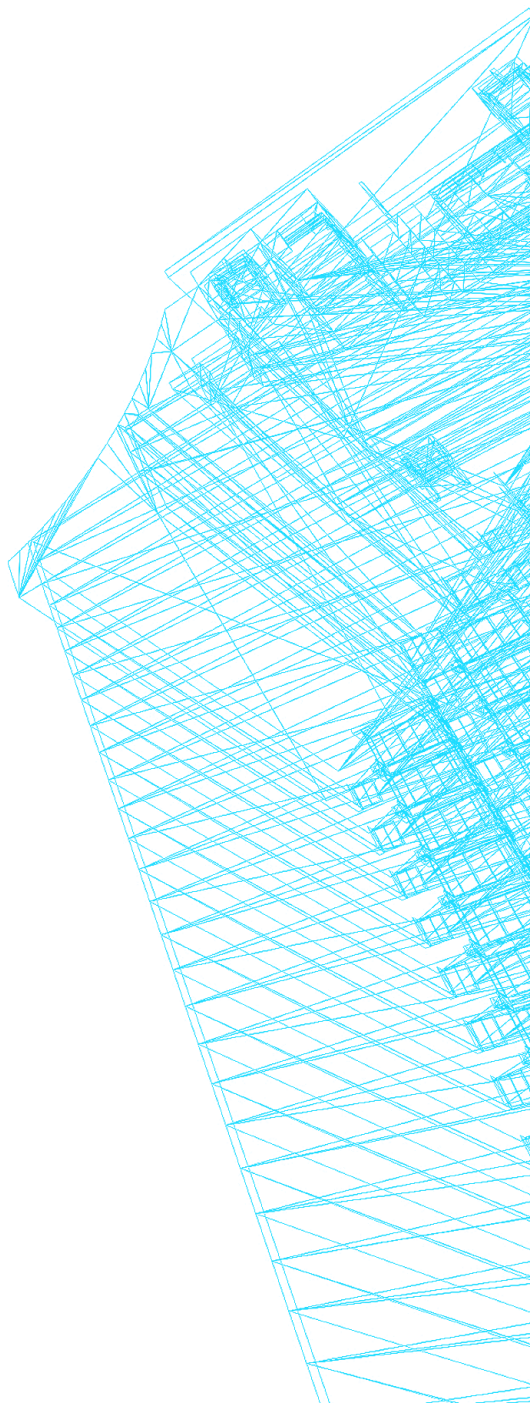
충북컴퓨터 꿈나무축제 피지컬컴퓨팅부문

2023. 기출문제 풀이

- 중등부 지역대회(청주시) Up and Down
- 중등부 지역대회(청주외) 주차타워
- 중등부 도대회 득점전광판

- 참고자료: https://arduino.datahub.pe.kr



2023. 기출문제 분석

구분	중등부 지역대회 청주시	중등부 지역대회 청주외	중등부 도대회
문제	**Up and Down Game**	주차 타워	득점 전광판
요구 기술	**난수 뽑기** **버튼 입력** **사운드 출력** 초음파 센서 시리얼 입력 1자리 FND출력	버튼 입력 사운드 출력 **초음파 센서** **시리얼 입력** **1자리 FND출력**	버튼 입력 사운드출력 1자리 FND출력 가변저항 **4자리 FND 출력** **FND 클래스**

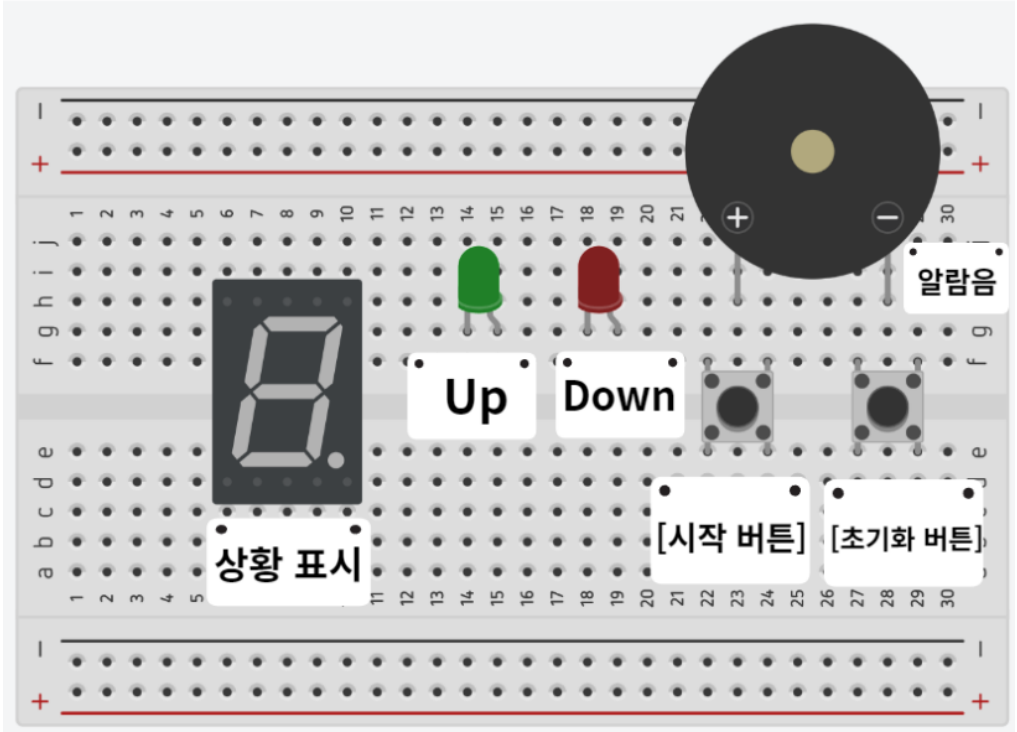
2023. 중등부 지역대회(청주시) UP & DOWN

- 상황제시

게임 개발에 관심이 많은 충북이는 Up & Down Game을 직접 제작하여 친구들과 함께 게임을 즐기고자 한다.

Up & Down Game은 1~100 사이의 임의로 정해진 숫자를 맞추는 게임이며 총 10번의 기회가 주어지고 적은 횟수로 빠른 시간 안에 정답을 맞추어야 높은 점수를 얻을 수 있다.

- 예시설계



2023. 중등부 지역대회(청주시) UP & DOWN

- 필수해결 요소

가. [시작 버튼]을 누르면 시작 알람음이 울리면서 세그먼트에는 0이 표시되고 2개의 LED가 동시에 켜지며 게임의 시작을 알린다.

나. 게임이 시작되면 1~100 사이의 임의의 숫자가 정답으로 정해지고 시리얼 모니터에 사용자가 예상하는 숫자를 입력하도록 안내하는 메세지가 출력된다.

다. 정답이 사용자가 입력한 숫자보다 크다면 Up, 작다면 Down, 일치한다면 Clear, 10번의 기회를 모두 사용하면 Fail로 상황을 정의하고 해당 상황에 맞는 안내 메세지를 아래와 같이 출력한다.

	Up	Down	Clear	Fail
LED	초록색 LED 켜짐	빨간색 LED 켜짐	모든 LED 켜짐	모든 LED 꺼짐
세그먼트	남은 횟수 표시			
피에조 스피커	상황에 적절한 알람음 출력			

2023. 중등부 지역대회(청주시) UP & DOWN

라. 사용자가 숫자를 입력하면 Up, Down, Clear, Fail 상황과 함께 남은 횟수, 현재 점수, 게임 진행 시간을 시리얼 모니터에 출력한다.
(점수는 100점부터 시작하며 1회씩 틀릴 때마다 10점씩 감점된다.)

◆ (예시) 정답이 77인 경우

<시리얼 모니터>		
숫자 입력: 50 	숫자 입력: 80 	숫자 입력: 77 
Up!	Down!	Clear!
남은 횟수: 9	남은 횟수: 8	남은 횟수: 8
현재 점수: 90	현재 점수: 80	현재 점수: 80
경과 시간: 10	경과 시간: 15	경과 시간: 20

마. [시작 버튼]을 누르면 다시 게임이 시작되며 [초기화 버튼]을 누르면 모든 게임 진행 상황과 기록이 초기화된다.

2023. 중등부 지역대회(청주시) UP & DOWN

◈ 창의적 문제 해결

가. 필수해결 요소의 (가~마) 항을 필히 포함하고 그 외 필수부품 내에서 창의적으로 작품을 완성하고 문제를 해결한다.

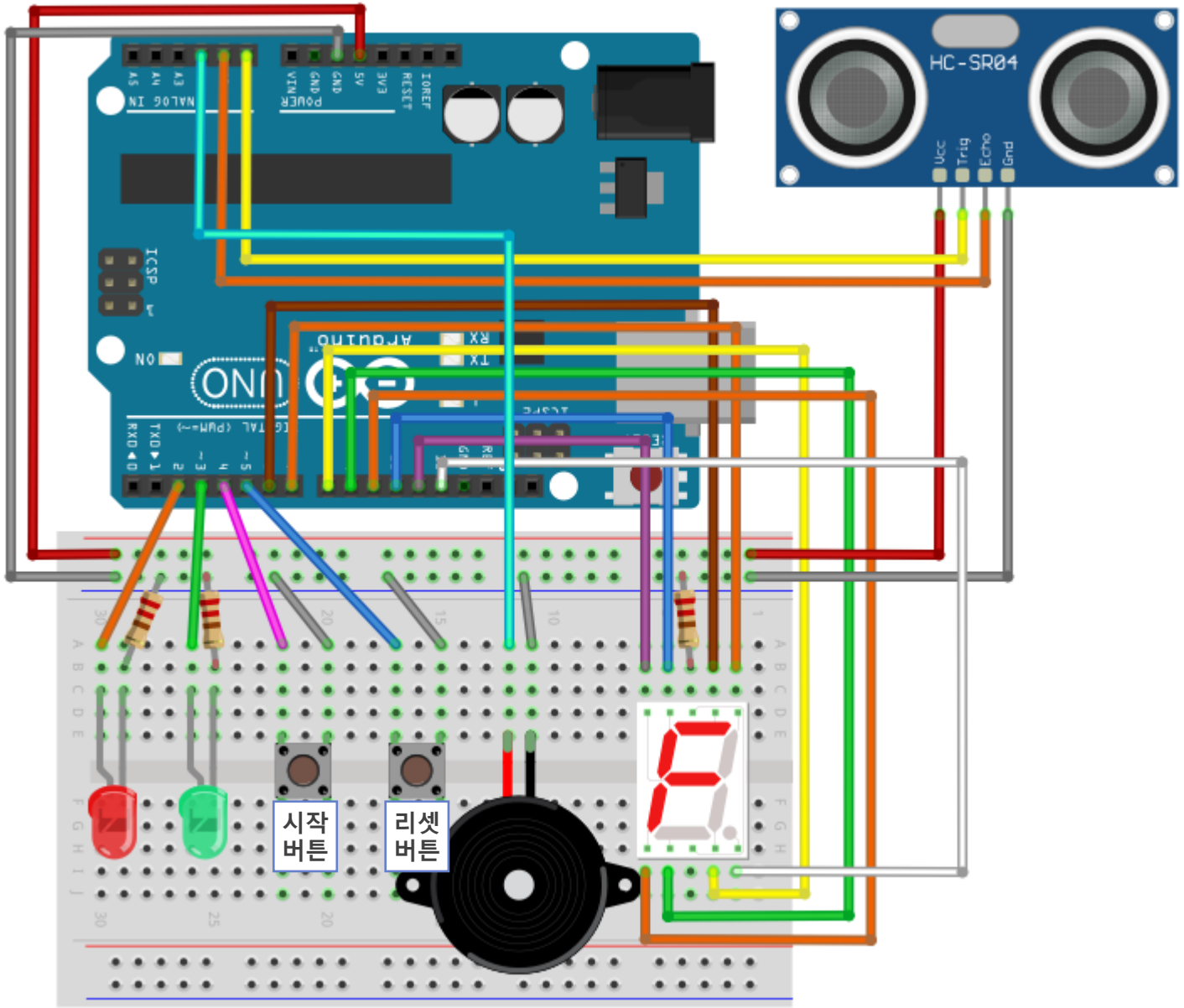
예1) 초음파 센서를 사용하여 센서에 손을 가까이 접근하면 현재 게임 진행 시간이 실시간으로 시리얼 모니터에 안내된다.

예2) 게임 기록을 내림차순 또는 오름차순으로 정렬하여 순위 정보와 함께 출력한다.

예3) 특정 시점에 사용자에게 힌트를 제공하는 기능을 추가한다.

2023. 중등부 지역(청주시) UP & DOWN

- 결선




```
#define USW_TRIG   A0
#define USW_ECHO   A1
#define BUZZER     A2
#define LED_RED     2
#define LED_GREEN   3
#define BTN_START   4
#define BTN_RESET   5

#define FND_A       6
#define FND_B       7
#define FND_C       8
#define FND_D       9
#define FND_E      10
#define FND_F      11
#define FND_G      12
#define FND_DP     13
```

```
#define USW_TRIG  A0
#define USW_ECHO  A1
#define LED_RED   2
#define LED_GREEN 3
#define BTN_START 4
#define BTN_RESET 5
#define BUZZER    6

#define OFF   LOW
#define ON    HIGH
typedef unsigned long ulong;

// 초음파 센서로 거리를 측정하여 cm단위로 반환
int getUswDistance() {
  long duration;
  int distanceCm;

  digitalWrite(USW_TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(USW_TRIG, LOW);

  duration = pulseIn(USW_ECHO, HIGH);
  distanceCm = 0.034 * duration / 2;
  return distanceCm;
}
```

```
const byte maxChars = 32;  // 최대 문자열 길이
char receivedChars[maxChars];
// endMarker 문자가 나올때까지 입력을 받아냄
// 받은 입력은 receivedChars 문자배열에 저장됨
void serialInputUntileEndMarker(char endMarker) {
  byte ndx = 0;
  char rc = -1;

  while (true) {
    checkSensors();
    if (Serial.available() > 0) {
      rc = Serial.read();

      if (rc != endMarker && ndx < maxChars) {
        receivedChars[ndx] = rc;
        ndx++;
      }
      else {    // endMaker에 닿았거나 maxChars를 넘어섰다면,
        receivedChars[ndx] = '\0';  // 문자열 종료
        Serial.println(receivedChars);
        return;  // 탈출
      }
    }
  }
}

// 모든 입력 버퍼를 비움, 시리얼 입력 받기 직전에 사용
void serialFlush() {
  while (Serial.available() > 0) {
    char t = Serial.read();
  }
}
```

```
void switchOnOffLed(bool ledRed, bool ledGreen) {
  digitalWrite(LED_RED, ledRed);
  digitalWrite(LED_GREEN, ledGreen);
}

bool isPressedStartButton() {
  static bool preState;
  bool curState = digitalRead(BTN_START);
  bool isPressed;

  // INPUT_PULLUP 버튼이므로, 버튼을 누르는 순간 하강에지가 발생한다.
  if (preState == HIGH && curState == LOW)  // 하강에지라면,
    isPressed = true;
  else
    isPressed = false;

  preState = curState;
  return isPressed;
}

bool isPressedResetButton() {
  static bool preState;
  bool curState = digitalRead(BTN_RESET);
  bool isPressed;

  // INPUT_PULLUP 버튼이므로, 버튼을 누르는 순간 하강에지가 발생한다.
  if (preState == HIGH && curState == LOW)  // 하강에지라면,
    isPressed = true;
  else
    isPressed = false;

  preState = curState;
  return isPressed;
}
```

```
void playStartSound() { // 미솔
  tone(BUZZER, 330, 400);
  delay(400);
  tone(BUZZER, 392, 400);
}

void playRightSound() { // 도미솔도
  tone(BUZZER, 262, 300);
  delay(300);
  tone(BUZZER, 330, 300);
  delay(300);
  tone(BUZZER, 392, 300);
  delay(300);
  tone(BUZZER, 523, 300);
}

void playResetSound() { // 시
  tone(BUZZER, 988, 800);
}

void playUpwardSound() { // 솔도
  tone(BUZZER, 391, 400);
  delay(400);
  tone(BUZZER, 523, 400);
}

void playFallingSound() { // 도솔
  tone(BUZZER, 523, 400);
  delay(400);
  tone(BUZZER, 391, 400);
}
```

2023. 중등부 지역(청주시) UP & DOWN

```
int addGameRecord(record rec) {
  int i = 0;
  for(i=0; i<gGameRecordsNum; i++) {
    if(rec.tryNum<gGameRecords[i].tryNum ||
       (rec.tryNum==gGameRecords[i].tryNum && rec.secs<gGameRecords[i].secs))
      break;
  }
  int suitableIdx = i;
  for(i=gGameRecordsNum; i>=suitableIdx; i--) {
    gGameRecords[i] = gGameRecords[i-1];
  }
  gGameRecords[suitableIdx] = rec;
  gGameRecordsNum++;
  if(gGameRecordsNum > MAX_RECORDS) // 최대 저장 갯수를 넘어서지 않도록..
    gGameRecordsNum = MAX_RECORDS;

  return suitableIdx;
}

void outputGameRecords() {
  char buf[60];
  for(int i=0; i<gGameRecordsNum; i++) {
    sprintf(buf, "[%d위] 시도횟수: %d회, 걸린시간: %d초, 정답숫자: %d",
      i+1, gGameRecords[i].tryNum, gGameRecords[i].secs, gGameRecords[i].answer);
    Serial.println(buf);
  }
}
```

6	60	33

index	tryNum	secs	answer
0	2	30	25
1	4	55	12
2	6	71	97
3	7	70	3
4			
5			
6			
7			
8			
9			

index	tryNum	secs	answer
0	2	30	25
1	4	55	12
2			
3	6	71	97
4	7	70	3
5			
6			
7			
8			

난수 생성

- radmon() 함수
  - 설명
    - 의사 난수(psudo-random)를 생성
  - 문법
    - random(max)
    - random(min, max)
  - 매개변수
    - min: 무작위 값의 하한 값(포함됨)
    - max: 무작위 값의 상한 값(제외됨).
  - 반환 값
    - min 과 max-1 사이의 임의의 숫자
  - 자료형: long.

