**전자전기컴퓨터설계실험2**

**Lab#06 Sequential\_Logic\_Design\_Ⅰ  
Flip-Flop, Register and SIPO**

전전컴

2012440096

이광원

**목차(Table)**

초록<Abstract>

**1. 실험 제목**

**2. 실험 목적**

**3. 실험 장치**

**4. 실험 이론**

**5. PreLab**

**6. 실험결과**

**7. 토의 및 결론**

**8. 참고문헌**

**1.실험 제목**

Flip-Flop, Register and SIPO

**2.실험 목적**

이번 실험은 조합 논리 회로를 배우고 verilog HDL을 활용해 플리플롭 회로를 Behavioral Modeling으로 설계한다.

**3. 실험 장치**

자일링스 프로그램, 디바이스 포트, 자일링스 디바이스

**4. 실험 이론**

(1)플립플롭 회로

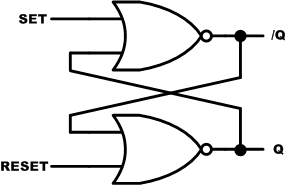
조합 논리 : 출력 결과가 입력으로 들어오는 값에 의해 정해짐. 이전의 결과 또는 입력 신호에 의하여 동작이 구성되기 위해서는, 그 값을 저장해 줄 기억소자가 필요함.



- 가장 많이 사용되는 기억소자가 플립플롭

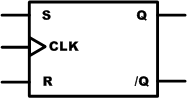
(2) 래치(LATCH)

2개의 NOR 게이트로 구성된 래치의 동작



(2) S-R 플립플롭

S-R 래치에 클럭의 입력이라는 부분을 추가한 회로

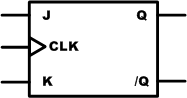




(3) J-K 플립플롭

J=K=1인 조건에서 모호한 출력상태를 갖지 않는 다는 동작을 제외하고, S-R 플립플롭과 동일한 상태를 제어함.

J=K=1인 조건에서 플립플롭은 클럭의 신호에 대하여 항상 출력값을 반전시킴.





**반전**

(4) D 플립플롭

오직 하나의 데이터 입력을 갖음.

클럭이 발생하였을 때, 입력 D의 상태를 Q에 전달함



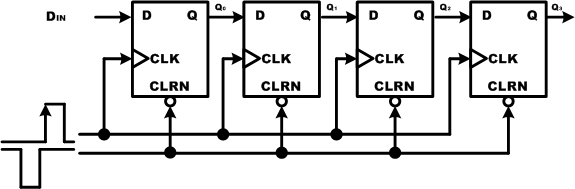
(5) 4비트 SIPO

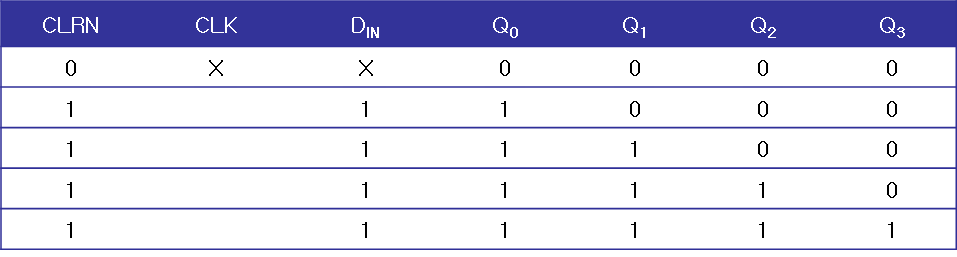
직렬입력/병렬출력 레지스터

SIPO(Serial Input Parallel Output) Register

직렬 데이터의 입력을 받아 병렬의 데이터를 출력하는 기능을 함.

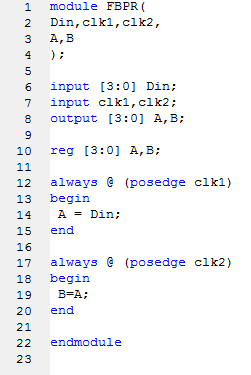
N 비트의 직렬 입력/병렬 출력 레지스터는 N개의 클럭이 발생하는 데이터에 의해서 병렬 데이터가 결정된다. 시프트 레지스터와 같은 동작을 함.



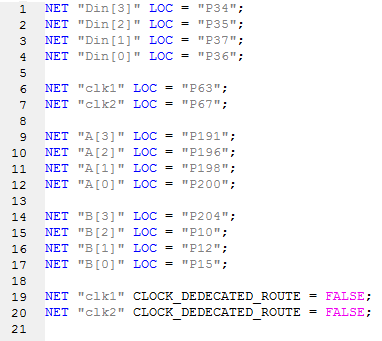


**5. PreLab**

1)Four bit Parallel Register

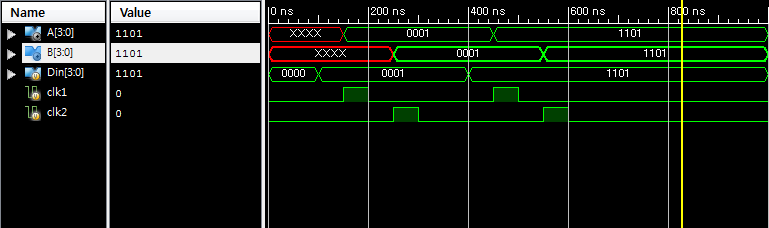


코드를 위와 같이 짜보았다. Clk1이 positive edge를 가질 때 A=Din의 값을 가지게 하고 clk2가 positive edge를 가질 때 B=A의 값을 가지게 나타내었다.



