

컴퓨터 공학 기초 실험2 보고서

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

실험일자: 2023년 10월 09일 (월)

제출일자: 2023년 10월 18일 (수)

학 과: 컴퓨터공학과

담당교수: 이준환 교수님

실습분반: 월요일 0, 1, 2

학 번: 20222020264

성 명: 최봉규

1. 제목 및 목적

A. 제목

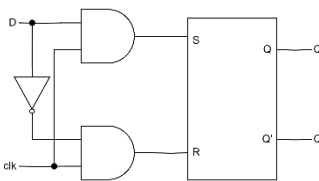
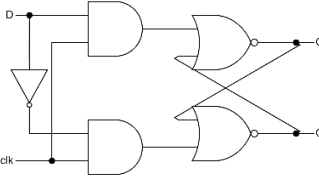
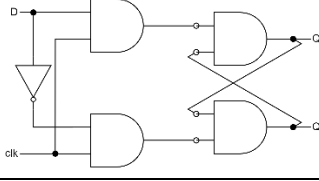
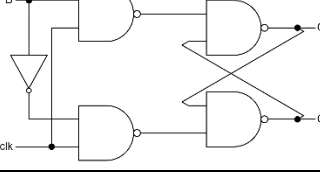
Latch & flip-flop design with/without reset/set

B. 목적

이번 실습에서는 이전에 값을 유지하고 있는 저장 소자 역할을 하는 latch와 flip-flop을 설계하고, 더불어 reset과 set기능을 구현한다. 또한, 구현한 flip-flop을 사용하여 N-bits register를 구현하는 방법에 대하여 살펴보도록 한다. D FF with active-low synchronous reset and set 과 D FF with active-low asynchronous reset and set의 차이를 설명한다.

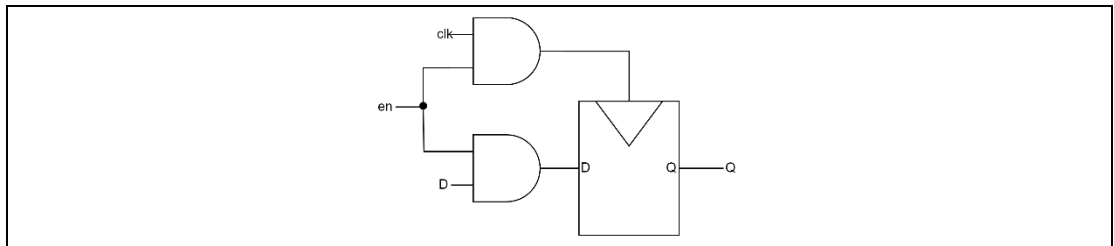
2. 원리(배경지식)

D latch는 clock이 enable 상태를 유지하는 동안 입력 D값의 변화를 출력한다. Nand gate 4개를 이용하여 D latch를 구현하는 방법은 다음과 같다. D Latch는 clk가 0일 때, Q는 prev 값을 갖는다. 만약 clk가 1이라면 Q는 D의 값을 갖는다. D Latch는 다음과 같은 단계를 거치며 2input nand gate 4개를 이용해서 만들 수 있다.

	초기 상태의 D-Latch schematic
	기본 RS Latch를 보면 nor을 이용해서 Q와 Q'가 연결되는 것을 볼 수 있음.
	이제 bubble pushing을 이용해서 and gate만 남긴다.
	각 버블이 gate 앞으로 보내면 4개의 nand gate로 구현할 수 있다.

현재 구상하는 enabled flip flop은 다음과 같다. Flip flop의 경우에는 D Latch 2개를 이용

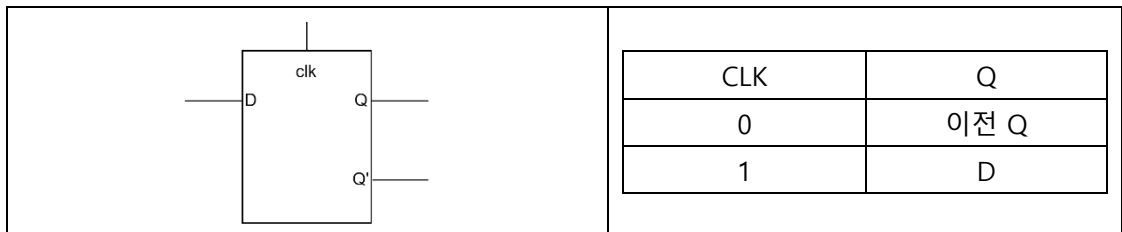
해서 만들 수 있다. 현재 truth table을 보았을 때 Q가 D의 값으로 업데이트 되는 것은 clk가 rising 될 때이며, 이 때의 en값은 반드시 1인 경우이다. En과 D가 and gate를 통하면 enabled flip flop이 가능하다고 생각한다. Clk이 1의 상태에서 계속 있을 때에는 flip flop이 업데이트를 하지 않을 것이다. En의 값이 1인 상태일 때에만 D의 값이 flip flop으로 전달된다. 나머지는 clk에 의해 q가 결정되는 것이다. 하지만, 이에도 단점이 있다. En이 0인 경우에는 flip flop이 이전 q 값을 가져야 하는데, 무조건 0의 값을 가지기 때문이다. 그렇기에 clk 부분 역시 en값과 and gate를 통해 가는 방법을 택했다. 아래는 mux 부분을 변경한 enabled D Flip-Flop의 symbol이다.



