Universidad Nacional Autónoma de México

Facultad de Ingeniería

Laboratorio de Microcomputadoras

Práctica No.10: Programación C Convertidor A/D e Interrupciones

Puertos Paralelos E/S, Puerto Serie

Profesor: Rubén Anaya García

Alumnos:

- Murrieta Villegas Alfonso
- Reza Chavarría, Sergio Gabriel
- Valdespino Mendieta Joaquín

Grupo: 4

Semestre: 2021-2

Práctica 10

Objetivo

Realización de programas usando programación en lenguaje C, utilización del puerto serie, convertidor analógico digital e introducción a aplicaciones con interrupciones.

Desarrollo

1. Escribir, comentar, indicar que hace; comprobar el funcionamiento del siguiente programa.

Código

E1_P10.c

```
#include<16F877.h>
#fuses HS,NOWDT,NOPROTECT
#use delay(clock=20000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)

#int_EXT
ext_int(){                    //Interrupción del Flanco por RB0
   output_toggle(PIN_D0);     //Cambio del estado de PIN D0 cada vez que se
                              //de la interrupción
}

void main(){
   ext_int_edge(L_TO_H);          //Detección de pin de subida
   enable_interrupts(INT_EXT);    //Activa la interrupción del flanco por RB0
   enable_interrupts(GLOBAL);     //Activación deneral de interrupciones
   output_low(PIN_D0);            //PIN D0 abajo

   while(TRUE){
   }
}
```

Código 1: Ejercicio de interrupción de flanco por RB0

Al enviar una señal de entrada en el pin RB0 se llevará a la interrupción del flanco por RB0. Las instrucción de la interrupción cambiará el estado del pin D0.

Diagramas

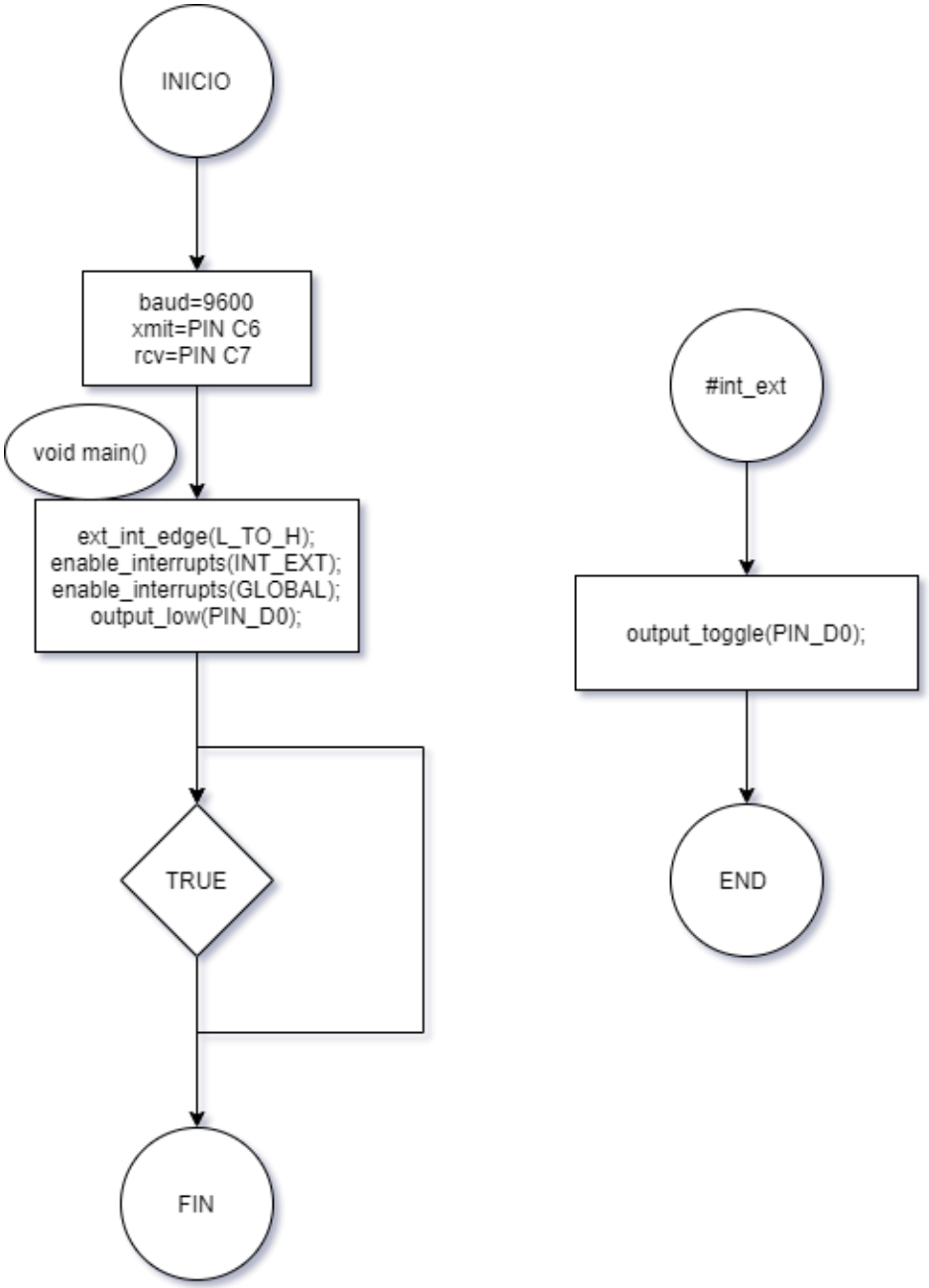
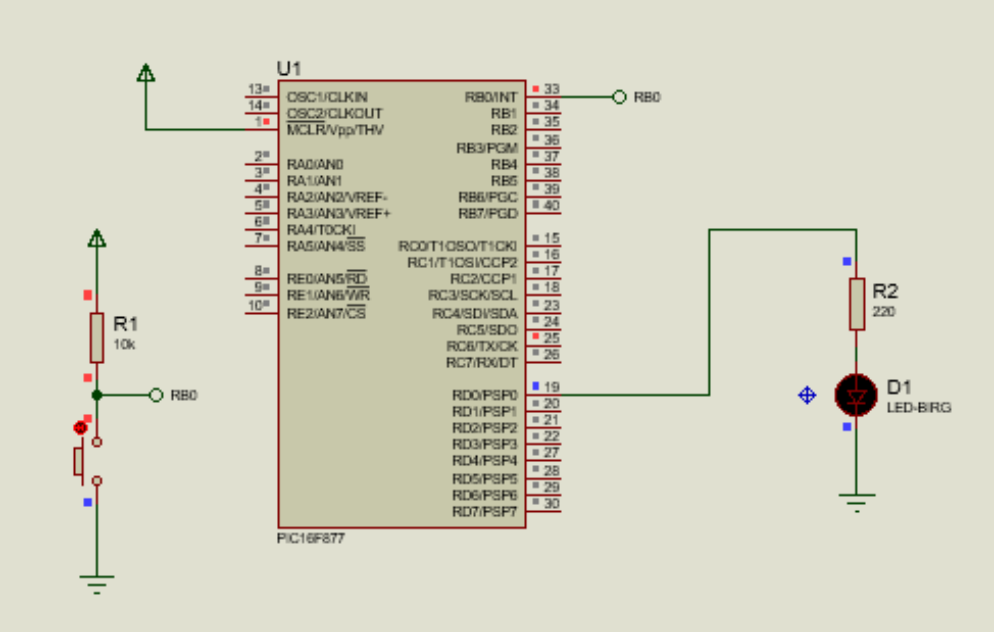
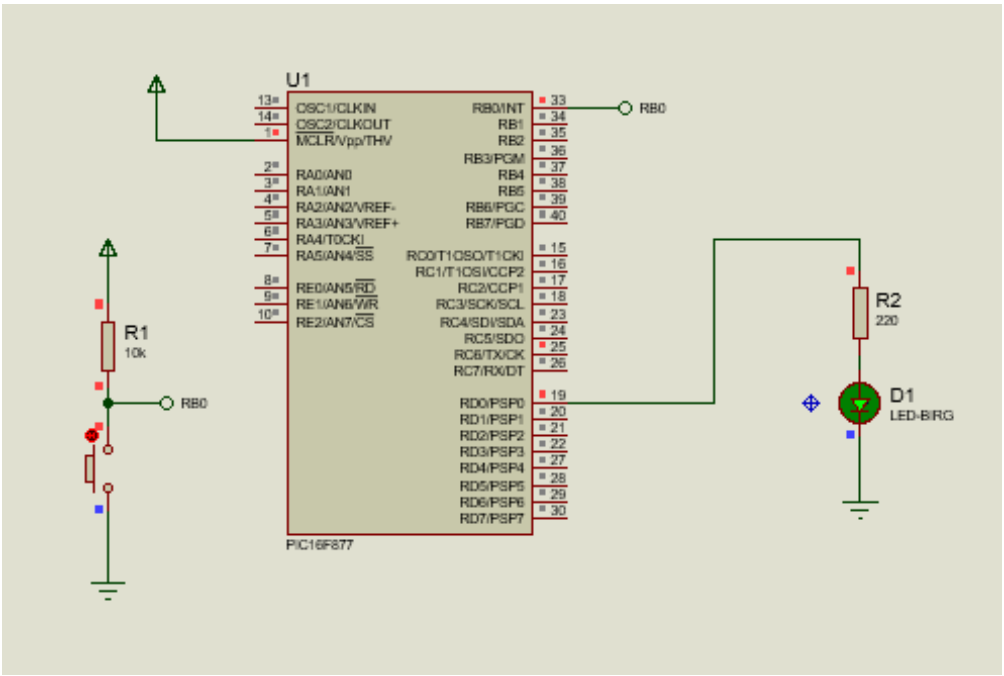


Diagrama 1: Código 1

Pruebas



Pruebas 1: Estado inicial de la simulación



Pruebas 2: Cambio del estado del pin D0

2. Escribir un programa, el cual obtenga una señal analógica a través del canal de su elección; el resultado de la conversión deberá ser desplegado en tres diferentes dispositivos, de acuerdo a la tabla 10.1.

