LABORATORIO VIRTUAL DE SISTEMAS DIGITALES

Memoria del Proyecto Final de Carrera
de Ingeniería en Informática
realizado por
Marcos Fernández Callejo
y dirigido por
Marta Prim Sabrià
Bellaterra, 15 de Junio de 2007

TABLA DE CONTENIDOS

1. INTRODUCCIÓN 9

1.2. ESTRUCTURA DE LA MEMORIA 10

2. ESTUDIO DE VIABILIDAD 12

2.1. INTRODUCCIÓN 12

2.2. OBJETO 13
2.2.1. Descripción a tratar 13
2.2.2. Perfil de usuario 13
2.2.3. Objetivos 13
2.2.4. Estado del arte 13

2.3. DESCRIPCIÓN DEL SISTEMA A REALIZAR 14
2.3.1. Descripción 14
2.3.2. Recursos 15
2.3.3. Evaluación de riesgos 15
2.3.4. Organización del proyecto 15

2.4. REQUERIMIENTOS FUNCIONALES Y NO FUNCIONALES 16
2.4.1. Requerimientos funcionales 16
2.4.2. Requerimientos no funcionales 16

2.5. PLANIFICACIÓN DEL PROYECTO 17

2.5.1. Tareas que componen el proyecto 17
2.5.1.1. Recopilación de conocimientos básicos necesarios 17
2.5.1.2. Diseño de la aplicación 18
2.5.1.3. Pruebas 18
2.5.1.4. Documentación 18

2.5.2. Tareas a realizar a lo largo del diseño de la aplicación 18
2.5.2.1. Diseño de la interfaz de usuario 18
2.5.2.2. Gestor de Cables 18
2.5.2.3. Gestor de Chips 18
2.5.2.4. Política de simulación 18
2.5.2.5. Integración del generador de ondas 19
2.5.2.6. Integración del Osciloscopio 19

2.5.3. Planificación temporal 19
2.5.3.1 Distribución temporal 20

2.6. VIABILIDAD TÉCNICA DEL PROYECTO 23

2.7. HERRAMIENTAS DE DESARROLLO 23

2.8. ANÁLISIS DE VIABILIDAD ECONÓMICA 23

2.8.1. Supuestos costes **23**

2.9. VIABILIDAD LEGAL **24**

2.10. CONCLUSIONES ESTUDIO DE VIABILIDAD **24**

3. FUNDAMENTOS TEÓRICOS **27**

3.1. SIMULADOR DEL LABORATORIO DE SISTEMAS DIGITALES **27**

3.2. PLACA **27**

3.3. CHIPS **29**
3.3.1. Tipología de Chips **29**

3.4. CABLES **33**

3.5. COMPONENTES DE ENTRADA **34**
3.5.1. Interruptores **34**
3.5.2. Fuente de alimentación **35**
3.5.3. Tierra (GND) **36**
3.5.4. Generador de Ondas **37**

3.6. COMPONENTES DE SALIDA **39**
3.6.1. Diodos emisores de Luz, Leds **39**
3.6.2. 7-segmentos **40**
3.6.3. Osciloscopio **42**

3.7. CONCLUSIONES FUNDAMENTOS TEÓRICOS **43**

4. ANÁLISIS DE LA APLICACIÓN **46**

4.1. ESTILO DE PROGRAMACIÓN **46**
4.1.1. Elaboración de identificadores **46**
4.1.1.2. Uso de mayúsculas 46
4.1.1.3. Formato de los bloques de código 46

4.2. ESTRUCTURA DEL PROGRAMA **47**
4.2.1. Pantalla principal **47**
4.2.1.1. Gestión de cables 48
4.2.1.1. Gestión de chips 48

4.3. POLÍTICA DE GESTIÓN DE CONEXIONES **49**

4.4. POLÍTICA DE SIMULACIÓN **50**

4.5. DESCRIPCIÓN DE LAS CLASES Y DE LOS PRINCIPALES MÉTODOS **51**
4.5.1. Las clases **51**
4.5.1.1 La clase Ariston 52

4.5.1.2 La clase AristonMain 52
4.5.1.3 La clase PlacaAriston 52
4.5.1.4 La clase PlacaAristonPaint 53
4.5.1.5 Las clase MouseDibuixarCableEstatic i MouseDibuixarCableDinamic 53
4.5.1.6 La clase Parcela 54
4.5.1.7 La clase ControlConnexionsCable 55
4.5.1.8 La clase MicroChip 56
4.5.1.9 La clase GeneradorOnes 57
4.5.1.10 La clase OsciloscopiMain 57
4.5.1.11 La clase PanelSimulacio 58

4.5.2. Los principales métodos 58
4.5.2.1 connectarValorsParceles() 58
4.5.2.2 construirArbresConnexions() 59
4.5.2.3 eliminarConnexio () 59

4.6. CONCLUSIONES DEL ANÁLISIS DE LA APLICACIÓN 59

5. DISEÑO 62

5.1. DISEÑO DE LA APLICACIÓN 62
5.1.1. Pantalla principal 62
5.1.2. Ventanas secundarias 63
5.1.3. Generador de Ondas 66
5.1.4. Osciloscopio 66

5.2. CONCLUSIONES DEL DISEÑO 67

6. MANUAL DE USUARIO 69

6.1 EJECUCIÓN APPLET JAVA 69

6.2 MENÚS 69

6.3 NAVEGACIÓN PANTALLA PRINCIPAL 70
6.3.1. Creación y manipulación de cables 70
6.3.2. Creación y manipulación de chips 71

6.4. GENERADOR DE ONDAS 73

6.5. SIMULACIÓN 74

6.6. OSCILOSCOPIO 75

6.7. LEDS Y 7-SEGMENTOS 77

6.8. CONCLUSIONES MANUAL DE FUNCIONAMIENTO 78

7. INTEGRACIÓN Y TEST 80

7.1. PRUEBA DE LA CREACIÓN Y MANIPULACIÓN DE CABLES 80

7.2. PRUEBA DEL FUNCIONAMIENTO DE LOS CHIPS..... 81

7.3. PRUEBA DE LAS CONEXIONES ENTRE LOS DIFERENTES COMPONENTES DEL SIMULADOR 81

7.4. PRUEBA DEL FUNCIONAMIENTO DE LOS COMPONENTES DE ENTRADA 82

7.5. PRUEBA DEL FUNCIONAMIENTO DE LOS COMPONENTES DE SALIDA 82

7.6. CONCLUSIONES INTEGRACIÓN Y TEST 82

8. CONCLUSIONES 85

8.1. APORTACIONES DEL TRABAJO DESARROLLADO 85

8.2. REPASO DE LOS OBJETIVOS PROPUESTOS 86

8.3. ASPECTOS ORIGINALES Y CRATIVOS DEL PROYECTO 87

8.4. OTRAS CONCLUSIONES DE CARÁCTER GENERAL 87

8.5. ASPECTOS NEGATIVOS 87

8.6. LÍNEAS ABIERTAS 88

9. BIBLIOGRAFÍA 90

9.1 LIBROS 90
9.1.1 Fundamentos teóricos 90
9.1.2 Aplicación Informática 90

9.2 INTERNET 90
9.2.1. Fundamentos teóricos 90
9.2.2. Aplicación informática 90
9.2.3. Estado del arte 90

CAPÍTULO 1:

INTRODUCCIÓN

1. INTRODUCCIÓN

Laboratorio virtual de sistemas digitales, es un portal web que ejecuta un applet, o aplicación web, cuyo objetivo es simular un laboratorio de prácticas de la asignatura de sistemas digitales de la titulación de ingeniería técnica informática.

Varias son las motivaciones que han conllevado el desarrollo de este proyecto. La primera es la de construir una herramienta útil para aquellos estudiantes y profesores de la materia de sistemas digitales. Útil por su sencillo funcionamiento, por ser accesible desde la red sin necesidad de ningún tipo de instalación y sobre todo, teniendo en cuenta que los estudiantes tienen limitado el uso de los laboratorios, por facilitar la configuración y prueba de los diseños realizados por los alumnos de prácticas de forma ilimitada.

Una motivación importante a la hora de implementar el proyecto es la de integrar dos módulos previamente desarrollados, osciloscopio y generador de ondas, con un tercer módulo que incluye placa, chips, leds y 7-segmentos. La interconexión y funcionamiento conjunto de los tres módulos forma el laboratorio virtual de prácticas y da sentido al desarrollo de cada módulo.

Otra de las motivaciones es la de poner en práctica los conocimientos adquiridos durante el estudio de la ingeniería informática. Sobre todo en los campos de programación orientada a objetos y sistemas digitales. En lo referente a la programación orientada a objetos la motivación aún ha sido mayor, porque el desarrollo del proyecto ha permitido adquirir y poner en práctica numerosos conocimientos del lenguaje de programación Java. Este lenguaje es uno de los más populares y utilizados en la actualidad, es muy importante para un ingeniero informático tener unas buenas nociones sobre Java.

Por último, y seguramente la más importante, desarrollar un proyecto válido para obtener el título de Ingeniero en Informática.

El proyecto se ha organizado a través de una serie de tareas formadas a partir de los objetivos propuestos. Dichas tareas se han repartido a lo largo de nueve meses de acuerdo con su dificultad y los recursos disponibles (entre ellos el propio proyectista). En el capítulo 2, “Estudio de viabilidad”, se desarrollan todos los aspectos de la estructuración del proyecto.

El elemento clave del proyecto es el applet que simula el laboratorio virtual de prácticas. Un applet es una aplicación codificada en Java que se ejecuta desde un navegador, un usuario accede a una determinada página, el navegador baja el código fuente del applet y lo ejecuta mediante la maquina virtual de Java. De esta manera un applet puede funcionar como cualquier aplicación sin necesidad de instalación y sobre cualquier sistema operativo.

1.2. ESTRUCTURA DE LA MEMORIA

La memoria del proyecto muestra los aspectos que han incidido en la planificación del mismo. En ella se separa la parte conceptual y teórica en que se inspira de la parte informática que lo implementa.

La memoria está dividida en las siguientes partes:

- **Estudio de viabilidad:** Se analiza la viabilidad del proyecto desde tres vertientes, la técnica, la económica y la legal. También se especifica la planificación del proyecto, los requerimientos funcionales y no funcionales y las herramientas de desarrollo.

- **Fundamentos teóricos:** Este capítulo está dedicado a los conceptos teóricos del proyecto. En concreto, se explican los elementos que componen un laboratorio de prácticas, su comportamiento y sus características propias.

- **Análisis de la aplicación:** Se detalla todo el proceso de codificación del proyecto, el estilo de programación y las clases en las que está estructurada la aplicación.

- **Diseño:** Tema que trata las soluciones que se han aplicado para lograr el aspecto final del proyecto.

- **Manual de funcionamiento:** En esta parte de la memoria se explica como debe utilizarse la aplicación para que el usuario pueda sacarle el máximo rendimiento.

- **Integración y test:** Se especifican las pruebas que se han llevado a cabo para asegurar un buen comportamiento de la aplicación.

- **Conclusiones:** Se comenta lo que ha aportado el proyecto a nivel personal, se repasan los objetivos propuestos, se destacan los aspectos originales, creativos y negativos del trabajo. También se proponen futuras mejoras y ampliaciones que podrían aplicarse al proyecto.

También se incluye un apartado bibliográfico que especifica las fuentes consultadas a lo largo de la construcción del proyecto.

Nota: A lo largo de la memoria se utilizarán diversas denominaciones sinónimas para la designación del proyecto, tales como: laboratorio virtual de sistemas digitales, simulador, programa, aplicación o aplicación web.

CAPÍTULO 2:

ESTUDIO DE VIABILIDAD

2. ESTUDIO DE VIABILIDAD

Los siguientes apartados del capítulo tratan los requerimientos a cumplir para que el proyecto sea viable. Para que el proyecto se pueda llevar a cabo tiene que existir simultáneamente una viabilidad técnica, económica y legal, puntos que se van a considerar a continuación.

2.1. INTRODUCCIÓN

El objetivo del proyecto es construir una aplicación web que simule el funcionamiento de un laboratorio de prácticas de la materia de sistemas digitales.

Los alumnos de sistemas digitales tienen limitado el acceso a los laboratorios de prácticas, esto implica que tienen que desarrollar la práctica que previamente han diseñado durante la sesión de laboratorio semanal. El tiempo destinado a esa sesión puede ser insuficiente a la hora de implementar físicamente el diseño previo realizado por el alumnado, detectar los problemas que causan el mal funcionamiento de un circuito suele ser muy costoso en tiempo debido a las muchas variables que participan en el proceso. La suma de la limitación temporal y la complejidad de encontrar errores en un circuito electrónico puede provocar la imposibilidad de completar correctamente las prácticas o el alargamiento excesivo de las sesiones semanales.

El profesorado de los laboratorios de prácticas de sistemas digitales tiene un tiempo limitado para evaluar los circuitos implementados por los estudiantes durante la sesión semanal de prácticas. Si a la complejidad temporal de encontrar errores en un circuito electrónico le multiplicamos el número de estudiantes asistentes en una sesión, el resultado puede provocar en el profesorado un alargamiento mucho más excesivo de las sesiones semanales que el que podrían sufrir los alumnos.

El proyecto pretende ser una herramienta útil para el funcionamiento de las prácticas de la asignatura. La posibilidad de que el alumnado pueda implementar virtualmente tantas veces como desee su diseño previo, conseguirá limitar el número de probabilidades de un mal comportamiento en la implementación física. Esto conllevará una reducción de los tiempos de implementación y evaluación de los circuitos digitales mitigando la principal problemática de la parte práctica de la materia.

Otro de los beneficios del proyecto es que el estudiante se familiarizará rápidamente con los componentes que se encontrará en el laboratorio físico, facilitando el aprendizaje de su uso y comportamiento. Esto agilizará el proceso de prácticas de la asignatura.

Las características propias de la aplicación implican una serie de ventajas, la primera es la no necesidad de instalación al ejecutarse desde el navegador, y la segunda el libre acceso a un recurso gratuito y abierto a todo aquel que disponga de conexión a Internet.

Una vez analizada la problemática de la asignatura y los beneficios que puede aportar el desarrollo del proyecto se puede afirmar que la aplicación será un elemento útil para el aprendizaje y funcionamiento de la materia.

2.2. OBJETO

2.2.1. Descripción a tratar

Se pretende que el usuario pueda construir virtualmente sobre una placa un circuito digital mediante una serie de chips, cables y elementos de entrada y salida interconectados entre ellos. El comportamiento del circuito virtual debe imitar al comportamiento del mismo circuito en su versión física.

2.2.2. Perfil de usuario

El usuario será toda aquella persona que disponga de ordenador y conexión a Internet y que tenga algún interés en el diseño, construcción y manipulación de circuitos digitales; en especial, alumnos de asignaturas como sistemas digitales, electrónica digital, fundamentos de computación,…

2.2.3. Objetivos

Los objetivos de la aplicación son:

- Ser una herramienta útil para el aprendizaje y manipulación de los componentes que se encuentran en un laboratorio de circuitos digitales.

- Ser una aplicación que permita simular el comportamiento de circuitos digitales de una forma ilimitada.

- Ser un recurso gratuito, de fácil acceso, multiplataforma y de instalación sencilla.

2.2.4. Estado del arte

Actualmente sólo existe en la red un applet simulador de circuitos digitales, hay alguna aplicación de uso limitado y diversos programas potentes y complejos de carácter industrial y de pago. No se ha encontrado ninguna aplicación web, o aplicación común, que integre un osciloscopio, generador de ondas, placa, cables y chips. A continuación se detallan algunas de las características de simuladores ya implementados.

- http://www.cs.york.ac.uk/netpro/bboard/

Desde esta dirección URL accedemos a un applet desarrollado en el departamento de ciencias de la University of York (Heslington, York, Reino Unido). El applet permite construir circuitos digitales sobre diversas placas, añadir Leds, chips y cables. La construcción de los cables es compleja y no existen otros elementos de salida como el 7-segmentos o el osciloscopio.

- http://www.yoeric.com/breadboard.htm

Yoeric Corporation es una empresa privada del estado de Carolina del Norte en EEUU que ha desarrollado una aplicación Windows que permite construir un circuito digital sobre una placa, añadiendo chips y cables, y pudiendo utilizar leds y 7-segmentos. La construcción de los cables es sencilla y cómoda, en la web de la empresa afirman que la aplicación es utilizada por un elevado número de universidades en todo el mundo. Su versión de prueba permite un uso limitado de los recursos del programa, la licencia para su uso completo cuesta 49 $ y sólo funciona bajo plataformas windows. Carece de generador de ondas y osciloscopio.

Las siguientes aplicaciones además de no ser un recurso gratuito y on-line, son extremadamente complejas y fuera de los objetivos del presente proyecto. No obstante merece la pena su mención por su popularidad y uso.

- http://www.electronicsworkbench.com/index.html

National Instruments Corporation es otra empresa privada del estado de Texas en EEUU que tiene una amplia gama de software relacionado con el diseño de circuitos digitales, sus aplicaciones se caracterizan por ser potentes y complejas. Las licencias software son de pago y sus precios oscilan entre los 1.500 $ y 20.000 $.

- http://www.cadence.com/products/orcad/index.aspx?lid=orcad

Orcad/Pspice es una familia de software relacionado con el diseño de circuitos digitales, aplicaciones desarrolladas por la empresa privada Cadence del estado de California en EEUU. Su software también es muy potente y complejo en su uso, existen versiones demo pero para poder utilizar todos los recursos hay que pagar licencias software. Las aplicaciones están más enfocadas al mundo profesional que el educativo.

- http://www.logicworks4.com/

Capilano computing es una empresa privada del estado de Vancouver en Canada. Desarrolla aplicaciones muy potentes como Logic Works, programa dedicado a la implementación y simulación de circuitos digitales. Su uso es muy habitual entre los estudiantes de ingeniería electrónica.

2.3. DESCRIPCIÓN DEL SISTEMA A REALIZAR

2.3.1. Descripción

Se implementará un módulo compuesto por una placa (similar a las que se puede encontrar en el laboratorio de sistemas digitales), 8 leds, 2 7-segmentos y 12 interruptores. El módulo tendrá la capacidad de añadir chips a la placa y cables que interconecten todos los componentes. Se integrará a este módulo dos aplicaciones ya desarrolladas, el generador de ondas y el osciloscopio.

2.3.2. Recursos

A parte de los recursos obvios como es una conexión a Internet y la disponibilidad de ordenador, el más importante es tener un navegador con intérprete de Java, independientemente del sistema operativo ya sea Windows, Unix o Linux. Actualmente los navegadores más populares, Explorer y Mozilla, tienen un intérprete de Java ya instalado.

La empresa Sun, desarrollador del lenguaje Java, distribuye libremente el software necesario para elaborar y ejecutar applets y aplicaciones Java.

2.3.3. Evaluación de riesgos

En el proceso físico de construcción de un circuito digital participan un elevado número de variables, existe el riesgo de no poderlas contemplar todas a la hora de implementar la simulación virtual.

La colocación física de cables que interconectan los elementos que componen un laboratorio virtual de sistemas digitales es trivial pero la simulación de ese proceso es bastante compleja computacionalmente.

Controlar el flujo de ejecución de la simulación exige de una política de gestión temporal de eventos, si dicha política no establece un orden correcto a la hora de evaluar los diferentes eventos que se producen en un circuito digital la simulación no será correcta.

La integración de los módulos ya existentes, generador de ondas y osciloscopio, no es trivial. Serán necesarios diversos procesos que permitan la interconexión y comprensión entre los tres módulos.

Se presupone que Java es un lenguaje lo suficientemente potente como para poder desarrollar el laboratorio virtual de prácticas, existe el riesgo de que pueda surgir alguna circunstancia que no se pueda resolver mediante las herramientas que proporciona Java.

2.3.4. Organización del proyecto

La estructura del proyecto deberá ser lo suficientemente clara para que otros programadores puedan realizar modificaciones y mejoras en el futuro. El código debe de ser entendible.

Java al ser orientado a objetos permite que se puedan realizar modificaciones y actualizaciones sin que eso suponga algo traumático. Su extensa librería de clases y funciones accesibles desde la red permite la adaptación a cualquier navegador. Siguiendo un mínimo de reglas sobre el código escrito, más la característica de Java de ser un lenguaje de alto nivel, debe permitir una escritura clara del código.

2.4. REQUERIMIENTOS FUNCIONALES Y NO FUNCIONALES

Los requerimientos son un conjunto de ideas sobre como desarrollar el software, los requerimientos pueden ser funcionales o no funcionales. Los funcionales describen que funcionalidad va a dar el software al usuario y los no funcionales determinan el funcionamiento del software.

2.4.1. Requerimientos funcionales

- **Placa:** La aplicación tiene que implementar una placa y permitir que el usuario pueda añadir y eliminar chips y cables.

- **Chips:** El usuario podrá insertar los chips que se utilizan habitualmente en las prácticas de la asignatura de sistemas digitales. La aplicación facilitará el esquema de cada chip para que el usuario pueda conocer el funcionamiento interno.

- **Cables:** El programa debe facilitar al usuario un sistema cómodo de construcción y eliminación de cables.

- **Componentes de entrada:** El programa facilitará los componentes habituales en un laboratorio para generar un flujo eléctrico que circule por el circuito diseñado. Estos componentes serán, una fuente de alimentación, una toma a tierra (GND) e interruptores que generarán señales a 1 o 0 según la voluntad del usuario. Así mismo se integrará el módulo ya implementado, generador de ondas, que produce una señal cuadrada de unos y ceros.

- **Componentes de salida:** La aplicación también implementará los componentes de salida más comunes de un laboratorio de circuitos digitales. Estos componentes son los leds y 7-segmentos. También se integrará el módulo ya terminado, osciloscopio, que permite visualizar las señales digitales que genera un circuito.

2.4.2. Requerimientos no funcionales

- **Sistema de menús:** El usuario podrá encontrar en la barra de menús algunas de las acciones más importantes de la aplicación, como es la de insertar un chip, eliminar todos los chips y eliminar todos los cables.

- **Botón directo de inserción de chips:** Insertar chips en la placa es una de la operaciones más habituales cuando se construye un circuito digital, para facilitar esta operación se añadirá un botón en la pantalla principal

- **Acceso al generador de ondas:** La aplicación debe incorporar un botón desde el cual se puede visualizar el generador de ondas, también debe de incorporar un puerto que facilite la conexión entre el generador y los elementos insertados en la placa.

- **Acceso al osciloscopio:** De la misma manera que en el generador, la aplicación debe tener un botón que permita visualizar el generador de ondas y un puerto para conectarlo con los elementos configurados en la placa.

- **Otros componentes de entrada y salida:** Todos los componentes de entrada y salida implementarán un puerto que los interconecte con la placa y los elementos insertados.

- **Botón de simulación:** La aplicación debe incluir un botón que permita al usuario iniciar y parar la simulación una vez ha diseñado el circuito digital.

- **Interfície de usuario:** El applet constará de una ventana principal donde se visualice la placa en su zona central, en las zonas periféricas a la placa se implementarán los botones, puertos y componentes que completan el laboratorio virtual de prácticas.

- **Control de errores:** El applet informará al usuario de los errores que haya podido cometer a la hora de interconectar los elementos que componen el circuito digital. Estos errores impiden realizar la simulación, por lo tanto cuando el usuario interconecte de forma errónea elementos del circuito se parará la simulación automáticamente.

2.5. PLANIFICACIÓN DEL PROYECTO

En este apartado se va a describir el plan de trabajo a seguir para poder llevar a cabo con éxito los objetivos planteados. Se definirán las tareas a realizar a lo largo del proyecto, y como se distribuirán sobre un eje temporal.

