컴퓨터 공학 기초 실험2 보고서

실험제목: 2-to-1 MUX

실험일자: 2021년 09월 13일 (화)

제출일자: 2021년 09월 18일 (일)

학 과: 컴퓨터공학과

담당교수: 공영호 교수님

실습분반: 화요일 0, 1, 2

학 번: 2021202058

성 명: 송채영

1. 제목 및 목적

A. 제목

2-to-1 MUX

B. 목적

Quartus의 사용법을 익히고 3개의 2-input NAND gates와 1개의 inverter로 이루어진 multiplexer 구조를 Verilog를 통해 설계해본다. 또한 test bench를 수행해 waveform을 통해 설계한 MUX가 잘 작동하는지 확인해 본다. 위 과정을 통해 verilog의 기초를 알아본다.

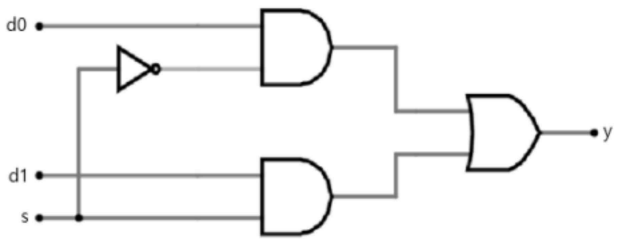
2. 원리(배경지식)

Verilog는 전자 회로 및 시스템에 사용되는 하드웨어 기술언어이며, if문이나 for문과 같은 구문이나 문법이 C언어와 유사하다. verilog를 통해 simulation(시뮬레이션), synthesis(합성), library, documentation(문서화)가 가능하다다.

verilog는 module로 시작하여 마지막은endmodule로 끝난다. 또한 작성한 코드를 컴파일 한 후 testbench를 통해 제대로 설계가 되었는지 설계된 회로도를 볼 수 있는 RTL viewer와 결과값을 확인하는 waveform으로 확인할 수 있다.

MUX는 여러 아날로그 또는 디지털 입력 신호 중 하나를 선택하여 선택된 입력을 하나의 라인에 전달하는 장치이며, multiplexer의 줄임말이다. 보통 선택선의 조합에 의해 선택된 2^n개의 입력선 중 하나를 선택해 출력선에 연결시켜 주며, 여러개의 회로가 단일 회선을 공동으로 이용해 신호를 전송하는데 사용한다.

실습에 나온 2 to 1 multiplexer은 입력값 두 개 중 1개를 출력선으로 연결시켜 준다. 아래의 그림은 2 to 1 multiplexer의 회로도이며 inverter 1개, AND gate 2개, OR gate 1개로 구성되어 있다.



2 to 1 multiplexer의 진리표는 다음과 같다.

s	y
0	d_0
1	d_1

위의 표에서 2 to 1 multiplexer는 입력값 s가 0일 때 d_0를 출력하고 입력값 s가 1일 때 d_1를 출력하는 것을 알 수 있다.

이를 바탕으로 MUX의 논리식을 카르노맵으로 나타내면

s \ $d_{1:0}$	00	01	11	10
0	0	1	1	0
1	0	0	1	1

논리식은 $y = \bar{s}d_0+sd_1$이다.

3. 설계 세부사항

2 input nand gate는 _nand2의 이름으로 하여 구현하였다. 이때 입력값은 a, b 출력값은 y를 가진다. 논리식은 y = ~(a&b)로 2 input nand gate의 진리표는 다음과 같다.

a	b	y
0	0	1
0	1	1
1	0	1
1	1	0

inverter는 _inv의 이름으로 구현하였다. 이때 입력값은 a, 출력값은 y를 가진다. 논리식은 y = ~a로 inverter의 진리표는 다음과 같다.

