UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

"DISEÑO Y MODELADO DE UNA ARQUITECTURA VLSI PARA EL RECONOCIMIENTO DE IMÁGENES EN UN SISTEMA DE VISIÓN ARTIFICIAL CON BASE EN LA TRANSFORMADA DISCRETA WAVELET Y MEMORIAS ASOCIATIVAS MORFOLÓGICAS"

TESIS

PARA OBTENER EL TÍTULO DE

INGENIERO EN ELECTRÓNICA

PRESENTA

SELENE ALVARADO LEGARIA

DIRECTOR DE TESIS

M.C. ENRIQUE GUZMÁN RAMÍREZ

HUAJUAPAN DE LEÓN, OAX.; AGOSTO DE 2007

Tesis presentada el 17 de Agosto de 2007
ante los sinodales:

Dr. Antonio Orantes Molina
M.C. Felipe Santiago Espinosa
C. Dr. Heriberto. I. Hernández Martínez

Director de Tesis:
M.C. Enrique Guzmán Ramírez

Dedicatoria

Con mucho cariño y admiración, a mis padres Carlos y Coral.

Selene.

Agradecimientos

A mis padres que siempre estuvieron ahí, y sin cuyo apoyo no hubiera podido concluir este trabajo de tesis.

A mi familia por sus consejos y apoyo moral.

A mis compañeros y amigos que de una u otra manera me soportaron y alentaron durante la licenciatura y el desarrollo de este trabajo.

A Eric, Edel y el Profr. Heriberto por su amistad, apoyo y consejos en la realización de este trabajo de tesis.

A mi director de tesis por compartirme sus conocimientos, su amistad y por su interés en la dirección y término de ésta.

A mis sinodales por la valiosa aportación a este trabajo, así como también a mis maestros de licenciatura a los que debo mi formación académica y demás profesores amigos.

A todos, muchas gracias.

Selene.

Índice

Índice de figuras

Índice de tablas

Introducción

El presente trabajo expone un algoritmo para el reconocimiento de imágenes y/o patrones implementado en un Dispositivo Lógico Programable, FPGA (*Field Programmable Gate Array*, Arreglo de Compuertas Programable en Campo), con la finalidad de obtener una arquitectura VLSI (*Very Large Scale Integration*, Integración en escala muy grande) para el reconocimiento de imágenes. Este algoritmo hace uso, por una parte de la Transformada Discreta Wavelet 2D (DWT, *Discrete Wavelet Transform*) cuya principal función es la de reducir el tamaño de la imagen, esto con la finalidad de ocupar menos recursos de hardware; por otra parte el reconocimiento de la imagen o patrón es hecho mediante el uso de las Memorias Asociativas Morfológicas (MAM, *Morphological Associative Memories*). Se pretende que este algoritmo sea aplicado a la visión artificial en robots móviles.

Para el modelado de dicho algoritmo en un FPGA, se propone el uso de un Lenguaje Descriptor de Hardware (HDL, *Hardware Description Language*) el cual facilita el modelado o descripción de un sistema digital. El HDL elegido para la descripción del algoritmo es el VHDL (*Very High Speed Integrated Circuit Hardware Description Language*, Lenguaje Descriptor de Hardware para Circuitos Integrados de muy Alta Velocidad) por tratarse de un lenguaje estandarizado [24], [34].

La DWT, es una técnica que se aplica a toda la imagen, se trata de una descomposición de una señal en frecuencias. En el proceso de análisis de los Wavelets las señales son representadas utilizando un grupo de funciones básicas producidas por el desplazamiento y escalamiento de una función madre o función principal. En cada iteración, la DWT unidimensional descompone a una señal de entrada, $f_o(n)$, en una parte de promedio, $f_i(n)$ (L), y otra de detalle, $W_i(n)$ (H), donde *i* representa el nivel de aplicación de la transformada.

Debido a que una imagen es una señal de dos dimensiones, $f_0(x,y)$, para el tratamiento de éstas se hace uso de la DWT bidimensional, obteniendo como resultado una función de escalamiento o parte promedio $f_i(x,y)$ (LL) y 3 Wavelets bidimensionales o partes detalle $W_i^H(x,y)$ (HL), $W_i^V(x,y)$ (LH) y $W_i^D(x,y)$ (HH) [1], [9], [13], [26].

En el diseño de la arquitectura propuesta se hace uso únicamente de la función de escalamiento para la siguiente etapa del algoritmo, descartando las partes de detalles.

La etapa de reconocimiento de la imagen o patrón es implementada utilizando Memorias Asociativas, particularmente MAM desarrolladas por Rittet, Sussner y Díaz de León, las MAM

basan su funcionamiento en las operaciones morfológicas de dilatación y erosión, es decir, hacen uso de máximos o mínimos de sumas [36].

Las MAM han demostrado ser una excelente herramienta en el reconocimiento y recuperación de patrones, sin importar que éstos contengan ruido aditivo, sustractivo o mixto [7], [53], [54]. Características que hacen atractivas a las MAM para desempeñar el reconocimiento de imágenes o patrones son, la robustez al ruido, la capacidad de almacenamiento, la capacidad de aprendizaje, la rapidez en el proceso de aprendizaje y la eficiencia y velocidad en el proceso de recuperación de patrones.

Marco teórico

La visión artificial es un sistema que los robots pueden utilizar para inferir el estado de su entorno. Se trata de un sistema complejo que entrega un flujo continuo de información que debe de ser procesado para intervenir directamente en la acción o comportamiento de un robot.

La visión del robot se puede definir como el proceso de extraer, caracterizar e interpretar información de imágenes de un mundo tridimensional. Este proceso, también comúnmente conocido como visión de máquina, se puede subdividir en seis áreas principales [URL13]:

- Captura.
- Pre-procesamiento.
- Segmentación.
- Descripción.
- Reconocimiento.
- Interpretación.

Estas áreas de la visión se agrupan, de acuerdo con la sofisticación que lleva su desarrollo, en tres niveles de procesamiento [12], [21], [URL10], [URL14]:

- **Nivel Bajo:** las técnicas utilizadas en este nivel son básicas y están orientadas a la definición y obtención de las propiedades generales de la imagen. Esta fase involucra tareas tales como:
  - Captura de la imagen: hecha mediante cámaras.
  - Pre-procesamiento: en la cual se contempla la digitalización de la señal de video; la obtención de las propiedades más representativas, como los bordes, el color, la textura, etc.; y la mejora de la imagen.
- **Nivel Intermedio:** en este nivel se incluyen las técnicas empleadas para obtener las propiedades de la escena.
  - Segmentación: técnica mediante la cual se extraen o aíslan los objetos particulares de la imagen.
  - Descripción: empleada para la caracterización de los objetos aislados.
  - Reconocimiento: identificación y etiquetado de los objetos de la escena.
- **Nivel Alto:** en esta categoría se aplica el proceso inteligente, la técnica más representativa es la:
  - Interpretación: que trata de estudiar la lógica de los objetos localizados en la escena, intentando emular el conocimiento.

Como son de particular interés para este trabajo las técnicas de descripción pertenecientes al nivel intermedio, a continuación se hace mención de las más utilizadas en este nivel de procesamiento

El reconocimiento de patrones es una disciplina científica de amplio uso en la etapa de descripción dentro de un sistema de visión artificial y cuyo principal objetivo es el de clasificar objetos en un número de categorías o clases [40].

Dependiendo de la aplicación, estos objetos pueden ser imágenes, señales o cualquier tipo de medición que necesita ser clasificada. A estos objetos se les denomina en forma genérica *patrones*.

Se puede decir que un *patrón* es un objeto que posee ciertas características que le permiten distinguirse de otros objetos que tienen propiedades similares. Los patrones que poseen ciertas similitudes entre ellos se reúnen en un conjunto denominado *clase de patrones* o simplemente *clase.* Las características propias que distinguen a los patrones en diversas clases se les llaman *descriptores*.

El reconocimiento de patrones tiene 4 enfoques [URL5]:

- **Reconocimiento Estadístico de Patrones:** este enfoque se basa en la teoría de probabilidad y estadística y supone que se tiene un conjunto de medidas numéricas con distribuciones de probabilidad conocidas y a partir de ellas se hace el reconocimiento.
- **Reconocimiento Sintáctico de Patrones:** este enfoque se basa en encontrar las relaciones estructurales que guardan los objetos de estudio, utilizando la teoría de lenguajes formales. El objetivo es construir una gramática que describa la estructura del universo de objetos.
- **Redes Neuronales Artificiales:** este enfoque se basa en una estructura de neuronas interconectadas que se estimulan unas a otras, las cuales pueden ser "entrenadas" para dar una cierta respuesta cuando se le presentan determinados valores.
- **Reconocimiento Lógico Combinatorio de Patrones:** este enfoque se basa en la idea de que el modelado del problema debe ser lo más cercana posible a la realidad del mismo, sin hacer suposiciones que no estén fundamentadas.

Planteamiento del problema

Como se ha mencionado anteriormente, un sistema de visión artificial está formado por varias etapas las cuales se encuentran clasificadas en tres niveles de acuerdo con la complejidad que requiere su implementación. Cada una de estas etapas puede ser efectuada utilizando diversas técnicas, con sus respectivas ventajas y desventajas.

El algoritmo propuesto en este trabajo de tesis interviene en el nivel intermedio de un sistema de visión artificial, particularmente en la función de identificación de imágenes, teniendo por objetivo la identificación, etiquetado y reconocimiento de imágenes digitales, es decir, identificar a cada objeto segmentado de una escena y asignarle una etiqueta o referencia para su posterior reconocimiento.

Por lo tanto, se plantea el modelado de una arquitectura VLSI de un algoritmo basado en la DWT y las MAM para el reconocimiento de imágenes, el cual pueda ser aplicado al nivel intermedio de un proceso de visión artificial.

Justificación

En el planteamiento del problema se ha delimitado la extensión del presente trabajo de tesis, el cual se enfocará únicamente a las etapas de descripción, etiquetado e identificación de un sistema de visión artificial, técnicas empleadas para la caracterización de los objetos aislados.

Se ha mencionado que la visión artificial es un proceso altamente complejo, debido a esto y con la finalidad de volverlo lo más eficaz posible, es deseable que cada una de sus etapas sea efectuada utilizando los menos recursos posibles, con una baja complejidad computacional y en el menor tiempo de procesamiento posible.

Al tratarse de un diseño VLSI, se debe limitar el uso de recursos de memoria que serán utilizados por el sistema. El utilizar la DWT, como una técnica de reducción de información de la imagen, en una etapa previa al reconocimiento de la imagen o patrón, tiene por función reducir los requerimiento de memoria que el sistema demande; de la misma forma, al ser menos los datos a procesar también repercute directamente en la velocidad de procesamiento, volviendo más eficaz al sistema y sin repercusión en la respuesta del mismo.

La etapa de descripción y reconocimiento de la imagen o patrón es implementada utilizando MAM, éstas basan su funcionamiento en las operaciones morfológicas de erosión y dilatación, la complejidad computacional se ve reducida drásticamente ya que las operaciones utilizadas son muy simples: máximos o mínimos de sumas.

La propuesta de usar MAM en esta etapa fundamenta sus bases principalmente en su alta velocidad de procesamiento, su alta inmunidad al ruido, y se trata de una técnica usada en el reconocimiento de patrones que a diferencia de otras técnicas no requiere converger para realizar una respuesta perfecta.

Objetivos

El objetivo general del presente trabajo de tesis es:

Diseñar y modelar una arquitectura VLSI para el reconocimiento de imágenes en un sistema de visión artificial con base en la DWT y MAM's.

Para cumplir con lo anterior se plantean los siguientes objetivos secundarios:

- Implementar un medio de transferencia de información de alta velocidad entre la PC (proveedora de la imagen) y el FPGA, utilizando una interfaz USB (*Universal Serial Bus*).
- Implementar la DWT de 2 dimensiones en el FPGA, para reducir el tamaño de la imagen a procesar sin pérdidas significativas de información.
- Implementar el algoritmo de Memorias Heteroasociativas Morfológicas en el FPGA y comprobar su eficiencia en dicho componente.
- Implementar un módulo central de proceso que permita el control y la interacción entre los módulos anteriores.

Figura I.1. Ciclo de desarrollo de un sistema empotrado.

Metodología de desarrollo

Como resultado del modelado del algoritmo propuesto para cumplir con el objetivo de este trabajo se obtiene un procesador de aplicación específica, el cual se denominará de aquí en adelante como "Procesador para el reconocimiento de imágenes", por tanto, se propone seguir la metodología de diseño de un sistema empotrado para la elaboración del sistema final.

Un sistema empotrado también llamado embebido, es una combinación de hardware, software y, eventualmente, componentes mecánicos diseñados para realizar una función determinada [20].

La figura I.1, muestra una representación esquemática del ciclo de desarrollo de un sistema empotrado, mismo que será utilizado para la implementación del presente trabajo.

El diagrama general del sistema propuesto en este trabajo de tesis puede ser apreciado en la figura I.2 y se divide en los siguientes bloques:

- **Interfaz con el usuario:** esta parte del sistema se encarga de proporcionar al procesador para el reconocimiento de imágenes el conjunto de imágenes a procesar y además muestra al usuario los resultados del procesamiento sobre la imagen de cada una de las etapas que forman el algoritmo. El software para la interfaz implementada en la PC es compilado utilizando la herramienta Borland C++ Builder en su versión 6.0. La dimensión de las imágenes utilizadas es de 720 X 480 píxeles en escala de grises.
- **Interfaz de alta velocidad:** se trata de una interfaz utilizando un puerto USB entre una computadora personal (PC) y el procesador para el reconocimiento de imágenes.
- **Sistema de memoria:** es un sistema de memoria SRAM (*Static Random Accsess Memory*, Memoria Estática de Acceso Aleatorio) organizado como un bloque de memoria de 512K X 16 bits. Está integrado por dos bancos de memorias cada uno de 512K X 8 bits. Tiene por función servir como buffer de almacenamiento, el cual contendrá a la Memoria Asociativa Morfológica generada por el módulo de aprendizaje y a la imagen a procesar proveniente de la PC.

Figura I.2. Esquema general de la solución propuesta.

- **Procesador para el reconocimiento de imágenes:** es la parte central de este trabajo de tesis, ya que en ésta se realiza el modelado del algoritmo de reconocimiento de imágenes propuesto, además de los módulos adicionales que intervienen en el funcionamiento del sistema completo. El modelado o descripción de cada uno de los módulos que intervienen en esta parte del sistema es hecho con el lenguaje descriptor de hardware VHDL utilizando la herramienta para la automatización del diseño electrónico (EDA, *Electronic Design Automation*), ISE Foundation versión 6.3i, e implementado en un FPGA XCV300 de la familia Virtex, ambos de la compañía Xilinx.
- **Actuadores:** con la finalidad de emular el control que se tendrá sobre los actuadores de un robot móvil, se han conectado al sistema dos motores de corriente directa. El control de los motores es hecho utilizando 2 puentes H a base de transistores que proporcionan la corriente requerida por dichos actuadores.

La implementación del procesador para el reconocimiento de imágenes fue hecha en un FPGA XCV300 de la familia Virtex de la compañía Xilinx, con la finalidad de obtener un algoritmo que pueda ser implementado en cualquier FPGA de la compañía Xilinx, y casi en cualquiera de otras compañías, se propone modelar al procesador sin considerar características específicas del FPGA utilizado, además como ya se mencionó se hace uso de un lenguaje estandarizado, VHDL, debido a las ventajas que éste otorga.

Para el modelado del procesador para el reconocimiento de imágenes se utilizó la metodología *Top-Down* (descendente), la cual requiere un nivel de abstracción alto, para después ir dividiendo el diseño en módulos jerárquicamente inferiores, e incrementando el nivel de detalle de cada uno de ellos según se requiera [31].

El algoritmo propuesto para el diseño del procesador para el reconocimiento de imágenes consta de dos fases:

1. **Fase de aprendizaje.** En esta etapa se crea la MAM, la cual albergará la información necesaria para identificar a cada una de las imágenes.
    i. Adquisición de la imagen o patrón y asignación de su correspondiente etiqueta, cada uno de estos conjuntos forma un elemento del conjunto fundamental de asociaciones.
    ii. Aplicación a la imagen de la DWT bidimensional de nivel 4.
    iii. Binarización de la función de escalamiento o parte promedio obtenida de la DWT bidimensional.
    iv. Obtención de la MAM aplicando el algoritmo correspondiente y su almacenamiento en el buffer de memoria asignado.

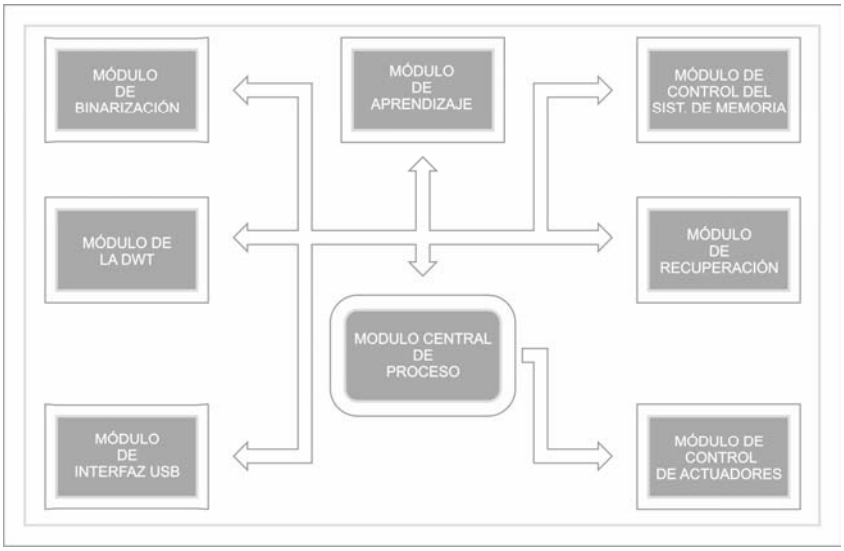

Figura I.3. Procesador para el reconocimiento de imágenes.

v. Repetir los pasos i a iv para cada uno de los elementos que forman el conjunto fundamental de asociaciones

2. **Fase de reconocimiento.** Esta etapa, al recibir una nueva imagen y con ayuda de la memoria creada en la fase anterior, será capaz de identificar cada imagen, generando la etiqueta asociada a la misma y en función de esta etiqueta determinar la acción a seguir.

   i. Adquisición de la imagen a identificar.

   ii. Aplicación a la imagen de la DWT bidimensional de nivel 4.

   iii. Binarización de la función de escalamiento o parte promedio obtenida de la DWT bidimensional.

   iv. Identificación de la imagen con ayuda de la Memoria Asociativa Morfológica obtenida en la fase de aprendizaje. Como resultado se consigue la etiqueta correspondiente a la imagen a identificar.

   v. Entrega la etiqueta obtenida a la etapa superior siguiente para su procesamiento.

Siguiendo la metodología de diseño descendente, el procesador para el reconocimiento de imágenes se puede dividir en los siguientes módulos y puede ser apreciado en la figura I.3:

- **Módulo central de proceso:** este módulo se encarga de gobernar el funcionamiento y la interacción de todos los módulos que forman al procesador de aplicación específica.
- **Módulo de interfaz USB:** la función de este módulo es proveer una interfaz de alta velocidad para la transferencia de información entre la PC y el procesador para el reconocimiento de imágenes. Mediante este módulo el procesador recibirá las imágenes o patrones para su posterior procesamiento.
- **Módulo de control del sistema de memoria:** implementa el protocolo de interfaz mediante el cual cualquier módulo del procesador para el reconocimiento de imágenes puede acceder al sistema de memoria.
- **Módulo de la DWT:** implementa el algoritmo de la Transformada Discreta Wavelet bidimensional sobre la imagen o patrón a procesar.

- **Módulo de binarización:** implementa el modelado de la binarización de la función de escalamiento o parte promedio obtenida de la DWT bidimensional.
- **Módulo de aprendizaje:** tiene por función implementar el algoritmo de aprendizaje de una Memoria Asociativa Morfológica, haciendo uso del conjunto fundamental de asociaciones, formado por imágenes, vectores de entrada, y etiquetas, vectores de salida. La MAM obtenida de este proceso es almacenada en el sistema de memoria.
- **Módulo de recuperación:** implementa el algoritmo de reconocimiento de una imagen, hace uso de la MAM almacenada en el sistema de memoria y genera una etiqueta o vector de salida para la imagen o vector de entrada.
- **Módulo de control de actuadores:** el control de actuadores está compuesto por dos puentes H de transistores que controlan dos motores de corriente continua. Este módulo modificará el comportamiento de los actuadores en función de la etiqueta obtenida por el módulo de recuperación.

1. Estado del arte del reconocimiento de patrones de un sistema de visión artificial

1.1. Sistema de visión artificial

La visión es un sistema que los robots pueden utilizar para inferir el estado de su entorno. Se trata de un sistema muy complejo que entrega un flujo continuo de información que debe de ser procesado para intervenir directamente en la acción o comportamiento de un robot.

La visión del robot se puede definir como el proceso de extraer, caracterizar e interpretar información de imágenes de un mundo tridimensional. Este proceso, también comúnmente conocido como visión de máquina, se puede subdividir en seis áreas principales [URL13]:

- Captura.
- Pre-procesamiento.
- Segmentación.
- Descripción.
- Reconocimiento.
- Interpretación.

Estas áreas de visión se agrupan, de acuerdo con la sofisticación que lleva su desarrollo, en tres niveles de procesamiento: nivel alto, nivel intermedio y nivel alto [12], [21], [URL10], [URL14].

Aunque estos niveles son aparentemente secuenciales, en muchas aplicaciones se requiere una interacción directa entre las diferentes áreas que los componen, incluyendo retroalimentación de los niveles altos a los inferiores.

1.1.1. Nivel bajo de procesamiento

Las técnicas utilizadas en este nivel son básicas y están orientadas a la definición y obtención de las propiedades generales de la imagen.

Esta fase involucra tareas tales como: la captura de la imagen, hecha mediante cámaras; y el pre-procesamiento, en la cual se contempla la obtención de las propiedades más representativas de la imagen.

1.1.1.1. Adquisición de imágenes

En general, esta etapa consiste en la obtención de imágenes digitales a partir de entradas analógicas, lo que puede conseguirse mediante dispositivos tan heterogéneos entre sí, como pueden ser cámaras fotográficas, cámaras de vídeo, escáneres, aparato de resonancia magnética o cualquier otro dispositivo que convierta una imagen analógica o refleje una situación real en una imagen digital.

1.1.1.2. Pre-procesado

La etapa de pre-procesamiento consiste en transformar los datos digitales en un formato más adecuado para etapas posteriores, consiguiendo imágenes que contengan características más adaptadas y preparadas para las tareas de clasificación que se deseen realizar.

Existen diversos tipos de pre-procesamiento de las imágenes, siendo algunas de las más destacables, las de manipulación de contraste, eliminación de ruido, realce de bordes, suavizado de imagen, detección de bordes, etc. A continuación se describen algunos de ellos [12], [19], [URL5], [URL10]:

- **Manipulación de contraste:** se lleva a cabo con operaciones puntuales sobre los píxeles de la imagen tales como binarización, contraste, ecualización o inversión.
- **Filtrado espacial:** conocido como convolución, es usado para el suavizado de imágenes, detección de bordes, y otros efectos. La forma de operar de los filtros espaciales es por medio de la utilización de máscaras que recorren toda la imagen centrando las operaciones sobre los píxeles que se encuadran en la región de la imagen original que coincide con la máscara y el resultado se obtiene mediante un cálculo (suma de convolución) entre los píxeles originales y los diferentes coeficientes de las máscaras. El filtrado, realizado por convolución, puede ser de cuatro tipos básicos en función de su modo de actuar: paso-bajo, paso-alto, pasa-banda y elimina-banda.
- **Filtrado frecuencial:** consigue agrupar en componentes armónicas un tipo de información disperso en la extensión espacial, lo que facilita en algunos casos el filtrado, la identificación de objetos, y la compresión de imágenes sin perder más calidad de la prevista. Su modo de proceder consiste en transformar la imagen, empleando técnicas como puede ser la Transformada Rápida de Fourier (FFT), para posteriormente realizar el filtrado mediante la convolución o la correlación (usada en la identificación de patrones) en el dominio transformado, y finalmente se retorna al dominio espacial con la transformada inversa correspondiente.
- **Filtrado no lineal:** los filtros no lineales operan sobre entornos. Su operación se basa directamente en los valores de los píxeles en el entorno en consideración. Los filtros mínimo, máximo y de mediana son conocidos como filtros de rango; el filtro de mediana tiene un efecto de difuminado de la imagen, mientras que el filtro de máximo se emplea para buscar los puntos más brillantes de una imagen produciendo un efecto de erosión, y el filtro de mínimo se emplea con el objetivo contrario, buscar los puntos más oscuros de una imagen produciendo un efecto de dilatación.
- **Procesado morfológico:** los operadores morfológicos se relacionan más directamente con la forma de los objetos y no con el posible ruido presente en la imagen, de ahí que son utilizados en entornos industriales en los que se precisa la detección o identificación de defectos. Los operadores básicos para imágenes binarias son, la dilatación y la erosión. Posteriormente estos operadores se pueden combinar produciendo la apertura

(*opening*) y la clausura (*closing*). En la práctica, los operadores de dilatación y erosión se suelen utilizar de dos en dos, aplicando una dilatación y luego una erosión o al revés. De esta manera se consigue el efecto de eliminar pequeños agujeros y salientes sin modificar el resto del contorno. La apertura consiste en aplicar una erosión seguida de una dilatación con el mismo elemento estructurante. El resultado es la desaparición de objetos más pequeños que la máscara utilizada y de nexos entre objetos demasiado finos. La clausura, por su parte, consigue el efecto complementario al utilizar primero la dilatación y luego la erosión, tapando agujeros pequeños y entrantes finos, consiguiendo suavizar el contorno.

1.1.2. Nivel intermedio de procesamiento

En este se incluyen las técnicas empleadas para obtener las propiedades de la escena, tales como: segmentación, técnica mediante la cual se extraen o aíslan objetos particulares de la imagen; descripción, empleada para la caracterización de los objetos aislados; reconocimiento, identificación y etiquetado de los objetos de la escena.

1.1.2.1. Segmentación

Es un término general que se aplica a varios métodos de reducción de datos; su objetivo es aislar los elementos de interés de una escena con la finalidad de comprenderla.

La investigación sobre la segmentación de imágenes discurre básicamente por dos direcciones: la estática y la dinámica. En la primera sólo se dispone de una imagen, la segunda se emplea en aquellos entornos donde el movimiento es importante, y por tanto, se puede acudir a información de las imágenes anteriores y posteriores en la secuencia para segmentar la actual.

En los métodos de segmentación aplicados a imágenes dinámicas destacan los basados en el flujo óptico, los basados en correspondencias y los métodos iterativos.

Los métodos de segmentación aplicados a imágenes estáticas suelen ser de dos tipos, los basados en el crecimiento de regiones, atendiendo consideraciones como la fijación de umbral y el crecimiento de región, y los basados en técnicas de detección de puntos de borde, como la detección de bordes, regiones, colores, texturas, etc. [12], [URL10].

- **La fijación de umbrales:** es una técnica de conversión binaria en la que cada píxel es convertido a un valor binario, blanco o negro. Esto se realiza mediante un histograma de frecuencias de la imagen y estableciendo la intensidad (nivel de gris), determinando de esta forma el límite entre el blanco y el negro. Es la técnica más utilizada para la segmentación en aplicaciones de visión industrial, ya que la iluminación se puede controlar en un establecimiento industrial y es más rápida y más fácil de llevar a cabo. Los métodos mas utilizados hacen uso de las máscaras de Roberts, de Prewitt y de Sobel.
- **El crecimiento de región:** es un conjunto de técnicas de segmentación en la que los píxeles se agrupan en regiones llamados elementos de cuadricula basada en similitudes de atributos. Las regiones definidas se podrán examinar en cuanto a si son independientes o se pueden fusionar a otras regiones por medio de un análisis de la diferencia de sus propiedades medias y su conectividad espacial. El crecimiento de regiones es una técnica que agrupa píxeles o subregiones en regiones más grandes; un método para este crecimiento, es la adición de píxeles, en donde comienza con un conjunto de *puntos-semilla* y se hace crecer las regiones añadiendo a cada *punto-semilla*

aquellos píxeles vecinos que tengan propiedades similares como intensidad, textura o color.

- **La detección de colores:** el color en un punto de una imagen puede representarse por tres números que representen, por ejemplo, los niveles de las componentes roja, verde y azul. Entonces una imagen digital a color es representada por 3 matrices de valores. Si esos niveles de los píxeles se grafican como puntos de un espacio cromático, el objeto y el fondo originan cúmulos de puntos. Tales cúmulos son análogos a los picos del histograma y la imagen se puede segmentar en regiones de diferente color dividiendo el espacio cromático de tal manera que se separen los cúmulos. Éste ha sido el método clásico utilizado para segmentar las imágenes obtenidas por los sensores remotos multiespectrales, pero aún no tiene utilización extensiva en visión robótica.
- **Análisis de textura:** si un objeto no presenta un brillo uniforme, sino que muestra cierta "trama", ni el umbral ni la detección de bordes son útiles para extraerlo, ya que sus píxeles no poseen niveles de gris en un rango estrecho y presenta muchos bordes internos. No obstante, tal objeto puede ser distinguible de su vecindad basándonos en su trama o patrón característico de niveles de gris o textura visual. Las texturas visuales se pueden caracterizar por conjuntos de propiedades locales de sus píxeles, o sea por el hecho de que en una región texturizada tienden a presentarse ciertos patrones locales de niveles de gris en la vecindad de cada píxel. Un método potente para caracterizar texturas consiste en realizar varios tipos de operaciones de adelgazamiento y expansión de regiones y analizar sus resultados; por ejemplo, las tramas visuales delgadas desaparecen al aplicar un poco de adelgazamiento, mientras las tramas con espaciamientos pequeños se funden al aplicar un poco de expansión.
- **La detección de borde:** considera el cambio de intensidad que se produce en los píxeles en el contorno o bordes de un objeto. Dado que se ha encontrado una región con atributos similares, pero se desconoce la forma del contorno, este último se puede determinar mediante un simple procedimiento de seguimiento del borde. El método clásico de detectar bordes en una imagen es aplicar un operador derivada isotrópico, tal como el gradiente o aproximaciones discretas al gradiente, como el operador cruzado de Roberts, y el operador de Sobel usado en los primeros sistemas de visión de Stanford. Otros métodos básicos de detección de bordes incluyen la comparación de la vecindad de cada píxel con patrones o plantillas de Funciones Paso en diferentes orientaciones, y en ajustar una superficie polinomial a los niveles de gris en la vecindad de cada píxel.

1.1.2.2. Descripción

Una vez que la imagen ha sido segmentada en regiones, el siguiente paso consiste en la extracción de características o descriptores, el procesamiento de la imagen, que se realiza hasta obtener el vector de características, puede realizarse globalmente (toda la imagen) o de forma local (un objeto dentro de la misma), ya que, en ocasiones, es necesario detectar o extraer las estructuras de objetos o regiones de interés de la imagen en sí.

El vector de características debe de componerse de propiedades de la imagen más o menos invariantes frente a rotaciones, traslaciones y escalados, para que el sistema sea lo más independiente posible del método utilizado para su adquisición, y ésta siga manteniendo, dentro de lo posible, valores aproximados para las mismas, tanto para el caso global como local.

Las características básicas y medidas para la identificación de un objeto bidimensional son las siguientes [12]:

- Nivel de gris (máximo, medio, mínimo).
- Área.
- Perímetro.
- Excentricidad o elongación.
- Los momentos.
- Descriptores topológicos.
- Codificación del contorno.
- Curvatura y longitud del arco.
- Puntos críticos, de inflexión, terminales o de intersección.

Los descriptores se dividen en tres categorías [URL10]:

- **Descriptores de frontera:** usa los códigos de cadena para representar una frontera como un conjunto de segmentos con longitud y dirección especificada. el procedimiento que usa el código de cadena es comenzando la codificación en la esquina superior izquierda y siguiendo en el sentido de las agujas del reloj.
- **Descriptores de región:** una región se puede describir por la forma de su frontera o por sus características internas. Este tipo de descriptor es útil cuando la geometría de la imagen es fija y los objetos se señalizan siempre aproximadamente a la misma distancia de la cámara.
- **Descriptotes de textura:** es utilizada para la identificación de objetos o regiones contenidas en una imagen.

1.1.2.3. Reconocimiento, identificación y etiquetado

La etapa de reconocimiento o clasificación de imágenes se ocupa de organizar las imágenes de tal forma que el proceso de reconocimiento de objetos sea un proceso factible.

