



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA
DIVISIÓN DE INGENIERÍA ELÉCTRICA
DEPARTAMENTO DE INGENIERÍA EN COMPUTACIÓN

PRACTICAS
LABORATORIO DE MICROCOMPUTADORAS
Basadas en el Microcontrolador PIC16F877

RUBÉN ANAYA GARCÍA
JESÚS SAVAGE CARMONA
CARLOS MUNIVE VÁZQUEZ

CIUDAD UNIVERSITARIA

Introducción

Se ha diseñado el siguiente material para su uso en el laboratorio de microcomputadoras, con la idea principal de servir como una guía a los alumnos que cursan el laboratorio, se proponen un conjunto de 9 prácticas, las cuales contienen información introductoria, que servirá al alumno como antecedente al inicio de la realización de cada sesión y con esta forma clarificar los objetivos de la práctica.

Se ha propuesto el uso del microcontrolador PIC16F877; no siendo este el único que podría atender, ya que en realidad se intenta poner en práctica los conocimientos que en la teoría se aprenden, por lo que los ejercicios aquí propuestos inducen y llevan al alumno a un mejor entendimiento de estos dispositivos y a un mayor aprovechamiento de los mismos.

Se iniciará con la familiarización del lenguaje ensamblador del PIC y la utilización del ambiente MPLAB, el cuál servirá para editar, ensamblar y simular los programas capturados. La práctica dos enfatizará al uso de un sistema de desarrollo con un microcontrolador, donde se desea que el alumno ejecute directamente programas previamente ensamblados y en algunos casos simulados directamente en la tarjeta física, así mismo de la manera usada para cargar los programas en el dispositivo.

Las prácticas 3, 4 y 5 afianzará el uso de los puertos paralelos mediante posibles interfaces que pueden conectar al el microcontrolador de manera sencilla; generando salidas de despliegue a través de leds, controlar motores de corriente directa, tomando lecturas de switches y sensores externos; para el término de éstas sesiones el alumno pueda tener una visión más amplia de la infinidad de aplicaciones que se pueden realizar con los puertos paralelos.

La práctica seis mostrará al alumno las ventajas de contar con un convertidor analógico-digital dentro de los recursos del microcontrolador, además de ampliar las posibilidades de aplicación de este recurso.

En la práctica siete aplicará la comunicación serie en la modalidad asíncrona, con la finalidad que el alumno controle las funciones del PIC por medio de otro dispositivo serie, que puede ser la computadora personal.

Una vez que el alumno ha logrado controlar y programar los recursos de un microcontrolador, mediante la programación en lenguaje ensamblador, se proponen dos prácticas en las cuales el alumno resolverá los ejercicios propuestos mediante la programación en lenguaje C. La práctica ocho usará puertos paralelos y el puerto serie, mientras que la práctica nueve empleará el convertidor A/D y las aplicaciones utilizando programación mediante la técnica de programación con interrupciones.

Al concluir las prácticas, el alumno deberá haber comprendido las ventajas que se tiene al realizar aplicaciones con microcomputadoras así mismo describir los diferentes elementos constituidos de éstas.

Contenido

Prácticas

Practica No. 1	Introducción General a un microcontrolador.
Practica No. 2	Sistema mínimo de microcontrolador y puertos paralelos I.
Practica No. 3	Puertos Paralelos II (Control de acciones).
Practica No. 4	Puertos Paralelos III (Control de motores de CD).
Practica No. 5	Puertos Paralelos IV (Sensores ópticos).
Practica No. 6	Convertidor Analógico Digital.
Practica No. 7	Interfaz de Comunicación Serial Asíncrona (SCI).
Practica No. 8	Programación lenguaje C.
Practica No. 9	Programación lenguaje C.
	Proyecto Final

Con las prácticas propuestas se desea cubrir los objetivos que la teoría demanda los cuales se reproducen en este manual:

Objetivo de la teoría de Microcomputadoras

El alumno aprenderá y aplicará los conocimientos de la teoría y funcionamiento de los microprocesadores y su interconexión con diferentes circuitos periféricos para la construcción y funcionamiento de microcomputadoras. Diseñará y construirá aplicaciones utilizando microprocesadores y sus periféricos para diferentes sistemas, simulando aplicaciones industriales en tiempo real, así como aplicaciones científicas.

Atendiendo al siguiente temario

-
1. Conceptos básicos
 2. Conjunto de instrucciones
 3. Modos de direccionamiento
 4. Señales de control y diseño de un sistema con microprocesadores
 5. Periféricos e interfaces con microprocesadores
 6. Técnicas de diseño de sistemas con microprocesadores
 7. Características generales de microprocesadores de 16 y 32 bits
-

Laboratorio de Microcomputadoras
Práctica No. 1
Introducción General a un Microcontrolador PIC16F877

Objetivo. Familiarizar al alumno en el conocimiento del ensamblador, del simulador, el conjunto de instrucciones de un microcontrolador y ejecutar programas en tiempo de simulación.

Introducción

Algunas de las características más importantes que tiene el microcontrolador son:

- 8K de memoria FLASH
- 368 bytes de memoria RAM
- 255 bytes de memoria EEPROM
- 35 instrucciones
- 5 puertos paralelos (A, B, C, D, E)
- Convertidor Analógico Digital
- Comunicación Serie Asíncrona
- Comunicación Serie Síncrona (paralela, I2C)
- Tres módulos temporizadores
- Dos módulos CCP que pueden operar como Comparación, Captura o PWM
- 14 posibles fuentes de interrupción

Los registros disponibles para el programador son:

W	Registro de trabajo W
PC	Registro Contador de Programa
STATUS	Registro de banderas

Tanto los registros PC y STATUS están ubicados en localidades de memoria RAM, dentro de los bancos en los que se divide los 368 bytes de memoria de datos, como se muestra a continuación.

								File Address
Indirect addr. ^(*)	00h	Indirect addr. ^(*)	80h	Indirect addr. ^(*)	100h	Indirect addr. ^(*)	180h	
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h	
PCL	02h	PCL	82h	PCL	102h	PCL	182h	
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h	
FSR	04h	FSR	84h	FSR	104h	FSR	184h	
PORTA	05h	TRISA	85h		105h		185h	
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h	
PORTC	07h	TRISC	87h		107h		187h	
PORTD ^(*)	08h	TRISD ^(*)	88h		108h		188h	
PORTE ^(*)	09h	TRISE ^(*)	89h		109h		189h	
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah	
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh	
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch	
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh	
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved ⁽²⁾	18Eh	
TMR1H	0Fh		8Fh	EEDRHH	10Fh	Reserved ⁽²⁾	18Fh	
T1CON	10h		90h		110h		190h	
TMR2	11h	SSPCON2	91h		111h		191h	
T2CON	12h	PR2	92h		112h		192h	
SSPBUF	13h	SSPADDD	93h		113h		193h	
SSPCON	14h	SSPSTAT	94h		114h		194h	
CCPR1L	15h		95h		115h		195h	
CCPR1H	16h		96h		116h		196h	
CCP1CON	17h		97h	General Purpose Register 16 Bytes	117h	General Purpose Register 16 Bytes	197h	
RCSTA	18h	TXSTA	98h		118h		198h	
TXREG	19h	SPBRG	99h		119h		199h	
RCREG	1Ah		9Ah		11Ah		19Ah	
CCPR2L	1Bh		9Bh		11Bh		19Bh	
CCPR2H	1Ch		9Ch		11Ch		19Ch	
CCP2CON	1Dh		9Dh		11Dh		19Dh	
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh	
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh	
	20h		A0h		120h		1A0h	
General Purpose Register 96 Bytes		General Purpose Register 80 Bytes		General Purpose Register 80 Bytes		General Purpose Register 80 Bytes		
		accesses 70h-7Fh	EFh	accesses 70h-7Fh	16Fh	accesses 70h - 7Fh	1EFh	
			FFh	accesses 70h-7Fh	170h		1F0h	
Bank 0	7Fh	Bank 1	FFh	Bank 2	17Fh	Bank 3	1FFh	

Figura 1.1 Mapa de memoria de datos

El registro **STATUS**, además de indicar el estado de lo que ocurrió en la última operación, se dispone de banderas que permiten seleccionar el banco de memoria RAM donde se desea acceder.

