14주차 결과보고서

전공: 아트&테크놀로지학과 학년: 4학년 학번: 20191048 이름: 김도솔

**1.**

**Sequence Detector 1101 Moore machine (Overlapping)**

* Verilog source code

|  |  |
| --- | --- |
| Design Source Code | Simulation Source Code |
| `timescale 1ns / 1ps  module seq\_moore(  input clk, rst, in,  output reg out  );  reg[3:0] seq;  reg[3:0] x = 4'b1101;  initial out = 1'b0;  always @(posedge clk) begin  if(rst) begin  seq <= 4'b0000;  out <= 1'b0;  end  else begin  seq <= seq << 1;  seq[0] <= in;  end  if(seq == x) out <= 1'b1;  else out <= 1'b0;  end  endmodule | `timescale 1ns / 1ps  module seq\_moore\_tb;  reg clk, rst, in;  wire out;  seq\_moore u\_seq\_moore (  .clk (clk ),  .rst (rst ),  .in (in ),  .out (out )  );  initial begin  clk = 1'b0;  rst = 1'b0;  in = 1'b0;  end  always #10 clk = ~clk;  always #200 rst = ~rst;  always@(in) begin  in = #10 ~in;  in = #10 ~in;  in = #10 ~in;  in = #30 ~in;  in = #10 ~in;  in = #10 ~in;  in = #10 ~in;  in = #20 ~in;  in = #10 ~in;  in = #40 ~in;  in = #30 ~in;  in = #30 ~in;  in = #10 ~in;  in = #20 ~in;  in = #10 ~in;  in = #30 ~in;  in = #10 ~in;  in = #10 ~in;  in = #10 ~in;  in = #10 ~in;  in = #20 ~in;  end  initial begin  #1000  $finish;  end  endmodule |

위는 Verilog 코드로 Overlapping 방식의 Sequence Detector 1101 Moore machine을 구현한 것이다. seq\_moore모듈은 3개의 입력(clk, rst, in)과 1개의 출력(out)을 가지고 있다. 이때 clk는 클럭 신호, rst는 리셋 신호, in은 1 bit 입력, out은 1 bit 출력을 의미한다. 또한 4비트 레지스터 seq와 x를 정의한다. seq는 무어 머신에 들어오는 가장 최근 4 bit의 sequence를 저장하기 위한 용도이고 x는 detect하고 싶은 sequence로 초기값을 4'b1101로 설정했다. 클럭의 상승 에지에서 리셋 신호가 활성화되면(rst가 1이면) seq는 0000으로, out은 0으로 초기화한다. 그렇지 않으면 seq를 한 비트 왼쪽으로 시프트하고, 가장 오른쪽 비트를 in으로 설정한다. 그 후, seq가 x와 같으면 out을 1로, 그렇지 않으면 0으로 설정한다. 따라서 이 모듈은 입력 신호 in에서 주어진 시퀀스 x와 일치하는 패턴을 감지하고, 일치할 경우 출력 신호 out을 1로 설정하는 Overlapping 방식의 Sequence Detector로 동작할 수 있다. 또한 각 변수에 값 할당 시 nonblocking 방식(<=)을 사용해 Moore machine으로 동작할 수 있도록 구현했다.

* Schematic Diagram

도표, 텍스트, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명

위에서 작성한 Verilog 코드로 Schematic Diagram을 생성한 결과이다. 이를 통해 작성한 코드가 회로에 어떤 식으로 구현되는지 알 수 있다.

