**디지털 논리 회로 실습**

**기말 프로젝트**

**- 도어 락 설계 -**

**컴퓨터 과학과**

**2013147555 김동일**

**작성일 : 2014년12월15일**

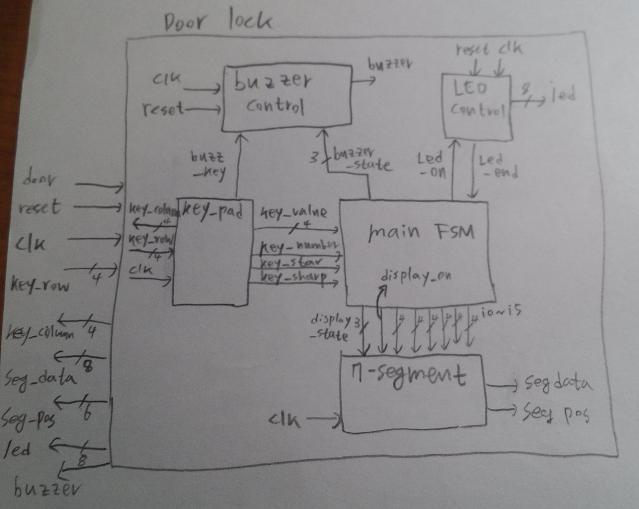
**1. 프로젝트 개요**

프로젝트 내용: Quartus 2 툴을 사용하여, Combinational Door lock을 FPGA 상에서 구현 하기.

프로젝트 목적: 수업에서 배운 여러 Combinational/Sequential Logic 설계 기법을 활용하여 회로를 설계

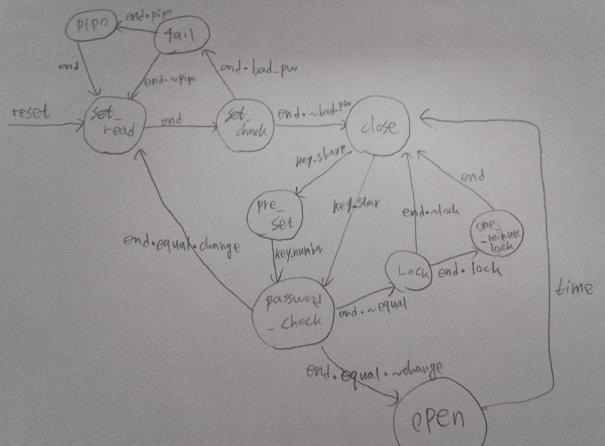
프로젝트 수행: Door lock을 설계하기 위하여 블록다이어그램과 스테이트 다이어그램을 설계하여 구현하였다. 2가지 부가기능이 작동하지 않지만 그 외에 기능은 작동하도록 하였다.

**2. 블록 다이어그램**



key\_pad 모듈은 키 값을 받아 부저가 울리는 신호를 buzzer\_controler로 보내고 main\_FSM으로 무슨 키를 받았는지를 보낸다. main\_FSM 에서는 기타 아웃풋 컨트롤 모듈들에게 제어 신호를 보낸다. buzzer\_control, LED\_control, 7\_segment 모듈들은 각각 제어신호를 받아 아웃풋 신호를 내보낸다.

