

# Contenido

1

## Editorial

---

### El Futuro de la Educación en Ingeniería de Cómputo

Miguel Lindig Bos  
Director del CIDETEC-IPN

---

3

### Implementación del Teorema de Cantor en la Solución del Direccionamiento de Memoria Mapeada

Gustavo Abraham Mas Levario (Profesor de la ESCOM-IPN),  
María Aurora Segura Corona, Pablo Manrique Ramírez (Profesores del CIC-IPN)

---

7

### Diseño de Sistemas Digitales Utilizando FPGA

Amadeo J. Argüelles Cruz (Profesor del CIC-IPN),  
José A. Ascencio Román, José F. Villalobos Baigorria (Alumnos del CIC-IPN)

---

15

21

### Uso de Multimedia en la Educación Basica de Ingeniería por Internet

Rodolfo Romero Herrera (Profesor del CIC-IPN)

---

### Desarrollo de un Manejador de Dispositivos Utilizando el Modelo Windows Driver Model (wdm) para Windows 98 y 2000

Rubén Peredo Valderrama, Rubén Aquino (CIC-IPN)

---

26

## ***Editorial***

**L**a importancia del presente número de polibits radica en la diversidad de artículos que la conforman. Así, se incluyen tanto un ensayo (*El Futuro de la Educación en Ingeniería de Cómputo*), donde se analizan los avances en computación y telecomunicaciones y sus efectos sobre la enseñanza de la Ingeniería de Cómputo, como artículos donde se describe el desarrollo e implementación de aplicaciones, lo mismo de hardware (*Diseño de Sistemas Digitales Utilizando FPGA*), que de software (*Desarrollo de un Manejador de Dispositivos Utilizando el Modelo Windows Driver Model (wdm) para Windows 98 y 2000*).

El material restante (*Uso de Multimedios en la Educación Básica de Ingeniería por Internet e Implementación del Teorema de Cantor en la Solución del Direccionamiento de Memoria Mapeada*) presenta propuestas de investigación en los campos de la educación semipresencial y de la arquitectura de computadoras.

Esta diversidad temática responde al interés de polibits de servir como vehículo para la difusión de la computación en su conjunto sin limitarse a un solo tema o enfoque en particular. De la misma forma, nuestro objetivo es publicar colaboraciones provenientes no solo de

las actividades desarrolladas en el Instituto Politécnico Nacional, sino de otras fuentes de investigación y estudio.

Esta es nuestra contribución al avance y difusión de las Ciencias de la Computación en México, reiterando nuevamente una invitación abierta a nuestros lectores y al público en general para que participen tanto con sus colaboraciones como con todas las sugerencias que coadyuben a potencializar este esfuerzo.

# ***El Futuro de la Educación en Ingeniería de Cómputo***

**Dr. Miguel Lindig Bos**  
Director del CIDETEC-IPN

**E**n el nacimiento de la era de la información, las disciplinas académicas consideradas como básicas son las Ciencias de la Computación, la Ingeniería de Cómputo y los Sistemas de Información. La Ingeniería de Cómputo aporta las innovaciones de "hardware". Además de los fundamentos teóricos de la computación, el científico de la computación aporta a las innovaciones de "hardware" los elementos de "software" que permiten su mejor aprovechamiento, y contribuye a establecer un ambiente viable para el usuario. El conjunto de estos elementos constituye las herramientas de los sistemas modernos de información. Dada la creciente complejidad de estos sistemas, aparece un elemento de enlace entre estas disciplinas tradicionales: el ingeniero de sistemas de información, que capta las necesidades informáticas de la organización, para diseñar y desarrollar sistemas que incluyan los aspectos de hardware, software y ambiente de usuario [1].

Los avances tecnológicos recientes en materia de computación y telecomunicaciones han tenido efectos importantes en los métodos de enseñanza en general, y brindan nuevas oportunidades en áreas tales como la educación a distancia. Pero el uso de las técnicas y herramientas proporcionadas por la computación no constituye, por sí solo, una solución

para las nuevas demandas que genera la sociedad de la información a la educación. Si esto es cierto en lo general, es particularmente aplicable a la enseñanza de la computación, como formación para un mercado de trabajo en constante evolución y creciente sofisticación. En este sentido, dos desarrollos tecnológicos en particular, la Internet, por una parte, y los nuevos procesadores de paralelismo explícito, por otra, ya afectan al mercado de trabajo y constituyen elementos a ser tomados en cuenta en la planeación de los programas de estudio del futuro.

---

## **CUESTIONAMIENTOS RECIENTES**

---

En un número relativamente reciente de la revista *Computer* de la *IEEE* (noviembre de 1997) se aborda el tema de la educación superior en general, y el de la computación, en particular. Algunas de las aportaciones expresadas en esta revista se resumen a continuación:

1.- Los elementos de una *Ciencia de la Computación* efectiva [2]. El autor defiende la necesidad de conservar un fuerte núcleo de ciencias de la computación, pero combinado con una visión que enfatice el interactuar con otras disciplinas. Esto es, la búsqueda de problemas que proporcionen una experiencia educativa en la cual la Ciencia de la Computación cons-

tituya un elemento efectivo en un contexto mayor, preferiblemente el mundo real.

2.- El caso para habilidades de cómputo más relevantes [3]. El autor cuestiona la eficacia del desarrollo de sistemas de aplicación que apoyen procesos en empresas, y atribuye la poca eficacia a la falta de habilidades suficientes de los participantes en el desarrollo. En particular, subraya la limitada capacidad de comprender el proceso, para poder ubicar los problemas y las necesidades de mejora. También cuestiona la capacidad de pensamiento sistémico, de cómo interactúan los procesos.

3.- Haciendo la estructura más flexible [4]. El autor sostiene que la educación en cómputo no debe adecuarse solamente a los rápidos cambios de la tecnología de la información, sino también a la forma en que operan los negocios. A manera de ejemplo, sugiere la enseñanza de la interacción en equipos interdisciplinarios. Como medidas concretas, el autor sugiere una mayor interacción con la industria, menores tiempos de adaptación curricular, un mayor énfasis en educación continua y en aspectos pedagógicos de la enseñanza.

Un campo que merece especial interés es el de la Ingeniería de Software. En otro artículo de la revista

citada [5], los autores consideran que la mayoría de los programas de estudio vigentes se concentran en “resolver los problemas de hoy con las tecnologías actuales”, y discuten un proyecto curricular basado en tres componentes fundamentales: la parte central o formativa, la parte aplicada (el desarrollo de sistemas prácticos) y materias de especialización. En síntesis, el proyecto busca mayores niveles de generalización y menor dependencia tecnológica, por medio del uso de modelos formales y, en general, de niveles de abstracción mayores.

Una mayor capacidad de resolver problemas, una menor profundidad en conceptos básicos a cambio de una visión más sistémica, una gran capacidad de comunicación y de trabajo en grupo, un enfoque más interdisciplinario y con mayor orientación a las aplicaciones y, finalmente, una mejor capacidad autodidacta, parecen ser los elementos comunes que encuentran los autores arriba citados y otros, para un mejor diseño curricular. Este tema se retoma por la misma revista en enero de 1998, bajo el título de “Engineering an Education for the Future” [6].

---

#### LOS CAMBIOS TECNOLÓGICOS

---

En dicho artículo, los autores sostienen que la ingeniería de cómputo ha sufrido una serie de cambios de enfoque, como consecuencia de ciertos desarrollos tecnológicos. Como ejemplos mencionan la transición de circuitos discretos a circuitos integrados, y más recientemente, de lógica digital discreta a lógica programable. En un contexto más amplio, puede citarse la transición de la electrónica analógica a la digital, en muchas áreas. Estas transiciones afectan al conocimiento fundamental requerido, a las habilidades necesarias para diseñar

sistemas y, de hecho, a la naturaleza misma de estos sistemas. En términos de una educación para la ingeniería, el cambio tecnológico conduce entonces a varias interrogantes: ¿Qué conocimiento sigue siendo necesario y fundamental? ¿Qué habilidades concretas deben enseñarse? ¿Cuáles son las características de los sistemas por desarrollar?, y ¿Cómo se reflejarán estas características en el currículum?

Además de los cambios tecnológicos reflejados en las transiciones de una tecnología a otra, ejemplificadas anteriormente, un segundo elemento a considerar es el incremento en capacidad, o rendimiento, de la tecnología como función del tiempo. Un ejemplo citado frecuentemente es el incremento en capacidad de cómputo, a costos constantes, de los sistemas durante el pasado reciente. El incremento en capacidad y calidad de la tecnología tiende a reducir las limitaciones en el desempeño de un sistema, para imponer, como nueva limitante, nuestra comprensión incompleta de cómo construir la aplicación. Esto es, factibilidad y costo adquieren una importancia menor, comparados con nuestra limitada capacidad de diseñar sistemas cada vez más complejos.

---

#### UBICACIÓN ACADÉMICA

---

Siguiendo el modelo del artículo [6], y desde una perspectiva histórica, la computación puede considerarse como fundamentada en tres pilares académicos:

- 1.- Ingeniería Eléctrica (teoría de circuitos, circuitos conmutados, dispositivos electrónicos, optoelectrónica, tecnología de procesos, etc.)
- 2.- Sistemas Electrónicos de Información (sistemas de información, co-

municaciones, teoría de colas, procesamiento de señales, etc.)

- 3.- Ciencias de la Computación (complejidad, autómatas, algoritmos, lenguajes, compiladores, bases de datos, etc.)

De la superposición o traslape de las disciplinas anteriores resultan entonces nuevas opciones curriculares. Así, un traslape entre los sistemas electrónicos de información, por una parte, y ciencias de la información, por otra, genera las opciones de multimedia, robótica, visión artificial, sistemas en tiempo real, software concurrente, simulación, redes, etc. Evidentemente, estas opciones son, en gran medida, responsables de la dinámica actual de la computación. Sin embargo, nuestros programas de estudio parecen poco adecuados a las necesidades que generan estas nuevas opciones. Por ejemplo, el estudio de eventos que ocurren de manera irregular en el tiempo generalmente no forma parte de programas tradicionales. Por otra parte, es cuestionable que semejante nivel de profundización en la teoría de sistemas sea viable en el marco de una licenciatura. La cuestión relativa a qué conocimiento sigue siendo necesario y fundamental, no puede contestarse simplemente con base en una apreciación de la cercanía relativa a una u otra de las disciplinas tradicionales.

---

#### FORMACIÓN Y PERTINENCIA

---

Entre los elementos más discutidos de cualquier currículum se encuentran, sin duda, los relativos a la importancia del proceso formativo del estudiante, y la pertinencia de la educación en términos del mercado de trabajo. El eterno conflicto entre las materias formativas y las de aplicación, no admite la solución fácil de incrementar el tiempo de estudio.

Como ya se señaló, la necesidad de una formación básica cada vez más profunda parece ser una de las consecuencias del cambio tecnológico. Claramente, no existe una solución sencilla en términos de una educación tradicional.

Una de las fortalezas de la Ingeniería Eléctrica, pero también de las Ciencias de la Computación, es el modelado formal de sistemas. Pero un modelo constituye un determinado nivel de abstracción; por ejemplo, considérese el diseño de sistemas digitales, en los cuales un circuito se modela como una función booleana. Existen lenguajes de alto nivel para el diseño asistido por computadora de sistemas digitales, basados en este modelo. Así, el diseño de un circuito se convierte en un problema de programación. Los conocimientos de electrónica digital son, de hecho, irrelevantes para diseñar sistemas digitales complejos a ese nivel de abstracción.

De manera similar, la mayoría del software de aplicación se desarrolla hoy en día con base en lenguajes de alto nivel y herramientas, cada vez más poderosas, de desarrollo de software. La complejidad de las aplicaciones simplemente no admite su desarrollo a nivel de lenguaje de máquina. En consecuencia, se usan niveles mayores de abstracción, aunque con ello se sacrifica, posiblemente, eficiencia y velocidad de ejecución.

De lo anterior se desprenden las consideraciones básicas para una posible solución entre las demandas conflictivas de formación y pertinencia. Si se concibe al posgrado como la oportunidad de profundizar en la técnica, en la teoría, y en los conocimientos formales, podemos distinguir entre los niveles de “diseñador” (licenciatura) y “experto” (posgrado). Aquí, el diseñador posee la capacidad de proyectar y desarrollar sistemas

con un alto nivel de abstracción, con base en habilidades de diseño de sistemas y conocimientos interdisciplinarios de aplicación de los mismos. Mayores niveles de complejidad, o de eficiencia de estos sistemas, podrán requerir el concurso del “experto”.

Un corolario de lo anterior es que el conocimiento básico, formativo, adquiere una dimensión distinta. Por una parte, al reducirse el tiempo efectivo dedicado a este conocimiento, los aspectos didácticos adquieren una importancia mayor. Esto es, factores tales como la edad del educando deberán ser tomados en cuenta, privilegiando a aquellas bases formales que, a una edad posterior, resultan de difícil asimilación. Por otra, las materias formativas deberán conducir a mayores habilidades de autoestudio, y establecer el vínculo con estudios a nivel posgrado.

## RETOS ADICIONALES

Entre los cambios tecnológicos recientes sobresalen dos que, posiblemente, constituyan retos adicionales para el diseño curricular. Uno de ellos es la red, esa tecnología de comunicación mundial cuyo potencial, en este momento, apenas se vislumbra. Pero está claro que será un elemento de vinculación entre las telecomunicaciones y la computación, vistas como disciplinas académicas. Además de conferir nuevos niveles de importancia a la telemática y otras disciplinas, es de suponerse que generará nuevas oportunidades en muchos campos de aplicación y, en consecuencia, de planes y programas de estudio. Estrechamente vinculado con lo anterior se encuentran los nuevos desarrollos en materia de arquitectura de procesadores. Las máquinas de paralelismo explícito, esto es, los procesadores que corresponden al modelo computacional SIMD (una instrucción,

múltiples datos) se encuentran ya comercialmente disponibles. Existen pocas herramientas de software para estos procesadores, cuyo potencial de aplicación en campos como la multimedia es, sin embargo, inmenso. Por otra parte, estos procesadores permiten la construcción de multiprocesadores, de máquinas paralelas con varios niveles de paralelismo simultáneos, para las cuales los actuales paradigmas de programación son, simplemente, insuficientes en términos de una explotación efectiva. Se requieren nuevos desarrollos en software, cuyo impacto en los planes y programas de estudio es, en este momento, tema de especulación.

## CONCLUSIONES

En materia de Ingeniería de Cómputo, los cambios vertiginosos de la tecnología han generado nuevas oportunidades, pero también nuevos retos para el diseño curricular. La complejidad creciente de las aplicaciones de la computación tiene el efecto de, por una parte, fragmentar el campo en un número creciente de especialidades y, por otra, incrementar el nivel educativo requerido para un desempeño profesional exitoso. Diferentes autores coinciden en que parte de la respuesta a esta problemática se encuentra en grupos de trabajo interdisciplinarios, y resaltan la necesidad de desarrollar habilidades de comunicación y de trabajo en grupo. Un segundo elemento para una posible respuesta consiste en la aplicación de niveles de abstracción cada vez mayores en el modelado de los sistemas, y un mayor énfasis en el diseño e implementación de los mismos. De acuerdo con esta visión, el posgrado constituye la oportunidad de profundizar en el conocimiento que, a nivel licenciatura, posee un enfoque más orientado a la solución de problemas prácticos. En esencia, se plantea una

solución basada en el estudio de lo general como punto de partida, para llegar a lo particular en el posgrado. En este sentido, el aspecto formativo adquiere un significado e importancia distintos. Además de una mayor amplitud temática con menor profundidad sobre dichos temas, debe constituir el puente hacia el posgrado y obedecer a modelos pedagógicos que incluyan la edad, entre otros factores.

#### BIBLIOGRAFÍA

- [1] Gordon E. Stokes: *"Rethinking the current formula"*, **Computer**, Vol.30, No.11, pp. 48-49, noviembre de 1997.
- [2] Peter A. Freeman: *"Elements of effective computer science"*, **Computer**, Vol.30, No. 11, pp. 47-48, noviembre de 1997.
- [3] Jimmie E. Haines: *"The case for more relevant computing skills"*, **Computer**, Vol.30, No.11, pp. 55-56, noviembre de 1997.
- [4] Joe Turner: *"Making the structure more flexible"*, **Computer**, Vol.30, No.11, pp. 56-57, noviembre de 1997.
- [5] David Garlan, David P. Gluch, James E. Tomaryko: *"Agents of Change: Educating Software Engineering Leaders"*, **Computer**, Vol.30, No.11, pp. 59-65, noviembre de 1997.
- [6] Edward A. Lee, David G. Messerschmitt: *"Engineering an Education for the Future"*, **Computer**, Vol.31, No.1, pp. 77-85, enero de 1998.

# ***Implementación del Teorema de Cantor en la Solución del Direccionamiento de Memoria Mapeada***

*M. en C. Gustavo A. Mas Levario, M. en C. María A. Segura Corona, M. en C. Pablo Manrique Ramírez.*  
Profesores del CIC-IPN

**E**l presente artículo propone una solución descriptiva al problema del mapeo computacional sobre la base del Teorema de Cantor. Para lograrlo se propone una serie de fundamentos heurísticos que permitan relacionar el Procedimiento de Cantor con el significado del direccionamiento de memoria mapeada que rebasa el tamaño del mapa de memoria de los procesadores.

