

INSTITUTO TECNOLÓGICO SUPERIOR DE TANTOYUCA

ILUMINACION RGBW PARA MURAL ITSTA

Autores: **Daniel Froylán Montero Gómez**. (Software)

Tonateolt Téllez del Ángel (Creador del diseño y hardware)

El proyecto consistió en iluminar un mural junto con el corredor donde se encuentra este, usando reflectores que brindaran buena iluminación y además que fuesen de bajo consumo de potencia.

La solución encontrada fue utilizar leds ultra brillantes de potencia, armando tres reflectores que contenían 4 leds ultra brillantes cada uno con los colores RGBW, es decir ROJO, VERDE, AZUL Y BLANCO (correspondientes a sus siglas en ingles).

La tecnología RGBW es utilizada para producir mejores matices y combinación de colores en las pantallas de alta definición, y en otras aplicaciones que se relacionen en la obtención de nuevos colores con la combinación de estos.

El desafío y lo que compete a este tutorial fue armar un controlador digital capaz de crear combinaciones y efectos de iluminación, que fuesen reflejados sobre un mural (el mural ITSTA) creando una visualización agradable y por qué no artística.

Dentro de este proyecto se utilizaron tres microcontroladores de la marca avr, los cuales son:

- 1.- ATtiny2313
- 2.- ATmega48.

También:

- 1 pantalla LCD 16x2 caracteres.
- 1 botón pulsador.

ILUMINACION RGBW PARA MURAL ITSTA

Este controlador quedó determinado de la siguiente forma: obtener 12 señales PWM distintas capaces de controlar 12 leds ultrabrillantes (los leds de potencia), utilizando un botón selector como única entrada al sistema que se relacionara con una pantalla LCD de 16x2, para así poder visualizar en pantalla el nombre de los efectos que produciría el controlador, así también una vez seleccionada la función de efecto de salida esta se mantuviera guardada hasta que el usuario la cambiara sin afectar que el controlador fuese desconectado (memoria eeprom).

La razón por la que se utilizaron tres microcontroladores es la siguiente:

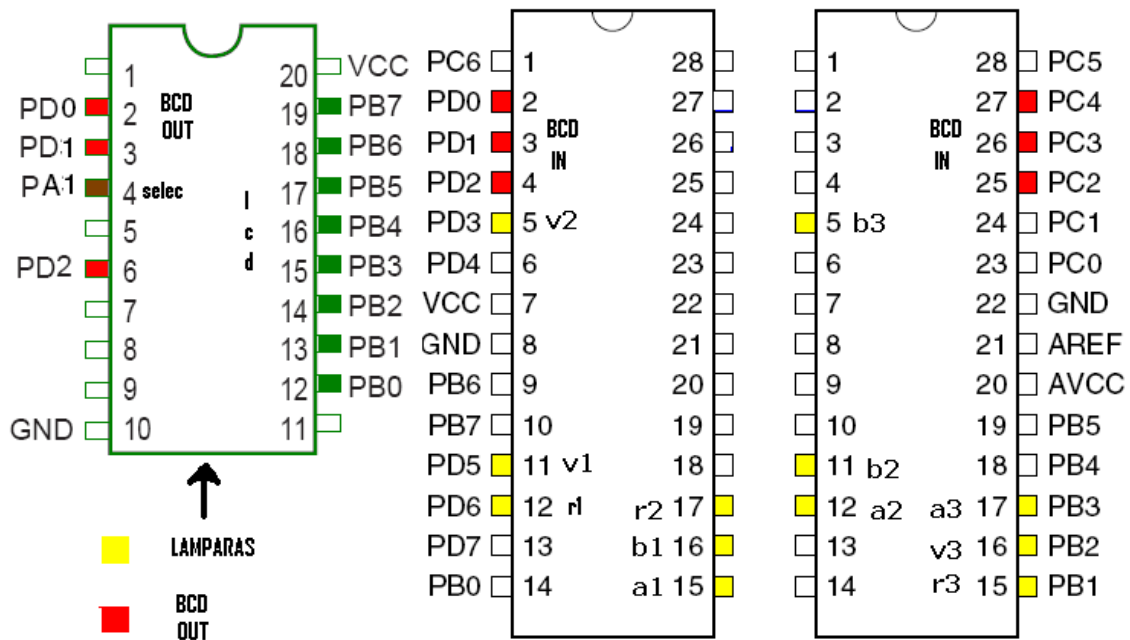


Diagrama de conexiones entre los microcontroladores, así como entradas y salidas.

La imagen anterior nos ayuda a comprender y relacionar el sistema en que se trabajó. Bien, se utilizaron tres microcontroladores ya que uno solo no podía brindar todos los requerimientos del sistema, es decir sus salidas no eran suficientes para abastecer 12 salidas PWM (para las lámparas) y por otra parte la pantalla LCD. Como se puede observar el microcontrolador más pequeño activa tanto a la LCD como a tres salidas más; esas salidas están conectadas a los otros dos micros en paralelo.

También podemos ver que las entradas de los dos ATMEGA48 están distribuidas de distinta forma. Esto es por cuestiones de diseño, en la explicación del código del programa se entenderá el reacondo.

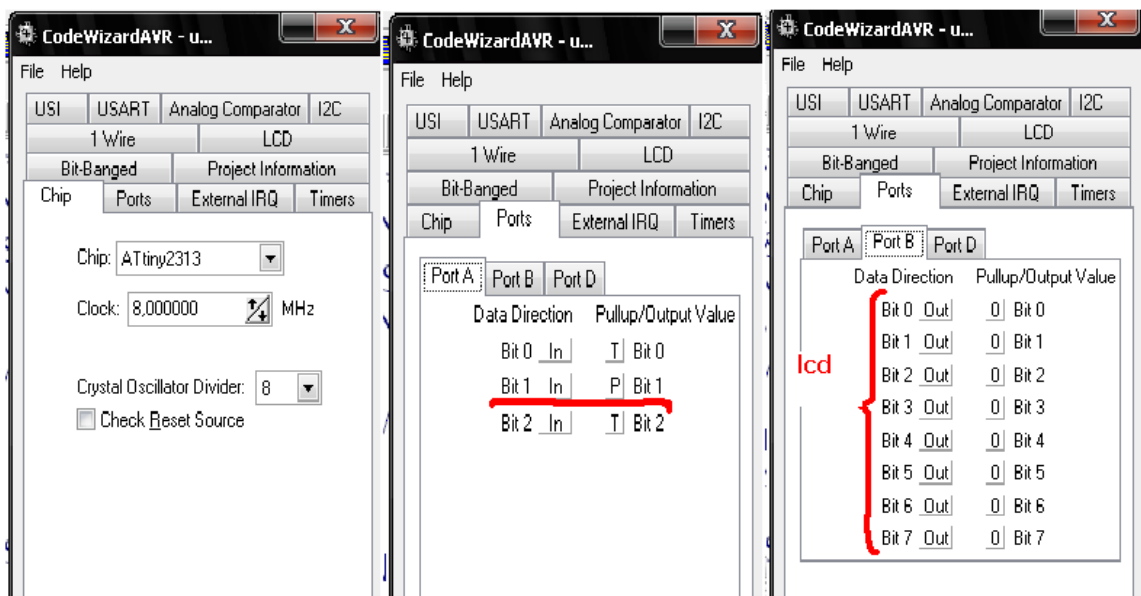
ILUMINACION RGBW PARA MURAL ITSTA

Empecemos con la programación del primer microcontrolador el ATtiny2313.

