1차 실험 보고서

실습명: 조합/순서 논리 회로 시뮬레이션과 분석

과목명 컴퓨터 구조 2반

실습 날짜: (조합논리회로)2021.9.21

(순서논리회로)2021.9.27

보고서 제출일:2021.10.13

학과/학번 :소프트웨어융합대학 콘텐츠it학과, 20185309

성명 : 황명원

I. 조합 논리 회로 (8비트 병렬 가감산기)

1. 실습 목적

8비트 병렬 가감산기를 모의실험으로 구현하여 동작을 분석하여, 조합 논리 회로를 이해한다. 동시에 직접 회로를 짜보고 시뮬레이션을 돌려보며 시행착오를 직접 겪어보며 혼자 공부하여 완성된 회로를 이용해 본인의 회로의 대한 이해와 역량을 기른다.

1. 기본 이론

조합논리회로는 논리곱(AND), 논리합(OR), 논리 부정(NOT)의 세 가지 기 본 논리 회로를 조합하여 구성한 논리 회로이다. 조합논리회로는 입력변수, 논리 게이트, 그리고 출력변수들로 구성된다.

이론적이고 이해해야할 부분을 살펴보면

1. 반 가산기(Half Adder)

ㅡ반 가산기는 2개의 2진수 A,B 논리변수를 더하여 합(Sum)과 캐리(Carry)를 산출하기 위한 조합 논리 회로이다.

반 가산기의 진리표와 논리식을 본다면 다음과 같다.

테이블이(가) 표시된 사진

자동 생성된 설명

1. 전가산기(full-adder, FA)

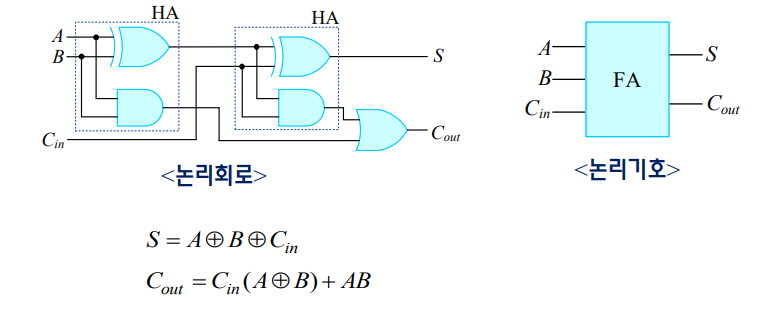
ㅡ 컴퓨터 내부에서 여러 비트로 된 두 수를 더할 때에는 두 비트에서 더해진 결과인 캐리는 더 높은 자리의 두 비트의 덧셈에 추가되어 더해진다.

이때 아래 자릿수에서 발생한 캐리까지 포함하여 세 비트를 더한다. 즉 전가산기는 자리 올림수(carry)를 고려하여 만든 덧셈 회로이다.

전가산기의 진리표와 논리식을 보면 다음과 같다.

텍스트이(가) 표시된 사진

자동 생성된 설명



전 가산기는 이처럼 반가산기 2개와 OR게이트를 이용하여 구성된다.

1. 병렬가감산기(parallel-adder/subtracter)

-병렬가산기 : 전가산기 여러 개를 병렬로 연결한 회로를 말한다.

텍스트, 시계, 게이지이(가) 표시된 사진

자동 생성된 설명

-병렬감가산기: 병렬가산기의 B입력을 부호 S(sign)와 XOR하여 전가산기의 입 력으로 사용함으로써 덧셈과 뺄셈이 모두 가능한 회로

텍스트, 하늘이(가) 표시된 사진

자동 생성된 설명

맨 아래 입력단의 캐리 입력을 부호(덧셈 이나 뺄셈)로 하고, 각 자리의 B입력과 부호를 XOR 게이트의 입력으로 하고,XOR 게이트의 출력을 전가산기의 두번째 입력으로 한다.

XOR 게이트의 동작은 부호(sign)는 0(덧셈)이면 원래의 B값은 그대로 출력되고, 부호가 1(뺄셈)이면 B값과 반대되는 값이 출력된다.

이 출력값이 전가산기의 입력으로 들어간다.

그러므로 뺄셈이면 XOR 게이트의 출력은 B값의 1의 보수가 된다.

그리고 맨 아랫단의 캐리가 뺄셈일 때 1이므로 1이 더해지는 결과가 된다.

그러므로 전체적으로 보았을 때 뺄셈은 A의 값과 B의 보수가 더해지고,

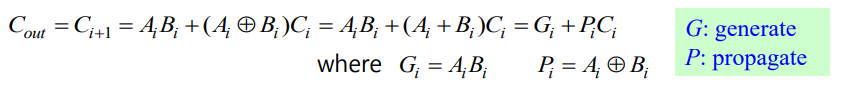
맨 아랫단의 1이 더해지므로 2의 보수 뺄셈이 된다.

1. 고속가산기

아랫단에서 윗단으로 전달되는 자리올림수 때문에 병렬가산기는 속도가 매우 느리다는 단점이 있다.

단점을 해결하기 위해 캐리예측가산기(carry-look-ahead-adder, CLA)를 사용한다.

CLA는 원리는 첫 번째는 Ai , Bi 모두가 1일 때, 또는 Ai , Bi 둘 중에 하나가 1이고 Ci 가 1일 때 캐리가 발생하므로 논리식은 다음과 같다.