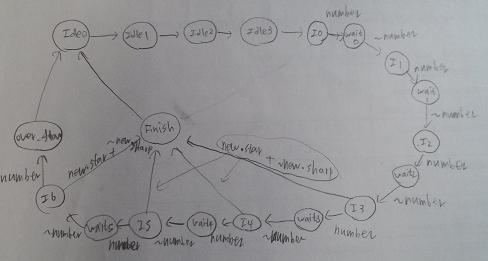
**3. FSM 상태 전이도 다이어그램**



주 스테이트 다이어그램

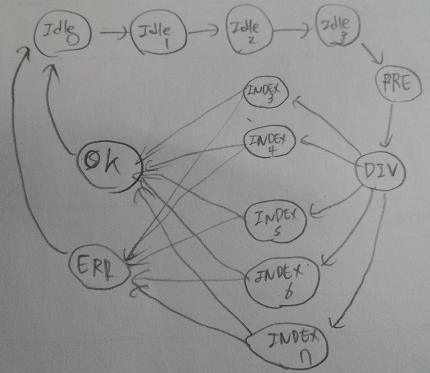
end라는 신호는 각각의 스테이트 마다 내부 스테이트들이 있어서 내부 스테이트가 끝나면 end 신호가 set된다. set\_read 스테이트는 비밀번호를 새로 받는 스테이트 이고, set\_check는 비밀번호가 적절한지를 판단하는 스테이트 이다. 적절하지 않으면 bad\_pw가 set되서 fail 스테이트로 넘어가고 5분이내 3번이상 틀렸는지를 확인해서 3번이상 틀렸을시 pipo가 set되서 pipo 상태로 넘어간다. 아니면 set\_read상태로 돌아온다. set\_check 상태에서 비밀번호가 적절할 때 close 스테이트로 넘어오고 별을 누르면 password\_check 스테이트로 넘어가고, 샵을 누르면 pre\_set 스테이트로 넘어간다 pre\_set 스테이트는 change 신호를 set해주는 역할을 한다. 비밀번호를 받고 체크해서 맞으면 equal이 set된다 틀리면 lock 스테이트로 넘어가고 여기도 fail, pipo 스테이트와 마찬가지로 3회이상 틀리면 one\_minute\_lock스테이트로 넘어간다. password\_check 스테이트에서 비밀번호가 맞을 때 change 신호까지 확인해서 open 스테이트로 가던지 비밀번호를 바꾸는 set\_read 스테이트로 간다.

내부 스태이트 중



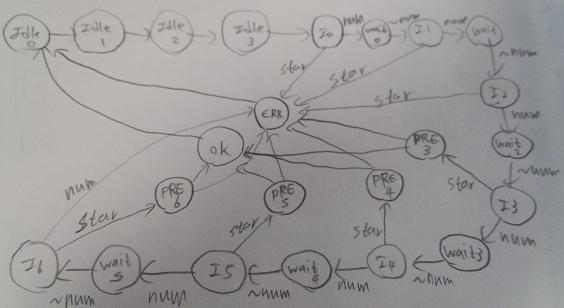
set\_read state의 내부 스테이트

idle state 들은 안정성을 높이려고 아무 것도 안하고 클럭마다 돌아가는 스테이트 이다. I0~I6는 숫자가 들어오면 저장하는 스테이트 이다. wait는 키의 숫자가 눌렸다가 땔때까지 기다린다. new 신호는 암호를 맨 처음 받을 때는 별을 눌러야 받는 것이 끝나고 재 설정 시에는 샵을 눌러야 끝나는 것을 바꾸기 위한 신호 이다. over\_flow 스테이트는 숫자가 7개 들어왔을 때 상태 이다.



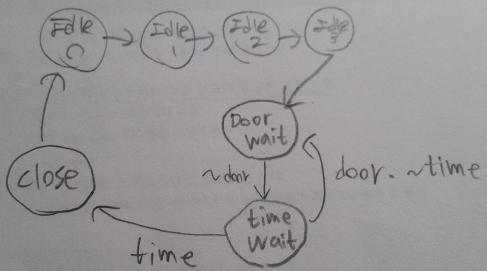
set\_check state 의 내부 스테이트

PRE 스테이트는 암호가 적절한지 체크하기 전에 미리 첫 번째 자리의 암호를 다른 래지스터에 받아 논다. div는 암호의 길이 별로 스테이트를 나누어 준다. OK와 ERR은 적절한 것, 부 적절한 것을 나누어 논 것이다.



password\_check state 내부 스테이트

I와 wait 스테이트를 통하여 입력하는 암호를 다른 레지스터에 받는다. I0, I1, I2에서 별을 눌러 입력을 종료하면 길이가 너무 짧아 바로 ERR스테이트로 간다. PRE스테이트 들은 암호가 맞는지 안 맞는지를 확인하여 OK로 갈지 ERR로 갈지 결정해서 보낸다.



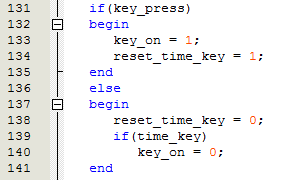
open state 내부 스테이트

door wait에서는 문이 열린 상태가 닫힐 때까지 기다리는 스테이트이다. 이 스테이트에서는 항상 타임을 리셋 시킨다. 문이 닫히면 time wait 스테이트로 넘어가고 이 스테이트는 시간이 지날 때까지 기다린다. time은 3초뒤 set 된다. 이 때 문이 열리면 다시 door wait 스테이트로 넘어가 타임이 리셋 된다.

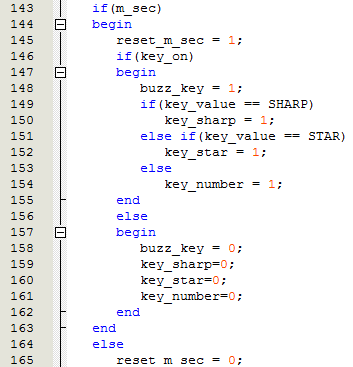
**4. 모듈 별 동작 설명**

**● key\_pad module**

수업 자료에 있는 키 패드 모듈에서 두 가지를 추가하였다. 우선 키를 누르고 있을 때 key\_row 신호가 100010001000100 이런 식으로 들어오게 되는데 이것을 1111111111111으로 바꾸어 주기 위하여 key\_press, key\_on 이란 레지스터를 활용했다. key\_press는 10001000 이런 식으로 key\_row가 1일때만 1이되는 신호이고 key\_on은 111111111 이렇게 출력하도록 만들어준 신호이다.



