



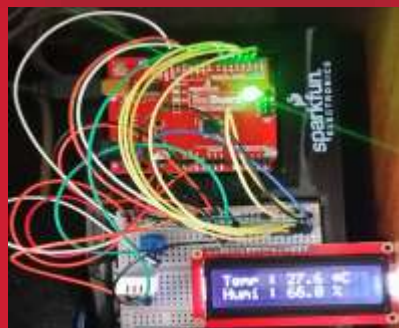
Arduino-basic

[wk06]



LED — II

FND & 4-digit FND



Learn how to code Arduino from scratch

Comsi, INJE University

2nd semester, 2018

Email : chaos21c@gmail.com



My ID (ARnn)

성명	ID
백동진	AR01
김도훈	AR02
김희찬	AR03
류재현	AR04
문민규	AR05
박진석	AR06
이승현	AR07
이승협	AR08
이후정	AR09
최민구	AR10

김다영	AR11
공진영	AR12
김해인	AR13
류성현	AR14
류재환	AR15
박상현	AR16
박해주	AR17
백지혜	AR18
송원식	AR19
신송주	AR20
윤지훈	AR21
정은성	AR22



[Review]

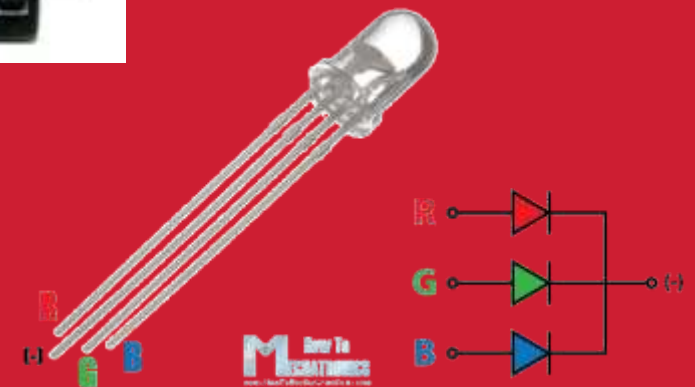
◆ [wk05]

- **Arduino LED - I**
- **Complete your project**
- **Submit file : ARnn_Rpt03.zip**



4. LED

Light Emitting Diode



4.1 LED 교차 점멸



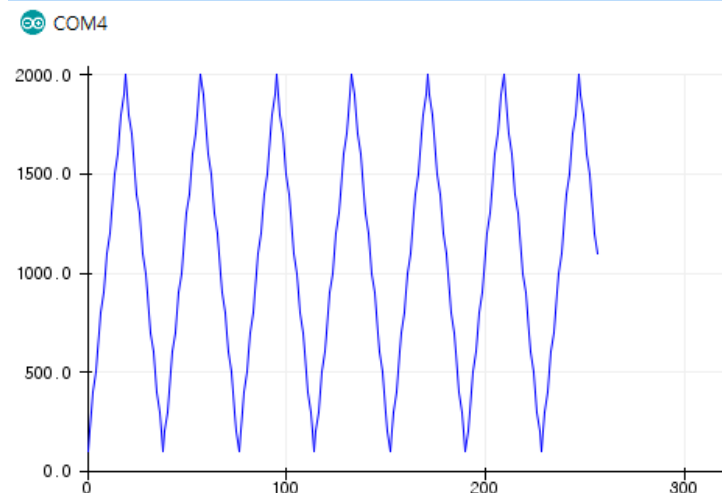
4.1.3 LED control - 교차 점멸

EX 4.1 LED 교차 점멸 (3/3)

실습 결과 LED A와 B가 0.1초 단위로 교차하며 점멸한다.

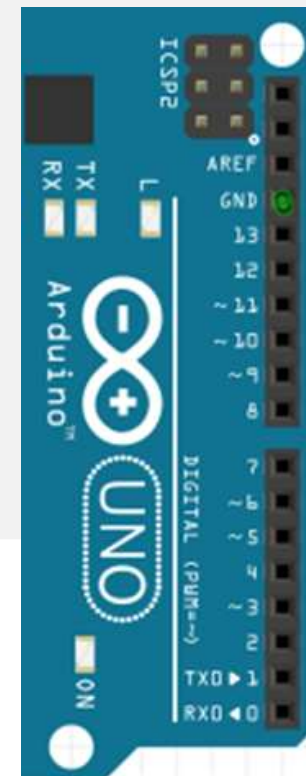
응용 문제 점멸 주기가 0.1초부터 2초로 0.1초 단위로 증가하였다가 다시 반대로 2초부터 0.1초까지 감소하는 동작을 반복하는 스케치를 작성해 보자.
(hint: delay 명령어의 괄호 안의 숫자를 증감시킨다.)

delay = 1600 msec	delay = 500 msec
delay = 1700 msec	delay = 400 msec
delay = 1800 msec	delay = 300 msec
delay = 1900 msec	delay = 200 msec
delay = 2000 msec	delay = 100 msec
delay = 1900 msec	delay = 200 msec
delay = 1800 msec	delay = 300 msec
delay = 1700 msec	delay = 400 msec
delay = 1600 msec	delay = 500 msec



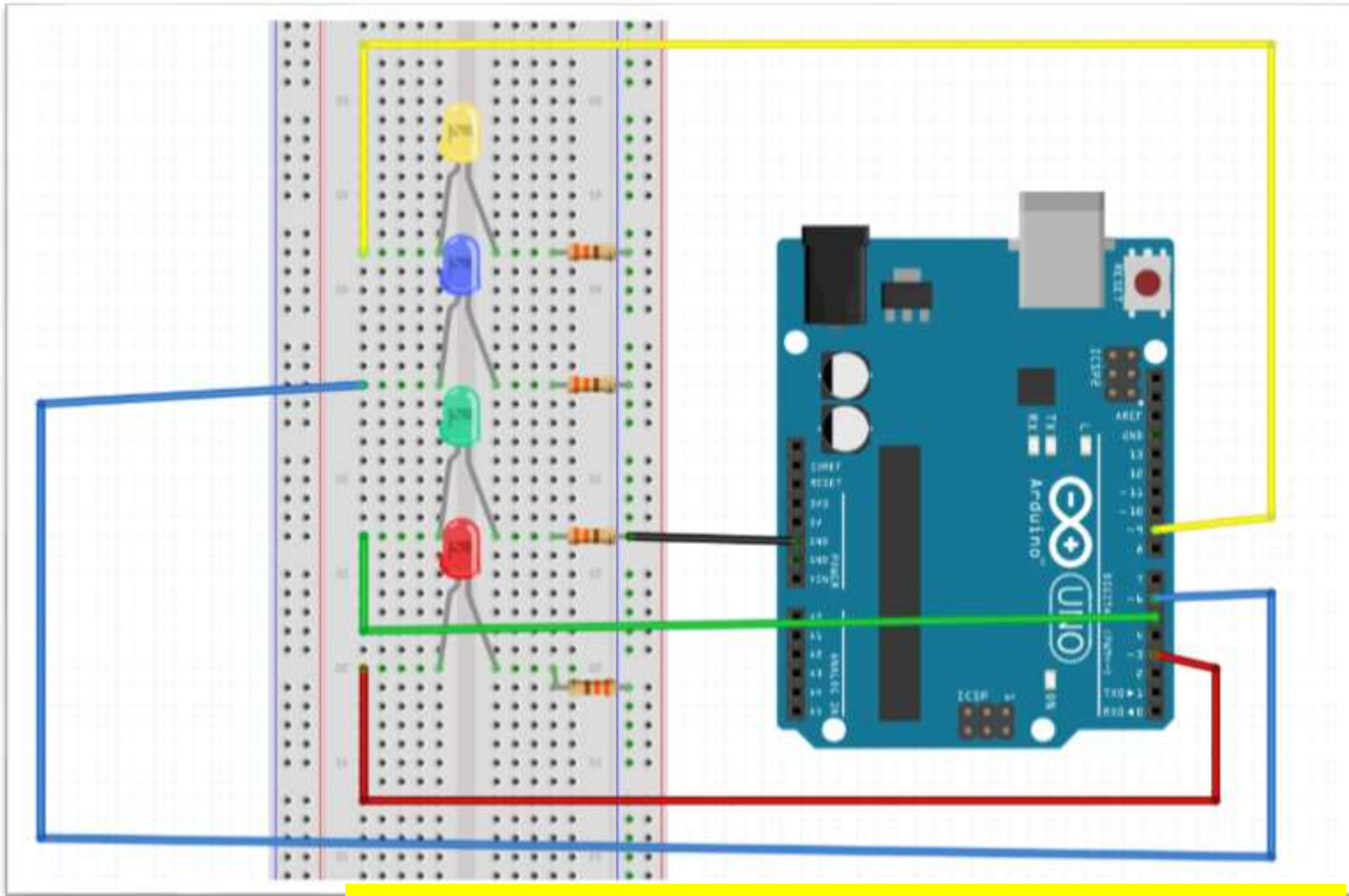
밝기 조절 : 디밍 (Dimming)

- ✓ LED에 입력되는 전력은 PWM (Pulse Width Modulation)을 이용하여 조절.
- ✓ PWM : 고속의 스위칭으로 High와 Low 신호의 비율을 조절하여
LED의 밝기, 모터의 회전 등을 조절하는 방법
- ✓ Arduino에서는 `analogWrite()` 명령어로 구현
- ✓ Arduino UNO의 경우 3, 5, 6, 9, 10, 11 번 핀이 PWM을 지원한다.



4.2.5 LED control - DIY

DIY. 1. 네개의 다른 색깔의 LED를 Arduino에 연결한다. (pwm pin: 3,5,6,9)



완성된 회로를 [ARnn_4led.fzz](#)
로 저장해서 제출.

4.3 RGB LED control - 색상 조절

RGB LED

- ✓ 빛의 삼원색인 빨강(Red), 초록(Green), 파랑(Blue)빛을 조절하여 다양한 색을 표현하는 LED.
- ✓ 각각의 색이 0~255단계로 조절됨.
- ✓ 간판, 조명기구 등에 사용
- ✓ 모든 색이 출력될 때 백색 빛을 출력

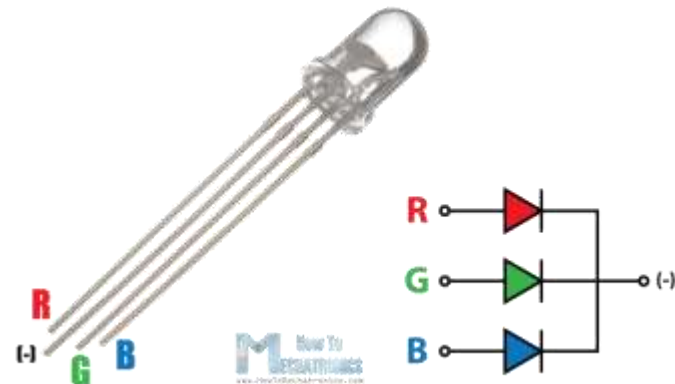
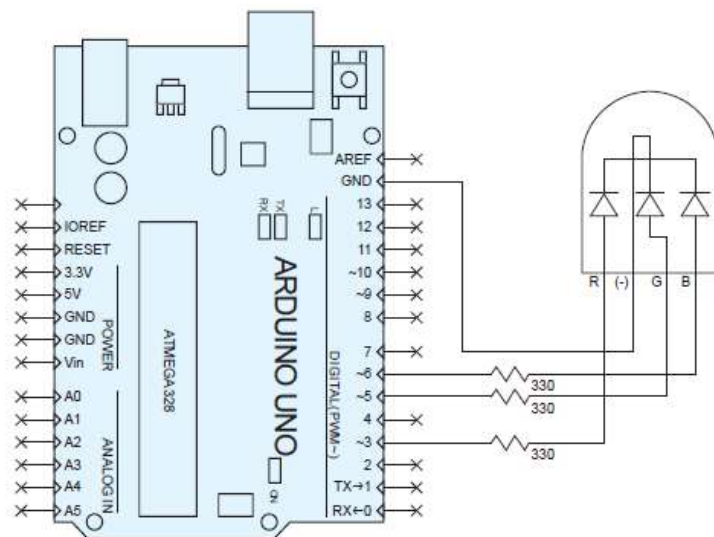