2.5.1. Tareas que componen el proyecto

2.5.1.1. Recopilación de conocimientos básicos necesarios

A partir de los objetivos planteados hay que buscar las herramientas existentes y capaces de ser manipuladas por el desarrollador del proyecto. De modo que habrá que recopilar información sobre dos áreas:

- **Teoría sistemas digitales:** Antes de ponerse a codificar hay que tener bien claro lo que se va a construir, habrá que comprender el comportamiento interno de todos los elementos que componen el laboratorio de sistemas digitales. Entender el funcionamiento de la placa, los cables, los chips y los elementos de entrada y salida.

- **Programación en Java:** Para diseñar y codificar el applet hay que tener muy claro los conceptos de programación en Java, su filosofía, y los productos que nos proporciona.

2.5.1.2. Diseño de la aplicación

Finalizada la etapa de recopilación de conocimientos básicos, comenzará el proceso de construcción del applet. Se estructurará el programa a partir de los objetivos marcados y se irá codificando cada elemento que compondrá la aplicación web.

2.5.1.3. Pruebas

Al finalizar la construcción de la aplicación se realizarán pruebas con casos prácticos para asegurar su correcto funcionamiento. No obstante durante todo proceso de elaboración del proyecto se irán realizando pruebas sobre cada elemento codificado.

2.5.1.4. Documentación

La fase final del proyecto servirá para redactar la memoria del mismo y un manual de instrucciones a nivel de usuario. El manual de usuario se colgará en la web desde donde se ejecutará el applet.

2.5.2. Tareas a realizar a lo largo del diseño de la aplicación

Ésta es la fase más larga y compleja del proyecto, para facilitar su implementación se dividirá en tareas más reducidas. Dichas tareas representan los objetivos planteados.

2.5.2.1. Diseño de la interfaz de usuario

Esta tarea constará del diseño y codificación de la pantalla principal. La pantalla principal del applet mostrará la placa en su zona central, y los componentes de entrada y salida en su zona periférica.

2.5.2.2. Gestor de Cables

En esta etapa se elaborará un módulo capaz de permitir la construcción y eliminación de cables de un modo sencillo e intuitivo.

2.5.2.3. Gestor de Chips

Fase en la que se construirán los elementos que permitirán la inserción de chips en la placa, su gestión de movilidad, su funcionamiento y especificaciones internas y su eliminación de la placa.

2.5.2.4. Política de simulación

Posiblemente, junto con la gestión de cables, sea la fase más compleja del proyecto, en esta tarea se deberá de interpretar el diseño construido sobre la placa para poder realizar una simulación los más real posible. A partir de las conexiones realizadas con los cables habrá que crear conjuntos de pares de

conexión, cada conjunto indicará los elementos relacionados entre sí y formará un evento de la simulación.

A cada evento de la simulación habrá que asignarle un orden temporal y finalmente ejecutar los eventos en el orden asignado. En el momento que se detecten conexiones que generen elementos ilegales para la simulación, habrá que paralizar el proceso e informar al usuario del suceso.

2.5.2.5. Integración del generador de ondas

Etapa en la que se integrará el módulo ya construido del generador de ondas, habrá que readaptar las características propias del módulo para poder permitir una interconexión con la placa y los elementos que la componen.

2.5.2.6. Integración del Osciloscopio

Etapa similar a la anterior pero con el módulo del Osciloscopio.

2.5.3. Planificación temporal

A partir de las tareas planteadas y los recursos existentes, el proyectista como desarrollador, se elabora una planificación del tiempo necesario para llevar a cabo el proyecto. Teniendo en cuenta que para iniciar algunas tareas hay que esperar a la finalización de otras y que algunas se pueden realizar en el mismo intervalo de tiempo por no existir dependencia ente ellas.

A continuación se muestra el cuadro 2.1 con las precedencias de las tareas. El cuadro indica: por cada tarea planteada a que tareas tiene que esperar a que se terminen para poderse iniciar.

Se observan tres columnas, el número de la tarea, el nombre de la tarea, y la tarea que hay que esperar a que se finalice.

La tarea número 7 “Gestor de Cables” tiene que esperar a la tarea número 6 “Diseño de la Interfaz de Usuario”. Implícitamente esto significa que la tarea número 7 deberá esperar también a que se finalicen las tareas de las cuales depende la tarea número 6. En conclusión, no se iniciará el proceso de gestión de los cables hasta haber diseñado la interfaz de usuario y haber recopilado los conocimientos básicos necesarios.

La tarea 5 está compuesta por las subtareas 6,7,8,9,10 y 11, su finalización dependerá de la finalización del conjunto de éstas.

El criterio para determinar que tareas tienen que esperar a que se lleven a cabo otras tareas es la dependencia. Para poder implementar la codificación de la “gestión de los cables” es necesario que exista una interfaz de usuario donde construir y manipular cables.

Hay tareas que por ser más sencillas se pueden ir desarrollando simultáneamente con otras tareas, es decir si cada día se dedican tres horas al

proyecto, una hora y media iría destinada a una tarea y la siguiente hora y media a otra tarea.

	Nombre de tarea	Predecesoras
1	**Proyecto de contrucción de un laboratorio virtual de sistemas digitales**	
2	**Recopilación de los conocimientos básicos necesarios**	
3	Teoria Sistema Digitales	
4	Programación Java	
5	**Diseño de la aplicación**	**3**
6	Diseño de la interfaz de usuario	
7	Gestor de Cables	6
8	Gestor de chips	7
9	Política de Simulación	8
10	Integración Generador de Ondas	9
11	Integración Osciloscopio	10
12	Pruebas	11
13	Documentación	11

Cuadro 2.1 Actividades predecesoras en el proyecto.

2.5.3.1 Distribución temporal

A partir de la complejidad de las tareas se ha elaborado una estimación de los días que se va a tardar en ser realizadas. El cálculo está basado a un trabajo diario de tres horas por día.

El siguiente cuadro 2.2 muestra la estimación de duración de cada tarea.

	Nombre de tarea	Duración
1	**Proyecto de contrucción de un laboratorio virtual de sistemas digitales**	**167 días**
2	**Recopilación de los conocimientos básicos necesarios**	**7 días**
3	Teoria Sistema Digitales	7 días
4	Programación Java	7 días
5	**Diseño de la aplicación**	**145 días**
6	Diseño de la interfaz de usuario	20 días
7	Gestor de Cables	60 días
8	Gestor de chips	20 días
9	Política de Simulación	30 días
10	Integración Generador de Ondas	5 días
11	Integración Osciloscopio	10 días
12	Pruebas	2 días
13	Documentación	15 días

Cuadro 2.2 Las tareas del proyecto y su duración estimada.

La tarea más compleja del proyecto es el diseño de la aplicación, se estima su duración en 145 días, de ellos 60 para la gestión de cables y 30 para la política de simulación, las dos tareas más complicadas de todo el proyecto.

A continuación se muestra el gráfico de Gantt, cuadro 2.3, donde se observa el desarrollo del proyecto a lo largo de 9 meses, se observa la dependencia de las tareas y su distribución entre los meses de septiembre y junio.

Los números que se encuentran a continuación de las tareas indican el número de días estimados para completarlas.

2.6. VIABILIDAD TÉCNICA DEL PROYECTO

Uno de los objetivos del proyecto es construir una herramienta de fácil acceso a los usuarios y sin necesidad de instalación. Los applets se amoldan perfectamente a esta idea ya que permiten la elaboración de aplicaciones sin la necesidad de ser instaladas, el concepto y filosofía de los applets está muy bien desarrollado en Java que proporciona unas herramientas muy potentes para facilitar su construcción. Además, la empresa responsable de Java, Sun Microsystems, también proporciona libremente las plataformas que permiten a los usuarios la ejecución de los applets. Por todos estos motivos se considera que Java es la herramienta idónea para la elaboración de la aplicación.

2.7. HERRAMIENTAS DE DESARROLLO

A la hora de programar y codificar el applet se utilizará el paquete SDK. SDK es un kit de recursos que ayuda al programador de Java a desarrollar código. El paquete lo distribuye libremente la empresa Sun Microsystems.

Por otra parte será necesario espacio libre en un servidor web para colgar un documento html que llame a la ejecución del applet. El servidor debe permitir que el cliente pueda bajarse las clases de Java que producen la puesta en marcha del applet. Se estima que serían necesarios unos 40 MB para poder alojar la aplicación web y soportar futuras ampliaciones.

2.8. ANÁLISIS DE VIABILIDAD ECONÓMICA

Este proyecto por su carácter docente no tiene ningún tipo de ambición lucrativa. No está enfocado hacia profesionales ni se pretende ningún tipo de beneficio económico.

Los costes de almacenamiento de un servidor web serán nulos o a cargo del departamento de microelectrónica y sistemas electrónicos de la Universitat Autònoma de Barcelona.

El uso de Java y sus clases tampoco implica coste alguno, ya que Sun Microsystems distribuye libremente sus kits de desarrollo de Java, siempre y cuando el objetivo de la aplicación codificada no sea lucrativo.

El único coste que se deriva del proyecto es la licencia de Microsoft Windows XP, sistema operativo que gestiona el ordenador desde donde se desarrollará el proyecto. El precio de la licencia de uso es aproximadamente de unos 97,70 euros.

El desarrollador del proyecto no va a recibir ningún tipo de beneficio económico, por todas estas razones el proyecto es viable económicamente.

2.8.1. Supuestos costes

De acuerdo con el apartado 2.5 que trata la planificación del proyecto, se pueden hacer una estimación del coste de desarrollo.

A partir de las horas estimadas que se van a dedicar a cada tarea, y poniendo un sueldo para el programador entre los 12 y los 18 euros por hora estos serían los costos del proyecto.

Mínimo estimado de 12 euros/hora:
- Duración del proyecto: 167 días.
- Horas de dedicación por día: 3 horas.
- 12 euros la hora.

Total: 6.012 euros.

Máximo estimado de 18 euros/hora:
- Duración del proyecto: 167 días.
- Horas de dedicación por día: 3 horas.
- 18 euros la hora.

Total: 9.018 euros.

En el supuesto de que el desarrollador cobrará algún incentivo económico, de acuerdo con las cifras expresadas anteriormente, el proyecto seguiría siendo viable ya que no son cifras prohibitivas. No obstante, este aspecto es un supuesto y algo completamente subjetivo que dependerá de la percepción de cada persona.

2.9. VIABILIDAD LEGAL

En este apartado se examina si el desarrollo de la aplicación web incumple alguna norma o regla de la legalidad vigente.

El proyecto no vulnera ningún derecho de autor ya que todos los elementos que lo componen son originales y no se han copiado de ninguna fuente.

En lo referente al uso de Java y el paquete proporcionado por Sun Microsystems, su uso es completamente gratuito y sólo en el caso que se le sacará algún rendimiento económico se podría estar incumpliendo alguna normativa sobre los derechos de autor.

No hay por tanto ningún factor legal que pudiera impedir el desarrollo del applet.

2.10. CONCLUSIONES ESTUDIO DE VIABILIDAD

El proyecto por sus características va a permitir desarrollar y poner a prueba los conocimientos adquiridos en diversos campos de la informática:

1. Programación orientada a objetos, gracias al lenguaje Java que mediante sus clases y objetos deberá desarrollar el applet.

2. Programación recursiva, necesaria para implementar la gestión de los pares de componentes conectados mediante cables.

3. Sistemas digitales, profundizar en la comprensión de los diferentes componentes que conforman un laboratorio de prácticas.

Por otra parte una vez finalizado el proyecto, será de gran utilidad para los estudiantes y profesores de las asignaturas de sistemas digitales, electrónica digital o fundamentos de computación, por los motivos anteriormente expuestos.

Además el proyecto cumple con les tres puntos de vista básicos para ser viable. Una vez determinadas las tareas del proyecto, y el tiempo de dedicación a cada tarea se demuestra que el proyecto es viable desde el punto de vista técnico. Desde el punto de vista económico el proyecto también es viable debido a que no va a comportar coste alguno, el proyecto no tiene fines lucrativos.

Finalmente el proyecto no vulnera la legalidad vigente como se ha expuesto en la sección 2.9 relacionada con la viabilidad legal.

Por todas estas razones se considera la aplicación “Laboratorio virtual de sistemas digitales” como algo viable como proyecto final de carrera.

CAPÍTULO 3:

FUNDAMENTOS TEÓRICOS

3. FUNDAMENTOS TEÓRICOS

3.1. SIMULADOR DEL LABORATORIO DE SISTEMAS DIGITALES

El alumnado de Sistemas Digitales se encuentra en los laboratorios de prácticas con una serie de componentes con los que realizar la implementación de los circuitos previamente diseñados. Es indispensable un buen conocimiento de los diferentes elementos para poder realizar correctamente la práctica.

Para diseñar la aplicación web que simula el laboratorio se ha basado en los elementos que se van a encontrar los estudiantes de la asignatura, la serie de componentes son los siguientes:

- **Placa:** Soporte básico donde se instalaran los elementos que configuran el circuito digital.

- **Chips:** Son circuitos integrados que producen una señal digital a partir de una señal de entrada y una función matemática definida por las especificaciones internas de cada chip.

- **Cables:** Hilos que permiten conectar y comunicar los diferentes elementos que participan en un circuito digital.

- **Componentes de entrada:** Son una serie de elementos que generan diferentes señales digitales para que posteriormente sean procesadas por los chips.

- **Componentes de salida:** Son una serie de elementos capacitados para interpretar y reproducir una señal digital de forma visual o sonora.

A continuación se profundiza en algunos aspectos teóricos de los componentes que participan en el simulador del laboratorio de sistemas digitales.

3.2. PLACA

La placa es la base donde se construye un circuito digital, en ella se permite la instalación de chips, resistencias, leds, 7-segmentos o cables entre otros elementos. La placa está formada por una serie de conexiones intercomunicadas internamente, los elementos se insertan en estas conexiones.

Existen dos series de conexiones diferentes en una placa, la primera es una colección horizontal de conexiones que siempre tendrán en común la misma señal digital. Las conexiones de la placa están comunicadas internamente de tal manera que cuando llega una señal a una conexión ésta se propaga horizontalmente a todas las de su serie.

La segunda serie de conexiones tiene un funcionamiento exactamente igual al primero pero su relación es vertical.

En la figura 3.1 se observa la fotografía de una de las placas más habituales del laboratorio de sistemas digitales, en la figura posterior 3.2 se muestra la placa diseñada para el simulador y como se distribuyen las diferentes conexiones de la placa.

Figura 3.1. Fotografía de una placa de laboratorio.

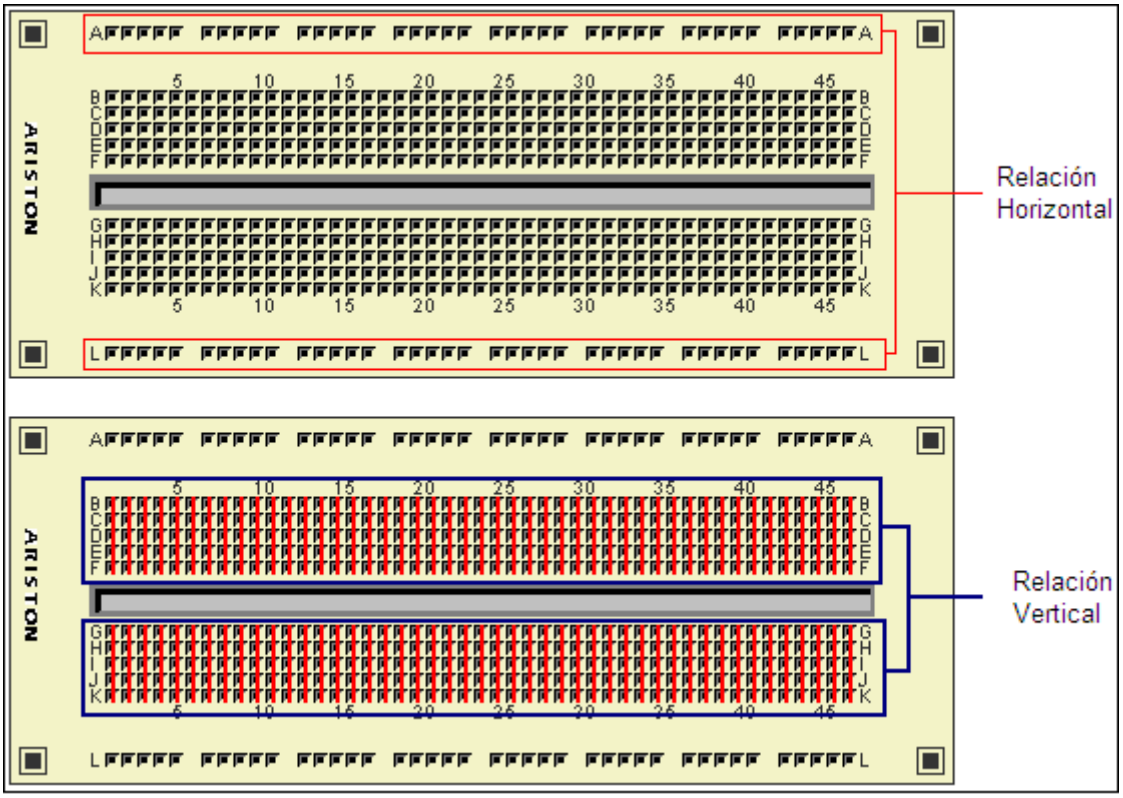


Figura 3.2. Esquema de relaciones internas de las conexiones de la placa.

La zona central de la placa está destinada a la instalación de los chips, la separación existente entre las matrices de conexiones corresponde a la medida estándar de la anchura de los chips. El vector horizontal superior de conexiones está pensado para que reciba una señal de alimentación (VDD), o lo que es lo mismo una señal digital constante a 1 lógico. Por la contra el vector horizontal inferior está pensado para que se le conecte una señal a tierra (GND) o señal digital constante a 0 lógico. No obstante estos dos puntos no son más que una convención a la hora de implementar correctamente circuitos digitales, ambos vectores de conexiones pueden recibir cualquier tipo de valor.
En la figura 3.3. Se observa como se ha insertado un chip en la placa y la zona de conexiones influenciadas por el chip. También se indica los vectores de conexiones horizontales y sus teóricas señales que deberían recibir para una correcta implementación de un circuito digital.

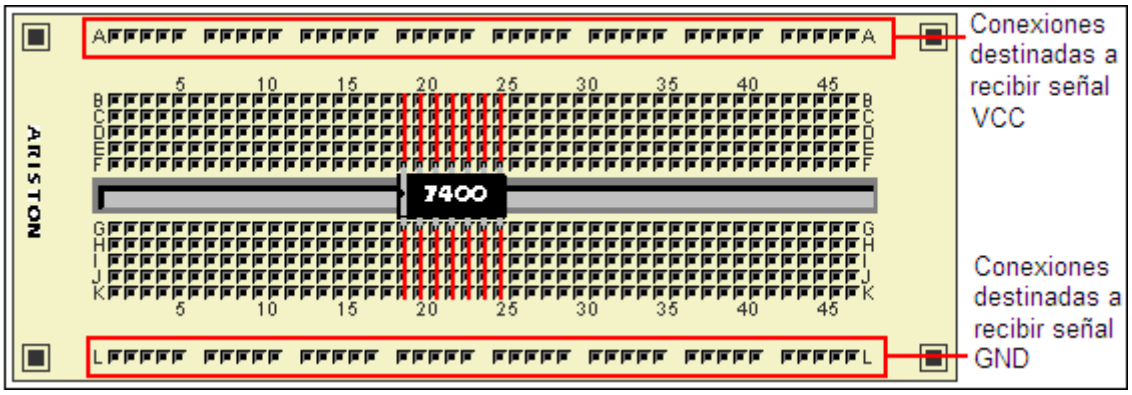


Figura 3.3. Instalación de un chip y zona de influencia.

3.3. CHIPS

Un chip es un circuito integrado en soporte de silicio formado por transistores y otros elementos electrónicos miniaturizados. Los chips disponen de una serie de patas que se conectan a la placa, estas patas son puertos de entrada o salida de señales digitales. Cada chip manipula, dependiendo de su función lógica, las señales que recibe por sus patas de entrada, el resultado de esa manipulación lo difunde mediante las patas de salida que propagarán la señal a las conexiones de la placa en las que estén insertadas.

En la imagen 3.4 podemos observar uno de los chips más habituales en el laboratorio de prácticas. El chip consta de catorce patas, diez de ellas son puertos de entrada a señales digitales y cuatro de ellas puertos de salida de señales digitales manipuladas por el chip.

Figura 3.4. Fotografía Chip 7400.

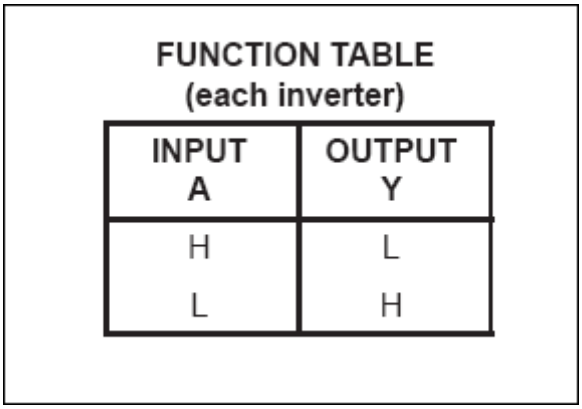
3.3.1. Tipología de Chips

Cada tipo de chip se caracteriza por su comportamiento interno, es decir, su forma de manipular las señales de entrada. Para manipular estas señales los chips implementan funciones lógicas. Una función lógica es una ecuación matemática cuyos valores sólo pueden ser binarios. Los valores binarios que adquieren las patas de salida de un chip están establecidos por el resultado de esta ecuación matemática.

Una de las maneras de expresar una ecuación lógica es la tabla de la verdad. La tabla de la verdad determina uno o más valores de salida a partir de uno o

más valores de entrada. Los fabricantes de chips facilitan las tablas de la verdad de las ecuaciones que implementan sus productos.

En la figura 3.5 se observa un extracto de la tabla de la verdad del datasheet (Documento de especificaciones técnicas) del chip SN7404 del constructor Texas Instruments. El chip SN7404 realiza la función de inversión, es decir, la señal de salida es la negación de la señal de entrada, cuando entra un 1 la salida es 0 y viceversa.



FUNCTION TABLE
(each inverter)

INPUT A	OUTPUT Y
H	L
L	H

Figura 3.5. Tabla de la verdad del chip SN7404.

En la figura 3.6 se observa un extracto del esquema del datasheet del chip SN7404 del constructor Texas Instruments. El esquema nos muestra la función de cada pata del chip. En este caso el chip tiene 14 patas, 8 de ellas son de entrada y 6 de salida. Todos los chips tienen dos patas que se conectan a una señal de alimentación (VCC) y a una señal de tierra (GND), es imprescindible que su conexión sea correcta para que el chip funcione correctamente.

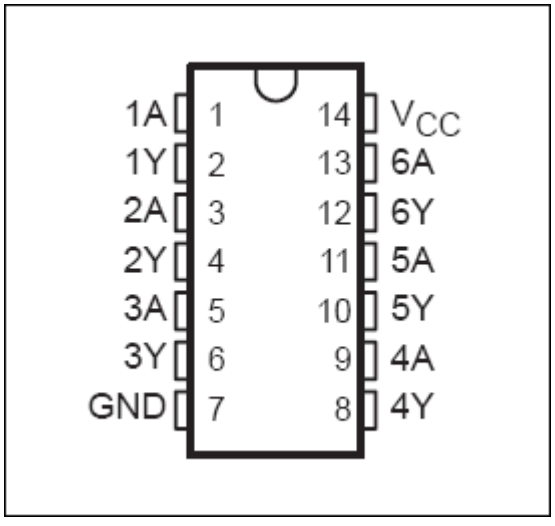


Figura 3.6. Esquema del chip sn7404.

