

2차 프로젝트 결과보고서

2024-50924 이상욱

목차

- 개요
 - 프로젝트 목적
 - 구현 기능
- 주요 코드 설명
- 회로도

1. 개요 및 기능 설명

A. 프로젝트 목적

ATTiny4313의 한정된 pin으로 최대한 많은 모듈을 동작 시킨다.

수업에서 지금까지 배웠던 것을 응용하여 HD44780, MAX7219, SH1106을 동시에 display한다. Interrupt를 활용하여 rotary switch, keyboard input을 처리한다. Timer를 활용한 PWM 제어를 통해 LED의 밝기를 조절한다. 또한 UART의 RX기능을 software 적으로 구현한다. TX기능은 ATTiny4313의 Timer는 총 2개이므로 하드웨어로 정의된 기능을 사용하여 구현한다.

B. 구현기능

Rotary switch를 돌릴 때마다 interrupt를 발생시켜 숫자를 증감시킨다. 증감시킨 숫자를 대응시켜 PWM의 OCR0B의 값을 변화시켜 LED의 밝기를 제어하고 밝기의 값을 SH1106에 display한다.

Keyboard가 눌리면 interrupt를 발생시켜 값을 ASCII코드 값으로 변환시킨 후 CP2102를 통해 PC로 전송한다. 전송시에는 칩 내부에 정의되어 있는 UART의 TX기능을 활용하여 값을 전송한다. 또한 ASCII코드 값 및 stop bit, parity bit, data, start bit를 MAX7219에 출력하여 값을 확인한다.

2. 주요 코드 설명

초기화 부분은 생략하도록 하겠습니다.

```
ISR(PCINT1_vect) {
```

```
    uint8_t a = ((ROTARY_A_PIN >> ROTARY_A_PIN_BIT)&1); /// A 핀의 값을 읽음.
    uint8_t b = ((ROTARY_B_PIN >> ROTARY_B_PIN_BIT)&1); /// B 핀의 값을 읽음.
    if(a != a0){

        if(a!=b){/// A 와 B 값이 다르면 counter 증가;
            counter++;
        }else{/// 그렇지 않으면 counter 감소;
            counter--;
        }
        if(counter > 250){/// counter 가 199 보다 크면 0으로 초기화
            counter=0;
        }else if(counter<0){/// counter 가 0 보다 작으면 199로 초기화
            counter=250;
        }
        a0= a;
    }
}
```

255가 OCR register의 최대값이므로 250을 기준으로 counter값이 증가, 감소 할 수 있도록 했습니다.

```

static void lcd_write_nibble(uint8_t rs, uint8_t data){
    LCD_RS_PORT &=~(1<<LCD_RS_BIT);
    LCD_RS_PORT |= (rs<<LCD_RS_BIT);
    LCD_EN_PORT |= (1<<LCD_EN_BIT);
    LCD_DB_PORT&=~(0XF0);
    data&=~(0x0F);
    LCD_DB_PORT|=data;
    LCD_EN_PORT&=~(1<<LCD_EN_BIT);
}

static void lcd_write_byte(uint8_t rs, uint8_t data){
    lcd_write_nibble(rs,data);//상위 비트
    lcd_write_nibble(rs,data<<4);//하위 비트
}

