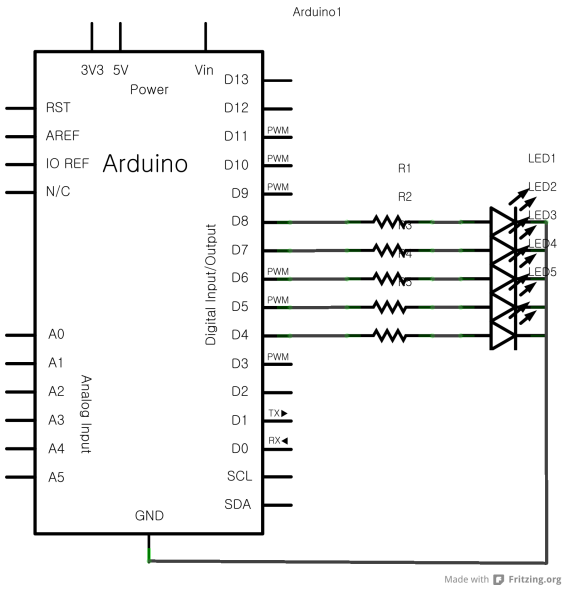
스케메틱




프로그램 코드

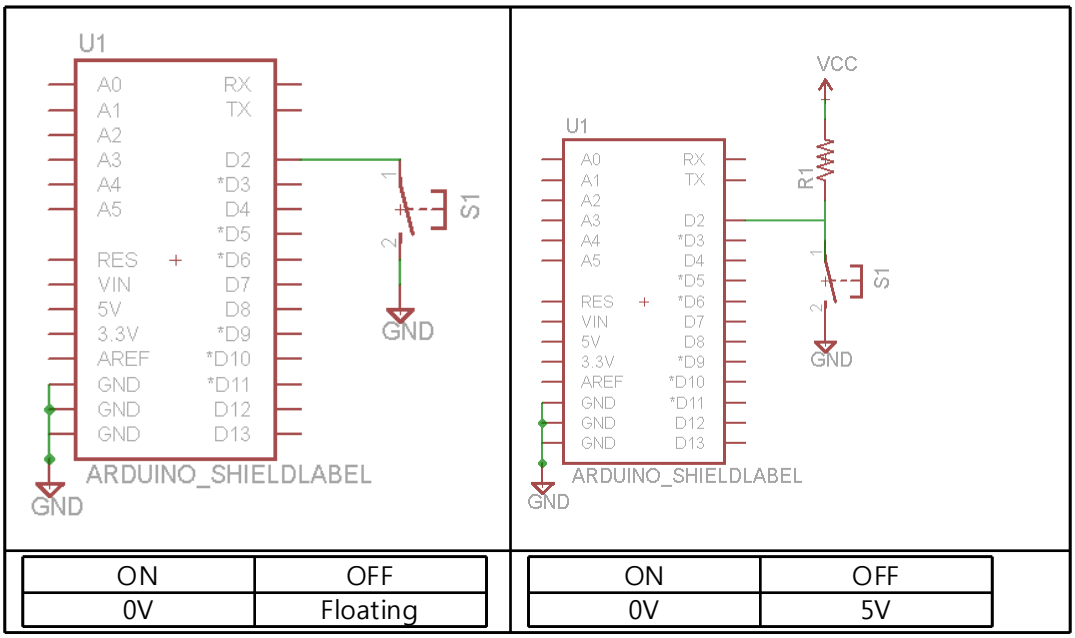
```
int del = 100;
void setup()
{
 for (int i=4; i<=8; i++) {
  pinMode(i, OUTPUT);
 }
}

void loop()
{
 for (int i=4; i<=8; i++) {
    digitalWrite(i, HIGH);    // turn on LED on pin i
    delay(del);                 // wait (length determined by vaue of 'del')
    digitalWrite(i, LOW);     // turn it off
 }
  for (int i=7; i>=5; i--) {
    digitalWrite(i, HIGH);    // turn on LED on pin i
    delay(del);                 // wait (length determined by vaue of 'del')
    digitalWrite(i, LOW);     // turn it off
 }
}
```

