



Universidad de La Laguna
Escuela Técnica Superior de Ingeniería Informática

SIMDE

**Un Simulador para el Apoyo
Docente en la Enseñanza de las
Arquitecturas ILP con
Planificación Dinámica y Estática**

Iván Castilla Rodríguez

MANUAL DE USUARIO

Junio 2004

1 Primer contacto

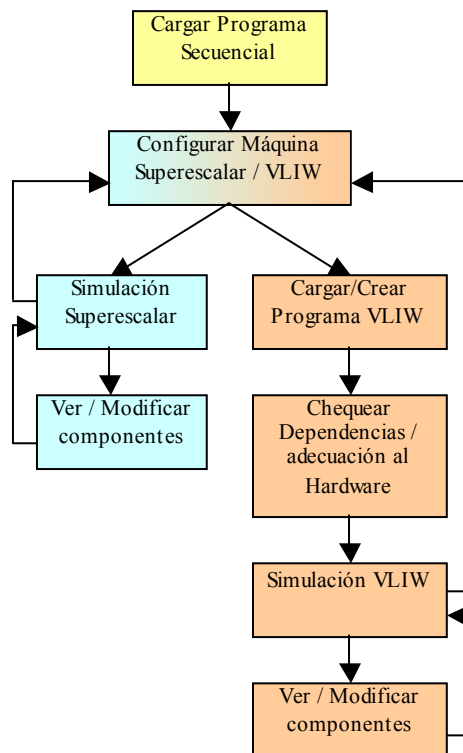


Ilustración 1. Esquema básico de uso del programa

Para comenzar a usar el programa basta con hacer *doble clic* sobre el icono del ejecutable. Se mostrará la pantalla principal del programa, desde la que se tiene acceso a todas las opciones del simulador mediante el menú y las barras de herramientas.

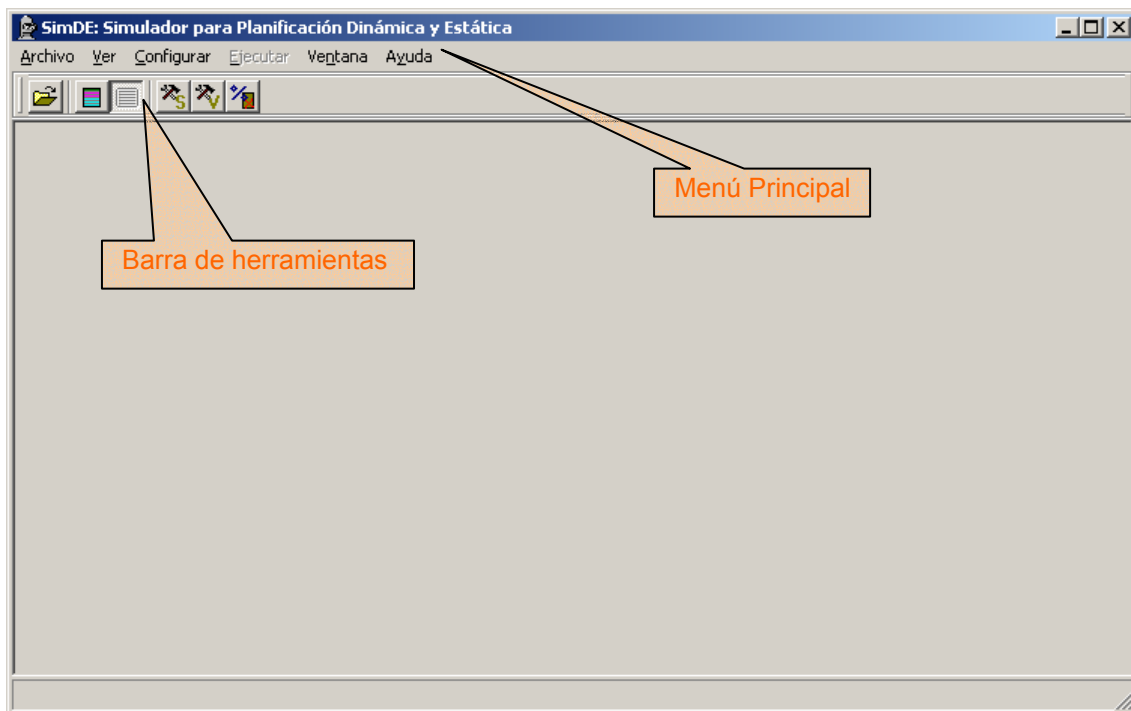


Ilustración 2. Pantalla principal del simulador

Un esquema del uso del programa puede verse en la Ilustración 2. En este esquema pueden verse a grandes rasgos las funcionalidades del programa.

1.1 Menú

Los menús del programa pueden resumirse de la siguiente manera:

- **Archivo:** Contiene las opciones para abrir un código secuencial que está en un fichero y para cerrar la aplicación.
- **Ver:** Permite activar o desactivar la visualización de las barras de herramientas, y también jugar con las opciones de visualización del código secuencial.
- **Configurar:** Contiene las opciones de configuración de los parámetros de las máquinas, así como el resto de parámetros más generales (como fallos de caché). Además contiene el acceso a las herramientas de construcción de código VLIW.
- **Ejecutar:** Permite escoger la máquina con la que realizar la simulación y controlar la simulación en sí, así como acceder a los componentes de las máquinas (memoria, registros...).
- **Ventana:** Son las opciones de visualización de las distintas ventanas de la aplicación
- **Ayuda:** Accede a la ayuda de la herramienta.

1.2 Acceso a la ayuda

En cualquier momento se tiene acceso a la ayuda usando la tecla **F1**.