La clasificación puede ser supervisada o no supervisada, dependiendo de que sea necesaria o no la intervención de un usuario experto. En la mayoría de los casos, se necesita una etapa de aprendizaje o entrenamiento, sobre todo en los métodos supervisados. En cuanto a los no supervisados, la mayoría se basan en agrupamientos con fronteras de decisión o diferentes medidas de distancia.

Se pueden considerar distintos tipos de clasificadores [URL6]: basados en regiones, en distancias, clasificadores estadísticos o bayesianos, clasificadores borrosos, aproximaciones sintácticas (basados en la descomposición de componentes básicos), redes neuronales, etc. Cada una de estas técnicas de clasificación se analiza con más detalle en la sección 1.2 de este capítulo.

Finalmente, para el buen funcionamiento de los clasificadores, es necesario establecer técnicas definidas en cuanto al conjunto de imágenes de entrenamiento y por otro lado de evaluación, sobre todo en el caso de un conjunto finito de imágenes.

1.1.3. Nivel alto de procesamiento

En esta categoría se aplica el proceso inteligente, la técnica más representativa es la: interpretación, que trata de estudiar la lógica de los objetos localizados en la escena, intentando emular el conocimiento.

1.2. Reconocimiento de patrones

El presente trabajo versa sobre las áreas de descripción, reconocimiento, identificación y etiquetado, las cuales pertenecen al nivel intermedio de un sistema de visión artificial. El reconocimiento de patrones es una disciplina científica de amplio uso en la etapa de reconocimiento, identificación y etiquetado en un sistema de visión artificial y cuyo principal objetivo es el de clasificar objetos en un número de categorías o clases en función de sus descriptores [40].

Dependiendo de la aplicación, estos objetos pueden ser imágenes o señales en forma de onda o cualquier tipo de medición que necesita ser clasificada. A estos objetos se les denomina en forma genérica *patrones*.

Se puede decir que un *patrón* es un objeto que posee ciertas características las cuales le permiten identificarse con otros objetos que tienen propiedades similares. Los patrones que poseen ciertas similitudes entre ellos se reúnen en un conjunto denominado *clase de patrones* o simplemente *clase.* Las características propias que distinguen a los patrones en diversas clases se les llaman *descriptores*.

El reconocimiento de patrones tiene 4 enfoques: reconocimiento estadístico de patrones, reconocimiento sintáctico de patrones, redes neuronales artificiales y reconocimiento lógico combinatorio d patrones [URL3].

1.2.1. Reconocimiento estadístico de patrones

En el reconocimiento estadístico de patrones se basa en la teoría de probabilidad y estadística y supone que se tiene un conjunto de medidas numéricas con distribuciones de probabilidad conocidas y a partir de ellas se hace el reconocimiento.

En este enfoque, un patrón se representa por un vector numérico de dimensión *n*. De esta forma, un patrón es un punto en un espacio n-dimensional (de características).

El reconocimiento estadístico de patrones tiene dos modos de operación [URL2]:

- **Modo de entrenamiento o aprendizaje:** en este modo se diseña el extractor de características para representar los patrones de entrada, y se entrena al clasificador con un conjunto de datos de entrenamiento de forma que el número de patrones mal identificados se minimice.
- **Modo de reconocimiento:** aquí, el clasificador ya entrenado toma como entrada el vector de características de un patrón desconocido y lo asigna a una de las clases o categorías.

El paradigma de aprendizaje utilizado es el *aprendizaje supervisado*, éste requiere disponer de un conjunto de patrones de los cuales se conoce su *clase*. El disponer de un conjunto de entrenamiento supone que alguien se ha preocupado de etiquetar los patrones de ese conjunto. Esta tarea la suele realizar un experto en el campo en el que se va a realizar el reconocimiento. El aprendizaje supervisado, a su vez, se divide en [URL4].

- **Aprendizaje supervisado paramétrico:** en este caso, el reconocimiento de patrones puede verse, desde un punto de vista muy general, como un problema de estimación de funciones de densidad de probabilidad en un espacio multidimensional para posteriormente dividir el espacio en regiones de decisión asociadas a clases. La herramienta más utilizada en este caso es la *teoría de la decisión de Bayes*.
- **Aprendizaje supervisado no paramétrico:** este modelo es utilizado cuando no puede suponerse un modelo paramétrico para las funciones de densidad de probabilidad asociadas a las clases. En estos casos la única información disponible para la clasificación es el conjunto de parámetros o prototipos. El método más simple es el del *vecino más cercano*, que consiste en etiquetar un patrón con la etiqueta del prototipo más cercano.

1.2.1.1. Teoría de la decisión de Bayes

Esta teoría es un acercamiento estadístico fundamental para la clasificación de patrones, está basado en cuantificar las compensaciones entre varias decisiones de clasificación usando probabilidad y el costo que acompaña cada decisión. El problema de decisión es planteado en términos probabilísticos y todos los valores relevantes de la probabilidad son conocidos [15].

En otras palabras, la regla de Bayes es una técnica para calcular probabilidades condicionales, y como regla de probabilidad es indiscutible así como su validez. A partir de un conjunto de probabilidades llamadas "*a priori*", calcula un conjunto de probabilidades "*a posteriori*" que no son mas que una modificación de las primeras ante la evidencia de que un determinado suceso ha ocurrido. Esta regla de decisión también es conocida como la regla de la Máxima Probabilidad A Posteriori (MAP).

1.2.1.2. Teoría de clasificación del vecino más cercano

Esta regla del vecino más cercano establece que un objeto debe ser clasificado como perteneciente a un solo individuo de una población conocida. En este caso al individuo de la población al que pertenece su vecino más cercano. Una extensión de la regla de vecino más cercano es la regla del *k- vecino más cercano* [15].

1.2.1.3. Trabajos destacados en este enfoque

En [16] utilizan un acercamiento Bayesiano para detectar, clasificar y reconocer señalamientos de carretera. La detección de las señales se basa en la información de color que presentan las imágenes capturadas y la clasificación depende de su contenido del color. La transformada de características de escala invariante (SIFT) se emplea para extraer el conjunto de características invariantes para las etiquetas detectadas de las señales de carretera. El reconocimiento es hecho comparando las características extraídas con las características previamente almacenadas de muestras estándares.

En [29] se estudian las posibilidades de un algoritmo rápido de búsqueda del vecino más cercano, el LAESA, para obtener una disminución en la tasa de error sin coste adicional, utilizando para ello *k* vecinos. A partir de esto se ha generalizado una nueva regla de clasificación: la regla de los *k* vecinos seleccionados más cercanos. Esta regla produce tasas de acierto de clasificación similares a las de los *k* vecinos más cercanos pero con un coste computacional de la regla del vecino más cercano.

En [3] se describe un sistema basado en la visión por computadora para la detección, seguimiento, y reconocimiento robusto de señales de tráfico en tiempo real. La técnica de

acercamiento propuesta consiste de dos componentes; primero, las señales son detectadas usando un conjunto de características Wavelets Haar obtenidas por entrenamiento Ada-Boost; y segundo, se realiza la clasificación usando modelado generativo Bayesiano.

Los experimentos demuestran alta exactitud de la detección y del reconocimiento y un índice del marco de aproximadamente 10 marcos por segundo en una PC estándar.

En [30] se introduce el concepto de medidas semejantes que se pueden entrenar (*Trainable similarity measure*), para resolver el problema de la presencia de los píxeles sin información y la demanda computacional que se presentan en la clasificación de señales de carretera utilizando el algoritmo de clasificación del vecino más cercano.

La semejanza de la correlación cruzada normalizada para clasificar los prototipos, se basa en un emparejamiento individual en un conjunto de regiones locales de la imagen. El conjunto de regiones, relevante para una evaluación particular de la semejanza, es refinado por el proceso del entrenamiento.

En los experimentos con problemas de clasificación de señales de carretera, las semejanzas entrenables producen un alto rendimiento en representaciones y clasificadores de datos; exactitud de la clasificación de multi-clases, así como una capacidad de rechazamiento de las no-señales.

1.2.2. Reconocimiento sintáctico de patrones

Este enfoque se basa en encontrar las relaciones estructurales que guardan los objetos de estudio, utilizando la teoría de lenguajes formales. El objetivo es construir una gramática que describa la estructura del universo de objetos.

En el reconocimiento sintáctico, cada patrón se describe en términos de sus elementos básicos, llamados primitivas y representados por P, y unas reglas sintácticas, llamadas gramática y representadas por G; en conjunto, especifican cómo se pueden generar patrones válidos de una determinada clase. El problema se reduce entonces a responder si un determinado patrón pertenece al lenguaje generado por una gramática.

1.2.2.1. Análisis gramaticales, reconocimiento con cadenas y lenguajes formales

Lenguajes formales

Se basan en la definición de una gramática que describe las estructuras sintácticas válidas capaces de identificar. De esta forma, un objeto viene definido por una palabra del lenguaje. El reconocimiento de un objeto equivale a identificar si la palabra asociada al objeto pertenece a la gramática. La gramática del lenguaje depende del ámbito al que va dirigida la aplicación.

El algoritmo general de reconocimiento con base en lenguajes formales consta de dos pasos principales:

1. La gramática generativa crea las cadenas de símbolos terminales usando las primitivas.
2. De modo analítico, dada una secuencia y una especificación de gramática, se determina si
    a. La secuencia fue generada por la gramática.
    b. Si es así, determina la estructura de la secuencia.

Reconocimiento con cadenas

Los patrones son representados por las secuencias ordenadas o cadenas de elementos discretos como una secuencia de letras en una palabra, la clasificación de patrones basada sobre dichas cadenas de símbolos discretos, difiere en el número de caminos de las técnicas más comúnmente usadas, debido a que los elementos de las cadenas (letras, caracteres o símbolos) son nominales; no hay una noción obvia de la distancia entre cadenas. Una dificultad se presenta del hecho que las cadenas necesitan ser de longitud diferente. Dichas cadenas no son vectores. Existe un gran número de problemas referentes a cadenas en el área de computación; los únicos que son de importancia en el reconocimiento de patrones son los siguientes:

- **Emparejamiento de cadenas:** dada una cadena *x*, determinar si *x* es una subcadena de la cadena *X*, si es así establecer su posición.
- **Corrección de distancias:** dadas dos cadenas *x* y *y,* calcular el número mínimo de operaciones básicas, como inserción, borrado e intercambio de caracteres, necesarias para trasformar *x* en *y*.

Reconocimiento usando gramáticas

Es muy similar a la aproximación general usada durante todo el reconocimiento de patrones. Con este método se supone que una oración de prueba fue generada por una de las *n* gramáticas, $G_1, G_2, \cdots, G_n$, las cuales pueden ser consideradas como modelos diferentes o clases. Una oración de prueba *x* es clasificada de acuerdo a la gramática que la produjo, o equivalentemente, el lenguaje que forma esa gramática del cual *x* es miembro.

Para el reconocimiento se debe emplear el proceso inverso, dada una *x* particular encontrar una derivación en la gramática que lleve a *x*. Este proceso es llamado *parking* o *análisis sintáctico*.

1.2.2.2. Árboles de decisión

Es uno de los métodos de aprendizaje inductivo supervisado no paramétrico más utilizado. Su dominio de aplicación no está restringido a un ámbito concreto sino que pueden utilizarse en diversas áreas: diagnóstico médico, juegos, predicción meteorológica, control de calidad, etc.

Puede verse como la estructura resultante de la partición recursiva del espacio de representación a partir de un numeroso conjunto de prototipos. Esta partición recursiva se traduce en una *organización jerárquica* del espacio de representación que puede modelarse mediante una estructura de tipo árbol. Cada *nodo interior* contiene una pregunta sobre un atributo concreto (con un hijo por cada posible respuesta) y cada *nodo hoja* se refiere a una decisión (clasificación) [15], [URL4], [URL8].

Entre los clasificadores basados en árboles descritos en la literatura se encuentran ID3, C4, C4.5, árboles Bayesianos y CART (*Classification And Regression Trees*, árboles de clasificación y regresión). Las diferencias principales entre los distintos algoritmos de construcción de árboles de decisión radican en las estrategias de *poda* y en la regla adoptada para dividir nodos.

1.2.2.3. Métodos gráficos

Se utiliza la notación de grafo para describir los elementos constituyentes de un objeto y las relaciones entre ellos. El reconocimiento de un objeto con un modelo consistirá en aplicar una operación de isomorfismo entre dos grafos, el del objeto y el del modelo. Este método presenta una mayor flexibilidad en cuanto al grado de similitud que la descripción en gramáticas.

La desventaja de este método es que, forzosamente se incrementa el costo computacional, tanto en tiempo de procesamiento como en la cantidad de espacio de almacenamiento requerido y rapidez de acceso.

1.2.2.4. Trabajos destacados en este enfoque

En [23], se presenta un método para la localización automática de acontecimientos fonéticos importantes en señales de discurso continuas. Estos acontecimientos se han elegido para emparejar los requisitos de un sistema que realza estas regiones para hacer la señal más robusta para acanalar la degradación. La técnica utilizada es un acercamiento sintáctico para reconocimiento de patrones distinguido del reconocimiento automático de discurso o de acercamientos basados en el conocimiento. El funcionamiento en discurso leído de muchos transmisores es aproximadamente del 20% de errores y 20% de falsas alarmas.

En [41], se analizan fundamentos teóricos del reconocimiento sintáctico de patrones y de sus relaciones con matemática lingüística, el reconocimiento de patrones estructural, y el reconocimiento de patrones estadístico. Finalmente, observan que los tres acercamientos principales al reconocimiento de patrones (estructural, estadístico y sintáctico) se pueden combinar en un método de gran alcance. Y se propone un sistema basado en estos principios y previsto para ser utilizado para el reconocimiento de escritura a mano.

Martí en [URL11] realiza un reconocimiento estructural mediante la estructura de grafos en documentos de tipo técnico (arquitectura e ingeniería) que contienen bloques gráficos, ya que en estos es importante realizar un análisis de las líneas que usualmente interconectan los símbolos para obtener una mejor descripción y significado del diseño, con el objetivo de extraer información contextual que en algunos casos se utiliza para verificar si el documento gráfico cumple las reglas de diseño propias del ámbito técnico al que hace referencia.

En [6] se usa el reconocimiento sintáctico de patrones para la interpretación automática de formas reflejadas, las pruebas preliminares se realizan sobre formas geométricas simples. Los lenguajes usados para describir patrones ruidosos y distorsionados son a menudo ambiguos, de tal manera que, una secuencia o patrón se puede generar por más de un lenguaje, así que los patrones que pertenecen a diversas clases pueden tener la misma descripción, pero con diversas probabilidades de ocurrencia.

También se proponen diferentes aproximaciones, ejemplo, cuando un patrón ruidoso tiene dos o más descripciones estructurales es recomendable utilizar gramáticas estocásticas.

Para las pruebas en este trabajo se usa una gramática libre de contexto sobre figuras simples. También se prueba un diseño de espécimen que usa una gramática estocástica de estados finitos, la cual se asemeja a un proceso Markov de estados finitos.

1.2.3. Redes neuronales artificiales

Este enfoque supone que tiene una estructura de neuronas interconectadas que se estimulan unas a otras, las cuales pueden ser “entrenadas” para dar una cierta respuesta cuando se le presentan determinados valores.

Adquieren, almacenan y utilizan conocimiento experimental, obtenido a partir de ejemplos por ajuste de parámetros de las neuronas mediante un algoritmo de aprendizaje.

Dependiendo de la manera por la cual una red neuronal se relaciona con el ambiente, existen los siguientes paradigmas de aprendizaje [URL1], [URL9], [URL12]:

- **Aprendizaje Supervisado:** se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada. Este tipo de aprendizaje es utilizado en las redes neuronales BackPropagation, Perceptron y ADALINE.
- **Aprendizaje por Refuerzo:** es una variante de aprendizaje supervisado, que se basa en la idea de no disponer de un ejemplo completo del comportamiento deseado, es decir, de no indicar durante el entrenamiento exactamente la salida que se desea que proporcione la red ante una determinada entrada. En el aprendizaje por refuerzo la función del supervisor se reduce a indicar mediante una señal de refuerzo si la salida obtenida en la red se ajusta a la deseada (éxito = +1 o fracaso = -1), y en función de ello se ajustan los pesos basándose en un mecanismo de probabilidades. Se podría decir que en este tipo de aprendizaje la función del supervisor se asemeja más a la de un crítico (que opina sobre la respuesta de la red) que a la de un maestro (que indica a la red la respuesta concreta que debe generar).
- **Aprendizaje No Supervisado (auto-organización):** no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta. Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada. Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado. En algunos casos, la salida representa el grado de familiaridad o similitud entre la información que se le está presentando en la entrada y las informaciones que se le han mostrado hasta entonces (en el pasado). En otro caso, podría realizar un agrupamiento o establecimiento de categorías (*clustering*), indicando la red a la salida a qué categoría pertenece la información presentada a la entrada, siendo la propia red quien debe encontrar las categorías apropiadas a partir de las correlaciones entre las informaciones presentadas. Este tipo de aprendizaje es utilizado en los modelos de Mapas de Kohonen [25], [43] y redes de tipo ART (*Adaptive Resonance Theory*).

Existen muchos algoritmos de aprendizaje, entre los principales se tienen: aprendizaje por corrección de error, aprendizaje competitivo, aprendizaje asociativo y aprendizaje estocástico o de Boltzmann [URL9].

1.2.3.1. Aprendizaje por corrección de error

Algoritmo muy conocido basado en la regla Delta o regla del mínimo error cuadrado LMS, que busca minimizar la función de error usando un gradiente descendente. Este es el principio BackPropagation, muy utilizado para el entrenamiento de redes de múltiples capas. Este método de entrenamiento es de gran alcance, útil, y relativamente fácil de entender [15].

A este tipo también pertenece la regla de aprendizaje del Perceptron, utilizada en el entrenamiento de la red del mismo nombre; consiste en ajustar los pesos de las conexiones de la red en función de la diferencia entre los valores deseados y los obtenidos a la salida de la red, es decir, en función del error cometido en la salida [URL12].

1.2.3.2. Aprendizaje competitivo

En las redes con aprendizaje competitivo, suele decirse que las neuronas compiten (y cooperan) unas con otras con el fin de llevar a cabo una tarea dada. Con este tipo de aprendizaje se pretende que cuando se presente a la red cierta información de entrada, sólo una de las neuronas de salida de la red, o una por cierto grupo de neuronas, se active (alcance su valor de respuesta máximo). Por tanto las neuronas compiten para activarse quedando finalmente una, o una por grupo, como neurona vencedora; el resto quedan anuladas y siendo forzadas a sus valores de respuesta mínimos.

El objetivo de este aprendizaje es categorizar los datos que se introducen en la red, de esta forma las informaciones similares son clasificadas formando parte de la misma categoría y por tanto deben activar la misma neurona de salida. Las clases o categorías deben ser creadas por la propia red, puesto que se trata de un aprendizaje no supervisado a través de las correlaciones entre los datos de entrada. [15], [URL7].

Este tipo de aprendizaje es usado en redes ART, redes Hamming y en mapas de Kohonen con sus dos variantes denominadas LVQ (*Learning Vector Quantization*) y TPM (*Topology Preserving Map*) o SOM (*Self Organizing Map*) [25].

1.2.3.3. Aprendizaje asociativo

Esta regla de aprendizaje es la base de muchas otras, la cual pretende medir la familiaridad o extraer características de los datos de entrada.

El fundamento es una suposición bastante simple conocido como principio de Hebb: si dos neuronas N_i y N_j toman el mismo estado simultáneamente (ambas activas o ambas inactivas), el peso de la conexión entre ambas se incrementa, en el caso contrario será debilitada. Las redes asociativas se utilizan principalmente para filtrado de información en la reconstrucción de datos, eliminando distorsiones o ruido, también se emplean para explorar relaciones entre informaciones similares, para facilitar la búsqueda por contenido en bases de datos y para resolver problemas de optimización [URL1]. Se utiliza en el Modelo de Hopfield [22], [URL9].

1.2.3.4. Aprendizaje estocástico o de Boltzmann

Consiste básicamente en realizar cambios aleatorios en los valores de los pesos de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de distribuciones de probabilidad.

El objetivo del aprendizaje de Boltzmann es ajustar los pesos de conexión de tal forma que el estado de las unidades visibles satisfaga una distribución de probabilidades deseada en particular. Según lo anterior, el aprendizaje consistiría en realizar un cambio aleatorio de los valores de los pesos y determinar la energía de la red. Si la energía es menor después del cambio, es decir, si el comportamiento de la red se acerca al deseado, se acepta el cambio; si, por el contrario, la energía es mayor, se aceptaría el cambio en función de una determinada y preestablecida distribución de probabilidades [15], [URL12].

1.2.3.5. Trabajos destacados en este enfoque

El algoritmo descrito en [11] aprovecha las características de color y forma de las señales de tráfico para llevar a cabo su reconocimiento. Éste lo realiza en 2 fases: primera, para la detección, utiliza una técnica de umbral, *thresholding*, aplicada al color del objeto para dividir el análisis de la imagen en segmentos y de la forma para detectar las muestras. Segundo, para la

clasificación, utiliza una red neuronal Perceptron multicapa pudiendo manipular 3 diferentes tamaños de imágenes. El algoritmo fue implementado en una PC 486 a 33 MHz con bus local. La velocidad de detección de fase es 220 ms para una imagen de 256 X 256 bits. La red neuronal en la CPU toma 1.2 s. La puesta en práctica de la red neuronal en un procesador digital de señales (DSP) se está experimentando en la investigación, y la velocidad prevista está entre 30 y 40 ms.

En [8], para el reconocimiento de patrones en imágenes digitales realiza en un programa computacional la implementación y comparación de dos redes neuronales, una red ART2 y una Backpropagation. Las imágenes utilizadas para probar las redes son figuras geométricas, letras y rostros; y para poder extraer las características de éstas y posteriormente reconocerlas, se utilizaron los momentos invariantes y la transformada rápida de Fourier discreta.

En general, se obtienen mejores resultados con la red ART2 usando menos tiempo para las pruebas, ya que no depende del azar. Pero la información corre el riesgo de degradarse si no se tiene cuidado al introducir nuevas imágenes que estén en valores más bajos que la mínima similitud requerida entre imágenes de la misma clase; ya que la red clasificará mal las imágenes. En oposición, la red Backprogation requiere más experimentos durante su fase de aprendizaje para encontrar la solución a un problema dado, pero no corre el riesgo de perder ninguna porción de su información durante su funcionamiento.

En [2], el algoritmo propuesto para la detección de señales de tráfico parte de un análisis del color, debido a los problemas de iluminación el espacio de colores utilizado es el HSI. Se utilizan algoritmos genéticos para realizar la búsqueda de los valores de la transformación que mejor se ajustan a la señal presente en la imagen. La función que mide lo bien que el modelo concreto se ajusta a la imagen se basa en la distancia de Hausdorff, que mide la separación entre dos conjuntos de puntos. Para el reconocimiento utilizan una red neuronal de tipo ART1 ya que es capaz de almacenar el conocimiento y no necesita repetir el proceso de entrenamiento si se presentan nuevos tipos, por lo que el funcionamiento de la red le sirve a su vez de entrenamiento.

En [10], utilizan un algoritmo genético para la tarea de la detección, permitiendo una localización invariante a los cambios en la posición, la escala, la rotación, condiciones atmosféricas, la obstrucción parcial, y la presencia de otros objetos del mismo color. Para el reconocimiento de las señales se usan las redes neuronales ART1 para entradas binarias. Estas redes son capaces de desarrollar grupos estables de secuencias arbitrarias de los patrones de entrada. Tienen dos capas con trayectorias abajo-arriba y arriba-abajo entre ellas. La capa de salida esta diseñada como una red competitiva, sólo el nodo que recibe el total más grande de entrada se activa. Este nodo genera un patrón arriba-abajo que se compara con la entrada. Si la diferencia es bastante grande el nodo se inhibe y el ciclo empieza de nuevo. Cuando la similitud es alta hay un estado resonante de la red: una categoría de vector prototipo iguala el actual vector de entrada bastante cerca. Si no, una señal es generada significando que un patrón diferente se presenta y se crean nuevas neuronas. Por lo tanto, no es necesario volver a entrenar la red si nuevos patrones tienen que ser agregados. Además, la red refina y enriquece su conocimiento con el uso.

1.2.4. Reconocimiento lógico combinatorio de patrones

Este enfoque se basa en la idea de que la modelación del problema debe ser lo más cercana posible a la realidad del mismo, sin hacer suposiciones que no estén fundamentadas. Uno de los

aspectos esenciales de este enfoque es que las características utilizadas para describir a los objetos de estudio deben ser tratadas cuidadosamente.

El objetivo en este enfoque es modelar problemas donde los patrones puedan estar formados por cualquier combinación de características tanto cualitativas como cuantitativas. Los problemas de clasificación se modelan según la *Teoría clásica de conjuntos* o la *Teoría de subconjuntos difusos*. Típicamente se aplica la *Teoría de testores* para determinar las características esenciales de cada patrón y la *Teoría Combinatoria* para considerar las posibilidades de representación y análisis de los patrones. Es común en este enfoque estudiar y modificar algoritmos generados originalmente en otros enfoques para remover algunas de sus limitaciones y extender su capacidad de aplicación [18].

La clasificación con el enfoque lógico combinatorio se aplica a problemas donde el universo de objetos se agrupa en un número dado de clases, de las cuales se tiene información de cada una de ellas. El problema consiste en poder establecer las relaciones de un nuevo objeto dado con cada una de dichas clases, basada en semejanzas parciales entre los objetos.

El paradigma de aprendizaje utilizado en este enfoque es el aprendizaje supervisado. Los principales algoritmos de clasificación son: el modelo ALVOT o algoritmo de votación y el modelo KORA, basado en analogías parciales.

1.2.4.1. Modelo ALVOT

El modelo de los algoritmos de votación se describe mediante 6 etapas [39]:

- **Sistema de conjunto de apoyo:** por un conjunto de apoyo se entiende un subconjunto no vacío de rasgos en términos de los cuales se analizarán los objetos. Cada conjunto da idea de puntos de vistas o representación de un subespacio. El sistema es el conjunto de los conjuntos de apoyo.
- **Función de semejanza:** establece los criterios de comparación de los valores para cada rasgo y para cada conjunto de apoyo.
- **Evaluación por fila dado un conjunto de apoyo fijo:** proceso de conteo de votación en cuanto a la medida de semejanza entre las diferentes partes de la subdescripción de los objetos ya clasificados y el que se desea clasificar.
- **Evaluación por clase dado un conjunto fijo:** aquí se totalizan las evaluaciones obtenidas para cada uno de los objetos de la matriz de aprendizaje respecto al objeto que se quiere clasificar.
- **Evaluación por clase para todo el sistema de conjuntos de apoyo:** se totalizan los mismos cálculos hechos para un conjunto de apoyo fijo ahora para todo el sistema seleccionado.
- **Regla de solución:** se trata ahora de establecer un criterio, para que sobre la base de cada una de las votaciones obtenidas en la etapa precedente dar una respuesta en cuanto a las relaciones del objeto a clasificar con cada una de las clases del problema, tomando en consideración la evaluación por clase para todo el sistema de conjuntos de apoyo.

1.2.4.2. Modelo KORA

Este algoritmo es también un algoritmo de precedencia parcial, es decir, analiza la experiencia acumulada. Existen dos variantes del modelo, el KORA 3 que es el modelo inicial y la extensión de la variable inicial o KORA Ω.

El algoritmo KORA 3 consta de 3 etapas [39]:

- **Etapa de aprendizaje:** define los parámetros iniciales de suficiencia a aplicar en el modelo y depende en gran medida de los especialistas y del problema.
- **Etapa de re-aprendizaje:** etapa de cálculo de los parámetros y rasgos óptimos del modelo.
- **Etapa de clasificación:** dado un nuevo objeto realiza la clasificación mediante la aplicación de una regla de solución.

El modelo KORA 3 tiene algunas restricciones para su correcta aplicación, entre ellas se pueden mencionar las siguientes: sólo admite dos clases, la descripción de los objetos es en rasgos booleanos y no admite ausencia de información en la descripción de los objetos.

KORA Ω consiste en permitir las siguientes características: uso de cualquier sistema de conjunto de apoyo, rasgos de cualquier naturaleza y cualquier cantidad de clases.

1.2.4.3. Trabajos destacados en este enfoque

En [39] se presenta un sistema de clasificación supervisada, basado en 2 diferentes algoritmos para reconocimiento de patrones: el modelo ALVOT y el modelo KORA extendida haciéndose un análisis crítico y experimental de diferentes criterios de comparación. Este estudio permite analizar las ventajas y limitaciones de un algoritmo respecto al otro, así como analizar la propuesta de clasificación que brindan los mismos.

En [14] se exponen las posibilidades de las herramientas del reconocimiento lógico combinatorio de patrones para la minería de datos de grandes grupos de datos mezclados incompletos. A partir de la existencia verdadera de muchas estructuras complejas de grandes grupos de datos, trabajan en la solución del problema de clasificación supervisado y no supervisado pero con estos tipos de grupos de datos

En [44], se proponen con una matriz de representación del conocimiento, efectivos algoritmos lógico-combinatorio-probabilístico de reconocimiento de patrones en un sistema inteligente. Los 3 algoritmos se construyen en base a pruebas de diagnóstico mínimas, incondicionales, o mezclas de pruebas de diagnóstico. El reconocimiento de patrones finalmente se realiza según el principio de la votación sobre el grupo de todas las pruebas y algoritmos. Los algoritmos son recomendados para una pequeña cantidad de realizaciones de cada patrón y de un grupo grande de características. Los algoritmos fueron puestos en ejecución y probados parcialmente en el sistema IMSLOG en la solución de problemas reales.

El sistema es lo suficientemente flexible para que se le puedan incorporar nuevos algoritmos de clasificación y modelar diferentes aplicaciones que requieran de la clasificación, permitiendo definir sus rasgos, clases y especificaciones propias.

En [18] se aborda el problema de diseñar una *metodología unificada* para evaluar algoritmos de clasificación, tanto para problemas *Supervisados* como para problemas *No-Supervisados*. La metodología propuesta se basa en el enfoque de reconocimiento lógico combinatorio de patrones unificando los criterios de evaluación para ambos tipos de problemas y logra superar las debilidades de los métodos actuales. Para colaborar en la solución de este problema se diseña un lenguaje formal para especificación de problemas en Reconocimiento de patrones (*LCARS*) y se propone una arquitectura general para herramientas de software que hagan uso del lenguaje propuesto. La arquitectura propuesta se valida mediante la construcción de un prototipo de software (SHELL y STUDIO).

2. Bases teóricas, memorias asociativas y DWT

Este capítulo versa acerca de las bases teóricas de las técnicas más importantes utilizadas en este trabajo de tesis, las memorias asociativas y la DWT.

En primer término se dan los conceptos básicos de memoria asociativa, conjunto fundamental, y se describe brevemente la evolución de los modelos propuestos más importantes relacionados con este tema. Adicionalmente se comenta el proceso de análisis de la DWT, técnica utilizada en este trabajo de tesis para reducir el tamaño de la imagen con el objetivo de disminuir el requerimiento de memoria de almacenamiento en el sistema.

2.1. Memorias asociativas

Una memoria asociativa es un elemento cuyo propósito fundamental es recuperar patrones completos a partir de patrones de entrada que pueden estar alterados con ruido sustractivo, aditivo o mixto [45].

El esquema de una memoria asociativa genérica **Mg** se observa en la figura 2.1, se trata de un sistema de entrada-salida, donde los patrones de entrada y salida están representados por **x** y **y** respectivamente.

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}; \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \qquad \textbf{Ecuación 1}$$

Donde *n* y *m* son números enteros positivos y representan las dimensiones de los patrones de entrada y salida respectivamente.

Además, sean $\left\{\left(\mathbf{x}^1,\mathbf{y}^1\right),\left(\mathbf{x}^2,\mathbf{y}^2\right),...,\left(\mathbf{x}^k,\mathbf{y}^k\right)\right\}$ *k* pares de asociaciones de vectores de entrada y vectores de salida, a esta combinación se le denomina *conjunto fundamental de asociaciones* [53]. El conjunto fundamental de asociaciones queda representado de la siguiente manera:

$$\left\{\left(\mathbf{x}^\mu,\mathbf{y}^\mu\right) \mid \mu = 1,2,...,k\right\} \qquad \textbf{Ecuación 2}$$

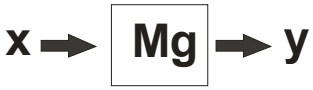

Figura 2.1. Esquema de una memoria asociativa [53].

La memoria asociativa **Mg** se representa mediante una matriz y se genera a partir del conjunto fundamental de asociaciones.

De acuerdo a su modo de operación, las memorias asociativas se clasifican en dos grupos:

- **Memorias Autoasociativas:** se dice que una memoria asociativa es de carácter autoasociativo si se cumple que $\mathbf{x}^{\mu} = \mathbf{y}^{\mu}\ \forall \mu \in \{1,2,...,k\}$ [53].
- **Memorias Heteroasociativas:** se dice que una memoria asociativa es de carácter heteroasociativo si se cumple que $\mathbf{x}^{\mu} \neq \mathbf{y}^{\mu}\ \exists \mu \in \{1,2,...,k\}$ [54].