3. 설계 세부사항

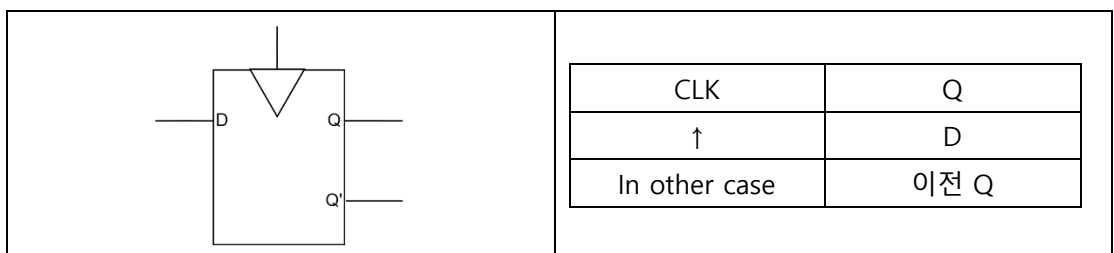
1) D Latch

다음은 D latch의 symbol과 truth table이다.

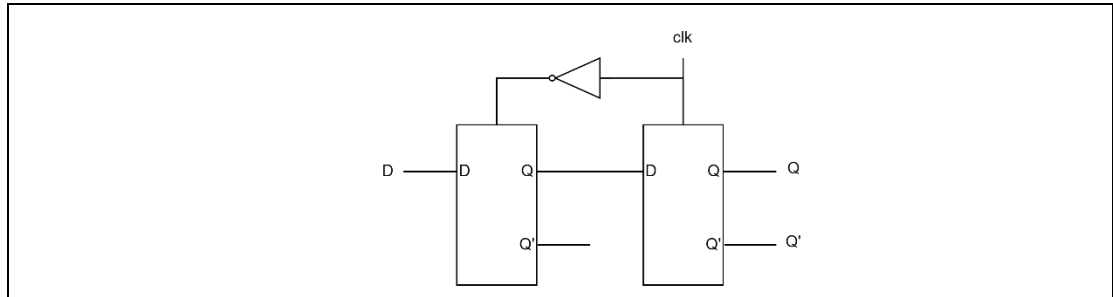


2) D Flip-Flop

D flip flop은 clock의 rising edge나 falling edge 에서만 D 값으로 출력이 바뀌게 된다. 다른 경우에는 D 값이 바뀌더라도 이전 Q값을 그대로 유지한다. 해당 실습에서는 clock의 rising edge를 사용한다. 다음은 D flip-flop의 symbol과 truth table이다.



다음은 structural Description으로 D-latch 두 개를 이용해서 만드는 d Flip flop이다.



