

# 예비보고서

## IEEE Code of Ethics

(출처: <http://www.ieee.org>)

We, the members of the IEEE, in recognition of the importance of our technologies in affecting the quality of life throughout the world, and in accepting a personal obligation to our profession, its members and the communities we serve, do hereby commit ourselves to the highest ethical and professional conduct and agree:

1. to accept responsibility in making decisions consistent with the safety, health and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;
2. to avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist;
3. to be honest and realistic in stating claims or estimates based on available data;
4. to reject bribery in all its forms;
5. to improve the understanding of technology, its appropriate application, and potential consequences;
6. to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;
7. to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others;
8. to treat fairly all persons regardless of such factors as race, religion, gender, disability, age, or national origin;
9. to avoid injuring others, their property, reputation, or employment by false or malicious action;
10. to assist colleagues and co-workers in their professional development and to support them in following this code of ethics.

위 IEEE 윤리헌장 정신에 입각하여 report를 작성하였음을 서약합니다.

학 부: 전자공학부

제출일: 2021-09-30

과목명: 논리회로실험

교수명: 박 성 진 교수님

분 반: 금 F

학 번: 201820814, 202021025

성 명: 윤 상 원, 안 준 영

$D_0 = !A_2 !A_1 !A_0$ $D_1 = !A_2 !A_1 A_0$ $D_2 = !A_2 A_1 !A_0$ $D_3 = !A_2 A_1 A_0$ $D_4 = A_2 !A_1 !A_0$ $D_5 = A_2 !A_1 A_0$ $D_6 = A_2 A_1 !A_0$ $D_7 = A_2 A_1 A_0$	<table><tr><th colspan="3">입력</th><th colspan="8">결과</th></tr><tr><th>A<sub>2</sub></th><th>A<sub>1</sub></th><th>A<sub>0</sub></th><th>D<sub>7</sub></th><th>D<sub>6</sub></th><th>D<sub>5</sub></th><th>D<sub>4</sub></th><th>D<sub>3</sub></th><th>D<sub>2</sub></th><th>D<sub>1</sub></th><th>D<sub>0</sub></th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	입력			결과								A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0
입력			결과																																																																																																												
A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>																																																																																																					
0	0	0	0	0	0	0	0	0	0	1																																																																																																					
0	0	1	0	0	0	0	0	0	1	0																																																																																																					
0	1	0	0	0	0	0	0	1	0	0																																																																																																					
0	1	1	0	0	0	0	1	0	0	0																																																																																																					
1	0	0	0	0	0	1	0	0	0	0																																																																																																					
1	0	1	0	0	1	0	0	0	0	0																																																																																																					
1	1	0	0	1	0	0	0	0	0	0																																																																																																					
1	1	1	1	0	0	0	0	0	0	0																																																																																																					
Boolean algebra	Truth table																																																																																																														

• Encoding

- 인코더란 디코더와 반대로  $m$ 개의 입력에 따라  $n$ 비트의 논리조합 값을 출력하고, 이를  $m \times n$  인코더라고 한다. 입력값은 10진수나 8진수를 받고, 출력값은 2진수나 BCD와 같은 코드로 변환하여 나타난다. 따라서  $2^n$ 개의 입력을 받고,  $n$ 개의 출력을 가지며 입력값에 대응하는 2진수를 생성하게 된다. 입력은 동시에 여러 입력을 할 수 없고, 하나의 input만 1이어야 한다.

아래에 4X2 인코더를 보면,  $D_0 \sim D_3$ 의 입력값이 있고,  $A_0 \sim A_1$ 의 출력 값이 있다. 출력값과 입력값에 대한 관계는 Boolean algebra를 보면 한눈에 알 수 있다. 10이나 8진수에서 1이나 3일 때에 2진수의 1의 자리에 1이 생기므로  $A_0 = D_1 + D_3$ , 입력값이 2나 4일 때에 2진수의 2의 자리에 1이 생기므로  $A_1 = D_2 + D_3$ 의 식이 성립하게 된다.

8X3 인코더를 보면 Truth table에서 한 예로  $D_4(100_2)$ 가 입력되었을 때  $A_{2,1,0}$ 은 1,0,0이 된다. 이 때  $A_0$ 는 최하위 비트 LSB를 의미하고,  $A_2$ 는 최상위 비트 MSB를 의미한다. 이를 통해 10진수나 8진수의 입력값이  $n$ 비트의 2진수 조합을 출력한다는 것을 알 수 있다.