4.3.1 RGB LED control - 색상 조절

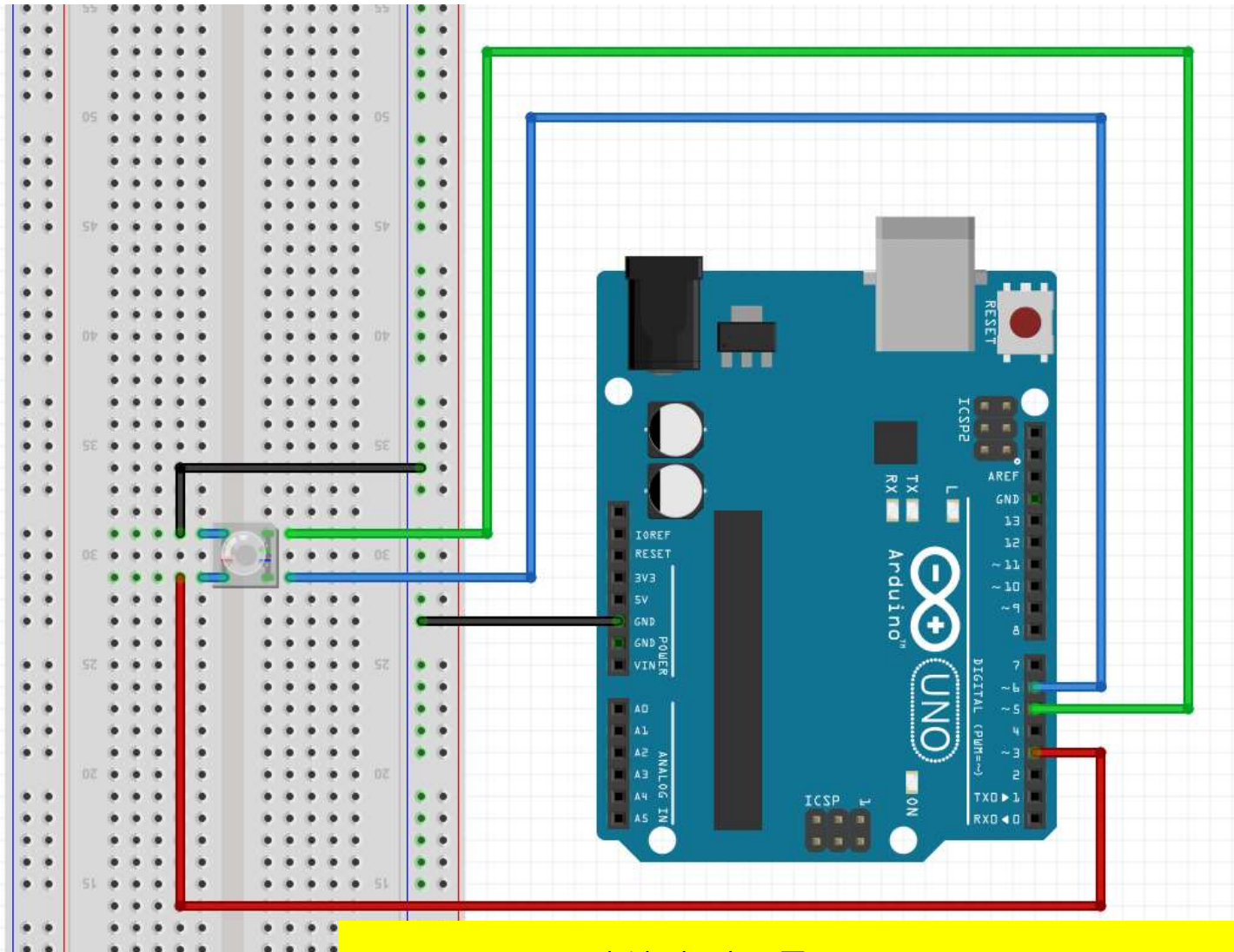
EX 4.3 RGB LED로 색상 표현하기 (1/2)

실습목표 RGB LED를 이용하여 다양한 색을 표현해 보자.

- Hardware**
1. RGB LED는 Red, Green, Blue의 세 개의 Anode 핀과 공통으로 연결된 캐소드핀으로 구성되어 있다.
 2. RGB LED 단독으로 연결하려면 **각 Anode 핀에 330Ω의 저항을 연결**해야 한다.
 3. **저항이 내장된 RGB LED 모듈을 사용한다면 별도의 저항이 필요 없다.**
 4. Red, Green, Blue의 세 개의 Anode 핀을 Arduino의 3, 5, 6 번핀에 연결한다.



<http://howtomechatronics.com/tutorials/arduino/how-to-use-a-rgb-led-with-arduino/>



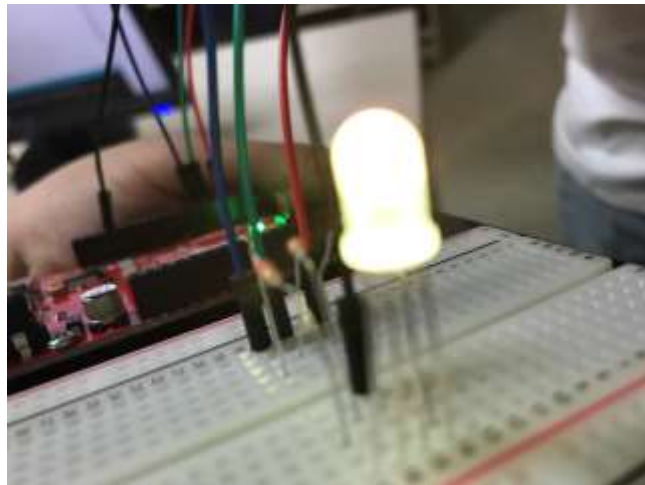
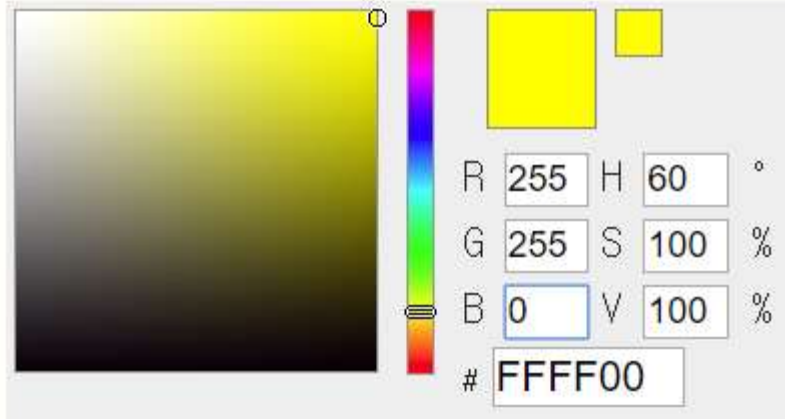
완성된 회로를 **ARnn_RGB.fzz**

로 저장해서 제출.

4.4 RGB LED control - 색상 조절 [DIY]

DIY. RGB LED의 색이 노란색일 때 사진을 촬영하시오.

RGB color picker



ARnn_RGB_Y.png 로 저장



4. LED II

FND

4 1EA



7세그먼트 1채널

공통 음극 7세그먼트
시계나 점수 등의 숫자를
표현 할 때 많이 사용됩니다.

5 1EA



74HC595N

기본 메인보드입니다.
74HC595N LED,
도트매트릭스, NFD 제어 IC 입니다.

3 1EA



7세그먼트 4채널

7세그먼트가 4개 연결된 형태의
부품입니다.
총 12개의 핀을 사용합니다.

23 1EA

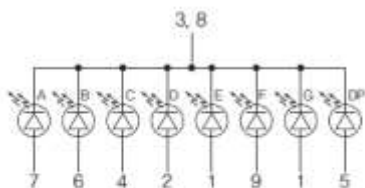
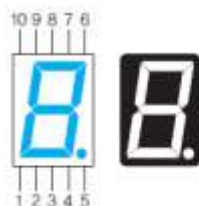


8x8 도트매트릭스 모듈

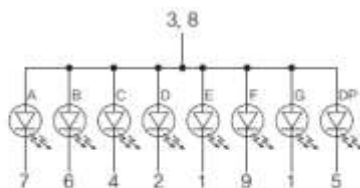
LED로 다양한 연출을
할 수 있습니다.

FND (Flexible Numeric Display)

- ✓ LED의 조합으로 숫자를 표시하는 장치
- ✓ 7개의 LED를 사용하기 때문에 7-segment 라고도 함.
- ✓ 숫자뿐만 아니라 간단한 기호나 16진수 까지 표현 가능



(a)



(b)

그림 4.4 Common Cathode 형(a)와 Common Anode 형(b)

표 4.1 Common Cathode FND 표시

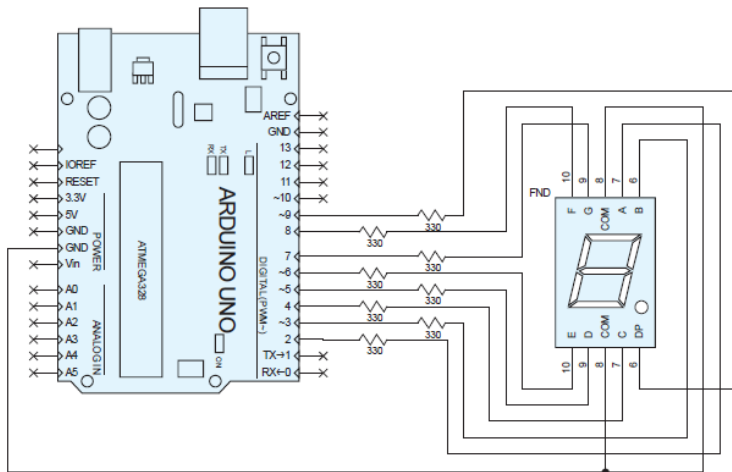
캐소드 공통 7-세그먼트 한 자리 제어 방법										7-Seg. 출력 내용
I/O 포트 출력 내용										16진수
Q0	DP	G	F	E	D	C	B	A	16진수	7-Seg. 출력 내용
1	X	X	X	X	X	X	X	X	X	8 (소등)
0	0	0	1	1	1	1	1	1	0x3f	8 (0)
0	0	0	0	0	0	1	1	0	0x06	8 (1)
0	0	1	0	1	1	0	1	1	0x5b	8 (2)
0	0	1	0	0	1	1	1	1	0x4f	8 (3)
0	0	1	1	0	0	1	1	0	0x66	8 (4)
0	0	1	1	0	1	1	0	1	0x6d	8 (5)
0	0	1	1	1	1	1	0	1	0x7d	8 (6)
0	0	0	0	0	0	1	1	1	0x27	8 (7)
0	0	1	1	1	1	1	1	1	0x7f	8 (8)
0	0	1	1	0	1	1	1	1	0x6f	8 (9)
0	0	1	1	1	0	1	1	1	0x77	8 (A)
0	0	1	1	1	1	1	0	0	0x7c	8 (b)
0	0	0	1	1	1	0	0	1	0x39	8 (C)
0	0	1	0	1	1	1	1	0	0x5e	8 (d)
0	0	1	1	1	1	0	0	1	0x79	8 (E)
0	0	1	1	1	0	0	0	1	0x71	8 (F)
0	1	0	0	0	0	0	0	0	0x80	8 (.)

EX 4.4

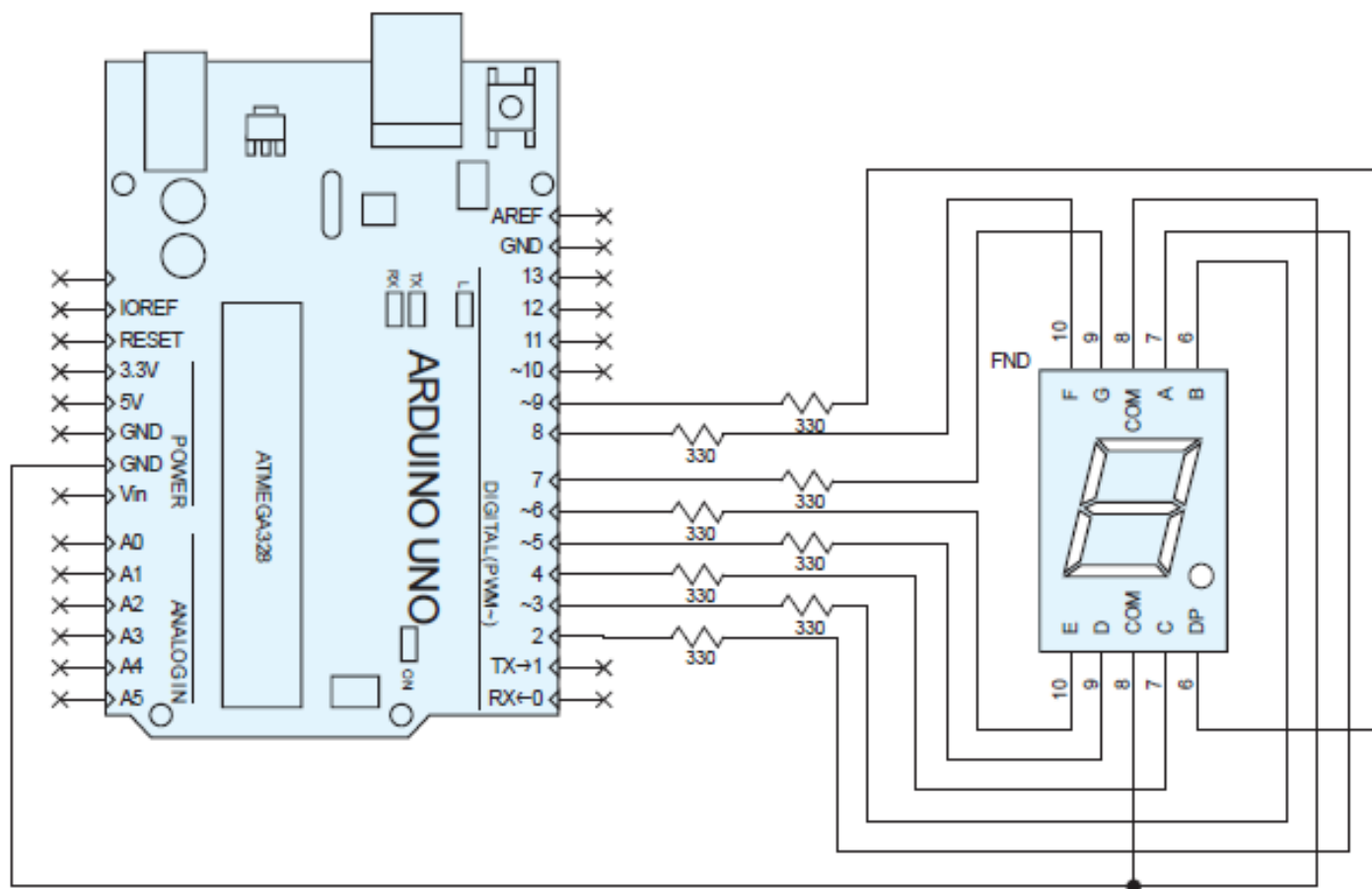
FND 제어 (1/3)

실습목표 Common Cathode FND를 이용하여 0~9의 숫자를 표시해보자.

