의 그림을 설명하자면 녹색 네모에 대응되는 LED만 빛나도록 한다. 그러면 왼쪽 상단부터 우측 하단 방향으로 빛나는 글씨만 읽게 되면 "오전 일곱시 사십팔분"이 된다. 이렇게 시각의 흐름에 따라서 글씨를 읽으면 자연스럽게 시간을 읽은 것이 된다.

- 스탑워치 읽는 법

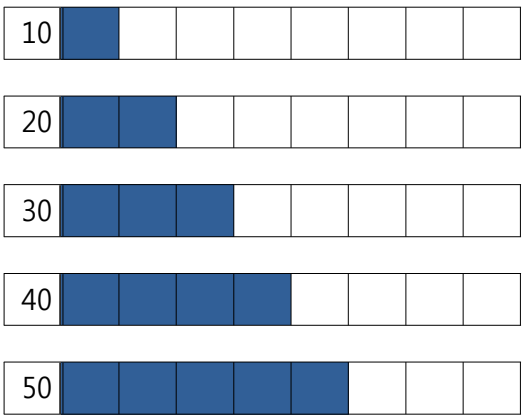
아래의 그림에서 대략적인 한글 시계의 구조를 나타낸다. 6X6의 한글판 이외의 LED는 스탑워치로 사용하는데 4개의 구역으로 나누었다. 각 구역에서 LED를 초, 분 으로 환산하는 방법을 설명하도록 한다.



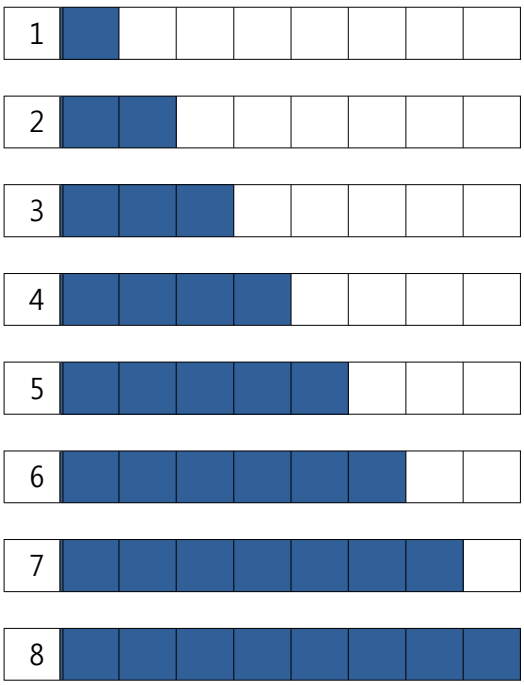
- 1번 구역은 '초'를 의미한다. 각 초에 대응되는 LED의 모양을 나타낸 그림이 있다.



- 2번 구역은 '10초'를 의미한다. 각 초에 대응되는 LED 모양을 나타낸 그림이 있다.

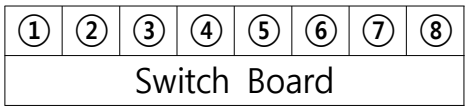


- 3번 구역은 '분'을 의미한다. 각 분에 대응되는 LED 모양을 나타낸 그림이 있다.



- 스위치 조작 법

스위치보드는 대략적으로 아래와 같이 스위치가 1열로 나열된 부품이다. 각 스위치마다 부여된 기능이 있다.



1. 알람 버튼
- 알람의 On / Off 상태를 변경하는 스위치이다.
- 알람이 On 상태이면 일반 모드일 때 "알람"에 LED등이 켜진다.

2. 시계 모드의 변경
- 버튼을 한번씩 누를 때 마다 일반, 시간 설정, 알람 시간 설정 순서로 Mode가 변경된다.
- 평소상태가 일반모드이다.
- 시간 설정 모드에 들어가면 "설정"에 LED등이 켜지며 2, 3번 스위치로 시간을 재 설정한다.
- 알람 시간 설정 모드에 들어가면 "알람설정"에 LED등이 켜지며 2, 3번 스위치로 알람 시간을 설정한다.