---

## **INTRODUCCIÓN**

---

Este trabajo sigue una línea pragmática principalmente, y hace uso de una serie de asociaciones entre la realidad del direccionamiento de memoria y la abstracción del Teorema de Cantor. El objetivo principal de esta asociación es proporcionar sustento a algo que está fuera del mundo real y que tiene su origen en el universo de los conceptos. El significado de sustentar aquí es materializar un concepto matemático en un campo de acción de la realidad mediante una idea o un pensamiento intuitivo, en este caso el campo de acción es el diseño de sistemas digitales para direccionar memoria mapeada.

Se sabe que en la generalización simple, la abstracción es sólo un instrumento de la operación mental entre otros, con el cual aislamos las cualidades que se desean obtener. Un pragmático debe partir de una condición de entidades con cualidades a una condición de cualidades sin entidades para plantear un problema de ingeniería bajo la abstracción, la precisión y el rigor lógico de ciertos procedimientos matemáticos, como en este caso el Procedimiento de Cantor[2].

Para poder materializar un concepto se debe entender que en la formación de los conceptos obtenidos por generalización simple aplicada sobre determinadas propiedades de los objetos reales o no, tales como el color de nuestra silla o el color de nuestra primera bicicleta, es posible distinguir tres fases de operaciones mentales. En la primera fase la propiedad se define comparando directamente los objetos: como en el caso de ciertas culturas primitivas en donde se asocian tantas manzanas como dedos de la mano. En la segunda aparece un adjetivo: cinco direcciones de memoria, o en casos un poco más abstractos: memoria mapeada, aquí el adjetivo es mapeada. En la tercera fase se abstrae la propiedad de los objetos y pueden aparecer cosas como verde, o el número abstracto 5 e incluso el adjetivo mapeado o mapeada.

Este estudio en sí trata con la segunda fase, ya que en lugar de confrontar dos hechos reales, se usa la técnica de la asociación de un hecho teórico ya estudiado con un hecho real sin estudio, desde la perspectiva de ese concepto. Para comprobar su funcionalidad se aplica la generalización simple a diferentes circunstancias del hecho real, postulando la obtención de su concepto que, como se verá, es el concepto del hecho teórico.

El hecho teórico es el Teorema de Cantor, que describe que la magnitud de una función y la magnitud de su variable siempre son inconmensurables. El hecho real que se desea estudiar es el originado al direccionar memoria mapeada en un nodo de procesamiento con memoria local. De este modo tenemos que la variable independiente es cada dirección absoluta del mapa de memoria del procesador, y su función es el mapeo de su dirección correspondiente sobre la memoria, en un momento dado.

En la actualidad hay sistemas de cómputo dedicados a aplicaciones específicas constituidos por nodos de procesamiento, en donde los procesadores tienen capacidades de direccionamiento limitado, pero que en ciertas aplicaciones requieren almacenar cantidades de datos que desbordan el límite del mapa de memoria de dichos procesadores.

El mapa de memoria de un procesador es el conjunto máximo de direcciones absolutas que puede referenciar desde su unidad de direccionamiento; de este modo, un procesador con 16 líneas de direccionamiento sólo puede alcanzar hasta 65536 direcciones absolutas, pero ciertas aplicaciones de diseño obligan a este procesador a demandar de su unidad de direccionamiento más de estas 65536 direcciones absolutas, que estarán en correspondencia biunívoca con 131072 direcciones mapeadas.

Ese inconveniente exige la existencia de otra línea de dirección en la unidad de direccionamiento del procesador misma que es imposible de obtener, así que es aquí donde se recurre a un procedimiento que permita concebir este problema computacional como un problema a resolver de la matemática aplicada.

Este procedimiento indica que siempre habrá la posibilidad de direccionar de un modo u otro más de lo que aparentemente se permite a partir del mapa de memoria de un procesador.

Se presenta lo anterior como un teorema computacional a demostrar tomando como su lema el Teorema de Cantor.

#### MÉTODO

Lo que se pretende es llevar el plano cartesiano del mapeo al plano de un cuadrado, en donde el origen del plano cartesiano coincide con el vértice inferior izquierdo del cuadrado; de este modo, la función del mapeo se superpone a la diagonal que se traza del vértice inferior izquierdo al vértice superior derecho, el eje de las abscisas se superpone al segmento de la base del cuadrado, y el eje de las ordenadas se superpone al segmento vertical izquierdo del cuadrado.

En la figura 1 se muestra computacionalmente en donde estarían ubicados el lema y el teorema que se desea demostrar. El Teorema de Cantor es el siguiente:

*“Los puntos del plano siguen la potencia del continuo”.*

El método se fundamenta en los tres aspectos[1] de las abstracciones de la matemática, con ellos se formulan las asociaciones necesarias que dan la estructura del procedimiento que se desarrollará en el método:

1. Las formas espaciales, que en este caso son la geometría euclídeana necesaria para diseñar los esquemáticos de un diseño digital, así como los principios necesarios para el plano cartesiano y el cuadrado.
2. Las relaciones cuantitativas del hecho real, o sea la aritmética correspondiente para el nombramiento de las localidades del direccionamiento de un nodo de procesamiento.

3. La sucesión de grados de abstracción creciente, que en este caso en particular son ciertos principios de la geometría analítica.

Con el primer aspecto enumerado anteriormente se traza un bosquejo, en donde las líneas rectas representan a las señales que emite un procesador a las memorias y, en general, a cualquier dispositivo de entrada/salida. Los cuadriláteros representan a las unidades de cómputo como son los procesadores, los mapeadores y las memorias. Este aspecto, en conjunto con el segundo, permiten localizar e identificar cada entidad llamada dirección con el fin de poder nombrarlas con los números reales. En la figura 2 se muestra un diagrama esquemático de un nodo de procesamiento.

Sin embargo, antes de entrar en detalle con relación a los tres aspectos es necesario resaltar la importancia del concepto de los números reales. Los números reales[1] representan un instrumento seguro y poderoso para la investigación matemática de

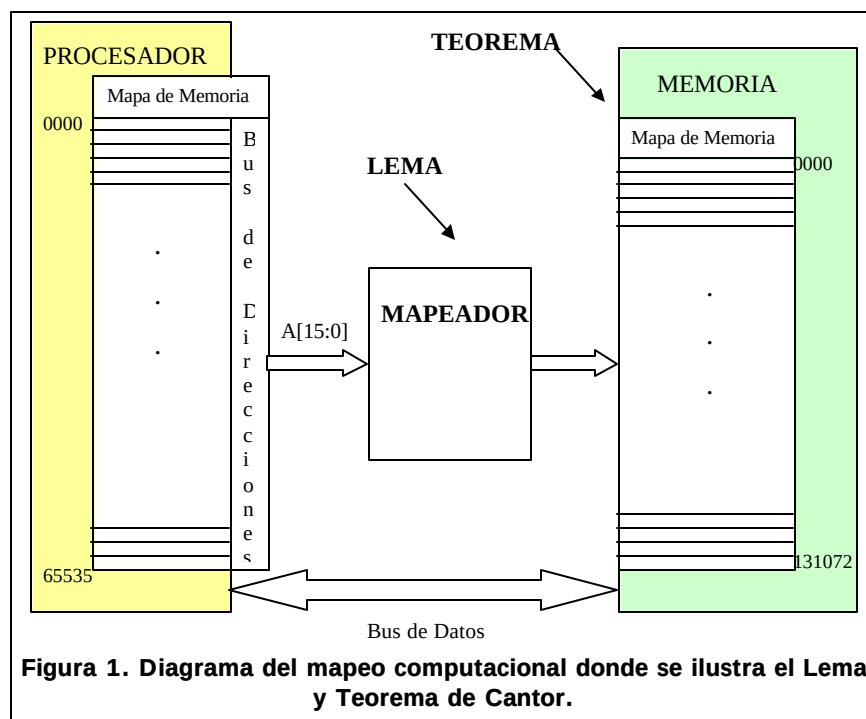


Figura 1. Diagrama del mapeo computacional donde se ilustra el Lema y Teorema de Cantor.

las magnitudes y procesos realmente continuos, tales como las direcciones del mapa de memoria y el mapeo de direcciones; la experiencia demuestra que el concepto de número real refleja con precisión las propiedades generales de las magnitudes continuas.

Así, con el segundo aspecto se propone cuantificar el direccionamiento realizado por un procesador mediante la aritmética. Para iniciar se parte de un axioma que se considera el más evidente en el direccionamiento. Se observa que en el direccionamiento, el nombramiento de las direcciones absolutas de los procesado-

res y memorias actuales va de modo consecutivo en incrementos unitarios de una dirección a otra hacia direcciones más altas, o en decrementos unitarios hacia otras más bajas, y este patrón es constante. Además, el mapeo básicamente es un problema de correspondencias entre las direcciones absolutas y las direcciones mapeadas, siendo esta relación lineal. De acuerdo a esto, se conciben el primero y el segundo de los postulados que se presentarán más adelante.

En general, se utilizan números enteros positivos para nombrar a cada localidad direccionada; no se nombra con números negativos ni imagina-

rios. Según este axioma se puede concebir parcialmente el lugar geométrico para el mapeo que se describe con el tercer postulado.

**Postulado 1:** la relación entre las direcciones absolutas y direcciones mapeadas es lineal, y por lo tanto una línea recta es la definición de esa relación.

**Postulado 2:** la razón entre cualquier par de nombramientos unitarios de las direcciones absolutas y mapeadas se supone igual a 1.

**Postulado 3:** por lo tanto, el plano cartesiano en su primer cuadrante

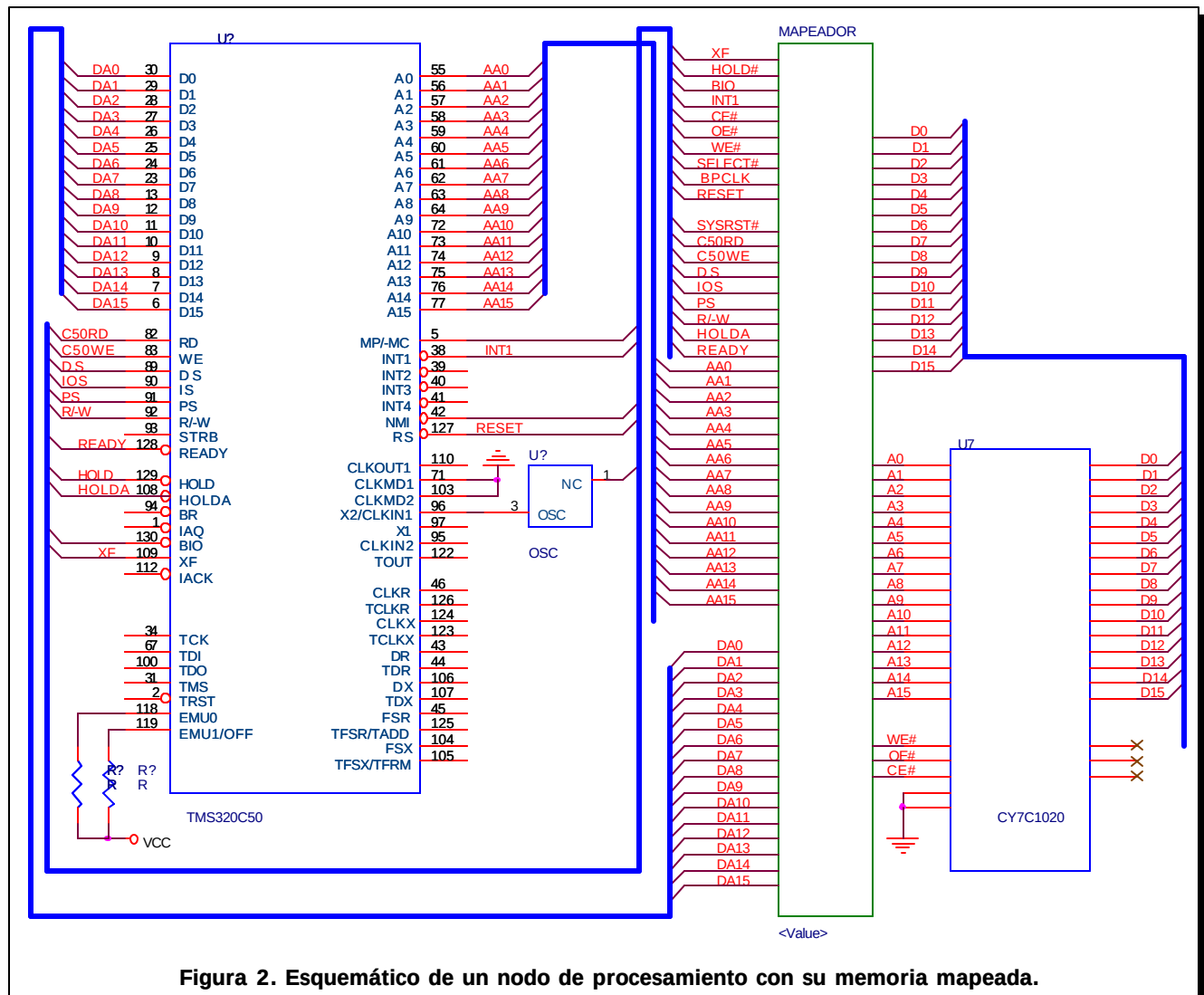


Figura 2. Esquemático de un nodo de procesamiento con su memoria mapeada.

se presenta como la mejor opción, en donde el eje vertical es la dirección mapeada, el eje horizontal es la dirección absoluta a mapear y la línea recta define al mapeo.

En el mapeo se permite que un procesador proyecte su dirección absoluta a una localidad de una memoria no inmediata a él, llamada memoria mapeada, cuyo nombramiento es de acuerdo al nombre de la dirección absoluta, lo cual genera el siguiente postulado:

**Postulado 4:** la dirección absoluta del procesador es la variable independiente  $x$  de la ecuación cartesiana y la dirección mapeada es la variable dependiente  $y$ .

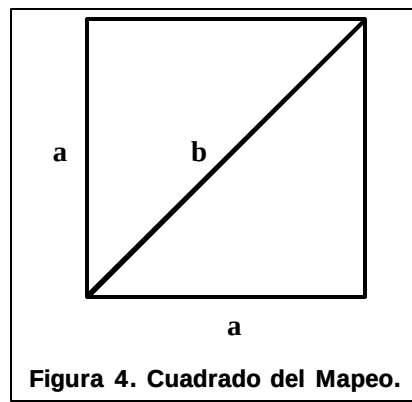
Aplicando una vez más el tercer aspecto, pero ahora bajo la perspectiva de los cuatro postulados anteriores, se puede abstraer el direccionamiento al primer cuadrante del plano cartesiano, en donde la función continua es una línea recta con una pendiente de 45 grados desde el origen del plano. La figura 3 muestra la abstracción del direccionamiento típico de un nodo de procesamiento, con la siguiente consideración: en el origen del plano, el cero, se encuentran las direcciones absolutas y mapeadas nombradas como las direcciones cero, ya que normalmente en los mapas de memoria se considera el

cero como el primer nombramiento[1] de las direcciones (es inusual ver mapas de memoria en donde la primera dirección sea nombrada con 1 u otro valor diferente a cero).

La figura 3 muestra el mapeo aparentemente imposible con el procesador cuyo mapa de direcciones va de 0 a 65535. A partir de aquí empezamos a considerar al direccionamiento como un ente más abstracto con el motivo de darle una demostración más rigurosa. Por tanto, se entrará en un nuevo nivel de abstracción al considerar un cuadrado en el plano cartesiano. Esta nueva etapa de razonamiento matemático busca demostrar que la comparación entre la variable independiente y el mapeo es irracional, y para ello se sigue la siguiente hipótesis[1]:

*"Se dice que el lado y la diagonal de un cuadrado son, de hecho, inconmensurables"*

A partir de este momento se entra en un razonamiento matemático que hace más tenue la conexión con el origen computacional del problema; por lo tanto, el nivel de abstracción es tal que el fenómeno real del mapeo ha dado paso a la comprobación de que la comparación entre el lado inferior horizontal del cuadrado y su diagonal es irracional. La figura 4 muestra el cuadrado obtenido a partir de la figura 3.



Se dice que no existe ninguna fracción tal que su cuadrado sea igual a 2. Bien, demostremos por contradicción, y sean  $p$  y  $q$  números enteros para los cuales

$$\frac{p}{q}^2 = 2$$

... donde se puede suponer que  $p$  y  $q$  no tienen factores comunes, porque en otro caso podríamos simplificar la fracción. Pero si  $(p/q)^2 = 2$ , entonces  $p^2 = 2q^2$ , y por tanto  $p^2$  es divisible por 2. En este caso  $p^2$  es también divisible por 4, puesto que es el cuadrado de un número par. Así  $p^2 = 4q_1^2$ ; esto es,  $2q^2 = 4q_1^2$ , y  $q^2 = 2q_1^2$ . De esto se sigue que  $q$  debe también ser divisible por 2. Pero esto contradice la suposición de que  $p$  y  $q$  no tienen factores comunes. Esta contradicción demuestra que el cociente  $b/a$  no puede ser expresado mediante un número racional, esto es: "La diagonal y el lado de un cuadrado son inconmensurables".

