

MANUAL DEL ENTORNO UVI-51-win DE ENSEÑANZA DEL MICROCONTROLADOR 8051 (v 1.0)

Alfredo del Río
Dpto. de Tecnología Electrónica
ETSII-Universidad de Vigo
36200 VIGO
Spain

If you are interested in an English version of this manual, this software, or both, please send an e-mail to the author: ario@uvigo.es.

Contenido

1. Introducción

2. Principales diferencias con respecto al SW UVI51 5.0 (MS-DOS)

3. Instalación

4. Asistente Uvi51

5. Ensamblador Uviens51

5.1 Introducción

5.2 Ejecución del ensamblador

5.3 Sintaxis del programa fuente

5.4 Limitaciones del ensamblador

6. Simulador Uvisim

6.1 Pantalla principal y manejo básico del simulador

6.2 Ventana del sistema

6.3 Ventana del microcontrolador

6.4 Ventana informativa de temporizadores

6.5 Ventana informativa de interrupciones

6.6 Ventana informativa del puerto serie

6.7 Edición de valores

6.8 Configuración simulada del 8051

7. Visualizador del analizador lógico (versión MSDOS)

7.1 Introducción

7.2 Descripción de la pantalla del visualizador

7.3 Búsqueda de patrones en el cronograma

Apéndice A1: Sintaxis de los archivos CNX

A1.1 Definición de microcontroladores

A1.2 Definición de periféricos

Apéndice A2: Personalización del entorno UVI51

Apéndice A3: Soporte técnico

1. Introducción

El entorno de enseñanza del microcontrolador 8051, denominado UVI-51-win, ha sido desarrollado en el Departamento de Tecnología Electrónica de la Universidad de Vigo (España), con el objetivo de facilitar la tarea de enseñanza/aprendizaje sobre el funcionamiento del microcontrolador 8051 de la firma Intel. Está basado en la versión 5.0 de UVI-51 para MSDOS.

Este entorno lo componen un conjunto de programas y archivos complementarios. Sus elementos principales son el asistente UVI51, el ensamblador UVIENS51, el simulador UVISIM y el analizador lógico UVILOGIC. En la versión actual UVI51-win 1.1, el asistente, el simulador y el analizador lógico son aplicaciones de tipo **Windows Framework v2.0**, mientras que se mantiene la antigua versión MS-DOS de UVIENS51. El asistente es el punto de entrada para las demás aplicaciones, aunque también es posible acceder a las aplicaciones directamente. En los apartados siguientes se describe de forma breve su manejo. El presente manual se suministra con la versión win 1.1, que es de libre distribución (freeware).

Este manual no explica los detalles de hardware ni de software del 8051. Por el contrario, presupone en el lector un conocimiento básico del mismo.

2. Principales diferencias con respecto al SW UVI51 5.0 (MSDOS)

La versión win está inspirada en la antigua versión MSDOS, pero presenta algunas diferencias con respecto a aquella. Por ejemplo, el aspecto de la interfaz gráfica se mantiene en lo sustancial, pero adquiere ahora un estilo windows. Las principales diferencias se resumen en la siguiente tabla.

Característica	Versión MS-DOS	Versión windows
ASISTENTE:		
Acceso al asistente	Línea de comando o acceso directo desde windows.	Acceso directo o doble click sobre archivo de tipo CNX
SIMULADOR		
Presentación general en pantalla	Pantalla completa. No se pueden observar varias ventanas de micro simultáneamente. Las ventanas de ayuda ocultan la ventana principal del micro.	Ventanas independientes para el sistema, para micros, y para ayuda de periféricos.
Presentación de pantallas LCD	Cada LCD ocupa 8 celdillas	Cada LCD ocupa 4 celdillas
Numeración de nodos	De 0 a 201.	De 0 a 199.
Resolución medida de tiempos	100 nanosegundos	1 nanosegundo
Modos	Monomicro y Multimicro	Multimicro solamente

3. Instalación

UVI51-win ha sido desarrollado para la plataforma “Windows Framework v2.0”, por lo que no precisa un programa de instalación. Sin embargo, para poder utilizarla es preciso haber instalado previamente en el ordenador personal la versión 2.0 de Windows Framework, que se puede obtener de la web de Microsoft de forma gratuita. Para instalar UVI51-win basta con seguir los siguientes pasos:

1.- Crear una carpeta en C:\ con el nombre UviCfg

2.- En esa carpeta (C:\UviCfg) copiar el archivo de configuración facilitado (Uvi51Cfg.txt).

3.- Este archivo (Uvi51Cfg.txt) tiene formato de texto y se puede editar para adaptarlo a las preferencias del usuario. Contiene 8 líneas. La primera línea indica el path de la carpeta en la que se instalan todos los componentes de Uvi51-win, por ejemplo C\uvi51w. Las restantes líneas indican los nombres de los programas (o archivos BAT) que se van a emplear para simular, depurar, editar archivos CNX, SRC y LST, activar el analizador lógico, acceder al programa terminal y ensamblar, respectivamente. alguna de estas opciones podría estar inactiva en ciertas configuraciones. Un ejemplo del contenido del archivo puede ser:

```
C:\uvi51w
Uvisim.exe
Uvialt.exe
notepad.exe
Uvilogic.exe
Uviterm.exe
ensambla.bat
compilar.bat
```

NOTA: El asistente llama a cada uno de estos programas pasándoles un parámetro que indica el archivo que van a manejar. Por ejemplo test.src para editar el archivo fuente con ese nombre.

En este otro ejemplo, se emplea el editor Crimson (freeware) en vez del notepad, y se llama a un ensamblador por medio de un archivo BAT.

```
C:\uvi51w
Uvisim.exe
Uvialt.exe
"c:\archivos de programa\Crimson Editor\cedt.exe"
graf51pt.exe
terminal.exe
ensambla.bat
compilar.bat
```

NOTA: Solamente en el caso de que en la línea 7 se indique “ensambla.bat”, el asistente hará una llamada al archivo bat pasando dos parámetros. El primer parámetro indica el nombre del archivo a ensamblar (path completo). El segundo parámetro indica el path de UVI51 (el que se indicó en la primera línea del archivo de configuración). Este segundo parámetro facilita la creación del archivo bat. Por ejemplo, para emplear el ensamblador asm51 de Intel, situado en la carpeta UVI51, se puede utilizar el archivo ensambla.bat siguiente:

```
@echo off
cd %2
copy %1.src /A %1.srb /B
```

```
rem suprime eco
rem cambia a la carpeta donde está asm51
rem adapta el formato a tipo binario (asm51 lo precisa)
```

```

%2asm51 %1.srb gen
%2hayerror %1
if errorlevel 1 goto salir
%2oh %1.obj to %1.hex > obj2hex
if errorlevel 1 goto salir
%2uvides51 %1.hex
:salir
del %1.srb
pause

```

```

rem llama a asm51 con el parámetro adecuado
rem llama al programa hayerror que detecta errores
rem verifica si hay error de ensamblado
rem convierte el archivo obj en código absoluto hex
rem vuelve a comprobar si hay error
rem llama al desensamblador de UVI51
rem sale
rem borra archivo de trabajo
rem espera a que el usuario lea el resultado

```

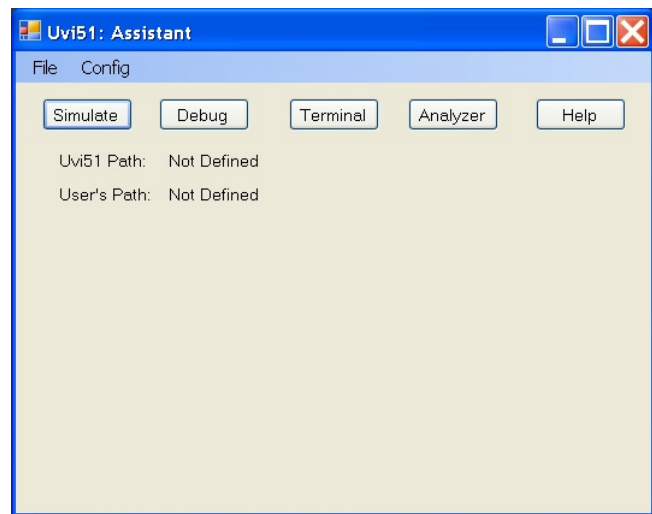
Ver también apéndice A2.

4. Asistente Uvi51

Para ejecutar el asistente se puede emplear un acceso directo, o simplemente hacer doble click sobre un archivo de tipo CNX. Los archivos CNX son archivos de texto que contienen información sobre todos los micros que se definen, así como de los periféricos externos y su conexionado. Para más detalles sobre los archivos CNX consultar el apéndice A1.

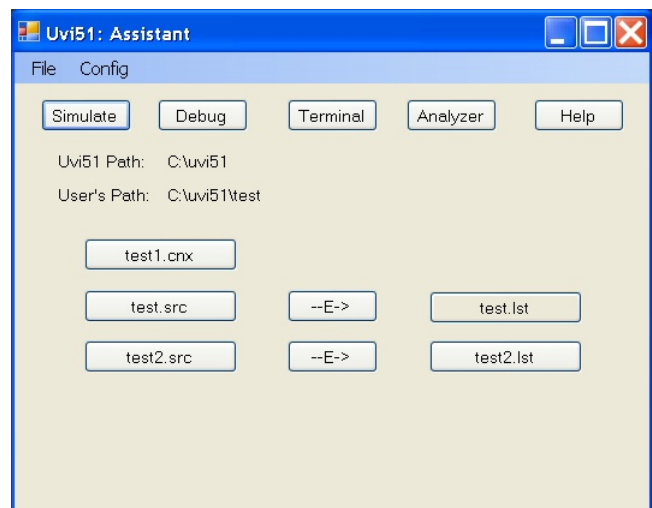
Si se ha entrado en el asistente sin indicar un archivo CNX como parámetro, se obtiene la ventana de la derecha.

El siguiente paso consiste en utilizar el menú File-Open CNX. De esta forma, se abre una ventana para seleccionar el CNX.



La ventana de la derecha muestra el aspecto del asistente una vez seleccionado el CNX, que equivale a haber empleado directamente el método del doble click. El asistente muestra los paths de UVI51 (archivos de programa) y de usuario (programas de los micros).

En la parte inferior de la ventana hay un conjunto de botones: La columna de la izquierda empieza con el botón "test.cnx" que se pulsa para editar el archivo CNX. Los archivos "test.src" y "test2.src" se pulsar para editar los programas fuente de cada micro. El número y nombres de los botones de edición del tipo *.src corresponden al



número de micros definidos en el archivo CNX y sus nombres. Los botones de la columna central "--E->" se emplean para llamar al ensamblador. Hay un botón para cada micro. Finalmente los botones de la columna derecha permiten abrir los archivos LST generados por el ensamblador para observar los errores de ensamblado. Cuando hay errores de ensamblado o se ha modificado el archivo fuente sin realizar después el ensamblado, el botón del tipo "lst" correspondiente se

muestra en color rojo. En estas condiciones no se permite dimular el sistema, como se indica en el siguiente párrafo.