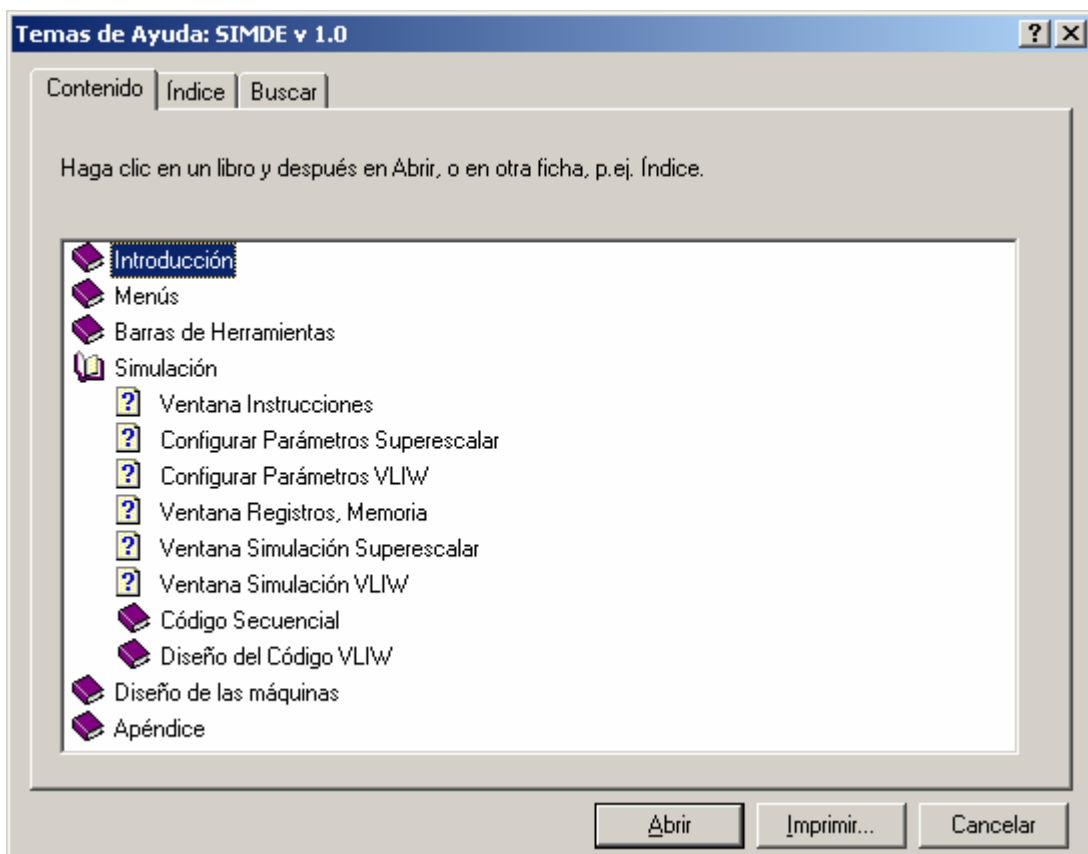


Ilustración 3. Ayuda de la aplicación

Existe ayuda acerca del uso del simulador en general, detallando el uso de menús y barras de herramientas, así como las opciones disponibles desde cada ventana. Además hay una amplia sección con una descripción de las máquinas y todos sus elementos, y un apéndice con referencias a otros simuladores y documentos.

1.3 Barra de Herramientas Principal

La barra de herramientas principal tiene a mano algunas de las opciones más importantes del simulador:



Abrir: Permite abrir un fichero de código secuencial (.pla) para usarlo en las simulaciones.



Colorear bloques básicos: Asigna un color diferente a cada bloque básico del código para poder distinguirlos. Esta opción sirve de ayuda en la construcción de códigos de instrucciones largas.



Mostrar/Ocultar código secuencial: Si está seleccionada muestra el código secuencial. Al desmarcar esta opción se oculta el código (pero no se descarga, y puede continuarse con las simulaciones).



Configurar Superescalar: Abre la ventana de configuración de la máquina superescalar.



Configurar VLIW: Abre la ventana de configuración de la máquina VLIW.



Porcentaje de fallos de caché: Permite modificar el porcentaje de fallos de la caché de datos.

2 Crear un código secuencial

Los ficheros básicos para la aplicación son los ficheros de código secuencial. Para crear un código secuencial debe usarse un **editor de texto** (*notepad* o similar), respetando la siguiente estructura:

- La primera línea del fichero (que no sea un comentario) contiene el número de instrucciones del fichero.
- Cada instrucción debe ponerse en una nueva línea.
- Se permiten como separadores de operandos tabuladores o espacios.
- Las etiquetas se ponen al principio de la línea, permiten cualquier carácter alfanumérico y deben terminar con “:”. Etiquetas válidas son: “LOOP:”, “bucle1:”, “2:”...
- Los comentarios se indican con “//” y abarcan hasta el final de la línea.

La sintaxis exacta de las instrucciones puede verse en la Tabla 1.

Tipo de instrucción	Sintaxis	
Entera	DADDUI	Rn Rm #i
	ADDI	Rn Rm Rp
	MULTI	Rn Rm Rp
Punto Flotante	ADDF	Fn Fm Fp
	MULTF	Fn Fm Fp
Memoria	LI	Rn i(Rm)
	LF	Fn i(Rm)
	SI	Rn i(Rm)
	SF	Fn i(Rm)
Salto	BNE	Rn Rm LAB
	BEQ	Rn Rm LAB

Tabla 1. Sintaxis de las instrucciones en los ficheros .pla

Un resumen de la nomenclatura empleada se detalla en la Tabla 2.

Símbolo	Explicación	Ejemplo
Rn	Registro de Propósito General n	R1, R0...
Fm	Registro de Punto Flotante m	F1, F0...
#i	Valor inmediato i	#12, #0...
i(Rm)	Dirección de memoria	(R1), 3(R4)...
LAB	Etiqueta destino de un salto	LOOP1, END...

Tabla 2. Nomenclatura empleada en el código de los ficheros de entrada

El fichero creado debe guardarse con la extensión “.pla”. Un primer fichero para realizar pruebas puede verse el Ejemplo 1.

3 Abrir un fichero de código secuencial

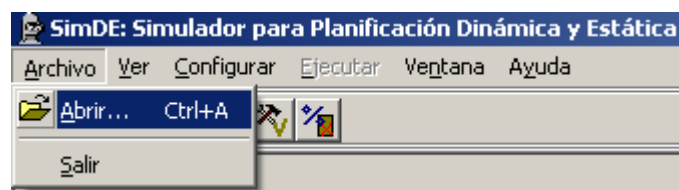


Ilustración 4. Abrir un fichero secuencial

Cargar el programa creado para usarlo en las simulaciones puede hacerse usando el menú **Archivo** y eligiendo la opción de **Abrir**, o presionando “CTRL + A”. Se permitirá navegar por los directorios para buscar el fichero “.pla” creado.

bucle.pla				
Nº	OPCODE	OP1	OP2	OP3
0	DADDUI	R2	R0	#50
1	DADDUI	R3	R0	#70
2	DADDUI	R4	R0	#40
3	LF	F0	(R4)	
4	DADDUI	R5	R2	#16
5 [LOOP:]	LF	F1	(R2)	
6	ADDF	F1	F1	F0
7	SF	F1	(R3)	
8	DADDUI	R2	R2	#1
9	DADDUI	R3	R3	#1
10	BNE	R2	R5	LOOP

Ilustración 5. Código secuencial cargado

El fichero abierto se mostrará en la parte izquierda de la pantalla (ver Ilustración 5). El código se muestra en el orden secuencial original. En la primera columna aparece el número de orden de la instrucción, que servirá como **identificador**. Si la instrucción tenía una etiqueta asociada, se mostrará ésta entre corchetes ([etiqueta:]) justo después del identificador. La segunda columna es el código de la operación (*opcode*) y en las siguientes se muestran los operandos.

```

// SIMDE v1.0
// Autor: Iván Castilla Rodríguez
// Descripción: El programa presupone q en la posición 50
// (R2) de memoria tienes un vector de 16 elementos y quieres
// sumar a cada elemento una cantidad fija (en la posición de
// memoria 40). El resultado se coloca a partir de la
// posición 70 (R3) de memoria.
11
    DADDUI    R2 R0 #50
    DADDUI    R3 R0 #70
    DADDUI    R4 R0 #40
    LF        F0 (R4)
    DADDUI    R5 R2 #16
LOOP:
    LF        F1 (R2)
    ADDF      F1 F1 F0
    SF        F1 (R3)
    DADDUI    R2 R2 #1
    DADDUI    R3 R3 #1
    BNE       R2 R5 LOOP
// Fin del programa