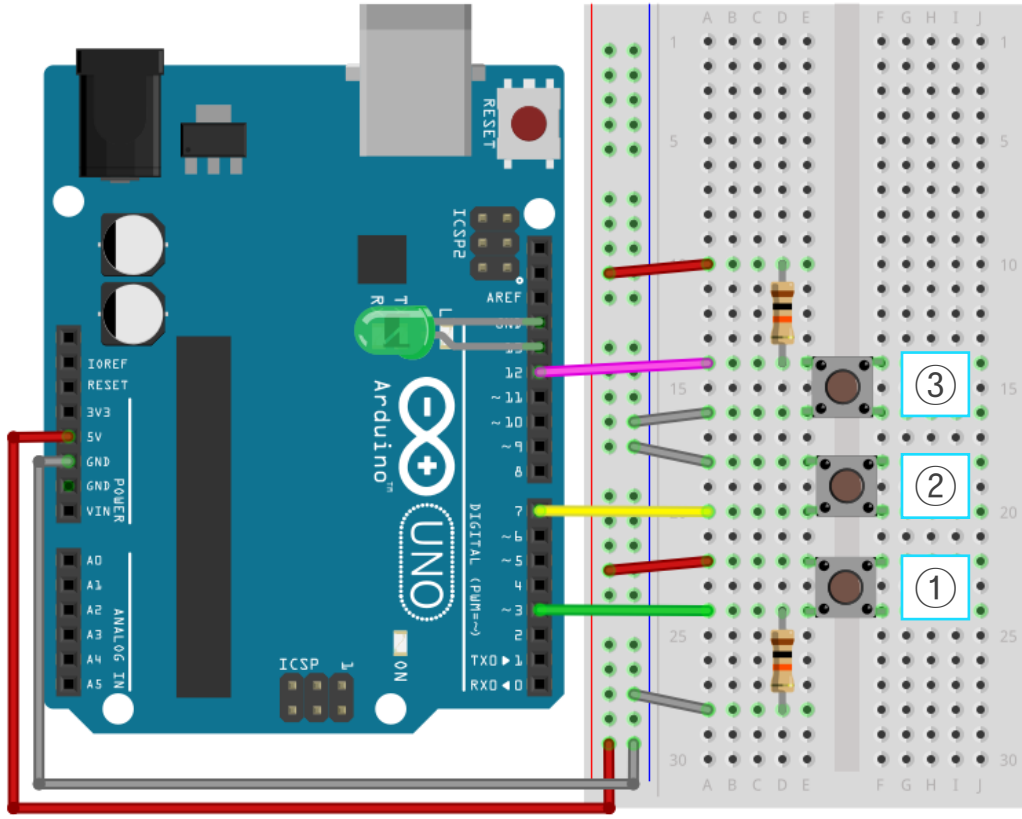
- Example coode

```
long randNumber;

void setup() {
  Serial.begin(9600);

  Serial.print("analog port noise: ");
  for(int i=0; i<10; i++) {
    Serial.print(analogRead(A5), DEC);
    Serial.print("-");
    delay(100);
  }
  randomSeed(analogRead(A5));
}

void loop() {
  // print a random number from 0 to 299
  randNumber = random(300);
  Serial.println(randNumber);

  // print a random number from 1 to 100
  randNumber = random(1, 101);
  Serial.println(randNumber);
  delay(2000);
}
```

버튼 회로 설계

- 버튼 회로별 설계 방법 요약

구분	①Active HIGH 버튼	②Active LOW 버튼	③Active LOW 버튼
눌렀을 때	HIGH	LOW	LOW
떼었을 때	LOW	HIGH	HIGH
버튼 연결	Vcc – IN	GND – IN	GND – IN
저항유무	필요함	불필요	필요함
저항위치	GND	-	Vcc
방법	풀다운 저항	MCU내부 저항	풀업 저항
pinMode	INPUT	INPUT_PULLUP	INPUT

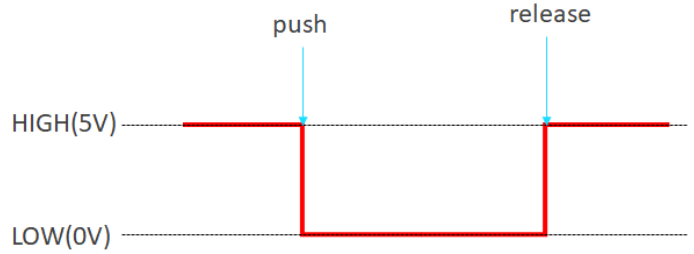


버튼 입력

- INPUT_PULLUP 버튼

```
#define BTN_START 4
#define BTN_RESET 5

void setup() {
  pinMode(BTN_START, INPUT_PULLUP);
  pinMode(BTN_RESET, INPUT_PULLUP);
}

bool isPressedStartButton() {
  static bool preState = HIGH;
  bool curState = digitalRead(BTN_START);
  bool isPressed;

  // INPUT_PULLUP 버튼이므로, 버튼을 누르는 순간 하강에지가 발생.
  if (preState == HIGH && curState == LOW)  // 하강에지라면,
    isPressed = true;
  else
    isPressed = false;

  preState = curState;
  return isPressed;
}
```

- 상태도

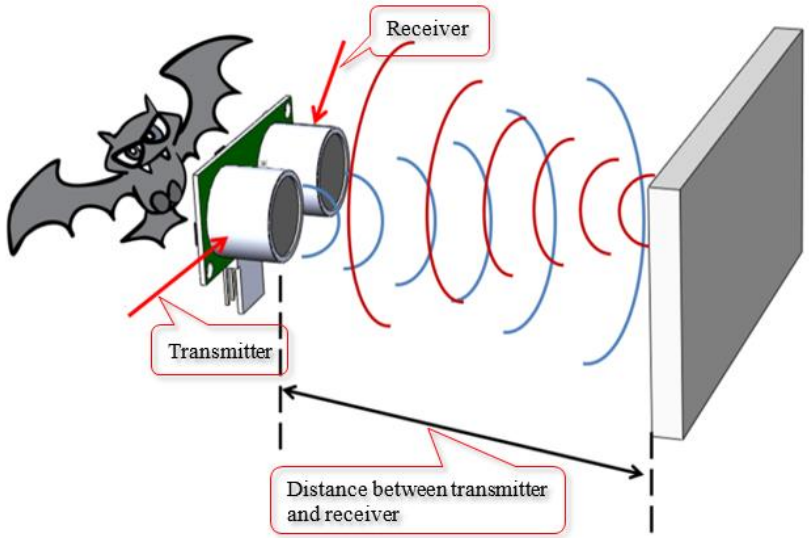



```
bool isPressedResetButton() {
  static bool preState = HIGH;
  bool curState = digitalRead(BTN_RESET);
  bool isPressed;

  // INPUT_PULLUP 버튼이므로, 버튼을 누르는 순간 하강에지가 발생
  if (preState == HIGH && curState == LOW)  // 하강에지라면,
    isPressed = true;
  else
    isPressed = false;

  preState = curState;
  return isPressed;
}

void loop() {
  if(isPressedStartButton()) {
    // to-do code
  }

  if(isPressedResetButton()) {
    // to-do code
  }
}
```

초음파 센서

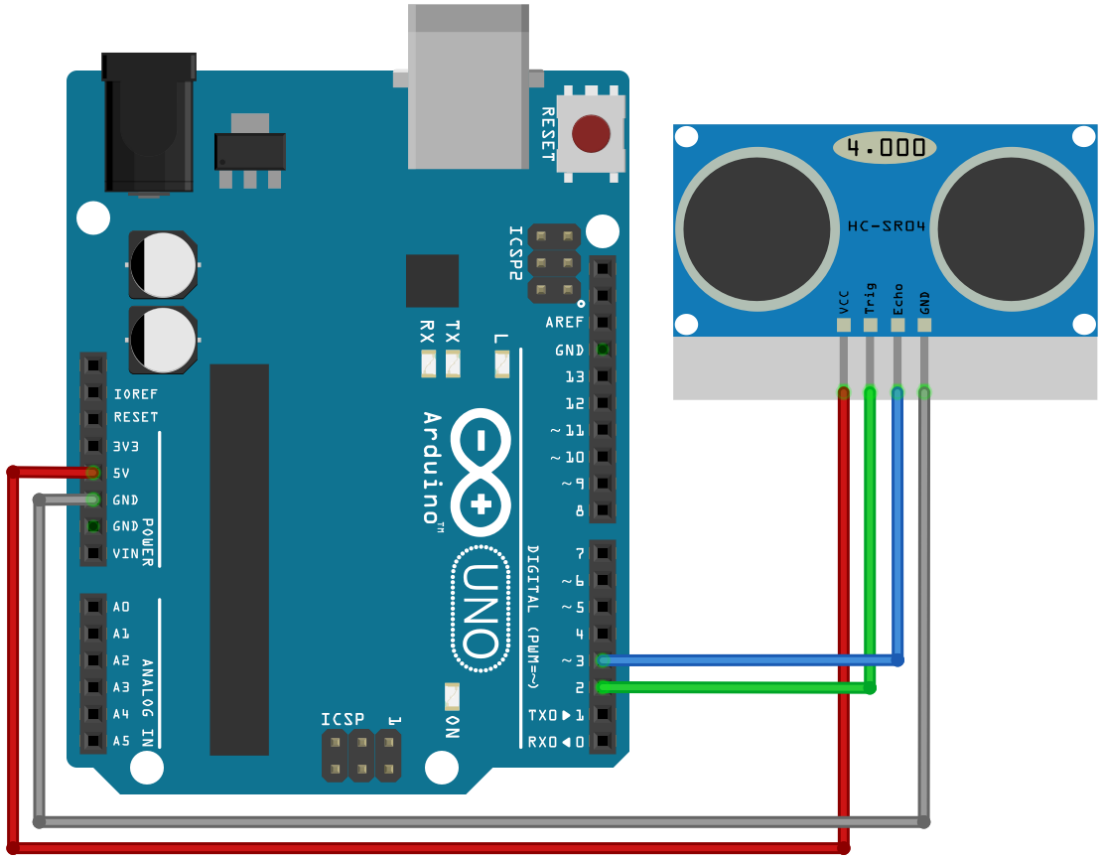
- 외관

- 작동원리



초음파 센서

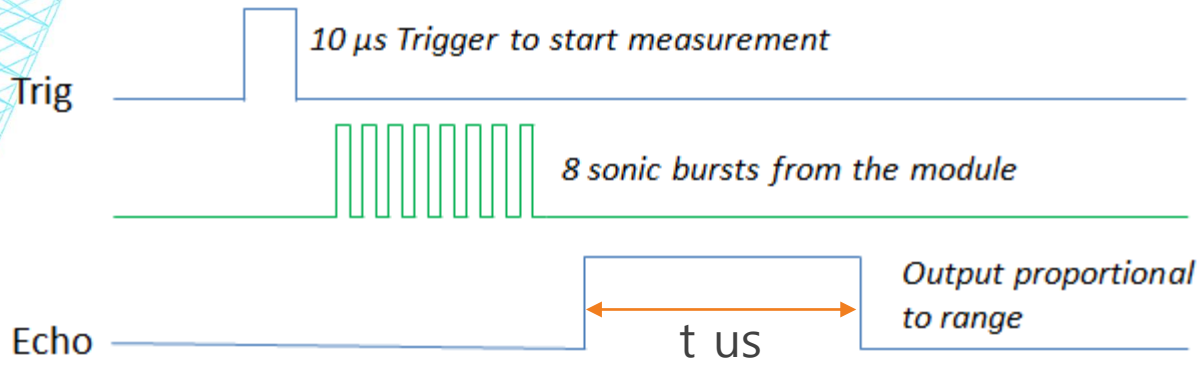
- 회로구성



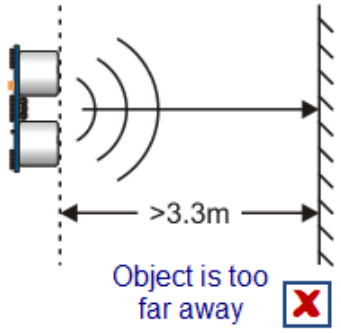

```
#define TRIG 2
#define ECHO 3

void setup() {
  pinMode(TRIG, OUTPUT);
  pinMode(ECHO, INPUT);
  Serial.begin(9600);
}
```

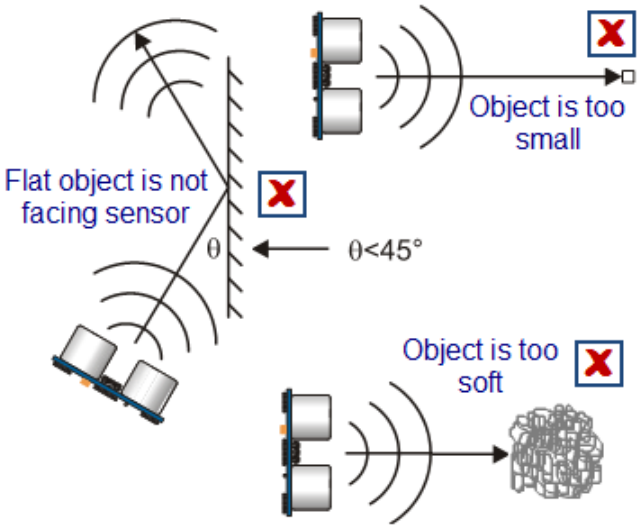
초음파 센서

- 거리 계산 방법



- 감지 한계





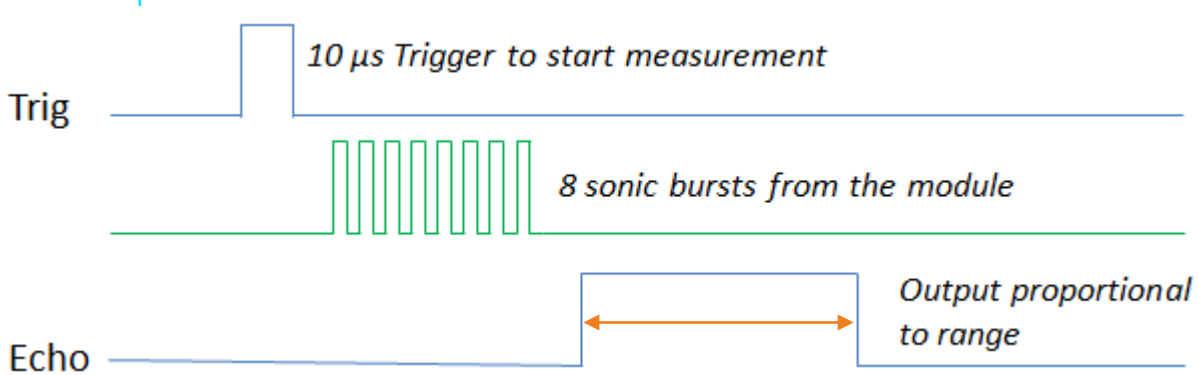
- 음속: 340m/s = 34000cm/s = 0.034cm/us
- v = d / t
- d = v * t
- 2d = 0.034cm x t;
- d = (0.034cm * t) / 2;

초음파 센서

4-7.ino

```
#define TRIG 2
#define ECHO 3

void setup() {
  pinMode(TRIG, OUTPUT);
  pinMode(ECHO, INPUT);
  Serial.begin(9600);
}

void loop() {
  long duration;
  int distanceCm;

  digitalWrite(TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG, LOW);

  duration = pulseIn(ECHO, HIGH);
  distanceCm= 0.034*duration/2;
  Serial.print(distanceCm);
  Serial.println("cm");

  delay(300);
}
```

- `pulseIn(pin, value)`
  - value가 HIGH이면, pulseIn()은 핀이 HIGH가 되었다가 LOW가 될 때까지의 펄스의 길이를 마이크로초 단위로 반환한다. 단, 타임아웃 내에 완전한 펄스가 수신되지 않은 경우 0을 반환합니다.



