13주차 결과보고서

전공: 컴퓨터공학과 학년: 3학년 학번: 20211547 이름: 신지원

**1.**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

4-bit Shift Register 는 bit 가 한칸씩 옆으로 밀리는 Register 다. 따라서 Verilog 코드도 배열로 구현하여 한 칸씩 옆으로 옮겨질 수 있도록 구현하였다. 중간에 reset 키가 작동할 시엔 0으로 초기화되는 if 문을 구현하였다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

Simulation을 살펴보면 d의 값이 주어지면 아래서(0번째 배열)부터 똑같은 주기로 밀리는 현상이 나타나는 것을 볼 수 있다. Reset 값이 1로 활성화되었을 때는 어떠한 값도 1을 출력하지 않고 0을 유지하는 것도 나타난다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Clock number | Output | | | | |
|  | IN | L1 | L2 | L3 | L4 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 0 | 0 |
| 3 | 0 | 0 | 1 | 0 | 0 |
| 4 | 0 | 0 | 0 | 1 | 0 |
| 5 | 1 | 0 | 0 | 0 | 1 |
| 6 | 1 | 1 | 0 | 0 | 0 |
| 7 | 0 | 1 | 1 | 0 | 0 |

위는 Simulation 의 결과를 표로 작성한 값이다. Clock 신호에 따라 입력이 한 칸씩 밀려 출력되는 것을 볼 수 있다.

**2.**

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명

4-bit Ring counter 는 ring 처럼 회전하는 counter 를 말한다. 따라서 초기값을 1로 잡아주었으며 shift register 와 마찬가지로 옆으로 한칸씩 옮겨주었다는 것은 같지만, = 이 아닌 <= 을 사용하여 non-blocking 으로 코드를 구현해 배열의 움직임이 병렬로 실행되도록 하였다. 그 이유는 out[3] 의 경우엔 0에서 받은 값으로 바뀌어야 하면서 2에게 값을 주어야 하기 때문에 nonblocking 방식을 선택하였다.

