USERS

TÉCNICO en
ELECTRÓNICA

CONCEPTOS FUNDAMENTALES Y PRÁCTICA PROFESIONAL

3

PROTEUS

USO BÁSICO DE PROTEUS VSM PARA SIMULAR CIRCUITOS ELECTRÓNICOS, UTILIZAR HERRAMIENTAS DE SIMULACIÓN VIRTUALES Y EJECUTAR ANÁLISIS CON LAS OPCIONES QUE OFRECE EL PROGRAMA

Interfaz ISIS

Controles de simulación

Generadores de señales

Instrumentos virtuales

Microcontroladores en Proteus

USERS

TÉCNICO en ELECTRÓNICA

CONCEPTOS FUNDAMENTALES Y PRÁCTICA PROFESIONAL

3

PROTEUS

TÍTULO: Proteus
AUTOR: Víctor Rossano
COLECCIÓN: Pocket Users
FORMATO: 19 x 13.5 cm
PÁGINAS: 96

Primera impresión realizada en julio de MMXIII.

Sevagraf, Costa Rica 5226, Grand Bourg, Malvinas Argentinas, Pcia. De Buenos Aires.

ISBN 978-987-1949-15-1

Rossano, Víctor
Proteus / Víctor Rossano; coordinado por Paula Budris. - 1a ed. - Buenos Aires: Fox Andina; Dalaga, 2013.
96 p. ; 19 x 13 cm. - (Pocket Users; 35)

ISBN 978-987-1949-15-1

1. Informática. I. Budris, Paula, coord. II. Título
CDD 005.3

VISITE NUESTRA WEB

EN NUESTRO SITIO PUEDE OBTENER, DE FORMA GRATUITA, UN CAPÍTULO DE CADA UNO DE LOS LIBROS EN VERSIÓN PDF Y PREVIEW DIGITAL. ADEMÁS, PODRÁ ACCEDER AL SUMARIO COMPLETO, LIBRO DE UN VISTAZO, IMÁGENES AMPLIADAS DE TAPA Y CONTRATAPA Y MATERIAL ADICIONAL.

RedUSERS
COMUNIDAD DE TECNOLOGIA

redusers.com

Nuestros libros incluyen guías visuales, explicaciones paso a paso, recuadros complementarios, ejercicios, glosarios, atajos de teclado y todos los elementos necesarios para asegurar un aprendizaje exitoso y estar conectado con el mundo de la tecnología.

LLEGAMOS A TODO EL MUNDO VÍA OCA* Y DHL**

* SÓLO VÁLIDO EN LA REPÚBLICA ARGENTINA // ** VÁLIDO EN TODO EL MUNDO EXCEPTO ARGENTINA

usershop.redusers.com usershop@redusers.com + 54 (011) 4110-8700

Prólogo al contenido

Hace algunas décadas, las computadoras personales eran algo prácticamente inalcanzable para la mayoría debido a sus elevados costos. Era muy poco común que alguien contara con una computadora personal en casa. Las únicas máquinas que podíamos usar eran las de los laboratorios de computación de la escuela, que solo corrían el sistema operativo MS-DOS, y teníamos que esperar largo tiempo para tomar un turno de una hora.

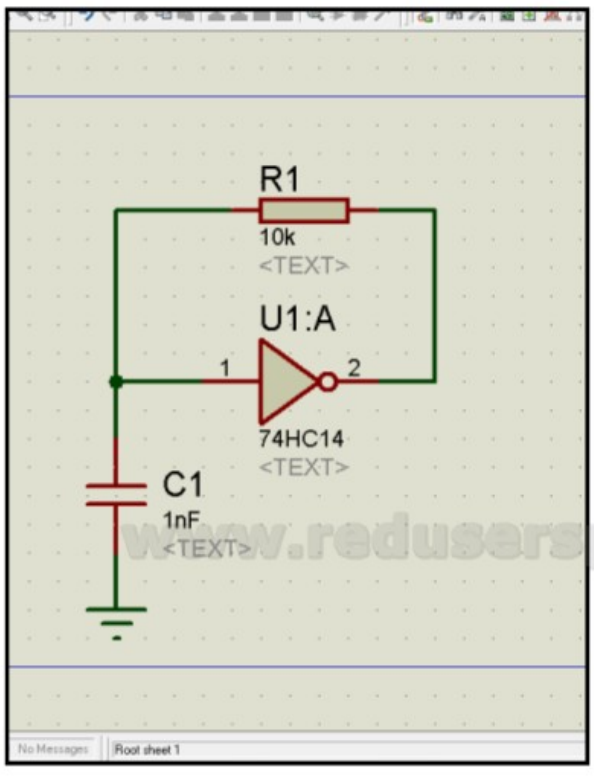
Cuando se comenzó a popularizar el acceso a Internet y los desarrolladores de programas volvieron su mirada hacia la electrónica, se hicieron populares aplicaciones como Proteus.

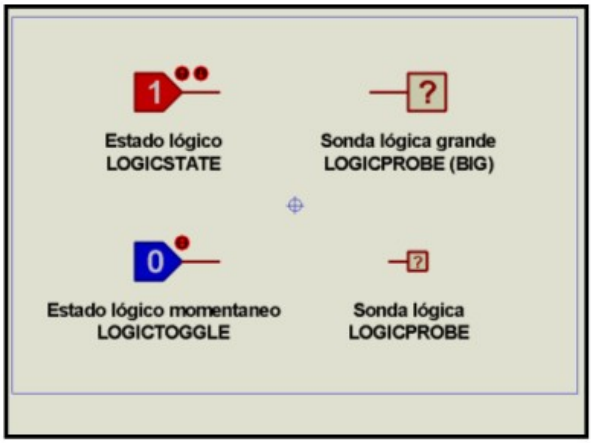
Podemos utilizar este extraordinario programa para simular todo tipo de circuitos electrónicos, y la gran mayoría de los circuitos simulados con éxito funcionaron correctamente en la realidad.

Sin duda nos hubiera servido de mucho contar con herramientas como Proteus en nuestro tiempo como estudiantes. Aunque no podemos quejarnos por haber tenido que armar todos los circuitos físicamente en la escuela, ya que esto nos entrega la experiencia y el conocimiento, pero el hecho de contar con un simulador tan poderoso acelera el aprendizaje y facilita el diseño. En la actualidad, las herramientas computacionales son bastante poderosas, y pueden ayudar en gran medida a los estudiantes o profesionales de diversas áreas.

Por esta razón, en el libro que está entre sus manos se ha volcado la experiencia de utilizar Proteus, para que el lector se convierta en un verdadero experto simulando circuitos.

Contenido del libro





*02

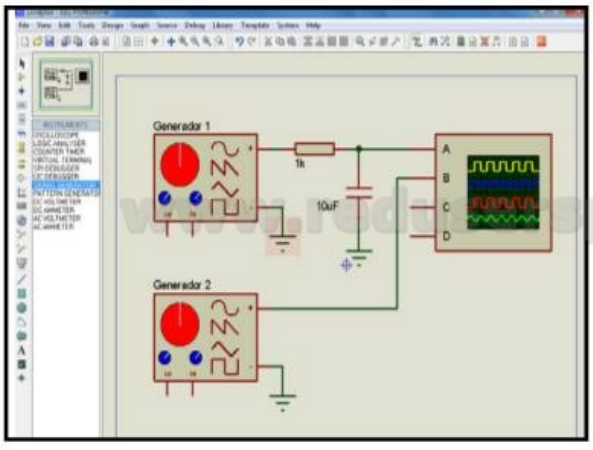
Simulación en Proteus

*03

Señales e instrumentos de medición

*04

Circuitos con microcontroladores PIC



*01

Introducción

En este capítulo introductorio conoceremos el programa Proteus VSM, su interfaz y sus funciones principales. También, aprenderemos a elegir y manejar componentes, y comenzaremos a dibujar diagramas de circuitos electrónicos en el módulo ISIS.

Qué es Proteus VSM

Proteus VSM es un sistema de diseño electrónico basado en la **simulación** analógica, digital o mixta de circuitos, que brinda la posibilidad de interacción con muchos de los elementos que integran el circuito. Incluye componentes animados para la visualización de su comportamiento en tiempo real, además de un completo sistema de generación y análisis de señales. También cuenta con un módulo para el diseño de circuitos impresos.

Las siglas **VSM** significan **Virtual System Modelling**, que en español podemos traducir como **sistema de modelado virtual**, ya que Proteus VSM permite modelar de forma virtual en la computadora prácticamente cualquier circuito. La característica que hace de Proteus VSM uno de los simuladores preferidos por muchos aficionados y profesionales de la electrónica es la posibilidad de simular circuitos que incluyen microprocesadores o microcontroladores.

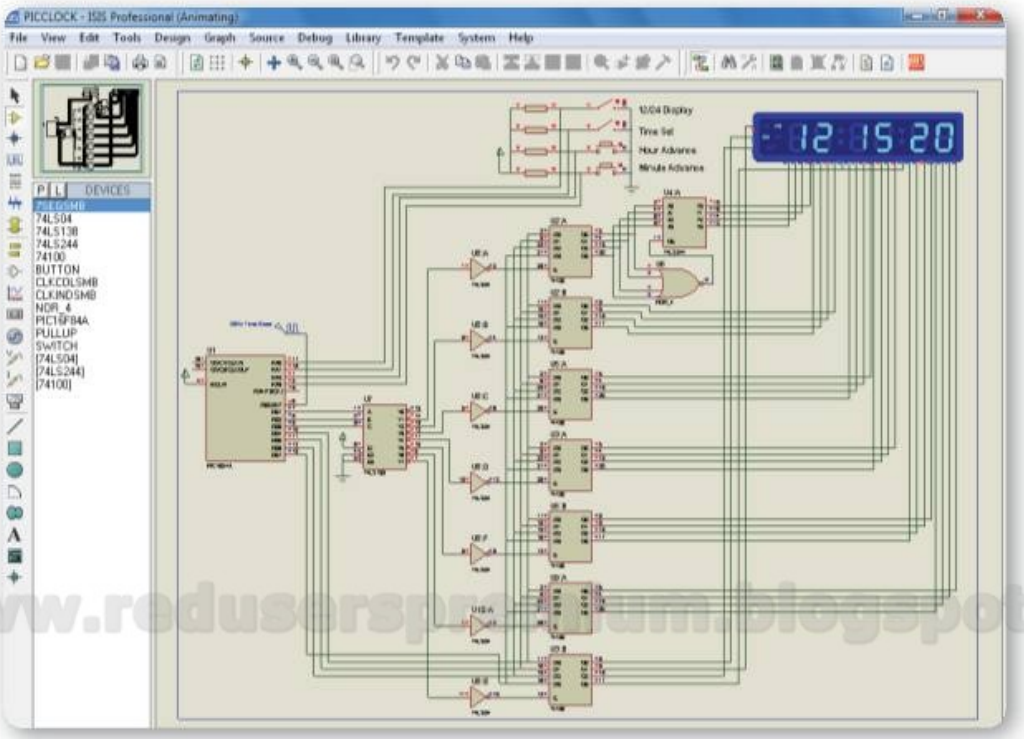


Figura 1. Proteus ofrece una amplia gama de herramientas de simulación en un entorno gráfico amigable y fácil de usar.

Aunque el nombre completo del programa es Proteus VSM, a partir de ahora nos referiremos a él solo como **Proteus**; de esta forma podremos identificarlo con mayor facilidad.

Si visitamos la página web **www.labcenter.com** de **Labcenter Electronics**, que es el desarrollador de Proteus, podremos descargar una versión demo del programa y, además, encontrar información acerca de las licencias, características, funciones, controladores y módulos extra, entre otros elementos.

Esta versión demo es completamente funcional, excepto por las siguientes limitaciones: no permite guardar nuestros circuitos si hemos creado uno desde cero, o si modificamos uno preexistente, no podremos almacenar los cambios realizados en él. Además, no es posible simular aquellos circuitos que incluyan microcontroladores.

Para instalar Proteus en nuestra computadora, simplemente debemos ejecutar el archivo de instalación y seguir los pasos tal como lo hacemos con cualquier otra aplicación en Windows. A partir de ahora, tomaremos una **versión completa** de Proteus para dar las explicaciones y ejemplos, es decir, una versión con licencia para usar todas sus funciones. Trabajaremos con la **versión 7.10**, que es la más reciente hasta el momento.

Las partes de Proteus

El programa cuenta con dos partes o componentes principales. Uno de ellos es el módulo **ISIS**, que es donde vamos a dibujar los diagramas de los circuitos electrónicos y, también, desde donde efectuaremos las simulaciones. Si es la primera vez que abrimos el módulo ISIS, después de instalar Proteus en el sistema, es posible que aparezca una ventana llamada **View Sample Designs**, que nos preguntará si queremos ver los diseños de ejemplo que se instalan junto con el programa.

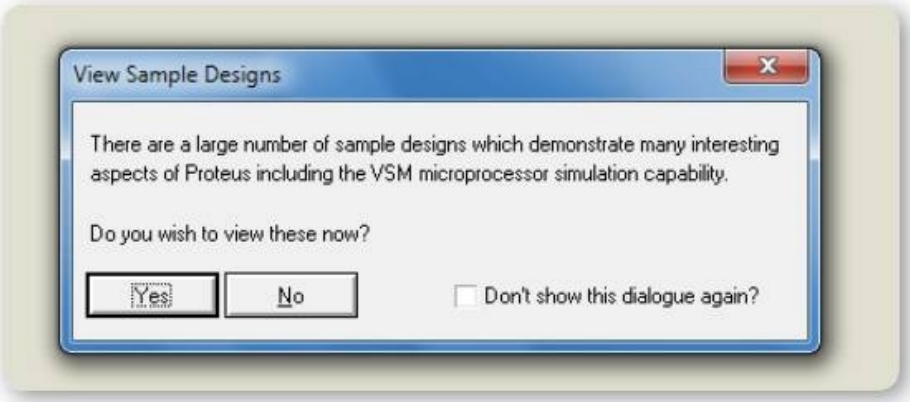


Figura 2. Al abrir Proteus desde el icono ISIS por primera vez, se nos preguntará si deseamos ver archivos de ejemplo.

Para ver los ejemplos, presionamos el botón **Yes**; de lo contrario, pulsamos **No**. Si marcamos la casilla **Don't show this dialog again?**, la ventana no volverá a aparecer; pero si después queremos acceder a los archivos de ejemplo, podremos hacerlo desde el menú denominado **Help/Sample Designs**.

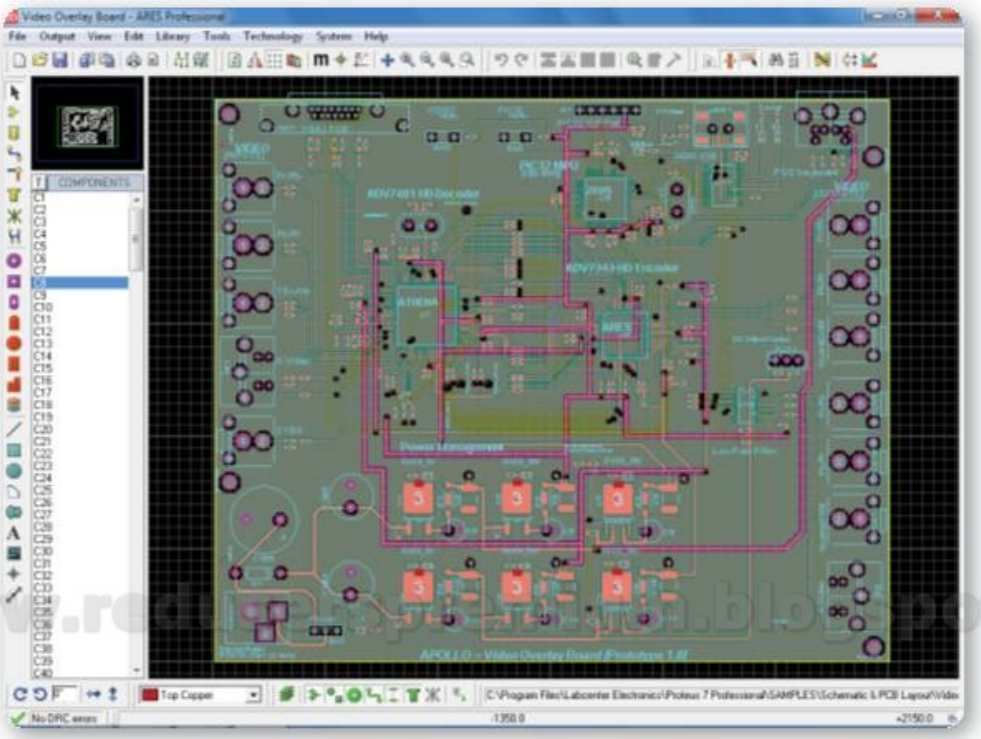


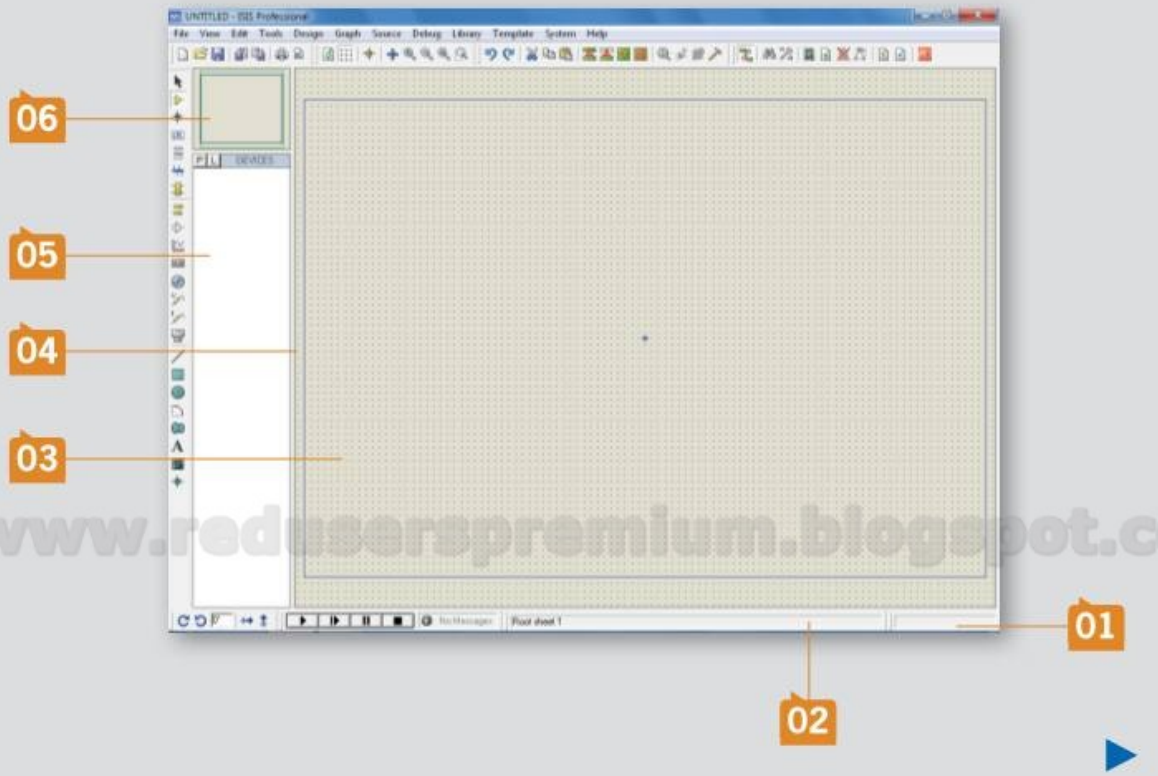
Figura 3. El módulo ARES permite el diseño de circuitos impresos donde construiremos los circuitos de forma física.

Además de ISIS, Proteus cuenta con otro módulo, llamado **ARES**, que es donde se diseñan las placas de **circuito impreso** (PCB) para el armado de los circuitos. Se puede importar un circuito dibujado en ISIS directamente al módulo ARES para diseñar fácil y rápidamente un circuito impreso para él.

La interfaz de ISIS

En principio, el módulo ISIS es un espacio para dibujar los diagramas de nuestros circuitos. Al abrir Proteus desde el icono ISIS, veremos la pantalla que se explica en la siguiente **Guía visual**.

GV: LA INTERFAZ DE ISIS



01 **VENTANA DE EDICIÓN:** es la ventana donde dibujaremos los circuitos electrónicos colocando los componentes, interconectándolos, y agregando también otros instrumentos y herramientas.

02 **BORDE DE HOJA:** la línea de color azul determina el límite de la hoja de trabajo; debemos colocar el circuito dentro de ella.

03 **SELECTOR DE OBJETOS:** en esta ventana aparecerán los componentes y otros elementos disponibles, dependiendo del modo seleccionado, y desde allí podremos elegirlos para colocarlos en el diseño.

04 **VENTANA DE VISTA PREVIA:** esta pequeña ventana nos ofrece una vista previa del circuito o de los elementos que vayamos a colocar en el diseño; más adelante veremos detalles sobre su uso.

05 **BARRA DE COORDENADAS:** en esta barra podemos ver las coordenadas donde se encuentra el cursor en todo momento, mientras lo desplazamos por la ventana de edición.

06 **BARRA DE ESTADO:** aquí aparece información útil de lo que tengamos seleccionado o del elemento sobre el cual se encuentre el cursor del mouse en ese momento.

La hoja de trabajo

En la ventana de edición tenemos dos elementos principales: la **hoja de trabajo** y **la rejilla**. La hoja de trabajo está delimitada por

NO ESTUDIAREMOS ARES

Hemos mencionado que Proteus cuenta con dos módulos principales: ISIS y ARES. En este último se diseñan placas de circuito impreso o PCB. En esta obra solo nos referiremos al módulo ISIS y a la simulación de circuitos en él. No estudiaremos ARES debido a que la extensión de la obra no lo permite.

el cuadro de color azul. Cuando accedemos a ISIS desde su icono, es decir, sin abrir ningún circuito, por defecto la ventana de edición está vacía, y el tamaño de la hoja de trabajo es de 10 por 7 pulgadas.

Si necesitamos cambiar el tamaño de la hoja de trabajo, podemos hacerlo mediante el menú **System/Set Sheet Sizes...**. Al seleccionar esta opción, aparece una ventana con el título **Sheet Size Configuration**, donde podemos elegir un tamaño de la lista o uno personalizado, llamado **User**. En realidad, es posible seleccionar un tamaño de la lista y luego ajustarlo a la medida que deseamos.

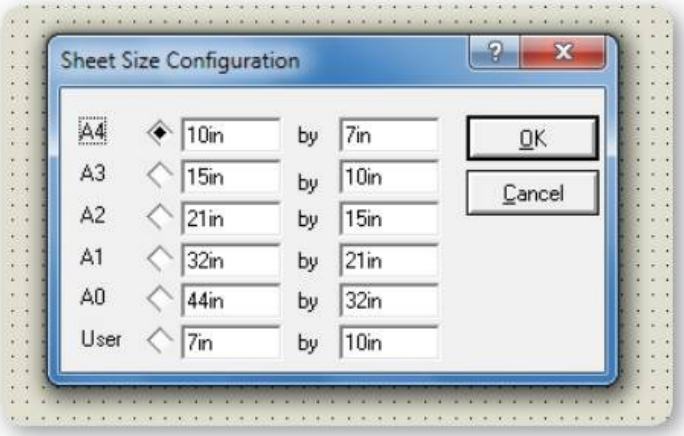


Figura 4. **`Sheet Size Configuration`** ofrece cinco tamaños de hoja, desde A4 a A0, y un tamaño definido por el usuario.

También podemos utilizar una plantilla diferente para la hoja de trabajo; si vamos al menú **File/New Design...**, aparecerá una lista de las plantillas que se instalan con Proteus.

EL BORDE DE LA HOJA DE TRABAJO

La línea de color azul delimita la hoja de trabajo en la ventana de ISIS, pero es solo una referencia. En caso de imprimir el circuito esta línea no se imprimirá ni se mostrará en la imagen guardada. Para que aparezca, colocamos un cuadro con elementos gráficos.

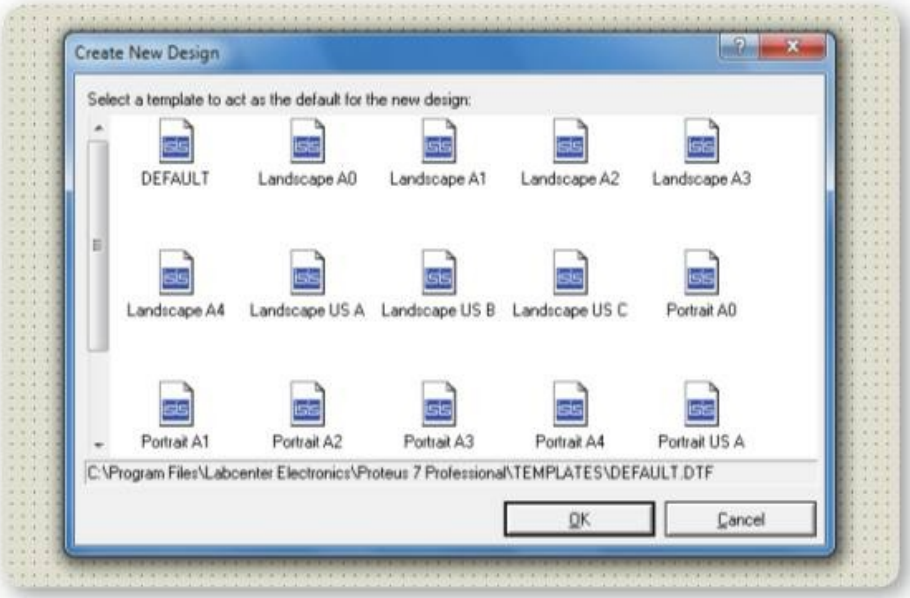


Figura 5. Podemos elegir una plantilla de la lista, incluyendo la que viene por defecto (**DEFAULT**).

Las plantillas adicionales contienen un **marco** con coordenadas y un **cuadro de datos** en la parte inferior derecha, que incluye la fecha, la hora y el nombre del archivo, entre otros datos.

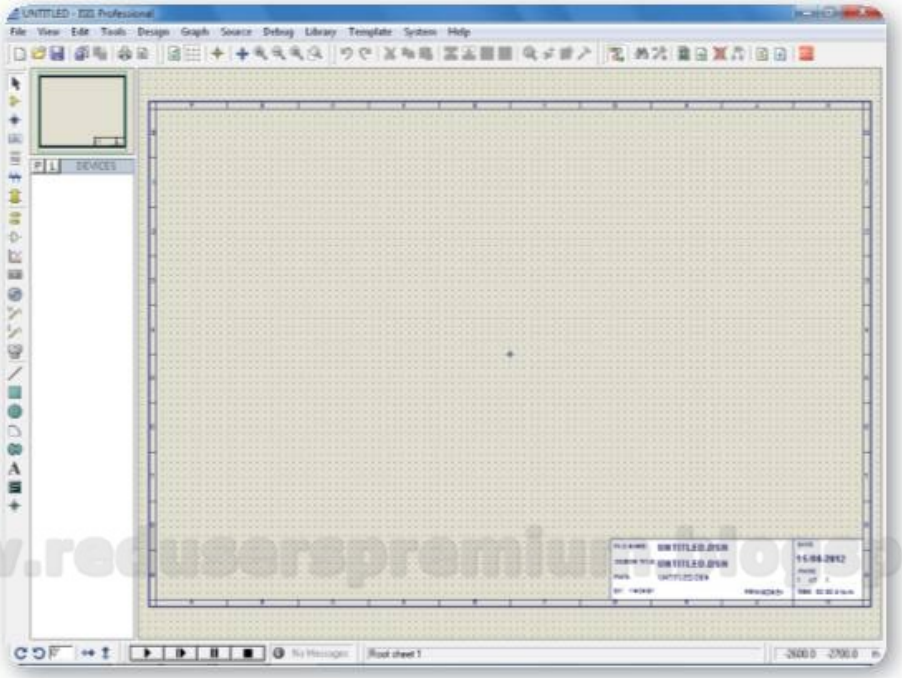


Figura 6. Un nuevo documento vacío en ISIS, esta vez, con la plantilla llamada **Landscape A4**.