```

기존 실습시에는 PB0부터 PB3까지 pin을 사용하였으므로 하위비트로 밀어서 PORTB에 전송하였지만 이번 회로에서는 PB4부터 PB7까지로 data pin을 설정했으므로 상위비트로 밀어서 전달한다.

```

void keyboard_func(){
    uint8_t temp;
    if (scanCode) {
        temp = ps2_scan_to_ascii(scanCode);
        if(temp != 0){
            //// scanCode 의 값 및 각 정보를 MAX7219 에 표시.
            MAX_CS_PORT &= ~(1<<MAX_CS_BIT);
            SpiUSITx(0X01);
            SpiUSITx(scanCode&(0x001));
            MAX_CS_PORT |= (1<<MAX_CS_BIT);

            MAX_CS_PORT &= ~(1<<MAX_CS_BIT);
            SpiUSITx(0X02);
            SpiUSITx((scanCode>>1) & (0x00F));
            MAX_CS_PORT |= (1<<MAX_CS_BIT);

            MAX_CS_PORT &= ~(1<<MAX_CS_BIT);
            SpiUSITx(0X03);
            SpiUSITx((scanCode>>5)&(0x00F));
            MAX_CS_PORT |= (1<<MAX_CS_BIT);

            MAX_CS_PORT &= ~(1<<MAX_CS_BIT);//parity bit
            SpiUSITx(0X04);
            SpiUSITx((scanCode>>9)&(0x001));
            MAX_CS_PORT |= (1<<MAX_CS_BIT);

            MAX_CS_PORT &= ~(1<<MAX_CS_BIT);
            SpiUSITx(0X05);
            SpiUSITx(scanCode>>10);
            MAX_CS_PORT |= (1<<MAX_CS_BIT);

```

```

        MAX_CS_PORT &= ~(1<<MAX_CS_BIT);
        SpiUSITx(0X07);
        SpiUSITx(temp & (0x0F));
        MAX_CS_PORT |= (1<<MAX_CS_BIT);

        MAX_CS_PORT &= ~(1<<MAX_CS_BIT);
        SpiUSITx(0X08);
        SpiUSITx(temp>>4);
        MAX_CS_PORT |= (1<<MAX_CS_BIT);
        uart_tx(temp);
    }
    scanCode = 0;
}
}

```

키보드에 ASCII값을 전송하기 위해 uart_tx()함수로 8비트씩 PC로 전송한다.

```

void rotary_func(){
    if(counter != prev_counter){
        prev_counter = counter;
        OCR0B = 250-counter;

        sh1106_text_font24(char_addr[counter/100],char_width[counter/100],4,34);

        sh1106_text_font24(char_addr[counter/10%10],char_width[counter/10%10],4,34+16*1);

        sh1106_text_font24(char_addr[counter%10],char_width[counter%10],4,34+16*2);
    }
}

```

OCR0B의 값이 작을수록 LED의 밝기는 감소함으로 counter의 값을 250을 기준으로 반전 시켜서 적용한다.

```

int main(void) {

    keyboard_init();//interrupt
    uart_init();//tx init
    rx_sw_init();//rx init
    rotary_init();
    sei();
    led_pwm_init();//pwm init

    //display things initial
    max7219_init();
    sh1106_init();

    USICR &= ~(1<<USIWM0);
    lcd_init();
    lcd_write_byte(0, 0x01);
    lcd_wait();
}

```

```

sh1106_clear();
sh1106_border();

USICR &= ~(1<<USIWM0);

while (1) {
    rotary_func();
    keyboard_func();// keyboard display
    if(s_rxByte){
        USICR &= ~(1<<USIWM0);//USCK mode off
        lcd_write_byte(1, s_rxByte);
        lcd_wait();
        s_rxByte = 0;
        num++;
        if(num==16){
            lcd_write_byte(0,0x80+0x40);
            lcd_wait();
        }
        if(num==32){
            lcd_write_byte(0, 0x01); // display clear
            lcd_wait();
            num=0;
        }
    }
}

