고도측정

An **Unmanned aerial vehicle** (UAV) is a Unmanned Aerial Vehicle. UAVs include both autonomous (means they can do it alone) drones and remotely piloted vehicles (RPVs). A UAV is capable of controlled, sustained level flight and is powered by a jet, reciprocating, or electric engine.

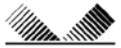
CONTENTS TITLE

Barometric 고도 측정

Barometric Altitude Sensor

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- Drone 의 고도 측정 방법
  - **압력센서(Barometer)**
    - 대기압과 고도의 상관관계를 이용 (10Km 이내)
  - **GPS**
    - 경도, 위도, 고도 정보를 얻을 수 있음.
  - **초음파 센서 (Ultrasonic)**
    - 음파의 반향을 이용하여 time of flight 를 측정.
  - **라이다(Lidar)**
    - 레이저의 반향을 이용하여 time of flight 를 측정



- 압력센서를 이용한 고도측정

Relations Pressure and Altitude

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- 고도 86 km 이내에서는 다음 두식을 사용
  - Equation 1

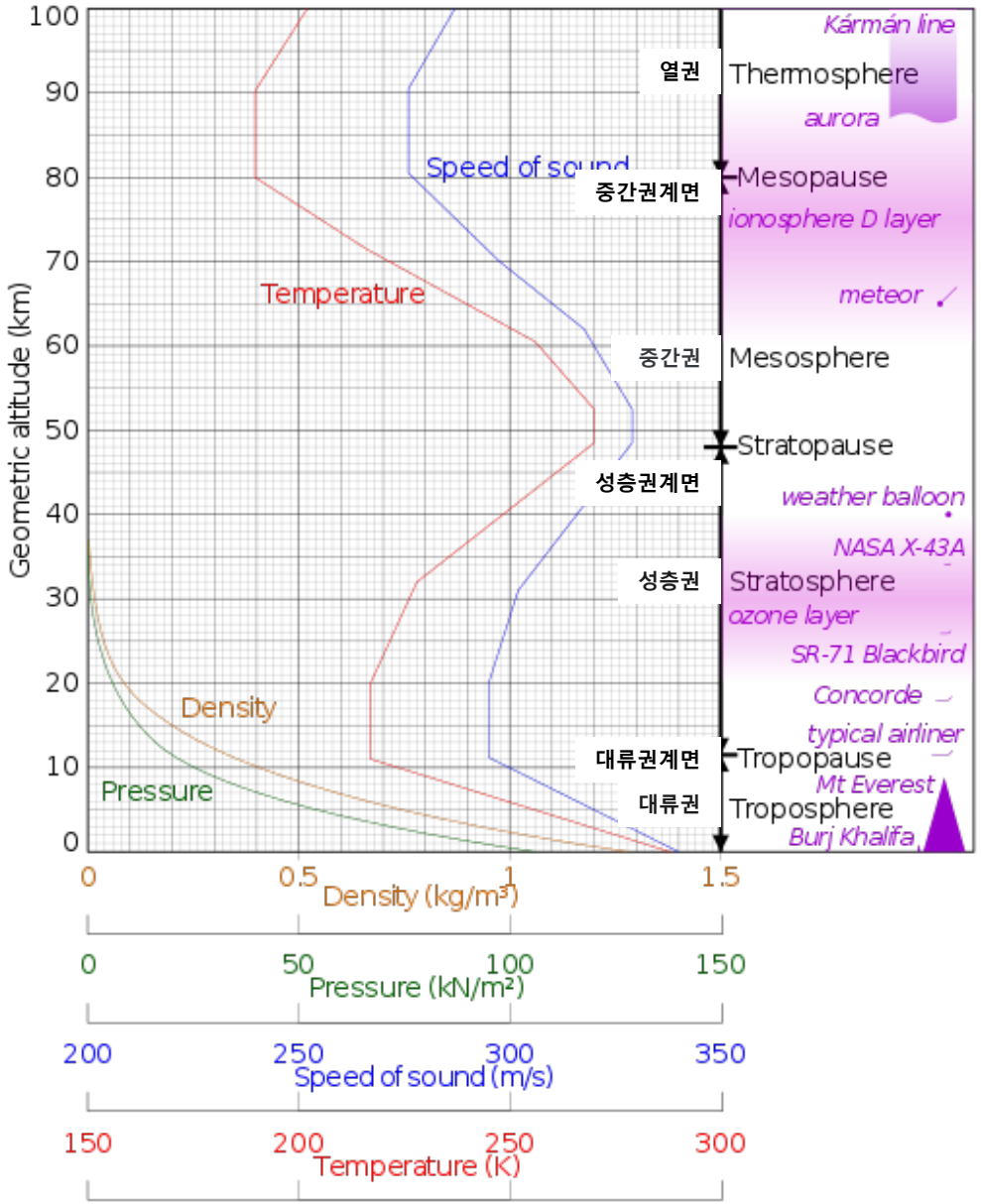
$$P = P_b \left[\frac{T_b}{T_b + L_b(h - h_b)} \right]^{\frac{g_0 M}{R^* L_b}}$$

  - Equation 2

$$P = P_b exp \left[\frac{-g_0 M (h - h_b)}{R^* T_b} \right]$$

  - where,
    - $P_b = Static\ pressure\ [Pa]$
    - $T_b = Standard\ temperature [K]$
    - $L_b = Standard\ temperature\ lapse\ rate [K/m]$
    - $h = height\ from\ sea\ level\ [m]$
    - $h_b = height\ of\ a\ location\ from\ sea\ level\ [m]$
    - $R^* = universal\ gas\ const. : 8.3144598\ J/mol/K$
    - $g_0 = Gravitational\ acceleration : 9.80665\ [\frac{m}{s^2}]$
    - $M = Molecular\ mass\ of\ Air: 0.0289644\ kg/mol$

https://en.wikipedia.org/wiki/International_Standard_Atmosphere



Relations Pressure and Altitude

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- 표준 대기에서의 압력 고도 관계식

$$h = 44{,}330 * \left\{1 - \left(\frac{p}{p_0}\right)^{\frac{1}{5.255}}\right\}$$

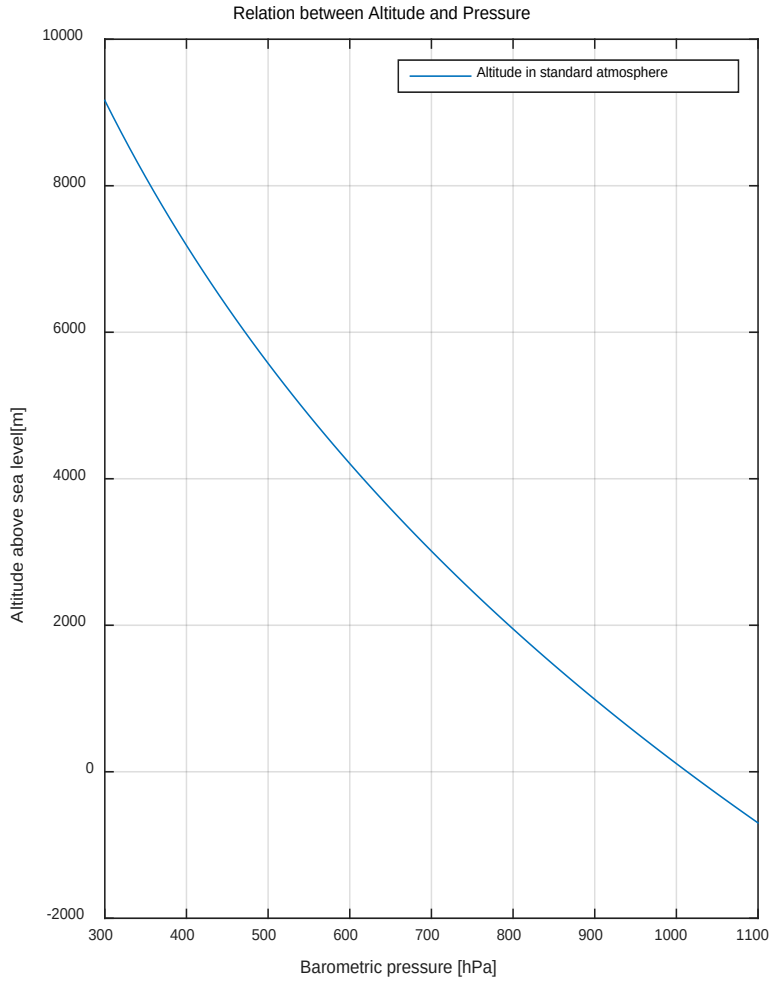
where,

p_0 : Pressure at sea level (1013.25 hPa = 101,325 Pa)

Note:

$1\ hPa\ =\ 100\ Pa$

$1\ Pa\ =\ 1\ N/m^2$



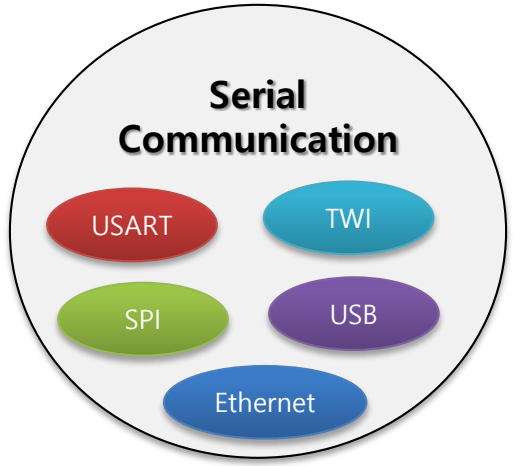
Arduino의 SPI 통신

직렬통신이란?

