7세그먼트 사용하기

7세그먼트 특징

카운터 만들기

버튼과 연결하기

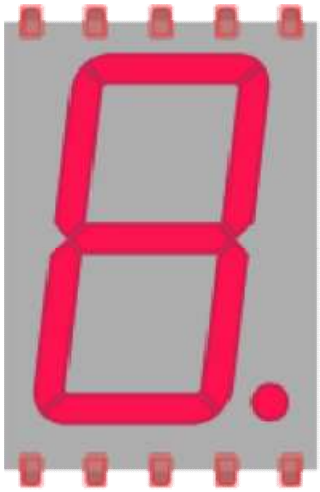
도전해보기

한국과학창의재단

1 7세그먼트 특징

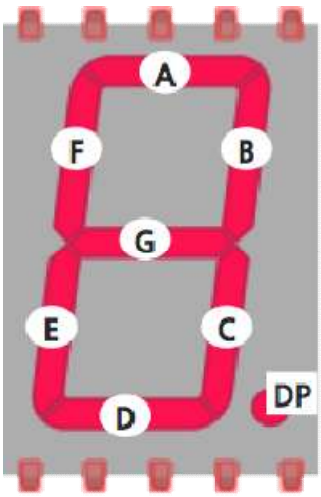
7세그먼트 사용하기

7세그먼트 특징



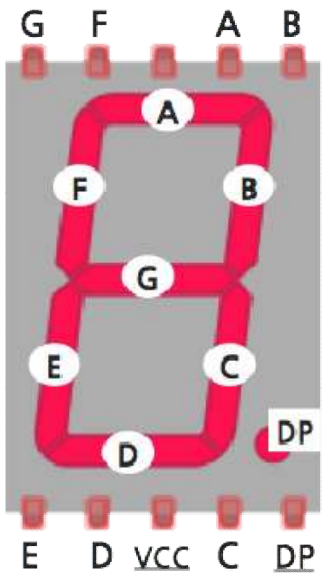
<그림 1> 7세그먼트

7세그먼트란 분할된 7개의 LED를 이용해 숫자 또는 글자를 표시하는 부품입니다.



분할된 영역은 각각 위와 같이 이름을 가지고 있습니다.

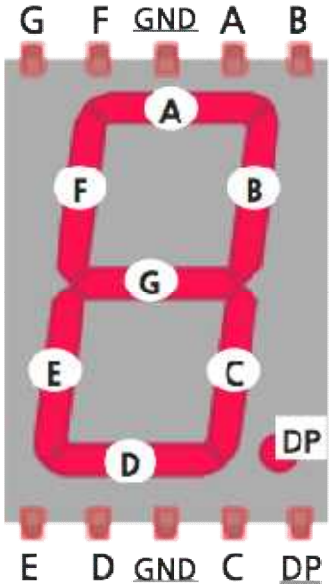
1 7세그먼트 특징



<그림 2> 아노드 방식

7세그먼트는 아노드, 캐소드 방식 두 종류로 구분됩니다. 아노드 방식은 <그림 2>와 같이 되어 있습니다. 아노드 방식은 다음과 같이 사용합니다. 먼저 VCC에 전원을 연결합니다. 그런 다음 A~G, DP핀에 전압을 0V로 설정하면 LED가 켜지고, 5V로 설정하면 꺼집니다. 우리가 사용할 것도 아노드 방식입니다.

1 7세그먼트 특징



<그림 3> 캐소드 방식

캐소드 방식은 VCC가 아닌 GND를 연결합니다. 그 다음 전압을 5V로 하면 켜지고, 0V로 하면 꺼집니다.