RP1	RP0	BANCO	UBICACIÓN
0	0	0	00H-7FH
0	1	1	80H-FFH
1	0	2	100H-17FH
1	1	3	180H-1FFH

Tabla 1.1 Selección del banco de memoria RAM

Una plantilla de programa sería:

```
processor 16f877          ;Indica la versión de procesador
include <p16f877.inc>     ;incluye la librería de la versión del procesador
org 0H                   ;Carga al vector de RESET la dirección de inicio
goto inicio
inicio: org 05H           ;Dirección de inicio del programa del usuario
.....
.....
.....
end                       ;directiva de fin de programa
```

La llamada gama baja y media de PIC's a la que pertenece el PIC16F877 tiene el siguiente conjunto de instrucciones.

Mnemonic, Operands	Description	Cycles	14-Bit Opcode		Status Affected	Notes
			MSb	LSb		
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d Add W and f	1	00	0111 dfff ffff	C,DC,Z	1,2
ANDWF	f, d AND W with f	1	00	0101 dfff ffff	Z	1,2
CLRF	f Clear f	1	00	0001 1fff ffff	Z	2
CLRWF	- Clear W	1	00	0001 0xxx xxxx	Z	
COMF	f, d Complement f	1	00	1001 dfff ffff	Z	1,2
DECf	f, d Decrement f	1	00	0011 dfff ffff	Z	1,2
DECFSZ	f, d Decrement f, Skip if 0	1(2)	00	1011 dfff ffff		1,2,3
INCF	f, d Increment f	1	00	1010 dfff ffff	Z	1,2
INCFSZ	f, d Increment f, Skip if 0	1(2)	00	1111 dfff ffff		1,2,3
IORWF	f, d Inclusive OR W with f	1	00	0100 dfff ffff	Z	1,2
MOVF	f, d Move f	1	00	1000 dfff ffff	Z	1,2
MOVWF	f Move W to f	1	00	0000 1fff ffff		
NOP	- No Operation	1	00	0000 0xxx0 0000		
RLF	f, d Rotate Left f through Carry	1	00	1101 dfff ffff	C	1,2
RRF	f, d Rotate Right f through Carry	1	00	1100 dfff ffff	C	1,2
SUBWF	f, d Subtract W from f	1	00	0010 dfff ffff	C,DC,Z	1,2
SWAPF	f, d Swap nibbles in f	1	00	1110 dfff ffff		1,2
XORWF	f, d Exclusive OR W with f	1	00	0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF	f, b Bit Clear f	1	01	00bb bfff ffff		1,2
BSF	f, b Bit Set f	1	01	01bb bfff ffff		1,2
BTFSC	f, b Bit Test f, Skip if Clear	1 (2)	01	10bb bfff ffff		3
BTFSS	f, b Bit Test f, Skip if Set	1 (2)	01	11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW	k Add literal and W	1	11	111x kkkk kkkk	C,DC,Z	
ANDLW	k AND literal with W	1	11	1001 kkkk kkkk	Z	
CALL	k Call subroutine	2	10	0kkk kkkk kkkk		
CLRWDt	- Clear Watchdog Timer	1	00	0000 0110 0100	TO,PD	
GOTO	k Go to address	2	10	1kkk kkkk kkkk		
IORLW	k Inclusive OR literal with W	1	11	1000 kkkk kkkk	Z	
MOVLW	k Move literal to W	1	11	00xx kkkk kkkk		
RETFIE	- Return from interrupt	2	00	0000 0000 1001		
RETLW	k Return with literal in W	2	11	01xx kkkk kkkk		
RETURN	- Return from Subroutine	2	00	0000 0000 1000		
SLEEP	- Go into standby mode	1	00	0000 0110 0011	TO,PD	
SUBLW	k Subtract W from literal	1	11	110x kkkk kkkk	C,DC,Z	
XORLW	k Exclusive OR literal with W	1	11	1010 kkkk kkkk	Z	

Figura 1.2 Conjunto de instrucciones del PIC 16F877

Herramienta de desarrollo MPLAB

El MPLAB es uno de los llamados Ambientes de Desarrollo Integrado IDE, que permite escribir, ensamblar y simular un programa, e incluso usando cierto hardware, se puede simular en circuito y programar al microcontrolador. Este programa lo puedes bajar de manera gratuita de la dirección electrónica de Microchip(www.microchip.com).

Al ejecutar MPLAB, presenta una pantalla como la siguiente:

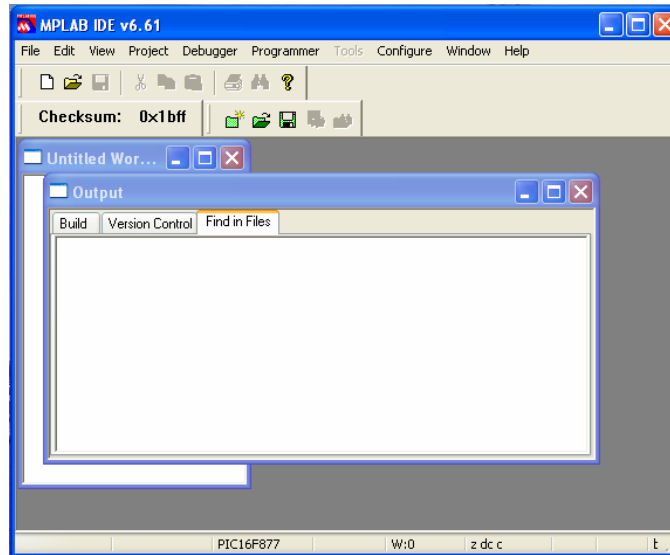


Figura 1.3 Entorno de MPLAB

En el menú **File** seleccionar **New**, entonces aparece la ventana de trabajo con el encabezado **Untitled**, escribir el programa en esta área, una vez terminado, salvarlo usando nuevamente el menú File y el submenú **Save as** del tipo ASM.

Para ensamblar el programa usar el comando **Project**, buscar el submenú **Quickbuild**, donde aparecerá incluido el nombre del programa a ensamblar que es el que está activo en el área de captura.

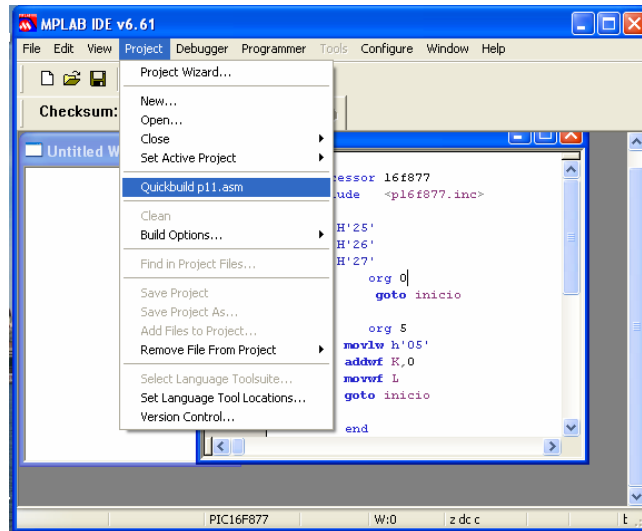


Figura 1.4 Ensamblar un programa

Si no existe problema en el proceso de ensamblado, genera el mensaje **BUILD SUCCEEDED**, lo cuál indica que el proceso de ensamblado ha concluido satisfactoriamente.

