

Basic para Pics

Ing. Wilfrido González Bonilla
www.electronicaestudio.com

Muchos aficionados a la electrónica aun no se animan a aprender a manejar los microcontroladores PIC debido a la creencia de que su lenguaje es muy complejo y difícil de aprender. Como hemos visto en los artículos anteriores esto es falso. El “Ensamblador” de los PIC es muy sencillo y el numero de instrucciones que el usuario necesita dominar se limita a unas cuantas decenas.

También se cree que los programas para los PIC son muy largos y que puede tomar muchos días o semanas el resolver una aplicación. Creo que esto no es necesariamente verdad. Como vimos en el **Secuenciador de Luces** de un artículo anterior, la aplicación se resuelve muy rápidamente con el uso de MACROS.

Bueno, quizás nunca podamos convencer a algunos de nuestros amigos lectores a estudiar con nosotros los programas en Ensamblador. Pero el mundo de los PIC no termina ahí, justamente lo contrario. El ensamblador es solo el principio porque los PIC se pueden programar en otros lenguajes: BASIC y C principalmente.

¿Cómo puede ser esto posible?

Existen compiladores “especiales” en los que se puede editar en el lenguaje Basic o C y obtener un file “hex” listo para que su grabador de PICs lo utilice.

Cuidado, algunos principiantes se confunden y piensan que con Visual Basic o que con Turbo C van a poder programa un PIC. No, lo que se necesita es un programa especial.

El lenguaje que quizás es el consentido de los programadores es el Basic y en él nos concentraremos para que más adelante en otros artículos toquemos el C.

Existen muchos proveedores de software especializados en ofrecer compiladores Basic para programar Pics. Algunos de los más conocidos los podemos encontrar en las siguientes páginas de Internet:

<http://www.letbasic.com/>

<http://www.melabs.com/>

<http://www.basicmicro.com/>

Algunos de ellos ofrecen versiones gratis o “Demos”. Otros ofrecen compiladores muy completos pero también mas caros.

PicBasic Pro de Micro Engineering Labs Inc. (<http://www.melabs.com/>) es uno de los más conocidos. Este poderoso compilador pone al alcance del usuario potentes instrucciones para comunicación serie, matemática de 16 bits, mediciones de sensores analógicos, PWM, sonido, y muchísimas más.

Además de general los files “hex” y también es capaz de generar los files “asm”. De tal manera que sí se pueden hacer modificaciones de bajo nivel.

Otra magnífica carteristica de este compilador es que además de soportar al PIC16F84 también soporta a muchos otros de la gran familia de MICROCHIP. Por ejemplo los micros Flash PIC16F628, 16F876 y el 16F877.

Mencionemos algunas instrucciones

If . . Then

Salto condicional a otra instrucción.

For . . Next

Repite varias veces una serie de instrucciones.

Gosub

Llamada a una subrutina

High

Enciende un pin

Low

Apaga un pin

Serin

Entrada serie asíncrona en un pin. (RS232)

Serout

Salida serie asincrona en un pin. (RS232)

Adcin

Lee el convertidor analógico digital

Write

Escribe en la memoria EEPROM del PIC

Hpwm

Salida para generar modulación en anchura de pulsos

Estas son solo algunas de las instrucciones que se encuentran disponibles en este compilador. Existen otras que nos permiten generar instrucciones para pantallas de cristal liquido LCD, o para programar memorias EEPROM.

Con la tarjeta Entrenadora del PIC16F84 (Clave 502) se entrega un disco en el que se incluye una seccion con programas en PBP (PICBasic Pro). A continuación vamos a ver algunos ejemplos.

1.- Editar

Para Editar los programas en PBP se requiere de un editor de texto ASCII. Una buena selección es el mismo paquete de MPLAB.

Cuando se edita un programa en Ensamblador el file que se genera tiene la extensión "asm" como ya lo hemos visto. En este caso como el programa que estamos editando lo haremos para Basic, la extensión será "bas".

