2023학년도 1학기 기말평가		과목명	임베디드 소프트웨어 실습	제출자	서창민	
실습형				학년	3	3
				학번	19017054	19017012
담당교수	김 홍 규			이름	서창민	권시우

[확인]
기말평가는 시험 + 실습제출물 + 레포트가 합산됩니다.
팀구성 가능 : 최대 2인
제한 시간 : 2시간

● 기본구성요소 : TFT LCD, RFId Reader, RFID tag(Smart phone), EEPROM, Servo Motor, Buzzer

● 문제 : 아래의 시나리오에 따라 프로그램을 완성하시오.

● 시나리오

1. 도어 잠김 표시 및 서보모터

2. RFID 카드 등록(등록 방법 자유)

Buzzer 기능 : Tag를 갖다 대면 소리 자유 출력(사용자 정의)

Buzzer 기능 : EEPROM에 TAG 저장 후 소리 자유 출력

3. 미등록 태그(EEPROM에 없는 TAG 정보)

Buzzer 기능 : 소리 자유 출력

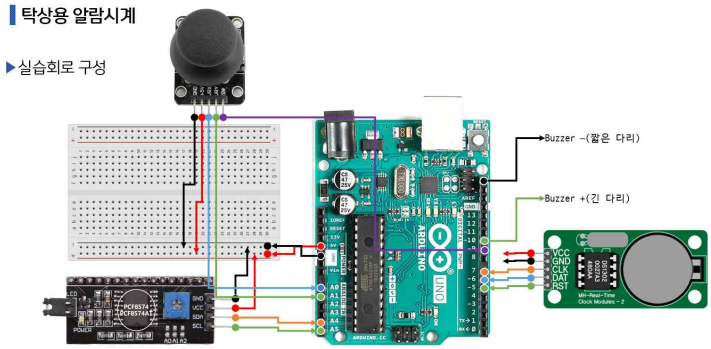
4. 등록 태그(EEPROM에 등록된 TAG 정보)

Buzzer 기능 : 소리 자유 출력

● 평가 요소 및 방법

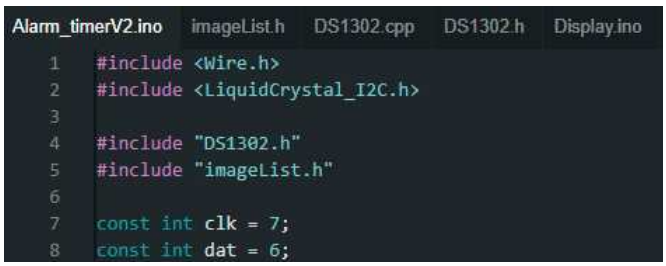
1. 문제의 시나리오에 의해 정상 동작 여부(30점)
2. 실습 회로 구성 표현 여부(4점)

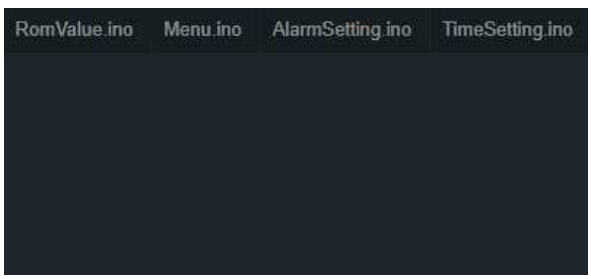
예)



2-1) 위와 같은 형식이 첨부되어 있는가.
2-2) 실제 구성된 사진이 포함되어 있는가.

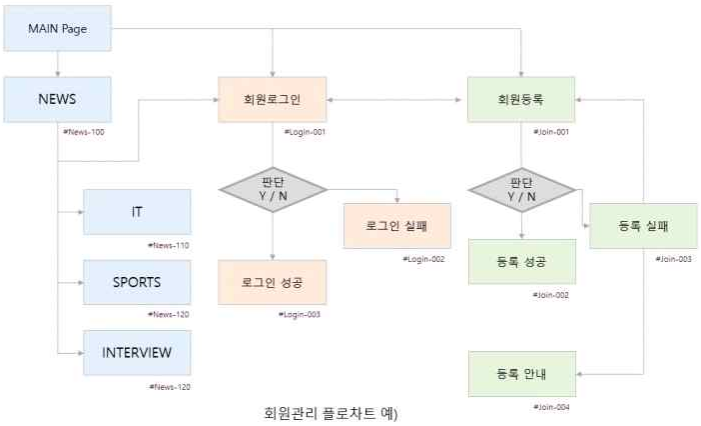
3. 소스 코드의 독립성(6점)
   - 프로그램 소스 코드를 하나의 파일로 작성하지 않고, 기능별, 모듈별로 나눠서 작성하였는가.





4. 소스 코드의 독창성(5점)
   - 수업시간에 작성된 코드 또는 인터넷 검색을 활용하되, 독창적인 형태(알고리즘, 프로그램 흐름 등)로 코드 구성이 되었는가.

6. 프로그램 플로차트가 포함되어 있는가.(5점)



회원관리 플로차트 예)

7. 자유기능이 추가 되어 있는가(40점).
   - 문제의 시나리오 외, 새로운 기능이 추가 되어 동작하는지에 대한 여부

   예시
   A. 사람이 TFT-LCD 앞 일정거리에 위치했을 때 TFT-LCD에 “Plz recognize the tag” 등의 문구 표시
   B. 인증된(EEPROM에 저장된) Tag인 경우 서보 모터를 약 30초간 열리고, 이후 자동으로 잠김
   C. 태그 등록 방법에 대한 창의성 포함

8. 기타(10점)
   - 새로 정의된 기능에 대한 상세 설명 포함 여부.
   - 소스코드 원본을 제출하였는가.
   - 소스코드 작성시 프로그램 작성 방법에 의해 작성되었는가.
   - 제출된 최종 작성 문서가 가독성 있게 작성되었는가.

