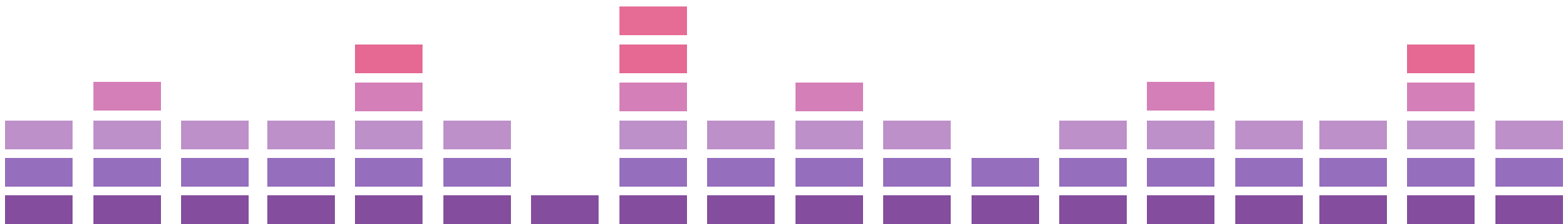
정교하고 독립적으로 사용 가능한

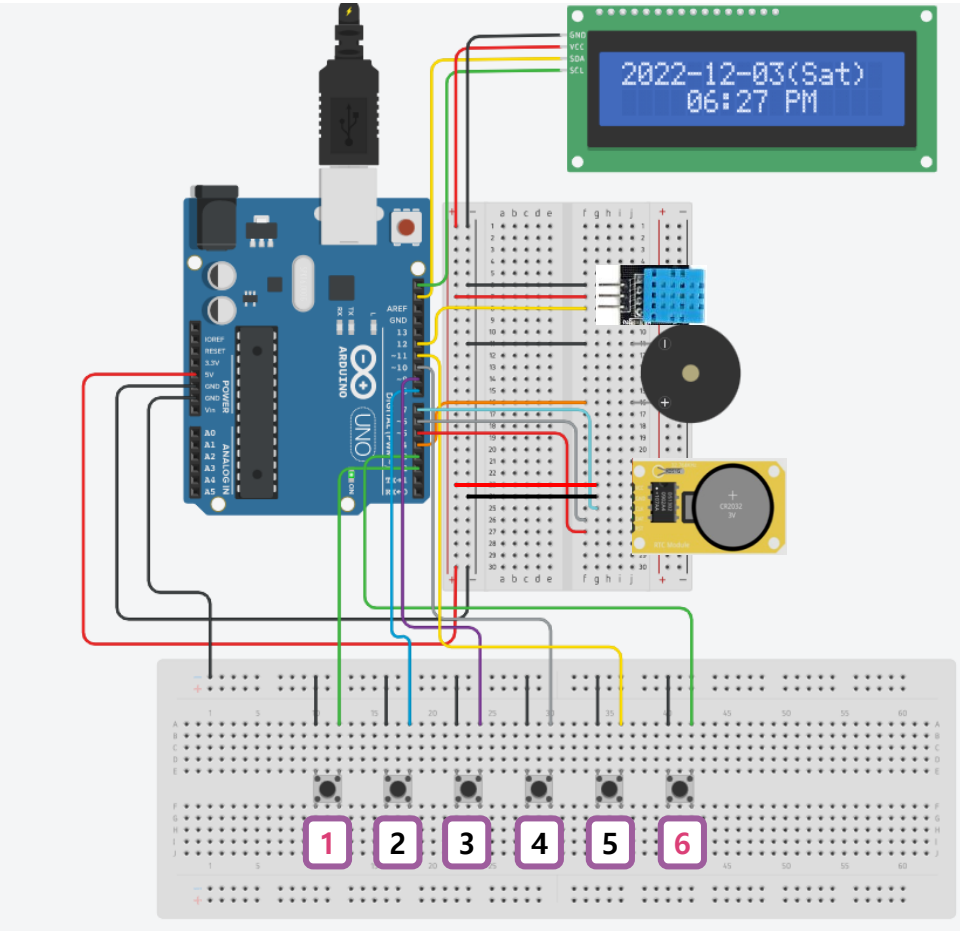
아두이노 알람시계 제작

2089055 조창희



목차

회로 구성



부품명	개수
아두이노 보드	1개
브래드보드	2개
CLCD(직렬타입)	1개
DHT11	1개
능동부저	1개
DS1302(RTC)	1개
버튼	6개

1. 모드 변경(인터럽트 0번)
2. 시간/연도 수정
3. 분/ 월 수정
4. AMPM/ 일 수정
5. 활성화 / 수정반영 버튼
6. 알람 종료(인터럽트 1번)

