컴퓨터 공학 기초 실험2 보고서

실험제목: Ripple Carry Adder(RCA)

실험일자: 2019년 08월 27일 (화)

제출일자: 2019년 09월 09일 (월)

학　　과: 컴퓨터정보공학부

담당교수: 이준환 교수님

실습분반: 화요일 0, 1, 2

학　　번: 2015722031

성　　명: 박 태 성

1. 제목 및 목적

A. 제목

Ripple Carry Adder(RCA)

B. 목적

컴퓨터의 명령어는 여러 비트의 2진수로 표현한다. 컴퓨터는 2진수 연산을 수행하여 원하는 결과를 얻는다. RCA의 원리를 이해하고 직접 구현해 봄으로서 가산기의 원리를 이해하는데 목적을 둔다.

2. 원리(배경지식)

A. 2의 보수

가장 널리 이용되는 2진수를 표현법에는 unsigned number, singed number, 그리고 two's complement numbers가 있다. Most Significant Bit(MSB)가 부호, 나머지 bit들이 magnitude number가 된다. Unsinged number는 MSB를 부호비트로 사용하지 않고 magnitude number로 사용한다. N비트에서 [0, 2^n-1]까지 표현할 수 있다. Signed number는 MSB가 1이면 양수, 0이면 음수를 뜻한다. Magnitude number의 값에 부호를 붙여주어 10진수로 변환할 수 있다. Unsigned number는 n비트에서 [-2^(n-1)+1, 2^(n-1)-1]을 표현한다. Unsigned number는 이해가 쉬우나 덧셈이 했을 때 틀린 결과값이 나온다는 단점이 있다. 이는 MSB를 제외한 bit들의 덧셈 중에 carry가 MSB로 전달되는 경우 발생한다. 이를 컴퓨터 용어로 overflow라고 부른다. 한 가지 단점이 더 있다. 0을 표현하는 수가 2개로 중복된다. 0을 0000과 1000으로 표현할 수 있다. 위의 단점을 극복하기 위하여 two's complement가 등장한다. MSB, magnitude number의 계산은 signed number와 동일하다. N비트에서 [-2^(n-1), 2^(n-1)-1]을 표현한다. 가장 작은 음수는 $100\ldots0_2$, 가장 큰 양수는 $011\ldots1_2$이다. Two's complement는 2^N에서 binary number를 뺀 결과값을 의미한다. Two's complement를 구하는 간단한 공식이 있다. 먼저, 각 bit를 반전시킨다. 그러면, 1을 더한다. 가장 흔히 사용되는 방법이다.

B. Half Adder(HA)

Half Adder는 2개의 1비트 입력을 받아 sum값과 carry를 출력하는 가산기이다. HA는 1비트 연산 밖에 수행할 수 없다. 그 이유는 이전 비트에서 carry가 발생하면 반영이 되지 않아 2비트 이상에서는 가산기로써의 역할을 수행할 수 없다. 다중 비트를 연산하기 위해 Full Adder가 등장한다.

C. Full Adder(FA)

FA는 2개의 1비트 입력과 1개의 1비트 carry를 입력으로 받아 sum값과 carry를 출력하는 가산기이다. 이전 비트에서 carry를 입력으로 받고 carry를 출력한다. 따라서, FA를 여러 개 연결하면 다중 비트 가산이 가능하다. FA를 4개 연결한 것을 FA는 HA 2개와 Or gate로 구성할 수 있다. 또 다른 방법으로 기본 gate들만을 이용하여 설계가 가능하다.

D. RCA

RCA의 ripple은 이전 비트의 carry가 계산되어 입력으로 넘어오면 다음 비트가 계산된다는 모양이 마치 파도치는 것과 같다하여 붙여진 이름이다. 속도가 느려 단점으로 꼽힌다. 느린 속도를 극복한 Carry Look-ahead Adder가 있다.

