**CH.12**

UDP (user-defined primitive)

* Combinational UDPs – when output is solely determined by combinational logic
* Sequential UDPs – when output takes current inputs and current outputs for next outputs (e.g., latches and flipflops)

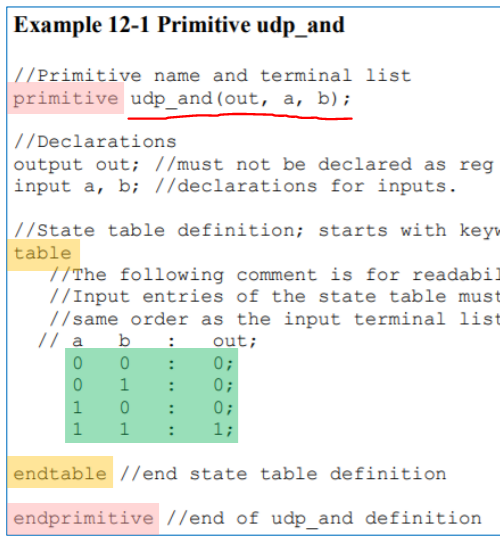
UDP 규칙

* 스칼라 인풋(1bit), 멀티 인풋터미널도 가능하긴 함 (스칼라1bit짜리 여러 개 가능)

Input a,b,c (이렇게 써야함), input[3:0] (불가능)

* 스칼라 아웃풋(1bit), 멀티 아웃풋터미널은 안됨, (output 은 1bit 하나만 써야함)
* 아웃풋터미널은 output 으로 선언 , 시퀸셜은 output reg로 선언
* Z는 다루지 않는다. X로 취급됨
* 모듈 안에 선언 불가
* udp설계시 다른 모듈과 구분 위해 앞에 udp\_붙인다.
* 테이블을 만들어서 설계

Udp 예시



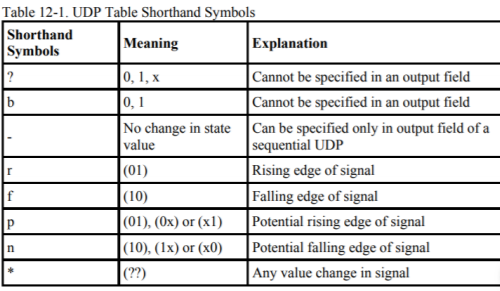
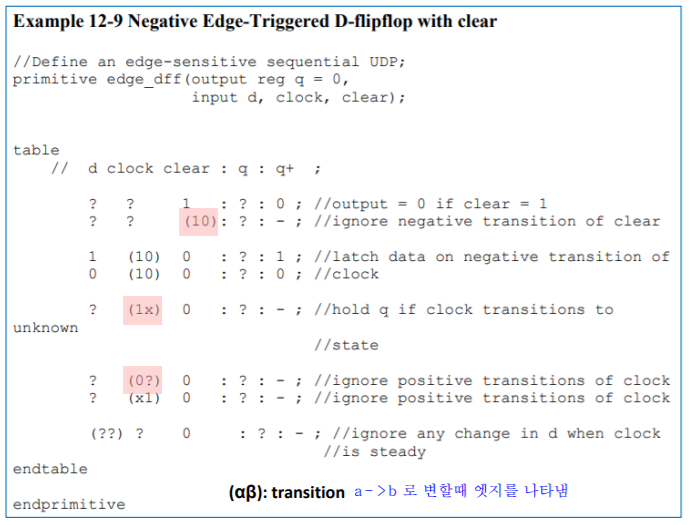
Combinational udp

* 테이블에 없는 인풋인 경우 x출력
* ? 는 돈케어로 취급
* Udp는 꼭 필요한경우 아니면 잘 안쓰인다. 더 쉽게 나타낼 수 있기 때문
* Sequential udp는 안쓰는걸 추천

Sequential udp

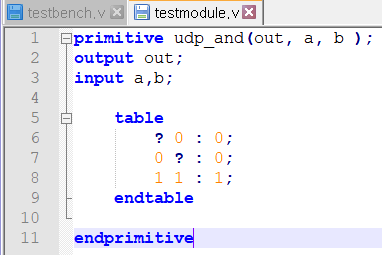
* ‘ - ’ 표시는 그대로 냅둔다는 뜻
* (ab) 표시는 a -> b 로 변할 때의 엣지를 나타냄

