컴퓨터 공학 기초 실험2 보고서

실험제목: Simple Memory & Bus

실험일자: 2023년 11월 20일 (월)

제출일자: 2023년 11월 29일 (수)

학 과: 컴퓨터정보공학부

담당교수: 이준환 교수님

실습 분반: 월요일 0, 1, 2

학 번: 2020202031

성 명: 김재현

1. 제목 및 목적

A. 제목

Simple Memory & Bus

B. 목적

Memory는 address에 기반하여 데이터를 저장하는 hardware이며, Bus는 여러 component 들 간에 data를 전송(transfer)할 수 있도록 연결해주는 component입니다. Term Project에서 factorial computation system을 구현하는데 Bus와 Memory를 사용하기 위해 간단한 구성의 버스와 메모리를 구현해봅니다.

2. 원리(배경지식)

A. Bus

request 신호를 통해 Master의 명령을 전달할 것인지를 결정하고, Master에서 전달하는 주소 값의 Base Address를 통해 여러 Slave 중 하나를 선택합니다.

Bus는 컴퓨터 시스템 아키텍처에서 중요한 개념 중 하나로, 여러 컴포넌트 간에 데이터, 주소, 제어 신호 등을 전송하기 위한 통신 경로로 사용됩니다. Bus는 시스템 내에서 정보를 효과적으로 교환하고 각각의 하드웨어 구성 요소 간의 동작을 용이하게 해줍니다.

다양한 종류의 버스가 있으며, 이들은 주로 데이터, 주소, 제어 버스로 구성됩니다.

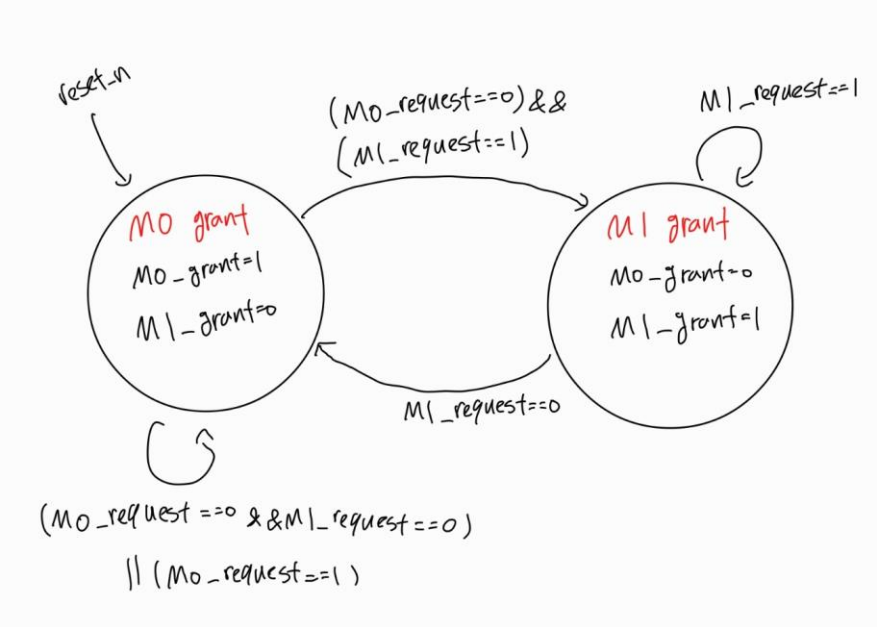
Data Bus: 시스템 모듈 간의 데이터 이동 경로를 제공해주며, 각 component 간에 데이터를 전송하는 데 사용됩니다.
Address Bus: CPU가 메모리나 입출력 장치에 특정 위치에 접근하기 위해 사용하는데, 주소 정보를 전송하는 데 활용되며, CPU가 어떤 메모리 주소로 데이터를 읽거나 쓸지를 결정하는데 관여합니다.
제어 버스 (Control Bus): 제어 버스는 각각의 구성 요소 간에 제어 신호를 전달하는 데 사용됩니다. 제어 버스는 시스템의 동작을 제어하기 위해 제어 신호를 전송하며, 이는 데이터의 읽기, 쓰기, 버스의 활성화 또는 비활성화 등을 포함합니다.

B. Memory(RAM)

RAM은 주로 컴퓨터가 현재 작업 중인 데이터 및 프로그램을 일시적으로 저장하는 데 사용됩니다. RAM은 휘발성 메모리로, 전원이 꺼지면 저장된 데이터가 소멸하는데, 주로

컴퓨터의 작업 메모리로 사용되어 빠른 읽기 및 쓰기 속도가 필요한 임시 데이터를 보관하는데 사용됩니다.

