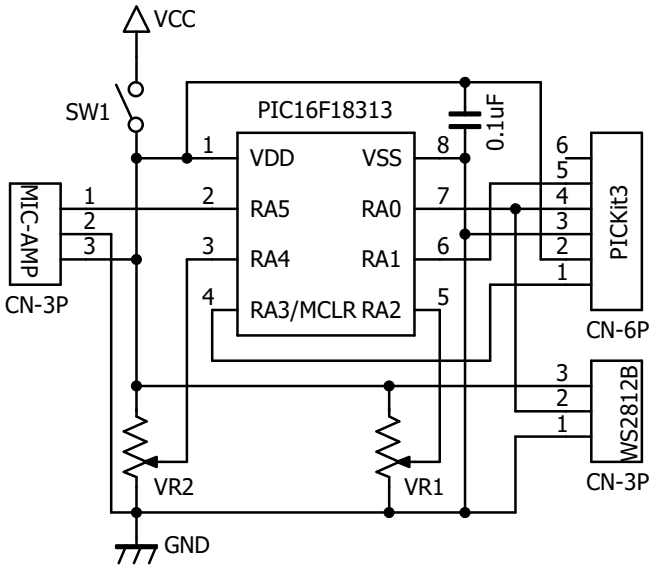

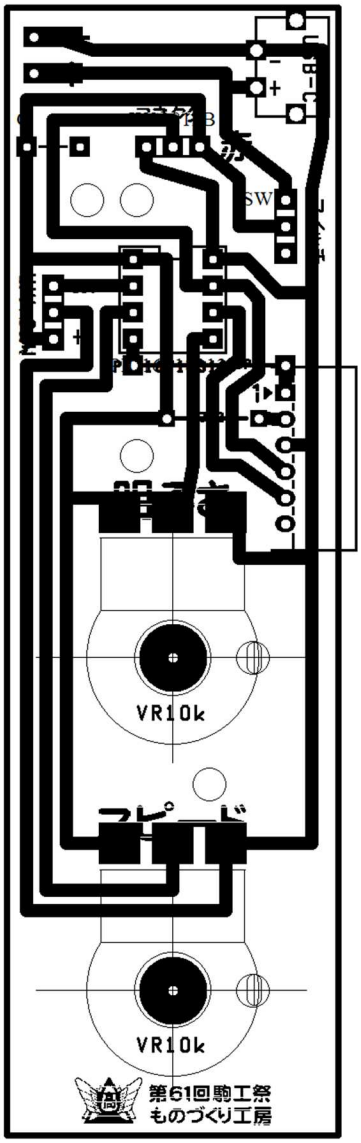


```
//---------------------------------------
//  R6 駒工祭　モノづくり工房　電子工作
//  「LED テープライトコントローラー」
//  PIC16F18313　　MAX188 ver
//  2024/9/29　K.Kobayashi Ver1.5
//---------------------------------------

// CONFIG1
#pragma config FEXTOSC = OFF　　// FEXTOSC External Oscillator mode Selection bits (Oscillator not enabled)
#pragma config RSTOSC = HFINT32 // Power-up default value for COSC bits (HFINTOSC with 2x PLL (32MHz))
#pragma config CLKOUTEN = OFF　　// Clock Out Enable bit (CLKOUT function is disabled; I/O or oscillator function on OSC2)
#pragma config CSWEN = OFF　　// Clock Switch Enable bit (The NOSC and NDIV bits cannot be changed by user software)
#pragma config FCMEN = OFF　　// Fail-Safe Clock Monitor Enable (Fail-Safe Clock Monitor is disabled)

// CONFIG2
#pragma config MCLRE = OFF　　// Master Clear Enable bit (MCLR/VPP pin function is MCLR; Weak pull-up enabled )
#pragma config PWRTE = OFF　　// Power-up Timer Enable bit (PWRT disabled)
#pragma config WDTE = OFF　　// Watchdog Timer Enable bits (WDT disabled; SWDTEN is ignored)
#pragma config LPBOREN = OFF　　// Low-power BOR enable bit (ULPBOR disabled)
#pragma config BOREN = OFF　　// Brown-out Reset Enable bits (Brown-out Reset disabled)
#pragma config BORV = LOW　　// Brown-out Reset Voltage selection bit (Brown-out voltage (Vbor) set to 2.45V)
#pragma config PPS1WAY = OFF　　// PPSLOCK bit One-Way Set Enable bit (The PPSLOCK bit can be set and cleared repeatedly (subject to the unlock sequence))
#pragma config STVREN = OFF　　// Stack Overflow/Underflow Reset Enable bit (Stack Overflow or Underflow will not cause a Reset)
#pragma config DEBUG = OFF　　// Debugger enable bit (Background debugger disabled)

// CONFIG3
#pragma config WRT = OFF　　// User NVM self-write protection bits (Write protection off)
#pragma config LVP = OFF　　// Low Voltage Programming Enable bit (HV on MCLR/VPP must be used for programming.)