Para este caso específico, las salidas que están marcadas con una Y dependen de una única entrada (A). Por lo tanto la salida 3Y correspondiente a la pata 6, dependerá de la entrada 3A (pata 5) y la ecuación lógica configurada en el chip.

Los chips son circuitos integrados que implementan puertas lógicas, las puertas lógicas realizan operaciones matemáticas de acuerdo con el álgebra de Boole. Existen hasta siete puertas lógicas para realizar las operaciones booleanas de suma, multiplicación, negación, negación de la suma, negación de la resta, suma exclusiva y negación de la suma exclusiva.

Cualquier ecuación lógica se puede realizar mediante la combinación de la lista de puertas de la figura 3.7, de hecho mediante procesos simplificadores de funciones booleanas se logra implementarlas tan sólo con una combinación de los tipos de puertas, NOR y NAND.

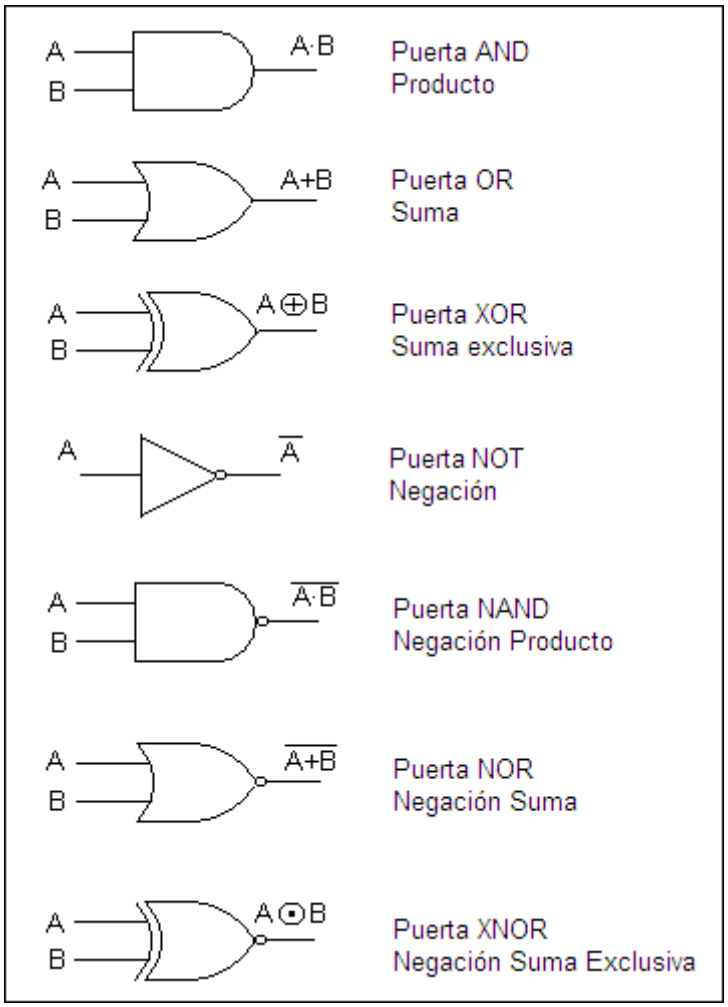


Figura 3.7. Esquema de relaciones entre puertas lógicas y sus operaciones.

En la figura 3.8 se observa las puertas que componen el circuito integrado SN7400, este chip implementa la función del producto negado mediante 4 puertas de tipo NAND.

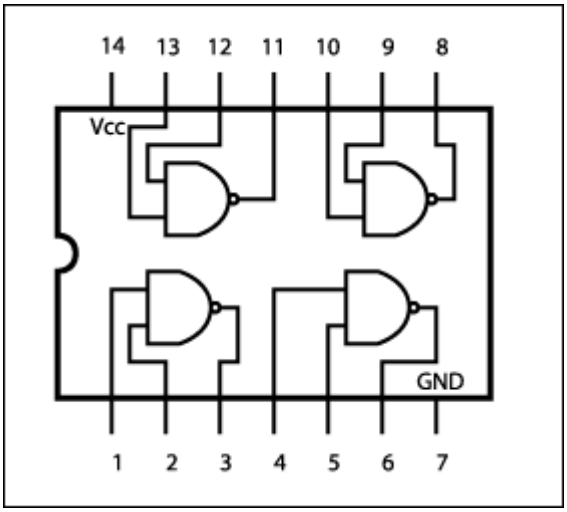


Figura 3.8. Esquema interno del Chip SN7400.

El número de ecuaciones lógicas existente es ilimitado por lo tanto el número de tipos de chips posibles también lo es. Existen en el mercado diversos fabricantes de chips, como Texas Instruments o FairChild Semiconductor entre muchos otros, que ofrecen una amplia gama de productos. Para implementar el simulador del laboratorio se ha elegido una lista de los chips de uso más habitual en las prácticas de la asignatura.

Finalmente, los diferentes chips existentes en el mercado se pueden clasificar según su funcionalidad. En la figura 3.9 se muestra la ventana de inserción de chip del simulador, en ella podemos ver como se clasifican los diferentes chips que ofrece la aplicación. Los chips implementados en la listan son los de uso más habitual en el laboratorio de prácticas de sistemas digitales.

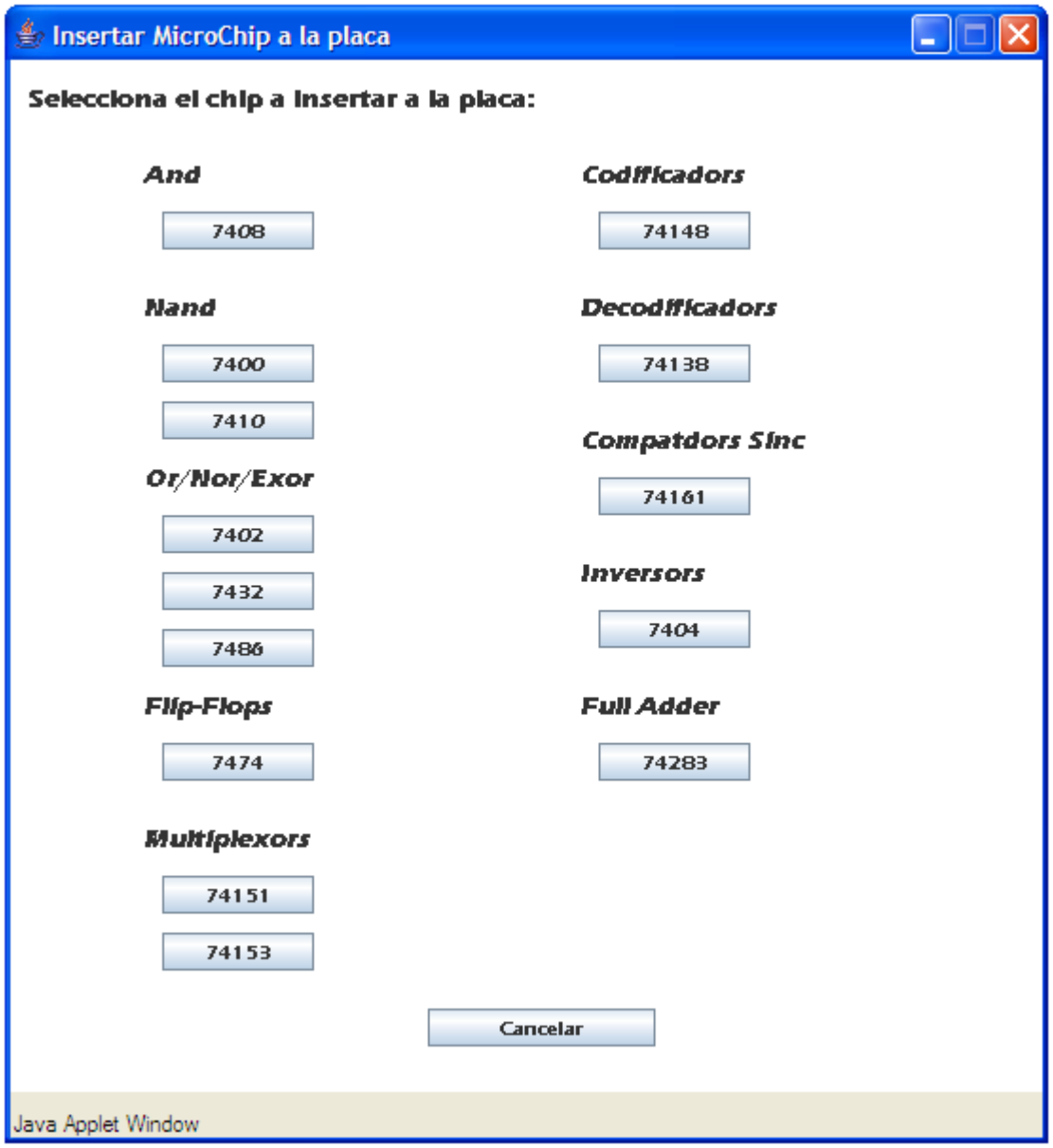


Figura 3.9. Ventana de inserción de chip.

3.4. CABLES

Los cables son un elemento esencial en el diseño de un circuito digital, su función es la de transportar la señal digital entre los diferentes componentes interconectados en el circuito. Un mal estado físico o una mala colocación impedirán el correcto funcionamiento de la práctica.

Los cables que se encuentran los estudiantes en el laboratorio de prácticas están formados por hilos de material conductor recubiertos por un plástico exterior. Estos cables están pelados en sus extremos para permitir una correcta conexión a la placa.

En la figura 3.10 se observa el diseño de un circuito digital en el simulador de laboratorio de sistemas digitales, en él se encuentran diversos cables que intercomunican los componentes del circuito.

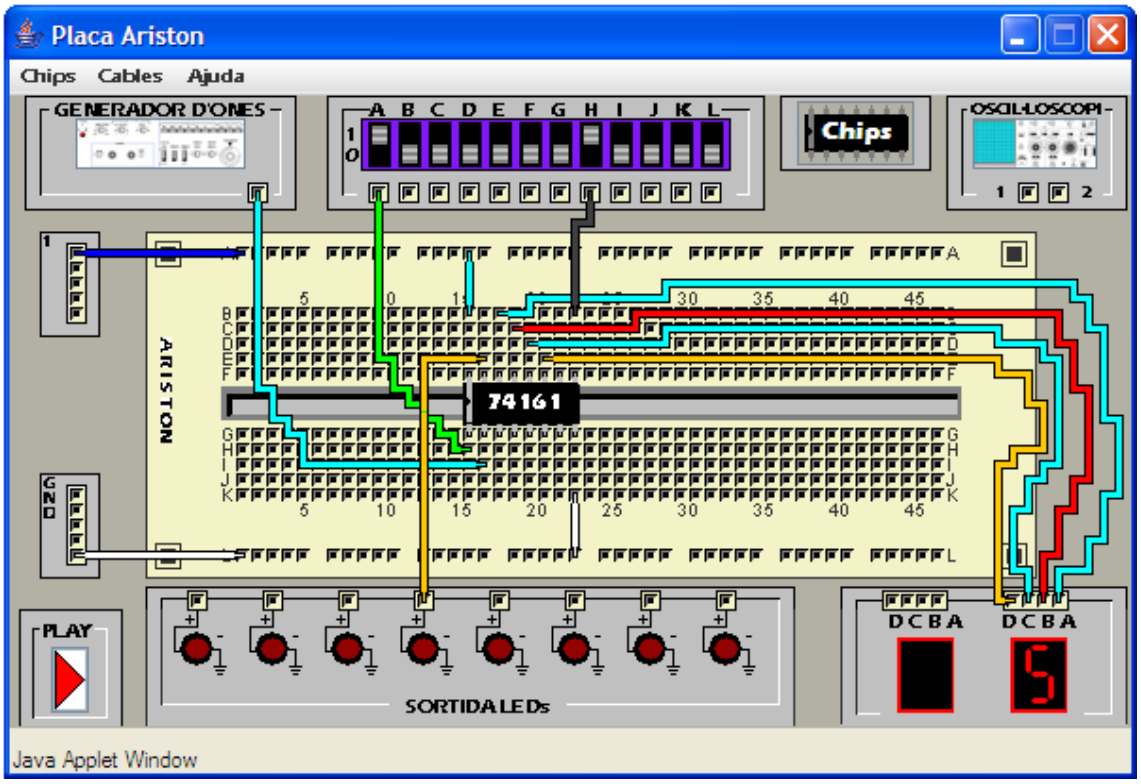


Figura 3.10. Cables en el diseño de un circuito digital.

3.5. COMPONENTES DE ENTRADA

Los componentes de entrada son una serie de elementos que generan diferentes señales digitales, son por lo tanto origen o fuente de un flujo digital. En el laboratorio de sistemas digitales se encuentran fuentes de alimentación y generadores de onda, en la aplicación web se han implementado estos componentes y se han añadido algunos elementos más que facilitan el uso del simulador.

3.5.1. Interruptores

Este elemento no existe en el laboratorio de prácticas, pero debido a que algunos ejercicios relacionados con las prácticas de la asignatura exigen que un mismo cable varíe el valor digital de forma manual, se ha dotado a la aplicación de un conjunto de interruptores para optimizar esta operación.

En la práctica real el alumno tiene que mover el extremo de un cable hacia la conexión de la placa con el valor deseado, en el simulador solo tendrá que mover el interruptor para obtener un 0 o un 1 lógicos.

En la figura 3.11 se observa como un mismo cable varía su valor lógico, dependiendo de la posición del interruptor. Cuando el cable tiene un valor lógico de 1, el led se ilumina, en caso contrario el led permanece apagado.

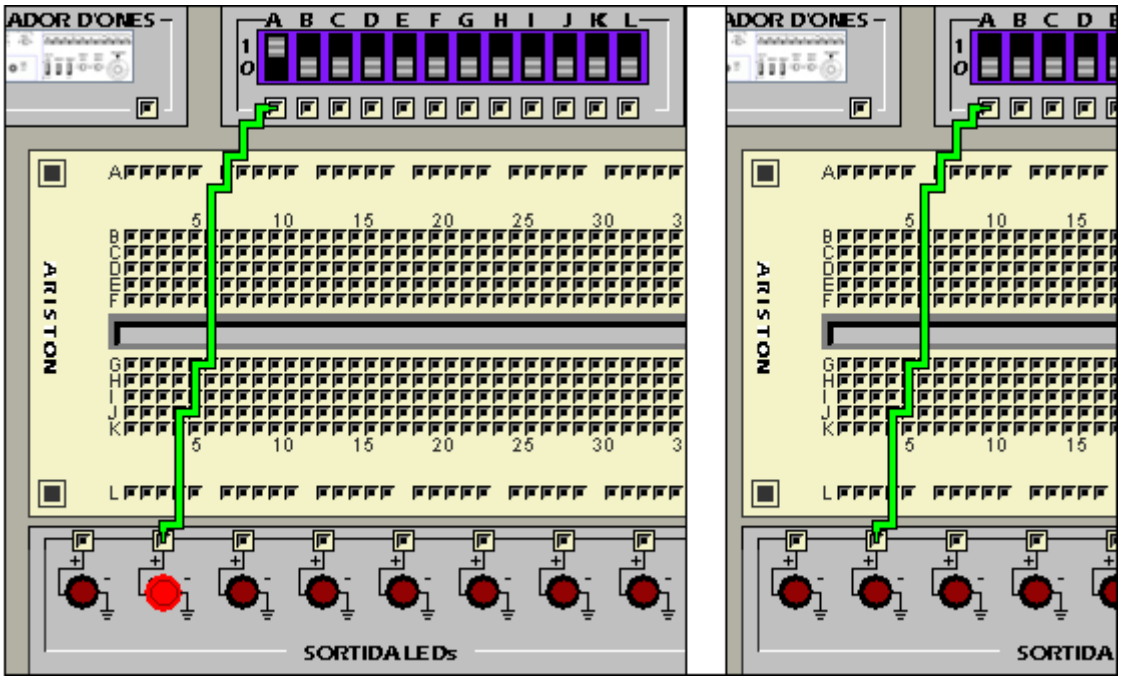


Figura 3.11. Modificación del valor lógico de un cable mediante interruptor.

3.5.2. Fuente de alimentación

Los alumnos de prácticas de sistemas digitales se encuentran en el laboratorio con la fuente de alimentación. Ésta genera un voltaje constante que es interpretado lógicamente como una señal digital constante a 1. El estudiante de la asignatura deberá configurar el voltaje del aparato de acuerdo a las especificaciones que le indique el profesor, que serán aquellas que impidan que se quemen los diferentes componentes del circuito.

En la figura 3.12 se observa una de las fuentes de alimentación más comunes del laboratorio de sistemas digitales.

Figura 3.12. Fuente de alimentación Tektronix CPS250.

A la hora de implementar el simulador del laboratorio se ha simplificado la existencia de la fuente de alimentación ya que su uso es trivial. De esta manera la fuente de alimentación la podemos encontrar en forma de 5 puertos de conexiones que generan señales digitales constantes a 1 lógico. En la figura 3.13 se observa como se ha solucionado la fuente de alimentación en la aplicación.

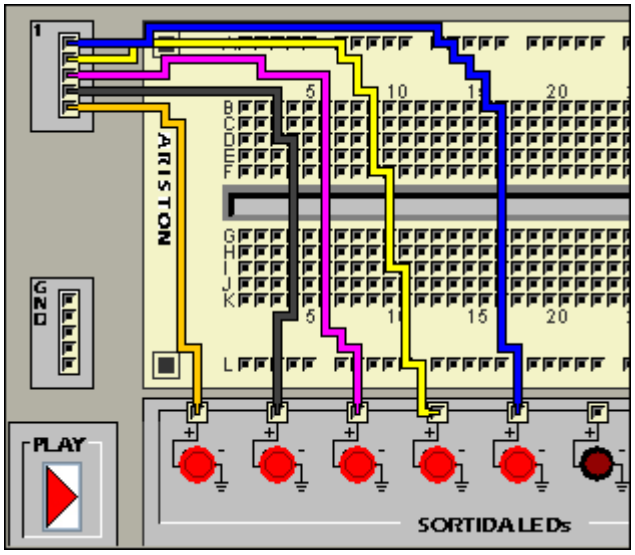


Figura 3.13. Detalle de la fuente de alimentación.

3.5.3. Tierra (GND)

Los chips, leds o 7-segmentos tienen que estar conectados a una señal de alimentación y/o también a una señal de tierra. Un voltaje cercano o igual a los 0 voltios corresponde a una señal de tierra o un 0 lógico. Este voltaje es generado por la fuente de alimentación.

Del mismo modo que en el punto anterior, para simplificar esta función de la fuente de alimentación en el simulador, se han implementado 5 puertos etiquetados como GND (Ground, tierra en inglés) que suministran señales digitales a 0 lógico.

Por cuestiones de simplificación, en el caso específico de los leds y los 7-segmentos no es necesaria esa conexión en el simulador. Además en los leds se deja indicada gráficamente.

En las figuras 3.14 y 3.15 se observa la conexión a tierra de los leds y la de un chip respectivamente.

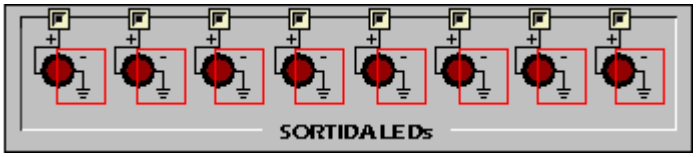


Figura 3.14. Detalle de las conexiones a tierra de cada led.

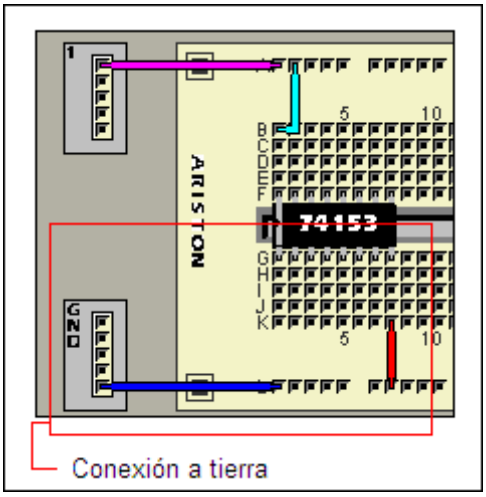


Figura 3.15. Conexión a tierra de un chip.

3.5.4. Generador de Ondas

El generador de ondas es un complejo aparato capaz de producir hasta tres tipos diferentes de señales electrónicas: señales sinusoidales, señales triangulares y señales digitales de onda cuadrada.

Las prácticas de la asignatura de sistema digitales sólo tratan señales digitales de onda cuadrada, por lo tanto sólo se utilizaran una serie de funciones del generador directamente relacionadas con la producción de señales de onda cuadrada.

En la figura 3.16 se muestra uno de los modelos de generador de ondas más habituales en los laboratorios de prácticas.

Figura 3.16. Fotografía del generador de ondas modelo Tektronix CFG250.

Las funciones de interés para el alumnado de prácticas se reducen al botón de selección del tipo de onda a generar, siempre será el de onda cuadrada, y la gama de botones relacionados con el cálculo de la frecuencia de la señal a generar.

El generador de ondas produce una señal de reloj o señal de pulsos, cada vez que se cumpla la mitad de un periodo del generador habrá un cambio en la señal producida, si la antigua señal era un 1 lógico la nueva señal será un 0

lógico y la posterior volverá a ser un 1. El periodo del generador dependerá de la frecuencia seleccionada por el usuario del aparato.

En la figura 3.17 se muestra un gráfico de evolución a lo largo del tiempo de la señal producida por un generador de ondas.

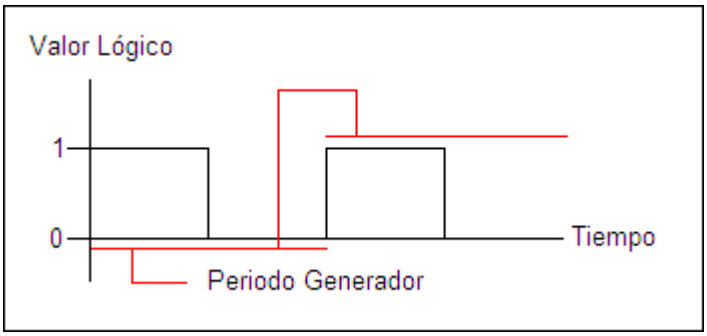


Figura 3.17. Señal Cuadrada Generador de Ondas.

En la aplicación se ha integrado un generador de ondas previamente implementado, este generador sólo tiene habilitadas las funciones relacionadas con la generación y gestión de señales digitales, únicas funciones necesarias para los objetivos de este proyecto.

El aspecto del generador es fiel al modelo Tektronix CFG250, por otra parte el más común entre los que se pueden encontrar en el laboratorio de sistemas digitales. La señal que produce el generador del simulador es transportada a un puerto desde donde se podrá distribuir, mediante las conexiones realizadas por los cables, a todos los elementos del circuito que desee el usuario.

En la figura 3.18 se observa el generador de ondas implementado y en la figura 3.19 el puerto desde donde se obtiene la señal del mismo.