3) Enabled D Flip-Flop

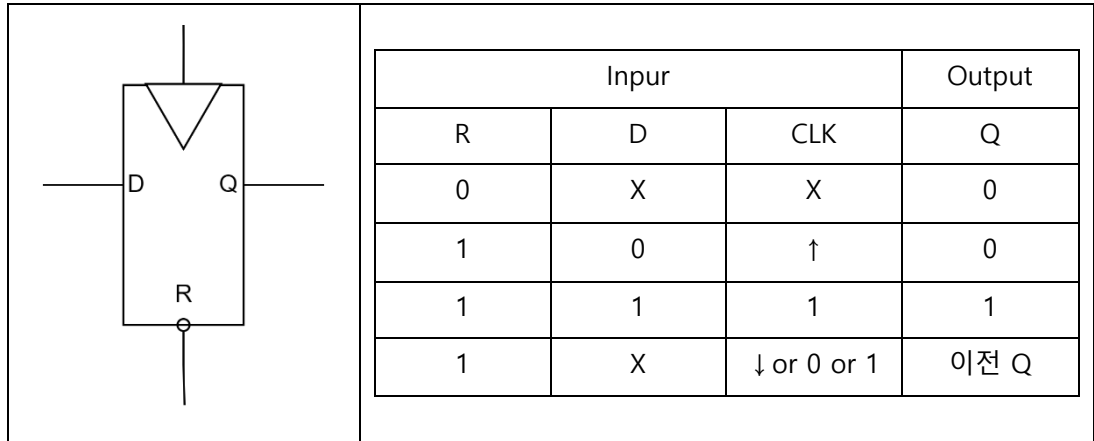
Enabled D Flip-Flop은 en의 값에 따라

현재 실습에서 구현한 enabled D flip-flop은 다음과 같다. 2-to-1 mux를 이용해서 en값을 받아 D의 값을 적용할지 아닐지를 결정한다. En이 1이 된다면 D, 1이 아니라면 이전 Q값으로 flip flop으로 들어간다. 다음은 Enabled D flip-flop의 symbol과 truth table이다.

clk	Input		output	
	En	D	Q	Q'
0	x	x	Q_{PREV}	$!Q_{PREV}$
1	0	0	Q_{PREV}	$!Q_{PREV}$
1	0	1	Q_{PREV}	$!Q_{PREV}$
1	1	0	Q_{PREV}	$!Q_{PREV}$
1	1	1	Q_{PREV}	$!Q_{PREV}$
↑	0	0	Q_{PREV}	$!Q_{PREV}$
↑	0	1	Q_{PREV}	$!Q_{PREV}$
↑	1	0	0	1
↑	1	1	1	0

