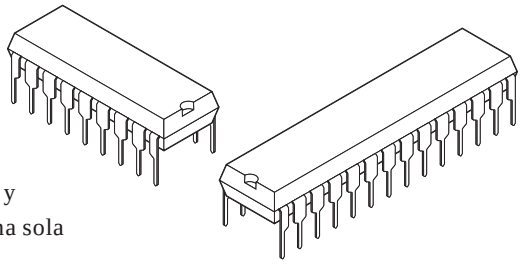
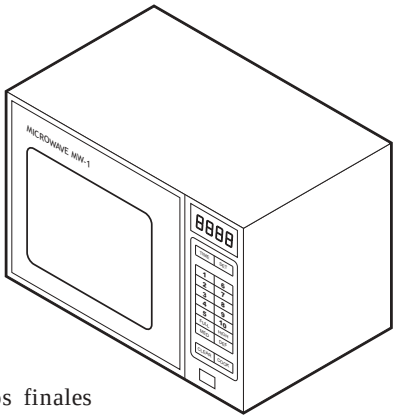
INTRODUCCIÓN AL SISTEMA PICAXE

El microcontrolador PIC (microcontrolador programable) es a menudo descrito como un "ordenador en un chip". Es un circuito integrado que contiene memoria, unidades procesadoras y circuitos de entrada/salida, en una sola unidad.



Los microcontroladores son comprados en "blanco" y luego programados con un programa específico de control. Una vez programado, este microcontrolador es introducido en algún producto para hacerlo más inteligente y fácil de usar.

A manera de ejemplo, un horno de microondas puede utilizar un solo microcontrolador para procesar información proveniente del teclado numérico, mostrar información para el usuario en la pantalla y controlar los dispositivos de salida (motor de la mesa giratoria, luz, timbre y magnetrón).



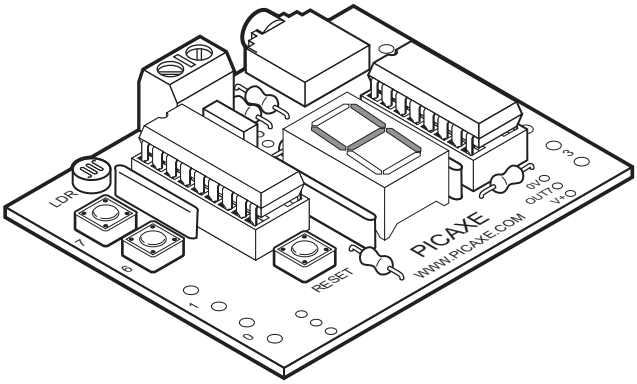
Un microcontrolador puede a menudo reemplazar a un gran número de partes separadas, o incluso a un circuito electrónico completo. Algunas de las ventajas obtenidas con el uso de microcontroladores en el diseño de productos son:

- aumento en la confiabilidad debido al menor número de partes
- reducción en los niveles de existencia ya que un microcontrolador reemplaza varias partes
- simplificación del ensamblaje del producto y productos finales más pequeños
- gran flexibilidad y adaptabilidad del producto ya que las funciones del producto están programadas en el microcontrolador y no incorporadas en el hardware electrónico
- rapidez en modificaciones y desarrollo del producto mediante cambios en el programa del microcontrolador, y no en el hardware electrónico

Algunas de las aplicaciones que utilizan microcontroladores incluyen artefactos domésticos, sistemas de alarma, equipo médico, subsistemas de automóviles y equipo electrónico de instrumentación. Algunos automóviles modernos contienen mas de treinta microcontroladores – utilizados en una amplia variedad de subsistemas desde el control del motor hasta el cierre a control remoto!

En la Industria, los microcontroladores son usualmente programados utilizando programación en lenguaje C. Sin embargo, debido a la complejidad de este lenguaje, es muy difícil para estudiantes muy jóvenes de bachillerato el uso adecuado de dichos lenguajes.

EL SISTEMA PICAXE



El sistema "PICAXE" es un sistema de microcontrolador fácil de programar que utiliza un lenguaje BASIC muy simple, el cual la mayoría de los estudiantes pueden aprender rápidamente. El sistema PICAXE explota las características únicas de la nueva generación de microcontroladores de bajo costo FLASH. Estos microcontroladores pueden ser programados una y otra vez sin la necesidad de un costoso programador PIC.

El poder del sistema PICAXE radica en su sencillez. No necesita de ningún programador, borrador o complejo sistema electrónico – el microcontrolador es programado (con un simple programa en BASIC o un diagrama de flujo) mediante una conexión de tres alambres conectada al puerto serie del ordenador. El circuito operacional PICAXE utiliza únicamente tres componentes y puede ser ensamblado fácilmente en un tablero experimental para componentes electrónicos, en una placa corriente o en una placa PCB.

EL sistema PICAXE está disponible en dos variedades – 18 pines y 28 pines. El controlador PICAXE-28 provee 22 pines de entrada/salida (8 salidas digitales, 8 entradas digitales y 4 entradas analógicas). El sistema PICAXE-18 provee 8 salidas y 5 entradas.

Las características principales del sistema PICAXE son las siguientes:

- bajo costo, circuito de fácil construcción
- hasta 8 entradas, 8 salidas y 4 canales analógicos
- rápida operación de descarga mediante el cable serial
- Software "Editor de Programación" gratuito y de fácil uso
- lenguaje BASIC simple y fácil de aprender
- editor de diagramas de flujo incluido
- puede ser programado también mediante el software "Crocodile Technology"
- extenso número de manuales gratuitos y foro de apoyo en línea
- tablero experimental tutorial y tutoriales incluidos
- paquete de control remoto infrarrojo disponible
- paquete de servocontrolador disponible

TUTORIAL 1 – EL SISTEMA PICAXE

El sistema PICAXE consiste en tres componentes principales:

El Software "Editor de Programación"

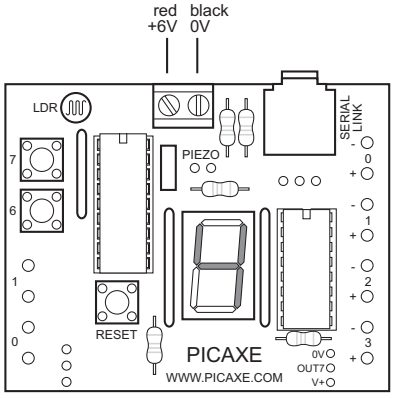
Este software debe ser ejecutado en un ordenador y permite utilizar el teclado del ordenador para escribir programas en un simple lenguaje BASIC. Los programas también pueden generarse dibujando diagramas de flujo. Alternativamente, el software "Crocodile Technology" puede ser utilizado para simular circuitos electrónicos completos, programándolos con diagramas de flujo. Por favor vea el apéndice de "Crocodile Technology" para mayor información.

El cable serie

Este es el cable que conecta el sistema PICAXE al ordenador. El cable sólo necesita ser conectado durante la descarga de programas. No debe ser conectado cuando el PICAXE está siendo ejecutado debido a que el programa esta permanentemente almacenado en el chip PICAXE - aún cuando la fuente de alimentación a sido desconectada!
Hay dos tipos de cables para descarga disponibles – al usar el tablero experimental tutorial cualquiera de los dos cables puede ser utilizado – los cuales se conectan ya sea a la cabecera de tres pines o al enchufe hembra estereo.

El chip PICAXE y el tablero electrónico

El microcontrolador PICAXE ejecuta programas que han sido descargados al mismo. Sin embargo, para operar, el chip debe ser montado en un tablero electrónico que provea una conexión al chip microcontrolador.



El tablero electrónico puede ser diseñado por el usuario en un circuito impreso, en una interfase prefabricada o, para ahorrar tiempo y por conveniencia, utilizar el tablero electrónico tutorial incluido. Este tutorial asume el uso del microcontrolador PICAXE-18 (18 pines) montado en el tablero electrónico tutorial.

Resumen – Procedimiento de programación

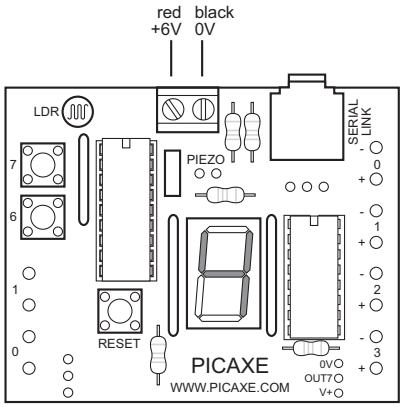
1. Escriba el programa en el ordenador utilizando el software "Programming Editor".
2. Conecte el cable de descarga desde el ordenador al PICAXE.
3. Conecte el acumulador eléctrico (batería) al PICAXE.
4. Utilice el software "Editor de Programación" para descargar el programa. El cable de descarga puede ser removido posterior a la descarga.

El programa comenzará a ejecutarse en el PICAXE automáticamente. Sin embargo, el programa puede ser reiniciado en cualquier momento presionando el interruptor de reinicio.

