12주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211522 이름: 김정환

**1.**

.................

2-bit counter를 Verilog 코드로 구현하면 아래와 같다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이는 input으로 clk, reset, x가 주어지고, output으로 2 bit array의 Q, z가 존재한다. Reset이 1일 때 Q가 00으로 초기화되고, z가 0으로 초기화된다. 그 후 x가 1이고 clk가 Rising edge일 때 Q가 1씩 증가하고, 11에 도달 시 z가 1로 출력되고 다시 Q가 00으로 돌아가는 형태이다. 이를 확인하기 위한 Testbench 코드는 아래와 같다.

텍스트, 영수증, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이를 이용한 Simulation 결과는 아래와 같다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이를 확인하면 처음 reset이 1이 되고 clk가 Rising edge였을 때 Q와 z가 각각 00과 0으로 초기화되는 것을 볼 수 있고, 이 후 x가 1이 되고 clk가 Rising edge인 때마다 증가하여 01, 10, 11이 되고 다시 00으로 돌아가는 것과 z가 Q가 00이 될 때 1이 되는 것도 확인할 수 있다. 해당 코드의 Schematic을 구하면 아래와 같다.

도표, 라인, 텍스트, 평면도이(가) 표시된 사진

자동 생성된 설명

이를 상태도로 나타내면 아래와 같이 나타낼 수 있다.

도표, 원, 스크린샷, 라인이(가) 표시된 사진

자동 생성된 설명

여기서 q0 = 00, q1 = 01, q2 = 10, q3 = 11이라고 할 수 있고, 이를 바탕으로 상태표를 나타내면 아래와 같이 나타낼 수 있다.

|  |  |  |
| --- | --- | --- |
| Current state  qi | Next state/output x/z | |
| x = 0 | x = 1 |
| q0=00 | 00/0 | 01/0 |
| q1=01 | 01/0 | 10/0 |
| q2=10 | 10/0 | 11/0 |
| q3=11 | 11/0 | 00/1 |

................

**2.**

.......................

4-bit decade counter를 Verilog로 구현하면 아래와 같은 코드로 나타낼 수 있다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이는 input으로 x, reset, clk가 주어지며, output으로 4 bit array인 Q, z가 존재한다. reset이 1이고 clk가 Rising Edge일 때는 Q가 0000으로 초기화되고 z가 0으로 초기화된다. 이외에 x가 1이고 clk가 Rising Edge일 때 Q가 1001까지 1씩 증가 후, 0000으로 반복되고, 1001에서 0000이 될 때 z가 1이 된다. 결과를 확인하기 위한 Testbench 코드는 다음과 같다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Simulation 결과는 아래와 같다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이를 살펴보면 reset이 1이고 clk가 Rising edge일 때 Q가 0000으로, z가 0으로 초기화되는 것을 볼 수 있고, 다음으로 순서대로 0000, 0001, 0010, … , 1001까지 clk가 Rising edge이고 x가 1일 때 1씩 올라가는 것을 확인할 수 있고 다시 0000이 될 때 z가 1이 된다. 이를 Schematic으로 나타낸 것은 아래와 같다.

도표, 라인, 평면도, 기술 도면이(가) 표시된 사진

자동 생성된 설명

이는 상태도로 나타내면 아래와 같이 나타낼 수 있다. 해당 상태도에는 z가 생략되어 있으나 상태표에는 포함하였다.

도표, 원, 스케치, 화이트이(가) 표시된 사진

자동 생성된 설명

이를 바탕으로 상태표로 나타내면 아래와 같다.

|  |  |  |
| --- | --- | --- |
| Current state  qi | Next state/output x/z | |
| x=0 | x=1 |
| q0 | 0000/0 | 0001/0 |
| q1 | 0001/0 | 0010/0 |
| q2 | 0010/0 | 0011/0 |
| q3 | 0011/0 | 0100/0 |
| q4 | 0100/0 | 0101/0 |
| q5 | 0101/0 | 0110/0 |
| q6 | 0110/0 | 0111/0 |
| q7 | 0111/0 | 1000/0 |
| q8 | 1000/0 | 1001/0 |
| q9 | 1001/0 | 0000/1 |

다음으로 FPGA에서 구현 시에는 다음과 같은 constraint를 사용하였다.

텍스트, 스크린샷, 폰트, 바이올렛색이(가) 표시된 사진

자동 생성된 설명

또한 reset에 연결한 버튼은 Active Low가 기본 설정이었으므로 위의 design 코드에서 reset에 대해서 ~을 붙이는 방식으로 수정하여 사용하였다. 보드에서 확인하였을 때 위의 Simulation이나 상태표에서 나타낸 결과가 나오는 것을 확인했다.

