12주차 예비보고서

전공: 기계공학과 학년: 3학년 학번: 20191820 이름: 김형준

**1.**

카운터는 2개 이상의 플립플롭으로 구성되어, 매 입력 펄스마다 미리 정해진 순서대로 상태가 주기적으로 변하는 순서 논리회로를 의미한다. 카운트 된 값을 나타내는 2진 비트 수만큼의 플립플롭이 필요하며, 이 플립플롭이 숫자를 저장하는 레지스터의 역할을 한다.

카운터의 계수(modulus of a count)는 카운터가 카운트를 시작해서 완전히 한 바퀴를 돌아 카운트를 끝낸 다음 다시 처음상태로 돌아올 때까지의 상태 수를 의미한다.

카운터는 계수, 타이머, 주파수 계수기 등의 발생 횟수를 세거나, 동작 순서를 제어하기 위한 타이밍 신호 생성에 사용된다.

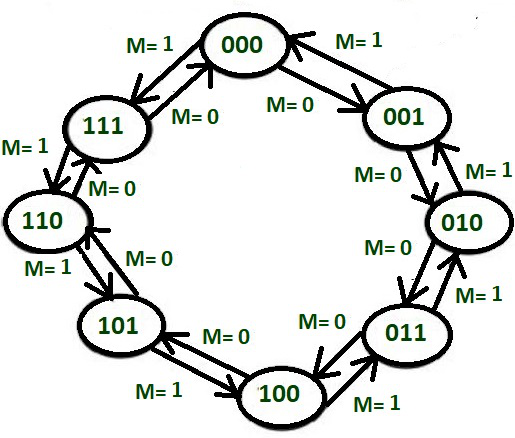
카운터는 클럭 펄스의 인가 방식에 따라 동기식과 비동기식으로 구분할 수 있다.

동기식 카운터는 하나의 클럭 펄스에 모든 플립플롭이 동시에 인가됨으로써 동작하는 카운터를 의미하고, 비동기식 카운터는 공통 클럭을 사용하지 않기 때문에 각각의 플립플롭들의 동작시간이 서로 일치하지 않을 수 있는 카운터를 의미한다.

가장 대표적인 카운터로는 2진 카운터가 있는데, n비트의 2진 카운터도 여러 종류가 존재한다. n-bit Binary Up Counter는 0부터 까지의 이진수를 차례대로 1씩 증가시키는 카운터로, N개의 T플립플롭으로 만들 수 있다. n-bit Binary down counter는 반대로, 부터 0까지의 이진수를 차례대로 1씩 감소시키는 카운터이다. n-bit Binary up/down counter는 앞선 두 counter의 기능을 합친 형태로, 입력값에 따라 Up counter의 기능을 할지, down counter의 기능을 할지를 선택할 수 있는 counter이다.

아래 그림은 3bit Binary up/down counter이다. 입력값 M이 0인 경우 up counter로 작

동하여 000부터 111까지 1씩 올라가는 방향으로 순환하고, 입력값 M이 1인 경우 down counter로 작동하여 111부터 000까지 1씩 내려가는 방향으로 상태가 순환한다.



[3Bit Binary Up/Down Counter의 상태도]

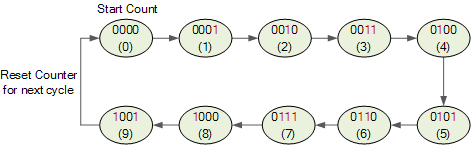
Binary Counter for Other Sequences는 up counter 또는 down counter처럼 이진수를 차례대로 증가/감소 시키는 것이 아닌, 사용자가 정한 임의의 순서대로 설계된 Counter를 의미한다.

이 외에도 0~9까지의 10개의 상태를 카운트하는 BCD Counter, 전체적으로 데이터가 회전하는 Shift Register의 기능을 하는 Ring Counter (맨 마지막 플립플롭의 출력이 첫번째 플립플롭의 입력에 연결되어 있음), 맨 마지막 플립플롭의 출력 중 NOT 출력을 첫번째 플립플롭의 입력과 연결한 Johnson Counter 등이 있다.

