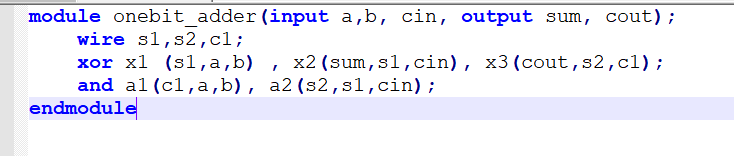
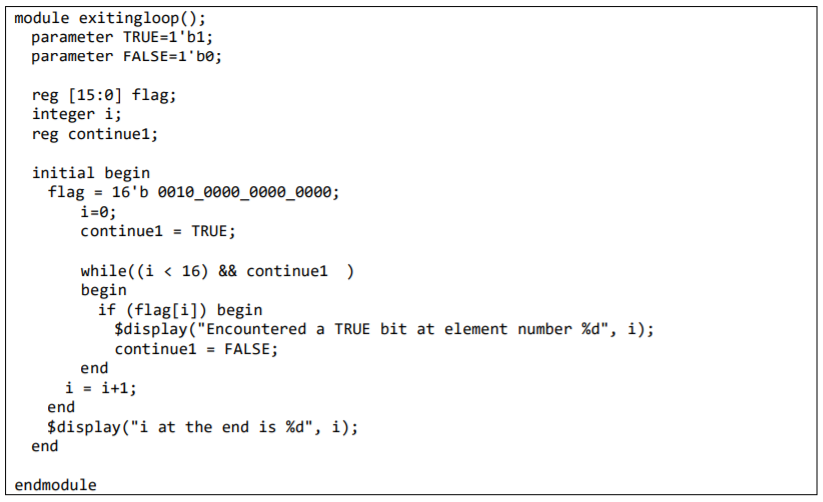
논회설 중간고사 대비

