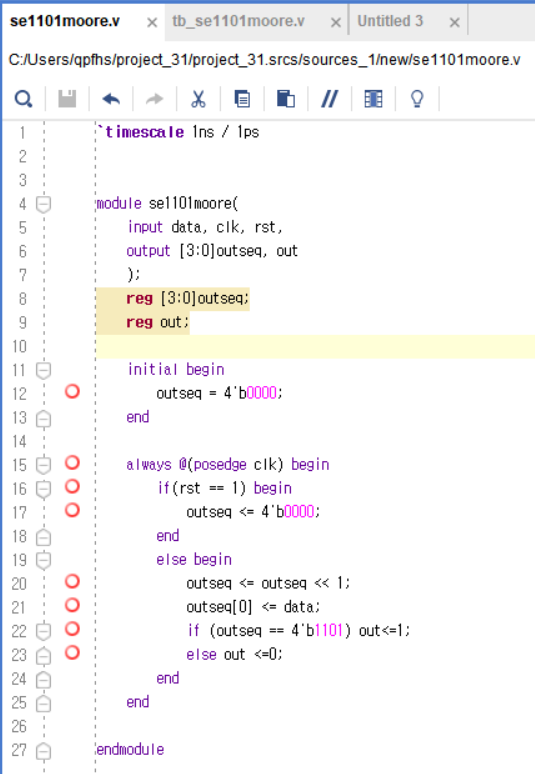
**14주차 결과보고서**

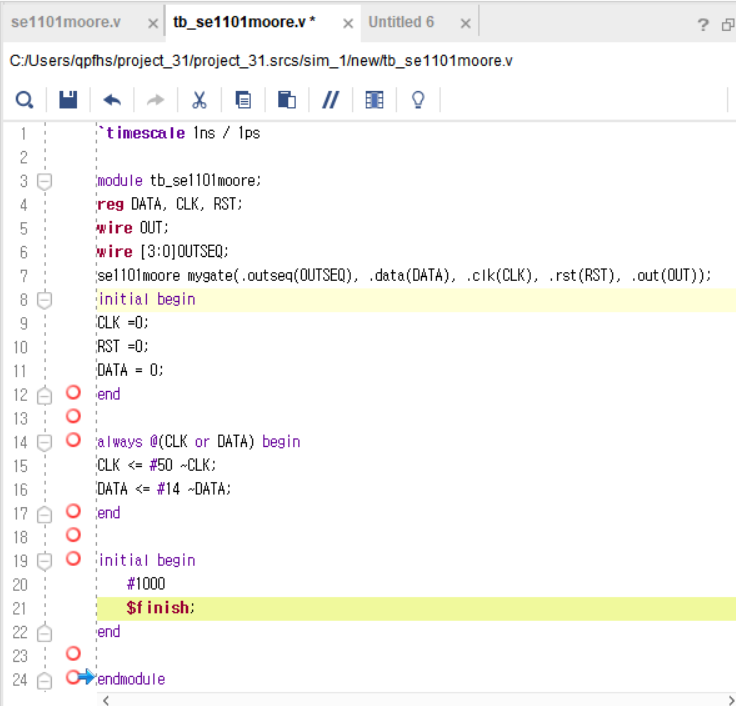
**전공: 생명과학과 학년: 4학년 학번: 20182186 이름: 김승원**