Esta demostración nos lleva a una etapa más de abstracción cuando se conecte con el Teorema de Cantor, pero antes cabe una conclusión parcial. Se concluye que la función del mapeo queda indeterminada por la exclusividad de las direcciones absolutas, ya que la ausencia de las direcciones mapeadas en los razonamientos matemáticos elimina la correspondencia biunívoca, y ésta en sí es la esencia del mapeo.

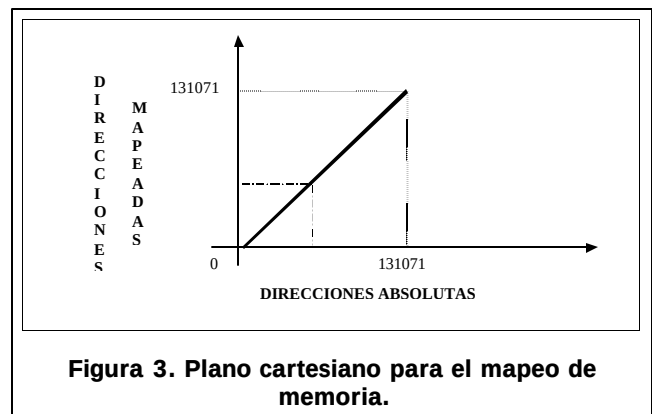


Figura 3. Plano cartesiano para el mapeo de memoria.

Si  $a$  es el lado y  $b$  la diagonal de un cuadrado, entonces según el teorema de Pitágoras:

$$b^2 = a^2 + a^2 = 2a^2,$$

y por tanto

$$\frac{b}{a}^2 = 2$$

Del cuadrado de la figura 4 de lado  $a$  y diagonal  $b$ , parece evidente, considerando toda su área, que hay muchos más puntos tomando en cuenta los lados verticales y el horizontal superior que en el lado horizontal inferior. O también se podría decir que, como la magnitud de la diagonal  $b$  es mayor que la magnitud del lado  $a$ , entonces hay más puntos en la diagonal que en cualquiera de los lados  $a$ . Lo anterior nos lleva a un axioma heurístico: "los puntos en la geometría euclídea son como las direcciones absolutas del direccionamiento de un procesador". Sin ellos los demás conceptos no tendrían existencia.

Para aclarar esto se va a demostrar que el lado inferior horizontal  $a$  tiene más puntos que la diagonal  $b$  del cuadrado. Básicamente se establece un procedimiento que haga corresponder un punto dado de la diagonal  $b$  con un punto del lado  $a$ . Siempre y cuando a cada punto de la diagonal  $b$  corresponda uno y nada más que uno del lado  $a$  y nunca uno del lado  $a$  corresponda a más de uno de la diagonal  $b$ . Es decir, se establece entre ambos conjuntos de puntos la correspondencia biunívoca.

Así, ante todo se necesita fijar una manera de localizar inequívocamente un punto cualquiera en el plano del cuadrado y el lado inferior  $a$ . Por tanto, por el método cartesiano un punto  $P$  de la diagonal (el plano) se distingue de todos los demás por sus distancias a los lados  $a$  (para el método cartesiano los lados que se consideran son el vertical izquierdo y el horizontal inferior) del cuadrado. El lado vertical  $a$  se superpone al eje positivo de las ordenadas y el lado horizontal  $a$  se superpone al eje de las abscisas. Para simplificar las cosas se considera que los lados del cuadrado tienen como medida la unidad. De esta forma, todas las distancias  $(x, y)$  medidas en el cuadrado serán inferio-

res o cuando más iguales a uno, y se expresarán como fracciones decimales.

A un punto del plano del cuadrado se le hace corresponder un punto del eje horizontal, el cual se obtiene mediante el cálculo del área del cuadrado delimitado por las coordenadas del punto del plano. Con este razonamiento se involucra a los lados que permiten la correspondencia biunívoca entre las ordenadas y las abscisas. De este modo, el punto se localizará mediante las siguientes expresiones:

$$y \neq 0.0$$

$$x \neq 0.a^2$$

Para establecer una correspondencia entre un punto del plano, como  $(0.3, 0.3)$ , con un punto en el eje de las abscisas se superpone el producto del área del cuadrado con el lado inferior horizontal  $a$ , de tal modo que el producto 0 del área coincida en el origen del plano cartesiano y su crecimiento siga por el eje de las abscisas. Así, el punto  $(0.3, 0.3)$  se hace corresponder con el punto donde  $y = 0.0$ , y  $x = 0.09$ .

La figura 5 muestra la correspondencia entre el punto  $(0.3, 0.3)$  del plano y el punto  $(0.09, 0.0)$ .

La razón de emplear el área del cuadrado es poder involucrar intuitivamente dos conceptos matemáticos, a saber: la *correspondencia biunívoca* entre los ejes del plano cartesiano, y la *potencia*. En el anterior procedimiento la potencia existe en el momento del cálculo de área del cuadrado.

En forma general se puede decir que, para todo punto  $P$  de coordenadas  $(0.a, 0.a)$ , existe un punto  $Q$  en el segmento de coordenadas  $[[0.a]^2, 0.0)$ . Este punto está en el segmento, puesto que su  $x$  es mayor que 0 y menor que 1. Este procedimiento se

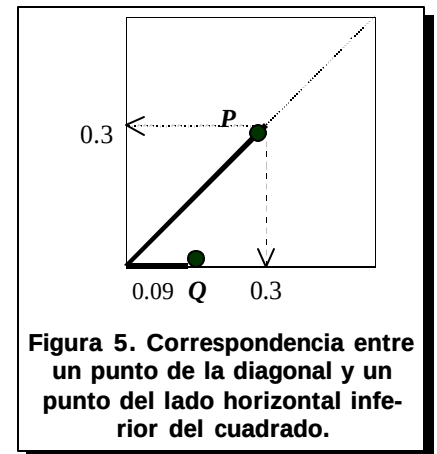


Figura 5. Correspondencia entre un punto de la diagonal y un punto del lado horizontal inferior del cuadrado.

conoce con el nombre de *Correspondencia de Cantor*, y el Teorema, que es también de Cantor, se enuncia como sigue:

*"Los puntos del plano tienen la potencia del continuo"*

Desde la perspectiva del direccionamiento de memoria lo anterior tiene el siguiente significado: de un modo u otro con las direcciones del mapa de memoria de un procesador se podría mapear tanta memoria como un factor conocido lo permitiese.

En el caso del diseño digital de un procesador el mapa de puertos es el factor multiplicativo más empleado. El método más usual es el usar comandos a través de un puerto o puertos. Haciendo una analogía con la Correspondencia de Cantor, se podría considerar que el comando del puerto tendría la misma función de las ordenadas: *compactar*. Con este método se compacta de un modo artificial el direccionamiento; así, un procesador con un mapa de 65536 podría llegar a mapear el orden de Gigas. Si el comando es de 16 bits de datos, entonces un puerto podría proponer  $2^{16}$  combinaciones posibles que permitirían establecer  $2^{16}$  mapas de memoria de 65536 direcciones, es decir, se podría tener un direccionamiento de hasta 4 Gigas.

La figura 6 muestra el diseño del nodo de procesamiento, en donde el procesador cuenta con un par de buses de 16 bits tanto para el área de datos y de direcciones; además tiene un puerto de 16 bits, y el banco de memoria lo constituyen 6 dispositivos SRAM de 64k x16. La entidad digital, por así decirlo, encargada de comprobar la Correspondencia de Cantor y de involucrar el factor para compactar el direccionamiento es un circuito programable: CPLD.

El listado 1 presenta el método codificado en VHDL que deberá implementarse para lograr la "compactación" del mapa de memoria original del procesador.

De este modo se puede decir que se ha comprobado un teorema computacional, el cual se enuncia diciendo:

*"el método de comandos por puertos es un procedimiento ade-*

*cuado de compactación del mapa de memoria de direcciones absolutas, y un eficiente mecanismo de expansión del mapa de memoria de direcciones mapeadas"*

Con este Teorema se explica la razón de la compactación del direccionamiento de un procesador al momento del mapeo de grandes cantidades de memoria mapeada. En la actualidad existen otros métodos de compactación más complejos, tal y como lo es el método de paginación[3] similar al usado por el kernel[3] de los sistemas operativos multiusuarios: UNIX, WINDOWS NT, etc.

## RESULTADOS

La investigación realizada en este trabajo proporciona un principio filosófico aplicado a partir de muchos, que permite entender la base con que se construyen los circuitos de hardware que proporcionan los diversos modos de direccionamiento en los bloques de memoria virtual[3] de los sistemas operativos.

## CONCLUSIONES

Cuando se localiza el elemento básico de un hecho real, como lo es una dirección de un direccionamiento, en esa medida se debería poder asociar ese hecho como uno de los significados del Teorema de Cantor. De hacerlo así se tendría

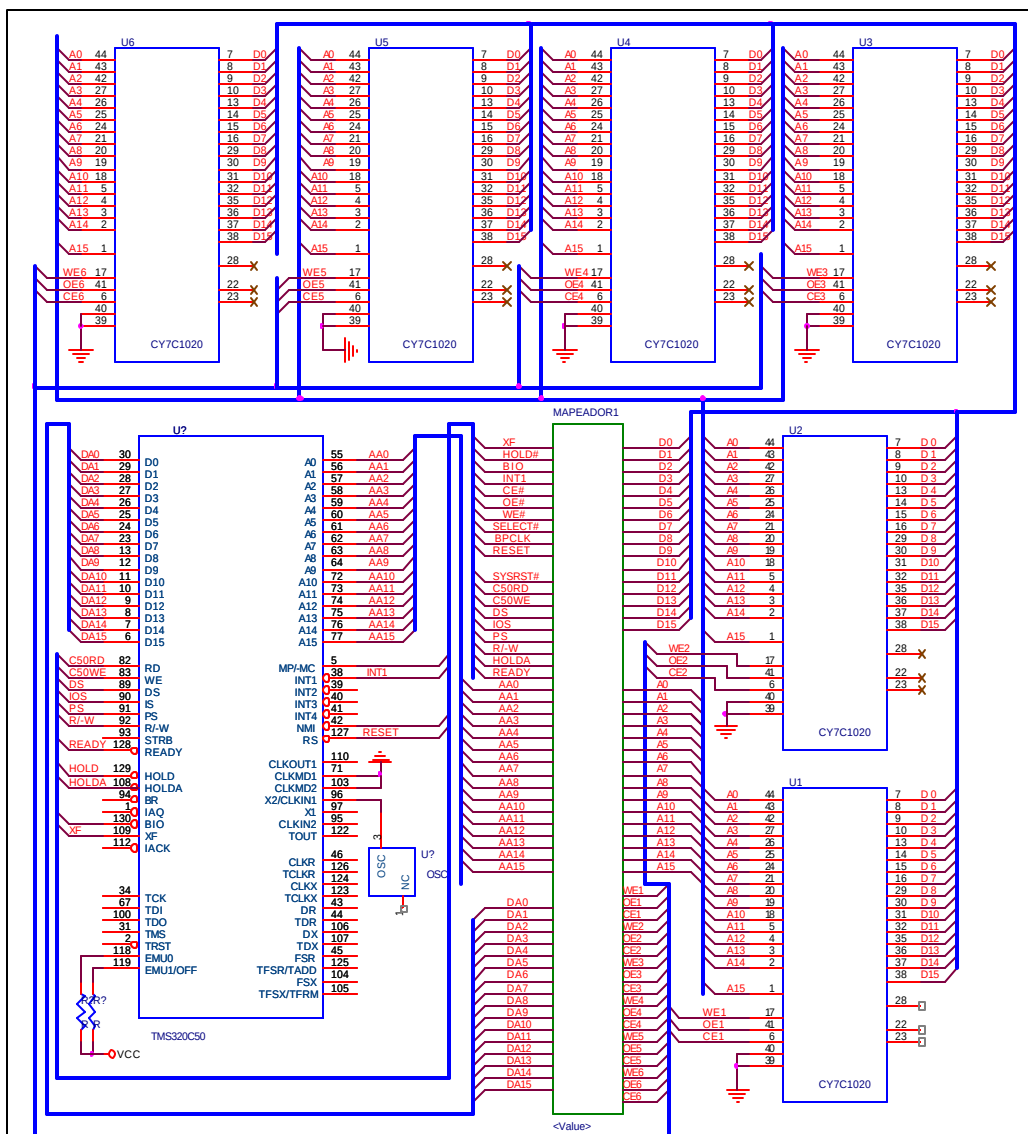


Figura 6. Esquemático del nodo de procesamiento con 6 dispositivos de memoria.

—Programa en VHDL para el plantear el comportamiento del Mapeador.

—En la figura 6, se muestra el esquemático para el direccionamiento

del Mapeo.

```
library ieee;
use ieee.std_logic_1164.all;
use work.numeric_std.all;
entity mapeador is
port(
  —buses del procesador.
  AA:in std_logic_vector(15 downto 0);
  DA:inout std_logic_vector(15 downto 0);
  —control del procesador.
  C50RD: in std_logic;
  C50WE: in std_logic;
  —selección de área de datos(ds)
  — o de puertos(is) del procesador.
  DS: in std_logic;
  IOS: in std_logic;
  —señales activas en bajo.
  —señales de control del mapeador.
  —Chip Select.
  CE: out std_logic_vector(5 downto 0);
  —Output Enable.
  OE: out std_logic_vector(5 downto 0);
  —Write Enable.
  WE: out std_logic_vector(5 downto 0);
  —buses del mapeador.
  A:out std_logic_vector(15 downto 0);
  D:inout std_logic_vector(15 downto 0));
end mapeador;
architecture comportamiento of mapeador is
signal pto: integer range 0 to 6;
begin
p1:process(IOS)
begin
  case DA(2 downto 0) is
    —comandos para identificar la activación
    —de cada una de las memorias mapeadas.
    when «000» => pto <= 0;
    when «001» => pto <= 1;
    when «010» => pto <= 2;
    when «011» => pto <= 3;
    when «100» => pto <= 4;
    when «101» => pto <= 5;
    when others => pto <= 6;
  end case;
end process p1;
p2:process(DS)
begin
case pto is
  when 0 => CE(0)<='0';
    OE(0)<='0';
    WE(0)<='0';
    A(15 downto 0)<=AA(15 downto 0);
    if (C50WE='0') then
      D(15 downto 0)<=DA(15 downto 0);
    else
      DA(15 downto 0)<=D(15 downto 0);
    end if;
```

```
  when 1 => CE(1)<='0';
    OE(1)<='0';
    WE(1)<='0';
    A(15 downto 0)<=AA(15 downto 0);

    if (C50WE='0') then
      D(15 downto 0)<=DA(15 downto 0);
    else
      DA(15 downto 0)<=D(15 downto 0);
    end if;
  when 2 => CE(2)<='0';
    OE(2)<='0';
    WE(2)<='0';
    A(15 downto 0)<=AA(15 downto 0);
    if (C50WE='0') then
      D(15 downto 0)<=DA(15 downto 0);
    else
      DA(15 downto 0)<=D(15 downto 0);
    end if;
  when 3 => CE(3)<='0';
    OE(3)<='0';
    WE(3)<='0';
    A(15 downto 0)<=AA(15 downto 0);
    if (C50WE='0') then
      D(15 downto 0)<=DA(15 downto 0);
    else
      DA(15 downto 0)<=D(15 downto 0);
    end if;
  when 4 => CE(4)<='0';
    OE(4)<='0';
    WE(4)<='0';
    A(15 downto 0)<=AA(15 downto 0);
    if (C50WE='0') then
      D(15 downto 0)<=DA(15 downto 0);
    else
      DA(15 downto 0)<=D(15 downto 0);
    end if;
  when 5 => CE(5)<='0';
    OE(5)<='0';
    WE(5)<='0';
    A(15 downto 0)<=AA(15 downto 0);
    if (C50WE='0') then
      D(15 downto 0)<=DA(15 downto 0);
    else
      DA(15 downto 0)<=D(15 downto 0);
    end if;
  when others =>
    A(15 downto 0)<=>ZZZZZZZZZZZZZZZZZZZZ;
    D(15 downto 0)<=>ZZZZZZZZZZZZZZZZZZZZ;
    CE(5 downto 0)<=>111111;
    OE(5 downto 0)<=>111111;
    WE(5 downto 0)<=>111111;
  end case;
end process p2;
end comportamiento;
```

**Listado 1. Programa en VHDL para el CPLD, que compacta el direccionamiento del procesador.**

un pensamiento en términos de conceptos simbolizados, un pensamiento altamente especializado y funcional, tal y como se presentó en este trabajo. El único punto débil, si lo hay, es que esto se construye con la intuición y no basándose en razonamientos lógicos. Aquí los razonamientos lógicos se emplearon en una fase posterior, para intentar un cierto grado de acuerdo con el lector.

Además, parece que la *intuición* es el mecanismo de captación de un concepto, que junto con la familiarización y el *know how* de un campo permiten conocer ciertos significados especializados (los conceptos simbolizados), y al alinearlos a los problemas que se intentan solucionar, los solucionan, o por lo menos permiten obtener descripciones que funcionalmente aclaran el problema.