El resultado debe ser desplegado de acuerdo a:

Periférico	Formato del despliegue	Dispositivo	Formato del despliegue
Puerto paralelo	Binario	Leds	11111111
Puerto paralelo	Voltaje	LCD	Vin= 5.00 V
Puerto serie	Decimal, hexadecimal	Terminal	Decimal=1023, Hexadecimal=0x3FF

Tabla 10.1 Formatos de resultados y periféricos

Código

E2_P10.c

```
#include<16F877.h>
#fuses HS,NOWDT,NOPROTECT
//Directivas CONVERTIDOR
#device ADC=8     //Bits de conversion
#use delay(clock=20000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7) //Configuración Serie
#define use_portb_lcd true       //Puerto B para LCD
#include <lcd.c>
#org 0x1F00, 0x1FFF void loader16F877(void) {}

long convert;              //Conversión A/D
float res;                 //Resultado de Volts

float volts(long con){
   res=(float)con*(0.01953);     //Conversión por la resolución
   return res;
}

void main(){
   lcd_init();               //Inicia le Display LCD
   setup_port_a(ALL_ANALOG);  //Define al puerto A como analógico
   setup_adc(ADC_CLOCK_INTERNAL); // Define frecuencia de muestreo del convertidor A/D
   set_adc_channel(0);          // Configura el canal a usar
   delay_us(20); // retardos
   set_tris_d(0x00);             //Salidas por puerto D
   while(TRUE){
      delay_ms(300);
      convert=read_adc();        // Obtener el resultado de la conversion
      res=volts(convert);        //Conversión a volts

      //Impresión por transmisión a puerto Serie
      printf("DECIMAL=  %04ld\n\r",convert);
      printf("HEXDECIMAL=  %04lx\n\r",convert);

      //Impresión a Display LCD
      lcd_gotoxy(1,1);
      printf(lcd_putc,"%1.2f V\n",res);

      //Salida por el puerto B
      output_d(convert);


   }
}
```

Código 2: Ejercicio de obtención de señal analógica e impresión de conversión

Diagramas

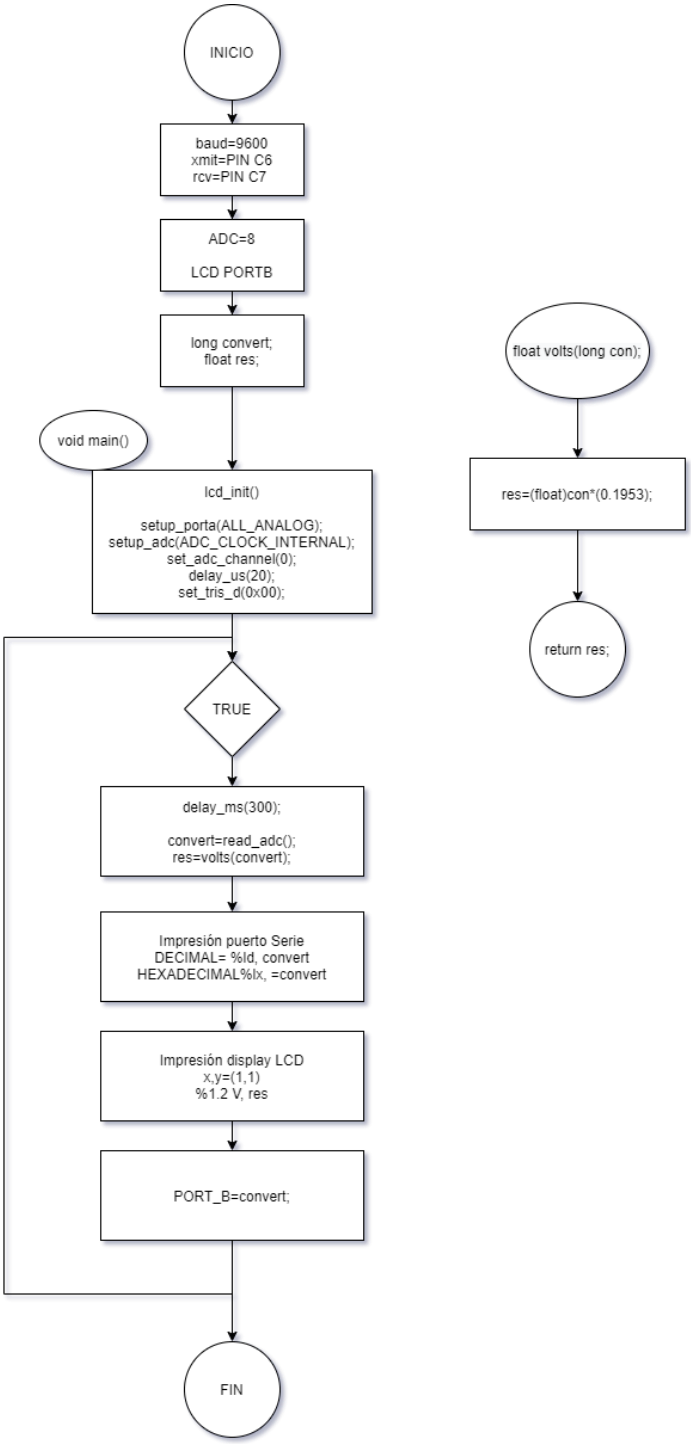
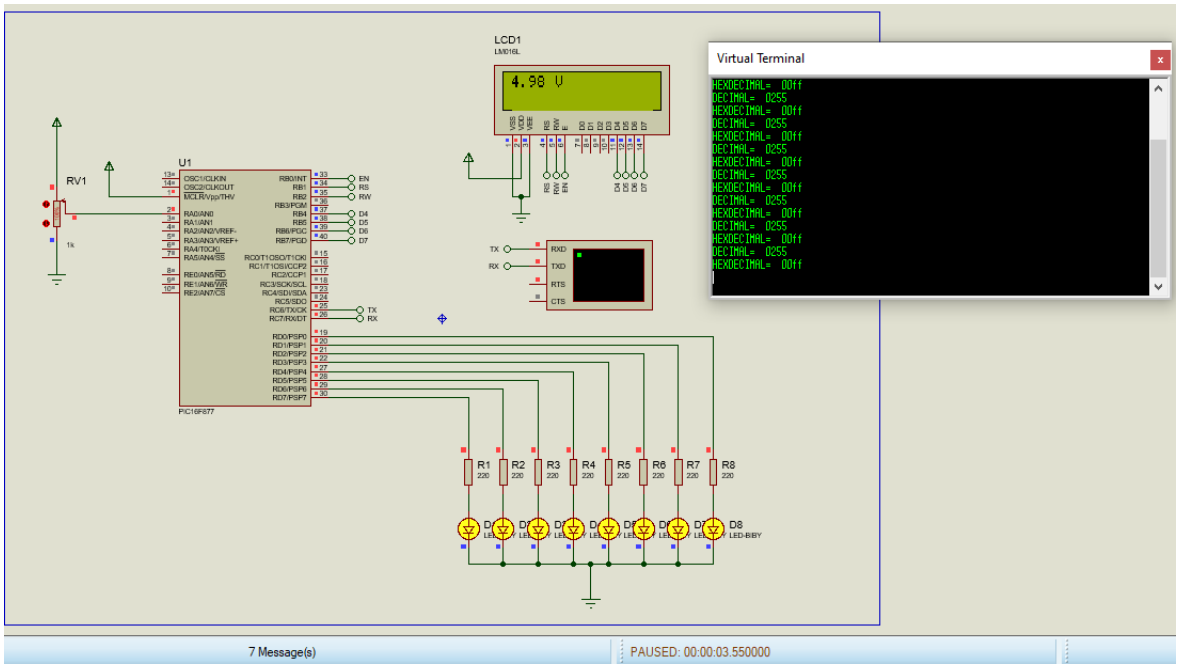
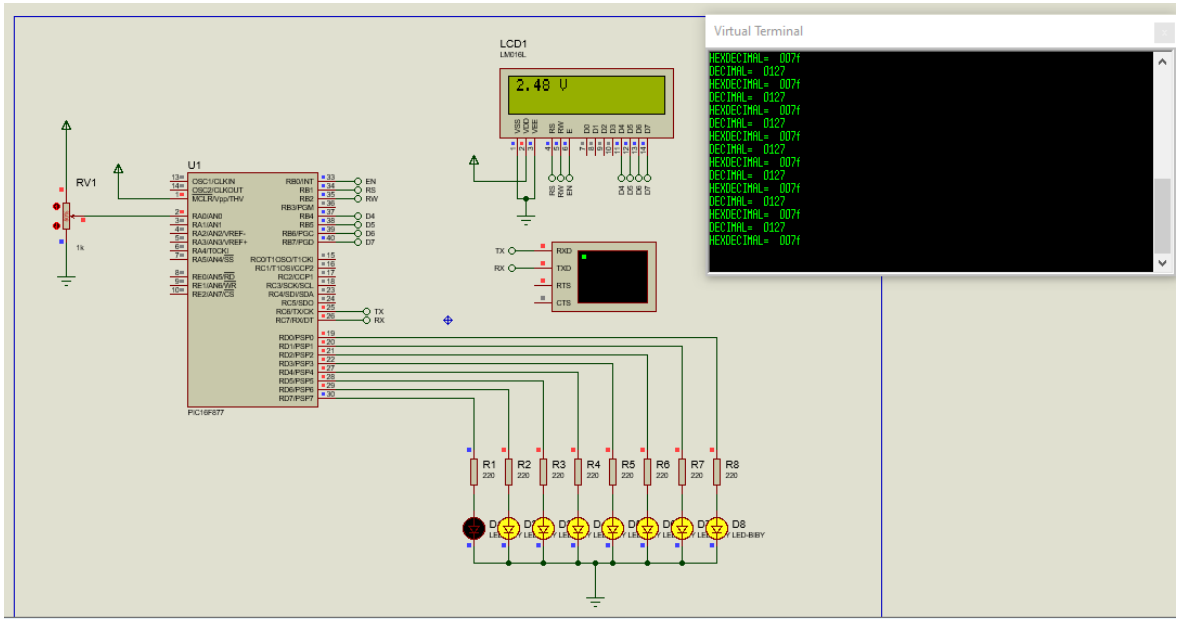


