사물인터넷

Chapter 5.
7-세그먼트 표시장치

1. 7-세그먼트 표시장치
2. 1자리 7-세그먼트 2개 연결
3. 4자리 7-세그먼트 표시장치

학습목표

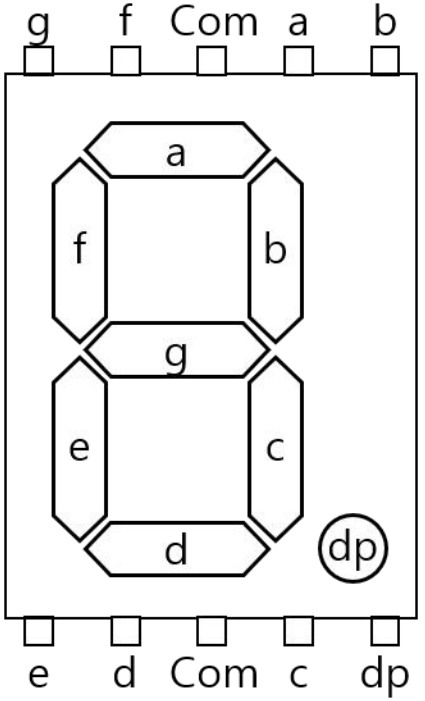
- 7-세그먼트 표시장치의 구조를 이해한다.
- 공통 양극 타입과 공통 음극 타입을 구별하고 적절하게 코딩할 수 있다.
- 7-세그먼트 표시장치의 회로를 바르게 연결할 수 있다.
- 7-세그먼트 표시장치 제어 코드를 이해하고 적절하게 활용할 수 있다.

1. 7-세그먼트 표시장치

- 7-세그먼트 표시장치
  - 숫자 혹은 영문 알파벳의 일부를 표시하기 위하여 7개의 선분 형태의 발광 다이오드(LED)를 '8' 모양으로 배치하여 만든 표시장치
  - 디지털 시계, 전자 계량기, 기본 계산기 및 숫자 정보를 표시하는 여러 전자 장치에 널리 사용
  - 숫자나 문자 표시는 8개 LED 세그먼트들 중 일부를 켜거나 끄는 것으로 제어하며 1바이트의 데이터로 구분

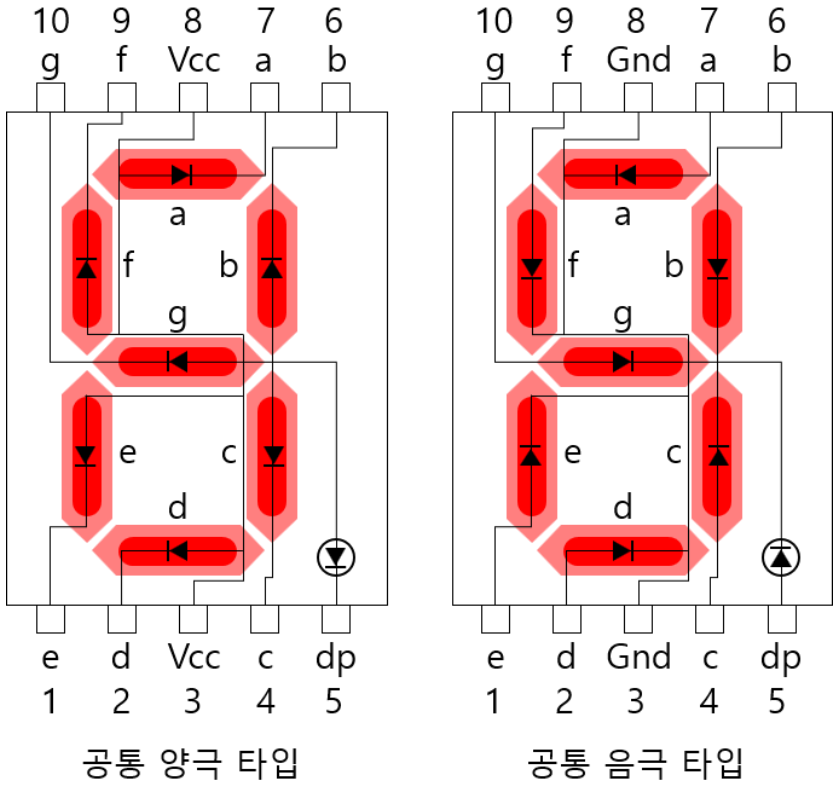
1. 7-세그먼트 표시장치

- 1자리 7-세그먼트 표시장치



1. 7-세그먼트 표시장치

- 7-세그먼트 표시장치 유형



공통 양극 방식(common anode)

- 7-세그먼트 표시장치의 공통 핀(Com)을 Vcc에 연결
- 공통 핀에 5V의 전원을 연결하고 a에서 g, 그리고 dp핀에 전압을 0V로 설정하면 LED가 켜지고, 5V로 설정하면 LED가 꺼짐

