컴퓨터 공학 기초 실험2 보고서

실험제목: Latch & flip-flop design with/without reset/set

실험일자: 2022년 10월 12일 (수)

제출일자: 2022년 10월 18일 (화)

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

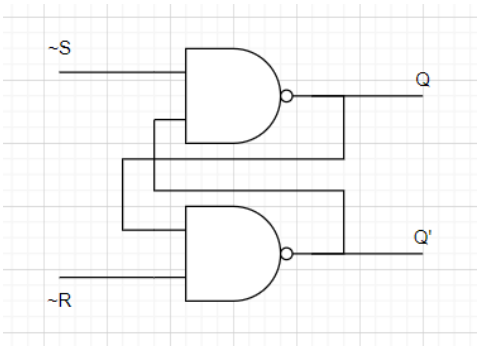
Latch & flip-flop design with/without reset/set

B. 목적

이전의 입력값을 유지/저장할 수 있는 Latch와 flipflop의 원리를 이해하고 설계한다. D-flipflop을 직접 설계하고 이와 더불어 reset과 set의 기능을 구현한다. synchronous와 Asynchronous set/resettable의 차이를 알아보고, 직접 설계한 D-flipflop을 이용하여 N-bit register을 구현해본다.

2. 원리(배경지식)

A. RS Latch

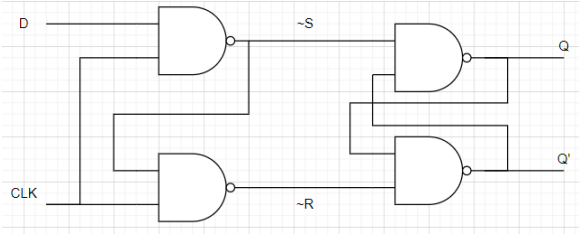


~SET	~RESET	Q
0	0	X
0	1	1
1	0	0
1	1	Qprev

RS-Latch는 값을 저장할 수 있는 기능이 있으며 nand gate 두개를 사용하여 구현할 수 있다. 또한 nand gate가 아닌 nor gate로도 구현이 가능하다. 이 때 nand gate를 이용하여 SR latch를 구현하면 입력 값이 반대로 된다.

위의 진리표르 보면 ~set과 ~reset 모두 0인 경우 오류가 발생하지만 ~set이 0이고 ~reset이 1인 경우 Q는 1을 출력하고, ~reset이 0이고 ~set이 1이면 Q는 0을 출력한다. 이 때 둘 다 1이라면 Q는 이전의 Q값을 그대로 불러온다. 이를 보고 저장되었다고 볼 수 있다.

B. Nand gate 4개를 이용한 D-Latch

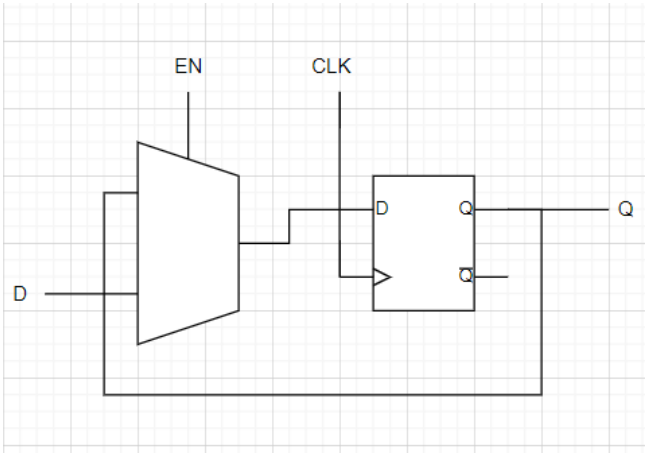


D	CLK	Q
0	0	Qprev
0	1	0
1	0	Qprev
1	1	1

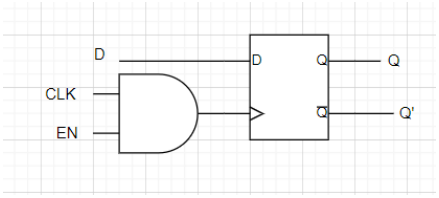
앞서 다뤘던 SR Latch에 nand 게이트 두개를 더 연결하여 nand 4개로 구성된

D-Latch를 구현할 수 있다. SR Latch 인 경우 둘 다 0일 때는 오류가 발생했지만 D-Latch는 왼쪽의 두개의 nand 게이트가 SR Latch에 0,0의 값을 입력되지 않도록 하기 때문에 오류 상태가 발생하지 않는다. 이 때 clk가 1일 때만 D의 값을 반환한다.

