컴퓨터공학실험2 14주차 예비 보고서

전공: 컴퓨터공학 학년: 2학년 학번: 20201635 이름: 전찬

**0. 목차**

1. FSM에 대해 설명한다.

1-1. Mealy machine을 조사한다.

1-2. Moore machine을 조사한다.

2. Sequence Detector에 대해 조사한다.

3. 기타 이론

**1. FSM**

FSM이란, finite set machine의 약자로 3가지 특징을 가지는 기계(회로)로 그 특징은 아래와 같다.

(1) Sequential circuit 이다.

(2) finite number of states 을 갖는다.

(3) occurring in a specified order 형태이다.

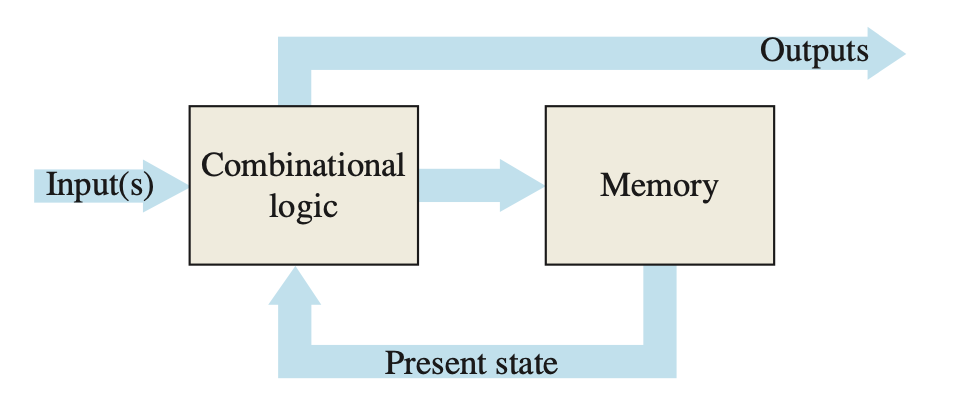
각 경우를 따져보면, (1) sequential circuit은 memory을 사용하며, 이전 상태(이전 상태의 output)이 현재 상태에 하나의 input 형태가 되어 영향을 줄 수 있는 상태를 의미한다. (2) finite number of states는 FSM의 직역과 동일하게, 이전 상태와 현재 들어오는 input에 따라 만들어지는 다음 상태가 finite set 안에 존재하는 것을 의미한다. (3) occurring in a specified order은 (2)에서 설명한 만들어진 상태가 임의로, 여러 가지로 만들어지는 것이 아닌, 미리 정의해 놓은 형태로 하나의 상태만을 갖는 것을 의미한다.

이러한 FSM의 가장 간단한 예시로는, counter이 존재하는데, counter FSM이 실행될 때마다, 이전 결과값에 +1을 해주는 형태라고 할 수 있다. 이때 counter에서 flip-flop의 개수에 따라서 총 count의 범위가 주어지는데, 이때 가능한 결과 상태를 modulus 라고 하기도 한다.(이때 clock pulse는 input으로 고려하지는 않는다.) 또한 현재 상태에서 다음 상태가 달라질 때, 이를 transition 이라고 한다.

이렇게 3가지 특성을 가지는 FSM은 Mealy / Moore 의 2 가지 형태로 분류할 수 있다.

**1-1. Mealy machine**

Mealy machine은 input이 존재하며, input이 다음 상태를 결정하는 데에 영향을 미치는 FSM이라 할 수 있다. 이에 대한 형태는 아래와 같다.

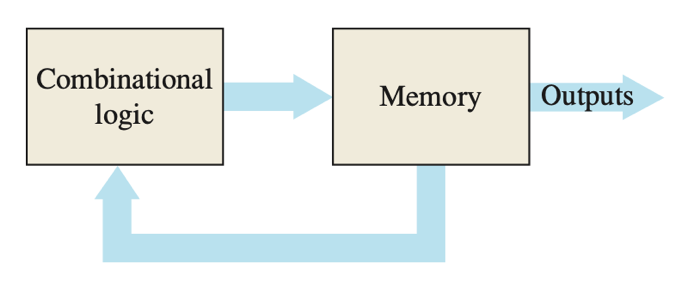


<Mealy machine의 형태>1

위 형태를 보면, input과 함께, memory에 존재하는 이전 상태를 토대로 logic을 거쳐 현재 상태(output)을 만들어내는 형태임을 파악할 수 있다. 간단한 예시로는 1-bit full adder을 생각할 수 있는데, 현재 상태 (0 / 1)와 input (0 / 1)에 따라서 다음 형태가 결정됨을 파악할 수 있다.(이때 full adder FSM의 output은 Sum 이라고 가정한다.)

