논리회로실험

2003년 7월 30일

목 차

제 1 장

실험1: AND, OR, NOT 게이트

제 1 절 이론 요약

논리연산

- 논리연산에는
  - AND 연산
  - OR 연산
  - NOT 연산이 있음
- AND 연산
  - XY 또는 X·Y 로표현
  - X 와 Y 모두 1일때만 출력이 1이됨
  - 집합론으로 보면 교집합
  - 스위칭논리로 보면 직렬연결
  - min, max 로 보면 min

X	Y	Z=XY
0	0	0
0	1	0
1	0	0
1	1	1

- OR 연산
  - X+Y 로표현
  - X 혹은 Y 가 1일때 출력이 1이됨

  - 집합론으로 보면 합집합
  - 스위칭논리로 보면 병렬연결
  - min, max 로 보면 max

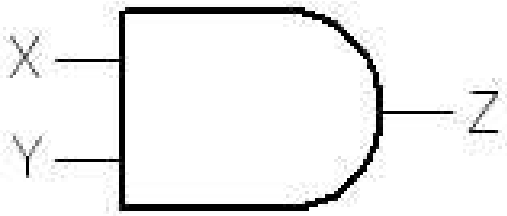
```
X  Y   Z=X+Y
--------------------
0  0    0
0  1    1
1  0    1
1  1    1
```

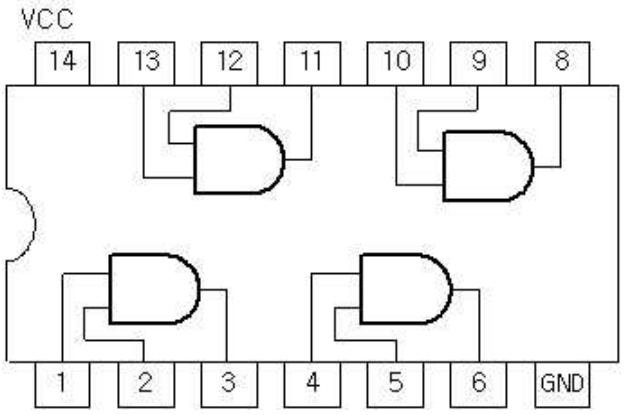
- NOT 연산
  - X'으로 표현
  - 입력이 1이면 0, 0이면 1이 출력됨
  - 집합론으로 보면 여집합

```
X  Z=X'
0   1
1   0
```

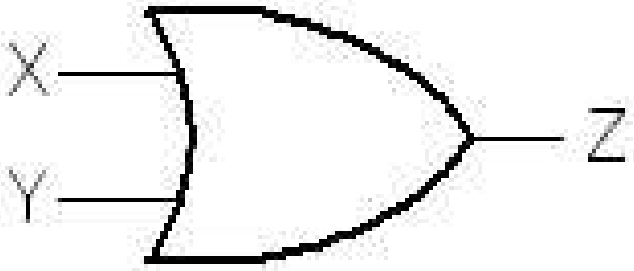
논리게이트

- 논리게이트란?
  - AND, OR, NOT 등의 논리연산을 전자회로로 구현한 것을 논리게이트라 함
  - AND, OR, NOT 게이트 이외에 NAND, NOR, XOR, XNOR등의 게이트가 있음
- AND 게이트
  - AND 연산을 구현
  - 논리 0은 보통 0V 로 논리 1은 보통 5V로 봄 (정논리)
  - 논리 0이나 논리 1로 볼수 있는 전압의 범위가 있음
  - AND 게이트를 구현한 상용 칩으로 7408이 있음
  - 74series 는 TTL(transistor-transistor logic) 로 구현한 칩으로서 많이 사용
  - 7408은 핀이 14개가 있고 내부에 2입력 AND 게이트가 4개가 구현되어 있음
  - 7408 칩
- OR 게이트
  - OR 연산을 구현





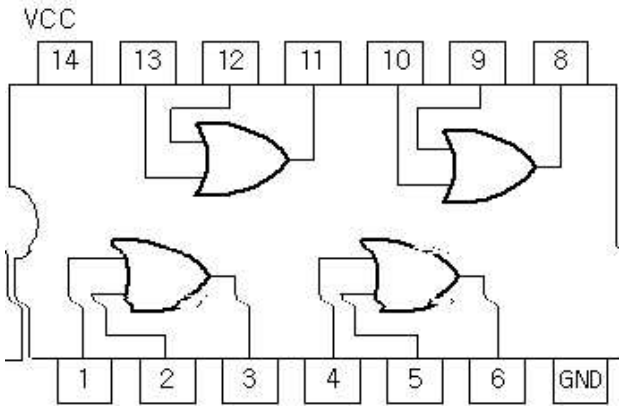
  - AND 게이트를 구현한 상용 칩으로 7432이 있음
  - 7432은 핀이 14개가 있고 내부에 2입력 OR 게이트가 4개가 구현되어 있음
  - 7432 칩

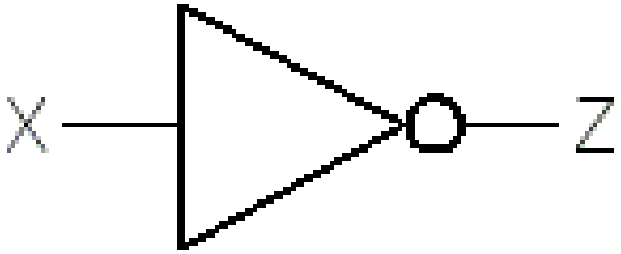


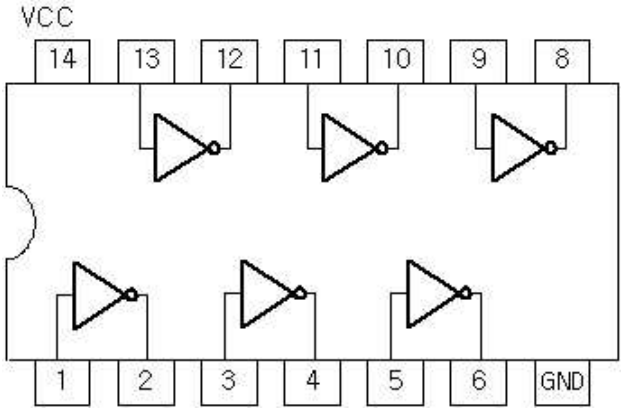
- NOT 게이트
  - NOT 연산을 구현
  - NOT 게이트를 구현한 상용 칩으로 7404이 있음
  - 7404는 핀이 14개가 있고 내부에 2입력 NOT 게이트가 4개가 구현되어 있음
  - 7404 칩

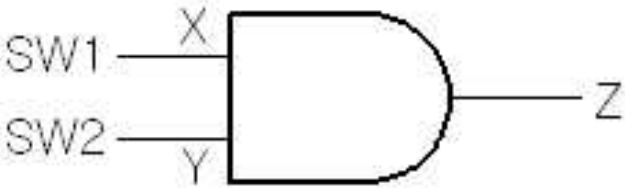
<u>실험</u>

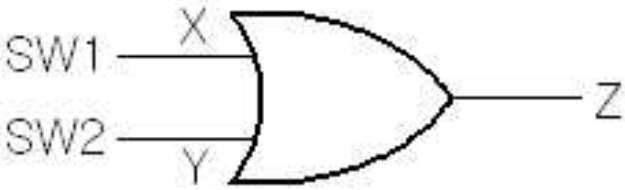
- 다음은 실험할 회로도 임
- 필요 실험부품
  - 2입력 AND 게이트: 7408
  - 2입력 OR 게이트: 7432

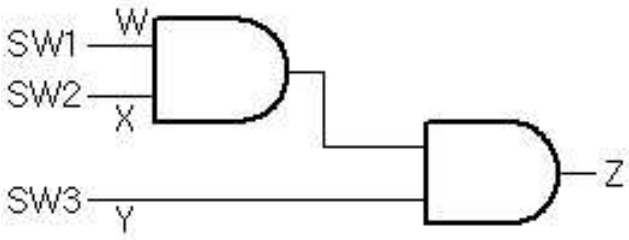

VCC
14
13
12
11
10
9
8
1
2
3
4
5
6
GND

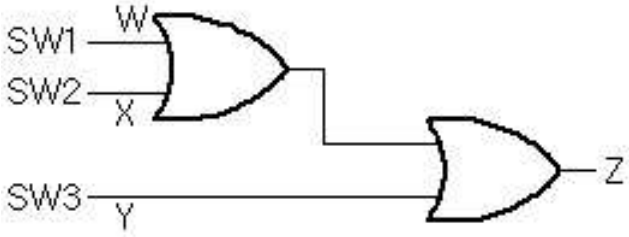

X
Z

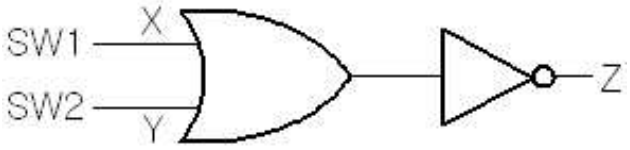

VCC
14
13
12
11
10
9
8
1
2
3
4
5
6
GND

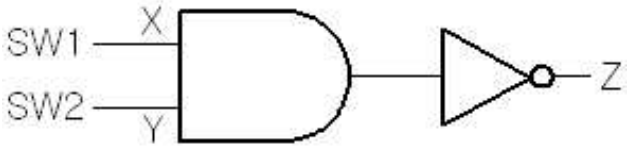

SW1
SW2
X
Y
Z

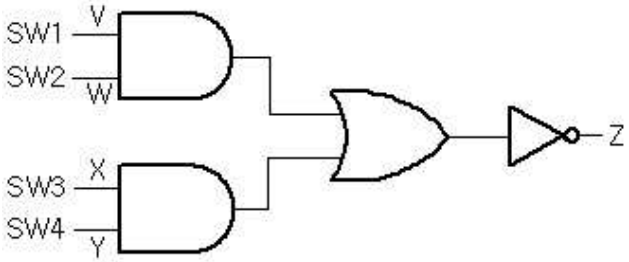

SW1
SW2
X
Y
Z


SW1
SW2
SW3
W
X
Y
Z


SW1
SW2
SW3
W
X
Y
Z


SW1
SW2
X
Y
Z


SW1
SW2
X
Y
Z


SW1
SW2
SW3
SW4
V
W
X
Y
Z

  - NOT 게이트: 7404

- 실험절차

  (1) 7408을 이용하여 그림 4(a) 회로를 구성하라. 입력 스위치를 이용하여 표 2와 같은 입력을 인가하고 그때의 전압 및 논리상태 값을 표 2의 AND 게이트 열에 기록하라. 논리상태 값은 게이트의 출력부를 로직 랩 유닛의 LED (light emitting diode) 에 연결하여 측정한다. LED가 켜진 경우는 논리상태 1을 나타내고, 꺼진 경우는 논리상태 0을 나타낸다.

  (2) 7432를 이용하여 그림 4(b) 회로를 구성하고, 절차 (1)을 반복하여 수행한 후, 결과를 표 2의 OR 게이트 열에 기록하라.

  (3) 7408을 이용하여 그림 4(c) 회로를 구성하고, 스위치를 이용하여 표 3과 같이 변화시키면서 출력의 논리상태값을 표 3 출력 첫 번째 열에 기록하시오.

  (4) 7432를 이용하여 그림 4(d) 회로를 구성하고, 스위치를 이용하여 표 3과 같이 변화시키면서 출력의 논리상태값을 표 3 출력 두 번째 열에 기록하시오.

  (5) 7408 및 7404를 이용하여 그림 4(e) 회로를 구성하고, 스위치를 이용하여 표 4와 같이 변화시키면서 출력의 논리상태값을 표 4 출력 첫 번째 열에 기록하시오.

  (6) 7408 및 7404를 이용하여 그림 4(f) 회로를 구성하고, 스위치를 이용하여 표 4와 같이 변화시키면서 출력의 논리상태값을 표 4 출력 두 번째 열에 기록하시오.

  (7) 7404, 7408 및 7432를 이용하여 그림 4(g) 회로를 구성하고, 스위치를 이용하여 표5와 같이 입력 V, W, X 및 Y를 변화시키면서 출력의 논리상태를 측정 기록하라.

제 2 절 결과보고서 다운로드

결과보고서

제 2 장

실험2: NAND, NOR, XOR 게이트

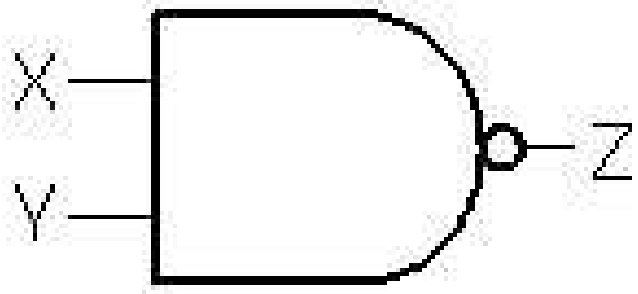
제 1 절 이론 요약

논리연산

- NAND, NOR, XOR 연산
  - NAND
    * AND 출력을 NOT 연산한 것
    * 수식: $Z = (XY)'$
    * 진리표

X	Y	(XY)'
0	0	1
0	1	1
1	0	1
1	1	0

    * 그림

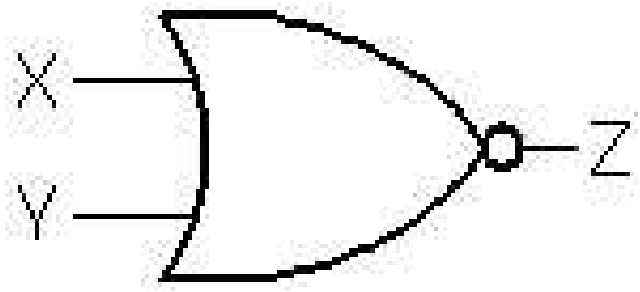


  - NOR
    * OR 출력을 NOT 연산한 것

* 수식: $X = (X + Y)'$
* 진리표

X	Y	(X+Y)'
0	0	1
0	1	0
1	0	0
1	1	0

* 그림



– NAND 나 NOR 는 AND 나 OR 보다 구현이 용이하여 더욱 많이 사용됨
– XOR
  * 2입력에서는 두개의 입력이 다른 경우에 출력이 1, 그렇지 않으면 출력이 0이 됨
  * 3입력 이상에서는 1의 갯수가 홀수이면 출력이1, 그렇지 않으면 출력이 0이됨
  * 기호: $\oplus$
  * 진리표

X	Y	X⊕Y
0	0	0
0	1	1
1	0	1
1	1	0

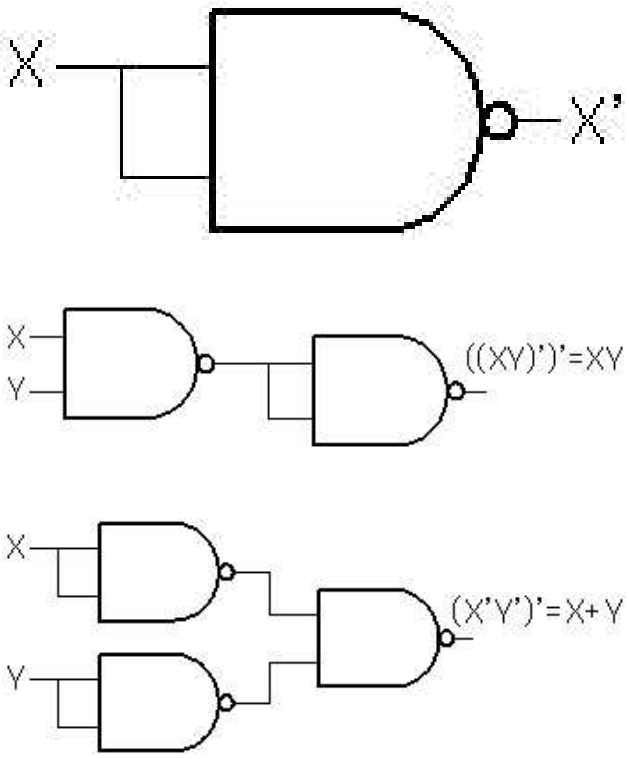
– XNOR (혹은 Equivalence 라고 불림)
  * XOR 출력을 NOT 연산한 것
  * 2입력에서는 두개의 입력이 같은 경우에 출력이 1, 그렇지 않으면 출력이 0이 됨

