디지털 논리2 및

컴퓨터 공학 기초 실험2   
보고서

과제제목: Ripple Carry Adder

실험일자: 2017년 09월 01일 (금)

제출일자: 2017년 09월 07일 (목)

학 과: 컴퓨터공학과

담당교수: 공진흥 교수님

실습분반: 금요일 5, 6, 7

학 번: 2012722028

성 명: 장한별

1. 제목 및 목적
   1. 제목

Ripple carry adder

* 1. 목적

각종 논리게이트를 이용해 Half Adder 를 설계하고, 이를 이용해 Full Adder 를 설계한다. 그 후 Full Adder를 통해 Ripple Carry Adder 의 작동원리와 구성을 이해하고 이를 FPGA board에 연결하여 동작을 검증한다.

1. 원리(배경지식)

1) 부호가 없는 수(unsigned number)

Sign Number 로 수식을 계산한다고 가정하자. 예를 들어 4-bit 표현으로 5 = 01012 이지만 -5 는 11012 로 표현한다. 겉보기에 11012 은 13을 2진법으로 나타낸 것처럼 보이는 문제가 발생한다.

이처럼 한 이진법으로 나타낸 수를 보고 2가지 수를 나타내는 문제가 발생하기 전에 부호를 생략해주는 것을 Unsigned Number라 한다. 간단하게 예를 들자면, 4-bits 로는 최대 7까지밖에 나타낼 수 없어 2 + 6 을 -8 로 나타내는 것을 보호가 없는 수, 즉, Unsigned number로 표현하면 8로 나타낼 수 있다. 이처럼 Unsigned 표현방식은 Bit의 메모리로 인한 표현의 한계나 이로 인해 생기는 오차를 없애고 실질적인 계산이 용이 하다는 장점이 있다.

2) 2의 보수(two’s complement)

2의보수(two’s complement) 란 컴퓨터가 양수와 음수를 구별하기위하여 만든 방법이다.

2의보수는 구하려는 숫자의 1의 보수를 구한 다음 그 숫자에 1을 더하는 방법으로 구할 수 있다. 1의보수는 주어진 숫자 이 모든 0은 1로, 모든 1은 0으로 바꾸면 된다.

예를 들어 11002 의 2의보수를 구한다고 하면, 11002 의 1의보수는 모든 1을 0으로, 모든 0을 1로 바꾼 00112 이 된다. 따라서 11002의 1의보수는 00112이고, 00112 에 1을 더한 01002 가 11002의 2의 보수가 된다.

이러한 방식으로 2의보수를 구할 수 있다.

1. 설계 세부사항

Ripple Carry Adder (RCA)를 설계하기 위하여 먼저 기본 gate들을 설계한 뒤 반가산기(Half Adder)를 설계한다.

Assign을 통해 INVERTER gate, NAND gate, AND gate, OR gate 를 설계하고, XOR gate는 INVERTER gate, AND gate, OR gate를 Instance 하여 설계한다.

1. 반가산기(Half Adder)

Half Adder는 2개의 Input A , B를 입력(Input)하고, 그 합인 Sum과 자리 올림수인 Co(Carry out)을 출력(Output)하는 논리회로다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** | | **Output** | |
| **A** | **B** | **Sum** | **Co** |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

<Half Adder 의 진리표>

.

그 후 Half Adder의 Karnaugh Map과 Boolean Equation을 구한다.

<Sum>

|  |  |  |
| --- | --- | --- |
| **a** **b** | **0** | **1** |
| **0** | 0 | 1 |
| **1** | 1 | 0 |

Sum = ab’+a’b

=

<Co>

|  |  |  |
| --- | --- | --- |
| **a**  **b** | **0** | **1** |
| **0** | 0 | 0 |
| **1** | 0 | 1 |

.

Co = ab

1. 전가산기Full Adder

Full Adder는 2개의 Input A, B 와 하위 자리 올림수 Cin 을 입력(Input)하고,

그 합인 Sum과 자리 올림수 Co(Carry out)을 출력(Output)하는 논리회로이다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | | | **Output** | |
| **A** | **B** | **Cin** | **Sum** | **Co** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

<Full Adder의 진리표>

그 후 Full Adder의 Karnaugh Map과 Boolean Equation을 구한다.