MPLAB no genera automáticamente esta extensión, por esta razón una vez que hemos tecleado el programa lo debemos salvar especificando la extensión "bas".

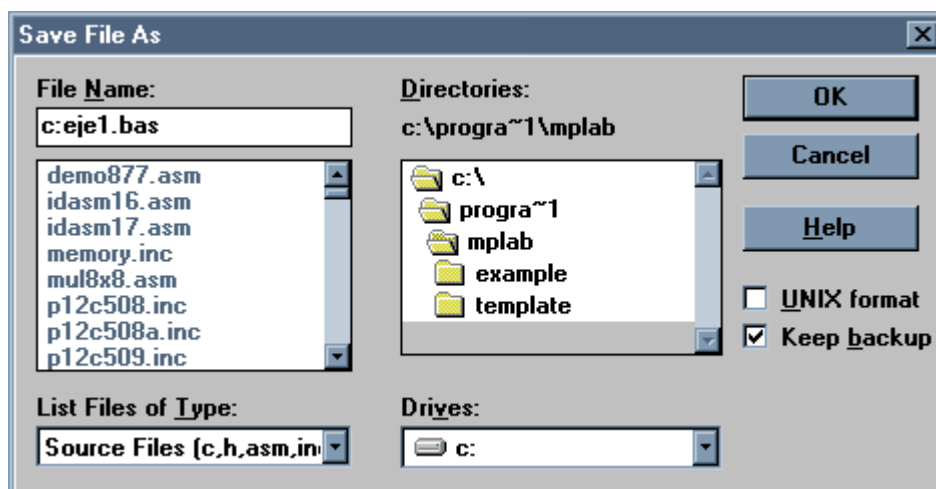


Fig. 1

Veamos un ejemplo sencillo eje1.bas

```
'Para Test1
'***Usar compilador PBP***
'Enciende S1 durante un segundo

'Declaracion de variables

        S0   VAR   PortB.0
        S1   VAR   PortB.1
        S2   VAR   PortB.2
        S3   VAR   PortB.3
        S4   VAR   PortB.4
        S5   VAR   PortB.5
        S6   VAR   PortB.6
        S7   VAR   PortB.7

'Define todos los bits del Puerto B como salidas

        TrisB = %00000000

Inicio:

        PortB=0

        S1=1
        Pause 1000
        S1=0

        End
```

El objetivo de este programa es que al energizar la tarjeta se encienda el bit 0 del Puerto B durante un segundo.

En primer lugar debemos observar que los comentarios ya no usan el ; (punto y coma) sino una sola comilla '. En los teclados en Español se encuentra justamente debajo del signo de interrogación. Al igual que en Ensamblador el comentario puede iniciar un renglón o usarse después de la instrucción. Todo lo que siga a la comilla simple (') será un comentario y no será considerado por el compilador.

Los pins del microcontrolador pueden accesarse de diferentes maneras. La más sencilla seria:

```
PortB.0 = 1           'Enciende el bit 0 del Puerto B
```

PortB.5=0 'Apaga el bit 5 del Puerto B

Para recordar fácilmente cual es el propósito de cada pin en un determinado programa se le puede asignar un nombre utilizando la instrucción VAR. De esta manera el nombre puede ser utilizado después en cualquier parte del programa.

En la instrucción:

S0 VAR PortB.0

El nombre del bit 0 del Puerto B sera para el resto del programa S0

Otro ejemplo seria:

LedEmergencia VAR PortA.2

Quiere decir que en el resto del programa el bit 2 del Puerto A se llamara LedEmergencia.

Volviendo a nuestro ejemplo eje1.bas

S0 será entonces el nombre del bit 0 del Puerto B
S1 será el nombre del bit 1 del Puerto B y así sucesivamente.

A continuación se definen cuales bits serán entradas y cuales serán salidas.

Mediante la instrucción

TrisB = %00000000

Programamos al PIC para que todos los bits del Puerto B sean salidas.