* 3이력이상에서는 0의 갯수가 짝수이면 1, 그렇지 않으면 출력이 0이됨
* 기호: ⊙
* 진리표

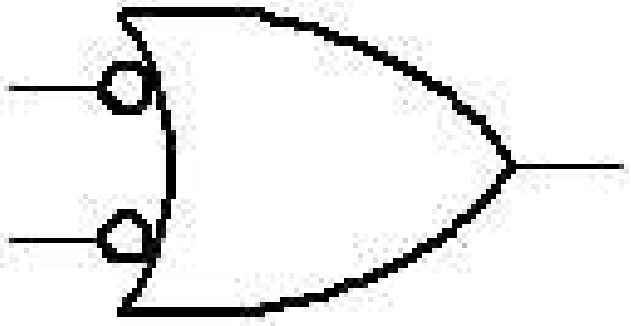
X	Y	X⊙Y
0	0	1
0	1	0
1	0	0
1	1	1

NAND 게이트로 표현

- NAND 게이트
  - 모든 논리 게이트가 NAND 게이트로 대치 가능 (범용 게이트)
  - AND, OR, NOT 게이트도 NAND 게이트로 표현 가능
  - 그림

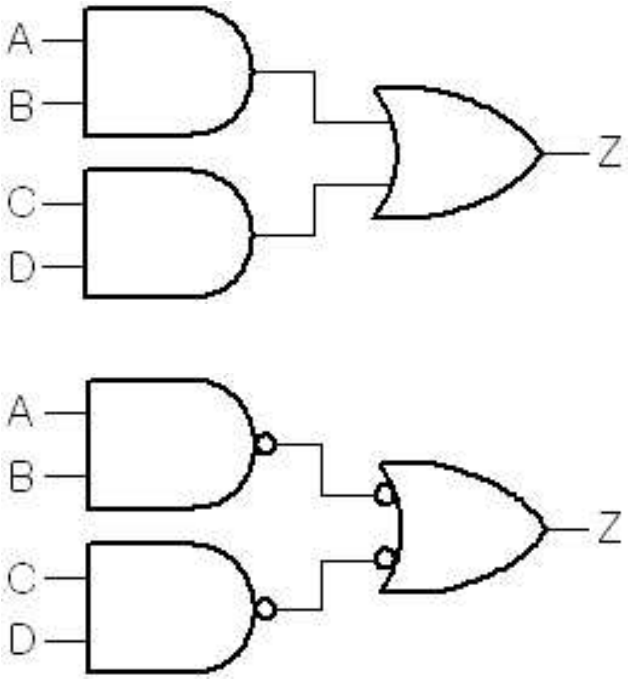


- NAND 게이트 등가회로
  - 드모르강 법칙: F = (XY)' = X' + Y'



  - 그림

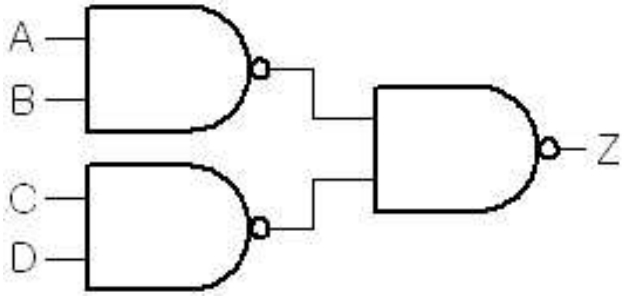
- Z=AB + CD 의 NAND 등가회로
  - (Z)" = ((AB)'(CD)')'
  - 그림

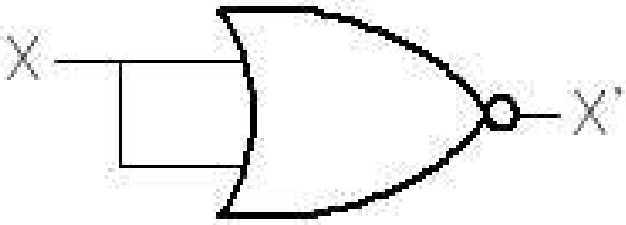


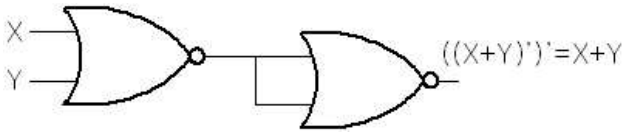
- NAND 구현 칩
  - 7400 은 4개의 NAND 게이트가 있음

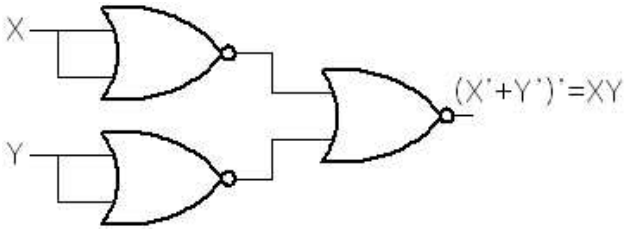
NOR 게이트로 표현

- NOR 게이트
  - 모든 논리 게이트가 NOR 게이트로 대치 가능 (범용 게이트)
  - AND, OR, NOT 게이트도 NOR 게이트로 표현 가능
  - 그림

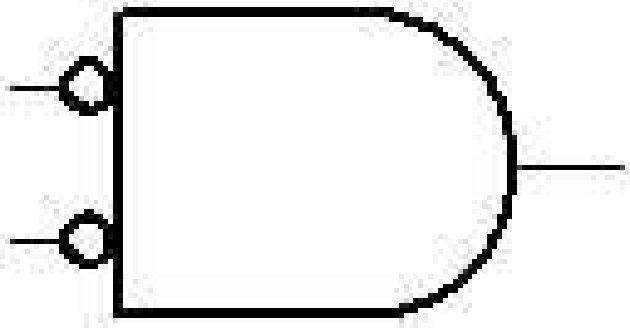

A
B
C
D
Z


X
X'

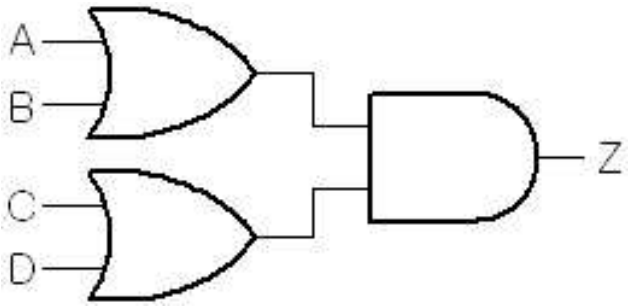

X
Y
((X+Y)')'=X+Y

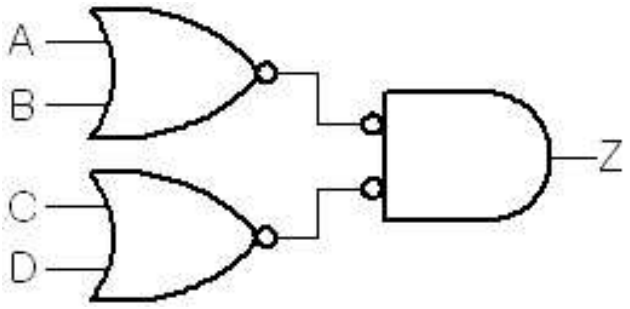

X
Y
(X'+Y')'=XY

- NOR 게이트 등가회로
  - 드모르강 법칙: F = (X+Y)' = X'Y'
  - 그림



- Z=(A+B)(C+D) 의 NOR 등가회로
  - (Z)" = ((A+B)'+(C+D)')'
  - 그림

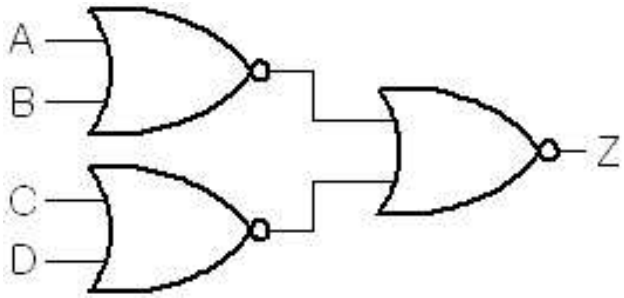




- NOR 구현 칩
  - 7402 은 4개의 NOR 게이트가 있음

XOR 게이트와 XNOR 게이트

- XOR



  - X ⊕Y=XY'+X'Y

- XNOR 는 XOR 출력을 NOT 한 것
  - (X ⊕Y)' = XY + X'Y'
- 3입력 XOR
  - P = W ⊕X ⊕Y
  - W, X, Y 중에서 1의 갯수가 홀수이면 P 가 1이 됨
  - 진리표

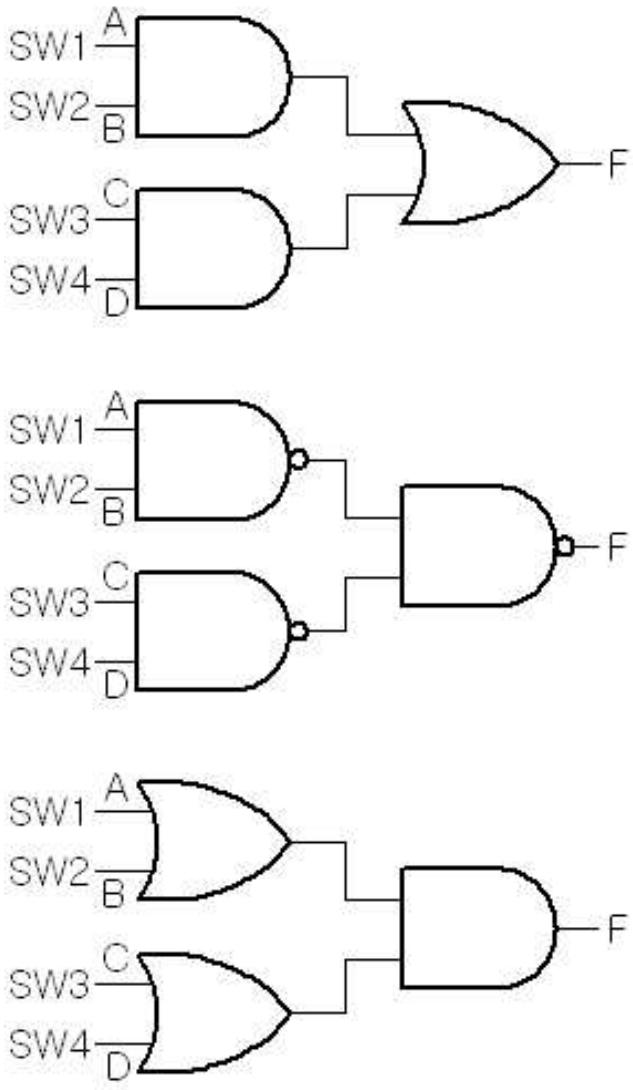
W	X	Y	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

- 에러 검출 방법
  - P = W ⊕X ⊕Y
  - P 를 패리티비트로 사용되어 전송중 에러를 검출할 수 있음
  - P 는 W, X, Y에서 1의 갯수가 홀수이면 1이되고 짝수이면 0이되어 P, W, X, Y 에서는 항상 1의 갯수가 짝수임
  - 그러므로 W, X, Y에 P를 추가하여 보내면 1비트에 에러 발생된 경우 1의 갯수를 검출하여 오류가 있는지 확인 가능
  - 에러 검출 식: C = W ⊕X ⊕Y ⊕P

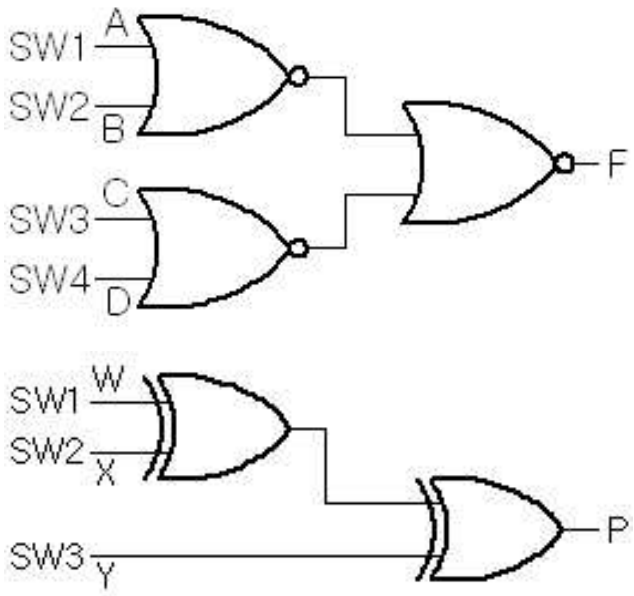
– C 가 0이면 에러가 없는 경우, C 가 1이면 에러가 있는 경우

실험

- 다음은 실험할 회로도 임



- 필요 실험부품
  - 2입력 NAND 게이트: 7400
  - 2입력 NOR 게이트: 7402
  - NOT게이트: 7404
  - 2입력 AND 게이트: 7408
  - 2입력 OR 게이트: 7432
  - XOR 게이트: 7486
- 실험절차



(1) 7408과 7432를 이용하여 그림 9(a) 회로를 구성하라. 입력 스위치를 이용하여 표 3과 같은 입력을 인가하고 그때의 전압 및 논리상태 값을 표 3의 출력 첫 번째 열에 기록하라.

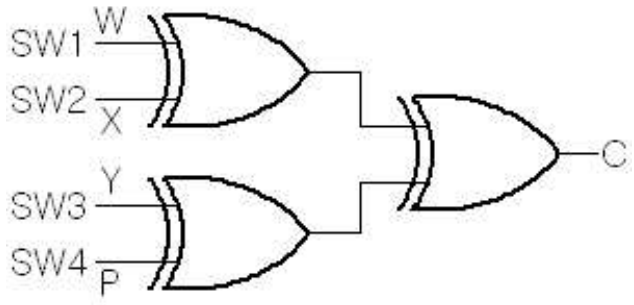
(2) 7400을 이용하여 그림 9(b) 회로를 구성하고, 절차 (1)을 반복하여 수행한 후, 결과를 표 3의 출력 두 번째 열에 기록하라.

(3) 7432와 7408을 이용하여 그림 9(c) 회로를 구성하고, 스위치를 이용하여 표 4와 같이 변화시키면서 출력의 논리 상태값을 표 4의 출력 첫 번째 열에 기록하시오.

(4) 7402을 이용하여 그림 9(d) 회로를 구성하고, 절차 (3)을 반복하여 수행한 후, 결과를 표 4의 출력 두 번째 열에 기록하라.

(5) 7486을 이용하여 그림 9(e) 회로를 구성하고, 스위치를 이용하여 표 5와 같이 변화시키면서 출력의 논리상태값을 표 5에 기록하시오. (이 회로는 다음 절차 (6)에 사용되니 선을 뽑지 말 것.)

