논리회로 설계 및 실험

7주차

목표

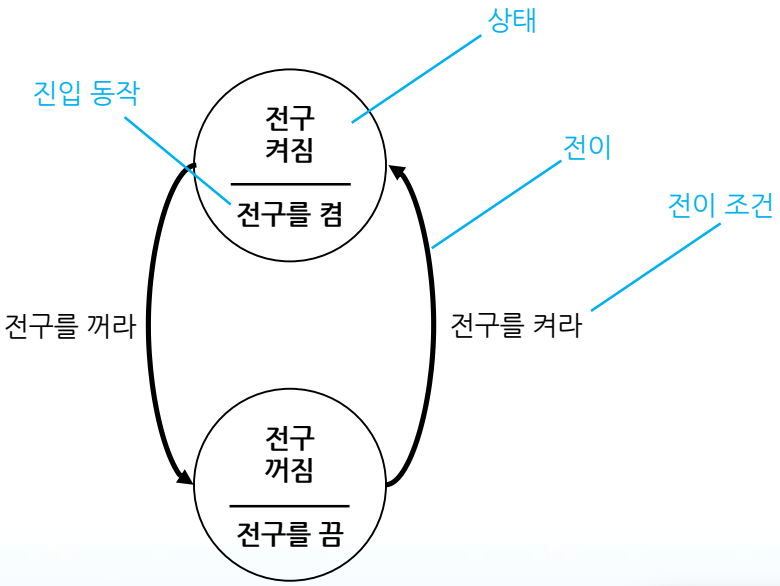
1. 유한상태기계(Finite State Machine, FSM)에 대한 이해
2. FSM을 이용한 CU 설계
3. Stadian을 이용한 FSM 설계 학습

유한상태기계

- 프로그램, 논리회로, 정규 표현식 등을 표현하고 설계할 수 있는 수학적 모델
- 유한상태기계의 특성
  - 한번에 하나의 상태만 가짐
  - 어떠한 사건에 의해 현재 상태에서 다른 상태로 변화함
  - 유한상태 변환기 모델로는 무어(Moore) 모델과 밀리(Mealy) 모델이 있음