Otro ejemplo sería:

TrisB = %00001111

En este caso los bits 0, 1, 2, y 3 del puerto B serian entradas y el resto salidas.

Siguiendo adelante, nos encontramos con la "etiqueta"

Inicio:

Una etiqueta es el nombre de un renglón. Se utilizan para controlar el flujo del programa, algunas instrucciones realizan saltos desde otros renglones del programa a etiquetas específicas.

Para indicar el fin de la etiqueta se deben de usar los dos puntos (:)

```
PortB = 0
```

Esta instrucción apaga todos los bits del Puerto B. Es una buena manera de asegurarse que estamos iniciando el programa con todo apagado.

Con las instrucciones siguientes, se enciende S1 se programa una Pausa de 1 seg. y se apaga S1

```
S1 = 1  
Pause 1000  
S1= 0
```

Pause es una instrucción. Así de simple. Pause y en seguida el numero de milisegundos que desamos que el microcontrolador espere.

```
End
```

Es el final del programa. No hay que olvidar ponerlo en todos los programas.

2.- Compilar

Para compilar usaremos PBP y obtendremos un file eje1.hex.

El programa pbp.exe es un programa DOS que debe ser invocado desde la línea de comandos con el siguiente formato:

```
C:\PBP> pbp Filename
```

En nuestro ejercicio seria así:

```
C:\PBP> pbp eje1.bas
```

Algunas observaciones son necesarias.

El file que se va a compilar (en este caso eje1.bas) debe de estar en la misma ruta que pbp. En nuestro ejemplo se ha creado una carpeta con el nombre PBP en donde se ha colocado tanto el compilador como el programa que se va a compilar.

Como normalmente se trabaja con Windows. Se puede crear un acceso directo de DOS, Fig. 2



Fig. 2

Para trabajar más cómodamente se pueden modificar este acceso directo. Con el botón derecho del mouse se puede acceder rápidamente a las propiedades. Fig. 3

