14주차 예비보고서

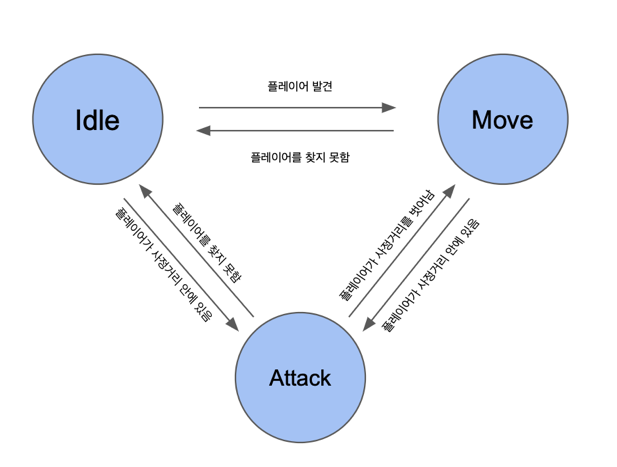
전공: 컴퓨터공학과 학년: 4학년 학번: 20191583 이름: 김태곤

**1. FSM 에 대해서 설명하시오(예시 포함)**

유한 상태 기계(Finite State Machine, FSM)는 주어진 입력에 따라 일련의 상태를 전이하며 출력을 생성하는 계산 모델이다. FSM은 일정한 개수의 상태를 가지고 있으며, 현재 상태와 입력의 조합에 따라 다음 상태와 출력을 결정한다. 이 모델은 시스템의 동작을 모델링하고 제어하는 데 유용하며, 다양한 산업에서 널리 활용된다. FSM의 주요 구성 요소로는 상태, 입력, 출력, 상태 전이, 출력 함수, 초기 상태가 있다. 상태는 시스템이 특정 시점에 있을 수 있는 각 상황을 나타내며, FSM은 항상 하나의 상태에 존재하며 시간의 흐름에 따라 상태가 변화한다. 입력은 시스템에 제공되는 외부 신호나 데이터로서, 상태 전이와 출력을 결정하는 데 중요한 역할을 한다. 출력은 시스템이 외부로 제공하는 신호나 데이터이며, 상태와 입력에 따라 결정된다.

FSM에는 크게 두 가지 유형이 있으며, 무어 머신(Moore Machine)과 밀리 머신(Mealy Machine)이 있다. 무어 머신은 출력이 오직 현재 상태에 의해 결정되고, 밀리 머신은 출력이 현재 상태와 입력의 조합에 의해 결정된다. FSM은 여러 응용 분야에서 중요한 역할을 한다. 제어 시스템에서는 자동화된 기계나 장치의 제어에 FSM을 사용하여 특정 조건에 따라 동작을 제어하며, 통신 시스템에서는 데이터 전송 중 특정 패턴을 감지하고 처리하는 데 사용된다. 또한, 디지털 회로 설계에서는 복잡한 논리 회로에서 상태 기반 제어를 구현하며, 소프트웨어 개발에서는 상태 기반의 동작을 모델링하고 제어하는 데 활용된다. FSM은 시스템의 상태와 행동을 명확하게 정의하고 제어할 수 있는 강력한 도구로서, 시스템의 복잡도를 단순화하고 예측 가능하고 안정적인 동작을 보장할 수 있다. 이러한 특성 덕분에 FSM은 다양한 산업과 기술에서 필수적인 역할을 하며, 시스템 설계의 기본이 된다.

아래는 게임 로직에 관한 간단한 FSM 예시이다.



주어진 FMS은 "Idle", "Move", "Attack"의 세 가지 상태를 가지며, 각각의 상태는 게임 캐릭터나 자동화된 시스템의 동작을 나타낸다. 이 FSM의 상태 전이와 동작을 줄글 형태로 설명하면 다음과 같다.

처음에 시스템은 "Idle" 상태에서 시작한다. "Idle" 상태는 캐릭터가 대기 중이며, 아무런 행동을 하지 않는 상태를 나타낸다. 이 상태에서 캐릭터는 새로운 명령을 기다린다. 만약 캐릭터가 이동 명령을 받으면, "Move" 상태로 전이한다. "Move" 상태는 캐릭터가 이동 중인 상태를 나타내며, 이동이 완료되거나 이동 명령이 취소될 때까지 지속된다. 이 과정에서 캐릭터는 주어진 목적지로 향하거나 경로를 따라 움직인다. 이동 명령이 완료되면 캐릭터는 다시 "Idle" 상태로 돌아간다.