3 Stop Watch Start / Stop
- 스탑 워치를 출발 및 일시정지 시킨다.

4. Stop Watch 모드 변경
- 스탑 워치모드를 On / Off 상태를 변경하는 스위치이다.

5. 분 조정
- 시간 설정, 알람 시간 설정 모드에서 1번 누를 때 마다 '분'이 1씩 증가한다.

6 시 조정
- 시간 설정, 알람 시간 설정 모드에서 1번 누를 때 마다 '시'가 1씩 증가한다.

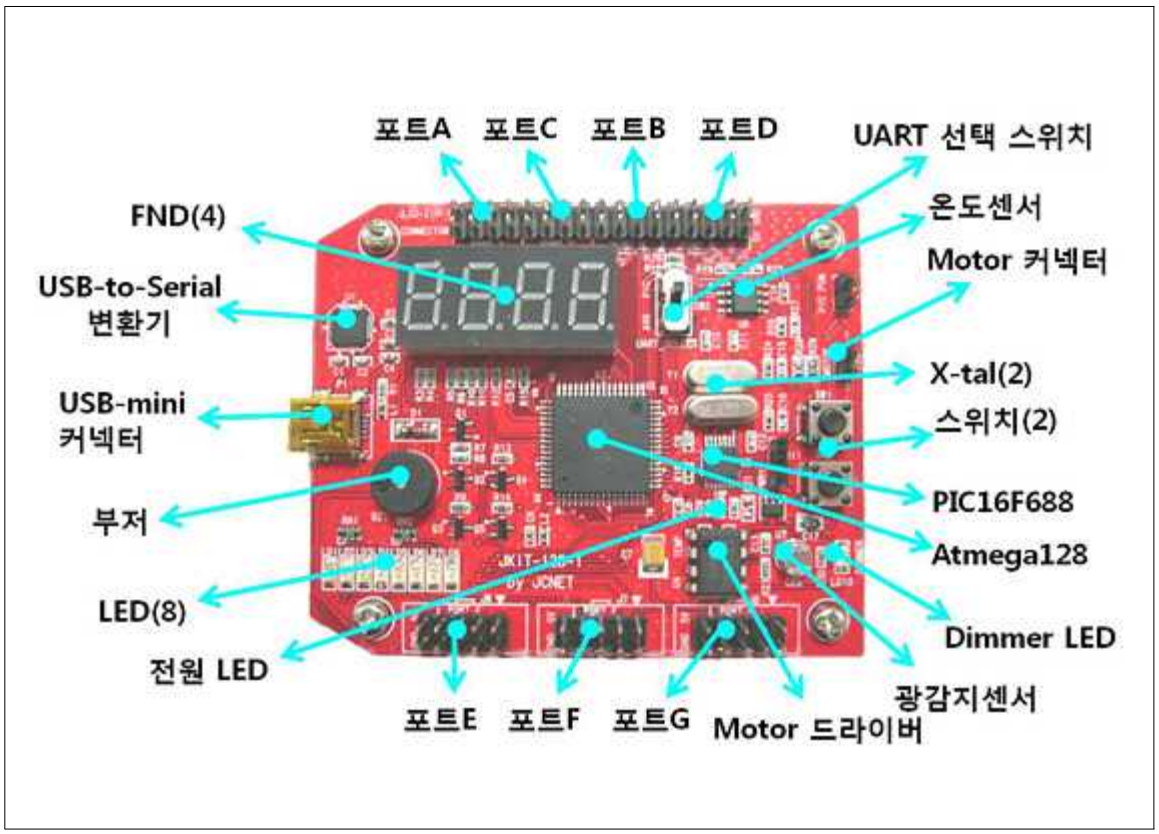
7. 없음

8. 없음

5주요 부품

