**Final Project : Ping Pong Game**

20190766 윤병준

20190943 박성현

1. **Abstract**

Final Project으로 제작할 FSM은 고전 게임인 Pong Game이다. Player는 버튼을 통해 패들의 위치를 조절하여 공을 받을 수 있고, VGA를 통해 모니터에 출력할 계획이다. 1인용 게임이며 점수는 시간이고, 공이 패들에 부딪힐 때마다 속도가 점점 빨라지도록 설정할 것이다.

1. **Background**
   1. FSM (Finite State Machine)

FSM, Finite State Machine은 전자 논리 회로를 설계하는 데 쓰이는 수학적 모델이다. 유한한 개수의 상태를 가지는 추상 기계로 한 번에 오로지 하나의 상태만을 가지고, 이때 어떤 사건에 의해 다른 상태로 변화할 수 있으며 이를 transition, 전이라 한다.

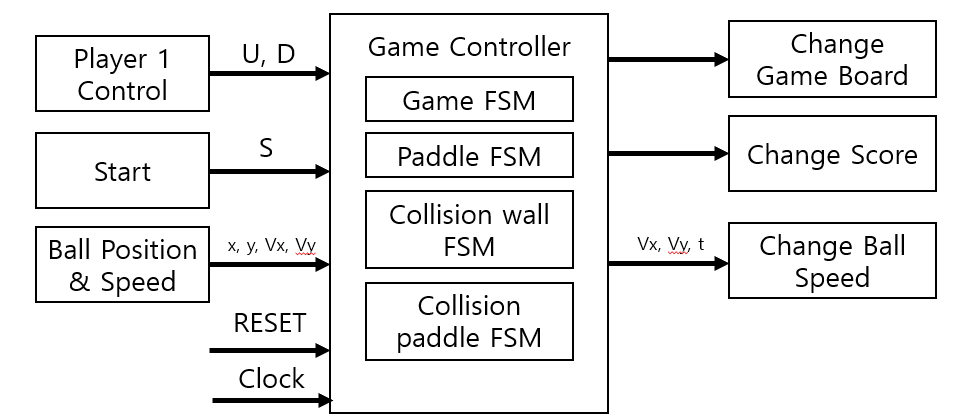
* 1. Pong Game

Ping Pong Game은 미국의 게임회사 Atari에서 1972년 개발한 게임으로, 2차원 탁구 게임이다. 플레이어는 게임 패들을 상하로 조작해 공을 주고받고, 받는 데 실패한 경우 상대가 점수를 획득하는 방식이다. 이번 Final Project에서는 패들을 1개만 사용하여 1인용 Pong Game을 만들 것이다.

1. Overall Circuit Structure
   1. Paddle - Move Up, Down
   2. Collision - Detect Collision between ball vs board & paddles
   3. Game Controller - Reset, Start, Score
   4. Map game data to panel
2. Input and Output
   1. Input: Player Control Up/Down, Reset Button, Start Button
   2. Output: VGA-to-Monitor, Score (LED)
3. FSM

이번 프로젝트에서 사용하는 FSM은 총 4개로, Game, Paddle, Wall Collision, Paddle Collision이다.

* 1. Block Diagram 설계

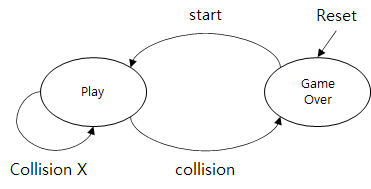


위 표는 Pong 게임의 동작을 이해하기 위해 Block Diagram으로 나타낸 것이다. 아래는 그 과정과 Input/Output이다.

게임은 Start을 통해 시작되고 Player의 입력이 입력될 때마다 Paddle FSM을 통해 공간이 충분할 경우 Paddle 위치를 움직인다. 또 현재 Ball의 위치, 속도의 값을 받아와 Game, Collision FSM에서 이를 이용해 Game Over의 여부와 공의 부딪힘 여부를 확인하고 속도를 수정한다.