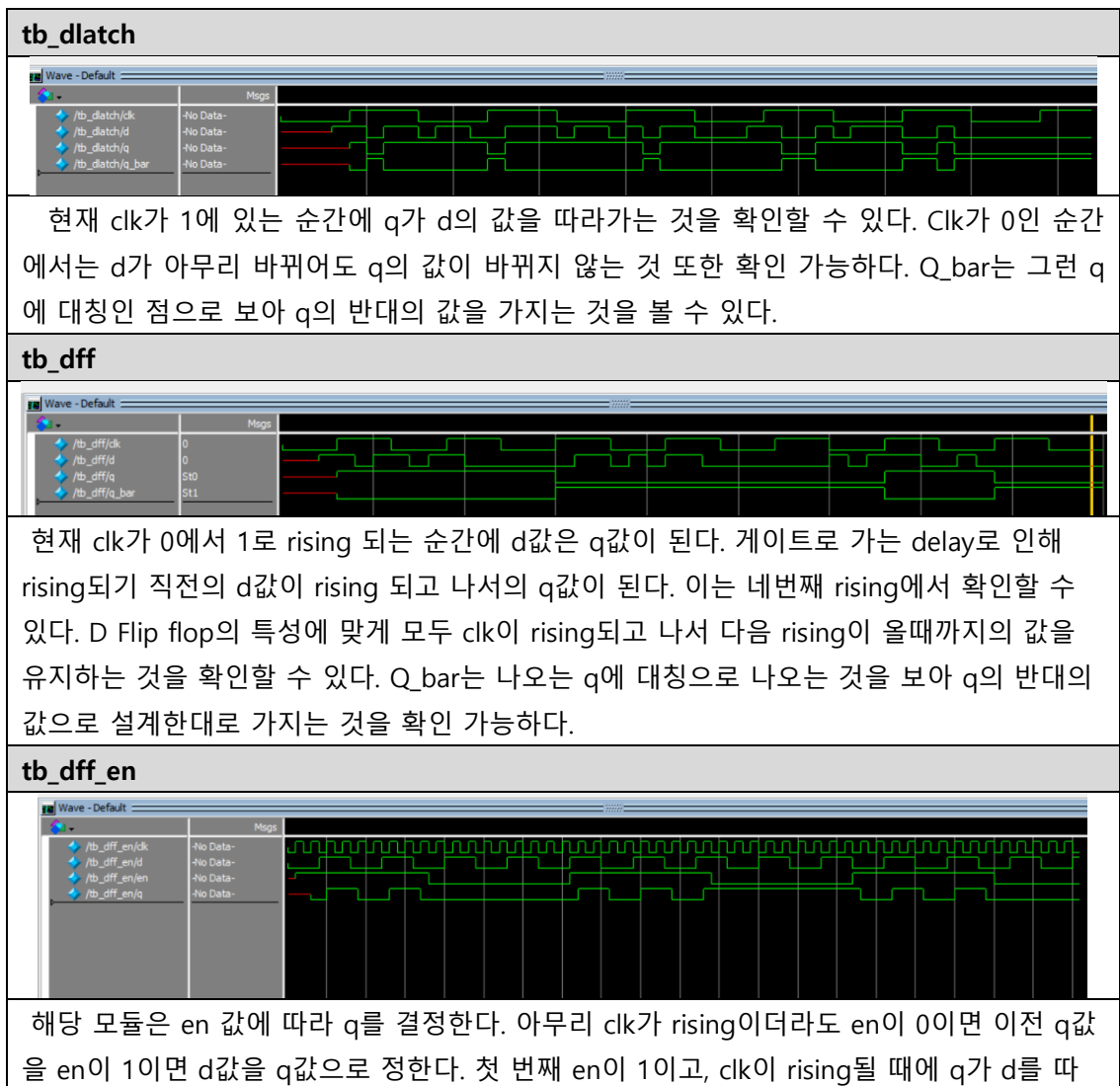
4) Resettable D Flip-Flop

Resettable D flip-flop은 D flip-flop에 reset기능이 추가된 D flip-flop이다. 실습에서 구현하는 resettable D flip-flop은 active low에 동작한다. Active low에 동작한다는 reset의 값이 0일 때, reset 기능을 수행한다는 의미이다. 다음은 Asynchronous resettable D flip-flop의 symbol과 truth table이다.



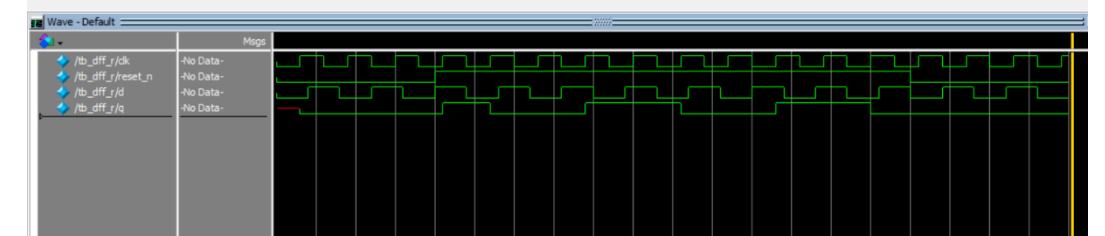
4. 설계 검증 및 실험 결과

A. 시뮬레이션 결과



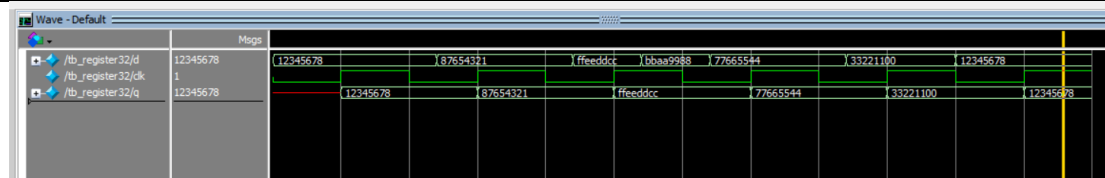
라가는 것을 확인할 수 있다. En이 0으로 가면서 q는 prev q를 따르게 된다. 이는 en이 0이 될 때, q가 0이 된 경우와 1이 된 경우에서 확인이 가능하다(첫 번째, 두 번째로 en이 0이 되는 경우).

tb_dff_r



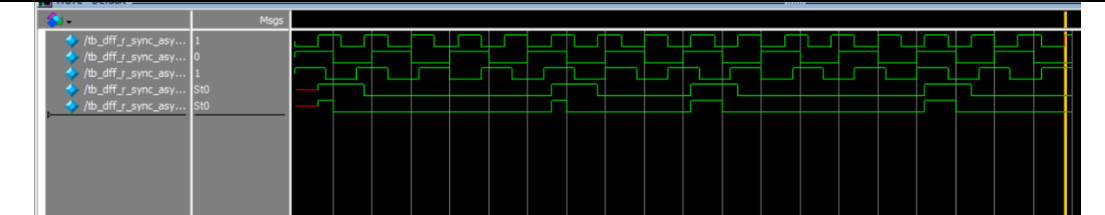
이 flip flop은 clk이 rising될 때, reset_n 이 0인 경우에는 q값이 d값에 상관없이 0으로 된다. 이 외에는 q는 clk이 rising될 때, d값으로 업데이트 되는 일반 flip flop과 동일하다. 전자의 상황은 세 번째 rising에서 확인할 수 있다. Reset_n이 1인 경우에는 D flip flop과 동일하게 작동하는지 살펴보면 D flip flop과 일치하게 동작하는 것을 확인 가능하다.