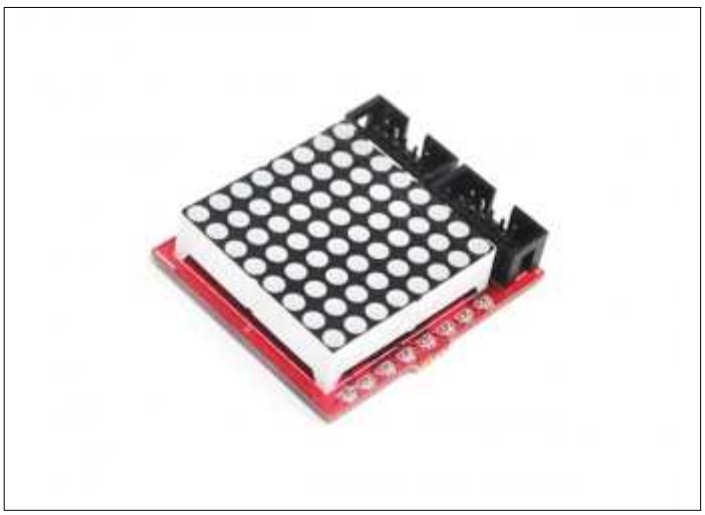
- JKIT-128-1

프로젝트의 기반이 되는 Atmega128 키트이다. 프로젝트를 위해서 내장된 기능 중 부저만 사용할 예정이다. 소리를 내는 것을 제외한 모든 기능은 외장 모듈을 연결해서 구현 할 예정이다.



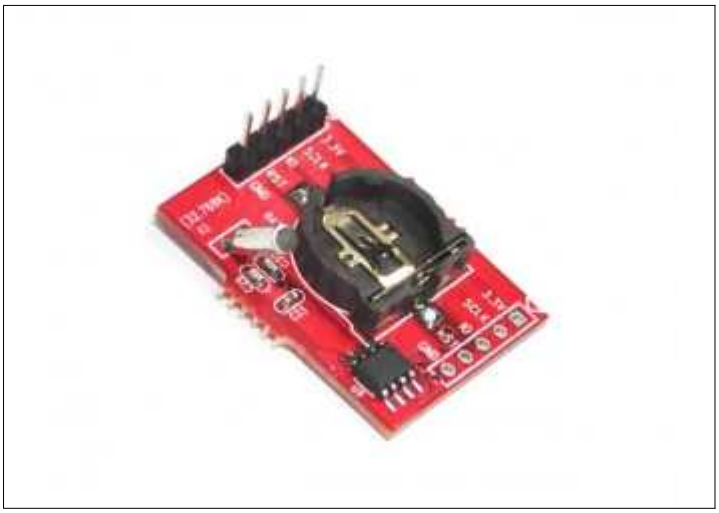
- 8X8 Dot Matrix

8X8 적색 모듈의 Dot Matrix를 이용한다. 6X6의 LED는 시간을 표시하는 용도로 사용하고 나머지 LED는 스탑워치에 사용한다.



