UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

"DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA PARA LA MANIPULACIÓN DE VIDEO EN TIEMPO REAL BASADA EN UN FPLD"

TESIS
PARA OBTENER EL TÍTULO DE
INGENIERO EN ELECTRÓNICA

PRESENTA
JUAN DIEGO BARRANCO CRUZ

DIRECTOR DE TESIS
DR. ENRIQUE GUZMÁN RAMÍREZ

HUAJUAPAN DE LEÓN, OAX.; AGOSTO DE 2008

Tesis presentada el 14 de Agosto de 2008, ante los sinodales:

M.C. Felipe Santiago Espinoza
Dr. Felipe de Jesús Rivera López
Dr. Antonio Orantes Molina

Director de Tesis:

Dr. Enrique Guzmán Ramírez

Agradecimientos

A la Universidad Tecnológica de la Mixteca por permitirme utilizar sus instalaciones para la realización de esta tesis.

Al Dr. Enrique Guzmán Ramírez por su paciencia, ayuda y consejos.

A los sinodales asignados por sus observaciones.

Dedicatoria

A ti jefa que herencia más valiosa, no se como agradecerte.

A ti Gris gracias por tu ayuda incondicional y tu amor.

Drew eres mi inspiración.

Resumen

El presente trabajo de tesis presenta el diseño e implementación de un circuito digitalizador de video enfocado a aplicaciones de procesamiento y análisis digital de imágenes en sistemas autónomos. El sistema de video utilizado en este proyecto es el proporcionado por el comité de sistemas de televisión nacional (NTSC, *National Television System Comite*) con las normas RS-170 o RS-170A, monocromático y de color respectivamente, este sistema se cumple en la mayoría de las fuentes que proporcionan señales de video compuestas.

El elemento principal del circuito digitalizador de video propuesto es un Dispositivo Lógico Programable en Campo (FPLD, *Field Programmable Logic Device*), específicamente un Arreglo de Compuertas Programables en Campo (FPGA, *Field Programmable Gate Array*), el modelado del procesador de video se implementa con el lenguaje descriptor de hardware para circuitos integrados de muy alta velocidad (VHDL, *Very High Speed Integrated Circuit Hardware Description Language*) utilizando la herramientas para la Automatización del Diseño Electrónico (EDA, *Electronic Design Automation*) Xilinx Foundation versión 4.1i de la compañía Xilinx.

Con la finalidad de obtener como resultado una descripción portable entre diversas tecnologías, el modelado del digitalizador de video propuesto no considera ninguna arquitectura específica. Para probar dicho modelado, la implementación del diseño fue hecha en dos FPGAs distintos, un dispositivo XC4010XL-EPC84 de la familia XC4000 y un dispositivo XC2S200 PQ208 de la familia Spartan 2 ambos de la compañía Xilinx.

El circuito digitalizador de video se integra por elementos que permiten el acoplo, sincronización y discretización de la señal de video, así como un sistema de memoria que permite almacenar el marco digitalizado para su posterior procesamiento. Además, el digitalizador de video cuenta con una interfaz a una PC, cuyo único propósito es evaluar los resultados obtenidos.

Índice General

Capítulo 2. Dispositivos lógicos programables en campo y VHDL.

Capítulo 3. Desarrollo del digitalizador de video.

Índice de figuras

Capítulo 1

Capítulo 2

Capítulo 3

Capítulo 4.

Apéndice A

Índice de tablas

INTRODUCCIÓN

Un digitalizador de video se conoce en el mercado como *frame grabber*, es capaz de digitalizar, almacenar y preprocesar video en tiempo real, proporcionando disponibilidad para aplicaciones como visión artificial en móviles, seguimiento de objetos, interpretación de imágenes, etc., estas aplicaciones necesitan de resultados en tiempos cortos, sin embargo también tienen aplicaciones en medicina, una área que muchas veces dispone de tiempos largos dedicados a interpretar resultados con el fin de obtener los más acertados.

Este proyecto de tesis presenta un sistema de adquisición de marcos de video de cualquier fuente que cumpla con la norma RS-170A, EIA170 o comúnmente conocido como señales de video compuestas, éstas actualmente se pueden obtener de conexiones RCA y de conectores de súpervideo.

Se propone utilizar un FPLD, específicamente un FPGA, como elemento central del digitalizador de video. Este dispositivo incluirá las descripciones de las funciones principales del sistema, sincronía, control, adquisición y almacenamiento de la información de video. Además de poder modelar algoritmos de tratamiento de la señal adquirida.

Para la descripción del digitalizador de video, se usará el lenguaje VHDL y con la finalidad de volver este diseño portable no se considerará la arquitectura del FPGA en el que se implementará.

Marco Teórico

Un digitalizador de video, también conocido como sujetador de marco (*frame grabber),* se emplea para capturar marcos de video de uno o varios sistemas, como NTSC, PAL o SECAM, que se pueden obtener de múltiples fuentes de video. Los digitalizadores de video comerciales varían en su composición y funciones, la mayoría son diseñados para integrarse a una computadora personal (PC, *Personal Computer*), sin embargo en ocasiones es deseable contar con un sistema de este tipo, que además de ser autónomo, tenga un interfaz con una PC que permita monitorear su desempeño.

Para la realización de este digitalizador de video, es necesario conocer las bases teóricas del sistema seleccionado y la tecnología del FPLD a utilizar así como la metodología que rige un diseño digital.

Sistema NTSC

En un sistema NTSC de video entrelazado 2:1, una imagen de video es generada por dos barridos de un haz de electrones a través de la superficie de un dispositivo TRC. Cada barrido es considerado como un campo y dos campos comprenden un marco de video. Un campo completo de video es desplegado antes de que el otro campo de video esté iniciando. Esencialmente, el campo uno pone las líneas horizontales de video numeradas como impar y va dejando un espacio para situar las líneas del campo dos [5], [8], [9].

La figura I.1 muestra el diagrama de coordinación para el RS-170A de señales de video a color. Esta figura muestra la relación entre la información de control e imagen en una señal de video, principalmente ilustra las señales que contienen la sincronización vertical. Todas las coordinaciones, también conocidas como sincronización, se representan a través de pulsos síncronos cortos, los cuales se limitan desde el nivel de blancura hasta el nivel síncrono. Las señales de video compuesta contienen componentes analógico y digital, el componente digital está hecho de varios pulsos síncronos, en tanto que el componente analógico contiene la información de la imagen.

El tipo, ancho y cantidad de pulsos síncronos son muy importantes cuando se intenta extraer información de la imagen de una señal de video. En la figura I.1 se identifican tres clases de pulsos [5], [9]:

- El *pulso síncrono,* tiene un ancho de aproximadamente 4.7us.
- El *pulso vertical*, tiene un ancho de aproximadamente 27.1us.
- El *pulso de ecualización*, tiene un ancho de 2.3 microsegundos. Estos pulsos son usados para mantener el nivel de CD de la señal de video.

Si el número total de pulsos (síncronos, verticales y ecualizados) son contabilizados en ambos campos de un marco de la norma RS-170A, resulta que el campo uno contiene 272 pulsos mientras que el campo dos está integrado de 271 pulsos.

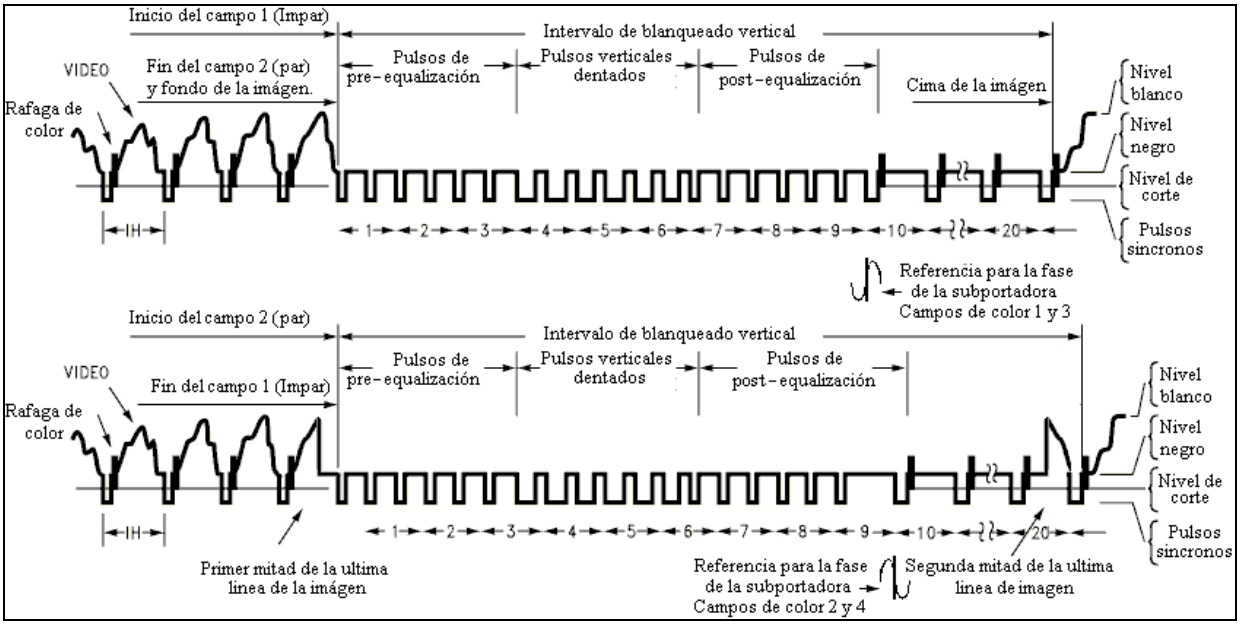

Figura I.1. Norma RS-170A.

El sistema NTSC de video consiste de 525 líneas horizontales de información de video, actualizadas 30 veces por segundo. Para que se realice este despliegue, cada campo requiere la mitad de ese tiempo, es decir, 1/60 de segundo. Debido a que las 525 líneas son divididas igualmente entre los dos campos, cada campo está comprendido de 262.5 líneas, la parte fraccional de este número es la razón de que ambos campos de video contengan la mitad de una línea de despliegue.

Desplegar 262.5 líneas de video en 1/60 de segundo significa que cada línea requiere aproximadamente 63.5µs. Este número es un parámetro importante para el diseño de un digitalizador de video, debido a que es la longitud del tiempo entre los píxeles posicionados verticalmente en un despliegue NTSC [5], [9].

No todas las 262.5 líneas de información de video en un campo son usadas para desplegar imágenes, aproximadamente 18.5 líneas son utilizadas para el blanqueado vertical y la sincronización del despliegue. Esto deja 244 líneas por campo para información de la imagen. Estas 244 líneas por campo o 488 líneas por marco, forman la imagen desplegada sobre monitores de computadoras o televisores (TV). De las líneas utilizadas para el blanqueado vertical, 17 líneas pueden además ser usadas para otros propósitos. Las aplicaciones de estas líneas de video oculto incluyen evaluación, diagnóstico, indicación y transferencia de señales de control. La información útil que no es relacionada al video puede ser transferida en éstas, esencialmente líneas de video sin uso. Esta información puede ser decodificada con un receptor configurado apropiadamente sin afectar el despliegue del video [5], [9].

Planteamiento del problema

Los digitalizadores de video, también conocidos como sujetadores de marco, se fabrican en muchas formas y tamaños con cantidades variantes de funciones, una característica que tienen en común estos sistemas comerciales es el ser un módulo que debe ser conectadas a una PC mediante diversos estándares, tales como el bus de interconexión de componentes periféricos (PCI, *Peripheral Component Interconnect*), el puerto Bus serial universal (USB, *Universal Serial Bus*), PS/2, puerto serie, puerto paralelo, entre otros. Además, la mayoría de las compañías en el mercado ofrecen digitalizadores de video con precios generalmente altos.

Las características mencionadas dificultan el uso de los digitalizadores de video comerciales en aplicaciones donde se ve involucrado un sistema autónomo.

Justificación.

El presente proyecto de tesis se enfoca en el diseño e implementación de un digitalizador de video, para el sistema de video NTSC.

Un digitalizador de video comercial generalmente está destinado a ser empleado en sistemas de adquisición de imagen en una PC, alcanzando precios elevados dependiendo de sus características y del tamaño de imagen que puedan digitalizar, debido a esto y con la finalidad de implementar un sistema que se pueda emplearse en forma autónoma para la digitalización y preprocesamiento de imágenes, en este trabajo de tesis se diseña e implementa un digitalizador de video utilizando un CDC, específicamente un FPGA, que permita la captura y almacenamiento de cuadros de video para su posterior procesamiento y/o análisis, contando además con una interfaz a una PC, cuyo único propósito es evaluar los resultados generados por el sistema.

Este sistema es parte de un sistema jerárquicamente superior cuyo propósito es brindar un sistema autónomo que permita realizar procesamiento de imágenes enfocadas a visión artificial.

Objetivos.

Objetivo general. Diseñar e implementar un sistema capaz de adquirir, almacenar y preprocesar señales de video del sistema NTSC empleando la norma RS-170A que sirva como plataforma para desarrollo de sistemas de video digitalizado.

Objetivos particulares.

- Digitalizar imágenes en tiempo real empleando una velocidad menor a 17ms para la captura de un marco de video.
- Implementar una interfaz con un PC mediante el protocolo RS-232, la cual fungirá como herramienta de evaluación de la adquisición y tratamiento de la imagen.

Metodología de diseño.

El digitalizador de video implementado en este proyecto de tesis cuenta con funciones para digitalizar, almacenar y procesar la información de video. Para el desarrollo en este proyecto de tesis se ha tomado como base la metodología de un sistema empotrado, esta metodología ha sido ajustada mediante algunos cambios, al desarrollo del sistema propuesto. La metodología debe cumplir con las siguientes fases:

Fase 1. Definición de las especificaciones. Limita las capacidades del digitalizador de video en cuanto al sistema de video soportado, tamaño de imagen, resolución, etc.

Fase 2. Elección de los componentes utilizados en el sistema propuesto. Aquí se detallan brevemente los CI's más importantes empleados en el digitalizador de video.

Fase 3. Diseño del sistema. El digitalizador de video está integrado principalmente por elementos electrónicos analógicos, digitales y algunos que manejan ambas partes, las subetapas listadas a continuación muestran una forma de clasificación particular y de cómo ha sido dividido el diseño para facilitar su implementación:

Fase 3.1. Procesador de video.

Fase 3.2. Módulo analógico.

Fase 3.3. Módulo digital.

Fase 4. Modelado del procesador de video. El procesador de video implementado en un FPGA de la familia de Xilinx, mediante lenguaje VHDL, realiza la digitalización y almacenamiento de las señales de video, la coordinación y control de las señales de sincronía y la transferencia de imágenes hacia la PC. Los procesos o módulos que complementan esta fase son:

Fase 4.1. Módulo de sincronía.

Fase 4.2. Módulo de control.

Fase 4.3. Módulo de almacenamiento.

Fase 4.4. Módulo para el protocolo RS-232.

Fase 5. Interfaz de usuario. Para verificar el correcto funcionamiento del digitalizador de video, se implementa un programa para recepción de imágenes vía el puerto serie de la PC, este programa permite verificar la digitalización y procesamiento de imágenes.

Fase 6. Integración del digitalizador de video. En esta fase las partes analógica y digital son acopladas para formar al digitalizador de video, el sistema resultante se complementa con el dispositivo optoelectrónico que proporcione la señal de video RS-170 (generalmente una cámara CCD), para adquirir las imágenes enfocadas, digitalizarlas y almacenarlas en el sistema de memoria.

Fase 7. Pruebas de funcionamiento. Finalmente empleando el digitalizador de video se verifican los resultados de las imágenes adquiridas mediante la interfaz con la PC.

Un diagrama de bloques del digitalizador de video se ilustra en la figura I.2 y se compone de los siguientes elementos:

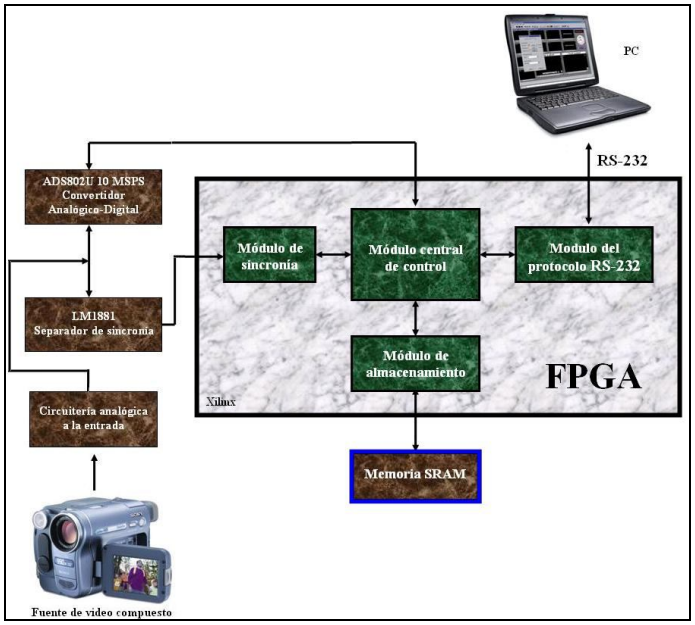

Figura I.2. Digitalizador de video

Procesador de video. Es la parte central de este trabajo de tesis, ya que en ésta se realiza el modelado del algoritmo encargado de controlar todas las funciones a realizar por el digitalizador de video. El modelado o descripción de cada uno de los módulos que intervienen en el procesador de video son hechos con el lenguaje descriptor de hardware VHDL utilizando la herramienta para la automatización del diseño electrónico (EDA, *Electronic Design Automation*), ISE Foundation versión 6.2i, e implementado en un FPGA XC4010XL y en un XC2S200 de la familia Spartan 2, ambos de la compañía Xilinx.

Para el modelado del procesador de video se utilizo la metodología descendente (*Top-Down*), 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 [3].

El procesador de video incluye varios módulos encargados del control de cada uno de los periféricos que forman el digitalizador de video, estos módulos son los siguientes:

- **Módulo SRAM**. En este modulo se implementa la coordinación de la memoria SRAM para almacenar temporalmente imágenes de hasta 712x480 píxeles, la memoria empleada tiene una capacidad de 512Kbytes.
- **Módulo RS232**. Corresponde a la implementación del protocolo de comunicación RS-232, está dividido en dos módulos correspondientes a la transmisión y a la recepción respectivamente, puede implementar todas las características asociadas al protocolo, sin embargo se emplean las básicas y más comunes como son, rango de baudios, tamaño de bits de los datos transmitidos, velocidad de transmisión y bits de parada e inicio.
- **Módulo de la señal de reloj del ADC.** Proporciona la señal de reloj que el ADC necesita para la velocidad de conversión de datos. El dispositivo empleado es un ADS802U proporcionado por la compañía Texas Instruments y alcanza una velocidad de muestreo de hasta 10Mbps.
- **Módulo de sincronía**. Manipula las señales de coordinación y control extraídas de la señal de video compuesta mediante la circuiteria analógica, con la finalidad de digitalizar y almacenar la parte analógica de las líneas de video de la norma RS-170A que representa la información de las imágenes.

Circuitería analógica. Una parte de la circuitería del digitalizador de video es analógica[1], ésta corresponde en parte al tratamiento de la señal de video y a los dispositivos empleados para tal propósito. En términos generales esta parte del sistema está compuesta por todos aquellos elementos que interactúan con señales analógicas:

- Acoplamiento de video con terminaciones de 75Ω.
- Corrección en CD de la señal de video.
- Rechazo a la subportadora de color.
- Almacenamiento parcial de la señal
- Generación de las tensiones de referencia para el ADC.
- Conversión analógica a digital.
- Generación de señales de sincronía.

[1] *Esta circuiteria es una versión modificada de la que originalmente fue presentada por Steve Ciarcia en la edición de Junio de 1986 de la revista BYTE. Para la gente interesada en investigar el diseño original se puede localizar en Ciarcia's Circuit Cellar, Vol. VII McGraw Hill 1990.* [5]

Circuitería digital. Está formada por aquellos elementos del sistema que trabajan con señales digitales.

