SOGANG UNIVERSITY

컴퓨터공학실험II

논리 및 연산 회로

Be as proud of Sogang
As Sogang is proud of you

실험 목표

- Adder(가산기)/ Subtractor(감산기) 의 개념 이해
- Code converter(부호 변환기)의 개념 이해
- Verilog를 사용하여 다양한 Adder 및 Subtractor 구현
- Verilog를 사용하여 다양한 Code converter 구현
- FPGA 통해서 Verilog로 구현된 회로의 동작 확인

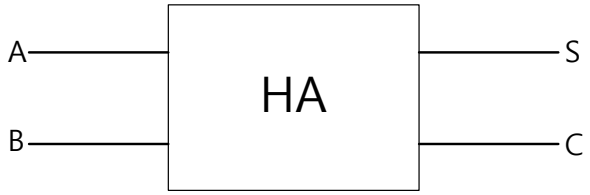
Half Adder / Full Adder

- Half Adder(HA)는 2개의 입력과 출력을 가진다.
- 입력은 1-bit(A,B) 수 2개, 출력은 Sum(S)과 Carry(C)로 구성된다.
- 2개의 1-bit 수를 더해서 1-bit로 표현할 수 있는 수보다 클 때, Carry가 생성된다.

$$S = \overline{A}B + A\overline{B} = A \oplus B$$

$$C = AB$$

Half Adder Boolean Function



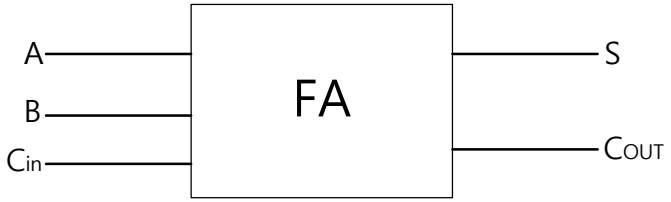
Half Adder

- Full Adder(FA)는 Carry 또한 더할 수 있는 가산기로서 실질적인 기초 연산회로로 사용된다.

$$S = (A \oplus B) \oplus C_{in}$$

$$C_{out} = C_{in}(A \oplus B) + AB$$

Full Adder Boolean Function



Full Adder

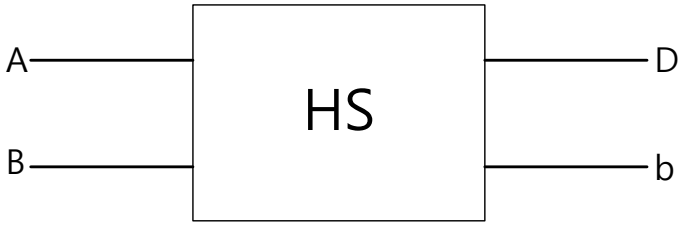
Subtractor

- Subtractor는 Adder와 반대로 1-bit 수의 뺄셈을 위한 논리회로이다.
- Half Subtractor(HS)는 2개의 1-bit 입력 A, B와 출력 A-B의 결과를 의미하는 Difference(D)와 Borrow(B)로 구성된다.
- 한 자리보다 더 큰 숫자의 뺄셈을 한다면 앞의 수에서 Borrow를 가져온다.

$$D = \overline{A} \cdot B + A \cdot \overline{B} = A \oplus B$$

$$b = \overline{A} \cdot B$$

Half Subtractor Boolean Function



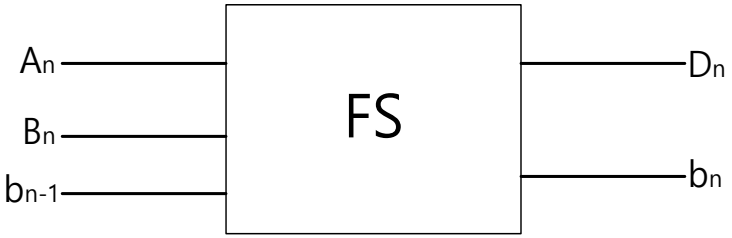
Half Subtractor

- Full Subtractor(FS)는 Borrow도 입력이 되어 완전한 Subtractor의 기능을 갖춘다.

$$D_n = A_n \oplus B_n \oplus b_{n-1}$$

$$b_n = \overline{(A_n \oplus B_n)} \cdot b_{n-1} + \overline{A_n} \cdot B_n$$

Full Subtractor Boolean Function



Full Subtractor

Logic Circuit Design Procedure

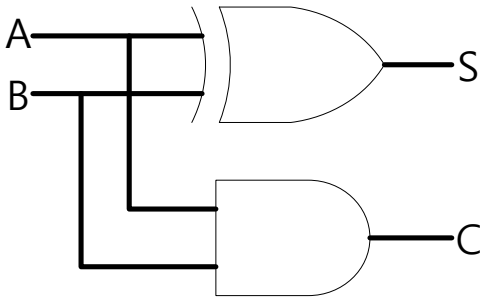
◆ Design Procedure

1. 설계하기 위한 회로의 구조와 동작을 고려하여 Truth Table로 작성한다.
2. Truth Table의 내용을 Karnaugh Map(K-map)으로 변환한다.
3. 작성된 K-map의 Minimization(POS, SOP)를 거쳐 최소화된 형태의 Boolean 함수를 작성한다.
4. 최대한 NAND 또는 NOR 게이트를 사용하여 구성한다.
5. 구성 및 실험 결과가 Truth Table과 동일한지 확인한다.