Diagrama 2: Código 2

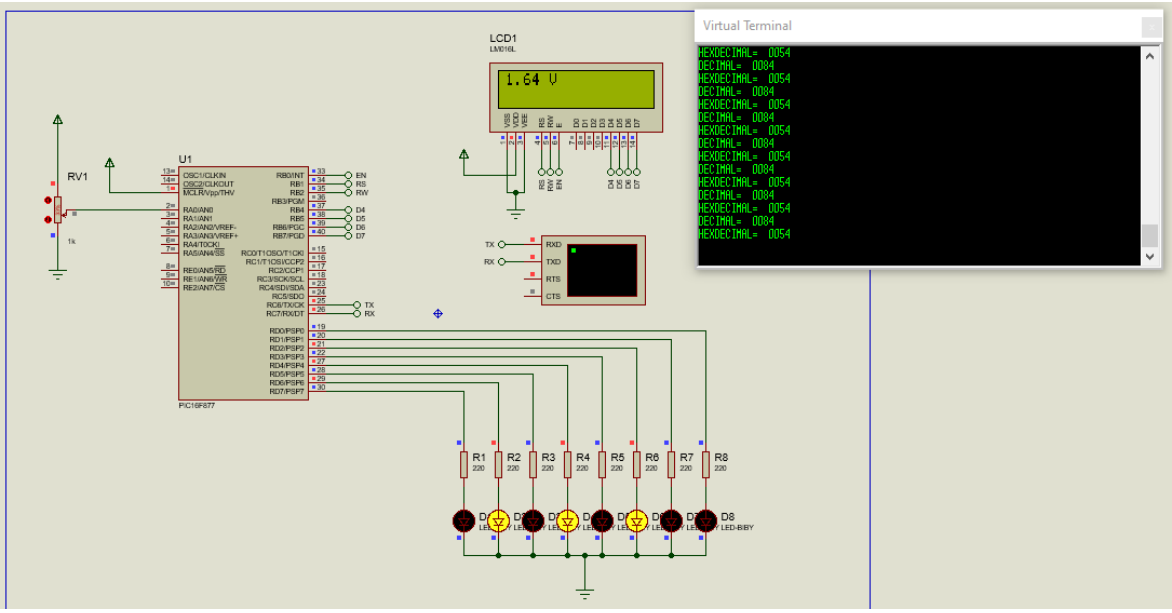
Pruebas



Pruebas 3: Obtención e impresión de señal (100%)



Pruebas 4: Obtención e impresión de señal (50%)



Pruebas 5: Obtención e información de señal (35%)

3. Utilizando la interrupción del TIMER0, realizar un programa que transmita el resultado de la conversión cada 10 segundos, usar el mismo formato del ejercicio anterior.

Código

E3_P10.c

```
#include<16F877.h>
#fuses HS,NOWDT,NOPROTECT
//Directivas CONVERTIDOR
#device ADC=8     //Bits de convertidor A/D
#use delay(clock=20000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)  //Configuración serie
#define use_portb_lcd true       //Puerto B para LCD
#include <lcd.c>
#org 0x1F00, 0x1FFF void loader16F877(void) {}

long convert;              //Conversión A/D
float res;                 //Resultado de Volts
long cont=0;

float volts(long conv);          //Función de conversión a volts
```

E3_P10.c

```
 #int_rtcc
 //Interrupción de desbordamiento de TIMER0
void clock_isr(){

   if(cont==763){       //Impresiones cada 10 seg
      convert=read_adc();        // Obtener el resultado de la conversion
      res=volts(convert);        //Conversión a volts

      //Impresión por transmisión a puerto Serie
      printf("DECIMAL=  %04ld\n\r",convert);
      printf("HEXDECIMAL=  %04lx\n\r\n\r",convert);

      //Impresión a Display LCD
      lcd_gotoxy(1,1);
      printf(lcd_putc,"%1.2f V\n",res);

      //Salida por el puerto B
      output_d(convert);
      cont=0;
   }else{
      cont++;        //Contador para realizar impresión
   }
}

float volts(long con){
   res=(float)con*(0.01953);     //Conversión por la resolución
   return res;
}
```

E3_P10.c

```

void main(){
   lcd_init();               //Inicia le Display LCD
   setup_port_a(ALL_ANALOG);  //Define al puerto A como analógico
   setup_adc(ADC_CLOCK_INTERNAL); // Define frecuencia de muestreo del
                                  //convertidor A/D
   set_adc_channel(0);          // Configura el canal a usar
   delay_us(20); // retardos

   set_timer0(0);               //Timer0 en 0X00
   setup_counters(RTCC_INTERNAL,RTCC_DIV_256);//Frecuencia y prescalado
   enable_interrupts(INT_RTCC); //Activación interrupción desbordamiento TIMER0
   enable_interrupts(GLOBAL);    //Activación interrupciones globales


   set_tris_d(0x00);             //Salidas por puerto D
   output_d(0x00);               //Salida 0x00 de puerto d
   while(TRUE){
   }

}
```

Código 3: Obtención e impresión de señal analógica en la interrupción de desbordamiento de TIMER0