- Almacenamiento de imágenes en memoria SRAM.
- Conexión a la PC mediante el puerto serie.
- Control del ADS802.
- Coordinación de las señales de sincronía provenientes del LM1881N.
- Implementación del protocolo RS-232 con el CI SIP233ACP para envió de imágenes completas y recepción de comandos.
- Programación del FPGA mediante una memoria EEPROM DE 512K.
- Generación de la señal de reloj para el ADC.

CAPÍTULO 1

Marco teórico, fundamentos para la digitalización de video

Considerando que el video es una secuencia de marcos, una imagen es la base para el funcionamiento de un digitalizador de video. El presente capítulo contiene las bases teóricas necesarias para el desarrollo del presente trabajo de tesis; describe como está integrada una señal de video compuesta y los parámetros más sobresalientes de la norma RS-170A para televisión a color utilizados en la digitalización de imágenes. Además se describen términos importantes como el teorema de muestreo y el proceso de cuantificación y codificación, necesarios para implementar la digitalización.

1.1 Imágenes en movimiento

Una sucesión de imágenes estáticas en ráfaga, capturadas y desplegadas en un rango de velocidad suficientemente alto o en tiempos suficientemente pequeños, pueden crear la ilusión de movimiento, generando así el sistema comúnmente conocido como video. La calidad del movimiento de las imágenes depende de muchos factores inherentemente vinculados a la fuente productora [1].

1.1.1 Exploración de imágenes

La subdivisión de una imagen en una secuencia de elementos individuales que pueden volver a combinarse con el fin de reproducir dicha imagen, se efectúa mediante una técnica denominada captación de imágenes[2]. Existen diferentes medios de exploración, tanto mecánicos como eléctricos, sin embargo, la mayoría de los sistemas modernos de televisión utilizan el movimiento de un haz de electrones que recorre la pantalla de los tubos de rayos catódicos o de los tubos receptores. La ventaja de la exploración mediante haz de electrones radica en que puede desplazarse con mayor rapidez y puede explorar una imagen completa en una fracción de segundo. La mayoría de las pantallas empleadas para reproducir el movimiento de las imágenes incluyen un periodo de tiempo durante el cual la imagen reproducida está ausente de la pantalla, esto es, una fracción de tiempo de marco[3], durante el cual la pantalla está en negro. Para evitar destellos desagradables, es necesario reproducir la imagen en un rango más alto que la velocidad de diferenciación del ojo entre una y otra imagen, simulando así el movimiento. El rango de refresco es proporcionalmente dependiente de la iluminación en el ambiente del espectador, ya que hasta cierto punto el umbral de destello es afectado por el brillo de la imagen, además de que es una función del ángulo del espectador con respecto de la imagen [1].

La frecuencia de refresco generalmente está generada dentro del sistema, una vez elegida no se puede modificar fácilmente. Algunas aplicaciones son diseñadas con frecuencias de refresco diferentes, dependiendo principalmente de la calidad de imagen necesitada y de las condiciones del espectador [1].

1.1.2 Adversidades en la captura de marcos de video

Es necesario incluir en el escenario la cantidad necesaria de luz para obtener la mejor imagen posible y así maximizar la sensibilidad y/o la relación señal a ruido. El tiempo de exposición es usualmente el tiempo efectivo de duración del campo, no de la duración del marco (dos campos comprenden un marco) y es necesario para conseguir buen movimiento de las imágenes o calidad de video, si la imagen tiene elementos que se movieron una distancia apreciable durante el tiempo de exposición entonces exhibirá manchas notables. Las manchas se pueden minimizar usando un tiempo de exposición adecuado además de la luz necesaria. Cuando el efecto de las manchas en la información de un marco persiste en marcos sucesivos, éste se exhibirá como retrasos en la reproducción del video, sin embargo estos retrasos no son problema para cámaras del tipo de dispositivo de carga acoplada [4] (CCD, *Charge Coupled Device*).

[2] *El objetivo va pasando por toda la imagen de forma análoga a como el ojo del lector recorre una página escrita, palabra a palabra y línea a línea, esta exploración genera una señal eléctrica proporcional a la luminosidad del punto explorado.*

[3] *Para el estándar RS-170A, el tiempo de marco ó imagen es típicamente de 33.3ms, la fracción se refiere a 1.27ms correspondientes a la sincronía vertical.*

[4] *Es un tipo de cámara muy común en aplicaciones informáticas, está formada por una matriz de semiconductores fotosensibles y tienen una resolución muy buena, además de que opera en condiciones de poca luz.*

Figura 1.1. Cámara CCD de estado sólido.

Otro factor que puede afectar considerablemente en la reproducción de las imágenes es el destello, pero éste se encuentra ausente en cualquier pantalla que presente estabilidad y las pantallas contemporáneas en su mayoría cumplen con ese requisito [2].

1.1.3 Definición en un marco de video

Cuanto mayor sea el número de líneas de barrido vertical en una imagen y cuanto mayor sea el número de elementos registrados en cada línea según se explora de izquierda a derecha, mayor es la definición o capacidad de la imagen para mostrar detalles minúsculos u objetos pequeños. En televisión, la frecuencia de repetición de marco y el número utilizado de líneas de barrido vertical tiene que ser estándar para un determinado sistema. Para mayor comodidad, estas normas de televisión se fijan para todas las emisoras y receptores de cada país.

En Europa y algunas otras partes del mundo se utiliza el sistema línea alternada en fase (PAL, *Phase Alternation by Line*), compuesto por 625 líneas horizontales y 25 marcos por segundo que proporcionan una alta definición. En Estados Unidos, sin embargo, las emisoras y los fabricantes de receptores adoptaron el sistema de televisión NTSC de 525 líneas horizontales por marco y una frecuencia de 30 marcos por segundo. El sistema electrónico para color con memoria francés (SECAM, *Systeme Electronique Couleur Avec Memoire*) al igual que PAL (Desarrollado por Telefunken) se compone de 625 líneas con 25 marcos por segundo [4].

En sistemas de video comerciales y de aplicación militar existen variaciones de los sistemas NTSC, PAL y SECAM, principalmente en las cámaras de estado sólido CCD y cámara infrarroja doble (CID, *Camera Infrared double*), entre otras, un ejemplo de cámara CCD se ilustra en la figura 1.1. Estas variaciones repercuten principalmente en la cantidad de líneas horizontales y marcos por segundo, además en algunos casos existen cambios en la amplitud de la señal [4].

1.2 Tubo de Rayos Catódicos

Para entender la operación de una plataforma de manipulación de video, es necesario saber que son las señales de video, inherentemente vinculadas a los dispositivos de video que operan con un tubo de rayos catódicos (TRC).

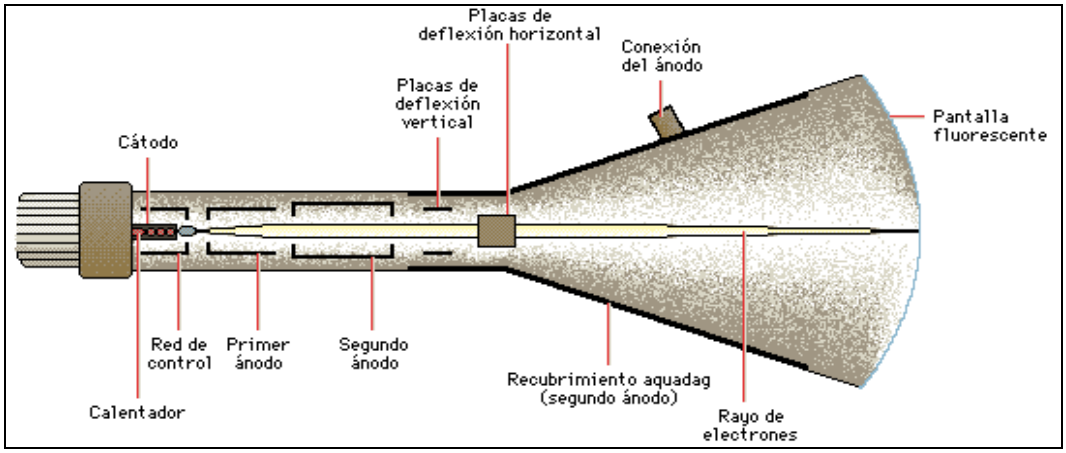

Figura 1.2. El tubo de imágenes ó cinescopio.

La figura 1.2 muestra el esquema general de un TRC y sus componentes más importantes, un TRC es el elemento del receptor de televisión que transforma la señal de video en la imagen que se muestra en pantalla. Si se desea que una imagen se despliegue sobre un TRC es necesario que un haz de electrones realice un barrido de izquierda a derecha y de arriba hacia a abajo a través de la pantalla. Los electrones impactados excitan puntos de fósforo depositados en el reverso de la pantalla y provocan que éstos brillen. La intensidad del brillo en el fósforo está determinada por la velocidad de impacto de los electrones y ésta es a su vez controlada por la tensión aplicada, como debe esperarse la tensión más alto produce el mayor resplandor del fósforo [5].

Los TRC's monocromáticos tienen un solo color de fósforo depositado sobre la pantalla, cuando el haz de electrones golpea el fósforo, éste resplandece en su color característico blanco, ámbar o verde, si un punto de fósforo no es excitado por el haz de electrones, éste se refleja como un punto oscuro o negro. Mediante una correcta modulación del haz de electrones se puede desplegar una imagen en un rango completo de tonos desde negro hasta blanco. Los TRC's de color tienen tres variedades de fósforo, cuando estos se excitan resplandecen como rojo, verde y azul, la intensidad de los colores primarios determina lo que el ojo humano percibe como el color de un píxel[5] o elemento de imagen sobre la pantalla, al impactarse los electrones sobre una matriz de puntos de fósforo depositados estrechamente unidos en el reverso de la pantalla. Los campos magnéticos aplicados al TRC controlan el haz producido por la pistola de electrones (con deflexión magnética) para que éstos se impacten en los puntos de fósforo, al mismo tiempo la intensidad del haz está siendo modulada para producir los diferentes niveles de intensidad. Los campos magnéticos son usados para llevar el haz de electrones a través del área de despliegue de la pantalla del TRC en ambas direcciones horizontal y vertical [5, 6].

1.3 Sistema NTSC

NTSC es la organización que se ha dedicado desde su fundación a la mejora de la calidad del video y establece todos los parámetros para su transmisión y recepción. Lo integran trabajadores de compañías líderes en el mercado de la electrónica, en sus inicios principalmente por RCA y *Columbia Broadcasting System, Inc* (CBS). NTSC es el nombre que recibió el primer sistema de televisión debido al nombre de la organización [7].

[5] *En informática, abreviatura fonética del concepto inglés elemento de imagen (picture element). Un píxel es el elemento más pequeño que el hardware y el software de pantalla pueden manipular.*

1.3.1 Historia del sistema NTSC

El primer sistema NTSC estableció los fundamentos que hicieron a la televisión monocromática, una realidad en los Estados Unidos, este sistema fue aprobado en 1941 y está todavía en uso. En los albores de la televisión a color parecía que ésta no sería compatible con la televisión monocromática y empezó una guerra por tratar de hacer compatibles los sistemas de televisión a color y monocromáticos hasta adoptar el segundo sistema NTSC que estableció las normas compatibles adoptadas por Estados Unidos y el resto del mundo [8].

En 1935 RCA desarrolló un sistema de televisión completamente electrónico de 343 líneas, el cual despertó las ambiciones de muchos en la industria del radio y derivó en la creación de asociaciones de fabricantes. Para la creación del primer sistema NTSC se revisaron en 1940 y 1941 los artículos existentes de televisión para obtenerlo, y fue adoptado por la comisión federal de comunicaciones (FCC, *Federal Communications Comisión*) como la base del servicio en blanco y negro. La mayoría tecnología involucrada en la televisión ha sido desarrollada por dos comités [6] de la asociación de fabricantes de radio (RMA, *Radio Manufacturers Association*) ahora Asociación de industrias electrónicas (EIA, *Electronics Industries Association*) [7].

En 1936 con insistencia de la FCC al tratar de establecer un estándar, el comité de asignaciones de televisión de la RMA realiza la propuesta más básica en la que el ancho de canal debería ser de 6MHz, debido a que se usaría modulación en amplitud de doble banda lateral, permitiendo un ancho de banda de video de no mas de 2.5MHz. El otro comité recomendó que fueran 441 líneas capturadas, 30 marcos por segundo entrelazados 2 a 1, modulación negativa de doble banda lateral para la señal de video, una razón de aspecto de 4 a 3, y modulación en frecuencia para la señal de audio, todo esto ocurría tan solo 11 años después de los primeros experimentos de Bair[7] y Jenkins que utilizaban adquisición mecánica de imágenes [4], [7].

En 1938 al agregar propuestas para la transmisión de brillo y polarización horizontal que son detalles de la sincronización, se incluyeron los pulsos de sincronización o ecualización, además de ampliar el ancho de banda de video de 2.5MHz a 4.2MHz y fue hasta la reunión del 8 de marzo de 1941 cuando el NTSC adoptó video de 525 líneas después de largos argumentos presentados por RCA, Philco y DuMont.

Después de muchos acuerdos y desacuerdos, formación de múltiples comités y cambios en asociaciones involucradas con la estandarización del video, encabezadas principalmente por los desacuerdos entre David Sarnoff, presidente de RCA-NBC, y William Paley, presidente de CBS, se estableció el segundo sistema NTSC sobre la norma monocromática existente. CBS y muchos otros fallaron en su intento por desarrollar un sistema para color, independiente del monocromático, la razón fue la gran cantidad de intereses técnicos, políticos y principalmente de tipo comercial por parte de las grandes empresas como RCA. Y fue hasta el 23 de diciembre de 1953 cuando la FCC autorizó el uso extensivo del segundo NTSC. Se puede decir pues que el segundo NTSC fue la operación más efectiva en la historia de la estandarización técnica, además de que el sistema ha sido adoptado en casi todos sus aspectos por el resto del mundo [7].

---

[6] *El comité de asignaciones de televisión y el comité de estándares de televisión miembros de RMA, establecieron parámetros para que cualquier televisor captara video transmitido bajo licencia de la FCC comandada por T.A.M Craven.*

[7] *Baird fue el primero en hacer una demostración pública de su primitivo sistema de televisión, el 26 de enero de 1926, en el Soho, Londres, este sistema transmitía imágenes de 30 líneas horizontales. Posteriormente participó en el desarrollo de un sistema mecánico de 240 líneas que fue sustituido por el electrónico de 405 inventado por EMI y Marconi.*

Tabla 1.1. Formatos de video.

Formato de video	*NTSC*	*PAL*	*SECAM*
VHS	√	√	√
Súper VHS[8]	√	√	√
Beta	√	√	√
Súper Beta	√	√	
ED-Beta	√		
D-VHS	√		
Video 8	√	√	√
Hi-8[7]	√	√	√
Video 2000[9]		√	
Visión Láser[10]		√	
Disco Láser[7]	√	√	√
U-Matic	√	√	
U-Matic SP	√	√	
Betacam SP	√	√	
1” C- Formato	√	√	
MII	√	√	
DVD	√	√	
DVCAM	√	√	
DVCPRO	√	√	
DVCPRO 50	√	√	
Digital S	√	√	
Digital Betacam	√	√	

1.3.2 Formatos de video

Existen muchos formatos de video para los sistemas de televisión, como se puede apreciar en la tabla 1.1, sin embargo todos están sustentados sobre de tres sistemas que son el NTSC, PAL y SECAM. Los formatos desde el VHS hasta el Disco Láser son analógicos domésticos, desde U-Matic hasta MII son analógicos profesionales y el resto son digitales profesionales.

1.3.3 Diferencias de NTSC con PAL y SECAM

Debido al éxito del segundo NTSC los demás sistemas cimbraron sus bases sobre éste. La mayor diferencia es que en PAL, la fase es invertida de línea a línea, con su correspondiente inversión en el receptor. El sistema SECAM empleado en Francia, Rusia y España, requiere que el receptor memorice el contenido de cada línea, además las señales de color son enviadas en una subportadora de crominancia a través de modulación en frecuencia. Ambos sistemas, a diferencia de NTSC, requieren receptores más complejos y tienen menos repeticiones de imagen por segundo, por lo cual poseen mayor definición.

[8] *Súper VHS, Hi-8 y Disco Láser son grabados en discos magnéticos como señales del estándar PAL y posteriormente son convertidos al estándar SECAM dentro de los reproductores vendidos en los mercados de productos SECAM.*

[9] *Video 2000 tiene un gran tiempo de reproducción que permite grabar 8 horas por lado de un VCC-480.*

[10] *Visión Láser (Sonido analógico) y Disco Láser (Sonido digital) son formatos diferentes dentro de PAL ya que no es posible retener la pista de sonido analógico cuando la señal de audio digital se agrega.*

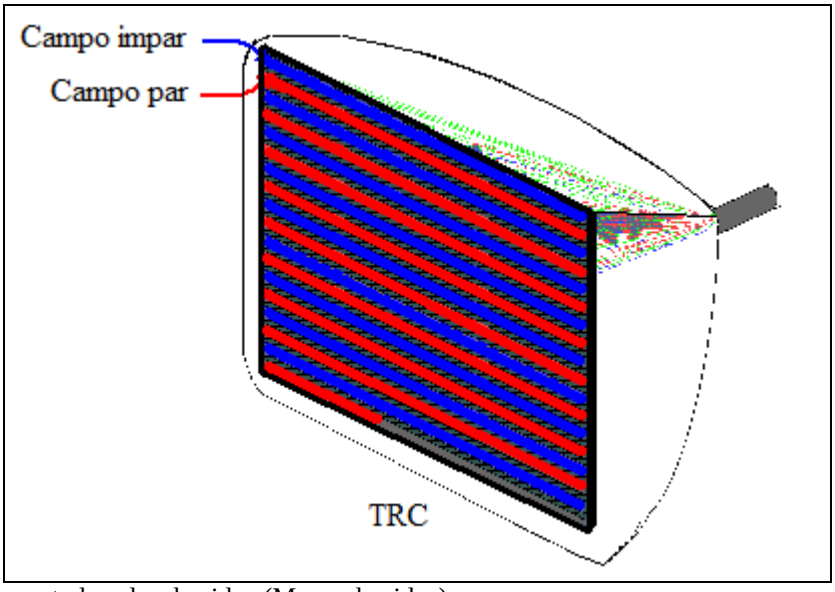

Figura 1.3. Campos entrelazados de video (Marco de video).

La mayor diferencia entre NTSC, PAL y SECAM es la resolución horizontal, con su correspondiente ancho de banda que varía desde 5.5 hasta 6.5 MHz, aunque también existe una diferencia en cuanto a los fotogramas o marcos utilizados por cada uno de los sistemas y en el cual NTSC toma ventaja con 30 marcos por segundo, sin embargo existen sistemas de video profesional que alcanzan 60 o 72 marcos por segundo, pero con un costo demasiado elevado [5].

1.4 Sistema NTSC de 525 líneas

La figura 1.3 ilustra el sistema NTSC de video entrelazado 2:1, como se muestra, una imagen de video es generada por dos barridos de un haz de electrones, a través de la superficie de un dispositivo TRC. Cada barrido es considerado como un campo y dos campos comprenden un marco de video. Un campo completo de video es desplegado antes de que el otro campo de video esté iniciando.

Cada campo contiene la mitad de la última línea de la imagen, el campo uno termina con la mitad de ésta, mientras que el campo dos inicia con la mitad de la misma. Esencialmente, el campo uno pone las líneas horizontales de video numeradas como impar y va dejando un espacio para situar las líneas del campo dos como se ve en la figura 1.3 [5], [8], [9].

El tiempo transcurrido al reposicionar el haz de electrones del final de una línea de video al inicio de otra se utiliza como referencia para digitalizar una línea de video o sincronizar el despliegue en el TRC. Éste es necesario durante ambos intervalos de retrazo vertical y horizontal para el correcto despliegue de la imagen.

Los intervalos transcurridos cuando el haz de electrones está apagado son llamados intervalos de blanqueado o espaciado.

El blanqueado vertical ocurre arriba y abajo del despliegue, mientras que el blanqueado horizontal ocurre en ambos lados, el izquierdo y el derecho. Para poder extraer la información de la imagen de video y realizar el proceso de digitalización es necesario entender donde ocurren los intervalos de blanqueado dentro de una señal de video.