Ahora, de acuerdo con la investigación de este trabajo se podría concebir que el analfabetismo funcional se podría extrapolar patológicamente en las actividades teóricas de un pragmático, y así uno podría comprender el porqué ciertas personas aunque son diestros en manipular el álgebra, el cálculo y la geometría analítica sean incapaces de dar soluciones matemáticas a ciertos problemas de la ingeniería y de las ciencias, y en esa medida presentarían la inhabilidad para generar trabajos originales. Otro aspecto patente es que, ante la carencia de un significado especializado, se entra en una situación de inacción, y sólo se accesa a un gran potencial de acción cuando se logra dicho significado. Quizás este párrafo presente una de las razones de la productividad inconstante al realizar esta investigación.

---

#### BIBLIOGRAFÍA

---

- [1] Aleksandrov A. D., Kolmogorov A. N., y Laurentiev, M. A., "*La matemática: su contenido, métodos y significado*", Alianza Universidad, Alianza Editorial, Vol. 1, México, 1994, Cap. 1.
- [2] Erro Luis Enrique, "*El Pensamiento Matemático Contemporáneo*", Instituto Politécnico Nacional, México, 1986, Cap. 1.
- [3] Freedman Alan, "*Diccionario de Computación Bilingüe*", McGraw-Hill Interamericana, S.A., Colombia, 1996.

# Diseño de Sistemas Digitales Utilizando FPGA

M. en C. Amadeo José Argüelles Cruz  
Profesor del CIC-IPN

Ing. José Angel Ascencio Roman,  
Ing. José Felipe Villalobos Baigorria  
CIC-IPN

**E**n la década de los ochenta se introdujo en el mercado una nueva tecnología para la implantación de sistemas digitales: los arreglos de compuertas programables en campo (FPGA, "Field Programmable Gate Array"). Estos dispositivos ofrecen varias ventajas con respecto a los componentes discretos de mediana y alta escala de integración (MSI/LSI) y de aplicación específica. Los FPGA son capaces de operar con lógica multinivel, mientras que dispositivos tales como los PLD se construyen específicamente para trabajar con lógica de dos niveles. Además, no precisan de fabricarse "a medida", lo que da como resultado una reducción tanto en los costos, cuando se genera un bajo volumen de partes, como en el tiempo de fabricación.

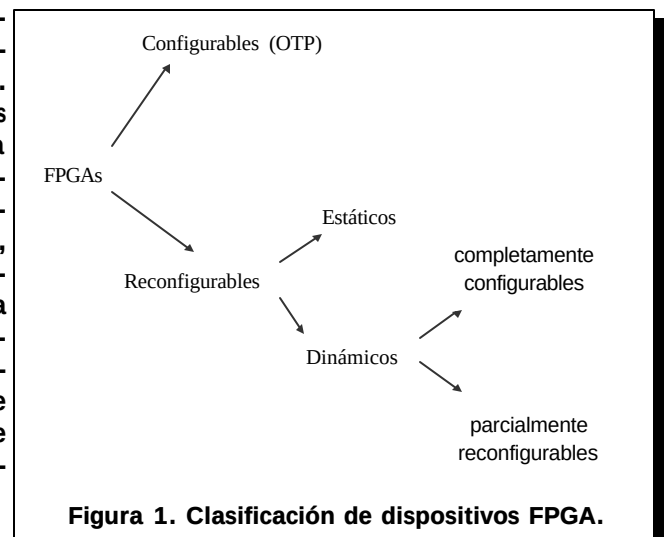
En este documento se exponen algunas de las características contenidas en la tecnología de esta clase de dispositivos y el proceso de diseño utilizado, así como algunas de las aplicaciones en las que actualmente se encuentran presentes.

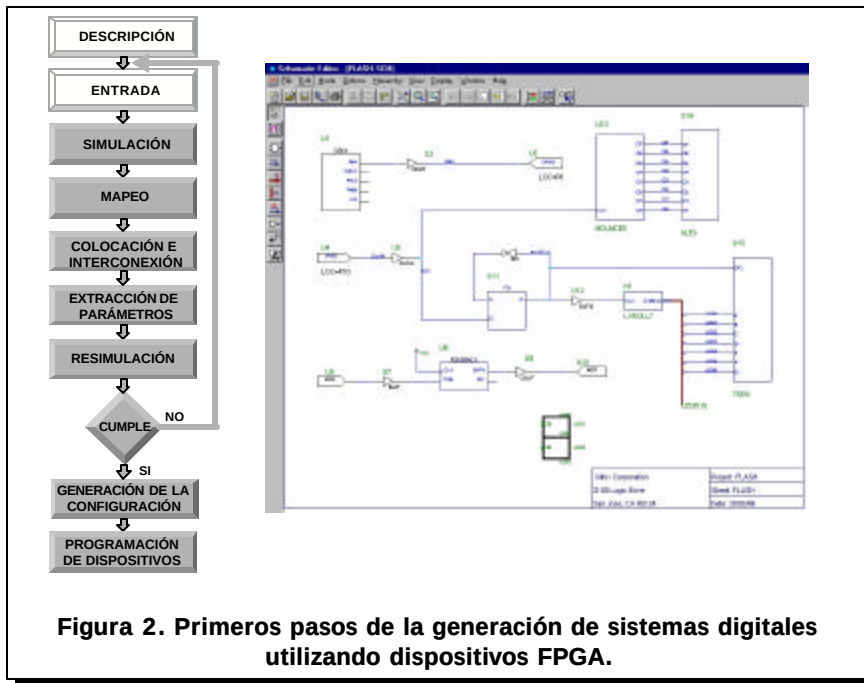
## TECNOLOGÍA DE FPGA

Esta es una clase de dispositivos prefabricados pero a diferencia de los PLD, que contienen lógica de dos niveles, presenta un diseño optimizado que permite el manejo de lógica multinivel. Esta característica permite diseñar circuitos más complejos que pueden contenerse en un solo dispositivo, aunque se tiene la desventaja de los retardos involucrados en el diseño, lo cual no se presenta en los PLD. Los FPGA proporcionan especificaciones especiales para su ajuste a determinado tipo de necesidades; tomando como base estas especificaciones y la manera en como pueden programarse, la figura 1 muestra una clasificación de estos dispositivos.

Con base en su forma de configuración, existen cuatro tipos de FPGA. En los dispositivos configurables, una vez que se programa el sistema digital que contendrán, no pueden realizarse cambios en la manera en que fueron dispuestas tanto la distribución de la lógica como de las conexiones internas.

Esta es la razón por la que a este tipo de FPGA se le da el nombre de programables una sola vez (OTP: "One Time Programming"). Los dispositivos reconfigurables pueden programarse varias veces, y se dividen en estáticos y dinámicos; en los primeros, una vez que se carga la configuración en uno o varios de los dispositivos que conforman al sistema, se ejecuta la tarea en cuestión, sin realizarse mas cambios en el funcionamiento. Con los circuitos dinámicos el hardware subyacente puede cambiar en cualquier momento durante la ejecución de la operación u operaciones que se llevan a cabo. A su vez, los circuitos dinámicos se dividen en dispositivos completamente configurables, en los cuales el dispositivo se reconfigura completamente; y los circuitos parcialmente reconfigurables, en los que solo se cambia una parte del circuito.





lidad, permitiendo al ingeniero explorar a fondo el diseño, con el objeto de incorporar cambios de ultimo minuto y sin incurrir en costos adicionales.

- Alta integración (alrededor de 2 millones de compuertas), lo cual reduce drásticamente el número de componentes en un diseño en comparación a su implantación con lógica MSI/LSI. Además, disminuye el consumo de energía, se obtienen costos de manufactura bajos, inventarios pequeños y un número reducido de defectos de manufactura y errores de diseño.

## CICLO DE DISEÑO

El ciclo de diseño (ver figura 2) para el desarrollo de sistemas digitales con FPGA requiere de los siguientes pasos:

1. La introducción o descripción del diseño.
2. La simulación del diseño para verificar su funcionamiento.

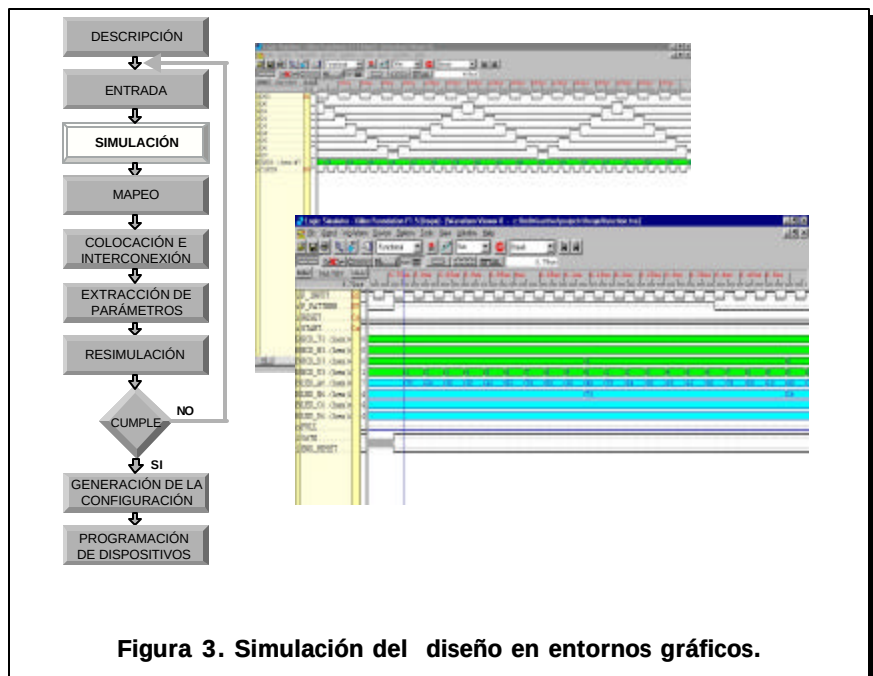
Otra manera de identificar a un FPGA es con base en la complejidad de sus celdas elementales. Los circuitos con “granularidad fina” presentan celdas básicas cuyo contenido son las compuertas lógicas AND, OR y NOT. Otro tipo de circuitos presentan “granularidad gruesa”, es decir, sus celdas básicas son más complejas, ya que pueden contener tablas de verdad, multiplexores y/o unidades aritmético-lógicas.

- Reducción en los tiempos de producción. Como ejemplo, el tiempo de manufactura aproximado de un dispositivo MPGA es del orden de 6 meses; con los FPGA, las partes pueden producirse después de solo unas cuantas horas de diseño. Esto proporciona un amplio grado de flexibi-

## VENTAJAS

Algunas de las ventajas de trabajar con estos dispositivos se mencionan a continuación:

- Bajo costo, ya que la configuración y programación del dispositivo se soporta por el software de diseño. Como resultado, los cambios en el diseño y mejoras en los productos obtenidos pueden realizarse sin incurrir en gastos adicionales.



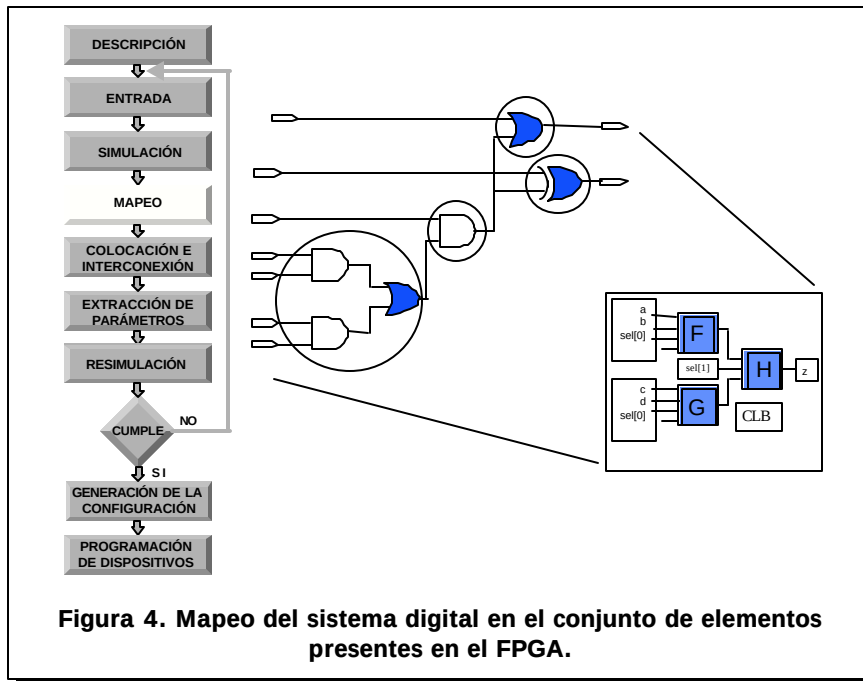


Figura 4. Mapeo del sistema digital en el conjunto de elementos presentes en el FPGA.

guajes de descripción de hardware, etc.

Cuando el diseño ha sido insertado en la herramienta de automatización, el siguiente paso consiste en la simulación del mismo (figura 3). Se puede observar si el diseño cumple con la funcionalidad deseada cuando se alimentan los vectores de prueba, mismos que servirán para modelar el comportamiento del sistema. Este paso puede omitirse para acelerar el proceso de obtención del producto final, pero esto puede ocasionar que el mismo no presente la calidad que se requiere, lo que redundará en un mayor costo si se involucra al diseño con dispositivos que se programan solo una vez, OTP.

En el paso 3, se realiza la reducción lógica y el mapeo del diseño con respecto a una tecnología determinada (figura 4). Esto implica que la descripción del diseño, que antes de esta etapa todavía se presenta de manera independiente a la tecnología, sea transferida a las entidades (celdas) básicas de la tecnología que contendrá al diseño.

3. El mapeo del diseño en la arquitectura del FPGA a utilizar.
4. La colocación e interconexión del diseño en el FPGA.
5. La extracción de parámetros, una vez que el diseño ha sido conectado.
6. Una nueva simulación para verificar los tiempos de propagación de las señales involucradas en el diseño.
7. La generación del formato de configuración del dispositivo FPGA.
8. La configuración o programación del dispositivo.
9. La prueba del producto para observar la existencia de algún funcionamiento no deseado.

En el paso número 1 se contempla la descripción e inserción del diseño con alguna de las herramientas de implantación del diseño (programas de software llamados EDA: "Electronic Design Automation", Automati-

zación del Diseño Electrónico) que los fabricantes de FPGA distribuyen junto con sus productos (figura 2). Con la utilización de estos programas, el diseñador puede describir de diferentes formas a los circuitos que intervienen en el sistema digital: utilizando esquemas, expresiones booleanas o lógicas, lista de conexiones y/o len-

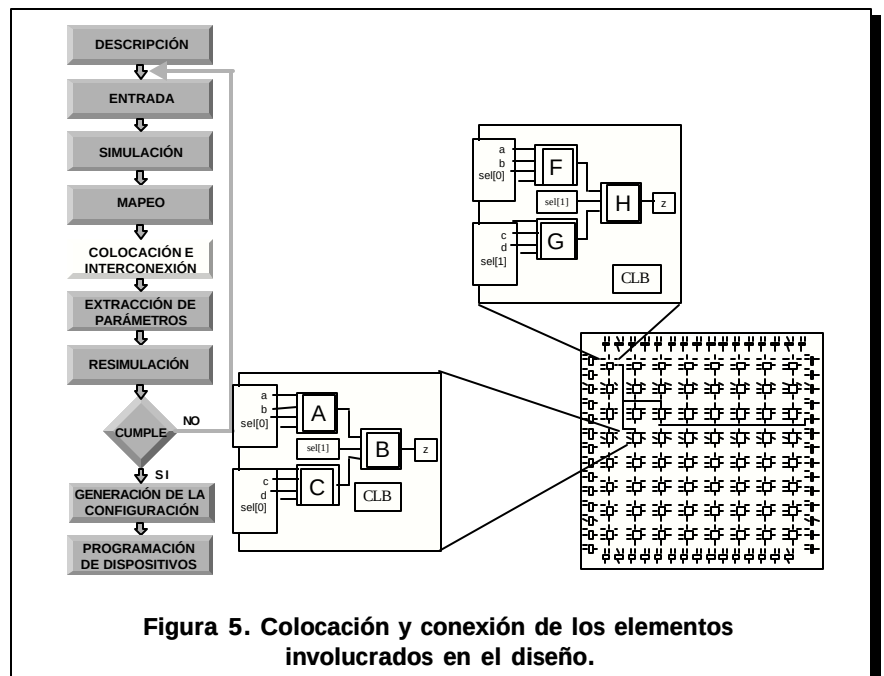


Figura 5. Colocación y conexión de los elementos involucrados en el diseño.

