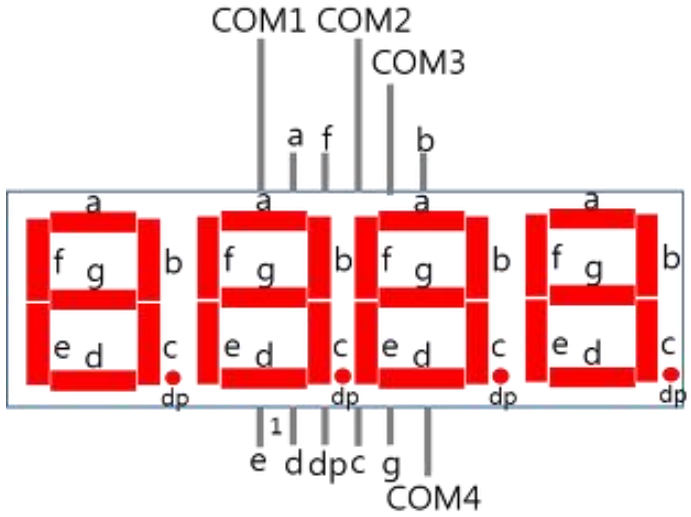
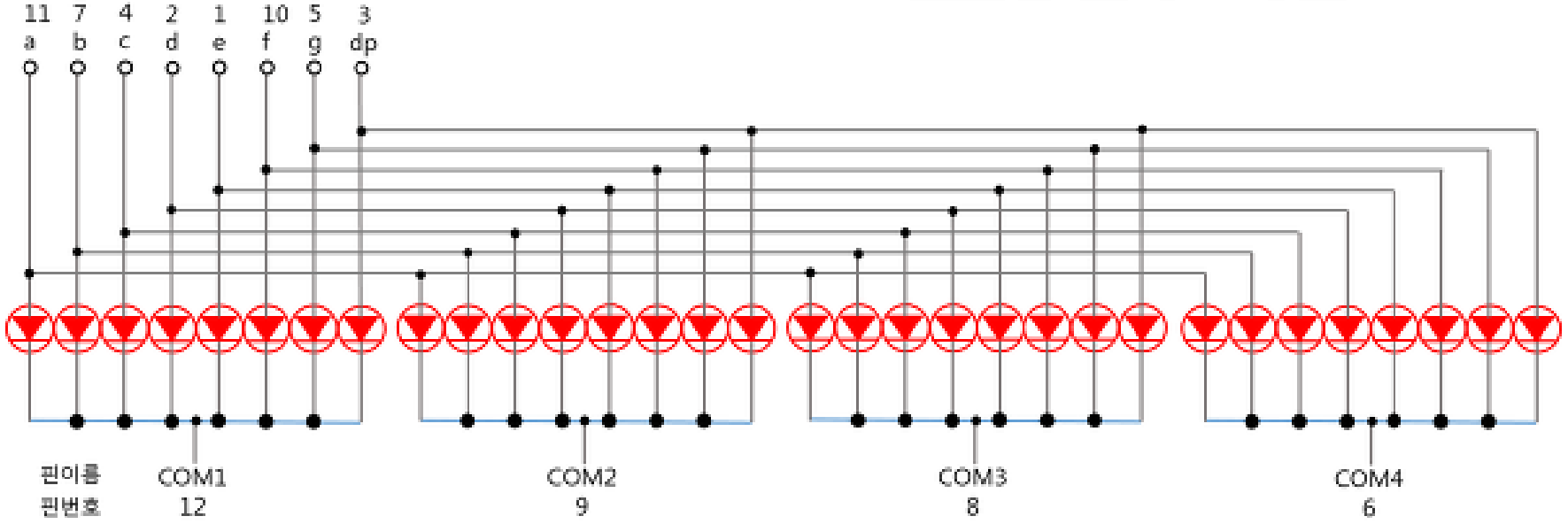
4자리 7-세그먼트