Tableros PICAXE-18

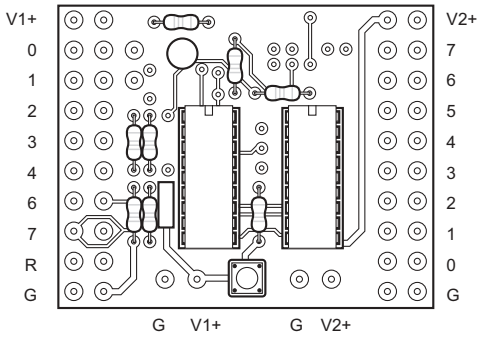
Hay tres tipos de tableros electrónicos de tutoriales/proyectos disponibles:

Tablero electrónico tutorial



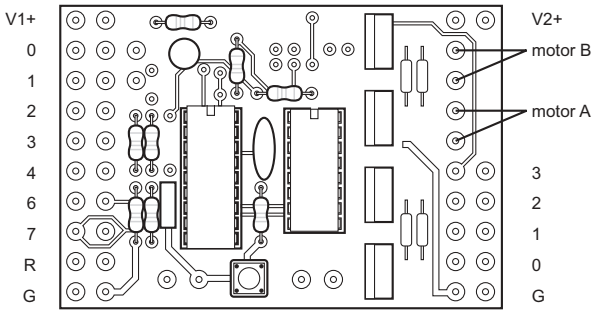
Este es un tablero tutorial que contiene interruptores, sensores, una pantalla de siete barras y conexiones para dispositivos de salida. Este es el tablero descrito en este documento.

Tablero de Proyecto Estándar



Este es un tablero de proyecto que provee de 8 salidas (encendido / apagado ó on/off) digitales mediante un controlador darlington.

Tablero para Proyecto de Alta Potencia

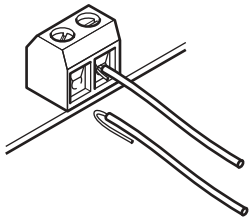


Este es un tablero que provee de 4 salidas digitales (mediante controladores FET) y dos salidas para motores reversibles.

Preparando el Tablero Tutorial

Antes de usar el tablero tutorial, el mismo debe ser conectado a las baterías eléctricas.

Enchufe hembra para caja de baterias



Localice los cables de la caja de baterías y doble el alambre desnudo sobre el aislante en ambos cables. Coloque el cable rojo en el enchufe hembra marcado "V+" y el negro en el enchufe marcado "O V".
Apriete los tornillos de manera que tanto el alambre desnudo como el aislante queden atrapados en el enchufe – esto provee una conexión más fuerte que simplemente prensando el alambre desnudo.

Utilice siempre la caja de baterías eléctricas de 6V (se requiere de 4 pilas AA) suministrada. NO use una bateria PP3 de 9V.

Revestimiento antisoldadura
El tablero tutorial ha sido fabricado utilizando soldadura de onda. Para evitar que soldadura se adhiera a los agujeros de repuesto (para componentes opcionales), la base del tablero está cubierto de un "revestimiento antisoldadura repelable". Este revestimiento debe ser removido antes de soldar componentes opcionales.

Instalando el Software

Requerimientos:
Windows 95/98/ME/NT/2000/XP

El software está incluido en un CD y debe ejecutarse automáticamente al insertarlo en el ordenador. Si el CD no se ejecuta automáticamente, use su programa buscador (por ejemplo el Explorer) para abrir el archivo index.htm.

Instale luego el software "Editor de Programación" siguiendo las instrucciones en su pantalla. Si no lo tiene en su ordenador, deberá también instalar el software "Adobe Acrobat Reader", ya que lo necesitará para poder leer los manuales de ayuda.

Bajando un Programa de Muestra

El siguiente programa enciende y apaga la salida 7 cada segundo. Cuando usted descarga este programa el punto decimal en la pantalla de siete barras del tablero electrónico debe encenderse y apagarse cada segundo.

```
main:
      high 7
      pause 1000
      low 7
      pause 1000
      goto main
```

Este programa utiliza los comandos *high* y *low* para controlar el pin de salida 7, y utiliza el comando de pausa para causar un retardo (1000 ms = 1 segundo).

El último comando, el comando *goto* hace que el programa salte a la etiqueta *main*, que es el comienzo del programa. Esto significa que el programa es un bucle perpetuo. Note que la primera vez que la etiqueta es utilizada debe estar seguida por un símbolo de dos puntos (:). Este símbolo indica al ordenador que la palabra es una nueva etiqueta.

Instrucciones detalladas:

1. Conecte el cable PICAXE a un puerto serie del ordenador y preste atención a cual de los puertos lo conecta (normalmente COM1 ó COM2).
2. Ejecute el Software “Programming Editor”.
3. En el menú desplegable escoja Ver>Opciones para acceder la pantalla de opciones (esta puede que aparezca automáticamente).
4. Haga clic en “Modo” y seleccione PICAXE-18
5. Haga clic en “Puerto Serie” y seleccione el puerto serie al cual el cable PICAXE está conectado, luego haga clic en OK.
6. Escriba el siguiente programa:

```
main:
      high 7
      pause 1000
      low 7
      pause 1000
      goto main
```

(Tome en cuenta el símbolo de dos puntos (:) directamente después de la etiqueta “main” y los espacios entre los comandos y los números)

7. Asegúrese que el circuito PICAXE esté conectado al cable serie y a las baterías.
8. Seleccione PICAXE>Ejecutar. Una barra de descarga de programa debe aparecer mientras el programa es descargado. Al terminar la descarga, el programa debe comenzar a ejecutarse automáticamente – el punto decimal LED (Light Emitting Diode – Diodo Emisor de Luz) en la salida 7 deberá encenderse y apagarse cada segundo.

Instrucciones para uso en Windows

Botones de comandos de la barra de herramientas:

Open = Abrir Save = Salvar Cut = Cortar Copy = Copiar Paste = Pegar Print = Imprimir Run = Ejecutar

Para descargar y ejecutar un programa:

1. Verifique que el cable de descarga esté conectado tanto al PICAXE como al puerto serie del ordenador.
2. Verifique que las baterías estén conectadas al PICAXE.
3. Asegúrese que el Software "Programming Editor" esté en el modo correcto (La palabra PICAXE-18 debe aparecer en la barra de estado en la esquina inferior izquierda de la pantalla).
4. Haga clic en Ejecutar (ó en el respectivo botón de la barra de herramientas)

Para salvar un programa:

1. Haga clic en Archivo- Guardar como…(ó en el respectivo botón de la barra de herramientas)
2. Escriba el nombre bajo el cual quiere guardar el archivo
3. Haga clic en <OK>

Para abrir un programa guardado:

1. Haga clic en Archivo- Abrir (ó en el respectivo botón de la barra de herramientas)
2. Seleccione el archivo deseado de la lista haciendo clic en el mismo.

Haga clic en <OK>

Para iniciar un nuevo programa:

1. Haga clic en Archivo- Nuevo

Para imprimir un programa:

1. Haga clic en Archivo- Imprimir… (ó en el respectivo botón de la barra de herramientas)
2. Si desea que a cada línea del programa se le asigne un número, asegúrese de marcar la casilla "Imprimir números de línea".
3. Haga clic en <OK>

TUTORIAL 2 – UTILIZANDO EL COMANDO *SYMBOL*

Algunas veces es difícil recordar cuales pines están conectados a cuales dispositivos. El comando *symbol* puede en estos casos ser utilizado al inicio del programa para renombrar a entradas y salidas. Note que este programa asume la conexión de un timbre externo al pin de salida 7.

```
symbol dp = 7                ‘renombrar salida 7 “dp” (punto decimal)
symbol buzzer = 1       ‘renombrar salida 1 “buzzer” (timbre)

main:                   ‘hacer una etiqueta llamada “main”
      high dp                ‘LED encendido
      low buzzer        ‘timbre apagado
      wait 1                 ‘esperar 1 segundo
      low dp                 ‘LED apagado
      high  buzzer           ‘timbre encendido
      wait 1                 ‘esperar 1 segundo
      goto main         ‘regresar al inicio (“main”)
```

Recuerde que los comentarios [explicaciones posteriores al símbolo de apóstrofe (‘)] facilitan mucho la comprensión de cada línea del programa. Estos comentarios son ignorados por el ordenador al descargar un programa al PICAXE.

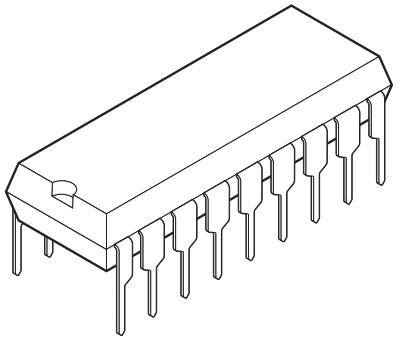
Las etiquetas (“main:” en el programa de arriba) pueden ser cualquier palabra (con la excepción de palabras claves como por ejemplo “switch”) pero DEBEN empezar con una letra. Cuando la etiqueta es definida por primera vez debe llevar al final el símbolo de dos puntos (:). Esto indica al ordenador que la palabra es una nueva etiqueta.

Este programa utiliza el comando *wait*. Los comandos *wait* y *pause* se utilizan para crear retardos ó tiempos muertos. Sin embargo, el comando *wait* puede ser utilizado únicamente con segundos enteros mientras que *pause* se puede utilizar para retardos más cortos (el mismo se asigna en milésimas de segundo).

Al comando *wait* se le pueden asignar números del 1 al 65, los cuales deber escribirse posterior al comando. Al comando *pause* se le pueden asignar números entre 1 y 65535.

Es una buena técnica de programación usar tabulaciones (o espacios) al inicio de líneas sin etiquetas de manera que los comandos estén alineados. El término “espacios en blanco” es utilizado por programadores para definir tabulaciones, espacios y líneas en blanco. Dichos “”espacios en blanco”, utilizados correctamente, hacen al programa mucho más fácil de leer y entender.