◆ Code Converter : 입력으로 들어온 특정 부호를 정해진 방식으로 변환된 부호로 출력으로 만들어주는 논리 회로

Half Adder(실습)

- Half Adder
  - (A)의 구조로 회로를 Verilog 코딩
  - Verilog의 simulation 결과를 통해 (B)의 Truth table 완성
  - 이론의 Boolean 함수와 일치하는 지 확인
  - FPGA를 통하여 동작 확인



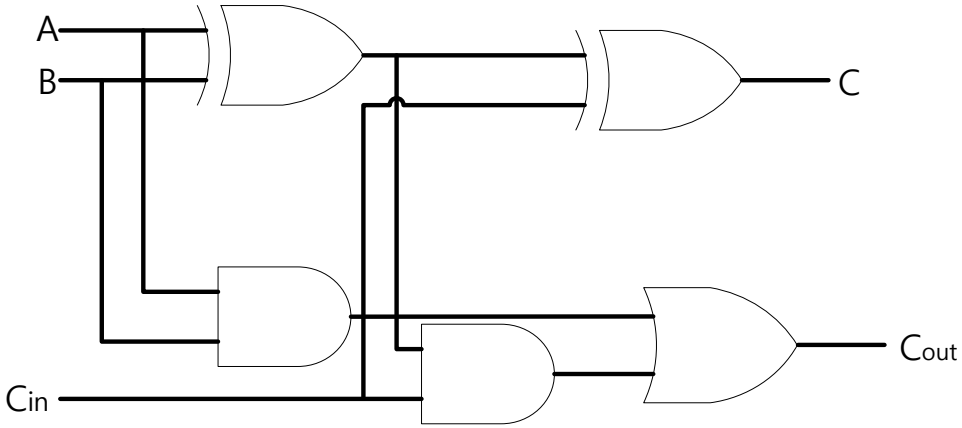
(A)

Input		Output	
A	B	S	C
0	0		
0	1		
1	0		
1	1		

(B)

Full Adder(실습)

- Full Adder
  - (A)의 구조로 회로를 Verilog 코딩
  - Verilog의 simulation 결과를 통해 (B)의 Truth table 완성
  - 이론의 Boolean 함수와 일치하는 지 확인
  - FPGA를 통하여 동작 확인



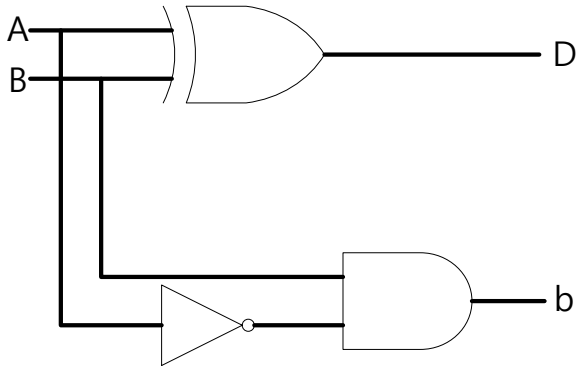
(A)

Input			Output	
C	B	A	S	C
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

(B)

Half Subtractor(실습)

- Half Subtractor
  - (A)의 구조로 회로를 Verilog 코딩
  - Verilog의 simulation 결과를 통해 (B)의 Truth table 완성
  - 이론의 Boolean 함수와 일치하는 지 확인
  - FPGA를 통하여 동작 확인



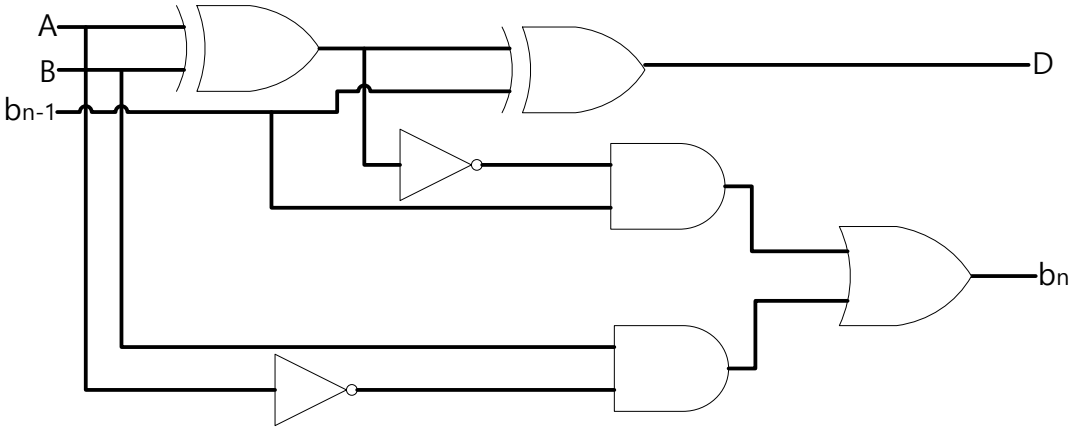
(A)

Input		Output	
A	B	b	D
0	0		
0	1		
1	0		
1	1		

(B)

Full Subtractor(실습)

- Full Subtractor
  - (A)의 구조로 회로를 Verilog 코딩
  - Verilog의 simulation 결과를 통해 (B)의 Truth table 완성
  - 이론의 Boolean 함수와 일치하는 지 확인
  - FPGA를 통하여 동작 확인



(A)

Input			Output	
A_n	B_n	B_{n-1}	b_n	D_n
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

(B)

Code Converter(실습)

- 8421(BCD)-2421 Code converter
  - (A)의 표를 이용하여 Truth Table 작성
  - 작성한 Truth Table을 이용하여 K-map 작성(4개)
  - K-map을 minimazation하여 Boolean 함수 작성(SOP form, POS form)
  - NAND 와 NOR를 활용한 형태로 각각 구성
  - Verilog로 NAND형태의 8421-2421 converter 구현
  - FPGA로 시뮬레이션 및 동작 확인

Decimal	8421 BCD	2421 BCD
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	1011
6	0110	1100
7	0111	1101
8	1000	1110
9	1001	1111

(A)