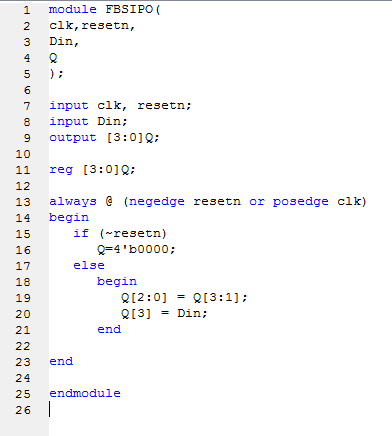
핀 번호 설정 후

Test Bench를 clk1과 clk2가 50 ns마다 positive edge를 가지게 하였다.

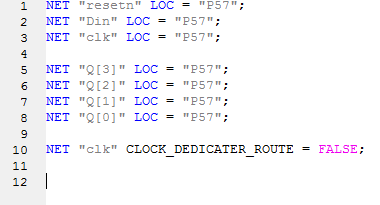


예상대로, Din에 0001값을 넣은 후 clk1가 positive edge를 가질 때 값이 알 수 없음-> 0001을 가지게 되었고, 그 이후에 clk2가 positive edge를 가질 때 B가 A값인 0001을 출력했다. 마찬가지로 Din에 이후에 1101을 넣고 clk1가 positive edge를 가질 때 값이 0001-> 1101을 가지게 되었고, 그 이후에 clk2가 positive edge를 가질 때 B가 A값인 1101을 출력했음을 알 수 있다.

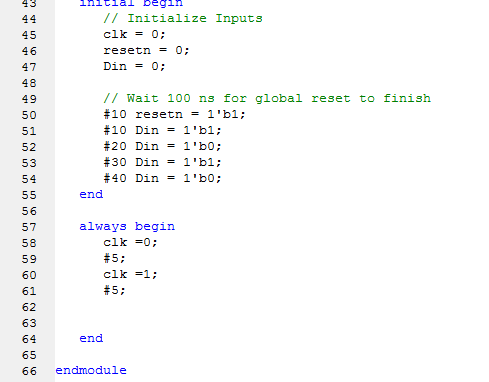
2)Four bit Serial Input Parallel Output



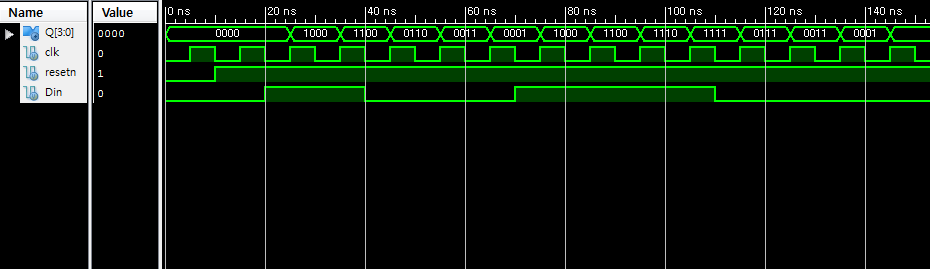
위와 같이 코드를 짜보았다. Reset을 시켜주면 모든 Q값은 0이 되고, 그 이외의 경우엔, clk 값이 positive edge를 가질 때 마다 Q3Q2Q1Q0 값이 오른쪽으로 한 칸 씩 병렬이동 한다. Q3Q2Q1값이 Q2Q1Q0 값이 되며 Q3는 새로들어오는 Din값이 된다.



핀번호를 이와 같이 설정하고



clk값이 5ns마다 positive edge를 가지게 Test Bench 설정후



시뮬레이션을 돌려보았다. 오른쪽으로 병렬이동한다는 것이 무슨 뜻이냐면, resetn이 1일 때 (resetn이 0일땐 모든 Q값이 0) Din에 1을 넣어주는 동안 Q값은 1000->1100 이런식으로 계속 오른쪽으로 병렬 이동하게 된다. 하지만 중간에 다시 Din에 0값을 넣었을 경우, Q3값은 새로들어오는 Din값이기 때문에 1100->0110->0011->0001 이렇게 이동하게 된다. 만약 Din에 1값을 계속 준다면, 1000->1100->1110->1111->1111 이런 양상을 보이게 될 것이다.

**6. 실험 결과**

**1) 4 bit Shift Register**

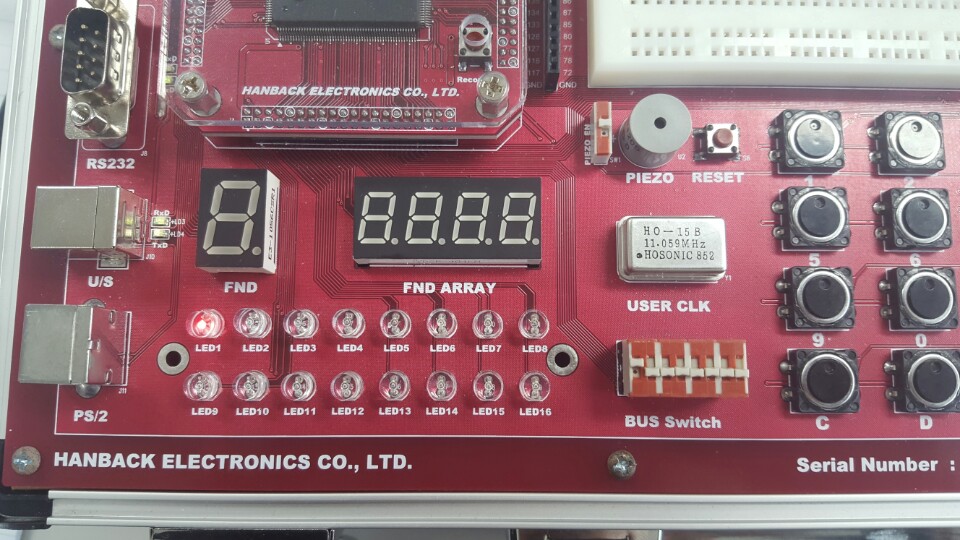
짜여진 Code에 ucf 파일을 만들어 bit파일을 생성하려고 했지만 계속 오류가 났다.