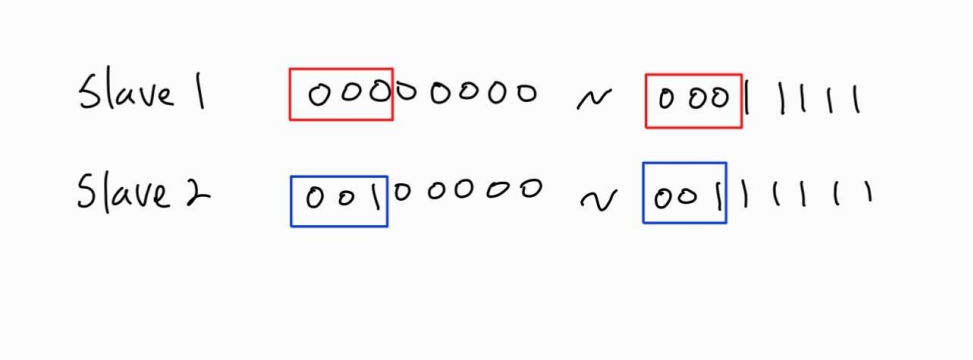
3. 설계 세부사항



output / state	M0 grant	M1 grant
M0_grant	1	0
M1_grant	0	1

bus_arbiter의 state diagram과 truth table입니다.

초기에 reset_n 신호를 받으면 state가 M0가 됩니다. 현재 state는 본인의 request 신호가 ON이면 계속해서 자신의 state를 유지합니다. 현재 state의 request 신호가 0이고 다른 state의 request가 1이 되면 state가 바뀌게 됩니다. 그러나 request 신호가 둘 다 0이 되면 M0가 우선순위를 가지게 되어 state가 무조건적으로 M0가 됩니다.

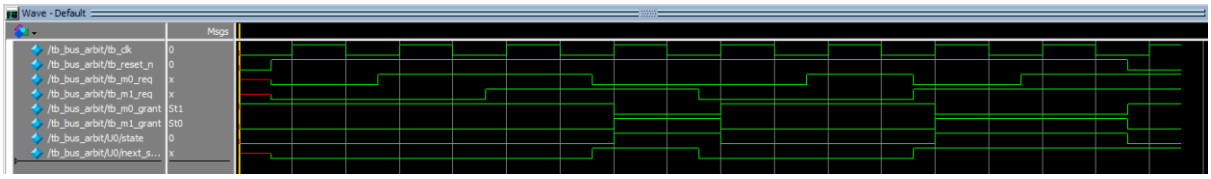


각 slave의 memory map입니다. slave1은 상위 3비트가 항상 000이고, slave2는 상위 3비트가 항상 001입니다. 따라서 master에서 부여하는 주소 값의 상위 3비트를 분석하면 어

떤 slave에 연결하려고 하는지 알 수 있습니다. 따라서 bus Address decorder는 주소 값의 상위 3비트를 분석하여 각 slave에 대한 select 값을 출력합니다.