[공통 사항]

* 제출방법
  1. 프로그램 소스코드를 포함한 제출하고자 하는 내용을 studyUploader로 제출하며, 16주차에 레포트로 대표 1인만 제출
  2. 현재의 이 문서에 포함된 자유양식(3페이지)부터 평가요소를 고려하여 작성한 후 pdf로 변환하여 대표 1인만 foxliver@naver.com로 제출

* 작성 문서 내에 소스 코드가 포함되어 있어야 함.
* 프로그램 작성시 수업자료 활용 가능하며, 인터넷 검색하여 코드 작성 가능.
* 인터넷 검색하여 참고한 소스 코드는 반드시 작성 문서 내에 출처를 명확히 기재해야 함.(인터넷인 경우 참고 사이트 주소 URL을 기재)
* 참고 사이트 기재로 인한 점수 불이익은 없으며, 반드시 기재.
* 실습 키트를 자유롭게 활용하여 기본 구성요소 외 다른 센서, 액추에이터를 추가 장착하여 시험 가능
* 코드 완료 후 반드시 각 기능에 따른 동작 사진이 첨부 되어 있어야 함. (시연 영상과 설명을 음성으로 하는 경우 사진 첨부 하지 않아도 인정, 시연 영상은 간단히 작성)

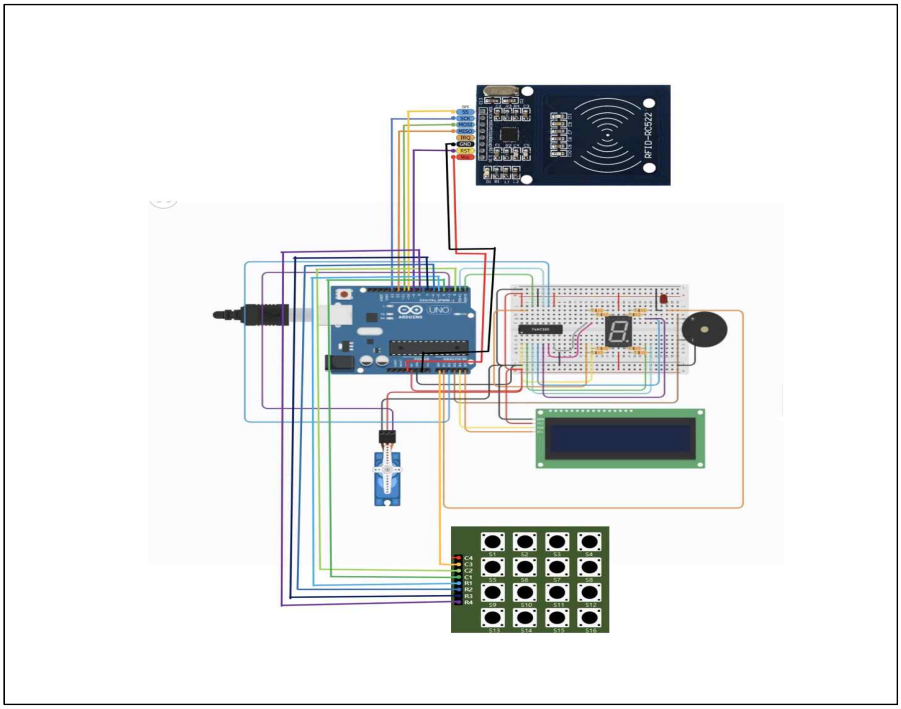
2023년 1학기 임베디드 소프트웨어 실습 실습형 기말평가(자유양식)