El siguiente proceso será simular el programa, para lo cuál del menú se elige el comando **View** y las opciones requeridas.

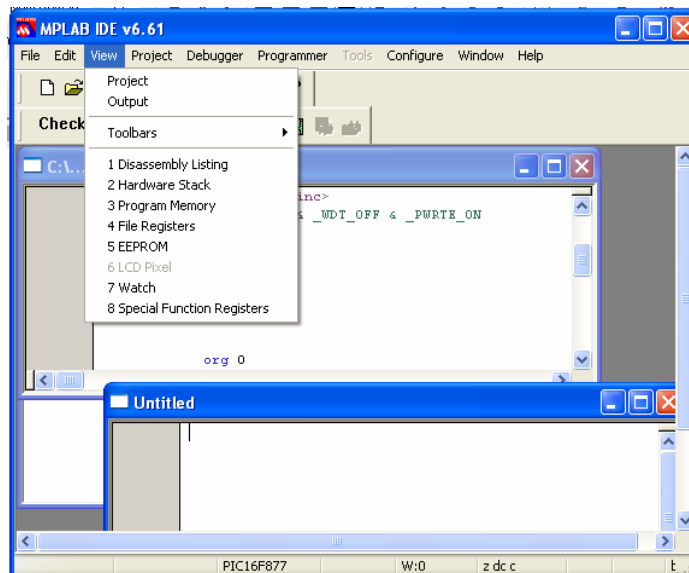


Figura 1.5 Selección de ventanas de visualización para el proceso de simulación

Por lo general solo se selecciona **File Registers**, el cuál muestra los registros y sus valores actuales; para modificar el contenido de alguna localidad, sólo se tiene que escribir el valor deseado y si el programa genera un valor, este será actualizado.

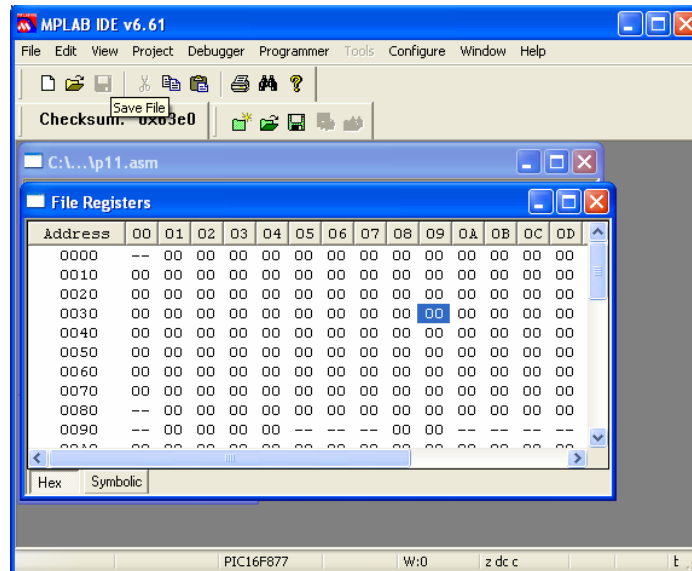


Figura 1.6 Mapa de memoria RAM

Para iniciar el proceso de simulación se debe seleccionar el simulador **MPLAB SIM**, accediendo al menú principal, dar click en **Debugger**, luego seleccionar **Select Tool** y entonces **Mplab Sim**; se habilitarán los iconos de simulación.

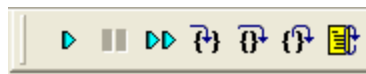


Figura 1.7 Iconos de simulación

Permitirá iniciar el proceso de simulación por instrucción o en forma continua, también es posible simular usando teclas de función, acceder al comando Debugger del menú principal.

Desarrollo. Para cada uno de los siguientes ejercicios, realizar los programas solicitados y simular el funcionamiento de ellos.

1.- Siguiendo las indicaciones previas, escribir el siguiente programa, ensamblar y simular el funcionamiento de este.

```

processor 16f877
include <p16f877.inc>

K    equ    H'26'
L    equ    H'27'

org    0
goto inicio
    
```

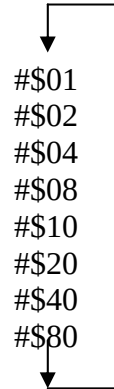
```
    org 5
inicio: movlw h'05'
    addwf K,0
    movwf L
    goto inicio
end
```

Ingresar un dato de 8 bits al la dirección reservada a la variable K.

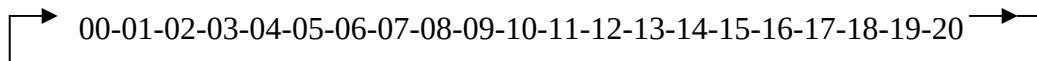
2.- Modificar el programa anterior, para que ahora los datos que operará se encuentren en las localidades reservadas para **J** y **K** respectivamente y el resultado almacenarlo en otras direcciones, reservadas para **C1** y **R1** donde C1 representará el valor de la bandera de acarreo y R1 el resultado.

3.- Realice un programa que ejecute la siguiente secuencia, misma que deberá ver en la dirección de memoria de su elección.

Secuencia:



4.- Desarrollar un programa que presente la cuenta en numeración decimal en la localidad de memoria de su elección, como se indica a continuación.



5.- Elaborar un programa que encuentre el número menor, de un conjunto de datos ubicados entre las localidades de memoria **20h** a **40h**; mostrar el valor en la dirección **41h**.

Laboratorio de Microcomputadoras
Práctica No. 2
Sistema mínimo microcontrolador PIC16F877

Objetivo. Conocer la estructura y características de la tarjeta que se dispone en el laboratorio, el software de comunicación, aplicaciones con puertos paralelos trabajando como salida y la ejecución de un programa en tiempo real.

Introducción

El microcontrolador PIC tiene 5 puertos paralelos, denominados A, B, C, D y E, todos ellos se pueden configurar para operar como puerto de salida o entrada.

Puerto	Tamaño	Función	TRISX	PORTX
A	6	E/S	85H	05H
B	8	E/S	86H	06H
C	8	E/S	87H	07H
D	8	E/S	88H	08H
E	3	E/S	89H	09H

Al emplear un puerto paralelo, lo primero que se debe de hacer es configurar su función, esto se realiza en las posiciones de memoria RAM denominados **TRISX** los cuales están ubicados en el banco número 1. Una vez ubicado en este banco se realiza la configuración, bajo la siguiente convención.

‘0’ Configura el bit del puerto como salida
‘1’ Configura el bit como entrada

Después que se ha configura todo el puerto, regresar al banco cero para enviar o recibir información a través de los registros de datos **PORTX**, a continuación se presenta las instrucciones que realizan lo anterior:

processor 16f877	;Indica la versión de procesador
include <p16f877.inc>	;Incluye la librería de la versión del procesador
org 0H	;Carga al vector de RESET la dirección de inicio
goto inicio	
inicio: org 05H	;Dirección de inicio del programa del usuario
BSF STATUS,RP0	;Cambia la banco 1
BCF STATUS,RP1	
MOVLW 0	;Configura al puerto B como salida
MOVWF TRISB	
BCF STATUS,RP0	;Regresa al banco cero
.....	
.....	
.....	
end	;Directiva de fin de programa

Programación del microcontrolador PIC.