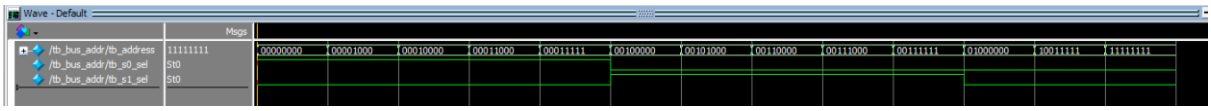
4. 설계 검증 및 실험 결과

A. 시뮬레이션 결과



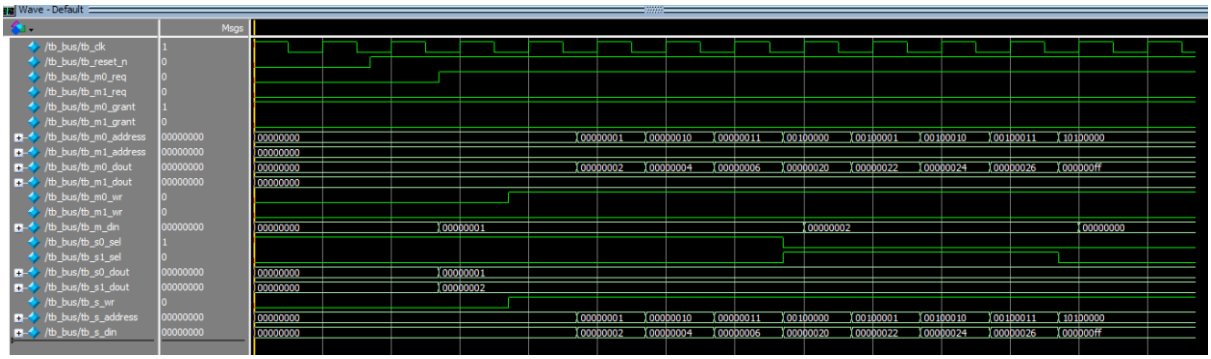
tb_bus_arbit의 testbench입니다.

m0_grant state일 때는 0, m1_grant state일 때는 1입니다. reset_n신호를 받으면 state가 m0가 됨을 확인할 수 있습니다. m0 req, m1 req가 둘 다 0이므로 계속해서 state가 m0를 유지하는 것을 확인할 수 있습니다. state가 m0인 상황에서 req 신호가 둘 다 1이 될 경우 계속해서 state m0를 유지합니다. m0 req가 0, m1 req가 1이 되는 순간 state가 m1으로 넘어갑니다. 그러나 req 신호가 둘 다 0이 될 경우 우선 순위가 있는 m0로 state가 넘어감을 확인할 수 있습니다.



tb_bus_addr의 testbench입니다.

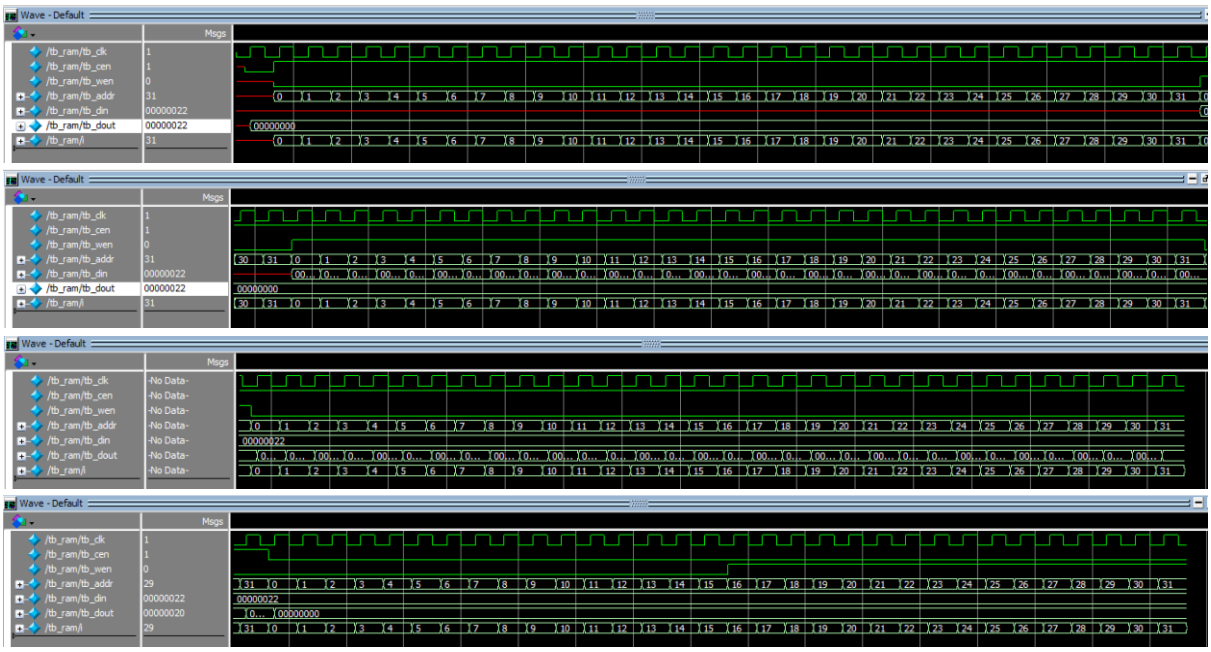
tb_address의 상위 3비트가 000이면 slave0가 select되고, 001이면 slave1이 select되는 것을 확인할 수 있습니다.



tb_bus의 testbench입니다.