제품: 스마트 도어락 시스템

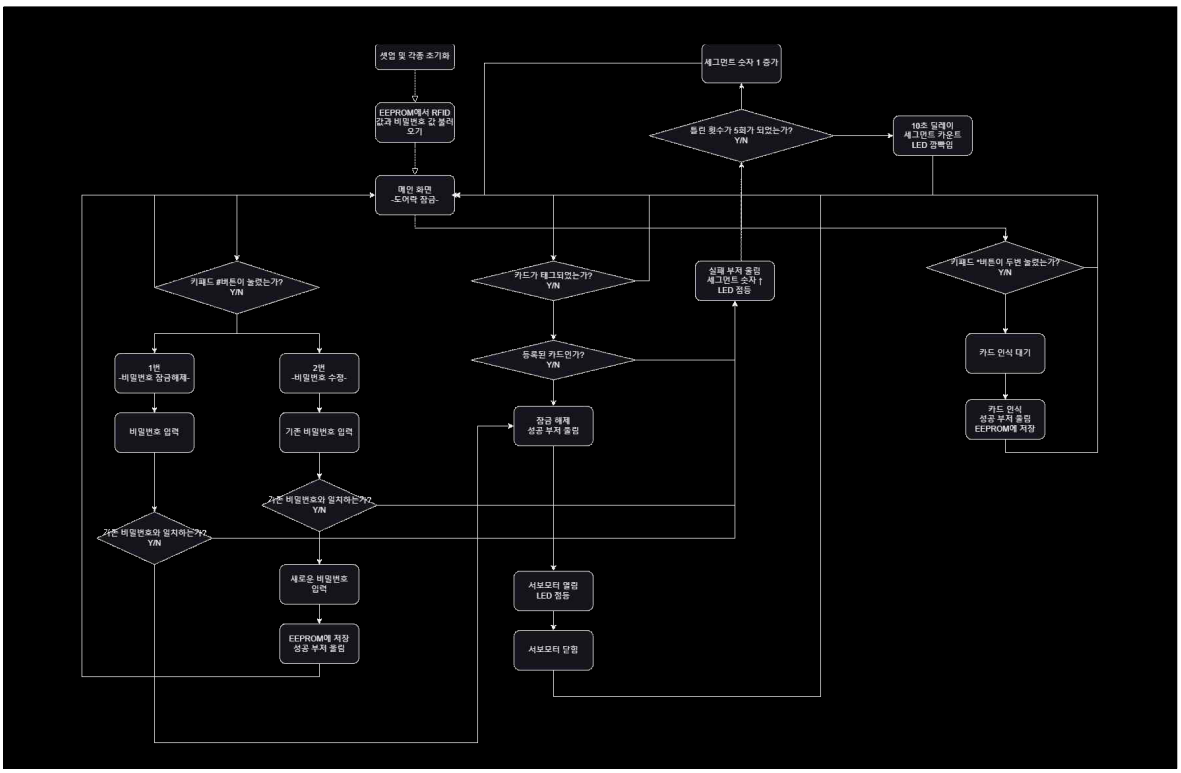
구성원: 19017054 서창민, 19017012 권시우

사용 부품 : TFT LCD, RFId Reader, RFID tag(Smart phone), EEPROM, Servo Motor, Buzzer, 7 Segment, 74HC595, Red LED, Key Matrix