인터럽트를 제외한 버튼은
일반적인 작업 순서에 맞게
지정하여 직관적이고,
어렵지 않게 구성함

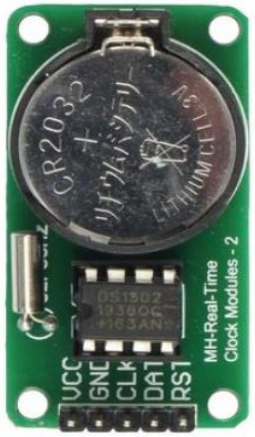
사용 모듈 소개

DHT11 온도/습도 센서

온도와 습도를 동시에 측정하는 센서

- 습도 측정 범위 : 20~90%(오차±5%)
- 온도 측정범위 : 0~50℃(오차±2 ℃)
- 사용 라이브러리 : SimpleDHT.h

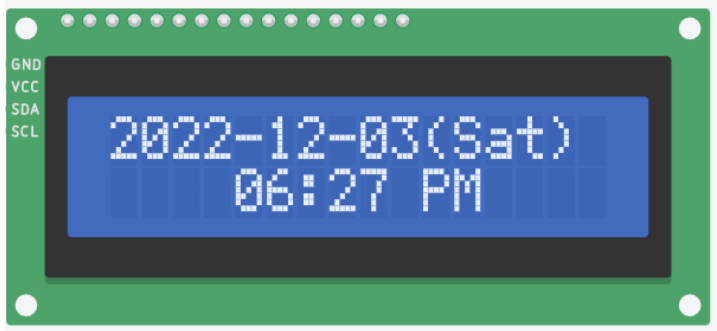
DS1302 RTC 모듈

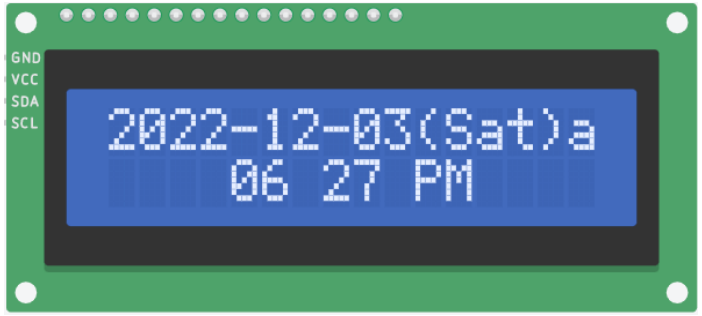


시계의 역할을 하는 모듈

- 전원을 끄더라도 모듈 자체의 배터리로 시간이 계속 흐름
- 시간 정확도가 높음(QUARTZ)
- 사용 라이브러리 : RtcDS1302.h

작품 소개

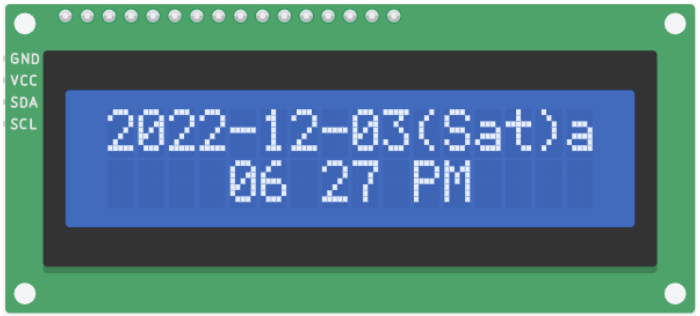




- 초기 화면 상단에 날짜와 요일, 하단에 시간을 표시하고 있어 명확한 시간 확인 가능
- 1초마다 시간 사이의 : 점멸하고 있어 시계 정상 작동 여부 확인 가능
- 온·습도 기능이 내장되어 있어 현재의 온도와 습도를 확인할 수 있음
- 버튼으로 시간 설정이 가능해, 컴퓨터를 연결하지 않아도 정상적으로 작동함
- 알람 설정 기능이 있어 설정해 놓으면 지정된 시간마다 소리로 알려줌
  (알람이 활성화 되면 시계모드에서 오른쪽 상단에 a를 표시해 줌)
- 어느 모드에 있던지 활성화된 알람이 있으면 알람이 울리고, 바로 끌 수 있음

모드 설명

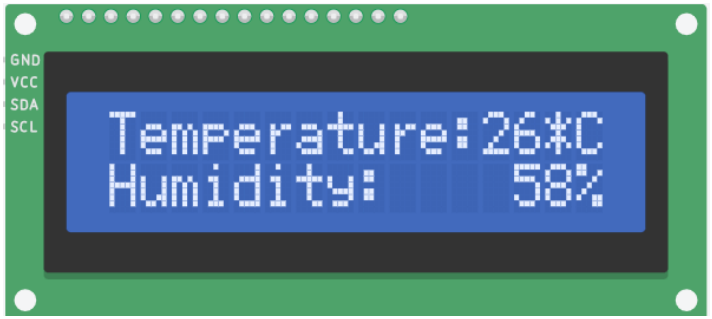
Mode 1 : 날짜 및 시간 표시



- 1차적으로 컴파일을 진행한다면 자동으로 컴파일 했을 당시 컴퓨터의 날짜와 시간이 저장됨
- 추후 설계된 장치의 버튼을 이용하여 날짜와 시간을 변경할 수 있음
- 시간 사이의 ":"은 매 1초마다 점멸하고 있어 시계가 정상적으로 작동하는 지 확인 가능
- 만약, 알람이 설정되어 있다면, 요일 옆에 "a"라고 표시함으로써 알람이 설정된 상태를 확인 가능

모드 설명

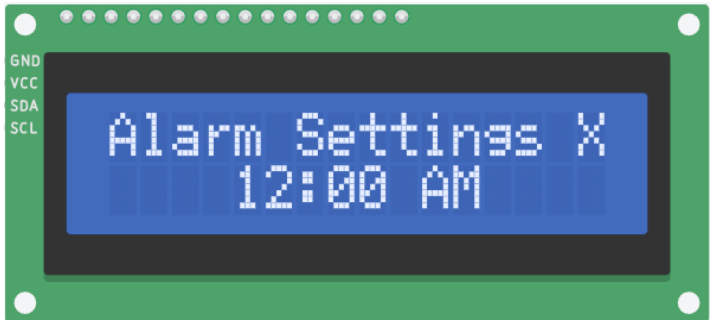
Mode 2 : 온도 및 습도 표시

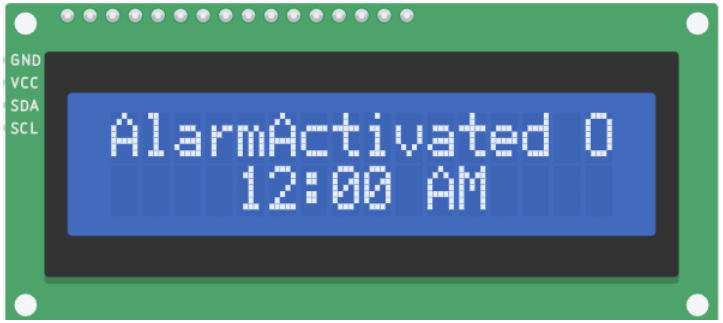


- DHT11 센서가 달려 있기에 이 모드에 진입하게 되면 1.5초 간격으로 현재 상태를 알려줌
- TinkerCad의 특성 상, 26*C라고 표시했지만, 실제 화면에서는 ℃로 잘 나옴