### 라. 버튼 입력 받기

풀업 저항 / 풀다운 저항

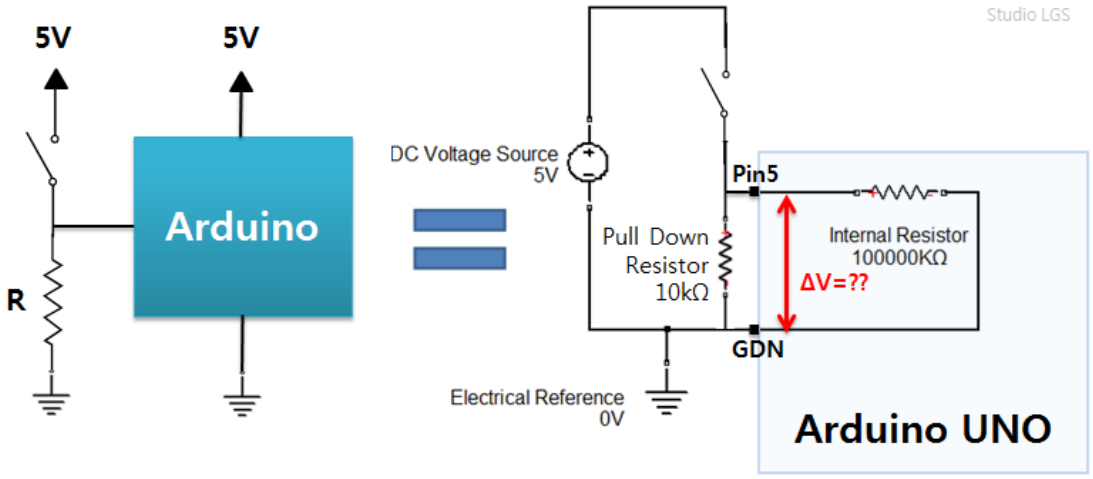
버튼이나 스위치 등의 전자적인 접촉이 발생하는 장치의 외부 입력을 받기 위해서는 digitalRead 함수를 사용하여 그 상태를 알 수 있게 된다. 저항을 연결하지 않고 스위치의 입력을 그대로 받아서 처리하는 경우 그 값은 일정하지 않은 상태를 나타내게 되며 이는 회로의 불안전한 동작을 유발시키게 된다. 이러한 문제점을 방지하기 위하여 다음과 같은 풀업저항과 풀다운 저항을 연결하는 기법이 있다.

