컴퓨터 공학 기초 실험2 보고서

실험제목: 2-to-1 MUX

실험일자: 2023년 09월 12일(일)

제출일자: 2023년 09월 13일(월)

학 과: 컴퓨터정보공학부

담당교수: 이준환 교수님

실습분반: 금요일 0, 1, 2

학 번: 2019202076

성 명: 정수필

1. 제목 및 목적

A. 제목

2-to-1 MUX ASSIGNMENT

B. 목적

S라는 signal의 값에 의해 output이 달라지는 Mux 또는 (멀티플렉서) 라고 불리는 gate 중 2개의 input을 가지는 MUX를 Verilog 언어로 프로그래밍이 가능한 쿼터스 프라임2 환경에서 이를 직접 설계하여 Mux의 동작과 회로들의 구성에 대해서 이해를 높인다.

2. 원리(배경지식)

이번 과제를 수행하기 위해 습득한 지식들은 하드웨어 프로그래밍 도구 쿼터스 프라임, verilog grammar 핵심적인 문법, digital logic circuit 과목에서 배운 카르노맵, MUX의 TRUTH TABLE, 등 이론 지식 활용, 쿼터스 프라임 개발도구의 사용법과 같이 위의 지식들을 먼저 확립하고 과제를 수행하였다.

-About Quartus Prime 2 tool

우리가 보통 소프트웨어 프로그램을 만들기 위해 여러 통합 환경에서 코딩을 하는데, 베릴로그 라는 하드웨어 프로그래밍 언어를 쿼터스 프라임 2 통합환경에서 개발한다는 것에 대해서 알게 되었다. 그동안 주로 사용해봤던 소프트웨어 도구는 vscode, vsstudio, eclipse, Intellij 을 사용해보았는데, 하드웨어 설계 프로그래밍 tool은 쿼터스 프라임2, 인텔에서 개발한 소프트웨어 도구를 이용하여 설계할 수 있다.

-How to use tool (Quartus Prime 2 tool)

처음 접하여 사용하게 된 쿼터스 프라임 2 도구 활용법에 대해서 강의 자료에서 나왔던 RTL Viewer을 이용하여 내가 설계했던 모듈을 그림으로 볼 수 있다. 이때 CPU 마다 컴파일러가 다르게 그림을 그려줄 수 있으므로 그림의 모양이 다를 수도 있다.

Ctrl+N 키를 이용하여 Verilog 문법을 작성할 수 있는 스크립트 파일을 생성할 수 있다. 설계를 하고 테스트벤치 코드를 작성하여 내가 설계한 module이 잘 동작하는지 waveform, 파동의 형태를 보면서 검증할 수 있다. 이때 시간이 조금 걸린다.

디지털 논리회로 1 수업때 배웠던 and, or, not 등의 논리 연산 gate를 따로 module 로 만들어서 다시 이를 재사용 할 수 있다. 즉 c++, java 와 같은 객체지향 프로그래밍 언어에서 사용해보았던 클래스와 객체의 개념과 유사하다.

-About Verilog grammar

Module 안 또 다른 Module을 구성하여 설계할 수 있다.
Endmodule 이라는 keyword를 이용하여 module 구성을 마칠 수 있으며
Module 을 각각 다른 파일로 저장 후 이를 사용하고자 하는 모듈명 사용할 모듈이름 (각각의 input, output mapping); 이라는 문법 구조를 이용하여 모듈 객체를 만들 수 있다.

이때 매개변수 mapping의 경우 두가지의 방법이 있다. 하나는 name mapping으로 .(dot)을 이용하는 법, orderd mapping 즉, 모듈의 input 순서에 따른 방식으로 연결 할 수 있다.

input -> 입력값을 지정해주는 키워드
output-> 결과값을 지정해주는 키워드
assign -> 값을 할당해주는 키워드

Initial begin end -> initial begin을 이용하여 테스트 케이스를 그 안에 넣고 end로 이를 종료한다.
테스트벤치 파일 작성시, 하드웨어 설계에서 모듈 설계는 중요한데, 이 또한 테스트 벤치 소스 코드 작성시에도 module 키워드를 이용하여 작성후 endmodule을 사용하여 끝내줘야한다.