1.4.1 Sincronía

La coordinación del haz de electrones es muy importante, ya que esta información debe estar presente en la señal de video para:

- Coordinación de retrazo horizontal.
- Coordinación de retrazo vertical.
- Blanqueado vertical.
- Blanqueado horizontal.

Adicionalmente, la amplitud de video es necesaria para definir una imagen desplegable en el caso de la norma RS-170 (monocromático) y para el estándar RS-170A (color) se requiere también la información del color. Las señales de video que contienen ambas informaciones de coordinación e imagen en una señal, se llaman señales de video compuestas.

La norma EIA para señal de video compuesta monocromático está definido por la especificación RS-170[11], el cual establece los límites sobre las características eléctricas de una señal de video tal que ésta pueda ser transferida entre equipos de video sin conversiones de señal. Las señales de video compuestas están disponibles en la mayoría de cámaras de televisión, algunos monitores de computadoras de color, grabadoras de video casete (VCR, *Video Cassette Recorder*) y algunas computadoras personales [5], [9].

La figura 1.4 muestra el diagrama de coordinación para la norma RS-170A de señales de video a color. Esta figura muestra la relación entre la información de control e imagen en una señal de video, principalmente ilustra las señales que contienen la sincronización vertical.

Todas las coordinaciones, también conocidas como sincronización, se representan a través de pulsos síncronos cortos, los cuales se limitan desde el nivel de blancura hasta el nivel síncrono. Las señales de video compuestas contienen componentes analógico y digital, el componente digital está hecho de varios pulsos síncronos, en tanto que el componente analógico contiene la información de la imagen.

El tipo, ancho y cantidad de pulsos síncronos son muy importantes cuando se extrae información de la imagen de una señal de video. En la figura 1.4 se identifican tres clases de pulsos [5], [9]:

- El *pulso síncrono*, tiene un ancho de aproximadamente 4.7us. Este pulso causa que el haz de electrones sea reposicionado desde el lado derecho del área de despliegue anterior al lado izquierdo preparándose para el despliegue de una nueva línea de video. La señal de video es blanqueada durante ese tiempo de retrazo horizontal, así que la ruta del retrazo no se muestra sobre la pantalla.
- El *pulso vertical*, tiene un ancho de aproximadamente 27.1 microsegundos. Este pulso provoca que el haz de electrones sea reposicionado desde el fondo derecho de la pantalla anterior al tope izquierdo, preparándose para el despliegue del siguiente campo de video.
- El *pulso de ecualización*, tiene un ancho de 2.3 microsegundos. Estos pulsos son usados para mantener el nivel de CD de la señal de video mientras se suministra información de sincronización al dispositivo de despliegue.

[11] *El estándar NTSC de color es un súper conjunto del estándar RS-170 y está regido por la especificación RS-170A.*

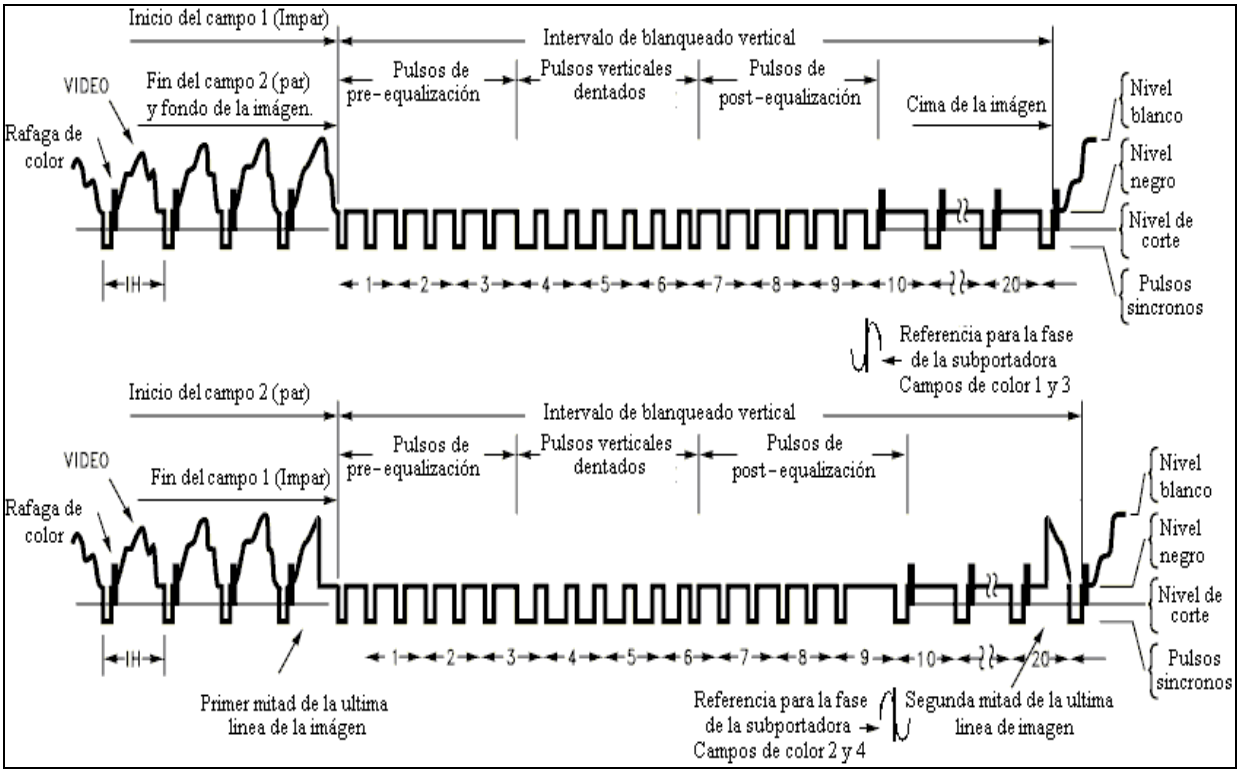

Figura 1.4. Pulsos de sincronía en la norma RS-170A.

Si el número total de pulsos con aristas negativas (síncronos horizontales, síncronos verticales y pulsos ecualizados) son contabilizados en ambos campos de un marco de video NTSC, resulta que el campo uno contiene 272 pulsos mientras que el campo dos está integrado de 271 pulsos. Estos datos son importantes para analizar una señal de video en tiempo real, ya que a través del conteo de pulsos en un campo, el software puede determinar cuál campo es el número uno y cuál es el número dos. Posteriormente se desarrollará esta técnica de muestreo[12] [5].

1.4.2 Líneas de video

El estándar NTSC norma RS-170A de video consiste de 525 líneas horizontales de información de video, actualizadas 30 veces por segundo. Para que se realice este despliegue, cada campo requiere la mitad de ese tiempo, es decir, 1/60 de segundo. Debido a que las 525 líneas son divididas igualmente entre los dos campos, cada campo está comprendido de 262.5 líneas, la parte fraccional de este número es la razón de que ambos campos de video contengan la mitad de una línea de despliegue.

Desplegar 262.5 líneas de video en 1/60 de segundo significa que cada línea requiere aproximadamente 63.5us. Este número es un importante parámetro para el diseño de un digitalizador de video, debido a que es la longitud del tiempo entre los píxeles posicionados verticalmente en un despliegue NTSC [5], [9].

No todas las 262.5 líneas de información de video en un campo son usadas para desplegar imágenes, aproximadamente 18.5 líneas son utilizadas para el blanqueado vertical y la sincronización del despliegue como se muestra en la figura 1.4. Esto deja 244 líneas por campo para información de la imagen. Estas 244 líneas por campo o 488 líneas por marco, forman la imagen desplegada sobre monitores de computadoras o TV's. De las líneas utilizadas para el

[12] *También se puede muestrear una imagen mediante tiempos, a partir de algún pulso de sincronía, aunque es una técnica mas complicada.*

blanqueado vertical, 17 líneas pueden además ser usadas para otros propósitos. Las aplicaciones de estas líneas de video oculto incluyen evaluación, diagnóstico, indicación y transferencia de señales de control. La información útil que no es relacionada al video puede ser transferida en éstas, esencialmente líneas de video sin uso. Esta información puede ser decodificada con un receptor configurado apropiadamente sin afectar el despliegue del video [5], [9].

La figura 1.5 muestra una línea de video compuesta, ésta contiene ambas señales de sincronización e información de video. Los diversos componentes de la forma de onda de video compuesta están etiquetados y se establecen sus tiempos aproximados. De los 63.5us que dura una línea de video, aproximadamente 53µs son dedicados a la información de la imagen, el resto del tiempo es empleado en la sincronización de la señal. Las amplitudes de las señales RS-170 están especificadas en términos de unidades del instituto de radioingenieros (IRE[13], *Institute of Radio Engineers*). La señal de video estándar de 1 volt pico a pico, se rompe sobre los 140 IRES. Todos los niveles de tensión entre –40 IRE's (el nivel síncrono) y 0 IRE's (el nivel de blanqueado) son considerados componentes de control o sincronización de la señal de video.

Los niveles de tensión entre + 7.5 IRE y + 100 IRE contienen la información de la imagen de video actual. Una tensión de +100 IRE (referenciado desde el nivel síncrono) es un punto blanco (píxel) que está siendo desplegado. Una tensión de 0.34V o 7.5 unidades IRE es desplegado como negro. Las tensiones entre estos extremos proporcionan píxeles de intensidad intermedia [5].

La norma RS-170 especifica que la impedancia de un dispositivo de video sobre el rango de frecuencia de 0 a 4.5MHz deberá ser 75Ω. La impedancia de entrada es un parámetro importante de diseño para el digitalizador de video. Para cumplir con la norma RS-170, un sistema de video deberá tener una potencia de resolución de por lo menos 330 líneas en la dirección vertical y 488 líneas en la dirección horizontal. Lo antes mencionado permite delinear detalles en una imagen de video [5].

Estas especificaciones son fácilmente encontradas en la mayoría de cámaras de video modernas y dispositivos de despliegue TRC. La mayoría de grabadoras de videocasete, sin embargo, no tienen las líneas verticales requeridas de resolución. Súper VHS es la excepción, porque tiene más de 400 líneas verticales de resolución. Esto es porque la calidad fotográfica es mejorada. En general, ya que el número de resolución de líneas horizontales se establece a través del sistema de video, todas las comparaciones entre los dispositivos de video deberán de hacerse en términos de líneas verticales de resolución [5], [9].

[13] *Las unidades IRE son un término empleado por el Instituto de Radio Ingenieros, con un factor de conversión a volts de 7.1429 mV por cada unidad IRE.*

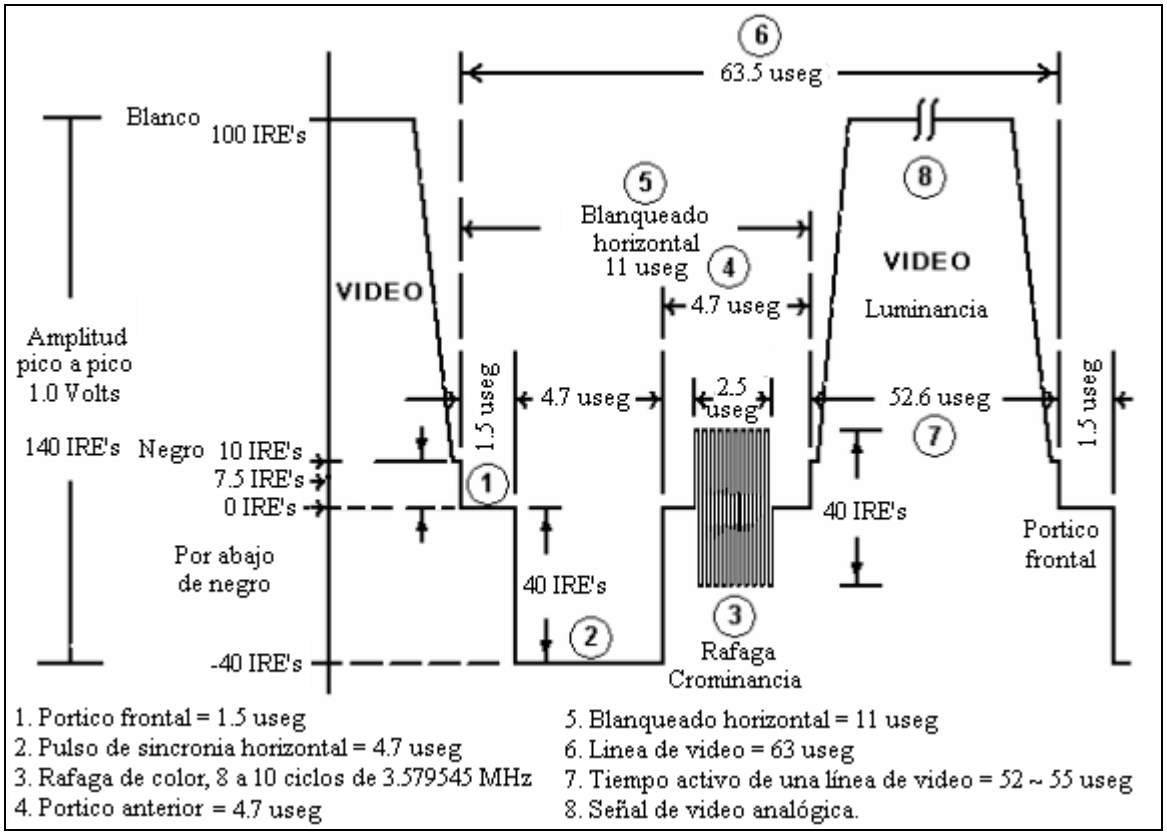

Figura 1.5. Línea de video compuesta.

Más líneas verticales de resolución producen una mejor imagen. Obviamente, los dispositivos de video con la más alta resolución tienen además los más altos precios.

De acuerdo con la figura 1.5, se consideran las siguientes definiciones [5], [9]:

1. ***Pórtico frontal***, es el tiempo de blanqueado con duración de 1.5us, se encuentra antes de la señal de sincronía horizontal.
2. ***Pulso de sincronía horizontal***, son los pulsos de sincronía de arista negativa al inicio de cada línea de la señal de video compuesta. Esta señal le indica al TRC que debe empezar a barrer los puntos a través de la pantalla.
3. ***Ráfaga***, es la frecuencia de la subportadora de color (3.579545 MHz para el estánsistedar NTSC), y es usada como referencia de fase para los procesadores de video.
4. ***Pórtico anterior***, es el tiempo de borrado después de la señal de sincronía, durante el cual se inserta la ráfaga de color.
5. ***Blanqueado horizontal***, intervalo de tiempo que se utiliza para reposicionar el TRC desde el final de una línea de video hasta el inicio de la siguiente.
6. ***Crominancia***, es la componente de color de la señal de video. El color es determinado por la fase de la componente de crominancia relativa a la señal de ráfaga.

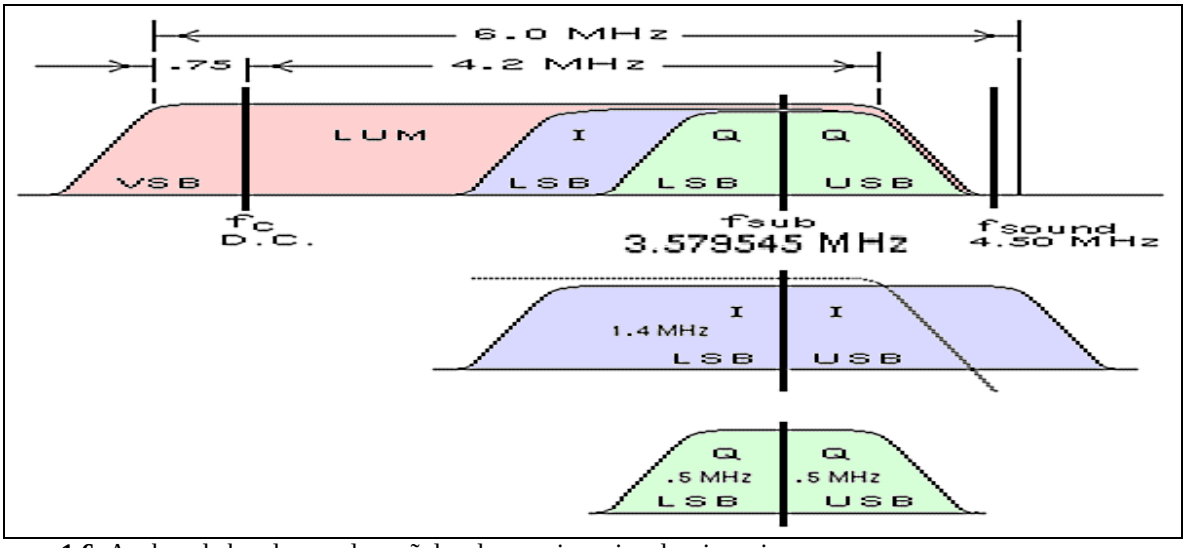

Figura 1.6. Anchos de banda para las señales de crominancia y luminancia.

1.4.3 RS-170A

El principio fundamental de los tres sistemas de codificación en color (NTSC, PAL y SECAM) es la fusión de la transmisión de dos imágenes separadas, una señal de banda ancha portando información de luminancia y una señal de crominancia con un ancho de banda más estrecho. La última es agregada a la señal de luminancia en la forma de una subportadora modulada. NTSC utiliza una frecuencia de subportadora crominante de 3.579545MHz mientras que los sistemas PAL y SECAM usan una frecuencia de 4.2MHz y 4.4 MHz respectivamente [10].

Las señales de los tres colores primarios son gamas exactas. La señal de luminancia es generada como la suma de tres fracciones de señales de color, estableciendo a través de sus combinaciones relativas a la luminancia de un estándar “blanco”. Los componentes de crominancia se obtienen restando primero la señal de luminancia de las señales de color, para luego obtener los diferentes valores del color: rojo, verde y azul sin luminancia. Debido a que la luminancia se transmite como una suma de fracciones de color, solamente dos de estas señales de diferentes colores deberán ser transmitidas para proporcionar una completa definición de los tres colores primarios.

En el sistema NTSC, las señales con diferencia de colores son sintonizadas y filtradas para producir una señal con ancho de banda naranja/cyan I (en fase) y una señal de banda estrecha magenta-verde Q (en cuadratura). Ambas son utilizadas posteriormente para modular dos señales subportadoras a 3.58 MHz las cuales están 90 grados fuera de fase una con respecto de la otra. Las subportadoras moduladas están además –57 grados y –147 grados alejadas de la fase de un estallido referente de subportadora, el cual acompaña cada pulso de sincronización horizontal como se puede ver en la figura 1.6 [10].

Los anchos de banda I y Q mostrados en la figura 1.6 y los colores representativos toman ventaja del hecho que el ojo humano puede percibir solamente una cantidad limitada de colores en pequeños detalles y es además relativamente limitado en su habilidad para resolver los detalles del magenta y el verde comparando a éstos con los colores naranja y cyan. El ancho de banda de la señal Q es llevado por esto a 0.5 MHz.

El ancho de banda de la señal I es aproximadamente tres veces más grande (1.3Mhz) pero solamente los componentes de bandas laterales mas bajos fuera de la doble banda lateral de la señal Q se conservan [10].

Cuando las dos señales subportadoras son combinadas, la fase resultante es una analogía directa del matiz, mientras que la amplitud de las señales combinadas es una medición indirecta de la saturación.

Además existen diferencias de coordinación entre el monocromo y las señales codificadas en color. Estas diferencias han hecho pequeños a los receptores monocromáticos y a los monitores que pueden mantener sincronización aunque una señal codificada en color esté siendo recibida.

1.5 Digitalización

En un instante dado existe sólo un punto activo sobre la pantalla de un TRC al realizar un despliegue de imagen, sin embargo, con una razón de refresco suficientemente rápida, la persistencia de visión del ojo hace posible ver la imagen completa.

Para digitalizar una imagen de video se debe considerar que ésta será almacenada como una matriz o arreglo de píxeles, dependiendo de la cantidad de píxeles o la definición de la imagen, el digitalizador de video requerirá más elementos e incrementará su robustez, pero principalmente necesitará de sistemas que digitalizan a frecuencias más altas. La señal de video del estándar RS-170A tiene un pulso de sincronía horizontal cada 63.5µs, éste marca el inicio de cada línea de video, además tiene una serie de pulsos verticales cada 16.7ms los cuales indican el inicio de cada campo de video.

