**소스 코드**

#include <mega128.h>

#include <delay.h>

#include <mem.h>

#include <stdio.h>

#include <string.h>

/\*\*\*\* LCD Address \*/

#define LCD\_DATA PORTA //command, data,etc

#define LCD\_RS PORTD.0 //PORTD pin 0

#define LCD\_RW\_L PORTD.1 //pin 1

#define LCD\_ENABLE PORTD.2 //pin2

#define FUNCSET 0x38 // Function Set, 8 bit mode

#define ENTMODE 0x06 // Entry Mode Set

#define ALLCLR 0x01 // All Clear

#define DISPON 0x0f // Display On, cursor, blink

이 전자 도어 로크 프로그램에 대한 전역 변수이다.

void LCD\_init(void);

void LCD\_String(char flash \*);

void Busy(void);

void Command(unsigned char);

void Data(unsigned char);

unsigned char FONT[7] = {0x3F,0x73,0x79,0x37,0x39,0x38,0x6D};

unsigned char PASS[4] = {0x73,0x77,0x6D,0x6D};

unsigned char FLSE[4] = {0x71,0x38,0x6D,0x79};

unsigned char USER\_MODE=0, KEY[2]={0x00, 0x00}; //define button

unsigned char cnt=0; //marking switch 1-4

unsigned char led=0xFE; //LED 초기 출력값

unsigned long i,j,k;

unsigned long count=0;

unsigned long count2=0;

int FLAG=0,digit=0,FLAG\_PASS=1; //FLAG=open-close function, FLAG\_PASS=for loop, digit=count input

int buff[4]={0,0,0,0}; //record inputted switch value

static int KEY\_PASS[4]={1,2,2,1}; //secret pin

void Delay(unsigned int); //시간 지연 함수

// Timer 0 overflow interrupt service routine

interrupt [TIM0\_OVF] void timer0\_ovf\_isr(void)

{

if(FLAG\_PASS==1)

{

TCNT0=0x06 ; //timer counter 0 register

DDRE = 0x02 ;

KEY[0] = KEY[1];

KEY[1] = (PINE ^ 0xF0) & 0xF0; //switch button (on/off)

if((KEY[0]==0)&&(KEY[1]==0x10)) //KEY[0]=compare previous state and current state

{

cnt=1;

digit++;

}

\*) FLAG\_PASS 함수는 암호 함수에 대한 루프를 설정

\*) 버튼 키와 1-4 cnt (카운트로 선언 4 버튼)로 정의.

\*) digit ++ 입력 된 숫자를 계산하는 데 사용된다.

if((KEY[0]==0&&KEY[1]==0x20))

{

cnt=2;

digit++;

}

if((KEY[0]==0&&KEY[1]==0x40))

{

cnt=3;

digit++;

}

if((KEY[0]==0&&KEY[1]==0x80))

{

cnt=4;

digit++;

}

\*) 버프 기능이 KEY\_PASS (비밀 핀 배열)와 비교하여 입력 버튼을 확인하는 데 사용된다.

“Correct”= 입력 정확한 숫자를 계속 \*)

\*) “False” cnt는 버프 = 0 확인하고 입력 된 번호는 0과 동일

if(buff[3]!=KEY\_PASS[3]){

if(buff[2]!=KEY\_PASS[2]){

if(buff[1]!=KEY\_PASS[1]){

if(buff[0]!=KEY\_PASS[0]){

//buff is used to compare inputted button with KEY\_PASS

buff[0]=cnt; //insert buff value

cnt=0;

}

buff[1]=cnt;

cnt=0;

}

buff[2]=cnt;

cnt=0;

}

buff[3]=cnt;

cnt=0;

}

if(digit==4){ //finish input

if((buff[3]==KEY\_PASS[3])&&(buff[2]==KEY\_PASS[2])&&(buff[1]==KEY\_PASS[1])&&(buff[0]==KEY\_PASS[0])) //correct input

\*) (digit == 4) 입력 버튼에 해당하는 경우 그럼이 FLAG\_PASS을 중지하고 개폐 기능을 포함 "FLAG"계속된다는 것을 의미한다.

