REVISTA DIGITAL
"INVESTIGACIÓN Y EDUCACIÓN"

NÚMERO 19

SEPTIEMBRE DE 2005

ISSN 1696-7208

La simulación en TkGate, el simulador de circuitos digitales para Linux.

Ricardo Valerio Bautista Cuéllar

Tras varios artículos dedicados al simulador de circuitos digitales TkGate para Linux, este es el momento en que nos centraremos en las opciones de simulación que la herramienta presenta.

Pretendemos aquí mostrar con detalle pero sin demasiada profundidad todas y cada una de las capacidades que la herramienta ofrece al diseñador para comprobar el funcionamiento de sus circuitos mediante esta potente herramienta de simulación.

Como se supone unos conocimientos previos mínimos sobre TkGate, recomiendo a aquellos lectores que no hayan podido seguir los artículos previos sobre TkGate que los lean antes de comenzar la lectura de este artículo, pues corren el riesgo de perder el contexto y el sentido de lo expuesto.

El texto aquí, por tanto, pretende compendiar todos los aspectos relativos a la simulación con TkGate (scripts de simulación, ficheros de retraso, análisis de caminos críticos...) para poder constituir una guía de acceso rápido sobre la simulación y análisis en TkGate para aquellos compañeros que quieran comenzar a emplear esta herramienta en su trabajo en clase.

Introducción

En este artículo nos centraremos en las capacidades de simulación de la herramienta TkGate. Ya hemos comentado con anterioridad en artículos previos sobre la materia que una de las características más importantes de TkGate es la posibilidad de implementar scripts para poder realizar simulaciones e iteraciones sobre los circuitos de forma que podamos conocer los valores y los impactos de ciertas modificaciones en el circuito. Aquí daremos algunas recomendaciones de cómo realizar esto así como explicaremos todas y cada una de las posibilidades que ofrece esta herramienta gratuita a la hora de ejecutar una simulación y conocer el funcionamiento de un circuito diseñado por nosotros.

Iniciando el simulador.

Los controles del simulador pueden ser accedidos tanto mediante el menú “Simulate” o desde la propia barra de botones. Comienza una simulación seleccionando “Begin Simulation” desde el menú “Simulate” o presionando el botón “play” en la barra de botones. Una ventana de rastreo aparecerá cuando comienzas el simulador, también ventanas de texto para los dispositivos “tty” en el circuito. Si hay scripts autoejecutables estos serán ejecutados también.

Las simulación podrá ser realizada con el módulo raíz seleccionado al más alto nivel. El simulador internamente expande cualquier instancia del módulo en el circuito. Puesto que el camino que toma para llegar a un módulo es significativo para el simulador, no puedes saltar

directamente a submódulos si no navegar a ellos seleccionando un módulo del nivel que estés usando para abrirlo usando el menú adecuado.

TKGate es un simulador dirigido por eventos. El tiempo es medido en unidades discretas denominadas “epochs”. Cada puerta tiene un retraso de una cierta cantidad de epochs. Algunas puertas complejas tienen varias constantes de retraso. Además, algunas puertas tales como registros y memorias tienen parámetros adicionales de retraso que afectan a cambios internos de estado.

Los comandos del simulador básicos son:

- Run: Que permite entrar en el modo de simulación. La simulación continua mientras que existan eventos en la cola de eventos. Si existen puertas de reloj en el circuito, esto significa que la simulación continuará de forma indefinida. Si el circuito es combinacional, la simulación continuará hasta que se alcance el estado estable.
- Pause: Origina que una simulación que está en progreso se detenga.
- Step Epoch: Esto se emplea para hacer avanzar la simulación un número fijo de epochs. El número de epochs que avanza puede ser establecido en las opciones del menú de simulación. Puedes también invocar este comando con al barra espaciadora.
- Step Cycle: Origina que la simulación avance al filo de subida de un reloj. Puedes establecer el número de ciclos de reloj para simular y el número de epochs que deben pasar para un ciclo. Se puede invocar este comando con la tecla de tabulación.

- End Simulation: Origina que la simulación sea terminada y todas las sondas sean borradas.

Opciones del simulador.

Varias opciones de simulación pueden ser establecidas a través de la caja de diálogo de opciones. Para editar las opciones del simulador, selecciona “Options...” desde el menú “File”, y luego selecciona el pestaña “Simulator”.

