컴퓨터 공학 기초 실험2 보고서

실험제목: 2-to-1 MUX

실험일자: 2022년 09월 14일 (수)

제출일자: 2022년 09월 20일 (수)

학 과:컴퓨터정보공학부

담당교수: 공진흥 교수님

실습분반: 수요일 0, 1, 2

학 번: 2019202021

성 명: 정 성 엽

1. 제목 및 목적

A. 제목

2-to-1 MUX

B. 목적

주어진 2-to-1 MUX를 Verilog를 통해 설계한다. Verilog를 통해 설계한 MUX가 제대로 동작하는지는 Testbench를 직접 제작하여 시뮬레이션 한다. 이 실험을 통해 Verilog를 통한 디지털논리회로 구현방법과 Testbench를 이용한 검증을 익힌다.

2. 원리(배경지식)

MUX(멀티플렉서)

멀티플렉서는 여러 개의 입력 중 특정 조건에 의해 input을 오직 하나만 선택하고 이를 출력하는 역할을 한다. 이를 MUX라고도 불리며 입력 한 선들 중 s 신호와 input에 조합에 의해 출력이 정해진다. 예를 들어 s가 0인 경우 d0 의 값이 출력되고, s가 1인 경우 d1의 값이 출력된다.

이와 반대로 디멀티플레서가 있으며 이는 단 하나의 입력을 통해 여러 개의 출력으로 만드는 역할을 한다. 디먹스에는 먹스와 다르게 enable input 기능이 있어 E신호가 0일때만 output을 출력하도록 할 수 있다. 종합해 봤을 때 디멀티플렉서는 데이터 분배기이다.

Verilog 문법

모듈 설정

Verilog 파일에서 모듈의 시작과 끝은 module로 시작하여 endmodule로 끝나야 한다. 그리고 해당 모듈의 로직은 그 사이에서 구현하게 된다.

assign이란

Assign은 할당하는 것으로 c 언어의 return과 비슷한 느낌이라고 볼 수 있다. 즉 대입의 역할을 하게 된다.

Wire과 reg이란

Wire은 메모리가 없는 것으로 입력이 끝나는 순간 출력이 사라져 module 사이에 신호 전달 역할로 쓰이지만, reg 같은 경우 wire과 비슷한 역할임과 동시에 메모리가 있어서 신

호를 저장한다. 이는 testbench에서 input은 reg로 표현하고 output할 것은 wire로 표현함을 통해 다시 생각할 수 있다.

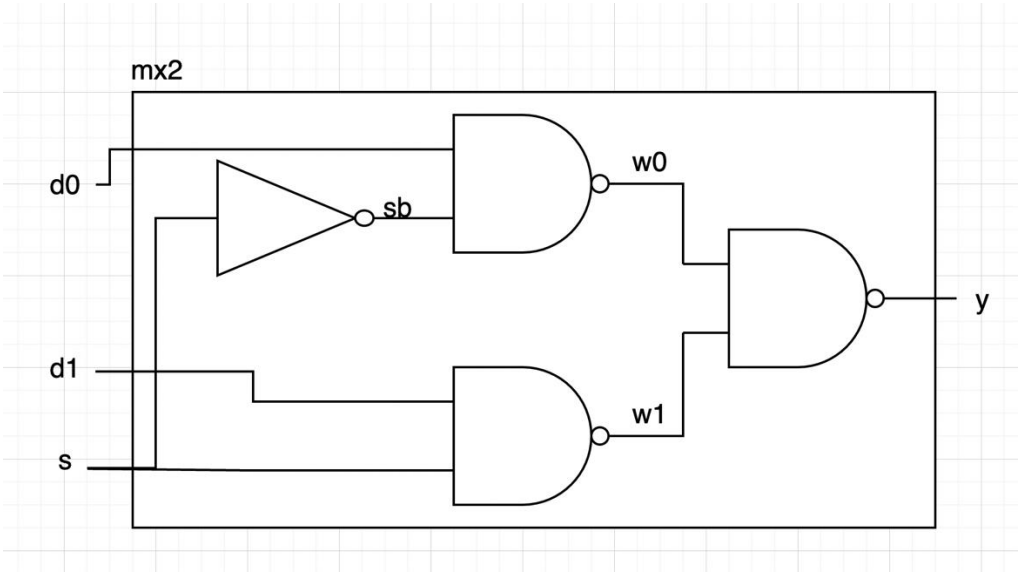
OR, AND, NOT 연산이란

1학기 컴퓨터공학기초실험1, 디지털논리회로1 에서 배웠던 논리 gate로 logic operation이다. OR 연산인 경우 |, AND 연산인 경우 &, NOT인 경우 ~를 활용해 논리를 나타낸다.

Top-level module이란

C언어에서 main 함수가 있다면 Quartus의 Verilog에는 top-level이 있다. 이는 컴파일을 돌릴 메인 모듈이 어떤 것인지 체크하는 것으로 이가 잘못되어있다면 디버깅 및 컴파일에 문제가 생긴다. 이는 Assignment의 setting에서 수정할 수 있다.