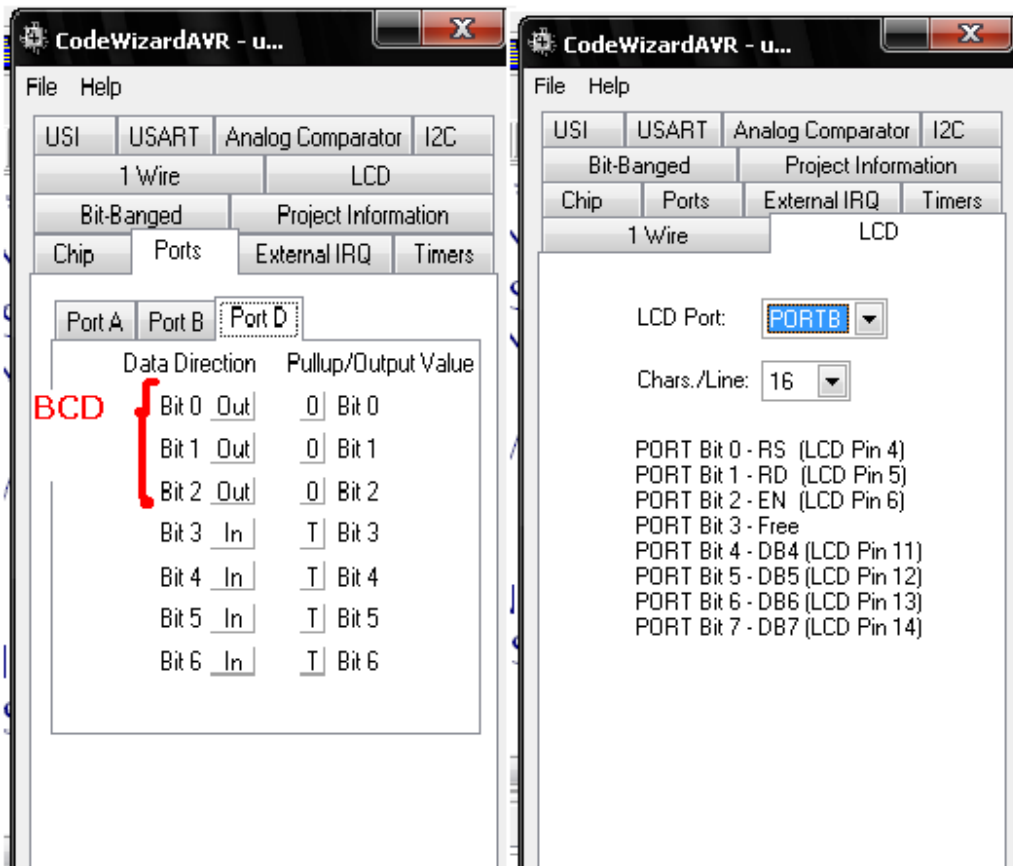
- Necesitamos un pin de entrada para el botón selector. No olvidar activar la resistencia de pull-up.
- Un puerto completo de 8 pin para la LCD, configurado como salida.
- Tres pines mas para las salidas que van a las entradas de los otros dos micros. En este caso serán los pines PD0, PD1 y PD2.

Utilizamos el wizard del codeVisionAVR.

File-new-project-ok. Y clickeamos lo acontinuacion mostrado en las ventanas.



ILUMINACION RGBW PARA MURAL ITSTA



Y por ultimo. File-generate-save and exit.

El wizard nos generará muchas líneas, por lo que el programa se verá muy extenso. Una técnica que aplico es hacer valida una regla de los registros o variables; que es “todo registro o variable se inicializa con un valor de cero”, y como al generar el programa, muchos registros propios de la librería del componente son plasmados, dentro del código, y además valen “0”, lo que hago es quitar esas líneas y también la de los comentarios y así el código queda más reducido.

Una vez que quite lo registros que valían cero y los comentarios que ocupan espacio; agregué nuevos comentarios; estos permiten saber una idea más clara de acuerdo con la programación en C, de cada línea de código.

El programa quedo de la siguiente manera:

- Nota: el código siguiente es solo para el primer microcontrolador, una vez entendido el código, podrían quitar los comentarios y el programa se verá más reducido.

ILUMINACION RGBW PARA MURAL ITSTA

/******

ESTE EL CHIP Nº 1 DE TRES QUE ESTÁN ANIDADOS___ ESTE ES EL PRINCIPAL

YA QUE LOS OTROS DOS DEPENDERAN DE ESTE, ESTE COMUNICARÁ A LOS 2 SIGUIENTES

A TRAVES DE UN BCD. Y GUARDARÁ EN EEPROM LA SELECCION REALIZADA POR EL USUARIO

Project : ILUMINACION RGBW PARA MURAL ITSTA

Version :

Date : 22/01/2011

Author : DANIEL FROYLAN MONTERO GOMEZ

Company : INSTITUTO TECNOLOGICO SUPERIOR DE TANTOYUCA

Comments:

Chip type : ATtiny2313

Clock frequency : 1,000000 MHz

Memory model : Tiny

External RAM size : 0

Data Stack size : 32

*****/

// Alphanumeric LCD Module functions

#asm

.equ __lcd_port=0x18 ;PORTB

#endasm

#include <lcd.h> // esta librería es para poder usar la LCD; la genera el wizard

#include <tiny2313.h> // esta librería es generada por el wizard

#include <delay.h> // esta librería es para hacer los retardos de tiempo,

ILUMINACION RGBW PARA MURAL ITSTA

```
void checa_boton (void); // se declara esta función que posteriormente
                        // Servirá para rescatar el cambio de selección
                        // al presionar un simple push-button.

bit botonp; // variable tipo bit solo almacenan dos valores "0" y "1".
bit botona; // variable tipo bit solo almacenan dos valores "0" y "1".
            // NOTA: estas variables cumplen una función específica
            // Explicada en otro tutorial.

unsigned char var,selec; // estas variables son del tipo carácter sin signo
                        // Pueden almacenar hasta 256 valores.
                        // Estas son variables FLASH

eprom unsigned char datoaguardar; // esta variable nos sirve
                                    // Para guardar información no volátil,
                                    // Es decir que sigue manteniendo su valor
                                    // Aun después de ser apagado el sistema.

#define boton PINA.1 // se define al pin 1 del puerto A con el apodo
                    // boton

void main(void)
{
#pragma optimize-
CLKPR=0x80;
#ifdef _OPTIMIZE_SIZE_
#pragma optimize+
#endif
PORTA=0x01; //registros del microcontrolador que fueron afectados
```