Las opciones del simulador son:

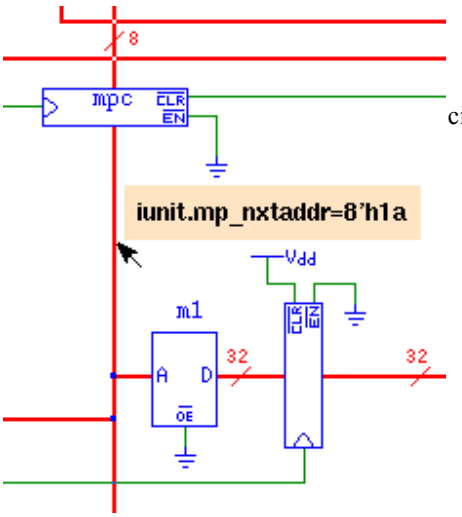
- Epoch Step Size: Especifica el número de epochs para avanzar el simulador cada vez que se avance un paso empleando el botón de avance o mediante la barra espaciadora.
- Clock Cycle Step Size: Especifica el número de ciclos de reloj para avanzar cada vez el reloj mediante la tecla de tabulación.
- Clock Overstep: Especifica el número de epochs para simular passado el flanco de reloj cuando estamos haciendo paso del reloj.
- Initialization Script: Especifica el script de simulación para ejecutar de forma automáticamente cuando se comienza la simulación. El fichero script especificado aquí es una propiedad global y se aplica a cualquier circuito que ha sido cargado en el TkGate.
- Clock sep stops on all clock posedges: Indica que el comando de paso de reloj debería ser disparado en los flancos positivos en todos los relojes del circuito.

- Clock sep on clock: Indica que el comando de paso de reloj debe ser disparado en los flancos positivos sólo en el reloj especificado. Esta opción sólo es útil cuando tenemos circuitos con varios relojes.
- Delay Files: Especifica ficheros adicionales de los cuales cargar especificaciones de retrasos de puerta.

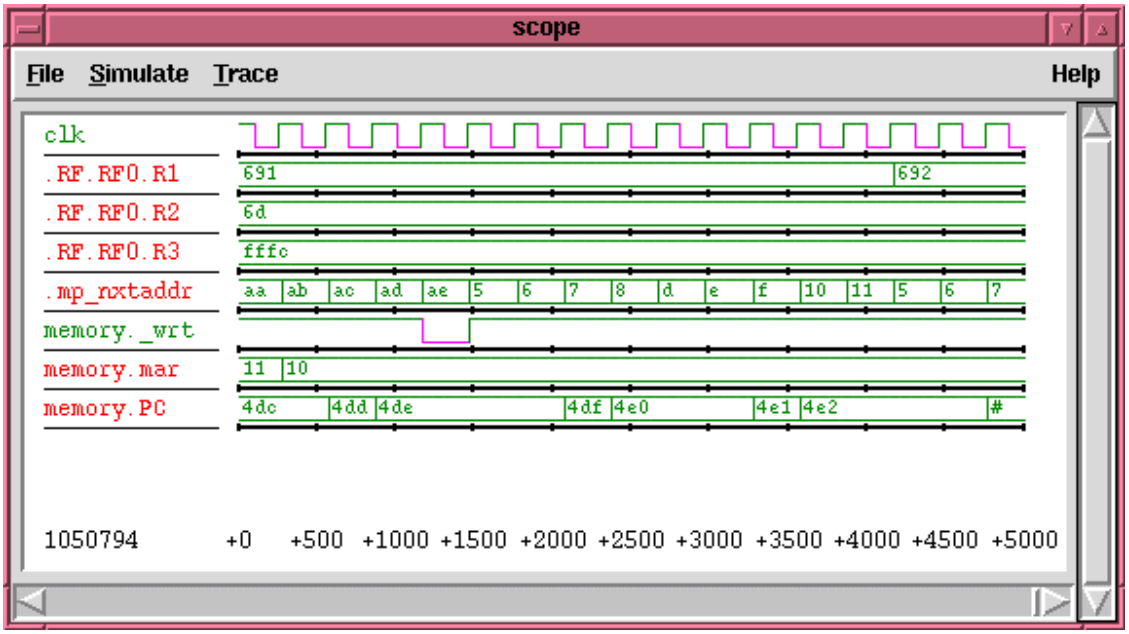
Observando la salida.