En la parte superior de la ventana se muestran 5 botones que se emplean para:

1. Simular el sistema
2. Depurar el código del primer micro mediante la placa Altair. Esta opción puede no estar disponible.
3. Iniciar un programa terminal del puerto serie del PC que puede ser útil en algunos proyectos.
4. Iniciar el analizador lógico.
5. Editar un texto de ayuda con información del 8051.

Las opciones 1, 2 pueden estar disponibles o no, dependiendo del estado de los ensamblados. La opción 4 no estará disponible si no se ha realizado una simulación previa empleando un "analizador lógico" en el archivo CNX (ver apéndice A1).

Cuando se llama a una utilidad (simulador, etc.) empleando los botones, el asistente queda detenido y no reanuda su ejecución hasta que se sale de la utilidad. En ciertas condiciones puede fallar el redibujo de la ventana del asistente en la pantalla. Esta situación carece de importancia, y el asistente se redibuja correctamente cuando recupera su actividad.

Para salir del asistente se pulsa el botón de cierre de su ventana.

5. Ensamblador Uviens51

En este apartado se describe el ensamblador que se incluye con el entorno UVI51. En esta versión de UVI-51-win se emplea el mismo ensamblador que en la versión UVI51 5.0 para MSDOS.

5.1. Introducción

El ensamblador UVIENS51 es un "mini-ensamblador" para la familia 8051. Ha sido creado para su uso en combinación con el entorno UVI-51-win. Puesto que su principal aplicación se encuentra en la enseñanza de microcontroladores, se caracteriza por su sencillez. A pesar de sus limitaciones, es una herramienta útil incluso para desarrollar aplicaciones.

5.2. Ejecución del ensamblador

El programa UVIENS51 se ejecuta mediante un botón [--E->] del entorno. Puesto que se trata de una aplicación MS-DOS, se puede ejecutar también desde la línea de comandos mediante:

UVIENS51 fuente

donde fuente es el *path* del archivo fuente a ensamblar. Debe indicarse la extensión (habitualmente SRC; no puede ser LST ni HEX).

El ensamblador genera un archivo de salida con el mismo nombre (y *path*) pero con la extensión LST. Además, si no encuentra errores, genera directamente un archivo (igual nombre y

path pero extensión HEX) en formato INTEL-HEX que contiene el código objeto absoluto (no reubicable).

El archivo LST resulta útil para buscar la descripción de los errores encontrados. Además, UVISIMOB utiliza este archivo para relacionar el código generado con el código fuente.

5.3. Sintaxis del programa fuente

UVIENS51 acepta archivos fuente en formato ASCII. No es un macroensamblador, es decir, no permite definir ni utilizar macroinstrucciones. Además, genera directamente código absoluto ejecutable a partir de un archivo fuente. Por lo tanto, no permite ensamblar módulos independientes y enlazarlos a continuación con un editor de enlaces (linker).

De lo dicho anteriormente se desprende que muchas de las directivas existentes en un macroensamblador profesional carecen de sentido en UVIENS51. Solamente se definen las siguientes, con una sintaxis y significado similar al del ensamblador ASM51 de Intel:

<u>Directiva</u>	<u>Función que realiza</u>
ORG valor	Actualiza el contador de programa.
#if valor	Ensamblado condicional.
#endif	Fin del ensamblado condicional.
símbolo EQU valor	Define un símbolo de tipo número en función de un valor numérico o de otro símbolo ya definido (1).
símbolo BIT valor	Define un símbolo de tipo dirección de bit en función de un valor numérico o de otro símbolo ya definido.
símbolo DATA valor	Define un símbolo de tipo dirección de dato (RAM) en función de un valor numérico o de otro símbolo ya definido.
símbolo CODE valor	Define un símbolo de tipo dirección de código en función de un valor numérico o de otro símbolo ya definido
DB valor	Genera el código correspondiente al valor indicado, que puede ser un número o un símbolo ya definido.
DB 'c'	Genera el código correspondiente al ascii del carácter indicado entre comillas simples.
DB "cad"	Genera los códigos correspondientes a la cadena indicada (máximo tres caracteres, el resto se ignora).
END	Marca el final del archivo (no es necesaria).

(1) En el caso de la directiva EQU se permite hacer referencia a una etiqueta incluida en una línea de instrucción posterior (forward reference).

La sintaxis empleada en las líneas de instrucción es similar a la utilizada por otros ensambladores comerciales. Ejemplos:

etiqueta:

<etiqueta:> nemónico lista_de_parámetros ;comentario opcional

<etiqueta:> nemónico ;comentario opcional

<etiqueta:> ;comentario opcional

;comentario sin más

Se deben utilizar los nemónicos definidos por Intel para la familia 8051, así como la sintaxis de parámetros (operandos) correspondiente. Se permite sustituir operandos numéricos por expresiones a evaluar numéricamente. La presente versión (2.4) de UVIENS51 realiza evaluación de expresiones en la fase de ensamblado. Todas las expresiones evalúan a enteros en el margen 0000h a 0FFFFh. Se evalúan de izquierda a derecha sin considerar prioridades de operadores. Para priorizar es necesario emplear paréntesis. Se permiten las siguientes operaciones:

+	suma
-	resta y operador unario de cambio de signo
*	producto sin signo
/	división entera sin signo
%	resto de división entera sin signo
&	y (<i>and</i> , producto lógico) sobre los 16 bits
	o (<i>or</i> , suma lógica) sobre los 16 bits
~	negación (complemento a 1) sobre los 16 bits
^	o exclusiva (<i>exclusive or</i>) sobre los 16 bits
<	desplazamiento a la izquierda (1 bit)
>	desplazamiento a la derecha (1 bit)
[	byte de mayor peso del operando (equivale a HIGH de Intel)
]	byte de menor peso del operando (equivale a LOW de Intel)

El evaluador representa los números negativos en complemento a 2. El tipo del resultado de una operación es:

- 👉 El mismo que el del operando en las unarias.
- 👉 En la suma y la resta, si un operando es número genérico y el otro no, el resultado es del tipo del que no es número genérico.
- 👉 Número genérico (como en una definición mediante EQU) en todos los demás casos.

En las instrucciones de salto se debe indicar la dirección absoluta de destino, bien de forma numérica, bien mediante un símbolo (etiqueta o *label*). No se deben emplear valores que indiquen un desplazamiento relativo, ya que el ensamblador realiza este cálculo de forma automática para las instrucciones SJMP, CJNE, etc.

Se permite el uso del símbolo \$ para hacer referencia al valor actual del contador de programa. Por ejemplo, SJMP \$ representa un bucle que se cierra sobre la misma instrucción.

Se permite el uso de las instrucciones genéricas JMP dirección y CALL dirección. El ensamblador elige la instrucción de salto (SJMP, AJMP, LJMP) o llamada a subrutina (ACALL, LCALL) óptima en las referencias hacia atrás (*backward*). En las referencias hacia adelante (*forward*) se elige siempre LJMP o LCALL.

El archivo fuente se traduce a mayúsculas antes de ser analizado (**a excepción de los comentarios y los caracteres entre comillas dobles y simples**). Por lo tanto, la reproducción en el archivo .LST aparece con esta traducción.

5.4. Limitaciones

A continuación se enumeran las principales **limitaciones de la presente versión del ensamblador**:

- 👉 Número máximo de símbolos permitidos= 1000 (incluyendo A, R0 a R7, \$ y los definidos mediante el archivo UVI51.REG.
- 👉 Número máximo de líneas del programa fuente= 3000 (incluyendo todas las líneas aunque sólo sean comentarios o estén vacías).
- 👉 La longitud total de las líneas se corta a 80 caracteres.
- 👉 La longitud de etiquetas u otros símbolos se corta a 31 caracteres.
- 👉 El bloque de parámetros de una instrucción se corta a 31 caracteres.

6. Simulador Uvisim

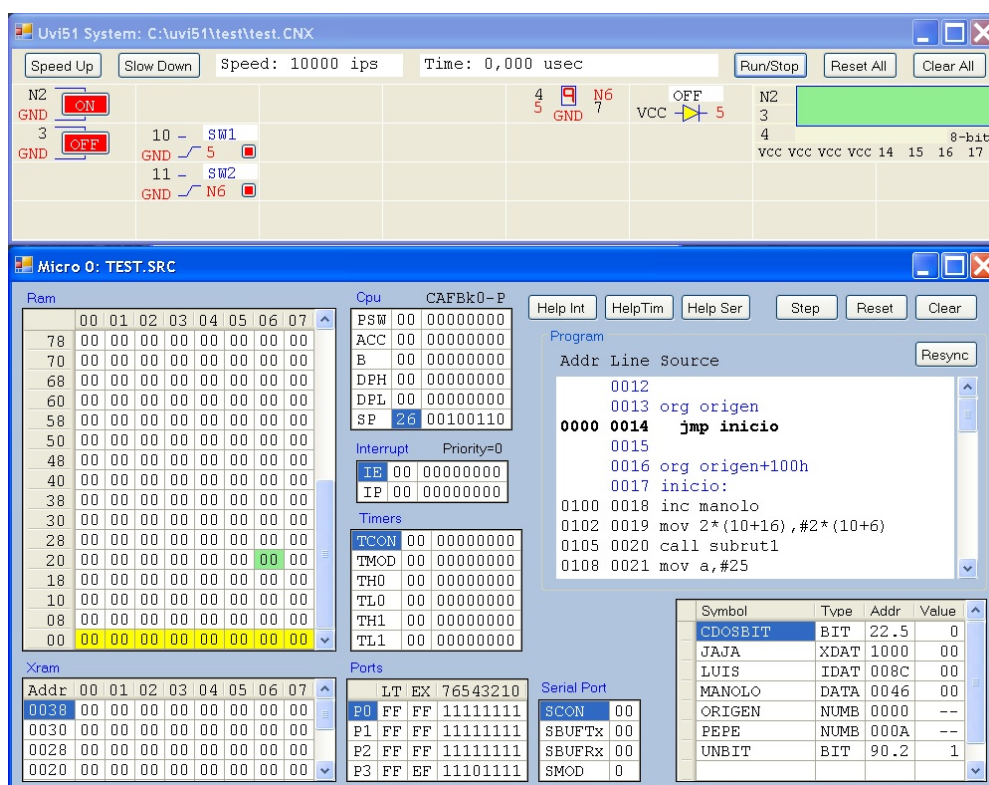
Este apartado describe las características y el manejo del simulador de UVI-51-win.

