8주차 결과보고서

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

**1.**

8주차 실험의 목적은 0에서 F까지 나타낼 수 있는 4-input 7-Segment Display를 FPGA에 직접 구현해보는 것이다. 실험의 세부 절차는 다음과 같다. 우선 7-Segment Display의 개념을 이해하고 verilog 언어를 사용해 7-Segment Display를 코드 상으로 구현한다. 다음으로 입력 신호를 생성한 후 Simulation을 통해 구현된 각 Gate가 잘 동작하는 지 확인한다. 다음으로 FPGA 보드와 연결해 Verilog로 구현된 회로의 동작을 확인한다.

**2.**

**7-Segment Display**

도표, 사각형, 흑백, 평면도이(가) 표시된 사진

자동 생성된 설명

* **Truth Table**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| In A | In B | In C | In D | Out A | Out B | Out C | Out D | Out E | Out F | Out G | Out Dp |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

위는 4개의 입력(In A-D)을 받아 8개의 출력(Out A-G, Dp)을 내보내는 7-Segment Display의 진리표를 작성한 것이다. 입력은 16진수 0에서 F 중 표현할 숫자를 나타내는 값이고 출력은 각 세그먼트에 불이 들어오는지 여부를 나타낸다. 이때 1은 불이 들어온 상태를, 0은 불이 꺼진 상태를 나타낸다.

* **K-map**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

위의 진리표를 바탕으로 카르노맵을 작성한 결과이다. minimum SOP를 구하는 과정을 통해 각 출력에 대한 Boolean function을 얻을 수 있었다.

* **Verilog 코딩**

|  |  |
| --- | --- |
| Design Source Code | Simulation Source Code |
| `timescale 1ns / 1ps  module seg(  input a, b, c, d,  output A, B, C, D, E, F, G, DP, W  );  assign A = (~b&~d)|(b&c)|(a&~d)|(~a&c)|(a&~b&~c)|(~a&b&d);  assign B = (~b&~d)|(~b&~c)|(~a&c&d)|(a&~c&d)|(~a&~c&~d);  assign C = (a&~b)|(~c&d)|(~a&b)|(~a&d)|(~b&~c);  assign D = (b&c&~d)|(a&b&~c)|(~b&~c&~d)|(b&~c&d)|(~a&~b&c)|(~b&c&d);  assign E = (~b&~d)|(c&~d)|(a&b)|(a&c);  assign F = (a&~b)|(b&~d)|(a&c)|(~c&~d)|(~a&b&~c);  assign G = (c&~d)|(a&~b)|(~b&c)|(a&d)|(~a&b&~c);  assign DP = 1;  assign W = A|B|C|D|E|F|G|DP;    endmodule | `timescale 1ns / 1ps  module seg\_tb;  reg aa, bb, cc, dd;  wire A, B, C, D, E, F, G, DP, W;  seg u\_seg (  .a (aa ),  .b (bb ),  .c (cc ),  .d (dd ),  .A (A ),  .B (B ),  .C (C ),  .D (D ),  .E (E ),  .F (F ),  .G (G ),  .DP (DP ),  .W (W )  );  initial begin  aa = 1'b0;  bb = 1'b0;  cc = 1'b0;  dd = 1'b0;  end  always aa = #80 ~aa;  always bb = #40 ~bb;  always cc = #20 ~cc;  always dd = #10 ~dd;  initial begin  #1000  $finish;  end  endmodule |

위에서 K-map을 통해 구한 Boolean function을 바탕으로 7-Segment Display의 Verilog 코드를 작성한 결과이다. seg 모듈은 4개의 입력(a, b, c, d)과 9개의 출력(A, B, C, D, E, F, G, DP, W)을 가지고 있다. 출력 A-G는 각 세그먼트, DP는 소수점을 나타낸다. W는 FPGA 보드 상에서 어느 digit에 나타낼지 지정하기 위한 용도로 쓰이는 추가적인 출력값이다. 출력 A-G에는 카르노맵을 통해 구한 boolean식을 할당해주었고, 실험 조건에서 DP는 항상 켜져 있을 것을 요구했기에 출력 DP에는 상수 1을 할당해주었다. 출력 W는 출력 A-G, DP를 OR 연산한 결과이다.

* **Schematic Diagram**

텍스트, 도표, 평면도, 라인이(가) 표시된 사진

자동 생성된 설명

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

* **Simulation 결과**

스크린샷, 컴퓨터, 키보드이(가) 표시된 사진

자동 생성된 설명

위에서 작성한 Verilog 코드를 시뮬레이션 돌린 결과이다. 시간에 따라 입력 a, b, c, d의 값을 바꾸면 그에 따라 출력 A~G, DP, W의 값이 변화하는 것을 보여준다. 출력 A-G는 각각 자신의 Boolean 식을 연산한 결과를 출력하고, 출력 DP는 언제나 1을 출력하며, 출력 W는 출력 A-G, DP 중 하나라도 1의 값을 가지면 1을 출력한다. 이를 통해 각 세그먼트와 연결된 출력 값들이 우리가 처음에 의도한 대로 잘 작동하는 것을 확인할 수 있었다.

**3.**

각 Segment A-G에 대해 K-map을 그려 올바른 Boolean function 얻은 덕분에 7-segment display가 의도한대로 잘 작동할 수 있었던 것 같다. 각 출력을 이전까지 사용해보지 않았던 Pin에 연결해보는게 흥미로웠고 출력 W를 DIGIT을 나타내는 Pin 중 하나에 연결해 해당 FPGA에서 원하는 위치에 불이 들어오게 하는 방법을 알게 되었다. 이전 실험들과 달리 Schematic Diagram이나 Simulation 결과를 보고 잘 동작하는지 확인하는 것보다 FPGA 보드와 연결해 잘 동작하는지 확인하는게 시각적으로 더 빠르고 편했다.

**4.**

이번 실험에서는 각 세그먼트의 Boolean function을 구해 7-Segment Display를 직접 동작시켰지만, 디코더를 7-Segment Display에 연결해 동작시키는 방식도 있다. 먼저, 사용하고자 하는 디코더를 선택한다. 일반적으로 BCD(이진 코드 십진 코드) 디코더를 사용하여 7-Segment Display를 제어한다. 예를 들어, 74LS47나 74LS48과 같은 BCD-7 Segment 디코더 칩을 선택할 수 있다. 다음으로 7-Segment Display와 디코더를 연결한다. 이때 각 세그먼트는 디코더의 출력 핀에 연결된다. 연결이 완료되면 이번에 디코더와 마이크로컨트롤러를 연결한다. 이때 마이크로컨트롤러의 출력 핀에 디코더의 입력 핀을 연결한다. 마이크로컨트롤러가 보내는 BCD 코드는 디코더를 통해 7-Segment Display에 숫자를 나타내는 신호로 변환된다. 예를 들어, 숫자 '0'은 BCD 코드로 '0000'에 해당된다. 디코더를 통해 이 코드를 변환시켜 7-Segment Display에 보내면 '0'이 표시된다.