// CONFIG4
#pragma config CP = OFF　　// User NVM Program Memory Code Protection bit (User NVM code protection disabled)
#pragma config CPD = OFF　　// Data NVM Memory Code Protection bit (Data NVM code protection disabled)

#include <xc.h>
//#include <stdlib.h>

#define _XTAL_FREQ 32000000 //32MHz
```

```
void ws2812b_init(void);
void ws2812b_reset(unsigned int led_number);
void ws2812b_flash(unsigned char r,unsigned char g,unsigned char b);
void ws2812b_end(void);

#define MAX_LED_MAX 80 // PIC16F18313 MAX 188LEDs
#define MAX_LUM 250 // 最大輝度（MAX 255）
unsigned char LUM = 128; // 輝度 変数
unsigned char SPEED = 128; // スピード 変数
unsigned char MIC = 128; // マイク 変数

unsigned char c[8][3] ={{ 0, 0, 0}, // 黒
                        {128, 0, 0}, // 赤
                        { 0,128, 0}, // 緑
                        {128,128, 0}, // 黄
                        { 0, 0,128}, // 青
                        {128, 0,128}, // 紫
                        { 0,128,128}, // 水色
                        {128,128,128}}; // 白

unsigned char C[MAX_LED_MAX];
unsigned char MAX_LED=60; //LED 数（最大）

void adconv(){
    ADCON1 = 0b00100000; // 左詰め出力、AD 変換クロック Fosc/32,正基準電圧 VDD
    ADCON0 = 0b00001001; // アナログ入力 AN2(RA2),ADC ON)
    __delay_ms(5);
    GO = 1; // AD 変換スタート
    while(GO); // 処理待ち
    LUM = ADRESH; // 輝度 VDD/256
    ADCON0 = 0b00001000; // アナログ入力 AN2(RA2),ADC OFF)

    ADCON0 = 0b00010001; // アナログ入力 AN4(RA4),ADC ON)
    __delay_ms(5);
    GO = 1; // AD 変換スタート
    while(GO); // 処理待ち
    SPEED = ADRESH; // スピード
    ADCON0 = 0b00010000; // アナログ入力 AN4(RA4),ADC OFF)
    LUM = LUM>MAX_LUM?MAX_LUM:LUM+1; // 輝度が最大以下で、0 にならないように
}

void adconv_mic(){
    ADCON1 = 0b00100000; // 左詰め出力、AD 変換クロック Fosc/32,正基準電圧 VDD
    ADCON0 = 0b00010101; // アナログ入力 AN5(RA5),ADC ON)
    __delay_ms(10);
    GO = 1; // AD 変換スタート
    while(GO); // 処理待ち
    MIC = ADRESH; // MIC_AMP
    ADCON0 = 0b00010100; // アナログ入力 AN5(RA5),ADC OFF)
}

void my_delay(unsigned int t){
    for(unsigned int i=0; i<t;i++)
        __delay_us(900);
}
void lum_set(int lum)
{
    int i,j;

    if(lum<256){
        for( i=0; i<8; i++ ){
            for( j=0; j<3; j++ ){
                c[i][j] = c[i][j]!=0?lum:0;
            }
        }
    }
    else{
        adconv();
        for( i=0; i<8; i++ ){
            for( j=0; j<3; j++ ){
                c[i][j] = c[i][j]!=0?LUM:0;
            }
        }
    }
}

void ws2812b_no1tono2_flash(unsigned char no1, unsigned char no2, unsigned char col,unsigned char lum)
{
    unsigned char a;

    for(a=0;a<MAX_LED;a++){
        C[a] = 0;
    }
    for(unsigned char i=no1;i<=no2;i++){
        C[i] = col;
    }

    lum_set(lum);
    for(a=0;a<MAX_LED;a++){
        //ws2812b_flash( r[a], g[a], b[a]);
        ws2812b_flash( c[ C[a] ][0] , c[ C[a] ][1] , c[ C[a] ][2] );

    }
    ws2812b_end();