"Idle" 상태에서 캐릭터가 공격 명령을 받으면 "Attack" 상태로 전이한다. "Attack" 상태는 캐릭터가 공격을 수행 중인 상태를 나타내며, 공격이 완료되거나 공격 명령이 취소될 때까지 유지된다. 이 상태에서 캐릭터는 목표물에 대해 물리적 또는 원거리 공격을 수행한다. 공격이 완료되면 캐릭터는 다시 "Idle" 상태로 전이된다.

이 FSM의 설계는 캐릭터가 각 상태에서 적절한 동작을 수행하도록 하며, 상태 전이를 통해 캐릭터의 동작을 제어한다. 예를 들어, 캐릭터가 "Move" 상태에 있을 때 공격 명령을 받으면, 즉시 "Attack" 상태로 전이하여 공격을 시작한다. 마찬가지로, "Attack" 상태에서 이동 명령을 받으면, "Move" 상태로 전이하여 공격을 멈추고 이동을 시작한다. 이러한 상태 전이는 캐릭터의 행동을 유연하게 제어하며, 명령이 들어오는 즉시 반응할 수 있도록 설계되어 있다.

**2. Mealy machine 에 대해 조사하시오.**

밀리 머신(Mealy Machine)은 유한 상태 기계(Finite State Machine, FSM)의 한 유형으로, 출력이 현재 상태와 입력에 의해 동시에 결정되는 방식이다. 이 구조에서 출력은 상태뿐만 아니라 입력에도 직접적으로 의존하기 때문에, 상태가 변하지 않아도 입력의 변화에 따라 출력이 실시간으로 변할 수 있다. 밀리 머신은 제어 시스템, 통신 시스템, 그리고 디지털 회로 설계에서 널리 사용된다.

밀리 머신의 동작 원리를 이해하기 위해 구성 요소들을 먼저 살펴보자. 밀리 머신의 주요 구성 요소는 입력, 조합 논리, 플립플롭, 출력 조합 논리, 클럭 신호, 그리고 출력으로 이루어진다. 입력은 시스템에 제공되는 외부 신호 또는 데이터로서, 상태 전이에 영향을 미치고 출력에도 직접적인 영향을 준다. 입력 신호는 조합 논리를 통해 처리되며, 조합 논리는 현재 상태와 입력을 받아 다음 상태와 출력을 결정하는 논리 회로이다. 이 조합 논리는 입력 신호와 현재 상태를 기반으로 새로운 상태와 출력을 계산한다.

계산된 새로운 상태 정보는 플립플롭에 의해 저장된다. 플립플롭은 클럭 신호에 동기화되어 상태를 저장하는 기억 장치로, 조합 논리에서 제공된 다음 상태 정보를 저장하며, 클럭 신호에 의해 상태가 갱신된다. 이 저장된 상태 정보는 다음 출력과 상태 전이를 결정하는 데 사용된다. 클럭 신호는 플립플롭의 상태 전이를 동기화하는 주기적인 신호로, 플립플롭이 새로운 상태 정보를 받아들이고 이를 출력으로 연결하도록 동기화하는 역할을 한다.

출력은 현재 상태와 입력을 기반으로 출력 조합 논리에서 생성된다. 이 논리는 플립플롭에 저장된 현재 상태 정보와 실시간 입력을 받아 해당 상태와 입력에 따른 출력을 결정한다. 밀리 머신의 출력은 현재 상태와 입력에 의해서만 결정되며, 입력이 변화할 때마다 출력이 실시간으로 변경될 수 있다. 이러한 데이터 흐름은 입력 신호가 조합 논리에 들어가고, 현재 상태와 결합하여 다음 상태와 출력을 결정하는 것으로 시작된다. 그 다음, 조합 논리에서 결정된 다음 상태가 플립플롭으로 전송되고, 플립플롭에 저장된 현재 상태와 실시간 입력이 출력 조합 논리에 입력으로 제공된다. 마지막으로, 출력 조합 논리는 현재 상태와 입력을 기반으로 출력을 생성한다.