```

Ejemplo 1. Fichero bucle.pla

Una vez abierto el fichero, pueden probarse algunas opciones. Para ocultar el código y dejar más espacio en la pantalla para seguir las simulaciones se puede acudir al menú **Ver** y seleccionar **Código Secuencial**. La misma opción está accesible desde la barra de herramientas (☰) o haciendo *doble clic* sobre la parte superior del código, donde se muestra el nombre del fichero cargado. Para volver a ver el código basta con seleccionar nuevamente esta opción. El código oculto no se descarga y puede continuarse con las simulaciones.

Otra de las opciones interesantes es el **coloreado de bloques básicos**. Esta opción puede activarse/desactivarse desde el menú **Ver**, desde la barra de herramientas (🎨) o pulsando “**CTRL + B**”. Los bloques básicos son especialmente útiles para construir códigos VLIW.

Tras cargar un código secuencial, puede comenzarse inmediatamente con las simulaciones superescalares. Las simulaciones VLIW necesitan construir o cargar un código VLIW previamente.

4 Ejecución de una simulación

Una vez cargado el fichero secuencial se permite el acceso a las opciones de ejecución mediante la barra de herramientas de ejecución (ver Ilustración 6) o el menú **Ejecutar**.



Ilustración 6. Barra de herramientas Ejecución

En ambos casos se puede acceder a las mismas opciones, que se detallan a continuación, incluyendo entre paréntesis la tecla de acceso rápido:



Iniciar (F9): Permite comenzar la ejecución continua de la simulación. Si se pulsa mientras está ejecutándose otra simulación, se puede escoger entre

comenzar desde el principio la ejecución o continuar la ejecución actual. La ejecución continua sólo se detiene al llegar al final del programa o si encuentra un *breakpoint*.



Paso a Paso (F8): Permite avanzar ciclo a ciclo en la ejecución de una simulación.



Parar (F6): Detiene una ejecución y pone el reloj a 0.



Pausa (F7): Permite pausar una ejecución iniciada en modo continuo. Una ejecución pausada puede proseguirse en modo continuo o en modo paso a paso.

Contador de ciclos (sólo en la barra de herramientas): Indica el ciclo de reloj actual en la simulación que se está ejecutando. Este valor es solamente informativo y no puede modificarse.

Velocidad de la simulación: Permite incrementar o decrementar la velocidad a la que se visiona la simulación continua. Se puede establecer cualquier valor desde 1 hasta 10 mediante los botones ▲ y ▼, donde 1 indica la velocidad más lenta de simulación y 10 es la velocidad más rápida.

Memoria	▼
Memoria	
Registros de Propósito General	
Registros de Punto Flotante	

Componentes: Mediante este cuadro combinado o el menú correspondiente pueden mostrarse los distintos componentes genéricos de las máquinas VLIW y Superescalar. Al seleccionar la memoria o cualquiera de los dos bancos de registros se mostrará una ventana de componentes como se verá en el punto 4.1. El componente que se muestra se corresponde con la máquina seleccionada para realizar la simulación.

Superescalar	▼
Superescalar	
VLIW	

Seleccionar Máquina (F2 para la máquina Superescalar, F3 para la VLIW): Con este cuadro combinado se escoge la máquina con la que se quiere llevar a cabo la simulación. La máquina VLIW sólo se permite si hay un código de instrucciones largas correctamente cargado o creado además del código secuencial.

Una ejecución, tanto continua como paso a paso, avanzará hasta llegar el final del programa. En ese momento aparecerá un mensaje informando de tal evento (ver Ilustración 7).



Ilustración 7. Fin de ejecución

Pueden realizarse tantas ejecuciones con una máquina como se quiera. Lo que hay que tener en cuenta es si se desea que con cada nueva ejecución la máquina se resetee o conserve los valores de la última ejecución. Esto se consigue en el menú **Configurar** → **Opciones**, y en el submenú se puede marcar o no la opción de **Resetear máquina al iniciar** (☐).

4.1 Ventanas de Componentes

Existen tres tipos de componentes genéricos: la memoria, los registros de propósito general (GPR) y los registros de punto flotante (FPR). Estos tres componentes aparecen en las dos máquinas, VLIW y Superescalar. Los componentes que se muestran en pantalla siempre se corresponden con la máquina actualmente seleccionada para hacer la simulación, es decir, no pueden verse simultáneamente la memoria de la máquina VLIW y la de la máquina Superescalar. En la primera columna se muestra el índice o la posición del elemento, mientras en la segunda columna puede verse (y **modificarse**) el valor correspondiente (por claridad, los números en punto flotante se presentan siempre redondeados a 3 decimales). Para este valor se permiten números enteros en el caso de los GPR y también flotantes en el caso de la memoria o FPR.

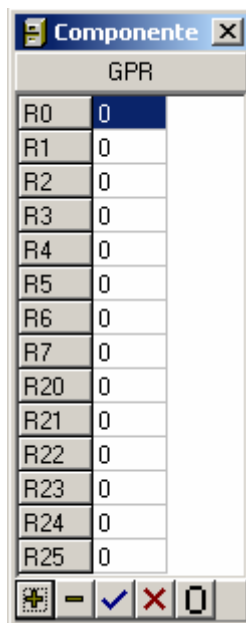


Ilustración 8. Ventana de componente genérico. En concreto, banco de GPR.

Tipo de número	Sintaxis permitida	Ejemplos
Entero	[+-]?[0-9]+	-98 1452 0
Flotante	[+-]?[0-9]*(";"[0-9]+)?([eE][+-]?[0-9]+)?	-1 0,345 0,1E-4 3E10

Tabla 3. Sintaxis permitida de los valores de los componentes

Por defecto se muestran los 8 primeros elementos del componente, aunque se dispone de una serie de botones con los que manipular la presentación y el propio contenido de los mismos:

- Añadir:** Muestra un cuadro de diálogo (Ilustración 9) mediante el que puede seleccionarse un subconjunto de elementos para **mostrar** mediante una lista de números o intervalos separados por comas.
- Quitar:** Muestra un cuadro de diálogo (Ilustración 9) mediante el que puede seleccionarse un subconjunto de elementos para **ocultar** mediante una lista de números o intervalos separados por comas.

