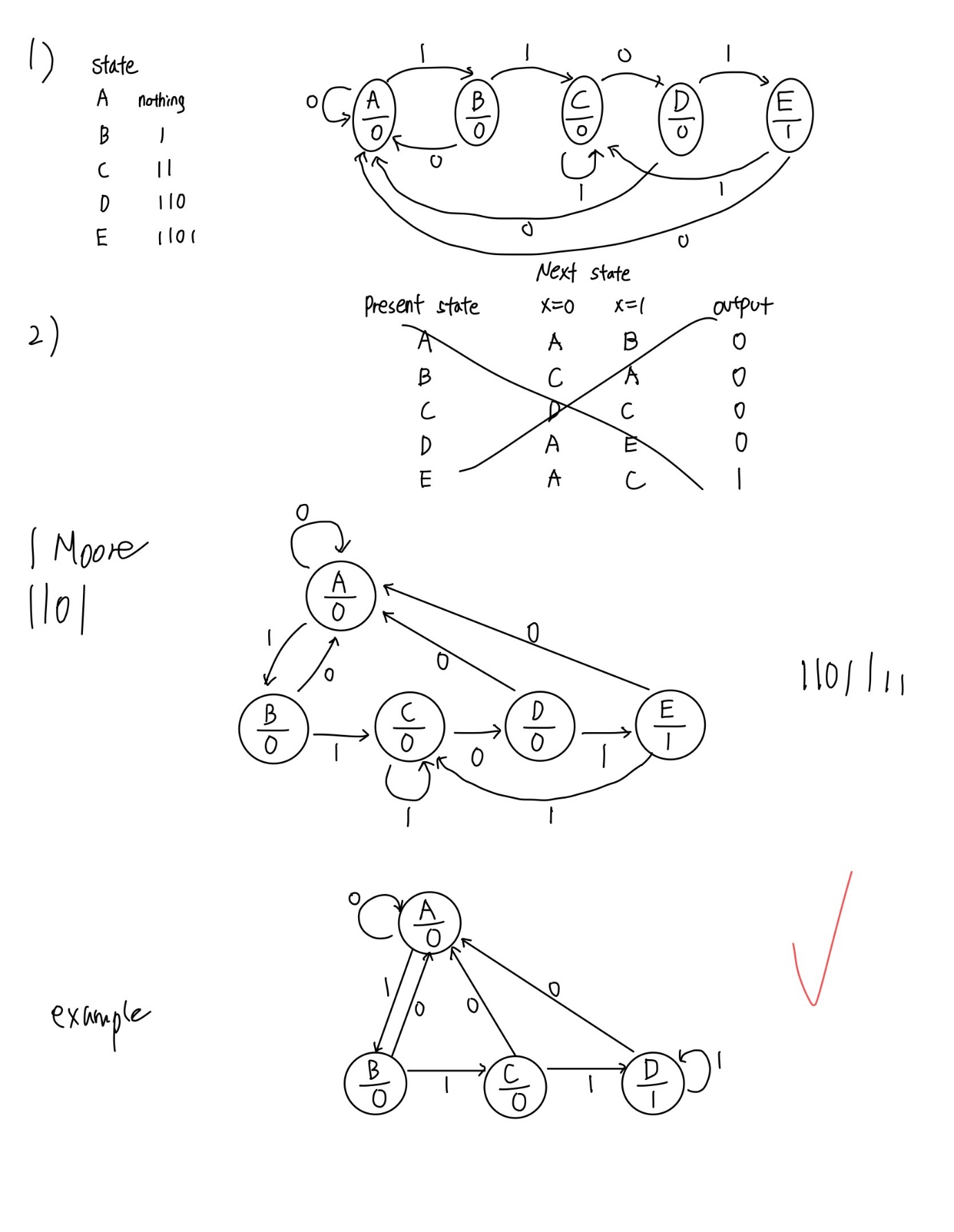
14주차 결과보고서

전공: 영미어문/컴퓨터공학과 학년: 3학년 학번: 20200185 이름: 박정주

**1. Sequence Detector 1101 Moore machine**

상태 A는 패턴이 아무것도 등장하지 않은 상태, B는 1이 등장한 상태, C는 11이 등장한 상태, D는 110이 등장한 상태, E는 1101이 등장한 상태를 의미한다. Moore machine의 출력값은 현재 상태에 의해서만 결정되므로, 현재 상태가 E이면 input 값에 상관없이 output은 1이 된다.

상태도는 다음과 같다.



