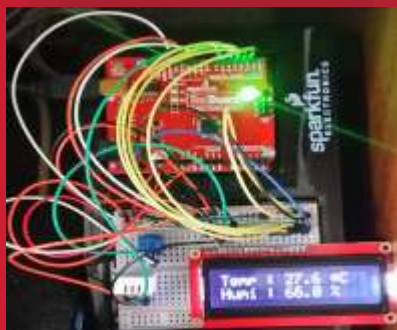




# Arduino-basic

## [wk05]

## LED - II



Learn how to code Arduino from scratch

Comsi, INJE University

1<sup>st</sup> semester, 2022

Email : [chaos21c@gmail.com](mailto:chaos21c@gmail.com)



# My ID (ARnn, github repo)

- [AR01 김정헌]
- [AR02 유석진]
- [AR03 김기덕]
- [AR04 강대진]
- [AR05 김성우]
- [AR06 김창연]
- [AR07 김창욱]
- [AR08 김태화]
- [AR09 박세훈]
- [AR10 박신영]

- [AR11 박제홍]
- [AR12 이승무]
- [AR13 이승준]
- [AR14 이재하]
- [AR15 이준희]
- [AR16 이현준]
- [AR17 임태형]
- [AR18 정동현]
- [AR19 정희서]
- [AR20 이한글]
- [AR21 황혁준]
- [AR22 김동영]

# wk04 : Practice-03 : ARnn\_Rpt03

## ◆ [Target of this week]

- Complete your works
- Save your outcomes
- Upload all in github.

**Upload 폴더 명 : ARnn\_Rpt03**

### 제출할 파일들

- ① **ARnn\_2led.ino**
- ② **ARnn\_4led.fzz**
- ③ **ARnn\_4led.ino**
- ④ **ARnn\_RGB.ino**
- ⑤ **ARnn\_RGB\_Y.png**



# 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

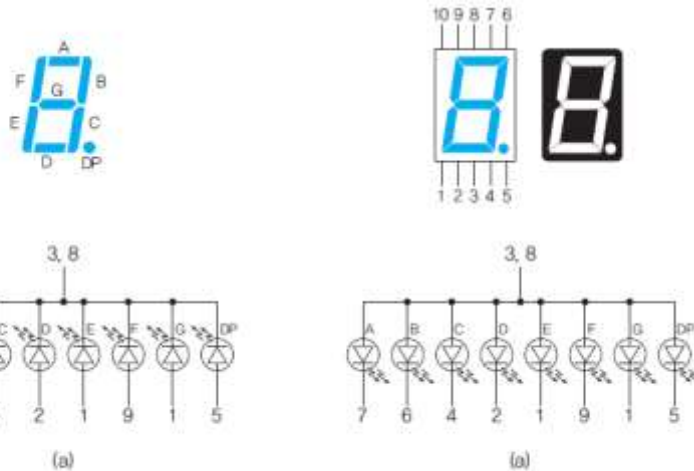
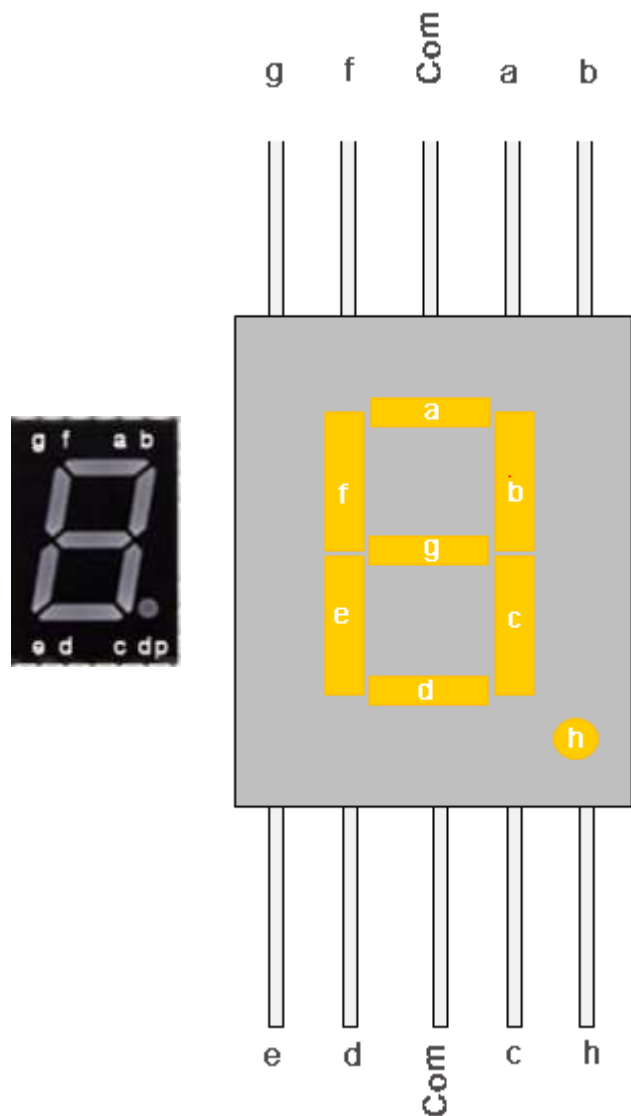


그림 4.4 Common Cathode 형(a)와 Common Anode 형(b)

표 4.1 Common Cathode FND 표시

캐소드 공통 7-세그먼트 한 자리 제어 방법										7-Seg. 출력 내용
Q0	DP	G	F	E	D	C	B	A	16진수	
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. (.)

# 4.5 FND 제어



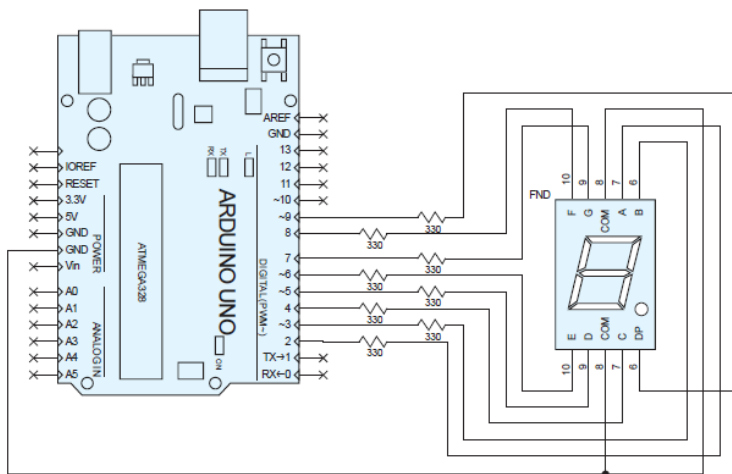
	h	g	f	e	d	c	b	a	hex value
0	0	0	1	1	1	1	1	1	3F
1	0	0	0	0	0	1	1	0	06
2	0	1	0	1	1	0	1	1	5B
3	0	1	0	0	1	1	1	1	4F
4	0	1	1	0	0	1	1	0	66
5	0	1	1	0	1	1	0	1	6D
6	0	1	1	1	1	1	0	1	7D
7	0	0	0	0	0	1	1	1	07
8	0	1	1	1	1	1	1	1	7F
9	0	1	1	0	1	1	1	1	6F
A	0	1	1	1	0	1	1	1	77
b	0	1	1	1	1	1	0	0	7C
c	0	0	1	1	1	0	0	1	39
d	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