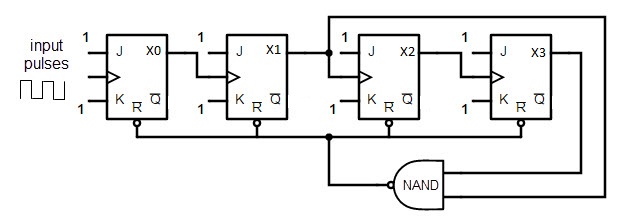
**2.**

Decade Counter는 BCD Counter로도 불리며, 10진수를 4비트로 표현하는 BCD코드를 세는 Counter이다. BCD 코드는 4비트의 이진수 중 0~9를 나타내기 때문에 BCD Counter는 0~9까지의 10가지 상태(State)를 갖고, 4개의 플립플롭으로 구성된다.

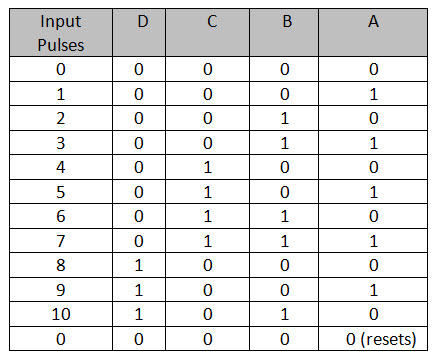
BCD Counter의 각각의 상태는 4비트의 BCD 코드가 된다.



[Decade Counter의 상태도]



[Decade Counter의 회로도]



[Decade Counter의 진리표]

일반적으로 하나의 Counter가 한 자리의 십진수를 표현하기 때문에 4bit ~ 32bit까지 여러 크기를 가지는 이진 Counter와 달리 BCD Counter는 대부분 4bit의 크기를 갖는다. 또한, 여러 자리 수의 십진수에 관한 Counter를 제작할 때 BCD Counter 여러 개를 연결하여 제작할 수 있는데, 위 표에서 1001(9) ->0000(0,resets)로 다시 순환할 때 resets 신호를 다음 자릿수의 BCD Counter에 연결하여 다음 자릿수에 1을 증가시키는 신호를 보낼 수 있다.

위 그림에서 회로도는 비동기식 Decade Counter를 JK 플립플롭 4개를 사용해 만들었는데, Decade Counter는 여러 형태로 구현 될 수 있으므로 다른 형식으로도 구현이 가능하다.

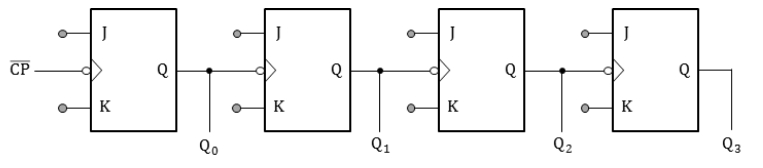
**3.**

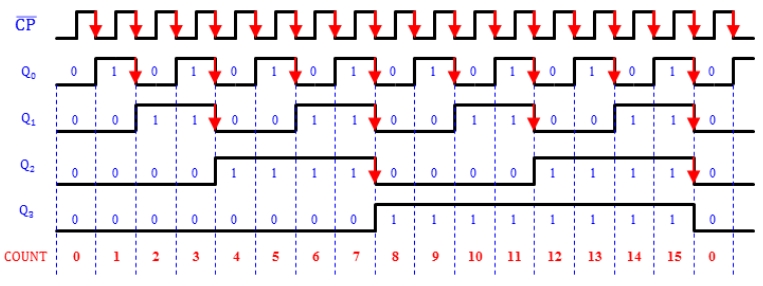
동기식 카운터(synchronous counter)는 하나의 클럭 펄스에 모든 플립플롭이 동시에 인가됨으로써 동작하는 카운터를 의미하며, 모든 플립플롭의 동작시간이 일치된다.

비동기식 카운터(asynchronous counter)는 리플 카운터로도 불리며, 공통 클럭을 사용하지 않기 때문에 각각의 플립플롭들의 동작시간이 서로 일치하지 않을 수 있다.

비동기식 카운터는 일반적으로 클럭 신호가 첫 번째 플립플롭에만 입력되고, 다른 플립플롭은 각 플립플롭의 출력을 다음 플립플롭의 클럭 입력으로 사용한다. 이때, 모든 플립플롭에서 동시에 출력이 발생하는 것이 아니라 이전 플립플롭의 출력에 의해 다음 플립플롭이 동작(Trigger)하기 때문에 전달 지연시간(propagation delay time)이 발생한다.