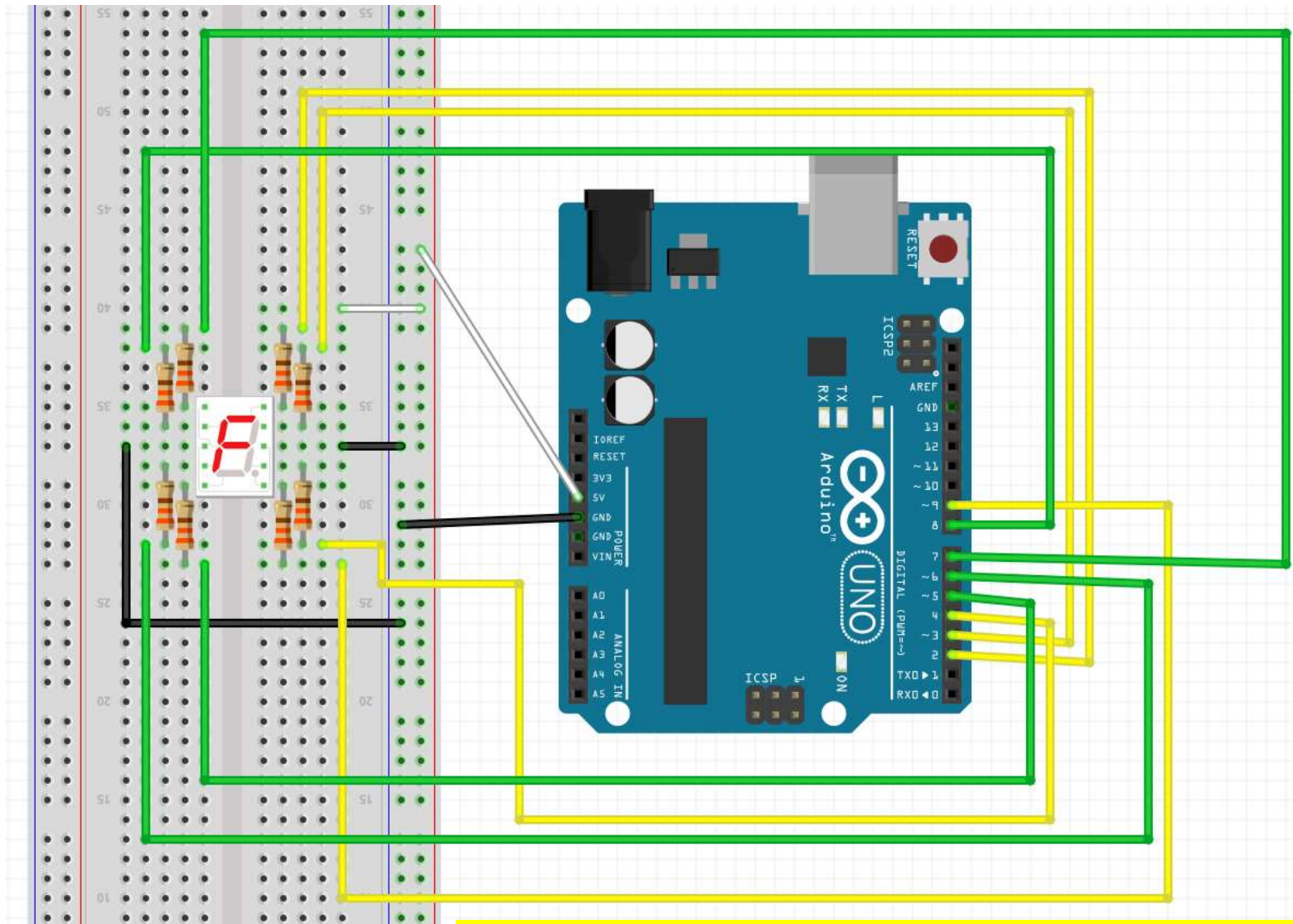
- Hardware**
1. Common Cathode형 FND는 그림 4.2의 (a)와 같이 3번과 8번핀이 Cathode 핀으로 함께 연결되어 있다. 즉 FND의 3번과 8번핀을 GND에 연결하고 나머지 핀들에 HIGH신호를 주어 FND에 숫자를 표시한다.
 2. GND에 연결되는 3번과 8번핀을 제외한 나머지 핀들에는 FND 내의 LED의 전류를 제한하기 위해 330Ω 저항을 연결한다.
 3. 원하는 숫자를 표시하기 위해선 2~9번핀에 표 4.1을 참고하여 신호를 출력한다.



4.5.2 FND 제어

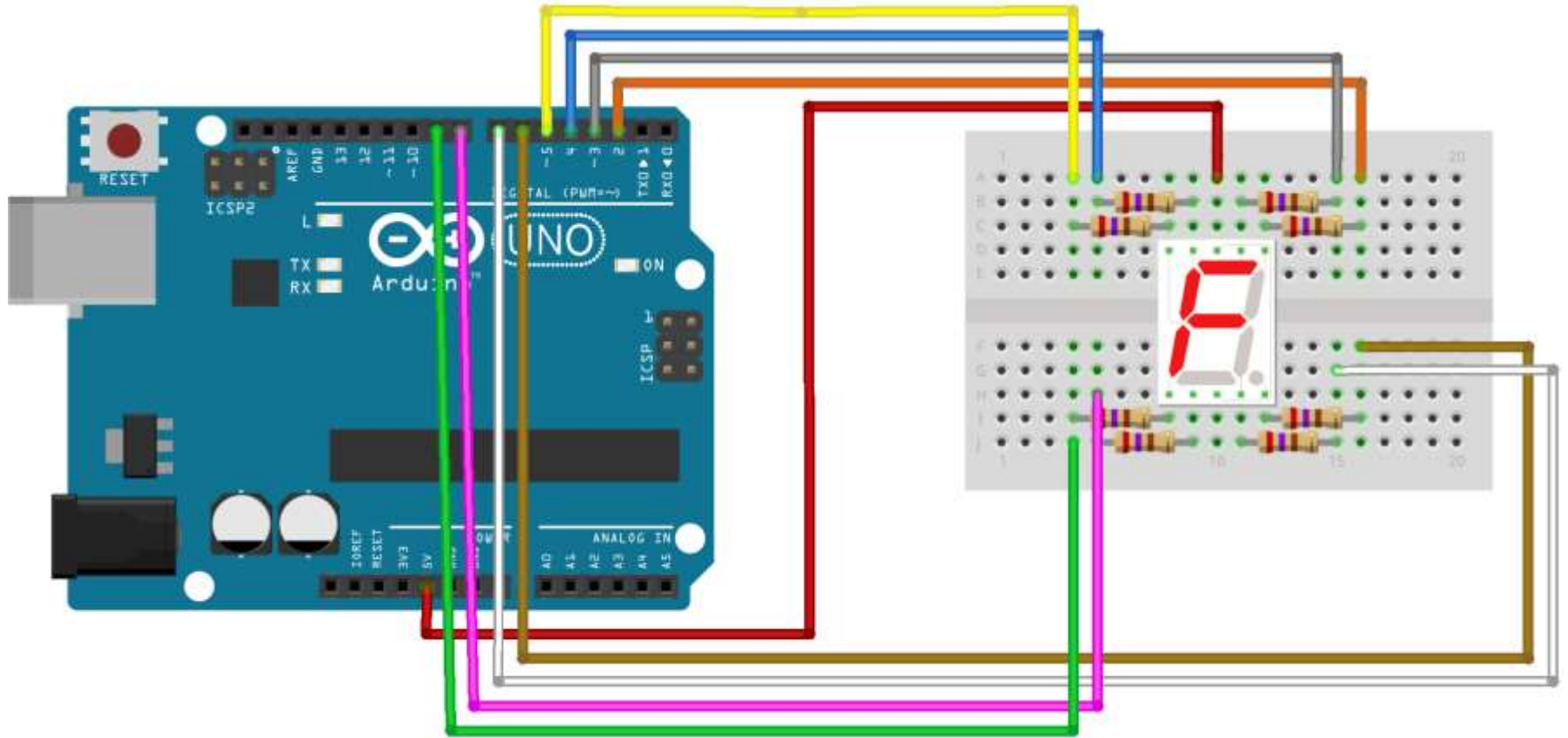


4.5.2.1 FND 제어



Fritzing 으로 회로를 디자인하고
[ARnn_fnd.fzz](#) 로 저장해서 제출.

4.5.2.2 FND 제어



fritzing

4.5.3 FND 제어

EX 4.4 FND 제어 (2/3)

Commands • void 함수(변수1, 변수2, ...){
};

‘함수(변수1, 변수2)’를 이용하여 ‘{ }’ 내의 명령을 호출하여 사용한다. ‘변수1’과 ‘변수2’등을 함께 선언하면 함수 내에서 그 변수를 사용할 수 있다. 반복되는 구문을 설정해 놓고 호출하여 사용하면 편리하다.

- pinMode(핀번호, 설정)

핀의 입출력 모드를 설정한다. ‘핀번호’에는 설정하고자 하는 핀의 번호와 ‘설정’에는 입력으로 사용하기 위해선 ‘INPUT’, 출력으로 사용하기 위해선 ‘OUTPUT’, 입력이며 풀업 사용시 ‘INPUT_PULLUP’을 적는다.

- digitalWrite(핀번호, 값)

핀에 디지털 출력 (High or Low) 을 한다. ‘핀번호’에는 출력하고자 하는 핀의 번호를, ‘값’에는 ‘HIGH’ 혹은 ‘LOW’를 설정하여 High 혹은 Low 출력을 한다.

- for(변수=시작 값 ; 조건 ; 변수의 증분){ }

변수의 시작 값부터 조건이 만족하는 경우 ‘{ }’ 내의 명령을 수행한다. ‘변수의 증분’에서는 1회 명령이 수행될 때 마다 변수를 증가 혹은 감소시킨다.

4.5.4 FND 제어

EX 4.4 FND 제어 (3/3)

- Sketch 구성
1. FND에 숫자를 표시할 때 어떤 LED를 켤지에 대한 정보를 담은 상수를 설정한다.
 2. FND동작에 필요한 핀을 출력으로 설정한다.
 3. FND를 동작시키는 'fndDisplay(int displayValue)' 라는 함수를 만든다.
 4. 함수를 이용하여 1초 간격으로 FND에 숫자를 표시한다.

실행 결과 FND의 숫자가 0~9까지 약 1초 간격으로 변화한다.