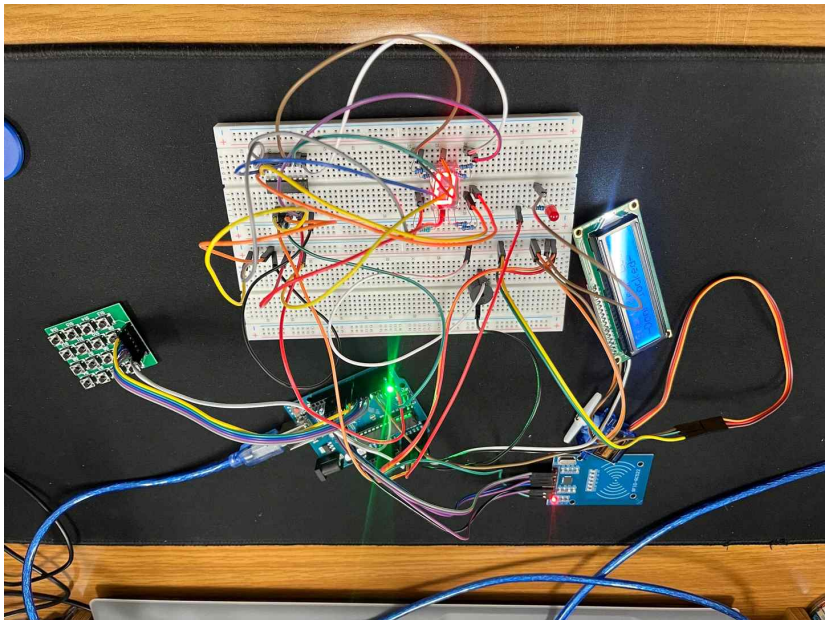
프로그램 회로도

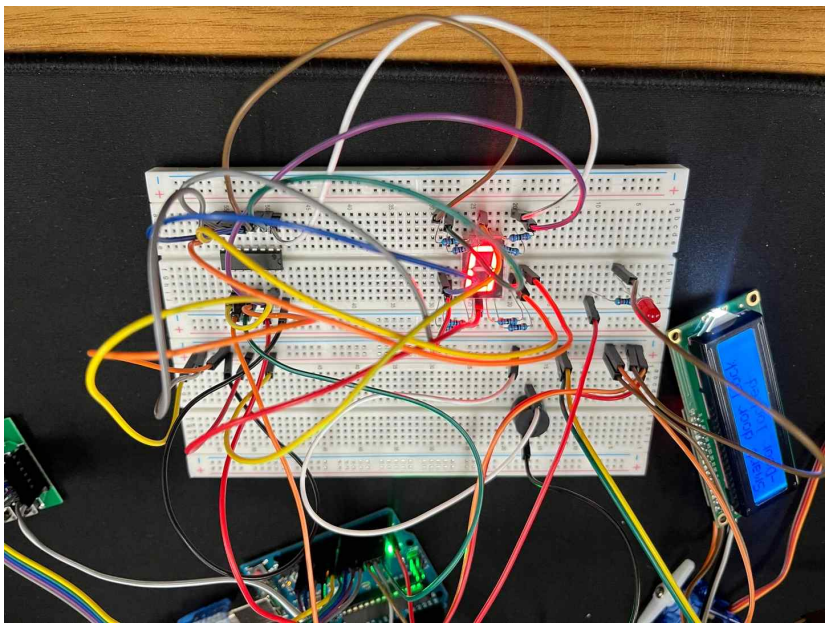


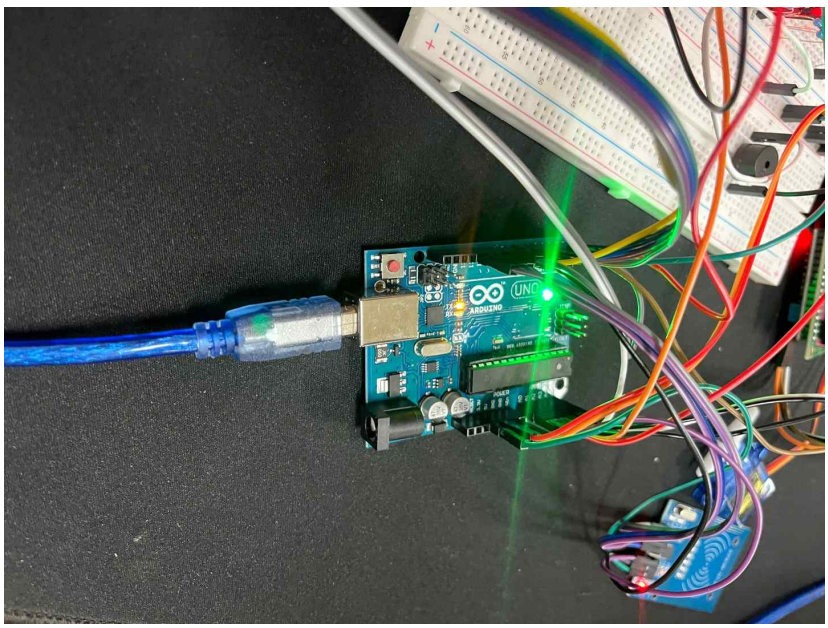
프로그램 플로차트



하드웨어 실제 구성도







코드 설명

report.ino

```
#include <SPI.h>
#include <MFRC522.h>
#include <EEPROM.h>
#include "Control_Pwd.h"
void setup() {
    Set_Matrix();//매트릭스 set함수
    Set_RFRC();//RFRC set함수
    Set_Lcd();//I2C_Lcd set함수
    Set_piezo();//piezo set함수
    Set_Servo();//Servo set함수
    Set_Segment();//7-Segment set함수
    pinMode(A1, OUTPUT);//A1 연결된 LED pinMode 설정
}
void loop() {
    switch(regi_cnt) {//regi_cnt 변수의 값에 따라 실행되는 함수가 달라짐
    case 0:
        Show_Lock_Door();//Lcd 화면 출력
        Open_Door();//RFRC 인식이 되면 등록된 태그인지 검사 후 결과 출력
        Set_RFRC_Pwd();//* 또는 #버튼 인식 및 눌림 횟수 저장
        break;
    case 1:
        Sign_up();//Lcd 화면 출력
        Save_EEPROM();//태그된 RFID EEPROM에 저장 후 regi_cnt = 2로 변경
        break;
    case 2:
        Success_sign();//Lcd 화면 출력
        Success_Regi();//성공부저울림
regi_cnt = 0;//regi_cnt 초기화하여 처음 화면으로 되돌아감
        break;
    case 3:
        Change_pwd();//#버튼이 2번 눌리면 regi_cnt가 3으로 변경됨, 해당 함수 실행
        break;
    case 4:
        Delay_10sec();//비밀번호 & RFID Tag 5회 틀리면 10초동안 잠금화면 출력하는 함수
regi_cnt = 0;
        break;
    case 5:
        Show_Enter_pwd();//Lcd 화면 출력
regi_cnt = 6;//출력하자마자 regi_cnt 6으로 변경
        break;
    case 6:
        Enter_pwd();//비밀번호 입력 값이 맞으면 잠금해제, 아니면 오류 뜨게 하는 함수
        break;
}
}
```

7Segment.h

```
#define dataP   0
#define latchP  1
#define clockPA2
byte digit[11] = {//세그먼트 0~.까지 바이트를 나타냄
B00010001,
B11010111,
B00110010,
B10010010,
B11010100,
B10011000,
B00011000,
B11010001,
B00010000,
B10010000,
B11101111
};


void Set_Segment(){//74HC595 set 함수
  pinMode(latchP, OUTPUT);
  pinMode(clockP, OUTPUT);
  pinMode(dataP, OUTPUT);
  digitalWrite(latchP, LOW);
}
void Show_Segment(inti){//7segment 출력 함수
  digitalWrite(latchP, LOW);
  shiftOut(dataP, clockP, LSBFIRST, digit[i]);
  digitalWrite(latchP, HIGH);
}
```