- RTC DS1302 모듈

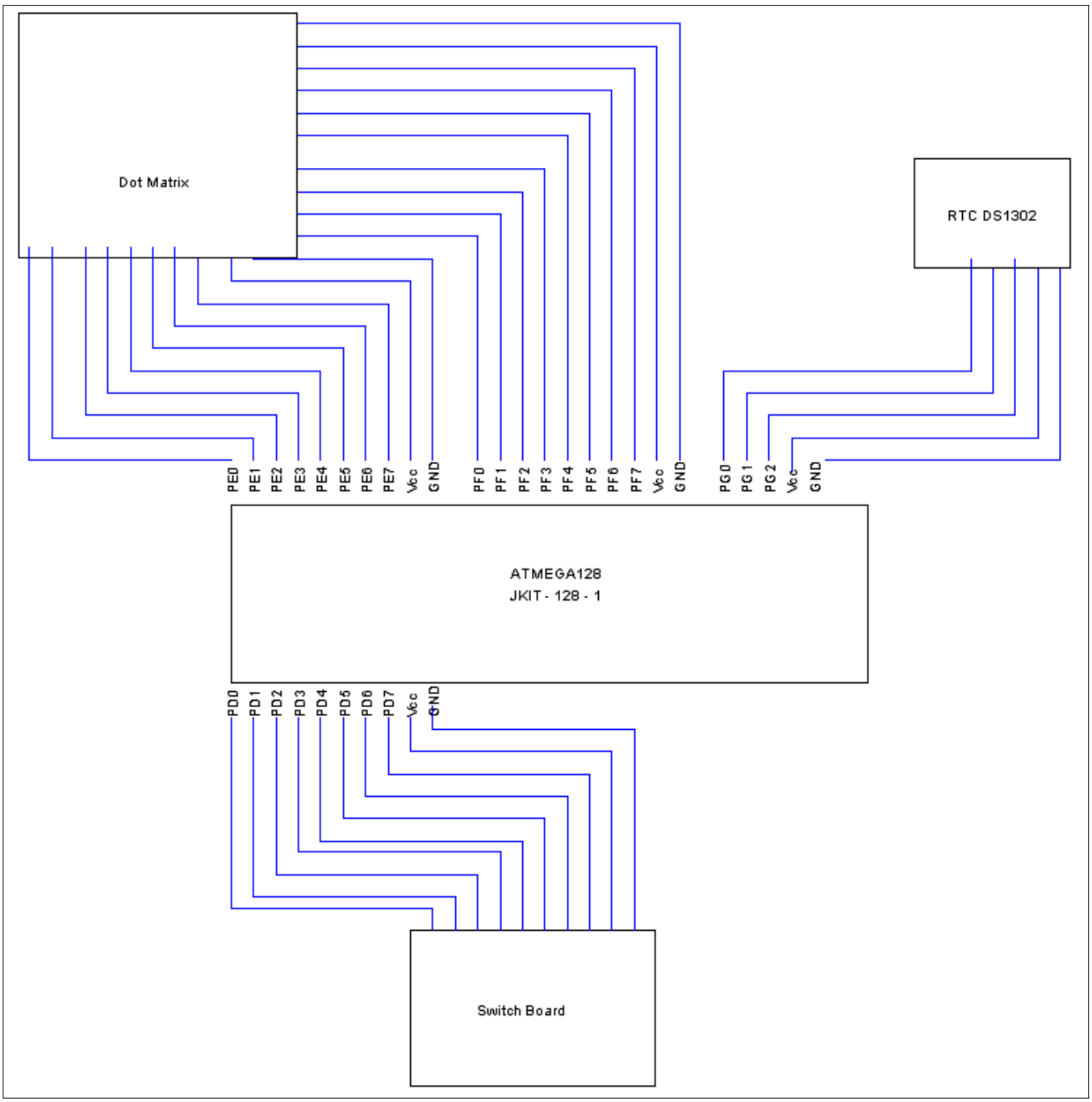
- Real Time Clock(RTC) 모듈
- 실시간으로 년, 월, 일, 시간, 분, 초를 카운트 하는 칩 (2100년 까지 보장)



- 스위치보드(AM-TS8)

8개의 스위치가 일렬로 나열되어있다. 각각 스위치의 사용 용도는 앞에서 설명하였다.

6 회로도


Dot Matrix
RTC DS1302
PE0 PE1 PE2 PE3 PE4 PE5 PE6 PE7 Vcc GND
PF0 PF1 PF2 PF3 PF4 PF5 PF6 PF7 Vcc GND
PG0 PG1 PG2 Vcc GND
ATMEGA128
JKIT - 128 - 1
PD0 PD1 PD2 PD3 PD4 PD5 PD6 PD7 Vcc GND
Switch Board

7 소스코드

- Dot Matrix에 불이 들어오게 하는 함수

- 8X8의 매트릭스를 이용한다. PORTE의 8개의 핀은 가로 8줄과 대응 되고 PORTF의 8개의 핀은 세로 8줄과 대응 된다. 차이점이 있다면 PORTE의 비트 '1'의 의미는 '켜다' 이고, PORTF의 비트 '1'의 의미는 '끄다'이다. 그렇기 때문에 세로줄의 불을 켜고 싶으면 NOT연산을 통해야 한다.