상태표는 다음과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| Present State | Next State (x=0) | Next State (x=1) | Output |
| A | A | B | 0 |
| B | A | C | 0 |
| C | D | C | 0 |
| D | A | E | 0 |
| E | A | C | 1 |

각 상태에 binary code를 다음과 같이 부여하였다.

A 000

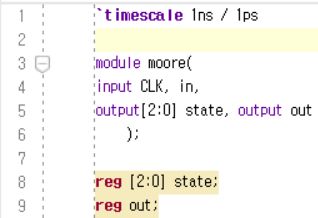
B 001

C 010

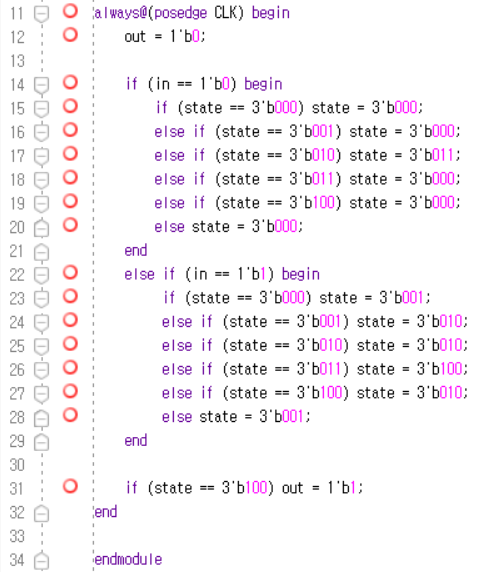
D 011

E 100

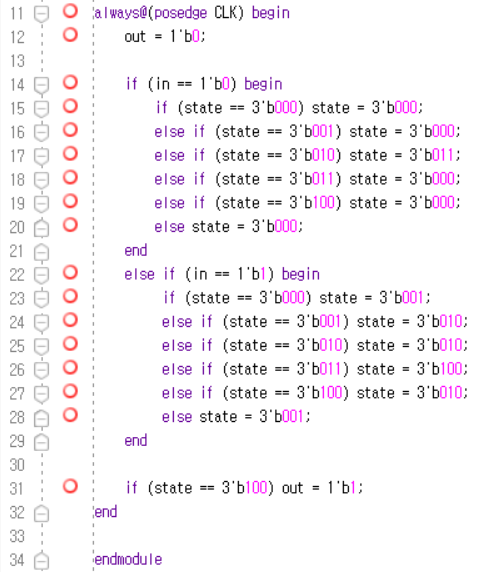
Design source는 다음과 같다.



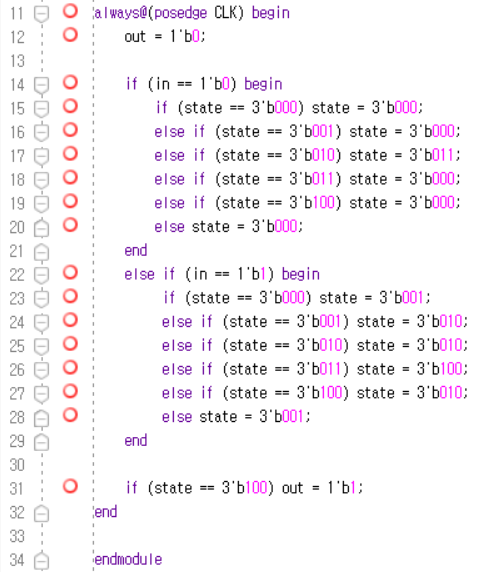
Clock pulse를 의미하는 CLK, 입력되는 값을 의미하는 in을 input으로 선언하였다. A부터 E까지 5개의 상태를 표현하기 위해 3bit 배열 state를 output과 reg로 선언하였다. 상태에 따른 출력값을 의미하는 out을 output과 reg로 선언하였다.



CLK의 값이 0에서 1이 될 때마다 always문이 동작하도록 하였다. 먼저 out을 0으로 초기화한다.