La rejilla

Además de la hoja de trabajo, en la ventana de edición podemos ver una **rejilla de puntos o líneas**, que nos servirá como guía al momento de dibujar los circuitos; podemos desactivarla u ocultarla si lo deseamos. Los puntos o líneas de la rejilla tienen, por defecto, una separación fija entre sí. Entre dos puntos o líneas hay una distancia de un décimo de pulgada.

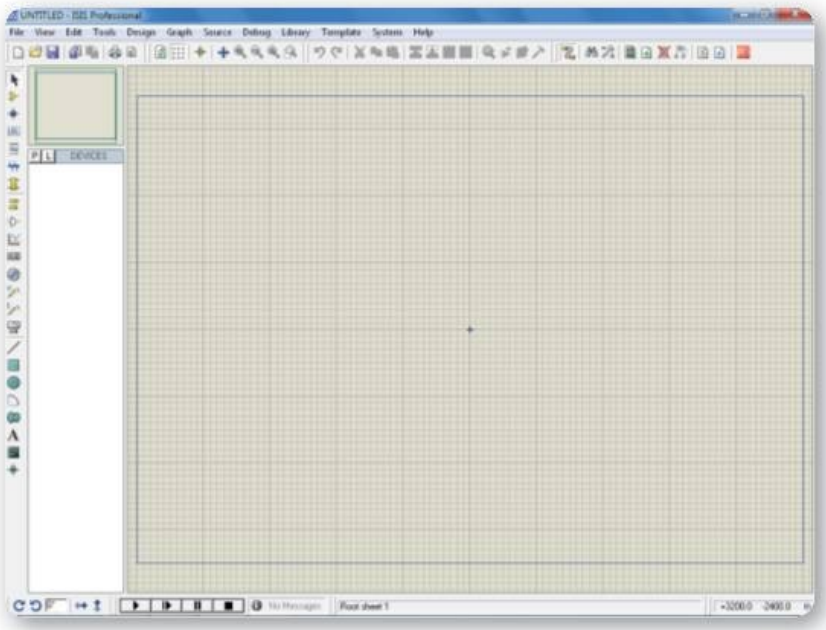
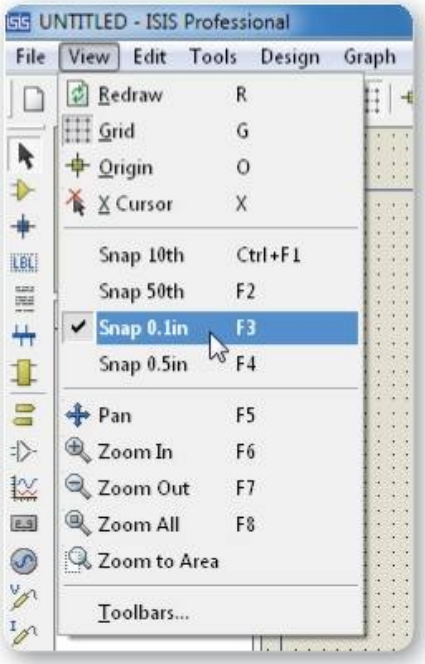


Figura 7. Además de la rejilla de puntos, las últimas versiones de Proteus poseen una rejilla conformada por líneas.

Podemos modificar la separación de los puntos o líneas de la rejilla, para ello vamos a **View** y elegimos entre las opciones:

PLANTILLAS: BORDE Y CUADRO DE DATOS

Si usamos una plantilla de las que tienen coordenadas en el borde de la hoja y un cuadro de datos, debemos tener en cuenta que están construidos a partir de elementos gráficos y texto y sí se imprimirán.



Snap 10th, **Snap 50th**, **Snap 0.1in** o **Snap 0.5in**, que cambian la división de cada pulgada en 100, 50, 10 o 2, respectivamente. A la derecha de cada opción, encontramos las teclas que podemos usar para la misma función, es decir, **el atajo de teclado**, no solo para estas opciones sino para muchas de las de los menús.

Figura 8. Desde el menú **`View`** cambiamos la configuración de la rejilla, y la desactivamos con la opción **`Grid`**.

Dibujar un circuito

En el siguiente **Paso a paso** veremos un ejemplo sencillo de cómo se dibuja un circuito en ISIS. Para hacerlo, partiremos de un circuito muy simple: un **oscilador** formado por una compuerta NOT con disparador Schmitt.

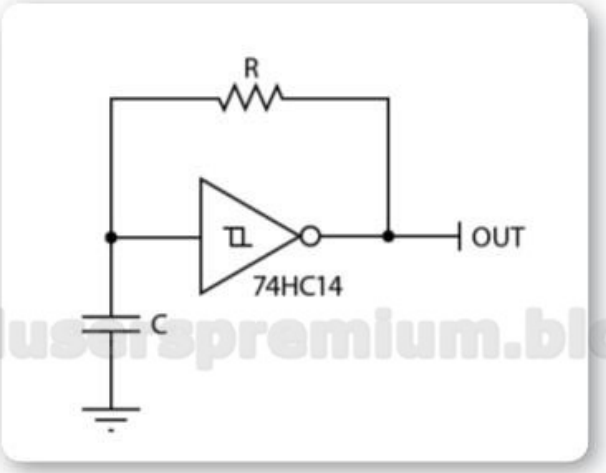
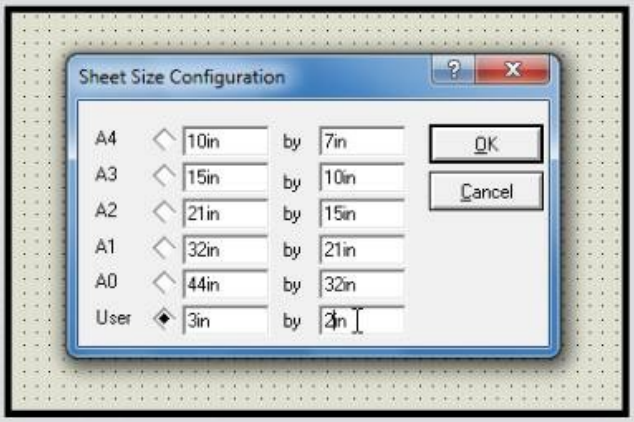


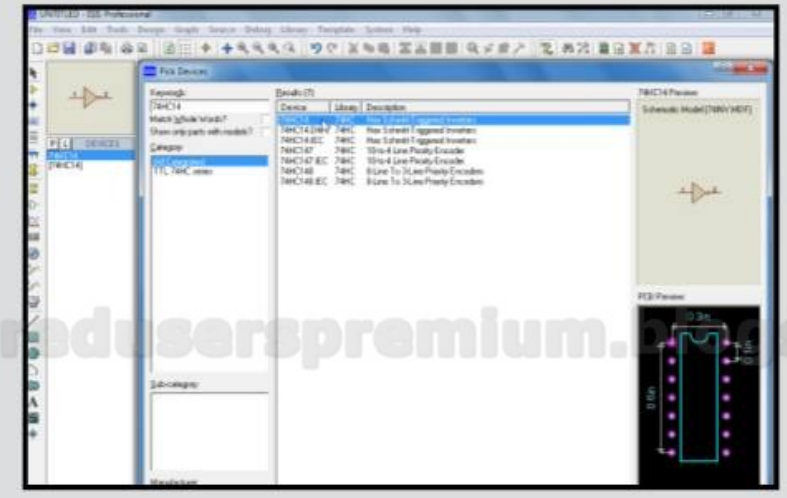
Figura 9. Tomaremos un circuito muy sencillo como ejemplo para aprender a dibujar en ISIS.

PaP: DIBUJAR UN CIRCUITO EN ISIS

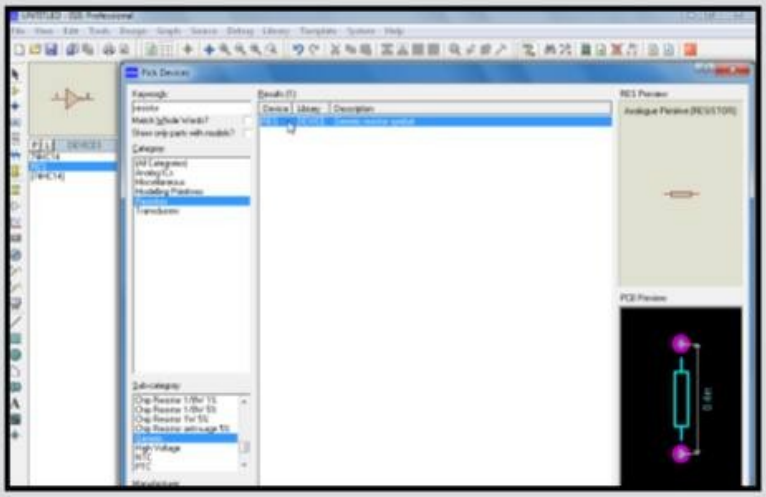
01 Abra ISIS desde su icono para tener un nuevo documento listo para dibujar el circuito, y cambie el tamaño de la hoja de trabajo desde el menú `System/Set Sheet Sizes...` a **3in x 2in**. Hará esto aquí para mejorar la visualización.



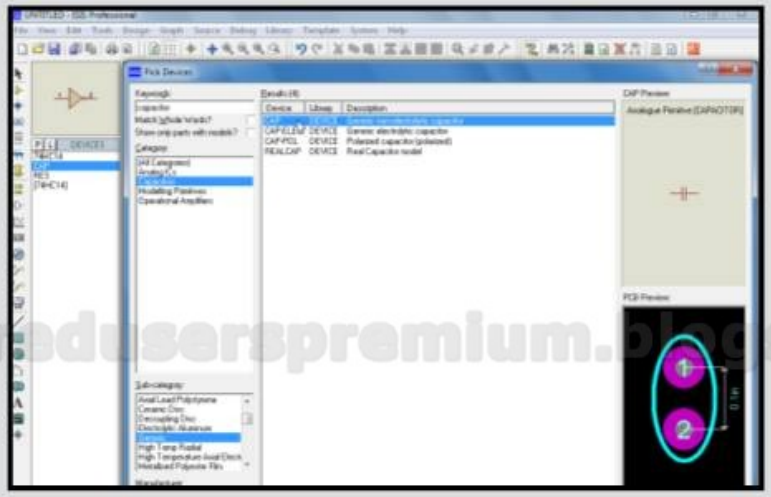
02 Vaya al modo `Componente` y presione el botón P para abrir la ventana `Pick Devices`, desde donde va a elegir los componentes necesarios. En este caso, busque y seleccione un 74HC14.



03 Ingrese en el cuadro de búsqueda la palabra **resistor**; en la ventana Category, seleccione Resistors y en Sub-category, busque y seleccione Generic.

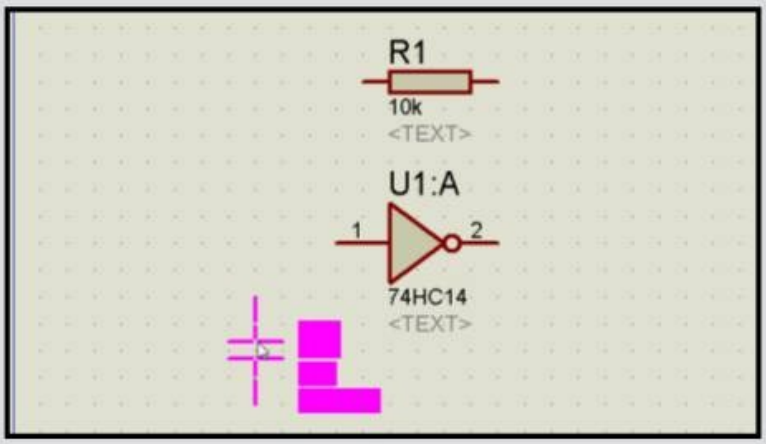


04 Ahora escriba en el cuadro de búsqueda la palabra **capacitor**; en la ventana Category, seleccione Capacitors y en Sub-category, busque y seleccione otra vez Generic. Una vez elegidos los componentes, puede cerrar la ventana Pick Devices.



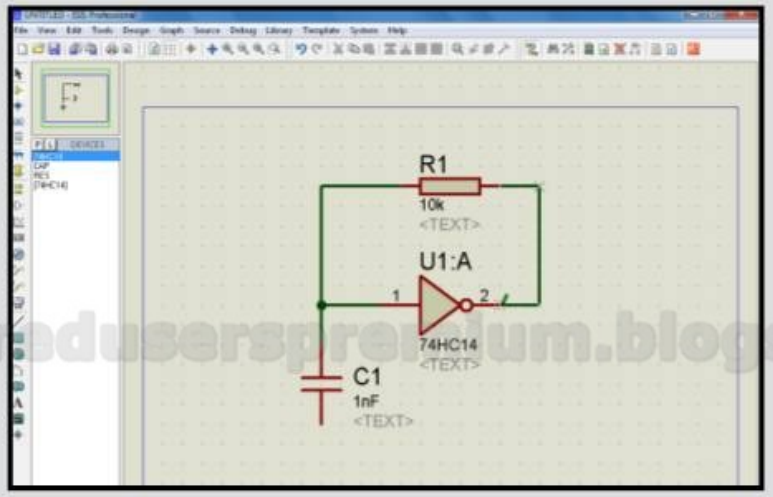
05

Coloque los componentes en la ventana de Edición, como se ve en el circuito de referencia. Tal vez tenga que rotar alguno de ellos para ubicarlo en la posición correcta. Como ya vimos, puede hacerlo antes de colocarlo, con las herramientas de Rotación o en el menú contextual, con el componente ya posicionado en la ventana de Edición.

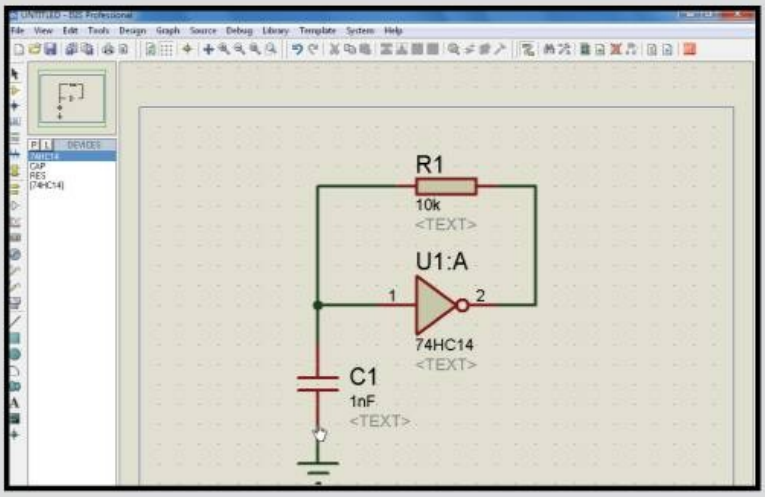


06

Desde el modo Componente, haga las conexiones correspondientes entre ellos.



07 Presione el botón para entrar en el modo `Terminales`; en la lista, seleccione la terminal GROUND, colóquela en el diseño debajo del capacitor y conéctela con él.



Hemos dibujado nuestro primer circuito en ISIS; ahora veamos cómo guardarlo en la computadora.

Guardar un diseño

Para guardar un archivo con el diseño que acabamos de dibujar, debemos ir al menú **File/Save Design As...** y, en la ventana **Save ISIS**

GRÁFICOS OPEN GL

Si recién instalamos Proteus, al abrir ISIS puede aparecer una advertencia que dice que nuestra tarjeta gráfica soporta gráficos **Open GL** con aceleración por hardware. Estos agregan funcionalidad y mejor apariencia a la interfaz de ISIS; para activarlos debemos ir al menú **System/Set Display Options**.

Design File, elegir una carpeta y colocar un nombre al circuito. Los diseños en ISIS se almacenan con la extensión **.DSN**. Si hacemos doble clic en cualquier archivo de este tipo, se abrirá automáticamente ISIS con el diseño que corresponde.

Otros tipos de archivos

Además del archivo **.DSN**, ISIS genera otros de forma automática y en la misma carpeta, al estar trabajando y guardar el diseño mediante el comando **Save Design**. Estos poseen la extensión **.DBK** y son archivos de respaldo. Por ejemplo, para un diseño de ISIS llamado **Circuito.dsn**, se genera un archivo **Last loaded Circuito.dbk** o **Backup Of Circuito.dbk**.

El **.DSN** contiene siempre la última versión guardada del diseño, en tanto que el **Backup Of** tiene una versión previa a la última guardada, y **Last Loaded**, la última versión del diseño abierta exitosamente. Cada vez que guardamos el diseño, los archivos de respaldo se actualizan de manera automática, para tener siempre un respaldo disponible en caso de necesitarlo. Como estos archivos tienen la misma extensión, siempre encontraremos solo uno de ellos a la vez.

Podemos usar los archivos **.DBK** para restaurar un diseño que se ha perdido o dañado. Para abrir estos archivos, vamos al menú **File/Open Design...** y, en la parte inferior, en la lista desplegable **Tipo**, elegimos la opción **Backup Design Files**, para que se muestren en la ventana de exploración. Durante la edición de un

OPEN GL CONTRA GDI

ISIS ofrece dos modos gráficos: **GDI**, en donde la apariencia y los efectos visuales están controlados por Windows; y **Open GL**, en donde los efectos están controlados por la tarjeta gráfica de la computadora. Este modo agrega mejoras, por ejemplo, animación al seleccionar los componentes.

circuito también pueden crearse archivos con el mismo nombre del diseño y con la extensión **.PWI**; estos guardan el estado de algunas ventanas auxiliares que se abren en ISIS y otras configuraciones.

La ventana de Vista previa

La pequeña ventana que está arriba del selector de objetos puede ayudarnos a navegar por los circuitos, especialmente si estos son grandes. Cuando estamos trabajando en la ventana de Edición, en la **ventana de Vista previa** tenemos una imagen en miniatura del diseño, que se corresponde con la hoja de trabajo completa. Además, visualizamos un **cuadro de color verde** que indica el área que aparece en la ventana de Edición. Esto nos sirve de referencia para ver en qué sección del diseño o de la hoja de trabajo nos encontramos.

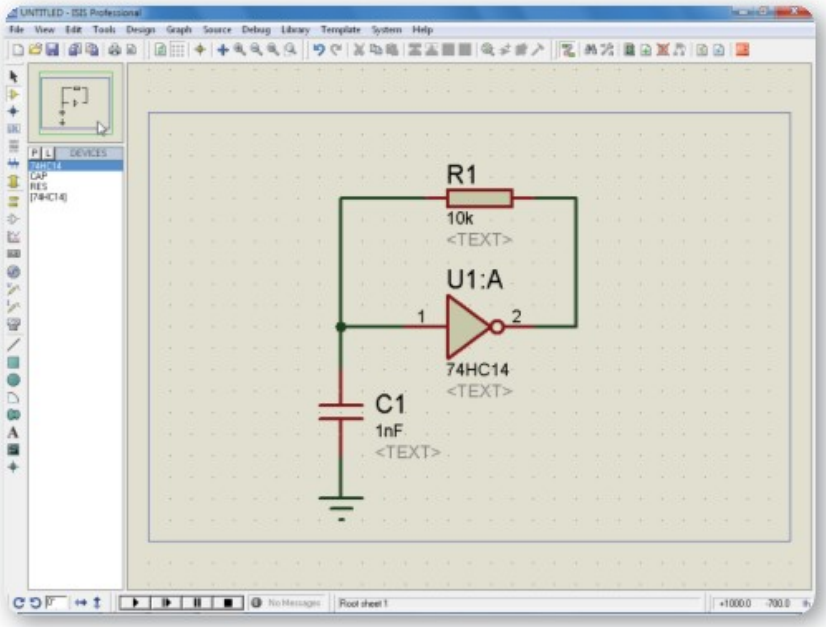


Figura 10. Podemos navegar por la hoja de trabajo desde la pequeña ventana de Vista previa.

Podemos hacer un clic en cualquier lugar de esta ventana de Vista previa, que nos llevará a esa área centrada en la ventana de Edición. Además, el cursor toma la forma de una cruz con cuatro

flechas, lo que determina que podemos mover el cuadro verde para desplazarnos a otra zona. Así, navegaremos por la hoja de trabajo desde la ventana de Vista previa, hasta que, al hacer un nuevo clic, la zona elegida quedará centrada en la ventana de Edición.

Componentes no utilizados

ISIS cuenta con una función para limpiar los diseños y retirar los componentes no utilizados. Se denomina **Tidy**, y podemos encontrarla en el menú **Edit**. Al seleccionarla, se nos pedirá confirmación para llevar a cabo la acción. Cuando presionamos el botón **OK** suceden dos cosas: primero, se eliminan todos los componentes en la ventana de Edición que se encuentran fuera de la hoja de trabajo; y segundo, se eliminan los componentes de la lista en el selector de objetos que no hayan sido utilizados en el diseño.

Figura 11. `Tidy` permite mantener nuestro diseño limpio y ordenado al eliminar los componentes que no se usan.

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

Esta operación es muy útil para limpiar nuestro diseño y el selector de objetos. Si hacemos clic con el botón derecho del mouse sobre el selector de objetos, mientras estamos en el modo Componente, también obtendremos la opción **Tidy**. En este caso, al elegirla, solo se borrarán los objetos no usados en el selector de objetos, pero los componentes que estén fuera de la hoja de trabajo en la ventana de Edición no se eliminarán.

RESUMEN

En este capítulo inicial hemos aprendido a utilizar las herramientas principales de ISIS. Nos familiarizamos con su interfaz y conocimos las bases para dibujar circuitos electrónicos en Proteus. Esto es muy importante porque, para simular cualquier circuito, primero debemos dibujarlo. También es importante saber crear circuitos si solo queremos usar Proteus como herramienta para el dibujo de diagramas.

*02

Simulación en Proteus

En este capítulo, continuaremos estudiando algunas opciones de dibujo de diagramas electrónicos en el módulo ISIS. Además, comenzaremos a aprender cómo se simula un circuito electrónico en Proteus, y entenderemos las bases de la simulación analógica, digital y mixta.

Los componentes

Proteus cuenta con más de **34.000** componentes y partes, entre dispositivos genéricos, animados, componentes específicos, circuitos integrados, conectores, interruptores, herramientas de simulación, baterías y fuentes, microcontroladores, etcétera. Podemos elegir entre una gran lista de componentes para dibujar los circuitos; en la **Tabla 1** veremos un resumen de las principales categorías de componentes y su contenido.

PRINCIPALES COMPONENTES

▼ CATEGORÍA	▼ DESCRIPCIÓN
Analog ICs	Circuitos integrados analógicos (filtros, reguladores de voltaje, amplificadores, etc.).
Capacitors	Todo tipo de capacitores (cerámicos, electrolíticos, etc.).
CMOS 4000 series	Circuitos integrados de la serie 4000 CMOS.
Connectors	Todo tipo de conectores (de audio, USB, headers, etc.).
Data converters	Convertidores (A/D, D/A y sensores de temperatura, entre otros).
Diodes	Todo tipo de diodos (rectificadores, zener, etc.).
Electromechanical	Diferentes tipos de motores.
Inductors	Bobinas y transformadores.

PRINCIPALES COMPONENTES	
Microprocessor ICs	Microprocesadores, microcontroladores y periféricos.
Modelling primitives	Componentes genéricos primarios analógicos y digitales (compuertas, resistores, transistores, etc.).
Operational amplifiers	Todo tipo de amplificadores operacionales.
Optoelectronics	Optoelectrónica (LEDs, displays, etc.).
Resistors	Todo tipo de resistores, genéricos y específicos.
Switches and relays	Interruptores, relevadores y teclados.
Switching devices	Dispositivos de conmutación (DIACs, TRIACs, etc.).
Transducers	Transductores (sensores de presión, distancia, etc.).
Transistors	Todo tipo de transistores, genéricos y específicos.
TTL 74xxx	Familia de circuitos integrados de las series 74.

Tabla 1. Principales categorías de componentes disponibles en Proteus.

Componentes simulables y no simulables

Dentro de los componentes de Proteus, encontramos una gran cantidad de elementos que tienen un **modelo de simulación**, es decir, que se pueden simular, y otros que carecen de este modelo y solo pueden ser usados para dibujar diagramas. Los componentes principales fueron analizados en la Tabla 1.

Al navegar por las librerías de ISIS, dentro de la ventana **Pick Devices**, notaremos que, cuando seleccionamos un componente, aparecerá un texto en la parte superior de la ventana de Vista

previa de símbolo, que en algunos casos dice: **VSM DLL Model**, **Schematic model** o **SPICE model**, y en otros: **Analogue Primitive** o **Digital Primitive**. Esto indica que el componente se puede simular. En otros casos se mostrará la leyenda: **No Simulator Model**, para informar que ese componente no tiene modelo y no puede formar parte de una simulación.

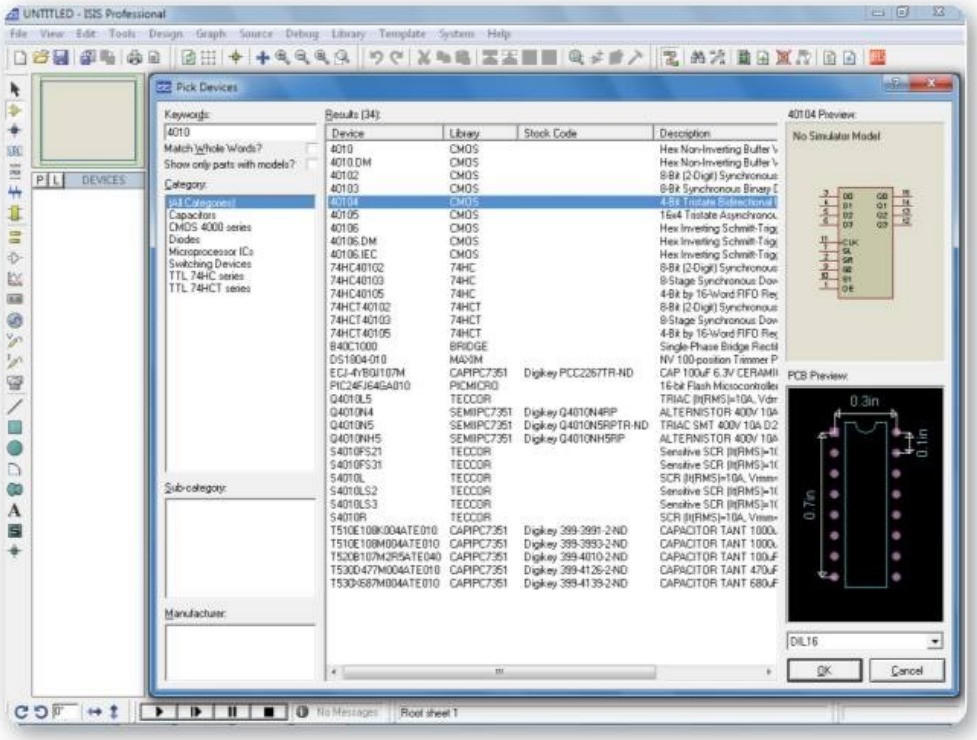


Figura 1. En la vista previa de símbolo se indica si un componente se puede simular o no.

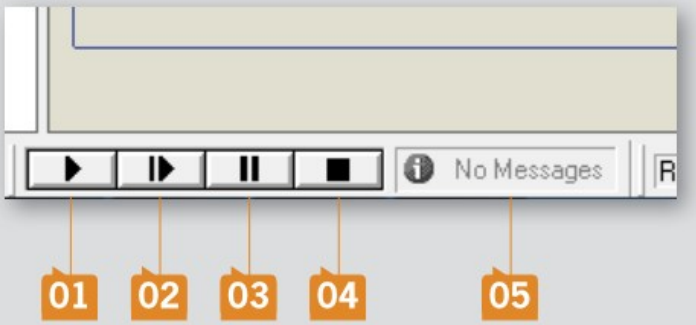
MODELOS PRIMARIOS

Existen diferentes tipos de componentes simulables, aunque la diferencia es interna y no debemos poner mayor atención en ella. Los modelos primarios (**Primitive**) son aquellos que están ya incluidos dentro del motor de simulación **ProSPICE** y son los más básicos, como: resistores, capacitores, diodos, compuertas, etcétera.

Los controles de simulación

Una vez que tenemos un circuito dibujado en ISIS, podemos simularlo. Para hacerlo, utilizaremos algunos botones que se encuentran en la barra de herramientas de Simulación y que explicamos en la siguiente **Guía visual**.