<Sum>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cin ab | 00 | 01 | 11 | 10 |
| 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |

Sum = a’b’cin+a’bcin’+abcin+ab’cin’

=cin’(a’b+ab’)+cin (a’b’+ab)

=

=abcin

<Co>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cin ab | 00 | 01 | 11 | 10 |
| 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 |

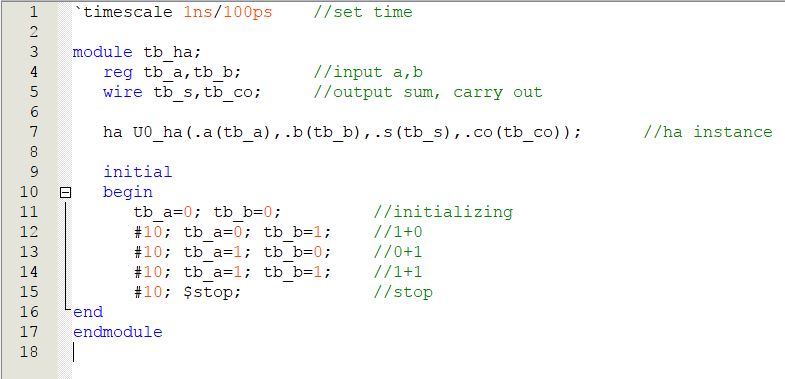
Cout = ab+bcin+cina

이렇게 구현한 Full Adder 4개를 이용하여 4-bit RCA 논리회로를 설계한다.

Input은 4비트 의 a,b 그리고 cin 으로 입력하고, Output은 4비트의 Sum과 1비트의 Co로 설정하고 설계하였다.

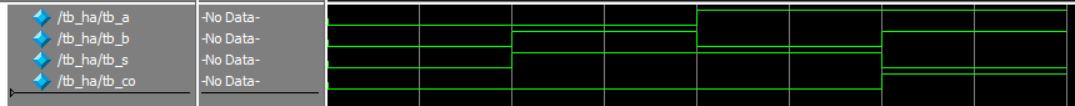
1. 설계 검증 및 실험 결과
   1. 시뮬레이션 결과

1) 반가산기(Half Adder)



<Half Adder 의 TestBench>

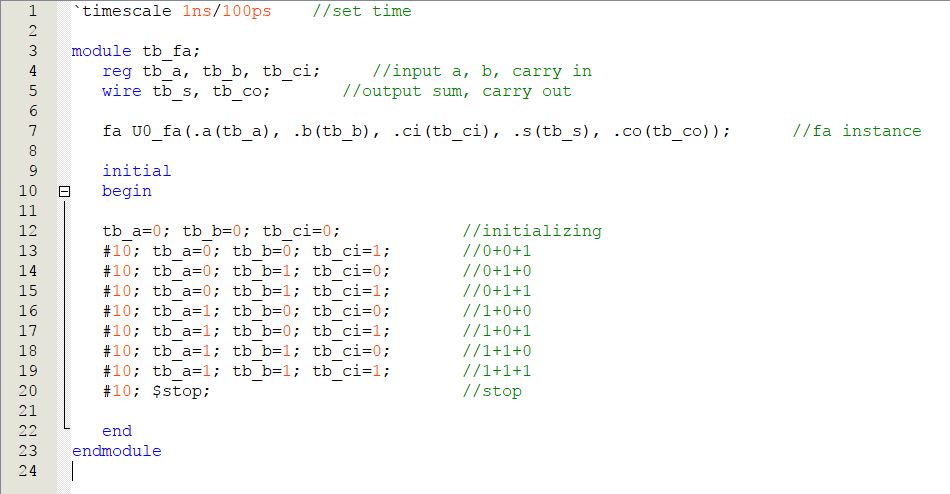
Input(a,b)에 (0,0), (0,1), (1,0), (1,1) 을 10ns 마다 대입 후, 각각의 Output s와 co을 도출한다.



<Half Adder 의 WaveForm>

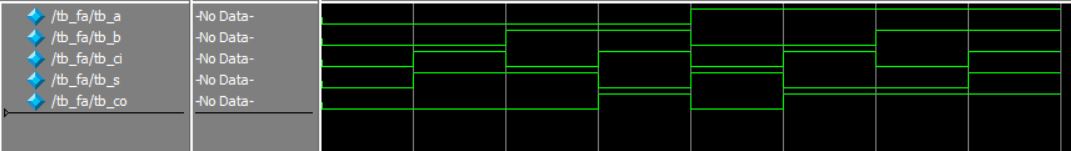
설계 세부사항에서 확인했던 Half Adder의 진리표의 내용을 WaveForm을 통해 확인 할 수 있다.

