컴퓨터 공학 기초 실험2 보고서

실험제목: Memory & Bus

실험일자: 2022년 11월 22일 (화)

제출일자: 2022년 11월 23일 (수)

학 과: 컴퓨터공학과

담당교수: 공영호 교수님

실습분반: 화요일 0, 1, 2

학 번: 2021202058

성 명: 송채영

1. 제목 및 목적

A. 제목

Memory & Bus

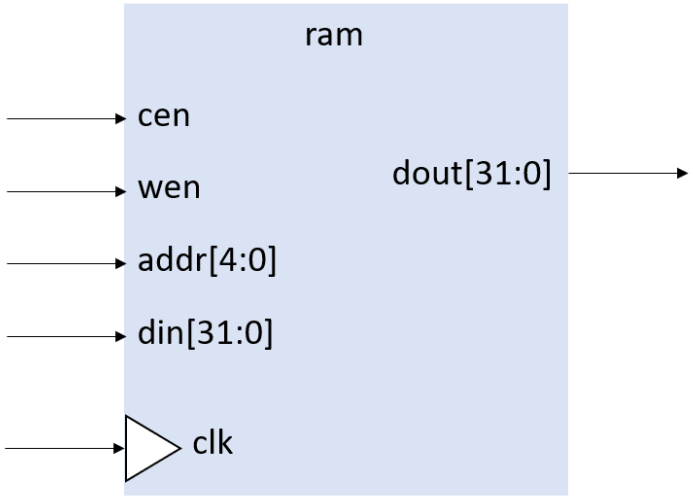
B. 목적

Address 에 기반해서 데이터를 저장하는 하드웨어인 ram 에 대해 알아보고 설계한다. 여러 component 들 간에 data 를 전송할 수 있도록 연결해주는 bus 에 대해 알아보고 설계해본다. 또한 Verilog HDL 에서 메모리를 선언하고 초기화 하는 방법을 알아본다.

2. 원리(배경지식)

- Memory

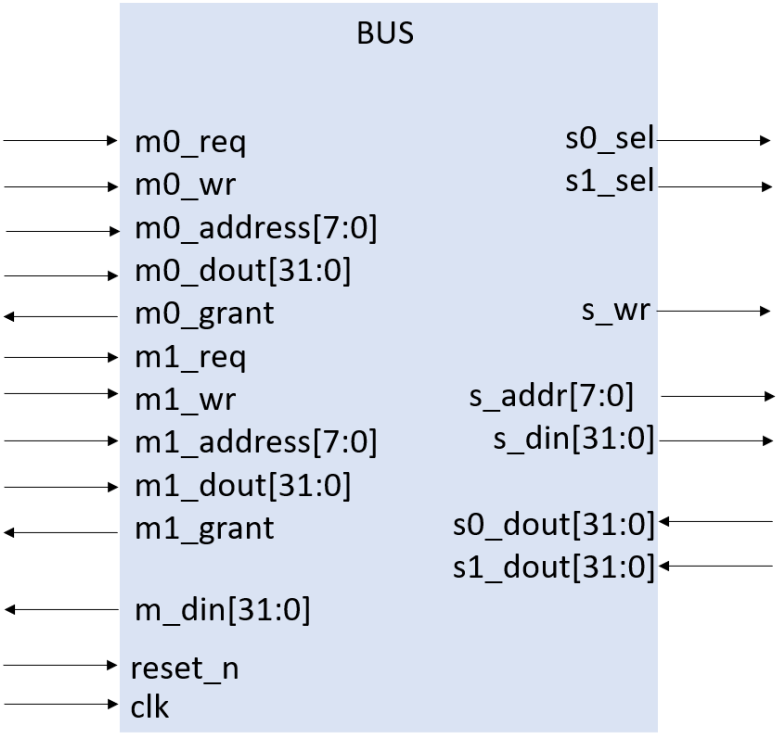
Memory는 크게 휘발성(Volatile)과 비휘발성(Non-Volatile)로 나뉘는데, 각각의 특성에 따라 ROM과 RAM으로 나뉜다. ROM은 Read Only Memory를 뜻하며, RAM은 Random Access Memory를 뜻한다. 그 중 이번 실습에서는 Ram을 설계했다.