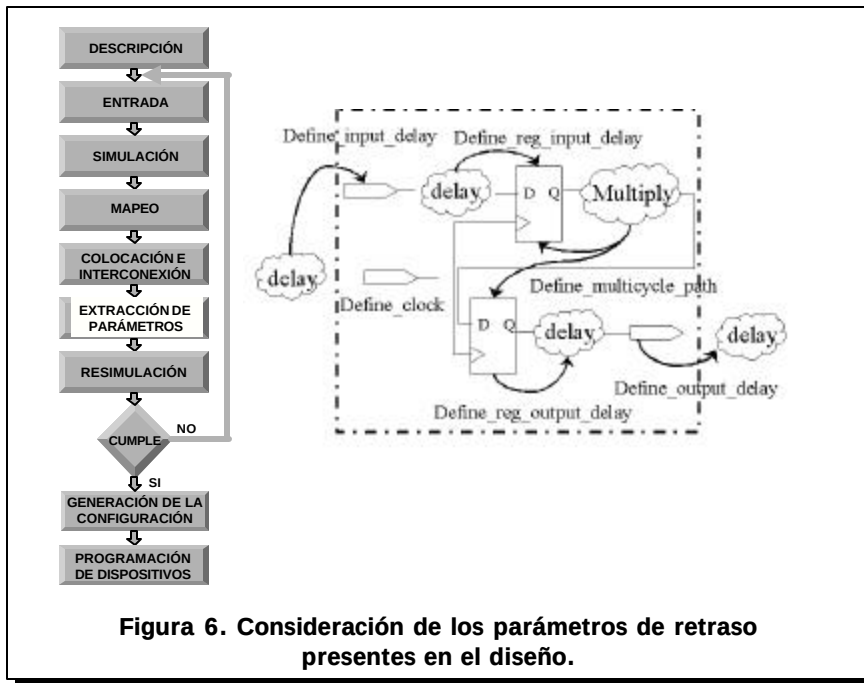


Figura 6. Consideración de los parámetros de retraso presentes en el diseño.

En el paso 4 se posicionan las entidades, elementos o celdas, utilizadas en el mapeo, sobre un bloque físico específico presente en el dispositivo (figura 5). A continuación se realiza la conexión de los diferentes bloques, la cual se lleva a cabo en dos etapas: global y detallada.

A continuación se lleva a cabo la adquisición de los parámetros de retraso de las señales involucradas en el diseño (figura 6). Esto es, ya que se han colocado y conectado todos los elementos que forman parte del diseño, la herramienta para el mismo toma de la base de datos correspondiente al dispositivo utilizado, las características que el fabricante declara en sus hojas de especificaciones.

En la etapa 6 se procede a una nueva simulación del diseño (figura 7), utilizando los mismos vectores de prueba que sirvieron para llevar a cabo el paso 2. Esto permite la verificación de los tiempos de propagación de las señales involucradas en el diseño, pero en esta ocasión, a diferencia de la simulación realizada en el paso

2, se toman en consideración los retardos producidos por las rutas de conexión existentes.

Si el resultado de la etapa anterior cumple con las especificaciones de diseño, entonces se procede a ejecutar el paso 7 (figura 8), que es la

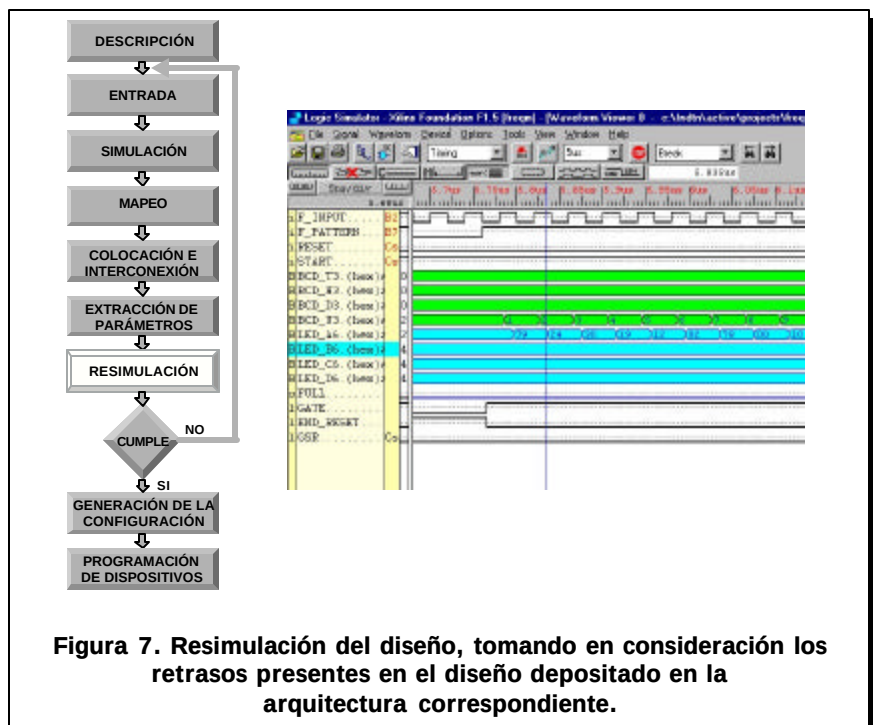
generación de los formatos de configuración del dispositivo o dispositivos indicados.

Concluido lo anterior, se procede a programar al dispositivo (figura 9), utilizando los archivos de configuración correspondientes.

Por último, se prueba el diseño para verificar su funcionamiento correcto.

## APLICACIONES

El campo de aplicación de los dispositivos FPGA es muy amplio. Inicialmente se utilizaban en los sistemas digitales como lógica adicional (*glue logic*), pero a medida que los fabricantes fueron desarrollando nuevas características (mayor integración, menor costo, menor consumo de energía, etc.), su utilización fue cambiando a tal grado que en la actualidad han cobrado gran importancia en sectores tales como computación, periféricos, telecomunicaciones, redes,



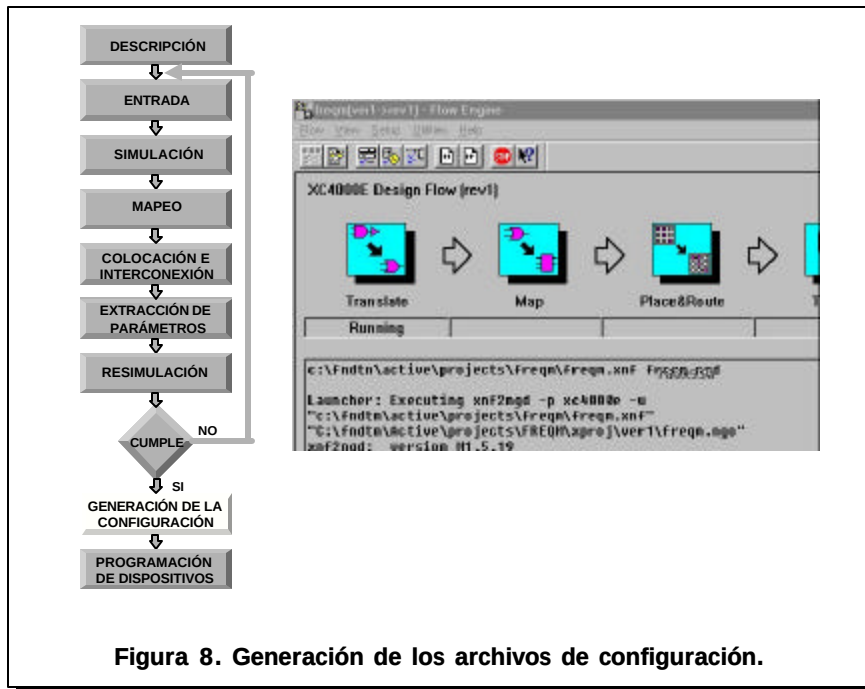


Figura 8. Generación de los archivos de configuración.

## RECONOCIMIENTOS

El autor agradece el apoyo de Anna Acevedo (Xilinx Corp.) y de Actel, por proporcionar información referente a la línea de dispositivos FPGA que fabrican; al Centro de Investigación en Computación (CIC), por el respaldo en las investigaciones realizadas en referencia a este tema y al Centro de Innovación y Desarrollo Tecnológico (CIDEDEC) por continuar con su esfuerzo por diseminar los conocimientos científicos y técnicos a través de la revista Polibits. Estos dos últimos Centros forman parte del Instituto Politécnico Nacional.

control industrial, instrumentación, militar y electrónica de consumo, utilizando tan solo un dispositivo que contenga a todo el sistema digital (SOC: "System On a Chip").

En nuestro país, actualmente se desarrollan prototipos encaminados a la generación no solo de elementos de apoyo académico; en el Centro de Investigación en Computación (CIC) se generan proyectos de investigación que tienen que ver con el desarrollo de sistemas digitales, utilizando lógica basada en FPGA reconfigurables estáticos (figura 10).

## CONCLUSIONES

Como se puede observar en el material presentado, los FPGA se utilizan en una amplia variedad de aplicaciones, involucrando todo tipo de sistemas de cómputo. La forma en como se desarrolla el diseño es bastante flexible y no se requiere de una inversión cuantiosa para poder realizar proyectos de baja y mediana esca-

la. Gracias al incremento en el número de compuertas y a la mejora en el rendimiento en los FPGA, existe un interés creciente en dirigir el diseño de sistemas digitales hacia una integración total.

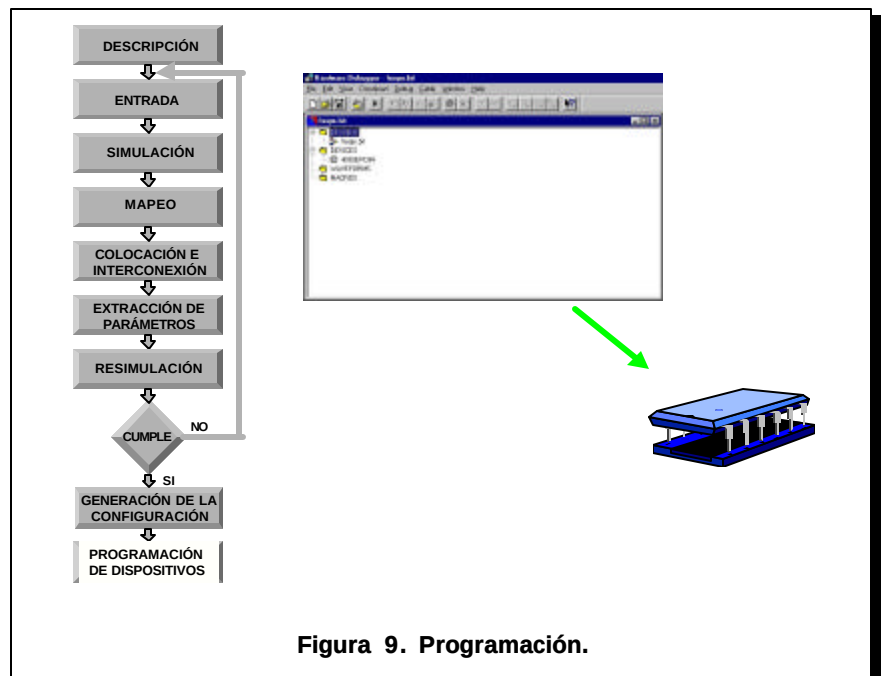


Figura 9. Programación.

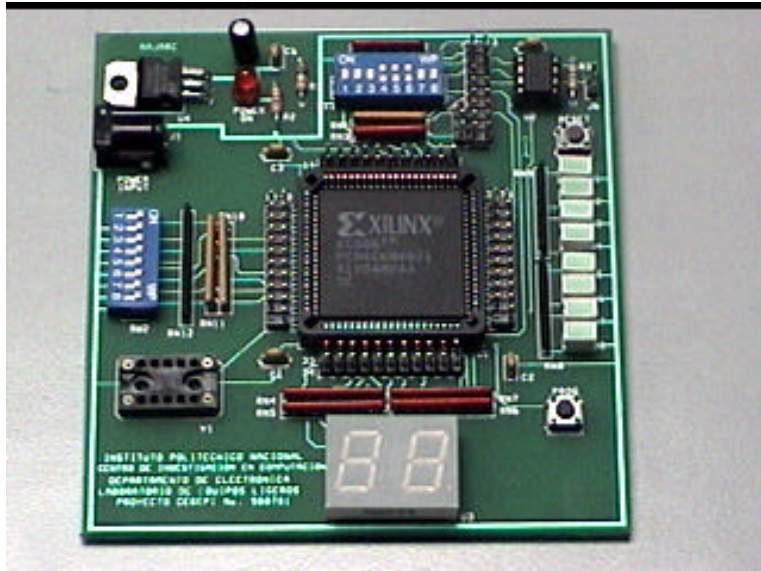


Figura 10. Prototipo para la generación de sistemas digitales empleando dispositivos FPGA XC4000 y Spartan del fabricante Xilinx.

#### BIBLIOGRAFÍA

- [1] Smith, M. J. S., *Application-Specific Integrated Circuits*. Addison-Wesley Longman, 1998.
- [2] Pierre Marchal, *Field-Programmable Gate Arrays*. *Communications of the ACM* Vol. 42, No. 4 April, 1999.
- [3] Xilinx, *The programmable Logic Data Book*, Xilinx, Inc. 1999.
- [4] Actel, *FPGA Databook and Design Guide*, Actel Corporation 1996.
- [5] Xilinx, *Xilinx University Program Workshop Labs*, Xilinx, Inc. 1998.

# Uso de Multimedia en la Educación Básica de Ingeniería por Internet

Rodolfo Romero Herrera  
ESCOM-IPN  
E-mail: rodolfo\_rh@hotmail.com

**E**ste documento presenta el estudio de diferentes sistemas de enseñanza basados en computadora, y los pasos a seguir para establecer un sistema de enseñanza semipresencial, soportado en metodologías y tecnologías diversas que emplean multimedia y, de manera específica, a través de Internet. Sobre esta base se cuestionó cuál sería el método más adecuado y con mejores resultados para la enseñanza de la ingeniería, usando esquemas en línea apoyados en computadora; así como las estrategias didácticas más apropiadas. Se da a conocer el sistema hipermmedia empleado, el cual está basado en el mindware, hardware y software disponible en la escuela.

---

## INTRODUCCIÓN

---

Cuando apareció el papel como soporte de la información, cambio entonces una serie de patrones culturales que afectaron la forma de vivir. De la misma forma, continuamente aparecen nuevas tecnologías que cambian los soportes de la información, pasando desde la pluma y el papel hasta los sistemas multimedia.

Hoy, la computadora deja de ser una sofisticada máquina de calcular, y se convierte en una máquina para

comunicarse, encontrándose en el ámbito multimedia, donde el sonido, la voz, el texto y la capacidad de trabajar conjuntamente a distancia son una realidad, tanto para la educación como para el desarrollo de los pueblos.

La escuela tradicional cumplía perfectamente con la función de distribución de la información necesaria en tiempo y forma, para que la gente que se formaba pudiera participar en la sociedad. En la actualidad, sin embargo, comparte esta función con muchos medios.

El conductismo ha resultado ser una de las corrientes que más han influido en la psicología contemporánea; surge como una corriente que vino a revolucionar la visión mentalista que imperaba en la época de su aparición. Sin embargo, actualmente el constructivismo goza de mayor popularidad en los sistemas educativos.

La necesidad de emplear los multimedia en la educación radica en el hecho de que no todos los individuos aprenden de la misma forma; el aprendizaje no siempre se da por el mismo canal de percepción, y las personas normalmente no tienen los mismos intereses o antecedentes. Es por esto que en el presente trabajo se justifica el uso del constructivismo combinado con el cómputo educativo, donde su principal aplicación será en la educación semipresencial.

---

## SISTEMAS DE ENSEÑANZA BASADOS EN COMPUTADORA

---

Hay muchos sistemas de enseñanza basados en computadora; sin embargo, cada uno de ellos termina ya sea en el conductismo o en el constructivismo. Es por eso que se realiza un análisis como primer paso para construir un sistema educativo basado en multimedia.

### CONDUCTISMO

Conceptos tales como motivación e impulso, así como sensación y percepción, entre otros, comenzaron a utilizarse para hacer referencia a las propiedades dinámicas e intrínsecas de la conducta humana. Sin embargo, el replanteamiento de sus teorías motivó que el conductismo presentara divergencias.

La relación estímulo-respuesta es el fiel reflejo de una concepción teórica que otorga primacía al objeto sobre el sujeto, en donde el estímulo provoca, activa y hace reaccionar al sujeto, pero no considera que el hombre interactúa con el medio social y biológico que lo rodea; debido a este proceso dinámico, el hombre tiene la posibilidad de desarrollarse y evolucionar.

El conductismo se concreta a concebir al hombre como un organismo dependiente de los cambios ambien-

tales, y que en forma automática responde a ellos. En el caso de los multimedia es prioritario que la computadora no sea la que tome el mando de las acciones; sin embargo, algunos conceptos del conductismo pueden ser de utilidad para ciertas actividades educativas.

## CONSTRUCTIVISMO

La base en este enfoque es el individuo, en el que una construcción propia se va produciendo como resultado de la interacción de su disposición interna y su interacción con el medio ambiente; así, el conocimiento es una construcción de la persona misma. Esta construcción resulta de la representación inicial de la información y de la actividad externa o interna al individuo; por esto resulta importante una concordancia alta entre la realidad y la construcción. De esta forma, es posible sugerir que en el constructivismo lo fundamental reside en el proceso de adquisición.

A través de los procesos de aprendizaje, el alumno construye estructuras, es decir, maneras de organizar la información, las cuales facilitarán mucho las adquisiciones futuras. Esto provoca la interactividad si se implementa en un sistema de cómputo educativo.

Las estructuras de conocimiento son las representaciones organizadas de experiencias previas; son relativamente permanentes y sirven como esquemas que funcionan activamente para filtrar, codificar, categorizar y evaluar la información que uno recibe con alguna experiencia relevante. Todo ello permite establecer un reto al estudiante, muy similar al que existe en los juegos por computadora.

La idea principal es que mientras aprendemos información estamos constantemente organizándola en

unidades. Tales unidades tienen algún tipo de organización, que llamamos estructura. La nueva información generalmente se asocia con la existente en estructuras anteriores, a la vez que es posible reorganizar o reestructurar la información original.

