

# ***CompiladorDLX***

---

*Compilador de ensamblador DLX a código máquina y  
generación de ficheros VHDL para la memoria de  
programa del procesador DLX32p*

---

## **Manual de Usuario**

## Índice

<b>Introducción .....</b>	<b>1</b>
<b>Instalación.....</b>	<b>2</b>
Requisitos Mínimos.....	2
Obtención del Software .....	3
Pasos de la Instalación .....	3
<b>Uso del programa.....</b>	<b>5</b>
Explicación del fichero de sintaxis.....	5
Carga del fichero de sintaxis .....	6
Compilar código ensamblador DLX .....	7
Flujo de trabajo del programa.....	9
Generar o guardar código máquina en formato VHDL .....	11
Otras características .....	11

## Introducción

Este documento pretende explicar el propósito del programa **CompiladorDLX**, el proceso de obtención e instalación del mismo (ver **Instalación**) y su uso, apoyándose en un ejemplo (ver **Uso del programa**).

El **CompiladorDLX** es un compilador, desarrollado en Java, de código ensamblador DLX a código máquina DLX, de acuerdo al repertorio de instrucciones de la arquitectura MIPS32<sup>1</sup>. Puede considerarse, por tanto, como un ensamblador. Sin embargo, no genera ficheros ejecutables para una máquina DLX. En su lugar generará la codificación en 32 bits de cada instrucción en ensamblador, en una línea de un texto plano que se obtiene como primer resultado. Este texto plano se podrá guardar como un fichero en VHDL que se podrá ejecutar o simular en el entorno Xilinx® para el procesador DLX32p, sustituyendo al programa por defecto, contenido en el componente **dlx\_prog2.vhd**.

La versión actual de la aplicación no permite optimizaciones del código ensamblador, si bien se trata de una de las posibles mejoras aplicables a la misma. De hecho, se considera que actualmente el nivel de optimización es el 0.

---

<sup>1</sup> El repertorio de instrucciones de la arquitectura MIPS32 está disponible en la sección “The MIPS32™ Instruction Set” del Volumen 2 de la misma, disponible a través del siguiente enlace:

<http://www.mips.com/content/Documentation/MIPSDocumentation/ProcessorArchitecture/documentationlibrary>.

## Instalación

El **CompiladorDLX** está desarrollado en Java<sup>TM2</sup>, con el J2SE 5.0 update 4<sup>3</sup>, mediante el IDE NetBeans 4.1<sup>4</sup>. Esto obliga a que el equipo del usuario final de la aplicación disponga de la JVM (Java Virtual Machine).

### Requisitos Mínimos

Como la aplicación está desarrollada en Java será bastante portable, de modo que el único requisito es el siguiente:

- JVM (Java Virtual Machine) 1.5 o superior (es equivalente a la 5.0) correctamente instalada en el equipo.

En el caso de un sistema operativo Windows® XP se requiere de los siguientes paquetes de instalación:

- JRE<sup>5</sup> 5.0 update 4 (o superior), disponible en el siguiente enlace: <http://java.sun.com/j2se/1.5.0/download.jsp>, en el que se tendrá que seleccionar y aceptar la licencia para poder realizar la descarga y posterior instalación.

El JDK<sup>6</sup> 5.0 update 4 (o superior) no es necesario, pero podría descargarse para el desarrollo y mantenimiento de la aplicación.

---

<sup>2</sup> La página web para desarrolladores Sun (**Sun Developer Network**), firma que gestiona la tecnología Java, es: <http://java.sun.com/>.

<sup>3</sup> El J2SE (Java 2 Platform Standard Edition) 5.0 update 4 es la 4ª actualización (disponible desde el 23 de junio de 2005) de la versión 5.0 (internamente es la versión 1.5), de la plataforma de desarrollo Java.

<sup>4</sup> El IDE (Integrated Development Environment) NetBeans está disponible desde la página web: [www.netbeans.org](http://www.netbeans.org). Es un IDE gratuito para uso no comercial que va por la versión 4.1.

<sup>5</sup> El JRE (J2SE Runtime Environment) permite solamente la ejecución de aplicaciones Java, sobre la JVM.

<sup>6</sup> El JDK (J2SE Development Kit) permite el desarrollo de aplicaciones Java.