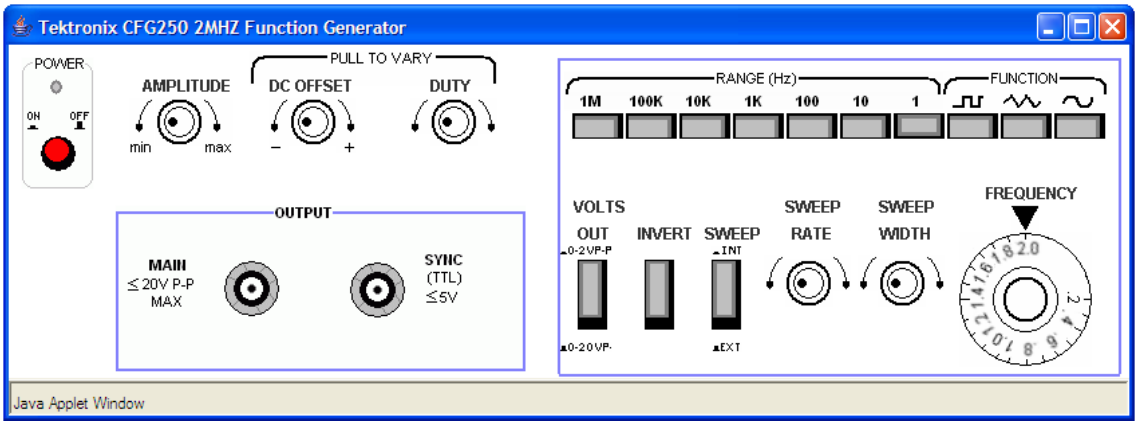


Figura 3.18. Generador de Ondas Implementado.

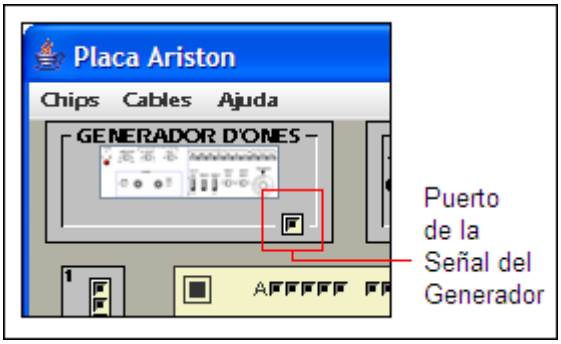


Figura 3.19. Puerto del Generador de Ondas.

3.6. COMPONENTES DE SALIDA

Los componentes de salida permiten interpretar una señal digital. En el laboratorio de sistemas digitales nos encontraremos con aparatos capaces de interpretar señales de una forma visual, también existen mecanismos que pueden interpretar señales digitales de manera sonora pero quedan fuera de los objetivos del proyecto.

3.6.1. Diodos emisores de Luz, Leds

Un LED, siglas en inglés *Light-Emmiting Diode* (Diodo emisor de Luz), es un dispositivo semiconductor que emite luz. El color de la luz dependerá del material empleado en la construcción del diodo, mayoritariamente encontraremos leds de color rojo en los laboratorios de prácticas, no obstante se pueden encontrar de cualquier color.

Los leds tienen dos patas, para su funcionamiento correcto una de ellas tiene que estar conectada a una señal de tierra y la otra se conectará a una conexión de la placa a la espera que le llegue una tensión eléctrica superior a un voltaje y medio y comenzar a emitir luz.

Los leds se activan entre el rango de voltajes que va del 1,5 a los 2,2 voltios, por lo tanto será necesaria la colocación de una resistencia que mengüe la tensión de la placa para poder proteger al led de posibles quemaduras.

En la figura 3.20 se observa la fotografía de un conjunto de leds.

Figura 3.20. Conjunto de leds.

Por cuestiones de simplificación se ha omitido el uso de resistencias en el simulador de circuitos digitales. Se ha habilitado un grupo de ocho leds para poder diseñar de un modo más sencillo los circuitos digitales y que no sea necesaria su inserción en la placa. Los leds emitirán luz al recibir una señal lógica a 1.

3.6.2. 7-segmentos

Un 7-segmentos es un tipo de display o visualizador formado por siete pequeñas fuentes luminosas que muestran los dígitos del 0 al 9 según las fuentes o segmentos iluminados.

Estos segmentos son leds situados geométricamente de tal forma que al estar todos iluminados forman un 8. Del mismo modo que los leds o los chips, una de sus patas debe de estar conectada a una fuente de alimentación y otra a una señal de tierra para su correcto funcionamiento.

Como los leds, el 7-segmentos debe de estar protegido de voltajes elevados por una resistencia. En la figura 3.21 observamos una fotografía de un 7-segmentos.

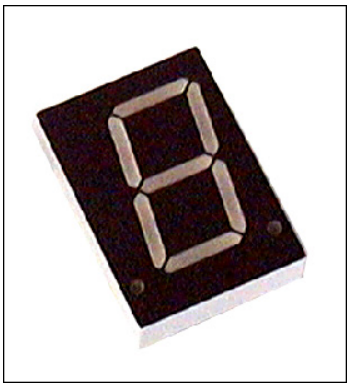


Figura 3.21. 7-segmentos.

Los 7-segmentos tienen siete patas de entrada correspondientes a cada segmento, cuando una pata recibe una señal lógica a 1 se ilumina el segmento correspondiente.

Para visualizar de un modo cómodo los dígitos del 0 al 9, en los diseños de los circuitos digitales se utiliza el chip conversor 7447. Este chip permite visualizar números a partir de cuatro variables, es decir recibe 4 valores de entrada y activa las patas del 7-segmentos adecuadas para representar el número decimal que corresponda. Las cuatro variables y el número que corresponde los podemos observar en la siguiente tabla.

Entrada D	Entrada B	Entrada C	Entrada A	Salida Decimal
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

En la figura 3.22 se observa el funcionamiento del chip 7447 y su relación con el 7-segmentos.

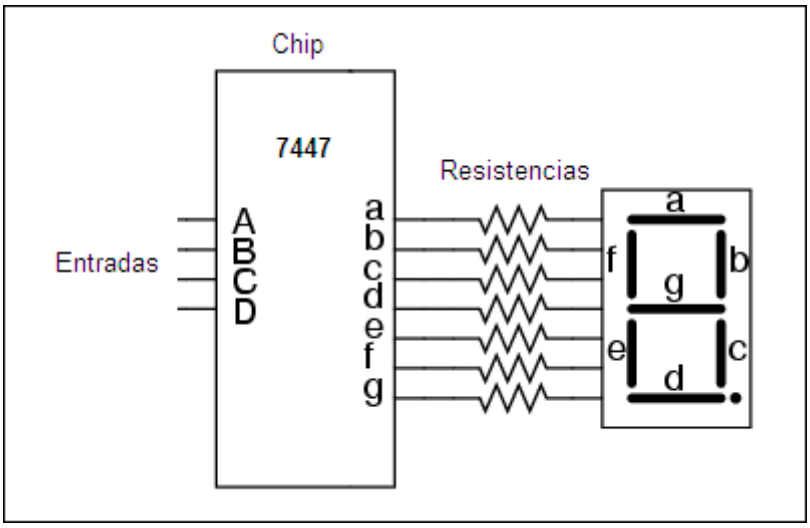


Figura 3.22.Funcionamiento del 7447.

Por motivos de simplificación al utilizar el 7-segmentos se ha omitido el uso del chip 7447 en la aplicación implementada, así pues se ha habilitado de cuatro puertos de entrada a los dos 7-segmentos que incorpora el applet. De este modo se puede aplicar directamente la tabla anterior sin tener que pasar por el chip 7447.

En la figura 3.23 se observa la solución diseñada para el simulador.

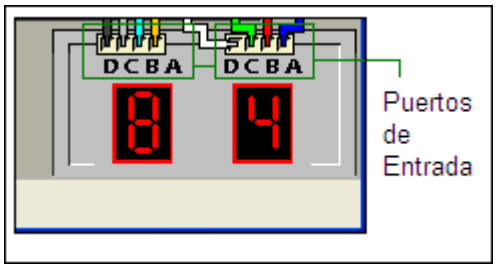


Figura 3.23. 7-segmentos.

3.6.3. Osciloscopio

Un osciloscopio es un instrumento de medición electrónico para la representación gráfica de señales eléctricas que pueden variar en el tiempo.

Presenta los valores de las señales eléctricas en forma de coordenadas en una pantalla, en la que normalmente el eje X (Horizontal) representa tiempos y el eje Y (Vertical) representa tensiones. La imagen obtenida se denomina oscilograma.

En un osciloscopio existen dos reguladores para ajustar la señal de entrada, estos permiten medirla en pantalla para posteriormente poderla evaluar. El primer regulador modifica el eje X (Horizontal) apreciando las fracciones de tiempo (segundos, milisegundos, microsegundos,…).

El segundo regula el eje Y (Vertical) controlando la tensión de entrada (voltios, milivoltios, microvoltios,…). La calidad de la imagen dependerá de la resolución del aparato.

Las regulaciones efectuadas determinan el valor de la escala cuadricular que divide la pantalla, permitiendo saber cuánto representa cada cuadrado de ésta para, en consecuencia, conocer el valor de la señal a medir, tanto en tensión como en frecuencia.

En la figura 3.24 se muestra la fotografía de uno de los osciloscopios más habituales de los laboratorios de prácticas de sistemas digitales.

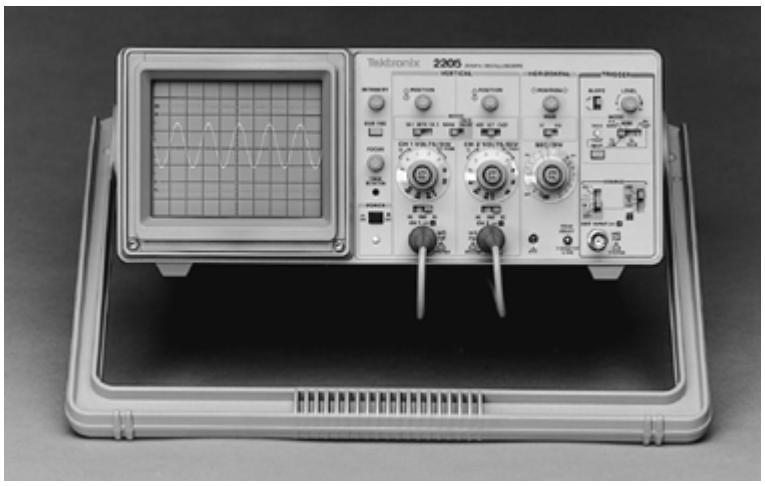


Figura 3.24. Osciloscopio Tektronix 2205.

Los osciloscopios que nos encontramos en el laboratorio, al igual que los generadores de ondas, son capaces de manipular diferentes señales electrónicas, las únicas funciones utilizadas por el alumnado de la asignatura son las que están relacionadas con las señales digitales.

Los modelos que van a utilizar los estudiantes permiten la entrada de dos señales digitales, ambas señales se pueden visualizar y medirlas mediante la

instrumentación del osciloscopio, además el aparato permite la visualización simultánea de las mismas.

El osciloscopio también está capacitado para sumar las dos señales de entrada, invertirlas o negarlas. También permite definir el trazo y la intensidad del gráfico creado a partir de la señal así como su posición a lo largo de los dos ejes de la pantalla.

El osciloscopio a integrar en la aplicación web tiene habilitadas exclusivamente las funciones relacionadas con el uso y manipulación de señales digitales, su imagen es fiel a la del modelo Tektronix 2205 de la figura 3.24. Para conectarlo con la placa se han colocado dos puertos para las dos señales que pueden ser visualizadas en el osciloscopio. En la figura 3.25 observamos los puertos de señales del osciloscopio y en la figura 3.26 la imagen del osciloscopio integrado en el simulador.

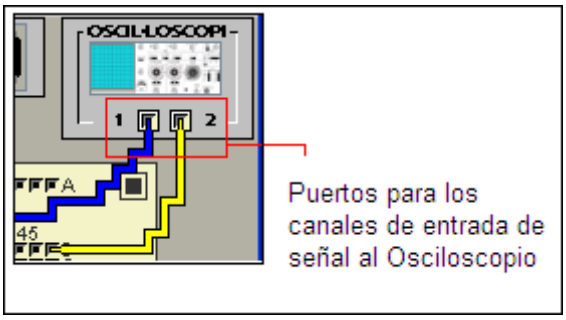


Figura 3.25. Puertos de conexiones Osciloscopio-Placa.

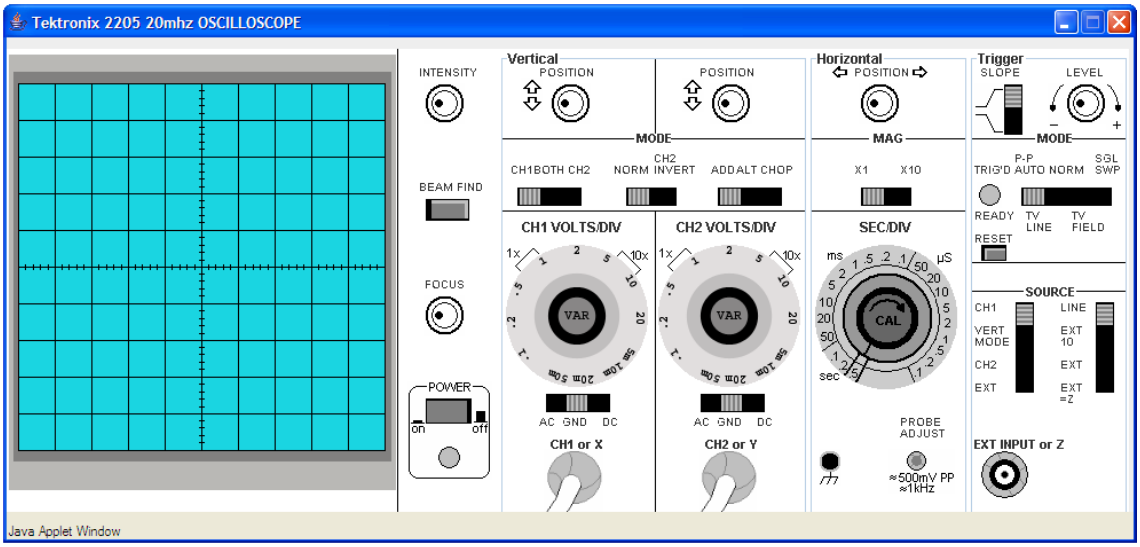


Figura 3.26. Osciloscopio integrado en el simulador.

3.7. CONCLUSIONES FUNDAMENTOS TEÓRICOS

En este capítulo hemos visto toda la teoría básica que hay detrás del laboratorio virtual de sistemas digitales. Se ha repasado el comportamiento, funcionamiento y uso de todos los elementos que participan en las prácticas de la asignatura.

También se han mostrado y explicado las soluciones implementadas en el simulador para cada componente.

CAPÍTULO 4:

ANÁLISIS DE LA APLICACIÓN

4. ANÁLISIS DE LA APLICACIÓN

Este capítulo trata de describir como se ha escrito, estructurado y organizado el código fuente del laboratorio virtual de sistemas digitales.

4.1. ESTILO DE PROGRAMACIÓN

Se ha elegido el criterio más común entre la comunidad de programadores en Java para garantizar que el código sea comprensible y claro, y permita su portabilidad y mantenimiento. Las normas que se describen a continuación son algunas de las que aconseja el autor Pedro Manuel Cuenca Jiménez en su libro "Programación en Java para Internet" publicado por Anaya Multimedia.

4.1.1. Elaboración de identificadores

Cuando los nombres de los identificadores constan de varias palabras se utiliza una mayúscula como separador. Esto son algunos ejemplos: "ControlConnexionsCable", "identificadorCable", "OsciloBasicControls", "senyalEntrada"….

4.1.1.2. Uso de mayúsculas

Java distingue entre mayúsculas y minúsculas, para diferenciar el tipo de elemento (variable, clase, constante) se siguen los siguientes convenios:

- **1:** Los nombres de las clases comienzan por mayúscula: PanelInsertaChip, Chip7400,ParellsConnectats….

- **2:** Los nombres de variables y métodos comienzan por minúscula: dibuixarLedsOn(), dibuixarLedsOff(), main…

- **3:** Los nombres de los paquetes suelen estar escritos con todas sus letras en minúsculas: java.applet.Applet. La clase Applet pertenece al paquete java.applet.

4.1.1.3. Formato de los bloques de código

La llave "{" que introduce un bloque de código Java, se escribe en la misma línea que la instrucción a la que acompaña. La llave que finaliza el bloque se sitúa en una línea aislada, identando el cuerpo del bloque.

```
for (int i = 0; i < 75; i++){
        for (int j = 0; j < 29; j++){
          if (  (parcela[i][j].getEstatParcela()!= 0) && (parcela[i][j].cablesEnParcela.existeixCable(idCable,0)) ){
             parcela[i][j].setEstatParcela(0);
             parcela[i][j].borrarCableComplet (idCable);
          }
        }
  }
```

Si un bloque consta de una sola línea y ésta no es demasiado larga, se escribe a continuación de la construcción de un bloque.

```
if (lustroX1 == 6) lustroX1 = 0;
```

La construcción de un try/catch se escribe con el término match siguiendo en la misma línea a la llave "}" del que precede.

```
try{
}catch(){
}
```

La construcción if/else se encuentra en dos maneras:

a) Como try/catch

```
if (a>b){
}else{
}
```

b) Estilo alternativo

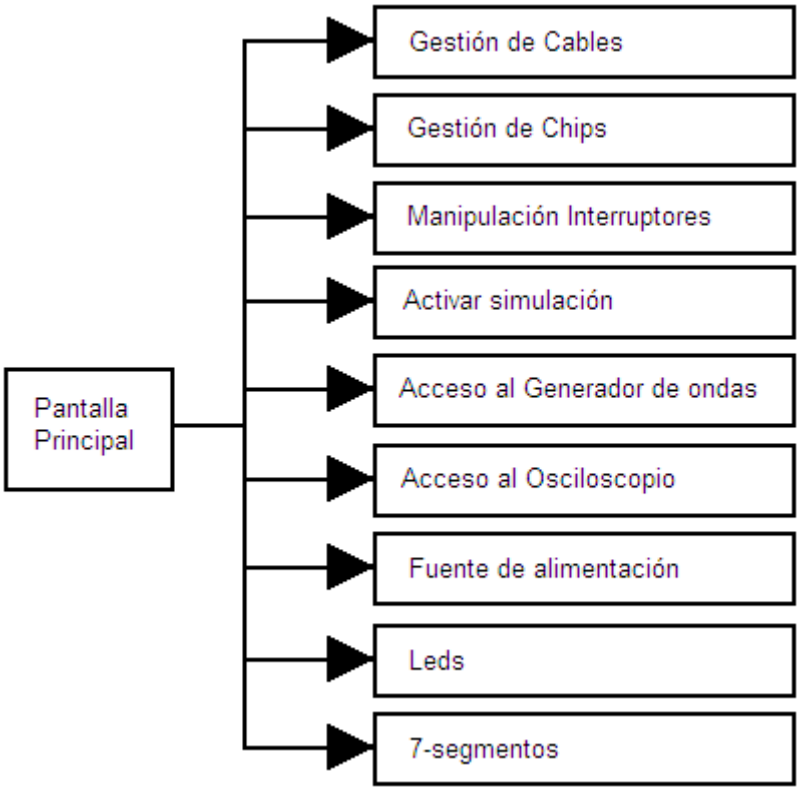
```
if (a>b){
}
else{
}
```

4.2. ESTRUCTURA DEL PROGRAMA

En este apartado se hace un análisis de las diversas opciones que se pueden realizar a lo largo de la ejecución del programa y su estructuración. A continuación, se muestran una serie de grafos que ilustran las diversas opciones del applet "Laboratorio virtual de sistemas digitales" y como están ordenadas. Partiremos de la pantalla principal que es el lugar donde se gestionan las diferentes acciones que implementa el programa.

4.2.1. Pantalla principal

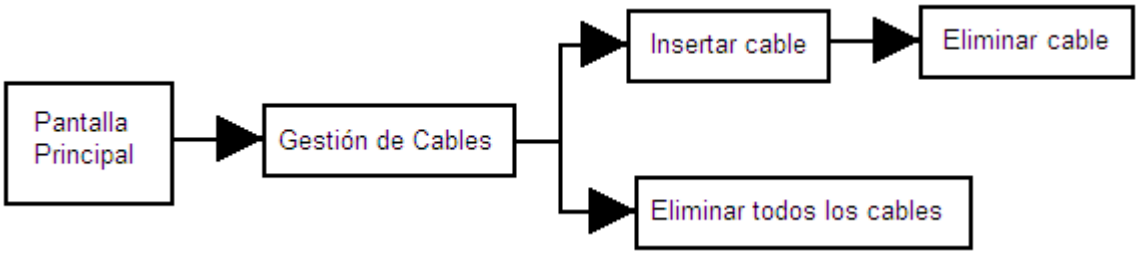
El siguiente grafo muestra las diferentes acciones que podemos realizar desde la pantalla principal del programa. En ella se visualizan la placa, leds, fuente de alimentación, interruptores y siete segmentos. Sobre los diferentes elementos se pueden construir cables y en concreto sobre la placa se pueden insertar chips. La pantalla principal también habilita dos botones para poder acceder al generador de ondas y el osciloscopio.



4.2.1.1. Gestión de cables

La pantalla principal está compuesta por una zona sensible a las acciones del ratón, esta zona afecta a la placa y a los diferentes puertos de conexiones de los componentes del simulador. La zona sensible permite construir cables de acuerdo a la trayectoria que dibuje el usuario con el ratón.

Una vez dibujado un cable, al clicar sobre el mismo con el botón secundario del ratón se podrá eliminar de la placa. Así mismo desde el menú de la pantalla principal se podrán eliminar todos los cables insertados en la placa. El siguiente grafo muestra como se ha estructurado la gestión de los cables:

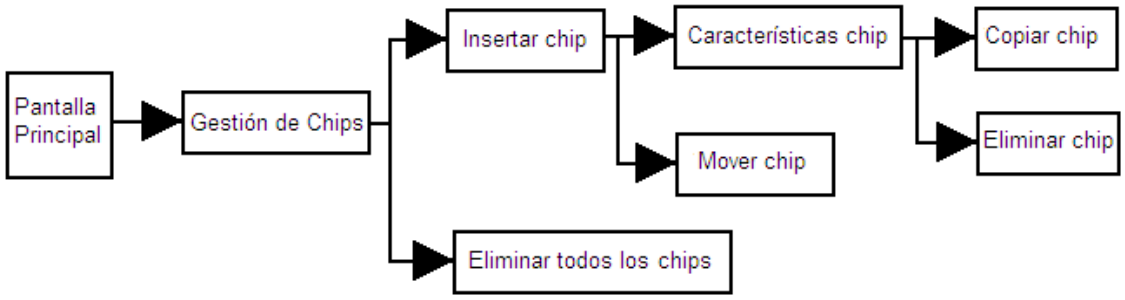


4.2.1.1. Gestión de chips

Desde pantalla principal se pueden insertar chips mediante un botón directo que abre la lista de chips a insertar o accediendo al menú chips que abrirá la misma lista. Una vez insertado un chip en la placa se podrá desplazar a lo largo de la misma sin que se sobreponga sobre otro chip previamente instalado.

Al hacer clic sobre un chip insertado, este parpadeará indicando que se puede desplazar, en caso de clicar el botón secundario del mouse se abrirá una pantalla en la que se permite la copia o eliminación del chip además de la visualización del esquema de conexiones.

Desde el menú chips de la pantalla principal se pueden eliminar todos los chips insertados en la placa. El siguiente grafo muestra como se ha estructurado la gestión de chips.



4.3. POLÍTICA DE GESTIÓN DE CONEXIONES

La colocación física de cables en un circuito digital es un hecho completamente trivial pero extremadamente complejo a la hora de simular esta acción.

La complejidad se divide en dos campos, el primero es el dibujado sobre la placa del cableado y el segundo es la gestión interna de las conexiones que provoca un cable.