Otro punto que se enfatiza en el constructivismo es que el conocimiento es un producto de la interacción social y de la cultura. Por tal motivo, no se debe permitir que el software aisle al individuo del mundo que le rodea, sino mas bien que sea una herramienta que le permita desenvolverse mas fácilmente dentro de él.

Piaget planteó que, para aprender, el alumno requiere de un estado de desequilibrio, una especie de ansiedad que sirve de motivación para aprender. Si nosotros provocamos ese deseo por aprender a través del empleo de multimedia, lograremos mejores resultados, ya que el estudiante siempre tendrá la necesidad de conocer más.

A pesar de la certidumbre de los conceptos en el constructivismo, recordemos que todos somos diferentes; por tal motivo es importante establecer una diferencia entre lo que el alumno es capaz de aprender solo y lo que puede aprender con el concurso de otras personas. Por ello, la educación escolar debe partir del nivel de desarrollo efectivo del alumno, pero no para acomodarse a él, sino para hacerlo progresar a través de la zona de desarrollo próximo, para ampliarla y generar eventualmente nuevas zonas de desarrollo.

Así, el punto más importante en el cómputo educativo es asegurar la realización de aprendizajes significativos, a través de los cuales el alumno construye la realidad atribuyéndole significados. Para tales fines, el contenido debe ser potencialmente importante y el alumno debe tener una

actitud favorable para aprender significativamente; cuanto mayor sea el grado de significatividad del aprendizaje realizado, mayor será también su funcionalidad.

En consecuencia, el aprendizaje (según el conductismo) es concebido como un cambio en la conducta por medio del condicionamiento, mientras que para el constructivismo el aprendizaje es una construcción propia que se va produciendo mediante la interacción de las estructuras internas con el medio ambiente. Entonces, si consideramos los puntos mas adaptables a los multimedia obtendremos un sistema educativo de enseñanza que considere tanto aquellos alumnos cuyo condicionamiento es mas adaptable al conductivismo como los alumnos que lo son más hacia el constructivismo, teniendo entonces un híbrido de diferentes estrategias de enseñanza.

---

## ESTRATEGIAS DE APRENDIZAJE

---

### \* ESTRATEGIAS DE PROCESAMIENTO

Las estrategias de procesamiento son aquellas que utiliza el aprendiz para asegurar un proceso de captación e ingreso de nueva información, destrezas, etcétera. Estas estrategias incluyen :

1. Estrategias de atención.
2. Estrategias físicas.
3. Estrategias de elaboración verbal.
4. Elaboración por vía de imágenes.
5. La comparación.
6. La inferencia y
7. La aplicación o ensayos futuros.

### \* ESTRATEGIAS DE PRODUCCIÓN

Las estrategias de producción recuperan información almacenada para hacer uso de ella y para que sea

capaz de transferir o generalizar lo que ha aprendido a problemas que encontrará en su vida fuera de la escuela. Estas estrategias pueden ser desarrolladas y aplicadas a cuatro formas principales de producción:

- Recuperación y uso de información específica.
- Generalización o transferencia de información, o aplicación de habilidades a nuevas situaciones.
- Identificación, representación y resolución de problemas (aplicando lo aprendido).
- Desarrollo y aplicación de la creatividad en la generación de respuestas.

#### \* ESTRATEGIAS AFECTIVAS

El alumno debe poseer el estado afectivo positivo hacia las estrategias para usarlas, por ejemplo: Interés, motivación extrínseca, imagen de sí mismo, atribución de control, ansiedad, conciencia afectiva, etc.

#### \* METACOGNICIÓN

El alumno debe tener conciencia de lo que uno está haciendo; esto incluye:

- La planificación.
- La supervigilancia del proceso (monitoreo).
- La evaluación y constatación de resultados.

---

#### MODELO DE ENSEÑANZA ESTRATÉGICA

---

Una concepción de inteligencia «desarrollable» como la antes planteada conduce, obligadamente, a un mayor compromiso por parte de los profesores para el diseño de las condiciones del aprendizaje.

En esta nueva concepción, la porción fluida de la inteligencia, la relacionada con las estrategias y los estilos de aprendizaje, es susceptible de ser moldeada.

El estudiante necesita reconocerse a sí mismo como estudiante; saber cuál es su estilo de comprensión y aprendizaje, qué tareas, materiales y materias le son difíciles, etc. En este modelo se requiere un entrenamiento para que el estudiante esté consciente del tipo de actividades que se realizarán. Para ello se debe planear una serie de actividades que le ayuden a mejorar en su aprendizaje, tales como estrategias para la organización y estructuración, etc.

---

#### COMPONENTES DE UN SISTEMA EDUCATIVO MULTIMEDIA

---

En un sistema educativo con multimedia se deben contar con tres partes básicas: Hardware. Mindware, y Software.

#### HARDWARE

Una parte fundamental para el sistema es la computadora, la cual debe estar equipada con unidad de CD-ROM, suficiente memoria principal, disco duro con espacio de almacenamiento adecuado, micrófono, tarjeta de sonido y acceso a internet. Si se trata de educación a distancia, se necesitara también un servidor, video cámara, etc.

Existen otros componentes que forman parte del hardware necesario para crear multimedia, como son: Digitalizador, cámaras fotográficas digitales, cámaras de video, tabla digitalizadora, videogradora, etc.

#### MINDWARE

Mindware es el componente humano y educativo, que ayuda a organizar la parte técnica (hardware) y la

paquetería (software), para conformar un sistema multimedia. Si este factor no esta bien capacitado, de nada servirá que se tenga el equipo más sofisticado.

---

#### ALCANCES DEL CÓMPUTO EDUCATIVO

---

Las perspectivas del uso de multimedia en la educación son cada día más prometedoras, además de que el costo de estos sistemas tiende a disminuir en forma dramática; esto pone a disposición de los educadores medios que en otros tiempos eran inalcanzables. Todo ello, junto con el internet, nos permite una educación a distancia o semipresencial más efectiva.

El uso adecuado de múltiples medios asociados para la instrucción es un factor complementario y en algunas ocasiones necesario para lograr un aprendizaje significativo. Las posibles aplicaciones de los multimedia ofrecen la opción de tener una forma de acceso a información diferente con representaciones y situaciones más reales, lo que a su vez nos permite explorar en la práctica diversas estrategias de enseñanza.

---

#### SISTEMAS BASADOS EN MULTIMEDIA

---

En este caso la computadora desarrolla una función eminentemente educativa, sobre todo si consideramos que los espacios requeridos para la educación presencial son cada día mas escasos y congestionados, y que la enseñanza de la educación básica de ingeniería juega un papel muy importante en el avance de la tecnología. Así, aunque el medio es diferente los contenidos serán los mismos, pero con una mayor interactividad; es decir, que no se pretende únicamente

emplear la computadora como medio de enseñanza de un lenguaje de programación o algún paquete en especial, sino para cualquier contenido a transmitir, en particular las materias básicas en la tecnología.

El papel de la computadora como instrumento de ayuda para la adquisición de determinados conocimientos implica la utilización de un software previamente elaborado y que se ofrece al alumno para alcanzar un objetivo determinado. El éxito dependerá, en gran medida, de la calidad del software.

Para lograr esto se requiere que se formen grupos con especialidades diferentes. Por ejemplo, un grupo pedagógico, un grupo de diseño gráfico, un grupo para desarrollo de contenidos, y un grupo de programadores. Es evidente entonces que el equipo debe ser multidisciplinario. Considerando lo anterior, es factible realizar un sistema de este tipo en un lapso de 6 meses, si se dispone de los medios suficientes.

Como resultado de la aplicación de un sistema de este tipo a un grupo de 15 alumnos, se obtuvo un 100% de resultados efectivos. Para conseguir estos resultados es indispensable contar previamente con los polilibros; un polilibro no es un pasa página, ni tampoco un libro de texto por internet. Es una clase planeada, considerando que el receptor debe interactuar con la computadora y que esta debe captar la atención del estudiante, sin llegar a tener distractores al concepto que se desea aprender.

A pesar de lo atractivo de la idea, un polilibro considera en un principio la corriente conductista, pues la mayoría de nuestros alumnos han trabajado de esta manera, lo que les dificulta entender que ellos son los que deben construir el conocimiento; sin embargo, al final de un polilibro debe-

mos tener un material totalmente constructivista.

En un polilibro se requiere utilizar los multimedia adecuadamente, esto es, sin llegar al abuso. Por ejemplo, si queremos que el alumno escuche una melodía para que recuerde al autor de la misma, no es necesario entonces un micrófono, mas bien necesitaremos una bocinas, y tal vez un video.

Por otra parte, si queremos que el receptor del conocimiento aprenda a manejar una bicicleta, lo mejor es que use una. Por este motivo se considera que la mejor manera de aprender es practicando, con lo que el conocimiento adquiere un verdadero significado para el alumno. El polilibro debe ser interactivo y valerse del medio apropiado (bocinas, micrófono, video, mundos virtuales, simulación, etc); además, no debemos olvidar que este material no es un sustituto del profesor, sino más bien un apoyo básico para su trabajo. El alumno no debe perder el ambiente del salón de clase, por esto recomendamos una educación semipresencial, es decir, la asistencia a una sesión de trabajo por lo menos una vez a la semana. Esto mejora el aprovechamiento de las instalaciones, sobre todo considerando que cada vez es más difícil atender la demanda educativa.

---

### CONCLUSIONES

---

El uso de multimedia en la educación básica a nivel ingeniería no se contempla como un hecho aislado, tal y como ocurre en cualquier aula tradicional, sino como un proceso continuo, permanente y personal.

La formación multimedia permite al usuario planificar su horario y programa formativo según sus necesidades concretas y su disponibilidad de tiempo, y puede hacerlo con solo tener los medios al alcance.

A diferencia de la pasividad que caracteriza al alumno asistente a la mayor parte de las clases que se imparten bajo el método tradicional, la formación multimedia está dotada de un alto grado de interactividad y continuo «ejercicio» de pasos prácticos, proporcionando incluso el atractivo de los video juegos.

Se amplía la capacidad de retención en el usuario, al tratarse de un método extraordinariamente visual e interactivo, donde el alumno es el único protagonista. Esto aumenta la motivación y centra el aprendizaje en objetivos concretos, reduciendo el miedo al fracaso.

La capacidad de respuesta del alumno aumenta ya que el usuario recibe atención personalizada según sus necesidades.

Los multimedia permiten que el aprendizaje se dé en un canal de percepción relevante, apoyado por otros canales, además de que el aprendizaje se desarrolla de acuerdo con las necesidades y características de cada usuario. Los multimedia también permiten un aprendizaje más significativo, ya que se aprende haciendo, gracias a la interactividad; para esto es necesaria la aplicación de diversas estrategias basadas no solo en el constructivismo, sino también en el conductivismo, ya que no debemos olvidar que somos personas diferentes y por tal motivo con diversas capacidades para aprender.

Un polilibro debe ser una clase interactiva que considere el medio más adecuado para un aprendizaje significativo, de tal manera que se construya un ambiente virtual de aprendizaje, con las características de un salón de clase, y el atractivo de un video juego, sin llegar al abuso en la aplicación de los multimedia.

---

**BIBLIOGRAFÍA**

---

**Ausbel D. (1983)** *"Psicología Educativa. Un punto de vista"*, Trillas, México.

**Carretero, M. (1993)** *"Constructivismo y Educación"*. EDELVIVES. España.

**Manuel Gandara, (1999)** *"Introducción al uso de los multimedia en educación"*. Dirección de Educación Continua y a Distancia.

# Desarrollo de un Manejador de Dispositivo Utilizando el Modelo Windows Driver Model (wdm) para Windows 98 y 2000

M. en C. Rubén Peredo Valderrama,  
Rubén Aquino  
 CIC-IPN  
 peredo@cic.ipn.mx

**E**l objetivo de éste trabajo es el desarrollo de un manejador de dispositivo que se encargue del control del puerto paralelo de una computadora personal (PC) bajo el esquema de *Windows Driver Model* (WDM) usado en Windows 98 y Windows 2000. Un manejador de dispositivo (*Device Driver*, DD) proporciona una interfase de software para el hardware conectado a una computadora. Los programas de usuario tienen acceso al hardware de una forma establecida, sin tener que preocuparse de los detalles de control del hardware; esto es, el manejador del dispositivo oculta las particularidades del acceso al mismo.

Un manejador es un programa que normalmente se vuelve parte del kernel del sistema operativo al momento de la carga, y hace disponibles uno o más dispositivos a los programas de usuario, representando una pieza de hardware, lógica o física.

En Windows, un manejador siempre hace que el dispositivo aparezca como un archivo para los programas que quieren utilizarlo. Así, para tener acceso al dispositivo solo se requiere abrir el manejador del mismo, e interactuar con él mediante peticiones de lectura y/o escritura.

El modelo compartido de WDM tiene beneficios significativos, ya que soporta ambas plataformas (w98 y w2000), reduciendo el tiempo de desarrollo para los programadores. Tres son los requerimientos para el desarrollo de DD con WDM: multiprocesador, independiente del procesador y Plug and Play (PnP).

Un driver WDM escrito correctamente requiere únicamente de una recompilación para funcionar en un ambiente Windows NT (*New Technology*), manteniendo compatibilidad hacia atrás.

ws 95, ya que los miniports SCSI y NDIS son compatibles a nivel binario con NT. Uno de los beneficios de un modelo compartido (WDM) es que los programadores de DD pueden reducir el tiempo de desarrollo en un 50%, además de que está diseñado para ser compatible hacia atrás.

El WDM proporciona un marco de device drivers para Windows 98 y 2000. Estos sistemas en apariencia son similares, pero trabajan de manera muy diferente: Windows 98 esta basado en DOS, evolucionando desde Windows 1.0 y hasta Windows 95, con un enfoque hacia el área de computadoras de escritorio. Windows 2000 parte de NT y se destina principalmente a equipos servidores.

## 1. INTRODUCCIÓN

El WDM ha tenido diversas modificaciones siguiendo el desarrollo del sistema operativo Windows. Windows 3.1 intentaba dar soporte a una gran cantidad de controladores SCSI diferentes, mientras que NT manejaba miniports; Windows 3.1 fue lanzado sin soporte para miniports SCSI. Se buscaba un modelo compartido para los drivers que comenzó a funcionar con Windo-

La figura 1 muestra la arquitectura de Windows 2000. El software se puede ejecutar en cualquiera de los dos modos; en el modo usuario es

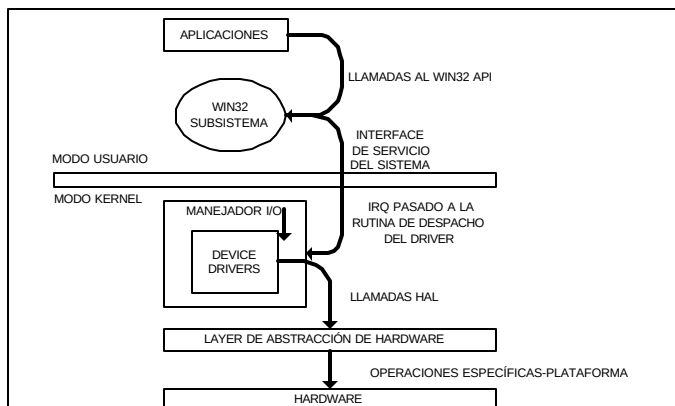


Figura 1. Arquitectura de Windows 2000.

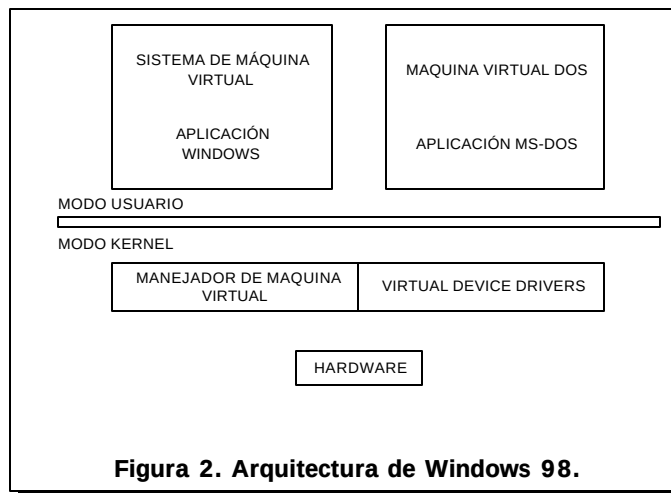


Figura 2. Arquitectura de Windows 98.

inseguro y restringido a ciertas actividades, mientras que en modo kernel es totalmente confiable y capaz de realizar cualquier tarea.