1번



2번

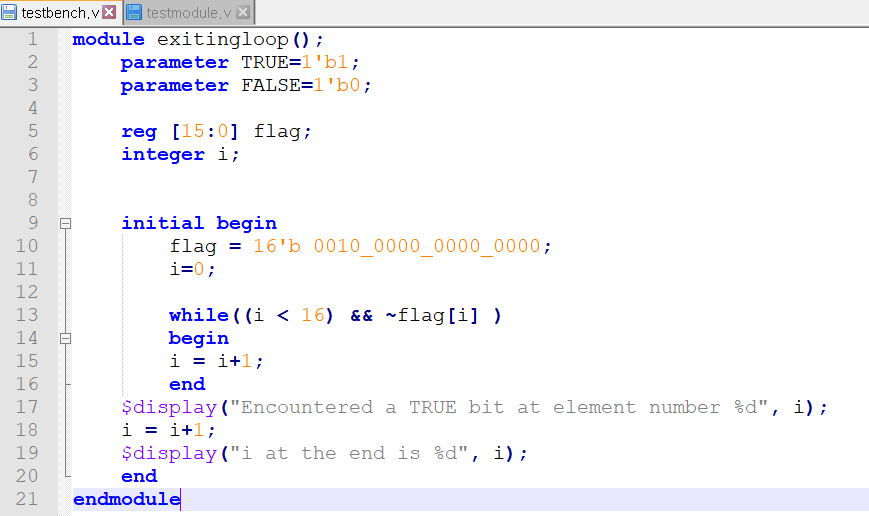


결과

# Encountered a TRUE bit at element number 13

# i at the end is 14

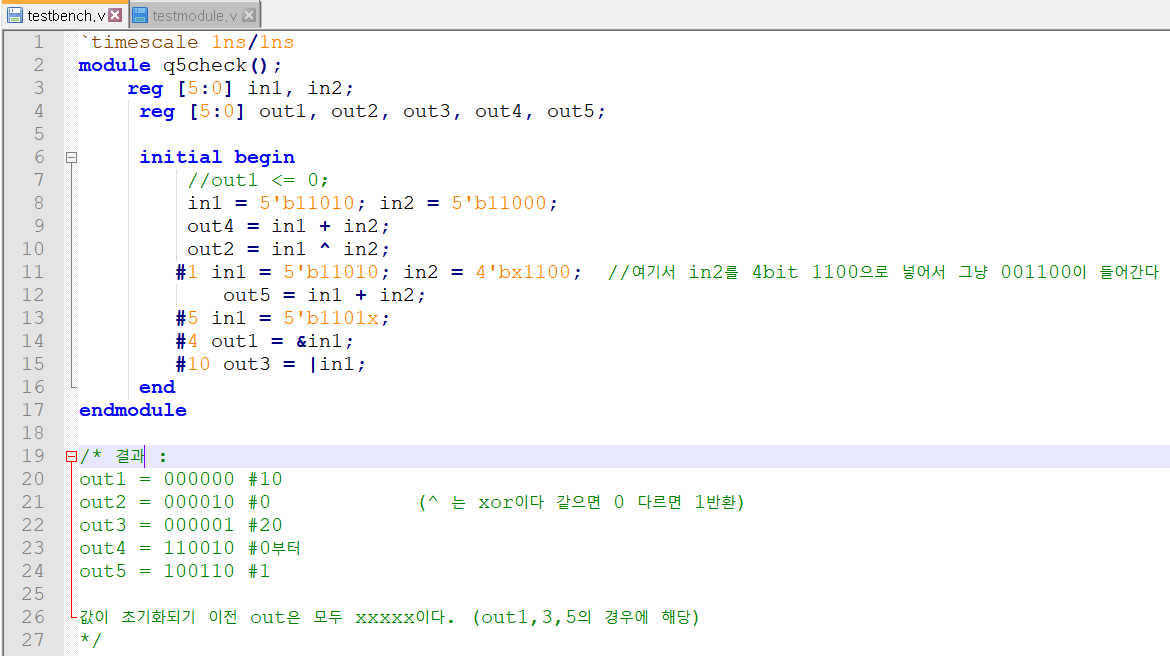
같은 결과가 나오도록 continue1 없이 while문 밖에서 값 출력



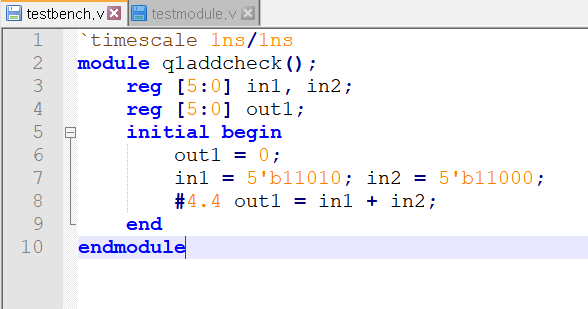
3번 4번 5번

시험범위 X

6번



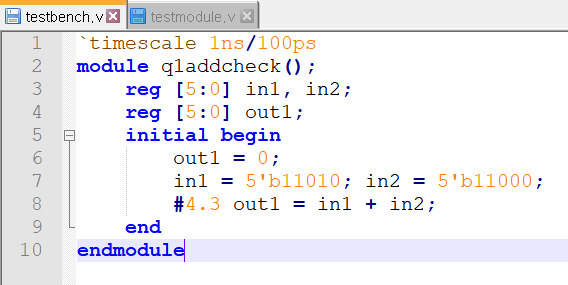
보너스



결과

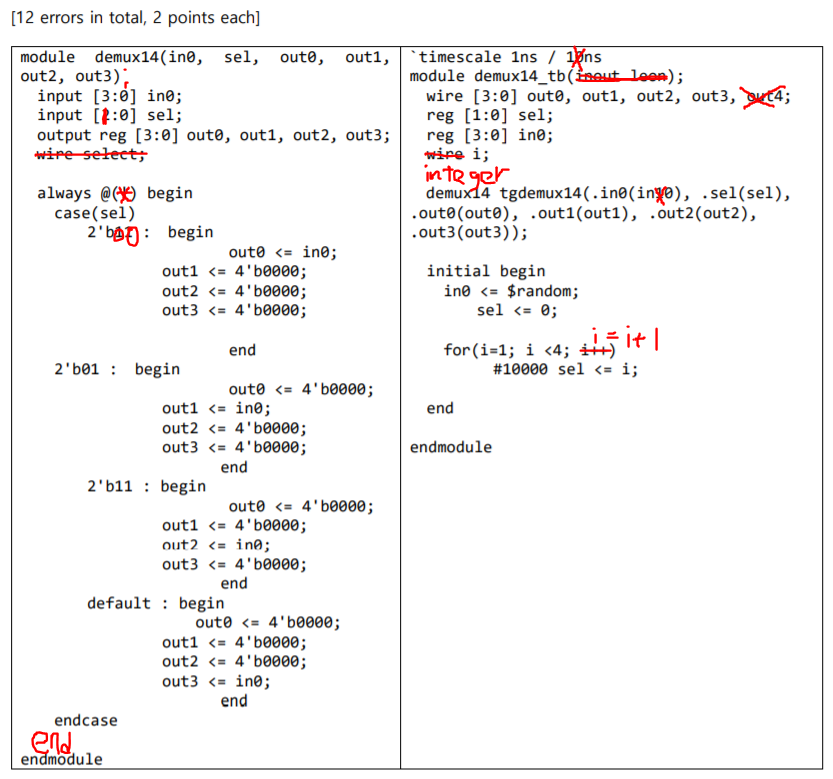
#뒤의 숫자는 반올림되어 적용되었다.