1. 전가산기(Full Adder)



<Full Adder 의 TestBench>

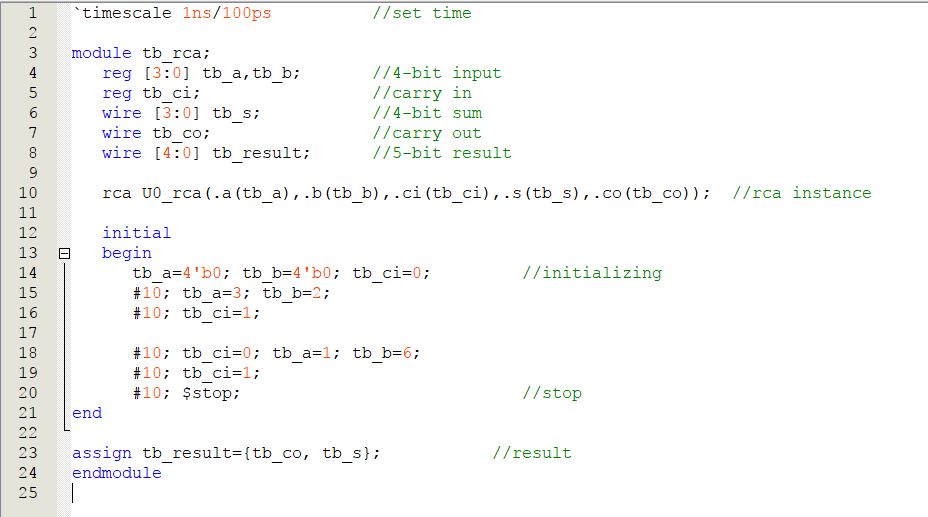
Input (a,b,ci)를 (0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1)을 10ns 마다 대입 후 Output s와 co를 도출한다.



<Full Adder 의 WaveForm>

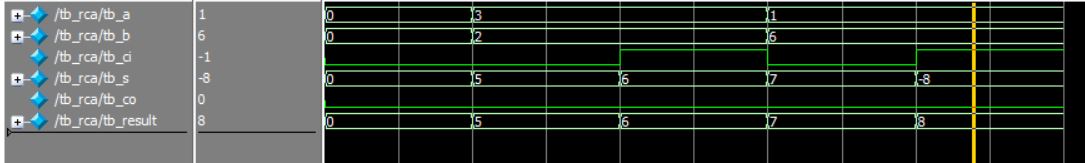
설계 세부사항에서 확인했던 Full Adder의 진리표의 내용을 WaveForm을 통해 확인 할 수 있다.

1. 4-bits Ripple Carry Adder



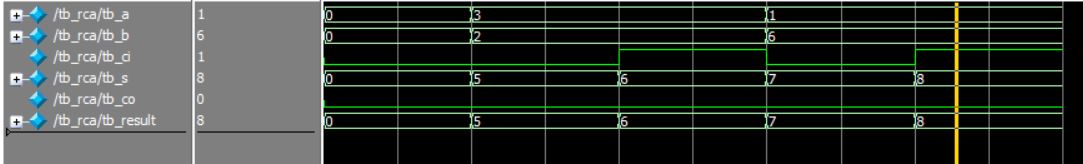
<4-bits Ripple Carry Adder 의 TestBench>

Input(a,b)에 (3,2) 과 (1,6) 을 대입 후 , 각각의 Output s와 co을 도출한다.



<4-bits Ripple Carry Adder 의 Decimal 형태의 WaveForm>

WaveForm 을 보면 문제가 없어 보이지만 Decimal 형태로 나타냈기 때문에 4-bit 에서는 최대 7까지만 나타낼 수 있기 때문에 Overflow가 발생해서 s에 -8 이 출력되었다. 이러한 문제를 해결하기위해 WaveForm을 Unsigned 형태로 나타낸다.

