

디지털논리회로 - Digital Logic Circuit

디지털논리회로 - Digital Logic Circuit

1 컴퓨터와 디지털 논리회로

1. 디지털 시스템
 - 시스템 정의
2. 아날로그와 디지털 비교
 - 데이터 표현 방법
3. 컴퓨터의 구성
 - 컴퓨터 시스템의 구성요소
 - 컴퓨터 하드웨어 5대 구성요소
 - 디지털 컴퓨터의 소개
4. 집적회로
 - 집적도
 - 표준특성
5. 양논리와 음논리

2 데이터 표현

1. 수치데이터
2. 디지털 코드

3 부울대수와 논리게이트

1. 논리연산과 논리게이트
 - 논리연산
 - 논리연산의 종류
 - 논리게이트의 종류 및 특성
 - 논리회로
 - 기본 논리게이트
 - 기타 논리게이트
2. 부울대수 (Boolean algebra)
 - 부울대수의 기본공식
 - 부울함수의 대수적 간소화
 - 대수적 간소화에서 주로 사용되는 방법
3. 부울함수의 정규형 및 표준형
 - 정규형
 - 최소항과 최대항
 - 최소항의 합
 - 최대항의 곱
 - 정규형 간의 관계
 - 표준형 (standard form)
 - 곱의 합
 - 합의 곱
4. 집적회로

4 부울함수의 간소화 및 구현

1. 개요
 - 부울함수의 간소화 방법
2. 카르노도표 도표방법
 - 개요

- 인접 사각형
 - 인접 사각형의 정의
 - 인접 사각형끼리 묶는 방법
- 2변수 카르노도표
- 3변수 카르노도표
- 4변수 카르노도표
- 5변수 카르노도표
- 6변수 카르노도표
- 무관조건
- 기타 카르노도표

3. NAND 게이트와 NOR 게이트

- NAND 게이트
- NOR 게이트

5 조합논리회로

1. 개요

- 디지털 시스템을 구성하는 논리회로
- 조합논리회로의 구성

2. 조합논리회로의 분석과 설계

- 조합논리회로의 분석
 - 부울함수의 유도
 - 진리표 작성
- 조합논리회로의 설계

3. 기본 연산회로

- 가산기
 - 반가산기 (HA)
 - 전가산기 (FA)
 - 직병렬 가산기
 - 직렬방법
 - 병렬방법
- 감산기
 - 반감산기 (HS)
 - 전감산기 (FS)
 - 병렬 가,감산기
 - BCD 가산기

4. 여러가지 조합논리회로

- 코드변환기
 - BCD-3초과 코드변환기
 - BCD-9의 보수변환기
- 패리티 발생기/검사기
 - 패리티 발생기
 - 패리티 검사기

5. MSI를 이용한 조합논리회로

- 인코더/디코더
 - 인코더(encoder)
 - 디코더(decoder)
 - 구동입력을 가진 디코더
 - 디코더의 확장
 - 디코더를 이용한 부울함수 구현
- 멀티플렉서/디멀티플렉서

- 멀티플렉서(multiplexer)
- 멀티플렉서를 이용한 부울함수 구현
- 디멀티플렉서(demultiplexer)

6 순서논리회로

1. 개요
2. 플립플롭
 - SR 래치
 - NOR 게이트로 된 SR 래치
 - NAND 게이트로 된 SR 래치
 - 제어입력을 가진 SR 래치(RS 플립플롭)
 - D 플립플롭
 - JK 플립플롭
 - T 플립플롭
3. 플립플롭의 트리거링
 - 트리거(trigger)
 - 레벨 트리거 방법
 - 에지 트리거 방법
 - 마스터-슬레이브 플립플롭
4. 순서논리회로의 분석
 - 입력방정식의 유도
 - 상태표 작성
 - 특성표
 - 상태도 (state diagram)
5. 순서논리회로의 설계
 - 플립플롭의 결정
 - 입력방정식의 유도
 - 입력방정식
 - 플립플롭의 여기표

7 레지스터와 카운터

1. 레지스터
 - 개요
 - 데이터 적재 레지스터
 - 직렬적재 레지스터
 - 병렬적재 레지스터
 - 시프트 레지스터 (shift register)
 - 병렬적재 양방향 시프트 레지스터
2. 카운터
 - 개요
 - 비동기식 카운터
 - 2진 리플 카운터
 - BCD 리플 카운터
 - 동기식 카운터
 - 2진 카운터
 - 모듈로 - N 카운터
 - 시프트 카운터
 - 링 카운터
 - 존슨 카운터
 - 카운터의 설계
3. 메모리셀

8 기억장치와 PLD

1. 개요

- 기억장치의 특성
- 기억장치의 종류

2. RAM

- 종류
- 구성
- 확장

3. ROM

- 종류
- 구조
- ROM을 이용한 조합논리회로 구현

4. PLD

- PLA
- PAL

1 컴퓨터와 디지털 논리회로

1. 디지털 시스템

- 시스템 정의

1. 검은상자형 시스템

입력(input)과 출력(output)이 있는 검은상자(black box)로 표현

검은 상자란 그 내부에 대해서는 큰 의미를 부여하지 않는다는 뜻 내포

입력과 출력에 관심을 갖고 해당 시스템을 고려하는 것

2. 구성요소의 집합으로서의 시스템

주로 검은상자 내부에 관해 규정하는 것

● 시스템

시스템에 부여된 목적(goal)을 달성하기 위해 상호작용(interaction)하는 구성요소(component)들의 집합

2. 아날로그와 디지털 비교

- 데이터 표현 방법

3. 컴퓨터의 구성

- 컴퓨터 시스템의 구성요소

하드웨어, 소프트웨어, 데이터, 프로세서, 사람

- 컴퓨터 하드웨어 5대 구성요소

입력장치, 기억장치, 연산장치, 제어장치, 출력장치

- 디지털 컴퓨터의 소개

- 디지털 컴퓨터

대표적인 디지털 시스템의 예로 하드웨어 구성을 따름

- 입력장치

컴퓨터에 들어가는 정보의 형태에 따라 여러 가지 형태를 띠

ex) 자기테이프, 키보드, 마우스, 스캐너

- 출력장치

프린터, 플로터(plotter), CRT(Cathode Ray Tube) 모니터(monitor) 또는 LCD 채널

- 연산장치와 제어장치(처리기 또는 프로세서)

수치연산 및 논리연산(가산, 감산, 승산, 제산, AND, OR, NOT 등)

자료의 전송(레지스터, 기억장치 사이 및 입출력 장치 사이의 자료 교환)

다음에 행하는 연산의 결정

4. 집적회로

디지털 게이트(digital gate)의 기능을 수행하는 전자소자를포함하는 작은 실리콘 반도체 크리스탈(silicon semiconductor crystal)로 칩(chip)이라고도 함

- 집적도

단위 실리콘칩에 집적할 수 있는 게이트의 수

- 소규모 집적(SSI: Small Scale Integration)

단위 패키지 내에 몇 개의 독립된 게이트를 내장하고 있는 게이트

내장한 게이트의 수는 대체로 10개 이하

- 중규모 집적(MSI: Medium Scale Integration)

단위 패키지 내에 대략 10 ~ 100개의 게이트를 내장한 복잡한 구조

대체로 특정 디지털 함수, 디코더(decoder), 가산기(adder), 레지스터(register)로서의 기능 수행

- 대규모 집적(LSI: Large Scale Integration)

단위 패키지당 수백 ~ 수천 개의 게이트 내장

프로세서(processor), 기억장치 칩(memory chip), 프로그램 가능한 모듈(programmable module) 등과 같은 디지털 시스템 포함

- 초대규모 집적(VLSI: Very LSI)

단위 패키지 내에 수천 개의 게이트 내장

대규모 기억장치 배열과 복잡한 마이크로컴퓨터칩, 등

-표준특성

평가에 사용되는 주요 기준

1. 팬-아웃(fan-out)

정상동작에 영향을 주지 않고 게이트 출력부에 연결할 수 있는 표준부하의 수

2. 전력소모(power dissipation)

게이트가 동작할 때 필요한 전력

3. 전파지연시간(propagation delay)

2진 신호가 그 값이 바뀌었을 때 입력에서 출력까지 신호의 변화가 전달되는 데 걸리는 평균시간

4. 잡음여유(noise margin)

회로의 출력을 바꾸지 않으면서 입력에 첨가되는 최대 잡음전압

5. 양논리와 음논리

게이트의 입력과 출력에 대한 2진 신호는 상태가 변하는 동안을 제외하고 두 값 중에 한 값을 가짐

- 양논리 시스템(positive logic system)

높은 신호값 H를 논리값 1로 하고 낮은 신호값 L을 논리값 0으로 할당하는 것

- 음논리 시스템(negative logic system)

H를 논리값 0, L을 논리값 1로 할당하는 것

양, 음은 단지 두 신호값에 논리값을 할당하는 방법을 나타낸 것

2 데이터 표현

1. 수치데이터

2. 디지털 코드

컴퓨터시스템구조 > 2 데이터 표현 및 연산 참고

3 부울대수와 논리게이트

1. 논리연산과 논리게이트

- 논리연산

2개의 이산값에 적용되는, 논리적 의미를 갖는 연산

- 논리연산의 종류

- AND 연산: $XY = F$

- **OR 연산** : $X + Y = F$
- **NOT 연산** : $\overline{B} = F$

- 논리게이트의 종류 및 특성

- 논리회로

적절하게 입력된 신호를 가지고 논리적 조작을 수행

출력신호를 생성하는 전자적 회로로 이때 사용되는 정보는 0이나 1을 나타내는 2진 신호로 표시

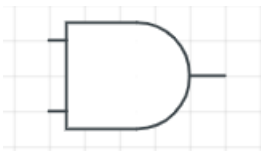
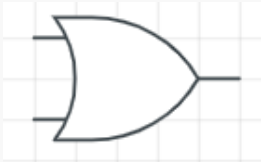

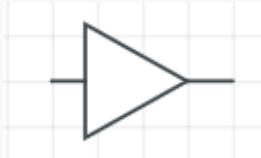
각각 2개의 분리된 전압 레벨의 출력으로 나타낼 수 있음

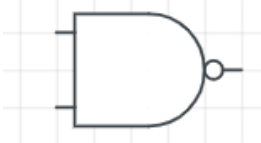
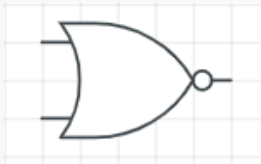
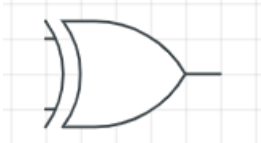
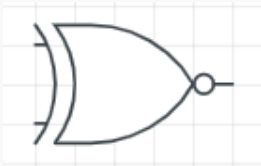
- 기본 논리게이트

- **AND 게이트**: 논리적 곱 수행
- **OR 게이트**: 논리적 합 수행
- **NOT 게이트**: 인버터(inverter) 수행

- 기타 논리게이트

- **NAND 게이트**: 논리적 곱의 보수(AND-NOT) 수행
- **NOR 게이트**: 논리적 합의 보수(OR-NOT) 수행
- **XOR 게이트**: 비교연산 수행
- **XNOR 게이트**: XOR 보수, 비교연산 수행

기호이름	변별기호	대수식	진리표															
AND		$F = XY$	<table><tr><th>X</th><th>Y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	X	Y	F	0	0	0	0	1	0	1	0	0	1	1	1
X	Y	F																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR		$F = X + Y$	<table><tr><th>X</th><th>Y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	X	Y	F	0	0	0	0	1	1	1	0	1	1	1	1
X	Y	F																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
NOT		$F = \overline{X}$	<table><tr><th>X</th><th>F</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	X	F	0	1	1	0									
X	F																	
0	1																	
1	0																	
버퍼		$F = X$	<table><tr><th>X</th><th>F</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	X	F	0	0	1	1									
X	F																	
0	0																	
1	1																	

NAND		$F = \overline{XY}$	<table> <tr> <th>X</th> <th>Y</th> <th>F</th> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </table>	X	Y	F	0	0	1	0	1	1	1	0	1	1	1	0
X	Y	F																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NOR		$F = \overline{X + Y}$	<table> <tr> <th>X</th> <th>Y</th> <th>F</th> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </table>	X	Y	F	0	0	1	0	1	0	1	0	0	1	1	0
X	Y	F																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
XOR (배타적-OR)		$F = X\overline{Y} + \overline{X}Y = X \oplus Y$	<table> <tr> <th>X</th> <th>Y</th> <th>F</th> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </table>	X	Y	F	0	0	0	0	1	1	1	0	1	1	1	0
X	Y	F																
0	0	0																
0	1	1																
1	0	1																
1	1	0																
XNOR (배타적-NOR)		$F = XY + \overline{X}\overline{Y} = \overline{X \oplus Y}$	<table> <tr> <th>X</th> <th>Y</th> <th>F</th> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </table>	X	Y	F	0	0	1	0	1	0	1	0	0	1	1	1
X	Y	F																
0	0	1																
0	1	0																
1	0	0																
1	1	1																

2. 부울대수 (Boolean algebra)

2진 변수와 논리연산을 다루는 대수

변수는 영문자로 표시

논리연산에는 AND, OR와 보수의 세 가지 연산이 있음

- 부울함수(Boolean function)

