1. Synchronous sequential circuits

   Combinational logic 뒤에 flip-flops이 존재하는 회로를 말한다. 회로의 currentstate라는 output은 다시 nextstate를 정하는데 feedback효과를 한다. Flip-flop은 register역할로써 clk의 positive edge 순간에 동기화하여 nextstate를 currentstate로 만든다.

2. Finite State Machine(FSM)

   Synchronous sequential circuits를 이용하여 구현한다. State register와 Combinational logic으로 구성된다.

   - Moore FSM: output은 오직 currentstate에만 의존하며 nextstate의 경우 currentstate와 input에 의존한다.
   - Mealy FSM: output은 currentstate뿐만 아니라 input에도 의존한다.

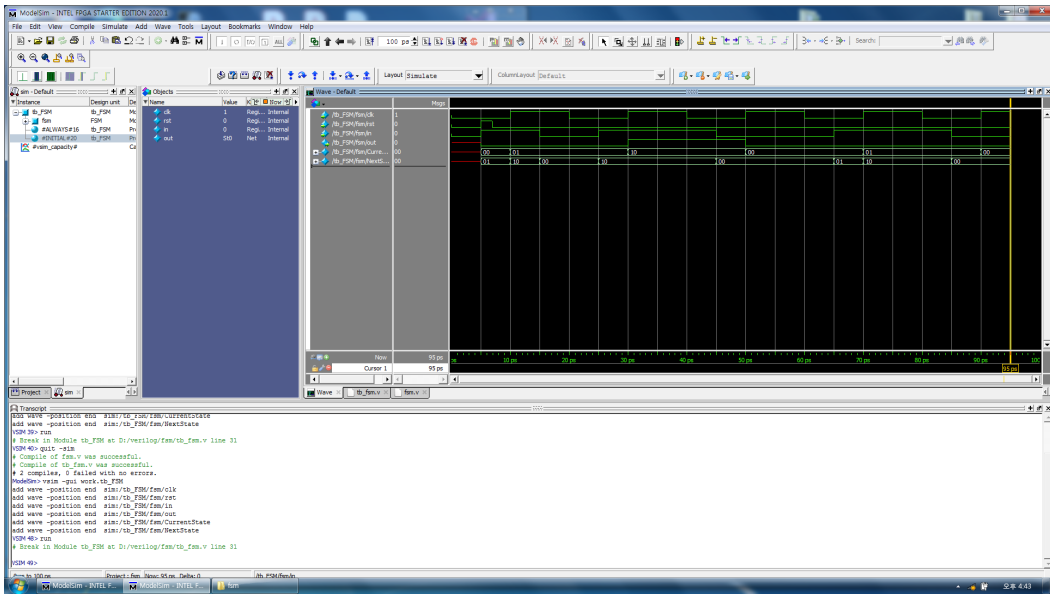
3. 과제에서 주어진 FSM

   3개의 state로 구성되며(IDLE, S0, S1) 각각 state code는 00, 01, 01로 mapping한다. clk의 positive edge순간에 currentstate가 변하고 currentstate에 따라 output값이 변한다. output은 0(IDLE, S0) 또는 1(S1)이다. clk의 positive edge전에 nextstate는 currentstate와 input으로 결정된다. S0에서 input이 0과 1일 때 각각 IDLE, S1으로 변하고 IDLE에서 input이 0과 1일 때 각각 IDLE, S0으로 변하고 S1에서 input이 0과 1일 때 각각 IDLE, S1으로 변한다.

4. 로직 구성

   본격적으로 clk를 변동하기 전에 rst를 통해 처음 state를 IDLE로 초기화한다. 이후 rst의 효과를 없애기 위해 0으로 다시 설정해 준다. clk는 10을 주기로 값을 변동하고 input의 경우 clk와 엇박자가 되도록 5, 15, 15… 주기로 값을 변화시킨다. clk의 positive edge순간인 10, 30, 50… 시간에 state transition을 관찰할 수 있도록 input값을 적절히 조정한다.

5. Waveform



state transition의 순서가 IDLE -> S0 -> S1 -> IDLE -> S0 -> IDLE 되도록 clk와 input를 설정했다. 예측한 대로 clk의 positive edge순간에 currentstate변화를 관찰할 수 있었다. currentstate에 따라 output값이 정해지는 것도 관찰할 수 있었다.

6. 결과 및 고찰

FSM를 베릴로그 문법을 통해 직접 구현해 보았다. currentstate, nextstate, input 그리고 output 사이의 의존관계를 정의하고 회로를 구성한 후 로직을 구상했다. always, initial, # 구문을 통해 flip-flop(register)를 직접 구현하기 위해서 flip-flop의 개념과 정의에 대해 복습이 필요했다. flip-flop은 clk의 주기가 도달하기 전에 nextstate를 저장해 두고 positive edge순간(동기화)에 nextstate를 currentstate로 할당한다. 이러한 개념을 이용하여 코딩을 하였고 생각보다 쉽게 코드를 구현할 수 있었다. 이번 과제를 통해 FSM과 Synchronous sequential circuits의 깊은 이해를 할 수 있었다.