bus_arbit를 통해 권한을 부여 받는 master가 정해지고, bus_addr를 통해 통제 받을 slave가 정해집니다. 그리고 이 권한을 가지는 master와 통제 받는 slave가 바뀌는 것은 한 사이클 단위로 행해집니다. 따라서 한 사이클 동안 m_address, m_wr, m_dout을 통해

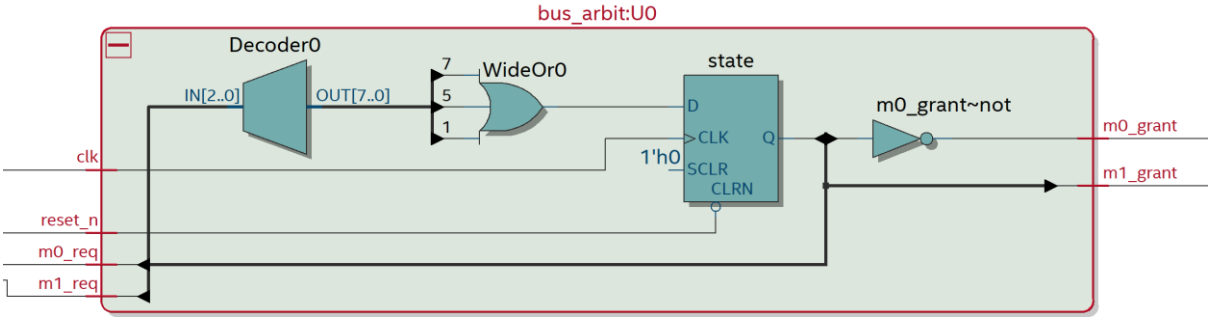
s_address, s_wr, s_din이 정해지는 동작과 s_dout을 통해 m_din이 정해지는 동작은 asynchronous하게 작동하는 것을 확인할 수 있습니다.



tb_ram의 testbench입니다.

초기 ram의 값은 initial문을 통해 모든 reg값을 0으로 초기화해줬기 때문에 cen 1, wen 0을 입력하여 read 동작을 시행한 결과 dout을 통해 0이 출력되는 것을 확인할 수 있습니다. 그 후 cen 1, wen 1을 입력하여 write 동작을 통해 addr 0~31까지의 메모리에 값을 입력하고 cen 1, wen 0을 입력하여 다시 read 동작을 실행한 결과, write했던 데이터 값들이 출력되는 것을 확인할 수 있습니다. cen이 0일 경우엔 wen값과 상관 없이 dout에 0이 출력되는 것을 확인할 수 있습니다.

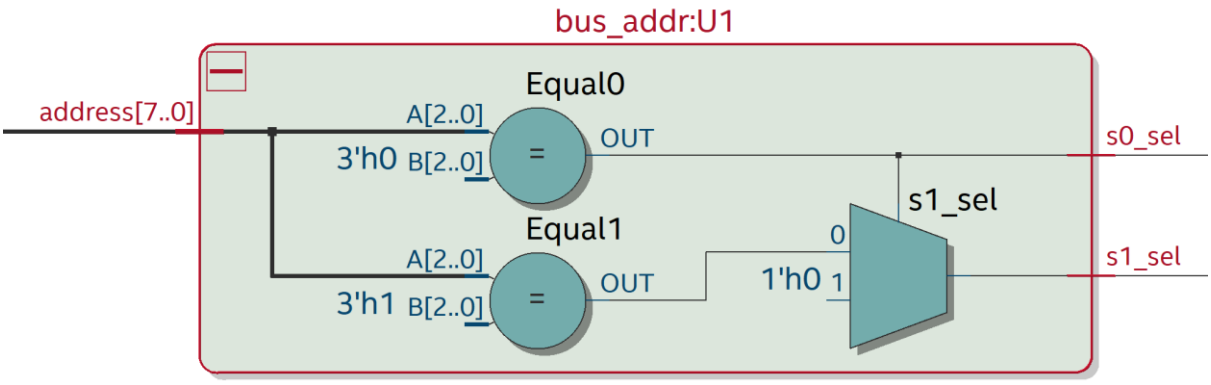
B. 합성(synthesis) 결과



bus_arbit의 RTL Viewer입니다.