(6) (5)에 사용된 회로에 XOR를 하나 더 연결하여 그림 9(f) 의 회로를 구성하시오. 그리고 스위치를 이용하여 표 6과 같이 변화시키면서 출력의 논리 상태값을 표 6에 기록하시오.



제 2 절 결과보고서 다운로드

결과보고서

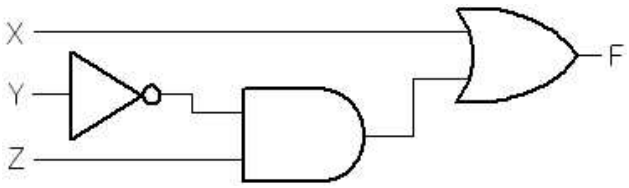
제 3 장

실험3: 부울함수 및 카르노 맵

제 1 절 이론 요약

부울 함수

- 부울함수란?
  - 부울대수는 논리값 0과 1 그리고 연산 AND, OR, NOT으로 정의
  - 부울함수는 이진 입력변수들의 출력 2진 값으로의 매핑을 부울연산으로 정의한 함수
  - 예)
    - F(X,Y,Z) = X + Y'Z
    - F = X + Y'Z 로도 표현
    - 논리게이트



    - 진리표

X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1

1	1	0	1
1	1	1	1

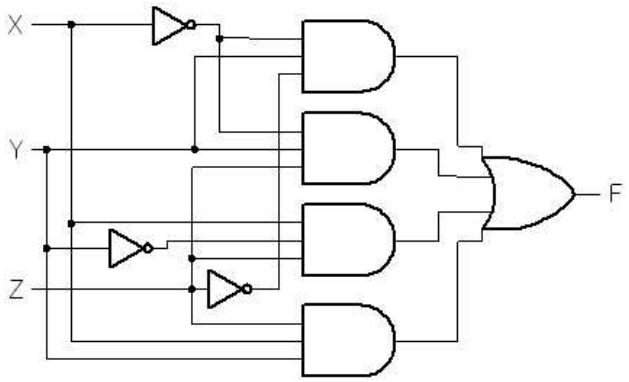
- 논리 간략화
  - 진리표가 다음과 같을 때

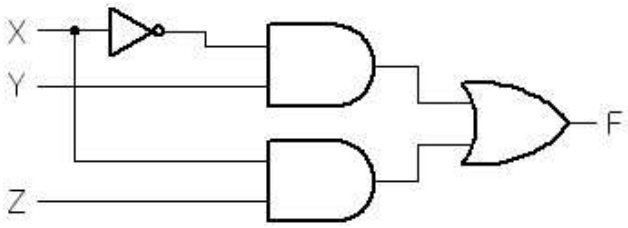
X	Y	Z	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

  - 부울함수로 나타내면
    * sum of minterm으로 나타내면 F=X'YZ'+X'YZ+XY'Z+XYZ
    * 이는 더욱 간략화될 수 있음

```
F = X'Y(Z'+Z) + X(Y'+Y)Z
  = X'Y · 1 + X·1·Z
  = X'Y + XZ
```

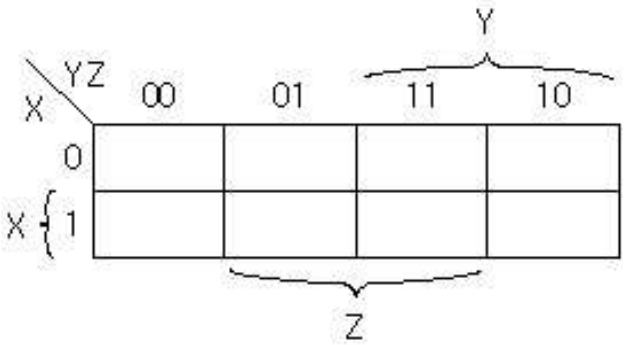
    * 그림 (간략화 이전과 이후)

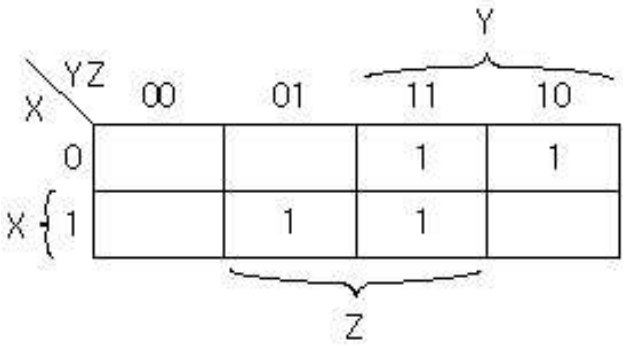




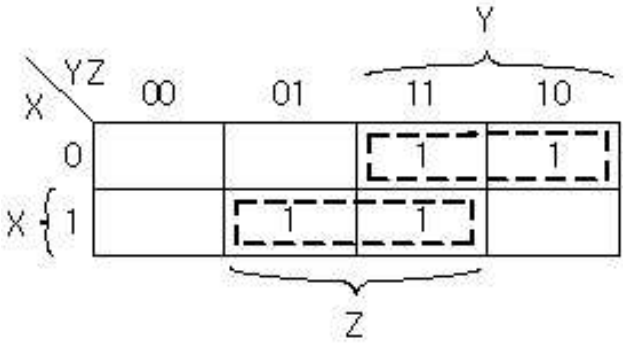
<u>카르노 맵을 이용한 논리 간략화</u>

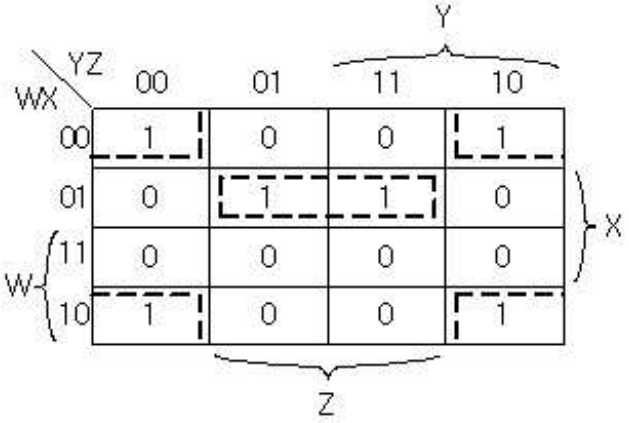
- X, Y, Z 3입력 함수를 간략화하기 위한 카르노 맵
  - X 를 세로로 배치하고 Y 와 Z를 가로로 배치
  - 변수값은 gray 코드로 배치
  - 그림

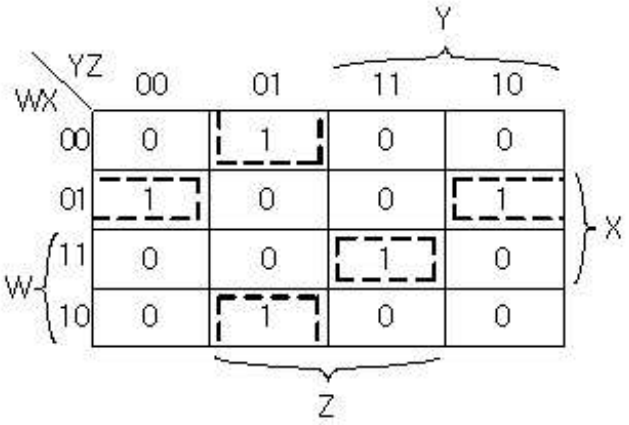




- 간략화 규칙
  - 1을 1, 2, 4, 8, 16 과 같이 2의 지수승이 되도록 묶는다
  - 가능하면 크게 묶는다.
  - 0 과 1에 걸쳐서 묶인 변수는 간략화 되어 사라진다
  - 좌우와 상하도 1비트만 틀린것으로서 묶일 수 있다
  - 그림 (변수가 4개인 카르노 맵의 예)
  - 위의 첫번째것은 F1 = X'Z' + W'XZ, 두번째것은 F2=W'XZ'+X'Y'Z+WXYZ 가 됨

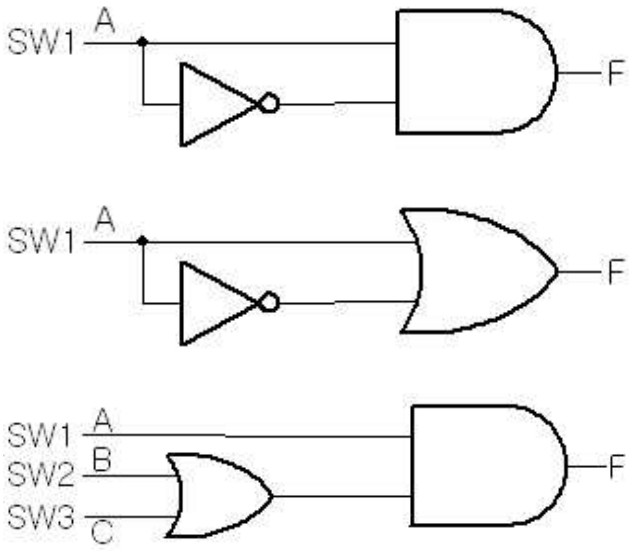

Y
YZ
X
00
01
11
10
0
1
1
1
1
1
X
Z


Y
YZ
WX
00
01
11
10
00
1
0
0
1
01
0
1
1
0
11
0
0
0
0
10
1
0
0
1
W
X
Z


Y
YZ
WX
00
01
11
10
00
0
1
0
0
01
1
0
0
1
11
0
0
1
0
10
0
1
0
0
W
X
Z

실험

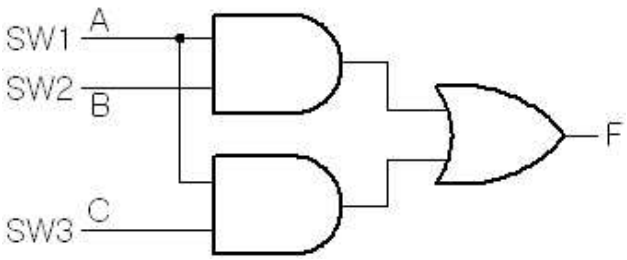
- 다음은 실험할 회로도 임



- 필요 실험부품
  - 2입력 NAND 게이트: 7400
  - 2입력 NOR 게이트: 7402
  - NOT게이트: 7404
  - 2입력 AND 게이트: 7408
  - 2입력 OR 게이트: 7432
- 실험절차
  (1) 7400과 7404를 이용하여 그림 5(a) 회로를 구성하라. 입력 스위치를 이용하여 표 4와 같은 입력을 인가하고 그때의 전압 및 논리상태 값을 표4에 기록하라.
  (2) 7432와 7400을 이용하여 그림 5(b) 회로를 구성하고, 절차 (1)을 반복하여 수행한 후, 결과를 표5에 기록하라.
  (3) 7432와 7408을 이용하여 그림 5(c) 회로를 구성하고, 스위치를 이용하여 표 6과 같이 변화시키면서 출력의 논리 상태값을 표 6에 기록하시오.
  (4) 7432와 7408을 이용하여 그림 5(d) 회로를 구성하고, 스위치를 이용하여 표 6과 같이 변화시키면서 출력의 논리 상태값을 표 6에 기록하시오.
  (5) 예비보고 (4)에서 구한 곱의 논리합 형태의 회로를 구성하고 스위치를 이용하여 표 7과 같이 변화시키면서 출력의 논리 상태값을 표 7에 기록하시오.
  (6) 예비보고 (5)에서 구한 합의 논리곱 형태의 회로를 구성하고 스위치를 이용하여 표7과 같이 변화시키면서 출력의 논리 상태값을 표 7에 기록하시오.

제 2 절 결과보고서 다운로드

결과보고서


SW1
A
SW2
B
SW3
C
F

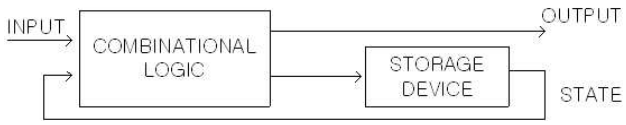
제 4 장

실험4: SR latch

제 1 절 이론 요약

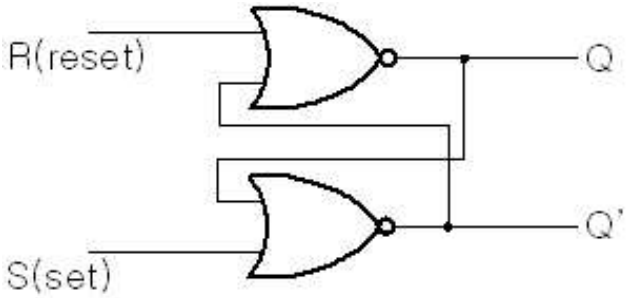
순서회로

- 순서회로란?
  - 출력이 입력에만 의존해서 결정되는 것을 조합회로라 함
  - 출력이 현재의 입력뿐만 아니라 과거의 입력에도 의존해서 결정되는 회로를 순서회로라 함
  - 그림



  - 순서회로에서는
    - 과거의 입력에 의한 출력중 몇개가 기억소자에 기억되고 있다가 현재의 입력과 같이 입력이 됨
    - 기억된 값을 상태라고하기도 함

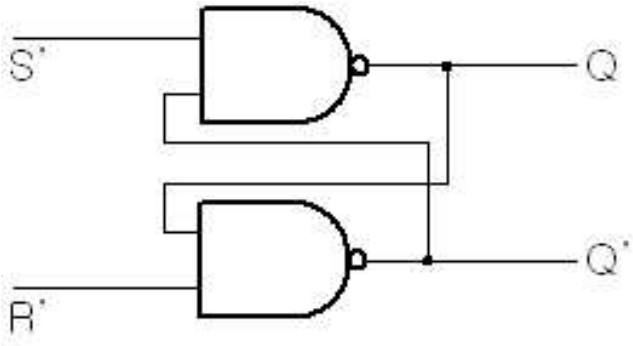
- latch
  - 이진값을 저장하기 위한 장치
  - NOR 게이트를 이용한 latch
  - 출력 Q 는 정상출력 Q' 은 보수출력을 의미
  - S 는 set을 R 은 reset 을 의미 (set 은 출력 Q를 1로 만들고 reset 은 출력 Q를 0으로 만듦)
  - 동작
    - S 와 R 이 모두 0 인경우: 현재의 Q 와 Q' 가 feedback 되어 현재의 상태을 유지함 (NOR 게이트에서 0 입력은 출력에 영향을 미치지 못함)



* S 가 1 이고 R 이 0 인 경우: S 가 1임으로 출력 Q'는 0이되고 R도 0이고 Q'도 0임으로 Q 는 1이됨
* S 가 0 이고 R 이 1 인 경우: R 이 1임으로 출력 Q는 0이되고 S도 0이고 Q도 0임으로 Q'는 1이됨
* S 와 R 이 모두 1 인 경우: Q 와 Q' 가 보수관계임에도 불구하고 Q와 Q' 가 모두 0이됨 (불안정)
* 함수표