모드 설명

Mode 3 : 알람 설정 기능

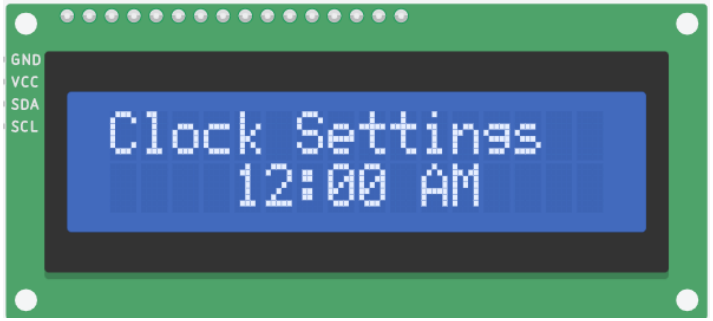




- 알람이 비활성화 되어 있다면, 그림 1과 같이 오른쪽 상단에 x가 나오고 알람 시간 설정도 가능함
- 알람이 활성화되면 그림 2와 같이 오른쪽 상단에 O 표시가 나오고 알람 시간 수정 불가능함
  알람 시간을 변경하고 싶다면 비활성화 후 알람 시간을 변경해야 함
- 알람이 활성화되면 초기 모드(모드 1) 화면 오른쪽 상단에 a 표시가 뜸
- 알람이 활성화 되면 어느 모드에 있는 지정한 시간이 되었을 때 알람이 발생하고, 어느 곳에서든
  알람 종료 버튼을 누르면 바로 끌 수 있음

모드 설명

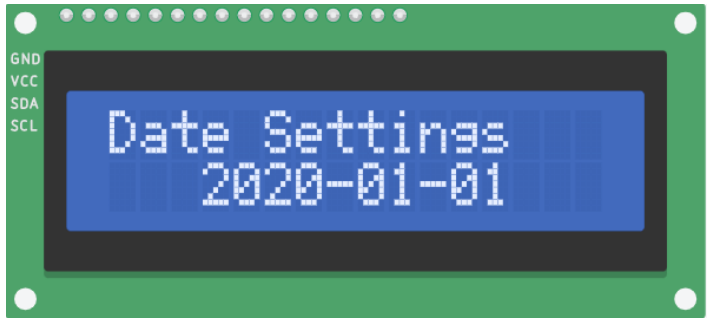
Mode 4 : 현재 시간 변경

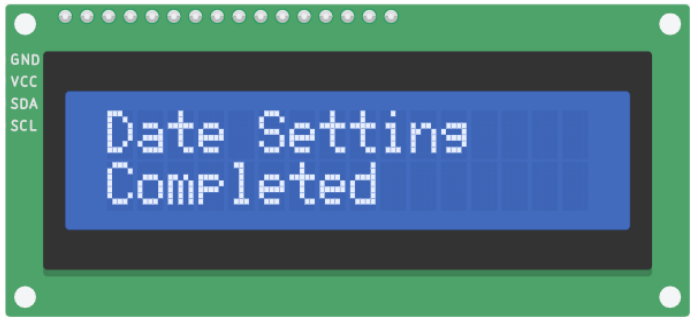


- 현재 시간을 바꾸고 싶다면 버튼을 이용해서 시간 변경이 가능함(2번:시, 3번:분: 4번:AM/PM)
- 초는 저장(5번 버튼)을 누르면 0초로 지정해서 저장됨
- 변경이 완료되면 그림2와 같이 2초간 뜨고 초기화면(모드 1)로 화면이 전환됨
- 변경된 시간에 설정된 알람이 있다면 바로 울리게 됨

모드 설명

Mode 5 : 현재 날짜 변경

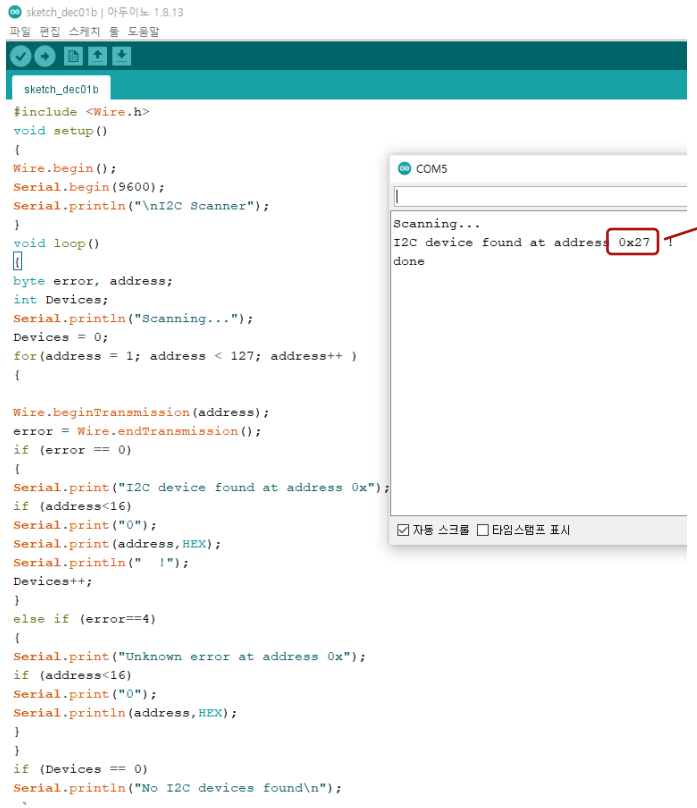




- 현재 날짜도 버튼을 이용해서 변경을 할 수 있음. 요일은 RTC모듈이 자동으로 계산해 지정됨
- 현재 시간을 바꾸고 싶다면 버튼을 이용해서 시간 변경이 가능함(2번:년, 3번:월: 4번:일)
- 변경이 완료되면 그림2와 같이 2초간 뜨고 초기화면(모드 1)로 화면이 전환됨

코드 설명(시작하기에 앞서)

CLCD 주소 찾기




```
LiquidCrystal_I2C lcd(0x27,16,2);
```

이번에 사용하는 직렬 CLCD는 코드에 CLCD의 주소값을 입력해야 하기에 시작하기에 앞서

LMS에 올라와 있는 I2C 주소 찾기 코드를 이용해 주소값을 찾아 주었다.