Reg -> register라는 뜻이 아닌, 테스트 벤치 코드 작성시 사용할 수 있는 일종의 바구니와 같은 저장소이다.
Wire-> 서로 다른 module의 input과 output을 연결하기 위한 전선과 같은 개념이다. 이는 Memory 가 없으므로 값을 저장하는 목적이 아니고 연결을 위해 사용한다.

테스트벤치 -> 큰 그림으로 보면 테스트 벤치도 모듈이다! (디지털 논리회로 수업). Verilog 문법을 이용하여 작성하기에 확장자명도.v가 붙는다.

컴퓨터공학기초실험2 수업의 강의자료, 조교님들이 강조했던 부분인 컴파일 과정시 가장 많이 접하는 에러, top module setting이 적절하지 않은 경우가 많다고 하셔서 그 부분에 대해서 더욱 유념하면서 설계를 진행하였다. 실제로 실험 시간때 스스로 하는 과정에서 컴파일이 안되어 조교님께 도움을 받았던 사례였기에 더욱 유념하였다.

-About digital logic circuit theory

- 카르노맵이란, 우리가 구현한 회로의 게이트들을 나타내는 수식을 간략화 할 수 있는 도구이다. 2변수, 3변수, 4변수 카르노맵이 있고 이는 디지털 논리회로 1 수업때 배웠던 내용이다.
- 진리표 (Truth Table) 을 먼저 작성후 카르노맵에 대입하여 간략화된 수식을 이끌어 낼 수 있다. 이 과정을 이용하면 회로의 최소화된 식을 도출할 수 있다.
-

3. 설계 세부사항

Truth table in 2 to 1 MUX

진리표 (Truth Table)

D0	D1	S	Y
0	0	0	0
0	1	0	0
1	0	0	1
1	1	0	1
0	0	1	0
0	1	1	1
1	0	1	0
1	1	1	1

Mux 2_to_1 에서의 s,d0,d1의 input signal 과 output signal Y의 값들을 진리표를 통해 그릴 수 있다. 진리표에서 보면 S가 0이 되는 경우, Y값이 D0의 INPUT 값으로 나오고 S가 1이 되는 경우 D1의 INPUT값이 Y 값으로 정해진다는 것을 도출할 수 있다.

카르노맵

S / D0 D1	0	1
00	0	0
01	0	1
11	1	1
10	1	0

$$Y = D0S' + D1S$$

Hamming distance 가 1씩 차이나는 input을 써주고 1이 나오는 것을 테두리로 그린 후 이를 정리하면 2_to_1 mux의 최소화된 식이 나온다.

이번 과제에서 그림으로 나와진 input signal 을 d0,d1,s 로 정의하고, output signal을 y로 정의한다. 각 베릴로그 소스코드 파일을 두개로 나누는데, 하나는 MUX를 구성하는 NAND, INVERTER 게이트의 각 모듈을 정의한 파일 gates.v 파일과 Mux 라는 최종적인 모듈의 소스코드를 서술한다. 이는 mx2.v 파일로 저장한다.

gates.v 파일에서는 2개의 입력을 받는 _nand2 모듈과 1개의 입력을 받은 _inv 모듈을 만든다. 각각의 기능은 _nand2는 and 논리연산자와 ~ 부정연산자를 이용하여 assign 해주고 , inv 모듈은 ~ 부정 연산자를 output y라는 변수에 assign 해주고 endmodule 키워드를 이용하여 모듈 정의를 종료한다.

mx2.v 파일에서는 gates.v 파일에서 정의한 nand게이트와 inv 게이트 모듈을 인스턴스화 하여 mx2 라는 Top module 생성과정에서 이를 더해준다.

그리고 nd20 의 output을 wire 변수 w0와 mapping, iv0 의 output 을 wire 변수 sb와

mapping, nd21의 output을 wire 변수 w1과 mapping을 해준다. (mx2 에서 nd20,nd21, nd22, iv0 게이트들을 인스턴스화 하는 과정에서)

위 과정이 끝나면 회로를 전선으로 연결한 것 과 같은 과정을 시행했다는 뜻이다.