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

직렬통신의 개념

- 디지털 데이터를 하나의 선으로 일렬로 전송하는 방식.
- 물리적으로는 Ethernet, USB, RS-232, CAN 등도 해당.
- Atmega2560에는 UART, TWI, SPI를 지원하여 주변 장치와 정보를 교환함



APM 제공 통신의 종류

- 시리얼 통신 3 종류 지원
  - **UART** (0~3), **SPI**, **TWI**(I2C)
- **SPI**
  - Full-duplex, Three-wire :
    - 전이중 통신, 동기식 데이터 전송.
  - Master or Slave Operation:
  - 최대 20MHz 전송속도
- **TWI(I2C)**
  - 7-bit Address: 총 128 Slave와 1:n 통신
  - Half-duplex, two-wire
    - 반이중, 동기식 데이터 전송
  - 최대 400kHz 전송속도

양방향 통신

- 양방향 통신은 다음 2가지 경우가 있음.
  - 전이중(Full duplex): 송신/수신을 동시에 가능
    - USART, SPI
  - 반이중(Half duplex): 송신/수신을 번갈아 가면서 수행
    - I2C

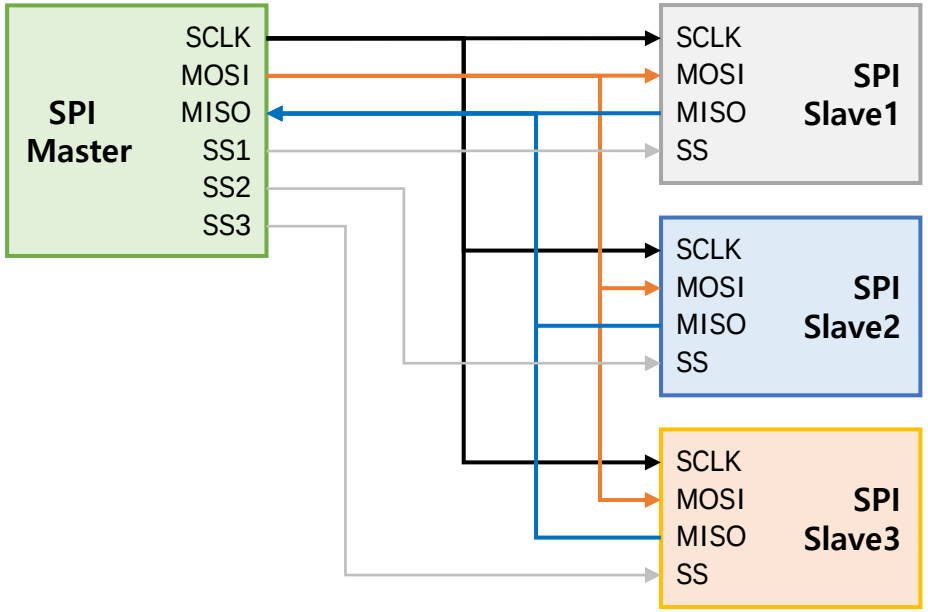
BUS	동기방식	1:n	양방향	1:1인 경우 최소 선수
UART	baud rate	1:1	Full	3 TX, RX
I2C	clock line (SCL)	ok (address방식)	Half	3 SDA, SCL
SPI	clock line (SCLK)	ok (chip select방식)	Full	4 MISO, MOSI, SCLK

SPI 개요

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- SPI : Serial Peripheral Interface
  - MCU와 다른 장치 (PC, MCU)들과 정보교환에 유용
  - 최소 4선이 필요

신호	의미	방향	비고
SCLK	Serial Clock	From Master	
MOSI	Master Output, Slave Input		
SS	Slave Select		Low active
MISO	Master Input, Slave Output	From Slave	



SPI 장단점

장점

- Full duplex communication (전이중 통신)
- 소전력, I²C(TWI)나 SMBus에 비해 효율적 전송
- 고속, 신호의 무결성 우수

단점

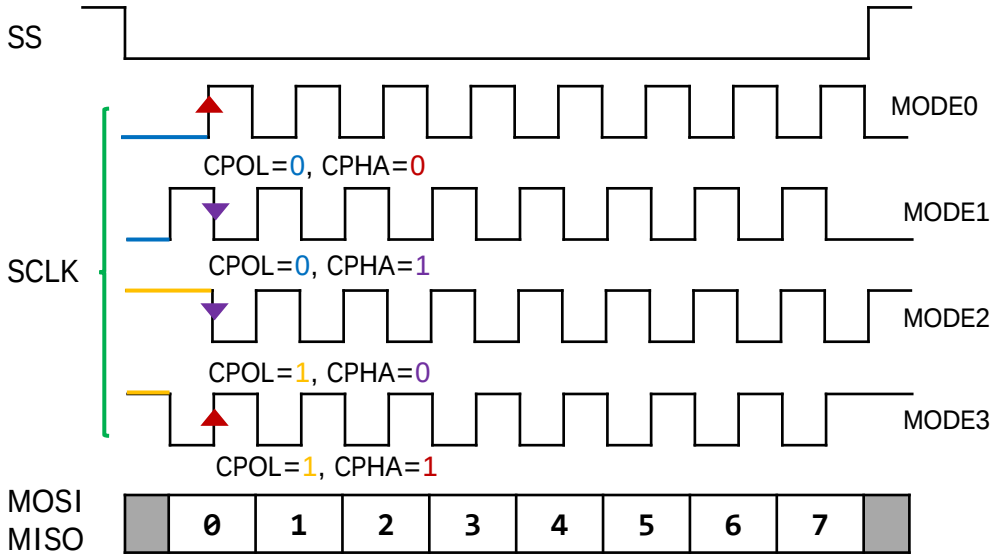
- 소요 핀수와 선수가 많음. (최소 4선)

SPI 동작 모드

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

SPI 동작 모드

- 모드: Clock 극성과 데이터 읽기 위상을 결정
  - CPOL : Clock이 어떤 상태에서 시작하는 지를 결정
  - CPHA : 데이터 읽는 엣지 시점 정의
    - = 0 : CPOL에서 지정한 상태의 반대로 바뀔 때
    - = 1 : CPOL에서 지정한 상태가 될 때



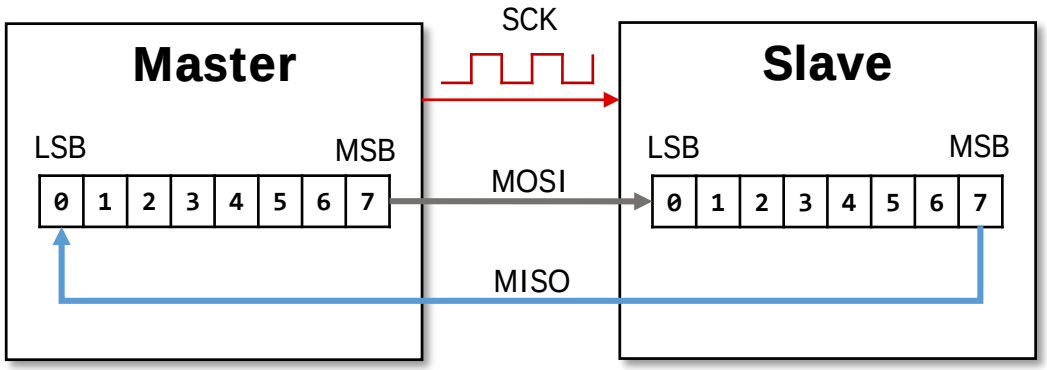
체크: 사용하고자하는 칩의 SPI가 어떤 모드를 사용하는지 확인 필요.

Mode	Clock Polarity (CPOL)	Clock Phase (CPHA)	Data Capture
SPI_MODE0	0	0	Rising
SPI_MODE1	0	1	Falling
SPI_MODE2	1	0	Falling
SPI_MODE3	1	1	Rising

SPI 개요

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- 데이터 전송 bit 순서
  - MSBFIRST: most-significant bit 먼저 전송
  - LSBFIRST: least-significant bit 먼저 전송
  - 대개 MSBFIRST 사용



MSBFIRST

Arduino에서의 SPI

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

SPI 객체 사용 (SPI.h)

- 라이브러리 사용

```
#include <SPI.h>
```

  - 라이브러리 include

- SPI 시작

```
SPI.begin()
```

  - Initializes the SPI bus by setting SCK, MOSI, and SS to outputs, pulling SCK and MOSI low, and SS high.

- SPI 설정

```
SPI.beginTransaction(SPISettings(clock, bitOrder, dataMode))
```

  - *clock*: clock speed 1~20MHz
  - *bitOrder*: MSBFIRST or LSBFIRST
  - *dataMode*: `SPI_MODE0, SPI_MODE1, SPI_MODE2, SPI_MODE3`

- SPI 전송 및 수신

```
ret = SPI.transfer(val)
```

  - *val*: the byte to send out over the bus
  - *ret*: received data

SPI 선언

SPI 선언

- 속도, 비트순서, 모드 설정
- 8MHz, MSB 우선, mode0 지정

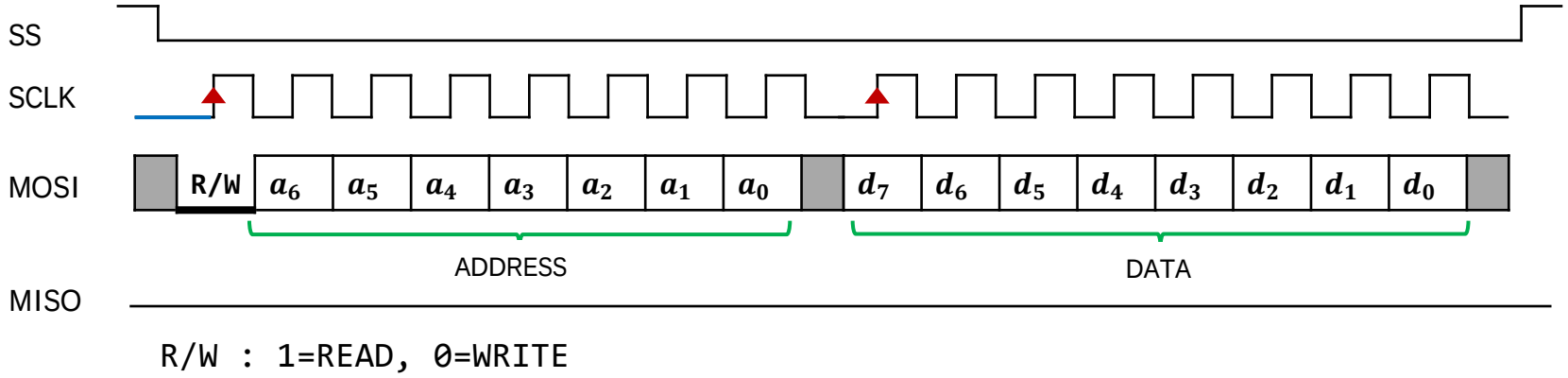
```
#include <SPI.h>
#define ChipSelPin 53;
...
  SPI.begin();
  SPI.beginTransaction(SPISettings(8000000, MSBFIRST, SPI_MODE0));
  pinMode(ChipSelPin, OUTPUT);
```

SPI 데이터 쓰기

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

SPI 데이터 쓰기 타이밍 차트

- 절차
  - Chip select: SS → low
  - Address 전송: 상위비트 Low + 7bit 어드레스
  - 데이터 전송 : 1바이트
  - Chip deselect: SS→ high



R/W : 1=READ, 0=WRITE

SPI 데이터 쓰기

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- 데이터 쓰기 코드 예
  - Chip select
    → 선택(LOW)
  - .transfer : address (8bits)
  - .transfer : data
  - Chip select release
    → 선택 해제(HIGH)