Con base en lo anterior, se puede establecer que una memoria autoasociativa puede ser considerada como un caso particular de una memoria heteroasociativa.

El desarrollo de las memorias asociativas se ha efectuado en forma paralela a las redes neuronales, desde que McCulloch y Pitts crearon el primer modelo de una neurona artificial [28], hasta los modelos de redes neuronales basados en conceptos modernos como la morfología matemática [37].

A continuación se hace mención de los modelos de memorias asociativas más importantes, poniendo énfasis en las MAM, parte esencial de este trabajo.

2.1.1. Lernmatrix

A principios de la década de los sesentas, 1961, Karl Steinbuch desarrolló un método para codificar información en arreglos cuadriculados conocidos como *crossbar*. Este modelo fue llamado Lernmatrix [46];

El Lernmatrix es una memoria heteroasociativa que con una selección adecuada de los patrones de salida puede funcionar como un clasificador de patrones binarios. Es un sistema que al operar acepta como entrada un patrón binario $x^{\mu} \in A^n, A = \{0,1\}$ y produce como salida la clase $y^{\mu} \in A^k$ que le corresponde. Para representar la clase $t \in \{1,2,...k\}$ se asignan a los componentes del vector de salida los siguientes valores: $y_t^{\mu} = 1$, y $y_j^{\mu} = 0$ para $j = 1,2,..t-1, t+1, k$.

En la fase de aprendizaje cada uno de los componentes de la matriz **M** tiene valor cero al inicio y se actualiza de acuerdo con la regla $m_{ij} + \Delta m_{ij}$ donde ε es una constante positiva escogida previamente:

$$\Delta m_{ij} = \begin{cases} +\varepsilon \quad \text{si} \quad x_i^{\mu} = 1 = y_i^{\mu} \\ -\varepsilon \quad \text{si} \quad x_i^{\mu} = 0 \quad \text{y} \quad y_i^{\mu} = 1 \\ 0 \quad \text{en otro caso} \end{cases} \qquad \textbf{Ecuación 3}$$

La fase de recuperación consiste en encontrar la clase a la que pertenece un vector de entrada x^w dado; que significa obtener las coordenadas del vector y^w que le corresponde a este patrón sin ambigüedad. La *i-ésima* coordenada se obtiene:

$$y_i^w = \begin{cases} 1 \text{ si } \sum_{j=1}^{n} m_{ij} \cdot x_j^w = \bigvee_{h=1}^{k} \left[\sum_{j=1}^{n} m_{hj} \cdot x_j^w \right] \\ 0 \text{ en otro caso} \end{cases} \qquad \textbf{Ecuación 4}$$

donde **V** es el operador *máximo*.

2.1.2. Linear associator

El Linear Associator, modelo precursor de memorias asociativas, es producto de los trabajos publicados en 1972 por James A. Anderson con su *Interative Memory*, y por Teuvo Kohonen, con sus *Correlation Matrix Memories* [47].

La fase de aprendizaje del Linear Associator consiste de dos etapas:

1. Para el conjunto fundamental de asociaciones antes definido, se encuentra la matriz $y^\mu \cdot \left(x^\mu\right)^t$ de dimensiones *m* x *n.*
2. Se suman las matrices para obtener la memoria

$$\mathbf{M} = \sum_{\mu=1}^{k} \mathbf{y}^\mu \cdot \left(\mathbf{x}^\mu\right)^t = \left[m_{ij}\right]_{m \times n}$$
$$m_{ij} = \sum_{\mu=1}^{k} y_i^\mu x_i^\mu \qquad \textbf{Ecuación 5}$$

La fase de recuperación consiste en presentar a la memoria un patrón de entrada x^w, donde $w \in \{1,2,\ldots k\}$ y realizar la operación

$$\mathbf{M} \cdot \mathbf{x}^w = \left[\sum_{\mu=1}^{k} \mathbf{y}^\mu \cdot \left(\mathbf{x}^\mu\right)^t \right] \cdot \mathbf{x}^w = \mathbf{y}^w \qquad \textbf{Ecuación 6}$$

Para que la expresión anterior emita como resultado al patrón y^w, es preciso que los vectores de entrada sean ortonormales, en caso contrario, debido al ruido producido por la interacción de los patrones de entrada la recuperación no es perfecta, excepto si el número de patrones almacenado es pequeño comparado con las dimensiones de los vectores de entrada. El número pequeño de patrones debe estar entre $0.1n$ y $1.2n$.

2.1.3. Amari

Shun-ichi Amari publicó en 1972 un trabajo teórico titulado *Self – Organizing Nets of Thershold Elements* que constituye un importante antecedente para la creación de lo que se convertiría en el modelo de la memoria asociativa [48].

El trabajo de Amari estrablece nuevas teorías sobre el funcionamiento de las memorias asociativas de patrones, proponiendo novedosos métodos y enfoques teóricos desarrollados en su artículo de 1972, entre ellos se encuentran:

1. Una corta descripción de las redes de elementos de umbral ya conocidas (en 1972), las transiciones de estados y estados de equilibrio.
2. Definiciones de los números de estabilidad de la transición de estados y los números de estabilidad de los estados de equilibrio.
3. Condiciones de convergencia para los números de estabilidad de los estados de equilibrio.
4. Los fundamentos de la auto-organización de las redes de elementos de umbral.
5. Las redes auto-organizativas de elementos de umbral como memorias asociativas.

2.1.4. Memoria asociativa Hopfield

La memoria de Hopfield introducida por J. Hopfield en 1982 [22], [49], es auto-asociativa, por consiguiente el conjunto fundamental para la memoria Hopfield está definido como $\left\{ \left(\mathbf{x}^{\mu}, \mathbf{x}^{\mu}\right) \;\middle|\; \mu = 1, 2, \ldots, \mathrm{k} \right\}$ con $x^{\mu} \in A^{n}$ y $A = \{1, -1\}$.

En la fase de aprendizaje, cada componente m_{ij} de la memoria **M** se construye en 3 etapas:

1. Para cada par del conjunto fundamental, construir una matriz $\mathbf{x}^{\mu} \cdot \left(\mathbf{x}^{\mu}\right)^{t}$ de dimensiones *n* x *n,* la cual tiene 1's en su diagonal principal.
2. A cada una de las matrices se le resta la matriz identidad **I** de dimensiones *n* x *n,* con el fin de lograr 0's en la diagonal principal.
3. Se suman las matrices $\mathbf{x}^{\mu} \cdot \left(\mathbf{x}^{\mu}\right)^{t} - \mathbf{I}$ para finalmente obtener la matriz **M**:

$$\mathbf{M} = \sum_{\mu=1}^{k} \left[\mathbf{x}^{\mu} \cdot \left(\mathbf{x}^{\mu}\right)^{t} - \mathbf{I} \right] = \left[m_{ij} \right]_{n \times n}$$

$$m_{ij} \begin{cases} \sum_{\mu=1}^{k} x_i^{\mu} x_j^{\mu} & \text{si} \quad x \neq j \\ 0 & \text{si} \quad x = j \end{cases}$$

Ecuación 7

Si la neurona x_i en el tiempo t se representa $x_i(t)$ y en el tiempo siguiente *(t+1)* se representa como $x_i(t+1)$.

Dado un vector columna de entrada $\tilde{\mathbf{x}}$, la fase de recuperación consta de 3 pasos:

1. Para *t=0*, se hace $\mathbf{x}(t) = \tilde{\mathbf{x}}$, es decir, $x_i(0) = \tilde{x}_i, \forall i \in \{1,2,3,...n\}$
2. $\forall i \in \{1,2,3,...,n\}$ se calcula de acuerdo con la condición siguiente:

$$x_i(t+1) = \begin{cases} 1 \text{ si } \sum_{j=1}^{n} m_{ij} x_j(t) > 0 \\ x_i(t) \text{ si } \sum_{j=1}^{n} m_{ij} x_j(t) = 0 \\ -1 \text{ si } \sum_{j=1}^{n} m_{ij} x_j(t) < 0 \end{cases} \qquad \textbf{Ecuación 8}$$

3. Si $x(t+1) = x(t)$ el proceso termina y el vector recuperado es $\mathbf{x}(0) = \tilde{\mathbf{x}}$. En otro caso los pasos 2 y 3 se iteran tantas veces como sea necesario hasta llegar al punto límite localmente estable en el tiempo *t*=τ, entonces el patrón recuperado es $\mathbf{x}(\tau)$.

2.1.5. Memorias ADAM, BAM y SDM

El modelo de Memoria Asociativa Distribuida Avanzada (ADAM, *Advanced Distributed Associative Memory*) fue propuesto por Jim Austin en 1986 [50]. Este modelo tenía como objetivo mejorar las *redes asociativas* de Willshaw y consiste de dos redes asociativas a las que agregó un sistema de *códigos de clase*.

Para contrarrestar la desventaja de la autoasociatividad de la memoria Hopfiel, Bart Kosko en 1988, propuso la Memoria Asociativa Bidireccional (BAM, *Bidirectional Associative Memory*), que consiste de un arreglo de dos memorias Hopfield capaz de asociar patrones de manera heteroasociativa [51]. En la BAM se generan dos matrices diferentes **M** y su transpuesta **W** en la fase de aprendizaje, sin la restricción de que la diagonal principal en 0's. La fase de recuperación es idéntica a la fase de recuperación de la memoria Hopfiel, sólo que en este caso se requiere verificar convergencia en ambos casos para x^{μ} y y^{μ}.

En 1989 Pentti Kanerva, en su búsqueda de un modelo para la memoria humana, desarrolló un tipo de memoria asociativa a la que llamó Memoria Distribuida Escasa (SDM, *Sparse Distributed Memory*) [52]. Este modelo es una variante de la RAM usual.

2.1.6. Memorias Asociativas Morfológicas

La era moderna de las memorias asociativas nace cuando en el año de 1982 Hopfiel describe la memoria asociativa Hopfield [55]. En el año de 1998 surge un nuevo y novedoso trabajo en esta área, cuando Ritter, Sussner y Díaz de León crean las MAM. Las MAM basan su operación en las operaciones morfológicas de dilatación y erosión, es decir, hacen uso de máximos o mínimos de sumas [36]. Esta es una de las características que las distingue de las memorias Hopfield, las cuales utilizan sumas de productos.

Las MAM han demostrado ser una excelente herramienta en el reconocimiento y recuperación de patrones, sin importar que éstos contengan ruido aditivo, sustractivo o mixto [7], [35], [53].

Características que hacen atractivas a las MAM son la robustez al ruido, la capacidad de aprendizaje, la rapidez en el proceso de aprendizaje y la eficiencia y velocidad en el proceso de recuperación de patrones.

Basado en el conjunto fundamental definido en la sección 2.1, se definen las operaciones necesarias para el proceso de aprendizaje y recuperación de una MAM, estas operaciones hacen uso de los operadores máximo **∨** y mínimo **∧** [53], [54].

Sean D un vector columna de dimensión m y F un vector fila de dimensión n, el *producto máximo* $D \nabla F$ da como resultado una matriz $C = \left[c_{ij} \right]_{m \times n}$, donde $c_{ij} = \left(d_i + f_j \right)$

Generalizando para un conjunto fundamental de asociaciones:

$$C_{ij} = \bigvee_{l=1}^{k} \left(d_{il} + f_{lj} \right) \qquad \textbf{Ecuación 9}$$

De igual manera, el *producto mínimo* $C = D \Delta F$ da como resultado la matriz $C = \left[c_{ij} \right]_{m \times n}$, para un conjunto fundamental de asociaciones tenemos que:

$$C_{ij} = \bigwedge_{l=1}^{k} \left(d_{il} + f_{lj} \right) \qquad \textbf{Ecuación 10}$$

Por otra parte, sea D una matriz $D = \left[d_{ij} \right]_{m \times n}$ y F un vector columna $\left[f_i \right]_n$, el cálculo del *producto máximo* $D \nabla F$ da como resultado un vector columna C de dimensión m:

$$C_i = \bigvee_{j=1}^{n} \left(d_{ij} + f_j \right) \qquad \textbf{Ecuación 11}$$

Para el *producto mínimo* $C = D \Delta F$:

$$C_i = \bigwedge_{j=1}^{n} \left(d_{ij} + f_j \right) \qquad \textbf{Ecuación 12}$$

Estas operaciones se aplican a las MAM en sus dos modos de operación heteroasociativo y autoasociativo.

2.1.6.1. Memorias Morfológicas Heteroasociativas

Una MAM es de carácter heteroasociativo si $\exists \mu \in \{1, 2, ..., k\}$ para el que se cumple que $\mathbf{x}^{\mu} \neq \mathbf{y}^{\mu}$. Existen dos tipos de Memorias Morfológicas Heteroasociativas (MMHA): *max*, simbolizadas con **M,** y *min*, simbolizadas con **W**.

2.1.6.1.1. Memorias Morfológicas Heteroasociativas max

Las MMH *max* (**M**) son aquellas que utilizan el producto mínimo y el operador máximo en su fase de aprendizaje y el producto mínimo en su fase de recuperación.

Fase de aprendizaje:

1. Se calculan las matrices $\mathbf{y}^{\mu} \Delta \left(- \mathbf{x}^{\mu} \right)^t$ para cada uno de los k elementos del conjunto fundamental de asociaciones $\left(\mathbf{x}^{\mu}, \mathbf{y}^{\mu} \right)$.

2. Se obtiene la memoria **M** aplicando el operador máximo **V** a las matrices resultantes del paso 1. **M** queda expresada como:

$$\mathbf{M} = \bigvee_{\mu=1}^{k} \left[\mathbf{y}^{\mu} \Delta \left(-\mathbf{x}^{\mu} \right)^{t} \right] = \left[m_{ij} \right]_{m \times n}$$
$$m_{ij} = \bigvee_{\mu=1}^{k} \left(y_i^{\mu} - x_j^{\mu} \right) \qquad \textbf{Ecuación 13}$$

Fase de recuperación:

1. Se calcula el producto mínimo $\mathbf{M} \Delta \mathbf{x}^{\omega}$,donde $\omega \in \{1, 2, ..., k\}$, obteniendo un vector columna $\mathbf{y} = \left[y_i \right]_m$:

$$\mathbf{y} = \mathbf{M} \Delta \mathbf{x}^{\omega}$$
$$y_i = \bigwedge_{j=1}^{n} \left(m_{ij} + x_j^{\omega} \right) \qquad \textbf{Ecuación 14}$$

Para que una MMH *max* asegure una respuesta perfecta debe de cumplir con el teorema 1 y el corolario 1.1 de [35].

Teorema 1: $\mathbf{M} \Delta \mathbf{x}^{\omega} = \mathbf{y}^{\omega}$ para todo $\omega = 1, ..., k$ si y sólo si para cada índice fila $i = 1, ..., m$ existen índices columna $j_i^{\varepsilon} \in \{1, ..., n\}$ tales que $m_{ij_i^{\omega}} = y_i^{\omega} - x_{j_i^{\omega}}^{\omega}$ para todo $\omega = 1, ..., k$.

Corolario 1.1: $\mathbf{M} \Delta \mathbf{x}^{\omega} = \mathbf{y}^{\omega}$ para todo $\omega = 1, ..., k$ si y sólo si para cada índice fila $i = 1, ..., m$ y cada $\gamma \in \{1, ..., k\}$ existen índices de columna $j_i^{\gamma} \in \{1, ..., n\}$ tales que;

$$x_{j_i^{\gamma}}^{\gamma} = \bigwedge_{\varepsilon=1}^{k} \left(x_{j_i^{\gamma}}^{\varepsilon} - y_i^{\varepsilon} \right) + y_i^{\gamma} \qquad \textbf{Ecuación 15}$$

Además, el teorema 5 de [35] indica la cantidad de ruido que es permisible en los patrones de entrada para obtener una respuesta perfecta:

Teorema 5: Para $\gamma = 1, ..., k$**,** sea $\tilde{\mathbf{x}}^{\gamma}$ una versión distorsionada del patrón $\mathbf{x}^{\gamma}$. Entonces $\mathbf{M} \quad \tilde{\mathbf{x}}^{\gamma} = \mathbf{y}^{\gamma}$ si y sólo si se cumple que:

$$\tilde{x}_j^{\gamma} \geq x_j^{\gamma} \wedge \bigvee_{i=1}^{m} \left(\bigwedge_{\varepsilon \neq \gamma} \left[y_i^{\gamma} - y_i^{\varepsilon} + x_i^{\varepsilon} \right] \right) \forall j = 1, ..., n \qquad \textbf{Ecuación 16}$$

y para cada índice renglón $i \in \{1, ..., m\}$ existe un índice columna $j_i \in \{1, ..., n\}$ tal que:

$$\tilde{x}_{ji}^{\gamma} = x_{ji}^{\gamma} \wedge \left(\bigwedge_{\varepsilon \neq \gamma} \left[y_i^{\gamma} - y_i^{\varepsilon} + x_{ji}^{\varepsilon} \right] \right) \qquad \textbf{Ecuación 17}$$

2.1.6.1.2. Memorias Morfológicas Heteroasociativas min

Las MMH *min* (**W**) son aquellas que utilizan el producto máximo y el operador mínimo en su fase de aprendizaje y el producto máximo en su fase de recuperación.

Fase de aprendizaje:

1. Se calculan las matrices $\mathbf{y}^{\mu}\Delta\left(-\mathbf{x}^{\mu}\right)^{t}$ para cada uno de los k elementos del conjunto fundamental de asociaciones $\left(\mathbf{x}^{\mu},\mathbf{y}^{\mu}\right)$.

2. Se obtiene la memoria **W** aplicando el operador mínimo **Λ** a las matrices resultantes del paso 1. **W** queda expresada como:

$$\mathbf{W}=\bigwedge_{\mu=1}^{k}\left[\mathbf{y}^{\mu}\nabla\left(-\mathbf{x}^{\mu}\right)^{t}\right]=\left[w_{ij}\right]_{m\times n}$$
$$w_{ij}=\bigwedge_{\mu=1}^{k}\left(y_{i}^{\mu}-x_{j}^{\mu}\right) \qquad \textbf{Ecuación 18}$$

Fase de recuperación:

1. Se calcula el producto máximo $\mathbf{W}\nabla x^{\omega}$,donde $\omega\in\{1,2,...,k\}$, obteniendo un vector columna $\mathbf{y}=\left[y_{i}\right]_{m}$:

$$\mathrm{y}=\underset{n}{\mathbf{W}}\nabla\mathbf{x}^{\omega}$$
$$y_{i}=\bigvee_{j=1}\left(w_{ij}+x_{j}^{\omega}\right) \qquad \textbf{Ecuación 19}$$

El teorema 2 y el corolario 2.1 de [35] rigen las condiciones que una MMH *min* debe cumplir para obtener una respuesta perfecta.

Teorema 2: $\mathbf{W}\nabla\mathbf{x}^{\omega}=\mathbf{y}^{\omega}$ para todo $\omega=1,...,k$ si y sólo si para cada índice fila $i=1,...,m$ existen índices columna $j_{i}^{\omega}\in\{1,...,n\}$ tales que $w_{ij_{i}^{\omega}}=y_{i}^{\omega}-x_{j_{i}^{\varepsilon}}^{\omega}$ para todo $\omega=1,...,k$.

Corolario 2.1: $\mathbf{W}\nabla\mathbf{x}^{\omega}=\mathbf{y}^{\omega}$ para todo $\omega=1,...,k$ si y sólo si para cada índice fila $i=1,...,m$ y cada $\gamma\in\{1,...,k\}$ existen índices de columna $j_{i}^{\gamma}\in\{1,...,n\}$ tales que

$$x_{j_{i}^{\gamma}}^{\gamma}=\bigvee_{\varepsilon=1}^{k}\left(x_{j_{i}^{\gamma}}^{\varepsilon}-y_{i}^{\varepsilon}\right)+y_{i}^{\gamma} \qquad \textbf{Ecuación 20}$$

Por otra parte, el teorema 6 de [35] indica la cantidad de ruido que es permisible en los patrones de entrada para obtener una respuesta perfecta:

Teorema 6: Para $\gamma=1,...,k$**,** sea $\tilde{\mathbf{x}}^{\gamma}$ una versión distorsionada del patrón $\mathbf{x}^{\gamma}$. Entonces $\mathbf{W}\quad\tilde{\mathbf{x}}^{\gamma}=\mathbf{y}^{\gamma}$ si y sólo si se cumple que:

$$\tilde{x}_{j}^{\gamma}\geq x_{j}^{\gamma}\vee\bigwedge_{i=1}^{m}\left(\bigvee_{\varepsilon\neq\gamma}\left[y_{i}^{\gamma}-y_{i}^{\varepsilon}+x_{i}^{\varepsilon}\right]\right)\forall j=1,...,n \qquad \textbf{Ecuación 21}$$

y para cada índice renglón $i \in \{1,...,m\}$ existe un índice columna $j_i \in \{1,...,n\}$ tal que:

$$\tilde{x}_{ji}^{\gamma} = x_{ji}^{\gamma} \vee \left(\bigvee_{\varepsilon \neq \gamma} \left[y_i^{\gamma} - y_i^{\varepsilon} + x_{ji}^{\varepsilon} \right] \right) \qquad \textbf{Ecuación 22}$$

2.1.6.2. Memorias Morfológicas Autoasociativas

Una MAM es de carácter autoasociativo si se cumple que $\mathbf{x}^{\mu} = \mathbf{y}^{\mu}\ \forall \mu \in \{1,2,...,k\}$. Existen dos tipos de Memorias Morfológicas Autoasociativas (MMAA): *max*, simbolizadas con **M,** y *min*, simbolizadas con **W**.

El conjunto fundamental de asociaciones queda definido como: $\left(\mathbf{x}^{\mu}, \mathbf{x}^{\mu}\right)$

2.1.6.2.1. Memorias Morfológicas Autoasociativas max

Las MAMH *max* (**M**) son aquellas que utilizan el producto mínimo y el operador máximo en su fase de aprendizaje y el producto mínimo en su fase de recuperación.

Fase de aprendizaje:

1. Se calculan las matrices $\mathbf{y}^{\mu} \Delta \left(-\mathbf{x}^{\mu}\right)^{t}$ para cada uno de los *k* elementos del conjunto fundamental de asociaciones $\left(\mathbf{x}^{\mu}, \mathbf{x}^{\mu}\right)$.
2. Se obtiene la memoria **M** aplicando el operador máximo **V** a las matrices resultantes del paso 1. **M** queda expresada como:

$$\mathbf{M} = \bigvee_{\mu=1}^{k} \left[\mathbf{x}^{\mu} \Delta \left(-\mathbf{x}^{\mu}\right)^{t} \right] = \left[m_{ij} \right]_{m \times n}$$

$$m_{ij} = \bigvee_{\mu=1}^{k} \left(w_i^{\mu} - x_j^{\mu} \right) \qquad \textbf{Ecuación 23}$$

Fase de recuperación:

1. Se calcula el producto mínimo $\mathbf{M} \Delta \mathbf{x}^{\omega}$, donde $\omega \in \{1,2,...,k\}$, obteniendo un vector columna $\mathbf{x} = \left[x_i \right]_n$:

$$\mathbf{x} = \mathbf{M} \Delta \mathbf{x}^{\omega}$$

$$x_i = \bigwedge_{j=1}^{n} \left(m_{ij} + x_j^{\omega} \right) \qquad \textbf{Ecuación 24}$$

Los Teoremas 1, 2 y 3 y el Corolario 1.1 de [35] rigen las condiciones de convergencia, aprendizaje y recuperación perfecta de patrones de una MAM *max*.

2.1.6.2.2. Memorias Morfológicas Autoasociativas min

Las MMH *min* (**W**) son aquellas que utilizan el producto máximo y el operador mínimo en su fase de aprendizaje y el producto máximo en su fase de recuperación.

Fase de aprendizaje:

1. Se calculan las matrices $\mathbf{y}^{\mu}\Delta\left(-\mathbf{x}^{\mu}\right)^{t}$ para cada uno de los *k* elementos del conjunto fundamental de asociaciones $\left(\mathbf{x}^{\mu},\mathbf{x}^{\mu}\right)$.

2. Se obtiene la memoria **W** aplicando el operador mínimo Λ a las matrices resultantes del paso 1. **W** queda expresada como:

$$\mathbf{W}=\bigwedge_{\mu=1}^{k}\left[\mathbf{x}^{\mu}\nabla\left(-\mathbf{x}^{\mu}\right)^{t}\right]=\left[w_{ij}\right]_{m\times n}$$

$$w_{ij}=\bigwedge_{\mu=1}^{k}\left(x_{i}^{\mu}-x_{j}^{\mu}\right)$$

Ecuación 25

Fase de recuperación:

1. Se calcula el producto máximo $\mathbf{W}\nabla\mathbf{x}^{\omega}$,donde $\omega\in\{1,2,...,k\}$, obteniendo un vector columna $\mathbf{x}=\left[x_{i}\right]_{n}$:

$$\mathbf{x}=\mathbf{W}\nabla\mathbf{x}^{\omega}$$

$$x_{i}=\bigvee_{j=1}^{n}\left(w_{ij}+x_{j}^{\omega}\right)$$

Ecuación 26

Los Teoremas 1, 2 y 3 y el Corolario 1.1 de [35] rigen las condiciones de convergencia, aprendizaje y recuperación perfecta de patrones de una MAM *min*.

2.2. Transformada Discreta Wavelet

La teoría de Wavelets involucra los conceptos de *wavelets*, Transformada Continua Wavelet y DWT. Esta teoría es una herramienta matemática reciente basada en la teoría de conjuntos y representaciones cuadradas integrables, la cual permite representar una señal tanto en escala, espacio y posibles direcciones.

La teoría de Wavelets trabaja de manera similar que la teoría de Fourier, la cual dice que una señal se puede descomponer en una serie de funciones sinusoidales con la finalidad de facilitar su análisis.

Las wavelets son funciones que se encuentran en el espacio y se emplean para analizar a una señal de interés con la finalidad de obtener sus características de espacio, tamaño y dirección; la familia está definida por:

$$f_{a,b}(x)=\frac{f\left(\dfrac{x-b}{a}\right)}{\sqrt{|a|}};\quad a,b\in\Re,\ a\neq 0$$

Ecuación 27

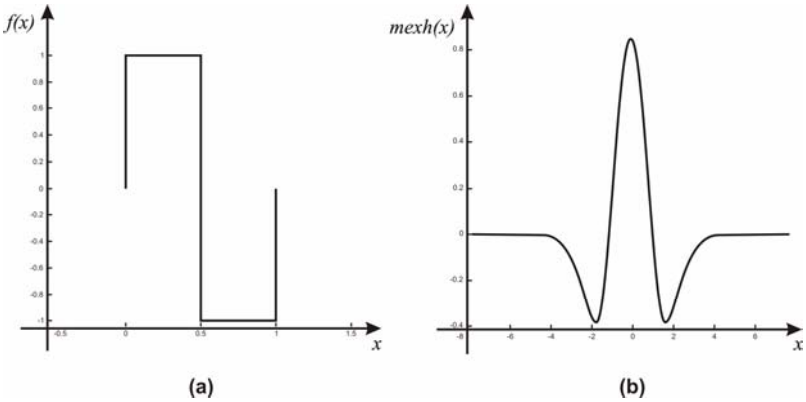

Figura 2.2. (a) Wavelet Haar; (b) Wavelet Mexican hat.

Las cuales son generadas a partir de funciones madre *f(x)*. A esta función madre se le agregan un par de variables, la escala *(a)*, que permite hacer dilataciones y contracciones de la señal, y la variable de traslación *(b)*, que permite mover a la señal en el tiempo. Estas variables son números reales y obviamente para una escala de 0 la función se indetermina. Las wavelets se usan en el procesamiento de señales.

Existen diferentes wavelets que ya son ampliamente utilizadas y que tienen definiciones establecidas. Sin embargo, la elección de un tipo de wavelet depende de la aplicación específica en la que se vaya a emplear.

Un concepto importante para el análisis de una wavelet es el *momento de desvanecimiento.* El *i*-ésimo momento de desvanecimiento de una wavelet se calcula con la siguiente integral:

$$\int_{-\infty}^{\infty} f(x)x^i dx = 0 \qquad \textbf{Ecuación 28}$$

De esta expresión se determina que una función tiene *v* momentos de desvanecimientos si la integral es cero para $i = 0,...,v-1$.

Dentro de las wavelets más utilizadas se puede citar a la Haar, la cual será utilizada en el algoritmo propuesto en el presente trabajo, esta wavelet es la más simple y la más antigua, se describe con la siguiente función [17][27]:

$$f(x) = \begin{cases} 1; & 0 \le x < 1/2, \\ -1; & 1/2 \le x < 1, \\ 0; & \text{otro valor.} \end{cases} \qquad \textbf{Ecuación 29}$$

La gráfica de esta wavelet se muestra en la figura 2.2 (a), se puede observar su sencillez. Es una wavelet muy utilizada para el análisis de señales usando transformadas continuas y discretas. Esta wavelet tiene un solo momento de desvanecimiento.

El nombre de la wavelet *Mexican hat,* muy utilizada en el análisis de señales, proviene de la forma que describe su grafica que está definida por [17][27]:

$$mexh(x) = \frac{2\left(1-x^2\right)e^{-\frac{x^2}{2}}}{\pi^{1/4}\sqrt{3}} \qquad \textbf{Ecuación 30}$$

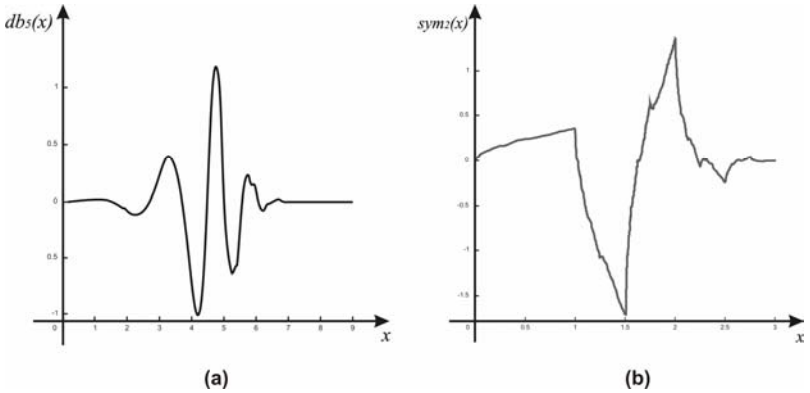

Figura 2.3. (a) Wavelet Daubechies de orden 5; (b) Wavelet Symmlet de orden 2.

Esta wavelet es la segunda derivada de la función de densidad de probabilidad gausiana. La figura 2.2 (b) muestra que la wavelet es simétrica, característica que le permite examinar a las señales de un modo simétrico y lineal en la fase, igual que la wavelet Haar.

Otro ejemplo de wavelet es la Daubechies, la cual dependiendo del número de momentos de desvanecimiento que se deseen puede tener un orden N, donde N, es un número entero y positivo y denota el número de coeficientes del filtro que tiene esta wavelet [17], [27]. La wavelet Daubechies de orden 1 es la wavelet Haar.

En la figura 2.3 (a) se puede observar la wavelet Daubechies de orden 5, donde el número de momentos de desvanecimiento es igual al orden del filtro que tiene esta wavelet. Para este caso, los coeficientes del filtro pasa baja de descomposición son: $\{C_{db5}\}$={0.0033, -0.0126, -0.0062, 0.0776, -0.0322,-0.2423, 0.1384, 0.7243, 0.6036, 0.1661}.

Esta wavelet cuenta con las características de ortogonalidad y biortogonalidad, además de que se pueden realizar transformadas wavelet continuas y discretas con ella.

La wavelet Symmlet [27] con las mismas características antes mencionadas de la wavelet Daubechies, posee la cualidad de obtener diferentes órdenes, la figura 2.3 (b) muestra la wavelet Symmlet de orden 2 cuyos coeficientes de filtro pasa baja de descomposición son: $\{C_{sym2}\}$={-0.1294, 0.2241, 0.8365, 0.4830}. Como se puede apreciar, esta wavelet es asimétrica y el número de momentos de desvanecimiento es directamente proporcional al orden de la wavelet.

La wavelet Coiflet [27] cuya gráfica puede ser apreciada en la figura 2.4 (a), posee un mayor número de momentos de desvanecimiento, ya que para cada orden diferente de la wavelet se tienen 2N momentos de desvanecimiento. Puede ser simétrica o asimétrica dependiendo del orden N de la wavelet. Los coeficientes de filtro pasa baja de descomposición de la wavelet Coiflet son: $\{C_{coif1}\}$={-0.0157, -0.0727, 0.3849, 0.8526, 0.3379, -0.0727}.

