

C h a p t e r

# 04



---

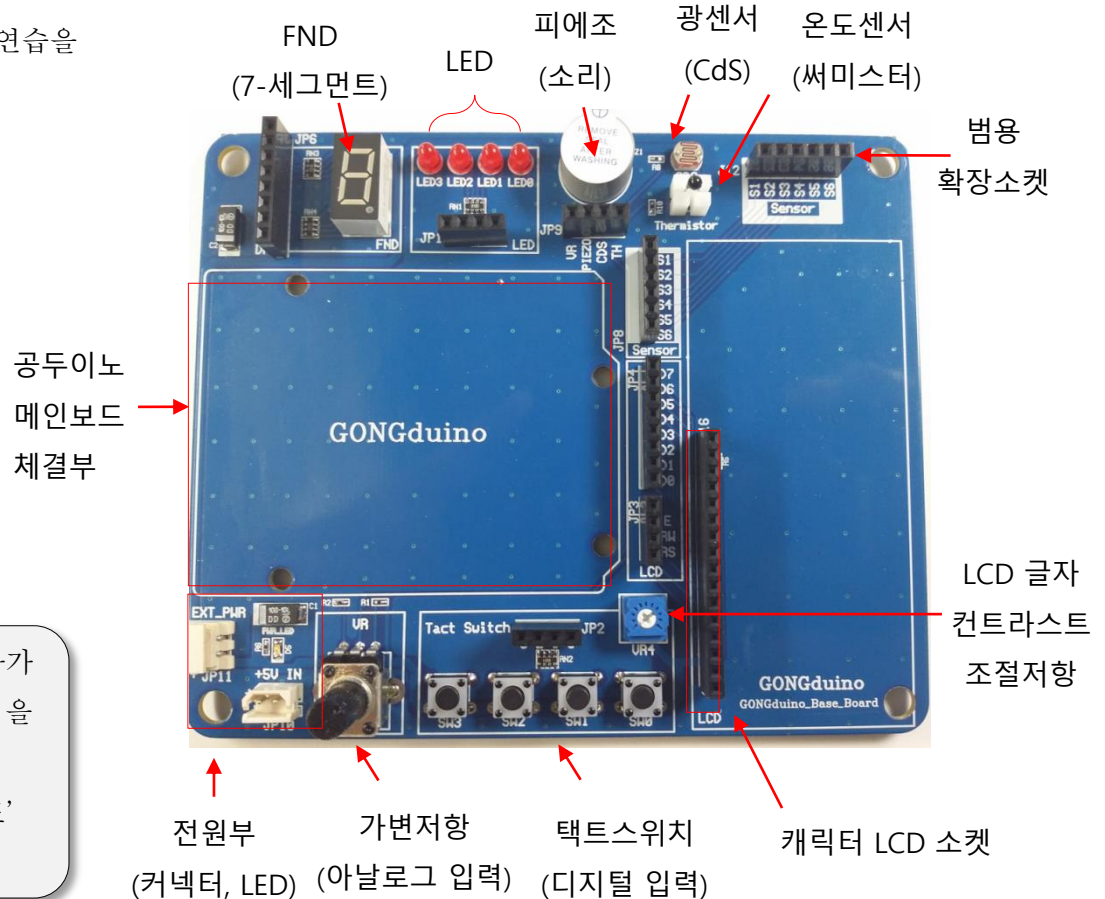
디지털 신호출력(digitalWrite)

## 공두이노 베이스 보드

기본적으로 많이 사용되는 소자(LED, 스위치 등)를 연습할 수 있는  
공두이노 베이스 보드(Gongduino-base-board)에 대해 알아봅니다.

### ■ 베이스 보드의 구조

공두이노를 이용하여 각종 소자에 대한 프로그램 연습을  
위해 베이스 보드(Gongduino-base-board)를  
사용하며, 보드에 탑재된 소자는 다음과 같습니다.