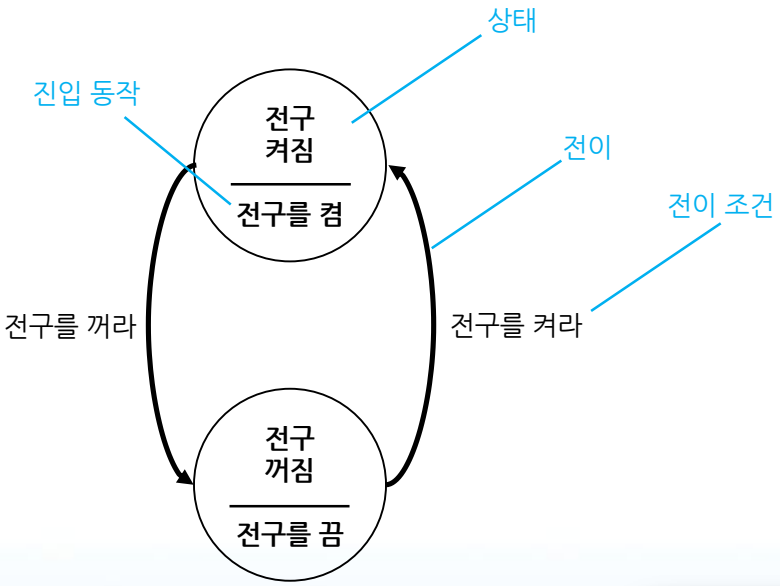
유한상태기계의 예시

- 전구를 켜고 끄는 FSM



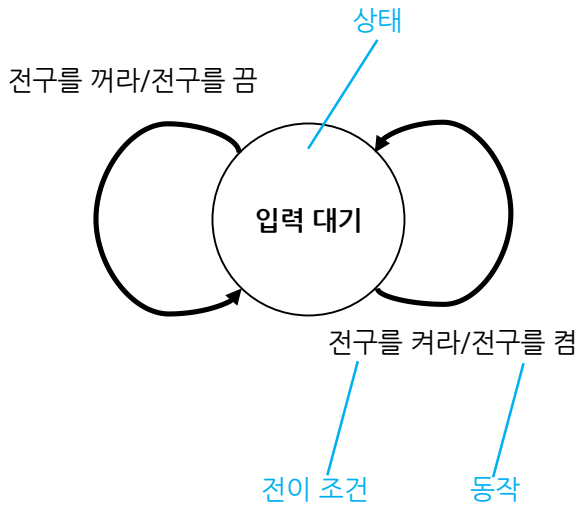
Moore Machine

- 출력이 현재 상태에 따라서 결정됨
- 상태에 진입할 때, 진입 동작을 수행함
- 단순하고 직관적이지만 상태의 수가 많음



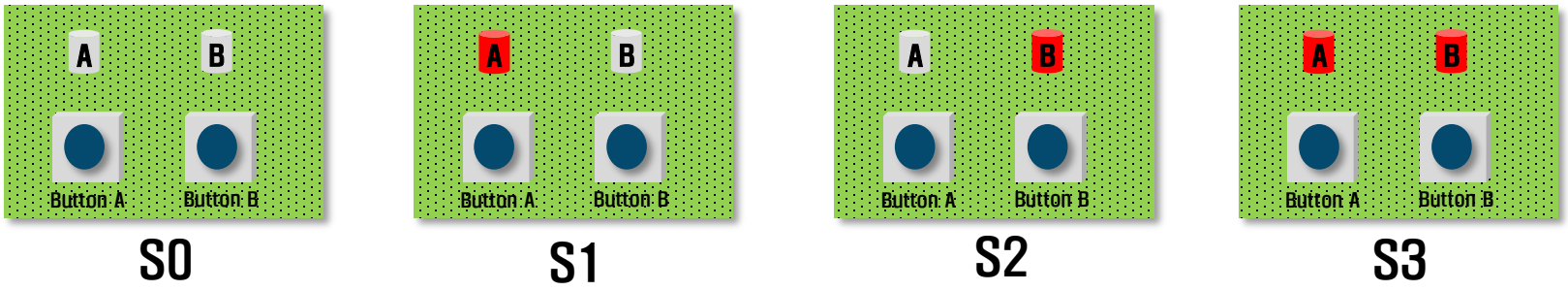
Mealy Machine

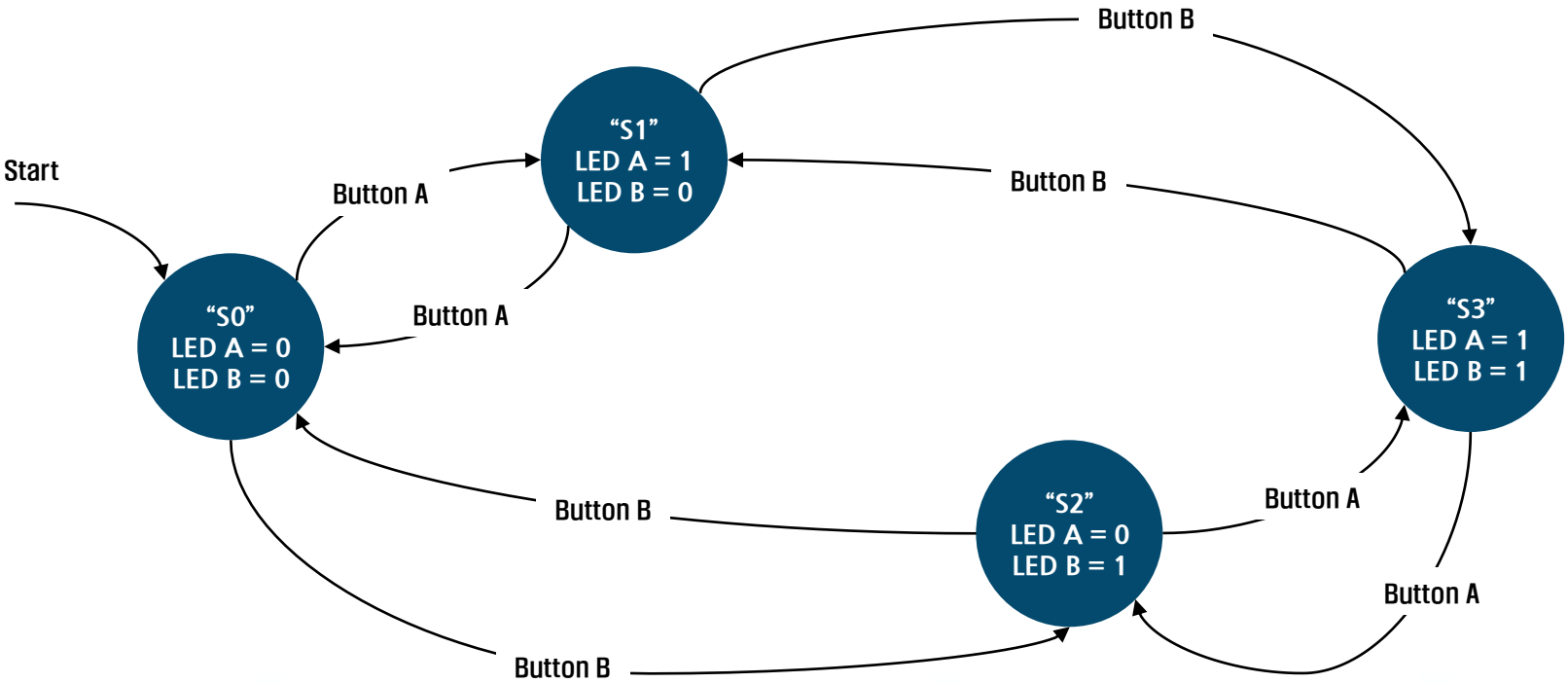
- 출력이 현재 상태와 입력에 따라서 결정됨
- 즉, 어떤 입력이 들어올 때 함께 지정된 동작이 동시에 발생함
- 진입 동작은 없음
- 상태의 수를 줄일 수 있으나 전이 조건 등이 복잡함