6.1. Pantalla principal y manejo básico del simulador

El simulador de esta versión ha sido desarrollado para operar bajo Windows Framework 2.0, y posee las siguientes características:

- 👍 Posibilidad de simulación de sistemas multimicro.
- 👍 Interfaz gráfica de usuario.
- 👍 Posibilidad de avance paso a paso parando (*trace into*) o no (*step over*) dentro de las subrutinas.
- 👍 Posibilidad de definir puntos de ruptura (breakpoints).

Al iniciar el simulador, se visualizan dos ventanas con el aspecto que se muestra a continuación.



La ventana superior, con fondo de color arena, se denomina **ventana del sistema**. Esta ventana incluye una cabecera con el nombre del sistema que se simula, un conjunto de botones relacionados con la simulación del sistema y un área dividida en 32 casillas donde se presentan los periféricos externos (ver apéndice A1). La inferior, con fondo de color azul, se denomina **ventana del microcontrolador** e incluye una cabecera con el nombre del micro (p. ej. micro 0) y su programa (p. ej. TEST.SRC), así como un conjunto de bloques que contienen información sobre el micro seleccionado.

Si el sistema define más de un microcontrolador, aparece una ventana para cada uno de ellos. Estas ventanas se muestran inicialmente en cascada.

El simulador se invoca pulsando el botón “Simulate” del asistente. Para salir del simulador, basta con pulsar el botón de cierre de la ventana del sistema. Las ventanas de los microcontroladores no se pueden cerrar, pero sí es posible minimizarlas. Al cerrar la ventana del sistema se cierran automáticamente todas las demás ventanas del simulador.

6.2. Ventana del sistema

La función de los botones de la ventana del sistema es la siguiente:

Speed Up y Slow Down.- Permiten modificar la velocidad de simulación.

Run/Stop.- Se utiliza para activar y desactivar la simulación de forma continua.

Reset All.- Realiza un reset de todos los micros que forman el sistema.

Clear All.- Similar al reset pero incluye la puesta a cero de la RAM de todos los micros, y suprime los puntos de ruptura (breakpoints).

La ventana de sistema incluye también un indicador de la velocidad de simulación expresada en instrucciones por segundo, y un contador de tiempo del sistema expresado en microsegundos (con resolución de 1 nanosegundo).

El área de periféricos externos muestra el estado de los periféricos definidos en el archivo CNX (ver apéndice A1). Cada tipo de periférico se muestra de una forma diferente para facilitar su identificación:

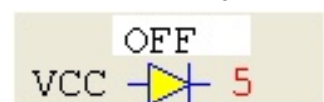
👉 Conmutador controlado manualmente. Se muestra el nombre del conmutador (asignado en el archivo CNX) y su esquema eléctrico con indicación de los números de los nodos a los que está conectado. La posición del contacto móvil se adapta al estado del conmutador. Para cambiar el estado se pulsa el botón pequeño situado a su derecha. **Los conmutadores controlados por el nivel de un nodo, no se presentan en la ventana de periféricos.**



👉 Pulsador. Se muestra un botón (pulsador) con el nombre asignado grabado. Se indican los números de los nodos a los que está conectado. El contacto se supone cerrado cuando se mantiene pulsado con el ratón. Por esta razón, no es posible mantener cerrados varios pulsadores de forma simultánea. Si ésto representa un problema en alguna aplicación, se pueden utilizar conmutadores en lugar de pulsadores.



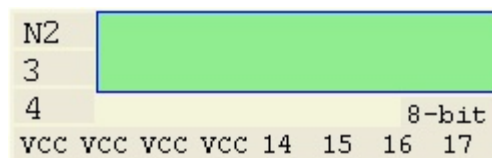
👉 Visualizador binario (diodo LED). Se representa mediante el símbolo de un diodo y se indican los números de los nodos a los que está conectado. También se muestra su nombre. Cuando está activo, el interior del símbolo se muestra coloreado, cuando no permanece gris. El color es rojo por defecto, pero puede definirse en el archivo CNX.



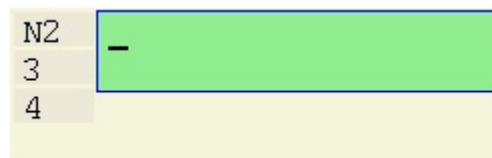
👉 Visualizador hexadecimal (7 segmentos) con capacidad de multiplexado. Los segmentos activos se muestran en color rojo. Se indican los números de los nodos a los que está conectado. Si el nodo de habilitación está a nivel '1' no se muestra ningún segmento (es activo a nivel bajo).



👉 Pantallas LCD. Se pueden utilizar dos tipos diferentes de pantallas LCD. La figura de la derecha muestra una pantalla LCD estándar.



La pantalla LCD de acceso serie se muestra a la derecha. Esta pantalla no se corresponde con ningún producto comercial. Su modo de funcionamiento está basado en un registro de desplazamiento.



La posición en que aparecen en la pantalla y la forma en que están conectados se define en el archivo CNX.

6.3. Ventana del microcontrolador

La ventana del microcontrolador contiene 7 botones, un listado de programa y 8 tablas de datos. Los botones tienen las siguientes funciones:

Step.- Ejecuta una instrucción del micro (paso) y las que correspondan de los demás micros para mantener el sincronismo. Si la instrucción a ejecutar es un CALL y se mantiene pulsada la tecla Shift del teclado cuando se pulsa el botón Step, entonces se ejecuta todo el código de la subrutina (como si fuera modo continuo) y se detiene en la instrucción siguiente al CALL (modo step-over).

Reset.- Realiza un reset del micro

Clear.- Realiza un reset del micro, pone a cero su memoria RAM y borra sus puntos de ruptura (breakpoints).

Resync.- Sincroniza el listado del programa del microcontrolador con la ejecución (ver “listado del programa” más abajo).

Help Int.- Abre la ventana de ayuda de Interrupciones.

Help Tim.- Abre la ventana de ayuda de Temporizadores.

Help Ser.- Abre la ventana de ayuda del Puerto Serie.

Las tres ventanas de ayuda pueden estar abiertas de forma simultánea, pero no es posible abrir más de una ventana de cada tipo. Estas ventanas se pueden minimizar y cerrar.

En la parte superior derecha de la ventana del microcontrolador se encuentra el listado del programa. Por defecto, la próxima instrucción a ser ejecutada (prox_inst) se muestra en la tercera línea, y resaltada en negrilla. La barra de desplazamiento vertical permite inspeccionar el programa completo. Al realizar un desplazamiento, es probable que la prox_inst desaparezca del listado. Para que la prox_inst vuelva a ocupar la tercera línea del listado basta con pulsar el botón Resync.

Las líneas de programa que generan código se muestran en color negro, mientras que las que no generan código (comentarios, equ, etc.) se muestran en azul.

Se puede marcar un punto de ruptura (breakpoint) haciendo click sobre la instrucción deseada. La línea marcada se muestra entonces resaltada sobre fondo

amarillo. Para quitar un breakpoint se hace click nuevamente sobre la línea correspondiente. Se pueden marcar tantos breakpoints como sea necesario. Para eliminar todos los breakpoints se recomienda utilizar el botón Clear, aunque ésto también provoca el reset del micro y el borrado de la RAM.

Se puede saltar desde la situación actual del contador de programa a cualquier otra instrucción. Para ello basta con hacer click sobre la instrucción deseada manteniendo pulsada la tecla Shift.

ATENCION: En la versión MSDOS la forma de marcar un breakpoint y forzar un salto eran exactamente al revés (el shift se usaba para el breakpoint). Además, en la versión MSDOS, estas operaciones se realizaban en una ventana especial “de programa fuente” y no en el listado de programa. En la versión win, no existe tal ventana especial.

A continuación se describen las 8 tablas de datos de la ventana de microcontrolador.

En la parte izquierda de la ventana se encuentra la tabla de la memoria RAM interna. La barra de desplazamiento permite visualizar la memoria RAM desde la dirección 00h hasta la FFh. La celdilla de la RAM a la que apunta el puntero de pila (SP) se muestra en color verde. El bloque de registros seleccionado se muestra en color amarillo. Si ambas propiedades coinciden, prevalece el color verde. Los contenidos de la memoria RAM se pueden editar haciendo click sobre la celdilla deseada y tecleando el valor (emplear dos cifras hexadecimales).

ATENCION: En la versión MSDOS se podían visualizar los símbolos en la misma ventana RAM. En la versión win existe una tabla de datos específica para los símbolos (ver más abajo).

Addr	Line	Source
	0012	
	0013	org origen
0000	0014	jmp inicio
	0015	
	0016	org origen+100h
	0017	inicio:
0100	0018	inc manolo
0102	0019	mov 2*(10+16),#2*(10+6)
0105	0020	call subrut1
0108	0021	mov a,#25

	00	01	02	03	04	05	06	07
78	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00
68	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00
58	00	00	00	00	00	00	00	00
50	00	00	00	00	00	00	00	00
48	00	00	00	00	00	00	00	00
40	00	00	00	00	00	00	00	00
38	00	00	00	00	00	00	00	00
30	00	00	00	00	00	00	00	00
28	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00
18	00	00	00	00	00	00	00	00
10	00	00	00	00	00	00	00	00
08	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00

Debajo de la tabla RAM se encuentra la tabla XRAM. Usando la barra de desplazamiento se puede acceder a toda la memoria XRAM asociada al microcontrolador (definida el archivo CNX). Los contenidos se editan de la misma forma que en la tabla RAM.

Xram									
Addr	00	01	02	03	04	05	06	07	
01F8	00	00	00	00	00	00	00	00	
01F0	00	00	00	00	00	00	00	00	
01E8	00	00	00	00	00	00	00	00	
01E0	00	00	00	00	00	00	00	00	

Cuando el programa del micro lee o escribe en la ram externa, realiza un acceso a la celdilla de memoria cuya dirección es “addr MODULO xramSize”, siendo:

addr .- dirección de xram indicada en la instrucción.

MODULO .- operador modulo.

XramSize .- tamaño de la memoria ram externa (definido en el archivo CNX).

De esta forma se obtiene un comportamiento equivalente al de un enlace real entre micro y memoria que solo conecta las líneas de dirección necesarias para direccionar la capacidad de la memoria utilizada.

La tabla Cpu muestra el contenido de los registros básicos del microcontrolador, tanto en hexadecimal como en binario. La línea CAFBk0-P facilita la identificación de los bits de la palabra de status (PSW).

ATENCION: En la versión MSDOS se visualizaban otros registros y valores que no están presentes en la versión win. En esta versión se puede observar el valor del contador de programa (PC) en el listado de programa.

Cpu			CAFBk0-P
PSW	00	00000000	
ACC	00	00000000	
B	00	00000000	
DPH	00	00000000	
DPL	00	00000000	
SP	07	00000111	

La tabla Interrupt permite visualizar el contenido de los registros de habilitación (IE) y de prioridad (IP) del gestor de interrupciones.

Interrupt		Priority=0
IE	00	00000000
IP	00	00000000

La tabla Timers permite visualizar el contenido de los registros TCON, TMOD, TH0-TL0 y TH1-TL1.

