UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

LABORATORIO DE MICROCOMPUTADORAS

PRÁCTICA 9:

PROGRAMACIÓN C, CONVERTIDOR A/D E INTERRUPCIONES

GRUPO: 12

PROFESORA: M.I LOURDES ANGÉLICA QUIÑONES JUAREZ

ALUMNO: CHÁVEZ DELGADO JORGE LUIS

N° DE CUENTA: 312217493

FECHA DE ASIGNACIÓN: *07/04/17*

FECHA DE ENTREGA: 18*/04/17*

PRÁCTICA 7: Puerto Serie SCI (ASÍNCRONO)

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

Ejercicio 1:

Para este ejercicio, configuramos el CAD para obtener la señal del potenciometro y mostrar la conversión en el puerto B.

Código:

```
#include <16f877.h>

#device adc=8 //en caso de emplear el conv. A/D indica resolución de 8 bits

#fuses HS,NOPROTECT

#use delay (clock=20000000)

#use rs232 (baud=38400,xmit=PIN_C6,rcv=PIN_C7)

#org 0x1F00, 0x1FFF void loader16F877(void){}

int converse; //Donde se almacenarà el resultado de la conversiòn

void main(){

      setup_port_a(ALL_ANALOG); /*Habilitamos el puerto análogico, en ensamblador hubieramos

      tenido que cambiar de banco para configurar con un H'00' para entrada analógica

    al ADCON1 y ser configurado el puerto A como entrada*/

     setup_adc (ADC_CLOCK_INTERNAL); //Indicamos que utilizaremos el reloj interno del pic

     set_adc_channel (3);//Indicamos que utilizaremos el canal 3 para la señal

    while(1){

      delay_us(20);//Retardo de 20 microsegs

          converse=read_adc(); //Realiza la conversión y la guarda en converse
```

```
      printf("La conversiòn es: =%x\r",converse);
       output_b(converse); //Se muestra el valor de la conversión en el
puerto B
    }
    }
```

Ejercicio 2:

En este ejercicio tomamos el resultado de la conversión y lo imprimimos en la hiperterminal cada diez segundos, ya que si no se vería parpadeando todo el tiempo.

Código:

```
#include <16f877.h>
#device adc=8 //en caso de emplear el conv. A/D indica resolución de 8
bits
#fuses HS,NOPROTECT
#use delay (clock=20000000)
#use rs232 (baud=38400,xmit=PIN_C6,rcv=PIN_C7)
#org 0x1F00, 0x1FFF void loader16F877(void){}
int conv; //Variable para guardar la conversiòn
long cont=0; //Contador para activar la interrupciòn
//Funcion de la interrupcion
#int_RTCC
clock_isr(){
cont++; //Implementa el contador, en ensamblador cont equ h'24'
//Determinamos 10 segundos mediante la formula
if(cont==769){ //En ensamblador se utilizaría btfsc para verificar si el
bit cambio y pasar a la subrutina que ejecuta el codigo dentro del if
printf("La conversiòn es: =%x\r",conv*0.019); //Cuando se active la
interrupciòn mostrarà el valor de la conversion
cont=0;
  }
```

```
}

void main(){

set_timer0(0); //Inicia timer0 en 00H

setup_counters(RTCC_INTERNAL,RTCC_DIV_256); //Fuente de reloj y pre-
divisor

enable_interrupts(INT_RTCC); //Habilita interrupcion del timer0

enable_interrupts(GLOBAL); //Habilita interrupciones generales


setup_port_a(ALL_ANALOG); /*Habilitamos el puerto análogico, en
ensamblador hubieramos

      tenido que cambiar de banco para configurar con un H'00' para
entrada analógica

    al ADCON1 y ser configurado el puerto A como entrada*/

setup_adc (ADC_CLOCK_INTERNAL);//Indicamos que utilizaremos el reloj
interno del pic

set_adc_channel (3);//Indicamos que utilizaremos el canal 3 para la señal

while(1){

delay_us(20); //Retraso para que termine la conversion


conv=read_adc(); //Guardamos el resultado de la conversiòn


}

}

//t=tciclo de reloj (255)(256) Tciclo
```

Ejercicio 3:

En este ejercicio realizamos lo mismo que en el anterior sólo que a los 30 segundos se imprimirá en la hiperterminal “El mejor grupo de microcomputadoras”.