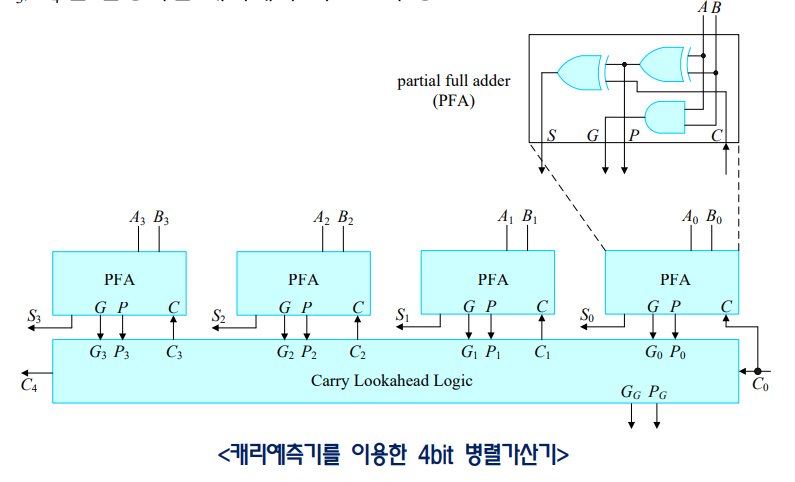


4비트 가산기에서 위 식을 써보면 다음과 같다.

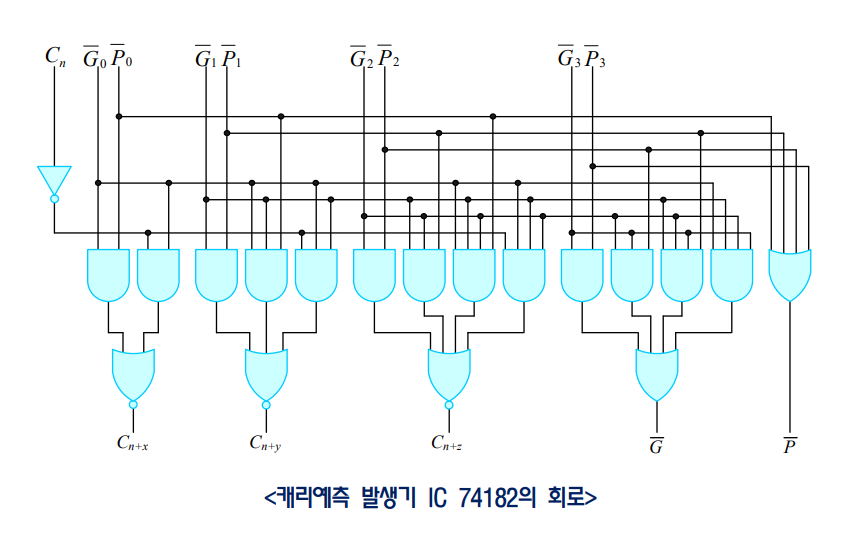
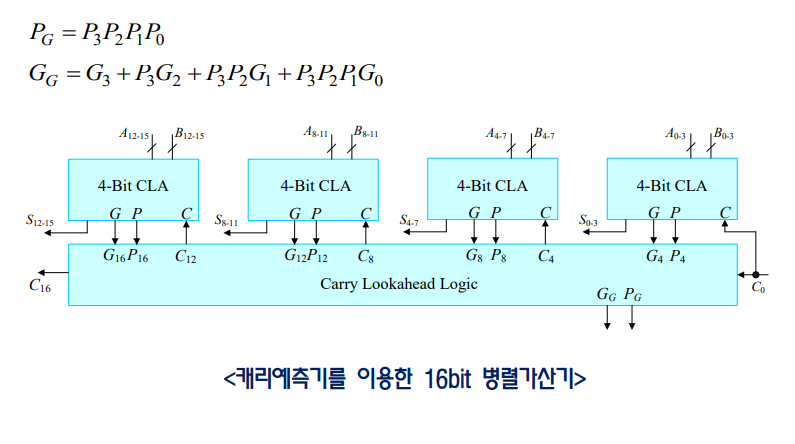
텍스트이(가) 표시된 사진

자동 생성된 설명

캐리예측가산기는 Si , Pi , Gi를 발생시키는 부분전가산기(PFA)와 위의 식 C1 , C2 , C3 , C4 을 발생하는 캐리예측 회로로 구성한다.

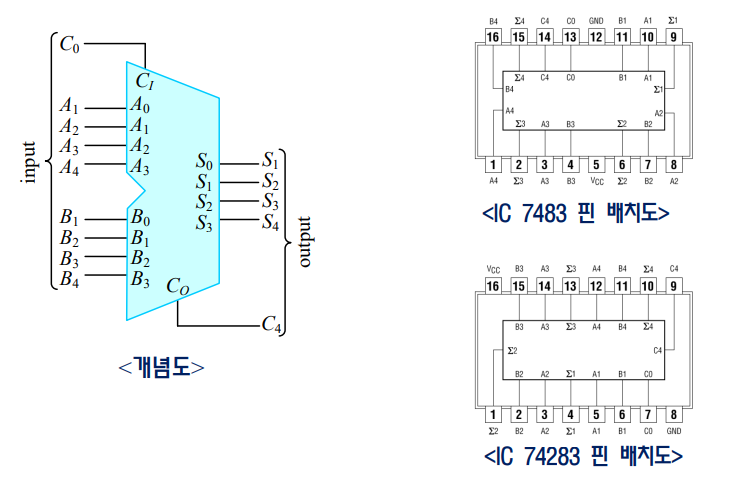


4비트 캐리예측가산기를 하나의 모듈로 만들어서 16비트 캐리예측가산기를 만들어 사용한다.



