

작업과제 01 LED 깜빡이기

```
#include <iom128v.h>
#include <macros.h>
```

```
void delay(long dt)
{
    while(dt--);
}
```

```
void main(void)
{
    DDRC = 0xFF;
    while(1){
        PORTC = 0x01;
        delay(10000);
        PORTC = 0x02;
        delay(10000);
    }
}
```

작업과제 02 내장 스위치 입력 받기

```
#include <iom128v.h>
#include <macros.h>
void delay(int dt){
    int i, j;
    for(i=0;i<=100;i++){
        for(j=0;j<=dt;j++);
    }
}

void main(void){
    char keyin;
    DDRC = 0x03;
    while(1){
        keyin = (PINB&0x0C)>>2;
        if(keyin == 0x01){
            PORTC ^= 0x01;
            delay(1000);
        }
        if(keyin == 0x02){
            PORTC ^= 0x02;
            delay(1000);
        }
    }
}
```

```
}  
}
```

작업과제 03 스위치 입력모듈(AM-TS8)을 이용하여 입력 받기

```
#include <iom128v.h>  
#include <macros.h>  
void delay(int dt)  
{  
    int i, j;  
    for(i=0;i<=100;i++){  
        for(j=0;j<=dt;j++);  
    }  
}  
void main(void)  
{  
    char keyin;  
    DDRA = 0x00;  
    DDRG = 0x03;  
    while(1){  
        keyin = PINA;  
        if(keyin == 0x01){  
            PORTG ^= 0x01;  
            delay(1000);  
        }  
        if(keyin == 0x02){  
            PORTG ^= 0x02;  
            delay(1000);  
        }  
    }  
}
```

작업과제 04 외부인터럽트

```
// ICC-AVR application builder : 2007-03-22 PM 7:32:10  
// Target : M128  
// Crystal: 16.000MHz  
#include <iom128v.h>  
  
#pragma interrupt_handler ISR0:iv_INT0  
#pragma interrupt_handler ISR1:iv_INT1
```

```

int i = 0;

void ISR0(void)
{
    i++; //external interrupt
}

void ISR1(void)
{
    i++; //external interrupt
}

void main ( void )
{
    DDRG = 0x03;
    EICRA |= 0x03; //INT0 rising edge detect
    EICRA |= 0x0c; //INT1 rising edge detect
    EIMSK |= 0x01; //INT0 Mask Enable
    EIMSK |= 0x02; //INT1 Mask Enable
    SREG = SREG | 0x80;

    while( 1 ){
        if ( i%2 )
            PORTG = 0x01;
        else
            PORTG = 0x02;
    }
}

```

작업과제 05 세븐세그먼트 제어하기

```

#include <iom128v.h>
#include <macros.h>

void delay(int n);

const char segment_data[10] = {63, 6,91,79,102,109,125,39,127,103};
unsigned char display_num[4]={0,0,0,0};           // 4 세븐세그먼트를 위한 버퍼

//TIMER0 initialize - prescale:64
// WGM: Normal
// desired value: 1KHz

```

```

// actual value: 1.000KHz (0.0%)
unsigned char digit_num=0;
#pragma interrupt_handler timer0_ovf_isr:17
void timer0_ovf_isr(void)
{
    TCNT0 = 0x06; //reload counter value
    digit_num++;
    digit_num = digit_num%4;
    PORTC = 0x0f;
    PORTA = segment_data[display_num[digit_num]];
    PORTC = ~(0x01 << digit_num);
}

//call this routine to initialize all peripherals
void init_devices(void)
{
    //stop errant interrupts until set up
    SREG = 0x00;
    DDRA = 0xff;
    DDRC = 0x0f;
    TCNT0 = 0x06; //set count
    TCCR0 = 0x04; //start timer
    TIMSK = 0x01; //timer interrupt sources
    SREG = 0x80;
    //all peripherals are now initialized
}

void main(void)
{
    int i=0;
    init_devices();

    while(1){
        i++;
        display_num[0] = (i%10000)/1000;
        display_num[1] = (i%1000)/100;
        display_num[2] = (i%100)/10;
        display_num[3] = (i%10);
        delay(1000);
    }
}

```

```

void delay(int n)
{
    volatile int i, j;
    for(i=1;i<n;i++)
    {
        for(j=1;j<600;j++);
    }
}

```

작업과제 06 UART 통신으로 데이터 출력하기(시리얼 통신 RS-232C)

```

#include <iom128v.h>
#include <macros.h>

void main(void)
{
    SREG = 0x00;
    UCSR0C = 0x06; // 8 bit, Asynchronous, parity none, stop bit 0
    UBRR0L = 103; //16000000/(9600*16) -1; //set baud rate lo 9,600bps
    UBRR0H = 0; //set baud rate hi
    UCSR0B = 0x18; // RX enable, TX enable
    SREG = 0x80; // Global interrupt enable

    while(1){
        UDR0 = 'a';
    }
}

```

예제 2 UART0과 UART1로 동시에 출력하는 예제입니다.

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>

void tx0Char(char message);
void tx1Char(char message);

int putchar(char c)
{
    tx0Char(c);
    tx1Char(c);
}

```

```

    return c;
}

// Output using UART0
void tx0Char(char message)
{
    while (((UCSR0A>>UDRE0)&0x01) == 0) ;    // UDRE, data register empty
    UDR0 = message;
}

// Output using UART1
void tx1Char(char message)
{
    while (((UCSR1A>>UDRE1)&0x01) == 0) ;    // UDRE, data register empty
    UDR1 = message;
}

int getchar(void)
{
    while ((UCSR0A & 0x80) == 0);
    return UDR0;
}

//UART0 initialize
//desired baud rate:9600 //actual: baudrate:9615 (0.2%) //char size: 8 bit / parity:
//Disabled
void uart0_init(void)
{
    UCSR0B = 0x00; //disable while setting baud rate
    UCSR0A = 0x00;
    UCSR0C = 0x06;
    UBRR0L = 0x67; //set baud rate lo
    UBRR0H = 0x00; //set baud rate hi
    UCSR0B = 0x18;
}

//UART1 initialize
//desired baud rate:9600 // actual baudrate:9615 (0.2%) //charsize: 8 bit //parity:
//Disabled
void uart1_init(void)

```

```

{
    UCSR1B = 0x00;    //disable while setting baud rate
    UCSR1A = 0x00;
    UCSR1C = 0x06;
    // UBRR1L = 0x2F; //set baud rate low under Crystal 7.3728 MHz
    // UBRR1L = 0x47; //set baud rate low under Crystal 11.0592 MHz
    UBRR1L = 0x67;    //set baud rate low under Crystal 16.000 MHz
    UBRR1H = 0x00;    //set baud rate hi
    UCSR1B = 0x18;
}

```

```

void init_devices(void)

```

```

{
    //stop errant interrupts until set up
    CLI(); //disable all interrupts
    DDRC = 0x03;
    uart0_init();          // UART 0 initialization
    uart1_init();          // UART 1 initialization
    SEI(); //re-enable interrupts //all peripherals are now initialized
}

```

```

// Time delay function

```

```

void delay(int n)
{
    volatile int i,j;
    for(i=1;i<n;i++){
        for(j=1;j<600;j++);
    }
}

```

```

void main (void)

```

```

{
    int i=0;
    init_devices();

    printf("WnWrWntesting...WnWr");
    printf("printf Testing...WnWr");
    while(1){
        printf("Test i : %dWnWr", i++);
        PORTG = i;          // LED output
        delay(100);
    }
}

```

```
}  
}
```

작업과제 07 UART 통신으로 데이터 입력받기(시리얼 통신 RS-232C)

```
#include <iom128v.h>  
#include <macros.h>  
#include <stdio.h>  
  
//UART0 initialize  
// desired baud rate: 9600 // actual: baud rate:9615 (0.2%) // char size: 8 bit //  
parity: Disabled  
void uart0_init(void)  
{  
    UCSR0B = 0x00; //disable while setting baud rate  
    UCSR0A = 0x00;  
    UCSR0C = 0x06;  
    UBRR0L = 0x67; //set baud rate lo  
    UBRR0H = 0x00; //set baud rate hi  
    UCSR0B = 0x18;  
}  
int putchar(char c)  
{  
    while (((UCSR0A>>UDRE0)&0x01) == 0) ; // UDRE, data register empty  
    UDR0 = c;  
    return c;  
}  
  
int getchar(void)  
{  
    while ((UCSR0A & 0x80) == 0);  
    return UDR0;  
}  
  
void init_devices(void)  
{  
    CLI(); //disable all interrupts  
    DDRG = 0x03;  
    uart0_init();  
    SEI(); //re-enable interrupts  
}
```

```

void main(void){
    char in;
    init_devices();
    while(1){
        //Press Character 'A'~'D'
        //scanf("%c", &in);      // 컴파일러에 HEX 사이즈 제한이 없을 때 사용한다.
        in = getchar();          // 데모 컴파일러가 4k 사이즈 제한에 걸렸을 때 사용한
다.
        switch(in){
            case 'A' : printf("WrWnPressed A");
                        break;
            case 'B' : printf("WrWnPressed B");
                        break;
            case 'C' : printf("WrWnPressed C");
                        break;
            case 'D' : printf("WrWnPressed D");
                        break;
            default  : printf("WrWnPressed E~Z or other");
                        break;
        }
    }
}

```

작업과제 08 시리얼 LCD 사용하기 (터미널 모드)

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>

//UART0 initialize
//desired baudrate:9600 //actual:baudrate:9615 (0.2%) //char size: 8 bit //parity:
Disabled
void uart0_init(void) {
    UCSR0B = 0x00; //disable while setting baud rate
    UCSR0A = 0x00;
    UCSR0C = 0x06;
    UBRR0L = 0x67; //set baud rate lo
    UBRR0H = 0x00; //set baud rate hi
    UCSR0B = 0x18;
}
void delay(int delaytime) {

```

```

    int i,j;
    for(i=0;i<1000;i++){
        for(j=0;j<delaytime;j++);
    }
}

int putchar(char c) {
    while (((UCSR0A>>UDRE0)&0x01) == 0) ; // UDRE, data register empty
    UDR0 = c;
    return c;
}

void main(void) {
    uart0_init();
    while(1){
        printf("testing...");
        delay(1000);
        printf("$$CSWr");
        delay(1000);
        printf("$$LOWr");
        delay(1000);
        printf("$$L1Wr");
        delay(1000);
    }
}

```

작업과제 09 시리얼 LCD 사용하기 (커맨드 모드)

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>

//UART0 initialize
//desired baudrate:9600 //actual:baudrate:9615 (0.2%) //char size: 8 bit //parity:
Disabled
void uart0_init(void) {
    UCSR0B = 0x00; //disable while setting baud rate
    UCSR0A = 0x00;
    UCSR0C = 0x06;
    UBRR0L = 0x67; //set baud rate lo
    UBRR0H = 0x00; //set baud rate hi
    UCSR0B = 0x18;
}

```

```

void delay(int delaytime) {
    int i,j;
    for(i=0;i<1000;i++){
        for(j=0;j<delaytime;j++);
    }
}

int putchar(char c) {
    while (((UCSR0A>>UDRE0)&0x01) == 0) ; // UDRE, data register empty
    UDR0 = c;
    return c;
}

void main(void) {
    uart0_init();
    while(1){
        printf("$L,1Wr");
        delay(1000);
        printf("$L,0Wr");
        delay(1000);
        printf("$T,Testing...Wr");
        delay(1000);
        printf("$D,0Wr");
        delay(1000);
        printf("$D,1Wr");
        delay(1000);
    }
}

```

작업과제 10 릴레이 전원공급모듈 구동하기

```

#include <iom128v.h>
#include <macros.h>

// Relay powersupply module Control
#define RELAY_PORT      PORTC
#define RELAY_PORT_DDR  DDRC
#define RELAY_0_ON      (RELAY_PORT=1<<0)
#define RELAY_1_ON      (RELAY_PORT=1<<1)
#define RELAY_2_ON      (RELAY_PORT=1<<2)
#define RELAY_3_ON      (RELAY_PORT=1<<3)
#define RELAY_0_OFF      (RELAY_PORT=0<<0)

```

