

5. Apéndice

5.1 manual de usuario



**MANUAL PARA LA INTERFAZ DE USUARIO PARA EL SISTEMA DE CONTROL
DE INTENSIDAD DE UN LASER UTILIZADO EN LA FABRICACIÓN DE
BIOPRÓTESIS CARDIACAS**



**CENTRO DE CIENCIAS APLICADAS Y
DESARROLLO TECNOLÓGICO**

Laboratorio de electrónica

OPERACIÓN DEL SISTEMA

1.- El usuario deberá conectar el sistema a la computadora por medio de un cable DB25 al puerto paralelo. **(nota: de no hacer esto el sistema estará bloqueado marcando ceros y haciendo un sonido)**

2.- Una vez que esté conectado el sistema deberá abrir la interfaz proporcionada cuyas indicaciones de uso se mostrarán posteriormente.

3.- El sistema puede trabajar de dos formas seleccionándose indistintamente:

a) Por medio de un teclado que se encuentra en el sistema y

b) Por medio de la PC utilizando el mouse y el teclado que se muestra en el software.



Figura 44. Vista posterior del módulo.

OPERACIÓN DEL SISTEMA MEDIANTE EL TECLADO QUE SE ENCUENTRA EN EL MÓDULO

El sistema inicialmente estará en espera de que el usuario introduzca el ciclo de trabajo deseado por medio del teclado (mostrará ceros hasta que se introduzca un número), una vez que el usuario haya introducido dicho ciclo de trabajo deberá oprimir la tecla "ENTER" para que el sistema comience a trabajar.

En el caso de haberse equivocado al introducir el ciclo de trabajo se tienen dos opciones:

- a) Oprimir la tecla "CLEAR" la cual hará que aparezcan ceros en el display sin modificar el ciclo de trabajo actual.
- b) Reingresar los números pertenecientes al ciclo de trabajo deseado, cabe mencionar que sólo se obtendrá la salida con el ciclo de trabajo mostrado en los displays hasta que se presione la tecla "ENTER".

El orden en el que se introduce el ciclo de trabajo es: Decenas, unidades y parte fraccionaria.

En el caso de necesitar un ciclo de trabajo entero múltiplo de 10 es decir 10%,20%, etc. Basta con introducir el número que pertenece a las decenas y oprimir la tecla "ENTER".

El sistema tiene una resolución de 0.1% el cual se puede modificar por medio de las 2 teclas "+0.1" y "-0.1".

El sistema cuenta con un zumbador el cual suena al oprimir una tecla.

En la parte inferior donde se ubican el despliegue de los números hay un botón el cual sirve como paro en caso de que el sistema tenga que ser detenido.



Figura 45. Vista frontal del módulo.

OPERACIÓN DEL SISTEMA MEDIANTE LA PC

Por medio del software Visual Basic se creó una interfaz con el sistema por medio del puerto paralelo.

El teclado que se muestra en dicha interfaz tiene las mismas características que el teclado que esta en el módulo, con la finalidad de que se pueda trabajar de la misma manera.

A diferencia del teclado físico, este teclado tiene dos teclas adicionales las cuales son:

- a) **Habilitar:** cuya función es habilitar ambos teclados. Una vez que se oprime esta tecla el usuario podrá activar la salida PWM por medio de la tecla "ENTER".
- b) **Deshabilitar:** cuya función es totalmente contraria a la tecla anterior. Una vez que se oprima dicha tecla, el teclado físico seguirá funcionando pero **deshabilita la función de la tecla "ENTER"**, esto es, que no habrá salida PWM.



Figura 46 Teclado virtual.

POSIBLES FALLAS Y POSIBLES SOLUCIONES

Falla	Solución
El sistema no enciende.	Verificar que la luz del botón de encendido este prendida, en caso contrario verificar que el fusible funcione, en el caso de que esto este bien verificar que haya energía eléctrica en el tomacorriente.
El display marca ceros y hace un sonido.	Verificar que este conectado el cable paralelo.
No hay salida PWM.	Verificar que la tecla enter haya sido oprimida, si esto persiste verificar que la tecla habilitar haya sido oprimida.
El teclado del sistema no funciona	Verificar que el conector no se haya salido de su lugar o que algún cable este desconectado.

