4주차 예비보고서

전공: 컴퓨터공학과 학년: 3학년 학번: 20211547 이름: 신지원

**1.**

FPGA 는 크게 5가지의 프로세스를 거쳐 진행된다. Design Entry – RTL simulation – Synthesis – P&R – Timing Simulation 으로 나누어진다. (실제 실습하였던 것보다 더 포괄적으로 작성하였다.)

Design Entry 과정에서는 HDL을 가지고 설계하거나 Schematic 을 가지고 설계할 수 있다. 전자는 코드를 작성하여, 후자는 그림을 그려 설계한다고 이해할 수 있다. 우리 실습 시간에는 Verilog HDL 을 사용하여 Design Entry 과정을 수행하고 있다.

RTL simulation 과정에서는 Design Entry 에서 구현한 코드(혹은 그림)을 시뮬레이션으로 검증한다. 시뮬레이션으로만 검증하는 것이기 때문에 시뮬레이션 결과값은 이상이 없더라도 실제 FPGA 에 올리게 되면 이상이 있을 수 있기 때문에 참고값으로 생각해야 한다.

Synthesis 는 합성 과정으로 구현한 High-Level 디자인을 Low-Level 바꾸어준다. 구현한 코드가 High-Level 이기 때문에 FPGA 보드가 인식하기 위해선 Low-Level 로 바꾸어주는 작업이 꼭 필요하다. 이 과정을 거치게 되면 output 파일인 netlist 가 나오게 된다. 또한 본인이 실제로 실습하였을 때는 이 과정을 거친 뒤에 스케마 파일을 볼 수 있었다.

Place and Route 과정에서는 Synthesis 후에 나온 셀을 배치히고 연결한다. P&R 엔진이 알아서 가장 효율적인 배치를 찾아 연결한다. 그 후에 이러한 구성 데이터는 비트스트림 생성기라는 프로그램에 의해 특수 파일에 기록됩니다.

마지막으로 timing simulation 과정을 거치는데, 각 셀들을 배치하고 연결하였다면 처음보다 wire 의 길이가 달라질 수 있다. 이는 딜레이 문제로 이어질 수 있기 때문에 시뮬레이션을 통하여 이를 고려한다.

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

**2.**

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

3input-2output AND gate의 inv와 inv\_tb

inv 파일에서 a,b,c 를 input 으로, d,e 를 output 으로 선언하였다. 따라서 d는 a와 b의 곱으로, e는 d 와 c의 곱으로 나타내었다. 테스트벤치 파일에서는 초기화와 시간을 정해주었다. 모든 case 를 출력하기 위하여 input 의 주기에 텀을 두고 설정하였다.

스크린샷, 텍스트, 도표, 디스플레이이(가) 표시된 사진

자동 생성된 설명

3input-2output AND gate의 simulation

simulation 을 통하여 볼 수 있듯이, d는 aa와 bb가 모두 1일 때만 1로 출력이 된다. 그 이유는 위 inv 파일에서 assign d = a&b 로 선언하였기 때문이다. 마찬가지로 e 는 d 와 cc 가 모두 1일 때만 1로 출력되는데, 이는 aa, bb, cc 가 모두 1일 때 e 도 1로 출력됨을 의미한다.

추가적으로 테스트벤치에서 always 문 내부에서 aa, bb, cc의 주기를 blocking 으로 선언하였기 때문에, aa가 #20에 시작하고 시작한지 #30이 되었을 때 bb가 시작하게 된다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **input a** | **input b** | **input c** | **output d** | **output 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 |

3input-2output AND gate의 진리표

Simulation 의 결과값을 바탕으로 작성한 진리표는 위와 같다. 입력 값이 모두 1일 때 출력 값은 1이 된다.

**3.**

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

4input-3output AND gate의 inv와 inv\_tb

inv 파일에서 a, b, c, d 를 input 으로, e, f, g 를 output 으로 선언하였다. 따라서 e는 a와 b의 곱으로, f는 e 와 c의 곱으로, g는 f와 d의 곱으로 나타내었다. 테스트벤치 파일에서는 초기화와 시간을 정해주었다. 모든 case 를 출력하기 위하여 input 의 주기에 텀을 두고 설정하였다.

스크린샷, 디스플레이, 멀티미디어 소프트웨어, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

4input-3output AND gate의 simulation

simulation 을 통하여 볼 수 있듯이, e는 aa와 bb가 모두 1일 때만 1로 출력이 된다. 그 이유는 위 inv 파일에서 assign e = a&b 로 선언하였기 때문이다. 마찬가지로 f 는 e 와 cc 가 모두 1일 때만 1로 출력되고, g는 aa, bb, cc, dd 가 모두 1일 때만 1로 출력됨을 의미한다.