1. IC 7483/74283

ㅡ4비트 2진 전가산기이며, 내부에 carry look ahead 회로를 내장한다.

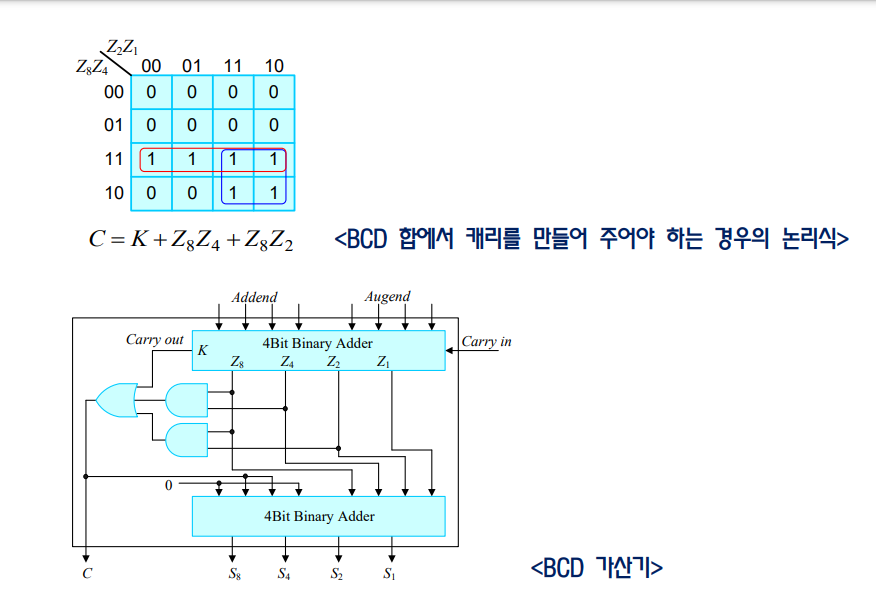


1. BCD 가산기

ㅡBCD 코드는 2진수와 달리 표현범위가 0에서 9까지이다. 그러므로 BCD 계산을 하려면 결과를 보정해 주어야 한다. 2진수 합의 결과가 1010~1111인 경우 보정한다.

ex)6+7=13인 경우

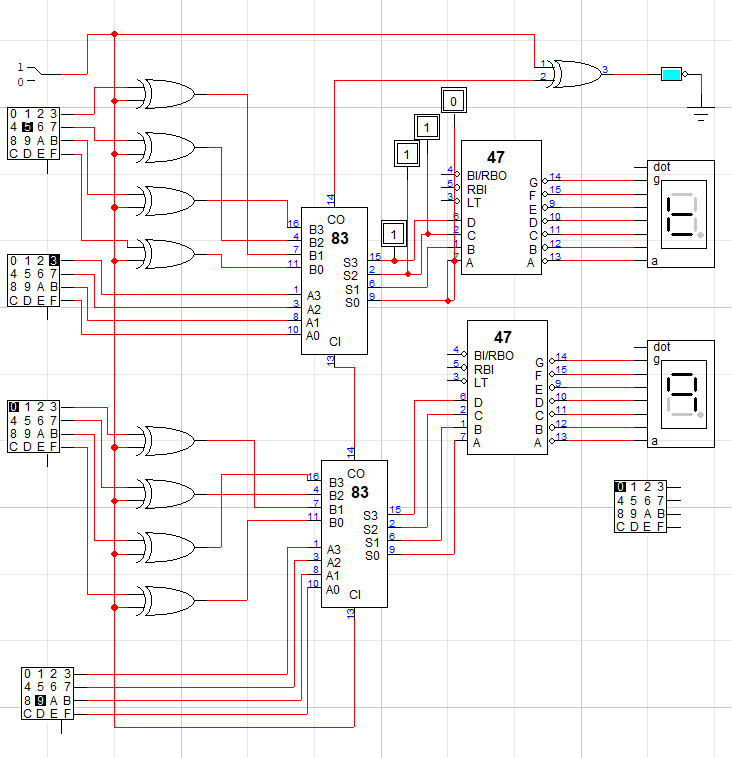
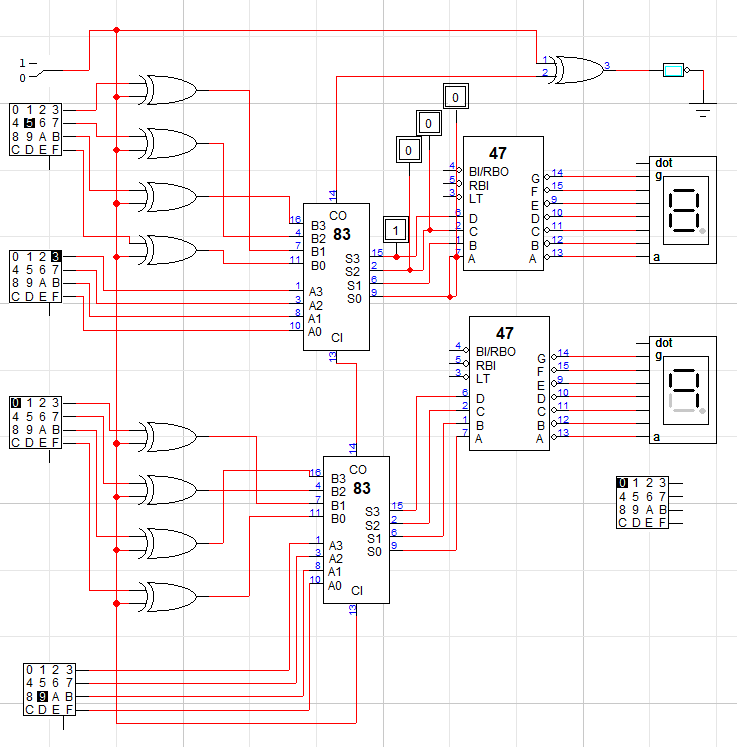
텍스트, 시계이(가) 표시된 사진