## EX 4.4

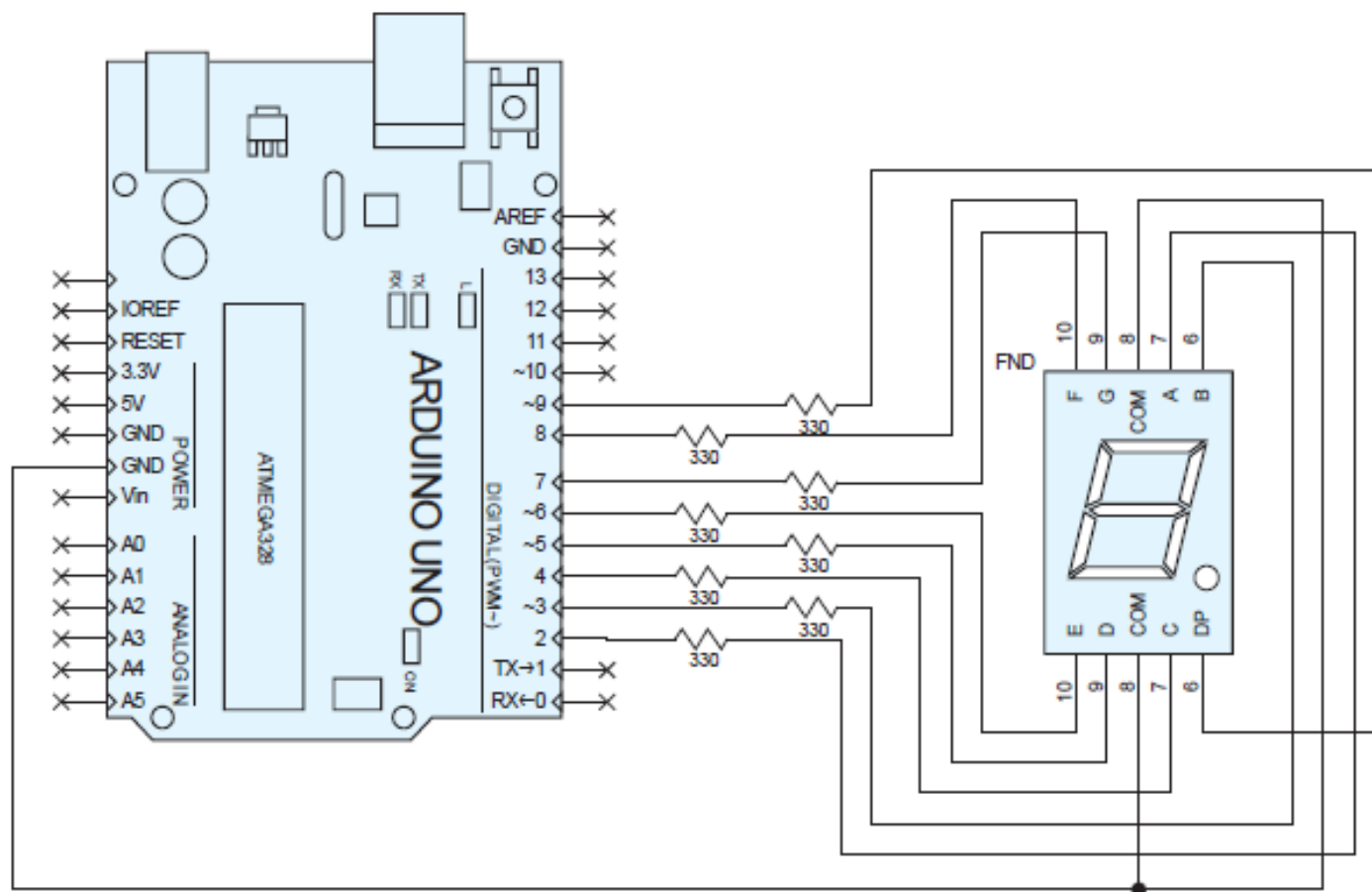
## FND 제어 (1/3)

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

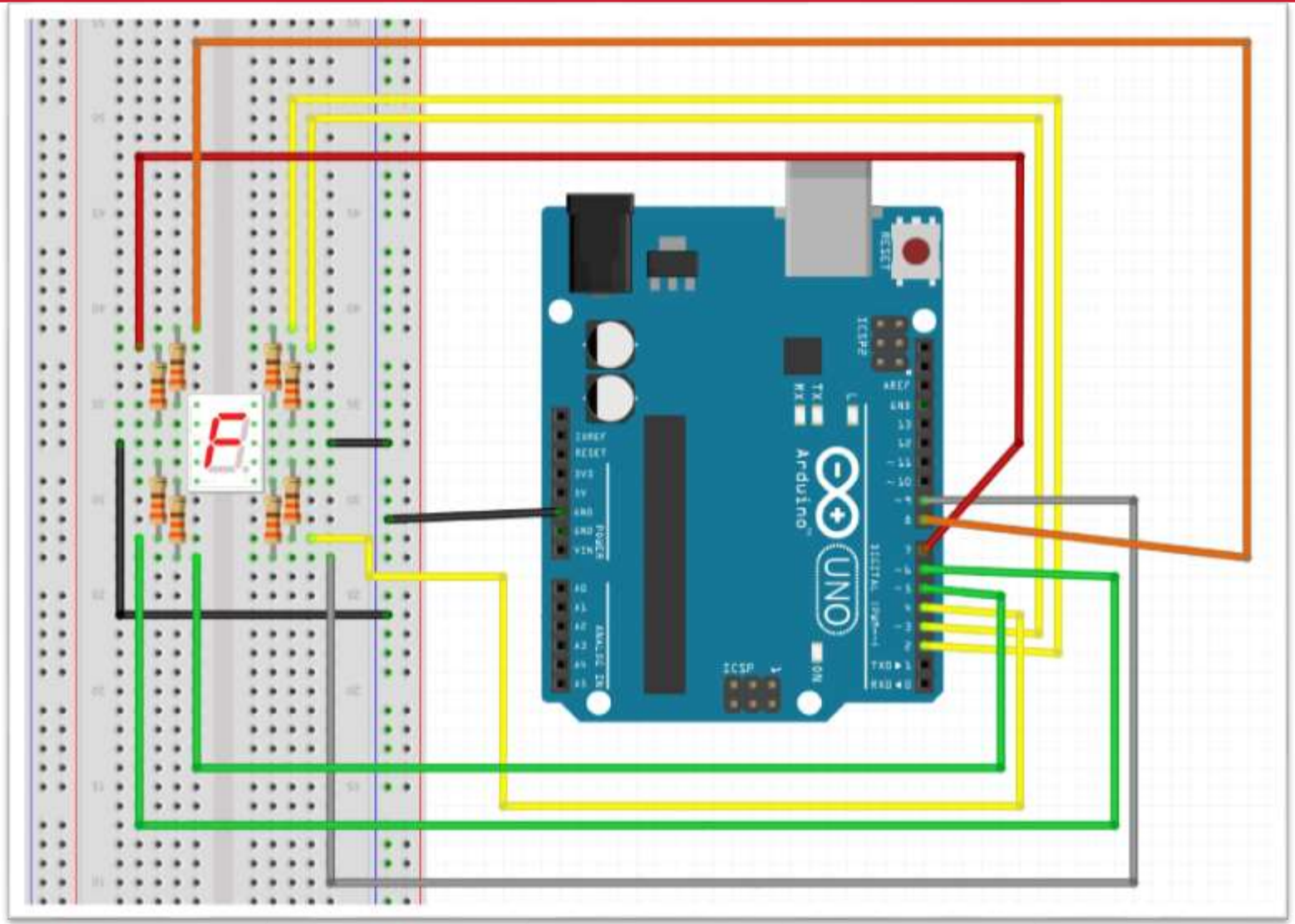
- Hardware**
1. Common Cathode형 FND는 그림 4.2의 (a)와 같이 3번과 8번핀이 Cathode 핀으로 함께 연결되어 있다. 즉 FND의 3번과 8번핀을 GND에 연결하고 나머지 핀들에 HIGH신호를 주어 FND에 숫자를 표시한다.
  2. GND에 연결되는 3번과 8번핀을 제외한 나머지 핀들에는 FND 내의 LED의 전류를 제한하기 위해 330 또는 220  $\Omega$  저항을 연결한다.
  3. 원하는 숫자를 표시하기 위해선 2~9번핀에 표 4.1을 참고하여 신호를 출력한다.