논리변수와 상호관계를 나타내기 위해 부울변수, 부울상수(0, 1), 부울연산기호(·, +, -), 괄호 및 등호 등으로 나타내는 대수적인 표현

부울함수를 진리표로 나타낼 때는 진리표 하나로 결정

부울함수를 대수식으로 나타낼 때는 다양한 형태의 식으로 표현할 수 있음

- 부울대수의 기본공식

번호	기본공식	번호	기본공식	
1	$X + 0 = X$	2	$X \cdot 1 = X$	
3	$X + 1 = 1$	4	$X \cdot 0 = 0$	
5	$X + X = X$	6	$X \cdot X = X$	
7	$X + \overline{X} = 1$	8	$X \cdot \overline{X} = 0$	
9	$\overline{\overline{X}} = X$			
10	$X + Y = Y + X$	11	$XY = YX$	교환법칙
12	$X + (Y + Z) = (X + Y) + Z$	13	$X(YZ) = (XY)Z$	결합법칙
14	$X(Y + Z) = XY + XZ$	15	$X + YZ = (X + Y)(X + Z)$	분배법칙
16	$\overline{X + Y} = \overline{X} \cdot \overline{Y}$	17	$\overline{X \cdot Y} = \overline{X} + \overline{Y}$	드모르간의 법칙
18	$X + X \cdot Y = X$	19	$X \cdot (X + Y) = X$	흡수정리

- 부울함수의 대수적 간소화

부울대수의 공식을 이용하여 변환한 후, 변환된 여러 함수 중에서 가장 간단한 형태의 함수를 찾아내는 것

- 대수적 간소화에서 주로 사용되는 방법

- 항 결합

$$XY = X\overline{Y} = X(Y + \overline{Y}) = X \cdot 1 = X$$

- 문자 소거

$$X + \overline{X}Y = (X + \overline{X})(X + Y) = 1 \cdot (X + Y) = X + Y$$

$$X(\overline{X} + Y) = X\overline{X} + XY = 0 + XY = XY$$

- 중복항 첨가

$$\begin{aligned}
 F &= X\overline{Y}Z + XYZ + \overline{X}YZ \\
 &= X\overline{Y}Z + XYZ + XYZ + \overline{X}YZ \\
 &= XZ(\overline{Y} + Y) + YZ(X + \overline{X}) \\
 &= XZ + YZ
 \end{aligned}$$

3. 부울함수의 정규형 및 표준형

- 정규형

부울함수를 표현하는 대표적인 방법 중 하나

함수의 항이 **최소항의 합(sum of minterm)**이나 **최대항의 곱(product of maxterm)**으로 표현된 식

이 외의 함수는 비정규형(non-canonical form)

- 최소항과 최대항

- 최소항

n개의 논리변수로 구성되는 부울함수에서 최소항이란 각 변수의 문자 1개씩 모두 n개의 문자의 논리곱항으로서 그 결과 논리-1인 경우

- **최대항**

n개의 논리변수로 구성되는 부울함수에서 최대항이란 각 변수의 문자 1개씩 모두 n개의 문자의 논리합항으로서 그 결과 논리-1인 경우

X	Y	Z	최소항-항	최소항-표시	최대항-항	최대항-표시
0	0	0	$\overline{X}\overline{Y}\overline{Z}$	m_0	$X + Y + Z$	M_0
0	0	1	$\overline{X}\overline{Y}Z$	m_1	$X + Y + \overline{Z}$	M_1
0	1	0	$\overline{X}Y\overline{Z}$	m_2	$X + \overline{Y} + Z$	M_2
0	1	1	$\overline{X}YZ$	m_3	$X + \overline{Y} + \overline{Z}$	M_3
1	0	0	$X\overline{Y}\overline{Z}$	m_4	$\overline{X} + Y + Z$	M_4
1	0	1	$X\overline{Y}Z$	m_5	$\overline{X} + Y + \overline{Z}$	M_5
1	1	0	$XY\overline{Z}$	m_6	$\overline{X} + \overline{Y} + Z$	M_6
1	1	1	XYZ	m_7	$\overline{X} + \overline{Y} + \overline{Z}$	M_7

- 최소항의 합

부울함수에서 n개의 2진 변수는 최대 2^n 개의 서로 다른 최소항(AND 연산) 구성할 수 있음

구성된 최소항을 OR 연산으로 결합하여 곱의 합으로 표시

ex) 부울함수 $F = X + Y\overline{Z}$ 를 최소항의 합으로 표현

$$\begin{aligned}
 F &= X + Y\overline{Z} \\
 &= X(Y + \overline{Y}) + (X + \overline{X})Y\overline{Z} \\
 &= XY + X\overline{Y} + XY\overline{Z} + \overline{X}Y\overline{Z} \\
 &= XY(Z + \overline{Z}) + X\overline{Y}(Z + \overline{Z}) + XY\overline{Z} + \overline{X}Y\overline{Z} \\
 &= XYZ + XY\overline{Z} + X\overline{Y}Z + X\overline{Y}\overline{Z} + XY\overline{Z} + \overline{X}Y\overline{Z} \\
 &= XYZ + XY\overline{Z} + X\overline{Y}Z + X\overline{Y}\overline{Z} + \overline{X}Y\overline{Z} \\
 &= m_2 + m_4 + m_5 + m_6 + m_7
 \end{aligned}$$

$$\therefore F(X, Y, Z) = \sum m(2, 4, 5, 6, 7)$$

- 최대항의 곱

n개의 변수로 구성된 2^n 가지 함수들의 각각은 최대항의 곱으로 표현 가능

ex) 부울함수 $F = XY + \overline{X}Z$ 를 최소항의 합으로 표현

$$F = XY + \overline{X}Z$$

$$\begin{aligned}
&= (XY + \overline{X}) + (XY + Z) \\
&= (X + \overline{X})(Y + \overline{X})(X + Z)(Y + Z) \\
&= (\overline{X} + Y)(X + Z)(Y + Z) \\
&= (\overline{X} + Y + Z\overline{Z})(X + Y\overline{Y} + Z)(X\overline{X} + Y + Z) \\
&= (\overline{X} + Y + Z)(\overline{X} + Y + \overline{Z})(X + Y + Z)(X + \overline{Y} + Z)(X + Y + Z)(\overline{X} + Y + Z) \\
&= (X + Y + Z)(X + \overline{Y} + Z)(\overline{X} + Y + Z)(\overline{X} + Y + \overline{Z}) \\
&= M_0 \cdot M_1 \cdot M_2 \cdot M_3
\end{aligned}$$

$$\therefore F(X, Y, Z) = \prod M(0, 2, 4, 5)$$

-정규형 간의 관계

최소항의 합으로 표시된 함수의 보수는 원래 함수에서 제외된 최소항의 합으로,

어떤 함수를 표현하는 최소항은 함수를 1로 하지만, 그것의 보수를 표현하는 최소값은 처음 함수를 0으로 함

$$F = (X, Y, Z) = \sum m(2, 4, 5, 6, 7) = m_2 + m_4 + m_5 + m_6 + m_7$$

$$\overline{F} = (X, Y, Z) = \sum m(0, 1, 3) = m_0 + m_1 + m_3$$

$$F = (X, Y, Z) = \overline{\overline{F}}(X, Y, Z) = \prod M(0, 1, 3) = \sum m(2, 4, 5, 6, 7)$$

$$\therefore \overline{m_i} = M_i$$

- 표준형 (standard form)

부울함수를 표현하는 또다른 방법

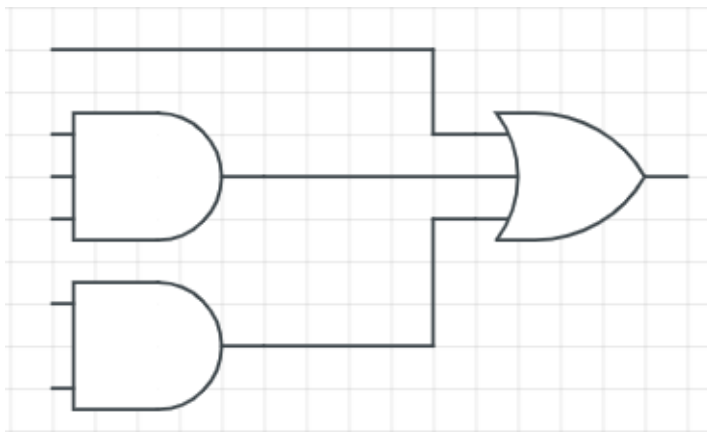
표준형의 각 항은 하나 또는 그 이상의 문자로 구성

곱의 합(sum of products)과 합의 곱(product of sums) 있음

- 곱의 합

부울함수에서 n개의 2진 변수는 최대 2^n 개의 서로 다른 최소항(AND 연산)을 구성할 수 있음

구성된 최소항을 OR 연산으로 결합

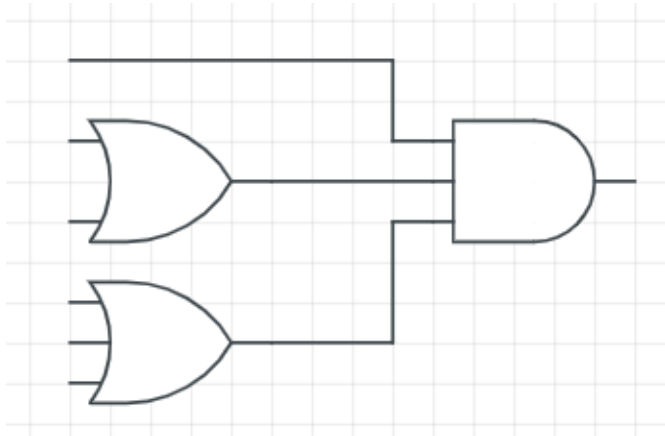


$$\blacksquare F = \overline{Y} + \overline{X}Y\overline{Z} + XY$$

- 합의 곱

합항의 논리곱 형태

각 논리합 항은 임의 개수의 리터럴(literal)을 가짐



$$\leftarrow F = X(\bar{Y} + Z)(Z + Y + \bar{Z})$$

4. 집적회로

4 부울함수의 간소화 및 구현

1. 개요

- 부울함수의 간소화 방법

- 대수적인 방법(algebraic method)

주어진 부울함수에 대해 부울대수의 정리를 대수적으로 적용하는 방법

- 도표방법(map method)

카르노도표(Karnaugh Map)라 불리는 도표를 사용하는 방법

부울함수의 각항은 하나의 곱형태로 쉽게 간소화

변수 많아짐에 따라 구성 복잡해져, 실제로 6개 또는 그 이하의 변수를 가진 변수만 대상으로 함

- 테이블 방법(tabular method)

퀸-맥클러스키 방법(Quine-Mccluskey Method)이라 함

테이블을 사용하여 간소화 알고리즘(minimization algorithm)을 쉽게 구현

많은 변수를 갖고 있는 부울함수의 간소화에 적당

2. 카르노도표 도표방법

- 개요

여러 개의 사각형으로 된 그림

사각형은 각각 하나의 최소항 또는 최대항 나타냄

직관적으로 부울함수 간소화

입력변수의 수가 n 인 경우 n 수 카르노도표라고 하며 2^n 개의 사각형으로 구성

- 인접 사각형

- 인접 사각형의 정의

카르노도표에서 각 정사각형은 하나의 최소항

부울공식인 $X + \overline{X} = 1$ 을 이용해 간소화함

- 인접 사각형끼리 묶는 방법

한 묶음 내의 정사각형 수는 $2n$ ($n = 0, 1, 2, \dots, n$)개가 되도록 묶음

한 묶음은 크게, 전체 묶음의 수는 작게 묶음

- 2변수 카르노도표

	Y	
	0	1
X 0	0	1
1	1	1

◀ ex) $F(X, Y) = \sum m(1, 2, 3) = \overline{X}Y + X\overline{Y} + XY = X + Y$

- 3변수 카르노도표

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

◀ ex) $F(X, Y, Z) = \sum m(0, 1, 2, 6) = \overline{X}\overline{Y} + Y\overline{Z}$

- 4변수 카르노도표

		Y,Z			
		00	01	11	10
W,X	00	0	1	1	0
	01	1	1	0	0
	11	1	1	0	0
	10	0	0	1	0

◀ ex)

$$F(W, X, Y, Z) = \sum m(1, 3, 4, 5, 11, 12, 13) = X\overline{Y} + \overline{W}\overline{X}Z + \overline{W}YZ$$

- 5변수 카르노도표

- 6변수 카르노도표

- 무관조건

논리회로에서는 입력변수들의 조합에 따라서 함숫값이 발생하지 않는 경우나 0이나 1 중 어떠한 함숫값이 출력값으로 나와도 무관한 경우가 있음

이 무관조건(don't care condition)은 함수를 더욱 간소화하는 데 사용됨

$$\text{ex) } F(W, X, Y, Z) = \sum m(0, 3, 6, 9)$$

$$\text{단, 무관조건은 } d(W, X, Y, Z) = \sum m(10, 11, 12, 13, 14, 15)$$

		Y,Z			
		00	01	11	10
W,X	00	1	0	1	0
	01	0	0	0	1
	11	-	-	-	-
	10	0	1	-	-

☐ 곱의 합형 $F = \overline{W}\overline{X}\overline{Y}\overline{Z} + WZ + \overline{X}\overline{Y}\overline{Z} + XY\overline{Z}$