**1-2. Moore machine**

Moore machine은 Mealy machine과 다르게, input이 존재하지 않으며, 다음 상태의 결정은 현재 상태에만 의존하는 형태이며, 아래와 같다.

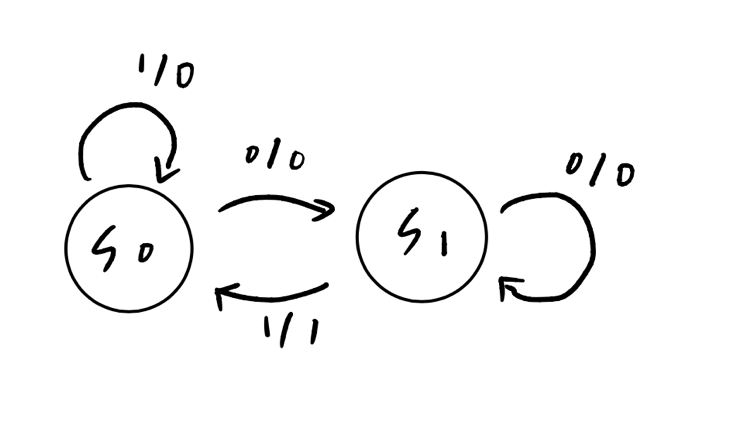


<Moore machine의 형태>2

memory에서 이전 상태를 받아와서 logic을 거쳐 memory에 저장하며, 이를 output으로 가지는 형태이다. 간단한 예시로는 위에서 설명한 counter을 예시로 들 수 있는데, 4-bit ring counter의 경우에는 clock pulse에 따라서 별다른 input 없이 0 -> 1 -> 2 -> 3 -> 0 형태가 반복되는 것을 파악할 수 있다.

**2. Sequence Detector**

sequence / pattern detector은 input로 계속해서 sequence(연속적인 data)가 들어올 때, 원하는 sequence가 input sequence 내에 존재하는지 확인할 수 있는 회로이다. 예를 들어 우리가 확인하고 싶은 sequence = 0110 에 대해 input = 011101…. 이면 output = 000000…, input = 001101…. 이면 output = 000010…. 형태가 되는 회로이다. 이때 output의 이전 상태가 0이라고 해도, 다음 상태에서 1이 나올 수 있다는 것은, 이전 상태를 저장해야 함을 의미하기도 한다. 예를 들어, sequence = 01, input = 101 인 경우에 output = 001이지만, output의 처음 0 과 두 번째 0은 다른 의미라고 할 수 있다. 이때 sequence = 01인 경우에 FSM 형태로 diagram을 만들어낼 수 있는데, 이는 아래와 같다.(edge에는 input/output)



<FSM diagram>

여기에서 위에서 설명한 output = 001에서 맨 처음 0은 S0의 1/0, 두 번째 0은 S0->S1의 0/0, 세 번째 1은 S1->S0의 1/1 임을 파악할 수 있다. 이처럼 sequence detector은 이전 상태와 input을 통해 현재 output을 만들어내는 하나의 Mealy machine 형태라고 할 수 있다.

추가적으로 실제로 sequence detector을 구현할 때, 여러 flip-flop을 토대로 sequence detector을 구현하게 된다. 만약 D flip-flop을 사용한다고 가정하면, k-map을 통한 논리식 간소화를 이용해서 각 J0, K0, J1, K1, ….에 대한 식을 표현하게 되는데, sequence = 01인 경우 아래와 같이 표현할 수 있다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 01 sequence detector truth table | | | | |
| input | 현재 S | 다음 S | flip-flop input | output |
| 0 | 0 | 1 | D = 1 | 0 |
| 0 | 1 | 1 | D = 1 | 0 |
| 1 | 0 | 0 | D = 0 | 0 |
| 1 | 1 | 0 | D = 0 | 1 |

<01 sequence detector truth table>

이를 통해 k-map을 이용하면 D = ~input, output = input&S 형태임을 파악할 수 있다. 이를 통해 회로, 혹은 Verilog code를 구현하면 01 sequence detector을 구현해낼 수 있다.