인코더 이미지

Input				Output	
D3	D2	D1	D0	A0	A1
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

$$A_0 = D_1 + D_3$$

$$A_1 = D_2 + D_3$$

4X2인코더 회로도, Truth table, Boolean algebra

8X3 Encoder Truth table

$$A_0 = D_1 + D_3 + D_5 + D_7$$

$$A_1 = D_2 + D_3 + D_6 + D_7$$

$$A_2 = D_4 + D_5 + D_6 + D_7$$

Boolean algebra

입력								출력		
$D_7$	$D_6$	$D_5$	$D_4$	$D_3$	$D_2$	$D_1$	$D_0$	$A_2$	$A_1$	$A_0$
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Truth table

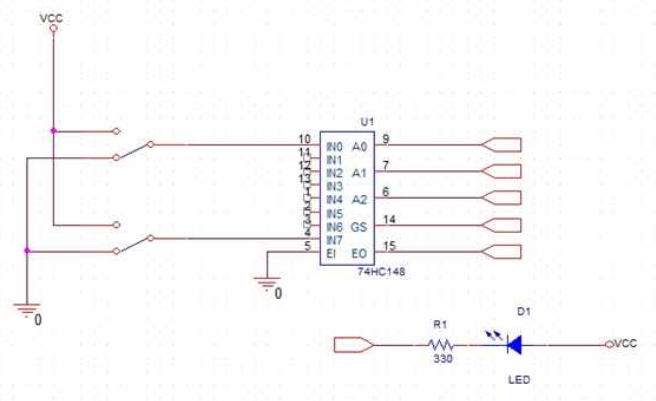
• BCD & Excess-3 code

- BCD 코드는 Binary-Coded Demical 코드를 의미하고, 각 자리수에 8,4,2,1의 크기를 갖는 가중치 코드이다. 이진수의 4비트가 10진수의 한자리를 의미하는데 따라서 0~9의 숫자만 갖게 되고, 아래의 표를 보면 10~15의 수는 버려지게 된다. 1010부터 1111까지는 쓰지 않는 수이다. 이 BCD코드에 3을 더하여 만든 코드가 Excess-3 code이다. 이는 각 자리수마다 의미하는 수가 없는 비가중치 코드의 한 종류이다. 이 코드 또한 BCD코드 기준으로 만들어졌기 때문에 10~15에 대해 대응하는 수는 없다. 특징으로는 9의 자기보수 성질이 있다. 예를 들어 2와 7은 0101, 1010으로 서로 1의 보수 관계에 있게 된다. Decimal digit에 대응하는 코드는 우측의 표와 같다.

Decimal digit	BCD (8421)	Excess-3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100
	1010	0000
	1011	0001
	1100	0010
	1101	1101
	1110	1110
	1111	1111

• Priority Encoder

- 동시에 입력이 제한되는 인코더와 다르게 높은 수에 우선순위를 줌으로써 동시에 2개의 Input이 입력될지라도 높은 수의 값을 우선순위에 두고 출력하는 인코더이다. 예를 들어 IN6,3이 동시에 입력이 된다면 IN6에 해당하는 Output이 출력된다.

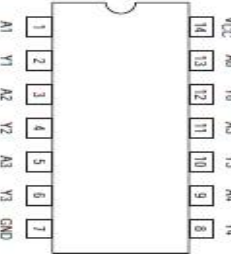


### 3.실험장비

IC칩 : 74HC04, 74HC08, 74HC42, 74HC148 - 1EA, 74HC20 - 2EA

LED, Resistor 330Ω 10EA

Power Supply

74HC04	Hex Inverters	Pin number			
		$V_{CC}$	GND	Input	Output
		14	7	A 1,3,5,9,11,13	Y 2,4,6,8,10,12
		Truth table			
		nA		nY	
		'0','L'		'1','H'	
		'1','H'		'0','L'	
		NOT Gate가 6개 있는 칩			



	1	0	0	1	1	1	1	1	1	1	1	1	1	0
	1	0	1	0	1	1	1	1	1	1	1	1	1	1
	1	0	1	1	1	1	1	1	1	1	1	1	1	1
	1	1	0	0	1	1	1	1	1	1	1	1	1	1
	1	1	0	1	1	1	1	1	1	1	1	1	1	1
	1	1	1	0	1	1	1	1	1	1	1	1	1	1
	1	1	1	1	1	1	1	1	1	1	1	1	1	1
BCD to Decimal Decoder 1-of-10 Gate가 1개 있는 칩														