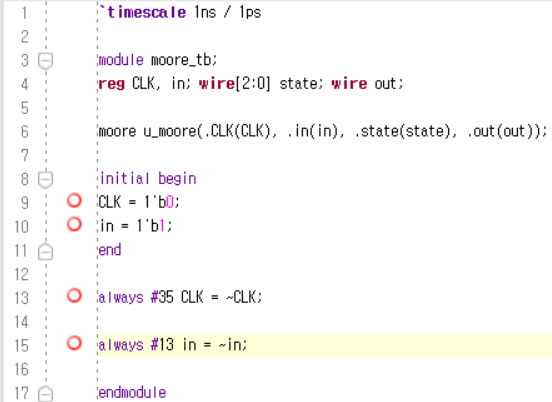


입력 in이 0일 경우, 위에서 작성한 상태표에 따라 상태가 변화하도록 처리한다. 이전 상태가 아무 패턴도 등장하지 않은 A(000)인 경우 다음 상태도 A(000)이 된다. 이전 상태가 C(010)인 경우 패턴이 11까지 등장한 상태이므로, 다음 상태는 110이 등장했음을 의미하는 D(011)이 된다. 마지막 else문을 통해 이전 상태가 존재하지 않을 경우 state를 000으로 초기화한다.

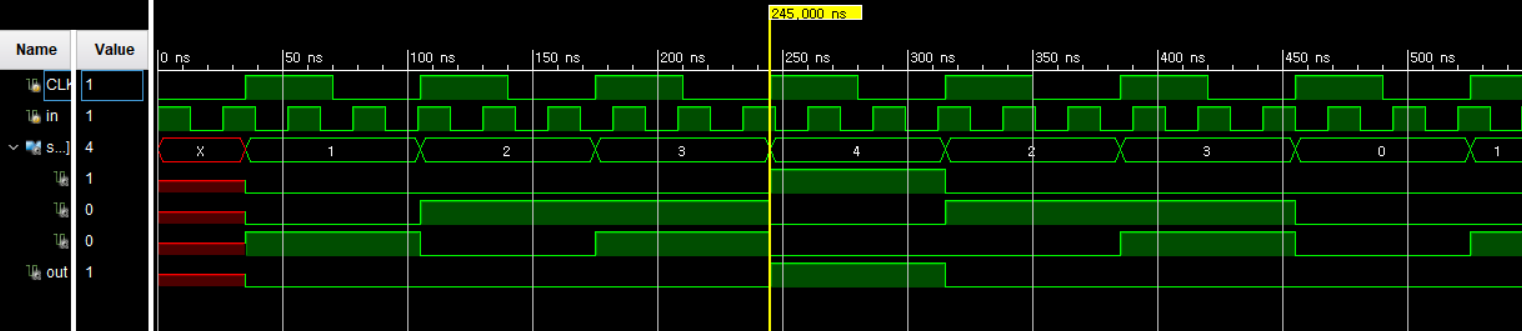


입력 in이 1일 경우도 위에서 작성한 상태표에 따라 상태가 변화하도록 처리한다. 이전 상태가 아무 패턴도 등장하지 않은 A(000)인 경우 다음 상태는 1이 등장했음을 의미하는 B(001)이 된다. 이전 상태가 D(011)인 경우 패턴이 110까지 등장한 상태이므로, 다음 상태는 타겟 패턴인 1101이 등장했음을 의미하는 E(100)이 된다. 마지막 else문을 통해 이전 상태가 존재하지 않을 경우 state를 1이 등장했음을 의미하는 B(001)으로 초기화한다. 이러한 과정을 거친 후 State가 E(100)라면 타겟 패턴이 등장했음을 의미하므로 out을 1로 바꾼다.

Simulation source는 다음과 같다. initial문을 통해 CLK는 0으로, in은 1로 시작하도록 하였다. always문을 통해 35 시간단위가 지날 때마다 CLK의 값을 부정하고, 13 시간단위가 지날 때마다 in의 값을 부정하도록 하였다.



Simulation 결과는 다음과 같다. CLK가 0에서 1로 바뀌는 순간의 입력 in값이 1101의 패턴을 보이자, state는 1001(E), out은 1이 출력된다. Moore machine이 정상적으로 동작하는 것을 확인할 수 있다.