**1. Sequence 1101 detector moore machine 구현(source code, simulation, 상태도, 상태표 작성)**

****

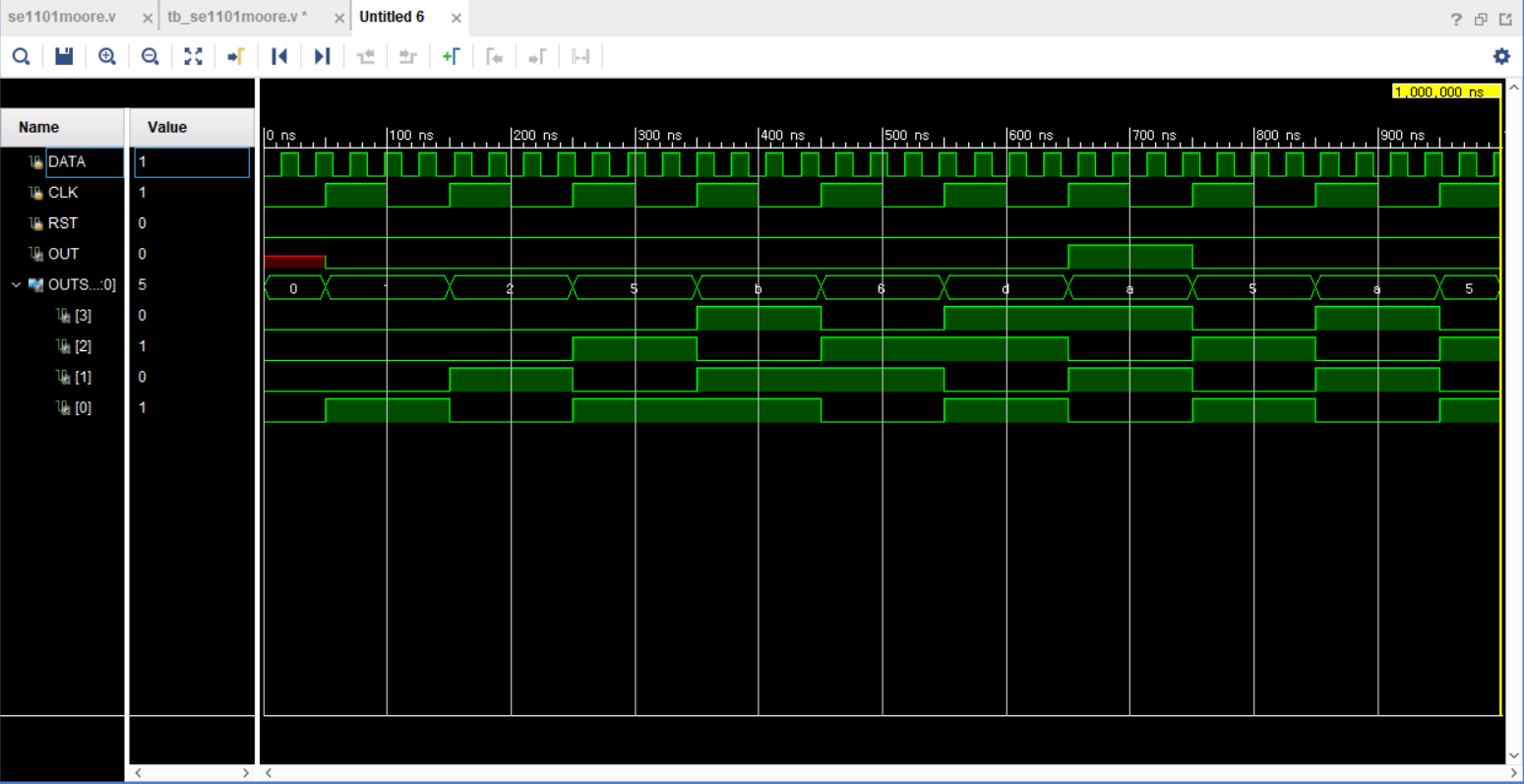
**<Figure 1. Sequence 1101 detector (moore machine) source code>**

Sequnce 1101 detector moore machine source code이다. Input으로 data, clk(clock), rst(reset)을 설정했고, output으로 배열 outseq와 sequence를 detect했는지 판별하는 out을 먼저 선언해주었다. 초기값은 0000으로 설정했고, rising edge마다 state transition이 일어나도록 했다. Rst가 1이면, outesq값을 0000이 되도록 값을 고정했고, 아닌 경우 else문을 실행하도록 했다. outseq <= outseq << 1를 통해 bit를 shift해주면서, data값을 계속해서 변화를 주면서 1을 outseq[0]에 줬으며, 1101이 나오면 out 출력이 1이 되도록 했다.