코드 설명(라이브러리 및 변수)

```
#define BOUNCE_TIME 190
#include <LiquidCrystal_I2C.h>
#include <SimpleDHT.h>
#include <MsTimer2.h>
#include <ThreeWire.h>
#include <RtcDS1302.h>

// HW핀 현황
int pinDHT11 = 12;
int mode_btn = 0; //아두이노 2번 핀
int alert_btn = 1; //아두이노 3번 핀
int btn1 = 8;
int btn2 = 9;
int btn3 = 10;
int btn4 = 11;
int buz = 4;
// CLK : 6번 핀
// DAT : 5번 핀
// RST : 7번 핀

char dow[7][4]={"Sun","Mon","Tue","Wed","Thu","Fri","Sat"};

// 알람용 변수
int a_hour = 0;
int a_min = 0;
int a_now = 0;//low:am high:pm
int a_status = LOW;
int a_hour_t = 0;
```

```
// 알람용 변수
int a_hour = 0;
int a_min = 0;
int a_now = 0;//low:am high:pm
int a_status = LOW;
int a_hour_t = 0;

// 시간변경용 변수
int c_hour = 0;
int c_hour_t = 0;
int c_min = 0;
int c_now = LOW;

// 날짜변경용 변수
int d_year = 2020;
int d_month = 1;
int d_date = 1;


volatile int alert_status=LOW;
volatile int changed = LOW;
volatile int mode_num = 0;
volatile unsigned long last_int_time1=0;
volatile unsigned long last_int_time2=0;
```

코드 설명(MSTimer2)

```
LiquidCrystal_I2C lcd(0x27,16,2);
SimpleDHT11 dht11(pinDHT11);
ThreeWire three_wire(5, 6, 7);
RtcDS1302<ThreeWire> Rtc(three_wire);

void alert() //MSTimer2 사용
{
  RtcDateTime now = Rtc.GetDateTime();
  int hour = now.Hour(); // 시
  int min = now.Minute(); // 분
  static int beep_status=LOW;
  if(a_status==HIGH && hour==a_hour && min==a_min && beep_status==LOW && alert_status==HIGH)
  {
    digitalWrite(buz,HIGH);
    beep_status=HIGH;
  }
  else
  {
    digitalWrite(buz,LOW);
    beep_status=LOW;
  }
  if(alert_status==LOW && hour>=a_hour && min>a_min && a_status==HIGH)
    alert_status=HIGH;
}
```

타이머 인터럽트로 알람 설정 현황을 체크!
어느 모드에 있던지 알람을 울리게 함!

코드 설명(setup, loop)

```
void setup() {
  // RTC 모듈 시작
  Rtc.Begin();

  // 현재 시간 설정
  Rtc.SetDateTime(RtcDateTime(__DATE__, __TIME__));

  // RTC 모듈에 쓰기 금지 모드인 경우
  if (Rtc.GetIsWriteProtected()) {
    Rtc.SetIsWriteProtected(false); // 쓰기 금지 모드 해제
  }

  // RTC 모듈이 동작중이 아닌 경우
  if (!Rtc.GetIsRunning()) {
    Rtc.SetIsRunning(true); // RTC 모듈 동작 시작
  }


  Serial.begin(9600);
  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);
  pinMode(btn1, INPUT_PULLUP);
  pinMode(btn2, INPUT_PULLUP);
  pinMode(btn3, INPUT_PULLUP);
  pinMode(btn4, INPUT_PULLUP);
  pinMode(buz, OUTPUT);
  lcd.init();
  lcd.backlight();
  MsTimer2::set(1000,alert);
  MsTimer2::start();
  attachInterrupt(mode_btn,change_mode,FALLING);
  attachInterrupt(alert_btn,alert_off,FALLING);
}
```

원래, 인터럽트 안에서 모드가 변경될 때마다
Lcd 초기화를 하고자 했으나,
인터럽트 안에서 lcd 명령 시 오류가 나기에, 조건을 걸어 실행
(너무 자주 초기화시 lcd가 무한히 반복하기에)

```
void loop() {
  if(changed == HIGH)
  {
    lcd.clear();
    changed=LOW;
  }
  switch(mode_num)
  {
    case 0:
      mode_1();
      break;
    case 1:
      mode_2();
      break;
    case 2:
      mode_3();
      break;
    case 3:
      mode_4();
      break;
    case 4:
      mode_5();
      break;
  }
}
```

0번 인터럽트 버튼을
누를 때마다 모드가
변경되는 구조

인터럽트 버튼을 풀업 내부저항을
이용했기에 같은 버튼이지만,
각각 번호가 틀리게 적용된 것을
볼 수 있음

코드 설명(blink, Interrupt)

```
void blink()
{
  static boolean colon_state=HIGH;
  if(colon_state == LOW)
  {
    lcd.print(":");
    colon_state=HIGH;
  }
  else
  {
    lcd.print(" ");
    colon_state = LOW;
  }
}
```

시계가 동작하고 있다는 의미에서 ":"을 사용하기에 mode_1이 동작할 때만 같이 작동함. (mode_1에서 추가 설명)

```
void change_mode()
{
  unsigned long int_time = millis();
  if(int_time - last_int_time1> BOUNCE_TIME)
  {
    mode_num++;
    changed = HIGH;
    if(mode_num == 5)
      mode_num = 0;
  }
  last_int_time1=int_time;
}
```

0번 인터럽트(모드 변경)

```
void alert_off()
{
  unsigned long int_time = millis();
  if(int_time - last_int_time2> BOUNCE_TIME)
  {
    RtcDateTime now = Rtc.GetDateTime();
    int hour = now.Hour(); // 시
    int min = now.Minute(); // 분
    if(alert_status==HIGH&hour==a_hour&&min==a_min)
      alert_status=LOW;
  }
  last_int_time2=int_time;
}
```

1번 인터럽트(알람 종료)

코드 설명(mode_1, 날짜 및 시간 표시)