* Simulation 결과

스크린샷이(가) 표시된 사진

자동 생성된 설명

위에서 작성한 Verilog 코드를 시뮬레이션 돌린 결과이다. 시간에 따라 입력 clk, rst, in의 값을 바꾸면 그에 따라 출력 out의 값이 변화하는 것을 확인할 수 있다. 클럭의 상승 에지 마다 들어오는 입력이 순차적으로 1101이 되면 out이 1을 출력한다. 이때 Moore machine으로 구현하였기에 sequence 1101이 입력으로 들어오고 한 클럭 뒤에 1을 출력하는 것을 확인할 수 있다. 또한 overlapping 방식으로 구현되어 1101101이 들어오면 1101을 두 번 detect해 out이 1101에서 한 번, 1101101에서 한 번 총 두 번 1을 출력한다. 마지막으로 rst가 1이면, 즉 리셋 신호가 활성화 되어 있으면 1101이 들어오더라도 out은 0을 출력하는 것을 관찰할 수 있다.

* State Table

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

A = 초기 상태, B = 1, C = 11, D = 110, E = 1101

위는 Overlapping 방식으로 구현된 Sequence Detector 1101 Moore machine의 상태표를 작성한 것이다. state A는 초기 상태를 나타내고 state B는 1이 detect 되었을 때, state C는 11이 detect 되었을 때, state D는 110이 detect 되었을 때, state E는 1101이 detect 된 상태를 나타낸다. Mealy machine과는 달리, Moore machine에서 출력은 현재 상태에만 종속되는 것을 확인할 수 있다. overlapping 방식으로 구현되었기 때문에 state E에서 input으로 1이 들어오면 초기 상태를 나타내는 state A가 아닌 11이 detect된 상태인 state C로 넘어가는 것을 확인할 수 있다. 또한 1101이 detect된 state E에서 input 값이 새로 들어올 때 output이 1로 업데이트되므로 출력이 delay되는 것을 알 수 있다.

* State Diagram

그림, 스케치, 라인 아트, 클립아트이(가) 표시된 사진

자동 생성된 설명

**2.**

**1) Sequence Detector 10101 Mealy machine (Overlapping)**

* Verilog source code

|  |  |
| --- | --- |
| Design Source Code | Simulation Source Code |
| `timescale 1ns / 1ps  module seq\_mealy(  input clk, rst, in,  output reg out  );  reg[4:0] seq;  reg[4:0] x = 5'b10101;  initial out = 1'b0;  always @(posedge clk) begin  if(rst) begin  seq = 5'b00000;  out = 1'b0;  end  else begin  seq = seq << 1;  seq[0] = in;  end  if(seq == x) out = 1'b1;  else out = 1'b0;  end  endmodule | `timescale 1ns / 1ps  module seq\_mealy\_tb;  reg clk, rst, in;  wire out;  seq\_mealy u\_seq\_mealy (  .clk (clk ),  .rst (rst ),  .in (in ),  .out (out )  );  initial begin  clk = 1'b0;  rst = 1'b0;  in = 1'b0;  end  always #10 clk = ~clk;  always #200 rst = ~rst;  always@(in) begin  in = #10 ~in;  in = #10 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #10 ~in;  in = #40 ~in;  in = #30 ~in;  in = #10 ~in;  in = #10 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  end  initial begin  #1000  $finish;  end  endmodule |

위는 Verilog 코드로 Overlapping 방식의 Sequence Detector 10101 Mealy machine을 구현한 것이다. seq\_mealy모듈은 3개의 입력(clk, rst, in)과 1개의 출력(out)을 가지고 있다. 이때 clk는 클럭 신호, rst는 리셋 신호, in은 1 bit 입력, out은 1 bit 출력을 의미한다. 또한 5비트 레지스터 seq와 x를 정의한다. seq는 무어 머신에 들어오는 가장 최근 5 bit의 sequence를 저장하기 위한 용도이고 x는 detect하고 싶은 sequence로 초기값을 5'b10101로 설정했다. 클럭의 상승 에지에서 리셋 신호가 활성화되면(rst가 1이면) seq는 00000으로, out은 0으로 초기화한다. 그렇지 않으면 seq를 한 비트 왼쪽으로 시프트하고, 가장 오른쪽 비트를 in으로 설정한다. 그 후, seq가 x와 같으면 out을 1로, 그렇지 않으면 0으로 설정한다. 따라서 이 모듈은 입력 신호 in에서 주어진 시퀀스 x와 일치하는 패턴을 감지하고, 일치할 경우 출력 신호 out을 1로 설정하는 Overlapping 방식의 Sequence Detector로 동작할 수 있다. 또한 각 변수에 값 할당 시 blocking 방식(=)을 사용해 Mealy machine으로 동작할 수 있도록 구현했다.