La aplicación permite construir el cable de acuerdo a la trayectoria que realiza el usuario con el ratón, esto obliga al programa a controlar el desplazamiento del mouse. Movimientos extremadamente rápidos del ratón impiden la construcción del cable por lo tanto el simulador tiene que reconstruirlo rápidamente desde las zonas que han quedado incomunicadas.

Como se explica en el apartado 3.2 (fundamentos teóricos de la placa) la placa tiene una serie de comunicaciones internas que interconectan sus conexiones. A la hora de gestionar las conexiones provocadas por la colocación de un cable hay que tener en cuenta las conexiones internas de la placa.

Debido a la complejidad de las conexiones existentes en un circuito digital es necesaria una estructura de árbol binario para poder representar los diferentes componentes conectados. Cada elemento del árbol tendrá dos hijos que representarán los elementos interconectados. Los componentes que se conecten a esos dos elementos interconectados se añadirán al árbol formando una nueva raíz cuyos dos hijos serán el nuevo elemento conectado y el anterior árbol. La figura 4.1 muestra este proceso.

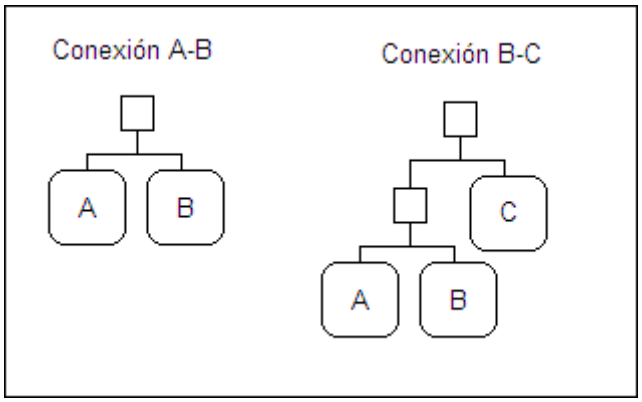


Figura 4.1.Creación árbol de conexiones.

En la figura 4.1 se observa en primera instancia el árbol de conexión correspondiente a la unión mediante un cable de dos componentes (A y B). Posteriormente se ha añadido un cable que ha conectado un nuevo elemento C con el B configurando un nuevo árbol.

Cuando se elimina la conexión habrá que reconstruir el árbol como se muestra en la figura 4.2.

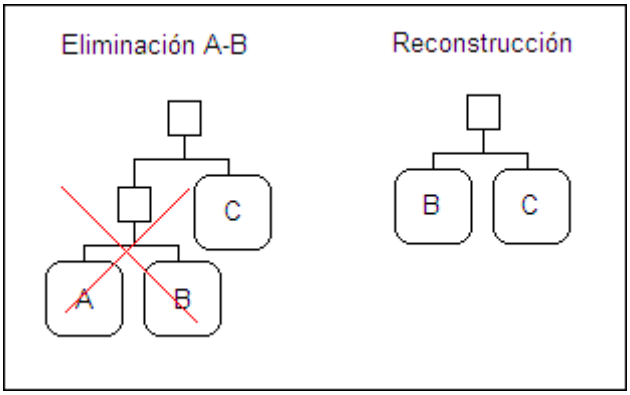


Figura 4.2. Reconstrucción árbol de conexiones

Las diferentes funciones para manipular árboles binarios son necesariamente recursivas para poder hacer una gestión eficaz.

4.4. POLÍTICA DE SIMULACIÓN

Cuando el usuario de la aplicación web pulsa el botón de simulación se analiza el circuito implementado generando una serie de eventos ordenados temporalmente, posteriormente se ejecutan los eventos que darán como resultado la simulación.

El proceso de simulación es el resultado de una serie de acciones ordenadas en el circuito diseñado, cada evento generado indica una de las acciones a realizar.

Para solucionar los diferentes pasos que participan en la simulación se ha implementado el siguiente algoritmo.

- **1:** Analizar los árboles de conexiones creando pares conectados.

- **2:** Analizar aquellos pares en los que participe un chip.

  - **2.1:** Asignar un orden temporal a los pares analizados.

- **3:** Asignar un orden temporal al resto de pares.

- **4:** Crear un vector de eventos a partir de los pares ordenados por su característica temporal.

- **5:** Analizar cada evento del vector ejecutando la acción indicada en el evento.

La asignación temporal dependerá de los elementos que estén conectados, las señales producidas por el generador de ondas, los interruptores y la fuente de alimentación serán las primeras en propagarse. La señales manipuladas por los chips serán las siguientes a analizar y las últimas serán las señales que lleguen a los leds, 7-segmentos y osciloscopio.

4.5. DESCRIPCIÓN DE LAS CLASES Y DE LOS PRINCIPALES MÉTODOS

El laboratorio virtual de sistemas digitales está formado por 49 clases. No es el objetivo de este capítulo explicar las 49 clases y las líneas de código que las componen. No obstante, los siguientes puntos se van a centrar en los asuntos y aspectos más interesantes de la codificación de la aplicación.

4.5.1. Las clases

La empresa Sun Microsystems dedicada en parte al desarrollo de una serie de paquetes de clases en Java, los distribuye libremente desde la red, ofreciendo dos productos JDK (Kit de desarrollo de Java) y SDK (Kit de desarrollo de Software). Ambos productos ofrecen además un compilador de Java o “maquina virtual” para poder codificar, compilar y ejecutar código Java.

A partir de las clases proporcionadas por Sun, se codifica el código para el *applet* “Laboratorio virtual de sistemas digitales”. Las clases que forman la columna vertebral del programa son: *Ariston, AristonMain, PlacaAriston y PlacaAristonPaint.*

Un “applet” es una aplicación o un programa que se ejecuta desde una página web, sin necesidad de instalación. Para ejecutar el programa hay que llamarlo desde el código fuente de la página web. La clase que inicia la ejecución del applet es *Ariston.* Esta clase utiliza los métodos implementados por la clase *JApplet* del paquete distribuido por Sun. Una vez puesto en marcha el applet la clase base es *AristonMain*, gestionará la ventana principal dando paso a las diferentes prestaciones de la aplicación.

A continuación se muestra una descripción detallada del comportamiento de las clases más significativas.

4.5.1.1 La clase Ariston

Nombre	*Ariston.class*
Clase de la cual deriva	*JApplet*
Métodos	*init()* *start()* *stop()* *carregarImatgesGenerador()* *carregarImatgesOsciloscopi()*
Descripción Funcional	Clase invocada desde el código HTML de la página donde se carga el applet. *Ariston* desde su método *start()* crea un objeto de la clase *AristonMain*, clase principal del applet. La clase *Ariston* carga todas las imágenes que utiliza la aplicación, también las de los módulos del Osciloscopio y Generador de Ondas.

4.5.1.2 La clase AristonMain

Nombre	*AristonMain.class*
Clase de la cual deriva	*JFrame*
Métodos	*start()*
Descripción Funcional	La clase *AristonMain* crea un ventana en la que añade una barra de menús desde donde se gestionan las acciones que hacen referencia a cables y chips. *AristonMain* crea un objeto de la clase *PlacaAriston* encargada de insertar los diversos elementos que se visualizan en la pantalla principal.

4.5.1.3 La clase PlacaAriston

Nombre	*PlacaAriston.class*
Clase de la cual deriva	*JPanel*
Métodos	Sólo el constructor.
Descripción Funcional	La clase *PlacaAriston* crea un panel rectangular donde colocará en su zona central un objeto de la clase *PlacaAristonPaint* que gestionará todo lo relacionado con la placa. En las zonas periféricas del panel rectangular se irán insertando diferentes clases derivadas de *JPanel* y que gestionarán el aspecto visual de los leds, 7-segmentos, botón de creación de chips, botón de acceso al osciloscopio, botón de acceso al generador de ondas, e interruptores. La fuente de alimentación estará gestionada desde la clase *PlacaAristonPaint*

4.5.1.4 La clase PlacaAristonPaint

Nombre	*PlacaAristonPaint.class*
Clase de la cual deriva	*Canvas*
Métodos	*paint()* *dibuixarIniCable()* *dibuixarCable()* *dibuixarFiCable()* *eliminarCable()*
Descripción Funcional	La Clase *PlacaAristonPaint* es un lienzo insertado en la zona central del panel de la pantalla principal donde se puede dibujar cables. El lienzo está dividido en 2146 parcelas, cada una de estas parcelas está gestionada por la clase *Parcela* que permite el dibujado de la placa, los puertos de conexiones de los diferentes componentes y los cables. Además la clase *Parcela* guarda otras características como pueden ser: Señal booleana, existencia de chip, o cables que pasan por la parcela entre muchas otras. *PlacaAristonPaint* gestiona el dibujado de los cables en cada parcela mediante un *MouseListener.* Los objetos de las clases *MouseDibuixaCableEstatic* y *MouseDibuixaCableDinamic*, creados desde *PlacaAristonPaint*, están escuchando las posibles acciones que realiza el usuario con el Mouse. Cuando detecta que el usuario quiere dibujar un cable informa a la clase *PlacaAristonPaint* que a su vez informará a la parcela responsable de dibujar el cable. *MouseDibuixaCableEstatic* crea una serie de objetos cuyas clases se encargan de reconstruir el cable cuando las acciones del mouse son demasiado bruscas como para poder dibujar correctamente el cable. Estas clases están gestionadas por un temporizador para chequear a intervalos pequeños de tiempo el estado del cable. Los diferentes métodos "*dibuixar*" ordenan dibujar a las parcelas que correspondan los diferentes tipos de cable, el inicio, la zona intermedia y el final de un cable. Además el método *dibuixarFiCable* informa a un objeto de la clase *ControlConnexionsCable* de las parcelas que conectan el cable dibujado. *eliminarCable()* ordena eliminar un cable determinado a las parcelas responsables del mismo.

4.5.1.5 Las clase MouseDibuixarCableEstatic i MouseDibuixarCableDinamic

Nombre	*MouseDibuixaCableEstatic.class* i *MouseDibuixaCableDinamic*
Clase de la cual deriva	Implementan la interface *MouseListener* i *MouseMotionListener*
Métodos	*MouseDibuixaCableEstatic* : *mouseClicked()* *mouseEntered()*

	mouseExited() *mousePressed()* *mouseReleased()* *MouseDibuixaCableEstatic* : *mouseDraged()* *mouseMoved()*
Descripción Funcional	Las clases *MouseDibuixaCableEstatic* i *MouseDibuixaCableDinamic* están constantemente atentas a los sucesos del Mouse que afectan al lienzo *PlacaAristonPaint* En concreto la clase *MouseDibuixaCableEstatic* gestiona aquellos eventos estáticos del Mouse como son un clic, un doble clic o un clic con el botón secundario del Mouse. Sus métodos capturan esos eventos generados por el usuario. La clase *MouseDibuixaCableDinamic* gestiona el movimiento del mouse sobre el lienzo *PlacaAristonPaint.* Los métodos de ambas clases informan a la clase *PlacaAristonPaint* sobre la acción que quiere realizar el usuario y la parcela del lienzo donde se encuentra el ratón. *MouseDibuixaCableEstatic* llama a los objetos de las clases encargadas del chequeo y reconstrucción (si es necesario) del proceso de dibujado del cable. Además detecta si el mouse se encuentra sobre un chip, en ese caso se desactiva la clase para dar paso a las clases que gestionan las acciones del ratón sobre un chip.

4.5.1.6 La clase Parcela

Nombre	*Parcela.class*
Clase de la cual deriva	Ninguna
Métodos	*setPin()* *getPin()* *setOrientacio()* *getOrientacio()* *setEstatParcela()* *getEstatParcela()* *setChip()* *getChip()* *setVoltatge()* *getVoltatge()* *setComponentParcela()* *getComponentParcela()* *setTipusParcela()* *getTipusParcela()* *estatParcela()* *reconstruirCablesParcela()* *dibuixarIniciCable()* *dibuixarPasCable()*

	dibuixarEsquinaCable() *borrarCable()* *correccioCable()* *historialCasella()* *parcelasNoPin()*
Descripción Funcional	Cada parcela en la que se divide el lienzo *PlacaAristonPaint* crea un objeto de la clase *Parcela.* La clase *Parcela* dibuja las conexiones de la placa y los puertos de los diferente componentes, este tipo de parcela se las etiqueta como parcelas “pin”. Las parcelas que no sean “pins” serán aquellas que no sean conexiones ni puertos de conexión y dibujarán la placa y otros detalles estéticos. Ambos tipos de parcela dibujan cables. Las parcela pin o parcelas de conexión guardarán los componentes del circuito a las que estén asociados y los valores del voltaje. Estos valores serán útiles posteriormente por las clases encargadas de gestionar la simulación. La clase *Parcela* gestiona los cables que pasan por encima de ella, estando capacitada para eliminarlos o mostrarlos en el orden en el que fueron insertados.

4.5.1.7 La clase ControlConnexionsCable

Nombre	*ControlConnexionsCable.class*
Clase de la cual deriva	ninguna
Métodos	*afegirConnexioCable()* *eliminarConnexio()* *buidar()* *getIdCable()* *getIniciCable()* *getFinalCable()* *construirArbresConnexions()* *eliminarConnexio()* *connectarValorsParceles()*
Descripción Funcional	La clase *ControlConnexionsCable* es la más importante en el proceso de simulación del circuito ya que es la clase que interpreta las conexiones del diseño de la práctica. *ControlConnexionsCable* es un vector donde cada elemento representa la conexión de dos puntos efectuada con un cable. A partir de las conexiones registradas en el vector de la clase se crean los diferentes árboles de conexiones explicados en el punto 4.3 que hace referencia a la política de gestión de cables. Para representar los árboles de conexiones se crea desde la clase un objeto de la clase *ArbreConexions*, una vez creado este objeto se almacena en un vector de árboles de conexiones que será

	posteriormente procesado en la simulación del circuito. *ArbreConexions*, es una clase que implementa diferentes métodos recursivos destinados a la gestión de árboles de conexión. *connectarValorsParceles()* es el método que se encarga de implementar el algoritmo expuesto en el punto 4.4 sobre la política de simulación. Este método es llamado cuando se pulsa el botón de simulación i durante el proceso crea diferentes objetos de clases que ayudan a implementar dicho algoritmo. Este método se explica más detalladamente en el punto 4.5.2 dedicado a los principales métodos de la aplicación.

4.5.1.8 La clase MicroChip

Nombre	*MicroChip.class*
Clase de la cual deriva	ninguna
Métodos	*crearChip()* *refrescar()* *refrescarCables()* *dibuixaIntermitencia ()* *dibuixarChip ()*
Descripción Funcional	El objeto de la clase *MicroChip* es creado desde el lienzo *PlacaAristonPaint*, su función es la de gestionar todas las incidencias que pueda tener un chip. Cuando el usuario quiere insertar un chip, desde el botón “Chip” o el menú de la pantalla principal, invoca a un objeto de la clase *Infochip* encargada de mostrar la lista de chips que se pueden insertar. Cuando se selecciona un chip a insertar se notifica al objeto de la clase *MicroChip* el tipo de chip a colocar en la placa. La clase *MicroChip* crea el chip y lo coloca en la placa, para realizar esta operación se ayuda de un objeto de la clase *MicroChipControl* que se encarga de coordinar todos los chips que hay en la placa. *MicroChipControl* gestiona el espacio libre para colocar un nuevo chip, controla el movimiento de un chip a lo largo de la placa y las modificaciones que eso supone, y conecta cada chip con los métodos que implementan las clases específicas del comportamiento interno de cada chip. *MicroChip* crea un método similar al implementado en la clase *PlacaAristonPaint* para coordinar las acciones que el usuario provoque con el ratón sobre un chip específico. *MouseMicroChipEstatic* i *MouseMicroChipDinamic* son dos clases que están escuchando las acciones del mouse sobre un chip, de este modo cuando un usuario quiere mover a lo largo de la placa un chip la clase *MouseMicroChipDinamic* notifica a *MicroChip* de la petición del usuario. *MouseMicroChipEstatic* informa a la clase *MicroChip*, cuando el usuario quiere eliminar, copiar o visualizar el esquema de un chip

	especifico. *MicroChip* implementa los métodos *refrescar() y refrescarCables()* para reconstruir la visualización de los cables que pasan por una parcela en la que se ha insertado un chip. El método dibuxarChip() dibuja un chip en el lienzo *PlacaAristonPaint* y la función *dibuixaIntermitencia ()* realiza una intermitencia sobre un chip dibujado, ayudada de un timer, para visualizar la selección de un chip por parte del usuario.

4.5.1.9 La clase GeneradorOnes

Nombre	*GeneradorOnes.class*
Clase de la cual deriva	*JFrame* Implementa la interfície *Port*
Métodos	generacioOnaCuadrada() connectarComponentParcela() portEntrada()
Descripción Funcional	Para integrar en la aplicación el generador de ondas se han implementado una serie de funciones en su clase principal para poderlo conectar con la placa. Una de las parcelas del lienzo *PlacaAristonPaint* construye un puerto de conexión para el generador de ondas. La interficie *Port* proporciona a la clase *GeneredadorOnes* los métodos para conectar el generador con su puerto de conexión. Los métodos *connectarComponentParcela* y *portEntrada* derivan de la interficie *Port* y su responsabilidad es la de informar a la clase *GeneradorOnes* del puerto que tiene asignado en el lienzo *PlacaAristonPaint.* La función *generacioOnaCuadrada* crea la señal digital de acuerdo a los valores internos del generador de ondas. Esta señal se difunde por el puerto asignado.

4.5.1.10 La clase OsciloscopiMain

Nombre	*OsciloscopiMain.class*
Clase de la cual deriva	*JFrame*
Métodos	*lecturaValorsEntrada()* *gestioSenyalEntrada1()* *gestioSenyalEntrada2()* *reset()*
Descripción Funcional	Para integrar en la aplicación el osciloscopio se han implementado una serie de funciones en su clase principal para poder conectarlo

	con la placa y leer las señales digitales. Dos parcelas del lienzo *PlacaAristonPaint* son puertos de conexión para los dos canales de entrada al osciloscopio. El método *lecturaValorsEntrada()* llama a las funciones *gestioSenyalEntrada1()* y *gestioSenyalEntrada2()* que se encargan de conectar el osciloscopio con los puertos de entrada de cada señal. A continuación, se hace una lectura de la señal digital para que el funcionamiento interno del osciloscopio la procese adecuadamente. *reset()* es un método que restablece los valores originales del osciloscopio que han sido manipulados durante la simulación. Esta función es llamada cuando se para la simulación.

4.5.1.11 La clase PanelSimulacio

Nombre	*PanelSimulacio.class*
Clase de la cual deriva	*JFrame*
Métodos	*simulacioOn()* *simulacioOff()*
Descripción Funcional	El panel insertado en la vetana principal *PlacaAriston* crea un objeto de la clase *PanelSimulacio.* *PanelSimulacio* implementa un botón que activa y desactiva el proceso de simulación. Cuando el botón está activado se llama al método *simulacioOn()* que desencadenará todo el proceso involucrado en la simulación. Si el botón se desactiva, se ejecutará la función *simulacioOff()* que detendrá el proceso de simulación y restaurará los valores que haya modificado la simulación en los componentes de salida.

4.5.2. Los principales métodos

Una vez explicadas las clases principales de la aplicación, a continuación se describen los métodos más interesantes de la codificación.

4.5.2.1 connectarValorsParceles()

El método o función *connectarValorsParceles()* pertenece a la clase *ControlConnexionsCable(),* es invocado al pulsar el botón de simulación de la aplicación e inicia el proceso de simulación del circuito diseñado en la placa.

Analiza los árboles de conexiones creando conjuntos de pares conectados a partir de estos árboles. Crea un objeto de la clase *ParellsConectats* encargada de almacenar información relevante de los conjuntos de pares creados.

La clase *ParellsConectats* implementa una serie de métodos que asignan un orden temporal a los pares creados.

connectarValorsParceles(), crea también un objeto de la clase *GestorEvents.* Esta clase crea una colección de eventos a partir de los pares creados y sus asignaciones temporales.

La clase *GestorEvents* implementa un método, *executarEvents()*, que ejecuta las acciones de los eventos almacenados en la clase. Este método es invocado desde *connectarValorsParceles()* una vez se han creado los eventos.

Finalmente, se llaman a los métodos de los componentes de salida (leds, 7-segmentos y osciloscopio) que leerán los valores digitales de sus puertos de conexiones.

4.5.2.2 construirArbresConnexions()

La función *construirArbresConnexions()* pertenece a la clase *ControlConnexionsCable* y gestiona la creación de conexiones entre puntos unidos por un cable. Es invocada cuando se notifica la colocación de un cable desde el método *dibuixarFiCable()* de la clase *PlacaAristonPaint.*

El método busca entre un vector de árboles de conexiones la existencia de alguno de los puntos interconectados por el cable. En caso de no existir se crea un nuevo objeto de la clase *ArbreConnexions*, objeto que se añadirá al vector de árboles de conexión.

Si existe algún punto ya interconectado se añade la nueva conexión al árbol de conexiones encontrado.

4.5.2.3 eliminarConnexio ()

La función *eliminarConnexio()* pertenece a la clase *ControlConnexionsCable* y gestiona la eliminación de conexiones entre puntos unidos por un cable. Es invocada cuando se notifica la eliminación de un cable.

El método busca entre un vector, el árbol de conexiones donde se encuentran los pares de puntos a eliminar. Se reconstruye el árbol de conexiones en el caso que hubiera más elementos interconectados, en caso contrario se elimina el árbol del vector de árboles de conexiones.

4.6. CONCLUSIONES DEL ANÁLISIS DE LA APLICACIÓN

En este capitulo se ha visto primeramente el estilo de codificación siguiendo unas pautas propuestas por Pedro Manuel Cuenca Jiménez en su obra “Programación en Java para Internet”. Siguiendo estas pautas se ha logrado hacer un código más claro y entendible.

Se ha mostrado el esquema general del applet, con una serie de grafos que da una idea de la estructura global de la aplicación. Con una idea clara del

funcionamiento del programa, se han descrito las clases vertebrales implementadas y los procedimientos más interesantes de la codificación.

CAPÍTULO 5:

DISEÑO

5. DISEÑO

En este capitulo se va a describir el diseño de los diferentes elementos del laboratorio virtual de sistemas digitales.

5.1. DISEÑO DE LA APLICACIÓN

Los applets son aplicaciones en Java que se ejecutan desde la red. Hay dos modos de diseñarlos, uno es incrustándolo dentro de la página web desde donde es llamado. Es decir visualizar la aplicación en la página web. El otro método es crear una ventana independiente a la página web y que funcione como una típica aplicación Windows.

La primera forma de diseño limita el potencial del applet, está enfocada al diseño de efectos visuales de una página web: marquesinas desplazándose, publicidad (banners), relojes u otras sencillas aplicaciones. Para dar aspecto de un clásico programa de ventanas es más idónea la segunda forma de diseño.

De este modo el usuario puede mover los diferentes bloques que conforman el simulador (osciloscopio, generador de ondas, y pantalla principal). Poder mover estos bloques es fundamental para que los alumnos puedan visualizar con mayor comodidad aquel componente que estén manipulando en un momento concreto.

5.1.1. Pantalla principal

Al ejecutar el laboratorio virtual de sistemas digitales se crea una pantalla principal donde aparece en su zona central la placa, en las zonas periféricas se sitúan los conjuntos de interruptores, leds y 7-segmentos. También se encuentran los botones que abren el osciloscopio, generador de ondas y activación de simulación. La pantalla principal implementa una barra de menús para acceder a diferentes opciones relacionadas con los chips y los cables.