tb_register32



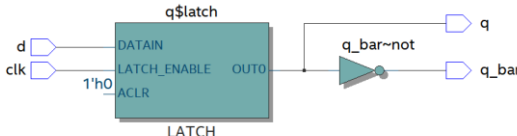
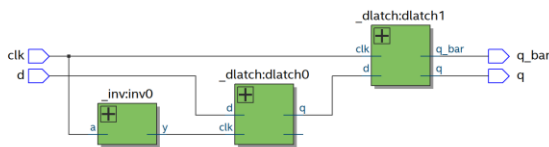
첫번째 값은 12345678로 clk이 상승한 이후 q에서 역시 같은 값을 가지고 있는 것 확인할 수 있다. 이후로 나오는 87654321, ffeeddcc도 마찬가지이다. 하지만 bbaa9988은 clk이 상승 옛지 이전에 77665544로 d가 업데이트 되면서 q까지 전달되지는 않는다. 이후로는 상승 이전에 값이 바뀌지 않고 그대로 유지되기 때문에 q까지 값이 잘 전달되는 것을 확인할 수 있다.

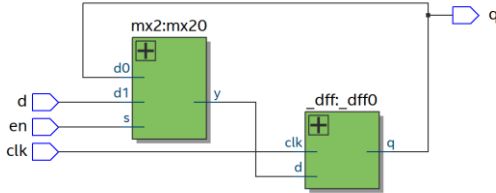
_dff_r_sync_async



아래에서 2번째는 synchronous d ff with reset이고 맨 아래에 있는 것은 asynchronous d ff with reset이다. 위에서 보면 알 수 있듯이 asynchronous한 것은 reset_n이 0이 될 때마다 값이 0으로 clear되고 있다. 하지만 synchronous한 것은 reset_n이 0이 되더라도 clk이 다음 상승할 때까지 값을 유지하며 나중에 0이 되는 것을 두 번째 클락 상승엿지에서 확인할 수 있다. 이 특이점을 제외하고는 다른 여타 flip flop과 마찬가지로 clk이 상승엿지일때 d의 값을 q로 되는 것을 확인할 수 있다.

B. 합성(synthesis) 결과

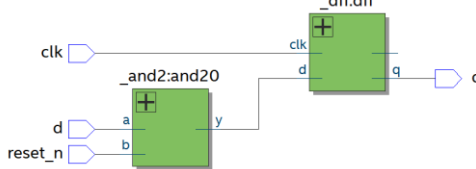
_dlatch																																							
	<div>Flow Summary</div> <div><<Filter>></div> <table><tr><td>Flow Status</td><td>Successful - Tue Oct 10 00:26:19 2023</td></tr><tr><td>Quartus Prime Version</td><td>18.1.0 Build 625 09/12/2018 SJ Lite Edition</td></tr><tr><td>Revision Name</td><td>assign05</td></tr><tr><td>Top-level Entity Name</td><td>_dlatch</td></tr><tr><td>Family</td><td>Cyclone V</td></tr><tr><td>Device</td><td>5CSXFC6D6F31C6</td></tr><tr><td>Timing Models</td><td>Final</td></tr><tr><td>Logic utilization (in ALMs)</td><td>1 / 41,910 (< 1 %)</td></tr><tr><td>Total registers</td><td>0</td></tr><tr><td>Total pins</td><td>4 / 499 (< 1 %)</td></tr><tr><td>Total virtual pins</td><td>0</td></tr><tr><td>Total block memory bits</td><td>0 / 5,662,720 (0 %)</td></tr><tr><td>Total DSP Blocks</td><td>0 / 112 (0 %)</td></tr><tr><td>Total HSSI RX PCSs</td><td>0 / 9 (0 %)</td></tr><tr><td>Total HSSI PMA RX Deserializers</td><td>0 / 9 (0 %)</td></tr><tr><td>Total HSSI TX PCSs</td><td>0 / 9 (0 %)</td></tr><tr><td>Total HSSI PMA TX Serializers</td><td>0 / 9 (0 %)</td></tr><tr><td>Total PLLs</td><td>0 / 15 (0 %)</td></tr><tr><td>Total DLLs</td><td>0 / 4 (0 %)</td></tr></table>	Flow Status	Successful - Tue Oct 10 00:26:19 2023	Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition	Revision Name	assign05	Top-level Entity Name	_dlatch	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 Status	Successful - Tue Oct 10 00:26:19 2023																																						
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition																																						
Revision Name	assign05																																						
Top-level Entity Name	_dlatch																																						
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 %)																																						
해당 RTL map는 dlatch로 q와 q_bar가 존재하는 것을 확인할 수 있다.	Top level은 _dlatch로 잘 컴파일 된 걸 확인할 수 있다. Logic Utilization은 1, Total pins는 4개이다. Resgister는 사용하지 않았으므로 0이다.																																						
_dff																																							
	<div>Flow Summary</div> <div><<Filter>></div> <table><tr><td>Flow Status</td><td>Successful - Tue Oct 10 00:38:00 2023</td></tr><tr><td>Quartus Prime Version</td><td>18.1.0 Build 625 09/12/2018 SJ Lite Edition</td></tr><tr><td>Revision Name</td><td>assign05</td></tr><tr><td>Top-level Entity Name</td><td>_dff</td></tr><tr><td>Family</td><td>Cyclone V</td></tr><tr><td>Device</td><td>5CSXFC6D6F31C6</td></tr><tr><td>Timing Models</td><td>Final</td></tr><tr><td>Logic utilization (in ALMs)</td><td>2 / 41,910 (< 1 %)</td></tr><tr><td>Total registers</td><td>0</td></tr><tr><td>Total pins</td><td>4 / 499 (< 1 %)</td></tr><tr><td>Total virtual pins</td><td>0</td></tr><tr><td>Total block memory bits</td><td>0 / 5,662,720 (0 %)</td></tr><tr><td>Total DSP Blocks</td><td>0 / 112 (0 %)</td></tr><tr><td>Total HSSI RX PCSs</td><td>0 / 9 (0 %)</td></tr><tr><td>Total HSSI PMA RX Deserializers</td><td>0 / 9 (0 %)</td></tr><tr><td>Total HSSI TX PCSs</td><td>0 / 9 (0 %)</td></tr><tr><td>Total HSSI PMA TX Serializers</td><td>0 / 9 (0 %)</td></tr><tr><td>Total PLLs</td><td>0 / 15 (0 %)</td></tr><tr><td>Total DLLs</td><td>0 / 4 (0 %)</td></tr></table>	Flow Status	Successful - Tue Oct 10 00:38:00 2023	Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition	Revision Name	assign05	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 %)
Flow Status	Successful - Tue Oct 10 00:38:00 2023																																						
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition																																						
Revision Name	assign05																																						
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 %)																																						
D flip flop은 dlatch 2개와 inverer를 이용해서 구현한다. 그림에서 볼 수 있듯이 잘 연결됨을 확인할 수 있다.	Top module은 _dff로 잘 컴파일 된 것을 확인할 수 있다. Logic Utilization은 2, Total pins는 4개이다. Resgister는 사용하지 않았으므로 0이다.																																						
_dff_en																																							