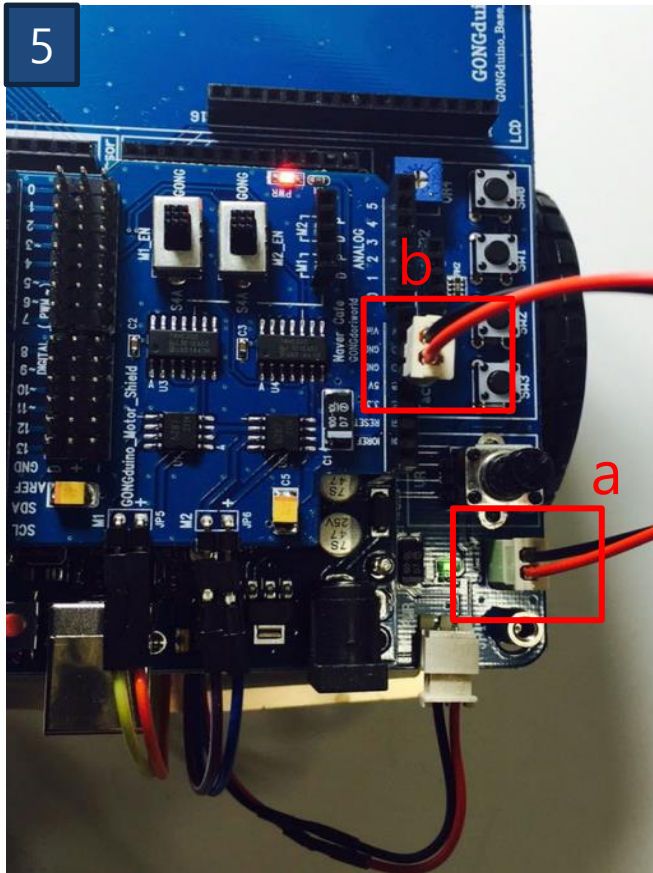


- ✓ 공급되는 베이스 보드에 따라 위 그림에서 소자가 없는 경우도 있습니다. 이 때, ‘범용확장소켓’을 활용합니다.
- ✓ 공두이노 제어보드를 간략하게 ‘공두이노 보드’라고 부릅니다.

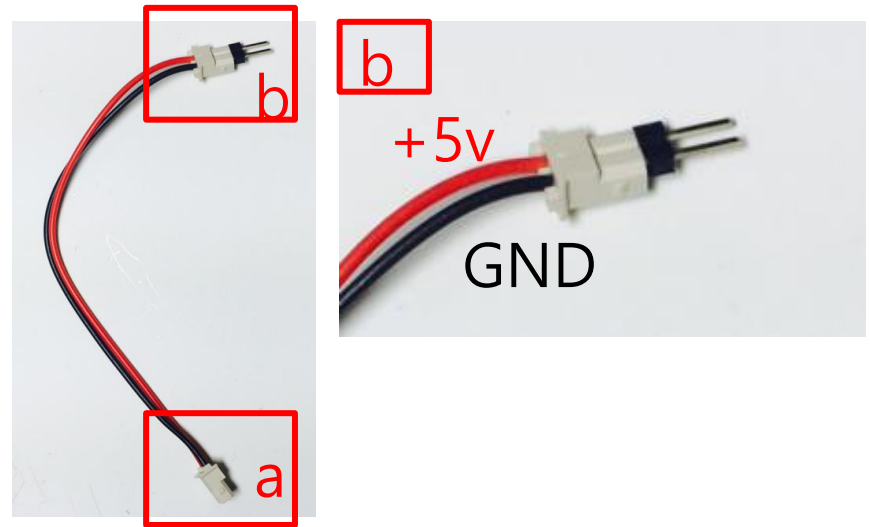
# 09

## 공두이노 베이스 보드

보드류에 신호 및 전원배선을 해보자.



베이스보드의 전원과 메인보드의 전원을 연결한다.  
즉 배터리→베이스보드→메인보드로 연결된다.



전원연결선을 a는 a와 b는 b와 연결.

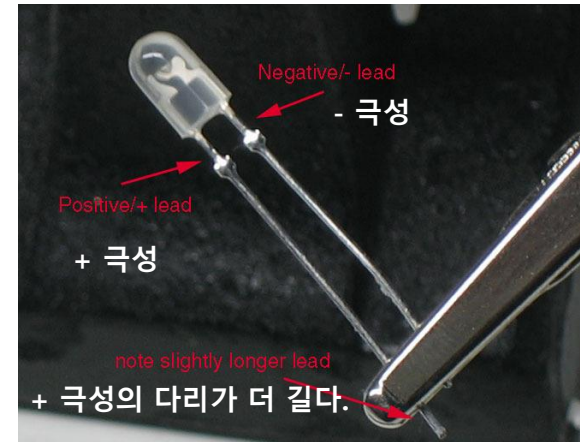
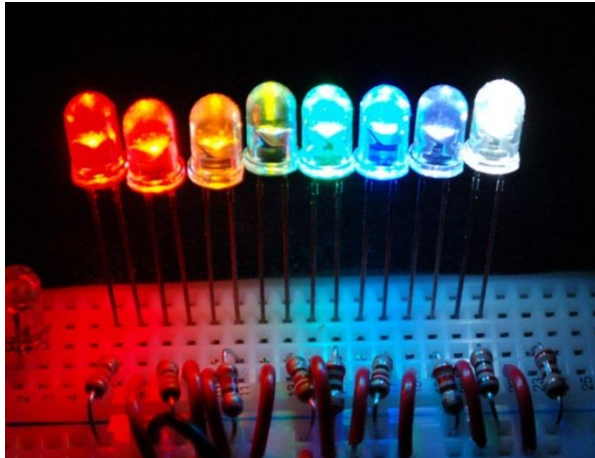
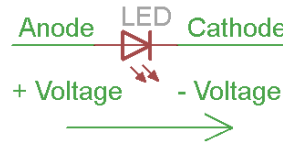
- b부분 연결시 모터 쉴드의 전원 극성을 주의한다.
- 전원 차단 시 베이스보드쪽의 물렉스를 뺐는다.
- 전체적으로 전원이 인가되면 바퀴가 돌 수 있으므로 적절히 바퀴를 띄운다.

LED 소자의 특성에 대해 알아보고, 공두이노를 이용하여 LED 소자를 제어하여 봅니다.