코드에서처럼 key\_press가 1이면 key\_on은 무조건 1이되고 key\_press가 0일떄는 key\_press가 1에서 0으로 바뀌고 어느 정도의 시간이 지나야 key\_on 이 0이 되도록 했다. time\_key는 key\_press가 10001000100 이이렇 나올 때 10001의 1사이의 간격 정도의 시간이다.



처음에는 이 코드 없이 작성을 하였는데 키 패드 모듈이 메인 모듈로 보내주는 키 신호가 불안정 하여 이 코드를 추가하였다. 메인 모듈로 보내는 키 신호를 1/100초 마다 한번씩만 바뀌도록 하였다. m\_sec가 1/100초 이다.

key\_number, key\_sharp, key\_star는 숫자, 샵, 별이 눌렸을 때 1이 되는 신호 이다.

**● Main FSM module**

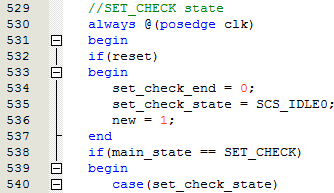
**암호 입력 방식**

처음/리셋 후 암호 설정 시: [“숫자 3~6자리” + “\*”] 적절하면 close, 부적절하면 다시 숫자 받음

암호 입력: (close상태에서) [“\*” + “숫자” + “\*”] 맞으면 open, 틀리면 close

암호 변경: (close상태에서) [“#” + “숫자” + “\*” + (암호 맞다면) + “숫자” + “#”]

위에서 설명한 것처럼 메인 스테이트의 각 스테이트마다 내부 스테이트를 두었는데 작동 방식을 예를 들어 설명하겠다. 만약에 메인 스테이트가 SET\_CHECK state 이라면 다른 내부 스테이트 들은 멈춰 있고, SET\_ CHECK 스테이트의 내부 스테이트만 작동 해야 된다. 따라서 아래 코드처럼 작성하였다.



다른 모든 내부 스테이트 들도 메인 스테이트가 해당 스테이트 일 때만 작동하도록 하였다.

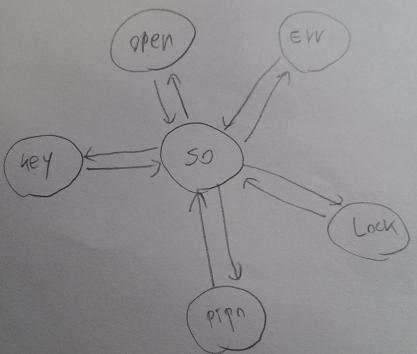
메인 스테이트의 상태 전이를 위한 always문을 아래쪽에 따로 만들었고, 메인 스테이트와 몇 가지 상태 값들로 아웃풋을 신호들을 결정하는 always문도 만들었다.

아웃풋 신호는 7-segment controller로 display\_state, display\_on과 i0~i5 값들은 보냈다. display\_state는 어떤 글자/숫자를 표시할지 정하는 신호이고 display\_on은 표시를 할지 말지를 정하는 신호이다. i0~i5값들은 표시할 숫자 값 들이다. led controller로는 led\_on신호를 보내고 led\_end 신호를 받도록 되있다. led\_on신호를 보내면 led가 켜지고 led 작동이 멈춘 후 led\_end 신호를 다시 메인 모듈로 보낸다. 하지만 FPGA 동작 시 작동이 안됐다. buzzer controller로 buzzer\_state신호를 보낸다. 어떤 부저음이 울리도록 할 것 인지를 정하는 신호이다.

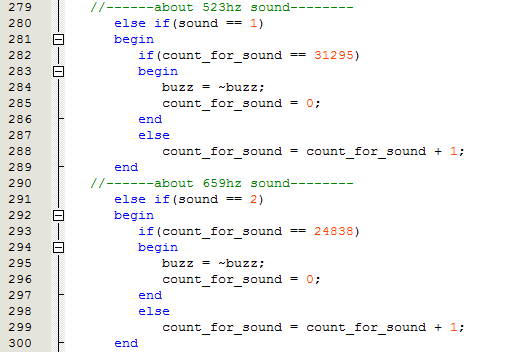
**● LED control module**

led\_on 신호를 받기 전까지는 쉬고 있다가 받으면 led를 순서대로 작동시켜 led 작동이 끝나면 led\_end 신호를 메인 모듈로 보낸다. 내부 카운터를 두어 1/3초마다 led가 켜지면서 동작하도록 설계 하였다. 하지만 동작하지 않았다.