비동기식 카운터는 동기식 카운터에 비해 회로가 간단하지만, 회로를 구성하는 플립플롭의 개수가 커질수록 전달 지연이 커진다는 단점이 있다.

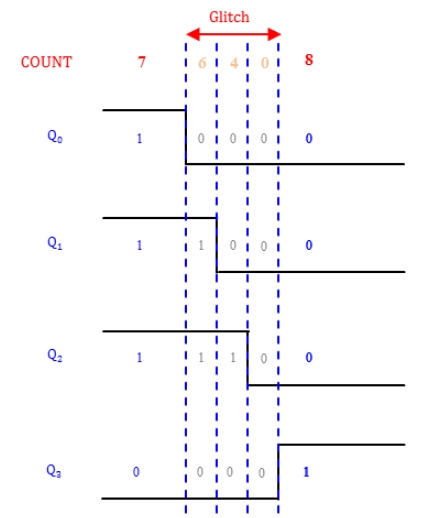
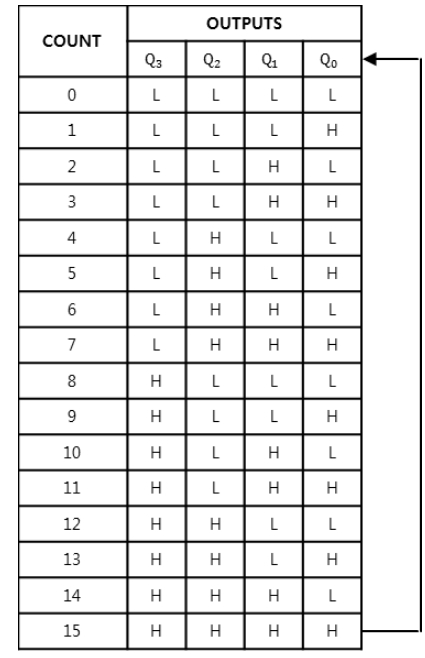




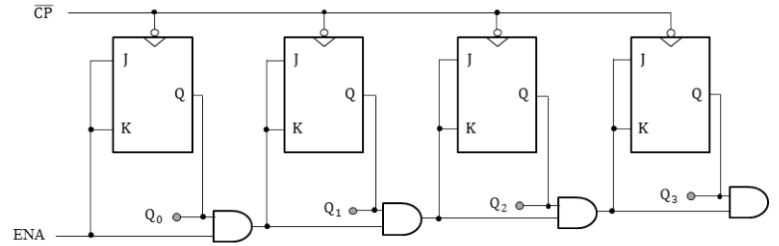
위의 그림은 falling edge에서 trigger되는 비동기식 4bit 상향 카운터의 회로도와 시간 차트이다.

아래의 왼쪽 그림은 진리표이며, 아래의 오른쪽 그림은 플립플롭의 전달 지연시간에 의해 glitch현상이 발생하는 경우를 나타낸 것이다. 아래 그림처럼 7에서 8로 넘어갈 때, 출력 4개가 한번에 동시에 변환되는 것이 아니기 때문에 전달 지연시간 동안 중간 과정의 다른 출력값들이 생성된다.

이러한 glitch현상은 정밀한 시간단위로 작동하는 회로에 영향을 줄 수 있으므로 회로 설계시 delay를 주어 glitch 발생을 방지해야 한다.



아래 그림은 클럭 펄스를 동시에 모든 플립플롭에 인가하는 4bit 동기식 카운터로, 모든 플립플롭이 하나의 공통 클럭 펄스에 의해 Trigger된다. 동기식 카운터는 모든 플립플롭들이 동시에 동작하므로 전달 지연이 작은 장점이 있으며, 잠정적인 중간 상태가 존재하지 않아 글리치 현상이 방지 된다는 장점이 있다. 그러나, 비동기식 카운터에 비해 회로가 복잡해지는 단점이 있다.