Nota:
Algunas versiones antiguas de lenguaje BASIC utilizan “números de línea” en vez de etiquetas para trabajar con los comandos *goto*. Desafortunadamente, este sistema puede ser muy inconveniente ya que si el programa es modificado posteriormente agregando o eliminando líneas, todos los números de línea posteriores deben ser modificados. El sistema de etiquetas utilizado en la mayor parte de las versiones modernas de lenguaje BASIC supera este problema automáticamente.



El “cerebro” del sistema PICAXE es el microcontrolador de 18 pines PIC 16F627. Aunque los microcontroladores son relativamente baratos (algunos microcontroladores cuestan menos de 1.50), los mismos son dispositivos muy complejos que contienen miles de transistores, resistencias y otros componentes electrónicos.

El microcontrolador PICAXE almacena sus programas en su memoria FLASH “no volátil”. La ventaja de esta memoria es que no pierde el programa descargado cuando la fuente de alimentación (baterías) es desconectada del circuito – cuando las baterías son reconectadas el programa se inicia nuevamente. Sin embargo, cuando desee reprogramar el PICAXE, puede descargar un nuevo programa; esta acción borra el viejo programa almacenado en la memoria y almacena el nuevo programa en la memoria. La memoria sólo permite el almacenamiento de un programa a la vez.

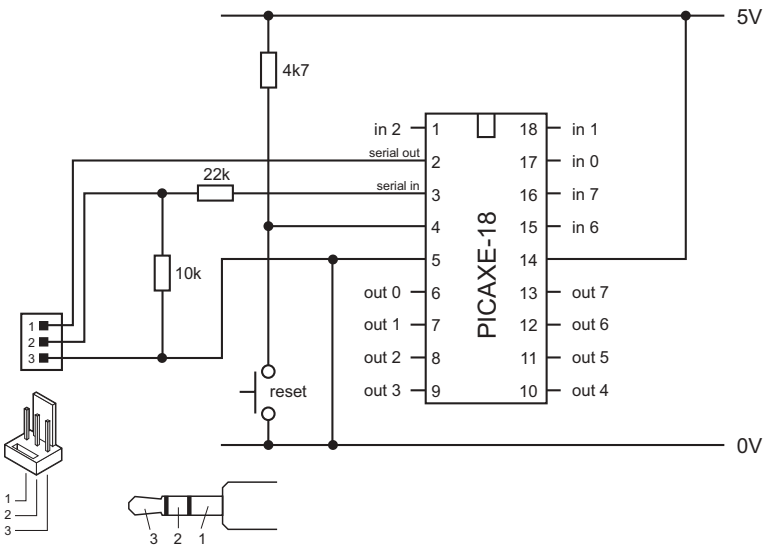
Tome en cuenta que no es posible sacar el programa fuera de la memoria del PICAXE para “leerlo”; por ende, si desea guardar el código de un programa para utilizarlo posteriormente debe guardarlo en su ordenador antes de descargarlo al PICAXE.

El microcontrolador contiene además de la memoria ROM (Read Only Memory - Memoria de sólo lectura), memoria temporal RAM (Random Access Memory - Memoria de Acceso Aleatorio).

La memoria RAM es una memoria “temporal” utilizada para almacenar información mientras el programa es ejecutado. La misma es utilizada normalmente para almacenar respuestas de sumas matemáticas que el microcontrolador hace mientras está trabajando. Esta memoria es “volátil”, lo cual significa que tan pronto las baterías son desconectadas, la información almacenada en la misma se pierde.
Hay 14 bytes de memoria temporal disponibles y los mismos son denominados desde b0 a b13 dentro de los programas.

El Circuito PICAXE-18

La siguiente figura muestra el circuito básico PICAXE-18:



The 4k7 resistor is used to pull the PICAXE microcontrollers reset pin (pin 4) high. If desired, a reset switch can also be connected between the reset pin (pin 4) and 0V. When the switch is pushed the PICAXE microcontroller 'resets' to the first line in the program.

El microcontrolador PICAXE-18

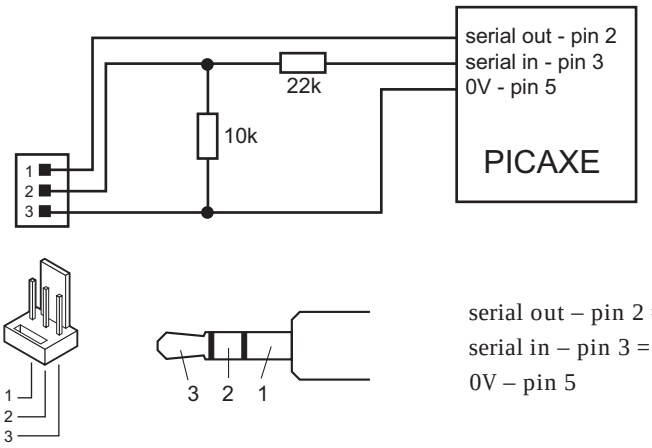
Tome en cuenta que el microcontrolador PICAXE no es un microcontrolador en blanco! El microcontrolador PICAXE esta preprogramado con un programa de carga que permite la descarga directa de programas con el cable suministrado. Los microcontroladores en blanco no tienen este programa y por ende no pueden ser programados mediante el sistema PICAXE.

El microcontrolador PICAXE-18A

El microcontrolador PICAXE-18A es casi idéntico al microcontrolador PICAXE-18 estándar, pero es ligeramente más costoso ya que tiene el doble de capacidad de memoria (aproximadamente 80 líneas de programación BASIC en vez de 40) y salidas analógicas de alta resolución (en vez de baja resolución).

El Circuito PICAXE de Interfase del Ordenador

El sistema PICAXE utiliza una interfase al puerto serie del ordenador muy simple. Aunque esta interfase no utiliza verdaderos voltajes RS232, es de muy bajo costo y ha tenido un desempeño confiable en casi todos los ordenadores modernos.



serial out – pin 2 = salida serie – pin 2
serial in – pin 3 = entrada serie – pin 3
0V – pin 5

It is strongly recommended that this interfacing circuit is included on every PCB designed to be used with the PICAXE microcontroller. This enables the PICAXE microcontroller to be re-programmed without removing from the PCB.

Nota:

La mayor parte de los ordenadores modernos tienen dos puertos serie, normalmente denominados COM1 y COM2. El software “Editor de Programación debe ser configurado con el puerto al cual el microcontrolador está conectado – en el menú desplegable seleccione Ver>Opciones>Puerto Serie para elegir el puerto serie correspondiente en su ordenador.

Si utiliza un ordenador que posee el antiguo conectador de puerto serie de 25 pines, utilice un adaptador 9-25 para poder conectar el cable PICAXE de 9 pines. Estos adaptadores pueden ser comprados en cualquier tienda especializada de ordenadores.

TUTORIAL 3 – BUCLES FOR…NEXT

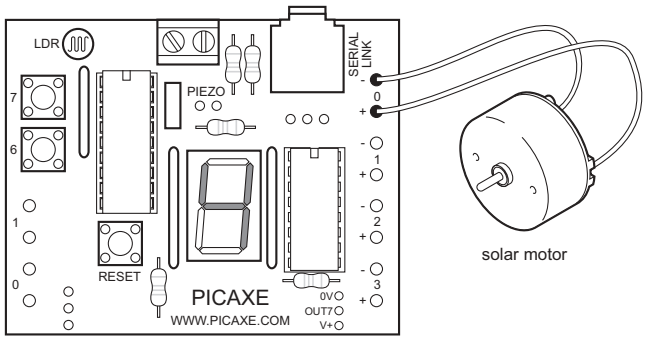
Con frecuencia es útil repetir una parte de un programa varias veces, por ejemplo al encender/apagar sucesivamente un diodo LED (Light Emitting Diode - Diodo emisor de luz). En estos casos un bucle *for…next* puede ser utilizado.

Este programa enciende y apaga 15 veces el diodo LED conectado al pin de salida 7. El número de veces que el código debe ser repetido es almacenado, usando la variable b0 (el PICAXE tiene 14 variables de 1 byte para uso general, nombradas de b0 a b13), en la memoria RAM del chip PICAXE. Estas variables pueden ser renombradas usando el comando *symbol* con el fin de hacerlas más fácil de recordar.

```
symbol counter =  b0          ‘definir la variable “counter” como b0
symbol dp = 7                       ‘asignar al pin 7 con el “dp”

main: for counter = 1 to 15         ‘iniciar un  bucle for…next
            high dp                 ‘encender pin 7
            pause 500         ‘esperar 0.5 segundos
            low dp                  ‘apagar pin 7
            pause 500         ‘esperar 0.5 segundos
      next counter                  ‘siguiente counter (b0)

      end                     ‘fin del programa
```

Note nuevamente como los espacios en blanco han sido utilizados para mostrar claramente todos los comandos contenidos entre los comandos *for* y *next*.