Las tarjetas que se cuentan en el laboratorio han sido programadas previamente con el código denominado Bootloader, el cuál permite cargar los programas del usuario una vez ensamblados al microcontrolador, haciendo uso de la comunicación serie; otra opción es utilizar un programador externo, en este caso tener cuidado al extraer y colocar sus circuitos de su tarjeta, en manual solo se describirá el primer método.

Ejecutar el programa **PICDOWNLOADER**, se mostrará una pantalla como en la figura 2.1a, seleccionar el programa que se desea cargar, la velocidad deberá ser de **19200** y comprobar su puerto serie disponible, en la mayoría de los casos será **COM1**. Dar click en **WRITE** y comenzará a mostrar el proceso de cargado, cuando termine mostrará un despliegue como en la figura 2-1b.

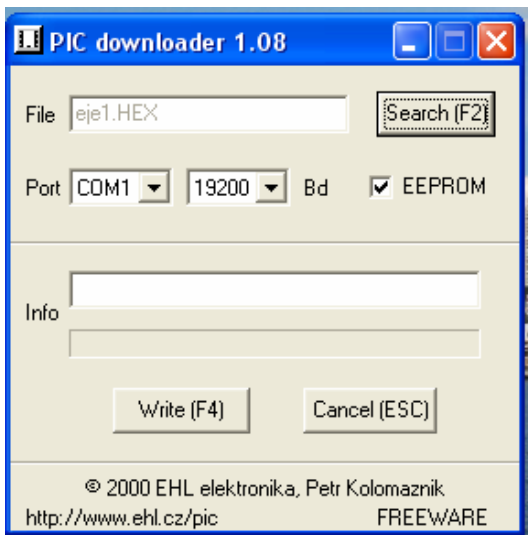


Figura 2.1a Picdownloader

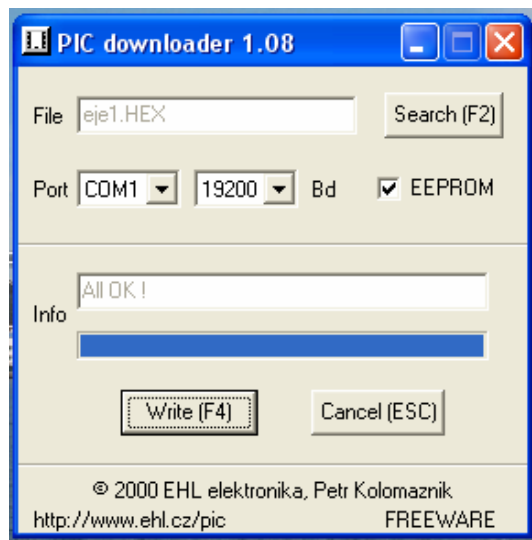


Figura 2.1b Programa cargado completamente

Desarrollo. Para cada uno de los siguientes ejercicios, realizar los programas solicitados y comprobar el funcionamiento de ellos.

1.- Escribir, comentar e indicar que hace el siguiente programa.

```
processor 16f877
include<p16f877.inc>

contador equ h'20'
valor1 equ h'21'
valor2 equ h'22'
valor3 equ h'23'
cte1 equ 20h
cte2 equ 50h
cte3 equ 60h

org 0
goto inicio

org 5
inicio bsf STATUS,5
      BCF STATUS,6
      MOVLW H'0'
      MOVWF TRISB
      BCF STATUS,5
      clrf PORTB

      loop2 bsf PORTB,0
            call retardo
            bcf PORTB,0
            call retardo
            goto loop2

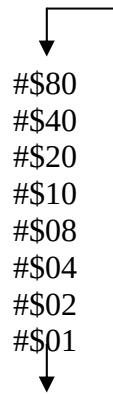
      retardo movlw cte1
            movwf valor1
      tres   movlw cte2
            movwf valor2
      dos    movlw cte3
            movwf valor3
      uno    decfsz valor3
            goto uno
            decfsz valor2
            goto dos
            decfsz valor1
            goto tres
            return
      END
```

2.- Ensamblar y cargar el programa anterior en memoria del microcontrolador.

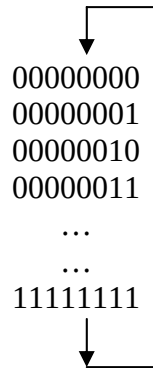
3.- Modificar el programa anterior, para que ahora se actualice el contenido de todos los bits del puerto B y se genere una rutina de retardo de un segundo.

4.- Realizar un programa que muestre la siguiente secuencia en el puerto B con retardos de ½ segundo.

Secuencia:



5.- Realizar un programa que muestre un contador binario por el puerto paralelo B, desde su valor mínimo B'00000000' hasta el máximo B'11111111' y se repita nuevamente el contador; usar retardos de ½ segundo.



Laboratorio de Microcomputadoras
Práctica No 3
Puertos Paralelos II (Control de acciones)

Objetivo. Emplear los puertos paralelos que contiene un microcontrolador para realizar funciones de control, configurando estos como entrada y salida.

Introducción

Cuando el microcontrolador PIC será configurado como entrada, se recomienda limpiar el contenido del registro de datos del puerto mediante la instrucción **CLRF PORTX**, esto con la finalidad de iniciar los latches de datos del puerto en cuestión, con esta instrucción se configurará al puerto de manera correcta.

Además de lo anterior, para el caso del puerto A y E se requiere indicar en el registro **ADCON1** ubicado en el banco 1 que se desea utilizar como E/S digitales, por lo que se escribirá un **06H** o **07H** en dicho registro, para posteriormente cargar el dato de configuración al registro **TRISA** o **TRISE**.

Ejemplo

processor 16f877	;Indica la versión de procesador
include <p16f877.inc>	;Incluye la librería de la versión del procesador
org 0H	;Carga al vector de RESET la dirección de inicio
goto inicio	
org 05H	;Dirección de inicio del programa del usuario
inicio: CLRF PORTA	
BSF STATUS,RP0	;Cambia la banco 1
BCF STATUS,RP1	
MOVLW 06H	;Configura puertos A y E como digitales
MOVWF ADCON1	
MOVLW 3FH	;Configura el puerto A como entrada
MOVWF TRISA	
BCF STATUS,RP0	;Regresa al banco cero
.....	
.....	
.....	
end	

Desarrollo. Para cada uno de los siguientes apartados, realizar los programas solicitados y comprobar el funcionamiento de ellos.

1.- Empleando dos puertos paralelos del microcontrolador PIC, uno de ellos configurado como entrada y el otro como salida; realizar un programa que de acuerdo al valor del bit menos significativo del puerto **A**, se genere la acción indicada en el puerto **B**.

Valor PA0	Acción puerto B
0	00000000
1	11111111

Tabla 3.1 Control de salidas controladas por un bit

2.- Realizar un programa, el cuál realice las siguientes acciones de control, para lo cuál requiere trabajar un puerto de entrada y otro puerto de salida, usar los sugeridos en el ejercicio anterior; generar retardos de ½ seg., en las secuencias que lo requieran.

DATO	ACCION	Ejecución
\$00	Todos los leds apagados	00000000
\$01	Todos los leds encendidos	11111111
\$02	Corrimiento del bit más significativo hacia la derecha	10000000 01000000 00100000 00000001
\$03	Corrimiento del bit menos significativo hacia la izquierda	00000001 00000010 00000100 10000000
\$04	Corrimiento del bit más significativo hacia la derecha y a la izquierda	10000000 01000000 00000001 00000010 10000000
\$05	Apagar y encender todos los bits.	00000000 11111111

Tabla 3.2 Control de salidas completo

Laboratorio de Microcomputadoras
Practica No. 4
Puertos Paralelos III (Control de Motores de CD)

Objetivo. Emplear los puertos paralelos que contiene un microcontrolador, para controlar la operación de dos motores de corriente directa.