Para mostrar y especificar valores se emplea sintaxis veilog. Un número en la sintaxis Verilog contiene un prefijo para especificar el ancho de bit, un carácter comilla, un carácter para la base y los dígitos del número. Los caracteres para la base usados en TkGate son “b” para binario y “h” para hexadecimal. Por ejemplo, “8”he” es el número hexadecimal de 8 bits 3e. El simulador soporta multitud de valores lógicos (0, 1, x(desconocido), z(flotante), L (bajo), H (alto)...).

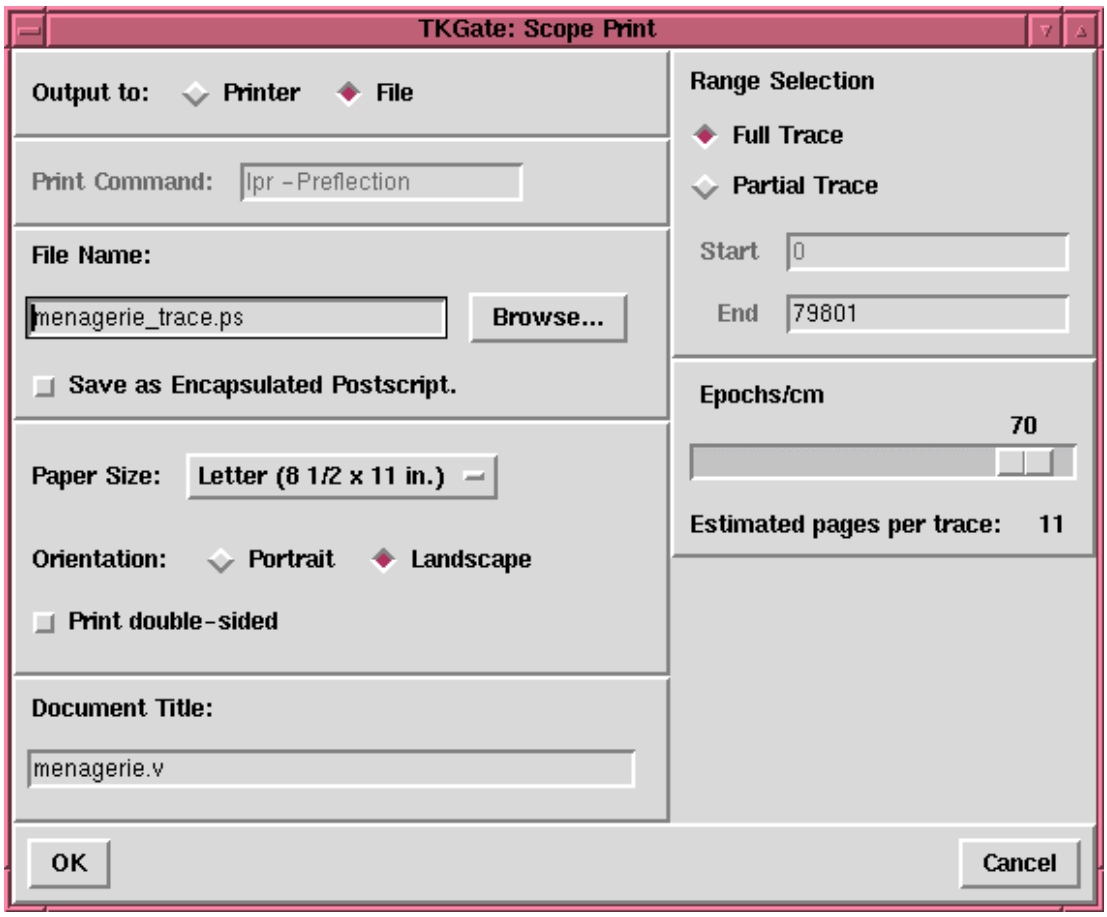
Para mostrar el valor de una señal en un circuito, haz clic y mantén pulsado el botón del ratón en el cable. Esto mostrará el valor conducido en el cable mediante sintaxis verilog. El valor desaparecerá cuando liberas el botón del ratón. Esta característica puede ser empleada tanto cuando la simulación está pausada como cuando está en modo de simulación contínua. Cuando el simulador es en el modo de simulación continua, el valor mostrado será el valor en el momento que el botón del ratón fue pulsado por primera vez.