Control_Pwd.h

```
#include "Sound.h"
#include "KeyMatrix.h"
#include "LCD.h"
#include "Motor.h"
#define SS_PIN 10
#define RST_PIN 9
MFRC522 rfidReader(SS_PIN, RST_PIN);
MFRC522::MIFARE_Key key;
byte readId[4];
byte regiId[4];
void Open_Door();
void Sel_Regi();
void Save_EEPROM();
void Set_RFRC(){
  SPI.begin();
  rfidReader.PCD_Init();
}
intx = 1;
inty = 2;
void Open_Door(){//RFID 인식 후 등록된 태그 몇 잠금해제 아니면 오류 출력 함수
  Show_Segment(fail_cnt);//7세그먼트에 실패 횟수를 띄움(초깃값 0)
  if(!rfidReader.PICC_IsNewCardPresent()) return;
  if(!rfidReader.PICC_ReadCardSerial()) return;
  for(inti = 0; i < 4; i++){//EEPROM에 저장된 RAID Tag값을 regiId배열에 저장
byte hiByte = EEPROM.read(x);
byte loByte = EEPROM.read(y);
    regiId[i] = word(hiByte, loByte);
x+=2;
y+=2;
}
  if(rfidReader.uid.uidByte[0] == regiId[0]//인식된 Tag 값과 regiId값이 같으면
      && rfidReader.uid.uidByte[1] == regiId[1]
      && rfidReader.uid.uidByte[2] == regiId[2]
      && rfidReader.uid.uidByte[3] == regiId[3]){
    Show_Open_Door();//열린다는 문구 Lcd에 출력
    digitalWrite(A1, HIGH);//LED ON
    Sound_Unlock();//열린다는 소리 부저 울림
    Unlock_Servo();//서보 모터 열림
fail_cnt = 0;//실패 횟수 초기화
    delay(3000);//3초동안 유지
    Sound_Lock();//닫힌다는 소리 부저 울림
    Lock_Servo();//서보 모터 닫힘
    digitalWrite(A1, LOW);//LED OFF
}
  else{
    Show_Unau_Door();//태그가 서로 다르다면 오류 문구 Lcd 출력
```

```
    digitalWrite(A1, HIGH);//LED ON
    Sound_Fail();//실패 소리 부저 울림
    digitalWrite(A1, LOW);//LED OFF
fail_cnt++;//실패횟수 1 증가
    Show_Segment(fail_cnt);//7-세그먼트에 실패횟수 출력
    if(fail_cnt == 5){//실패횟수가 5회가 된다면
regi_cnt = 4;//regi_cnt = 4 로 변경 -> Delay_10sec()함수 (10초동안 잠금)
fail_cnt = 0;//실패횟수 초기화
}
}
x=1;
y=2;
}
void Save_EEPROM(){
  if(!rfidReader.PICC_IsNewCardPresent()) return;
  if(!rfidReader.PICC_ReadCardSerial()) return;
MFRC522::PICC_Type picc_type = rfidReader.PICC_GetType(rfidReader.uid.sak);
  if(picc_type != MFRC522::PICC_TYPE_MIFARE_MINI
&& picc_type != MFRC522::PICC_TYPE_MIFARE_1K
&& picc_type != MFRC522::PICC_TYPE_MIFARE_4K
&& picc_type != MFRC522::PICC_TYPE_ISO_14443_4){
    return;
}
  if(rfidReader.uid.uidByte[0] != readId[0]//RFRC가 인식이 된다면
|| rfidReader.uid.uidByte[1] != readId[1]
|| rfidReader.uid.uidByte[2] != readId[2]
|| rfidReader.uid.uidByte[3] != readId[3]){
    Tag_RFRC();
    for(inti = 0;i < rfidReader.uid.size; i++){
byte hiByte = highByte(rfidReader.uid.uidByte[i]);
byte loByte = lowByte(rfidReader.uid.uidByte[i]);
      EEPROM.write(x, hiByte);
      EEPROM.write(y, loByte);
2;
2;
}
}
  rfidReader.PICC_HaltA();
  rfidReader.PCD_StopCrypto1();
regi_cnt = 2;//저장이 완료되면 regi_cnt = 2 -> 성공알림부저 및 Lcd 출력하는 곳으로 이동하게끔 함
push_cnt = 0;//해당 함수로 들어오려면 *클릭횟수(push_cnt)가 2가 되야하므로 등록이 완료되면 초기화
fail_cnt = 0;
x = 1;//EEPROM 주소 초기화
y = 2;
}
int pwd[4];
int check_pwd[4];
```

```
int match_pwd[4];
int new_pwd[4];
intstatus_cnt = 0;
intpwd_cnt = 0;
intlcd_cnt = 11;
intnew_cnt = 0;
inta = 9;
intb = 10;
void Change_pwd(){//비밀번호 변경 함수
  switch(status_cnt){//status_cnt에 따라 달라짐
  case 0://기본값 0
    Lcd.clear();//Lcd 출력하자마자
    Lcd.setCursor(0,0);//status_cnt = 1, case 1로 넘어감
    Lcd.print("Change Password");
    Lcd.setCursor(0,1);
    Lcd.print("Enter PWD: ");
status_cnt = 1;
    break;
  case 1:
    Lcd.setCursor(lcd_cnt,1);//비밀번호 입력하는 위치
    for(intj = 0; j < numCols; j++){
      digitalWrite(pinCols[j], LOW);
      for(inti = 0; i < numRows; i++){
        if(digitalRead(pinRows[i]) == LOW){
          check_pwd[pwd_cnt] = numpad[i][j].toInt();//numpad는 String배열이므로 toInt()로 int형으
로 전환
          Lcd.print(check_pwd[pwd_cnt]);//입력된 숫자를 check_pwd배열에 넣은 후 Lcd에 출력
lcd_cnt++;//lcd 커서 위치 1 증가
pwd_cnt++;//배열 위치 1 증가
}
}
      if(pwd_cnt == 4){//배열 위치가 4가 된다면
pwd_cnt = 0;//0으로 초기화
status_cnt = 2;//status_cnt = 2 로 변경, case 2로 이동하게끔 함
}
      digitalWrite(pinCols[j], HIGH);
}
    delay(250);
    break;
  case 2:
    for(inti = 0; i < 4; i++){//EEPROM에 저장되어 있는 비밀번호 값을
byte hibyte = EEPROM.read(a);//match_pwd배열에 저장
byte lobyte = EEPROM.read(b);
      match_pwd[i] = word(hibyte, lobyte);
a+=2;
b+=2;
}
```

```
a = 9;
b = 10;
    if(match_pwd[0] != check_pwd[0]//입력된 비밀번호와 EEPROM에 저장된 값과 다르면
|| match_pwd[1] != check_pwd[1]
|| match_pwd[2] != check_pwd[2]
|| match_pwd[3] != check_pwd[3]){
      Lcd.clear();
      Lcd.setCursor(0,0);
      Lcd.print("Not match");//매치가 안된다는 문구 lcd 출력
      digitalWrite(A1, HIGH);//LED ON
      Sound_Fail();//실패부저울림
      digitalWrite(A1, LOW);//LED OFF
fail_cnt++;//실패횟수 1 증가
regi_cnt = 0;//초기 화면으로 가기 위해 regi_cnt 초기화
status_cnt = 0;//status_cnt도 초기화
pwd_cnt = 0;//입력된 비밀번호 배열의 위치도 초기화
lcd_cnt = 11;//lcd 커서 설정
push_cnt2 = 0;//비밀번호를 변경하려면 #클릭횟수(push_cnt2)가 2가 되야했으므로, push_cnt2 초기화
      if(fail_cnt == 5){//실패횟수가 5회가 되면
regi_cnt = 4;//regi_cnt = 4 로 해당하는 case문으로 이동하게끔 함
fail_cnt = 0;//실패횟수 초기화
push_cnt2 = 0;//눌림횟수 초기화
}
}
    else if(match_pwd[0] == check_pwd[0]//입력한 값과 저장된 값이 같으면
      && match_pwd[1] == check_pwd[1]
      && match_pwd[2] == check_pwd[2]
      && match_pwd[3] == check_pwd[3]){
lcd_cnt = 9;//lcd 커서 설정
status_cnt = 3;//case 3으로 이동하게끔 함
}
    break;
  case 3:
    Lcd.clear();
    Lcd.setCursor(0,0);
    Lcd.print("Change Password");
    Lcd.setCursor(0,1);
    Lcd.print("New PWD: ");
    Lcd.setCursor(lcd_cnt,1);
status_cnt = 4;//Lcd 출력하자마자 case 4문으로 이동하게끔 함
    break;
  case 4:
    for(intj = 0; j < numCols; j++){
      digitalWrite(pinCols[j], LOW);
      for(inti = 0; i < numRows; i++){
        if(digitalRead(pinRows[i]) == LOW){
          new_pwd[new_cnt] = numpad[i][j].toInt();//입력받은 값은 new_pwd배열에 저장
```

```
          Lcd.print(new_pwd[new_cnt]);
byte hibyte = highByte(new_pwd[new_cnt]);//new_pwd에 있는 값을 EEPROM에 저장
byte lobyte = lowByte(new_pwd[new_cnt]);
          EEPROM.write(a, hibyte);
          EEPROM.write(b, lobyte);
new_cnt++;//new_pwd 배열 위치 1 증가
lcd_cnt++;//lcd 커서 위치 1 증가
a+=2;//EEPROM 주소 증가
b+=2;
}
}
      if(new_cnt == 4){//배열 끝에 도달하면
new_cnt = 0;//초기화
status_cnt = 5;//case 5문으로 이동
}
      digitalWrite(pinCols[j], HIGH);
}
    delay(250);
    break;
  case 5:
    Lcd.clear();
    Lcd.setCursor(0,0);
    Lcd.print("Success Change!");//성공 문구 Lcd 출력
    Success_Regi();//성공부저울림
lcd_cnt = 11;//각종 변수들 초깃값으로 초기화
regi_cnt = 0;
status_cnt = 0;
new_cnt = 0;
push_cnt2 = 0;
fail_cnt = 0;
x = 1;
y = 2;
}
}
int enter_pwd[4];
int save_pwd[4];
intenter_cnt = 0;
void Enter_pwd(){//비밀번호로 잠금해제하는 함수
  for(inti = 0; i < 4; i++){//EEPROM에 저장된 비밀번호 값을 불러와 save_pwd배열에 저장
byte hibyte = EEPROM.read(x);
byte lobyte = EEPROM.read(y);
    save_pwd[i] = word(hibyte, lobyte);
x+=2;
y+=2;
}
x = 9;//EEPROM 주소 초기화
y = 10;
```

```
  Lcd.setCursor(0, 1);
  Lcd.print("Enter PWD: ");
  for(intj = 0; j < numCols; j++){
    digitalWrite(pinCols[j], LOW);
    for(inti = 0; i < numRows; i++){
      if(digitalRead(pinRows[i]) == LOW){
        if(numpad[i][j] == "#"){//만약 #이 한번 더 눌린다면 push_cnt2 1 증가 -> push_cnt2가 2가 되
므로
push_cnt2++;//비밀번호 변경 함수로 넘어감
regi_cnt = 3;//loop문 case 3으로 넘어가게끔 함
}
       else{
          enter_pwd[enter_cnt] = numpad[i][j].toInt();//#이 아닌 다른 버튼이 눌렸다면 enter_pwd에
입력받은 값 저장
         Lcd.setCursor(enter_cnt + 11, 1);
         Lcd.print(enter_pwd[enter_cnt]);
enter_cnt++;
}
}
}
    digitalWrite(pinCols[j], HIGH);
    if(enter_cnt == 4){//입력이 끝나면
enter_cnt = 0;
      if(enter_pwd[0] != save_pwd[0]//만약 입력한 값과 저장된 값이 다르다면
|| enter_pwd[1] != save_pwd[1]
|| enter_pwd[2] != save_pwd[2]
|| enter_pwd[3] != save_pwd[3]){
       Lcd.clear();
       Lcd.setCursor(0, 0);
       Lcd.print("Smart door Lock");//오류 문구 Lcd 출력
       Lcd.setCursor(0, 1);
       Lcd.print("Wrong Password");
       digitalWrite(A1, HIGH);//LED ON
       Sound_Fail();//실패부저울림
       digitalWrite(A1, LOW);//LED OFF
fail_cnt++;//실패횟수 1 증가
regi_cnt = 0;//초기화면으로 넘어가기 위해 초기화
push_cnt2 = 0;//#클릭횟수 변수 초기화
       if(fail_cnt == 5){//실패횟수가 5회가 된다면
regi_cnt = 4;//loop문 case 4로 넘어가게끔 함
fail_cnt = 0;//실패횟수 초기화
push_cnt2 = 0;//#클릭횟수 변수 초기화
}
}
      else if(enter_pwd[0] == save_pwd[0]//만약 입력값과 저장된 값이 같으면
         && enter_pwd[1] == save_pwd[1]
         && enter_pwd[2] == save_pwd[2]
```

```
          && enter_pwd[3] == save_pwd[3]){
        Show_Open_Door();//열렸다는 문구 Lcd 출력
        digitalWrite(A1, HIGH);//LED ON
        Sound_Unlock();//열림부저울림
        Unlock_Servo();//서보모터 열림
fail_cnt = 0;//실패횟수 초기화
        delay(3000);//3초 유지
        Sound_Lock();//닫힘부저울림
        Lock_Servo();//서보모터 닫힘
        digitalWrite(A1, LOW);//LED OFF
push_cnt2 = 0;//#클릭횟수 변수 초기화
regi_cnt = 0;//초기화면으로 넘어가기 위해 초기화
}
}
}
  delay(250);
}
```