ILUMINACION RGBW PARA MURAL ITSTA

```
DDRB=0xFF; // al generar el programa
DDRD=0x07; //con el wizard
ACSR=0x80;
lcd_init(16);
var=datoaguardar; //se recupera la eeprom. Quiero decir el estado antes
//de ser apagado.

lcd_gotoxy(0,0); //En la primera fila comienza el mensaje
lcd_putsf("TTA ELECTRÓNICA");
lcd_gotoxy(0,1); //En la segunda fila comienza el mensaje
lcd_putsf("__BIENVENIDO!!_");
delay_ms(1000); // espera
delay_ms(1000); //un rato
delay_ms(1000); //para hacer visible
delay_ms(1000); // la presentación
// Nota: este mensaje aparecerá cada vez que se encienda
// el sistema.

lcd_clear(); // limpia la pantalla de la lcd
while (1) //entra al ciclo infinito
{
checa_boton(); // va directo a verificar si el boton selector
// ha sido presionado
PORTD=var; // se le asigna el valor de la variable var al puerto
// de salida D. el puerto D como salida no valdrá mas
// de 5. ya que "var" es acotada en la función "checa_boton".
selec=var; // el valor de "var" también es utilizado para sedercelo
// a la variable "selec"; esta variable es utilizada para ser
```

ILUMINACION RGBW PARA MURAL ITSTA

// evaluada en el menú.

lcd_gotoxy(0,0); //En la primer fila comienza el mensaje

lcd_putsf("___DMX PS125___"); // es el nombre del dispositivo que contiene

// la etapa de acoplamiento y potencia que hace

// disparar las salidas hacia los leds rgbw.

//NOTA: este mensaje se mantendrá fijo

// despues de que pase el tiempo de la presentacion.

switch (selec) // comienza el menú. se utiliza la sentencia switch-case
// como herramienta (ventajas de c), para evaluar a la variable
// selec; esta variable también depende de la variable "var", ya
//mencionado

{

case 0: //se evalúa a "selec" con "0" si llega a coincidir
// que selec efectivamente valga "0", lo que este
// dentro del cuerpo se ejecutará.

lcd_gotoxy(0,1); //En la segunda columna comienza el mensaje

lcd_putsf("fun__aPaGaDo___");

break;

case 1: // se evalúa a selec con "1".

lcd_gotoxy(0,1); //En la segunda columna comienza el mensaje

lcd_putsf("fun_Enc._sUaVe");

break;

case 2: // se evalúa a selec con "2".

lcd_gotoxy(0,1); //En la segunda columna comienza el mensaje

lcd_putsf("fun__BaNdErA___");

ILUMINACION RGBW PARA MURAL ITSTA

```
    break;
    case 3: // se evalúa a selec con "3".
        lcd_gotoxy(0,1); //En la segunda columna comienza el mensaje
        lcd_putsf("fun__NaViDaD__");
        break;
    case 4: // se evalúa a selec con "4".
        lcd_gotoxy(0,1); //En la segunda columna comienza el mensaje
        lcd_putsf("fun_cOrrImIEnTo");
        break;

    case 5: // se evalúa a selec con "5".
        lcd_gotoxy(0,1); //En la segunda columna comienza el mensaje
        lcd_putsf("fun_luz_blanca__");
        break;
} // llave de fin de la sentencia switch-selec.
}; //fin del while
} //fin de la función main o función principal

void checa_boton (void) // cuando se llama a la función checa_boton, la
                        // el compilador brinca hasta esta linea de código.
{
    if (boton==0)
        botona=0;
    else
        botona=1;
    if ((botonp==1)&&(botona==0)) //hubo cambio de flanco de 1 a 0
    {
```

ILUMINACION RGBW PARA MURAL ITSTA

```
var++; //Se incrementa la variable
if (var==6)
var=0;
datoaguardar=var; //se grabara la eeprom con el valor de var
delay_ms(40); //Se coloca retardo de 40mS para eliminar rebotes
}
if ((botonp==0)&&(botona==1)) //hubo cambio de flanco de 0 a 1
delay_ms(40); //Se coloca retardo de 40mS para eliminar rebotes
botonp=botona;
} // fin de la funcion checa_boton, regresará a la línea de código
// siguiente en donde fue llamada la función.
```

Hasta este paso solo se podrá visualizar en la pantalla mensajes, cuando se encienda el dispositivo y cada vez que se presione el botón selector.

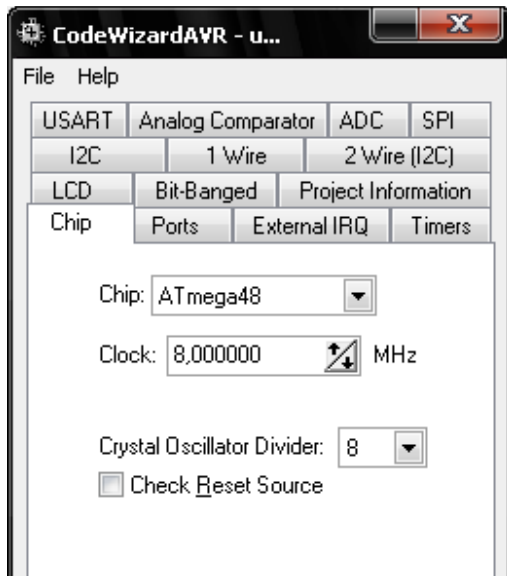
Es hora de configurar a los otros dos microcontroladores, que dependen de las salidas P0, P1 y P2 del primer microcontrolador. A lo que he llamado BCD.

Estos 2 microcontroladores contienen la relación entrada-salida muy semejante, de hecho dentro del sistema tienen el mismo objetivo; reciben la misma señal, los pines de salida son los mismos aunque no se activan al mismo tiempo.