공통 음극 방식(common cathode)

- 7-세그먼트 표시장치의 공통 핀(Com)을 Gnd에 연결
- 공통 핀에 Gnd를 연결하고 a에서 g, 그리고 dp핀에 전압을 5V로 설정하면 LED가 켜지고, 0V로 설정하면 LED가 꺼짐

1. 7-세그먼트 표시장치

- 7-세그먼트 표준 디스플레이 코드
  - 본 교재의 실험은 공통 음극 방식을 사용

(예) 숫자 0

- 공통 음극 방식 :

  2진 코드 0b11111100

  16진 코드 0xFC

- 공통 양극 방식 :

  2진 코드 0b00000011

  16진 코드 0x03

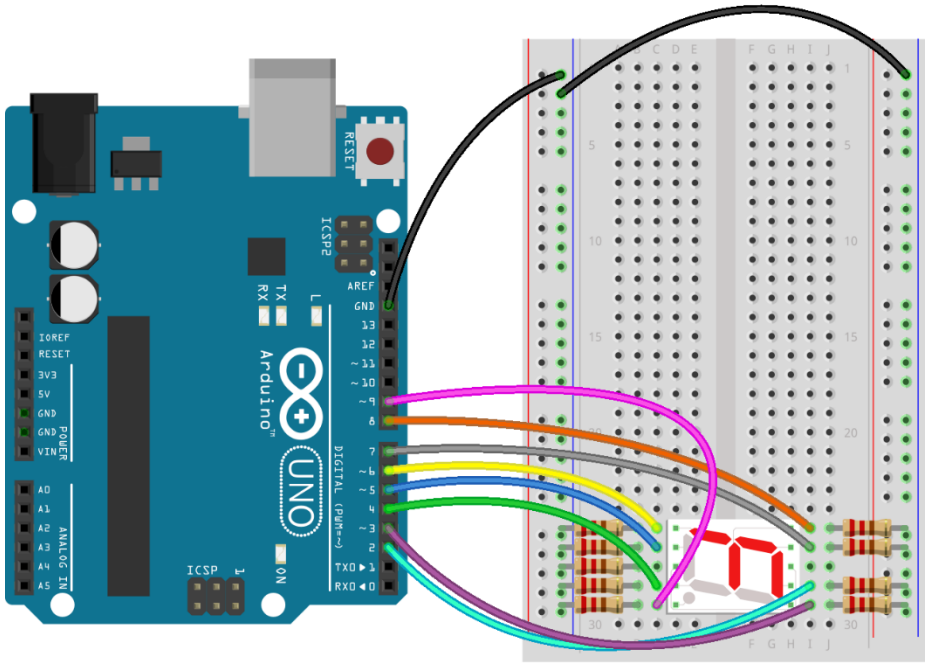
	D7	D6	D5	D4	D3	D2	D1	D0	16진코드
	a	b	c	d	e	f	g	dp	(C언어)
0	1	1	1	1	1	1	0	0	0xFC
1	0	1	1	0	0	0	0	0	0x60
2	1	1	0	1	1	0	1	0	0xDA
3	1	1	1	1	0	0	1	0	0xF2
4	0	1	1	0	0	1	1	0	0x66
5	1	0	1	1	0	1	1	0	0xB6
6	1	0	1	1	1	1	1	0	0xBE
7	1	1	1	0	0	1	0	0	0xE4
8	1	1	1	1	1	1	1	0	0xFE
9	1	1	1	1	0	1	1	0	0xF6
A	1	1	1	0	1	1	1	0	0xEE
b	0	0	1	1	1	1	1	0	0x3E
C	1	0	0	1	1	1	0	0	0x9C
d	0	1	1	1	1	0	1	0	0x7A
E	1	0	0	1	1	1	1	0	0x9E
F	1	0	0	0	1	1	1	0	0x8E

1. 7-세그먼트 표시장치

- 1자리 7-세그먼트 회로 결선

[필요한 HW장치 및 부품]

- 아두이노 우노 보드 1개
- 1자리 7-세그먼트 1개
- 220Ω 저항 9개
- 브레드보드 1개
- 약간의 점퍼선



7-세그먼트	Gnd	a	b	c	d	e	f	g	dp
아두이노	Gnd	2	3	4	5	6	7	8	9

1. 7-세그먼트 표시장치

- LED 요소를 차례대로 제어하기

  1초 간격으로 a→b→c→d→e→f→g→dp 순서로 LED 불 켜기

```
int pin = 2;    // 7-세그먼트의 LED 요소 a를 아두이노의 디지털 2번 핀과 연결
void setup() {
    for (int i = 2; i <= 9; i++) {
        pinMode(i, OUTPUT);

    digitalWrite(2, HIGH);    // 시작과 동시에 LED 요소 a에 불이 켜짐
}
void loop() {
    for(int i = 0 ; i < 8 ; i++) {
        for(int j = 0 ; j < 8 ; j++) {
            if (i == j) digitalWrite(pin+j, HIGH);      // 한 개의 LED에 불을 켜고
            else digitalWrite(pin+j, LOW);              // 나머지 LED의 불은 끈다.
        }
        delay(1000);    // 1초 지연
    }
}
```