# 4.5.2 FND 제어







Fritzing 으로 회로를 디자인하고  
[ARnn\\_fnd.fzz](#) 로 저장해서 제출.

## 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.

## 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.

**실습 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 }
```

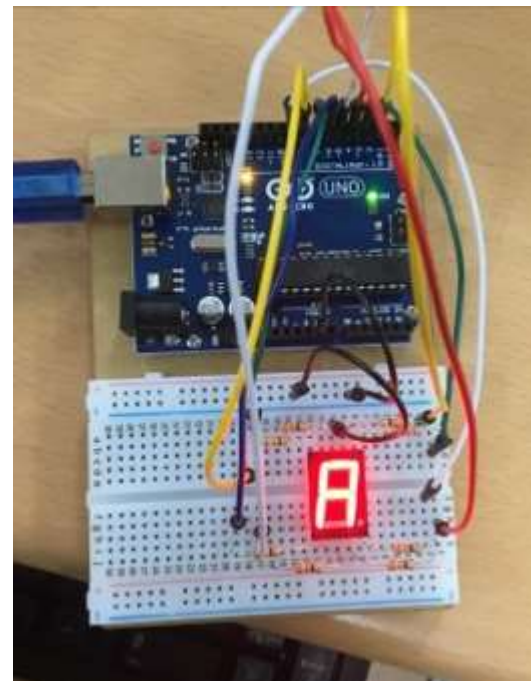
## EX 4.4 FND 제어 (3/3)

**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

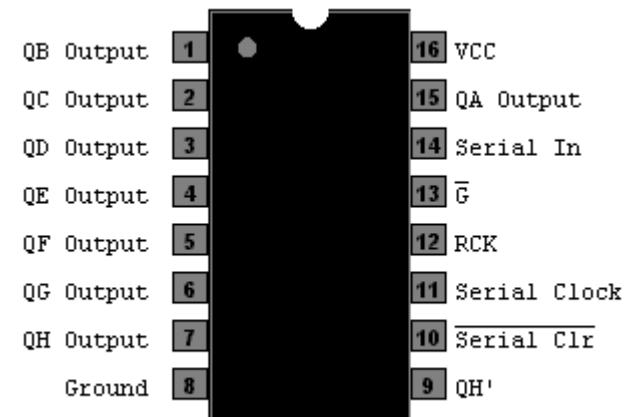
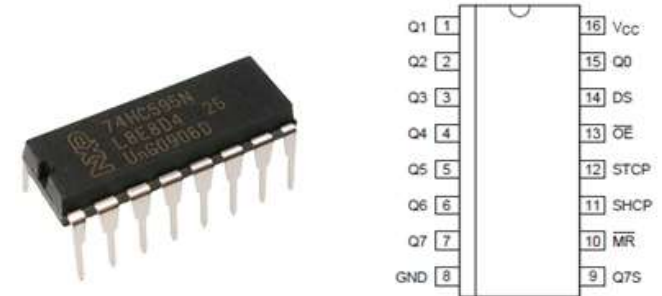
로 저장해서 제출. (아두이노 회로를 포함해서 촬영)



# 4.6 FND 제어 : 74595 IC (74HC595N)

## 74595 IC

- ✓ 직렬 신호로 입력된 데이터를 병렬 신호로 변환
- ✓ FND의 8개의 LED를 켜기 위한 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**

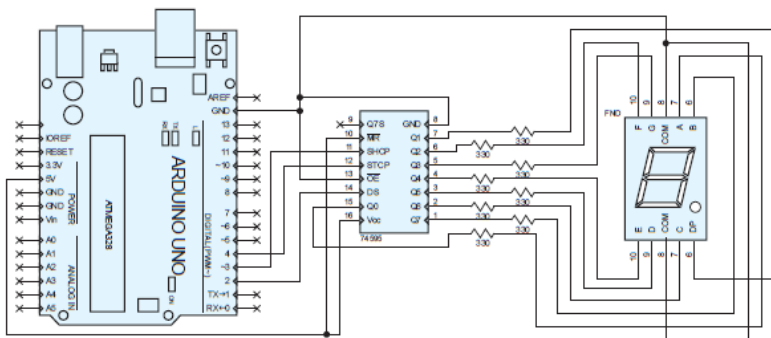
- ✓ DS, SHCP, STCP 세 핀으로 FND 제어
- ✓ 동작 순서
  1. STCP (12) 에 'LOW' 신호 입력
  2. SHCP(11)의 클럭에 맞춰 DS(14)로 데이터 전송
  3. 전송 후, STCP에 'HIGH' 신호를 주어 출력핀으로 신호를 출력
- ✓ Shiftout( ) 함수로 동작 시킴.

# 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.

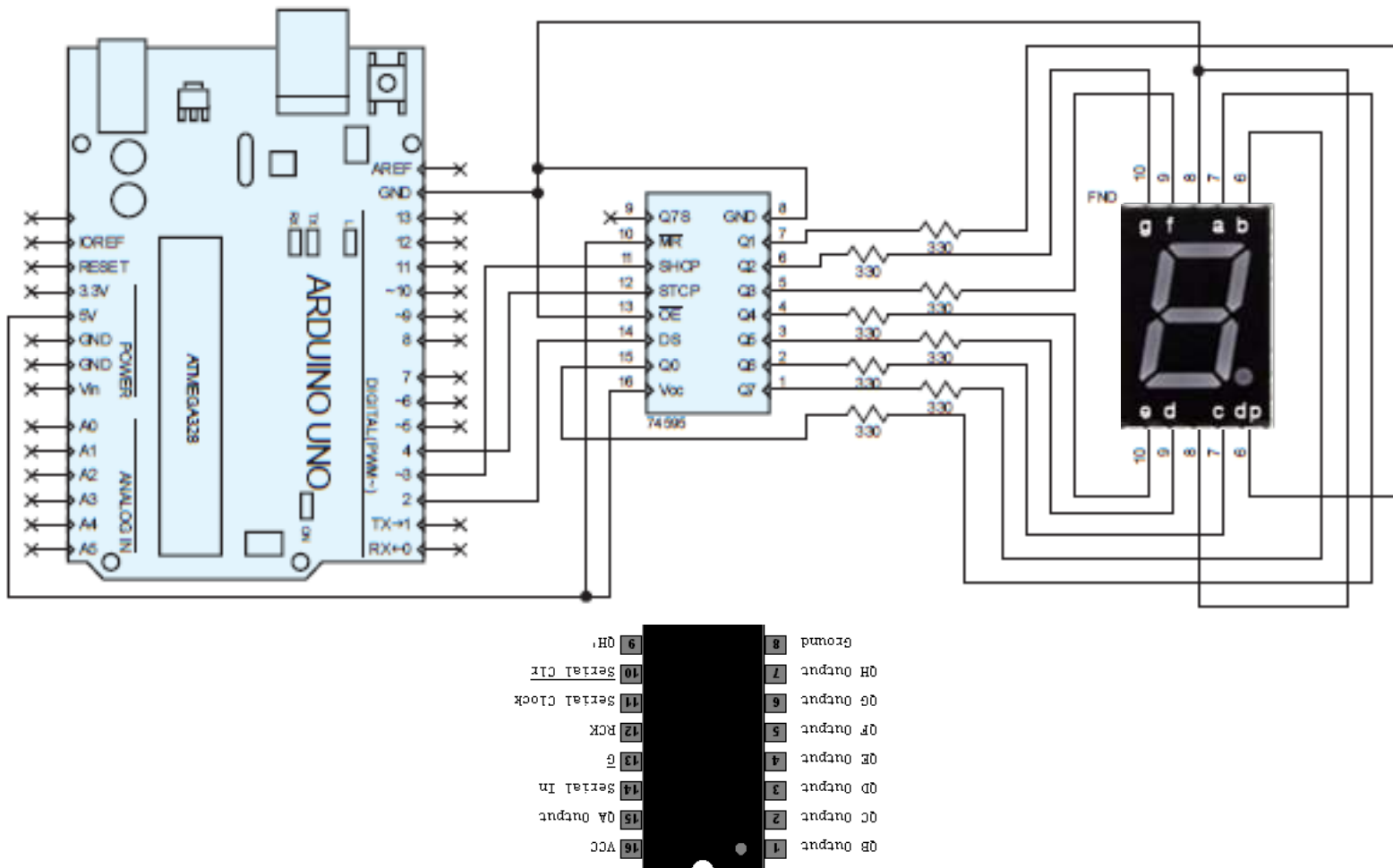
- Hardware**
