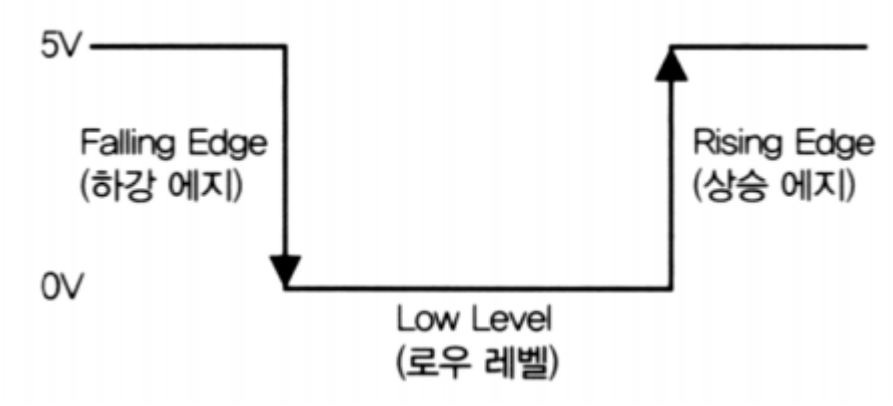
외부 인터럽트

외부 인터럽트

❖ 외부 인터럽트 감지 신호

아두이노 보드						ATmega328P MCU 핀 번호 및 포트명
아두이노 핀 명	디지털 입출력 핀	아날로그 입력 핀	PWM 출력 핀	외부 인터럽트 핀	시리얼 통신 핀	
3	3		3	INT1		5. PD3/INT1/OC2B
2	2			INT0		4. PD2/INT0



외부 인터럽트

❖ 외부 인터럽트 감지 신호

신호 분류	신호 상태
LOW 레벨	OFF
하강 에지	OFF → ON
상승 에지	ON → OFF
상태 변화	OFF → ON 또는 ON → OFF

외부 인터럽트

void attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)			
기능	외부 인터럽트에 대한 핀 번호, 동작 모드, 인터럽트 함수 지정		
매개변수	pin	외부 인터럽트를 사용할 수 있는 핀 번호 (우노보드에서는 2번핀과 3번핀 사용 가능) ※ pin에 상수 값이나 변수를 넣어 digitalPinToInterrupt(pin) 함수를 호출하면, 인터럽트로 사용할 수 있는 핀이면 인터럽트 번호를 사용할 수 없으면 –1 값이 반환된다.	
	ISR	인터럽트 발생시 자동 실행되는 함수명	
	mode	인터럽트 발생 모드	
		LOW	핀에 입력되는 신호가 LOW일 때
		CHANGE	핀에 입력되는 신호가 변할 때 (LOW→HIGH 또는 HIGH→LOW)
		RISING	LOW에서 HIGH로 변할 때(상승에지)
		FALLING	HIGH에서 LOW로 변할 때(하강에지)
리턴 값	없음		

외부 인터럽트

void attachInterrupt(interrupt, ISR, mode)			
기능	외부 인터럽트에 대한 ID 번호, 동작 모드, 인터럽트 함수 지정		
매개변수	interrupt	인터럽트 ID번호 (우노보드에는 2개 0(2번핀) 또는 1(3번핀))	
	ISR	인터럽트 발생시 자동 실행되는 함수명	
	mode	인터럽트 발생 모드	
		LOW	핀에 입력되는 신호가 LOW일 때
		CHANGE	핀에 입력되는 신호가 변할 때 (LOW->HIGH 또는 HIGH->LOW)
		RISING	LOW에서 HIGH로 변할 때(상승에지)
		FALLING	HIGH에서 LOW로 변할 때(하강에지)
리턴 값	없음		

외부 인터럽트

void deattachInterrupt(digitalPinToInterrupt(pin))		
기능	지정된 인터럽트를 해제	
매개변수	pin	해제할 핀 번호
리턴 값	없음	

void deattachInterrupt(interrupt)		
기능	지정된 인터럽트를 해제	
매개변수	interrupt	해제할 인터럽트 ID 번호
리턴 값	없음	