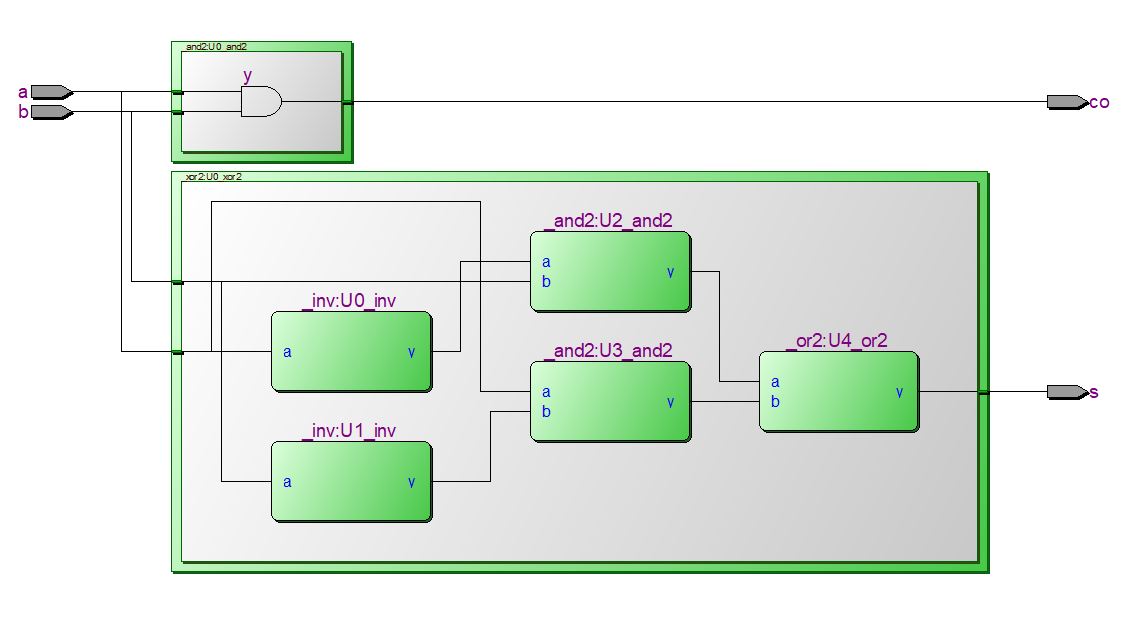


<4-bits Ripple Carry Adder 의 Unsigned 형태의 WaveForm>

Decimal 형태로 나타냈을 땐 -8 이었던 결과가 Unsigned 로 즉, 부호가 없는 형태로 8이 출력 된 것을 확인 할 수 있다.

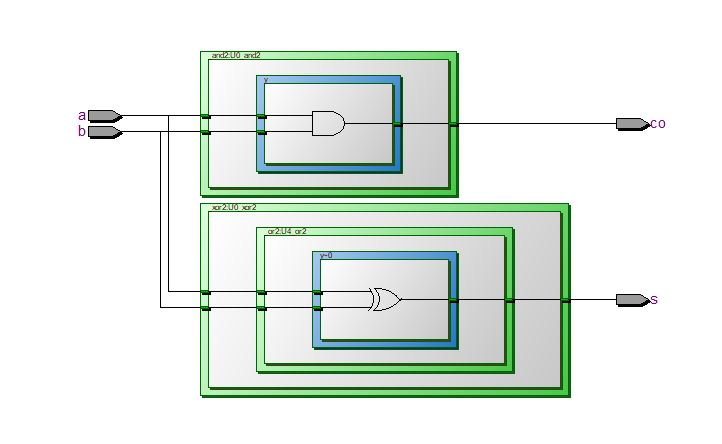
* 1. 합성(synthesis) 결과

1) 반가산기(Half Adder)

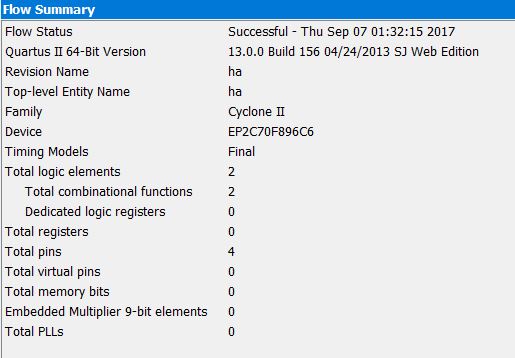


<Half Adder 의 RTL viewer>

input a, b 과 output co, s 그리고 AND Gate 와 XOR Gate 로 이루어져있는 것을 확인 할 수 있다.