3. 설계 세부사항

A. Half Adder

i. Functional Description

Half Adder는 입력으로 2개의 1-bit을 받아 sum과 carry out을 출력한다. Sum은 두 입력의 XOR gate로 carry는 And gate으로 표현한다.

ii. Truth Table

input		output	
a	**b**	**co**	**s**
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

iii. Karnaugh Map & Boolean Equation

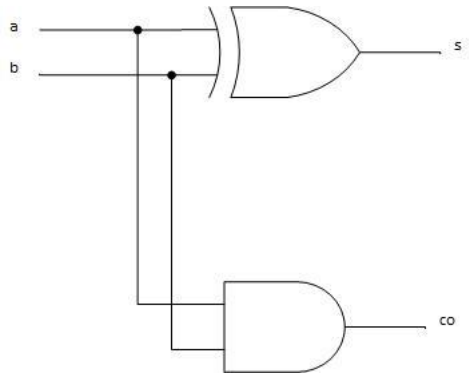
a \ b	0	1
0	0	0
1	0	1

co = a & b

a \ b	0	1
0	0	1
1	1	0

$s = a \oplus b$

iv. Logic Circuit



B. Full Adder

i. Functional Description

Full Adder는 입력으로 2개의 1-bit과 1개의 1-bit carry in을 받아서 sum과 carry out을 출력한다. Bit들간의 자리올림이 불가능한 Half Adder의 한계를 극복한 가산기이다. 출력은 Half Adder와 같지만 입력이 carry in을 포함하여 3개이다.

ii. Truth Table

input			output	
ci	**a**	**b**	**co**	**s**
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

iii. Karnaugh Map & Boolean Equation

ci \ ab	00	01	11	10
0	0	1	0	1
1	1	0	1	0

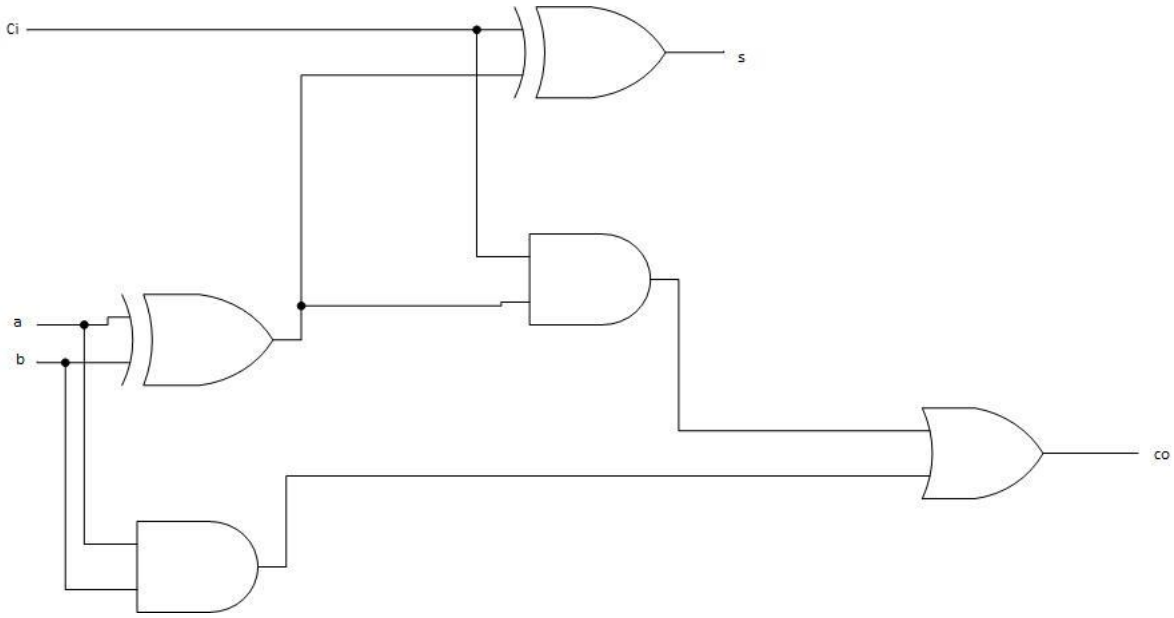
$$s = a \oplus b \oplus ci$$

ci \ ab	00	01	11	10
0	0	0	1	0

1	0	1	1	1

co = ab + aci + bci

iv. Logic Circuit



And, or과 같은 기본 gate들로 구성한 logic circuit이다. Verilog HDL로 코딩할 때는 Half Adder 2개와 or gate로 구성하였다.

