CS211 Lab#2 Final Report

20160410 윤석훈

1. Source Code

library IEEE;

use IEEE.std\_logic\_1164.all;

ENTITY CIR IS

PORT (X1,X2,X3: IN std\_logic;

ANS: OUT std\_logic);

END CIR;

ARCHITECTURE REDBOX of CIR is

begin

ANS<=((X1 OR X2) AND ((NOT X1) AND X3));

end REDBOX;

library IEEE;

use IEEE.std\_logic\_1164.all;

ENTITY LAB2 IS

PORT (sw: in std\_logic\_vector (3 downto 0);

ledr: out std\_logic\_vector (0 downto 0));

end entity;

ARCHITECTURE YSH of LAB2 is

component CIR

port(X1, X2, X3 : in std\_logic;

ANS: out std\_logic);

end component;

signal A: std\_logic;

begin

U0: CIR port map

(X1 => sw(0), X2 => sw(1), X3=> sw(2), ANS=>A);

ledr(0) <= sw(3) xor A;

end architecture;

🡺((X1 or X2) and ((not X1) and X3)) xor X4 회로를 구현하는 코드인데, X1, X2, X3, X4는 FPGA의 SW[3 downto 0] 스위치를 통해 입력되고 결과값은 FPGA의 LEDR[0]를 통해 출력된다.

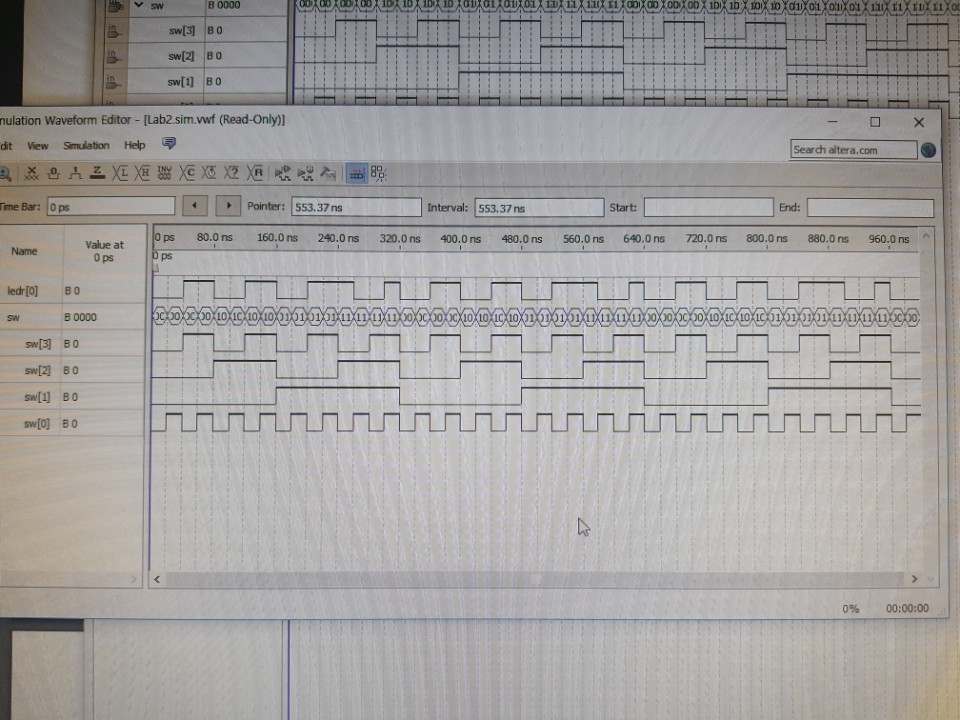
위 소스코드에서는 ((X1 or X2) and ((not X1) and X3))가 CIR이라는 component를 통해 구현되었다. Component를 이용하기 위해서 LAB2라는 본 함수를 선언하기 전에 component의 함수의 입출력과 아키텍쳐를 선언하였다.