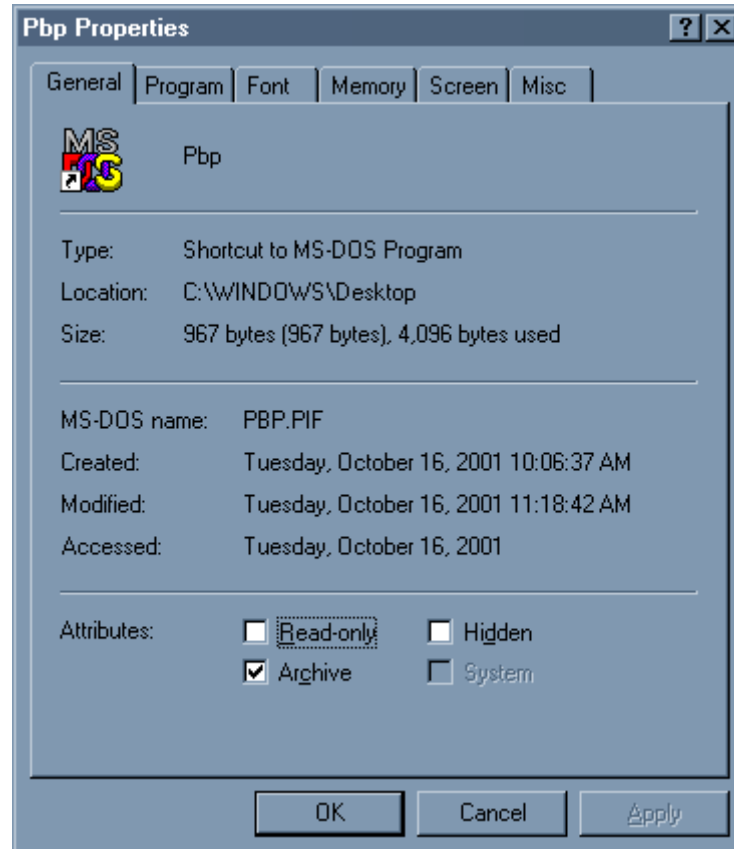
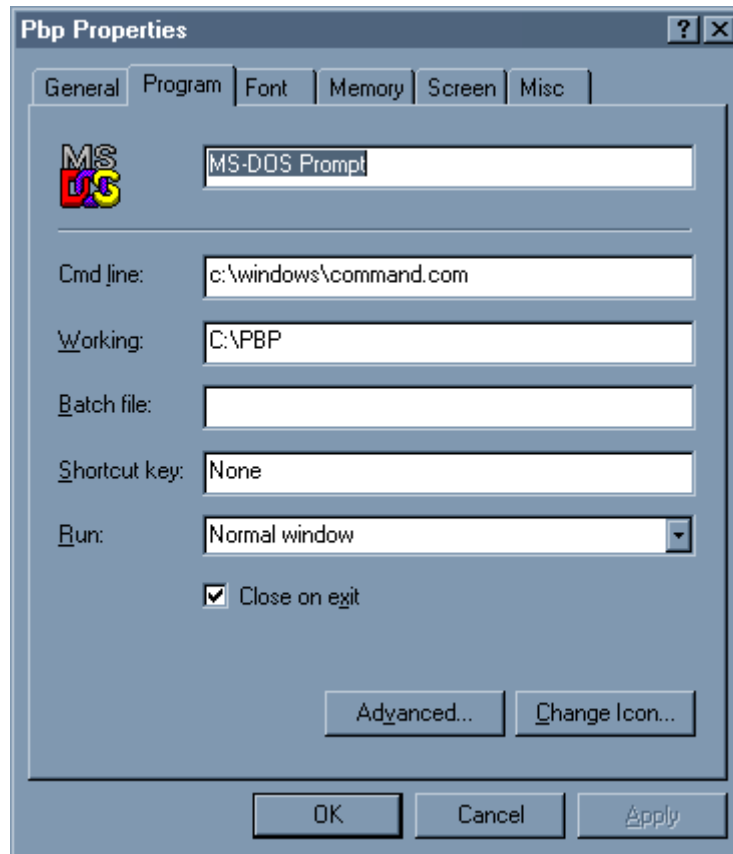


Fig. 3

En la solapa "Program" se hacen las modificaciones que se indican en la Fig. 4 para obtener un acceso más rápido al programa pbp.exe



3.- Quemar el PIC

Para quemar el PIC se requiere un hardware, una tarjeta, pues bien, dos alternativas:

a).- PICSTART PLUS

Es el programador de MICROCHIP. Con él se puede quemar toda la línea de PICs y además está bien integrado con MPLAB. Desafortunadamente este no es gratis.

b).-Prog2

Existen varios programadores que se ofrecen en Internet. Uno de ellos es el llamado JDM84 que se ofrece con el número de parte Prog2 (Clave 501). Con este no se pueden grabar todos los PIC pero sí los más importantes. El software necesario para usar este programador viene en el disco que acompaña a la tarjeta. A saber: Icprog.exe y Pic2.exe. Este programador tampoco es gratis pero sí es más económico. Fig. 5