## ▪ LED 소자

LED는 작은 불빛을 발산하는 소자로, 다양한 색상의 LED 종류가 존재하며, 크기에 따라 아주 소형부터 대형에 이르기까지 많은 종류가 있습니다. 특히, 제어 장치에서는 LED가 소모하는 전류가 적고, 제어하기가 쉬워 가장 많이 사용되는 출력 소자 중의 하나입니다.

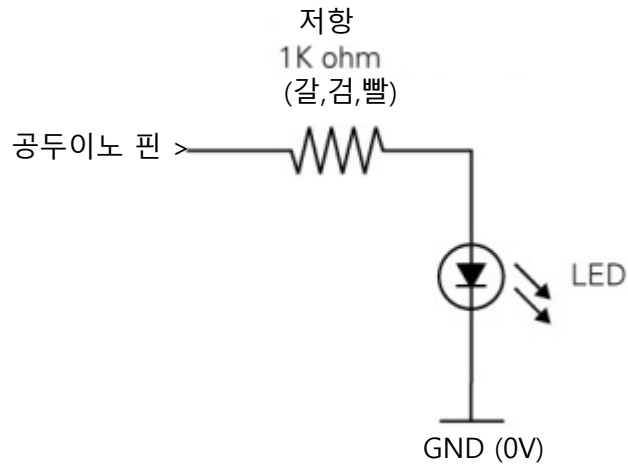
LED는 다이오드의 특성이 있으므로, 극성(방향성)이 존재하며, 긴 다리(애노드, Anode)가 +, 짧은 다리(캐소드, Cathode)가 - 이며 전류는 +에서 -로 흐르도록 회로를 구성합니다.



+5V DC 전압을 사용하는 디지털 회로에서 LED 소자는 저항과 함께 연결하는데, 보통 470옴 또는 1K옴의 저항을 직렬로 연결합니다. 연결하는 저항의 값이 작을 수록 LED에 흐르는 전류의 양이 많아져 LED의 빛이 더 밝게 빛납니다.

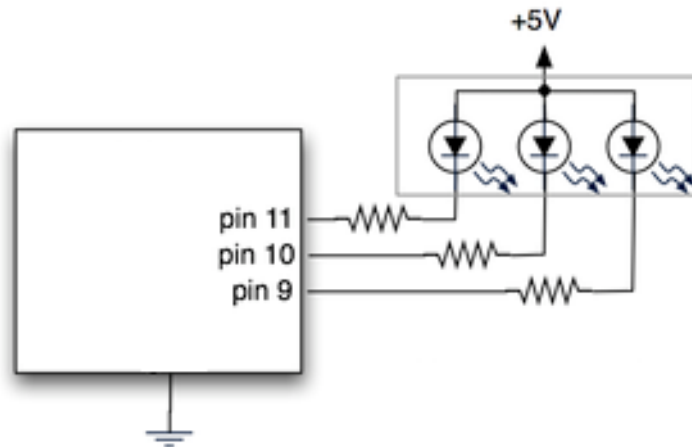
## ■ 공두이노 보드와 LED 연결

보통, 공두이노 보드의 디지털 핀(0번 ~ 13번)과 저항, LED를 아래 그림과 같이 연결합니다.



좌측 그림의 경우에는 공두이노 핀에서 HIGH (+5V) 신호를 출력하면 LED가 켜지고, LOW (0V) 신호를 출력하면 LED가 꺼집니다.

⇒ 정논리 방식



공두이노의 출력 핀에 HIGH 신호를 출력하면 LED가 꺼지고, LOW 신호를 출력하면 LED가 켜집니다.

⇒ 부논리 방식

이와 같은 회로는 LED 동작 전류를 공두이노의 마이크로 컨트롤러가 흡수(싱크전류, Sink current)하게 되므로, 마이크로 컨트롤러가 부담하는 전류를 줄여주어, 장시간의 시스템 동작이 안정적으로 됩니다.

# 실습 01

## 1개 LED ON/OFF/Blink

LED의 회로를 참고하여, 공두이노 보드로 1개 LED를 켜고, 끄며 이를 깜빡이는 프로그램을 실습합니다.

### 하드웨어 연결

1. 공두이노 보드의 13번 핀에는 LED(기판에 LED라고 표현되어 있는 녹색 LED)가 연결되어 있으므로, 별도의 하드웨어 구성은 하지 않습니다.
2. 13번 핀의 LED는 정논리 방식으로 회로가 연결되어 있습니다.

