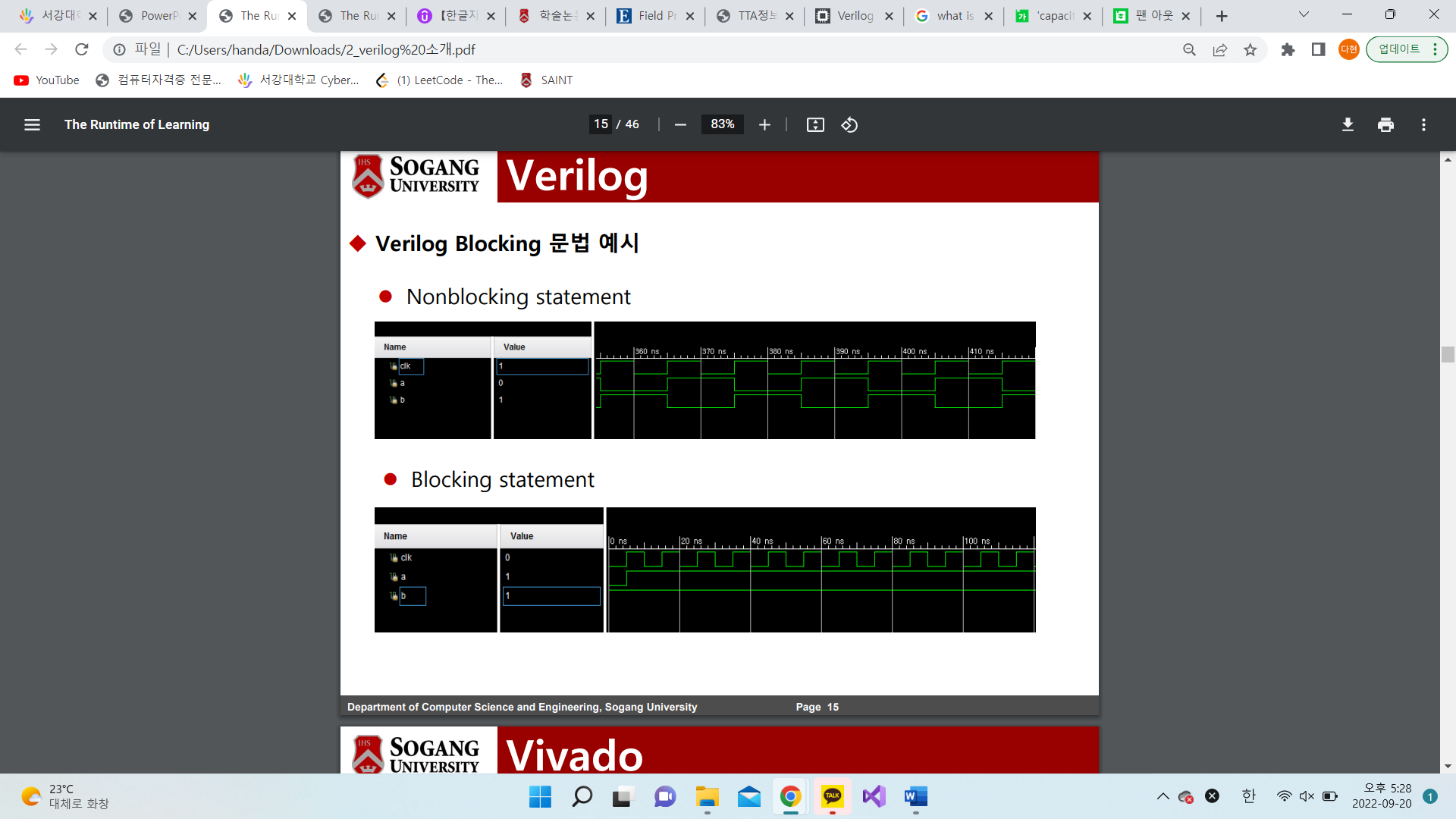
2주차 결과보고서

전공: 경영학과 학년: 3학년 학번: 20190963 이름: 한다현

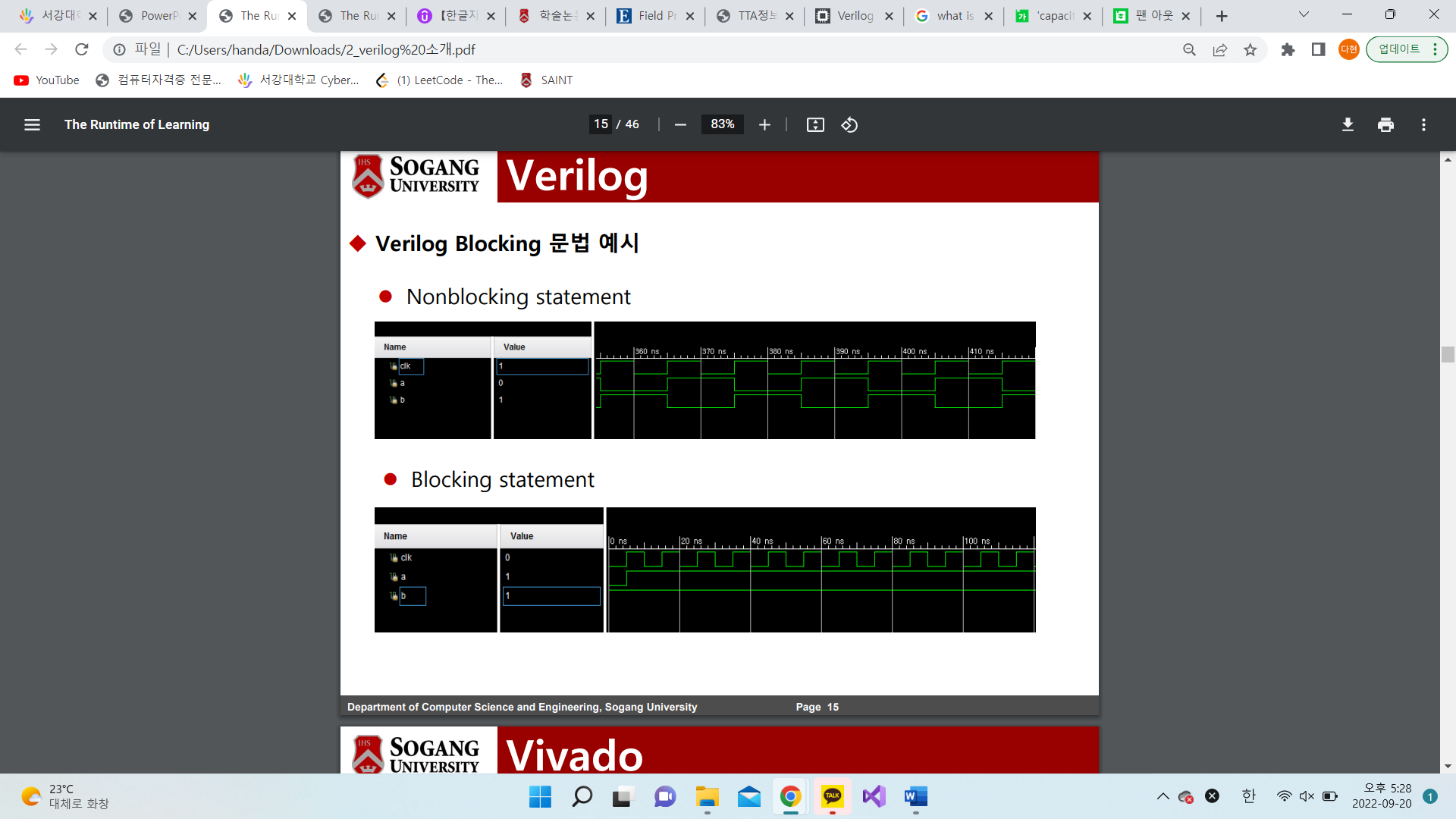
**1. 절차형 할당문은 다른 많은 프로그래밍 언어에서 사용되는 것과 같이 프로그램이 진행되는 절차에 따라 순차적으로 실행되는 문장이다. 반면 Verilog에서는 하드웨어가 동시에 실행될 수 있도록 설계해야 한다. 이러한 이유 때문에 연속 할당문을 사용해야 하는데, 연속 할당문은 할당을 항상 활성화하는 문장이며 따라서 여러 문장들이 동시에 실행될 수 있다. Verilog에서 연속 할당문은 assign으로 시작하고, 절차형 할당문은 주로 always 구문이나 initial 구문 내에서 사용한다.**

**2. Blocking statement는 절차형 할당문처럼 순차적으로 수행되고 blocking assignment(‘=’)를 사용한다. 반면 Non-blocking statement는 연속 할당문처럼 모든 계산을 동시에 수행하며 non-blocking assignment(‘<=’)를 사용한다. 2주차 강의자료에 있는 코드를 이용해 시뮬레이션을 돌려보면 다음과 같은 결과를 볼 수 있다. Non-blocking statement에서는 a와 b의 값이 동시에 새로 할당되기 때문에 계속해서 값이 바뀌지만 blocking statement에서는 먼저 a에 b의 값이 할당된 후 b에 a의 값이 할당되기 때문에 a와 b 모두 1을 값으로 가진다.**

**Non-blocking statement**



Blocking statment