```

#define RELAY_1_OFF      (RELAY_PORT=0<<1)
#define RELAY_2_OFF      (RELAY_PORT=0<<2)
#define RELAY_3_OFF      (RELAY_PORT=0<<3)
#define DELAYTIME        20000

void delay(int delaytime){
    int i,j;
    for(i=0;i<1000;i++){
        for(j=0;j<delaytime;j++);
    }
}

void main(void)
{
    RELAY_DDR  = 0xFF;
    while(1){
        RELAY_PORT = 0xFF;
        // RELAY_0_ON;      // RELAY_1_ON;
        delay(DELAYTIME);
        RELAY_PORT = 0x00;
        // RELAY_0_OFF;      // RELAY_1_OFF;
        delay(DELAYTIME/2);
    }
}

```

작업과제 11 타이머오버플로우 인터럽트를 이용한 스피커 출력 실습 1

```

// ICC-AVR application builder, Target : M128, Crystal: 16.000MHz
#include <iom128v.h>
#include <macros.h>

#define C1      523      // 도
#define C1_      554
#define D1      587      // 레
#define D1_      622
#define E1      659      // 미
#define F1      699      // 파
#define F1_      740
#define G1      784      // 솔
#define G1_      831
#define A1      880      // 라
#define A1_      932

```

```
#define B1      988      // 시
```

```
#define C2      C1*2      // 도
```

```
#define C2_      C1_*2
```

```
#define D2      D1*2      // 레
```

```
#define D2_      D1_*2
```

```
#define E2      E1*2      // 미
```

```
#define F2      F1*2      // 파
```

```
#define F2_      F1_*2
```

```
#define G2      G1*2      // 솔
```

```
#define G2_      G1_*2
```

```
#define A2      A1*2      // 라
```

```
#define A2_      A1_*2
```

```
#define B2      B1*2      // 시
```

```
#define DLY_1    DLY_4*4    // 온 음표
```

```
#define DLY_2    DLY_4*2    // 2분 음표
```

```
#define DLY_4    400        // 4분 음표
```

```
#define DLY_8    DLY_4/2    // 8분 음표
```

```
#define DLY_16   DLY_8/2    // 16분 음표
```

```
volatile int schoolsong[100]=
```

```
{G1,G1,A1,A1,G1,G1,E1,G1,G1,E1,E1,D1,G1,G1,A1,A1,G1,G1,E1,G1,E1,D1,E1,C1};
```

```
volatile int length[100]=
```

```
{DLY_4,DLY_4,DLY_4,DLY_4,DLY_4,DLY_4,DLY_2,DLY_4,DLY_4,DLY_4,DLY_4,  
DLY_1,DLY_4,DLY_4,DLY_4,DLY_4,DLY_4,DLY_4,DLY_2,DLY_4,DLY_4,DLY_4,  
DLY_4,DLY_1,};
```

```
volatile unsigned char TIMER=0xFF;
```

```
volatile long value;
```

```
#pragma interrupt_handler ISR0:iv_TIM0_OVF // 17번 인터럽트벡터
```

```
void ISR0(void)
```

```
{  
    TCNT0 = TIMER; //reload counter value  
    PORTC ^= 0x80;  
}
```

```
void delay(int delaytime){
```

```
    int i,j;
```

```

    for(i=0;i<=1000;i++){
        for(j=0;j<=delaytime;j++){
            }
        }
    }

void main(void)
{
    int freq = 1000, i;

    DDRC = 0x80; //Setting speaker pin as output
    TIMSK = 0x01; //timer interrupt sources
    TCCR0 = 0x04; //Timer initialize (Prescaler & Normal)
    SREG = 0x80; //글로벌 인터럽트 Enable

    while(1){
        for(i=0;i<25;i++){
            freq = schoolsong[i];
            value = 255-(1000000/(8*freq)); // 몇 번 ON/OFF변경할 것인지 결정.
            TIMER = value;
            delay(length[i]);
            TIMER=255;
            delay(5);
        }
        TIMSK = 0x00; //Timer interrupt OFF
        delay(10000); //Delay with soundoff for sometime
        TIMSK = 0x01; //Timer interrupt ON
    }
}

```

작업과제 12 타이머오버플로우 인터럽트를 이용한 스피커 출력 실습2

```

#include <iom128v.h>
#include <macros.h>

#define C1      523      // 도
#define C1_     554
#define D1      587      // 레
#define D1_     622
#define E1      659      // 미
#define F1      699      // 파
#define F1_     740

```

```

#define G1      784      // 솔
#define G1_     831
#define A1      880      // 라
#define A1_     932
#define B1      988      // 시

#define C2      C1*2     // 도
#define C2_     C1_*2
#define D2      D1*2     // 레
#define D2_     D1_*2
#define E2      E1*2     // 미
#define F2      F1*2     // 파
#define F2_     F1_*2
#define G2      G1*2     // 솔
#define G2_     G1_*2
#define A2      A2*2     // 라
#define A2_     A2_*2
#define B2      B2*2     // 시

#define DLY_1   DLY_4*4   // 온 음표
#define DLY_2   DLY_4*2   // 2분 음표
#define DLY_4   400       // 4분 음표
#define DLY_8   DLY_4/2   // 8분 음표
#define DLY_16  DLY_8/2   // 16분 음표

```

```

volatile int schoolsong[100]=
    {G1,G1,A1,A1,G1,G1,E1,G1,G1,E1,E1,D1,G1,G1,A1,A1,G1,G1,E1,
     G1,E1,D1,E1,C1};
volatile int length[100]=
    {DLY_4,DLY_4,DLY_4,DLY_4,DLY_4,DLY_4,DLY_2,DLY_4,DLY_4,DLY_4,DLY_4,
     DLY_1,DLY_4,DLY_4,DLY_4,DLY_4,DLY_4,DLY_4,DLY_2,DLY_4,DLY_4,DLY_4,
     DLY_4,DLY_1,};

```

```

volatile unsigned char TIMER=0x06;
volatile long value;

```

```

//TIMER0 initialize - prescale:64
// WGM: Normal
// desired value: 1KHz
// actual value: 1.000KHz (0.0%)
void timer0_init(void)

```

```

{
    TCCR0 = 0x00; //stop
    ASSR = 0x00; //set async mode
    TCNT0 = 0x06; //set count
    OCR0 = 0xFA;
    TCCR0 = 0x04; //start timer
}
#pragma interrupt_handler T0_OVF:iv_TIM0_OVF
void T0_OVF(void)
{
    TCNT0 = TIMER; //reload counter value
    PORTC ^= 0x10;
}

//call this routine to initialize all peripherals
void init_devices(void)
{
    CLI(); //disable all interrupts
    DDRC = 0x10;
    timer0_init();
    TIMSK = 0x01; //timer interrupt sources
    SEI(); //re-enable interrupts
}

void delay(int delaytime){
    int i,j;
    for(i=0;i<=1000;i++){
        for(j=0;j<=delaytime;j++){
        }
    }
}

void main(void)
{
    int freq = 1000, i;
    init_devices();

    while(1){
        for(i=0;i<25;i++){
            freq = schoolsong[i];
            value = 255-(1000000/(8*freq)); // value
            =255-((1/(2*freq))/(4/1000000));

```

```

        TIMER = value;
        delay(length[i]);
        TIMER=255;
        delay(5);
    }
    TIMSK = 0x00;
    delay(10000);
    TIMSK = 0x01;
}
}

```

작업과제 13 DC 모터 구동모듈(AM-CD2L(소형))을 이용한 DC 모터 제어

```

#include <iom128v.h>
#include <macros.h>

#define MOTORPORT1 PORTC
#define ON          1
#define OFF         0
#define CW          1
#define CCW         0

// Motor Control
void M1A(int onoff){
    if(onoff==ON)
        MOTORPORT1 = MOTORPORT1|0x01;
    else
        MOTORPORT1 = MOTORPORT1&0xFE;
}

void M1A_(int onoff){
    if(onoff==ON)
        MOTORPORT1 = MOTORPORT1|0x02;
    else
        MOTORPORT1 = MOTORPORT1&0xFD;
}

void M1B(int onoff){
    if(onoff==ON)
        MOTORPORT1 = MOTORPORT1|0x04;
    else
        MOTORPORT1 = MOTORPORT1&0xFB;
}

```

```
}
```

```
void M1B_(int onoff){  
    if(onoff==ON)  
        MOTORPORT1 = MOTORPORT1|0x08;  
    else  
        MOTORPORT1 = MOTORPORT1&0xF7;  
}
```

```
void Motor1(int CWCCW){  
    if(CWCCW==CW){  
        M1A(ON);  
        M1A_(OFF);  
    }else{  
        M1A(OFF);  
        M1A_(ON);  
    }  
}
```

```
void Motor2(int CWCCW){  
    if(CWCCW==CW){  
        M1B(ON);  
        M1B_(OFF);  
    }else{  
        M1B(OFF);  
        M1B_(ON);  
    }  
}
```

```
void delay(int delaytime){  
    int i,j;  
    for(i=0;i<1000;i++){  
        for(j=0;j<delaytime;j++){  
        }  
    }  
}
```

```
void main(void){  
    SREG = 0x00; //disable all interrupts  
    DDRC = 0xFF;  
    SREG = 0x80; //Enable all interrupts  
    while(1){
```

```

        Motor1(0); // Clockwise
        Motor2(0);
        delay(1000);
        Motor1(1); // Counter Clockwise
        Motor2(1);
        delay(1000);
    }
}

```

작업과제 14 DC 모터제어 (GPIO로 모터방향제어)

```

#include <iom128v.h>
#include <macros.h>

#define MOTOR_PORT          PORTE
#define MOTOR_PORT_DDR      DDRE

#define PWM1_ON              (MOTOR_PORT|=0x01)
#define PWM1_OFF             (MOTOR_PORT&=0xFE)
#define DIR1_CW              (MOTOR_PORT|=0x02)
#define DIR1_CCW             (MOTOR_PORT&=0xFD)
#define ENABLE1_OFF          (MOTOR_PORT|=0x04)
#define ENABLE1_ON           (MOTOR_PORT&=0xFB)
#define BREAK1_ON            (MOTOR_PORT|=0x08) //For AM-DC1-3C
#define BREAK1_OFF           (MOTOR_PORT&=0xF7)

#define PWM2_ON              (MOTOR_PORT|=0x10)
#define PWM2_OFF             (MOTOR_PORT&=0xEF)
#define DIR2_CW              (MOTOR_PORT|=0x20)
#define DIR2_CCW             (MOTOR_PORT&=0xDF)
#define ENABLE2_OFF          (MOTOR_PORT|=0x40)
#define ENABLE2_ON           (MOTOR_PORT&=0xBF)
#define BREAK2_ON            (MOTOR_PORT|=0x80)
#define BREAK2_OFF           (MOTOR_PORT&=0x7F)

void delay(int delaytime){
    int i,j;
    for(i=0;i<1000;i++){
        for(j=0;j<delaytime;j++){
        }
    }
}

void motor_init(void){

```

```

    MOTOR_PORT_DDR = 0xff;
    ENABLE1_ON;
    PWM1_ON;
    BREAK1_OFF;
    DIR1_CW;

    ENABLE2_ON;
    PWM2_ON;
    BREAK2_OFF;
    DIR2_CW;
}

//
void main(void)
{
    motor_init();
    while(1){
        DIR1_CW;
        DIR2_CW;
        delay(3000);

        DIR1_CCW;
        DIR2_CCW;
        delay(3000);
    }
}

```

작업과제 15 DC 모터제어 (타이머0 오버플로우 인터럽트로 PWM 속도제어)

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>

// DC MOTOR Control
#define MOTOR_PORT      PORTE
#define MOTOR_PORT_DDR  DDRE

#define PWM_PORT        0
#define PWM1_ON          (MOTOR_PORT|=0x01)
#define PWM1_OFF         (MOTOR_PORT&=0xFE)
#define DIR1_CW          (MOTOR_PORT|=0x02)
#define DIR1_CCW         (MOTOR_PORT&=0xFD)
#define ENABLE1_OFF      (MOTOR_PORT|=0x04)
#define ENABLE1_ON       (MOTOR_PORT&=0xFB)