```

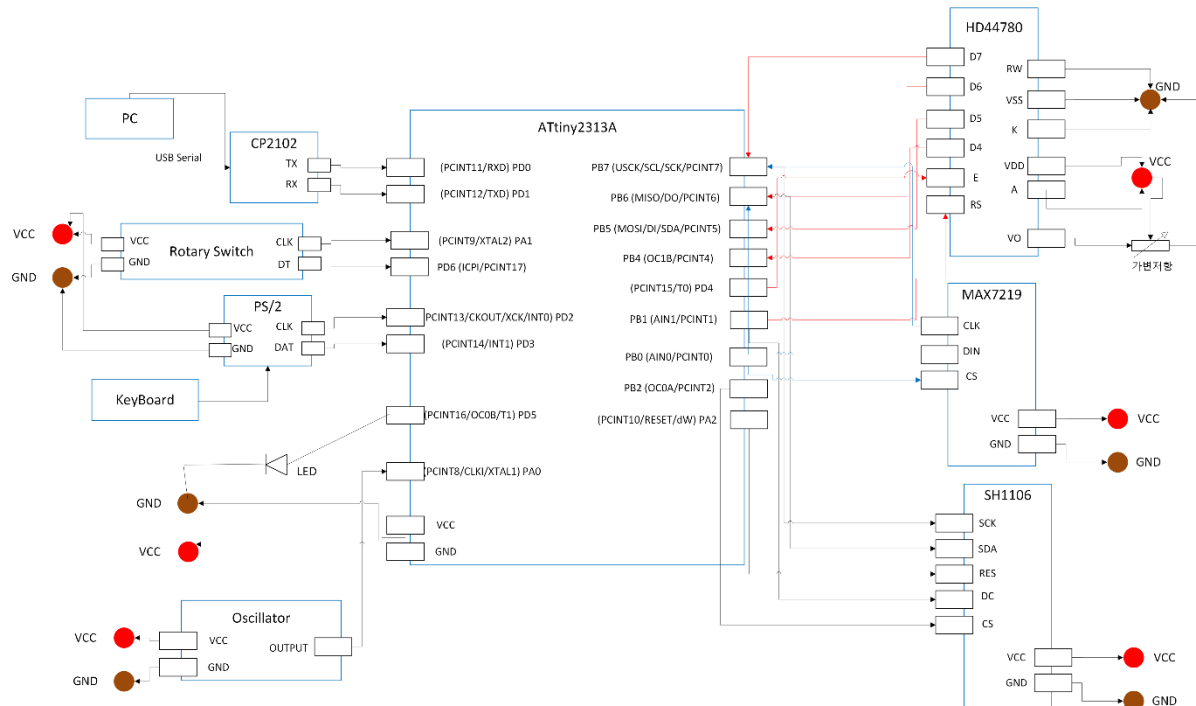
전체 main함수이다. s_rxByte의 값이 0이 아니면 lcd에 값을 출력한다. 또한 HD44780의 한 줄의 최대값을 16이므로 16개를 전송하면 줄을 바꾸는 명령어를 전송하고 32개를 전송하면 화면을 모두 지운다. 또한 HD44780의 data pin과 SPI의 CLK과 DO핀을 공유하므로 다음 표를 참고하여 레지스터 값을 설정해 주어야 한다.

USIWM1	USIWM0	Description
0	0	Outputs, clock hold, and start detector disabled. Port pins operate as normal.
0	1	Three-wire mode. Uses DO, DI, and USCK pins. The <i>Data Output</i> (DO) pin overrides the corresponding bit in the PORTA register. However, the corresponding DDRA bit still controls the data direction. When the port pin is set as input the pin pull-up is controlled by the PORTA bit. The <i>Data Input</i> (DI) and <i>Serial Clock</i> (USCK) pins do not affect the normal port operation. When operating as master, clock pulses are software generated by toggling the PORTA register, while the data direction is set to output. The USITC bit in the USICR Register can be used for this purpose.

Datasheet p.164

SPI를 사용하지 않고 data pin의 용도로 사용할 때는 USICR의 USIWM0를 0으로 설정한다.

3. 회로도



PB4~PB7까지는 HD44780의 D4~D7과 연결된다. PB7,PB6는 MAX7219와 SH1106과 공유된다. HD44780의 E, MAX7219의 CS, SH1106의 CS는 각각 PD4, PB0, PB2에 연결되면 이 pin들은 공유될 수 없다. HD44780의 RS와 SH1106은 PB1으로 공유된다.

PD0는 RX로 사용되며 software로 정의해서 사용하였고 CP2102의 TX와 연결된다.

PD1은 TX로 사용되며 CP2102의 RX와 연결된다.

Rotary switch의 CLK, DT는 각각 PA1, PD6에 연결된다.

PS/2 Connector의 CLK, DAT는 각각 PD2, PD3에 연결된다.

LED는 PWM 사용을 위해 OCROB가 있는 PD5에 연결된다.