4자리 7-세그먼트

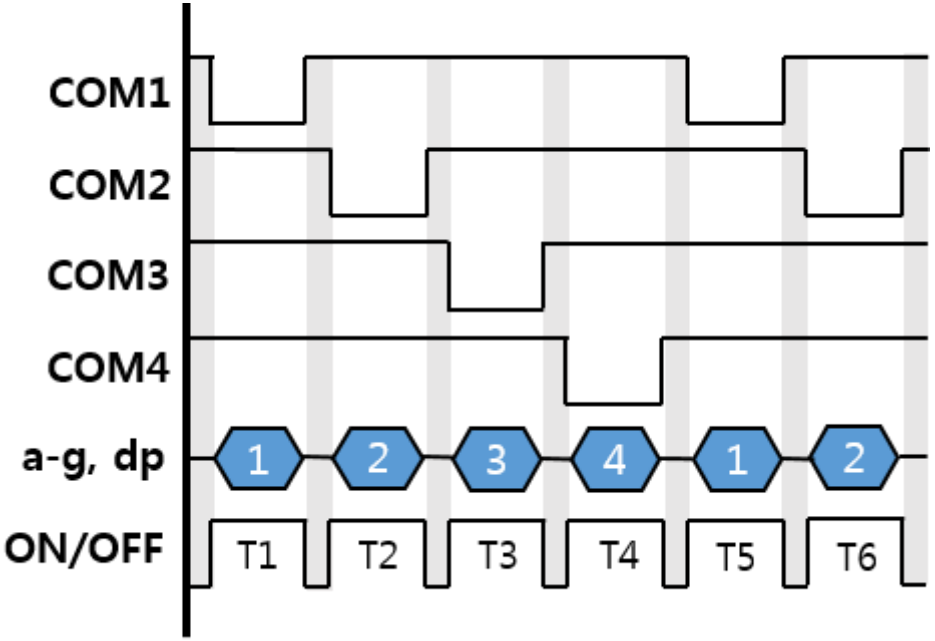
❖ 4자리 7-세그먼트

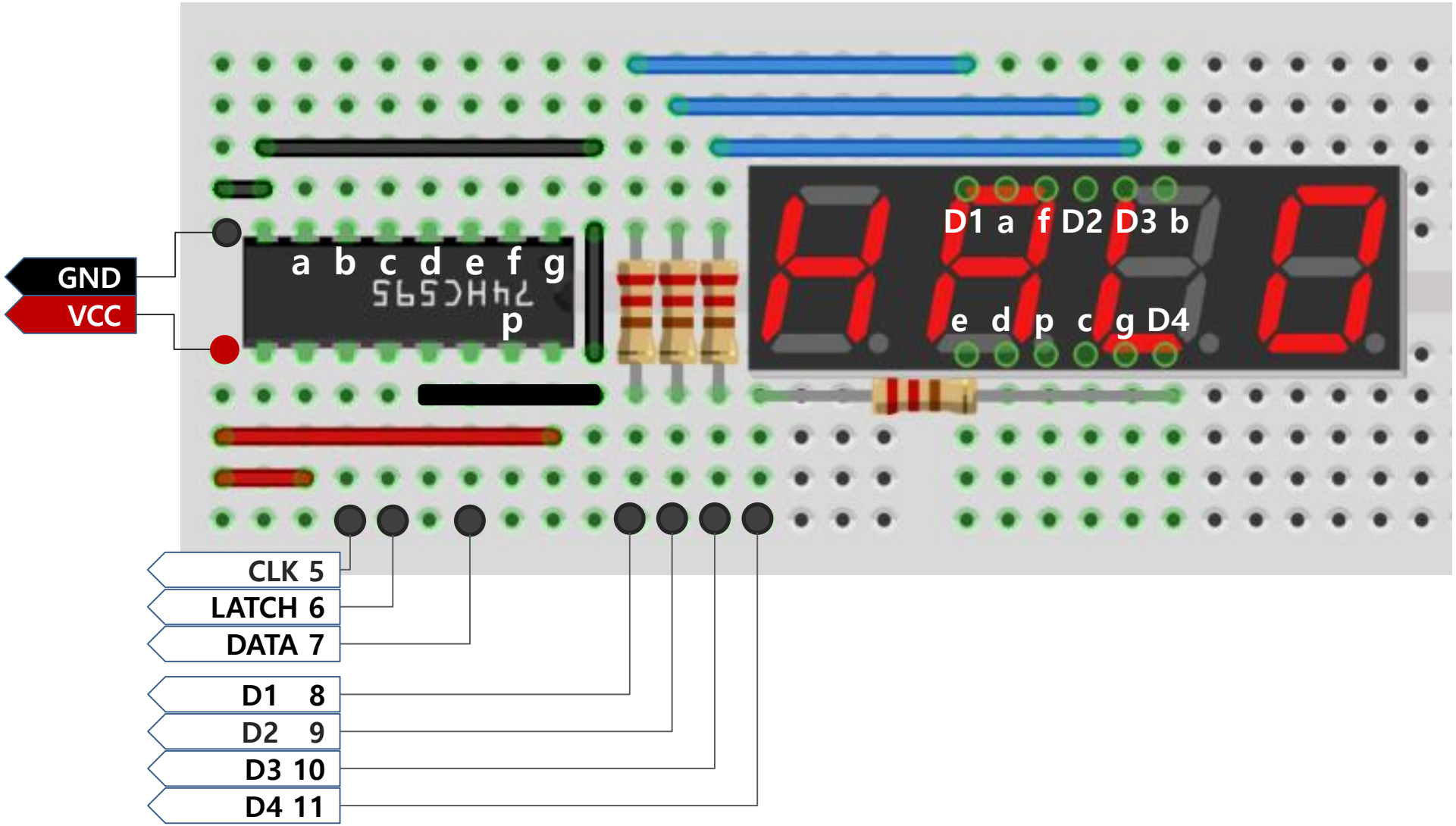




4자리 7-세그먼트

❖ 동적 디스플레이 제어를 위한 각 핀들의 타이밍 차트




GND
VCC
a b c d e f g
p
D1 a f D2 D3 b
e d p c g D4
CLK 5
LATCH 6
DATA 7
D1 8
D2 9
D3 10
D4 11

4자리 7-세그먼트

❖ fnd4_1.ino

```
// fnd4_1
// 4자리 7-세그먼트 '1234' 표시하기
// 0123 표시: 100ms, 10ms, 5ms 간격으로 DIG1(1000자리), DIG2, DIG3, DIG4 표시
const int dig_sel_pin[4] = {A4, 13, 12, 11};     // 1, 10, 100, 1000자리 전원핀
const int fnd_pin[7] ={10, 9, 8, 7, 6, 5, 4};    // A,B,C,D,E,F,G
const int fnd_pat[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

const int disp_val[4] = {4, 3, 2, 1}; // 표시값 1234
void  fnd_dsp100(void);               // 7-세그먼트 4자리 표시(100ms 간격)
void  fnd_dsp40(void);                // 7-세그먼트 4자리 표시(40ms 간격)
void  fnd_dsp20(void);                // 7-세그먼트 4자리 표시(20ms 간격)
```

4자리 7-세그먼트

❖ fnd4_1.ino

```
void setup()
{
  byte n;

  // fnd_pin[n] 핀 출력 설정
  for(n = 0;n < 7;n++) pinMode(fnd_pin[n], OUTPUT);

  // Dight(자리) 선택핀 출력 설정
  for(n = 0;n < 4;n++) pinMode(dig_sel_pin[n], OUTPUT);
}

void loop()
{
  fnd_dsp100();   // 4개 표시 주기 100ms로 5초 동안 표시하기
  delay(1000);    // 1초 대기

  fnd_dsp40();    // 4개 표시 주기 40ms로 5초 동안 표시하기
  delay(1000);    // 1초 대기

  fnd_dsp20();    // 4개 표시 주기 20ms로 5초 동안 표시하기
  delay(1000);    // 1초 대기
}
```