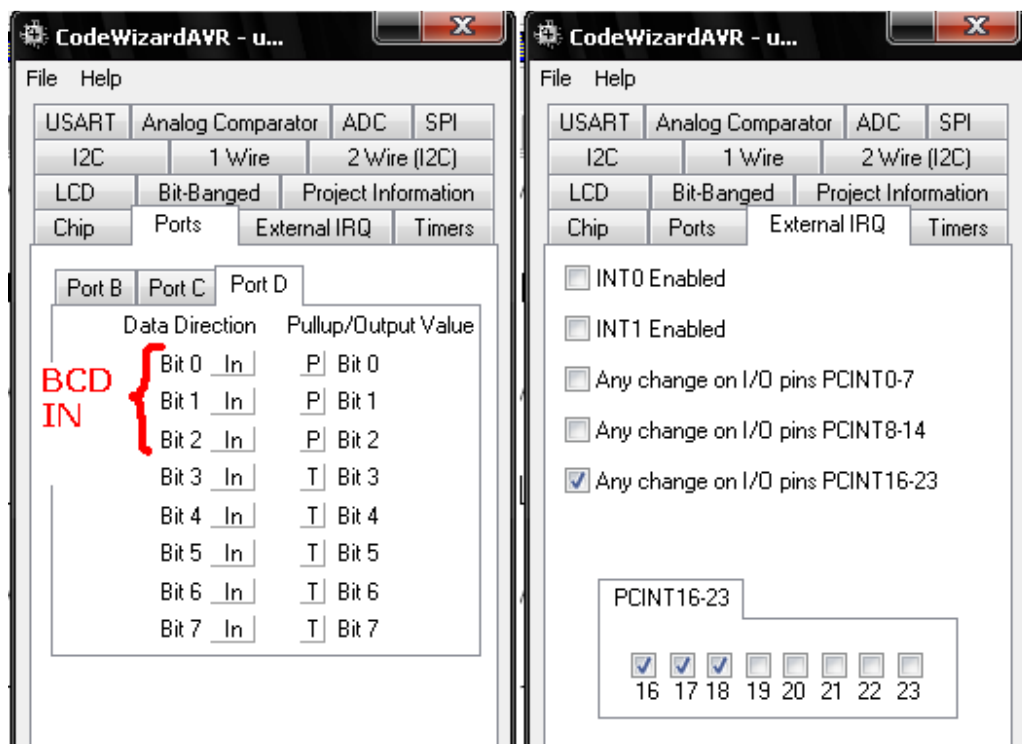
Las salidas que son PWM, son correspondientes a los pines de salida relacionados a los “timer0, timer1 y timer2” que son correspondientes a los registros OCR0A, OCR0B, OCR1AL, OCR1BL, OCR2A y OCR2B. En la primera imagen se aprecia la distribución de los registros con relación a los pines de salida. Utilizamos los mismos pasos para llegar al asistente (el wizard).

PARA AMBOS MICROS

ILUMINACION RGBW PARA MURAL ITSTA

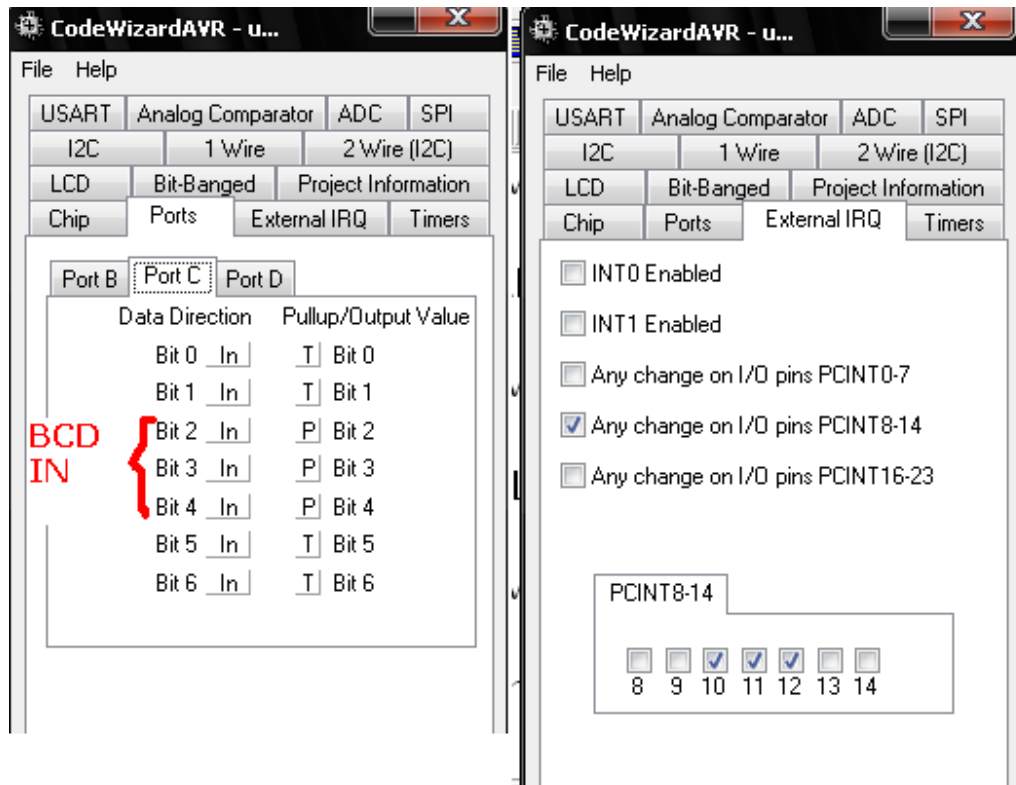


- Tendremos que configurar los pines de entrada:
Para este caso dentro del asistente, se configuraran interrupciones globales referentes a los pines de entrada.
Para el caso del primer microcontrolador serán los pines P0, P1 y P2 y en cuanto a las interrupciones globales activaremos las PCINT16, PCINT17 y PCINT18. Hacemos los siguientes cliKQueos.



ILUMINACION RGBW PARA MURAL ITSTA

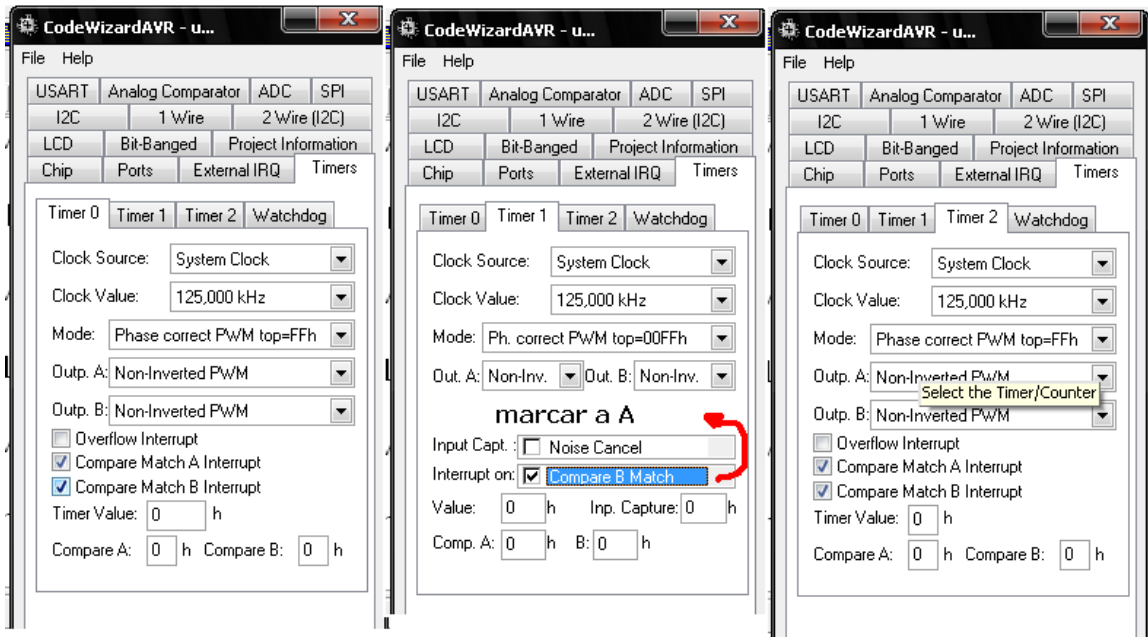
Para el caso del SEGUNDO microcontrolador serán los pines P0, P1 y P2 y en cuanto a las interrupciones globales activaremos las PCINT16, PCINT17 y PCINT18. Hacemos los siguientes cliKQueos.