2 카운터 만들기

카운터 만들기

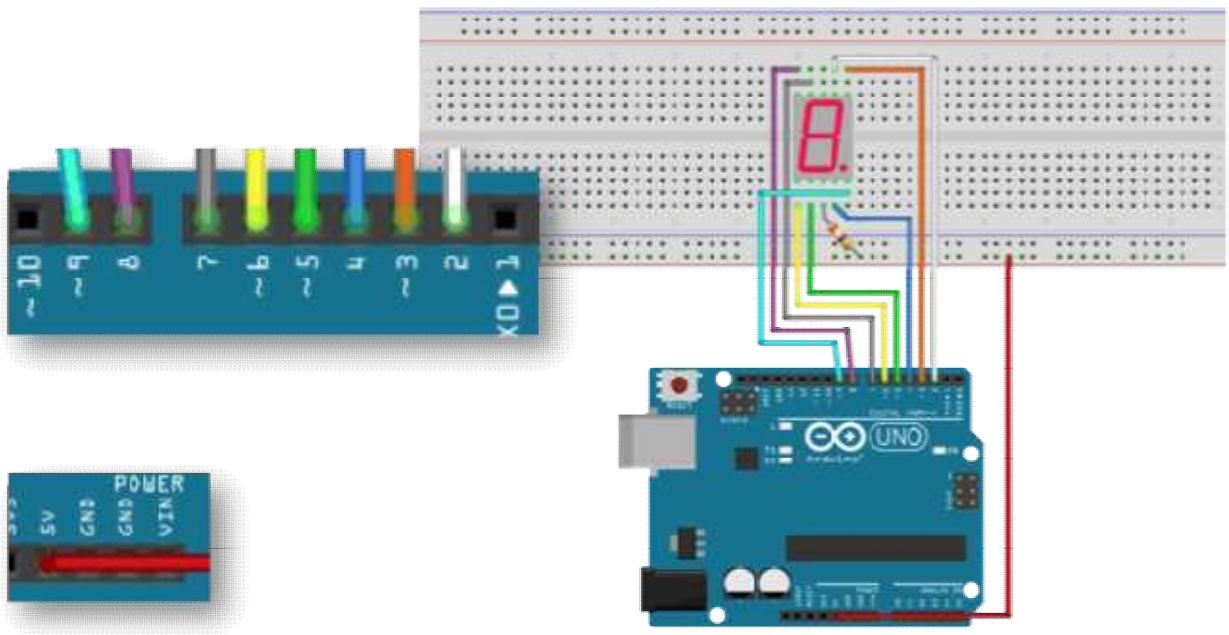
레시피

7세그먼트에 자동으로 숫자가 표시되도록 해봅시다.

<그림 4> 재료

재료는 7세그먼트 1개, 390 ohm 저항 1개입니다.

2 카운터 만들기



<그림 5> 회로도

회로는 <그림 5>와 같이 연결합니다. 각각 회로도에 표시된대로 연결해주시되 VCC 사이에 390 ohm 저항을 연결합니다.

2 카운터 만들기

| 아두이노 코드 작성하기

<코드 1> 카운터 만들기

```
// 7세그먼트에 표시할 숫자를 배열로 준비해놓습니다.
// 0은 끄는 것이고, 1은 켜는 것입니다.
byte digits[10][7] =
{
  { 0,0,0,0,0,0,1 }, // 0
  { 1,0,0,1,1,1,1 }, // 1
  { 0,0,1,0,0,1,0 }, // 2
  { 0,0,0,0,1,1,0 }, // 3
  { 1,0,0,1,1,0,0 }, // 4
  { 0,1,0,0,1,0,0 }, // 5
  { 0,1,0,0,0,0,0 }, // 6
  { 0,0,0,1,1,1,1 }, // 7
  { 0,0,0,0,0,0,0 }, // 8
  { 0,0,0,1,1,0,0 }  // 9
};

void setup(){
  // 2~9번 핀들을 모두 출력 모드로 설정합니다.
  for(int i=2;i<10;i++){
    pinMode(i, OUTPUT);
  }
  // DP, 바로 점에 해당하는 부분을 켜줍니다.
  digitalWrite(9, HIGH);
}
```

2 카운터 만들기

```
void loop(){
  // 10번 반복하는 반복문을 만듭니다.
  // 0에서 9까지 반복합니다.
  for(int i=0;i<10;i++){
    // 1초 멈춥니다.
    delay(1000);
    // 7세그먼트에 숫자를 표시합니다.
    displayDigit(i);
  }
}

// 숫자를 표시하기 위해 만든 함수입니다.
void displayDigit(int num){
  // 핀이 2번부터 시작하기 때문에
  // 핀 번호를 맞춰주기 위해 2라는 값을 pin이란 변수를 선언했습니다.
  int pin = 2;
  for(int i=0;i<7;i++){
    // 앞서 준비한 배열에서 값을 불러와서
    // 숫자를 7세그먼트에 표시합니다.
    digitalWrite(pin+i, digits[num][i]);
  }
}
```