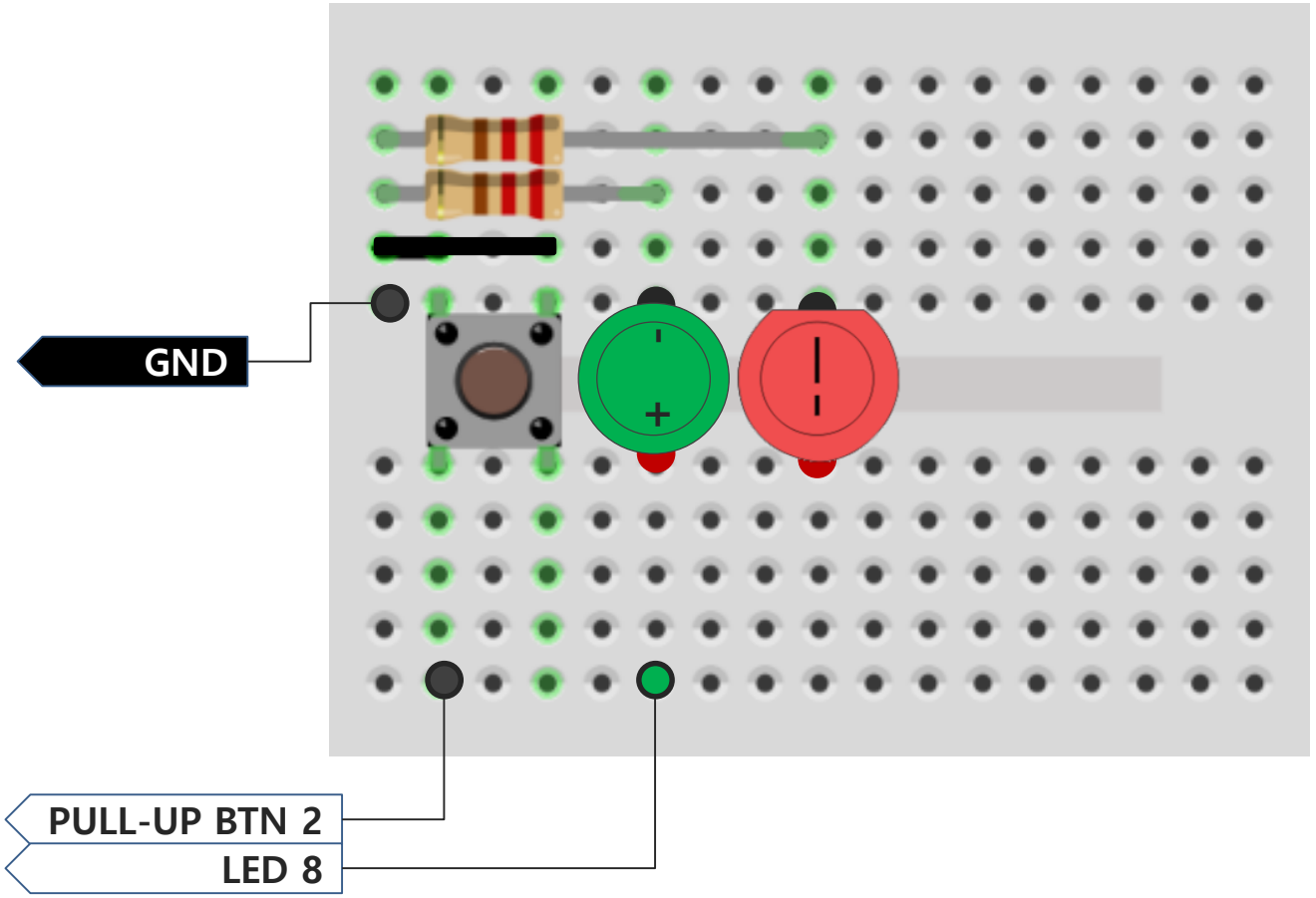
외부 인터럽트

void interrupts()	
기능	noInterrupts()로 비활성화된 인터럽트를 모두 활성화시키는 함수
매개변수	없음
리턴 값	없음

void noInterrupts()	
기능	전체 인터럽트를 비활성화시키는 함수
매개변수	없음
리턴 값	없음

실습1: Switch 눌려질 때마다 LED on/off 점멸

❖ 회로도



실습1: 외부 인터럽트 확인 - 채터링 발생

❖ ex01/app.ino

```
const int sw_pin = 2;
int count =0;

// 외부 인터럽트1 처리 함수
void flash(void) {
    count++;
    Serial.println(count);
}

void setup() {
    Serial.begin(115200);
    pinMode(sw_pin, INPUT_PULLUP);
    // 외부 인터럽트 설정
    attachInterrupt(digitalPinToInterrupt(sw_pin), flash, FALLING);
}

void loop() {}
```

실습2: 외부 인터럽트 확인 - 채터링 제거

❖ **ex01/app.ino**

```
const int sw_pin = 2;
int count =0;

int t1, t2;  // 인터럽트 발생 시간 저장

// 외부 인터럽트1 처리 함수
void flash(void) {
    // 채터링 체크 : 200ms 이내에 스위치가 또 눌려진 상태이면 무시
    t2 = millis();        // 현재 시간 저장

    // 인터럽트 시간 간격 체크
    if((t2 - t1) < 200) return;   // 200ms 보다 작으면 무시
    else t1 = t2;                 // 인터럽트 발생 시간 갱신

    count++;
    Serial.println(count);
}
```

실습2: 외부 인터럽트 확인 - 채터링 제거

❖ **ex01/app.ino**

```
void setup() {
    Serial.begin(115200);

    pinMode(sw_pin, INPUT_PULLUP);
    attachInterrupt(digitalPinToInterrupt(sw_pin), flash, FALLING);

    t1 = millis(); // 프로그램 시작 시간 저장