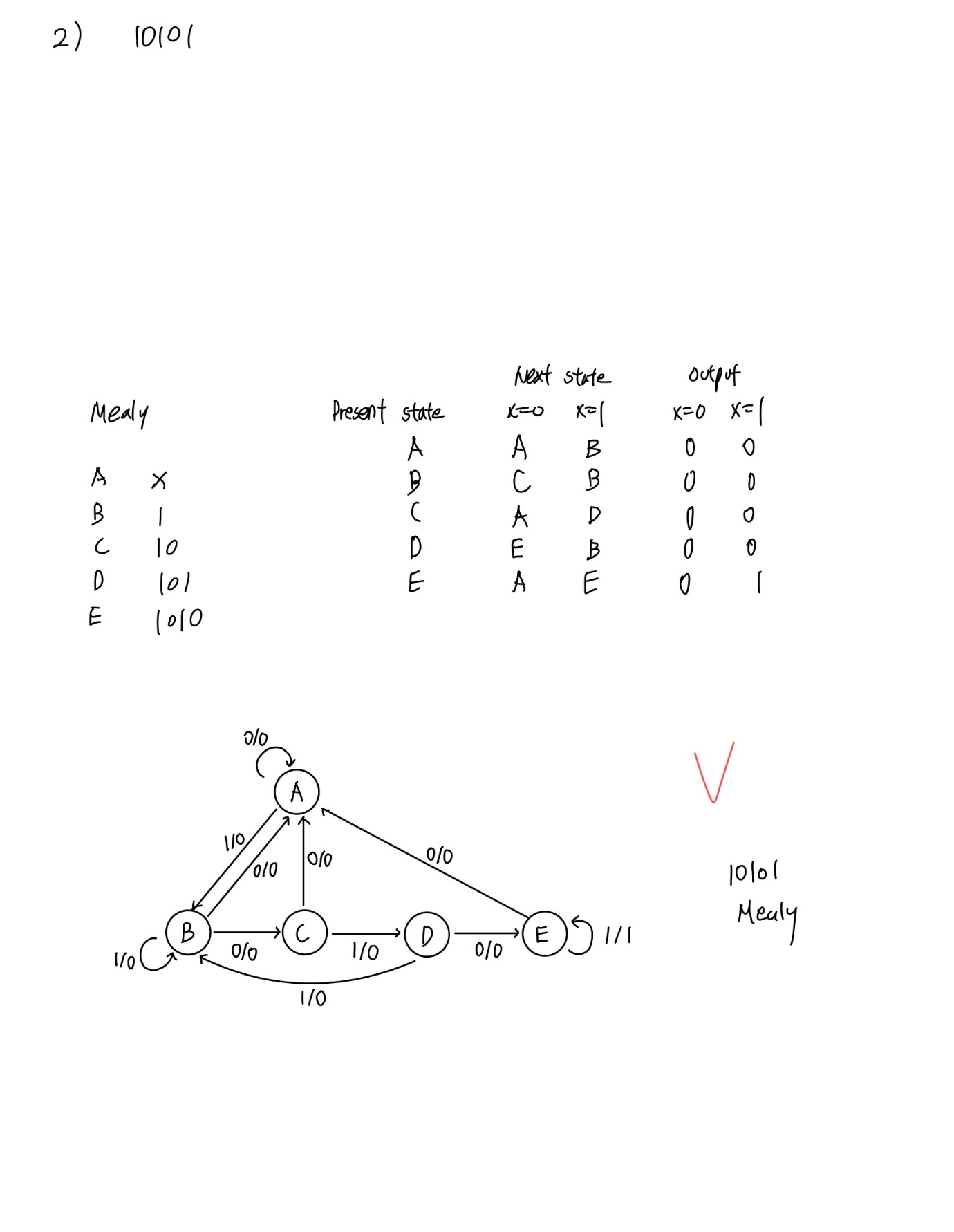


**2. Sequence Detector 10101**

**1) Mealy machine**

상태 A는 패턴이 아무것도 등장하지 않은 상태, B는 1이 등장한 상태, C는 10이 등장한 상태, D는 101이 등장한 상태, E는 1010이 등장한 상태를 의미한다. Mealy machine의 출력값은 현재 상태와 현재 입력에 의해 결정되므로, 현재 상태가 E이고 입력값이 1이면 output은 1이 된다.

상태도는 다음과 같다.



상태표는 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Present State | Next State (x=0) | Next State (x=1) | Output (x=0) | Output (x=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 |

각 상태에 binary code를 다음과 같이 부여하였다.