Diagramas

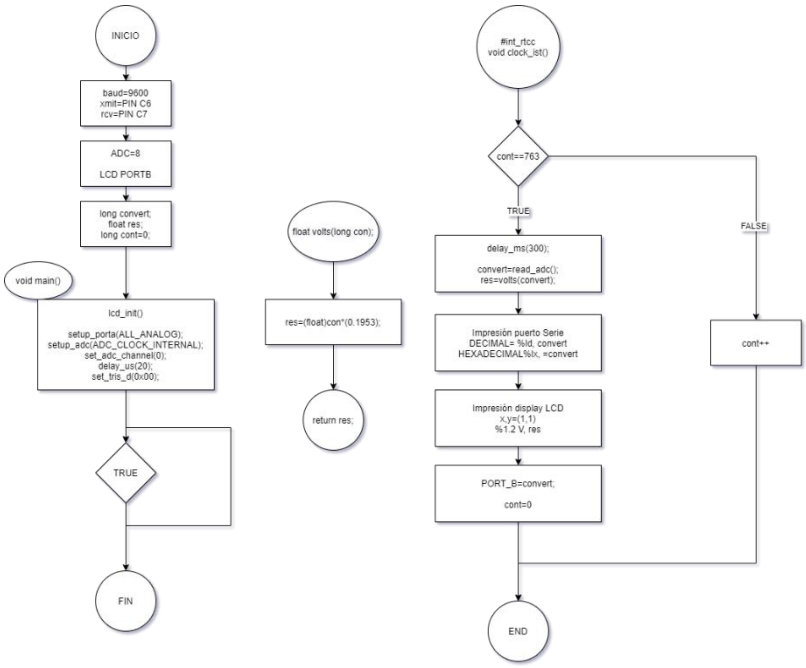
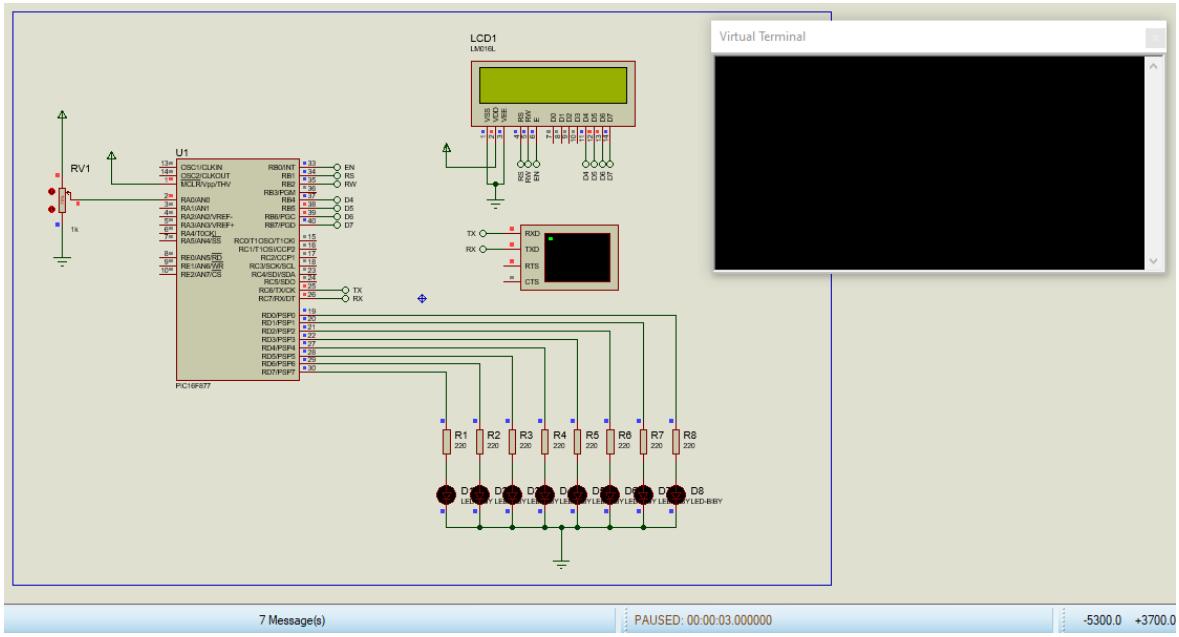
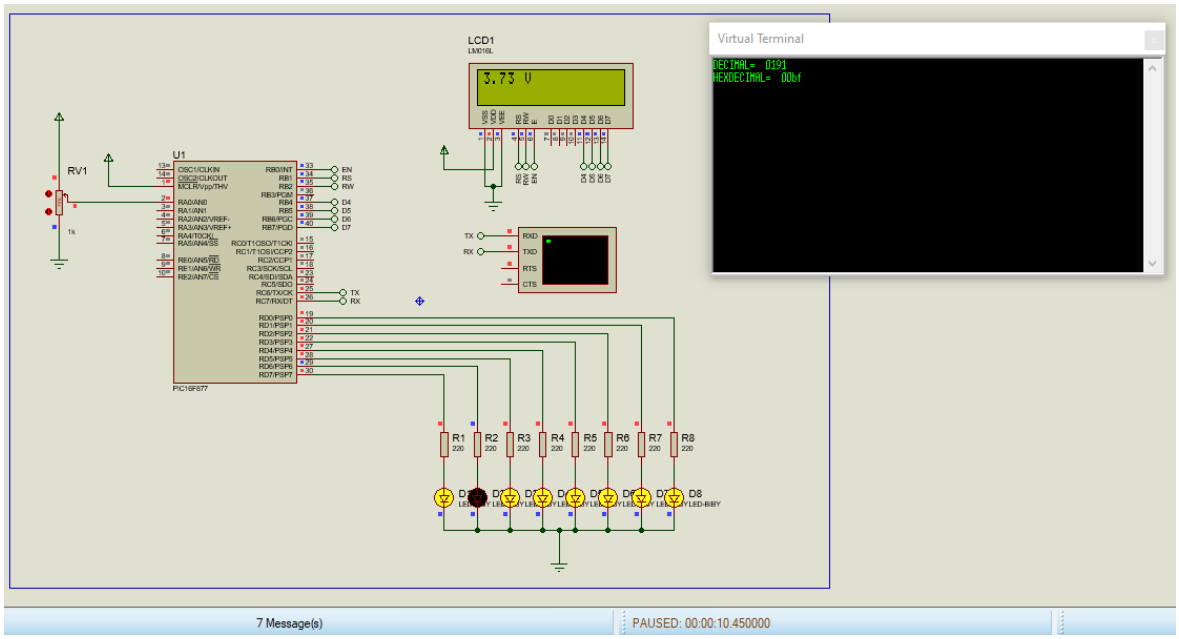


Diagrama 3: Código 3 con código de interrupción de TIMER0

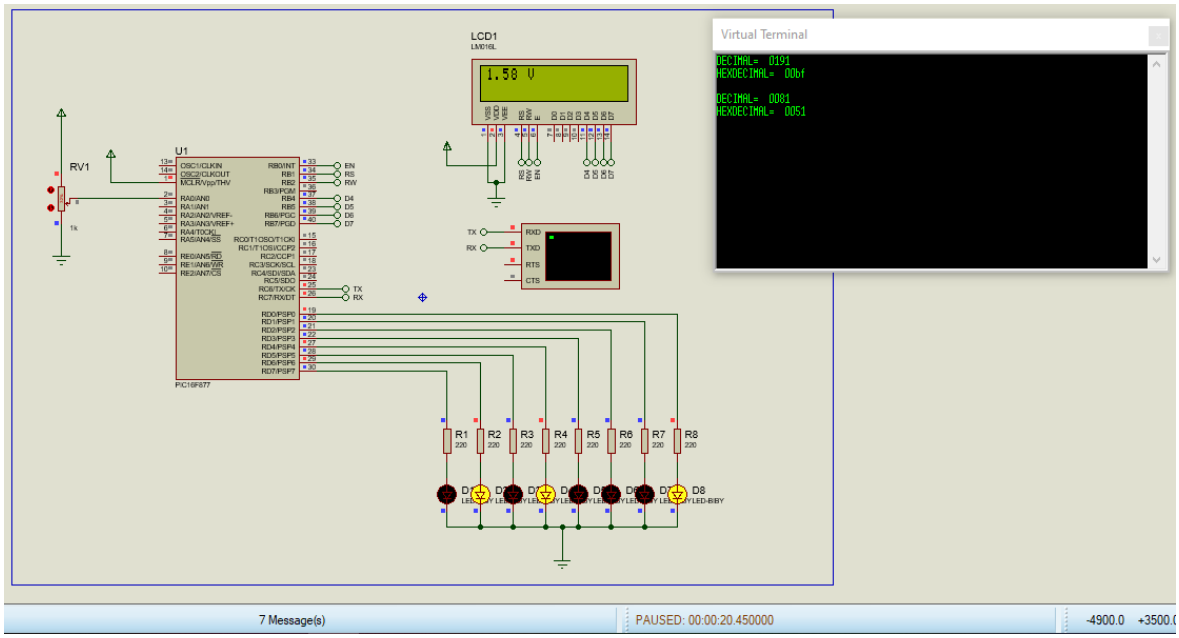
Pruebas



Pruebas 6: Estado inicial



Pruebas 7: Obtención e impresión de señal a los 10 seg.



Pruebas 8: Obtención e impresión de señal a los 20 seg.

4. Realizar un programa que muestre un contador binario de 8 bits en un puerto paralelo (usar leds y retardos de 250 ms), cada 10 segundos muestre el voltaje de la señal que ingrese en el canal 0 del convertidor A/D en el LCD y cada 25 segundos despliegue en la terminal los nombres, número de cuenta, grupo de teoría y laboratorio del o los integrantes del equipo.