### 프로그램 작성 1

exam001

```
void setup() {  
    pinMode(13, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(13, HIGH);  
}
```

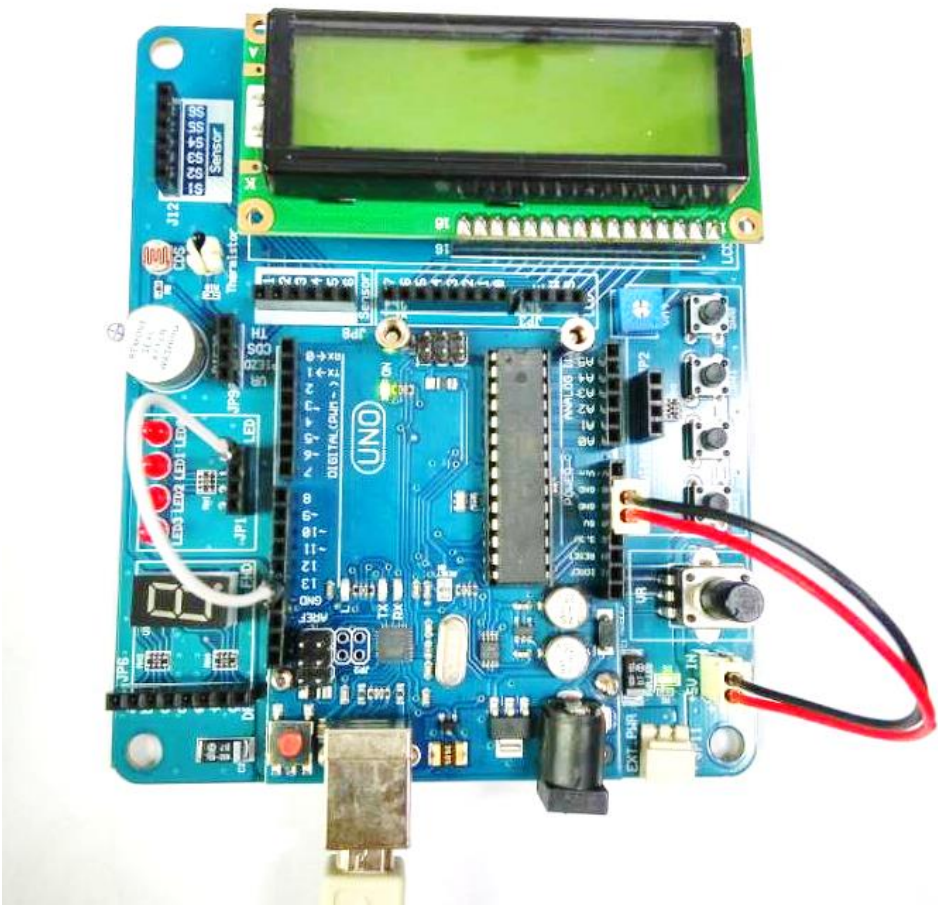
#### pinMode (pin, mode)

지정한 pin 을 입력(INPUT) 또는 출력(OUTPUT) 으로 설정합니다.  
특히, 입력일 때는 일반 입력(INPUT)과 내부 풀업 저항을 사용하는  
풀업입력(INPUT\_PULLUP)의 사용이 가능합니다.

#### digitalWrite (pin, value)

디지털 출력 pin 에 HIGH 또는 LOW 신호를 출력합니다.

- ✓ LED를 끄려면 digitalWrite 함수로 LOW 신호를 출력합니다.
- ✓ 프로그램 업로드시 PWR SEL 슬라이드 스위치는 USB 쪽, 우측 상단의 스위치도 USB(Default)쪽으로 위치시키십시오.





```
void setup() {  
    pinMode(13, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(13, HIGH);  
    delay(500);  
    digitalWrite(13, LOW);  
    delay(500);  
}
```

### delay (ms)

지정한 밀리초(ms, 1/1000초 단위) 만큼 프로그램의 수행을 지연합니다. 시간의 지정은 unsigned long (0-4,294,967,295)까지 가능합니다.

- ✓ loop 함수는 무한히 반복하므로, 위 프로그램 예제는 0.5초 LED를 켜고, 0.5초 LED를 끄는 동작을 무한 반복하게 됩니다.
- ✓ 시간을 조절하여 깜빡이는 속도를 바꾸어 봅시다.

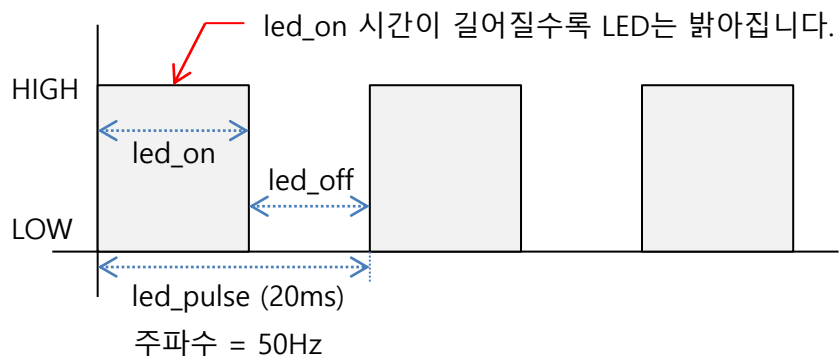


13번 핀의 LED를 3번만 깜빡이고, 정지합니다.

```
void setup() {  
    pinMode(13, OUTPUT);  
}  
  
void loop() {  
    for (int i=0; i<3; i++){  
        digitalWrite(13, HIGH);  
        delay(500);  
        digitalWrite(13, LOW);  
        delay(500);  
    }  
  
    while(1);           // 여기서만 무한히 반복 (정지)  
}
```

- ✓ for 문을 이용한 반복문의 사용법을 잘 익혀둡니다. - 횟수를 정확히 지정할 때 for 반복문을 사용합니다.
- ✓ while(1); 자체가 무한 반복을 하기 때문에 loop 함수 안의 while(1)은 프로그램을 더 이상 진행하지 않고 정지합니다.