* Schematic Diagram

도표, 라인, 텍스트, 평면도이(가) 표시된 사진

자동 생성된 설명

위에서 작성한 Verilog 코드로 Schematic Diagram을 생성한 결과이다. 이를 통해 작성한 코드가 회로에 어떤 식으로 구현되는지 알 수 있다.

* Simulation 결과

스크린샷이(가) 표시된 사진

자동 생성된 설명

위에서 작성한 Verilog 코드를 시뮬레이션 돌린 결과이다. 시간에 따라 입력 clk, rst, in의 값을 바꾸면 그에 따라 출력 out의 값이 변화하는 것을 확인할 수 있다. 클럭의 상승 에지 마다 들어오는 입력이 순차적으로 10101이 되면 out이 1을 출력한다. 이때 Mealy machine으로 구현하였기에 sequence 10101이 입력으로 들어오는 동시에 1을 출력하는 것을 확인할 수 있다. 또한 overlapping 방식으로 구현되어 1010101이 들어오면 10101을 두 번 detect해 out이 10101에서 한 번, 1010101에서 한 번 총 두 번 1을 출력한다. 마지막으로 rst가 1이면, 즉 리셋 신호가 활성화 되어 있으면 10101이 들어오더라도 out은 0을 출력하는 것을 관찰할 수 있다.

* State Table

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

A = 초기 상태, B = 1, C = 10, D = 101, E = 1010

위는 Overlapping 방식으로 구현된 Sequence Detector 10101 Mealy machine의 상태표를 작성한 것이다. state A는 초기 상태를 나타내고 state B는 1이 detect 되었을 때, state C는 10이 detect 되었을 때, state D는 101이 detect 되었을 때, state E는 1010이 detect된 상태를 나타낸다. Mealy machine은 Moore machine과는 다르게, 출력이 입력 신호와 현재 상태에 종속되는 것을 확인할 수 있다. overlapping 방식으로 구현되었기 때문에 state E에서 input으로 1이 들어오면 101이 detect된 상태인 state D로 넘어가는 것을 확인할 수 있다. 또한 1010이 detect된 state E에서 input 값이 1이 들어오는 순간에 output이 1로 업데이트되어 출력이 delay되지 않는 것을 알 수 있다.

* State Diagram

도표, 그림, 폰트, 원이(가) 표시된 사진

자동 생성된 설명

**2) Sequence Detector 10101 Moore machine (Overlapping)**

* Verilog source code

|  |  |
| --- | --- |
| Design Source Code | Simulation Source Code |
| `timescale 1ns / 1ps  module seq\_moore(  input clk, rst, in,  output reg out  );  reg[4:0] seq;  reg[4:0] x = 5'b10101;  initial out = 1'b0;  always @(posedge clk) begin  if(rst) begin  seq <= 5'b00000;  out <= 1'b0;  end  else begin  seq <= seq << 1;  seq[0] <= in;  end  if(seq == x) out <= 1'b1;  else out <= 1'b0;  end  endmodule | `timescale 1ns / 1ps  module seq\_moore\_tb;  reg clk, rst, in;  wire out;  seq\_moore u\_seq\_moore (  .clk (clk ),  .rst (rst ),  .in (in ),  .out (out )  );  initial begin  clk = 1'b0;  rst = 1'b0;  in = 1'b0;  end  always #10 clk = ~clk;  always #200 rst = ~rst;  always@(in) begin  in = #10 ~in;  in = #10 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #10 ~in;  in = #40 ~in;  in = #30 ~in;  in = #10 ~in;  in = #10 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  in = #20 ~in;  end  initial begin  #1000  $finish;  end  endmodule |