****

**<Figure 2. Sequence 1101 detector의 test bench>**

Sequence 1101 detector의 test bench code이다. 위를 통해 simulation을 진행했다.



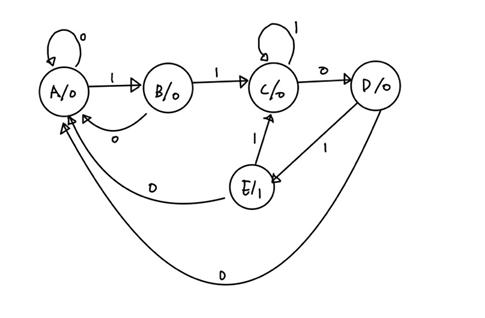
**<Figure 3. Sequence 1101 detector의 simulation 결과>**

Moore machine은 현재 상태가 1101이면, 다음 상태에서 out에서 1이 출력되어야 한다. Data input에 변화를 주면서 outseq의 LSB에 1bit를 한 번씩 입력해주면서, shift 연산자를 통해 좌측으로 bit를 이동시켰다. 0001, 0010, 0101, 1011, 0110, 1101이 나오면서, 마지막에 위의 simulation 결과를 확인해볼 때 약 600ns에서 1101이 감지됐다. 이후 그 다음 rising edge에서 out이 1로 transition된 것을 확인할 수 있다. Simulation 결과 1101 sequence를 잘 검출했음을 확인했다.

**<Table 1. Sequence 1101 detector (Moore machine) state table>**

|  |  |  |  |
| --- | --- | --- | --- |
| Present state | Next state | | Output(out) |
| input=0 | input=1 |
| A | A | B | 0 |
| B | A | C | 0 |
| C | D | C | 0 |
| D | A | E | 0 |
| E | A | C | 1 |

맨 좌측 열은 현재 상태이고, 중간은 다음 상태를 나타내는 열들이다. 1101을 검출하는 단계를 생각해보면, 현재 상태 A에서 X=1이 될 경우 B 상태로 넘어가고, X=1일 경우 C로 넘어가며, X=0인 상태가 될 경우 D, 최종적으로 X=1이면, 1101이 완성되고, E 상태가 되면서 output out으로 1을 출력한다. 1101이 아닌 각각의 다른 자리에서 다른 sequence가 검출되면 이전 상태로 돌아가는 등의 로직을 거친다.

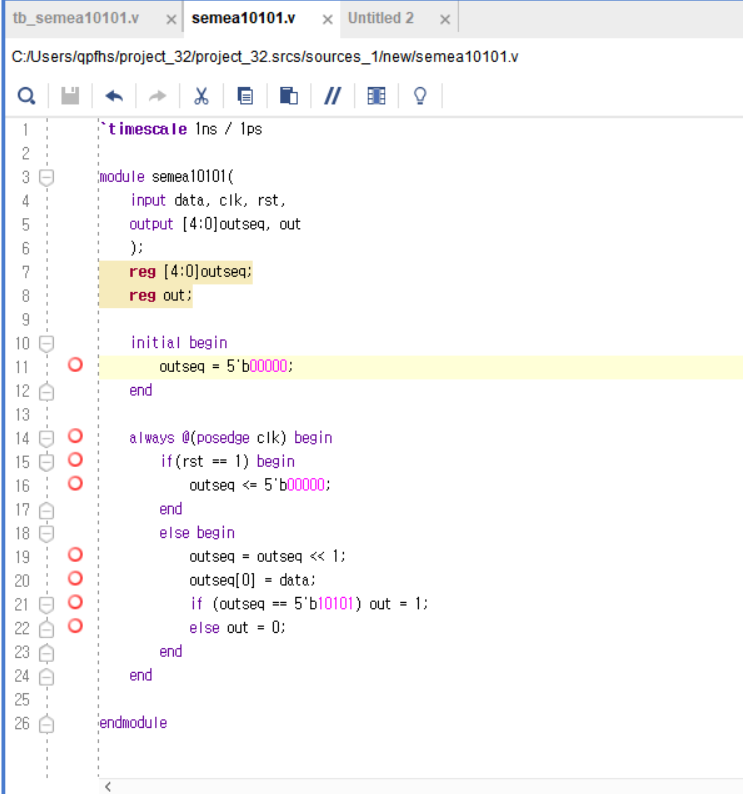


**<Figure 4. Sequence 1101 detector (Moore machine) state diagram>**

위의 state table을 바탕으로 state diagram의 모습을 나타내면 위와 같다.