A 000

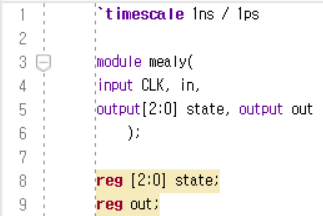
B 001

C 010

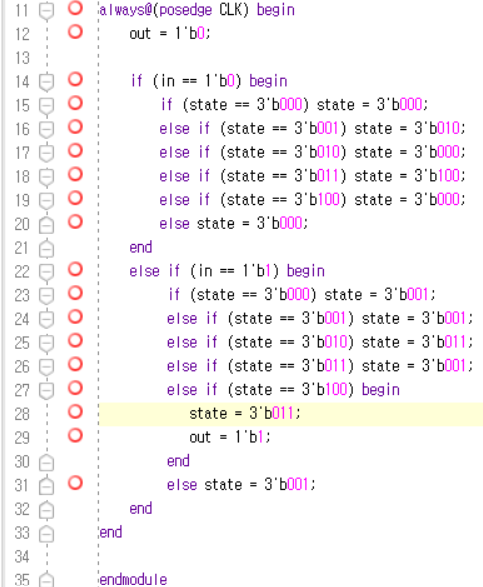
D 011

E 100

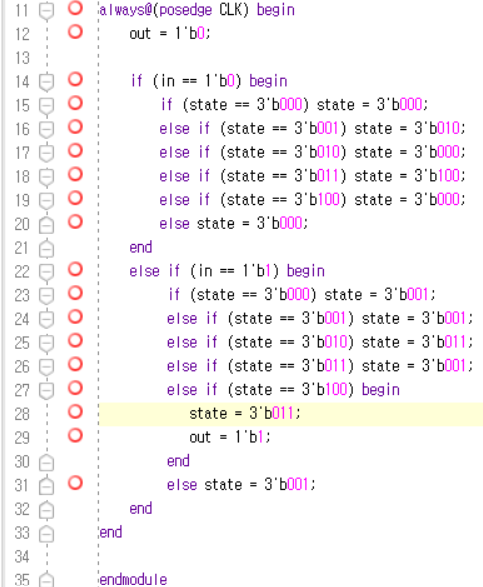
Design source는 다음과 같다.



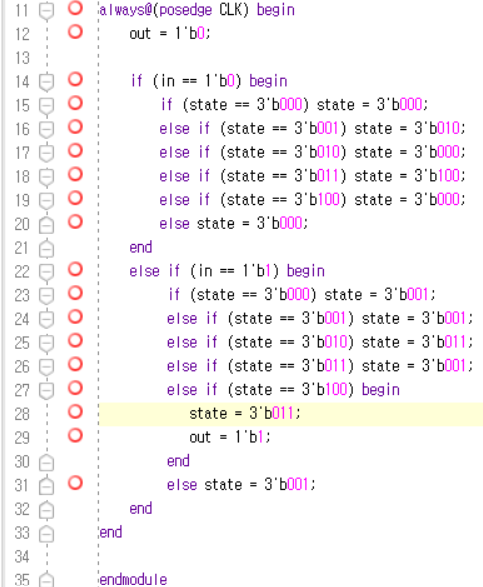
Clock pulse를 의미하는 CLK, 입력되는 값을 의미하는 in을 input으로 선언하였다. A부터 E까지 5개의 상태를 표현하기 위해 3bit 배열 state를 output과 reg로 선언하였다. 상태와 입력에 의해 결정되는 출력값을 의미하는 out을 output과 reg로 선언하였다.



CLK의 값이 0에서 1이 될 때마다 always문이 동작하도록 하였다. 먼저 out을 0으로 초기화한다.

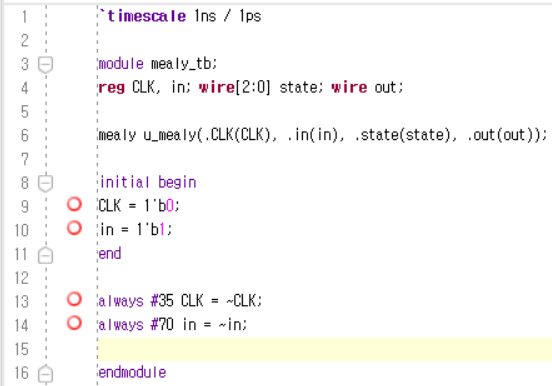


입력 in이 0일 경우, 위에서 작성한 상태표에 따라 상태가 변화하도록 처리한다. 이전 상태가 아무 패턴도 등장하지 않은 A(000)인 경우 다음 상태도 A(000)이 된다. 이전 상태가 D(011)인 경우 101이 등장한 상태이므로, 다음 상태는 1010이 등장했음을 의미하는 E(100)이 된다. 마지막 else문을 통해, 이전 상태가 존재하지 않을 경우 state를 000으로 초기화한다.