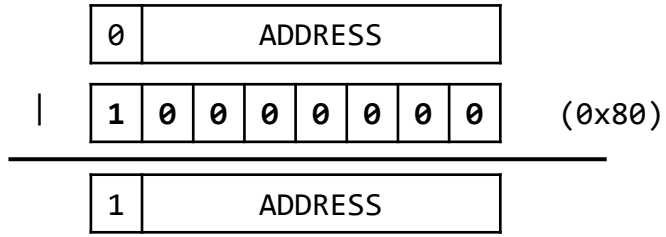
```
void SPIwrite(byte reg, byte data) {
  uint8_t dump;
  digitalWrite(ChipSelPin,LOW);
  dump=SPI.transfer(reg);
  dump=SPI.transfer(data);
  digitalWrite(ChipSelPin,HIGH);
}
```

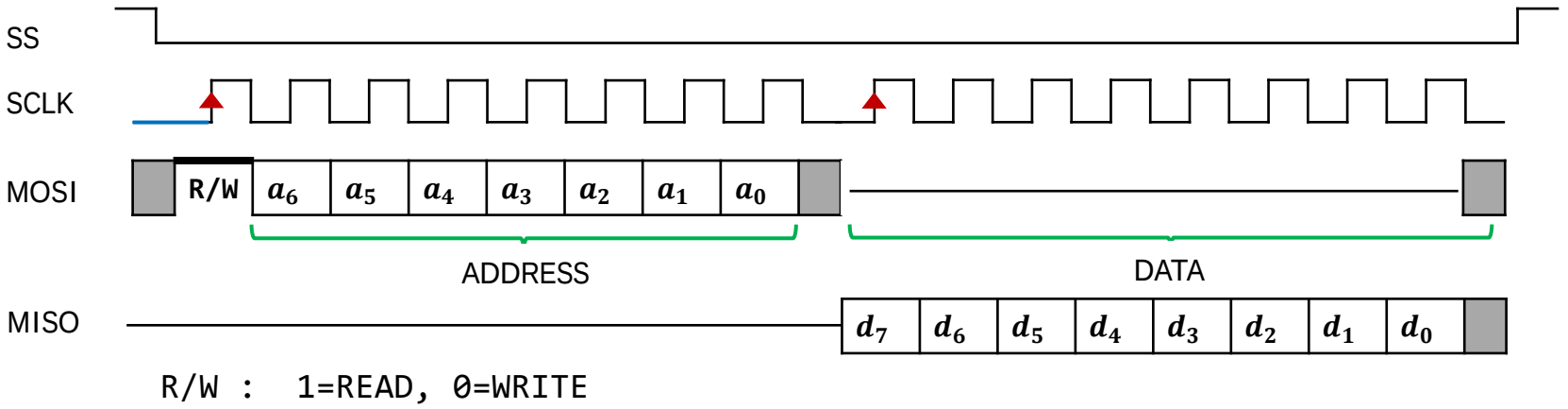
SPI 데이터 읽기

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

SPI 데이터 읽기 타이밍 차트

- 절차
  - Chip select: SS → low
  - Address 전송
    - 주소는 최상위 비트 1 (addr = ADDRESS| 0x80;)
  - 0 전송 → 데이터 반환
  - Chip deselect: ss → high





SPI 데이터 읽기

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

데이터 읽기 코드

- 주소 준비 :

  addr = reg| 0x80;

- Chip select

  → 선택(LOW)

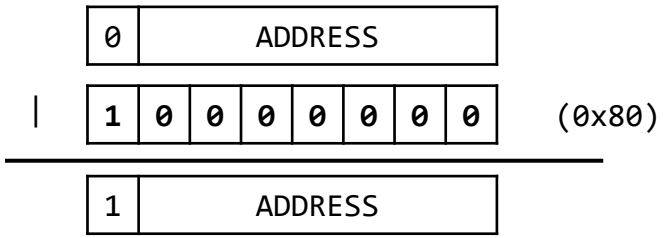
- .transfer : address (8bits)
- .transfer : 0 (아무거나)

  → data return !

- Chip select

  → 선택해제(HIGH)

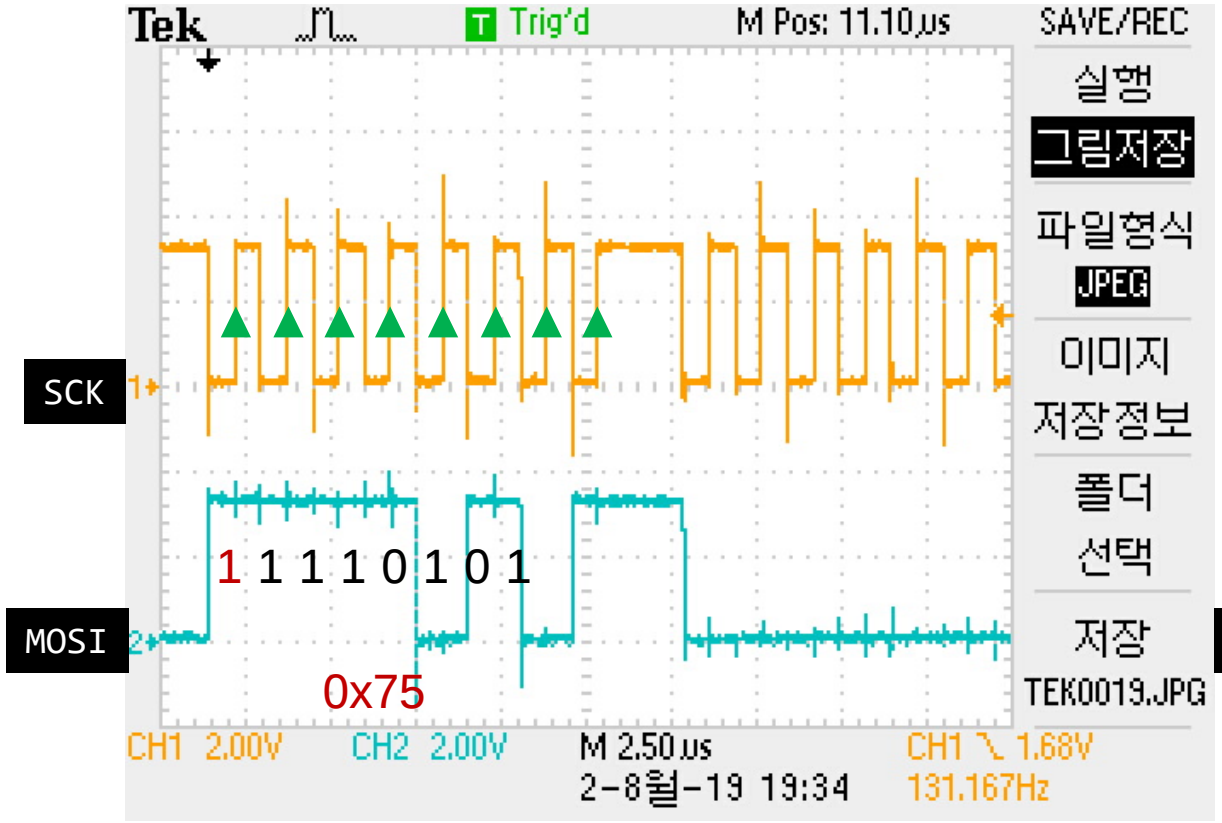
```
uint8_t SPIread(byte reg,int ChipSelPin) {
  uint8_t dump;
  uint8_t return_value;
  uint8_t addr=reg|0x80;
  digitalWrite(ChipSelPin,LOW);
  dump=SPI.transfer(addr);
  return_value=SPI.transfer(0x00);
  digitalWrite(ChipSelPin,HIGH);
  return(return_value);
}
```



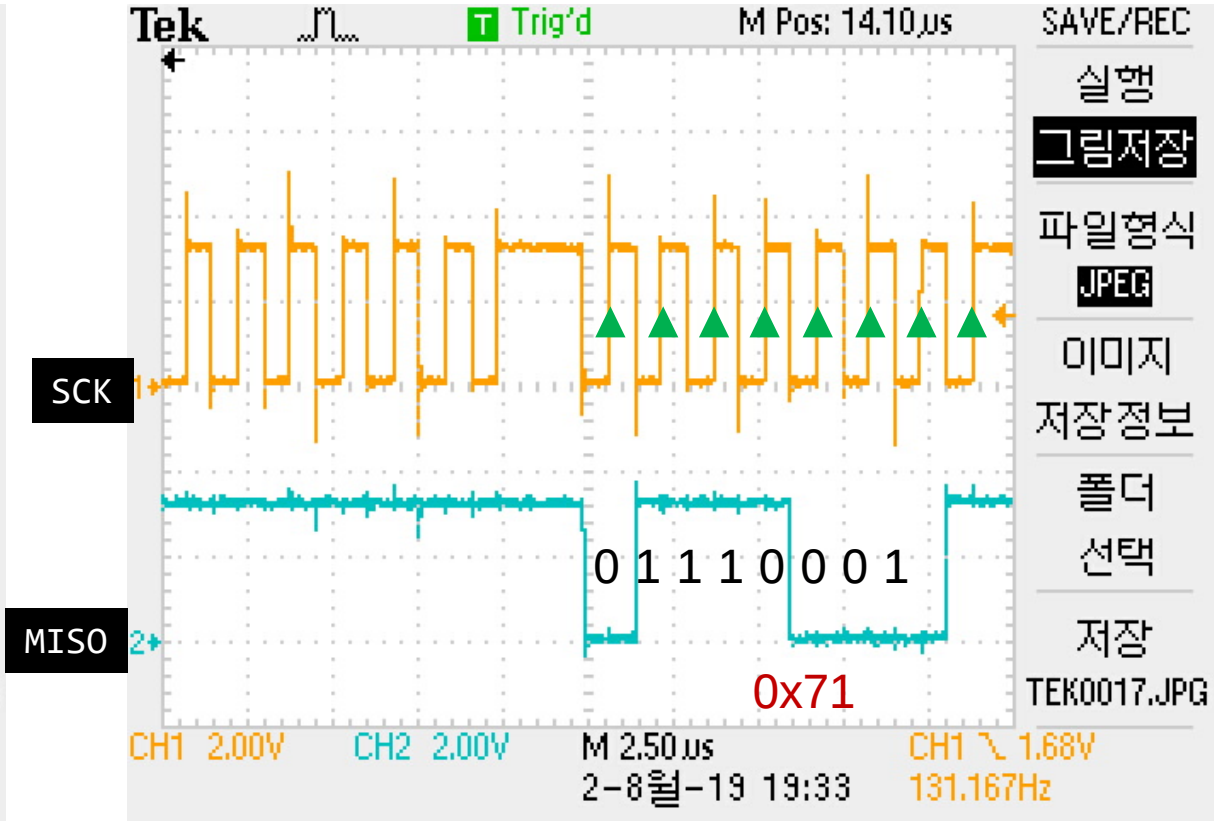
SPI 신호 확인

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- 설정(mode3: 상승엣지, 클럭 1Mhz)



