전공: 컴퓨터공학 학년: 2학년 학번: 20161603 이름: 신민준

1. 실험 목적

8주차 실험에서는 7-segment display의 개념을 이해하고, Verilog를 사용해 4개의 input을 가지는 7-Segment Display를 구현한 후, 이를 FPGA 보드 상에서 실제로 동작을 확인해보았다.

1. 7-Segment Display의 결과 및 Simulation 과정에 대해서 설명하시오(Truth table 작성 및 K-map 포함, 0~F, DP)

이 실습에서 구현한 7-segment display는 4x7 BCD Decoder가 내장된 형태로 구현해야 했기 때문에, 이 Decoder에 대해 다음과 같은 Truth table을 작성했다.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **In A** | **In B** | **In C** | **In D** | **Out A** | **Out B** | **Out C** | **Out D** | **Out E** | **Out F** | **Out G** | **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 |

이 때, 실험의 조건에서 DP(Decimal Point)에 해당하는 값은 언제나 1로 고정해 놓아야 했기 때문에 위 표와 같이 모든 경우의 수에 대해 1의 값을 가지도록 설계했다. 이 부분은 Verilog code를 작성할 때 상수 값 1을 output으로 넘겨주는 방법으로 구현했다.

위 진리표를 기반으로 7개의 각 segment들에 대해 Karnaugh map을 작성했다. 이 K-map들과 그로부터 도출한 Boolean function의 식은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ab cd | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 1 | 1 |
| 01 | 0 | 1 | 1 | 1 |
| 11 | 1 | 0 | 1 | 1 |
| 10 | 1 | 1 | 0 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ab cd | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | 1 | 1 |
| 01 | 1 | 0 | 1 | 0 |
| 11 | 0 | 1 | 0 | 0 |
| 10 | 1 | 1 | 0 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ab cd | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | 1 | 0 |
| 01 | 1 | 1 | 1 | 1 |
| 11 | 0 | 1 | 0 | 0 |
| 10 | 1 | 1 | 1 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ab cd | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 1 | 1 |
| 01 | 0 | 1 | 0 | 1 |
| 11 | 1 | 1 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ab cd | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 0 | 1 |
| 01 | 0 | 0 | 0 | 1 |
| 11 | 1 | 1 | 1 | 1 |
| 10 | 1 | 0 | 1 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ab cd | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 0 | 0 |
| 01 | 1 | 1 | 0 | 1 |
| 11 | 1 | 0 | 1 | 1 |
| 10 | 1 | 1 | 1 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ab cd | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 1 | 1 |
| 01 | 1 | 1 | 0 | 1 |
| 11 | 0 | 1 | 1 | 1 |
| 10 | 1 | 1 | 1 | 1 |

7-segment display에 필요한 값 을 구했으니, 이 Boolean식을 그대로 Verilog code로 작성하면 구현이 완성된다.

추가적으로, FPGA에서 원하는 위치에 해당하는 7-segment display에 출력을 하기 위해선 해당 DIGIT 번호에 대응하는 FPGA PIN에 high값을 넣어야 한다. 이런 조건들을 고려해 1번 7-segment display에 결과값을 출력하고, DP도 추가로 켜 주는 Verilog code를 다음과 같이 작성했다.

|  |
| --- |
| `timescale 1ns / 1ps  module segment(      input a, b, c, d,      output ao, bo, co, do, eo, fo, go, dp, digit      );      assign ao = (~b&~d)|(a&~d)|(a&~b&~c)|(~a&b&d)|(~a&c)|(b&c);      assign bo = (~b&~c)|(~b&~d)|(~a&~c&~d)|(~a&c&d)|(a&~c&d);      assign co = (~a&b)|(a&~b)|(~c&d)|(~a&~c)|(~a&d);      assign do = (~b&~c&~d)|(~b&c&d)|(~a&c&~d)|(b&~c&d)|(a&b&~d);      assign eo = (~b&~d)|(c&~d)|(a&b)|(a&c);      assign fo = (~c&~d)|(b&~d)|(~a&b&~c)|(a&c)|(a&~b);      assign go = (~a&~b&c)|(c&~d)|(~a&b&~c)|(a&d)|(a&~b);      assign dp = 1;      assign digit = 1;  endmodule |

