2주차 예비보고서

전공: 컴퓨터공학과 학년: 3학년 학번: 20211547 이름: 신지원

**1.**

HDL 이란 Hardware Description Language의 줄임말로 전자회로를 정밀하게 기술하는 데 사용되는 하드웨어 기술 언어다. 회로의 동작이나 구조를 직접 구현하여 시뮬레이션을 통해 구현 여부를 확인할 수 있다. 실제로 사용하는 하드웨어(디지털 기기)는 클럭에 따라 작동하기 때문에 HDL의 시뮬레이션은 이와 유사하게 클럭의 신호에 따라 작동 여부를 보여준다. 이처럼 HDL 은 소프트웨어 프로그래밍 언어와 달리 하드웨어를 다루기 때문에 시간과 동시성을 설정할 수 있는 기능들이 존재한다. 원래는 회사 마다 각기 다른 고유의 HDL 포맷으로 구현하여 표준 EDIF 포맷으로 변환하였다면, 최근에는 우리가 실습에서 사용하는 Verilog 와 VHDL 과 같이 표준 HDL 을 직접 사용하는 것으로 변화하였다.

VHDL 은 Verilog 와 마찬가지로 HDL 의 종류 중 하나며 디지털 회로의 설계, 검증, 구현 등의 용도로 사용되고 있다. 원래는 ASIC 의 문서화에 사용하기 위한 언어였으나 논리합성을 하여 실제 회로 형태를 출력하는 기능을 개발하여 오늘날에는 위와 같이 활용된다. VHDL 은 Verilog 와 차이점을 가지는데, Verilog 에 비해 더 높은 수준의 모델링이 가능하며 pascal 과 비슷한 문법을 가지고 있다. 따라서 당연히 코드의 구현에서도 차이가 있다. 아래는 각각 AND 게이트를 구현한 코드 중 일부다.

- VHDL

**entity** **ANDGATE** **is**

**port** (

I1 : **in** std\_logic;

I2 : **in** std\_logic;

O : **out** std\_logic);

**end** **entity** **ANDGATE**;

- Verilog HDL

**module inv(**

input a,b,

output c

**);**

**assign** c = a+b;

**endmodule**

**2.**

Verilog 는 본래 모의시험용 하드웨어 기술 언어였지만 회사 내 표준 HDL 로 발전하게 되었다. 1983년 Gateway Design Automation에서 개발되었고 이후엔Cadence Design System이라는 회사에 인수되어 업계 표준으로 자리잡게 되었다. 1991년에 내부 LRM(Language Reference Manual)을 공개함으로써, 1995년도에 IEEE에서 최초로 표준화되었다. 현재 VHDL과 Verilog는 Accellera라는 단체가 유지, 관리하고 있으며 지속적으로 확장되고 있다.

**3.**

Verilog 는 기본적인 구조는 module 이다. “module” 로 시작하여 “endmodule” 로 종료를 표현하며 보통 파일 1개에 module 1개만 작성한다. module 과 endmodule 사이는 크게 머리부, 선언부, 몸체부로 나누어진다. 머리부는 모듈이름과 포트목록들을 지정하며, 선언부에서는 모듈에서 필요한 것들을 선언하고, 몸체부에서는 선언부에서 선언한 기능들의 동작, 구조들을 표현하여 실질적으로 회로를 설계한다.

- 머리부

“module” 로 시작하여 모듈이름과 포트목록들을 지정하는 부분이다. 예를 들어 ‘module module\_name(port\_list); ‘ 로 작성할 수 있다.

- 선언부

포트방향, 비트 폭, 자료형, 매개변수 등 모듈에서 필요한 것들을 선언하며 구성된다. port, reg, wire, paramenter 를 선언할 수 있다.

**port** 는 컴퓨터와 다른 컴퓨터 사이에서 인터페이스의 역할을 담당한다. **reg** 는 always나 initial 구문 내부의 절차적 할당문으로 값을 받는 객체의 자료형으로 always 내부에서 값을 할당할 때는 항상 reg 자료형으로 선언해야 한다. **wire** 는 변수들이 모듈내에서 어떻게 연결되어 있는지 알려주는 변수이다. reg 가 실제로 값을 변경한다면, wire 는 그 자체로 값 변경은 불가하지만 모듈과 모듈을 이어주는 선으로 이해할 수 있다. **paramenter** 는 c언어의 const 와 비슷하게 사용하며 상수값을 의미한다.

- 몸체부

몸체부는 선언부에서 선언한 기능들의 관계가 나타난다. 따라서 회로를 구성할 수 있다.

<예시>

1. module full\_add (a, b, cin, sum, cout);

2. input a, b, cin;

3. output sum, cout;

4. wire w1, w2, w3;

5. half\_add1 U1(a, b, w1, w2);

6. half\_add2 U2(w1, cin, sum, w3);

7. endmodule

위의 예시를 통해 1번줄이 머리부이며 모듈이름과 포트목록이 선언되었음을 알 수 있으며, 2번부터 4번까지 선언부로 모듈에 필요한 것들을 선언하였음을 알 수 있다. 5번에서 6번은 몸체부로 회로의 기능을 표현하였으며 마지막 7번으로 모듈을 종료하였다.

Verilog 의 데이터 타입은 register 와 net 로 구분된다. register 는 추상적 저장 장치로 always, initial 과 같은 절차형 할당문에 의해 값을 받는 객체를 말한다. 예를 들어 정수형 변수인 integer 이나 시간형 변수인 time, realtime 등이 있다. net는 디바이스의 물리적인 연결을 말하며 위에서 언급한 wire 이 대표적이다. 그 외에도 tri 와 같은 것도 있다.

상수의 표현은 보통 비트로 표현한다. ‘[bit 수]’[입력 형식][입력 값]’ 의 형식으로 표현하며 예를 들어, ‘4’b0110’ 과 같이 표현할 수 있다. 비트로 표현하지 않을 때는 ‘integer 213’, ‘hexadecimal 22’ 과 같이 표현할 수 있다.

연산자는 보통의 컴퓨터언어와 대부분이 비슷하다. 산술연산자(+, -, %, \*, /), 관계연산자(==, !=, >, >=), 논리 연산자(&&, ||, !, &, |, ~, ^, ^~, ~^), 시프트연산자(>>, <<) 등을 가지고 있다. 특히 시프트 연산자를 사용할 때 밀려난 비트는 사라지고 새로 생긴 자리는 0으로 채워진다.

조건문과 반복문 또한 본교 1,2 학년에 배웠던 파이썬, C언어와 비슷하게 처리된다. 특징이 있다면 if 문에서 할당문을 작성할 때 ‘begin (할당문) end’ 의 구조로 작성한다는 점과, for문도 마찬가지로 순차구문을 작성하기 전에 begin 을 작성하고 반복할 내용을 작성한다.

**참고문헌**

- https://ko.wikipedia.org/wiki/하드웨어\_기술\_언어

- https://ko.wikipedia.org/wiki/VHDL