Master Write : 0x75



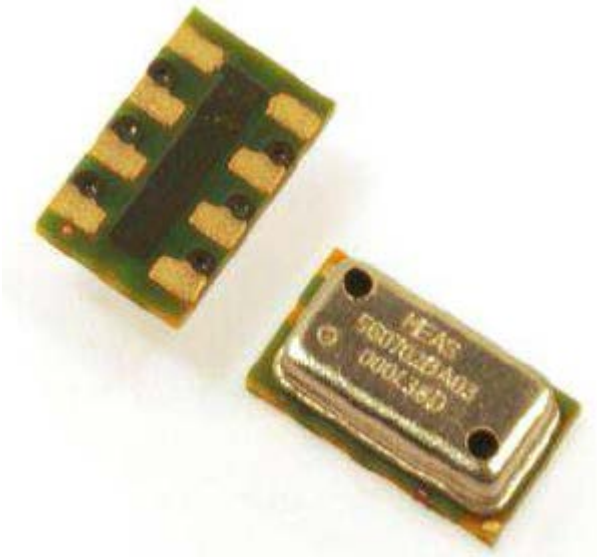
Slave Reply: 0x71

Barometer MS5611

Barometer MS5611소개

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- **Barometric Pressure Sensor by MEAS**:
  - **SPECIFICATIONS**
    - Resolution: ∓10 cm
    - Conversion speed: 1 ms
    - Power: 1 μA (standby < 0.15 μA)
    - Package: QFN 5.0 x 3.0 x 1.0 mm^3
    - Supply voltage: 1.8 to 3.6 V
    - AD conversion: 24 bit ΔΣ ADC
    - Operating range: 10 to 1200 mbar, -40 to +85 °C
    - Interface: I2C or SPI up to 20 MHz
    - Principle: piezoresistive sensor

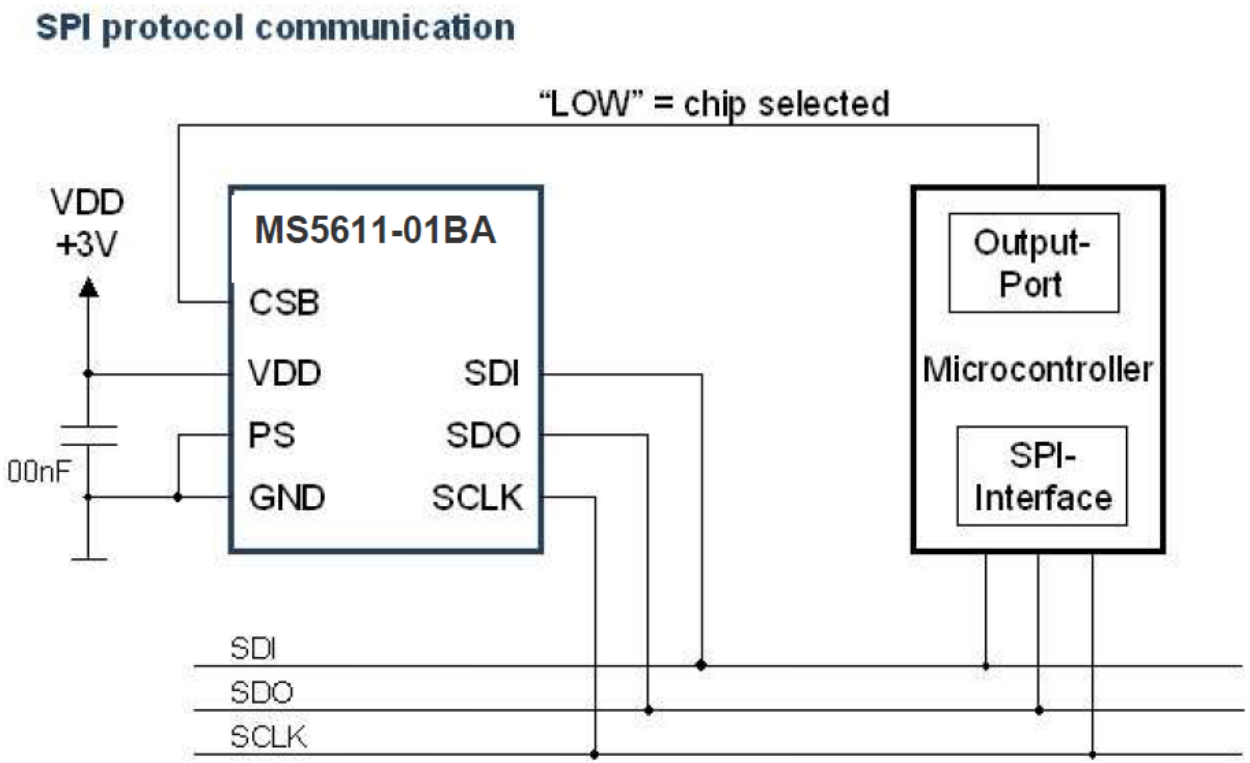


출처: https://www.te.com/usa-en/product-CAT-BLPS0036.html?q=&n=135117&type=products&samples=N&instock=N

MS5611의 회로

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- **SPI connections**:
  - **PS**:
    - 통신방식 I2C/SPI 선택
  - **CSB**:
    - chip select
    - (pin 40 in APM)
  - **SDI** : MOSI
  - **SDO**: MISO
  - **SCLK**: SPI clock



출처: https://www.te.com/usa-en/product-CAT-BLPS0036.html?q=&n=135117&type=products&samples=N&instock=N

MS5611의 명령

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- **SPI Commands**:
  - **Reset**: reset the chip
  - **Read PROM**: calibration 데이터
    - C1~C6
  - **D1** conversion:
    - Digital **pressure** value
    - OSR =(256, 512,2048,4096)
  - **D2** conversion:
    - Digital **temperature** value
  - **Read ADC** result:
    - 24bit pressure/temperature

명령과 주소를 동일하게 본다.

	Command byte								hex value
Bit number	0	1	2	3	4	5	6	7	
Bit name	PRM	COV	-	Typ	Ad2/Os2	Ad1/Os1	Ad0/Os0	Stop	
Command									
Reset	0	0	0	1	1	1	1	0	0x1E
Convert D1 (OSR=256)	0	1	0	0	0	0	0	0	0x40
Convert D1 (OSR=512)	0	1	0	0	0	0	1	0	0x42
Convert D1 (OSR=1024)	0	1	0	0	0	1	0	0	0x44
Convert D1 (OSR=2048)	0	1	0	0	0	1	1	0	0x46
Convert D1 (OSR=4096)	0	1	0	0	1	0	0	0	0x48
Convert D2 (OSR=256)	0	1	0	1	0	0	0	0	0x50
Convert D2 (OSR=512)	0	1	0	1	0	0	1	0	0x52
Convert D2 (OSR=1024)	0	1	0	1	0	1	0	0	0x54
Convert D2 (OSR=2048)	0	1	0	1	0	1	1	0	0x56
Convert D2 (OSR=4096)	0	1	0	1	1	0	0	0	0x58
ADC Read	0	0	0	0	0	0	0	0	0x00
PROM Read	1	0	1	0	Ad2	Ad1	Ad0	0	0xA0 to 0xAE

OSR: Oversampling Ratio

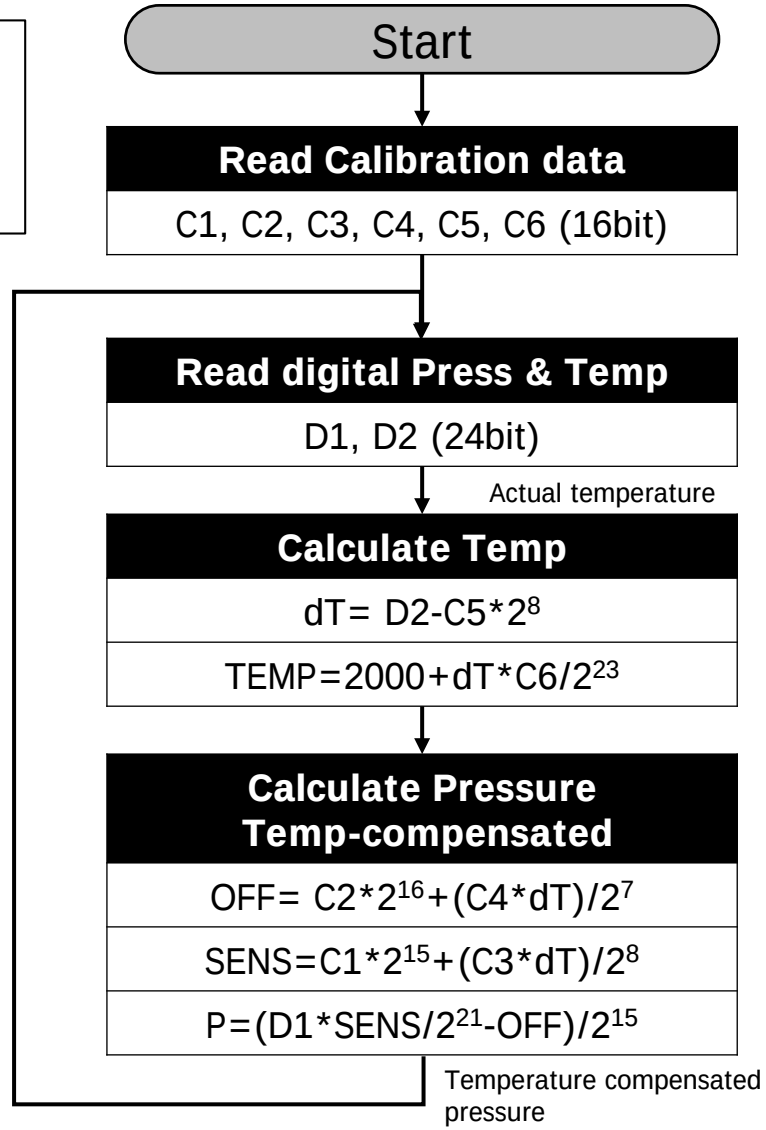
출처: https://www.te.com/usa-en/product-CAT-BLPS0036.html?q=&n=135117&type=products&samples=N&instock=N