La frecuencia de sincronía horizontal (f_H) es el número de veces por segundo que el haz de electrones barre una línea horizontal y está relacionado a través de [9]:

$$f_H = \frac{1}{t_H} = \frac{1}{63.5useg} = 15748Hz \qquad \textbf{Ecuación (1.1)}$$

El tiempo utilizado por la parte visible de una línea de video compuesta es de aproximadamente 53.5µs, los restantes 10µs son utilizados para la inserción de los pulsos de sincronía.

La frecuencia de sincronía vertical, se expresa comúnmente como el número de cuadros por segundo en una imagen de video y está expresada a través de [9]:

$$f_V = \frac{1}{t_V} = \frac{1}{16.7mseg} = 30 \text{ marcos/seg} \qquad \textbf{Ecuación (1.2)}$$

No sólo es necesario utilizar el teorema de muestreo para digitalizar una imagen, también se requiere de algún método[14] de cuantificación y de codificación.

[14] *De hecho el método para cuantificar una señal es el mismo, la variación se refiere al error de cuantificación ya que éste involucra utilizar algún método de redondeo o truncamiento, de acuerdo al dispositivo utilizado.*

1.5.1 Teorema de muestreo

Dada una señal analógica cualquiera, se puede elegir el periodo de muestreo o lo que es lo mismo, la velocidad de muestreo, si se sabe cierta información general sobre el contenido frecuencial de la señal a muestrear, luego entonces se puede utilizar el teorema del muestro, el cual establece que una señal de banda limitada a B Hz, puede reconstruirse a partir de sus muestras tomadas uniformemente a una razón no menor de 2B muestras por segundo.

El máximo intervalo de muestreo permisible, Ts = 1/2B, se conoce como intervalo de Nyquist[15] y el correspondiente índice o razón de muestreo (2B) muestras por segundo se conoce como el índice de muestreo de Nyquist [9].

Para digitalizar entonces el formato NTSC a una resolución de 488 píxeles desplegados horizontalmente se necesita muestrear a razón de 9.12 MHz, esto implica que para poder reconstruir una señal de video a partir de sus muestras al menos debemos tener una taza de muestreo de 18.24 MHz [11].

Con base en estos parámetros se pretende, construir un grabador de marco que digitalice imágenes con una resolución de 480 x 480 píxeles.

1.5.2 Cuantificación

Es el proceso de convertir una señal de amplitud continua en tiempo en una señal digital, expresando cada muestra por medio de un número finito de dígitos, en lugar de infinito (números reales). El error cometido al representar la señal de valor continuo por un conjunto finito de valores discretos se denomina error de cuantificación o ruido de cuantificación. Los valores permitidos en la señal digital se denominan niveles de cuantificación, mientras que la distancia entre dos niveles de cuantificación sucesivos se denomina escalón de cuantificación o resolución, en la tabla 1.2 se pueden observar valores para una señal de video cuantificada en 256 niveles [11].

En la práctica se puede reducir el error de cuantificación a niveles insignificantes, eligiendo un número suficiente de niveles de cuantificación. Teóricamente, la cuantificación de las señales analógicas resulta siempre en una perdida de información. Este es el resultado de la ambigüedad introducida por la cuantificación. La cuantificación es un proceso no invertible[16].

1.5.3 Codificación

El proceso de cuantificación consiste en asignar un número binario único a cada nivel de cuantificación diferente. Con una longitud de palabra de x bits se pueden crear 2^X números binarios diferentes [11].

Se puede observar en la tabla 1.2 una correspondencia entre datos cuantificados y datos codificados. En esta comparación cada 31 mV corresponden a un incremento binario. En la celda de cuantificación no se muestran más dígitos a la derecha debido al proceso de truncamiento efectuado.

[15] *Representa el máximo tiempo de muestreo sin perder información de la señal en proceso.*

[16] *Debido a que todas las muestras a una distancia inferior a la mitad de un determinado nivel se les asigna el mismo nivel.*

Tabla 1.2. Correspondencia entre valores cuantificados y codificados.

Nivel	*Señal analógica (Volts)*	*Cuantificación(Volts)*	*Codificación*
0	1.30000000	1.3000	00000000
1	1.303137254	1.3031	00000001
2	1.306274509	1.3062	00000010
3	1.309411764	1.3094	00000011
4	1.312549019	1.3125	00000100
5	1.315686274	1.3156	00000101
6	1.318823529	1.3188	00000110
…	…	…	…
…	…	…	…
251	2.087450980	2.0874	11111011
252	2.090588235	2.0905	11111100
253	2.093725490	2.0937	11111101
254	2.096862745	2.0968	11111110
255	2.100000000	2.1000	11111111

1.6 Características generales de un digitalizador de video comercial

En los últimos años, se han multiplicado las aplicaciones que emplean digitalizadores de video para desarrollar sistemas de visión artificial, sistemas de seguimiento, sistemas dedicados, seguimiento de satélites, etc.

A la par, el ámbito de laboratorio se está dejando para implementar sistemas comerciales que beneficiarán al público; algunos de los sistemas implementados para ejecución en tiempo real desde la aparición de los digitalizadores de video son los siguientes:

- **Sistema de seguimiento de misiles para pilotos**. Es un *display* integrado en el casco del piloto donde se muestra un punto de mira que es gobernado por los ojos, lo que le permite tener las manos libres y poder utilizarlas para otras tareas. El piloto utiliza sus ojos como un manipulador extra que activa cuando cree oportuno y el cálculo de la posición se realiza *on-line* con gran precisión.
- **Detección e identificación de caras en sistemas de monitorización y seguridad.** Se distinguen dos aplicaciones: las que tienen como objetivo la detección de caras humanas sin reconocimiento de las mismas. En este caso lo que se busca es detectar la presencia de una cara o bien contar el número de caras que pueden aparecer en una imagen compleja con multitud de usuarios. Por otro lado, están las aplicaciones destinadas a reconocer caras, lo que se busca es identificar si el rostro a analizar corresponde a alguno de los almacenados en una base de datos.
- **Desplazamiento del entorno en sistemas de realidad virtual**. En estos sistemas se presenta el control de un visualizador de imagen panorámico. El sistema permite hacer el control de "scroll" sobre una imagen panorámica en 360 º mediante un seguidor de mirada y permite actuar sobre el "zoom" mediante un reconocedor de voz. La empresa alemana SMI ("*Sensor Motoric Instruments*") comercializa, desde 1995, un sistema de seguimiento ocular sobre unas gafas de RV llamado H.M.D. ("*Head Mounted Display*"). El sistema permite moverse sobre un mundo virtual y realizar teleoperaciones.
- **Lectores de labios**. Cada vez son más las aplicaciones que, además de calcular la dirección de la mirada, realizan el seguimiento de la forma de características faciales (boca, cejas, ojos) con el fin de hacer reconocedores gesticulares destinados a desarrollar lectores de labios que puedan complementar a los reconocedores de voz.

- **Medicina**. Múltiples empresas disponen de digitalizadores de video de gama alta adecuados para la adquisición y tratamiento de imágenes médicas. Algunas aplicaciones como la tomografía emplean digitalizadores de video para capturar una secuencia de imágenes que permiten analizar ciertos órganos vitales.

En el mercado existen varios tipos de sistemas de procesamiento de imagen: *frame grabbers* cableados orientados a la aplicación, modulares con bus, con procesador programable y por último procesadores con bus rápido. Estos últimos delegan la mayoría de las tareas al procesador principal del computador que las alberga. En la tabla 1.3 se muestran los principales fabricantes de *frame grabbers*.

Tabla 1.3. Principales fabricantes de *frame grabbers.*

Fabricante	Modelo	Precio (dólares)	Comentario	Pagina Web	Salida	Entrada	Bus	Resolución
Píxel Smart	PS512-8	Mas de $199	El mejor precio, software y soporte técnico.	www.PixelSmart.com	RGB	NTSC PAL LVDS RS422	ISA PCI PC104	640x480 512x480 1Kx1K Cámara digital
Ellips	RIO PCI	$599	Selección pequeña y buena calidad de imagen.	www.ellips.nl	PAL	PAL	PCI	640x480
MeB Systems GmbH	Twist3	$999	Buen rendimiento	www.pci-tools.de	Si	RGB	PCI	746x556
Mikrotron	Inspecta	$1195	Caro.	www.mikrotron.de	Si	CCIR	PCI	640x480
Curtech	FF Frame Grabber	$850	Algunas opciones.	www.curtech.com	RS170	RS170	ISA	1024x1024 16 memorias
Optimum vision	MVision 100		Soporta cambios manuales	www.optimumvision.co.uk	No	RGB+ NTSC PAL		640x480
inX Systems	imPro	$1295	Multiples frame grabbers	www.inx.fi/index.html	No	PAL		640x480
Coreco	Bandit II RGB	$995		www.imaging.com	Si	RGB YUV	PCI	YUV 4:2:2
Matrox	Title motion	$595	Su negocio son las tarjetas VGA	www.Matrox.com	VGA	NTSC PAL	PCI	NTSC Marco completo
Epix	PIXCI D	$995	Soporta cámaras analógicas y digitales	www.Epix.com	No	LVDS RS422	PCI	1Kx1K
Scion	Cms800 NTSC	$4895	Rendimiento enorme.	www.scioncorp.com	Si	RGB, NTSC PAL	PCI	640x480

1.7 Estado del arte de los *frame grabbers* .

Los mecanismos de visión artificial tienen grandes ventajas sobre otros sistemas sensoriales ya que, bajo procedimientos adecuados, la cantidad de información que brinda una imagen es ampliamente superior a la de cualquier otro tipo de sensor. Esta característica incrementa enormemente el campo de aplicación de estos sistemas, más aún cuando el rápido avance tecnológico permite que el procesamiento de la imagen se realice en tiempo real.

La captura de imágenes en un sistema de visión artificial es un proceso que emplea *frame grabbers* ya sea de forma autónoma o supervisada, interfazados principalmente mediante las siguientes herramientas:

- FPGA's
- Procesador de señales digital (DSP's, *Digital Signal Processor*)
- Microcontroladores
- PC

Los *frame grabbers* que se pueden implementar con estas herramientas van desde los sistemas sencillos que emplean los requisitos mínimos hasta sistemas bastante complejos que emplean tecnología de punta de alto costo.

1.7.1 *Frame grabbers* con FPGA's (FgFPGA).

Este tipo de *frame grabbers* proporciona la posibilidad de diseñar sistemas autónomos que permitan realizar todos los procesos de un sistema de visión artificial, desde la captura pasando por los algoritmos de procesamiento hasta la interpretación de la información obtenida de la imagen. Aunque ofrecen la desventaja de poseer recursos limitados en cuanto a memoria, pueden ejecutar decisiones en tiempo real y son sistemas poco robustos en comparación con sistemas controlados.

1.7.1.1 Trabajos destacados con FgFPGA.

En [52], se desarrolla el sistema MIRIADS que proporciona despliegue en tiempo real y funciones de almacenamiento de datos digitales, además de realizar una amplia variedad de algoritmos de procesamiento basados en la imagen. Consta de dos DSP's que implementan funciones de coprocesadores matemáticos de la familia Quick DSP y cuatro CPLD's Lattice 1048EA y una arquitectura de procesamiento de imagen reconfigurable. Cuenta con 32000 PLD's y 40MB de SRAM. El uso de múltiples tarjetas en este sistema diseñado por Nova permite procesamiento serie o paralelo para operaciones de alta velocidad sofisticada.

En [59], el sistema de adquisición de imagen ha sido ha sido desarrollado con el FPGA EPF10K10TC144-3 de Altera, realiza las etapas de sincronía y almacenamiento temporal en memorias SRAM, éste se emplea para realizar procesamiento digital, transferencia y análisis de imágenes médicas a partir de equipamiento analógico, que necesita un sistema de alta resolución de 1024 líneas por 1024 columnas como máximo y 10 bits de resolución de conversión que permite su uso en aplicaciones portátiles. La transferencia de datos es basada en el estándar IEEE -1284 "Enhanced Parallel Port".

En [60], se implementa un sistema de tiempo real para la extracción y análisis de movimiento implementado en una tarjeta PCI, cuyo núcleo es un procesador FPGA de alta densidad APEX 20K400E de la marca Altera, las imágenes se almacenan temporalmente en 4 bancos de SRAM de 1Mx8 conectados al FPGA, el flujo de datos de 32 bits se envía al bus PCI. La velocidad de reloj del FPGA para coordinar los bloques de procesamiento y de almacenamiento es de 80MHz. La finalidad es que a partir de dos o más imágenes de una misma escena tomadas por cámaras ubicadas en posiciones distintas, es posible conocer la distancia que separa un objeto del sistema de cámaras.

1.7.2 *Frame grabbers* con DSP's (FgDSPs).

Los sistemas de adquisición de video con DSP's resultan en sistemas bastante completos debido a la capacidad de procesamiento y la alta velocidad de reloj que manejan, realizan la implementación de algoritmos de adquisición, mejora y procesamiento de imagen y al igual que los sistemas basados en FGPA's entregan resultados en tiempo real.

1.7.2.1 Trabajos destacados con FgDSP.

En [55] se implementa un *frame grabber* con DSP para realizar mediciones estáticas y dinámicas de superficies rugosas combinando el concepto de láser Dopler e interferometria de Speckle. El *frame grabber* es un Shark de Analog Devices, está basado en el concepto de funciones analíticas. Una función analítica se crea multiplicando la transformada rápida de fourier (FFT, *Fast Fourier Tansform*) por un filtro pasabanda centrado en la frecuencia de la portadora y con respuesta cero para frecuencias negativas.

1.7.3 *Frame grabbers* con Microcontroladores (FgM)

Debido a la muy limitada capacidad de los microcontroladores en cuanto a su velocidad de ejecución, a los pines entrada-salida, o a la memoria SRAM de la que disponen es difícil implementar por si solos un frame grabber. Por tanto necesitan hacer uso de compuertas que le reduzcan el número de tareas que debe realizar u otra forma más común es que el microcontrolador ejecute tareas e interfazado a la PC complementen las funciones necesarias para implementar un *frame grabber*.

1.7.3.1 Trabajos destacados con FgM.

En [53] Un microcontrolador de la familia HC11 DE Motorota busca la marca de sincronismo vertical que indica el comienzo de un cuadro de la imagen, comienza con un pulso de sincronismo estrecho y hay 312 líneas en cada cuadro que debe de informar a la PC. La PC mediante programación en C recibe estas señales de sincronía como las indicaciones de cuando leer cada byte o píxel de la imagen que está siendo digitalizada.

1.7.4 *Frame grabbers* con la PC (FgPC)

Este tipo de *frame grabbers* son interconectables con lo que se puede aprovechar la potencia de los procesadores de distintos *frame grabbers* para resolver una aplicación compleja.

Al poseer una gran cantidad de recursos en cuanto a memoria, frecuencia de operación, despliegue, el sistema se vuelve bastante completo, sin embargo ofrece un alto nivel de robustez, que implican no obtener resultados en tiempo real, no debido a la etapa de adquisición si no por las etapas de pre-procesamiento y de procesamiento de la imagen que muchas veces resultan ser bastante complejas.

1.6.4.1 Trabajos destacados con FgPC.

En [50], se utiliza esencialmente una cámara de video, un sistema electromecánico para movimientos en acimut y elevación de la cámara, una tarjeta de adquisición de video y una unidad para el control de motores a pasos y una computadora personal para la adquisición, almacenamiento y despliegue de imágenes.

El sistema utiliza una microcámara de alta resolución con función de auto iris que le permite adaptarse a diferentes niveles de iluminación. La computadora se comunica al controlador a través de su puerto serie RS-232 y un adaptador RS-232-485. La adquisición de imágenes se realiza a través de una tarjeta de adquisición de video PCI colocada en el interior de la computadora de control.

En [51], se utilizan los *frame grabbers* Matorx Metoer 2/MC2 y el Neotech IG24-PCI de Nacional Instruments con alta calidad y alta resolución en color y monocromático e interfaz PCI. Se emplea el sofware de Labview para la adquisición de datos, el acondicionamiento de señal y la adquisición de campos impar de imagen. También se utiliza el IMAQ PCI-1480 de alta exactitud que cumple con la norma RS-170 tiene un ADC de 8 bits y es capaz de transferir 132Mbytes/s y transfiere imágenes en tiempo real.

En [54], El sistema de adquisición está constituido por un computador Pentium 166 MHz, que dispone de un *frame grabber* imagenation PXC 200 conectada a una cámara CCD color JAI. La conexión al sistema de experimentación (microbot) se hace a través del puerto serie RS-232 (en aplicaciones típicas de control, la interacción con el sistema de experimentación se suele hacer a través de tarjetas entrada-salida).

En [56], se emplea un *frame grabber* construido en una tarjeta PCMCIA obteniendo imágenes a un rango de 25 marcos por segundo, con la finalidad de incrementar el nivel de autonomía de los robots móviles. Este sistema trabaja en tiempo real y está integrado en una laptop Pentium III colocada sobre el robot Nomad 200 que se comunica con la PC a través del protocolo Ethernet TCP-IP a 10 Mbps.

En [57], se emplea un *frame grabber* PX 500 que soporta ajustes de offset y ganancia, y además tiene incorporado un modulo de tablas de búsqueda (LUT, *Look Up Table*) para ajustes de contraste. La imagen digital con la que se trabaja es monocromática. Las dimensiones de la matriz son 640x512, y la magnitud de cada elemento o píxel varía entre 0 (negro) y 255 (blanco), resolución de 8 bits. Luego con la imagen digitalizada se aplica un operador de derivada para realizar la detección de bordes. Esta función permite resaltar la zona iluminada por la luz láser. A continuación se realiza una umbralización de la imagen resultante. Esta transformación permite extraer objetos de una imagen cuando las intensidades de los píxeles del objeto de interés y del entorno tienen niveles de gris agrupados en dos tonos dominantes.

En [58], se emplea un *frame grabber* modelo Meteor II de MATROX en bus PCI con entrada de video estándar y una velocidad de transferencia en tiempo real a la memoria a 130MB/s con una resolución de 1280x1024. El sistema propone una silla de ruedas motorizada guiada en entornos interiores, este sistema genera una serie de comandos mediante los movimientos de cabeza, ojos y boca.

CAPÍTULO 2

Métodos utilizados, dispositivos lógicos programables en campo y VHDL.

Los FPLD’s existen en una gran variedad y han revolucionado la implementación de lógica digital en diseños electrónicos; en este capítulo se desarrollara un panomara global de éstos.

Para configurar al dispositivo es necesario conocer la metodología de diseño empleada durante este proceso, por ello se analiza en forma general un método de diseño vinculado a la mayor parte de FPLD’s existentes en el mercado. Se analizan también las características más importantes de los lenguajes descriptores de hardware (HDL’s, *Hardware Description Languaje*) debido a que estos representan un método de descripción ampliamente utilizado, profundizando en VHDL al ser éste el lenguaje utilizado en el presente diseño.

En resumen, este capítulo presenta el marco teórico que envuelve el uso de estas herramientas para poder así aplicarlas correctamente en el diseño e implementación de este trabajo de tesis.

2.1 FPLD's

En términos generales, un FPLD se refiere a cualquier tipo de circuito integrado (CI), utilizado para implementar hardware digital, donde el chip puede ser configurado por el usuario final para realizar diferentes diseños. Es distribuido por el fabricante en un estado no programado a diferencia de los arreglos de compuerta programable mediante mascara (MPGAs, *Mask Programmable Gate Array)*). Los FPLD's pueden ser programados en el campo por el usuario, para ejecutar las funciones deseadas como parte de sistemas de alto rendimiento. Los beneficios que se obtienen de utilizar FPLD's, entre otros son: el rápido desarrollo de prototipos de CI's a un bajo costo y que pueden implementarse diferentes diseños, es decir, son programables múltiples veces [12], [13].

Paralelamente se puede definir a un dispositivo lógico programable (PLD, *Programmable Logic Device*) como un dispositivo con lógica configurable y flip-flop's unidos con interconexión programable. Las celdas de memoria controlan y definen las funciones que debe realizar la lógica configurable y como se deben de interconectar las funciones lógicas. Aunque la mayoría de dispositivos usan diferentes arquitecturas todas se basan en esta idea fundamental [15].

