****

**Escuela:** Centro de enseñanza técnica industrial – Colomos  
**Carrera:** Desarrollo de software  
**Materia:** Sistemas embebidos.  
**Tema:** Práctica 1 Parcial 1, Menú Básico. **Nombre del alumno:** Carlos Daniel Lozano Vázquez **Registro:** 18300249. **Grupo y grado:** 6ºB1. **Fecha:** 28/02/2021

**2.-Objetivo de la práctica**: Implementar en un microcontrolador RISC un programa en lenguaje C para un menú básico de 6 opciones que permita demostrar el uso de los puertos de entrada y salida.

Microcontrolador Pic16F877A

**3.-Descripción de la práctica:**

Conectar al microcontrolador un dip switch para que funcione como entrada de datos en el puerto que elijas y conectar en otro, 8 leds para que funcionen como dispositivos de salida.

Realizar un programa en lenguaje C para el Pic 16F877A para implementar un menú de 6 opciones como se indica:

Pedir un número op, para indicar la opción:

Si Op = 0: Mostrar en P1un rotabit a la izquierda o a la derecha (tu elige, y decide si se repite 8 posiciones o de forma cíclica).

OP= 1: Pedir por Pentrada, tres números x, y, z (de 6 u 8 bits) y mostrar en Psalida el promedio de ellos.

OP= 2: Pedir por Pentrada, tres números x, y, z (de 6 u 8 bits) y mostrar en Psalida el Mayor de ellos.

Op= 3: Mostrar por Psalida los números pares de 0 a 90 H. (utilizar retardo para mostrar los números).

Op= 4: Pedir dos números x,y ( de 6 u8 bits, ) mostrar en Psalida los números de x a y. (utilizar retardo para mostrar los números).

Op= 5: Pedir dos números x, y (de 4 bits), mostrar en P1 el resultado de la ecuación x2-2y.

**4.-Resumen.**

Se trata de un programa hecho con la IDE de CCS la cual utiliza lenguaje C para programar ciertos circuitos integrados, más específicamente microcontroladores de la familia PIC.  
En este programa se hace uso de las diversas funciones que tiene dicho microntrolador, además de que se usan sus termianales para el ingreso y salida de datos, al igual que otro pin como banderas para el programa.

El circuito usado es el PIC-16F877, el cual tiene 4 puertos de entrada y salida de datos, los cuales son:  
PA, PB, PC, PD.

**Funcionamiento de entrada o salida:**

Para que sirva como salida hay que poner un 0 y 1 para entrada, los primeros 6 bits son los menos significativos y en el **TRIS-A** hay que ponerle la cantidad de bits que queremos activar, si queremos usar 8bits = FFH, para que funcione la mitad hay que poner 11110000, **PORT-A** sirve para mandar un dato de salida, es decir el dato que quieres que salga por el puerto A es el dato que está en PORTA, los primeros 6 bits del puerto A funcionan, los otros 2 no sirven.

Para usar banderas es recomendable usar uno de los 96 registros que tenemos.

Para trabajar con el programa se trabaja con el PORT, es decir para mostrar los datos, ya sea por el PORTB o A.

**Carry**

Carry Bit: 0

Trabaja con 8 bits

1 si hay acarreo en el nivel alto es decir en los 4 bits más significativos, es cuando ya no cabe el resultado (se pasa de 8 bits), por lo que hay desbordamiento

0 si no hay desbordamiento

Carry digital bit: 1

Hay carry cuando en el nivel bajo hay carry es decir en los 4 bits menos significativos

1: si hay desboirdamiento

0: si no hay desbordamiento

Bandera de 0 es la del carry "2".

**Registro RP**

Bits 6 y 5, son los RP0 y RP1, por lo cual hay 7 bancos de memoria del 00h-1FFh

00 Banco 0: 00H-7FH

01 Banco 1: 80H-FFH

10 Banco 2: 100H-17FH

11 Banco 3: 180H-1FFH

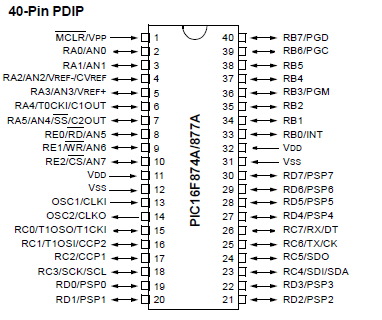
La instrucción BSF pone en set 1 bit, es decir en 1.

