디지털시스템설계 실습 보고서

불 대수식의 단순화

20210716 최대현

1. 실습 개요

이번 실습에서는 불 대수식을 단순화하는 방법에 대해 이해한다. 또한, K-map 기법을 통해 불 대수식을 단순화하고 단순화 전후를 비교해서 효과를 확인한다.

또한, k-map 알고리즘을 명확히 이해하고 불 대수식의 단순화를 와이어 개수와 논리 게이트 개수라는 정량적인 수치를 통해 그 효과를 확인한다.

2. 이론적 배경

1) 불 대수식

결과값이 항상 0과 1로만 나오는 대수식으로, 그 이름은 영국의 수학자 부울(G. Boole)의 이름을 따서 지어졌다. 불 대수는 논리회로를 설계하고 분석하는데 필요하다. 불 대수 덧셈(Boolean addition)은 OR 게이트를 이용하고 연산자는 +이다. 불 대수 곱셈(Boolean product)은 AND 게이트를 이용하고 연산자는 ·(곱셈)이다.

2) 불 대수식의 단순화

불 대수식의 복잡도는 이 식을 회로로 구현하는데 사용된 와이어와 논리 게이트(not 게이트는 포함되지 않을 수 있음)의 개수를 비교하여 평가한다. 불 대수식을 단순화하면 와이어와 논리 게이트를 적게 사용할 수 있고, 이는 곧 소비 전력를 감소시키면서도 작동 속도를 빠르게 할 수 있다. 따라서, 불 대수식의 단순화는 여러모로 장점이 지배적이므로 단순화가 가능하다면 최대한 단순화한 뒤 회로로 구현하는 것이 바람직하다. 불 대수식을 단순화하는 대표적인 방법으로는 카르노 맵(Karnaugh Map, K-map)과 퀸-매컬러스키(Queen-McCluskey, QM) 알고리즘이 있다.

3. 실험 준비

1) 2-Bit Magnitude Comparator의 세 출력 각각에 대한 식을 단순화하지 않고 작성한다.

i) A>B

GT=$A1'A0B1'B0 + A1A0B1'B0' + A1A0B1'B0 + A1A0B1B0' + A1A0'B1'B0' + A1A0'B1'B0$

ii) A=B

EQ = $A1'A0'A1'B0' + A1'A0B1'B0 + A1A0B1B0 + A1A0'B1B0'$

iii) A<B

LT = $A1'A0'B1'B0 + A1'A0'B1B0 + A1'A0B1B0 + A1'A0B1B0' + A1'A0'B1B0' + A1A0'B1B0'$

2) k-map을 활용하여 세 식을 단순화한다.

단순화하기 위해 각 경우에 대한 K-map을 그려보면 다음과 같다.

i. A>B

A1A0 | B1B0	**00**	**01**	**11**	**10**
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

EPI를 먼저 구한 뒤 부족한 부분을 PI로 채우는 방식으로 식을 단순화해보았다. 그 결과, 다음과 같다.

GT = $A1B1' + A1A0B0 + A0B1'B0'$

ii. A=B

A1A0 | B1B0	**00**	**01**	**11**	**10**
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

EPI를 먼저 구한 뒤 부족한 부분을 PI로 채우는 방식으로 식을 단순화해보았다. 그 결과, 다음과 같다.

EQ= $A1'A0'B1'B0' + A1'A0B1B0 + A1A0B1B0 + A1A0'B1B0'$

iii. A>B

A1A0 | B1B0	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

EPI를 먼저 구한 뒤 부족한 부분을 PI로 채우는 방식으로 식을 단순화해보았다. 그 결과, 다음과 같다.