a	y
0	1
1	0

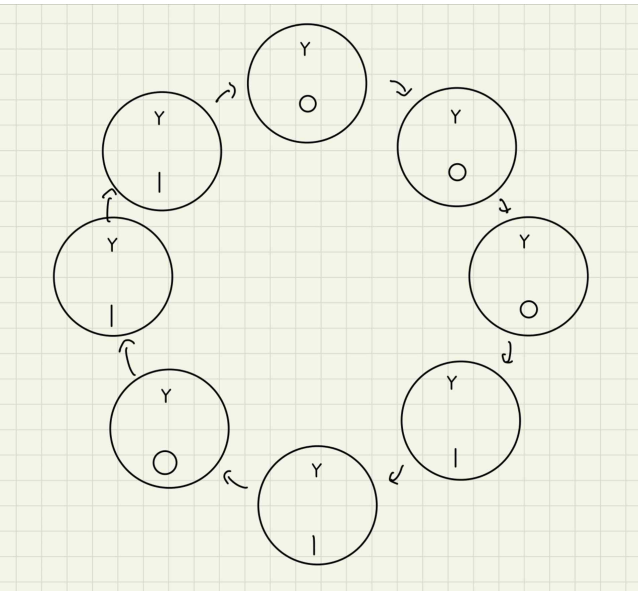
2 to 1 multiplexer는 mx2의 이름으로 구현하였다. 이때 입력값은 d0, d1, s이며 출력값은 y를 가진다. 위의 두가지, _nand와 _inv를 사용해서 mx2를 구현하였는데, 이때 iv0에서 nd20으로 가는 wire를 sb, nd20에서 nd22로 가는 wire를 w0, nd21에서 nd22로 가는 wire를 w1이라고 선언해주었고, iv0, nd20, nd21, nd22를 표현하였다.

tb_mx2는 mx2파일이 제대로 설계 되었는지 확인하는 test bench이다. 이때 tb_d0, tb_d1, tb_s에 값을 바꿔 넣어주어 결과값 y를 확인하였다.

밑의 표는 이를 진리표(truth table)과, 상태천이도(state diagram)로 나타낸 것이다.

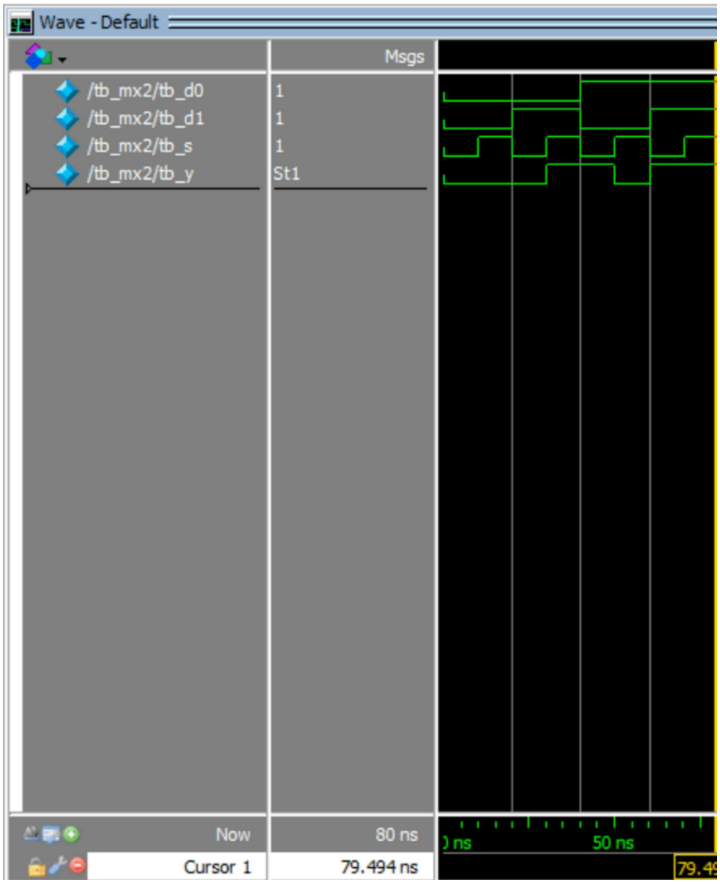
d0	d1	s	sb	wo	w1	y
0	0	0	1	1	1	0
0	0	1	0	1	1	0
0	1	0	1	1	1	0
0	1	1	0	1	0	1
1	0	0	1	0	1	1
1	0	1	0	1	1	0