GV: LOS CONTROLES DE SIMULACIÓN



01 **PLAY (REPRODUCIR):** con este botón iniciamos la simulación del circuito que tenemos dibujado en ISIS.

02 **STEP (PASO):** permite ejecutar la simulación por pasos. Cada vez que lo presionamos, la simulación avanza un tiempo determinado y se pone en pausa automáticamente.

03 **PAUSE (PAUSA):** con este botón podemos pausar una simulación que se está llevando a cabo.

04 **STOP (DETENER):** detiene la simulación, sin importar si está corriendo o en pausa.

05 **LOG MESSAGES (MENSAJES):** en este cuadro encontraremos mensajes importantes acerca de la simulación. Podemos hacer un clic en él para que se abra un informe detallado acerca de la simulación en curso o la última realizada.

La primera simulación

Con lo que hemos aprendido hasta ahora, podemos comenzar a simular un circuito. El simple hecho de dibujar un circuito en ISIS nos permite simularlo, con tan solo hacer un clic en el botón **Play** (reproducir) de la barra de Simulación.

Al presionar **Play**, comenzará la simulación en forma continua, es decir que el oscilador empezará a funcionar.

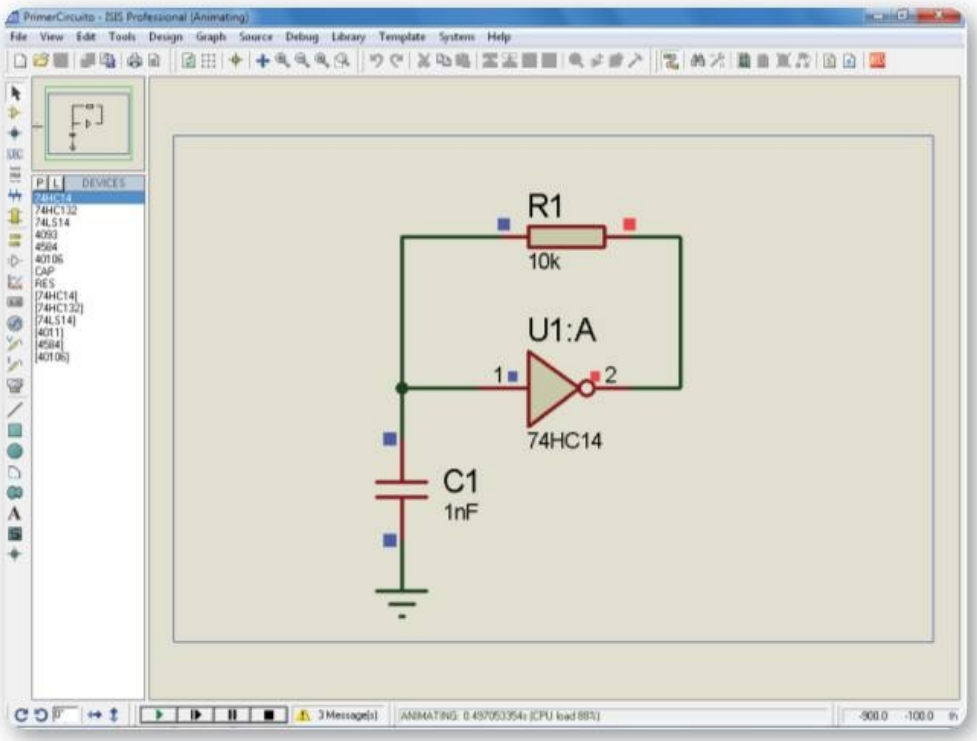


Figura 2. Podemos apreciar algunos detalles especiales en la interfaz de ISIS al momento de simular un circuito.

LOS BOTONES DURANTE LA SIMULACIÓN

Algunos de los botones de las diferentes barras de herramientas cambiarán y se mostrarán en color gris durante la simulación. Esto se debe a que esas funciones o herramientas no pueden utilizarse mientras una simulación está corriendo. Si necesitamos usarlos, debemos detener la simulación.

En la ventana de ISIS, observaremos algunos cambios. El botón **Play** se muestra en color verde, indicando que la simulación está corriendo. En la barra de título, aparece el nombre del archivo y, entre paréntesis, la palabra **Animating**. En la barra de estado también encontramos **ANIMATING** y un reloj que cuenta el tiempo que ha pasado de la simulación (por defecto, este correrá en tiempo real); además, se informa el porcentaje de uso de la CPU (**CPU Load**). En el cuadro de mensajes de la barra de Simulación veremos un símbolo y un número de mensajes; hablaremos de esto más adelante.

En el circuito aparecerán unos pequeños cuadrados de colores **azul**, **rojo** y **gris**; se trata de los niveles lógicos que tenemos en las diferentes partes del circuito. Notaremos que los cuadrados cambian de color constantemente, para indicar que el oscilador está funcionando. Por ahora, esto es lo que podemos observar en la simulación, pero poco a poco conoceremos más herramientas de visualización y análisis que nos ampliarán en gran medida la comprensión de las simulaciones. Para detener la simulación presionamos el botón **Stop**.

El informe de simulación

En el **cuadro de mensajes** ubicado en la barra de herramientas de Simulación podemos ver un reporte con los detalles del proceso. Cuando abrimos un circuito o creamos uno nuevo, en el cuadro encontraremos la leyenda **No Messages** (no hay mensajes). Al correr una simulación, la leyenda cambiará e indicará un número definido de mensajes. La cantidad depende de cada simulación en particular y de lo que suceda en ella.

Si hacemos un clic con botón principal del mouse en el cuadro cuando indica algún número de mensajes, se abre la ventana **SIMULATION LOG**, en donde podemos leer los mensajes en detalle. Es posible acceder a este informe tanto si la simulación está corriendo como si no lo está. Cuando está detenida, el reporte nos dará información del último proceso llevado a cabo.

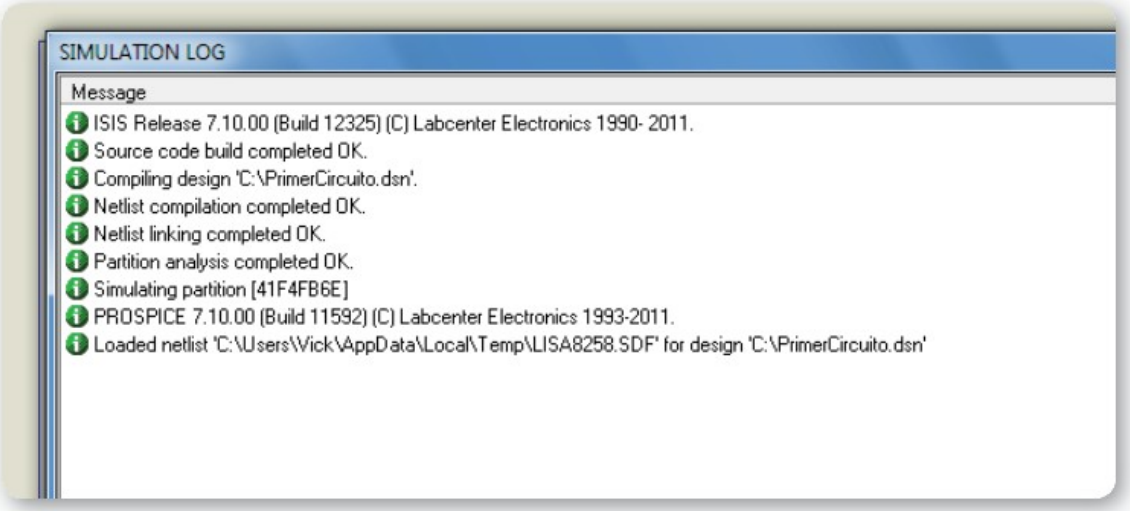


Figura 3. El informe muestra los detalles de lo que sucede al correr una simulación.

Si la simulación se desarrolla de manera exitosa, en el cuadro de mensajes veremos un icono de color verde con una letra **i** (**information**). En cambio, si hay alguna advertencia sobre algo que no funciona bien, aparecerá un triángulo de color amarillo con un signo de exclamación (**!**), para indicarnos que hay mensajes a los que debemos prestarles atención.

Por último, tengamos en cuenta que si al intentar correr una simulación ocurre algún error que impide que esta se inicie, o si al estar corriendo no puede continuar, la simulación se detendrá y se abrirá automáticamente la ventana con el informe, en el cual se indicarán los errores mediante un icono con una **x** y un texto explicativo, en color rojo.

FORMATOS DE VALORES

Para definir los valores de los componentes (como resistores, capacitores, inductores, etcétera), podemos usar prefijos que indiquen valores muy grandes o muy pequeños: p=pico, n=nano, u=micro, m=mili o k=kilo.

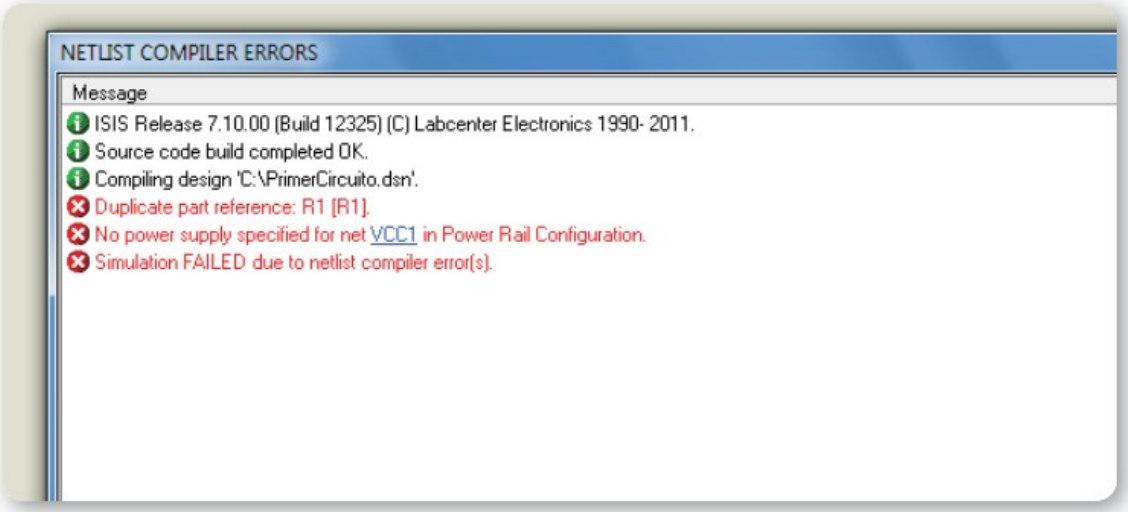


Figura 4. Los textos en rojo y con una **x** indican una falla por la cual la simulación no puede llevarse a cabo.

Con esta información sabremos cuáles son los errores y podremos tratar de corregirlos para que la simulación funcione correctamente.

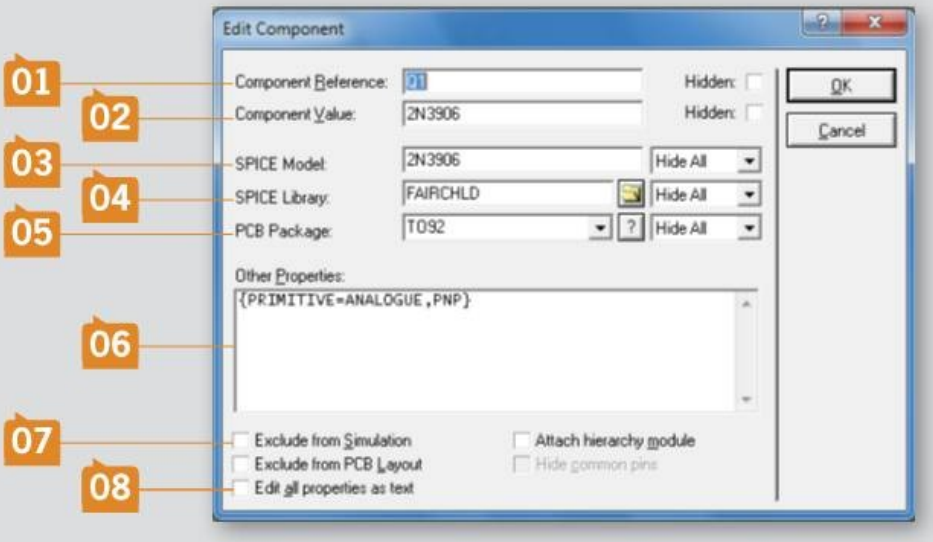
Propiedades de los componentes

Cada componente tiene sus propiedades específicas, a las cuales podemos acceder de dos maneras. Una es hacer clic derecho del mouse sobre un componente colocado en la ventana de Edición y, desde el menú contextual, elegir la opción **Edit Properties**. La otra forma es dar un doble clic sobre el componente (o solo un clic si ya está seleccionado), para que se abra la ventana **Edit Component**.

SCHEMATIC MODELS

Los **Schematic Models** son componentes complejos que se forman por elementos primarios para emular el funcionamiento de un componente específico, como un circuito integrado.

GV: PROPIEDADES DE LOS COMPONENTES



01 **COMPONENT REFERENCE (IDENTIFICADOR DEL COMPONENTE):** muestra la identificación del componente en el circuito. En este caso, como es un transistor, la referencia es **Q**, pero esto depende del componente de que se trate, y podemos cambiarla si lo deseamos. Si marcamos la casilla **Hidden** (oculto) de la derecha, esta propiedad no aparecerá en el diagrama junto al componente.

02 **COMPONENT VALUE (VALOR DEL COMPONENTE):** indica el valor, el nombre o el número de parte del componente; en este caso es **2N3906**, pero también podemos cambiarlo si lo deseamos. Al igual que la referencia, es posible ocultar este valor al marcar la casilla **Hidden**.

03 **SPICE MODEL (MODELO SPICE):** en este campo aparece el modelo de simulación. Por lo general, no debemos cambiar este dato, ya que identifica el modelo matemático que usará Proteus para simular el componente. También podemos mostrar u ocultar todo este campo o parte de él.

04 **SPICE LIBRARY (LIBRERÍA SPICE):** en esta sección se indica la librería a la que pertenece el componente. Al igual que el modelo, no debemos cambiar este valor.

05 **PCB PACKAGE (EMPAQUE PARA CIRCUITO IMPRESO):** aquí veremos el empaque del componente; esto es especialmente útil si vamos a diseñar un circuito impreso para nuestro diagrama. En algunos casos, tendremos más de un empaque que podremos elegir de la lista desplegable.

06 **OTHER PROPERTIES (OTRAS PROPIEDADES):** presenta propiedades adicionales que puede tener el componente. Como se muestran en formato texto, podemos editarlas, borrarlas o agregar otras, si lo necesitamos.

07 **EXCLUDE FROM SIMULATION (EXCLUIR DE LA SIMULACIÓN):** si seleccionamos esta casilla, el componente no se incluirá en la simulación, y será como si no estuviera presente en el circuito.

08 **EDIT ALL PROPERTIES AS TEXT (EDITAR TODAS LAS PROPIEDADES COMO TEXTO):** si marcamos esta opción, tendremos la posibilidad de editar las propiedades como texto. Al hacerlo, algunas de las opciones aparecerán como texto en la ventana **`Other Properties`** y podremos modificarlas desde ahí.

Las propiedades de los diferentes componentes presentan un formato similar al mostrado anteriormente, aunque los campos dependen de cada caso en particular. Algunos podrán tener más elementos, en especial, los circuitos integrados y los microcontroladores, mediante los cuales es posible definir parámetros como: resistencia, capacitancia, voltajes, tolerancias, tiempos, frecuencias, etcétera. Conforme vayamos utilizando algunos de los componentes, veremos cómo manejar sus propiedades en cada caso.

La mayoría de estas propiedades serán útiles al momento de realizar la simulación, ya que se encargan de definir los valores y los parámetros de funcionamiento del componente. Algunas otras se utilizan solo para etiquetar los componentes dentro del circuito; por ejemplo, la referencia que se mostrará en el diagrama para identificar al componente en él.

Editar las etiquetas de texto de los componentes

Es posible editar las etiquetas de los textos que acompañan a cualquiera de los componentes, por ejemplo, el **identificador** o el **valor**. Para hacerlo, debemos hacer doble clic sobre cualquier etiqueta y así abrir el cuadro de diálogo de edición.

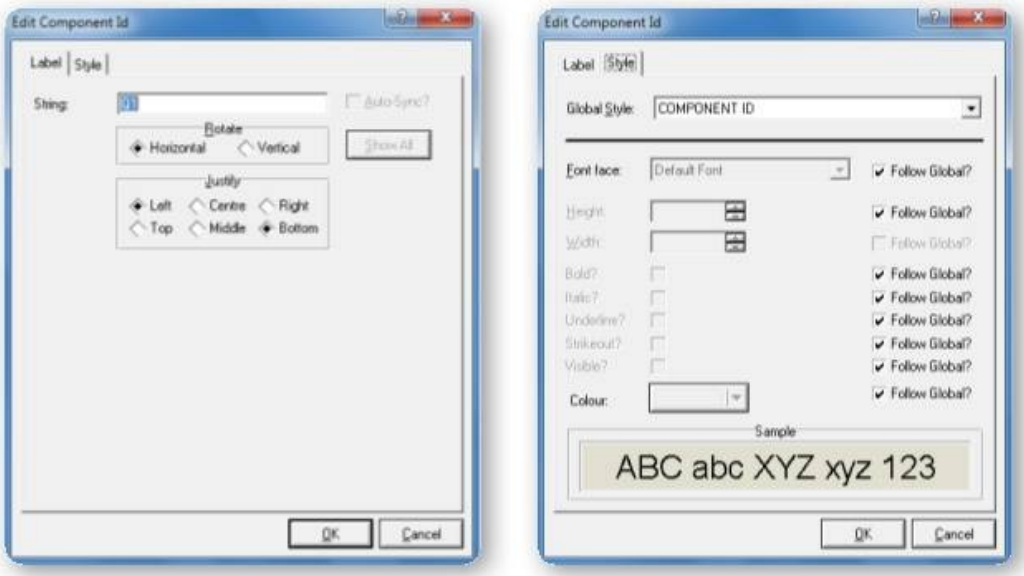


Figura 5. Podemos editar las etiquetas de texto, por ejemplo, las propiedades del identificador de componente.

En la pestaña **Label**, editamos el texto dentro del campo **String**, lo ubicamos en forma horizontal o vertical con la opción **Rotate**, y cambiamos la justificación en la sección **Justify** para alinearlo a la derecha, izquierda, centro, etcétera.

En la pestaña **Style**, cambiamos la apariencia del texto. Desde el cuadro **Global Style**, podemos elegir un **estilo global**, que agrupa distintos estilos definidos en Proteus para diferentes elementos. El estilo por defecto es el que corresponde al identificador de componente: **COMPONENT ID**. Debajo encontramos varias opciones, y a la derecha de cada una, el texto **Follow Global?**, que nos pregunta si seguiremos el estilo global, correspondiente a la selección de arri-

ba. Si desmarcamos alguna de las opciones, habilitamos la edición o elección de ese estilo: fuente (**Font face:**), altura (**Height:**), ancho (**Width:**), negrita (**Bold?**), itálica (**Italic?**), subrayado (**Underline?**), tachado (**Strikeout?**), visibilidad (**Visible?**) o color (**Colour:**) del texto.

Ejemplo de edición de propiedades

Ahora que sabemos cómo editar las propiedades de los componentes, haremos un ejercicio, tomando otra vez el ejemplo del oscilador de la primera simulación. Al dibujar el oscilador por primera vez, dejamos los valores por defecto del resistor y del capacitor: **R=10kohm** y **C=1nF**. Con estos valores, el oscilador generará una señal de **120 kHz**, lo cual es muy alto y no podremos apreciar en detalle. Cambiaremos los valores a **R=15kohm** y **C=47uF**; así el oscilador tendrá una frecuencia aproximada de **1 Hz** y podremos ver su funcionamiento con más claridad.

Las terminales

El módulo ISIS cuenta con una función que resulta muy útil en muchos casos: **las terminales**. Con ellas podemos identificar o marcar puntos en un circuito, e incluso, hacer conexiones entre diferentes puntos sin tener que trazar líneas de conexión. Para acceder a las terminales disponibles, pulsamos **Terminals Mode**.

NIVELES LÓGICOS EN EL DESTELLADOR

En la simulación del destellador vemos que, a pesar de tener activadas todas las opciones de animación, los cuadros que indican niveles lógicos no aparecen al simular. Esto se debe a que el circuito no es digital.

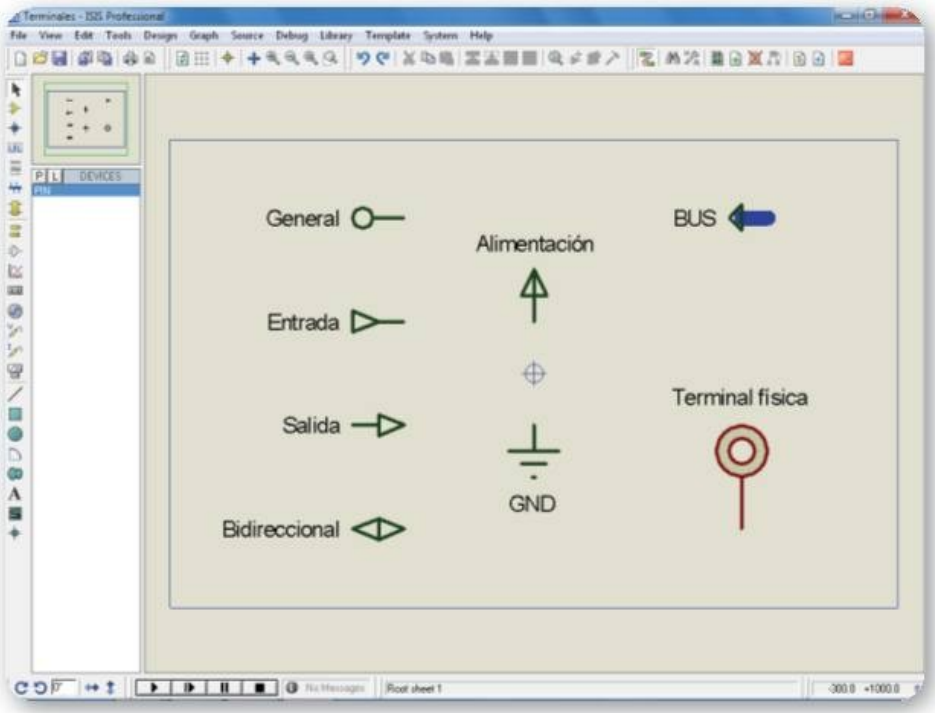


Figura 6. En el módulo ISIS podemos utilizar siete terminales lógicas y una física.

Las cuatro primeras de la lista tienen un uso general, y es posible emplearlas para realizar desde simples referencias en los circuitos hasta conexiones entre distintos puntos. Por ejemplo, la terminal **DEFAULT** (general) puede utilizarse para marcar un punto en el circuito; **INPUT** (entrada), para definir un punto de entrada; **OUTPUT** (salida), para indicar un punto de salida; y **BIDIR** (bidireccional), para señalar un punto que puede ser de entrada o salida. El uso de estas cuatro terminales depende de la decisión de cada usuario.

MODELOS DLL

Los modelos llamados **VSM DLL model** son componentes que tienen que ser programados externamente debido a su funcionamiento o a las características que debe tener en la simulación o animación; por ejemplo, un display **LM016L**. Se agregan a Proteus mediante librerías **.DLL**.

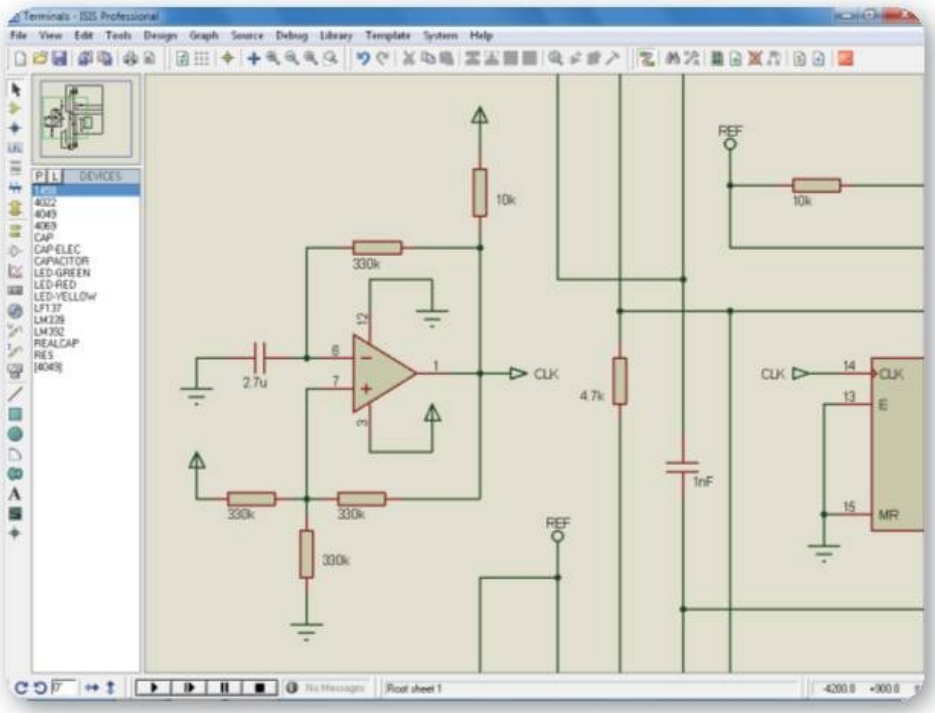


Figura 7. Algunas variantes de aplicación de las diferentes terminales del módulo ISIS.

Las terminales **REF** marcan un punto en común, es decir, como si hubiera una línea de conexión entre los dos puntos. El mismo caso es para las terminales **CLK**, aunque en la imagen vemos una terminal de salida y otra de entrada; esto no importa, ya que al tener el mismo nombre, ambas indican conexión entre dos puntos.

La característica que diferencia una terminal de otra es, precisamente, su **nombre** o **etiqueta**. Es por eso que cada vez que usemos una terminal debemos darle una denominación. Si hacemos doble

TERMINALES LÓGICAS Y FÍSICAS

Las terminales disponibles en el modo de terminales son solo lógicas, es decir, no representan ningún componente o elemento físico en el circuito. Si necesitamos colocar una terminal física, podemos usar el componente **PIN**, que encontramos en la ventana **Pick Devices**.

clic en cualquier terminal colocada en un diseño, se abre la ventana **Edit Terminal Label**. Como no son componentes, las terminales carecen de propiedades; solo tienen etiquetas.

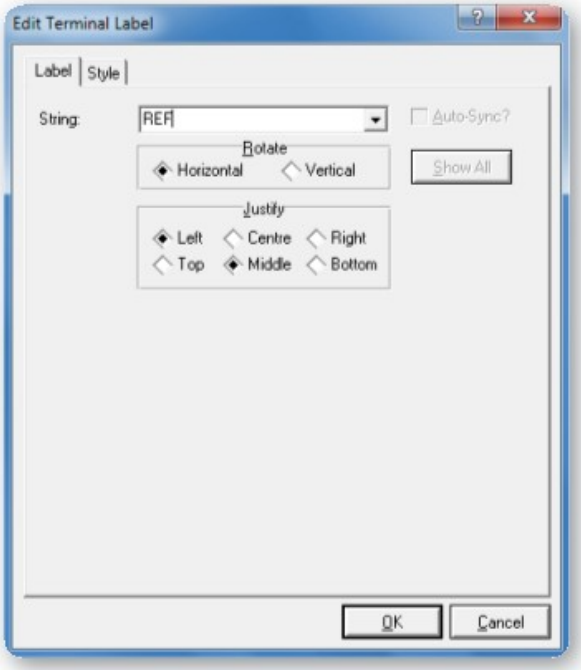


Figura 8. Siempre debemos dar un nombre a las terminales para identificarlas entre las demás.

En el campo **String** tenemos que escribir el nombre para cada terminal. Este campo es una lista desplegable; por eso, si presionamos la flecha para abrir la lista, veremos los nombres de todas las demás terminales que ya tenemos en el circuito. En caso de que queramos unir dos terminales, podemos simplemente elegir un nombre de la lista en lugar de escribirlo. Las demás opciones de las pestañas **Label** y **Style** son idénticas a las de las etiquetas de los componentes.