RAM 은 Address 에 기반하여 데이터를 저장하는 hardware 로 프로그램이 실행되는 동안 기억된 정보를 읽거나 다른 정보를 기억할 수 있는 메모리이다. RAM 은 휘발성 Memory 이기 때문에 전원이 꺼지면 가지고 있던 데이터가 사라지게 된다. 또한 미리 정해진 순서로만 저장 미디어에 있는 데이터를 액세스 할 수 있다는 특징이 있다. ROM 은 RAM 과는 반대로 비휘발성 Memory 이므로 전원이 꺼져도 데이터가 사라지지 않는다. Ram 의 종류에는 SRAM, DRAM, SDRAM 등이 있다. SRAM 은 Static Ram 으로 전원이 공급되는 한 저장된 데이터가 지워지지 않는다. 또한 DRAM 보다 일반적으로 속도가 빠르지만 용량이 작고 가격이 비싸다. DRAM 은 Dynamic RAM 으로 휘발성

메모리이다. SRAM 보다 가격도 싸고 용량이 크지만 대비 속도가 느리다. 또한 DRAM 은 capacitor 를 통해 데이터를 저장하는 방식을 사용한다.

- Bus

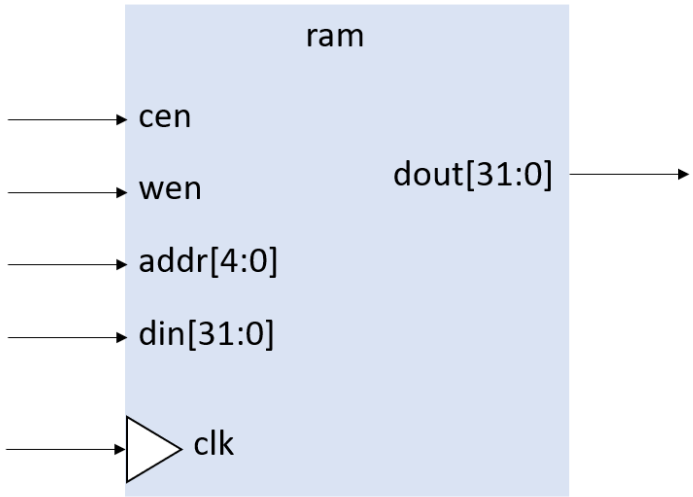


Bus는 component들 간에 data를 전송할 수 있도록 연결해주는 통로이다. Bus는 크게 데이터 버스, 주소 버스, 제어 버스로 나뉜다. 데이터 버스는 시스템 모듈 사이의 데이터 이동 경로를 제공한다. 주소 버스는 데이터의 근원지나 목적지에 일정한 메모리 주소를 전달하는 버스를 말한다. 마지막으로 제어 버스는 데이터 버스와 주소 버스를 제어하기 위해 사용한다. Operation에는 Read와 Write가 있다. BUS에는 master와 slave가 있는데, master은 bus를 통해 데이터를 전송하라고 요청하거나 시작하며, slave는 요청된 데이터 전송을 처리하는 역할을 한다. Bus의 내부에는 Arbiter, 중재자가 있으며 bus master를 조절하는 역할을 한다. 즉 여러 Bus Master들이 서로 Bus를 장악하려고 할 때 특정 Bus Master에게 Bus를 제공해준다. Bus는 새로운 component들을 추가하기가 쉬우며, 가격이 저렴한 특징을 가지고 있다.

3. 설계 세부사항

- Simple Memory

구분	이름	비트 수	설명
input	Clk	1-bit	Clock
	Cen	1-bit	Chip Enable
	wen	1-bit	Write enable
	Addr	5-bit	Address
	Din	32-bit	Data in
output	dout	32-bit	Data out