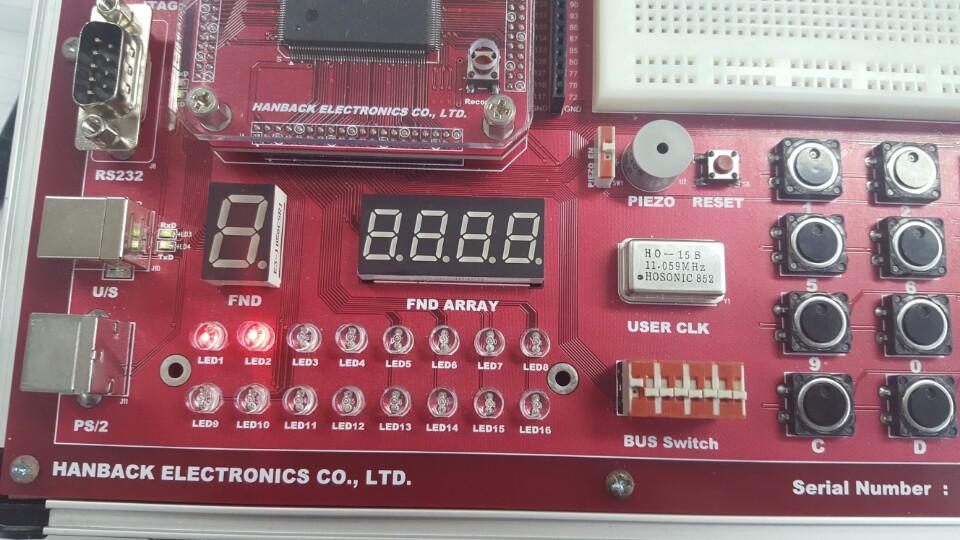
그 문제를 해결하기 위하여 ucf파일 마지막줄에 다음과 같은 문장을 써주었더니 에러가 사라졌다.



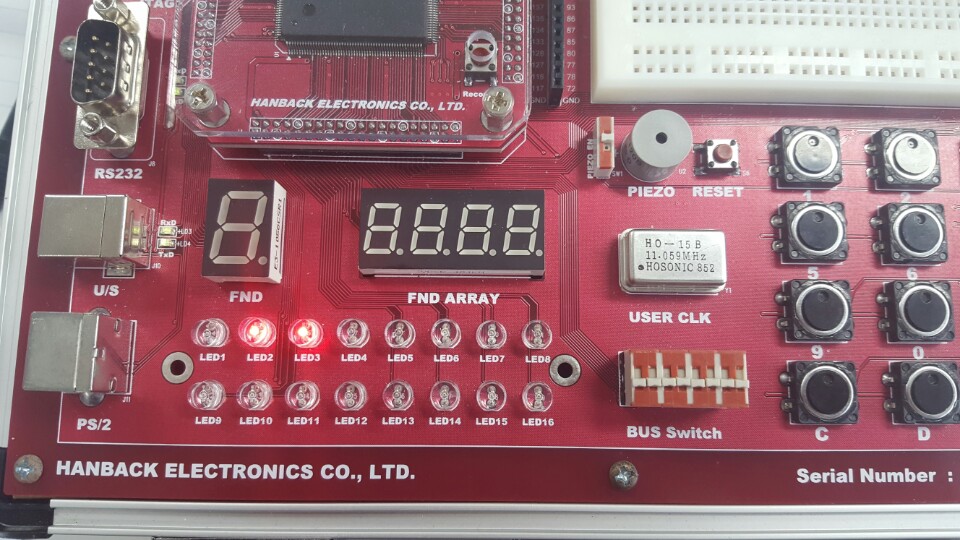
원래는 CLK의 입력을 Switch 버튼으로 지정을 해 줄 수 가 없기 때문에 다음과 같은 문장을 써서 에러를 방지하는 것 인 것 같다.

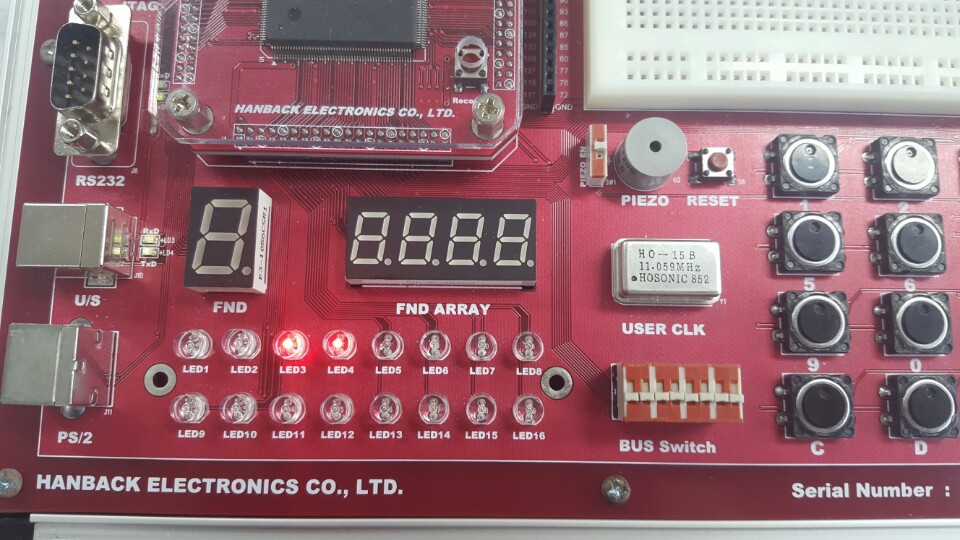
그리고 Bit파일을 만들어서 장비를 작동시켜보았다.