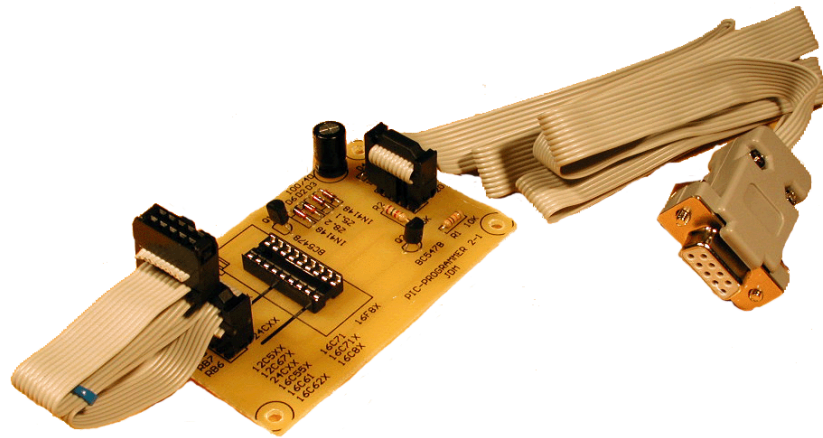


Fig. 5

Entonces colocamos el PIC en el Programador Prog2, abrimos el archivo eje1.hex y hacemos CLIK en programar .

Probar el Programa

Ya tenemos el PIC con su programa grabado. Lo que resta por hacer es insertarlo en la tarjeta Test1 (Clave 502) y probar. Fig. 6

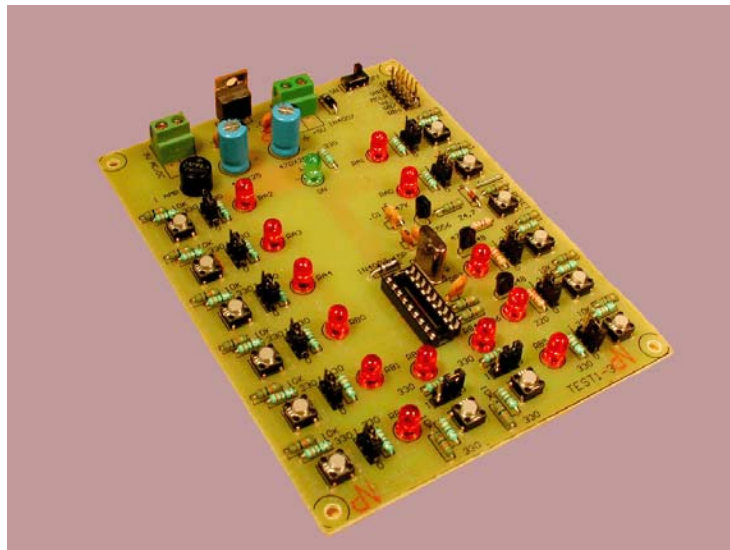


Fig. 6

Veamos el ejemplo eje2.bas

S7. Aquí se trata de encender durante un segundo y en forma consecutiva las salida S0 a

```
'Para Test1
'***Usar compilador PBP***
'Enciende S0 a S7 durante un segundo en forma consecutiva
```

```
S0  VAR  PortB.0
S1  VAR  PortB.1
S2  VAR  PortB.2
S3  VAR  PortB.3
S4  VAR  PortB.4
S5  VAR  PortB.5
S6  VAR  PortB.6
S7  VAR  PortB.7
```

```
TrisB=%00000000
```

```
PortB=0
```

Inicio:

```
S0=1
Pause 1000
S0=0
Pause 500
```

```
S1=1
Pause 1000
S1=0
Pause 500
```

```
S2=1
Pause 1000
S2=0
Pause 500
```

```
S3=1
Pause 1000
S3=0
Pause 500
```

```
S4=1
```

```
Pause 1000
S4=0
Pause 500

S5=1
Pause 1000
S5=0
Pause 500

S6=1
Pause 1000
S6=0
Pause 500

S7=1
Pause 1000
S7=0
Pause 500

Goto Inicio
```

```
End
```

Observe que se forma un ciclo infinito con la instrucción Goto Inicio.

Pause se puede utilizar muchas veces en el programa y se puede cambiar su argumento. En este ejemplo se mantienen encendidas las salidas 1 seg y apagadas 500 milisegundos.

Estos dos ejemplos solo nos enseñan de manera somera la mecanica que se debe seguir para programar un PIC en lenguaje BASIC. En un futuro cercano haremos uso de este poderoso lenguaje para resolver aplicaciones como la comunicaci3n de un PIC con la PC via el canal seria RSC232.