**2주차 결과보고서**

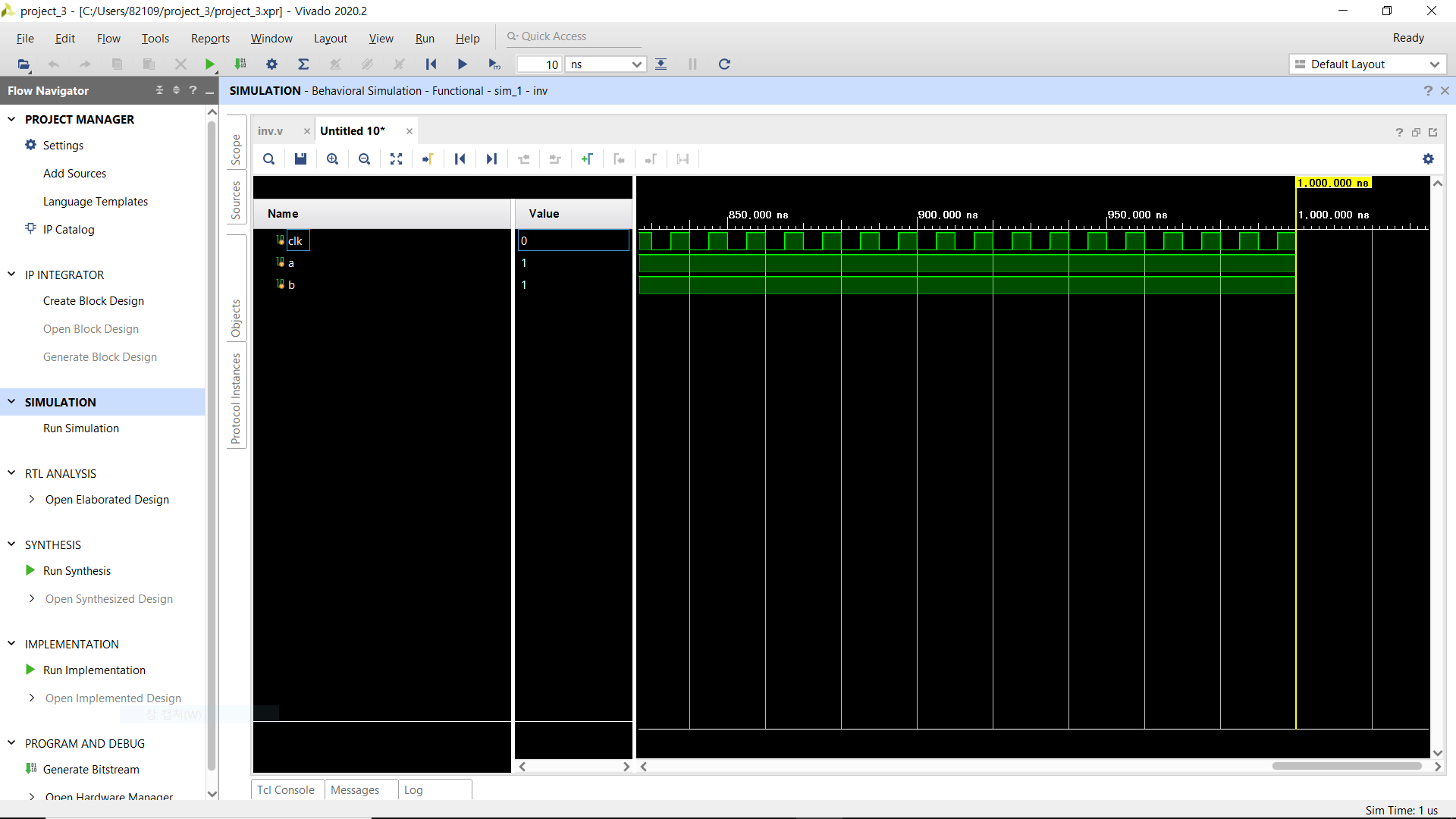
**전공: 중국문화/컴퓨터공학 학년: 4학년 학번: 20180305 이름: 고혜정**