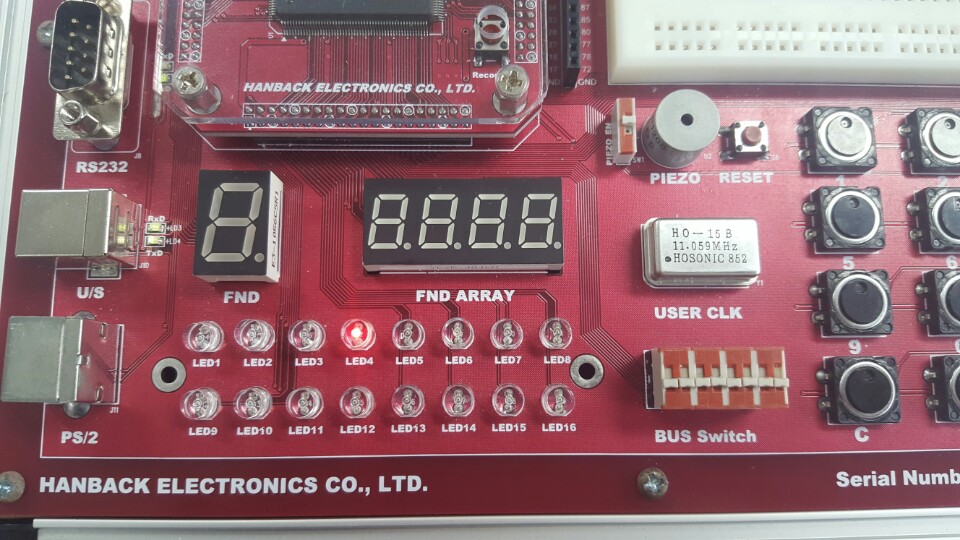




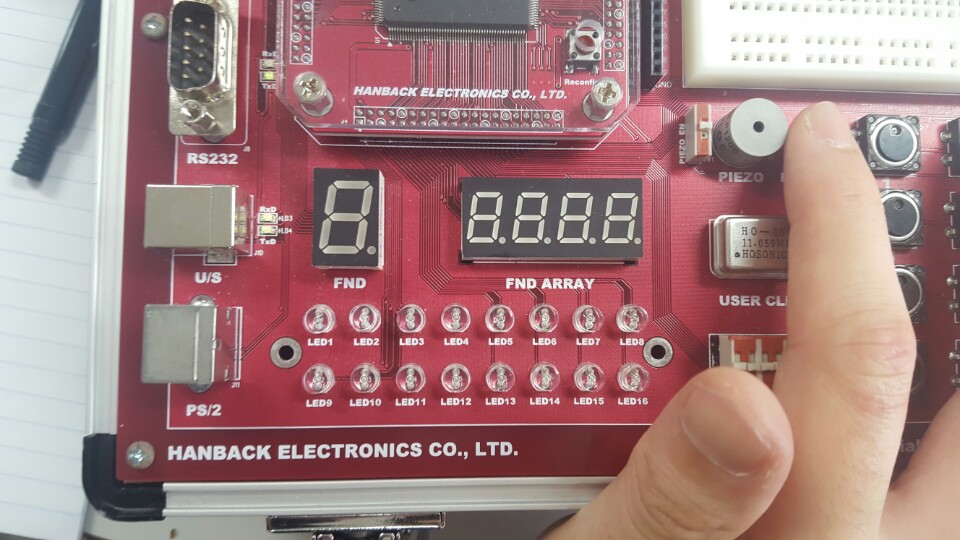
버스 스위치를 1로 두고 버튼스위치륻 두 번 눌렀더니 두 번의 클락동안 0000->1000->1100 이 됬다.





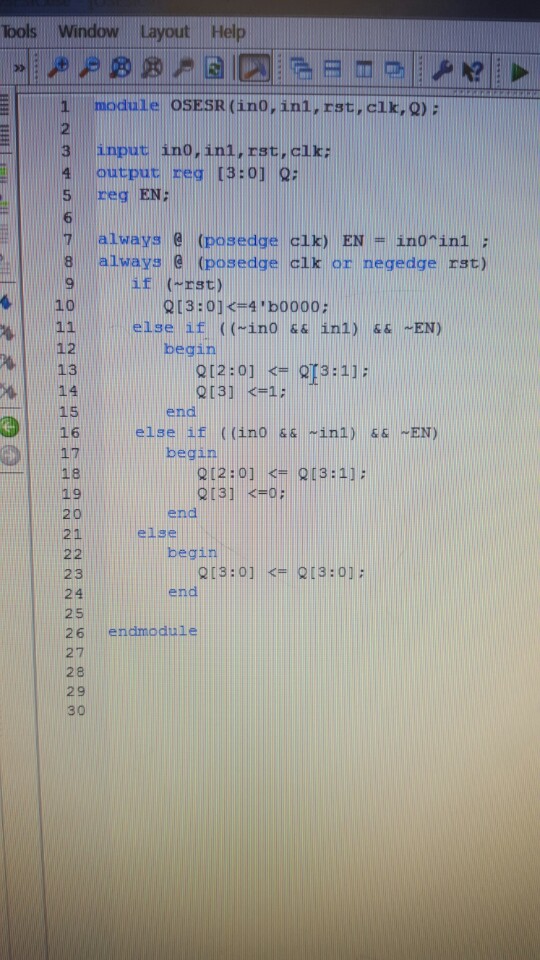


이번엔 버스스위치를 0에 두고 버튼스위치를 세번 눌렀더니 1100->0110->0011->0001로 변했다.