압력 측정 과정

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- **Procedure of operations**:
  - **Configure**
    - Reset: reset the chip
    - Read: **C1, C2, C3, C4, C5, C6**
  - **Loop**
    - Read: digital **D1, D2**
    - Calculate : Temperature
    - Calculate : Temperature-compensated Pressure

$P_{MIN} = 10mbar,$
$P_{MAX} = 1200mbar,$
$T_{MIN} = -40\,℃,$
$T_{MAX} = 85℃,$
$T_{REF} = 20\,℃$



출처: https://www.te.com/usa-en/product-CAT-BLPS0036.html?q=&n=135117&type=products&samples=N&instock=N

MS5611의 SPI 프로그래밍

MS5611 설정

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

Configuration

- 속도, 비트순서, 모드 설정
- 5MHz, MSB 우선, mode0 지정

```
#include <SPI.h>
#define MS5611_CS 53;
...
  SPI.begin();
  SPI.beginTransaction(SPISettings(5000000, MSBFIRST, SPI_MODE0));
  pinMode(ChipSelPin1, OUTPUT);
```

명령 전송 프로그램

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

Writing a command

- Chip select → LOW
- .transfer: reg

```
void _spi_write(uint8_t reg){
  uint8_t dump;
  digitalWrite(MS5611_CS, LOW);
  dump = SPI.transfer(reg);
  digitalWrite(MS5611_CS, HIGH);
}
```

16비트 데이터 읽기

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- **Read 16bit data read**
  - chip select
  - send address
  - send 0 and get H- byte
  - send 0 and get L- byte
  - ➔ Big endian

```
uint16_t _spi_read_16bits(uint8_t reg){
  uint8_t dump, byteH, byteL;
  uint16_t return_value;
  uint8_t addr = reg; //reg already has | 0x80;
  digitalWrite(MS5611_CS, LOW);
  dump = SPI.transfer(addr);    //write Address
  byteH = SPI.transfer(0);      //read High
  byteL = SPI.transfer(0);      //read Low
  digitalWrite(MS5611_CS, HIGH);
  return_value = ((uint16_t)byteH<<8) | (byteL);
  return return_value;
}
```

24비트 데이터 읽기

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- Read 24bit data ADC read
  - set address
  - send 0 and get Hi
  - send 0 and get Mid
  - send 0 and get Lo
  - 3 바이트 결합

```
uint32_t _spi_read_adc() {
  uint8_t dump,byteH,byteM,byteL;
  uint32_t return_value;
  uint8_t addr = 0x00;
  digitalWrite(MS5611_CS, LOW);
  dump = SPI.transfer(addr);
  byteH = SPI.transfer(0);
  byteM = SPI.transfer(0);
  byteL = SPI.transfer(0);
  digitalWrite(MS5611_CS, HIGH);
  return_value=
     (((uint32_t)byteH)<<16)|(((uint32_t)byteM)<<8)|(byteL);
  return return_value;
}
```

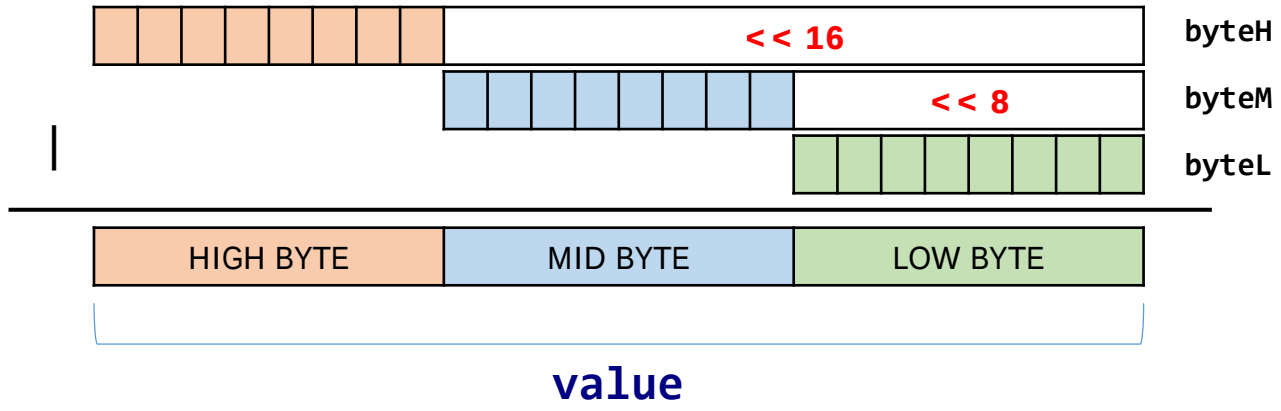
24비트 데이터 처리

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- Read 24bit data 생성
  - `value = (((uint32_t)byteH)<<16)|(((uint32_t)byteM)<<8)|(byteL);`

<< 비트단위 쉬프트

| 비트단위 OR

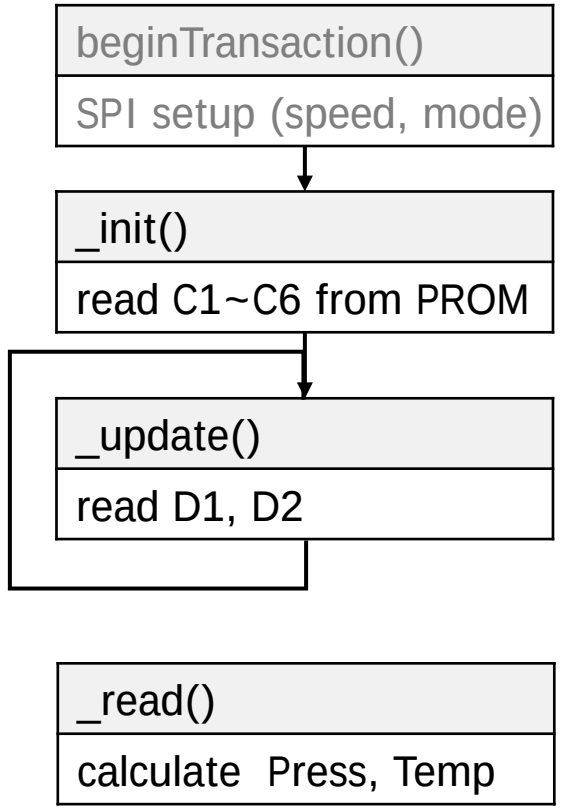


측정 프로그래밍

프로그램의 개요

주요 함수

- beginTransaction()
  - SPISettings(5000000, MSBFIRST, SPI_MODE0)
- _init()
  - PROM 으로부터 교정 데이터 읽어옴. C1~C6
  - initialize variables
- _update()
  - OSR=4096일 때 D1, D2를 읽는데 8.2 ms 소요됨.
  - 온도보다 압력이 더 중요하기 때문에
    - D1(압력) 4회 당 D2(온도) 1회 배분으로 읽어 들임.



초기화

- 초기화 함수 _init()
  - Chip deselect (해제)
  - Reset chip
  - Read C1~C6
  - Command for D2(온도)
  - 변수들의 초기화

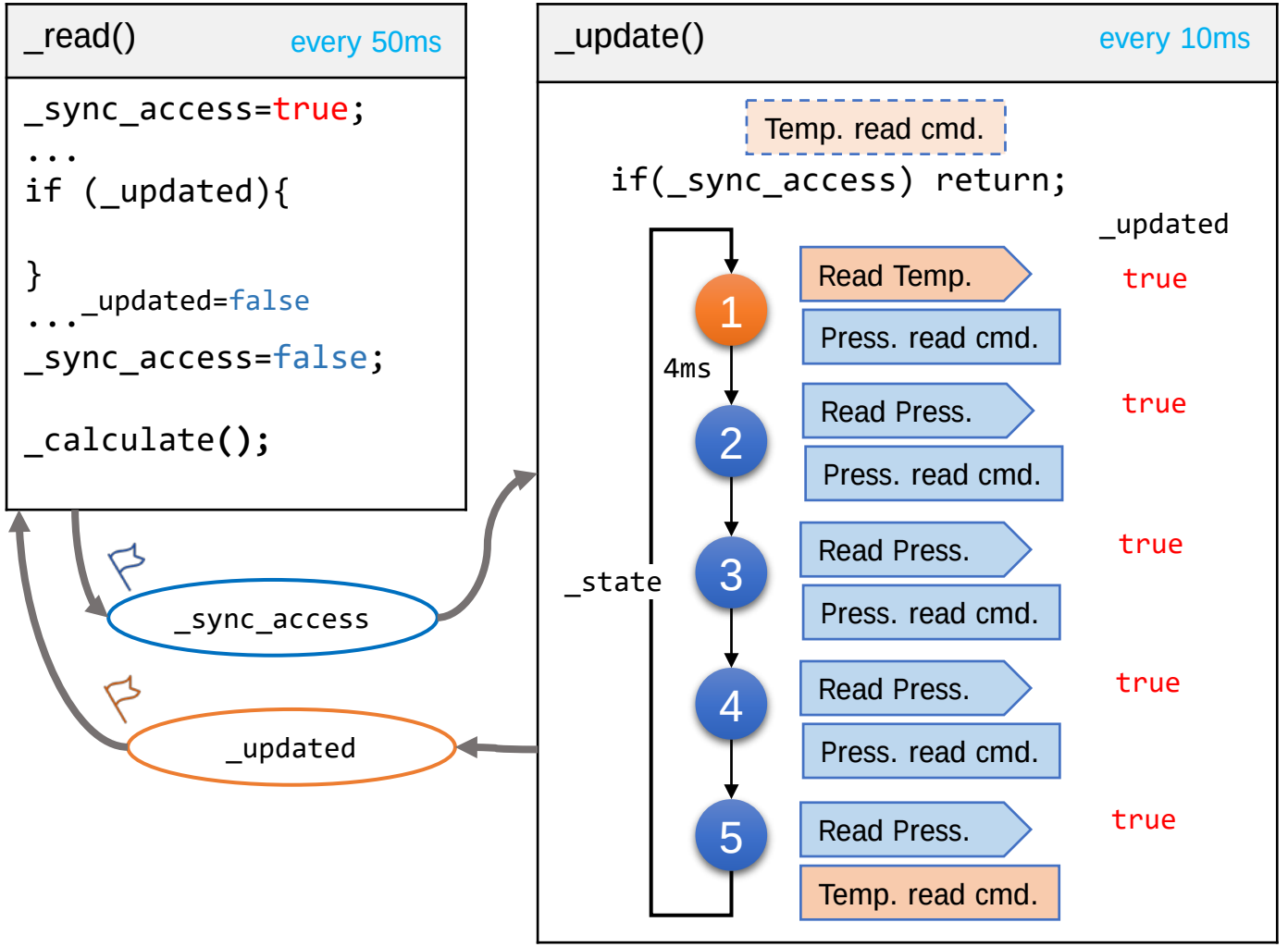
```
bool _init(){
  pinMode(MS5611_CS, OUTPUT);  // Chip select Pin
  digitalWrite(MS5611_CS, HIGH);  delay(1);
  _spi_write(CMD_MS5611_RESET);   delay(4);
  // Read 6 calibration data
  C1 = _spi_read_16bits(CMD_MS5611_PROM_C1);
  C2 = _spi_read_16bits(CMD_MS5611_PROM_C2);
  C3 = _spi_read_16bits(CMD_MS5611_PROM_C3);
  C4 = _spi_read_16bits(CMD_MS5611_PROM_C4);
  C5 = _spi_read_16bits(CMD_MS5611_PROM_C5);
  C6 = _spi_read_16bits(CMD_MS5611_PROM_C6);
  //Send a command to read Temp first
  _spi_write(CMD_CONVERT_D2_OSR4096);
  _timer = micros();
  _state = 1;  Temp=0;   Press=0;
  return true;
}
```