$$LT = A1'B1 + A1'A0'B0 + A0'B1B0$$

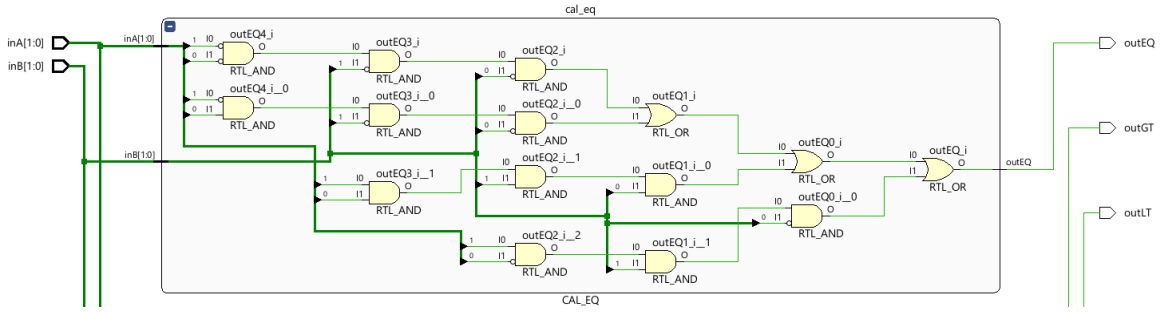
4. 실험

본 실험은 Verilog 프로그래밍으로 회로를 구현하며 진행되었다.

4-1. lab4_1.v

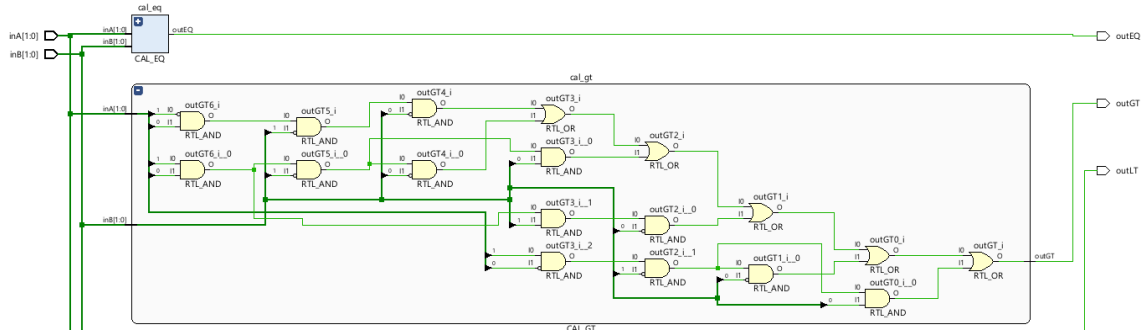
CalLT, CalEQ, CalGT 3개의 모듈을 Gate-Level Modeling으로 구현하되, assign keyword와 ~, &, | operator를 사용해서 코드를 작성했다. 그 뒤, schematic 기능으로 회로도를 출력해보았고 Netlist를 참조하여 와이어 개수와 논리 게이트 개수를 확인했다.

i) Module calEQ:



