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

1. 실험 목적

이 실험은 Verilog를 사용해 작성한 코드를 직접 FPGA에 업로드해 실제 FPGA의 각 기기들을 작동해 코드에서 의도한 결과가 나오는지 확인하는 실험이다. 이 때, 사용한 코드는 3-input/4-input 으로 각각 프로그래밍 한 AND, OR gate이며, 이 각 4가지 코드는 또 simultaneous 한 gate와 그렇지 않고 순차적으로 실행하는 gate로 나눠 총 8가지의 코드를 FPGA에 적용해 실험했다.

1. FPGA 동작법을 설명하시오

FPGA를 동작하는 방법은 다음과 같다.

1. Verilog coding

Verilog를 사용하여 input에 대한 원하는 output, 결과를 도출해내는 Verilog code를 작성한다. 이를 Verilog coding이라 부른다.

1. 합성 단계(Run synthesis)

원하는 코드를 작성한 이후, Run Synthesis 메뉴를 눌러 합성을 진행한다.

1. Device/Pin Assignment

FPGA가 컴퓨터와 연결되어있는 것을 확인한 후, Verilog coding 단계에서 만든 코드들의 input과 output을 실제 FPGA의 핀과 매칭시킨다. 이는 Window 메뉴의 I/O Ports 탭에서 확인 가능하다.

핀 내용이 전부 설정되었으면, Constraint file을 생성하여 설정한 모든 내용을 저장하도록 한다.

1. Synthesis / Implement

Run Synthesis 메뉴를 통해 합성을 진행하고, 이후 Run Implementation으로 과정을 진행한다. Design Runs 탭에서 모든 Implementation이 완료된 것을 확인했으면, Generate Bitstream 메뉴를 선택해 비트 스크림을 생성한다.

1. Device configuration

이후, Open Hardware Manager > Open Target > Auto Connect, Program Device > xc7a75t\_0를 선택해 생성한 비트스트림을 실제 FPGA에 프로그래밍하고, 지정한 input 값을 변경해보며 Verilog code를 코딩할 때의 의도가 충분히 FPGA의 output에서 반영되었는지 확인한다.

1. 3-input AND gate의 simulation 결과 및 과정에 대해 설명하시오 (3 input, 2 output)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In C | Out D | Out E |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

Simulation 결과, A와 B 두 스위키가 켜져있을 때는 하나의 불이 들어왔고, A, B, C 세 스위치가 모두 켜져있을 때에는 두 불이 모두 들어왔지만, 나머지의 경우에는 LED에 불이 들어오지 않았다. 3-input AND gate로 생각했을 때, A, B, C 모두 다 신호가 들어와야만 E에 신호가 가므로 올바르게 구현한 것이라 생각할 수 있다.

1. 4-input AND gate의 simulation 결과 및 과정에 대해 설명하시오 (4 input, 3 output)

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

4-input AND gate 또한 3-input AND gate와 비슷한 양상을 띄며, 4개의 input, A, B, C, D가 모두 다 1의 값을 가져야만 출력값인 G에 LED 불이 켜졌다.

1. 3-input OR gate의 simulation 결과 및 과정에 대해 설명하시오 (3 input, 2 output)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In C | Out D | Out E |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

3-input OR gate의 경우, AND gate와는 반대로, 모든 3개의 input이 0의 값을 가질 때 에만 최종 output인 E에 불이 들어오지 않았고, 이 외의 모든 경우에서 E의 LED가 켜지는 것을 확인할 수 있었다. 따라서 이 3-input OR gate는 정확하게 구현한 것이라 확인할 수 있다.

1. 4-input OR gate의 simulation 결과 및 과정에 대해 설명하시오 (4 input, 3 output)

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

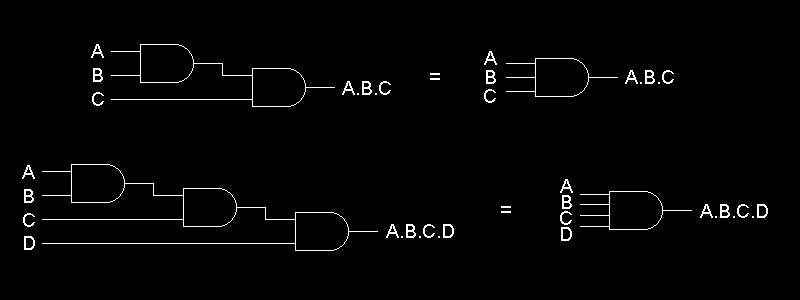
3-input OR gate의 진리표와 비슷한 양상을 보이고 있다. 네 input이 모두 0의 값을 가지고 있을 때에만 최종 output인 G가 꺼지고, 그 외의 모든 case들에 대해서는 항상 G가 켜져있는 모습을 확인할 수 있다.

1. 결과 검토 및 논의사항

지금껏 Verilog Code를 작성하면서 실제로 기계로 이 코드가 구현되었을 때 어떻게 작동할지에 대해서는 쉽게 감이 오지 않았지만, 이번 실험에서 AND gate와 OR gate가 정상적으로 제작됬다는 것을 확인할 수 있었다. AND gate는 모든 input이 1의 값을 가지지 않는 이상 언제나 output이 0이고, OR gate는 모든 input이 0을 가질 때에만 output이 0인 특징을 가지고 있는데, 이 특징은 실험 결과를 정리한 표들에서 확연하게 드러난다. 따라서 정확하게 만들었다 볼 수 있다.

다만 이러한 회로의 경우, input이 많지 않아 진리표를 작성하고 원하는 결과가 나왔음은 검증이 가능하지만, 복잡한 회로에서 수많은 input을 가지고 있는 경우 값에 대한 검증은 이 방식대로는 어려울 수 밖에 없다. 이럴 때 사용하는 것이 2주차 때 사용한 Simulation이라고 볼 수 있다. 자신이 원하는 동작을 미리 따로 명시해두면, 값에 대한 검증이 더 빠르고 쉽게 이루어질 수 있을 것이다.

1. 추가 이론 조사 및 작성



Multi-input gate들은 이번 실험에서 만든 것 처럼 여러 개의 gate들을 체인처럼 연결시켜 구현한다. 이렇게 만든 gate들은 위 그림에서 오른쪽에 보인 것처럼 그림에 나타낼 수 있다.