유한상태기계(Finite State Machine, FSM)

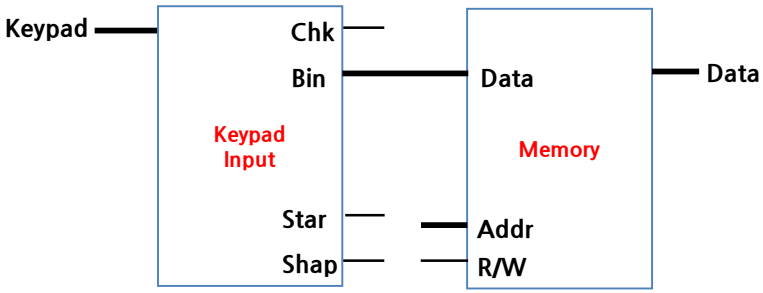
Ex. Button A를 누르면 LED A, Button B를 누르면 LED B가 토글되는 기계



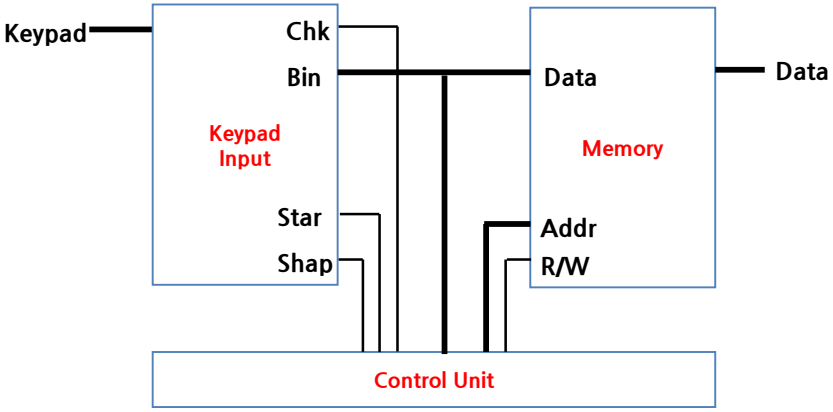


Control Unit

- 데이터 처리(연산, 저장, 쉬프트 등)를 할 수 있도록 제어 신호를 공급함
  - 지난 실습에서 메모리의 Addr, R/W나 레지스터의 Ce 등이 제어 신호
- 데이터의 흐름을 조절하고 시스템의 정해진 작업을 수행