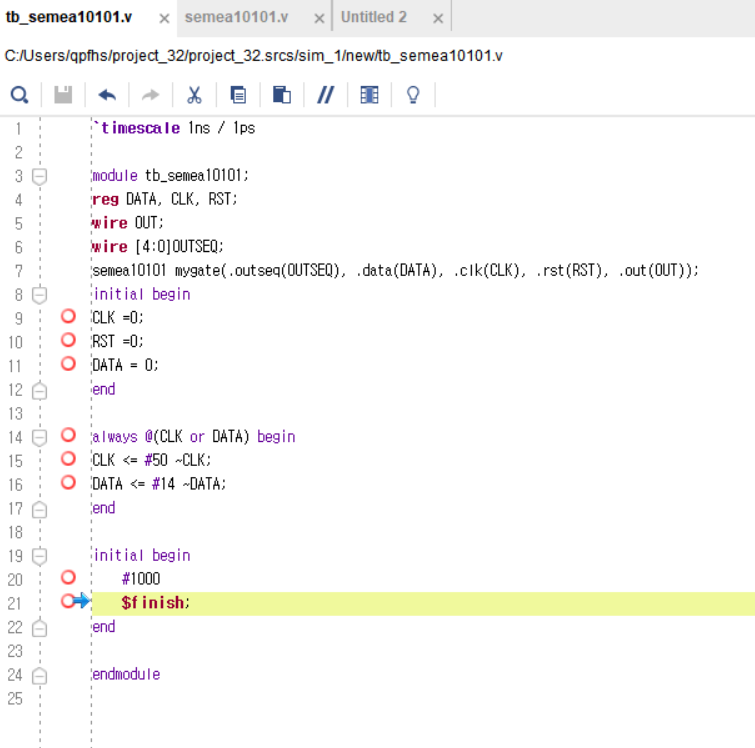
**2. Sequence 10101 detector mealy, moore machine 구현(source code, simulation, 상태도, 상태표 작성)**