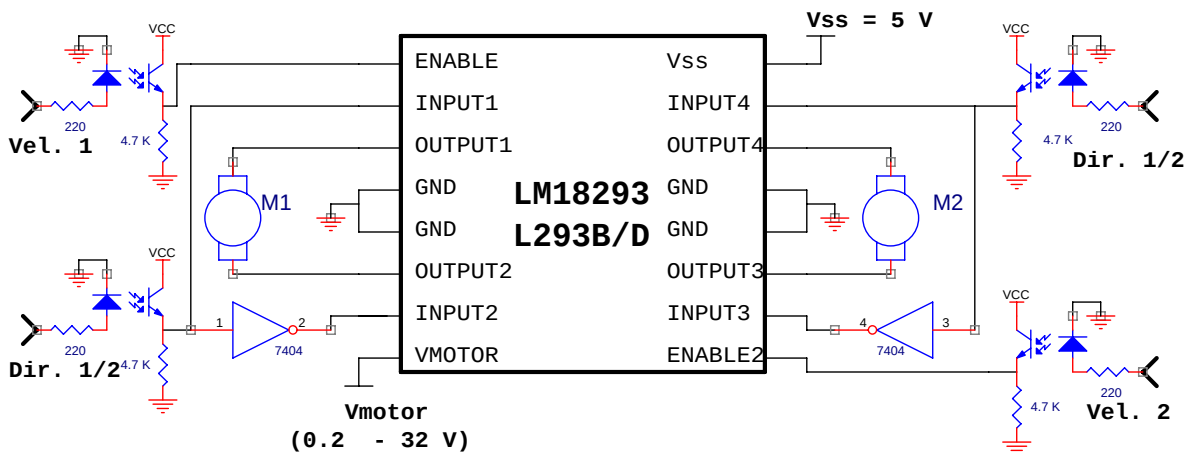
Introducción

El circuito que nos permite entregar la potencia y señales de control a motores de corriente directa es el L293 B/D o en caso de requerir mayor corriente usar el L298; se sugiere para una mayor información revisar la hoja de datos de este circuito.

El circuito tiene dos terminales para alimentación, una de ellas es para el propio dispositivo, el cuál debe ser de 5 volts, y otra para la tensión en los motores la cual puede ser desde 0.2 volts hasta 32 volts (de acuerdo al voltaje de operación del motor), así mismo permite tener el control de la velocidad de rotación del motor, mediante las terminales EN1 y EN2; por último la dirección de rotación se establece de acuerdo al nivel lógico existente entre las terminales identificadas como DIR1 y DIR2 para un motor, DIR3 y DIR4 para el otro motor.

Por ejemplo si EN=1, Dir1=1 y Dir2=0, el motor girará hacia un sentido y cuando DIR1=0 y DIR2=1, el motor girará en sentido contrario. El motor se mantiene parado cuando EN1=0 o el valor en Dir1= Dir2.

A manera de protección del microcontrolador, se recomienda contar con dos fuentes de alimentación independientes (una para el PIC y otra para el circuito L293B) y contar con una etapa de acoplamiento óptico para tener un mejor desempeño del sistema.



Circuito 4.1. Driver L293 empleando optoacopladores

Instrucciones para uso de los módulos del laboratorio:

- Se alimentará el driver de motores con voltaje superior a 8 volts, ya que cuenta con un regulador de voltaje LM7805.
- Un motor se controlará a través de las señales indicadas como Vel1 (EN1) y Dir ½, para que funcione el motor la señal Vel1 deberá estar en alto y Dir ½ podrá valer 0 o 1 lo cuál indicará el sentido de giro; de igual forma para el otro motor.
- Por lo tanto solo requiere de 2 señales de control para cada motor, es decir cuatro en total, que serán enviadas por el puerto paralelo seleccionado.
- De acuerdo a la tarjeta empleada debe identificar en que pines del puerto se encuentran asignadas estas señales (En1 En2 Dir ½ Dir ¾).
- En los sistemas con el driver integrada en la tarjeta de puertos están asignados en los cuatro bits menos significativos, mientras que en las otras tarjetas en los cuatro bits más significativos.

Desarrollo. Utilizando el circuito de potencia de motores de corriente directa y el sistema de desarrollo del microcontrolador PIC, realizar los programas solicitados.

1.- De acuerdo a la asignación de la tarjeta del driver de motores realizar un programa, el cual permita controlar el funcionamiento y sentido de giro de cada uno de ellos por separado, a través del puerto paralelo A, el puerto B deberá mandar las señales al driver, como se indica en la tabla 4.1.

Entrada binaria PuertoA	Motor		Sentido de giro Puerto B
	Izquierdo	Derecho	
000000	OFF	OFF	Paro
000010	OFF	ON	Horario
000100	OFF	ON	Antihorario
001000	ON	OFF	Horario
010000	ON	OFF	Antihorario

Tabla 4.1

2.- Considerando la información y los circuitos del ejercicio anterior, realizar un programa que de acuerdo a una señal de control ingresada por el puerto A, se genere la acción indicada en la tabla 4.2.

DATO Puerto A	ACCION	
	MOTOR M1	MOTOR M2
\$00	PARO	PARO
\$01	DERECHA	DERECHA
\$02	IZQUIERDA	IZQUIERDA
\$03	DERECHA	IZQUIERDA
\$04	IZQUIERDA	DERECHA

Tabla 4.2

Laboratorio de Microcomputadoras
Práctica No. 5
Puertos Paralelos IV (Lectura de sensores ópticos)

Objetivo: Emplear los puertos paralelos, para hacer lecturas de señales externas (sensores reflectivos) con el microcontrolador y realizar operaciones de acuerdo a los valores recibidos.

Introducción

Un sensor de efecto reflectivo consta de un led infra-rojo y de un fototransistor, es usado como un sistema de detección de objetos, lectura de encoders y en aplicaciones de robótica, entre otros. Para esta práctica se desea obtener el rebote que se produce cuando el led emite el haz infra-rojo y lo detecte el fototransistor, el cuál operará en las regiones de corte y saturación; esta señal se adecuará para obtener un nivel lógico, mediante un circuito comparador que opera de la siguiente manera:

$$\begin{aligned} \text{Si } V_{+}(V_{\text{ref}}) \text{ es mayor a } V_{-}(V_{\text{sensor}}) &\Rightarrow V_{\text{salida}} = 5 \text{ Volts} \\ \text{Si } V_{-}(V_{\text{sensor}}) \text{ es mayor a } V_{+}(V_{\text{ref}}) &\Rightarrow V_{\text{salida}} = 0 \text{ Volts} \end{aligned}$$

El circuito siguiente muestra un sistema de detección opto-reflectivo, para uso en robots seguidores de línea.