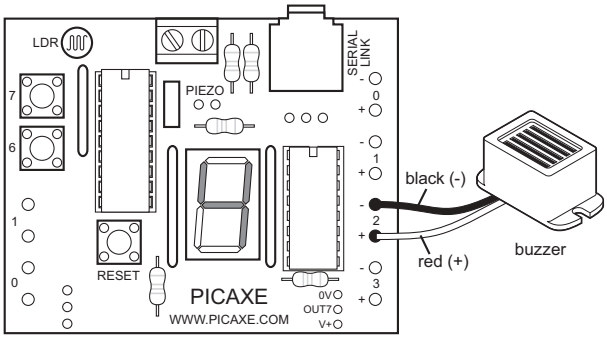
Controlando la velocidad de un motor



Debido a que el sistema PICAXE opera muy rápidamente, es posible controlar la velocidad de motores, encendiéndolos y apagándolos muy rápidamente. Este tipo de control se conoce como PWM (Pulse Width Modulation – Modulación de la anchura del impulso). La PWM es una buena técnica de control ya que permite a los motores operar a bajas velocidades manteniendo un alto torque (fuerza de giro). La PWM es utilizada frecuentemente en muchas aplicaciones, por ejemplo, para controlar la velocidad de taladros y destornilladores eléctricos. Para que la PWM funcione correctamente se necesita de motores de alta calidad. Los programas aquí mostrados están diseñados para motores "solares" y puede que no funcionen correctamente si se utilizan como motores de juguete baratos.

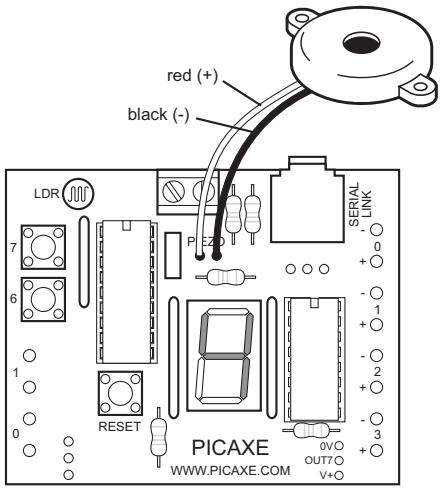
```
symbol mark1 = b6 'renombrar  variables
symbol space1 = b7
symbol mark2 = b8
symbol space2 = b9

let mark1 = 2     'precargar mark1/space1 con relación de 2:10 (1:5)
let space1 = 10

let mark2 = 20    ' precargar mark2/space2 con relación de 20:10 (2:1)
let space2 = 10

main:
      for b2 = 1 to 200 'iniciar un bucle for…next
            high 0            'encender motor
            pause mark1 'esperar tiempo  indicado por mark1
            low 0       'apagar motor
            pause  space1     'esperar tiempo indicado por space1
      next b2                 'siguiente b2
      pause 2000        'detener motor por 2 segundos
      for b2 = 1 to 200 'iniciar un bucle for…next
            high 0            'encender motor
            pause mark2 'esperar tiempo  indicado por mark2
            low 0       'apagar motor
            pause space2      'esperar tiempo indicado por space1
      next b2                 'siguiente b2
      pause  2000       'detener motor por 2 segundos
      goto  main
```

TUTORIAL 4 – TIMBRES Y ZUMBADORES ELECTRÓNICOS



black = negro
red = rojo
buzzer = timbre

Los timbres emiten un sonido cuando están conectados a una fuente de alimentación. Este sonido usualmente esta "fijo" a una frecuencia determinada; así, los timbres solo pueden emitir un solo "tono". Los zumbadores electrónicos usan un tipo de sistema diferente para emitir sonidos y pueden ser utilizados para emitir sonidos en diferentes tonos al proveerlos con una salida "pulsada."



El sistema PICAXE puede crear automáticamente sonidos de diferentes frecuencias utilizando el comando *sound*.

```
main:
      sound 6, (50,100) 'emitir un sonido en salida 6 con
            'frecuencia 50 y longitud 100
      sound 6, (100,100)      'emitir un sonido en salida 6
      sound 6, (120,100)      'emitir un sonido en salida 6
      pause 1000        'esperar 1 segundo
      goto main         'saltar al inicio del programa (main)
```

Para probar este programa se debe instalar un zumbador electrónico (no suministrado, número de parte: SPE002) en el tablero tutorial. Para hacer esto, ubique los conectores del zumbador, marcados con la palabra PIEZO, aproximadamente en el centro del tablero tutorial. Luego solde el alambre rojo al agujero marcado "+" y el alambre negro al agujero marcado "-".

En el programa, el primer número indica el número de pin (en el tablero tutorial el pin de salida 6 es utilizado). El siguiente número es el tono, seguido por último de la duración del sonido (longitud). Mientras más alto sea el número de tono, mayor será la "altura tonal" del sonido. (note que algunos zumbadores no pueden producir tonos muy altos y por lo tanto números de tono mayores de 127 puede que no sean escuchados).

El siguiente programa utiliza un bucle *for…next* para producir 120 sonidos diferentes.

```
main:
      for b0 =  1 to 120           'iniciar un bucle for…next
            sound 6, (b0,50)  'emitir  sonido en salida 6
                  'con frec. b0 y longitud 50
      next b0                      'siguiente b0
      end
```

El número almacenado en la variable b0 aumenta 1 unidad en cada bucle (1-2-3, etc.). Por consiguiente, al utilizar la variable b0 para asignar el tono, el mismo puede ser cambiado en cada bucle.

El siguiente programa realiza la misma función pero cambiando el tono en orden descendente, es decir de 120 a 1.

```
main:
      for b0 =  120 to 1 step -1    'iniciar un bucle for…next
                              '(en cuenta regresiva)
            sound 6, (b0,50)  'emitir  sonido en salida 6 con
                              'frec. b0 y longitud 50
      next b0                       'siguiente b0
      end
```

El siguiente programa emite todos los 256 sonidos posibles:

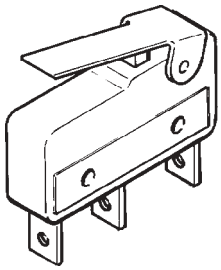
```
main:
      sound 6, (b0,50)  'emitir  sonido en salida 6
      let b0 =  b0 + 1  'sumar 1 al valor de la variable b0
      goto main         'ir a inicio del programa (main)
```

En este último caso el programa es ejecutado indefinidamente. Sin embargo, es importante comprender como el PICAXE ejecuta las operaciones matemáticas.

El PICAXE sólo interpreta números byte, o sea números enteros del 0 al 255. No puede interpretar fracciones, ni números negativos, ni números mayores de 255. Así, si se trata de sumar 1 a 255, el PICAXE saltará de nuevo a 0. Por consiguiente, en el programa anterior, el valor de la variable b0 se comportará de la siguiente manera mientras el programa se ejecuta: …252-253-254-255-0-1-2- etc.

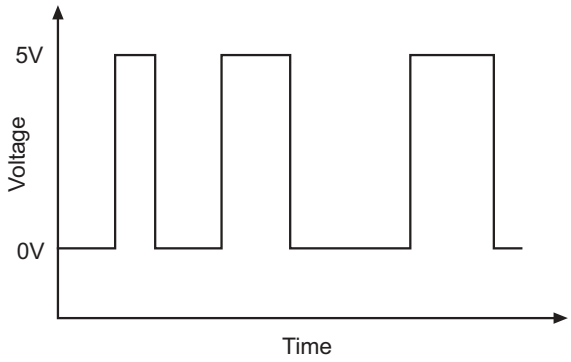
TUTORIAL 5 – UTILIZANDO ENTRADAS (INPUTS)

Sensores Digitales



Un sensor digital es un simple sensor del tipo “interruptor” que solo puede estar en dos posiciones: encendido ó apagado.

Voltage = voltaje
time = tiempo

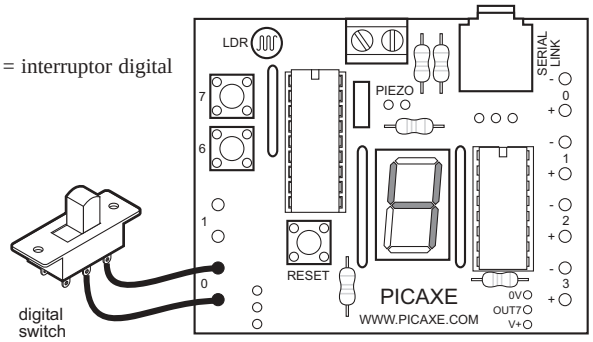


Algunos ejemplos de sensores digitales comunes son:

- microinterruptores
- interruptores de botón de presión e interruptores oscilantes
- interruptores de lengüeta

El tablero tutorial tiene dos interruptores de botón de presión conectados a las entradas 6 y 7. Adicionalmente se pueden conectar otros dos interruptores a las entradas 0 y 1 si se desea.

digital switch = interruptor digital



El siguiente programa indica al PICAXE como reaccionar cuando los interruptores de botón de presión son presionados. En el programa el pin de salida 7 se ilumina cada vez que el interruptor de botón de la entrada 6 es presionado.

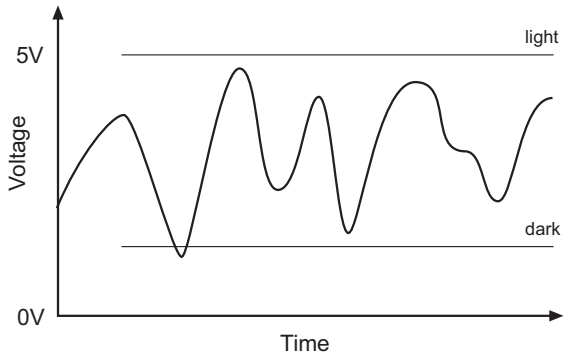
```
main:                          'hacer etiqueta llamada "main"
      if input6 is on then flash    'si la entrada 6 (input 6) está
                               'encendida ir a "flash"
      goto main                'sino  ir a  "main"

flash:                              'hacer etiqueta llamada "flash"
      high 7                        'encender salida 7 (output 7)
      pause 2000               'esperar 2 segundos
      low 7                    'apagar salida7
      goto main                ' ir a "main"
```

En este programa las tres primeras líneas forman un bucle continuo. Si la entrada está apagada el programa se reiniciará una y otra vez.
Una vez que el interruptor es presionado, el programa salta a la etiqueta llamada "flash". El programa luego enciende la salida7 por dos segundos antes de regresar nuevamente a "main".