예시 (플립플롭)

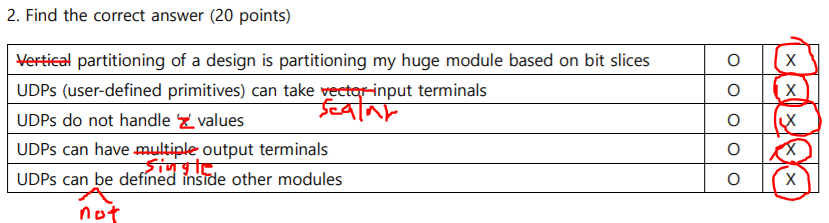


**Quiz 2**

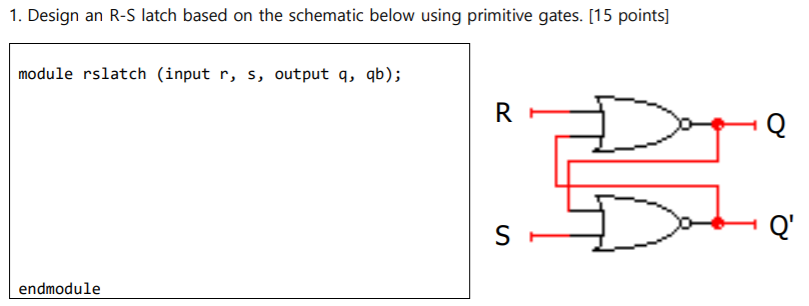
1번. User defined primitive AND gate

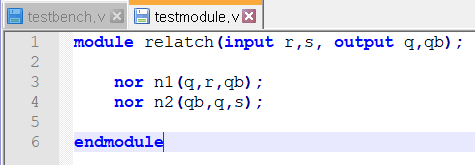
 마지막에 **endprimitive** 주의

2번



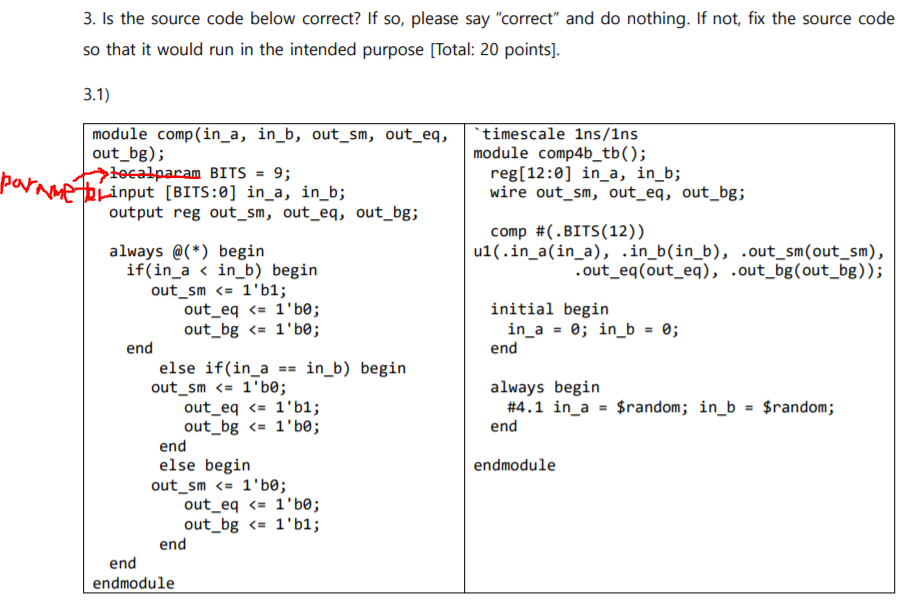
