14주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211522 이름: 김정환

**1.**

.................

Sequence Detector 1101을 Moore machine으로 구현하는 경우 입력된 시점이 아닌 저장 후 다음 입력이 들어올 때 결과값을 출력하는 방식으로 이루어지게 된다. 이를 Verilog 코드로 구현하면 아래와 같다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이는 rst가 1일 때는 4bit를 저장하는 out은 0000으로, 출력인 z도 0으로 초기화한다. 그 외에 clk이 들어올 때는 입력인 in을 저장하는데, 이때 먼저 저장된 4bit가 1101인지 if믄을 통해 확인 후 맞으면 z를 1, 이외에는 0으로 출력한다. 그 후 out을 왼쪽으로 1 shift 연산하고 out[0]에 in을 저장하여 입력을 갱신한다. 또한 이외에 처리는 없으므로 overlapping, 즉 중첩되어서도 맞는 sequence인 경우 1을 출력하게 된다. 이를 Simulation하기 위한 Testbench 코드는 아래와 같다.

텍스트, 스크린샷, 영수증이(가) 표시된 사진

자동 생성된 설명

Simulation 결과는 아래와 같다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이를 확인하면 위에서 out이 1101이 되고 난 후에 다음 입력에 1이 출력되는 점을 확인할 수 있다. 또한 이어서 1101이 바로 만들어지는 경우에도 다음 입력에 1을 출력하는 것으로 Overlapping 방식이 구현된 점도 확인할 수 있었다.

이에 대해서 State Table과 State Diagram을 작성하면 아래와 같이 나타낼 수 있다.

우선 State Diagram은 다음과 같다.

그림, 스케치, 라인 아트, 클립아트이(가) 표시된 사진

자동 생성된 설명

아무 입력이 없는 초기 상태와 같은 A부터 시작하여 각각에 맞는 입력이 들어올 때마다 다음 상태로 이동하는데, 이를 State Table로 나타내면 아래와 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| Present  State | Next State | | Output  z |
| in=0 | in=1 |
| A | A | B | 0 |
| B | A | C | 0 |
| C | D | C | 0 |
| D | A | E | 0 |
| E | A | C | 1 |

................

**2.**

.......................

Sequence Detector 10101을 Mealy Machine으로 구현하는 경우 아래와 같은 Verilog 코드로 나타낼 수 있다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

rst가 1일 때는 5 bit를 저장하는 out을 00000, 출력인 z를 0으로 초기화한다. 이 외에는 out을 왼쪽으로 한 번 Shift 연산 후에 입력인 in을 넣어주고 이 값이 10101이 나오는 경우에 z가 1이 되도록 하고, 이외에는 0이 되도록 한다. clock은 Rising edge에서 trigger되도록 구현했다. 이 경우 역시도 중첩에 대해 처리가 없으므로 Overlapping 방식이다. 이를 Simulation하기 위한 Testbench 코드는 아래와 같다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

Simulation 결과는 아래와 같다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

결과를 살펴보면 1010이 저장되어 있는 상태에서 1의 입력이 들어올 때 z가 1이 되는 것을 볼 수 있고, 이후에도 이어서 입력으로 0, 1이 들어왔을 때 z가 다시 1이 나와 Overlapping 방식임도 확인할 수 있다. 이외의 경우에는 z가 0으로 정상적으로 출력된다. 이에 대해서 State Diagram과 State Table을 작성하면 아래와 같다. State Diagram은 아래와 같다.

그림, 스케치, 라인 아트, 원이(가) 표시된 사진

자동 생성된 설명

다음으로 State Table은 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Present  State | Next State | | Output z | |
| in=0 | in=1 | in=0 | in=1 |
| A | A | B | 0 | 0 |
| B | C | B | 0 | 0 |
| C | A | D | 0 | 0 |
| D | E | B | 0 | 0 |
| E | A | B | 0 | 1 |

다음으로 Moore Machine으로 구현하는 경우 Verilog 코드는 아래와 같다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이는 이외 부분은 Mealy Machine과 유사하지만, 차이점으로는 rst가 1이 아닌 경우에서 out에 10101이 저장되어 있는지 확인하고 맞을 경우 z를 1로 출력하고 아니면 0으로 출력한 후에 out을 Shift 연산 후 입력인 in을 넣어주는 부분이 있다. 결과를 확인하기 위한 Testbench 코드는 아래와 같다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

이를 사용한 Simulation 결과는 아래와 같다.

스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

이를 살펴보면 처음 10101이 out에 저장된 후 다음 입력이 들어올 때 1을 출력하는 것을 확인할 수 있고, 1이 한 번 더 들어옴으로써 그전 입력 0과 합쳐져 다시 10101이 되었을 때 다음 입력에 z가 1이 되는 것을 볼 수 있다. 즉, 앞서 구현한 Mealy Machine보다 한 입력 뒤에 결과가 나온다. 이에 대해서 State Diagram과 State Table을 나타내면 아래와 같다. 우선 State Diagram은 아래와 같다.

그림, 스케치, 라인 아트, 클립아트이(가) 표시된 사진

자동 생성된 설명

State Table은 아래와 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| Present  State | Next State | | Output  z |
| in=0 | in=1 |
| A | A | B | 0 |
| B | C | B | 0 |
| C | A | D | 0 |
| D | E | B | 0 |
| E | A | F | 0 |
| F | E | B | 1 |

.........................