Otra de estas funciones finitas es la wavelet Gaussiana, que se define como la derivada de la función de densidad de probabilidad Gaussiana, expresada matemáticamente como sigue [27]:

$$gaus(x,n) = Cn\frac{d}{dx}\left(e^{-x^2}, n\right) \qquad \textbf{Ecuación 31}$$

donde *Cn* es una constante determinada por 2 – *norm* de *gaus*(x,n) = 1. Esta wavelet puede ser simétrica o asimétrica según el valor de *n* y con ella sólo es posible realizar la Transformada Continua Wavelet. La figura 2.4 (b) muestra la wavelet Gaussiana de orden 1.

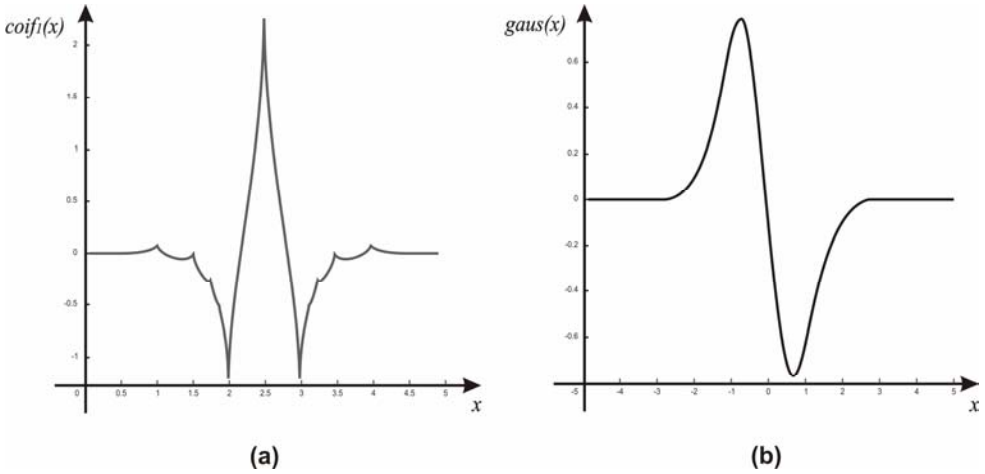

Figura 2.4. (a) Wavelet Coiflet; (b) Wavelet Gaussiana de orden 1.

La siguiente expresión matemática define la wavelet Morlet [17][27]:

$$morl(x) = e^{-\frac{x^2}{2}} \cos(5x) \qquad \textbf{Ecuación 32}$$

En la figura 2.5 se puede apreciar que se trata de una wavelet simétrica. Por otro lado esta wavelet no posee características de ortogonalidad ni biortogonalidad, además de que sólo es útil para realizar la Trasformada Continua Wavelet.

Las Transformadas de Wavelets comprenden la Transformada Continua Wavelet y la DWT. Se trata de herramientas matemáticas que permiten el análisis de señales proporcionando información de éstas en el dominio del tiempo y en el dominio de la frecuencia.

Este trabajo aplica la Transformada Wavelet a imágenes fijas, por tanto, en función de la naturaleza de la señal, es de particular y único interés la DWT, por tanto sólo se realizará la descripción de ésta y se omitirá la de la transformada Continua Wavelet.

La DWT es una descomposición de una señal en frecuencias que se aplica a toda la imagen.

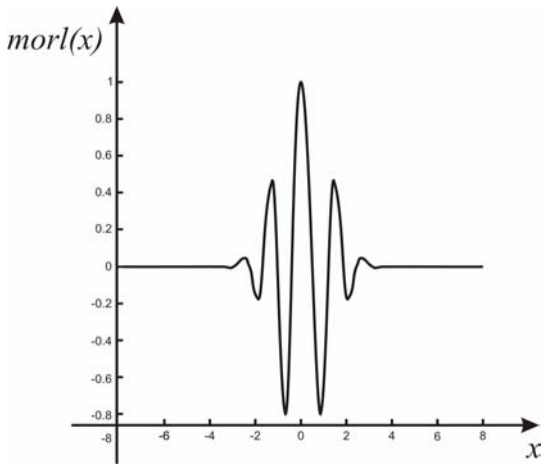

Figura 2.5. Wavelet Morlet.

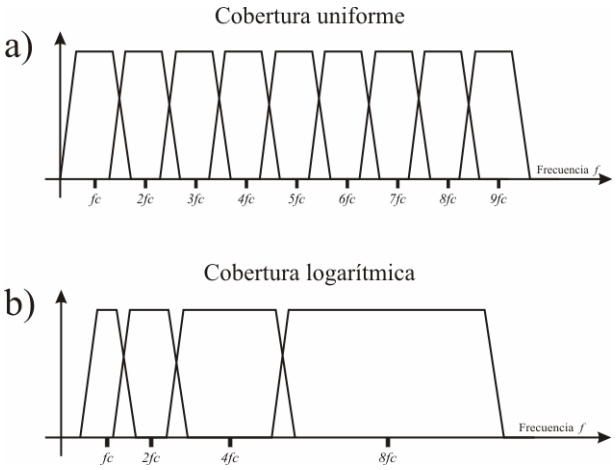

Figura 2.6. División de la señal en el dominio de la frecuencia, (a) cobertura uniforme, (b) cobertura logarítmica.

El uso de la DWT en el tratamiento de señales es adecuado debido a varios factores:

- Se consigue una alta compactación de energía ya que la mayoría de las imágenes tiene la mayor parte de la energía concentrada en frecuencias bajas.
- Al ser una transformada que se aplica a toda la imagen, no presenta la distorsión que suma el dividir a una imagen en bloques para su procesamiento, como lo hace la DCT.
- Al tener los Wavelet localización espacial y de frecuencia, hace viable el tratamiento de cambios bruscos en una imagen.

Existen dos esquemas generales que implementan la DWT, denominadas DWT de cobertura uniforme y DWT de cobertura logarítmica [32], [38], la división de la señal en el dominio de la frecuencia para ambos casos se puede observar en la figura 2.6.

Debido a la forma en que una imagen se ve afectada por cada uno de estos esquemas, el de cobertura uniforme es el idóneo para la aplicación, por tanto, se analizará con detalle este esquema.

Los trabajos más destacados, en la aplicación de la DWT sobre análisis de imágenes, empiezan en el año de año de 1992, cuando Ronald A. DeVore y sus colaboradores desarrollaron una teoría matemática que permite emplear la Transformada Wavelet en esta área [1].

En el proceso de análisis de los Wavelets, las señales son representadas utilizando un grupo de funciones básicas producidas por el desplazamiento y escalamiento de una función madre o función principal [9].

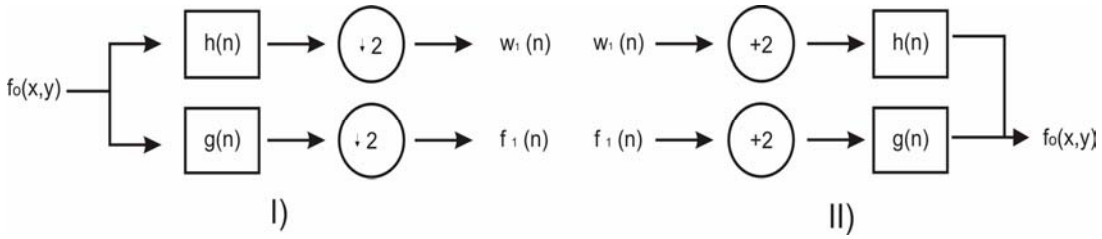

Figura 2.7. (I) Un banco de filtro de análisis de la DWT, (II) Un banco de filtro de síntesis de la DWT inversa.

En cada iteración, la DWT unidimensional descompone a una señal de entrada, $f_o(n)$, en una parte de promedio $f_i(n)$ (L), y otra de detalle $W_i(n)$ (H), donde *i* representa el nivel de aplicación de la transformada.

La aproximación de la señal $f_i(n)$, y el detalle $W_i(n)$, en el nivel $i+1$ están dadas por las siguientes ecuaciones:

$$f_{i+1}(n) = \sum_k g(k) f_i(2n-k) \qquad \textbf{Ecuación 33}$$

$$W_{i+1}(n) = \sum_k h(k) f_i(2n-k) \qquad \textbf{Ecuación 34}$$

Estas ecuaciones describen el cálculo de la DWT. La figura 2.7 muestra el diagrama bloques del cálculo de la DWT y de la DWT inversa (IDWT) de 1 nivel.

Debido a que una imagen es una señal de dos dimensiones, $f_0(x,y)$, para el tratamiento de éstas se hace uso de la DWT bidimensional, obteniendo como resultado una función de escalamiento o parte promedio $f_i(x,y)$ (LL) y 3 Wavelets bidimensionales o partes detalle $W_i^H(x,y)$ (HL), $W_i^V(x,y)$ (LH) y $W_i^D(x,y)$ (HH).

Estos Wavelet miden las variaciones de intensidad o nivel de gris en una imagen, de esta forma, W^H mide las variaciones a lo largo de las columnas (esquinas horizontales), W^V a lo largo de los renglones (esquinas verticales) y W^D variaciones diagonales.

La DWT bidimensional descompone una imagen de acuerdo con las siguientes expresiones [9]:

$$f_{i+1}(x,y) = \sum_{k_1}\sum_{k_2} g(k_1)g(k_2) f_i(2x-k_1, 2y-k_2) \qquad \textbf{Ecuación 35}$$

$$W_{i+1}^H(x,y) = \sum_{k_1}\sum_{k_2} h(k_1)g(k_2) f_i(2x-k_1, 2y-k_2) \qquad \textbf{Ecuación 36}$$

$$W_{i+1}^V(x,y) = \sum_{k_1}\sum_{k_2} g(k_1)h(k_2) f_i(2x-k_1, 2y-k_2) \qquad \textbf{Ecuación 37}$$

$$W_{i+1}^D(x,y) = \sum_{k_1}\sum_{k_2} h(k_1)h(k_2) f_i(2x-k_1, 2y-k_2) \qquad \textbf{Ecuación 38}$$

Donde $h(k)$ y $g(k)$ son filtros Wavelet de una dimensión. La imagen $f_{i+1}(x,y)$ es un suavizado de baja resolución de la imagen $f_i(x,y)$. Este suavizado se calcula mediante un filtro pasa bajo y diezmado de filas y columnas. Las imágenes $W_{i+1}^H(x,y)$, $W_{i+1}^V(x,y)$ y $W_{i+1}^D(x,y)$ contienen el detalle de $f_i(x,y)$.

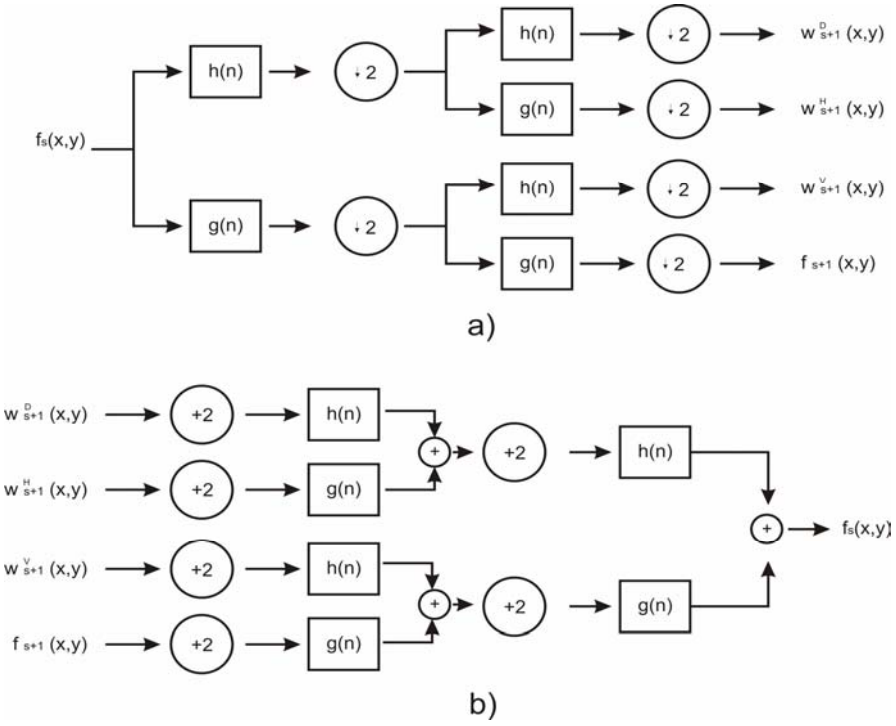

Figura 2.8. (a) Esquema para el cálculo de la DWT bidimensional; (b) Esquema para el cálculo de la IDWT bidimensional.

La DWT bidimensional puede ser implementada usando filtros digitales de 1 dimensión; aplicando la DWT de una dimensión, primero sobre las columnas y luego sobre los renglones (o viceversa). La figura 2.8 (a) muestra el esquema para la implementación de la DWT bidimensional de 1 nivel, este bloque puede se iterado para producir una transformada de "*P*" niveles. La figura 2.8 (b) muestra el esquema para el proceso de síntesis de la IDWT bidimensional.

En la figura 2.9, se ejemplifica la aplicación de la DWT de nivel 1, 2 y 3 a la imagen Lena.

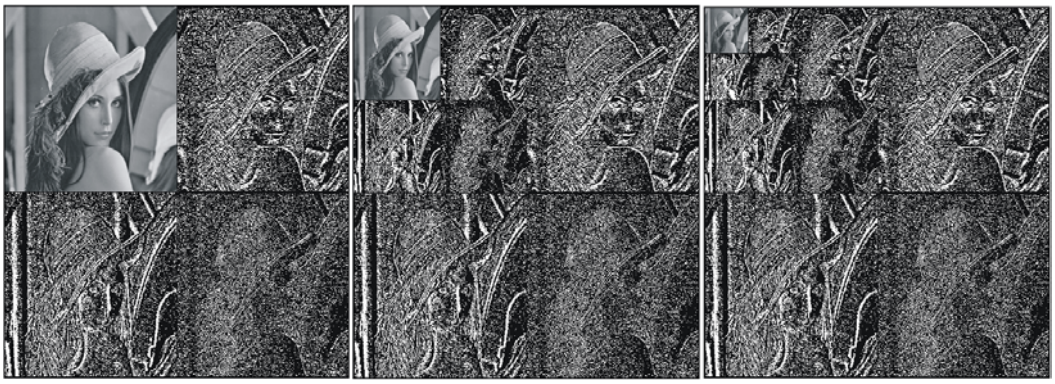

Figura 2.9. DWT aplicada a imagen Lena, 1, 2 y 3 niveles.

3. Implementación del sistema de reconocimiento de imágenes

En este capítulo se detalla la implementación de un sistema de reconocimiento de imágenes para ser utilizado por sistemas autónomos, la cual incluye la descripción de una arquitectura VLSI para el reconocimiento de imágenes con base en la DWT y las MAM.

Para tal fin, se propone el modelado de un procesador de aplicación específica en un circuito digital configurable utilizando un lenguaje descriptor de hardware. Al diseñar un procesador de aplicación específica se propone seguir la metodología de diseño de un sistema empotrado para la elaboración del sistema final.

3.1. Diseño del sistema

Un sistema empotrado también llamado embebido, es una combinación de hardware, software y, eventualmente, componentes mecánicos diseñados para realizar una función determinada [20].

La figura I.1 muestra el ciclo de vida del diseño de un sistema empotrado, este proceso se lleva a cabo con la interacción y optimización de 7 etapas que se describen a continuación [4]:

- **Etapa 1. Especificación del producto**: en esta fase se define de forma clara y precisa lo que hará el sistema, se determinan las entradas y salidas reales de éste y su interfaz con el operador. Consiste en una descripción de los requerimientos técnicos y funcionales del sistema que se va a implementar para asegurar un producto robusto.
- **Etapa 2. Selección del procesador:** en esta fase se elige el procesador que representa la mejor opción para obtener el producto especificado, para esta decisión se toman en cuenta los siguientes puntos, pines requeridos para entradas y salidas, interfaces requeridas, consideraciones en tiempo real, ambiente de desarrollo, velocidad de procesamiento, etc.
- **Etapa 3. Definición HW y SW:** durante esta etapa se debe definir el hardware que se va a utilizar y sus características principales. Las especificaciones de software incluyen una declaración de requisitos basada en la definición del hardware, las especificaciones de ingeniería e incluso en la definición general de requisitos del sistema.
- **Etapa 4. Sistema de evaluación:** en esta fase los componentes hardware y software son simulados o probados con la ayuda de una herramienta especializada, emulador, para conocer con bastante anticipación el comportamiento del sistema que se está

implementando, pudiendo de esta forma anticipar y si fuera necesario corregir su funcionalidad.

- **Etapa 5. Diseño paralelo HW y SW:** en esta fase se diseñan detalladamente todos los componentes HW y SW especificados en los requerimientos, poniendo especial atención en mantener la coherencia entre los componentes en el producto final.
- **Etapa 6. Integración de componentes HW y SW:** en esta fase primordialmente se deben integrar los componentes desarrollados para verificar que interaccionan entre sí de manera correcta. Para ello se debe contar con herramientas y métodos especiales para el manejo de la complejidad del sistema.
- **Etapa 7. Prueba y verificación del producto:** en esta fase se realizan las pruebas del producto, y si es preciso se realizan modificaciones y/o correcciones para adaptarlo a las necesidades del usuario y posteriormente liberar el producto.

3.1.1. Especificación del producto

El producto es un sistema de reconocimiento de patrones y/o imágenes digitales autónomo, el cual forma parte de un proceso de visión artificial para la navegación de robots móviles. Las especificaciones iniciales del sistema de reconocimiento de imágenes son:

- Debe ser un sistema autónomo, por tanto tendrá como parte central un procesador.
- Debe incluir una interfaz con el usuario a través de una computadora personal (PC), la interfaz debe cumplir los siguientes requisitos:
  - Ser una interfaz de alta velocidad.
  - Emular el sistema de adquisición de la imagen.
  - Permitir monitorear el funcionamiento local y/o global del sistema.
- Capacidad suficiente para reconocer 5 imágenes de dimensión 720x480 píxeles en escala de grises. Y una fácil ampliación de la capacidad de imágenes que pueden ser reconocidas.
- Optimizar los requerimientos de memoria del sistema.
- Ser un sistema de alta velocidad de procesamiento.
- Ser robusto en el reconocimiento de imágenes alteradas por ruido.
- Las imágenes usadas como patrones en las fases de aprendizaje y reconocimiento se muestran de manera simbólica en la figura 3.1. y de tamaño normal en el apéndice B.
- Para el reconocimiento de las imágenes, se hará uso de un algoritmo con base las MAM que han demostrado ser una excelente herramienta en el reconocimiento y recuperación de patrones.
- Con base en las características de las MAM, el reconocimiento de las imágenes se hará en dos fases: una fase de aprendizaje y una de reconocimiento.

Figura 3.1. Conjunto de imágenes patrón: siga, alto, flecha a la derecha, retorno, flecha ala izquierda.

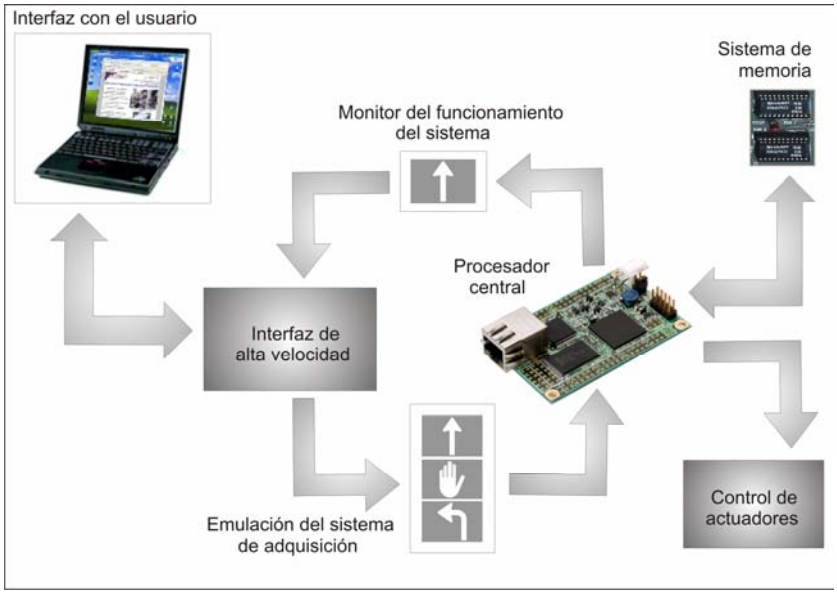

Figura 3.2. Esquema general del sistema para el reconocimiento autónomo de imágenes.

- El sistema contará con actuadores que permitan visualizar la tarea que un robot móvil realizaría al ser reconocida cada una de las imágenes. En caso de no reconocerse la imagen, no se realiza tarea alguna.

Considerando las especificaciones iniciales del sistema, se puede organizar el esquema general mostrado en la figura 3.2, cuyos principales componentes son: el procesador central, sistema de memoria, control de actuadores, interfaz de alta velocidad con una PC y una interfaz de usuario.

3.1.2. Selección del procesador

Un FPGA es un circuito digital configurable que permite el modelado de sistemas que pueden realizar tareas en forma paralela y a altas frecuencias, repercutiendo directamente en la velocidad de procesamiento, además se trata de una arquitectura flexible que permite modelar cualquier sistema digital mediante un lenguaje descriptor de hardware con un alto nivel de abstracción. Debido a estas razones, en lugar de elegir un procesador de arquitectura fija, se propone el modelado de un procesador de aplicación específica. Con esto se obtendrán las siguientes ventajas:

- Modelado de un procesador de arquitectura especializada que cumple tareas de reconocimiento de imágenes.
- Permite fácilmente hacer modificaciones a la arquitectura del procesador.
- Poder ser programado en diversas arquitecturas de FPGA's.

Para implementar la arquitectura VLSI para el reconocimiento de imágenes, se ha elegido el circuito digital configurable FPGA XCV300 de la familia Virtex de la compañía Xilinx. Este dispositivo cuenta con las siguientes características:

- Arquitectura flexible que equilibra velocidad y densidad.
- Lógica de acarreo dedicada para aritmética de alta velocidad.
- Densidad de 300,000 compuertas.
- Tecnología SRAM, no tiene límites en el número de veces que puede ser reconfigurado.

- Frecuencia interna de operación de hasta 200Mhz.
- Cumplen con las especificaciones de la interfaz PCI 66 MHz.

3.1.2.1. Herramientas EDA utilizadas

Las herramientas para la Automatización del Diseño Electrónico (EDA, *Electronic Design Automation*) están formadas por el conjunto de herramientas, tanto hardware como software, que son empleadas en el diseño de sistemas electrónicos. Dentro de EDA, las Herramientas de Diseño Asistido por Computadora (CAD, *Computer Aided Design*) juegan un importante papel en el diseño de hardware a través de software.

Las herramientas EDA-CAD automatizan el proceso de desarrollo, repercutiendo en una disminución en el tiempo de diseño, aumentando la calidad del producto y reduciendo los costos de producción [31].

En la actualidad existen variados ambientes de desarrollo para automatizar un diseño basado en un circuito digital configurable, ejemplos de estos son: ISE Foundation de la compañía Xilinx, Quartus II de la compañía Altera, Libero de la compañía Actel, ispLEVER de la compañía Lattice, entre otros. Estas herramientas están compuestas por programas que permiten al usuario modelar un sistema digital, simularlo, sintetizarlo, implementarlo y su configuración física en un dispositivo. Estas herramientas tienen disponibles diversos métodos para modelar un sistema digital, los más comunes son: diagrama de esquemático, editor de máquinas de estado y lenguaje descriptor de hardware.

Existen algunos lenguajes descriptores de hardware estandarizados, como el VHDL y el Verilog, que permiten describir diseños portables entre herramientas de diversas compañías.

Por otro lado, las herramientas EDA hardware facilitan el diseño e implementación de un prototipo. La creciente demanda por la tecnología configurable ha permitido la existencia en el mercado de una gran cantidad de este tipo de herramientas.

Para el desarrollo de este sistema, se han elegido el conjunto de herramientas EDA formado por las herramientas software ISE (*Integrated Software Environment*) Foundation versión 6.3i de la compañía Xilinx, y el lenguaje descriptor de hardware VHDL para el modelado del sistema; y la tarjeta de desarrollo XSV300 Board V1.0 de la compaña Xess como herramienta hardware.

La herramienta CAD ISE Foundation versión 6.3i, figura 3.3, es un entorno integrado de desarrollo (IDE, *Integrated Development Environment*) que permite transitar fácilmente por las diferentes fases del proceso de modelado de un sistema digital, desde el diseño hasta la implementación sobre una arquitectura configurable.

ISE Foundation se encarga de llamar de manera automática a las diferentes utilidades y herramientas disponibles dependiendo de la etapa de diseño actual. Las tareas automatizadas en el diseño de un sistema digital son las siguientes:

- **Captura del diseño:** permite al usuario modelar un sistema mediante uno o varios de los siguientes métodos: diagrama de esquemático, lenguaje descriptor de hardware, editor de maquinas de estado y/o mediante un archivo de conexiones (*Netlist*). En el caso de la captura de esquemático, ISE Foundations cuenta con bibliotecas muy extensas de componentes de uso común, por lo que sólo basta con realizar interconexiones entre estos componentes para describir un sistema digital. Para un modelado usando un lenguaje descriptor de hardware, ISE Foundation integra los dos estándares más populares VHDL y Verilog.

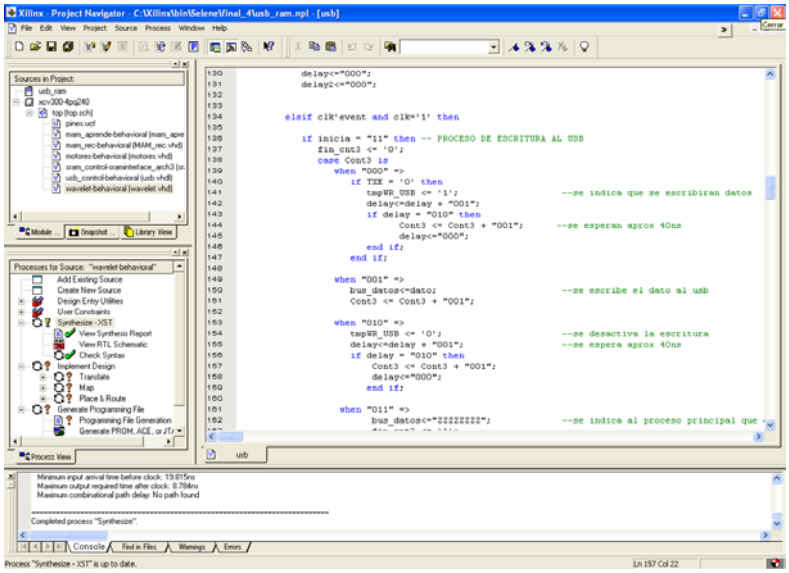

Figura 3.3. Herramienta CAD, ISE Foundation version 6.3i de Xilinx.

- **Simulación comportamental:** una vez terminado el modelado del sistema digital y antes de sintetizarlo, ISE Foundation permite una simulación denominada comportamental. Esta primera simulación es realizada para verificar el correcto funcionamiento del diseño a un nivel de transferencia de registros, esta simulación no considera retardos en las señales producidas por la implementación del diseño en el dispositivo.
- **Síntesis Lógica:** ISE Foundations incluye la herramienta propia para realizar la síntesis lógica de un código en HDL o de un diagrama esquemático. A través de una compilación es posible producir un *netlist* a partir de cualquiera de los métodos de captura. Un *netlist* es un archivo que registra la descripción de las celdas lógicas (primitivas) y las conexiones entre ellas para conformar un diseño. El formato de *netlist* más utilizado es el Formato de Intercambio de Diseño Electrónico (EDIF, *Electronic Design Interchange Format,*).
- **Implementación:** es el proceso siguiente a la síntesis y comprende las siguientes fases:
  - Mapeo o Planeación de la Superficie (Floorplanning): el archivo netlist generado por la síntesis es mapeado sobre el FPGA seleccionado. La configuración de las celdas lógicas y sus respectivas conexiones, descritas por el netlist, se distribuyen sobre la superficie del circuito integrado a manera de identificar recursos. Los algoritmos involucrados en este proceso permiten minimizar espacio físico y retardos de señal, en una primera aproximación.
  - Colocación (Placement): estratégicamente, el software decide la colocación de las primitivas sobre un bloque del dispositivo físico (módulo lógico). Los algoritmos inteligentes con los que trabaja el sistema de colocación, deciden la mejor ubicación para cada celda lógica, considerando las redundancias en el diseño (componentes o conexiones repetidas) y una segunda aproximación para eliminar los retardos críticos.
  - Trazado o Ruteo (Rutting): realiza la conexión física entre los módulos lógicos y determina el tipo de interconexión proponiendo rutas cortas o largas, eligiendo la mejor opción.

– Traducción (Translate): es la extracción de características del circuito, determinando la resistencia y capacitancia eléctrica entre las interconexiones, para verificar un correcto ruteo de las líneas. El software de diseño se encarga automáticamente de generar el ruteo y la extracción de impedancias.

- **Simulación temporal:** una vez que se han colocado y ruteado las celdas primitivas y los módulos que las contienen, la configuración física que ha adquirido el dispositivo puede simularse. A este tipo de simulación se le conoce como *física*, debido a su proximidad con el comportamiento real que tendrá el diseño. Es posible considerar los retardos de propagación que sufren las señales. El retardo generado dentro de un módulo lógico será igual a la suma de los retardos de los elementos contenidos por el mismo (flip flops, tablas de búsqueda, multiplexores, y demás). Para determinar la *ruta crítica*, el analizador del software suma los retardos de cada módulo lógico más los retardos de las interconexiones que participan en la ruta, desde un puerto de entrada hasta un puerto de salida. Las bibliotecas de cada dispositivo en particular, contienen información referente a los retardos de cada elemento dentro del módulo lógico, por lo que los resultados entregados por la simulación son certeros y confiables.
- **Programación del dispositivo:** consiste básicamente en un conjunto de módulos que permiten configurar al dispositivo elegido, estos módulos son: el modulo generador de un archivo con la información necesaría para configurar al dispositivo y el módulo que permite la interfaz con la herramienta hardware necesaria para la configuración del dispositivo.

La tarjeta de desarrollo XSV300 Board V1.0 de la compaña Xess, figura 3.4, integra gran cantidad de recursos con la finalidad de facilitar el diseño basado en un FPGA XCV300 de la familia Virtex de la compañía Xilinx. En el apéndice A se encuentra una descripción detallada de esta tarjeta.

Entre los recursos más relevantes de la tarjeta XSV300 se incluyen los siguientes:

- Circuitos digitales configurables: FPGA XCV300 Virtex, CPLD XC95108.
- Memoria flash RAM de 16 Mbits.
- Dos bancos de SRAM de 512kx16.
- Decodificador de vídeo, acepta formatos NTSC/PAL/SECAM.

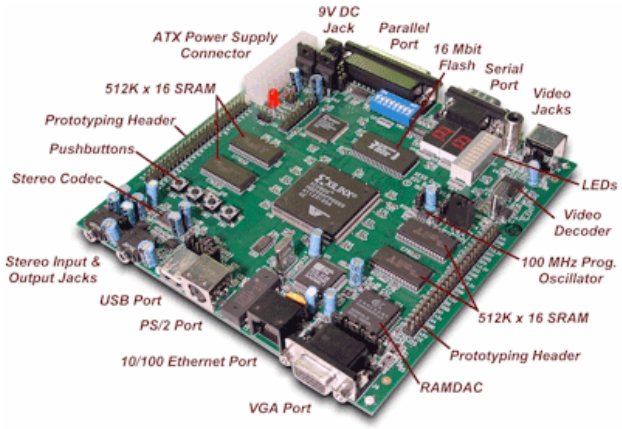

Figura 3.4. Tarjeta de desarrollo XSV300 de la compañía Xess.

- CODEC stereo usado para digitalizar o generar señales de audio de 0 a 50 KHz.
- Controlador Ethernet 10 Base-T/100 Base-Tx.
- Controlador USB.
- Interfaz por puerto serie y paralelo con una PC.

3.1.3. Definición de los componentes HW y SW

La figura I.2 muestra el esquema general resultante de analizar las especificaciones del sistema, este esquema auxilia en la definición del los componentes hardware y las especificaciones de los componentes software que integran el sistema.

- Componentes Hardware.
  - Interfaz de alta velocidad.
  - Sistema de memoria.
  - Actuadores.
- Componentes Software
  - Procesador de aplicación especifica para el reconocimiento de imágenes.
  - Interfaz con el usuario.