5.2 Código fuente del PIC

```
#include <18f452.h>
#device ADC=10
#fuses HS,WDT,NOPROTECT,NOLVP,PUT,BROWNOUT,
#use delay(clock=10000000)
#use standard_io(c)
#use standard_io(e)

byte          const          LED_MAP[16]          =
{0x7e,0x0d,0xb6,0x9e,0xcc,0xda,0xfa,0x0e,0xfe,0xce,0x7e,0x7e,0x7e,0x7e,0x7e};

short pcen = false;
short pdec = false;
short puni = false;
short boton = false;
short conversion;
short habilitado = false;
int cont=1;
int tempo;
int tecla;
int tec;
int centenas, decenas, unidades;
int n = 0;
int p = 0;
int c = 0;
long valor;
long duty;
long tol;

int teclado(void){

output_d(0x10);
if (input(pin_d3)){
    tecla = 14;
}

if (input(pin_d2)){
    tecla = 0;
}

if (input(pin_d1)){
    tecla = 15;
}

if (input(pin_d0)){
    tecla = 13;
}

output_d(0x20);
if (input(pin_d3)){
    tecla = 7;
}

if (input(pin_d2)){
    tecla = 8;
}

if (input(pin_d1)){
```

```

        tecla = 9;
    }

    if (input(pin_d0)){
        tecla = 12;
    }

    output_d(0x40);
    if (input(pin_d3)){
        tecla = 4;
    }

    if (input(pin_d2)){
        tecla = 5;
    }

    if (input(pin_d1)){
        tecla = 6;
    }

    if (input(pin_d0)){
        tecla = 11;
    }

    output_d(0x80);
    if (input(pin_d3)){
        tecla = 1;
    }

    if (input(pin_d2)){
        tecla = 2;
    }

    if (input(pin_d1)){
        tecla = 3;
    }

    if (input(pin_d0)){
        tecla = 10;
    }
    return tecla;
}

int tecladovb(void){
    tecla = 0;
    if (input(pin_c4)){
        tecla = tecla + 1;
    }
    if (input(pin_c5)){
        tecla = tecla + 2;
    }
    if (input(pin_c6)){
        tecla = tecla + 4;
    }
    if (input(pin_c7)){
        tecla = tecla + 8;
    }
    return tecla;
}

```



```

void reset(void){
    set_pwm1_duty(0);
    unidades =0;
    decenas =0;
    centenas=0;
}

#INT_TIMER0
void toma_adc(void){ // Lectura del canal 0
    disable_interrupts(INT_TIMER0);
    conversion = adc_done();
    while(!conversion) {
        conversion = adc_done();
    }
    valor = read_adc();

    if (duty <= valor){
        tol = valor - duty;
    }else{
        tol = duty - valor;
    }
    set_timer0(0);
    enable_interrupts(INT_TIMER0);
}

void enter(void){
    if(input(pin_e2)){
        duty = _mul(centenas,10) + decenas;
        duty = _mul(duty,1000)/100;
        duty = duty + unidades;

        if (duty <= 1 ){
            duty = 0;
        }
        set_pwm1_duty(duty);
        pcen = false;
        pdec = false;
        puni = false;
        n = 0;
    }
}

#INT_EXT
void paro_de_emergencia(){
    ext_int_edge(0,L_TO_H);
    if(input(pin_b0)){
        reset();
        pcen = false;
        pdec = false;
        puni = false;
    }
}

#INT_TIMER1
void adquisicion(void){

    if(cont==1){