13번 핀의 LED의 밝기를 밝게 또는 어둡게 켜봅니다.



```
int led_pulse = 20;
int led_on = 10; ← 숫자를 바꾸어서 동작시켜 봅니다.
int led_off = led_pulse - led_on;

void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);
  delay(led_on);
  digitalWrite(13, LOW);
  delay(led_off);
}
```

- ✓ 요즘 대부분의 LED를 사용하는 조명 장치에는 이와 같은 **펄스 구동 방식**(pulse) 으로 밝기를 조절합니다.
- ✓ 펄스 구동 방식은 ON/OFF 시간을 조절하기 때문에 구동 전류를 줄일 수 있어 에너지 절약의 효과가 높습니다.
- ✓ LED 밝기 조절을 위한 펄스는 보통 60Hz 이상 (120Hz 가 일반적)의 주기를 사용합니다.

# 실습 02

## 4개 LED 다루기

4개의 LED를 순차적으로 제어하거나, 임의로 제어하는 프로그램하는 방법을 실습합니다.

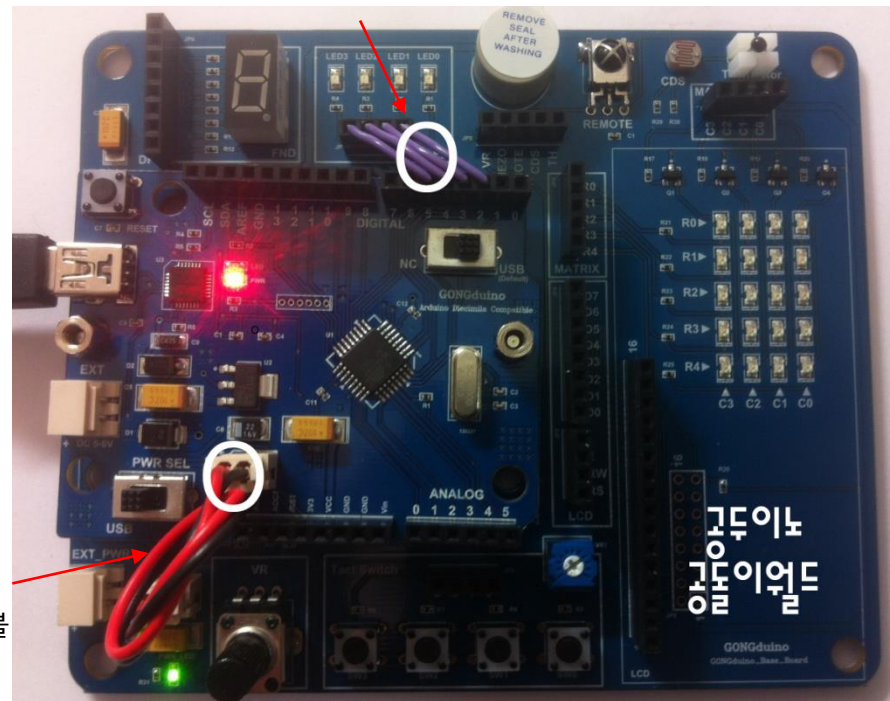
### 하드웨어 연결

1. 공두이노 보드와 베이스 보드(Gongduino-base-board)의 전원 커넥터를 케이블로 연결합니다.
2. 베이스 보드의 LED 4개를 연결선으로 공두이노 보드와 연결합니다.

연결선 (4개)

공두이노 보드	연결방향	베이스 보드
2번 PIN	→	LED0
3번 PIN	→	LED1
4번 PIN	→	LED2
5번 PIN	→	LED3

전원  
케이블



1. LED 4개를 0.5 초 간격으로 모두 깜빡입니다.

```
void setup() {  
    pinMode(2, OUTPUT);    // 2번-5번 핀 모두 출력 설정  
    pinMode(3, OUTPUT);  
    pinMode(4, OUTPUT);  
    pinMode(5, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(2, HIGH); // 2번-5번 핀 LED ON  
    digitalWrite(3, HIGH);  
    digitalWrite(4, HIGH);  
    digitalWrite(5, HIGH);  
  
    delay(500);            // 0.5초 시간 지연  
  
    digitalWrite(2, LOW); // 2번-5번 핀 LED OFF  
    digitalWrite(3, LOW);  
    digitalWrite(4, LOW);  
    digitalWrite(5, LOW);  
  
    delay(500);            // 0.5초 시간 지연  
}
```

✓ 반복되는 구문은 모두 for 문으로 바꾸어 봅니다.