- Los timers. La configuración de los timers (nos servirá para mostrar la señal en los pines de salida), es la misma para los dos micros.

Seguir estos pasos.

ILUMINACION RGBW PARA MURAL ITSTA



File-generate-save and exit.

Aparecerán unos cuadros que dicen “desea configurar al pin tal como salida”

Les daremos que si a todos. En total serán 6 mensajes.

Se generará el código siguiente; bueno este ya esta reducido, programado y explicado.

/******

ESTE EL CHIP Nº 2 DE TRES QUE ESTAN ANIDADOS___ LAS SALIDAS CON PWM SON DEPENDIENTES DE LAS ENTRADAS; SON DECODIFICADOS A UN BCD. ATRAVEZ DE UNA INTERRUPCION GLOBAL EXTERNA.

Project : ILUMINACION RGBW PARA MURAL ITSTA

Date : 22/01/2011

Author : DANIEL FROYLAN MONTERO GOMEZ

Company : INSTITUTO TECNOLÓGICO SUPERIOR DE TANTOYUCA

Comments:

ILUMINACION RGBW PARA MURAL ITSTA

Chip type : ATmega48
Clock frequency : 1,000000 MHz
Memory model : Small
External RAM size : 0
Data Stack size : 128

*****/

```
#include <mega48.h>
```

```
#include <delay.h>
```

```
bit ret1,uno,dos;/-- bandera de retorno para la función 1 es del tipo bit.
```

```
unsigned char selec;
```

```
//1 2 3 4 5 6 7 8 9 A B C ---- LAMPARAS RGB
```

```
unsigned char r1, v1, a1, b1, r2, v2, i,i2,i3,a2, b2, r3, v3, a3, b3;/-- SIMULAN EL  
//COLOR DE LAS LAMPARAS Y SIRVEN DE VARIABLES CONTROLADORAS  
//DE SALIDAS RGBW
```

```
unsigned char lamparas[12]; // se declara un arreglo de 12 casillas del tipo
```

```
//unsigned char, se utilizará para posicionar
```

```
//en cada una de las casillas las variables
```

```
//arriba declaradas, que son las que se modificarán
```

```
//en el programa para hacer varios efectos
```

```
//cambiantes.
```

```
void rojo_f (void); //estas funciones son las correspondientes a una función
```

```
void verde_f (void); //de salida. Lo que hacen estas funciones es un corrimiento
```

```
void azul_f (void); // infinito al estar posicionada la variable selec en 4.
```

```
void amarillo_f (void); // la explicación de estos eventos corresponden a
```

```
// una imagen, explicada fuera del programa.
```

ILUMINACION RGBW PARA MURAL ITSTA

```
void funcion_incdec (void); // esta función es correspondiente al llamado
                               // en la posición selec cuando vale "1".

void funcionban(void);      //correspondiente a la función que muestra el
//encendido

                               // de la bandera de MEXICO.

void funcion_navidad(void); //muestra una representación de la generación de
//distintos

                               //colores

bit a,b,c;      // estas variables son para capturar que número
                //binario entra a los micros 2 y 3.

// Pin change 16-23 interrupt service routine
interrupt [PCINT2] void pin_change_isr2(void) // esta es la generación de código
{
    //proporcionada por el wizard al seleccionar la
    // interrupcion
    // global externa. Cada vez que suceda cualquier
    //cambio en los pines configurados la línea de código
    //saltará hasta aquí.

a=PIND.0;    //cada vez que exista un cambio en cualquiera
b=PIND.1;    // de los tres pines
c=PIND.2;    // se asignará el valor que tenga el pin a las variables "a, b y c"
    delay_ms(40); // espera un momento para evitar rebotes.
}

interrupt [TIM0_COMPA] void timer0_compa_isr(void)
{
```

ILUMINACION RGBW PARA MURAL ITSTA

```
OCR0A=0+lamparas[0] ;
}
interrupt [TIM0_COMPB] void timer0_compb_isr(void)
{
OCR0B=0+lamparas[1] ;
}
interrupt [TIM1_COMPA] void timer1_compa_isr(void)
{
    OCR1AL=0+lamparas[2];
}
interrupt [TIM1_COMPB] void timer1_compb_isr(void)
{
    OCR1BL=0+lamparas[3];
}
interrupt [TIM2_COMPA] void timer2_compa_isr(void)
{
OCR2A=0+lamparas[4];
}
interrupt [TIM2_COMPB] void timer2_compb_isr(void)
{
    OCR2B=0+lamparas[5];
}
```

/******

El código de arriba, tiene que ver con las interrupciones configuradas en un Principio con el wizard, esto es para alterar el valor correspondiente a cada registro de control: para el timer0, timer1 y timer2. Como podemos observar

ILUMINACION RGBW PARA MURAL ITSTA

cada vez que tenga que ser comparado, el registro del timer correspondiente con el registro de comparación este cambiara a su salida (el pin correspondiente) un voltaje PWM, que se mantendrá hasta que la variable correspondiente a las casillas del arreglo, cambie de valor en el programa, estos valores cambian según, la función que las altere.

```
*****/
```

```
void main(void)
```

```
{
```

```
// Declare your local variables here
```

```
// Crystal Oscillator division factor: 8
```

```
#pragma optimize-
```

```
CLKPR=0x80;
```

```
CLKPR=0x03;
```

```
#ifdef _OPTIMIZE_SIZE_
```

```
#pragma optimize+
```

```
#endif
```

```
DDRB=0x0E;
```

```
PORTD=0x07;
```

```
DDRD=0x68;
```

```
TCCR0A=0xA1;
```

```
TCCR0B=0x02;
```

```
TCCR1A=0xA1;
```

```
TCCR1B=0x02;
```

```
TCCR2A=0xA1;
```

```
TCCR2B=0x02;
```