2.2 Surgimiento de los FPLD's

Desde la aparición del transistor a finales de la década de los 40's, el campo de la microelectrónica ha tenido una rápida evolución con diversos tipos de cambios dentro del nivel de desarrollo de sistemas electrónicos; en los albores de los 60's la posibilidad de hardware reconfigurable dinámicamente partió de la idea del arreglo de compuertas como una vía para realizar el diseño de CI's más fácilmente y más baratos, con esto solamente cambiaban el nivel de complejidad de los componentes del sistema, pero la llegada de los procesadores[17] de propósito general en los años 70's produjo importantes cambios, como la incorporación de sistemas que podían ser adaptados a través de software [14], [15], [16].

El primer tipo de chip programable por usuario que pudo implementar circuitos lógicos fue la memoria de solo lectura programable (PROM, *Programmable Read Only Memory*), en el cual las líneas de dirección pueden ser utilizadas como entradas de circuitos lógicos y las líneas de datos como salidas. Sin embargo las funciones lógicas, raramente requerían más de algunos términos producto, y la PROM[18] contiene un decodificador completo capaz de implementar funciones lógicas de gran complejidad.

El primer dispositivo desarrollado específicamente para implementar circuitos lógicos fue el arreglo lógico programable en el campo (FPLA, *Field Programmable Logic Array*), o simplemente arreglo lógico programable (PLA, *Programmable Logic Array*).

Cuando los PLA's fueron introducidos en los primeros años de los 70's, por Phillips, sus principales inconvenientes fueron que era muy caro fabricarlos y ofrecían una pobre velocidad de ejecución. Ambas desventajas fueron debido a los niveles de lógica configurable, ya que los planos lógicos programables son difíciles de fabricar e introducen retardos de propagación significativos. Para superar estos problemas, se desarrollaron los dispositivos tipo lógica de arreglos programables (PAL, *Programmable Array Logic*).

[17] *En 1971, Intel Corporation y el talento creativo de Marcian E. Hoff Lanzaron el primer procesador: el 4004 de 4 bits.*

[18] *Las PROM's son arquitecturas ineficientes para realizar circuitos lógicos y raramente se usan en la práctica para estos propósitos*

Un PAL característico tiene solamente un nivel de programabilidad, éste consiste de un plano AND programable que alimenta compuertas OR fijas. Para compensar la falta de generalidad debido a que el plano OR es fijo, se produjeron múltiples variantes de PAL's, con diferentes números de entradas y salidas y varios tamaños de compuertas OR. Posteriormente surgieron la lógica de arreglos genérica (GAL's, *Generic Array Logic*), las cuales usualmente contienen flip-flop's conectados a las salidas de las compuertas OR para que puedan realizarse circuitos secuénciales. Los dispositivos GAL son muy importantes porque tuvieron un profundo efecto en el diseño del hardware digital, además de que son las bases de las arquitecturas más sofisticadas mencionadas posteriormente [12].

Todos los dispositivos PLA's, PAL's GAL[19] se agrupan dentro de una categoría llamada dispositivos lógicos programables simples (SPLD's, *Simple Programmable Logic Devices*), cuyas características más importantes son el bajo costo y la alta velocidad de trabajo.

2.3 Tipos de FPLD's

En la actualidad existe un incremento en el número de fabricantes de FPLD's y una gran variedad de dispositivos y familias lo que hace difícil su clasificación debido a la ausencia de un cuerpo de doctrina que lo armonice y a la falta de métodos de clasificación; aunque desde finales de los años 80's se han llevado a cabo diversos intentos para metodizar el estudio y clasificarlos, en general son poco sistemáticos, se centran en la descripción de los circuitos de los fabricantes y no analizan ni comparan adecuadamente las diferentes arquitecturas [15], [19], [20], [21], [22].

Sin embargo teniendo como referencia diversos autores se utilizaran los métodos de clasificación más generales, como son:

- Tecnología de reconfiguración.
- Nivel de integración (Cuantas compuertas contiene el dispositivo).
- Arquitectura interna.

2.3.1 Tecnología de reconfiguración

La tecnología de reconfiguración se define en base a cuantas veces y que tan fácil es reconfigurar los FPLD's, los primeros FPLD's programados con tecnología antifusible, solamente podían ser programados una vez, sin embargo hoy en día los que se basan en tecnología de memoria estática de acceso aleatorio (SRAM, *Static Random Access Memory*) pueden ser programados ilimitadamente.

Por tanto de acuerdo a su esquema de configuración, los dispositivos FPLD's se pueden clasificarse dentro de cuatro clases, que son:

- Programables.
- Reconfigurables.
- Parcialmente reconfigurables.
- Dinámicamente reconfigurables.

Las dos ultimas clases de dispositivos están aun en desarrollo, todavía en sus fases iniciales en nuevos campos como lo son sistemas de computo configurable, ya que deberán resolver previamente algunos problemas principalmente debido a herramientas de diseño asistido por

[19] *Este tipo de dispositivos emulan una gran cantidad de PAL's mediante el uso de macroceldas.*

computadora (CAD, *Computer Design Assisted*) sin embargo existen algunos de estos dispositivos disponibles hoy en día [15], [16], [32], [33], [36].

2.3.1.1 FPLD's programables

Los primeros dispositivos programables aparecieron una década antes de la evolución de los PLA's y los PAL's, los cuales permitieron al usuario configurar su funcionalidad. Eran muy simples, pero su bajo costo y alta velocidad, aunadas a su flexibilidad, permitieron el reemplazo de la mayoría de las interfaces lógicas entre componentes, integradas dentro de uno de estos dispositivos [15], [32].

Como la tecnología fue mejorando sus niveles de integración, también lo hicieron la velocidad y capacidad de los dispositivos programables[20]. Una vez que la funcionalidad del dispositivo está determinada, se lleva a cabo la configuración quemando ciertos fusibles seleccionados, lo cual controla físicamente el comportamiento del dispositivo. Ya que éste es un proceso destructivo solamente puede ser realizado una vez, debido a ello son llamados dispositivos programables como se bosqueja en la figura 2.1.

2.3.1.2 FPLD's reconfigurables

La mayoría de los FPLD's han superado la limitación anterior y pueden ser programados más de una vez, por lo cual son llamados FPLD's reconfigurables, sin embargo el número de reescrituras y que tan fácil pueden ser realizadas depende de la tecnología utilizada para la implementación de los switches reconfigurables. En la tabla 2.1 se ilustran tecnologías reprogramables y volátiles.

Como se muestra en la tabla 2.1 los dispositivos implementados con memoria de solo lectura programable y borrable (EPROM, *Erasable Programmable Read Only Memory*) y memoria de solo lectura programable y borrable eléctricamente (EEPROM, *Electrically Erasable Programmable Read Only Memory*) almacenan su configuración hasta la siguiente reprogramación, es decir, no son volátiles.

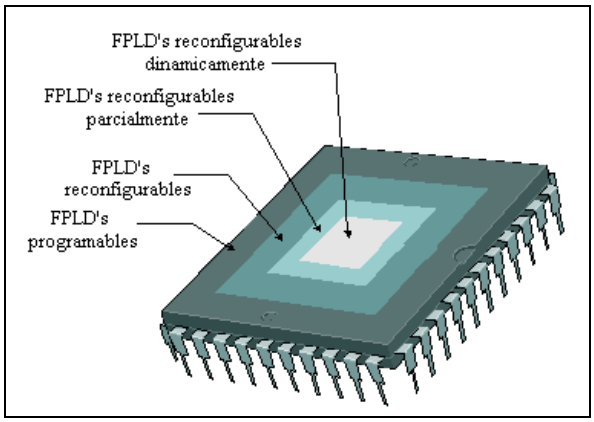

Figura 2.1. Tipos de FPLD's con base en su configuración.

[20] *Así nacieron los dispositivos que actualmente se conocen como FPGA's y CPLD's, estos dispositivos heredaron de sus predecesores el método de programación basado en fusibles y aifusibles.*

Tabla 2.1. Tecnología para programación de switches.

Tipo de Switch	¿Reprogramable?	¿Volátil?
Fusible	No	No
Anti-fusible	No	No
EPROM	Si (fuera del sistema)	No
EEPROM	Si (dentro del sistema)	No
SRAM	Si (dentro del sistema)	Si

En el caso de las EPROM la programación se lleva a cabo fuera del circuito. Los dispositivos basados en tecnología SRAM y EEPROM pueden ser reprogramados dentro del sistema, y aunque la SRAM pierde su configuración cuando el circuito ya no está siendo alimentado, su reconfiguración es más rápida y fácil, además de que puede realizarse un número ilimitado de veces. Esta es la razón por la cual son los dispositivos más utilizados para aplicaciones comerciales.

Otro parámetro que depende bastante del tipo de switches es el tiempo de reconfiguración, éste rara vez está por arriba de varios milisegundos, entre 5ms y 30ms. Usualmente el contenido de los arreglos de compuertas programables en campo (FPGA's, *Field Programmable Gate Arrays*) se carga durante el encendido del sistema y nunca se cambia durante el tiempo de su ejecución. Una vez que la aplicación ha terminado se puede cargar una nueva configuración y como esta tarea consume mucho tiempo raramente se realiza en tiempo de ejecución [15], [16], [32], [33], [36].

2.3.1.3 FPLD's parcialmente reconfigurables

Un subconjunto especial de los FPLD's reconfigurables son los FPLD's parcialmente reconfigurables que permiten modificar selectivamente los bits de programación de determinadas partes del dispositivo, sin afectar la lógica permanente, este tipo de FPLD's se ilustra en la figura 2.1. La ventaja obvia de este tipo de dispositivos sobre los FPLD's reconfigurables es que requieren menos tiempo de reconfiguración, ya que no es necesario configurar el FPLD completo. Este tiempo de reconfiguración es proporcional al tamaño del dispositivo a reconfigurar [15], [36].

Una característica común a todos estos dispositivos es que son finamente granulados[21] y su configuración está controlada a través de la lectura y escritura de ciertas direcciones de memoria estática (basadas en tecnología SRAM) de un dispositivo externo, generalmente la unidad central procesadora (CPU, *Central Processor Unit)* principal [36].

Otra característica es la latencia de reconfiguración que se define como el tiempo empleado desde que se da la orden de reconfiguración hasta que el circuito está listo para operar de nuevo. Esta latencia, en el peor de los casos, equivaldrá a reconfigurar totalmente el dispositivo, pero generalmente depende del número de celdas que se reconfiguren, por lo tanto el tiempo de reconfiguración se reduce considerablemente [15], [36].

[21] *Conjunto de CLBs simples con una función lógica de dos entradas o un multiplexor de 4 a 1 y flip-flops.*

2.3.1.4 FPLD's reconfigurables dinámicamente

Si es necesario detener la ejecución en el dispositivo para reprogramarlo se puede utilizar las anteriores clases de configuración de otra forma es necesario utilizar FPLD's reconfigurables dinámicamente, los que también son llamados FPLD's configurables en línea o en tiempo de ejecución. Sus tiempos de ejecución no son solamente bajos comparados con un FPLD reconfigurable o parcialmente reconfigurable, si no que además la reconfiguración de una parte del dispositivo no evita que la lógica restante ejecute simultáneamente su operación normal. Dentro de los FPLD's reconfigurables dinámicamente, encontraremos dos arquitecturas bastante distintas, las cuales tienen mecanismos y latencias de reconfiguración completamente diferentes.

La primer arquitectura surge de la evolución natural de los FPLD's reconfigurables (que podían ser programados globalmente y fuera de ejecución), donde todos sus recursos son direccionables independientemente (así entonces se permite la reconfiguración parcial), por lo cual se ha proporcionado un nuevo mecanismo de reconfiguración.

Para que la ejecución de las partes no reconfiguradas no afecte durante el reemplazo de las otras partes cada nueva configuración se almacena dentro de una memoria externa. La otra alternativa consiste en tener más de un plano de memoria reconfigurable (llamado contexto), pero solamente uno activo en un tiempo. La reconfiguración consiste entonces en programar el contenido de los planos inactivos, sin perturbar la ejecución del activo y cambiar al nuevo tan pronto como sea requerido [15], [36].

De esta manera toda la configuración del dispositivo puede ser reemplazada en un solo ciclo de reloj[22], así la ejecución de la aplicación no se detiene. Por otro lado, existe un precio en términos de área que debe ser pagado por contextos adicionales, sin embargo se ha demostrado que este gasto no es muy alto, ya que la parte funcional de los FPGA's no es grande comparada con el tamaño dedicado a las interconexiones, es mas, los planos de memoria adicional pueden ser utilizados como memoria de acceso aleatorio (RAM, *Random Access Memory*) [15], [36].

En la tabla 2.2 se muestran algunos ejemplos de este tipo de arquitecturas, los cuales se pueden diferenciar por el número de contextos de reconfiguración. Se puede apreciar que el tiempo de reconfiguración para los dispositivos de arreglo de compuertas programable dinámicamente (DPGA, *Dynamic Programmable Gate Array*) y Sistemas dentro de CI's programables en el campo (FIPSOC, *Field Programmable System-on-Chip*) es de 10ns, mientras que para los dispositivos de Xilinx y Atmel va desde 40ns hasta 200ns. En contraparte con su capacidad numerada en compuertas.

Tabla 2.2. Ejemplos de arquitecturas reconfigurables dinámicamente.

Fabricante	Familia	Tipo	Contextos	Tiempo de reconfiguración	Capacidad
Xilinx	XC6200	Comercial	1	40 ns / celda	100K compuertas
Atmel	AT6000	Comercial	1	0.2 μs /celda - 8ms / total	200K compuertas
MIT	DPGA	Académico	4	9.5 ns [23]	Desconocido
SIDSA	FIPSOC[24]	Comercial[25]	2	10 ns (1 ciclo)	10K compuertas

[22] *Por lo tanto la latencia de reconfiguración se reduce considerablemente, es decir, el tiempo empleado no es más grande que un periodo de reloj.*

[23] *No está en uso, es para futuras implementaciones*

[24] *FIPSOC no es solamente un dispositivo programable, además se compone de un microcontrolador 8081, un arreglo programable digital y algunas celdas análogas configurables.*

[25] *Todavía es un prototipo.*

2.3.2 Nivel de integración

Este tipo de clasificación se basa en el número de compuertas que contiene un CI. Aquí no se realiza este tipo de clasificación debido a la gran cantidad de dispositivos existentes. Para más información consultar [20], [21], [22]. Sin embargo la siguiente clasificación en base a la escala de integración es de gran ayuda:

- Pequeña escala de integración **(SSI,** *Small Scale Integration)*.
  Número de compuertas: Menos de 12.
  Dispositivos digitales típicos: Compuertas tipo NAND, OR, AND, NOT, flip-flop's.

- Mediana escala de integración **(MSI,** *Medium Scale Integration).*
  Número de compuertas: De 12 a 99.
  Dispositivos digitales típicos: Sumadores, contadores, decodificadores, codificadores, multiplexores, demultiplexores, registros.

- Grande escala de integración **(LSI,** *Long Scale Integration*).
  Número de compuertas: De 100 a 9,999.
  Dispositivos digitales típicos: Relojes digitales, chips grandes de memoria, calculadoras avanzadas, microprocesadores simples, dispositivos para interfase, PLA's, PAL's, GAL's.

- Muy grande escala de integración **(VLSI,** *Very Long Scale Integration*).
  Número de compuertas: De 10,000 a 99,999.
  Dispositivos digitales típicos: Microprocesadores como el 4004, Z80, 8008, 8080, 8085, 8086, 8088, chips grandes de memoria, calculadoras avanzadas.

- Ultra grande escala de integración **(ULSI,** *Ultra Long Scale Integration*).
  Número de compuertas: De 100,000 a 999,999.
  Dispositivos digitales típicos: Microprocesadores avanzados tales como el 286, 386.

- Giga grande escala de integración **(GLSI,** *Giga Long Scale Integration*).
  Número de compuertas: De 100,000 a 999,999.
  Dispositivos digitales típicos: Procesadores como el 486, Pentium, Pentium MMX, Pentium II, Pentium III, Pentium IV, ASIC's, Controladores dedicados como teléfonos celulares, reproductores de CD's, DSP's (Procesadores de señales digitales), FPGA's, CPLD's.

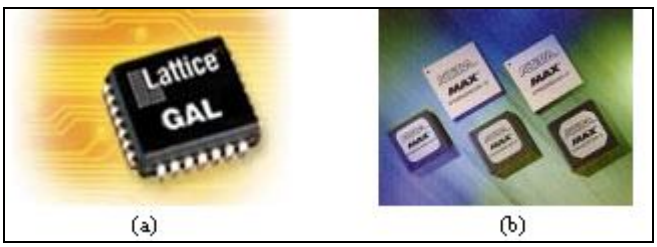

Figura 2.2. SPLD's. (a)GAL22V10 de Lattice, (b) SPLD's de Altera.

2.3.3 Arquitectura interna

Este tipo de clasificación surge debido a la gran cantidad de fabricantes que existe hoy en día y a las diferentes familias de dispositivos que cada uno de estos fabricantes desarrolla; esta clasificación brinda al investigador un panorama global de los dispositivos FPLD's. La mayoría de familias se incluyen en:

- Dispositivo lógico programable simple **(SPLD's,** *Simple Programmable Logic Device***)**
- Dispositivos lógicos programables complejos **(CPLD's,** *Complex Programmable Logic Devices*)
- **FPGA's**
- Interconexión programable en campo **(FPIC's-** *Field Programmable Interconnection*)

2.3.3.1 SPLD's

Los SPLD's[26] son los integrantes más pequeños de la lógica programable y consecuentemente los menos caros. Un SPLD está constituido de 4 a 22 macroceldas típicamente y puede con esto sustituir unos cuantos dispositivos de lógica transistor- transistor (TTL, *Transistor Transistor Logic*) de la serie 7400, como los que se muestran en la figura 2.2. La mayoría de SPLD's usan fusibles o celdas de memoria no volátiles tales como las EPROM, EEPROM ó FLASH para definir su funcionalidad. Los SPLD's también se conocen como:

- **PLA's**
- **PAL's**.
- **GAL's**.
- Secuenciador lógico programable en el campo **(FPLS's,** *Field Programmable Logic Sequencer*)

PLA's

Un PLA es un dispositivo AND-OR de dos niveles combinaciónales que pueden programarse para realizar cualquier expansión lógica de suma de productos como se ve en la figura 2.3, sujeta a las limitaciones[27] de tamaño del dispositivo. Los primeros PAL usaron tecnología bipolar y tenían fusibles semejantes a los de las memorias de solo lectura (ROM, Read Only Memory) bipolares. Las PAL metal óxido semiconductor complementario (CMOS, Complementary Metal Oxide Semiconductor) usan tecnología EPROM de compuertas flotantes y son reprogramables (cuando tienen una cubierta transparente para el borrado). Cabe mencionar también una ROM puede verse como un caso especial de un PLA [23], [25].