Para establecer un punto de prueba permanente en una señal, haz doble clic en un cable. Esto añadirá o eliminará el punto de prueba. Cuando un punto de prueba es establecido en un cable, su valor será mostrado de forma contínua en la ventana objetivo. Los valores del las señales multi-bit serán mostrados como números hexadecimales. Para cambiar el rango de los valores mostrados y la escala de la ventana puedes emplear la barra de desplazamiento, el ratón o los comandos de teclado. Moviendo la barra de desplazamiento o establecerás el rango relativo a la simulación entera. Puedes obtener un control más preciso mediante un clic en la ventana de traceo con el botón izquierdo del ratón y desplazando la traza a la izquierda o a la derecha. Para hacer zoom, se puede tanto presionar el botón izquierdo del ratón mientras se mantiene la tecla shift pulsada o presionar la tecla ">". Para hacer zoom out, puedes emplear el botón derecho del ratón o la tecla "<".



Para imprimir una traza, elige la opción “Print...” del menú “File” en la ventana de interés o emplea la combinación “Ctl-s p”. Una caja de diálogo aparece. La parte mitad izquierda de la caja de diálogo es la misma que la mitad izquierda de la caja de diálogo empleada para imprimir circuitos. La parte derecha incluye opciones para seleccionar el rango temporal para imprimir y el factor de escala a emplear. Hay que tener en cuenta que las trazas de la simulación pueden crecer mucho, rápidamente y especialmente cuando se emplea el modo de simulación continuo. Una estimación del número de páginas necesarias para cada traza es mostrada como ayuda entre los parámetros.

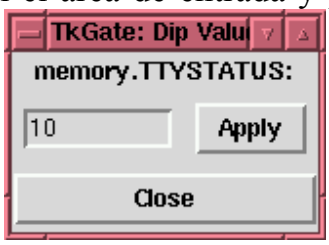


Cómo controlar la entrada de datos.

Existen tres tipos de elementos circuitales que pueden ser empleados para controlar el circuito: el switch de un único bit, el dip switch y el tty. Estas ya han sido comentadas en [9] pero aquí incidiremos en sus características.

Switches y Dip Switches.

Los switches o interruptores pueden ser manipulados simplemente haciendo clic en ellos para modificar sus valores. Para cambiar un valor de dipswitch, haz clic en el mismo para abrir una caja de diálogo para establecer el valor deseado en el área de entrada y presionando el botón “apply” para establecer el valor elegido.



TTys

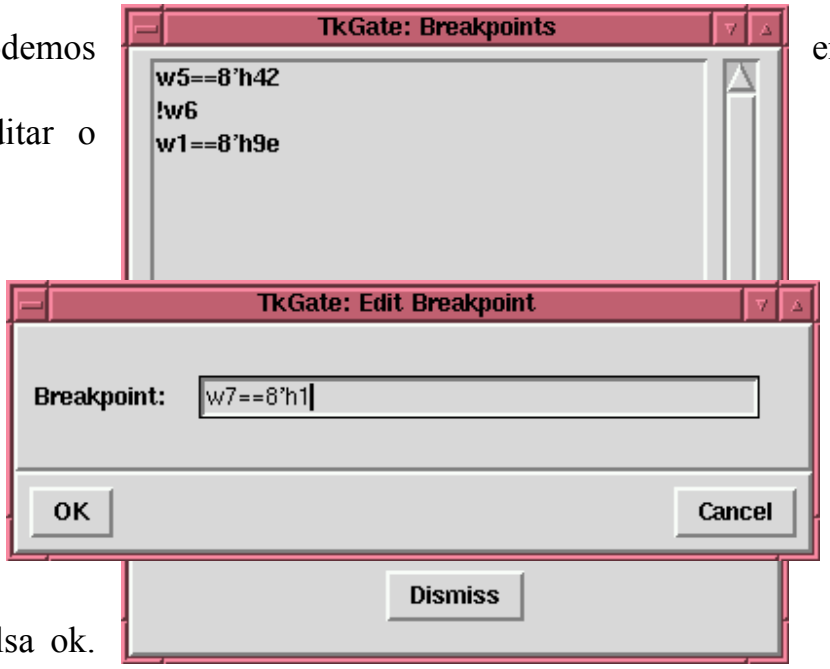
El tercer tipo de elemento de entrada de datos es éste. El tty es realmente no sólo una entrada si no también una salida, simulando o modelando un terminal interactivo. Cuando comienzas el simulador, una ventana tty aparecerá para cada realización concreta de una puerta tty en el circuito. Tu circuito puede enviar caracteres y recibir caracteres desde la puerta tty.