KeyMatrix.h

```
const intnumRows = 4;
const intnumCols = 3;
int pinRows[numRows] = {5, 6, 7, 8};
int pinCols[numCols] = {4, 2, A0};
intKey_result;
intregi_cnt = 0;
intpush_cnt = 0;
intpush_cnt2 = 0;
intfail_cnt = 0;
String numpad[4][3] = {//키패드
{"1", "2", "3"},
{"4", "5", "6"},
{"7", "8", "9"},
{"*", "0", "#"}
};
void Set_Matrix(){//매트릭스 set함수
  for(inti = 0; i < numRows; i++){
    pinMode(pinRows[i], INPUT_PULLUP);
}
  for(intj = 0; j < numCols; j++){
    pinMode(pinCols[j], OUTPUT);
    digitalWrite(pinCols[j], HIGH);
}
}
void Set_RFRC_Pwd(){//RFRC 등록하기 위해 *클릭 두번, 비밀번호 입력 #클릭 한번, 비밀번호 변경 #클릭 두
번
  for(intj = 0; j < numCols; j++){
    digitalWrite(pinCols[j], LOW);
    for(inti = 0; i < numRows; i++){
      if(digitalRead(pinRows[i]) == LOW){
        if(numpad[i][j] == "*") push_cnt++;//*이 눌리면 push_cnt 증가
        else if(numpad[i][j] == "#") push_cnt2++;//#이 눌리면 push_cnt2 증가
}
}
    digitalWrite(pinCols[j], HIGH);
}
    if(push_cnt == 2){//*이 2번 입력되면 RFRC등록으로 넘어가기
regi_cnt = 1;
push_cnt = 0;
push_cnt2 = 0;
}
    if(push_cnt2 == 1){//#이 1번 입력되면 비밀번호 입력으로 넘어가기
regi_cnt = 5;
push_cnt = 0;
}
    if(push_cnt2 == 2){//#이 2번 입력되면 비밀번호 변경으로 넘어가기
```

```
regi_cnt = 3;
push_cnt = 0;
push_cnt2 = 0;
}
  delay(250);
}
```