[26] *Aun y cuando parezca que un SPLD no cumple con las características de un dispositivo FPLD's, se incluirá en esta investigación porque se considera que son la base para el desarrollo de los dispositivos FPLD's.*

[27] *Las limitaciones son el número de entradas, salidas y el número de términos producto.*

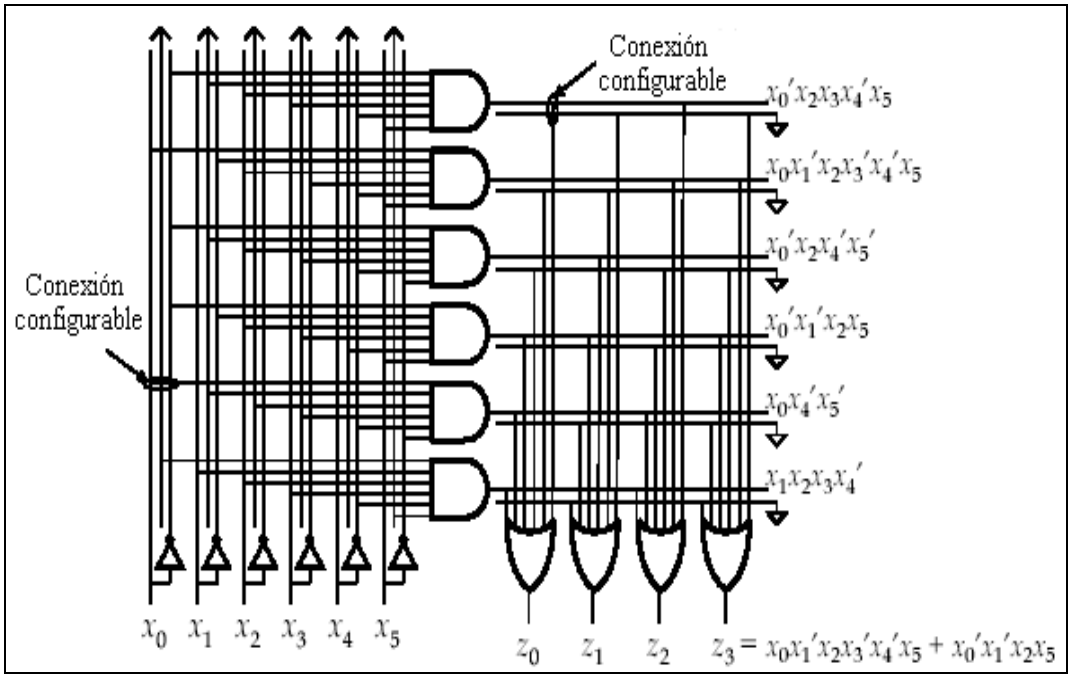

Figura 2.3. Esquema general de un PLA.

PAL's.

Este tipo de dispositivos fueron introducidos por la compañía *Monolithic Memories*, ahora parte de AMD (*Advanced Micro Devices*, Inc.) bajo el nombre de Vantis, en las postrimerías de la década de los 70's. Las PAL's[28] son pequeños FPLD's que tienen un arreglo AND programable y un arreglo OR fijo como se puede ver en la figura 2.4.

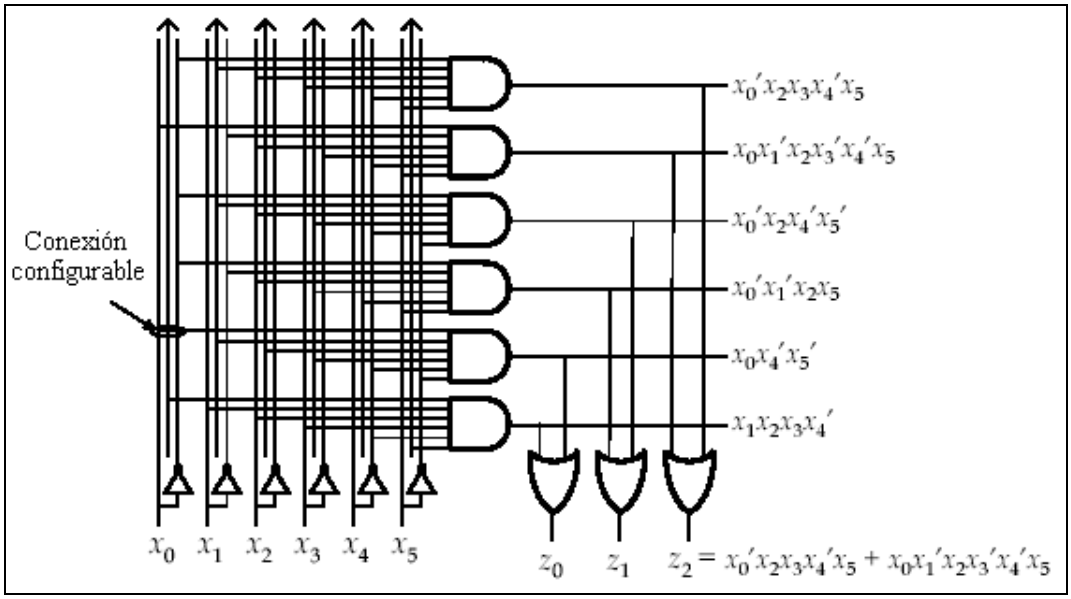

Figura 2.4. Esquema general de un PAL.

[28] *Una sola PAL puede reemplazar un circuito con un gran número ó quizás cientos de compuertas lógicas.*

Las innovaciones aportadas por este tipo de dispositivos a la electrónica digital, además de un acrónimo atrayente fueron el uso de entradas/salidas bidireccionales y retroalimentación mediante registros. El arreglo programable contiene compuertas lógicas, arregladas en funciones con interconexiones programables, puede crear cualquier función booleana desde una selección de las entradas en cualquiera de sus salidas. En una PAL las compuertas lógicas están colocadas como un arreglo de suma de productos, siendo que cualquier función booleana se puede reducir a esta forma [23].

Para configurar una PAL es necesario utilizar un programa que genere un formato estándar definido por el concilio ingenieríl de dispositivos electrónicos unidos (JEDEC, *Join Electron Device Engineering Council*), este tipo de archivos pueden ser editados manualmente por el diseñador, posteriormente es necesario utilizar un programador para cargarlos al dispositivo PAL, cabe mencionar que una desventaja en las PAL's es que solamente pueden ser programadas una vez.

GAL's

Las GAL's son dispositivos de matriz lógica genérica, como el que se muestra en la figura 2.2(a). Están diseñadas para emular muchos dispositivos tipo PAL que utilizan macrocélulas. Si un usuario tiene un diseño que se implementa usando varias PAL's, puede configurar un número menor de GAL's para emular los dispositivos anteriores, esto reducirá el número de dispositivos diferentes en existencia y la cantidad comprada. Comúnmente, una cantidad grande del mismo dispositivo debería rebajar el costo individual del dispositivo. Estos dispositivos también son eléctricamente borrables, lo que los hace muy útiles para los ingenieros de diseño.

FPLS's

Los FPLS's fueron introducidos al mercado por Signetics en 1979, son algunos de los elementos lógicos programables más antiguos, desarrollados para apoyar la implementación de circuitos lógicos. Su arquitectura es similar a un dispositivo tipo PLA organizada en torno de un arreglo lógico programable mediante campos, con un arreglo AND programable cuyas salidas alimentan a un arreglo OR programable y las salidas del arreglo OR controlan salidas combinatorias [24].

2.3.3.2 CPLD's

Con el avance de la tecnología, ha sido posible producir dispositivos con mas alta capacidad que los SPLD's. La dificultad para incrementar la capacidad de una arquitectura SPLD es que la estructura de los planos lógicos programables crece demasiado en tamaño así como también se incrementa el número de sus entradas. La única forma factible de proporcionar dispositivos de gran capacidad basados en arquitecturas SPLD's, es integrar múltiples SPLD's dentro de un solo chip y proporcionar interconexión a través de programación. Muchos FPLD's comerciales, se encuentran hoy dentro del mercado con esta estructura básica y son llamados CPLD's. Los CPLD's fueron lanzados por Altera en su primer familia de CI's llamada EPLD y posteriormente en tres series adicionales, MAX5000, MAX7000 y MAX9000. A causa de que el mercado crece rápidamente para los grandes FPLD's otros fabricantes desarrollan dispositivos en la categoría de CPLD's y hoy existen muchas opciones disponibles. Los CPLD's proporcionan alta capacidad lógica, equivalente a 50 o más dispositivos SPLD, pero es algo difícil extender estas arquitecturas a densidades mas altas, para construir FPLD's con capacidad lógica muy alta, es necesario utilizar algo diferente como se vera en apartados posteriores [12].

Las capacidades lógicas en CI's de propósito general, hoy están disponibles en los arreglos de compuertas algunas veces llamado MPGA's. Los MPGA's consisten de un arreglo de transistores prefabricados que pueden ser adaptados dentro del circuito lógico por el usuario conectándolos por medio de cables. La adaptación se lleva a cabo durante la fabricación del chip, especificando la interconexión del metal y esto significa que para que un usuario emplee un MPGA se involucra un costo grande de equipo y el tiempo de fabricación es largo, además los MPGA's no son claramente FPLD's[29] y son mencionados aquí porque motivan el diseño mediante programación por el usuario. Por ser el único tipo de FPLD que tiene capacidad lógica muy alta, los FPGA's tendrán la responsabilidad de un mejor cambio en el diseño de circuitos digitales.

Los CPLD's como los ilustrados en la figura 2.5 son similares a los SPLD's excepto que los primeros tienen una escala de integración significativamente más alta. Un CPLD típico es el equivalente de 2 a 64 SPLD's, también contiene desde decenas a unas cuantas centenas de macroceldas.

Generalmente son mejores para diseño orientado a control, debido en parte a su rápida ejecución pín a pín. El amplio fan-in[30] de sus macroceldas permite implementar problemas complejos, como maquinas de estado de alta ejecución.

Los CPLD's proporcionan una ruta de migración natural para los diseñadores de SPLD's que buscan mayor densidad. Tienen una arquitectura tipo PAL y generalmente 4 o más de éstas dentro de ellos, además de que la mayoría soporta lenguajes de desarrollo[31].

Un grupo de 8 a 16 macroceldas es asociado dentro de un bloque de función más grande como se ve en la figura 2.6, las macroceldas de éste usualmente pueden conectarse entre si.

Algunas de las múltiples variaciones entre arquitecturas CPLD incluyen el número de términos producto por macrocelda, si los términos producto de una macrocelda son prestados o alojados en otra macrocelda y si la matriz de switches interconectados es completa o parcialmente poblada.

Figura 2.5. CPLD's de Xilinx.

[29] *Tal como los MPGA's, los FPGA's constan de un arreglo de elementos de circuito, llamados bloques lógicos y recursos de interconexión, sin embargo la configuración de un FPGA es realizada a través de programación por el usuario final.*
[30] *Representa el número de Compuertas que se puede conectar a un pín de entrada de un circuito integrado, con el incremento en la cantidad se degradan las propiedades estáticas y dinámicas.*
[31] *ABEL, CuPL, PALASM.*

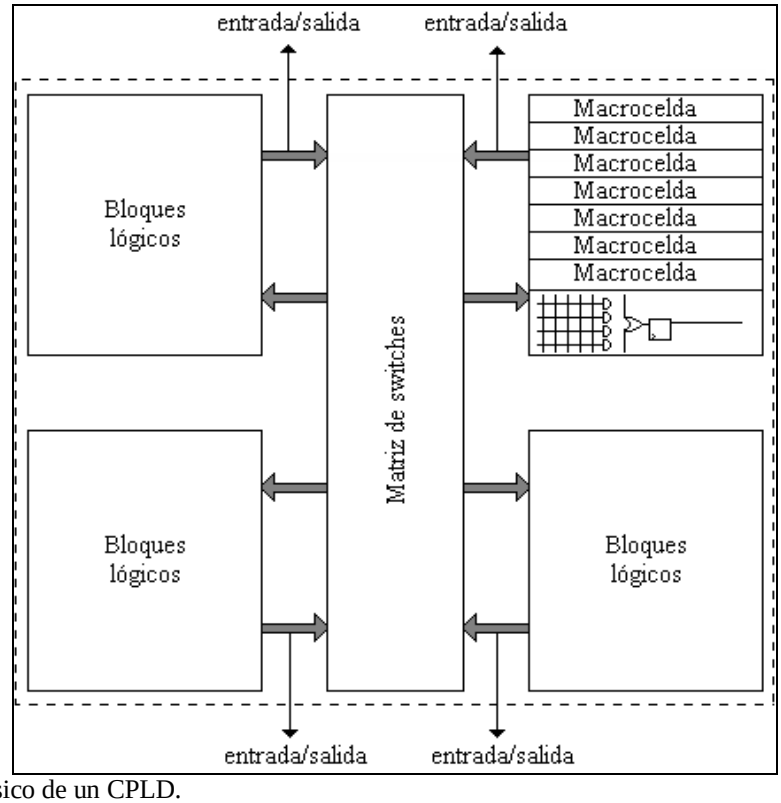

Figura 2.6. Esquema básico de un CPLD.

En algunas arquitecturas, cuando el número de términos producto requerido excede el número disponible en la macrocelda, términos producto adicionales son prestados por macroceldas adjuntas, aunque si una macrocelda adjunta presta términos producto, ésta puede que ya no sea útil, pero puede aun conservar algunas funciones básicas. El préstamo de términos producto usualmente incrementa el retardo de propagación. Otra diferencia en las arquitecturas es el número de conexiones dentro de la matriz de switches.

Una matriz de switches que mantiene todas las posibles conexiones está completamente poblada, una parcialmente poblada tiene libres más switches, pero no todas las conexiones.

El número de conexiones dentro de la matriz de switches determina que tan fácil se colocará un diseño dentro de un dispositivo. Con una matriz de switches completamente poblada, un diseño se ruteara incluso con la mayoría de recursos del dispositivo utilizado y con una asignación fija de pines de entrada / salida. Generalmente, los retardos dentro de una matriz de switches completamente poblada son fijos y predecibles [12], [26], [27].

Un dispositivo con una matriz de switches parcialmente poblada puede tener problemas para rutear diseños complejos, además, puede tener dificultades para realizar cambios de diseño en estos dispositivos sin utilizar un pín de salida diferente, por ello es importante realizar el ruteo a un pín fijo, ya que es más fácil realizar cambios internos a un dispositivo lógico programable que rediseñar una tarjeta de circuito impreso. Los retardos dentro de una matriz de switches parcialmente poblada no son fijos y menos fácilmente predecibles, similar a la mayoría de dispositivos FPGA's. Aunque una matriz de switches parcialmente poblada tiene algunas limitaciones potenciales, es más barato fabricarla [27].

Los CPLD's se fabrican utilizando uno de los tres procesos tecnológicos siguientes: EPROM, EEPROM o FLASH. Los CPLD's basados en EPROM usualmente son programables una vez, a menos que estén dentro de un encapsulado de ventana borrable por luz ultravioleta. Generalmente, los CPLD's son CMOS y utilizan celdas de memoria no volátiles tales como EPROM, EEPROM o FLASH para definir la funcionalidad. Muchas de las familias de CPLD's recientemente introducidas utilizan una EEPROM o FLASH y han sido diseñados para que puedan ser programados dentro del circuito [12], [15], [27].

Los CPLD's también se conocen como:

- Dispositivo lógico programable eléctricamente. (**EPLD,** *Electrically Programmable Logic Device*)
- Lógica borrable y programable eléctricamente. **(PEEL,** *Programmable Electrically Erasable Logic*)
- Dispositivo lógico programable y borrable eléctricamente. **(EEPLD (***Electrically Erasable Programmable Logic Device*).
- Matriz de arreglos múltiples **(MAX,** *Multiple Array Matrix*).
- Súper PAL y Mega PAL.

2.3.3.3 FPGA's

Un FPGA [32] como el que se muestra en la figura 2.7, es un arreglo de bloques lógicos programables colocados en una infraestructura de interconexiones programable, en donde es posible programar la funcionalidad de los bloques lógicos, las interconexiones entre bloques y las conexiones entre entradas y salidas.

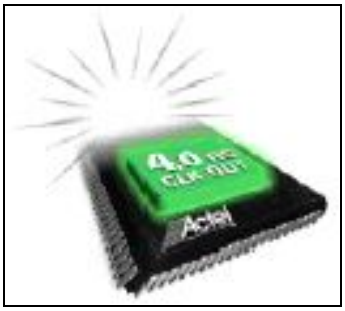

Figura 2.7. FPGA de Actel.

[32] *Una descripción general de un FPGA lo define como un dispositivo programable con un arreglo interno de bloques lógicos, rodeado por un anillo de bloques programables de entrada / salida, conectados a través de interconexión programable.*

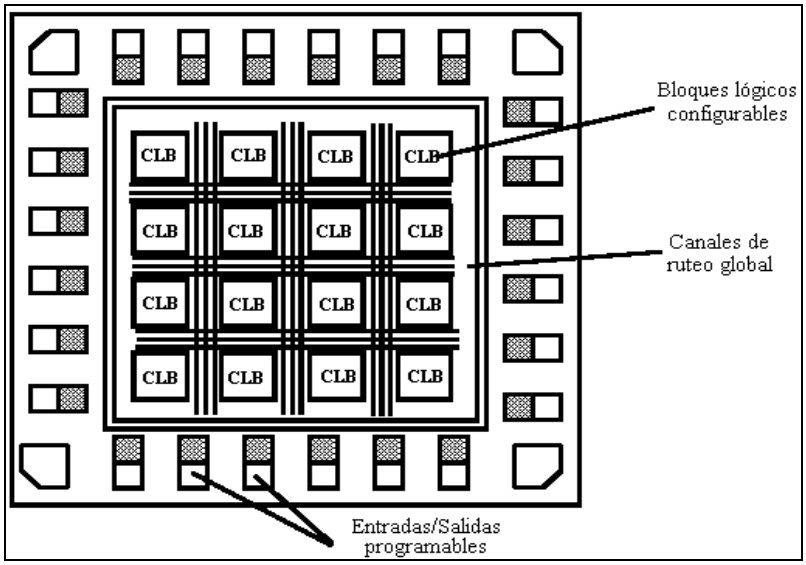

Figura 2.8. Arquitectura básica de un FPGA.

Un esquema básico de la arquitectura FPGA se ilustra en la figura 2.8. Existe una amplia variedad de subarquitecturas dentro de los FPGA's. El secreto para la densidad y ejecución en estos dispositivos recae en la lógica contenida en sus bloques lógicos y en la ejecución y eficiencia de su arquitectura de ruteo. Un FPGA es programable a nivel hardware, por lo que proporciona las ventajas de un procesador de propósito general y un circuito especializado. Un FPGA típico contiene desde 64 hasta decenas de miles de bloques lógicos y un número más grande de flip-flop's.

La mayoría de FPGA's no proporcionan el 100% de interconexión entre bloques lógicos para que no sean tan caros, en lugar de esto, mediante software sofisticado se coloca y rutea la lógica en el dispositivo, tal y como lo hace un autoruteador PCB[33] con los componentes de un diseño [28], [29], [30].

Un bloque lógico configurable (CLB, *Configurable Logic Block*) contiene tablas de búsqueda (LUT's, *Look Up Tables*) para la implementación de lógica digital (con n entradas a la tabla, existen 2^n valores de búsqueda). Los puntos de interconexión programable (PIP's, *Programmable Interconnection Point*) crean conexiones dentro del circuito y entre celdas lógicas y los multiplexores, controlan entradas y salidas.

Un FPGA puede estar conectado a un microprocesador externo y usado para tareas específicas. La aplicación de los FPGA's va más allá de la implementación de lógica digital, éstos pueden ser utilizados para la implementación de arquitecturas específicas. Los sistemas basados en FPGA's proporcionan un mejor desempeño que sus correspondientes implementaciones en software. Las aplicaciones que requieren de un gran número de operaciones simples son adecuadas para su implementación en FPGA's puesto que un elemento de procesamiento puede diseñarse para efectuar esta operación y varias instancias de éste pueden reproducirse para llevar a cabo procesamiento paralelo. Los FPGA ofrecen los beneficios de los arreglos de lógica programable y arreglos de compuertas. De manera similar a los MPGA's, los FPGA's implementan miles de compuertas lógicas en un CI además de que son programados por los diseñadores reduciendo los tiempos y costos. Estas ventajas han hecho que los FPGA's se popularicen. El desarrollo con FPGA's dirige aplicaciones en nuevos caminos como:

[33] *Programa de software, capaz de interconectar componentes electrónicos para su posterior implementación en una placa de circuito impreso.*

- **Implementación instantánea.** Debido a que el cableado en memoria es usualmente corto, los diseños pueden ser implementados "instantáneamente" de manera efectiva. Esto es una distinción del convencional diseño VLSI, donde toma varias semanas desde el diseño a la fabricación.
- **Reconfiguración Dinámica.** Con algunas arquitecturas, parte del FPGA puede ser configurado en tiempo de ejecución. Los circuitos podrían incluso ser auto modificados.
- **Seguridad del Diseño.** La configuración de un FPGA desaparece cuando éste deja de ser alimentado. Existe un rango de aplicaciones donde este nivel adicional de seguridad del diseño es importante. Sin embargo existe otra familia de FPGA's donde la configuración puede programarse una sola vez permaneciendo fijos.