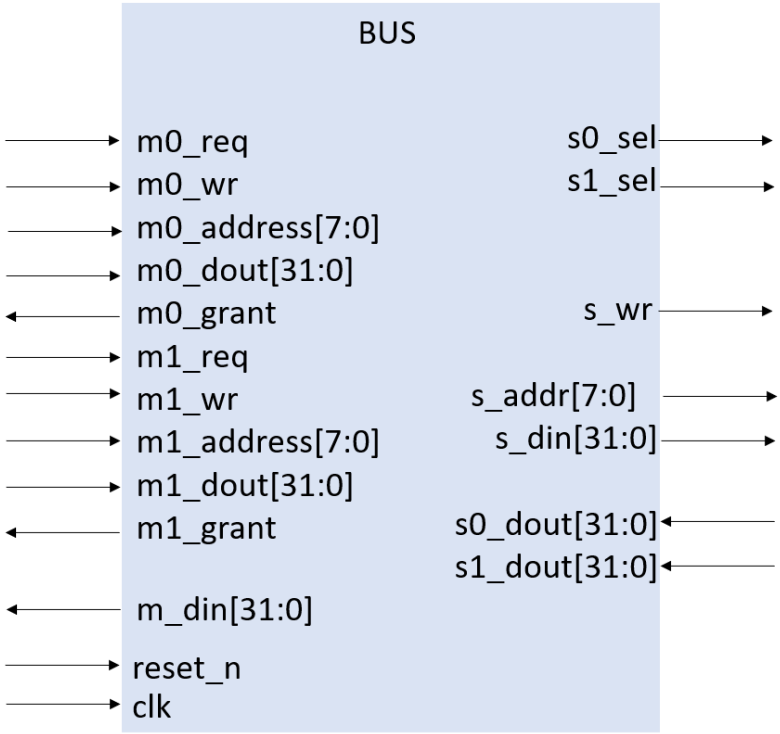


Cen과 wen이 모두 1일 때 address가 가리키는 memory에 din을 write하고 dout은 0을 출력하도록 한다. Cen이 1이고, wen이 0이면 address가 가리키는 memory의 값을 dout에 write하도록 한다. Cen이 0일 경우 dout은 0이 되도록 한다.

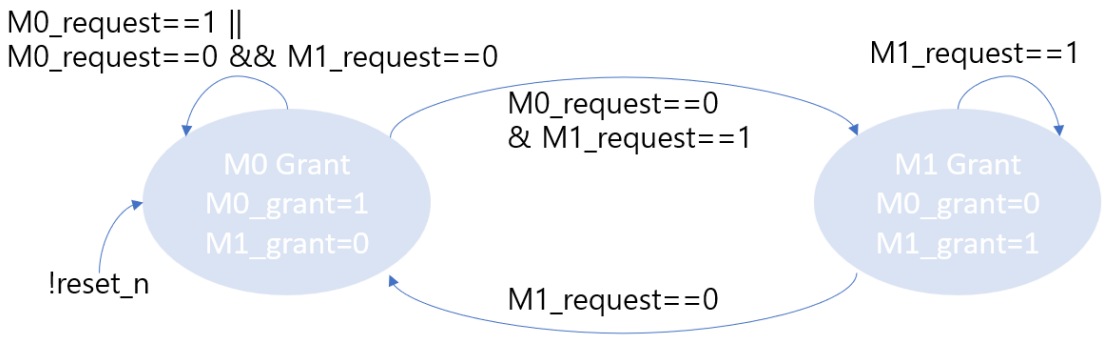
- Bus

Direction	Port name	Description
Input	Clk	Clock
	Reset_n	Active low reset
	M0_req	Master 0 request
	M0_wr	Master 0 write/read
	M0_address[7:0]	Master 0 address
	M0_dout[31:0]	Master 0 data output
	M1_req	Master 1 request
	M1_wr	Master 1 write/read
	M1_address[7:0]	Master 1 address
	M1_dout[31:0]	Master 1 data output
	S0_dout[31:0]	Slave 0 data output
	S1_dout[31:0]	Slave 1 data output

Output	M0_grant	Master 0 grant
	M1_grant	Master 1 grant
	M_din[31:0]	Master data input
	S0_sel	Slave 0 select
	S1_sel	Slave 1 select
	S_address[7:0]	Slave address
	S_wr	Slave write/read
	S_din[31:0]	Slave data input