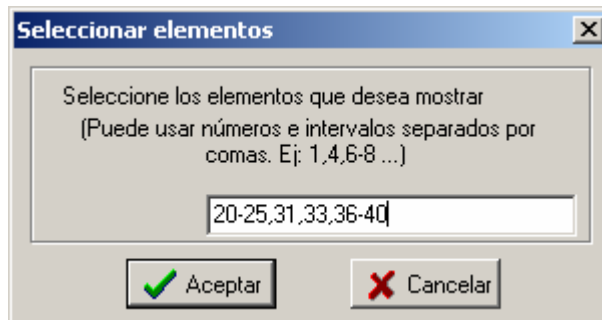


Ilustración 9. Cuadro de diálogo de selección de elementos. En el ejemplo se están seleccionando los elementos: 20, 21, 22, 23, 24, 25, 31, 33, 36, 37, 38, 39 y 40.

- Guardar cambios:** Guarda los valores modificados en el componente de la máquina.
- Cancelar cambios:** Carga de nuevo los valores que había almacenados en ese componente obviando los cambios realizados.
- Resetear valores:** Pone a 0 todos los componentes seleccionados de la máquina.

4.2 Porcentaje de fallos de caché de datos

Uno de los parámetros fundamentales durante la ejecución es el **porcentaje de los fallos de la caché de datos**. Este porcentaje establece la probabilidad de que un acceso de lectura a memoria no encuentre el dato en la caché y deba acudir a la memoria principal, con el consecuente incremento en el tiempo de acceso. Para modificar este valor puede irse al menú **Configurar** → **Opciones** y escoger el botón correspondiente (🔧). La misma opción está disponible desde la barra de herramientas principal. Cualquier valor entre 0 (sin fallos) y 100 (todos los accesos se hacen a memoria principal) está permitido.

5 Simulación Superescalar

5.1 Configuración de la máquina Superescalar

Antes de comenzar una simulación superescalar debe establecerse la configuración de la máquina sobre la que se harán las pruebas. Para ello se puede ir al menú **Configurar** y escoger la opción **Configurar Superescalar**, o presionar “**ALT + F2**”. La ventana de configuración (Ilustración 10) permite modificar el número de unidades funcionales de cada tipo, la latencia o duración de la ejecución en cada unidad, y el grado de emisión de la máquina.

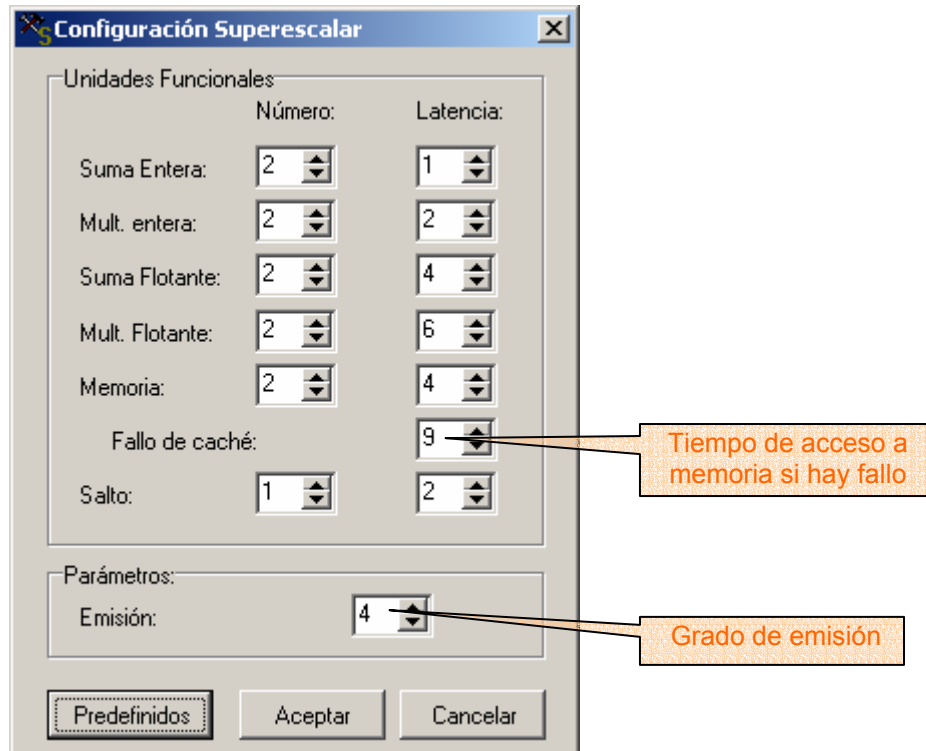


Ilustración 10. Pantalla de configuración de la máquina Superescalar

El grado de emisión se corresponde con el número de instrucciones que se emiten desde el decodificador al ROB y las estaciones de reserva por ciclo, pero también con el número de instrucciones graduadas (*commit*) por ciclo.

Entre las latencias puede indicarse también la duración del acceso a memoria si ocurre un fallo de caché. Este parámetro no se suma a la latencia de la unidad funcional de memoria, sino que es un tiempo absoluto. P. Ej. si la latencia de la UF de memoria es 4 y el tiempo del fallo de caché es 9, en caso de fallo se tardarán 5 ciclos adicionales en acceder a esa palabra de memoria.

La modificación de los valores se hace efectiva al hacer *clik* en el botón de **Aceptar**. Con **Cancelar** no se guardan los cambios y se puede volver a los valores por defecto con el botón de **Predefinidos**.

5.2 Ejecución de una simulación Superescalar

Para ejecutar una simulación superescalar basta con tener cargado un código secuencial y seleccionar la máquina Superescalar (desde la barra de herramientas de ejecución, desde el menú **Ejecutar** → **Seleccionar Máquina**, o presionando **F2**). Al iniciar una ejecución continua o “paso a paso” se presentará en pantalla el esquema de la máquina Superescalar (Ilustración 11) con los siguientes componentes:

