1. 아두이노 센서와 파이썬 이용한 실시간 데이터 수집

1.1. 아두이노 센서 활용하기

1.1.1. 센서의 종류: 무궁무진. 이게 있을까? -> 있다

1.1.2. 좋은 센서 고르기

1. 측정 범위
2. 측정 단위
3. 측정 오차: 뒤이어 배울 통계학적 지식 활용하면 유용
4. 측정 방식: 과학적 지식이 필요

1.1.3. 아두이노 코드 : C, CPP

시리얼 통신으로 값 바로 받기, 센서 자체 라이브러리 활용, 구글 검색(구글은 프로그래머의 바이블)

1.1.4. 아두이노에 센서 라이브러리 설치 방법

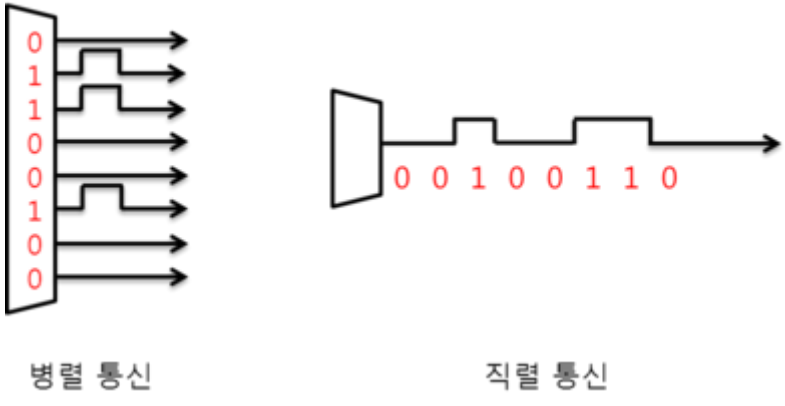
1. 스케치 → 라이브러리 포함하기 → .ZIP라이브러리 추가...
2. 스케치 → 라이브러리 포함하기 → 라이브러리 관리 → 센서의 모델명으로 검색(부품명 아닌 영어+숫자로 된 이름)

- 파일 → 예제로 들어가면 해당 라이브러리를 활용한 코드가 있다.

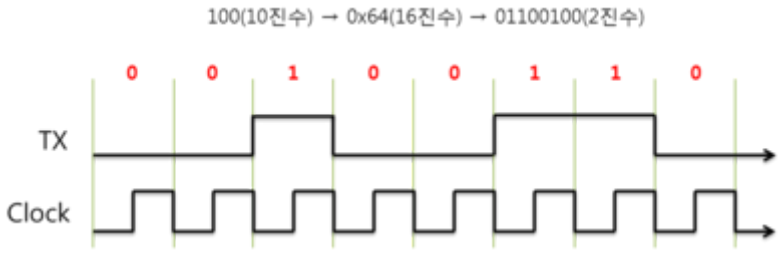
1.2. 시리얼 통신 활용하기

1.2.1. 시리얼 통신? 비트 단위로 직렬 방식+동기식으로 통신하는 것.

100(10진수) → 0x64(16진수) → 01100100(2진수)



병렬 통신 직렬 통신



- 직렬방식: 한 번에 한 비트씩
- 동기식: 보내는 쪽과 받는 쪽이 서로 호흡(시간)을 맞춰서

출처:https://m.blog.naver.com/yuyyulee/220301424499

1.2.2. 통신 흐름

아두이노 코드 컴파일 → 아두이노 ↔ 시리얼 통신 ↔ 파이썬 ↔ 데이터 출력 및 저장. 아두이노 제어.

1.2.3. 아두이노 상의 코드

```
void setup() {

   Serial.begin(9600);  //괄호안은 서로 맞출 시간의 단위(Baud rate라고도 함)

  }

  void loop() {

    input = Serial.read();      //시리얼 통신으로 값을 받아서 저장하기

    Serial.println(값); //값을 보내기(프린트하기)
  }
```

1.2.4. 파이썬 상의 코드

In [ ]:

```python
import serial   #pySerial 관련 라이브러리 불러오기. pip install pyserial
from datetime import *  #시간 관련 라이브러리

serial1 = serial.Serial('포트이름', 9600) #Baud rate를 아두이노와 똑같이 맞춘다.

if serial1.readable():          #시리얼 통신이 정상 작동중이라면

    # 아두이노로 신호를 보내기(신호로 제어 가능)
    input_value = input()
    input_value.encode('utf-8') #아두이노가 읽을 수 있는 형태로 데이터를 변환
    serial1.write(input_value)        #입력받은 값을 시리얼 통신으로 아두이노에 보낸다.

    # 아두이노에서 데이터를 받아 저장
    output_value = serial1.readline()
    output_value = output_value.decode() #아두이노 신호를 읽을 수 있는 형태로 해석
    print(output_value)

    # 실시간으로 그래프를 그려보자.

    # 다음 코드로 파일에 값을 저장 할 수도 있다.
    with open(f"{datetime.now.strftime('%Y-%m-%d %H:%M')}.txt",'a') as f:
        f.write(output_value)
```

1.3. SD카드 사용하기

- 라이브러리 활용: 파일 → 예제 → SD의 Files, ReadWrite를 활용하자. (핀 번호를 잘 맞추고, 쓸 데이터만 잘 입력해주면 된다.)