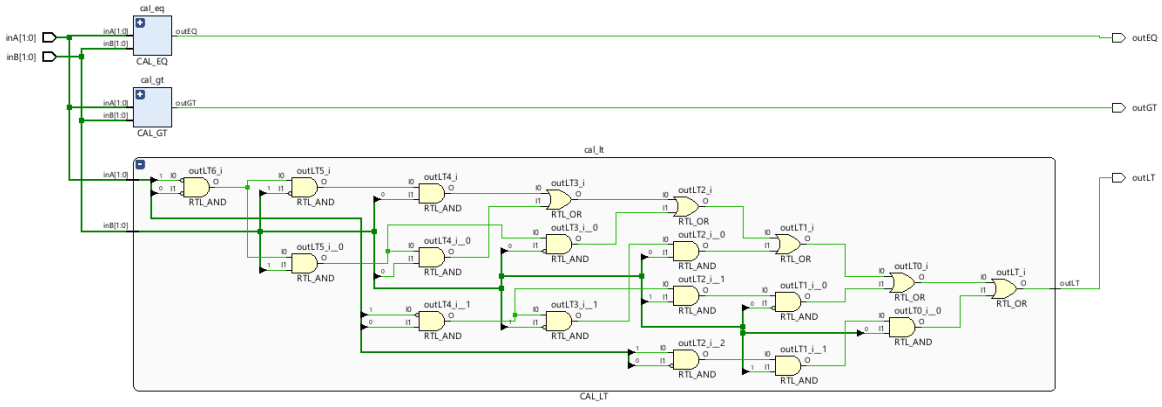
<Schematic 기능으로 출력한 회로도>

ii) Module calGT

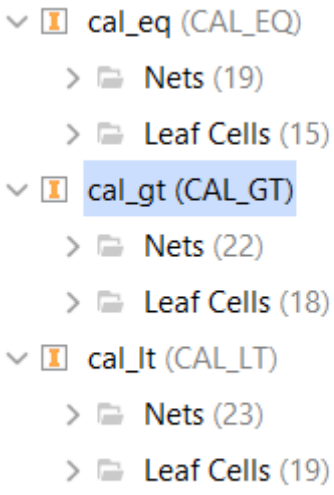


<Schematic 기능으로 출력한 회로도>

iii) Module calLT



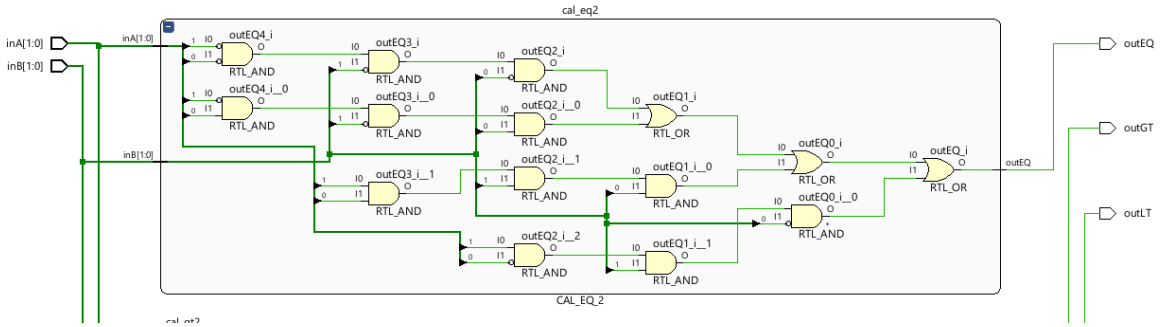
<Schematic 기능으로 출력한 회로도>



<각 모듈의 와이어 개수와 논리 게이트 개수>

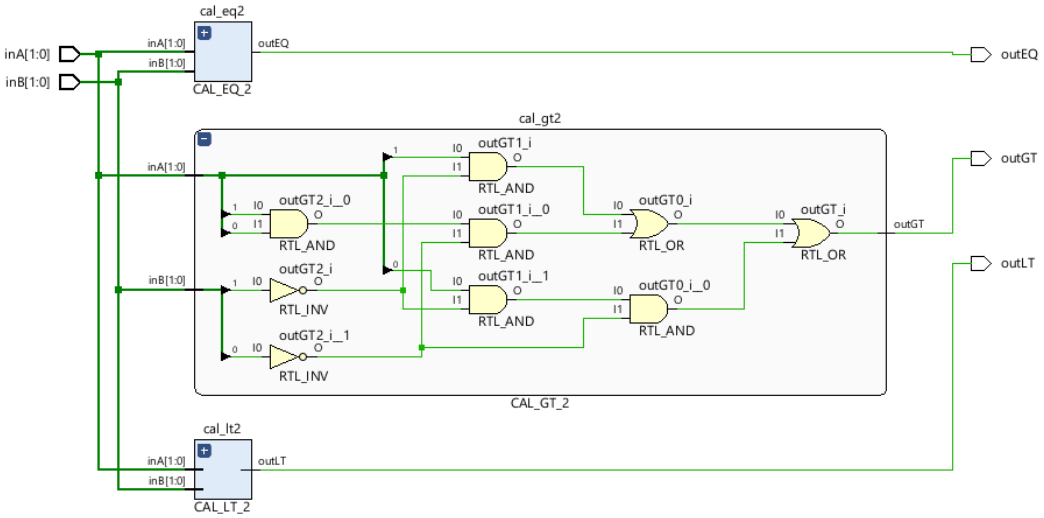
4-2. lab4_2.v

i) Module calEQ_2



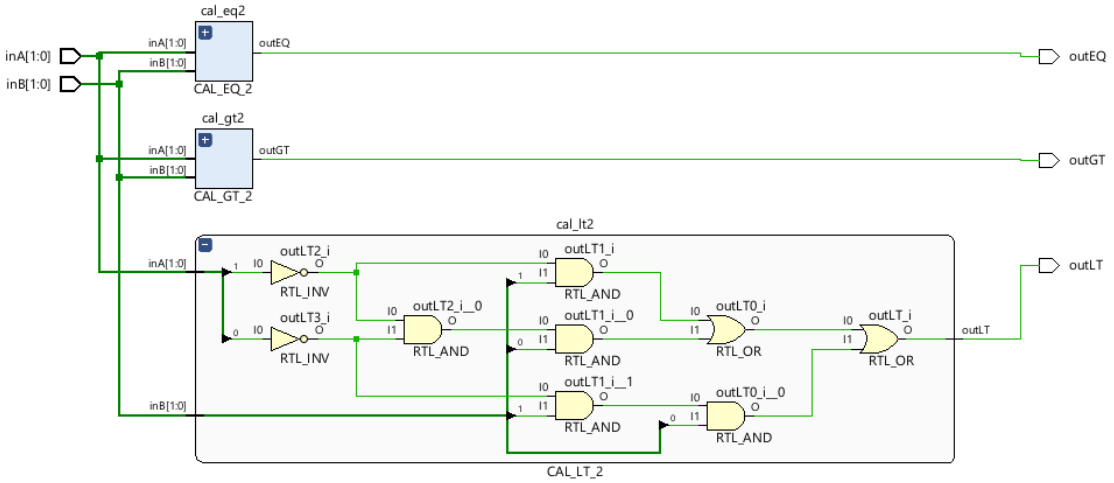
<Schematic 기능으로 출력한 회로도>

i) Module calLT_2

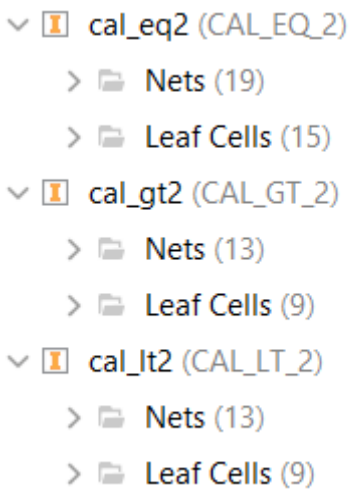


<Schematic 기능으로 출력한 회로도>

ii) Module calLT_2



<Schematic 기능으로 출력한 회로도>



<각 모듈의 와이어 개수와 논리 게이트 개수>

5. 결론

Cal_eq의 경우 k-map 내에서 묶을 수 있는 경우가 없어서 simplify되지 않은 결과로 nets와 leaf cells가 그대로이다. 그러나, cal_gt와 cal_lt에서는 각각 literal을 24개에서 8개로 줄인 것에서 예상할 수 있듯이 nets 22개, cell 18개에서 net 13개, leaf cell 9개로 줄일 수 있었다. 즉, 와이어와 게이트 수를 둘 다 유의미하게 줄일 수 있었고 결론적으로 전체 회로를 보았을 때 회로의 작동 속

도는 빨라지고 사용 전력은 줄어들었다. 따라서, k-map 알고리즘을 통해 불 대수식을 성공적으로 simplify했다고 볼 수 있다.

6. 고찰

이번 실습에서는 불 대수식을 단순화하는 방법을 알아보고, 그 방법 중 하나인 K-map 알고리즘을 적용해서 실제로 불 대수식을 단순화하기 전과 후를 나눠 verilog로 구현해보는 시간을 가졌다. 그 결과 회로가 효율적으로 simplify되었음을 schematic을 통해 확인할 수 있었다. 기회가 된다면 QM method를 적용하는 방법을 익혀 QM method로 simplify한 회로도 구현해보고 싶다.