## 2. for(반복) 문을 이용한 LED 4개 깜빡이기

```
void setup() {  
    for (int i=2; i<=5; i++){  
        pinMode(i, OUTPUT);  
    }  
}  
  
void loop() {  
    for (int i=2; i<=5; i++){  
        digitalWrite(i, HIGH);  
    }  
  
    delay(500);  
  
    for (int i=2; i<=5; i++){  
        digitalWrite(i, LOW);  
    }  
  
    delay(500);  
}
```

LED 4개에서 1개의 LED를 왼쪽으로 이동

```
int timer = 100;           // 이동 시간 0.1 초

void setup() {
    for (int i = 2; i <= 5; i++) {
        pinMode(i, OUTPUT);
    }
}

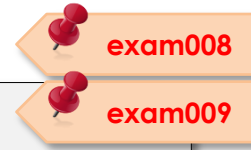
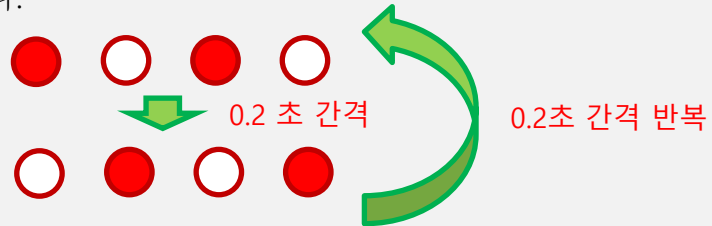
void loop() {
    for (int i = 2; i <= 5; i++) {
        digitalWrite(i, HIGH);
        delay(timer);
        digitalWrite(i, LOW);
    }
}
```

} 2-5번 핀 출력

} LED 왼쪽(←) 이동



✓ 아래와 같이 LED 4개가 동작하도록 프로그램을 작성합니다.





LED 4개에서 1개의 LED를 왼쪽/오른쪽으로 계속 이동

```
int timer = 100;           // 이동 시간 0.1 초

void setup() {
    for (int i = 2; i <= 5; i++) {
        pinMode(i, OUTPUT);
    }
}

void loop() {
    for (int i = 2; i <= 5; i++) {
        digitalWrite(i, HIGH);
        delay(timer);
        digitalWrite(i, LOW);
    }

    for (int i = 5; i >= 2; i--) {
        digitalWrite(i, HIGH);
        delay(timer);
        digitalWrite(i, LOW);
    }
}
```

} 2-5번 핀 출력

} LED 왼쪽(←) 이동

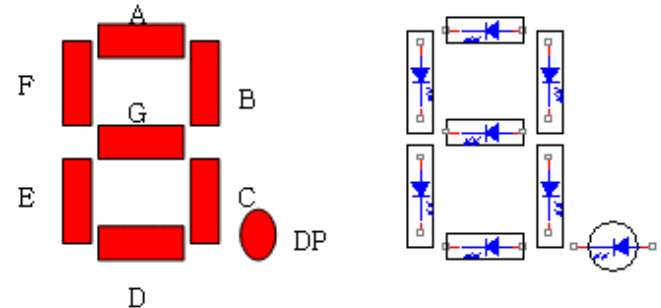
} LED 오른쪽(→) 이동

FND는 숫자 또는 간단한 알파벳을 표시하기 위한 배열된 LED 배열 소자로 이를 프로그램으로 제어하는 방법에 대해서 익히게 됩니다.

## ■ FND 소자

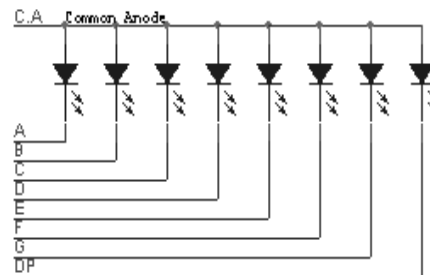
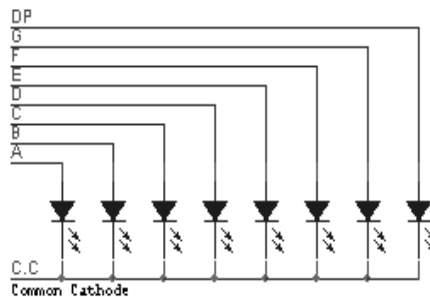
FND 또는 7세그먼트 LED는 7개의 세그먼트 LED들이 모여 숫자를 표시할 수 있고, 가격이 저렴하고 어두운 장소에서도 상태 관찰이 가능하므로 많이 사용됩니다.

일반적으로 7개의 세그먼트에 1개의 점(DOT)까지 하여 8개의 LED로 구성되어 있습니다.













위의 그림에서와 같이 8개의 LED가 8자의 숫자 모양으로 배치되어 있으며, 보통 8개의 LED는 A, B, C, D, E, F, G, DP의 이름으로 부르고 위와 같이 배치되어 있습니다.

FND의 LED는 8개의 LED의 한쪽 부분은 공통으로 접속되어 있는데, Cathode(-)가 공통으로 접속되는 경우(왼쪽)는 Common-Cathode형이고, Anode(+)가 공통으로 접속되는 경우는 Common-Anode형이라 합니다.



- ✓ 공두이노 베이스 보드에 장착된 FND는 Common-Anode 형이며, 공통 선이 VCC에 연결되어 있어, LED의 ON/OFF 동작은 “부논리” 입니다.

