**8주차 결과보고서**

**전공: 경영학과 학년: 3학년 학번: 20190963 이름: 한다현**

**1.**

**8주차 실험은 seven segment display의 개념을 이해하고 0부터 F까지 총 16개의 숫자와 알파벳을 seven segment display에 표시하는 것이다. Seven segment display는 4개의 입력과 7개의 출력을 갖는데, 이번 실험에서는 소수점까지 포함하여 총 8개의 출력을 구현하였다. 실험은 먼저 verilog로 디자인 코드를 작성하여 seven segment를 구현하고 이를 시뮬레이션을 통해 동작이 정상적으로 이루어지는지 확인한다. 이후에 각각의 입력과 출력을 FPGA의 스위치와 seven segment display와 연결한 후 입력의 변화에 따라 0부터 F가 모두 seven segment display에서 정상적으로 나타나는지 확인한다. 이러한 과정을 통해 seven segment display의 개념과 구현하는 방법을 공부한다.**

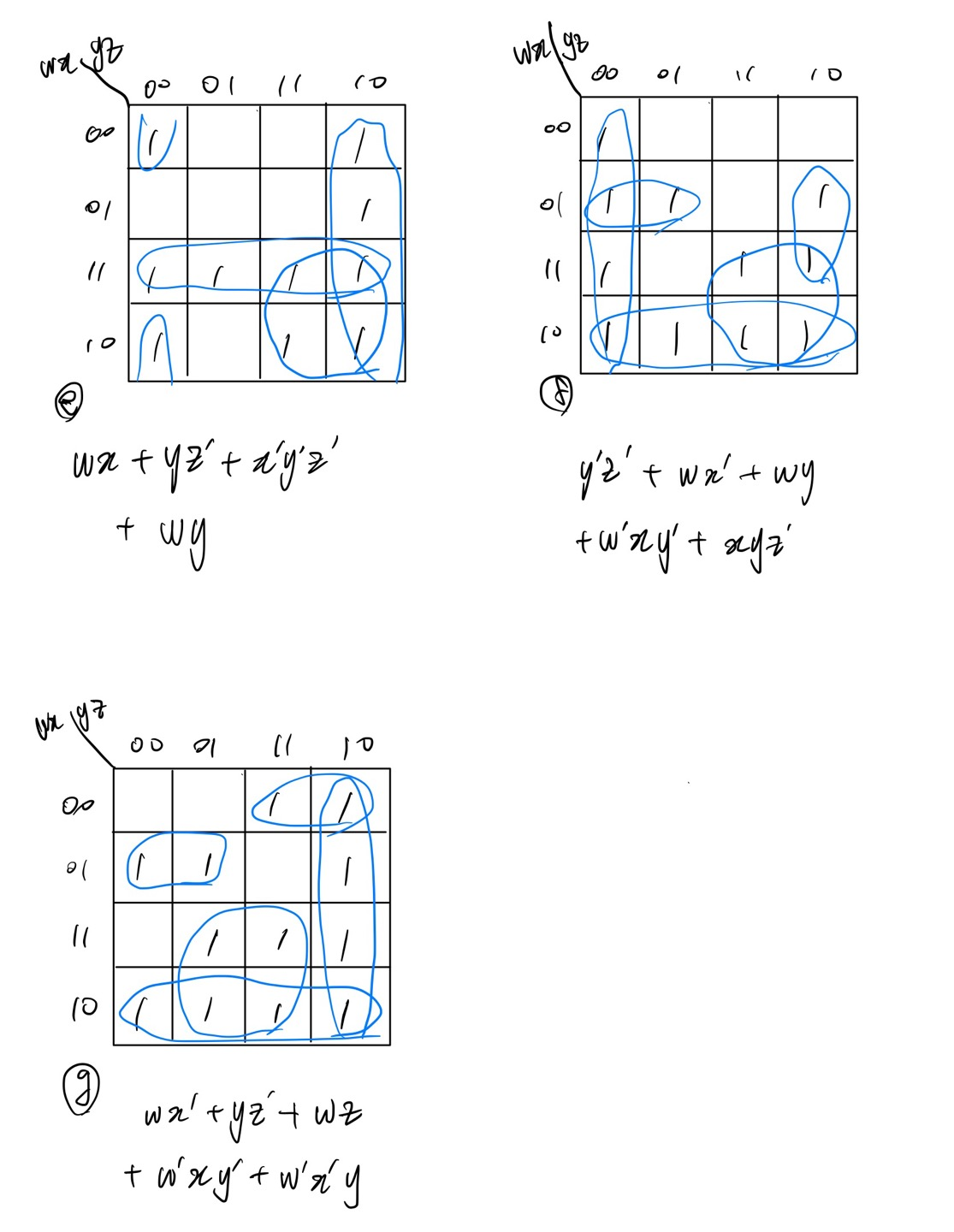
**2.**

**Seven segment display는 4개의 입력 w, x, y, z와 8개의 출력 a, b, c, d, e, f, g, dp로 구성되어 있다. 8개의 출력이 독립적으로 출력되야 하기 때문에 총 8개의 카르노 맵이 필요하다. 하지만 이 때 dp는 항상 1을 출력하도록 설정하기 때문에 dp를 제외한 7개의 출력에 대해 카르노 맵을 작성한다. 카르노 맵을 작성하기 위해 진리표를 작성하면 다음과 같다.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **In w** | **In x** | **In y** | **In z** | **Out a** | **Out b** | **Out c** | **Out d** | **Out e** | **Out f** | **Out g** | **dp** |
| **0** | **0** | **0** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **1** |
| **1** | **0** | **0** | **0** | **1** | **0** | **1** | **1** | **0** | **0** | **0** | **0** | **1** |
| **2** | **0** | **0** | **1** | **0** | **1** | **1** | **0** | **1** | **1** | **0** | **1** | **1** |
| **3** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **1** | **1** |
| **4** | **0** | **1** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **1** |
| **5** | **0** | **1** | **0** | **1** | **1** | **0** | **1** | **1** | **0** | **1** | **1** | **1** |