Código:

```
#include <16f877.h>

#device adc=8 //en caso de emplear el conv. A/D indica resolución de 8
bits

#fuses HS,NOPROTECT
```

```
#use delay (clock=20000000)

#use rs232 (baud=38400,xmit=PIN_C6,rcv=PIN_C7)

#org 0x1F00, 0x1FFF void loader16F877(void){}

int conv; //Variable para guardar la conversiòn

long cont=0; //Contador para activar la interrupciòn

#int_RTCC //Para generar una interrupción

clock_isr(){

cont++; //Implementa el contador

//Determinamos 30 segundos mediante la formula

if(cont==2307){

printf("El mejor laboratorio de microcomputadoras\n"); //Cuando se
active la interrupciòn mostrarà el valor de la conversion

cont=0;

  }

}

void main(){

set_timer0(0); //Inicia timer0 en 00H

setup_counters(RTCC_INTERNAL,RTCC_DIV_256); //Fuente de reloj y pre-
divisor, configura la razón

//de tiempo en la que se prenderá;

enable_interrupts(INT_RTCC); //Habilita interrupcion del timer0

enable_interrupts(GLOBAL); //Habilita interrupciones generales


setup_port_a(ALL_ANALOG); /*Habilitamos el puerto análogico, en
ensamblador hubieramos

      tenido que cambiar de banco para configurar con un H'00' para
entrada analógica

    al ADCON1 y ser configurado el puerto A como entrada*/

setup_adc (ADC_CLOCK_INTERNAL);//Indicamos que se usará el reloj
interno del pic

set_adc_channel (3);//Configuramos el canal 3
```

```
while(1){
delay_us(20); //Retraso para que termine la conversion

conv=read_adc(); //Guardamos el resultado de la conversiòn

}
}
```

Ejercicio 4:

En este ejercicio configuramos como entrada para el puerto A los dipswitch para mandar un mensaje a la terminal, si se habilita manda el mensaje “Pbx Activado” y si se deshabilita “Pbx de Bajada”.

Código:

```
#include <16f877.h>
#device adc=8 //en caso de emplear el conv. A/D indica resolución de 8 bits
#fuses HS,NOPROTECT
#use delay (clock=20000000)
#use rs232 (baud=38400,xmit=PIN_C6,rcv=PIN_C7)
#org 0x1F00, 0x1FFF void loader16F877(void){}
int var1; //En ensamblador hubieramos declarado
 var1 equ h'Xx'
#int_rb //Habilitamos rb para los dipswitch
int_p(){
if(input(pin_b4))//En ensamblador hubieramos llamadoa  una subrutina
printf("\nPB4 ACTIVADA"); //Manda un mensaje si se active el bit RB4
else
printf("\nPB4 DE BAJADA"); //Manda un mensaje si se active el bit RB4
if(input(pin_b5))
printf("\nPB5 ACTIVADA"); //Manda un mensaje si se active el bit RB5
```

```
else
printf("\nPB4 DE BAJADA"); //Manda un mensaje si se active el bit RB4
if(input(pin_b6))
printf("\nPB6 ACTIVADA"); //Manda un mensaje si se active el bit RB6
else
printf("\nPB4 DE BAJADA"); //Manda un mensaje si se active el bit RB4
if(input(pin_b7))
printf("\nPB7 ACTIVADA"); //Manda un mensaje si se active el bit RB7
else
printf("\nPB4 DE BAJADA"); //Manda un mensaje si se active el bit RB4
}
void main(){
setup_counters(RTCC_INTERNAL,RTCC_DIV_256); //Fuente de reloj y pre-
divisor,configura la razón
//de tiempo en la que se prenderá;
enable_interrupts(INT_RB); //Habilita interrupcion del timer0, cuando se
reciba por RB
enable_interrupts(GLOBAL); //Habilita interrupciones generales
while(1){
var1=input_b(); //Ciclo para poder cambiar y asi activar la interrupcion
de puerto b
}
}
```

Conclusiones:

Logramos configurar con lenguaje C el convertidor análogico digital, para utilizarlo en conjunto con el puerto A para entradas y el puerto B para mandar salidas, aunque también enviamos mensajes a la terminal. Además nos dimos cuenta que las interrupciones son de gran ayuda cuando queremos enviar mensajes de alerta , por ejemplo si tenemos un sistema de control de cuartos, que nos avise que cuartos estan abiertos, o cuales se han abierto en un determinado momento.