Timers		
TCON	00	00000000
TMOD	00	00000000
TH0	00	00000000
TL0	00	00000000
TH1	00	00000000
TL1	00	00000000

La tabla Ports permite visualizar el contenido de los puertos paralelo de entrada y salida. La columna LT muestra el valor del cerrojo (latch), mientras que la columna EX muestra el estado de los terminales (pins) de cada puerto. Además, la columna de la derecha muestra el estado de los terminales en formato binario. La línea “76543210” facilita la identificación de los terminales. Se recuerda que en el 8051 se emplean puertos cuasi-bidireccionales, por lo que es imprescindible distinguir entre “latch” y “pins”. El simulador no permite editar el valor (estado) de los terminales.

Ports				76543210
	LT	EX		
P0	FF	FF		11111111
P1	FF	FF		11111111
P2	FF	FF		11111111
P3	FF	EF		11101111

ATENCION: En la versión MSDOS se permitía editar el estado de los terminales, salvo en el caso de los terminales que estuvieran conectados a otros terminales o a apariféricos externos. Para editar terminales en la versión win, es necesario definir pulsadores o conmutadores en el archivo CNX.

La tabla del puerto serie muestra el estado de los registros SCON y SBUF, así como el estado del bit SMOD de PCON. Aunque en el 8051 sólo se define un registro SBUF, lo cierto es que existen dos registros (aquí denominados SBUFTx y SBUFRx). Las instrucciones que leen el SBUF, leen SBUFRx. Las instrucciones que escriben en SBUF generan la transmisión por el puerto serie del valor escrito. De esta forma, el simulador muestra los últimos valores enviados y recibidos por el puerto serie empleando los registros SBUFTx y SBUFRx respectivamente.

Serial Port	
SCON	00
SBUFTx	00
SBUFRx	00
SMOD	0

La tabla de símbolos se muestra en la parte inferior derecha de la ventana. La barra de desplazamiento permite visualizar cualquier símbolo. La primera columna indica el nombre del símbolo. La segunda indica el tipo (Type) y la tercera indica la dirección (Addr). Si el tipo es “número genérico” (NUMB), entonces el campo Addr muestra su valor. Para los demás tipos, el valor se muestra en la columna de valor (Value). Solamente se permite editar la columna Value.

Symbol	Type	Addr	Value
CDSBIT	BIT	22.5	0
JAJA	XDAT	1000	00
LUIS	IDAT	008C	00
MANOLO	DATA	0046	00
ORIGEN	NUMB	0000	--
PEPE	NUMB	000A	--
UNBIT	BIT	90.2	1

Cuando el simulador se encuentra en modo de ejecución continuo, se inhiben los botones “Step”, “Reset” y “Clear” de las ventanas de microcontroladores. En esta situación la simulación se detiene en los siguientes casos:

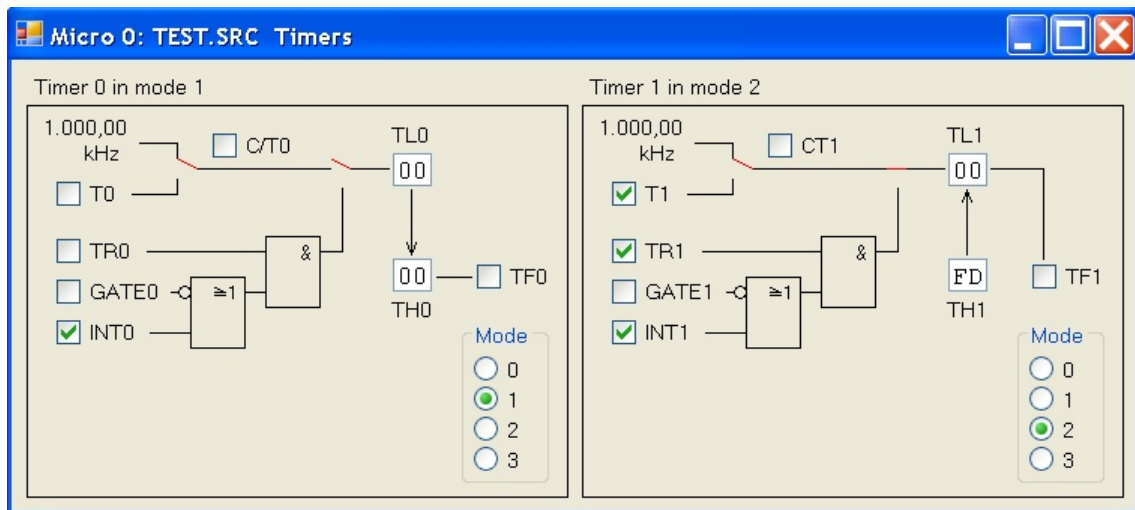
- Si el usuario pulsa el botón “Run/Stop”.
- Si el contador de programa apunta al código de operación A5 (reservado).
- Si el contador de programa apunta a una posición de ROM cuyo contenido no ha sido definido por el programa. El comportamiento del microcontrolador real, en estas condiciones, no está definido.
- Si se alcanza una instrucción marcada con un punto de ruptura.

ATENCION: En la versión MSDOS se disponía de un mecanismo de parada por valor, que detenía la ejecución cuando una determinada dirección de la ram interna obtenía un determinado valor. La version win actual carece de esta opción. Se está estudiando la posibilidad de crear un mecanismo más flexible, como por ejemplo el de forzar la detención de la ejecución cuando se satisfaga una condición compleja expresada de forma matemática, por ejemplo:

```
If ( (ram[20h]==32h) && (xram[9000h]<0efh) )
```

6.4. Ventana informativa de temporizadores

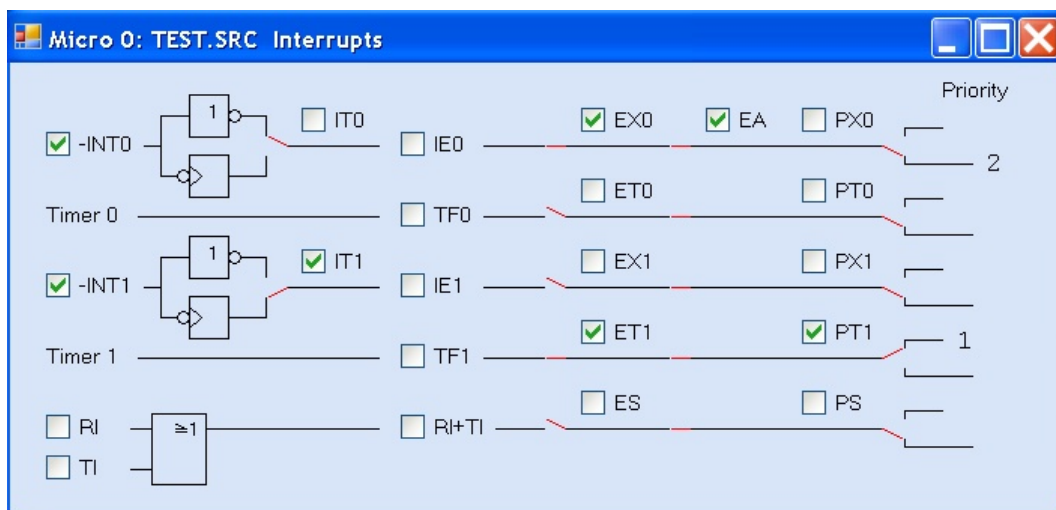
Presenta un diagrama funcional del estado de los temporizadores con indicación del modo de operación. Los bits de TMOD y TCON que intervienen en el funcionamiento de los temporizadores se pueden editar. No se pueden editar los bits que son terminales de puertos. Además, es posible definir el modo de cada temporizador empleando los botones del “grupo Mode”. Los cambios de configuración repercuten inmediatamente en los esquemas. Esta ventana se actualiza automáticamente cuando el simulador trabaja en modo continuo.



ATENCIÓN: En la versión MSDOS se permitía editar algunos terminales de puertos en este tipo de ventana. En la versión win no se permite editar ningún terminal de puertos. Además, la versión win incluye el grupo de “botones de radio” Mode para facilitar el cambio de modo de cada temporizador.

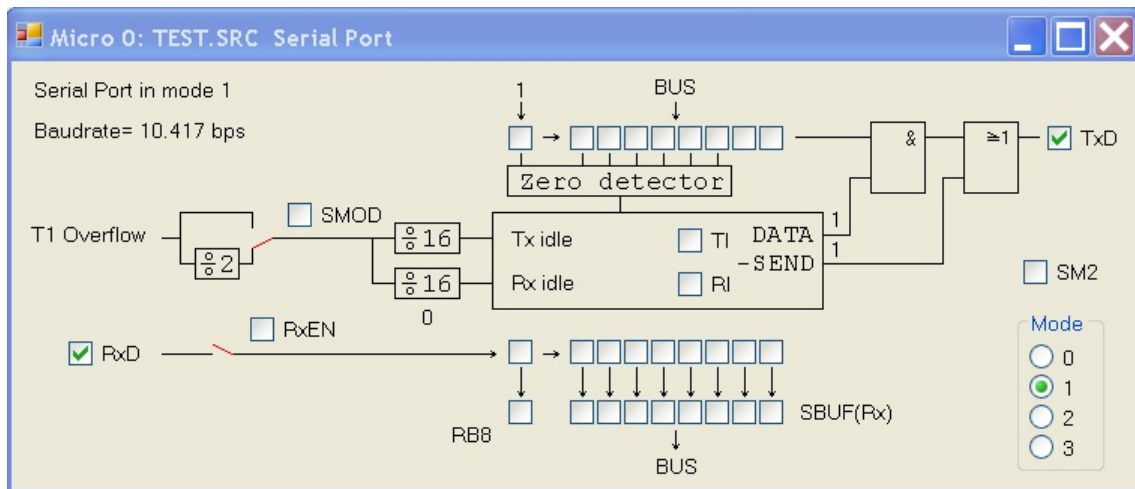
6.5. Ventana informativa de interrupciones.

Presenta un diagrama explicativo del estado en que se encuentran los registros de habilitación y prioridades, así como de las peticiones de interrupción presentes y el orden en que se prevé atenderlas. Se permite la edición de los registros y bits asociados a las interrupciones (IE, IP, etc.). No se permite la edición de terminales de puertos. Esta ventana se actualiza automáticamente cuando el simulador trabaja en modo continuo.



6.6. Ventana informativa del puerto serie.

Para los modos 1 y 3 (los que simula esta versión) indica el modo y la velocidad si está definida. Además se muestra el estado en que se encuentran la transmisión y la recepción.



En la parte transmisora se indica el bit que se está transmitiendo.

En la parte receptora se indica el último bit que ha sido recibido (muestreado). Además, se indica el estado del contador divisor por 16 empleado para el muestreo (de 0 a F). La línea RxD se muestrea en los estados 7, 8 y 9, de la misma forma que en un 8051 real.