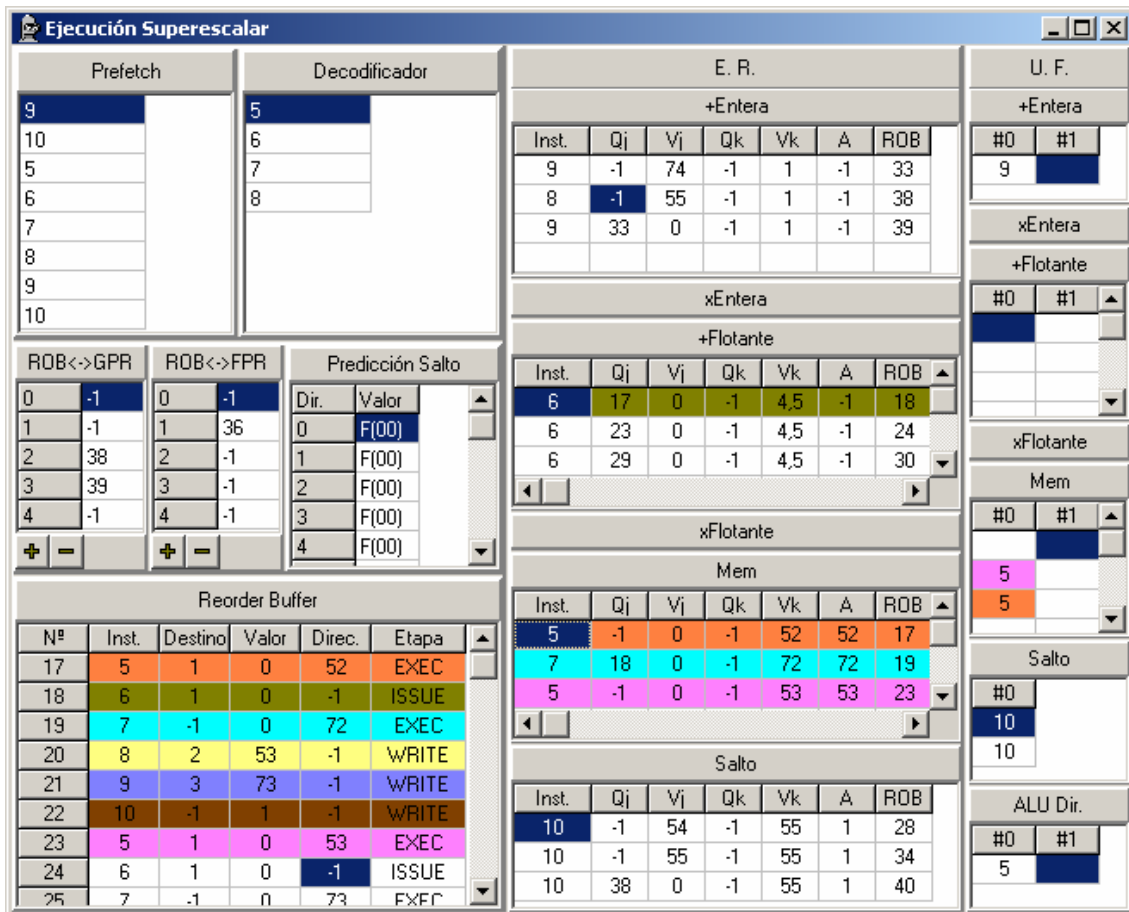


Ilustración 11. Ventana de ejecución Superescalar

- **Prefetch:** Unidad de prebúsqueda de instrucciones. Esta unidad busca las 2**emisión* próximas instrucciones del programa. Sin embargo no las busca en orden secuencial, sino que tiene en cuenta la tabla de predicción de salto para cargar aquellas instrucciones que se correspondan con la predicción.
- **Decodificador:** Decodificador de las instrucciones. A esta unidad pasan las *emisión* primeras instrucciones de la unidad de prebúsqueda. Después se encarga de decodificarlas y de la emisión en sí, comprobando si hay espacio libre en el *reorder buffer* y en las estaciones de reserva correspondientes.
- **ROB<->GPR:** Posición del ROB donde se está procesando un registro de propósito general (-1 si el registro no está asociado a ninguna entrada del ROB). Se muestra sólo un subconjunto de los elementos. Se pueden añadir o quitar elementos mediante los botones **+** y **-**, indicándolos como una lista de números (o intervalos) separada por comas (Ej. 2,12-18,20).
- **ROB<->FPR:** Posición del ROB donde se está procesando un registro de punto flotante (-1 si el registro no está asociado a ninguna entrada del ROB). Se muestra sólo un subconjunto de los elementos. Se pueden añadir o quitar elementos mediante los botones **+** y **-**, indicándolos como una lista de números (o intervalos) separada por comas (Ej. 2,12-18,20).
- **Predicción Salto:** Tabla de predicción de salto de 16 entradas (4 bits). La predicción de un salto se coloca en la entrada cuyo índice se corresponda con los 4 últimos bits de la dirección en memoria de dicho salto. Si una entrada tiene el

valor F es que el salto que coincide con esa dirección no se toma; si vale V el salto se toma. Además de esto se muestra el valor binario de la entrada de la tabla entre paréntesis.

- **Reorder Buffer (ROB):** Buffer circular que contiene todas las instrucciones que han sido emitidas y aún no han finalizado completamente su ejecución (no se han graduado). Si está lleno obliga a detener la emisión de las instrucciones hasta que disponga de una posición libre. Muestra la siguiente información de cada instrucción:
 - **Identificador** de la instrucción.
 - Índice del registro **Destino** de la operación que realiza la instrucción (si lo tiene).
 - **Valor** resultado de la operación que ha realizado la instrucción (si lo tiene).
 - **Dirección** de memoria destino (*STORE*) u origen (*LOAD*) de la instrucción
 - **Etap**a en la que está en ese momento la instrucción. Su valor puede ser *ISSUE*, *EXECUTE*, *WRITERESULT* ó *COMMIT*.

- **E.R.:** Estaciones de Reserva. Son *buffers* que contienen a las instrucciones desde que son emitidas hasta que termina su ejecución (terminan de usar la unidad funcional correspondiente). La instrucción se mantiene mientras está a la espera de tener todos sus operandos disponibles, es decir, hasta que puede ser enviada (*dispatched*) a una unidad funcional para ser ejecutada. La instrucción se retira de la ER cuando ha finalizado su procesamiento en esa unidad funcional. Existen una ER por cada tipo de unidad funcional, estando etiquetadas en su parte superior indicando esta relación. Los campos que almacena la ER son:
 - **Identificador** de la instrucción.
 - **Q_j:** Disponibilidad del primer operando. -1 indica que el valor se encuentra en V_j ; otro valor indica la posición del ROB donde se está procesando el operando.
 - **V_j:** Valor del primer operando si Q_j vale -1.
 - **Q_k:** Disponibilidad del segundo operando. -1 indica que el valor se encuentra en V_k ; otro valor indica la posición del ROB donde se está procesando el operando.
 - **V_k:** Valor del segundo operando si Q_k vale -1.
 - **A:** Almacenamiento temporal para calcular la dirección efectiva de una instrucción de memoria. Inicialmente contiene el *offset*.
 - **ROB:** Posición del ROB donde se encuentra también la instrucción, y donde se almacenará, si corresponde, el resultado de la misma.

- **U.F.:** Unidades Funcionales. Son las unidades básicas de ejecución de la máquina. En la parte superior aparece el tipo de UF de que se trata entre suma entera (+Entera), multiplicación entera (xEntera), suma flotante (+Flotante), multiplicación flotante (xFlotante), memoria (Mem) y salto. Por cada tipo de UF hay una tabla con tantas columnas como UF de ese tipo estén declaradas. Las filas representan el *pipeline* de la UF.