Note cuidadosamente la ortografía en la línea del comando *if…then* – entrada6 (input6) es una sola palabra (sin espacios en blanco). Note también que únicamente se debe escribir la etiqueta posterior al comando *then* – no se permite ninguna otra palabra aparte de la etiqueta.

Sensores analógicos

Los sensores analógicos miden señales continuas tales como luz, temperatura o posición. El sensor analógico provee de una señal que consiste en un voltaje variable. Este voltaje puede luego ser representado con un número del 0 al 255 (Por ejemplo muy oscuro = 0, luz muy brillante = 255)



Ejemplos típicos de sensores analógicos son:

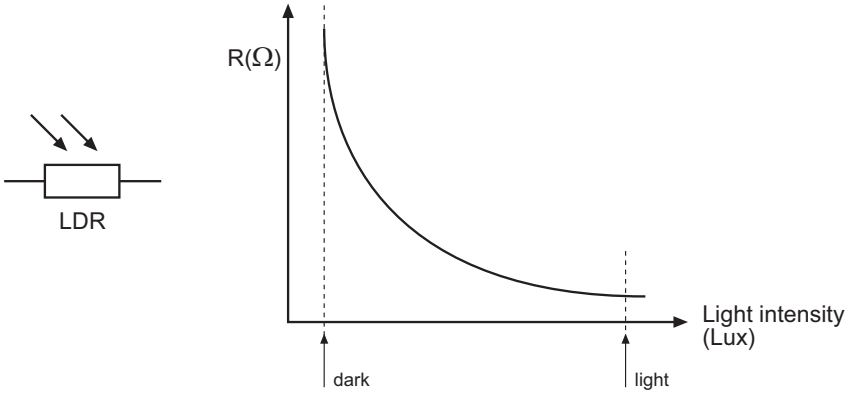
- Fotorresistencias ó LDRs (Light Dependant Resistors - Resistencias variables con la luz)
- Termistores
- Resistencias variables (potenciómetros)

El tablero tutorial consta de una fotorresistencia montada en el mismo , la cual está conectada a la entrada 2 (input2).

Fotorresistencia (LDR)

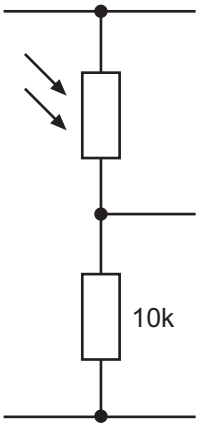
La fotorresistencia es un componente cuya resistencia varia al variar la intensidad de luz que incide sobre la misma, es decir, su resistencia cambia según el nivel de luz. Bajo luz brillante su resistencia es baja (típicamente alrededor de 1k) mientras que en la oscuridad su resistencia es muy alta (típicamente alrededor de 1M).

A continuación se muestra el símbolo y la curva típica de resistencia contra intensidad de luz para la fotorresistencia:



Dark = oscuro light = claro Light intensity (Lux) = intensidad de luz (Lux)

La fotorresistencia está conectada a la entrada 2 en configuración de divisor de voltaje.



Leyendo canales de entradas analógicas

El valor de una entrada analógica puede ser fácilmente copiado dentro de una variable utilizando el comando *readadc*. El valor de la variable (0 a 160) puede luego ser probado. El siguiente programa enciende un diodo LED si el valor es mayor que 120 y otro diodo LED si el valor es menor de 70. Si el valor está entre 70 y 120, ambos diodos LED permanecen apagados.

```
main:                   'hacer etiqueta llamada "main"
      readadc 2,b0              'leer señal de canal 2 en variable b0
      if b0 > 120 then top      'si b0 >120 ir a "top"
      if b0 < 70 the bot        'si b0 <70 ir a "bot"
      low 1             'sino apagar 1
      low 2             ' y apagar 2
      goto main         'ir a "main"

top:                    'hacer etiqueta llamada "top"
      high 1                    'encender 1
      low 2             'apagar 2
      goto main         'ir a "main"

bot:                    'hacer etiqueta llamada "top"
      high 2                    'encender 2
      low 1             'apagar 1
      goto main         'ir a "main"
```

Note que el microcontrolador PICAXE-18 tiene tres canales analógicos nombrados del 0 al 2. En el tablero tutorial la fotorresistencia está conectada permanentemente al pin 2, pero los otros dos canales (0 y 1) están libres para conectar otros sensores.

Frecuentemente al utilizar sensores análogos es necesario calcular el valor de "umbral" necesario para el programa (esto es, los valores 70 y 120 en el programa anterior). El comando *debug* permite ver fácilmente el valor en "tiempo real" de un sensor permitiendo calcular el valor umbral experimentalmente.

```
main:                   'hacer etiqueta llamada "main"
      readadc 2,b0              'leer señal de canal 2 en variable b0
      debug b0          'transmitir valor a la  pantalla del
                  'ordenador
      pause 100         'pausa corta
      goto main         'ir a "main"
```

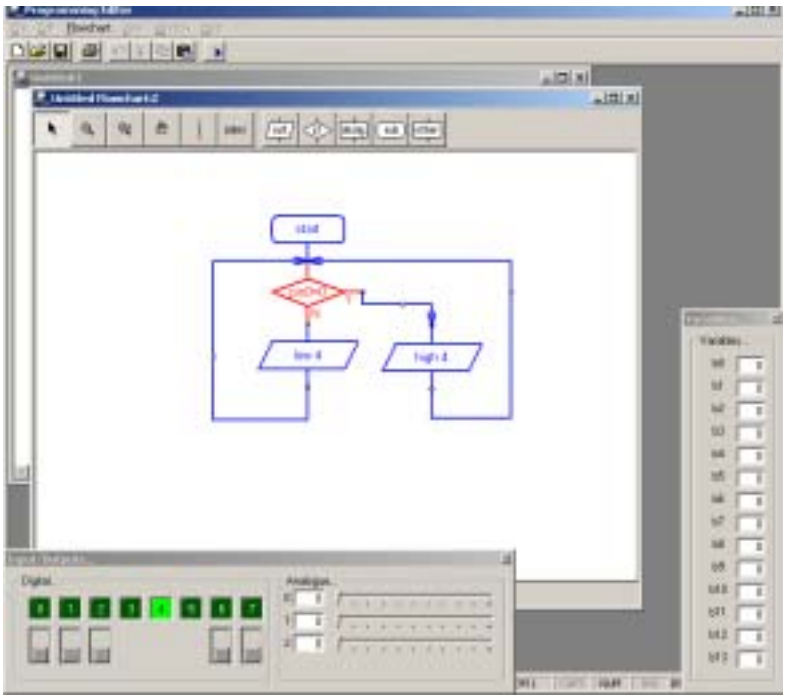
Al ejecutar este programa aparecerá una ventana de depuración (debug) en la pantalla del ordenador, indicando el valor de la variable b0. A medida que el sensor es probado con la variable, la ventana va indicando la lectura actual del sensor.

TUTORIAL 6 – DIBUJANDO ORGANIGRAMAS

Los diagramas de flujo u organigramas son herramientas muy útiles que permiten representar programas gráficamente para hacerlos más fáciles de entender. El software "Programming Editor" incluye un editor de organigramas que permite dibujar los mismos en la pantalla del ordenador. Estos organigramas pueden luego ser convertidos en código BASIC para descargarlos en el PICAXE. Los organigramas pueden también ser impresos o exportados como figuras para incluirlos en reportes de proyectos.

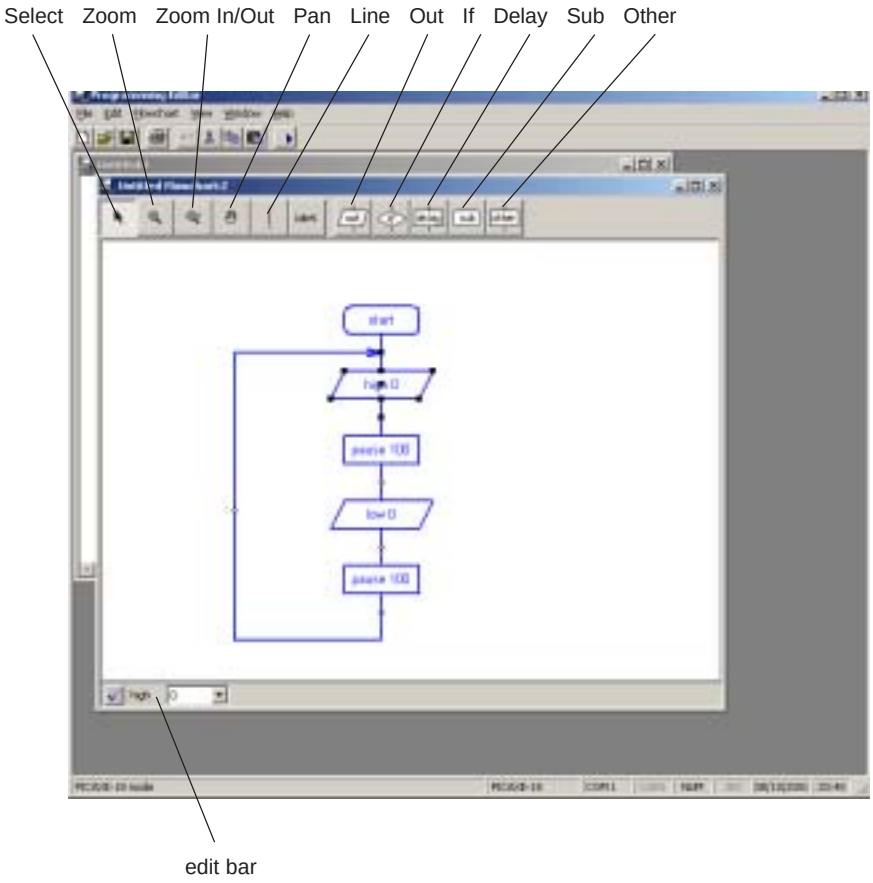
Instrucciones detalladas:

