컴퓨터공학실험2 12주차 예비 보고서

전공: 컴퓨터공학 학년: 2학년 학번: 20201635 이름: 전찬

**0. 목차**

1. Counter을 조사한다.

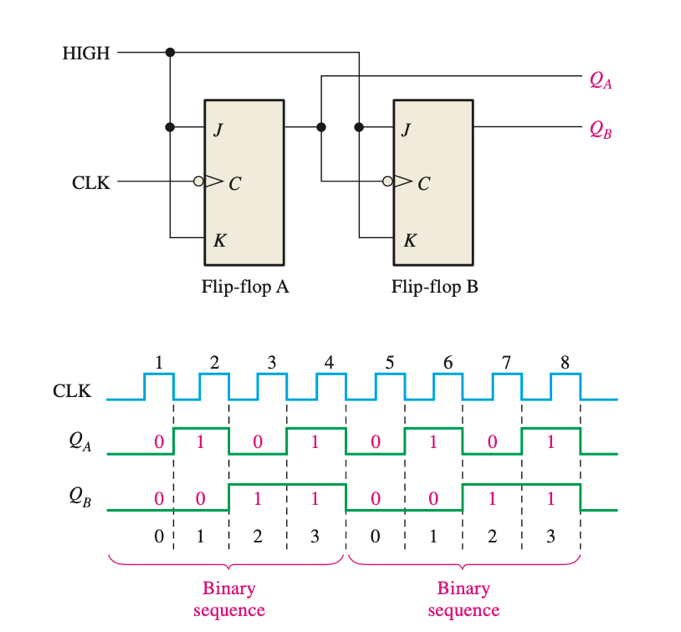
2. Decade Counter을 조사한다.

3. 비동기식 Counter 및 동기식 Counter을 조사한다.

4. FSM(finite-state machine)에 대해 조사한다.

**1. Counter**

Counter은 어떠한 현상이 반복된 횟수를 저장해주는 하나의 장치(회로)이다. 회로에서는 반복되는 현상에 주로 clock pulse를 사용하며, clock pulse를 count한 횟수와 각 clock pulse의 주기를 곱하면 실제 시간을 알아낼 수도 있다. 이는 회로 또는 프로그램에서 실행 시간을 구해내거나, 현재 시각을 7-segment display에 표시할 때 이용되기도 한다. 우리가 Cpp에서 time.h 헤더 파일과 함께 (end-start)/CLOCKS\_PER\_SEC 형태로 실행 시간을 구하는 것도 이에 해당된다. 이를 통해서 프로그램이 몇 초 동안 실행되었는지 파악할 수 있다. 주로 Counter은 JK filp-flop을 이용해 구성한다. 이는 아래와 같다.



<2bit counter와 timing diagram>1

위 결과를 토대로, n bit, 2n-1까지를 count 하는 counter를 만들기 위해서는, n개의 JK flip-flop이 필요하다는 것을 파악할 수 있다.