Los primeros FPGA's disponibles comercialmente contenían solamente pequeños números de celdas lógicas y tardaban segundos en reconfigurarse, hoy en día un FPGA contiene cientos de miles de celdas lógicas y puede ser reconfigurado solamente en microsegundos [12], [17].

Aparición de los FPGA's

Con la aparición de los Circuitos integrados de aplicación específica (ASIC's, *Application Specific Integrated Circuit*), el rango en las expectativas de diseño aumento y los ingenieros electrónicos fueron habilitados para incluir dispositivos específicos dentro de sus aplicaciones. Los dispositivos ASIC's, debido a su alto riesgo y costo, nunca alcanzaron el éxito que se esperaba a principios de los 80's, sin embargo, teniendo como base las tecnologías de diseño y proceso de circuitos integrados ASIC nacieron los FPGA's en 1985 fabricados por Xilinx. Este tipo de FPLD's es capaz de implementar lógica de múltiples niveles cuya funcionalidad es establecida por el diseñador al tiempo de desarrollo, en lugar de ser fijado físicamente durante el proceso de fabricación, como sucede con los CI's y los ASIC's [15].

El bajo costo de los FPLD's comparado con los ASIC's en volúmenes de producción bajo y medio, el poco tiempo de producción y desarrollo y su flexibilidad han hecho que sean ampliamente utilizados para diferentes propósitos: como el reemplazo de dispositivos SSI y MSI, además del desarrollo de prototipos inmediatos, volúmenes de producción medianos y bajos ó solamente para obtener mas flexibilidad a nivel del sistema y diseño. Mientras que los procesadores permiten el uso de la misma arquitectura para varias aplicaciones a través del uso de programas adaptados (flexibilidad del software), los FPLD's permiten el desarrollo de dispositivos de aplicación específica, para que los ingenieros sean capaces de adaptar arquitecturas al nivel de hardware. Esta alternativa proporciona un buen intercambio de costo/rendimiento entre las soluciones extremas: sistemas de propósito general (la mayor flexibilidad y el menor rendimiento al costo más bajo) o sistemas completos de aplicación especifica (la más baja flexibilidad y el mayor rendimiento a precios altos) [15], [17].

Clases de FPGA's

Existen cuatro clases primarias de arquitecturas de FPGA's.

- **Arreglos simétricos,** en esta clase los bloques lógicos están ordenados dentro de una cuadricula y las interconexiones se encuentran en forma horizontal y vertical dentro de los bloques lógicos.
- **Arreglos con base en líneas**, aquí los bloques lógicos están colocados en líneas y las rutas de interconexión dentro de los canales, abajo y arriba de los bloques.
- **Mar de compuertas,** donde los bloques lógicos están colocados uno al lado del otro y los ruteos corren a través de ellos (tal como un PCB).

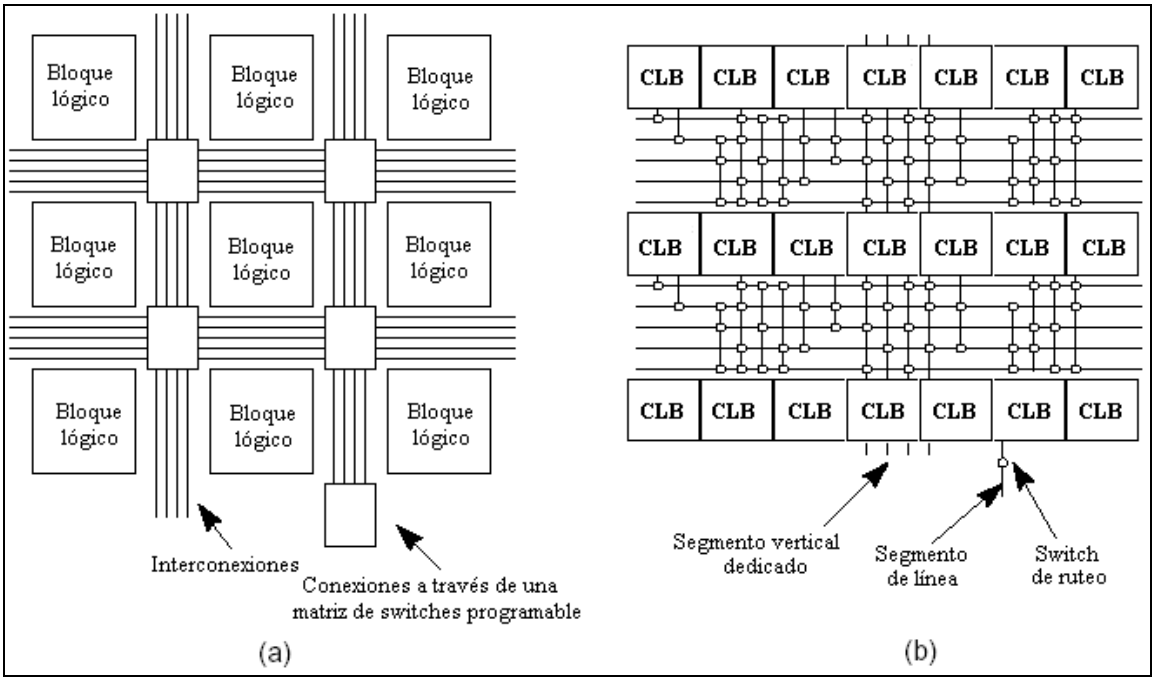

Figura 2.9. Clases de FPGA's. (a) Simétrica, (b) Con base en líneas.

- **PLD's jerárquicos,** donde los dispositivos lógicos programables se interconectan a lo largo de rutas fijas.

Los arreglos simétricos y basados en renglones son dos de las clases más populares y se ilustran en la figura 2.9. Dentro del arreglo simétrico las conexiones están hechas entre metal horizontal y vertical enlazados a través de transistores y dentro de una matriz de switches programables. En la clase basada en renglones la interconexión del bloque lógico es alcanzada con líneas horizontales corriendo a través de los canales de interconexión con switches de ruteo para enlazar a las conexiones verticales [31, 34].

También se puede clasificar las arquitecturas de acuerdo al tipo de granulado, y se clasifican en:

- Granulado ordinario *(Coarse grained*)
- Granulado fino *(Fine grained*)

Las arquitecturas de granulado ordinario consisten de bloques lógicos grandes, frecuentemente contienen dos o más LUT's y dos o más flip-flop's. En la mayoría de estas arquitecturas, una tabla de cuatro entradas de búsqueda (como una ROM de 16x1) implementa la lógica actual. El bloque lógico más grande usualmente corresponde a la ejecución mejorada [32].

Los dispositivos de granulado fino, tienen un gran número de bloques lógicos simples. Los bloques lógicos usualmente contienen una función lógica de dos entradas o un multiplexor de 4 a 1 y flip-flop's. Estos dispositivos son buenos en funciones sistólicas y tienen algunos beneficios para diseños creados por síntesis lógica. [32]

Los FPGA's también se conocen como:

- Arreglo de celdas lógico **(LCA,** *Logic Cell Array*)
- **FLEX, APEX** (FPGA de Altera)
- **ACT** (FPGA de Actel)
- **ORCA** (FPGA de Lucent)
- **Virtex** (FPGA de Xilinx)
- **PASIC** (FPGA de Quick Logic)

Figura 2.10. FPIC de Aptix.

2.3.3.4 FPIC's

Un FPIC[34] es un dispositivo recientemente lanzado al mercado que actúa como una matriz de switches basada en tecnología SRAM o antifusible. Este tipo de dispositivos se puede reconfigurar dinámicamente, de la misma forma como un FPGA estándar basado en SRAM.

Debido a que cada FPIC tiene aproximadamente 1000 pines como el que se muestra en la figura 2.10, un circuito típico requiere solamente unos cuantos dispositivos. Aptix ha desarrollado una aplicación basada en este tipo de dispositivos en la forma de una tarjeta de desarrollo reconfigurable, lo que permite conectar efectivamente cualquier punto de la tarjeta a otro punto o puntos.

I-Cube proporciona dispositivos creados con tecnología antifusibles como componentes stand-alone[35] [37].

2.4 Flujo de diseño

Existen diversos procedimientos aplicados al flujo de diseño de un circuito configurable, los cuales dependen principalmente de las herramientas CAD que cada fabricante ha desarrollado, aunque estos procedimientos tienen mucho en común, no existe un flujo de diseño estandarizado; en este apartado se presenta una forma de diseño lo más genérica posible.

Una vista global del flujo de diseño, sin tener presente el software del fabricante del dispositivo que se utilizará en este proyecto se ilustra en la figura 2.11.

La mayoría de los fabricantes de FPLD's proveen software para la implementación de hardware digital conservando como fundamento el diagrama ilustrado incluyendo a los fabricantes de ASIC's, entre otros. Las etapas que involucra el flujo de diseño son:

1. Diseño inicial.
2. Síntesis lógica.
3. Implementación.
4. Verificación.
5. Programación.

[34] *FPIC es una marca registrada de Aptix, sin embargo no es la única compañía que desarrolle este tipo de dispositivos, también existe I-Cube.*

[35] *Los diseños stand-alone sirven como ejemplos de como utilizar las capacidades periféricas de una tarjeta que contiene FPGA's. Muchos de estos diseños incluyen módulos que pueden ser usados de forma separada dentro de un diseño de usuario.*

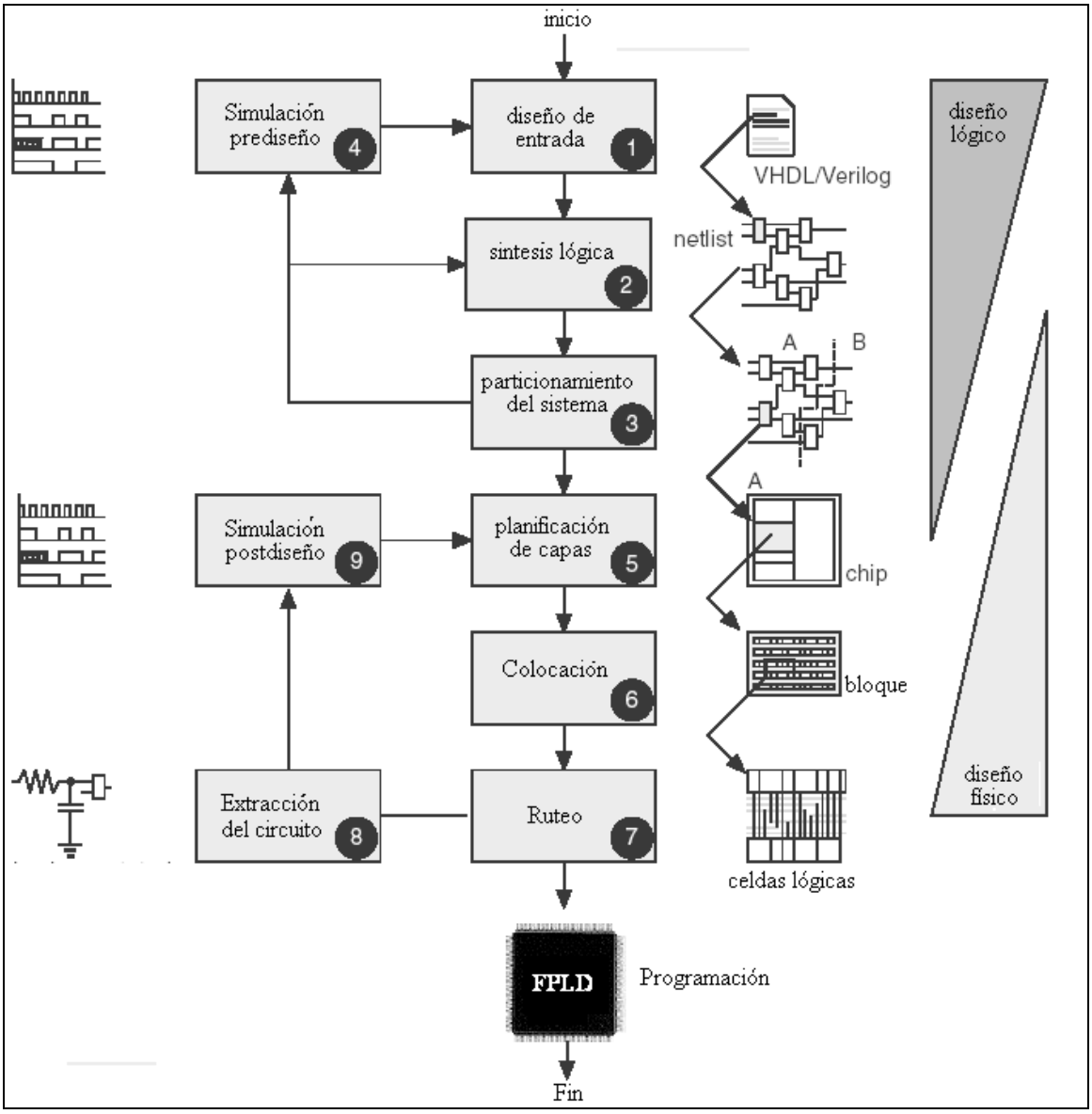

Figura 2.11. Flujo de diseño

2.4.1 Diseño inicial

Existe una gran variedad de herramientas disponibles para realizar el diseño inicial. Algunos diseñadores prefieren usar su software favorito para el desarrollo de esquemáticos ó diagramas de flujo, mientras que otros prefieren usar un lenguaje de descripción de hardware tal como Abel, VHDL o Verilog, otros prefieren mezclar ambos en el mismo diseño y debido a ello existe una lucha para decidir cual es mejor. Tradicionalmente, las herramientas basadas en esquemáticos proporcionan experiencia a los diseñadores para tener más control sobre la colocación y partición de lógica en el dispositivo. Sin embargo, las herramientas basadas en lenguaje permiten un diseño mas rápido, muchas veces sacrificando ejecución o densidad.

En resumen, para realizar el modelado de un sistema digital existen 2 métodos principales:

- **Diseño y captura del esquemático manualmente**, en el cual es necesario dibujar e interconectar los elementos estructurales (compuertas, flip-flops, multiplexores, etc.). El tipo de diseño que se puede implementar puede ser combinaciónal o secuencial y las liberarías se actualizan de acuerdo al diseñador.
- **El diseño basado en HDL** se puede describir en forma textual usando construcciones a través de programación, con herramientas adicionales[36]. Sin embargo las herramientas anteriores no son todas, ya que en este tipo de diseño se puede realizar compilación y síntesis del mismo, esto produce una lista de conexiones entre compuertas, también optimiza el diseño minimizando el área utilizada, la velocidad, potencia, etc. Y finalmente la simulación y verificación aseguran que el diseño realice lo planeado [38].

2.4.2 Síntesis lógica

El proceso de síntesis consiste en reducir el número de dispositivos lógicos interconectados con la finalidad de obtener una lista de conexiones reducida y un tiempo de ejecución menor.

La síntesis para diseño basado en lenguaje tiene mejoras significativas en los últimos años, especialmente para el diseño de los FPGA's, además para llevar a cabo una buena síntesis es necesario aprender la arquitectura y el manejo de las herramientas que ayudan a crear mejores diseños.

Sin embargo es posible realizar síntesis del diseño aun sin conocer la arquitectura profundamente, esta es otra de las ventajas que el diseño basado en código proporciona a los desarrolladores de prototipos.

2.4.3 Implementación

Después de que el diseño se modela y sintetiza, está listo para la implementación sobre la tarjeta del dispositivo. Los niveles 3, 5, 6, 7 y 8 ejemplificados en la figura 2.11 se localizan en la etapa de implementación.

El primer paso es convertir el diseño a un formato soportado internamente por las herramientas. La mayoría de herramientas leen formatos de listas de conexiones estándar y el proceso de interpretación es usualmente automático.

Después de que la lógica se particiona dentro de los bloques lógicos, el software de implementación busca la mejor localización para colocar los bloques lógicos considerando todas las posibilidades. La principal meta es reducir la cantidad de recursos consumidos en el ruteo y maximizar la ejecución del sistema. Esto termina en operaciones intensivas de cómputo para los FPGA's y para los CPLD's.

En la fase siguiente el software de implementación monitorea la longitud del ruteo y resuelve congestiones de las vías mientras coloca los bloques[37]. Cuando el proceso de ruteo y colocación está completo, el software crea el archivo de programación binario que será utilizado en la configurar del dispositivo.

[36] *Las colocaciones de los flip-flops y elementos básicos en el diseño son realizados automáticamente por herramientas CAD, esto es diseño semiautomatizado.*

[37] *En algunos sistemas el software de implementación además rastrea los retardos de ruta absolutos para que el usuario conozca las restricciones de tiempo especificadas, el proceso imita a una tarjeta de circuito impreso colocando y ruteando.*

Las sub etapas comprendidas aquí se realizan de la siguiente forma:

- Particionamiento del sistema ***(Partitioning*)**. Una vez interpretado, las herramientas realizan un chequeo obligado del diseño y optimización sobre la lista de nodos entrante. Después el software particiona el diseño dentro de los bloques lógicos disponibles del dispositivo, la partición es un paso importante para FPGA's y CPLD's. Una buena partición repercute en un mejor ruteado y una mejor ejecución para los FPGA's e incrementa la densidad y ejecución de los CPLD's.
- Mapeo o Planeación de la superficie (***Floorplanning***). Una vez que se simuló el diseño, se procede a mapear el netlist[38] 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 sintetización, 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, según sea el mejor caso.
- Extracción o Traducción (***Translate***). Es la extracción de características del circuito, determinando la resistencia y capacitancía 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, sin embargo, algunas herramientas incluyen editores para realizar las conexiones de modo personalizado.

En aplicaciones grandes o complejas, el software no es tan hábil para colocar y rutear con éxito el diseño. Algunos paquetes permiten al software intentar opciones diferentes o realizar muchas iteraciones para obtener un diseño de ruteo cercano al óptimo. Generalmente, se intenta utilizar menos del 85% de los recursos disponibles del dispositivo. Esta técnica proporciona recursos extras al software para ayudarlo a rutear el diseño. Además, algunos fabricantes proporcionan herramientas de planificación para agregarlas en el esquema físico. El esquema es especialmente importante para FPGA's más grandes debido a que algunas herramientas tienen problemas en el reconocimiento de la estructura del diseño. Una buena herramienta de planeación permite al diseñador llevar esta estructura al software de colocación y ruteo.

2.4.4 Verificación

La verificación del diseño ocurre en varios niveles y pasos a través del diseño (niveles 4 y 9 en la figura 2.11). Hay unos diversos tipos fundamentales de verificación aplicados a la lógica programable. La simulación funcional se realiza en conjunción con el diseño inicial, pero antes de colocar y rutear, verifica la correcta funcionalidad de la lógica. La completa simulación de la coordinación deberá esperar hasta después de los pasos de colocación y ruteo.

[38] *Un netlist es un archivo que registra la descripción de las celdas lógicas (primitivas) y sus conexiones específicas, que conforman un circuito. El formato de netlist más utilizado es EDIF (Electronic Design Interchange Format - Formato de Intercambio de Diseño Electrónico).*

Después de colocar y rutear, el software rescribe la lógica y los retardos de ruteo a la lista de nodos para simulación. Los dos tipos de simulación se especifican a continuación:

- Simulación Prediseño (***Prelayout Simulation***). También conocida como *Preruteado*. Hasta el segundo paso aún no se ha comenzado con el *diseño físico*, solamente se ha realizado el *diseño lógico* del circuito modelado. La mayoría de los ambientes de desarrollo incluyen dos tipos de simulación, una llamada *lógica* y la otra *física*. La diferencia radica en que la primera es antes del ruteo o trazado (interconexión de líneas) y la otra es posterior. La simulación lógica solamente permite observar el comportamiento del diseño con las celdas lógicas más básicas del dispositivo seleccionado.
- Simulación Postdiseño (***Postlayout Simulation***). 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.

Debido a que los cambios en el diseño típicamente involucran miles de dólares extras en costos de ingeniería no recurrente y semanas de trabajo, la simulación completa del diseño es una parte esencial en las mejores herramientas desarrolladas para el diseño de MPGA's y ASIC's, entre otros circuitos.

Sin embargo, en un FPLD los cambios llevan de minutos hasta horas con un costo pequeño o sin costo, además de que no se encuentra un error a nivel de silicio.