자동 생성된 설명

1. 시뮬레이션 과정

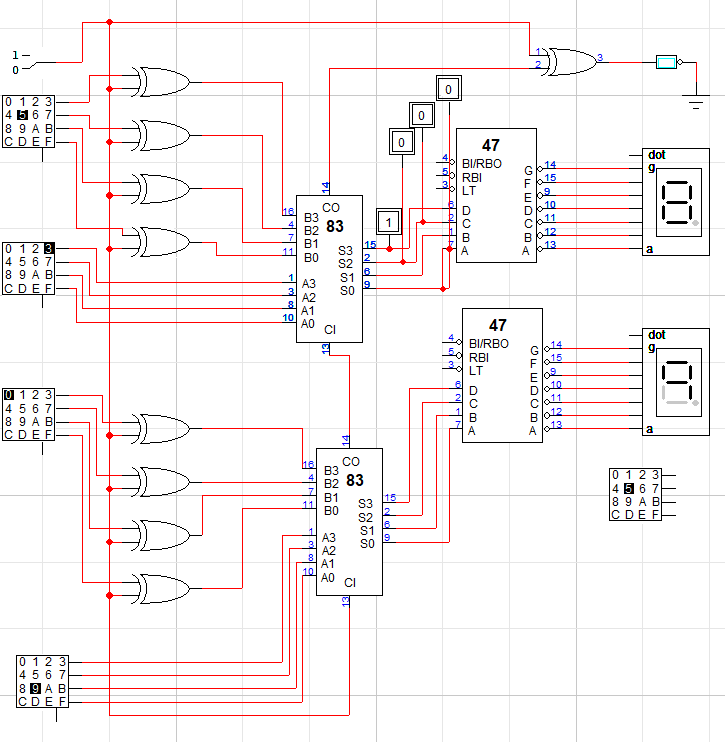
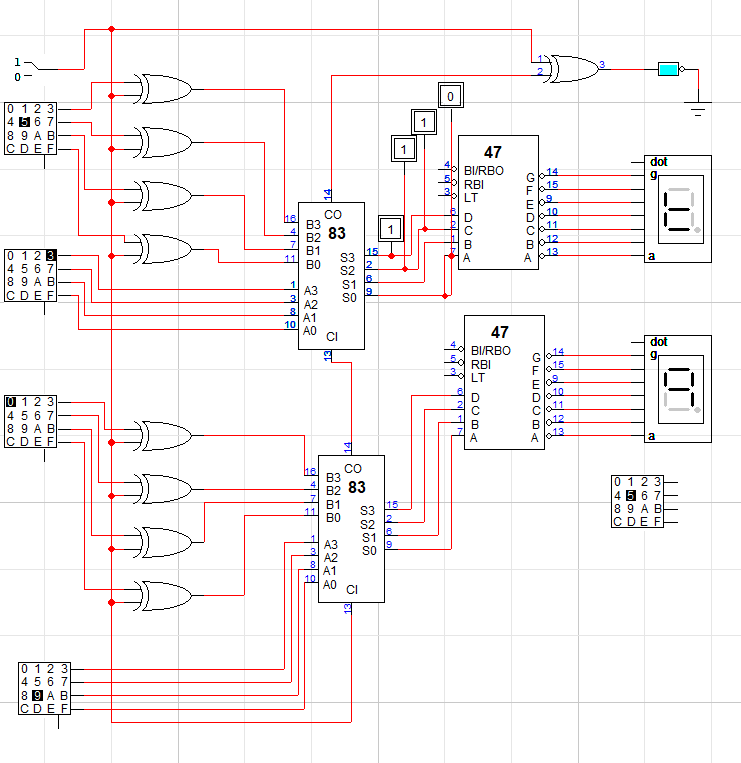
실험 과정은 아래 그림과 같이 회로를 구성한후, 순서대로 실험을 시행 하였

고 다음표와 같은 결과를 얻었다.( 학번 20185309 에서 1 unit에서 5+3 덧셈/5-3 뺄셈, 10unit에서 0+9 덧셈/0-9뺄셈을 수행한 결과를 그림으로 첨부하였다.)