C. 4-bit RCA

i. Functional Description

Full Adder의 출력 carry를 다음 bit의 입력 carry로 전달되게 하여 총 4개의 Full Adder를 연결하면 된다. 결과 값은 1-bit full adder의 carry delay를 모두 더한 시간이다. 따라서 연산 속도가 느리다.

ii. I/O Description(including wire)

Port	Name	Bandwidth	Description
Input	a	4bit	Input data A
	b	4bit	Input data A
	Ci	1bit	Carry in
Output	co	1bit	Carry out
	s	4bit	Sum
Wire	c	3	Internal Carry

iii. Module Description

Classification	Name	Decription
Module	rca	4-bit ripple carry adder
Instance	U0_fa	Full adder
	U1_fa	
	U2_fa	
	U3-fa	

4. 설계 검증 및 실험 결과

A. 시뮬레이션 결과

i. Half Adder

1. Testbench

```
begin
a=1'b0; b=1'b0; // s = 0, co = 0
#10; a=1'b1; b=1'b0; // s = 1, co = 0
#10; a=1'b0; b=1'b1; // s = 1, co = 0
#10; a=1'b1; b=1'b1; // s = 0, co = 1
#10; $stop;
end
```

Exhaustive verification을 통하여 검증하였다. 입력으로 발생 가능한 모든 경우를 주어 오류가 발생하지 않는지 검증하였다. 1비트 2개의 입력은 4가지 경우가 발생 가능하다.

2. Wave Form

S와 co가 예측값과 동일한 것을 확인하였다. Boolean Equation으로 계산한 예측값과 wave form에서 보여주는 값이 동일하다. Carry도 정상적으로 출력된다.

ii. Full Adder

1. Testbench

```
begin
a = 1'b0; b = 1'b0; ci = 1'b0; // s = 0, co = 0
#10; a = 1'b0; b = 1'b0; ci = 1'b1; // s = 1, co = 0
#10; a = 1'b0; b = 1'b1; ci = 1'b0; // s = 0, co = 1
#10; a = 1'b0; b = 1'b1; ci = 1'b1; // s = 0, co = 1
#10; a = 1'b1; b = 1'b0; ci = 1'b0; // s = 1, co = 0
#10; a = 1'b1; b = 1'b0; ci = 1'b1; // s = 0, co = 1
#10; a = 1'b1; b = 1'b1; ci = 1'b0; // s = 0, co = 1
#10; a = 1'b1; b = 1'b1; ci = 1'b1; // s = 1, co = 1
#10; $stop;
end
```

Exhaustive verification을 통하여 검증하였다. 입력으로 발생가능한 모든 경우의 수를 검증하였다. 1비트 3개의 입력은 8가지 경우가 발생 가능하다. 모든 경우를 검증하여 오류가 없음을 확인한다.

2. Wave Form

S와 co가 예측값과 동일한 것을 확인하였다. Ci가 0일 때와 1일 때 모두 정상적으로 s와 co를 출력한다.

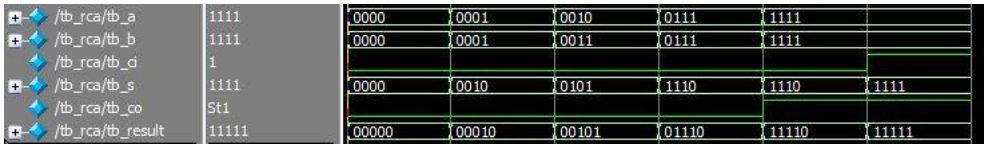
iii. 4-bit RCA

1. Testbench

```
begin
tb_a = 4'b0; tb_b = 4'b0; tb_ci = 0; // tb_s = 4'b0000, tb_co = 0
#10; tb_a = 4'b0001; tb_b = 4'b0001; // tb_s = 4'b0010, tb_co = 0
#10; tb_a = 4'b0010; tb_b = 4'b0011; // tb_s = 4'b0101, tb_co = 0
#10; tb_a = 4'b0111; tb_b = 4'b0111; // tb_s = 4'b1110, tb_co = 0
#10; tb_a = 4'b1111; tb_b = 4'b1111; // tb_s = 4'b1110, tb_co = 1
#10; tb_ci = 1; // tb_s = 4'b1111, tb_co = 1
#10; $stop;
end
```

Directed verification을 통하여 검증하였다. 4비트 2개의 입력은 발생 가능한 경우가 너무 많다. 따라서 표본을 뽑아내서 검증을 진행한다. 6개의 표본을 뽑아 검증하였다.