```

```

#define BREAK1_ON          (MOTOR_PORT|=0x08)
#define BREAK1_OFF        (MOTOR_PORT&=0xF7)
#define CW                 1
#define CCW                2
#define STOP               3

// SPEED Control
#define MIN_SPEED          1
#define MAX_SPEED          1700
#define PLUS               10

// SWITCH Control
#define SW_PORT             PINA
#define SW_PORT_DDR        DDRA
#define PRESSED            1
#define OPEN               0

volatile unsigned long T20ms=0, motor_state=CW;
volatile unsigned int Tcount=0, motor_speed, motor_dir=CW;

//TIMER0 initialize – prescale:1
// WGM: Normal
// desired value: 1uSec
// actual value: 1.000uSec (0.0%)
void timer0_init(void)
{
    TCCR0 = 0x00; //stop
    ASSR = 0x00; //set async mode
    TCNT0 = 0xF0; //set count
    OCR0 = 0x10;
    TCCR0 = 0x01; //start timer
}

#pragma interrupt_handler timer0_ovf_isr:iv_TIM0_OVF
void timer0_ovf_isr(void)
{
    TCNT0 = 0xF0; //reload counter value
    T20ms++;
    if(T20ms >= 1700){
        T20ms = 0;
    }
}

```

```

        if(motor_speed > T20ms)
            MOTOR_PORT |= 0x01;
        else
            MOTOR_PORT &= 0xfe;
    }
    //call this routine to initialize all peripherals
    void init_devices(void)
    {
        //stop errant interrupts until set up
        CLI(); //disable all interrupts
        XDIV = 0x00; //xtal divider
        XMCRA = 0x00; //external memory
        timer0_init();
        TIMSK = 0x01; //timer interrupt sources
        SEI(); //re-enable interrupts
    }

    void delay(int cnt)
    {
        unsigned int i, j;
        for(i=0;i<cnt;i++){
            for(j=0;j<1000;j++){
            }
        }
    }

    void motor_init(void){
        ENABLE1_ON;
        PWM1_ON;
        BREAK1_OFF;
        DIR1_CW;
    }

    void main(void)
    {
        init_devices();
        motor_init();
        SW_PORT_DDR = 0x00;
        MOTOR_PORT_DDR = 0xff;
        motor_speed = 800;
        DIR1_CW;
    }

```

```

while(1){
    switch(SW_PORT&0x3f){
        case 0x01 : while((SW_PORT&0x0f)!=0x00);
                    motor_speed = motor_speed + PLUS;
                    break;

        case 0x02 : while((SW_PORT&0x3f)!=0x00);
                    BREAK1_OFF;
                    break;

        case 0x04 : while((SW_PORT&0x3f)!=0x00);
                    BREAK1_ON;

                    break;

        case 0x08 : while((SW_PORT&0x0f)!=0x00);
                    motor_speed = motor_speed - PLUS;

                    break;

        case 0x20 : while((SW_PORT&0x3f)!=0x00);
                    if(motor_dir == CW){
                        DIR1_CCW;
                        motor_dir = CCW;
                    }else{
                        DIR1_CW;
                        motor_dir = CW;
                    }
                    break;

        default : break;
    }
    if(motor_speed >= MAX_SPEED){
        motor_speed = MAX_SPEED;
    }
    if(motor_speed <= MIN_SPEED){
        motor_speed = MIN_SPEED;
    }
}
}

```

작업과제 16 DC 모터제어 (Native PWM)

//ICC-AVR application builder : 2007-11-19 PM 5:41:05

// Target : M128

// Crystal: 16.000MHz

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>

int putchar(char c);
int getchar(void);
void uart0_init(void);
void Initialize_Timer0(void);
void Initialize_port(void);

#define MOTOR_DDR      DDRB
#define MOTOR_PORT      PORTB

#define PWM1_ON()      (MOTOR_PORT|=0x01)
#define PWM1_OFF()     (MOTOR_PORT&=0xFE)
#define DIR1_ON()      (MOTOR_PORT|=0x02)
#define DIR1_OFF()     (MOTOR_PORT&=0xFD)
#define ENABLE1_OFF() (MOTOR_PORT|=0x04)
#define ENABLE1_ON()   (MOTOR_PORT&=0xFB)
#define BREAK1_ON()    (MOTOR_PORT|=0x08)
#define BREAK1_OFF()   (MOTOR_PORT&=0xF7)

#define PWM2_ON()      (MOTOR_PORT|=0x10)
#define PWM2_OFF()     (MOTOR_PORT&=~0x10)
#define DIR2_ON()      (MOTOR_PORT|=0x20)
#define DIR2_OFF()     (MOTOR_PORT&=~0x20)
#define ENABLE2_OFF() (MOTOR_PORT|=0x40)
#define ENABLE2_ON()   (MOTOR_PORT&=~0x40)
#define BREAK2_ON()    (MOTOR_PORT|=0x80)
#define BREAK2_OFF()   (MOTOR_PORT&=~0x80)

void Initialize_Timer0(void)
{
    TCCR0=0x00;
    TCCR0=(0<<CS02)|(0<<CS01)|(1<<CS00)|(1<<WGM00)|(1<<WGM01)|(1<<COM01);
}

void Initialize_port(void)
{
    MOTOR_DDR=0xFF;
    MOTOR_PORT=0x00;
}

```

```

}

int main(void)
{
    int i;
    Initialize_port();
    Initialize_Timer0();

    ENABLE1_ON();
    BREAK1_OFF();
    ENABLE2_ON();
    BREAK2_OFF();
    while(1)
    {
        for(i=0;i<100;i++)
        {
            OCR0=150;
            _delay_ms(1000);
        }
    }
}

```

프로그램 (PORTB D5, OC1A PWM test)

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>

```

```

int putchar(char c);
int getchar(void);
void uart0_init(void);

```

```

void Initialize_Timer1(void);
void delay(long dt);

```

```

volatile int value = 0;

```

```

int putchar(char c)
{
    while (((UCSR0A>>UDRE0)&0x01) == 0) ; // UDRE, data register empty
    UDR0 = c;
    return c;
}

```

```

}

int getchar(void)
{
    while ((UCSR0A & 0x80) == 0);
    return UDR0;
}

//UART0 initialize
// desired baud rate: 9600
// actual: baud rate:9615 (0.2%)
// char size: 8 bit
// parity: Disabled
void uart0_init(void)
{
    UCSR0B = 0x00; //disable while setting baud rate
    UCSR0A = 0x00;
    UCSR0C = 0x06;
    UBRR0L = 0x67; //set baud rate lo
    UBRR0H = 0x00; //set baud rate hi
    UCSR0B = 0x18;
}

void delay(long dt){
    while(dt--);
}

void Initialize_Timer1(void)
{
    CLI(); //disable all interrupts
    TCCR1A = (1<<COM1A1)|(1<<WGM11)|(1<<WGM10);
    TCCR1B = (1<<WGM12)|(1<<CS12)|(0<<CS11)|(1<<CS10);
    TIMSK = (1<<OCIE1A); //timer interrupt sources
    SEI(); //re-enable interrupts
}

#pragma interrupt_handler timer1_compa_isr:iv_TIM1_COMPA
void timer1_compa_isr(void)
{
    //compare occurred TCNT1=OCR1A
}

```

```

#pragma interrupt_handler adc_isr:iv_ADC
void adc_isr(void)
{
    //conversion complete, read value (int) using...
    value=ADCL;           //Read 8 low bits first (important)
    value|=(int)ADCH << 8; //read 2 high bits and shift into top byte
    value &=0x03ff;
}

int main(void)
{
    DDRB  = 0xFF;
    DDRC  = 0xFF;
    PORTB = 0xFF;

    uart0_init();
    Initialize_Timer1();
    printf("WrWnTest native PWM");

    ADCSRA |= 0x28;
    ADCSRA &= 0x7f;
    ADMUX = 0x01;
    ADCSRA |= 0xC0;
    OCR1A = 0;

    while(1){
        printf("WrWn ADC : %d", value);
        OCR1A = value;      // motor PWM control
        PORTC = value>>2;   // LED output
    }
    return 0;
}

```

작업과제 17 스텝핑 모터 구동모듈(AM-CS2P)을 이용한 스텝핑 모터 제어

[프로그램 \(1상여자방식\)](#)

```

#include <iom128v.h>
#include <macros.h>

```

```

#define MOTOR_PORT      PORTC
#define MOTOR_PORT_DDR  DDRC

```

```

void delay(int n)
{
    volatile int i,j;
    for(i=1;i<n;i++)
    {
        for(j=1;j<600;j++);
    }
}

void main(void)
{
    SREG = 0x80;
    MOTOR_PORT_DDR = 0xff;
    while(1){
        delay(10);
        MOTOR_PORT = 0x11;
        delay(10);
        MOTOR_PORT = 0x22;
        delay(10);
        MOTOR_PORT = 0x44;
        delay(10);
        MOTOR_PORT = 0x88;
    }
}

```

프로그램 (2상여자방식)

```

#include <iom128v.h>
#include <macros.h>

#define MOTOR_PORT      PORTC
#define MOTOR_PORT_DDR  DDRC

void delay(int n)
{
    volatile int i,j;
    for(i=1;i<n;i++){
        for(j=1;j<600;j++);
    }
}

```

```

void main(void)
{
    SREG = 0x80;
    MOTOR_PORT_DDR = 0xff;
    while(1){
        delay(10);
        MOTOR_PORT = 0x33;
        delay(10);
        MOTOR_PORT = 0x66;
        delay(10);
        MOTOR_PORT = 0xcc;
        delay(10);
        MOTOR_PORT = 0x99;
    }
}

```

프로그램 (1-2상여자방식)

```

#include <iom128v.h>
#include <macros.h>

#define MOTOR_PORT      PORTC
#define MOTOR_PORT_DDR  DDRC

void delay(int n)
{
    volatile int i,j;
    for(i=1;i<n;i++){
        for(j=1;j<600;j++);
    }
}

void main(void)
{
    SREG = 0x80;
    MOTOR_PORT_DDR = 0xff;
    while(1){
        delay(10);
        MOTOR_PORT = 0x11;
        delay(10);
        MOTOR_PORT = 0x33;
    }
}

```

```

        delay(10);
        MOTOR_PORT = 0x22;
        delay(10);
        MOTOR_PORT = 0x66;
        delay(10);
        MOTOR_PORT = 0x44;
        delay(10);
        MOTOR_PORT = 0xcc;
        delay(10);
        MOTOR_PORT = 0x88;
        delay(10);
        MOTOR_PORT = 0x99;
    }
}

```

작업과제 18 스텝핑 모터 구동모듈(AM-MS2)를 이용한 마이크로스텝 구동하기

```

/*
Microstepping motor control (AM-MS2)
Enable      : PORTC #D7
Mode1       : PORTC #D5
Mode0       : PORTC #D4
R_DIR       : PORTC #D3
R_CLK       : PORTC #D2
L_DIR       : PORTC #D1
L_CLK       : PORTC #D0
*/
#include <iom128v.h>
#include <macros.h>

#define MOTOR_PORT      PORTC      // Stepping motor controller
#define MOTOR_PORT_DDR  DDRC       // DDRx of Stepping motor controller

#define MOTOR_ENABLE    (MOTOR_PORT = MOTOR_PORT|0x80)
#define MOTOR_DISABLE   (MOTOR_PORT = MOTOR_PORT&0x7f)
#define MOTOR_STEP_M0   (MOTOR_PORT = (MOTOR_PORT&0xcf)|0x00)
#define MOTOR_STEP_M1   (MOTOR_PORT = (MOTOR_PORT&0xcf)|0x10)
#define MOTOR_STEP_M2   (MOTOR_PORT = (MOTOR_PORT&0xcf)|0x20)
#define MOTOR_STEP_M3   (MOTOR_PORT = (MOTOR_PORT&0xcf)|0x30)
#define MOTOR_LEFT_CLK  (MOTOR_PORT = MOTOR_PORT^0x01)

```

```

#define MOTOR_RIGHT_CLK      (MOTOR_PORT = MOTOR_PORT^0x04)
#define MOTOR_LEFT_CW        (MOTOR_PORT = MOTOR_PORT&0xfd)
#define MOTOR_LEFT_CCW       (MOTOR_PORT = MOTOR_PORT|0x02)
#define MOTOR_RIGHT_CW       (MOTOR_PORT = MOTOR_PORT&0xf7)
#define MOTOR_RIGHT_CCW      (MOTOR_PORT = MOTOR_PORT|0x08)

void delay(int n)
{
    volatile int i,j;
    for(i=1;i<n;i++)
    {
        for(j=1;j<600;j++)
        {
        }
    }
}

void main(void)
{
    SREG = 0x80;
    MOTOR_PORT_DDR = 0xff;
    MOTOR_ENABLE;
    MOTOR_STEP_M3;
    MOTOR_LEFT_CCW;
    MOTOR_RIGHT_CCW;

    /* Microstep driver control by delay funtion */
    while(1){
        PORTG++;
        MOTOR_LEFT_CLK;
        MOTOR_RIGHT_CLK;
        delay(2);
        MOTOR_LEFT_CLK;
        MOTOR_RIGHT_CLK;
        delay(2);
    }
}