4.5.4.1 FND 제어 - code

```
ex_4_4_1_start$
1 /*
2  예제 4.4.1
3  FND 제어 0~9까지 1초단위로 표시하기
4 */
5
6 // 0~9까지 LED 표시를 위한 상수 설정
7 const byte number[10] = {
8 //dot  gfedcba
9  B00111111,  //0
10 B00000110,  //1
11 B01011011,  //2
12 B01001111,  //3
13 B01100110,  //4
14 B01101101,  //5
15 B01111101,  //6
16 B00000111,  //7
17 B01111111,  //8
18 B01101111,  //9
19 };
20
```

```
21 void setup()
22 {
23   // 2~9번 핀을 a b c d e f g dot 의 순서로 사용한다
24   // 2~9번핀을 출력으로 초기화 시킨다.
25   for(int i = 2; i <= 9; ++i){
26     pinMode(i, OUTPUT);
27   };
28   // 점은 표시하지 않는다
29   digitalWrite(9, LOW);
30 }
31
32 void loop()
33 {
34   // k값을 0~9로 변화시킨다.
35   for(int k = 0; k <= 9; ++k){
36     fndDisplay(k);  // k값을 출력한다
37     delay(1000);
38   };
39 }
40
41 // LED 점등
42 void fndDisplay(int displayValue){
43   // bitValue 변수를 선언한다.
44   boolean bitValue;
45
46   for(int i=2; i<=9; ++i){
47     // 2~9번핀에 모두 LOW 신호를 줘서 소등시킨다
48     digitalWrite(i, LOW);
49   };
50   for(int i=0; i<=7; ++i){
51     // number 상수의 하나의 비트값을 읽는다
52     bitValue = bitRead(number[displayValue], i);
53     // 앞서 읽은 비트값을 2~9번핀에 출력시킨다
54     digitalWrite(i+2, bitValue);
55   };
56 }
```

EX 4.4 FND 제어 (3/3)

DIY 위의 예제를 0~F까지의 16진수를 표시하도록 스케치를 수정하여 보자.
(hint: LED 표시를 위한 상수에 A~F를 추가시켜서 불러와 사용하자)

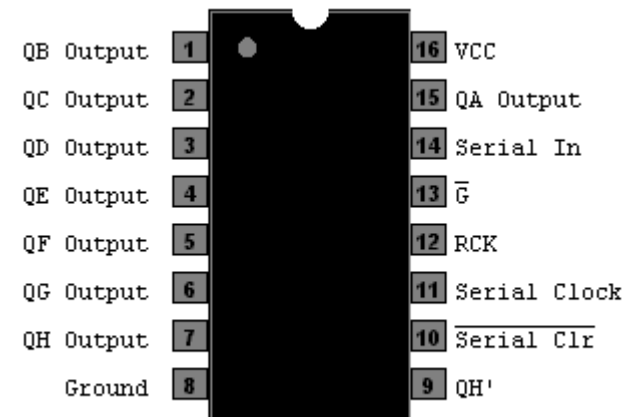
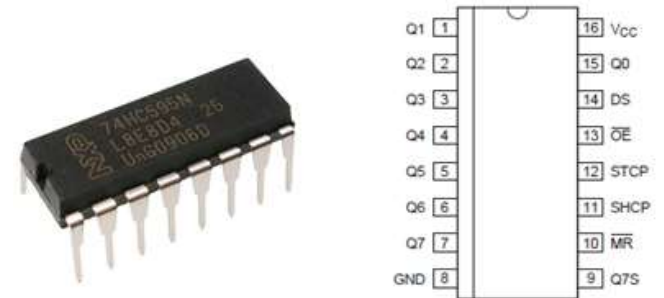
‘A’가 출력된 화면을 **ARnn_A.png**

로 저장해서 제출. (아두이노 회로를 포함해서 촬영)

4.6 FND 제어 : 74595 IC (74HC595N)

74595 IC

- ✓ 직렬 신호로 입력된 데이터를 병렬 신호로 변환
- ✓ FND의 8개의 LED를 켜기위한 신호를 3개의 신호선으로 입력 받아 8개의 FND 신호로 출력
- ✓ shiftout() 명령어로 구현.



SHCP : shift register clock input
STCP : storage register clock input
DS : serial data input

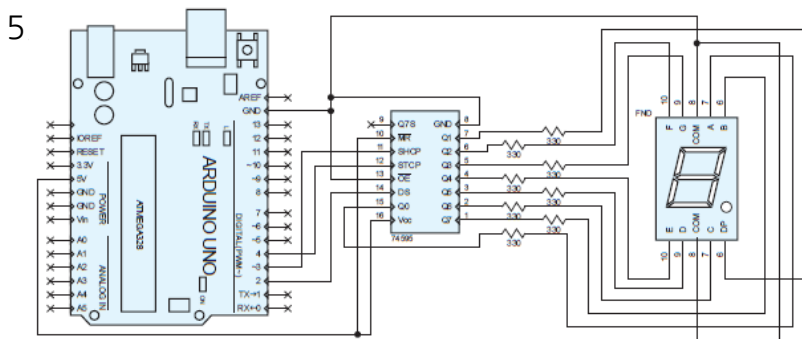
- ✓ DS, SHCP, STCP 세 핀으로 FND 제어
- ✓ 동작 순서
 1. STCP에 'LOW' 신호 입력
 2. SHCP의 클럭에 맞춰 DS로 데이터 전송
 3. 전송 후, STCP에 'HIGH' 신호를 주어 출력핀으로 신호를 출력
- ✓ Shiftout() 함수로 동작 시킴.

4.6.1 FND 제어 : 74595 IC (74HC595N)

EX 4.4.2 74595를 이용한 FND 제어 (1/3)

실습목표 Common Cathode FND를 이용하여 0~9의 숫자를 표시해보자.

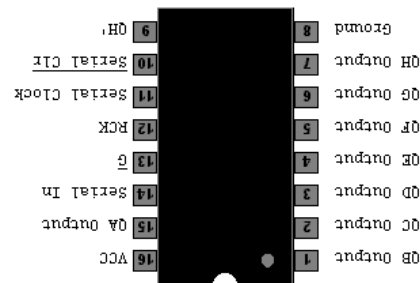
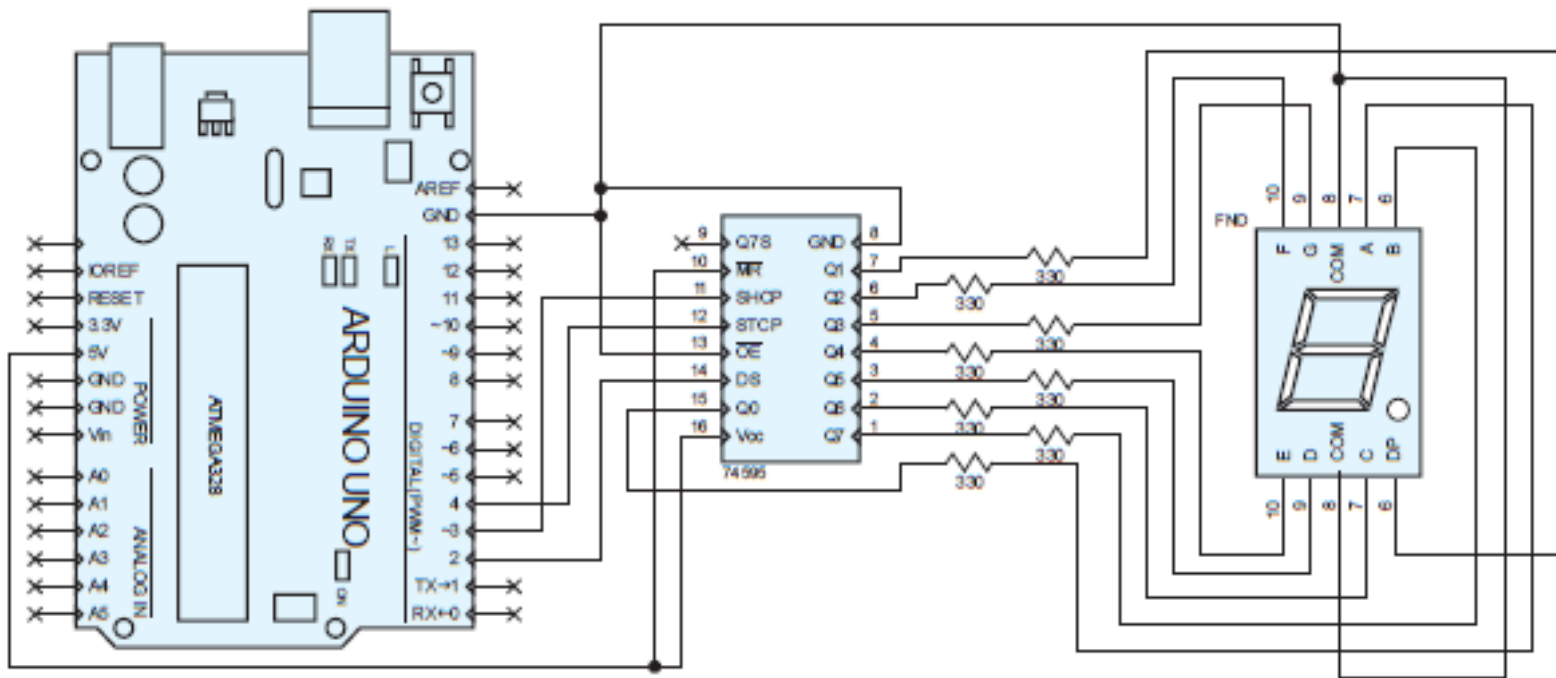
- Hardware**
1. 예제 4.4.1과 동일한 동작을 하지만 Arduino의 입출력 핀을 절약하기 위해 74595 IC를 중간에 연결한다.
 2. **Arduino에서는 2, 3, 4 세 개의 핀을 이용하여 74595 IC로 신호를 출력**한다.
각 핀을 Arduino에 연결한다.
 3. 74595 IC의 (MR) $\bar{}$ 핀과 Vcc 핀에는 5V를 연결하고 (OE) $\bar{}$ 와 GND핀은 Arduino의 GND에 연결한다.
 4. 74595 IC에서는 DS, SHCP, STCP 핀으로부터 입력된 신호를 이용하여 Q0~Q7 핀에 신호를 출력한다. Q0~Q7 핀을 FND의 Anode 핀에 연결한다.



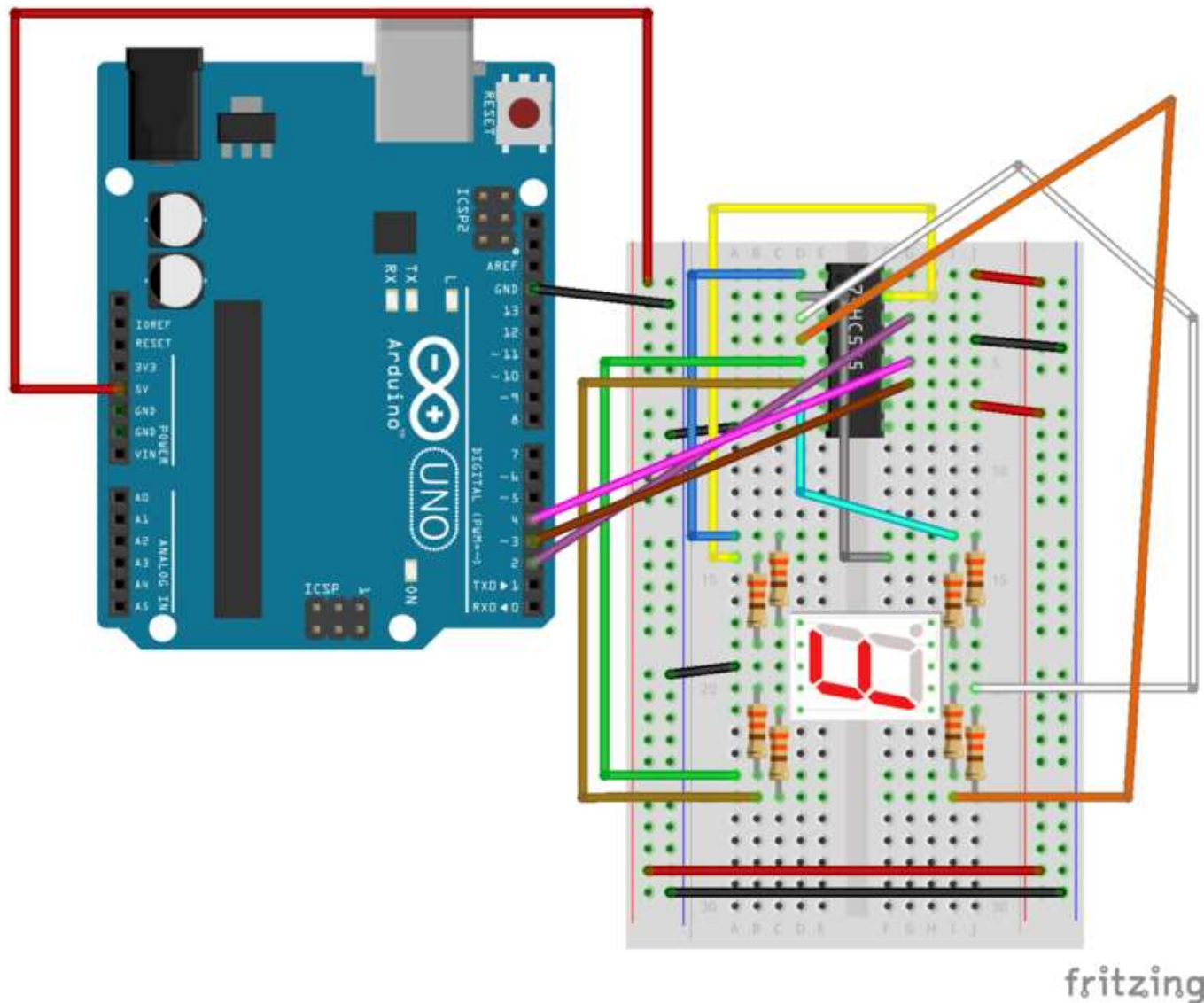
4.6.2.1 FND 제어 : 74595 IC

EX 4.4.2

74595를 이용한 FND 제어 (1/3)