C. enabled D flip flop의 다른 방법으로의 구현



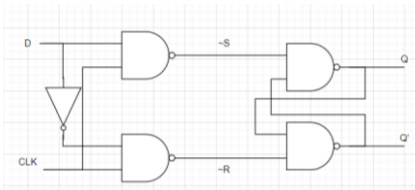
과제에서 주어진 Enabled D flip-flop의 회로도는 위와 같다. 위처럼 mux를 활용하여 D의 값을 선택하고 이를 D Flip-flop에 넘기는 것을 볼 수 있다. 이와 다른 방법으로는 D값은 그대로 D flip-flop에 넣고 clk의 값을 조절하는 것이다.



위 그림은 D의 값의 조절이 아닌 CLK 값을 조절하기 위해 AND gate를 사용하여 CLK와 en 모두 1일 때만 1이 CLK에 입력되도록 조절하였다.

3. 설계 세부사항

A. D-Latch

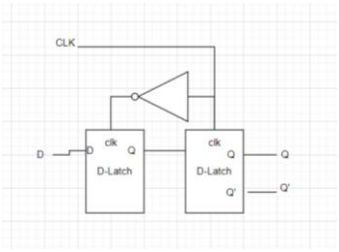


D	CLK	Q
0	0	Qprev
0	1	0
1	0	Qprev
1	1	1

이번 실험에서는 위의 원리(배경지식)에서 소개한 nand 게이트 4개만 사용해서 구현하는 D-Latch가 아니라 Inverter도 사용하여 만드는 D-Latch를 소개하고 있고 이를 바탕으로 실험을 구현하였다.

구분	이름	비트 수(bit)
Input	d	1
	clk	1
output	q	1
	q_bar	1
wire	n1,n2,d_inv	1

B. D-flipflop

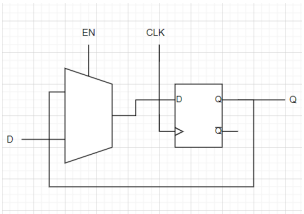


CLK	Q
↑	D
↓	Qprev

이번 실험에서 제작한 D-Latch 두개와 Inverter 1개를 연결하여 D-flipflop을 제작한다. 이 D-flipflop은 CLK가 rising edge 일 때만 D의 값을 반환하고 아닐 때는 이전의 Q값을 반환한다.

구분	이름	비트 수(bit)
Input	d	1
	clk	1
output	q	1
	q_bar	1
wire	clk_inv, w1	1

C. Enabled D-flipflop

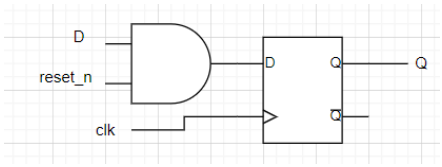


D	EN	CLK	Q
X	0	X	Qprev
X	1	↓ or →	Qprev
0	1	↑	0
1	1	↑	1

기존의 D-flipflop의 2 to 1 mux를 연결하여 en이 1일 때만 D의 값을 D-flipflop으로 넘길 수 있도록 설계된 것이다. 또한 clk가 rising edge 일 때만 D의 값이 Q의 값으로 반환되는 것을 볼 수 있다.

구분	이름	비트 수(bit)
Input	d	1
	clk	1
	en	1
output	q	1
wire	w1	1

D. Resettable D-flipflop



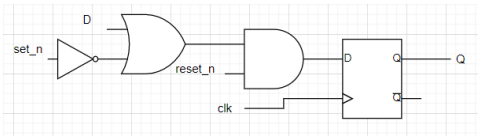
D	reset_n	CLK	Q
x	0	↑	0
x	0	↓ or →	Qprev
0	1	↑	0
1	1	↑	1

reset_negative의 값에 따라 달라지는 D-flipflop으로 reset_n의 값이 0일 때 clk가 rising edge이면 무조건 q는 0이고 clk가 rising edge가 아니면 Q의 이전값을 그대로 가져온다.

만약 reset_n이 1이고 clk가 rising edge이면 D의 값을 Q의 값으로 반환한다.