En la figura 5.1 se muestra la pantalla principal de la aplicación. El diseño de la placa es fiel al modelo de placas más habituales en el laboratorio de sistemas digitales.

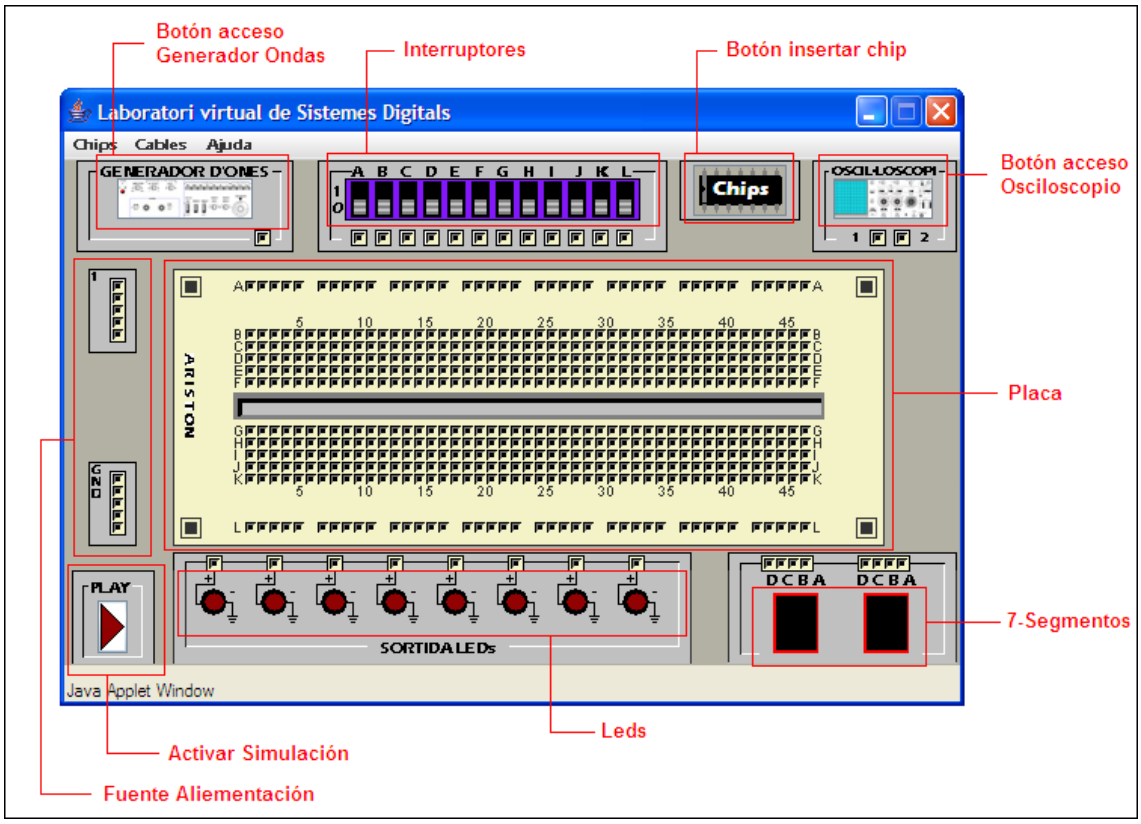


Figura 5.1. Pantalla principal del simulador.

Los botones de acceso al generador de ondas y osciloscopio toman la forma miniaturizada del diseño de ambos instrumentos de modo que se puede identificar su función rápidamente.

La distribución de los diferentes componentes permite una construcción cómoda de los circuitos digitales. Se han agrupado los elementos de entrada en la esquina noroeste y los de salida en la esquina sureste dando la sensación que los flujos de la simulación van de izquierda a derecha y de arriba a bajo.

5.1.2. Ventanas secundarias

De la aplicación pueden surgir diversas ventanas secundarias de información de errores, manipulación de chips, creación de chips y eliminación de cables.

En la figura 5.2 se observa la ventana de eliminación de cable, ésta aparecerá al clicar con el botón secundario sobre el cable a eliminar.

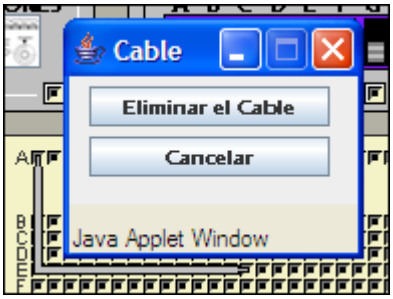


Figura 5.2. Ventana de eliminación de cable.

Del mismo modo que en los cables, al clicar sobre un chip con el botón secundario aparece una ventana (figura 5.3) de información relevante sobre el chip. También muestra el esquema del funcionamiento interno.

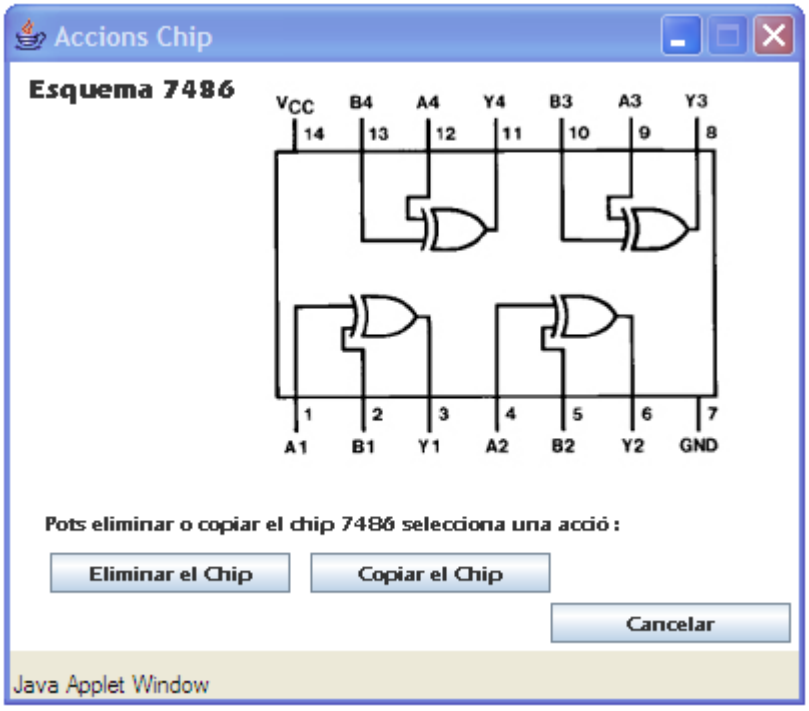


Figura 5.3. Ventana de información de chip.

Cuando el usuario quiere insertar un chip en la placa se muestra la pantalla de la figura 5.4 con los chips posibles a insertar en la placa ordenados de acuerdo a su función.

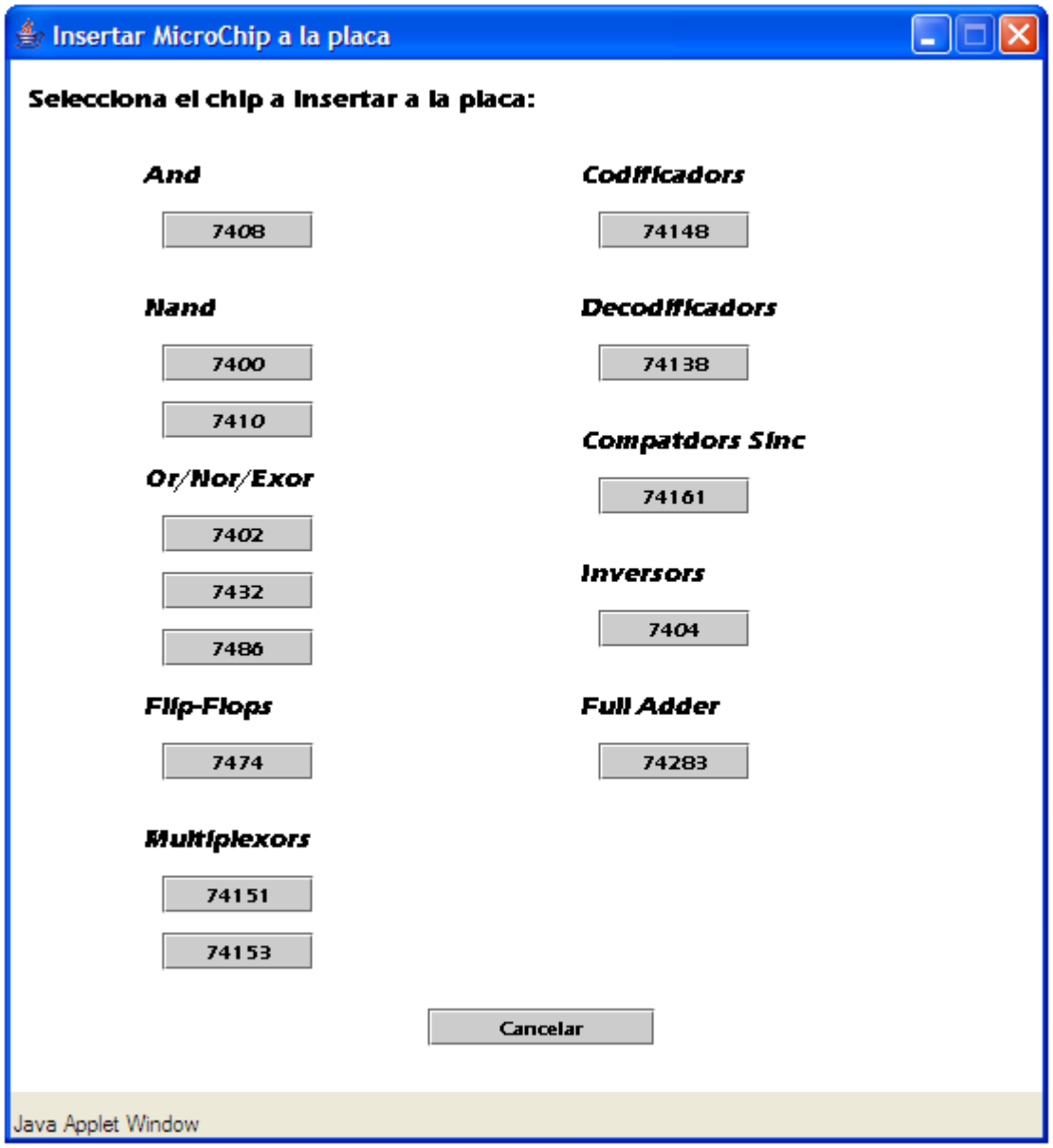


Figura 5.4. Ventana de inserción de chips.

Cuando el usuario intenta realizar una acción no permitida en el laboratorio virtual de sistemas digitales aparecerá una pantalla en fondo rojo o rosada informando sobre la incidencia provocada en el programa. La figura 5.5 muestra las diferentes ventanas de información de error.

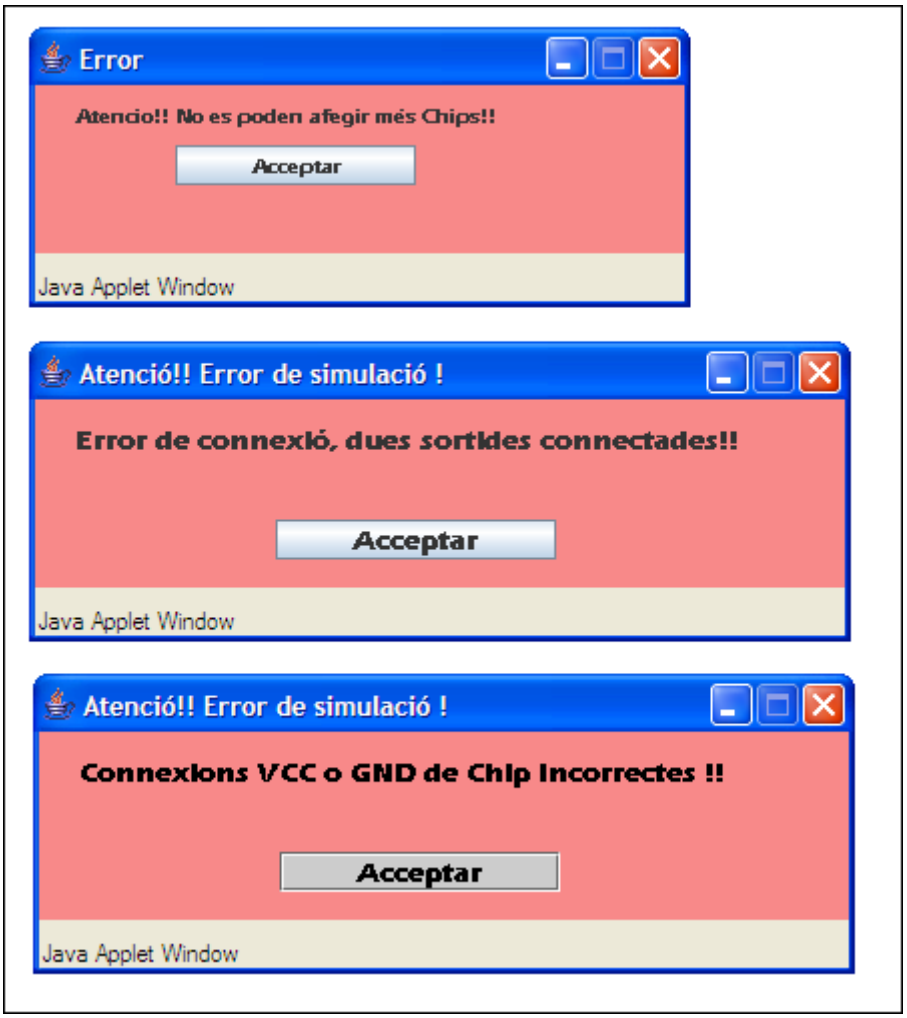


Figura 5.5. Ventanas de avisos de error.

Se ha seleccionado el catalán como la lengua vehicular de la aplicación por ser la propia y natural de Cataluña, además por su raíz latina, la hace altamente íntercomprensible con otras lenguas de su mismo origen (castellano, francés, italiano, gallego o portugués.).

El tipo de letra seleccionado es *Eras Bold ITC* por criterios estéticos y por su fácil lectura.

5.1.3. Generador de Ondas

El diseño del generador de ondas no pertenece al desarrollo del proyecto. No obstante y como ya se ha expuesto en la sección 3.6.3 sobre los fundamentos teóricos del generador de ondas, el diseño del mismo se inspiró en el modelo Tektronix CFG250 que es el más habitual en el laboratorio de prácticas.

5.1.4. Osciloscopio

De igual manera que el generador de ondas, el diseño del osciloscopio no pertenece al desarrollo del proyecto de laboratorio virtual de sistemas digitales. Como ya se explica en el apartado 3.5.4 sobre los fundamentos teóricos del

osciloscopio, el diseño del mismo es fiel al modelo Tektronix 2205 mayoritario en el laboratorio de la materia de sistemas digitales.

5.2. CONCLUSIONES DEL DISEÑO

A lo largo del capítulo 5 se han descrito las diferentes características relacionadas con el diseño del proyecto, una aplicación inspirada en el clásico entorno de ventanas.

El applet crea una ventana principal y a partir de ésta el usuario podrá acceder a todas las prestaciones que ofrece la aplicación.

CAPÍTULO 6:

MANUAL DE USUARIO

6. MANUAL DE USUARIO

En el capítulo 6 se describe el manual de funcionamiento a nivel de usuario del applet “Laboratorio virtual de Sistemas Digitales”. Es recomendable su lectura para que el alumnado de la asignatura pueda manejar correctamente la aplicación. No obstante se ha intentado que el programa sea lo más intuitivo posible.

6.1 EJECUCIÓN APPLET JAVA

La gran ventaja de un applet es que no necesita ningún tipo de instalación, es una de las maneras más sencillas de poner en funcionamiento una aplicación informática. El usuario sólo tiene que acceder a la web donde se aloje el applet y esperar unos segundos a que arranque.

Nota Importante: Se puede dar la situación de que el usuario no tenga instalado en su browser o navegador la máquina virtual de Java, en ese caso no se ejecutaría el applet. También puede ocurrir que el usuario tenga instalada una máquina virtual antigua que no sea capaz de ejecutar algunas funciones del paquete suministrado por Sun al ser relativamente recientes. Si se detecta alguna de estas anomalías la solución es muy sencilla, desde la página web de Sun Microsystems (http://java.sun.com/) se suministra completamente gratis las últimas actualizaciones de los paquetes JDK o SDK, con las nuevas clases, compilador y máquina virtual que ejecutarán sin problemas el applet “Laboratorio virtual de sistemas digitales”.

6.2 MENÚS

Al ejecutarse el applet se abre una ventana principal, en la parte superior se encuentra la barra de menús. Desde la barra de menús se accede a funciones relacionadas con el manejo de los chips y los cables.

1. Chips
   Chip: Abre la pantalla para seleccionar un chip a insertar.
   Esborrar els chips: Elimina todos los chips insertados en la placa.

2. Cables
   Esborrar els cables: Elimina todos los cables de la placa.

3. Ajuda
   Informació: Muestra información relevante sobre la aplicación.

La figura 6.1 muestra la barra de menús con todas las opciones desplegadas.

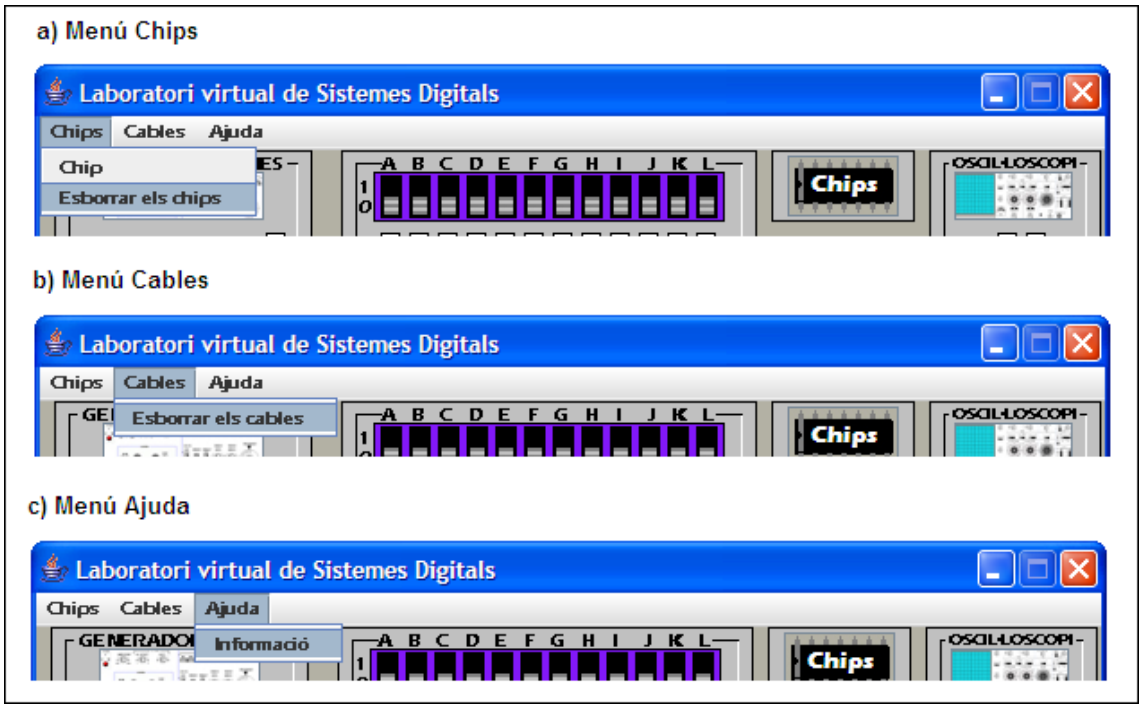


Figura 6.1. Menús de la aplicación.

6.3 NAVEGACIÓN PANTALLA PRINCIPAL

6.3.1. Creación y manipulación de cables

La zona de la aplicación de la figura 6.2 marcada con un recuadro rojo es sensible a las diferentes acciones del mouse y sobre esa zona se podrán dibujar cables. Se aconseja no hacer movimientos extremadamente bruscos con el mouse en el momento de dibujar la trayectoria del cable.

Un cable sólo puede iniciarse y terminarse en una conexión de la placa o un puerto de conexiones de cualquiera de los componentes de la aplicación.

Al mantener pulsado el botón principal del mouse sobre una conexión o puerto de conexiones comenzará la construcción del cable que dibujará la trayectoria que siga el ratón.

En el momento de soltar el botón principal del mouse se terminará el proceso de dibujado del cable. Si se ha soltado el botón en una zona donde no hay ninguna conexión o puerto de conexiones no se dibujará el cable.

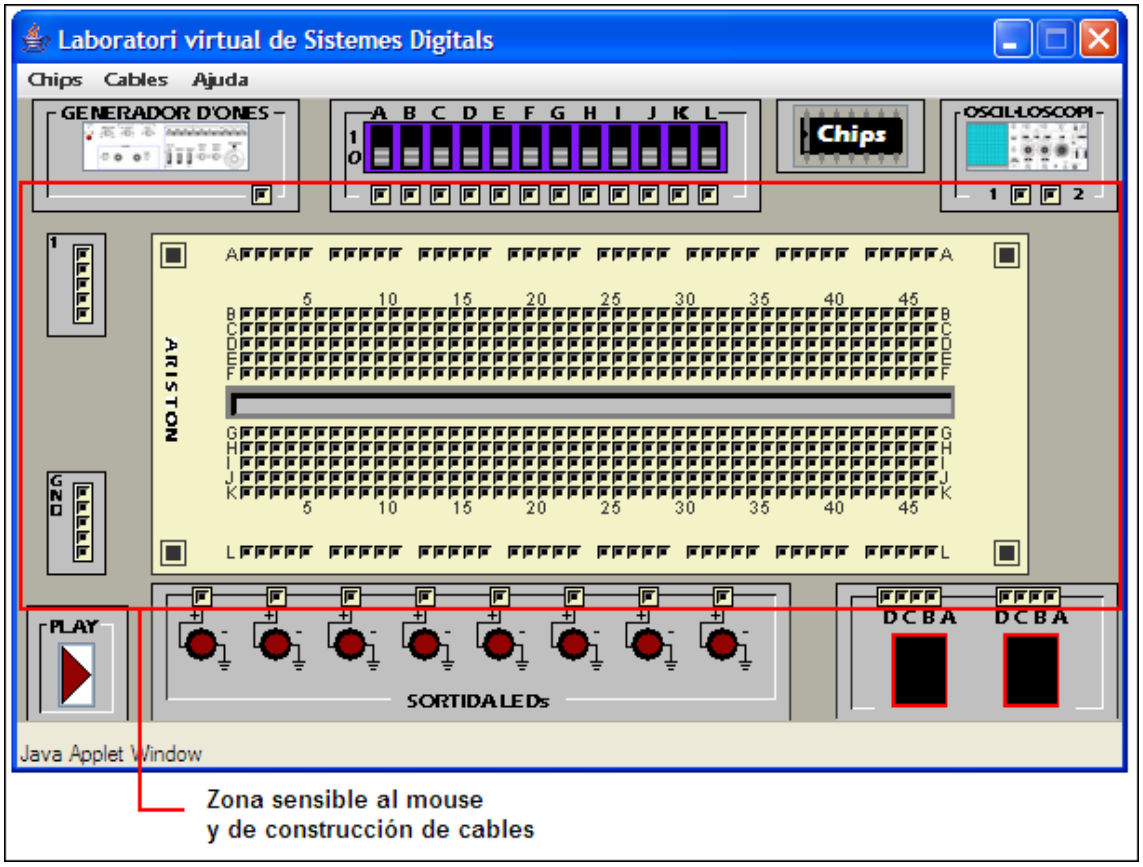


Figura 6.2. Zona de construcción de cables.