74HC148	8 to 3 line priority encoder	Pin number								
		$V_{CC}$	GND	Input				Output		
		16	8	$I_0$	$I_1$	$I_2$	$I_3$	$A_0$	$A_1$	$A_2$
				10	11	12	13	9	7	6
				$I_4$	$I_5$	$I_6$	$I_7$			
1	2	3	4							

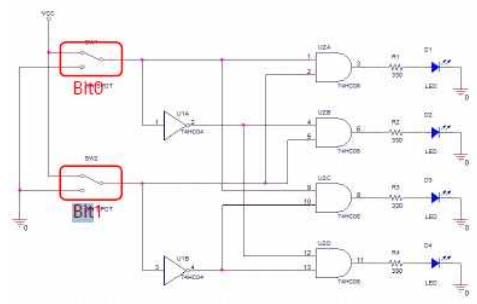
Truth Table											
$E$	$I_0$	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$	$I_7$	$A_0$	$A_1$	$A_2$
1	X	X	X	X	X	X	X	X	1	1	1
0	X	X	X	X	X	X	X	0	0	0	0
0	X	X	X	X	X	X	0	1	0	0	1
0	X	X	X	X	X	0	1	1	0	1	0
0	X	X	X	0	1	1	1	1	1	0	0
0	X	X	0	1	1	1	1	1	1	0	1
0	X	0	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	1

8 to 3 linepriority encoder Gate가 1개 있는 칩, Enable signal도 있으며 Negative logic  
이므로 Enable signal에 L가 입력되어야 정상적인 작동이 가능하다.

#### 4.실험과정 및 예상결과

##### 1. 2X4 Encoder

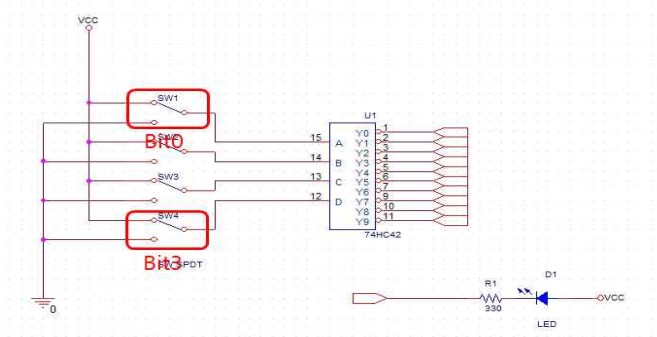
2x4 Decoder를 작동시키는 것이다. AND gate (74HC08)와 NOT gate (74HC04)를 이용하여 주어진 회로와 같이 구성하여 2×4 Decoder를 구현한다. 출력 부에 발광 LED를 연결하여 출력을 확인한다. 이때 LED에 저항을 연결하여 LED로 들어오는 전류를 제한한다. input에 차례대로 (0,0) (0,1) (1,0) (1,1) 로 입력하여 출력 D1, D2, D3, D4를 관찰한다.



INPUT		OUTPUT			
SW1	SW2	D1	D2	D3	D4
0	0	0	0	0	1
1	0	0	0	1	0
0	1	0	1	0	0
1	1	1	0	0	0

-예상결과 : 다음과 같은 진리표를 만들 수 있다.

## 2. BCD to Decimal Decoder



74HC42를 사용하여 주어진 회로를 구성한다. 출력에 LED를 연결하여 출력을 확인한다. 이때 LED에 저항을 연결하여 LED로 들어오는 전류를 제한한다. input에 (0,0,0,0) (0,0,0,1) (0,0,1,0) (0,0,1,1) (0,1,0,0) (0,1,0,1) (0,1,1,0) (0,1,1,1) (1,0,0,0) (1,0,0,1) 입력하여 출력을 확인한다.