Código

E4_P10.c

```c
#include <16f877a.h>   // Librería del microcontrolador
#device adc=8
#fuses HS,NOWDT,NOPROTECT //activa alta velocidad y no protege el código
#use delay(clock=20000000) //f=20Mhz
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)
#define use_portb_lcd true
#include <lcd.c>
#org 0x1F00, 0x1FFF void loader16F877(void) {}

int  cont_hex=0;     //Contador de 8 bits
int  convert=0;      //Conversión de A/D
long cont_v=0;       //Contador para impresión de voltajes
long cont_n=0;       //Contador para impresión de info en serie
float res=0;         //Conversión a Volts
```

E4_P10.c

```c
void clock_isr(){
//código de la rutina
//printf("%ld\n\r",cont_v);
   cont_v++;
   cont_n++;
   if(cont_v==40){         //10 segundos conversión e impresión de Volts en LCD
      delay_us(20); // retardos
      convert=read_adc();     //CONVERTIDOR A/D
      res=(float)convert*(0.01953); //Conversión a volts, 0.1953 resolución
      lcd_gotoxy(1,1);
      printf(lcd_putc,"%1.2f V",res);     //Impresión
      cont_v=0;
   }

   if(cont_n==100){                //Impresión cada 25 seg
      printf("Alumno:Murrieta Villegas, Alfonso\n\r");
      printf("\t-->No. de Cuenta: 315048937\n\r");
      printf("\n\r");
      printf("Alumno:Reza Chavarria Sergio Gabriel\n\r");
      printf("\t-->No. de Cuenta: 315319077\n\r");
      printf("\n\r");
      printf("Alumno:Valdespino Mendieta, Joaquin\n\r");
      printf("-->No. de Cuenta: 315115501\n\r");
      printf("\n\r");
      printf("Grupo de teoria: 01\n\r");
      printf("Grupo de laboratorio: 04\n\r");
      printf("\n\r\n\r");
      cont_n=0;
   }
}
```

E4_P10.c

```
47
48  void main(){
49     lcd_init();
50     setup_port_a(ALL_ANALOG);  //Define al puerto A como analógico
51     // Define frecuencia de muestreo del convertidor A/D
52     setup_adc(ADC_CLOCK_INTERNAL);
53     set_adc_channel(0);          // Configura el canal a usar
54     delay_us(20); // retardos
55
56     set_timer0(0); // Inicia TIMER0 en 00H
57     setup_counters(RTCC_INTERNAL,RTCC_DIV_256); //Fuente de reloj y pre-divisor
58     enable_interrupts(INT_RTCC); //Habilita interrupción por TIMER0
59     enable_interrupts(GLOBAL); //Habilita interrupciones generales
60
61     set_tris_d(0x00);
62      while(true){
63
64        output_d(cont_hex);      //Salida de contador por puerto D
65        if(cont_hex==0xff){      //Si llega a 0xff reinicia conteo
66           cont_hex=0;
67        }else{                   //Aumenta el contador
68           cont_hex++;
69        }
70        delay_ms(250);
71
72
73     }
74  }
```

Código 4: Contador de 8 bits con interrupción TIMER0

Diagramas

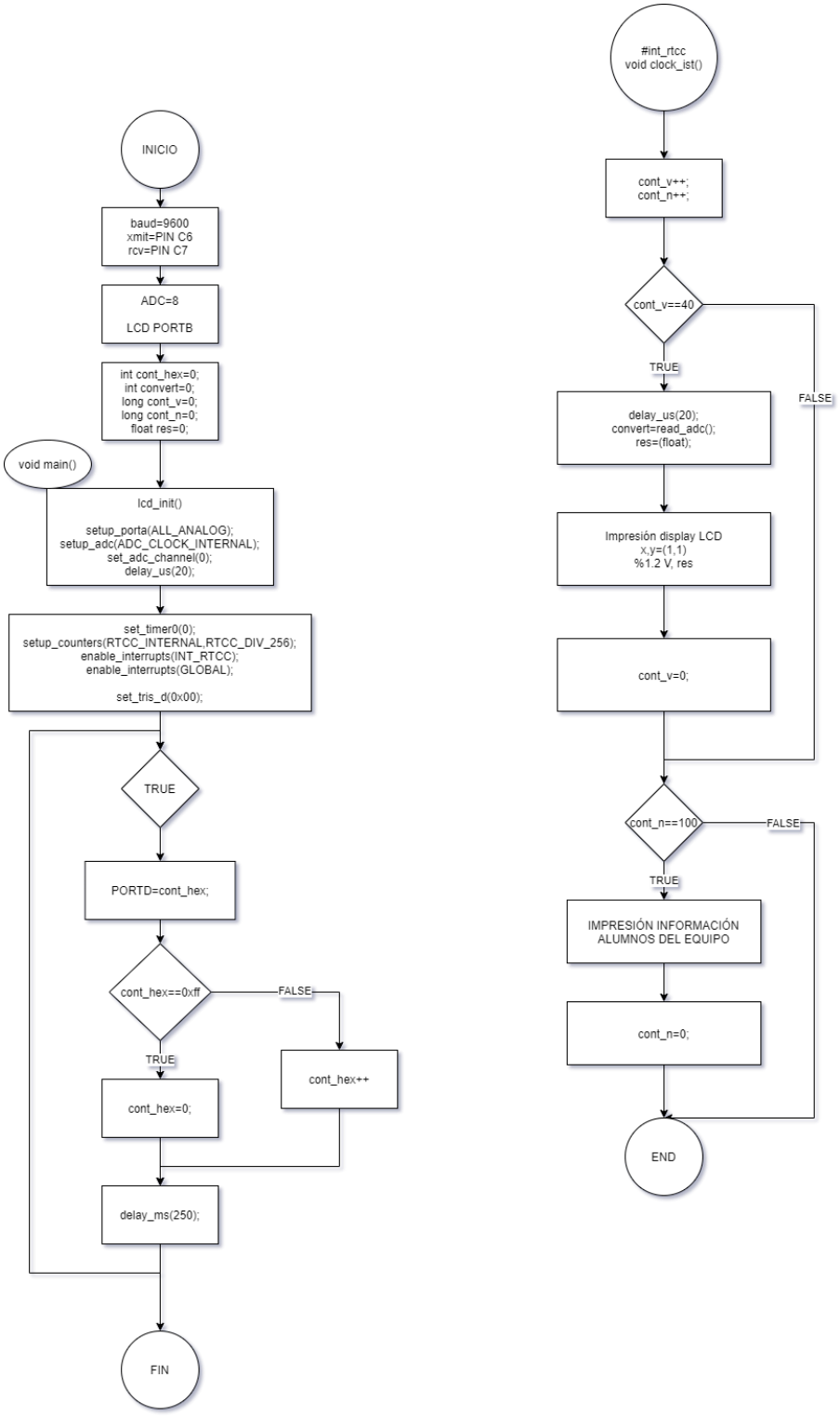
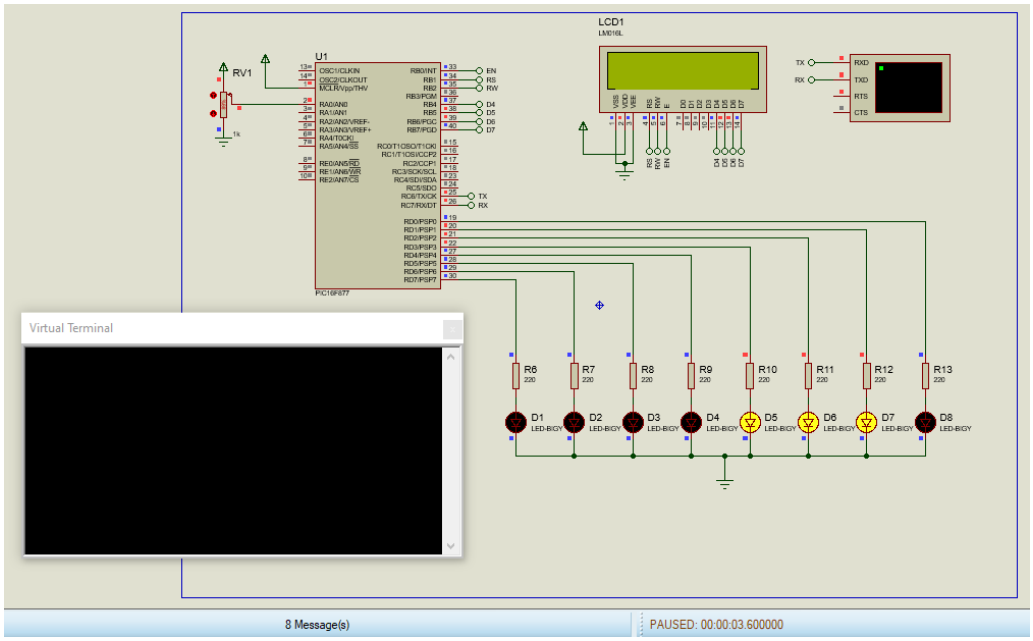
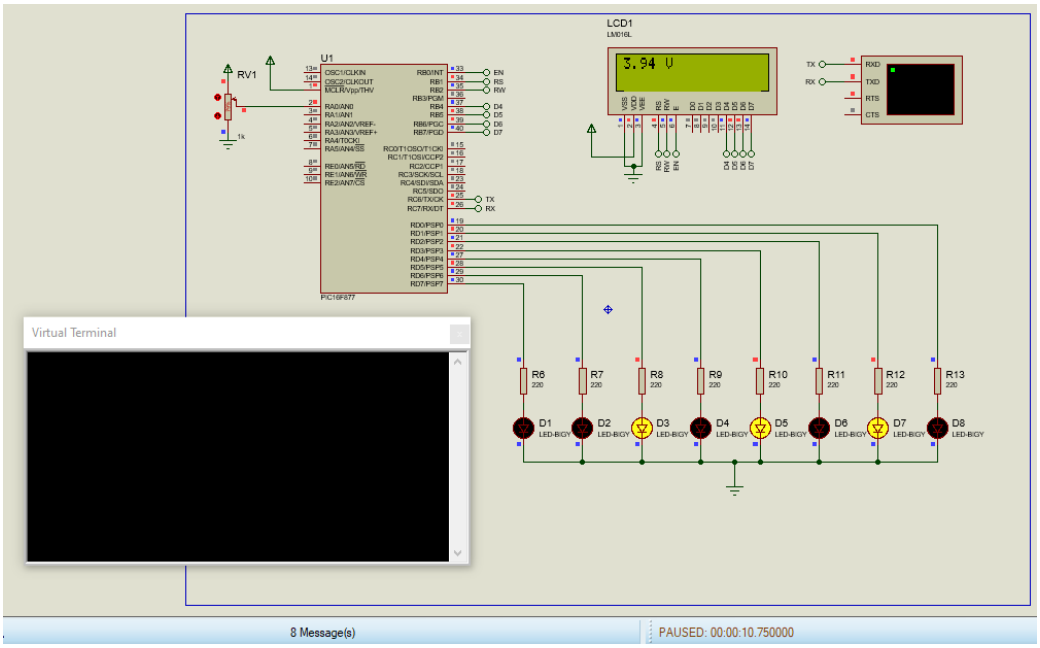