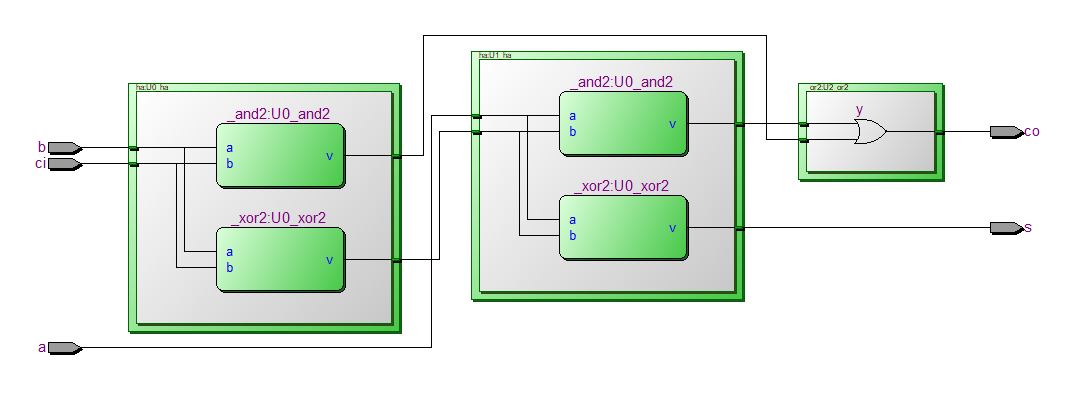


<Half Adder 의 Technology Map Viewer>



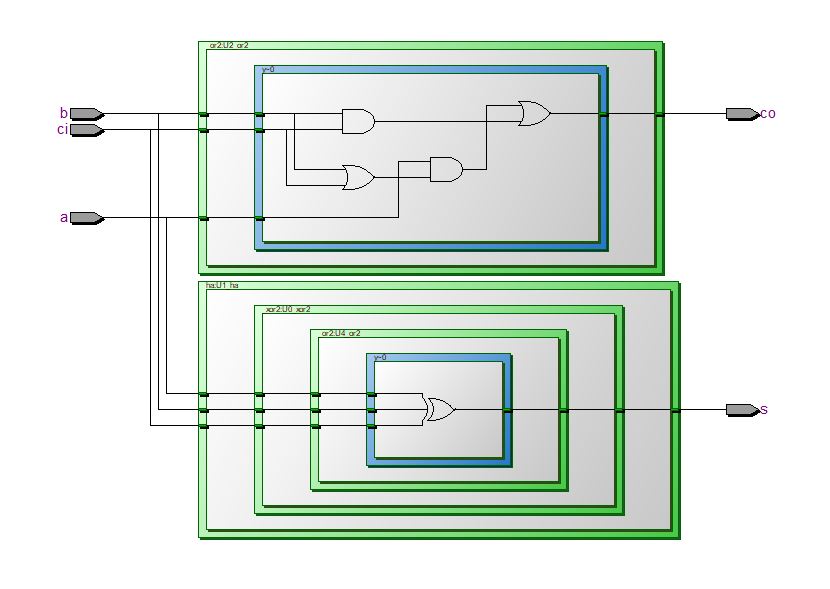
<Half Adder 의 Flow Summary>

2) 전가산기(Full Adder)

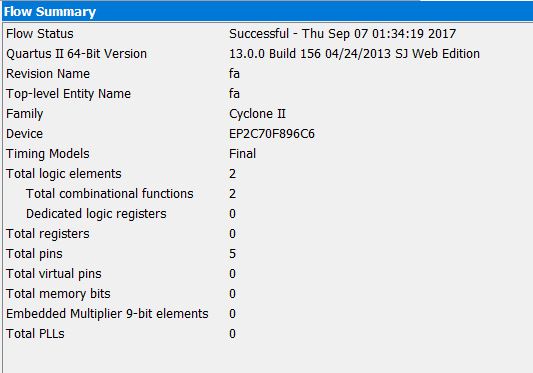


<Full Adder 의 RTL Viewer>

input a, b, ci 과 output co, s 그리고 2개의 Half Adder 와 OR Gate 로 이루어져있는 것을 확인 할 수 있다.

