실험 2. 불 대수식의 단순화

2024. 03. 27.

디지털 시스템 설계 (CSED273)

1. 개요

불 대수식을 단순화하는 방법에 대해 이해하고, 단순화 전후를 비교하여 그 효과를 확인한다. 세부적인 학습 목표는 다음과 같다

.

- K-map 알고리즘 이해
- 와이어와 논리 게이트 개수를 확인하여 단순화 효과 확인

2. 이론적 배경

1) 불 대수식의 단순화

불 대수식의 복잡도는 그 식을 실제로 구현하는데 사용된 와이어와 논리 게이트 개수로 평가한다. 불 대수식을 단순화하면 자연스럽게 회로에 들어가는 와이어와 게이트의 수를 줄일 수 있다. 또 회로를 단순화시킴으로써 소비 전력을 줄이고, 작동 속도를 높일 수 있다

.

대표적인 단순화 방법으로 카노 맵(Karnaugh-Map, K-Map)과 퀸 매클러스키 (Quine-McCluskey, QM) 알고리즘이 있다.

2) 2-Bit Magnitude Comparator

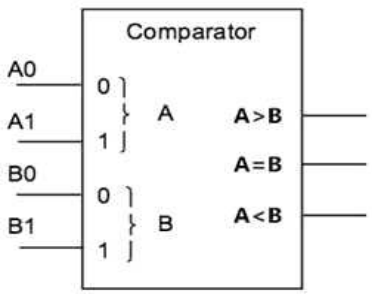

그림 1 2-bit Magnitude Comparator

2-Bit Magnitude Comparator란 서로 다른 두 개의 2-Bit 수(A, B)가 입력으로 주어졌을 때, 둘의 대소 관계를 판별하여 알맞은 출력을 1로, 나머지 출력은 0으로 설정하여 출력해 주는 회로를 의미한다. 그림 1과 같이 입력이 주어지고 입력 A1 과 B1이 더 높은 자릿수라 가정할 경우, 출력 “A>B”는 0, 출력 “A=B”는 1, 출력 “A<B”는 0이 된다.

.

입력		A1A0			
		00	01	11	10
B1 B0	00	1	0	0	0
	01	0	1	0	0
	11	0	0	1	0
	10	0	0	0	1

표 1 입력 A, B에 따른 출력 "A=B"

3. 실험 준비

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

2) K-map을 활용하여 세 식을 단순화한다.

* 보고서 필수 내용
  - 단순화시키기 전의 식
  - K-map 적용 과정과 최종 식

4. 실험

1) 공통 유의사항

- 주어진 코드에서 */* Add your code here */* 로 주석 처리된 부분만 수정하여 구현한다.
- 각 모듈은 Gate-Level Modeling으로 구현한다.
  (assign keyword와 "~","&","|" operator 사용 가능)

* 보고서 필수 내용
  - Schematic 기능으로 생성한 회로도 캡처
    - 하위 모듈의 회로도도 캡처해야 한다.

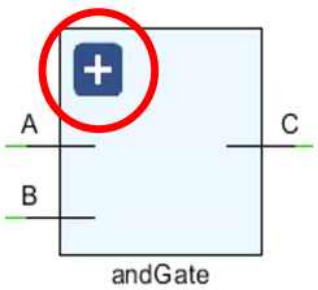

그림 2

- Netlist를 참조하여 와이어 개수와 논리 게이트 개수 캡처
  - 와이어는 Nets, 논리 게이트는 Leaf Cells 수로 확인
  - Inverter(NOT)의 수는 구현에 따라 포함되지 않을 수 있음

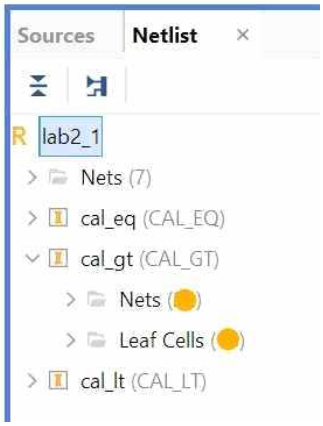

그림 3

2) 단순화 이전 - lab2_1.v

ㄱ. 2-Bit Magnitude Comparator를 단순화하기 전 식을 구현한다.
ㄴ. Schematic 기능으로 회로가 잘 구현되었는지 확인한다.
ㄷ. RTL ANALYSIS > Netlist를 참조하여 와이어와 논리 게이트 개수를 확인한다.

3) 단순화 이후 - lab2_2.v

ㄱ. K-map으로 단순화한 세 가지 출력을 구현한다.
ㄴ. Schematic 기능으로 회로가 잘 구현되었는지 확인한다.
ㄷ. RTL ANALYSIS > Netlist를 참조하여 와이어와 논리 게이트 개수를 확인한다.
ㄹ. 실험 1과 와이어 및 논리 게이트 개수를 비교한다.

5. 제출

{학번}_lab2.zip으로 다음 파일을 압축하여 PLMS로 제출한다.

- lab2_1.v // ***코드***
- lab2_2.v
- lab2_report.pdf // ***보고서***