```

작업과제 19 RC용 서보모터 구동하기

```

#include <iom128v.h>
#include <macros.h>

```

```
#define SERVO_PORT1          PORTC
```

```
#define t20ms                360
```

```
#define t20mstimer_expired   0
```

```
#define ON                    1
```

```
#define OFF                    0
```

```
#define MAX_SERVO             8
```

```
#define MIN                    10
```

```
#define MAX                    35
```

```
volatile int t100ustimer=0, servo[MAX_SERVO], servo_num;
```

```
//TIMER0 initialize – prescale:8
```

```
// WGM: Normal
```

```
// desired value: 1uSec
```

```
// actual value: 1.000uSec (0.0%)
```

```
void timer0_init(void)
```

```
{
```

```
    TCCR0 = 0x00; //stop
```

```
    ASSR = 0x00; //set async mode
```

```
    TCNT0 = 0xFE; //set count
```

```
    OCR0 = 0x02;
```

```
    TCCR0 = 0x02; //start timer
```

```
}
```

```
#pragma interrupt_handler T0_OVF_isr:17
```

```
void T0_OVF_isr(void)
```

```
{
```

```
    TCNT0 = 0xE7; //reload counter value
```

```
    t100ustimer++;
```

```
    if(t100ustimer > t20ms){
```

```
        t100ustimer = t20mstimer_expired;
```

```
    }
```

```
    for(servo_num=0;servo_num<8;servo_num++){
```

```
        if(servo[servo_num] > t100ustimer){
```

```
            SERVO_PORT1 = SERVO_PORT1|(0x01<<servo_num);
```

```
        }else{
```

```
            SERVO_PORT1 = SERVO_PORT1&(~(0x01<<servo_num));
```

```
        }
```

```
    }
```

```
}
```

```
//call this routine to initialize all peripherals
```

```
void init_devices(void)
```

```
{
```

```
    //stop errant interrupts until set up
```

```
    SREG = 0x00;
```

```
    DDRC = 0xff;
```

```
    timer0_init();
```

```
    uart0_init();
```

```
    TIMSK = 0x01; //timer interrupt sources
```

```
    SREG = 0x80;
```

```
}
```

```
void delay(int delaytime)
```

```
{
```

```
    int i, j;
```

```
    for(i=0;i<100;i++){
```

```
        for(j=0;j<delaytime;j++){
```

```
            }
```

```
    }
```

```
}
```

```
void main(void)
```

```
{
```

```
    int i;
```

```
    init_devices();
```

```
    for(i=0;i<MAX_SERVO;i++){
```

```
        servo[i]=0;
```

```
    }
```

```
    // Min : 10, Max : 35
```

```
    while(1){
```

```
        servo[1] = MIN;
```

```
        delay(100);
```

```
        servo[1] = MAX;
```

```
        delay(100);
```

```
    }
```

```
}
```

작업과제 20 텍스트 LCD 구동하기

```
//PORTA : RS:D0 RW:D1 E:D2
//PORTF : DATA

#include <iom128v.h>
#include <macros.h>

#define ON 1
#define OFF 0
#define DELAYTIME 10

void delay(int dd){
    int i,j;
    for(i=0 ; i<1000 ; i++){
        for(j=0 ; j<dd ; j++){
        }
    }
}

// RS : PORTA D0
void LCD_RS(int onoff){
    if(onoff == ON)
        PORTA = PORTA | 0x01;
    else
        PORTA = PORTA & 0xFE;
}

// RW : PORTA D1
void LCD_RW(int onoff){
    if(onoff == ON)
        PORTA = PORTA | 0x02;
    else
        PORTA = PORTA & 0xFD;
}

// RS : PORTA D2
void LCD_E(int onoff){
    if(onoff == ON)
        PORTA = PORTA | 0x04;
    else
        PORTA = PORTA & 0xFB;
}

void E_Pulse(void){
```

```
        char i;  
        LCD_E(ON);  
        delay(DELAYTIME*10);  
        LCD_E(OFF);  
        delay(DELAYTIME*10);  
    }
```

```
void write_char(char s){  
    LCD_RS(ON);  
    delay(DELAYTIME);  
    LCD_RW(OFF);  
    delay(DELAYTIME);  
    LCD_E(ON);  
    delay(DELAYTIME);  
    PORTF = s;  
    delay(DELAYTIME);  
    LCD_E(OFF);  
    delay(DELAYTIME);  
    LCD_E(ON);  
    delay(DELAYTIME);  
}
```

```
void clrscr(){  
    LCD_RS(OFF);  
    delay(DELAYTIME);  
    LCD_RW(OFF);  
    delay(DELAYTIME);  
    LCD_E(ON);  
    delay(DELAYTIME);  
    PORTF = 0x01;  
    delay(DELAYTIME);  
    LCD_E(OFF);  
    delay(DELAYTIME);  
    LCD_E(ON);  
    delay(DELAYTIME);  
}
```

```
void returnhome(){  
    LCD_RS(OFF);  
    delay(DELAYTIME);  
    LCD_RW(OFF);
```

```

        delay(DELAYTIME);
        LCD_E(ON);
        delay(DELAYTIME);
        PORTF = 0x02;
        delay(DELAYTIME);
        LCD_E(OFF);
        delay(DELAYTIME);
        LCD_E(ON);
        delay(DELAYTIME);
    }

void LCDONOFF(int onoff){
    LCD_RS(OFF);
    delay(DELAYTIME);
    LCD_RW(OFF);
    delay(DELAYTIME);
    LCD_E(ON);
    delay(DELAYTIME);
    if(onoff == ON){
        PORTF = 0x0F;
    }else{
        PORTF = 0x08;
    }
    delay(DELAYTIME);
    LCD_E(OFF);
    delay(DELAYTIME);
    LCD_E(ON);
    delay(DELAYTIME);
}

void lcd_disp(char x, char y){
    LCD_RS(OFF);
    LCD_RW(OFF);
    if(y==1)y++;
    else if(y==2) y--;
    if(y<=1){
        PORTF=y*20+x+0x80;
    }else{
        y=2;
        PORTF=y*20+x+0xc0;
    }
}

```

```

    E_Pulse();
}

void write_lcd(char x, char y, char *str){
    lcd_disp(x,y);
    while(*str) write_char(*str++);
}

void main(void){
    int i;
    DDRA = 0xFF;
    DDRF = 0xFF;
    returnhome();
    clrscr();
    while(1){
        LCDONOFF(ON);
        delay(1000);
        write_char('s');
        LCDONOFF(OFF);
        delay(1000);
    }
}

```

작업과제 21 텍스트 LCD의 사용자 정의 문자 만들기

[mynew.c](#)

```

#include <iom128v.h>
#include <macros.h>
#include "textlcd.h"

void main(void)
{
    DDRA = 0xFF;
    DDRF = 0xFF;
    InitTLCD();

    WriteStr(0,0,"NEWTC Corp.. Thank you!!!");
    WriteStr(0,1,"KD-128ALL ");
    delay(5000);

    CGRAM_Set(0x00);
    WriteData(0xFF);
}

```

```

WriteData(0xFF);
WriteData(0x11);
WriteData(0x11);
WriteData(0x11);
WriteData(0x11);
WriteData(0xFF);
WriteData(0xFF);
DDRAM_Set(0x00);
WriteData(0x00);

```

```

while(1){
}
}

```

[textlcd.h](#)

```
#include <stdio.h>
```

```

// PORT definition
#define CONTROL      PORTA
#define DATA  PORTF
// Control Pin set up
#define rs(m) m?(CONTROL|=0x01):(CONTROL&=0xFE)
#define rw(m) m?(CONTROL|=0x02):(CONTROL&=0xFD)
#define en(m) m?(CONTROL|=0x04):(CONTROL&=0xFB)

```

// printf 함수 사용시 추가할 것.

```

int putchar(char c)
{
    WriteData(c);
    return c;
}

```

// scanf 함수 사용시 추가할 것.

```

int getchar(void)
{
    unsigned char data;
    return data;
}

```

```

void delay_lcd(void){
    int i=2000;

```

```

        while(i--){
        }
    }

    void delay(int delaytime){
        int i=1000,j;
        while(i--){
            for(j=0;j<delaytime;j++){
            }
        }
    }

    void WriteCommand(unsigned char command){
        rs(0);
        // en(0);delay_lcd();    // LCD에 따라서 이렇게 OFF 시켜 줘야 할 때도 있음.
        rw(0);delay_lcd();
        //en(1);delay_lcd();    // LCD에 따라서 이렇게 OFF 시켜 줘야 할 때도 있음.
        DATA = command;delay_lcd();
        en(0);delay_lcd();
        en(1);delay_lcd();
    }

    void WriteData(unsigned char data){
        rs(1);
        //en(0);delay_lcd();    // LCD에 따라서 이렇게 OFF 시켜 줘야 할 때도 있음.
        rw(0);delay_lcd();
        //en(1);delay_lcd();    // LCD에 따라서 이렇게 OFF 시켜 줘야 할 때도 있음.
        DATA = data;delay_lcd();
        en(0);delay_lcd();
        en(1);delay_lcd();
    }

    unsigned char ReadData(void){
        unsigned char data;
        rs(1);
        en(0);delay_lcd();
        rw(1);delay_lcd();
        en(1);delay_lcd();
        data = DATA;delay_lcd();
        en(0);delay_lcd();
        en(1);delay_lcd();
    }

```

```

void ClearDisplay(void){
    WriteCommand(0x01);
}

void ReturnHome(void){
    WriteCommand(0x02);
}

void EntryModeSet(unsigned char I_D, unsigned char SH){
    WriteCommand(0x04|(I_D<<1)|SH);
}

void DisplayOnOff(unsigned char D, unsigned char C, unsigned char B){
    WriteCommand(0x08|(D<<2)|(C<<1)|B);
}

void Shift(unsigned char S_C, unsigned char R_L){
    WriteCommand(0x10|(S_C<<3)|(R_L<<2));
}

void FunctionSet(unsigned char DL, unsigned char N, unsigned char F){
    WriteCommand(0x20|(DL<<4)|(N<<3)|(F<<2));
}

void CGRAM_Set(unsigned char addr){
    WriteCommand(0x40|(addr&0x3f));
}

void DDRAM_Set(unsigned char addr){
    WriteCommand(0x80|(addr&0x7f));
}

void InitTLCD(void){
    delay(30);
    FunctionSet(1,1,1);
    delay(10);
    FunctionSet(1,1,1);
    delay(1);
    FunctionSet(1,1,1);
    FunctionSet(1,1,1);
    DisplayOnOff(0,0,0);
    DisplayOnOff(1,0,0);
    EntryModeSet(0,0);
}

```

```

        ClearDisplay();
    }

void Move(unsigned char X, unsigned char Y){
    unsigned char addr;
    if((X>40)||((X<0))){ // X : Column
        X=0;
    }
    if((Y>=2)||((Y<0))){ // Y : Line
        Y=0;
    }
    addr = Y*40+X;
    DDRAM_Set(addr);
}

void WriteStr(int X, int Y, char *str){
    Move(X,Y);
    printf(str);
}

```

작업과제 22 그래픽 LCD 구동하기

프로그램 1

```

#include <iom128v.h>
#include <macros.h>

#define MaxLine2
#define MaxCol      16

#define ON          1
#define OFF         0
#define DELAYTIME   0
#define DELAYTIME1  0
#define CS1         100
#define CS2         200

int OneUsCounter=0;
int OneMsCounter=0;

void port_init(void)
{
    DDRA  = 0xFF;
    DDRC  = 0xFF;
    DDRD  = 0xFF;
}

```

```

//TIMER1 initialize - prescale:1
// WGM: 0) Normal, TOP=0xFFFF
// desired value: 1uSec
// actual value: 0.995uSec (0.5%)
void timer1_init(void)
{
    TCCR1B = 0x00; //stop
    TCNT1H = 0xFF; //setup
    TCNT1L = 0xF5;
    OCR1AH = 0x00;
    OCR1AL = 0x0B;
    OCR1BH = 0x00;
    OCR1BL = 0x0B;
    OCR1CH = 0x00;
    OCR1CL = 0x0B;
    ICR1H = 0x00;
    ICR1L = 0x0B;
    TCCR1A = 0x00;
    TCCR1B = 0x01; //start Timer
}

#pragma interrupt_handler timer1_ovf_isr:15
void timer1_ovf_isr(void)
{
    OneUsCounter++;
    if(OneUsCounter >=1000){
        OneMsCounter++;
        OneUsCounter=0;
    }
    //TIMER1 has overflowed
    TCNT1H = 0xFF; //reload counter high value
    TCNT1L = 0xF5; //reload counter low value
}

void delay(int delaytime){
    OneMsCounter=0;
    OneUsCounter=0;
    while(OneMsCounter<=delaytime){}
}

//call this routine to initialize all peripherals
void init_devices(void)
{
    //stop errant interrupts until set up
    SREG = 0x00;
    port_init();
    timer1_init();
    TIMSK = 0x04; //timer interrupt sources
    SREG = 0x80;
    //all peripherals are now initialized