Para enviar una carácter a un elemento tty, maneja la entrada RD con el código ascii del carácter a transmitir y espera la salida DTR que salga. Luego mantén la señal DSR. El carácter será transmitido en el flanco positivo de la señal DSR.

Para recibir un carácter, primero libera la señal CTS y espera que la señal RTS sea alta. Entonces podremos leer el valor en la salida TD del tty. Una vez que hayas leído el valor, asegura la señal CTS para indicar que has recibido el carácter.

Estableciendo puntos de ruptura.

Los puntos de ruptura pueden ser empleados para fijar condiciones que causarán que una simulación en marcha se detenga. Para mostrar la caja de diálogo deel punto de ruptura, elige la opción " Breakpoint..." del menú "Simulate". Esto causará que se muestre el editor de Breakpoints. Podemos emplear el botón para añadir, editar o elminar breakpoints.



Cuando añadimos o editamos un punto de ruptura, una caja de diálogo se mostrará. Entra o edita la condición para el punto de ruptura y pulsa ok. Los operadores relacionales son los operadores relacionales empleados en C. Los valores a emplear deben seguir la sintaxis verilog que hemos explicado anteriormente. De forma alternativa, podemos simplemente teclear un nombre de señal para producir la ruptura

cuando la señal llega a ser no cero o mediante el operador '!' delante del nombre de la señal para indicar que la ruptura debe producirse cuando la señal se hace cero.

Inicializando memorias.

Un circuito puede contener una o más memorias (las puertas ROM y RAM definidas en [9]). Puedes inicializar memorias de un fichero o bien volcar contenidos de memoria en un fichero. Emplea las opciones que damos a continuación para cargar o volcar memorias:

- Load Memory: Carga memorias desde el fichero seleccionado. Si una puerta de memoria es seleccionada, la memoria será la memoria por defecto a cargar. Si el fichero de memoria contiene uno o más parabras "memory", las memorias especificadas serán cargadas con los contenidos del fichero. Cuando cargamos un fichero, el directorio actual, el del circuito y el de usuario serán los directorios donde se buscará.
- Dump Memory: Vuelca el contenido de una memoria seleccionada en un fichero.

Los ficheros de memoria tienen extensión por defecto ".mem". Un fichero de memoria está compuesto de un número de líneas. Líneas en blanco y líneas empezando con '#' son ignorados. Otras líneas deben ser una declaración de memoria o una dirección en hexadecimal seguida por un guión y una lista de valores de memoria en hexadecimal. Un ejemplo de un fichero de memoria puede ser:

```
100/ e1 f0 0 0 e1 e0 0 0
108/ 81 0 0 0 12 1 bd 0
```

```
110/ e 1 e1 d0 dc 7 85 0
118/ 6f 6 81 0 0 0 4e 4
120/ 69 f0 2 0 85 0 64 0
128/ 81 0 0 0 26 4 69 f0
130/ 2 0 ed 0 60 6 62 6
138/ ed 0 5e 6 1 0 85 0
```

La dirección al principio de la línea especifica la dirección de memoria en la que comenzar a almacenar los valores. Los valores son almacenados secuencialmente desde la dirección especificada.

Si no hay ninguna palabra memory en el fichero, la puerta de memoria seleccionada será cargada. La palabra memory requiere un argumento único especificando el nombre de una memoria. Por ejemplo:

```
Memory memory.m1
100/ e1 f0 0 0 e1 e0 0 0
108/ 81 0 0 0 12 1 bd 0
```

Cargará la mermoria m1 en el elemento de memoria denominado “memory” que es un submódulo del módulo raíz. Con el fin de ayudar a la creación de ficheros de memoria, existe una utilidad denominada gmac para compilar descripciones de microcódigo y macrocódigo en ficheros de memoria compatibles con TkGate.

Para memorias RAM, los contenidos son congelados mientras no se produzca cambio en la línea de selección de escritura. Esto evita que los datos sean destruidos con accesos no deseados.