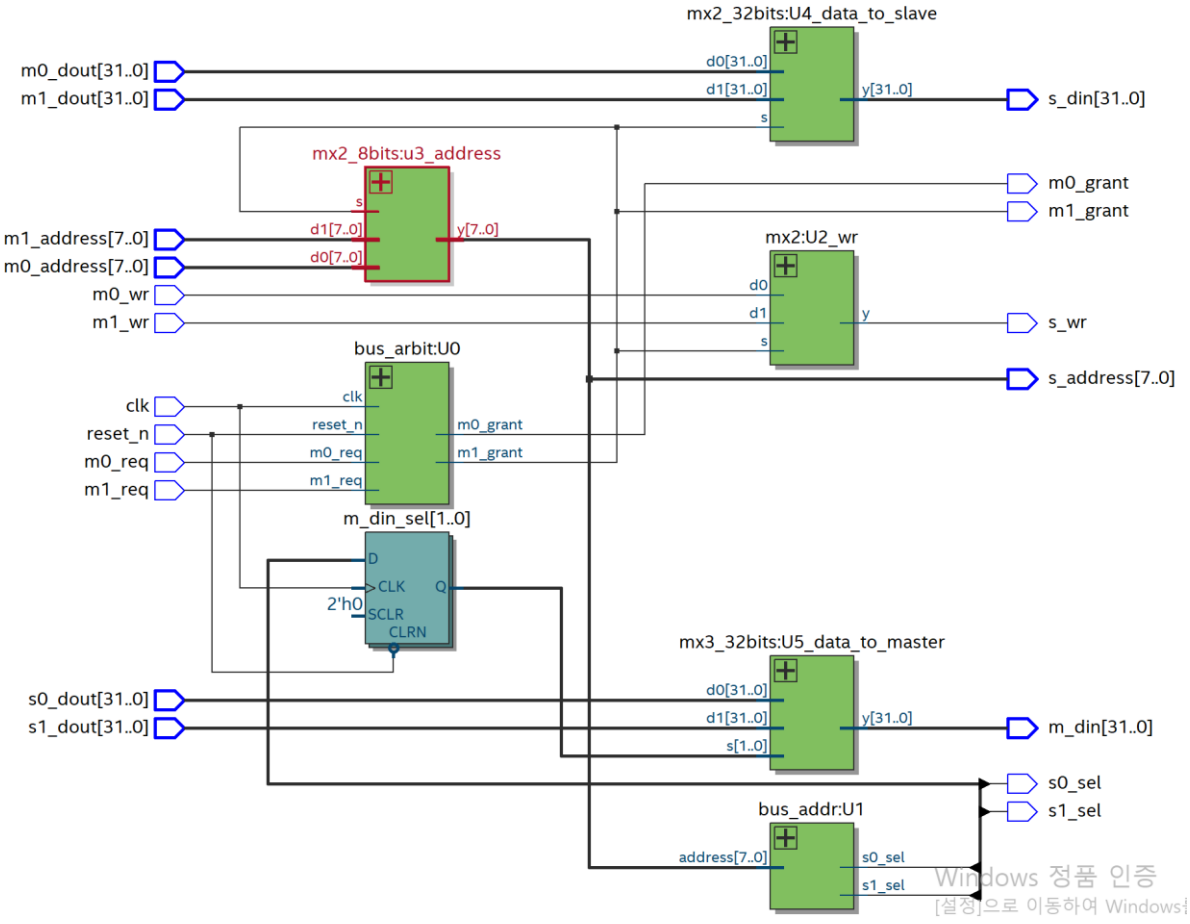
현재 state와 m0_req, m1_req를 input으로 3to8 decorder를 통해 next_state를 정합니다. state M0_grant는 0, M1_grant는 1이므로 출력 값 m0_grant는 현재 state의 not gate를

거친 값입니다.



bus_addr 의 RTL Viewer입니다.

address 상위 3bits가 000이면 s0_sel에 1을 출력합니다. s0_sel이 1이 되면 s1_sel은 무조건 0을 출력합니다. s0_sel이 0이고, (address 상위 3bits==3'b001)의 논리 값을 출력합니다.



bus의 RTL Viewer입니다.

clk의 posedge에서 bus_addr와 bus_arbit를 통해 사용할 slave와 master를 결정해주고, mux를 통해 연결된 master, slave 간에 데이터 전송 및 수신을 할 수 있도록 구현해준 것

을 확인할 수 있습니다.