S	R	Q	Q'	상태
1	0	1	0	set
0	0	1	0	상태유지
0	1	0	1	reset
0	0	0	1	상태유지
1	1	0	0	불안정

- NAND 게이트를 이용한 S'R' 래치
  - 그림

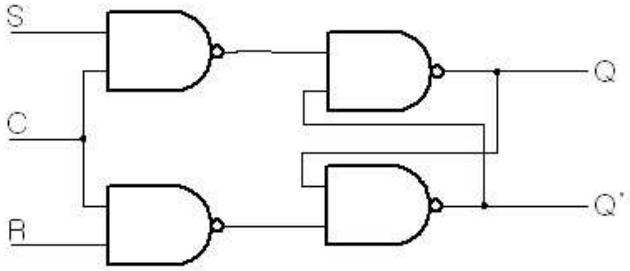


  - 동작
    * S' 와 R' 가 모두 1인 경우: 현재의 Q 와 Q' 가 feedback 되어 현재의 상태을 유지함 (NAND 게이트에서 1입력은 출력에 영향을 미치지 못함)

  * S' 가 1 이고 R' 가 0 인 경우: R'가 0임으로 Q'는 1이되고 S' 와 Q'가 1임으로 Q는 0이됨
  * S' 가 0 이고 R' 가 1 인 경우: S'가 0임으로 Q는 1이되고 Q 와 R' 가 모두 1임으로 Q'는 0이됨
  * S' 와 R' 가 모두 0인 경우: Q 와 Q' 가 보수관계임에도 불구하고 Q와 Q' 가 모두 0이됨 (불안정)
- S'R' 래치
  * 출력 Q 를 1로 하기위해 S' 를 0으로 해야하기 때문에 S' 로 이름 붙임
  * 출력 Q 를 0로 하기위해 R' 를 0으로 해야하기 때문에 R' 로 이름 붙임
  * 함수표

S'	R'	Q	Q'	상태
0	1	1	0	set
1	1	1	0	상태유지
1	0	0	1	reset
1	1	0	1	상태유지
0	0	0	0	불안정

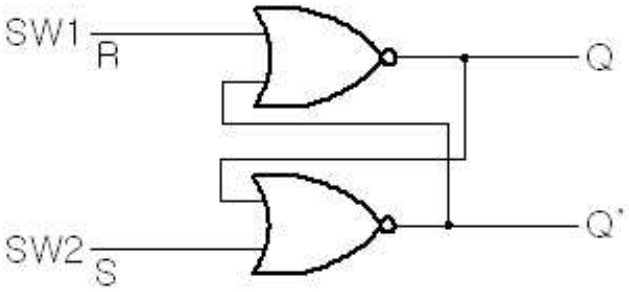
- 제어입력을 가진 latch
  - latch 가 set 되거나 reset 되는 것을 조정하기 위한 입력 C 를 추가적으로 구성
  - 그림

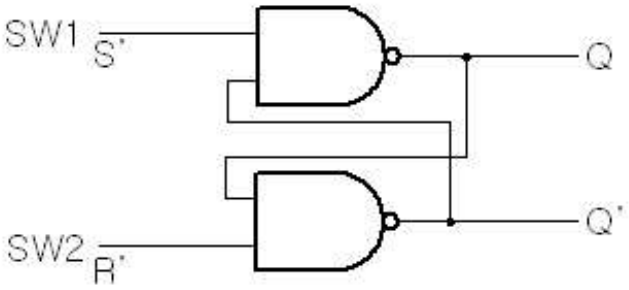


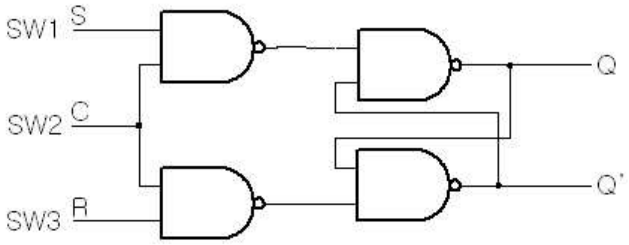
  - 동작
    * C 가 0이면 S' 와 R' 가 1이되어 상태유지
    * C 가 1인 경우 S 와 R 값이 S' 와 R' 로 입력됨으로 SR latch 로 동작

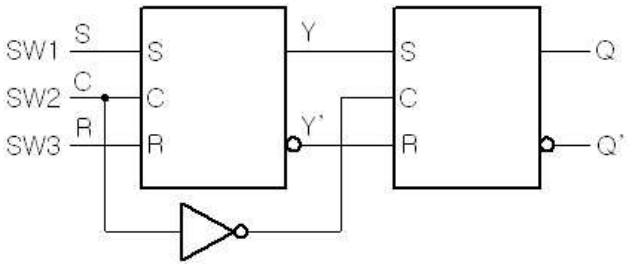
실험

- 다음은 실험할 회로도 임
- 필요 실험부품


SW1
R
Q
Q'
SW2
S


SW1
S'
Q
Q'
SW2
R'


SW1
S
SW2
C
SW3
R
Q
Q'


SW1
S
SW2
C
SW3
R
S
C
R
Y
Y'
S
C
R
Q
Q'

  - 2 입력 NAND: 7400
  - 2 입력 NOR: 7402
  - NOT: 7404

- 실험절차

  (1) 7402 를 이용하여 그림 5(a) 회로를 구성하라. 스위치를 이용하여 표 3에 제시된 순서대로 S 및 R 입력을 변화시키고 및 의 상태를 측정하여 표 3에 기록하라.

  (2) 7400을 이용하여 그림 5(b) 회로를 구성하라. 스위치를 이용하여 표 4에 제시된 순서대로 및 입력을 변화시키고 및 의 상태를 측정하여 표 4에 기록하라.

  (3) 7400 을 이용하여 그림 5(c) 회로를 구성하라. 스위치를 이용하여 표 5에 제시된 순서대로 S, R 및 C 입력을 변화시키고 및 의 상태를 측정하여 표 5에 기록하라.

  (4) 7400 및 7404를 이용하여 그림 5(d) 회로를 구성하라. 스위치를 이용하여 표 6에 제시된 순서대로 S, R 및 C 입력을 변화시키고 및 의 상태를 측정하여 표 6에 기록하라.

제 2 절 결과보고서 다운로드

결과보고서

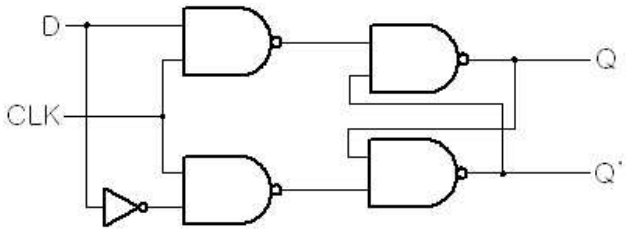
제 5 장

실험5: D 및 JK 플립플롭

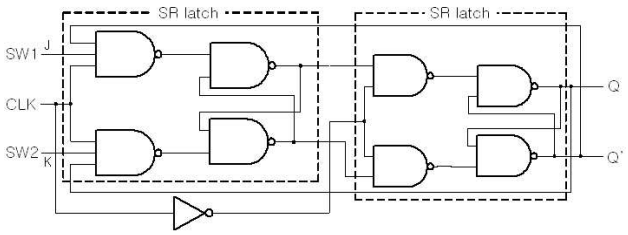
제 1 절 이론 요약

래치 와 플립플롭

- D 래치 논리도

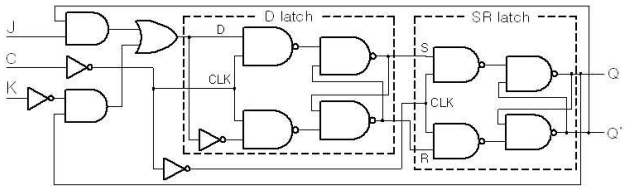


- 래치의 동작특성
  - 래치는 level trigger 로서 CLK 가 1인 동안의 입력변화가 그대로 출력에 전달됨
  - 그러므로 CLK가 1인 동안에 입력이 계속적으로 변화하면 출력도 계속적으로 변화함
  - 이는 시스템을 결정적이지 못하고 불안정하게 할 수 있음
  - 이러한 래치의 단점을 보완하고자 플립플롭이 대두됨
- 플립플롭
  - 래치의 레벨트리거와는 다른게 입력의 변화가 계속적으로 출력에 영향을 미치지 않음
  - 플립플롭을 구현하는 두가지 방법
    1. 마스터 슬레이브 JK 플립플롭
    2. 에지트리거 플립플롭
- 마스터 슬레이브 JK 플립플롭
  - 회로도



  - 동작
    * CLK=1 인경우 마스터래치가 입력을 받아들여 정보를 저장한다.
    * CLK=1 인경우 슬레이브래치는 CLK=0 임으로 상태유지를 하고 있다.
    * CLK=0 이되면 마스터래치는 상태유지로 들어가고
    * CLK=0 이되면 슬레이브래치가 마스터래치의 출력을 받아서 정보를 저장한다.
    * 슬레이브의 출력이 플립플롭의 출력임으로 CLK 가 1에서 0으로 되는 순간에 출력이 변하며
    * 마스터래치가 상태유지를 하고있기 때문에 더이상의 출력변화는 없다 (플립플롭의 특성)
    * 단, 슬레이브래치에 최종적으로 저장되는 정보는 CLK 가 1인 동안에 마스터래치에 가해진 입력에 의존한다.
    * 위와같은 특성에 의하여 마스터슬레이브 플립플롭은 펄스 트리거 플립플롭이라고도 함

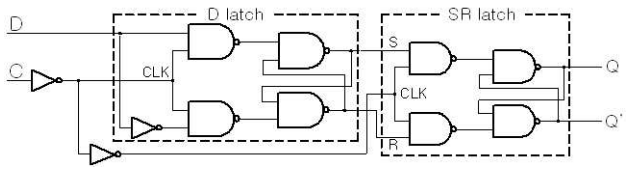
- 에지 트리거 플립플롭
  - 에지 트리거 JK 플립플롭 회로도

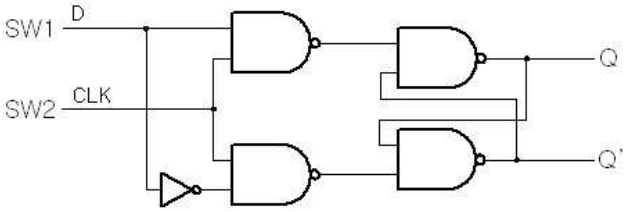


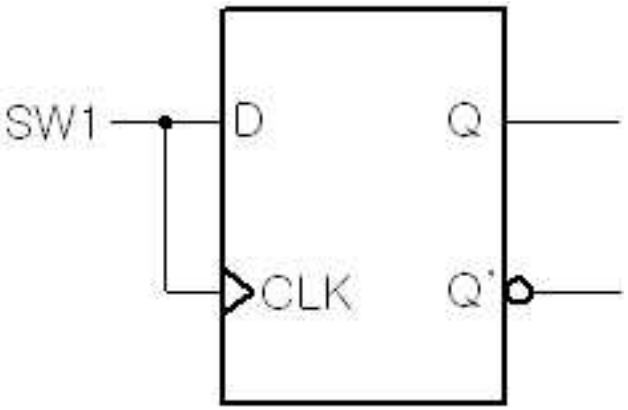
  - 동작
    * CLK 가 1에서 0으로 변하거나 0에서 1로 변하는 시점에 출력값이 변하는 것은 마스터슬레이브 플립플롭과 같다.
    * 단, 마스터슬레이브 플립플롭에서의 출력은 CLK 가 1이거나 0인 동안에 입력 값에 의존하지만
    * 에지 트리거 플립플롭에서는 CLK가 1에서 0으로 변하거나 (negative edge) 0에서 1로 변하기 (positive edge)직전의 입력값이 출력으로 전달된다.
  - 에지트리거 D 플립플롭 회로도

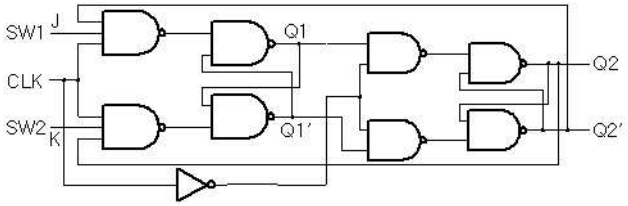
<u>실험</u>

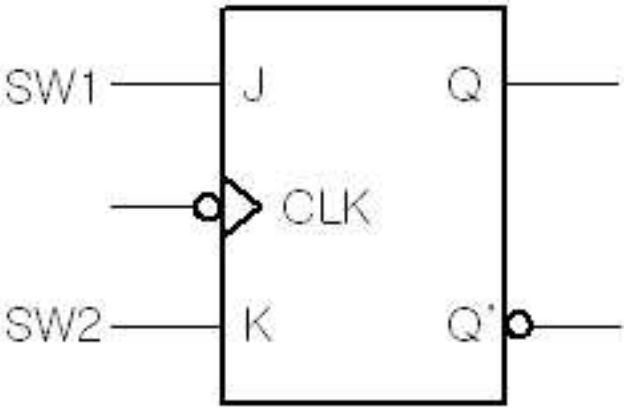
- 다음은 실험할 회로도 임


D
C
D latch
CLK
SR latch
S
CLK
R
Q
Q'


SW1
D
SW2
CLK
Q
Q'


SW1
D
Q
CLK
Q'


SW1
J
CLK
SW2
K
Q1
Q1'
Q2
Q2'


SW1
J
Q
CLK
SW2
K
Q'

- 필요 실험부품
  - 2 입력 NAND: 7400
  - 3 입력 NAND: 7410
  - NOT: 7404
  - D 플립플롭: 7474
  - JK 플립플롭: 7476
- 실험절차

(1) 7400 및 7404를 이용하여 그림 5(a) 회로를 구성하고, 스위치를 이용하여 표1과 같은 순서로 D 및 CLK 입력을 변화시키면서 Q 및 의 상태를 측정하여 표1의 첫째 열에 기록하라.

(2) 7474를 이용하여 그림 5(b) 회로를 구성하고, 스위치를 이용하여 표1과 같은 순서로 D 및 CLK 입력을 변화시키면서 및 의 상태를 측정하여 표1의 둘째 열에 기록하라. 이 때, 7474의 CLR 및 PRS 신호는 활성화되지 않도록 적절히 연결하라.

(3) 7400, 7404 및 7410을 이용하여 그림 5(c) 회로를 구성하고, 스위치를 이용하여 표2와 같은 순서로 J, K 및 CLK 입력을 변화시키면서 , 및 , 의 상태를 측정하여 표2 에 기록하라.

(4) 7476을 이용하여 그림 5(d) 회로를 구성하고, 스위치를 이용하여 표2와 같은 순서로 J, K 및 CLK 입력을 변화시키면서 및 의 상태를 측정하여 표2 에 기록하라. 이 때, 7476의 CLR 및 PRS 신호는 활성화되지 않도록 적절히 연결하라.