4자리 7-세그먼트

❖ fnd4_1.ino

```
void fnd_dsp100()
{
  byte    cnt, dig, n, tmp;

  // 5초(50 x 100ms) 동안 표시하기
  for(cnt = 0;cnt < 50;cnt++){
    // 4개 100ms(4 x 25ms) 주기로 표시
    for(dig = 0;dig < 4;dig++){
      // 모든 자리 OFF 한 후 dig(표시위치) 자리만 ON
      for(n = 0;n < 4;n++) digitalWrite(dig_sel_pin[n], LOW);
      digitalWrite(dig_sel_pin[dig], HIGH);    // dig 위치만 ON

      // dig값에 해당하는 패턴 출력
      tmp = disp_val[dig];   // dig 위치에 표시할 값
      for(n = 0;n < 7;n++)
         digitalWrite(fnd_pin[n], fnd_pat[tmp][n]);
      delay(25);             // 자리 출력 간격 25ms
    }
  }
}
```

4자리 7-세그먼트

❖ fnd4_1.ino

```
void fnd_dsp40()
{
  byte    cnt, dig, n, tmp;

  // 5초(125 x 40ms)  동안 표시하기
  for(cnt = 0;cnt < 125;cnt++){
    // 4개 40ms(4 x 10ms) 주기로 표시
    for(dig = 0;dig < 4;dig++){
      // 모든 자리 OFF 한 후 dig(표시위치) 자리만 ON
      for(n = 0;n < 4;n++) digitalWrite(dig_sel_pin[n], LOW);
      digitalWrite(dig_sel_pin[dig], HIGH);    // dig 위치만 ON

      // dig값에 해당하는 패턴 출력
      tmp = disp_val[dig];   // dig 위치에 표시할 값
      for(n = 0;n < 7;n++)
         digitalWrite(fnd_pin[n], fnd_pat[tmp][n]);
      delay(10);             // 자리 출력 간격 10ms
    }
  }
}
```

4자리 7-세그먼트

❖ fnd4_1.ino

```
void fnd_dsp20()
{
  byte    cnt, dig, n, tmp;

   // 5초(250 x 20ms) 동안 표시하기
   for(cnt = 0;cnt < 250;cnt++){
    // 4개 20ms(4 x 5ms) 주기로 표시
    for(dig = 0;dig < 4;dig++){
      // 모든 자리 OFF 한 후 dig(표시위치) 자리만 ON
      for(n = 0;n < 4;n++) digitalWrite(dig_sel_pin[n], LOW);
      digitalWrite(dig_sel_pin[dig], HIGH);    // dig 위치만 ON

      // dig값에 해당하는 패턴 출력
      tmp = disp_val[dig];  // dig 위치에 표시할 값
      for(n = 0;n < 7;n++)
         digitalWrite(fnd_pin[n], fnd_pat[tmp][n]);
      delay(5);              // 자리 출력 간격 5ms
    }
  }
}
```

4자리 7-세그먼트

❖ fnd4_2.ino

```
// 0000~9999표시(DIG1, DIG2, DIG3, DIG4)
const int dig_sel_pin[4] = {A4, 13, 12, 11};    // 1, 10, 100, 1000자리 전원핀
const int fnd_pin[7] ={10, 9, 8, 7, 6, 5, 4};   // A,B,C,D,E,F,G
const int fnd_pat[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

int   dsp_no = 0;          // 표시 값
void  segment_dsp4();      // 4자리 7-세그먼트 표시 함수
```

4자리 7-세그먼트

❖ fnd4_2.ino

```
void setup()
{
  byte  n;

  // fnd_pin[n] 핀 출력 설정
  for(n = 0;n < 7;n++) pinMode(fnd_pin[n], OUTPUT);

  // Dight(자리) 선택핀 출력 설정
  for(n = 0;n < 4;n++) pinMode(dig_sel_pin[n], OUTPUT);
}

void loop()
{
  segment_dsp4();
  dsp_no++;
  if(dsp_no == 10000) dsp_no = 0;
}
```

4자리 7-세그먼트

❖ fnd4_2.ino

```
void segment_dsp4() {
  byte  dig, n, dig_val[4], cnt;
  int   tmp;

  // 1, 10, 100, 100자리 추출
  tmp = dsp_no;
  for(n = 0;n < 4;n++){
    // dig_val[0]: 1자리, dig_val[1]: 10자리,
    // dig_val[2]: 100자리, dig_val[3]: 1000자리,
    dig_val[n] = tmp % 10;
    tmp = tmp / 10;
  }
```