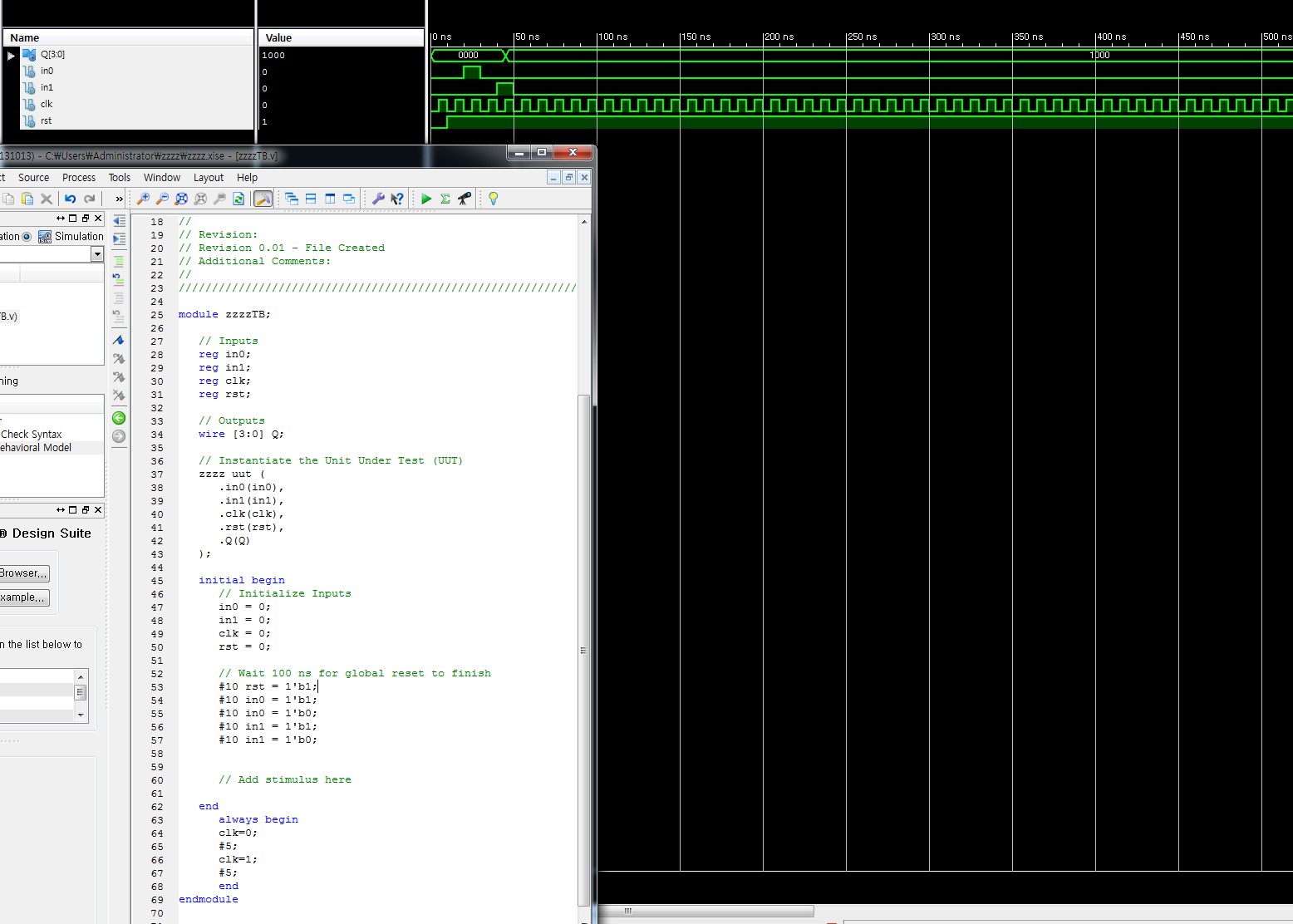


Reset 버튼을 눌렀을때는 모든 Q값이 0000이 된다.

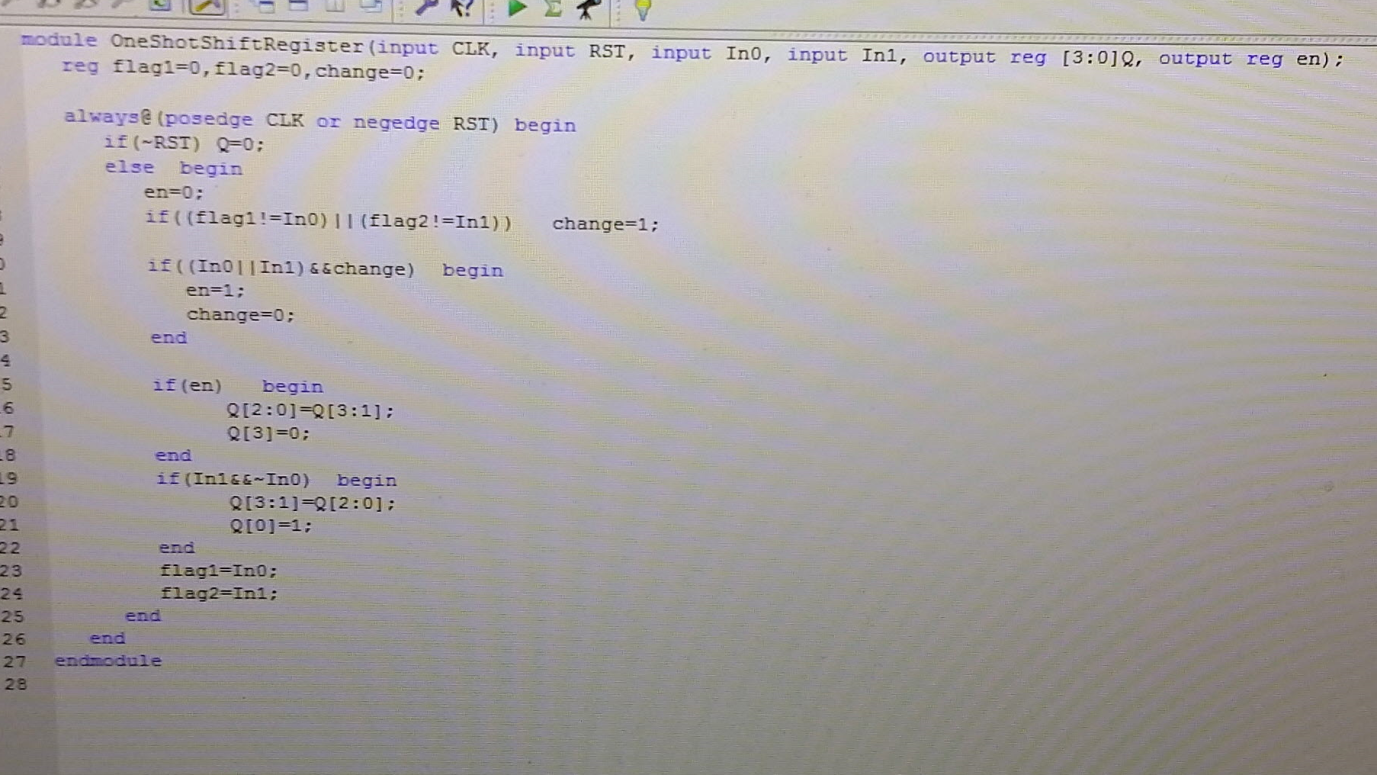
2) 4 bit Shift Register included One Shot Enable