사운드 출력

- 능동 부저
  - 전압이 걸리면 내부 발진기를 사용하여 미리 정해 놓은 소리가 만들어진다.

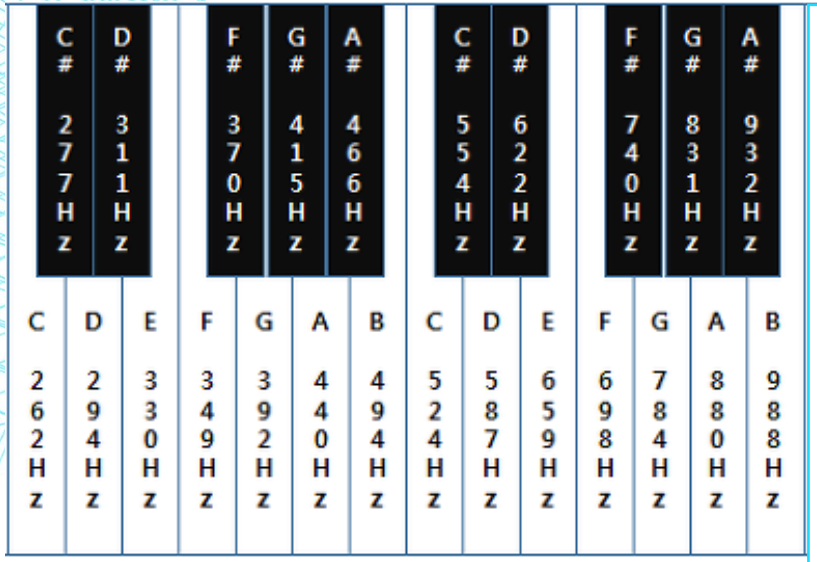
- 수동 부저
  - 내부 발진기가 없으므로 직접 주파수를 발생시켜 소리를 만들어 내야 한다.

tone/noTone 함수

- **tone(pin, freq [, duration]);**
  - pin : 부저나 스피커가 연결된 디지털 핀 번호
  - freq : 주파수 (범위 : 31 ~ 65535), 31보다 작은 값을 넣어주면 잡음 출력
  - duration : (옵션) 음의 발생 시간 (밀리초 단위)
- 핀에 지정된 주파수(50 % 듀티 사이클)의 구형파를 발생시킨다. 지속 시간을 정할 수 있으며 따로 지정하지 noTone()을 호출 할 때까지 계속 된다. 핀은 피에조 부저 또는 스피커에 연결하여 톤을 재생할 수 있다.
- 한 번에 하나의 톤 만 생성 할 수 있다. tone()이 이미 다른 핀에서 재생 중이면 tone () 호출은 아무 효과가 없다. tone() 함수를 사용하면 (Mega 이외의 보드에서) 3번과 11번 핀 의 PWM 출력을 방해한다.
- 이 함수는 non-blocking 함수입니다. 즉, duration 매개변수를 제공하여 톤 재생이 완료되지 않은 경우에도 다음 코드의 실행이 즉시 계속됩니다.
- **noTone(pin);**

사운드 출력

- 적절한 효과음 출력 함수 제작

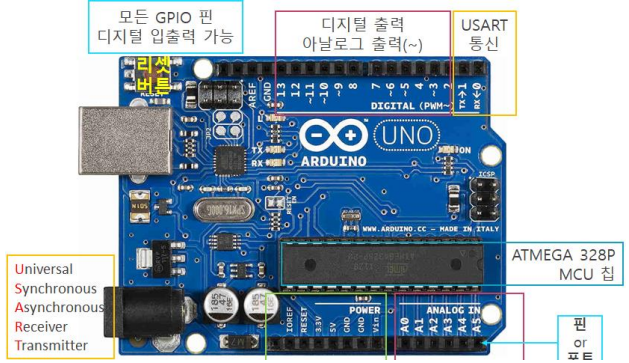



```
#define BUZZER    A2

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

void playStartSound() { // 미솔
  tone(BUZZER, 330, 400);
  delay(400);
  tone(BUZZER, 392, 400);
}

void playRightSound() { // 도미솔도
  tone(BUZZER, 262, 300);
  delay(300);
  tone(BUZZER, 330, 300);
  delay(300);
  tone(BUZZER, 392, 300);
  delay(300);
  tone(BUZZER, 523, 300);
}

void playResetSound() { // 시
  tone(BUZZER, 988, 800);
}
```

```
void playUpwardSound() { // 솔도
  tone(BUZZER, 391, 400);
  delay(400);
  tone(BUZZER, 523, 400);
}

void playFallingSound() { // 도솔
  tone(BUZZER, 523, 400);
  delay(400);
  tone(BUZZER, 391, 400);
}

void loop() {
  Serial.println("playStartSound()");
  playStartSound();
  delay(2000);

  Serial.println("playRightSound()");
  playRightSound();
  delay(2000);

  Serial.println("playResetSound()");
  playResetSound();
  delay(2000);

  Serial.println("playUpwardSound()");
  playUpwardSound();
  delay(2000);

  Serial.println("playFallingSound()");
  playFallingSound();
  delay(10000);
}
```

시리얼 입력 얻어내기

- 문자열 입력 받기 & 숫자로 변환

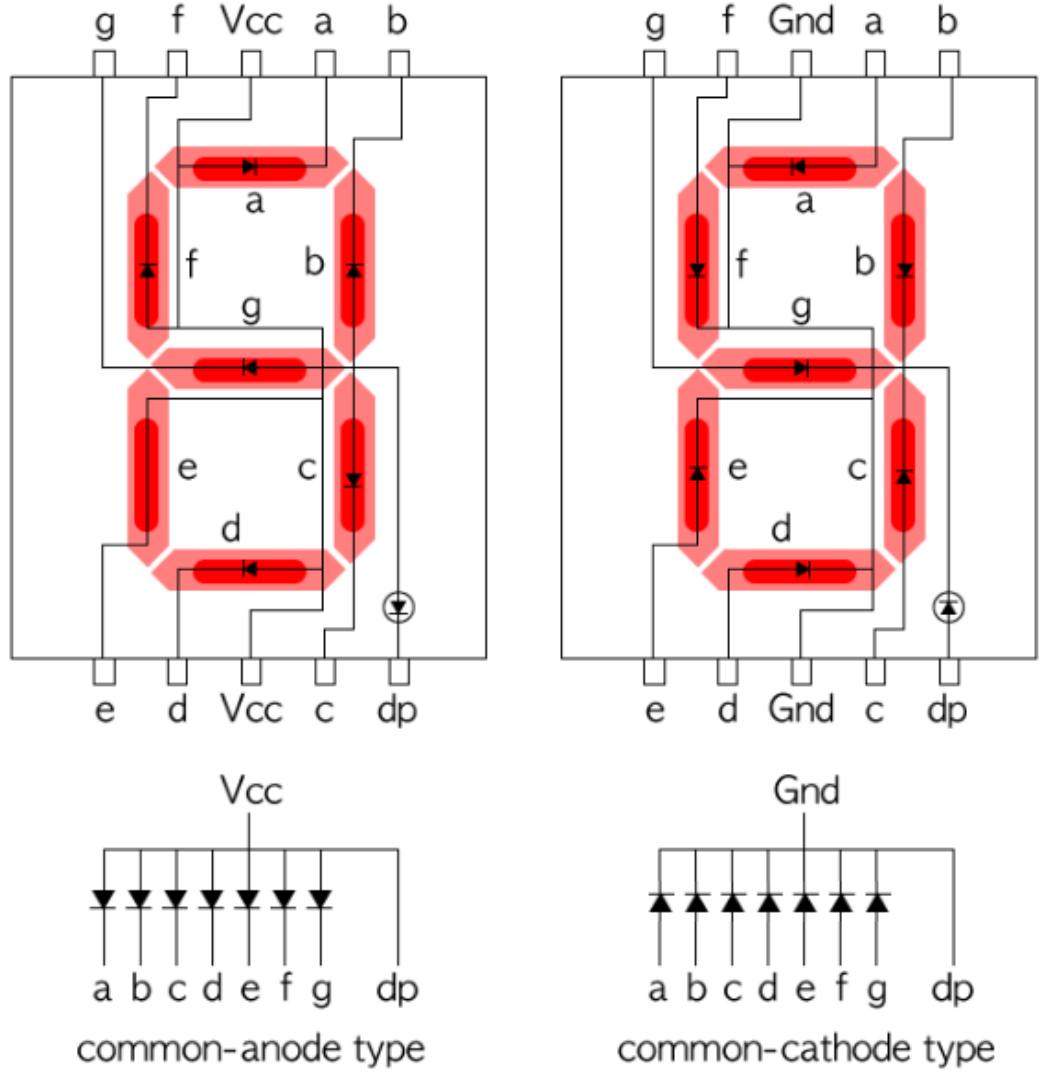



```ino
void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.print("\n숫자 입력: ");
  serialFlush();
  serialInputUntileEndMarker('\n');
  int num = atoi(receivedChars);
  Serial.println(num, DEC);
}

// 모든 입력 버퍼를 비움, 시리얼 입력 받기 직전에 사용
void serialFlush() {
  while (Serial.available() > 0) {
    char t = Serial.read();
  }
}
```

```ino
const byte maxChars = 32;  // 최대 문자열 길이
char receivedChars[maxChars];
// endMarker 문자가 나올때까지 입력을 받아냄
// 받은 입력은 receivedChars 문자배열에 저장됨

void serialInputUntileEndMarker(char endMarker) {
  byte ndx = 0;
  char rc = -1;

  while(true) {
    if (Serial.available() > 0) {
      rc = Serial.read();

      if (rc != endMarker && ndx < maxChars) {
        receivedChars[ndx] = rc;
        ndx++;
      }
      else {   // endMaker에 닿았거나 maxChars를 넘어섰다면,
        receivedChars[ndx] = '\0';  // 문자열 종료
        return;  // 탈출
      }
    }
  }
}
```

FND(7 Segment LED) 내부구조

- 공통 양극(common-anode) type
  - 공통 Vcc
  - 전류가 Vcc에서 공급되어 MCU의 핀으로 흘러 들어가는 방식
  - MCU 핀이 LOW 일 때 켜짐
- 공통 음극(common-cathode) type
  - 공통 GND
  - 전류가 MCU의 핀에서 공급되어 GND 단자로 흘러 나가는 방식
  - MCU 핀이 HIGH 일 때 켜짐



FND에 숫자 출력하기

- 고찰
  - 조각난 LED를 조합하여 특정 글자를 만들려면 어떤 LED를 켜고 꺼야 하는가에 대한 정보가 필요
  - LED 세그먼트 개수는 (a, b, c, d, e, f, g, dp) 총 8개
  - 1 세그먼트마다 on 인지 off 인지 두 가지 상태만 저장하면 되므로 1비트로 충분
  - 8세그먼트 = 8bit = 1byte
  - 0 을 그리려면 a,b,c,d,e,f를 켜야 한다라는 정보를 단, 1바이트인 3F 로 표현

- FND_FONT

DISP	DP	g	f	e	d	c	b	a	HEX
	8	4	2	1	8	4	2	1	
	0	0	1	1	1	1	1	1	3F
	0	0	0	0	0	1	1	0	06
	0	1	0	1	1	0	1	1	5B
	0	1	0	0	1	1	1	1	4F
	0	1	1	0	0	1	1	0	66
	0	1	1	0	1	1	0	1	6D
	0	1	1	1	1	1	0	1	7D
	0	0	0	0	0	1	1	1	07
	0	1	1	1	1	1	1	1	7F
	0	1	1	0	0	1	1	1	67

FND에 숫자 출력하기

DISP	DP	g	f	e	d	c	b	a
	8	4	2	1	8	4	2	1
	0	0	1	1	1	1	1	1

fnd1_num.ino

```
// common cathode example
#define FND_A  2
#define FND_B  3
#define FND_C  4
#define FND_D  5
#define FND_E  6
#define FND_F  7
#define FND_G  8
#define FND_DP 9

byte FND_FONT[] = {
   0x3F, 0x06, 0x5B, 0x4F, 0x66,
   0x6D, 0x7D, 0x07, 0x7F, 0x67  };

bool dot = false;

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

```
void outputFND(int num, bool dot) {
    byte maskbit = B00000001;
    byte font = FND_FONT[num] | (dot<<7);

    for(int i=FND_A; i<=FND_DP; i++) {
        if((font & maskbit) != 0)
            digitalWrite(i, HIGH);
        else
            digitalWrite(i, LOW);

        maskbit = maskbit << 1;
    }
}

void loop() {
    for(int num=0; num<=9; num++) {
        outputFND(num, dot);
        delay(500);
    }
    dot = !dot;
}
```