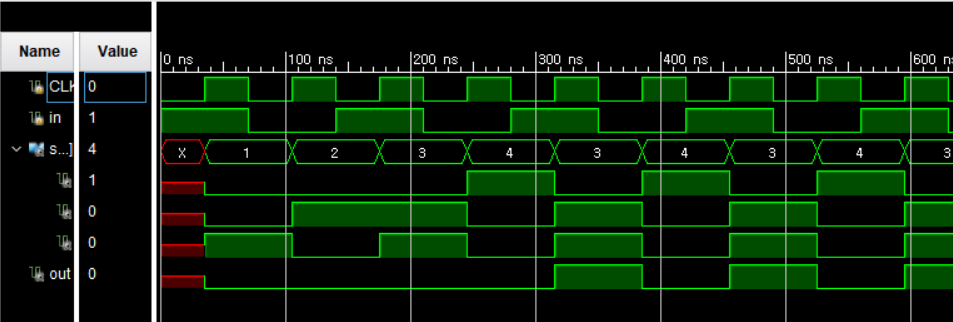


입력 in이 1일 경우도 위에서 작성한 상태표에 따라 상태가 변화하도록 처리한다. 이전 상태가 아무 패턴도 등장하지 않은 A(000)인 경우 다음 상태는 1이 등장했음을 의미하는 B(001)이 된다. 이전 상태가 E(100)인 경우 state=E, in=1을 만족하므로 out의 값을 1로 바꾸고, 다음 상태는 1010이 등장했음을 의미하는 D(011)이 된다. 마지막 else문을 통해, 이전 상태가 존재하지 않을 경우 state를 1이 등장했음을 의미하는 B(001)으로 초기화한다.

Simulation source는 다음과 같다. initial문을 통해 CLK는 0으로, in은 1로 시작하도록 하였다. always문을 통해 35 시간단위가 지날 때마다 CLK의 값을 부정하고, 70 시간단위가 지날 때마다 in의 값을 부정하도록 하였다.



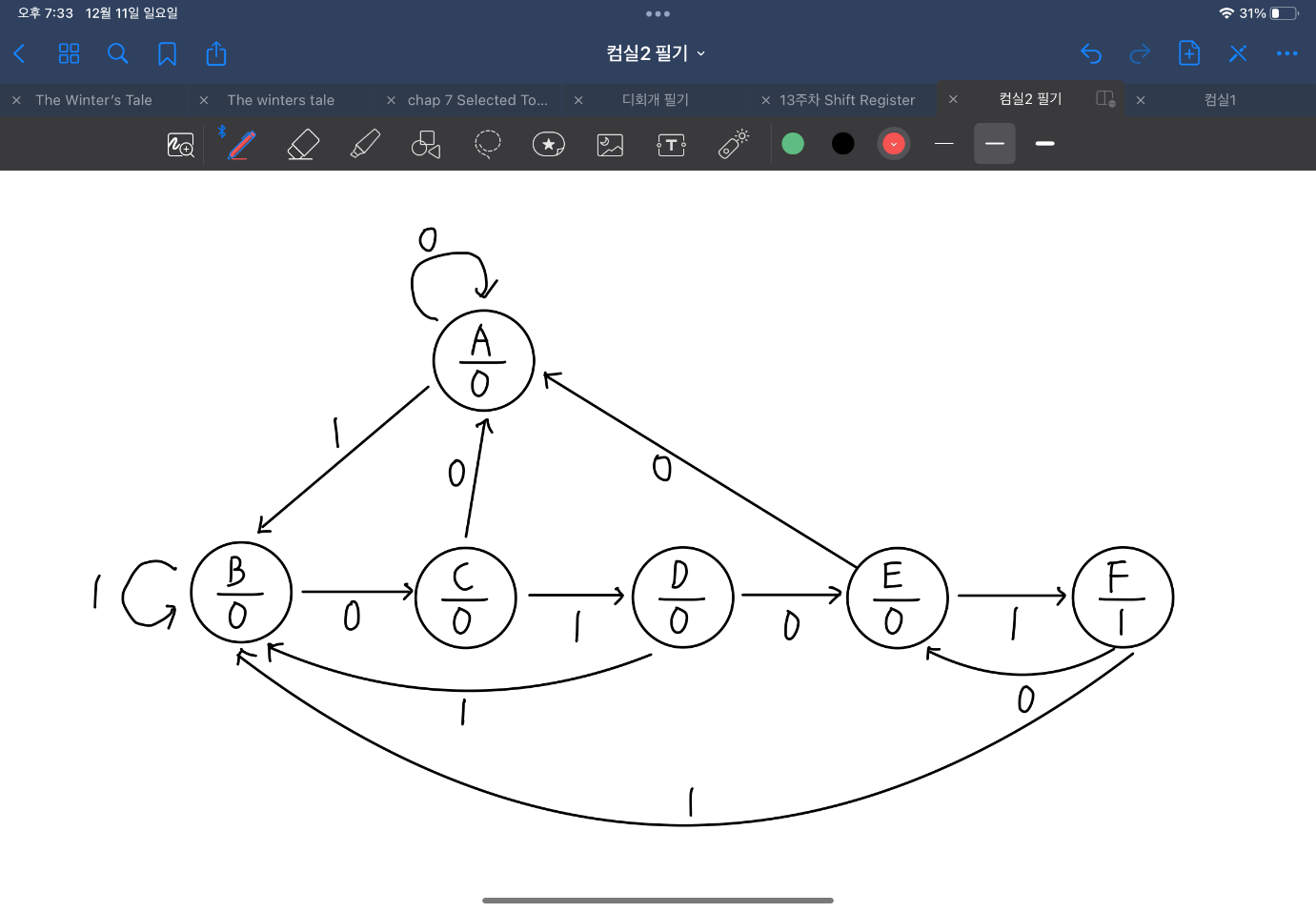
Simulation 결과는 다음과 같다. 1, 0, 1, 0이 등장할 때마다 state는 각각 B(001), C(010), D(011), E(100)으로 변화한다. State가 E일 때 in 값으로 1이 입력되면 out의 값으로 1이 출력되고, 다음 상태인 D(011)로 변화하는 모습을 볼 수 있다. Mealy machine이 정상적으로 작동함을 확인하였다.