**3. Verilog의 If-Else문은 always 구문이나 initial 구문 안에서만 사용할 수 있다. If-Else문을 사용하기 위해서는 begin-end문을 사용해야 하는데, 이는 C언어에서 {}를 사용하는 것과 같은 의미이다. 구문 안에 하나의 문장만 존재한다면 begin-end문을 사용하지 않을 수 있다. Verilog의 Case문도 If-Else문과 마찬가지로 always구문이나 initial 구문 내에서만 사용 가능하고, C언어에서와 마찬가지로 여러 조건의 발생 가능성이 있는 시나리오 상황에서 사용하기 유용하다. Verilog의 for문은 C언어의 for문과 마찬가지로 초기 조건이 있으며 종료 조건과 통제 변수가 만날 때까지 실행된다. While구문도 C언어와 비슷하게 조건이 참일 동안에 반복해서 실행되며 참이 아닐 때 종료된다. Verilog의 반복문 역시 always 구문이나 initial 구문 내에서 사용 가능하다.**

**4. Net 자료형은 다른 하드웨어 요소 사이의 전기적 상호 연결을 의미한다. Net 자료형에는 wire, wand, tri, wor, trireg 등 여러 종류가 있는데 가장 흔히 사용되는 종류는 wire이다. 이 때 wire는 모듈 내에 있는 변수들이 어떻게 상호 연결 되어있는지를 나타내며 wire의 디폴트 값은 ‘z’이다. Net 자료형은 Verilog에서 프로그램이 특정한 타이밍에 실행될 수 있도록 지연시키는 것을 의미하는 Delay에 사용되기도 하는데, 이러한 종류의 delay를 inertial delay라고 부른다.**

**참조:**

**Field Programmable Gate Array and Applications**

(http://libproxy.sogang.ac.kr/\_Lib\_Proxy\_Url/https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1718987&lang=ko&site=ehost-live&ebv=EB&ppid=pp\_5\_12)