구분	이름	비트 수(bit)
Input	d	1
	clk	1
	reset_n	1
output	q	1
wire	w1	1

E. Synchronous set/resettable D-flipflop

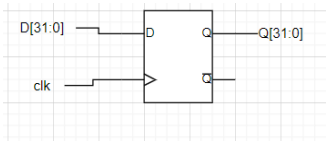


D	set_n	reset_n	clk	Q
X	0	1	↑	0
X	1	0	↑	1
0	1	1	↑	0
1	1	1	↑	1
x	1	1	→ or ↓	Qprev
x	0	0	↑	0

set_n과 reset_n의 값에 따라 Q의 값이 달라지는 D-flipflop으로 And, or, inv gate를 이용하여 제작할 수 있다. 둘 다 1이고 clk가 rising edge라면 D의 값을 반환하고 clk가 rising edge가 아니라면 Q의 이전값을 가져온다. set_n이 0이고 reset_n이 1이면 clk가 rising edge일 때 0을 반환하고, 반대라면 1을 반환한다. 둘 다 0일 때는 rising edge에서 0을 반환한다.

구분	이름	비트 수(bit)
Input	d	1
	clk	1
	reset_n	1
	set_n	1
output	q	1
wire	w1	1

F. 32bit register

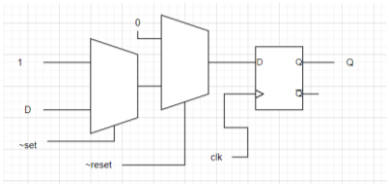


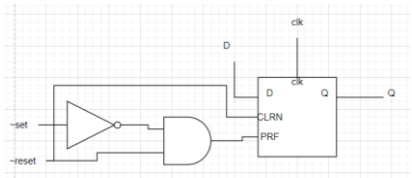
CLK	Q
↑	D
→ or ↓	Qprev

D-flipflop을 32개를 써서 나타낸 것이 32bit register이다 clk가 rising edge일 때 D값을 반환하고 아니면 Q의 이전 값을 가져온다.

구분	이름	비트 수(bit)
Input	d	32
	clk	1
output	q	32

G. Async/Sync Set/resettable D-flipflop





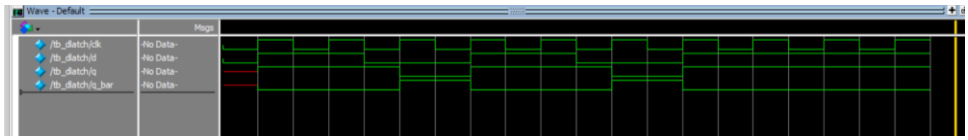
Sync Set/Resettable Async Set/Resettable

구분	이름	설명
Top module	_dff_rs_sync_async	sync Set/Resettable과 async Set/Resttable을 합친 모듈
Sub module	_dff_rs_sync	동기 reset, set D-flipflop
	_dff_rs_async	비동기 reset,set D-flipflop

4. 설계 검증 및 실험 결과

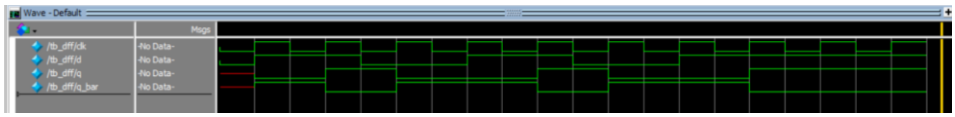
A. 시뮬레이션 결과

i. D-Latch



D-latch에서는 clk가 1일 때 D의 값을 Q로 반환한다.

ii. D-Flipflop



D-flipflop에서는 clk가 rising edge일 때 D의 값을 반환한다.

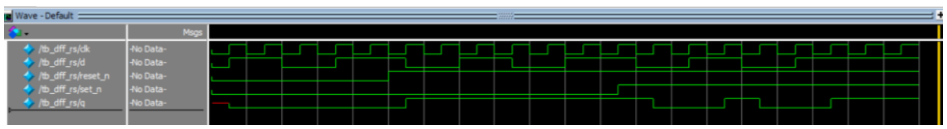
iii. Enabled D-Flip-Flop

En이 0일 때는 변화가 없으며 1일 때 clk가 rising edge일 때 d의 값을 반환한다.

iv. Resettable D-Flip-Flop

reset_n이 0일 때는 clk에 동기화 되어 0으로 reset 되고 1일 때는 clk가 rising edge일 때 d의 값을 반환한다.