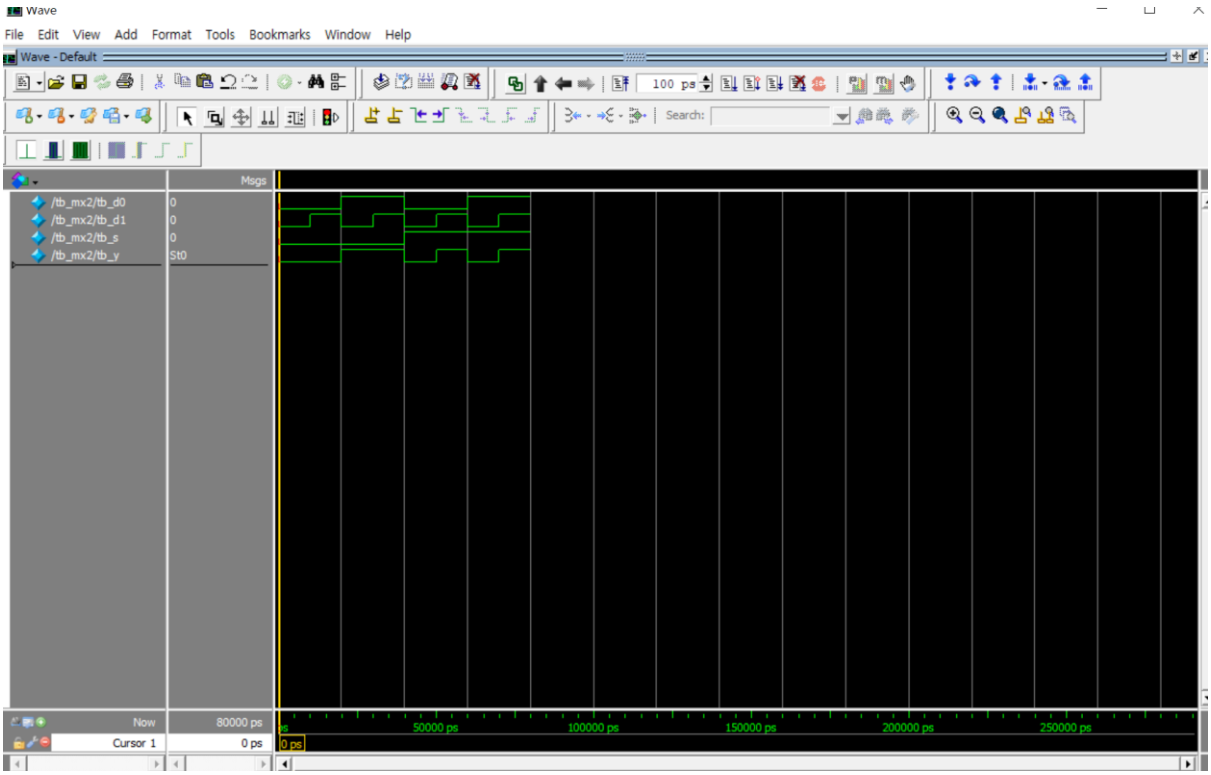
Top module mx2를 컴파일 하려면 top module과 sub module 설정을 진행한다.
RTL VIEWER를 통해서 내가 설계한 회로의 모양을 보고 과제에서 구현된 3 NAND GATE와 1 INVERTER를 구현한 모양을 볼 수 있다. 일단 VIEWER를 통해 구현된 회로의 모양으로 1차 검증을 한다.

2차 검증은 실제로 내가 구현한 회로가 INPUT에 따른 OUTPUT이 잘 출력된다는 것을 보여야 하므로 테스트 벤치 파일인 tb_mx2.v 파일을 만든 후 이를 검증하는 코드를 작성한다. 그후 Simulation tool을 이용하여 waveform을 통해 output이 잘 그려지는지 확인함으로써 회로 설계가 최종적으로 잘 마무리되었는지 확인한다.

4. 설계 검증 및 실험 결과

A. 시뮬레이션 결과

설계한 디자인을 검증하기 위하여 작성한 testbench에 대하여 간단하게 설명하고, waveform을 통하여 원하는 결과가 제대로 나오는 지를 확인한다.



`TIMESCALE 이라는 키워드를 이용해서 정확성과 시간 단위를 설정한다.