}

void loop() {}
```

실습3: 인터럽트로 LED on/off 점멸 (채터링)

❖ ex03/app.ino

```
#include <Led.h>

Led led(8);
boolean  led_st = LOW;

const int sw_pin = 2;
int t1, t2;

void flash(void) {
    // 채터링 처리
    t2 = millis();
    if((t2 - t1) < 200) return;
    else t1 = t2;

    led_st = !led_st;      // LED 상태 반전
    led.setValue(led_st);
}
```

실습3: 인터럽트로 LED on/off 점멸 (채터링)

❖ ex03/app.ino

```
void setup() {
    pinMode(sw_pin, INPUT_PULLUP);
    attachInterrupt(digitalPinToInterrupt(sw_pin), flash, FALLING);
    t1 = millis();
}

void loop() {}
```

실습4: Button 클래스에 인터럽트처리 추가

❖ Button.h

```
…

class Button {
protected:
        …
    unsigned long t1;

public:
    Button(int pin);
    void setCallback(button_callback_t callback);
    int read();
    void check();
    void attachInterrupt(button_callback_t callback, int mode);
    bool debounce();
};
```

실습4: Button 클래스에 인터럽트처리 추가

❖ Button.cpp

```
void Button::attachInterrupt(button_callback_t callback, int mode) {
    ::attachInterrupt(digitalPinToInterrupt(pin), callback, mode);
    t1 = millis();
}

bool Button::debounce() {
    unsigned long t2 = millis();
    if((t2-t1) < 200) return false;

    t1 = t2;
    return true;
}
```