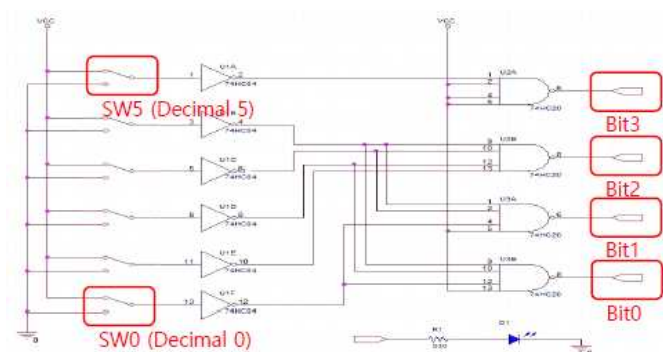
-예상결과:

출력 Y0~Y9에는 모두 Inverter가 부착되어 있기 때문에 active low이다. 따라서 실험 결과, 진리표를 작성하면 아래 표와 같이 나올 것이다.

INPUT				OUTPUT									
SW4	SW3	SW2	SW1	Y <sub>0</sub>	Y <sub>1</sub>	Y <sub>2</sub>	Y <sub>3</sub>	Y <sub>4</sub>	Y <sub>5</sub>	Y <sub>6</sub>	Y <sub>7</sub>	Y <sub>8</sub>	Y <sub>9</sub>
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0

## 3. 인코딩 - 10진 / Excess - 3 코드

NAND gate (74HC20)와 NOT gate (74HC04)를 이용하여 인코딩 역할을 하는 회로를 구성한다. 입력에는 모두 NOT Gate가 연결되어 있고 따라서 입력의 반대에 대한 보수 장치를 NAND Gate로 하는 것이다. Encoder의 정의에 따라 입력은 010000, 000001과 같이 하나의 입력만 1이고 나머지는 0이어야 하며 그렇지 않으면 Encoder 역할을 하지 않는다. 각 신호가 1이 될 때마다 출력으로 Encoding 된 것을 얻을 수 있으며 이것이 Excess-3 code이다.



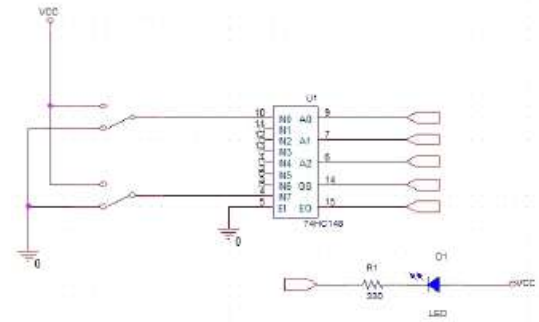
-예상 결과: 오른쪽 Truth Table을 얻을 수 있다.

SW5	SW4	SW3	SW2	SW1	SW0	Bit3	Bit2	Bit1	Bit0
0	0	0	0	0	1	0	0	1	1
0	0	0	0	1	0	0	1	0	0
0	0	0	1	0	0	0	1	0	1
0	0	1	0	0	0	0	1	1	0
0	1	0	0	0	0	0	1	1	1
1	0	0	0	0	0	1	0	0	0



#### 4. 8×3 Priority Encoder

위와 같은 회로를 구성한 후 (IN1~IN6도 IN0과 IN7과 같이 연결), 입력을 다양하게 넣으면서 이론의 8×3 Encoder와 같은지 확인한다.



- 예상결과 : 원래 인코더는 동시에 여러 입력이 들어갈 수 없지만 우선순위 인코더에서는 높은 비트의 수부터 우선순위를 매겨 연산을 처리한다. EI이 0일 때 Input에 따른 출력값을 확인할 수 있고, GS는 EI이 0일 때 EI를 제외한 INPUT 중 하나가 0인 경우에 L로 출력된다. 예를 들어 IN5이 0, IN6,7의 값이 1이라면 IN5~4의 값과 상관없이 IN5=0에 대한 결과값이 출력될 것이다. 따라서 Input에 대한 Out의 값은 아래의 표와 같이 예상된다.