**Final exam**







**Register Transfer Level**



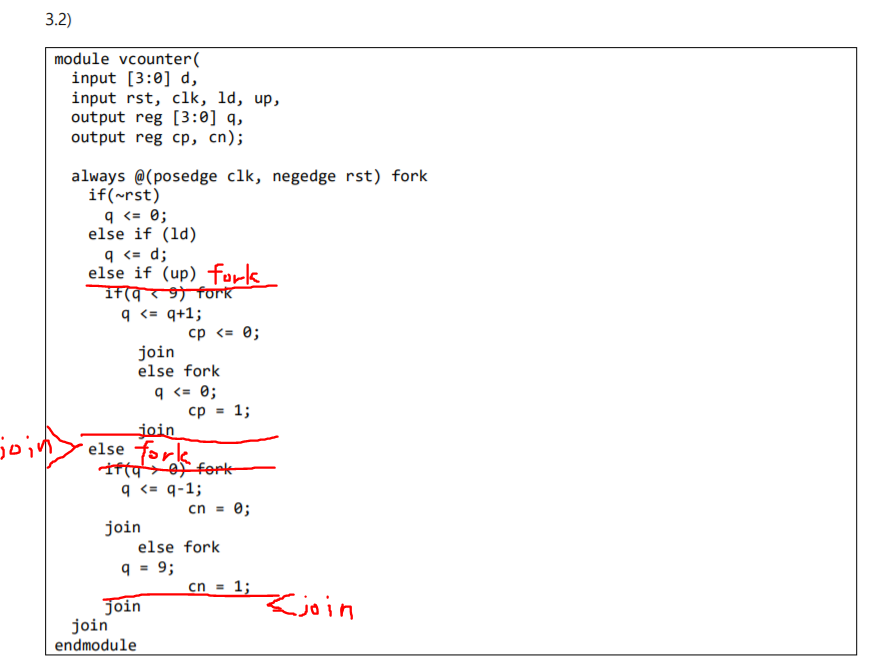
**Localparam은 모듈 밖에서 오버라이딩 할 수 없다.**

* **parameter**로 변경

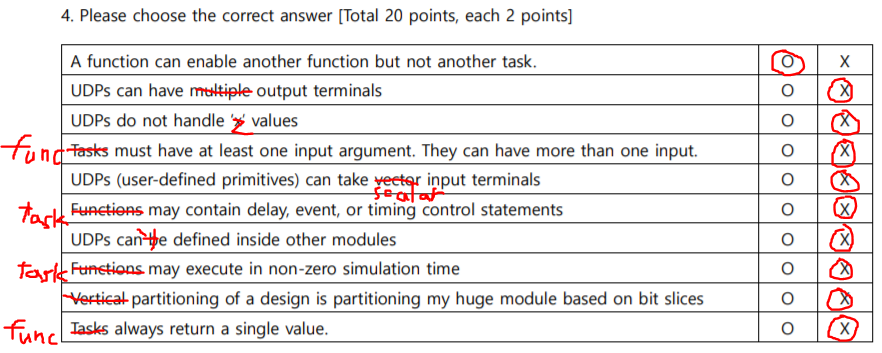
모듈 파라미터 오버라이딩 방법

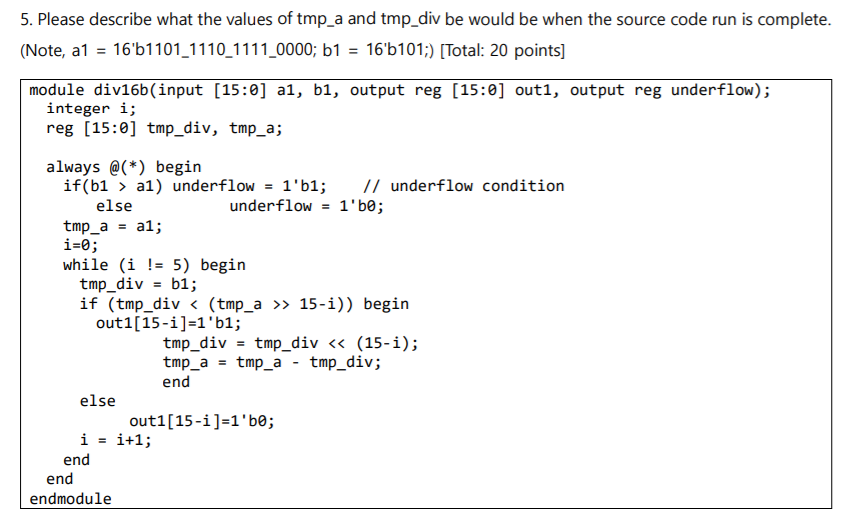
moduleA (.pa(10)) u1 (); //모듈 moduleA의 pa 파라미터에 10값 입력