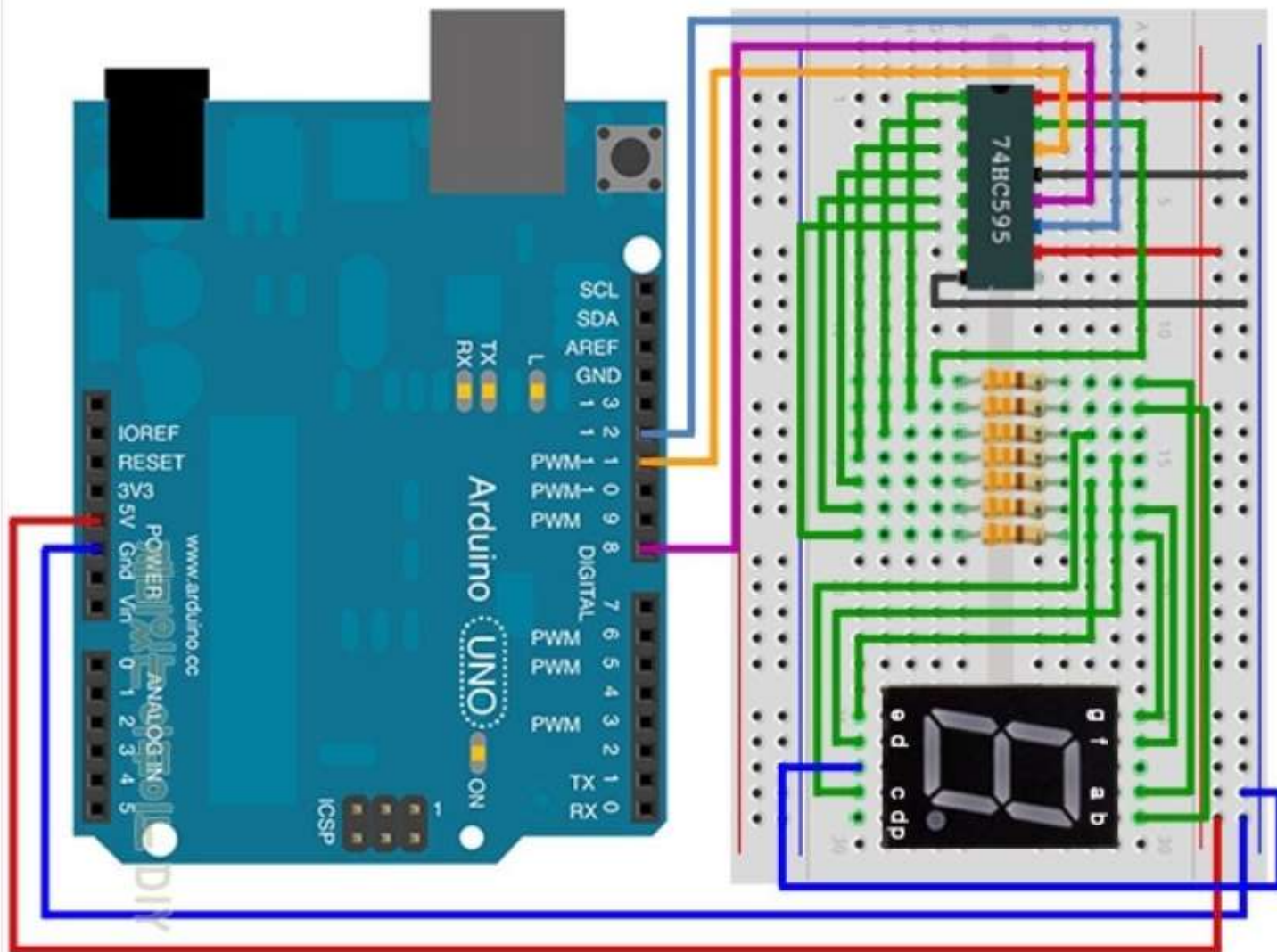


4.6.2.2 FND 제어 : 74595 IC



QB Output	1	16	VCC
QC Output	2	15	QA Output
QD Output	3	14	Serial In
QE Output	4	13	\bar{G}
QF Output	5	12	RCK
QG Output	6	11	Serial Clock
QH Output	7	10	$\overline{\text{Serial Clr}}$
Ground	8	9	QH'

4.6.2.3 FND 제어 : 74595 IC



EX 4.4.2 74595를 이용한 FND 제어 (2/3)

Commands • void 함수(변수1, 변수2, ...){
};

‘함수(변수1, 변수2)’ 를 이용하여 { } 내의 명령을 호출하여 사용한다. ‘변수1’과 ‘변수2’등을 함께 선언하면 함수 내에서 그 변수를 사용할 수 있다. 반복되는 구문을 설정해 놓고 호출하여 사용하면 편리하다.

- pinMode(핀번호, 설정)

핀의 입출력 모드를 설정한다. ‘핀번호’ 에는 설정하고자 하는 핀의 번호와 ‘설정’에는 입력으로 사용하기 위해선 ‘INPUT’, 출력으로 사용하기 위해선 ‘OUTPUT’, 입력이며 풀업 사용시 ‘INPUT_PULLUP’을 적는다.

- digitalWrite(핀번호, 값)

핀에 디지털 출력 (High or Low) 을 한다. ‘핀번호’ 에는 출력하고자 하는 핀의 번호를, ‘값’에는 ‘HIGH’ 혹은 ‘LOW’ 를 설정하여 High 혹은 Low 출력을 한다.

- shiftOut(데이터 핀, 클럭 핀, 출력비트 순서, 출력 값)

데이터 핀으로는 비트단위로 출력될 핀 번호를 써준다. 클럭 핀에는 데이터가 출력될 때 토글되는 클럭 출력에 사용할 핀 번호를 써준다. 출력비트 순서는 비트 데이터의 맨 왼쪽부터 순차적으로 출력하고자 하면 ‘MSBFIRST’, 맨 오른쪽부터 순차적으로 출력하고자 하면 ‘LSBFIRST’를 써 분다. 출력 값에는 실제 출력할 데이터를 써 준다. 이 때 데이터는 8비트 즉 2진수 8자리의 숫자를 갖는다.

EX 4.4.2 74595를 이용한 FND 제어 (2/3)

Commands • void 함수(변수1, 변수2, ...){
};

‘함수(변수1, 변수2)’ 를 이용하여 { } 내의 명령을 호출하여 사용한다. ‘변수1’과 ‘변수2’등을 함께 선언하면 함수 내에서 그 변수를 사용할 수 있다. 반복되는 구문을 설정해 놓고 호출하여 사용하면 편리하다.

- pinMode(핀번호, 설정)

핀의 입출력 모드를 설정한다. ‘핀번호’ 에는 설정하고자 하는 핀의 번호와 ‘설정’에는 입력으로 사용하기 위해선 ‘INPUT’, 출력으로 사용하기 위해선 ‘OUTPUT’, 입력이며 풀업 사용시 ‘INPUT_PULLUP’을 적는다.

- digitalWrite(핀번호, 값)

핀에 디지털 출력 (High or Low) 을 한다. ‘핀번호’ 에는 출력하고자 하는 핀의 번호를, ‘값’에는 ‘HIGH’ 혹은 ‘LOW’ 를 설정하여 High 혹은 Low 출력을 한다.

- shiftOut(데이터 핀, 클럭 핀, 출력비트 순서, 출력 값)

데이터 핀으로는 비트단위로 출력될 핀 번호를 써준다. 클럭 핀에는 데이터가 출력될 때 토글되는 클럭 출력에 사용할 핀 번호를 써준다. 출력비트 순서는 비트 데이터의 맨 왼쪽부터 순차적으로 출력하고자 하면 ‘MSBFIRST’, 맨 오른쪽부터 순차적으로 출력하고자 하면 ‘LSBFIRST’를 써 분다. 출력 값에는 실제 출력할 데이터를 써 준다. 이 때 데이터는 8비트 즉 2진수 8자리의 숫자를 갖는다.

4.6.4 FND 제어 : 74595 IC

EX 4.4.2 74595를 이용한 FND 제어 (3/3)

- Sketch 구성**
1. FND에 숫자를 표시할 때 어떤 LED를 켤지에 대한 정보를 담은 상수를 설정한다.
 2. FND동작에 필요한 핀을 출력으로 설정한다.
 3. FND를 동작시키는 'fndDisplay74595(int displayValue)' 라는 함수를 만든다.
 4. 'fndDisplay74595(int displayValue)'에는 'shiftOut()' 명령어를 이용한 FND 동작 스케치를 넣는다.
 5. 함수를 이용하여 1초 간격으로 FND에 숫자를 표시한다.

실행 결과 FND의 숫자가 0~9까지 약 1초 간격으로 변화한다.

4.6.4.1 FND 제어 : 74595 IC - code

ex_4_4_2_start\$

```

1  /*
2  예제 4.4.2
3  74595를 이용하여 FND 제어 0~9까지
4  1초단위로 표시하기
5  */
6
7  // 0~9까지 LED 표시를 위한 상수 설정
8  const byte number[10] = {
9  //dot  gfedcba
10 B00111111, //0
11 B00000110, //1
12 B01011011, //2
13 B01001111, //3
14 B01100110, //4
15 B01101101, //5
16 B01111101, //6
17 B00000111, //7
18 B01111111, //8
19 B01101111, //9
20 };
21
22 int ds = 2; // 74595의 DS핀을 Arduino의 2번핀에 연결
23 int shcp = 3; // 74595의 STCP핀을 Arduino의 3번핀에 연결
24 int stcp = 4; // 74595의 SHSP핀을 Arduino의 4번핀에 연결
25
26 void setup()
27 {
28 // 2~9번핀을 출력으로 초기화 시킨다.
29 for(int i = 2; i <= 9; ++i){
30 pinMode(i,OUTPUT);
31 };
32 digitalWrite(9,LOW); // 점은 표시하지 않는다
33 }

```

```

35 void loop()
36 {
37 // k값을 0~9로 변화시킨다.
38 for(int k = 0; k <= 9; ++k){
39 fndDisplay74595(k); // k값을 출력한다
40 delay(1000); // 1초간 지연시킨다.
41 };
42 }
43
44 // LED 점등
45 void fndDisplay74595(int displayValue){
46 // STCP에 LOW 신호를 보내서 74595로 데이터전송을 시작한다.
47 digitalWrite(stcp, LOW);
48 // shiftOut 명령어로 74595에 출력을 보낸다.
49 shiftOut(ds, shcp, MSBFIRST, number[displayValue]);
50 // STCP에 HIGH 신호를 보내서 74595로 데이터전송을 종료한다.
51 digitalWrite(stcp, HIGH);
52 }

```

4.6.5 FND 제어 : 74595 - DIY

EX 4.4.2 74595를 이용한 FND 제어 (3/3)

DIY 위의 예제를 0~F까지의 16진수를 표시하도록 스케치를 수정하여 보자.