1. Conecte el cable PICAXE a uno de los puertos serie del ordenador. Recuerde tomar nota del puerto serie al cual conecta el cable (normalmente COM1 ó COM2).
2. Inicie el software "Editor de Programación"
3. En el menú desplegable seleccione Ver>Opciones para acceder a la pantalla de opciones (esta puede que aparezca automáticamente).
4. Haga clic en la lengüeta "Modo" y seleccione PICAXE-18.
5. Haga clic en la lengüeta "Puerto Serie" y seleccione el puerto serie al cual ha conectado el cable PICAXE. Haga clic en "OK"
6. Cree un nuevo organigrama haciendo clic en el menú Archivo>Nuevo Organigrama.
7. Dibuje el organigrama mostrado abajo arrastrando los bloques requeridos a la pantalla y luego utilizando el ratón para dibujar las flechas para conectar los bloques.
8. Cuando termine de dibujar el organigrama, puede convertirlo en un programa BASIC seleccionando el menú Organigrama>Convertir Organigrama a BASIC. Luego el programa BASIC puede ser descargado al PICAXE de la manera usual.
9. Para imprimir o salvar el organigrama, use el menú de Archivo. Para exportar el organigrama como una figura, utilice el menú Archivo>Exportar. Para exportar la imagen a un documento de Word seleccione el archivo tipo EMF. Para exportar el organigrama a una página web use el archivo tipo GIF.



El Editor de Organigramas permite dibujar y simular organigramas en la pantalla. El organigrama puede luego ser convertido automáticamente en un programa BASIC para ser descargado en el microcontrolador.

Pantalla de Editor de Organigramas



Select = Seleccionar Zoom = Zoom Zoom In/Out = Acercar/Alejar
Pan = Mover Line = Línea edit bar = barra editora

Seleccionar
Utilice este comando para seleccionar y mover bloques. Cuando un sólo bloque es seleccionado, su código BASIC puede ser editado en la barra editora en la parte inferior de la ventana.

Zoom
Utilice para acercar una parte del diagrama. Use el clic derecho para alejar.

Acercar/Alejar
Para acercar haga clic y mueva el ratón hacia arriba. Para alejar haga clic y mueva el ratón hacia abajo.

Mover
Utilice este comando para mover el organigrama completo alrededor de la pantalla.

Línea
Utilice este comando para dibujar líneas entre los bloques. Se pueden hacer quiebres en las lineas haciendo clic una vez. Cuando la línea está cerca de un bloque, esta se pegará al punto de conexión del mismo.

Etiqueta
Utilice este comando para añadirle etiquetas o títulos a los elementos del organigrama.

Out / If / Delay / Sub / Other
Haga clic en estos botones para ir al submenú de estos comandos y seleccionar el comando deseado.

Dibujando Organigramas

Para dibujar un organigrama haga clic en uno de los botones de menús de comandos (Salida/Si/ Retardo/Sub/Otro) de la barra de herramientas para ir al submenú de comandos requerido. Seleccione el comando deseado y luego haga clic en la pantalla, en el lugar donde desea situar al comando. No trate de colocar el bloque exactamente en posición en primera instancia – póngalo en la pantalla en las cercanías del área donde desea ubicarlo y luego use el comando Seleccionar *para mover el bloque a la posición correcta.*

Una vez que el bloque esté en posición, haga clic en él de manera que sea resaltado. El código BASIC del objeto aparecerá en la barra editora en la parte inferior de la pantalla. Edite el código si lo requiere y luego presione el botón de tic para salvar los cambios.

Para información adicional acerca de cada comando vea los archivos de ayuda "Comandos BASIC". Note que algunos comandos únicos (por ejemplo *servo* para el PICAXE28) sólo aparecerán cuando el software esté en el modo apropiado (menú Ver>Opciones).

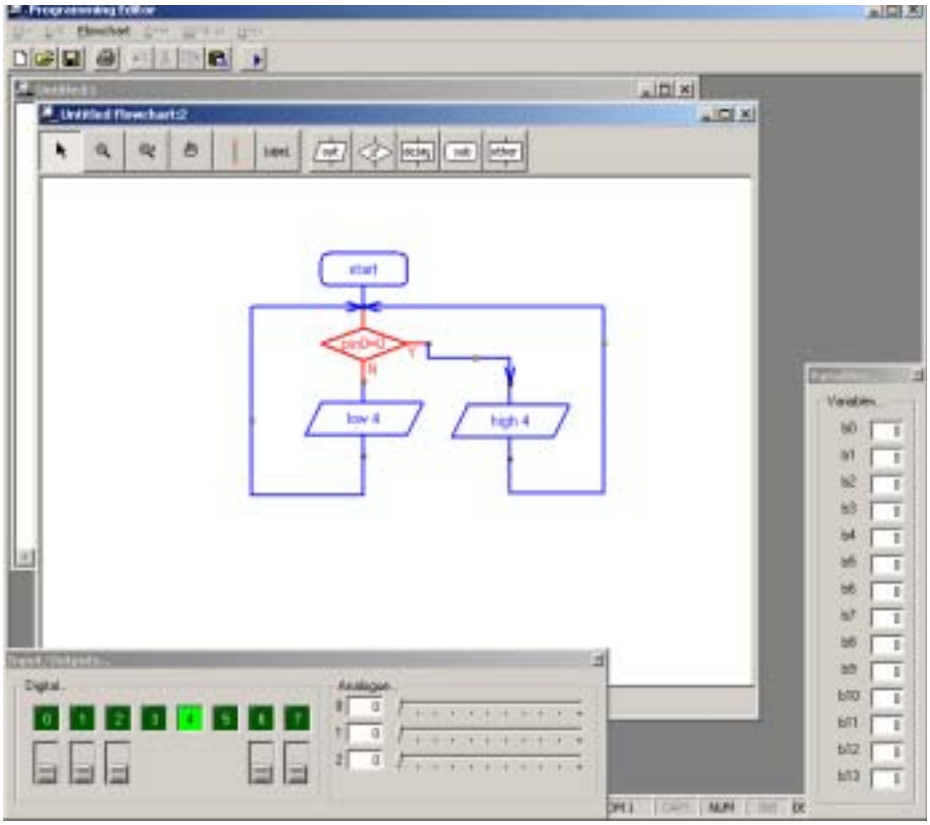
Uniendo bloques

Para unir bloques, se debe acercarlos uno al otro hasta que se junten. Alternativamente, se pueden dibujar líneas entre los mismos usando el comando *línea* en la barra de herramientas. Note que sólo es posible unir la parte inferior de un bloque únicamente con la parte superior de otro. Además, sólo se permite sacar una línea de la parte inferior de conexión de cada bloque.

Para hacer diagramas ordenados, se pueden agregar quiebres a las líneas haciendo clic en las mismas. Cuando una línea es movida cerca de un punto de conexión, la misma se pegará a este; para terminar la línea haga clic una vez mas y la misma quedara en posición.

Las líneas no pueden ser movidas. Si trata de mover una línea la misma será borrada y tendrá que crear una nueva línea.

Simulación de Pantalla



Para simular el organigrama, haga clic en “Simular” en el menú Organigrama. El programa comenzara a ejecutarse en pantalla.

Mientras el programa se ejecuta, los bloques cuyos comandos están siendo ejecutados se irán resaltando en rojo. Las ventanas de “Entradas/Salidas” y “Variables” también aparecerán mientras se ejecuta la simulación. Para cambiar los valores de las entradas haga clic en el respectivo interruptor en pantalla o utilice la barra deslizadora de entradas analógicas.

El tiempo de retardo entre un objeto y otro puede ser ajustado en las Opciones del Organigrama (menú Ver>Opciones>Organigrama).

Note que algunos comandos representan acciones que no pueden ser simuladas en pantalla. En estos casos el comando es simplemente ignorado al ejecutar el organigrama.

Descargando Organigramas

Los organigramas no son descargados directamente al microcontrolador. Primero el organigrama es convertido en un programa BASIC, el cual luego es descargado.

Para convertir un organigrama seleccione “Convertir” en el menú Organigrama; el programa BASIC del organigrama será creado.

Aquellos bloques que no estén conectados a los bloques “inicio” ó “sub” en el organigrama, serán ignorados al momento de hacer la conversión. La conversión se detendrá si se encuentra un bloque no conectado; por lo tanto, utilice siempre un bloque “detener” para terminar el diagrama antes de iniciar una simulación o de convertir el diagrama.