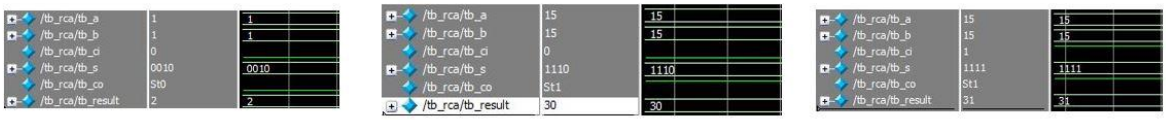
2. Wave Form



Tb_s, tb_co, 그리고 tb_result가 예측값과 동일한 것을 확인하였다. Tb_result를 출력하니 결과값을 한 눈에 볼 수 있어 편리하다.

Decimal

Unsigned

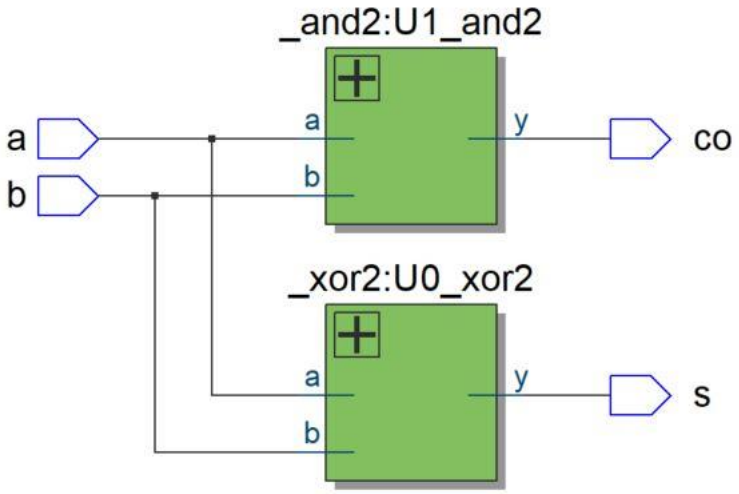


Radix에서 decimal, unsigned로 설정했을 때의 wave form이다. Decimal일 때는 two's complement의 정의를 감안하여 10진수로 변환해보면 결과값이 정확히 출력된다. Unsinged일 때는 MSB도 magnitude number 계산에 포함시켜야 함을 감안하고 10진수로 계산하면 결과값이 정확히 출력된다는 것을 확인했다.

B. 합성(synthesis) 결과

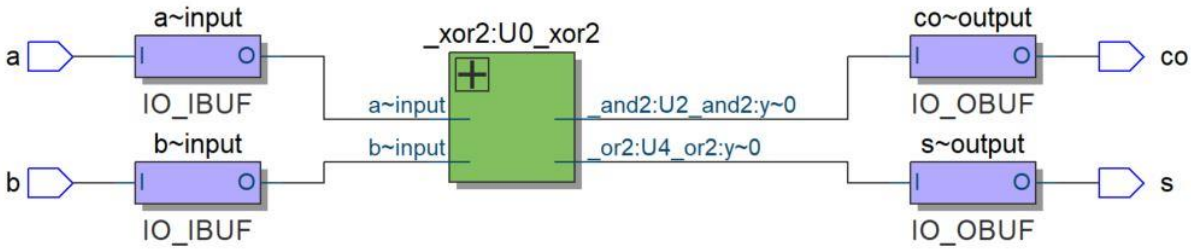
i. Half Adder

1. Register Transfer Level(RTL) Viewer



Boolean Equation에 맞게 설계한 half adder이다.