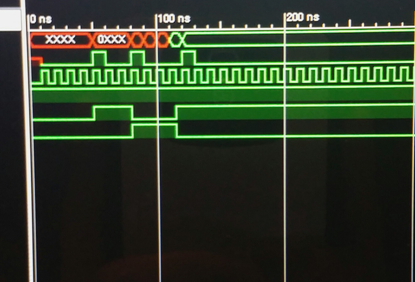
처음 실험 할 때 짰던 코드이다. 일단 실험실에서 위의 코드로 장비에 연결했을 때 결과는 정상적으로 나왔다. 하지만 코드를 계속 보면 분명히 문제가 있다는 것을 알 수 있다.



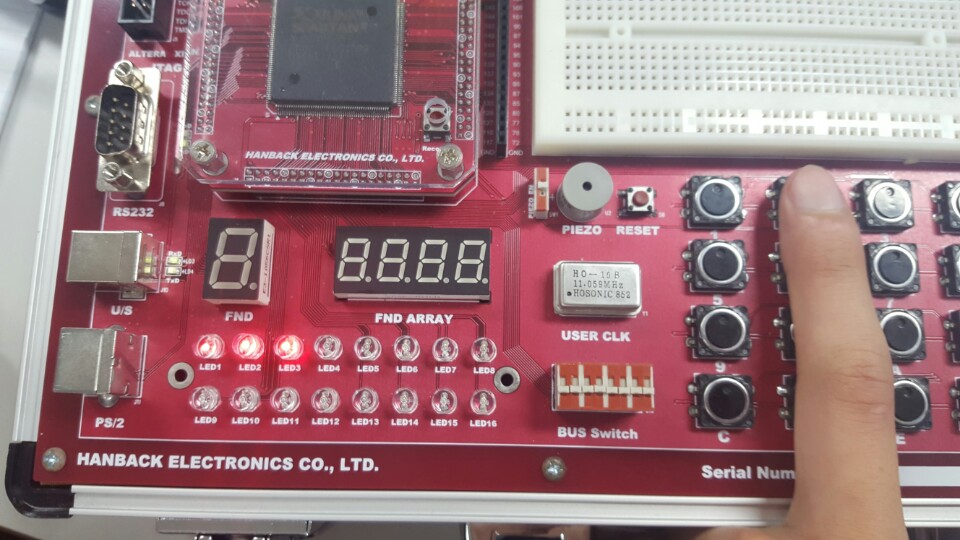
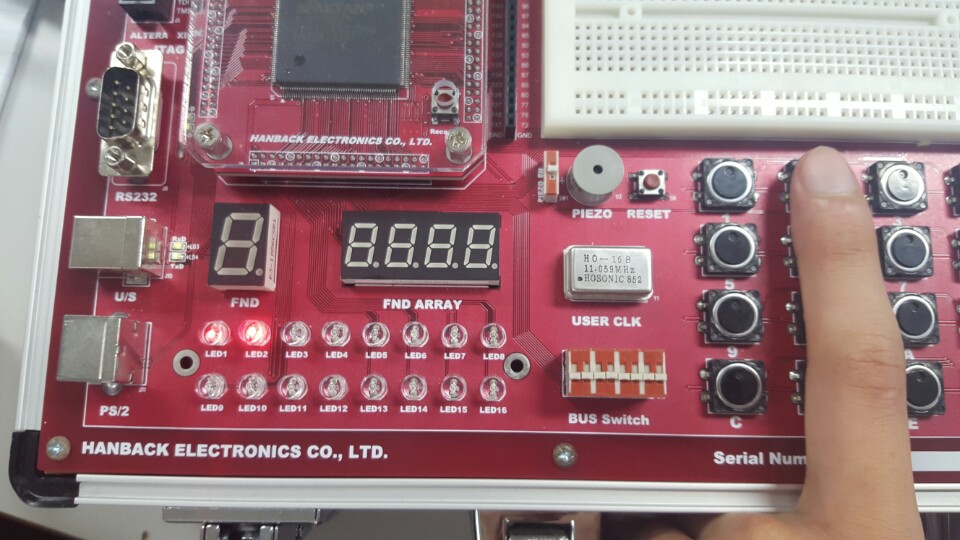
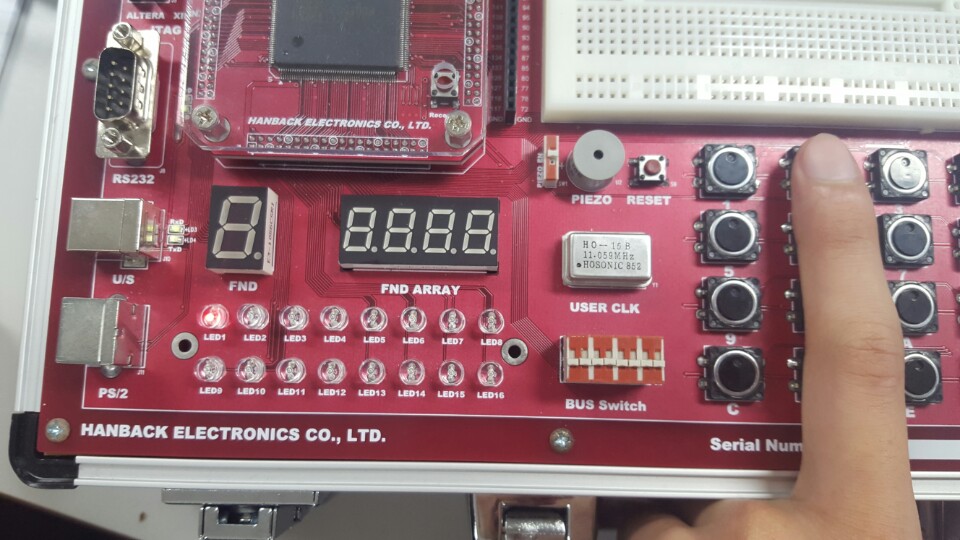
시뮬레이션으로 확인하면 EN이 1이되면 계속 1인상태가 유지되고 0으로 떨어지지 않는 것을 확인 할 수 있다. 장비상으로는 enable값을 확인 할 수 없으므로 결과가 정상적으로 나왔던 것이다.