풀업저항

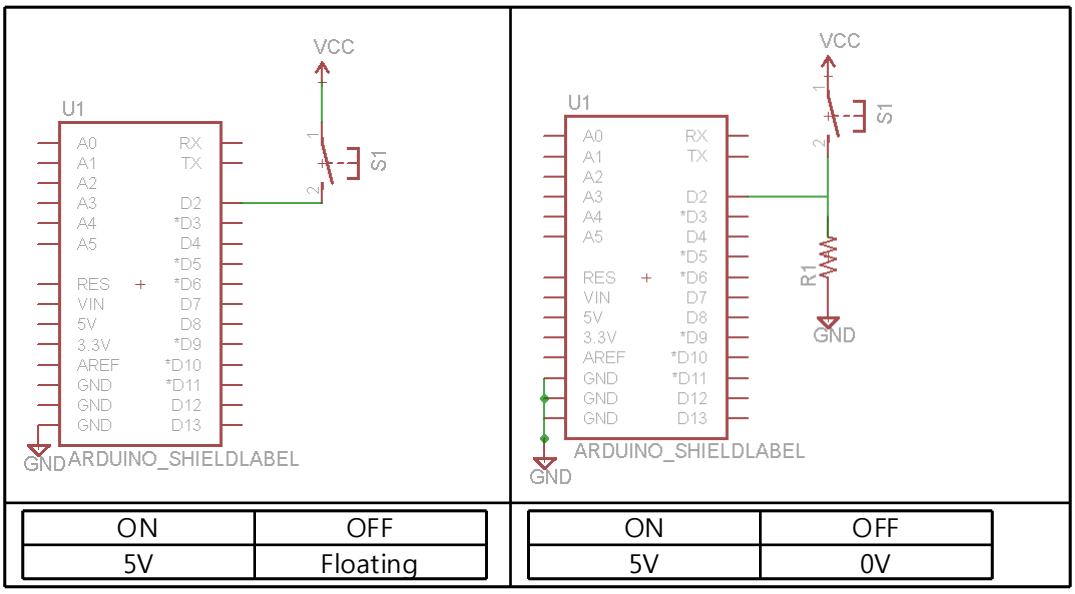



| ON | OFF |
|---|---|
| 0V | Floating |

| ON | OFF |
|---|---|
| 0V | 5V |

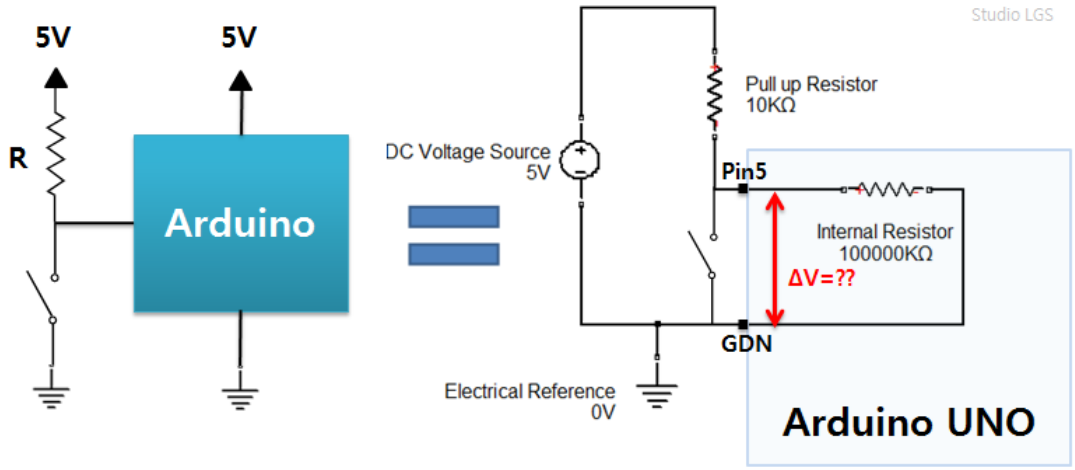


풀다운 저항




| ON | OFF |
|---|---|
| 5V | Floating |

| ON | OFF |
|---|---|
| 5V | 0V |

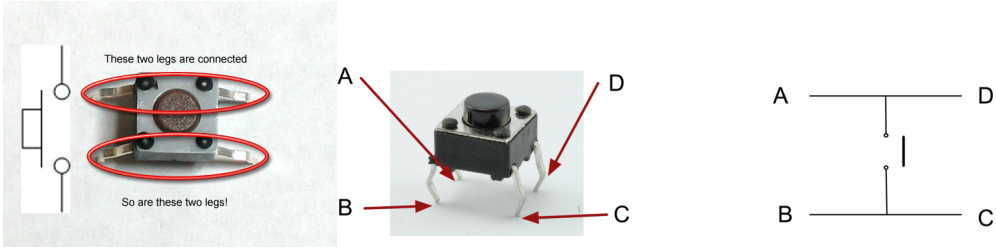



필요부품

아두이노보드, LED, 저항(220옴, 10K옴), 스위치, 케이블

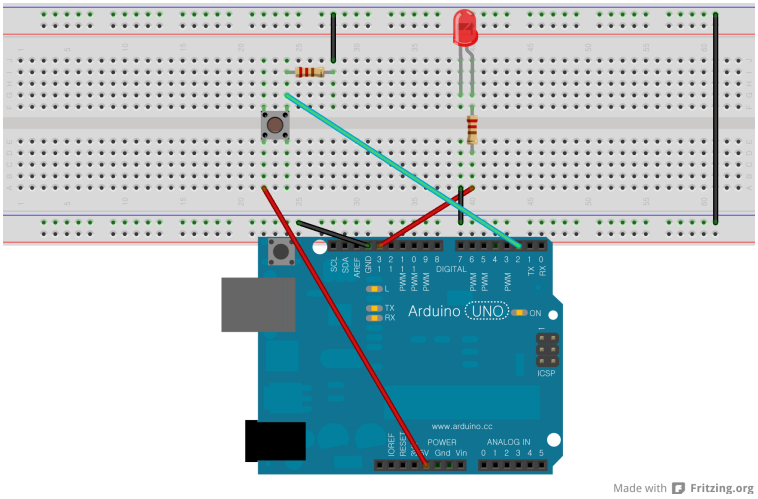
Push Switch

- 넓은 쪽의 다리는 서로 연결이 되어 있는 구조이다.