**1. 연속 할당문, 절차형 할당문의 차이를 비교하여 설명하시오.**

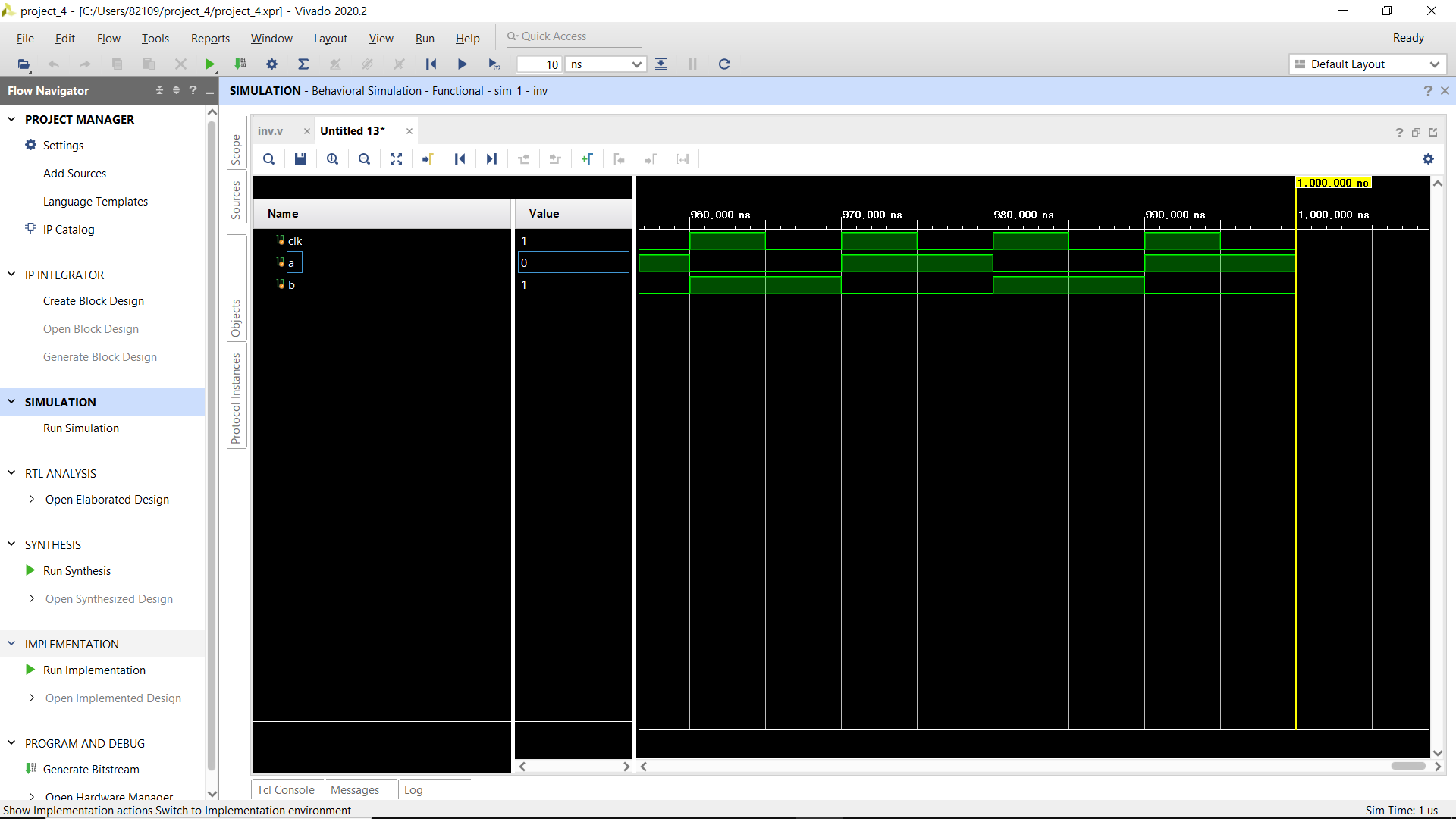
먼저, 할당문이라는 개념은 말 그대로 객체에 값을 ‘assign’ 하는 방법이다. Verilog에는 연속 할당과 절차형 할당이 있다. 연속 할당문은 게이트의 상호 연결을 따로 지정하지 않고도 combinational logic을 모델링한다. 이는 논리식으로 네트워크를 구동하는게 더 쉬워지게 한다. 연속 할당문은 net형 객체에 scalar 또는 vector 형태의 값을 할당한다.