(hint: LED 표시를 위한 상수에 A~F를 추가시켜서 불러와 사용하자)

‘E’가 출력된 화면을 ARnn_E.png

로 저장해서 제출. (아두이노 회로를 포함해서 촬영)

4.7 4-gigit FND 제어

4-digit FND

- ✓ FND 네 개를 이용하여 네 자리 숫자를 표시하는 부품
- ✓ Common Cathode형과 Common Anode형
- ✓ FND와 핀 구조는 동일하지만 각 자릿수를 선택하는 핀 추가

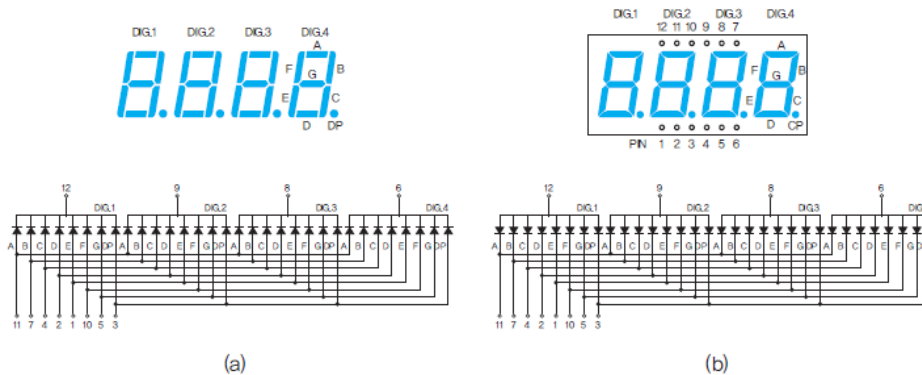


그림 4.6 4-digit FND와 내부 회로. Common Cathode (a), Common Anode (b)



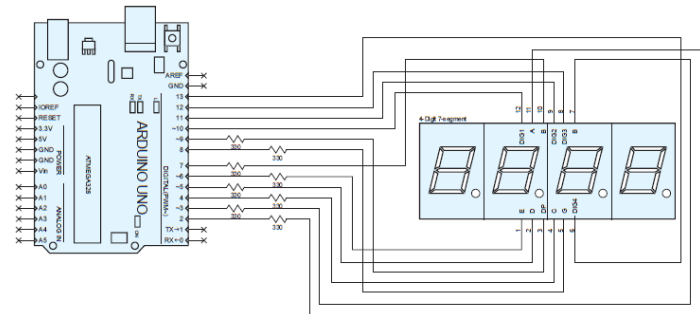
그림 4.7 실험에 사용한 4-digit FND

4.7.1 4-digit FND 제어

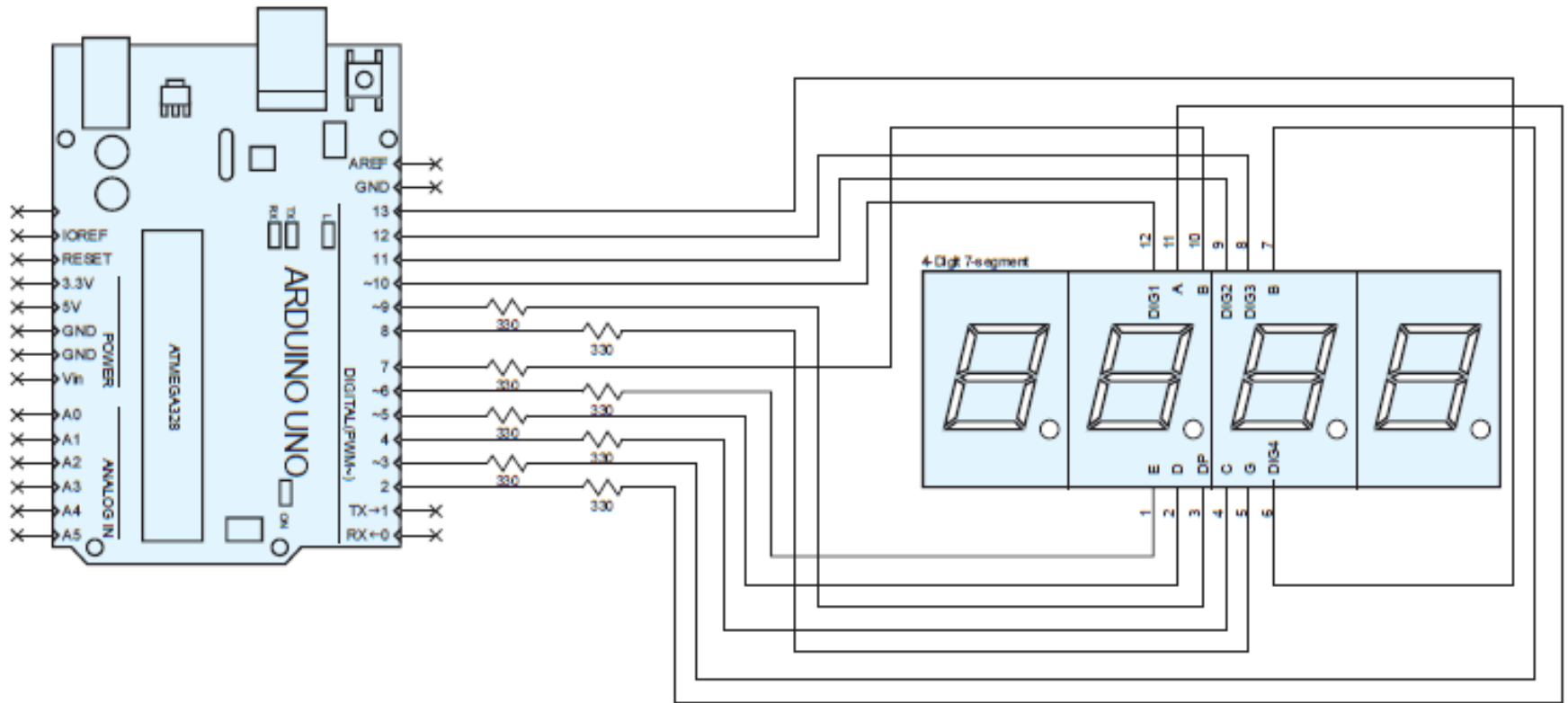
EX 4.5.1 4-digit FND로 0000~9999 숫자 표시하기 (1/3)

실습목표 Common Cathode 4-digit FND를 이용하여 0000~9999까지 1초 간격으로 증가하는 스케치를 작성해 보자.

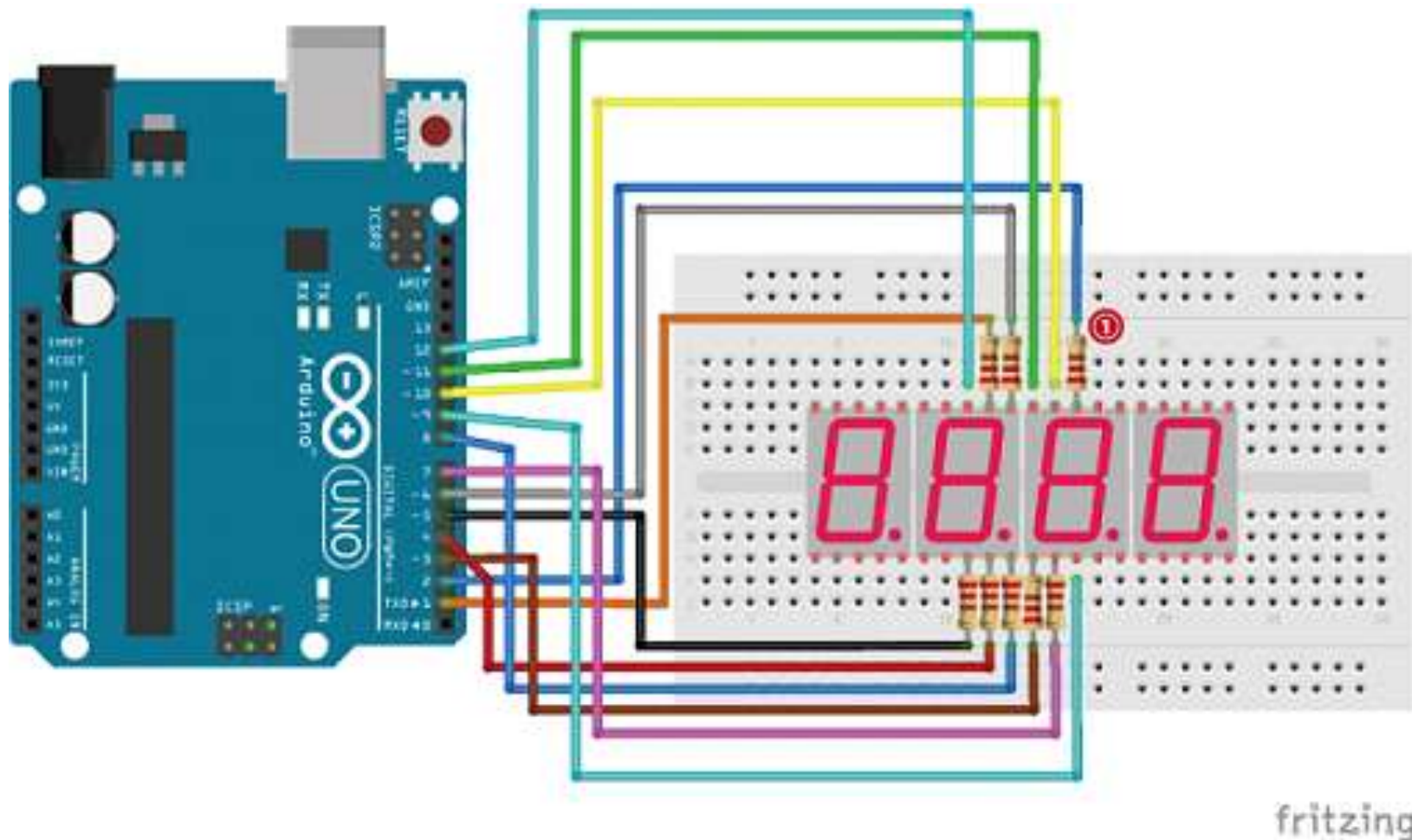
- Hardware**
1. 4-digit FND는 4개의 FND를 연결한 부품이다.
 2. 각각의 FND에는 DIG1~DIG4 네 개의 핀이 각각의 FND의 Common Cathode로 연결되어 있다.
 3. A~G, DP핀은 하나의 FND를 동작시킬 때와 같이 330Ω저항을 통하여 Arduino 2~9번핀에 연결한다.
 4. 맨 왼쪽 FND를 동작시키려면 DIG1에만 LOW신호를 준 상태에서 A~G, DP 핀에 원하는 숫자를 쓰기 위한 신호를 주어야 한다.
 5. 두번째 FND를 동작시키려면 DIG2에만 LOW신호를 준 상태에서 A~G, DP 핀에 원하는 숫자를 쓰기 위한 신호를 주어야 한다.
 6. DIG1~DIG4에 모두 LOW신호를 주면 모두 같은 숫자가 표시된다.



4.7.2 4-gigit FND 제어



4.7.2.1 4-digit 7-segment FND 제어 (참고 회로)



4.7.3 4-gigit FND 제어

EX 4.5.1 4-digit FND로 0000~9999 숫자 표시하기 (2/3)

Commands • void 함수(변수1, 변수2, ...){
};

‘함수(변수1, 변수2)’ 를 이용하여 { } 내의 명령을 호출하여 사용한다. ‘변수1’과 ‘변수2’등을 함께 선언하면 함수 내에서 그 변수를 사용할 수 있다. 반복되는 구문을 설정해 놓고 호출하여 사용하면 편리하다.

- pinMode(핀번호, 설정)

핀의 입출력 모드를 설정한다. ‘핀번호’ 에는 설정하고자 하는 핀의 번호와 ‘설정’에는 입력으로 사용하기 위해선 ‘INPUT’, 출력으로 사용하기 위해선 ‘OUTPUT’, 입력이며 풀업 사용시 ‘INPUT_PULLUP’을 적는다.

- digitalWrite(핀번호, 값)

핀에 디지털 출력 (High or Low) 을 한다. ‘핀번호’ 에는 출력하고자 하는 핀의 번호를, ‘값’에는 ‘HIGH’ 혹은 ‘LOW’ 를 설정하여 High 혹은 Low 출력을 한다.

- for(변수=시작 값 ; 조건 ; 변수의 증분){ }

변수의 시작 값부터 조건이 만족하는 경우 { } 내의 명령을 수행한다. ‘변수의 증분’에서는 1회 명령이 수행될 때 마다 변수를 증가 혹은 감소시킨다.

4.7.4 4-gigit FND 제어

EX 4.5.1 4-digit FND로 0000~9999 숫자 표시하기 (3/3)

- Sketch 구성**
1. FND에 숫자를 표시할 때 어떤 LED를 점등할 지에 대한 정보를 담은 상수를 설정한다.
 2. FND동작에 필요한 핀을 출력으로 설정한다.
 3. DIG에 연결된 핀을 모두 LOW로 설정하여 모든 FND가 켜지도록 한다.
 4. 예제 4.4에서 설정한 'fndDisplay(int displayValue)' 함수를 응용하여 1초 간격으로 0~9까지의 숫자를 모든 FND에 표시한다.

실행 결과 4개의 FND의 숫자가 0~9까지 1111 단위로 약 1초 간격으로 변화한다.