Dos casos especiales son las terminales **GROUND** (tierra) y **POWER** (alimentación). La primera se usa como referencia o tierra, y es nombrada internamente de forma automática como **GND** o **VSS**. La terminal de alimentación, de la misma manera, recibe la denominación **VCC/VDD**. Así, queda conectada a la red **VCC** o **VDD** y genera el voltaje correspondiente a ella, que por defecto

es **5 V**. Podemos configurar las terminales **POWER** para crear nuevas líneas de alimentación o cambiar el voltaje que entregarán.

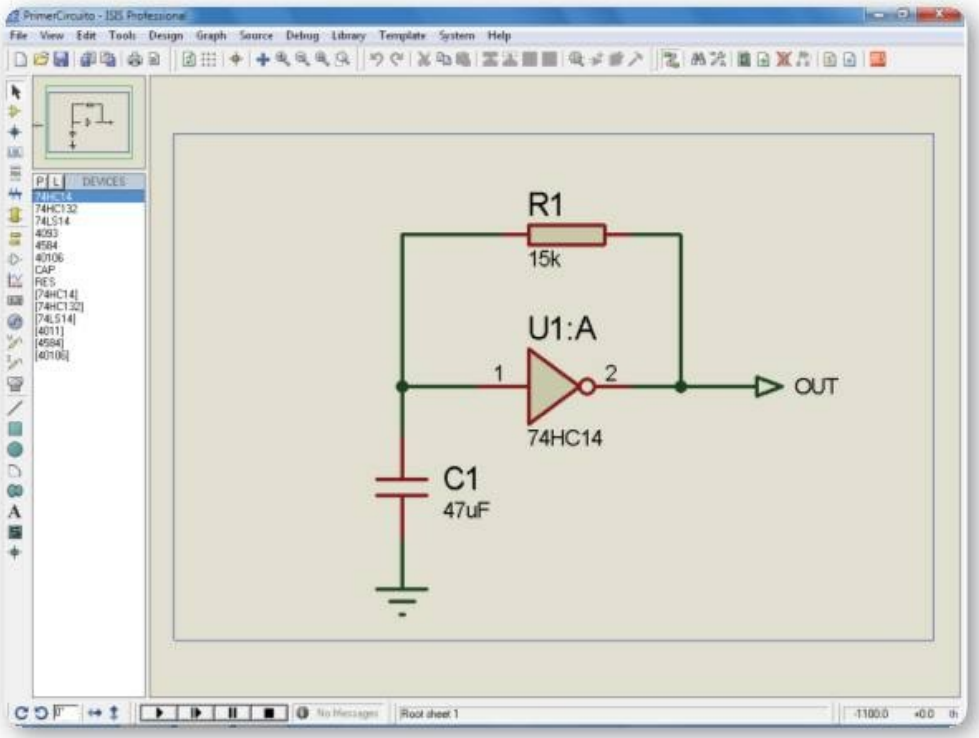


Figura 9. El circuito oscilador con nuevos valores de resistencia, capacitancia y una terminal de salida.

Podemos colocar una terminal de salida a nuestro diagrama del oscilador para completarlo, como en la imagen que tomamos de referencia para dibujarlo.

Identificar una red de conexiones

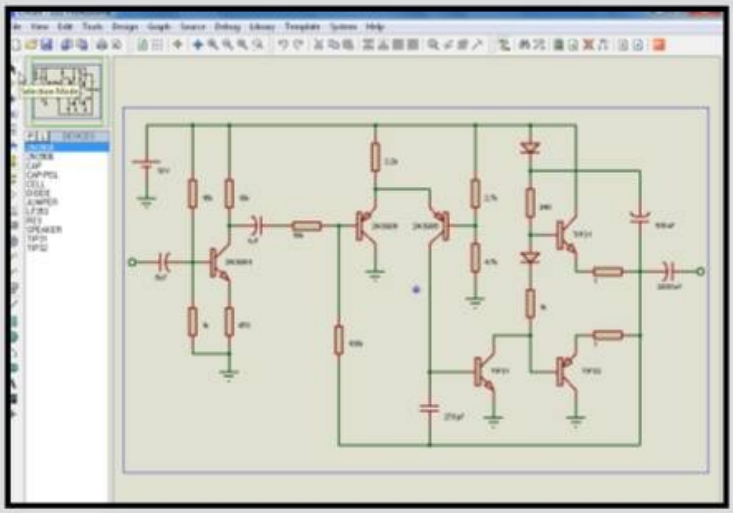
En los circuitos complejos es difícil realizar las conexiones y hay una mayor probabilidad de cometer errores.

En estos casos, podemos usar la función **Highlight net on Schematic**, muy útil para identificar cómo están realizadas las conexiones y observar si son correctas.

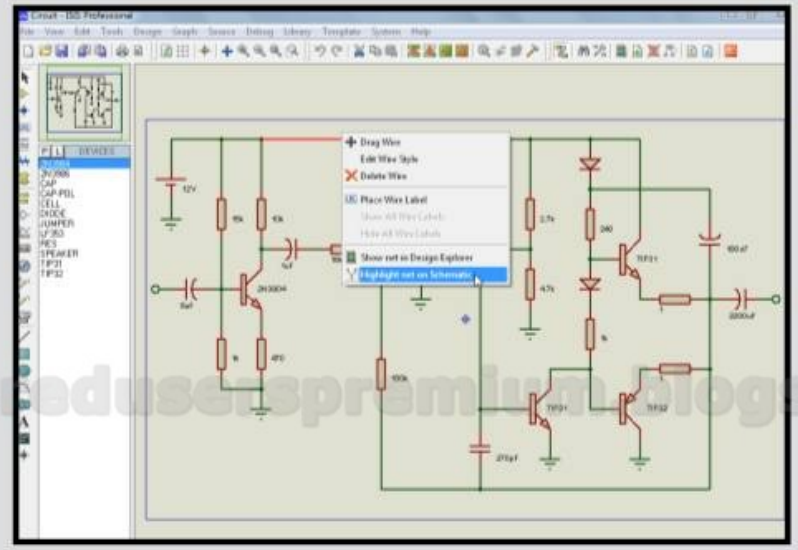
En el siguiente **Paso a paso** veremos en detalle la forma adecuada de realizar este procedimiento.

PaP: RESALTAR CONEXIONES

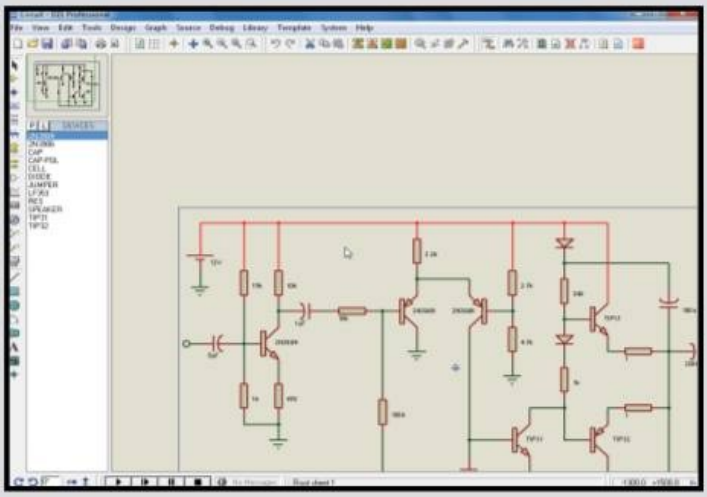
01 En un diseño que posea líneas de conexión, presione el botón para entrar en el modo de Selección (`Selection Mode`).



02 Haga un clic con el botón derecho sobre la línea de conexión en donde desea identificar la red de conexiones. Con esto, seleccionará el segmento de la línea y se abrirá un menú contextual.



03 En el menú, elija `Highlight net on Schematic,` la opción para resaltar la red de conexiones. Todas las conexiones que forman esta red quedarán marcadas en color rojo.



Con la función para identificar una red de conexiones veremos si las que hemos hecho están completas, y si todos los puntos están unidos en esa red o nodo. Esto es útil en circuitos de gran tamaño.

Simulación analógica

Veamos un primer ejemplo de un circuito analógico y su simulación en Proteus, usaremos un destellador con transistores.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

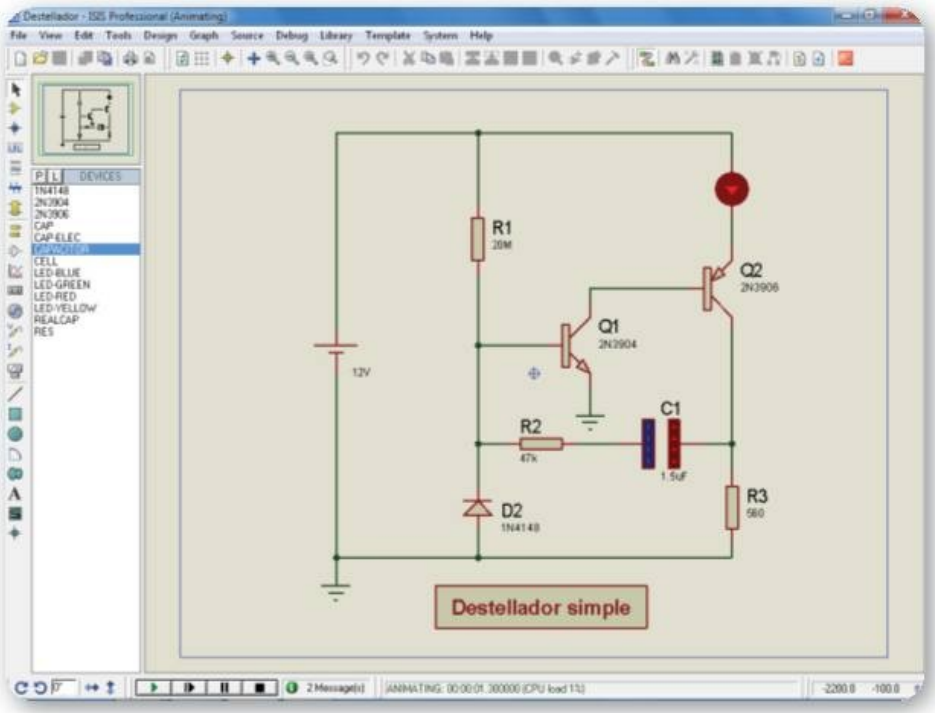


Figura 10. Un circuito destellador analógico simple, construido con solo dos transistores.

Este circuito está diseñado para hacer destellar un LED, que puede ser útil en los automóviles como indicación falsa de una alarma activada. Es fácil de construir y requiere muy poca potencia.

En este primer ejemplo hemos colocado un **capacitor animado** en el circuito, para observar su funcionamiento interno. Al correr la simulación, presionando el botón **Play** de la barra de herramientas de Simulación, veremos cómo el capacitor se carga y descarga durante el funcionamiento del circuito destellador, siendo parte fundamental en él.

EJEMPLOS DE USO

Debemos tener en cuenta que en la red es posible encontrar diversos archivos de ejemplo, principalmente, con la extensión **.DSN**. De esta forma podremos contar con ejemplos de uso de Proteus y trabajar sobre ellos para lograr resultados en menor tiempo.

Simulación digital

La simulación digital es más eficiente que la analógica. Aunque un circuito puramente digital sea muy complejo, será bien simulado, sin usar muchos recursos de procesamiento y corriendo en tiempo real. Proteus posee algunos componentes o herramientas que permiten simular los circuitos digitales de forma eficiente, rápida y fácil.

Sondas lógicas y estados lógicos

Dos herramientas que serán de suma utilidad son los **estados lógicos** y las **sondas lógicas**.

Podemos encontrarlas en la categoría **Debugging Tools** en la ventana **Pick Devices**, o también buscarlas por sus nombres: **LOGICSTATE**, **LOGICPROBE** o **LOGICTOGGLE**.

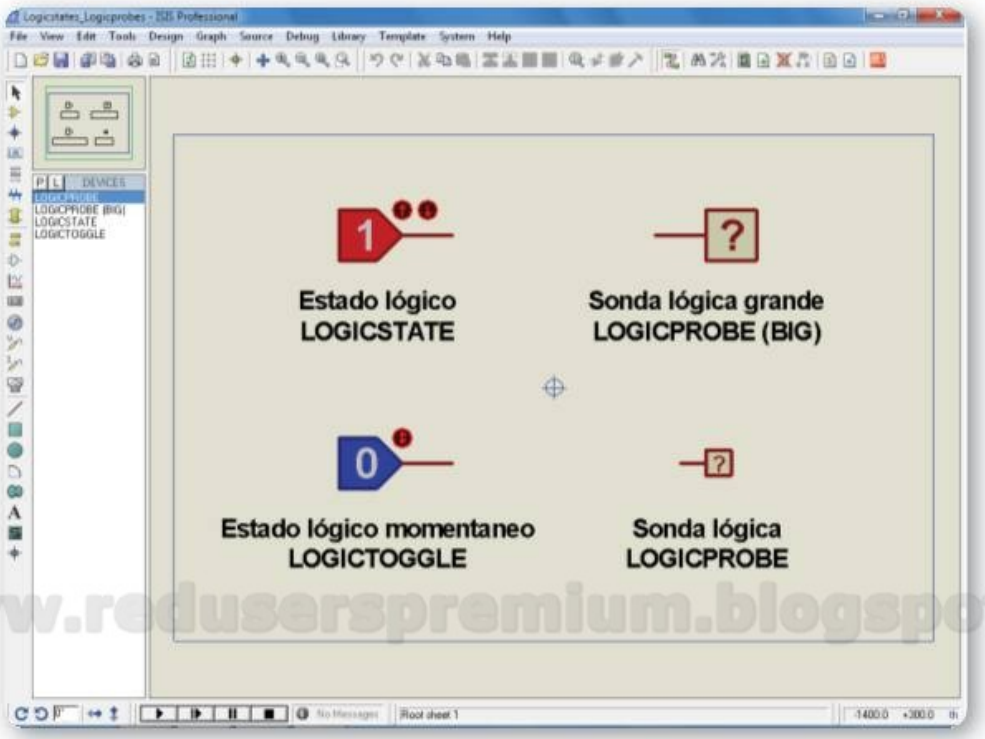


Figura 11. Las sondas lógicas analizan los niveles digitales, y los estados lógicos introducen niveles en el circuito.

Los estados lógicos nos permiten definir un nivel lógico en alguna parte de nuestro circuito, por ejemplo, en una entrada de una compuerta lógica. Mientras tanto, las sondas nos muestran el nivel lógico en algún punto, por ejemplo, en la salida de una compuerta. En las sondas lógicas se leerá el nivel actual como **0**, **1**, o **?**, si hay un nivel indefinido. Las sondas tienen dos tamaños, que podemos elegir: pequeño o grande, para una mejor visualización.

Para los estados lógicos, tenemos **LOGICSTATE**, que puede cambiar su estado en cualquier momento, haciendo un clic en el respectivo **actuador** (las flechas rojas dentro de un círculo) o en el cuerpo del estado lógico. El **LOGICTOGGLE** se mantendrá en un estado fijo, y si hacemos un clic sobre él, modificará su estado solo mientras mantengamos presionado el botón del mouse. Si usamos el actuador, cambiará el estado permanentemente hasta que volvamos a hacer un clic sobre él o en el cuerpo del estado lógico. Si deseamos cambiar el estado inicial, es decir, el estado permanente, debemos entrar en sus propiedades y elegir **Low** o **High** en la lista **Initial State**. Los estados y sondas lógicas se colocan y conectan como cualquier otro componente en el circuito.

Ejemplo

Veamos un ejemplo del uso de los estados lógicos y las sondas lógicas en un circuito. Debemos descargar el archivo **BCDa7seg.dsn**,

SPICE MODELS

Muchos fabricantes proporcionan modelos SPICE de sus componentes. En Proteus estos modelos son precisamente tomados de los fabricantes y agregados al programa. Podemos agregar modelos SPICE a Proteus, se debe realizar una serie de configuraciones especiales para lograrlo.

que posee un circuito integrado **4543**, el cual es un decodificador de BCD a 7 segmentos y encenderá un display con el valor que hay en sus entradas en BCD.

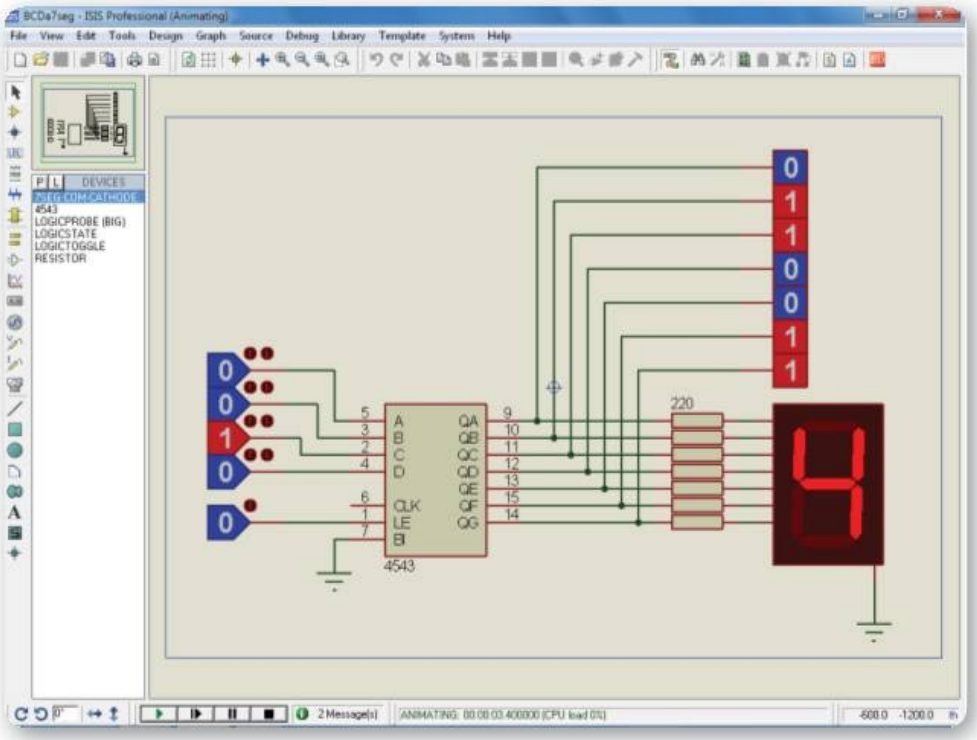


Figura 12. Un controlador de display es un buen ejemplo para estudiar las sondas y los estados lógicos.

Hemos colocado estados lógicos en las entradas **A**, **B**, **C** y **D** para poder dar un valor en BCD al 4543 con ellos. También agregamos un estado lógico momentáneo en la entrada **Latch Enable**

COMPONENTES EN NUEVAS VERSIONES

Aunque hay componentes que no pueden simularse por carecer de modelo para hacerlo, cada actualización de una versión de Proteus suele incluir nuevos componentes que se pueden simular. Además, algunos que no tenían modelo en versiones anteriores ya lo tienen en las nuevas para poder simularlos.

(**LE**); de esta forma, después de cambiar el valor de los estados lógicos en **A**, **B**, **C**, **D**, debemos presionar el estado colocado en **LE** para dar un pulso y hacer que el nuevo valor se refleje en las salidas del 4543 y en el display.

En las salidas **QA** a **QG**, ubicamos sondas lógicas para ver su nivel lógico en todo momento, independientemente de que se reflejen en el display. De esta forma, podemos apreciar el funcionamiento y la utilidad de los estados lógicos y las sondas lógicas en los circuitos digitales.

Simulación mixta

La simulación mixta es la que involucra una parte digital y otra analógica. Proteus es capaz de simular circuitos mixtos sin ningún problema, aunque debemos considerar la complejidad del circuito si necesitamos hacer una simulación en tiempo real.

Proyecto: efecto de luces

Veamos un ejemplo de una simulación mixta para la cual utilizaremos el archivo **EfectoLuces.dsn**.

Este circuito es muy simple, ya que solo requiere un circuito

ESTADOS EN LUGAR DE PULSADORES

En los circuitos digitales es común usar pulsadores para dar pulsos o cambiar de estado algunas entradas del circuito. Podemos utilizar estados lógicos en vez de los pulsadores y obtener el mismo resultado, además de quitar carga de procesamiento. También, al momento de dibujar el circuito puede ser más rápido usar estados lógicos.

integrado para lograr un efecto de luces desplazándose con LEDs. Debemos tener en cuenta que está construido a partir de un inversor con disparador Schmitt; en este caso usamos el circuito integrado **74HC14**.

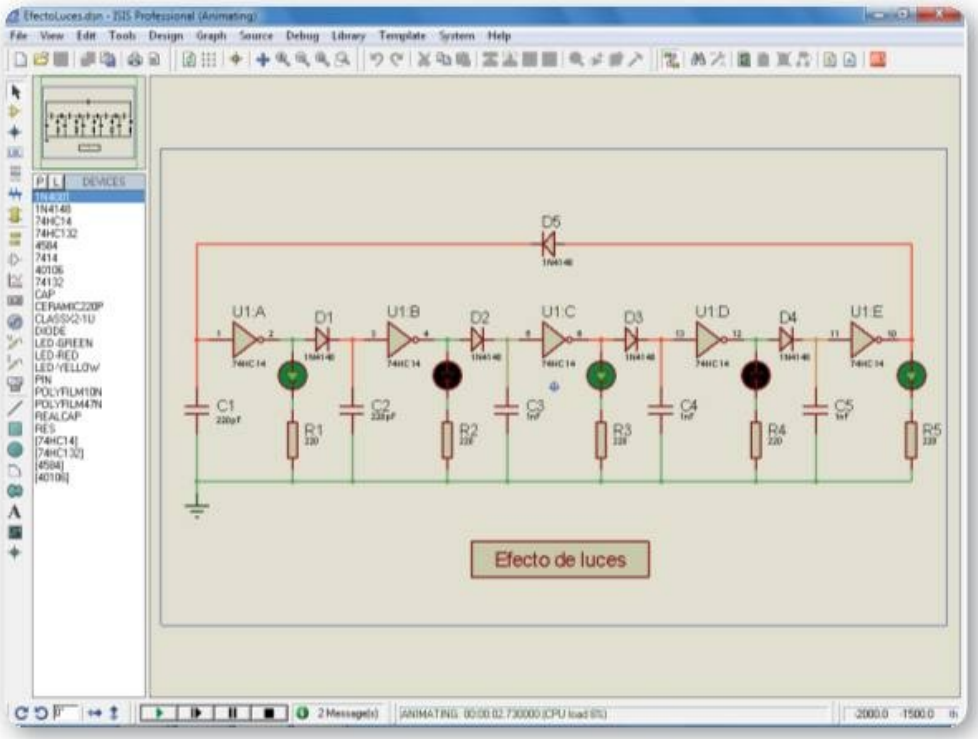


Figura 13. Un efecto de luces con LEDs es un ejemplo de un circuito mixto sencillo.

Es mixto porque el 74HC14 es un circuito digital, pero los diodos 1N4148 y los capacitores son componentes analógicos, y de ellos depende en gran medida el correcto funcionamiento de este circuito. Al abrir el archivo **EfectoLuces.dsn** y correr la simulación, veremos en funcionamiento un circuito mixto en Proteus.

En este sentido debemos considerar que es posible notar dos detalles muy importantes:

Capacitores: son algo especiales. Además de los específicos y genéricos, existe un modelo llamado **REALCAP**, que es un tipo de capacitor real, en donde es posible especificar no solo la capacitancia sino también otros parámetros.

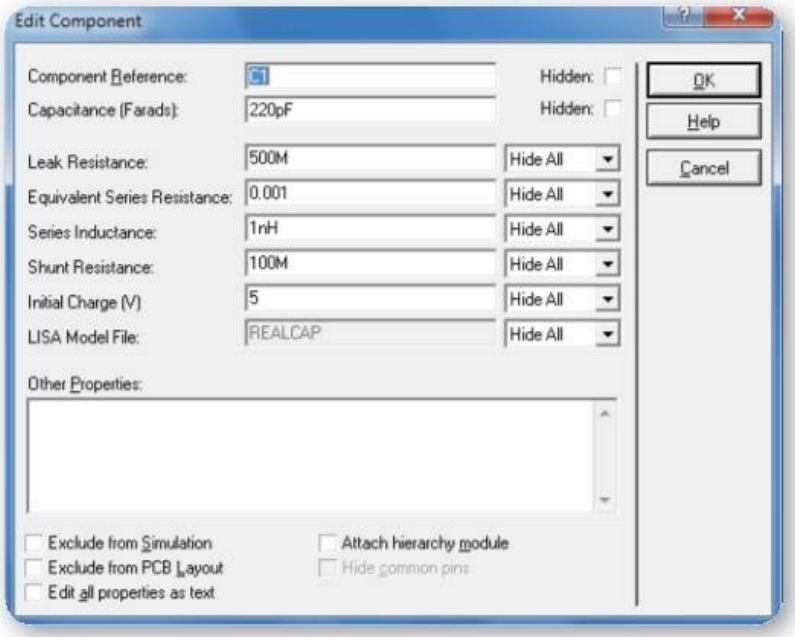


Figura 14. El modelo REALCAP es un capacitor que se asemeja mucho al comportamiento de un capacitor verdadero.

RESUMEN

En este capítulo hemos concluido con las opciones básicas para dibujar diagramas de circuitos electrónicos en ISIS. Además, comenzamos a simular circuitos en Proteus y aprendimos cómo configurar las opciones de animación y de velocidad. Ya tenemos las bases de la simulación, pero esto es solo el principio. También estudiamos algunos componentes especiales y herramientas útiles para nuestras simulaciones.

*03

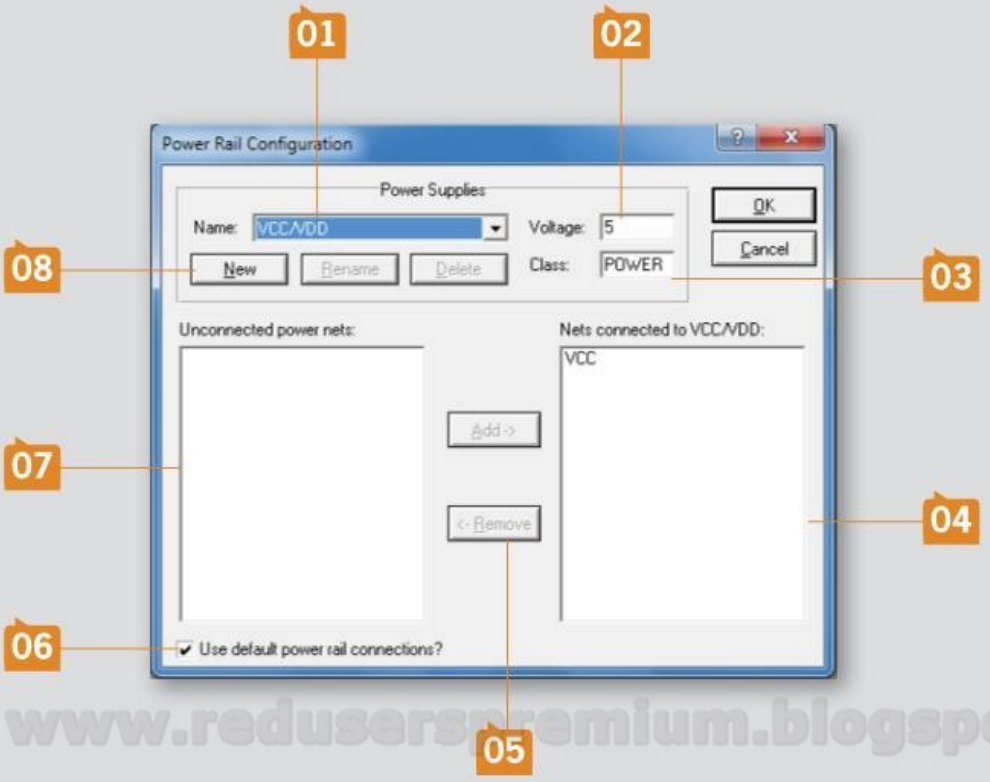
Señales e instrumentos de medición

En este capítulo aprenderemos cómo ISIS provee de alimentación a los circuitos y la obtención de señales mediante los generadores. También conoceremos las herramientas virtuales de medición y análisis.