{

FLAG\_PASS=0; //stop loop

FLAG=1; //go to open close function

digit = 0; //reset digit

}

else //false input

{

buff[3]=0; //reset all buff to 0 , loop once more

buff[2]=0;

buff[1]=0;

buff[0]=0;

while(count<300) //마음대로

{

while(count2<80) //led blinks first

{

PORTC=led;

Delay(5); //blink delay , 5ms

led<<=1; //blink in order, led변수의 좌 쉬프트

led |=0x01; //쉬프트 후에 0 비트에는 '0'이 입력되므로 '1'로 셋

if(led==0xFF) led=0xFE; //led 변수 초기화

count2++;

}

PORTB = FLSE[0]; PORTF = 0x80; //PORTB=LED ON "F" (단어 출력), PORTF=SEG1 ON

delay\_ms(5);

PORTB = FLSE[1]; PORTF =0x40; //PORTB=LED ON "L" PORTF=SEG2 ON

delay\_ms(5);

PORTB = FLSE[2]; PORTF = 0x20; //PORTB=LED ON "S" PORTF=SEG3 ON

delay\_ms(5);

PORTB = FLSE[3]; PORTF = 0x10; //PORTB=LED ON "E" PORTF=SEG4 ON

delay\_ms(5);

count++;

}

PORTC=0xFF; //turn off

DDRC=0xFF;

PORTB =0x00; PORTF = 0x80;

for( i = 0; i<125 ; i++); //delay\_ms과 for 똑깥다

PORTB =0x00; PORTF = 0x40;

for( i = 0; i<125 ; i++);

Else 기능이 동일한 조건이 포함되는 거짓이면, 실행되는 코드 블록을 지정하는 데 사용된다. 'FLSE'(false)를 배열, 그것은 7 세그먼트에 표시된다. 버프 [3] ~ 버프 [0] 0 버튼을 다시 사용하고 루프가 한 번 더 할 것이다.

PORTB =0x00; PORTF = 0x20;

for(i = 0; i<125 ; i++);

PORTB =0x00; PORTF = 0x10;

for(i = 0; i<125 ; i++);

DDRB=0xFF;

digit=0;

count=0;

PORTF=0x00;

PORTB=0x00;

}

}

}

\*) FLAG function은 2 키를 포함하는 개폐 기능을 위해 루프를 설정한다. 각 키는 USER\_MODE 기능이 포함되어 있다.

if(FLAG==1) //Open Close Function

{

PORTF=0xF0;

PORTB=0xFF;

DDRB=0xFF;

KEY[0] = KEY[1];

KEY[1] = (PINE ^ 0xF0) & 0xF0;

if((USER\_MODE==0)|(USER\_MODE==2))

//When USER\_MODE=1 or USER\_MODE=2 can operate USER\_MODE=1

if((KEY[0]==0x00)&&(KEY[1]==0x10)) //switch 1 as Open Function

{

USER\_MODE=1; //USER\_MODE1=Open Fucntion

k=0;

}

if(USER\_MODE==1)

if((KEY[0]==0x00)&&(KEY[1]==0x20))

//When USER\_MODE=1 can operate USER\_MODE=2

{

USER\_MODE=2;

k=0; //USER\_MODE2=Close Function

}

if(USER\_MODE == 0) //USER\_MODE0=Display "PASS" on 7-Segment

{

while(count<300)

{

\*) USER\_MODE == 0 비밀 핀에 해당하는 경우, 7 세그먼트는 'PASS'를 보여줄 것이다.

PORTB =PASS[0]; PORTF = 0x80 ; //"P"

delay\_ms(5);

PORTB =PASS[1]; PORTF = 0x40 ; //"A"

delay\_ms(5);

PORTB =PASS[2]; PORTF = 0x20 ; //"S"

delay\_ms(5);

PORTB =PASS[3]; PORTF = 0x10 ; //"S"

delay\_ms(5);

count++;

}

PORTB =0x00; PORTF = 0x80 ;

for( i = 0; i<125 ; i++);

PORTB =0x00; PORTF = 0x40 ;

for( i = 0; i<125 ; i++);

PORTB =0x00; PORTF = 0x20 ;

for(i = 0; i<125 ; i++);

PORTB =0x00; PORTF = 0x10 ;

for(i = 0; i<125 ; i++);

}

if(USER\_MODE == 1 )

{

LCD\_init();

// 첫 번째 라인에 출력

LCD\_String("Door is Opened"); //LCD에 문구 출력

for(; k<5; k++) //7-Segment Blink 5 times

{

for(j=0; j<5000; j++)

{

\*) USER\_MODE == 1은 점멸 기능이 7- 세그먼트에서 'OPEN'표시하는 데 사용된다.

\*) 점멸 기능 미만 5로 for사용하여 설정되었다가 5 회까지 작동된다는 것을 의미한다.