(1) 입력장치와 저장장치

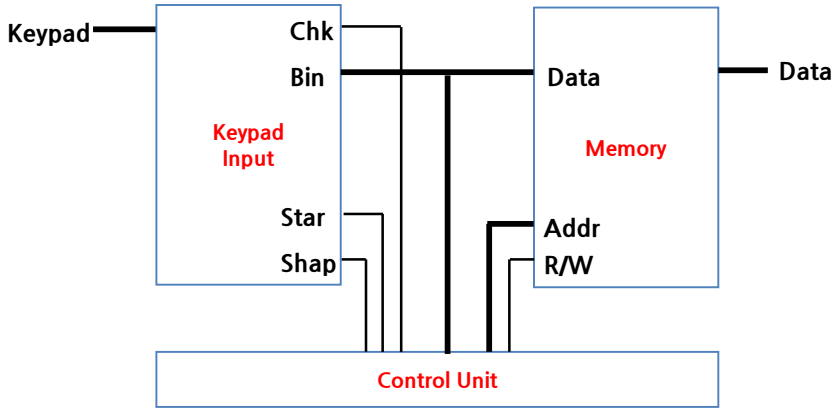


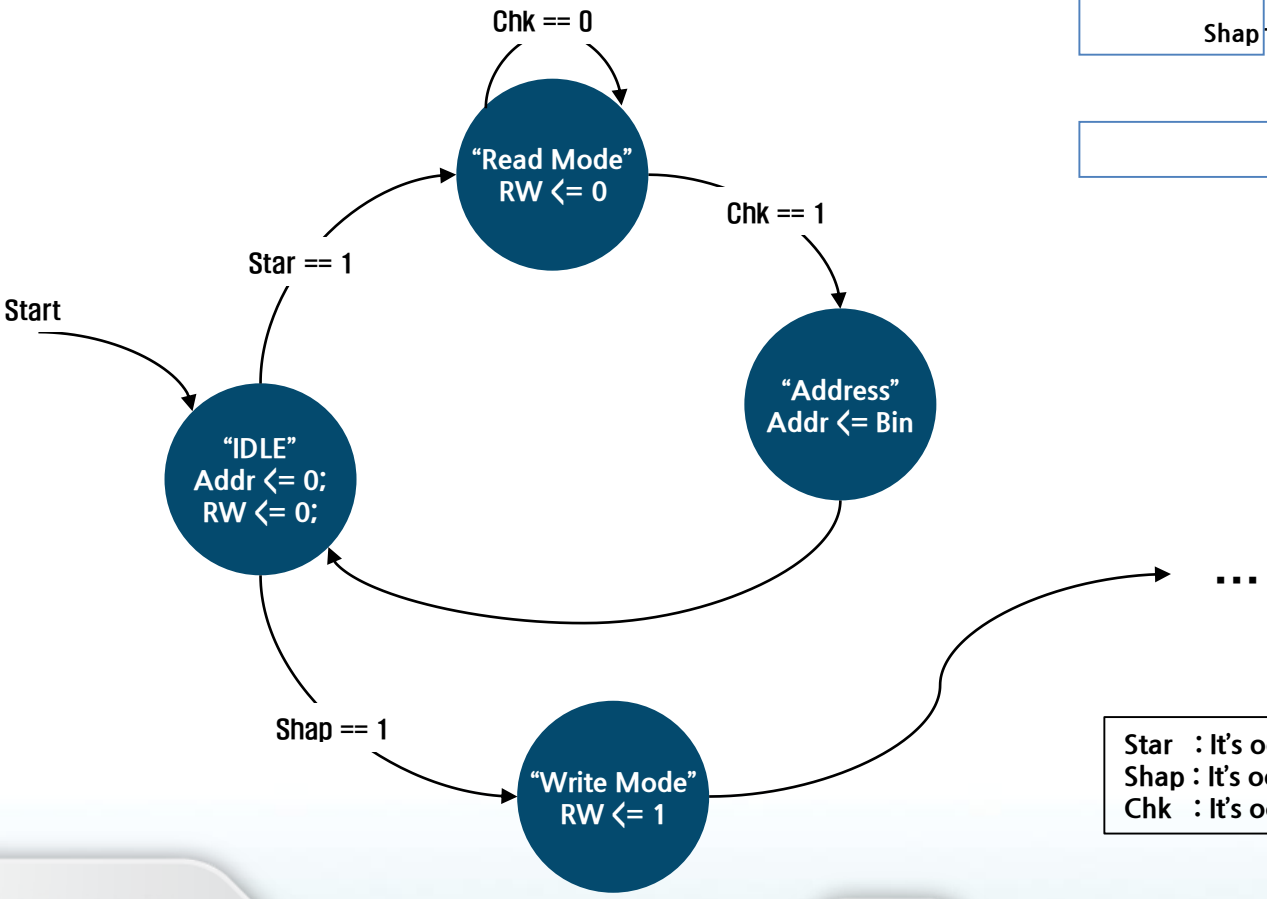
(2) 컨트롤 유닛의 제어를 받는 입력장치와 저장장치

(1) 의 경우 메모리에 원하는 값을 저장하기 위해서는 Keypad 입력 외에 Addr 값과 R/W 값을 직접 지정해줘야 함
(2) 의 경우 Keypad 입력만으로 메모리에 읽기, 쓰기, 주소 지정 등이 가능함

Ex. Keypad의 입력 조합으로 메모리를 제어하는 Control Unit의 FSM

(1) *을 누르면 읽기 모드
(2) #을 누르면 쓰기 모드
(3) 읽기 모드에서 숫자를 누르면 값을 읽을 주소 지정
(4) …



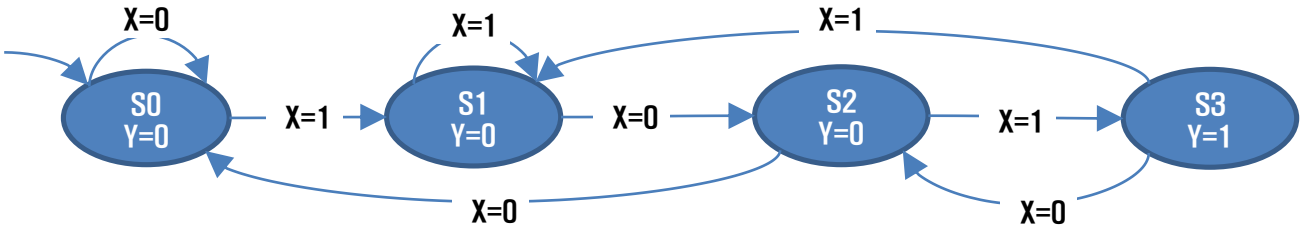


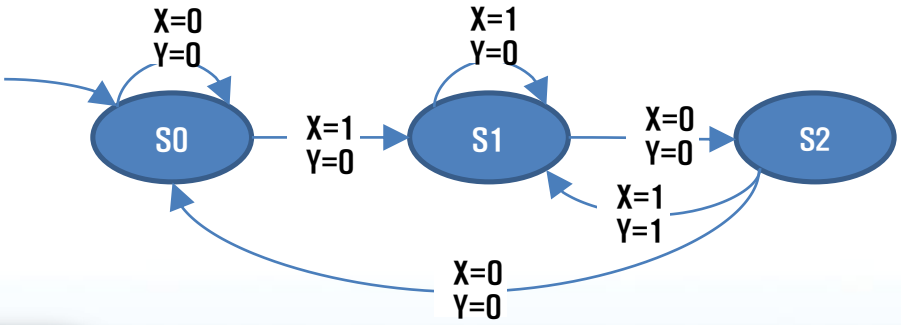
Star : It's occur when pressed keypad's * button
Shap : It's occur when pressed keypad's # button
Chk : It's occur when pressed any number on keypad