También se indica el estado de los registros SBUF(tx) y SBUF(rx) y del registro de desplazamiento de recepción. Se recuerda que en el 8051, el propio SBUF(tx) actúa como registro de desplazamiento de transmisión. SBUF(rx) se carga con los 8 bits de menor peso del registro de desplazamiento al final de la recepción de una trama (como en el microcontrolador real).

Esta ventana se actualiza automáticamente cuando el simulador trabaja en modo continuo.

ATENCIÓN: En la versión MSDOS se indicaba también si la trama era de 10 u 11 bits. En la versión win se ha añadido un grupo de botones de radio (Mode) para facilitar el cambio de modo.

6.7. Edición de valores

ATENCIÓN: En la versión MSDOS, en modo de ejecución continua sólo se podían modificar los terminales de los puertos. En la versión win, no se pueden editar nunca los terminales de los puertos de forma directa, sino que es necesario emplear pulsadores o conmutadores. Sin embargo, la versión win permite editar cualquiera de las otras tablas de datos (ram, xram, ect.) incluso en modo continuo. Esta posibilidad debe emplearse con prudencia.

La edición se realiza de la forma habitual en las aplicaciones windows. Se hace click con el ratón sobre la celdilla a editar, se teclea el nuevo valor, y se usa Intro para aceptar la modificación. En algunos casos puede ser necesario borrar previamente el contenido actual antes de escribir el nuevo. Todos los valores deben introducirse en notación hexadecimal o binaria, según el caso. Si se intenta introducir un valor no válido, aparece un mensaje advirtiendo del error.

6.8. Configuración simulada del 8051

Por defecto, se simula un 8051 con un cristal de 12 MHz (ciclo de máquina de 1 μ seg), con 64 Kbytes de memoria de programa interna (rom) y una memoria ram externa de 8 Kbytes (xram). Esta memoria se considera cíclica como si estuviera conectada sin obedecer a las líneas de dirección A13 a A15. Por lo tanto, cualquier acceso a la xram (al ejecutar el programa) en una dirección (ADDR) superior a la 8191 (hexadecimal 1FFF) tiene lugar en la dirección ADDR & 8191, equivalente a ADDR MODULO 8192. Esta forma de operación se aplica de forma similar a cualquier otra capacidad de memoria que se defina. La capacidad de la memoria xram será siempre una potencia de 2 (máximo 64 Kbytes).

Se debe tener en cuenta que en una aplicación con el microcontrolador real, la conexión de ram externa utiliza los puertos 0 y 2 como buses. En el simulador, este hecho no se refleja y, a diferencia del caso real, los ports 0 y 2 siguen funcionando normalmente.

Sin embargo, el simulador tiene en cuenta el valor del puerto 2 en la ejecución de las instrucciones: `movx a,@r0` `movx a,@r1` `movx @r0,a` `movx @r1,a`.

Otro aspecto a tener en cuenta es la inclusión en el simulador de los 128 bytes altos de ram interna. Esta ram no existe en el 8051, pero sí en el 8052 y otros derivados. El simulador no considera, sin embargo, otros aspectos del 8052 como la existencia de un tercer temporizador.

La versión actual simula el funcionamiento del puerto serie solamente en los modos 1 y 3. Esta simulación se realiza a nivel de señales lógicas en los terminales P3.0 y P3.1 de tal forma que se puede simular la comunicación serie entre micros de un sistema multimicro. Esto incluye también la simulación de comunicación serie realizada mediante programa (*UART software*). La simulación incluye el mecanismo de *muestreo triple* de bits en la parte receptora (consultar la documentación del 8051).

7. Visualizador del analizador lógico

ADVERTENCIA: Este apartado (7) hace referencia al analizador lógico de la versión DOS. En su lugar, para la versión windows debe utilizarse el documento “Manual Uvilogic”.

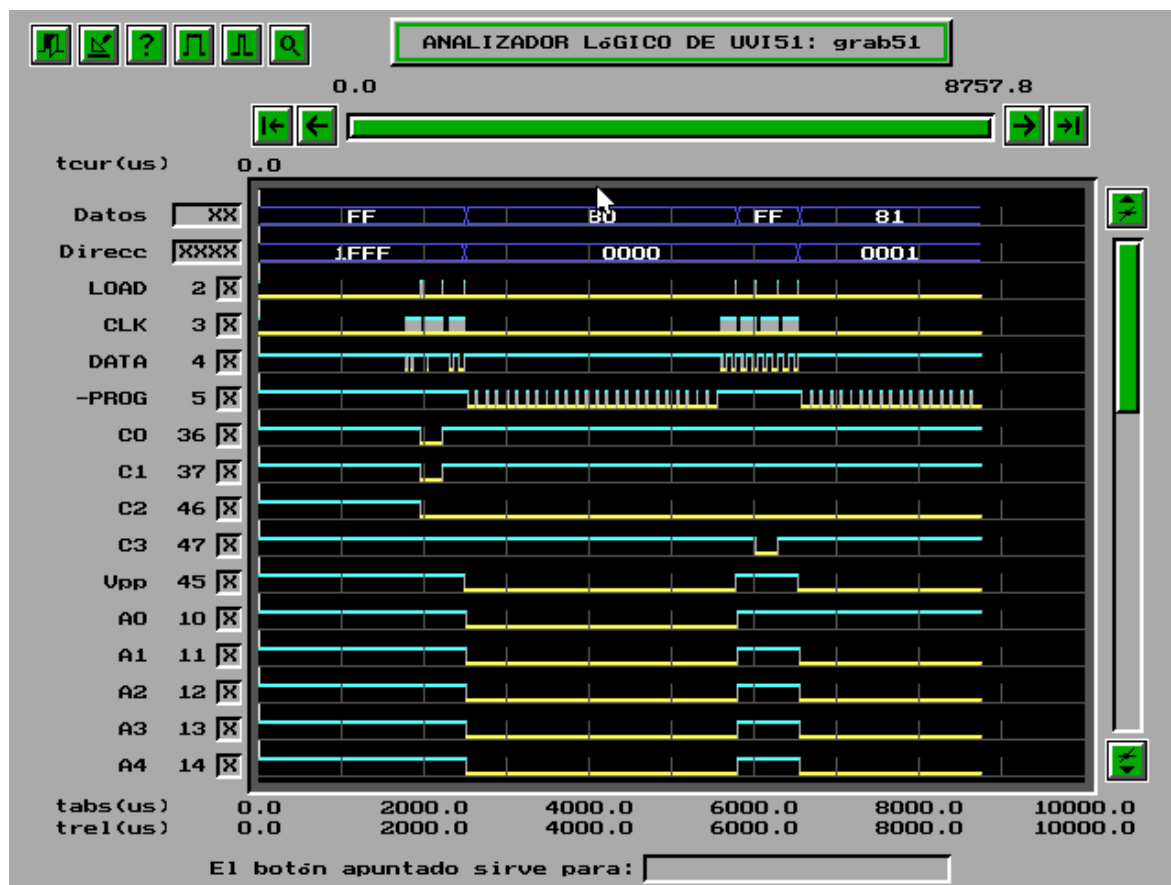
7.1 Introducción.

Desde la versión 4.0 se incluye en UVI51 la posibilidad de registrar, al tiempo que se realiza la simulación, la evolución temporal de los niveles lógicos de todos los nodos del sistema. Para ello se debe definir un periférico del tipo *analizador* en el archivo de configuración (ver apéndice A1, apartado A1.2).

Una vez realizada la simulación, los datos registrados se pueden visualizar con un formato típico de analizador lógico mediante el programa GRAF51.EXE. Para ello basta con pulsar el botón [ANALIZA] del entorno. Esta opción no está disponible en el modo monomicro. En este apartado se describe el manejo del programa GRAF51, que se denominará en adelante programa analizador.

7.2 Descripción de la pantalla del analizador.

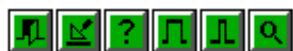
El analizador es una aplicación DOS que emplea un modo gráfico. Se puede controlar mediante el teclado o el ratón. Algunas de las operaciones requieren explícitamente el uso del ratón.



La figura presenta el contenido de la pantalla del analizador. Se compone de los siguientes elementos:

👉 Título. Situado en la parte superior de la pantalla, muestra el texto "ANALIZADOR LÓGICO DE UVI51" seguido de dos puntos y el nombre del archivo LOG visualizado.

👉 Barra de botones superior



Incluye 6 botones cuya función se indica a continuación.



Se pulsa para salir del programa. Equivale a la tecla Esc.



Genera un archivo en formato HPGL con el contenido de la ventana de cronogramas. Equivale a la tecla 'p'.



Presenta una ventana de ayuda sobre el manejo del analizador mediante el teclado. Equivale a la tecla '?'.



Amplia la escala de tiempos (zoom in). Equivale a la tecla '+'.



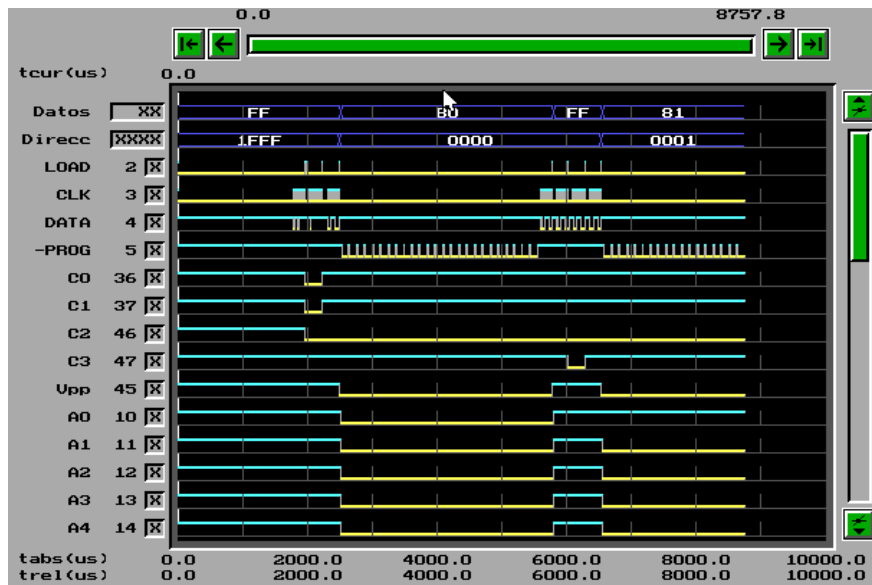
Reduce la escala de tiempos (zoom out). Equivale a la tecla '- '.



Realiza la búsqueda del patrón desde el cursor hasta el final del trozo de cronograma visualizado. Si se pulsa manteniendo pulsada la tecla de mayúsculas (shift), la búsqueda se extiende más allá del trozo visualizado. Equivale a las teclas 'f' y 'F' respectivamente. Ver apartado 7.3.