1. 예제 4.4.1과 동일한 동작을 하지만 **Arduino의 입출력 핀을 절약하기 위해 74595 IC를** 중간에 연결한다.
  2. **Arduino에서는 2, 3, 4 세 개의 핀을 이용하여 74595 IC로 신호를 출력**한다.  
각 핀을 Arduino에 연결한다.
  3. 74595 IC의 (MR)<sup>-</sup>핀과 Vcc 핀에는 5V를 연결하고 (OE)<sup>-</sup>와 GND핀은 Arduino의 GND에 연결한다.
  4. 74595 IC에서는 DS, SHCP, STCP 핀으로부터 입력된 신호를 이용하여 Q0~Q7 핀에 신호를 출력한다. Q0~Q7 핀을 FND의 Anode 핀에 연결한다.
  5. FND의 Cathode 핀인 3번과 8번핀은 GND에 연결한다.



# 4.6.2.1 FND 제어 : 74595 IC

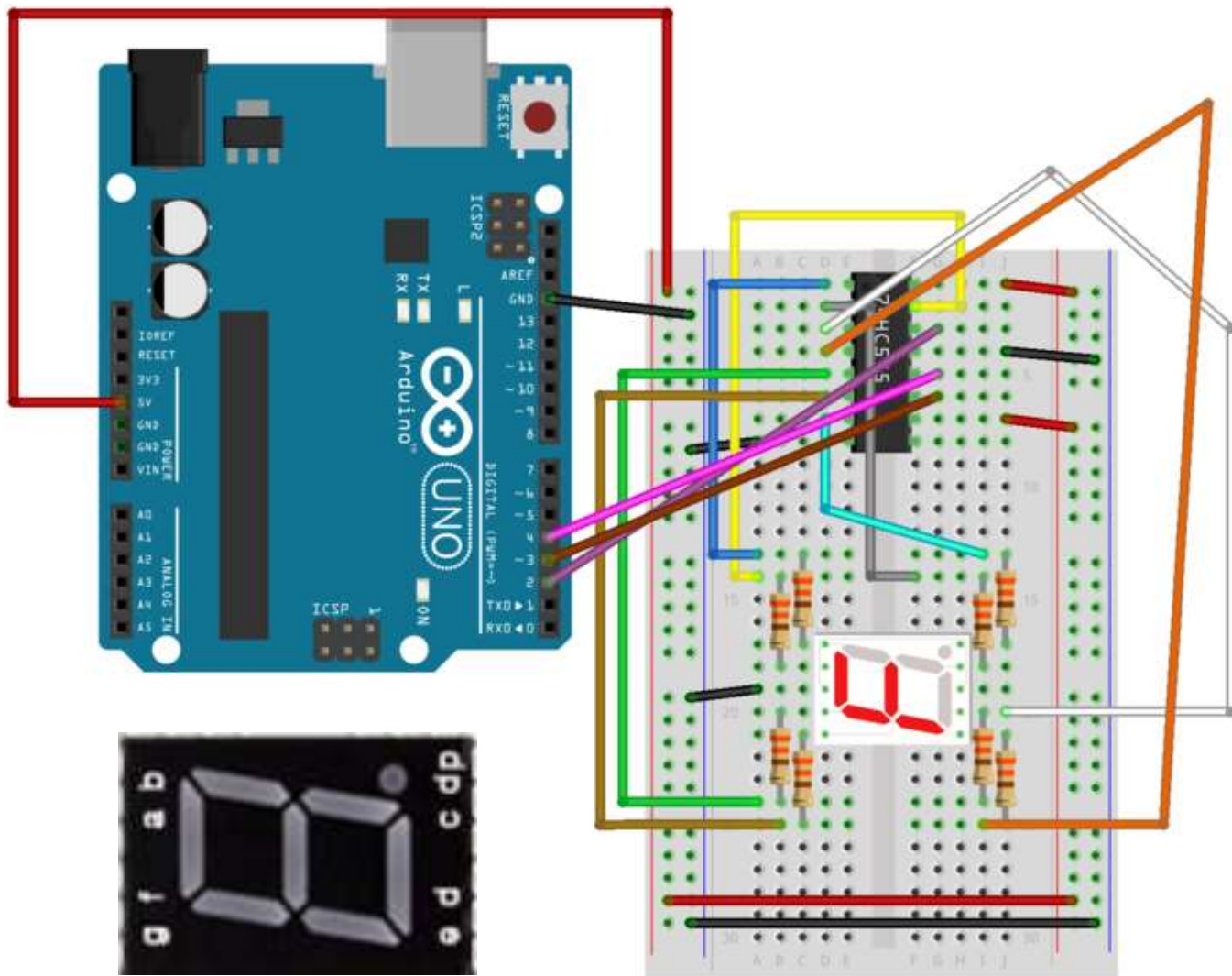
EX 4.4.2

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	Serial Clr
Ground	8	9	QH'

**11 → D3**

**12 → D4**

**14 → D2**

**15 → a**

**1  $\rightarrow$  b**

**2 → c**

**3 → d**

**4 → e**

**5 → f**

**6 → g**

**7 → h**

## 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'.

출력 값에는 실제 출력할 데이터를 써 준다. 이 때 데이터는 8비트 즉 2진수 8자리의 숫자를 갖는다.

## 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 SHCP핀을 Arduino의 3번핀에 연결
24 int stcp = 4; // 74595의 STSP핀을 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 }