FSM을 이용한 Sequential Filter

- 어떤 문자열에서 특정한 패턴이 발견될 때 지정한 동작을 수행하는 필터

Ex. 문자열에서 101 패턴을 찾아내는 Sequential Filter



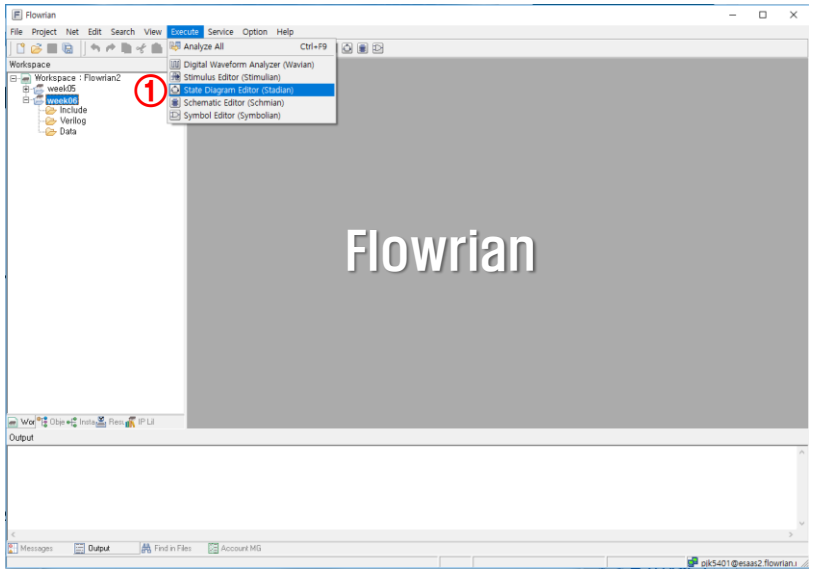


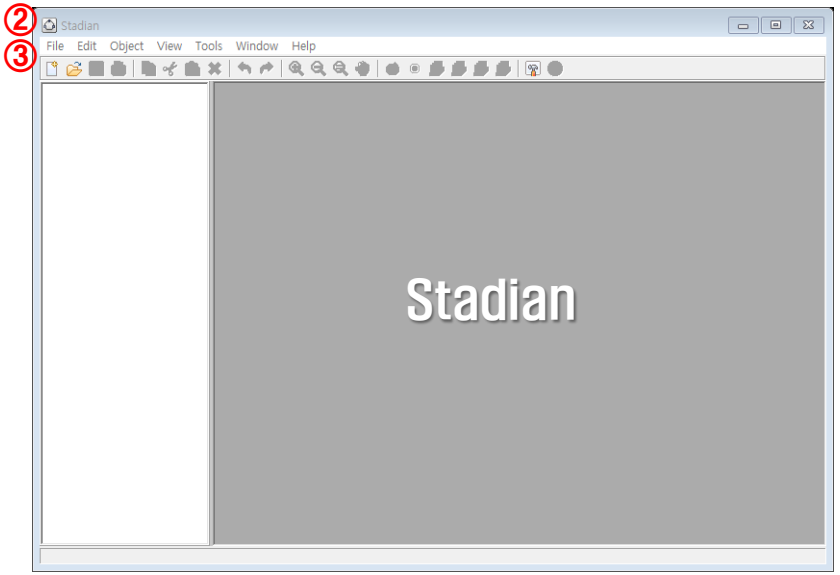
실습

Stadian을 이용한 FSM 설계

상태도 생성 (1/2)

① Flowrian에서 [Excute] -> [State Diagram Editor] 실행
② Stadian 툴이 실행 됨
③ Stadian에서 [File] -> [New State Map] 실행