		Y,Z			
		00	01	11	10
W,X	00	1	0	1	0
	01	0	0	0	1
	11	-	-	-	-
	10	0	1	-	-

☐ 합의 곱형 $(\overline{X} + Y)(\overline{X} + \overline{Z})(\overline{W} + \overline{Z})(X + \overline{Y} + Z)$

- 기타 카르노도표

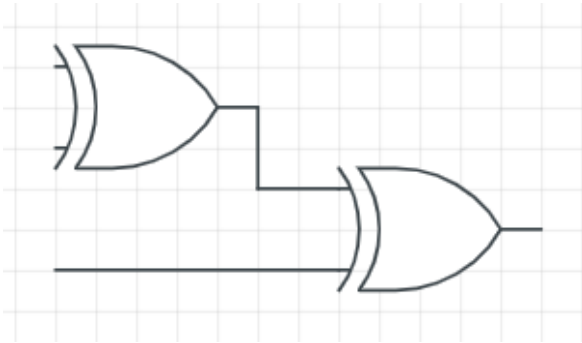
XOR 논리게이트는 등가의 논리식으로 바꾸어 일반의 논리함수로 변환 가능

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

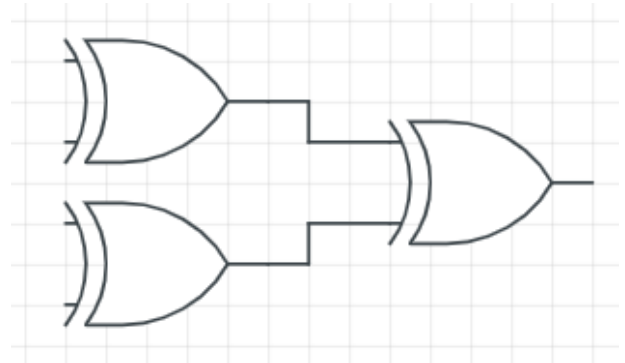
$$X \oplus Y \oplus Z = (\overline{X}\overline{Y} + \overline{X}Y)\overline{Z} + (XY + \overline{X}\overline{Y})Z = \overline{X}\overline{Y}\overline{Z} + \overline{X}Y\overline{Z} + \overline{X}\overline{Y}Z + XYZ$$

		C,D			
		00	01	11	10
A,B	00	0	1	0	1
	01	1	0	1	0
	11	0	1	0	1
	10	1	0	1	0

◀ $A \oplus B \oplus C \oplus D$



▲ $P = X \oplus Y \oplus Z$



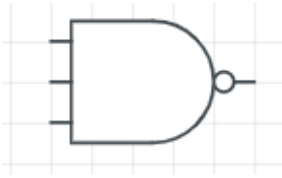
▲ $P = A \oplus B \oplus C \oplus D$

3. NAND 게이트와 NOR 게이트

- NAND 게이트

논리적 곱의 보수(AND-NOT) 수행하는 기능

NOT		\overline{X}
AND		$\overline{\overline{XY}} = XY$
OR		$\overline{\overline{XY}} = X + Y$

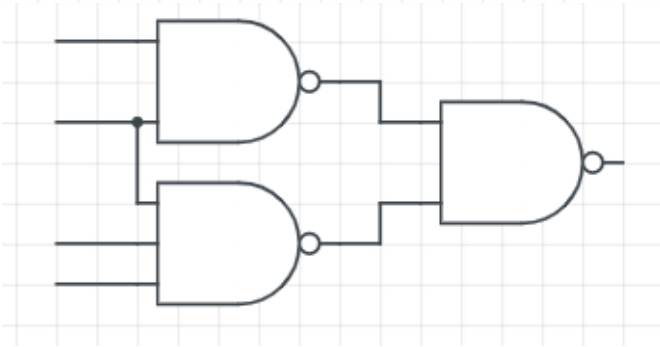
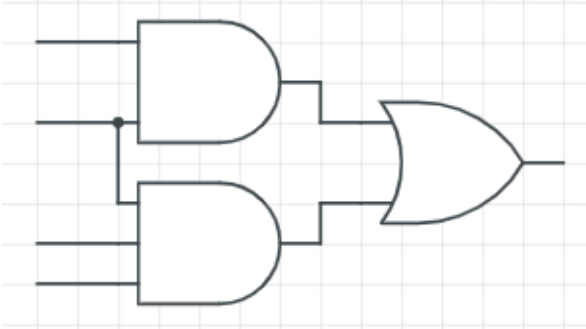


◀ AND-NOT \overline{XYZ}



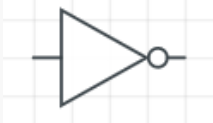
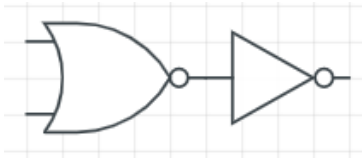
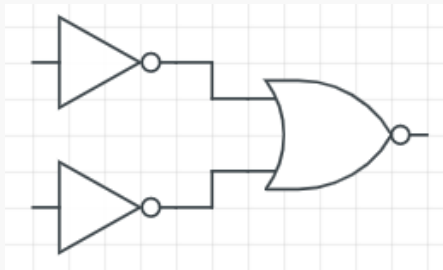
◀ NOT-OR $\overline{X} + \overline{Y} = \overline{XY}$

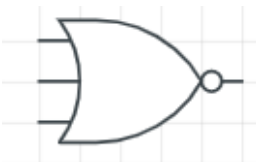
ex) $F = XYZ + WX$



- NOR 게이트

논리적 합의 보수(OR-NOT) 수행하는 기능

NOT		\overline{X}
OR		$\overline{\overline{XY}} = X + Y$
AND		$\overline{\overline{XY}} = XY$



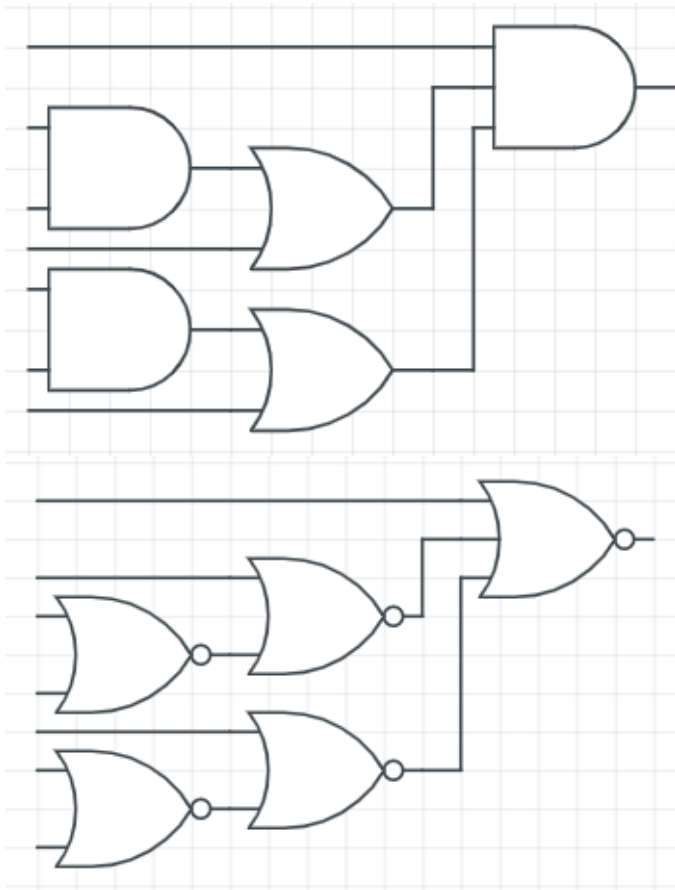
◀ OR-NOT $\overline{X + Y + Z}$

$$\overline{\overline{XY}} = \overline{X + Y}$$



◀ NOT-AND

ex) $F = W(XY + Z)(\overline{Y}Z + \overline{W})$



5 조합논리회로

1. 개요

- 디지털 시스템을 구성하는 논리회로

- 조합논리회로

현재의 입력에 대해 현재 입력의 논리조합에 의해서만 출력값이 결정되는 회로

부울대수의 집합에 의해 표현되는 논리연산을 수행하는 여러 논리게이트(AND, OR, NOT)로 구성

- 순서논리회로

- 조합 논리게이트에 플립플롭(flip-flop)이라는 저장요소 추가한 회로

- 저장요소의 상태와 입력변수에 의해 출력이 결정됨

- 조합논리회로의 구성

입력변수와 논리게이트, 출력변수로 구성

각 출력변수에 해당하는 m개의 부울함수로 나타낼 수 있음

각각의 출력 부울함수는 n개의 입력변수의 함수로 표현

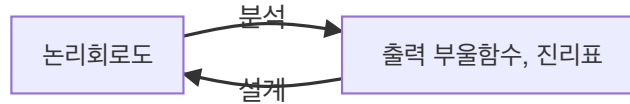
2. 조합논리회로의 분석과 설계

- 조합논리회로의 분석

논리도로부터 출력 부울함수나 진리표를 구하는 것

- 조합논리회로의 설계

주어진 회로에 대한 설명으로부터 논리도를 구하는 것



- 조합논리회로의 분석

- 부울함수의 유도

1. 모든 게이트의 출력에 임의의 기호를 부여, 부울함수 구함
2. 1에서 만든 출력을 입력으로 하는 게이트의 출력에 다시 임의의 기호를 붙여, 각 게이트에 대한 부울함수를 구함
3. 입력변수의 함수로 된 회로출력을 얻을 때까지 과정 2 반복

- 진리표 작성

1. 회로에서 입력변수의 개수를 정하고, n개의 입력에 대해 표에 0부터 $2^n - 1$ 까지 모두 2^n 개의 2진수 리스트 작성
2. 선택한 중간 출력게이트에 임의의 기호 붙임
3. 입력변수만으로 이루어진 출력게이트에 대해 진리표 작성
4. 모든 출력란을 작성할 때까지 중간 출력함수 정리

- 조합논리회로의 설계

1. 입력, 출력 변수를 적당한 기호로 표시, 블록도 그림
2. 입력, 출력 변수의 관계를 정의하는 진리표 작성
3. 각각의 출력을 입력변수의 함수로 나타내고 간소화
4. 논리회로도 그림

3. 기본 연산회로

- 가산기

- 반가산기 (HA)

한 비트의 2진수에 다른 한 비트 2진수를 더하는 산술회로

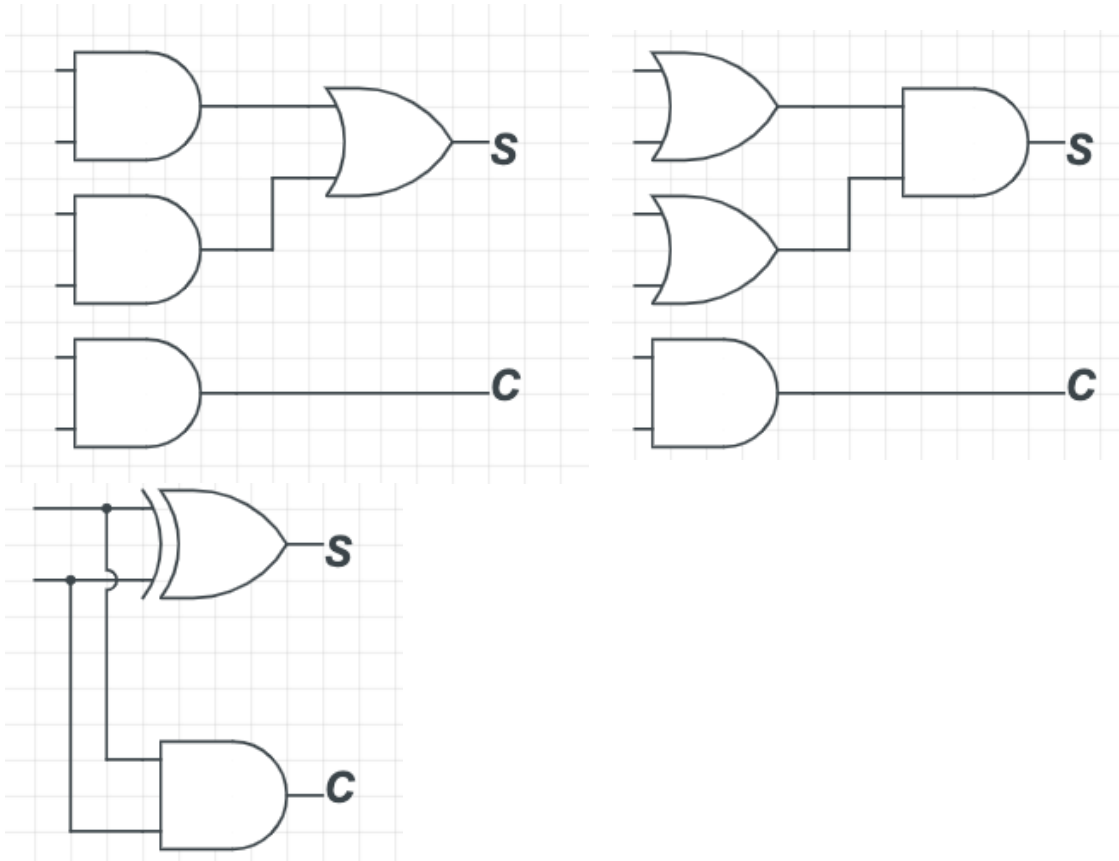
2개의 입력, 2개의 출력을 가짐

$$S = \overline{X}Y + X\overline{Y} = X \oplus Y$$

$$C = XY$$

- 반가산기의 진리표

입력- X	입력- Y	출력- S	출력- C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



$$\blacksquare S = X\bar{Y} + \bar{X}Y ; C = XY$$

$$S = X \oplus Y ; C = XY$$

$$\blacksquare S = (X + Y)(\bar{X} + \bar{Y}) ; C = XY \blacksquare$$

- 전가산기 (FA)