```

```

output_low(pin_a1);
output_low(pin_a2);
    output_b(LED_MAP[centenas]);
    if (p <=50 || pcent == false){
        output_high(pin_a3);
    }else{
        output_low(pin_a3);
    }
}
if(cont==2){
    output_low(pin_a1);
    output_low(pin_a3);
    output_b(LED_MAP[decenas]);
    if (p <=50 || pdec == false){
        output_high(pin_a2);
    }else{
        output_low(pin_a2);
    }
}
if(cont==3){
    output_low(pin_a2);
    output_low(pin_a3);
    output_b(LED_MAP[unidades]);
    if (p <=50 || unp == false){
        output_high(pin_a1);
    }else{
        output_low(pin_a1);
    }
    cont=0;
}
cont=cont+1;
p = p + 1;
if (p == 100){
    p = 0;
}

set_timer1(65000);
enable_interrupts(INT_TIMER1);
}

#INT_TIMER3
void P_paralelo(void){ // Lectura del canal 0
    if (input(pin_c0)){ //bit de paro
        set_pwm1_duty(0);
        unidades =0;
        decenas =0;
        centenas=0;
    }
    if (input(pin_e2)){
        habilitado = true;
    }else{
        habilitado = false;
    }
    set_timer3(65435); //40 us
}

void main() {
    set_tris_b(0xfe);

```

```

set_tris_d(0x0F);
set_tris_a(0xF0);
set_tris_e(0xF0);

setup_wdt(WDT_ON);
tempo=249;
setup_adc(ADC_CLOCK_INTERNAL);
set_adc_channel(0);
setup_adc_ports(RA0_analog);
setup_ccp1(CCP_PWM); // Configura CCP1 como PWM
setup_timer_2(T2_DIV_BY_1,tempo,1); // f
setup_timer_1(T1_INTERNAL|T1_DIV_BY_8);
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_2);//
setup_timer_3 (T3_INTERNAL | T3_DIV_BY_1);
set_timer3(65435); //para un T= 40us
set_timer1(65000); // para un T=5 ms
set_timer0(0); //para T=4 us
enable_interrupts(INT_TIMER3);
enable_interrupts(INT_TIMER1);
enable_interrupts(INT_TIMER0);
enable_interrupts(INT_EXT);
enable_interrupts(GLOBAL);
reset();
duty =0;

do {
restart_wdt();
output_d(0xF0);
if (duty !=0){
do {
}while(tol >= 50);
}

if (input(pin_d0)||input(pin_d1)||input(pin_d2)||input(pin_d3)||input(pin_c3)){
delay_ms(10);
if (boton == false){
boton = true;
if (input(pin_c3)){
tec = tecladovb();
}else{
tec = teclado();
}
output_high(pin_a5);

if(tec==10 && duty <=998 ){ //Aumentar 1%
unidades = unidades + 1;
if (unidades >=10){
unidades = unidades - 10;
decenas = decenas + 1;
if (decenas ==10){
decenas = 0;
centenas = centenas + 1;
}
}
}
enter();
output_low(pin_a5);
}
}
}

```

```

if(tec==11 && duty >=1 ){          //Reducir 1%
    if (unidades == 0){
        unidades = 9;
        if (decenas >=1){
            decenas = decenas - 1;
        }else{
            decenas = 9;
            if (centenas >=1){
                centenas = centenas - 1;
            }
        }
    }else{
        unidades = unidades -1;
    }
    enter();
    output_low(pin_a5);
}
if(tec==13){          //clear
    unidades = 0;
    decenas = 0;
    centenas = 0;
    n = 0;
    pcen = true;
    pdec = true;
    puni = true;
}
if (tec == 12){          //enter, C1 es el pin para el bit de inicio
    enter();
}

if( tec <= 9 || tec >= 14 ){          // E y F = 0

    if (n==0){
        centenas = tec;
        pcen = false;
        pdec = true;
        puni = true;
    }
    if (n==1){
        decenas = tec;
        pdec = false;
        puni = true;
    }
    if (n==2){
        unidades = tec;
        puni = false;
    }
    n = n + 1;
    if (n==3){
        n = 0;
    }
}
}else{
    c = c + 1;
    if (c==25){
        boton = false;
        c = 0;
        output_low(pin_a5);
    }
}

```

```
    }  
  }else{  
    if(boton == true){  
      delay_ms(50);  
      boton = false;  
      c = 0;  
    }  
    output_low(pin_a5);  
  }  
}while(true);  
}
```