PORTB = FONT[0]; PORTF = 0x80; //"O"

for(i = 0; i<10 ; i++);

PORTB = FONT[1]; PORTF = 0x40; //"P"

for(i = 0; i<10 ; i++);

PORTB = FONT[2]; PORTF = 0x20; //"E"

for(i = 0; i<10 ; i++);

PORTB = FONT[3]; PORTF = 0x10; //"N"

for(i = 0; i<10 ; i++);

}

for(j=0; j<500; j++)

{

PORTB = 0x00; PORTF = 0x80;

for(i = 0; i<10 ; i++);

PORTB = 0x00; PORTF = 0x40;

for(i = 0; i<10 ; i++);

PORTB = 0x00; PORTF = 0x20;

for(i = 0; i<10 ; i++);

PORTB = 0x00; PORTF = 0x10;

for(i = 0; i<10 ; i++);

}

}

}

if(USER\_MODE == 2)

{

LCD\_init();

// 첫 번째 라인에 출력

\*) 이전과 처럼, USER\_MODE의 == 2 가까이 'CLSE'을 () 표시하는 데 사용된다.

LCD\_String("Door is Closed"); //LCD에 문구 출력

for(; k<5; k++) //7-Segment Blink 5 times

{

for(j=0; j<5000; j++)

{

PORTB = FONT[4]; PORTF = 0x80; //"C"

for( i = 0; i<10 ; i++);

PORTB = FONT[5]; PORTF = 0x40 ; //"L"

for( i = 0; i<10 ; i++);

PORTB = FONT[6]; PORTF = 0x20 ; //"S"

for(i = 0; i<10 ; i++);

PORTB = FONT[2]; PORTF = 0x10 ; //"E"

for(i = 0; i<10 ; i++);

}

for(j=0; j<500; j++)

{

PORTB = 0x00; PORTF = 0x80;

for(i = 0; i<10 ; i++);

PORTB = 0x00; PORTF = 0x40;

for(i = 0; i<10 ; i++);

PORTB = 0x00; PORTF = 0x20;

for(i = 0; i<10 ; i++);

PORTB = 0x00; PORTF = 0x10;

for(i = 0; i<10 ; i++);

}

}

}

}

}

void main(void)

{

PORTF=0x00; DDRF=0xF0; //PF4-PF7 출력 설정

PORTB=0x00; DDRB=0xFF; //포트 B 출력 설정

PORTC=0xFF; DDRC=0xFF; //포트 C에 출력

TCCR0=0x04; //clkT0s/64 (prescaler)

TCNT0=0x00; TIMSK=0x01; //Timer/Counter0 overflow interrupt is enabled

#asm("sei") //enable all interrupt

LCD\_init();

// 첫 번째 라인에 출력

LCD\_String(" "); //LCD Blank

while (1)

{

}

}

void Delay(unsigned int cnt){

unsigned int i,j;

for(i=0;i<cnt;i++){

j=50000;

while(--j);

}

}

// LCD 초기화

void LCD\_init(void)

{

DDRD = 0xFF; // LCD Control, 포트 D 출력 설정

DDRA = 0xFF; // LCD Control

LCD\_ENABLE=0; // E = 0;

delay\_ms(15);

Command(0x30); //setting cursor

delay\_ms(5);

Command(0x30);

delay\_us(100);

Command(0x30);

Command(FUNCSET);

Command(DISPON);

Command(ALLCLR);

Command(ENTMODE);

}

// 문자열출력함수

void LCD\_String(char flash str[])

{

char flash \*pStr=0;

pStr = str;

while(\*pStr) Data(\*pStr++);

}

// 인스트럭션 쓰기함수

void Command(unsigned char byte) //RS=register select, RW=Register write, Enable, command enter register

{

Busy();

PORTA = byte; // 데이터

LCD\_RS = 0; // RS = 0;

LCD\_RW\_L = 0; // RW = 0;

delay\_us(1);

LCD\_ENABLE = 1; // E = 1;

delay\_us(1);

LCD\_ENABLE = 0; // E = 0;

}

//데이터쓰기함수

void Data(unsigned char byte)

{

Busy();

PORTA = byte; // 데이터

LCD\_RS = 1; // RS = 1;

LCD\_RW\_L = 0; // RW = 0;

delay\_us(1);

LCD\_ENABLE = 1; // E = 1;

delay\_us(1);

LCD\_ENABLE = 0; // E = 0;

}

// Busy Flag Check -> 일반적인 BF를 체크하는 것이 아니라

// 일정한 시간지연을 이용한다.

void Busy(void)

{

delay\_ms(2);

}

**마이크로프로세서**

**전자식 도어 로크 (Electronic Door Lock)**

**이반**

**마이크로프로세서**

**전자식 도어 로크 (Electronic Door Lock)**

**이반**

**마이크로프로세서**

**전자식 도어 로크 (Electronic Door Lock)**

**이반**