세 입력 비트의 합을 계산하는 조합논리회로

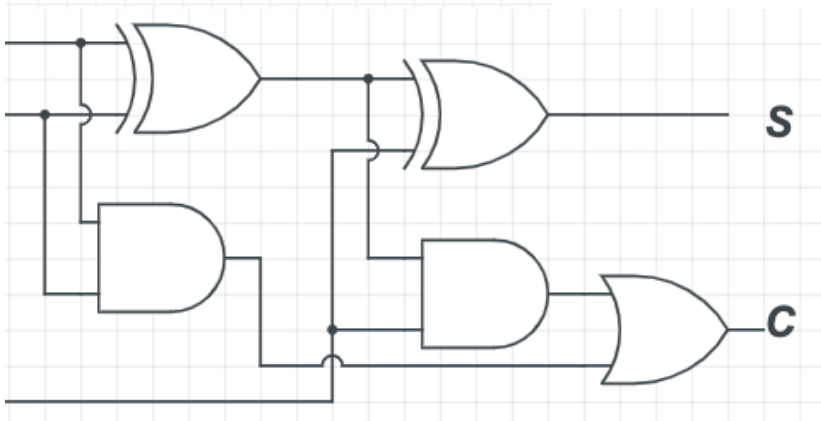
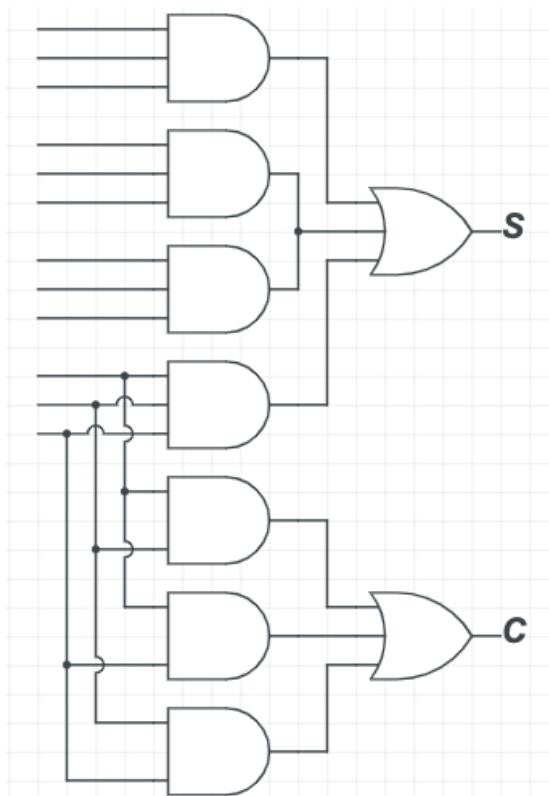
입력변수를 X, Y, Z 로 할때 X, Y 는 더해질 현재 위치의 자릿수, Z 는 올림수

$$S = \bar{X}\bar{Y}Z + \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + XYZ = X \oplus Y \oplus Z$$

$$C = XY + XZ + YZ = XY + Z(X \oplus Y)$$

- 전가산기의 진리표

입력- X	입력- Y	입력- $Z(= C_i)$	출력- S	출력- C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



▲ 전가산기의 논리회로도 1

▲ 전가산기의 논리회로도 2 (HA 2개 + OR)

- 직병렬 가산기

- 직렬방법

전가산기(FA) 1개와 출력 올림수를 유지하는 저장요소(F/F)로 구성

회로 구성비용 저렴

연산속도 느림

- 병렬방법

두 2진수의 산술합을 병렬로 수행하는 방법

2진 병렬가산기는 전가산기를 연속연결(cascade connection)하여 구성

- 감산기

- 반감산기 (HS)

뺄셈은 $0 - 0 = 0$, $1 - 0 = 1$, $1 - 1 = 0$, $0 - 1 = 1$ 의 네 가지로 수행

0-1일 때는 바로 앞자리에서 1을 빌려 와야 하고, 빌린 1은 2진수에서 2와 같으므로 결국 $2-1=1$ 이 됨

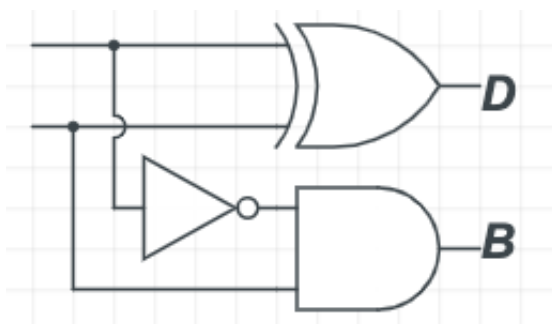
빌려 온 수를 **빌림수(borrow)**라고 함

$$D = \overline{X}Y + X\overline{Y} = X \oplus Y$$

$$B = \overline{X}Y$$

- 반감산기의 진리표

입력- X	입력- Y	출력- D	출력- B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0



반감산기의 논리회로도

- 전감산기 (FS)

바로 앞자리에서 빌려 온 1을 고려해 두 비트 사이의 뺄셈을 수행하는 조합 논리회로

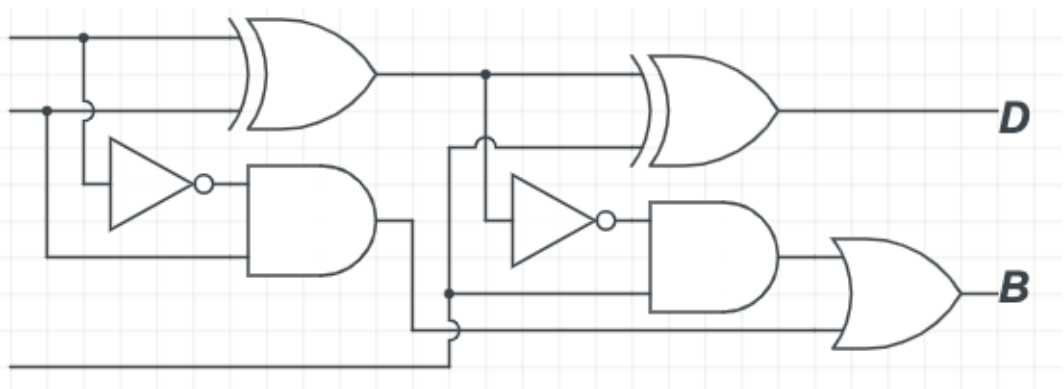
빌림수 입력을 취급하기 위해 입력회수 X, Y에 추가로 Z의 입력이 필요함

$$D = \overline{X} \overline{Y} Z + \overline{X} Y \overline{Z} + X \overline{Y} \overline{Z} = X \oplus Y \oplus Z$$

$$B = \overline{X} Y + \overline{X} Z + Y Z = \overline{X} Y + (\overline{X \oplus Y}) Z$$

- 전감산기의 진리표

입력-X	입력-Y	입력-Z	출력-D	출력-B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



◀ 전감산기의

논리회로도 (HS 2개 + OR)

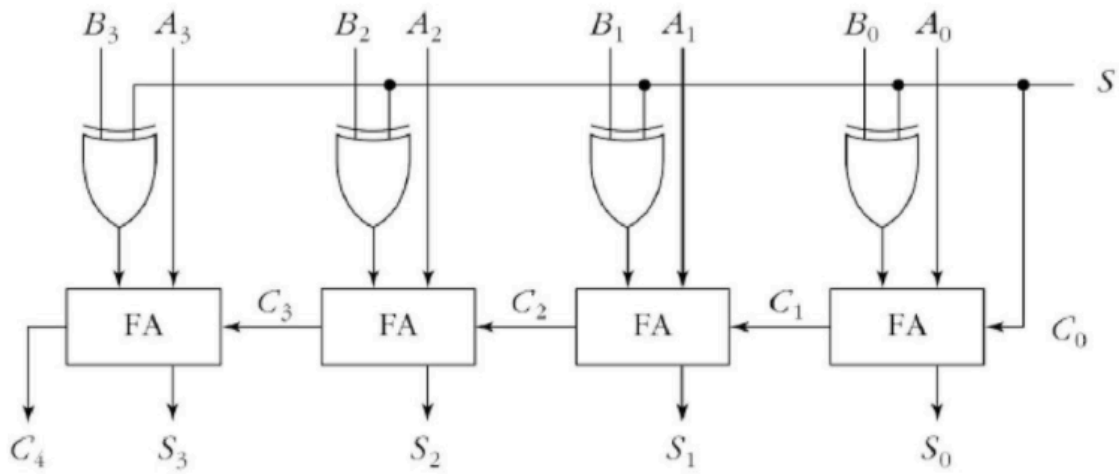
- 병렬 가,감산기

입력 S는 회로의 연산을 선택하는 제어단자, 즉 S가 0일 때 회로는 가산기 연산, S가 1일 때 회로는 감산기 연산 수행

XOR 게이트는 B와 S 중에서 하나를 선택하여 전가산기에 입력

S가 0이면, XOR는 B의 값을 그대로 통과시킴. C_0 의 값도 0이기 때문에 회로는 A에 B를 더하는 연산 수행

S가 1이면, XOR는 $B \oplus 1$ 로서 \overline{B} 가 되고 C_0 는 1이 되므로 회로는 A에 B의 2의 보수 값을 더하는 뺄셈 연산 수행



◀ 4비

트 가,감산기

- BCD 가산기

표현번호가 0부터 9까지

1010부터 1111까지는 사용하지 않으므로 9보다 큰 수가 나오면 보정작업 필요

- BCD 가산기의 진리표

K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	10진수
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

4. 여러가지 조합논리회로

- 코드변환기

- BCD-3초과 코드변환기

BCD코드를 3초과 코드로 바꾸는 변환기

- 카르노도표를 이용하여 구해진 곱의 합형태의 출력함수

$$W = A + BC + BD = A + B(CD)$$

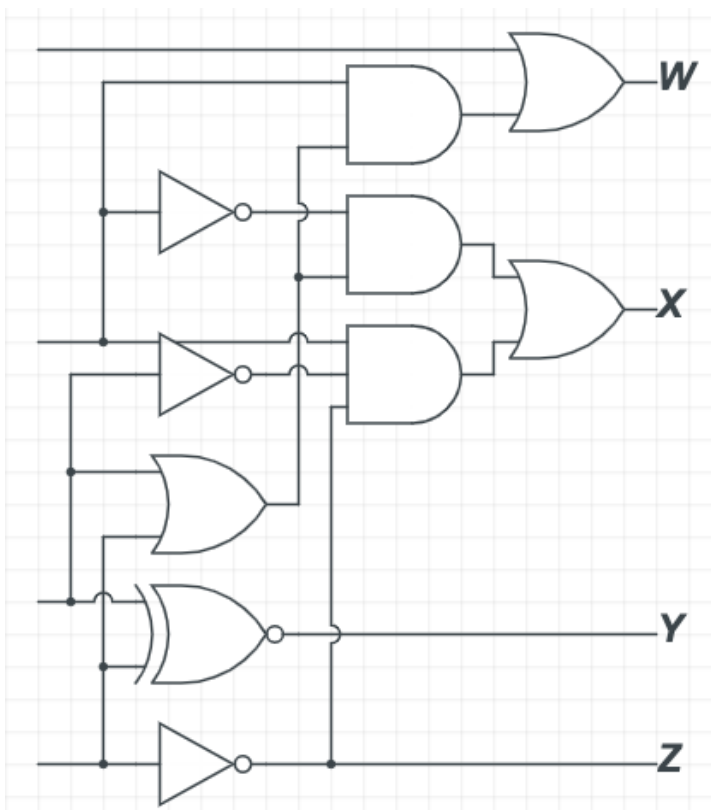
$$X = \overline{B}C + \overline{B}D + B\overline{C}\overline{D} = \overline{B}(C + D) + B\overline{C}\overline{D}$$

$$Y = CD + \overline{C}\overline{D} = \overline{C} \oplus \overline{D}$$

$$Z = \overline{D}$$

- 코드변환기의 진리표

10진 숫자	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0



◀ BCD-3초과 코드변환기의 논리회로도

- BCD-9의 보수변환기

- 카르노도표를 이용하여 구해진 곱의 합형태의 출력함수

$$W = \overline{A}B\overline{C}$$

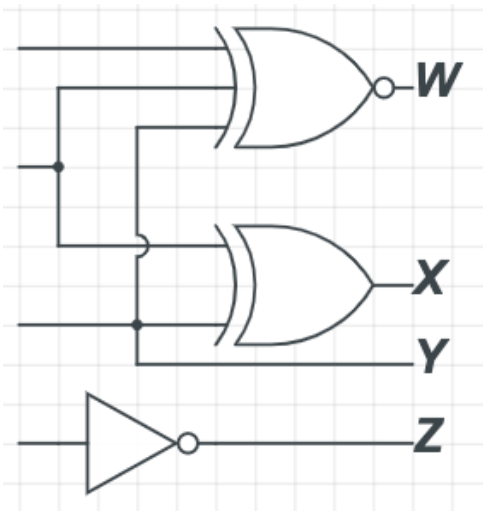
$$X = B\overline{C} + \overline{B}C = B \oplus C$$

$$Y = C$$

$$Z = \overline{D}$$

- BCD-9의 보수변환기 진리표

10진수	A	B	C	D	9의 보수	W	X	Y	Z
0	0	0	0	0	9	1	0	0	1
1	0	0	0	1	8	1	0	0	0
2	0	0	1	0	7	0	1	1	1
3	0	0	1	1	6	0	1	1	0
4	0	1	0	0	5	0	1	0	1
5	0	1	0	1	4	0	1	0	0
6	0	1	1	0	3	0	0	1	1
7	0	1	1	1	2	0	0	1	0
8	1	0	0	0	1	0	0	0	1
9	1	0	0	1	0	0	0	0	0