측정 update와 read

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- _update()
  - D1, D2를 읽는데 각각 8.2ms 소요
  - 총 5회로 나누어 중요도에 따라 Temp : Press = 1 : 4로 배분

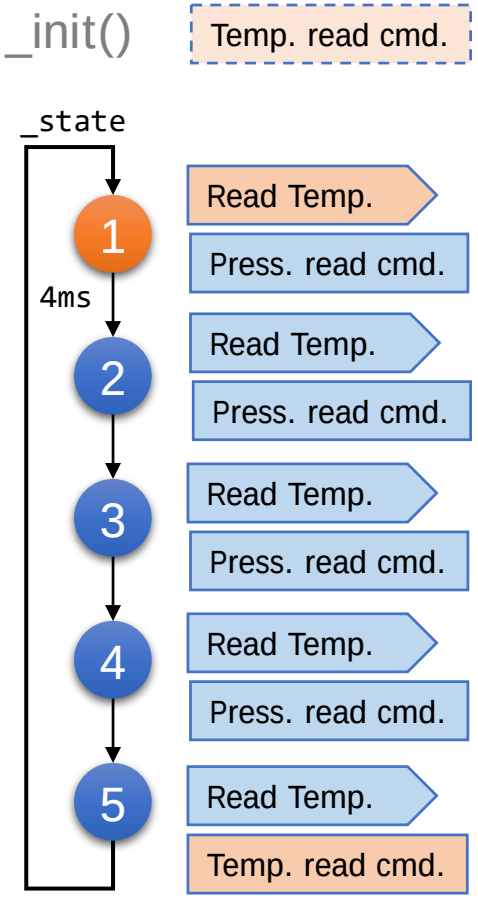
- _update와 _read의 상호작용
  - _sync_access 플래그
    - 데이터를 읽는 동안 업데이트가 되지 않도록 함.
    - 이런 것을 **locking**이라 함
  - _updated 플래그
    - 업데이트된 경우만 읽기
    - 불필요한 읽기 배제



Update 함수

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

_update()




```
void _update(uint32_t tnow) {
   if (_sync_access) return;
   if (tnow - _timer < 9500) {return; }    //>>8.2 ms
   _timer = tnow;
    if (_state == 1) {
      _s_D2 = _spi_read_adc();              // Read temp
      _state++;
      _spi_write(CMD_CONVERT_D1_OSR4096);  // Pressure Read cmd.
    } else if (_state == 5) {
      _s_D1 = _spi_read_adc();
      _state = 1;                           // again state = 1
      _spi_write(CMD_CONVERT_D2_OSR4096);  // temperature read cmd.
      _updated = true;                      // Update done
    } else {                                 // state==2,3,4
      _s_D1 = _spi_read_adc();
      _state++;
      _spi_write(CMD_CONVERT_D1_OSR4096);  // Pressure Read cmd.
      _updated = true;                      // Update done
    }
}
```

Read 함수

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- _read()
  - locking구간
    - 이 기간은 update금지
    - _sync_access 플래그

```
uint8_t _read(){
    _sync_access = true;          // update()중지
     bool updated = _updated;
     _updated = 0;
     if (updated > 0) {
         D1 = _s_D1;              // 압력
         D2 = _s_D2;              // 온도
         _raw_press = D1;
         _raw_temp = D2;
     }
     _sync_access = false;
     _calculate();
     return updated ? 1 : 0;
}
```

locking 구간

Calculate 함수

- _calculate()
  - 압력/온도 보정

```
void  _calculate(){ //Calculate Temp & compensated P
// Data expressed in Celsius degrees*10, mbar*100
  int32_t dT;
  int64_t TEMP, OFF, SENS, P;
  dT = D2-((long)C5*256); // Formulas from datasheet
  TEMP = 2000 + ((int64_t)dT * C6)/8388608;
  OFF = (int64_t)C2 * 65536 + ((int64_t)C4 * dT ) / 128;
  SENS = (int64_t)C1 * 32768 + ((int64_t)C3 * dT) / 256;
  if (TEMP < 2000){   // second order temperature compensation
    int64_t T2 = (((int64_t)dT)*dT) >> 31;
    int64_t Aux_64 = (TEMP-2000)*(TEMP-2000);
    int64_t OFF2 = (5*Aux_64)>>1;
    int64_t SENS2 = (5*Aux_64)>>2;
    TEMP = TEMP - T2;
    OFF = OFF - OFF2;
    SENS = SENS - SENS2;  }
  P = (D1*SENS/2097152 - OFF)/32768;
  Temp = TEMP;
  Press = P;
}
```

get_altitude 함수

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- get_altitude()
  - 보정된 압력으로 부터 고도 산출

$$h = 44{,}330 * \left\{1 - \left(\frac{p}{p_0}\right)^{\frac{1}{5.255}}\right\} = 44{,}330 * \left\{1 - \left(\frac{p}{101325.0}\right)^{0.190295}\right\}$$

  - 고도-압력 관계식을 그대로 코드화

```
float get_altitude() {
  float tmp_float;
  float Altitude;
  tmp_float = (Press / 101325.0);
  tmp_float = pow(tmp_float, 0.190295);
  Altitude = 44330.0 * (1.0 - tmp_float);
  return Altitude;
}
```

APM 코드

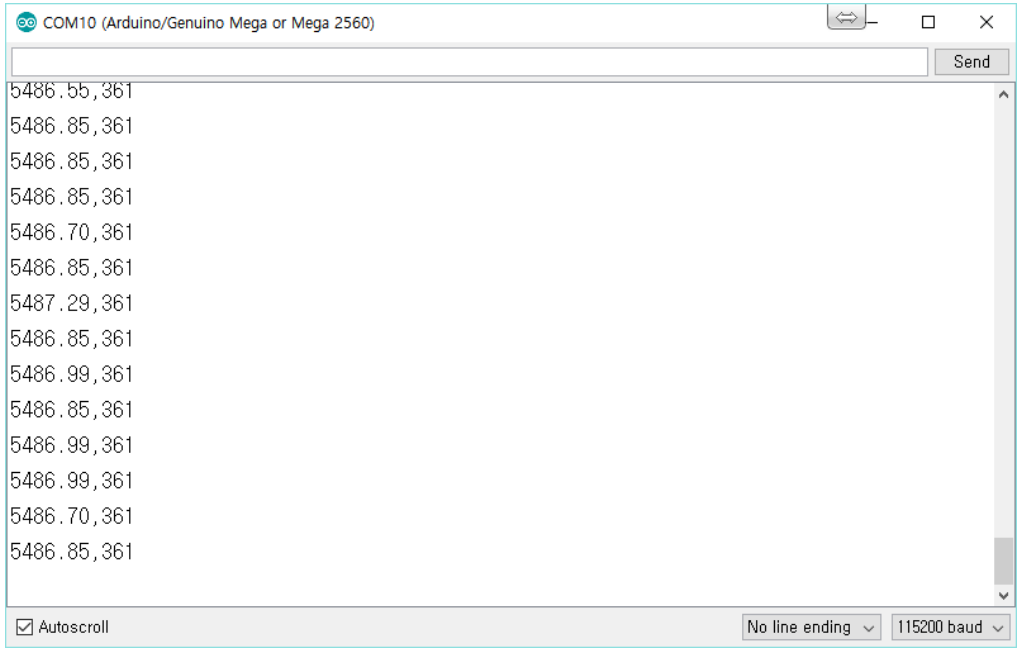
- setup, loop 함수

```
#include "MS5611.h"
#include <SPI.h>
uint32_t curtime=0, prevtime=0,count=0;
void setup() {
  Serial.begin(115200);
  SPI.begin();
  SPI.beginTransaction(SPISettings(5000000, MSBFIRST, SPI_MODE0));
  _init();
  prevtime=micros();
}
void loop() {
  do {curtime=micros();} while (curtime-prevtime<1000);   // 1ms
    prevtime=curtime;
    if (count% 10==0)                 // 10ms
          _update(curtime);
    if (count% 50==0) {               // 50ms
         _read();
         Serial.println(get_altitude());
      }
  count ++;
}
```

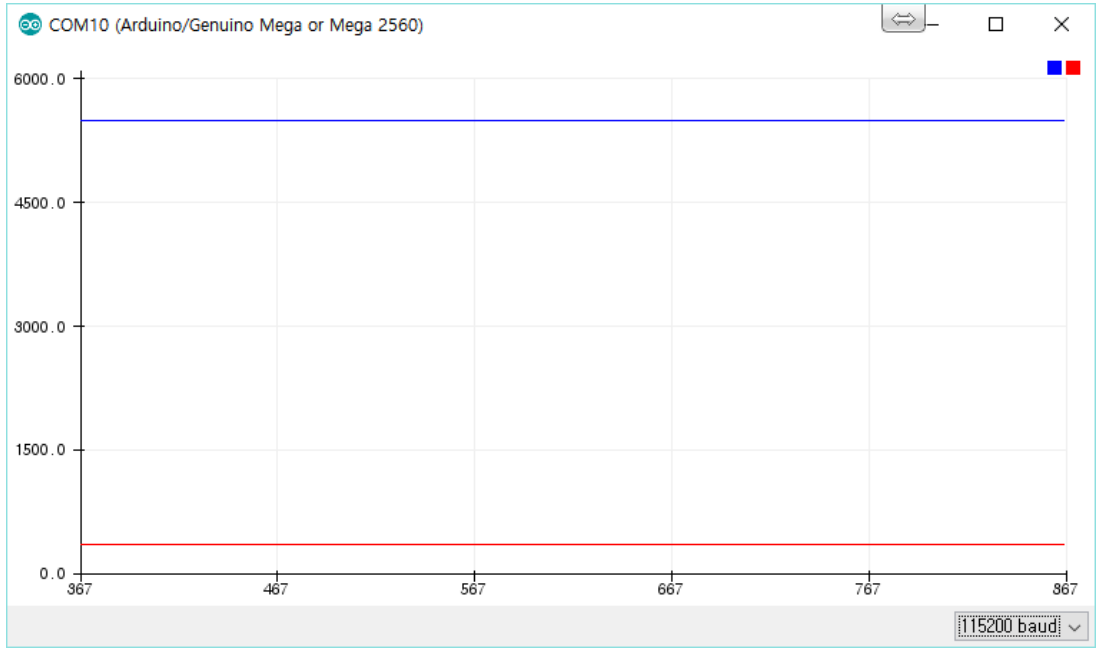
측정 결과

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

Results



고도(x100m), 온도(x10 ℃)