## ■ 부논리의 FND 숫자 표시

숫자	FND	A	B	C	D	E	F	G	D P	16 진수
0		0	0	0	0	0	0	1	1	0x03
1		1	0	0	1	1	1	1	1	0x9f
2		0	0	1	0	0	1	0	1	0x25
3		0	0	0	0	1	1	0	1	0x0d
4		1	0	0	1	1	0	0	1	0x99
5		0	1	0	0	1	0	0	1	0x49
6		0	1	0	0	0	0	0	1	0x41
7		0	0	0	1	1	1	1	1	0x1f
8		0	0	0	0	0	0	0	1	0x01
9		0	0	0	1	1	0	0	1	0x19

# 실습 03

## FND 에 숫자 출력하기

FND에 숫자를 출력하는 프로그램을 작성합니다.

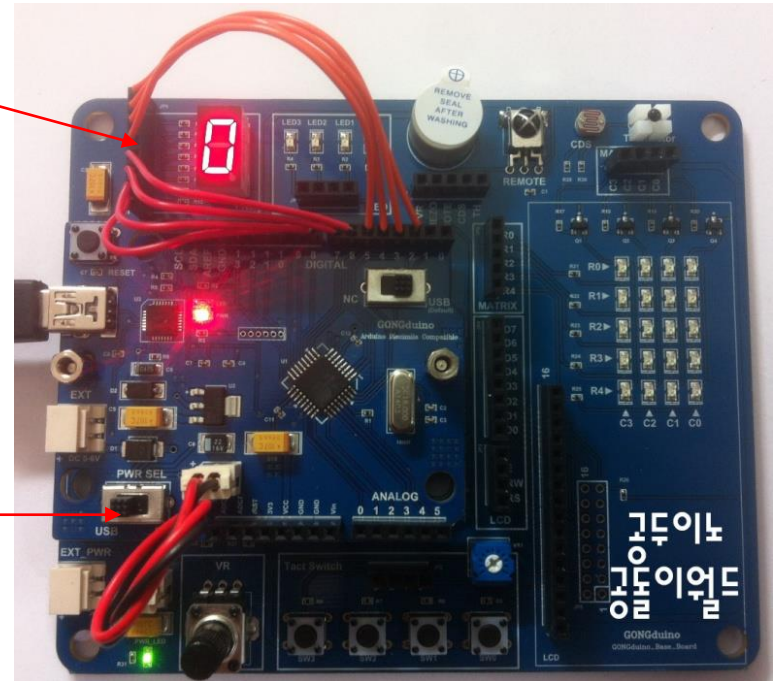
### 하드웨어 연결

1. 공두이노 보드와 베이스 보드 (Gongduino-base-board)의 전원 커넥터를 케이블로 연결합니다.
2. 베이스 보드의 FND 연결 핀 8개를 연결선으로 공두이노 보드와 연결합니다.

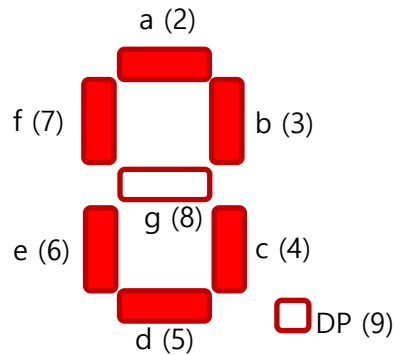
공두이노 보드	연결방향	베이스 보드
2번 PIN	→	FND a
3번 PIN	→	FND b
4번 PIN	→	FND c
5번 PIN	→	FND d
6번 PIN	→	FND e
7번 PIN	→	FND f
8번 PIN	→	FND g
9번 PIN	→	FND DP

연결선  
(8개)

전원  
케이블



FND에 숫자 '0' 을 표시합니다.



```
void setup() {
    for (int i = 2; i <= 9; i++) {
        pinMode(i, OUTPUT);
    }
}

void loop() {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
    digitalWrite(7, LOW);
    digitalWrite(8, HIGH);
    digitalWrite(9, HIGH);
}
```

숫자 '0'



✓ 숫자 '1' 부터 '9' 까지 하나씩 FND에 표시하는 프로그램을 작성합니다.

1초마다 FND에 숫자 '0' 부터 '9' 까지 반복 표시합니다.

- ✓ 앞의 FND 부논리 표를 참조합니다.
- ✓ FND 배열을 사용하여 for (반복)문을 사용합니다.