#4.3ns 를 나타내려면 timescale 1ns/100ps로 변경하면 된다.





퀴즈 내용



에러 유형

* 문장 끝 ; 확인
* 필요 없는 wire나 reg 확인
* 인풋 아웃풋 비트크기 확인 및 매칭이 잘 되어있는지 확인
* Begin / end 잘 묶였는지
* 케이스문 case 마지막은 default 하고 endcase로 종료
* 타임스케일 기본주기는 해상도보다 크거나 같아야 한다. `timescale 기본주기/해상도
* 테스트벤치에는 전달인자가 없다
* Integer 확인
* 인풋 아웃풋 이름 확인
* Output reg인지 확인 -> always@(\*) begin 확인
* i++은 없다 i = i+1 사용

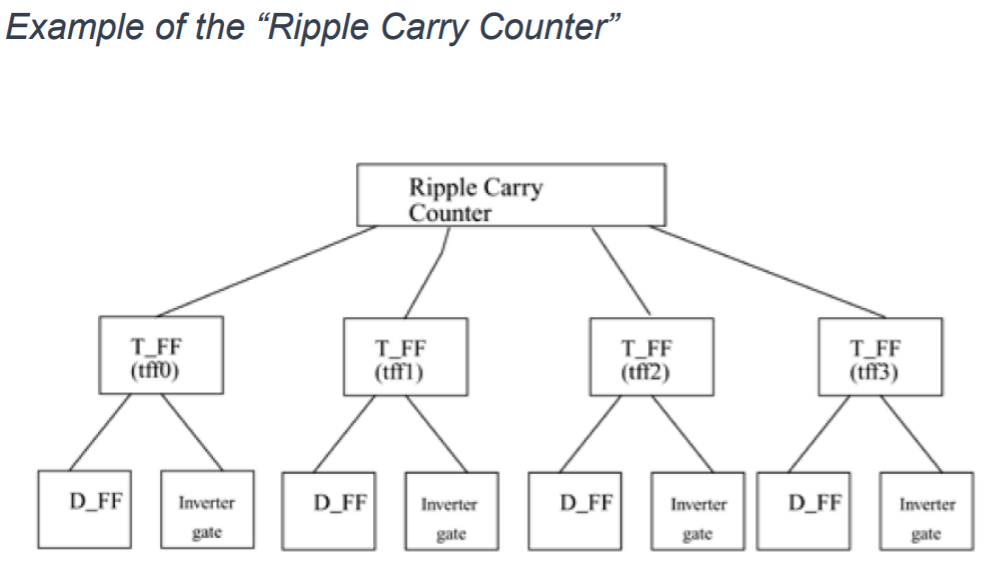
CH.1

HDL : Hardware Description Languages

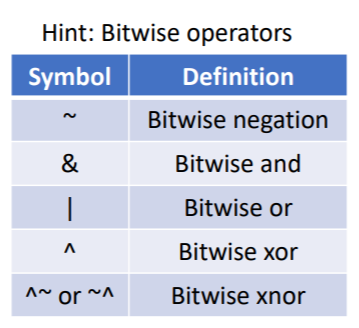
RTL : Register Transfer Level

CH.2

Design Hierarchy



CH.3

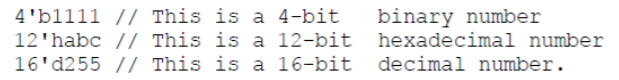


XOR 은 같으면 0 다르면 1 반환

3항 연산자

a = b ? c : d ;

sized number



Unsized number 는 32bits 이다

X ; Unknown Value

Z : High Impedance

Data types

Nets (wire) : 상태 표시

Registers (reg) : 변하는 값 assign 으로 값 변경 가능

Vectors : 여러 비트 묶음 [3:0]

벡터 접근시 변수명[인덱스] 로 접근가능

Integer (integer) : 카운팅에 사용 int 아님! , 모듈 안에서는 사용안함! 테스트벤치에서 사용

Arrays : n차원 벡터나 매트릭스 나타낼 때 사용 A[3:0][3:0];