BCD-9의 보수변환기 논리회로도

- 패리티 발생기/검사기

에러를 검출해 내는 방법

전송된 2진 정보에 여분의 에러 검출용 비트를 1개 추가해 전송하는 방법

- 패리티 비트(parity bit)

에러 검출용 비트

- 짝수 패리티 검출(even parity check)

2진 정보 속에 있는 '1'의 개수가 패리티 비트를 합하여 짝수가 되도록 패리티 비트를 부가하는 방식

- 홀수 패리티 검출(odd parity check)

2진 정보 속의 '1'의 개수가 패리티 비트를 합하여 홀수가 되도록 패리티 비트를 부가하는 방식

- 패리티 발생기

패리티 비트를 생성해 내는 조합논리회로

- 패리티 검사기

에러를 검출하기 위해 패리티를 검사할 때 사용되는 조합논리 회로

- 패리티 비트가 부가된 BCD 코드

10진 수	BCD 코드	짝수 패리티의 BCD	1의 개수	BCD 코드	홀수 패리티의 BCD	1의 개수
0	0000	00000	0	0000	00001	1
1	0001	00011	2	0001	00010	1
2	0010	00101	2	0010	00100	1
3	0011	00110	2	0011	00111	3
4	0100	01001	2	0100	01000	1
5	0101	01010	2	0101	01011	3
6	0110	01100	2	0110	01101	3
7	0111	01111	4	0111	01110	3
8	1000	10001	2	1000	10000	1
9	1001	10010	2	1001	10011	3

- 패리티 발생기

홀수 패리티 발생기: 데이터 비트에 홀수 패리티 비트를 부가하는 방식

3비트 홀수 패리티 발생기의 설계

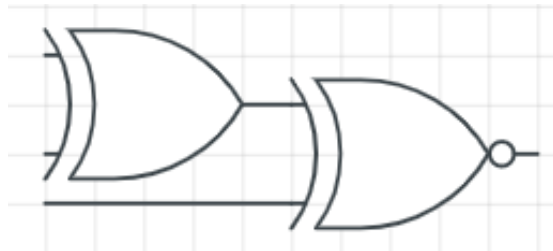
- 홀수 패리티 비트 발생기의 진리표

X	Y	Z	패리티 비트- P
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$$\begin{aligned}
 P &= \overline{X}\overline{Y}\overline{Z} + \overline{X}YZ + X\overline{Y}Z + XY\overline{Z} \\
 &= \overline{X}(\overline{Y}\overline{Z} + YZ) + X(Y\overline{Z} + \overline{Y}Z) \\
 &= \overline{X}(\overline{Y \oplus Z}) + X(Y \oplus Z) \\
 &= \overline{X \oplus Y \oplus Z}
 \end{aligned}$$

Y, Z

	00	01	11	10
X 0	1	0	1	0
1	0	1	0	1



▲ 홀수 패리티 발생기의 도표

▲ 3비트 홀수 패리티 발생기의 논리회로도

- 패리티 검사기

4비트 홀수 패리티 검사기의 설계

P가 '1'이면 에러 발생

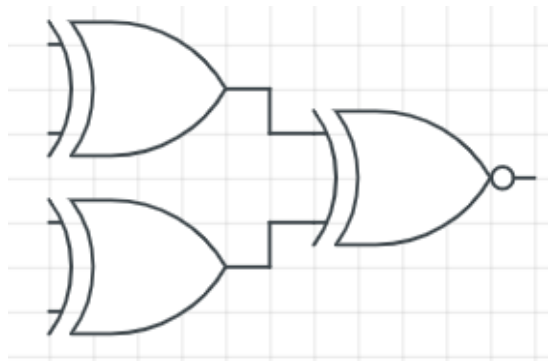
P가 '0'이면 에러 없음

- 홀수 패리티 검사기의 진리표

W	X	Y	Z	패리티 에러 검사- P
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

$$\begin{aligned}
 P &= \overline{W}\overline{X}\overline{Y}\overline{Z} + \overline{W}\overline{X}YZ + \overline{W}X\overline{Y}\overline{Z} + \overline{W}XY\overline{Z} + W\overline{X}\overline{Y}\overline{Z} + W\overline{X}YZ + WX\overline{Y}\overline{Z} + WXYZ \\
 &= \overline{W \oplus X \oplus Y \oplus Z}
 \end{aligned}$$

		Y,Z			
		00	01	11	10
W,X	00	1	0	1	0
	01	0	1	0	1
	11	1	0	1	0
	10	0	1	0	1



▲ 패리티 검사기의 카르노도표

▲ 4비트 홀수 패리티 검사기의 논리회로도

5. MSI를 이용한 조합논리회로

효과적인 조합논리회로 설계를 하려면 함수를 실현하는 데 필요한 게이트의 수를 최소화해야 함

집적회로(IC: Integrated Circuits)의 패키지화된 내부 게이트를 이용한다면 경제적인 설계가 될 수 있음

- 인코더/디코더

- 인코더(encoder)

부호화되지 않는 입력을 받아서 부호화된 출력으로 내보내는 부호화기

2^n 개의 입력과 n 개의 출력을 갖고 있으며, 출력은 입력값에 대응하는 2진 코드를 생성함

ex) 8진을 2진으로 바꾸는 인코더 내부회로의 설계

- 8진-2진 인코더 진리표

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

- 진리표에 의해 구해진 출력함수

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

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

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

- 디코더(decoder)

n 비트의 2진 코드를 최대 2^n 개의 서로 다른 정보로 바꿔 주는 조합논리회로

입력이 n 개 출력이 m 개인 디코더를 $n \times m$ 디코더라고 함

ex) 3×8

- 3×8 디코더 진리표

A_2	A_1	A_0	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	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

3×8 디코더는 3개의 입력선을 가지고, 출력은 2^n 개, 즉 8개의 출력으로 해독되며 각 출력은 3개 입력변수 중의 어느 하나를 나타냄

$$D_0 = \overline{A_2} \overline{A_1} \overline{A_0} \quad ; \quad D_1 = \overline{A_2} \overline{A_1} A_0$$

$$D_2 = \overline{A_2} A_1 \overline{A_0} \quad ; \quad D_3 = \overline{A_2} A_1 A_0$$

$$D_4 = A_2 \overline{A_1} \overline{A_0} \quad ; \quad D_5 = A_2 \overline{A_1} A_0$$

$$D_6 = A_2 A_1 \overline{A_0} \quad ; \quad D_7 = A_2 A_1 A_0$$

- 구동입력을 가진 디코더

회로의 동작을 제어하기 위해 구동입력(enable input)을 가짐

구동입력 E가 0일 때, 입력에 무관하게 모든 출력은 0

구동입력 E가 1일 때, 디코더로서 정상적 동작

- 디코더의 확장

크기가 큰 디코더가 필요하지만 작은 디코더만 사용 가능할 때는 작은 디코더를 여러 개 결합하여 필요한 크기의 디코더를 만들 수 있음

ex) 6×64 라인 디코더가 필요한 경우, 4개의 4×16 라인 디코더를 결합하여 필요한 디코더로 이용 가능

- 디코더를 이용한 부울함수 구현

디코더는 n개의 입력변수에 대해 2^n 개의 최소항을 만들어 냄

모든 부울함수는 최소항의 합으로 표현할 수 있으므로, 디코더를 이용해 부울함수를 구현할 수 있음

n개의 입력과 m개의 출력을 가진 조합논리회로를 $n \times 2^n$ 디코더와 m개의 OR 게이트로 만들 수 있음

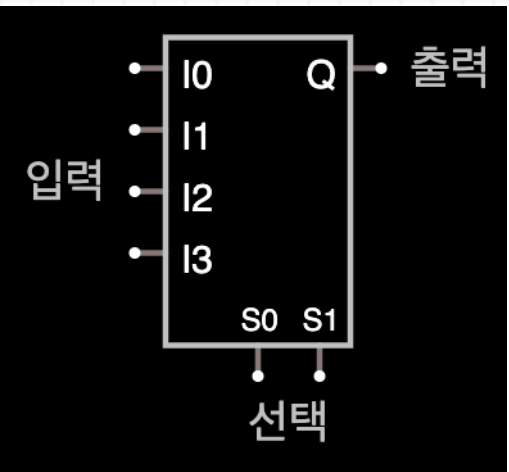
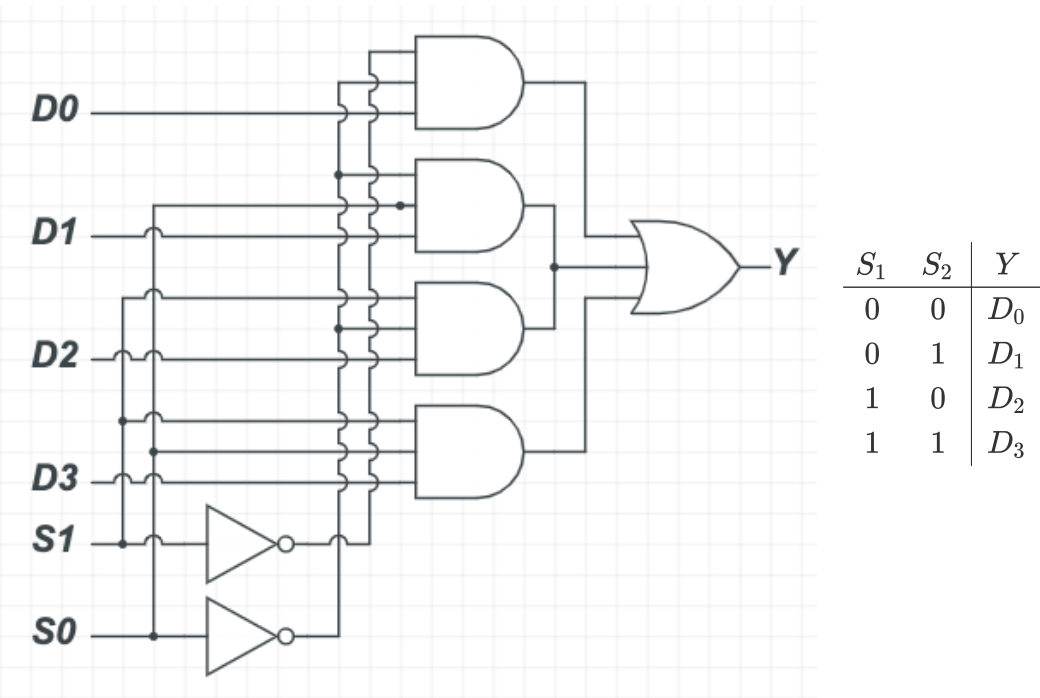
- 멀티플렉서/디멀티플렉서

- 멀티플렉서(multiplexer)

여러 개의 입력선 중에서 하나를 선택하여 단일의 출력으로 내보내는 조합논리회로

데이터 선택기(data selector)라고도 하고 약어는 MUX

ex) 입력이 4개이고 출력 1개인 4×1 멀티플렉서의 논리회로도, 진리표, 블록도



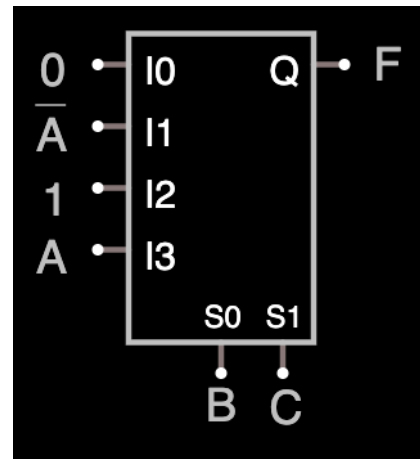
- 멀티플렉서를 이용한 부울함수 구현

ex) $F(A, B, C) = \sum m(1, 2, 6, 7)$ 멀티플렉서를 이용한 부울함수 구현

1. 2개의 선택선과 4개의 입력선을 가진 4×1 멀티플렉서가 필요
2. 주어진 부울함수에 대한 진리표 작성, 입력변수 중 가장 순서가 높은 A를 입력단으로 사용
3. 다음 순서인 B, C를 선택선 S_1 , S_0 에 각각 연결하고 입력단으로 사용될 변수 A의 논리를 알기 위해 구현표 작성
4. 구현표에 의해 멀티플렉서에 입력단 연결

최소항	A	B	C	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

	I_0	I_1	I_2	I_3
\overline{A}	0	(1)	(2)	3
A	4	5	(6)	(7)