Diagrama 4: Código 4 con interrupción de timer 0

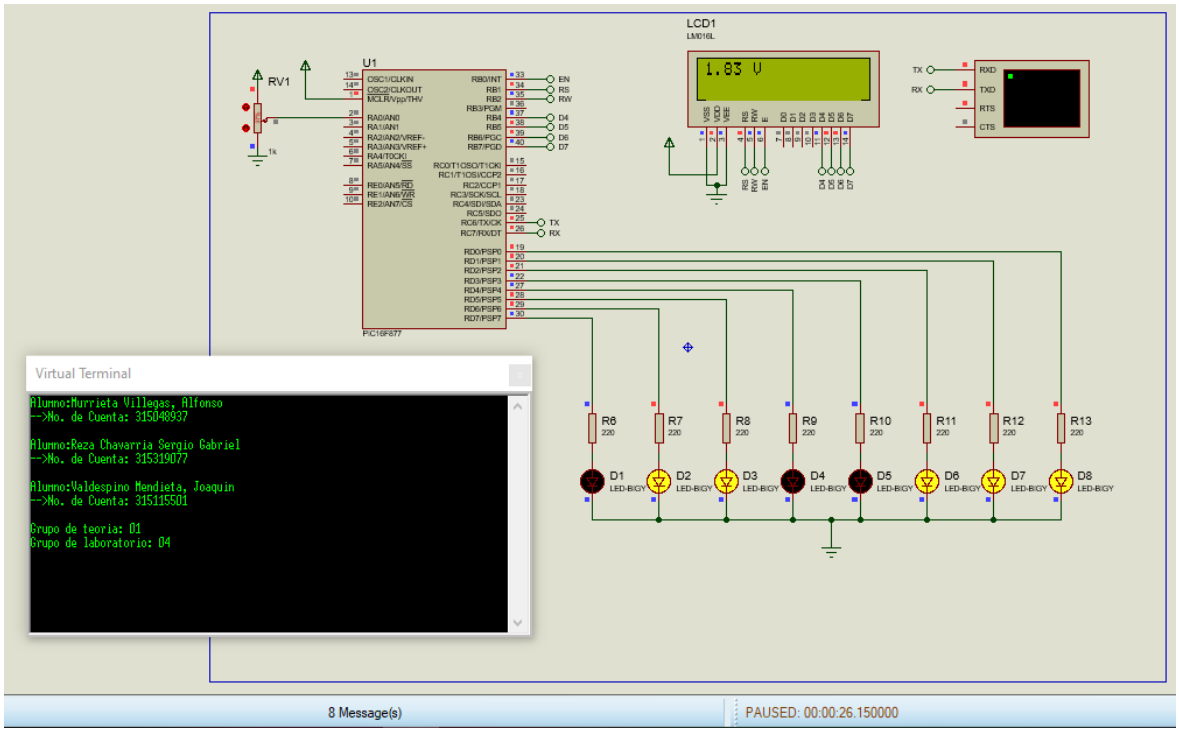
Pruebas



Pruebas 9: Conteo de 8 bits inicial



Pruebas 10: Obtención e impresión de señal en LCD a los 10 seg



Pruebas 11: Impresión de información del equipo en terminal a los 25 seg

5. Realizar un programa que permita atender las interrupciones indicadas en la tabla 10.2 y realice las acciones indicadas en esta, usando el dispositivo que considere para cada caso. El programa principal ejecutará un contador decimal ascendente y descendente, de 0 a 20 y 20 a 0 de manera indefinida con retardos de 1 segundo (usar display de 7 segmentos)

Interrupción	Acción
RB0	Despliegue la cuenta de las veces que ha sido activada
Recepción de datos del puerto serie	Cada que llegue un dato muestre un mensaje y las veces que ha ocurrido este evento
RB4 – RB7	Cuando alguna de los pines cambie de bajo a alto, indique en cual de ellos ha ocurrido
Desbordamiento TIMER0	Contador de ocho bits; cambio cada 200 ms

Tabla 10.2 Acciones actividad 5

Código

E5_P10.c

```
#include <16f877a.h>   // Librería del microcontrolador
#fuses HS,NOPROTECT //activa alta velocidad y no protege el código
#use delay(clock=20000000) //f=20Mhz
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)
#org 0x1F00, 0x1FFF void loader16F877(void) {}
long cont_RB0=0;//Contador de INTERRUPCIÓN RB0
long rec_data=0;//Contador de recibimiento de datos por serie
int  cont_hex=0;//Contador de 8 bits

long contador=0;
long aux=0;
int unidad=0;//Unidades de contador decimal en display de 7 segm
int decena=0;//Decenas de contador decimal en display de 7 segm
int asc_dec=0;//Conteo ascendente=0, descentende=1
char info;//Caracter obtenido

#int_rtcc      //Interrupción desbordamiento TIMER0
void clock_isr(){

   aux++;
   if(aux==15){
      output_d(cont_hex);
      cont_hex++;
      aux=0;
   }
   if(cont_hex==0xff){
      cont_hex=0;
   }
}
```

E5_P10.c

```
#int_rb        //Interrupción RB4-RB7
void port_rb(){
//Revisa que pin fue activado
   if(input(PIN_B4)){
      printf("INTERRUPCIÓN RB4 ACTIVADA\n\r");
   }else if(input(PIN_B5)){
      printf("INTERRUPCIÓN RB5 ACTIVADA\n\r");
   }else if(input(PIN_B6)){
      printf("INTERRUPCIÓN RB6 ACTIVADA\n\r");
   }else if(input(PIN_B7)){
      printf("INTERRUPCIÓN RB7 ACTIVADA\n\r");
   }
}

#int_ext         //Interrupción de flanco de RB0
void detecta_rb0(){
   cont_RB0++;

   printf("INTERRUPCIÓN RB0: %ld\n\r",cont_RB0);
}

#int_rda          //Interrupción de recibimiento de datos por puerto serie
void recepcion_serie(){
//código de la rutina de interrupción
   info=getchar();
   rec_data++;
   printf("\tDATOS RECIBIDOS <%c>. TOTAL=%ld\n\r",info,rec_data);
}
```

E5_P10.c

```
59
60  void main(){
61     set_timer0(0); // Inicia TIMER0 en 00H
62     setup_counters(RTCC_INTERNAL,RTCC_DIV_256); //Fuente de reloj y pre-divisor
63     enable_interrupts(INT_RTCC); //Habilita interrupción por TIMER0
64     ext_int_edge(L_TO_H);
65     enable_interrupts(INT_RB);
66     enable_interrupts(INT_EXT);
67     enable_interrupts(INT_RDA);
68     enable_interrupts(GLOBAL); //Habilita interrupciones generales
69     set_tris_a(0x00);
70     set_tris_e(0x00);
71     set_tris_d(0x00);
72      while(true){
73
74        output_a(unidad);
75        output_e(decena);
76        if(asc_dec==0){   //Conteo Ascendente
77           contador++;
78           unidad++;
79           if(contador==10){ //Caso 10-19
80              unidad=0;
81              decena=1;
82           }else if (contador==20){   //Caso 20
83              unidad=0;
84              decena=2;
85              asc_dec=1;              //Activar conteo descendente
86           }
87
```

