

5 차시

아두이노 시계만들기

학습목표

- 전원이 꺼져도 시간이 유지되어 동작할 수 있도록 한다.

목차

1. 준비물
2. 하드웨어
3. 소프트웨어

준비물

- 아두이노 우노
- DS3231 RTC 모듈
- SSD1306 OLED 모듈
- 점퍼선

하드웨어

하드웨어

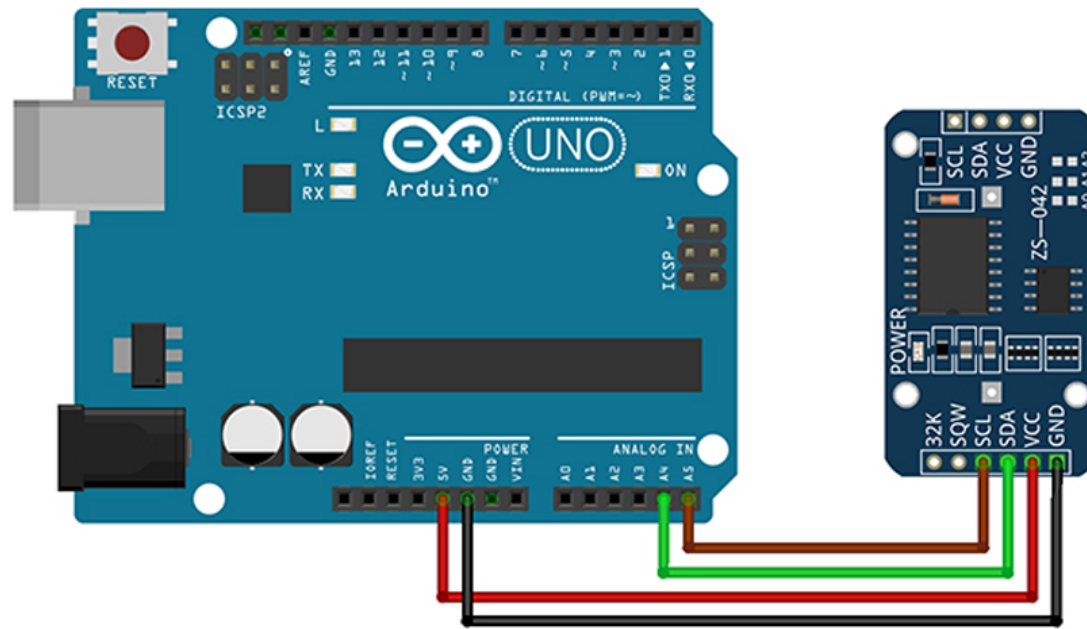
작업순서

1. RTC 연결
2. OLED 연결

하드웨어

RTC 연결회로

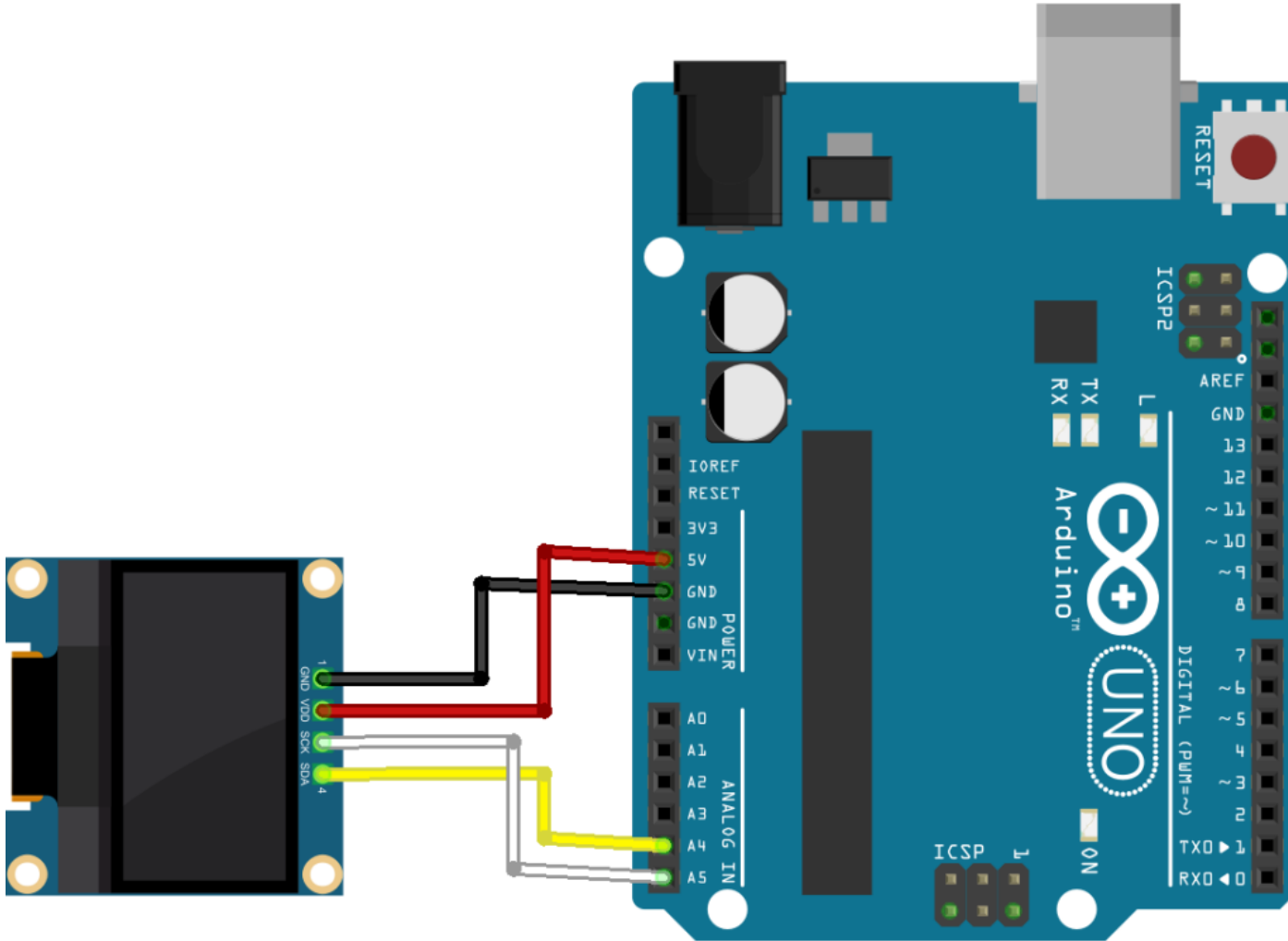
DS3231 RTC Module Arduino Circuit



For Complete Details Visit :
www.Circuits-DIY.com

하드웨어

OLED 연결회로



소프트웨어

소프트웨어

작업순서

1. RTC로 현재 날짜와 시간 세팅하기
2. RTC로 시간을 시리얼을 출력하기
3. OLED로 문자 출력하기
4. OLED로 날짜와 시간 출력하기

소프트웨어

I2C 주소 검색

```
from machine import Pin
from maine import I2C

sdaPIN = Pin(8)      # 데이터 핀
sclPIN = Pin(9)      # 클럭 핀

i2c = I2C(0, sda=sdaPin, scl=sclPin)    # 0번 I2C 포트 사용

device = i2c.scan()    # 주소검색

if len(devices) == 0:
    print('* No I2C device!')
else:
    print('* I2C devices found :', len(devices))

for device in devices:
    print(" => HEX address: ", hex(device))    # 슬레이브 주소 출력
```

네트워크에 접속하여 RTC 시간 세팅

라이브러리

```
import network
from simpletest.mywifi import networksetting
from machine import RTC
from machine import Pin
from machine import I2C
import utime as time
import usocket as socket
import ustruct as struct
from ds3231_port import DS3231
```

네트워크에 접속하여 RTC 시간 세팅

와이파이 설정

```
ssid, password = networksetting()
```

GMT 시간을 참조

```
# wintertime / Summerzeit
#GMT_OFFSET = 3600 * 1 # 3600 = 1 h (wintertime)
#GMT_OFFSET = 3600 * 2 # 3600 = 1 h (summertime)
GMT_OFFSET = 3600 * 9 # 3600 = 1 h (KST)

# NTP-Host
NTP_HOST = 'pool.ntp.org'

# Funktion: get time from NTP Server
def getTimeNTP():
    NTP_DELTA = 2208988800
    NTP_QUERY = bytearray(48)
    NTP_QUERY[0] = 0x1B
    addr = socket.getaddrinfo(NTP_HOST, 123)[0][-1]