사진과 같이 더하기는 잘 작동하는듯 하나 뺄셈에서 뭔가 값이 원하는대로 나오지 않았다. 회로를 잘못 추가했다는 판단이 들기전 회로와 회로사이를 잘못 연결한거 같다는 생각이 들어서 XOR게이트와 전가산기 사이의 연결을 수정하여 다시 실험을 해봤다. 회로와 회로를 잘못연결했다고 생각이 든 이유는

Binary Probe를 선위에 연결시켜 어떤 값이 나오는지 보자 회로를 잘못 넣었다는 생각이 먼저 안들어서 이다.

(뺄셈에서 5-3의 결과값이 잘못나왔다고 생각이 들었지만 교수님이 올려주신 Function Table을

참고하여 비교해보니 값이 맞게 나온 것을 확인 했다.)

테이블이(가) 표시된 사진

자동 생성된 설명

-동작원리

8비트 데이터 A(5+3부분)와 B(0+9부분)의 입력은 딥 스위치(DIP switch)를 이용한다.

B입력을 부호S(sign)와 XOR하여 전가산기의 입력으로 사용하면 덧셈과 뺄셈이

모두 가능하다. 덧셈인지 또는 뺄셈인지는 슬라이드 스위치를 이용하는데, 1과

연결되면 S=1(High), 0과 연결되면 S=0(Low)가 입력된다.

S=0이면 B의값이 그대로 전가산기로 입력되어 덧셈이 되고, S=1이면 B의 값이

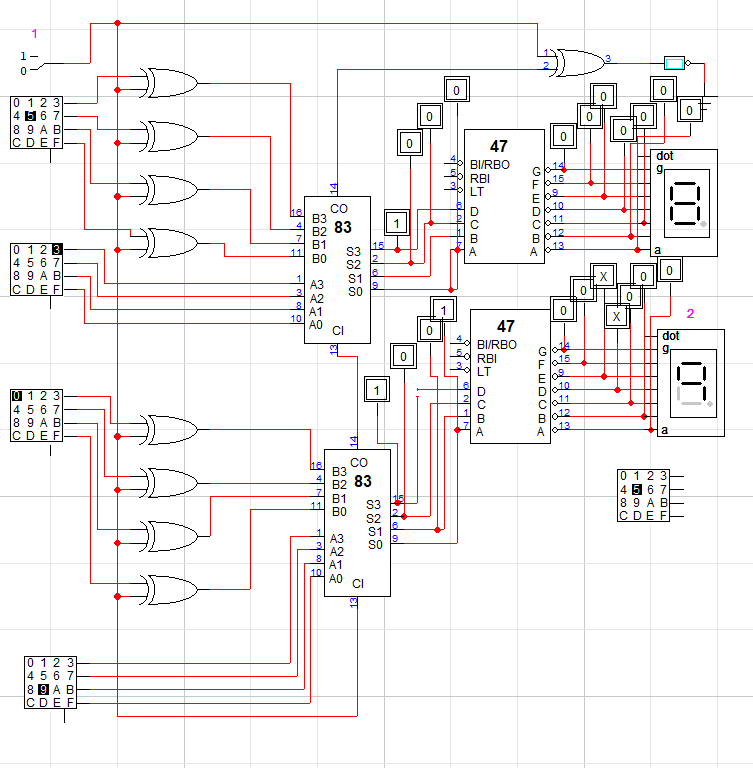
반전, 즉 1의 보수가 되어 입력된다. 또한 뺄셈의 경우(S=1)의 13번 핀의 캐리

입력이 1이 되어 결과적으로 B의 1의 보수에 1이 더해져서 B의 2의 보수가

만들어진다.

IC5는 자리올림/자리내림을 나타내기 위한 것이다.

1. 실험결과와 분석



테이블이(가) 표시된 사진

자동 생성된 설명

위 사진을 통해 살펴보면 우선 Hex keyboard 를 통해 입력한 값은 4개의 숫자로 나타내어진다(ex 5=0101) 그렇게 나눠진 숫자들은 각각 XOR 게이트를 통과하는데 그전에 맨위에있는 바이너리스위치에서 연결한 값과 같이 통과하여 0 혹은 1이 통과하게 된다 계산을 하게 될 경우 자리올림수가 존재하므로 전가산기를 통해 덧셈과 carry를 처리해준다. 사진을 보면 전가산기 뒤쪽에 연결되있는 바이너리프로브를 통해 5+3=8(1000), 0+9=9(1001)이 잘 나온걸 확인할수있다. 그렇게 통과된 숫자들은 BCD가산기를 통과해 Function Table에 맞게 불이 들어오고 꺼지고가 조절되어 원하는 값이 들어온다.

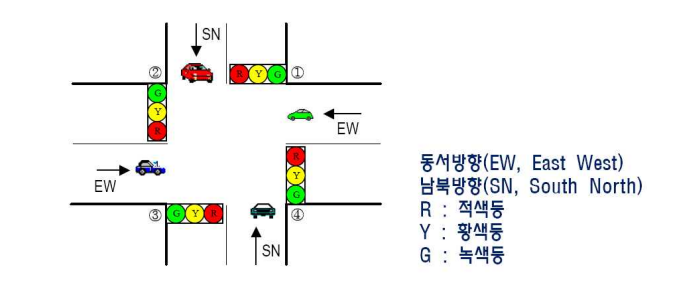
이렇게 조합논리회로 실험결과를 통해 나는 간단한 덧셈/뺄셈이 가능한 회로를 만들어 시뮬레이션을 돌려볼수 있었고 단순히 AND,OR 게이트들로 정말 단순한 숫자들의 합을 구할줄 알았지만 이번 실험을 통해 다양한 숫자들의 연산을 할수 있었고 carry를 통해 자리올림수를 이용한 계산을 할수있다는 걸 알았다.