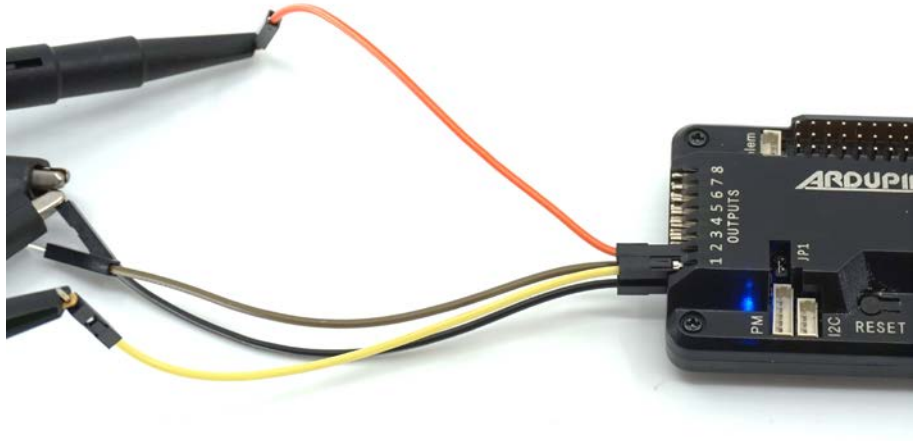


고도, 온도

_update() 실행 시간 측정

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- Ch1:_read 주기, Ch2: _update 실행시간
  - OUT1: 12번핀
  - OUT2: 11번핀

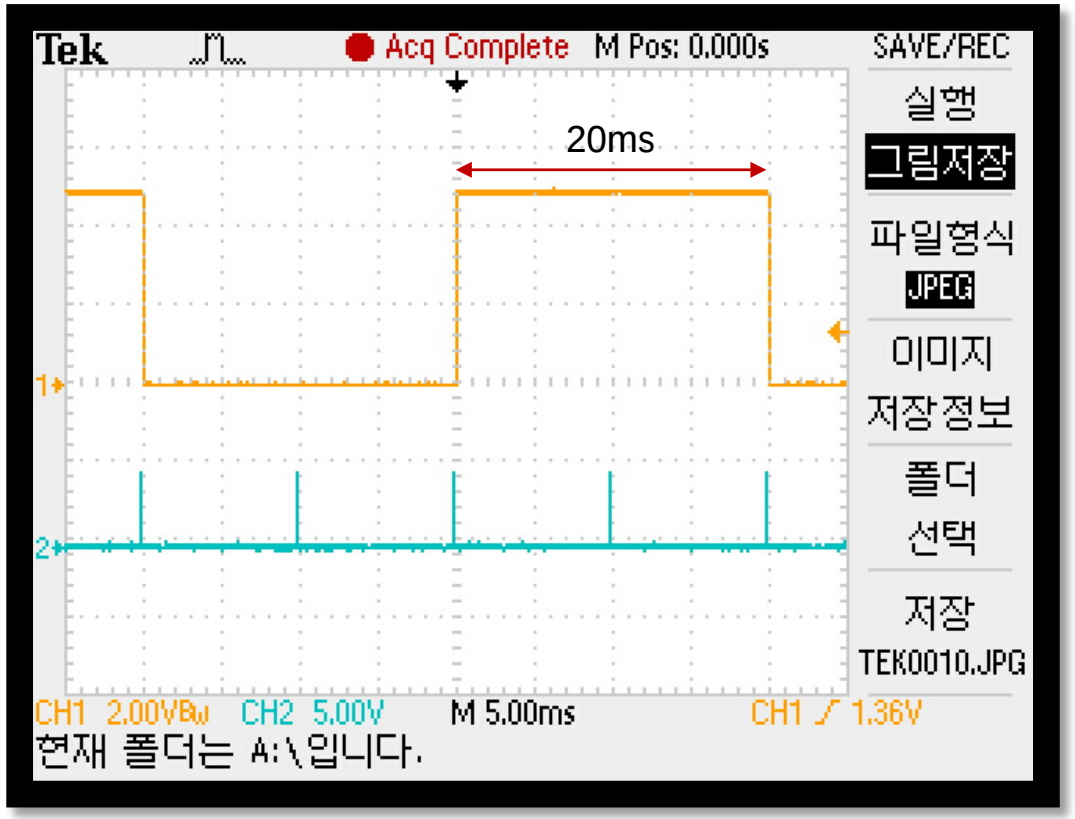


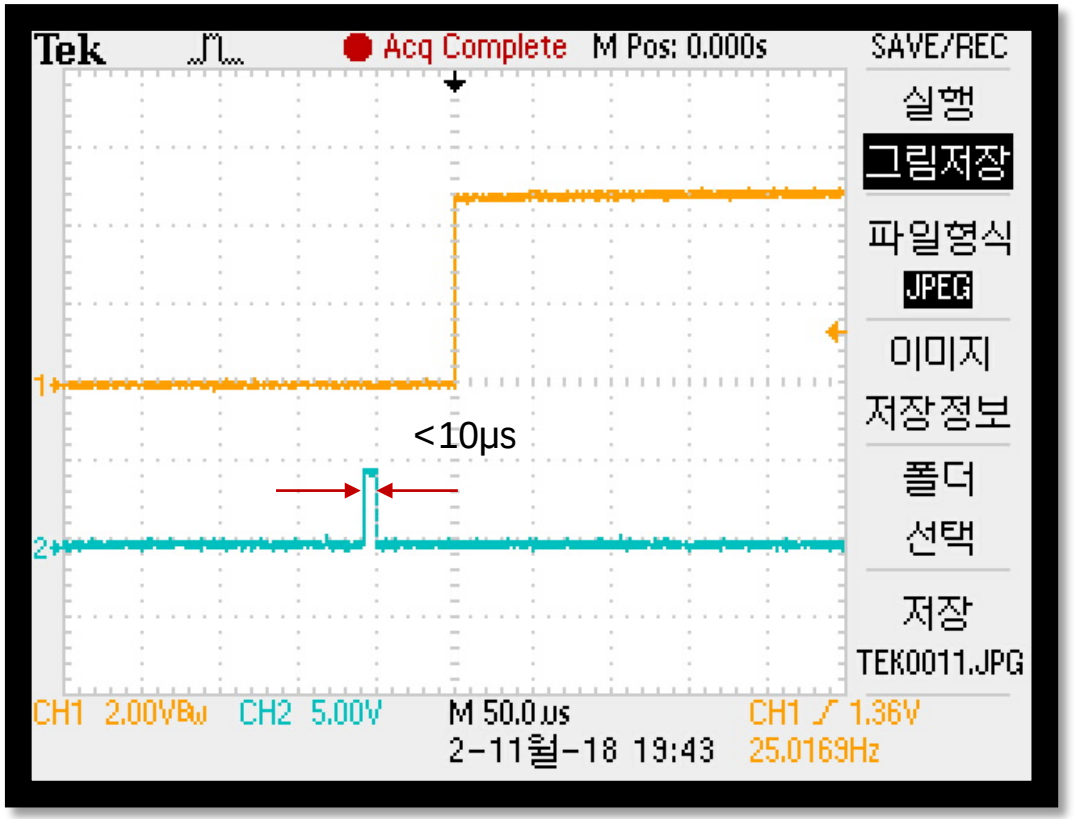

```
void setup() {
  pinMode(12,OUTPUT);    pinMode(11,OUTPUT);
  Serial.begin(115200);
  ...
}
void loop() {
  do {curtime=micros();}  while (curtime-prevtime<1000);
    prevtime=curtime;
    if (count% 10==0)
          digitalWrite(11,HIGH);              //OUT2
        _update(curtime);
        digitalWrite(11,LOW);
     if (count% 20==0) {
          digitalWrite(12,!digitalRead(12));  //OUT1
        _read();
          Serial.println(get_altitude());
     }
  count ++;
}
```

_update() 실행 시간 측정

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- Ch1:_read 주기, Ch2: _update 실행시간





측정에 Class 적용

MS5611 클래스선언

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

- **MS5611 Class:**
  - Property
    - _s_D1, _s_D2 : 임시 압력, 온도
    - _state : 읽기 순서
    - _sync_access : 읽기 중 업데이트 방지
    - C1,C2,C3,C4,C5,C6 : 교정 데이터
  - Method
    - _init(): 초기화 설정
    - _update(tnow)(): 순차적으로 데이터 읽기
    - _read(): 업데이트 결과 반영
    - _calculate(): 온도 보상 계산
    - get_altitude(): 압력에서 고도 구하기.