1. 7-세그먼트 표시장치

- 0부터 9까지의 숫자 출력하기

```
byte numbers[10][8]= {
  {1, 1, 1, 1, 1, 1, 0, 0}, // 0
  {0, 1, 1, 0, 0, 0, 0, 0}, // 1
  {1, 1, 0, 1, 1, 0, 1, 0}, // 2
  {1, 1, 1, 1, 0, 0, 1, 0}, // 3
  {0, 1, 1, 0, 0, 1, 1, 0}, // 4
  {1, 0, 1, 1, 0, 1, 1, 0}, // 5
  {1, 0, 1, 1, 1, 1, 1, 0}, // 6
  {1, 1, 1, 0, 0, 1, 0, 0}, // 7
  {1, 1, 1, 1, 1, 1, 1, 0}, // 8
  {1, 1, 1, 1, 0, 1, 1, 0} // 9
};
```

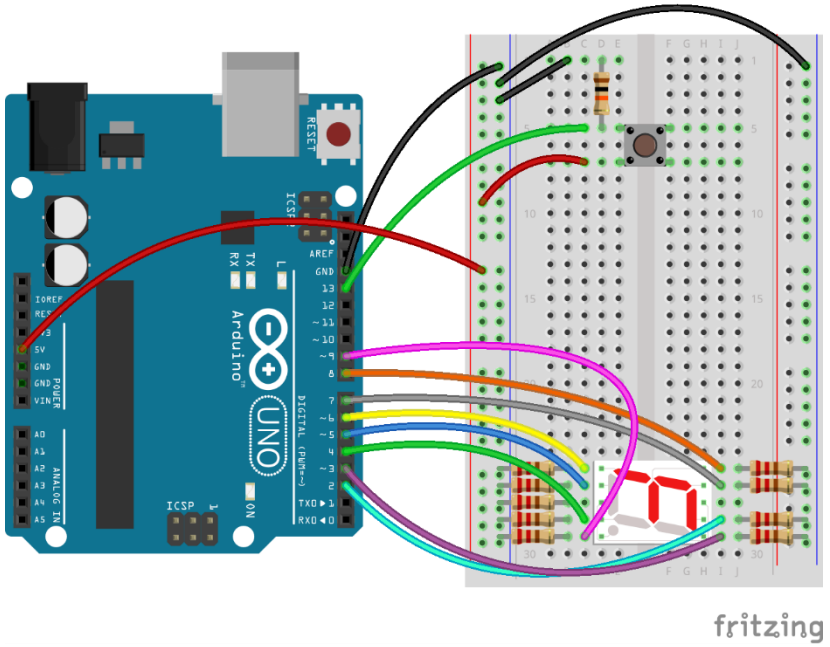
```
int pin = 2;
int num = 0;
void setup() {
    for (int i = 2; i <= 9; i++) {
        pinMode(i, OUTPUT);
    }
    digitalWrite(9, HIGH);
}
void loop() {
    for(int i = 0; i < 8; i++) {
        digitalWrite(pin+i, numbers[num][i]);
    }
    num++;
    if (num > 9) num = 0;
    delay(1000);
}
```

1. 7-세그먼트 표시장치

- 푸시버튼을 이용한 카운터의 구현

[필요한 HW장치 및 부품]

- 아두이노 우노 보드 1개
- 1자리 7-세그먼트 1개
- 푸시버튼 1개
- 220Ω 저항 9개, 10kΩ 저항 1개
- 브레드보드 1개
- 약간의 점퍼선