그 후 각각 시간 지연을 줌으로써 검증하고 싶은 INPUT 값을 변환하면서 OUTPUT값이 WAVEFORM에 잘 그려지는지 시각적으로 확인하기 위하여 INITIAL BEGIN ~ END 구문에서 검증하고 싶은 INPUT SIGNAL들의 값을 지정해준 후 RTL SIMULATION을 통해 확인하는 틀이다.

맨 처음 TIMESCALE을 지정할 때 1ns/100ps 로 지정했기 때문에 initial begin 구문 안에서 조건을 지정한 후 시간 지연을 #10을 써주면 timescale에서 지정했던 1ns/100ps는 10^3 이고 이를 *10 하면 10000ps을 기준으로 테스트 케이스들이 구분되어 waveform에 그려지는 형태를 시각적으로 확인할 수 있다.

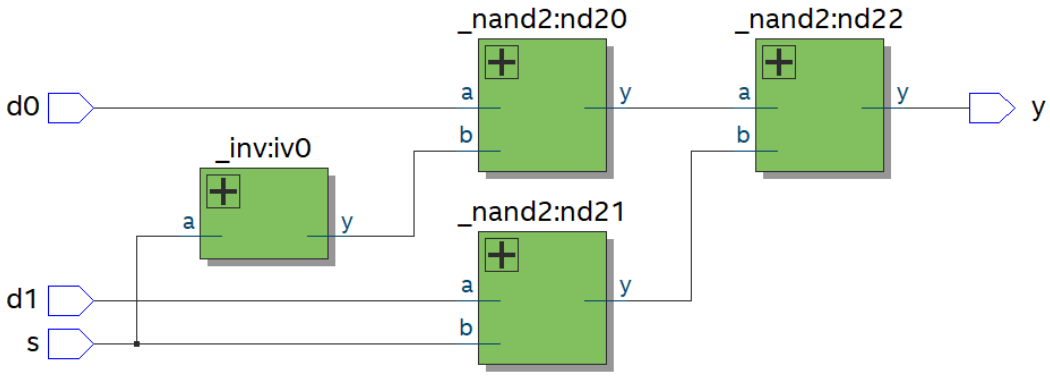
테스트 벤치 소스 파일에 검증했던 구문에 대해서 설명해보면, 10000ps 기준으로 맨 처음 d0 d1 s 에 각각 0,0,1 이 그려지고, y 결과값에 따른 값이 0임을 알 수 있다.

근데 여기서 d0 d1이 둘다 0이기에 input signal s가 어떤 input을 output으로 선택했는지 검증을 해봐야 한다. 즉 truth table에 작성한 총 8가지 case에 대해서 검증을 해야한다.

위 과정을 검증하기 위해 initial begin ~end 구문안에 총 8가지의 input set를 넣고 waveform에 따른 output이(카르노맵을 통해 도출한 수식)

예상한 것처럼 s가 0일때 d0의 값을, s가 1일때 d1의 값을 output값인 y로 출력해준다는 것을 waveform의 파형으로 검증함으로써,구현한 2 to 1 mux가 3개의 2 input nand gate와 1개의 inverter gate로 구현이 되었음을 확인 할 수 있다.

B. 합성(synthesis)결과



(=

COMPILE 후, RTL MAP VIEWER 로 내가 만든 게이트들의 모양을 그림으로 확인할 수 있다.

Nand gate 3개 와 1개의 inverter gate 가 구성되어 있고, 각 모듈의 구현했던 이름까지 동일하다는 것을 알 수 있다. 또한 2to 1 mx2 모듈의 input signal d0 d1 s, output signal y 까지 과제의 모듈 요구사항들의 모양에 맞게 잘 서술되어 있다.

컴퓨터공학기초실험2, 디지털논리회로2 수업시간때 코드를 실행하는 cpu의 종류에 따라 위의 시각적인 모델의 모양의 차이가 다소 있다는 점 또한 고려하여 회로의 구성이 잘 되어있는지 정도를 먼저 확인하였다.