Configuración de líneas de alimentación

ISIS puede generar de forma automática los **voltajes de alimentación de corriente directa** para las simulaciones, especialmente, para los circuitos integrados digitales. Para ver o cambiar la configuración de estas líneas de alimentación, debemos ir al menú **Design/Configure Power rails**. Se abrirá la ventana **Power rail configuration** (configuración de líneas de alimentación), que explicaremos en la siguiente **Guía visual**.

GV: CONFIGURACIÓN DE LÍNEAS DE ALIMENTACIÓN



01 **NAME (NOMBRE):** en esta lista tenemos los nombres de las líneas o fuentes de alimentación disponibles.

02 **VOLTAGE (VOLTAJE):** aquí podemos ver o cambiar el valor de voltaje que provee la línea de alimentación elegida en el campo **NAME**.

03 **CLASS (CLASE):** establece la clase; por defecto, todas serán **POWER**, y normalmente no necesitaremos cambiar este campo.

04 **UNCONNECTED POWER NETS (REDES DE ALIMENTACIÓN NO CONECTADAS):** en este recuadro aparecen las redes de alimentación que no están conectadas a ninguna fuente de alimentación.

05 **ADD, REMOVE (AGREGAR, REMOVER):** estos botones permiten agregar o quitar redes para conectarlas o desconectarlas de las líneas de alimentación.

06 **NETS CONNECTED TO (REDES CONECTADAS A):** en este recuadro figuran las redes que ya están conectadas a la fuente de alimentación definida en el campo Name.

07 **USE DEFAULT POWER RAIL CONNECTIONS? (¿USAR CONEXIONES DE LÍNEAS DE ALIMENTACIÓN POR DEFECTO?):** permite cargar las opciones predefinidas de la configuración. Podemos dejarla marcada y realizar cambios, pero si la desmarcamos y luego la seleccionamos otra vez, se nos pedirá confirmación para cargar los valores por defecto. Al hacerlo, se borrarán las configuraciones personalizadas.

08 **NEW, RENAME, DELETE (NUEVO, RENOMBRAR, BORRAR):** estos botones nos permiten crear una nueva línea o fuente de alimentación, renombrarla o borrarla.

Al crear un nuevo documento o diseño en ISIS, la configuración de las líneas o fuentes de alimentación tendrá valores por defecto. En el campo **Name**, encontraremos tres líneas de alimentación: **VCC/VDD** con un voltaje por defecto de **5 V**, **GND** con un voltaje de **0 V** y **VEE** con **-5 V**. Para la línea GND no es posible cambiar ningún parámetro, excepto la clase, ya que, de manera

predefinida, GND o tierra siempre tendrá un valor de 0 volts. Para las líneas VCC/VDD y VEE solo podremos modificar el valor del voltaje y la clase. No está permitido renombrar ni borrar estas fuentes de alimentación.

Líneas de alimentación y terminales POWER

Para crear nuevas líneas de alimentación, es posible utilizar las terminales **POWER**. Como hemos visto, estas terminales por defecto se conectan a la línea **VCC/VDD**, a menos que cambiemos su nombre. Para crear una línea o fuente de alimentación con una terminal POWER, podemos cambiar su nombre ingresando directamente el valor de voltaje deseado, pero siempre debemos comenzar con un signo más (+) o menos (-), ya que con esto especificamos la polaridad del nuevo voltaje.

Por ejemplo, consideremos que si necesitamos crear una línea de alimentación de **12 volts**, colocaremos una terminal POWER con el nombre **+12** o **+12 V**.

En la **Figura 1** vemos ejemplos de líneas de alimentación que utilizan terminales POWER, con un valor de voltaje ingresado directamente como nombre. Si vamos a la ventana de configura-

ELEMENTOS INDIVIDUALES Y PINES OCULTOS

Hemos visto que algunos circuitos integrados presentan elementos individuales en el diagrama, como en el caso de las compuertas lógicas. Sin embargo, si necesitamos cambiar el nombre de los pines ocultos en ellos, debemos hacerlo en todos los elementos; por ejemplo, en el 74HC14 tendremos que hacerlo en todos los inversores.

ción de estas líneas, notaremos que en el campo **Name** aparecen las nuevas líneas o fuentes en la lista.

Los valores del campo **Voltage** están en gris y no podemos modificarlos, ya que están definidos por el propio nombre de la terminal POWER. Si ingresamos un valor sin signo en el nombre, se tomará como texto, por eso es importante incluir el signo.

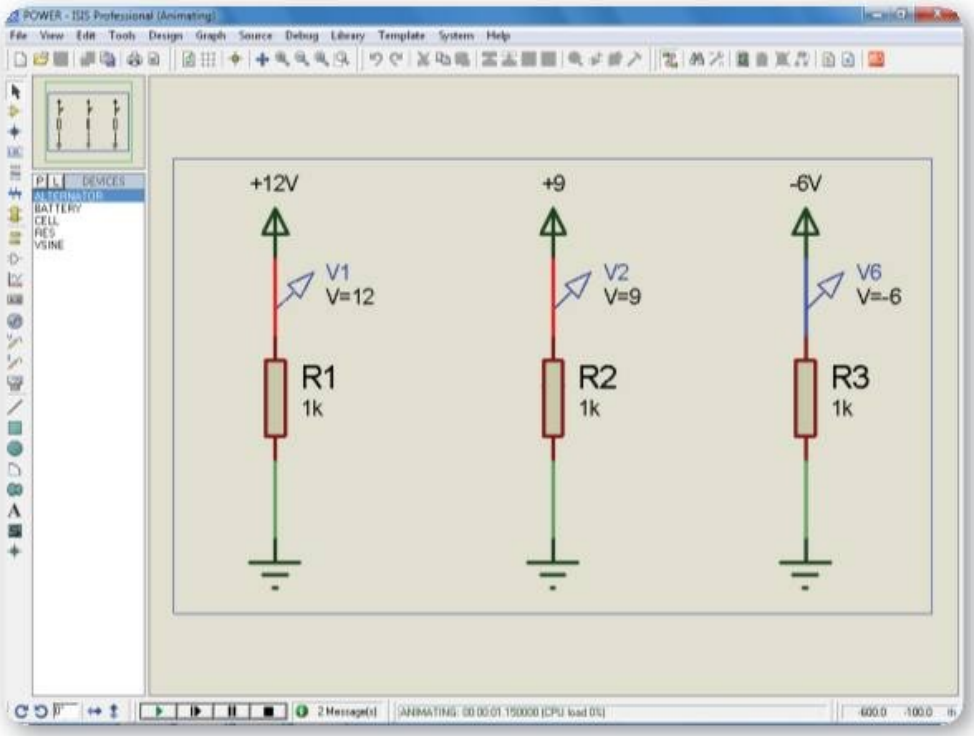


Figura 1. Ejemplos de líneas de alimentación con valores de voltaje ingresados directamente.

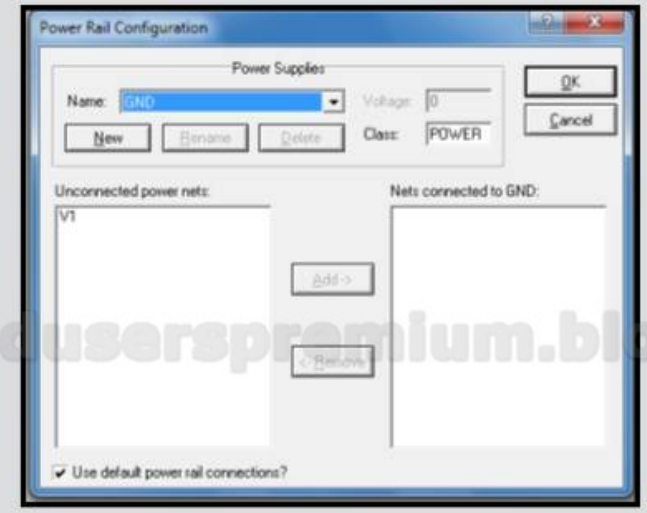
También es posible crear nuevas líneas de alimentación con **nombres** en lugar de valores de voltaje. Al agregar una terminal POWER, podemos colocarle un nombre, por ejemplo **V1**, pero debemos asignarle una fuente de voltaje nueva o conectarla a una ya existente en la ventana de configuración.

En este sentido, si no creamos esta línea de alimentación, debemos considerar que cuando intentemos realizar la simulación, obtendremos un error y por esta razón la simulación no podrá iniciar en forma correcta.

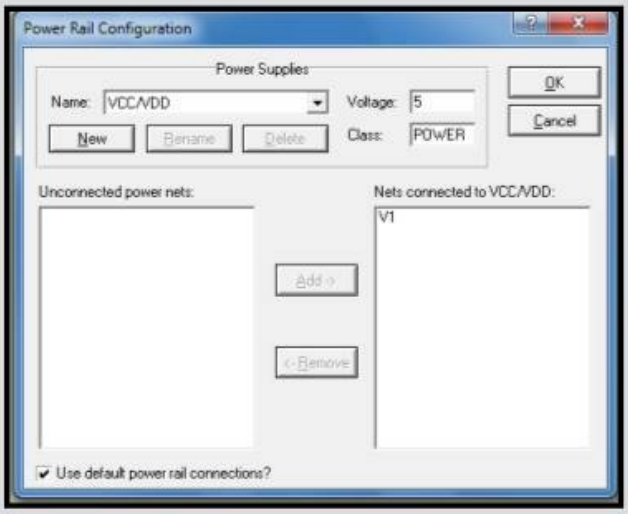
PaP: CREAR LÍNEA DE ALIMENTACIÓN CON NOMBRE

01 En un circuito, coloque una nueva terminal POWER. Haga doble clic sobre ella para acceder a sus propiedades e ingrese un nombre (en este ejemplo, **V1**), y presione el botón **OK**.

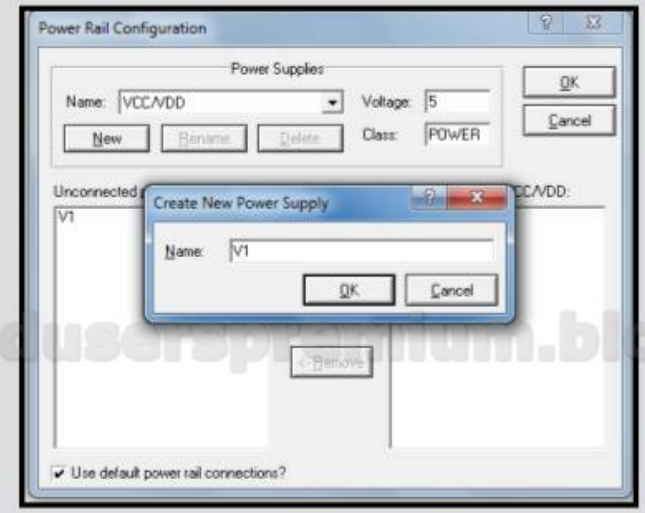
02 Vaya al menú `Design/Configure Power Rails...` y, en la ventana de configuración, observe que la nueva fuente aparece en la lista `Unconnected power nets`, pero no en la lista Name.



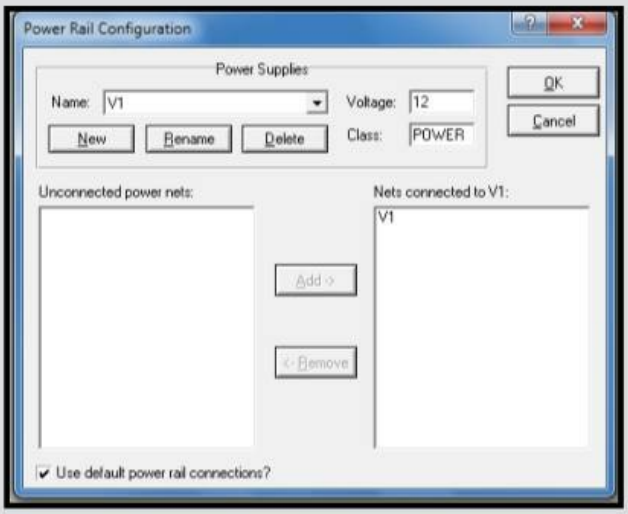
03 Puede conectar la nueva fuente a una existente, como VCC/VDD, elija V1 en la lista Unconnected power nets y presione Add para agregarla a Nets connected to VCC/VDD.



04 La otra opción es crear una línea de alimentación nueva presionando New y, en la ventana Create New Power Supply, ingresar el nombre de la línea, que puede ser igual o diferente del nombre de la terminal **POWER**.



05 Para concluir, ingrese un valor de voltaje en el campo `Voltage`, por ejemplo, 12; luego seleccione V1 en la lista `Unconnected power nets` y presione Add para asignar esta línea a la fuente V1 en la lista `Nets connected to V1`. Finalmente, pulse OK y ya tendrá su nueva línea de alimentación configurada a **12 V**.



Si queremos conectar una nueva terminal POWER a una fuente existente, en sus propiedades podemos elegirla de la lista desplegable del campo **String**, donde aparecen todas las fuentes disponibles.

SCHEMATIC MODELS

Podemos usar las fuentes o líneas de alimentación para dar los voltajes de corriente directa a las simulaciones, aunque también es posible utilizar baterías (**BATTERY**). Consideremos que si usamos la batería que pertenece a la librería **ACTIVE**, podemos definir la resistencia interna (**Internal Resistance**) en sus propiedades.

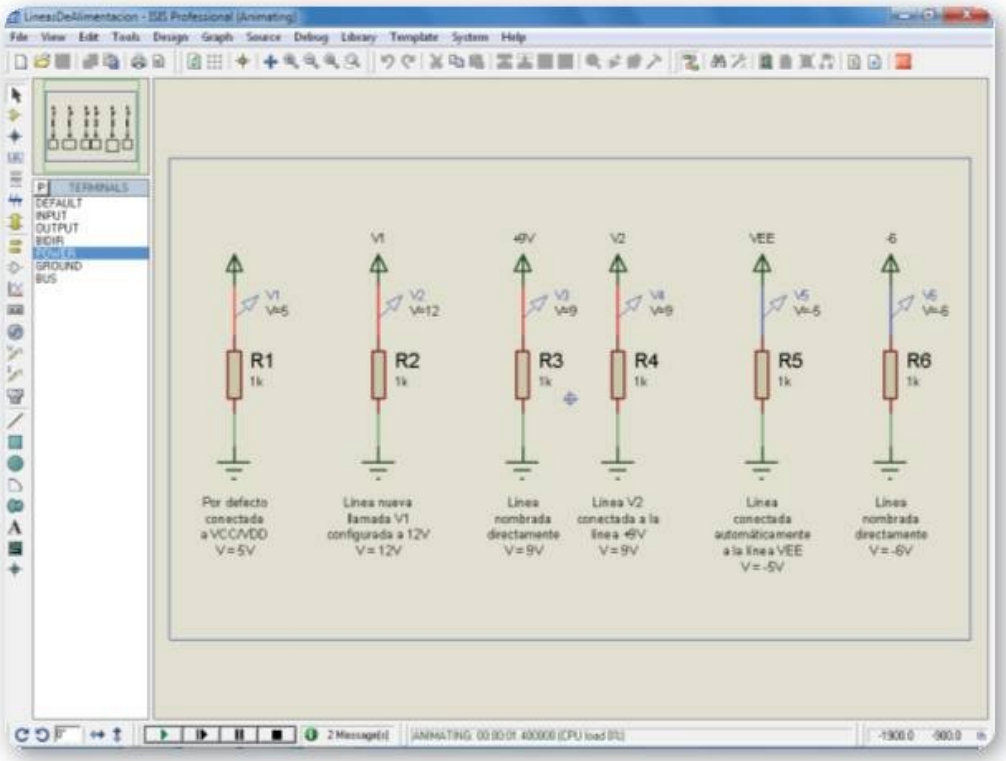


Figura 2. Ejemplo de configuración de múltiples líneas de alimentación en ISIS.

Los generadores de señales

En muchas de las simulaciones, además de los voltajes de alimentación, necesitaremos otros tipos de señales. En estos casos, ISIS cuenta con herramientas para la generación de señales digitales o analógicas, nos referimos a los **generadores**. Para acceder a los diferentes tipos disponibles, pulsamos el botón **Generator Mode** (un círculo gris con una señal senoidal azul) en la barra de herramientas de Modo. Entraremos en **modo Generador** y en el selector de objetos aparecerá una lista con todos los existentes.

Los generadores tienen la forma de un triángulo azul con una punta, y cada uno se acompaña de un símbolo que representa el tipo de señal que producirá. Para colocar un generador en un circuito, solo debemos entrar en **modo Generador**, seleccionar

uno de la lista en el Selector de objetos y ubicarlo en el circuito. Se puede poner el nuevo generador sobre una línea de conexión o en un espacio vacío, y luego conectarlo a un pin o a una línea de conexión existente.

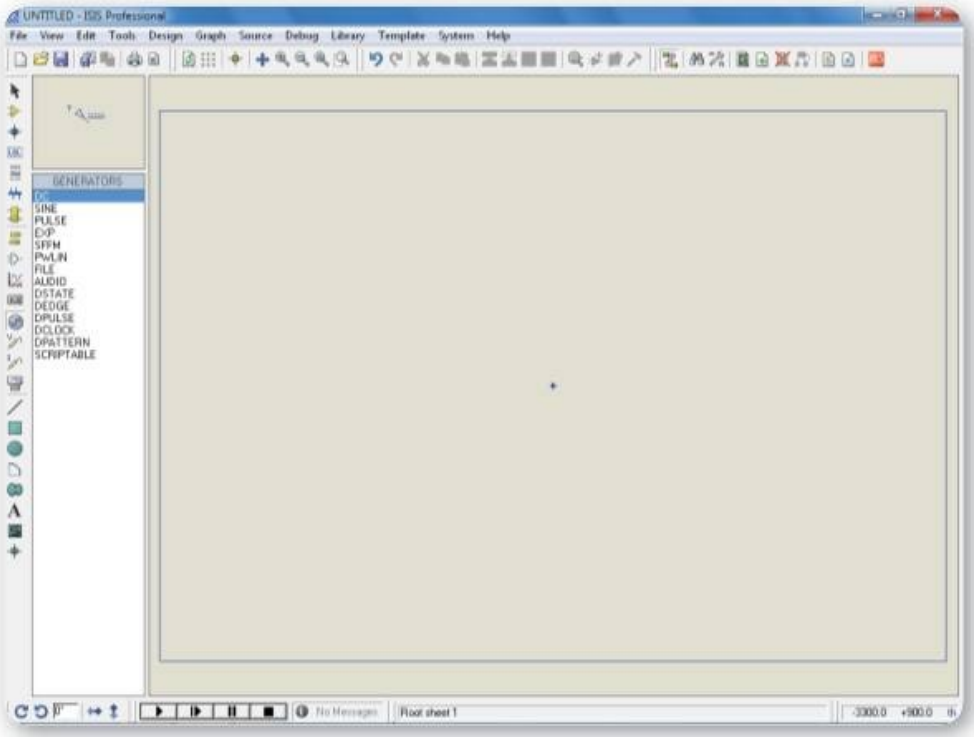


Figura 3. En modo Generador, el título del selector de objetos es **GENERATORS**, con los generadores disponibles.

Los instrumentos de medición virtuales

ISIS cuenta con múltiples **instrumentos virtuales** de medición y análisis, que imitan a las herramientas que usamos en nuestro laboratorio día a día. La forma de utilizarlos y sus funciones son muy similares a las de los instrumentos reales, lo cual nos permite tener todo un laboratorio virtual en nuestra computadora.

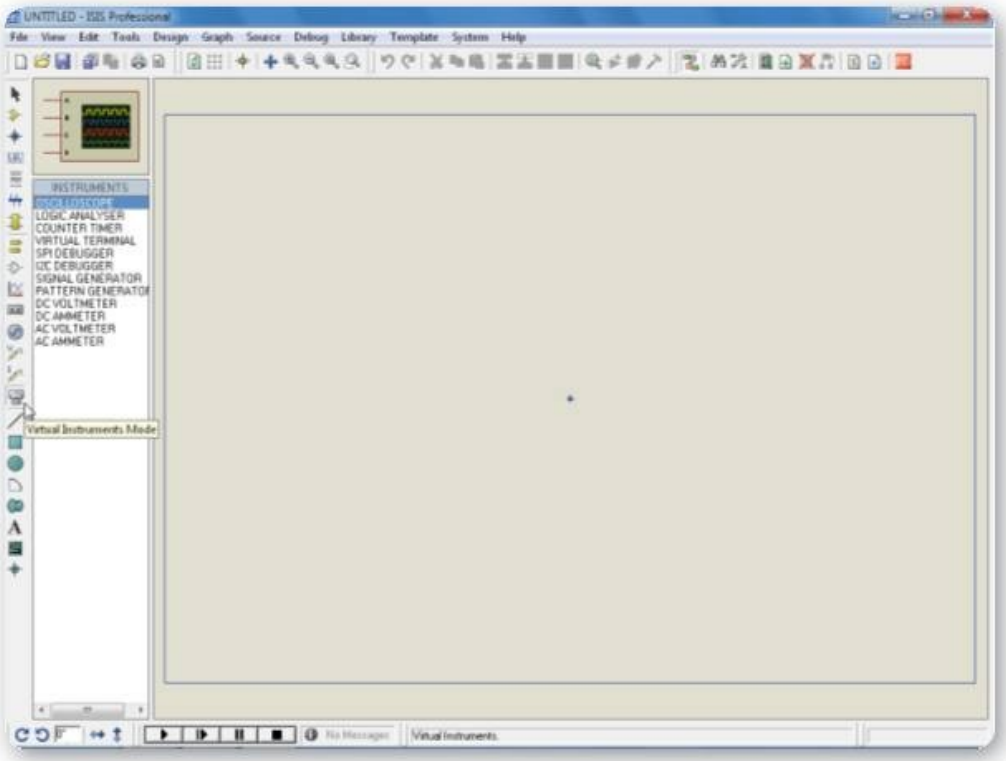


Figura 4. En el modo de instrumentos virtuales el título del selector de objetos es **INSTRUMENTS**.

Para acceder a los instrumentos virtuales en ISIS, tenemos un botón en la barra de herramientas de Modo con la forma de un pequeño medidor analógico, llamado **Virtual Instruments Mode**. Al presionarlo, entraremos en el modo de instrumentos virtuales, y de esta forma el selector de objetos se encargará de mostrar la lista con los instrumentos disponibles.

Voltímetros y amperímetros

Algunos de los instrumentos que utilizamos con mayor frecuencia suelen ser los voltímetros y amperímetros. Estos se encuentran al final de la lista y son: voltímetro de corriente directa (**DC VOLTMETER**), voltímetro de corriente alterna (**AC VOLTMETER**), amperímetro de corriente directa (**DC AMMETER**) y amperímetro de corriente alterna (**AC AMMETER**).

Si vamos a las propiedades del voltímetro de corriente directa, veremos dos campos: **Display Range** para elegir el rango, que puede ser **volts**, **milivolts** o **microvolts**; y **Load Resistance** para seleccionar la resistencia interna del voltímetro, que por defecto es **100M**. En el voltímetro de corriente alterna, además, disponemos de la opción **Time Constant**, para definir la constante de tiempo. En los amperímetros también tenemos la posibilidad de cambiar el rango en **amperes**, **miliamperes** o **microamperes**.

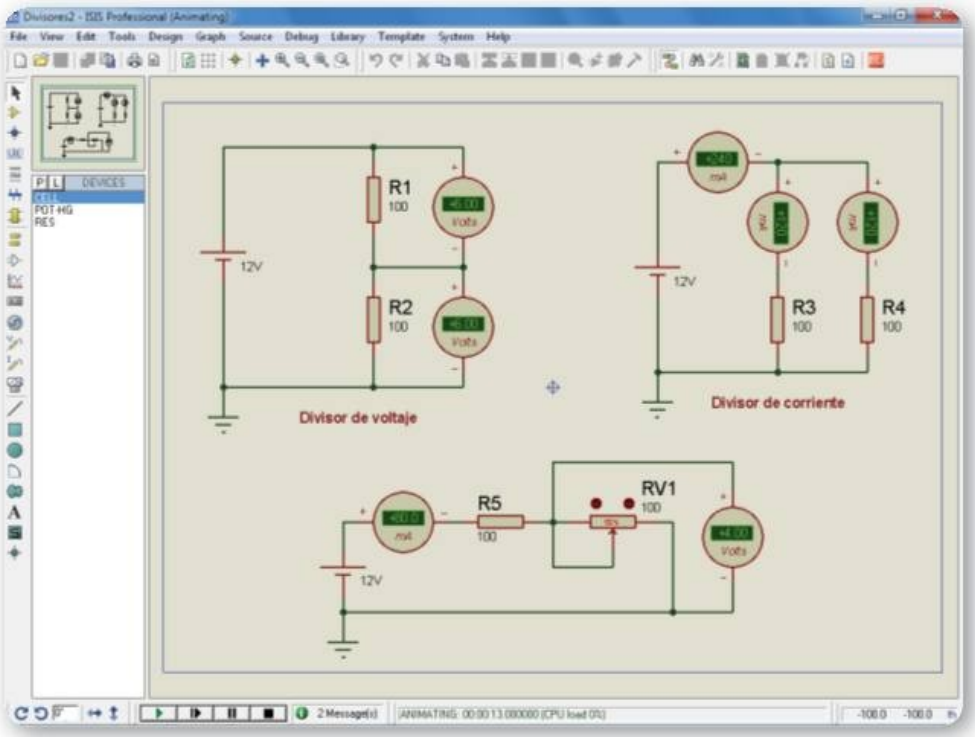


Figura 5. Ejemplo de divisores de voltaje y corriente con instrumentos virtuales.

VIRTUALES VS. REALES

La intención de los instrumentos virtuales es que sean muy parecidos a los reales. Así que los voltímetros y amperímetros deben colocarse de la misma forma como se hace en la vida real.

Debemos saber que, tal como ocurre en la realidad, el voltímetro de corriente alterna de ISIS mide los valores en **RMS**.

Osciloscopio virtual

ISIS cuenta con un **osciloscopio digital virtual** de cuatro canales que permite ver las formas de onda generadas en los circuitos que simulemos. Para usarlo, basta con seleccionar **OSCILLOSCOPE** en la lista de instrumentos virtuales y colocarlo en el diseño como si fuera un componente más.