```

```

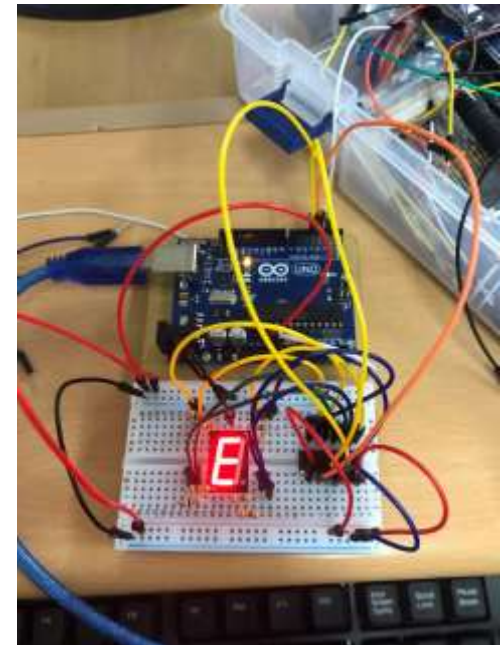
35 void loop()
36 {
37 // k값을 0~9로 변화시킨다.
38 for(int k = 0; k <= 9; ++k){
39 fndDisplay74595(k); // k값을 출력한다
40 delay(1000); // 1초간 지연시킨다.
41 };
42 }
43
44 // LED 점등
45 void fndDisplay74595(int displayValue){
46 // STCP에 LOW 신호를 보내서 74595로 데이터전송을 시작한다.
47 digitalWrite(stcp, LOW);
48 // shiftOut 명령어로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

```

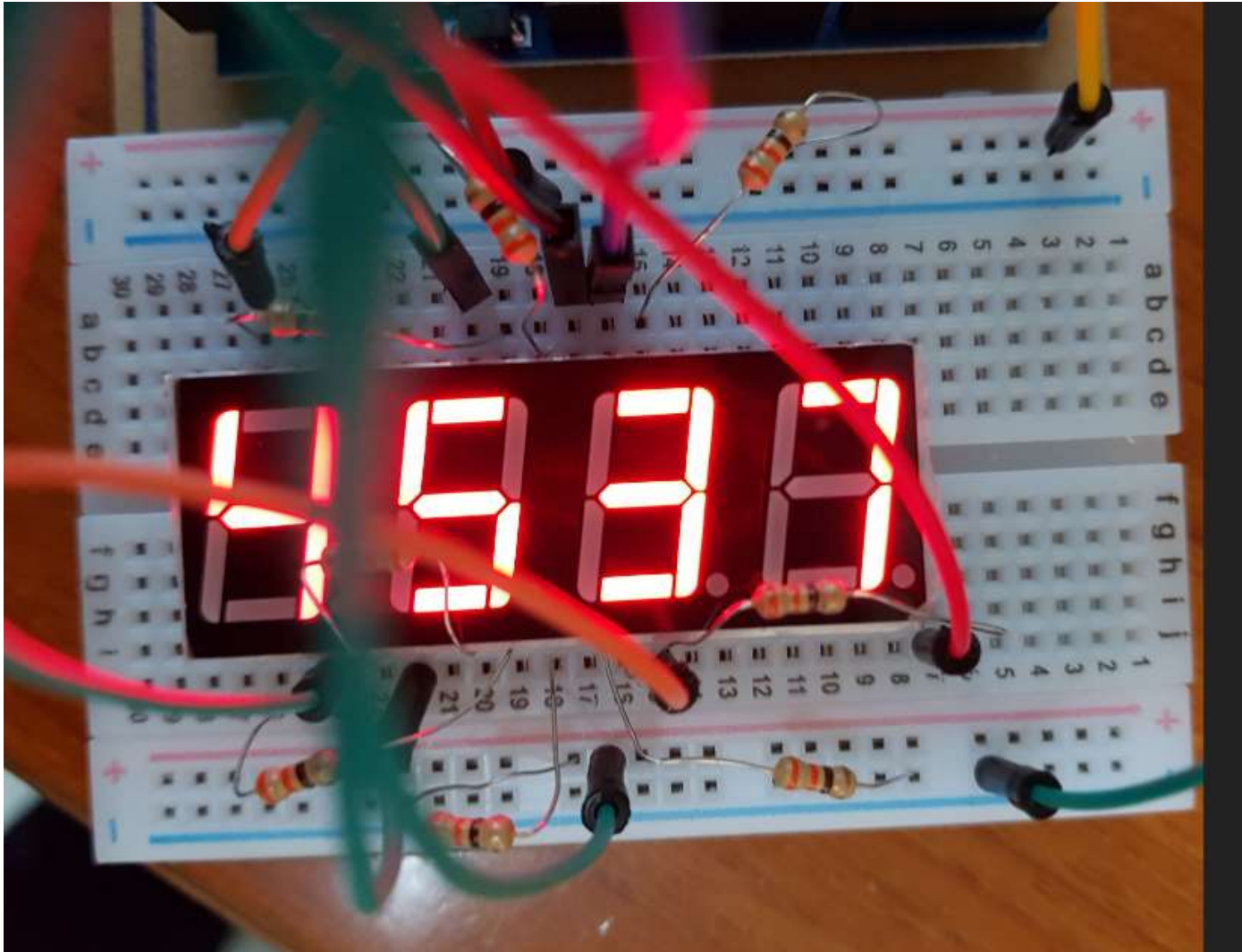
## EX 4.4.2 74595를 이용한 FND 제어 (3/3)

**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_74595_E.png`  
로 저장해서 제출. (아두이노 회로를 포함해서 촬영)



## 4.7. 4-digit FND 제어





# 4.7 4-digit 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

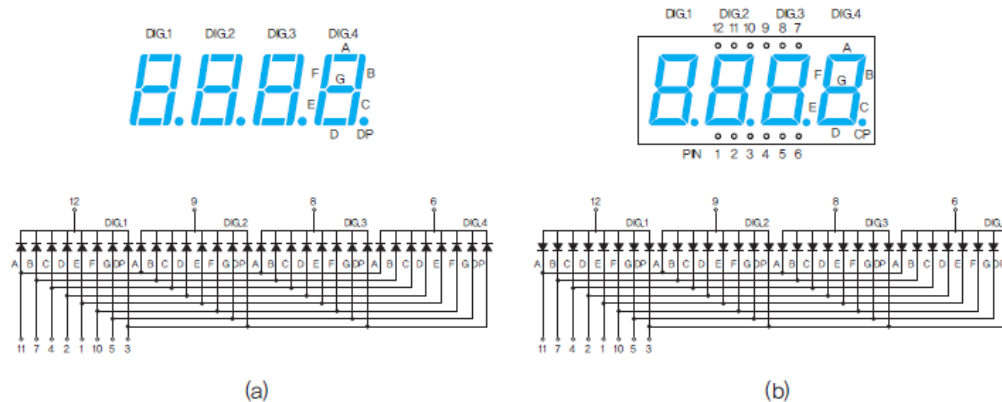


그림 4.6 4-digit FND와 내부 회로. Common Cathode (a), Common Anode (b)