Note que es posible convertir y descargar un organigrama presionando dos veces la tecla F5.

Utilizando Símbolos

Entradas, Salidas y Variables pueden ser renombradas utilizando la “Tabla de Símbolos” del menú Organigrama. Cuando un símbolo es renombrado el nuevo nombre aparecerá en los menús desplegables en la barra editora. No deben utilizarse nombres de comandos (por ejemplo *switch* o *sound*) como símbolos ya que esto puede generar errores en el programa BASIC convertido.

Guardando y Imprimiendo Organigramas

Los organigramas pueden ser guardados, impresos y exportados como figuras (para ser insertados en documentos de procesadores de palabras) utilizando el menú Archivo. Los organigramas pueden también ser copiados al portapapeles de Windows (para pegarlos luego a otras aplicaciones) mediante el menú Editar.

TUTORIAL 7 – SISTEMAS DE NÚMEROS

Los microcontroladores operan realizando un gran número de comandos en un espacio de tiempo muy corto procesando señales electrónicas. Estas señales están codificadas en sistema binario – la señal puede ser *high* (1) o *low* (0).

El sistema numérico utilizado diariamente es el sistema decimal. Este sistema numérico utiliza diez dígitos (del 0 al 9) para explicar que tan grande o pequeño es el número.

Sin embargo al trabajar con microcontroladores es muchas veces más fácil trabajar en código binario; especialmente al tratar de controlar múltiples salidas al mismo tiempo.

Un sólo dígito binario es conocido como un "bit" (binary digit – dígito binario). El sistema PICAXE utiliza 8 bits (1 byte), teniendo al dígito menos significativo en el extremo derecho y al dígito más significativo en el extremo izquierdo.

Por consiguiente, el número binario %11001000 pone a los bits 7,6,3 en *high* (1) y al resto en *low* (0). El símbolo % indica al ordenador que está trabajando en sistema binario y no en decimal.

La utilización del código binario permite controlar las ocho salidas al mismo tiempo, en vez de sólo utilizar los comandos *high* y *low*. El siguiente programa demuestra como hacer que la pantalla de siete barras del tablero tutorial cuente del 0 al 9.

```
main:
      let pins = %00111111          'dígito 0
      pause 250               'esperar 0.25 segundos
      let pins = %00000110          'dígito 1
      pause 250               'esperar 0.25 segundos
      let pins = %01011011          'dígito 2
      pause 250               'esperar 0.25 segundos
      let pins = %01001111          'dígito 3
      pause 250               'esperar 0.25 segundos
      let pins = %01100110          'dígito 4
      pause 250               'esperar 0.25 segundos
      let pins = %01101101          'dígito 5
      pause 250               'esperar 0.25 segundos
      let pins = %01111101          'dígito 6
      pause 250               'esperar 0.25 segundos
      let pins = %00000111          'dígito 7
      pause 250               'esperar 0.25 segundos
      let pins = %01111111          'dígito 8
      pause 250               'esperar 0.25 segundos
      let pins = %01101111          'dígito 9
      pause 250               'esperar 0.25 segundos
      goto main
```

Cada línea "let pins =" cambia el número de barras encendidas en la pantalla de siete segmentos. Esto es mucho más rápido que utilizar muchísimas veces los comandos *high* y *low* para hacer lo mismo.

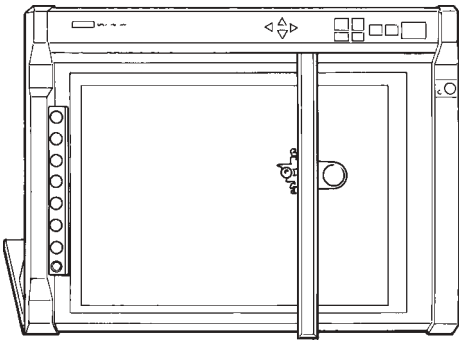
Visualizando Cifras Analógicas en la Pantalla de Siete Barras

Este programa “lee” la cantidad de luz detectada por la fotorresistencia en la entrada 2 y luego visualiza una cifra asignada a dicha cantidad de luz en la pantalla de siete segmentos.

```
main: readadc 2,b1                ‘ leer señal analógica de canal 2
                                ‘ en variable b1
      if b1 > 150 then show9      ‘ probar  variable b1 y saltar al
                        ‘ respectivo comando
      if b1 > 130 then show8
      if b1 > 110 then show7
      if b1 > 90 then show6
      if b1 > 70 then show5
      if b1 > 50 then show4
      if b1 > 30 then show3
      if b1 > 20 then show2
      if b1 > 10 then show1
show0:
      let pins = %00111111        ‘dígito 0
      goto main
show1:
      let pins = %00000110        ‘dígito 1
      goto main
show2:
      let pins = %01011011        ‘dígito 2
      goto main
show3:
      let pins = %01001111        ‘dígito 3
      goto main
show4:
      let pins = %01100110        ‘dígito 4
      goto main
show5:
      let pins = %01101101        ‘dígito 5
      goto main
show6:
      let pins = %01111101        ‘dígito 6
      goto main
show7:
      let pins = %00000111        ‘dígito 7
      goto main
show8:
      let pins = %01111111        ‘dígito 8
      goto main
show9:
      let pins = %01101111        ‘dígito 9
      goto main
```

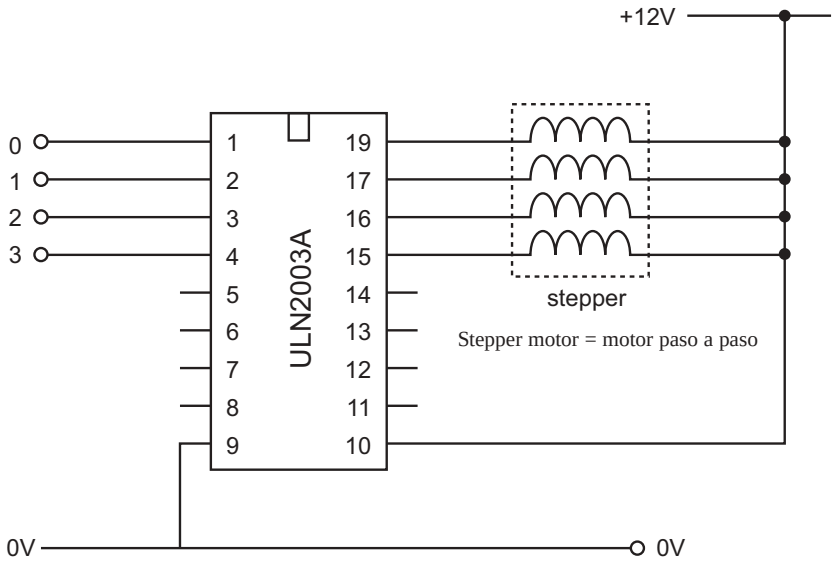
Controlando motores paso a paso

Los motores paso a paso, son motores de alta precisión comúnmente utilizados en unidades de disco, impresoras, plotters y relojes de ordenadores. A diferencia de los motores de CC, los cuales giran libremente al aplicarles potencia, los motores paso a paso requieren que su fuente de alimentación sea continuamente "impulsada" en cuatro patrones diferentes. Por cada impulso, el motor se mueve un "paso", típicamente 7.5° (requiriendo por lo tanto 48 pasos para una revolución completa).

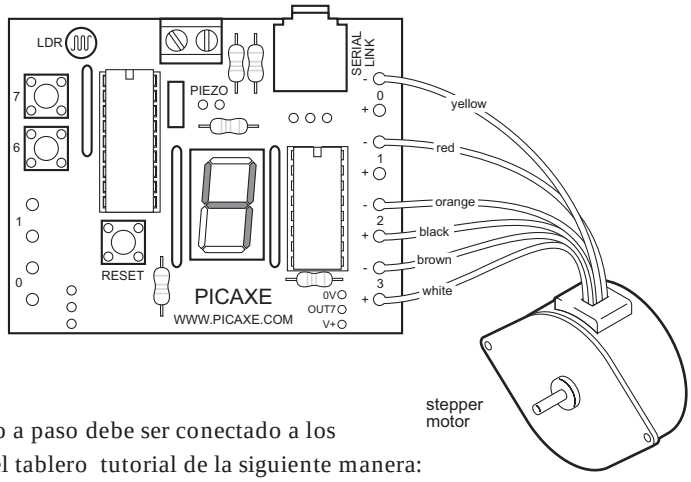


Los motores paso a paso tienen algunas limitaciones. Primero, el consumo de potencia es mayor cuando el motor está detenido (debido a que todas las bobinas requieren estar energizadas). Segundo, la velocidad de operación está limitada a aproximadamente 100 "pasos" por segundo, lo cual equivale a 2 revoluciones por segundo ó 120 RPM.

El motor paso a paso contiene una serie de electroimanes fijos a la armadura central y cuatro bobinas ubicadas alrededor de la carcasa del motor. Cuando corriente eléctrica pasa por estas bobinas, las mismas generan un campo magnético el cual atrae ó repele a los electroimanes permanentes en la armadura, provocando que la armadura gire un "paso" hasta que los campos magnéticos estén alineados. Luego, las bobinas son energizadas con un patrón diferente para crear un campo magnético diferente y provocar que la armadura gire otro "paso".



Para hacer que la armadura gire continuamente, las cuatro bobinas internas del motor paso a paso deben ser encendidas y apagadas continuamente en cierto orden. El chip controlador ULN2003A del tablero tutorial provee del método necesario para interrelacionar a estas cuatro bobinas.