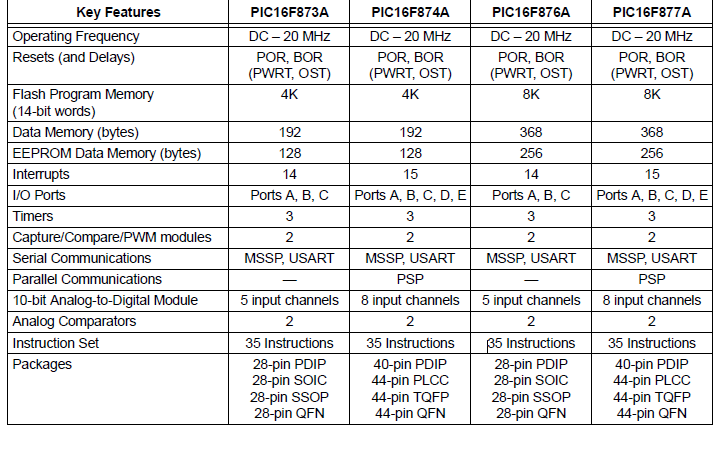
Puerto para activar el Puerto A como analógico o digital:

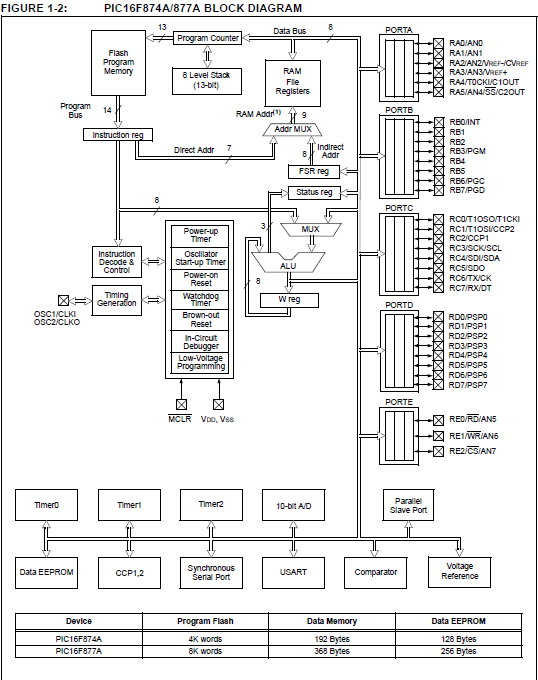
ADDCON1

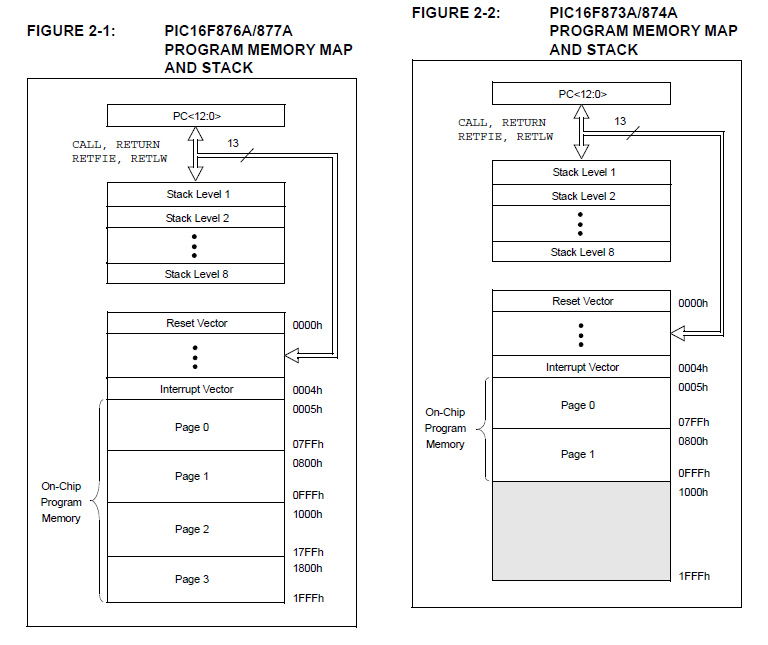
L/literal es cómo # y w es el registro de work/trabajo es como si fuera el acumulador