```

```

        OneUsCounter=0;
        OneMsCounter=0;
    }
    void E_Pulse(void){
        char i;
        GLCD_E(ON);
        delay(DELAYTIME1);
        GLCD_E(OFF);
    }

    void Func_set(void){
        GLCD_RW(OFF);
        GLCD_RS(OFF);
        PORTC=0x38;
        E_Pulse();
    }

    void GLCD_RS(int onoff){
        if(onoff==ON)
            PORTD = PORTD|0x04;
        else
            PORTD = PORTD&0xFB;
    }

    void GLCD_RW(int onoff){
        if(onoff==ON)
            PORTD = PORTD|0x08;
        else
            PORTD = PORTD&0xF7;
    }

    void GLCD_E(int onoff){
        if(onoff==ON)
            PORTD = PORTD|0x20;
        else
            PORTD = PORTD&0xDF;
    }

    void GLCD_DI(int onoff){
        if(onoff==ON)
            PORTD = PORTD|0x10;
        else
            PORTD = PORTD&0xEF;
    }

    void GLCD_CS2(int onoff){
        if(onoff==ON)
            PORTD = PORTD|0x02;
        else
            PORTD = PORTD&0xFD;
    }

```

```

}

void GLCD_CS1(int onoff){
    if(onoff==ON)
        PORTD = PORTD|0x01;
    else
        PORTD = PORTD&0xFE;
}

void GLCD_BackLight(int onoff){
    if(onoff==ON)
        PORTD = PORTD|0x40;
    else
        PORTD = PORTD&0xBF;
}

void lcd_disp(char x, char y){
    GLCD_RS(OFF);
    GLCD_RW(OFF);
    if(y==1)y++;
    else if(y==2) y--;
    if(y<=1){
        PORTC=y*20+x+0x80;
    }else{
        y=2;
        PORTC=y*20+x+0xc0;
    }
    E_Pulse();
}

void write_lcd(char x, char y, char *str){
    lcd_disp(x,y);
    while(*str) write_char(*str++);
}

void Init_GLcd(void){
    int i;
    GLCD_RS(OFF);
    GLCD_RS(ON);
    GLCD_E(OFF);
    GLCD_CS1(OFF);
    GLCD_CS2(OFF);
    GLCD_DI(OFF);
}

void display(int onoff){
    if(onoff==ON){
        GLCD_CS1(ON);
        GLCD_CS2(ON);
        TestGLCDcode(0,0,0x3F);
    }
}

```

```

    }else{
        GLCD_CS1(OFF);
        GLCD_CS2(OFF);
        TestGLCDcode(0,0,0x3E);
    }
}

void write_char(char s){
    GLCD_RS(ON);
    PORTC=s;
    E_Pulse();
}

void GLCD_write_char(char chip, char data){
    switch(chip){
        case CS1 : GLCD_CS1(ON);
                    break;
        case CS2 : GLCD_CS2(ON);
                    break;
        default  : break;
    }
    GLCD_RW(OFF);
    write_char(data);
    GLCD_CS1(OFF);
    GLCD_CS2(OFF);
}

void TestGLCDcode(int RW, int DI, char data){
    GLCD_RW(RW);
    GLCD_DI(DI);
    write_char(data);
}

void TestGLCD(void){
    int i=0,j=0, which=0;
    char font7x5[2][5]={0x41, 0x22, 0x1C, 0x22, 0x41}, //character 'X'
                          {0x7E, 0x09, 0x09, 0x09, 0x7E}}; //character 'A'
    for(j=0;j<8;j++){
        TestGLCDcode(0,0,0b10111000+j);
        which = j%2;
        for(i=0;i<64;i++){
            TestGLCDcode(0,0,0x40+i);
            TestGLCDcode(0,1,font7x5[which][i%5]);
        }
    }
}

void TestGLCDCLR(void){
    int i=0,j=0;

```

```

        for(j=0;j<8;j++){
            TestGLCDcode(0,0,0b10111000+j);
            for(i=0;i<64;i++){
                TestGLCDcode(0,0,0x40+i);
                TestGLCDcode(0,1,0x00);
            }
        }
    }
}

void main(void){
    int i,j;
    init_devices();
    Init_GLcd();
    display(ON);
    GLCD_BackLight(ON);
    GLCD_CS1(ON);
    GLCD_CS2(ON);
    while(1){
        GLCD_BackLight(OFF);
        delay(50);
        GLCD_BackLight(ON);
        TestGLCDCLR();
        TestGLCD();
    }
}

```

프로그램 2

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>

```

```

#define MOTOR        PORTE
#define LED           PORTF
#define GLCD_C        PORTA
#define GLCD_D        PORTC
#define T8            unsigned char

```

```

#define RS_ON         GLCD_C=(GLCD_Control|=0x01)
#define RS_OFF        GLCD_C=(GLCD_Control&=0xfe)
#define RW_ON         GLCD_C=(GLCD_Control|=0x02)
#define RW_OFF        GLCD_C=(GLCD_Control&=0xfd)
#define EN_ON         GLCD_C=(GLCD_Control|=0x04)
#define EN_OFF        GLCD_C=(GLCD_Control&=0xfb)
#define CS2_ON        GLCD_C=(GLCD_Control|=0x10)
#define CS2_OFF       GLCD_C=(GLCD_Control&=0xef)
#define CS1_ON        GLCD_C=(GLCD_Control|=0x08)
#define CS1_OFF       GLCD_C=(GLCD_Control&=0xf7)
#define RESET_ON      GLCD_C=(GLCD_Control|=0x20)
#define RESET_OFF     GLCD_C=(GLCD_Control&=0xdf)

```

```

#define CS_ALL          GLCD_C=(GLCD_Control|=0x18)
#define ON              1
#define OFF             0
#define delaytime      1
#define delaytime1     2

```

```

char GLCD_Control;
const char x=0xb8, y=0x40;
int status=0;
char compare[8][128]={0x00,};

```

```

void port_init(void)
{
    PORTA = 0x00;
    DDRA  = 0xFF;
    PORTB = 0x00;
    DDRB  = 0xFF;
    PORTC = 0x00; //m103 output only
    DDRC  = 0xFF;
    PORTD = 0x00;
    DDRD  = 0xFF;
    PORTE = 0xFC;
    DDRE  = 0xFC;
    PORTF = 0x00;
    DDRF  = 0xFF;
    PORTG = 0x00;
    DDRG  = 0x00;
}

```

```

//TIMER0 initialize - prescale:64
// WGM: Normal
// desired value: 1KHz
// actual value: 1.000KHz (0.0%)

```

```

void timer0_init(void)
{
    TCCR0 = 0x00; //stop
    ASSR  = 0x00; //set async mode
    TCNT0 = 0x06; //set count
    OCR0  = 0xFA;
    TCCR0 = 0x04; //start timer
}

```

```

#pragma interrupt_handler timer0_ovf_isr:iv_TIM0_OVF
void timer0_ovf_isr(void)
{
    TCNT0 = 0x06; //reload counter value
    MOTOR ^= 0x04;
}

```

```

//UART0 initialize

```

```

// desired baud rate: 9600
// actual: baud rate:9615 (0.2%)
// char size: 8 bit
// parity: Disabled
void uart0_init(void)
{
    UCSRB = 0x00; //disable while setting baud rate
    UCSRA = 0x00;
    UCSRC = 0x06;
    UBRR0L = 0x67; //set baud rate lo
    UBRR0H = 0x00; //set baud rate hi
    UCSRB = 0x18;
}

```

```

//call this routine to initialize all peripherals
void init_devices(void)
{
    //stop errant interrupts until set up
    CLI(); //disable all interrupts
    XDIV = 0x00; //xtal divider
    XMCRA = 0x00; //external memory
    port_init();
    timer0_init();
    uart0_init();

    MCUCR = 0x00;
    EICRA = 0x00; //extended ext ints
    EICRB = 0x00; //extended ext ints
    EIMSK = 0x00;
    TIMSK = 0x01; //timer interrupt sources
    ETIMSK = 0x00; //extended timer interrupt sources
    SEI(); //re-enable interrupts
    //all peripherals are now initialized
}

```

```

void E_Pulse(void){
    EN_ON;
    delay(delaytime);
    EN_OFF;
}

```

```

void DisplayOnOff(int onoff){
    RW_OFF;
    RS_OFF;
    if(onoff==ON){
        DataInput(0x3F);
    }else{
        DataInput(0x3E);
    }
}

```

```

void DataInput(char data){

```

```

        RS_OFF;
        RW_OFF;
        GLCD_Control = data;
        E_Pulse();
    }

void DisplayStartLine(int displayline){
    DataInput(displayline | 0xc0);
}

void SetX(int X){
    DataInput(X | 0xb8);
}

void SetY(int Y){
    DataInput(Y | 0x40);
}

void Reset(void){
    DataInput(0x10);
}

void Init_Graph(void){
    CS2_ON;
    CS1_ON;
    DisplayOnOff(ON);
    Reset();
}

void delay(int cnt){
    int i , j;
    for(i=0; i<cnt; i++){
        for(j=0; j<100; j++){
        }
    }
}

void GWrite_Control(char CS, char cmd){
    delay(delaytime);
    EN_OFF;
    delay(delaytime);
    RS_OFF;
    delay(delaytime);
    RW_OFF;
    delay(delaytime);
    GLCD_C = CS;
    delay(delaytime);
    EN_ON;
    delay(delaytime);
    GLCD_D = cmd;
    delay(delaytime);
}

```

```

    EN_OFF;
    delay(delaytime);
}

int putchar(char c)
{
    while (((UCSR0A>>UDRE0)&0x01) == 0) ; // UDRE, data register empty

    UDR0 = c;
    return c;
}

void GWrite_Data(char CS, char data){
    delay(delaytime);
    EN_OFF;
    delay(delaytime);
    RS_ON;
    delay(delaytime);
    RW_OFF;
    delay(delaytime);
    GLCD_C = CS;
    delay(delaytime);
    EN_ON;
    delay(delaytime);
    GLCD_D = data;
    delay(delaytime);
    EN_OFF;
    delay(delaytime);
}

void GLcd_Clear(){
    int i, j, x, y;
    GLCD_Control = 0x00;
    GWrite_Control(CS_ALL, 0x3f);
    GWrite_Control(CS_ALL, 0xc0);
    x=0xb8;
    y=0x40;
    for(i=0; i<=7; i++){
        GWrite_Control(CS_ALL, x);
        GWrite_Control(CS_ALL, y);
        for(j=0; j<64; j++){
            GWrite_Data(CS_ALL, 0x00);
        }
        x++;
    }
}

void GLcd_init(){
    RESET_OFF;
    RESET_ON;
}

```

```

    CS_ALL;
    GWrite_Control(CS_ALL, 0x3f);
    GWrite_Control(CS_ALL, 0xc0);
    GWrite_Control(CS_ALL, 0xb8);
    GWrite_Control(CS_ALL, 0x40);
}

void gotoxy(int i, int j, char data){
    int x1=i%128, y1=j%8;
    if(x1<64){
        GWrite_Control(CS_ALL, x+y1);
        GWrite_Control(CS_ALL, y+x1);
        GWrite_Data(CS2_OFF, data);
    }else{
        GWrite_Control(CS_ALL, x+y1);
        GWrite_Control(CS_ALL, y+x1-64);
        GWrite_Data(CS1_OFF, data);
    }
}

void PointXY(int xpoint, int ypoint){
    int point=0x01, value;
    int x1, y1, y2, y3;
    x1=xpoint%128;
    y1=ypoint%64;
    y2=y1/8;
    y3=y1%8;
    compare[y2][x1]|=(point<<y3);
    value=compare[y2][x1];
    if(x1<64){
        GWrite_Control(CS_ALL, x+y2);
        GWrite_Control(CS_ALL, y+x1);
        GWrite_Data(CS2_OFF, value);
    } else {
        GWrite_Control(CS_ALL, x+y2);
        GWrite_Control(CS_ALL, y+x1-64);
        GWrite_Data(CS1_OFF, value);
    }
}

void Point_Del(int xpoint, int ypoint){
    int point=0x01, value;
    int x1, y1, y2, y3;
    x1=xpoint%128;
    y1=ypoint%64;
    y2=y1/8;
    y3=y1%8;
    compare[y2][x1]&=~(point<<y3);
    value=compare[y2][x1];
    if(x1<64){
        GWrite_Control(CS_ALL, x+y2);

```