제 2 절 결과보고서 다운로드

결과보고서

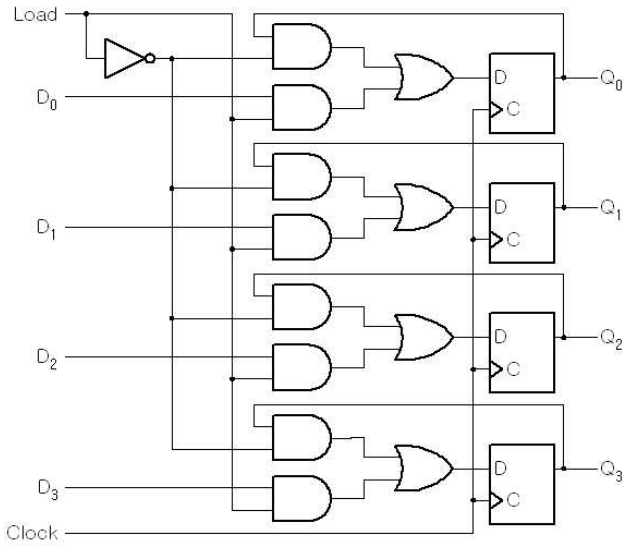
제 6 장

실험6: 시프트 레지스터

제 1 절 이론 요약

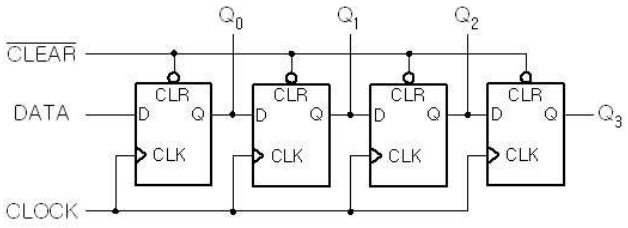
레지스터

- 레지스터
  - n 비트를 저장하는 장치
  - 보통 D 플립플롭을 이용하여 제작
  - 로드 (load): 새로운 입력을 받아 레지스터에 저장하는 작업
  - 4비트 병렬로드 기능의 레지스터

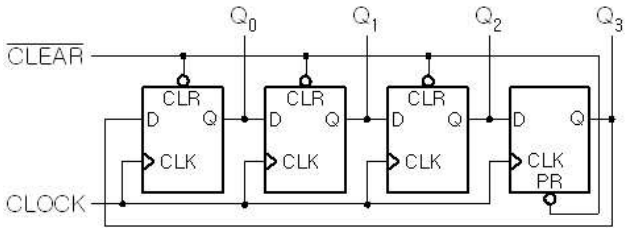


  - 동작
    - Load 가 0 이면 현재의 D 플립플롭의 출력이 feedback 되어 유지됨
    - Load 가 1 이면 외부입력 D0, D1, D2, D3 가 Clock 의 상승에지에서 D 플립플롭에 저장됨
- 시프트 레지스터

– 저장된 이진 데이터를 인접한 플립플롭을 통해 1bit 씩 이동시킬 수 있는 레지스터



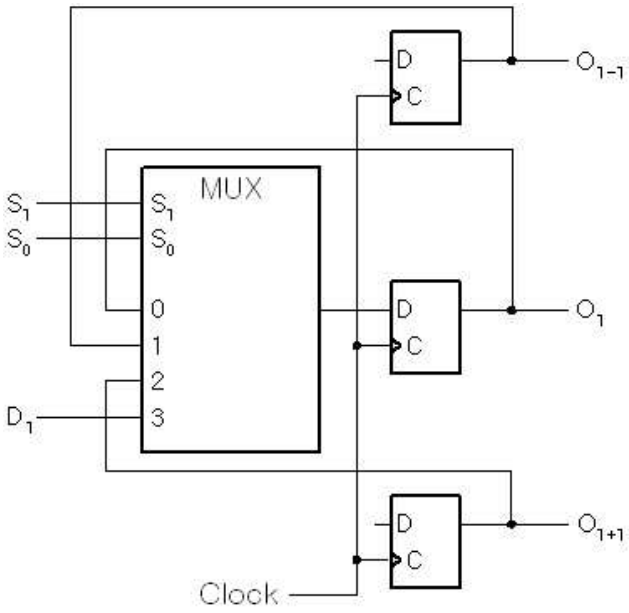
- 시프트 레시스터를 이용한 링 카운터
  - 플립플롭의 최종 출력을 처음 플립플롭의 입력으로 feedback 하여 구현
  - 초기 값은 0001 에서 시작

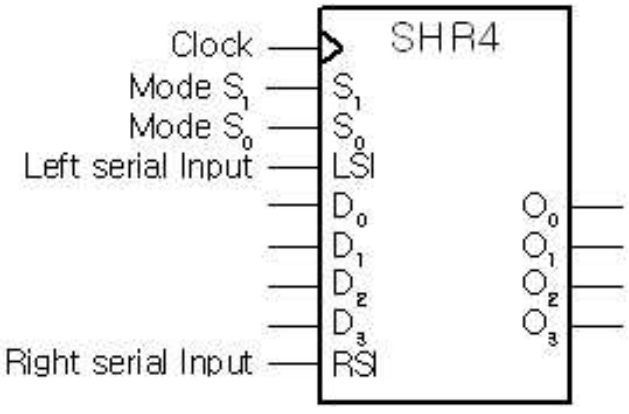


CLK	Q_0	Q_1	Q_2	Q_3
0	0	0	0	1
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0

- 범용 시프트 레지스터
  - 양방향으로 시프트 가능하고 병렬로드 기능이 있는 시프트 레지스터를 범용 시프트 레지스터라 함
  - 회로도
  - 기호
  - 동작

Mode	Control

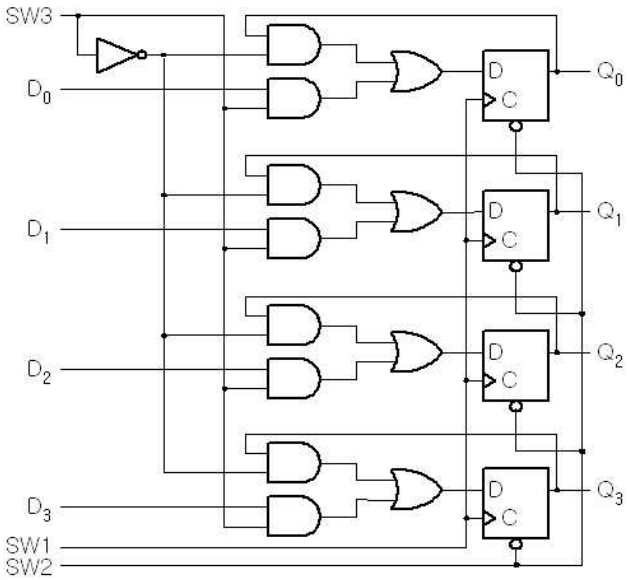

D
C
O_{1-1}
MUX
S_1
S_0
0
1
2
3
D_1
O_1
O_{1+1}
Clock

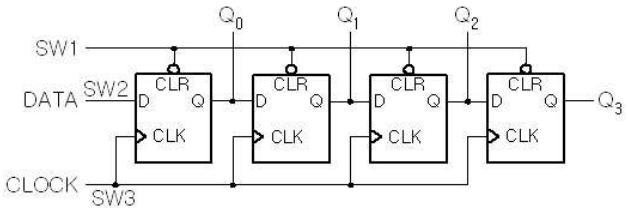

SHR4
Clock
Mode S_1
Mode S_0
Left serial Input
D_0
D_1
D_2
D_3
Right serial Input
LSI
RSI
S_1
S_0
O_0
O_1
O_2
O_3

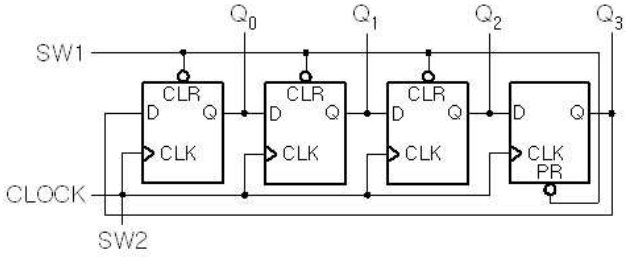
S_1	S_0	Register Operation
0	0	No Change
0	1	Shift down
1	0	Shift up
1	1	Parallel load

실험

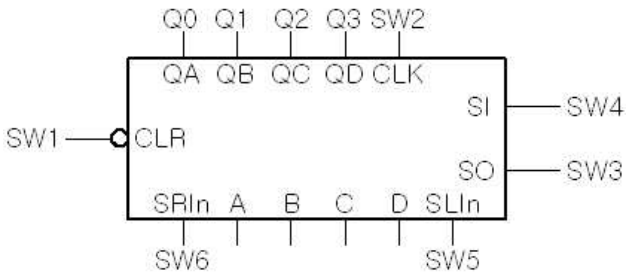
- 다음은 실험할 회로도 임







- 필요 실험부품



  - D 플립플롭: 7474 NOT: 7404
  - AND: 7408 OR: 7432
  - 4 bit Universal 시프트 레지스터: 74194

- 실험절차

  (1) 7404, 7408, 7432, 7474를 이용하여 그림 5(a) 회로를 구성하고, 스위치를 이용하여 표1과 같은 순서로 입력을 변화시키면서 의 상태를 측정하여 표1에 기록하라.

  (2) 7474를 이용하여 그림 5(b) 회로를 구성하고, 스위치를 이용하여 표2와 같은 순서로 CLEAR, D 및 CLK 입력을 변화시키면서 의 상태를 측정하여 표2에 기록하라.

  (3) 7474를 이용하여 그림 5(c) 회로를 구성하고, 스위치를 이용하여 표3과 같은 순서로 입력을 변화시키면서 의 상태를 측정하여 표3에 기록하라.

  (4) 74194 Universal 시프트 레지스터를 이용하여 그림 5(d) 회로를 구성하고, 스위치를 이용하여 표4와 같은 순서로 입력을 변화시키면서 의 상태를 측정하여 표4에 기록하라.

제 2 절 결과보고서 다운로드

결과보고서

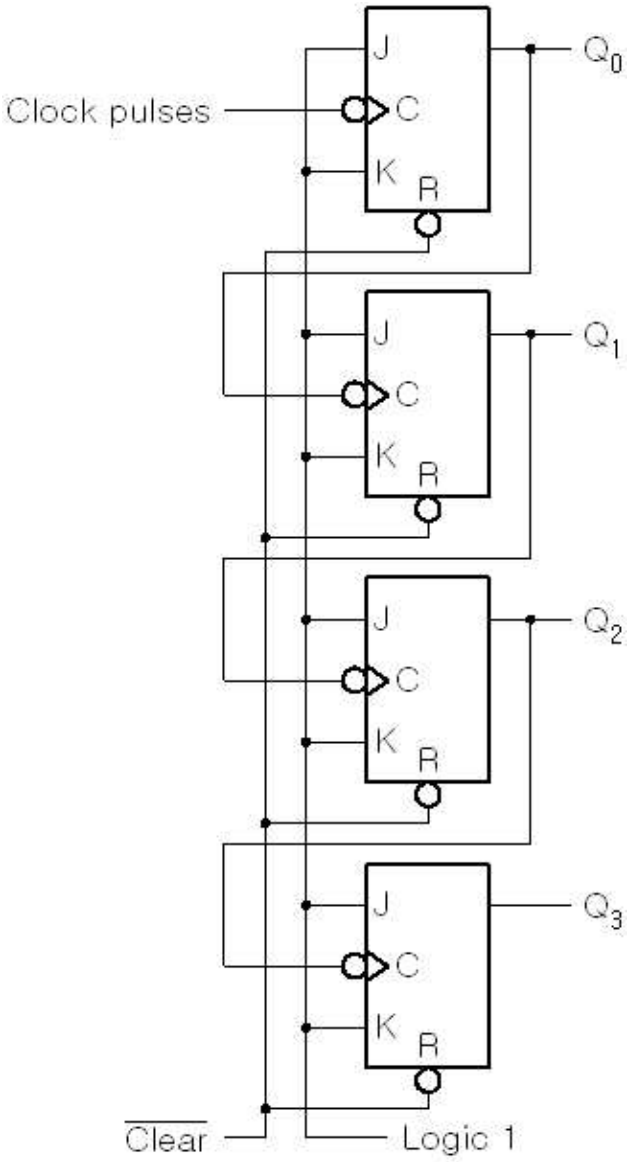
제 7 장

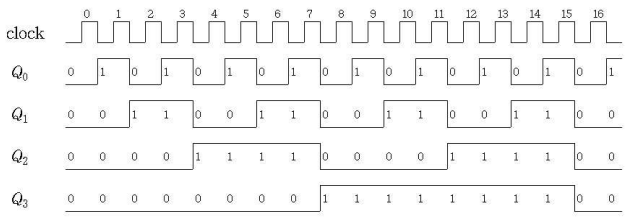
실험7: 비동기형 카운터

제 1 절 이론 요약

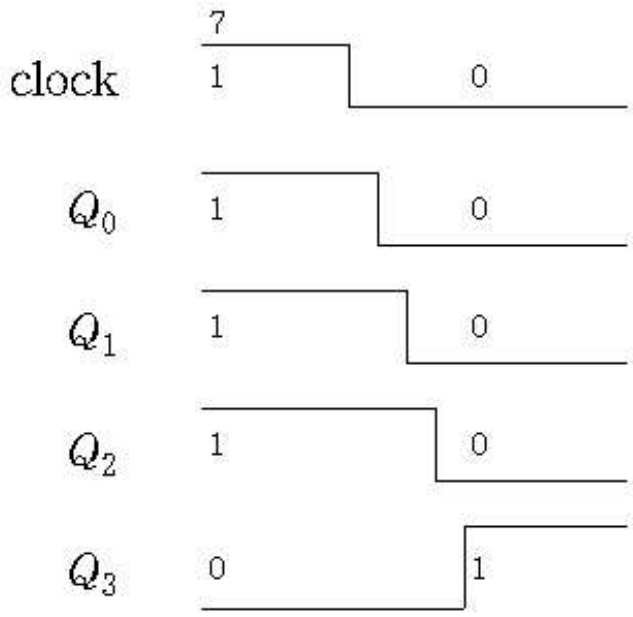
비동기형 카운터와 동기형 카운터

- 카운터
  - 출력의 상태변화가 정해진 순서를 반복하는 특별한 레지스터
  - 종류
    - 동기형 카운터
      - 카운터의 모든 레지스터가 클럭에의해 트리거됨
      - 모든 레지스터의 상태변화가 동기되어 있음 (동시에 변화함)
    - 비동기형 카운터
      - 다른 플립플롭의 출력에 의하여 트리거됨
      - 모든 레지스터의 상태변화가 동기되어 있지 않음 (변하는 순간이 다름)
      - 전단의 변화가 다음단에 클럭으로 쓰이므로 차례로 변화가 일어남 →ripple counter 라고도 함
      - 동기형 보다 회로가 간단하나 전파지연에 의하여 고주파에서 동작이 어려움
      - 동기형 카운터는 체계적인 설계 기법이 존재하나 비동기형 카운터는 설계자의 직관에 의존
- 2진 카운터와 십진(BCD) 카운터
  - 2진 카운터: 카운터의 상태변화 순서가 이진수의 차례를 따름
  - 십진 카운터: BCD 코드에 따라 상태가 변함 즉 0000에서 1001까지 반복적으로 카운트
  - 3bit 2진 카운터: 0에서 7까지 변하는 카운터
- 비동기형 이진 상승 카운터
  - 회로도