Para eliminar un cable hay que posicionarse con el ratón sobre el mismo y hacer clic con el botón secundario, en ese momento aparece una ventana (Figura 6.3) para confirmar la eliminación del cable. Al hacer clic en el botón "Eliminar el cable" se borrará el cable.

Desde el menú "Cables" se pueden eliminar todos los cables insertados al hacer clic en la opción "Esborrar els cables".

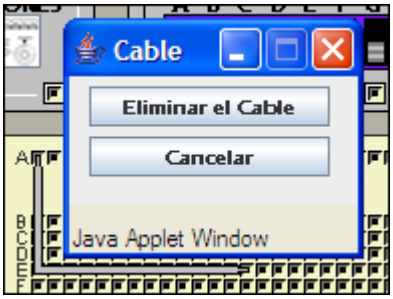


Figura 6.3. Ventana de eliminación de cable.

6.3.2. Creación y manipulación de chips

Para insertar un chip determinado en la placa hay que seleccionarlo entre la lista de chips posibles a colocar.

Hay dos modos de visualizar la lista de chips a insertar en la placa, el primero es desde la opción “chip” del menú principal “Chips” y el segundo pulsando el botón chips situado al lado del conjunto de interruptores.

Una vez se ha visualizado la lista cada botón coloca el chip que indica su etiqueta en el primer lugar libre que encuentre en la placa.

La figura 6.4 muestra los procesos a seguir para insertar un chip.

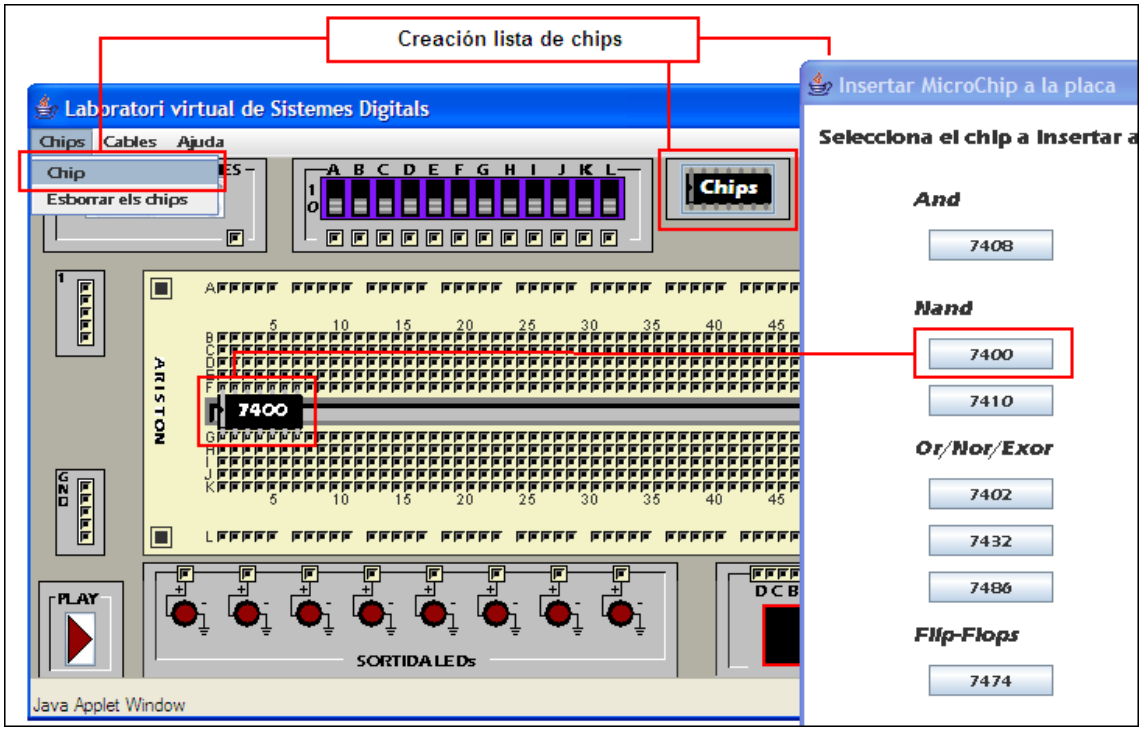


Figura 6.4. Proceso de creación de un chip.

Al clicar con el botón principal del mouse sobre el chip éste comenzará a parpadear con una luz roja, a partir de ese momento todos las acciones que se realicen con el ratón afectaran al chip.

Si se vuelve a clicar sobre el mouse manteniendo pulsado el botón principal se podrá deslizar el chip por los lugares libres que haya en la placa. Al hacer clic con el botón secundario del ratón mientras el chip está parpadeando aparecerá una pantalla de información relevante del chip (figura 6.5). Esta pantalla permite visualizar el funcionamiento interno del chip, copiarlo o eliminarlo.

Mientras el chip está parpadeando si se clica con el botón principal del mouse fuera de la superficie del chip se podrá seguir con el diseño del circuito.

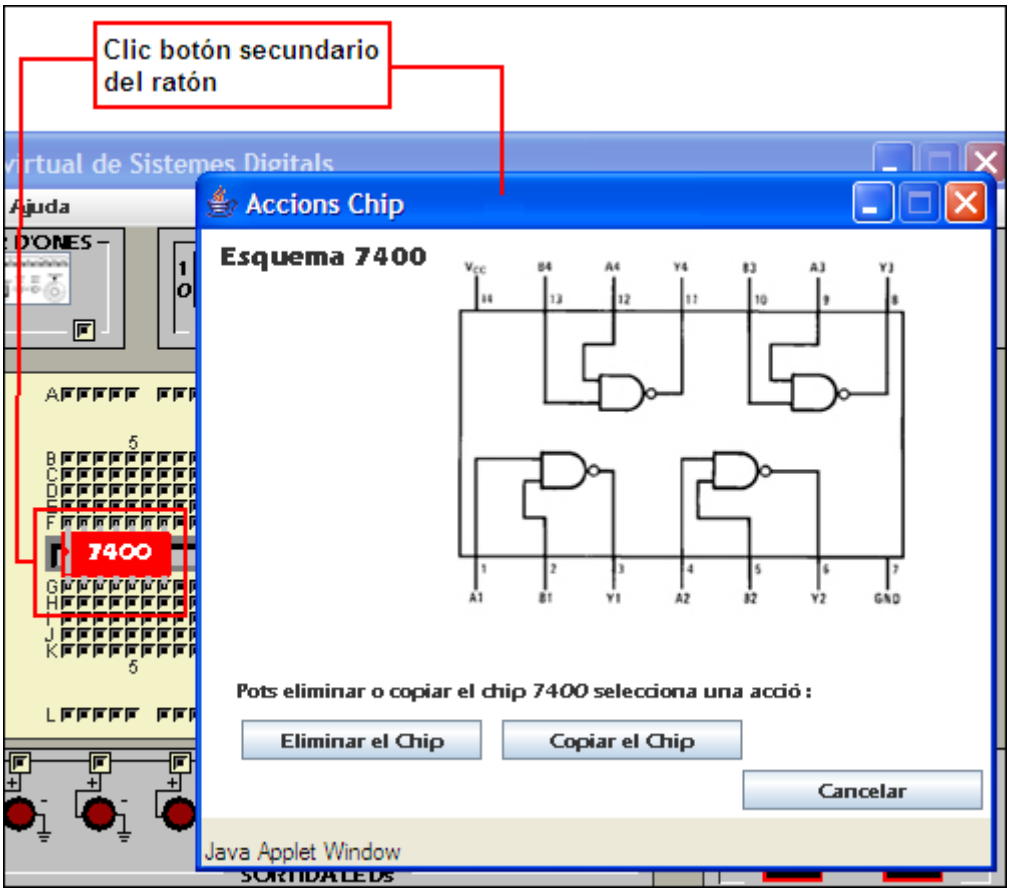


Figura 6.5. Proceso de visualización acciones de chip.

6.4. GENERADOR DE ONDAS

Para generar una señal de pulsos o de reloj hay que abrir el generador de ondas desde el botón de la pantalla principal (Figura 6.6). A continuación, hay que encender el generador de ondas desde su botón POWER. En este momento empezará a generarse una señal de reloj con la frecuencia que este seleccionada en el aparato.

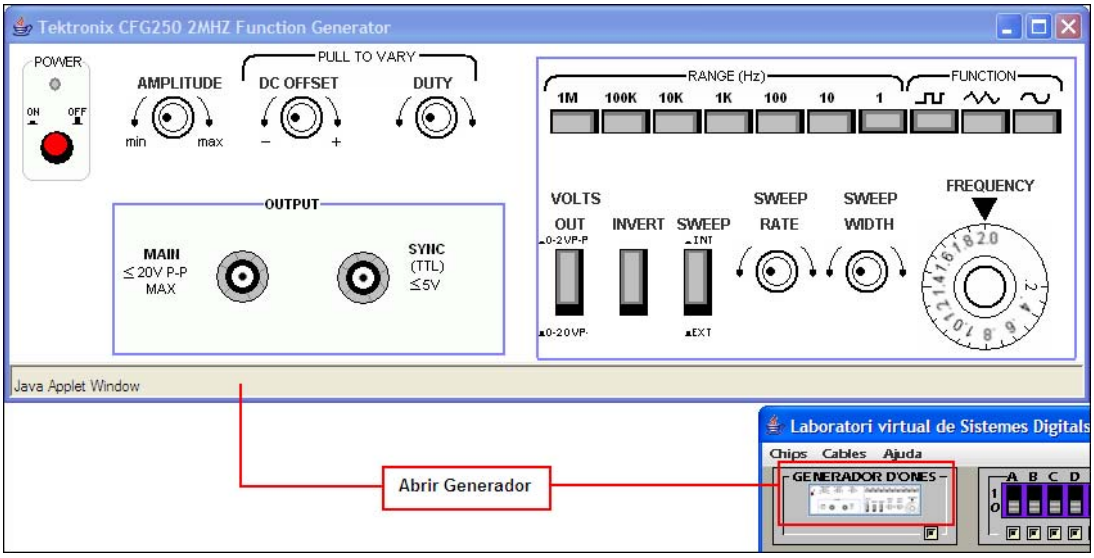


Figura 6.6. Acción abrir generador.

Al cerrar la ventana del generador, este quedará activado pero no visible, si se quiere detener el generador hay que volver a pulsar el botón POWER.

Las únicas acciones del generador habilitadas son las de la serie de botones "range" y la ruleta "frequency", estas acciones seleccionan la frecuencia del generador. La frecuencia seleccionada en los interruptores "range" será multiplicada por el valor de la ruleta "frequency". En la figura 6.7 se muestran los diferentes botones habilitados en el generador.

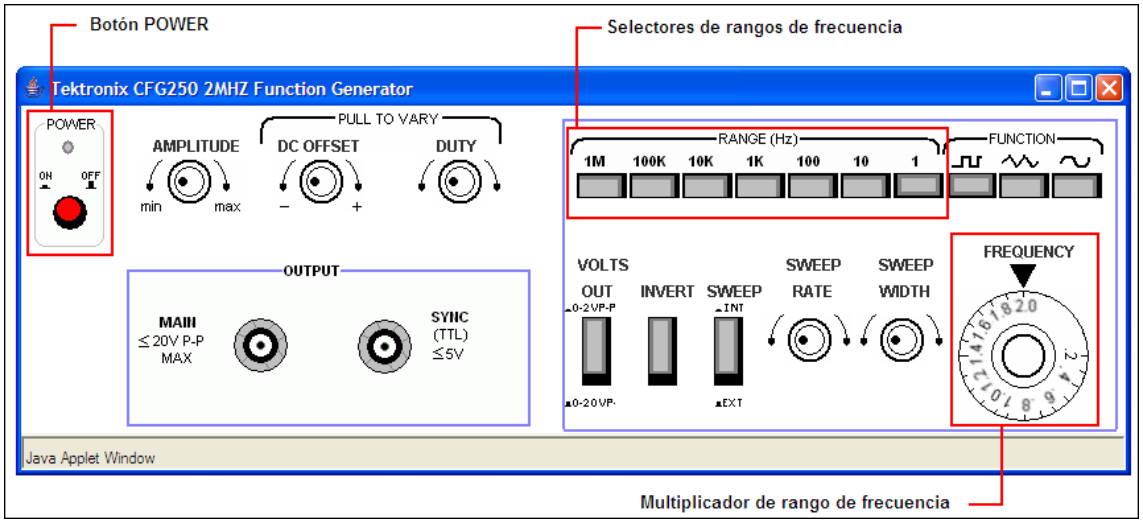


Figura 6.7. Acciones del generador.

6.5. SIMULACIÓN

Cuando se ha finalizado el diseño e implementación del circuito, esto es tener interconectados todos los elementos de entrada, salida y/o chips, se puede iniciar la simulación.

Mediante la activación del botón "PLAY" de la pantalla principal (figura 6.8) se lanzará el proceso de simulación.

El proceso de simulación se detendrá automáticamente si existe alguna acción ilegal en el diseño del circuito. Estas acciones ilegales pueden ser: No tener conectado correctamente un chip a las señales de tierra (GND, señal digital 0) y alimentación (VCC, señal digital 1) o tener interconectadas conexiones de salida de señal digital. En estos casos se notifica al usuario la incidencia ocurrida.

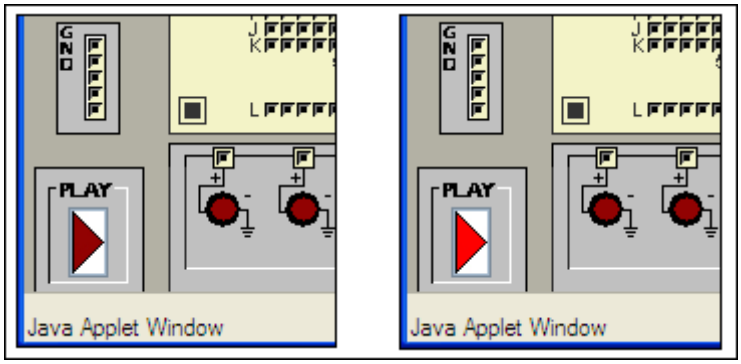


Figura 6.8. Botón PLAY de activación y desactivación de simulación.

Cuando la simulación esté activada se pueden cambiar los valores de los interruptores e incluso eliminar e insertar cables. No obstante es recomendable detener la simulación si se eliminan o insertan cables porque puede ser necesaria en algunas ocasiones resetear valores iniciales.

Nota Importante: Un cambio en la frecuencia del generador durante el proceso de simulación activado no tendrá efecto. Será necesario por tanto detener y reiniciar la simulación. En los circuitos dependientes de la señal de un generador se detendrá automáticamente el proceso de simulación si se desactiva el aparato (Botón POWER).

6.6. OSCILOSCOPIO

Para visualizar una señal digital durante un periodo de tiempo determinado hay que conectarla al osciloscopio. El osciloscopio permite la entrada de dos señales digitales mediante dos canales. Desde el botón “Oscil·loscopi” de la pantalla principal se puede abrir la visualización del mismo además se pueden conectar dos señales digitales mediante los puertos de conexiones que se sitúan en la parte inferior del botón tal y como se muestra en la figura 6.9.

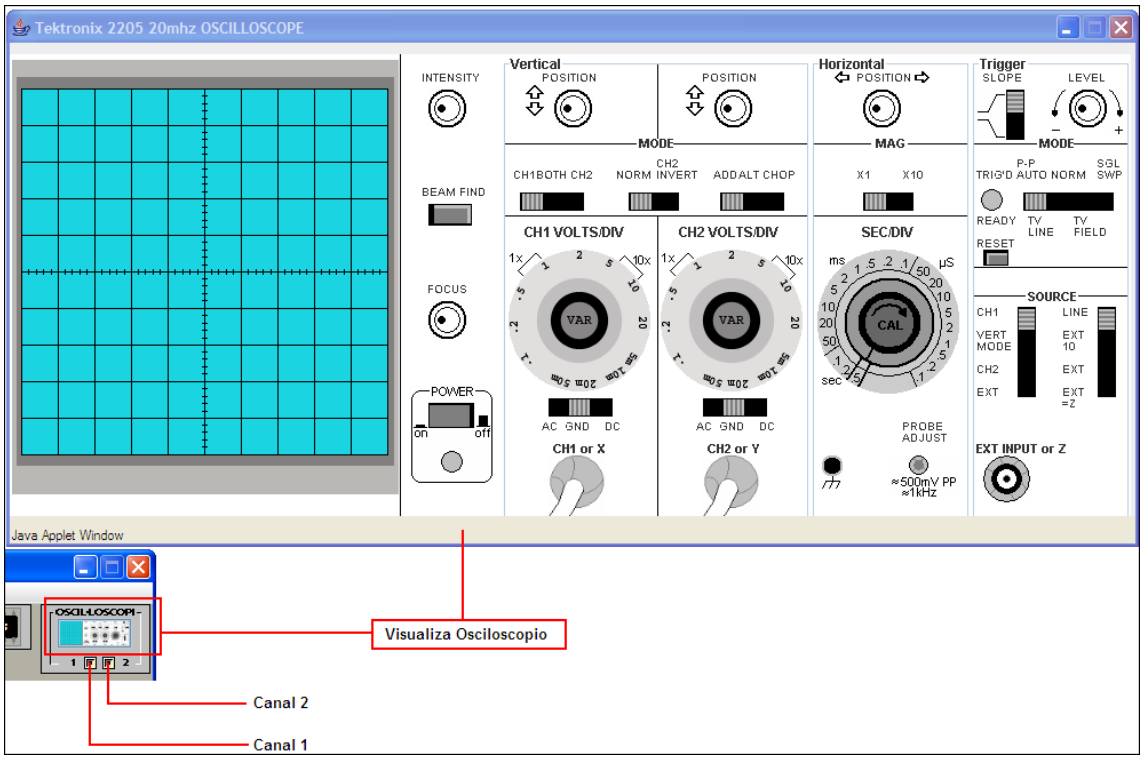


Figura 6.9. Acción visualizar Osciloscopio y puertos de conexiones a los canales

El osciloscopio implementa toda la funcionalidad relacionada con señales digitales, por lo tanto una vez encendido el aparato mediante el botón “Power” hay que colocar el selector de señales del canal en el que esté conectada la señal en posición DC.

Mediante el regulador “Intensity” se puede modificar la intensidad de la senyal que se muestra en pantalla, “Focus” aumenta el grosor de la misma.

Cada canal establece la altura de la señal a mostrar sobre el eje Y mediante los reguladores Vertical Position. Mediante las ruletas “VOLTS/DIV” se puede modificar la medición de los voltajes de la señal introducida en el canal correspondiente.

Los selectores “Vertical Mode” permiten realizar una serie de diferentes modificaciones en la visualización de las señales digitales:

- **CH1:** Muestra en pantalla únicamente la señal conectada en el canal1.

- **CH2:** Muestra en pantalla únicamente la señal conectada en el canal2.

- **BOTH:**

  - ADD: Muestra la señal resultante de la suma de las señales del canal 1 y 2. Siempre y cuando ambos canales estén en la misma posición “VOLTS/DIV”.

- CHOP: Muestra simultáneamente las señales de los canales 1 y 2 siempre y cuando el selector “Trigger Source” este en posición “CH2”.

- **CH2 INVERT:** Invierte la señal del canal 2.

Los comandos “Horizontal” modifican las características, relacionadas con el tiempo, del eje X en la visualización de las señales digitales. El medidor “position” desplaza la señal digital sobre el eje X, el selector “MAG” realiza un efecto “lupa” sobre la señal aumentando o disminuyendo por 10 la medición sobre el eje X.

La ruleta “SEC/DIV” permite realizar modificaciones en la medición de la señal sobre el eje horizontal.

El selector “Trigger Slope” permite negar la visualización de las señales introducidas.

En la figura 6.10 se indican las acciones implementadas en el osciloscopio.

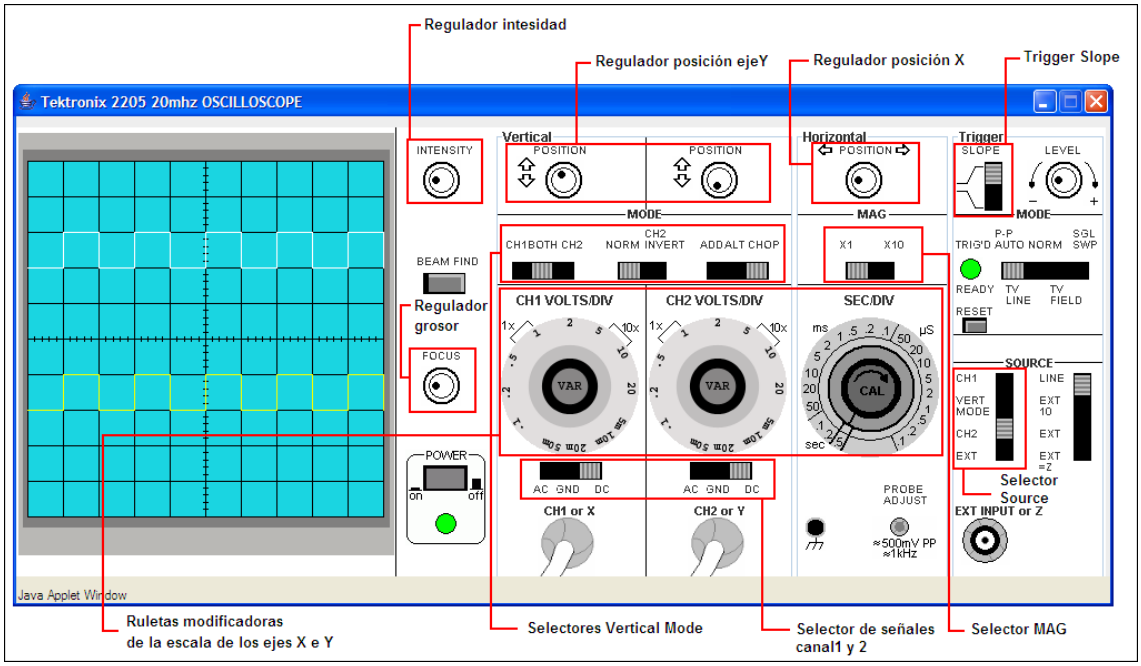


Figura 6.10. Funciones Osciloscopio

6.7. LEDS Y 7-SEGMENTOS

Los leds y 7-segmentos interpretan una señal digital en un instante determinado de tiempo. En la implementación real del circuito digital hay que proteger las conexiones de ambos componentes de salida mediante la colocación de resistencias, en el simulador se ha obviado el uso de resistencias para simplificar el diseño del circuito. Tampoco es necesario realizar las conexiones a tierra (GND) ni el uso del chip 7447 en la manipulación de los 7-

segmentos tal y como se explica en el apartado 3.6.2 del capítulo de fundamentos teóricos.

6.8. CONCLUSIONES MANUAL DE FUNCIONAMIENTO

En el capítulo 6 se ha explicado el funcionamiento completo del applet “Laboratorio virtual sistemas digitales”. Estas “instrucciones” son las necesarias para que el usuario disfrute de todas las prestaciones del applet.

En el capítulo se explica algunas consideraciones previas a la ejecución del applet, como son la de tener la máquina virtual de Java en su versión más moderna funcionando en el navegador del usuario.

Seguidamente se explica como manipular la barra de menús de la pantalla principal y como utilizar las diferentes opciones que permiten el diseño y simulación de un circuito digital.

CAPÍTULO 7:

INTEGRACIÓN Y TEST

7. INTEGRACIÓN Y TEST

En este capítulo se van a describir las pruebas realizas para asegurar un funcionamiento cercano al óptimo del laboratorio virtual de sistemas digitales. Se han testado individualmente las diferentes acciones que implementa la aplicación y se ha comprobado su funcionamiento global.