ILUMINACION RGBW PARA MURAL ITSTA

```
PCICR=0x04;
```

```
PCMSK2=0x07;
```

```
PCIFR=0x04;
```

```
TIMSK0=0x06;
```

```
TIMSK1=0x06;
```

```
TIMSK2=0x06;
```

```
ACSR=0x80;
```

```
// Global enable interrupts
```

```
#asm("sei")
```

```
while (1)
```

```
{
```

```
    // estamos dentro del ciclo infinito
```

```
    // cada vez que el seguimiento del código pase
```

```
    //por aquí, se actualizará el valor de cada una de
```

```
    //las casillas del arreglo. Este valor lo tomaran las
```

```
    //interrupciones de comparación de los timers,
```

```
    // para crear la visualización de los colores de salida.
```

```
    lamparas[0]=r1;
```

```
    lamparas[1]=v1;
```

```
    lamparas[2]=a1;
```

```
    lamparas[3]=b1;
```

```
    lamparas[4]=r2;
```

```
    lamparas[5]=v2;
```

```
    lamparas[6]=a2;
```

```
    lamparas[7]=b2;
```

ILUMINACION RGBW PARA MURAL ITSTA

```
lamparas[8]=r3;
lamparas[9]=v3;
lamparas[10]=a3;
lamparas[11]=b3;
```

```
    if(a==0 && b==0 && c==0)    // en esta parte del programa
selec=0;                        // los valores rescatados de los
    if(a==1 && b==0 && c==0)    // pines de entrada. (que fueron)
selec=1;                        // rescatados gracias a la interrupción
                                // externa global) son diagnosticados
                                // con sentencias and.
    if(a==0 && b==1 && c==0)    // esto para saber que numero
selec=2;                        // binario fue enviado por el
                                // primer micro, y se lo asigna
    if(a==1 && b==1 && c==0)    // a la variable selec, ya en
selec=3;                        // decimal, para tener un uso
                                // mas viable de esta informacion,
    if(a==0 && b==0 && c==1)    // y así interpretarlo a través de
selec=4;                        // un menú utilizando switch case.

                                // una vez rescatada la información de entrada.
                                // se evalúa la información para interpretarla
                                // con las funciones de salida.

    if(a==1 && b==0 && c==1)
selec=5;
switch (selec)
{
```

ILUMINACION RGBW PARA MURAL ITSTA

```
case 0: // se evalúa a "selec" con cero, nos dice que las salidas
r1=0; // deben estar apagadas.
v1=0;
a1=0; // efectivamente
b1=0; //a todas las variables
r2=0; // referentes a las lámparas (los leds de potencia)
v2=0; //les asignamos "0".
a2=0; // por lo que al trasladarse al arreglo.
b2=0; // y después entre a la interrupción
r3=0; // de los timers
v3=0; // este mande cero a la salida
a3=0; // esto significa cero a la salida.
b3=0;
```

```
break;
```

```
case 1: // si es que selec vale "1"
r1=i; // las variables referentes a las lámparas
v1=i; // se les asignara el valor de la variable i
a1=i; // el valor de la variable cambia
b1=i; // dentro de la función referenciada
r2=i; // dentro de este cuerpo de la estructura
v2=i; // case.
a2=i;
b2=i;
```

ILUMINACION RGBW PARA MURAL ITSTA

```
r3=i;
v3=i;
a3=i;
b3=i;
funcion_incdec();
    break;
```

case 2:

```
        funcionban(); // es llamada función "funcionban"

r1=0;
v1=i3;
a1=0;
b1=0;           //las variables toman
r2=0;           // valores distintos
v2=0;           // referentes a las
a2=0;           // variables i3,i2,i2
b2=i;           // se encuentra afectadas
r3=i2;          //en la función referenciada
v3=0;
a3=0;
b3=0;
    break;
```

case 3:

```
r1=i;
v1=i2;          // en este cuerpo de la estructura
a1=i3;          // tambien se ven afectadas las
```

ILUMINACION RGBW PARA MURAL ITSTA

```
b1=0x1F;//blanco      variables
r2=i;                 //i,i2 e i3
v2=i2;                // pero esta vez la función que las
a2=i3;                // afecta es diferente.
b2=0x1F;//blanco
r3=i;
v3=i2;                // también deja al color blanco con
a3=i3;                // un valor constante en todas los reflectores
b1=0x1F;//blanco

funcion_navidad();
    break;
    case 4:
rojo_f();             // dentro de este cuerpo
verde_f();            // se puede hacer visible en la salida
azul_f();             // un corrimiento
amarillo_f();         // el como lo hace se explica fuera del programa.
    break;
    case 5:
r1=0;
v1=0;
a1=0;
b1=0xFF;             // cuando el código del programa
r2=0;                // cae aquí lo que hace es
v2=0;                // que solo activa las lámparas blancas
a2=0;                // y las mantiene fijas como cualquier reflector
b2=0xFF;             // mientras que las demás las mantiene apagadas.
```

ILUMINACION RGBW PARA MURAL ITSTA

```
r3=0;
v3=0;
a3=0;
b3=0xFF;
```

```
    break;
```

```
    }
```

```
};
```

```
}
```

```
void funcion_incdec(void){
```

```
    // aproximadamente tarda 10s en encenderse moduladamente y de regreso
```

```
    if (i<=0xFE && ret1==0){
```

```
        i++;
```

```
        delay_ms(50);
```

```
    }
```

```
    if (i==0xFF){
```

```
        ret1=1;
```

```
    }
```

```
    if (i>0 && ret1==1){
```

```
        i--;
```

```
        delay_ms(50);
```

```
            if (i==0)
```

```
                ret1=0;
```

```
        }
```

```
    }
```

ILUMINACION RGBW PARA MURAL ITSTA

```
void funcionban(void){ //se prenderan los colres de nuestra bandera en el mural
if (i2 <=0xFE ){
    i=0;
    i3=0;
    i2++;
    delay_ms(10);
}
if (i2>0xFE && uno==0 ){
    i3=0;
    i++;
    delay_ms(10);
}
if (i==0xFE){
    dos=1;
}
if(i2>0xFE && dos==1) {
    uno=1;
    i3++;
    delay_ms(10);
}
if (i3>0xFE)
    i3=0xFE;
}

void funcion_navidad (void) {
    i++;
    delay_ms(2);
}
```

ILUMINACION RGBW PARA MURAL ITSTA

```
i3++;  
delay_ms(1);  
i2++;  
delay_ms(3);  
  
}  
  
void rojo_(void) {  
  
    lamparas[7]=0x00;  
    lamparas[0]=0xFF;  
    delay_ms(1000);  
  
    lamparas[0]=0x00;  
    lamparas[5]=0xFF;  
    delay_ms(1000);  
  
    lamparas[5]=0x00;  
    lamparas[10]=0xFF;  
    delay_ms(1000);  
  
    lamparas[10]=0x00;  
    lamparas[7]=0xFF;  
    delay_ms(1000);  
  
}
```