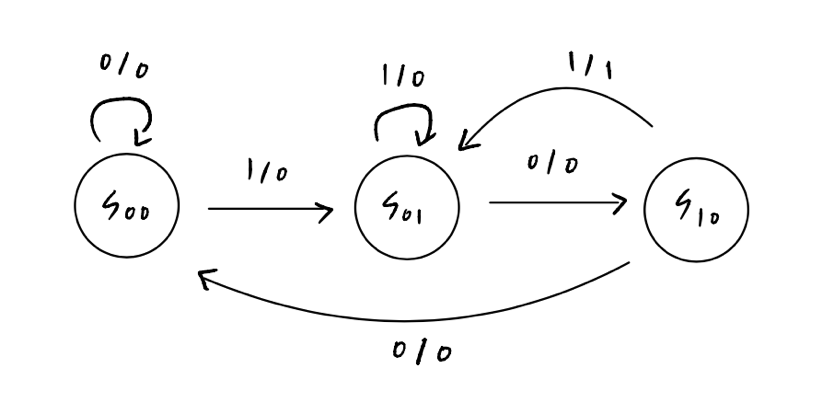
**3. 기타 이론**

위에서 구현한 형태는 sequence detector의 가장 기본적인 형태로, overlapping을 포함한회로이다. 여기에서 overlapping이란, sequence = 111일 때 input = 1111111이라면 output = 0011111 형태로 구현되는 machine을 의미한다. 이와 반대 개념으로 non-overlapping 형태의 sequence detector도 존재하는데, 이는 다음과 같은 과정을 따른다.

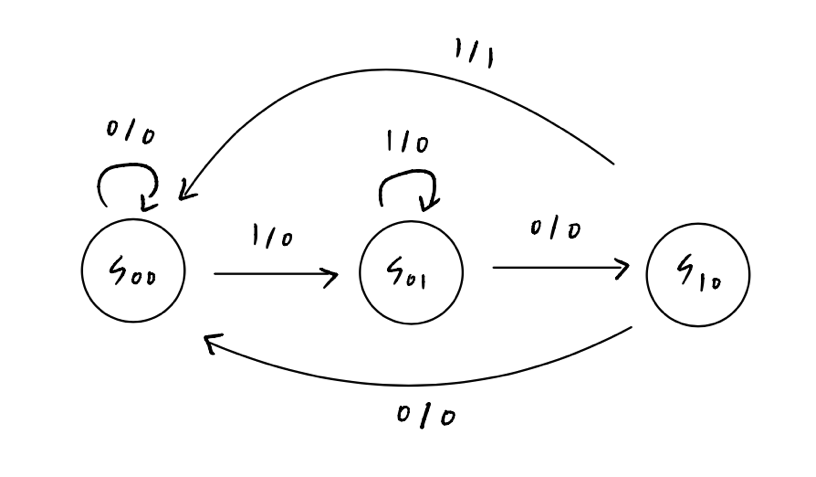
(1) sequence을 찾아내지 못한 경우(output = 0인 경우) 동일한 FSM을 기반으로 한다.

(2) 찾아낸 경우, 맨 처음 상태(감지를 시작하지 않은 상태)로 돌아간다.

만약 input = 101인 경우, diagram은 아래와 같은 형태임을 파악할 수 있다.



<overlapping의 경우>



<non-overlapping의 경우>

non-overlapping의 경우, 101이라는 sequence을 판단한 이후, 1을 이전 상태로 기억하는 것이 아닌, 아예 처음부터 다시 sequence을 검출해내는 형태임을 파악할 수 있다.

**6. 출처**

1. Thomas L. Floyd, Person, Digital Fundamentals Eleventh edition(global edition), 499p

2. Thomas L. Floyd, Person, Digital Fundamentals Eleventh edition(global edition), 499p