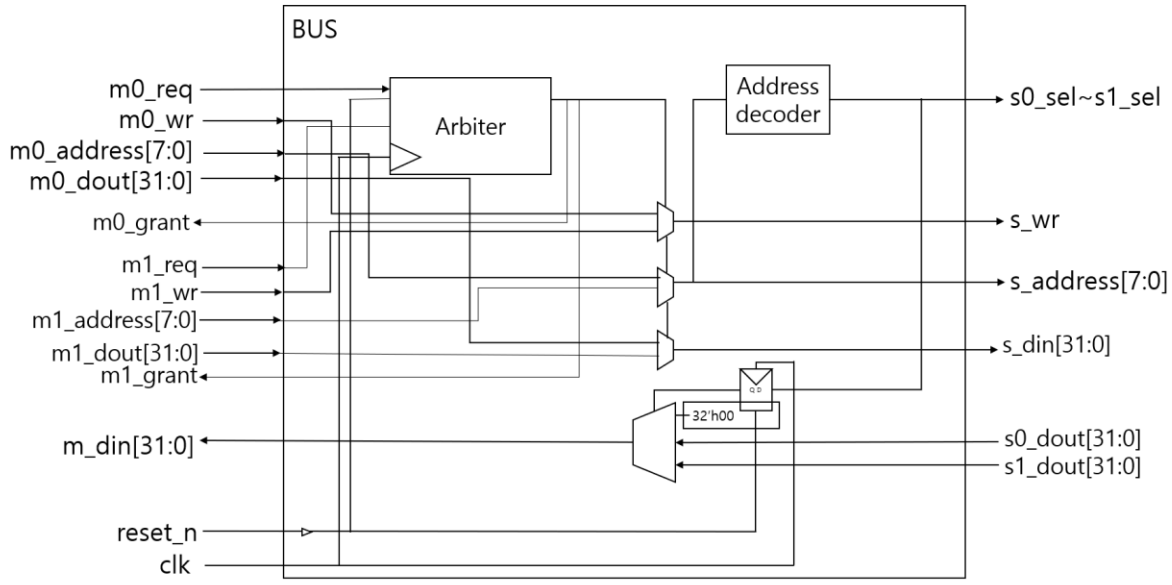


2개의 master와 slave를 가지고 있으며, slave0은 0x00~0x1F까지의 주소 범위를 가지고, slave1은 0x20~0x3F의 주소 범위를 가지며 이 외의 경우는 선택하지 않는다.



master가 bus를 통해서 data를 전송하고자 할 때, 해당 request signal을 1로 만든 뒤, grant signal을 받은 후 데이터 전송을 한다. 또한 grant signal을 받고 request signal이 1

이라면 bus는 data transfer을 계속할 수 있도록 bus의 소유권을 빼앗기지 않는다. 만약 두 master 모두 request를 진행하지 않는다면 grant signal은 Master 0이 받게 된다.

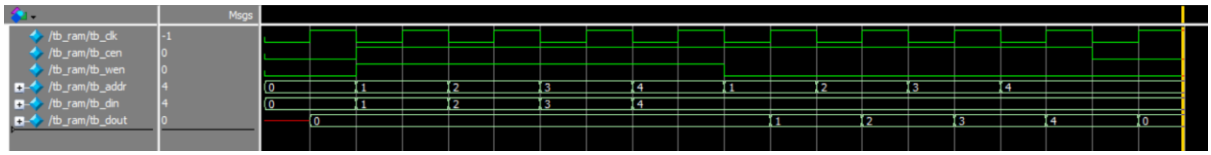


Bus Design으로 Bus의 각 내부에 grant에 따라 현재 상태를 결정해주는 Arbiter와 mux, Address decoder가 있다. Address decoder에서는 slave의 각 주소에 따라 s0_sel, s1_sel을 결정하며 s_address의 상위 3bit를 비교해서 000이면 s0_sel을 1, s1_sel을 0로, 001이면 s0_sel을 0로, s1_sel을 1로 다른 경우에는 s0_sel을 0으로, s1_sel을 1로 출력한다.