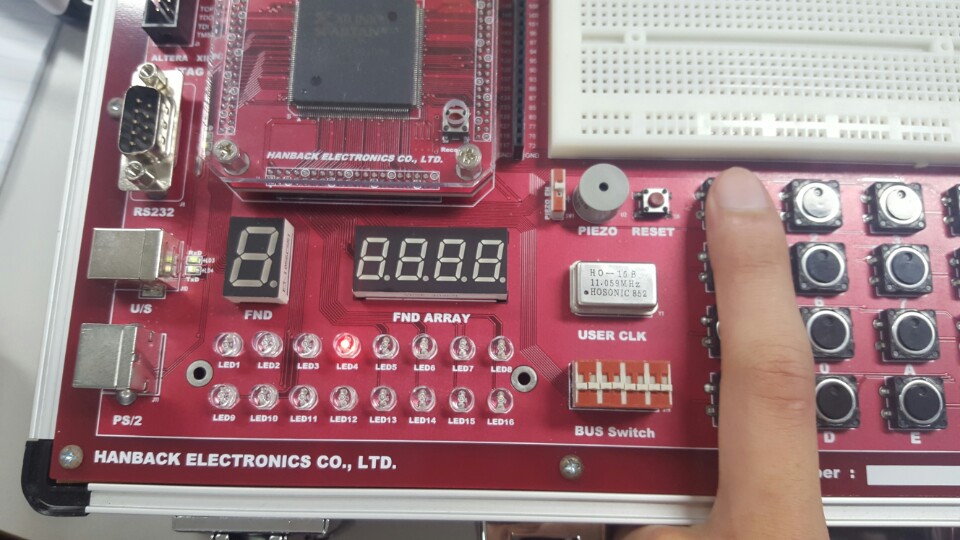
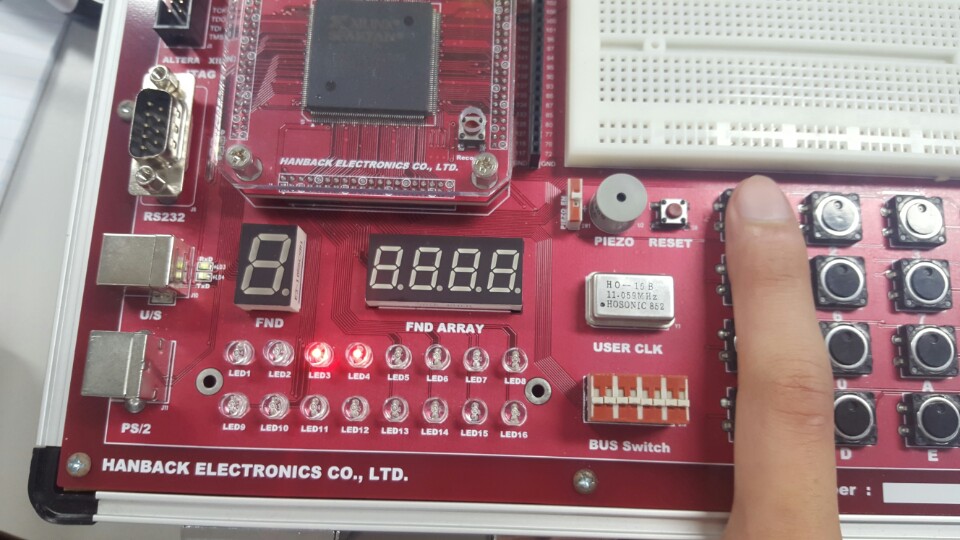
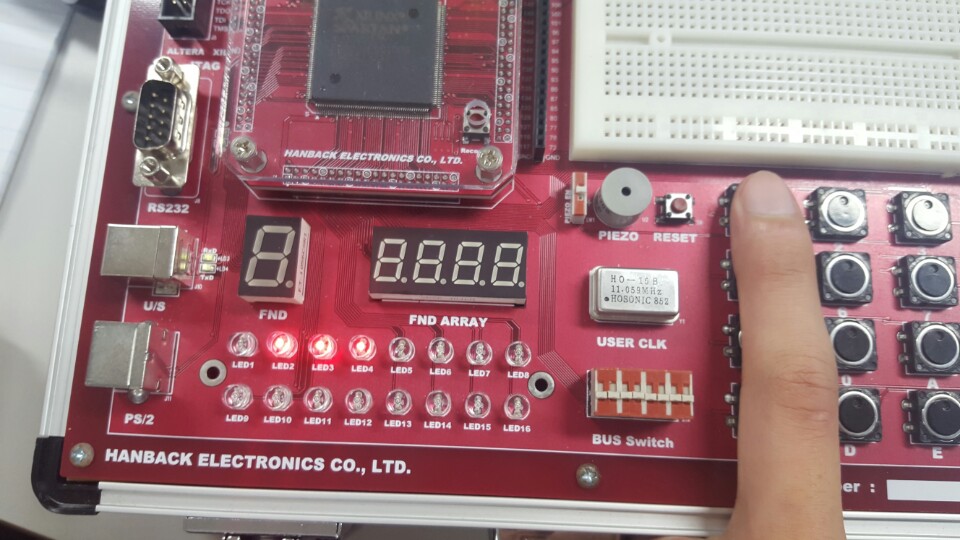
그래서 코드를 다시 짜보았다. 이번엔 flag와 chage라는 변수를 추가했다. Flag에 in0값이 들어가다가 바뀌면 첫번째 if문 조건 충족되서 change 변수가 1이 되고, change가 1이고 in0또는 in1이 1이면 en에 1이 들어가고 change는 변수 초기화 시켜주고 아래 if문을 shift 해주는 것이다. 이렇게 설계함으로써 en은 in0,in1 일때 튀어 오를 것이다.