절차 할당문은 ‘always’, ‘initial’, ‘task’나 ‘functions’와 같은 절차 내에서 수행되며 값을 변수에 배치할 때 사용된다. 변수는 동일한 변수를 다음 할당 때까지 유지하기 때문에 ‘memory word’라고도 부른다. 절차 할당문은 if-else-if 또는 looping 매커니즘과 같은 제어 흐름문을 사용하여 제어 및 수정이 가능해진다.

**2. Blocking 및 nonblocking 문법의 차이를 simulation을 통해 설명하시오.**



(blocking을 직접 simulation 돌린 결과 화면을 첨부합니다.)



(non blocking을 직접 simulation 돌린 결과 화면을 첨부합니다.)

먼저, 직접 vivado에서 blocking과 non-blocking을 시뮬레이션 돌린 결과는 위와 같다. 간단히 말하자면 simulator에서 값을 할당할 때 선언하는 부분이 ‘=’이면 blocking 방법이고 ‘<=’이면 non-blocking 방법이라고 할 수 있다.

blocking부터 살펴보자면, 말 그대로 신호의 흐름을 막는다고 이해할 수 있다. 즉, blocking 구문을 사용하면 c언어 처럼 순차적으로 실행하게 된다. 즉, 각 문장의 할당에 대한 실행이 완료되어야 다음 문장으로 제어가 넘어가는 것이다. 따라서 delay가 발생하면 블록문 내에서 계속 누적된다.

반면에 non-blocking은 신호의 흐름을 막지 않는 것이기 때문에 순서와 상관없이 동시 실행이 가능해진다. 정해진 assignment scheduling에 의해 좌변에 할당되며 우변은 블록문의 실행과 동시에 평가된다.

**3. Verilog 의 for문, if문, while문, case문 을 C언어와 비교하여 설명하시오.**

먼저, verilog의 for이 c언어와 같은 점은 해당 블록을 조건식에서 지정한 만큼 반복한다는 것이다. 다른 점이 있다면 verilog의 for문에서 사용하는 인덱스의 경우 무조건 정수형으로 선언되어야 한다.

verilog의 if문은 조건이 참이면 다음 문장을 실행하고, 아니면 else문으로 넘어가는 기본적인 형태와 논리적 흐름은 c언어와 같지만 여러 줄을 쓸 때는 반드시 실행 범위에 begin - end이 추가된다는 차이점이 있다. 이는 c언어에서 {} 중괄호와 같은 역할이다.

verilog의 while문 형식은 while ( 조건문 )이고 이는 c언어와 크게 다른 점이 없다. 마지막으로 verilog의 case문은 c언어의 switch-case 구문과 거의 비슷하지만 verilog의 case는 마지막에 endcase를 사용한다. 그리고 c언어와는 다르게 ‘casex’ 또는 ‘casez’와 같은 ‘don’t care condition’을 사용할 수 있다.

**4. Verilog 의 net 형 자료형에 대해서 조사하시오.**

Net형 자료형은 논리 게이트 혹은 모듈 사이의 연결을 나타내기 위해 사용되는 자료형이다. net형 자료형에는 wire, tri, wand, wor, triand 등등이 있다. net형 자료형은 연속 할당문이나 gate primitive와 같은 드라이버의 값에 의해 값이 연속적으로 유지된다. 만약, 별도로 driver가 연결되지 않으면 default값으로 high-impedance가 된다. net 값이 변할 때마다 자동적으로 net에 새로운 값이 전달되며 따로 크기를 정하는 vector를 지정해주지 않으면 1 bit net이다.

**[참고문헌]**

Alan Marcovitz, McGrawHill (2008), Introduction to Logic and Computer Design(3rd ed.)

컴퓨터공학실험II, ‘verilog 소개’ 강의자료

한국기술교육대학교 전기전자통신공학부, ‘디지털 시스템 설계 및 실습’ 강의자료

(https://cms3.koreatech.ac.kr/sites/yjjang/down/dsys15/DSD02\_verilog2b.pdf)