2. 실제로 실습해보기

2.1. 센서

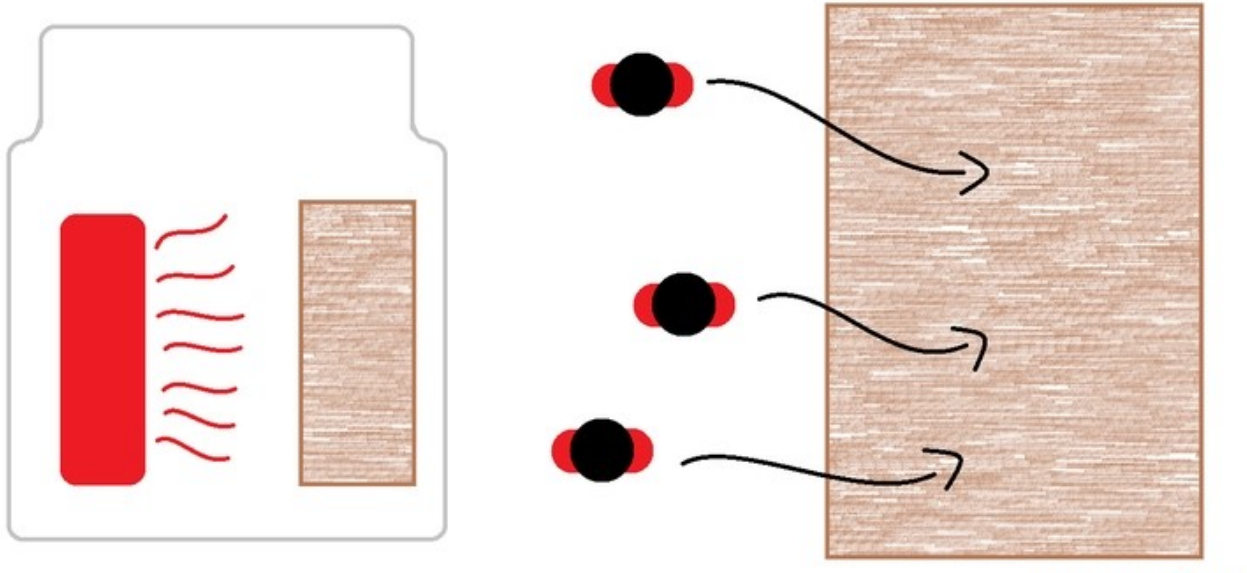
1. 이산화탄소 센서 - analogRead()로 읽을 수 있다.
2. 메탄가스 센서 - analogRead()로 읽을 수 있다.
3. MicroSD card: 아두이노 파일 > 예제 > SD > ReadWrite 예시 참고

2.1.1. 이산화탄소 센서의 원리

전압값(mV)이 아날로그 입력값으로 들어옴.

특수한 금속판에 열을 가해 달라붙는 이산화탄소/메탄 양에 따라 저항이 변화함.

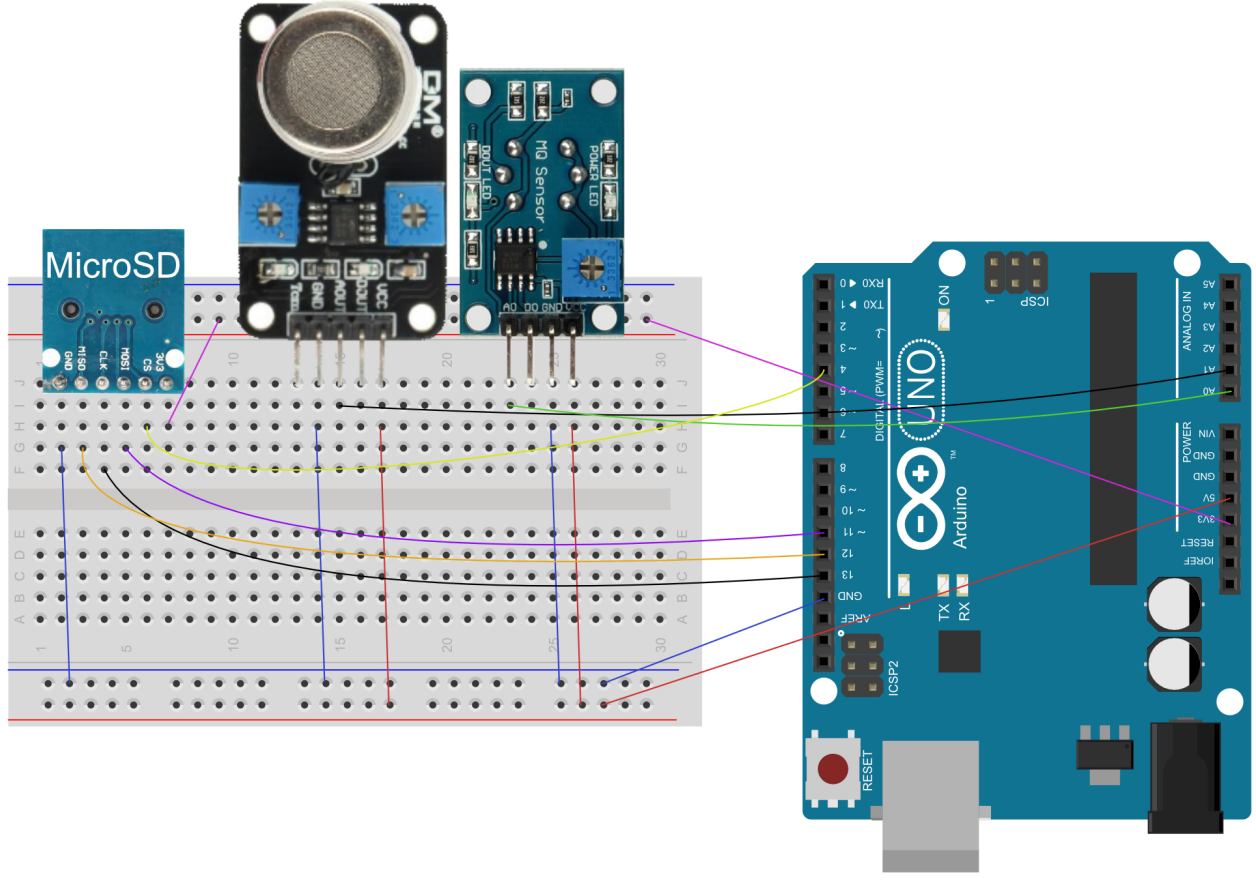
많이 달라 붙을 수록 저항값이 내려감 → 전압 떨어짐.