E5_P10.c

```
87
88        }else{        //Conteo descendente
89           contador--;
90           unidad--;
91           if(contador==19){ //Caso 10-19
92              decena=1;
93              unidad=9;
94           }else if(contador==9){  //Caso 0-9
95              unidad=9;
96              decena=0;
97           }else if(contador==0){  //Caso 0
98              unidad=0;
99              decena=0;
100             asc_dec=0;           //Activar conteo ascendente
101          }
102
103       }
104       delay_ms(1000);
105    }
106 }
```

Código 5: Contador decimal con interrupciones múltiples

Diagramas

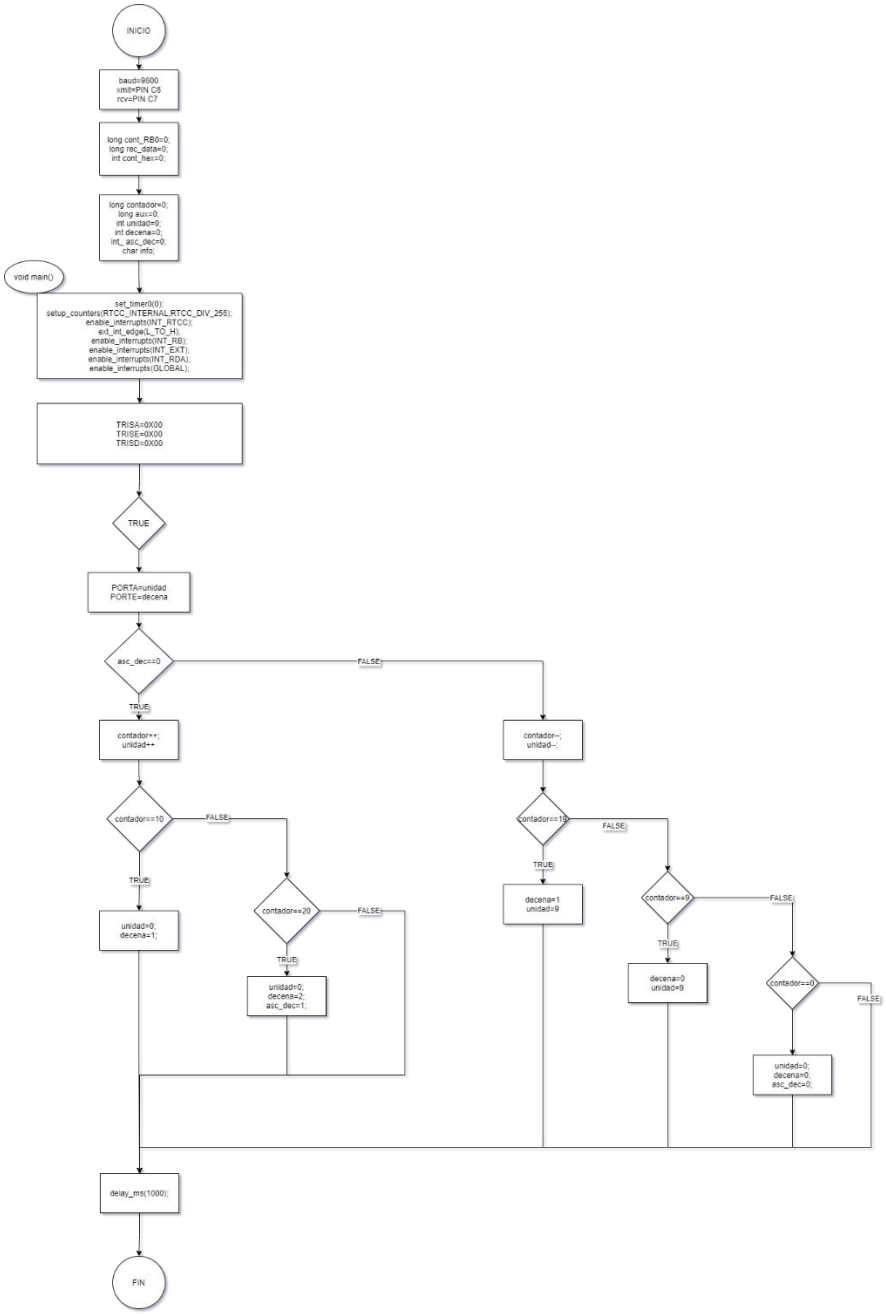


Diagrama 5: Código inicial del ejercicio 5

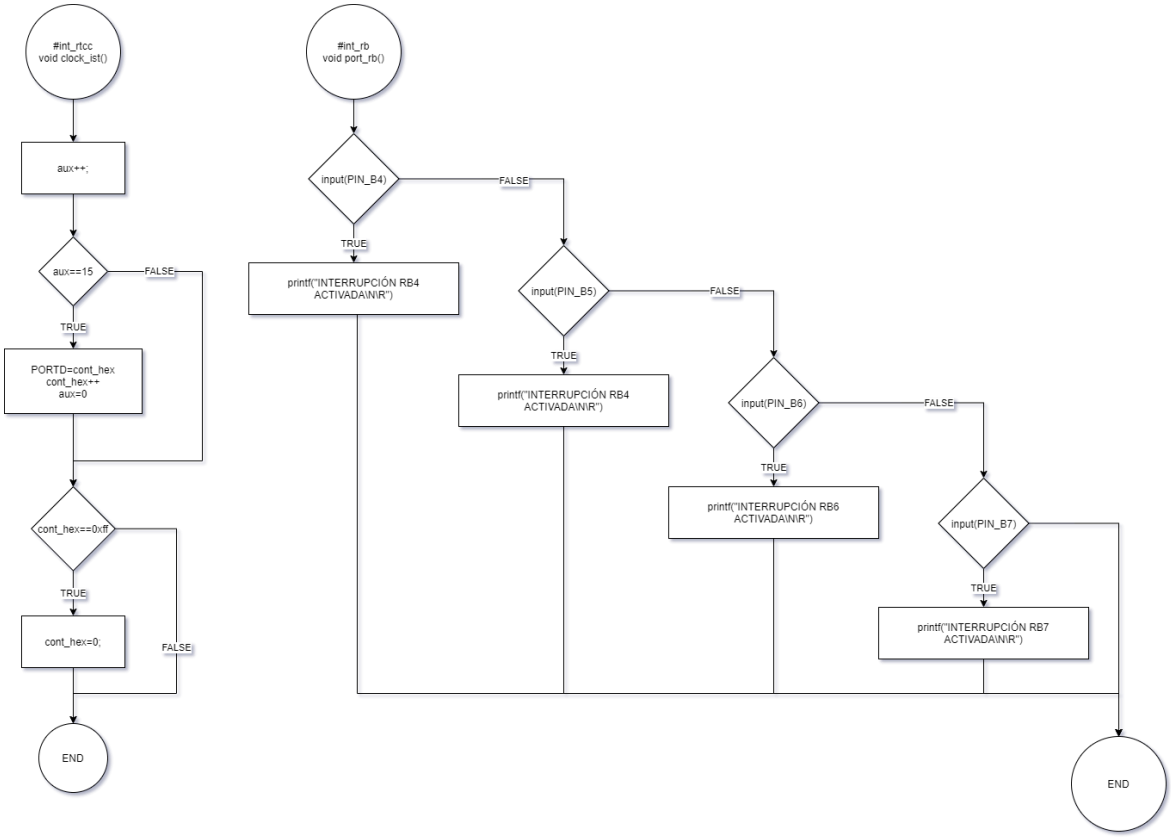


Diagrama 6: Interrupción de RB4-7 y desbordamiento TIMER0

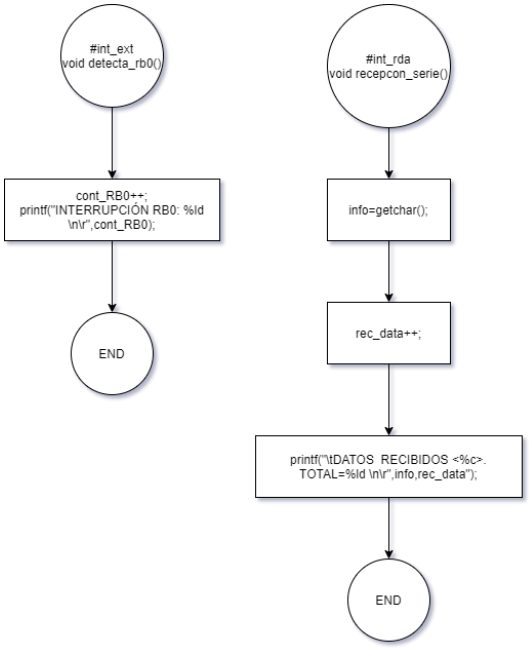
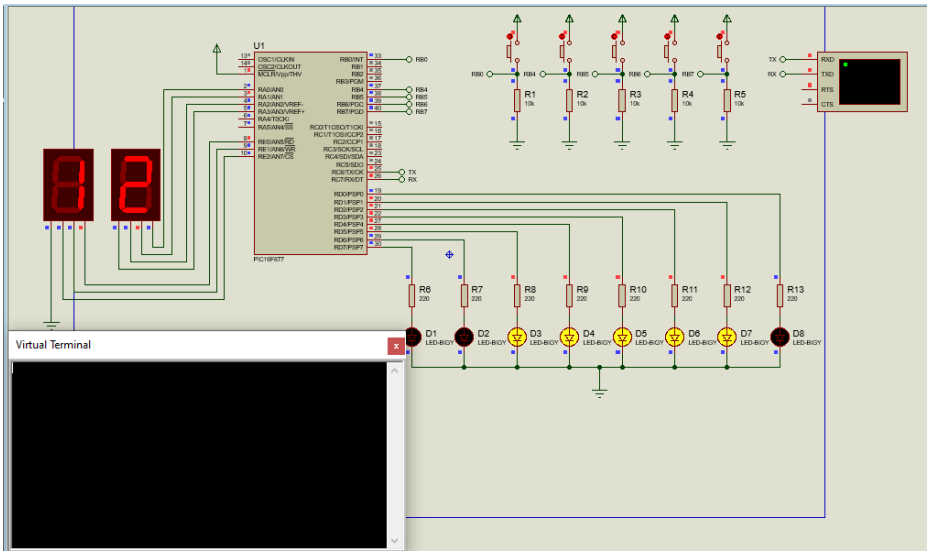
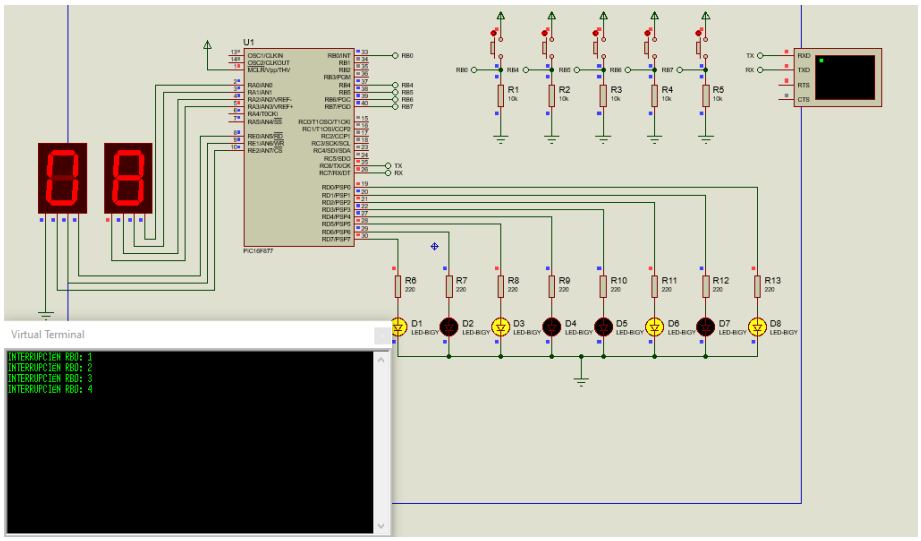


Diagrama 7: Interrupción de recepción del puerto serie y flanco de RB0

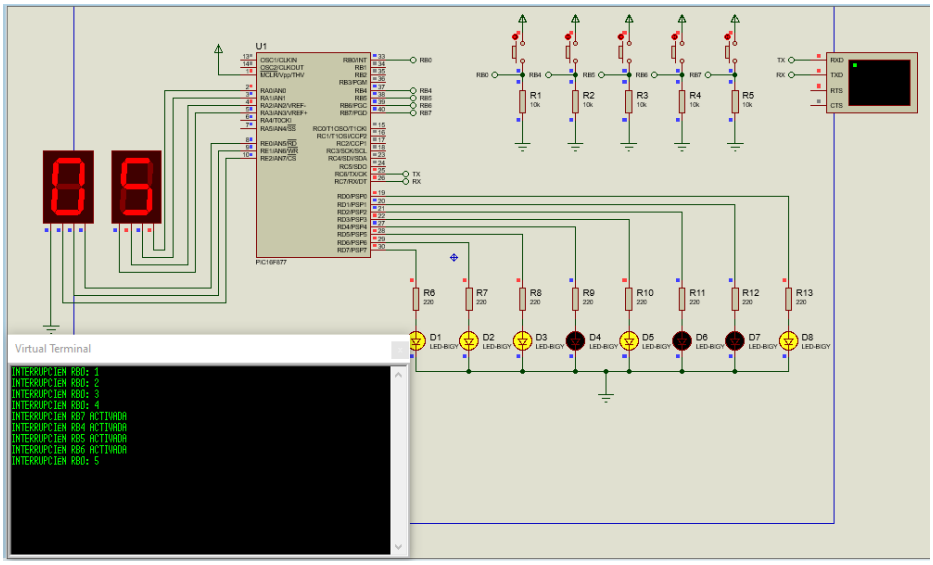
Pruebas



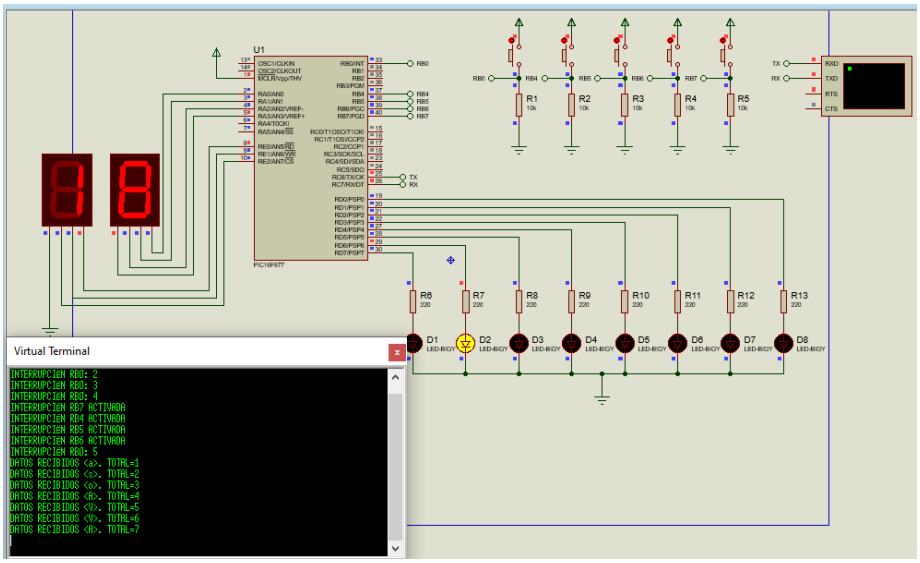
Pruebas 12: Conteo inicial



Pruebas 13: Conteo de flanco de RB0



Pruebas 14: Obtención de interrupción de RB4-7



Pruebas 15: Recepción de información por puerto serie

Conclusión

Murrieta Villegas Alfonso

En la presente práctica aprendimos una de las acciones más comunes de cualquier sistema que son las interrupciones, de manera general a lo largo de diferentes materias hemos aprendido que las interrupciones son un componente esencial sobre todo al trabajar con un sistema que debe ser sensible a fenómenos que lo afecten, ya sea que estos estén planeados o no.