ILUMINACION RGBW PARA MURAL ITSTA

```
void verde_f(void){
```

```
    lamparas[7]=0x00;
```

```
    lamparas[1]=0xFF;
```

```
    delay_ms(1000);
```

```
    lamparas[1]=0x00;
```

```
    lamparas[6]=0xFF;
```

```
    delay_ms(1000);
```

```
    lamparas[6]=0x00;
```

```
    lamparas[11]=0xFF;
```

```
    delay_ms(1000);
```

```
    lamparas[11]=0x00;
```

```
    lamparas[4]=0xFF;
```

```
    delay_ms(1000);
```

```
}
```

```
void azul_f(void){
```

```
    lamparas[4]=0x00;
```

```
    lamparas[2]=0xFF;
```

```
    delay_ms(1000);
```

ILUMINACION RGBW PARA MURAL ITSTA

```
lamparas[2]=0x00;  
lamparas[5]=0xFF;  
delay_ms(1000);
```

```
lamparas[5]=0x00;  
lamparas[8]=0xFF;  
delay_ms(1000);
```

```
lamparas[8]=0x00;  
lamparas[7]=0xFF;  
delay_ms(1000);  
}
```

```
void amarillo_f(void){
```

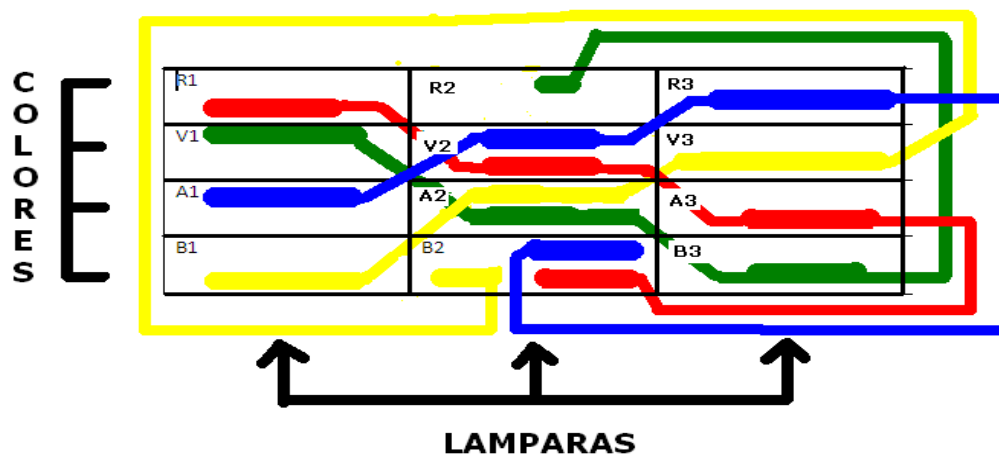
```
lamparas[7]=0x00;  
lamparas[3]=0xFF;  
delay_ms(1000);
```

```
lamparas[3]=0x00;  
lamparas[6]=0xFF;  
delay_ms(1000);
```

```
lamparas[6]=0x00 ;  
lamparas[9]=0xFF;  
delay_ms(1000);
```

ILUMINACION RGBW PARA MURAL ITSTA

```
lamparas[9]=0x00;  
lamparas[7]=0xFF;  
delay_ms(1000);  
}
```



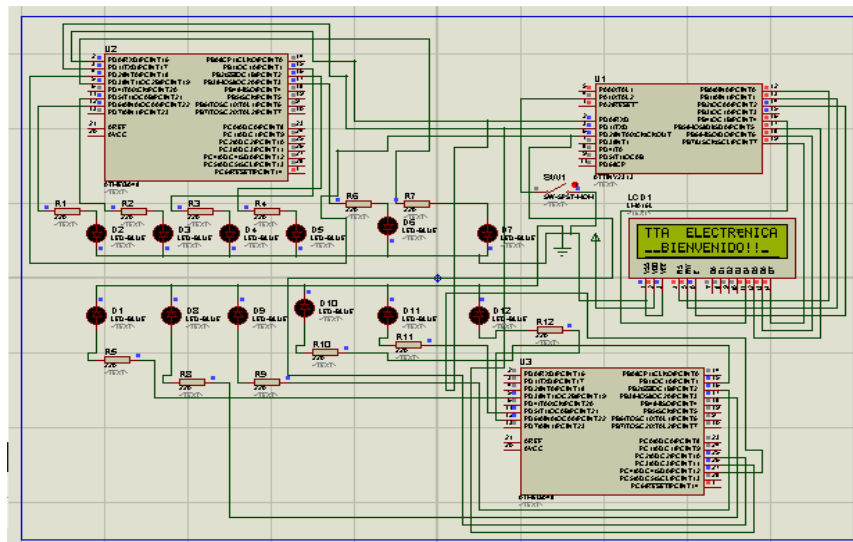
SEGUIMIENTO DE EL ENCENDIDO DE LAS LAMPARAS, RESPECTO A LAS FUNCIONES DEL MISMO NOMBRE DEL COLOR, ENCONTRADAS EN LOS MICROS 2 Y 3.

La imagen de arriba nos ayuda a entender lo que pasa al escoger la función de corrimiento. Como vimos en el programa, cuando selee vale 4, manda a llamar las funciones rojo, verde, azul y amarillo. Una a una; quiero decir como la primer función es rojo esta encenderá a su máxima intensidad (un 255 o 0xFF) la lámpara r1, esperará un segundo y hará el recorrido (encendiendo la siguiente y apagando la anterior), así estará cíclicamente hasta que la variable selee sea modificada.

Nota: esta función está estratégicamente diseñada para que recorra reflector a reflector encendiendo un led de distinto color siempre.

Para ahorrar un poco de tiempo en estar programando los micros y haciendo pruebas físicamente, se utilizó la herramienta para simulación de circuitos proteus de isis, el circuito del controlador quedó armado de la siguiente manera:

ILUMINACION RGBW PARA MURAL ITSTA



Programa simulado en proteus.

En este momento aun no hemos programado el tercer microcontrolador. Para ello usaremos casi el mismo código del segundo microcontrolador, salvo a unas pequeñas modificaciones. Revisaremos que es lo que tenemos.

- La relación entrada y salida del circuito.
- El esquema de conexiones
- La configuración del micro a través del wizard.
- Sabemos reducir el código para que sea menos extenso.
- Tenemos el código completo del segundo microcontrolador con explicación.

Una vez reducido el código, nos damos cuenta de que el código en la parte de la interrupción global externa cambia ligeramente, esto es por lógica porque ocupamos otro bloque de interrupciones; aun así pegaremos el código referente al segundo microcontrolador. Y cambiaremos el nombre de los pines, en este caso pues porque los pines no son el PD0, PD1 Y el PD2 sino el PC2, PC3 y PC4.

Por otra parte lo correspondiente a las salidas, esto es en donde están las interrupciones de los timer, no ocuparemos las casillas del 0 al 5, sino del 6 al 11.

Y también modificaremos esa parte.