Flow Summary

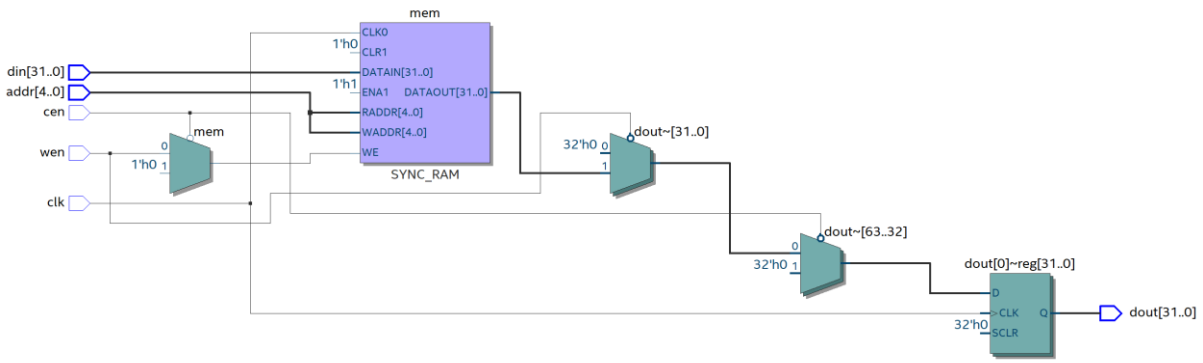
<<Filter>>

Flow Status	Successful - Wed Nov 29 14:44:46 2023
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)	39 / 41,910 (< 1 %)
Total registers	3
Total pins	227 / 499 (45 %)
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 %)

bus의 flow summary입니다.

logic utilization은 39입니다.

total pins는 clk 1bit + reset_n 1bit + m0_req 1bit + m1_req 1bit + m0_wr 1bit + m1_wr 1bit + m0_address 8bits + m1_address 8bits + m0_dout 32bits + m1_dout 32bits + s0_dout 32bits + s1_dout 32bits + m0_grant 1bit + m1_grant 1bit + m_din 32bits + s0_sel 1bit + s1_sel 1bit + s_wr 1bit + s_address 8bits + s_din 32bits = 227 입니다.



ram의 RTL Viewer입니다.

마지막 mux에서 cen이 0일 때의 경우를 1차적으로 걸러줌과 동시에 wen이 mem의 we의 입력으로 들어가는 것을 막아, mem에 어떤 값도 write되지 않게 합니다. 이때, dout에 0을 출력합니다.

cen이 1일 때는 wen이 0일 때와 1일 때의 경우로 나뉘는데, wen이 0일 때는 mem에 저장된 값을 읽어 dout에 출력해줍니다. wen이 1일 때는 dout에 0을 출력해주고, mem의 we에 1을 입력하여 din값이 mem의 올바른 주소 값에 저장되도록 합니다.

Flow Summary

<<Filter>>

Flow Status	Successful - Wed Nov 29 19:17:56 2023
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	5CSXFC6C6U23C6
Timing Models	Final
Logic utilization (in ALMs)	521 / 41,910 (1 %)
Total registers	1056
Total pins	72 / 342 (21 %)
Total virtual pins	0
Total block memory bits	0 / 5,662,720 (0 %)
Total DSP Blocks	0 / 112 (0 %)
Total HSSI RX PCSs	0 / 6 (0 %)
Total HSSI PMA RX Deserializers	0 / 6 (0 %)
Total HSSI TX PCSs	0 / 6 (0 %)
Total HSSI PMA TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 12 (0 %)
Total DLLs	0 / 4 (0 %)

bus의 flow summary입니다.

logic utilization은 521입니다.

total pins는 clk 1bit + cen 1bit + wen 1bit + addr 5bits + din 32bits + dout 32bits = 72입니다.

5. 고찰 및 결론

A. 고찰

머리 속으로 생각하고 생각한 것을 그대로 구현하는데, 생각보다 자잘한 실수들이 많이 나오는 것을 경험했습니다. 또한 case문으로 쓰이는 단자들은 always문에도 포함시키지 않으면 오류가 날 확률이 높아진다는 것을 알게 됐습니다.

B. 결론

이번 과제에서 bus의 주어진 design을 보고 그대로 구현하는 실습을 진행하면서 그 때 그 때 떠오르는 것을 구현하기 보다, 전체적인 틀의 design을 그리고 그것을 토대로 구현하는 것이 훨씬 효율적이고 실수도 적다는 것을 깨달았습니다.

6. 참고문헌

이준환 교수님/디지털논리회로2/광운대학교(컴퓨터정보공학부)/2023

이준환 교수님/컴퓨터공학기초실험2/광운대학교(컴퓨터정보공학부)/2023