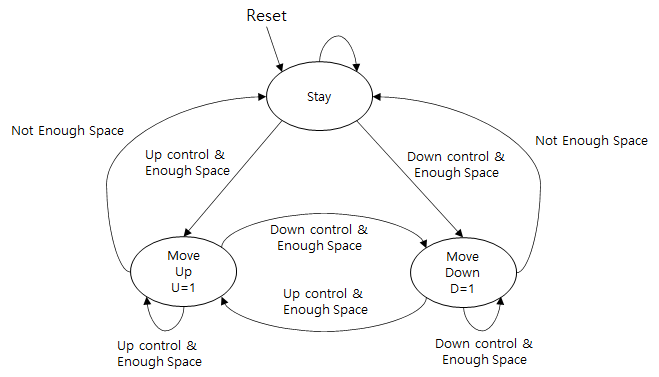
* 1. State Diagram of FSM

i) Game State Diagram



위 그림은 게임 진행의 동작을 state diagram으로 나타낸 것이다. 초기 상태는 Game Over이고, Reset Input이 들어오면 이로 초기화된다. 버튼을 누르면 Play 상태로 transition이 일어난다. Play 상태에서는 왼쪽 벽과의 collision이 없으면 상태가 유지되고, collision이 일어나면 다시 Game Over로 transition이 일어난다. 이 FSM의 input은 S, C이며 점수와 화면 출력을 담당한다.