Ventana de cronogramas:



En esta ventana se realiza la presentación de los cronogramas de los nodos. Posee 2 barras de desplazamiento. La horizontal permite recorrer los cronogramas a lo largo del eje de tiempos. La vertical permite seleccionar el bloque de nodos que se visualizan. Las barras se pueden desplazar pulsando los botones siguientes.



Avanza en la escala de tiempos. Equivale a la tecla "flecha derecha".



Retrocede en la escala de tiempos. Equivale a la tecla "flecha izquierda".



Avanza al final del cronograma (tiempo máximo). Equivale a pulsar la tecla "Fin".



Retrocede al principio del cronograma (tiempo mínimo). Equivale a pulsar la tecla "Inicio".



Visualiza nodos anteriores (con numeración más baja). Equivale a pulsar la tecla "RePág".



Visualiza nodos posteriores (con numeración más alta). Equivale a pulsar la tecla "AvPág".

También se puede arrastrar la barra con el ratón o pulsar sobre el surco para provocar el salto a la posición indicada.

A la izquierda aparecen unas etiquetas que indican el nombre y número asociado a cada nodo representado. Asimismo se representan los valores de los patrones de búsqueda que para los nodos pueden ser X= indiferente, 0= nivel bajo ó 1= nivel alto. En el caso de los buses, el patrón

se indica en formato hexadecimal, con el menor número posible de cifras que permitan incluir todos los nodos que lo forman (X= indiferente, ?= indefinido, otro= valor hexadecimal). En el apartado 7.3 se explica la forma de establecer y utilizar patrones de búsqueda.

En la parte inferior de la ventana se presentan dos escalas de tiempos expresados en microsegundos, una absoluta que indica el tiempo del micro durante la simulación y otra relativa que siempre comienza en cero.

Además, se incluye un cursor de tiempo en forma de barra vertical a trazos de color blanco.. Su valor temporal se muestra en la parte superior de la ventana, y se desplaza con el mismo. Para desplazarlo se pueden utilizar las combinaciones de teclas o botones siguientes.

Mayúsculas +  o Mayúsculas + tecla "flecha derecha" para desplazar el cursor a la derecha.

Mayúsculas +  o Mayúsculas + tecla "flecha izquierda" para desplazar el cursor a la izquierda.

Mayúsculas +  o Mayúsculas + tecla "Fin" para desplazar el cursor al final de la ventana.

Mayúsculas +  o Mayúsculas + tecla "Inicio" para desplazar el cursor al principio de la ventana.

Para mover el cursor a una posición determinada basta con pulsar con el ratón en la posición deseada del cronograma. También se puede arrastrar el cursor con el ratón.

El valor temporal del cursor indica, por defecto, el tiempo absoluto que corresponde a su posición. Si se mantiene pulsada la tecla "Mayúsculas" al tiempo que se pulsa con el ratón sobre la posición del cursor, el valor temporal se pone a cero. Esto facilita la medida de tiempos. Para volver a la indicación normal se debe realizar la misma operación pero sustituyendo la tecla "Mayúsculas" por la tecla "Control".

Algunas operaciones, como la presentación de cronogramas muy largos y con muchas conmutaciones, pueden precisar bastante tiempo. Para cancelar una operación de este tipo basta con pulsar la tecla "Supr".

En la parte inferior de la pantalla se ofrece una ayuda sobre la función que realiza cada botón. Para utilizar esta ayuda basta con pasar el ratón sobre el botón cuya función se desea conocer.

7.3 Búsqueda de patrones en el cronograma.

El analizador permite localizar patrones en los cronogramas. Un patrón es un conjunto de valores (vector) asociado a un conjunto de nodos. Antes de iniciar el proceso de búsqueda, es necesario definir el patrón que se va a utilizar.

Para definir un patrón, basta con asignar valores 0 ó 1 a los nodos que corresponda. Si el valor de un nodo resulta indiferente se le asignará el *valor* X. Cada vez que se pulse con el ratón sobre el patrón de un nodo, el valor cambiará según la secuencia cíclica X, 0 , 1, X, etc. Inicialmente, Los patrones de todos los nodos se encuentran inicialmente asignados como indiferentes. Cuando se modifica el patrón de un nodo, se modifica de forma automática el patrón del bus (o buses) que contienen ese nodo.

También es posible modificar directamente el patrón de un bus. La modificación se realiza sobre cada cifra hexadecimal por separado. Para ello se debe pulsar con el ratón sobre la cifra. La cifra se borra y aparece en su lugar un cuadrado de color rojo. A continuación se debe pulsar la tecla con el valor hexadecimal que se quiere asignar a esa cifra (de 0 a F). También se puede pulsar la tecla 'X' para indicar que los patrones de los 4 nodos que forman esa cifra deben asignarse como indiferentes. Cuando se modifica el patrón de un bus, se modifican de forma automática los patrones de los nodos que lo componen.

El símbolo "?" en una cifra del patrón de un bus indica que alguno de los nodos que la forman, pero no todos, tiene asignado su patrón como indiferente.

Una vez definido el patrón, se puede realizar su búsqueda en los cronogramas. La búsqueda comienza en la posición actual del cursor, y se extiende al intervalo mostrado en la pantalla o hasta el final del cronograma, según se invoque.

Para realizar una búsqueda limitada a la pantalla, se debe pulsar el botón  o la tecla 'f'. Si la tecla "Mayúsculas" (*shift*) está también pulsada, la búsqueda se extiende hasta el final del cronograma. Cuando el analizador encuentra una coincidencia, traslada el cursor al instante en que se produce. De esta forma se simplifica la realización de búsquedas sucesivas del mismo patrón.

APENDICE A1: Sintaxis de los archivos CNX

Los archivos CNX son archivos ASCII formados por dos bloques. En el primero se definen los micros utilizados y en el segundo los periféricos externos. Cuando en este apéndice se haga mención a un *número de nodo* se utiliza el término en un sentido equivalente al empleado en SPICE, con las peculiaridades siguientes:

- 👍 Debe ser un número comprendido entre 0 y 199. **Para optimizar la velocidad de simulación, se recomienda utilizar números correlativos a partir del 2.**
- 👍 Los números de nodo cero y uno se reservan para indicar:
 - 👍 nodo 0 → conexión a masa (nivel lógico 0).
 - 👍 nodo 1 → conexión a +5V a través de una resistencia de pull-up. Puede interpretarse como un nivel lógico 1 susceptible de pasar a ser 0 mediante AND por conexión (*Wired-AND* o cableado lógico).

A1.1. Definición de microcontroladores

Para cada uC utilizado se escribe una línea con el formato:

Uprograma frecreloj tamañoxram

en donde la U inicial debe figurar como primer carácter de la línea y debe ir seguida sin espacios del nombre del programa que se ejecuta en ese micro. Después, separados por espacios, se indican la frecuencia del oscilador de reloj (cristal) **en MHz**, y el tamaño de la memoria ram externa en bytes. El tamaño de la xram se redondea a la potencia de 2 mayor o igual a la indicada. Los límites para los parámetros son:

1 Hz <= frecreloj <= 1 GHz

64 <= tamañoxram <= 65536 bytes

El tamaño de la memoria de programa (rom) es fijo e igual a 64 kbytes.

NOTA: En la versión MSDOS, el tamaño de la memoria xram se limitaba a 32768 bytes y el tamaño de la memoria rom se podía definir hasta un máximo de 8192 bytes. Por compatibilidad, la version win permite la existencia del parámetro tamañorom (utilizado por la versión MSDOS) después del parámetro tamañoxram, pero la ignora. La versión MSDOS limitaba la frecuencia de reloj al margen 1MHz – 100 MHz.

La línea de definición puede ir seguida de la asignación de números de nodo a los terminales de los puertos paralelo, con el formato:

Pxy=N

que significa bit 'y' del puerto 'x' conectado al nodo N. **Se recomienda utilizar números correlativos a partir del 2**, que en ningún caso deben superar el 199. Todas las líneas del tipo Pxy=n se suponen referidas al último micro definido.

A1.2. Definición de periféricos

Se pueden utilizar los siguientes periféricos, mediante la sintaxis que se indica.

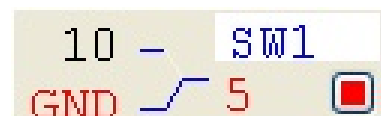
Conmutador

C<nombre> <nodo_comun> <nodo_si0> <nodo_si1> <nodo_control>

Si nodo_control es negativo, el estado del conmutador (posición del contacto) se controla mediante el nivel de ese nodo. Si nodo_control es positivo, el estado lo controla el usuario haciendo click sobre el pequeño pulsador rojo que forma parte de la celdilla del conmutador. En este caso, el conmutador se muestra en la celdilla cuyo número es igual al valor asignado a <nodo control>. Ejemplos:

CSW1 10 0 5 1

define un conmutador con el contacto móvil conectado al nodo 5, el contacto de reposo conectado al nodo 10 y el activo conectado al nodo 0 (masa). El estado del conmutador (reposo o activado) se controla pulsando con el ratón en el botón que aparece en la pantalla del simulador (en la celdilla 1, identificada por el nombre "SW1").



Ca 10 0 5 -2

define un conmutador similar al anterior salvo que en este caso no aparecerá dibujado en la pantalla del simulador, porque su estado se controla de forma automática mediante el nivel lógico presente en el nodo 2. Este conmutador actúa de forma similar a un relé.

Pulsador (tecla)

T<nombre> <nodo_A> <nodo_B> <posición>



El pulsador cierra el contacto entre los nodos indicados cuando se pulse el botón marcado con <nombre> en la pantalla del simulador. Ejemplo:

TON 2 0 8

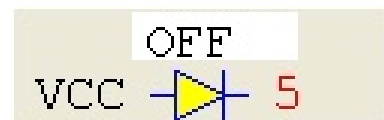
Define un pulsador o tecla que conecta el nodo 2 a masa cuando se pulsa. Se dibuja en forma de botón en la posición (celdilla) 8.

Visualizador binario (LED)

L<nombre> <nodo_ánodo> <nodo_cátodo> <posición> [<color>]

El led se activa si nodo_ánodo se encuentra a nivel lógico 1 y nodo_cátodo a nivel lógico 0. Cuando el led está encendido, el simulador lo representa con el símbolo coloreado utilizando el <color> indicado, o rojo si no se indica. El color se indica mediante una letra:

rojo	R
amarillo	A o bien Y
verde	V o bien G
azul	Z o bien B



Por ejemplo:

LOFF 1 5 9 A

define un LED con el ánodo conectado a +Vcc a través de una resistencia (pull-up) y el cátodo conectado al nodo 5. Se muestra amarillo si está encendido (en este caso cuando el nodo 5 se encuentra a nivel bajo). El simulador lo presentará en la posición (celdilla) 9, con el aspecto de la figura.

