5주차 예비보고서

전공: 컴퓨터공학과 학년: 4학년 학번: 20191583 이름: 김태곤

**1.De Morgan의 정리에 대해 조사하시오.**

De Morgan의 정리는 논리합은 논리곱과 부정기호로, 논리곱은 논리합과 부정기호로 표현할 수있음을 가리키는 법칙이다. 즉, 회로 연산에서 AND와 OR 연산을 서로 바꾸고, 각 변수의 보수를 취한 것을 의미한다. 이는 논리식의 변환에 있어 필수적인 도구로 활용되고 있다. De Morgan의 정리는 두 가지 기본 형태로 표현된다.

드모르간의 제1 법칙 : 두 입력 A와 B의 AND 연산에 대한 NOT이 A의 NOT과 B의 NOT의 OR 연산과 같다는 것을 의미한다.

드모르간의 제2 법칙 : 두 입력 A와 B의 OR 연산에 대한 NOT이 A의 NOT과 B의 NOT의 AND 연산과 같다는 것을 의미한다.

드모르간의 법칙은 논리 회로를 단순화 시키고 최소화 시키며, 논리식 변환에 있어 중요한 역할을 한다. 논리 회로의 설계 및 분석, 소프트웨어 개발 과정에서 논리 연산을 효과적으로 다루기 위한 기본적인 도구로 사용되며 복잡한 논리 문제를 해결하는 데 필수적인 역할을 하기 때문에 디지털 논리, 컴퓨터 과학, 전자공학 분야에서 광범위하게 사용한다..

**2.논리회로의 간소화에 대해 조사하시오(예시 포함).**

|  |  |  |
| --- | --- | --- |
| Identity Law | x+0=x | x\*1=x |
| Negation Law | **x+x’=1** | **x\*x’=0** |
| Idempotent Law | **x+x=x** | **x\*x=x** |
| Domination Law | **x+1=1** | **x\*0=0** |
| Double Negation Law | **(x’)’=x** | |
| Commutative Law | **x+y=y+x** | **xy=yx** |
| Associative Law | **x+(y+z)=(x+y)+z** | **x(yz)=(xy)z** |
| Distributive Law | **x(y+z)=xy+xz** | **x+yz=(x+y)(x+z)** |
| De-Morgan’s Law | **(x+y)’=x’y’** | **(xy)’=x’+y’** |
| Absorption Law | **x+xy=x(1+y)=x\*1=x** | **x(x+y)=x** |

논리회로의 간소화는 디지털 회로 설계에서 중요한 단계 중 하나로, 회로의 복잡성을 줄이고 효율성을 높이기 위해 사용된다. 간소화된 회로는 일반적으로 더 적은 수의 논리 게이트를 사용하며, 이는 제조 비용을 줄이고, 속도를 향상시키며, 전력 소모를 감소시키는 효과가 있다. 위의 표는 부울 대수의 기본 공리이다. 논리회로의 간소화를 위해서는 기본적으로 위 표에 나온 법칙을 이용하여 간소화 시킨다. 예를 들어 A’B + ABC’ + ABC라는 식을 간소화 시키면 다음과 같다. A’B+ABC’+ABC = A’B+AB(C+C’) = A’B+AB = (A+A’)B = B 이 과정에서 Associative law, Negation law등의 기본 법칙이 사용되었다.

이러한 기본적인 방법 이외에도 간소화에는 여러 방법이 존재한다. 대표적으로 카르노 맵과 Quine-McCluskey 방법이 있다.

**3.카르노 맵에 대해 조사하시오(예시 포함).**

카르노 맵은 논리식을 시각적으로 간소화하는 데 사용되는 방법으로, 2차원 그리드 형태로 표현된다. 각 셀은 입력 변수의 특정 조합을 나타내며, 1 또는 0 값을 할당하여 출력을 표시한다. 카르노 맵을 사용하여 인접한 셀들을 그룹화함으로써, 논리식을 간단한 형태로 나타낼 수 있다. 맵의 크기는 입력 변수에 수에 따라 결정되며, 변수가 n개일 때 개의 셀을 생성한다. 카르노 맵은 변수의 배치 – 값의 할당 – 그룹화 – 논리식 도출의 순서르 간소화를 진행한다.

변수의 배치 : 변수는 그레이 코드 순서로 맵의 가장자리에 배치되어, 인접한 셀 간에는 단 하나의 변수만이 변하도록 한다.

값의 할당 : 각 셀에는 논리 함수의 출력값(보통 1 또는 0)을 할당한다.

그룹화 : 하나 또는 그 이상의 1이 포함된 셀을 최대한 큰 직사각형으로 그룹화한다. 각 그룹은 의 셀 수를 포함해야 하며, n은 정수이다. 그룹화를 통해 논리식을 간소화할 수 있다.