Ⅱ. 순서 논리 회로 (교통 신호등)

1. 실습목적

NOR 래치회로와 NAND 래치회로의 동작을 이해하고 설명할 수 있다.

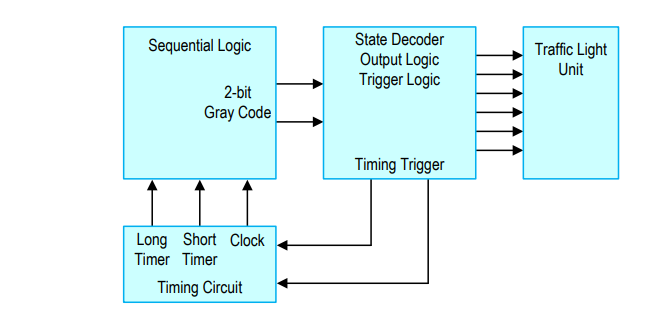
신호등의 점멸시간은 실제와 달리 녹색등(G)은 5초, 황색등(Y)은 1초 동안 켜지는 것으로 가정하여 설계하고 동서방향(EW)의 신호등 ②와 ④의 신호체계는 같고, 남북방향(SN)의 신호등 ①과 ③은 같다.



1. 기본이론

우선 이 회로는 순서논리회로인데, 순서논리회로는 현재의 입력과 이전의 출력상태에 의해서 출력이 결정되는 논리회로이며 순서논리회로는 신호의 타이밍(timing)에 따라 동기 순서논리회로와 비동기 순서논리회로로 분류된다. 또한동기 순서회로에서 상태(state)는 단지 이산된(discrete) 각 시점 즉, 클록펄스가 들어오는 시점에서 상태가 변화하는 회로이며 비동기 순서회로는 시간에 관계없이 단지 입력이 변화하는 순서에 따라 동작하는 논리회로이다. 즉 순서는 상태마다 다른 신호가 필요하므로 이번 실습에서는 순서논리회로로 이용해 신호등을 구현한다.

시스템 블록도에 대해 살펴보면 타이밍 회로는 녹색등과 황색등이 켜지는 5초와 1초 간격의 시간구간 및 시스템을 주기 적으로 반복 동작시키기 위한 클록신호를 발생하는 회로이다. 순서논리회로는 타이밍 회로에서 발생하는 신호를 입력으로 하여 4가지 상태를 발생하는 그레이 코드 카운터이다 상태 디코더는 4가지 상태의 순서에 따라 신호등을 점멸시키기 위한 출력 로직과 5초와 1초의 시간을 트리거시키는 신호를 발생시키기 위한 트리거 로직으로 이루어져 있다. 신 호등 유니트는 출력로직에 따라 신호등이 점멸되는 회로이다.



위 사진에서 하나하나 보면

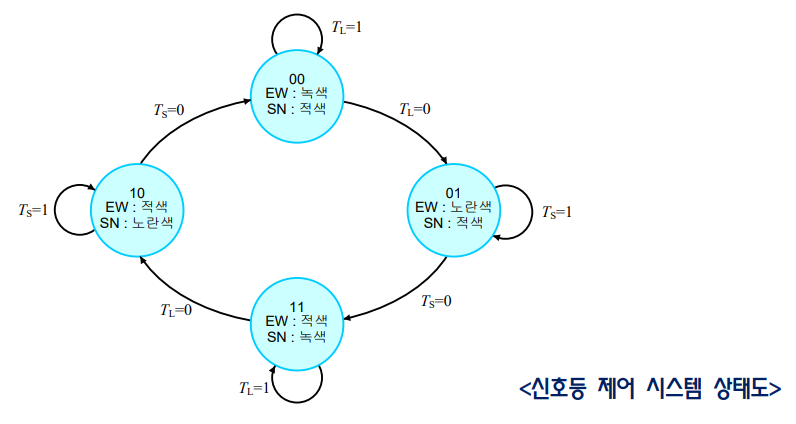
첫번째로 Timing Circuit은 5초,1초 간 시스템을 주기적으로 반복동작을 시키는 회로이다.

두번째로 Sequential Logic 부분은 타이밍 회로에서 발생하는 신호를 입력으로 신호등의 4가지 상태를 발생시키는 회로이다.

세번째로 State Decoder 부분은 Output Logic,Trigger Logic,Timing Trigger 세가지로 나눌수 있는데 Output Logic은 4가지 상태의 순서에 따라 신호등을 점멸시키기 위한 출력 논리회로이다. Trigger Logic은 5초와 1초의 시간에 반응하여 트리거시키는 신호를 발생시키기 위한 트리거 논리 회로 이다. Timming Trigger는 Timing Circuit의 입력을 담당하는 회로이다.