LAB2라는 본 함수에서는 입출력을 FPGA의 스위치와 LEDR로 설정하고, 아키텍쳐의 시작에서 CIR component를 선언하였다. 그리고 CIR을 통한 출력값(ANS)은 Signal 변수 A가 매개하도록 설정했다. U0: CIR port map에서는 LAB2함수의 변수 SW[0~2]와 Signal A를 각각 CIR에서 대응되는 변수 X1, X2, X3, ANS에 대응시킨다. 그러면 Signal A는 component를 통한 결과값이 전달하고 ledr[0] <= A xor SW[3] 은 주어진 전체 회로를 구현하게 된다.

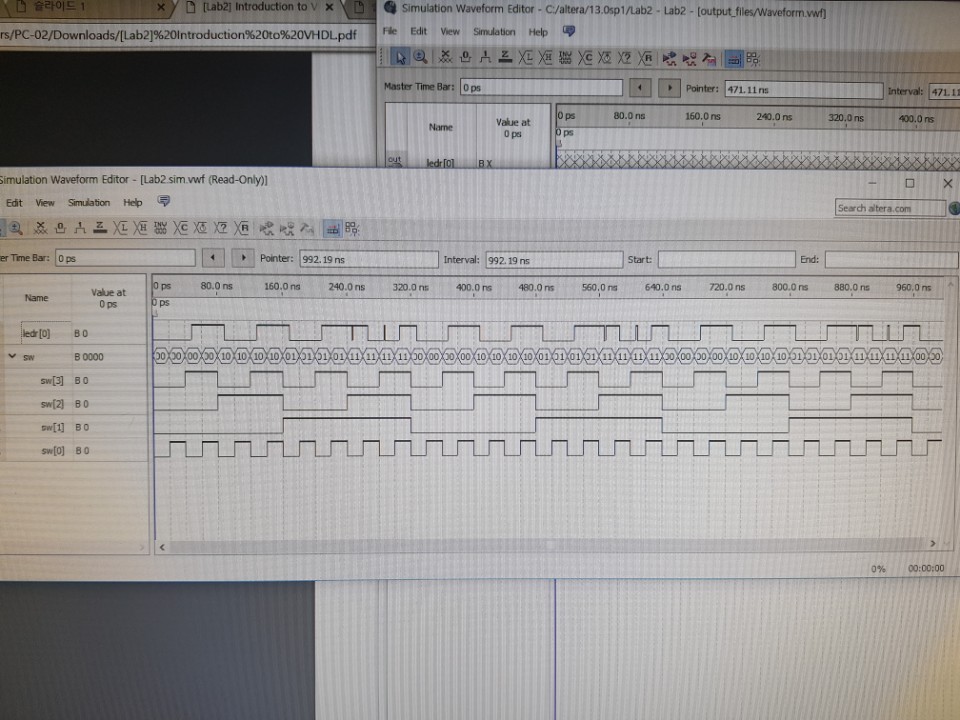
2. Component를 이용하면 좋은 점

같은 형태의 회로나 기능이 여러 군데에서 쓰일 경우 그 형태를 component로 미리 지정하면 매번 코드를 다시 적지 않아도 쉽고 효율적이고 빠르게 구현할 수 있을 것이다. C에서의 구조체나 JAVA에서의 클라스 선언과 같은 유용한 효과를 가질 것이다.