2 카운터 만들기

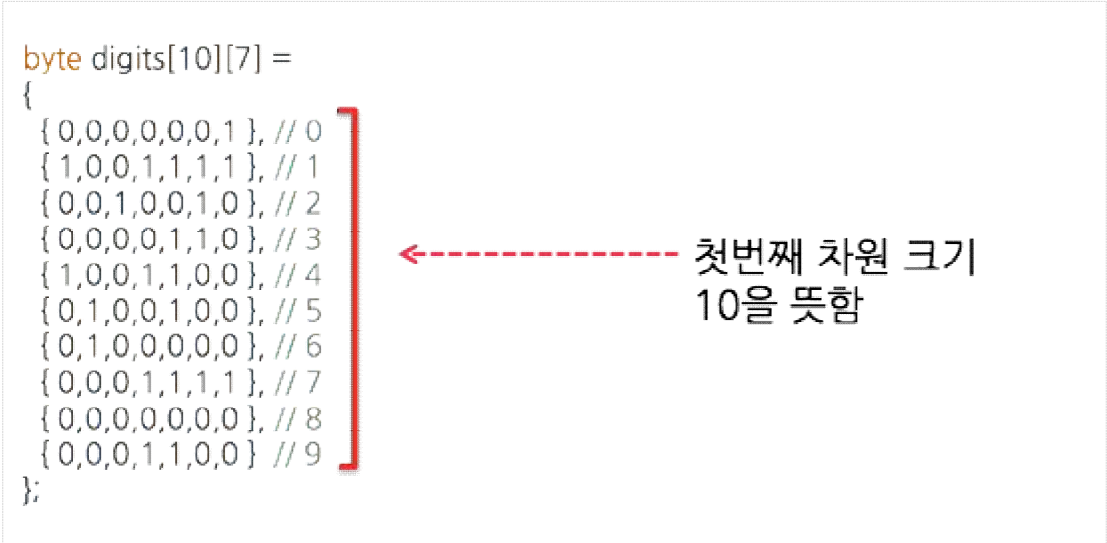
▪ 2차원 배열

```
byte digits[10][7] = …
```

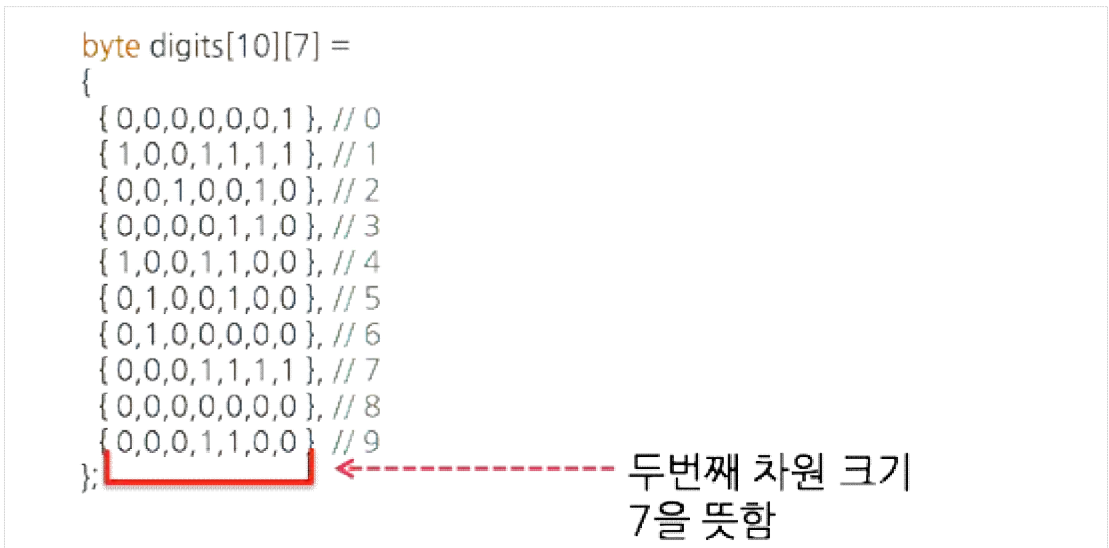
byte는 0~255의 값을 넣을 수 있는 변수형입니다. 여기서는 byte형 2차원 배열을 선언한 것입니다.

2차원 배열을 선언할때는 첫번째 차원 크기와 두번째 차원 크기를 순서대로 넣어줍니다.

2 카운터 만들기



첫번째 차원 크기로 10을 넣은 것은, 위와 같이 가장 바깥쪽 배열의 크기가 10이기 때문입니다.



두번째 차원크기 7은 안쪽 배열의 크기를 말합니다.