```
(FONT)   0 0 1 1 1 1 1 1
(MASK) & 0 0 0 0 0 0 0 1
ON/OFF   0 0 0 0 0 0 0 1

(FONT)   0 0 1 1 1 1 1 1
(MASK) & 0 0 0 0 0 0 1 0
ON/OFF   0 0 0 0 0 0 1 0

(FONT)   0 0 1 1 1 1 1 1
(MASK) & 0 0 0 0 0 1 0 0
ON/OFF   0 0 0 0 0 1 0 0

(FONT)   0 0 1 1 1 1 1 1
(MASK) & 0 0 0 0 1 0 0 0
ON/OFF   0 0 0 0 1 0 0 0

(FONT)   0 0 1 1 1 1 1 1
(MASK) & 0 0 0 1 0 0 0 0
ON/OFF   0 0 0 1 0 0 0 0

(FONT)   0 0 1 1 1 1 1 1
(MASK) & 0 0 1 0 0 0 0 0
ON/OFF   0 0 1 0 0 0 0 0

(FONT)   0 0 1 1 1 1 1 1
(MASK) & 0 1 0 0 0 0 0 0
ON/OFF   0 0 0 0 0 0 0 0

(FONT)   0 0 1 1 1 1 1 1
(MASK) & 1 0 0 0 0 0 0 0
ON/OFF   0 0 0 0 0 0 0 0
```

common cathode 일 때 (active high)

DISP	DP	g	f	e	d	c	b	a	HEX
	8	4	2	1	8	4	2	1	
0	0	0	1	1	1	1	1	1	3F
1	0	0	0	0	0	1	1	0	06
2	0	1	0	1	1	0	1	1	5B
3	0	1	0	0	1	1	1	1	4F
4	0	1	1	0	0	1	1	0	66
5	0	1	1	0	1	1	0	1	6D
6	0	1	1	1	1	1	0	1	7D
7	0	0	0	0	0	1	1	1	07
8	0	1	1	1	1	1	1	1	7F
9	0	1	1	0	0	1	1	1	67
A	0	1	1	1	0	1	1	1	77
b	0	1	1	1	1	1	0	0	7C
C	0	0	1	1	1	0	0	1	39
d	0	1	0	1	1	1	1	0	5E
E	0	1	1	1	1	0	0	1	79
F	0	1	1	1	0	0	0	1	71

상호 변환은
bit not 연산자
~ 로 가능하다.

~(0x3F) = 0xC0

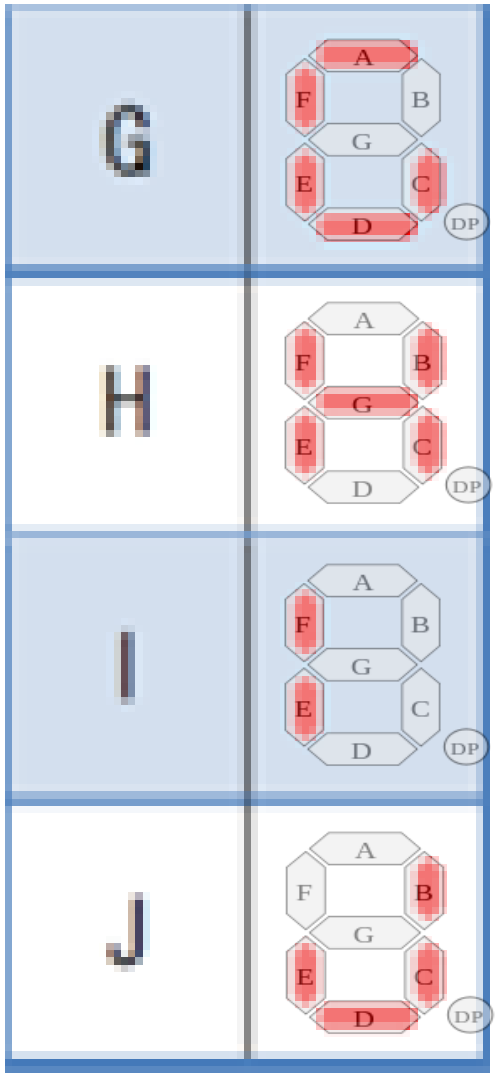
common anode 일 때 (active low)

DISP	DP	g	f	e	d	c	b	a	HEX
	8	4	2	1	8	4	2	1	
0	1	1	0	0	0	0	0	0	C0
1	1	1	1	1	1	0	0	1	F9
2	1	0	1	0	0	1	0	0	A4
3	1	0	1	1	0	0	0	0	B0
4	1	0	0	1	1	0	0	1	99
5	1	0	0	1	0	0	1	0	92
6	1	0	0	0	0	0	1	0	82
7	1	1	1	1	1	0	0	0	F8
8	1	0	0	0	0	0	0	0	80
9	1	0	0	1	0	0	0	0	90
A	1	0	0	0	1	1	0	0	8C
b	1	0	1	0	1	1	1	1	BF
C	1	1	0	0	0	1	1	0	C6
d	1	0	1	0	0	0	0	1	A1
E	1	0	0	0	0	1	1	0	86
F	1	0	0	0	1	1	1	0	8E

FND_FONT 추가

- 2022 지역대회 (중) 7세그먼트의 원리와 활용

0	0	A	A	K	K	U	U
1	1	B	b	L	L	V	V
2	2	C	c	M	M	W	W
3	3	D	d	N	n	X	X
4	4	E	E	O	o	Y	Y
5	5	F	F	P	P	Z	Z
6	6	G	G	Q	q		
7	7	H	H	R	r		
8	8	I	I	S	S		
9	9	J	J	T	t		



DISP	DP	g	f	e	d	c	b	a	HEX
	8	4	2	1	8	4	2	1	
G	0	0	1	1	1	1	0	1	3D
H	0	1	1	1	0	1	1	0	76
I	0	0	1	1	0	0	0	0	30
J	0	0	0	1	1	1	1	0	1E
K	0	1	1	1	1	0	1	0	7A
L	0	0	1	1	1	0	0	0	38
M	0	1	0	1	0	1	0	1	55
N	0	1	0	1	0	1	0	0	54
O	0	1	0	1	1	1	0	0	5C
P	0	1	1	1	0	0	1	1	73
Q	0	1	1	0	0	1	1	1	67
R	0	1	0	1	0	0	0	0	50
S	0	1	1	0	1	1	0	1	6D
T	0	1	1	1	1	0	0	0	78
U	0	0	1	1	1	1	1	0	3E
V	0	1	1	1	1	1	1	0	7E
W	0	1	1	0	1	0	1	0	6A
X	0	0	1	1	0	1	1	0	36
Y	0	1	1	0	1	1	1	0	6E
Z	0	1	0	0	1	0	0	1	49

FND에 숫자&문자 출력하기

fnd1_num_char.ino

```
#define FND_A  2
#define FND_B  3
#define FND_C  4
#define FND_D  5
#define FND_E  6
#define FND_F  7
#define FND_G  8
#define FND_DP 9

unsigned char FND_FONT[] =
   // 0,    1,    2,    3,    4,    5,    6,    7,    8,    9,
  {0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x67,
   // A,    B,    C,    D,    E,    F,    G,    H,    I,    J,
   0x77, 0x7C, 0x39, 0x5E, 0x79, 0x71, 0x3D, 0x76, 0x30, 0x1E,
   // K,    L,    M,    N,    O,    P,    Q,    R,    S,    T,
   0x7A, 0x38, 0x55, 0x54, 0x5C, 0x73, 0x67, 0x50, 0x6D, 0x78,
   // U,    V,    W,    X,    Y,    Z,   공백
   0x3E, 0x7E, 0x6A, 0x36, 0x6E, 0x49, 0x00 };

bool dot = false;

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

```
void outputFND(int num, bool dot) {
    byte maskbit = B00000001;
    byte font = FND_FONT[num] | (dot<<7);

    for(int i=FND_A; i<=FND_DP; i++) {
        if((font & maskbit) != 0)
            digitalWrite(i, HIGH);
        else
            digitalWrite(i, LOW);

        maskbit = maskbit << 1;
    }
}

void loop() {
    for(int num=0; num<=9; num++) {
        outputFND(num, dot);
        delay(500);
    }
    for(int ch='A'; ch<='Z'; ch++) {
        int idx = ch-'A'+10;
        outputFND(idx, dot);
        delay(500);
    }
    dot = !dot;
}
```

2023. 중등부 지역(청주외) 주차타워

- 상황제시

주차 타워 시스템을 제작하려고 한다. 주차 타워 입구에는 무인 주차 차단기가 설치되어 있으며 자동차 운전자가 수동으로 번호판을 입력하고 출입이 허가되면 진입할 수 있다. 주차 타워에는 [주차 버튼]과 [출고 버튼]을 통해 주차와 출고가 가능하며 출고 시 자동으로 주차료가 계산되어 안내된다.

2023. 중등부 지역(청주외) 주차타워

◈ 필수해결 요소

가. 자동차가 무인 주차 차단기에 접근하는 거리를 초음파 센서로 측정하고, 일정 거리 이하로 접근하면 시리얼 모니터에 차량 번호판을 입력할 수 있도록 안내한다. (초음파 센서 사용이 힘들 경우, 시리얼 입력 등을 사용하여도 되지만 감점 처리 됨)

나. 자동차 번호판 4자리를 시리얼 모니터로 입력 받으면 주차장 타워 입구로 출입이 허가되었음을 시리얼 모니터, LED, 피에조 스피커로 안내한다.

◆ (예시) 차량 번호판이 입력 되면 시리얼 모니터로 인식이 완료되었음을 알리고 LED의 색깔이 빨간색에서 초록색으로 바뀌며 피에조 스피커에서 출입 알람음이 출력된다.

<시리얼 모니터>

차량 번호판 4자리 입력: 3739 

3739 차량 입장 허가!

다. 무인 주차 차단기를 통과하고 [주차 버튼]을 누르면 해당 차량이 가장 낮은 층수부터 주차되고 시리얼 모니터에 주차 현황이 출력된다. 주차 타워는 총 6층이며 1층에 1대씩 주차가 가능하도록 구현한다.

(주차 현황 등의 안내말은 영어를 사용해도 무방하며 자유로운 형식으로 출력하되, 사용자가 알기 쉽게 표현할 것.)

- **(예시)** [주차 버튼]을 누르면 차량이 가장 낮은 층수부터 주차되고 시리얼 모니터 아래와 같은 주차 현황이 출력된다. (X: 주차 불가능, O: 주차 가능)

```
<시리얼 모니터>
[주차 현황]
6층: O  5층: O  4층: O
3층: O  2층: O  1층: X(3739)
3739 차량 1층 주차 완료
```

라. [출고 버튼]을 누르면 출고할 차량의 번호판을 시리얼 모니터로 입력 받고, 일정 시간 후 출고 완료 안내 및 주차 현황을 출력한다.

마. 차량 출고가 완료되면 시리얼 모니터에 주차료를 안내하도록 한다.

- **(예시)** 주차료는 대회 채점 시간을 고려하여 10초당 1,000원 등의 방법으로 변형할 수 있음.

```
<시리얼 모니터>
[주차료]
1. 차량번호: 3739
2. 주차시간: 10분
3. 주 차 료: 1,000원(10분당  1,000원)
```

2023. 중등부 지역(청주외) 주차타워

◈ 창의적 문제 해결

가. 필수해결 요소의 (가~마) 항을 필히 포함하고 그 외 필수부품 내에서 창의적으로 작품을 완성하고 문제를 해결한다.

예1) 세그먼트를 이용하여 현재 주차된 층수를 표시한다.

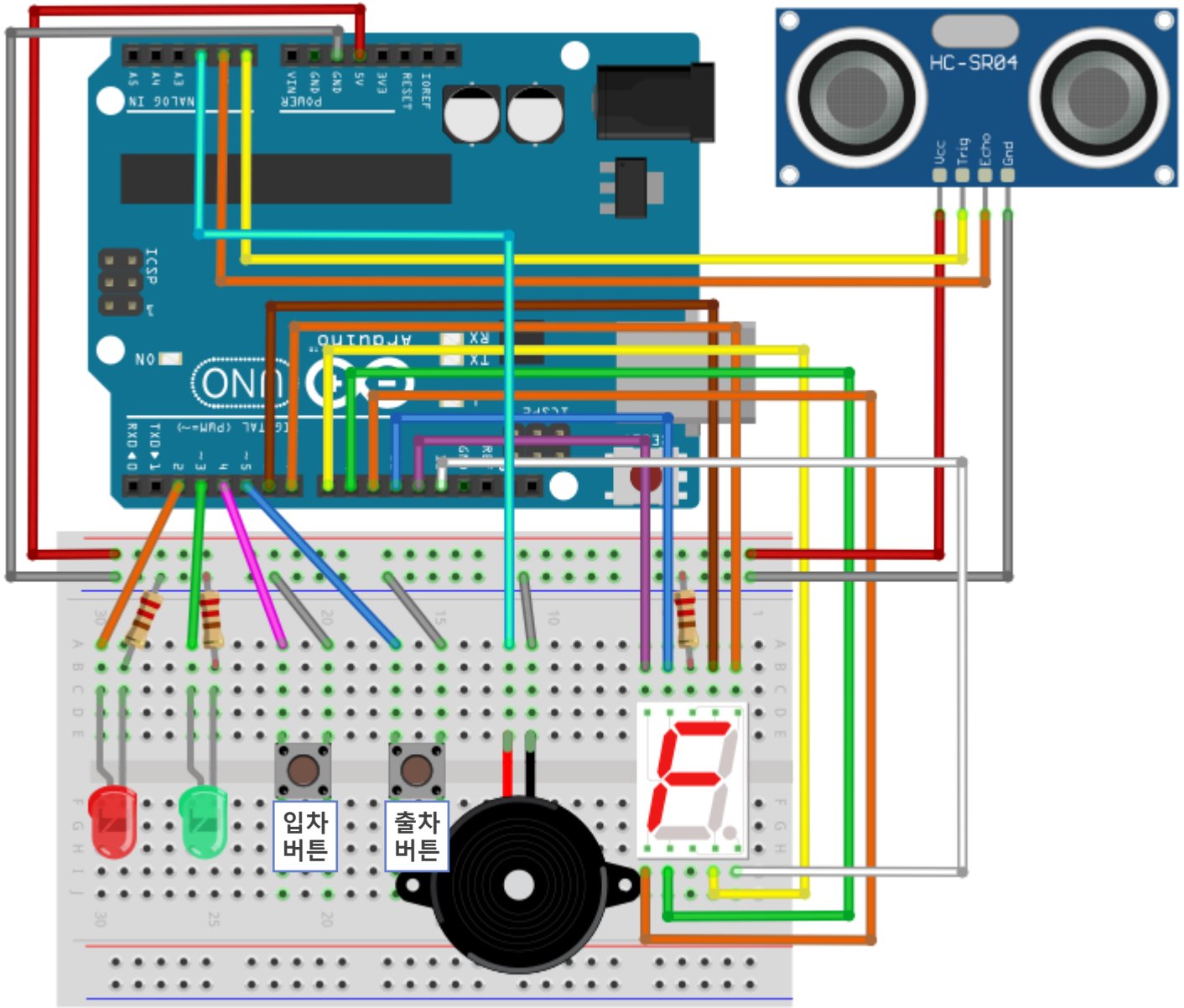
예2) 시리얼 모니터로 주차된 차량의 주차 시간과 주차료를 실시간으로 확인할 수 있다.

예3) 건물 이용객들은 주차 쿠폰을 시스템에 인증하면 주차료 할인 혜택을 받을 수 있다.

예4) 차량 번호판을 잘못 입력한 경우 오류 메세지를 출력하여 안내한다.

2023. 중등부 지역(청주외) 주차타워

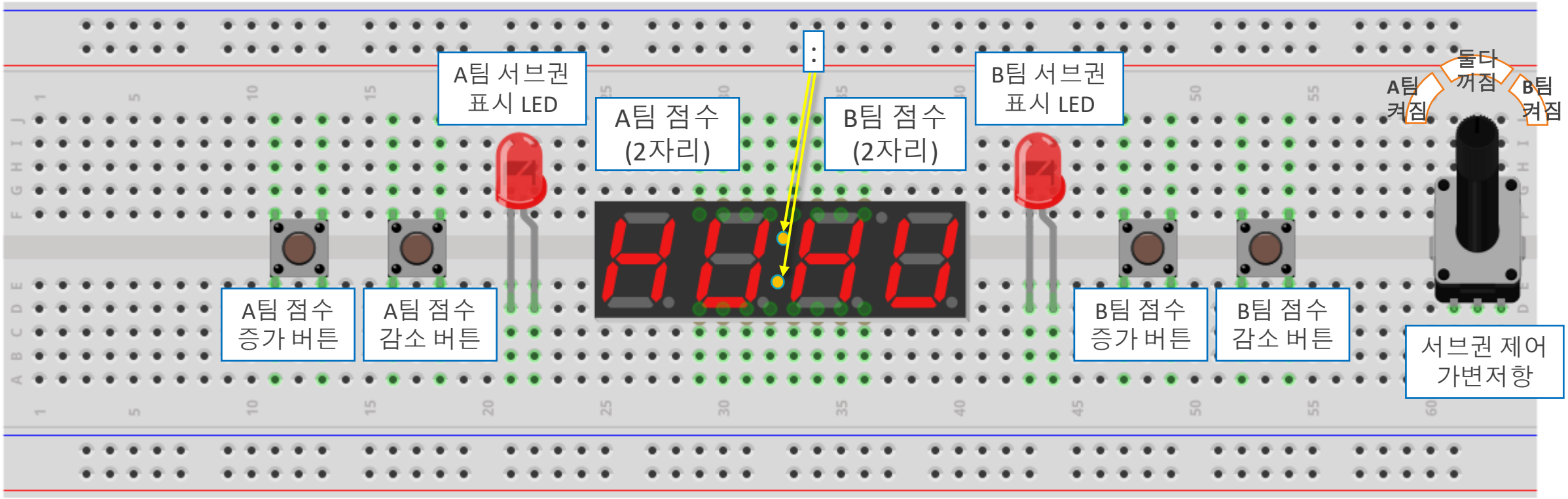
- 결선




```
#define USW_TRIG   A0
#define USW_ECHO   A1
#define BUZZER     A2
#define LED_RED     2
#define LED_GREEN   3
#define BTN_ENTER   4
#define BTN_EXIT    5

#define FND_A       6
#define FND_B       7
#define FND_C       8
#define FND_D       9
#define FND_E      10
#define FND_F      11
#define FND_G      12
#define FND_DP     13
```

2023. 중등부 도대회 - 득점전광판

- 회로 구성 예시



전원이 들어오면 현재 점수 ' 0:0 '을 4자리 FND(7-segmented LED)에 출력한다. A팀의 점수는 오른쪽 정렬하여 표시하고 B팀의 점수는 왼쪽 정렬하여 표시하여야 한다. 따라서 양측 모두 점수가 한 자리 일 때는 ' 0:0 ' 로 표시하고, 두 자리 숫자가 되면 '00:00' 형태로 표시한다. (점수와 점수 사이에 : 표시되어야 함)

2023. 중등부 도대회 득점전광판

◈ 필수해결 요소

가. A팀의 점수와 B팀의 점수가 FND에 정상 표시되어야 한다(FND 제어방법을 모르는 경우 시리얼 모니터에 출력으로 대체, 감점 있음).