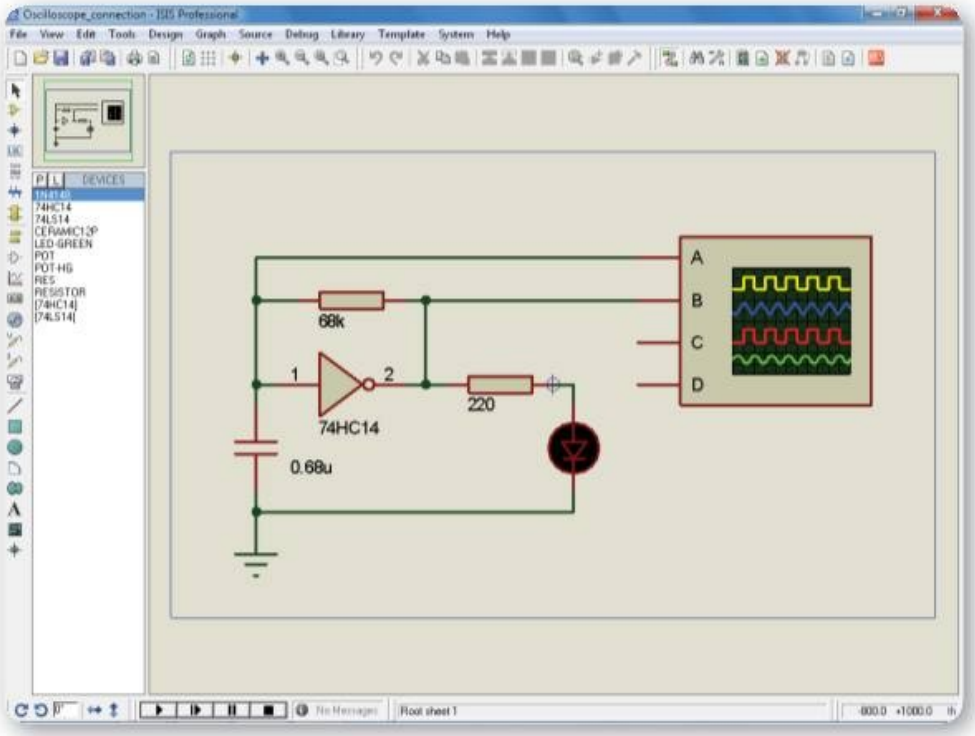
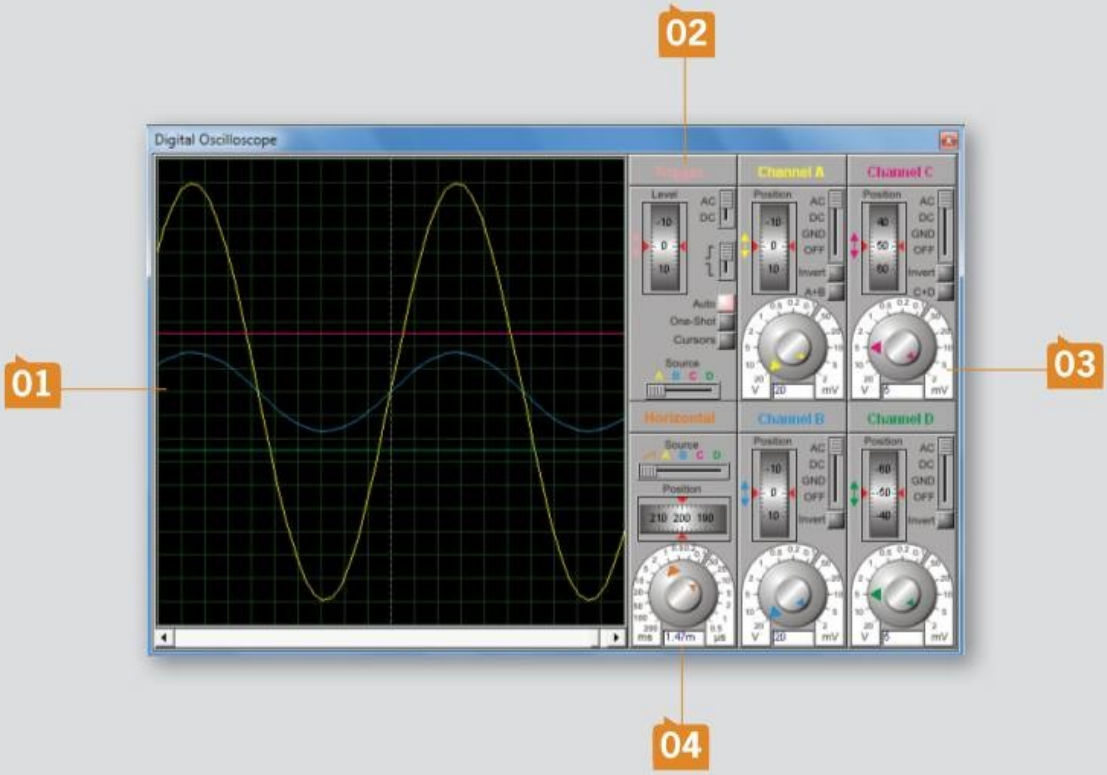


Figura 6. El osciloscopio virtual en ISIS se coloca y conecta como cualquier otro componente.

El símbolo del osciloscopio contiene las terminales de conexión de los cuatro canales, **A**, **B**, **C** y **D**, que se conectan a los puntos donde tomaremos la señal que vamos a medir. Al correr la simu-

lación después de haber agregado y conectado el osciloscopio, aparecerá el instrumento completo en una pantalla más grande con el título **Digital Oscilloscope**. En la siguiente **Guía visual** conoceremos cada uno de sus elementos.

GV: OSCILOSCOPIO VIRTUAL



01 **PANTALLA:** en ella vemos las señales que vamos a medir; cada canal tiene un color diferente.

02 **TRIGGER (DISPARO):** esta es la sección de disparo, muy similar a la de un osciloscopio real.

03 **CANALES:** incluye los controles para los cuatro canales del osciloscopio. En cada uno podemos controlar la posición y la escala vertical.

04 **HORIZONTAL (AJUSTE HORIZONTAL):** permite controlar la posición y la escala horizontal en la pantalla.

Las funciones del osciloscopio virtual son idénticas a las de uno real, así que será muy fácil acostumbrarse a usar este instrumento en Proteus.

Cursores

En la sección **Trigger** del osciloscopio, hay una opción llamada **Cursors**, que permite utilizar cursores en la pantalla para realizar mediciones. Si la activamos, notaremos que al colocar el cursor del mouse en la pantalla del osciloscopio, mientras está corriendo una simulación, aparecen dos líneas, una vertical y una horizontal, que lo siguen en todo momento e indican la posición horizontal (tiempo) y el voltaje de las señales.

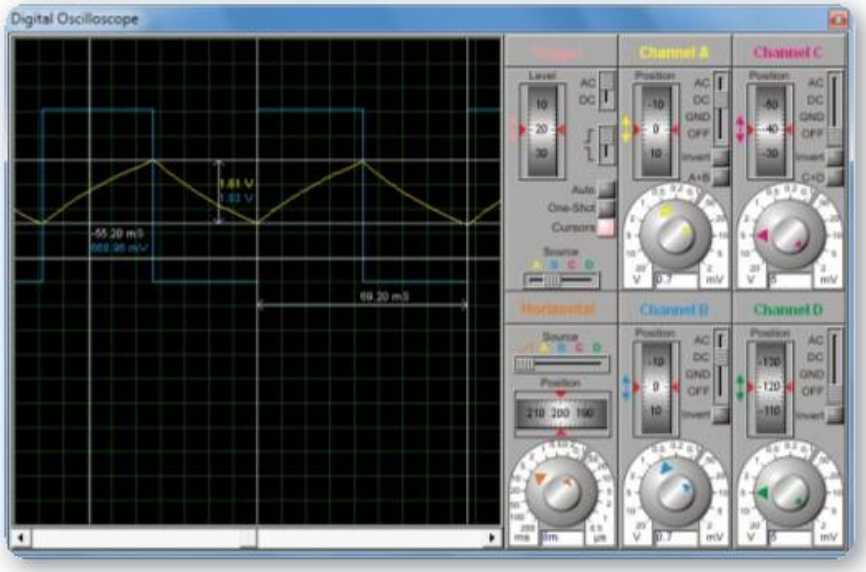


Figura 7. Los cursores permiten obtener medidas de tiempo y voltajes directamente en la pantalla del osciloscopio.

Si hacemos un clic en algún lugar de la pantalla y arrastramos hacia arriba o hacia abajo, aparece una flecha vertical que mide el voltaje comprendido en ese intervalo. Esto puede ser útil para conocer los voltajes de las señales en pantalla. Si arrastramos horizontalmente, se mide el intervalo de tiempo; así obtendremos, por ejemplo, los períodos de las señales.

Podemos mover las flechas o los cursores simplemente colocándonos sobre ellos y arrastrándolos para modificar su posición en cualquier momento. Si hacemos un clic con el botón derecho del mouse sobre un cursor, se abre un menú contextual que nos permitirá borrarlo (**Delete Cursor**) y también eliminar todos los cursores en la pantalla (**Clear All Cursors**). Para ocultar o mostrar los cursores en cualquier momento, solo debemos presionar el botón **Cursors**.

Imprimir y personalizar la pantalla del osciloscopio

Al hacer un clic con el botón derecho del mouse en cualquier parte del osciloscopio, se abre un menú contextual donde encontramos la opción **Setup**, que permite personalizar los colores en la pantalla. Al presionarla, aparece una ventana para elegir los colores del fondo, de los cuatro canales y de los cursores. Podemos cambiar tanto los colores que se mostrarán en el monitor de nuestra computadora (**Display**) como los que saldrán en una impresión (**Printer**). Si seleccionamos la opción **Black and White**, las impresiones saldrán en blanco y negro. Por último, el botón **Reset** devolverá los colores por defecto.

Para imprimir una hoja con las señales del osciloscopio debemos presionar **Print** en el menú contextual. Obtendremos una hoja que contiene la pantalla del osciloscopio, con las señales presentes en ella y, debajo, la información completa de la configuración del osciloscopio en ese momento. Si los cursores están activados, estos también se imprimirán.

Proyecto: control de velocidad de un motor DC por PWM

Como ejemplo del uso del osciloscopio, realizaremos la simulación de un sencillo circuito para controlar la velocidad de un mo-

tor de CD por modulación de ancho de pulso (PWM). Debemos descargar el archivo **PWM.dsn** para ver esta simulación.

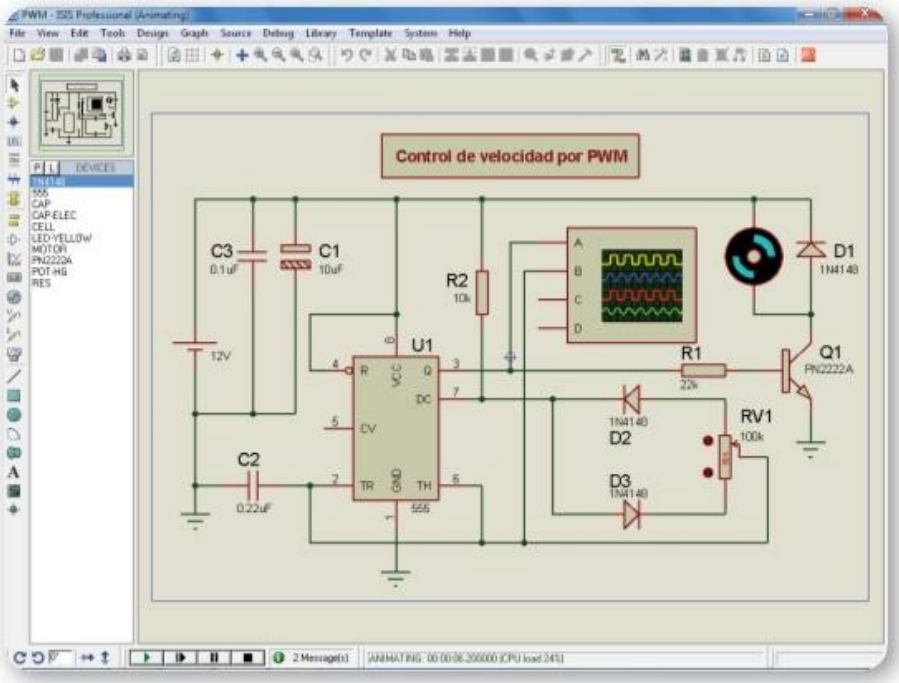


Figura 8. Un circuito PWM utilizando un 555 ilustra un ejemplo del uso del osciloscopio en ISIS.

En el circuito hemos incluido el osciloscopio; conectamos el **canal A** a la salida **Q** del 555, que es donde veremos la señal PWM, y el **canal B** a la terminal **TH** para observar la carga y descarga del capacitor **C2**. Al correr la simulación, aparecerá el osciloscopio con las señales en la pantalla. Si modificamos el

valor de **RV1**, observaremos el cambio en la velocidad del motor y también en las señales en la pantalla del osciloscopio.

Durante la simulación, podemos cerrar la ventana del osciloscopio para ver el circuito con más detalle. Para recuperarlo, hacemos un clic con el botón derecho del mouse sobre su símbolo, y en la parte inferior aparecerá la opción **Digital Oscilloscope**, al igual que en el menú **Debug**.

Contador/temporizador

El siguiente instrumento virtual que analizaremos es el **contador/temporizador**, que será útil en las simulaciones digitales para medir tiempos, frecuencias o contar pulsos. Lo encontramos en la lista con el nombre **COUNTER TIMER**. Tal como los demás, lo agregamos al circuito como si fuera un componente y lo conectamos al punto donde queremos tomar la medición.

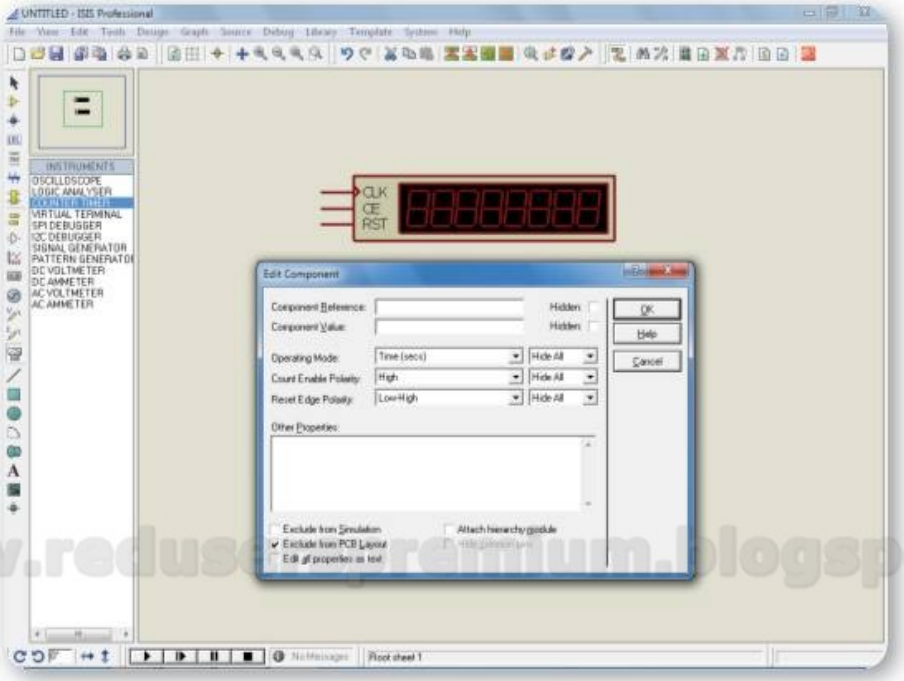


Figura 9. En las propiedades del **contador/temporizador** elegimos el modo de funcionamiento.

El contador/temporizador tiene tres terminales de conexión: **CLK** es la que recibirá la señal para medir la frecuencia o contar pulsos, **CE** o **Count Enable** sirve para habilitar o deshabilitar el instrumento, y **RST** o **Reset**, para borrar y reiniciar la cuenta o medición. En las propiedades del contador/temporizador podemos configurar el modo y la función de los pines CE y RST. En el campo **Operating Mode** es posible elegir entre cuatro modos de operación: **Time (secs)** para medir el tiempo en segundos; **Time (hms)** para medir el tiempo en horas, minutos y segundos; **Frequency** para medir la frecuencia; y **Count** para contar los pulsos. El campo **Count Enable Polarity** permite seleccionar el nivel para habilitar o deshabilitar el instrumento con el pin CE; por ejemplo, si elegimos **High** (alto) el contador/temporizador estará habilitado con un estado alto y deshabilitado con un estado bajo. En el campo **Reset edge Polarity** podemos establecer el flanco con el cual se dará el reset al instrumento: **High-Low** para un flanco de bajada o **Low-High** para un flanco de subida.

Al correr la simulación se mostrará la medición en el propio símbolo del contador/temporizador, pero si hacemos un clic sobre él, aparecerá una ventana llamada **VSM Counter Timer**, que tiene diferentes botones para controlar su funcionamiento durante la simulación, incluyendo un botón de reset manual (**MANUAL RESET**).

Veamos un ejemplo del uso de este instrumento, para lo cual vamos a descargar el archivo **monoestable555.dsn**.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

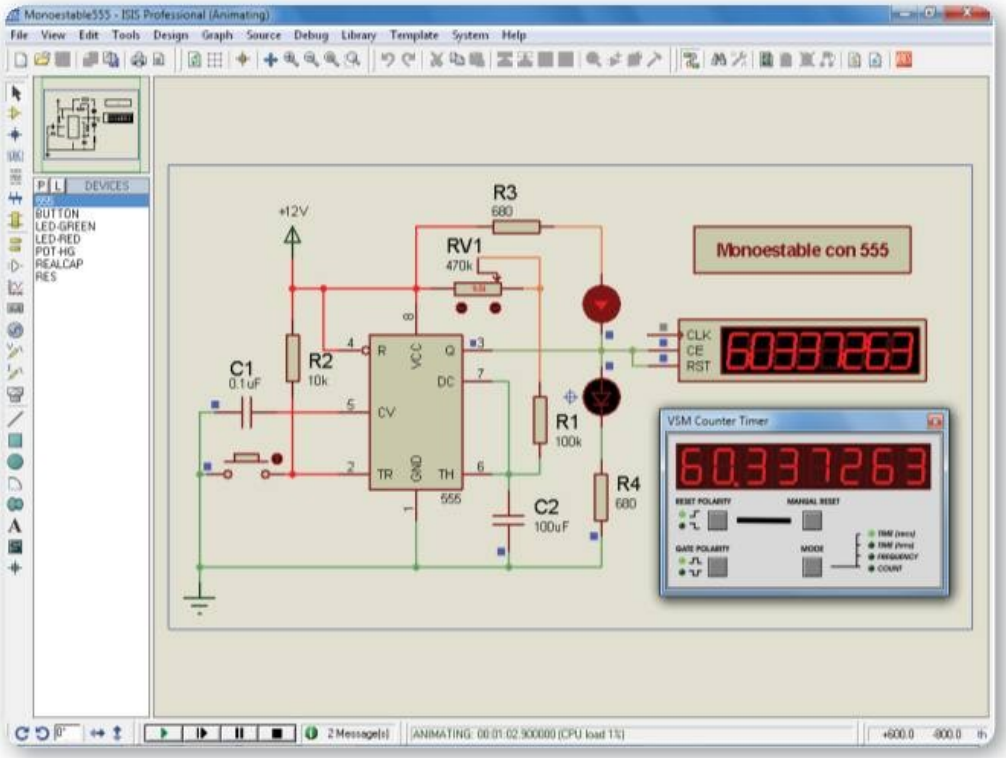


Figura 10. El **contador/temporizador** nos permite medir tiempos y otros parámetros en nuestras simulaciones.

Este es un circuito monoestable construido con un 555, en el que, al presionar el pulsador, tendremos la salida **Q** en un estado alto por un tiempo determinado, que podemos variar usando el potenciómetro **RV1**. Hemos colocado un contador/temporizador en modo **Time (secs)** para medir el tiempo del pulso en el circuito. Las terminales CE y RST se han conectado a la salida Q del

LECTURA DEL CONTADOR/TEMPORIZADOR

En ocasiones, el símbolo del contador/temporizador puede ser muy pequeño (por ejemplo, en el caso de un circuito grande), además de que en la lectura no se muestra el punto decimal y esto puede ser confuso. Para una correcta visualización recomendamos siempre usar la ventana completa del instrumento, que aparece al hacer clic sobre él durante la simulación.

555 y se ha elegido un nivel alto para habilitar el contador; de esta forma, al iniciar el pulso comenzará la cuenta del tiempo. La terminal CE se ha configurado para borrar el contador con un flanco de subida, de modo que, al iniciarse el pulso en Q, la cuenta empezará desde cero. Así podemos usar el contador/temporizador para medir de forma automática el tiempo del pulso de este monoestable.

Generador de señales

Entre los instrumentos virtuales, encontramos el **SIGNAL GENERATOR**, que es capaz de generar señales senoidales, diente de sierra, triangulares y cuadradas, de una frecuencia de 0 Hz hasta 12 MHz y con una amplitud de hasta 12 Volts. Es completamente interactivo, ya que ni en su símbolo ni en sus propiedades podemos ver ni configurar nada.

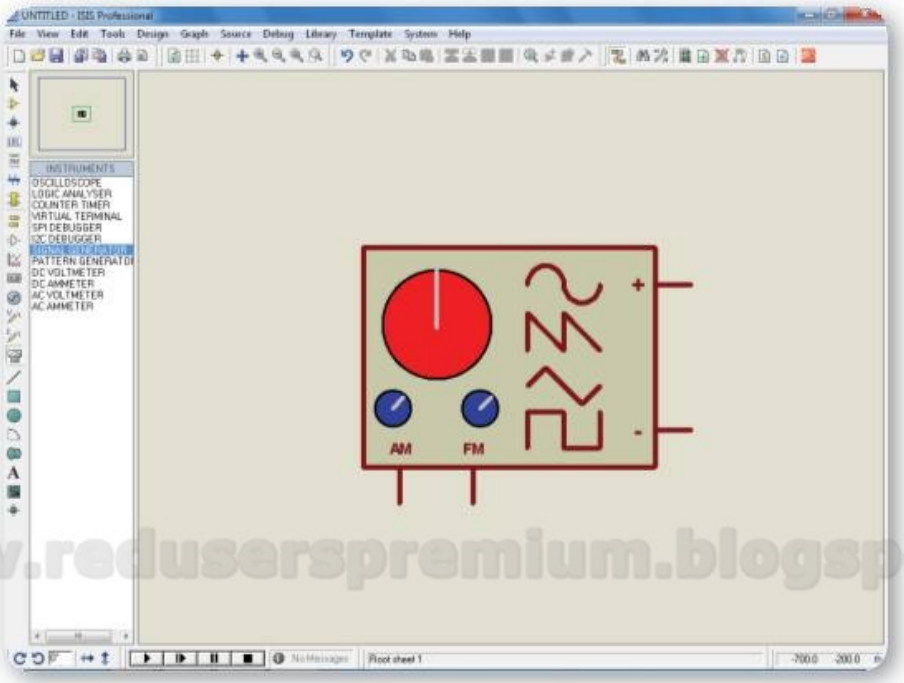
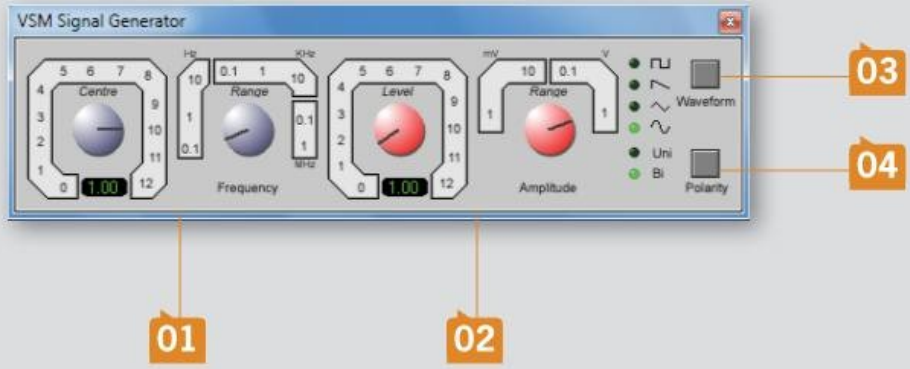


Figura 11. El **generador de señales** es una forma más de obtener señales para usarlas en los circuitos.

Como podemos apreciar, el generador de señales es bastante simple: solo contiene cuatro terminales. Las marcadas con más (+) y menos (-) son las que se conectarán a los puntos del circuito donde inyectaremos la señal. Las terminales **AM** y **FM** sirven para inyectar una señal externa al generador y obtener una señal modulada en amplitud (AM) o en frecuencia (FM). La señal del generador es la que servirá de portadora, y la señal entrante será la moduladora; esta puede ser generada por algún circuito, por un generador o, incluso, por otro generador de señales. Al iniciar la simulación, aparecerá la interfaz del generador de señales en una ventana con el título **VSM Signal Generator**. En la siguiente **Guía visual** conoceremos cada uno de sus elementos.

GV: GENERADOR DE SEÑALES



01 **CENTRE/RANGE (CENTRO/RANGO):** este par de controles se usa para elegir la frecuencia de la señal que se va a generar.

02 **LEVEL/RANGE (NIVEL/RANGO):** con este par de controles definimos la amplitud de la señal.

03 **WAVEFORM (FORMA DE ONDA):** este botón permite cambiar la forma de onda generada: cuadrada, diente de sierra, triangular o senoidal.

04 **POLARITY (POLARIDAD):** mediante este botón, elegimos la polaridad. `Uni` para una señal unipolar, es decir, verticalmente solo estará en el lado positivo; y `Bi` para una señal bipolar, es decir, centrada en el eje horizontal, la mitad de la señal será negativa y la otra mitad positiva.

Figuras de Lissajous

A continuación veremos un ejemplo práctico y sencillo de los generadores de señales. Realizaremos una simulación con el archivo **Lissajous.dsn**, en el cual hemos utilizado un par de generadores de señales, para poder ver las **figuras de Lissajous** en el osciloscopio virtual en ISIS.

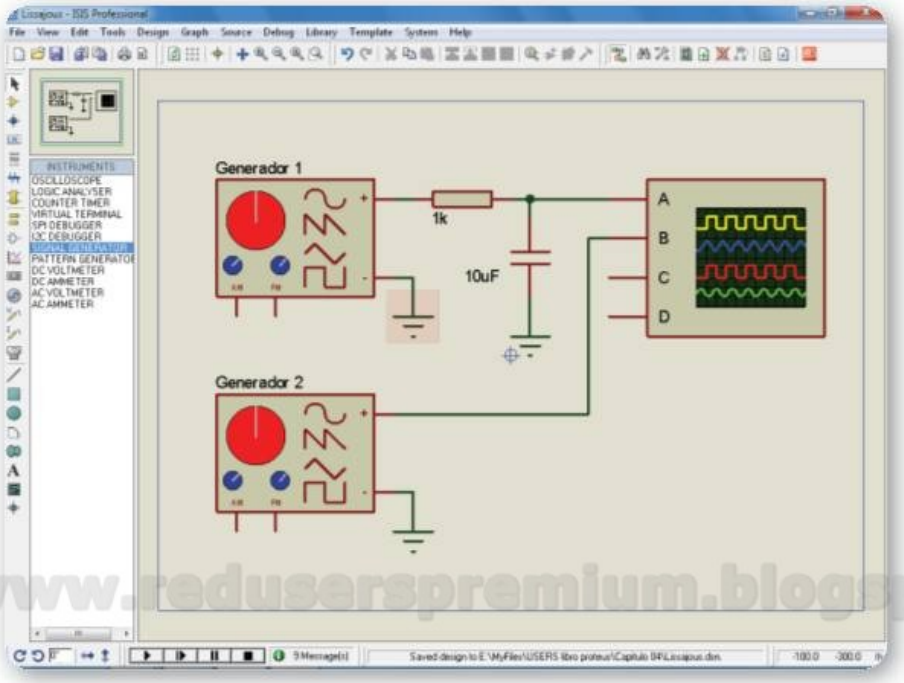


Figura 12. Conexiones de los generadores de señales para obtener las figuras de Lissajous.

Hemos colocado un filtro RC a la salida del primer generador de señales para obtener una señal con fase diferente de la segunda, y con el osciloscopio mediremos ambas señales. Al correr la simulación, veremos los generadores de señales y el osciloscopio con la figura de Lissajous.

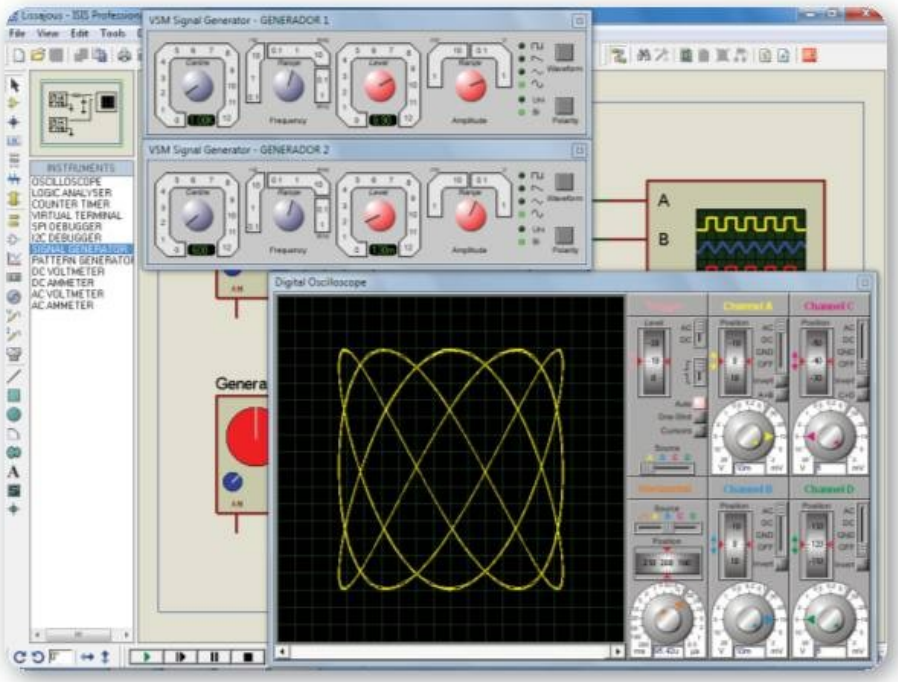


Figura 13. El osciloscopio virtual mostrará las figuras de Lissajous en su pantalla al iniciar la simulación.