De hecho, es tal el caso que existe paradigmas como el orientado a eventos que están precisamente planteados para este tipo de situaciones, en el caso concreto de los sistemas vistos en Microcomputadoras sobre todo son para brindarnos una mayor flexibilidad, por ejemplo, es el caso de cuando hacemos uso de distintos puertos.

Reza Chavarria Sergio Gabriel

Los ejercicios de las prácticas anteriores se centraban en acciones realizadas en el bloque principal de nuestros programas, esto implicaba el manejo individual de las instrucciones a realizar. Con el uso de las interrupciones se puede dar un manejo simultaneo del programa principal y de las instrucciones realizadas a partir de elementos externos que nos proporciona un aumento en la funcionalidad de proyectos.

Los manejos diferentes de las interrupciones, como el desbordamiento de TIMER0, obtención de información por el puerto serial o por los flancos de RB, proporcionan diferentes maneras de manejar los programas a realizar como si fuera una función normal.

Valdespino Mendieta Joaquin

En la presente práctica, pudimos implementar y comprender un evento variante de lo más común y esencial, que son las interrupciones, permitiendo un manejo en el flujo de datos ramificado, permitiendo realizar tareas a la par de un bloque principal, para

manipular elementos o periféricos de nuestro microcontrolador, permitiendo la escalabilidad y modularidad de los proyectos, como bien se dijo mediante diversos ejercicios presentados en la práctica.