실험 조건에서 DP는 언제나 켜져 있을 것을 요구했기에 dp변수에 상수 1을 할당했고, 1번째 7-segment display에서 출력을 하기 원했기에 다음과 같은 constraint source를 만들어 적용했다.

|  |
| --- |
| set\_property PACKAGE\_PIN J4 [get\_ports a]  set\_property PACKAGE\_PIN L3 [get\_ports b]  set\_property PACKAGE\_PIN K3 [get\_ports c]  set\_property PACKAGE\_PIN M2 [get\_ports d]  set\_property PACKAGE\_PIN D20 [get\_ports ao]  set\_property PACKAGE\_PIN C20 [get\_ports bo]  set\_property PACKAGE\_PIN C22 [get\_ports co]  set\_property PACKAGE\_PIN B22 [get\_ports do]  set\_property PACKAGE\_PIN B21 [get\_ports eo]  set\_property PACKAGE\_PIN A21 [get\_ports fo]  set\_property PACKAGE\_PIN E22 [get\_ports go]  set\_property PACKAGE\_PIN D22 [get\_ports dp]  set\_property PACKAGE\_PIN E14 [get\_ports digit] |

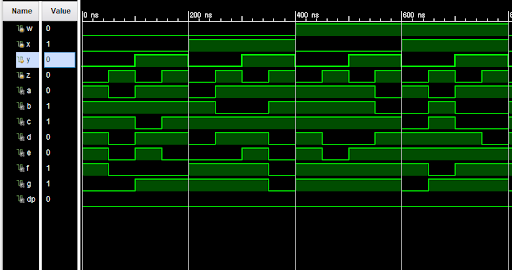
Digit 변수가 DIGIT 1에 해당하는 E14 pin에 연결되어있는 것을 확인할 수 있다.

RTL Schematic의 결과 나온 위 Verilog code의 Schematic은 다음과 같다.

텍스트, 지도이(가) 표시된 사진

자동 생성된 설명

위 Verilog code를 검증하기 위해 다음과 같이 Simulation을 수행했고 그 결과는 다음에서 보이는 것과 같다.



각 a, b, c, d, e, f, g, dp의 각 요소가 의도했던, 아래 사진에서 보이는 segment들의 출력과 같음을 알 수 있다.



1. 결과 검토 및 논의 사항

다행히도, 정확한 Truth table과 각 Segment A~G에 대해 올바른 K-map을 그려 Boolean function 식을 도출한 덕분에 결과적으로 7-segment display에서 의도했던 출력 값이 나왔음을 확인할 수 있었다. 추가적으로, 해당 FPGA에서 원하는 7-segment display에 출력을 하는 방법을 확인할 수 있었다.

1. 추가 이론 조사 및 작성

* 이번 실험에서 제작한 코드의 RTL Simulation은 매우 복잡하고 비용이 많이 드는 설계였다. 만약 각 Segment를 표현하는 Boolean 식들이 서로 공유하는 term이 최대가 되도록 식들을 정리한다면, 비용을 크게 줄일 수 있을 것이다.
* FPGA에서 원하는 7-segment display의 led에 값을 전달하기 위해선 DIGIT 1 ~ DIGIT 8에 해당하는 pin에 Verilog code에서의 모듈 output을 연결해야 하는데, 이 때 한번에 신호를 줄 수 있는 DIGIT 핀은 단 하나이다. 따라서, n개의 숫자를 동시에 보이기 위해선 Dynamic 동작 방식을 채용해 빠른 시간동안 원하는 자릿수의 display들의 DIGIT pin에 신호를 넣고 빼고를 반복해야 할 것이다.