밀리 머신은 여러 장점을 갖고 있다. 첫째, 출력의 반응 속도가 빠르다. 출력이 상태와 입력에 의해 동시에 결정되기 때문에 입력의 변화에 실시간으로 반응할 수 있다. 둘째, 메모리 사용량이 효율적이다. 상태 전이와 출력이 함께 계산되므로 출력만을 위해 추가적인 메모리가 필요하지 않다. 셋째, 디지털 회로 설계에서 효율적으로 사용될 수 있다. 입력 변화에 따라 즉각적인 반응이 필요한 경우에 유리하다.

밀리 머신은 다양한 응용 분야에서 활용된다. 예를 들어, 제어 시스템에서는 즉각적인 출력 반응이 필요한 경우에 적합하며, 통신 시스템에서는 데이터 전송 중 실시간 오류 검출 및 수정에 사용된다. 또한, 디지털 회로 설계에서는 실시간 입력 변화에 따른 반응이 요구되는 복잡한 논리 회로에서 상태 기반 제어를 구현하는 시스템으로 활용된다.

텍스트, 도표, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

원, 도표, 텍스트, 폰트이(가) 표시된 사진

자동 생성된 설명

**3. Moore machine 에 대해 조사하시오.**

무어 머신(Moore Machine)은 유한 상태 기계(Finite State Machine, FSM)의 한 유형으로, 출력이 현재 상태에 의해서만 결정되는 방식이다. 이 구조에서 출력은 입력에 직접적으로 의존하지 않고, 오직 현재 시스템의 상태와 연결되어 있으며, 상태가 변경될 때만 출력이 변화한다. 무어 머신은 주로 제어 시스템, 통신 시스템, 그리고 디지털 회로 설계에서 사용된다.

무어 머신의 동작 원리를 이해하기 위해서는 구성 요소들을 먼저 살펴봐야 한다. 무어 머신의 주요 구성 요소는 입력, 조합 논리, 플립플롭, 출력 조합 논리, 클럭 신호, 그리고 출력으로 이루어진다. 입력은 시스템에 제공되는 외부 신호 또는 데이터로서, 상태 전이에 영향을 주지만 출력에는 직접적인 영향을 미치지 않는다. 입력 신호는 조합 논리를 통해 처리되며, 조합 논리는 현재 상태와 입력을 받아 다음 상태를 결정하는 논리 회로이다. 이 조합 논리는 입력 신호와 현재 상태를 기반으로 새로운 상태 정보를 계산한다.

계산된 새로운 상태 정보는 플립플롭에 의해 저장된다. 플립플롭은 클럭 신호에 동기화되어 상태를 저장하는 기억 장치로, 조합 논리에서 제공된 다음 상태 정보를 저장하며, 클럭 신호에 의해 상태가 갱신된다. 이 저장된 상태 정보가 다음 출력과 상태 전이를 결정하는 데 사용된다. 클럭 신호는 플립플롭의 상태 전이를 동기화하는 주기적인 신호로, 플립플롭이 새로운 상태 정보를 받아들이고 이를 출력으로 연결하도록 동기화하는 역할을 한다.

출력은 현재 상태를 기반으로 출력 조합 논리에서 생성된다. 이 논리는 플립플롭에 저장된 현재 상태 정보를 입력으로 받아 해당 상태에 대한 출력을 결정한다. 무어 머신의 출력은 현재 상태에 의해서만 결정되며, 상태가 변화할 때만 출력이 변경된다. 이러한 데이터 흐름은 입력 신호가 조합 논리에 들어가고, 현재 상태와 결합하여 다음 상태를 결정하는 것으로 시작된다. 그 다음, 조합 논리에서 결정된 다음 상태가 플립플롭으로 전송되고, 플립플롭에 저장된 현재 상태는 출력 조합 논리에 입력으로 제공된다. 마지막으로, 출력 조합 논리는 현재 상태에 기반하여 출력을 생성한다.

무어 머신은 여러 장점을 갖고 있다. 첫째, 예측 가능성이 높다. 출력이 상태에 의해서만 결정되기 때문에 시스템의 출력이 상태 변화에 따라 일정하게 유지된다. 둘째, 디버깅이 용이하다. 상태에 따른 출력이 명확히 구분되므로 시스템의 상태를 쉽게 추적하고 디버깅할 수 있다. 셋째, 설계가 직관적이다. 상태 다이어그램을 통해 시스템의 상태와 출력을 쉽게 설계하고 이해할 수 있다.