.........................

**3.**

.......................

4-bit 2421 decade counter를 Verilog로 구현하면 아래와 같은 코드로 나타낼 수 있다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이는 6주차 실습에서 진행한 BCD to 2421 converter를 이용하여 구현하였는데, 앞에서 실습에 사용한 decade counter의 Q를 4 bit array의 res에 BCD to 2421 converter로 변환하여 넣어주도록 구현하였다. 이를 Simulation할 Testbench 코드는 아래와 같다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Simulation 결과는 아래와 같다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

각각을 살펴보면 reset이 1이고 clk가 Rising edge일 때 Q가 0000, z가 0으로 초기화된다. 이 후에 x가 1이고 clk가 Rising edge일 때 Q는 0001, 0010, 0011, 0100까지는 이전의 decade counter와 동일하지만, 2421 BCD 코드를 사용하므로 이후부터는 1011, 1100, 1101, 1110, 1111로 Q가 증가하고, 0000으로 되돌아갈 때 z가 1이 되는 것을 확인할 수 있다. 이를 Schematic으로 나타내면 아래와 같다.

도표, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명

이는 상태도로 나타냈을 때는 2번의 decade counter와 구조는 같으나 상태의 값만 바뀐 형태가 된다.

이를 상태표로 나타내면 아래와 같다.

|  |  |  |
| --- | --- | --- |
| Current state  qi | Next state/output x/z | |
| x=0 | x=1 |
| q0 | 0000/0 | 0001/0 |
| q1 | 0001/0 | 0010/0 |
| q2 | 0010/0 | 0011/0 |
| q3 | 0011/0 | 0100/0 |
| q4 | 0100/0 | 1011/0 |
| q5 | 1011/0 | 1100/0 |
| q6 | 1100/0 | 1101/0 |
| q7 | 1101/0 | 1110/0 |
| q8 | 1110/0 | 1111/0 |
| q9 | 1111/0 | 0000/1 |

이를 FPGA 보드에서 결과를 확인할 때는 아래와 같은 constraint를 사용하였다. Reset 버튼이 Active low이므로 보드에서 실습 시만 reset에 ~를 통해 값을 반대로 하고 진행하였다.

텍스트, 스크린샷, 폰트, 바이올렛색이(가) 표시된 사진

자동 생성된 설명

FPGA에서도 Simulation과 동일하게 작동하는 것을 확인할 수 있었다.

.........................

**4.**

.......................

본 실험에서는 2-bit counter, 4-bit decade counter, 4-bit 2421 decade counter로 총 3가지의 decade counter의 동작을 이해하고 Verilog로 구현하였다. 이 때 4bit의 array로 구현한 Q에 1씩 더하는 것을 통하여 각 상태에 맞는 값을 가지도록 구현하였고, 처음으로 posedge를 사용하여 구현하였다. 이는 값이 변화하는 지점에서만 동작하도록 구현할 수 있도록 하는데, posedge의 경우에는 Rising edge에 동작하도록 한다. 또한 clock과 reset은 FPGA 보드에서 사용할 때 버튼을 사용하였는데, 이는 Active low로 정해져 있어 Simulation으로 확인할 때와 다르게 수정이 필요했다.

이외에 검토해볼 사항으로는 Counter에 대해서 조사할 때 Flip-Flop을 연결하여 구현하는 것에 대한 내용을 찾을 수 있었는데, Verilog로 구현 시에 지난 주차에서 실습한 것처럼 Flip-Flop을 구현한 후에 연결하는 방식으로 구현하는 방식에 대해서 추가로 알아볼 수 있을 것이다.

.........................

**5.**

.......................

이외의 Counter의 종류 중에서 추가로 조사하였을 때 존슨 카운터에 대해 알 수 있었다. 이는 처음 0000부터 시작하여 1로 하나씩 채운 후에, 다시 0으로 채우는 과정을 통해서 나타내는 Counter이다. 이는 회로도로 나타내는 경우 아래와 같이 나타낼 수 있다. 이는 4 bit Johnson Counter의 예시이다.

도표, 평면도, 기술 도면, 직사각형이(가) 표시된 사진

자동 생성된 설명

회로의 특징으로는 마지막 Flip-Flop의 NOT 출력이 첫 Flip-Flop의 입력으로 들어간다는 특징이 있다. 상태를 모두 나타내면, 0000, 1000, 1100, 1110, 1111, 0111, 0011, 0001, 0000, … 순서대로 반복된다. 따라서 4 bit의 경우에는 8개의 상태가 존재하고, n bit의 경우에는 2\*n개의 상태가 존재한다.

.........................

출처

https://blog.naver.com/lagrange0115/220730343069