4. 설계 검증 및 실험 결과

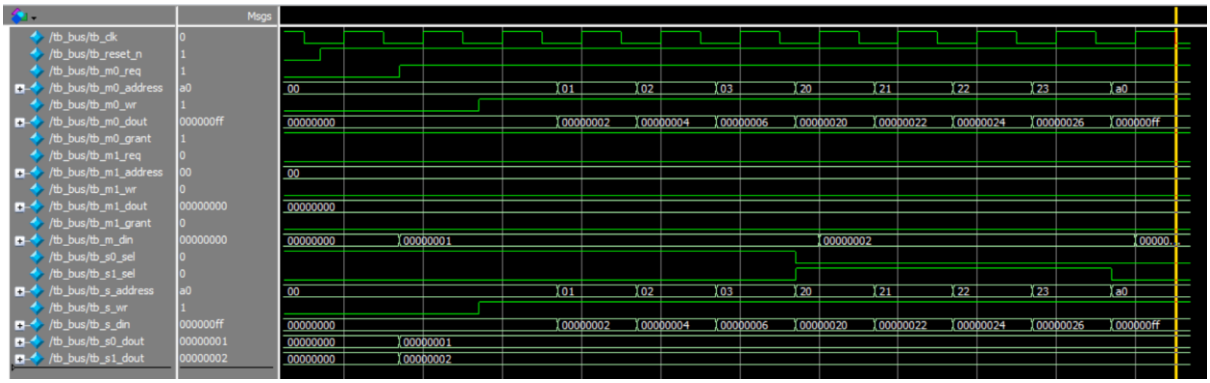
A. 시뮬레이션 결과

- Memory



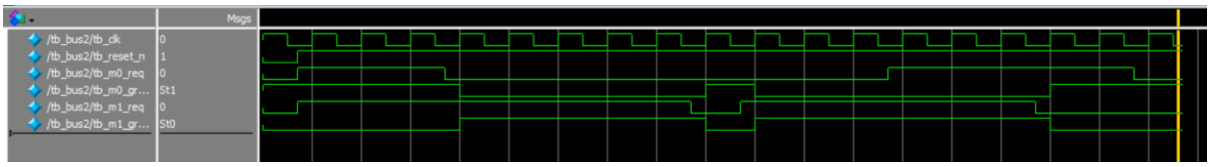
Cen이 1, wen이 1일 때 address가 가리키는 메모리 1, 2, 3, 4에 차례로 din, 1, 2, 3, 4가 write 되었다. Cen이 1이고 wen이 0일 때 address가 가리키는 메모리의 값인 1, 2, 3, 4가 dout에 write 되었다. Cen이 0일 때 dout이 0이 되었다. 따라서 결과가 잘 나온 것을 확인할 수 있다.

- Bus(testbench 1)



강의자료 ppt에 나온 testbench1 사진과 동일하며, Bus를 가진 master가 주는 address에 따라 sel 신호가 변경된다. 즉 tb_m0_address가 0x1F를 넘어가는 경우 s0_sel이 0, s1_sel이 1이 되는 것을 확인할 수 있으며 tb_m0_address가 slave0과 slave1의 범위를 넘어가는 경우 0이 출력되는 것을 확인할 수 있다.

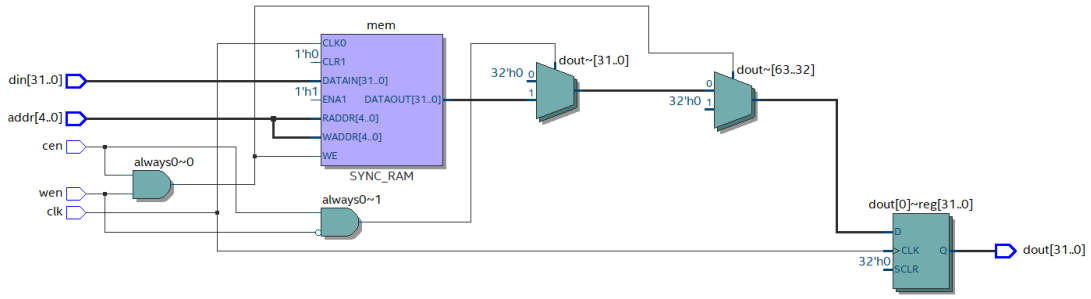
- Bus arbiter(testbench 2)