진리표

구현표

멀티플렉서에 의한 구현

1. 열에 속하는 두 최소항이 둘 다 원으로 둘러싸여 있지 않다면 그 열에 대응되는 MUX 입력은 0
2. 두 최소항이 둘 다 원으로 둘러싸여 있다면 MUX 입력은 1
3. 아래쪽 최소항이 원으로 둘러싸여 있다면 MUX 입력은 A
4. 위쪽 최소항이 원으로 둘러싸여 있다면 MUX 입력은 \overline{A}

- 디멀티플렉서(demultiplexer)

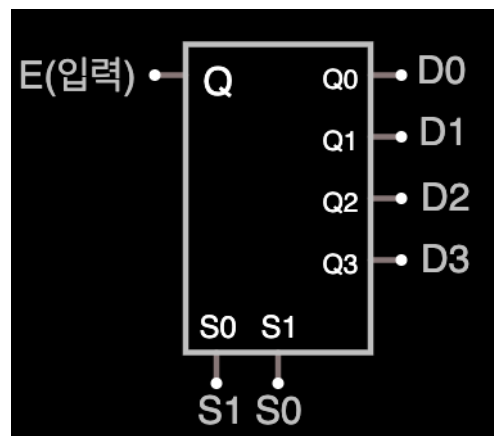
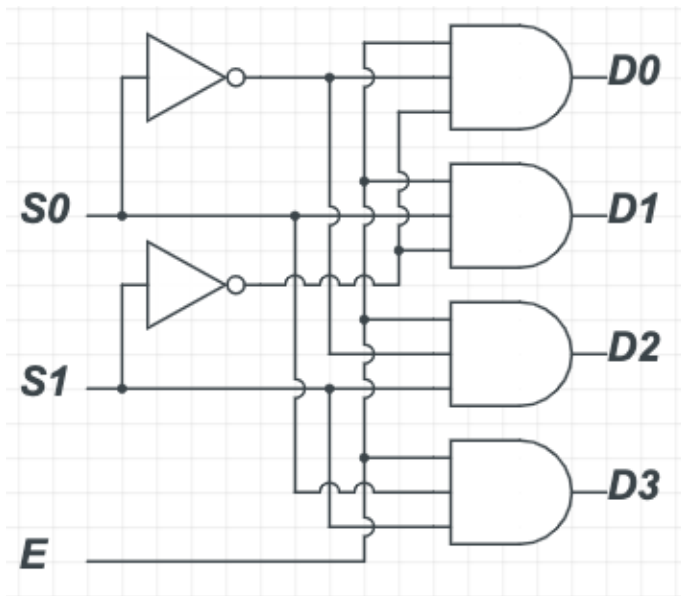
데이터 분배기(data distributor)라고도 불림, 약어는 DEMUX

멀티플렉서와 반대되는 연산을 수행하는 조합논리회로

1개의 입력선으로부터 정보를 받아 이를 2^n 개의 출력선 중의 하나로 내보냄

특정 출력선의 선택은 n개의 선택입력의 조합으로 제어

ex) 1×4 디멀티플렉서의 논리도와 블록도



6 순서논리회로

1. 개요

현재의 입력뿐만 아니라 과거의 입력과 시간에도 출력이 영향을 받게 됨

조합논리회로와 피드백(feed back)을 형성하는 저장요소로 구성

저장요소는 2진 정보를 저장할 수 있는 소자, 이 소자의 2진 정보는 주어진 시간에서의 순서 논리회로의 상태(state) 저장

- 동기 순서논리회로

회로의 상태가 정해진 순간의 입력에 의해서만 변화

- 비동기 순서논리회로

회로의 상태가 어느 순간에서나 입력의 변화에 따라 변화

2. 플립플롭

입력신호에 의해 상태를 바꾸도록 지시할 때까지는 현재의 2진 상태를 유지하는 논리소자

- 래치(latch)

플립플롭의 가장 기본이 되는 형태

클럭 신호에 관계없이 모든 입력을 계속 감시하다가 클럭과는 관계없이 언제든지 출력을 변화시키는 **비동기 순서논리소자**

- SR 래치

- NOR 게이트로 된 SR 래치

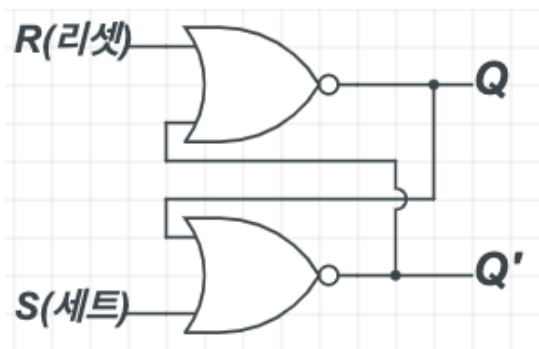
2개의 사용 가능 상태

Q가 1이고, \overline{Q} 가 0인 **세트 상태**

Q가 0이고, \overline{Q} 가 1인 **리셋 상태**

이때 출력 Q와는 서로 1의 보수관계

두 입력 S와 R이 모두 1일 때에는 출력값 Q, \overline{Q} 가 모두 0이 되는 **미정상태(undefined state)**를 가짐

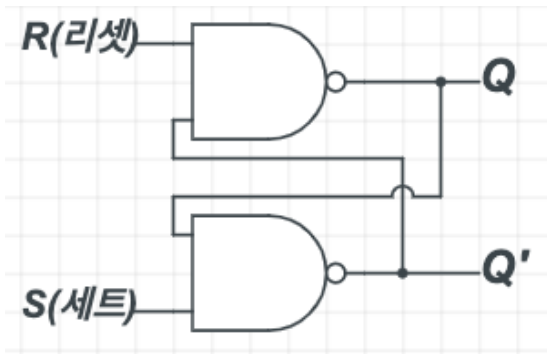


S	R	$Q(t+1)$
0	0	$Q(t)$ (무변화 상태)
0	1	0 (리셋 상태)
1	0	1 (세트 상태)
1	1	미정 상태

- NAND 게이트로 된 SR 래치

NOR 게이트로 된 SR 래치와 그 수행기능 동일

단지 입력에 따른 출력값을 나타내는 동작상태가 **정반대로 나타남**



S	R	$Q(t+1)$
0	0	미정상태
0	1	1(세트상태)
1	0	0(리셋상태)
1	1	$Q(t)$ (무변화상태)

- 제어입력을 가진 SR 래치(RS 플립플롭)

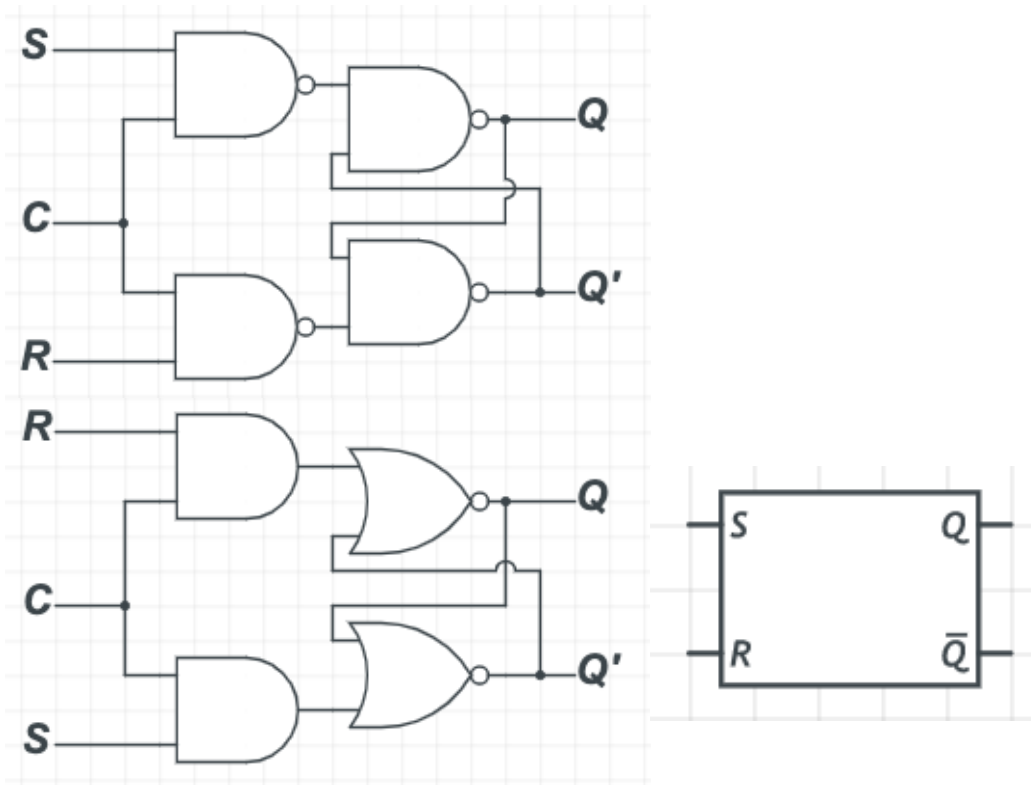
클럭을 가진 SR 래치, 또는 클럭을 가진 RS 플립플롭이라고 함

동기순서논리회로

기본적인 SR 래치에 2개의 NAND 게이트 추가

제어입력인 클럭 C가 1인 경우에는 보통의 래치 동작

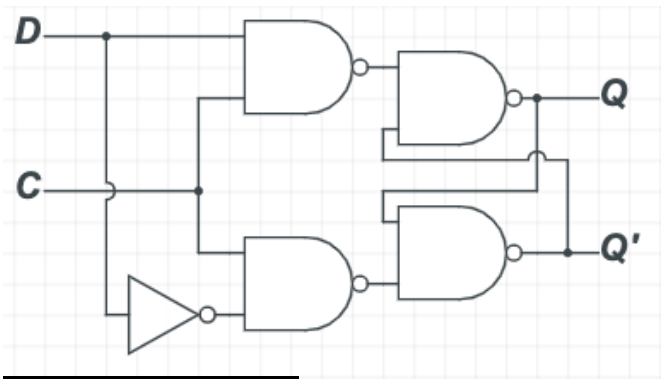
C가 0인 경우, 출력에는 아무런 영향 미치지 않음



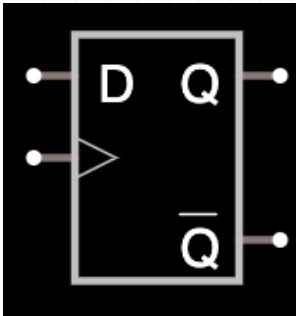
- D 플립플롭

RS 플립플롭에서의 문제점인 미정상태를 제거하는 방법

S와 R이 동시에 1을 갖지 않도록 함



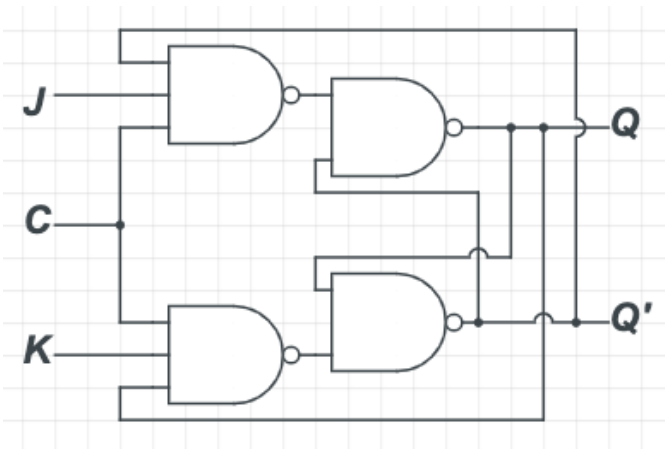
C	D	Q 의 다음 상태
0	\times	무변화 상태
1	0	0(리셋 상태)
1	1	1(세트 상태)



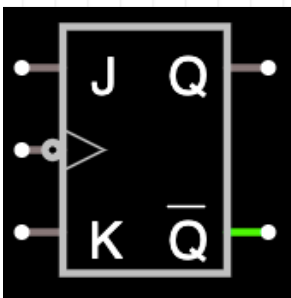
- JK 플립플롭

입력 J와 K는 플립플롭을 세트, 리셋시키며, J와 K가 둘 다 동시에 1이 가해지면 현재 상태의 보수를 취함

입력 J와 K가 모두 1일 때, 출력이 보수가 취해진 다음에도 클럭 펄스가 남아 있으면 또다시 보수를 취하는 반복적인 출력변화를 나타내는 레이스 현상 나타남



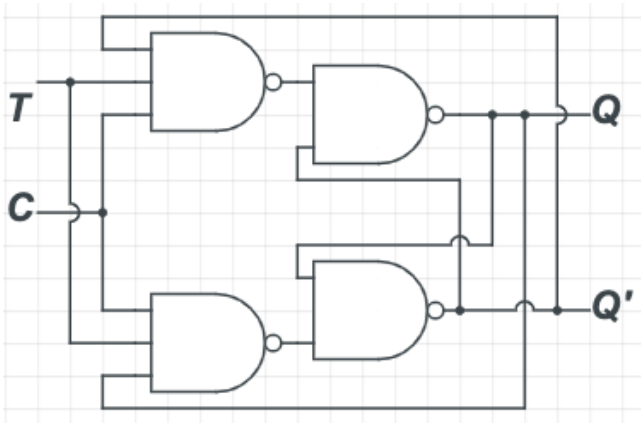
C	J	K	Q 의 다음 상태
0	\times	\times	무변화 상태
1	0	0	무변화 상태
1	0	1	리셋 상태
1	1	0	세트 상태
1	1	1	보수 (\bar{Q})