3.1.3.1. Definición de los componentes HW

Considerando las especificaciones iniciales del sistema, se ha seleccionado una interfaz USB para la transferencia de alta velocidad entre el procesador y la PC, se ha determinado que el sistema de memoria estará formado por dos bancos de 512kx8 bits para un total de 512kx16 bits, cumpliendo con el requisito de poder reconocer 5 imágenes y una fácil ampliación de la capacidad de imágenes que pueden ser reconocidas, teniendo un máximo aproximado de 500 imágenes; por último el control de los actuadores es una simple interfaz de potencia utilizado puentes H y el control de un display de 7 segmentos como indicador de la imagen reconocida.

3.1.3.1.1. Interfaz de alta velocidad

Para obtener una transferencia de información de alta velocidad entre el procesador para el reconocimiento de imágenes y la PC se ha elegido el controlador USB DLP-USB245M, el cual cuenta con las siguientes características:

- Compatible con USB 1.1 y USB 2.0
- Velocidad de transferencia máxima: 8 Mbits por segundo. Buffers FIFO para la transmisión y recepción de datos.
- Controlador D2XX (directo a USB + interfaz a través de DLL).
- Virtual Com Port (VCP), controladores para Windows, MAC y Linux
- Interfaz simple con PC o MCU.
- El protocolo es manejado automáticamente dentro del módulo.

La descripción detallada de los pines de este dispositivo, se puede observar en la tabla 3.1.

La figura 3.5 muestra el diagrama de tiempos que rige una transferencia de datos del procesador hacia la PC a través del controlador USB DLP-USB245M; la tabla 3.2 indica los tiempos que deben cumplir las señales involucradas.

Tabla 3.1. Tabla de descripción de pines del DLP-USB245M.

Pin#	Descripción
1	**BOARD ID** (Salida) Identifica la tarjeta como un DLP-USB245M en bajo, o un DLP-USB232M en alto.
2, 5, 7	**GROUND**
3	**RESET#** (Entrada) Puede ser usado por un dispositivo externo para resetear el FT245BM. Si no se requiere este pin deberá ser conectado a VCC.
4	**RESETO#** (Salida) Salida del generador de reset interno. Se encuentra en alta impedancia por 2ms después de que VCC sea mayor a 3.5 V y el reloj interno haya iniciado, entonces se activa el regulador interno de 3.3 V. RESET# en bajo también forzará a RSTOUT# a la alta impedancia. RSTOUT# no afecta para un reset del Bus USB.
6	**3V3OUT** (Salida) Salida del regulador integrado, provee de un suministro interno de 3.3V para el transceiver del USB y el pin RSTOUT# .
8	**SLEEP** (Salida) Se pone en bajo después de que el dispositivo es configurado vía USB, entonces durante el tiempo en alto el USB se suspende. Puede ser usado para controlar la potencia mediante lógica externa usando un interruptor MOSFET canal P.
9	**SND/WUP** (Entrada) Si el DLP-USB245M está suspendido, un pulso positivo en este pin inicializa una secuencia remota para despertar. Si el dispositivo está activo un pulso positivo en este pin carga los datos en el buffer de escritura para ser enviados a la PC en el próximo dato de entrada del USB sin tomar en cuanta cuantos bytes están en el buffer.
10	**VCC-IO** (Entrada) 3.0 volts a +5.25 volts VCC.
11	**EXTVCC** (Entrada) Usado principalmente para aplicar alimentación (4.4 to 5.25 Volts) a el módulo. Debe conectarse a PORTVCC si el modulo va a ser alimentado por el Puerto USB.
12	**PORTVCC** (Salida) Alimentación para el Puerto USB. Se conecta a EXTVCC si el modulo está conectado al Puerto USB (típica configuración). 500mA de corriente máxima en el caso de usar un adaptador de USB.
13	**RXF#** (Salida) Cuando está en bajo, almenos un byte está presente en el buffer de recepción FIFO de 128 bytes y esta listo para ser leído con RD#. RXF# está en alto cuando el buffer de recepción está ocupado.
14	**TXE#** (Salida) Cuando está en alto, el buffer FIFO de 385 bytes está lleno u ocupado guardando el último byte escrito. No intente escribir datos al buffer de transmisión cuando TXE# está en alto.
15	**WR** (Entrada) Cuando pasa de un estado lógico alto a un bajo, WR lee las líneas de 8 datos y escribe el byte en el buffer de transmisión FIFO. Para la escritura de los datos en el buffer de transmisión se envía a la computadora dentro del buffer TX una interrupción (default 16ms) y se pone en el RS-232 el buffer abierto para el programa de aplicación.
16	**RD#** (Entrada) Cuando se encuentra en bajo, RD# toma las líneas de 8 datos de un estado de alta impedancia para recibir el byte actual en el buffer FIFO. Tomando RD# en alto regresa los pines a un estado de alta impedancia y prepara el siguiente byte para ser leído.
17… 24	**D7… DO** I/O Bus de datos bidireccional

Tabla 3.2. Tiempos para el ciclo de escritura del DLP-USB245M

Tiempo	Descripción	Min	Max	Unidad
T7	Ancho de pulso activo para WR	50		ns
T8	Tiempo entre deshabilitación y habilitación de WR	50		ns
T9	Tiempo de establecimiento de datos antes de WR inactivo		20	ns
T10	Tiempo de mantenimiento de datos con WR inactivo	10		ns
T11	WR inactivo para TXE#	5	25	ns
T12	TXE inactivo después del ciclo de RD	80		ns

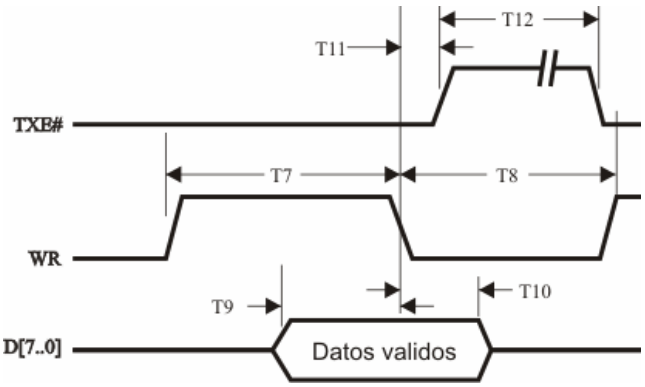

Figura 3.5. Diagrama de tiempos para el ciclo de escritura del DLP-USB245M.

Para que el procesador pueda realizar una transferencia, la señal TXE# debe estar en un nivel lógico bajo; los datos presentes en el bus D[7..0] se almacenarán en el buffer de transmisión del DLP-USB245M en el flanco de caída de la señal WR. Si el buffer de transmisión, con capacidad de 384 Bytes, está lleno o realizando una operación de escritura el dispositivo pondrá TXE# en un nivel lógico alto, no permitiendo la escritura de datos hasta que algunos de los datos de FIFO sean transferidos por el USB hacia la PC.

La transferencia de información de la PC hacia el procesador se rige con el diagrama de tiempos de la figura 3.6, los tiempos de estas señales se pueden consultar en la tabla 3.3. Una vez que el controlador DLP-USB245M recibe datos desde la PC, éste pone la señal RXF# en un nivel lógico bajo indicándole al procesador que tiene datos disponibles para él; el procesador lee el buffer de recepción del DLP-USB245M usando la señal RD# y el bus D[7..0], tal como lo indica la figura 3.6.

Tabla 3.3. Tiempos para el ciclo de lectura del DLP-USB245M

Tiempo	Descripción	Min	Max	Unidad
T1	Ancho de pulso activo para RD	50		ns
T2	Tiempo entre deshabilitación y habilitación de RD	50		ns
T3	Datos válidos para RD activo		30	ns
T4	Tiempo de mantenimiento de datos validos para RD inactivo	10		ns
T5	RD inactivo para RXF#	5	25	ns
T6	RXF inactivo después del ciclo de RD	80		Ns

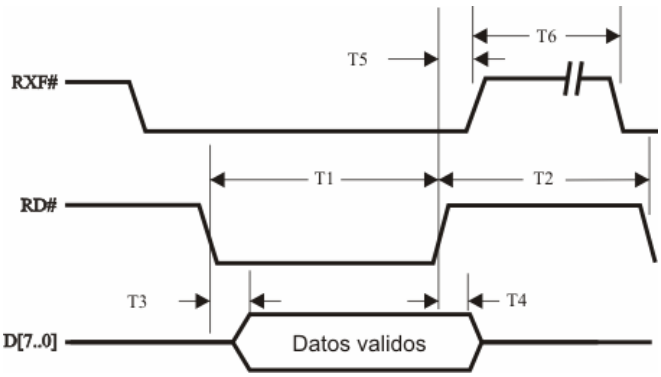

Figura 3.6. Diagrama de tiempos para el ciclo de lectura del DLP-USB245M.

Tabla 3.4. Tabla de descripción de pines de la SRAM AS7C4096-15.

Pin#	Descripción
9, 27	**VCC**
10, 28	**GROUND**
6	**/CE** (Entrada) Señal que habilita la SRAM
13	**/WE** (Entrada) Señal que activa la escritura ala SRAM
31	**/OE** (Entrada) Señal que activa la lectura a la SRAM
7, 8	**D0, D1** (I/O) Bits 0 y 1 del buffer bidirecional de datos
11, 12	**D2, D3** (I/O) Bits 2 y 3 del buffer bidirecional de datos
25, 26	**D4, D5** (I/O) Bits 4 y 5 del buffer bidirecional de datos
29, 30	**D6, D7** (I/O) Bits 6 y 7 del buffer bidirecional de datos
1… 4	**A0… A4** (Entrada) Bits del 0 al 4 del bus de direcciones
14… 18	**A5… A9** (Entrada) Bits del 5 al 9 del bus de direcciones
20… 24	**A10… A14** (Entrada) Bits del 10 al 14 del bus de direcciones
32… 35	**A15… A18** (Entrada) Bits del 15 al 18 del bus de direcciones

3.1.3.1.2. Sistema de memoria

El sistema de memoria tiene una capacidad de 512kx16 bits, pudiendo almacenar la información de aproximadamente 500 posibles imágenes a reconocer. Este sistema está formado por dos memorias SRAM (*Static Random Accsess Memory*, Memoria Estática de Acceso Aleatorio) AS7C4096-15 conectadas para conseguir una expansión del bus de datos, las características principales de estas memorias son las siguientes:

- Voltaje de alimentación 5V.
- Organización de 512kx8 bits.
- Tiempos de acceso:
  - 15 ns, con respecto al bus de direcciones.
  - 7 ns, con respecto a la señal OE (*output enable*, habilitación de la salida).
- Bajo consumo de potencia:
  - Estado activo: 1375 mW max.
  - Modo de bajo consumo: 55 mW max.

La descripción detallada de los pines de este dispositivo, se puede observar en la tabla 3.4.

Tabla 3.5. Tiempos para el ciclo de escritura de la SRAM AS7C4096.

Parámetro	Sim	Min	Max	Unidad
Tiempo del ciclo de escritura	t_{WC}	15	-	ns
Activación de /CE para la escritura	t_{CW}	12	-	ns
Establecer la dirección para finalizar la escritura	t_{AW}	12	-	ns
Tiempo para establecer la dirección	t_{AS}	0	-	ns
Ancho de pulso de la escritura	t_{WP}	12	-	ns
Mantener la dirección para finalizar la escritura	t_{AH}	0	-	ns
Datos validos al finalizar la escritura	t_{DW}	7	-	ns
Tiempo de mantenimiento de datos	t_{DH}	0	-	ns
/WE para la salida en alta impedancia	t_{WZ}	-	6	ns

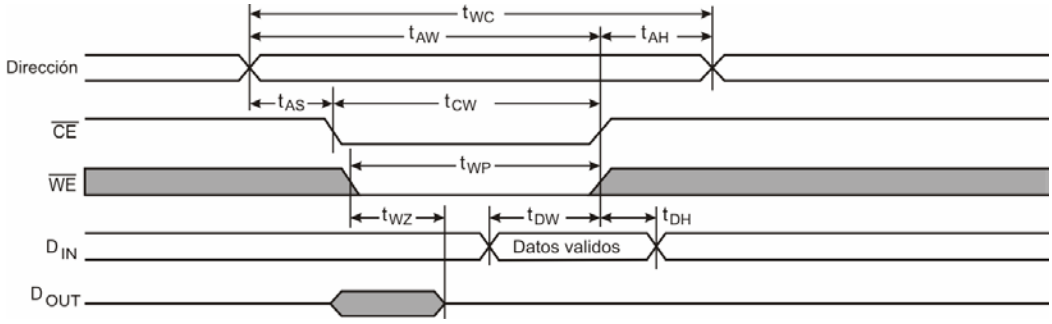

Figura 3.7. Diagrama de tiempos para el ciclo de escritura de la SRAM AS7C4096.

El ciclo de escritura de una memoria AS7C4096-15 involucra al bus de datos, al de direcciones y a las señales de control WR y CS, tal como se puede apreciar en la figura 3.7; los tiempos que deben de cumplir estas señales están disponibles en la tabla 3.5.

La localidad a ser manipulada es indicada por el bus de direcciones, un nivel lógico bajo en la señal CS activa o selecciona a la memoria, por último los datos presentes en el bus de datos de la memoria son escritos en el flanco de subida de la señal WE o de la señal CE.

En la figura 3.8 y la tabla 3.6 se puede observar el diagrama de tiempos y la duración de las señales en un ciclo de lectura de la memoria AS7C4096-15. En el ciclo de lectura de esta memoria, un nivel lógico bajo en la señal CS activa o selecciona a la memoria y un nivel lógico bajo en la señal RD permite que la información presente en el bus de datos sea almacenada en la localidad indicada por el bus de direcciones.

Tabla 3.6. Tiempos para el ciclo de lectura de la SRAM AS7C4096.

Parámetro	Sim	Min	Max	Unidad
Tiempo del ciclo de lectura	t_{RC}	15	-	ns
Tiempo de acceso a la dirección	t_{AA}	-	15	ns
Tiempo de acceso al /CE	t_{ACE}	-	15	ns
Tiempo de acceso al /OE	t_{OE}	-	5	ns
Mantener la salida para el cambio de dirección	t_{OH}	3	-	ns
/CE en bajo para la salida en baja impedancia	t_{CLZ}	0	-	ns
/CE en alto para la salida en alta impedancia	t_{CHZ}	-	4	ns
/OE en bajo para la salida en baja impedancia	t_{OLZ}	0	-	ns
/OE en alto para la salida en alta impedancia	t_{OHZ}	-	4	ns
Tiempo de estado activo	t_{PU}	0	-	ns
Tiempo de bajo consumo	t_{PD}	-	15	ns

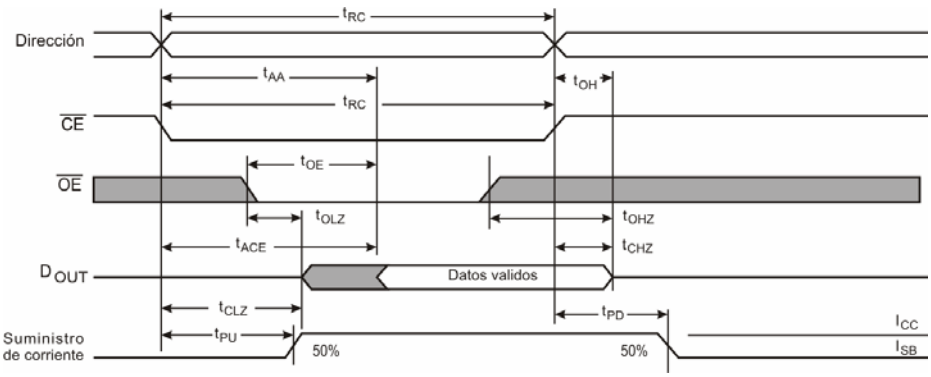

Figura 3.8. Diagrama de tiempos para el ciclo de lectura de la SRAM AS7C4096.

Tabla 3.7. Tabla de verdad para el funcionamiento de un motor de DC controlado por un puente H.

C1	C0	dirección
0	0	Alto
0	1	Avance
1	0	Retroceso
1	1	Inválido

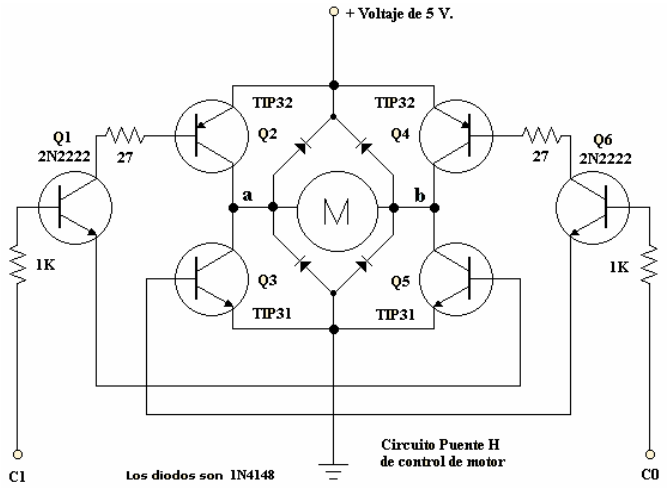

Figura 3.9. Puente H de transistores.

3.1.3.1.3. Actuadores

El reconocimiento de una imagen del conjunto fundamental de asociaciones, se verá reflejado en el control de dos actuadores (motores de corriente directa, CD) por parte del procesador para el reconocimiento de imágenes y en un dsipaly de 7 segmentos.

La potencia necesaria para mover a los motores y los elementos para el cambio de dirección de los mismos, es suministrada por un puente H a base de transistores.

Un puente H es un conjunto de 4 transistores, utilizados como interruptores, que sirven para controlar el sentido de giro de un motor CD. De las posibles combinaciones de los transistores, algunas deben ser evitadas ya que pueden producir un corto circuito. El sentido de giro de los motores CD depende de la polaridad del voltaje de alimentación. En la figura 3.9 se muestra el diagrama del puente H, a base de transistores TIP's 31, TIP's 32 y 2N2222, utilizado en este trabajo; la tabla 3.7 muestra las posibles combinaciones de los transistores de este puente H y los efectos en el funcionamiento del motor

Otra forma, disponible en el sistema, en la que el procesador simboliza la imagen reconocida es mediante la representación de la etiqueta correspondiente en un display de 7 segmentos, figura 3.10. La tabla 3.8 muestra la codificación asignada a cada una de las imágenes.

Tabla 3.8. Codificación de las imágenes del conjunto fundamental.

Número	a	b	c	d	e	f	G	Imagen representada
0	1	1	1	1	1	1	0	-no reconocida-
1	0	1	1	0	0	0	0	1
2	1	1	0	1	1	0	1	2
3	1	1	1	1	0	0	1	3
4	0	1	1	0	0	1	1	4
5	1	0	1	1	0	1	1	5

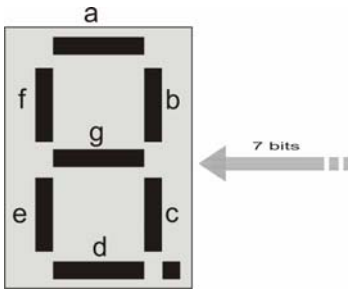

Figura 3.10. Display de 7 segmentos.

3.1.3.2. Especificación de los componentes SW

Los componentes software son, por una parte, la aplicación implementada en el procesador encargado de controlar al sistema y, por otra, la interfaz con el usuario para evaluar el comportamiento de cada una de las etapas del sistema. En la elección del procesador se optó por el modelado de un procesador de aplicación específica en un FPGA, para la descripción del mismo se elije el lenguaje descriptor de hardware VHDL. El componente de software que complementa al sistema, es la interfaz con el usuario, la cual será implementada con el lenguaje de alto nivel C++ Builder 6.0 de la compañía Borland.

3.1.3.2.1. Procesador para el reconocimiento de imágenes

Para cumplir con las especificaciones funcionales del sistema y poder llevar a cabo el control de los módulos hardware del sistema y siguiendo los principios de la metodología elegida (sección 3.1.5.1), el modelado del algoritmo de reconocimiento de imágenes y el de los módulos adicionales que intervienen en el funcionamiento del sistema completo, es dividido en los siguientes módulos, mismos que pueden ser apreciados en la figura I.3.

- **Módulo central de proceso:** este módulo se encarga de gobernar el funcionamiento y la interacción de todos los módulos que forman al procesador de aplicación específica.
- **Módulo de interfaz USB:** la función de este módulo es implementar el protocolo de comunicaciones con el controlador USB, DLP-USB245M, con la finalidad de proveer una interfaz de alta velocidad de transferencia de información entre la PC y el procesador de aplicación específica. Mediante este módulo el procesador de aplicación específica recibirá las imágenes o patrones para su posterior procesamiento y podrá enviar los resultados de dicho procesamiento.
- **Módulo de control del sistema de memoria:** implementa el protocolo de interfaz para que cualquier módulo del procesador para el reconocimiento de imágenes pueda acceder al sistema de memoria.
- **Módulo de la DWT:** implementa el algoritmo de la Transformada Discreta Wavelet bidimensional sobre la imagen o patrón a procesar, logrando de esta manera reducir el tamaño de la imagen y por consiguiente la disminución de los requerimientos de memoria de almacenamiento.
- **Módulo de binarización:** implementa el modelado de la binarización de la función de escalamiento o parte promedio obtenida de la DWT bidimensional.
- **Módulo de aprendizaje:** implementa el algoritmo de aprendizaje de una Memoria Asociativa Morfológica, haciendo uso del conjunto fundamental de asociaciones (vectores de salida, imágenes, y vectores de entrada, etiquetas). La MAM obtenida de este proceso es almacenada en el sistema de memoria.

- **Módulo de recuperación:** implementa el algoritmo de reconocimiento de la imagen o recuperación de un vector de entrada, haciendo uso de la MAM almacenada en el sistema de memoria, la etiqueta recuperada es utilizada por el módulo central de proceso con el objetivo de efectuar las acciones apropiadas.
- **Módulo de control de actuadores:** este módulo, mediante señales de control, modificará el comportamiento de los actuadores en respuesta a la etiqueta obtenida por el módulo de recuperación. También desplegará el número correspondiente a la imagen que ha sido reconocida en un display de 7 segmentos.

3.1.3.2.2. Interfaz con el usuario

Esta parte del sistema se encarga de proporcionar el conjunto de imágenes que formarán parte del conjunto fundamental de asociaciones utilizado por el procesador para el reconocimiento de imágenes. Además fungirá como monitor del procesamiento que lleva a cabo cada uno de los módulos del procesador sobre las imágenes.

Este software para la interfaz con el usuario es implementado en una PC y compilado utilizando la herramienta Borland C++ Builder en su versión 6.0, debido a que el controlador de USB, el DLP-USB245M cuenta con archivos controladores para Windows que son fáciles de acceder mediante esta herramienta.

Las opciones que presentará el programa de interfaz, son las siguientes:

- Abrir o inicializar el uso del puerto USB.
- Transmisión de las imágenes/patrón para su aprendizaje.
- Visualización simbólica de los patrones enviados.
- Selección, envío y visualización de la imagen real que se desea reconocer.
- Adquisición y visualización de la imagen reducida y binarizada, una vez aplicados los procesos de DWT y de binarización en el procesador para el reconocimiento de imágenes.
- Despliegue de la etiqueta que corresponde a la imagen que ha sido reconocida utilizando los algoritmos de las Memorias Asociativas Morfológicas *max* y las Memorias Asociativas Morfológicas *min*.

3.1.4. Sistema de evaluación

En esta fase los componentes hardware y software son evaluados mediante el uso de herramientas especializadas, como emuladores, con la finalidad de conocer o predecir su comportamiento dentro del sistema final. La realización de esta fase es cumplida, en forma particular para cada uno de los módulos que integran el sistema final, dentro de la etapa 5, Diseño Paralelo de Hardware y Software, del ciclo de desarrollo de un sistema empotrado (sección 3.1.5).

3.1.5. Diseño paralelo HW y SW

Una vez definidas las tareas que serán realizadas por hardware y software y sus respectivas especificaciones, se procede a realizar un diseño detallado y en paralelo de cada uno de los componentes del sistema con la finalidad de asegurar congruencia entre ellos para su correcto funcionamiento en el sistema final. A continuación se describe el diseño detallado de cada uno de los módulos que conforman el sistema para el reconocimiento de imágenes. Teniendo en

cuenta que un módulo puede contener elementos de software y hardware, estos elementos deben ser desarrollados en paralelo para evaluar su funcionamiento en forma conjunta.

3.1.5.1. Procesador para el reconocimiento de imágenes

Un diseño mediante un HDL inicia con el planteamiento inicial del tipo de modelado a utilizar y la metodología de diseño a seguir. La elección de estos puntos iniciales de diseño depende directamente de la complejidad de la aplicación y del nivel de abstracción que se desee manejar.

Existen dos formas de modelar un sistema digital utilizando un HDL, mediante su estructura o mediante su comportamiento. En el modelado de un sistema digital mediante su estructura, se especifican los componentes que integran al diseño y las interconexiones entre cada uno de ellos, esto implica un nivel bajo de abstracción. En el modelado mediante su comportamiento, no existe la necesidad de conocer la estructura interna del sistema, es posible describirlo explicando su funcionalidad mediante un algoritmo, connotando un nivel alto de abstracción [34].

Independientemente del tipo de modelado elegido para la descripción del sistema, comportamental o estructural, es necesario definir la metodología que se seguirá para jerarquizar los diferentes niveles de concepción con los que se trabajara.

Existen dos metodologías principales utilizadas para modelar un sistema digital usando un HDL [31], [33]:

- **Metodología Ascendente (Bottom-Up):** esta metodología empieza con la descripción de los elementos básicos del sistema, conocidos como primitivas, para posteriormente agruparlos con la finalidad de formar módulos de aplicación especifica, estos a su vez se agrupan para formar nuevos módulos hasta llegar a uno solo que representa el sistema completo. Esta metodología empieza con un nivel bajo de abstracción dirigiéndose progresivamente a un nivel alto de abstracción. En general esta metodología no es del todo recomendable para diseños de complejidad media o alta, partiendo del hecho que tras unir cientos o miles de primitivas, no es factible que el diseño funcione correctamente.
- **Metodología Descendente (Top-Down):** esta metodología parte de una idea en un nivel alto de abstracción e inicia la descripción en forma descendente, incrementando el nivel de detalle según sea necesario. El diseño inicial se divide en diferentes módulos, cada uno de los cuales se encuentra a su vez subdividido hasta llegar a elementos primarios. El detalle de descripción de estos elementos primarios no necesariamente debe alcanzar el nivel de las primitivas.

La metodología descendente es ampliamente utilizada en la actualidad para el modelado de un sistema digital, ya que permite dividir un diseño complejo en otros de menor complejidad, lo que permite detallar un módulo tanto como sea necesario sin llegar a un nivel de abstracción bajo.

Con base en lo expuesto en los párrafos anteriores, la metodología elegida para el modelado del procesador de nuestro sistema es la descendente y el tipo de modelado empleado en cada módulo es el comportamental.

El modelado o descripción de cada uno de los módulos que intervienen en esta parte del sistema es realizado mediante el lenguaje descriptor de hardware VHDL utilizando la herramienta de desarrollo ISE Foundation versión 6.3i e implementado en un FPGA XCV300 de la familia Virtex, ambos de la compañía Xilinx.

3.1.5.1.1. Módulo central de proceso

El **módulo central de proceso,** se encarga de gobernar el funcionamiento y la interacción de todos los módulos que forman al procesador para el reconocimiento de imágenes. Actúa como el nivel más importante de este diseño y está constituido por una máquina de estados con dos fases, operando con una frecuencia de 50 MHz. Este módulo, representado en la figura 3.11, incluye al **módulo de interfaz USB,** debido a que es utilizado en numerosas ocasiones por el **módulo central de proceso**. La descripción del **módulo de interfaz USB** se explica detalladamente en la sección 3.1.5.1.2.

La primera fase se encarga del aprendizaje, y está constituida por los estados *rd_usb, wr_ram_imag, wavelet_binari* y *mam_aprende*. La segunda, se encarga de la recuperación o reconocimiento de imágenes y la forman los estados *rd_usb, wr_ram_imag, wavelet_binari, mam_recupera, wr_ram_resul, rd_ram* y *wr_usb.*

Como se muestra en la figura 3.12, cuando la señal de *reset* es activa con un nivel lógico alto, las memorias *min* y *max* de aprendizaje son inicializadas; posteriormente, la máquina de estados que se encargará de la fase de aprendizaje en el proceso de reconocimiento da inicio. con el envió de las imágenes desde la PC.

La fase de aprendizaje principia en el estado *rd_usb,* en este estado, se leen del controlador de USB dos bytes o píxeles de información de la imagen proveniente de la PC. La lectura de cada byte se realiza cuando el **módulo de interfaz USB** indica con la señal *dato_usb* puesta en *'1'*, que el dato puede ser leído.

El siguiente estado, es el *wr_ram_imag,* donde se hace la petición al **módulo de control del sistema de memoria** mediante la señal *do_usb* puesta en *'1',* y la señal *usb_ram* puesta *'0'* de almacenar en una localidad de memoria a partir de la 0x00 los dos bytes anteriormente leídos. La secuencia de estos estados se repite hasta que se han recibido y almacenado los 345,600 bytes de la imagen completa enviada desde la PC.

En el estado *wavelet_binari,* la señal *ini_wav* puesta en *'1'* inicia la ejecución del **módulo de la DWT**, en este estado se aplica la DWT sobre la imagen, de este proceso únicamente se considera la parte promedio para las siguientes etapas, descartando las partes detalle. El término del cálculo de la DWT de nivel 4 inicia, en este mismo estado, la ejecución del **módulo de binarización**.

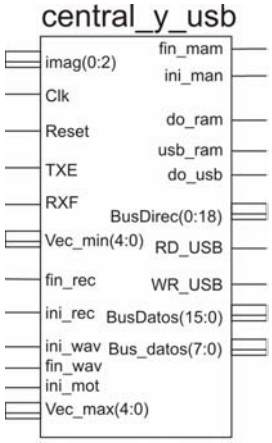

Figura 3.11. Módulo central de proceso e interfaz usb.

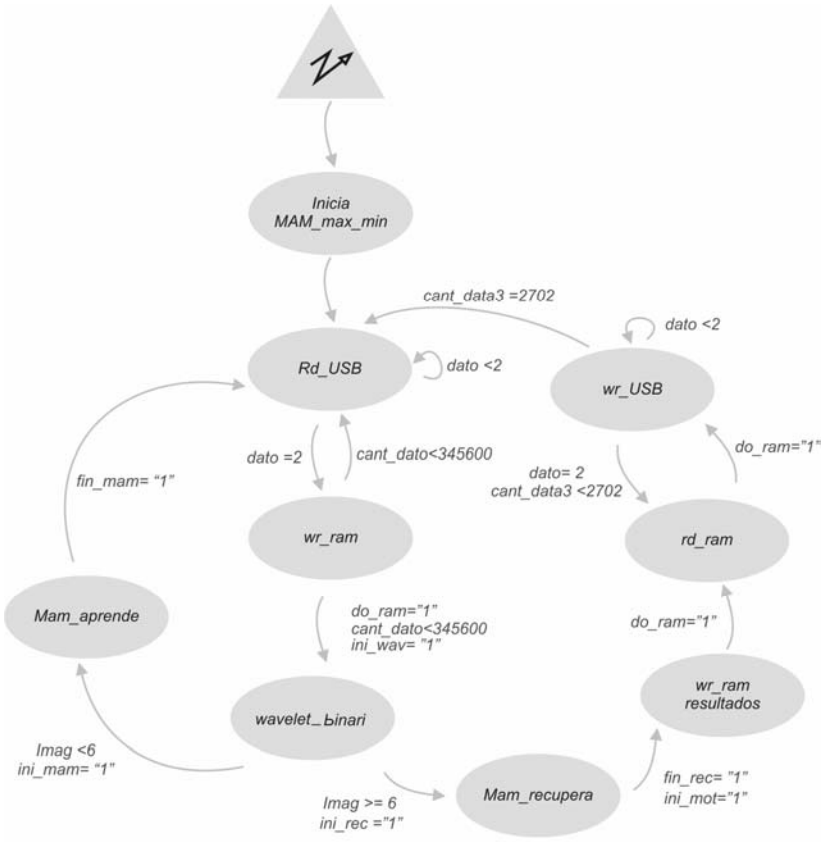

Figura 3.12. Máquina de estados del módulo central de proceso.

En este proceso, los coeficientes de la parte promedio de la DWT de nivel 4 son binarizados, teniendo por finalidad simplificar las operaciones a realizar con estos coeficientes en posteriores procesos.

Esta fase finaliza con el estado *mam_aprende,* el cual da inicio con la señal *ini_mam* puesta en *'1'* y desarrolla el proceso de aprendizaje de imágenes haciendo uso del **módulo de aprendizaje**. El fin de este estado es denotado mediante la señal *fin_mam* puesta en *'1'*; como resultado se obtienen 2 memorias que contienen la información codificada de las imágenes, MMH **W** (*min*) y MMH **M** (*max*). La generación de estas memorias hace uso de la imagen recibida y de un vector de 5 elementos, *vec_min* o *vec_max*, asociado a la imagen. Cada nueva imagen es asociada con un vector diferente, este vector funge como el vector de salida y la imagen como el vector de entrada del conjunto fundamental de asociaciones.