- 첫 번째 가로줄의 불을 켜고 그다음 두 번째 가로줄의 불을 켠다. 이를 8번째 가로줄의 불을 켜게 될 때 까지 진행되는 로직을 while문 안에 삽입하여 잔상 효과를 이용해서 우리 눈에는 총 64개의 LED등이 동시에 제어되는 것처럼 보이게 한다.

- 따라서 __oclock_dot이라는 byte 배열을 이용한다. byte는 8비트 크기의 자료구조이고 이 byte를 size가 8인 배열로 선언하면 이를 8X8 도트매트릭스에 대응 하는 것은 어렵지 않을 것이다. 켜고 싶은 LED의 비트를 1로 하고 끄고 싶은 LED의 비트를 0으로 한 8개의 byte 배열을 통해서 Dot Matrix에 불이 들어오게 한다.

```
void display_clock_to_dot(){
    unsigned int i=0;
    if(++i == 8){
        i=0;
    }
    PORTE = 0x80 >>i;
    PORTF = ~__oclock_dot[0][i];
}
```

- 시계를 나타내기 위해 미리 선언된 byte array

- 시, 10분, 분, 오전/오후 등등 시간을 나타내는 여러 가지 요소들을 나누어서 byte 2차원 배열로 미리 저장하고 있다. 시간을 나타내는 모든 요소들을 OR 연산한 Byte array를 위의 display_clock_to_dot() 함수로 보내면 우리는 Dot Matrix에 시간이 표시되는 것을 볼 수 있다. 아래에는 시와 10분단위를 나타내는 array들이 있다.

```
unsigned char __oclock_dot[12][8] = {
    {0x00, 0x00, 0x44, 0x00, 0x00, 0x00, 0x00, 0x00}, // 1시
    {0x00, 0x00, 0x24, 0x00, 0x00, 0x00, 0x00, 0x00}, // 2시
    {0x00, 0x00, 0x14, 0x00, 0x00, 0x00, 0x00, 0x00}, // 3시
    {0x00, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x00, 0x00}, // 4시
    {0x00, 0x60, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00}, // 5시
    {0x20, 0x20, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00}, // 6시
    {0x08, 0x08, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00}, // 7시
    {0x30, 0x00, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00}, // 8시
    {0x04, 0x04, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00}, // 9시
    {0x00, 0x10, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00}, // 10시
    {0x00, 0x10, 0x44, 0x00, 0x00, 0x00, 0x00, 0x00}, // 11시
    {0x00, 0x10, 0x24, 0x00, 0x00, 0x00, 0x00, 0x00} // 12시
};

unsigned char __ten_min_dot[6][8] = {
    {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, // 00분
    {0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x00},  // 10분
    {0x00, 0x00, 0x00, 0x44, 0x00, 0x00, 0x00, 0x00}, // 20분
    {0x00, 0x00, 0x00, 0x24, 0x00, 0x00, 0x00, 0x00}, // 30분
    {0x00, 0x00, 0x00, 0x14, 0x00, 0x00, 0x00, 0x00}, // 40분
    {0x00, 0x00, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x00}// 50분
};
```

- 외부 인터럽트 작성

- D 포트에서 사용할 수 있는 인터럽트 4개를 스위치 보드의 버튼 4개와 대응 시켜서 스위치가 눌렸을 때 하게 되는 인터럽트 서비스 루틴을 작성하였다. 인터럽트에서는 많은 작업을 하지 않고 Mode를 바꾸는 정도의 아주 간단한 작업만 하도록 하였다.