v. Synchronous Set/Resettable D-Flip-Flop



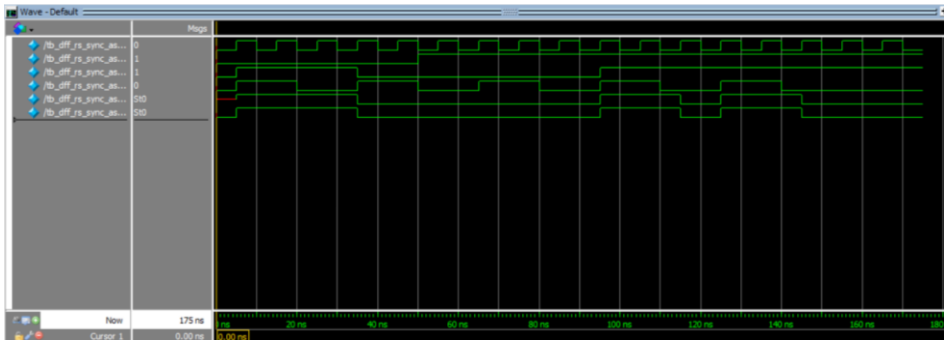
set_n이 0 ,reset_n이 1일 때 clk에 동기화 되어 Q가 1이 되고 둘다 1이면 clk가

rising edge일 때 d의 값을 반환하는 것을 확인할 수 있다.

vi. 32bit Register

clk가 rising edge 일때 32bit D의 값을 32bit Q의 값으로 반환한다.

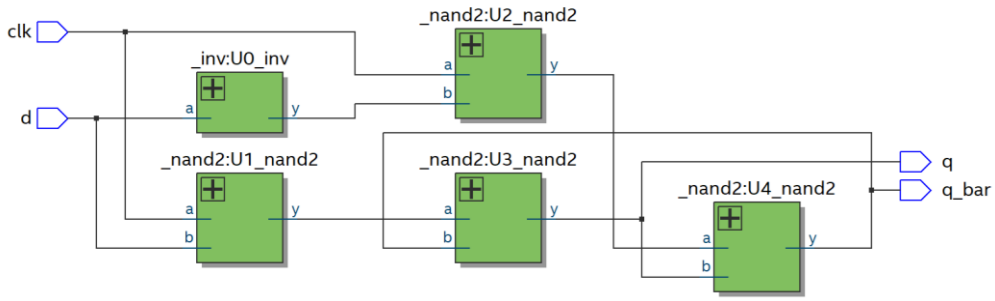
vii. Async/Sync Set/Resettable D-Flip-Flop



Async와 sync의 출력 q가 달라지는 것을 확인할 수 있다.

B. 합성(synthesis) 결과

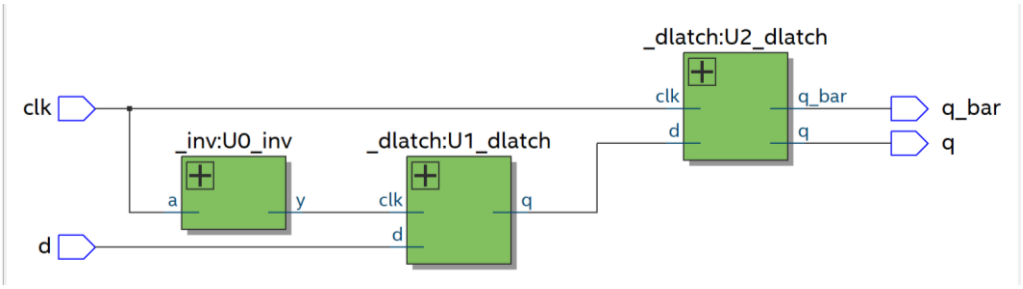
i. D-Latch



Flow Summary	
<<Filter>>	
Flow Status	Successful - Tue Oct 18 15:33:21 2022
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	latch
Top-level Entity Name	_dlatch
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	2 / 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 %)

4개의 nand 그리고 1개의 inverter를 사용하여 구현하였다.