Podemos cambiar la frecuencia de las señales para observar las variaciones en las figuras dibujadas en la pantalla del osciloscopio.

RESUMEN

Así como en la realidad existen diferentes formas de obtener señales para los circuitos electrónicos, desde un voltaje de corriente directa hasta señales complejas, en Proteus también encontramos múltiples posibilidades. Aquí vimos cómo se configuran las líneas o fuentes de alimentación de corriente directa en ISIS para la alimentación de los circuitos.

*04

Circuitos con microcontroladores PIC

Proteus es una herramienta muy poderosa porque permite verificar si los programas funcionan correctamente, mediante la simulación de circuitos que contienen microcontroladores y otros periféricos.

Microcontroladores en Proteus

Proteus cuenta con una amplia gama de **microcontroladores** de diferentes familias que pueden simularse con todas sus funciones; los principales son: PIC y dsPIC, AVR, BASIC Stamp, 8051 de Intel, MSP430 y PICAXE, entre otros. Además, posee un gran número de **periféricos** que pueden interactuar con los microcontroladores, como: RTCs, convertidores A/D y D/A, memorias, potenciómetros digitales, sensores de temperatura, etcétera. Para acceder a los microcontroladores y periféricos disponibles, en la ventana **Pick Devices** elegimos la categoría **Microprocessor ICs** o realizamos una búsqueda específica.

Muchos simuladores de microcontroladores solo permiten simular el funcionamiento del propio microcontrolador, pero no es posible observar el comportamiento de elementos externos ni de los periféricos u otros circuitos conectados a él. En cambio, ISIS puede simular el circuito completo, incluyendo todos los periféricos o circuitos que interactúan con el microcontrolador. Es por eso que Proteus es uno de los simuladores preferidos por quienes diseñan circuitos con microcontroladores. En este programa, los microcontroladores son considerados como un componente más en los circuitos.

Buses

Una función muy útil para dibujar diagramas en ISIS y, en especial, en circuitos con microcontroladores, es la posibilidad de usar **buses** para realizar la conexión de múltiples líneas, sin tener un exceso de ellas por todo el diagrama.

Etiquetas de líneas de conexión

Es posible asignar un **nombre** o una **etiqueta** a una línea de conexión, para lo cual debemos presionar el botón **Wire Label Mode** de la

barra de herramientas de Modo. Luego, posicionamos el cursor del mouse sobre una línea de conexión y, cuando este toma la forma de un lápiz de color blanco con una cruz en la punta, hacemos un clic. Aparecerá la ventana **Edit Wire Label**, donde debemos escribir un nombre para la línea de conexión y, a continuación, pulsar el botón **OK**. El nombre asignado aparecerá sobre la línea de conexión elegida; esto sirve como una referencia visual o indicaciones especiales. También podemos nombrar una línea si hacemos un clic derecho sobre ella y, en el menú contextual, elegimos la opción **Place Wire Label**.

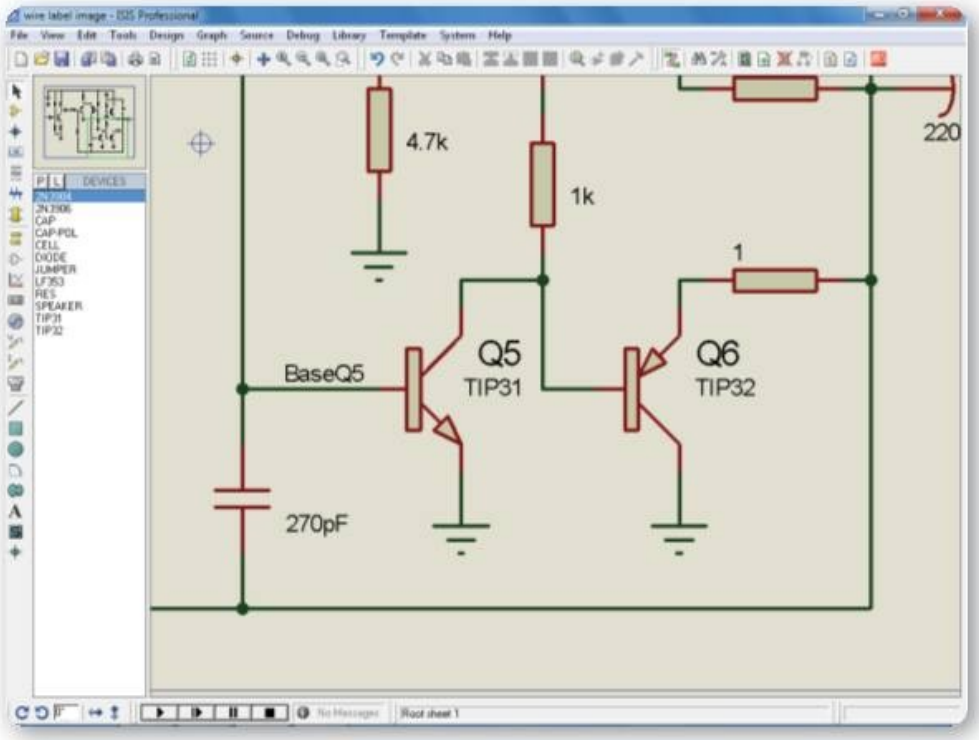


Figura 1. Ejemplo de una línea de conexión etiquetada como **BaseQ5**.

Si queremos retirar el nombre de una línea de conexión, basta con hacer un doble clic sobre la etiqueta y, en la ventana **Edit Wire Label**, borrarlo. Al presionar **OK**, este desaparecerá. Hay que tomar en cuenta que las etiquetas también implican interconexión, si dos o más líneas tienen el mismo nombre. El uso de nombres en las líneas de conexión no solo es útil como referencias visuales, sino que también nos ayudará al momento de usar los buses.

Elementos con pines de bus

Algunos elementos y componentes cuentan con **pines** que son **buses**; se identifican por ser más gruesos que un pin simple y de color azul. Un ejemplo es el generador de patrones, que tiene un pin de bus llamado **B[0..7]**.

Dibujo de un bus

Un bus es un **grupo de líneas de conexión** representado por una sola línea que es un poco más gruesa que las líneas simples. Para colocar un bus en un circuito, tenemos que presionar el botón **Buses Mode** en la barra de herramientas de Modo. Los buses se dibujan de forma similar a las líneas de conexión, con algunas particularidades. Deben conectarse a un pin de bus en algún elemento que cuente con ellos o pueden dibujarse **sin conexión**, es decir, aislados de cualquier elemento o componente.

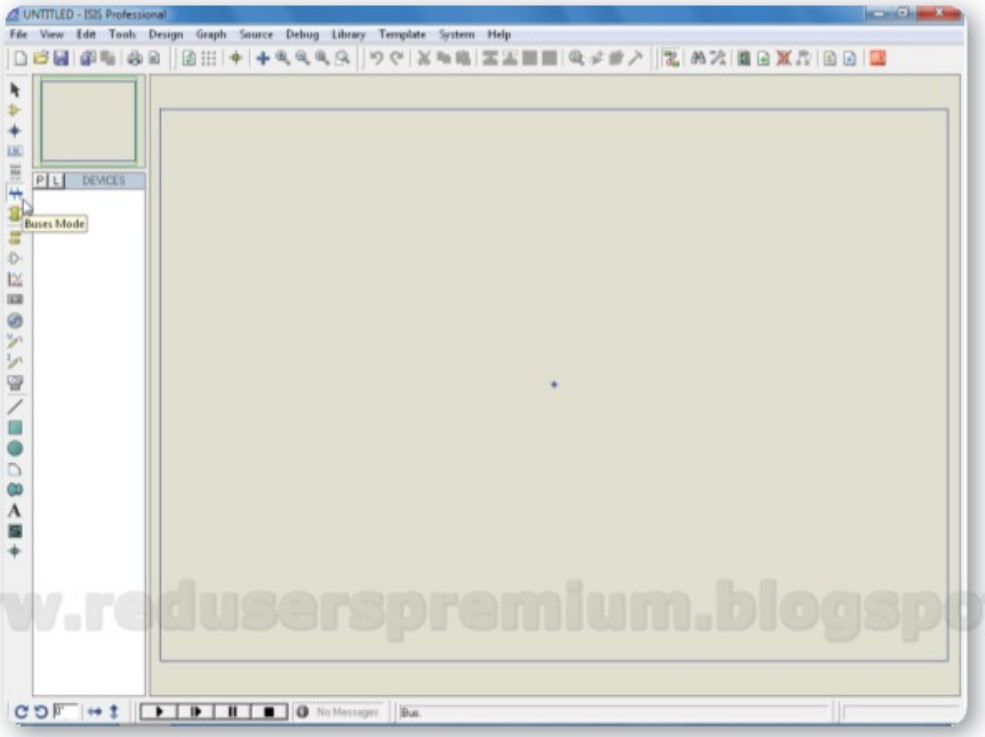


Figura 2. Para dibujar un bus en el circuito hay que entrar en **modo de buses**.

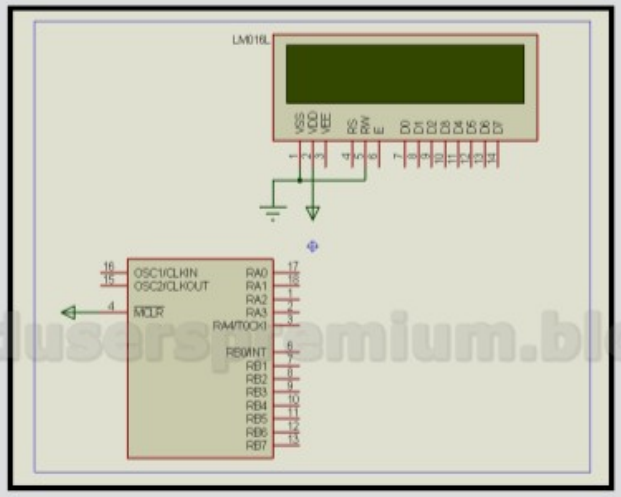
Para agregar un bus a partir de un pin de bus, procedemos de igual forma que con las líneas de conexión simples. Ubicamos el cursor del mouse en el extremo del pin y hacemos un clic para comenzar el dibujo; podemos iniciarlo en un punto vacío de la ventana de edición. Si necesitamos insertar un cambio de dirección del bus, solo debemos hacer un clic en el lugar donde lo deseamos. Para finalizar, conectamos el bus a otro pin de bus, a un bus ya existente o, en su defecto, hacemos un doble clic en un lugar vacío.

Interconexión de pines usando buses

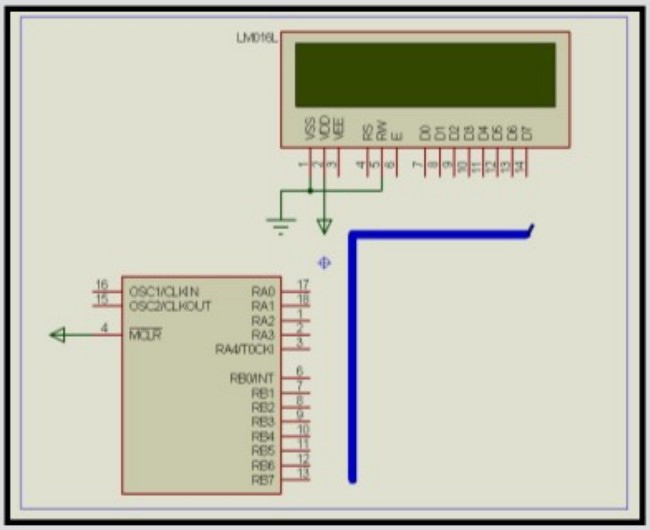
Una vez dibujado el bus, podemos conectarle pines de componentes. Procedemos de manera usual; al conectar una línea al bus debemos etiquetarla para que este sepa dónde va esa línea.

PaP: CONECTAR UN LCD A UN PIC MEDIANTE UN BUS

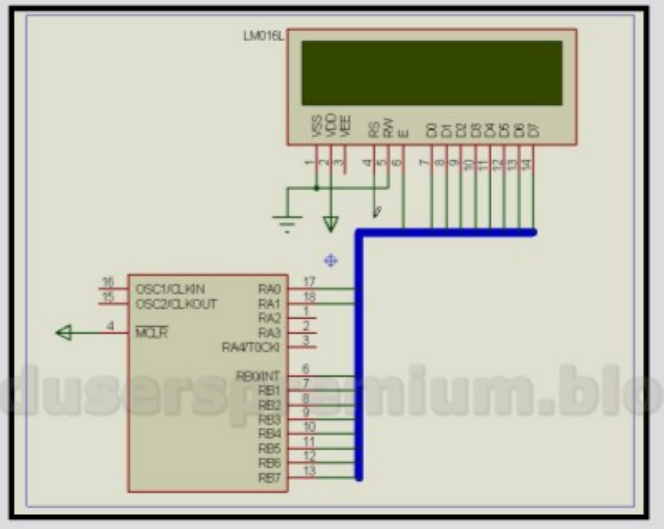
01 Suponga que desea conectar un LCD LM016L a un PIC16F84A mediante un bus. Coloque los componentes en un nuevo diseño y presione el botón Buses Mode para entrar en modo de buses.



02 Haga un clic para iniciar el dibujo del bus y siga la dirección necesaria hasta terminar. Haga un `doble clic` en el lugar donde desea finalizar el dibujo.

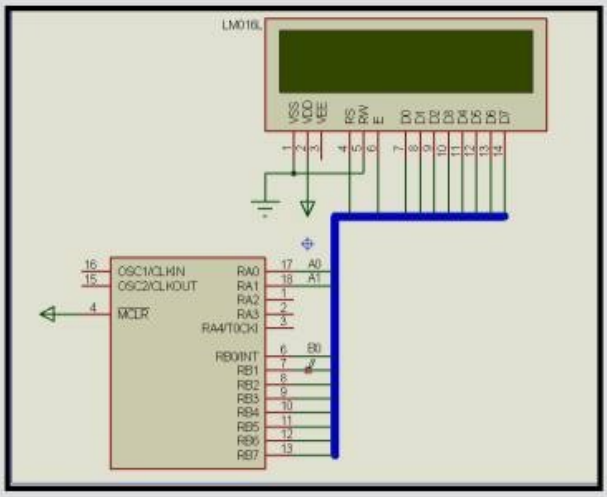


03 Conecte los pines del PIC al bus mediante líneas de conexión simples, y haga lo mismo con los pines del LCD.



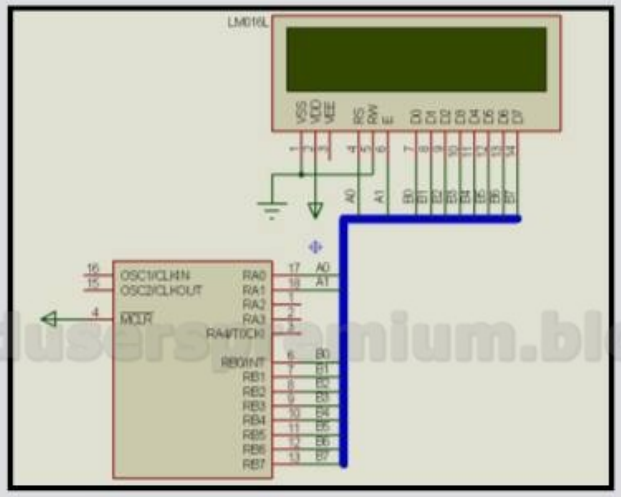
04

Seleccione el botón `Wire Label Mode` y, luego, la línea que une el pin RA0 del PIC al bus. Se abrirá la ventana `Edit Wire Label`; ingrese un nombre en el campo `String`, por ejemplo **A0**, y pulse OK. Continúe con los pines restantes del PIC.



05

Seleccione la línea que conecta al pin RS del LCD, en la ventana `Edit Wire Label` haga un clic sobre la flecha de la lista desplegable del campo `String` y elija A0.



Debemos tener en cuenta que los buses permiten mantener los diagramas ordenados y limpios. Cada línea se identificará en el bus mediante su nombre y se conectará con la línea o las líneas que tengan el mismo nombre o etiqueta.

Etiquetas de bus y terminales de bus

Así como las líneas de conexión pueden ser etiquetadas o nombradas, sucede lo mismo con los buses, aunque en estos se debe especificar un nombre y un rango para las líneas que llevan. Un ejemplo del nombre de un bus es: **D[0..7]**, que representa las líneas llamadas **D0**, **D1**, **D2** hasta **D7**; es un bus de 8 bits. Esto es útil cuando se usan terminales de bus, que, como sabemos, deben tener el mismo nombre para que haya conexión entre ellas.

Simular con archivos HEX y COF

Si lo que necesitamos es solo simular el funcionamiento del circuito, podemos hacerlo con el archivo **.HEX** generado por el MPLAB o algún otro ensamblador o compilador para PIC. Simplemente, en el campo **Program File** de las propiedades del PIC, elegimos el ar-

CONEXIONES CON DOBLE CLIC

Se pueden hacer conexiones rápidas con un **doble clic**. Al tener un bus frente a un grupo de pines de un circuito integrado, por ejemplo, podemos conectar el primero de ellos al bus. En los siguientes haremos **doble clic**, y la conexión al bus se hará de forma automática. Esto permite realizar conexiones a un bus de manera rápida; funciona en todo tipo de conexiones múltiples contiguas.

chivo **.HEX** correspondiente y así simulamos nuestro circuito. Este archivo contiene los datos binarios que el PIC ejecutará, tal como si lo hubiésemos grabado en su memoria de programa.

Muchos ensambladores y compiladores generan también un archivo **.COF**, que puede usarse para depurar el código fuente. Para trabajar con este tipo de archivos, debemos elegirlo en el campo **Program File** en las propiedades del PIC.

En principio, la simulación continua, es decir, la que se realiza al presionar el botón **Play** de la barra de simulación, será idéntica sin importar si se usa un archivo **.HEX** o **.COF**.

Si optamos por este método para simular o hacer depuración, debemos trabajar con el ensamblador o compilador junto con Proteus. Cada vez que necesitemos hacer un cambio en el código fuente, tenemos que hacerlo desde el ensamblador o compilador; luego ensamblamos o compilamos el programa para generar otra vez el archivo **.HEX** o **.COF**; vamos a la ventana de Proteus y corremos la simulación para ver los resultados de los cambios.

Ensamblar desde ISIS

Junto con Proteus se instalan herramientas para ensamblar y depurar programas desde ISIS, sin necesidad de tener instalado ningún ensamblador extra en la computadora. Desde el menú **Source**, podemos configurar las herramientas de ensamblado mediante la opción **Define Code Generation Tools...**. Al seleccionarla, se abrirá la ventana **Add/Remove Code Generation Tools**, donde configuraremos los diferentes ensambladores que tiene Proteus. En nuestro caso, nos interesa el **MPASMWIN**, que es el ensamblador para PICs.

En el campo **Tool**, elegimos **MPASMWIN**. Si no está en la lista, presionamos el botón **New** y buscamos el archivo **MPASMWIN.EXE**, que se encuentra en la carpeta MPASM, el cual, a su vez, está en la carpeta Tools, dentro de la carpeta de instalación de Proteus. En la

sección **Debug Data Extraction**, elegimos el archivo **mpasmddx.exe** presionando el botón **Browse**, que es el que generará el archivo **.SDI** para la depuración de los programas.

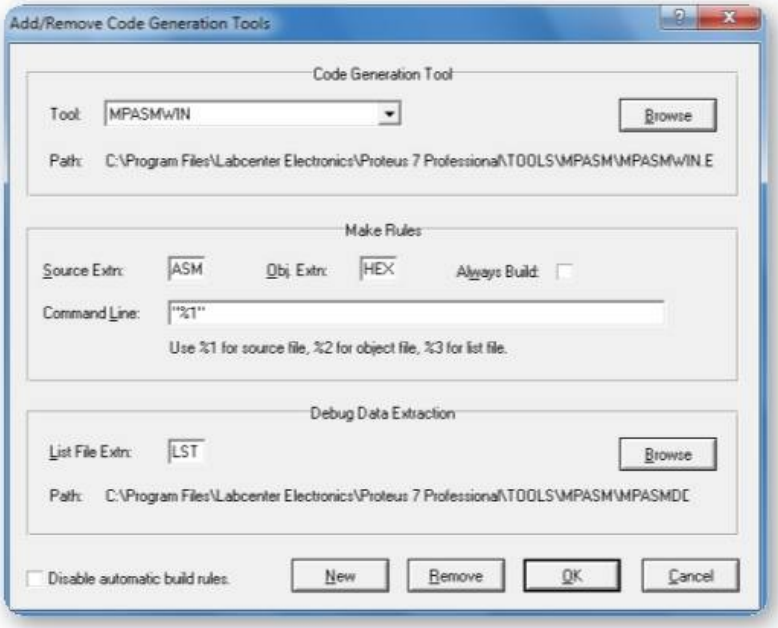


Figura 3. Debemos configurar el ensamblador que vamos a usar con Proteus.

Asignar el código fuente a un PIC

Para ensamblar desde ISIS, debemos asignar el **código fuente** (el archivo **.ASM**) al microcontrolador. Para comprenderlo de forma

¿Y EL CRISTAL?

En la simulación del contador binario con el PIC16F84A, podemos notar que **no hay cristal** conectado al PIC en el diagrama y, sin embargo, la simulación funciona perfectamente. Esto es porque Proteus genera de manera automática la señal de reloj; solo es necesario configurar su frecuencia en las propiedades del PIC.

más clara, veamos un ejemplo real. En el archivo **ContadorPIC.dsn** (y en el archivo **ContadorPIC.asm**) tenemos un PIC16F84A que muestra una cuenta en binario en 8 LEDs conectados a su puerto B.

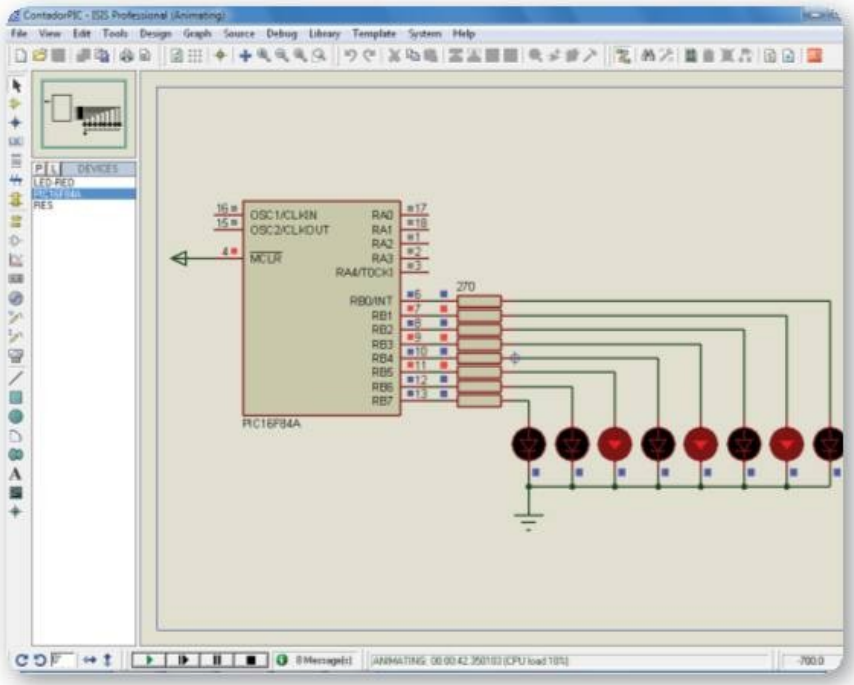


Figura 4. Un ejemplo de un sencillo contador para aprender a ensamblar desde Proteus.

Al asignar un código fuente a un PIC, ya podemos ensamblarlo; para esto solo debemos ir al menú **Source/Build All**, luego se mostrará la ventana con el progreso del ensamblado.

MICROPROCESADORES

Además de los microcontroladores, Proteus cuenta con algunos microprocesadores, como los siguientes: la familia **68000** de **Freescale**, el **8086** de la familia **i86** de **Intel** y algunos de la familia **Z80**. Desafortunadamente, el único de estos que tiene modelo, es decir, que puede ser utilizado para simulaciones, es el **8086**.

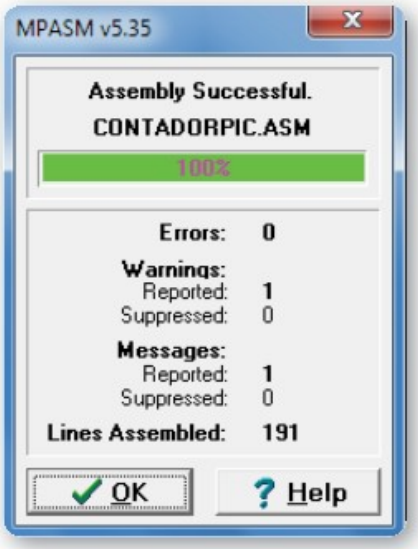


Figura 5. La ventana **MPASM v5.35** muestra el progreso y un pequeño reporte del ensamblado.

Si no se detectan errores, la barra de progreso será de color verde y se mostrará el texto **Assembly Successful**. Luego, al presionar el botón **OK**, se abrirá el informe de simulación con un reporte del ensamblado. En cambio, si hay algún error durante el ensamblado, la barra de progreso será de color rojo y aparecerá el texto **Errors Found**. Al presionar **OK**, se abrirá el informe de simulación con el detalle de los errores.

Al ensamblar sin errores por primera vez, el archivo **.HEX** generado en el proceso se asignará automáticamente al PIC, si es que no existía o no lo habíamos hecho ya. Si por alguna razón esto no sucede, bastará con hacerlo manualmente. Ya podemos correr la simulación para ver el funcionamiento.

El editor de código fuente SRCEDIT

Proteus cuenta además con un sencillo editor de código fuente en caso de que necesitemos hacer cambios en un código. En el menú **Source** encontramos la lista numerada de los códigos fuente asignados a uno o más PICs; si hacemos clic en alguno de ellos, se abrirá el editor SRCEDIT con el código fuente.

Figura 6. Proteus cuenta con su propio editor de código llamado **SRCEDIT**.

El editor no es más que una versión modificada y adaptada del Bloc de notas de Windows, que se instala con Proteus. Al abrir un código fuente en él, podemos hacer modificaciones o corregir errores. ISIS detecta automáticamente que se han efectuado variaciones en el editor y, al correr la simulación otra vez, vuelve a ensamblar el código para registrar los cambios. De esta manera, podemos ensamblar y editar nuestros códigos fácilmente desde ISIS y sin necesidad de usar un programa externo.

Analizador I2C

En los circuitos con microcontroladores existen herramientas para el análisis de comunicaciones **I2C** y **SPI**. Estos analizadores pueden usarse para monitorear las comunicaciones en tiempo real o para emular un elemento que participa en la transferencia o recepción de datos. Solo estudiaremos el analizador I2C, ya que ambos funcionan de manera muy similar, lo único que cambia es el protocolo.

Figura 7. El analizador I2C es totalmente interactivo, tal como los demás instrumentos virtuales.

El analizador I2C tiene tres terminales: **SDA** para la transmisión de datos del BUS I2C, **SCL** para la señal de reloj en el bus I2C, y **TRIG**, que es una terminal de control. Para usar el analizador I2C como monitor, basta con conectar las terminales SDA y SCL a las respectivas terminales en un bus existente, y el analizador mostrará en su pantalla toda la actividad del bus I2C en tiempo real.

Analizador I2C como monitor

Veamos un ejemplo de cómo usar el analizador I2C para monitorear una transmisión en un bus existente.

En el archivo **PIC-24LC256.dsn** podemos observar un circuito con un microcontrolador PIC16F84A y una memoria EEPROM 24LC256.