}
/*
void ws2812b_all_flash(unsigned char col,unsigned char i)
{
    unsigned char r,g,b;

    r = c[col][0]!=0?i:0;
    g = c[col][1]!=0?i:0;
    b = c[col][2]!=0?i:0;
```

```
    for(unsigned char a=0;a<30;a++){
        ws2812b_flash(r,g, b);
    }
    ws2812b_end();
    adconv();
}
 */

void ws2812b_rotate_left(unsigned char n)
{
    unsigned char cc,i;
    cc = C[MAX_LED-1];
    for(i=MAX_LED-1;i>0;i--){
        C[i] = C[i-1];
    }
    C[0] = cc;
}

void ws2812b_rotate_right(unsigned char n)
{
    unsigned char cc,i;
    cc = C[0];
    for(i=0;i<MAX_LED;i++){
        C[i] = C[i+1];
    }
    C[MAX_LED-1] = cc;
}
/*
void ws2812b_shift_left(unsigned char mode)
{
    unsigned char i;
    for(i=29;i>0;i--){
        r[i] = r[i-1]!=0?LUM:0;
        g[i] = g[i-1]!=0?LUM:0;
        b[i] = b[i-1]!=0?LUM:0;
    }
    if(mode==0){
        r[0] = g[0] = b[0] = 0;
    }
    else{
         r[0] = r[1]; g[0] = g[1]; b[0] = b[1];
   }
}

void ws2812b_shift_right(unsigned char n)
{
    unsigned char i;
    for(i=0;i<30;i++){
        r[i] = r[i+1]!=0?LUM:0;
        g[i] = g[i+1]!=0?LUM:0;
        b[i] = b[i+1]!=0?LUM:0;
    }
    r[29] = g[29] = b[29] = 0;
}
*/
unsigned char ws2812b_rotate_roop(void)
{
    unsigned char i,k,spd;
    lum_set(300);
    for(i=0;i<MAX_LED;i++){
         ws2812b_flash(   c[ C[i]][0] , c[ C[i] ][1] , c[ C[i] ][2] );
    }
    if(SPEED<128){
        spd=128-(128-SPEED);
        ws2812b_rotate_left(1);
    }
    else{
        spd=256-(SPEED);
        ws2812b_rotate_right(1);
    }
    my_delay(spd*3);
}

unsigned char COL=1;

void ws2812b_demo0(void){
    for(unsigned char col=1; col<=7;col++){
        lum_set(300);
        ws2812b_no1tono2_flash(0,MAX_LED-1,col,LUM);
        my_delay(SPEED*6);
    }
    ws2812b_end();
}

void ws2812b_demo1(void){
    unsigned char p;
    p = 1; //LUM/64+1;
    adconv();
    for(unsigned char lum=1;lum<LUM;lum+=p){
        ws2812b_no1tono2_flash(0,MAX_LED-1,COL,lum);
        my_delay(SPEED/20);
        //my_delay(SPEED/lum*2);
    }

    for(unsigned char lum=LUM;lum>0;lum-=p){
        ws2812b_no1tono2_flash(0,MAX_LED-1,COL,lum);
        my_delay(SPEED/20);
        //my_delay(SPEED/lum*2);
    }
    ws2812b_no1tono2_flash(0,MAX_LED-1,0,1);
    //my_delay(SPEED/lum*2);
    COL = COL==7?1:COL+1;
    ws2812b_end();
}

void ws2812b_demo2(void){
    for(unsigned char col=1; col<=7; col++){
        lum_set(300);
        for(unsigned char i=0; i<MAX_LED-2; i++){
            ws2812b_no1tono2_flash(i,i+2,col,LUM);
            my_delay(SPEED/4);
            //lum_set(300);
        }
```

```
            lum_set(300);
            for(unsigned char i=MAX_LED-3; i>0; i--){
                ws2812b_no1tono2_flash(i,i+2,col,LUM);
                my_delay(SPEED/4);
                //lum_set(300);