Lo interesante de esta técnica es que colocamos aparentemente el mismo programa en los dos microcontroladores, esto nos permitió tener una mejor estructura del diseño de software, también como los ambos microcontroladores fueron configurados a la misma frecuencia, nos reduce el error a atrasos en el

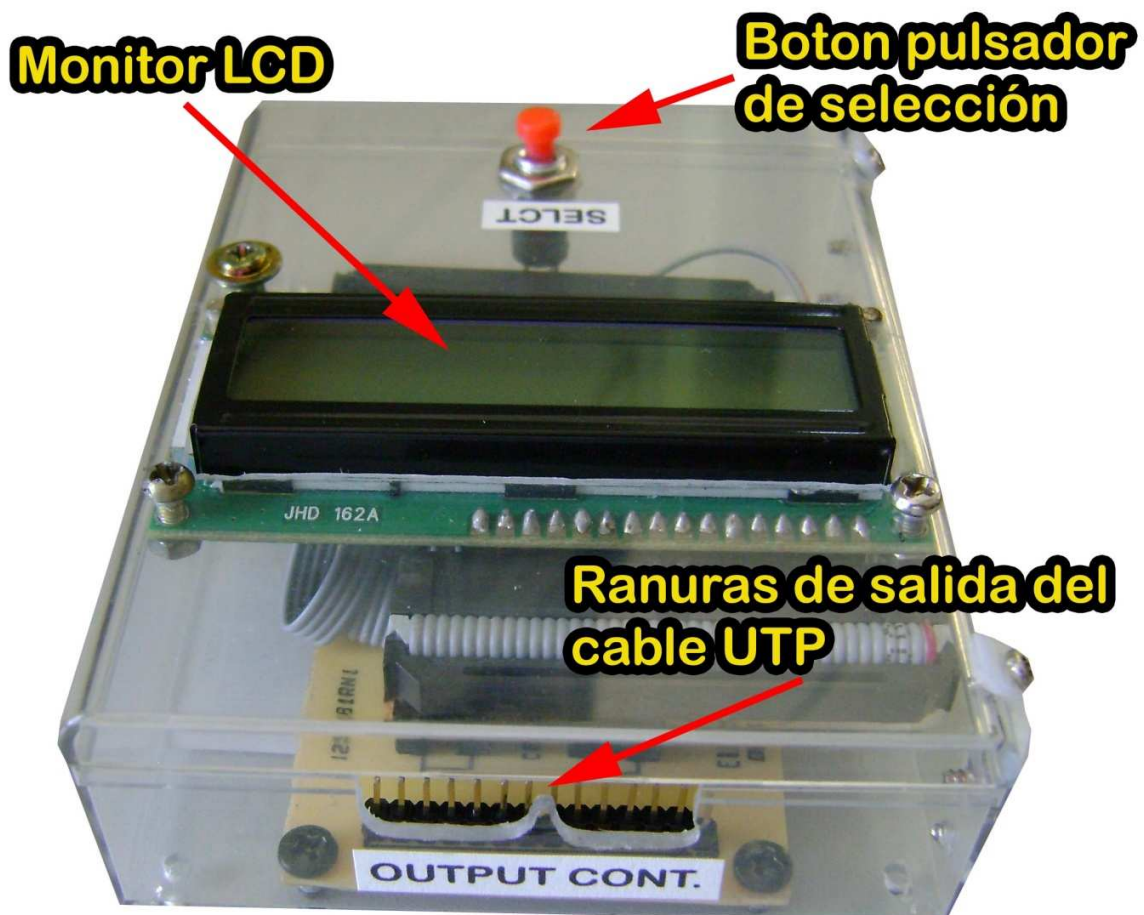
ILUMINACION RGBW PARA MURAL ITSTA

proceso, y como tanto las variables como las funciones fueron declaradas y también nombradas al compilar cada programa no nos genera errores ni de lógica ni de sintaxis, y solo tuvimos que cambiar la interpretación a la salida.

Conclusión al utilizar esta técnica de programación

- facilita la interpretación del código de todo el sistema.
- Nos permitiría trabajar con n número de microcontroladores sincronizados.

Imágenes del controlador descrito y su etapa ampliadora.

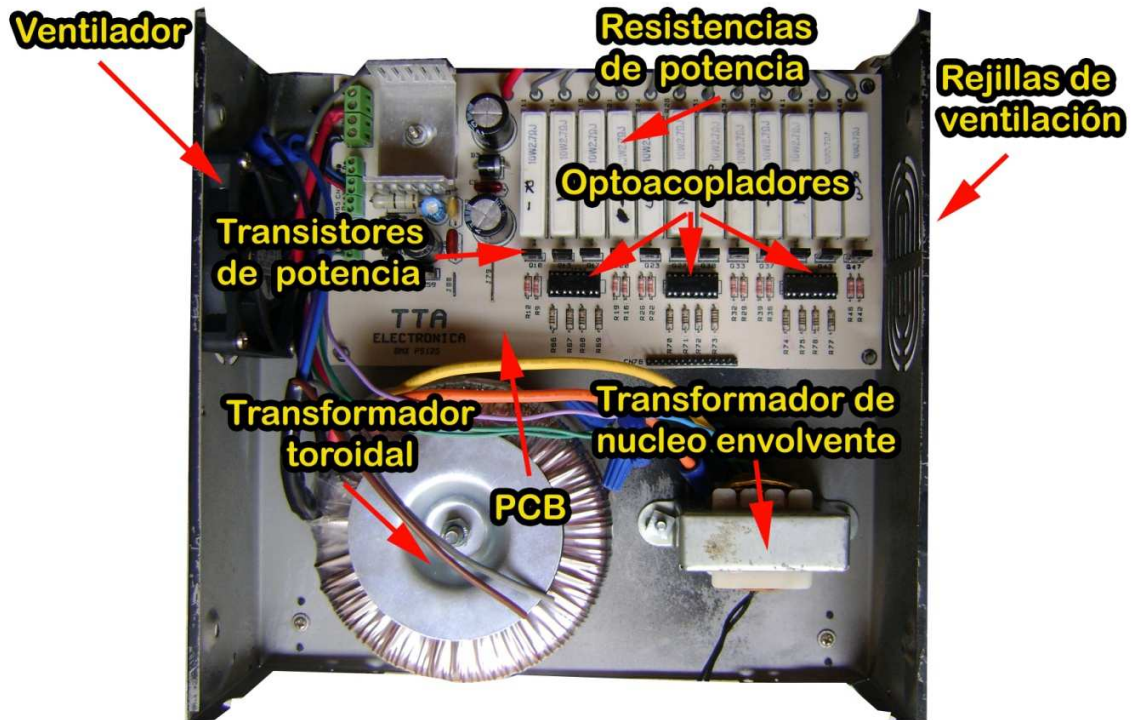


ILUMINACION RGBW PARA MURAL ITSTA



ILUMINACION RGBW PARA MURAL ITSTA

Circuito de amplificación o de potencia.



Imágenes del mural iluminado.





Referencias:

Infante S. David. *Notas del curso programación en C de los microcontroladores ATMEL*. Instituto Tecnológico de Morelia. Versión 8.10.