네번째로 Traffic Light Unit은 Output Logic에 따라 신호등이 점멸되는 회로를 나타내는데, 총 6가지의 신호가 존재하므로 화살표가 6가지가 있음을 확인할 수 있다.

신호등 제어 시스템 상태도를 보면 TL=1은 5초 타이머가 on됨을 의미하고, TL=0은 off되었다는 것을 의미한다. TS=1은 1초 타이머가 on됨을 의미하고, TS=0은 off되었다는 것을 의미한다. 상태도의 원 안에 보이는 2비트는 4가지 상태를 각각 그레이 코드로 표시한 것이다.

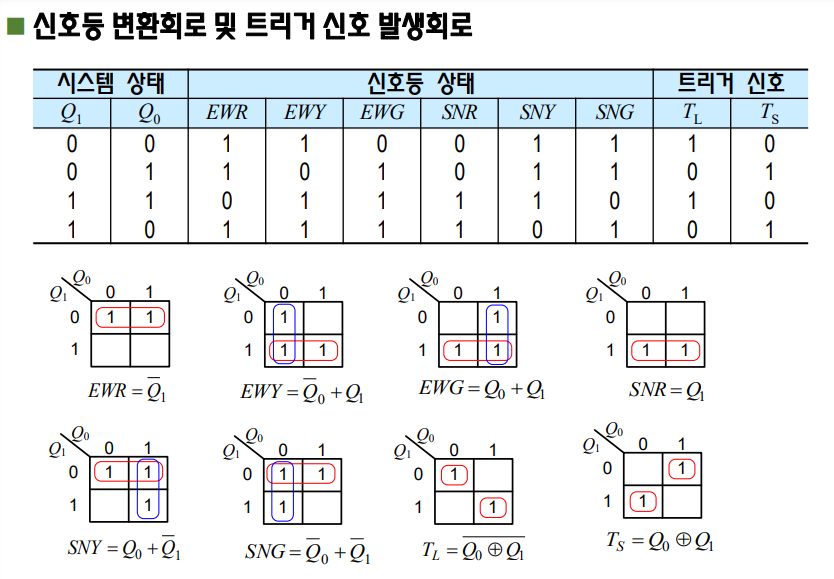


두 타이머의 입력에 따라 표를 나타내면 다음과 같고 카르노맵을 이용해 플립플롭의 입력의 수식을 알수있다. 위의 상태도를 상태표로 나타내면 밑의 그림과 같다.

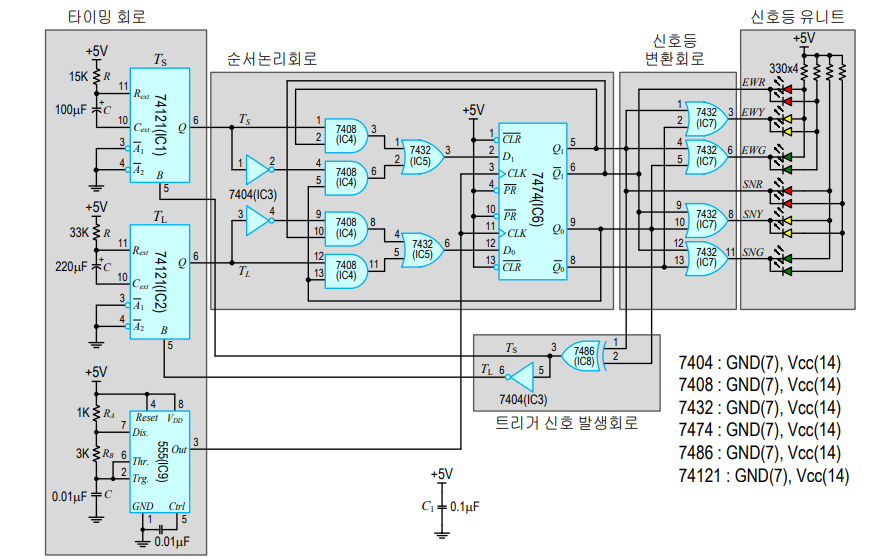
테이블이(가) 표시된 사진

자동 생성된 설명

밑의 사진과같이 신호등을 4가지 상태로 나눈다.



전체 회로도를 보면 이러하다.



1. 시뮬레이션 과정

3-1 상태도

텍스트, 시계이(가) 표시된 사진

자동 생성된 설명

입력변수만 있고 출력변수는 없는 상태에서 상태변화가 일어난다

3-2 상태표

위의 상태도를 상태표로 나타내면 다음과 같다

테이블이(가) 표시된 사진

자동 생성된 설명

3-3 플립플롭 결정,수

정의해야할 상태의 수가 n가지이면 log\_2 n 개의 플립플롭이 필요하므로

상태의 수는 4가지 이므로 2개의 플립플롭이 필요하다.

이 회로는 두 입력을 출력값으로 갖는 D플립플롭을 사용한다.

3-4. 상태여기표 작성

우리는 4가지의 상태를 만들고 플립플롭이 두개가 필요하므로

상태여기표를 작성하면

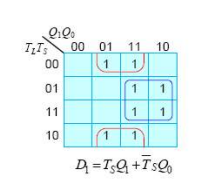
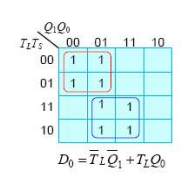
텍스트, 낱말맞추기게임, 영수증이(가) 표시된 사진

자동 생성된 설명

이렇게 나온다.