ii) Paddle State Diagram

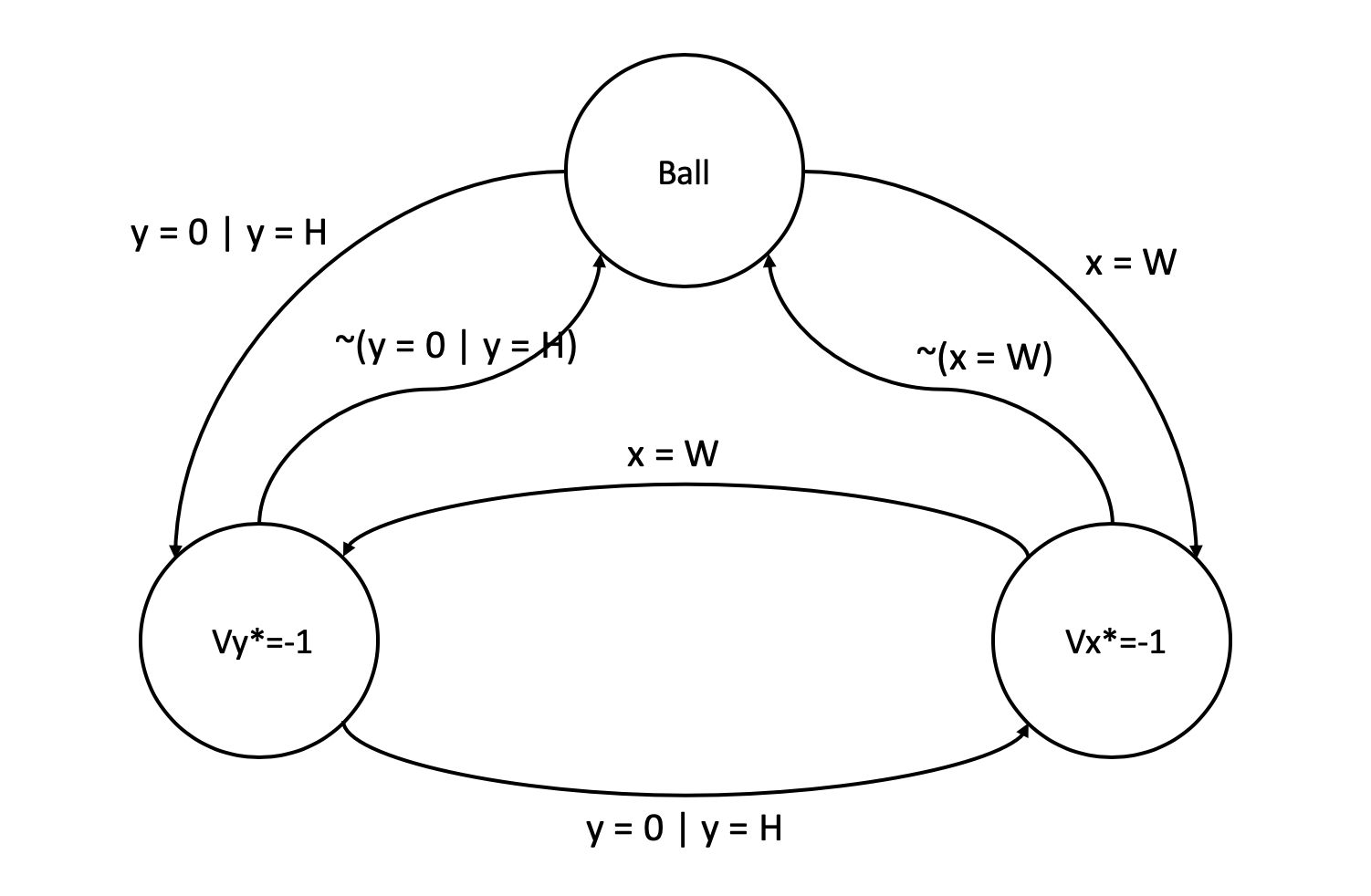


위 그림은 게임 내 플레이어가 조종하는 Paddle의 동작을 State Diagram으로 나타낸 것이다. 초기 상태는 Stay이고, Reset Input이 들어오면 이로 초기화된다. 공간이 충분하고(Enough Space) 이때 위 또는 아래 버튼 input이 들어오면 Move Up / Move Down으로 transition이 일어난다. Move Up, Move Down에서도 이는 마찬가지이며, 움직일 공간이 없을 때에는 Stay로 Transition이 일어난다. Move Up / Move Down으로 transition이 일어나면 Output U / D가 생겨 패들을 움직이게 된다. 이 FSM의 input은 U\_C(Up Control), D\_C(Down Control), E\_U(Enough Up space), E\_D(Enough Down space), 출력신호는 U, D 2가지다.

iii) Wall Collision State Diagram

W: Width of window

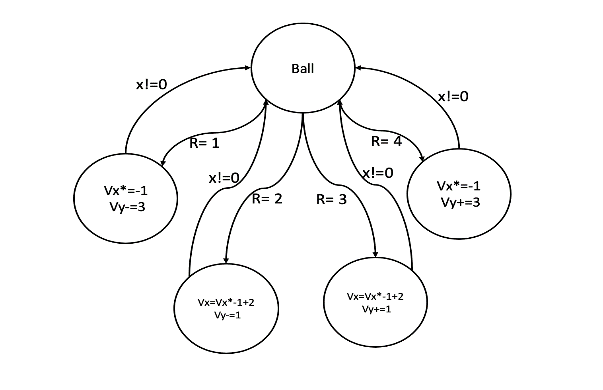
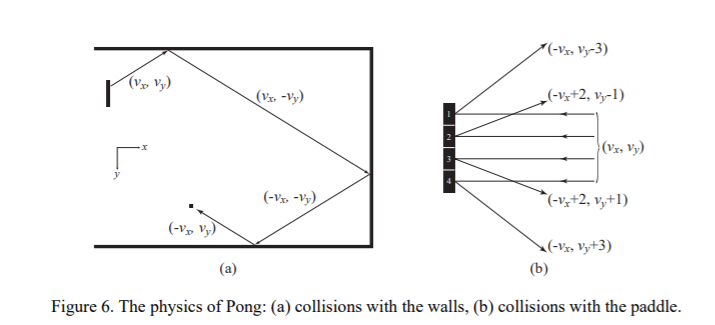
H: Height of window



Paddle이 왼쪽 테두리에서 움직이고 나머지 3면에 부딪칠때 속도 변화를 나타낸다. x,y는 좌표를 나타내고, Vx, Vy는 해당 방향으로의 속도 변화를 나타낸다