무어 머신의 이러한 특성 때문에 다양한 응용 분야에서 활용된다. 예를 들어, 제어 시스템에서는 특정 조건에 따라 기계를 제어하는 데 사용되며, 통신 시스템에서는 데이터 전송 중 특정 패턴을 감지하고 처리하는 데 사용된다. 또한, 디지털 회로 설계에서는 복잡한 논리 회로에서 상태 기반 제어를 구현하는 시스템으로 활용된다.

텍스트, 도표, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 원, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

**4. Sequence Detector 에 대해 조사하시오.**

시퀀스 디텍터(Sequence Detector)는 디지털 시스템에서 특정 비트 패턴을 입력 데이터 스트림에서 인식하고, 해당 패턴이 발견되면 신호를 출력하는 순차 상태 기계이다. 시퀀스 디텍터의 주요 기능은 입력 비트 스트림에서 사전에 정의된 패턴을 감지하는 것으로, 예를 들어 "1101"과 같은 특정 비트 열을 인식하여 해당 패턴이 검출되었을 때 특정 출력 신호를 활성화한다. 이 검출 방식은 통신, 제어 시스템, 데이터 처리와 같은 다양한 분야에서 유용하게 사용된다.

시퀀스 디텍터는 입력 신호, 상태 레지스터, 조합 논리, 출력 신호, 상태 전이 함수, 클럭 신호와 같은 여러 구성 요소로 이루어져 있다. 입력 신호는 디텍터가 분석할 데이터 스트림으로, 0과 1의 연속적인 비트로 구성된다. 이 스트림은 조합 논리에서 분석되어 패턴을 찾는 데 사용된다. 상태 레지스터는 현재 상태를 저장하는 플립플롭이나 레지스터로 구성되어 있으며, 현재 입력과 과거의 입력 비트들이 반영된 상태를 저장하여 다음 상태를 결정하는 데 중요한 역할을 한다.

조합 논리는 현재 상태와 입력 비트를 조합하여 다음 상태와 출력을 결정하는 논리 회로로, 입력 신호와 현재 상태를 기반으로 다음 상태와 출력 신호를 결정한다. 출력 신호는 시퀀스가 검출되었음을 나타내는 신호로서, 미리 정의된 패턴이 감지되면 이를 알린다. 상태 전이 함수는 현재 상태와 입력 비트에 따라 다음 상태를 결정하며, 상태 레지스터를 업데이트하는 역할을 한다. 클럭 신호는 상태 레지스터의 업데이트를 동기화하는 주기적인 신호로, 입력 데이터 스트림의 각 비트를 처리할 때마다 상태를 업데이트한다.

시퀀스 디텍터의 동작 원리는 초기 상태 설정, 입력 비트 처리, 상태 전이, 출력 결정, 패턴 검출로 이루어진다. 먼저 시스템이 초기 상태에서 시작하며, 이 상태는 아무 비트도 검출되지 않은 상태이다. 입력 데이터 스트림의 비트들이 순차적으로 들어오고, 각 클럭 주기마다 입력 비트가 하나씩 처리된다. 조합 논리는 현재 상태와 입력 비트를 분석하여 다음 상태를 결정하며, 상태 전이 함수는 현재 상태와 입력 비트를 기반으로 다음 상태를 계산한다. 조합 논리는 현재 상태와 입력 비트를 사용하여 패턴이 검출되었는지를 확인하고, 검출되었을 때 출력 신호를 활성화한다. 미리 정의된 패턴이 입력 스트림에 나타나면 시퀀스 디텍터는 출력 신호를 통해 이를 알린다.

시퀀스 디텍터는 오버랩(Overlapping)과 논오버랩(Non-Overlapping) 모드로 동작할 수 있다. 오버랩 모드에서는 패턴이 검출된 후 새로운 패턴 검출이 직후에 시작될 수 있으며, 패턴의 일부가 다음 패턴 검출의 일부로 중첩될 수 있다. 이러한 방식은 연속적인 패턴을 탐지할 때 유용하다. 반면 논오버랩 모드에서는 패턴이 검출된 후 패턴의 끝 다음 비트부터 새로운 패턴 검출을 시작하며, 중복 검출이 방지된다. 예를 들어, 입력 비트 스트림에서 "101" 패턴을 검출할 때 오버랩 방식에서는 첫 번째 "101"이 검출된 후 바로 다음 "101" 검출이 시작되지만, 논오버랩 방식에서는 첫 번째 "101"이 검출된 후 해당 패턴의 끝 다음 비트부터 새로운 "101" 검출이 시작된다.

