2주차 예비보고서

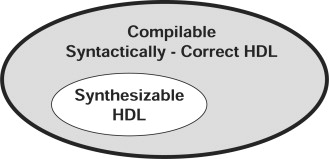
전공: 컴퓨터공학과 학년: 2학년 학번: 20191619 이름: 이동석

**1. HDL(Hardware Description Language)**

HDL은 이름 그대로 원하는 디지털 회로(하드웨어)의 동작이나 구조를 컴퓨터로 기술하는 데 사용되는 언어이며, 1주차에 배웠던 시뮬레이션 단계를 통해 언어로 구현한 회로를 검증할 수 있다.

과거 회로도를 표현할 땐 직접 도면에 소자를 배치하였지만, IC(직접회로)기술의 발달로 하나의 칩에 수 천개 이상의 트랜지스터, 5만 이상의 게이트정도 되는 하드웨어를 기존의 방법으로 설계하는 것은 불가능에 가까웠다. 이에 앞서 말했듯 디지털 회로 설계를 위한 언어로 HDL이 등장하였다. HDL의 특징으로는 추상화, 동시성 및 병행성, 그리고 병렬성이 있다.

HDL 등장 초기에는 소규모의 논리 시뮬레이션만 가능했지만, 기술의 발달에 따라 디지털시스템의 대부분의 방면에서 사용이 가능해졌다. 또한, 일반적인 프로그래밍언어(ex: C, C++)와 다르게 코드를 구현할 때는 **하드웨어에 대해 이해**한 후 작성해야한다.



위 그림은, HDL의 이중적 속성을 나타낸다. 신기하게도, HDL로 구현한 코드가 컴파일이 가능하더라도 FPGA에서 동작가능한 디자인으로 합성(Synthesis)이 가능하다는 보장은 없다고 한다.

대표적인 HDL로는 실습때 배운 Verilog와 미국국방성에서 제작한 VHDL이 있다. 이외에도 자바를 기반으로 한 JHDL, 파이썬을 기반으로한 MyHDL 등이 있다. VHDL과 Verilog HDL은 IEEE라고 하는 전기전자기술협회의 공식적인 인증을 받아 거의 표준으로 사용된다.

**2. Verilog 역사 및 발전**

Verilog HDL은 1984년에 Phil Moorby와 Prabhu Goel에 의해 개발되었으며, Gateway Design Automation Inc. 소유의 독점언어였다. Verilog는 특히 HILO-2라는 Brunel 대학에서 개발된 언어의 영향을 가장 크게 받았다. 이후, 1990년까지 자체적으로 수정을 거듭했다. 1990년에 Cadence Design System에 인수가 되었는데, Cadence는 Verilog의 관리를 위해 OVI(Open Verilog International)을 설립하고, Verilog를 공개도메인으로 이전하도록 했다. 이런 노력으로 Verliog HDL은 IEEE에게 공식적으로 인정받아 Std1364-1995로 표준이 되었다. 이후 5-6년간 사용자로부터 피드백을 받고 IEEE std1364-2001로 더 나은 Verilog가 표준이 되았다. 마지막으로 2005년에 System Verilog가 발표되고 2009년에 Verilog 및 System Verilog를 통합한 IEEE std1800-2009로 통합되었다.

**3. Verilog 구조와 문법**

우선 Verilog의 Lexical Tokens을 알아야한다. 이는 White Space, Comment, Operator, Numbers, String, Identifier, Keyword로 되어있다.

우선 White Space는 이름 그대로 공백, 탭, 줄바꿈을 의미하며 다른 언어들과 마찬가지로 정리를 위하거나, 다른 어휘토큰들을 분리하기 위해 사용된다. 그러나, 문자열에서는 공백과 탭을 중요시 한다.

Comment는 주석으로 C언어에서 사용법과 동일하다. // 또는 /\* ~ \*/를 사용한다.

Operator는 단항, 2항, 3항 연산자가 존재한다.

Numbers는 정수형 10진수, 16진수, 8진수 그리고 2진수의 형태가 있다. 형식은 첫번째 토큰으로 size\_constant 두번째 base\_format 마지막, unsinged\_number이다.

String은 reg 타입의 변수이다. 부호 “ “ 의 사이에 있는 문자들을 의미하며 12개의 문자는 8\*12 = 96bit 사이즈가 필요하다.