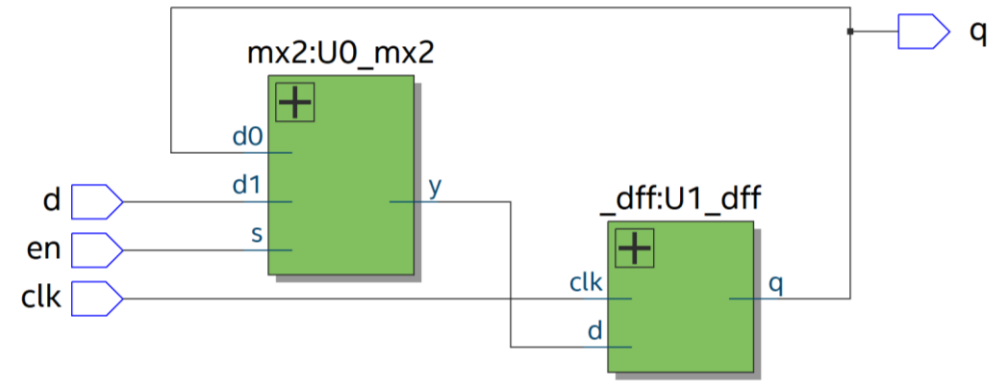
ii. D-Flipflop



Flow Summary	
<<Filter>>	
Flow Status	Successful - Tue Oct 18 16:38:01 2022
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	latch
Top-level Entity Name	_dff
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	2 / 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 %)

구현한 d-latch 2개와 inverter 1개를 이용해 구현했다.

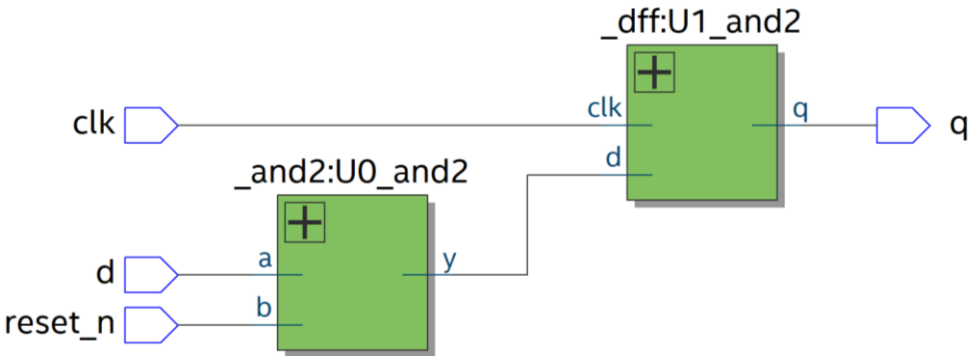
iii. Enabled D-Flip-Flop

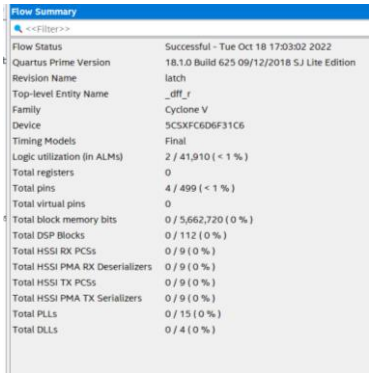


Flow Summary	
Flow Status	Successful - Tue Oct 18 16:50:38 2022
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	latch
Top-level Entity Name	_dff_en
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	2 / 41,910 (< 1 %)
Total registers	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 %)

이전에 구현해본 2 to 1 MUX를 이용해 en 값에 따른 d값의 전달 유무를 정했다.

iv. Resettable D-Flip-Flop

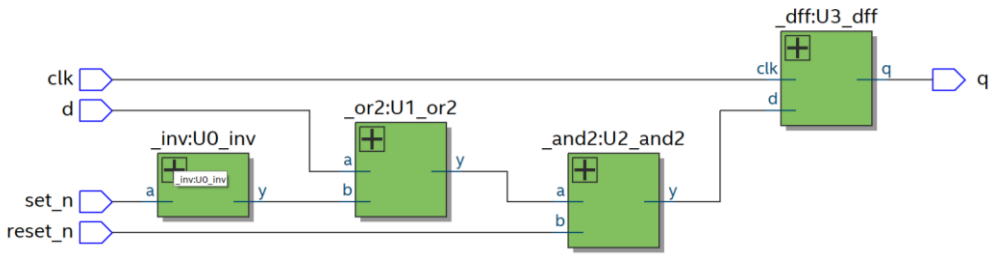


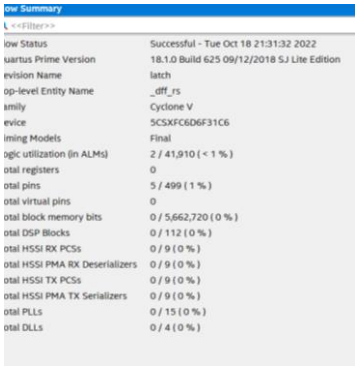


Flow Summary	
Flow Status	Successful - Tue Oct 18 17:03:02 2022
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	latch
Top-level Entity Name	_dff_r
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	2 / 41,910 (< 1 %)
Total registers	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 %)

And gate를 이용하여 reset_n이 0일 때 D의 값이 무조건 0이 되도록 하여 Reset 되도록 하였다.