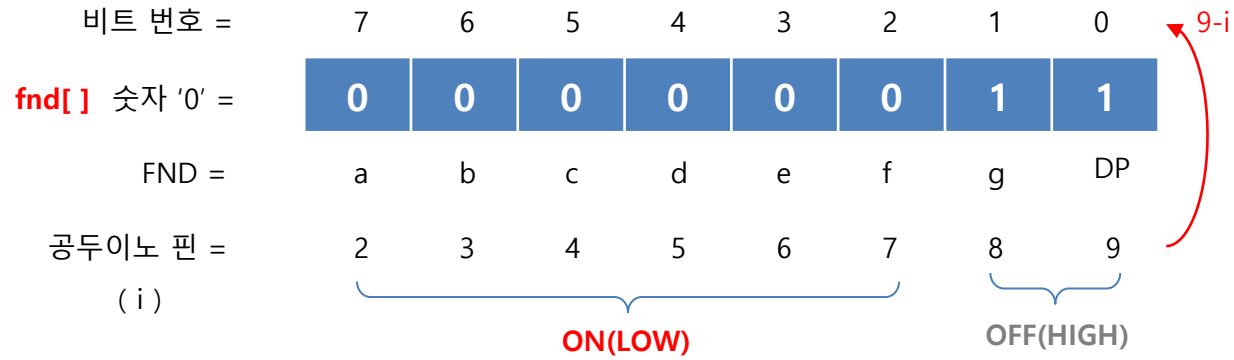
```
void setup() {  
    for (int i = 2; i <= 9; i++) pinMode(i, OUTPUT);  
}  
  
void loop() {  
    unsigned int fnd[] =  
        { 0x03, 0x9f, 0x25, 0x0d, 0x99, 0x49, 0x41, 0x1f, 0x01,  
          0x19};  
  
    for (int j = 0; j <= 9; j++) {  
        for (int i = 2; i <= 9; i++) {  
            if (bitRead(fnd[j], 9-i)) {digitalWrite(i, HIGH);}  
            else {digitalWrite(i, LOW);}  
        }  
        delay(1000);  
    }  
}
```

숫자  
0-9  
출력

2-9번 핀  
출력

bitRead (x, n )함수는 x 변수 또는 숫자에서 n 번째 비트 값을 읽어옵니다.

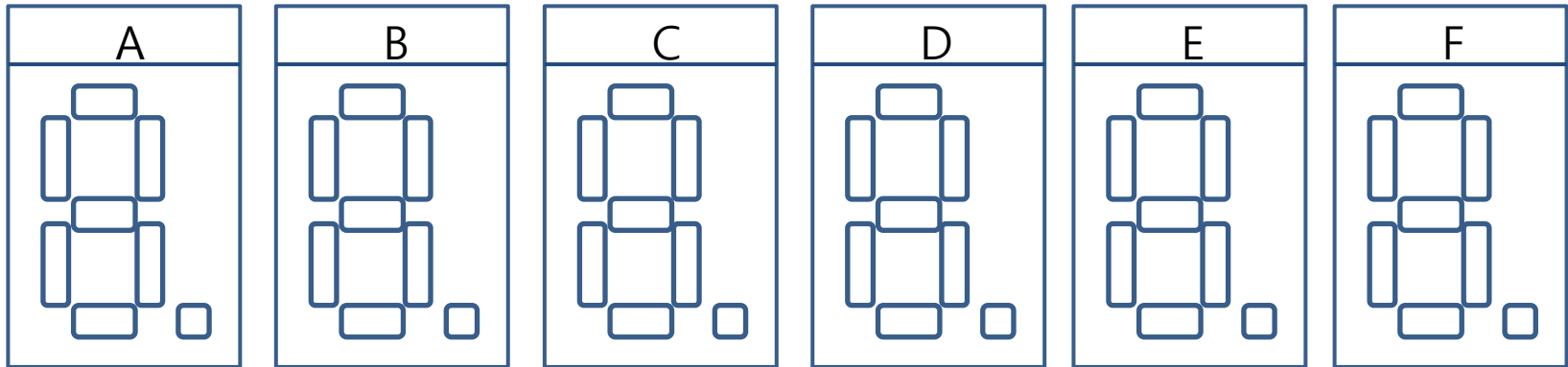
bitRead ?







숫자 '0' 부터 '9' , 알파벳 "A" 부터 "F" 까지 16진수 값을 1초마다 계속해서 FND에 표시하는 프로그램을 작성합니다.



- ✓ 위 그림을 이용하여 A 부터 F 까지 표시할 16진 수를 계산합니다.
- ✓ B와 D는 소문자로 표시합니다.
- ✓ 16진수 'A' 부터 'F' 까지의 수를 `fnf[]` 에 추가합니다.
- ✓ 16진수 '0' 부터 'F' 까지 표시하려면 반복하는 `j` 의 값이 15이어야 합니다.

1초마다 FND에 숫자 '0' 부터 '9' 까지 반복 표시합니다. 이 때, 0.5 초 마다 FND 오른쪽 아래의 점(.)을 깜빡입니다.

```
void setup() {
  for (int i = 2; i <= 9; i++) pinMode(i, OUTPUT);
}

void loop() {
  unsigned int fnd[] =
    { 0x03, 0x9f, 0x25, 0x0d, 0x99, 0x49, 0x41, 0x1f, 0x01,
      0x19};

  for (int j = 0; j <= 9; j++) {
    for (int i = 2; i <= 9; i++) {
      if (bitRead(fnd[j], 9-i)) {digitalWrite(i, HIGH);}
      else {digitalWrite(i, LOW);}
    }
    delay(500);
    digitalWrite(9, LOW);    // 점(.)을 ON
    delay(500);
  }
}
```