1. 7-세그먼트 표시장치

- 푸시버튼을 이용한 카운터의 구현

```
int num = 0;   // 7-세그먼트에 표시되는 숫자를 0으로 지정
byte numbers[10][8]= {
   {1, 1, 1, 1, 1, 1, 0, 0}, // 0
   {0, 1, 1, 0, 0, 0, 0, 0}, // 1
   {1, 1, 0, 1, 1, 0, 1, 0}, // 2
   {1, 1, 1, 1, 0, 0, 1, 0}, // 3
   {0, 1, 1, 0, 0, 1, 1, 0}, // 4
   {1, 0, 1, 1, 0, 1, 1, 0}, // 5
   {1, 0, 1, 1, 1, 1, 1, 0}, // 6
   {1, 1, 1, 0, 0, 1, 0, 0}, // 7
   {1, 1, 1, 1, 1, 1, 1, 0}, // 8
   {1, 1, 1, 1, 0, 1, 1, 0} // 9
};
void displayNumbers(int n){
// 7-세그먼트의 LED 요소들을 D2번 핀부터 차례대로 연결
   int pin = 2; // a --> 2, b --> 3, ... , dp --> 8
   for(int i = 0; i < 8; i++) {
      digitalWrite(pin+i, numbers[n][i]);
   }
}
```

```
void setup() {
   pinMode(12, INPUT);   // 푸시버튼
   for (int i = 2; i <= 9; i++) {
      pinMode(i, OUTPUT);
   }
   digitalWrite(9, HIGH);
}
void loop() {
   if (digitalRead(12) == HIGH){
      ++num;
      if (num > 9) {
         num = 0;
      }
   }
   displayNumbers(num);
   delay(300);
}
```

1. 7-세그먼트 표시장치

- 푸시버튼을 이용한 카운터의 구현

만약, 푸시버튼을 누를 때마다 7-세그먼트 표시장치에 1에서 6까지의 숫자 중에서 임의의 한 숫자를 출력하는 주사위를 구현하려면 loop() 함수를 다음과 같이 수정한다.

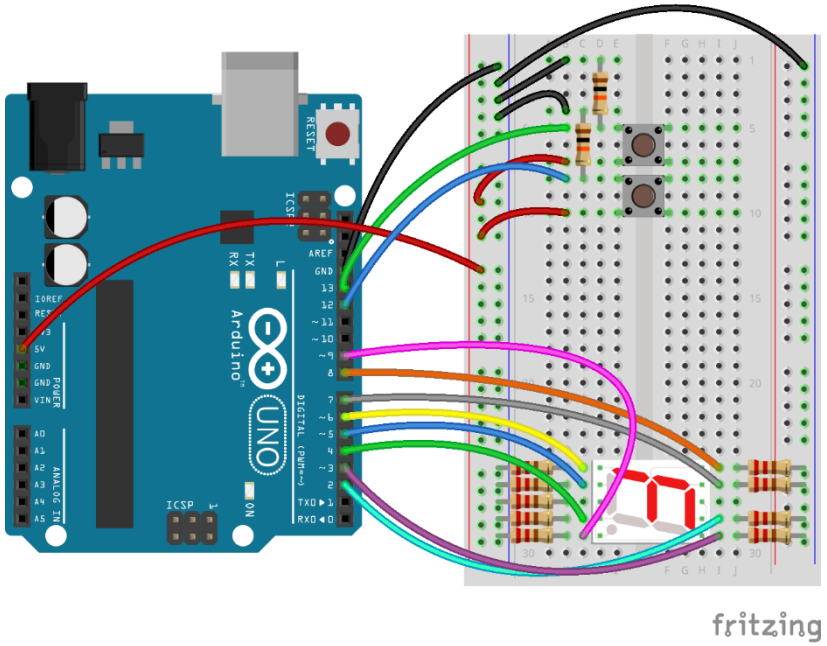
```
void loop() {
  if (digitalRead(12) == HIGH){
       num = random(6);
       num = num + 1;
  }
  displayNumber(num);
  delay(300);
}
```

1. 7-세그먼트 표시장치

- 2개의 푸시버튼으로 숫자 증가/감소

[필요한 HW장치 및 부품]

- 아두이노 우노 보드 1개
- 1자리 7-세그먼트 1개
- 푸시버튼 2개
- 220Ω 저항 9개, 10kΩ 저항 2개
- 브레드보드 1개
- 약간의 점퍼선



- 문법 : bitRead(x, n)
- 매개변수
  x: 읽을 숫자
  n: 읽을 비트의 위치, LSB(맨 오른쪽 비트)가 0, 왼쪽으로 갈수록 1씩 증가

1. 7-세그먼트 표시장치

- 2개의 푸시버튼으로 숫자 증가/감소

```
int num = 0;
byte numbers[10]= {0xFC, 0x60, 0xDA, 0xF2, 0x66, 0xB6, 0xBE, 0xE4, 0xFE,
0xF6};
void displayNumber(int n){
  int pin = 2;
  for(int i = 0; i < 8; i++) {
    boolean state = bitRead(numbers[n], 7-i);
    digitalWrite(pin+i, state);
  }
}
void setup() {
  pinMode(12, INPUT);   // 숫자 증가를 위한 푸시버튼 연결
  pinMode(13, INPUT);   // 숫자 감소를 위한 푸시버튼 연결
  for (int i = 2; i <= 9; i++) {
    pinMode(i, OUTPUT);
  }
  digitalWrite(9, HIGH);
}
```