El motor paso a paso debe ser conectado a los agujeros en el tablero tutorial de la siguiente manera:

Cable Negro 2 +
Cable Blanco 3 +
Cable Amarillo 0 –
Cable Rojo 1 –
Cable Naranja 2 –
Cable Marrón 3 –

La siguiente tabla muestra los cuatro “pasos” distintos requeridos para hacer girar el motor:

Paso	**Bobina 4 (Output 3)**	**Bobina 3 (Output 2)**	**Bobinba 2 (Output 1)**	**Bobina 1 (Output 0)**
1	1	0	1	0
2	1	0	0	1
3	0	1	0	1
4	0	1	1	0
1	1	0	1	0

Para hacer girar al motor en dirección contraria, los pasos deben ser invertidos (4-3-2-1-4-etc. en vez de 1-2-3-4-1-etc.)

Nota:
La configuración del alambrado de los motores paso a paso puede variar según el fabricante. Por lo tanto, puede que sea necesario reorganizar las conexiones de las bobinas para que la secuencia mostrada arriba opere correctamente. Un arreglo incorrecto de las bobinas puede causar que el motor vibre en una dirección y otra en vez de girar continuamente. La mayoría de los motores paso a paso están diseñados para trabajar a 12 V, pero generalmente pueden trabajar sin problemas (aunque con un torque reducido) a 6 V.

El siguiente programa también puede utilizar un número binario para encender y apagar todas las líneas de salida al mismo tiempo. La siguiente tabla muestra el número binario de salida para cada paso:

Paso	**Salida binaria**
1	%00001010
2	%00001001
3	%00000101
4	%00000110
1	%00001010

Intente cambiar la velocidad de giro alterando el valor del retardo (delay) en el siguiente programa

```
symbol delay = b0       'definir  variable
      let delay = 100   'fijar el retardo (delay) en 0.1 segundos

main: let pins = %00001010          'primer paso
      pause delay             'pausa de  0.1 seg. (valor
                              'asignado al retardo)
      let pins = %00001001          'siguiente paso
      pause delay             'pausa de  0.1 seg.
      let pins = %00000101          'siguiente paso
      pause delay             'pausa de  0.1 seg.
      let pins = %00000110          'siguiente paso
      pause delay             'pausa de  0.1 seg.
      goto main               'ir a  "main" (bucle  perpetuo)
```

TUTORIAL 8 – SUB-PROCEDIMIENTOS

Un sub-procedimiento es un "mini-programa" separado el cual puede ser llamado desde el programa principal. Una vez que el sub-procedimiento ha sido ejecutado, el programa principal continua.

Los sub-procedimientos son frecuentemente utilizados para separar el programa principal en pequeñas secciones para hacerlo más fácil de comprender. Sub-procedimientos que realizan tareas comunes pueden también ser copiados de programa a programa para ahorrar tiempo.

El siguiente programa utiliza dos sub-procedimientos para separar las dos secciones principales del programa ("flash" y "noise").

```
symbol dp = 7                ‘renombrar salida7 “dp”
symbol buzzer = 6       ‘renombrar  salida6 “buzzer”
symbol counter = b0          ‘ definir a la variable “counter” como b0

main:                   ‘hacer etiqueta llamada “main”
      gosub flash       ‘ir al sub-procedimiento “flash”
      gosub noise       ‘ir al sub-procedimiento “noise”
      goto main         ‘ir a  “main”

      end               ‘fin del programa principal


flash:                  ‘hacer un sub-procedimiento llamado “flash”
      for counter = 1 to 25   ‘iniciar un bucle for…next
            high  dp    ‘encender diodo LED
            pause 50    ‘esperar 0.05 segundos
            low dp            ‘apagar diodo LED
            pause 50    ‘esperar 0.05 segundos
      next counter            ‘siguiente counter (b0)
      return                  ‘retornar del sub-procedimiento

noise:                  ‘hacer un sub-procedimiento llamado “noise”
      high buzzer       ‘encender timbre
      pause 2000        ‘esperar 2 segundos
      low buzzer        ‘apagar timbre
      return                  ‘retornar del sub-procedimiento
```

Este segundo programa muestra como una variable puede ser utilizada para transferir información hacia un sub-procedimiento. En este caso la variable b2 es utilizada para indicar al controlador que debe ejecutar el sub-procedimiento flash primero cinco y luego quince veces.

```
symbol dp = 7                ‘renombrar salida7 “dp”
symbol counter = b0          ‘ definir a la variable “counter” como b0

main:                   ‘hacer etiqueta llamada “main”
      let b2 = 5        ‘precargar a b2 con el  número 5
      gosub flash       ‘ir al sub-procedimiento “flash”
      pause 500         ‘esperar 0.5 segundos
      let b2 = 15       ‘precargar a b2 con el  número 5
      gosub flash       ‘ir al sub-procedimiento “flash”
      pause 500         ‘esperar 0.5 segundos
      goto main         ‘ir a “main”

      end               ‘fin del programa principal


flash:                  ‘hacer un sub-procedimiento llamado “flash”
      for counter = 1 to b2   ‘iniciar un bucle for…next
            high dp           ‘encender diodo LED
            pause 250   ‘esperar 0.25 segundos
            low dp            ‘apagar diodo LED
            pause 250   ‘esperar 0.25 segundos
      next counter            ‘siguiente counter
      return                  ‘retornar del sub-procedimiento
```

¿QUE SIGUE?

Al completar estos tutoriales usted ha aprendido todas las funciones básicas del sistema PICAXE - como configurar el sistema, como desarrollar programas, como dibujar organigramas y como conectar dispositivos de entrada y salida. En este CDROM también hay algunas otras guías de referencia muy útiles que le proveerán con información adicional.

Proyectos Modelo
Su siguiente punto de referencia deberán ser los *proyectos modelo*, los cuales dan ejemplos de como el sistema PICAXE puede ser utilizado en aplicaciones en la vida real. Cada proyecto provee de un diagrama de circuito y un programa el cual puede ser copiado o alterado para cumplir con los requerimientos de su proyecto.

Guía de Comandos BASIC
El lenguaje BASIC utilizado por el sistema PICAXE tiene alrededor de 30 comandos de los cuales sólo unos pocos se han utilizado en este tutorial. Échele un vistazo a los otros comandos disponibles. En la guía, cada comando tiene un pequeño programa para demostrar como el mismo puede ser utilizado dentro de un proyecto.

Guía de Interfase Electrónico
Esta guía explica como adaptar un gran número de dispositivos de entrada y salida al microcontrolador PICAXE. Si desea saber como conectar un timbre, motor, solenoide o fotorresistencia al PICAXE, la respuesta está aquí!

Finalmente, toda la última información y un foro de soporte técnico están disponibles en Internet en

www.picaxe.co.uk

BUENA SUERTE CON SU PROYECTO PICAXE!

APÉNDICE 1: EQUIPAMIENTO REQUERIDO

Todo el equipamiento puede ser comprado en nuestra tienda “en línea” en:

www.tech-supplies.co.uk

Equipamiento requerido para tutoriales dentro de este folleto:
Paquete de tablero tutorial PICAXE18 (AXE050)
4 pilas AA (BAT002)

Conectores opcionales:
3 bloques de terminales atornillables de 4 polos (CON005)

Dispositivos de Salida opcionales:

SPE002	Zumbador electrónico
GBX007	Motor Solar de CC
GBX008	Motor Unipolar paso a paso

APÉNDICE 2: "CROCODILE TECHNOLOGY"

El software Crocodile Technology, de la compañía Crocodile Clips Ltd, permite al estudiante construir y probar circuitos de microcontroladores mediante simulaciones en pantalla. Adicionalmente también permite generar programas creando organigramas y luego descargándolos directamente al microcontrolador. El CDROM suministrado provee de una versión de demostración.

Compatibilidad:

Debe utilizar la versión 1.52 Build 428 (o una más reciente) para poder operar con el sistema PICAXE. Si está utilizando una versión más antigua, por favor descargue una versión más reciente gratis en www.crocodile-clips.com. El software está disponible en el menú Help>About Crocodile Technology.

El más reciente plugin PICAXE para el Crocodile Technology puede ser descargado en www.tech-supplies.co.uk (sección Software>Crocodile Technology).

La versión 1.52 de Crocodile Technology soporta al PICAXE-28 y tiene soporte limitado para el PICAXE-18. Se espera que versiones posteriores de Crocodile Technology den soporte completo al PICAXE-18.

Restricciones de la Versión 1.52 con el PICAXE-28:
Ninguna.

Restricciones de la Versión 1.52 con el PICAXE-18

1. Sólo las entradas digitales 0-2 son soportadas. Las entradas 6 y 7 no pueden ser utilizadas.
2. La entrada 3 aparece en la pantalla aunque la misma no está presente en el PICAXE-18. Por lo tanto esta entrada no debe ser utilizada en simulaciones.
3. Las funciones analógicas del PICAXE-18 no están soportadas.

Configurando el software "Crocodile Technology"

Antes de dibujar un circuito u organigrama, se recomienda configurar el sistema PICAXE mediante el menú Options>Programmer.

En el menú "Options" seleccione "Programmer":

En "Programmer" seleccione "PICAXE"
En "COM Port" seleccione COM1 ó COM2 según sea el caso
En "Default Class" seleccione "18 pin PIC" para PICAXE-18 ó "28 pin PIC" para PICAXE-28

Los valores en "Target microcontroller" no son utilizados y pueden dejarse con los valores por defecto.