    s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    try:
        s.settimeout(1)
        res = s.sendto(NTP_QUERY, addr)
        msg = s.recv(48)
    finally:
        s.close()
    ntp_time = struct.unpack("!I", msg[40:44])[0]
    return time.gmtime(ntp_time - NTP_DELTA + GMT_OFFSET)
```

RTC 시간을 설정

```
# Funktion: copy time to PI pico's RTC
def setTimeRTC():
    tm = getTimeNTP()
    rtc.datetime((tm[0], tm[1], tm[2], tm[6] + 1, tm[3], tm[4], tm[5], 0))
```

와이파이를 접속을 시도

```
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect(ssid, password)

max_wait = 10
print('Waiting for connection')
while max_wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
        break
    max_wait -= 1
    sleep(1)
status = wlan.status()
if status != 3:
    raise RuntimeError('Connections failed')
else:
    status = wlan.ifconfig()
    print('connection to', ssid, 'succesfull established!', sep=' ')
    print('IP-adress: ' + status[0])
ipAddress = status[0]
```


DS3231과 I2C로 통신하기

```
# Connect to DS3231
print ('Syncing with DS3231')
sdaPIN = Pin(8) # SDA pin
sclPIN = Pin(9) # SCL pin
i2c = I2C(0, sda=sdaPIN, scl=sclPIN) # Init I2C using pins sda and scl

ds3231 = DS3231(i2c) # Create DS3231 object
```

DS3231와 RTC 시간을 비교하기

```
print('Initial values')
print('DS3231 time:', ds3231.get_time())
print('RTC time:    ', time.localtime())

print('Setting DS3231 from RTC')
# DS3231와 RTC 시간의 차이를 확인하고 싶으면 아래 주석을 하고 실행
#ds3231.save_time() # Set DS3231 from RTC
print('DS3231 time:', ds3231.get_time())
print('RTC time:    ', time.localtime())

# D3231와 RTC 시간의 차이를 확인하고 싶으면 아래 주석을 해제하고 실행
#print('Running RTC test for 2 mins')
#print('RTC leads DS3231 by', ds3231.rtc_test(120, True), 'ppm')
```