4자리 7-세그먼트

❖ fnd4_2.ino

```
  for(cnt = 0;cnt < 3;cnt++){  // 20ms x 3 = 60m간 표시
    for(dig = 0;dig < 4;dig++){
       // 모든 자리 OFF 한 후 dig(표시위치) 자리만 ON
       for(n = 0;n < 4;n++) digitalWrite(dig_sel_pin[n], LOW);
       digitalWrite(dig_sel_pin[dig], HIGH);     // dig 위치만 ON

       tmp = dig_val[dig];           // dig 위치 표시할 값
       for(n = 0;n < 7;n++) digitalWrite(fnd_pin[n], fnd_pat[tmp][n]);
       delay(5);
    }
  }
}
```

4자리 7-세그먼트

❖ fnd4_3.ino

```
// 폴링 방식에 의한 네 자리 입력 실험

#define  ON    LOW
#define  OFF   HIGH

const int dig_sel_pin[4] = {A4, 13, 12, 11};     // 1, 10, 100, 1000자리 전원핀
const int fnd_pin[7] ={10, 9, 8, 7, 6, 5, 4};    // A,B,C,D,E,F,G
const int fnd_pat[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

const int  dp_pin = A5;     // 7-세그먼트 DP 연결핀(SCL핀)
const int  sw1_pin = 3;     // 선택된 자리값 +1
const int  sw2_pin = 2;     // 자리 선택(1자리 -> 10자리 -> 100자리 -> 1000자리)
```

4자리 7-세그먼트

❖ fnd4_3.ino

```
byte    pos = 0;                  // 0:1자리, 1:10자리, 2:100자리, 3:1000자리
byte    num[4] ={0, 0, 0, 0};     // 4자리 저장 변수
void    sw1_on(void);             // SW1 눌려지는 순간(선택된 자리 값 +1)
void    sw2_on(void);             // SW2 눌려지는 순간(선택 자리 이동)
void    segment_dsp4(void);      // 4자리 7-세그먼트 표시 함수

void setup()
{
  byte   n;

  // fnd_pin[n] 핀 출력 설정
  for(n = 0;n < 7;n++)
     pinMode(fnd_pin[n], OUTPUT);

  pinMode(dp_pin, OUTPUT);    // DP 핀 출력 설정

  // Dight(자리) 선택핀 출력 설정
  for(n = 0;n < 4;n++) pinMode(dig_sel_pin[n], OUTPUT);

  pinMode(sw1_pin, INPUT);    // SW1 연결핀 입력 설정
  pinMode(sw2_pin, INPUT);    // SW2 연결핀 입력 설정
}
```

4자리 7-세그먼트

❖ fnd4_3.ino

```
void loop()
{
  boolean  o_sw1, o_sw2, n_sw1, n_sw2;

  // SW1, SW2 첫 번째 상태 읽기
  o_sw1 = digitalRead(sw1_pin);
  o_sw2 = digitalRead(sw2_pin);

  segment_dsp4();

  // SW1, SW2 두 번째 상태 읽기
  n_sw1 = digitalRead(sw1_pin);
  n_sw2 = digitalRead(sw2_pin);

  //  스위치 눌려지는 순간 체크
  if(o_sw1 == OFF && n_sw1 == ON)  sw1_on();      // SW1 눌려질 때
  else if(o_sw2 == OFF && n_sw2 == ON) sw2_on(); // SW2 눌려질 때
}
```

4자리 7-세그먼트

❖ fnd4_3.ino

```
// 4자리 7-세그먼트 표시
void segment_dsp4()
{
  byte  dig, n, tmp;

  for(dig = 0;dig < 4;dig++){
    // 모든 자리 OFF 한 후 dig(표시위치) 자리만 ON
    for(n = 0;n < 4;n++) digitalWrite(dig_sel_pin[n], LOW);
    digitalWrite(dig_sel_pin[dig], HIGH);  // 표시 세그먼트만 ON

    tmp = num[dig];         // 표시값
    for(n = 0;n < 7;n++) digitalWrite(fnd_pin[n], fnd_pat[tmp][n]);

    // 선택 자리 DP LED ON
    if(pos == dig) digitalWrite(dp_pin, LOW);
    else digitalWrite(dp_pin, HIGH);
    delay(5);
  }
}
```