2 카운터 만들기

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

digits[4]

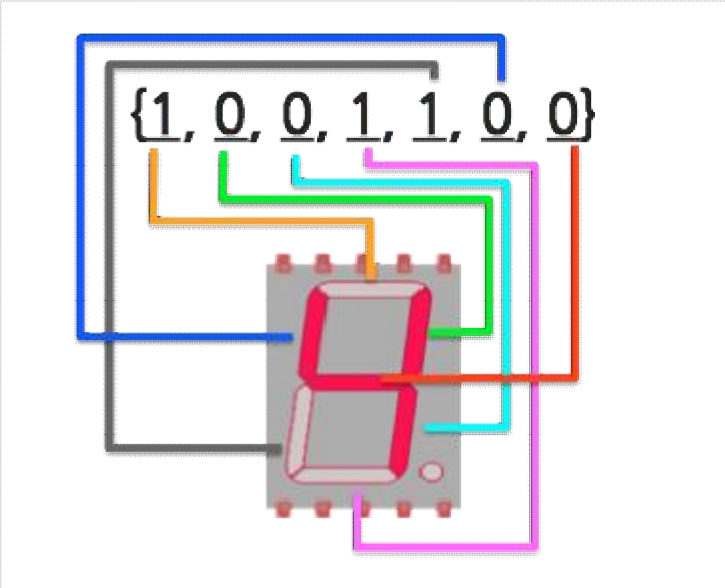
만약 digits[4]라고 적으면 5번째 있는 작은 배열을 가지고 오는 것입니다.

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

digits[4][2]

이번에는 digits[4][2]라고 적으면 5번째 작은 배열에서 3번째 값을 가지고 오겠다는 뜻입니다.

2 카운터 만들기



앞서 다섯번째 작은 배열은 digits[4]는 위와 같이 표시하란 뜻입니다. 0과 1이 각각 해당 핀의 LED를 켜고 끄라는 뜻이기 때문에 그걸 통해 4라는 숫자가 표시됩니다.

2 카운터 만들기

▪ **함수**

```
void setup(){
  for(int i=2;i<10;i++){
    pinMode(i, OUTPUT);
  }
  digitalWrite(9, HIGH);
}

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

displayDigit 함수 호출

<코드 1>에 보면 displayDigit이라는 함수를 사용합니다. 위에서와 같이 displayDigit(i);가 호출되면 프로그램은 함수 부분으로 이동해서 함수의 내용을 실행합니다.

```
void displayDigit(int num){
  int pin = 2;
  for(int i=0;i<7;i++){
    digitalWrite(pin+i, digits[num][i]);
  }
}
```

displayDigit(i);
여기서 i값이 num값으로
들어간다.

앞서 i에 들어갔던 값이 num이라는 값에 들어가게 되고 함수 안의 코드가 실행됩니다.

확인하기

1초마다 0에서 9까지 차례대로 7세그먼트에 표시됩니다.

3 버튼과 연결하기

버튼과 연결하기

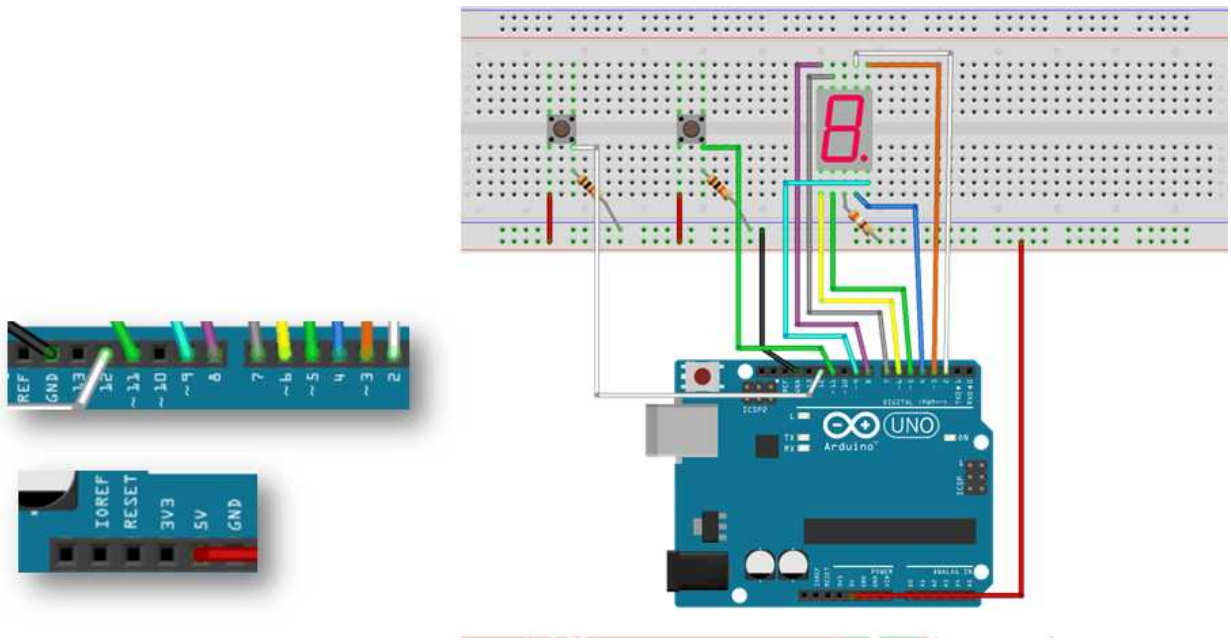
레시피

버튼을 눌러서 7세그먼트의 숫자를 증가시키거나 감소시킬 수 있도록 해봅시다.

<그림 6> 재료

재료는 7세그먼트 1개, 버튼 2개, 390 ohm 저항 1개, 10k ohm 저항 2개입니다.

3 버튼과 연결하기



<그림 7> 회로도

회로는 <그림 7>과 같이 연결합니다. 11, 12에 각각 버튼을 연결합니다. 7세그먼트는 앞에서와 동일하게 연결하면 됩니다.

3 버튼과 연결하기

| 아두이노 코드 작성하기

<코드 2> 버튼과 연결하기