| **6** | **0** | **1** | **1** | **0** | **1** | **0** | **1** | **1** | **1** | **1** | **1** | **1** |
| **7** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **1** |
| **8** | **1** | **0** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** |
| **9** | **1** | **0** | **0** | **1** | **1** | **1** | **1** | **0** | **0** | **1** | **1** | **1** |
| **a** | **1** | **0** | **1** | **0** | **1** | **1** | **1** | **0** | **1** | **1** | **1** | **1** |
| **b** | **1** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** |
| **c** | **1** | **1** | **0** | **0** | **1** | **0** | **0** | **1** | **1** | **1** | **0** | **1** |
| **d** | **1** | **1** | **0** | **1** | **0** | **1** | **1** | **1** | **1** | **0** | **1** | **1** |
| **e** | **1** | **1** | **1** | **0** | **1** | **0** | **0** | **1** | **1** | **1** | **1** | **1** |
| **f** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | **1** | **1** | **1** | **1** |

**이를 바탕으로 a부터 g까지 7개의 출력에 대한 카르노 맵을 작성하면 다음과 같다.**

텍스트, 낱말맞추기게임이(가) 표시된 사진

자동 생성된 설명

**위 카르노 맵을 바탕으로 Verilog의 디자인 코드를 작성하면 각각의 출력은 다음과 같이 설계된다.**

**assign a = ~w&y | x&y | y&~z | ~w&x&z | w&~y&~z | w&~x&~y | ~x&~y&~z;**

**assign b = ~w&~x | ~w&~y&~z | ~w&y&z | w&~y&z | w&~x&~z;**

**assign c = ~y&z | ~w&x | w&~x | ~w&~y| ~w&z;**

**assign d = ~w&~x&~z | w&~y&~z | x&~y&z | ~x&y&z | x&y&~z;**

**assign e = w&x | y&~z | ~x&~y&~z | w&y;**

**assign f = ~y&~z | w&~x| w&y| ~w&x&~y| x&y&~z;**

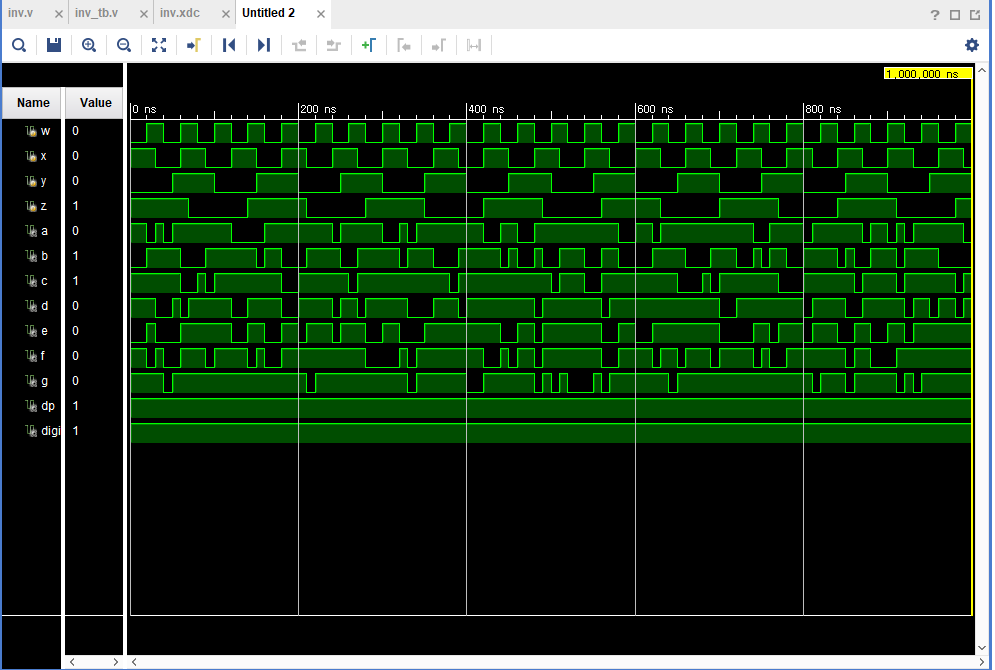
**assign g = w&~x | y&~z | w&z | ~w&x&~y| ~w&~x&y;**