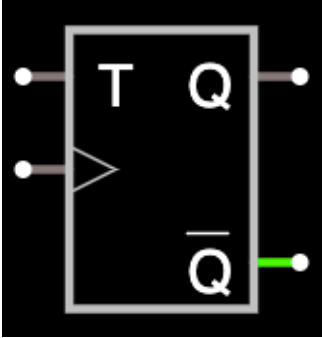
- T 플립플롭

JK 플립플롭의 변화된 형태

JK 플립플롭의 두 입력을 하나로 묶어서 만듦



T	Q 의 다음 상태
0	무 변화 상태
1	반전 (\overline{Q})



3. 플립플롭의 트리거링

- 트리거(trigger)

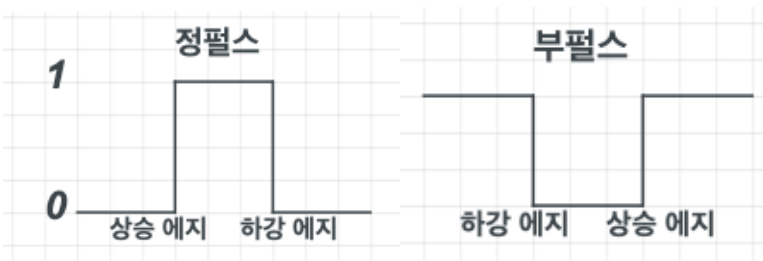
플립플롭의 상태는 제어입력시호의 순간적인 변화에 따라서 바뀌며, 이 순간적인 변화를 부르는 말 트리거에 의해 플립플롭의 상태를 변이시키는 것을 '플립플롭을 트리거한다'라고 함

- 레벨 트리거 방법

클럭 펄스가 논리-1인 동안 내내 입력이 출력에 영향을 미침

- 에지 트리거 방법

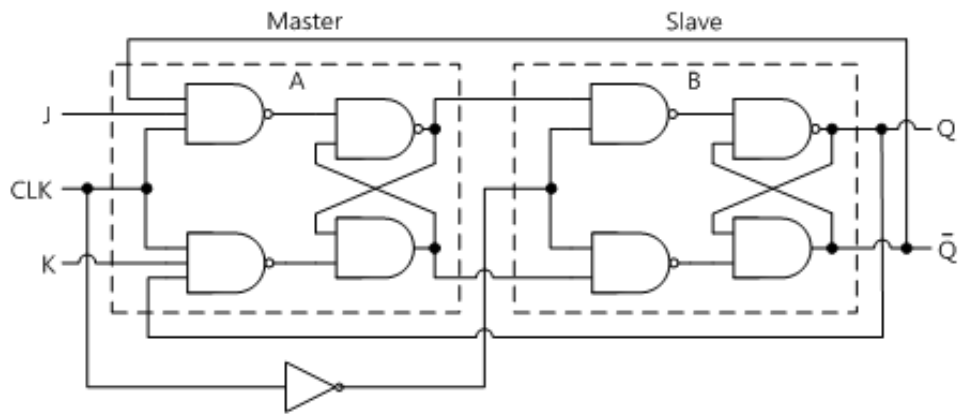
클럭 펄스의 에지, 즉 상승 에지나 하강 에지 동안에 입력이 출력에 영향을 미침



- 마스터-슬레이브 플립플롭

주(master)와 종(slave)의 역할을 하는 2개의 플립플롭으로 구성

클럭 펄스의 상승 에지에서 첫 번째 플립플롭을 세트, 클럭 펄스의 하강 에지에서 두 번째 플립플롭에 신호 전달하도록 구성



4. 순서논리회로의 분석

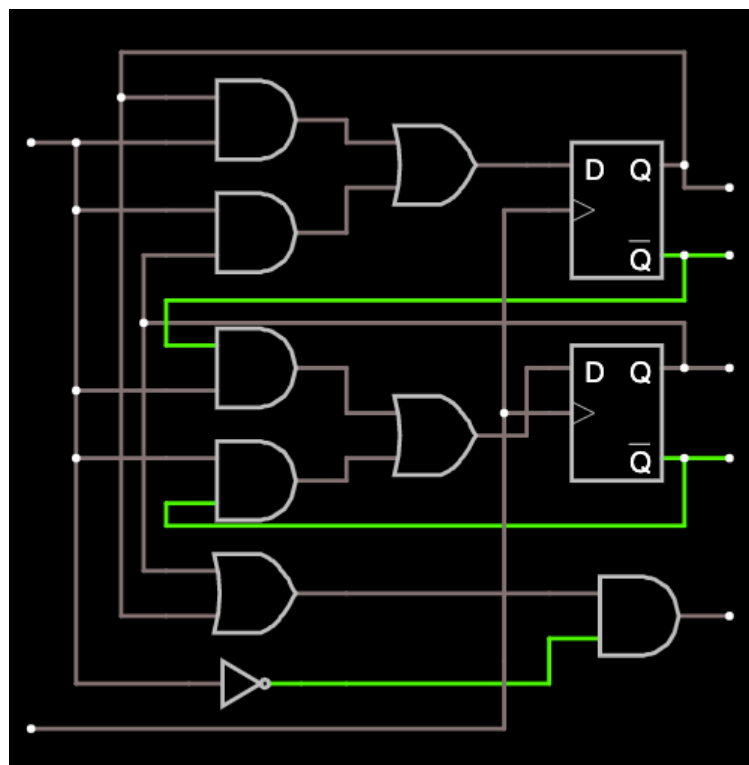
플립플롭의 입력방정식을 구한 다음 상태표를 작성하면 됨

상태표는 입력, 현재 상태, 다음 상태, 출력의 네 부분으로 구성

상태표를 완성하려면 다음 상태값이 결정되어야 함

다음 상태값을 알기 위해서는 해당 플립플롭에 들어오는 입력을 알면 구할 수 있으므로 플립플롭의 입력방정식으로 구하면 됨

ex) D 플립플롭을 가진 순서논리회로의 분석



- 입력방정식의 유도

1. 상태표 작성을 위한 다음 상태값을 알기 위해 순서논리회로에서 입력방정식을 구함
상태표를 구성하는 현재 상태와 입력란은 2진 조합에 의해 만들어짐
2. 플립플롭의 입력방정식은 조합논리회로의 출력 부울함수로 나타나며 입력은 X, 출력은 Y

$$D_A = AX + BX$$

$$D_B = \overline{A}X + \overline{B}X$$

$$Y = (A + B)\overline{X}$$

3. D_A , D_B 플립플롭의 다음 상태는 D_A , D_B 의 입력방정식에 의해 결정

$$A(t+1) = D_A = AX + BX$$

$$B(t+1) = D_B = \overline{a}X + \overline{B}X$$

- 상태표 작성

구해진 위의 입력방정식을 이용해 상태표의 다음 상태란 완성

나머지 출력란은 앞에서 구해진 다음의 출력방정식을 이용해 구함

$$Y = (A + B)\overline{X}$$

- 회로에 대한 상태표

현재 상태-A	현재 상태-B	입력-X	다음 상태-A	다음 상태-B	출력-Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	0	0

- 특성표

플립플롭의 논리적 성질 정의, 그 동작 특성을 표로 나타냄

S	R	$Q(t+1)$
0	0	$Q(t)$
0	1	0
1	0	1
1	1	미정


J	K	$Q(t+1)$
0	0	$Q(t)$
0	1	0
1	0	1
1	1	$\overline{Q}(t)$

D	$Q(t+1)$
0	0
1	1

T	$Q(t+1)$
0	$Q(t)$
1	$\overline{Q}(t)$

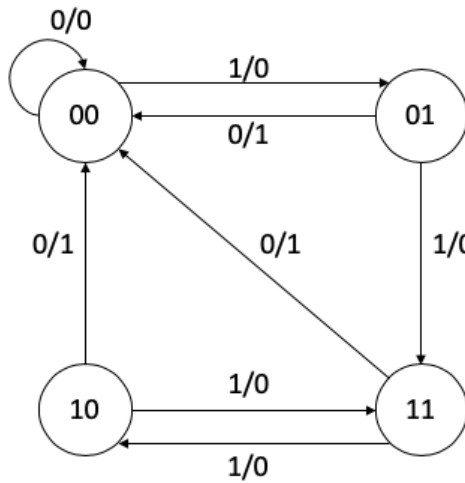
 RS 플립플롭

 JK 플립플롭

 D 플립플롭

 T 플립플롭

- 상태도 (state diagram)



상태표로 주어진 정보를 도식적으로 나타내는 것

상태는 하나의 원으로 표시, 그 상태 간의 변화는 원을 연결하는 선으로 표시

원을 연결하는 화살표는 현재상태에서 다음 상태로의 변화를 표현함

원 안의 2진수는 플립플롭의 상태

원들을 연결하는 선에 표현된 2진수는 입력과 출력을 나타냄

5. 순서논리회로의 설계

1. 주어진 문제 설명이나 상태도로부터 상태표 작성
2. 플립플롭의 종류와 개수 결정, 각 플립플롭에 기호 할당
3. 상태표의 다음 상태에서부터 플립플롭의 입력방정식을 구함
4. 상태표에 출력이 있으며 출력방정식을 구함
5. 구해진 입력방정식과 출력방정식을 간소화
6. 간소화된 입출력방정식을 이용하여 논리도 작성

- 플립플롭의 결정

일반적으로 데이터 전송을 위한 설계에는 D 플립플롭을, 보수를 포함한 응용에는 T 플립플롭이 사용되지만 일반적으로 JK 플립플롭을 많이 사용

- 입력방정식의 유도

- 입력방정식

플립플롭의 입장에서 보면 조합논리회로의 출력값은 입력방정식이 되고 외부입력과 플립플롭의 현재 상태에 의해 결정됨

입력방정식을 구하려면 현재 상태에서 다음 상태로의 변화를 야기하는 입력조건을 구하면 됨

• 여기표

현재 상태에서 다음 상태로의 변화를 일으키는 입력조건의 리스트


- 플립플롭의 여기표

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

 RS 플립플롭

 JK 플립플롭

 D 플립플롭

 T 플립플롭

JK 플립플롭의 경우 무관조건의 표현이 가장 많이 나타남

따라서 회로설계 시 입력함수를 카르노도표를 이용하여 간소화할 때 더욱 간단한 형태로 조합회로를 만들 수 있음

순서회로의 설계 시 JK 플립플롭을 사용하는 것이 효과적임

7 레지스터와 카운터

1. 레지스터

- 개요

- 레지스터

데이터를 일시 저장하거나 전송하는 장치

- 레지스터의 구성

여러 개의 플립플롭을 연결하여 구성

1개의 플립플롭은 1비트의 2진 정보를 저장하므로 n비트 레지스터는 n개의 플립플롭으로 구성되며 n비트의 2진 정보를 저장

- 레지스터의 기능

여러 비트를 일시적으로 저장하거나 배열된 비트를 좌우로 자리이동(shift)을 시키는 데 사용

- 레지스터의 종류

- 데이터의 입출력방식에 따라

직렬입력-직렬출력 레지스터

직렬입력-병렬출력 레지스터

병렬입력-직렬출력 레지스터

병렬입력-병렬출력 레지스터

- 데이터의 자리이동 방식에 따라

좌측 시프트(left shift) 레지스터

우측 시프트(right shift) 레지스터

양방향 시프트(bidirectional shift) 레지스터

- 데이터 적재 레지스터

- **적재(load)**

레지스터에 새로운 데이터 기억시키는 과정

- **데이터 적재 레지스터**

입력된 데이터를 그대로 기억하는 역할을 수행하는 레지스터

일반적으로 D 플립플롭을 사용하여 구성

- 직렬적재 레지스터

데이터를 순차적으로 받아들이는 방식

1비트씩 입력

- 병렬적재 레지스터

4개의 플립플롭의 입력단에 연결된 4개의 비트를 하나의 공통 연결된 클럭 펄스에 의해 동시에 레지스터로 적재

- 시프트 레지스터 (shift register)