5.3 Seguimiento de instrucciones

Debido a la ingente cantidad de información que se muestra en la ejecución superescalar resulta complicado el seguimiento de una instrucción en concreto. Para facilitar esta tarea se permite colorear cada instrucción del ROB con un color elegido por el usuario. Haciendo *dobles clic* sobre una entrada del ROB que contenga una instrucción se presentará una ventana con la que escoger el color. La instrucción se coloreará en el ROB y también en la estación de reserva y/o unidad funcional que corresponda.

Para quitarle el color a la instrucción basta con volver a hacer *dobles clic* con el ratón en la entrada correspondiente del ROB.

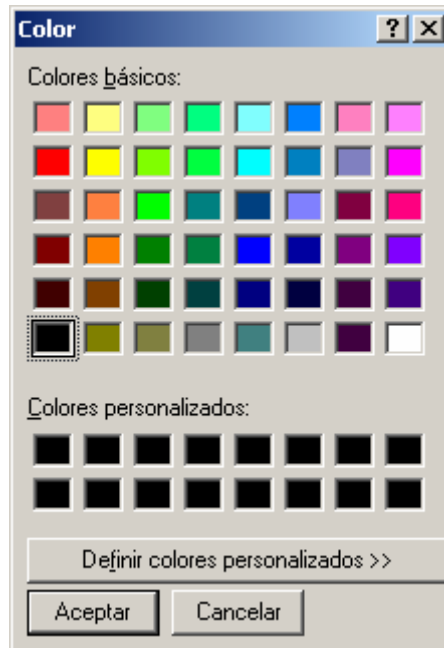


Ilustración 12. Cuadro de diálogo de selección de color de la instrucción

5.4 Breakpoints

Los *breakpoints* sirven para detener una ejecución continua en un punto del código determinado. En el caso de la máquina Superescalar, el *breakpoint* se indica haciendo *dobles clic* sobre la instrucción en la ventana que muestra el código secuencial cargado. El identificador de la instrucción queda marcado en rojo hasta que se vuelva a hacer *dobles clic* en la instrucción.

La ejecución se detendrá cuando se intente cargar la instrucción marcada en la unidad de prebúsqueda (*prefetch*), y podrá reiniciarse en modo continuo (▶) o paso a paso (⏏).

5.5 Redimensionamiento de los componentes

Otra de las opciones más interesantes de la ventana de ejecución superescalar es la posibilidad de cambiar el tamaño de casi cualquier componente para ajustar mejor la presentación a las necesidades del usuario. El **alto y ancho de la mayoría de los componentes puede modificarse** poniendo el ratón sobre alguno de sus bordes y arrastrando manteniendo presionado el botón izquierdo del ratón.

Tanto las estaciones de reserva como las unidades funcionales pueden **ocultarse** haciendo *dobles clic* sobre el nombre del componente correspondiente. Para volver a mostrarlo basta con volver a hacer *dobles clic*.

6 Código de instrucciones largas

Para poder realizar simulaciones con la máquina VLIW debe disponerse de un código de instrucciones largas, construido a partir de algún código secuencial. El código VLIW depende también de la configuración actual de la máquina (ver sección 7.1), ya que existe una correspondencia directa entre el número y tipo de unidades funcionales y el formato de la instrucción larga.







Estos códigos pueden crearse directamente a partir de un código secuencial o cargarlos desde un fichero. Para cargar un programa VLIW desde un fichero hay que ir al menú **Configurar** → **Instrucciones VLIW** y escoger la opción de “**Cargar Programa VLIW...**” (📁). Se presentará un cuadro de diálogo para escoger el fichero “.vliw” deseado. Tras escoger un fichero se mostrará la ventana de edición de código VLIW (Ilustración 13).

6.1 Diseño del código VLIW

Tanto si se quiere editar un código VLIW ya creado cargado desde un fichero, como si lo que se quiere es construir un nuevo código partiendo de cero, hay que trabajar con la ventana de diseño de Código VLIW. Esta ventana se compone de una **Rejilla de Diseño** en la que cada fila es una instrucción y cada columna, una UF de la máquina. La primera columna es el identificador de la instrucción larga.

Para crear un código partiendo de cero hay que ir al menú **Configurar** → **Instrucciones VLIW** y escoger la opción de “**Crear Nuevo Programa VLIW**” (📁).

Se dispone de una serie de botones para facilitar la creación del código:

-  **Añadir:** Permite añadir más instrucciones (filas) a la Rejilla de Diseño. Al pulsarlo se solicitará el número de instrucciones que se quiere añadir.
-  **Eliminar:** Permite eliminar la instrucción seleccionada de la Rejilla de Diseño. Al eliminar una instrucción todas las instrucciones posteriores decremantan su identificador.
-  **Limpiar:** Deja la ventana de diseño en blanco, borrando todo el código construido hasta el momento.
-  **Aceptar:** Guarda los cambios realizados en el código. Al pulsar **Aceptar** se realizan una serie de comprobaciones sobre el código si está marcada la opción “**Chequear código VLIW**” en el menú **Configurar** → **Opciones**. Los mensajes de error se devuelven de la forma: “*Error en la operación # de la instrucción larga #*”, con lo que resulta sencillo localizar el fallo.
-  **Cancelar:** Cancela los cambios realizados desde la última vez que se usó el botón de **Aceptar**.
-  **Guardar:** Guarda el código en un fichero. Se presenta un cuadro de diálogo que permite escoger el nombre del fichero “.vliw” donde guardar el código creado. Este código puede cargarse posteriormente con la opción “**Cargar Programa VLIW...**” (📁) del menú **Configurar** → **Instrucciones VLIW**.