시퀀스 디텍터는 디지털 신호의 특정 패턴을 감지하여 다양한 응용에서 중요한 역할을 하며, 시스템의 요구에 따라 오버랩 또는 논오버랩 모드로 구성될 수 있다. 이 기능을 통해 복잡한 데이터 스트림에서 유용한 정보를 추출하고, 실시간으로 신호를 처리할 수 있다.

도표, 폰트, 원, 스케치이(가) 표시된 사진

자동 생성된 설명

1001을 검출하는 밀리 머신 Sequence Detector

**5. 기타이론**

무어 머신을 사용한 시퀀스 디텍터는 디지털 시스템에서 특정 비트 패턴을 인식하고, 해당 패턴이 입력 데이터 스트림에서 감지되었을 때 신호를 출력하는 순차 상태 기계이다. 이 시퀀스 디텍터는 입력 비트 스트림에서 사전에 정의된 패턴을 찾아내며, 특정 상태에 도달했을 때 이를 감지하여 출력 신호를 생성한다. 무어 머신의 특성상 출력은 현재 상태에 의해서만 결정되므로 상태 전이와 출력을 쉽게 설계할 수 있으며, 예측 가능한 동작을 제공한다.

아래 내용은 무어 머신을 적용하여 0001을 검출하는 Sequence Detector이다.

스케치, 도표, 원, 그림이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명

이 시퀀스 디텍터는 다섯 개의 상태(S0, S1, S2, S3, S4)로 구성되어 있으며, 각 상태는 특정 비트 패턴의 부분을 인식하도록 설계되어 있다. 상태 다이어그램에서 초기 상태는 S0로 시작하며, 입력 비트에 따라 상태가 전이된다. 초기 상태 S0에서는 입력 비트가 0일 경우 상태 S0를 유지하고, 입력 비트가 1일 경우 상태 S1으로 전이된다. 이와 같은 방식으로 상태가 전이되며, 각 상태는 입력 비트의 조합에 따라 다음 상태로의 전이 경로가 정해져 있다.

상태 S1에서는 입력이 0일 때 상태 S2로 전이되며, 입력이 1일 때 상태 S1을 유지한다. 상태 S2에서는 입력이 0일 때 상태 S3으로 전이되며, 입력이 1일 때 상태 S1으로 되돌아간다. 상태 S3에서는 입력이 0일 때 상태 S0으로 전이되고, 입력이 1일 때 상태 S4로 전이된다. 마지막으로 상태 S4에서는 출력이 1로 설정되며, 이는 특정 패턴이 감지되었음을 의미한다. 상태 S4에서는 입력이 0일 때 상태 S2로 전이되며, 입력이 1일 때 상태 S1으로 전이된다.

상태 전이 표는 상태 다이어그램을 기반으로 상태 전이와 출력을 명확히 정리한 것으로, 현재 상태와 입력에 따라 다음 상태와 출력을 나타낸다. 예를 들어, 상태 S0에서 입력이 0이면 상태 S0를 유지하고 출력은 0이며, 입력이 1이면 상태 S1으로 전이되고 출력은 여전히 0이다. 이러한 상태 전이와 출력은 모든 상태에서 입력에 따라 정의되며, 상태 S4에서 출력이 1로 설정되는 것을 제외하고 나머지 상태에서는 출력이 0으로 유지된다.

시퀀스 디텍터의 동작은 초기 상태에서 시작하여 입력 비트 스트림을 처리하며 상태가 갱신된다. 각 클럭 주기마다 입력 비트가 하나씩 처리되고, 조합 논리는 현재 상태와 입력 비트를 분석하여 다음 상태와 출력을 결정한다. 미리 정의된 패턴이 입력 스트림에서 검출되면 시퀀스 디텍터는 상태 S4에 도달하여 출력 신호를 1로 설정한다. 이는 특정 비트 패턴이 감지되었음을 의미하며, 이후 입력 비트에 따라 상태가 다시 갱신되어 새로운 패턴 검출을 시작한다.