MS5611 Class

data

```
_s_D1, _s_D2;
_state, _timer;
_sync_access, _updated;
C1,C2,C3,C4,C5,C6;
D1,D2;
Temp, Press, Alt;
_raw_press, _raw_temp;
```

functions

```
public:
_init();
_update(tnow);
_read();
_calculate();
get_altitude();
get_pressure();
get_temperature() ;
private:
_spi_read_adc();
_spi_write(reg);
_spi_read_16bits(reg);
```

클래스 정의

클래스 선언

- MS5611.h

MS5611.h

```
#ifndef __MS5611_H__
#define __MS5611_H__
#define MS5611_CS 40
#define CMD_MS5611_RESET 0x1E
#define CMD_MS5611_PROM_Setup 0xA0
#define CMD_MS5611_PROM_C1 0xA2
#define CMD_MS5611_PROM_C2 0xA4
#define CMD_MS5611_PROM_C3 0xA6
#define CMD_MS5611_PROM_C4 0xA8
#define CMD_MS5611_PROM_C5 0xAA
#define CMD_MS5611_PROM_C6 0xAC
#define CMD_MS5611_PROM_CRC 0xAE
#define CMD_CONVERT_D1_OSR4096 0x48
#define CMD_CONVERT_D2_OSR4096 0x58
```

```
class MS5611{
    uint32_t  _spi_read_adc();
    void      _spi_write(uint8_t reg);
    uint16_t  _spi_read_16bits(uint8_t reg);
    uint32_t  _s_D1,  _s_D2;
    uint8_t   _state;
    uint32_t  _timer;
    bool      _sync_access,  _updated;
    uint16_t  C1,C2,C3,C4,C5,C6;
    uint32_t  D1,D2;
    int16_t   Temp;
    int32_t   Press, Alt;
    int32_t   _raw_press;
    int32_t   _raw_temp;
  public:
    bool      _init(void );
    void      _update(uint32_t tnow);
    uint8_t   _read();
    void      _calculate();
    float     get_altitude();
    int32_t   get_pressure();
    int16_t   get_temperature() ;
};
#endif
```

클래스 멤버함수 구현

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

MS5611.cpp

```
#include <SPI.h>
#include "MS5611.h"
uint32_t MS5611::_spi_read_adc() {
  uint8_t dump,byteH,byteM,byteL;
  uint32_t return_value;
  uint8_t addr = 0x00;
  digitalWrite(MS5611_CS, LOW);
  dump = SPI.transfer(addr);
  byteH = SPI.transfer(0);
  byteM = SPI.transfer(0);
  byteL = SPI.transfer(0);
  digitalWrite(MS5611_CS, HIGH);
  return_value = (((uint32_t)byteH)<<16) | (((uint32_t)byteM)<<8) | (byteL);
  return return_value;
}
void MS5611::_spi_write(uint8_t reg){
  uint8_t dump;
  digitalWrite(MS5611_CS, LOW);
  dump = SPI.transfer(reg);
  digitalWrite(MS5611_CS, HIGH);
}
...
```

APM코드

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

주요 코드

- 객체 생성
  - MS5611 baro :
- SPI 설정
- 객체 초기화
  - baro._init():

Baro_5611_class.ino <1/2>

```
#include "MS5611.h"
#include <SPI.h>
uint32_t curtime=0, prevtime=0,count=0;
MS5611 baro;
void setup() {
  pinMode(12,OUTPUT); pinMode(11,OUTPUT);
  Serial.begin(115200);
  SPI.begin();
  SPI.beginTransaction(SPISettings(5000000, MSBFIRST,SPI_MODE0));
  baro._init();
  prevtime=micros();
  digitalWrite(27,HIGH);
}
```

APM코드

Dept. of Mechanical System Design, Seoul National University of Science and Technology.

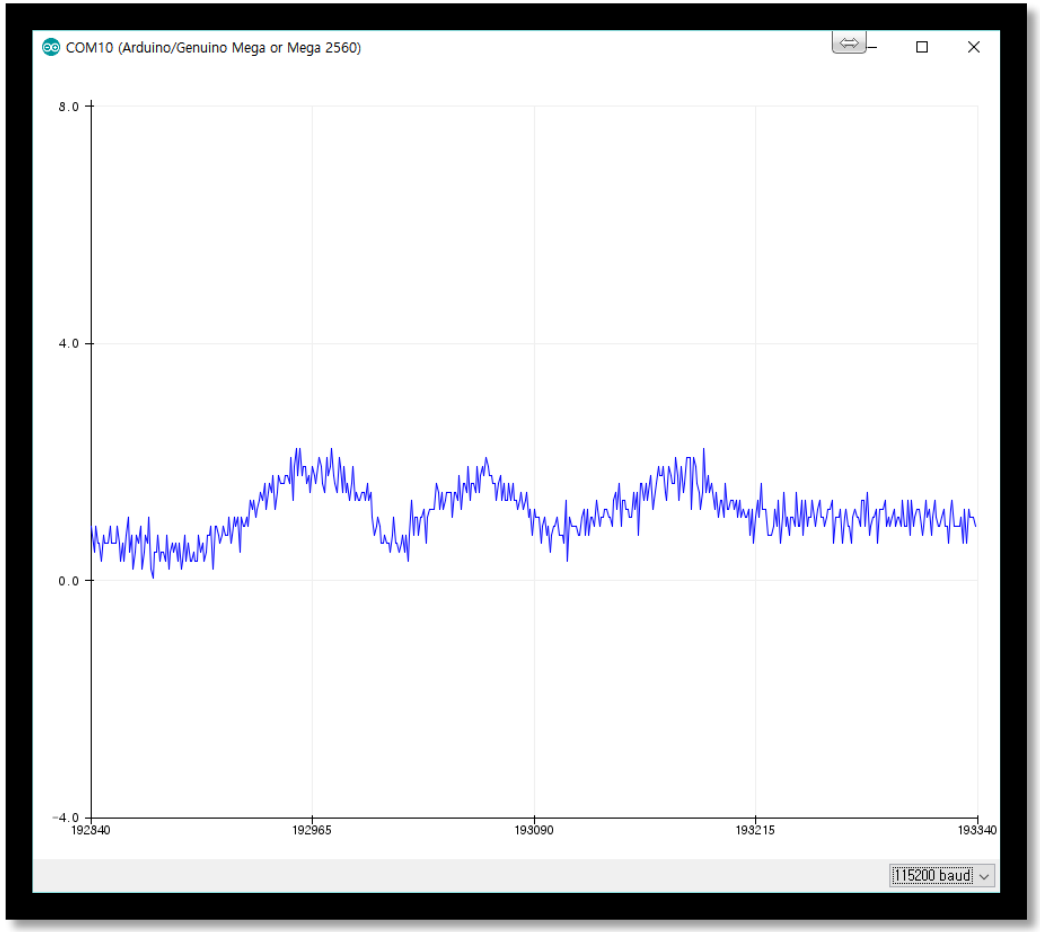
- 읽기
  - baro._update(curtime)
- 변수에 보관
  - baro._read()
- 고도 읽기
  - baro.get_altitude()
  - 기본 고도를 뺌 (-5420)

Baro_5611_class.ino <2/2>

```
void loop() {
  do {curtime=micros();}  while (curtime-prevtime<1000);
    prevtime=curtime;
    if (count% 10==0)
        digitalWrite(11,HIGH);
           baro._update(curtime);
           digitalWrite(11,LOW);
     if (count% 20==0) {
         digitalWrite(12,!digitalRead(12));
         baro._read();
         Serial.println(baro.get_altitude()-5420.);
     }
  count ++;
}
```

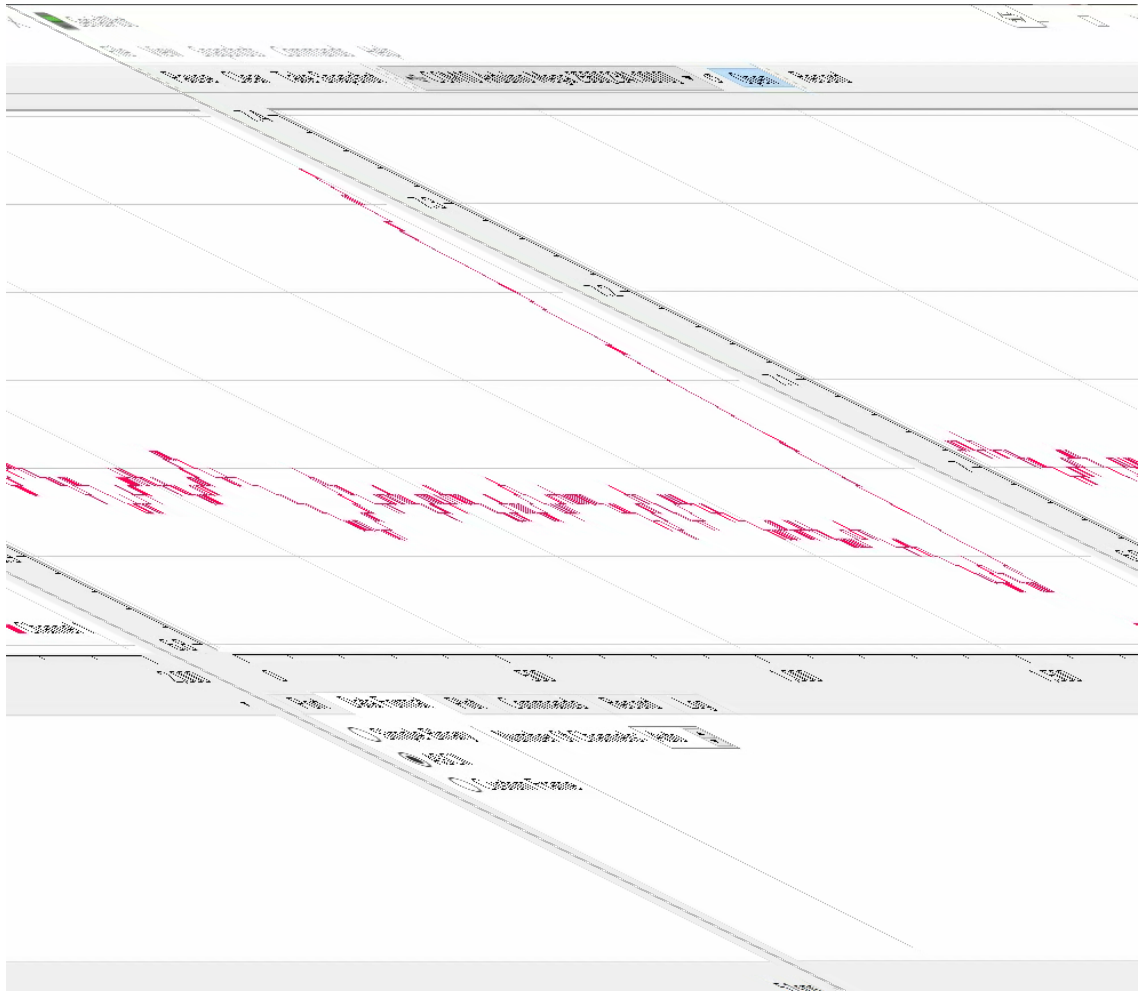
측정 실행결과

- 조건
  - 지면의 고도를 뺌.



결과 동영상

Dept. of Mechanical System Design, Seoul National University of Science and Technology.



THANK YOU

Powerpoint is a complete presentation graphic package it gives you everything you need to produce a professional-looking presentation