Esta fase se ejecuta para cada una de las 5 imágenes que deben ser aprendidas por el sistema. Una vez conseguidas las memorias **W** y **M** finales, puede iniciar su operación la fase de reconocimiento de imágenes.

Antes de aplicar el proceso de reconocimiento en el estado *mam_recupera*, la imagen es adquirida mediante el controlador de USB, almacenada en el sistema de memoria y se le aplican los procesos DWT y binarización.

En el estado *mam_recupera*, da inicio la ejecución del **módulo de recuperación** mediante la señal *ini_rec* puesta en *'1';* de este proceso se obtiene el vector o etiqueta de 5 elementos, *vec_min* o *vec_max,* asociado a la imagen reconocida. El término de este proceso se indica

activando la señal *fin_rec* (puesta en *'1'*) y con ello, se da inicio al funcionamiento del **módulo de control de actuadores**, mediante la señal *ini_mot* puesta en *'1';* que se encarga de interpretar la etiqueta obtenida y realiza la tarea correspondiente.

Los mismos vectores, *vec_min* y *vec_max,* en el siguiente estado, *wr_ram_resultados,* son almacenados en el sistema de memoria para ser enviados posteriormente a la PC.

Para permitir al usuario visualizar los resultados del procesamiento sobre la imagen, en el estado *rd_ram* la parte promedio de la DWT de nivel 4 y la binarización de la imagen son leídos de la memoria y en el estado *wr_usb* son enviados por USB a la PC; en total la transferencia incluye 2,702 bytes, que corresponden a la DWT, la binarización y las etiquetas obtenidas del reconocimiento. Al finalizar este estado, se regresa al estado *rd_usb* para recibir una nueva imagen para reconocer.

3.1.5.1.2. Módulo de interfaz USB

En el **módulo de interfaz USB**, figura 3.11, se generan las señales de control que permitirán hacer uso del controlador de USB DLP-USB245M, el cual servirá para realizar la interfaz de alta velocidad de transferencia de información entre la PC y el procesador para el reconocimiento de imágenes.

Este módulo es utilizado sólo cuando el **módulo central de proceso** lo requiere, ya sea para adquirir la imagen de la PC y su posterior almacenamiento en el banco de memoria, o bien, para enviar a la PC la información que permitirá visualizar el funcionamiento del sistema.

Como se muestra en la figura 3.13, la lectura es iniciada en el momento en que la señal *rxf='0',* indicando que hay datos en el controlador de USB para ser leídos. De acuerdo con el diagrama de tiempos de la figura 3.6 del ciclo de lectura del DLP-USB245M, en este **módulo de interfaz USB**, la señal *rd_usb* es puesta en *'0'* por un periodo de 80 ns activando la lectura de un byte, la señal *dato_usb* es puesta en *'1'* a los 60 ns de activada la lectura, indicando que el dato puede ser leído del bus de datos del controlador y respaldado en la señal *res_date* para ser almacenado posteriormente en el sistema de memoria.

En la misma figura, 3.13, se observa que la escritura se realiza si la señal *txe='0',* indicando que el buffer está desocupado o vacío y se puede realizar la escritura. Regido por el diagrama de tiempos de la figura 3.5, la señal *wr_usb* se pone en *'1'* por 80 ns activando la escritura de un byte. Para que la operación de escritura se realice exitosamente, el byte a ser escrito deberá encontrarse disponible en el bus *dato* (conectado al bus de datos del DLP-USB245M) mínimo 20 ns antes de desactivar la señal *wr_usb*.

La asignación de las señales de datos y control del DLP-USB245M en el FPGA es mostrada en la tabla 3.9 y la figura 3.14.

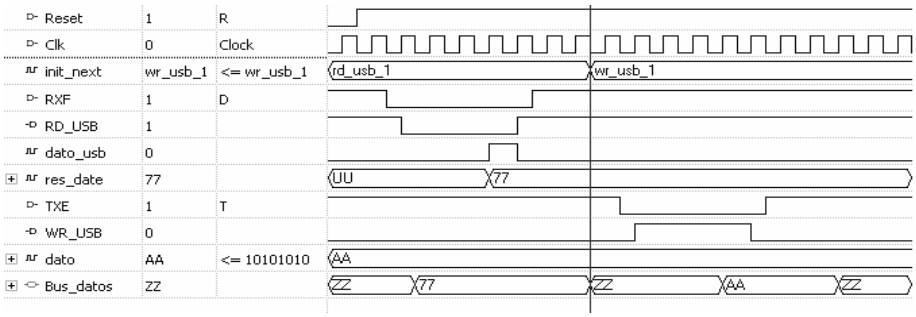

Figura 3.13. Diagrama de tiempos del módulo de interfaz USB modelado en el FPGA.

Tabla 3.9. Asignación de pines del FPGA y las líneas de control y datos del DLP-USB245M.

DLP-USB245M	FPGA XCV300
WR#	82
RD	81
TXE	84
RXF	85
D0	70
D1	71
D2	72
D3	73
D4	74
D5	78
D6	79
D7	80

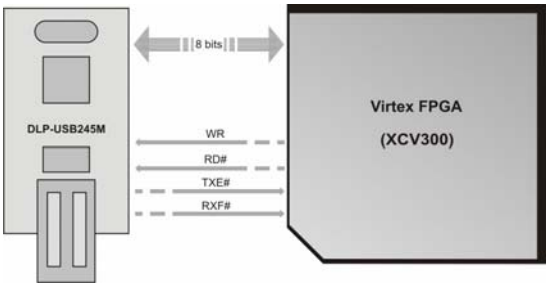

Figura 3.14. Esquema de conexión del DLP-USB245M con el FPGA.

3.1.5.1.3. Módulo de control del sistema de memoria

En el **módulo de control del sistema de memoria**, figura 3.15, se implementa un protocolo de interfaz para que cualquier módulo del procesador para el reconocimiento de imágenes pueda acceder al sistema de memoria.

Este módulo, indica mediante la señal *do_ram= '0'* que el sistema de memoria se encuentra disponible para acceder a él.

Los **módulos de aprendizaje, recuperación, DWT e interfaz USB** hacen la petición de acceso al banco de memoria mediante una señal de control; de igual forma, indican al **módulo de control del sistema de memoria** el tipo de operación a realizar (lectura o escritura) y la dirección de la localidad de acceso.

La figura 3.16 muestra el protocolo implementado por el **módulo de interfaz USB** para un acceso al sistema de memoria (el protocolo es el mismo para cualquier módulo).

El módulo hace la petición de acceso a la memoria poniendo la señal *do_usb* a un nivel lógico *'1'*. La operación a realizar se determina por la señal *usb_ram*, si *usb_ram* = '*1'* se trata de una operación de escritura, si *usb_ram* = '*0'* se trata de una operación de lectura.

El bus *direc_usb* define la dirección de acceso al sistema de memoria. El **módulo de control del sistema de memoria** indica la finalización de la operación poniendo la señal *do _ram* a un nivel lógico *'1'*, en ese momento el **módulo de interfaz USB** libera al **módulo de control del sistema de memoria** (*do_usb='0'*) para que pueda ser usado por otro módulo.

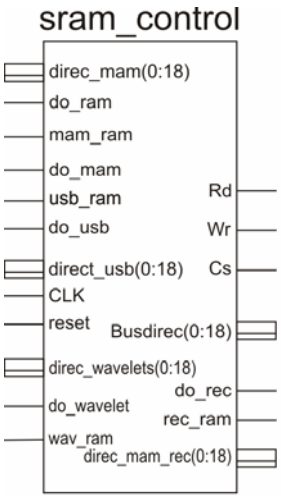

Figura 3.15. Módulo de control del sistema de memoria.

Una vez recibida la información para acceder a la memoria por parte de un módulo y considerando el diagrama de tiempos de la figura 3.8, el **módulo de control del sistema de memoria** implementa el protocolo de acceso al las memorias AS7C4096-15.

Primero, la localidad a acceder es puesta en el bus *Busdirec;* segundo, se activa la señal que determina el tipo de operación a realizar, *Rd* (/OE en la SRAM) para una lectura, *Wr* (/WR en la SRAM) para una escritura; tercero, la señal *Cs* se pone en *'0'* activando el /CE de la SRAM durante 20 ns; cuarto, la señal *do _ram* se pone en *'1'* indicando que la operación ha concluido; quinto, la señal *Rd* o *Wr* es deshabilitada.

La conexión del sistema de memoria con el FPGA se muestra en la figura 3.17 y en la tabla 3.10 la respectiva asignación de pines.

El sistema de memoria está formado por 512Kx16 localidades. La distribución del espacio de memoria disponible para el procesador es mostrado en la figura 3.18.

De la localidad 0x00000 a la 0x2A2FF son utilizadas para almacenar la imagen (720x480 píxeles) que será procesada.

La parte promedio de la DWT de nivel 4 de la imagen es almacenada de la localidad 0x00000 a la 0x002A2.

El resultado del proceso de binarización se almacena de la localidad 0x002A3 a la localidad 0x00545.

El espacio reservado para las MMH *min* y *max* es de la localidad 0x2A300 a la 0x2B02E y de la localidad 0x2B02F a la 0x2BD5D respectivamente.

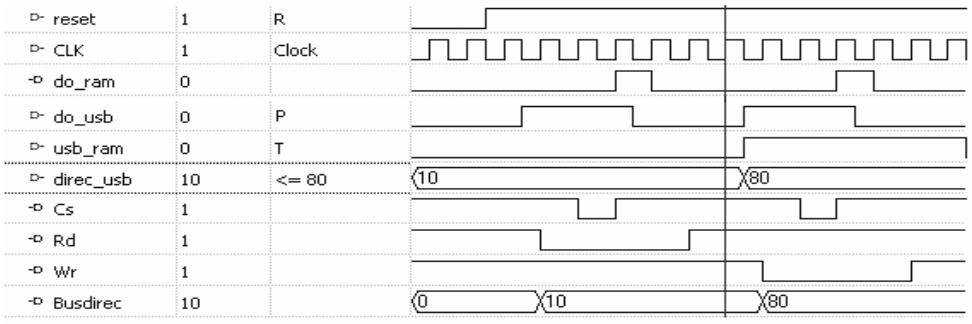

Figura 3.16. Diagrama de tiempos del protocolo de interfaz del módulo de control del sistema de memoria.

Tabla 3.10. Asignación de pines del FPGA para la interfaz con la SRAM AS7C4096

SRAM AS7C4096	FPGA XCV300	SRAM AS7C4096	FPGA XCV300	SRAM AS7C4096	FPGA XCV300
/CE	186	D10	218	A7	189
/OE	228	D11	220	A8	188
/WE	201	D12	221	A9	187
D0	202	D13	222	A10	238
D1	203	D14	223	A11	237
D2	205	D15	224	A12	236
D3	206	A0	200	A13	235
D4	207	A1	199	A14	234
D5	208	A2	195	A15	232
D6	209	A3	194	A16	231
D7	215	A4	193	A17	230
D8	216	A5	192	A18	229
D9	217	A6	192		

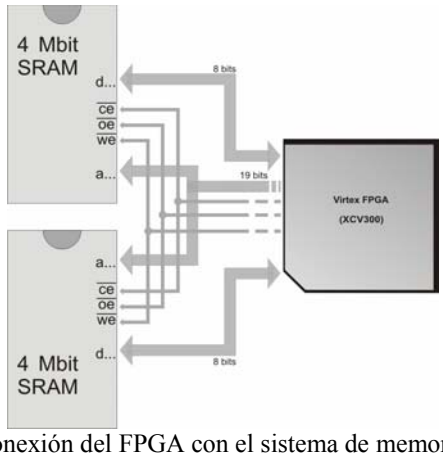

Figura 3.17. Esquema general de conexión del FPGA con el sistema de memoria.

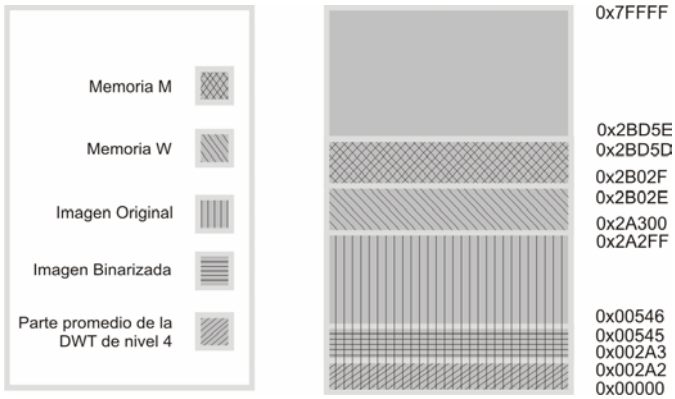

Figura 3.18. Mapa de organización del sistema de memoria.

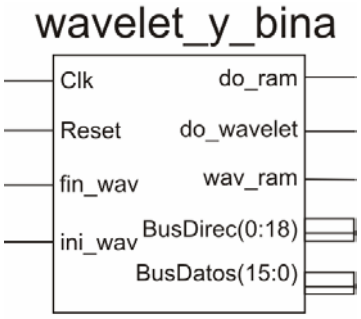

Figura 3.19. Módulo de la DWT y binarización.

3.1.5.1.4. Módulo de la DWT

En el **módulo de la DWT**, se implementa el algoritmo de la DWT bidimensional de nivel 4 sobre una imagen de 720x480 píxeles, como resultado se obtiene una imagen de 45x30 píxeles (coeficientes de la parte promedio) la cual es almacenada en el sistema de memoria a partir de la localidad 0x00000, espacio reservado para este propósito, como lo muestra la figura 3.18.

En este módulo, representado en la figura 3.19, se incluye el **módulo de binarización,** el cual da inicio al término de la ejecución del proceso que calcula la DWT de nivel 4; en la sección 3.1.5.1.5 se explica con más detalle el funcionamiento de este módulo.

La figura 3.20, muestra un esquema a bloques del procedimiento desarrollado para calcular la DWT bidimensional de un solo nivel. Este proceso hace uso de una Wavelet Haar, debido a que sólo se utilizaran los coeficientes correspondientes a la parte promedio, se aplican únicamente los filtros pasa bajo sobre la imagen almacenada en memoria a partir de la localidad 0x00000.

La imagen diezmada y con un suavizado de baja resolución resultante, es almacenada a partir de esta misma localidad de memoria.

Para calcular la DWT de cuarto nivel, el procedimiento anterior se ejecuta cuatro veces, figura 3.21. El **módulo central de proceso** indica el inicio del cálculo de la Wavelet de nivel cuatro mandando a la señal *ini_wav* a un nivel lógico alto.

La finalización del proceso es señalada con un nivel lógico alto de la señal *fin_wav*. Internamente, el nivel de la DWT que se va a calcular es indicado mediante la señal *nivel* y se conoce con un nivel lógico alto de la señal *fin_waval* cuando éste ha concluido.

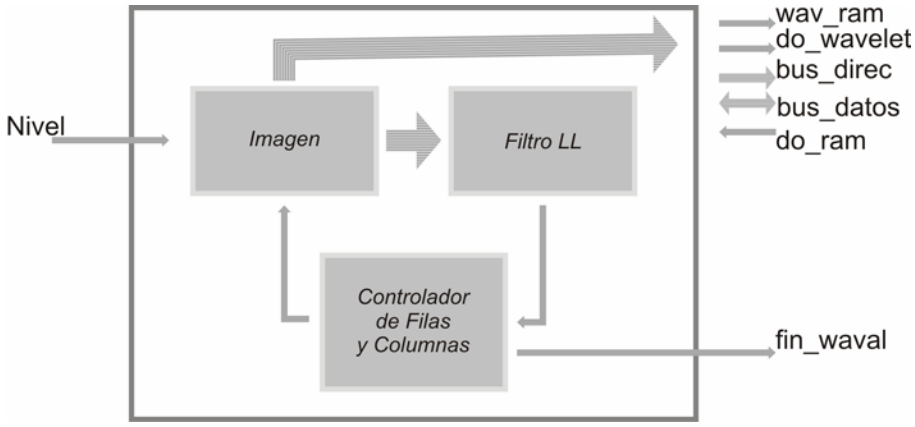

Figura 3.20. Proceso para realizar la DWT de 2 dimensiones.

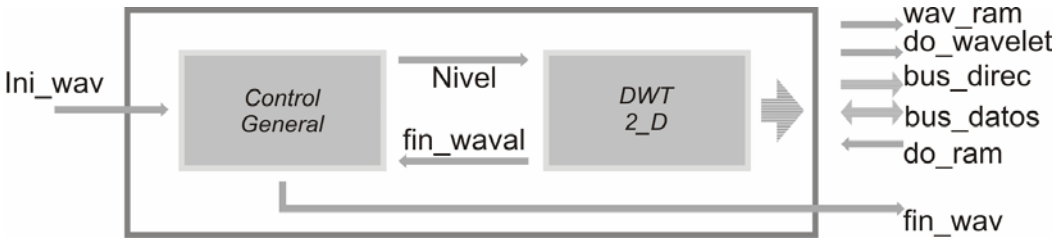

Figura 3.21. Módulo para calcular los cuatro niveles de la DWT bidimensional.

Tabla 3.11 Tabla del conjunto fundamental de asociaciones

Patrón/imagen de entrada	Patrón de salida o etiqueta
Imagen 1: avance	"0 0 0 0 1"
Imagen 2: alto	"0 0 0 1 1"
Imagen 3: vuelta a la derecha	"0 0 1 1 1"
Imagen 4: retorno	"0 1 1 1 1"
Imagen 5: vuelta la izquierda	"1 1 1 1 1"

3.1.5.1.5. Módulo de binarización

En el **módulo de binarización**, figura 3.19, la imagen en tonos de grises de 45x30 píxeles obtenida del **módulo de la DWT** y almacenada en el sistema de memoria a partir de la localidad 0x00000, se convierte a una imagen binaria mediante una comparación de cada píxel con un "*nivel de umbral*" de 0x25. Para los valores mayores o menores a este valor de comparación, cada píxel toma los valores de "0000001" y "0000000" respectivamente. Este proceso tiene por función obtener un mejor desempeño de las etapas siguientes, aprendizaje y recuperación. La imagen binarizada es almacenada en el sistema de memoria a partir de la localidad 0x002A3.

3.1.5.1.6. Módulo de aprendizaje

El **módulo de aprendizaje,** figura 3.22, implementa los algoritmos de la fase de aprendizaje de las Memorias Heteroasociativas Morfológicas **M** y **W**.

Este módulo hace uso del conjunto fundamental de asociaciones de la tabla 3.11, formado por las 5 imágenes de la figura 3.1 que simbolizan a los patrones de entrada y los 5 patrones de salida que representan las etiquetas asociadas a cada imagen. Este módulo genera las memorias heteroasociativas **W** y **M**, cada una de ellas constituida por 5x1350 elementos, estas matrices son almacenadas en el sistema de memoria a partir de las localidades 0x2A02F y 0x2B300 respectivamente, espacio reservado para este propósito como se observa en la figura 3.18.

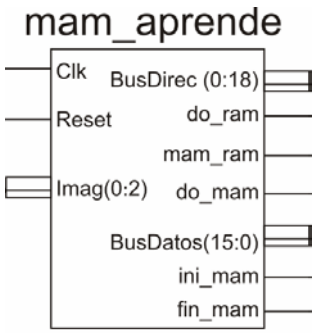

Figura 3.22. Módulo de aprendizaje.

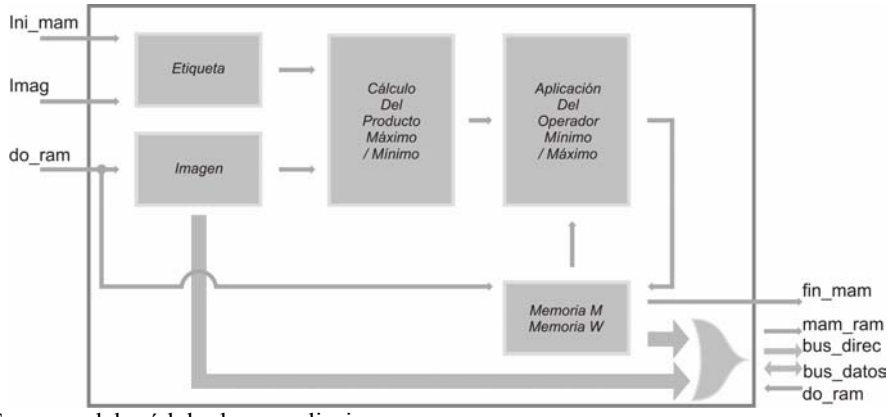

Figura 3.23. Esquema del módulo de aprendizaje.

El esquema de la figura 3.23 muestra el procedimiento que se implementa sobre cada una de as imágenes para generar una memoria heteroasociativa **W** o **M**. Este proceso da inicio cuando el **módulo central de proceso** activa la señal *ini_mam* con un nivel lógico alto. La señal *imag* indica el número de patrón de entrada o imagen (para esta aplicación es entre 1 y 5) que será procesada, a ésta se le asigna su correspondiente vector de salida; dicha imagen proviene del **módulo de binarización** y está almacenada en el sistema de memoria en las localidades 0x002A3 hasta 0x00545.

El módulo de aprendizaje generara la memoria en dos fases. Primero, para cada imagen se calcula la matriz $y^{\mu} \nabla \left(-x^{\mu}\right)^{t}$ o $y^{\mu} \Delta \left(-x^{\mu}\right)^{t}$ para una memoria **M** o una **W**, donde y y x representan a la etiqueta e imagen respectivamente. Segundo, se aplica el operador mínimo o el operador máximo entre **M** o **W** y la memoria almacenada en el sistema de memoria, el resultado substituye al contenido por la memoria.

Este proceso se repite para cada una de las imágenes que forman parte del conjunto fundamental de asociaciones.

Antes de iniciar el proceso de creación de las memorias **W** y **M**, éstas son inicializadas con valores predeterminados.

La conclusión del proceso de aprendizaje, es indicada al **módulo central de proceso** activando la señal *fin_mam* con un nivel lógico alto.

3.1.5.1.7. Módulo de recuperación

En el **módulo de recuperación**, figura 3.24, se implementan los algoritmos de recuperación para las Memorias Heteroasociativas Morfológicas **M** y **W**.

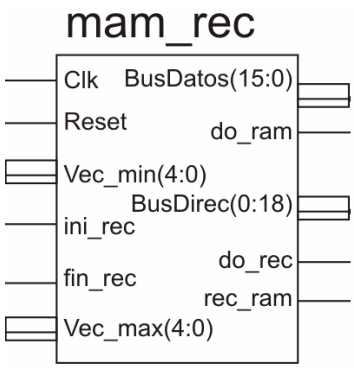

Figura 3.24. Módulo de recuperación.

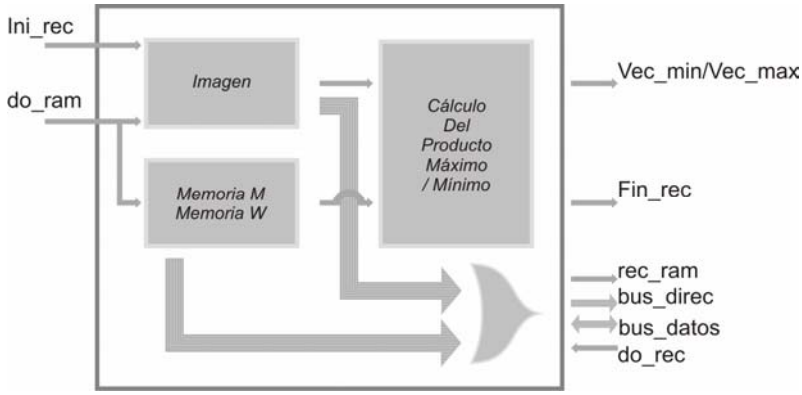

Figura 3.24. Esquema del módulo de recuperación.

El proceso de recuperación, como se muestra en el esquema de la figura 3.25, inicia cuando el **módulo central de proceso** activa la señal *ini_rec* con un nivel lógico alto. Este proceso hace uso de las memorias generadas por el **módulo de aprendizaje** (almacenadas en el sistema de memoria a partir de la localidad 0x2A02F) y de la imagen a reconocer, esta imagen está contenida en el sistema de memoria en las localidades 0x002A3-0x00545 y se trata de una imagen de 45x35 píxeles resultante de aplicar los procesos de DWT y binarización sobre una imagen de 720x480 píxeles entregada por la interfaz de usuario.

La fase de recuperación consiste en aplicar el producto máximo, ecuación 19, entre la memoria heteroasociativa **W** y la imagen de entrada a reconocer, o aplicar el producto mínimo, ecuación 18, entre la memoria heteroasociativa **M** y la imagen de entrada a reconocer.

El vector resultante de esta operación, *vec_min* o *vec_max*, corresponde a la etiqueta asociada a la imagen introducida para su identificación, estos vectores son parámetros de entrada del **módulo de control de actuadores**. La finalización del proceso de recuperación se indica al **módulo central de proceso** activando la señal *fin_rec* con un nivel lógico alto.

3.1.5.1.8. Módulo de control de actuadores

El **módulo de control de actuadores,** figura 3.25**,** tiene como funciones interpretar el valor del vector resultante del proceso de recuperación, generar las señales de control para dos puentes H que controlan el giro de dos motores y mostrar en un display de 7 segmentos el número correspondiente a la imagen reconocida. Estas acciones expresan el reconocimiento de una imagen del conjunto fundamental de asociaciones hecho por el procesador para el reconocimiento de imágenes.

El **módulo central de proceso** da inicio al funcionamiento de este módulo mediante la señal *ini_mot='1',* en respuesta, el proceso de interpretación de la etiqueta es realizado.

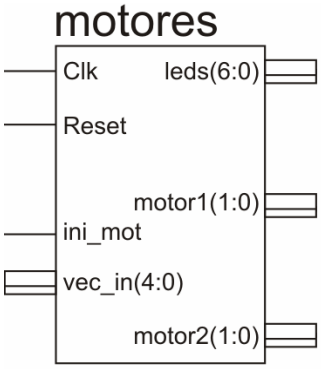

Figura 3.25. Módulo de control de actuadores.

Tabla 3.12. Asignación de pines de la interfaz de los actuadotes con el FPGA.

PUENTE H1	FPGA XCV300
C1	102
C0	101

PUENTE H2	FPGA XCV300
C1’	103
C0’	108

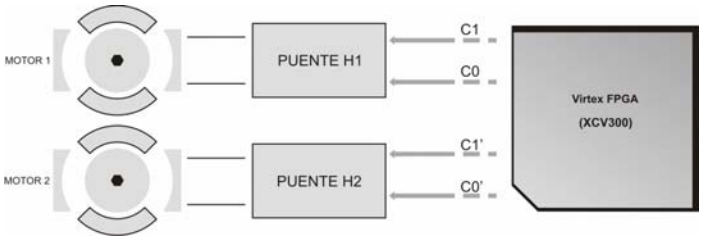

Figura 3.26. Esquema general de la interfaz del FPGA con los puentes H.

La interpretación, regida por la tabla 3.11, modifica el estado de los actuadores y muestra en el display el número de la imagen reconocida.

Cuando *tmpvec_in* ="0000001", ambos motores avanzan y el número ‘1’ es desplegado.

Cuando *tmpvec_in* ="0000011", los motores se detienes y el número ‘2’ es desplegado.

Cuando *tmpvec_in* ="0000111", el motor izquierdo avanza y el derecho es detenido, después de 20 seg. aproximadamente ambos motores avanzan, y el número ‘3’ es desplegado.

Cuando *tmpvec_in* ="0001111", el motor izquierdo avanza y el derecho es detenido, después de 40 seg. aproximadamente ambos motores avanzan, y el número ‘4’ es desplegado.

Cuando *tmpvec_in* ="00011111", el motor derecho avanza y el izquierdo está detenido, después de 20 seg. aproximadamente ambos motores avanzan, y el número ‘5’ es desplegado.

Si la etiqueta no es reconocida, el estado de los motores y el despliegue no se ven afectados.

La interfaz de los actuadotes y el dispaly de 7 segmentos al FPGA es mostrada en las figuras 3.26 y 3.27 respectivamente, la asignación de pines de estas conexiones puede ser consultada en las tablas 3.12 y 3.13.

Tabla 3.13. Asignación de pines de la interfaz del display de 7 segmentos con el FPGA.

DISPLAY	FPGA
A	139
B	144
C	133
D	124
E	132
F	141
G	174

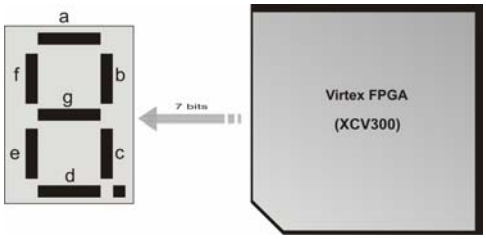

Figura 3.27. Esquema general de la interfaz del FPGA con el display de 7 segmentos.

3.1.5.2. Interfaz con el usuario

El programa de interfaz de usuario fue realizado utilizando la herramienta Borland C++ Builder en su versión 6.0, mediante una programación orientada a objetos.

La interfaz de usuario tiene por funciones, primero, permitir enviar las imágenes que forman parte del conjunto fundamental de asociaciones al procesador para el reconocimiento de imágenes; segundo, permite al usuario visualizar los resultados del procesamiento realizado sobre la imagen por cada uno de los módulos que componen al procesador para el reconocimiento de imágenes.

La figura 3.28 muestra el diagrama de flujo del programa de la interfaz de usuario, en él se puede apreciar que se basa en un programa compuesto por funciones que se ejecutan al presionar el botón que indica el inicio de la tarea deseada.

Con la finalidad de poder entablar una comunicación con el procesador para el reconocimiento de imágenes mediante la interfaz de alta velocidad, se debe inicializar el puerto USB.

Realizada esta función, el usuario puede elegir la transmisión de las imágenes que forman parte del conjunto fundamental de asociaciones hacia el sistema de reconocimiento de imágenes, cada una de estas imágenes es mostrada simbólicamente en la interfaz de usuario.

Posteriormente, el usuario puede elegir enviar alguna imagen al sistema para el reconocimiento de imágenes, éste regresara los resultados de cada una de las etapas del procesamiento sobre la imagen para que la interfaz de usuario pueda presentarlos en forma visual al usuario.

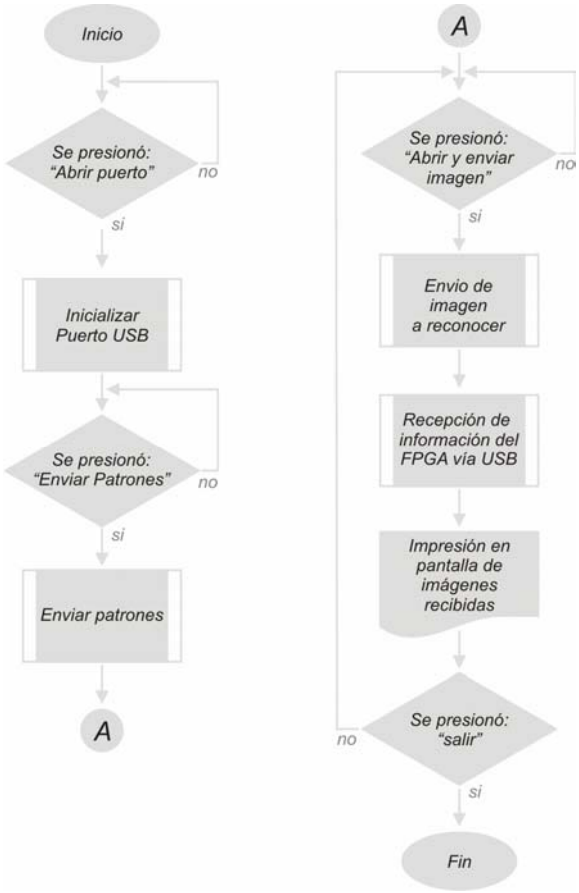

Figura 3.28. Diagrama de flujo del programa interfaz de usuario.

