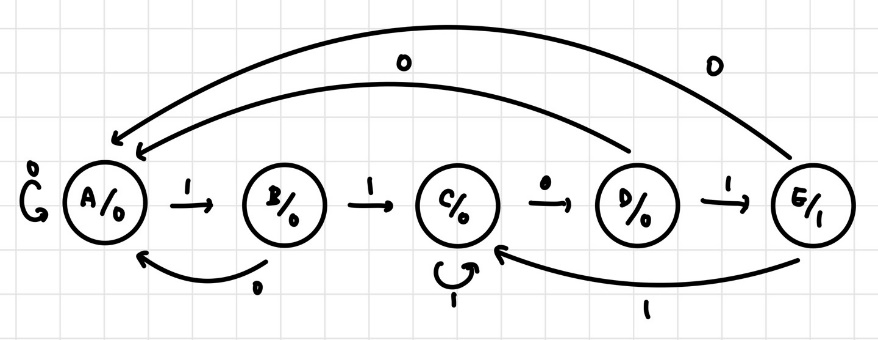
14주차 결과보고서

전공: 컴퓨터공학과 학년: 3학년 학번: 20191599 이름: 송경호

**1.**

다음은 sequence 1101을 overlapping 방식으로 detecting 하는 moore machine에 대한 구현이다. Verilog 코드의 구현에 앞서 state diagram과 state table을 구한다.

먼저 state diagram이다.



Moore machine은 현재 상태에 대해서만 출력이 결정되므로 State/Output 형태로 출력을 구성했다. 1101은 총 4개의 수로 이루어져 있으므로 A부터 아무런 수도 충족되지 않았을 때, 이후 한 수 씩 충족되었을 때를 state로 두어 총 5개의 state가 발생한다. 주의해야 할 점은 C와 E State로 먼저 C State는 두 번의 1이 들어온 상태이다. 이때 만약 1이 한번 더 입력되게 된다면 다시 처음 state로 돌아가는 것이 아닌 다시 C State로 들어와야 한다. 또한 E state는 1101 sequence가 충족된 상태로 해당 state에서 1이 들어오면 역시 C State로 돌아가야 한다.

다음은 state table이다. State Diagram을 바탕으로 구성할 수 있다. 직관적이므로 부연설명은 생략한다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Current State** | **Next State** | | **Output** |
| **Input 0** | **Input 1** |
| A | A | B | 0 |
| B | A | C | 0 |
| C | D | C | 0 |
| D | A | E | 0 |
| E | A | C | 1 |

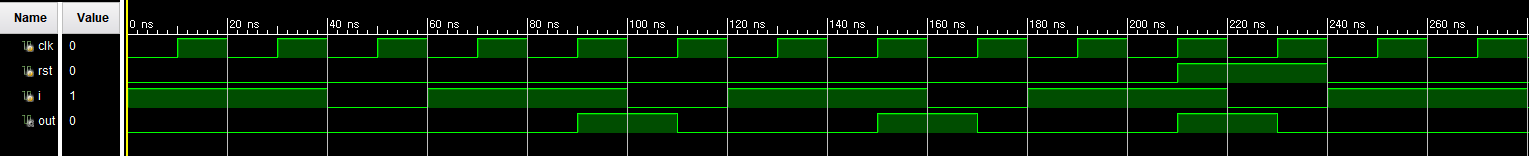
구한 State Table을 바탕으로 verilog의 디자인, 테스트밴치 코드를 작성한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Moore Machine의 경우 실제 수업 때 진행했던 코드에서 blocking을 non blocking문으로 치환해서 구성했다. 이는 State의 변화 이후 출력이 발생하는 Moore Machine의 특성을 고려하면 자명하다. tmp는 길이 4의 배열이며 입력이 tmp[0]으로 들어가고 tmp는 한 칸 씩 밀리게 된다. 이 때 tmp가 1101인 순간에 출력을 발생시켜 구현했다. 다음은 시뮬레이션 결과이다.



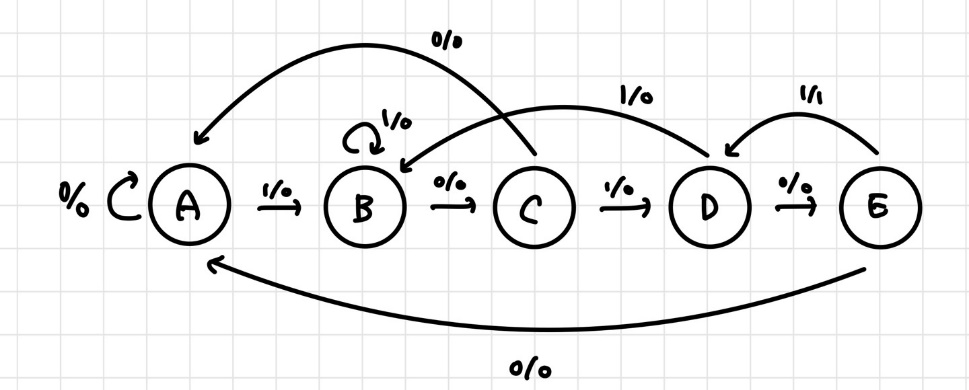
테스트밴치코드에 따라 10ns마다 clock이 진동한다. 이 때 입력이 80ns까지 clock의 pos edge마다 1101로 들어갔으므로 다음 posedge인 90ns에서 1이 출력된다. 이후 다시 101의 입력이 들어가는데 해당 코드는 overlapping을 지원하므로 101이 입력된 상태인 150ns에서 다시 한번 출력이 발생한다. 또한 이후 reset bit를 발생시켜 sequence가 초기화되는 것 역시 확인할 수 있다.