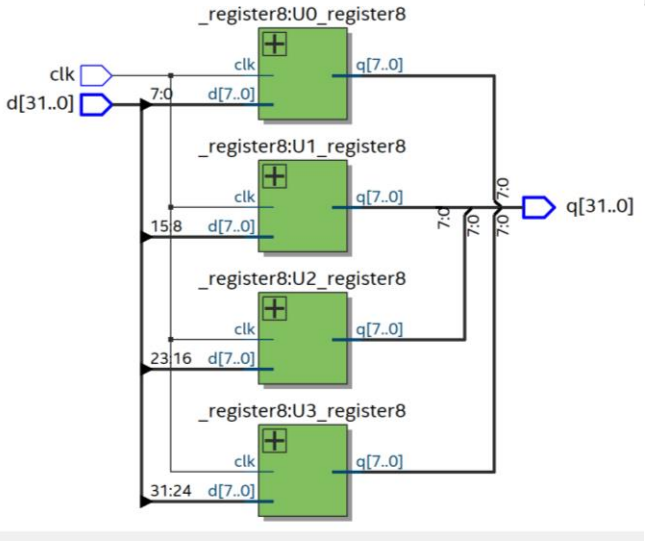
v. Synchronous Set/Resettable D-Flip-Flop

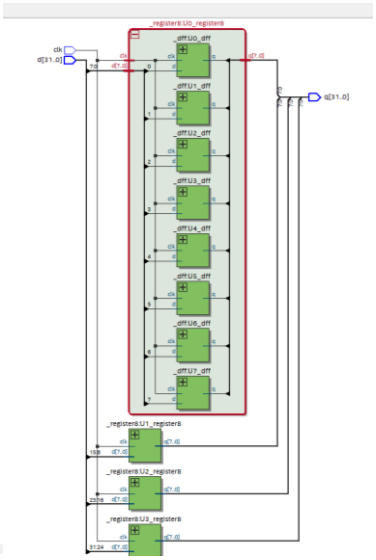




inv, or, and gate를 이용하여 set_n이 0이면 D에 무조건 1이 되도록하여 Set이 되고 reset_n이 0이면 D가 0이 되도록 했다.