## Obtención del Software

El **CompiladorDLX** se distribuye en un fichero comprimido en formato **rar**, bajo el nombre **CompiladorDLX.rar**. Este paquete consta de los siguientes ficheros:

- **Compilador.jar** → Paquete en formato **jar** con el programa **CompiladorDLX** ejecutable en la JVM.
- **Sintaxis.txt** → Fichero de sintaxis para el repertorio de instrucciones de la arquitectura MIPS32, que permite la edición de la misma.
- **autojar.reg** → Modificador del registro de Windows® XP que permite que los ficheros con extensión **jar** se ejecuten con la aplicación **java** (intérprete de Java, para la JVM).
- **Ejemplos/** → Carpeta con ejemplos de programas en código ensamblador DLX compilados a código máquina DLX (disponibles en formato VHDL para el procesador DLX32p).
  - **Fibonacci.txt** → Algoritmo del Fibonacci recursivo en ensamblador DLX.
  - **Fibonacci.vhd** → Algoritmo del Fibonacci recursivo en código máquina DLX, en formato VHDL para el procesador DLX32p.

Es posible obtenerlo de la web de la asignatura de Arquitectura de Computadores<sup>7</sup> o bien haciendo clic **aquí**.

## Pasos de la Instalación

Partiremos de un equipo con la JVM correctamente instalada y con una versión compatible con el **CompiladorDLX**. Una vez descargado el paquete **CompiladorDLX.rar** los pasos a realizar para la instalación son los siguientes:

1. Descomprimir **CompiladorDLX.rar** con WinRAR<sup>8</sup>.

---

<sup>7</sup> La página web de la asignatura Arquitectura de Computadores tiene el **CompiladorDLX** en la sección de prácticas, accesible desde el siguiente enlace: <http://serdis.dis.upgc.es/~ii-ao/practicas.htm>.

2. Ejecutar la aplicación **CompiladorDLX.jar**, que puede realizarse desde un terminal en modo texto o desde la interfaz gráfica de Windows®.

- a. **Terminal en modo texto.** Desde el intérprete de comandos de Windows (**cmd**) o desde el terminal de Linux, lanzaremos el programa según la sintaxis de Java, es decir, ejecutaremos el siguiente comando:

```
java -jar Compilador.jar
```

- b. **Interfaz gráfica de Windows®.** Desde el explorador Windows es posible asociar la ejecución de los ficheros con extensión **jar** con el intérprete de Java, que es el programa **java.exe** (que aparecerá como **Java(TM) 2 Platform Standard Edition binary**). Para facilitar esta tarea la distribución contiene un fichero que realiza automáticamente este proceso. Este fichero es **autojar.reg**, que se ejecutará haciendo doble clic sobre el mismo y respondiendo **Sí** a la pregunta que aparecerá, para que modifique el registro de Windows, realizando la asociación antes comentada. Luego basta picar en **Aceptar**.

---

<sup>8</sup> La página oficial del compresor WinRAR en castellano se encuentra en <http://winrar.com.es/>. Desde la sección de descargas pueden obtenerse los binarios para distintas plataformas o sistemas operativos.

## Uso del programa

Partiremos del **CompiladorDLX** abierto, lo cual se podrá haber conseguido por alguno de los métodos explicados previamente (ver *Pasos de Instalación*). Para poder empezar a usar el compilador, es necesario cargar un fichero de sintaxis. En la distribución se suministra un fichero de sintaxis por defecto, con el nombre *Sintaxis.txt*.