그림 4.7 실험에 사용할 4-digit FND

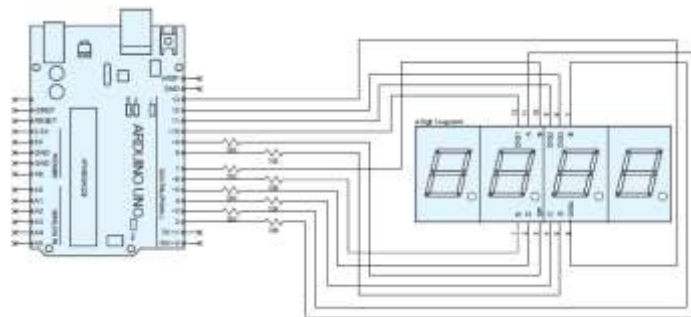


# 4.7.1 4-digit FND 제어

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

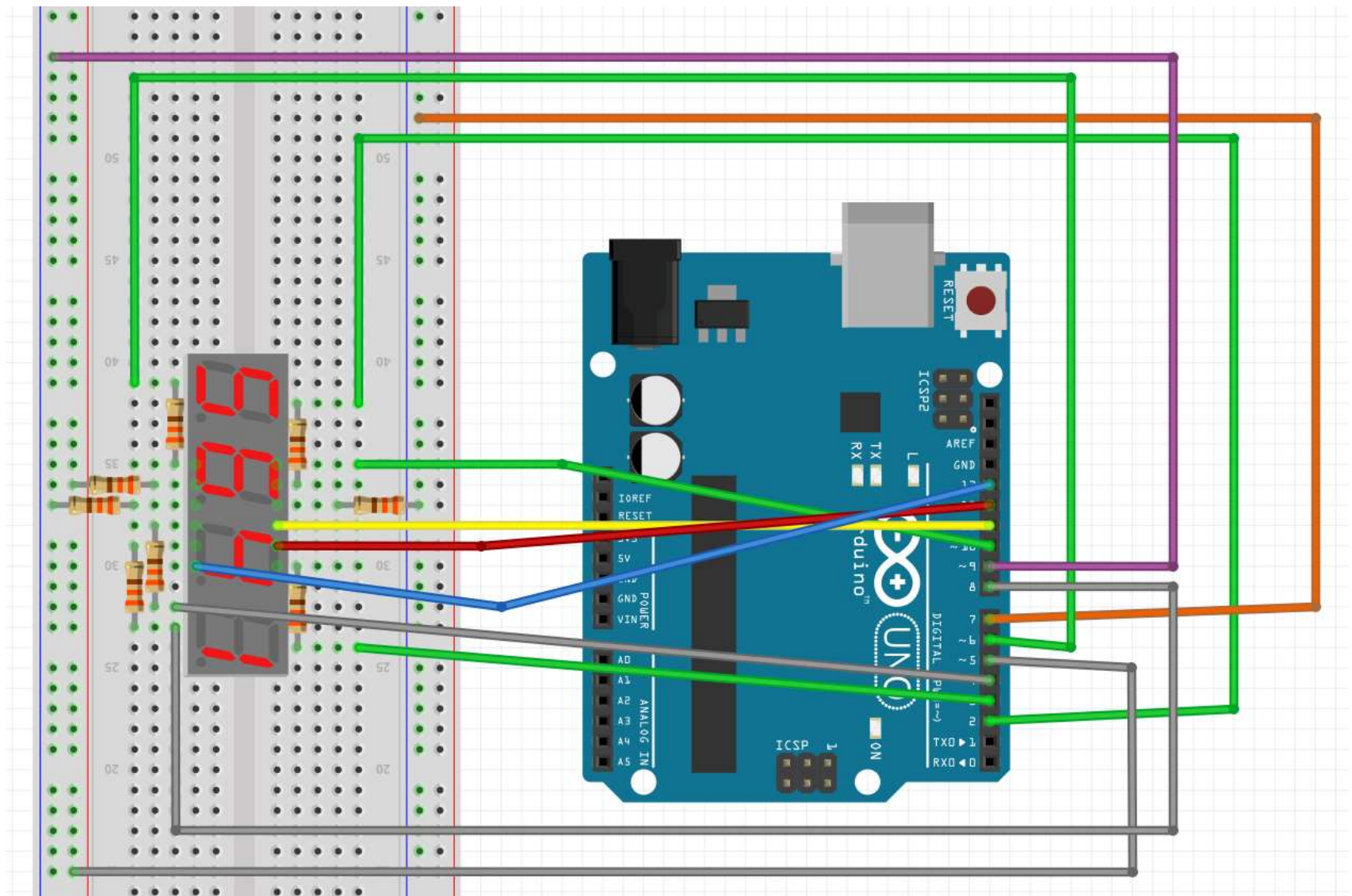
**실습목표** Common Cathode 4-digit FND를 이용하여 0000~9999까지 1초 간격으로 증가하는 스케치를 작성해 보자.

- 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 330 $\Omega$ 저항을 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-digit FND 제어 – circuit2



Fritzing 으로 회로를 디자인하고  
ARnn\_4digit.fzz 로 저장해서 제출.



# [Practice]

## ◆ [wk05]

- **Arduino LED-II.**
- **Complete your project**
- **Submit folder : ARnn\_Rpt04**

# wk05 : Practice-04 : ARnn\_Rpt04

## ◆ [Target of this week]

- Complete your works
- Save your outcomes
- Upload all in github.