나. A팀의 점수는 오른쪽 정렬, B팀 점수는 왼쪽 정렬되어 표시되어야 한다.

다. 점수 증가 버튼을 누르면 해당 팀의 점수가 1 증가 하여야 한다.

라. 점수 감소 버튼을 누르면 해당 팀의 점수가 1 감소 하여야 한다.

마. 증가/감소 버튼을 동시에 눌렀을 때 해당 팀의 점수가 0점으로 초기화되어야 한다.

바. 가변저항을 조정하면 그에 따라 서브권을 가진 팀을 알려주는 LED의 출력 상태가 적절하게 변경되어야 한다.

2023. 중등부 도대회 득점전광판

◈ 창의적 문제 해결

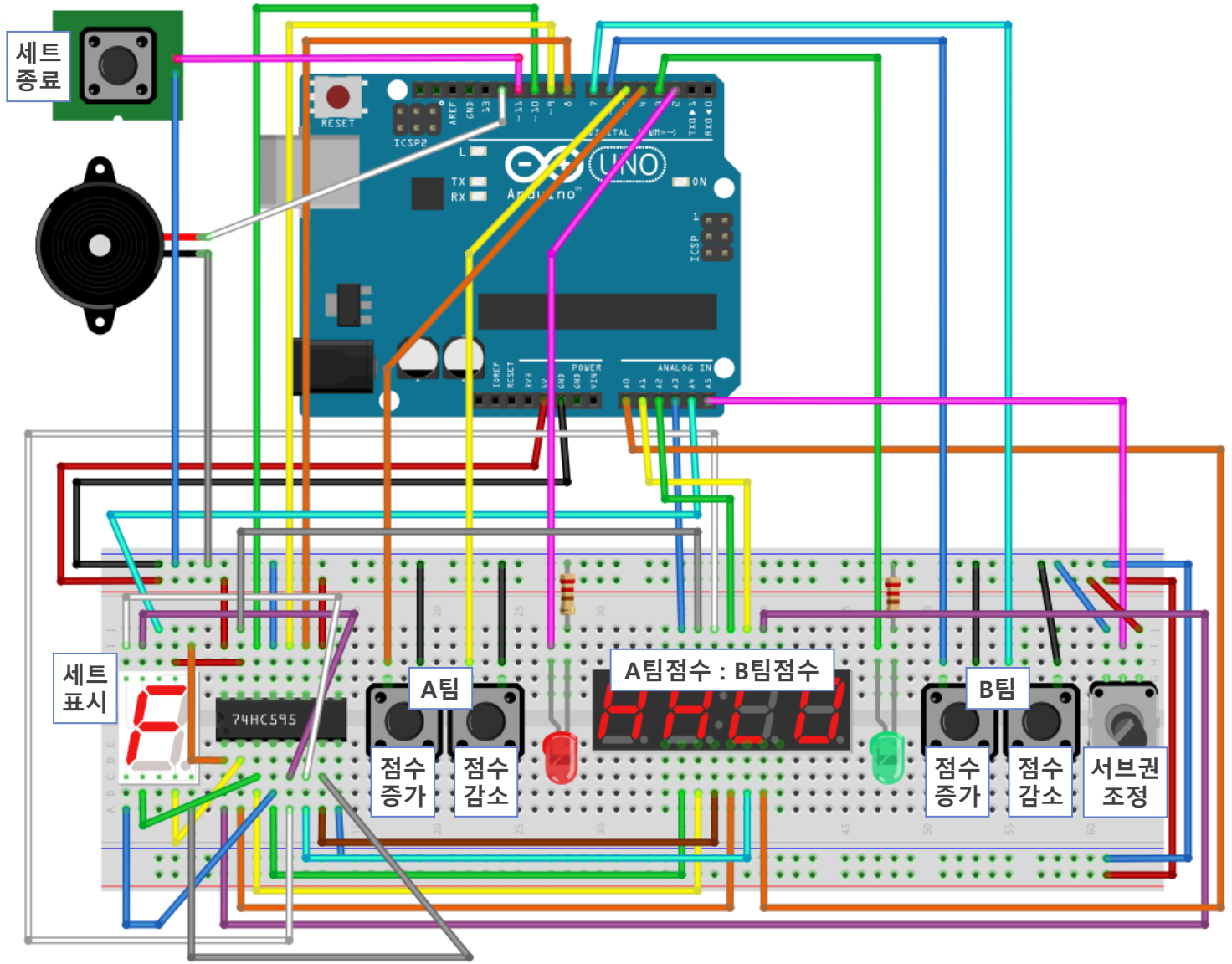
가. 세트와 관련한 기능을 창의적으로 추가하여 작품을 완성한다.

예1) 현재 세트를 표시함. (1자리 FND에 번호로, 시리얼 창에 숫자로, LED에 불켜기 등 다양한 방법 사용 가능)
예2) 세트 종료 버튼(버튼 추가)을 눌러서 현재 세트 스코어를 저장하고 새로운 세트를 0:0부터 시작함.
예3) 지금까지 진행된 세트별 스코어를 시리얼 포트에 출력함.
예4) 총 세트 스코어를 출력하고 어느 팀이 승리하고 있는지 표시함.

나. 세트와 관련 없는 기능은 점수로 인정되지 않는다.

2023. 중등부 도대회 득점전광판

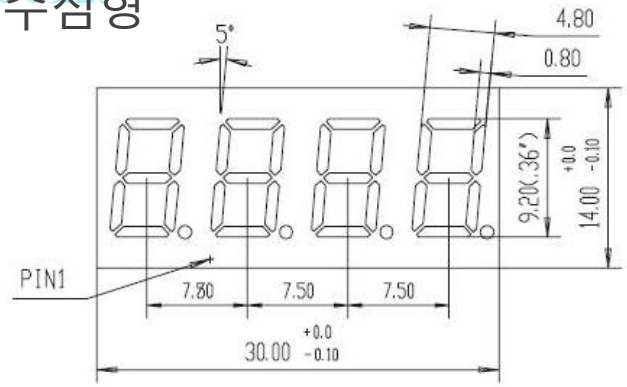
- 결선

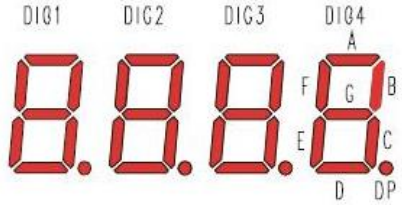


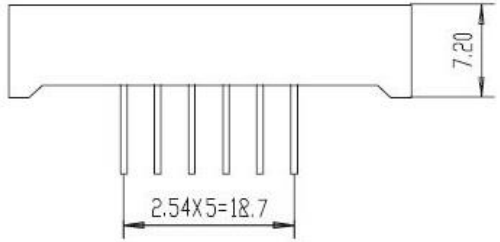

```
#define LED_SRV_A     2
#define LED_SRV_B     3
#define BTN_A_INC     4
#define BTN_A_DEC     5
#define BTN_B_INC     6
#define BTN_B_DEC     7
#define FND_CLOCK     8
#define FND_LATCH     9
#define FND_DATA     10
#define BTN_END_SET  11
#define BUZZER       12
#define POT_SRV      A5
```

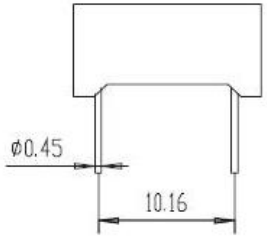
4-Digit FND

- 소수점형

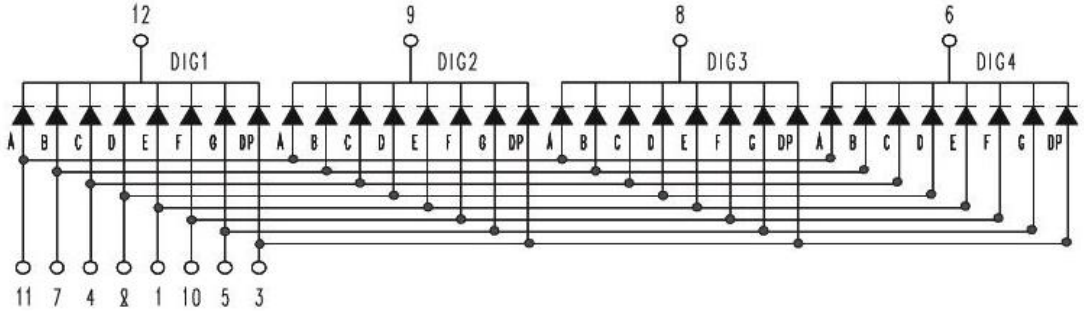




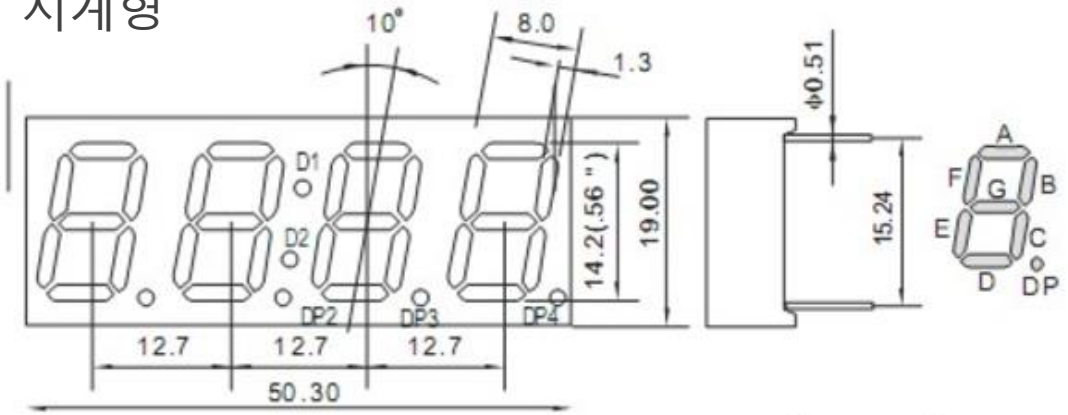


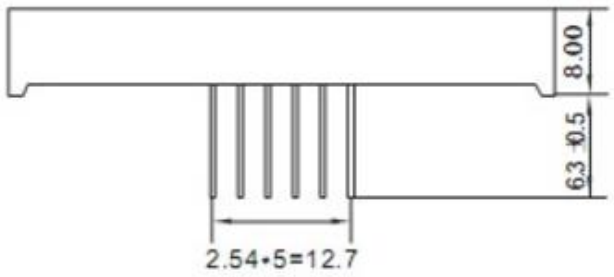


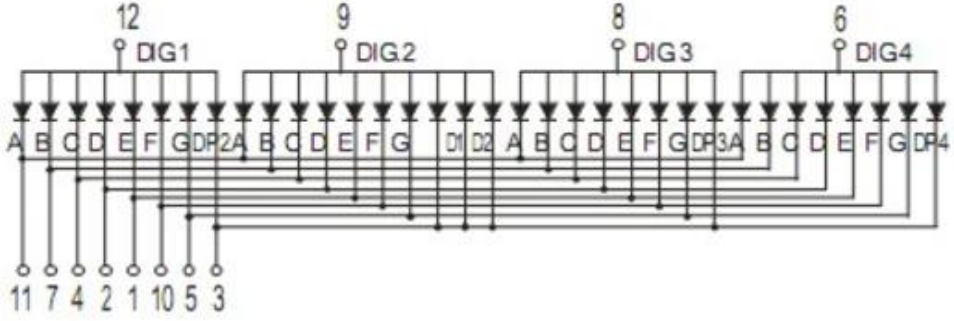
HS-3461A



- 시계형

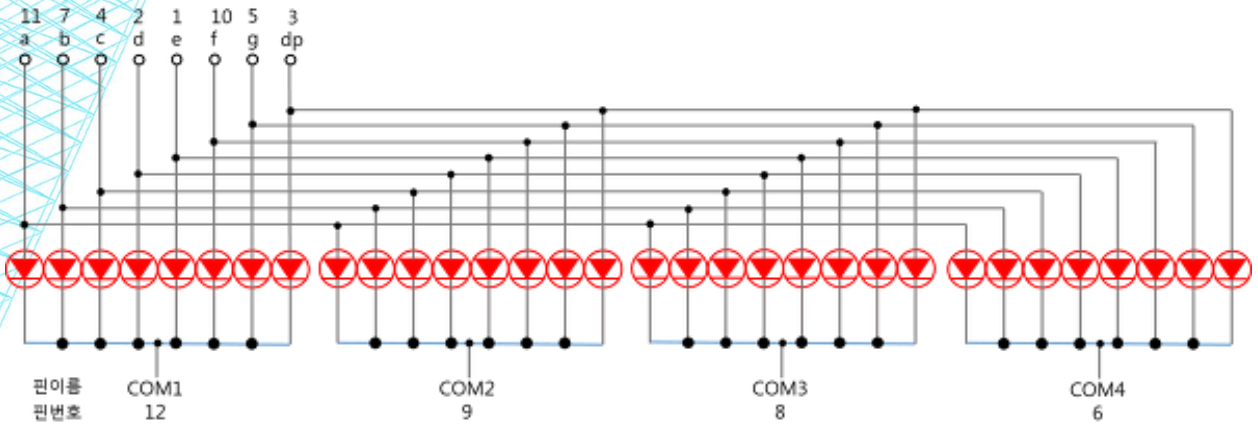




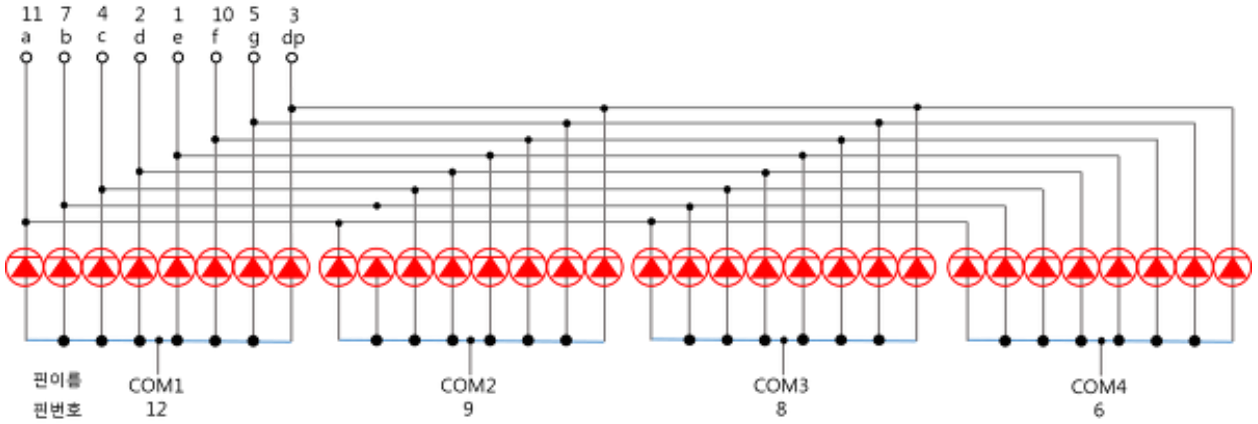


4-Digit FND

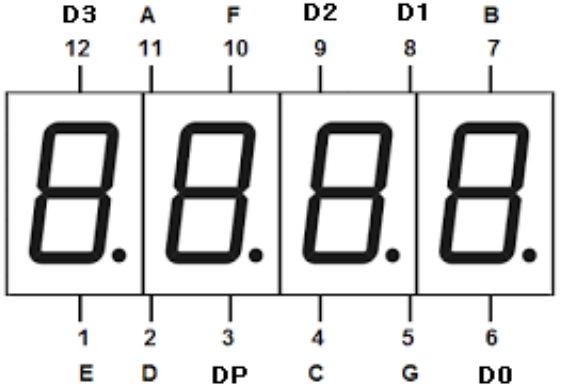
- 공통 음극(common cathode)