Visualizador hexadecimal (p. ej. 7 segmentos) multiplexable.

H <nodo_0(LSB)> ... <nodo_3(MSB)> <nodo_-enable> <posición>

El visualizador presenta el valor dado por LSB...MSB solamente si nodo_-enable=0. Esta característica permite simular visualizadores (*displays*) multiplexados con un comportamiento equivalente al de un *display* de 7 segmentos de cátodo común.

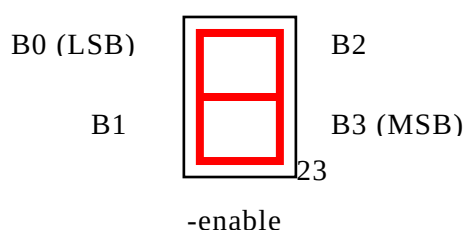


Por ejemplo:

H 4 5 6 7 0 12

define un visualizador hexadecimal con su terminal “-enable” a nivel ‘0’ (siempre visible). El simulador lo mostrará en la posición (celdilla) 12, con el aspecto indicado en la figura. En este caso el valor numérico de entrada es 1001 binario, equivalente al 9 hex.

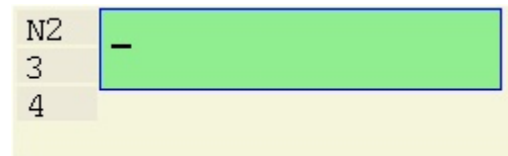
El orden en que se indican las conexiones de los terminales es:



Pantalla LCD de acceso serie

D <nodoCLK> <nodoDAT> <nodoLOAD> <posición>

Esta pantalla LCD posee 2 líneas de 16 caracteres y se controla mediante un enlace serie síncrono que emplea 3 señales. La información se envía bit a bit a través del terminal DAT. Cuando se produce un flanco de bajada en la señal de reloj CLK, el LCD captura el valor presente en DAT en un registro de desplazamiento interno. Cuando se produce un flanco de bajada en la señal LOAD, la pantalla ejecuta el comando contenido en el registro de desplazamiento.



La pantalla reconoce los siguientes comandos:

0 d7 d6 d5 d4 d3 d2 d1 d0

Escribe carácter d7...d0 (ASCII ext.) en la posición actual del cursor, en modo normal o de inserción (ver comandos 1100 y 1101).

1 0 0 a4 a3 a2 a1 a0

Desplaza el cursor a la posición a4...a0. (0= inicio, 31= fin)

1 0 1 0

Ocultar el cursor

1 0 1 1

Muestra el cursor

1 1 0 0

Activa modo INSERTAR

1 1 0 1

Desactiva modo INSERTAR

1 1 1 0

Borra el carácter apuntado por el cursor y desplaza los que se encuentran a su derecha.

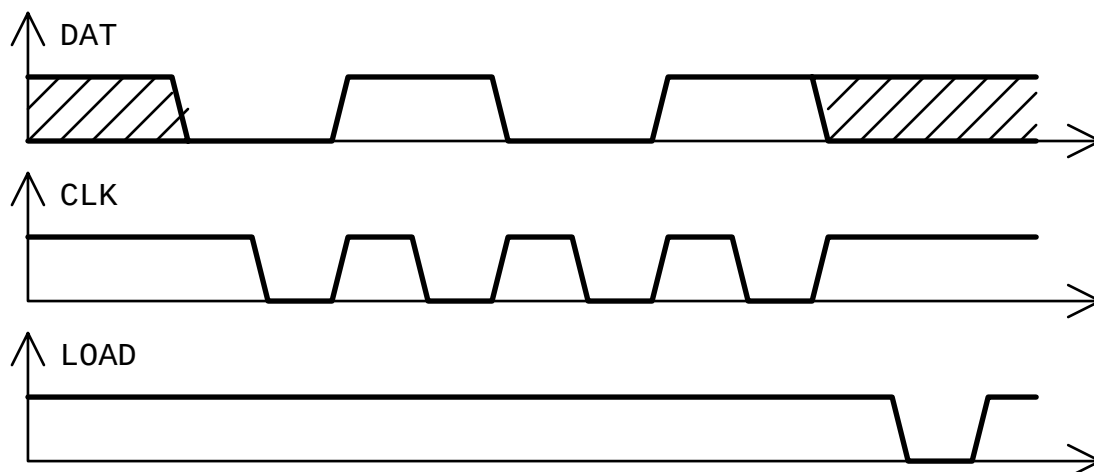
1 1 1 1

Borra toda la pantalla y pone el cursor en la dirección 0 (al inicio).

El envío de los comandos debe empezar con el bit menos significativo. Por ejemplo, para ocultar el cursor se envían los bits:

1 0 1 0

empezando por el 0 de la derecha. El siguiente cronograma muestra la evolución de las señales. Los flancos activos de CLK y LOAD son los de bajada.



Pantalla LCD estándar (de acceso paralelo)

X <nodoRS> <nodoRW> <nodoE> <nodoD0> ... <nodoD7> <posición>

Esta pantalla equivale funcionalmente a una pantalla LCD comercial de 2 líneas de 16 caracteres (p. ej. trident MDLS 16265). Sin embargo, existen algunas diferencias y limitaciones que se comentan más adelante.

Se deben indicar los números de nodo de todos los terminales. En el caso de una conexión mediante bus de 4 bits, los terminales no utilizados (D0 a D3) se deben conectar al nodo 1. Por ejemplo:

N2	
3	
4	
VCC	VCC VCC VCC 14 15 16 17

X 2 3 4 1 1 1 1 14 15 16 17 6

La columna de la izquierda representa de arriba abajo las conexiones de las señales RS, RW y E respectivamente. La fila inferior indica las conexiones del bus de datos con el D0 a la izquierda. Además, se indica el modo actual de operación del LCD (4-bit / 8-bit).

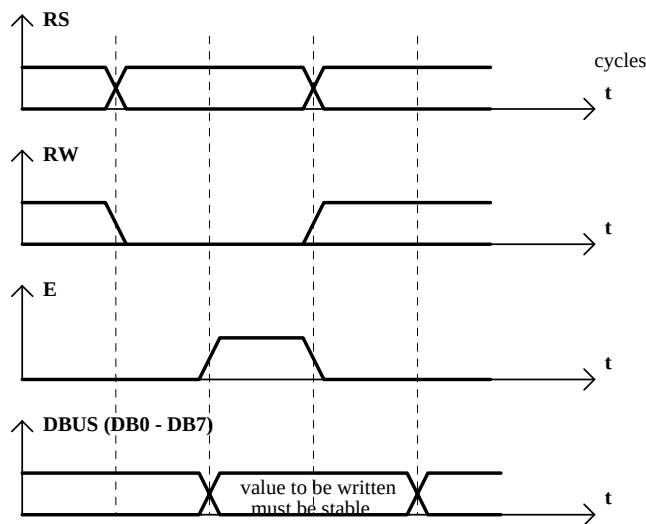
NOTA: En la versión MSDOS no se indicaba el modo de operación (4 bits , 8 bits).

Todos los comandos del conjunto de instrucciones estándar están disponibles. Sin embargo, el formato *de una sola línea* (N=0) no está disponible.

La versión actual no simula los tiempos de operación del procesador de la pantalla. Por lo tanto, la lectura del indicador *busy* (BF) obtendrá siempre el valor 0 (*can accept instruction*).

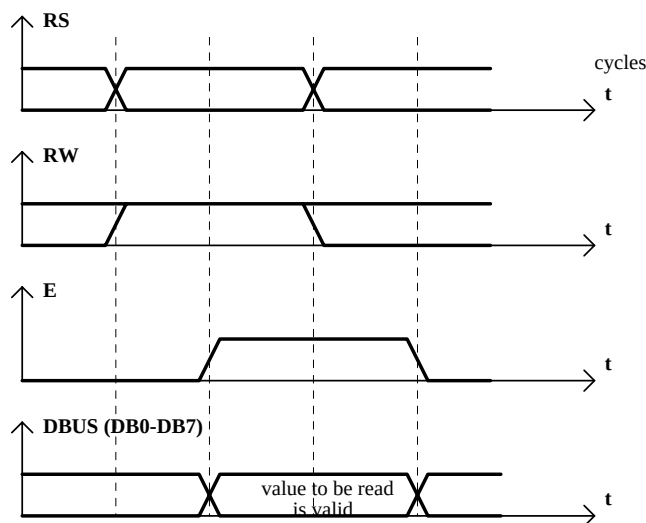
Se ha intentado obtener el mayor realismo posible en la gestión de las señales de control y datos. Sin embargo, dado que el menor incremento de tiempo de la simulación corresponde a la ejecución de una instrucción (ciclo), los cronogramas son ligeramente diferentes a los de una pantalla LCD real. A continuación se indican los cronogramas típicos del LCD simulado.

La figura siguiente muestra el cronograma de escritura empleando un bus de 8 bits. Las señales RS y RW deben definirse al menos un ciclo antes que la subida de la señal E. El valor que se desea escribir debe encontrarse estable en el bus de datos al menos un ciclo antes que la bajada de E y permanecer hasta un ciclo después. La señal E debe permanecer al menos un ciclo a nivel alto. Se permite que las señales RS y RW cambien de estado un ciclo después de la subida de E.



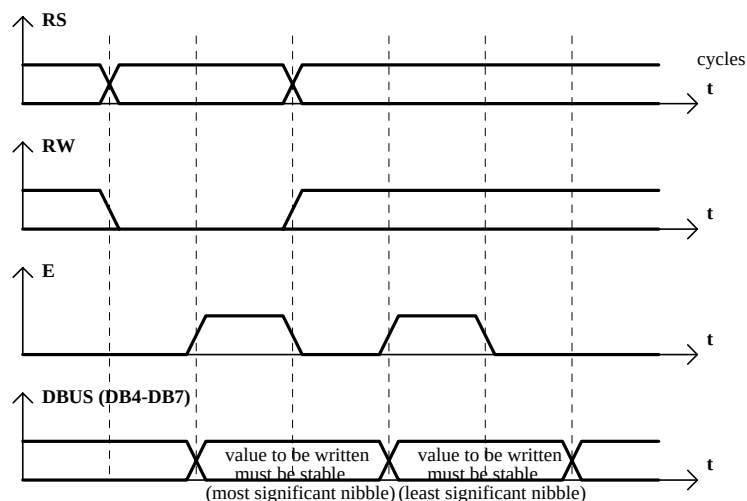
CRONOGRAMA DE ESCRITURA CON BUS DE 8 BITS

La siguiente figura muestra el cronograma de lectura empleando un bus de 8 bits. Las señales RS y RW deben definirse al menos un ciclo antes que la subida de la señal E. El valor leído se hace presente en el bus al subir E y el bus vuelve al tercer estado al bajar E. La señal E debe permanecer al menos dos ciclos a nivel alto. Esto se debe a la necesidad de que el micro realice la lectura del bus mientras el valor permanece estable, es decir en el ciclo intermedio. Se permite que las señales RS y RW cambien de estado un ciclo después de la subida de E.