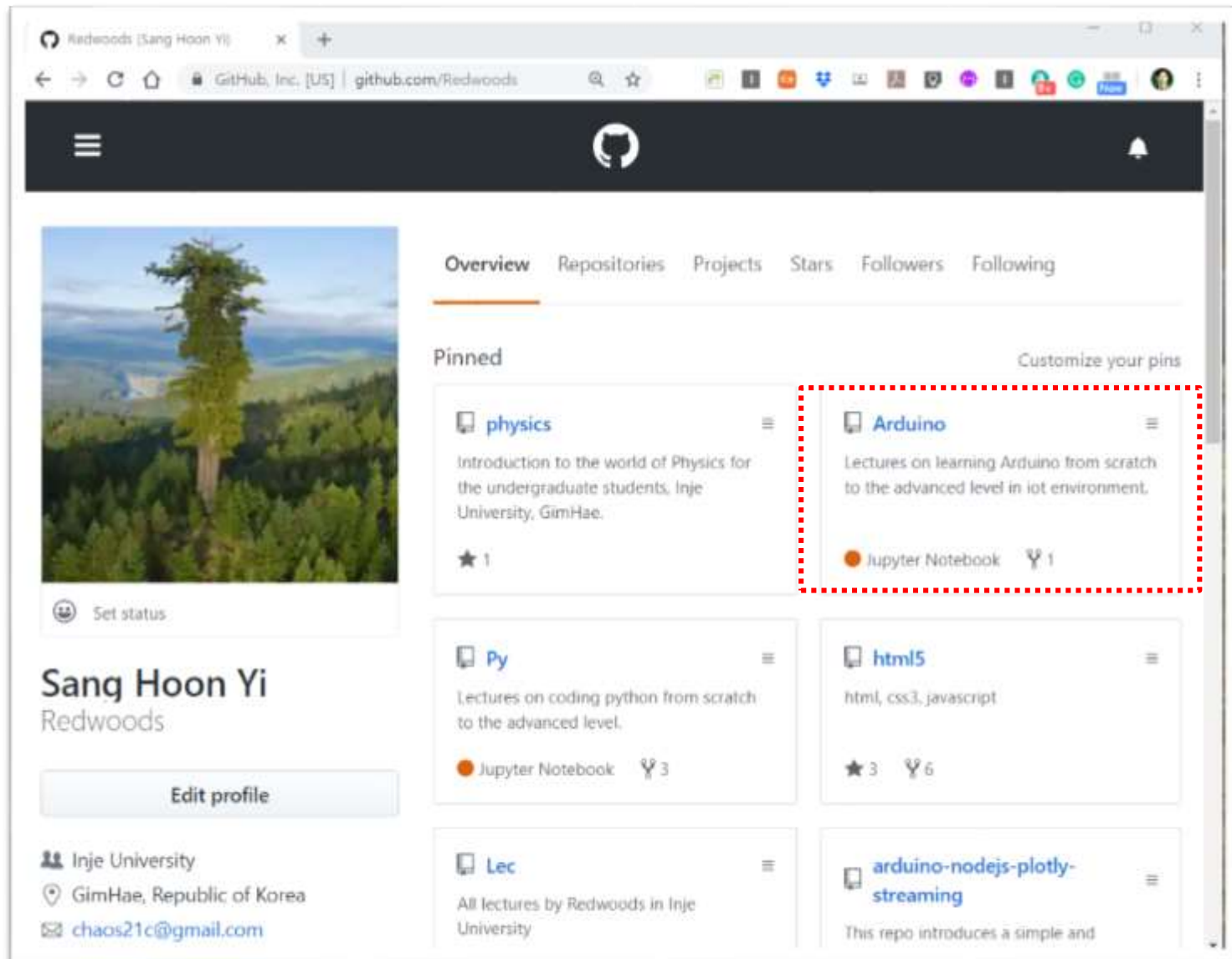
제출폴더명 : **ARnn\_Rpt04**

### - 제출할 파일들

- ① **ARnn\_fnd.fzz**
- ② **ARnn\_A.png**
- ③ **ARnn\_74595\_E.png**
- ④ **ARnn\_4digit.fzz**
- ⑤ **All \*.ino**

## ● References & good sites

- ✓ <http://www.nodejs.org/ko> Node.js
- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.w3schools.com> By w3schools
- ✓ <http://www.github.com> GitHub
- ✓ <http://www.google.com> Googling



Redwoods (Sang Hoon Yi)

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

Overview Repositories Projects Stars Followers Following

Pinned Customize your pins

**physics**  
Introduction to the world of Physics for the undergraduate students, Inje University, GimHae.  
★ 1

**Arduino**  
Lectures on learning Arduino from scratch to the advanced level in iot environment.  
Jupyter Notebook 🍴 1

**Py**  
Lectures on coding python from scratch to the advanced level.  
Jupyter Notebook 🍴 3

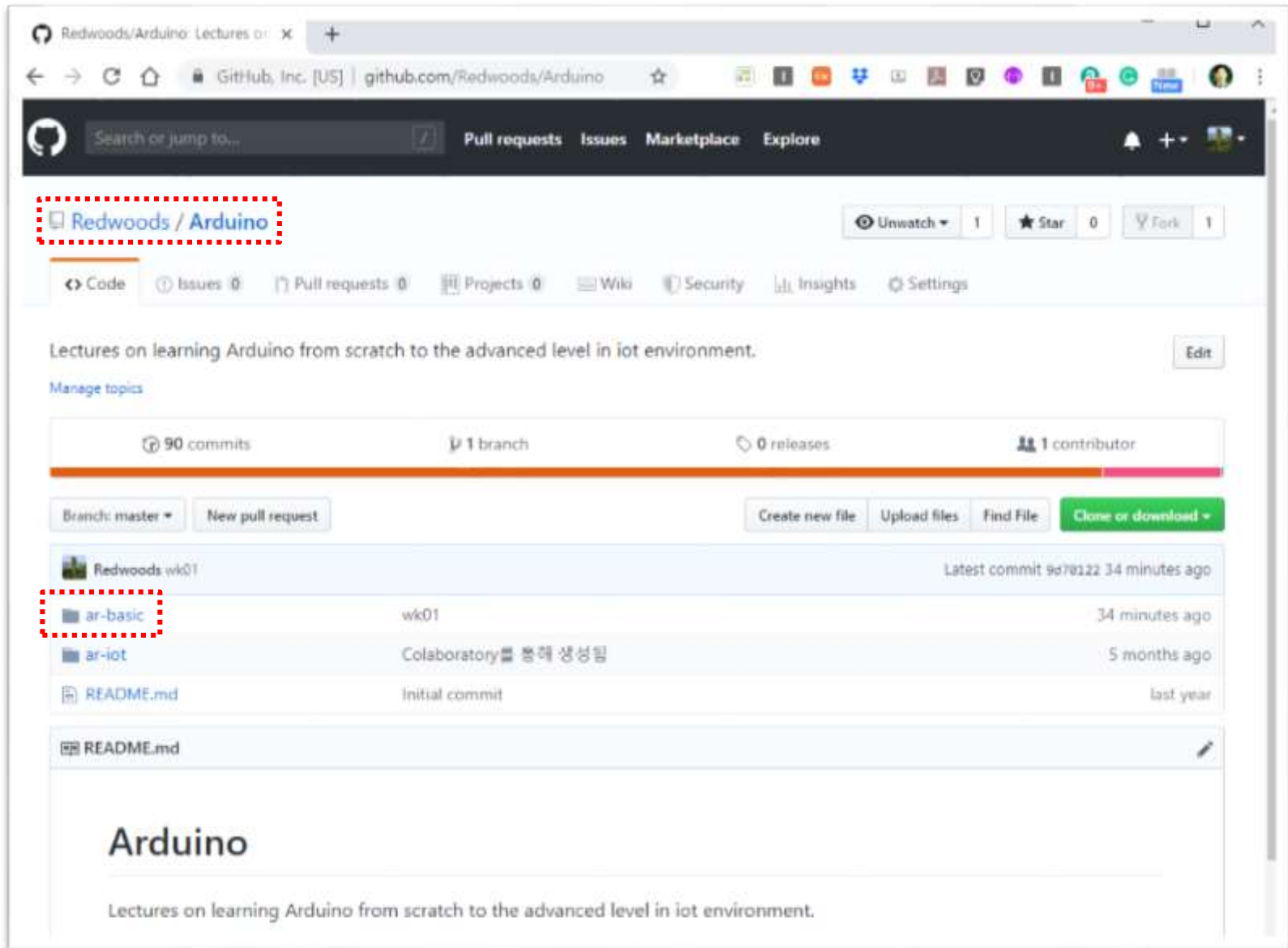
**html5**  
html, css3, javascript  
★ 3 🍴 6