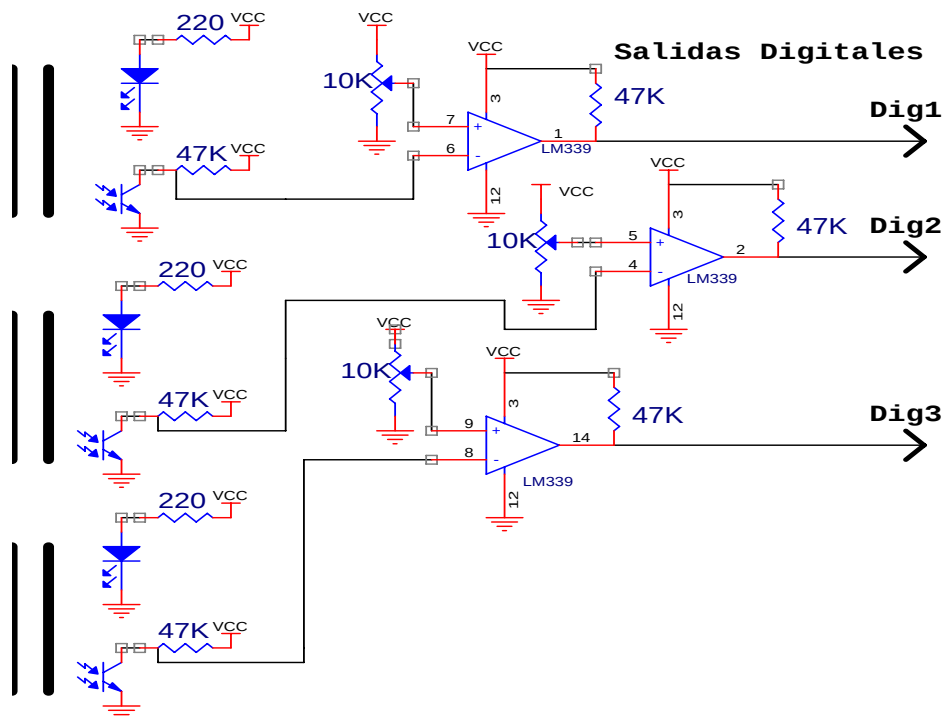


Figura 5.1 Sensores opto-reflectivos

Desarrollo: Realizar los apartados siguientes.

1.- Conectar la tarjeta de sensores reflectivos al puerto paralelo A y la tarjeta de leds al puerto paralelo B.

2.- Realizar un programa, de tal forma que indique cual sensor refleja la luz infla-roja mediante el equivalente despliegue al puerto B, representado mediante la siguiente tabla.

Tomar en cuenta que cuando es detectado el reflejo, el microcontrolador recibe un '1'.

ENTRADAS			SALIDAS			
Sensor Izquierdo PA2	Sensor Central PA1	Sensor Derecho PA0	PB3	PB2	PB1	PB0
N	N	N	0	0	0	0
N	N	B	0	0	0	1
N	B	N	0	0	1	0
N	B	B	0	0	1	1
B	N	N	0	1	0	0
B	N	B	0	1	0	1
B	B	N	0	1	1	0
B	B	B	0	1	1	1

Tabla 5.1

N (Línea negra) – '0'

B (Línea blanca) – '1'

3. Realizar un programa que de acuerdo a la entrada generada por los sensores, se controle la operación de los motores, tal como se muestra en la siguiente tabla.

ENTRADAS			ACCION	
Sensor Izquierdo	Sensor Central	Sensor Derecho	MOTOR IZQUIERDO	MOTOR DERECHO
B	N	N	ATRÁS	ADELANTE
N	B	N	ADELANTE	ADELANTE
N	N	B	ADELANTE	ATRÁS
N	N	N	PARO	PARO

Tabla 5.2

Nota. Considerar que los sensores están conectados en la parte baja del puerto A y los bits restantes no tienen un nivel lógico definido, así mismo tomar en cuenta la asignación de los motores que depende de la tarjeta del microcontrolador empleada.

Laboratorio de Microcomputadoras
Práctica No. 6
Convertidor Analógico/Digital

Objetivo. Familiarizar al alumno con el uso y aplicación del Convertidor Analógico/Digital de un microcontrolador.

Introducción

El microcontrolador PIC16F877 tiene 8 posibles canales de entrada por los cuales se pueden procesar señales analógicas, de 8 o 10 bits de resolución.

Los registros involucrados para este periférico son los mostrados a continuación, la dirección y banco donde están ubicados se pueden consultar en la información dada en la práctica uno.

ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	-	ADON
---------------	-------	-------	------	------	------	---------	---	------

ADCS1:ADCS0 Seleccionan la frecuencia de reloj
CHS2-0 Selección del canal de entrada
GO/DONE Si GO/DONE=1; inicia el proceso de conversión
 Si GO/DONE=0; terminó la conversión
ADON Enciende al convertidor A/D

ADCON1	ADFM	-	-	-	PCFG3	PCFG2	PCFG1	PCFG0
---------------	------	---	---	---	-------	-------	-------	-------

ADFM El resultado de la conversión se almacena en los registros ADRESH:ADRESL, formando un dato de 10 bits, pudiendo ser reflejado de la siguiente manera:

Si ADFM=1; el resultado es justificado en el registro ADRESH (los seis bits más significativos de este registro valen cero).

Si ADFM=0; el resultado es justificado en el registro ADRESL (Los seis bits menos significativos de este registro valen cero)

PCFG3-0 Configura a los puertos paralelos A y E como entradas al convertidor A/D; en el caso de utilizar ambos puertos como entradas analógicas, se configuran estas banderas en cero.

ADRESH							
---------------	--	--	--	--	--	--	--

Parte alta del resultado de la conversión

ADRESL							
---------------	--	--	--	--	--	--	--

Parte baja del resultado de la conversión

Un algoritmo a emplear para el uso del convertidor A/D, con resolución de 8 bits:

1. Estando en el banco cero, limpiar el puerto A, usando CLRF PORTA.
2. Cambiar al banco uno.
3. Configurar el puerto A como entradas analógicas, escribir 00H al registro ADCON1.
4. Realizar la configuración de la fuente de reloj, el canal de entrada y prender al convertidor A/D, en el registro ADCON0.
5. Iniciar la conversión colocando un '1' a la bandera GO/DONE#.
6. Generar un tiempo de retardo de 20 microsegundos.
7. Esperar a que GO/DONE# sea igual a cero, lo que indica que ha concluido el proceso de conversión.
8. Lee el resultado de la conversión del registro ADRESH.

Desarrollo. Realizar los programa solicitados y comprobar su funcionamiento.

1.- Empleando el canal de su elección del convertido A/D, realizar un programa en el cuál, de acuerdo a una entrada analógica que se ingrese por este canal, se represente el resultado de la conversión en un puerto paralelo utilizar el arreglo de leds para ver la salida, como se muestra en la figura 6.1.

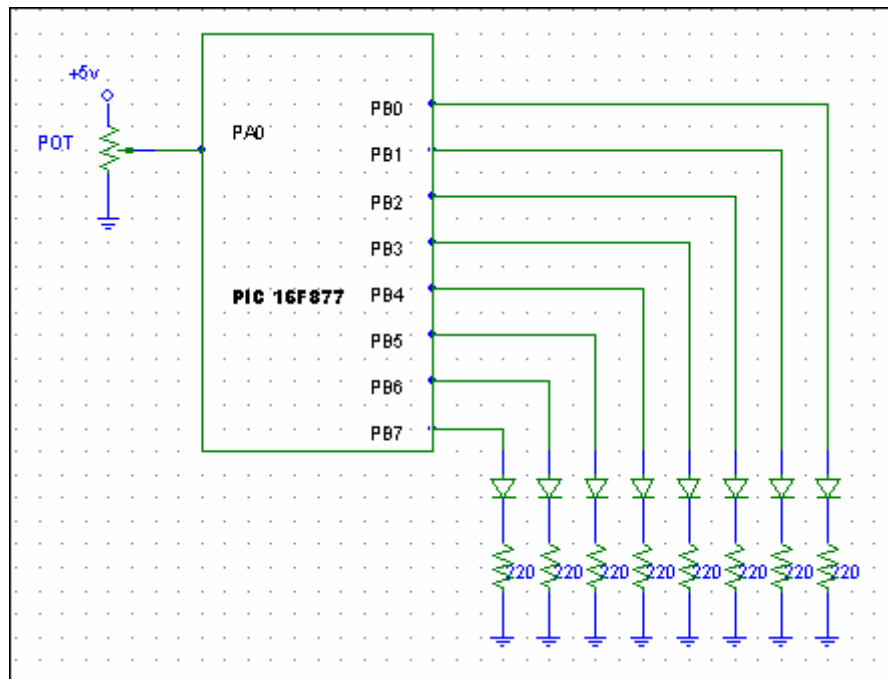


Figura 6.1 Circuito con lectura de una señal analógica

2.- Utilizando el circuito anterior, realizar un programa que indique si el valor del voltaje a la entrada del convertidor A/D, se encuentra entre los siguientes rangos de voltaje.