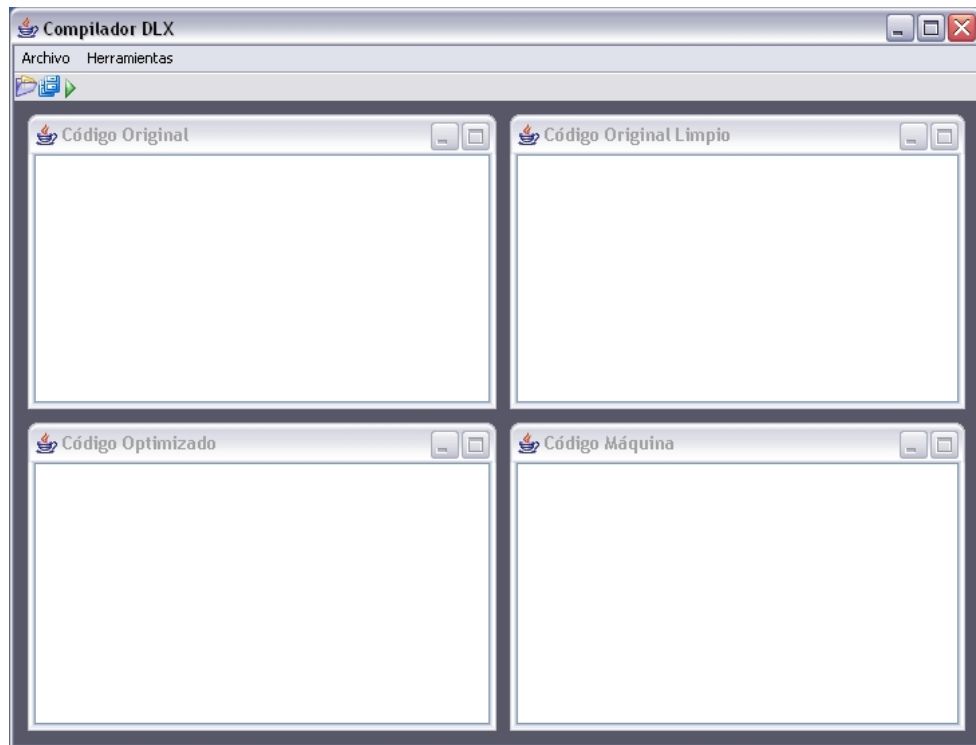


Ilustración 1: Aspecto del CompiladorDLX abierto (en Windows® XP)

### Explicación del fichero de sintaxis

El fichero de sintaxis contiene una serie de etiquetas que delimitan secciones de información que usará el **CompiladorDLX** para compilar de ensamblador DLX a código máquina DLX y poder generar los ficheros en formato VHDL para su ejecución o simulación en el procesador DLX32p. En la primera sección (<COMENTARIO> ... </COMENTARIO>) se explica el resto de las secciones, que fundamentalmente indican el formato del fichero VHDL y finalmente las instrucciones del repertorio de la arquitectura MIPS32, junto con un indicador del tipo de operación y el código de operación (o de la ALU) asociado.

Este fichero puede modificarse para que el fichero VHDL tenga una nueva estructura, o bien para ampliar o modificar el repertorio de instrucciones o modificar el código de operación asociado.

### *Carga del fichero de sintaxis*

Los pasos para la carga de un fichero de sintaxis serán los siguientes:

1. Ir al menú **Herramientas** y picar en el submenú **Opciones**.



Ilustración 2: Abrir Cuadro de Diálogo de Opciones

Se abrirá el cuadro de diálogo de **Opciones** y nos situaremos en la pestaña de **Sintaxis**.

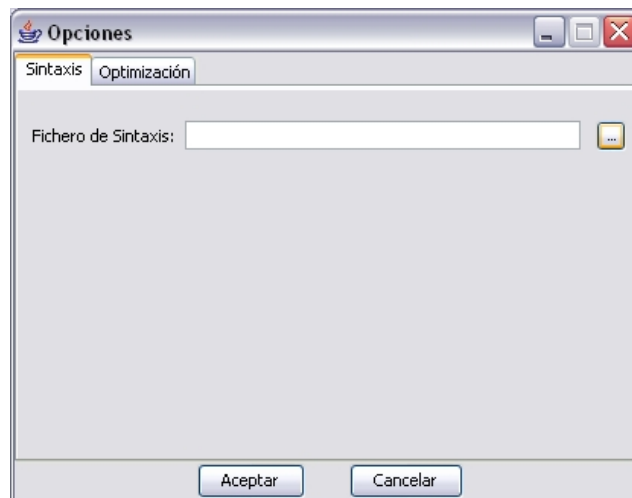



Ilustración 3: Cuadro de Diálogo de Opciones

2. Buscar el fichero de sintaxis deseado y abrirlo, a través del cuadro de diálogo de apertura de ficheros. Para abrir el cuadro de diálogo de apertura de ficheros bastará picar en el botón .