```

        GWrite_Control(CS_ALL, y+x1);
        GWrite_Data(CS2_OFF, value);
    } else {
        GWrite_Control(CS_ALL, x+y2);
        GWrite_Control(CS_ALL, y+x1-64);
        GWrite_Data(CS1_OFF, value);
    }
}

void Draw_Line(int X0, int Y0, int X1, int Y1){
    int x0=X0, y0=Y0, x1=X1, y1=Y1;
    int dy = y1 - y0;
    int dx = x1 - x0;
    int stepx, stepy;

    if (dy < 0) {
        dy = -dy;
        stepy = -1;
    } else {
        stepy = 1;
    }

    if (dx < 0) {
        dx = -dx;
        stepx = -1;
    } else {
        stepx = 1;
    }

    dy <<= 1;
    dx <<= 1;
    PointXY(x0, y0);
    if (dx > dy) {
        int fraction = dy - (dx >> 1);
        while (x0 != x1) {
            if (fraction >= 0) {
                y0 += stepy;
                fraction -= dx;
            }
            x0 += stepx;
            fraction += dy;
            PointXY(x0, y0);
        }
    } else {
        int fraction = dx - (dy >> 1);
        while (y0 != y1) {
            if (fraction >= 0) {
                x0 += stepx;
                fraction -= dy;
            }
            y0 += stepy;
            fraction += dx;
            PointXY(x0, y0);
        }
    }
}

```

```

    }
    y0 += stepy;
    fraction += dx;
    PointXY(x0, y0);
}
}
}

```

```

void Line_Del(int X0, int Y0, int X1, int Y1){
    int x0=X0, y0=Y0, x1=X1, y1=Y1;
    int dy = y1 - y0;
    int dx = x1 - x0;
    int stepx, stepy;
    if (dy < 0) {
        dy = -dy;
        stepy = -1;
    } else {
        stepy = 1;
    }

    if (dx < 0) {
        dx = -dx;
        stepx = -1;
    } else {
        stepx = 1;
    }

    dy <<= 1;
    dx <<= 1;
    Point_Del(x0, y0);
    if (dx > dy) {
        int fraction = dy - (dx >> 1);
        while (x0 != x1) {
            if (fraction >= 0) {
                y0 += stepy;
                fraction -= dx;
            }
            x0 += stepx;
            fraction += dy;
            Point_Del(x0, y0);
        }
    } else {
        int fraction = dx - (dy >> 1);
        while (y0 != y1) {
            if (fraction >= 0) {
                x0 += stepx;
                fraction -= dy;
            }
            y0 += stepy;
            fraction += dx;
        }
    }
}

```

```

        Point_Del(x0, y0);
    }
}

void Draw_Rect(int x1, int y1, int x2, int y2, int fill){
    if(fill==1){
        if(x1<x2){
            for(x1; x1<=x2; x1++){
                Draw_Line(x1,y1,x1,y2);
            }
        } else {
            for(x1; x1>=x2; x1--){
                Draw_Line(x1,y1,x1,y2);
            }
        }
    } else {
        Draw_Line(x1,y1,x1,y2);
        Draw_Line(x1,y1,x2,y1);
        Draw_Line(x2,y2,x2,y1);
        Draw_Line(x2,y2,x1,y2);
    }
}

void Rect_Del(int x1, int y1, int x2, int y2){
    if(x1<x2){
        for(x1; x1<=x2; x1++){
            Line_Del(x1,y1,x1,y2);
        }
    } else {
        for(x1; x1>=x2; x1--){
            Line_Del(x1,y1,x1,y2);
        }
    }
}

void Draw_Circle(int x1, int y1, int radius, int fill){
    int a, r, cond;
    a = 0;
    r = radius;
    cond = 1 - r;
    do{
        if(fill==1){
            Draw_Line(x1-a, y1+r, x1+a, y1+r);
            Draw_Line(x1-a, y1-r, x1+a, y1-r);
            Draw_Line(x1-r, y1+a, x1+r, y1+a);
            Draw_Line(x1-r, y1-a, x1+r, y1-a);
        } else {
            PointXY(a+x1, r+y1);
            PointXY(r+x1, a+y1);
        }
    }
}

```

```

        PointXY(x1-a, r+y1);
        PointXY(x1-r, a+y1);
        PointXY(x1+r, y1-a);
        PointXY(x1+a, y1-r);
        PointXY(x1-a, y1-r);
        PointXY(x1-r, y1-a);
    }
    if(cond < 0){
        cond += 3 + 2*a++;
    } else {
        cond+= 5 + 2*(a++ - r--);
    }
}while(a <= r);
}

```

```

void Circle_Del(int x1, int y1, int radius){
    int a, r, cond;
    a = 0;
    r = radius;
    cond = 1 - r;
    do{
        Line_Del(x1-a, y1+r, x1+a, y1+r);
        Line_Del(x1-a, y1-r, x1+a, y1-r);
        Line_Del(x1-r, y1+a, x1+r, y1+a);
        Line_Del(x1-r, y1-a, x1+r, y1-a);

        if(cond < 0){
            cond += 3 + 2*a++;
        } else {
            cond+= 5 + 2*(a++ - r--);
        }
    }while(a <= r);
}

```

```

void main(void)
{
    int i, j, k, on=1, off=0;
    T8 Ti, Tj, Li=0;
    char temp;
    int arr[17][4]={{{13,3,29,19},{55,24,71,40},{98,45,114,61},
                    {55,3,71,19},{98,3,114,19},{13,24,29,40},{98,24,114,40},
                    {13,45,29,61},{55,45,71,61},{55,24,71,40},{13,3,29,19},
                    {55,24,71,40},{98,45,114,61},{55,3,71,19},{98,3,114,19},
                    {13,24,29,40},{98,24,114,40}}};
    int arr1[15][2]={{{21,11},{21,32},{21,53},{63,11},{63,32},{63,53},
                    {106,11},{106,32},{106,53},{21,11},{21,53},{63,11},{63,53},
                    {106,11},{106,53}}};

    init_devices();
    GLcd_init();
    GLcd_Clear();
}

```

```

Draw_Line(1,0,126,0);
Draw_Line(1,21,126,21);
Draw_Line(1,42,126,42);
Draw_Line(1,63,126,63);
Draw_Line(0,0,0,63);
Draw_Line(42,0,42,63);
Draw_Line(84,0,84,63);
Draw_Line(127,0,127,63);
Draw_Circle(106,32,8,off);
Draw_Circle(106,53,8,on);
Draw_Rect(55,4,71,18,on);
Draw_Rect(55,46,71,60,off);
Draw_Rect(98,4,114,18,off);
Draw_Circle(63,32,8,off);
Draw_Rect(55,24,71,40,off);
Draw_Circle(21,11,8,off);
Draw_Circle(21,32,8,on);
Draw_Rect(13,46,29,60,on);

while(1){

}
}

```

작업과제 23 A/D 컨버터 활용하기 (가속도 센서, 자이로센서, 적외선센서)

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>

char *str[5] = {"X", "VAR Resistor : ", "Y", "None", "Z"};

void port_init(void)
{
    PORTA = 0x00;
    DDRA  = 0x00;
    PORTB = 0x00;
    DDRB  = 0x00;
    PORTC = 0x00; //m103 output only
    DDRC  = 0x00;
    PORTD = 0x00;
    DDRD  = 0xff;
    PORTE = 0x00;
    DDRE  = 0x00;
    PORTF = 0x00;
    DDRF  = 0x00;
}

```

```

PORTG = 0x00;
DDRG  = 0x03;
}

//UART0 initialize
// desired baud rate: 9600
// actual: baud rate:9615 (0.2%)
// char size: 8 bit
// parity: Disabled
void uart0_init(void)
{
    UCSRB = 0x00; //disable while setting baud rate
    UCSRA = 0x00;
    UCSRC = 0x06;
    UBRR0L = 0x67; //set baud rate lo
    UBRR0H = 0x00; //set baud rate hi
    UCSRB = 0x08;
}

//ADC initialize
// Conversion time: 75uS
void adc_init(void)
{
    ADCSRA = 0x00; //disable adc
    ADMUX = 0x00; //select adc input 0
    ACSR = 0x80;
    ADCSRA = 0xC6;
}

//call this routine to initialize all peripherals
void init_devices(void)
{
    //stop errant interrupts until set up
    CLI(); //disable all interrupts
    XDIV = 0x00; //xtal divider
    XMCRA = 0x00; //external memory
    port_init();
    uart0_init();
    adc_init();

    MCUCR = 0x00;
    EICRA = 0x00; //extended ext ints
    EICRB = 0x00; //extended ext ints
    EIMSK = 0x00;
    TIMSK = 0x00; //timer interrupt sources
    ETIMSK = 0x00; //extended timer interrupt sources

```

```

SEI(); //re-enable interrupts
//all peripherals are now initialized
}

// printf 함수 사용시 추가할 것.
int putchar(char c)
{
    while (((UCSR0A>>UDRE0)&0x01) == 0) ; // UDRE, data register empty

    UDR0 = c;
    return c;
}

// scanf 함수 사용시 추가할 것.
int getchar(void)
{
    while ((UCSR0A & 0x80) == 0);
    return UDR0;
}

// 입력으로 들어오는 채널의 ADC를 스타트 시킨다.
void startConversion(unsigned char ch)
{
    ADCSRA = ADCSRA & 0x3f;
    ADMUX = 0x60 | (ch & 0x0f);
    ADCSRA = ADCSRA | 0xc0;
}

// startConversion() 후에 수행되며 컨버팅 된 값을 리턴한다.
unsigned char readConvertData(void)
{
    volatile unsigned char temp;
    while((ADCSRA & 0x10)==0);
    ADCSRA = ADCSRA | 0x10;
    temp = ADCL;
    temp = ADCH;
    ADCSRA = ADCSRA | 0x10;
    return temp;
}

void delay(int n)
{
    volatile int i,j;
    for(i=1;i<n;i++)
    {
        for(j=1;j<600;j++);
    }
}

```

```

}

void main(void)
{
    volatile unsigned char i, temp;
    init_devices();
    //insert your functional code here...
    printf(" WnWnWrADC test program...WnWr");
    PORTG = 0x0f;
    delay(1000);
    PORTG = 0x00;
    PORTF = 0x00;
    while(1){
        for(i=0; i<5; i=i+2){
            startConversion(i);          // ACC Sensor Converting
            temp = readConvertData();
            printf("%s Channel[%d]  :  %dWnWr", str[i], i, temp);
        }
        delay(600);
        printf("WnWr");
    }
}

```

작업과제 24 I2C를 이용한 RTC DS1307 제어하기 (AVR의 포트 제어 이용)

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>
#include "i2c.h"

void delay(int n);

void port_init(void)
{
    PORTA = 0x00;
    DDRA  = 0xff;
    PORTB = 0x00;
    DDRB  = 0x00;
    PORTC = 0x00; //m103 output only
    DDRC  = 0x0f;
    PORTD = 0x00;
    DDRD  = 0x00;
    PORTE = 0x00;
    DDRE  = 0x00;
}

```

```

    PORTF = 0x00;
    DDRF  = 0x00;
    PORTG = 0x00;
    DDRG  = 0x03;
}

//TIMER0 initialize - prescale:64
// WGM: Normal
// desired value: 1KHz
// actual value: 1.000KHz (0.0%)
void timer0_init(void)
{
    TCCR0 = 0x00; //stop
    ASSR  = 0x00; //set async mode
    TCNT0 = 0x06; //set count
    OCR0  = 0xFA;
    TCCR0 = 0x04; //start timer
}

//UART0 initialize
// desired baud rate: 9600
// actual: baud rate:9615 (0.2%)
// char size: 8 bit
// parity: Disabled
void uart0_init(void)
{
    UCSRB = 0x00; //disable while setting baud rate
    UCSRA = 0x00;
    UCSRC = 0x06;
    UBRR0L = 0x67; //set baud rate lo
    UBRR0H = 0x00; //set baud rate hi
    UCSRB = 0x18;
}