J
C
K
R
Q0
Q1
Q2
Q3
Clock pulses
Clear
Logic 1


clock
Q0
Q1
Q2
Q3

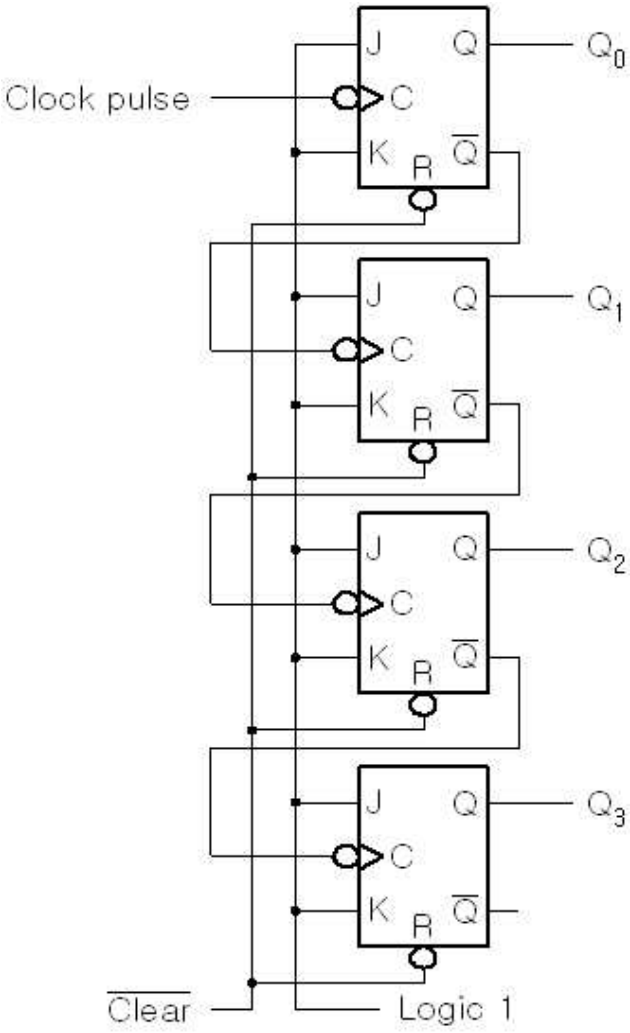
- 타이밍도
- 7번째 클럭에서 (0111) ~ (1000) 변이의 순간을 확대한 그림 (비동기 동작)

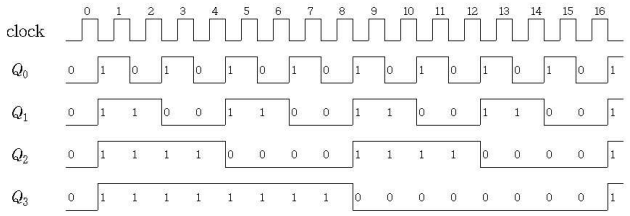


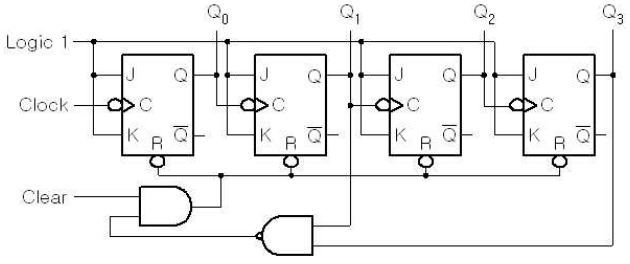
- 비동기형 이진 하강 카운터
  - 상승 카운터에서는 Q_i 를 다음 플립플롭의 clock 으로 사용
  - 하강 카운터에서는 Q_i' 를 다음 플립플롭의 clock 으로 사용
  - 회로도
  - 타이밍도
- 비동기형 십진 카운터
  - 0부터 9까지 카운트하는 회로
  - 4비트를 이용함으로 16개의 상태를 가지나 0000 부터 1001 까지 10개만 이용
  - 회로도
  - 동작
    - 1010이 되면 즉 Q_3 과 Q_1 이 1이되면 비동기 clear 가 발생
    - 1010 상태가 없고 바로 0000으로 감
    - 그러므로 0000 부터 1001까지를 카운트

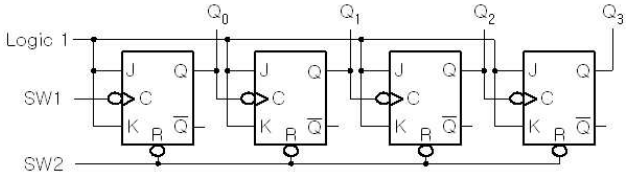
<u>실험</u>

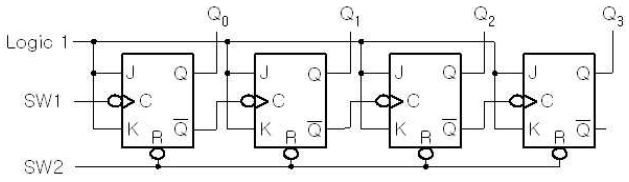
- 다음은 실험할 회로도 임
- 필요 실험부품
  - 2 입력 NAND: 7400

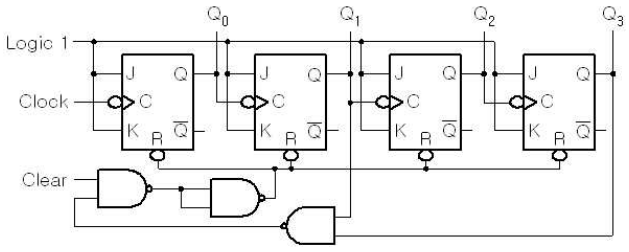

Clock pulse
J
Q
C
K
R
$\overline{Q}$
Q_0
Q_1
Q_2
Q_3
Clear
Logic 1

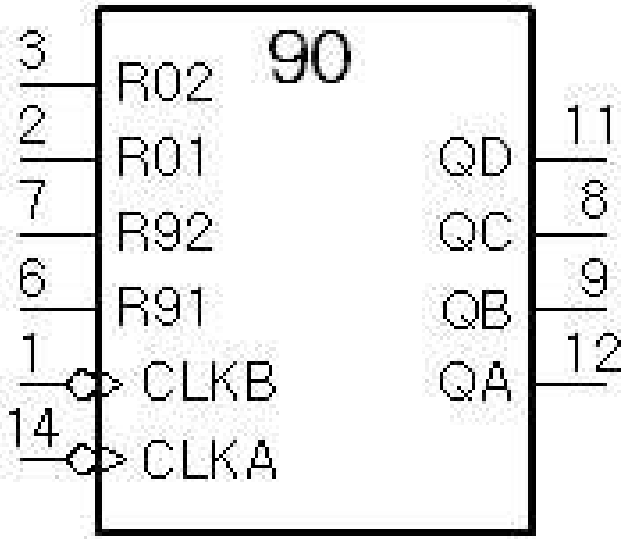

clock
Q_0
Q_1
Q_2
Q_3


Logic 1
Clock
Clear
Q_0
Q_1
Q_2
Q_3


Logic 1
SW1
SW2
Q_0
Q_1
Q_2
Q_3


Logic 1
SW1
SW2
Q_0
Q_1
Q_2
Q_3
J
C
K
R
Q
$\overline{Q}$


Logic 1
Clock
Clear
Q_0
Q_1
Q_2
Q_3
J
C
K
R
Q
$\overline{Q}$


90
3
R02
2
R01
7
R92
6
R91
1
CLKB
14
CLKA
11
QD
8
QC
9
QB
12
QA

  - JK 플립플롭: 7476
  - 십진 카운터: 7490

- 실험절차

  (1) 7476을 이용하여 그림 7(a) 회로를 구성하고, SW1을 이용하여 CLK 입력을 변화시키면서 의 상태를 측정하여 표1에 기록하라.

  (2) 7476을 이용하여 그림 7(b) 회로를 구성하고, SW1을 이용하여 CLK 입력을 변화시키면서 의 상태를 측정하여 표2에 기록하라.

  (3) 7476 및 7400을 이용하여 그림 7(c) 회로를 구성하고, SW1을 이용하여 CLK 입력을 변화시키면서 의 상태를 측정하여 표3의 왼쪽에 기록하라.

  (4) 그림 7(d) 의 7490을 이용하여 BCD 카운터를 구현하되, SW1은 클럭신호로, SW2는 CLEAR 신호로 동작하도록 각 입력을 적절히 연결하라(예비보고서 (5)②). SW1을 이용하여 CLK 입력을 변화시키면서 의 상태를 측정하여 표3의 오른쪽에 기록하라.

제 2 절 결과보고서 다운로드

결과보고서

제 8 장

실험8: 동기형 카운터

제 1 절 이론 요약

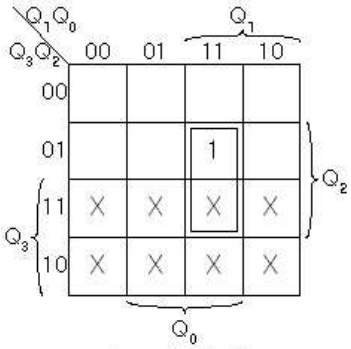
동기형 카운터

- 특징
  - 모든 플립플롭의 클럭에 동일한 클럭펄스가 가해짐
  - 상태전이가 동시에 발생
  - 높은주파수에서 작동 가능
  - 순차회로 설계기법으로 설계가능
- 동기형 2진 카운터의 설계
  - 순차회로 설계기법에 따라서 상태표를 얻고 여기표를 적용하여 설계
  - JK 플립플롭을 이용한 4비트 2진카운터의 설계

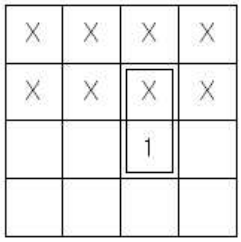
Present state				Next state				Flip-flop inputs							
Q_3	Q_2	Q_1	Q_0	Q_3	Q_2	Q_1	Q_0	J_{Q3}	K_{Q3}	J_{Q2}	K_{Q2}	J_{Q1}	K_{Q1}	J_{Q0}	K_{Q0}
0	0	0	0	0	0	0	0	0	X	0	X	0	X	1	X
0	0	0	1	0	0	0	1	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	0	0	X	0	X	X	0	1	X
0	0	1	1	0	0	1	1	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	0	0	X	X	0	0	X	1	X
0	1	0	1	0	1	0	1	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	0	0	X	X	0	X	0	1	X
0	1	1	1	0	1	1	1	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	0	X	0	0	X	0	X	1	X
1	0	0	1	1	0	0	1	X	0	0	X	1	X	X	1
1	0	1	0	1	0	1	0	X	0	0	X	X	0	1	X
1	0	1	1	1	0	1	1	X	0	1	X	X	1	X	1
1	1	0	0	1	1	0	0	X	0	X	0	0	X	1	X
1	1	0	1	1	1	0	1	X	0	X	0	1	X	X	1
1	1	1	0	1	1	1	0	X	0	X	0	X	0	1	X
1	1	1	1	1	1	1	1	X	1	X	1	X	1	X	1

  - 각 플립플롭의 여기표는 다음과 같음
  - 논리간략화
  - 논리도

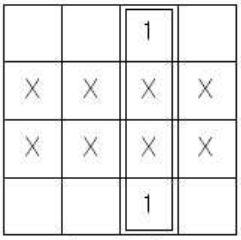
$Q(t)$	$Q(t+1)$	J	K	S	R	D	T
0	0	0	x	0	x	0	0
0	1	1	x	1	0	1	1
1	0	x	1	0	1	0	1
1	1	x	0	x	0	1	0



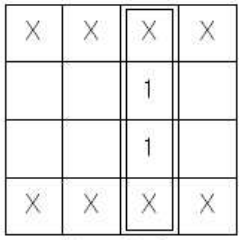
$J_{Q3}=Q_0Q_1Q_2$



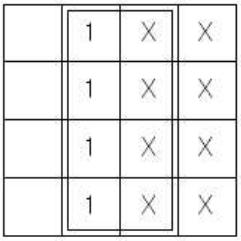
$K_{Q3}=Q_0Q_1Q_2$



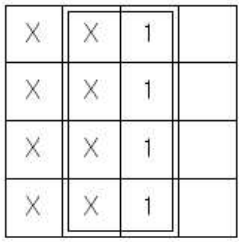
$J_{Q2}=Q_0Q_1$



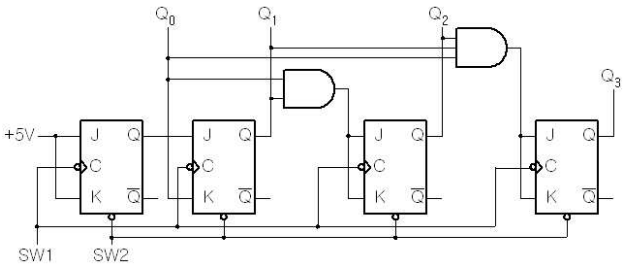
$K_{Q2}=Q_0Q_1$



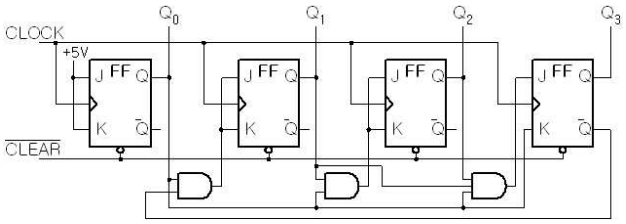
$J_{Q1}=Q_0$



$K_{Q1}=Q_0$

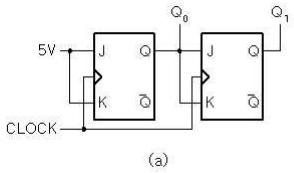


- 동기형 10진 카운터
  - 0부터 9까지 카운트하는 10진 카운터
  - 4비트로 설계하여야만 10개의 상태를 나타냄
  - 4비트임으로 16개의 상태이나 10개의 상태만 사용
  - 사용하지 않는 1010에서 1111은 dont' care 로 놓음



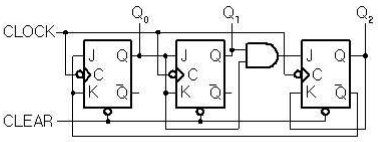
  - 동작
    - 상태 $Q_0 \sim Q_3$ 가 1001일때 각 플립플롭의 JK 입력은
    - 각각 11, 00, 00, 01 이므로 Q_0는 반전 Q_3는 reset 됨
    - 그러므로 1001 다음 상태는 0000
- Modulo N 카운터
  - 서로 다른 N개의 상태를 갖는 카운터
  - Modulo 4 카운터와 Modulo 5 카운터

CLK	Q_1	Q_0
0	0	0
1	0	1
2	1	0
3	1	1



(a)

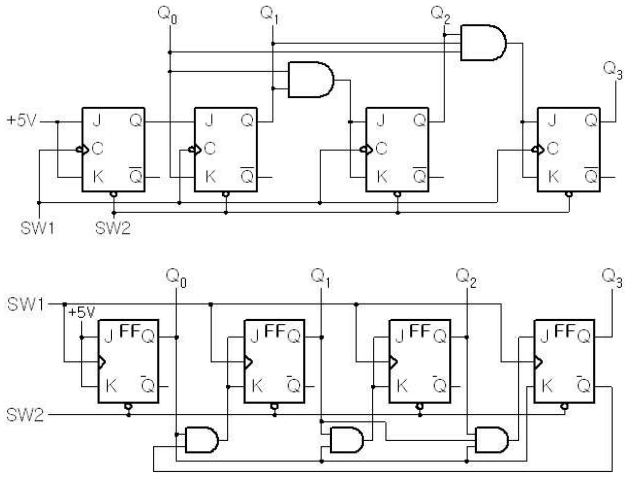
CLK	Q_2	Q_1	Q_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0



(b)

실험

- 다음은 실험할 회로도 임
- 필요 실험부품
  - 2 입력 AND: 7408
  - JK 플립플롭: 7476
- 실험절차



(1) 7476을 이용하여 그림 5(a) 회로를 구성하고, SW2를 0 → 1 로 하여 카운터를 clear 하라. SW1을 이용하여 CLK 입력을 변화시키면서 의 상태를 측정하여 표1에 기록하라.