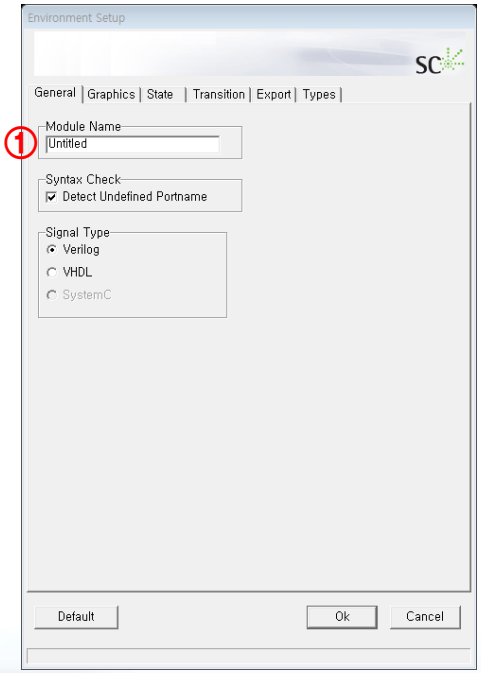


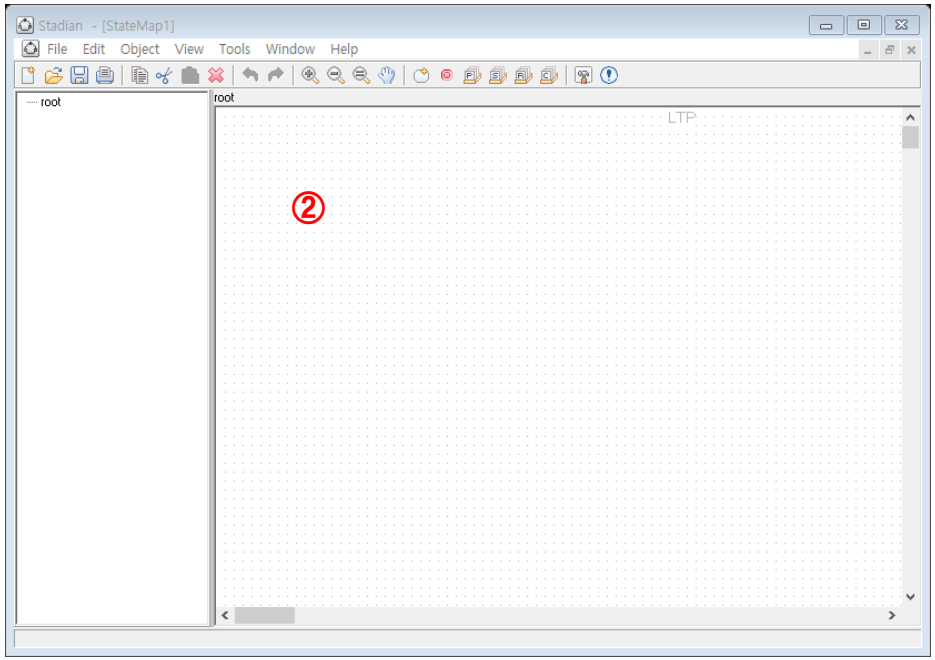
상태도 생성 (2/2)

① Module Name 지정
- 모듈명은 영어로 시작해야 하며 숫자, 또는 특수기호로 시작할 수 없음
- 올바른 모듈명이 아니면 정상 동작을 하지 않고 Compile error가 발생

② Signal Type은 Verilog 선택

③ 신규 회로도 생성

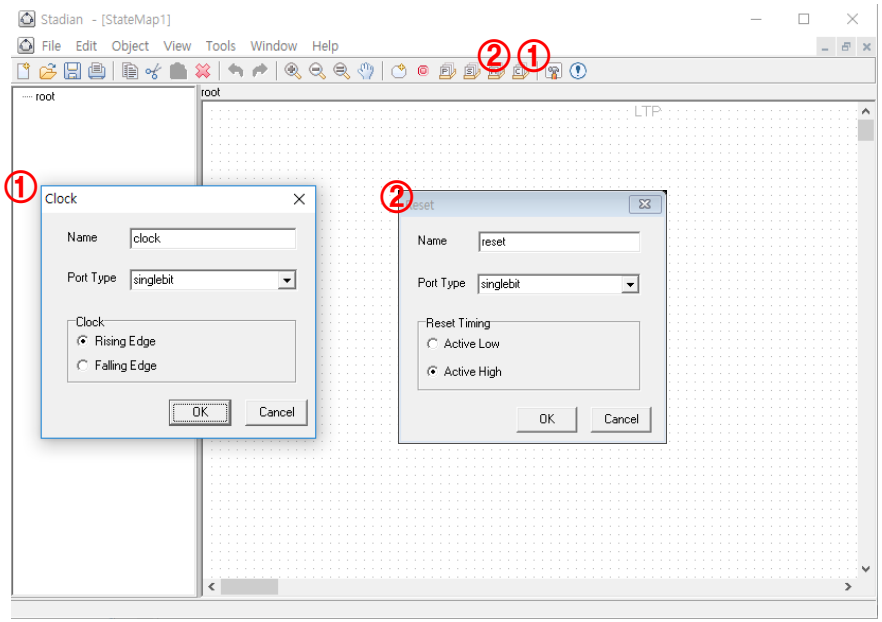




상태도 작성 (1/4)