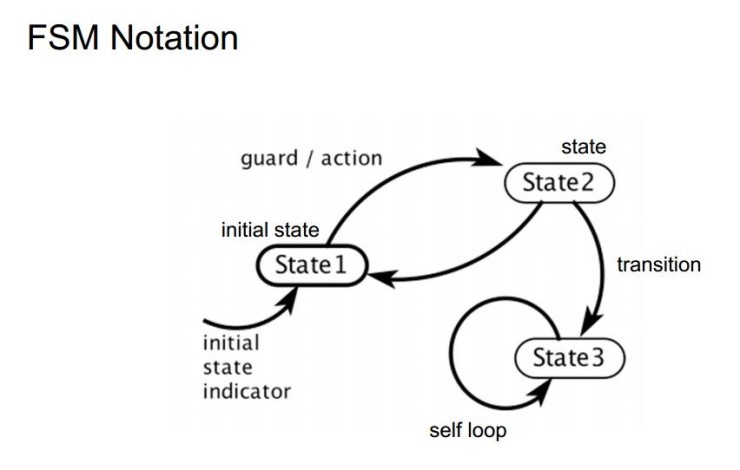
**4.**

FSM(finite state machine)은 유한 상태 기계 또는 유한 오토마타(finite automata)를 의미하며, 유한한 기억장치를 갖는 자동기계에 대한 추상적인 모형이다.

FSM은 기본적으로 내부에 유한한 메모리가 있는 기계에 대한 추상적인 모형으로 사용되며, 디지털 시스템 및 컴퓨터 등에 대한 추상적인 모델로 볼 수 있다.

FSM은 이름대로 유한한 개수의 상태를 가질 수 있으며, 한번에 하나의 상태(State)만을 가질 수 있다. 이 때의 FSM의 상태를 현재 상태(Current State)라고 한다.

FSM의 상태가 변화하는 것을 전이(Transition)이라 하며, 상태간 전이를 유발시키는 사건(입력)을 이벤트(Event)라 하고, 행동(출력)을 내놓는다.



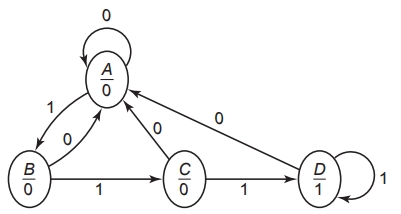
FSM의 각각의 상태들을 하나의 vertex로 두고 상태 간의 이동경로를 edge로 표현하면 위와 같이 그래프가 되는데, 이 그래프를 상태도(state diagram)이라 한다. 상태도를 사용하면 순차회로에서 각각의 상태를 구분하고 입력에 따른 상태변화를 도식화 할 수 있다.

FSM은 머신의 행동을 직관적으로 간단하게 구현할 수 있고 오류의 수정이 용이하며 유연성이 있다는 장점이 있지만, 규모가 커지면 설계가 복잡해지고 제한된 범위의 문제에만 적용이 가능하다는 단점이 있다.

FSM은 출력이 사용자의 입력과 무관하고 현재 상태에 의해서만 출력이 결정되는 Moore FSM과 현재 상태와 입력 모두에 의해 출력이 결정되는 Mealy FSM으로 나뉜다.

**5.**

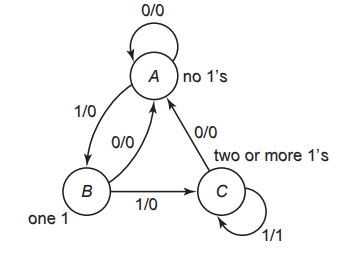
- Moore Machine과 Mealy Machine



Moore Machine은 입력과 관계없이 현재 상태에 의해서만 출력이 결정된다.

위는 3번 연속으로 입력으로 1이 들어왔을 때 출력으로 1을 갖는 Moore Machine이다.

111이라는 입력 패턴을 찾고, 그 결과값을 출력하기 까지 4개의 state가 필요함을 알 수 있는데, 이는 Moore Machine이 state에 출력이 정의 되어있기 때문이다.



Mealy Machine은 입력값과 현재 상태 모두에 의해서 출력이 결정된다. 위는 아까와 같이 3번 연속으로 입력으로 1이 들어왔을 때 출력으로 1을 갖는 Mealy Machine이다.

위의 Moore Machine과 똑같은 기능을 수행하는데 3개의 state만이 필요함을 알 수 있는데, 이는 같은 기능을 수행하더라도 Mealy Machine이 Moore Machine보다 적은 개수의 state를 가지고도 수행할 수 있으며, 사용자의 입력에 더 빠르게 응답이 가능하다는 장점이 존재한다.

그러나, Moore Machine은 설계가 직관적이여서 이해하기 쉽고, Mealy Machine은 이해가 어렵고, 입력에 의해 출력결과가 바뀔 수 있으므로 노이즈나 글리치(Glitch)같은 오동작 요소에 취약하다는 단점이 있다.