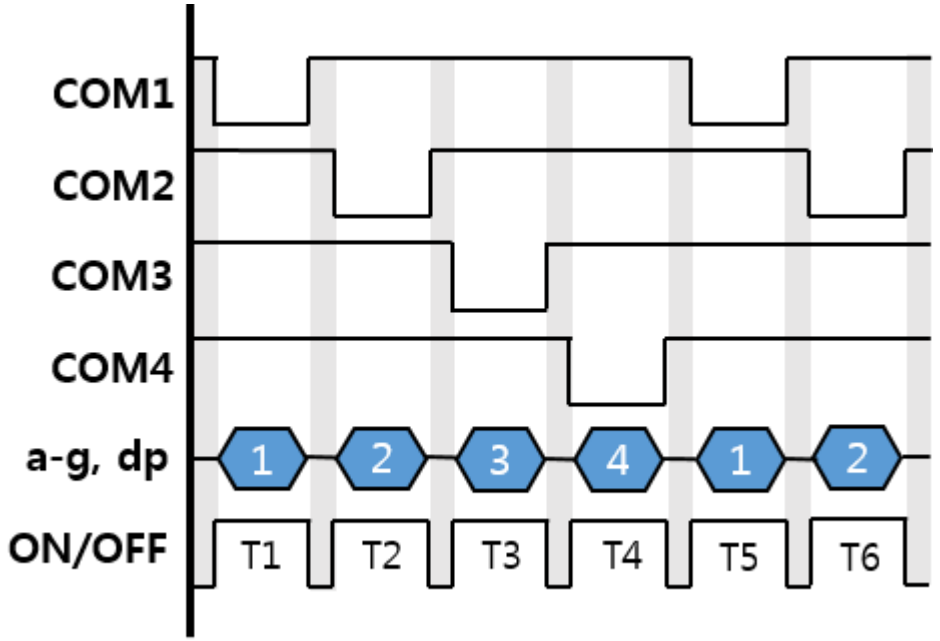


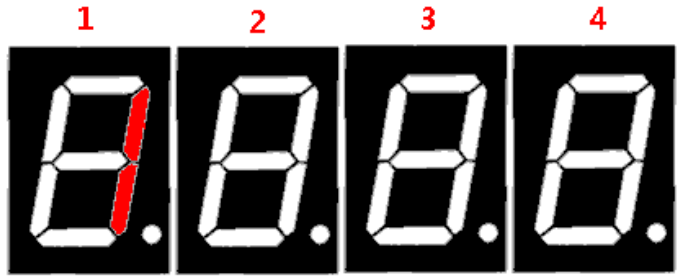
- 공통 양극(common anode)



- 제어 요령

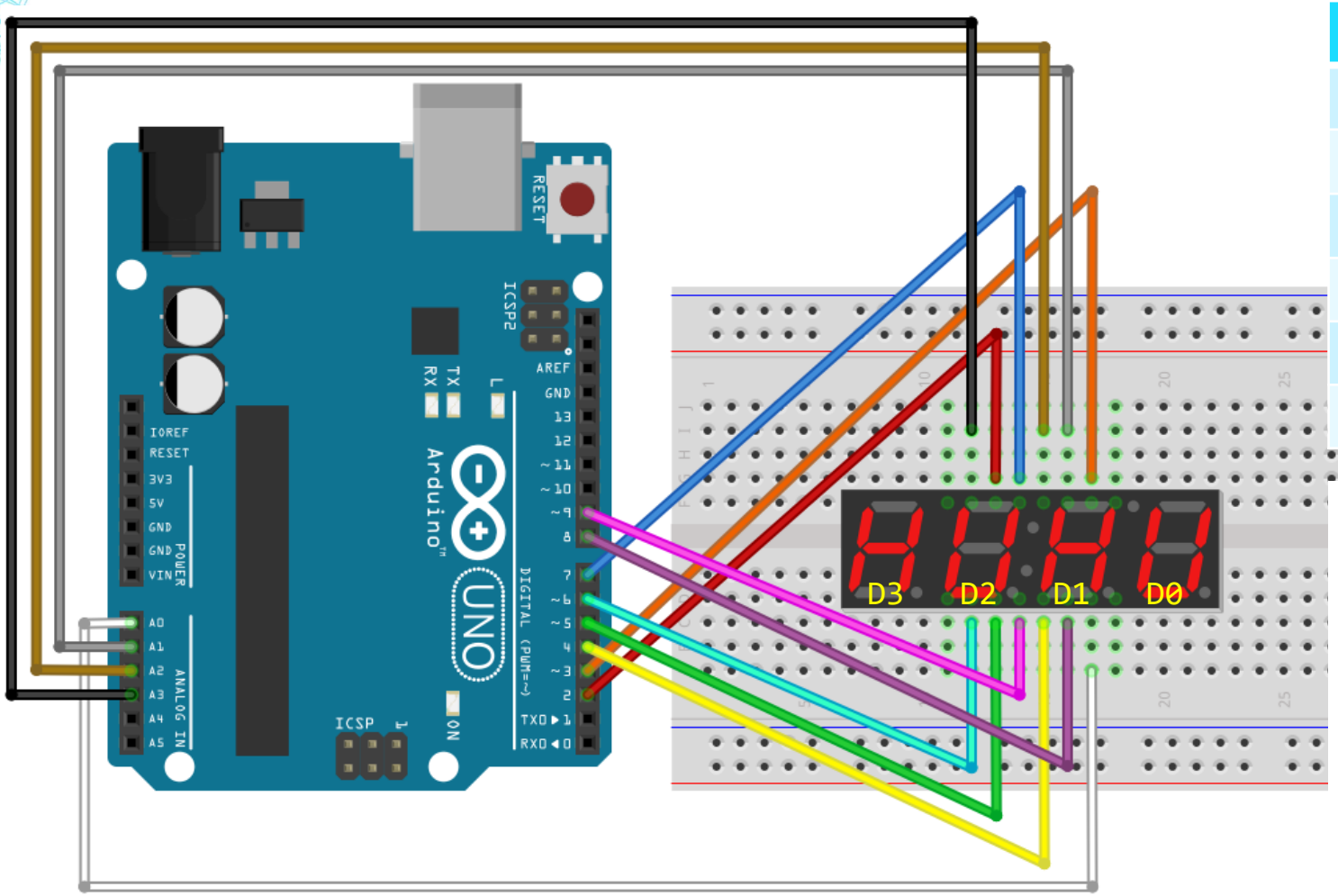






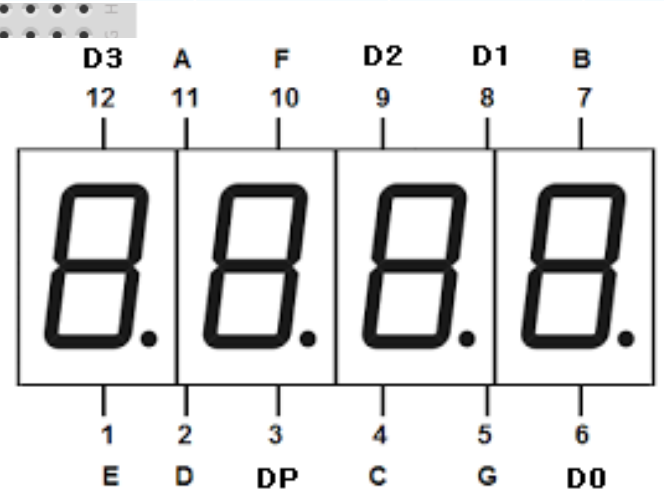
4-Digit FND

- 회로도



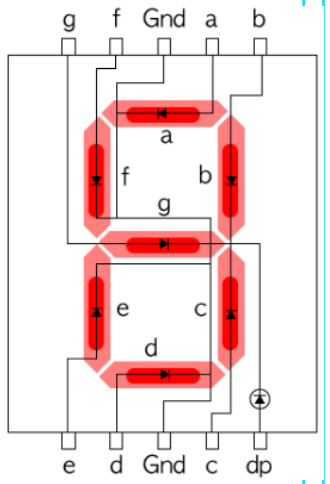
- 결선

FND	아두이노	FND	아두이노
A	2	G	8
B	3	DP	9
C	4	D0	A0
D	5	D1	A1
E	6	D2	A2
F	7	D3	A3

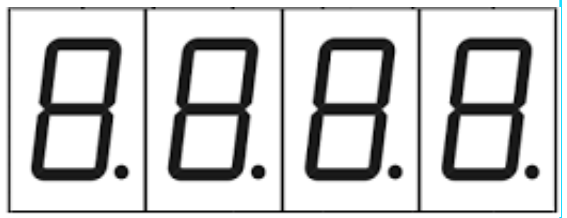


fnd4_output_cc.ino

```
// 4-digit FND for common cathode example
#define FND_D0 A0
#define FND_D1 A1
#define FND_D2 A2
#define FND_D3 A3

int FND_NUMBER;   // 출력할 숫자
int FND_PNTPOS=1; // 점을 찍을 위치 3210

void setup() {
    int i = 0;
    for(i=FND_A; i<=FND_DP; i++)
        pinMode(i, OUTPUT);
    for(i=FND_D0; i<=FND_D3; i++) {
        pinMode(i, OUTPUT);
        digitalWrite(i, HIGH);   // 모든 FND를 OFF
    }
}

unsigned long preTime = 0;
void loop() {
    unsigned long nowTime = millis()/10;
    static int count = 0;
    if(preTime != nowTime) {
      count++;
      preTime = nowTime;
    }
    outputNumber(count, FND_D2);
}
```

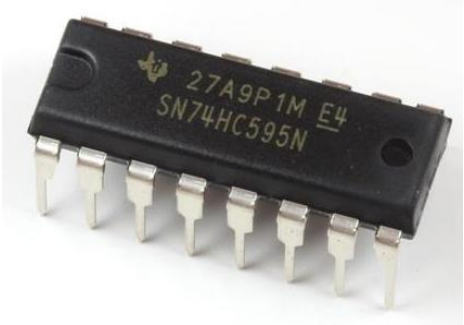


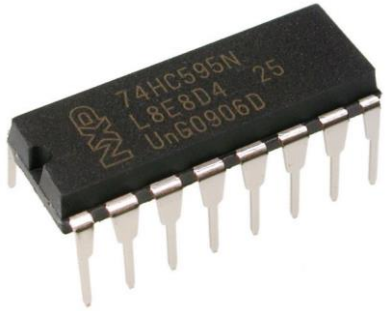

```
// outputFND()는 이전과 동일
void outputFND(int num, bool dot) {
    byte maskbit = B00000001;
    byte font = FND_FONT[num] | (dot<<7);

    for(int i=FND_A; i<=FND_DP; i++) {
        if((maskbit & font) != 0)
            digitalWrite(i, HIGH);
        else
            digitalWrite(i, LOW);

        maskbit = maskbit << 1;
    }
}

void outputNumber() {
    int number = FND_NUMBER;
    for(int p=FND_D0; p<=FND_D3; p++) {
        int digit = number % 10;
        outputFND(digit, p == FND_PNTPOS);
        digitalWrite(p, LOW);      // p번 FND ON
        delayMicroseconds(200);    // 0.2ms
        digitalWrite(p, HIGH);     // p번 FND OFF

        number = number /10;
        if(number <= 0)
           break;
    }
}
```