① Clock 포트 설정
- Clock 포트명, 타입, Edge 방식을 선택

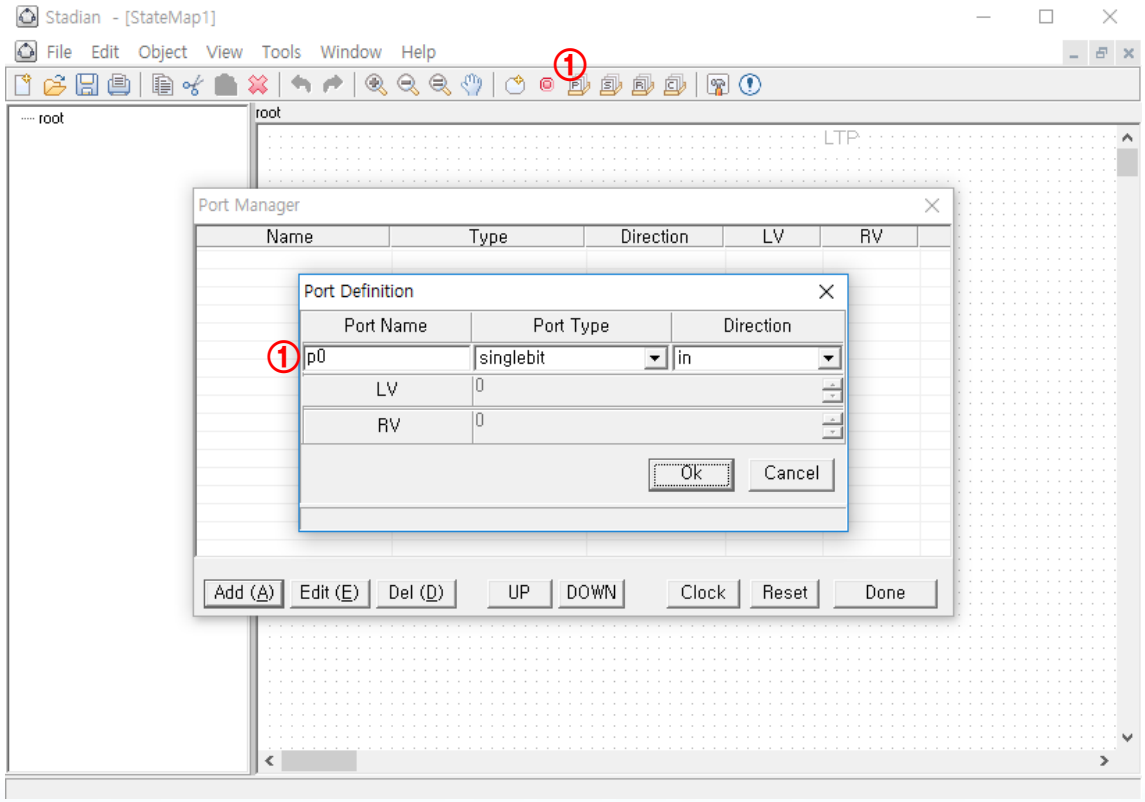
② Reset 포트 설정
- Reset 포트명, 타입, Reset 타이밍을 선택, 타이밍은 Active High로 할 것



상태도 작성 (2/4)

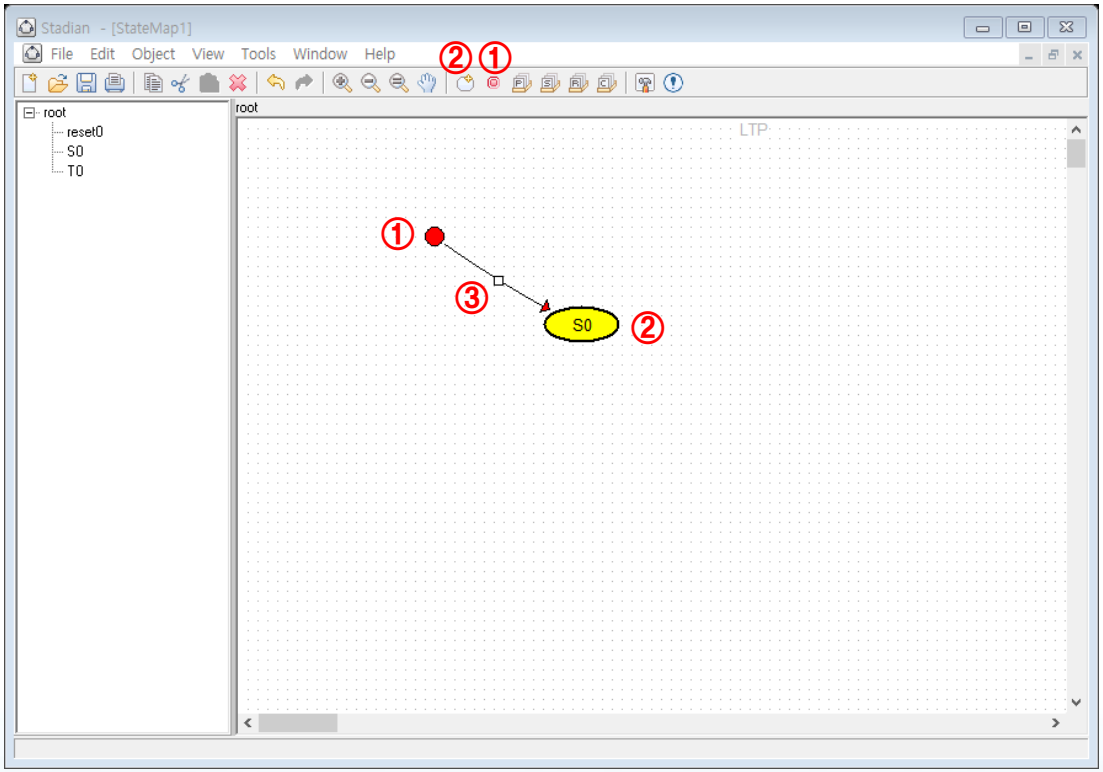
① 포트 설정

- 포트명, 타입, 입출력을 선택



상태도 작성 (3/4)