시스템 명령어

$display(A,B); 값 나타내기

$monitor(); 이니셜 단계에서 사용, 값 변할때마다 표시

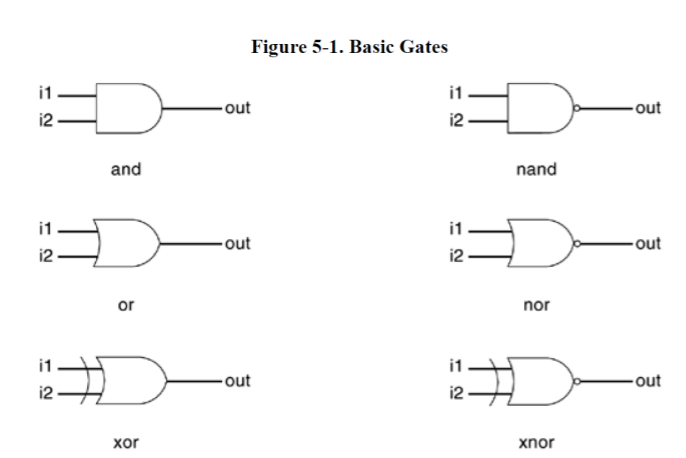
$stop

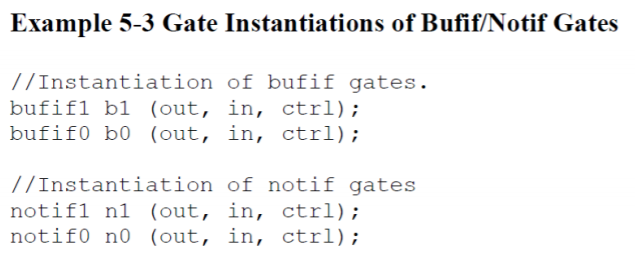
$finish

$random

CH.4

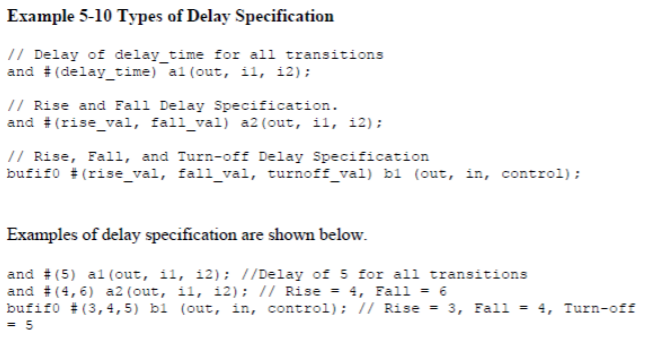
CH.5





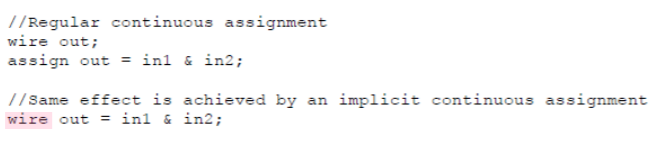
Bufif1 : 컨트롤이 1인 경우 버퍼기능 , 0인경우 Z 반환

딜레이

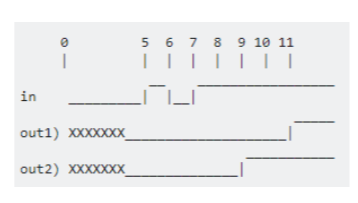


CH.6

Implicit (함축적인) Continuous Assignment



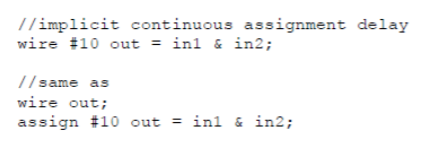
Regular Assignment Delay

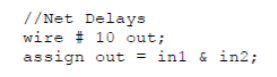


만약 4ns의 딜레이가 있는 작업이 있다면 입력유지시간이 4ns보다 작을 경우 중간에 작업이 취소되어 값이 나오지 않는다.

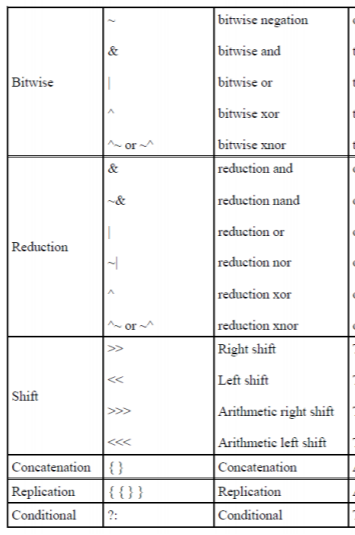
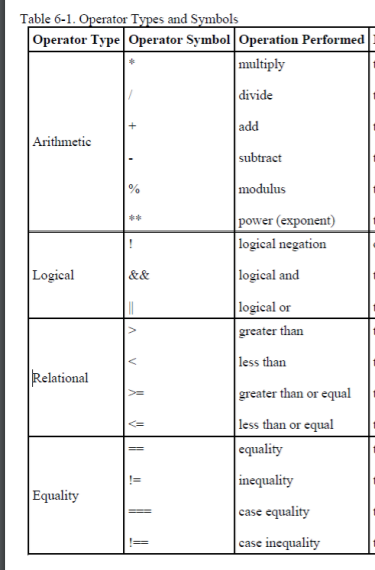
딜레이 전에 값이 바뀔 경우 다시 계산한 값 사용