3. Functional Diagram & Timing Diagram



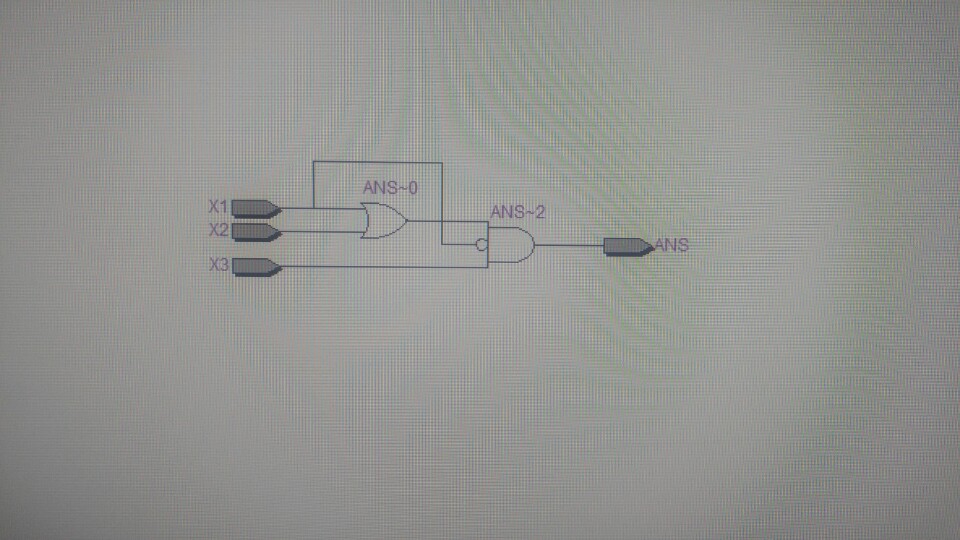
<Functional Diagram>

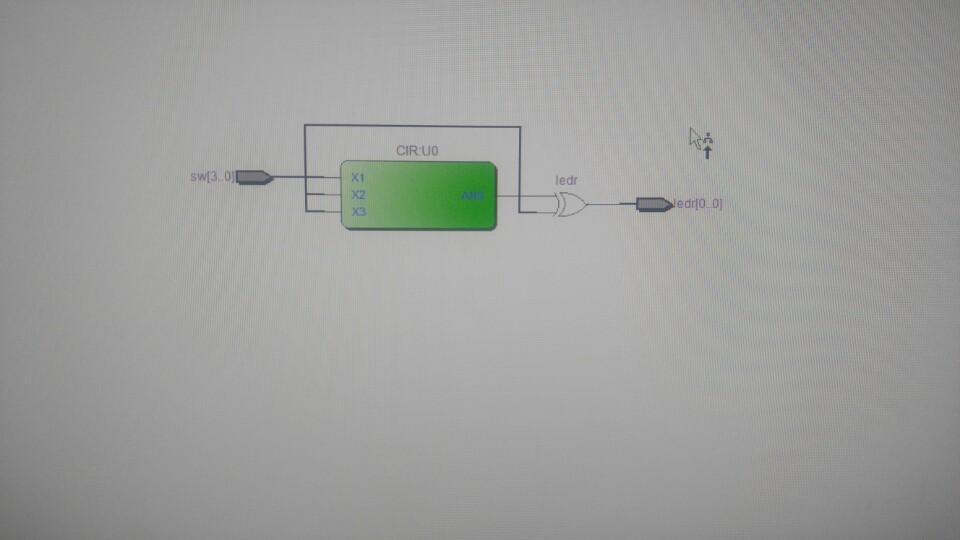


<Timing Diagram>

Functional Diagram은 주어진 코드로 입출력 값이 논리적으로 어떻게 조합되는지 보여준다. Timing Diagram은 프로그램이 진행될 때 실제 FPGA의 I/O 핀에서의 반응을 반영하기 때문에 결과값의 중간중간에서 가느다란 선들이 보이는 것 같다.

4. RTL Diagram





RTL Diagram에서는 소스 코드가 구현하는 회로의 모양을 변수명과 함께 제공해준다(아래 그림). 특히, component를 선언했으면 컴포넌트 이름과 port map 명과 함께 녹색 컴포넌트 표시를 해주는데, 컴포넌트를 클릭하면 컴포넌트만의 회로의 모양도 변수명과 함께 볼 수 있다(위 그림).