따라서 e,f,g 가 1에서 0으로 출력이 바뀌는 구간은 지연을 무시한다면 가장 빨리 끝나는 aa의 입력이 0 이 될 때와 모두 같을 것이다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **input a** | **input b** | **input c** | **input d** | **output e** | **output f** | **output 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 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

4input-3output AND gate의 진리표

Simulation 의 결과값을 바탕으로 작성한 진리표는 위와 같다. 입력 값이 모두 1일 때 출력 값은 1이 된다.

**4.**

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

3input-2output OR gate의 inv와 inv\_tb

inv 파일에서 a,b,c 를 input 으로, d,e 를 output 으로 선언하였다. 따라서 d는 a와 b의 합으로, e는 d 와 c의 합으로 나타내었다. 테스트벤치 파일에서는 초기화와 시간을 정해주었다. 모든 case 를 출력하기 위하여 input 의 주기에 텀을 두고 설정하였다.

스크린샷, 소프트웨어, 텍스트, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

3input-2output OR gate의 simulation

simulation 을 통하여 볼 수 있듯이, d는 aa와 bb 중 하나라도 1일 때 1로 출력이 된다. 그 이유는 위 inv 파일에서 assign d = a|b 로 선언하였기 때문이다. 마찬가지로 e 는 d 나 cc 가 하나라도 1일 때 1로 출력되는데, 이는 aa, bb, cc 중에 하나라도 입력값이 1일 때 e 도 1로 출력됨을 의미한다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **input a** | **input b** | **input c** | **output d** | **output 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 |

3input-2output OR gate의 진리표

Simulation 의 결과값을 바탕으로 작성한 진리표는 위와 같다. 입력 값 중 하나라도 1이 있을 때 출력 값은 1이 된다.

**5.**

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

4input-3output OR gate의 inv와 inv\_tb

inv 파일에서 a,b,c,d 를 input 으로, e,f,g 를 output 으로 선언하였다. 따라서 e는 a와 b의 합으로, f는 e 와 c의 합으로, g는 f와 d의 합으로 나타내었다. 테스트벤치 파일에서는 초기화와 시간을 정해주었다. 모든 case 를 출력하기 위하여 input 의 주기에 텀을 두고 설정하였다.

멀티미디어 소프트웨어, 소프트웨어, 그래픽 소프트웨어, 편집이(가) 표시된 사진

자동 생성된 설명

4input-3output OR gate의 simulation

simulation 을 통하여 볼 수 있듯이, e는 aa와 bb 중 하나라도 1일 때 1로 출력이 된다. 그 이유는 위 inv 파일에서 assign e = a|b 로 선언하였기 때문이다. 마찬가지로 f 는 e 와 cc 중 하나라도 1일 때 1로 출력되고, g는 aa, bb, cc, dd 중 하나라도 1일이 입력될 때 1로 출력됨을 의미한다.

따라서 e,f,g 가 0에서 1로 출력되는 구간은 지연을 무시한다면 시작이 가장 빠른 aa의 입력이 1이 될 때와 모두 똑같을 것이다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **input a** | **input b** | **input c** | **input d** | **output e** | **output f** | **output 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 |

4input-3output OR gate의 진리표

Simulation 의 결과값을 바탕으로 작성한 진리표는 위와 같다. 입력 값 중 하나라도 1이 있을 때 출력 값은 1이 된다.

**6.**

실습에서 Simulation 을 통하여 작성한 inv, inv\_tb 가 어떻게 실행될지 시뮬레이션 값을 알 수 있었다. AND gate는 OR gate 보다 만족하는 경우가 적기 때문에 시뮬레이션 상에서도 출력 값 1인 경우가 더 좁은 영역으로 나타났다. AND gate를 구현할 때 입력 값 중 a 가 가장 먼저 1에서 0으로 입력값을 바꾸도록 하였다. AND gate 의 출력 값은 입력 값 모두가 1이 되어야 1을 출력한다. 따라서 2output, 3ouput 모두 a가 1에서 0으로 입력이 바뀌는 구간이 출력 값들이 모두 1에서 0으로 바뀌는 구간이 되었다. OR gate 도 마찬가지로 입력 값 중 a 가 가장 먼저 0에서 1으로 입력 값이 바뀌도록 하였기 때문에 2output, 3ouput 모두 a가 0에서 1로 입력이 바뀌는 구간이 출력 값들이 모두 0에서 1으로 바뀌는 구간이 되었다.

**7.**

3주차에서 논리합과 논리곱을 이용하여 input, output 의 개수를 달리하여 simulation 을 살펴보는 실습을 진행하였다. 따라서 논리합, 논리곱 뿐만 아니라 이를 이용하여 도출한 논리 연산에 필요한 법칙을 조사하였다. Verilog 또한 논리 연산에서 교환 법칙, 결합 법칙, 분배 법칙이 사용될 수 있다는 점을 알게 되었다.

텍스트, 폰트, 스크린샷, 화이트이(가) 표시된 사진

자동 생성된 설명