	<div>Flow Summary</div> <div><<Filter>></div> <div>Flow Status Successful - Tue Oct 10 00:50:48 2023</div> <div>Quartus Prime Version 18.1.0 Build 625 09/12/2018 S.J Lite Edition</div> <div>Revision Name assign05</div> <div>Top-level Entity Name _dff_en</div> <div>Family Cyclone V</div> <div>Device 5CSXFC6D6F31C6</div> <div>Timing Models Final</div> <div>Logic utilization (in ALMs) 2 / 41,910 (< 1 %)</div> <div>Total registers 0</div> <div>Total pins 4 / 499 (< 1 %)</div> <div>Total virtual pins 0</div> <div>Total block memory bits 0 / 5,662,720 (0 %)</div> <div>Total DSP Blocks 0 / 112 (0 %)</div> <div>Total HSSI RX PCSs 0 / 9 (0 %)</div> <div>Total HSSI PMA RX Deserializers 0 / 9 (0 %)</div> <div>Total HSSI TX PCSs 0 / 9 (0 %)</div> <div>Total HSSI PMA TX Serializers 0 / 9 (0 %)</div> <div>Total PLLs 0 / 15 (0 %)</div> <div>Total DLLs 0 / 4 (0 %)</div>
---	--

Enabled d flip flop은 2-to-1 mux와 dff 1개로 구성된다. 해당 RTL map은 그림에서 볼 수 있듯이 잘 연결됨을 확인할 수 있다.

Top module은 _dff_en로 잘 컴파일 된 것을 확인할 수 있다.

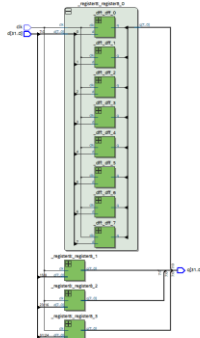
Logic Utilization은 2, Total pins는 4개이다. Resgister는 사용하지 않았으므로 0이다.