//call this routine to initialize all peripherals
void init_devices(void)
{
    //stop errant interrupts until set up
    CLI(); //disable all interrupts
    port_init();
    uart0_init();
}

```

```

    TIMSK = 0x01; //timer interrupt sources
    SEI(); //re-enable interrupts
    //all peripherals are now initialized
}

// printf 함수 사용시 추가할 것.
int putchar(char c)
{
    while (((UCSR0A>>UDRE0)&0x01) == 0) ; // UDRE, data register empty

    UDR0 = c;
    return c;
}

// scanf 함수 사용시 추가할 것.
int getchar(void)
{
    while ((UCSR0A & 0x80) == 0);
    return UDR0;
}

void main(void)
{
    unsigned char second, min, hour;
    unsigned char old_second, old_min, old_hour;

    init_devices();
    init_ds1307();
    writebyte(0x00,0); //초
    writebyte(0x01,0); //분
    writebyte(0x02,0); //시

    printf("WrWn I2C TestWrWn");
    while(1){
        second = readbyte(0x00);
        min = readbyte(0x01);
        hour = readbyte(0x02);

        second = 10*((second&0x70)>>4)+(second&0x0F);
        min = 10*((min&0x70)>>4)+(min&0x0F);
        hour = 10*((hour&0x10)>>4)+(hour&0x0F);
    }
}

```

```

        if((second != old_second)|| (min != old_min)|| (hour != old_hour)){
            printf("WrWn (Hour:Min:Sec) %d:%d:%d", hour, min, second);
        }

        old_second = second;
        old_min = min;
        old_hour = hour;
    }
}

```

헤더파일 i2c.h

```

// *****
// *** I2C Hardware Interface ***
// *****

#define SDABIT                BIT1
#define SCLBIT                BIT0
#define SDIROUT               (SCLBIT|SDABIT)
#define SDIRIN                0x00
#define I2C_PORT              PORTD
#define I2C_DIR                DDRD
#define I2C_IN                PIND
#define HIGH                  1
#define LOW                   0
#define IN                    1
#define OUT                   0

// enables an unsigned char to be used as a series of booleans
#define BIT(x)                (1 << (x))
#define SETBIT(x, y)          (x |= y)
#define CLEARBIT(x, y)        (x &= ~y)
#define CHECKBIT(x, y)        (x & y)
// ***** Define I/O pins *****
#define BIT7                  0x80
#define BIT6                  0x40
#define BIT5                  0x20
#define BIT4                  0x10
#define BIT3                  0x08
#define BIT2                  0x04
#define BIT1                  0x02

```

```

#define BIT0          0x01
#define ACK           0
#define NACK          1
#define ADDRRTC       0xd0
#define DS1307
#define scl(m)         m? SETBIT(I2C_PORT,SCLBIT):CLEARBIT(I2C_PORT,SCLBIT)
#define sda(m)         m? SETBIT(I2C_PORT,SDABIT):CLEARBIT(I2C_PORT,SDABIT)

```

```

volatile static unsigned char sec=4, min=5, hr=6, dy, dt, mn, yr;

```

```

void delay(int n){
    volatile int i,j;
    for(i=1;i<n;i++){
        for(j=1;j<600;j++);
    }
}

```

```

void I2C_delay(void){
    volatile int i,j;
    for(j=1;j<10;j++);
}

```

```

void I2C_start(void){
    I2C_DIR=SDIROUT;
    sda(1);
    scl(1);I2C_delay();
    sda(0);I2C_delay();
}

```

```

void I2C_stop(void){
    I2C_DIR=SDIROUT;
    scl(1);I2C_delay();
    sda(0);I2C_delay();
    sda(0);I2C_delay();
    sda(1);I2C_delay();
    //I2C_DIR=SDIRIN;
}

```

```

void I2C_write(unsigned char data){
    unsigned char i;
    I2C_DIR=SDIROUT;

```

```

    scl(0);I2C_delay();
    for(i=1;i<=8;i++){
        sda(data>>7);I2C_delay();
        scl(1);I2C_delay();
        data=data<<1;
        scl(0);I2C_delay();
    }
    sda(0);I2C_delay();
    scl(0);I2C_delay();
    scl(1);I2C_delay();
    scl(0);I2C_delay();
    sda(1);I2C_delay();
}

unsigned char I2C_read(unsigned char b)
{
    unsigned char data, i;
    sda(1);I2C_delay();I2C_delay();I2C_delay();I2C_delay();
    scl(0);I2C_delay();I2C_delay();I2C_delay();I2C_delay();
    for(i=1;i<=8;i++){
        scl(1);I2C_delay(); I2C_delay();I2C_delay();I2C_delay();
        I2C_delay();I2C_delay();I2C_delay();I2C_delay();
        I2C_DIR=SDIRIN; I2C_delay();
        data=data<<1; I2C_delay();
        data = data | ((I2C_IN>>1)&0x01);I2C_delay();
        I2C_DIR=SDIROUT; I2C_delay();
        scl(0);I2C_delay(); I2C_delay();
    }
    sda(b); I2C_delay();
    scl(0); I2C_delay();
    scl(1); I2C_delay();

    if(b==NACK){
        I2C_delay();
        sda(1);
        I2C_delay();
    }else{
        while((I2C_IN&0x02));
    }
    scl(0);I2C_delay();
    sda(1);I2C_delay();
}

```

```
        return data;
    }
```

```
unsigned char readbyte(unsigned char addr){
    unsigned char byte;
    I2C_start();
    I2C_write(ADDRRTC);
    I2C_write(addr);
    I2C_start();
    I2C_write(ADDRRTC|1);
    byte = I2C_read(NACK);
    //byte = I2C_read(0);
    I2C_stop();
    return byte;
}
```

```
void writebyte(unsigned char addr, unsigned char data){
    I2C_start();
    I2C_write(ADDRRTC);
    I2C_DIR=SDIROUT;
    I2C_write(addr);
    I2C_write(data);
    I2C_stop();
}
```

```
void init_ds1307(void){
    I2C_start();
    I2C_write(ADDRRTC);
    I2C_write(0x00);
    I2C_write(0x00);
    I2C_stop();

    I2C_start();
    I2C_write(ADDRRTC);
    I2C_write(0x00);
    I2C_write(sec);
    I2C_write(min);
    I2C_write(hr);
    I2C_write(dy);
    I2C_write(dt);
}
```

```

    I2C_write(mn);
    I2C_write(yr);
    I2C_write(0x10); // sqwe:enable 1Hz output
    I2C_stop();
}

```

작업과제 25 I2C를 이용한 RTC DS1307 제어하기(내장 TWI 인터페이스 이용)

```

#include <iom128v.h>
#include <macros.h>
#include <stdio.h>
#include "i2c.h"

void delay(int n);

void port_init(void)
{
    PORTA = 0x00;
    DDRA  = 0xff;
    PORTB = 0x00;
    DDRB  = 0x00;
    PORTC = 0x00; //m103 output only
    DDRC  = 0x0f;
    PORTD = 0x00;
    DDRD  = 0x00;
    PORTE = 0x00;
    DDRE  = 0x00;
    PORTF = 0x00;
    DDRF  = 0x00;
    PORTG = 0x00;
    DDRG  = 0x03;
}

//TIMER0 initialize - prescale:64
// WGM: Normal
// desired value: 1KHz
// actual value: 1.000KHz (0.0%)
void timer0_init(void)
{
    TCCR0 = 0x00; //stop
    ASSR  = 0x00; //set async mode

```

```

    TCNT0 = 0x06; //set count
    OCR0  = 0xFA;
    TCCR0 = 0x04; //start timer
}

//UART0 initialize
// desired baud rate: 9600
// actual: baud rate:9615 (0.2%)
// char size: 8 bit
// parity: Disabled
void uart0_init(void)
{
    UCSR0B = 0x00; //disable while setting baud rate
    UCSR0A = 0x00;
    UCSR0C = 0x06;
    UBRR0L = 0x67; //set baud rate lo
    UBRR0H = 0x00; //set baud rate hi
    UCSR0B = 0x18;
}

//call this routine to initialize all peripherals
void init_devices(void)
{
    //stop errant interrupts until set up
    CLI(); //disable all interrupts
    XDIV  = 0x00; //xtal divider
    XMCRA = 0x00; //external memory
    port_init();
    uart0_init();

    MCUCR = 0x00;
    EICRA = 0x00; //extended ext ints
    EICRB = 0x00; //extended ext ints
    EIMSK = 0x00;
    TIMSK = 0x01; //timer interrupt sources
    ETIMSK = 0x00; //extended timer interrupt sources
    SEI(); //re-enable interrupts
    //all peripherals are now initialized
}

int putchar(char c){

```

```

    while (((UCSR0A>>UDRE0)&0x01) == 0) ; // UDRE, data register empty
    UDR0 = c;
    return c;
}

int getchar(void){
    while ((UCSR0A & 0x80) == 0);
    return UDR0;
}

void main(void)
{
    unsigned char second, min, hour;
    unsigned char old_second, old_min, old_hour;

    init_devices();
    init_ds1307();

    writebyte(SEC,0); //초
    writebyte(MIN,0); //분
    writebyte(HOUR,0); //시
    printf("WrWn I2C TestWrWn");
    while(1){
        second = readbyte(SEC);
        min = readbyte(MIN);
        hour = readbyte(HOUR);

        second = 10*((second&0x70)>>4)+(second&0x0F);
        min = 10*((min&0x70)>>4)+(min&0x0F);
        hour = 10*((hour&0x10)>>4)+(hour&0x0F);

        if((second != old_second)|| (min != old_min)|| (hour != old_hour)){
            printf("WrWn (Hour:Min:Sec) %d:%d:%d", hour, min, second);
        }
        old_second = second;
        old_min = min;
        old_hour = hour;
    }
}

```

헤더파일

```
// *****  
// *** I2C Hardware Interface ***  
// *****  
  
// ID : DS1307  
#define DS1307          0xd0  
  
// CLOCK  
#define I2C_CLK          100000  
#define TWBR_SET        (16000000-(I2C_CLK*16))/(I2C_CLK*8)  
// MODE  
#define WMODE            0x00 // DS1307 Write mode  
#define RMODE            0x01 // DS1307 Read mode  
// ADDRESS  
#define SEC              0x00  
#define MIN              0x01  
#define HOUR             0x02  
#define DAY              0x03  
#define DATE             0x04  
#define MON              0x05  
#define YEAR             0x06  
#define CONTROL          0x07  
  
void delay(int n)  
{  
    volatile int i,j;  
    for(i=1;i<n;i++){  
        for(j=1;j<600;j++);  
    }  
}  
  
void I2C_delay(void)  
{  
    volatile int i,j;  
    for(j=1;j<10;j++);  
}  
void I2C_init(void){  
    DDRD = 0x03;  
    PORTD = 0x03;  
    TWBR = TWBR_SET;
```

```

    TWSR = 0x00;
}

void I2C_start(void){
    TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);
}

void I2C_stop(void){
    TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);
}

void writebyte(unsigned char address, unsigned char data){
    I2C_start();                // START condition 송신
    TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);

    // START condition 신호가 전송완료까지 대기
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x08);
    TWDR = DS1307|WMODE;        // DS1307 address & write mode
    TWCR = 0x84;                // 마스터 송신 모드

    // 전송 완료시 슬레이브에서 ACK 대기
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x18);
    TWDR = address;             // Write address
    TWCR = 0x84;                // 마스터 송신 모드

    // 전송 완료시 슬레이브에서 ACK 대기
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x28);
    TWDR = data;                // write data
    TWCR = 0x84;                // 마스터 송신 모드

    // 전송 완료시 슬레이브에서 ACK 대기
    while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x28);

    TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO); // STOP condition
    I2C_delay();
}

unsigned char readbyte(unsigned char address){
    unsigned char data = 0;
    I2C_start();                // START condition 송신
    TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);

```

```

// START condition 신호가 전송완료까지 대기
while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x08);
TWDR = DS1307|WMODE;          // DS1307 address & write mode
TWCR = 0x84;                  // 마스터 송신 모드

// 전송 완료시 슬레이브에서 ACK 대기
while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x18);
TWDR = address;                // Write address (읽고자 하는 주소)
TWCR = 0x84;                  // 마스터 송신 모드

// 전송 완료시 슬레이브에서 ACK 대기
while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x28);
TWCR = 0xA4;                  // RESTART