            }
        }
}

void ws2812b_demo3(void){
        unsigned char spd;
        ws2812b_no1tono2_flash(0,MAX_LED-1,0,1);
        lum_set(300);
        if(SPEED<128){
            spd=128-(128-SPEED);
            for(unsigned char no0=0;no0<MAX_LED;no0++){
                ws2812b_no1tono2_flash(0,no0,COL,LUM);
                my_delay(spd/2);
            }
            for(unsigned char no0=0;no0<MAX_LED;no0++){
                ws2812b_no1tono2_flash(no0,MAX_LED-1,COL,LUM);
                my_delay(spd/2);
            }
        }
        else{
            spd=256-(SPEED);
            for(unsigned char no0=MAX_LED;no0>0;no0--){
                ws2812b_no1tono2_flash(no0,MAX_LED-1,COL,LUM);
                my_delay(spd/2);
            }
            for(unsigned char no0=MAX_LED;no0>0;no0--){
                ws2812b_no1tono2_flash(0,no0,COL,LUM);
                my_delay(spd/2);
            }
        }
        COL = COL==7?1:COL+1;
}
/*
void ws2812b_demo4(void){
        unsigned char i,col,rr,gg,bb;
        while(1){
            for(col=1;col<8;col++){
                rr = c[col][0]!=0?LUM:0;
                gg = c[col][1]!=0?LUM:0;
                bb = c[col][2]!=0?LUM:0;
                for(i=0;i<5;i++){
                    r[i] = rr; g[i] = gg;    b[i] = bb;
                }
                ws2812b_rotate_roop();
            }
        }
}
*/

void ws2812b_demo4(void){
        unsigned char i,col;
        for(i=0;i<MAX_LED;i+=3){
            C[i    ] = COL;
            C[i+1] = 0 ;
            C[i+2] = 0 ;
        }
        for(i=0;i<MAX_LED;i++)
            ws2812b_rotate_roop();
        COL = COL==7?1:COL+1;
}

void ws2812b_demo5_set(void){
        unsigned char i,j;
        for(j=0;j<MAX_LED;j+=15){
            for(i=j;i<j+5;i++){
                C[i     ] = 1;
                C[i+ 5] = 2;
                C[i+10] = 4;
            }
        }
}

void ws2812b_demo6(void){
        for(unsigned char col=1; col<=7;col++){
            lum_set(300);
            ws2812b_no1tono2_flash(0,MAX_LED-1,col,LUM);
            my_delay(SPEED);
            ws2812b_no1tono2_flash(0,MAX_LED-1,0,LUM);
            my_delay(SPEED);
            ws2812b_no1tono2_flash(0,MAX_LED-1,col,LUM);
            my_delay(SPEED);
            ws2812b_no1tono2_flash(0,MAX_LED-1,0,LUM);
            my_delay(SPEED*10);
        }
        ws2812b_end();
}

void ws2812b_mic(){
        unsigned char mic,ledcol;
        adconv();
        //ledcol=SPEED/36+1;
        adconv_mic();
        if(MIC>128){
            mic = (MIC-128)/(128/MAX_LED+1);
            ledcol = 1;mic/(128/7+1);
            ws2812b_no1tono2_flash(0,mic,ledcol,LUM);
            //my_delay(SPEED/5);
        }
}

void led_max_set()
{
        MAX_LED = MAX_LED_MAX;
        while(1){
            adconv();