**assign dp = 1;**

**디자인 코드를 설계할 때 위의 8개의 출력 외에 하나의 변수가 더 필요하다. FPGA에는 8개의 seven segment display가 있다. 만약 위와 같은 디자인 코드를 실행하게 되면 8개의 디스플레이 모두에 결과가 출력되지만 이번 실험에는 하나의 seven segment display만 필요하기 때문에 digit이라는 변수를 추가하여 하나의 디스플레이에만 우리가 원하는 결과를 출력하도록 설계한다. 따라서 디자인 코드에 다음과 같은 코드를 추가한다.**

**assign digit = a|b|c|d|e|f|g|dp;**

**이제 4개의 입력 w, x, y, z가 일정한 간격에 따라 변화하도록 설정한 후 시뮬레이션을 실행한 결과는 다음과 같다.**



**시뮬레이션 결과 w, x, y, z가 각각 변화함에 따라 a, b, c, d, e, f, g가 그에 맞게 변화하는 것을 볼 수 있고, 소수점을 나타내는 dp는 항상 1이 되도록 설계했기 때문에 w, x, y, z의 변화와 상관없이 1을 출력하고 있다. a~g의 출력을 조합해보면 0부터 F까지 정상적으로 출력될 것이라는 것을 예상할 수 있다.**

**3.**

**시뮬레이션 결과에서 볼 수 있듯이 dp를 제외한 나머지 7개의 아웃풋은 개별적으로 출력된다. 예를 들어, 앞서 작성했던 진리표를 바탕으로, 입력 w, x, y, z가 각각 0, 0, 1, 0이면 seven segment display에 2가 출력되어야 하기 때문에 seven segment display에서 2를 구성하는 아웃풋인 a, b, d, e, g의 값이 1이 되고 c와 f의 값은 0이 되어 seven segment display에서 2의 모양이 출력된다. 예를 하나 더 살펴보면, w, x, y, z각 각각 1, 1, 1, 0이면 seven segment display에서 e가 출력되어야 한다. 따라서 seven segment display 상에서 e를 구성하는 아웃풋인 a, d, e, f, g의 값이 1이 되고 b와 c의 값은 0이 되기 때문에 디스플레이에서 e가 출력될 수 있다. 또한 이번 실험에는 digit이라는 변수를 사용하여 8개의 seven segment display 중 하나의 디스플레이만 사용할 수 있도록 설계했는데, digit 외에 다른 변수를 추가로 할당하면 추가된 변수의 개수에 따라 더 많은 수의 디스플레이를 사용하는 것이 가능할 것이고, 더 나아가 a~f까지의 출력 외에 더 많은 출력을 추가하면 각각의 디스플레이마다 다른 결과를 출력하는 것이 가능할 것이라고 생각한다.**

**4.**

**Seven segment display를 구현하는 방법은 우리가 실험에서 했던 것처럼 a~g를 카르노맵을 사용해 각각 구현하는 방법과 TTL 7447과 같은 디코더를 사용하여 디코더를 Seven segment display에 연결하는 방법이 있다. Seven segment를 구현하는 디코더의 종류로는 애노드 공통형인 74LS46, 74LS246와 캐소드 공통형인 74LS48, 74LS248 등이 있다. 8주차 예비보고서에서 조사했던 바와 같이 애노드 공통형은 입력으로 LOW 신호를 받고 캐소드 공통형은 입력으로 HIGH 신호를 받는다. 이를 각각 active-low, active-high라고도 한다. 74LS46과 74LS246의 차이점은 74LS46의 경우 6을 표현할 때 c, d, e, f, g가 활성화되고, 9를 표현할 때는 a, b, c, f, g가 활성화된다. 반면, 74LS246의 경우에는 6을 표현할 때는 a, c, d, e, f, g가 활성화되고, 9를 표현할 때는 a, b, c, d, f, g가 활성화되어 seven segment display에서 표시되는 6과 9의 형태의 차이가 있다. 이러한 차이는 74LS48과 74LS248에서도 나타난다.**

**참조: 디지털공학(조현철)**