1. 7-세그먼트 표시장치

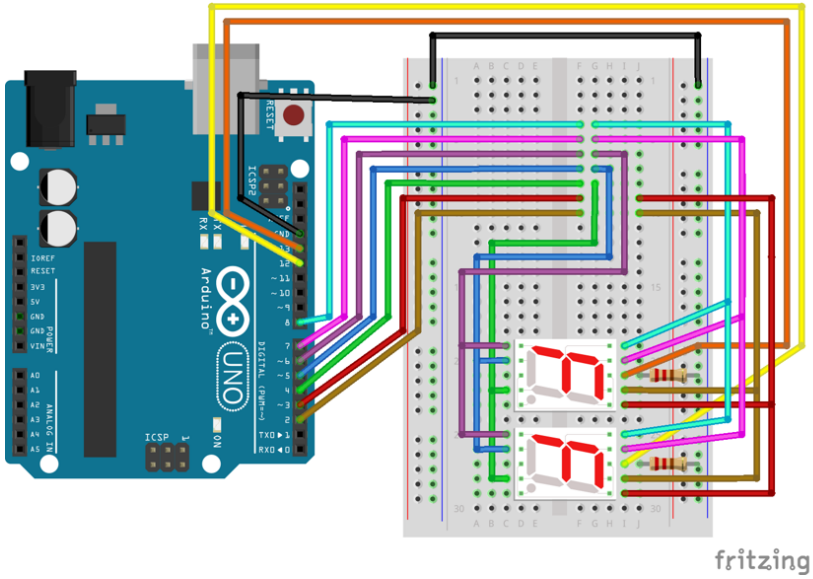
- 2개의 푸시버튼으로 숫자 증가/감소

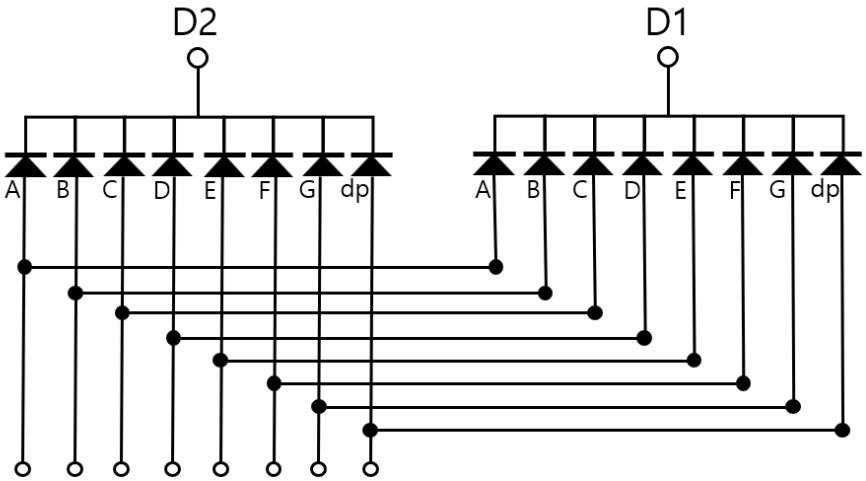
```
void loop() {
  if (digitalRead(12) == HIGH){
    ++num;
    if (num > 9) {
      num = 0;
    }
  }
  if (digitalRead(13) == HIGH){
    --num;
    if (num < 0) {
      num = 9;
    }
  }
  displayNumber(num);
  delay(300);
}
```

2. 1자리 7-세그먼트 2개 연결

[필요한 HW장치 및 부품]

- 아두이노 우노 보드 1개
- 1자리 7-세그먼트 2개
- 330Ω 저항 2개
- 브레드보드 1개
- 약간의 점퍼선





2. 1자리 7-세그먼트 2개 연결

- 0에서 99까지의 수를 표시하는 프로그램

```
byte Numbers[10]= {0b11111100, 0b01100000, 0b11011010, 0b11110010, 0b01100110,
                   0b10110110, 0b10111110, 0b11100100, 0b11111110, 0b11110110};
int segPins[7] = {2,3,4,5,6,7,8};
int digNum = 2;
int digPin[] = {12, 13};
void displayDigit(int x, int d) {
      int value = Numbers[x];
      boolean segSet = LOW;
      digitalWrite(digPin[0], LOW);
      digitalWrite(digPin[1], LOW);
      for (int led=0; led<7; led++) {
          if (value & 0x02)  segSet = HIGH;
          else                segSet = LOW;
          digitalWrite(10-segPins[led], segSet);
          value >>= 1;
      }
      digitalWrite(digPin[d], HIGH);
}
```