            ws2812b_no1tono2_flash(0,LUM,4,100);
            ws2812b_no1tono2_flash(0,SPEED,1,100);
            if( LUM<=SPEED) break;
```

```
    }
    eeprom_write(1,LUM);
    __delay_ms(100);
    ws2812b_no1tono2_flash(0,LUM,1,100);
    while(1);
}

__EEPROM_DATA(0,15,0,0,0,0,0,0);
void main(void) {
    unsigned char mode,next_mode,leds;
START:
    ws2812b_init();
    __delay_ms(100);
    ws2812b_reset(MAX_LED_MAX);

    mode = eeprom_read(0);
    MAX_LED =eeprom_read(1);
    __delay_ms(100);

    adconv();
    if(LUM<5 && SPEED<5)
        led_max_set();

    if(mode < 7 )
        next_mode = mode+1;
    else
        next_mode = 0;
    eeprom_write(0,next_mode);
    __delay_ms(100);

    while(1){
      //mode = 7;
        switch( mode ){
            case 0 : ws2812b_demo0(); break;
            case 1 : ws2812b_demo1(); break;
            case 2 : ws2812b_demo2(); break;
            case 3 : ws2812b_demo3(); break;
            case 4 : ws2812b_demo4(); break;
            case 5 : ws2812b_demo5_set();
                     while(1) ws2812b_rotate_roop(); break;
            case 6 : ws2812b_demo6(); break;
            case 7 : //adconv_mic();
                     //if(MIC<80) goto START;
                     ws2812b_mic();   break;
        }
    }
}
void ws2812b_init(void){
    //ピン設定
    ANSELA = 0b00110100;  // RA5,RA4,RA2 アナログ、他はデジタル
    //ANSELA = 0b00010100;  //      RA4,RA2 アナログ、他はデジタル
    TRISA  = 0b00110100;  // RA5,RA4,RA2 入力設定
    //ANSA2 = 1;            // ANA2(RA2)アナログ入力
    //ANSA4 = 1;            // ANA4(RA4)アナログ入力
    ADACT  = 0b00000000;  // AD 変換はプログラムの命令で開始
    //ADCON0 = 0b00001001;  // アナログ入力 AN2(RA2),ADCON)
    //ADCON1 = 0b00100000;  // 左詰め出力、AD 変換クロック Fosc/32,正基準電圧 VDD

    //PORTA = 0;
    //RA5PPS = 0x04;    //RA5->CLC1:CLC1OUT;
    RA0PPS = 0x04;    //RA0->CLC1:CLC1OUT;
    //Timer2 設定
    PR2 = 0x09;
    TMR2 = 0x00;
    PIR1bits.TMR2IF = 0;
    T2CON = 0x04;
        //PWM5 設定
    PWM5CON = 0x80;
    PWM5DCH = 0x02;
    PWM5DCL = 0x40;

    //SPI1 設定
    SSP1STAT = 0x00;
    SSP1CON1 = 0x23;
    SSP1ADD = 0x00;
        //CLC1 設定
    CLC1POL = 0x01;
    CLC1SEL0 = 0x10;
    CLC1SEL1 = 0x12;
    CLC1SEL2 = 0x13;
    CLC1SEL3 = 0x00;
    CLC1GLS0 = 0x05;
    CLC1GLS1 = 0x10;
    CLC1GLS2 = 0x08;
    CLC1GLS3 = 0x20;
    CLC1CON = 0x80;
}
void ws2812b_reset(unsigned int led_number){
    for(unsigned int i=0;i<led_number;i++){
        ws2812b_flash(0,0,0);
    }
    __delay_us(100);
}
void ws2812b_flash(unsigned char r,unsigned char g,unsigned char b){
    SSP1CON1bits.WCOL = 0;
    SSPBUF = g;
    while(SSP1STATbits.BF == 0);

    SSP1CON1bits.WCOL = 0;
    SSPBUF = r;
    while(SSP1STATbits.BF == 0);

    SSP1CON1bits.WCOL = 0;
    SSPBUF = b;
    while(SSP1STATbits.BF == 0);
}
void ws2812b_end(void){
    __delay_us(100);
}
```