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

#define ON 1
#define OFF 0

#define NORMAL_MODE 1
#define SETTING_MODE 2
#define ALARM_SETTING_MODE 3

volatile int mode = NORMAL_MODE;
volatile int stop_watch_mode = OFF;
volatile int is_stop = OFF;
volatile int is_alarm = ON;
volatile int is_time_save = ON;

volatile int stop_cur_time = 0;
volatile int stop_stop_time = 0;

SIGNAL(SIG_INTERRUPT0){
...
}

SIGNAL(SIG_INTERRUPT1){
    //interrupt1 : mode change NORMAL -> SETTING -> ALARM_SETTING
    if(mode == NORMAL_MODE){
        mode = SETTING_MODE;
        if(stop_watch_mode == ON){
            stop_watch_mode = OFF;
            stop_cur_time = 0;
            stop_stop_time = 0;
        }
    }
    else if(mode == SETTING_MODE){
        mode = ALARM_SETTING_MODE;
    }
    else if(mode == ALARM_SETTING_MODE){
        mode = NORMAL_MODE;
    }
}

SIGNAL(SIG_INTERRUPT2){
...
}

SIGNAL(SIG_INTERRUPT3){
...
}
```

- RTC 모듈로 시간을 읽어오는 부분

- RTC 모듈을 판매하는 곳에서 헤더 파일을 공개 하였다. 7칸의 byte 배열을 파라미터로 넘기면 아래 ds1302_read_time 함수에 나와있는 순서대로 년도부터 초단위의 시간 까지 받아온다. byte 변수에 값이 저장되는데 이를 알맞게 parsing을 해야 원하는 시간을 얻을 수 있다. 예를 들어서 min이 0x21 이라는 값이라면 16진수를 10진수로 바꾸면 0x21 = 33 이 된다. 하지만 0x21는 33분이 아니다. 0x21은 21분이다. 그렇기 때문에 byte 변수에서 10의 자리수와 1의 자리수를 따로 빼서 10진수로 합쳐야 한다.

```
void ds1302_read_time (byte set_time[])
{
    set_time[0] = ds1302_read(ds1302_year_add);  // years
    set_time[1] = ds1302_read(ds1302_month_add); // month
    set_time[2] = ds1302_read(ds1302_date_add); // day
    set_time[6] = ds1302_read(ds1302_day_add); // weeks
    set_time[3] = ds1302_read(ds1302_hr_add); // time
    set_time[4] = ds1302_read(ds1302_min_add); // min
    set_time[5] = ds1302_read(ds1302_sec_add); // sec
}

byte ds1302_read(byte add)
{
    byte i=0,data=0;

    add+=1;                  // read flag
    set_ds1302_io_ddr();     // output ports
    delay_us(2);
    clr_ds1302_rst();        // clear reset
    delay_us(2);
    clr_ds1302_sclk();       // clear clock
    delay_us(2);
    set_ds1302_rst();        // Set Reset
    delay_us(2);

    for(i=8;i>0;i--)         // This loop writes the address code
    {
        if(add&0x01)
        {
          set_ds1302_io();
        }
        else
        {
        clr_ds1302_io();
        }

        delay_us(2);
        set_ds1302_sclk();
        delay_us(2);
        clr_ds1302_sclk();
        delay_us(2);
        add>>=1;
    }
    clr_ds1302_io_ddr();      // input port
    delay_us(2);
    for(i=8;i>0;i--)         // This loop reads data from 1302
```

```
    {
        data>>=1;
        if(in_ds1302_io())
        {
         data|=0x80;
        }
        delay_us(2);
        set_ds1302_sclk();
        delay_us(2);
        clr_ds1302_sclk();
        delay_us(2);
    }

    clr_ds1302_rst();
    delay_us(2);

    return(data);