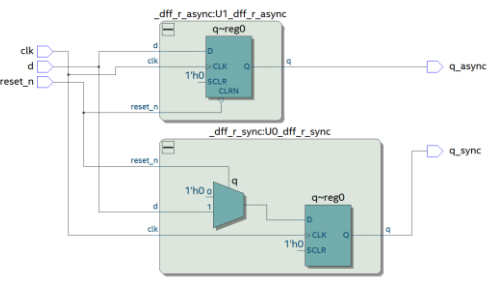
<div>_dff_r</div> 	<div>Flow Summary</div> <div><<Filter>></div> <div>Flow Status Successful - Tue Oct 10 01:02:33 2023</div> <div>Quartus Prime Version 18.1.0 Build 625 09/12/2018 S.J Lite Edition</div> <div>Revision Name assign05</div> <div>Top-level Entity Name _dff_r</div> <div>Family Cyclone V</div> <div>Device 5CSXFC6D6F31C6</div> <div>Timing Models Final</div> <div>Logic utilization (in ALMs) 2 / 41,910 (< 1 %)</div> <div>Total registers 0</div> <div>Total pins 4 / 499 (< 1 %)</div> <div>Total virtual pins 0</div> <div>Total block memory bits 0 / 5,662,720 (0 %)</div> <div>Total DSP Blocks 0 / 112 (0 %)</div> <div>Total HSSI RX PCSs 0 / 9 (0 %)</div> <div>Total HSSI PMA RX Deserializers 0 / 9 (0 %)</div> <div>Total HSSI TX PCSs 0 / 9 (0 %)</div> <div>Total HSSI PMA TX Serializers 0 / 9 (0 %)</div> <div>Total PLLs 0 / 15 (0 %)</div> <div>Total DLLs 0 / 4 (0 %)</div>
---	---

D flip flop with reset은 d flip flop과 d로 들어갈 때 reset_n이 and gate를 거쳐 가는 것을 설계와 같은 것을 확인할 수 있다.

Top module은 _dff_r로 잘 컴파일 된 것을 확인할 수 있다.

Logic Utilization은 2, Total pins는 4개이다. Resgister는 사용하지 않았으므로 0이다.

<div>_register32</div> 	<div>Flow Summary</div> <div><<Filter>></div> <div>Flow Status Successful - Tue Oct 10 01:49:13 2023</div> <div>Quartus Prime Version 18.1.0 Build 625 09/12/2018 S.J Lite Edition</div> <div>Revision Name assign05</div> <div>Top-level Entity Name _register32</div> <div>Family Cyclone V</div> <div>Device 5CSXFC6D6F31C6</div> <div>Timing Models Final</div> <div>Logic utilization (in ALMs) 33 / 41,910 (< 1 %)</div> <div>Total registers 0</div> <div>Total pins 65 / 499 (13 %)</div> <div>Total virtual pins 0</div> <div>Total block memory bits 0 / 5,662,720 (0 %)</div> <div>Total DSP Blocks 0 / 112 (0 %)</div> <div>Total HSSI RX PCSs 0 / 9 (0 %)</div> <div>Total HSSI PMA RX Deserializers 0 / 9 (0 %)</div> <div>Total HSSI TX PCSs 0 / 9 (0 %)</div> <div>Total HSSI PMA TX Serializers 0 / 9 (0 %)</div> <div>Total PLLs 0 / 15 (0 %)</div> <div>Total DLLs 0 / 4 (0 %)</div>
--	--