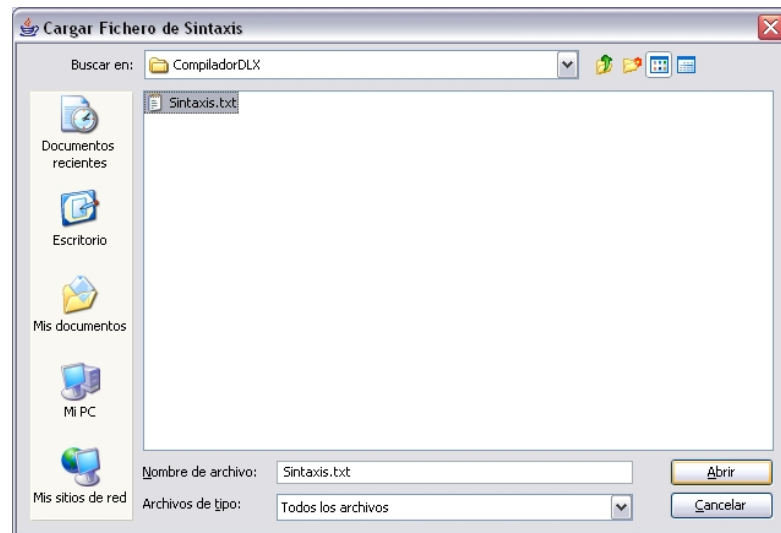


Ilustración 4: Cuadro de Diálogo de Apertura de Ficheros. Abrir fichero de sintaxis

3. **Aceptar** el fichero de sintaxis seleccionado; o bien **Cancelar**.

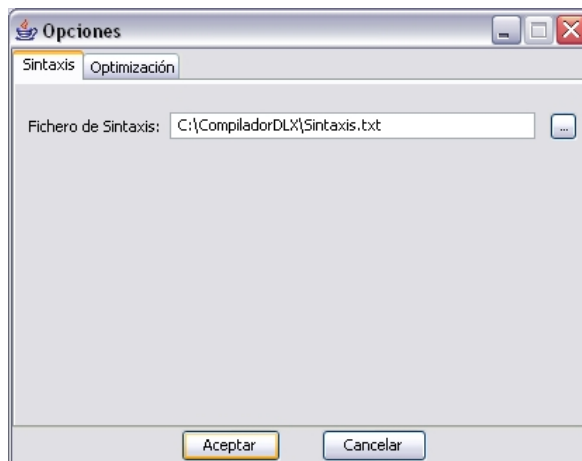



Ilustración 5: Cuadro de Diálogo de Opciones. Aceptar fichero de sintaxis a usar

## Compilar código ensamblador DLX

El primer paso para compilar un programa en ensamblador DLX consiste en obtener el código de dicho programa. Este código se pondrá en la ventana interna *Código Original*. Existen dos formas de introducir dicho código:

1. **Escribiéndolo directamente.** Este es el caso más simple, pues basta con escribirlo en la ventana *Código Original* directamente.
2. **Abriendo un fichero.** Para abrir el fichero basta picar en el botón , o bien en el menú **Archivo** y picar en el submenú **Abrir**. De este modo se abrirá un cuadro de diálogo de apertura de ficheros en el que seleccionaremos el fichero con código ensamblador DLX a compilar.

En la distribución se dispone del algoritmo del Fibonacci recursivo, en la carpeta *Ejemplos*, con el nombre *Fibonacci.txt*, que se muestra en la siguiente ilustración.

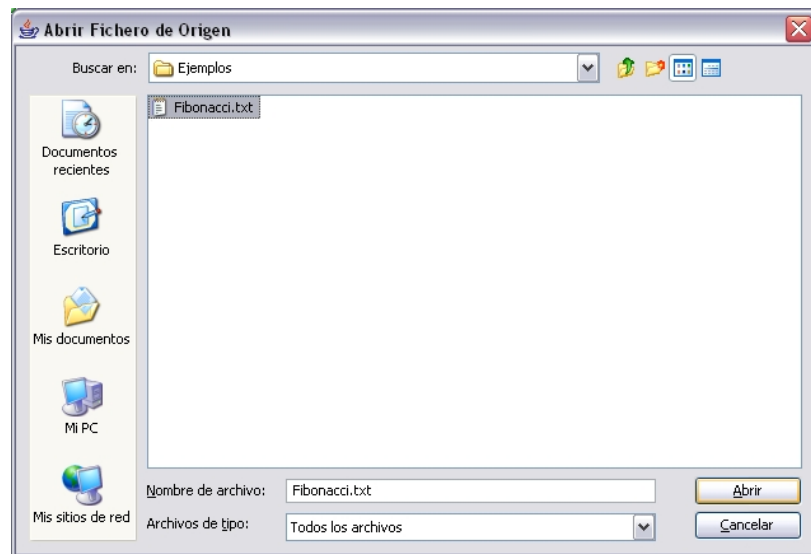


Ilustración 6: Cuadro de Diálogo de apertura de fichero con código ensamblador DLX

Una vez abierto el fichero, se tendrá su contenido en la ventana *Código Original*.