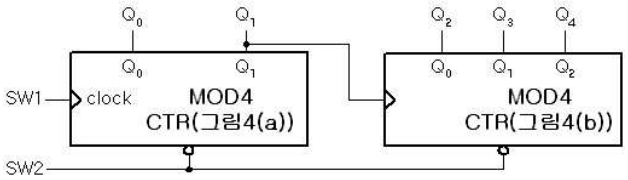
(2) 7476을 이용하여 그림 5(b) 회로를 구성하고, SW2를 0 → 1 로 하여 카운터를 clear 하라. SW1을 이용하여 CLK 입력을 변화시키면서 의 상태를 측정하여 표2에 기록하라.

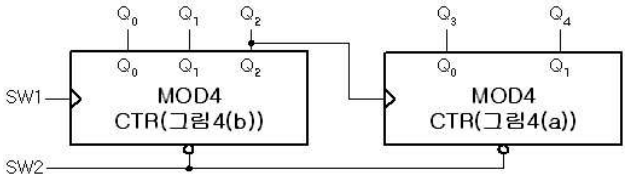
(3) 7476 및 7400을 이용하여 그림 5(c) 회로를 구성하고, SW2를 0 → 1 로 하여 카운터를 clear 하라. SW1을 이용하여 CLK 입력을 변화시키면서 의 상태를 측정하여 표3의 왼쪽에 기록하라.

(4) 7476 및 7400을 이용하여 그림 5(d) 회로를 구성하고, SW2를 0 → 1 로 하여 카운터를 clear 하라. SW1을 이용하여 CLK 입력을 변화시키면서 의 상태를 측정하여 표3의 오른쪽에 기록하라.

제 2 절 결과보고서 다운로드

결과보고서




Q0
Q1
Q2
Q3
Q4
SW1
SW2
MOD4
CTR(그림4(b))
MOD4
CTR(그림4(a))

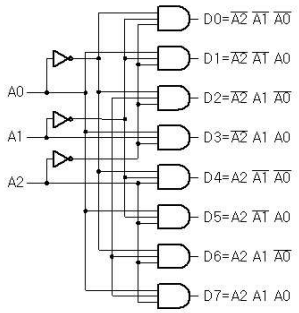
제 9 장

실험9: 디코더 및 인코더

제 1 절 이론 요약

디코더

- 디코더란?
  - 디코더는 코드를 정보화하는 회로
  - n비트의 2진수는 2^n의 정보를 담고 있음
  - 그러므로 보통 $n \times 2^n$ 디코더라 함
- 3×8 디코더
  - 회로도



(a)

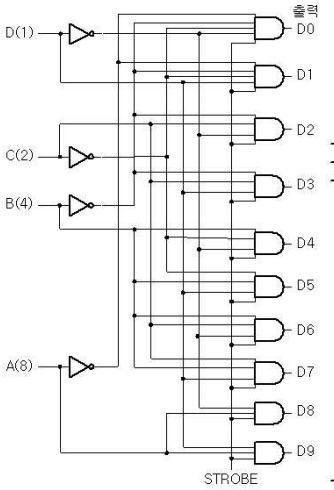
입 력 들			출 력 들							
A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

(b)

- BCD-to-Decimal 디코더
  - 회로도
- 2개의 2×4 디코더를 사용하여 3×8 디코더의 구현
  - 회로도

인코더

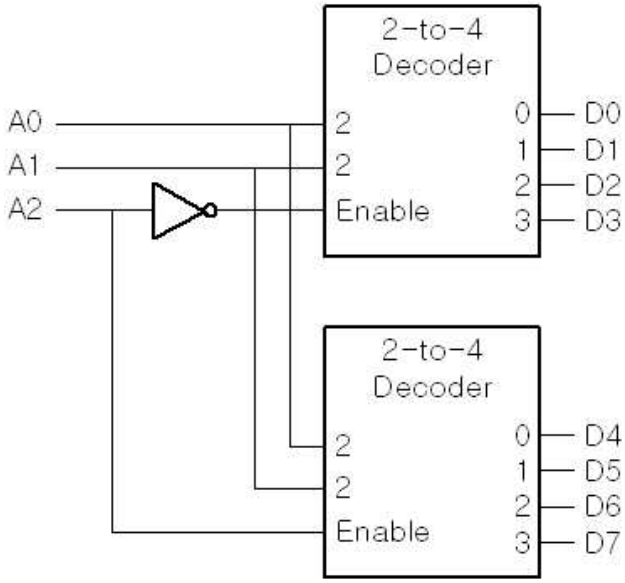
- 인코더란?



(a)

입 력 들				출 력 들									
A	B	C	D	D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	1	1	0	0	0	0	0	1	0	0	0
1	1	0	0	0	1	0	0	0	1	0	0	0	0
1	1	0	1	1	0	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	1	0	0	0	0	0	0
1	1	1	1	1	0	1	0	0	0	0	0	0	0

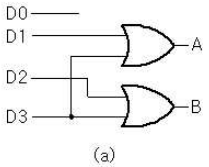
(b)



- 디코더의 반대 동작
- 정보를 코드화함
- 2^n개의 정보가 들어오면 n비트로 코드화 가능

• 4×2 인코더

- 회로도



(a)

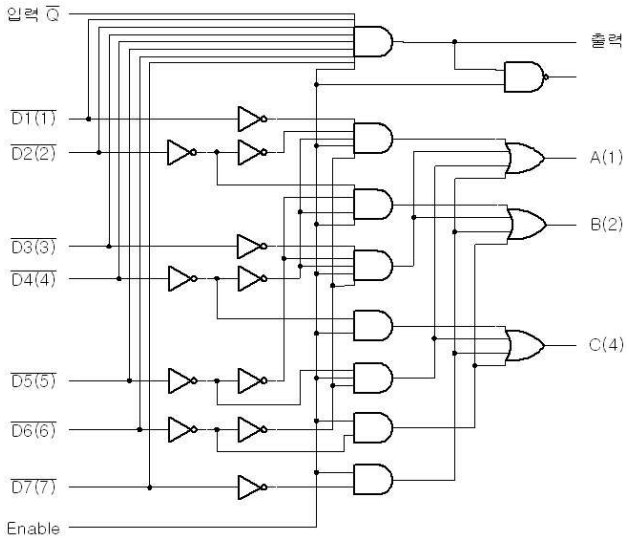
입 력				출력(부호)	
D0	D1	D2	D3	A	B
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

(b)

- 조건
  * 입력중 하나만 1이어야함
  * 동시에 2개가 1이면 정보가 2개이므로 코드화가 부정확해짐

• 우선순위 인코더

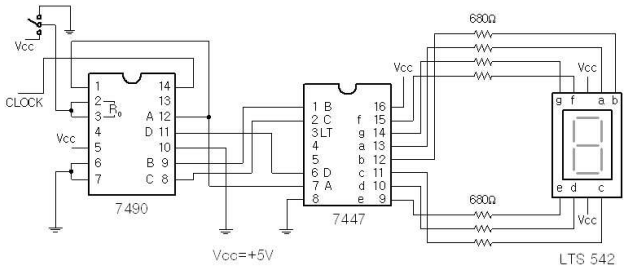
- 입력중 동시에 2개가 1인 경우 우선순위를 두어 코드화를 함
- 회로도

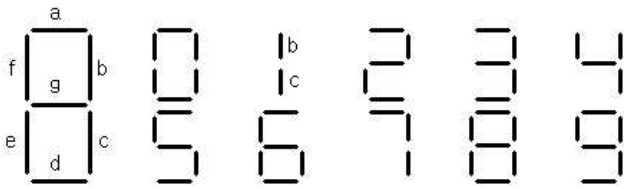


<u>10진 수의 표시</u>

• 방법

- BCD 코드를 사람의 눈으로 보게하기 위해 7개의 LED 로 이루어진 7세그먼트 디코더를 사용
- BCD-to-7 세그먼트디코더를 사용

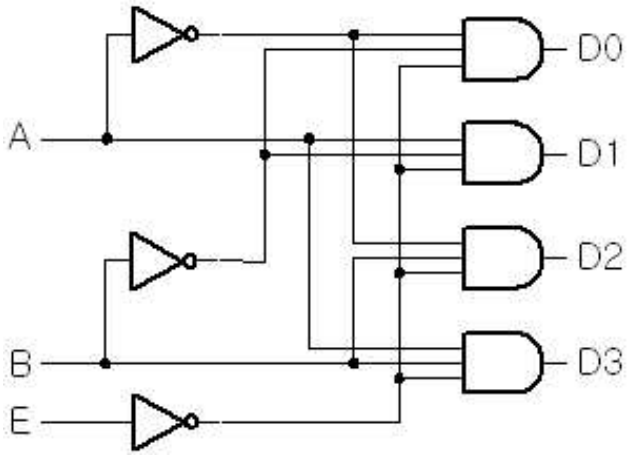




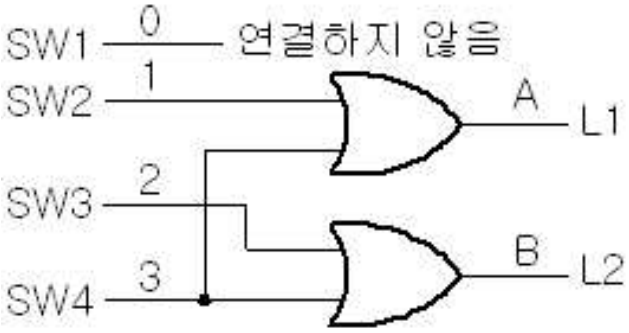
- 회로도
- 7세그먼트의 동작

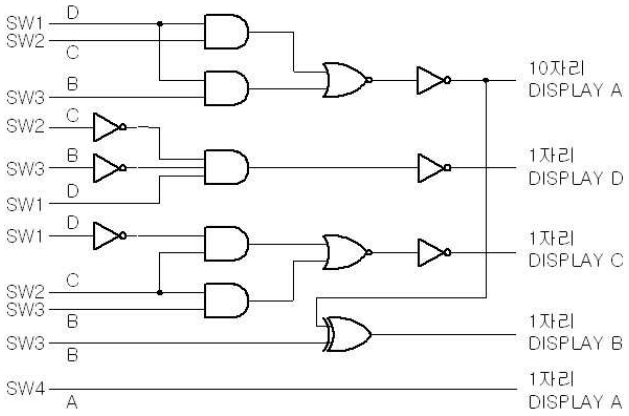
실험

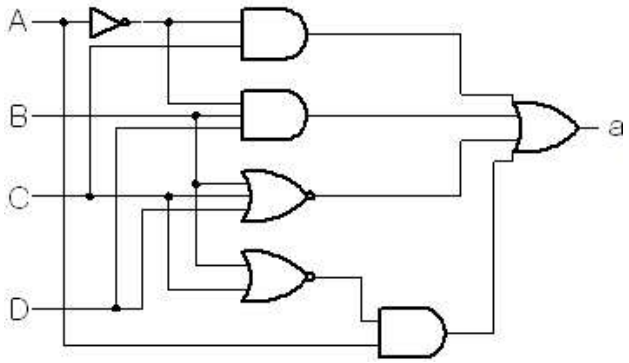
- 다음은 실험할 회로도 임

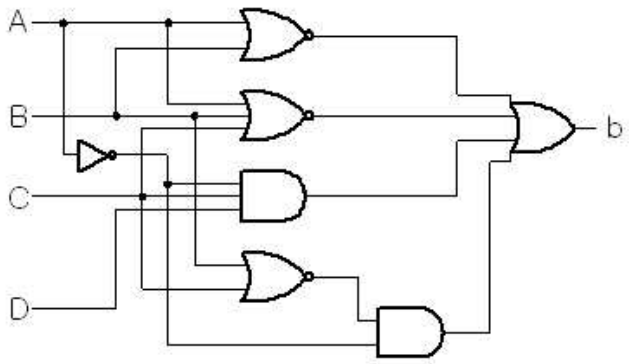


- 필요 실험부품
  - NOT : 7404
  - AND : 7408
  - NAND : 7410
  - OR : 7432
  - NOR : 7402
  - XOR : 7486
  - 10진 카운터 : 7490
  - BCD-to-7-세그먼트 디코더 : 7447


SW1
0
연결하지 않음
SW2
1
A
L1
SW3
2
B
L2
SW4
3


SW1
SW2
D
C
SW3
B
SW2
C
SW3
B
D
SW1
SW1
D
SW2
SW3
C
B
SW3
B
SW4
A
10자리
DISPLAY A
1자리
DISPLAY D
1자리
DISPLAY C
1자리
DISPLAY B
1자리
DISPLAY A


A
B
C
D
a


A
B
C
D
b

- 수치표시기(numeric display with socket) LTS 542

- 실험절차

(1) 7404 및 7408을 이용하여 그림 7(a) 회로를 구성하고, 스위치를 이용하여 표1과 같은 순서로 A, B 입력을 변화시키면서 D0 D3의 상태를 측정하여 표1을 완성하라.

(2) 7432를 이용하여 그림 7(b) 회로를 구성하고, 스위치를 이용하여 표2와 같은 순서로 SW1 SW4의 입력을 변화시키면서 L1, L2의 상태를 측정하여 표2를 완성하라.

(3) 7490, 7447, LTS 542 등을 사용하여 회로도 (c)를 구성하고, 7490의 초기상태를 "0"으로 둔 후 클럭 펄스를 넣으면서 십진수치가 표시되는 것을 확인하라.

(4) 그림 7(d), (e)와 같이 BCD-to-7 세그먼트 디코더를 위한 회로 중 a, b편에 대한 회로를 구성하여 입력 SW0(A) SW3(D)를 변화시키며 표 3을 완성하고 예비보고사항 (3)에서 설계한 BCD-to-7 세그먼트 디코더 회로의 나머지 편에 대한 회로도를 추가하여 표 3을 완성하고 (c)의 7447과 교체한 후 절차 (3)을 반복하여 이 설계가 옳게 되었음을 확인하라.