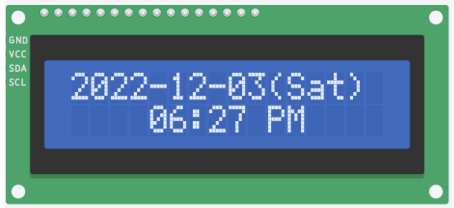
```
void mode_1()
{
  // RTC 모듈의 현재 시간 얻기
  RtcDateTime now = Rtc.GetDateTime();

  // 변수 선언
  int year = now.Year(); // 년
  int month = now.Month(); // 월
  int day = now.Day(); // 일
  int DayOfWeek = now.DayOfWeek(); // 요일
  int hour = now.Hour(); // 시
  int minute = now.Minute(); // 분
  int second = now.Second(); // 초
  lcd.setCursor(0,0);
  lcd.print(year);
  lcd.print("-");
  if(month<10)
    lcd.print("0");
  lcd.print(month);
  lcd.print("-");
  if(day<10)
    lcd.print("0");
  lcd.print(day);
  lcd.print("(");
  lcd.print(dow[DayOfWeek]);
  lcd.print(")");
  lcd.setCursor(4,1);
  if((hour<10||(hour>12&&hour<22))&&(hour!=0))
    lcd.print("0");
  if(hour>=13)
    lcd.print(hour-12);
  else if(hour==0)
    lcd.print("12");
  else
    lcd.print(hour);
  blink();
  if(minute<10)
    lcd.print("0");
  lcd.print(minute);
  if(hour>=12)
    lcd.print(" PM");
  else
    lcd.print(" AM");
  if(a_status==HIGH)
  {
    lcd.setCursor(15,0);
    lcd.print("a");
  }
  delay(1000);
}
```

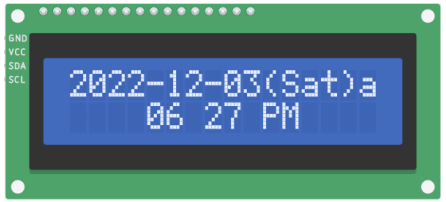
업로딩을 하면 1차적으로 컴퓨터 시간을 저장

앞서 blink 함수를 봤을 때,
상태변화는 있었지만, 몇 초마다
시행하라는 것이 없었음

시계는 초단위로 움직이기에
같이 움직일 수 있어서
그렇게 적용한 것임

RTC에서 시간은 0~23으로 표현되기에
13부터는 다시 01로 표현해 주고
AM대신 PM을 표시함



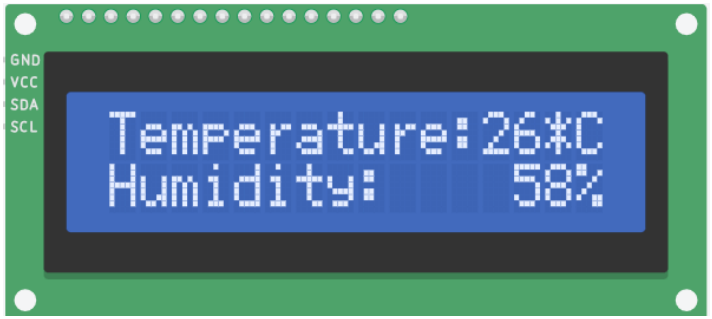


코드 설명(mode_2, 온도 및 습도 표시)

```
void mode_2()
{
  byte temperature = 0;
  byte humidity = 0;
  int err = SimpleDHTErrSuccess;
  if((err=dht11.read(&temperature, &humidity, NULL)) != SimpleDHTErrSuccess) {
      Serial.print("Read DHT11 failed, err="); Serial.print(SimpleDHTErrCode(err));
      Serial.print(","); Serial.println(SimpleDHTErrDuration(err)); delay(1000);
      return;
    } //DHT11 샘플링 실패시 시리얼 모니터에 출력
  lcd.setCursor(0,0);
  lcd.print("Temperature:");
  lcd.print((int)temperature); lcd.print((char)223); lcd.print("C");
  lcd.setCursor(0,1);
  lcd.print("Humidity:    ");
  lcd.print((int)humidity); lcd.print("%");
  delay(1500);
  // DHT11 sampling rate is 1HZ;
}
```

DHT11 작동 오류 시
시리얼 모니터에 오류코드 출력

라이브러리 주석에서도 볼 수 있지만, DHT11은 샘플링
할 수 있는 주기가 있기에 1.5초 delay를 줌(없으면 에러코드 발생)



코드 설명(mode_3-1, 알람 설정 기능)

```
void mode_3()
{
  if(a_status==LOW)
  {
    lcd.setCursor(0,0);
    lcd.print("Alarm Settings");
    lcd.setCursor(4,1);
    if(a_hour>0&&a_hour<10)
      lcd.print("0");
    else if(a_hour==0)
      lcd.print("12");
    if(a_hour!=0&&a_hour<=12)
      lcd.print(a_hour);
    if((a_hour>12))
    {
      if((a_hour-12)<10)
      {
        lcd.print("0");
        lcd.print(a_hour-12);
      }
      else if(a_hour==12)
        lcd.print("12");
      else
        lcd.print(a_hour-12);
    }
    lcd.print(":");
    if(a_min<10)
      lcd.print("0");
    lcd.print(a_min);
```

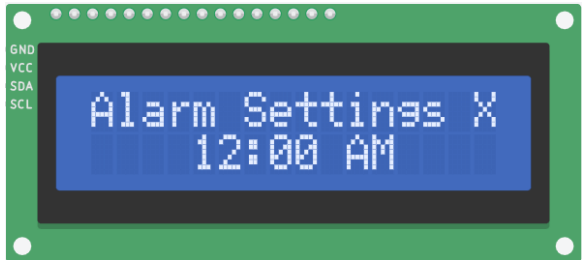
1. Btn1
   시간 증가
   (13이 되면 1이 됨,
   0는 12시로 표시,
   9 이하는 0을 앞에
   붙임)
2. btn2
   분 증가
   (61이 되면 0이 됨
   9 이하면 0을 앞에
   붙임)
3. btn3
   AM/PM 선택
   (누를 때마다 바뀜,
   PM이 선택된다면
   시간에다가 12 증가)

a_status가 LOW면
수정이 가능하다.
하지만, HIGH가 되면
수정할 수 없다.
이 경우, 버튼4
(다음장)로 status를
LOW로 바꿔줘야 한다.

```
if(a_now==HIGH)
  lcd.print(" PM");
else
  lcd.print(" AM");
lcd.setCursor(15,0);
lcd.print("X");
if(!digitalRead(btn1))
{
  a_hour_t++;
  if(a_hour_t==12)
    a_hour_t=0;
  lcd.setCursor(4,1);
  if(a_hour_t<10&&a_hour_t>0)
    lcd.print("0");
  if(a_hour_t==0)
    lcd.print("12");
  else
    lcd.print(a_hour_t);
  delay(300);
}
if(!digitalRead(btn2))
{
  a_min++;
  if(a_min==60)
    a_min=0;
  lcd.setCursor(7,1);
  if(a_min<10)
    lcd.print("0");
  lcd.print(a_min);
  delay(300);
}
```

```
if(!digitalRead(btn3))
{
  a_now = !a_now;
  if(a_now == LOW)
    lcd.print(" AM");
  else
    lcd.print(" PM");
  delay(300);
}
```

계속 loop를 돌기에 언젠가는 시간이 표시되어도 다시 표시를 해주는 이유는 스위치를 누름으로써 생길 수 있는 바운싱 현상을 방지해주기 위해 넣어준 delay가 lcd에 표시하는 시간도 늦추기 때문에 빨리 표시하기 위함이다.