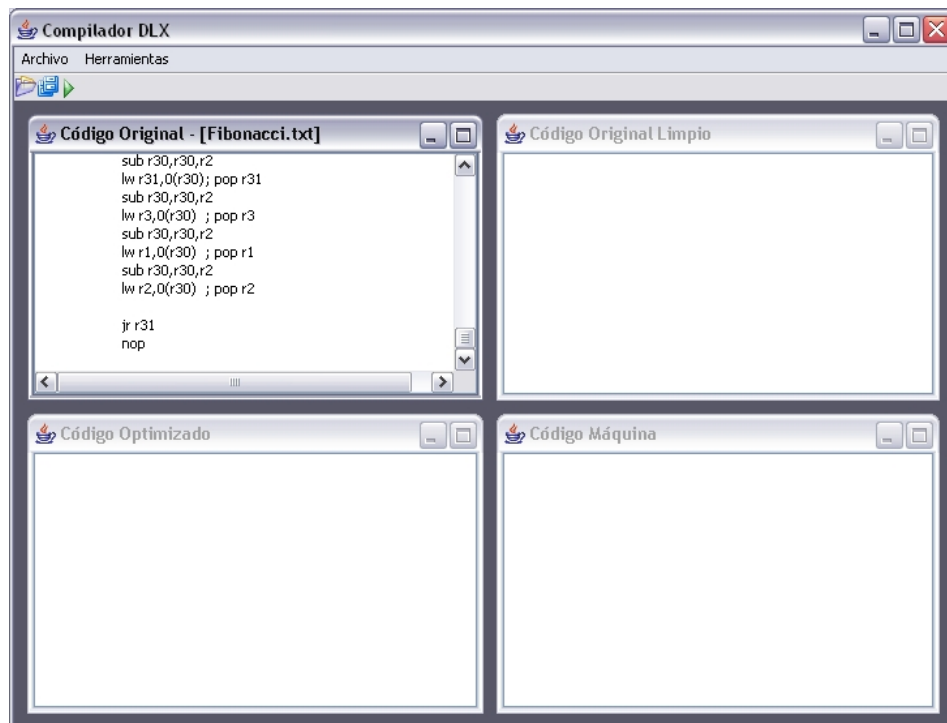



Ilustración 7: CompiladorDLX con código ensamblador DLX en la ventana Código Original - [Fibonacci.txt]

A continuación es posible compilarlo picando en el botón , o bien en el menú **Herramientas** picando en el submenú **Compilar**. Tras compilar se obtendrá el código máquina en la ventana **Código Máquina**, a parte de que las ventanas **Código Original Limpio** y **Código Optimizado** tendrá el código limpio de comentarios e identaciones extra y el código optimizado (en esta versión no se aplica ninguna optimización), respectivamente.