제 2 절 결과보고서 다운로드

결과보고서

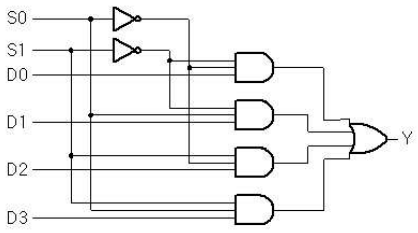
제 10 장

실험10: 멀티플렉서 및 디멀티플렉서

제 1 절 이론 요약

멀티플렉서

- 멀티플렉서란?
  - 여러개의 입력중에서 하나를 선택해 출력해 주는 장치
  - 여러개의 입력중 하나를 선택하는 선택입력이 있음
  - 선택입력은 2^n개의 입력에 대해 n비트가 있어야함
- 4×1 MUX
  - 회로도



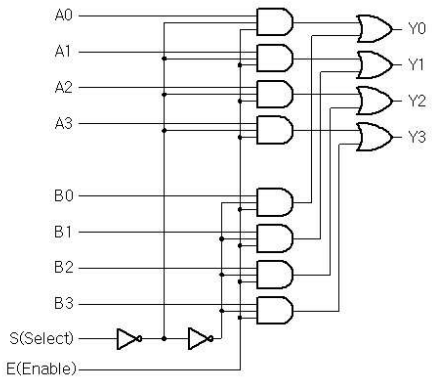
Function table

S1	S0	Y
0	0	D0
0	1	D1
1	0	D2
1	1	D3

- 다중 입/출력 멀티플렉서
  - 두개의 4비트 입력이 그룹으로 하나의 선택선 S 에 의하여 출력됨
  - 회로도
- 멀티플렉서를 이용한 논리회로 구현
  - 선택선과 입력을 이용하여 논리회로 구현
  - 예) F(A,B,C,D) = $\sum_m(1, 3, 4, 11, 12, 13, 14, 15)$

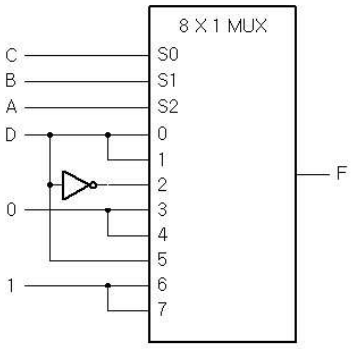
디멀티플렉서

- 동작



Function table

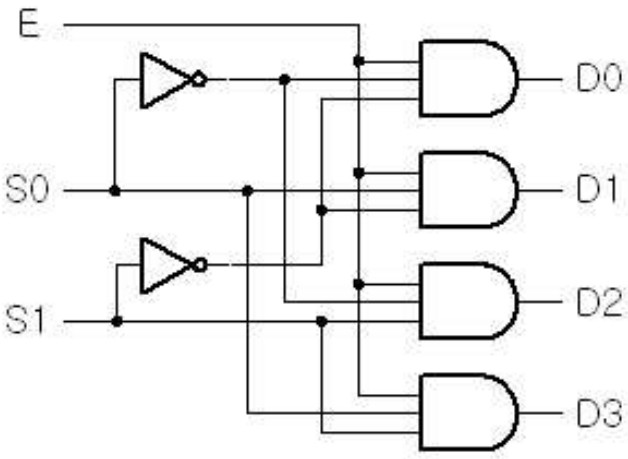
E	S	Output Y
0	X	All 0's
1	0	Select A
1	1	Select B



A	B	C	D	F	
0	0	0	0	0	F=D
0	0	0	1	1	
0	0	1	0	0	F=D
0	0	1	1	1	
0	1	0	0	1	F=D'
0	1	0	1	0	
0	1	1	0	0	F=0
0	1	1	1	0	
1	0	0	0	0	F=0
1	0	0	1	0	
1	0	1	0	0	F=D
1	0	1	1	1	
1	1	0	0	1	F=1
1	1	0	1	1	
1	1	1	0	1	F=1
1	1	1	1	1	

  - 멀티플렉서의 동작을 반대로 수행
  - 즉, 하나의 입력에서 들어온 정보를 여러개로 나누어 출력함
  - 2^n 개의 출력에 대하여 n비트의 선택선이 있어야함

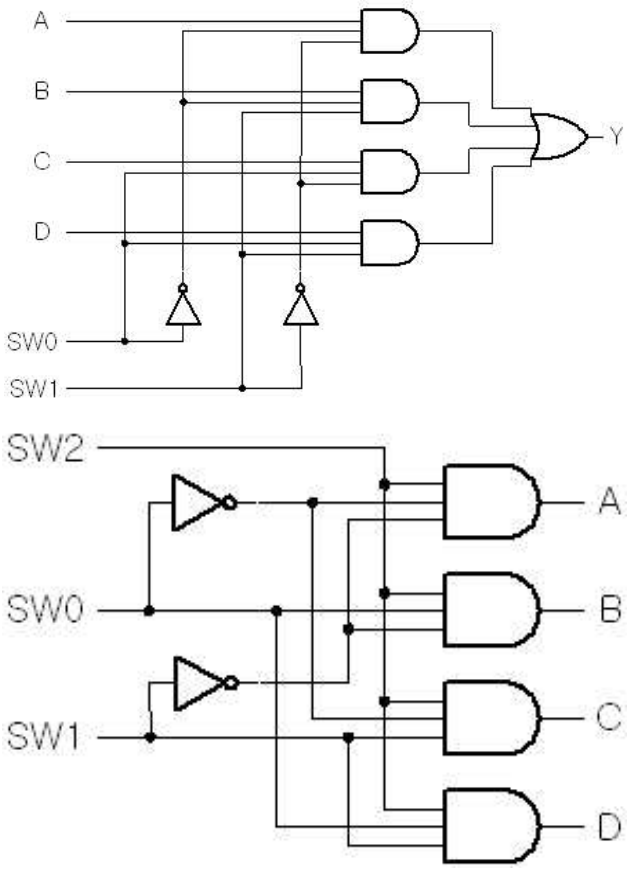
- 1×4 디멀티플렉서
  - 회로도



  - 위 회로도는 enable 이 있는 2×4디코더와 같음

실험

- 다음은 실험할 회로도 임



- 필요 실험부품
  - NOT : 7404
  - AND : 7408
  - OR : 7432
- 실험절차

  (1) 멀티플렉서 회로 그림 5(a)를 구성하고 SW0, SW1 값을 표 1과 같이 변화시키며 Y 값을 읽어서 표1에 기록한다.

  (2) 디멀티플렉서 회로 그림 5(b)를 구성하고 표 2와 같이 입력을 변화시키며 출력값을 읽어서 표 2에 기록한다.

제 2 절 결과보고서 다운로드

결과보고서

제 11 장

실험11: 가산기와 감산기

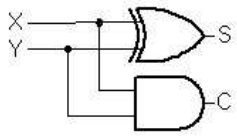
제 1 절 이론 요약

가산기

- 반가산기
  - 2비트 가산기
  - 회로도

반가산기의 진리표

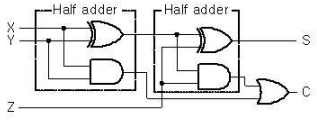
입력들		출력들	
X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



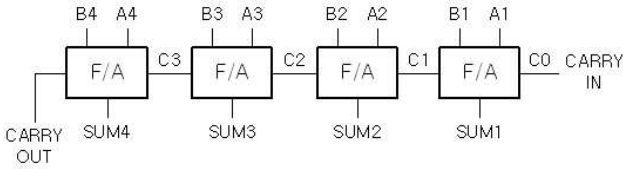
- 전가산기
  - 3비트 가산기
  - 회로도

전가산기의 진리표

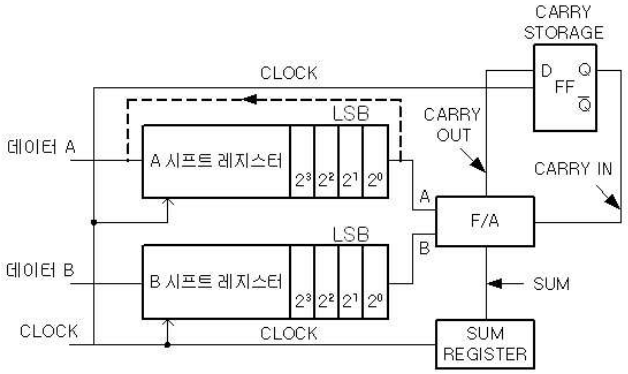
입력들			출력들	
X	Y	Z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



- 병렬가산기
  - 두개의 N비트를 가산하기 위해서는 N개의 전가산기가 필요
- 직렬가산기



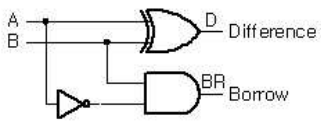
- 한 순간에 한 비트씩 더함
- 하나의 전가산기만 필요 (비용이 적게 듦)
- 단 2개의 N비트를 더하기 N개의 클럭이 필요 (속도 느림)



감산기

- 반감산기
  - 2비트 감산기
  - 회로도

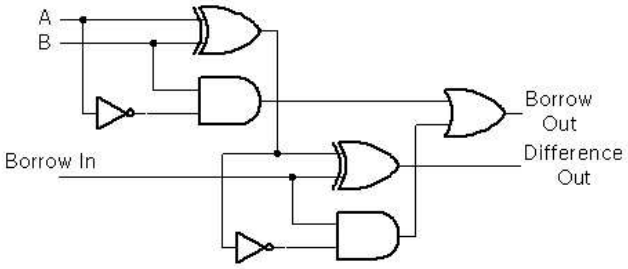
A	B	D	BR
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

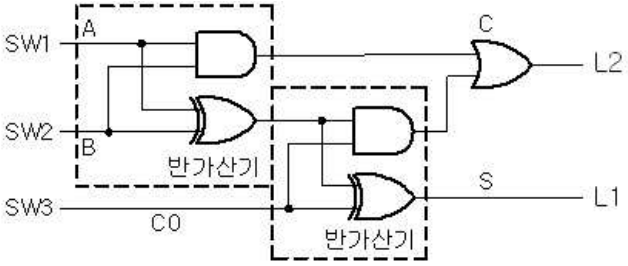


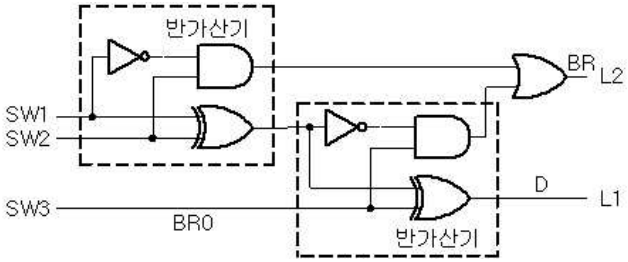
- 전감산기
  - 3비트 감산기
  - 회로도

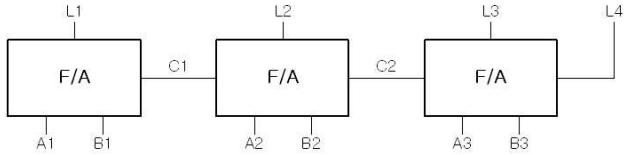
실험

- 다음은 실험할 회로도 임
- 필요 실험부품
  - NOT : 7404
  - AND : 7408


A
B
Borrow In
Borrow Out
Difference Out


SW1
SW2
SW3
A
B
C0
반가산기
C
S
L2
L1
반가산기


반가산기
SW1
SW2
SW3
BR0
BR
L2
D
L1
반가산기


L1
L2
L3
L4
F/A
F/A
F/A
C1
C2
A1
B1
A2
B2
A3
B3

  - OR : 7432
  - XOR : 7486

- 실험절차

  (1) 7408, 7432, 7486을 이용하여 회로도 (a)의 전가산기를 구성하고 SW1, SW2, SW3를 변화시켜가며 표 1을 완성하라.

  (2) 7404, 7408, 7432, 7486을 이용하여 회로도 (b)의 전감산기를 구성하고 SW1, SW2, SW3를 변화시켜가며 표 1을 완성하라.

  (3) 회로도 (a)를 이용하여 회로도 (c)와 같이 3 비트 병렬 가산기를 구성하여 입력 A1, B1 A4, B4를 변화시키며 표 2를 완성하라.

제 2 절 결과보고서 다운로드

결과보고서

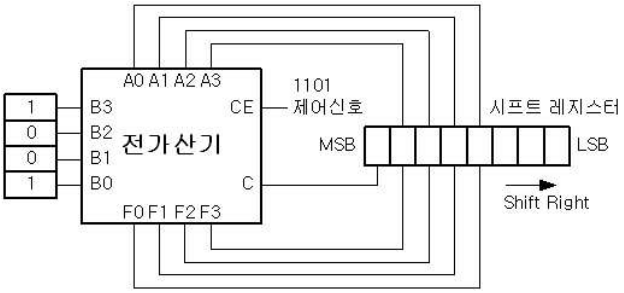
제 12 장

실험12: 곱셈기와 논리연산장치

제 1 절 이론 요약

곱셈기

- 곱셈기란?
  - 두개의 2진수를 곱하는 장치 (1001 x 1101의 예)
  - 회로도

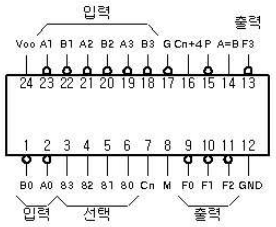


  - 동작
    * 1101 의 하위비트부터 제어신호에 입력 즉 1011을 입력
    * 이때 제어신호가 1이면 더하고 0이면 더하지 않음
    * 한번 계산이 끝나면 시프트 레지스터를 오른쪽으로 한비트 시프트
    *

논리연산장치(ALU)

- ALU 는?
  - 여러가지 연산을 수행하는 장치
  - 4비트 ALU 인 74181 사용
  - 그림

실험

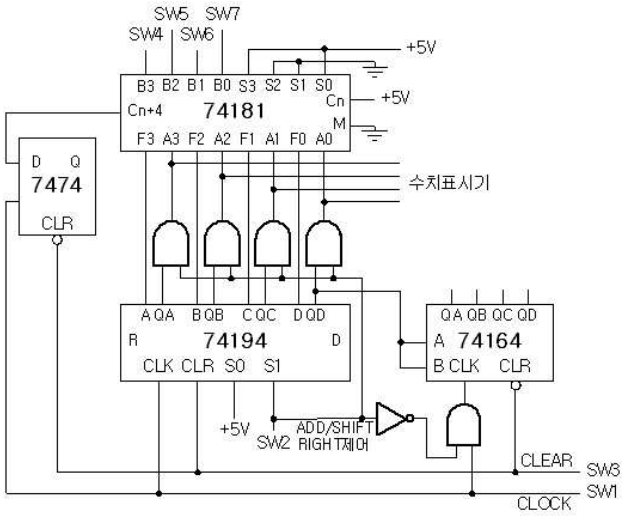


(a)

1	A	9	A+B
2	A′	10	(A+B)′
3	B	11	A′B
4	B′	12	뮤′
5	AA′=0	13	A′+B
6	A+A′=1	14	A+B′
7	AB	15	A⊕B
8	(AB)′	16	(A⊕B)′

(b)

연산선택입력				논리	연산
S_3	S_2	S_1	S_0	(M=HOGH)	((M=LOW) (Cn=HIGH)
0	0	0	0	$\overline{A}$	A
0	0	0	1	$\overline{A+B}$	A+B
0	0	1	0	$\overline{AB}$	$A+\overline{B}$
0	0	1	1	Logic 0	Minus 1
0	1	0	0	$\overline{AB}$	A plus $A\overline{B}$
0	1	0	1	$\overline{B}$	(A+B) plus AB
0	1	1	0	$A\oplus B$	A minus B minus 1
0	1	1	1	$A\overline{B}$	$A\overline{B}$ minus 1
1	0	0	0	$\overline{A}+B$	A plus AB
1	0	0	1	$\overline{A\oplus B}$	A plus B
1	0	1	0	B	$(A+\overline{B})$ plus AB
1	0	1	1	AB	AB minus 1
1	1	0	0	Logic 1	A plus A
1	1	0	1	$A+\overline{B}$	(A+B) plus A
1	1	1	0	A+B	$(A+\overline{B})$ plus A
1	1	1	1	A	A minus 1



- 다음은 실험할 회로도 임
- 필요 실험부품
  - NOT : 7404
  - AND : 7408
  - D 플립플롭 : 7474
  - 8 비트 시프트 레지스터 : 74164
  - 4 비트 ALU : 74181
  - 만능 시프트 레지스터 : 74194
  - 수치표시기(numeric display with socket) LTS 542
- 실험절차

  (1) 그림 4와 같은 실험회로도를 구성하고 SW1을 클럭 입력에, SW2를 Add/Shift Right(Add="1", Shift Right="0"), SW3를 "0" 상태에, SW4 SW7을 곱해지는 수에 각각 연결시킨다.

  (2) 1011을 SW4 SW7에 넣고 1101을 SW2 제어신호로 사용하여 곱셈 1101 x 1011을 수행한 후 표 1에 기록한다.

  (3) 임의의 다섯 쌍의 값들을 더 선정해서 곱셈을 수행한 후 표 1에 기록한다..

  (4) 그림 4의 회로도를 변경하여 덧셈과 뺄셈을 하여 표 1을 완성하라.

제 2 절 결과보고서 다운로드

결과보고서