2주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20231523 이름: 김민정

**1. 연속 할당문과 절차형 할당문**

연속 할당문은 주로 net형 변수에 사용된다. (net형 변수 : tri, wire 등) 이는 할당된 변수의 종속변수가 변화할 때마다, 변수의 값을 업로드 시키는 것이다. 예를 들어 wire a= b+c가 있다고 하자. 여기서 b나 c값이 변화한다면 이를 a에도 반영시키는 것이다. 처음 할당할 때는 assign을 쓰고, 이에 대한 계속된 할당을 제거하고 싶다면 deassign을 써서 연속된 흐름을 차단한다.

절차형 할당문은 주로 reg형 변수에 사용된다. 이는 특정 조건이 만족할 때마다 절차적으로 값을 갱신하다. 즉 조건을 만족할 때만 변수의 값을 업로드 시킨다. 관련 구문에는 always와 initial이 있다. always와 initial 모두 조건이 성립하면 변수가 업로드 된다는 공통점이 있지만, always는 조건이 성립할 때마다 계속, initial은 한번만 실행된다는 차이점이 있다.

**2. Blocking 및 nonBlocking문법**

Blocking은 begin과 end 사이의 선언이 순차적으로 이뤄지고 이와 달리 nonBlocking은 begin과 end 사이의 선언이 병렬적으로 이뤄지고 마지막 end에서 모든 값이 업로드 된다. 아래 사진에서 코드의 차이는 always @(posedge clk)begin ~ end에서 ~ 에서밖에 보이지 않는다. 여기서 Blocking은 =으로 표현되며, nonBlocking은 <=으로 표현된다. Blocking에서는 a값에 1을 저장하고 이후 b에 업로드 된 a값인 1이 업로드 되어 결국 a, b모두 1로 고정되는 결과가 나타난다. 하지만 nonBlocking은 a에다가 b의 값인 1을 업로드 한다는 계산을 예약하고, 아래 선언인 b에다가 a의 값 0을 업로드한다는 계산을 예약한다. 그래서 end에 도착하면 a에는 1의 값, b에는 0의 값이 저장되는 결과가 나타난다. 그러므로 nonBlocking에서는 a, b 둘 다 하나의 값으로 고정되는 결과가 나타나지 않고 계속 수가 교차하는 결과가 나타난다. 이는 아래 두번째 사진인 simulation에서 쉽게 확인할 수 있다.

**텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명**

**텍스트, 스크린샷, 라인, 도표이(가) 표시된 사진

자동 생성된 설명**

**3.** **Verilog의 for문, if문, while문, case문(C언어와 비교를 통해)**

for문에서는 Verilog는 C언어와 비슷한 양상을 보이지만, 중괄호를 쓰지 않고 begin과 end를 이용한다는 점과 ++과 – 연산자를 쓰지 못한다는 것에 차이점을 보인다. 또한 순차적으로 명령을 수행하는 C언어와 달리 Verilog는 nonblocking를 통해 명령을 병렬적으로 수행할 수 있다.

if문와 while문은 for문과 같이 begin-end 차이점이 적용된다. 그 외의 문법은 동일하다.

마지막으로 case문은 위의 내용과 상이하다. C언어에서 하나의 case만을 실행시키기 위해서는 break를 써서 행동을 하나하나 나눠줘야 한다. 하지만 Verilog에서는 break를 쓰지 않아도 프로그램은 해당 case 하나의 행동을 실행한다. case문에서도 중괄호를 쓰지 않고 마지막에 endcase를 쓰면서 명령을 구분한다.

**4. Verilog의 net형 자료형**

Verilog에서 net자료형은 회로의 소자 사이의 물리적인 연결을 추상화 하여 표현한 자료형이다. 이러한 자료형은 앞에서 말했 듯 논리 게이트나 모듈과 같은 하드웨어 소자 간의 물리적 연결을 표현하기 위해 사용된다. Net 자료형에는 wire, tri, wand, triand, trior, tri0, tri1 등이 있다. 여기서 예시 몇 개를 소개해보겠다. wire은 default net 자료형 이며, 회로에서 신호를 연결할 때 사용된다. tri는 여러 드라이버가 신호를 고유할 때 사용된다. wand는 드라이버 중 and 연산을 한다.