① 리셋 추가
② 상태 추가
③ 오브젝트들을 우클릭하여 전이 추가



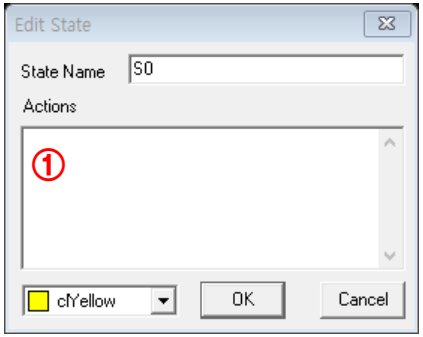
상태도 작성 (4/4)

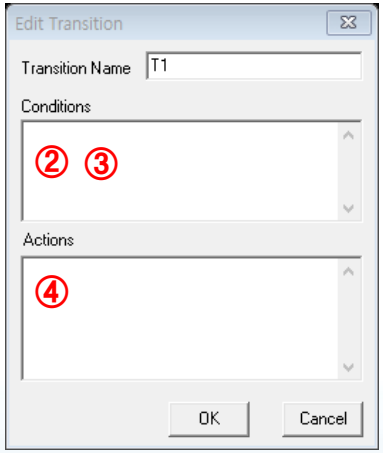
① 상태를 더블클릭하면 상태 진입 동작을 기술할 수 있음
- Moore machine의 경우 진입 동작을 기술

② 전이를 더블클릭하면 전이조건과 동작을 기술할 수 있음
- Moore machine은 전이 조건만, Mealy machine은 전이 조건과 동작 둘 다 기술

③ 전이 조건 기술방법
- C 언어 if 문의 괄호( ) 안에 들어가는 내용처럼 조건을 기술
- ex : Ce == 1

④ 동작 기술 방법
- C 언어의 문장처럼 동작을 기술하고 세미콜론(;)으로 문장을 매듭
- ex : Addr <= 3;

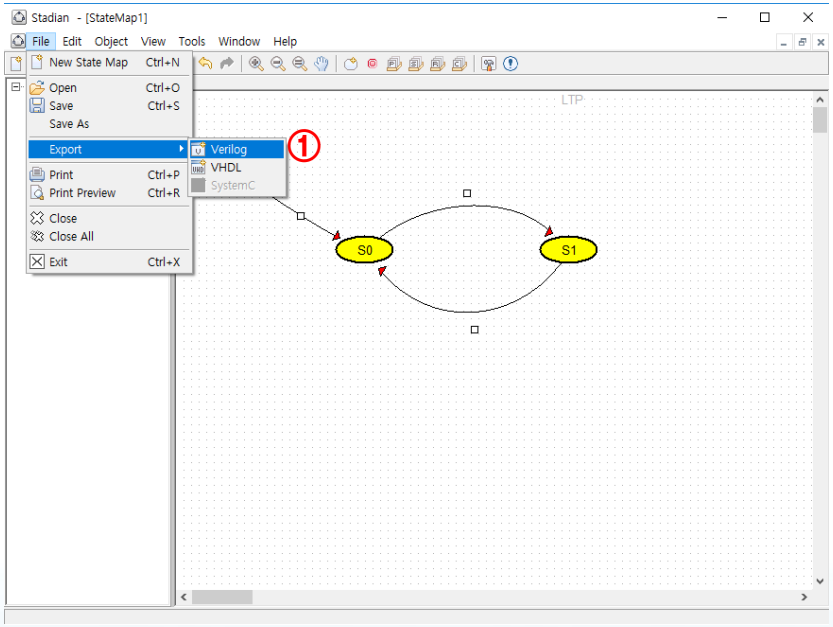




상태도 저장 및 변환

① [Export] -> [Verilog] 실행
- 파일명 지정 후 저장하면 .v 파일 생성
- 이 때, 파일명은 반드시 상태도의 모듈명과 일치시킬 것

※ Save와 Export의 차이
- [Save] 는 상태도(.std)를 저장
- [Export] 는 상태도를 HDL 파일(.v)로 변환 후 저장



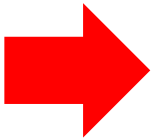
주의사항

① Export한 v파일 실행
- [Verilog] -> [Add Existing File] 클릭해서 v파일 추가시킨 다음 더블클릭

② negedge -> posedge

③ rst == 1'b0 -> rst == 1'b1

```
always @(posedge clk or negedge rst)
begin: SYNCH
  if (rst == 1'b0)
     current_state <= S0;
  else
     current_state <= next_state;
end
```



```
always @(posedge clk or posedge rst)
begin: SYNCH
  if (rst == 1'b1)
     current_state <= S0;
  else
     current_state <= next_state;
end
```