ENTRADAS	SALIDAS		
	PX2	PX1	PX0
$V_e < 1/3 V_{cc}$	0	0	1
$1/3 V_{cc} < V_e < 2/3 V_{cc}$	0	1	1
$2/3 < V_e < V_{cc}$	1	1	1

Tabla 6.1
Donde $V_{cc} = 5$ volts

3.- Realizar un programa, de manera que identifique cuál de tres señales analógicas que ingresan al convertidor A/D es mayor que las otras dos; representar el resultado de acuerdo al contenido de la tabla 6.2.

Señal	PB2	PB1	PB0
$V_{e1} > V_{e2}$ y V_{e3}	0	0	1
$V_{e2} > V_{e1}$ y V_{e3}	0	1	1
$V_{e3} > V_{e1}$ y V_{e2}	1	1	1

Tabla 6.2

Circuito empleado para este ejercicio.

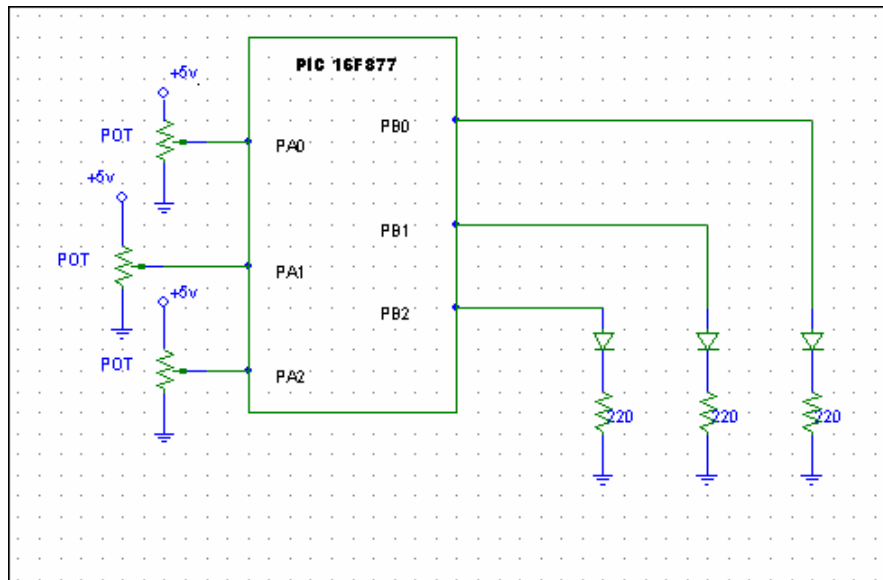


Figura 6.2 Tres señales analógicas

Laboratorio de Microcomputadoras
Práctica No. 7
Puerto Serie SCI (Asíncrono)

Objetivo. Familiarizar al alumno en el uso de una Interfaz de Comunicación Serie Asíncrona de un microcontrolador.

Introducción

El microcontrolador PIC16F877 contiene un módulo USART, el cuál permite la comunicación de tipo Asíncrona, con el uso de los pines RC6 y RC7 del puerto C, la velocidad de comunicación se configura por software, dentro de una gama amplia de valores, además de contar con banderas que indican la terminación, ya sea de la transmisión o la recepción de datos.

Registros ocupados en la comunicación serie:

Registro usado en el módulo generador de Baud Rate

SPBRG

--	--	--	--	--	--	--	--

Con este registro se configura la velocidad de comunicación utilizando una expresión matemática para encontrar un valor de 8 bits que será cargado en el, la velocidad y fórmula dependerá de valor que sea cargado la bandera BRGH del registro TXSTA

Registro usado en el módulo transmisor

TXSTA

CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D
------	-----	------	------	---	------	------	------

Donde:

CSRC	Bit de selección del reloj, aplicable solo en modo de comunicación síncrona
TX9	Habilita el 9° bit de transmisión
TXEN	Activa la transmisión
SYNC	Selección del modo de comunicación a emplear SYNC=0 Comunicación asíncrona SYNC=1 Comunicación síncrona
BRGH	Bit de selección de baudios BRGH=0 Baja velocidad BRGH=1 Alta velocidad
TRMT	Estado del registro de corrimiento de transmisión, indica que se ha transmitido el dato si esta bandera es igual a uno.
TX9D	9° bit de datos a transmitir

Registro del módulo receptor

RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
--------------	------	-----	------	------	-------	------	------	------

Donde:

SPEN		Habilita el puerto serie SPEN=0 Deshabilitado SPEN=1 Habilitado
RX9		Habilita el 9º bit de recepción
SREN		Configura la recepción sencilla, aplicable solo para comunicación síncrona
CREN		Configura la recepción continua en modo de comunicación asíncrona
ADDEN, FERR, OERR		Indicadores de posibles errores en la recepción de datos
RX9D		9º bit de dato

Registro de banderas

PIR1			RCIF	TXIE				
-------------	--	--	------	------	--	--	--	--

RCIF		Bandera de recepción completa RCIF=0 Recepción en proceso RCIF=1 Recepción completa; indica que es posible leer el registro de recepción RCREG
TXIF		Bandera de transmisión completa TXIF=0 Recepción en proceso TXIF=1 Recepción completa; indica que es posible escribir otro dato al registro TXREG

Registro de transmisión

TXREG								
--------------	--	--	--	--	--	--	--	--

Registro de recepción

RCREG								
--------------	--	--	--	--	--	--	--	--

Algoritmo de empleo del módulo USAR en la modalidad Asíncrona utilizando transmisor y receptor en el mismo programa.

1. Cambiar al banco uno
2. Configura la bandera BRGH para seleccionar alta o baja velocidad
3. Cargar el valor correspondiente a la velocidad requerida (consultar los valores del data sheet)
4. Configurar el modo asíncrono SYNC=0 del registro TXSTA
5. Habilita la transmisión TXEN=1 del registro TXSTA
6. Regresar al banco cero
7. Habilita la recepción de datos CREN=1 del registro RCSTA
8. Habilita el puerto serie SPEN del registro RCSTA
9. Realizar la operación deseada por el programa
 - a. Transmisión: Escribir el dato al registro TXREG y esperar a la transmisión del mismo, esperar a que TRMT=1 en el registro TXSTA (considerar que este registro está ubicado en el banco uno)
 - b. Recepción: Esperar hasta que la bandera RCIF del registro PIR=1, indicador de recepción completa (tomar en cuenta que este registro esta ubicado en el banco cero)

Desarrollo. Realizar los siguientes apartados:

- 1.- Utilizando el programa resuelto en la practica No. 3, ejercicio 2 (Control de acciones), realizar las modificaciones necesarias para que ahora se controle por medio del teclado de su PC, el cuál transmitirá el comando de la acción a ejecutar.

Abrir una Terminal, usando la Hyper Terminal que contiene Windows o la Terminal incluida en el IDE PIC C Compiler, consultar los apéndices A y B.

TECLA	ACCION Puerto B
0	Todos los bits del puerto apagados
1	Todos los bits del puerto encendidos
2	Corrimiento del bit más significativo hacia la derecha
3	Corrimiento del bit menos significativo hacia la izquierda
4	Corrimiento del bit más significativo hacia la derecha y a la izquierda
5	Apagar y encender todos los bits.