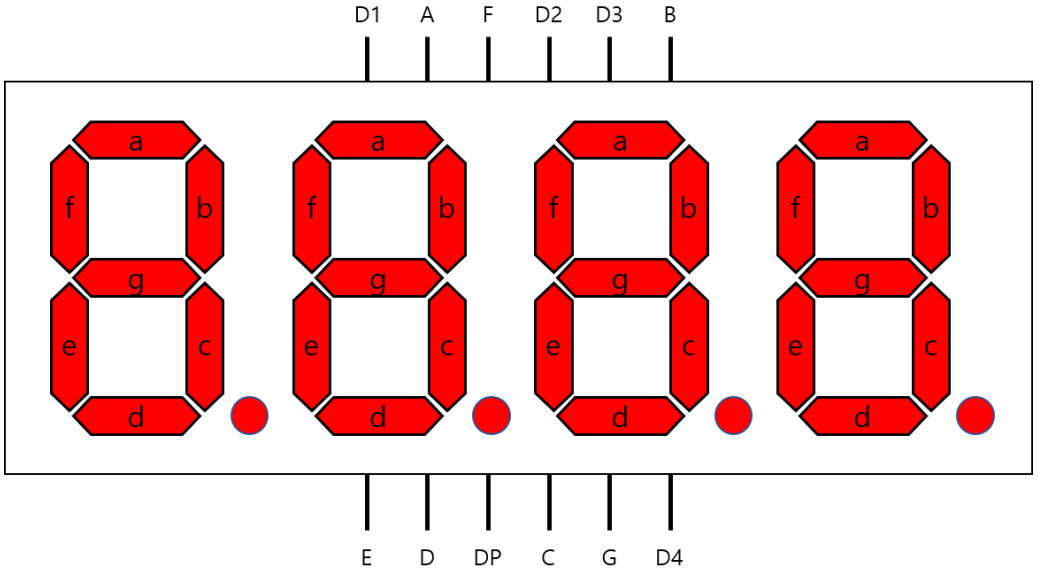
2. 1자리 7-세그먼트 2개 연결

```
void displayNumber(int n) {
    if (n==0) displayDigit(0, 0);
    else {
        for (int i=0; i<digNum; i++) {
        if (n > 0) {
            displayDigit(n % 10, i);
            n = n / 10;
        }
            delay(10);
        }
    }
}
```

```
void setup() {
    for(int i=0;i<8; i++)
        pinMode(segPins[i], OUTPUT);
    for(int j=0; j<digNum; j++)
        pinMode(digPin[j], OUTPUT);
}

void loop() {
    for(int i=0; i<1000; i++) {
        displayNumber(i/10);
    }
}
```