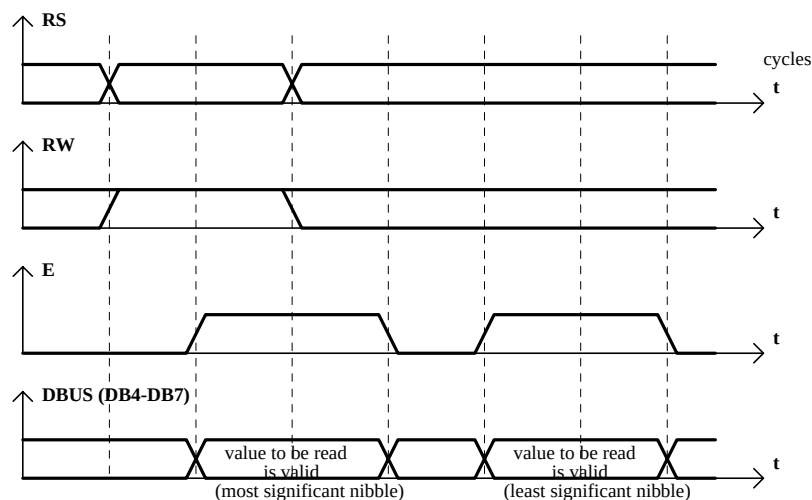
CRONOGRAMA DE LECTURA CON BUS DE 8 BITS

La figura siguiente muestra el cronograma de escritura empleando un bus de 4 bits. Las señales RS y RW deben definirse al menos un ciclo antes que la subida del primer impulso de la señal E. Los 4 bits más significativos del valor que se desea escribir deben encontrarse estables en el bus de datos (DB4 a DB7) al menos un ciclo antes que la bajada del primer impulso de E y permanecer hasta un ciclo después. Los 4 bits menos significativos del valor que se desea escribir deben encontrarse estables en el bus de datos (DB0 a DB3) al menos un ciclo antes que la bajada del segundo impulso de E y permanecer hasta un ciclo después. La señal E debe permanecer, en ambos impulsos, al menos un ciclo a nivel alto.. Se permite que las señales RS y RW cambien de estado un ciclo después de la subida del primer impulso de E.



CRONOGRAMA DE ESCRITURA CON BUS DE 4 BITS

La siguiente figura muestra el cronograma de lectura empleando un bus de 4 bits. Las señales RS y RW deben definirse al menos un ciclo antes que la subida de la señal E. Los 4 bits más significativos del valor leído se hacen presentes en el bus (DB4 a DB7) al subir el primer impulso de E y el bus vuelve al tercer estado al bajar el primer impulso de E. Los 4 bits menos significativos del valor leído se hacen presentes en el bus (DB4 a DB7) al subir el segundo impulso de E y el bus vuelve al tercer estado al bajar el segundo impulso de E. La señal E debe permanecer, en ambos impulsos, al menos dos ciclos a nivel alto. Esto se debe a la necesidad de que el micro realice las lecturas del bus mientras los valores permanecen estables, es decir en los ciclos intermedios. Se permite que las señales RS y RW cambien de estado un ciclo después de la subida del primer impulso de E.



CRONOGRAMA DE LECTURA CON BUS DE 4 BITS

A título de ejemplo se indican las secuencias de pseudo-código típicas de las operaciones de escritura y lectura en el modo de 4 bits.

ESCRITURA:

- 1) RS= {0,1} // según sea envío de un comando o escritura en memoria respectivamente.
- 2) RW= 0.
- 3) E= 1.

- 4) ENVÍO DE LOS 4 BITS MAS SIGNIFICATIVOS AL BUS.
- 5) E= 0.
- 6) E= 1.
- 7) ENVÍO DE LOS 4 BITS MENOS SIGNIFICATIVOS AL BUS.
- 8) E= 0.
- 9) DEJAR EL BUS EN TERCER ESTADO (por si se hace después una lectura).
- 10) YA SE PUEDE MODIFICAR RS Y RW SI SE QUIERE.

LECTURA:

- 1) RS= {0,1} // según sea lectura de flag+address o de memoria respectivamente.
- 2) RW= 1.
- 3) ASEGURARSE DE QUE EL PUERTO DEL BUS ESTÁ A NIVEL ALTO (Hi-Z).
- 4) E= 1.
- 5) LECTURA DE LOS 4 BITS MAS SIGNIFICATIVOS DEL BUS.
- 6) E= 0.
- 7) E= 1.
- 8) LECTURA DE LOS 4 BITS MENOS SIGNIFICATIVOS DEL BUS.
- 9) E= 0.
- 10) YA SE PUEDE MODIFICAR RS Y RW SI SE QUIERE.

En las descripciones anteriores se ha utilizado el término *posición*, que debe ser interpretado como la ubicación en la que el simulador dibuja el periférico dentro del área correspondiente en la ventana del sistema. Existen 32 posiciones posibles que se corresponden con la tabla que se indica a continuación.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32

Analizador lógico

Aarchivo <tinicial> <tfinal>

El parámetro `archivo` define el nombre que el simulador empleará para el archivo de registro del analizador lógico.

Los parámetros siguientes indican los instantes (en tiempo del sistema simulado) en que debe comenzar y finalizar el registro. Si no se indican, se registrará el proceso de simulación completo, desde que se inicie hasta que se salga del simulador. También se permite indicar solamente el instante de inicio, en cuyo caso la simulación no se detiene hasta que se sale del simulador.

El registro se realiza mediante marcas. Cada marca contiene la información sobre el estado de todos los nodos coincidiendo con el cambio en alguno de ellos. El número máximo de marcas está limitado a 1.000.000 (un millón). A partir de este valor, se da por finalizado el registro. De esta forma se limita el tamaño del archivo a un máximo de aproximadamente 30 Mbytes.

NOTA: En la versión MSDOS el límite de marcas se fijaba en 100.000.

A diferencia de los periféricos anteriores, solamente se permite definir un analizador en cada sistema.

La definición de un analizador puede ir seguida de líneas de definición de etiquetas de nodos y de definición de buses.

Etiquetas de nodos

`Nn etiqueta`

siendo *n* el número del nodo (entre 2 y 199) al que se le asigna el nombre *etiqueta*. Ejemplos:

`N5 CLK`

asigna el nombre CLK a la señal presente en el nodo 5. Este nombre se visualizará en la pantalla del analizador junto al cronograma del nodo 5.

`N0 GND`

Asigna el nombre GND al nodo 0.

Definición de buses

`Bnombre nn n0 n1 n2 ...`

siendo *nombre* el nombre asignado al bus y *nn* el número de nodos que lo componen. A continuación se indican los números de los nodos que lo forman empezando por el de menor peso. Ejemplo:

`BDATA 8 2 3 4 5 6 7 8 9`

asigna el nombre DATA a un bus de 8 bits formado por los nodos del 2 al 9, siendo el 2 el de menor peso (LSB).

El orden en que se definen tanto las etiquetas de los nodos como los buses determina el orden en que se presentarán sus cronogramas en el analizador.

Definición de frames

El analizador lógico es capaz de identificar tramas serie asíncronas. Para ello es preciso definir una línea denominada “frame” con el formato

Fnombre nodo baudios

El parámetro “nombre” va a continuación de la F inicial (sin espacios) y es la etiqueta que identifica este “frame”. El número del nodo en el que se busca la trama asíncrona se indica en el parámetro “nodo”. El parámetro “baudios” es un valor numérico que indica la velocidad del enlace serie en bps.

El analizador determina el comienzo de cada trama buscando el primer flanco de bajada de la señal (bit de start) comprendido en la visualización actual de la pantalla. Esta forma de sincronización es similar a la que haría el usuario localizando visualmente el flanco del bit de start. La pantalla del analizador muestra el valor hexadecimal del dato de la trama, suponiendo que se trata de una trama de 8 bits, sin paridad. Si el bit de stop de la trama es incorrecto, el valor del dato se muestra en tono gris para indicar el error.

APENDICE A2: Personalización del entorno UVI-51

Este apéndice describe el método utilizado para elegir los programas empleados para editar y ensamblar. Cuando se le indica al entorno que edite un archivo, éste ejecuta el programa indicado en la cuarta línea del archivo de configuración “Uvi51Cfg.txt”. En la invocación del editor se envía el path completo (nombre y extensión incluidos) del archivo a editar, como parámetro.

Cuando se indica al entorno que ensamble un programa fuente, éste determina en primer lugar cual es el contenido de la línea 7 del archivo de configuración “Uvi51Cfg.txt”:

Si la línea es igual a “ensambla.bat”, entonces ejecuta ese archivo por lotes pasando como primer parámetro el path del archivo SRC y como segundo parámetro el path de Uvi51 (donde se encuentran los ejecutables de la aplicación). En caso contrario ejecuta el programa indicado en la línea 7 pasando solamente un parámetro que contiene el nombre del SRC.

Es muy importante tener en cuenta que sólo se puede utilizar un ensamblador que genere ficheros .LST y .HEX compatibles con UVIENS51.EXE o compatibles con ASM51.EXE de Intel. En caso de duda, se recomienda consultar con los autores del sistema UVI51-win.

Para más detalles se recomienda consultar también el apartado 3.

APENDICE A3: Soporte técnico

El autor no garantiza el correcto funcionamiento del software aquí descrito ni se hace responsable de las consecuencias de su utilización.

Asímismo, no se compromete a facilitar soporte a nivel de consultas técnicas ni garantiza la existencia de nuevas versiones.

Sin embargo intentará, dentro de la medida de sus posibilidades, atender consultas puntuales sobre el funcionamiento del entorno UVI51-win en la siguiente dirección postal:

Alfredo del Río (UVI-51-win)
Dpto. Tecnología Electrónica
ETSII-Universidad de Vigo
36200 VIGO
SPAIN

o en la dirección de correo electrónico: ario@uvigo.es

indicando como asunto UVI51-win.

El autor agradecerá que la consulta venga acompañada de la siguiente información:

- 1) Nombre de la persona que hace la consulta y cargo que ostenta (alumno de determinada Universidad, Profesor, etc.)
- 2) Dirección de correo electrónico a la que desea que se le envíe la respuesta.
- 3) Aplicación a la que destina o va a destinar el entorno UVI51-win (docencia o desarrollo, en qué nivel formativo, etc.).
- 4) Opinión que le merece el entorno UVI-51-win. Errores detectados y/o sugerencias.
- 5) Indicación de si la persona que realiza la consulta ha utilizado previamente alguna de las versiones MS-DOS de Uvi51.

El autor agradece sinceramente su colaboración.

Vigo, 1 de Septiembre de 2008

Alfredo del Río.