iv) Paddle Collision State Diagram

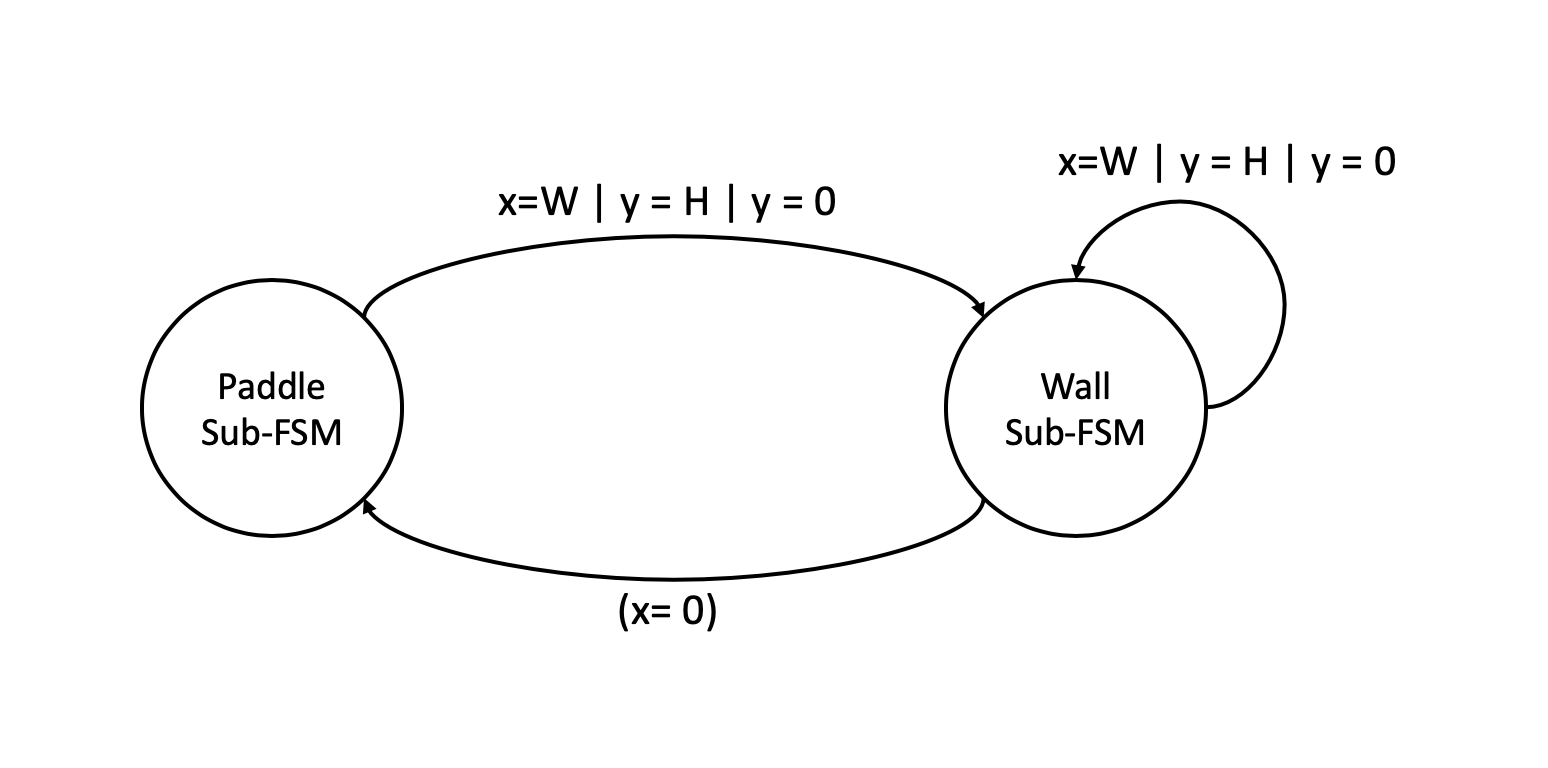
R: Collision region of paddle based on the following diagram