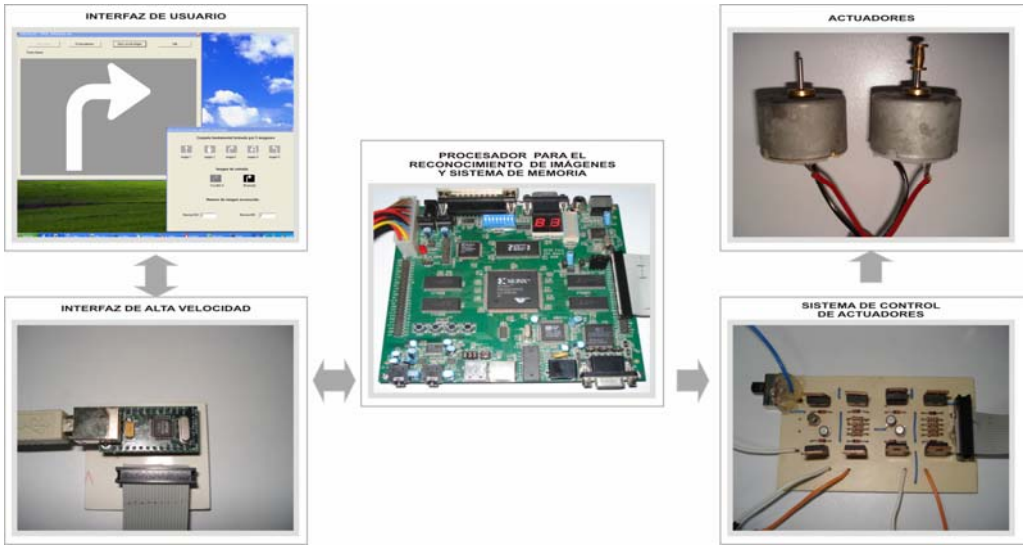

Figura 3.29. Sistema final para el reconocimiento de imágenes digitales.

3.1.6. Integración de componentes HW y SW

La figura 3.29 muestra la integración de los elementos hardware y software para formar el sistema para el reconocimiento de imágenes. Cabe aclarar que este sistema para el reconocimiento de imágenes no se ha integrado en una sola tarjeta de aplicación y se sigue haciendo uso de una tarjeta de desarrollo de un dispositivo como parte central del sistema hardware, debido a que es sólo una parte de un sistema más complejo encaminado a aplicaciones relacionadas con la visión artificial.

La parte central del sistema está compuesta por la tarjeta de desarrollo XSV300 Board V1.0 de la compañía Xess, la cual contiene al banco de memoria a utilizar, al display de 7 segmentos que desplegará la etiqueta asignada a la imagen reconocida y al FPGA XCV300 Virtex, circuito digital configurable utilizado para la implementación del procesador para el reconocimiento de imágenes.

La interfaz de alta velocidad con una PC, la cual permite emular la adquisición de imágenes mediante el programa de interfaz de usuario, se realiza usando el controlador de USB DLP-USB245M.

El sistema de actuadores está formado por dos motores de CD y los elementos de potencia, puentes H, que proporcionan la corriente necesaria para su operación y permiten el control por parte del procesador para el reconocimiento de imágenes.

3.1.7. Prueba y verificación del producto

Una vez concluida la integración del sistema se procede a realizar las pruebas necesarias para verificar su correcto funcionamiento e identificarlo como un sistema terminado; las pruebas efectuadas fueron las siguientes:

- Para corroborar el correcto funcionamiento de la interfaz de alta velocidad y del sistema de memoria, se enviaron imágenes desde la PC al procesador para el reconocimiento de imágenes, este último después de almacenarlas en el sistema de memoria, las recupera y reenvía a la PC donde la interfaz de usuario las recibe y muestra al usuario, figura 3.30. La velocidad de transferencia de información que puede alcanzar el procesador para el reconocimiento de imágenes con el DLP-USB245M es de 100 Megabaudios. Cuando la

información es leída o almacenada en el banco de memoria, la velocidad de transferencia es de 61.5 Megabaudios.

- La figura 3.31 muestra los resultados de la evaluación del cálculo de la DWT bidimensional enviados por el procesador para el reconocimiento de imágenes, en la figura se puede apreciar la función de escalamiento o parte promedio de los niveles 1, 2, 3 y 4 de una de las imágenes del conjunto fundamental de asociaciones
- El resultado del proceso de binarización de la función de escalamiento de nivel 4 de la DWT por parte del procesador para el reconocimiento de imágenes se muestra en la figura 3.32.
- La valoración de los procesos de aprendizaje y reconocimiento de las imágenes se llevó acabo en dos pasos. Primero, una vez activado el proceso de aprendizaje del procesador para el reconocimiento de imágenes, le fueron enviadas las imágenes que forman al conjunto fundamental de asociaciones, terminando con esto la fase de aprendizaje. Segundo, se enviaron imágenes al sistema, éste después de aplicar el proceso de reconocimiento indicó mediante un display de 7 segmentos el resultado del proceso y lo envió a la interfaz de usuario donde fueron mostrados en pantalla al usuario, figura 3.33. El porcentaje de reconocimiento de las imágenes del conjunto fundamental y que sometidas a esta prueba fue del 100%, siendo este comportamiento el necesario para considerar terminado el sistema.

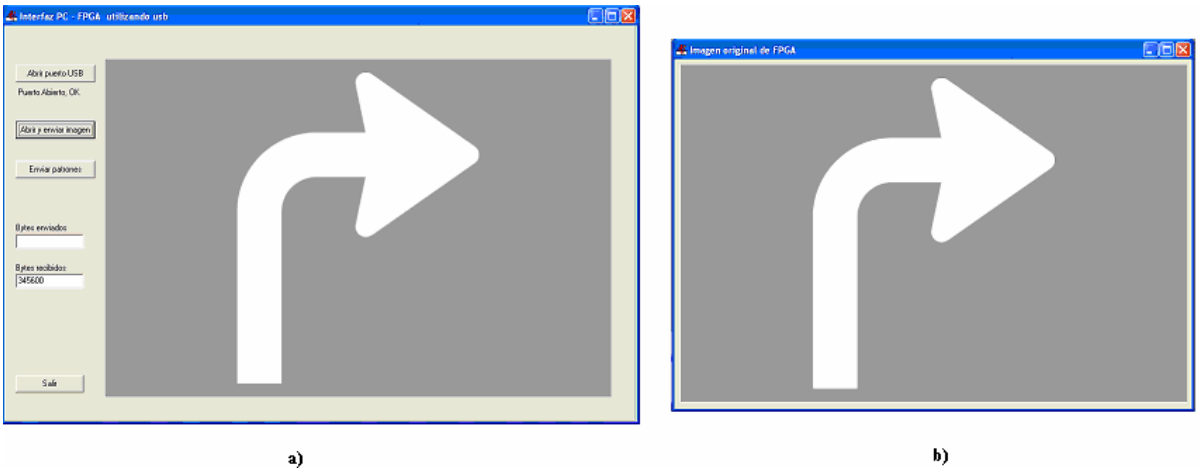

Figura 3.30. Prueba de envío y recepción de imágenes al procesador para reconocimiento de imágenes, (a) imagen trasmitida, (b) imagen recibida.

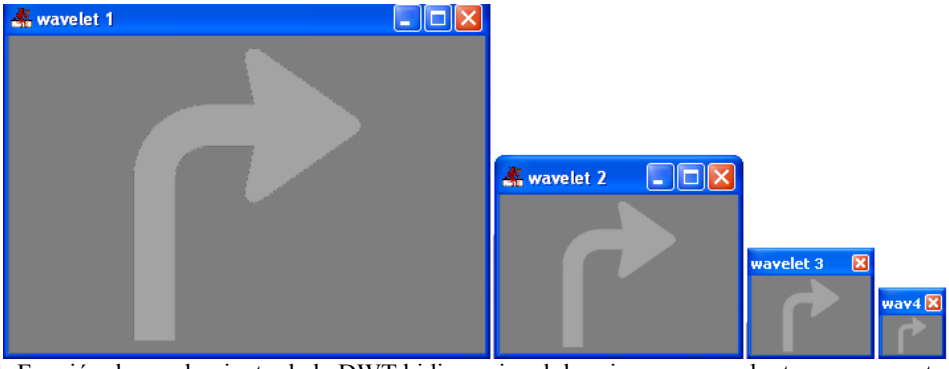

Figura 3.31. Función de escalamiento de la DWT bidimensional de primero, segundo, tercero y cuarto nivel.

Figura 3.32. Binarización de la función de escalamiento del nivel 4 de la DWT.

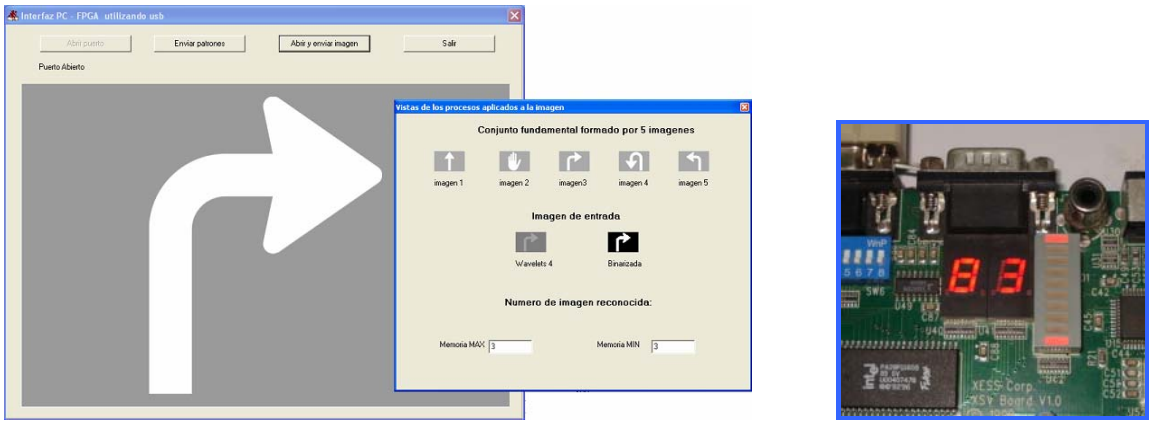

Figura 3.33. Visualización en el programa de interfaz de usuario y en el dispaly derecho de 7 segmentos la etiqueta asignada a la imagen reconocida; en este caso ‘3’.

4. Resultados

Este capítulo presenta los resultados experimentales realizados sobre el sistema para el reconocimiento de imágenes, cuando se utilizan imágenes alteradas con ruidos típicos que pueden ser inducidos por etapas previas a la del reconocimiento. También incluye los resultados generados por la herramienta ISE Foundation en la implementación del sistema.

4.1. Resultados experimentales con imágenes alteradas con ruido

El "ruido" corresponde a una distorsión o alteración de los valores de los píxeles en una imagen modelo u original. Cuando se adquiere una imagen digital suele estar contaminada por ruido. El ruido se debe, en muchas ocasiones, al equipo electrónico utilizado en la captura de las imágenes y a los efectos físicos implícitos en la transmisión.

Para simular los efectos que causan los procesos de captura y transmisión sobre las imágenes, se eligió añadir ruido Gaussiano y ruido Uniforme a éstas, para este propósito se utilizó la herramienta de diseño gráfico CorelDRAW X3.

El *ruido Gaussiano*, se caracteriza por tener un espectro de energía constante para todas las frecuencias. Cuando se presenta este problema, el valor exacto de cualquier píxel es diferente cada vez que se captura la misma imagen. Este efecto, suma o resta un determinado valor al nivel de gris real y es independiente de los valores que toma la imagen.

Para este ruido, la densidad de probabilidad responde a una distribución normal (o distribución de Gauss) definida por *f(x),* donde *m* es la media y σ^2 es la varianza [42]:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-m)^2}{2\sigma^2} \right] \qquad \textbf{Ecuación 39}$$

La figura 4.1(b) expresa como el ruido Gaussiano tiene un efecto general en toda la imagen, es decir, la intensidad de cada píxel de la imagen se ve alterada en cierta medida con respecto a la intensidad en la imagen original.

El *ruido uniforme,* se caracteriza por estar distribuido uniformemente en toda la imagen. Éste, en un intervalo real [a,b] con a < b, se representa mediante la siguiente función de densidad de probabilidad *f(x)* [42]:

$$f(x) = \begin{cases} \dfrac{1}{(b-a)} & \text{si a} \le \text{x} \le \text{b} \\ 0 & \text{en otro caso} \end{cases} \qquad \textbf{Ecuación 40}$$

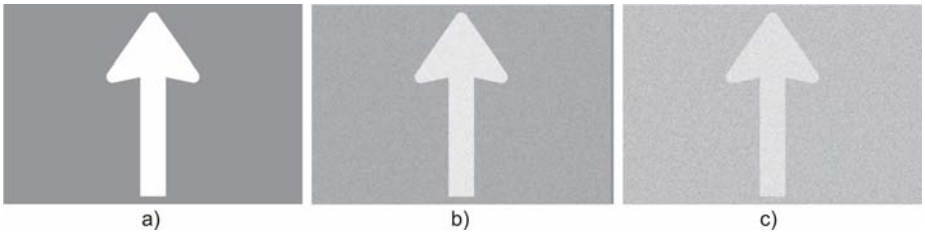

Figura 4.1. Efectos de aplicar ruido Gaussiano y Uniforme a una imagen; (a) imagen original, (b) imagen con ruido Gaussiano, (c) imagen con ruido Uniforme.

La figura 4.1(c) muestra el efecto resultante de aplicar este tipo de ruido a una imagen.

Para evaluar la eficiencia y desempeño de la implementación de las Memorias Heteroasociativas Morfológicas **M** y **W** dentro del sistema para el reconocimiento de imágenes, las imágenes pertenecientes al conjunto fundamental de asociaciones, figura 3.2, fueron alteradas con ruido Gaussiano y Uniforme en sus tres variantes, aditivo, sustractivo y mixto, cada uno de ellos en diferentes porcentajes, 5%, 10%, 15%, 20%, 25% y 30%, con un nivel base de gris de 154. En total fueron generadas 180 imágenes alteradas, 36 por cada una de las imágenes originales. La figura 4.2 muestra imágenes alteradas con ruido Gaussiano y Uniforme.

Cada una de las imágenes generadas fue enviada 10 veces al sistema para el reconocimiento de imágenes, los resultados obtenidos en el proceso de reconocimiento se resumen en las tablas 4.4, 4.5, 4.6, 4.7 y 4.8.

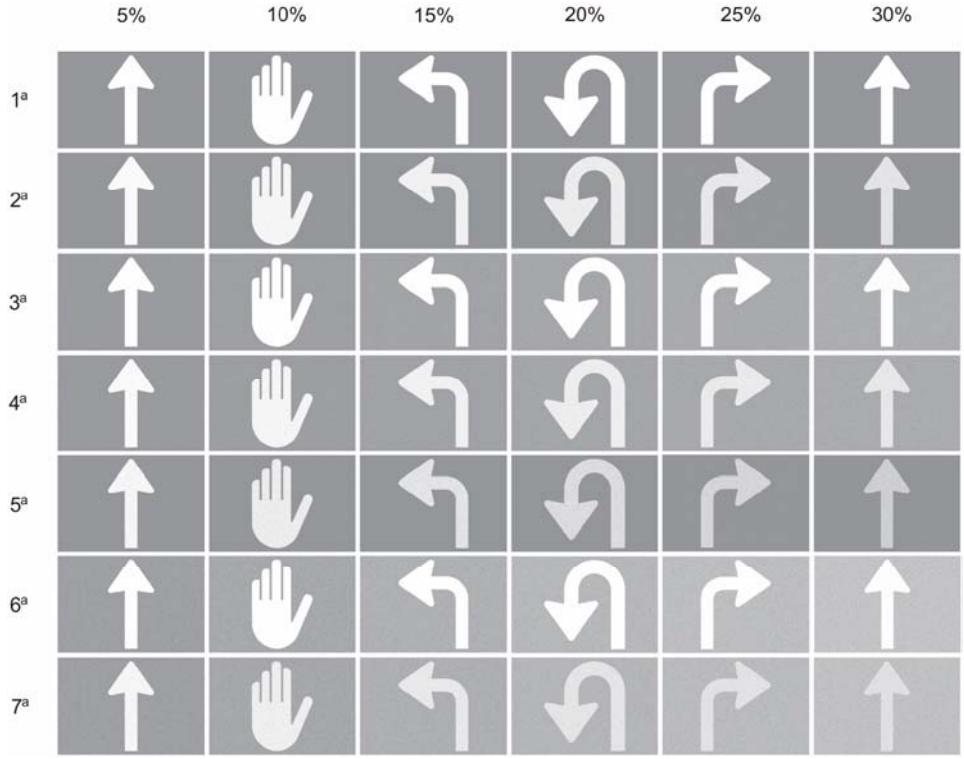

Figura 4.2. Imágenes alteradas con ruido Gaussiano y Uniforme. La 1ª fila contiene las imágenes originales; 2ª fila imágenes con ruido Gaussiano sustractivo de 5%, 10%, 15%, 20%, 25% y 30%; 3ª fila imágenes con ruido Gaussiano aditivo de 5%, 10%, 15%, 20%, 25% y 30%; 4ª fila imágenes con ruido Gaussiano mixto de 5%, 10%, 15%, 20%, 25% y 30%, 5ª fila imágenes con ruido Uniforme sustractivo de 5%, 10%, 15%, 20%, 25% y 30%; 6ª fila imágenes con ruido Uniforme aditivo de 5%, 10%, 15%, 20%, 25% y 30%; 7ª fila imágenes con ruido Uniforme mixto de 5%, 10%, 15%, 20%, 25% y 30%.

Tabla 4.4. Tabla de resultados del reconocimiento de la imagen 'Siga'.

	RUIDO GAUSSIANO		RUIDO UNIFORME	
Ruido Aditivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	0	100
20%	0	100	0	100
25%	0	100	0	100
30%	0	100	0	100
Ruido Sustractivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	100	0
20%	100	100	100	0
25%	100	0	100	0
30%	100	0	100	0
Ruido Mixto	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	0	100
20%	100	100	0	100
25%	100	100	0	100
30%	100	100	0	100

Tabla 4.5. Tabla de resultados del reconocimiento de la imagen 'Alto'.

	RUIDO GAUSSIANO		RUIDO UNIFORME	
Ruido Aditivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	0	100
10%	100	100	0	100
15%	0	100	0	100
20%	0	100	0	100
25%	100	100	0	100
30%	0	100	0	100
Ruido Sustractivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	100	100
20%	100	100	100	100
25%	100	100	100	100
30%	100	100	100	100
Ruido Mixto	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	100	100
20%	100	100	0	100
25%	100	100	0	100
30%	0	100	0	100

Tabla 4.6. Tabla de resultados del reconocimiento de la imagen 'Derecha'.

	RUIDO GAUSSIANO		RUIDO UNIFORME	
Ruido Aditivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	0	100
10%	100	100	0	100
15%	100	100	0	100
20%	100	100	0	100
25%	0	100	0	100
30%	0	100	0	100
Ruido Sustractivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	100	0
20%	0	100	100	0
25%	0	100	100	0
30%	0	100	100	0
Ruido Mixto	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	0	100
20%	100	100	100	100
25%	100	100	0	100
30%	100	100	0	100

Tabla 4.7. Tabla de resultados del reconocimiento de la imagen 'Retorno'.

	RUIDO GAUSSIANO		RUIDO UNIFORME	
Ruido Aditivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	100	100
20%	100	100	100	100
25%	100	100	0	100
30%	100	100	100	100
Ruido Sustractivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	100	0
20%	100	100	100	0
25%	100	0	100	0
30%	100	0	100	0
Ruido Mixto	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	0	100
20%	100	100	100	100
25%	100	100	0	100
30%	100	100	0	100

Tabla 4.8. Tabla de resultados del reconocimiento de la imagen 'Izquierda'.

	RUIDO GAUSSIANO		RUIDO UNIFORME	
Ruido Aditivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	0	100
20%	0	100	0	100
25%	0	100	0	100
30%	0	100	0	100
Ruido Sustractivo	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	0
10%	100	0	100	0
15%	100	0	100	0
20%	100	0	100	0
25%	100	0	100	0
30%	100	0	100	0
Ruido Mixto	**% de Rec.**		**% de Rec.**	
	MAM Min	**MAM Max**	**MAM Min**	**MAM Max**
5%	100	100	100	100
10%	100	100	100	100
15%	100	100	0	100
20%	100	100	100	100
25%	100	100	0	100
30%	100	100	0	100

Estas tablas comparan el desempeño y eficiencia que tienen las MAM W y M dentro del procesador para el reconocimiento de imágenes.

En términos generales, las tablas ilustran que para imágenes modificadas con los diferentes porcentajes de ruido aditivo Gaussiano y Uniforme, las MAM **M** reconocieron en un 100% a cada una de ellas, mientras que las MAM's **W** reconocieron en un 100% las imágenes a las que les fue aplicado los diferentes porcentajes de ruido sustractivo Gaussiano y Uniforme.

Dos hechos que indican que las MAM **M** resultan con un mejor desempeño que las **W** son, primero, en algunas imágenes, como por ejemplo la imagen 'Alto', con ruido sustractivo Gaussiano y Uniforme son reconocida en un 100% por este tipo de memorias; segundo, como es ilustrado en las tablas, las imágenes con los diferentes porcentajes de ruido mixto Uniforme y Gausssiano, fueron reconocidas en un 100% por las MAM **M**, mientras que las MAM **W** sólo reconocen las imágenes a las que se ha aplicado ruido mixto Gaussiano.

4.2. Resultados de la implementación

El resumen de los resultados generados por la herramienta ISE Foundation en la implementación del procesador para el reconocimiento de imágenes al utilizar un FPGA Virtex XCV300 se muestra en la tabla 4.1. Como puntos importantes de la implementación pueden ser resaltados la frecuencia máxima de operación del sistema indicada por la herramienta, 34.254 MHz (periodo de 29.194ns), y el porcentaje de utilización del dispositivo; de este último punto se puede concluir, considerando el número de *slices* que contiene este FPGA, que quedan suficientes recursos como para hacer adecuaciones y/o ampliaciones al diseño del sistema para reconocimiento de imágenes.

Tabla 4.1. Reporte de Place & Route

Dispositivo	Recursos uso / total	Porcentaje de uso
Slices	1327 / 3027	43%
Slice Flip Flops	1025 / 6144	16%
LUTs de 4 entradas	2270 / 6144	36%
IOBs	62 / 166	37%
TBUFs	48 / 3200	1%
GCLKs	1 / 4	25%

El apéndice C muestra el esquema general de la implementación del procesador para el reconocimiento de imágenes.

El sistema para el reconocimiento de imágenes tiene por finalidad ser orientado a aplicaciones de visión artificial en sistemas autónomos en tiempo real, un aspecto importante a considerar en este tipo de sistemas es la velocidad de procesamiento, las tablas 4.2, 4.3 y 4.4 muestran la velocidad de procesamiento, expresada en ciclos de reloj consumidos, de los procesos de DWT, aprendizaje de las MAM y recuperación de las MAM respectivamente.

Tabla 4.2. Implementación del algoritmo de la DWT de nivel 4 aplicado a una imagen de 720x480 píxeles.

WAVELET	CICLOS DE RELOJ
Nivel 1	1,684,800
Nivel 2	421,200
Nivel 3	105,300
Nivel 4	26,325
Total	2,237,625

Tabla 4.3. Implementación del algoritmo de aprendizaje de las MAM.

MEMORIA DE APRENDIZAJE	CICLOS DE RELOJ
1 imagen	77,625
5 imágenes	388,125

Tabla 4.4. Implementación del algoritmo de recuperación de las MAM.

MEMORIA DE RECUPERACIÓN	CICLOS DE RELOJ
1 imagen	54,000

La tabla 4.5, muestra las velocidades de transferencia de información alcanzada entre el procesador para el reconocimiento de imágenes y el controlador de USB el DLP-USB245M.

Tabla 4.5. Velocidad alcanzada por la interfaz USB

Transferencia de información	Megabaudios
Vía FPGA a DLPUSB245M	100
Vía FPGA a DLPUSB245M con acceso a SRAM	61.5

5. Conclusiones y perspectivas

En este trabajo se ha abordado el diseño e implementación de una arquitectura en un FPGA utilizando un lenguaje descriptor de hardware, y el estudio y modelado de un algoritmo empleado en el nivel intermedio de procesamiento de un sistema de visión artificial.

Para emular las tareas de adquisición y digitalización de imágenes hechas por el nivel bajo de procesamiento de un sistema de visión artificial, se ha propuesto un sistema de interfaz de alta velocidad entre la PC y el FPGA XCV300 de la firma Xilinx. Para este propósito se eligió una interfaz por un puerto USB utilizando el controlador DLP-USB245M. La velocidad de transferencia que puede alcanzar el sistema para el reconocimiento de imágenes está en función de la frecuencia a la que opera, con 50 MHz es capaz de transferir hasta 61.5 Megabaudios hacia el DLP-USB245M, por lo que la transferencia de información entre el sistema y la PC es limitado a la alcanzada por el DLP-USB245M que es de 8 Megabaudios, por lo tanto, esta interfaz cumple satisfactoriamente con los requerimientos del sistema.

Un sistema autónomo tiene recursos de memoria limitados, por lo tanto, éstos deben ser optimizados; además se pretende que este sistema opere en tiempo real, entonces la velocidad de procesamiento es un factor importante a considerar. La DWT demostró ser una herramienta que permitió reducir los requerimientos de memoria del sistema al utilizar únicamente la parte promedio de la DWT de nivel 4 de la imagen en el procesamiento de la información; además al hacer uso de una wavelet Haar la obtención de la parte promedio de la WDT se realiza en una forma rápida y eficiente debido a que usa sólo 4 sumas y un corrimiento en la obtención de un píxel del nivel siguiente, teniendo efectos mínimos sobre la velocidad de procesamiento del sistema.

El uso de las Memorias Asociativas Morfológicas en el proceso de reconocimiento del sistema para el reconocimiento de imágenes permite concluir que, primero, el sistema opere a altas velocidades de procesamiento debido a que las MAM no requieren converger y basan su funcionamiento en las operaciones morfológicas de dilatación y erosión (es decir hacen uso de máximos o mínimos de sumas); segundo, sea robusto al ruido inducido por etapas previas al reconocimiento desde que una MAM ha demostrado ser robusta a ruido presente en los patrones, sin importar si se trata de ruido sustractivo o aditivo; y tercero, tenga una gran capacidad de almacenamiento, característica implícita de las MAM.

El sistema para el reconocimiento de imágenes propuesto en este trabajo de tesis combina características como la flexibilidad para describir algoritmos que pueden operar en forma paralela, y un alto desempeño que otorga la implementación de un sistema en hardware con los bajos requerimientos de memoria, el uso de baja precisión aritmética, y de un reducido y simple

número de operaciones que una MAM requiere, aunado a las ventajas, ya mencionadas. Como conclusión general se puede mencionar que esta combinación genera un sistema robusto que puede ser utilizado en sistemas autónomos en tiempo real orientados a aplicaciones de visión artificial, y cuyas principales características son las siguientes: robusto a ruido inducido en la imagen por etapas previas, puede operar a altas velocidades de procesamiento y es fácilmente adaptable a diversas circunstancias de operación.

Los resultados experimentales obtenidos en las pruebas realizadas al sistema para el reconocimiento de imagenes mostraron que las MAM **M** tiene un mejor desempeño cuando las imágenes presentan ruido aditivo, mientras las MAM **W** lo presentan para imágenes con ruido sustractivo. En condiciones reales normalmente una imagen presenta ruido mixto, para estas condiciones las memorias que presentan un mejor desempeño en el reconocimiento son las MAM **M**. Con base en estos resultados, se concluye que las MAM **M** son las memorias idóneas para operar en el sistema para el reconocimiento de imágenes propuesto.

El diseño de la arquitectura VLSI para reconocimiento de imágenes se basó en el ciclo de vida de desarrollo de sistemas empotrados, la implementación del procesador para reconocimiento de imágenes en el FPGA siguió una metodología descendente y utilizó una descripción comportamental. El uso de dichas metodologías facilito el establecer, dar seguimiento y cumplir los requerimientos establecidos para el sistema, además permitió obtener un sistema de reconocimiento de imágenes robusto y de gran facilidad para realizar adecuaciones, mejoras o un rediseño parcial o completo del sistema.

Este trabajo de tesis forma parte de un proyecto más complejo enfocado a la visión artificial, con la finalidad de dar continuidad al sistema desarrollado, se proponen los siguientes trabajos de investigación:

- Hacer uso de memorias asociativas alternativas como las memorias αβ, las memorias AB, con la finalidad de comparar el desempeño de estas memorias con las utilizadas en este trabajo.
- Diseñar e implementar el modelado en hardware de un sistema de segmentación de imágenes, el cual sustituiría a la interfaz de alta velocidad con una PC empleada en este sistema.

Bibliografía

[1] ANTONINI, M.; et al. “Image Coding Using Wavelet Transform”. *IEEE Transactions on Image Processing*, 1992, Vol. 1, No. 2, pp. 205-220.

[2] ARMINGOL, J. M.; et al. “Visión por Computador para Vehículos Inteligentes”. Proyecto inédito del Grupo de Sistemas Inteligentes. Escuela Politécnica Superior de Madrid. 2003.

[3] BAHLMANN, C.; et al. “A system for traffic sign detection, tracking, and recognition using color, shape, and motion information”. *Proceedings in IEEE Intelligent Vehicles Symposium*, 2005, vol. 6, p. 255 – 260.

[4] BALL, Stuart R. *Embedded Microprocessor System Real World Desing*, Ed. Newnes, Second edition, pp 308.

[5] BARRÓN FERNÁNDEZ, Ricardo. “Memorias asociativas y redes neuronales morfológicas para la recuperación de patrones”. Tesis doctoral. Instituto Politécnico Nacional. Centro de investigaciones en computación. Febrero de 2005.

[6] BELLONE, T.; E. Borgogno; G. Comoglio. “Comparison of Parsing Techniques for The Syntactic Pattern Recognition of Simple Shapes”. *International Society for Photogrammetry and Remote Sensing (ISPRS).* July 2004, Istanbul, Turkey, Vol. 12, p. 683 ff.

[7] CASTELLANOS, C.; J. L. Díaz de León y A. Sánchez. "Análisis Experimental con las Memorias Asociativas Morfológicas”. XXI Congreso Internacional de Ingeniería Electrónica, Electro 99, Instituto Tecnológico de Chihuahua, Chih., México, Octubre 1999,pp. 11-16.

[8] CRUZ LÓPEZ, Maria Luisa. “Implementación y análisis comparativo de una red neuronal ART2 y una Backpropagation, aplicadas al reconocimiento de patrones en imágenes digitales”. Tesis de ingeniería. Universidad Tecnológica de la Mixteca. 2001.

[9] COLOM, Ricardo J.; Rafael Gadea. “Transformada Discreta Wavelet 2-D para Procesamiento de Video en tiempo real”. XII Jornadas de Paralelismo Valencia, septiembre 2001.

[10] DE LA ESCALERA, A.; J. M. Armigol; M. Mata. “Traffic Sign Recognition and Analysis for Intelligent Vehicles”. *Image and Vision Computing*, 2003, vol. 11, No. 3, p. 247-258.

[11] DE LA ESCALERA, A.; L. E. Moreno; M. A. Salichis; J. M. Armigol. “Road traffic sign detection and classification”. *IEEE on Industrial Electronics* , Dec 1997, Vol. 44, p. 848-859

[12] DE PABLO GÓMEZ, Santiago. “Arquitecturas y algoritmos para el tratamiento y la segmentación de imágenes en tiempo real”.Tesis doctoral. Universidad de Valladolid, Diciembre de 1995.

[13] DEVORE, R. A.; B. Jawerth; y B. J. Lucier. “Image Compression Through Wavelet Transform Coding”. *IEEE Transactions on Information Theory*, 1992, Vol 38, No. 2, pp 719-746.

[14] DÍAZ, G. S.; y J. R. Shulcloper, "Mid Mining: A Logical Combinatorial Pattern Recognition Approach to Clustering in Large Data Sets" *Memorias del V Simposio Iberoamericano de Reconocimiento de Patrones,* Lisbon, Portugal, September 2000 pp. 475-483.

[15] DUDA, Richard O.; Meter E. Hart; David G. Stork. *Pattern Classification*. Second Edition. A Wiley- Interscience Publication, United Sates of América, ISBN 0-471-05669-3, 2001.

[16] FARAG, Aly A.; Alaa E. Abdel-hakim. “Detection, categorization and recognition of road sign for autonomous navigation”. *Proceedings of Acivs (Advanced Concepts for Intelligent Vision Systems)*, 2004, No. 23.

[17] FUGAL, Lee D. “Conceptual Wavelets in Digital Signal Procesing”, Spaces & Signals Technologies, 2006.

[18] GODOY, Salvador C. “Evaluación de algoritmos de clasificación basada en el modelo estructural de cubrimientos”. Tesis doctoral. Instituto Politécnico Nacional, Centro de Investigaciones en Cómputo. Mayo de 2006.

[19] GONZALES, R. C.; y R. E. Woods. *Digital Image Processing*. Second Edition. Prentice-Hall, USA, ISBN 81-203-2758-6, 2002.

[20] GRIMBLANTT, Víctor. Sistemas embebidos *Semiconductor Products Sector.* Motorola Electro Industria, Abril 2002

[21] GROOVER, M. P., M. Weiss; R. N. Nagel; y N. G. Odrey. *Robótica Industrial: tecnología, programación y aplicaciones*. Mc. Graw hill, 1990.