**2) Moore machine**

A는 패턴이 아무것도 등장하지 않은 상태, B는 1이 등장한 상태, C는 10이 등장한 상태, D는 101이 등장한 상태, E는 1010이 등장한 상태, F는 10101이 등장한 상태를 의미한다. Moore machine의 출력값은 현재 상태에 의해 결정되므로, 현재 상태가 F이면 output은 1이 된다.

상태도는 다음과 같다.



상태표는 다음과 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| Present State | Next State (x=0) | Next State (x=1) | Output |
| A | A | B | 0 |
| B | C | B | 0 |
| C | A | D | 0 |
| D | E | B | 0 |
| E | A | F | 0 |
| F | E | B | 1 |

각 상태에 binary code를 다음과 같이 부여하였다.

A 000

B 001

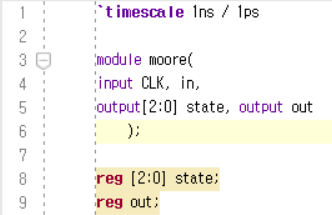
C 010

D 011

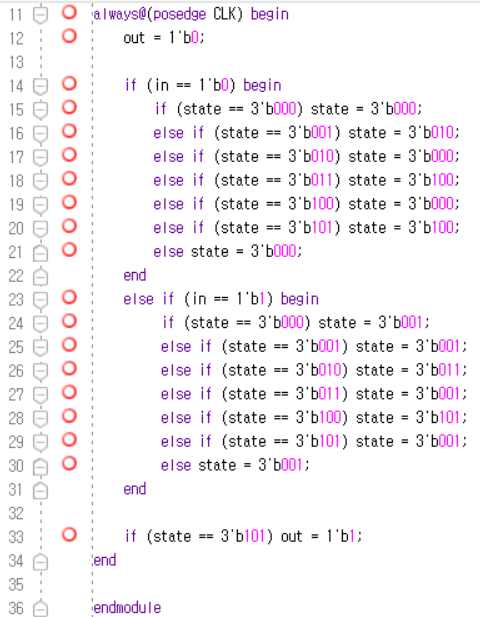
E 100

F 101

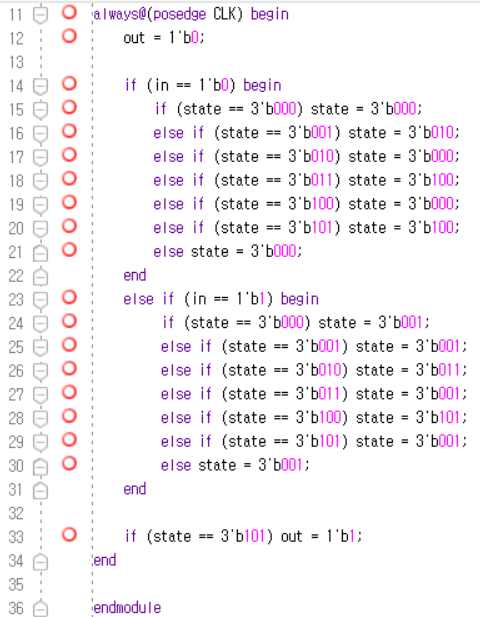
Design source는 다음과 같다.