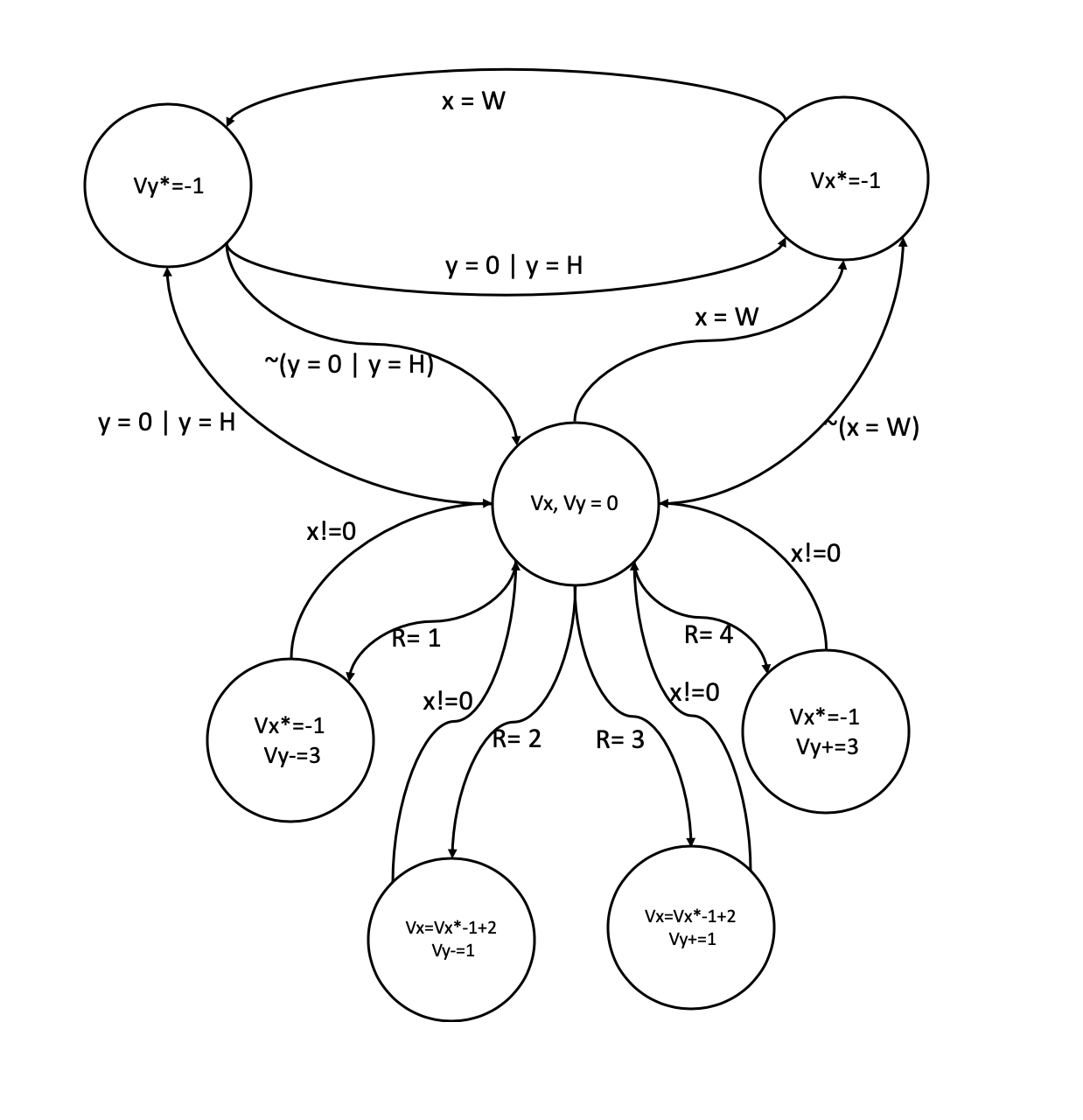
Paddle의 충돌하는 위치에 따라서 각 축으로의 속력 변화량이 다르다. 왼쪽 그림은 Pong game의 충돌의 따른 속도 변화량이고 오른쪽은 이에 해당하는 FSM이다.

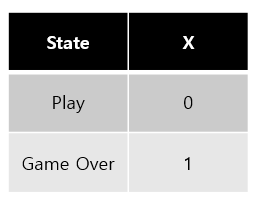
* 1. State Minimization: Collision State Diagram

Ball State는 현재 속도를 유지하며 이동하고 있는 공의 상태이다. 원래는 위 두 충돌 FSM 은 하나로 합쳐져야하는데 너무 복잡해져 두개로 분리하고 각 Ball (Wall FSM)과 Ball (Paddle FSM)을 이동하는 FSM을 따로 만들었다.



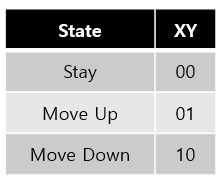
Ball state의 전체 FSM은 다음과 같다



* 1. State Assignment

i) Game State

표현할 state의 상태가 2개이므로 flip-flop은 1개만 사용하면 된다. 이를 X라고 하면 표1과 같이 2개의 state들을 나타낼 수 있다.

ii) Paddle State

표현할 state의 상태가 3개이므로 flip-flop은 최소한 2개를 필요하다. 이를 X, Y라고 하면 표2과 같이 3개의 state들을 나타낼 수 있다.

iii) Wall Collision State

iv) Paddle Collision State

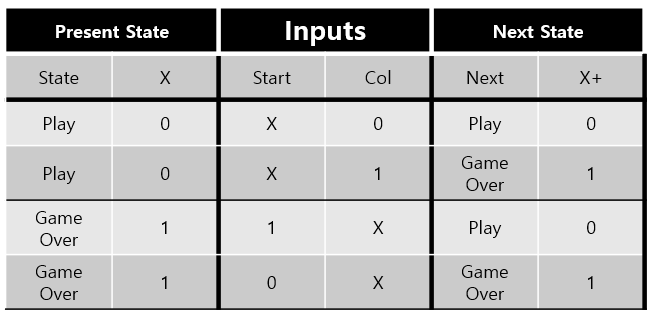
v) Collision State

충돌되는 벽의 종류는 2종류, Free moving ball, paddle의 충돌 범위 4종

|  |  |
| --- | --- |
| State | XYZ |
| Top, Bottom Wall | 000 |
| Right Wall | 001 |
| Paddle 0 | 010 |
| Paddle 1 | 011 |
| Paddle 2 | 100 |
| Paddle 3 | 101 |
| Free Ball | 110 |