위는 Verilog 코드로 Overlapping 방식의 Sequence Detector 10101 Moore machine을 구현한 것이다. seq\_moore모듈은 3개의 입력(clk, rst, in)과 1개의 출력(out)을 가지고 있다. 이때 clk는 클럭 신호, rst는 리셋 신호, in은 1 bit 입력, out은 1 bit 출력을 의미한다. 또한 5비트 레지스터 seq와 x를 정의한다. seq는 무어 머신에 들어오는 가장 최근 5 bit의 sequence를 저장하기 위한 용도이고 x는 detect하고 싶은 sequence로 초기값을 5'b10101로 설정했다. 클럭의 상승 에지에서 리셋 신호가 활성화되면(rst가 1이면) seq는 00000으로, out은 0으로 초기화한다. 그렇지 않으면 seq를 한 비트 왼쪽으로 시프트하고, 가장 오른쪽 비트를 in으로 설정한다. 그 후, seq가 x와 같으면 out을 1로, 그렇지 않으면 0으로 설정한다. 따라서 이 모듈은 입력 신호 in에서 주어진 시퀀스 x와 일치하는 패턴을 감지하고, 일치할 경우 출력 신호 out을 1로 설정하는 Overlapping 방식의 Sequence Detector로 동작할 수 있다. 또한 각 변수에 값 할당 시 nonblocking 방식(<=)을 사용해 Moore machine으로 동작할 수 있도록 구현했다.

* Schematic Diagram

도표, 라인, 텍스트, 평면도이(가) 표시된 사진

자동 생성된 설명

위에서 작성한 Verilog 코드로 Schematic Diagram을 생성한 결과이다. 이를 통해 작성한 코드가 회로에 어떤 식으로 구현되는지 알 수 있다.

* Simulation 결과

스크린샷이(가) 표시된 사진

자동 생성된 설명

위에서 작성한 Verilog 코드를 시뮬레이션 돌린 결과이다. 시간에 따라 입력 clk, rst, in의 값을 바꾸면 그에 따라 출력 out의 값이 변화하는 것을 확인할 수 있다. 클럭의 상승 에지 마다 들어오는 입력이 순차적으로 10101이 되면 out이 1을 출력한다. 이때 Moore machine으로 구현하였기에 sequence 10101이 입력으로 들어오고 한 클럭 뒤에 1을 출력하는 것을 확인할 수 있다. 또한 overlapping 방식으로 구현되어 1010101이 들어오면 10101을 두 번 detect해 out이 10101에서 한 번, 1010101에서 한 번 총 두 번 1을 출력한다. 마지막으로 rst가 1이면, 즉 리셋 신호가 활성화 되어 있으면 10101이 들어오더라도 out은 0을 출력하는 것을 관찰할 수 있다.

* State Table

|  |  |  |  |
| --- | --- | --- | --- |
| Present State | Next State | | Output |
| Input = 0 | Input = 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 |

A = 초기 상태, B = 1, C = 10, D = 101, E = 1010, F = 10101

위는 Overlapping 방식으로 구현된 Sequence Detector 10101 Moore machine의 상태표를 작성한 것이다. state A는 초기 상태를 나타내고 state B는 1이 detect 되었을 때, state C는 10이 detect 되었을 때, state D는 101이 detect 되었을 때, state E는 1010이 detect 되었을 때, state F는 10101이 detect된 상태를 나타낸다. Mealy machine과는 달리, Moore machine에서 출력은 현재 상태에만 종속되는 것을 확인할 수 있다. overlapping 방식으로 구현되었기 때문에 state F에서 input으로 0이 들어오면 1010이 detect된 상태인 state E로 넘어가고, input으로 1이 들어오면 1이 detect된 상태인 state B로 넘어가는 것을 확인할 수 있다. 또한 10101이 detect된 state F에서 input 값이 새로 들어올 때 output이 1로 업데이트되므로 출력이 delay되는 것을 알 수 있다.

* State Diagram