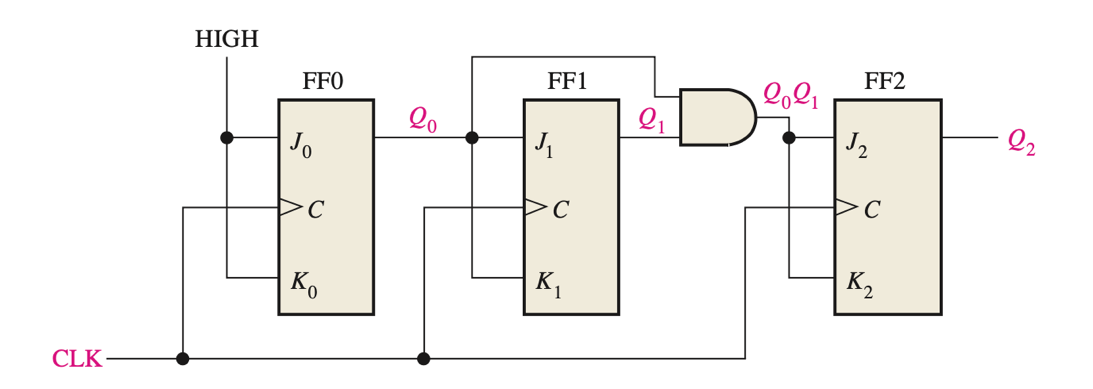
**2. Decade Counter**

BCD가 10진수의 각 자릿수를 4bit 내에 저장하는 형태이다. 이렇게 binary bit를 BCD 형태로 표현하면 여러 가지 장점이 존재하며, Decade Counter 또한 binary bit를 BCD 형태로 표현하는 것이다. 기본적인 decade counter input에 clock pulse가 존재하며, output으로 4bit BCD 형태를 내보내는 것이다. 따라서 0~9까지를 카운트하며, 그 다음 pulse = active(설계에 따라 High/Low)라면, 다시 0이 되는 형태이다. 이렇게 Decade Counter을 구성할 경우 장점이 존재하는데, 이후에 BCD to 7-Segment decoder을 통해서 간단하게 10진수로 변환하여 현재의 counting 횟수를 사용자에게 보여줄 수 있다는 점이 있다. 이에 대한 형태는 기본 counter에 AND / NAND(설계에 따라) gate를 추가한 형태인데, 기존에 구현한 4bit binary counter가 output으로 1010을 갖는다면, 모든 JK flip-flop의 output Q = 0으로 초기화하는 것이다.

**3. 비동기식 Counter 및 동기식 Counter**

모든 회로들은 회로 내부에 존재하는 작은 module 들이 모두 같은 시간에 동작하는지, 각각 다른 시간에 동작하는지(clock pulse에 맞춰서)에 따라서 동기식 / 비동기식 으로 구분할 수 있다. 각각이 장단점이 존재하는데, 동기식은 비동기식에서 일어나는 시간 지연이 줄어든다는 점, 비동기식은 clock pulse가 없어도 실행할 수 있어서 더 실행 시간을 많이 가질 수 있다는 점이 있다.

Counter 또한 비동기식 / 동기식 두 가지 형태로 구현할 수 있다. 우선 비동기식 counter은 위에서 설명한 JK flip-flop을 병렬적으로 이용한 Counter이 대표적이다. 이 형태는 flip-flop의 Clock input에 이전 flip-flop의 output Q를 대입하는 형태이다. 이와 다르게 동기식 Counter은 모든 flip-flop의 활동이 동시에 발생해야 하므로, 현재 clock input에 이전 output을 사용할 수 없으며, 동일한 clock input을 사용해주어야 한다. 3bit synchronous counter의 형태는 아래와 같다.



<3bit synchronous counter의 형태>2

위 회로를 보면 이해하기 간단한 비동기식 counter와는 다르게, 반복적인 구조가 아니라는 것을 파악할 수 있다. 이는 상태도를 통해서 따로 구해야 하는데, 000->001->…->000 형태임을 파악하며, 각 case에 따라서 k-map을 이용해 입력 함수를 표현하며, 이에 맞춰 회로를 구성해야 한다. 따라서 동기식 counter이 비동기식 counter보다 구현하기에 더 어렵지만, 구현한다면 더 적은 전파 지연을 가질 수 있는 구조임을 알 수 있다.

**4. FSM(finite-state machine)**

FSM은 output으로 유한한 상태들 중 단 하나의 상태만을 가질 수 있는 형태로, 현재 존재하는 output과 들어오는 input에 따라서 다음 output을 하나로 결정 할 수 있는 기계를 의미한다. 예를 들면, 게임에서 존재하는 NPC가 있는데, 이 NPC는 플레이어와 상호작용 할 때 플레이어의 행동에 따라서 그에 알맞는 행동을 따라하도록 설계할 수 있다. 이때 각 행동은 이전 NPC의 행동 + 현재 플레이어의 행동을 반영하여 그 input에 맞는 하나의 형태로 나타나게 되는데, 이를 FSM의 한 종류라고 할 수 있다. FSM을 표현하는 방법으로는 Graph를 표현하는 것과 같이, 각 상태(현재의 output)를 vertex, 각 조건(들어오는 input)을 edge로 표현하는 방법이 존재한다. 이렇게 표현할 경우, 일반적인 C style code보다 훨씬 가독성이 쉽다는 장점이 존재하기도 한다.

**6. 출처**

1. Thomas L. Floyd, Person, Digital Fundamentals Eleventh edition(global edition), 414p

2. Thomas L. Floyd, Person, Digital Fundamentals Eleventh edition(global edition), 510p