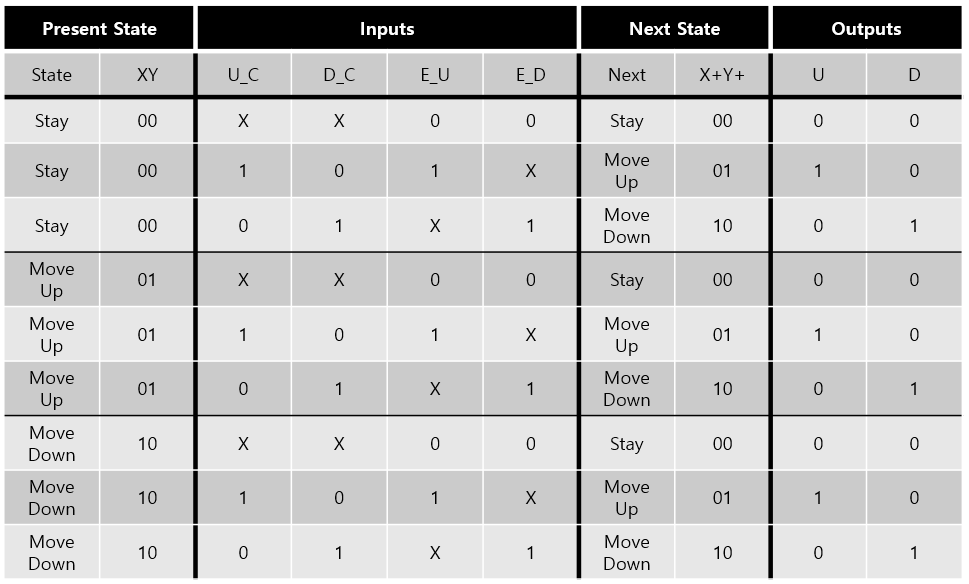
* 1. Choosing Flip-flop
     1. Will use D-FF
  2. State transition table 작성

1. Game State



Game에서 Play시 Output으로 화면과 점수를 출력한다.

ii) Paddle State



iii) Wall Collision State

iv) Paddle Collision State

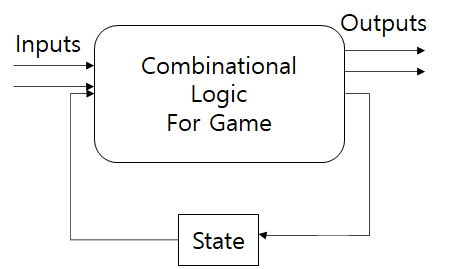
v) Collision State

조교님 살려주세요….저 스파게티 괴물의 Transition state를 어떻게 그려야할까요

* 1. Flip-flop Input Equation and Implementation

i) Game State

X+ = X`C + XS



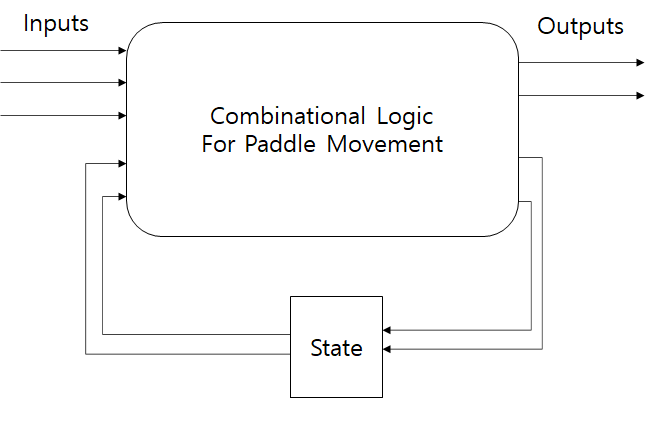
ii) Paddle State

X+ = D\_C&E\_D

Y+ = U\_C&E\_U

U = U\_C&E\_U

D = D\_C&E\_D



iii) Wall Collision State

iv) Paddle Collision State

v) Collision State

1. Required Components
   1. Switch (Player Control)
   2. Switch (Reset) - using button in FPGA
   3. Switch (Start) - using button in FPGA
   4. VGA Cable