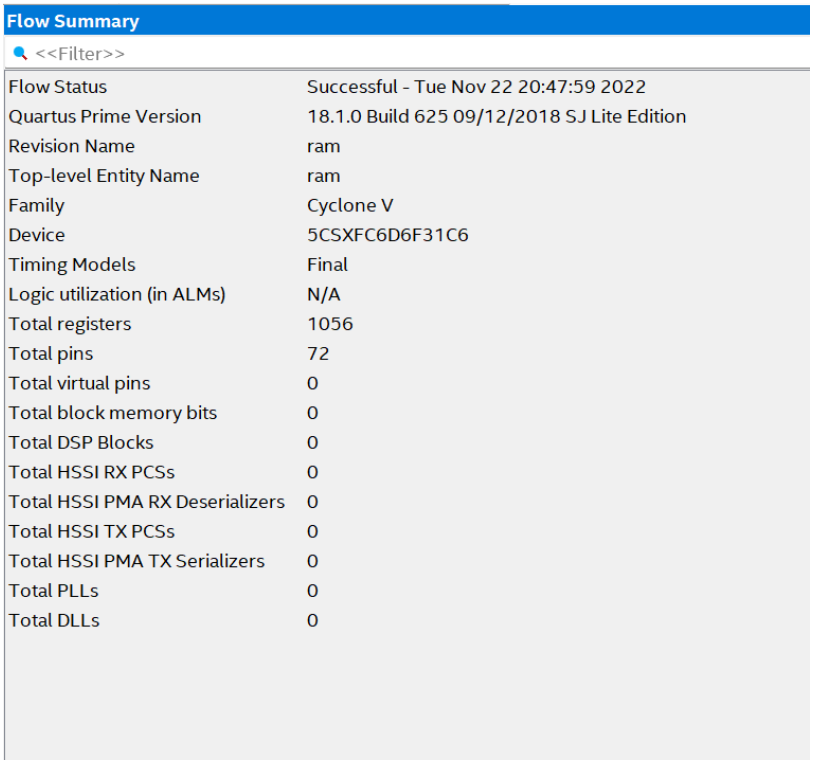
강의자료 ppt에 나온 testbench2 사진과 동일하며 tb_m0_req와 tb_m1_req의 값을 변화시켰다. Tb_m0_req가 1이면 tb_m1_req의 값과 상관없이 tb_m0_grant가 1이며, tb_m0_req가 0이고 tb_m1_req가 1이면 tb_m1_grant가 1이된다. 또한 tb_m0_req와 tb_m1_req가 0이면 tb_m0_grant가 1이 되는 것을 알 수 있다. 따라서 결과가 잘 나오는 것을 확인할 수 있다.

B. 합성(synthesis) 결과

- Memory



Memory, ram의 rtl viewer로, 회로가 잘 설계된 것을 확인할 수 있다.