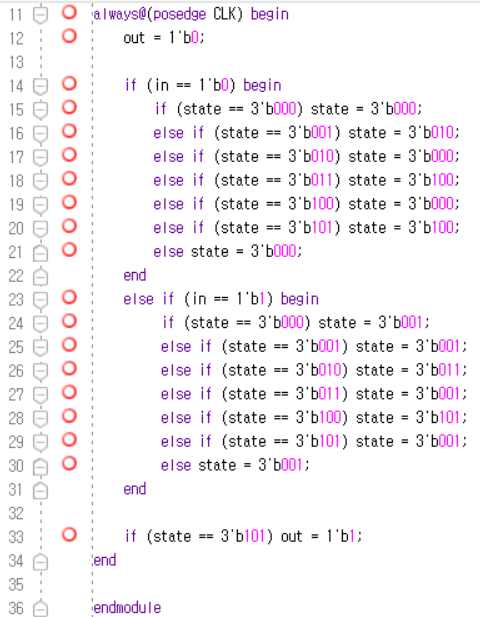
Clock pulse를 의미하는 CLK, 입력되는 값을 의미하는 in을 input으로 선언하였다. A부터 F까지 6개의 상태를 표현하기 위해 3bit 배열 state를 output과 reg로 선언하였다. 상태와 입력에 의해 결정되는 출력값을 의미하는 out을 output과 reg로 선언하였다.



CLK의 값이 0에서 1이 될 때마다 always문이 동작하도록 하였다. 먼저 out을 0으로 초기화한다.

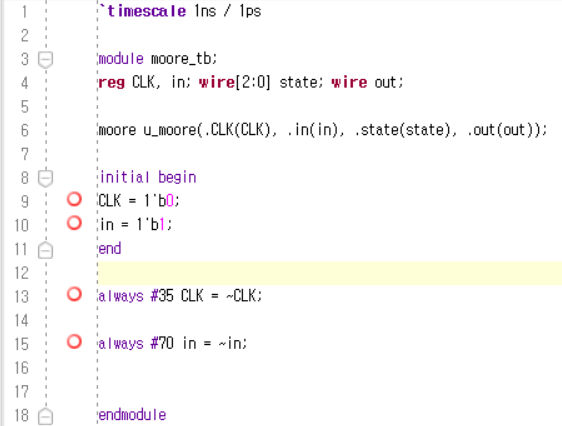


입력 in이 0일 경우, 위에서 작성한 상태표에 따라 상태가 변화하도록 처리한다. 이전 상태가 아무 패턴도 등장하지 않은 A(000)인 경우 다음 상태도 A(000)이 되도록 한다. 이전 상태가 D(011)인 경우 101이 등장한 상태이므로, 다음 상태는 1010이 등장했음을 의미하는 E(100)이 된다. 이전 상태가 F(101)인 경우 10101이 등장한 상태이므로, 입력이 0인 경우 다음 상태는 1010이 등장했음을 의미하는 E(100)이 된다. 마지막 else문을 통해, 이전 상태가 존재하지 않을 경우 state를 A(000)으로 초기화한다.



입력 in이 1일 경우도 위에서 작성한 상태표에 따라 상태가 변화하도록 처리한다. 이전 상태가 아무 패턴도 등장하지 않은 A(000)인 경우 다음 상태는 1이 등장했음을 의미하는 B(001)이 된다. 이전 상태가 1010이 등장했음을 의미하는 E(100)인 경우 다음 상태는 10101이 등장했음을 의미하는 F(101)이 된다. 마지막 else문을 통해, 이전 상태가 존재하지 않을 경우 state를 1이 등장했음을 의미하는 B(001)으로 초기화한다. 만약 상태가 F(101)일 경우, out의 값을 1로 바꾼다.

Simulation source는 다음과 같다. initial문을 통해 CLK는 0으로, in은 1로 시작하도록 하였다. always문을 통해 35 시간단위가 지날 때마다 CLK의 값을 부정하고, 70 시간단위가 지날 때마다 in의 값을 부정하도록 하였다.



Simulation 결과는 다음과 같다. 1, 0, 1, 0, 1이 등장할 때마다 state는 각각 B(001), C(010), D(011), E(100), F(101)으로 변화한다. State가 F(101)가 되면 out의 값으로 1이 출력되고, 다음 in값으로 0이 들어오자 state가 E(100)로 변화하는 모습을 볼 수 있다. Moore machine이 정상적으로 작동함을 확인하였다.