**<Mealy machine>**

****

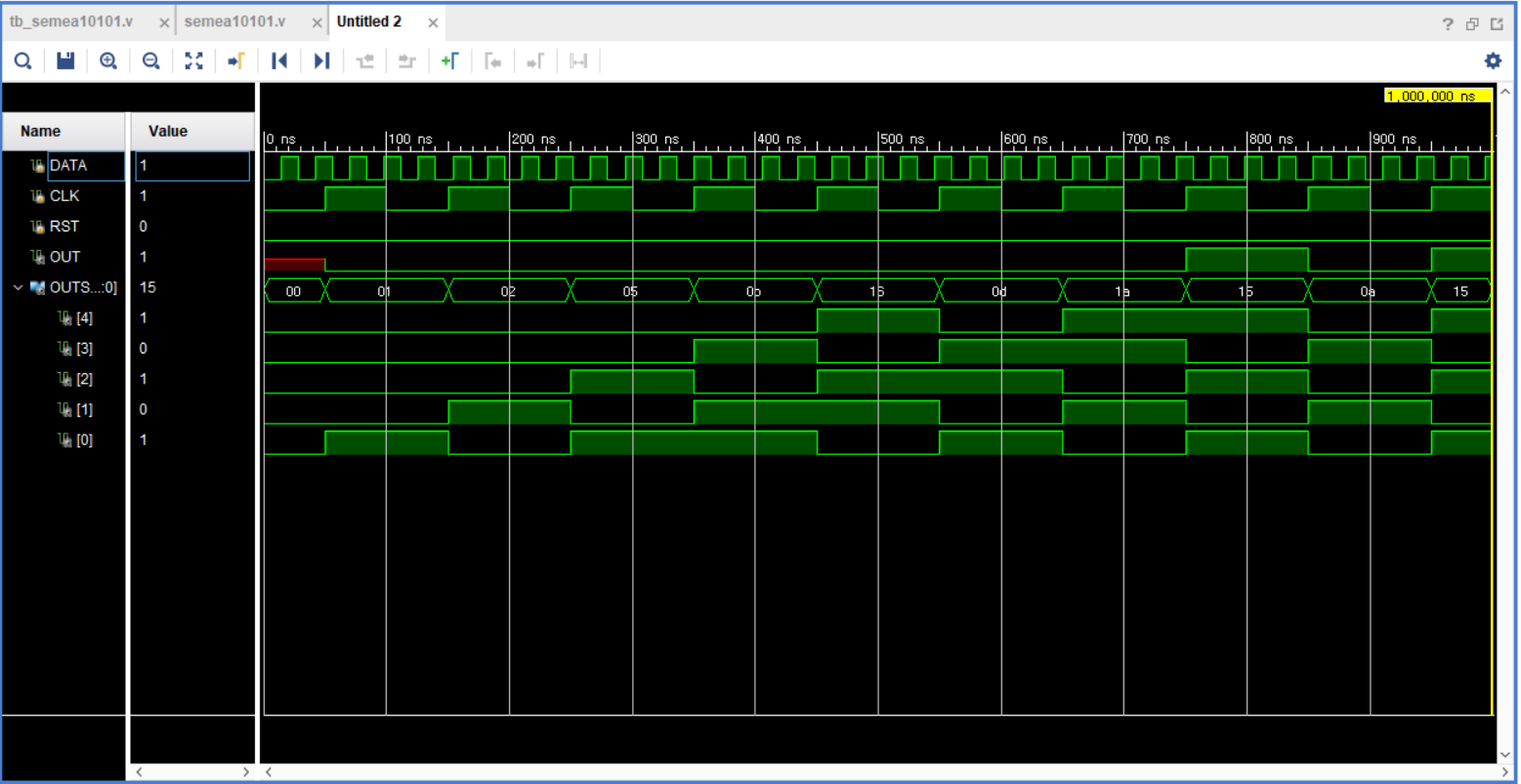
**<Figure 5. Sequence 10101 detector (Mealy machine) source code>**

Sequence 10101 detector을 Mealy machine으로 구현한 source code이다. input으로 data, clk(clock), rst(reset)을 선언했고, output으로 배열을 사용하여 outseq, 그리고 10101을 검출했을 때 1을 출력해줄 out을 선언했다. outseq의 초기값은 00000으로 설정했고, reset이 1인 경우에는 값을 0으로 고정했고, else문을 활용해서 outseq의 bit를 왼쪽으로 shift해주었다. Outseq[0]에는 data input을 통해서 값 변화를 주었고, outseq값이 10101이 됐을 때 그 시점에 1이 바로 출력되도록 Mealy machine으로 구현하였다.

****

**<Figure 6. Sequence 10101 detector (Mealy machine) testbench code>**

위의 simulation을 위한 test bench를 작성하여 simulation을 진행했다.

****

**<Figure 7. Sequence 10101 detector (Mealy machine) simulation 결과>**

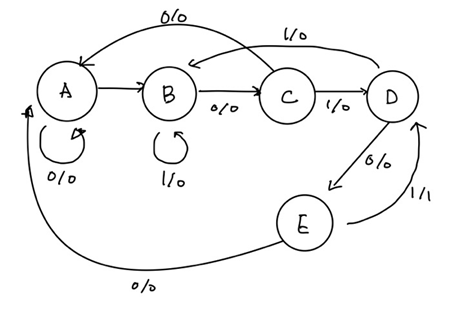
Sequence 10101 detector simulation 결과이다.