4자리 7-세그먼트

❖ fnd4_3.ino

```
// SW1 눌려지는 순간(선택된 자리값 +1)
void sw1_on()
{
  num[pos] = (num[pos] + 1) % 10;   // pos 자리값 +1(0~9)
}

// SW2 눌려지는 순간(입력 자리 이동)
void sw2_on()
{
  pos = (pos + 1) % 4;      // 0->1->2->3->0->1...
}
```

4자리 7-세그먼트

❖ fnd4_4.ino

```
// 외부 인터럽트 방식에 의한 네 자리 입력 실험
// 채터링 방지 기능
const int dig_sel_pin[4] = {A4, 13, 12, 11};     // 1, 10, 100, 1000자리 전원핀
const int fnd_pin[7] ={10, 9, 8, 7, 6, 5, 4};    // A,B,C,D,E,F,G
const int fnd_pat[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

const int  dp_pin = A5;    // 7-세그먼트 DP 연결핀(SCL핀)
const int  sw1_pin = 3;    // 선택된 자리값 +1
const int  sw2_pin = 2;    // 자리 선택(1자리 -> 10자리 -> 100자리 -> 1000자리)
```

4자리 7-세그먼트

❖ fnd4_4.ino

```
volatile byte  pos = 0;              // 0:1자리, 1:10자리, 2:100자리, 3:1000자리
volatile byte  num[4] ={0, 0, 0, 0}; // 4자리 저장 변수
volatile unsigned long  t1, t2;

void    segment_dsp4(void);    // 4자리 7-세그먼트 표시 함수

void setup()
{
  byte   n;

  // fnd_pin[n]번핀 출력 설정
  for(n = 0;n < 7;n++) pinMode(fnd_pin[n], OUTPUT);
  pinMode(dp_pin, OUTPUT);    // DP 핀 출력 설정


  // Dight(자리) 선택핀 출력 설정
  for(n = 0;n < 4;n++) pinMode(dig_sel_pin[n], OUTPUT);

  pinMode(sw1_pin, INPUT);    // SW1 연결핀 입력 설정
  pinMode(sw2_pin, INPUT);    // SW2 연결핀 입력 설정
```

4자리 7-세그먼트

❖ fnd4_4.ino

```
  // 외부 인터럽트 설정
  attachInterrupt(digitalPinToInterrupt(sw1_pin), sw1_on, FALLING);  // INT1
  attachInterrupt(digitalPinToInterrupt(sw2_pin), sw2_on, FALLING);  // INT0

  t1 = millis();          // 프로그램 시작 시간 저장
}

void loop()
{
  segment_dsp4();
}
```

4자리 7-세그먼트

❖ fnd4_4.ino

```
// 4자리 7-세그먼트 표시
void segment_dsp4()
{
  byte  dig, n, tmp;

  for(dig = 0;dig < 4;dig++){
    // 모든 자리 off 한 후 dig(표시위치) 자리만 on
    for(n = 0;n < 4;n++) digitalWrite(dig_sel_pin[n], LOW);
    digitalWrite(dig_sel_pin[dig], HIGH);  // 표시 세그먼트만 ON

    tmp = num[dig];         // 표시값
    for(n = 0;n < 7;n++) digitalWrite(fnd_pin[n], fnd_pat[tmp][n]);

    // 선택 자리 dp LED on
    if(pos == dig) digitalWrite(dp_pin, LOW);
    else digitalWrite(dp_pin, HIGH);
    delay(5);
  }
}
```

4자리 7-세그먼트

❖ fnd4_4.ino

```
// SW1 눌려지는 순간(선택된 자리값 +1) : INT1
void sw1_on() {
  // 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

  num[pos] = (num[pos] + 1) % 10;  // pos 자리값 +1(0~9)
}
// SW2 눌려지는 순간(입력 자리 이동) : INT0
void sw2_on() {
  // 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

  pos = (pos + 1) % 4;    // 0->1->2->3->0->1...
}
```