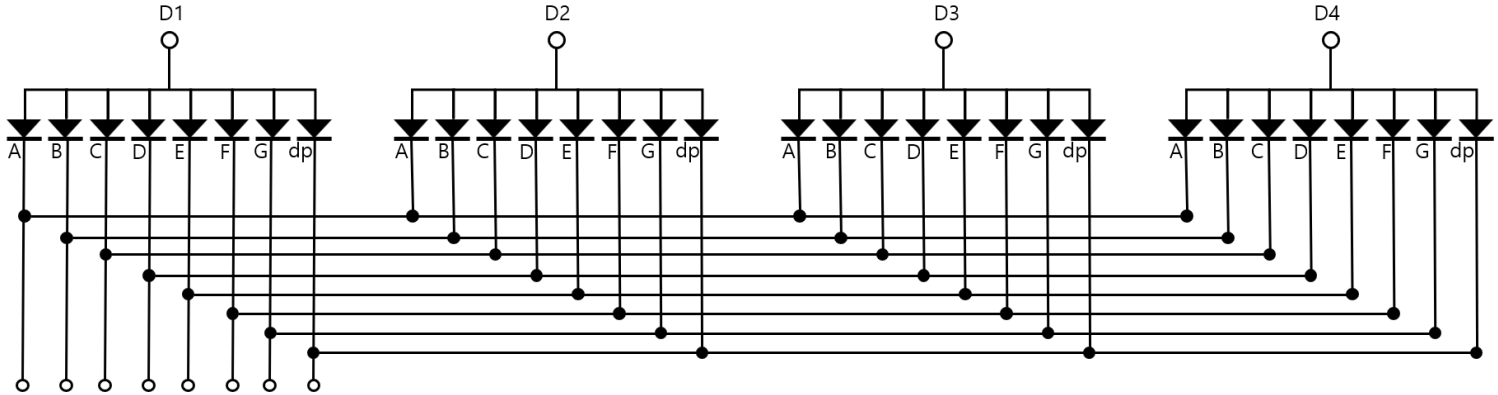
3. 4자리 7-세그먼트 표시장치



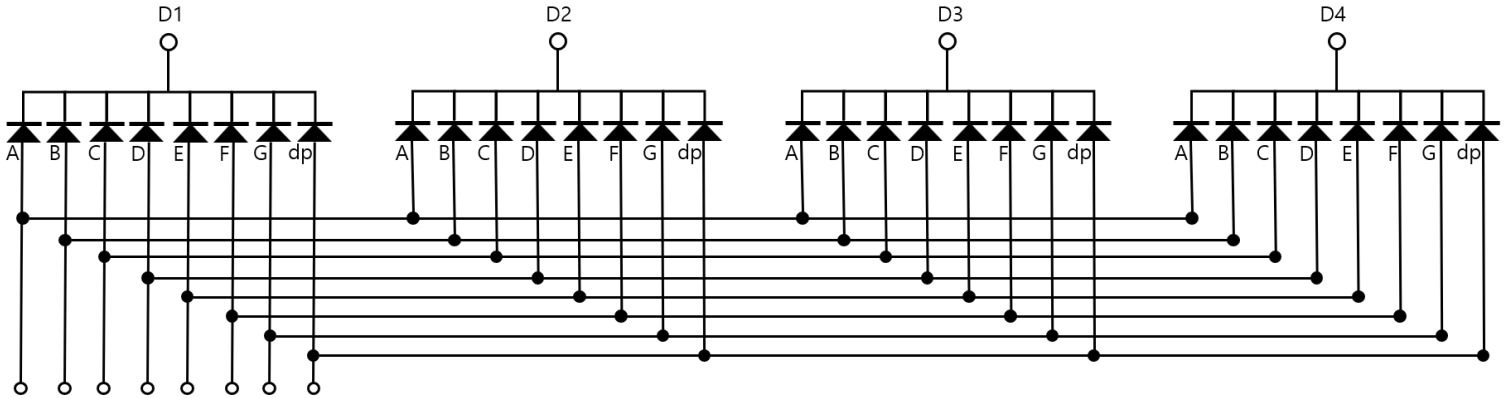
- 12개의 핀
- 8개의 핀은 a에서 g 및 dp(소수점)를 포함하여 7 개의 각 세그먼트 표시장치에 있는 8개의 LED에 사용
- 4개의 핀은 4자리 숫자(D1-D4)를 가리킴

3. 4자리 7-세그먼트 표시장치

- 공통 양극 방식의 4자리 7-세그먼트



- 공통 음극 방식의 4자리 7-세그먼트



3. 4자리 7-세그먼트 표시장치

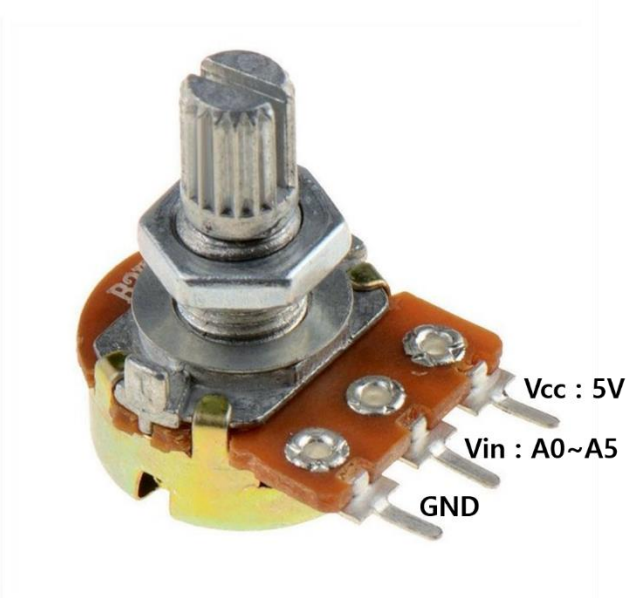
- 멀티플렉싱 기술(Multiplexing Technique)
  - 4자리 7-세그먼트 표시장치는 한 순간에는 한 자리 숫자만 표시하고 다음 자리의 표시를 위하여 매우 빠르게 전환
  - 예 : 시력의 지속성을 이용하여 “1234” 표시
    - “1”과 관련된 세그먼트를 켜고 첫 번째 디스플레이 장치를 켠다.
    - 신호를 보내어 "2"를 표시하고 첫 번째 디스플레이 장치를 끈 다음 두 번째 디스플레이 장치를 켠다.
    - 다음 두 숫자도 이 과정을 반복하고 디스플레이 장치 간 전환을 매우 빠르게 (약 1초 내에) 수행한다.
    - 사람의 눈은 1초 안에 어떤 물체에 반복적으로 발생하는 변화를 구분할 수 없기 때문에 디스플레이에 1234가 동시에 나타나는 것처럼 보인다.