논리식 도출 : 그룹에서 변수의 공통된 형태를 식별하여 논리식을 도출한다. 존재하지 않는 변수는 그룹화 과정에서 제거된다.

위의 과정을 바탕으로 를 간소화 해보자.

먼저 3개의 변수가 있으므로 카르노 맵은 의 셀이 있는 2\*4 또는 4\*2의 맵을 갖는다. 이후 논리식에 따라 각 셀에 값을 할당하면 아래와 같다.

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

이 경우 1의 값을 갖는 셀을 그룹화하면 빨간색과 노란색, 파란색 세 그룹을 형성할 수 있다. 카르노 맵을 그룹화 할 때는 2의 제곱수 단위로 묶어야 하므로 3 그룹이 형성된 것이다. 이를 바탕으로 논리식을 도출하면 다음과 같다. AB + BC + CA임을 알 수 있다.

**4.Quine-McCluskey 최소화 알고리즘에 대해 조사하시오.**

Quine-McCluskey 최소화 알고리즘은 디지털 논리 회로를 최소화하기 위해 개발된 방법 중 하나로, 논리 함수의 가장 간결한 형태를 찾는 데 사용된다. 이 알고리즘은 카르노 맵과 유사한 결과를 제공하지만, 주로 더 많은 변수를 가진 복잡한 함수에 적합하며, 컴퓨터를 통한 자동 처리에 용이하다. Quine-McCluskey 알고리즘의 기본 원리와 단계는 다음과 같다.

표준형 생성 : 주어진 논리 함수의 모든 minterm을 나열하고, 이를 이진수로 표현한다.

그룹화 : 이진수 표현에서 1의 개수에 따라 minterm들을 그룹화한다. 각 그룹은 1의 개수가 같은 minterm들로 구성된다.

minterm 비교 및 단순화 : 인접한 그룹 간의 minterm들을 비교하여, 단 하나의 비트만이 다른 minterm 쌍을 찾는다. 이러한 minterm 쌍은 하나의 비트 차이를 제외하고 동일하므로, 이 비트를 '-'(동시에 0 또는 1일 수 있음)로 표시하여 단순화할 수 있다.

불필요한 minterm 제거 : 모든 가능한 단순화가 수행된 후, 필수 minterm(cover 필수적인 경우에만 1을 출력하는 minterm)을 식별하고 선택한다. 필수 민텀은 결과 식에서 반드시 포함되어야 한다.

최종 논리식 도출 : 필수 minterm과 함께 선택적으로 추가된 minterm들을 합하여 최소화된 논리식을 구성한다.

이러한 과정의 Quine-McCluskey 최소화 알고리즘은 카르노맵 방법보다 더 체계적이고, 변수가 많을 때 유리하다. 또한 컴퓨터 프로그램을 통해 자동으로 실행시키는 자동화에 적합하다. 하지만 변수의 수가 증가함에 따라 처리해야 할 minterm의 수가 기하급수적으로 증가하기 때문에, 매우 복잡한 함수에 대해서는 계산 시간과 메모리 요구량이 크게 증가한다.

**5.기타이론.**

카르노 맵, Quine-McCluskey 방식 이외에도 간소화 하는 방법에는 여러 방법들이 있다.

그중 이진 결정 다이어그램(Binary Decision Diagrams, BDDs)는 복잡한 논리 시스템을 표현하고 조작하기 위한 데이터 구조이다. 주로 하드웨어 검증, 논리 회로 설계 최적화, 함수의 논리적 속성 분석 등에 활용된다. BDD는 논리 함수를 이진 트리 형태로 표현한다. 이 트리는 결정 노드와 두 개의 터미널 노드(0과 1)로 구성된다. 각 결정 노드는 하나의 입력 변수를 나타내며, 두 개의 자식 노드(하나는 변수가 참일 때를, 다른 하나는 거짓일 때를 나타냄)를 가진다. 함수의 값을 결정하기 위해, 노드에서 정의된 변수의 값을 따라 트리를 순회한다.

BDD는 일반적으로 정규화된 형태로 표현된다. 이는 모든 가능한 입력에 대해 동일한 함수가 동일한 BDD로 표현된다는 것을 의미한다. BDD는 종종 감소된 형태로 사용되는데 중복되는 서브트리가 제거되고, 동일한 테스트 결과를 갖는 노드가 합쳐진다. 이 감소 과정을 통해 BDD는 함수를 가능한 한 간결하게 표현할 수 있다.

BDD는 많은 경우에 매우 효율적이나, 일부 함수는 BDD 표현이 지나치게 크게 되어 메모리 사용량과 처리 시간 측면에서 비효율적일 수 있다.

도표, 스케치, 화이트, 라인이(가) 표시된 사진

자동 생성된 설명스케치, 그림, 도표, 화이트이(가) 표시된 사진

자동 생성된 설명