3. 설계 세부사항



위 그림을 참고하여 3개의 2-input NAND gate와 1개의 inverter로 이루어진 2-to-1 MUX를 Verilog를 사용하여 설계한다. 2-to-1 MUX의 input은 d0, d1, 및 s이며 output은 y로 총 한개이다. 2-to-1 MUX 내부 게이트에서의 결과는 각각 sb, w0, w1으로 설정한다.

이를 진리표로 나타내면 아래와 같다

s	Input		mx2 내부			output
	d0	d1	sb	w0	w1	y

0	0	0	1	1	1	0
0	0	1	1	1	1	0
0	1	0	1	0	1	1
0	1	1	1	0	1	1
1	0	0	0	1	1	0
1	0	1	0	1	0	1
1	1	0	0	1	1	0
1	1	1	0	1	0	1

위의 진리표를 바탕으로 분석하면 s가 0일때에는 d0의 값이 y에 출력되고 s가 1일때에는 d1의 값을 그대로 출력하게 된다.

이를 카르노 맵으로 다시 분석해보면 아래와 같다.

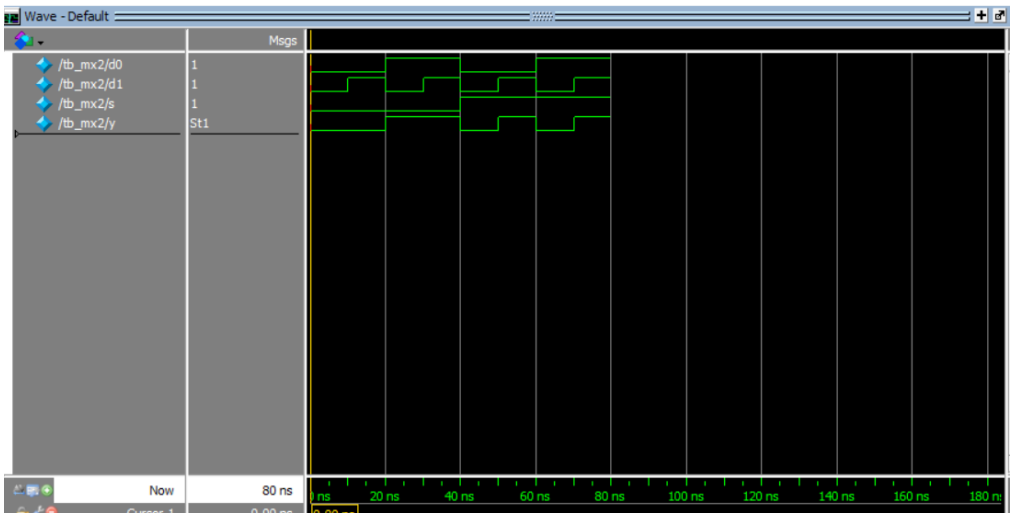
s \ d1, d0	00	01	11	10
0	0	1	1	0
1	0	0	1	1

카르노 맵을 통해 $\bar{s}d0 + sd1$ 라는 식을 얻을 수 있고 이를 드 모르간 법칙을 통해 $\overline{\overline{\bar{s}d0} \cdot \overline{sd1}}$으로 나타낼 수 있으며 이는 위에서 보인 회로 그림과 같은 논리이다.

그래서 mx2 모듈 전 _nand2 와 _inv 모듈을 설계하였으며 _nand2 같은 경우 input에 a, b 그리고 output y로 설정하고 assign을 통해 y=~(a&b)로 할당하였으며 _inv 모듈에서는 input에 a 그리고 output y로 설정하고 assign을 통해 y=~a로 할당하였다. 그리고 mx2 모듈에서는 만든 NAND와 INV 모듈을 활용하여 위와 같은 논리식으로 설계하였다.