Input									Output				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
1	X	X	X	X	X	X	X	X	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	X	X	X	X	X	X	X	0	0	0	0	0	1
0	X	X	X	X	X	X	0	1	0	0	1	0	1
0	X	X	X	X	X	0	1	1	0	1	0	0	1
0	X	X	X	0	1	1	1	1	1	0	0	0	1
0	X	X	0	1	1	1	1	1	1	0	1	0	1
0	X	0	1	1	1	1	1	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	1	1	1	0	1

#### 참고문헌

1. Nexperia (IC 74HC00, 74HC02, 74HC04(2개), 74HC08, 74HC32, 74HC86)
2. "Digital Design: Principles and Practices", 4thEdition, J. F. Wakerly
3. 강의노트, 아주대학교 Black Board

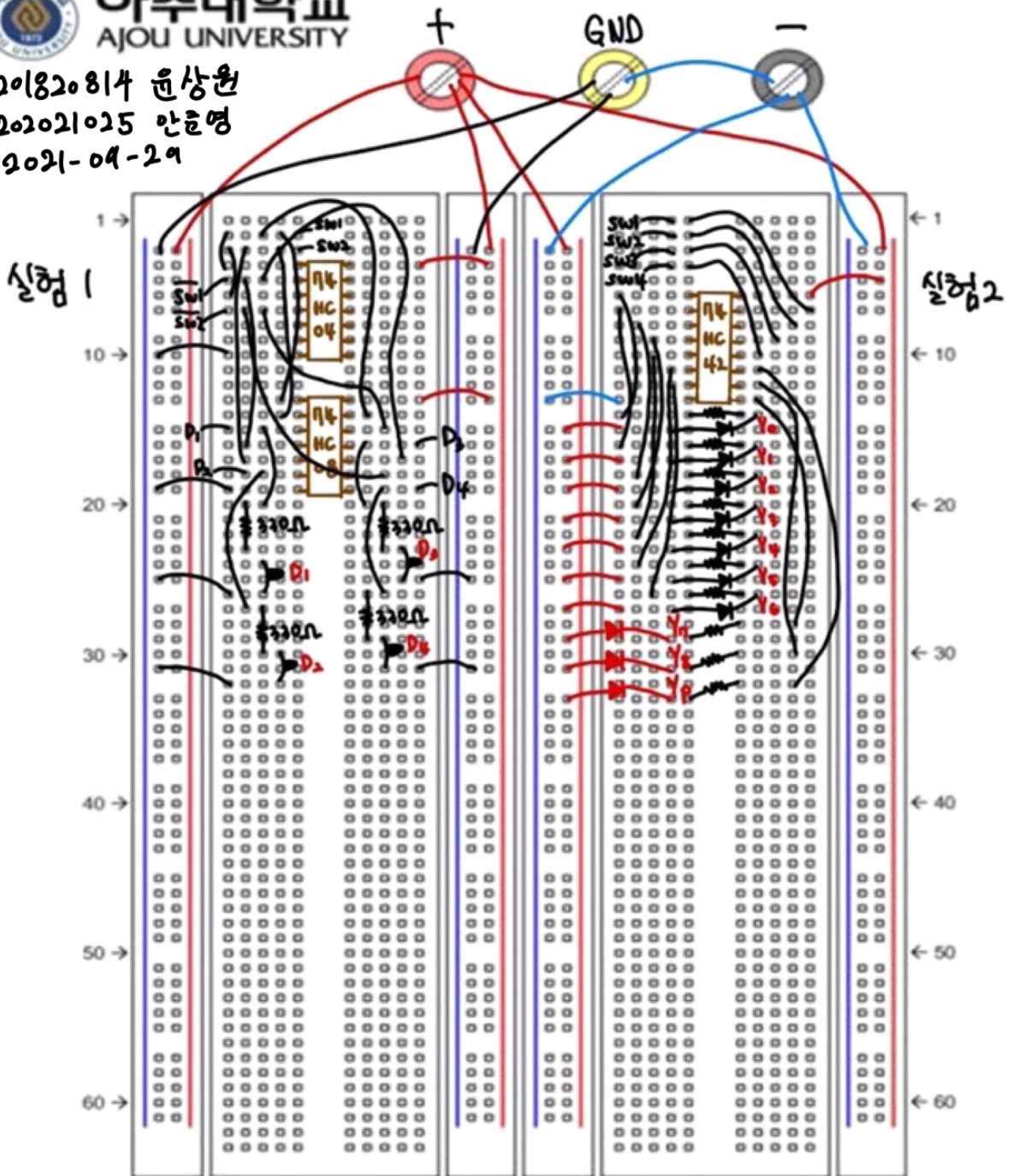


## 5. 회로 결선도



아주대학교  
AJOU UNIVERSITY

201820814 윤상원  
202021025 안준영  
2021-04-29





아주대학교  
AJOU UNIVERSITY

201820814 윤상원  
202021025 안준영  
2021-04-29