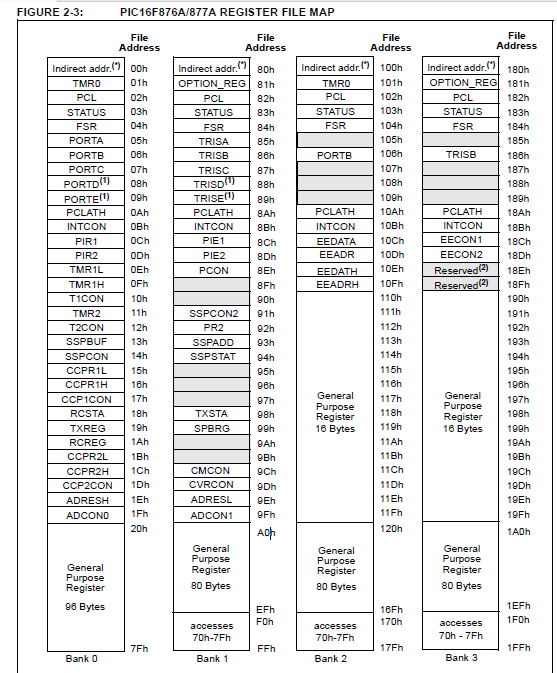
f es para direccionamiento



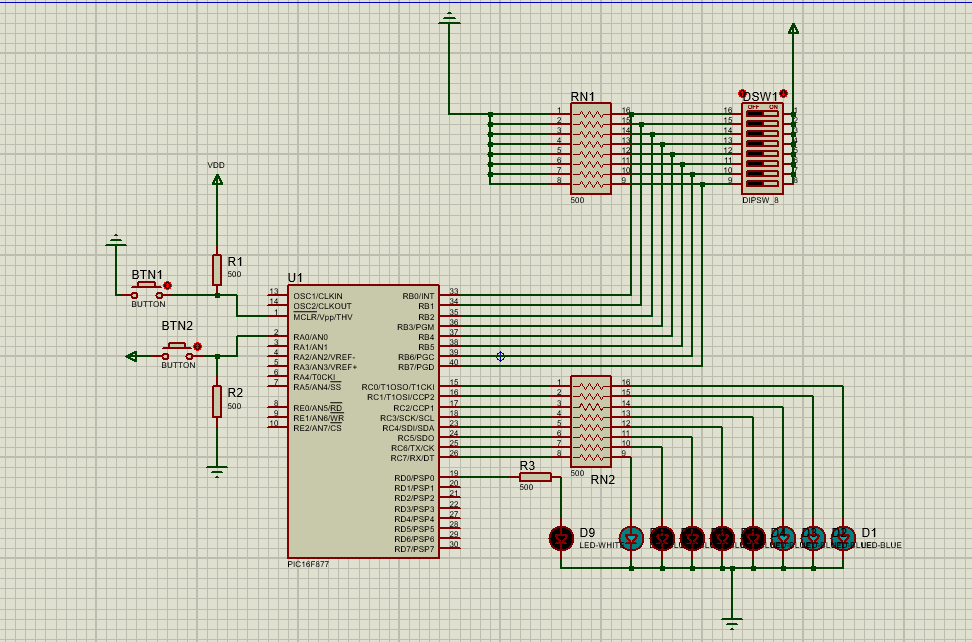








**5.-Diagrama a bloques.**



**6.- Diagrama del circuito eléctrico (fotos en físico).**

Es simulada

**7.-Programa.**

#include <16F877.h>

#fuses NOWDT, XT

#use delay(clock = 4M)

#byte TRISA = 0x85

#byte PORTA = 0x05

#byte TRISB = 0x86

#byte PORTB = 0x06

#byte TRISC = 0x87

#byte PORTC = 0x07

#byte TRISD = 0x88

#byte PORTD = 0x08

#define button PIN\_A0

#use fast\_io(A)

#use fast\_io(B)

#use fast\_io(C)

#use fast\_io(D)

void ctrlRetra()//bandera para que funcione el programa

{

while (!input(button)) {}

delay\_ms(**500**);