```
// 버튼 연결 핀 번호를 매크로 상수로 지정합니다.
#define PLUS 11
#define MINUS 12

// 현재 숫자를 기록하는 변수를 선언합니다.
int digit = 0;

// 7세그먼트에 표시할 숫자를 배열로 준비해놓습니다.
// 0은 끄는 것이고, 1은 켜는 것입니다.
byte digits[10][7] =
{
  { 0,0,0,0,0,0,1 }, // 0
  { 1,0,0,1,1,1,1 }, // 1
  { 0,0,1,0,0,1,0 }, // 2
  { 0,0,0,0,1,1,0 }, // 3
  { 1,0,0,1,1,0,0 }, // 4
  { 0,1,0,0,1,0,0 }, // 5
  { 0,1,0,0,0,0,0 }, // 6
  { 0,0,0,1,1,1,1 }, // 7
  { 0,0,0,0,0,0,0 }, // 8
  { 0,0,0,1,1,0,0 }  // 9
};

void setup(){
  // 버튼 핀들을 입력 모드로 설정합니다.
  pinMode(PLUS, INPUT);
  pinMode(MINUS, INPUT);
  // 2~9번 핀들을 모두 출력 모드로 설정합니다.
  for(int i=2;i<10;i++){
    pinMode(i, OUTPUT);
  }
  // DP, 바로 점에 해당하는 부분을 켜줍니다.
  digitalWrite(9, HIGH);
}

void loop(){
  // 증가 버튼이 눌렸는지 확인합니다.
  if(digitalRead(PLUS) == HIGH){
    //눌렸다면 digit을 증가시킵니다.
    ++digit;
```

3 버튼과 연결하기

```
    // digit이 9를 넘었는지 확인하고 넘었으면 0으로 만듭니다.
    if(digit>9){
      digit=0;
    }
  }

  // 감소 버튼이 눌렸는지 확인합니다.
  if(digitalRead(MINUS) == HIGH){
    // 눌렸다면 digit을 감소시킵니다.
    --digit;

    // digit이 0 밑으로 내려갔는지 확인하고 내려갔으면 9로 만듭니다.
    if(digit<0){
      digit=9;
    }
  }

  // digit을 displayDigit 함수를 이용해 7세그먼트에 표시합니다.
  displayDigit(digit);
  // 0.1초 멈춥니다.
  delay(100);
}

// 숫자를 표시하기 위해 만든 함수입니다.
void displayDigit(int num){
  // 핀이 2번부터 시작하기 때문에
  // 핀 번호를 맞춰주기 위해 2라는 값을 pin이란 변수를 선언했습니다.
  int pin = 2;
  for(int i=0;i<7;i++){
    // 앞서 준비한 배열에서 값을 불러와서
    // 숫자를 7세그먼트에 표시합니다.
    digitalWrite(pin+i, digits[num][i]);
  }
}
```

확인하기

두 버튼을 이용해 7세그먼트에 표시된 숫자를 증가시키거나 감소시킬 수 있습니다.

4 도전해보기

도전해보기

7세그먼트 2개를 연결해봅시다.

숫자가 아닌 알파벳을 연결해봅시다.