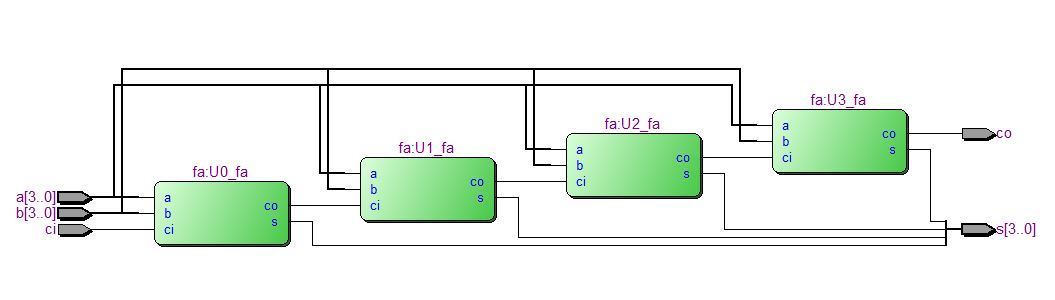


<Full Adder 의 Technology Map Viewer>



<Full Adder 의 Flow Summary>

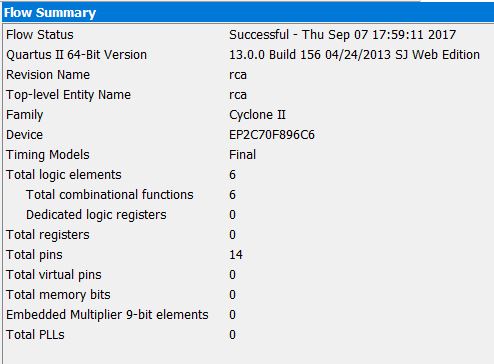
3) 4-bits Ripple Carry Adder



<4-bits Ripple Carry Adder 의 RTL Viewer>

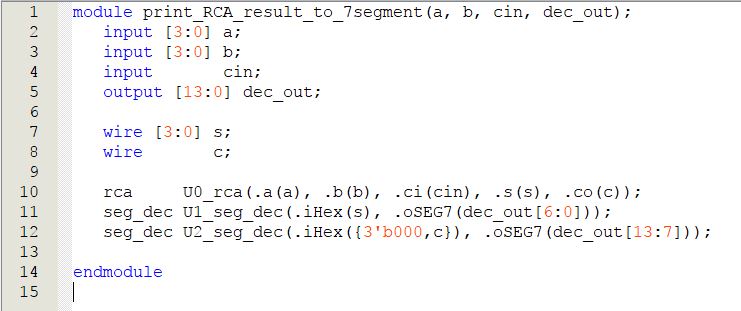
Input a, b, ci 과 output co, s 그리고 4개의 Full Adder 로 Ripple Carry Adder를 구성하고 있음을 확인 할 수 있다.

이때, input a, b 는 4비트, ci는 1비트로 설정하였고, output co는 1비트, s는 4비트로 설정하였다.



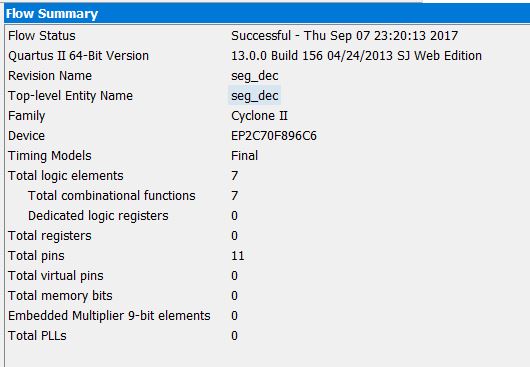
<4-bits Ripple Carry Adder 의 Flow Summary>

4) Seg\_dec module

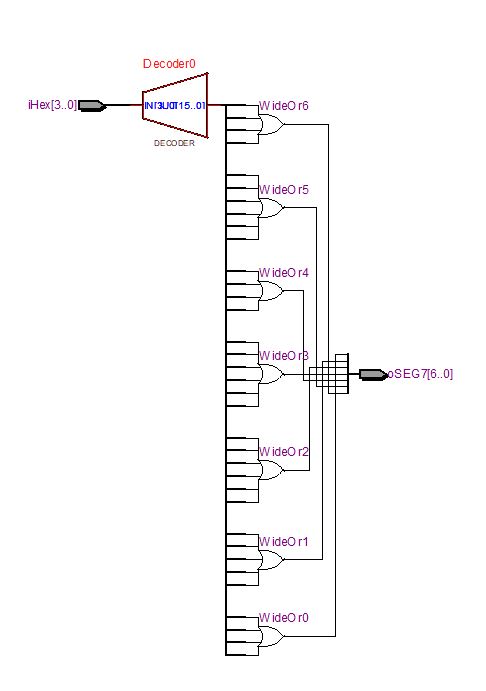


<Seg\_dec 의 Test Bench>

추가적으로 Seg\_Dec TestBench와 Flow Summary 그리고 RTL Viewer를 확인했다.