2. Technology Map Viewer



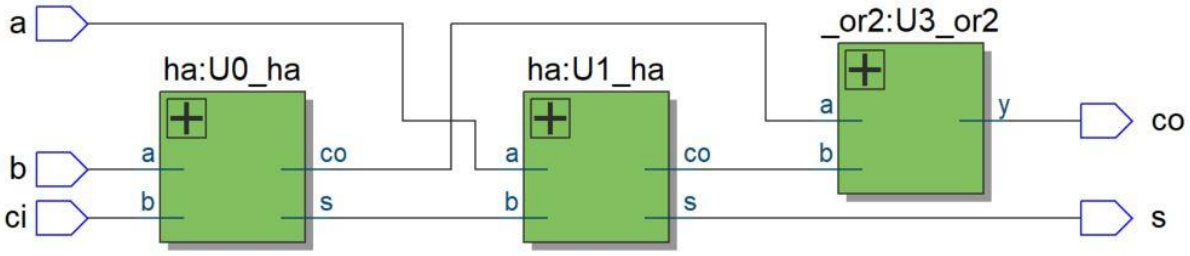
3. Flow Summary

Flow Summary	
Flow Status	Successful - Wed Aug 28 21:47:40 2019
Quartus Prime Version	15.1.0 Build 185 10/21/2015 SJ Lite Edition
Revision Name	RCA
Top-level Entity Name	ha
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	4
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

학교에서 제공하는 보드에 테스트 할 수 있도록 Cyclone V 5CSXFC6D6F31C7를 프로젝트 생성할 때 설정하였다. Flow Summary에서 확인하였다.

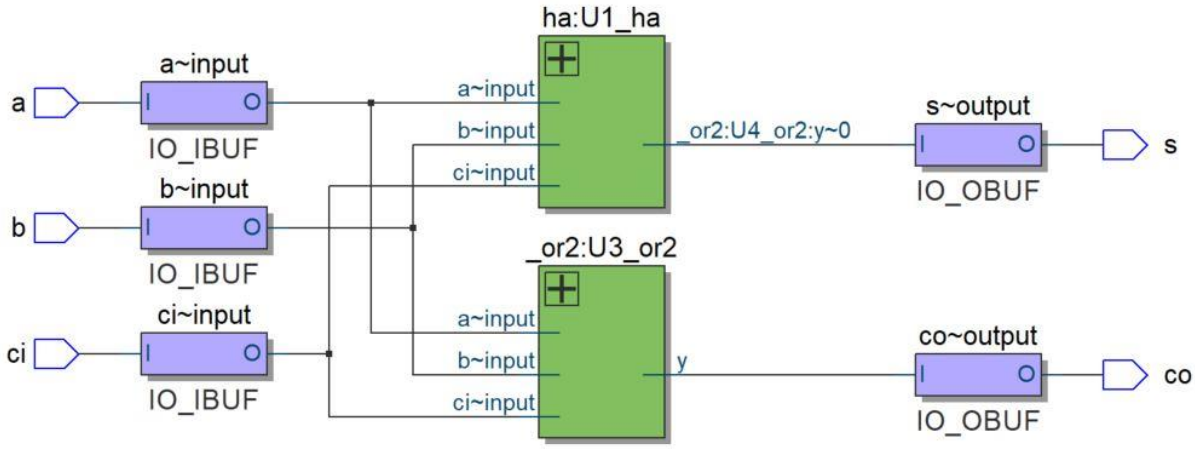
ii. Full Adder

1. Register Transfer Level(RTL) Viewer



Half Adder 2개와 or gate로 설계한 full adder이다. 기본 gate들로 구성 할 수도 있고 위와 같이 구성할 수도 있다.