LCD.h

```
#include <LiquidCrystal_I2C.h>
#include "7Segment.h"
LiquidCrystal_I2C Lcd(0x27, 16, 2);
char buffer[3];
void Set_Lcd(){
  Lcd.init();
  Lcd.backlight();
}
void Show_Lock_Door(){//초기 화면 출력
  Lcd.clear();
  Lcd.setCursor(0, 0);
  Lcd.print("Smart door Lock");
  Lcd.setCursor(0, 1);
  Lcd.print("-Door locked-");
}
void Show_Open_Door(){//열렸을때 화면 출력
  Lcd.clear();
  Lcd.setCursor(0, 0);
  Lcd.print("Smart door Lock");
  Lcd.setCursor(0, 1);
  Lcd.print("-Door unlocked-");
}
void Show_Unau_Door(){//태그값이 다를때 화면 출력
  Lcd.clear();
  Lcd.setCursor(0, 0);
  Lcd.print("Smart door Lock");
  Lcd.setCursor(0, 1);
  Lcd.print("Unauthorized tag");
}
void Show_Enter_pwd(){//비밀번호 값 입력할때 화면 출력
  Lcd.clear();
  Lcd.setCursor(0, 0);
  Lcd.print("Smart door Lock");
}
void Sign_up(){//태그 인식 대기할때 화면 출력
  Lcd.clear();
  Lcd.setCursor(0, 0);
  Lcd.print("TAG Register");
  Lcd.setCursor(0, 1);
  Lcd.print("STEP.1:tagging");
  delay(250);
}
void Success_sign(){//인식 성공 후 저장했을때 화면 출력
  Lcd.clear();
  Lcd.setCursor(0, 0);
  Lcd.print("TAG Register");
```

```
  Lcd.setCursor(0, 1);
  Lcd.print("STEP.2:SAVE OK");
}
void Delay_10sec(){//10초 잠금할때 화면 출력
  Lcd.clear();
  Lcd.setCursor(0, 0);
  Lcd.print("5 Failed Unlock");
  for(inti = 10; i > 0; i--){
    Lcd.setCursor(0, 1);
    sprintf(buffer, "%02d", i);
    Lcd.print(buffer);
    Lcd.setCursor(2, 1);
    Lcd.print(" second Left");
    Show_Segment(i);
    digitalWrite(A1, HIGH);
    delay(500);
    digitalWrite(A1, LOW);
    delay(500);
}
}
```