[22] HOPFIELD, J. J. “Neural Networks and Physical Systems with Emergent Collective Computational Abilities”. *Proceedings of the National Academy of Sciences of the USA*, Vol. 79, pp. 2554-2558, 1982.

[23] HUCKVALE, Mark “A syntactic pattern recognition method for the automatic Location of potential enhancement regions in running speech”. UCL Working Papers Speech Hearing and Language, 1997.

[24] IEEE std 1076-1993. *IEEE Standard VHDL Language Referente Manual*. ANSI/IEEE Std 1076-1993, SH16840. June 1994.

[25] KOHONEN, T. *Self-Organizing Maps*. Third Edition. Springer-Verlag Berlin Heidelberg New York, ISSN 0720-678X, ISBN 3-540-67921-9, 2001.

[26] LEWIS, A. S.; and G. Knowles. "Image Compression Using the 2-D Wavelet Transform". *IEEE Transactions on Image Processing*, 1992, Vol. 1, No. 2, pp. 244-250.

[27] LORD, G. J.; E. Pardo-Igúzquiza; y I. M. Smith. "A Practical Guide to Wavelets for Metrology". National Physical Laboratory, Queens Road, Teddington, Middlesex, United Kingdom, June 2000, ISSN 1471–0005.

[28] MCCULLOCH, W. S.; W. H. Pitts. "A logical calculus of the ideas immanent in nervous activity". Bulletin of Mathematical Biophysics, 1943, 5:115-133.

[29] MORENO, Francisco S. "Clasificadores eficaces basados en algoritmos rápidos de búsqueda del vecino más cercano". Tesis doctoral. Universidad de Alicante, Febrero de 2004.

[30] PACLIK, P.; J. Duin Novovicova. "Building road sign classifiers using a trainable similarity measure". *Proceedings in IEEE Intelligent Transportation Systems,* 2006, Vol. 7, p. 309- 321.

[31] PARDO, Fernando y José A. Boluda. *VHDL lenguaje para síntesis y modelado de circuitos*. 2ª Edición. Editorial RA-MA, España, 2004. ISBN 84-7897-595-0.

[32] PATEL, Jamshed N.; Ashfaq A. Khokhar; y Leah H. Jamieson. "Scalability of 2-D Wavelet Transform Algorithms: Analytical and Experimental Results on MPPs". IEEE Transactions on Signal Processing. December 2000, Vol. 48, No. 12, pp. 3407-3419.

[33] PELLERIN, David and Douglas Taylor. *VHDL Made Easy!*. Prentice-Hall, USA, 1997. ISBN 0-13-650763-8.

[34] PÉREZ, S. A.; E. Soto; y S. Fernández. *Diseño de Sistemas Digitales con VHDL*. Ed. Thomson, Madrid España, 2002.

[35] RITTER G. X.; J. L. Díaz de León; y P. Sussner. "Morphological Bidirectional Associative Memories". *Neural Networks,*1999, Vol. 12, No. 6, pp. 851-867.

[36] RITTER, G. X.; P. Sussner; and J. L. Díaz de León. "Morphological Associative Memories". *IEEE Transactions on Neural Networks*, 1998,Vol. 9, No. 2, pp. 281-293.

[37] RITTER, G. X.; P. Sussner. "An Introductions to Morphological Neural Networks". Proceedings of the 13th International Conference on Pattern Recognition, 1996, Vol. IV, Track D, pp. 709-717.

[38] RIOUL, O.; y M. Vetterl. "Wavelet and Signal Processing". IEEE SP Magazine, The Institute of Electrical and Electronics Engineers, Inc. October 1991. 1053-5088/91/1 00000 14S1 .00IEEE. pp. 14-38.

[39] RODRIGUEZ DIEZ, Héctor; et al. Sistema para la Clasificación basado en Técnicas de Reconocimiento de Patrones. Cuba: Santiago de Cuba. Universidad de Oriente. 2000.

[40] THEODORIDIS, Sergios; Konstantinos Koutroumbas. *Pattern Recognition.* Second Edition. Elsevier. Greece. 2003.

[41] VENGUEROV, Mark. “Generalised Syntactic Pattern Recognition as a Unifying Approach in Image Analysis”. 2nd International workshop on Statistical Techniques in Pattern Recognition, Sydney, Australia. August 1998.

[42] VERRI, A.; y E. Trucco. *Introductory Techniques for 3-D Computer Vision*, Prentice Hall, 1998.

[43] VILLASEÑOR, Elio Atenógenes G. “Análisis inteligente de datos con redes neuronales artificiales”.Tesis de licenciatura inédita. Universidad Nacional Autónoma de México (UNAM), Agosto de 2004

[44] YANKOVSKAYA, A. E. “Logical-Combinatorial-Probabilistic Recognition Algorithms”. *Pattern Recognition and Image Analysis*, 2001, Vol. 11, No. 1, pp. 123–126.

[45] YAÑEZ, C. “Memorias Asociativas αβ”. Tesis Doctoral. Instituto Politecnico Nacional, Centro de Investigación en Computación. 2002.

[46] YÁÑEZ, C.; y J. L. Díaz de León. “Lernmatrix de Steinbuch”. Centro de Investigación en Computación, IPN, México, No. 48, Serie Verde, ISBN, 2001.

[47] YÁÑEZ, C.; y J. L. Díaz de León. “Linear Associator de Anderson-Kohonen”. Centro de Investigación en Computación, IPN, México, No. 50, Serie Verde, ISBN, 2001.

[48] YÁÑEZ, C.; y J. L. Díaz de León. “Algunas Aportaciones de S. Amari en Memorias Asociativas”. Centro de Investigación en Computación, IPN, México, No. 51, Serie Verde, ISBN, 2001.

[49] YÁÑEZ, C.; y J. L. Díaz de León. “Memoria Asociativa Hopfield”. Centro de Investigación en Computación, IPN, México, No. 52, Serie Verde, ISBN, 2001.

[50] YÁÑEZ, C.; y J. L. Díaz de León. “Modelo ADAM”. Centro de Investigación en Computación, IPN, México, No. 54, Serie Verde, ISBN, 2001.

[51] YÁÑEZ, C.; y J. L. Díaz de León. “Modelo BAM”. Centro de Investigación en Computación, IPN, México, No. 55, Serie Verde, ISBN, 2001.

[52] YÁÑEZ, C.; y J. L. Díaz de León. “Modelo SMD”. Centro de Investigación en Computación, IPN, México, No. 56, Serie Verde, ISBN, 2001.

[53] YÁÑEZ, C.; y J. L. Díaz de León. “Memorias Morfológicas Heteroasociativas”. Centro de Investigación en Computación, IPN, México, IT 57, Serie Verde, ISBN 970-18-6697-5, 2001.

[54] YÁÑEZ, C.; y J. L. Díaz de León. “Memorias Morfológicas Autoasociativas”. Centro de Investigación en Computación, IPN, México, IT 58, Serie Verde, ISBN 970-18-6698-3, 2001. (a)

[55] YÁÑEZ, C.; y J. L. Díaz de León. “Introducción a las Memorias Asociativas”. Instituto Politécnico Nacional, Centro de Investigación en Computación. ISBN 970-36-0116-2. México 2003.

Referencias de recursos electrónicos

[URL1] ACOSTA, Maria Isabel; Salazar Harold; Camilo Zuluaga M. *Tutorial de redes neuronales.* [en línea] Colombia: Universidad de Pereira, 2000. <http://ohm.utp.edu.co/neuronales/main2.htm> [Consulta:13 de septiembre de 2006].

[URL2] ALONSO ROMERO, Luis; Teodoro Calonge Cano, Aplicaciones e Implementaciones de las Redes Neuronales en Reconocimiento de Patrones (AIRENE). [en línea] Versión preliminar del libro AIRENE. Proyecto CYTED VII.15, 30 de junio de 1999. <http://lisisu02.fis.usal.es/~airene/airene.html> [Consulta: 28 de septiembre de 2006]

[URL3] CARRASCO OCHOA, Jesús Ariel; Martínez Trinidad, José Francisco. Reconocimiento de Patrones. [en línea] Puebla, México: Instituto Nacional de Astrofísica Óptica y Electrónica. <http://ccc.inaoep.mx/~ariel/> [Consulta: 13 de septiembre de 2006].

[URL4] CORTIJO BON, Francisco José. "Introducción al reconocimiento de formas" y "Técnicas supervisadas II: Aproximación no paramétrica al Reconocimiento de Formas." En: Reconocimiento de Formas [en línea] Granada, España: Universidad de Granada, 30 de octubre de 2001. <http://www-etsi2.ugr.es/depar/ccia/rf/material.html> [Consulta: 14 de septiembre de 2006].

[URL5] DAPENA, Adriana. Técnicas de procesado de imagen [en línea] Coruña, España. <http://www.des.udc.es/~adriana/TercerCiclo/CursoImagen/curso/web/Indice.html > [Consulta: 13 de septiembre de 2006]

[URL6] Grupo de Ingeniería de Medios. *El Reconocimiento de Patrones y Análisis de Imagen* [en línea] Cáceres: Escuela Politécnica de Cáceres, 1 de mayo de 2006. <http://gim.unex.es/gim/index.php?content=imagendigital&lang=es >[Consulta: 17 de junio de 2006]

[URL7] GÓMEZ MARTÍNEZ, Mario. *Redes Neuronales Artificiales.* [en línea] Barcelona, España: IIIA – Instituto de Investigación en Inteligencia Artificial y CSIC - Spanish Scientific Research Council, marzo de 2004. <http://www.iiia.csic.es/~mario/rna/tutorial/RNA_intro.html#Introducción> [Consulta: 13 de septiembre de 2006].

[URL8] GUERRA HERNÁNDEZ, Alejandro. "Árboles de decisión" En: *Aprendizaje automático* [en línea] Veracruz, México: Universidad Veracruzana, 12 de mayo de 2004. <http://www.uv.mx/aguerra/teaching/MIA/MachineLearning/> [Consulta: 18 de octubre de 2006].

[URL9] GUTIÉRREZ CÁCERES, Juan Carlos. *Inteligencia Artificial.* [en línea] Brasil: Universidad de Sao Paolo Brasil, 2003. <http://www.usp.edu.pe/~jc.gutierrez/> [Consulta: 20 de agosto de 2006]

[URL10] HERNÁNDEZ MORA, José Juan. "Introducción a la robótica" En: *Tutorial de Técnicas Modernas I.* [en línea] Tlaxcala, México: Instituto Tecnológico de Apizaco, 22 de julio de 2004. <http://www.itapizaco.edu.mx/paginas/ttm/PAG1.html>, [Consulta: 28 de septiembre de 2006]

[URL11] MARTÍ, Enric. “Análisis de Elementos Gráficos en Documentos”. [el línea] *Revista electrónica de visión por computador*. Octubre 1999-0. <http://revc.uab.es/revista/000/ > [19 de octubre de 2006]

[URL12] MATICH, Damián Jorge. “Redes neuronales: conceptos básicos y aplicaciones”. En: *Informática aplicada a la ingeniería de procesos I.* [en línea] Universidad Tecnológica Nacional, Facultad Regional Rosario. Marzo de 2001. <http://www.modeladoeningenieria.edu.ar/> [Consulta: 29 de septiembre de 2006]

[URL13] *Robótica.* [en línea]: *contenidos por Programa Informática Aplicada al Trabajo Social* Murcia, España: Universidad de Murcia , 18 /09/05. <http://www.mundopc.net/actual/tecnologia/inforaplicada/3.php>[Consulta: septiembre de 2006]

[URL14] ROLÓN GARRIDO, Julio C. **“**Algoritmos de Reconocimiento de Patrones para Visión Robótica”. En: *Proyectos de investigación institucionales autorizados CGPI - IPN 2004-2005*. [abstract] [en línea] México: Centro de Investigación y Desarrollo de Tecnología Digital, IPN, 13 de Diciembre de 2004. <http://www.citedi.mx/docs/proy_ipncgpiinv.htm#3> [Consulta: 10 de junio de 2006].

Apéndice A. Características principales de la tarjeta de desarrollo XSV300.

El apéndice contiene las características principales de la tarjeta XSV300 Board V1.0 de la compaña Xess, la figura A.1 muestra su diagrama a bloques, entre los cuales se incluyen las siguientes:

- **FPGA Xilinx XCV300 Virtex***:* el FPGA Virtex empaquetado en 240 pines, cuenta con 300 mil compuertas y es el principal almacén de lógica programable en la tarjeta.
- **CPLD Xilinx XC95108:** es usado para manejar la configuración del FPGA vía el puerto paralelo, puerto serie o Flash RAM. También controla la configuración del Ethernet PHY.
- **Oscilador (Reloj) programable de 100 MHz:** el oscilador programable Dallas DS1075 que genera una señal de reloj para el FPGA y para el CPLD tiene una frecuencia máxima de 100 MHz, esta es divida para generar frecuencias de 100 MHz, 50 MHz, 33.3 MHz hasta 48.7 KHz. Para fijar el valor del divisor, el DS1075 debe ser puesto en modo de programación, esto se hace poniendo la salida de reloj a Vcc al encender con un puente a través de los pines 1 y 2 del jumper J22. El divisor es guardado en una EEPROM en el DS1075 y será restaurado siempre que se alimente a la tarjeta XSV. El puente sobre el jumper J22 debe ser a través de los pines 2 y 3 para hacer la salida del oscilador una señal de reloj.
- **RAM flash de 16 Mbits:** el FPGA y el CPLD tienen acceso a la memoria flash RAM de Intel 28F016S5 con 16 Mbits de almacenamiento. Después de alimentar la tarjeta, el FPGA puede leer y/o escribir a la flash siempre y cuando el CPLD no intente acceder también a la flash. La flash puede deshabilitarse poniendo el pin CE a Vcc en tal caso las líneas de entrada/salida conectadas a la flash pueden ser usadas para la comunicación entre el CPLD y el FPGA.
- **SRAM:** el FPGA tiene acceso a dos bancos independientes de SRAM, cada banco de SRAM esta formado por 2 SRAMs AS7C4096 y organizado como 512K x 16 bytes.
- **Decodificador de video:** la tarjeta XSV puede digitalizar señales de video SECAM, PAL y NTSC usando el decodificador de video SAA7113. El video digitalizado llega al FPGA por el bus VPO. La llegada de datos de video está sincronizada con el flanco de subida del LLC (line locked clock) del decodificador de video. El FPGA programa las opciones de video del SAA7113 usando el bus I^2C (SCL y SDA).

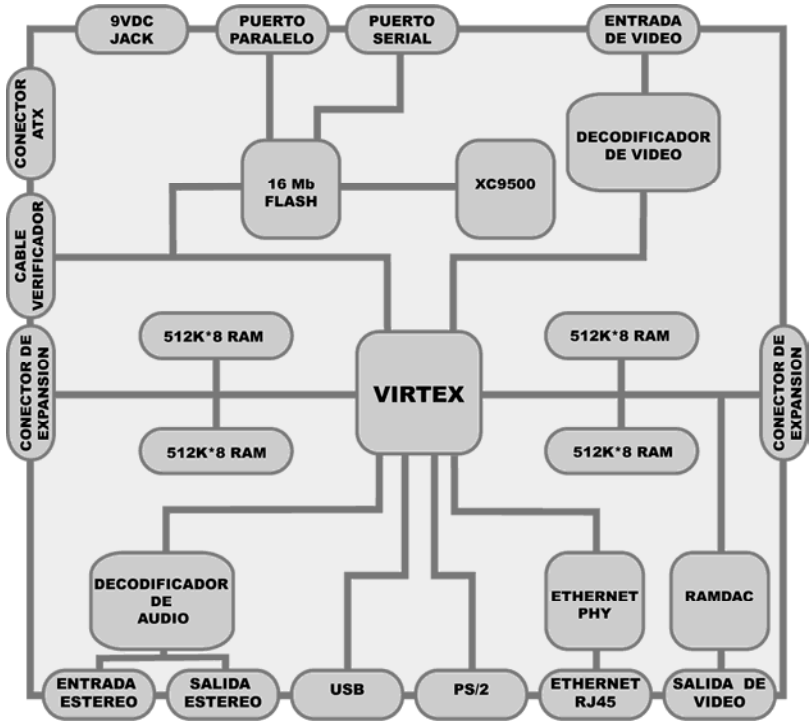

Figura A.1. Diagrama a bloques de la tarjeta de desarrollo XSV300.

- **RAMDAC y puerto VGA:** el FPGA puede generar una señal de video para desplegarla directamente en un monitor VGA o puede usar una RAMDAC BT481A dependiendo del arreglo de los jumpers J5, J6 y J7. Cuando el FPGA esta generando directamente las señales de VGA, la información del color es enviada mediante un DAC formado por resistores en escalera junto con las señales de sincronía horizontal y vertical (/HSYNC, /VSYNC).
- **Codec de Stereo:** la tarjeta XSVtiene un codec de stereo AK4520A que acepta 2 canales de entrada analógicas del jack J1, digitaliza los valores analógicos y envía los valores digitales en forma serial al FPGA. El codec también acepta bits en forma serial de la tarjeta XSV y las convierte en 2 señales de salida analógicas que salen de la tarjeta a través del jack J2. Los bits seriales son sincronizados con el reloj del FPGA, el cual entra al codec por la señal SCLK. El FPGA usa la señal LRCK para seleccionar el canal izquierdo o derecho como la fuente/destino de los datos seriales. El reloj principal del FPGA MCLK sincroniza todas las operaciones internas del codec.
- **Ethernet PHY:** el chip Ethernet PHY LXT970A está conectado al FPGA y al CPLD. El FPGA actúa como MAC (media access controller) y dirige la transferencia de paquetes de datos hacia y desde el chip PHY cuando el CPLD controla la configuración de pines que determina el modo de operación de el chip PHY. El FPGA habilita el transmisor TX_EN y envía los bits por el bus TXD en sincronización con el reloj de transmisión TX_CLK generada por el chip PHY. El FPGA recibe también una indicación de que el dato ha sido recibido RX_DV y el dato RXD está sincronizado con el reloj del receptor RX_CLK del chip PHY. Cualquier error en la recepción se indica al FPGA por la señal RX_ER. El FPGA puede deshabilitar la interfaz con el chip PHY poniendo el control en tercer estado. El FPGA pasa información a la PHY y de la PHY sobre la línea de datos seriales en sincronía con un reloj. El CPLD pone valores estáticos sobre los pines que controlan la configuración de la PHY. Pines MF0 a MF4,

fija el modo de auto-negociación, repetición, transmisión de símbolos, etc. De la misma manera, las señales de configuración CFG0-1 seleccionan la velocidad de operación del chip PHY, 10 Mbps o 100 Mbps. FDE selecciona también el modo de comunicación half-duplex o full-duplex.

- **Puerto PS/2:** la tarjeta XSV provee una interfaz PS/2 para un teclado o Mouse. EL FPGA recibe dos señales de la interfaz PS/2: una señal de reloj y una trama de datos seriales que están sincronizados con los flancos de caída de la señal de reloj.
- **Puerto USB:** la tarjeta XSV tiene un controlador de USB que puede estar conectada a una variedad de periféricos USB de alta o baja velocidad. El FPGA interfasa las dos señales de datos diferenciales del puerto USB a través de un chip de interfaz de USB (PDIUSBP11A). El puerto USB es puesto en alta o baja velocidad 12 Mbps y 1.5 Mbps respectivamente desviando los jumpers J18 y J37.Una carga de 15 KΩ puede ser puesta sobre las señales USB D^- y D+ poniendo los jumpers J33 y J34. Si le periférico USB conectado a el puerto necesita alimentación de la tarjeta XSV entonces se debe mover el jumper J16.
- **Puerto paralelo:** el CPLD maneja la interfaz del puerto paralelo. Las 17 líneas del puerto paralelo se conectan a pines de entrada/salida de propósito general en el CPLD. Cuatro líneas del puerto paralelo están conectadas a los pines JTAG a través de los cuales el CPLD es programado. La señal de TCK cronometra los datos de entrada a través del pin TDI mientras la señal de TMS dirige las acciones de la programación de la máquina de estados,
- **Conectores de expansión:** la tarjeta XSV tiene 2 expansiones de 50 pines que conectan el FPGA con sistemas externos. Los pines del FPGA con los conectores de expansión izquierdo y derecho también están conectados al banco izquierdo y derecho de la SRAM respectivamente. El chip-enable de la SRAM puede ser usado para deshabilitar la SRAMs de ese lado si el conector de expansión está siendo usado como entrada o salida externa.
- **Pushbuttons y DIP switch de 8 posiciones:** la tarjeta XSV tiene un banco de 8 DIP switches y 4 pushbuttons que son accesibles desde el FPGA. El CPLD también está conectado a los DIP switches y a uno de los pushbuttons. Cuando se presiona cada pushbutton pone el pin conectado al FPGA y al CPLD a tierra. De la misma manera el DIP switch pone el pin conectado al FPGA y al CPLD a tierra cuando está cerrado o encendido. Cuando el DIP switch esta abierto o apagado, el pin es puesto en alto a través de una resistencia.
- **Dígitos y leds bargraph:** la tarjeta XSV tiene un led bargraph de 10 segmentos y dos más de siete segmentos para uso del FPGA y del CPLD. Todos estos leds son activos en alto.

Apéndice B. Conjunto de patrones utilizados, en tamaño real.

Figura B.1. Imagen real del patrón número 1.

Figura B.2. Imagen real del patrón número 2.

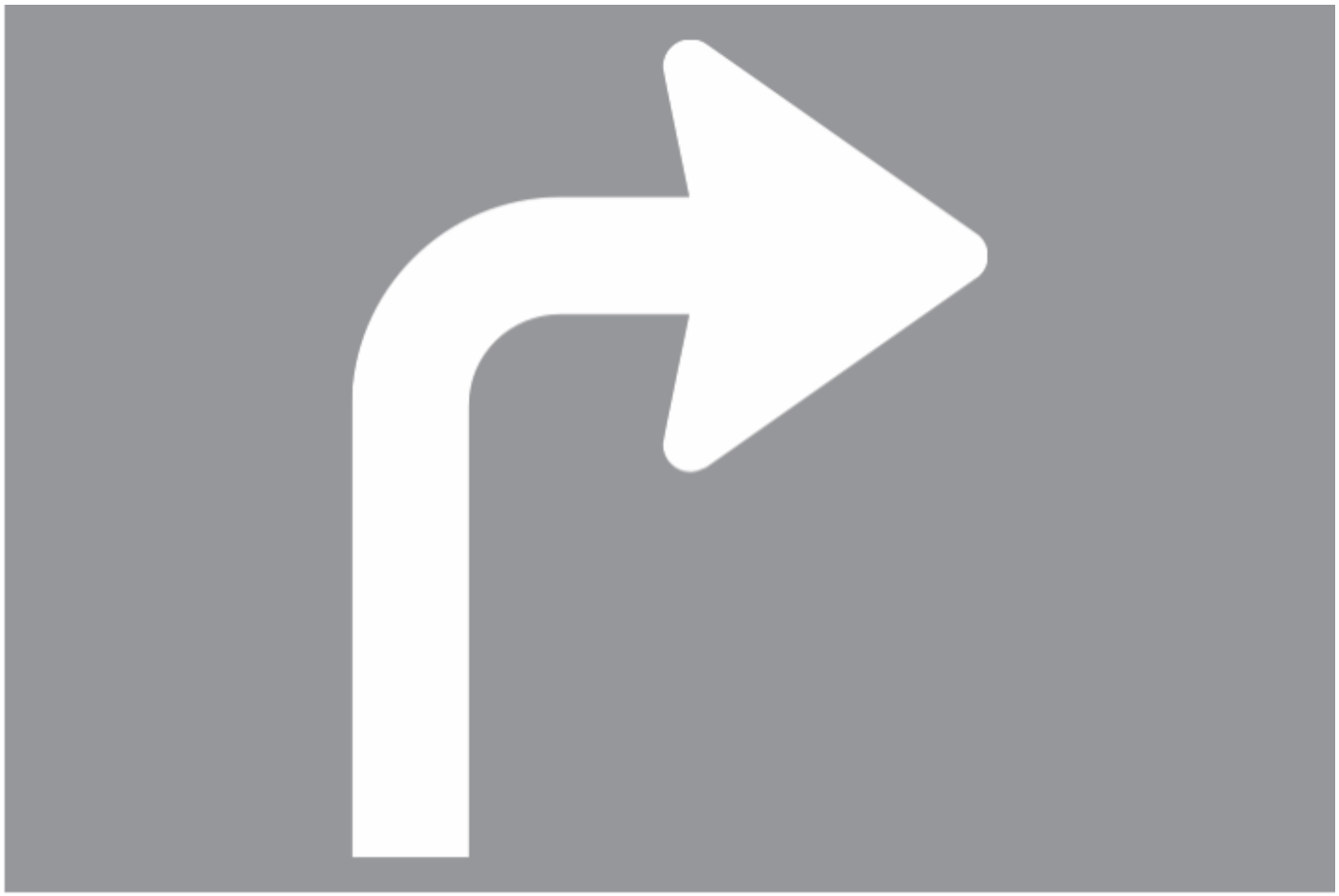

Figura B.3. Imagen real del patrón número 3.

Figura B.4. Imagen real del patrón número 4.

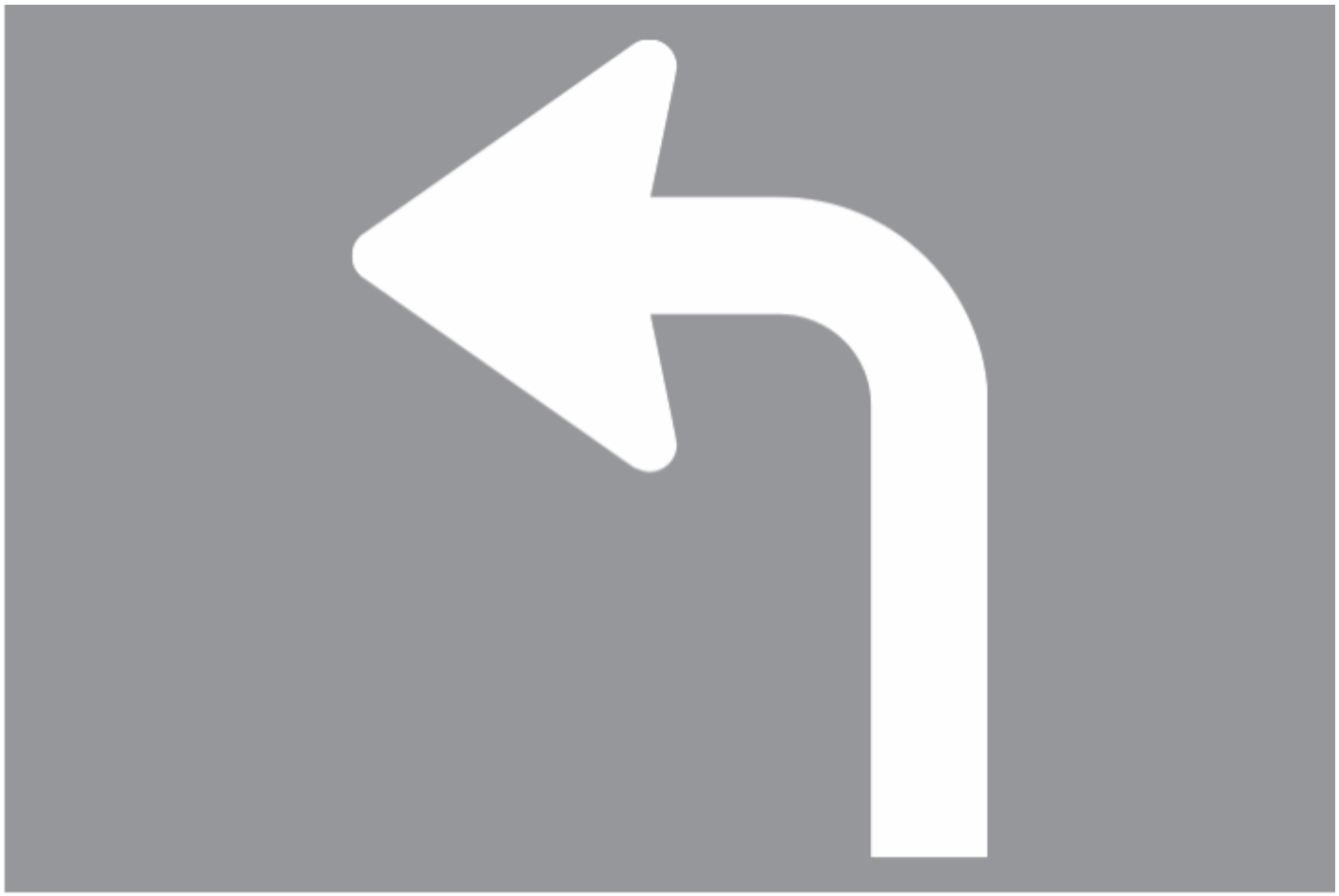

Figura B.5. Imagen real del patrón número 5.

Apéndice C. Esquemático general del procesador para el reconocimiento de imágenes.

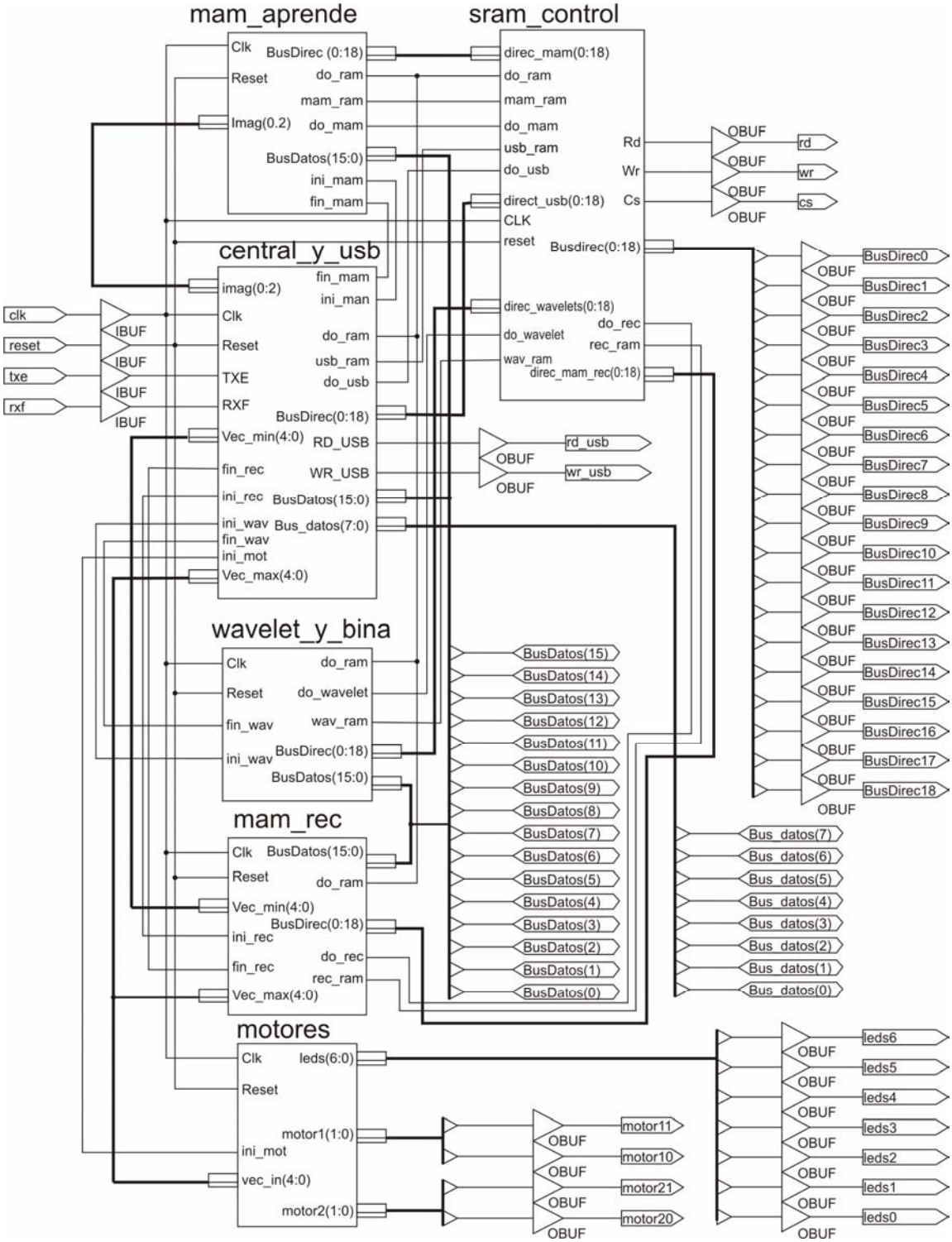

Figura C.1. Esquema general de la implementación del procesador para reconocimiento de imágenes en el FPGA XCV300.