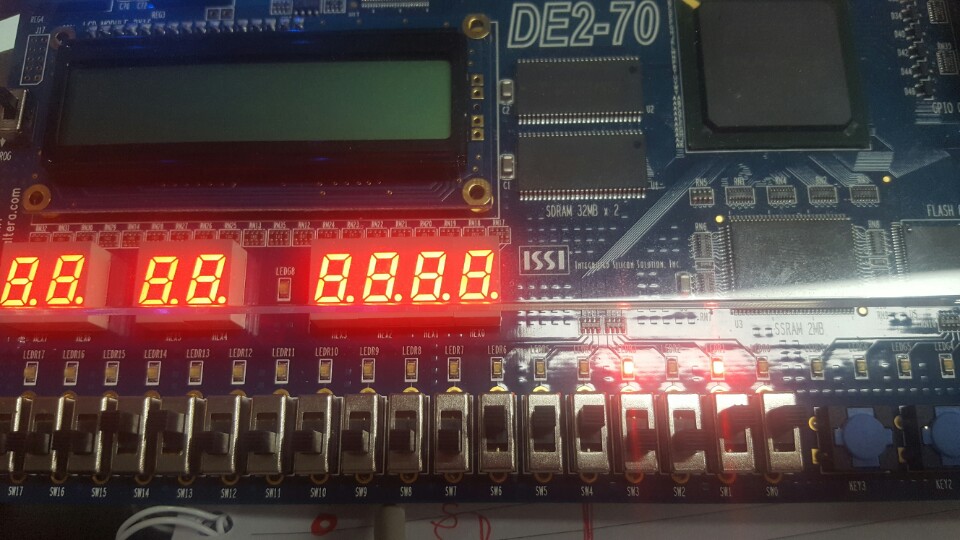


<Seg\_dec 의 Flow Summary>



<Seg\_dec 의 RTL Viewer>

* 1. FPGA board targeting 결과



Input a 에 00112 , b에 01112 을 설정하면 00112 + 01112 = 10102

이므로 오른쪽 기준으로 두번째 불과 네번째 불이 들어오는 것을 예상할 수 있다. 위 사진으로부터 그 결과를 확인 할 수 있다.

1. 고찰 및 결론
   1. 고찰

시작부터 순탄하지 않았다. Quartus II를 다운 받기부터 Project 와 Verilog HDL File을 생성하고 저장하는 과정에 한글이 포함되어 있으면 인식을 못해서 많은 애를 먹었다. 컴퓨터공학과 학생으로서 많이 접해왔던 C나 C++ 에서 간단한 논리식으로 표현 가능한 것을 TestBench 로 표현 하는 것이 어려웠다. 특히 Output과 Input을 wire로 연결해 주는 것이 익숙하지 않아서 초반에 실수가 많았다.

하지만, 직접 코드를 짜보고 RTL Viewer 와 Technology map Viewer 를 통해 어떤 구조로 이루어져 있는지, WaveForm 을 통해 어떤 출력 결과를 나타내는지 시각적으로 확인 할 수 있어서 Quartus 만의 매력을 알 수 있는 시간이었고 앞으로도 유용하게 이용할 수 있을 것 같다.

* 1. 결론

이번 실험을 통해 Half Adder는 1개의 XOR Gate 와 1개의 AND Gate로 이루어져 있고, Full Adder는 2개의 Half Adder 와 1개의 OR Gate로 이루어져 있는 것을 확인 할 수 있었다. 또한, 4-bits RCA의 구조는 4개의 Full Adder로 이루어져 있는 것을 확인할 수 있었다. 이를 통해 32-bits RCA는 8개의 4-bits RCA 로 설계 하거나 32개의 Full Adder로 만들 수 있을 것 같다.

그러나 bit의 수가 늘어날수록 사용해야하는 Full Adder의 수도 늘어나므로 작동 시간도 덩달아 늘어날 것으로 보이니 이 문제를 해결하는 것이 앞으로의 과제가 될 것 같다.

1. 참고문헌

공진흥 / 컴퓨터공학기초실험2 / 새빛관303호(광운대학교) / 2017년.

이준환 / 디지털논리회로2 / 참빛관B101호(광운대학교) / 2017년.