Motor.h

```
#include <Servo.h>
#define servoPin  3
Servo myServo;
void Set_Servo() {
  myServo.attach(servoPin);
  myServo.write(0);
}
void Unlock_Servo() {//서보모터 열림
  for(intangle = 0; angle < 91; angle++){
    myServo.write(angle);
    delay(15);
}
}
void Lock_Servo(){//서보모터 닫힘
  for(intangle = 91; angle > 0;angle--){
    myServo.write(angle);
    delay(15);
}
}
```

Sound.h

```
#define piezoA3
intSuccess_tones[] = {659, 784, 880, 784, 880, 988, 880, 988, 1047, 784, 659, 784, 659, 587,
523};
intUnlock_tones[] = {731, 747, 850};
intLock_tones[] = {850, 747, 431};
intFail_tones[] = {800, 500, 800, 500};
void Set_piezo(){
  pinMode(piezo, OUTPUT);
}
void Tag_RFRC(){//RFRC 인식되면 소리나게 하는 함수
  tone(piezo, 1000);
  delay(1000);
  noTone(piezo);
}
void Success_Regi(){//인식 성공후 태그 저장 시 소리나게 하는 함수
  for(inti = 0; i < 15; i++){
    tone(piezo, Success_tones[i]);
    delay(200);
}
  noTone(piezo);
}
void Sound_Unlock(){//잠금해제 되었을 때 소리나게 하는 함수
  for(inti = 0; i < 3; i++){
    tone(piezo, Unlock_tones[i]);
    delay(200);
}
  noTone(piezo);
}
void Sound_Lock(){//잠겼을 때 소리나게 하는 함수
  for(inti = 0; i < 3; i++){
    tone(piezo, Lock_tones[i]);
    delay(200);
}
  noTone(piezo);
}
void Sound_Fail(){//비밀번호가 다르거나 태그의 값이 다르면 소리나게 하는 함수
  for(inti = 0; i < 4; i++){
    tone(piezo, Fail_tones[i]);
    delay(100);
}
  noTone(piezo);
}
```

reset.ino

```
#include <EEPROM.h>
intdefault_pwd = 0;
intx = 9;
inty = 10;
inta = 1;
intb = 2;
//EEPROM에 저장된 값들을 전부 0으로 초기화하는 함수
void setup() {
  for(inti = 0; i < 4; i++){
byte hibyte = highByte(default_pwd);
byte lobyte = lowByte(default_pwd);
    EEPROM.write(x, hibyte);
    EEPROM.write(y, lobyte);
x += 2;
y += 2;
}
  for(inti = 0; i < 4; i++){
byte hibyte = highByte(default_pwd);
byte lobyte = lowByte(default_pwd);
    EEPROM.write(x, hibyte);
    EEPROM.write(y, lobyte);
a += 2;
b += 2;
}
}
void loop() {
// put your main code here, to run repeatedly:
}
```

추가 사항

추가 부품: Key Matrix, 7-Segment, 74HC595, LED

추가 내용 설명

Key Matrix

-RFID Tag 외에도 비밀번호로도 도어락을 잠금 해제 가능하도록 하기 위함
-*두번 클릭하면 RFID Tag를 등록하게끔 작동시켰음
-#한번 클릭하면 비밀번호를 입력하여 도어락을 잠금 해제하도록 하였음
-#두번 클릭하면 기존 비밀번호를 입력한 후 새로운 비밀번호를 등록할 수 있게끔 하였음
-기본적으로 스트링 배열로 선언하였지만, 비밀번호를 입력받거나 EEPROM에 저장할 때는 toInt()를 이용하여 int형으로 변환하여 저장하였음

7-Segment

-기본적인 값은 0을 출력하게 하였음
-해당 값은 오류 횟수이며 등록되지 않은 태그를 대거나 비밀번호 오류가 발생하면 숫자가 1씩 증가함
-해당 숫자가 5가 되면 10초 동안 작동이 중지되면 10초 카운트가 됨
-이 때 세그먼트는 .부터 0까지 카운트하게 되어있음

74HC595

-해당 부품은 세그먼트를 작동시키기 위해 장착하였으며 shiftOut함수를 사용하였음
-디지털 포트 부족으로 인해 시리얼 포트인 0,1번 아날로그 포트를 사용하였음

LED

-도어락 잠금해제 시도가 발생하거나 세그먼트 변화가 생기면 점등하게 만들었음
-잠금이 해제되거나 실패하면 점등을 유지함
-10초동안 작동이 중지될때는 1초 간격을 깜빡이게 하였음