La figura 2 muestra la arquitectura de Windows 98. El kernel se denomina Virtual Machine Manager, VMM, porque su trabajo principal es crear una o más máquinas virtuales que comparten el mismo hardware de la máquina física. El propósito original de un manejador virtual de dispositivo (virtual device driver, VxD) en Microsoft Windows 3.0

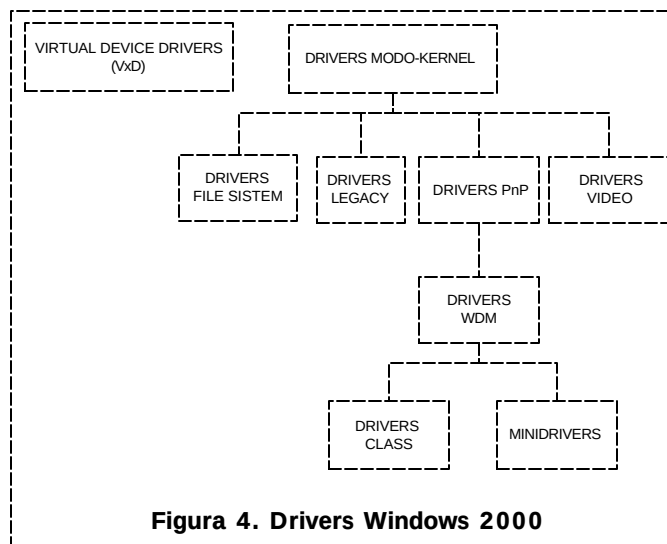


Figura 4. Drivers Windows 2000

La figura 4 muestra los diferentes tipos de drivers que maneja Windows 2000, los cuales son:

- Virtual Device Driver (VDD), es un componente en modo-usuario que permite a las aplicaciones tipo DOS tener acceso al hardware sobre plataformas x86.
- Drivers Modo-Kernel, los cuales incluyen muchas subcategorías. Un driver Plug and Play (PnP) es un driver en modo-kernel que maneja PnP.

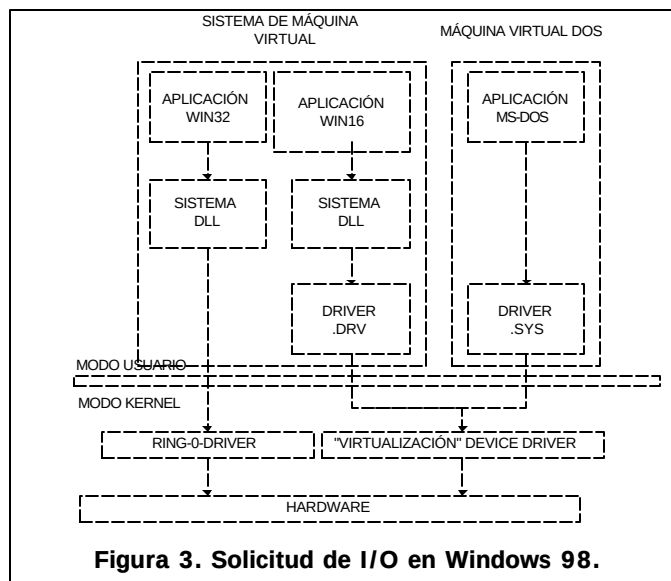


Figura 3. Solicitud de I/O en Windows 98.

era tratar en forma virtual a un dispositivo específico para ayudar a la VMM a crear una simulación, de tal forma que cada una de las máquinas virtuales tenga un complemento total del hardware. La misma arquitectura VMM de Windows 3.0 se utiliza en w98,

cación WIN32 accesa a las solicitudes de I/O por medio de un sistema DLL como KERNEL32.DLL. Pero las aplicaciones pueden únicamente usar ReadFile para leer archivos de disco, puertos de comunicación, y dispositivos que tengan drivers WDM. Para cualquier otra clase de dispositivo, una aplicación debe usar un mecanismo basado en Device IoControl. La columna central corresponde a una aplicación WIN16 ejecutando una solicitud de I/O y en la parte derecha se muestra una aplicación DOS ejecutando una solicitud de I/O.

pero con una agrupación para manejar nuevo hardware y aplicaciones de 32-bits.

La figura 3 muestra como una aplicación tanto WIN32 como WIN16 y DOS realiza solicitudes de I/O en Windows 98. En la columna de la izquierda se indica como una apli-

- Driver WDM, es un driver PnP compatible con Windows 98 y Windows 2000. Dentro de la categoría de los drivers WDM, también se distingue entre los drivers tipo class y los minidivers, los cuales proveen ayuda a un driver class.
- Video drivers: son drivers en modo-kernel para dispositivos de despliegue e impresoras; su función principal es visualizar datos.
- Drivers del sistema de archivos: implementan el modelo de siste-

ma de archivos estándar para la PC en discos duros o sobre conexiones de red.

- Legacy device drivers: son manejadores en modo-kernel, que directamente controlan un dispositivo del hardware sin ayuda de otros drivers. Esta categoría incluye esencialmente drivers para las primeras versiones de Windows NT, que están ejecutándose sin cambio en Windows 2000.

## 2. DESARROLLO DEL TRABAJO Y TECNOLOGÍA UTILIZADA

### 2.1 MANEJADOR DE DISPOSITIVO WDM

Microsoft proporciona varios manejadores genéricos que realizan tareas comunes. Los manejadores de dispositivos pueden usar las características de estos elementos estándar.

La labor de escribir un manejador, por lo tanto, inicia regularmente con la identificación de qué manejador genérico se puede usar. Un manejador de bus de bajo nivel puede usarse para todas las comunicaciones básicas con el hardware. Un manejador de clase intermedia podría proporcionar las características que son comunes a una categoría completa de dispositivos.

### 2.2 EL MODELO DE MANEJADOR DE WINDOWS (WDM)

El WDM tiene dos aspectos separados pero igualmente importantes. Primero, este modelo describe la estructura estándar para manejadores de dispositivos; segundo, Microsoft proporciona una serie de manejadores de bus y clase para dispositivos comunes.

El modelo WDM describe cómo se instalan y configuran inicialmente los dispositivos y cómo deben servir a las peticiones de usuario e interactuar con el hardware. Un manejador WDM debe encajar en el sistema Plug and Play (PnP) que permite a los usuarios conectar dispositivos que pueden configurarse por software.

### 2.2 COMPATIBILIDAD DE CÓDIGO Y BINARIA

Originalmente, Microsoft estableció que los manejadores WDM serían compatibles a nivel binario entre Windows 98 y Windows 2000 x86, y compatibles en código para Windows 2000 en plataformas Alfa. Sin embargo, la compatibilidad de binarios no está garantizada.

Es probable que se tengan problemas cuando no se usan las herramientas para el mismo sistema operativo; esto es, es mejor usar el DDK para Windows 98 para manejadores en Windows 98 y el DDK W2000 para W2000.

### 2.3 MANEJADORES ESTILO WDM VS. NT

La principal diferencia en el código del dispositivo de estos dos estilos radica en la forma en que se reconocen los dispositivos.

En un driver WDM, el Administrador Plug and Play avisa cuando se agrega o quita un dispositivo del sistema. El administrador Plug and Play usa archivos de instalación INF para encontrar el manejador correcto para el nuevo dispositivo. En contraste, un manejador estilo NT tiene que encontrar sus propios dispositivos, regularmente en su rutina de inicialización. Este tipo de manejadores usualmente se instalan usando un programa especial.

Los nuevos manejadores de bus y clase sólo están disponibles para los manejadores WDM. Los manejadores WDM nuevos y los de NT deben soportar las características de administración de energía y Administración de Windows.

### 2.4 COMPONENTES DE UN MANEJADOR DE DISPOSITIVO

Estas son algunas de las tareas que puede hacer un manejador de dispositivo:

- Configurarse inicialmente
- Crear y borrar dispositivos
- Procesar solicitudes Win32 para abrir y cerrar un manejador de archivo.
- Procesar solicitudes Win32 de Entrada/Salida
- Serializar el acceso al hardware
- Comunicarse con el hardware
- Invocar a otros manejadores
- Cancelar solicitudes de E/S
- Limitar (time-out) peticiones de E/S
- Responder si un dispositivo se agrega o remueve durante ejecución
- Manejar las solicitudes de administración de energía.

### 2.5 ENTRADA/SALIDA BÁSICA

En Win32, el acceso a un dispositivo es similar al de un archivo, por lo que se usan las mismas funciones, tales como CreateFile, ReadFile, WriteFile, etc.

### 2.6 AMBIENTE

Cualquier número de hebras (Thread) Win32 podrían tener acceso a un dispositivo en forma simultánea, así que el dispositivo debe contemplar este aspecto y ser capaz de manejarlo. Esto involucra también bloquear el acceso al dispositivo para

que sólo una de estas hebras pueda operar en forma exclusiva el dispositivo en un tiempo determinado. En este sentido, ayuda el administrador de I/O del kernel.

#### PUNTOS DE ENTRADA DEL DISPOSITIVO Y CALLBACKS

El kernel ejecuta regularmente el código del manejador de dispositivo, enviando IRP (I/O Request Packets). La estructura IRP es fundamental para los manejadores.

Un manejador tiene un punto de entrada principal, una rutina que debe llamarse DriverEntry, la cual tiene un prototipo de función estándar. Cuando se carga el driver, el kernel llama a la rutina DriverEntry, y después puede invocar a otras rutinas, denominadas *callbacks*. Los nuevos manejadores también pueden proporcionar una interfase Modelo de Objeto Común (Common Object Model, COM) para el kernel, mediante una serie de rutinas que implementa el manejador.

### 2.7 RUTINAS DE DESPACHO (DISPATCH)

La rutina DriverEntry de un manejador debe establecer o configurar una serie de callbacks para procesar IRP. Además, debe establecer las rutinas Unload, AddDevice y StartIo, si se necesitan.

En la tabla 1 se muestran las funciones de dispositivo de I/O comunes en Win32 y sus IRP correspondientes.

Los manejadores (*handlers*) para las IRP de la tabla 1 se llaman comúnmente rutinas de despacho (*dispatch routines*), porque con frecuencia solo realizan algún proceso inicial del IRP, como por ejemplo, validar los parámetros.

## 2.8 MANEJADORES

Los manejadores pueden generarse en versiones "free" (*release*) o "checked" (*debug*). La versión free debe ser la versión final, es decir, la que ya no tiene símbolos de depuración. La versión checked es la versión de depuración y que no está optimizada, e incluye símbolos de depuración.

El Driver Development Kit (DDK) de Microsoft genera ambos ambientes para la compilación y el enlace, como se muestra en la figura 5.

### 2.9 LENGUAJE Y BIBLIOTECAS

Un manejador es una Biblioteca de Enlace Dinámico (*Dynamic Link Library, DLL*) con la extensión .sys. Comúnmente está escrito en C o C++ y puede incluir recursos tales como un bloque de versión, mensajes de eventos y definiciones de clase de

Instrumentación de Administración de Windows. En Windows 98, el binario de un manejador debe tener un nombre de 8.3, es decir de 8 caracteres como máximo y la extensión .sys.

No debe usarse la palabra clave *new* de C++, ya que ésta puede estar implementada usando *malloc*, que no está disponible para los dispositivos en modo kernel.

### 2.10 DRIVER DEVELOPMENT KIT

Microsoft recomienda usar el Driver Development Kit (DDK) [3] para el desarrollo de manejadores de dispositivos. Esta es una herramienta que contiene documentación acerca de la elaboración de éste tipo de programas, así como algunas herramientas muy útiles para la compilación y el enlace de los programas; es útil también porque maneja algunas características que facilitan la generación del manejador. El DDK trabaja conjuntamente con Visual C++ y es por ello que para su instalación se requiere que dicho programa esté previamente instalado.

En general, existen versiones de DDK para cada una de las versiones de Windows que Microsoft ha liberado. Es recomendable usar el DDK que corresponda a la versión de Windows que se esté utilizando; la razón es que algunas características del kernel han sufrido cambios importantes de ver-

| Tabla 1. Función Win32 IRP |                   |
|----------------------------|-------------------|
| CreateFile                 | Create            |
| CloseHandle                | Close IRP         |
| ReadFile, etc.             | Read IRP          |
| WriteFile, etc.            | Write IRP         |
| DeviceIoControl IOCTL IRP  | IOCTL IRP interna |

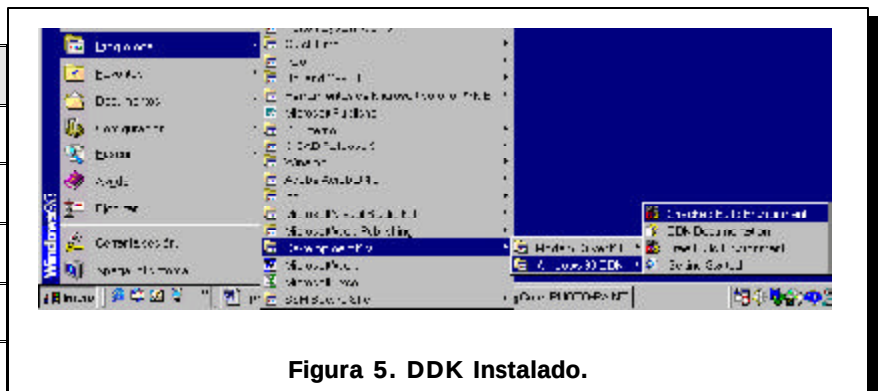


Figura 5. DDK Instalado.

sión a versión y, en un momento dado, es posible encontrarse con algunos problemas.

## 2.11 ALGUNAS RECOMENDACIONES PARA EL CÓDIGO

Es recomendable hacer el código tan sencillo como se pueda y documentarlo bien. Se deben tratar todas las advertencias como si fueran errores a corregir. También hay que asegurarse el cálculo de los valores de retorno para todas las funciones del kernel en uso; se debe tener cuidado en la manera de usar los recursos del kernel.

## 2.12 LA UTILERÍA BUILD

La utilería build de línea de comandos del DDK es la herramienta para construir manejadores. Esta llama a la utilería nmake para construir el manejador usando las configuraciones adecuadas del compilador y del enlazador.

Para esto, debe crearse un archivo SOURCES, un archivo makefile, la estructura de directorio y, opcionalmente, los archivos makefile.inc y dirs.

La utilería build despliega los detalles del proceso y los errores en la salida estándar. Además, genera una lista de los errores en un archivo llamado build.err, las advertencias en build.wrn y una bitácora en build.log.

En la figura 6 se muestra el ambiente que genera el DDK para la construcción de los manejadores.

## 2.13 ARCHIVOS MAKEFILE

La utilería nmake usa el archivo makefile, en el que se encuentran las instrucciones que determinan qué

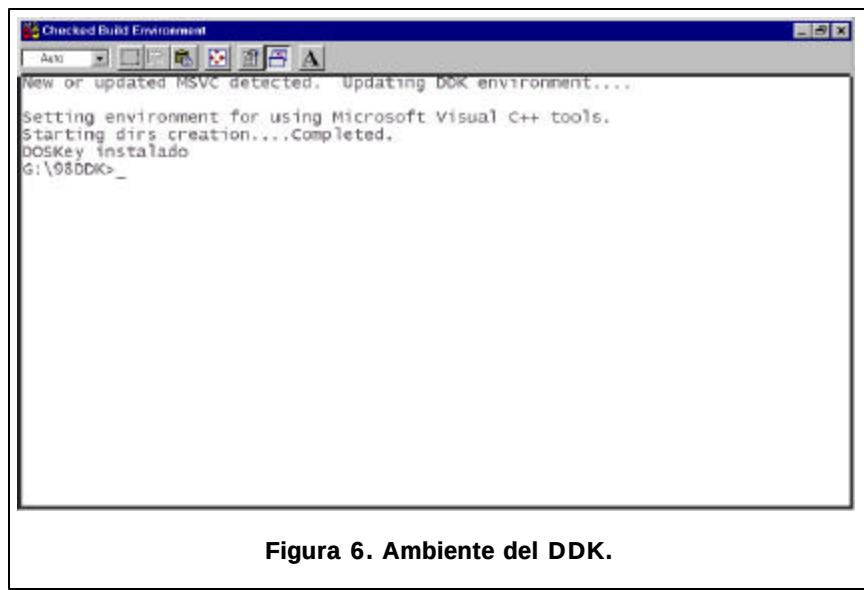


Figura 6. Ambiente del DDK.

comandos ejecutar para actualizar un proyecto. Por ejemplo, la tabla 2 muestra que si el programa prueba.cpp se actualiza, entonces se compila en el archivo prueba.obj usando el compilador cc. prueba.obj se enlaza para crear prueba.exe usando el programa link.

| Tabla 2. Archivo makefile |                                                |
|---------------------------|------------------------------------------------|
| prueba.exe:               | prueba.obj<br>link -o prueba.exe<br>prueba.obj |
| prueba.obj:               | prueba.cpp<br>cc prueba.cpp                    |

## 2.14 ARCHIVO SOURCES (FUENTES)

La utilería build busca un archivo de macroinstrucciones llamado SOURCES en el directorio actual, en el que se especifica qué es lo que se va a construir.

Por ejemplo, para el caso de nuestro manejador, el archivo SOURCES contiene lo siguiente:

```
TARGETNAME=pparallel
TARGETTYPE=DRIVER
DRIVERTYPE=WDM
TARGETPATH=OBJ
INCLUDES=$(BASEDIR)\inc;
SOURCES= Init.cpp \
```

```
Dispatch.cpp \
Pnp.cpp \
DeviceIo.cpp \
DebugPrint.c \
pparallel.rc
```

NTTARGETFILES=PostBuildSteps

Esto le indica a la utilería build que debe construirse un manejador WDM de nombre pparallel, que se pondrá en el directorio OBJ. Se incluye la ruta del directorio inc del DDK para que busque también archivos de cabecera que puedan requerirse. Luego se listan los archivos que deberán compilarse. NNTARGETFILES especifica algunos pasos posteriores a la construcción del manejador.