<p>_register32는 regitster를 32개를 이은 것으로 _register8을 4개로 구성한 것을 확인 가능하다.</p>	<p>Top module은 _register32로 잘 컴파일 된 것을 확인할 수 있다.</p> <p>Logic Utilization은 33, Total pins는 65 개이다. 32bit register를 구현하는 했지만 이는 dff 기억 소자를 만들고 이를 이은 것으로 프로그램의 resgister는 사용하지 않았으므로 0개이다.</p>																																						
<p>_dff_r_sync_async</p>																																							
	<p>Flow Summary</p> <p><<Filter>></p> <table border="1"> <thead> <tr> <th>Flow Status</th> <th>Successful - Tue Oct 10 02:28:41 2023</th> </tr> </thead> <tbody> <tr> <td>Quartus Prime Version</td> <td>18.1.0 Build 625 09/12/2018 S.J Lite Edition</td> </tr> <tr> <td>Revision Name</td> <td>assign05</td> </tr> <tr> <td>Top-level Entity Name</td> <td>_dff_r_sync_async</td> </tr> <tr> <td>Family</td> <td>Cyclone V</td> </tr> <tr> <td>Device</td> <td>5CSXFC6D6F31C6</td> </tr> <tr> <td>Timing Models</td> <td>Final</td> </tr> <tr> <td>Logic utilization (in ALMs)</td> <td>1 / 41,910 (< 1 %)</td> </tr> <tr> <td>Total registers</td> <td>2</td> </tr> <tr> <td>Total pins</td> <td>5 / 499 (1 %)</td> </tr> <tr> <td>Total virtual pins</td> <td>0</td> </tr> <tr> <td>Total block memory bits</td> <td>0 / 5,662,720 (0 %)</td> </tr> <tr> <td>Total DSP Blocks</td> <td>0 / 112 (0 %)</td> </tr> <tr> <td>Total HSSI RX PCSs</td> <td>0 / 9 (0 %)</td> </tr> <tr> <td>Total HSSI PMA RX Deserializers</td> <td>0 / 9 (0 %)</td> </tr> <tr> <td>Total HSSI TX PCSs</td> <td>0 / 9 (0 %)</td> </tr> <tr> <td>Total HSSI PMA TX Serializers</td> <td>0 / 9 (0 %)</td> </tr> <tr> <td>Total PLLs</td> <td>0 / 15 (0 %)</td> </tr> <tr> <td>Total DLLs</td> <td>0 / 4 (0 %)</td> </tr> </tbody> </table>	Flow Status	Successful - Tue Oct 10 02:28:41 2023	Quartus Prime Version	18.1.0 Build 625 09/12/2018 S.J Lite Edition	Revision Name	assign05	Top-level Entity Name	_dff_r_sync_async	Family	Cyclone V	Device	5CSXFC6D6F31C6	Timing Models	Final	Logic utilization (in ALMs)	1 / 41,910 (< 1 %)	Total registers	2	Total pins	5 / 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 Status	Successful - Tue Oct 10 02:28:41 2023																																						
Quartus Prime Version	18.1.0 Build 625 09/12/2018 S.J Lite Edition																																						
Revision Name	assign05																																						
Top-level Entity Name	_dff_r_sync_async																																						
Family	Cyclone V																																						
Device	5CSXFC6D6F31C6																																						
Timing Models	Final																																						
Logic utilization (in ALMs)	1 / 41,910 (< 1 %)																																						
Total registers	2																																						
Total pins	5 / 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 %)																																						
<p>synchronous dff과 asynchronous dff를 구현한 모습이다. Aynchornous는 mux를 이용하여 reset이 sel의 역할을 한다. Reset_n이 0인 경우 register에 오는 d값은 0이다. 이 역시 잘 연결돼 있음을 확인할 수 있다.</p>	<p>Top module은 _dff_r_sync_async로 잘 컴파일 된 것을 확인할 수 있다.</p> <p>Logic Utilization은 1, Total pins는 5개이다. register를 이용했기 때문에 resgister의 수는 2개이다.</p>																																						

5. 고찰 및 결론

A. 고찰

Synchronous reset/set은 clock의 edge에서 reset이나 set의 상태에 따라 발생한다. 이 때의 우선권은 reset이 가진다. Asynchronous reset/set은 clock의 edge에도 하지만 또 하나의 조건을 얻는다. 해당 신호가 발생하면 바로 실행한다. 만약 clk이 상승하지는 않으나 reset_n이 하강하거나, set_n이 하강하게 되면 해당 신호가 걸려 전자의 경우에는 Q가 0 후자의 경우에는 Q가 1이 걸릴 것이다. reset_n이 set보다 더 큰 우선순위를 갖는다. 둘 모두 0으로 하강하더라도 q는 0의 값을 갖는다는 이야기이다.

B. 결론

enabled flip flop을 mux를 이용하여 구현이 아닌 따로 생각해서 구현한 것이 조금 까다로웠다. 현재의 구현은 flip flop의 D 쪽에 En값과 D값을 2 inputs and gate에 연결했지만 이전에는 flip flop의 clk부분에 clk신호와 En값을 and gate를 통하는 것으로 생각했었다. 만약 이

경우에는 clk가 1로 고정되어있고, en값이 상승하여도 flip flop은 clk가 상승하는 것으로 받아들일테니 enabled flip flop과는 조금 다르다고 생각했었다. 그렇기 때문에 edge에 의존하지 않는 쪽으로 신호를 주게 되었는데, 만들고 보니 resettable flip flop처럼 되어 잘못 설계한 것과 같은 생각이 들었다. 그래서 논리적으로 접근하기 보다는 원래 enabled flip flop의 testbench와 비교하면서 봤는데, clk와 d 모두에 en 값으로 2input and gate를 주면 똑 같은 testbench를 확인할 수 있었다. 논리적으로만 구상할 것이 아닌 검증이 왜 설계에 필요한지 느꼈던 실습이었다.

6. 참고문헌

이형근 교수님/컴퓨터공학기초실험1/광운대학교(컴퓨터정보공학부)/2023

이준환 교수님/컴퓨터공학기초실험2/광운대학교(컴퓨터정보공학부)/2023