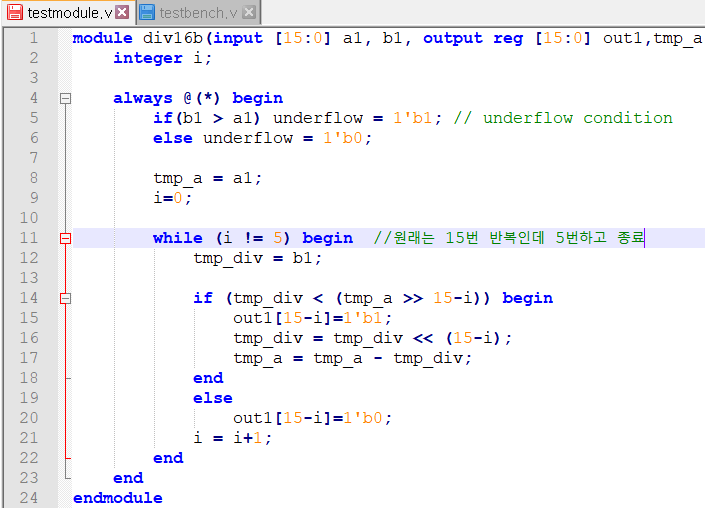
defparam u1.pa = 10; //위와 같음

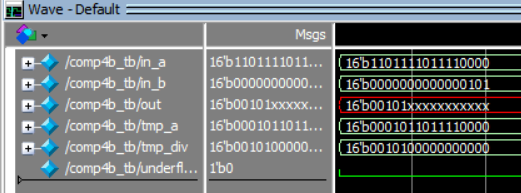


Fork – join , begin – end 잘 묶여있는지 확인하기!



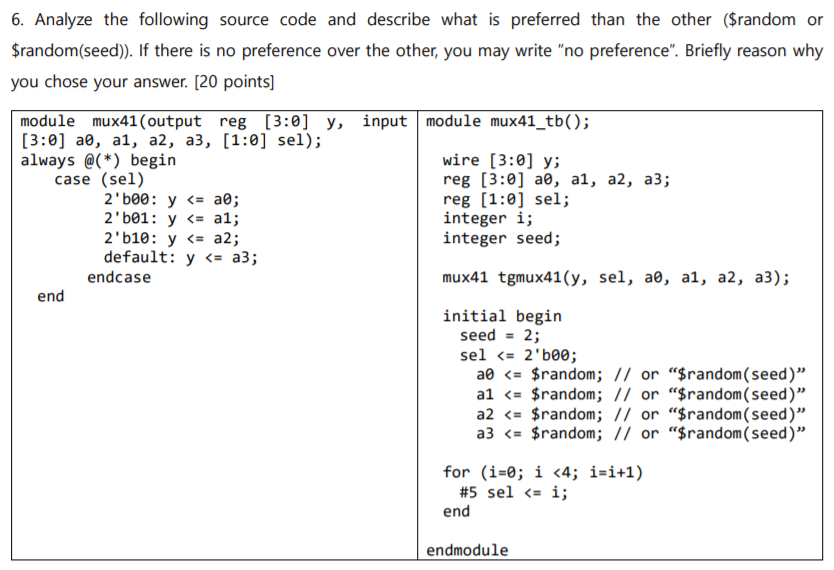






Tmp\_a : 16’b0001\_0110\_1111\_0000

Tmp\_div : 16’b0010\_1000\_0000\_0000 (손으로 잘 계산해나가면 된다!)



$random : 무작위 난수의 집합을 생성, 하지만 난수의 집합이 변하지 않기 때문에 프로그램을 다시 실행할 경우 같은 난수가 나오게 된다.

$random(seed) : 시드값에 따라 다른 난수의 집합을 생성, 프로그램을 실행할 때 마다 다른 난수값을 얻고 싶은경우 시드값만 바꿔 주면 되기 때문에 $random보다 더 좋다.



**$strobe 명령어 사용**