Identifier는 객체의 고유한 이름을 지정하기 위해서 사용하며 대소문자를 구별한다. ( \_ 나 $ 도 사용 가능하다,) escaped identifier는 \로 시작된다.

Keyword는 미리 정의되어 있는 identifier를 의미한다. 그러나, escaped형태의 식별자는 포함하지 않는다.

Verilog에서 사용하는 Data type은 디지털 하드웨어에서 데이터의 저장 및 전송 요소들을 나타내기 위해 설계되었다.

우선 다음 4가지의 logical complement가 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명

또한, Net이라는 자료형도 존재하는데 이는 게이트와 같은 구조적 엔티티 간 물리적 연결을 나타낸다. Net의 default 초기화 값은 z 라고 한다. wire, trieg가 이 자료형에 해당한다.

다음으로 Variable자료형이 있다. 데이터 저장의 추상화라고 표현되어있는데 이는 기존 언어에서 사용되는 변수와 유사한 개념으로 reg, integer, time이 이 자료형에 해당하며 default 초기화 값은 x이다.

이 외에도 벡터, 배열, parameter가 있다.

Verilog HDL은 계층구조를 기본적으로 지원한다. 모듈의 정의는 키워드 module과 endmodule 사이에 들어간다. Module 뒤에 오는 것은 정의할 모듈의 이름이 된다. 모듈의 매개변수는 모듈에 대해 사용할 포트목록으로 정의할 수 있다. 마지막은 세미콜론(;)으로 끝나며 여기까지를 수업자료에 따르면 머리부라고 부른다.

이제 선언부에서는 port, reg, wire, parameter 등 모듈에서 사용할 변수들을 선언한다.

마지막으로, 몸체부에서는 구문을 통해 실질적인 회로의 동작이나 기능, 구조를 표현한다. 여기에서 연속할당문, 절차형 할당문을 쓰며 또한 객체지향프로그래밍처럼 하위모듈을 인스턴스 할 수 있다. C언어에서 사용하는 사용자 정의함수나 매크로 처럼 Verilog에서도 task와 function을 이용해 비슷한 동작이 가능하다. 그러나 예를들어 task는 delay, timing과 같은 제어구조가 있거나, input이 없는 경우에 사용해야한다는 등의 조건이 있다.

Verilog에서 대부분의 연산자와 구문은 앞서 말했듯 C언어와 비슷하다. 사칙연산을 포함한 산술연산자, 두 변수를 비교하는 관계연산자나 등가연산자(비트수를 자동으로 맞춰주거나, 실수형일 경우 자료형을 변환 후 비교함)가 있다. 등가연산자에는 처음보는 === 이나 !==가 있는데 합성을 위한 RTL수준에서는 사용하지 않는 것을 권장한다고 한다. 이 외에도 비트, 축약, 시프트, 결합, 반복, 조건 연산자 등이 있다. 당연하게도 계산시 우선순위도 존재한다.

C언어나 다른 언어와는 다르게 Verilog에는 특이하게 시간에 관한 구문이 존재한다. 논리 합성용 구문인 assign, if~else, case, always 등이 있다. Simulation용 구문은 initial, $fopen 그리고 Library인 table, sepcify가 있다.

간략하게 소개하자면 timescale은 뒤에 오는 모듈의 시간 단위와 시간 정밀도를 지정한다. 이때 시간단위란, 시뮬레이션의 시간 및 딜레이 값과 같은 시간의 측정단위이다.



**4. 참고문헌**

Hardware Description Languages. (2022). <https://www.sciencedirect.com/topics/computer-science/hardware-description-languages>.

한국기술교육대학교 전기전자통신공학부 . (n.d.). <https://cms3.koreatech.ac.kr/sites/yjjang/down/dsys11/M01_VerilogHDL01.pdf>.

R.C. Cofer, Benjamin F. Harding, in [Rapid System Prototyping with FPGAs](https://www.sciencedirect.com/book/9780750678667/rapid-system-prototyping-with-fpgas), 2006

"IEEE Standard for Verilog Hardware Description Language," in IEEE Std 1364-2005 (Revision of IEEE Std 1364-2001) , vol., no., pp.1-590, 7 April 2006, doi: 10.1109/IEEESTD.2006.99495.

Verilog HDL Background and History . (n.d.). https://digilent.com/reference/learn/fundamentals/digital-logic/verilog-hdl-background-and-history/start.