컴퓨터 공학 기초 설계 및 실험1

예비 & 결과 보고서

실험제목: **De Morgan’s theorem, Karnaugh map, XOR gate**

실험일자: 2018년 03월 22일 (금)

제출일자: 2018년 03월 27일 (수)

학 과:컴퓨터정보공학부

담당교수: 이혁준

실습분반: 03-금012

학 번: 2018202074

성 명: 김상우

결과보고서

1. 제목 및 목적
   1. 제목

De Morgan’s theorem, Karnaugh map, XOR gate

* 1. 목적

이전보다 복잡한 회로를 설계하여 기존의 이해도를 발전시킨다. 더불어 데모르간의 정리와 카르노맵을 활용해 회로를 표현, 간소화하며 새로운 방법론을 익히는 걸 주목적으로 한다. 이 과정을 통해 ‘말아감기’를 비롯한 회로의 간소화 방법 및 그들간의 법칙을 직접 체험, 이해한다. 추가적으로 XOR gate를 사용, 앞서 나온 논리게이트와 비교하며 어떠한 결과를 산출해내는지 이해한다.

1. 실험 결과

<실험 1> 출력값이 2개인 NOT, NAND, NOR의 혼합 회로



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | U | V | W | X | Y |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | +5 | 0 | 0 | 0 | 0 | 0 |
| 0 | +5 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | +5 | +5 | 0 | 1 | 1 | 1 | 1 |
| +5 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| +5 | 0 | +5 | 1 | 0 | 1 | 1 | 1 |
| +5 | +5 | 0 | 0 | 0 | 1 | 0 | 0 |
| +5 | +5 | +5 | 1 | 1 | 1 | 1 | 1 |

U = ( ( A \* C )’ )’ = A \* C //정리 9

V = ( ( B \* C )’ )’ = B \* C //정리 9

X = ( (U + V)’ )’ = U + V = ( A \* C )+( B \* C ) = C \* (A + B) //정리 3

W = ( B’ \* A’ )’ = B + A //데모르간의 정리

Y = ( ( W \* C )’ )’ = W \* C = ( A + B ) \* C //정리 9

논리식 : X = Y = ( A + B ) \* C

<실험 2> 3-input AND, OR, NOT이 사용된 회로



**카르노 맵 진리표:**

|  |  |  |  |
| --- | --- | --- | --- |
| 입력 | | | 출력 |
| A | B | C | X |
| 0 | 0 | 0 | 0 |
| 0 | 0 | +5V | 0 |
| 0 | +5V | 0 | 1 |
| 0 | +5V | +5V | 0 |
| +5V | 0 | 0 | 0 |
| +5V | 0 | +5V | 0 |
| +5V | +5V | 0 | 1 |
| +5V | +5V | +5V | 0 |

**카르노 맵:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | BC | 00 | 01 | 11 | 10 |
| 0 | | 0 | 0 | 0 | 1 |
| 1 | | 0 | 0 | 0 | 1 |

카르노 맵을 보면 B=1, C=0일때만 1이며 A에는 상관없는 모습을 보인다.

즉 논리식은 B \* C’가 된다.

<실험 3> 입력값이 4개인 AND, NOT, OR로 이뤄진 회로



**카르노 맵 진리표:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **입력** | | | | **출력**  **X** |
| **A** | **B** | **C** | **D** |
| **0** | **0** | **0** | **0** | **1** |
| **0** | **0** | **0** | **+5** | **1** |
| **0** | **0** | **+5** | **0** | **0** |
| **0** | **0** | **+5** | **+5** | **0** |
| **0** | **+5** | **0** | **0** | **1** |
| **0** | **+5** | **0** | **+5** | **1** |
| **0** | **+5** | **+5** | **0** | **0** |
| **0** | **+5** | **+5** | **+5** | **1** |
| **+5** | **0** | **0** | **0** | **1** |
| **+5** | **0** | **0** | **+5** | **0** |
| **+5** | **0** | **+5** | **0** | **0** |
| **+5** | **0** | **+5** | **+5** | **0** |
| **+5** | **+5** | **0** | **0** | **0** |
| **+5** | **+5** | **0** | **+5** | **1** |
| **+5** | **+5** | **+5** | **0** | **0** |
| **+5** | **+5** | **+5** | **+5** | **1** |

**카르노맵 :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **AB** | **CD** | **00** | **01** | **11** | **10** |
| **00** | | **1** | **1** | **0** | **0** |
| **01** | | **1** | **1** | **1** | **0** |
| **11** | | **0** | **1** | **1** | **0** |
| **10** | | **1** | **0** | **0** | **0** |

**논리식 : (A’ \* C’) + (B \* D) + (B’ \* C’ \* D’) = B \* D + (B + C + D)’ + (A + C)’**

<실험 4> XNOR회로의 출력값



|  |  |  |  |
| --- | --- | --- | --- |
| **입력** | | **출력** | |
| **A** | **B** | **Z** | **Z’** |
| **0** | **0** | **1** | **0** |
| **0** | **+5** | **0** | **1** |
| **+5** | **0** | **0** | **1** |
| **+5** | **+5** | **1** | **0** |