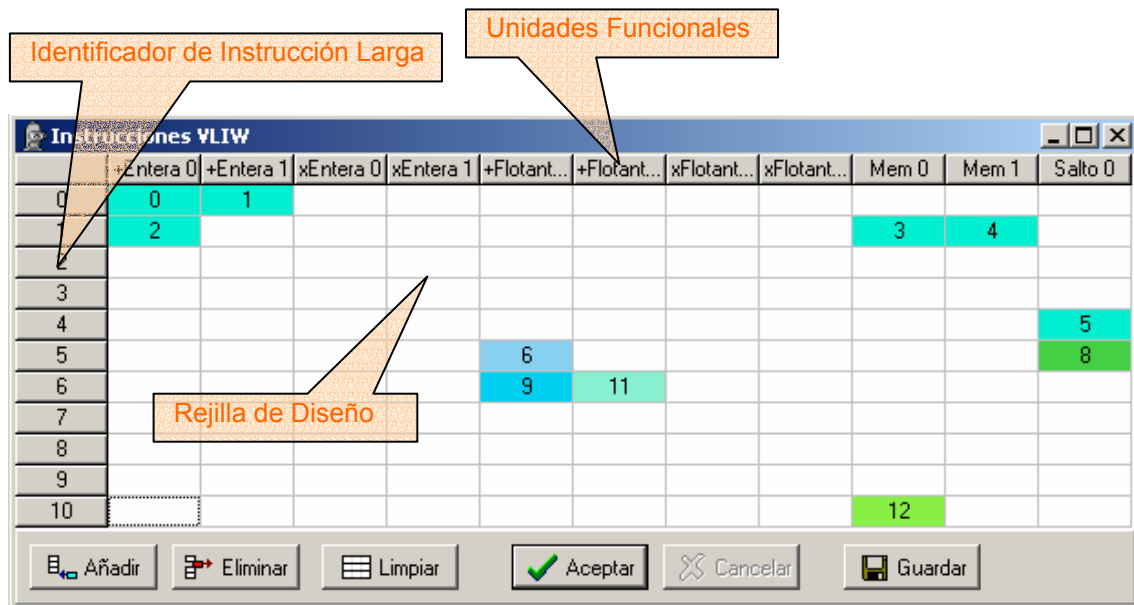


Ilustración 13. Pantalla de construcción de código VLIW

6.1.1 Añadiendo-Eliminando operaciones

La creación del código VLIW es muy sencilla. Basta con arrastrar las operaciones desde la ventana de código secuencial hasta la rejilla de diseño. El programa sólo permite arrastrar las operaciones a una UF donde puedan ser ejecutadas. Una vez copiada una operación, puede eliminarse seleccionándola y pulsando la tecla **SUPR**.

6.1.2 Zona de influencia

Durante la creación del código puede observarse la zona de influencia de una operación haciendo **CTRL + clic** con el botón izquierdo del ratón sobre la misma. De esta forma se colorean las instrucciones largas durante las cuales no pueden planificarse operaciones que tengan operandos dependientes de la operación marcada o, en el caso de los saltos, las instrucciones en las que pueden usarse los registros de predicado asociados a ese salto.

6.1.3 Operaciones de salto

Las operaciones de salto deben ser configuradas en el momento de arrastrarlas. Para ello se mostrará un cuadro de diálogo (ver Ilustración 14) en el que debe indicarse:

- **Instrucción destino:** La instrucción larga destino del salto.
- **Registro Predicado Verdadero:** El Registro de Predicado que cambiará su valor a VERDADERO si el salto es tomado.
- **Registro Predicado Falso:** El Registro de Predicado que cambiará su valor a VERDADERO si el salto NO es tomado.
- **Registro Predicado:** Registro de Predicado asociado a esta operación.

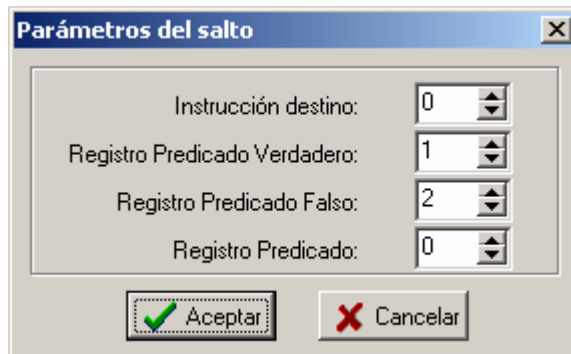


Ilustración 14. Cuadro de diálogo de parámetros del salto VLIW

6.1.4 Parámetros de las operaciones

Los parámetros de una operación pueden ser modificados en cualquier momento haciendo *doble clic* sobre la operación en concreto. Si se trata de una operación de salto aparecerán los parámetros que se nombraron antes; en otro caso simplemente se solicita el Registro de Predicado asociado a la operación.

6.2 Consejos en la creación del código

La creación del código VLIW a partir del código secuencial es una tarea compleja, más aun si es el propio usuario el que tiene que hacer el trabajo del compilador. Se incluye en esta sección una serie de consejos y recomendaciones que pueden ayudar a obtener un código que funcione lo más rápidamente posible.

6.2.1 Consejos Generales

- A partir de un código secuencial sólo se puede jugar con la colocación de las operaciones en el código VLIW. Las optimizaciones del código (desenrollado de bucles, software-pipelining...) se realizan sobre el código secuencial, creando un nuevo fichero de entrada.
- El hardware de la máquina VLIW no realiza ninguna optimización sobre el código que se le da, sino que todas las optimizaciones las realiza el compilador-usuario. Para obtener tiempos de ejecución similares a la máquina Superescalar debe tenerse especial cuidado en la colocación de las operaciones en las instrucciones largas y la mayor optimización posible del código secuencial.

6.2.2 Consejos de creación de código

- La forma más sencilla de construir el código es comenzar a arrastrar las operaciones en orden desde el código secuencial a la primera UF libre del tipo correspondiente que se encuentre.
- Es importante fijarse en las dependencias RAW antes de arrastrar una operación. Colorear la zona de influencia de una operación (haciendo doble clic) puede ayudar a saber dónde colocar la siguiente instrucción.
- Los bucles son una parte fundamental del código. Es importante tener bien presente que una operación de salto cuyo destino sea una operación secuencial se convierte en un salto a una instrucción larga al pasarlo al código VLIW.
- El destino más obvio de un salto es la primera instrucción que contenga una operación del bloque básico destino del salto, pero no siempre es la mejor opción.

- Debe recordarse que las instrucciones largas se ejecutan completamente, no la mitad o sólo algunas de las operaciones. Hay que tener esto en cuenta a la hora de poner en una misma instrucción operaciones que son destino de un salto con otras que no lo son.
- Es interesante aprovechar la posibilidad de repetir operaciones en el código. Tal vez se quiera que la primera iteración de un bucle se ejecute de una forma para después aplicar predicación el resto de iteraciones.
- El uso de registros de predicado es una herramienta bastante potente que permite ejecutar en paralelo operaciones de las dos ramas de un salto.
- Debe recordarse que se puede predicar cualquier operación, incluso otra operación de salto.
- La opción de Chequear Código VLIW no asegura que el código obtenido sea 100 % fiable, pero sí ayudará a detectar algunos errores.
- El aprovechamiento de la configuración de los bancos de registros, que leen en la primera mitad del ciclo y escriben en la segunda mitad puede ahorrar un ciclo en muchas ocasiones.

6.2.3 Posibles errores

- Hay que tener cuidado al asignar un registro de predicado a una operación. Debe asegurarse que esté en la zona de influencia del salto que utiliza esos registros de predicado.
- Para poder predicar una operación no debe terminar su ejecución antes de evaluar el salto, porque en ese caso no podrá cancelarse a tiempo si resulta pertenecer a la rama no tomada del salto.
- El algoritmo empleado para chequear el código creado es básicamente heurístico. En ocasiones detectará dependencias RAW que realmente no lo son, o no detectará otras que sí lo son. Su uso es válido como una referencia pero la comprobación final debe realizarla el usuario.