Implicit continuous assignment delay





Operator Types



operands : 피연산자

operators : 연산자

산수 연산자(Arithmetic operators) : +, -, \*, /, % (modulus 나머지), \*\*(power 제곱)

+,- 는 양,음수 나타내는데도 사용

논리 연산자 : &&, ||, !

값이 있는거는 논리 1(값이 1이든 2이든 3이든 어떤 숫자라도 논리1),

값이 0이면 논리0,

값이 없으면 논리x

x && 1 == x

x || 0 == 0

관계 연산자 : 크기비교

equality 연산자 : ==, !=, ===, !==

x나 z가 들어오면 unknown 반환, ===이나 !==은 이것 또한 비교해준다.

bitwise 연산자 : ~, &, |, xor ^, xnor (^~,~^)

reduction 연산자 : 여러 비트 한번에 비교할때 사용

shift 연산자 : >>, <<, >>>, <<<

지정된 비트만큼 지정한 방향으로 이동시킨다(나머지는 0으로 채워짐)

concatenation 연산자 : {A,B} A와 B의 비트를 합쳐준다.

조건 연산자 : 3항 연산자 많이씀

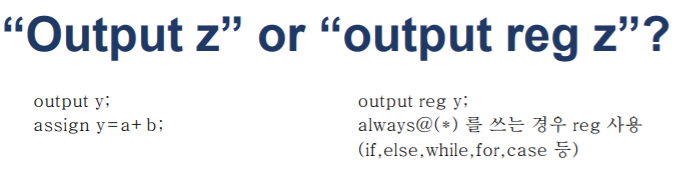
숫자 시스템

Sign and Magnitude : 알아보기 쉽지만 +- 계산시 비효율적

1s Complement : 0이 두번 사용됨, 0을 넘어가는 계산시 값이 이상하게 나옴

2s Complement : 오버플로우 시 음수로 넘어갈 수 있다. 거의 모든 곳에서 2의보수 사용

CH.7



assign문 에서는 reg 쓰지 않는다.

하드웨어는 초기화가 안된다.

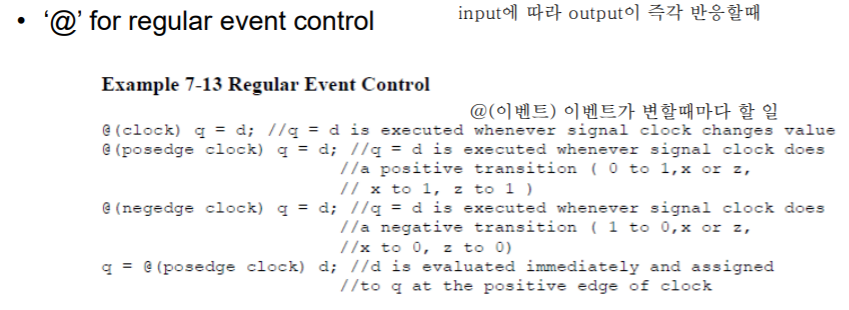
Assign문 안의 변수 중 하나라도 값이 바뀌면 그 즉시 결과값이 변한다.

= 연산자는 차례대로 실행

<= 연산자는 동시에 실행

Delay 베이스 타이밍 컨트롤 : # 사용 시간 조절

Event 베이스 타이밍 컨트롤 : @ 사용





엣지가 아닌 레벨 기준 컨트롤 시 wait 명령어 사용

case문에서 변수가 정상적인 값이 아닐 경우를 대비해서 default 동작 설정

casez 문에서 10XZ 는 default로 가지만 (모든 z값이 돈케어)

casex 에서는 10XZ 는 1XXX로 간다. (모든 x,z값이 돈케어)

루프문

For

While

Repeat() : 괄호안에는 숫자만 사용 가능 , 지정된 변수도 가능

Forever : 컨트롤 어려워서 잘안씀

Sequential blocks (순차적 진행) : begin / end 사용

Parallel blocks (동시 진행) : fork / join 사용

모듈안에 모듈은 안되지만 블록안 블록은 가능

블록 이름 지정

begin : block1

반복문 탈출

disable block1;