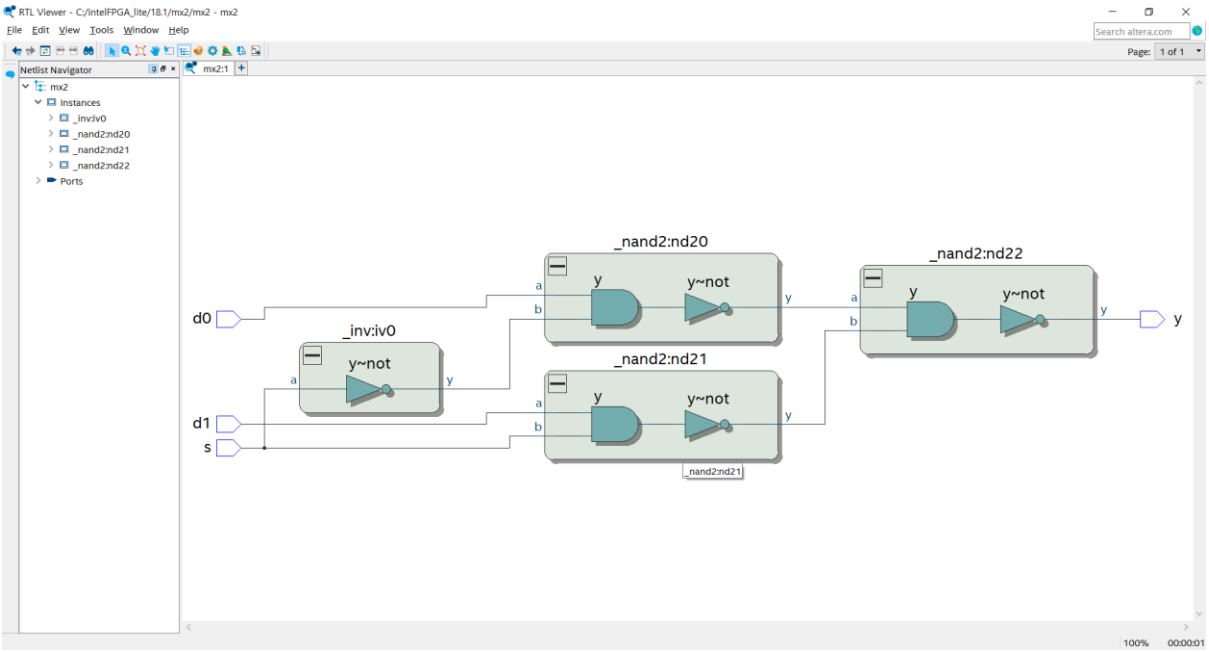
4. 설계 검증 및 실험 결과

A. 시뮬레이션 결과



2-to-1 MUX의 truth table의 모든 경우인 8개를 testbench에 직접 작성하여 input s, d0, d1의 조합에 따른 output y의 결과를 waveform으로 출력하였고 이에 대할 결과를 캡처하였다. S가 0일 때 d0의 값이, s가 1일 때 d1의 값이 y로 출력됨을 확인할 수 있다.

B. 합성(synthesis) 결과



과제에서 주어진 것 그리고 카르노 맵을 통해 논리식을 나타낸 것과 같이 1개의 inverter와 3개의 nand gate를 통해 2-to-1 mux를 구현함을 보일 수 있다. 기존에 각 모듈에 대한 설명이 생략되어 있었으나 이를 펼친 것이 위의 사진과 같다. 이를 보면 input, output의 개수와 변수명 등 조건에 있는 2-to-1 MUX 구조를 설계했음을 볼 수 있다.

Flow Summary

Flow Summary

<<Filter>>

Flow Status	Successful - Tue Sep 20 23:42:17 2022
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 Summary는 위와 같으며 Flow status에 Successful이 출력되어 컴파일이 잘 완료되었음을 볼 수 있고 logic 개수 또한 logic utilization을 통해 볼 수 있다. 사용된 Register의 개수는 Total registers를 통해 알 수 있지만 이번엔 사용된 registers는 없다.

5. 고찰 및 결론

A. 고찰

이번 실험은 Verilog의 사용법을 익히고, 간단한 2-to-1 MUX를 설계하는 것으로 처음 접해본 Quartus와 Verilog는 여러 어려운 부분이 많았다. 특히 이전까지 배운 top level entity를 정해주는 것과 testbench를 작성하여 waveform을 구현하는 것은 처음 배운 입장에서는 매우 헷갈리는 부분이었다.

이번 MUX 설계에서는 다행히 mx2 부터 설계하였기 때문인지 top-level에서는 문제가 없었다. 다만 gates.v 라는 파일로부터 모듈을 잘 못 가져온 것인지 모를 문제 때문에 다시 제작했을 때 정상 작동하였다.

Testbench 이후에는 Testbench 파일 설정을 통해 직접 등록 해주고 ModelSim을 실행하여 waveform을 출력하여 결과를 확인했다. 기존에 배운 프로그래밍과 이질적인 부분이 많아 설계하고 실행하는데 어려움이 있었다.

B. 결론

간단한 2-to-1 MUX를 구현함으로써 Verilog의 기초적인 사용법과 ModelSim의 사용법 그리고 그를 이용한 MUX의 waveform을 시뮬레이션 하였고, 특히 Verilog는 기존에 배웠던 C, C++ ,Python과 같은 순서/객체 지향적 언어와 달리 디지털논리적으로 하드웨어를 직접 설계하는 HDL언어로 써 기존에 배웠던 언어들과 차이점과 Verilog의 특징과 기본적인 사용법을 익힐 수 있었다.

6. 참고문헌

1) David Mondy Harris and Sarah L. Harris / Digital Design and Computer Architecture/ Elsevier Korea L.L.C / 2012
2) 공진흥 교수님/ 디지털논리회로1/광운대학교 컴퓨터정보공학부/2022
3) 공영호 교수님/ 디지털논리회로2/광운대학교 컴퓨터정보공학부/2022
4) 공진흥 교수님/ 컴퓨터공학기초실험2/광운대학교 컴퓨터정보공학부/2022