코드 설명(mode_3-2, 알람 설정 기능)

```
if(!digitalRead(btn4))
{
  if(a_status==LOW)
  {
    alert_status=HIGH;
    a_status=HIGH;
  }
  else
  {
    alert_status=LOW;
    a_status=LOW;
  }
  delay(300);
}
if(a_now==LOW)
  a_hour=a_hour_t;
else
  a_hour=a_hour_t+12;
}
else
{
  lcd.setCursor(0,0);
  lcd.print("AlarmActivated");
  lcd.setCursor(4,1);
  if(a_hour>0&&a_hour<10)
    lcd.print("0");
  else if(a_hour==0)
    lcd.print("12");
  if(a_hour!=0&&a_hour<=12)
    lcd.print(a_hour);
```

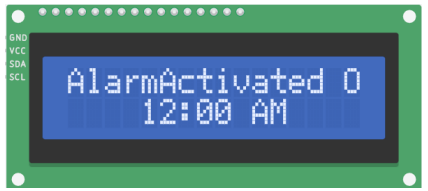
btn4: Alarm을 활성화한다. 활성화 되면, MsTimer2가 계속 시간을 비교하다가 시간이 되면 알람을 울릴 것이다.

알람 활성화가 되면, 수정 불가 알람 시간만 표시

```
if((a_hour>12))
{
  if((a_hour-12)<10)
  {
    lcd.print("0");
    lcd.print(a_hour-12);
  }
  else if(a_hour==12)
    lcd.print("12");
  else
    lcd.print(a_hour-12);
}
lcd.print(":");
if(a_min<10)
  lcd.print("0");
lcd.print(a_min);
if(a_now==HIGH)
  lcd.print(" PM");
else
  lcd.print(" AM");
lcd.setCursor(15,0);
lcd.print("O");
  if(!digitalRead(btn4))
  {
    a_status=LOW;
    lcd.init();
    lcd.setCursor(0,0);
    lcd.print("Alarm Settings");
    lcd.setCursor(4,1);
    if(a_hour>0&&a_hour<10)
      lcd.print("0");
    else if(a_hour==0)
      lcd.print("12");
```

btn4를 한번 더 누르면 알람 활성화를 취소한다.

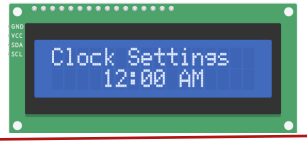
```
  if(a_hour>0&&a_hour<10)
    lcd.print("0");
  else if(a_hour==0)
    lcd.print("12");
  if(a_hour!=0&&a_hour<=12)
   lcd.print(a_hour);
  if((a_hour>12))
  {
    if((a_hour-12)<10)
    {
      lcd.print("0");
      lcd.print(a_hour-12);
    }
    else if(a_hour==12)
      lcd.print("12");
    else
      lcd.print(a_hour-12);
  }
  lcd.print(":");
  if(a_min<10)
    lcd.print("0");
  lcd.print(a_min);
  if(a_now==HIGH)
    lcd.print(" PM");
  else
    lcd.print(" AM");
  lcd.setCursor(15,0);
  lcd.print("X");
  delay(300);
}
```



코드 설명(mode_4, 현재 시간 변경)

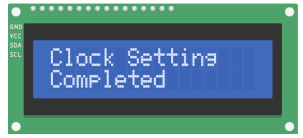
```
void mode_4()
{
  lcd.setCursor(0,0);
  lcd.print("Clock Settings")
    lcd.setCursor(4,1);
  if(c_hour>0&&c_hour<10)
    lcd.print("0");
  else if(c_hour==0)
    lcd.print("12");
  if(c_hour!=0&&c_hour<=12)
    lcd.print(c_hour);
  if((c_hour>12))
  {
    if((c_hour-12)<10)
    {
      lcd.print("0");
      lcd.print(c_hour-12);
    }
    else if(c_hour==12)
      lcd.print("12");
    else
      lcd.print(c_hour-12);
  }
  lcd.print(":");
  if(c_min<10)
    lcd.print("0");
  lcd.print(c_min);
  if(c_now==HIGH)
    lcd.print(" PM");
  else
    lcd.print(" AM");
```

1. Btn1
   시간 증가
   (13이 되면 1이 됨,
   0는 12시로 표시,
   9 이하는 0을 앞에 붙임)
2. btn2
   분 증가
   (61이 되면 0이 됨
   9 이하면 0을 앞에 붙임)
3. btn3
   AM/PM 선택
   (누를 때마다 바뀜,
   PM이 선택된다면
   시간에다가 12 증가)




```
if(!digitalRead(btn1))
{
  c_hour++;
  if(c_hour==12)
    c_hour=0;
  lcd.setCursor(4,1);
  if(c_hour<10&&c_hour>0
    lcd.print("0");
  if(c_hour==0)
    lcd.print("12");
  else
    lcd.print(c_hour);
  delay(300);
}
if(!digitalRead(btn2))
{
  c_min++;
  if(c_min==60)
    c_min=0;
  lcd.setCursor(7,1);
  if(c_min<10)
    lcd.print("0");
  lcd.print(c_min);
  delay(300);
}
if(!digitalRead(btn3))
{
  c_now = !c_now;
  lcd.setCursor(10,1);
  if(c_now == LOW)
    lcd.print("AM");
  else
    lcd.print("PM");
  delay(300);
}
```

```
if(!digitalRead(btn4))
{
  if(c_now==HIGH)
    c_hour+=12;
  RtcDateTime now = Rtc.GetDateTime();
  int year = now.Year(); // 년
  int month = now.Month(); // 월
  int day = now.Day(); // 일
  Rtc.SetDateTime(RtcDateTime(year,month,day,c_hour,c_min,0));
  lcd.init();
  lcd.setCursor(0,0);
  lcd.print("Clock Setting");
  lcd.setCursor(0,1);
  lcd.print("Completed");
  c_hour=0;
  c_min=0;
  c_hour_t=LOW;
  delay(2000);
  mode_num=0;
  lcd.init();
}
}
```