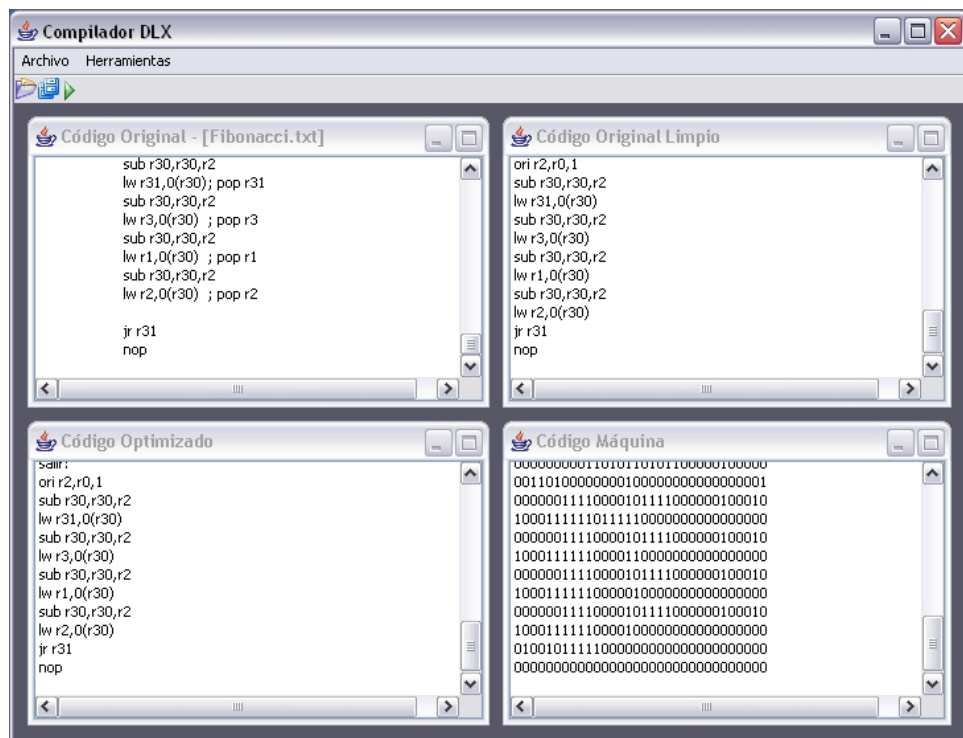


Ilustración 8: CompiladorDLX con código ensamblador DLX compilado a código máquina

### Flujo de trabajo del programa

El flujo de trabajo del programa, que se produce al picar en el botón de compilar, se puede ver gráficamente en la siguiente ilustración.

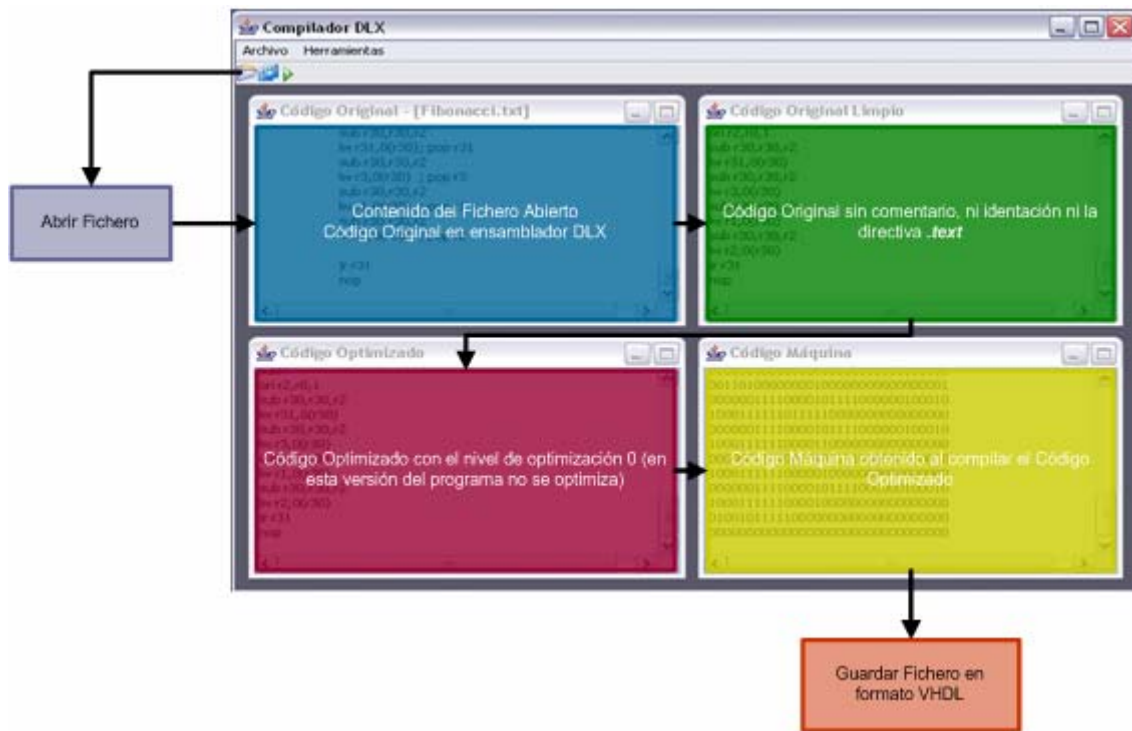


Ilustración 9: Flujo de trabajo del CompiladorDLX


Como puede verse, a partir del código ensamblador DLX original (de la ventana **Código Original**), que incluirá indentación de código y comentarios (además de la sección **.text**; esta versión no admite la sección **.data**), se obtiene el mismo código pero limpio, es decir, sin comentarios, ni indentación ni el indicador **.text**. Este código limpio se tendrá en la ventana **Código Original Limpio**, y su finalidad es el apoyo al desarrollo de la aplicación, como mecanismo de depuración y detección de errores.