## 2.15 LOS DIRECTORIOS

En Windows 98 y en Windows NT, la herramienta build siempre pone los archivos objeto en el directorio OBJ\i386 para arquitecturas x86

La macro TARGETPATH del archivo SOURCES especifica dónde se alojará el archivo ejecutable final. Si se especifica OBJ, entonces el ejecutable estará en los directorios OBJ\i386\free y OBJ\i386\checked, según sea el caso.

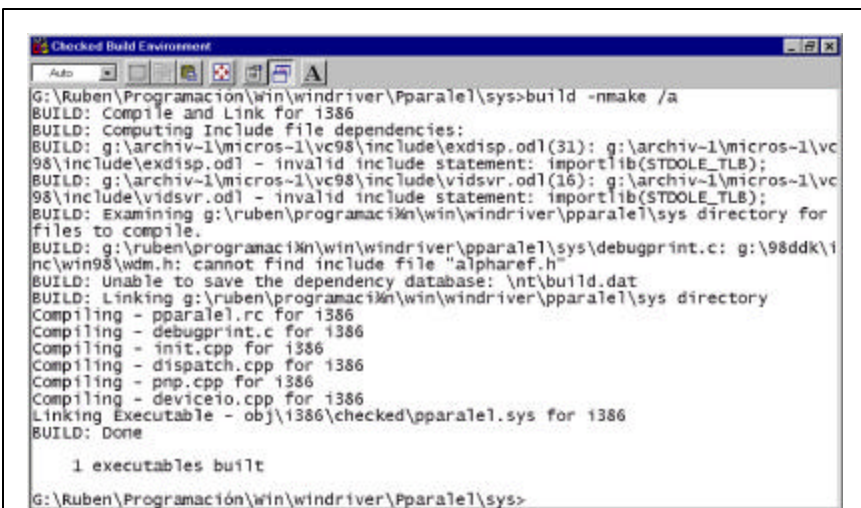


Figura 7. Proceso de compilación del manejador *pparalel* con build.

En Windows 2000, build mantiene separados los archivos objeto free y checked. Si la variable TARGETPATH es OBJ, los archivos objeto de la versión free x86 y el manejador final estarán en el directorio OBJ-FREE\i386 y en OBJCHK\i386 para el caso de la versión checked.

En la figura 7 se muestra el proceso de compilación de nuestro manejador *pparalel*, usando la herramienta build.

## 2.16 ARCHIVOS INF

Un archivo INF contiene toda la información necesaria para instalar un manejador de dispositivo WDM. Esto incluye la lista de los archivos a copiar, las entradas que deben aparecer en el registro de Windows, etc.

Windows proporciona un instalador estándar para la mayoría de las clases de dispositivos y un instalador por omisión, el cual procesa archivos INF.

Un archivo INF es un archivo de texto que se parece a los archivos INI. Tiene secciones, cada una de las cuales inicia con una línea que contiene su nombre encerrado entre paréntesis

cuadrados, y luego el contenido de la sección. Cada línea tiene una entrada simple o asigna un valor. Las secciones pueden estar en cualquier orden; a veces puede ser importante el orden de las líneas dentro de una misma sección. Los comentarios se escriben después del punto y coma.

## SECCIONES ESTÁNDAR

### VERSIÓN

Regularmente se encuentra al inicio del archivo INF y contiene

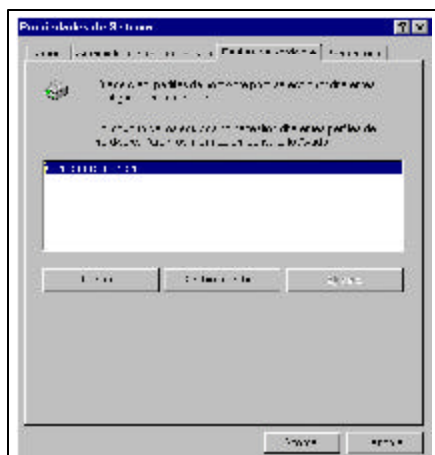


Figura 8. Perfiles de hardware del sistema.

información sobre la versión del manejador.

## STRINGS

Esta sección define cadenas que se sustituyen en alguna otra parte del archivo.

## 2.17 INSTALACIÓN DEL MANEJADOR

Para instalar el manejador creado, es recomendable copiar el perfil de hardware que se está usando para tener un respaldo y poder realizar modificaciones (ver figuras 8 y 9).

Para modificar la copia del perfil de hardware, al que hemos llamado PTO-Paralelo, es necesario reiniciar la máquina e iniciar con éste perfil de

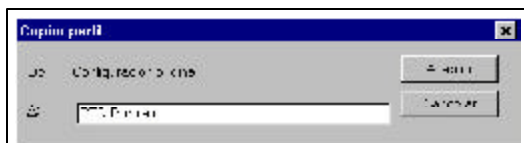


Figura 9. Copiado del perfil de hardware.

hardware. Al reiniciar, Windows encontrará dos perfiles de hardware y solicitará que se elija con cual debe iniciar, en este caso PTO-Paralelo. Después, en el Panel de Control, debe eliminarse el manejador que ya está controlando dicho puerto, como se muestra en las figuras 10 y 11.

En este momento ya puede instalarse el manejador elaborado; para ello es necesario usar la opción Agregar nuevo hardware del Panel de Control (figura 12).

Ahí debe indicársele a Windows que el dispositivo y el controlador serán indicados por el usuario, para evitar que Windows los busque. Windows indicará que falta un controlador para el puerto paralelo y mostrará dicho dispositivo. Debe indicársele que no instale al dispositivo, pues de lo contrario, utilizará su manejador

por omisión, como se muestra en las figuras 13 y 14.

Al seleccionar el dispositivo, debe indicarse que se trata de Otros Dispositivos, como se muestra en la figura 15.

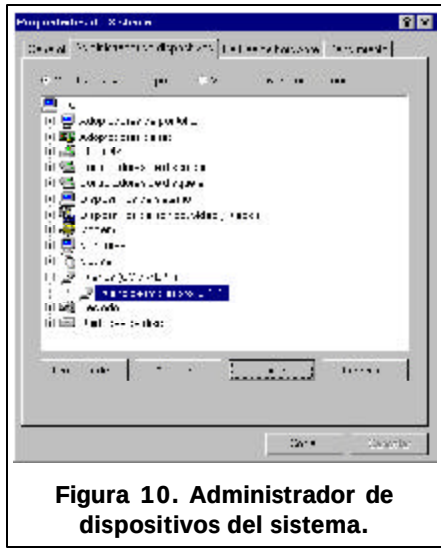


Figura 10. Administrador de dispositivos del sistema.

En el siguiente cuadro de diálogo, figura 16, hay que elegir que se use un disco y entonces se elige el directorio en el que se encuentre el archivo \*.inf correspondiente. Este archivo es el que contiene información importante para que Windows sepa de qué tipo de manejador se trata, el nombre, la ubicación, etc.

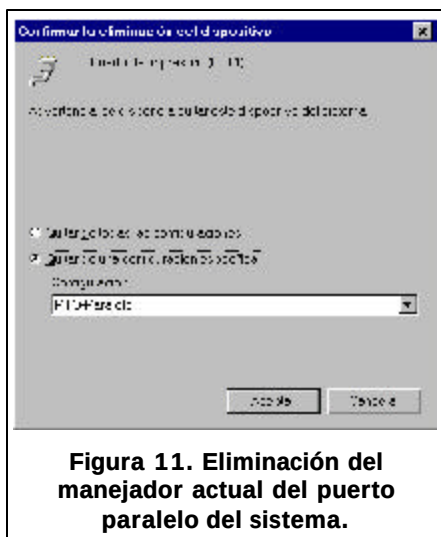


Figura 11. Eliminación del manejador actual del puerto paralelo del sistema.



Figura 12. Panel de control, agregar nuevo hardware.

Aparecerá entonces, un cuadro de diálogo con los manejadores para los cuales se encontraron archivos .inf. Se elige entonces la versión checked del manejador CIC-IPN (figura 17), mostrándose los recursos que el manejador usará (figura 18).

Una vez hecho esto, el manejador ya está instalado y es necesario arrancar nuevamente el sistema para que los cambios tomen efecto.

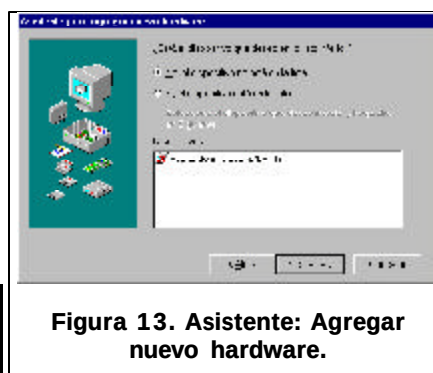


Figura 13. Asistente: Agregar nuevo hardware.

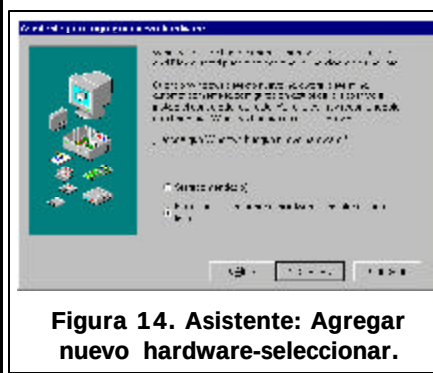


Figura 14. Asistente: Agregar nuevo hardware-seleccionar.



Figura 15. Asistente: Agregar nuevo hardware-otros dispositivos.

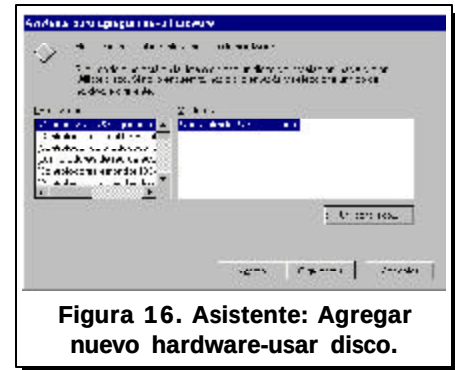


Figura 16. Asistente: Agregar nuevo hardware-usar disco.

Podemos observar al reiniciar, que en el Panel de Control, en la parte de Sistema y en la de Administrador de Dispositivos, se encuentra ya instalado el nuevo manejador (figura19).

## 2.18 PRUEBA DEL MANEJADOR PARA PUERTO PARALELO BAJO AMBIENTE DOS EN WINDOWS 98

En este caso, se elaboró un programa sencillo que sólo envía datos al puerto paralelo y puede observarse la salida en una serie de LED conectados a él (figura 20). El código del programa se muestra en la tabla 3.

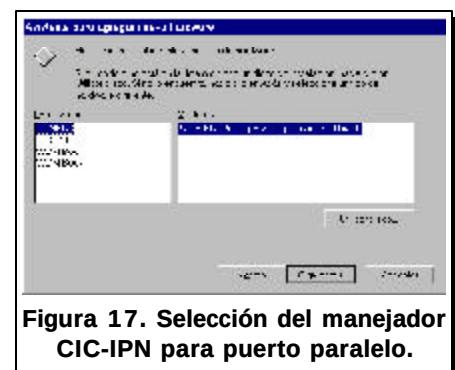


Figura 17. Selección del manejador CIC-IPN para puerto paralelo.



Figura 18. Recursos asignados por el SO para el manejador CIC-IPN.

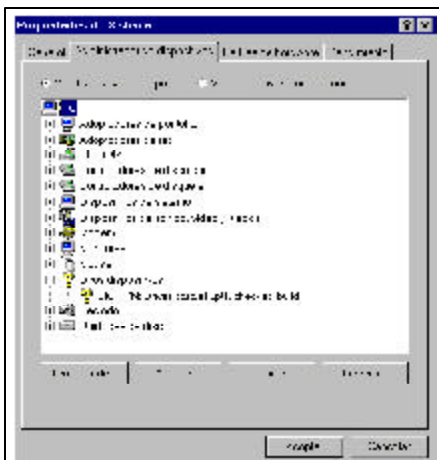


Figura 19. Administrador de dispositivos: manejador para el puerto paralelo CIC-IPN.

## 2.19 PRUEBA DEL MANEJADOR PARA PUERTO PARALELO BAJO AMBIENTE WINDOWS

Tabla 3. Código de comunicación con el manejador

```
do
{
printf("Dato a transmitir ( 1 <= dato <=255 ) (0 =
terminar): ");
scanf("%d",&dato);
while((dato<0 || dato>255) && dato!=1)
{
printf("Dato no válido. Introduzca
nuevamente el
número: ");
scanf("%d",&dato);
}
EnviarByte[2]=dato;
if(DeviceIoControl(hWdmlo,
IOCTL_PHDIO_RUN_
CMDS,EnviarByte,length(EnviarByte),&num,
1,
&BytesReturned, NULL))
{
printf("Comandos en EscribeByte
almacenados. OK.");
}
else{printf("XXX falló el almacenamiento de
comandos
EscribeByte %d\n",GetLastError());
}
}
```

Este programa es similar al anterior, ya que envía datos al puerto paralelo y se observa la salida en una serie de LED. Es importante mencionar que se elaboró el programa utilizando C++Builder de Borland. La ejecución del programa se muestra en la figura 21 y el listado del mismo se incluye en la tabla 4.

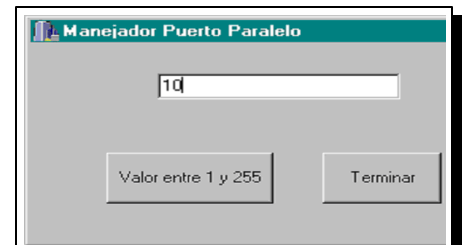


Figura 21. Prueba del nuevo manejador del puerto paralelo (Windows).

Tabla 4. Código de comunicación con el manejador

```
// ABRE EL PUERTO
hComm = CreateFile("LPT1",
GENERIC_READ | GENERIC_WRITE,
0,
0,
OPEN_EXISTING,
0,
0);
```

cesador, debe ser independiente del procesador y PnP; pero un driver WDM escrito correctamente requiere únicamente una recompilación para funcionar en un ambiente NT 64-bits, ya que WDM mantiene la compatibilidad hacia atrás.

Se utilizó DDK de Microsoft para el desarrollo de este manejador, además de que fue necesario instalar Visual C++. Así mismo, se utilizaron varias herramientas adicionales del DDK, como son:

- Build
- Makefile
- Inf
- Sources

Una vez terminado el manejador fue necesario darlo de alta en el SO, y posteriormente se realizaron dos programas sencillos para pruebas del manejador, uno en DOS y otro bajo ambiente Windows.

El proceso de elaboración de un manejador WDM es más complicado que utilizar los puertos en un progra-

## 3. SUMARIO

Este trabajo desarrolló un manejador de dispositivo que se encarga del control del puerto paralelo de una PC bajo el nuevo esquema de Windows Driver Model(WDM) usado en Windows 98 y Windows 2000.

El modelo compartido de WDM tiene beneficios significativos, ya que soporta ambas plataformas reduciendo el tiempo de desarrollo para los desarrolladores de Device Driver(DD). Tres son los requerimientos para el desarrollo de un DD WDM: multipro-

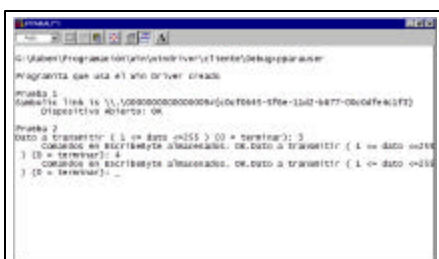


Figura 20. Prueba del nuevo manejador del puerto paralelo (ventana DOS).

ma en lenguaje C. Pero los beneficios son que un pequeño programa en C en los diferentes Windows utilizando el puerto paralelo no funciona, mientras que el utilizar DDK y el nuevo modelo WDM funciona adecuadamente en Windows y manteniendo un control de la tarjeta vía el puerto paralelo.

Actualmente, el desarrollo de estos manejadores en el área de ingeniería de hardware es de gran importancia, ya que los diseñadores de tarjetas requieren una interfaz ya no con DOS sino con Windows. Finalmente, el presente trabajo muestra el desarrollo de un manejador para el puerto paralelo, mostrando todos los pasos y herramientas necesarias para un buen desarrollo y pruebas necesarias para su buen funcionamiento.

---

#### 4. BIBLIOGRAFÍA

---

- [1] *"Writing Windows WDM Device Drivers"*, Chris Cant, R&D Books.
- [2] *"Programming the Microsoft Windows Driver Model"*, Walter Oney, Microsoft Press.

---

#### 5. REFERENCIAS

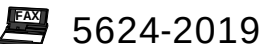
---

- [1] <http://msdn.microsoft.com>
- [2] <http://msdn.microsoft.com/msdnmag>
- [3] <http://www.microsoft.com/ddk>



Centro de Innovación y Desarrollo Tecnológico en Cómputo  
Consejo Editorial Revista POLIBITS  
Av. Té No. 950, Edificio de Graduados 2o. piso.  
Colonia Granjas México. C.P. 08400  
México D. E.

5624-2019, 5624-2018, 5624-2000 ext. 70261



**e-mail** *erodrig@ipn.mx*

Si tu contribución es un artículo, favor de enviarlo por e-mail y anexar los siguientes datos: Nombre del autor, Grado Académico e Institución o Empresa de procedencia.