3. 4자리 7-세그먼트 표시장치

- 4자리 7-세그먼트로 가변저항 값 출력

[필요한 HW장치 및 부품]

- 아두이노 우노 보드 1개
- 1자리 7-세그먼트 2개
- 330Ω 저항 2개
- 브레드보드 1개
- 약간의 점퍼선



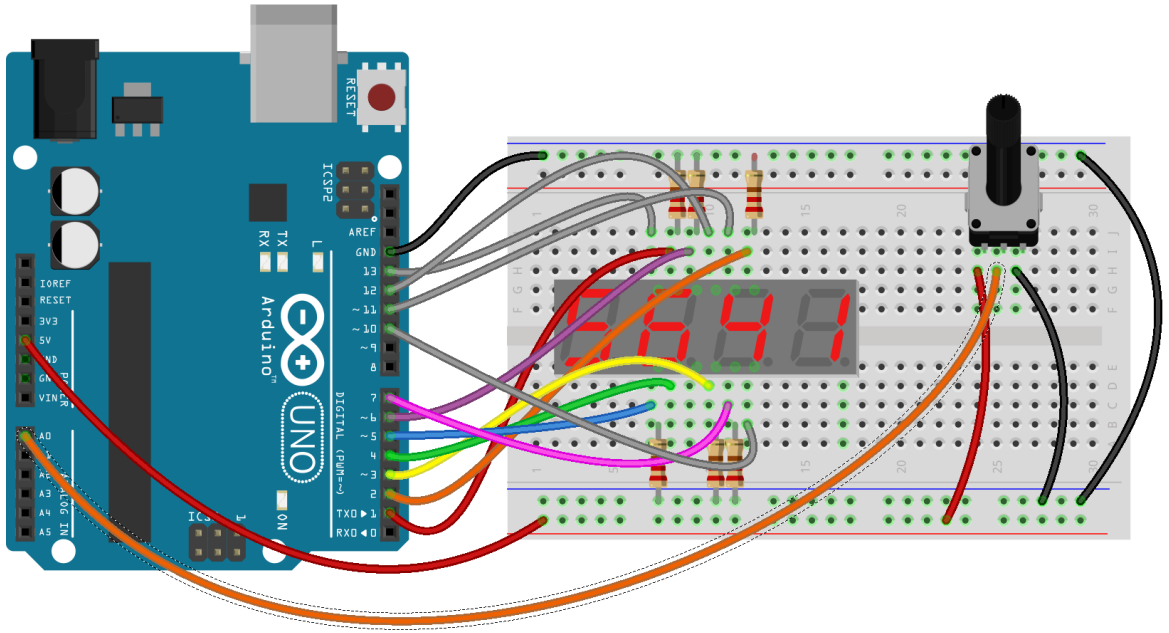
가변저항기

3. 4자리 7-세그먼트 표시장치

- 4자리 7-세그먼트로 가변저항 값 출력

세그먼트와 아두이노 보드의 회로 연결

7-세그먼트	A	B	C	D	E	F	G	dp	D4	D3	D2	D1
아두이노 핀	2	3	4	5	6	7	8	9	10	11	12	13



3. 4자리 7-세그먼트 표시장치

```
int segPin[8] = {2, 3, 4, 5, 6, 7, 8,9};
int digPin[4] = {13, 12, 11, 10};
int delayTime = 3;
byte Numbers[10] = {0b11111100, 0b01100000, 0b11011010, 0b11110010, 0b01100110,
                    0b10110110, 0b10111110, 0b11100100, 0b11111110, 0b11110110};
void displayNumber(int pos, int n) {
  // 출력할 자리 찾기
    for (int i=0; i<4; i++) {
        if (digPin[i] == pos) {
            digitalWrite(digPin[i], LOW);
        }
        else {
            digitalWrite(digPin[i], HIGH);
        }
    }
  // 8개 세그먼트를 제어해서 원하는 숫자 출력
    for(int i = 0; i < 7; i++){
        byte segment = (Numbers[n] & (0x01 << i)) >> i;
        if(segment == 1){
            digitalWrite(segPin[7 - i], HIGH);
        } else {
            digitalWrite(segPin[7 - i], LOW);
        }
    }
    delay(delayTime);
}
```

3. 4자리 7-세그먼트 표시장치

```
void setup() {
      Serial.begin(9600);
      for(int i=0; i<4; i++) {
          pinMode(digPin[i], OUTPUT);
      }
      for(int i=0; i<8; i++) {
          pinMode(segPin[i], OUTPUT);
      }
}
void clearLEDs() {
     for(int i=0; i<4; i++) {
          digitalWrite(digPin[i], LOW);
     }
     for(int i=0; i<8; i++) {
          digitalWrite(segPin[i], LOW);
     }
}
```

```
int n =0;
int readValue;

void loop() {
     n++;
     if (n % 50 == 1) {
          readValue = analogRead(A0);
          Serial.println(readValue);
     }
     int c[4] = {0, 0, 0, 0};
     c[0] = readValue/1000;      // 1000의 자리 숫자
     c[1]=(readValue/100)%10; // 100의 자리 숫자
     c[2]=(readValue/10)%10;   // 10의 자리 숫자
     c[3]=readValue%10;          // 1의 자리 숫자
     for (int i=0; i<4; i++) displayNumber(digPin[i],c[i]);
     clearLEDs();
     delay(delayTime);
}
```