**Lec**  
All lectures by Redwoods in Inje University

**arduino-nodejs-plotly-streaming**  
This repo introduces a simple and

Sang Hoon Yi  
Redwoods  
Edit profile

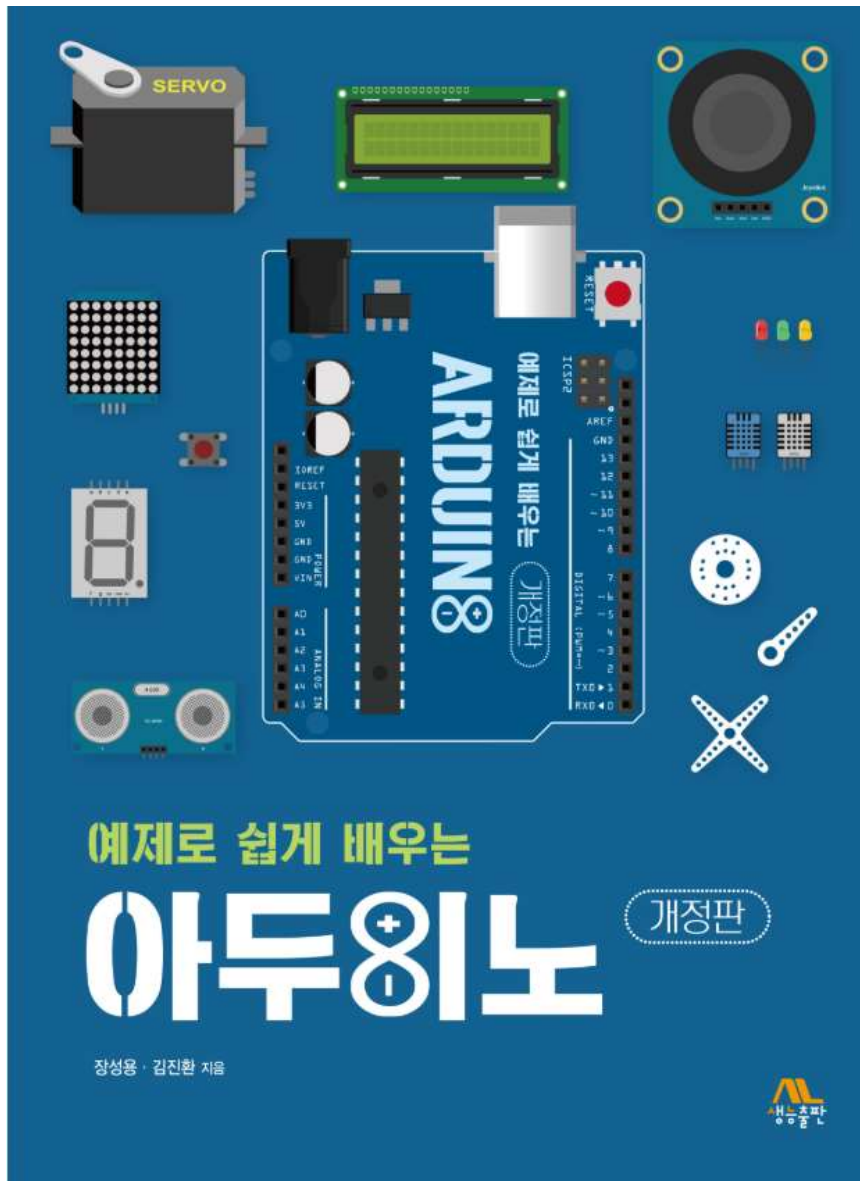
Inje University  
GimHae, Republic of Korea  
chaos21c@gmail.com



The screenshot shows the GitHub repository page for **Redwoods/Arduino**. The repository is described as "Lectures on learning Arduino from scratch to the advanced level in iot environment." It has 90 commits, 1 branch, 0 releases, and 1 contributor. The repository is currently on the **master** branch. The file list shows the following files and their commit history:

File	Commit	Time
ar-basic	wk01	34 minutes ago
ar-iot	Colaboratory를 통해 생성됨	5 months ago
README.md	Initial commit	last year

The **ar-basic** file is highlighted with a red dashed box. Below the file list, the **README.md** content is visible, starting with the title **Arduino** and the description "Lectures on learning Arduino from scratch to the advanced level in iot environment."

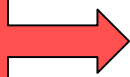




# 아두이노 키트(Kit)



**web**



<https://www.devicemart.co.kr/goods/view?no=12170416>



## 아두이노 레벨업키트(골드) 구성품

						
아두이노 UNO	USB 케이블	830핀 브레드보드	미니 브레드보드	점퍼와이어 세트	듀폰케이블 M/F	듀폰케이블 M/M
						
저항 220Ω	저항 1KΩ	저항 10KΩ	가변저항 10KΩ	빨강 LED	녹색 LED	파랑 LED
						
노랑 LED	RGB LED (CA)	RGB LED 모듈	1digit FND (CA)	4digit FND (CA)	8x8 도트 매트릭스	택트 스위치
						
택트 스위치 캡	눌 스위치	리드 스위치 센서	4x4키 매트릭스	5V 릴레이 모듈	조이스틱 모듈	수위 센서
						
온도센서 LM35	터미스터	온습도센서	CDS 조도센서	불꽃감지센서	적외선 수신기	IR 리모컨

# 아두이노 키트(Kit) : Part-2

						
TCRT5000 적외선 센서	인체감지센서 모듈	사운드센서	능동부저	수동부저	초음파센서	I2C 1602 LCD 모듈
						
서보모터	스텝모터	스텝모터드라이버	RFID 수신 모듈	RFID 카드	RFID 태그	DS1302 RTC 모듈
						
1N4001 다이오드	2N2222 트랜지스터	74HC595	1x40 핀헤더	9V 배터리 스냅	아크릴 고정판	

■ 아두이노 UNO × 1	■ USB 케이블 × 1	■ 830핀브레드보드 × 1	■ 미니 브레드보드 × 1	■ 점퍼와이어세트 × 1
■ 듀폰케이블 × 80 (M/F,M/M)	■ 저항 × 30	■ 가변저항 × 1	■ LED × 20	■ RGB LED × 1
■ RGB LED 모듈 × 1	■ 1digit FND(CA) × 1	■ 4digit FND(CA) × 1	■ 8×8도트 매트릭스 × 1	■ 탭스위치 × 5
■ 탭스위치 캡 × 5	■ 볼스위치 × 1	■ 리드 스위치 센서 × 1	■ 4×4 키 매트릭스 × 1	■ 5V 릴레이 모듈 × 1
■ 조이스틱 모듈 × 1	■ 수위 센서 × 1	■ 온도센서 LM35 × 1	■ 써미스터 × 1	■ 온습도센서 × 1
■ CdS 조도센서 × 1	■ 불꽃감지센서 × 1	■ 적외선 수신기 × 1	■ IR 리모컨 × 1	■ TCRT5000 적외선 센서 × 1
■ 인체감지센서 모듈 × 1	■ 사운드센서 × 1	■ 능동부저 × 1	■ 수동부저 × 1	■ 초음파센서 × 1
■ I2C 1602 LCD 모듈 × 1	■ 서보모터 × 1	■ 스텝모터 × 1	■ 스텝모터드라이버 × 1	■ RFID 수신 모듈 × 1
■ RFID 카드 × 1	■ RFID 태그 × 1	■ DS1302 RTC 모듈 × 1	■ 1N4001 다이오드 × 1	■ 2N2222 트랜지스터 × 1
■ 74HC595 × 1	■ 1x40 핀헤더 × 1	■ 9V 배터리 스냅 × 1	■ 아크릴 고정판 × 1	