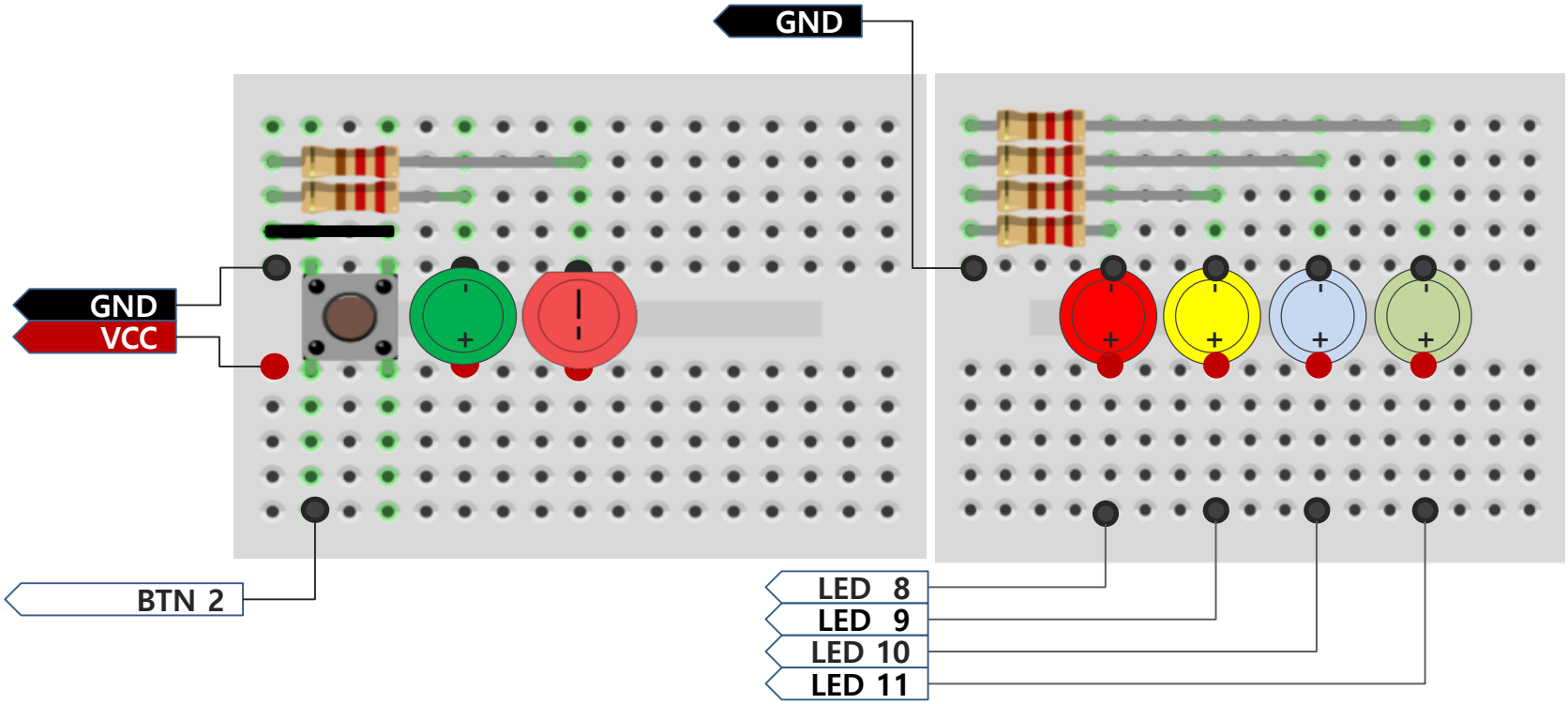
실습4: Button 클래스에 인터럽트처리 추가

❖ ex04/app.ino

```
#include <Led.h>
#include <Button.h>

Led led(8);
Button btn(2);

boolean  led_st = LOW;

void flash(void) {
    if(!btn.debounce()) return;

    led_st = !led_st;      // LED 상태 반전
    led.setValue(led_st);
}

void setup() {
    btn.attachInterrupt(flash, FALLING);
}

void loop() {}
```

실습3: Switch 눌려질 때마다 4개의 LED 순차 점멸

❖ 회로도



실습5: Switch 눌려질 때마다 4개의 LED 순차 점멸

❖ ex05/app.ino

```
#include <Led.h>
#include <Button.h>

Led leds[] = {
    Led(8), Led(9), Led(10), Led(11),
};

Button btn(2);

int out_no = -1;

void move_led() {
    out_no = (++out_no)%4;          // 다음 출력 패턴 번호 설정
    Serial.println(out_no);
    for(int n = 0;n < 4;n++){
        leds[n].setValue(n==out_no);
    }
}
```

실습5: Switch 눌려질 때마다 4개의 LED 순차 점멸

❖ **ex05/app.ino**

```
void flash(void) {
    if(!btn.debounce()) return;
    move_led();
}

void setup() {
    btn.attachInterrupt(flash, FALLING);
}

void loop() {}
```