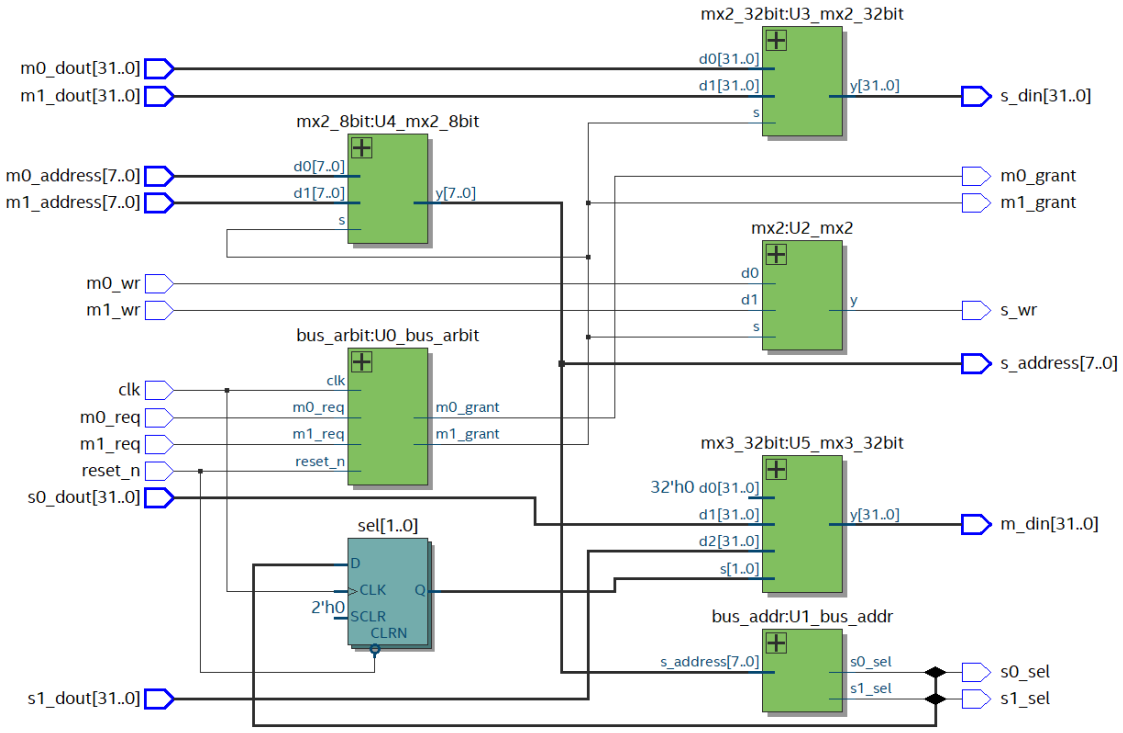
Flow Summary

<<Filter>>

Flow Status	Successful - Tue Nov 22 20:47:59 2022
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	ram
Top-level Entity Name	ram
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	1056
Total pins	72
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

Memory, ram의 flow summary이다. 총 1056개의 register와 72개의 pin을 사용한 것을 알 수 있으며 성공적으로 컴파일이 완료된 것을 알 수 있다.

- Bus



Bus의 rtl viewer로, arbit와 addr와 4개의 mux와 sel이 연결되어 있으며 회로가 잘 설계된 것 확인할 수 있다.

Flow Summary

<<Filter>>

Flow Status	Successful - Tue Nov 22 18:59:08 2022
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	bus
Top-level Entity Name	bus
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	3
Total pins	227
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

Bus의 flow summary이다. 총 3개의 register와 227개의 pin을 사용한 것을 알 수 있으며 성공적으로 컴파일이 완료된 것을 알 수 있다.

5. 고찰 및 결론

A. 고찰

이번 실습에서는 주어진 design과 강의자료를 보고 코드를 짜 난이도가 어렵지는 않았지만, 예를 들어 input에서 m0_req로 선언하고 M0_req로 사용하기, 스펠링 틀리기 등등의 사소하지만 중요한 실수들로 인해 오류를 수정하는데 더 많은 시간이 걸렸던 것 같다. 또한 testbench의 중요성을 다시 한 번 알았다. Rtl viewer와 flow summary에서는 문제가 없어 보였지만, testbench에 값을 넣어 돌려보니 이상한 값이 나오거나 빨간줄, 즉 제대로 연결이 되지 않는 결과가 나왔다. Testbench를 통해 잘못된 부분을 찾을 수 있어 전체 코드를 보지 않고 해결할 수 있었다.

B. 결론

Ram module을 구현하는 것은 시간이 오래 걸리지 않았지만, 메모리를 선언하고 초기화하는 것을 처음 해보았다. Initial 구문 안에서 for문을 사용하기 위해 integer i, 즉 변수 i를 선언해주었다. 메모리 선언 방식은 강의자료를 참고하였고 C언어에서의 for문과 비슷하게 코드를 구현하였다. Master와 Slave를 처음 개념으로는 이해가 잘 안됐지만, 코드를 짜보고 testbench에 값을 넣어 waveform을 확인하면서 직접 해보니 몰랐던 부분을 알 수 있었고, 도움이 많이 돼 프로젝트도 잘 진행할 수 있을 것 같다.

6. 참고문헌

공영호 교수님/디지털논리회로2 강의자료/광운대학교 컴퓨터 공학과 2022 강의자료
공영호 교수님/컴퓨터공학기초실험2/광운대학교/2022 강의자료