7.1. PRUEBA DE LA CREACIÓN Y MANIPULACIÓN DE CABLES

Pocos simuladores de circuitos digitales permiten construir cables a medida que el mouse va dibujando una trayectoria. La mayoría de aplicaciones solucionan la colocación de cables exigiéndole al usuario un clic con el mouse cada vez que quiera realizar un cambio de sentido en el dibujado del cable. Esto provoca una implementación sencilla en la codificación de la gestión de la visualización del cable, pero un funcionamiento poco intuitivo y tedioso al usuario.

El problema de la construcción de los cables según la trayectoria trazada por el ratón radica en la velocidad de movimientos de dicho ratón, a mayor velocidad de desplazamiento mayor probabilidad de que la aplicación no pueda seguir la trayectoria del ratón y por tanto deje sin dibujar zonas por las que pasa el cable.

Para solucionar esa problemática se han implementado una serie de clases que observan la trayectoria del mouse y el cable dibujado, en el caso de que no coincidan se reconstruyen aquellas zonas por las que se debería visualizar el cable.

Después de realizar un número elevadísimo de construcciones de cables se han obtenido los siguientes resultados ordenados según la velocidad del mouse:

Movimientos del mouse	**Visualización del cable**
Velocidad normal, movimientos suaves.	100% Correcta.
Velocidad elevada, sin cambios instantáneos de trayectoria.	99% Correcta. 1% reconstrucciones de trayectoria imprecisas.
Velocidad normal, con cambios instantáneos de trayectoria poco frecuentes.	99% Correcta. 1% reconstrucciones de trayectoria imprecisas
Velocidad elevada, con cambios instantáneos de trayectoria poco frecuentes.	95% Correcta. 2,5% reconstrucciones de trayectoria imprecisas 2,5% Sin reconstrucción de cable.
Velocidad normal, con cambios instantáneos de trayectoria frecuentes.	90% Correcta. 10% Sin reconstrucción de cable.
Velocidad elevada, con cambios instantáneos de trayectoria frecuentes.	85% Correcta. 10% Sin reconstrucción de cable. 5% reconstrucciones de trayectoria imprecisas.

Después de las pruebas realizadas se extrae como conclusión que con un comportamiento normal por parte del usuario será muy difícil que detecte anomalías en la visualización de los cables. La única manera de visualizar incorrecciones en el cable es provocarlas, realizando movimientos bruscos y absurdos con el ratón.

Las posibles anomalías en la visualización del cable no afectarán en ningún caso al proceso de simulación ya que lo importante es conocer los puntos que interconecta el cable.

7.2. PRUEBA DEL FUNCIONAMIENTO DE LOS CHIPS

Cada chip tiene que configurar una ecuación lógica de acuerdo a su funcionalidad, además los chips implementados tienen 14 o 16 patas de conexiones. Se han realizado pruebas individuales a cada chip, comprobando todas y cada una de sus patas y el funcionamiento interno.

A partir de las pruebas realizadas se puede garantizar que todos los chips funcionan correctamente.

A la hora de insertar chips sobre la placa hay que controlar que no se inserten más de los que permiten las medidas de la placa. Después de las pruebas realizadas también se puede garantizar que en la placa sólo se insertarán chips cuando haya espacio suficiente para ello.

Al desplazar un chip a lo largo de la placa éste deberá de permitir siempre la visualización de los cables que estuvieran previamente insertados en el nuevo espacio a ser ocupado por el chip. Además un chip no deberá colocarse en el espacio ocupado por otro chip.

Las pruebas realizadas garantizan que el funcionamiento de la movilidad de los chips sobre la placa es correcto, nunca invaden el espacio de otro chip y siempre permiten la visualización de los cables.

7.3. PRUEBA DE LAS CONEXIONES ENTRE LOS DIFERENTES COMPONENTES DEL SIMULADOR

Esta prueba está de por si implícita en las anteriores pruebas de los puntos 7.2 y 7.3, y en las posteriores 7.4 y 7.5. Si las conexiones ente los diferentes componentes del simulador fallan, fallan la mayoría de todas las prestaciones de la aplicación.

Durante las pruebas del funcionamiento de los chips se comprueban simultáneamente la correcta conexión entre los componentes de entrada y los chips y los componentes de salida y los chips. Las pruebas realizadas garantizan una correcta interconexión entre los chips y cualquier otro elemento de la aplicación web.

A lo largo de las pruebas realizadas en los puntos posteriores (donde se controla el correcto funcionamiento de los componentes de entrada y salida), no se ha detectado ninguna anomalía entre la conexiones realizadas.

7.4. PRUEBA DEL FUNCIONAMIENTO DE LOS COMPONENTES DE ENTRADA

Se han realizado diversas pruebas para controlar la correcta generación de señales digitales por parte de los interruptores, generador de ondas, y fuente de alimentación (Salida digital a 1 y Salida digital a 0).

De acuerdo a las pruebas realizadas se puede garantizar que:

- **Interruptores**: Todos generan correctamente señales lógicas a 1 y 0 de acuerdo a sus posiciones seleccionadas.

- **GND**: Los 5 puertos de conexiones asociados a la etiqueta “GND” producen siempre señales digitales a 0.

- **1**: Los 5 puertos de conexiones asociados a la etiqueta “1” producen siempre señales digitales a 1.

- **Generador de ondas**: La señal testeada del puerto de conexiones del generador de ondas siempre se corresponde con la esperada de acuerdo a la frecuencia seleccionada en el instrumento.

7.5. PRUEBA DEL FUNCIONAMIENTO DE LOS COMPONENTES DE SALIDA

Los componentes de salida del simulador interpretan de modo visual una señal digital. Se han realizado pruebas individuales y colectivas para comprobar la correcta interpretación de las señales digitales entre los diferentes componentes de salida.

Las pruebas individuales constan de un único chequeo sobre un componente de salida determinado (led, 7-segmentos, osciloscopio). Las pruebas colectivas constan de diversos chequeos de una misma señal sobre los diferentes elementos de salida, esperando a que los resultados en todos ellos sean equivalentes.

Todos los chequeos han tenido resultados positivos al 100%, por lo tanto se puede garantizar un comportamiento correcto de los elementos que interpretan señales digitales.

7.6. CONCLUSIONES INTEGRACIÓN Y TEST

Es imposible garantizar un funcionamiento perfecto de cualquier aplicación informática pero con las pruebas realizadas se puede afirmar que la aplicación implementada va a tener un comportamiento muy cercano al óptimo.

Las diferentes pruebas practicadas en la aplicación tienen relación entre sí, por lo tanto un mal resultado de alguna de ellas se refleja en el resto. El correcto resultado de cada una de las pruebas también confirma el buen comportamiento global del applet.

El número de pruebas totales practicado ha sido exhaustivo y se considera suficiente para afirmar que la aplicación del laboratorio virtual de sistemas digitales tendrá un buen comportamiento.

CAPÍTULO 8:

CONCLUSIONES

8. CONCLUSIONES

En esta sección se van a plasmar las conclusiones extraídas una vez se ha concluido el proyecto. Son varias y diferentes las conclusiones generadas por el proyecto, las cuales se pueden clasificar en los siguientes apartados.

8.1. APORTACIÓNES DEL TRABAJO DESARROLLADO

La aportación más importante del proyecto es la de facilitar a los futuros alumnos de la asignatura de sistemas digitales de una herramienta fácil de utilizar y que les ayudará a realizar las prácticas de la materia.

La aplicación implementada puede ser la base de un proyecto más ambicioso, cuyos objetivos vayan más allá de ser un recurso útil para estudiantes de la asignatura de sistemas digitales. En la sección 8.6 se describen las líneas futuras en las que trabajar para que el proyecto pueda interesar a estudiantes, profesores y profesionales relacionados con la electrónica.

A continuación se describen una serie de aportaciones a nivel personal proporcionadas por el desarrollo del proyecto.

El proyecto ha permitido desarrollar y profundizar uno de los aspectos claves en informática, el paradigma de la orientación a objetos. El applet se ha codificado en Java, lenguaje enfocado a la programación orientada a objetos, esto ha permitido entender y comprobar la potencia de este tipo de programación, muy superior a la programación lineal.

Durante el desarrollo de la aplicación también ha sido necesario el uso de la programación recursiva, vista en algunas asignaturas de la titulación. La programación recursiva se caracteriza por permitir una gestión cómoda de estructuras en forma de árbol binario y de ser necesarias pocas líneas de código. Eso si, exige más imaginación que otros paradigmas de programación a la hora de implementar y entender el código escrito.

Java está presente en muchas aplicaciones actuales, el mundo de los applets tan sólo es un pequeño campo de los muchos en los que trabaja. Por citar algunos, Java trabaja en comunicaciones, programación web, programación entornos de ventanas, aplicaciones para móviles, aplicaciones para PDAs,…La codificación del proyecto en Java ha supuesto el privilegio de dominar este lenguaje, lo que supone la apertura de muchas puertas en el campo profesional.

Java está basado en C y C++, su sintaxis es prácticamente la misma y en el caso de C++ su filosofía también es similar, por lo tanto una consecuencia indirecta de la práctica de Java es una mejora en el uso y comprensión de C y C++, lenguajes impartidos en diversas asignaturas de la ingeniería.

En el estudio de viabilidad (Capítulo 2) se explica que el proyecto no tiene fines lucrativos y por lo tanto el desarrollador no recibe ningún beneficio económico. Sin embargo es difícil hacer una estimación del valor económico de los

beneficios conceptuales e intelectuales que se han adquirido gracias al proyecto.

El proyecto también ha servido para aprender a organizar y dividir una gran tarea en módulos más pequeños, y responsabilizarse de tenerlos acabados en los términos previamente estimados. No ha sido algo fácil ya que en algunas ocasiones se ha consumido mucho más tiempo que el inicialmente esperado, pero al final se ha logrado tener la aplicación terminada en el tiempo calculado.

Por último, una de las aportaciones del trabajo realizado es la de conocer con detalle y profundidad el uso y funcionamiento de los diferentes instrumentos y componentes que participan en el laboratorio virtual de sistemas digitales. Componentes e instrumentos de los cuales sólo se tenían unas nociones básicas adquiridas durante el curso de la asignatura de sistemas digitales.

8.2. REPASO DE LOS OBJETIVOS PROPUESTOS

Como se explica en el capítulo 2, dedicado al estudio de viabilidad del proyecto, se había propuesto que el mismo cumpliera los siguientes requisitos:

1. Una aplicación que simulara el laboratorio de la asignatura de sistemas digitales.

2. Una aplicación que se descargara desde Internet sin necesidad de instalación.

3. Un programa multiplataforma y gratuito, intuitivo y fácil en su uso.

4. Un recurso que permitiera a los alumnos probar sus diseños de circuitos digitales ilimitadamente.

El resultado obtenido respeta estos cuatro puntos principales. La aplicación, en su diseño, es lo más fiel posible al aspecto que toman los diferentes elementos que pueden encontrar los estudiantes de prácticas de la asignatura sistemas digitales.

El programa, configurado en forma de applet Java, permite una descarga y ejecución sencilla desde cualquier navegador, y desde los sistemas operativos de Microsoft, Unix o Linux.

Se ha pretendido que el programa fuera sencillo e intuitivo en su uso, y a pesar que son dos aspectos completamente subjetivos, se puede considerar que se han logrado.

Por otro lado, la problemática de los alumnos de tener un tiempo limitado para probar sus diseños se ha mitigado al poderlos implementar todas las veces que deseen y en el momento que quieran.

En conclusión, se han alcanzado satisfactoriamente los objetivos planteados antes de la inicialización del proyecto.

8.3. ASPECTOS ORIGINALES Y CREATIVOS DEL PROYECTO

A lo largo del capítulo 5 se detallan todos los aspectos relacionados con el diseño y aspecto visual del proyecto. La distribución de los diferentes elementos que componen la aplicación se ha realizado en base a su funcionalidad. De esta manera los componentes de entrada se han situado en la parte noroeste de la ventana principal y los componentes de salida en la parte sureste. Con esta distribución se pretende dar la sensación de que los flujos digitales circulan por los circuitos implementados de izquierda a derecha y de arriba a bajo.

Como se ha comentado en varias ocasiones el aspecto visual de la placa, los chips, leds y 7-segmentos ha tratado de ser lo más cercano al aspecto real.

Los aspectos más originales y creativos de la codificación del proyecto han sido los relacionados con la construcción del cable y la simulación.

No se encontró ningún precedente que indicará como implementar el dibujado de un cable de acuerdo a la trayectoria realizada con el ratón. Por lo tanto todas las clases y métodos encargados de controlar la construcción de cables han sido los más laboriosos, originales y creativos de la aplicación.

El proceso de simulación también tiene un alto grado de originalidad, menos que la construcción de los cables ya que la simulación es un campo de la informática muy desarrollado y por lo tanto se pudo encontrar bastante información. No obstante el algoritmo que asigna los tiempos de los diferentes eventos que configuran la simulación exigió un esfuerzo creativo y de investigación.

8.4. OTRAS CONCLUSIONES DE CARÁCTER GENERAL

En líneas generales, la evolución del proyecto se ha ajustado bastante a la planificación prevista desde un principio. La coordinación y seguimiento de la tutora han sido básicos en el desarrollo del proyecto y han permitido su conclusión en los plazos esperados.

8.5. ASPECTOS NEGATIVOS

El plazo estimado para concluir el proyecto se ha cumplido, pero ha habido diversos desajustes en el plan original. Algunas tareas han necesitado de más horas que las previamente establecidas, y otras se han terminado antes de lo que en un principio se pensó.

En el estudio de viabilidad del capítulo 2 se indica que el proyecto tendrá una dedicación diaria de tres horas, en la realidad se han dedicado muchas más. En ocasiones por causa natural de la informática, se suele perder la consciencia del tiempo cuando se trabaja con un ordenador, en otras ocasiones para acabar las tareas en el tiempo estimado se han tenido que dedicar horas extras.

Las fechas de entrega del proyecto en Junio han coincidido con las fechas de exámenes de las asignaturas cursadas en el segundo semestre. Sería positivo que en convocatorias posteriores se tuviera en cuenta este hecho a la hora de elaborar el calendario de proyectos.

8.6. LÍNEAS ABIERTAS

Una vez terminado el proyecto, surgen de él muchas ideas de cara al futuro. Ideas que de ser llevadas a cabo harían del presente proyecto una herramienta mucho más potente y útil para a los estudiantes de la asignatura de sistemas digitales y para otros estudiantes o profesionales relacionados con la informática y la electrónica.

Estas ideas son las siguientes:

1: Realizar un módulo para guardar y cargar circuitos diseñados, esto implicará crear una política de permisos en el applet.

2: Implementar la capacidad de añadir más placas a la aplicación.

3: Dotar a la aplicación de un sistema que permita al usuario crear nuevos chips.

4: Si la aplicación soporta diversas placas, también tendría que facilitar un sistema ilimitado de colocación de leds y 7-segmentos.

5: La simulación sería más fiel a la real si implementará el uso de resistencias.

6: Podría ser interesante implementar el funcionamiento analógico del generador de ondas y el osciloscopio.

CAPÍTULO 9:

BIBLIOGRAFÍA

9. BIBLIOGRAFÍA

9.1 LIBROS

9.1.1 Fundamentos teóricos

- José Mª Angulo Usategui, Javier García Zubía, *Sistemas Digitales y Tecnología de computadores*. Paraninfo,2002.
- Ronald J. Tocci, Neal S.Widmer, *Sistemas Digitales, Principios y aplicaciones*. Pearson Educación, 2003.
- *2205 Oscilloscope. Operator's Manual*. Tektronix.
- *CFG250 Function generator***.** *Operator's Manual.*Tektronix.

9.1.2 Aplicación Informática

- Patrick Niemeyer y Jonathan Knudsen, *Curso de Java*. Anaya Multimedia, 2000.
- Pedro Manuel Cuenca Jiménez, *Programación en JAVA para Internet.* Anaya Multimedia, 1997.

9.2 INTERNET

9.2.1. Fundamentos teóricos

- www.tektronix.com (Fabricante de la mayoría de los componentes que se encuentran en el laboratorio de prácticas).
- www.fairchildsemi.com/ (Fabricante de chips).
- www.onsemi.com/ (Fabricante de chips).
- www.wilbrecht.com/ (Fabricante de chips).

9.2.2. Aplicación informática

- java.sun.com/ (Paquete SDK, entorno de programación en Java, y su API (Definición y uso de las clases proporcionadas en el paquete SDK)).

9.2.3. Estado del arte

Aplicación on-line, simulador de circuitos digitales:

- www.cs.york.ac.uk/netpro/bboard/ (Applet desarrollado en la universidad de York)

Otro tipo de aplicaciones que simulan el comportamiento de circuitos digitales:

- www.yoeric.com/breadboard.htm (Simulador de pago.)
- www.electronicsworkbench.com/index.html (Simulador de carácter profesional y de pago.)

- www.cadence.com/products/orcad/index.aspx?lid=orcad (Simulador de carácter profesional y de pago.)
- www.logicworks4.com/ (Simulador de carácter profesional y de pago.)

Resum: (Català)

El projecte “Laboratori virtual de sistemes digitals” és un simulador de l’aula de pràctiques de la assignatura de sistemes digitals, desenvolupat en Java. Integra tots els components que es troben els estudiants de la matèria quant han de realitzar les pràctiques: placa, xips, cables, generador d’ones, oscil·loscopi, ... Permetent així la implementació i prova de circuits digitals.

També, pretén ser una eina útil pel funcionament de la part pràctica de l’assignatura, sent un recurs gratuït i accessible gràcies a Internet.

Resumen: (Castellano)

El proyecto “Laboratorio virtual de sistemas digitales” es un simulador del aula de prácticas de la asignatura de sistemas digitales, desarrollado en Java. Integra todos los componentes que se encuentran los estudiantes de la materia cuando tienen que realizar las prácticas: placa, chips, cables, generador de ondas, osciloscopio,…. Permitiendo así la implementación y prueba de circuitos digitales.

También, pretende ser una herramienta útil para el funcionamiento de la parte práctica de la asignatura, siendo un recurso gratuito y accesible gracias a Internet.

Summary: (English)

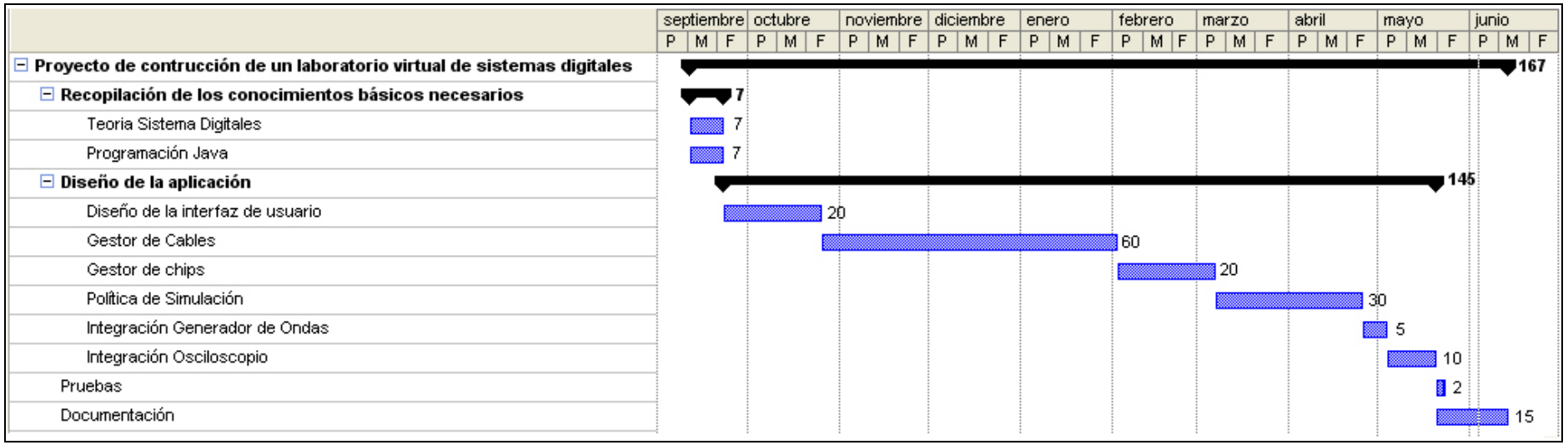
The project “Digital systems virtual laboratory” is a simulator of the practical classroom of digital system’s subject, developed in Java. Compiles all components that students can find when they are doing the practical part of the subject: breadboard, chips, cables, function generator, oscilloscope… It allows digital circuit’s implementation and testing.

Also, tries to be an useful tool to the behaviour of the practical subject’s part, being a free software and accesible by Internet.

Resume: (Galego)

O proxecto “Laboratorio virtual de sistemas dixitais” é un simulador da aula de prácticas da materia de sistemas dixitais, desenvolvido en Java. Integra todos os compoñentes que se atopan os estudantes cando teñen que facer as prácticas: placa, chips, cables, xerador de ondas, osciloscopio,... Permitindo deste xeito a implementación e proba de circuítos dixitais.

Tamén, pretende ser unha ferramenta útil para o funcionamento da parte práctica da materia, sendo un recursos gratuíto e accesible grazas a Internet.



Leyenda:

Duración de tareas compuestas por un conjunto de subtareas

Duración de tareas básicas, no descompuestas en subtareas

Cuadro 2.3. Diagrama de Gannt.

El proyecto y la presente memoria han sido realizados por Marcos Fernández Callejo.

Firmado: Marcos Fernández Callejo.
Viernes, 15 de junio de 2007.

Resum: (Català)

El projecte “Laboratori virtual de sistemes digitals” és un simulador de l’aula de pràctiques de la assignatura de sistemes digitals, desenvolupat en Java. Integra tots els components que es troben els estudiants de la matèria quant han de realitzar les pràctiques: placa, xips, cables, generador d’ones, oscil·loscopi, ... Permetent així la implementació i prova de circuits digitals.

També, pretén ser una eina útil pel funcionament de la part pràctica de l’assignatura, sent un recurs gratuït i accessible gràcies a Internet.

Resumen: (Castellano)

El proyecto “Laboratorio virtual de sistemas digitales” es un simulador del aula de prácticas de la asignatura de sistemas digitales, desarrollado en Java. Integra todos los componentes que se encuentran los estudiantes de la materia cuando tienen que realizar las prácticas: placa, chips, cables, generador de ondas, osciloscopio,…. Permitiendo así la implementación y prueba de circuitos digitales.

También, pretende ser una herramienta útil para el funcionamiento de la parte práctica de la asignatura, siendo un recurso gratuito y accesible gracias a Internet.

Summary: (English)

The project “Digital systems virtual laboratory” is a simulator of the practical classroom of digital system’s subject, developed in Java. Compiles all components that students can find when they are doing the practical part of the subject: breadboard, chips, cables, function generator, oscilloscope… It allows digital circuit’s implementation and testing.

Also, tries to be an useful tool to the behaviour of the practical subject’s part, being a free software and accesible by Internet.

Resume: (Galego)

O proxecto “Laboratorio virtual de sistemas dixitais” é un simulador da aula de prácticas da materia de sistemas dixitais, desenvolvido en Java. Integra todos os compoñentes que se atopan os estudantes cando teñen que facer as prácticas: placa, chips, cables, xerador de ondas, osciloscopio,... Permitindo deste xeito a implementación e proba de circuítos dixitais.

Tamén, pretende ser unha ferramenta útil para o funcionamento da parte práctica da materia, sendo un recursos gratuíto e accesible grazas a Internet.