}
```

- 시간을 읽어서 Dot Matrix에 나태날 8개의 byte array를 만드는 함수

- 위에서 시간을 읽어 왔다면 Dot Matrix에 표시될 비트 배열을 만들어야한다. 원리는 매우 간단하다. 오전/오후, 시, 분을 파악해서 각각의 요소에 맞는 비트를 OR 연산하면 된다. 초시계를 나타내야 할 때에는 마찬가지로 초를 나타내는 비트를 OR연산하면 된다. 이렇게 모든 OR연산을 끝낸 Byte Array는 위의 display_clock_to_dot 함수로 보내서 Dot Matrix에 시간을 표시한다.

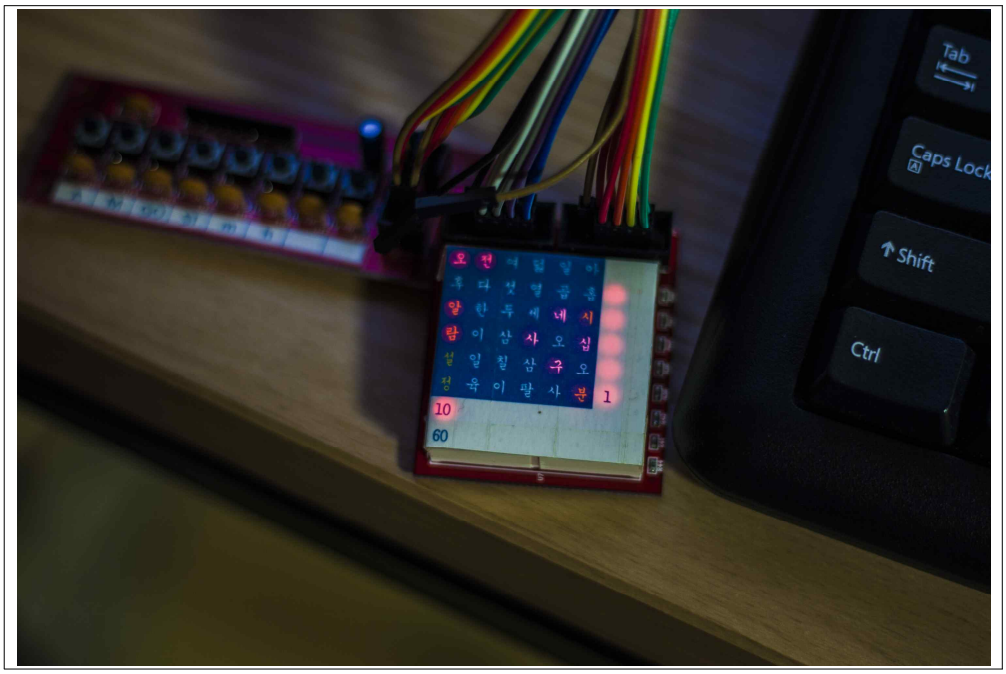
```
void __get_clock_dot(USHORT time, USHORT min){
    int hour, ten_min, one_min;
    hour = ((time &0xf0)>>4)*10;
    hour = hour + (time &0x0f);
    ten_min = (min &0xf0)>>4;
    one_min = min &0x0f;

    if(hour >=12){
        __get_or_oper(after_noon_dot);
    }
    else{
        __get_or_oper(before_noon_dot);
    }

    if(hour == 0){
        hour += 12;
    }
    else if(hour > 12){
        hour -= 12;
    }

    __get_or_oper(__oclock_dot[hour-1]);

    if(ten_min == 0 && one_min == 0){
    }
    else{
        __get_or_oper(__ten_min_dot[ten_min]);
```

```
        __get_or_oper(__min_dot[one_min]);
    }

    if(mode == NORMAL_MODE &&is_alarm){
        __get_or_oper(__alarm_dot[0]);
    }
    if(mode == SETTING_MODE){
        __get_or_oper(__setting_dot[0]);
    }
    if(mode == ALARM_SETTING_MODE){
        __get_or_oper(__setting_dot[0]);
        __get_or_oper(__alarm_dot[0]);
    }

    if(stop_watch_mode == ON){
        if(is_stop == OFF){
            __get_or_oper(__stop_watch_sec[(stop_cur_time%60)%10]);
            __get_or_oper(__stop_watch_10sec[(stop_cur_time%60)/10]);
            __get_or_oper(__stop_watch_min[stop_cur_time/60]);
        }
        else if(is_stop == ON){
            __get_or_oper(__stop_watch_sec[(stop_stop_time%60)%10]);
            __get_or_oper(__stop_watch_10sec[(stop_stop_time%60)/10]);
            __get_or_oper(__stop_watch_min[stop_stop_time/60]);
        }
    }

