Vectorr

vector3 Previous Next vector4

Given an 8-bit input vector [7:0], reverse its bit ordering.

See also: Reversing a longer vector.

Module Declaration

```
module top_module(
    input [7:0] in,
    output [7:0] out
);
```

Hint...

- `assign out[7:0] = in[0:7];` does not work because Verilog does not allow vector bit ordering to be flipped.
- The concatenation operator may save a bit of coding, allowing for 1 assign statement instead of 8.

설명

1. 주어진 8-bit 벡터를 순서대로 뒤집어라

힌트

`assign out[7:0] = in[0:7];`는 Verilog에서 벡터의 비트 순서를 뒤집어서 할당하려고 할 때 작동하지 않습니다. 왜냐하면 Verilog는 벡터 비트 순서를 뒤집는 것을 허용하지 않기 때문입니다. 그러나 **연결 연산자(Concatenation operator)**를 사용하면 8개의 `assign` 문 대신 1개의 `assign` 문으로 코드를 간결하게 만들 수 있습니다.

상세 설명:

- **벡터 비트 순서**: `in[0:7]`는 `in` 벡터의 첫 번째 비트(0번 비트)부터 마지막 비트(7번 비트)까지를 선택하는 의미입니다. 하지만 Verilog에서는 `assign out[7:0] = in[0:7];`와 같이 비트 순서를 뒤집는 방식으로 직접 할당하는 것이 허용되지 않습니다.
- **연결 연산자(Concatenation)**: 비트 순서를 뒤집어야 할 때는 연결 연산자를 사용할 수 있습니다. 연결 연산자를 사용하면 여러 비트를 결합하여 순서를 바꾸거나, 특정 패턴을 만들 수 있습니다.

이전 문제에서 사용했던 Concatenation을 사용하자.

코드

```
module top_module(
    input [7:0] in,
    output [7:0] out
);
    assign out[7:0] = {in[0], in[1], in[2], in[3], in[4], in[5], in[6], in[7]};
endmodule

```

결과

vectorr — Compile and simulate

Running Quartus synthesis. Show Quartus messages...
Running ModelSim simulation. Show Modelsim messages...

Status: Success!

You have solved 17 problems. See my progress...

Timing diagrams for selected test cases

These are timing diagrams from some of the test cases we used. They may help you debug your circuit. The diagrams show inputs to the circuit, outputs from your circuit, and the expected reference outputs. The "Mismatch" trace shows which cycles your outputs don't match the reference outputs (0 = correct, 1 = incorrect).