}

void show1(unsigned int Value)

{

PORTC = Value;//muestra el valor pasado

delay\_ms(**500**);

}

void enter(unsigned int& Num)

{

while (!input(button))//bandera para poder ingresar numeros

{

Num = PORTC = PORTB;

}

delay\_ms(**500**);

}

void menu(unsigned int OPC)

{

unsigned int Promedio = **0**, X1 = **0**, Y1 = **0**, Z1 = **0**;

switch (OPC)

{

case **0**:

byte X2 = **0x05**;

show1(X2);

for (unsigned int cont = **0**; cont <= **15**; cont++)

{

rotate\_right(&X2, **1**);

show1(X2);//funcion para mostrar valor en PuertoC

}

break;

case **1**:

enter(X1);//funcion input

enter(Y1);

enter(Z1);

Promedio = (X1 + Y1 + Z1) / **3**;

show1(Promedio);

break;

case **2**:

enter(X1);

enter(Y1);

enter(Z1);

show1(0);

if (X1 >= Y1 && X1 >= Z1)

Show1(X1);

else

if (Y1 >= Z1)

show1(Y1);

else

show1(Z1);

break;

case **3**:

byte Y2 = **0**;//creacion de un byte para una secuencia más sencilla

while (Y2 <= **0x90**)

{

PORTC = Y2;

delay\_ms(**300**);

Y2 += **0x02**;//se incrementa en 2 para solo mostrar pares

}

break;

case 4:

enter(X1);

enter(Y1);

if (X1 == Y1)//son iguales innecesario seguir

{

show1(X1);

break;

}

if (X1 > Y1)//se analiza cual es mayor para asi saber que secuencia tendra

{

while (X1 >= Y1)

{

show1(X1);//ya que x es mayor se requiere decrementar

X1--;

}

}

else

{

while (X1 <= Y1)//como x es menor se recquiere incrementar

{

show1(X1);

X1++;

}

}

break;

case 5:

enter(X1);

enter(Y1);

int result = **0**;

int Va1 = X1 & **0x00001111**;//mascara de 4 bits

int Va2 = Y1 & **0x00001111**;

result = (Va1 \* Va1) - (Va2 \* **2**);

int8 value = read\_eeprom(**0X000003**);//se analiza el registro 3

if (bit\_test(value, **1**))//se analiza el borrow de 4 bits

bit\_set(PORTD, **0**);// signo - activado

show1(result);

break;

default:

show1(0xFF);//FF si no corresponde a algun caso anterior

break;

}

}

void main()

{

unsigned int Opc = **0**;

TRISA = TRISB = **0xFF**;

TRISC = TRISD = **0**;

PORTB = PORTA = PORTD = **0**;

while (!input(button))//ciclo para indicar al usuario el inciio del programa

{

delay\_ms(**500**);

PORTC = **0xFF**;

delay\_ms(**500**);

PORTC = **0x00**;

}

delay\_ms(**500**);

enter(Opc); //se ingresa opcion de menu

menu(Opc);//llama a la funcion menu.

}

**8.-Explicacion.**

El programa sirve de la siguiente manera  
Al inicializarse entra en un bucle para así poder saber por cual puerto saldrán los datos, se sale del ciclo presionando el push button del pin A0 del puerto A, esto para cualquier ciclo para ingresar datos.  
Lo siguiente es que ingresa a una función en la cual se encuentra un Switch case para así poder ejecutar el fragmento de código para hacer la opción que selecciono el usuario; dependiendo de la opción es lo que se hará y de las funciones que se utilizarán.  
Se tiene 2 funciones que se usan demasiado en el código, las cuales son enter y show1, enter sirve para ingresar datos y show1 para mostrar los resultados.

El delay\_ms se usa para evitar efecto rebote y para que se puedan apreciar los datos mostrados en los leds por un determinado tiempo.

**9.-Observaciones.**

Se me complicó el aspecto de elaborar las funciones en CCS ya que sólo permite funciones e 1 parámetro y al querer usar más de 1 me marcas error, por lo que tuve que adaptar demasiado mi código a las limitantes de la IDE.

**10.-Conclusiones.**

Es interesante ver cómo se puede programar un microcontrolador de manera tan sencilla por medio de una IDE como lo es CCS o Arduino.