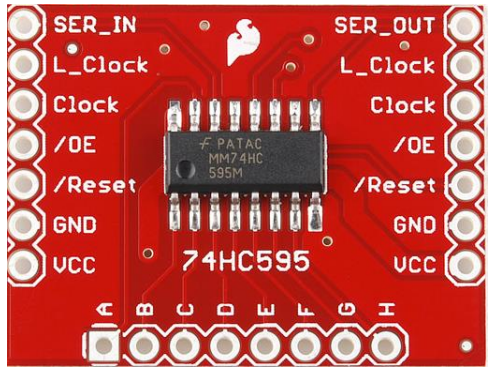


74HC595 Shift Register

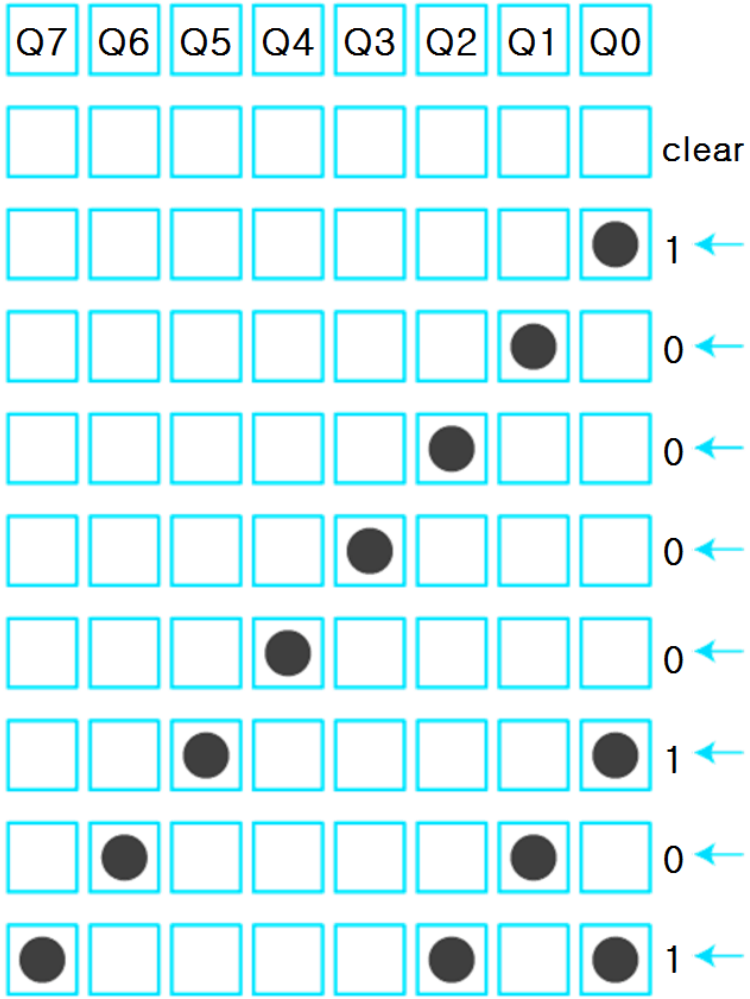
- 칩 외관





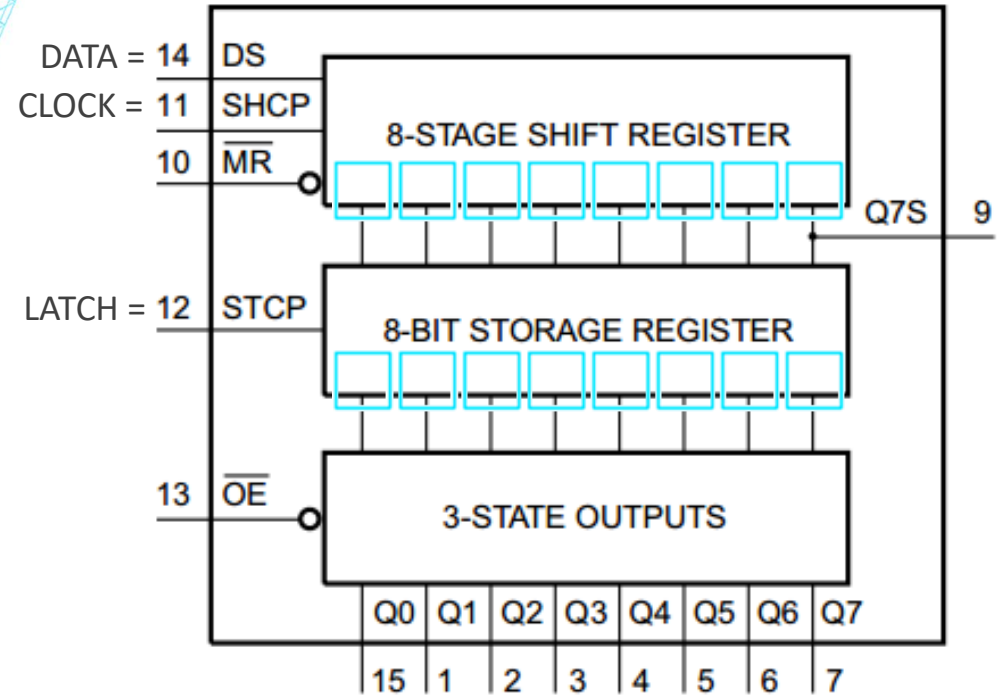


- Shift 동작의 이해



74HC595 Shift Register

Functional diagram

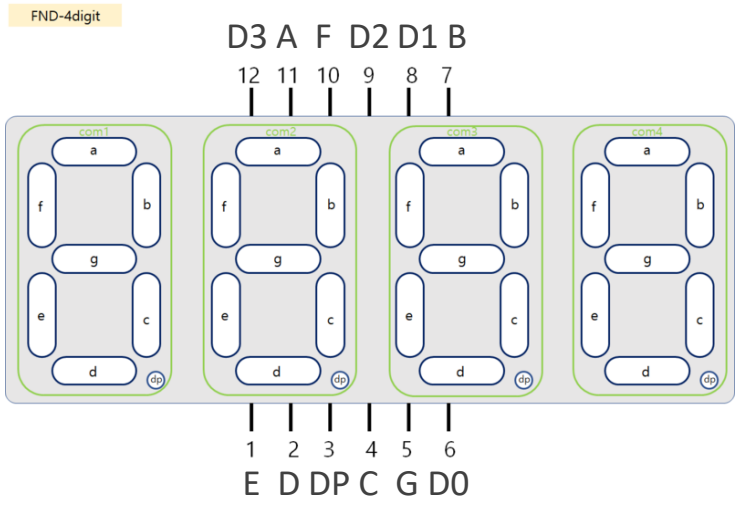


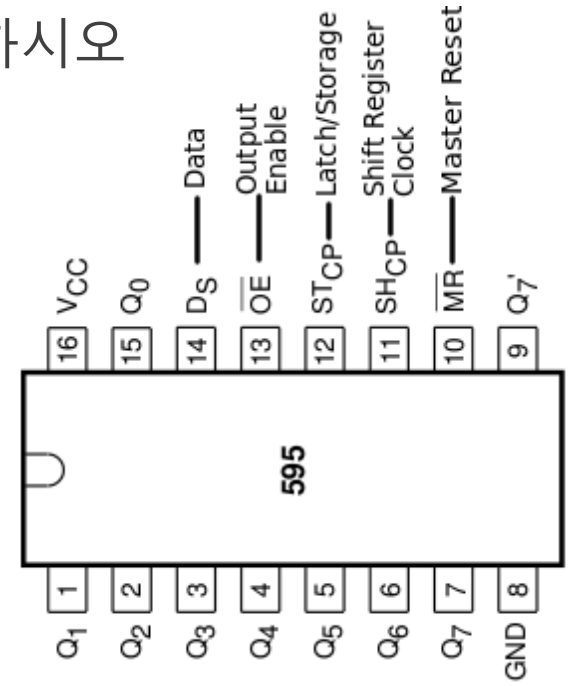
The 74HC595 is an 8-bit serial-in / parallel-out shift register with a storage register and 3-state outputs. Both the shift and storage register have separate clocks. The device features a serial input (DS) and a serial output (Q7S) to enable cascading and an asynchronous reset MR input. **A LOW on MR will reset the shift register. Data is shifted on the LOW-to-HIGH transitions of the SHCP(CLOCK) input. The data in the shift register is transferred to the storage register on a LOW-to-HIGH transition of the STCP input.** If both clocks are connected together, the shift register will always be one clock pulse ahead of the storage register. **Data in the storage register appears at the output whenever the output enable input (OE) is LOW.** A HIGH on OE causes the outputs to assume a high-impedance OFF-state. Operation of the OE input does not affect the state of the registers. Inputs include clamp diodes. This enables the use of current limiting resistors to interface inputs to voltages in excess of VCC.

data sheet: http://assets.nexperia.com/documents/data-sheet/74HC_HCT595.pdf

4-digit FND using 74HC595

- 실습과제
  - shift register를 이용하여 4-digit 7-segment led를 구동하는 회로를 제작하고,
  - 프로그램을 작성하시오





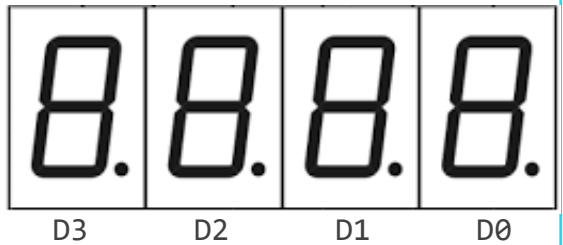
Pin	Symbol	연결
1	Q1	FND_B
2	Q2	FND_C
3	Q3	FND_D
4	Q4	FND_E
5	Q5	FND_F
6	Q6	FND_G
7	Q7	FND_DP
8	GND	GND
9	Q7`	다음 DS에 연결 (지금은 사용 안함)
10	$\overline{MR}$	LOW이면 리셋되어 버리므로 5V에 연결
11	SH_CP	CLOCK
12	ST_CP	LATCH
13	$\overline{OE}$	LOW 일 때 enable 되므로 GND에 연결
14	DS	DATA
15	Q0	FND_A
16	Vcc	5V

fnd4_sreg_cc.ino

fnd4_sreg_ca.ino

```
// 4digit FND using 74HC595 example
#define FND_D0 A0
#define FND_D1 A1
#define FND_D2 A2
#define FND_D3 A3
#define CLOCK  10
#define LATCH  11
#define DATA   12
//#define COMMON_CATHODE
#ifdef COMMON_CATHODE
const int D_ON = LOW, D_OFF = HIGH;
#else
const int D_ON = HIGH, D_OFF = LOW;
#endif

byte FND_FONT[] = {0x3F, 0x06, 0x5B, 0x4F, 0x66,
   0x6D, 0x7D, 0x07, 0x7F, 0x67 };

unsigned long preTime = 0;

void setup() {
    for(int p=FND_D0; p<=FND_D3; p++) {
        pinMode(p, OUTPUT);
        digitalWrite(p, D_OFF); //@
    }
    pinMode(CLOCK, OUTPUT);
    pinMode(LATCH, OUTPUT);
    pinMode(DATA,  OUTPUT);
}
```

```
void outputFND(int num, bool dot) {
    byte font = (FND_FONT[num] | (dot<<7));
    if(D_ON==HIGH) font = ~font;
    digitalWrite(LATCH, LOW);
    shiftOut(DATA, CLOCK, MSBFIRST, font);  // MSB부터 밀어넣기
    digitalWrite(LATCH, HIGH); // 출력에 적용
}

void outputNumber(int number, int dotpos) {
    for(int p=FND_D0; p<=FND_D3; p++) {
        int digit = number % 10;
        outputFND(digit, p == dotpos);
        digitalWrite(p, D_ON);     // p FND ON
        delayMicroseconds(200);    // 0.2ms
        digitalWrite(p, D_OFF);    // p FND OFF
        number = number /10;
        if(number <= 0)
           break;
    }
}
void loop() {
    unsigned long nowTime = millis()/10;
    static int count = 0;
    if(preTime != nowTime) {
      count++;
      preTime = nowTime;
    }
    outputNumber(count, FND_D2);
}
```



```
// Shift Register 미사용시

void outputFND(int num, bool dot) {
    int maskbit = 0x01;
    unsigned char font = FND_FONT[num] | (dot<<7);

    for(int i=FND_A; i<=FND_DP; i++) {
        if((maskbit & font) != 0)
            digitalWrite(i, HIGH);
        else
            digitalWrite(i, LOW);
        maskbit = maskbit << 1;
    }
}

void outputNumber(int number, int dotpos) {
    for(int p=FND_D0; p<=FND_D3; p++) {
        int digit = number % 10;
        outputFND(digit, p == dotpos);
        digitalWrite(p, LOW);      // p FND ON
        delayMicroseconds(1000);   // 1ms
        digitalWrite(p, HIGH);     // p FND OFF

        number = number /10;
        if(number <= 0)
           break;
    }
}
```

```
// Shift Register 사용시

void outputFND(int num, bool dot) {
    byte font = FND_FONT[num] | (dot<<7);
    digitalWrite(LATCH, LOW);
    // MSB부터 데이터 밀어넣기
    myShiftOut(DATA, CLOCK, MSBFIRST, font);
    digitalWrite(LATCH, HIGH); // 출력에 적용
}

void outputNumber(int number, int dotpos) {
    for(int p=FND_D0; p<=FND_D3; p++) {
        int digit = number % 10;
        outputFND(digit, p == dotpos);
        digitalWrite(p, LOW);      // p FND ON
        delayMicroseconds(1000);   // 1ms
        digitalWrite(p, HIGH);     // p FND OFF

        number = number /10;
        if(number <= 0)
           break;
    }
}
```

FND 클래스 설계

FND.h

```
#include <stdarg.h>
#include <ctype.h>

class FND {
  private:
    int _pinsLed[8];
    int _pinsDx[4];
    int _pinClock, _pinLatch, _pinData;
    int _outNumber;  // 출력할 숫자
    char _outBuffer[10];  // 출력할 내용
    int _posPoint;     // 소수점 위치
    int _numDigits;    // 자리수
    bool _enabled;
    bool _shiftRegMode;  // 시프트레지스터 사용여부
    bool ON;   // commonCathod이면 HIGH로 설정하시오.
    bool OFF;
    bool cmmON, cmmOFF;
    static unsigned char FND_FONT[];

  public:
    FND(bool levelON = HIGH) {
      _outNumber = 0;
      _posPoint = -1;  // -1: 소수점 숨김, 1: 소수1자리
      _numDigits = 1;  // 자리수
      _enabled = true;
      setLevelON(levelON);
    }
```

```
    void setLevelON(bool level) {
      ON = level;
      OFF = !ON;
      cmmON = OFF, cmmOFF = ON;
    }
    void allocLedPins(int a, int b, int c, int d,
                      int e, int f, int g, int dp) {
      pinsLed[0]=a, pinsLed[1]=b;
      pinsLed[2]=c, pinsLed[3]=d;
      pinsLed[4]=e, pinsLed[5]=f;
      pinsLed[6]=g, pinsLed[7]=dp;

      for(int i=0; i<8; i++) {
        pinMode(pinsLed[i], OUTPUT);
        digitalWrite(pinsLed[i], OFF);
      }
      shiftRegMode = false;
    }

    void allocRegisterPins(int clock, int latch, int data) {
      pinMode(clock, OUTPUT);
      pinMode(latch, OUTPUT);
      pinMode(data, OUTPUT);
      pinClock = clock;
      pinLatch = latch;
      pinData  = data;
      shiftRegMode = true;
    }
```

FND 클래스 설계 cont.

```
void allocDxPins(int cnt, ...) {
  va_list ap;
  va_start(ap, cnt);

  for(int i=0; i<cnt; i++) {
    pinsDx[i] = va_arg(ap, int);
    pinMode(pinsDx[i], OUTPUT);
    digitalWrite(pinsDx[i], cmmOFF); // <=
  }
  va_end(ap);
  numDigits = cnt;
}

void outputLedPins(int num, bool dot) {
  int maskbit = 1;
  byte font = FND_FONT[num] | (dot<<7);

  for(int i=0; i<8; i++) {
      if((maskbit & font) != 0)
        digitalWrite(pinsLed[i], ON);
      else
        digitalWrite(pinsLed[i], OFF);

      maskbit = maskbit << 1;
  }
}
```

```
void outputShiftRegPins(int num, bool dot=false) {
  byte font = FND_FONT[num] | (dot<<7);
  if(ON == LOW) font = ~font;

  digitalWrite(_pinLatch, LOW);
  shiftOut(_pinData, _pinClock, MSBFIRST, font);
  digitalWrite(_pinLatch, HIGH);
}

void outputFND(int num, bool dot=-1) {
  if(!_enabled) return;

  if(_shiftRegMode)
    outputShiftRegPins(num, dot);
  else
    outputLedPins(num, dot);
}

// pinDx[d] 핀에 전류가 흐르도록 설정함.
void onDx(int d) {
  digitalWrite(pinsDx[d], cmmON);
}

// pinDx[d] 핀에 전류가 흐르지 않도록 설정함.
void offDx(int d) {
  digitalWrite(pinsDx[d], cmmOFF);
}
```

FND 클래스 설계 cont.

```
int getCharFontIndex(char c) {
  int pos;
  if(isdigit(c))
    pos = c - '0';
  else if('A'<=c && c<='Z' || 'a'<=c && c<='z')
    pos = toupper(c) - 'A' + 10;
  else if(c == '-')
    pos = 36; // 마이너스
  else //출력할 내용이 숫자도 문자도 아니면,
    pos = 37; // 마지막 공백 문자
  return pos;
}

void outputFND(char ch, bool dot=false) {
  if(!_enabled) return;

  int font_idx = getCharFontIndex(ch);
  if(_shiftRegMode)
    outputShiftRegPins(font_idx, dot);
  else
    outputLedPins(font_idx, dot);
}

void outputNumber(int number, int pos=-1) {
  _outNumber = number;
  itoa(number, _outBuffer, 10);
  outputString(pos);
}
```

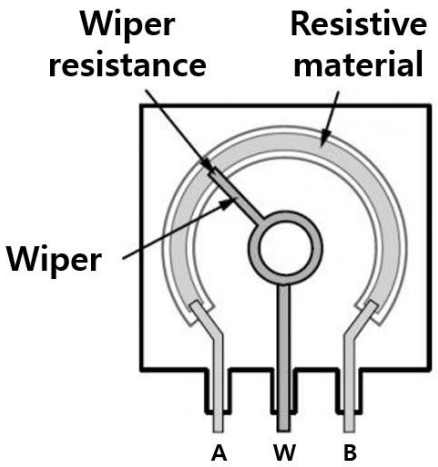
```
void outputCharAt(int pos, char font_idx, bool dot=false) {
  for(int p=0; p<_numDigits; p++)
    digitalWrite(_pinsDx[p], cmmOFF);    // p FND OFF

  if(_shiftRegMode)
    outputShiftRegPins(font_idx, dot);
  else
    outputLedPins(font_idx, dot);

  digitalWrite(_pinsDx[pos], cmmON);     // d FND ON
}

//문자열의 가장 오른쪽부터 _numDigits개 출력
void outputString(char* strOutput, int dotpos=-1) {
  if(!_enabled) return;

  int idx = strlen(strOutput)-1;  //문자열의 마지막
  for(int d=0; d<_numDigits; d++, idx--) {
    if(idx < 0) return;
    outputFND(strOutput[idx], d==dotpos);
    digitalWrite(_pinsDx[d], cmmON);   // d FND ON
    delayMicroseconds(200);            // 0.2ms
    digitalWrite(_pinsDx[d], cmmOFF);  // d FND OFF
  }
}
```

FND 클래스 설계 cont.

```
    void outputString(int dotpos=-1) {
      outputString(_outBuffer, dotpos);
    }

    void enable()    { _enabled = true;  }
    void disable()   {
      _enabled = false;
      for(int d=0; d<min(4,_numDigits); d++)
        digitalWrite(_pinsDx[d], cmmOFF);
    }
    bool enabled()   { return _enabled;  }
};

// Active HIGH 일 때 폰트임.
// FND가 common annode 인 경우 아래 데이터를 비트 NOT하여 사용할 것.
unsigned char FND::FND_FONT[] =
   // 0,    1,    2,    3,    4,    5,    6,    7,    8,    9,
  {0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x67,
   // A,    B,    C,    D,    E,    F,    G,    H,    I,    J,
   0x77, 0x7C, 0x39, 0x5E, 0x79, 0x71, 0x3D, 0x76, 0x30, 0x1E,
   // K,    L,    M,    N,    O,    P,    Q,    R,    S,    T,
   0x7A, 0x38, 0x55, 0x54, 0x5C, 0x73, 0x67, 0x50, 0x6D, 0x78,
   // U,    V,    W,    X,    Y,    Z,   - ,  공백
   0x3E, 0x7E, 0x6A, 0x36, 0x6E, 0x49, 0x40, 0x00 };
```

가변저항(potentiometer) 연결하기

- 회로구성




https://en.wikipedia.org/wiki/Potentiometer

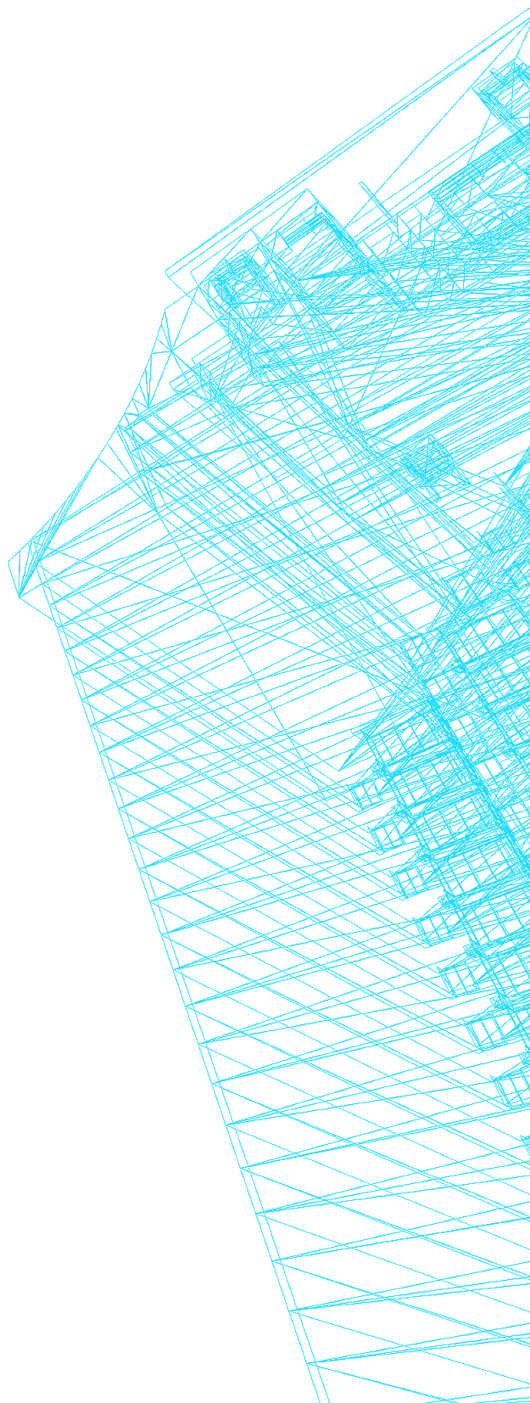
- 소스코드

3-1.ino

```
void setup() {
  Serial.begin(9600);
  Serial.println("a, V");
}

// A0에 연결된 선을 각 탐측지점에
// 꽂아 보면서 실험한다.
void loop() {
  int a = analogRead(A0);
  // 0~1023 범위의 값을 0~5범위로.
  double v = a/1023.0 * 5;

  Serial.print(a);
  Serial.print(", ");
  Serial.print(v);
  Serial.println("V");
  delay(200);
}
```

그밖의 기출문제 풀이

- 2018 도대회 (고) 알람시계
- 2019 도대회 (중) 추억의 뽑기 게임
- 2022 도대회 (고) 3·6·9게임 연습기계



2018 도대회 (고) 알람시계

철수는 시계의 기능을 아두이노로 구현하려고 한다. 철수가 한 자리에 앉아서 얼마나 집중했는지 시간을 측정하려고 한다. 이를 위해 앉았을 때 버튼을 누르면, 시간이 흐르기 시작하고, 자리에서 일어나면서 버튼을 누르면 시간이 멈추고 저장된다. 이런 과정을 최대 10회까지 저장하려고 한다.