스크린샷이(가) 표시된 사진

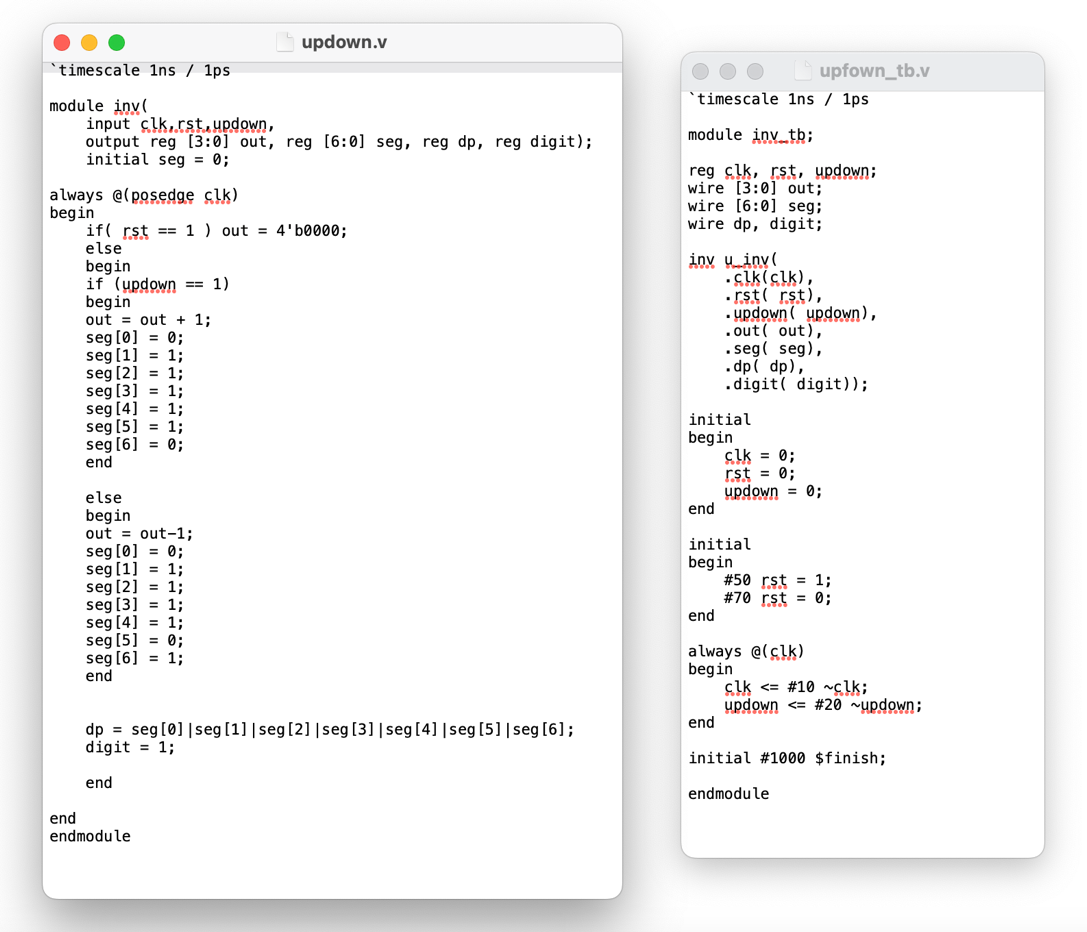
자동 생성된 설명

Simulation을 살펴보면 처음 주어진 0001의 값이 0에서부터 1, 2 로 올라가다 reset 버튼이 활성화 되어 다시 0001 로 초기화되는 것을 볼 수 있다. 그 뒤에 다시 ring 처럼 1의 값이 계속 해서 일정하게 도는 모습을 볼 수 있다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clock number | Output | | | |
|  | L1 | L2 | L3 | L4 |
| 1 | 1 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 0 | 1 | 0 |
| 4 | 0 | 0 | 0 | 1 |
| 5 | 1 | 0 | 0 | 0 |
| 6 | 0 | 1 | 0 | 0 |
| 7 | 0 | 0 | 1 | 0 |

위는 Simulation 의 결과를 표로 작성한 값이다. Clock 신호에 따라 입력이 한 칸씩 밀리며 반복적으로 출력되는 것을 볼 수 있다.

**3.**



Up/Down counter 는 CLOCK 입력에 맞춰 카운터의 값이 증가하거나 감소하는 카운터를 말한다. 따라서 updown 키가 1이라면 clock 신호에 맞춰 out 값을 증가시키며 segment 는 ‘U’를 출력하도록 구현하였다. Updown 키가 1이 아니라면 down 을 의미하기 때문에 clock 신호에 맞춰 out 값을 감소시키며 segment 는 ‘d’를 출력하도록 구현하였다. 또한, reset 키가 1일 경우엔 초기화 해주는 if 문으로 시작하여 분기처리 해주는 코드를 작성하였다.

스크린샷, 라인, 텍스트이(가) 표시된 사진

자동 생성된 설명

Updown 의 값에 따라 out 값이 0,f 를 오가는 것을 볼 수 있다. 또한 reset 이 활성화 되었을 때는 clock 신호가 활성화되더라도 0으로 초기화 되는 것을 볼 수 있다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clock number | Ouput (updown = 1) | | | |
|  | L1 | L2 | L3 | L4 |
| 1 | 0 | 0 | 0 | 0 |
| 2 | 1 | 0 | 0 | 0 |
| 3 | 0 | 1 | 0 | 0 |
| 4 | 1 | 1 | 0 | 0 |
| 5 | 0 | 0 | 1 | 0 |
| 6 | 1 | 0 | 1 | 0 |
| 7 | 0 | 1 | 1 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clock number | Ouput (updown = 0) | | | |
|  | L1 | L2 | L3 | L4 |
| 1 | 1 | 1 | 1 | 1 |
| 2 | 0 | 1 | 1 | 1 |
| 3 | 1 | 0 | 1 | 1 |
| 4 | 1 | 1 | 0 | 1 |
| 5 | 0 | 1 | 0 | 1 |
| 6 | 1 | 0 | 0 | 1 |
| 7 | 0 | 0 | 0 | 1 |

위는 updown 값이 각각 0과 1일 때를 나누어 결과를 나타낸 표다.

**4.**

4-bit Shift Register, 4-bit Ring counter, 4-bit Up/Down counter 의 코드를 구현하고 simulation 으로 확인하였다. 4-bit Up/Down counter는 segment 까지 표기하는 방식으로 구현하였다. Nonblocking 의 방식을 더 정확히 알아볼 필요가 있을 것 같다.

**5.**

그레이코드에 대해 알아보고자 한다. 그레이 코드란 이웃하는 숫자와 한 비트만 다른 코드를 의미한다. 그레이 코드 카운터는 동시에 하나의 비트만 변하지 않게 코드를 할당하는 카운터를 말한다.

원, 도표, 스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명텍스트, 번호, 스크린샷, 달력이(가) 표시된 사진

자동 생성된 설명

위는 그레이 코드 상태도와 여기표다.