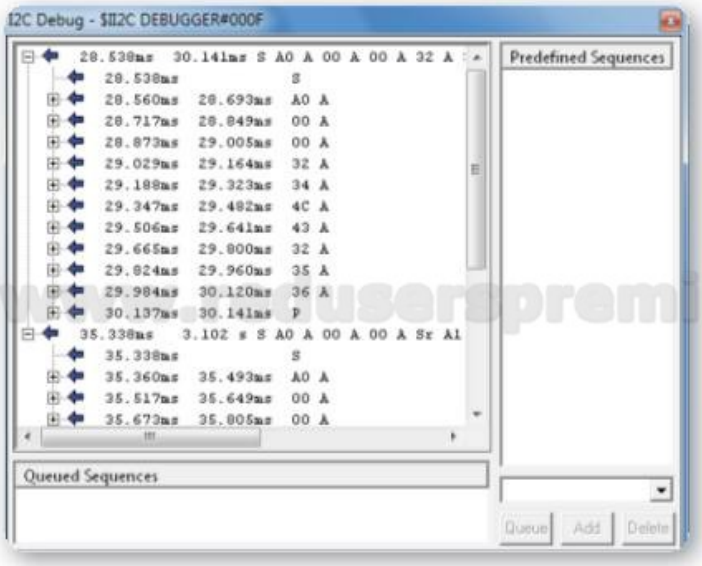


Figura 8. El analizador I2C como **`monitor`** en una conexión PIC16F84A con una memoria 24LC256.

Si abrimos el archivo y corremos la simulación, veremos cómo se abre la ventana **I2C Debug** y en ella se muestra en detalle la transmisión. En este ejemplo, se escribe una serie de datos en la memoria 24LC256 a través del bus I2C. Al finalizar la escritura, se leen los datos y se envían al LCD para verificar que se escribieron correctamente. Es posible leer en la primera línea del analizador el tiempo de inicio y fin de la escritura, que es el primer grupo de envío de datos (28.538 ms a 30.141 ms), y después, todo el protocolo de transmisión comenzando por **Start** (S), para finalizar con **Stop** (P).

ANALIZADOR I2C

▼ SÍMBOLO	▼ DESCRIPCIÓN
S	Condición Start
A	Bit de reconocimiento (ACK)
Sr	Condición Start repetido
N	Bit de reconocimiento negado (NACK)
P	Condición Stop

Tabla 1. Símbolos en el analizador I2C.

Además, los números representan los datos transmitidos a través del bus en hexadecimal. Si hacemos un clic en los signos más (+),veremos con mayor detalle la transmisión de cada condición o dato, incluso, de cada bit de un dato enviado. En la segunda línea que comienza en 35.338 ms tenemos la transmisión al momento de la lectura de la EEPROM. Hemos agregado la ventana **Watch** para verificar los datos de la EEPROM. Si pausamos la simulación, veremos además la ventana **I2C Memory Internal Memory - U2**, que muestra la memoria del 24LC256; también está disponible en el menú **Debug**.

Datos persistentes

En el ejemplo que recién vimos, al correr la simulación una segunda vez (podemos presionar **Pause**), notaremos que los datos en la memoria 24LC256 ya están escritos en ella. Tal como en la realidad, los datos en este tipo de memorias **no volátiles** (la EEPROM interna del PIC es otro de los ejemplos existentes) quedarán grabados hasta que se cambien. Esto puede ser conveniente en algunos casos, pero no en otros.

Si necesitamos borrar los datos en este tipo de dispositivos, podemos ir al menú **Debug/Reset Persistent Model Data** antes de iniciar una simulación, y con esto se borrarán todas las memorias no volátiles que tengamos en el circuito.

Consideremos que es posible hacer esto para comprobar que nuestro ejemplo realmente graba los datos en la EEPROM.

Analizador I2C como maestro

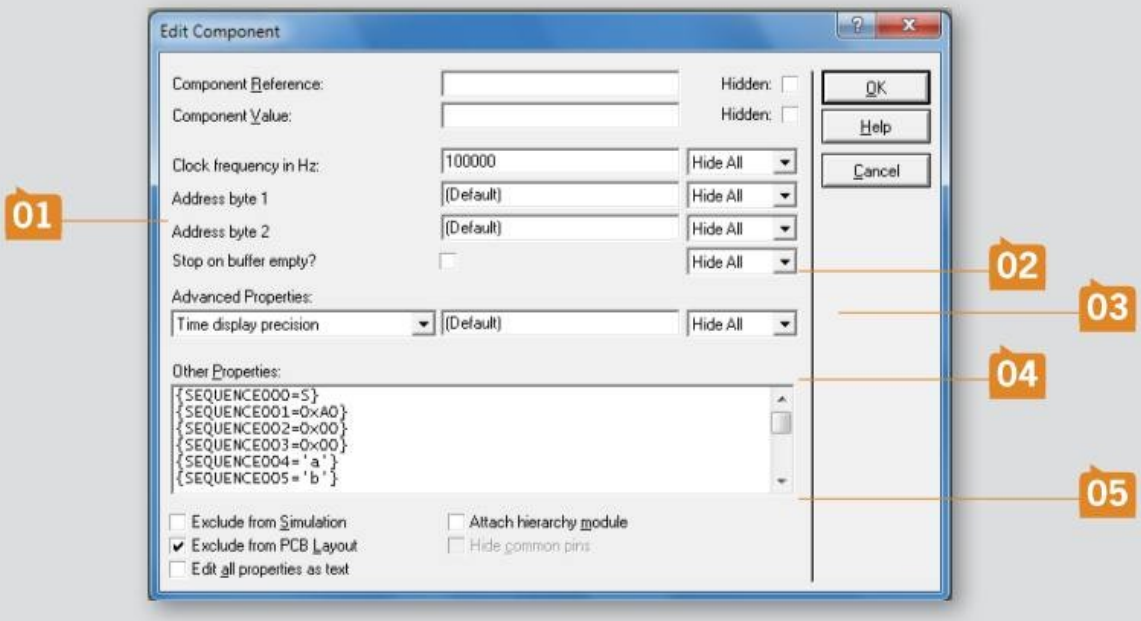
Para usar el analizador I2C como dispositivo maestro, debemos establecer la secuencia de comunicación. Podemos hacerlo desde sus propiedades o de forma interactiva.

En la **Guía Visual** que presentamos a continuación nos encargaremos de conocer en detalle algunas de las propiedades que corresponden a este tipo de analizador.

TRANSMISIÓN Y RECEPCIÓN I2C

Las flechas de color **azul** en el analizador I2C representan recepción, y las de color **rosa**, transmisión. Si se detecta una secuencia no válida, en lugar de una flecha, aparecerá un signo de interrogación. En el informe de simulación habrá más datos en caso de secuencias no válidas o violaciones al protocolo en el bus.

GV: PROPIEDADES DEL ANALIZADOR I2C



01 **CLOCK FRECUENCY IN HZ:** define la velocidad del bus I2C; por defecto, es 100 KHz.

02 **ADDRESS BYTE 1 Y ADDRES BYTE 2:** en caso de usar el analizador como esclavo, aquí se define su dirección en el bus, ya sea de 7 bits con solo la opción 1 o de 10 bits con ambas.

03 **STOP ON BUFFER EMPTY?:** para detener (pausar) la simulación si el buffer de datos ha quedado vacío.

04 **ADVANCED PROPERTIES:** para establecer las propiedades avanzadas:

- `Time display precision`: define el número de decimales en los valores del tiempo al monitorear la conexión; por defecto es 3.

- New `line after`: determina el número de datos que se muestran por línea en la ventana de monitoreo; por defecto es 64.

- `Queue stored sequances at startup`: para elegir si las secuencias almacenadas se emiten al iniciar la simulación o no.

- `Sequence file`: establece un archivo de texto que contiene secuencias.

05 **OTHER PROPERIES:** en este campo se define la secuencia deseada que se enviará, además de otras propiedades.

Se puede establecer una secuencia completa de transmisión simplemente escribiéndola, por ejemplo:

```
S 0xA0 0x00 0x00 0x32 P
```

De esta manera, se enviará toda la secuencia en un solo paso. También se puede definir la secuencia en forma de lista, por ejemplo:

```
S
0xA0
0x00
0x00
0x32 P
```

Así se enviará la secuencia línea por línea; además, podemos hacer una lista de secuencias completas. Es posible definir una secuencia de datos desde la ventana interactiva, presionamos **Pause** para acceder a ella y construimos las secuencias. Los datos de las secuencias se pueden escribir en diferentes formatos.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

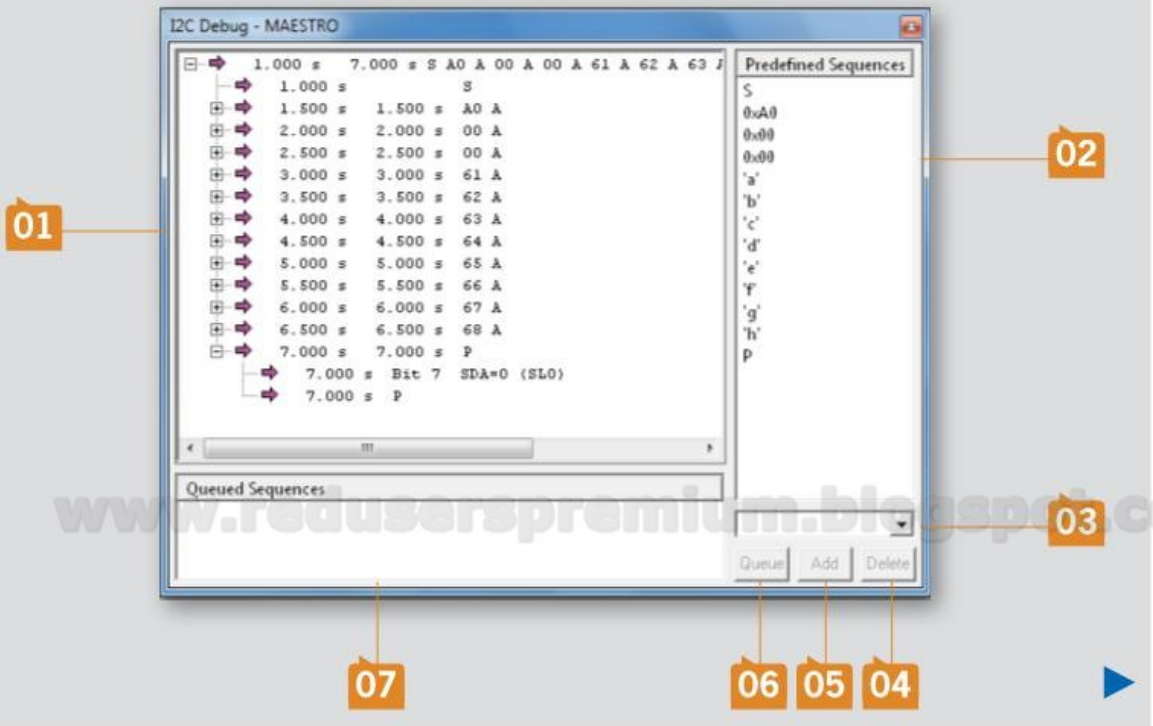
DATOS DEL ANALIZADOR I2C

▼ FORMATO	▼ ESCRITURA
Binario	%_ o _b ejemplos: %01001101, 01001101b
Decimal	_d, ejemplo: 99d
Hexadecimal	0x_, $_ o _h, ejemplos: 0x32, $32, 32h
ASCII	'_', ejemplo: 'S'

Tabla 2. Formato para datos en el analizador I2C.

En la **Guía visual** que presentamos a continuación se enumeran las principales secciones que podemos encontrar en la ventana del analizador I2C.

GV: LA VENTANA DEL ANALIZADOR I2C



01 **VENTANA DE MONITOREO:** aquí aparece toda la actividad del bus I2C en tiempo real.

02 **PREDEFINED SEQUENCES (SECUENCIAS ALMACENADAS):** aquí se almacenan las secuencias que definiremos.

03 **LISTA DE DATOS/COMANDOS:** aquí escribimos o elegimos de la lista las secuencias deseadas.

04 **QUEUE:** con este botón las secuencias almacenadas pasan al campo Queued Sequences y de ahí se envían inmediatamente al bus, si la simulación está corriendo.

05 **ADD:** para agregar la secuencia, comando o dato escrito en la Lista de datos/comandos a la sección de secuencias almacenadas.

06 **DELETE:** para borrar una o más secuencias o datos seleccionados en la ventana de secuencias almacenadas.

07 **QUEUED SEQUENCES:** este es el buffer donde las secuencias se envían al bus en cuanto es posible.

Podemos enviar las secuencias almacenadas de forma manual, mediante alguno de estos procedimientos en la ventana interactiva:

- Seleccionamos alguna y presionamos el botón **Queue**.
- Hacemos un doble clic sobre ella.
- Seleccionamos una secuencia en la **Lista de datos/comandos** y presionamos **Queue**.

Si queremos enviar secuencias automáticamente, podemos usar la terminal **TRIG**; cada vez que se dé un flanco de subida en ella, una secuencia almacenada será enviada por el bus. Para tener un ejemplo de la manera en que se usa el analizador I2C como maestro podemos reemplazar el PIC con un analizador I2C, y desde él enviamos las secuencias necesarias para escribir las ocho primeras letras del alfabeto en la EEPROM. También podemos definir la secuencia en forma de lista.

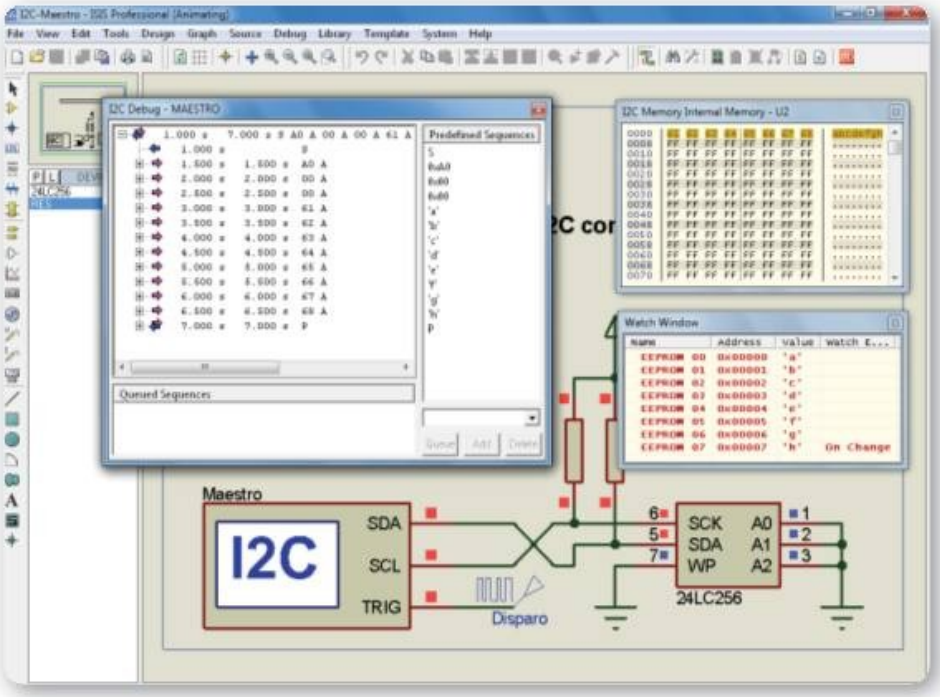


Figura 9. El analizador I2C como dispositivo **maestro** para escribir en la EEPROM.

Podemos ver en este ejemplo cómo efectivamente se escriben los datos en la EEPROM sin necesidad de un microcontrolador. La simulación se pone en pausa de manera automática con una condición de vigilancia (es importante que no nos olvidemos de borrar los datos persistentes).

Analizador I2C como esclavo

Para usar el analizador I2C como esclavo solo hace falta que procedamos a realizar la definición de su dirección como tal en los campos respectivos, así responderá ante las peticiones del maestro en la dirección correspondiente. Por otra parte, debemos indicar también una secuencia de respuesta, que dependerá del dispositivo que intentemos emular.

Debemos tener en cuenta que es posible citar un nuevo ejemplo; esta vez, podemos sustituir la memoria 24LC256 por un analizador I2C que la emulará como esclavo.

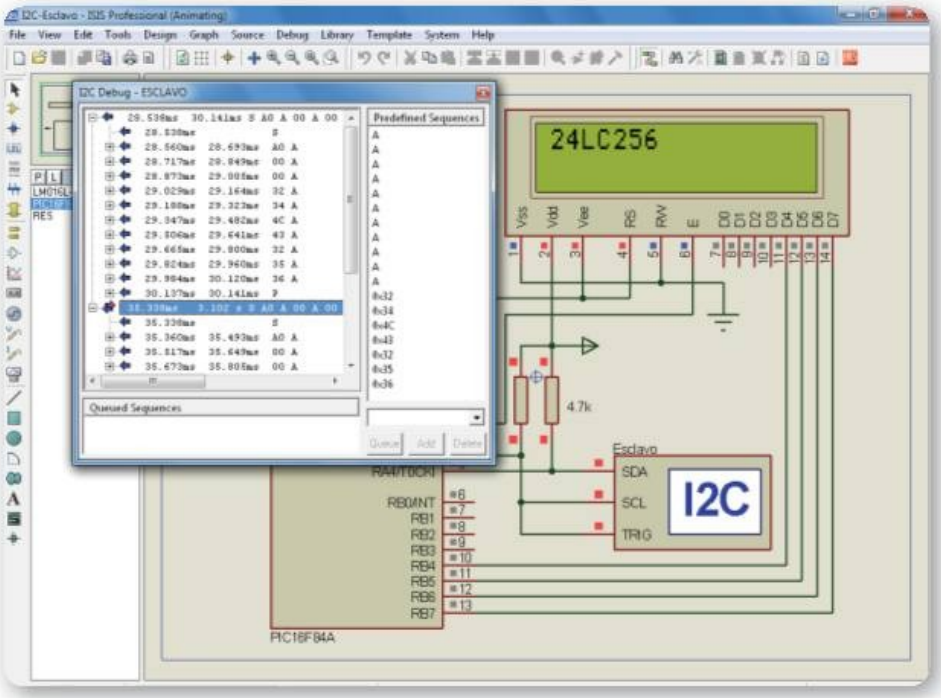
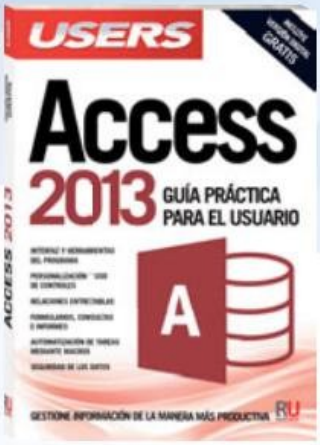


Figura 10. El analizador I2C como dispositivo **esclavo** emulando a la memoria 24LC256.

Hemos conectado la terminal **TRIG** a la línea **SCL** para que la señal de reloj dispare las secuencias de respuesta del analizador cuando se requiera. Definimos una secuencia de respuestas que emula a la memoria 24LC256, de tal modo que el circuito completo funciona exactamente igual que si la memoria estuviera ahí. Colocamos los datos de forma manual, ya que el analizador recibe los datos del PIC, pero como no es una memoria, no los escribirá en su interior, y no habrá lectura real de datos.

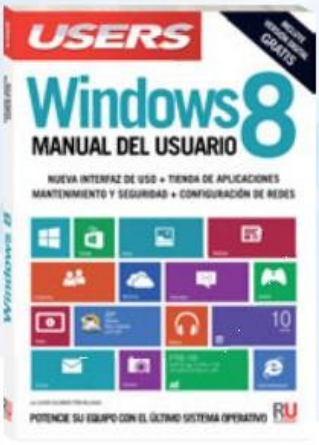
RESUMEN

Una de las ventajas de Proteus es la capacidad de simular circuitos que contienen diversos tipos de microcontroladores, con la posibilidad de simular perfectamente estos dispositivos con todas sus funciones. Además, es posible efectuar una depuración para localizar errores o realizar cambios en los códigos fuente.



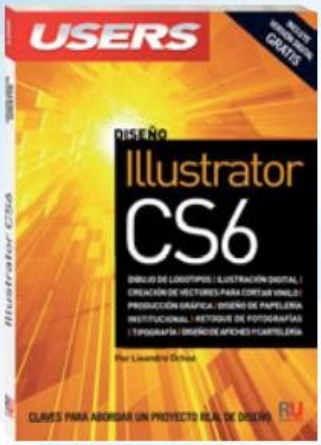
Simplifique tareas cotidianas de la manera más productiva y obtenga información clave para la toma de decisiones.

→ 320 páginas / ISBN 978-987-1949-17-5



Acceda a consejos indispensables y aproveche al máximo el potencial de la última versión del sistema operativo más utilizado.

→ 320 páginas / ISBN 978-987-1949-09-0



La mejor guía a la hora de generar piezas de comunicación gráfica, ya sean para web, dispositivos electrónicos o impresión.

→ 320 páginas / ISBN 978-987-1949-04-5

Aprenda a simplificar su trabajo, convirtiendo sus datos en información necesaria para solucionar diversos problemas cotidianos.

→ 320 páginas / ISBN 978-987-1949-08-3

Un libro ideal para ampliar la funcionalidad de las planillas de Microsoft Excel, desarrollando macros y aplicaciones VBA.

→ 320 páginas / ISBN 978-987-1949-02-1

El libro indicado para enfrentar los desafíos del mundo laboral actual de la mano de un gran sistema administrativo-contable.

→ 352 páginas / ISBN 978-987-1949-01-4

Un libro ideal para ampliar la funcionalidad de las planillas de Microsoft Excel, desarrollando macros y aplicaciones VBA.

→ 320 páginas / ISBN 978-987-1857-99-9

Un libro para maestros que busquen dinamizar su tarea educativa integrando los diferentes recursos que ofrecen las TICs.

→ 320 páginas / ISBN 978-987-1857-95-1

Libro ideal para introducirse en el mundo de la maquetación, aprendiendo técnicas para crear verdaderos diseños profesionales.

→ 352 páginas / ISBN 978-987-1857-74-6

CURSOS INTENSIVOS CON SALIDA LABORAL

Los temas más importantes del universo de la tecnología, desarrollados con la mayor profundidad y con un despliegue visual de alto impacto: explicaciones teóricas, procedimientos paso a paso, videotutoriales, infografías y muchos recursos más.

- 25 Fascículos
- 600 Páginas
- 2 DVDs / 2 Libros

Curso para dominar las principales herramientas del paquete Adobe CS3 y conocer los mejores secretos para diseñar de manera profesional. Ideal para quienes se desempeñan en diseño, publicidad, productos gráficos o sitios web.

Obra teórica y práctica que brinda las habilidades necesarias para convertirse en un profesional en composición, animación y VFX (efectos especiales).

- 25 Fascículos
- 600 Páginas
- 2 CDs / 1 DVD / 1 Libro

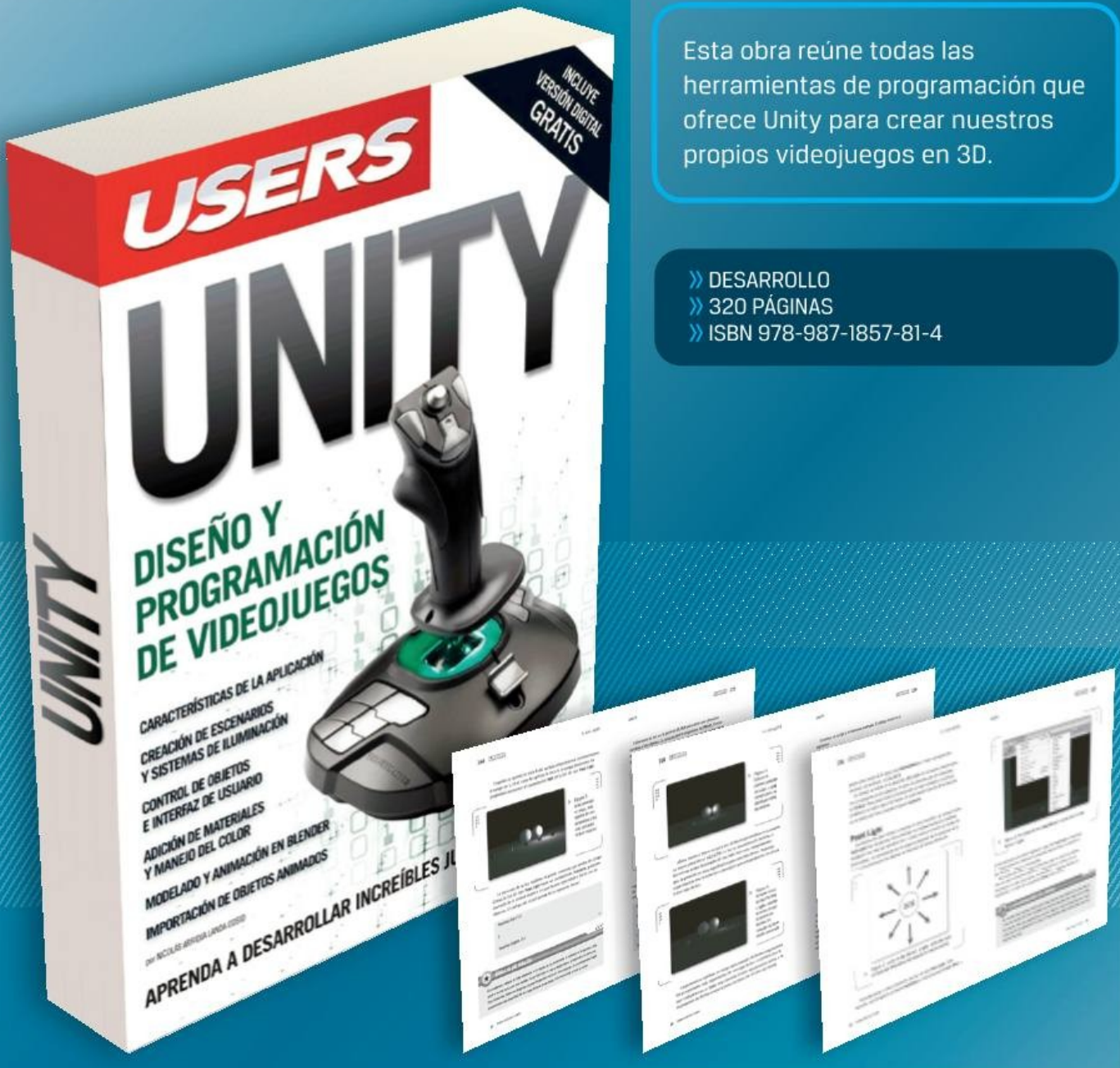
- 25 Fascículos
- 600 Páginas
- 4 CDs

Obra ideal para ingresar en el apasionante universo del diseño web y utilizar Internet para una profesión rentable. Elaborada por los máximos referentes en el área, con infografías y explicaciones muy didácticas.

Brinda las habilidades necesarias para planificar, instalar y administrar redes de computadoras de forma profesional. Basada principalmente en tecnologías Cisco, busca cubrir la creciente necesidad de profesionales.

- 25 Fascículos
- 600 Páginas
- 3 CDs / 1 Libros

APRENDA A DESARROLLAR INCREÍBLES JUEGOS 3D



Esta obra reúne todas las herramientas de programación que ofrece Unity para crear nuestros propios videojuegos en 3D.

» DESARROLLO
» 320 PÁGINAS
» ISBN 978-987-1857-81-4

USERS

TÉCNICO en
ELECTRÓNICA

PROTEUS

Proteus es considerado uno de los mejores y más completos programas para el diseño de circuitos electrónicos en la actualidad, no solo por su capacidad de simulación y análisis sino también por su capacidad de utilizar una gran cantidad de microcontroladores de diferentes familias. Esta obra está dirigida a todos aquellos que tienen conocimientos de electrónica y desean aprender el uso de este poderoso software. En esta obra describiremos la interfaz de ISIS, que es el módulo donde se realiza la simulación, y aprenderemos a dibujar diagramas en él. Esta es la base para poder simular cualquier circuito, porque dibujar correctamente un diagrama facilita y acelera el proceso de creación. Conoceremos una gran cantidad de opciones que Proteus pone a nuestra disposición para hacerlo de forma eficiente y veremos las diferentes opciones de simulación de Proteus.

CONTENIDO

EXCLUSIVO PARA LECTORES

Profesores en línea:
profesor@redusers.com

TÉCNICO en ELECTRÓNICA
9 789871 949151