7 Simulación VLIW

7.1 Configuración de la máquina VLIW

Al igual que con la máquina Superescalar, antes de comenzar una simulación VLIW debe establecerse la configuración de la máquina sobre la que se harán las pruebas. Para ello se puede ir al menú **Configurar** y escoger la opción **Configurar VLIW**, o usar la combinación de teclas “**ALT + F3**”. La ventana de configuración (Ilustración 15) permite modificar el número de unidades funcionales de cada tipo excepto de la de salto, y la latencia o duración de la ejecución en cada unidad. El número de unidades funcionales de cada tipo establece también el formato de la instrucción larga; es por eso que el número de unidades funcionales de salto no puede modificarse, ya que está limitado a una operación de salto por instrucción larga.

El parámetro de los *Fallos de caché* se interpreta igual que en la máquina superescalar.

La modificación de los valores se hace efectiva al hacer *clic* en el botón de **Aceptar**. Con **Cancelar** no se guardan los cambios y se puede volver a los valores por defecto con el botón de **Predefinidos**.

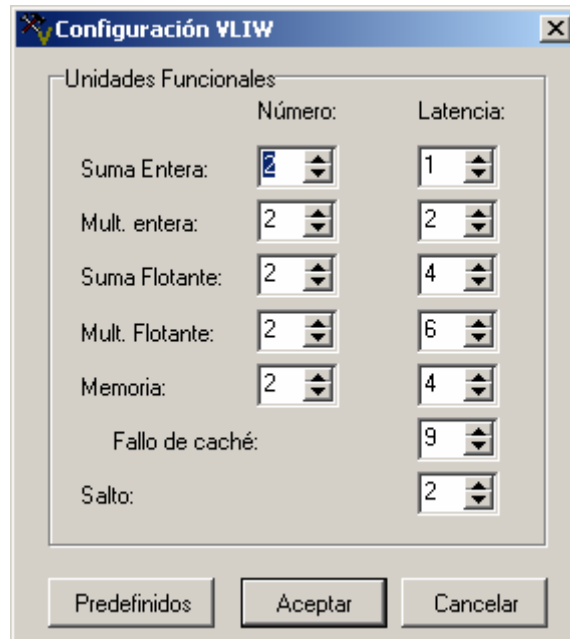


Ilustración 15. Pantalla de configuración de la máquina VLIW

7.2 Ejecución de una simulación VLIW

Para ejecutar una simulación VLIW se debe tener cargado un código secuencial, pero también debe haberse creado un código de instrucciones largas como se vio en la sección 6. Tras seleccionar la máquina VLIW (desde la barra de herramientas de ejecución, desde el menú **Ejecutar** → **Seleccionar Máquina**, o presionando **F3**) puede iniciarse una ejecución continua o “paso a paso” que presentará en pantalla el esquema de la máquina VLIW (Ilustración 16) con los siguientes componentes:

- **NaT (GPR):** Bits de NaT (*Not a Thing*) asociados a los registros de propósito general. Si muestran una *V* es que el registro está siendo usado como destino de un *LOAD* y por tanto su valor no está disponible; una *F* indica que el valor del registro es válido. Se pueden añadir o quitar elementos mediante los botones **+** y **=**, indicándolos como una lista de números (o intervalos) separada por comas (Ej. 2,12-18,20).
- **NaT (FPR):** Bits de NaT (*Not a Thing*) asociados a los registros de punto flotante. Si muestran una *V* es que el registro está siendo usado como destino de un *LOAD* y por tanto su valor no está disponible; una *F* indica que el valor del registro es válido. Se pueden añadir o quitar elementos mediante los botones **+** y **=**, indicándolos como una lista de números (o intervalos) separada por comas (Ej. 2,12-18,20).
- **Reg. Pred.:** Registros de Predicado. Una *V* indica que las operaciones asociadas a este registro se ejecutarán. Con *F* no se ejecutarán. Se pueden añadir o quitar elementos mediante los botones **+** y **=**, indicándolos como una lista de números (o intervalos) separada por comas (Ej. 2,12-18,20).
- **U.F.:** Unidades Funcionales. Son las unidades básicas de ejecución de la máquina. En la parte superior aparece el tipo de UF de que se trata entre suma entera (+Entera), multiplicación entera (xEntera), suma flotante (+Flotante), multiplicación flotante (xFlotante), memoria (Mem) y salto. Por cada tipo de UF hay una tabla con tantas columnas como UF de ese tipo estén declaradas. Las filas representan el *pipeline* de la UF.

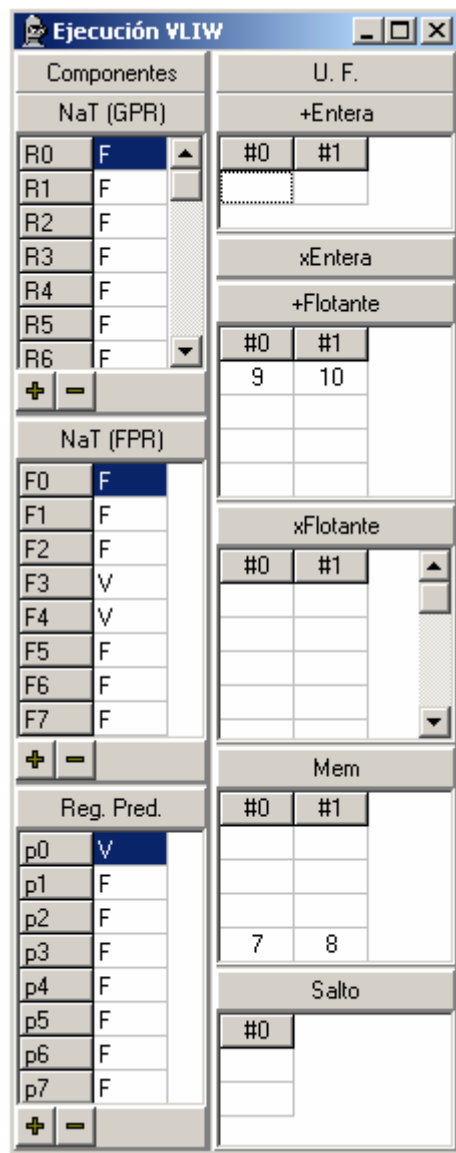


Ilustración 16. Ventana de ejecución VLIW

7.3 Breakpoints

Los *breakpoints* sirven para detener una ejecución continua en un punto del código determinado. En el caso de la máquina VLIW, el *breakpoint* se indica haciendo *dobles clic* sobre el identificador de la instrucción larga en la ventana del código VLIW. El identificador de la instrucción queda marcado en rojo hasta que se vuelva a hacer *dobles clic* en el mismo.

La ejecución se detendrá cuando se intente ejecutar la instrucción marcada, y podrá reiniciarse en modo continuo (▶) o paso a paso (⏏).

7.4 Redimensionamiento de los componentes

El **alto y ancho de la mayoría de los componentes puede modificarse** poniendo el ratón sobre alguno de sus bordes y arrastrando manteniendo presionado el botón izquierdo del ratón. Las unidades funcionales pueden **ocultarse** haciendo *dobles clic* sobre su nombre. Para volver a mostrarla hay que hacer *dobles clic* otra vez.