Mealy machine은 현재 상태가 10101이 되는 순간 바로 out에서 1이 출력되어야 한다. Data input에 변화를 주면서 outseq의 LSB에 1bit를 한 번씩 입력해주면서, shift 연산자를 통해 좌측으로 bit를 이동시켰다. 00001, 00010, 00101, 01011, 10110, 01101, 11010, 10101이 나오고, 약 750~850ns에서 10101이 검출됐다. 바로 10101이 검출되자마자 out값이 1로 출력된 것으로 보아 Mealy machine으로 sequence 10101 detector이 잘 구현 됐음을 확인했다.

**<Table 2. Sequence 10101 detector (Mealy machine) state table>**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Present state | Next state | | Output(out) | |
| 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 | D | A | 0 | 1 |

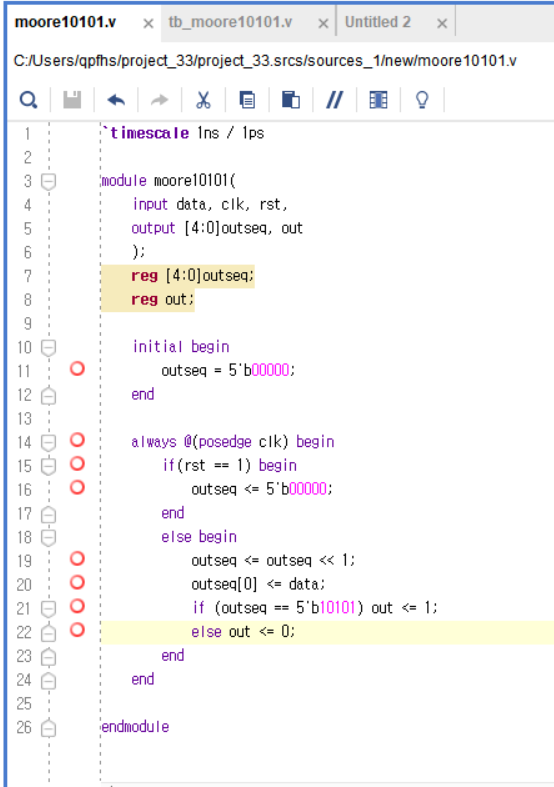
맨 좌측 열은 현재 상태이고, 중간은 다음 상태를 나타내는 열들이다. 10101을 검출하는 단계를 생각해보면, 현재 상태 A에서 input=1이 될 경우 B 상태로 넘어가고, input=0일 경우 C로 넘어가며, input=1인 상태가 될 경우 D, input=0이면, 1010까지 완성되고, 최종적으로 input이 1이 되면 E 상태가 되면서 output인 out으로 1을 출력한다. Moore machine과 다른 점은 output이 input값에 따라 곧바로 출력 값의 변화가 생기는 점인데, E상태에서 input이 1이 들어오면 곧바로 10101을 검출하여 output은 1을 출력한다. 10101이 아닌 각각의 다른 자리에서 다른 sequence가 검출되면 특정 상태로 돌아가는 등의 로직을 거친다.