2. Technology Map Viewer



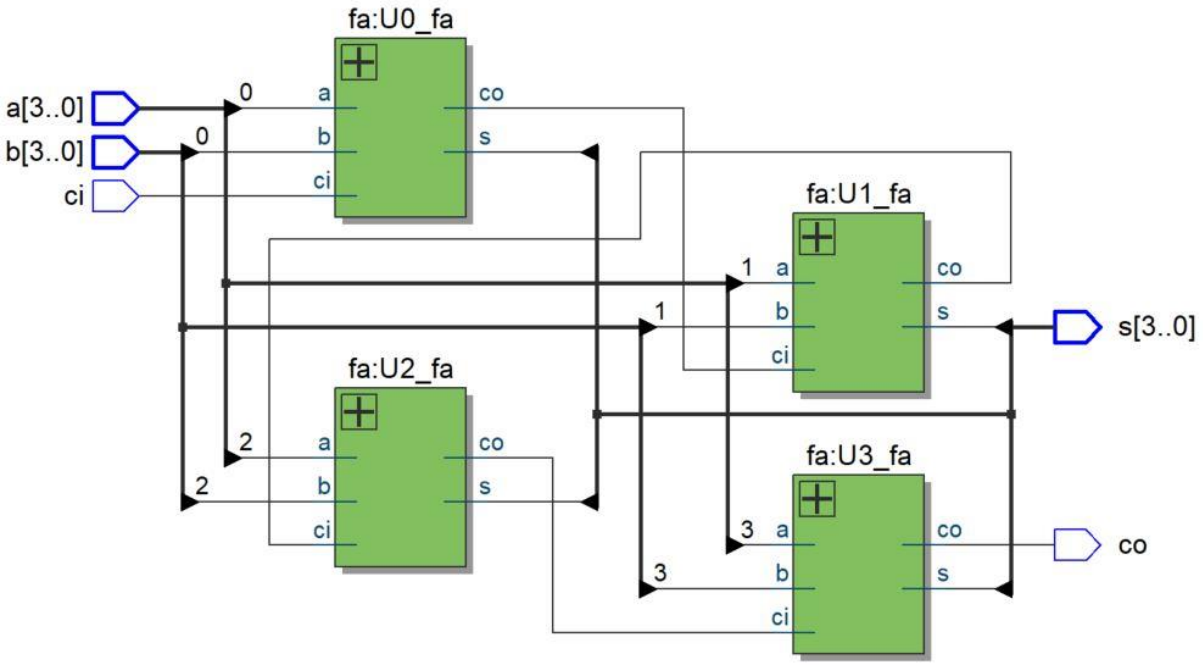
3. Flow Summary

Flow Summary	
Flow Status	Successful - Wed Aug 28 21:43:47 2019
Quartus Prime Version	15.1.0 Build 185 10/21/2015 SJ Lite Edition
Revision Name	RCA
Top-level Entity Name	fa
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	5
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

1비트 a, b, c와 1비트 ci, co total pins는 5개이다.

iii. 4-bit RCA

1. Register Transfer Level(RTL) Viewer



4비트 2개가 1비트씩 full adder에 입력으로 들어가는 것을 확인하였다. 4비트 s와 1비트 co이 출력되는 것을 확인하였다.

2. Flow Summary

Flow Summary	
Flow Status	Successful - Wed Aug 28 20:49:42 2019
Quartus Prime Version	15.1.0 Build 185 10/21/2015 SJ Lite Edition
Revision Name	RCA
Top-level Entity Name	rca
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	14
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

4비트 a, b, c와 1비트 co, ci를 포함하여 total pins이 14가 된다.

5. 고찰 및 결론

A. 고찰

Half Adder testbench 검증시 wave form이 예상한 것과 다르게 나왔다. Co는 정상적으로 출력되는데 s가 정상적으로 출력되지 않았다. 따라서 s를 구성하는 port에 문제가 있다고 생각하여 코드를 다시 보았다. 입력이 잘못들어가서 발생한 문제였다. Wave form은 디버깅에 유용한 tool이다.

B. 결론

당연해서 모르고 있던 사실이 있다. Verilog HDL을 포함한 소프트웨어, 하드웨어는 two's complement를 notation으로 사용하고 있다. Radix에 signed number로 수를 변환하여 보면서 느꼈다. 어찌보면 당연한 것이 unsigned number를 notation으로 사용하면 덧셈에 문제가 발생한다.

4-bit RCA를 구현하였다. 2개를 연결하면 1byte RCA를 만들 수 있다. 1byte RCA 4개를 연결한다면 32-bit RCA를 설계할 수 있다. 또 다른 방법으로는 full adder 32개를 연결하여 설계할 수 있다.

6. 참고문헌

David Money Harris & Sarah L. Harris / Digital Design and Computer Architecture / Elsevier Korea L.L.C / November 2013