**논리식 : Z = ( ( ( A’ \* B )’ )’ + ( ( A \* B’ )’ )’ )’=( ( A’ \* B ) + ( A \* B’ ) )’**

**=(A’\*B)’\*(A\*B’)’=(A+B’)\*(A’+B)=AA’+AB+B’A’+BB’=AB+A’B’**

<실험 5>2진수 비교기 회로



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A1 | A2 | A3 | A4 | B3 | B2 | B1 | B0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | +5 | +5 | 0 | 0 | 1 | 1 | 0 |
| +5 | +5 | +5 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | +5 | 0 | 0 | 0 | 1 |
| +5 | 0 | +5 | 0 | 1 | 0 | 1 | 0 |

1. 고찰

이번 실험에서 가장 문제가 되었던 것은 다름아닌 부품간의 접촉 정도였습니다. 회로가 제대로 꽂히지 않아 0과 1이 반복되거나 혹은 한동안 1이 출력되다가 0이 출력되는 경우도 생겼습니다. 이 때문에 상당히 많은 시간이 낭비되었으며 적절하지 못한 결과 또한 출력되었습니다. 이는 선을 비롯한 부품의 개수를 늘릴 때마다 발생하는 빈도가 늘었는데, 접지되는 것이 아니더라도 서로가 접촉하여 연결부위를 헐겁게 만든 것이 원인으로 보입니다. 이점을 방지하기 위해선 인접한 선끼리 묶어주는 도구를 사용하거나 빠지지 않도록 고정해 줄 필요가 있다고 생각합니다. 또한 XOR회로는 입력 값을 비교하는 데 사용하여 새로운 조건을 생성하고 더 다양한 회로를 설계 및 구현할 수 있을 것입니다.

예비보고서

1. 제목 및 목적
   1. 제목

Latch & Flip Flop

* 1. 목적

기억소자로 사용되는 Latch의 종류와 그 쓰임새에 대해 이해하고 부품이 가지는 특성과 작동 원리를 이해하는 걸 목적으로 한다. 또한 Flip Flop의 정의와 쓰임을 이해하고 Latch와의 차이점을 확실히 이해하는 것도 목적으로 한다. 결과적으로 출력 값이 순간적으로 나간 후 사라지는 게 아니라 결과값이 보존되어 연속적으로 사용되는 회로를 만들 수 있는 능력을 익히는 걸 최종 목적으로 한다.

1. 원리(배경지식)

기억소자 : 입력 값에 의해 발생된 상황이 어느 정도 유지되게 만드는 소자이다.

Latch: 1 이상의 bit들을 일시적으로 유지, 기억하기 위한 디지털논리회로

R-S래치(Latch) :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | |  | | | | |
| S | R | Qa | Qb | 특징 | S | R | Qa | Qb | 특징 |
| 0 | 0 | 0/1 | 1/0 | 값의 유지 | 0 | 0 | 0/1 | 1/0 | warning |
| 0 | 1 | 0 | 1 |  | 0 | 1 | 1 | 0 |  |
| 1 | 0 | 1 | 0 |  | 1 | 0 | 0 | 1 |  |
| 1 | 1 | 0 | 0 | warning | 1 | 1 | 0 | 0 | 값의 유지 |

\*S는 set R은 reset을 의미하며 값의 유지는 이전에 입력 값을 유지함을 의미

Warning: 해당 입력 값을 넣고 결과값을 받은 후 값을 유지했을 때 값이 변경됨. Ex)XOR-회로에서는 S와 R에 1을 입력하고 0을 넣으면 결과값이 둘 다 1로 변경 또한 노이즈 발생시 쉽게 상태가 뒤집어짐.

\*S-R latch의 상위 latch

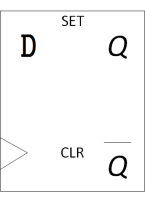
|  |  |  |
| --- | --- | --- |
| 이름 | Gated S-R Latch | Gated D latch |
| 회로  (XOR) |  |  |
| 회로  (NAND) |  |  |
| 도식 |  |  |
| 장점 | ‘Clk’의 도입으로  노이즈 위험을 줄임 | S-R latch와 Gated S-R latch의  warning 문제도 해결 |

그러나 S-R랜치를 보완한 D래치도 Clk 신호가 켜져있는 동안 D라는 신호가 불안정할 시 오작동 가능성 존재한다는 단점이 있다.

플립 플롭(Flip flop) : 2개의 안정상태를 가지는 회로로, 특정 입력 값이 변하지 않는 이상 상태를 유지한다.

일반적으로 플립-플롭은 1 비트의 데이터를 저장하는 장치를 의미하며 래치는 단일 트리거를 사용하여 임의의 수의 데이터 비트를 저장하는 장치를 나타낼 수도 있다.

:RS 플립플롭: Gated S-R Latch와 유사하게 생긴 것을 알 수 있다.

:D 플립플롭: D입력을 사용하는 플립플롭의 통칭 Gated D latch와 유사하게 생긴 것을 알 수 있다.

플립플롭의 활용

-레지스터: 플립플롭의 대량 연결을 통해 여러비트를 저장

-카운터: 플립플롭의 대량 연결로 클럭 펄스(clk)가 펄스 수를 세거나 회로 동작제어

1. 참고문헌

이원석 정길수 / 논리회로실험 / 생능출판 / 2010

플립플롭/ <https://blog.naver.com/roboholic84/221239779527>

래치, 플립플롭/ <https://blog.naver.com/hafs_snu/221277740243>