****

**<Figure 8. Sequence 10101 detector (Mealy machine) state diagram>**

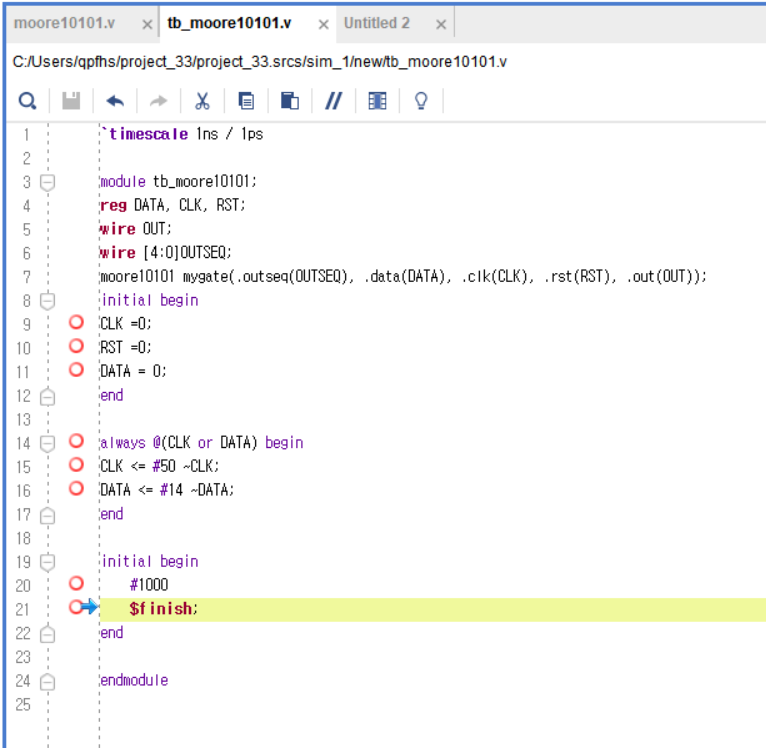
위의 state table을 바탕으로 state diagram의 모습을 나타내면 위와 같다.

**<Moore machine>**



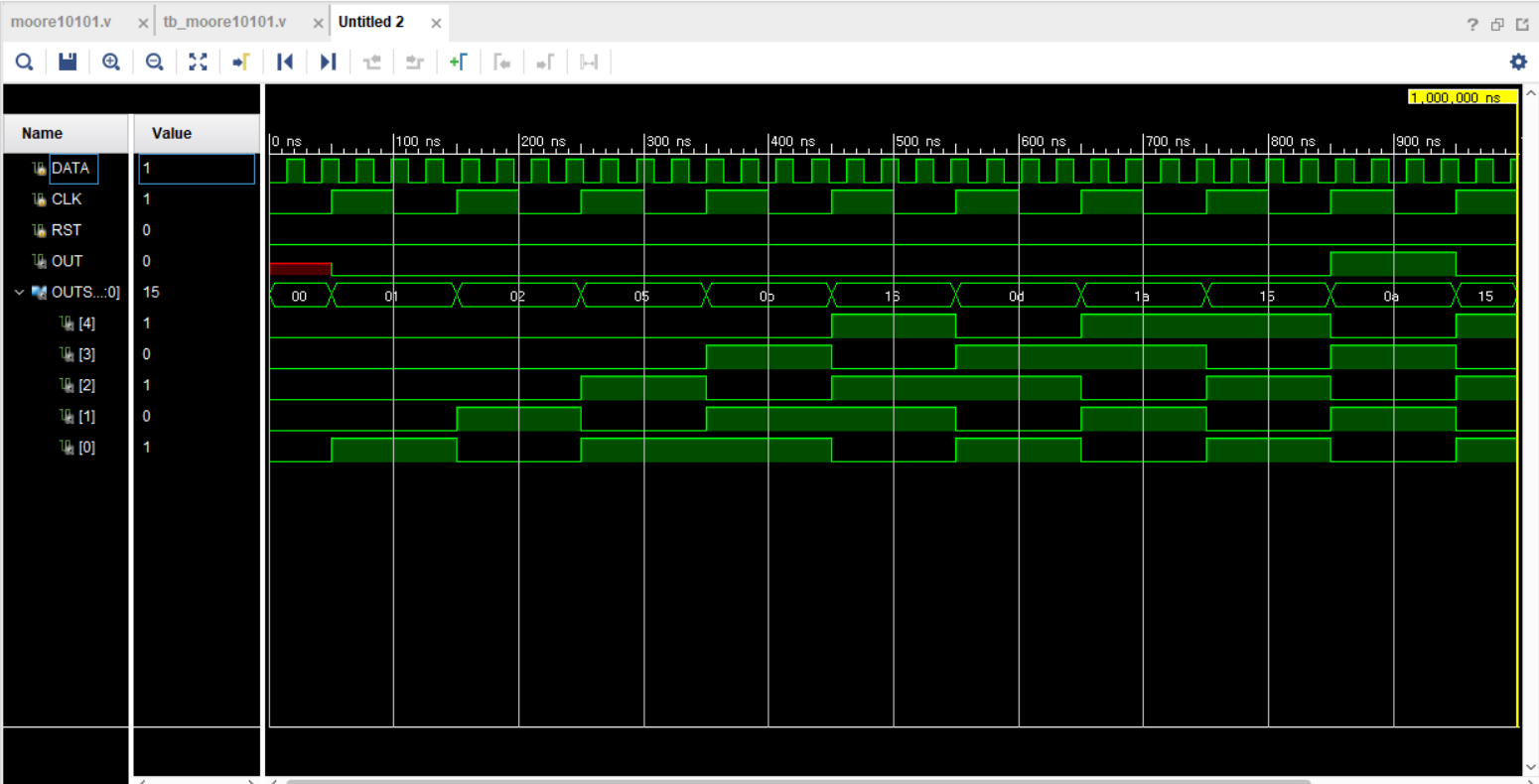
**<Figure 9. Sequence 10101 detector (Moore machine) source code>**