Scripts de simulación

Al igual que en Matlab se emplean scripts para ejecutar cálculos complejos que requieren diversos pasos para realizarse, en TkGate, tal como en otros programas, se pueden emplear ficheros sript para realizar la mayor parte de las operaciones que pueden realizarse manualmente. Puedes establecer y eliminar puntos de prueba, cambiar valores de interruptores, cargar memorias, establecer puntos de ruptura y parar o arrancar la simulación. Los scripts de simulación son útiles para establecer una simulación antes de empezar o para arrancar una simulación en modo batch. Podemos cargar una script de simulación mediante el botón de la barra de botones o seleccionando “Exec. Script...” del menú “Simulate”. La extensión por defecto para scripts de simulación es “.gss”. Cuando cargamos un fichero, el directorio actual, el directorio del circuito actual y el de usuario serán usados en la búsqueda. También se puede establecer que scripts de simulación sean automáticamente ejecutados cuando comenzamos una simulación especificando un script de simulación en la caja de diálogo de opciones de simulación o añadiendo scripts de simulación como opciones de circuito. Los siguientes comandos son reconocidos en los scripts de simulación. Argumentos especificados entre caracteres ‘?’ indican argumento opcional.

- include “file”. Incluye el fichero como si el contenido estuviera a partir de esa línea. Permite simplificar la ejecución de varias operaciones repetitivas.
- Sep n: Avanza el número de epochs establecido.

- Clock(+/-) ?[name]?n?+m? Avanza n pasos de reloj. Si el carácter tras clock es + el avance es en los flancos positivos de reloj. Si name es especificado, sólo los flancos en ese clock serán empleados. Si se especifica m, el simulador avanzará ese número de epochs tras el flanco final de reloj.
- Run: Pone el simulador en el modo de simulación contínua.
- Break ?[name]?cond: Establece puntos de ruptura mediante condiciones que se definen en cond de la misma forma que habíamos comentado se definían en los cuadros de diálogo para breakpoint. Si un name es definido entre corchetes, ese breakpoint será asignado un simbólico de forma que puede ser referido posteriormente para borrar el punto de ruptura. Si no fuera así, no habría modo de eliminar puntos de ruptura desde un script de simulación.
- Delete [name]: Borra el punto de ruptura especificado en name.
- Set name value: Establece el valor de un interruptor o de un dip switch. El nombre debe ser el de un switch o un dip switch del circuito y value debe estar en la sintaxis verilog explicada anteriormente.
- Probe name: Establece un punto de prueba en la señal denominada name en el circuito.
- Unprobe name: Elimina el punto de prueba establecido mediante probe.
- Load ?name? “file”: Carga memorias desde fichero. Si especificamos un nombre de memoria en name, la memoria especificada será la memoria por defecto.
- Dump name “file”: Vuelca la memoria “name” en el fichero “file”.

- Zoom n: Establece el factor de zoom para la ventana de traza.

Reporte de Errores

Cuando lanzamos la simulación, si el circuito o la simulación presentan errores, una caja con la lista de errores se mostrará con los errores y los módulos afectados por dichos errores. Bastará observar dichos módulos para comprobar el error.

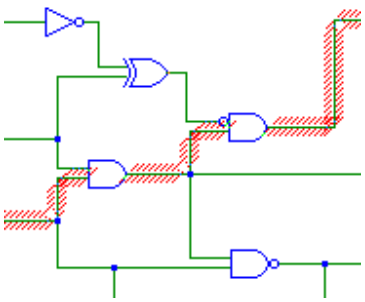
Ficheros de retraso de puerta.

Parámetros de retraso de puerta, de área y de potencia pueden ser especificados mediante una colección de ficheros de especificación gdf (Gate Delay File). El fichero por defecto "gdf/default.gdf" en el directorio base de TkGate está siempre cargado pero las definiciones pueden ser cambiadas cargando ficheros adicionales para retrasos a través de las opciones de retraso.