2.1.2. 측정값(MV)을 PPM 변환하기

- MQ 센서
- CO2 센서

2.2. 회로 구성하기



읽은 값을 txt파일에 시간과 함께 저장만하면 끝.

기판의 몇 가지 용어 설명

- VCC(Voltage of Common Collector): 전원 공급(5V, V3V 등)
- GND(Ground) : 접지, -극
- A0/AOUT(Analog Output): 아날로그 출력. 센서가 측정한 값을 아날로그 신호로 출력해 준다.
- CLK(Serial Clock) : Clock을 전송해주는 신호(from Master)
- MISO(Master Input Slave Output) : Master 입력 신호.
- MOSI(Master Output Slave Input) : Master 출력 신호.
- CS(Chip Select) : Serial Flash를 사용하기 위해서 Chap을 선택하는 신호.

MicroSD 통신에 대해 더 자세히 알고 싶다면: https://blog.daum.net/trts1004/12108902

파일 쓰기 코드

```
/*
  SD card read/write

  This example shows how to read and write data to and from an SD card file
  The circuit:
   SD card attached to SPI bus as follows:
 ** MOSI - pin 11
 ** MISO - pin 12
 ** CLK - pin 13
 ** CS - pin 4 (for MKRZero SD: SDCARD_SS_PIN)

  created   Nov 2010
  by David A. Mellis
  modified 9 Apr 2012
  by Tom Igoe

  This example code is in the public domain.

*/

#include <SPI.h>
#include <SD.h>

File myFile;

void setup() {
  // 시리얼 통신 열기:
  Serial.begin(9600);
  Serial.print("MicroSD카드를 준비하는 중...");
```

```
  // MicroSD카드가 연결되지 않은 경우
  if (!SD.begin(4))
  {
    Serial.println("MicroSD카드가 연결되지 않았습니다.");
    // 연결되지 않은 경우 동작을 아예 멈춤. 연결 후 다시 시도할 것.
    while (1);
  }
  Serial.println("연결 완료");

  // 쓰기 모드로 파일 열기. 한 번에 한 파일만 열 수 있고 다른 파일을 열려면 열려 있는 파일을 닫아
야 함.
  myFile = SD.open("test.txt", FILE_WRITE);

  // 파일 열기가 성공하면:
  if (myFile)
  {
    Serial.print("test.txt 에 작성을 시작합니다...");
    // 이 곳에서 원하는 값들을 작성한다. 이 부분을 루프로 가져가면 계속해서 작성하게 할 수 있다.
    // println은 텍스트 파일에 입력하고 싶은 것을 한 줄씩 입력해준다.
    myFile.println("테스트 1, 2, 3.");
    // 파일을 닫는다 :
    myFile.close();
    Serial.println("done.");
  }
  else
  {
    // 파일 열기가 실패한 경우:
    Serial.println("test.txt을 열 수 없습니다.");
  }

}

void loop() {
  // nothing happens after setup
}
```

아날로그 입력 읽기 코드

```
float val = 0;

void setup() {
  // 시리얼 통신 열기:
  Serial.begin(9600);
  Serial.print("아날로그 입력을 준비하는 중...");
}

void loop() {

  // A0핀의 아날로그 값을 읽어서 val에 저장
  val = analogRead(A0);

  // 시리얼 화면에 읽은 값을 출력한다.
  Serial.print("입력 값:");
  Serial.println(val);

  // 1000ms=1초 간 딜레이를 준다.
  delay(1000);
}
```