Sequence detector 10101을 Moore machine으로 구현한 source code이다. input으로 data, clk(clock), rst(reset)을 선언했고, output으로 배열을 사용하여 outseq, 그리고 10101을 검출했을 때 1을 출력해줄 out을 선언했다. outseq의 초기값은 00000으로 설정했고, reset이 1인 경우에는 값을 0으로 고정했고, else문을 활용해서 outseq의 bit를 왼쪽으로 shift해주었다. Outseq[0]에는 data input을 통해서 값 변화를 주었고, outseq값이 10101이 되고난 뒤에 다음 state에서 1이 출력되도록 Moore machine으로 구현하였다.



**<Figure 10. Sequence 10101 detector (Moore machine) test bench>**

위의 simulation을 위한 test bench를 작성하여 simulation을 진행했다.



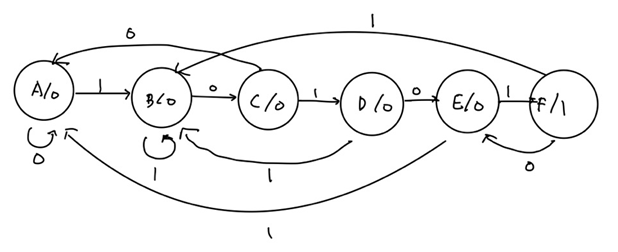
**<Figure 11. Sequence 10101 detector (Moore machine) simulation 결과>**

Sequence 10101 detector Moore machine은 현재 상태가 10101이면, 다음 상태에서 out에서 1이 출력되어야 한다. Data input에 변화를 주면서 outseq의 LSB에 1bit를 한 번씩 입력해주면서, shift 연산자를 통해 좌측으로 bit를 이동시켰다. 00001, 00010, 00101, 01011, 10110, 01101, 11010, 10101이 나오고, 약 750~850ns에서 10101이 검출되고, 현 시점이 아닌 다음 state에서 out 1을 출력했다. Simulation 결과 10101 sequence를 잘 검출했음을 확인했다.

**<Table 3. Sequence 10101 detector (Moore machine) state table>**

|  |  |  |  |
| --- | --- | --- | --- |
| Present state | Next state | | Output(out) |
| 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 |

맨 좌측 열은 현재 상태이고, 중간은 다음 상태를 나타내는 열들이다. 10101을 검출하는 단계를 생각해보면, 현재 상태 A에서 input=1이 될 경우 B 상태로 넘어가고, input=0일 경우 C로 넘어가며, input=1인 상태가 될 경우 D, 그 다음 input이 0이면 E, 최종적으로 input=1이면, 10101이 완성되고, F 상태가 되면서 output out으로 1을 출력한다. 10101이 아닌 각각의 다른 자리에서 다른 sequence가 검출되면 이전 상태로 돌아가는 등의 로직을 거치게 된다.



**<Figure 12. Sequence 10101 detector (Moore machine) state diagram>**

위의 state table을 바탕으로 state diagram의 모습을 나타내면 위와 같다.