A continuación, se realiza el trabajo útil del compilador, que se dividirá en una fase inicial de optimización y una final de compilación o generación de código máquina. La optimización dependerá del nivel de optimización. En esta versión no se optimiza, es decir, el nivel de optimización es el 0, de modo que en la ventana **Código Optimizado** se tiene el mismo código que en la ventana **Código Original Limpio**.

Finalmente se compila el código de la ventana **Código Optimizado** y se obtiene el código máquina, en formato de instrucciones de 32 bits, en la ventana **Código Máquina**.

## Generar o guardar código máquina en formato VHDL

Para guardar el código máquina obtenido tras la compilación, en formato VHDL para ejecutarlo o simularlo con el procesador DLX32p, hay que seguir los siguientes pasos:

1. Picar en el botón , o bien ir al menú **Archivo** y picar en el submenú **Guardar Código Máquina como ....**
2. Se abre un cuadro de diálogo para guardar el fichero en el que hay que introducir el nombre deseado, con extensión **vhd**, que es la usado por Xilinx® para el formato VHDL.

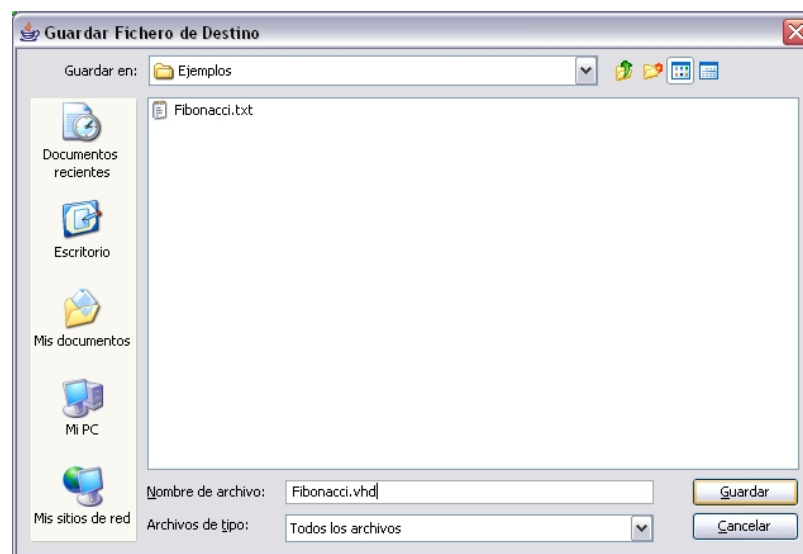


Ilustración 10: Cuadro de Diálogo para guardar el fichero en formato VHDL

Tras guardar el archivo, a través del explorador de Windows podrá verse, de la siguiente forma:

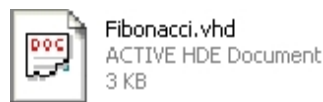


Ilustración 11: Fichero en formato VHDL

## Otras características

El **CompiladorDLX** ofrece otras posibilidades, como es la capacidad de copiar el contenido de las 4 ventanas y la edición del mismo. Como ya se ha comentado, también se permite indicar el nivel de optimización, a través del cuadro

de diálogo de **Opciones** en la pestaña **Optimización**, pero esta versión sólo admite el nivel 0, es decir, no se optimiza.

Por otro lado, también es posible guardar el código ensamblador DLX que se tiene en la ventana **Código Original**, permitiendo así que se guarden las modificaciones realizadas o el código escrito directamente. Para poder hacer esto hay que seguir los siguientes pasos:

1. Picar en el menú **Herramientas** y en el submenú **Guardar Código Original como ....**
2. Se abrirá un cuadro de diálogo para guardar el fichero, en el que bastará con indicar su nombre.