**● Buzzer control module**



부저 컨트롤러의 스테이트는 대략 저렇다. S0에서 시작해서 key를 누르면 key 스테이트로 넘어가 소리를 내고 다시 S0로 돌아온다. 문이 열리면 open 스테이트로 넘어가 소리를 내고 S0로 돌아오고, 문이 닫히면 ERR 스테이트로 넘어가 소리를 내고 돌아온다. lock은 1분 잠겻을 때 소리를 내고 pipo는 3번이상 부적절한 암호 입력 시 소리를 낸다. 한 소리가 나고 있는 동안에는 다른 소리는 나지 않고 S0로 돌아왓을때 소리를 낸다. open, err, lock, pipo 스테이트에는 내부 스테이트가 있어서 0.2초마다 다른 음을 낸다.



윗 코드처럼 일정한 소리를 만들어 냈다.



sound의 상태에 따라 나는 음들을 만들어 놓았다.

**스테이트 별 소리**

key 🡪 기준 도 (0.2초)

open 🡪 미 솔 도 (각각 0.2초)

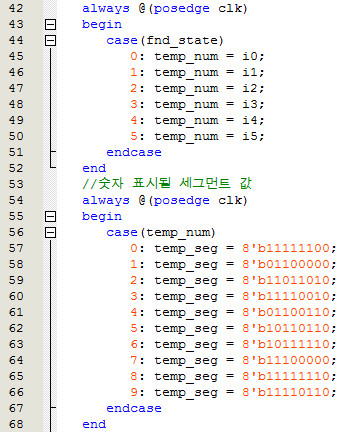
err 🡪 시 라# 솔 (각각 0.2초)

lock 🡪 솔# 솔 파# 솔 반복 (각각 0,2초)

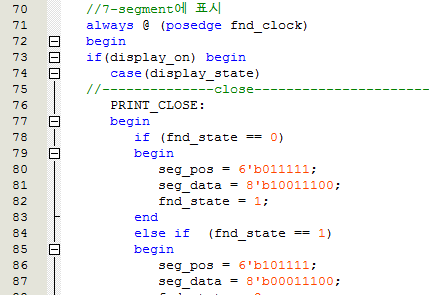
pipo 🡪 high 미 high 솔 반복 (각각 0.4초) [동작 안함…]

**● 7 – segment module**

이전에 수업에서 했던 방식 거의 그대로 만들었다. 추가한 것은 숫자 값 표시할 때 미리 숫자에 대응하는 세그먼트 값을 레지스트리에 저장해 놓는 것이다.

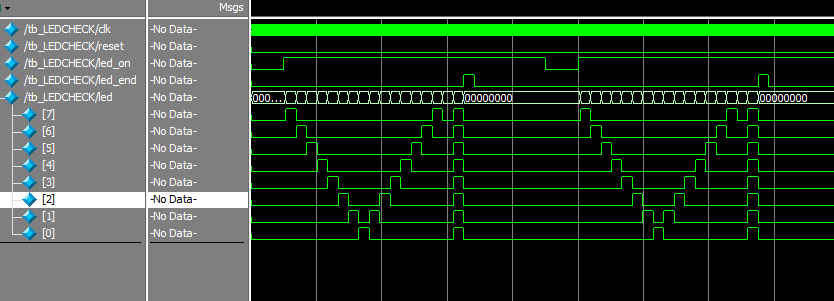


그 이외에는 아래와 같이 display\_state 신호를 받아 출력하게 하였다.



**5. 시뮬레이션 및 FPGA 동작 결과**

led control module만 시뮬레이션 해보았다.



시뮬레이션에서는 잘 작동 하는데 FPGA상에서 LED가 작동이 안되었다.

FPGA 동작 결과는 비밀번호를 오래 바꾸지 않았을 때 경고음 내는 것과, led 깜빡이면서 삐뽀삐뽀 하는 것 두 개를 제외하고는 잘 되었다.

**6. 결과 분석 및 토의, 고찰할 점**

수업 시간에 키 패드와 부저를 사용 했다면 조금 더 수월하게 만들 수 있었을 것 같다. 또한 프로젝트 세부 작동 기능이 조금 더 정확하면 좋겠다. 참고 자료는 김신덕 교수님 논리 회로 설계 수업 ppt 자료 일부분과, 대부분 디지털 논리 회로 수업 자료를 보고 만들었다.

**7. 시연 동영상**

별도 첨부 하였다.