Flow Summary

Flow Summary

<<Filter>>

Flow Status	Successful - Tue Sep 12 17:53:47 2023
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	mx2
Top-level Entity Name	mx2
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	1 / 41,910 (< 1 %)
Total registers	0
Total pins	4 / 499 (< 1 %)
Total virtual pins	0
Total block memory bits	0 / 5,662,720 (0 %)
Total DSP Blocks	0 / 112 (0 %)
Total HSSI RX PCSs	0 / 9 (0 %)
Total HSSI PMA RX Deserializers	0 / 9 (0 %)
Total HSSI TX PCSs	0 / 9 (0 %)
Total HSSI PMA TX Serializers	0 / 9 (0 %)
Total PLLs	0 / 15 (0 %)
Total DLLs	0 / 4 (0 %)

Flow summary 를 보면 Top-level Entity name에 우리가 설계 한 최상단 모듈인 mx2가 적힌 것으로 보아 설계한 모듈이 잘 작동한다는 것을 알 수 있다.

우리가 프로젝트 생성 과정때 선택했던 device 명인 5CSXFC6D6F31C6 또한 나타나져 있다. 또한 설계를 하면서 사용했던 레지스터의 수와 PIN 수 또한 알 수 있다.

C. FPGA board targeting 결과

이번 ASSIGNMENT 에서는 FPGA board targeting 을 이용한 과제가 아닙니다.

5. 고찰 및 결론

A. 고찰

compile 을 하기 전에 top module 설정을 잘 해줘야 한다는 것을 깨닫게 되었다. top module 설정은 c언어와 같은 program language에서 main 함수 설정과 유사한데, top module setting이 적절치 않은 경우 qutaurs prime 환경에서는 error 가 뜬다.

처음에 친숙하지 않은 환경에서 프로그래밍을 하기때문에 가장 중요한 부분을 인지하지 못하고 살짝 헤매게 되었는데, 앞으로는 compile 전 module setting을 유의하면서 진행해야겠다 생각했다. 이때 오타도 상당히 조심해야한다.

물론 변수명이 달라서 틀리는 경우는 금방 해결하겠다만, top module error에서도 input output 과 같은 port들이 정확하게 definite 되지 않는 경우도 error 를 접할 수 있다. 또한 module을 인스턴스화 할 때도 source file 작성시 오타가 있는 경우도 error가 뜨니 다시 한번 검사해보는 습관을 길러야겠다. 맨 밑 script 창에 오류 id 와 message를 통해서 인텔 레퍼런스나 검색을 이용하여 해결하였다.

B. 결론

멀티플렉서 중 2 to 1 mux를 설계하는 과정에서 꼭 한가지 형태의 gate들의 조합으로 만들어진다고 생각하였지만, nand gate와 inverter gate를 이용함으로써 MUX를 설계할 때 대부분 많이 설계하는 and gate 2개와 or gate, inverter gate 1개 와 같이 고정적인 회로 설계의 사고에 갇혀 있었는데, 위와 같이 회로의 종류를 바꿈으로써 cost가 nand gate가 and gate보다 작음으로써 어떻게 설계를 하느냐 에 따라서 cost와 circuit의 효율성이 달라진다는 것을 몸소 체험하게 되었다. 복잡한 반도체 칩 설계시 위와 같은 과정들이 기초가 되어 설계시 이를 유의하면서 또 고안하는 습관이 하드웨어 설계 엔지니어들의 기본 소양임을 다시 깨닫게 되었다.

또한 베릴로그라는 언어를 이용하여 하드웨어 모듈을 처음으로 설계해봄으로써 디지털 논리회로 수업때 배웠던 이론들을 적용시켜서 간단한 회로구성이지만 머리속의 이론들을 직접 적용해봄으로써 아직 미숙하지만 베릴로그라는 언어와 하드웨어에 친해지게 되는 계기가 되었다.

6. 참고문헌

DAVID MONEY HARRIS & SARAH L.HARRIS /DIGITA DESIGN AND COMPUTER ARCHITECTURE SECOND EDITION/Elsevier Korea L.L.C/2012 JANUARY 1

이준환 교수님/디지털논리회로1,2/ 새빛관 광운대학교 컴퓨터 정보공학부 교수님/ 2020(1학기 수강시 년도),2023(2과목 수강년도)

INTEL공식사이트/
https://www.intel.com/content/dam/www/centrallibraries/us/en/documents/quartus-prime-software-brochure.pdf