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



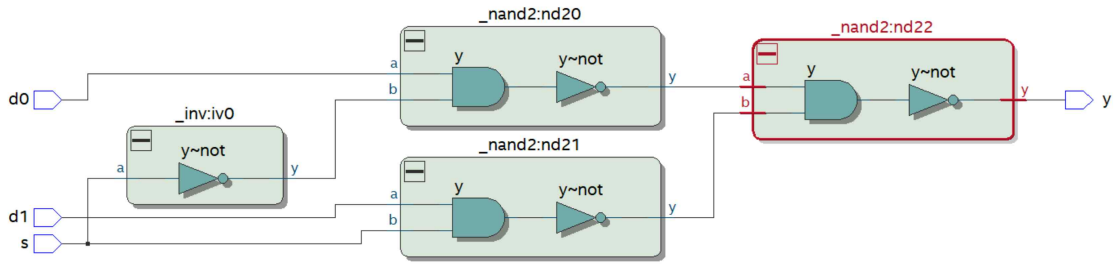
4. 설계 검증 및 실험 결과

A. 시뮬레이션 결과



d0, d1, s, y의 순서대로이며, test bench인, tb_mx2는 mx2를 load하여 8개의 경우에 대해 d0, d1, s의 값을 변경시키며 y값의 변화를 확인하였다. 이때 `timescale의 unit/precision은 강의 자료에 나온 것을 참고하여 1ns/100ps로 설정해 주었다. waveform을 통해 위의 진리표와 동일한 결과임을 알 수 있다.

B. 합성(synthesis) 결과



compile을 확인하기 위해 위의 그림과 같은 RTL viewer의 결과를 볼 수 있었다. 이를 통해 주어진 2 to 1 multiplexer의 그림과 같으며 설계가 잘 되었다는 것을 알 수 있었다.

Flow Summary

<<Filter>>

Flow Status	Successful - Sun Sep 18 19:50:36 2022
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	assign1
Top-level Entity Name	mx2
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	4
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

위의 그림은 flow summary로 flow status가 successful임을 확인하여 코드에 문제가 없음을 확인하였다. 또한 초기에 설정하였던 내용도 확인할 수 있었다.

5. 고찰 및 결론

A. 고찰

quartus를 처음 다루어보니 파일을 생성하는 것 부터 컴파일 후 waveform을 확인하는 것 까지 아직 익숙하지 않은 것 같다. 또한 간단한 실수, 예를 들어 _inv를 __inv로 쓴다던가, test banch에서는 `가 아닌'를 써서 오류가 난다던가 등등 작은 실수들이 있어 다소 시간이 더 걸렸던 것 같다. module과 endmodule, ;를 쓰는 것, 주석 처리 하는 것 등등이 C언어와 비슷해 그런 점에서는 익숙했다. gates와 mx2의 코드를 구성하는 것 보다는 test banch 부분이 많이 헷갈렸던 것 같다. 실습 자료에 있는 xor gate의 test banch를 먼저 이해한 후 코드를 짜보니 많은 도움이 되었다. 또한 test banch의 코드에서 8개의 경우를 넣어주는 부분에서 1을 0으로 써서 오류가 났었는데 이부분을 waveform에서 확인하고 수정할 수 있어 좋았던 것 같다. 코드를 다 짜고 나서도 설계 검증 및 실험결과에 넣어야할 것들을 확인하는 것에서도 많은 시간이 걸렸지만 다음 실습 때는 시간 단축이 가능할 것 같다.

B. 결론

2학년 1학기 때 다른 교수님의 디지털논리회로1 수업을 들어 verilog를 접해보지 못해 처음 겁부터 먹었었다. 아직 첫 실습이다 보니 다소 쉽게 나왔겠지만, 생각했던 것 만큼 어렵게 느껴지지는 않았다. 또한 이론으로만 배웠던 내용들을 verilog를 통해 구현하는 점과 배웠던 내용을 활용한다는 점이 좋았고 복습도 된 것 같다. 또한 1학기 때는 손으로 그려가며 설계를 했었는데 verilog를 이용하여 더욱 편리하게 설계 할 수 있었던 것 같고 시각적인 효과도 볼 수 있었다. 다음에는 2 to 1 multiplexer뿐만 아니라 다른 것들도 구현할 수 있을 것 같다.

또한 위의 고찰에서 언급했듯이 첫 번째 실습에서 나왔던 실수들로 오류를 범하지 않도록 주의해야 겠다고 느꼈다.

6. 참고문헌

공영호 교수님/디지털논리회로2 강의자료/광운대학교컴퓨터 공학과 2022 강의자료

한국기술교육대학교/Verilog HDL 문법/한국기술교육대학교 전기전자통신공학부/강의자료