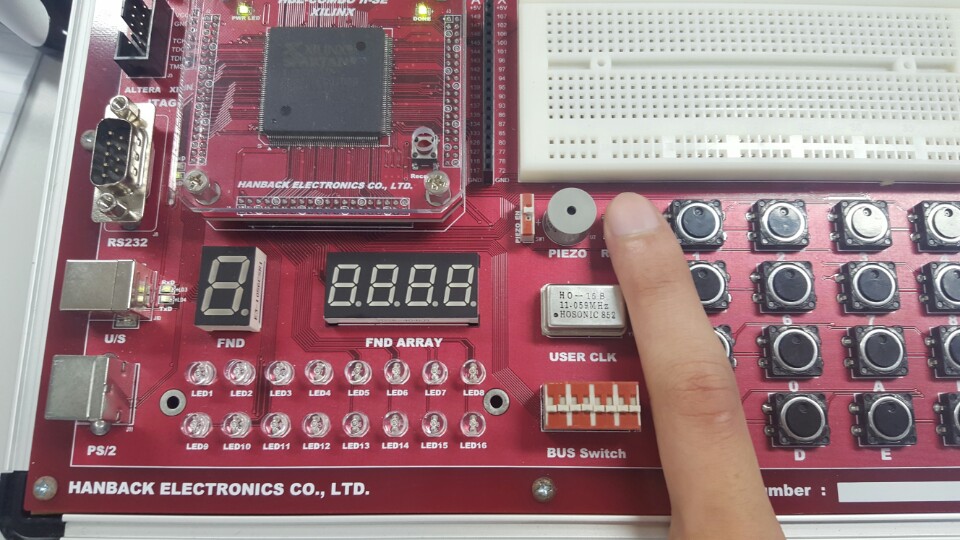
시뮬레이션 결과이다. 사진처럼 클락은 5ns마다 1과 0이 되고, en값은 in0이나 in1 일 때 튀어 올랐고 en이 한 클락 만큼만 튀어 오르게 해야 하는데 그렇게 하는 법을 몰라서 그 다음 클락이 시작할 때 떨어지게 하였다.



위에는 in1을 세 번 연속 눌렀을 때 모습이다. 차례로 1이 들어옴을 볼 수 있다.



그 이후 in0을 세 번 연속 누른 모습이다. 0값이 차례로 들어옴을 볼 수 있다.



Reset을 시켰더니 모든 출력값이 0이 되었다.

**7. 토의 및 결론**

이번실험에서 무엇보다도 Flip-Flop에 대한 회로 이해가 가장 중요하다고 생각 된다. 회로를 설계하는 것은 단순히 그 회로를 언어로 표현하는 것일 뿐 실제로는 회로에 대한 구성과 작동원리 인풋 아웃풋 선택 버튼 등에 대한 이해와 입력이 출력에 미치는 영향 등을 전반적으로 고려 할 줄 알아야 한다.

그리고 이번 실험을 통해 always구문을 사용하여 설계하는 방법과, if문을 여러 개 썼을 시에 Begin 과 end를 잘 사용하여야 한다는 점을 배울 수 있었다. 그리고 서로 다른 always구문에서는 같은 변수 값을 대입할 수 없었다는 것도 알게 되었다. 그래서 en이 1이 될 때는 지정해줄 수 있었지만 0이 될 때를 지정해주지 못해서 계속 en이 1이 되었고 결국 flag와 change라는 변수를 추가해주어 만들어주게 되었다. 소스를 구성함에 있어서 다른 특별한 어려운 점은 없었지만, , Reset 버튼을 눌렀을시 초기화 되는 방법과, 클락이 정확이 입력되었을때에 불이 들어오게 하는 방법등 이번 실험을 생각 해야 할 점이 많았던 실험이었다. 그래서 그런지 여태까지 배운것들이 잘 조합되었고, 좀더 코드짜는 실력을 늘릴 수 있었다

**8.참고문헌**

-전전설2 교안 7주차

http://waltz.uos.ac.kr/ece-experiments-%e2%85%a1