// 상태코드가 0x10 확인
while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x10);

TWDR = DS1307|RMODE;          // DS1307 address & read mode
TWCR = 0x84;                  // 마스터 송신 모드

// 전송 완료시 슬레이브에서 ACK 대기
while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x40);
TWCR = 0x84;                  // 마스터 수신대기 모드
// data 수신 대기
while(((TWCR & 0x80) == 0x00) || (TWSR & 0xF8) != 0x58);

data = TWDR;                  // data 수신
TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);    // STOP condition

return data;
}

void init_ds1307(void){
    I2C_init();
    I2C_start();

    writebyte(SEC,1);
    writebyte(MIN,1);
    writebyte(HOUR,1);

```

```

    writebyte(DAY,1);
    writebyte(DATE,1);
    writebyte(MON,1);
    writebyte(YEAR,1);
    writebyte(CONTROL,0x10); // SQWE:enable 1Hz output

    I2C_stop();
}

```

작업과제 26 SSR8 모듈 구동하기

```

// SSR module Control
#include <iom128v.h>
#include <macros.h>

#define SSR_PORT          PORTC
#define SSR_PORT_DDR      DDRC

#define SSR_D0(n)         n? (SSR_PORT|=0x01) : (SSR_PORT&=~0x01) //ch0
#define SSR_D1(n)         n? (SSR_PORT|=0x02) : (SSR_PORT&=~0x02) //ch1
#define SSR_D2(n)         n? (SSR_PORT|=0x04) : (SSR_PORT&=~0x04) //ch2
#define SSR_D3(n)         n? (SSR_PORT|=0x08) : (SSR_PORT&=~0x08) //ch3
#define SSR_D4(n)         n? (SSR_PORT|=0x10) : (SSR_PORT&=~0x10) //ch4
#define SSR_D5(n)         n? (SSR_PORT|=0x20) : (SSR_PORT&=~0x20) //ch5
#define SSR_D6(n)         n? (SSR_PORT|=0x40) : (SSR_PORT&=~0x40) //ch6
#define SSR_D7(n)         n? (SSR_PORT|=0x80) : (SSR_PORT&=~0x80) //ch7

void delay(int delaytime){
    int i,j;
    for(i=0;i<667;i++){
        for(j=0;j<delaytime;j++);
    }
}

int main(void)
{
    SSR_PORT_DDR = 0xff;
    while(1) {
        SSR_D6(0);    SSR_D7(0);    delay(8000); // D6 OFF // D7 OFF
        SSR_D6(1);    SSR_D7(1);    delay(8000); // D6 ON  // D7 ON
    }
    return 0;
}

```

```
}
```

작업과제 27 Toggle8 입력 모듈 구동하기

```
#include <iom128v.h>
#include <macros.h>

#define SWITCH_PORT      PINA
#define SWITCH_PORT_DDR  DDRA

#define LED_PORT         PORTC
#define LED_PORT_DDR     DDRC

void delay(int dt)
{
    int i, j;
    for(i=0;i<=667;i++){
        for(j=0;j<=dt;j++);
    }
}

void main(void)
{
    char keyin;
    SWITCH_PORT_DDR = 0x00;
    LED_PORT_DDR = 0xff;
    while(1){
        keyin = SWITCH_PORT;
        LED_PORT = SWITCH_PORT;
        delay(10);
    }
}
```

작업과제 28 자석감지용 실린더형 리드센서입력 모듈(LEAD8) 구동하기

```
#include <iom128v.h>
#include <macros.h>

#define SWITCH_PORT      PINA
#define SWITCH_PORT_DDR  DDRA

#define LED_PORT         PORTC
#define LED_PORT_DDR     DDRC
```

```

void delay(int dt)
{
    int i, j;
    for(i=0;i<=667;i++){
        for(j=0;j<=dt;j++);
    }
}

void main(void)
{
    char keyin;
    SWITCH_PORT_DDR = 0x00;
    LED_PORT_DDR = 0xff;
    while(1){
        keyin = SWITCH_PORT;
        LED_PORT = SWITCH_PORT;
        delay(10);
    }
}

```

작업 과제 29 DC24V 입력감지 모듈(INPUT-DC8) 구동하기

```

#include <iom128v.h>
#include <macros.h>

#define SWITCH_PORT      PINA
#define SWITCH_PORT_DDR  DDRA

#define LED_PORT         PORTC
#define LED_PORT_DDR     DDRC

void delay(int dt)
{
    int i, j;
    for(i=0;i<=667;i++){
        for(j=0;j<=dt;j++);
    }
}

void main(void)
{

```

```

char keyin;
SWITCH_PORT_DDR = 0x00;
LED_PORT_DDR = 0xff;
while(1){
    keyin = SWITCH_PORT;
    LED_PORT = SWITCH_PORT;
    delay(10);
}
}

```

작업 과제 30 AC220V 입력감지 모듈(INPUT-AC8) 구동하기

```

#include <iom128v.h>
#include <macros.h>

#define SWITCH_PORT      PINA
#define SWITCH_PORT_DDR  DDRA

#define LED_PORT          PORTC
#define LED_PORT_DDR      DDRC

void delay(int dt)
{
    int i, j;
    for(i=0;i<=667;i++){
        for(j=0;j<=dt;j++);
    }
}

void main(void)
{
    char keyin;
    SWITCH_PORT_DDR = 0x00;
    LED_PORT_DDR = 0xff;
    while(1){
        keyin = SWITCH_PORT;
        LED_PORT = SWITCH_PORT;
        delay(10);
    }
}

```

작업과제 31 I2C를 이용한 디지털 입출력 포트확장 모듈(IOEXT) 출력하기

```
#include <iom128v.h>
#include <macros.h>
#include "i2c.h"

void delay_ms(int n);

void delay_ms(int n)
{
    volatile int i,j;
    for(i=1;i<n;i++)
    {
        for(j=1;j<667;j++){
        }
    }
}

void main(void)
{
    unsigned char input;
    init_mcp23017(0x00,0x00,0x00); // OUTPUT setting

    while(1){
        mcp23017_write(MCP23017_GPIOB,0x00, 0xFF);
        mcp23017_write(MCP23017_GPIOA,0x00, 0x00);
        delay_ms(1000);
        mcp23017_write(MCP23017_GPIOB,0x00, 0x00);
        mcp23017_write(MCP23017_GPIOA,0x00, 0xFF);
        delay_ms(1000);
    }
}

헤더파일
// i2c.h
typedef unsigned char u8;
```

```
#define SDABIT                BIT1
#define SCLBIT                BIT0
#define SDIROUT               (SCLBIT|SDABIT)
#define SDIRIN                0x00
#define SDIRIN_SDAONLY       0x01
```

```

#define I2C_PORT          PORTD
#define I2C_DIR           DDRD
#define I2C_IN            PIND
#define HIGH             1
#define LOW              0
#define IN               1
#define OUT              0

#define BIT(x)            (1 << (x))
#define SETBIT(x, y)      (x |= y)
#define CLEARBIT(x, y)    (x &= ~y)
#define CHECKBIT(x, y)    (x & y)

// ***** Define I/O pins *****
#define BIT7              0x80
#define BIT6              0x40
#define BIT5              0x20
#define BIT4              0x10
#define BIT3              0x08
#define BIT2              0x04
#define BIT1              0x02
#define BIT0              0x01

#define ACK               0
#define NACK              1
#define ADDR7C            0xd0
#define DS1307

#define scl(m)             m? SETBIT(I2C_PORT,SCLBIT):CLEARBIT(I2C_PORT,SCLBIT)
#define sda(m)             m? SETBIT(I2C_PORT,SDABIT):CLEARBIT(I2C_PORT,SDABIT)

// MCP23017 register MAP
#define MCP23017_BASE_ADDRESS 0x40
#define MCP23017_IODIRA      0x00
#define MCP23017_IODIRB      0x01
#define MCP23017_GPPUA        0x0C
#define MCP23017_GPPUB        0x0D
#define MCP23017_GPIOA        0x12
#define MCP23017_GPIOB        0x13
#define MCP23017_IOCON        0x0A

// Not using for the future only

```

```

#define MCP23017_OLATA                0x14
#define MCP23017_OLATB                0x15
#define MCP23017_IPOLA                0x02
#define MCP23017_IPOLB                0x03
#define MCP23017_GPINTENA             0x04
#define MCP23017_GPINTENB             0x05
#define MCP23017_DEFVALA              0x06
#define MCP23017_DEFVALB              0x07
#define MCP23017_INTCONA              0x08
#define MCP23017_INTCONB              0x09
#define MCP23017_INTFA                0x0E
#define MCP23017_INTFB                0x0F
#define MCP23017_INTCAPA              0x10
#define MCP23017_INTCAPB              0x11
#define MCP23017_INT_ERR 255

// CLOCK
#define I2C_CLK                        100000
#define TWBR_SET                      (16000000-(I2C_CLK*16))/(I2C_CLK*8)

// MODE
#define WMODE                          0x00    // I2C Write mode
#define RMODE                          0x01    // I2C Read mode

// ADDRESS
#define CONTROL                        0x07

void init_mcp23017(u8 DIPAddress, u8 DIRA, u8 DIRB);
void delay(int n);
void I2C_delay(void);
void I2C_init(void);
void I2C_start(void);
void I2C_stop(void);
void mcp23017_write(u8 reg, u8 DIPAddress, u8 data);
void writebyte(u8 address, u8 DIPAddress, u8 data);
u8 mcp23017_read(u8 reg, u8 DIPAddress);
u8 readbyte(u8 address, u8 DIPAddress);

// Initializing MCP23017
// init_mcp23017(A, DIRA, DIRB)
// A: u8 DIPAddress

```

```

// DIRA: DIRECTION of PORTA : output(0), input(1)
// PUA: PULL UP of PORTA : Pullup(1), noPullup(0)
// DIRB: DIRECTION of PORTB : output(0), input(1)
// PUB: PULL UP of PORTB : Pullup(1), noPullup(0)

void init_mcp23017(u8 DIPaddress, u8 DIRA, u8 DIRB)
{
    u8 PUA, PUB;
    PUA = ~DIRA;
    PUB = ~DIRB;
    I2C_init();
    I2C_start();
    mcp23017_write(MCP23017_IOCON,DIPaddress,0x00); // set addressing style
    mcp23017_write(MCP23017_IODIRA,DIPaddress,DIRA); // set as output PORTA
    mcp23017_write(MCP23017_IODIRB,DIPaddress,DIRB); // set as output PORTB
    mcp23017_write(MCP23017_GPPUA,DIPaddress,PUA); // set as output PORTA
    mcp23017_write(MCP23017_GPPUB,DIPaddress,PUB); // set as output PORTB
}

void delay(int n)
{
    volatile int i,j;
    for(i=1;i<n;i++){
        for(j=1;j<667;j++){
        }
    }
}

void I2C_delay(void)
{
    volatile int i,j;
    for(j=1;j<2;j++){
    }
}

void I2C_init(void){
    DDRD = 0x03;
    PORTD = 0x03;
}

void I2C_start(void){
    I2C_DIR=SDIROUT;
    sda(1);
}

```

```

    scl(1);I2C_delay();
    sda(0);I2C_delay();
    scl(0);I2C_delay();
}

```

```

void I2C_stop(void){
    I2C_DIR=SDIROUT;
    scl(0);
    sda(0);I2C_delay();
    scl(1);I2C_delay();
    sda(1);I2C_delay();
    I2C_DIR=SDIRIN;
}

```

```

void I2C_write(u8 data){
    u8 i;
    I2C_DIR=SDIROUT;
    scl(0);I2C_delay();
    for(i=1;i<=8;i++){
        sda(data>>7);I2C_delay();
        scl(1);I2C_delay();
        data=data<<1;
        scl(0);I2C_delay();
    }
    sda(0);I2C_delay();
    scl(0);I2C_delay();
    scl(1);I2C_delay();
    scl(0);I2C_delay();
    sda(1);I2C_delay();
}

```

```

void mcp23017_write(u8 reg, u8 DIPaddress, u8 data)
{
    writebyte(reg, DIPaddress, data);
}

```

```

void writebyte(u8 address, u8 DIPaddress, u8 data)
{
    I2C_start();
    I2C_write(MCP23017_BASE_ADDRESS|(DIPaddress*2)|WMODE);
}

```

```
I2C_DIR=SDIROUT;  
I2C_write(address);  
I2C_write(data);  
I2C_stop();  
}
```