Tabla 7.1 Controla a través del puerto serie

- 2.- Realizar un programa que muestre las vocales (mayúsculas y minúsculas en un display de 7 segmentos, las cuales serán enviadas vía serie a través del teclado de la PC.

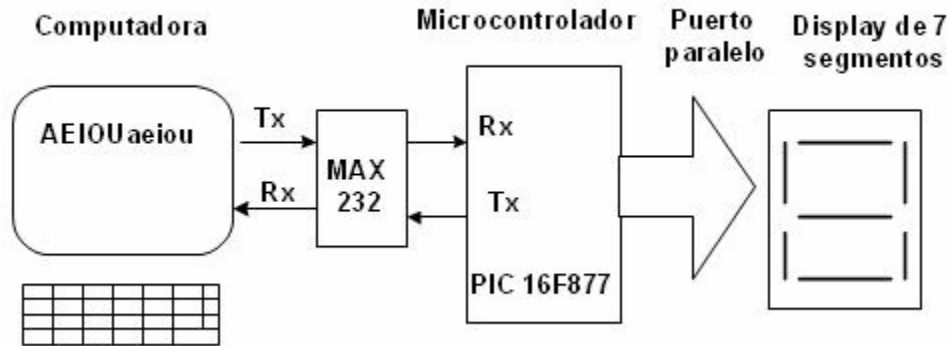


Figura 7.1 Control de despliegue de vocales

- 3.- Empleando el programa No. 3 de la práctica 6 (convertidor analógico digital), realizar las modificaciones necesarias para desplegar el número de canal, de valor mayor a las otras entradas, en el monitor de la PC.

Laboratorio de Microcomputadoras
Práctica No. 8
Puerto serie y programación en C

Objetivo. Realización de programas a través de programación en C y empleo del puerto serie para visualización y control.

Introducción

Consultar la información contenida en el apéndice C, D y la ayuda del compilador.

Desarrollo. Realizar los siguientes ejercicios.

1.- Escribir, comentar, compilar y ejecutar el siguiente programa usando el ambiente del PIC C Compiler.

```
#include <16f877.h>
#fuses HS,NOPROTECT,
#use delay(clock=20000000)
#org 0x1F00, 0x1FFF void loader16F877(void) {}

void main(){
while(1){
    output_b(0x01);
    delay_ms(1000);
    output_b(0x00);
    delay_ms(1000);
} //while
} //main
```

2.- Modificar el programa para que active y desactive todos los bits del puerto B.

3.- Escribir, comentar, compilar y ejecutar el siguiente programa usando el ambiente del PIC C Compiler.

```
#include <16f877.h>
#fuses HS,NOPROTECT,
#use delay(clock=20000000)
#org 0x1F00, 0x1FFF void loader16F877(void) {} //for the 8k 16F876/7

int var1;
void main(){
while(1){
    var1=input_a();
    output_b(var1);
} //while
} //main
```

4.- Escribir, comentar, compilar y ejecutar el siguiente programa usando el ambiente del PIC C Compiler.

```
#include <16f877.h>
#fuses HS,NOPROTECT,
#use delay(clock=20000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)
#org 0x1F00, 0x1FFF void loader16F877(void) {} //for the 8k 16F876/7

void main(){
while(1){
    output_b(0xff); //
    printf(" Todos los bits encendidos \n\r");
    delay_ms(1000);
    output_b(0x00);
    printf(" Todos los leds apagados \n\r");
    delay_ms(1000);
} //while
} //main
```

Para comprobar el funcionamiento de este programa debe cerrar el Picdownloader y abrir una Terminal de comunicación, tal como se explicó en la práctica 7.

5.- Realizar las modificaciones necesarias al ejercicio 2 de la práctica tres para que ahora el comando que selecciona la acción sea a través del puerto serie, usar retardos de ½ segundos, usando programación en C.

DATO	ACCION Puerto B	Ejecución
0	Todos los bits apagados	00000000
1	Todos los bits encendidos	11111111
2	Corrimiento del bit más significativo hacia la derecha	10000000 00000001
3	Corrimiento del bit menos significativo hacia la izquierda	00000001 10000000
4	Corrimiento del bit más significativo hacia la derecha y a la izquierda	10000000 00000001 10000000
5	Apagar y encender todos los bits.	00000000 11111111

Tabla 8.2 Control a través del puerto serie

Laboratorio de Microcomputadoras
Práctica No. 9
Programación C, Convertidor A/D e Interrupciones

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

Introducción

Funciones recomendadas para esta practica:

- Convertidor analógico digital
 - o #device ADC=8
 - o setup_port_a(ALL_ANALOG);
 - o setup_adc(ADC_CLOCK_INTERNAL);
 - o set_adc_channel(num);
 - o delay_us(20);
 - o read_adc();
- Interrupciones
 - o enable_interrupts(fuente);
 - o enable_interrupts(GLOBAL)
 - o Colocar la rutina de atención a la interrupción antes del main
 - o Fuentes de interrupción usadas en esta práctica
 - #INT_RB ; cambio de nivel del los cuatro bits más significativos del puerto B
 - #INT_RTCC; Sobreflujo del Timer0
- Funciones para uso del TIMER0
 - o set_timer0(0) ;inicia el timer0 en 00H
 - o setup_counters(RTCC_INTERNAL,RTCC_DIV_256); configura la razón de tiempo en la que TOIF se prenderá; en este caso será $t = T_{\text{ciclo}}$ de reloj(255)(256), donde 256 es el pre-escalador, $T_{\text{ciclo}} = 4/XTAL$ y 255 l cantidad de pulsos requeridos para que a partir del valor cargado al timer0, se produzca un sobreflujo.
 - o enable_interrupts(INT_RTCC); habilita la interrupción TIMER0

Plantilla de la rutina de interrupción del timer 0

```
#int_rtcc
clock_isr(){
    //código de la rutina
}
```

- Funciones para uso de PB4 al PB7
 - Enable_interrupts(INT_RB): habilita interrupción por cambio de nivel en los cuatro bits más significativos del puerto B

Plantilla de interrupción de puerto B (PB4-PB7)

```
#int_rb
port_rb(){
    //código de la rutina de interrupción
}
```

Finalmente el programa que empleará las interrupciones tendrá la siguiente forma:

```
#include <16f877.h>
#device adc=8 //en caso de emplear el conv. A/D indica resolución de 8 bits
...          //configuración general
...
          //declaración de variables
#int_rtcc    // rutina de interrupción del timer0
clock_isr(){
    //código de la rutina de interrupción
}

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 interrupción por TIMER0
    enable_interrupts(GLOBAL); //Habilita interrupciones generales

          // programa principal

}
```

Desarrollo. Realizar los siguientes ejercicios.

- 1.- Programa el cual obtenga una señal analógica a través del canal de su elección, se realice la conversión y el resultado de esta, la muestre en un puerto paralelo y a su vez lo transmita al puerto serie.
- 2.- Utilizando la interrupción del TIMER0, realizar un programa que transmita el resultado de la conversión cada 10 segundos.
- 3.- Realizar un programa el cual constantemente transmita el resultado de la conversión a la terminal, y cada 30 segundos interrumpa la ejecución de este y envíe el siguiente texto “Laboratorio de Microcomputadoras”
- 4.- Utilizando la interrupción por cambio de nivel del puerto paralelo, realizar un programa que reconozca un flanco positivo en los pines PB4, PB5, PB6 o PB7 del puerto B, y cuando se presente, envíe a la terminal el siguiente texto; de acuerdo a la entrada en la que ha ocurrido el evento.

Interrupción PB4 Activada
Interrupción PB5 Activada
Interrupción PB6 Activada
Interrupción PB7 Activada

Cuando se detecte la transición de alto a bajo, se debe mostrar:

Pulso de bajada