4.7.4.1 4-gigit FND 제어 - code

```

6 // 0~9까지 LED 표시를 위한 상수
7 const byte number[10] = {
8   //dot  gfedcba
9   B00111111, //0
10  B00000110, //1
11  B01011011, //2
12  B01001111, //3
13  B01100110, //4
14  B01101101, //5
15  B01111101, //6
16  B00000111, //7
17  B01111111, //8
18  B01101111, //9
19 };
20
21 // 표시할 숫자 변수
22 int count = 0;
23
24 void setup()
25 {
26   // 2~9번 핀을 a b c d e f g dot 의 순서로 사용한다.
27   // 10~13번 핀을 Digit 1~4 의 순서로 사용한다.
28   for(int i = 2; i <= 13; ++i){
29     pinMode(i,OUTPUT); // 2~13번핀을 출력으로 설정한다.
30   };
31
32   // 4 digit와 연결된 10~13번핀에 모두 LOW 신호를 줘서
33   for(int i=10; i<=13; ++i){
34     digitalWrite(i, LOW);
35   };
36 }

```

```

35 void loop()
36 {
37   // count 변수값을 FND에 출력한다.
38   fndDisplay(count);
39
40   // count 변수값이 0~9의 범위를 갖도록한다.
41   if(count >=9) count = 0;
42   else ++count;
43
44   delay(1000);
45 }
46
47 // LED 켜는 루틴
48 void fndDisplay(int displayValue){
49   // bitValue 변수를 선언한다.
50   boolean bitValue;
51
52   // 2~9번핀에 모두 LOW 신호를 줘서 소등시킨다.
53   for(int i=2; i<=9; ++i){
54     digitalWrite(i, LOW);
55   };
56
57   for(int i=0; i<=7; ++i){
58     // number 상수의 하나의 비트값을 읽는다.
59     bitValue = bitRead(number[displayValue], i);
60     // 앞서 읽은 비트값을 2~9번핀에 출력시킨다.
61     digitalWrite(i+2, bitValue);
62   };
63 }

```

4.7.5 4-gigit FND 제어 - DIY

DIY

‘XXX1’, ‘XX2X’, ‘X3XX’, ‘4XXX’ 의 표시가 1초 간격으로 반복하는 스케치를 작성해 보자. (X:는 꺼짐을 나타낸다)

(hint: DIG1~4에 연결된 핀을 제어해보자.)

완성된 코드를 [ARnn_4digit.ino](#)
로 저장해서 제출. (실기 시험 1.)

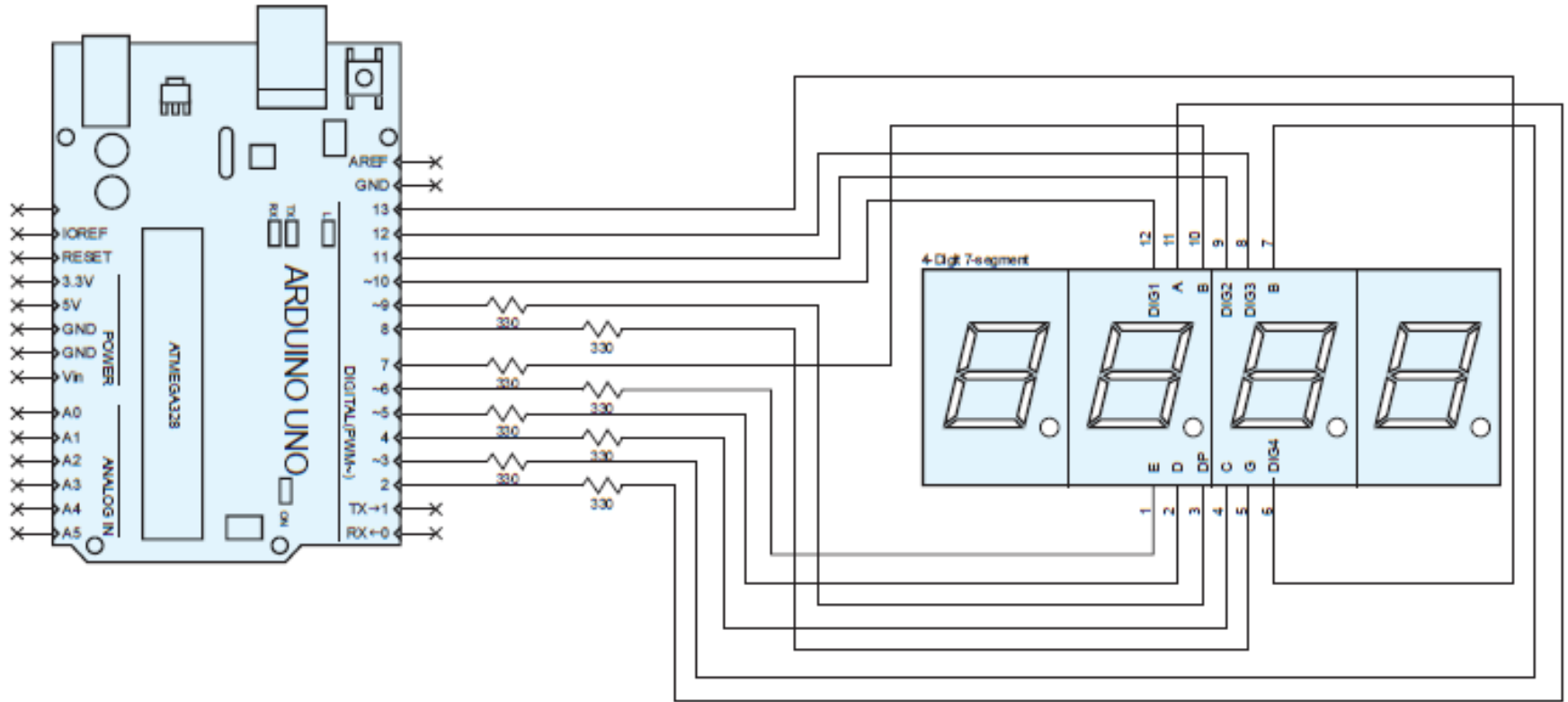
4.8.1 4-digit FND 제어 응용

EX 4.5.2 4-digit FND에 1초마다 증가하는 0~9999 숫자 표시하기 (1/3)

실습 목표 Common Cathode 4-digit FND를 이용하여 0~9999까지 1초 간격으로 증가하는 스케치를 작성해 보자.

- Hardware**
1. 4-digit FND는 4개의 FND를 연결한 부품이다.
 2. 각각의 FND에는 DIG1~DIG4 네 개의 핀이 각각의 FND의 Common Cathode로 연결되어 있다.
 3. A~G, DP핀은 하나의 FND를 동작시킬 때와 같이 330Ω저항을 통하여 Arduino 2~9번 핀에 연결한다.
 4. 맨 왼쪽 FND를 동작시키려면 DIG1에만 LOW신호를 준 상태에서 A~G, DP 핀에 원하는 숫자를 쓰기 위한 신호를 주어야 한다.
 5. 두번째 FND를 동작시키려면 DIG2에만 LOW신호를 준 상태에서 A~G, DP 핀에 원하는 숫자를 쓰기 위한 신호를 주어야 한다.
 6. DIG3, DIG4에 대해서도 동작을 반복한다.
 7. 각각의 FND를 선택하여 점등하는 동작을 빠른 속도로 반복하면 마치 모든 FND가 점등된 것으로 인식된다.

4.8.2 4-digit FND 제어 응용



4.8.3 4-digit FND 제어 응용

EX 4.5.2

4-digit FND에 1초마다 증가하는 0~9999 숫자 표시하기 (2/3)

Commands • void 함수(변수1, 변수2, ...){
};

‘함수(변수1, 변수2)’ 를 이용하여 ‘{ }’ 내의 명령을 호출하여 사용한다. ‘변수1’과 ‘변수2’등을 함께 선언하면 함수 내에서 그 변수를 사용할 수 있다. 반복되는 구문을 설정해 놓고 호출하여 사용하면 편리하다.

- pinMode(핀번호, 설정)

핀의 입출력 모드를 설정한다. ‘핀번호’ 에는 설정하고자 하는 핀의 번호와 ‘설정’에는 입력으로 사용하기 위해선 ‘INPUT’, 출력으로 사용하기 위해선 ‘OUTPUT’, 입력이며 풀업 사용시 ‘INPUT_PULLUP’을 적는다.

- digitalWrite(핀번호, 값)

핀에 디지털 출력 (High or Low) 을 한다. ‘핀번호’ 에는 출력하고자 하는 핀의 번호를, ‘값’에는 ‘HIGH’ 혹은 ‘LOW’ 를 설정하여 High 혹은 Low 출력을 한다.

- millis()

현재 스케치가 시작된 이후로 경과된 시간 값을 가져온다. 밀리세컨즈(1/1000초) 단위의 값을 갖는다.

4.8.4 4-digit FND 제어 응용

- Sketch 구성**
1. FND에 숫자를 표시할 때 어떤 LED를 켤지에 대한 정보를 담은 상수를 설정한다.
 2. FND동작에 필요한 핀을 출력으로 설정한다.
 3. DIG1~4중 하나만 점등 한 뒤 해당 DIG 핀에 연결된 자릿수의 표시를 예제 4.4에서 설정한 'fndDisplay(int displayValue)' 함수를 응용하여 표시한다.
 5. DIG1->DIG2->DIG3->DIG4 순서로 돌아가며 점등시킨다. 해당 DIG핀에 신호를 LOW 했을 때 해당 자릿수가 점등된다.
 6. 빠른 시간으로 4개의 DIG핀을 제어하면 시각적으로 모든 FND가 점등된 것 처럼 보인다.

실습 결과 4개의 FND의 숫자가 0~9999까지 1단위로 약 1초 간격으로 변화한다.

응용 문제 숫자가 증가하는 간격을 0.5초로 변경하여라.

4.8.4.1 4-gigit FND 제어 응용 - code1

```

6 // 0~9까지 LED 표시를 위한 상수
7 const byte number[10] = {
8     //dot  gfedcba
9     B00111111, //0
10    B00000110, //1
11    B01011011, //2
12    B01001111, //3
13    B01100110, //4
14    B01101101, //5
15    B01111101, //6
16    B00000111, //7
17    B01111111, //8
18    B01101111, //9
19 };
20
21 // 4개의 digit에 연결된 핀 설정
22 const byte digitNumber[4] = {13, 12, 11, 10};
23
24 // 표시할 숫자 변수
25 int count = 0;
26
27 // 각 자릿수를 저장하기 위한 변수
28 int value[4];
29
30 // 4개의 digit에 각각 다른 숫자를 표시하기 위해 사용하는 변수
31 int digitSelect = 1;
32
33 // 시간을 측정하는데 사용되는 변수
34 long sampleTime;
35 int count5ms;

```

```

37 void setup()
38 {
39     // 2~9번 핀을 a b c d e f g dot 의 순서로 사용한다.
40     // 10~13번 핀을 Digit 1~4 의 순서로 사용한다.
41     for(int i = 2; i <= 13; ++i){
42         pinMode(i, OUTPUT); // 2~13번핀을 출력으로 설정한다.
43     };
44
45     // 4 digit와 연결된 10~13번핀에 모두 HIGH 신호를
46     // 줘서 소등시킨다.
47     for(int i=10; i<=13; ++i){
48         digitalWrite(i, HIGH);
49     };
50 }

```

4.8.4.2 4-gigit FND 제어 응용 - code2

```

51 void loop()
52 {
53   // 현재 시간을 저장한다.
54   sampleTime = millis();
55
56   // count 변수값을 FND에 출력한다.
57   fndDisplay(digitSelect, value[digitSelect-1]);
58   ++digitSelect;
59   if(digitSelect >= 5) digitSelect = 1;
60
61   // count 변수값이 0~9999의 범위를 갖도록한다.
62   if(count >= 9999) count = 0;
63   else{
64     // 앞서 저장한 시간에서 현재까지의 시간이 5ms일 경우에 다음 명령어를 실행한다.
65     while(millis()-sampleTime < 5);
66
67     ++count5ms;
68     if(count5ms > 200){ //
69       // 5ms * 100 = 1s 가 되었을 때 count를 하나 올려준다
70       ++count;
71
72       // 변수를 각 자릿수로 나눈다
73       value[3] = count / 1000;
74       value[2] = (count - (value[3]*1000)) / 100;
75       value[1] = (count - (value[3]*1000) - (value[2]*100)) / 10;
76       value[0] = count - (value[3]*1000) - (value[2]*100) - (value[1]*10);
77
78       count5ms = 0;
79     }
80   }
81 }

```

```

82 // LED 켜는 루틴
83 void fndDisplay(int digit, int displayValue){
84   // bitValue 변수를 선언한다.
85   boolean bitValue;
86
87   // 4 digit와 연결된 10~13번핀에 모두 HIGH 신호를 줘서 소등시킨다.
88   for(int i=1; i<=4; ++i){
89     digitalWrite(digitNumber[i-1], HIGH);
90   };
91
92   // FND에 원하는 숫자를 표시한다.
93   for(int i=0; i<=7; ++i){
94     // number 상수의 하나의 비트값을 읽는다.
95     bitValue = bitRead(number[displayValue], i);
96     // 앞서 읽은 비트값을 2~9번핀에 출력시킨다.
97     digitalWrite(i+2, bitValue);
98   };
99
100   // 4 digit중 표시를 원하는 digit만 켜다
101   for(int i=1; i<=4; ++i){
102     // 표시하기 원하는 자릿수는 LOW신호를 주어 켜고 나머진 OFF시킨다.
103     if(digit == i) digitalWrite(digitNumber[i-1], LOW);
104     else digitalWrite(digitNumber[i-1], HIGH);
105   };
106 }

```

4.8.5 4-gigit FND 제어 응용 - DIY

DIY 숫자가 증가하는 간격을 0.5초로 변경하여라.