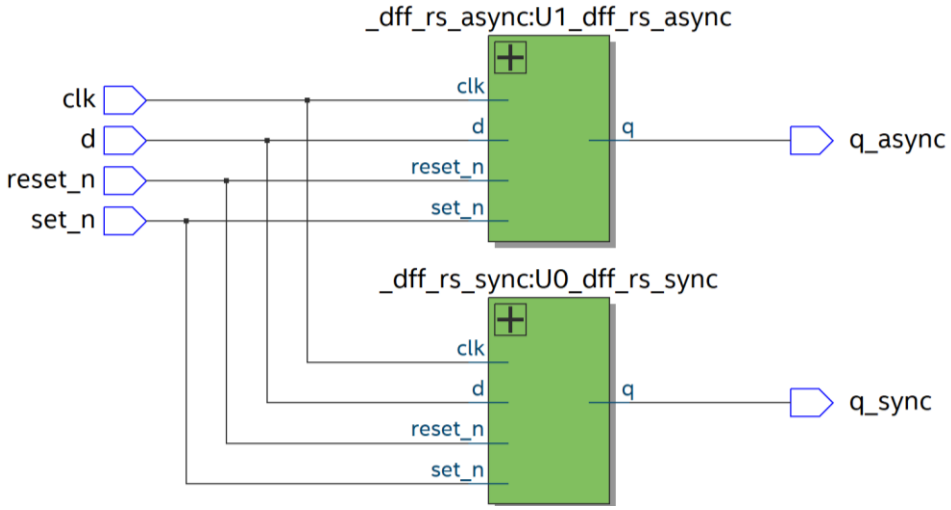
vi. 32bit Register

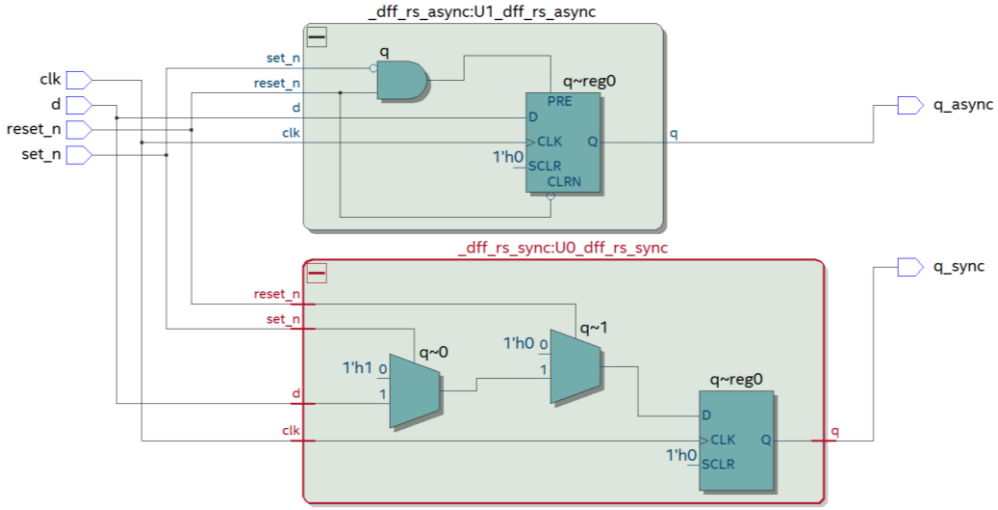




8개의 D-flipflop을 이용해 8bit register을 먼저 만들고 이 register을 4개를 사용해 32bit register을 구현하였다.

vii. Async/Sync Set/Resettable D-Flip-Flop





Flow Summary	
Flow Status	Successful - Tue Oct 18 17:34:09 2022
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	latch
Top-level Entity Name	_dff_rs_sync_async
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	3 / 41,910 (< 1 %)
Total registers	2
Total pins	6 / 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 %)

clk에 동기화 되어 Reset/set이 동작하도록 하며 Sync 반대로 누르면 동장하는 Async 를 always 문을 통해 구현하다.

5. 고찰 및 결론

A. 고찰

이번 실험에서 D-latch를 기반으로 flipflop의 다양한 종류를 구현하였으며, sync 그리고 Async set/resettable flipflop을 구현하였다. testbench를 7개를 구현해야 하고 이때까지

사용한 gate들을 다시 사용하여 구현을 완성하였다.

D-flip/flop을 synchronous reset and set과 asynchronous reset and set을 gate로 구현하는 것이 아닌 always문을 활용하여 각 posedge와 engedge에 반응하도록 하였으며 sync의 경우 posedge일 때 set/reset 동기화가 되며 0일 때 작동한다. 따라서 reset/set이 바뀌어도 다음 posedge일 때 넘어가는 것을 볼 수 있다. 그리고 async에서는 clk에 동기화되지 않으며 engedge일 때 d-flipflop의 preset과 clear를 통하여 바로 reset/set이 적용되는 것을 확인할 수 있다.

B. 결론

이번 실험을 통해 clk에 따른 동기 그리고 비동기에 대한 개념을 익히고 이를 구현하는 것을 gate 단이 아닌 always를 통해 구현할 수 있었다. 동기 같은 경우 기존 회로의 입력단에서 MUX 등을 이용하여 조정하면 되고 clk에 의존하여 즉각적 반응을 이끌어 내기 힘들었다면 비동기는 입력단이 아닌 기존 회로의 설계를 수정하여 동기보다는 코딩이 복잡하지만 즉시 반응한다는 차이점이 있다.

마지막으로 d-flipflop을 32개를 사용하여 32bit register을 구현하고 clk가 rising edge 일 때만 D의 값을 Q로 반환하도록 만들었다.

6. 참고문헌

1) 공진흥 교수님/디지털논리회로1/광운대학교 컴퓨터정보공학부/2022
2) 공진흥 교수님/컴퓨터공학기초실험2/광운대학교 컴퓨터정보공학부/2022
3) 공영호 교수님/디지털논리회로2/광운대학교 컴퓨터정보공학부/2022