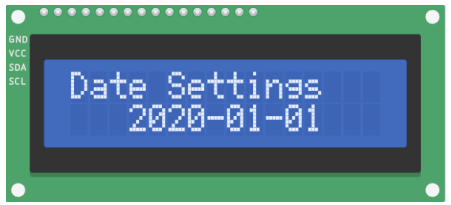
btn4를 누르면 지정한 시간으로
RTC time이 재설정 됨

설정 후 Clock Setting Completed을
띄우고 2초 후 초기화면(모드1)로 돌아감

계속 loop를 돌기에 언젠가는 시간이 표시되어도
다시 표시를 해주는 이유는 스위치를 누름으로써 생길 수 있는
바운싱 현상을 방지해주기 위해 넣어준 delay가 lcd에 표시하는
시간도 늦추기 때문에 빨리 표시하기 위함이다.

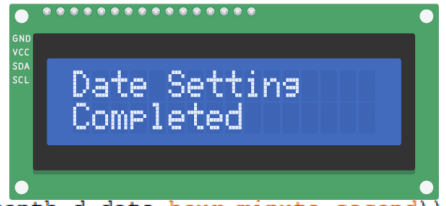
코드 설명(mode_5, 현재 날짜 변경)

```
void mode_5()
{
  lcd.setCursor(0,0);
  lcd.print("Date Settings");
  lcd.setCursor(3,1);
  lcd.print(d_year);
  lcd.print("-");
  if(d_month<10)
    lcd.print("0");
  lcd.print(d_month);
  lcd.print("-");
  if(d_date<10)
    lcd.print("0");
  lcd.print(d_date);
  if(!digitalRead(btn1))
  {
    d_year++;
    if(d_year==2031)
      d_year=2020;
    lcd.setCursor(3,1);
    lcd.print(d_year);
    delay(300);
  }
```




```
  if(!digitalRead(btn2))
  {
    d_month++;
    if(d_month==13)
      d_month=1;
    lcd.setCursor(8,1);
    if(d_month<10)
      lcd.print("0");
    lcd.print(d_month);
    delay(300);
  }
  if(!digitalRead(btn3))
  {
    d_date++;
    if(d_date==32)
      d_date=1;
    lcd.setCursor(11,1);
    if(d_date<10)
      lcd.print("0");
    lcd.print(d_date);
    delay(300);
```

```
  if(!digitalRead(btn4))
  {
    RtcDateTime now = Rtc.GetDateTime();
    int hour = now.Hour(); // 시
    int minute = now.Minute(); // 분
    int second = now.Second(); // 초
    Rtc.SetDateTime(RtcDateTime(d_year,d_month,d_date,hour,minute,second));
    lcd.init();
    lcd.setCursor(0,0);
    lcd.print("Date Setting");
    lcd.setCursor(0,1);
    lcd.print("Completed");
    d_year=2020;
    d_month=1;
    d_date=1;
    delay(2000);
    mode_num=0;
    lcd.init();
  }
}
```