3-5. 간소화 방법으로 출력함수, 플립플롭 입력함수 구하기

위에 표를 통해 플립플롭의 입력 함수를 구해보면 다음과같다.



그리고 출력함수는 신호등상태에서 나오므로 신호등 상태의 함수 식을 구해보면

테이블이(가) 표시된 사진

자동 생성된 설명

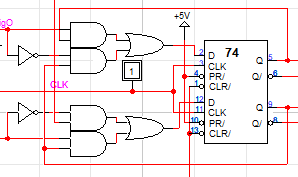
따라서 식은

텍스트, 시계, 묶음, 다른이(가) 표시된 사진

자동 생성된 설명

3-6. 순서 논리회로도 작성

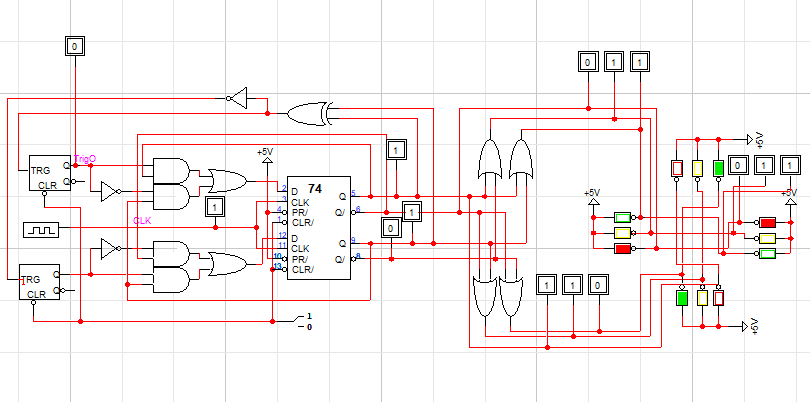
위에 순서대로 구한결과를 토대로 신호등에서의 순서논리회로를 작성해보면



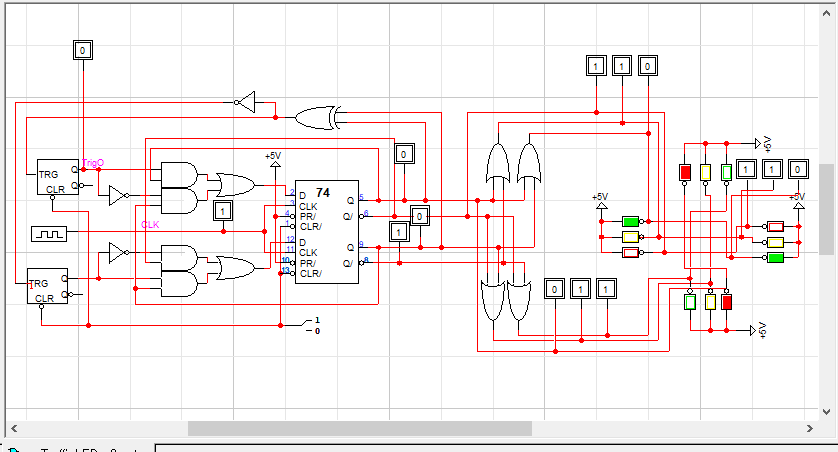
이렇게 구현할 수가 있다.

1. 실험결과와 분석

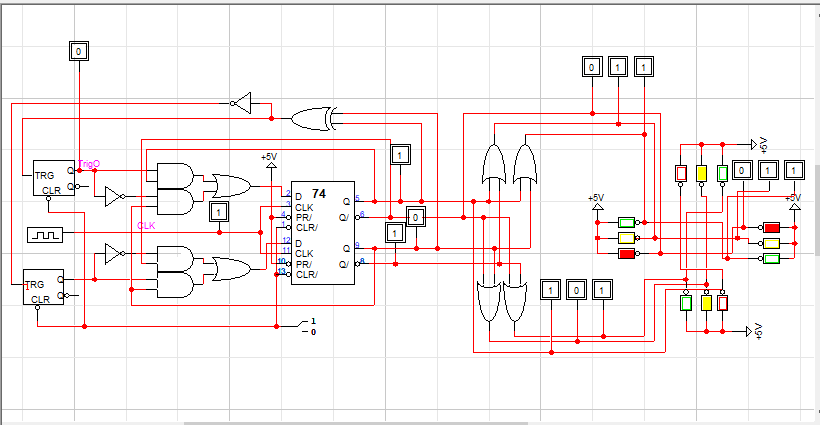
EW 교통 대기(적색), SN 교통 통과(녹색) 실험 결과



EW 교통 통과(녹색), SN 교통 대기(적색) 실험 결과



EW 교통 대기(적색), SN 교통 통과멈춤 (황색) 실험 결과



신호등 회로를 만들 때 클럭을 이용해 일정주기마다 신호를 전달해주어 회로의 출력결과를 계속 바꿔줄 수 있다는 걸 알았다. 보면 110 일땐 녹색, 101일땐 황색, 011이면 적색으로 신호등을 나타내는 led에 빛이 난다.

순서논리회로로 클럭을 통해 값이 계속 일정하게 바뀌는걸 알았고, 실제로 바이너리프로브 출력으로 그 값을 확인했다.