**2.**

10101 sequence에 대한 overlapping 방식의 mealy와 moore machine을 구현한다.

**2-1) Mealy**

먼저 state diagram이다.



Mealy machine은 현재 상태에 입력을 모두 고려하여 출력이 결정되므로 위와 같이 구성했다. 10101은 총 5개의 수로 이루어져 있으므로 A부터 아무런 수도 충족되지 않았을 때, 이후 한 수 씩 충족되었을 때를 state로 두어 총 4개의 state가 발생한다. 즉 마지막 E state는 1010까지 입력된 상태이다.

다음은 state table이다. State Diagram을 바탕으로 구성할 수 있다. 직관적이므로 부연설명은 생략한다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Current State** | **Next State** | | **Output** | |
| **Input 0** | **Input 1** | **Input 0** | **Input 1** |
| A | A | B | 0 | 0 |
| B | C | B | 0 | 0 |
| C | A | D | 0 | 0 |
| D | E | B | 0 | 0 |
| E | A | D | 0 | 1 |

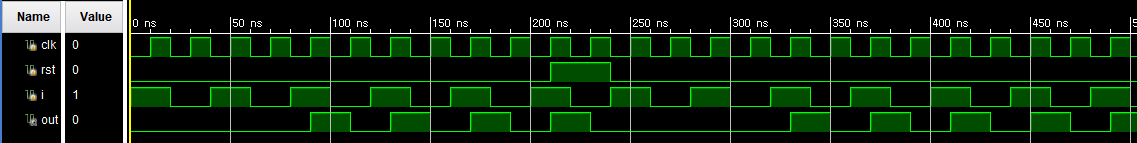
구한 State Table을 바탕으로 verilog의 디자인, 테스트밴치 코드를 작성한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Mealy Machine은 non blocking문을 통해 구현한다. tmp는 길이 5의 배열이며 마찬가지로 입력이 tmp[0]으로 들어가고 tmp는 한 칸 씩 밀리게 된다. 이 때 tmp가 10101인 순간에 출력을 발생시켜 구현했다. 다음은 시뮬레이션 결과이다.



테스트밴치 코드에 따라 10ns마다 clock이 진동한다. 이 때 입력이 90ns까지 clock의 pos edge마다 10101로 들어갔으므로 바로 해당 pos edge에서 출력 1이 발생한다. 이후 계속해서 0과 1이 입력 되는데 해당 코드는 overlapping을 지원하므로 매 두번의 posedge마다 출력 1이 발생한다. 이후 reset 비트에서 다시 초기화 되어 10101의 입력 이후에 1이 발생하는 것을 알 수 있다.

**2-2) Moore**

먼저 state diagram이다.

**도표, 스케치, 라인, 그림이(가) 표시된 사진

자동 생성된 설명**

Moore machine은 이전 mealy machine과 달리 하나의 state가 추가로 발생한다. 이외의 구성은 모두 동일하다.

다음은 state table이다. State Diagram을 바탕으로 구성할 수 있다. 직관적이므로 부연설명은 생략한다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Current State** | **Next State** | | **Output** |
| **Input 0** | **Input 1** |
| A | A | B | 0 |
| B | C | B | 0 |
| C | A | D | 0 |
| D | E | B | 0 |
| E | A | F | 0 |
| F | E | B | 1 |

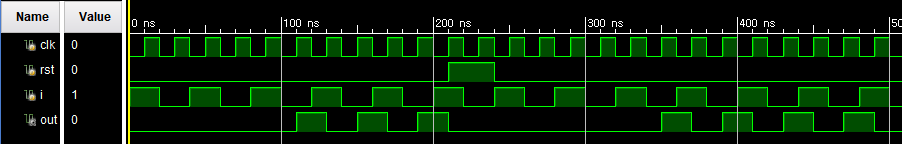
구한 State Table을 바탕으로 verilog의 디자인, 테스트밴치 코드를 작성한다.

텍스트, 스크린샷, 폰트, 대수학이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Moore Machine은 이전 코드의 non blocking문을 blocking문으로 치환하여 구현한다. tmp는 길이 5의 배열이며 마찬가지로 입력이 tmp[0]으로 들어가고 tmp는 한 칸 씩 밀리게 된다. 이 때 tmp가 10101인 순간에 출력을 발생시켜 구현했다. 다음은 시뮬레이션 결과이다.



테스트밴치 코드에 따라 10ns마다 clock이 진동한다. 이 때 입력이 90ns까지 clock의 pos edge마다 10101로 들어갔는데 이전과 달리 다음pos edge인 110ns에서 출력 1이 발생한다. 이후 계속해서 0과 1이 입력 되는데 해당 코드는 overlapping을 지원하므로 매 두번의 posedge마다 출력 1이 발생한다. 이후 reset 비트에서 다시 초기화 되어 10101의 입력 이후에 1이 발생하는 것을 알 수 있다.