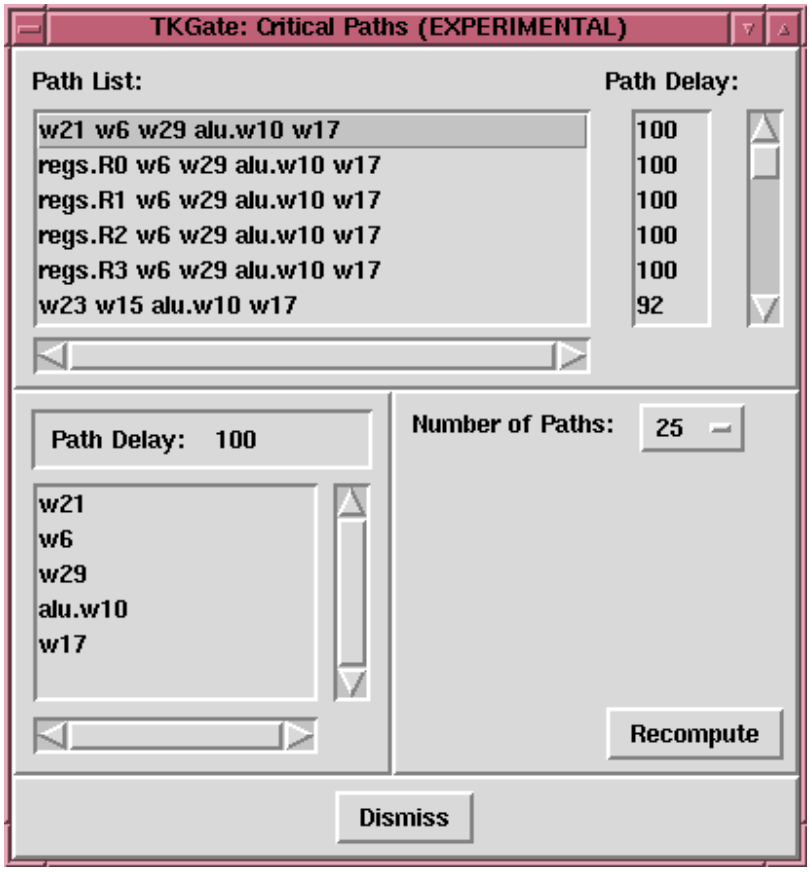
Debido a la complejidad de la definición de estos ficheros de retraso, los abordaremos en un artículo posterior.

Análisis de camino crítico.

El simulador TkGate incorpora un analizador de circuitos para el análisis de caminos críticos. Hacer notar que el analizador emplea un algoritmo de camino crítico estático y no puede detectar caminos falsos. El analizados encontrará y mostrará una lista de un número especificado de caminos de retraso largo en el circuito. Solo caminos que son formados encontrando los retrasos en el peor caso como salida y el peor caso como entrada para algunos nodos internos son listados.



Para invocar el analizador, haz clic en el botón de análisis de camino crítico en el lado derecho de la barra de herramientas o selecciona "Critical Path..." del menú "Circuit". Un cuadro de diálogo como el mostrado aparece mostrando los caminos críticos y sus retrasos. Puedes seleccionar un camino para que sea mostrado en el circuito.



Bibliografía

[1] “Herramientas de simulación de circuitos para LINUX.” Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

[2] “Introducción a gEDA.Una herramientas de simulación de circuitos para LINUX”. Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

[3] “gEDA. La simulación SPICE“. Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

[4] “Elaborando circuitos impresos empleando gEDA. Las herramientas de síntesis de circuitos para LINUX “. Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

[5] “Sigamos reduciendo costes. gschem de gEDA. Captura de esquemáticos en Linux“. Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

[6] “gschem de gEDA (II). Captura de esquemáticos en Linux“. Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

[7] “Herramientas electrónicas para LINUX. Circuitos digitales. TKGate“. Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

[8] “Trabajando con módulos y puertas en TKGate, el simulador de circuitos digitales para LINUX”. Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

[9] “Puertas, impresión de esquemáticos e hiperenlaces en el simulador de circuitos digitales para LINUX TKGate“. Revista Digital Investigación y Educación. Ricardo Bautista Cuellar.

Enlaces de utilidad

- Tkgate: http://www.tkgate.org/
- Vipec: http://vipec.sourceforge.net
- PCB: http://bach.ece.jhu.edu/~haceaton/pcb/

- The gEDA project:http:/www.geda.seul.org
- SPICE3 syntax and commands: http://newton.ex.ac.uk/teaching/CDHW/Electronics2/userguide/
- Ngspice: http://ngspice.sourceforge.net/
- Tclspice: http://tclspice.sourceforge.net/
- LTSpice: http://www.linear.com/software/
- Spice on Linux resources: http://www.brorson.com/gEDA/SPICE/
- Free Dog -- The Free EDA Users Group: http://www.freeedaug.org/