}
```

8 실물 사진

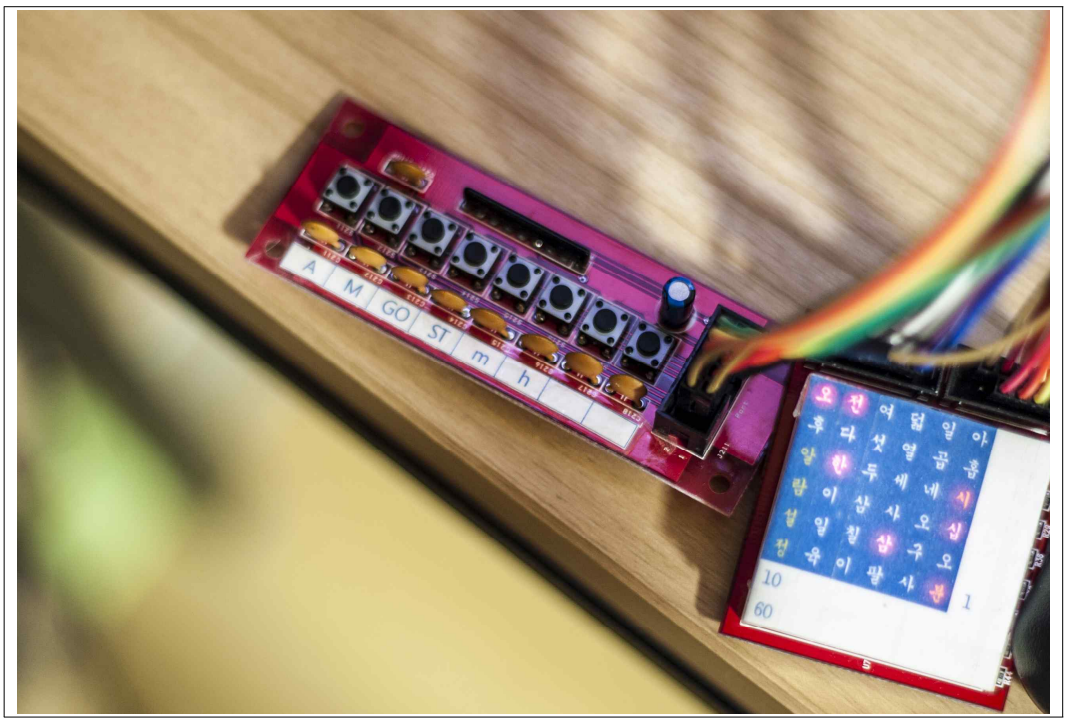
- 전체 모습

한글 시계의 스위치 보드와 시간을 나타내는 한글 기판의 사진이다. 시간은 물론 스탑워치도 실행되고 있는 모습이다. 사진 속 시계는 오전 4시 49분을 가르키고 있고 알람이 켜져 있으며, 스탑워치는 15초를 지나고 있다.



- 스위치 보드

8개의 스위치가 있는 스위치 보드의 사진이다. 8개중 6개의 스위치만을 사용한다.



- 빛이 없을때의 한글 시계

불이 꺼졌을 때의 한글시계가 작동하는 모습이다. 마찬가지로 시계는 오전 4시 52분을 나타내고 있고 알람이 켜져 있다. 스탑워치는 2분 58초를 나태나고 있다.

⑨부품 견적

No.	제품명	상품 코드	단가 (원)	수량	합계 (원)
1	CR1220-BP(3V)Ma	4188	1,000	1	1,000
2	RTC DS1302 모듈	36972	6,000	1	6,000
3	도트매트릭스 Dot Matrix 8x8 적색 모듈	36972	8,000	1	8,000
4	스위치 보드(AM-TS8)	11701	8,000	1	8,000
5	테스트[CH254]소켓 점퍼 케이블 40P (칼라) (F/F)	32284	4,800	1	4,800

상품 가격	27,800 원
부가세	2,780 원
견적 총액	30,580 원