레지스터가 기억하고 있는 정보를 한 방향, 혹은 양방향으로 이동시킬 수 있는 레지스터

플립플롭을 직렬로 연결한 것

한 플립플롭의 출력을 다음 플립플롭의 입력에 연결한 것

레지스터 A에서 레지스터 B로의 정보 직렬전송도 가능

- 병렬적재 양방향 시프트 레지스터

왼쪽과 오른쪽의 양쪽 방향으로의 시프트와 병렬적재가 가능한 레지스터

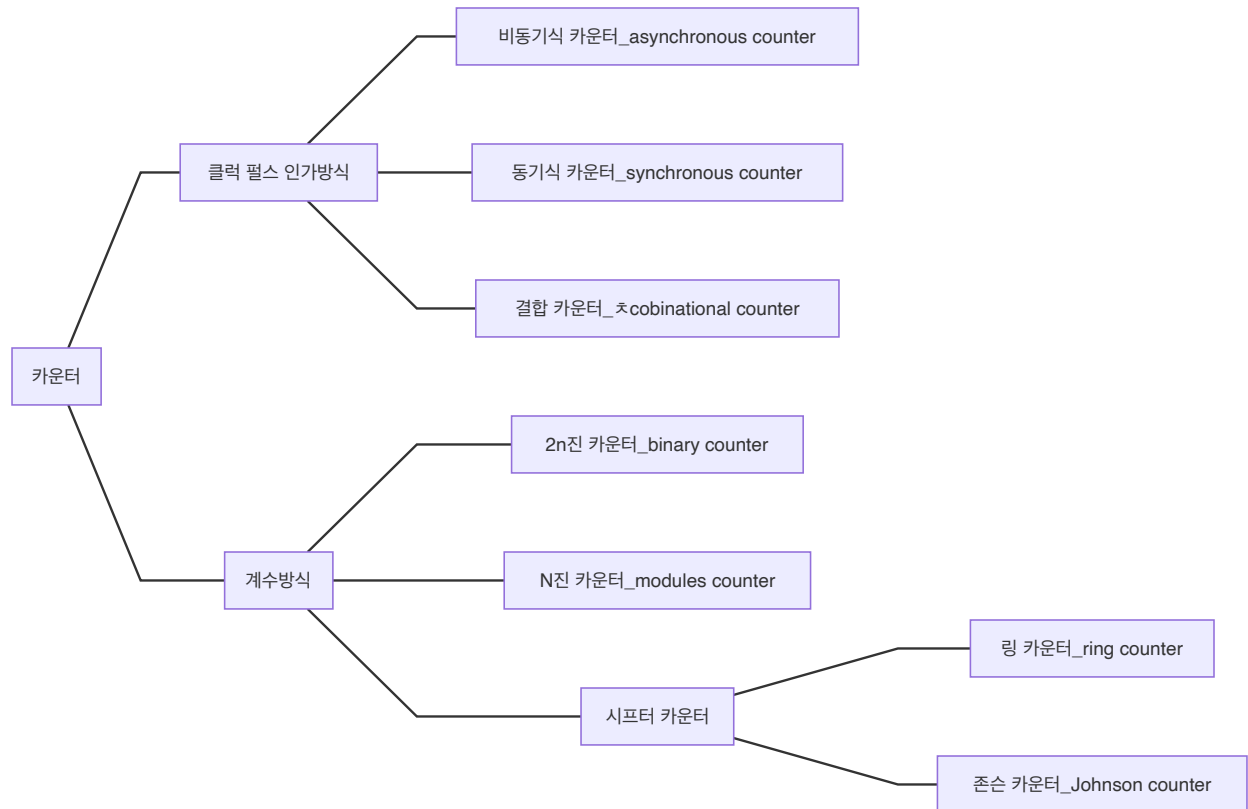
D 플립플롭과 멀티플렉서로 구성

2. 카운터

- 개요

입력 펄스의 적용에 따라 미리 정해진 순서를 밟아 가는 순서논리회로

- 카운터의 분류



- 비동기식 카운터

리플 카운터(ripple counter)라고도 함

카운터를 구성하고 있는 각 플립플롭에 동시에 클럭이 가해지지 않는 카운터로 입력 클럭 펄스가 앞 단의 출력값에 의해서 영향을 받음

- 2진 리플 카운터

비동기식 2진 카운터

카운터를 구성하는 플립플롭이 동시에 상태가 변화하지 않음

클럭 입력이 플립플롭의 첫 번째 단, 즉 가장 낮은 자리의 비트를 저장하는 플립플롭에만 연결

두 번째 플립플롭부터는 앞의 플립플롭 출력에 의해 트리거됨

4비트 2진 리플 카운터의 계수순서를 보면 0에서 시작하여 계수 클럭 펄스마다 1씩 증가하고 15를 계수한 다음 0으로 되어 다시 같은 순서 반복

- 업 카운터(up counter)

계수를 상향계수하는 카운터

- 다운 카운터(down counter)

하향계수하는 카운터

- BCD 리플 카운터

0부터 9까지 10개의 상태를 계수하는 카운터

각 상태는 10진수를 나타내는 2진 코드로 표현하므로 적어도 4비트 필요

- 동기식 카운터

카운터를 구성하고 있는 모든 플립플롭에 동시에 클럭 펄스가 가해지고, 모든 플립플롭이 한꺼번에 동작함
동작속도의 향상

- 2진 카운터

동기식 2진 카운터

가장 낮은 위치의 플립플롭은 클럭 펄스가 입력될 때마다 보수를 취함

=> 계수 구동입력 E가 1값을 가짐

다음 플립플롭은 앞쪽의 낮은 위치의 모든 플립플롭이 1을 취할 때 보수를 취함

- 모듈로 - N 카운터

원하는 수만큼의 상태순서만을 계수하도록 설계된 카운터

N개의 계수(0, 1, 2, ..., N-1)의 순서를 반복하여 계수

간단히 **모드-N 카운터(modulo-N counter)**라고 함

- 시프트 카운터

- 링 카운터

시프트 레지스터를 응용한 가장 간단한 카운터

출력 비트 중 1비트만 1이 되고 입력 펄스에 의해 한쪽 방향으로 1의 위치가 순환됨

시프트 레지스터의 최종 출력을 다시 입력에 피드백시키는 구조를 갖는 일종의 **순환 시프트 레지스터**

- 존슨 카운터

시프트 레지스터에 피드백을 이용한 구조

일명 **트위스트 링 카운터(twist ring counter)**

시프트 레지스터의 최종단 플립플롭의 출력 Q가 아닌 \overline{Q} 에서 입력 쪽으로 피드백시키는 것이 링 카운터와 다른 점

- 카운터의 설계

논서논리회로의 설계과정을 그대로 따르면 됨

먼저 상태도나 주어진 명세에 의해 상태표를 작성하면 됨

카운터의 상태도는 클럭 펄스가 유일한 입력으로 동작, 출력이 플립플롭의 현재 상태에 의해 표현

=> 순서논리회로의 상태도와는 달리 입력과 출력값이 명시되지 않음

3. 메모리셀

8 기억장치와 PLD

1. 개요

- 기억장치의 특성

기억장치는 많은 양의 2진 정보를 저장할 수 있는 능력을 가진 **기억소자의 모임**

기억장치에서는 데이터나 프로그램을 일렬로 기억할 수 있도록 일정한 번지인 주소(address)가 부여

주소는 바이트(byte)나 워드(word) 단위로 됨

주로 기억용량을 나타내는 단위로 바이트를 사용함

- **기억장치의 성능**

기억장치에 저장된 정보를 읽거나 기록하는 속도에 따라 좌우됨

일정한 양의 정보를 읽어 내는 데 걸리는 **평균시간**을 측정하여 평가

- **액세스 타임(access time)**

한 워드의 정보를 읽어 내는 데 걸리는 시간

- 기억장치의 종류

1. 반도체 메모리를 사용한 기억장치

2. **RAM(Random Access Memory)**

임의접근이 가능한 기억장치

3. **ROM(Random Only Memory)**

읽기만을 허용하는 기억장치

4. **PLD(Programmable Logic Device)**

읽기만을 허용하는 기억장치로 프로그램이 가능한 논리장치

5. **PLA(Programmable Logic Array)**

프로그램이 가능한 논리배열

6. **PAL(Programmable Array Logic)**

프로그램이 가능한 배열논리

2. RAM

- 종류

정적인 **SRAM(Static RAM)**과 동적인 **DRAM(Dynamic RAM)**으로 구분

- **SRAM**

플립플롭으로 구성되어 액세스 타임이 고속

주로 캐시 메모리(cache memory)로 사용

- **DRAM**

커패시터에 충전되는 전하의 형태로 정보를 저장하는 RAM

충전된 전하는 시간에 따라 방전하려는 경향이 있음

=> RAM 내용을 소멸시킬 수 있음

=> 컴퓨터의 주기억장치에 주로 사용됨

- 구성

기본단위인 기억소자는 1비트를 저장할 수 있는 플립플롭 회로로 되어 있음

RAM의 내부구성은 크게 **기억부**, **해독부**, **제어부**의 세 부분으로 구분

- **기억부**

여러 개의 기억소자로 이루어진 부분

2차원의 테이블 혹은 행렬형태로 이루어져 있음

디코더에 의해 기억소자가 선택

- **해독부**

디코더로 구성

2개의 주소입력은 2×4 디코더에 의해 4개의 단어 중 하나를 선택하게 됨

디코더는 선택입력에 의해 구동

선택입력이 0, 모든 디코더의 출력은 0이 되어 단어를 선택하지 않음

선택입력이 1, 주소선의 입력값에 따라 4개의 단어 중 해당 단어 선택

- **제어부**

읽기/쓰기 단자의 입력신호에 따라 데이터 입력과 출력 제어

해독부에 의해 선택된 기억소자는 제어부의 읽기/쓰기 신호에 따라 정보를 읽을 것인지, 써넣을 것인지 결정

- 확장

3. ROM

- 종류

1. **마스크 ROM**

사용자의 요구에 따라 제조 시에 데이터를 기록해, 한번 저장한 것은 그 내용을 바꿀 수 없음

=> 융통성 없음, 대량생산 적합

2. **PROM(Programmable ROM)**

사용자가 데이터를 기록하는 것이 가능한 ROM

PROM writer를 이용해 프로그램 기록

3. **EPROM(Erasable PROM)**

한 번 쓴 내용을 지울 수 있는 PROM

4. EEPROM(Electrically Erasable PROM)

기억시킨 내용을 전기적으로 기울 수 있는 PROM

- 구조

n개의 입력선과 n개의 출력선으로 구성

입력선은 기억장치의 주소(address)

출력선은 주소에 의해 선택된 단어의 데이터를 출력

- ROM을 이용한 조합논리회로 구현

ROM은 디코더와 OR 게이트로 이루어진 논리회로

디코더는 n개의 입력변수를 가지고 2^n 개의 최소항을 만들어 내는 조합논리회로

디코더에 OR 게이트를 연결하면 부울함수의 **최소항의 합**을 만들어 낼 수 있으므로, ROM을 이용해 조합논리회로 구현 가능

- ROM을 하드웨어적으로 프로그램하는 방법

주어진 진리표에 따라 전자 퓨즈선을 유지/절단시키는 것

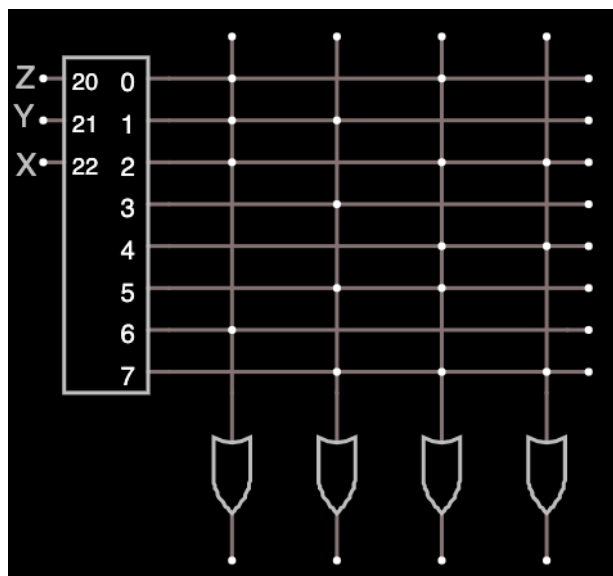
전자 퓨즈선의 유지는 진리표상의 1을 의미

절단은 0을 의미

ex)

$$F_0(X, Y, Z) = \sum m(0, 1, 3, 5, 6) \quad F_1(X, Y, Z) = \sum m(1, 3, 6)$$

$$F_2(X, Y, Z) = \sum m(0, 2, 4, 6) \quad F_3(X, Y, Z) = \sum m(3, 4, 5, 7)$$



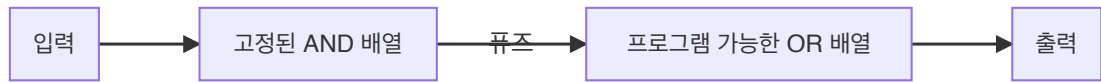
4. PLD

프로그램이 가능한 논리장치

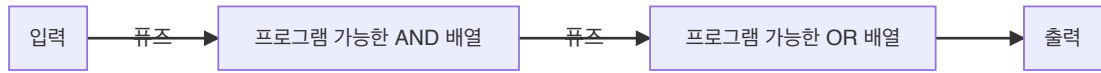
프로그램이 가능한 전자 퓨즈선으로 연결된 게이트의 배열로 구성된 집적회로

- PLD 형태

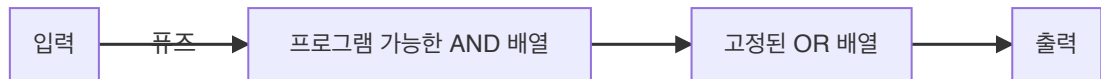
1. PROM



2. PLA



3. PAL



- PLA

ROM의 단점인 기억공간의 낭비 보완

모든 입력변수를 디코딩하지 않으며, 모든 최소항도 만들지 않음

크기는 입력의 수, 곱항의 수, 출력의 수로 결정

- PAL

OR 게이트 배열은 고정되어 있고, AND 게이트 배열은 프로그래밍이 가능한 소자

AND 게이트 배열만 프로그램이 가능하므로 PLA보다는 프로그래밍상 제한이 따름

일반적인 PLA보다는 값이 싼

간단한 논리함수의 실현에 효과적

가장 보편적으로 사용