2주차 예비보고서

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

**1. HDL과 Verilog이외의 HDL 예시**

HDL은 hardware description language의 약자로 전자회로를 조정하기 위해 이용되는 프로그래밍 언어이다. 여기서 HDL을 통해 회로의 동작과 회로의 구조를 원하는 대로 조작할 수 있다.

HDL은 두 가지의 시스템을 설계할 수 있는데, 그 두가지는 프로세서/집적회로와 FPGA이다. 전자에서는 회로가 설계되고 제작하기 전에 그 회로의 동작을 예상하는 모델을 제공한다. 후자는 설계 중간중간 회로의 퍼포먼스를 보기 위해 테스트용으로 쓰인다.

HDL 종류에는 Verilog, VHDL, 그리고 AHDL이 있다. 여기서 VHDL과 AHDL에 대해 중점적으로 설명하겠다. VHDL은 VHSIC Hardware Description Language의 약자로 디지털 회로와 mixed-signal을 표현하는 하드웨어 기술이다. 여기서 VHSIC는 Very-High-Speed Integrated Circuits의 약자이다. 다시 돌아와서 VHDL은 위에서 서술했던 것처럼 회로를 조작하기 위해 이용되며, 시간과 동시성을 표현할 수 있는 표기법이 있다. AHDL은 altera hardware description language의 약자이며 CPLD와 FPGA을 조작할 수 있는 알테라사 그들만의 HDL이다. 그래서 알테라 사의 쿼터스와 맥스의 컴파일러로 컴파일된다. 위 두가지의 언어는 C언어와 같은 문법을 가지는 특징이 있다.

**2. Verilog의 역사와 발전 과정**

Verilog는 1983년 게이트웨이 디자인 오토메이션 회사에서 개발된 HDL이다. 후에 게이트웨이 디자인 오토메이션 회사는 카덴스 디자인 시스템 회사에 인수되어 그들이 만든 Verilog는 업계의 표준 HDL로 자리를 잡는다. 또한 1991년 내부 LRM이 공개됨에 따라 1995년도에 IEEE에서 최초로 표준화되어 현재까지도 널리 쓰이고 있다

**3. Verilog의 기본적인 구조와 문법**

Verilog의 구조는 머리부, 선언부, 몸체부로 구성된다. 여기서 머리부는 module module의 이름(port list); 구문으로 이루어진다. 뒤에 세미콜론으로 끝내는 문법이 C언어와 비슷함을 확인할 수 있다. 여기서 모듈의 이름을 정할 때는 Verilog에서 사용되는 명령어의 이름을 쓰면 안된다. 선언부에서는 필요한 변수들을 모두 선언한다. 여기서 포함되는 변수는 비트 폭, reg 및 wire, parameter, 그리고 포트들의 방향들이 포함된다. 여기서 reg는 always, initial과 같은 절차형 할당문에 의해 값을 얻는 객체이며 wire은 변수들의 연결을 알려주는 변수이다. 마지막으로 몸체부는 위에서 선언했던 변수들을 이용해 회로의 동작과 구조를 조정하는 부분이다. 마지막으로 endmodule을 써서 하나의 모듈을 완성한다. 주의를 해야하는 점은, 머리부에서 세미콜론으로 부분을 완성시킨 것과 달리 마지막 endmodule에서는 세미콜론을 쓰지 않는다.

Verilog의 문법은 C와 비슷한 맥락을 가진다. 첫번째 Verilog에서 상수의 선언은 다음과 같다. 비트 수가 제한이 있는 reg값으 선언시에는 (비트수)’(입력형식)(입력값)이 된다. 예시를 보여주자면 4bit의 2진수의 1000을 선언하고 싶다면 4’b1000;으로 서술하면된다. 만약 비트수를 정해주지 않을 거라면 4를 빼고 ‘b1000이라고 서술하면된다. 수를 signed로 표현하고 싶을 수 있다. 그럴 때에는 맨 앞에 -를 붙여주면 된다.

두번째, Verilog 연산자는 C와 모두 동일하다. 여기서 추가되는 것이 있는 ~^과 ~& 이다. 둘 다 원래의 ^과 &연산에서 정반대의 결과를 내는 것이다. ~^는 서로 같은 값이면 1이 되고 아닌 경우에는 0이된다. ~&은 원래 둘 다 1인 경우에 1의 output을 내는 &와 달리, 둘 다 1일 때 0을 낸다. 조건 연산자는 C언어에서 축약형인 a = A? B:C가 된다. A의 조건문이 참이 되면 B를 실행하고 거짓이 되면 C가 되는 것이다. 결합 연산자인 {}는 내부의 원소를 모두 이진수를 기준으로 합치는 연산자이다. 마지막으로 반복연산자인 {{}}은 앞의 숫자만큼 원소를 복제시키는 것이다.

프로그래밍언어와 구분을 만들어내는 구문인 Timescale에 대해 설명하겠다. 구문은 timescale 시간단위 / 정밀도 이다. 여기서 시간단위는 선언한 파일 내의 기준 시간점이 되는 것이고, 정밀도는 그 시간단위 내에서 허용되는 오차범위이다.