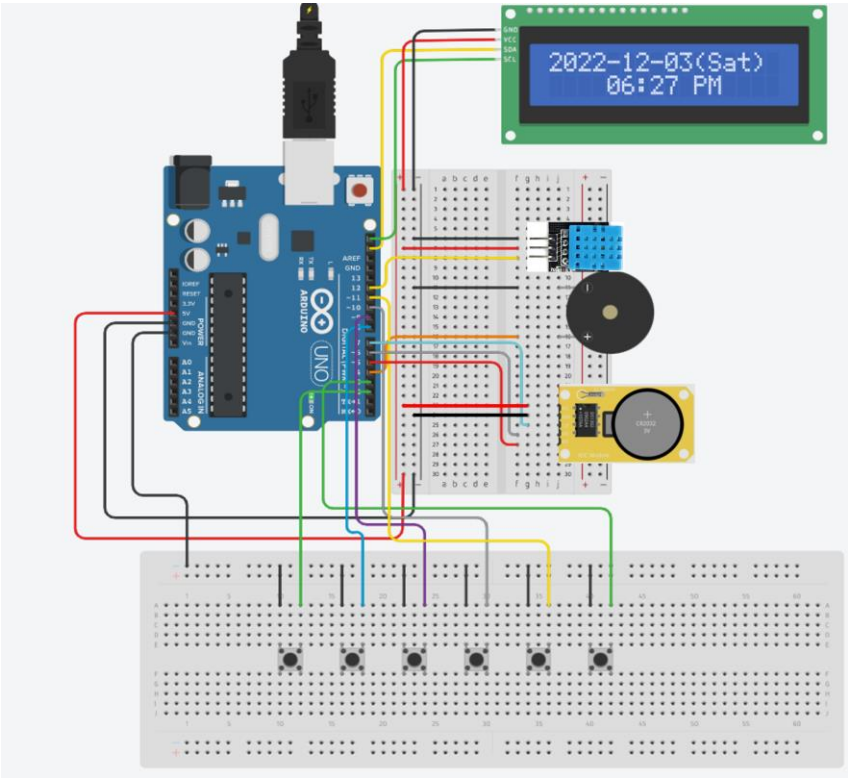
btn4를 누르면 지정한 날짜로
RTC time이 재설정 됨

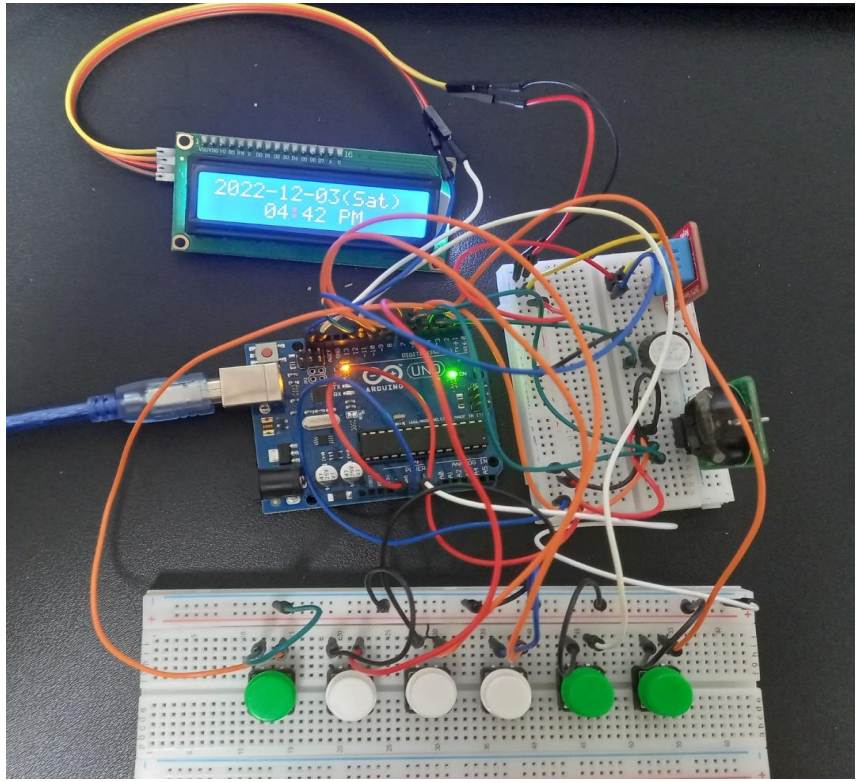
설정 후 Date Setting Completed을
띄우고 2초 후 초기화면(모드1)로 돌아감

RTC의 시간을 조정할 때, SetDateTime(RtcDateTime())을 사용하는데, 이때 년, 월, 일, 시, 분, 초를 다 입력해야 한다. Mode_4와 마찬가지로, 받지 못하는 수는 현재 RTC의 날짜를 먼저 호출에 다시 적용한다.

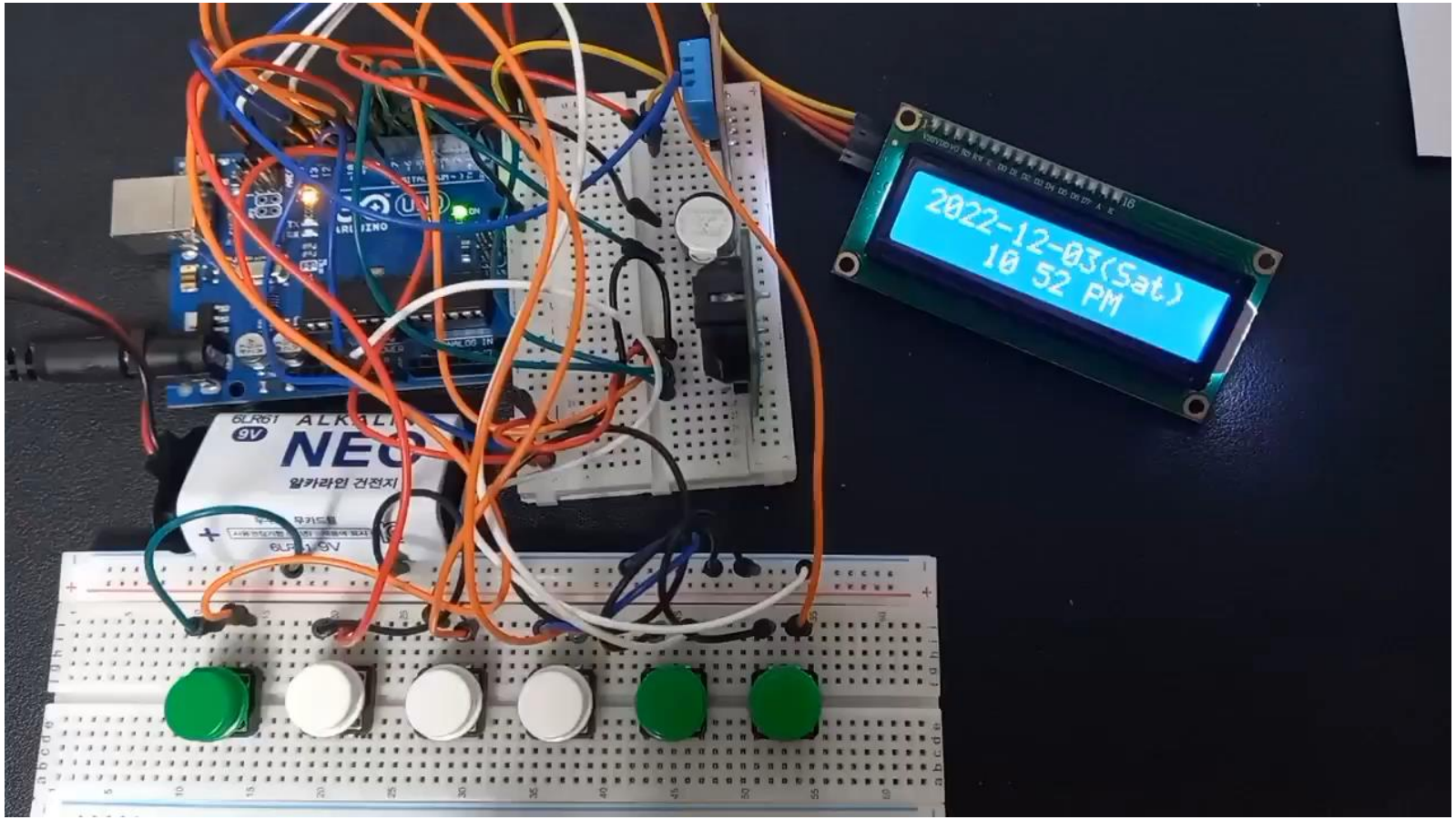
요일은 Rtc에 날짜를 설정해 주면 자동으로 요일이 계산된다.
이때, 요일은 숫자로 반환되는데, 앞에서 배열을 선언해 미리 작성해 주어 인덱스 번호에 맞는 요일을 호출한다.

실제 적용 모습





실제 적용 모습



감사합니다