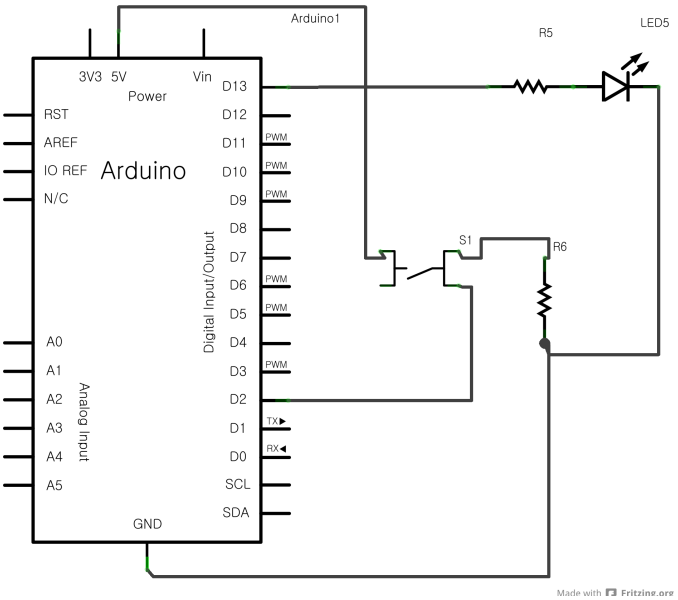



브레드보드연결

-풀 다운 저항을 연결하여 스위치가 오프일 때 입력이 0임을 확실히 보장해준다.




스케메틱




프로그램코드

- 스위치를 누를 때 만 13번 포트에 연결된 LED에 불이 들어오게 된다.

```
const int led = 13;
const int button = 2;
int val = 0;
void setup()
{
  pinMode(led, OUTPUT);
  pinMode(button, INPUT);
}
void loop ()
{
  val = digitalRead(button);

  if (val == HIGH)
    digitalWrite(led, HIGH);
  else
    digitalWrite(led, LOW);
}
```

도전과제1
위의 회로의 구조를 풀업 저항형태로 변경하고 동일하게 작동하도록 변경하시오.

도전과제2
스위치를 누를 때 마다 이전 상태를 변경하는 작업 - 이번 작업에서는 스위치를 누를 때 마다 LED의 상태가 변경되도록 해보자. 상태를 유지하는 변수 값을 보관하고 있다가 스위치의 상태 변화에 따라 이 값을 반영하도록 한다.
다음과 같은 스케치를 작성하고 업로드 후 예상대로 회로가 동작하는지 테스트 해보자.

```
const int led = 13;
const int button = 2;
int val = 0;
int state = 0;
void setup()
{
  pinMode(led, OUTPUT);
  pinMode(button, INPUT);
}
void loop () {
  val = digitalRead(button);

  if (val == HIGH)
    state = 1- state;

  if (state == 1)
    digitalWrite(led, HIGH);
  else
    digitalWrite(led, LOW);
}
```

```
const int led = 13;
const int button = 2;
int val = 0;
int old_val = 0;
int state = 0;
void setup()
{
  pinMode(led, OUTPUT);
  pinMode(button, INPUT);
}
void loop () {
  val = digitalRead(button);
  //이전 값을 저장해두고 이 값을 비교한다.
  if ((val == HIGH) && (old_val == LOW)) {
    state = 1- state;
    delay(10);
}
  old_val = val;
  if (state == 1)
    digitalWrite(led, HIGH);
  else
    digitalWrite(led, LOW);
}
```

아두이노의 MPU는 생각보다 빠르며, 위의 loop문의 digitalRead 함수는 그 상태가 변화를 것을 수천번 읽게 되며, 이는 회로의 불안정한 상태를 가져오게 되는 것이다. 따라서, 회로의 동작은 의도한 대로 동작하지 않게 된다. 오른쪽의 개선된 코드는 이전의 값을 저장해 두고 스위치가 눌려졌고 또한 이전의 상태가 서로 다를 때만 상태를 변경시켜 LED의 변경이 제대로 이루어지게 하고 있다.