4개의 숫자가 다르게 출력된 화면을 [ARnn_4digit.png](#)
로 저장해서 제출. (아두이노 회로를 포함해서 촬영)



[Practice]

◆ [wk06]

- **Arduino LED – II : FND**
- **Complete your project**
- **Submit file : Arnn_Rpt04.zip**

wk06 : Practice-04 : ARnn_Rpt04.zip

◆ [Target of this week]

- Complete your works
- Save your outcomes and compress all.

제출파일명 : **ARnn_Rpt04.zip**

- 압축할 파일들

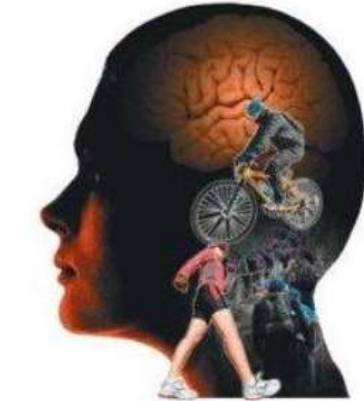
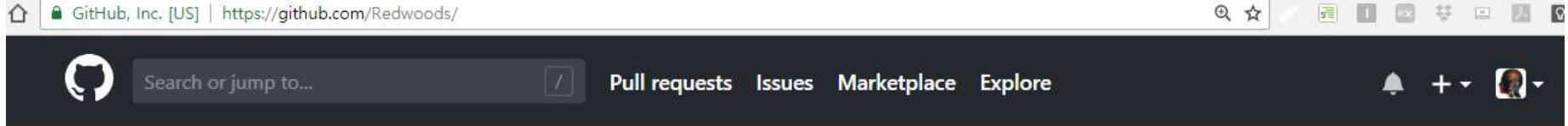
- ① **ARnn_fnd.png**
- ② **ARnn_A.fzz**
- ③ **ARnn_E.png**
- ④ **ARnn_4digit.ino**
- ⑤ **ARnn_4digit.png**

Email : chaos21c@gmail.com

[제목 : id, 이름 (수정)]

● References & good sites

- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.github.com> GitHub
- ✓ <http://www.google.com> Googling
- ✓ <https://www.youtube.com> Youtube



Redwoods Yi

Redwoods

Add a bio

GimHae, Republic of Korea

chaos21c@gmail.com

Overview

Repositories 7

Stars 2

Followers 1

Following 0

Pinned repositories

Customize your pinned repositories

Py

Lectures on coding python from scratch to the advanced level.

Jupyter Notebook

Arduino

Lectures on learning Arduino from scratch to the advanced level in iot environment.

Lec

All lectures by Redwoods in Inje University

Jupyter Notebook

hw-coding

Resource for lecture of Hardware Programming (2017, Inje university)


Arduino

171 contributions in the last year

Contribution settings

Redwoods/Arduino: Lect

GitHub, Inc. [US] | https://github.com/Redwoods/Arduino






Search or jump to...

Pull requests

Issues

Marketplace

Explore



Redwoods / Arduino

Unwatch

1

Star

0

Fork

0

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Insights

Settings

Lectures on learning Arduino from scratch to the advanced level in iot environment.

Edit

Add topics

2 commits

1 branch

0 releases

1 contributor

Branch: master


New pull request


Create new file


Upload files


Find file

Clone or download


 Redwoods 2018 start Latest commit 38ca9e0 28 minutes ago

 ar-basic 2018 start 28 minutes ago

 ar-iot 2018 start 28 minutes ago

 README.md Initial commit 43 minutes ago

README.md



Arduino

Lectures on learning Arduino from scratch to the advanced level in iot environment.

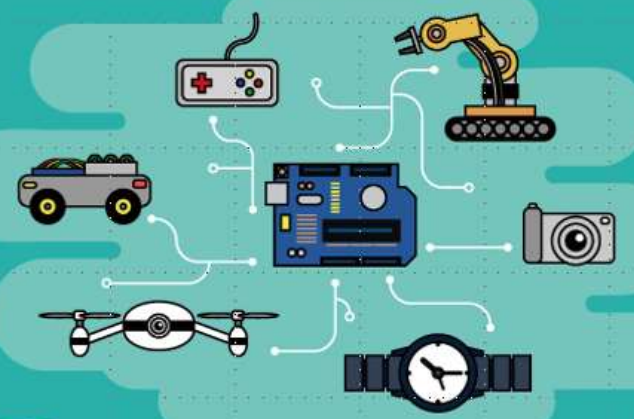




아두이노 **상급** 키트

High-Level Kit for Arduino

LEVEL 03



당신의 상상을
실현해줄
아두이노스토리

http://arduinostory.com/goods/goods_view.php?goodsNo=1000000306

상급키트 구성품

1 1EA 아두이노 우노 R3 DIP 아두이노 우노 R3 (DIP) 호환보드 기본 메인보드입니다.	2 1EA 9V 배터리 홀더 9V 배터리를 연결하여 아두이노에 외부전원을 공급할 수 있습니다.	3 1EA 7세그먼트 4채널 7세그먼트가 4개 연결된 형태의 부품입니다. 총 12개의 핀을 사용합니다.	4 1EA 7세그먼트 1채널 공통 음극 7세그먼트 시계나 점수 등의 숫자를 표현 할 때 많이 사용됩니다.
5 1EA 74HC595N 기본 메인보드입니다. 74HC595N LED, 드로메트릭스, NFD 제어 IC 입니다.	6 1EA 65핀 점퍼 와이어 브레드보드에 연결할 때 사용하는 65핀 점퍼와이어 입니다.	7 1EA 무지개 점퍼선 F-M 20cm M타입과 F타입이 양쪽으로 달린 무지개 점퍼선입니다.	8 1EA 투명 부품 케이스 대,소 키트 구성품을 담을 수 있는 투명 부품 케이스입니다.
9 1EA 가변저항10K 물리변 저항값이 바뀝니다. (0~10KΩ)	10 1EA 1602 I2C LCD 아두이노 16x2 I2C LCD 모듈입니다. LCD입니다.	11 1EA 저항 100, 220, 330, 1K, 2K, 4.7K, 10K, 47K, 100K	12 1EA 브레드 보드 830울 브레드 보드 830울(봉무형) 센서 테스트나, 회로 프로토타입을 작성할 때 사용됩니다.

13 1EA 수동부저 아두이노의 tone함수를 통해 소리를 내는 부저입니다.	14 6EA 택트스위치 (12x12x7) 스위치를 누르고 있을 경우만 ON됩니다.	15 3EA 택트스위치 캡 (피랑,노랑,초록,빨강,하양) 택트스위치를 사용할 때 스위치간의 구분을 할 수 있습니다.	16 3EA 조도센서 빛을 감지하거나 빛의 밝기를 아날로그로 출력해주는 CDS 센서입니다.
17 5EA LED 5mm (빨강,노랑,초록,하양,파랑) 기본으로 사용되는 LED입니다. 동작전압 : 2.2~2.4V 사용전류 : 20mA 미만	18 1EA 헤더핀 1x40/2.54mm 핀 간격은 2.54mm이며 헤더핀의 길이는 약 1.15cm입니다.	19 1EA USB케이블 50cm PC와 아두이노 우노 보드를 연결하여 프로그램을 다운로드 할 때 사용합니다.	20 1EA 저항값 카드 저항값을 쉽게 확인 할 수 있는 카드입니다. 사이즈 : 60mm x 50mm
21 1EA 능동부저 Signal 단자가 HIGH 일 때 약 2.5kHz의 음이 발생됩니다.	22 1EA 5V 1채널 릴레이 모듈 아두이노의 디지털 핀과 모듈 하단의 IN 핀들을 연결해 릴레이를 제어할 수 있는 모듈입니다.	23 1EA 8x8 도트 매트릭스 모듈 LED로 다양한 연출을 할 수 있습니다.	24 1EA 4x4 16 키패드 모듈 16개의 버튼을 사용할 수 있습니다.

아두이노 키트(Kit) : Part-2

<div>25</div> <div>1EA</div> <div></div> <div>무선 리모콘 키트</div> <div>핵파선을 사용해서 리모콘 기능을 구현할 수 있습니다.</div>	<div>26</div> <div>2EA</div> <div></div> <div>가열기 센서 스위치</div> <div>센서의 가열기에 따라 스위치 역할을 합니다.</div>	<div>27</div> <div>1EA</div> <div></div> <div>or</div> <div>사운드 센서 모듈</div> <div>아두이노와 호환되는 사운드센서 모듈입니다.</div>	<div>28</div> <div>1EA</div> <div></div> <div>불꽃 센서</div> <div>근거리 화재, 불꽃을 감지하는 센서입니다.</div>	<div>37</div> <div>1EA</div> <div></div> <div>DC 5V 스텝 모터</div> <div>28BYJ 28BYJ48 스텝 모터 중 저렴한 편에 속하는 모델입니다. 5개의 핀을 사용합니다.</div>	<div>38</div> <div>1EA</div> <div></div> <div>DS1302 RTC 모듈</div> <div>아두이노 등 마이크로컨트롤러에서 사용이 가능합니다.</div>	<div>39</div> <div>1EA</div> <div></div> <div>아두이노 우노 프로토 쉼드</div> <div>UNO 보드에서 회로를 간단히 짜기 위해 보드 위에 얹어 사용하는 쉼드입니다.</div>	<div>40</div> <div>1EA</div> <div></div> <div>3축 가속도 센서 모듈</div> <div>가속도를 측정할 수 있는 센서입니다.</div>
<div>29</div> <div>1EA</div> <div></div> <div>모터 드라이버 모듈</div> <div>ULN2003 스텝 모터 드라이버 모듈 5V ~ 12V를 사용할 수 있습니다.</div>	<div>30</div> <div>1EA</div> <div></div> <div>LM35 온도 센서</div> <div>온도를 아날로그 값으로 출력합니다.</div>	<div>31</div> <div>1EA</div> <div></div> <div>수위 센서 모듈</div> <div>센서가 액체에 잠긴 정도를 아날로그 값으로 출력합니다.</div>	<div>32</div> <div>1EA</div> <div></div> <div>SG90 서보모터</div> <div>Vcc, GND, 신호선, 총 3개의 핀이 있습니다. 로봇팔이나 자동차, 비행기 조종에 사용됩니다.</div>	<div>41</div> <div>1EA</div> <div></div> <div>5V DC모터</div> <div>5V DC모터</div>	<div>42</div> <div>1EA</div> <div></div> <div>인체 감지 센서 모듈</div> <div>핵파선을 이용해 움직임 감지하는 센서입니다. 오선이 감지되면 HIGH 신호를 출력합니다.</div>	<div>43</div> <div>5EA</div> <div></div> <div>다이오드 1N4001</div> <div>다이오드 1N4001</div>	<div>44</div> <div>5EA</div> <div></div> <div>세라믹 캐패시터 (22pF)</div> <div>세라믹 캐패시터 (22pF)</div>
<div>33</div> <div>1EA</div> <div></div> <div>초음파 거리 센서 모듈</div> <div>5V를 사용하여 인식 거리는 2cm에서 500cm입니다.</div>	<div>34</div> <div>1EA</div> <div></div> <div>조이스틱 모듈</div> <div>기본적으로 조이스틱 모듈은 두개의 가변저항이 서로 수직으로 회전하는 형태로 되어있습니다.</div>	<div>35</div> <div>1EA</div> <div></div> <div>온습도 센서 모듈</div> <div>아두이노 온습도 센서중 가장 대중적으로 사용되는 DHT11 디지털 센서입니다.</div>	<div>36</div> <div>1EA</div> <div></div> <div>RGB LED 모듈</div> <div>RGB LED 모듈로 RGB LED 세개를 하나로 묶은 상품입니다.</div>	<div>45</div> <div>5EA</div> <div></div> <div>세라믹 캐패시터 (1uF)</div> <div>세라믹 캐패시터 (1uF)</div>	<div>46</div> <div>5EA</div> <div></div> <div>트랜지스터 2N2222</div> <div>트랜지스터 2N2222</div>	<div>47</div> <div>5EA</div> <div></div> <div>트랜지스터 BC547</div> <div>트랜지스터 BC547</div>	<div>48</div> <div>5EA</div> <div></div> <div>트랜지스터 BC557</div> <div>트랜지스터 BC557</div>
<div>49</div> <div>2EA</div> <div></div> <div>전해 캐패시터 (50V 10uF)</div> <div>전해 캐패시터 (50V 10uF)</div>	<div>50</div> <div>2EA</div> <div></div> <div>전해 캐패시터 (50V 100uF)</div> <div>전해 캐패시터 (50V 100uF)</div>	<div></div>					

[참고 : 저항 값 읽기]



Color	First	Second	Third	Multiplier	Tolerance
Black	0	0	0	x1	
Brown	1	1	1	x10	1%
Red	2	2	2	x100	2%
Orange	3	3	3	x1000	
Yellow	4	4	4	x10 000	
Green	5	5	5	x100 000	0,50%
Blue	6	6	6	x1 000 000	0,25%
Violette	7	7	7	x10 000 000	0,10%
Gray	8	8	8		
White	9	9	9		
Silver				x0,01	10%
Gold				x0,1	5%