◈ 필수해결 요소

① 버튼을 설치하고, 버튼을 누르면 시간이 0에서부터 시작된다.
② LCD에는 버튼을 누른 시점부터 시간의 변화를 초단위로 출력한다.
③ 다시 버튼을 눌렀을 때, 시간이 멈추고 현재 공부한 시간이 출력되며, 현재까지의 공부한 시간 중 몇 번째로 많은지 출력하게 한다.
④ 최대 10개의 시간이 저장되게 만든다.
⑤ 종료 버튼을 추가하고 종료를 한 경우에는 현재까지의 공부한 시간 중 최저 시간과 최고 시간이 출력되며, 평균의 시간을 출력한다.
⑥ 다시 한번 종료 버튼을 누르면, 현재까지 공부한 시간 중 긴 시간부터 차례대로 출력하도록 만든다.

2018 도대회 (고) 알람시계

2018-DO-HS%20StopWatch.ino

```
#include "IntTimer1.h"
#include "Button.h"
#include "FND.h"

#define  STUDY_NUM_MAX  10
#define  BUZ  8

FND fnd4(HIGH);  // common cathode
//FND fnd4(LOW);  // common anode

// start/stop toggle button
Button btnToggle(INPUT_PULLUP, 2, 40);
Button btnStore(INPUT_PULLUP, 3, 40);

unsigned long TIME;
int tmrIdTimeTick;
unsigned int studyTime[STUDY_NUM_MAX];
unsigned int studyNum = 0;  // 공부시간 저장횟수
unsigned int studyTimeRank[STUDY_NUM_MAX];
```

```
void setup() {
  Serial.begin(9600);

  fnd4.allocDxPins(4,A0,A1,A2,A3);
  fnd4.allocRegisterPins(10,11,12);
  tmrIdTimeTick = IntTimer1::add(timeTick, 100);
  IntTimer1::add(checkButton, 1);
  IntTimer1::start();
}

void timeTick() { // 0.1초 마다 실행됨
  TIME++;          // 시간변수 증가
  Serial.println(TIME/10.0); // _._ 형태로 출력
}

void checkButton() {
  btnToggle.update();
  btnStore.update();
}
```

2018 도대회 (고) 알람시계

```
void loop() {
  fnd4.outputNumber(TIME, 1);

  if(btnToggle.fell()) {  // 버튼이 눌리는 순간
    if(IntTimer1::enabled(tmrIdTimeTick))
      IntTimer1::disable(tmrIdTimeTick);
    else
      IntTimer1::enable(tmrIdTimeTick);

    tone(BUZ, 880, 100);
    delay(100);
    tone(BUZ, 440, 100);
  }

  if(btnStore.fell()) {
    // 시간이 멈춰 있을 때만 작동하도록...
    if(!IntTimer1::enabled(tmrIdTimeTick)) {
      studyTime[studyNum] = TIME;
      outputRank();
      studyNum++;
      if(studyNum >= STUDY_NUM_MAX)
        studyNum = 0;

      tone(BUZ, 440, 100);
      delay(100);
      tone(BUZ, 880, 100);
      TIME = 0;
    }
  }
}

void filloutRank() {
  for(int i=0; i<STUDY_NUM_MAX; i++)
    studyTimeRank[i]=1; // 일단 1등이라고 가정

  for(int i=0; i<STUDY_NUM_MAX; i++) { // i의 순위 결정
    for(int j=0; j<STUDY_NUM_MAX; j++) // 모든 j와 비교하여
      if(studyTime[i]<studyTime[j]) // j가 더 공부시간이 많으면,
        studyTimeRank[i]++; // i의 순위 뒤로 밀림
  }
}

void outputRank() {  // 순위 출력
  filloutRank();
  for(int i=0; i<STUDY_NUM_MAX; i++) {
    Serial.print(i+1);
    Serial.print(": ");
    Serial.println(studyTime[i]/10.0);
  }
  Serial.print(studyTimeRank[studyNum]);
  Serial.println("번째로 공부를 오래 하였습니다.");
}
```

2019 도대회 (중) 추억의 뽑기 게임

◈ 필수해결 요소

① 가위 바위 보 대결 시스템. ex: 시리얼 모니터에 가위, 바위, 보(1,2,3//a,b,c등으로 대체 가능)를 입력하면 승 또는 패를 알려주기(LED, 소리 등을 활용)

② 가위 바위 보 대결에서 이겼을 경우 랜덤하게 1~9까지의 숫자가 나오도록 하기(시리얼 모니터 활용), 졌을 경우 게임 종료

③ 게임이 종료되면 메달 획득 개수를 기준으로 랭킹 나오기(시리얼 모니터 활용)

④ 랭킹이 나오고 나면 다음 사람도 게임에 참여할 수 있도록 처음 상태로 되돌린다.(메달 획득 랭킹은 유지)

◈ 창의적 문제 해결

① 가위, 바위, 보를 입력하는 시스템을 버튼으로 직접 구현한다.

② 1~9까지의 숫자가 FND를 통해서 나오도록 표현한다.

③ 메달의 보상이 9까지가 아니라 20까지 표현되도록 FND를 구성한다.

④ 적절한 상황에 재미를 더할 수 있는 음향 효과를 추가한다.

2019 도대회 (중) 추억의 뽑기 게임

2019-DO-HS%20PickGame.ino

```
#include "MillisTimer.h"
#include "Button.h"
#include "FND.h"
#include "MelodyPlayer.h"

#define BTN_C 2
#define BTN_R 3
#define BTN_P 7
#define BUZ   8
enum SRP { NOTYET=0, SCI=1, ROC=2, PAP=3 };
char *strSRP[]={" ", "가위","바위","보"};

Note noteWin[] = { {523,250}, {659,250},
{784,250} };  // 딩동댕~ (도미솔~)
Note noteLose[] = { {659,750} };  // 땡~ (미~~~~)
Note noteTick[] = { {440, 80} };  // 띠(아주짧음)
MelodyPlayer mPlayer(BUZ);

FND fnd4(LOW);  // common anode mode
Button btnC(INPUT_PULLUP, BTN_C, 40); //가위
Button btnR(INPUT_PULLUP, BTN_R, 40); //바위
Button btnP(INPUT_PULLUP, BTN_P, 40); //보
```

```
float TIME = 0;
void tick() {
  if((int)(TIME*10) % 10==0) // 매 1초마다 시간 출력
    Serial.println((int)TIME);

  TIME += 0.1;
}

void setup() {
  Serial.begin(9600);
  randomSeed(analogRead(0));

  fnd4.allocDxPins(4,A0,A1,A2,A3);
  fnd4.allocRegisterPins(10,11,12);
  MillisTimer::add(tick, 100);
}

char getCRPcode(SRP num) {
  if(num == SCI) return 'C';
  else if(num == ROC) return 'R';
  else if(num == PAP) return 'P';
  else return ' ';
}
```

2019 도대회 (중) 추억의 뽑기 게임

```
void loop() {
  TIME = 0.1;
  int  computerPick = NOTYET;
  int  gamerPick = NOTYET;
  bool isWin = false;  // 승리했는가?
  int  numMedal;

  // 카운트 다운 3.2.1.
  while(TIME <= 3) {
    MillisTimer::run();
    int outNum = (int)(4-TIME)*1111;
    fnd4.outputNumber(outNum);
  }

  // 버튼 눌러 가위바위보 내기
  fnd4.disable();
  computerPick = random(3)+1;
  while(TIME<=4 && gamerPick==NOTYET) {
    MillisTimer::run();
    btnC.update();
    btnR.update();
    btnP.update();

    if(btnC.fell()) gamerPick = SCI;
    if(btnR.fell()) gamerPick = ROC;
    if(btnP.fell()) gamerPick = PAP;
  }
```

```
  // 컴퓨터:사람 승부 출력
  char str[]="    ";
  str[0]=getCRPcode(computerPick);
  str[3]=getCRPcode(gamerPick);
  Serial.print("컴| ");
  Serial.print(strSRP[computerPick]);
  Serial.print(" : 나| ");
  Serial.println(strSRP[gamerPick]);
  fnd4.enable();

  if(gamerPick==SCI && computerPick==PAP ||
     gamerPick==ROC && computerPick==SCI ||
     gamerPick==PAP && computerPick==ROC)
    isWin = true;

  if(isWin) {
      mPlayer.setNote(noteWin, 3);
      mPlayer.start();
  }
  else {
      mPlayer.setNote(noteLose, 1);
      mPlayer.start();
  }
  while(TIME <= 5) {
    MillisTimer::run();
    fnd4.outputString(str, 2);
    mPlayer.play();
  }
```

2019 도대회 (중) 추억의 뽑기 게임

```
  // 메달 보너스 출력
  if(isWin) numMedal = random(20)+1;
  else return; // loop 탈출

  unsigned long startTime = millis();
  int preOutNum=-1;
  while(TIME <= 8) {
    MillisTimer::run();

    // 메달 출력 루틴에 진입한 이후 흐른 시간(0.1초단위)
    int outNum = (millis() - startTime) / 100;
    if(preOutNum!=outNum && outNum<numMedal) {
      mPlayer.setNote(noteTick, 1);
      mPlayer.start();
      preOutNum = outNum;
    }
    fnd4.outputNumber(min(outNum, numMedal));
    mPlayer.play();
  }
}
```

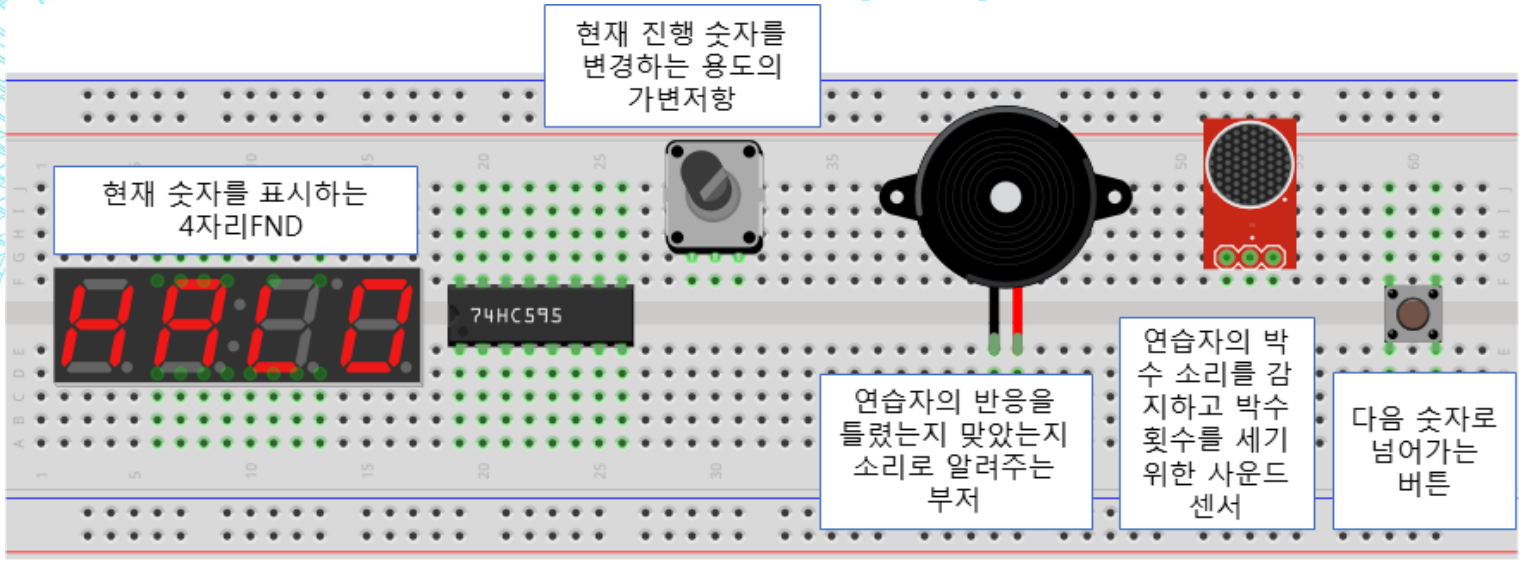
2022 도대회 (고) 3·6·9게임 연습기계

네 자리 FND에 현재 진행 중인 숫자를 표시한다. 그런데 매번 연습할 때마다 1부터 시작하면 재미가 없으니 가변저항 이용하여 진행 중인 숫자를 바꿀 수 있도록 한다. 가변저항을 왼쪽 끝으로 돌리면 1이 되고, 오른쪽 끝으로 돌리면 1000이 되는 식이다. 네 자리 FND에 현재 진행 중인 숫자가 표시되면(또는 가변저항을 돌려서 현재 진행 중인 숫자가 결정되면) 연습자는 1.5초 내에 올바른 반응을 해야 한다.

만약 현재 숫자가 3,6,9가 포함되는 숫자가 아니면 다음 버튼을 눌러야 하고 현재 숫자가 3,6,9가 포함되는 숫자이면 박수로 반응을 해야 하는데 현재 숫자에 포함된 3,6,9 갯수 만큼 박수를 쳐야 한다. (예를 들어 13이면 박수 1회, 36이면 박수 2회)

연습자의 반응에 기계가 응답을 해야 하는데 만약 올바르게 반응하였다면 '딩동~'하는 소리를 출력하고, 틀리게 반응하였다면'땡~'하는 소리를 출력한다. 만약 FND의 숫자가 바뀐지 1.5초가 지나도록 아무런 반응을 하지 않았다면 틀린 것으로 간주한다. 그리고 연습자의 반응에 따른 응답을 출력하였으면 즉시 다음 숫자로 넘어가야 한다.

2022 도대회 (고) 3·6·9게임 연습기계



① 현재 진행 중인 숫자가 네 자리 FND에 깜빡임 없이 자연스럽게 표시되어야 한다.
② 연습자의 박수 소리와 횟수를 올바르게 인식하여야 한다. (박수 횟수를 인식하기 어려우면 박수 소리만 감지하는 방식으로 간략화 가능. 단, 감점 있음)
③ 연습자가 반응하도록 주어지는 시간 타임아웃 1.5초가 올바르게 작동하여야 한다.
④ 연습자의 응답을 판단한 결과를 적절한 소리로 출력하여야 한다.
⑤ 가변저항으로 현재 진행 중인 숫자를 변경하는 기능이 자연스럽게 작동하여야 한다.
⑥ 시리얼 모니터에 게임 현황이 올바르게 출력되어야 한다.

해답: https://arduino.datahub.pe.kr/2023/src/5.%20Solution/2022-DO-HS%20369Game.ino