Una técnica exitosa para el diseño de lógica programable es simular el diseño para garantizar una funcionalidad apropiada, verificar el tiempo usando una calculadora de coordinación estática y después verificar la completa funcionalidad evaluando el diseño dentro del sistema.

Los FPLD tienen distintas ventajas sobre los arreglos de compuertas, además de que los cambios son prácticamente fáciles. Con dispositivos Programables dentro del sistema (ISP, *In System Programmable*), tal como FPGA's basados en tecnología SRAM y CPLD's ISP, los cambios son posibles aun y cuando los chips estén montados sobre el sistema. Usando técnicas de verificación dentro del sistema los diseños son comprobados a su máxima velocidad, utilizando los recursos de hardware y software.

Algunos de los fabricantes de dispositivos proporcionan capacidades de depuración dentro del sistema. Por ejemplo, Xilinx proporciona un pequeño cable llamado Xchecker que se conecta al puerto serial y permite bajar rápidamente un diseño. Con unos cuantos cambios al diseño y a la tarjeta, el cable Xchecker es capaz de parar el reloj o de llevarlo paso a paso además de releer el estado de los flip-flops.

También, la tecnología de prueba de acción de Actel proporciona acceso a nodos internos dentro de sus FPGA's basados en tecnología antifusible.

2.4.5 Programación

Después de crear un archivo que puede ser utilizado en la configuración de un dispositivo, el dispositivo programable puede ser configurado y habilitado para su funcionamiento. El método de configuración depende de la tecnología utilizada en la tarjeta, la mayoría de tecnologías lógicas programables, incluyendo las PROM's para FPGA's basadas en tecnología SRAM, necesitan de un dispositivo programador. Por una cuota nominal, un distribuidor puede programar una mediana cantidad de dispositivos, pero si lo que se requiere es un bajo volumen de dispositivos, es mejor tener un dispositivo programador de laboratorio.

Los dispositivos programables dentro del sistema, incluyendo FPGA's basados en tecnología SRAM, no requieren un programador físico, pero requieren sistemas inteligentes para bajar el archivo de programación dentro del dispositivo. Esto es realizado con un microprocesador, un microcontrolador, una EEPROM o a través de un puerto de evaluación JTAG.

2.5 VHDL

Una alternativa para modelar diseños en un FPGA es utilizar HDL's[39], los más conocidos son VHDL y Verilog ya que se trata de lenguajes estandarizados; sin embargo no son la única opción, existen otras alternativas como Handel-C y Abel, entre otros. VHDL fue desarrollado como un lenguaje para el modelado de sistemas digitales. Proporciona una sintaxis amplia y flexible que permite el modelado estructural, en flujo de datos y de comportamiento de hardware. VHDL está regido bajo el estándar IEEE 1076-1993, lo que favoreció su adopción en la industria y se ve reflejado en las constantes mejoras de sus herramientas; debido a su estandarización, un código en VHDL puede ser portable a diferentes herramientas y también puede ser reutilizado en diferentes diseños [39], [40].

VHDL es un lenguaje de descripción de hardware que se utiliza para modelar, documentar, simular, verificar y sistematizar un sistema digital. Por tanto abarca el ciclo completo de diseño, salvo el trazo físico o layout, desde las especificaciones iniciales hasta la construcción del prototipo hardware. Proporciona soporte suficiente para especificar su comportamiento o su estructura, incluyendo jerarquías. Asimismo, es útil para metodología de diseño ascendente[40] pero sobre todo descendente[41]. La semántica y construcciones del lenguaje permiten también diseñar con facilidad bancos de prueba (*test-benches*), mediante los que se llevan acabo la simulación de los sistemas modelados.

En esta descripción de los HDL's no se profundizara, solo se pretende mencionar las características mas importantes de VHDL, algunos detalles de este lenguaje quedaran inmersos en el desarrollo del diseño, posteriormente.

2.5.1 Historia

En 1980 el Departamento de Defensa (DoD, *Department of Defense*) de Estados Unidos inicio un proyecto denominado Circuitos integrados de muy alta velocidad (VHSIC, *Very High Speed Integrated Circuit*), con el principal objetivo de desarrollar circuitos integrados en tecnología de 0,5 micras con muy altas prestaciones y resistencias a la radiación. Estos circuitos se habrían de integrar en los sistemas militares y mejorarlos en gran medida [41].

Antes del término de ese mismo año se hizo evidente que para poder organizar y coordinar el desarrollo de los 28 circuitos integrados propuestos por diversas compañías, era necesario el empleo de un lenguaje de descripción de hardware que permitiera el flujo de información entre diseñadores, fabricantes y usuarios. De esta forma, en otoño de 1980, se iniciaron los trámites para el desarrollo de un nuevo lenguaje de descripción denominado "Lenguaje para diseño y descripción de hardware VHSIC o VHDDL", que posteriormente se simplificaría en el acrónimo VHDL.

---

[39] *También se utilizan herramientas que permiten realizar esquemáticos o diagramas de estado, sin embargo, para diseños complejos son mas tediosos y no compiten con la flexibilidad de los HDL's.*

[40] *Esta forma de diseño se realiza a partir de un prototipo formado por módulos sencillos y la comprobación de su funcionamiento antes de proceder a la integración de múltiples módulos para constituir un bloque funcional más complejo.*

[41] *En este tipo de diseño es necesario simular y verificar el correcto comportamiento de los circuitos integrados antes de integrarlos, mediante métodos de diseño asistidos por computadora.*

Con la participación de tres compañías, Intermetrics, Texas Instruments e IBM, en julio de l983 se inicio formalmente el proyecto de desarrollo del lenguaje VHDL. El documento de partida del DoD especificaba que el VHDL debería ser un lenguaje para diseño y descripción del hardware y concretamente para ser utilizado en:

- **Documentación del diseño.** En principio, VHDL se estandarizó para la descripción del hardware, pero no para diseño.
- **Diseño en alto nivel**.
- **Simulación**.
- **Síntesis**.
- **Verificación.** Descripción de entradas (netlist) para la herramienta de diseño físico.

Otros requerimientos del DoD, para un modelo eficiente fueron [41]:

- **Descripción genérica de modelos.** De tal forma que resultaría sencillo configurar un componente en cuanto a tamaño, características físicas temporales, fan-out[42], etc. Para ello se utilizan los denominados puertos genéricos.
- **Declaración y uso de tipos de datos.** Debido a los diversos niveles de abstracción posibles, el lenguaje no se puede restringir a los tipos más básicos, como bít o booleano. Por tanto define también tipos enteros, reales, físicos, enumerados, array, record, etc. y permite al usuario la definición de cualquier otro. Es por ello que se dice que VHDL, está fuertemente orientado a tipos y es una de las características que le otorgan mayor potencial y flexibilidad.
- **Subprogramas.** Se permite la declaración y definición de funciones y procedimientos para conversiones de tipos, redefinición de operadores, creación de otros nuevos, entrada y salida de datos desde el exterior y otras tareas comunes a los demás lenguajes de propósito general.
- **Control temporal.** VHDL dispone de sentencias para detectar flancos, especificar retardos, especificar tiempos de set-up y hold, comprobar anchura de pulso, establecer restricciones temporales, etc.
- **Descripción estructural**. Los requerimientos del DoD para un HDL estándar especificaba que éste debería tener recursos para especificar el hardware a todos los niveles, incluso llegar a describir un diseño genérico de un bít y utilizarlo para descripción de estructuras regulares multibít en una o más dimensiones.

La primera fase del desarrollo del lenguaje finalizo en julio de 1984 y en ese mismo año el IEEE comenzó a trabajar en la estandarización. En 1985 apareció el primer prototipo de lenguaje y en 1987 se aprobó finalmente el estándar con el numero 1076. La aparición del mismo supuso un fuerte impulso y numerosas firmas de herramientas Automatización mediante diseño electrónico (EDA, *Electronic design automation*) comenzaron a incorporar compiladores y simuladores en sus paquetes de diseño a partir de 1990. Posteriormente llegaron las herramientas de síntesis. En 1994 el instituto de ingenieros eléctricos y electrónicos (IEEE, *Institute of Electrical and Electronic Engineers*) publicó la revista del estándar IEEE Std 1076-1993 que es la que se encuentra actualmente en vigor.

[42] *El fan-out representa la cantidad de dispositivos que puede alimentar o controlar un pín de salida de un CI, generalmente este parámetro es especificado por el fabricante.*

2.5.2 Niveles de descripción

VHDL es un lenguaje descriptor de hardware de gran generalidad derivada del lenguaje de alto nivel ADA[43]. Dispone de tipos abstractos para definir el formato y valores de señales, variables, constantes, etc. y proporciona amplias facilidades para la realización de algoritmos. Los diferentes niveles de descripción que maneja VHDL son los siguientes:

- **Nivel algorítmico.** Es el nivel con mayor grado de abstracción. Aquí el diseñador solo describe el comportamiento del sistema, sin preocuparse de las señales o componentes internos del mismo. Por ello al referirse a él se suele hablar de nivel de comportamiento o descripción de alto nivel.
- **Nivel de transferencia de registros (RTL,** *Register Transfer Level*). Este nivel proporciona un cierto grado de abstracción con respecto al hardware, pero el diseñador se ve obligado a describir las distintas señales que interactúan en un circuito y su comportamiento en función de las entradas por medio de ecuaciones lógicas y sentencias de asignación.
- **Nivel lógico.** Utiliza los recursos que el lenguaje proporciona para describir las interconexiones entre los distintos componentes de un circuito. Otra denominación habitual para referirse a este nivel es la de estructural.

De los niveles anteriores, el algorítmico ofrece las más grandes ventajas y generalmente es el más empleado, ya que no necesita de saber detalles específicos de las arquitecturas a programar, entre otras cosas.

2.5.3 Etapas básicas en el proceso de diseño

El proceso de diseño se puede dividir en seis etapas bien definidas [41], [42], [43]:

- Definición de los requerimientos del diseño.
- Modelado del diseño en VHDL.
- Simulación del código fuente.
- Síntesis, optimización y ajuste del diseño.
- Simulación "post-layout".
- Programación del dispositivo.

2.5.3.1 Definición de los requerimientos del diseño

Antes de empezar a escribir líneas de código, se debe tener una idea clara de los objetivos y requerimientos del diseño (especificaciones): ¿Qué funcionalidad debe tener el diseño?, esto es, ¿Para que sirve? ¿Cuáles son los tiempos requeridos para la inicialización o la relación reloj-salida? ¿Cuál es la frecuencia máxima de operación? ¿Cuales son los caminos críticos? Responder de forma adecuada a éstas y otras preguntas ayudarán a elegir una metodología de diseño y una arquitectura de dispositivo adecuada.

2.5.3.2 Modelado del diseño en VHDL

A partir de las especificaciones se puede tener la tentación de empezar a escribir líneas de código, pero es recomendable decidir una metodología de diseño. Esto es, elegir la forma en que será descrito, el resultado será un diseño más eficiente.

[43] *Lenguaje de programación basado en procedimientos diseñado bajo la dirección del DOD a finales de la década de los 70's. ADA, llamado así en honor a Augusta Ada Byron, condesa de Lovelace y pionera en el campo de la informática, se desarrolló a partir del Pascal, aunque incluía importantes extensiones semánticas y sintácticas, incluyendo la ejecución simultánea de tareas.*

Existen dos tipos de metodología: ascendente, descendente. La primera implica el crear estructuras jerárquicas, mientras que la última ve el diseño como un todo.

- **La metodología ascendente (*Bottom-Up*)**, Esta metodología tiene la finalidad de formar módulos de aplicación especifica mediante la descripción de elementos básicos o primitivas, que formaran nuevos módulos hasta llegar a uno solo que representa el sistema completo, desde un nivel bajo hasta uno alto de abstracción.
- **La metodología descendente (*Top-Down*)**, Esta metodología parte de un nivel alto de abstracción y detalla los módulos de aplicación específica según se necesite. No es necesario que un modulo que ha alcanzado el nivel primario alcance el nivel de primitivas.

Posteriormente a la decisión de la metodología a aplicar, es posible describir diagramas de flujo, diagrama de bloques, etc. con el lenguaje de descripción elegido. Hay que ser muy cuidadoso con la sintaxis y con la semántica. Un modo de trabajo utilizado por muchos diseñadores consiste en editar un ejemplo y adaptarlo a las necesidades del diseño concreto [41], [42], [43].

2.5.3.3 Simulación del Código Fuente

La simulación de código permite depurar errores funcionales antes de la síntesis, también es conocida como simulación lógica ya que se realiza antes de rutear al dispositivo a diferencia de la simulación post diseño.

2.5.3.4 Síntesis, Optimización y Ajuste del diseño

Síntesis

Se puede definir como la traducción de la descripción de un diseño a una representación de circuito de bajo nivel (netlist)[44]. El proceso de síntesis depende de la tecnología empleada, en otras palabras, el paso de una descripción en VHDL hacia un conjunto de netlist es diferente de un dispositivo a otro. El proceso de síntesis convierte el diseño a una estructura de datos interna, traduciendo el "comportamiento" descrito en alto nivel a una descripción de nivel RTL. La descripción RTL especifica registros, señales de entrada y salida y la lógica combinaciónal entre ellas. Algunas herramientas de síntesis traducen estructuras de datos en funciones lógicas optimizadas según la arquitectura elegida

Optimización

El proceso de optimización depende de tres variables:

- La forma de las expresiones booleanas.
- El tipo de recursos disponibles.
- Las directivas de síntesis utilizadas (tanto automáticas como propias de usuario).

La optimización de una estructura PLD o CPLD implica la simplificación de las expresiones lógicas a una suma mínima de términos producto, además también se optimiza el número de literales. Para ello se utilizan técnicas de simplificación de la forma canónica en una suma de términos producto. La optimización para FPGAs típicamente requiere que la lógica se exprese en factores comunes que se puedan utilizar en diferentes partes del diseño.

[44] *Es el proceso por el cual se crean las netlist o ecuaciones a partir de descripciones de diseño, en principio abstractas.*

Ajuste

El ajuste es el proceso por el que se toma la lógica producida por la síntesis y la optimización y se "coloca" en un dispositivo lógico, transformando la lógica (en caso de ser necesario) para obtener el mejor ajuste. Ajuste es un término utilizado habitualmente para describir el proceso de colocar los recursos en arquitecturas del tipo CPLD.

Cuando la arquitectura es una FPGA el proceso se suele denominar ruteo y colocación, ya que se colocan bloques lógicos en diferentes células del FPGA y posteriormente se interconectan entre sí o hacia bloques de entrada/salida. El proceso de ajuste en un CPLD puede ser complejo, ya que el modo en que la lógica se puede poner en un dispositivo concreto suele ser variado.

2.5.3.5 Programación del dispositivo

En la figura 2.12 se ejemplifica el uso de VHDL para programar un FPLD. Note el uso de cadenas de bits, que corresponden a los estados de los switches en las matrices de ruteo, los switches electrónicos en el FPGA se abren o cierran de acuerdo al bít correspondiente en la cadena de bíts [43].

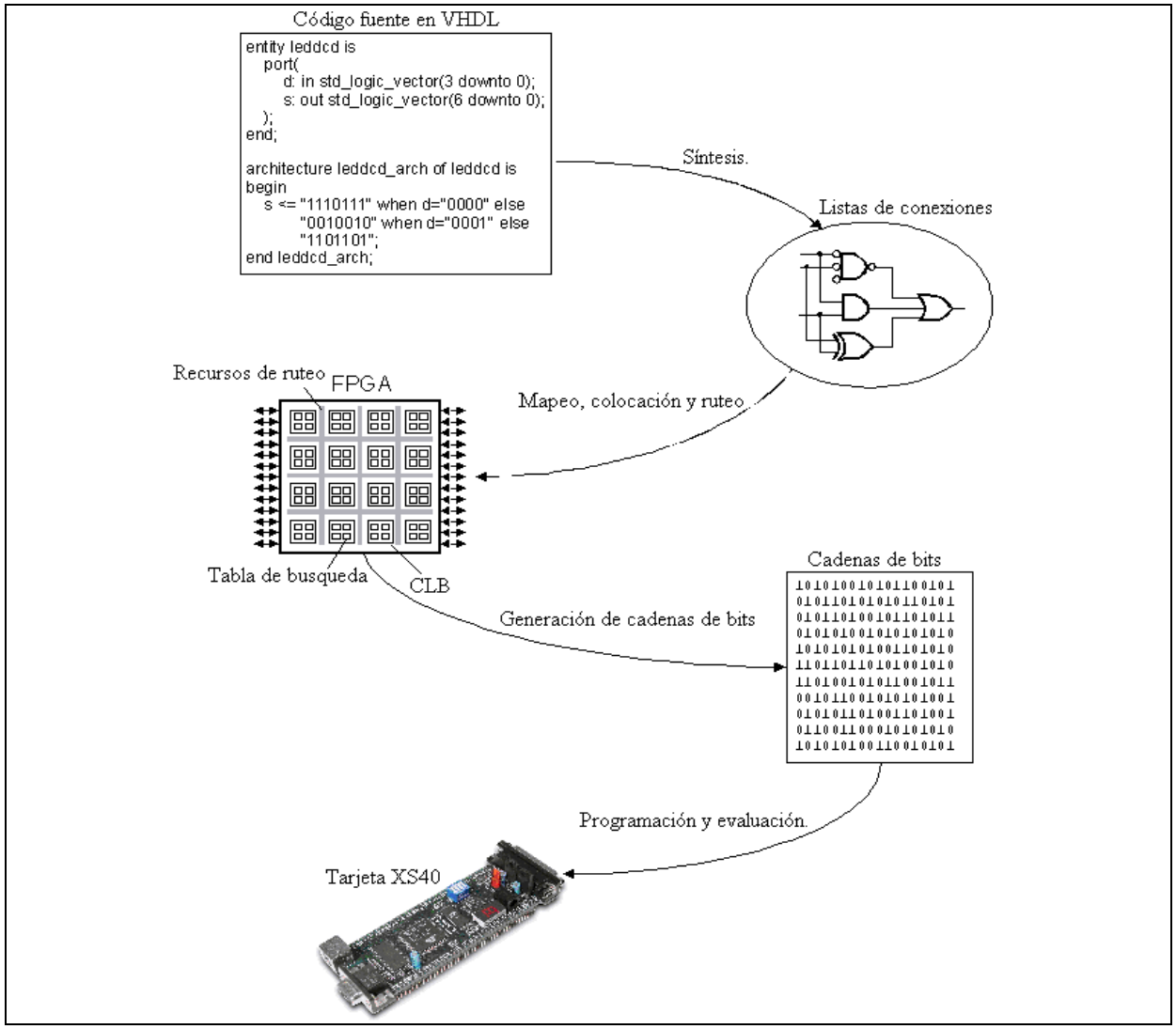

Figura 2.12. Programación mediante VHDL

CAPÍTULO 3

Desarrollo del sistema propuesto

En este capítulo se describe el diseño y la implementación del digitalizador de video propuesto en este trabajo de tesis, teniendo como elemento principal un FPGA de la compañía Xilinx. Para tal propósito, el diseño se ha dividido en dos partes: la primera formada por los elementos analógicos y suya función es el acoplo y tratamiento de la señal de video compuesta para su posterior digitalización, la segunda conteniendo a los elementos digitales encargados de realizar la coordinación, control y el manejo de los datos digitales de la imagen.

Con la finalidad de obtener como resultado una descripción portable entre diversas tecnologías, el modelado del sistema propuesto no considera ninguna arquitectura específica. Para probar dicho modelado, la implementación del diseño fue hecha en dos FPGAs distintos, un dispositivo XC4010XL-EPC84 de la familia XC4000 y un dispositivo XC2S200 PQ208 de la familia Spartan 2 ambos de la compañía Xilinx

Figura 3.1. Digitalizador de video desarrollado por Tetra Picolo.

3.1 Digitalizador de video

Los digitalizadores de video, también conocidos como *frame grabbers*, se fabrican en muchas formas y tamaños con cantidades variantes de funciones incluyendo diferentes conexiones a la computadora, ya que puede ser mediante el bus de la Industria de la arquitectura estándar (ISA, *Industry Standard Architecture*) ó Interconexión de componentes periféricos (PCI, *Peripheral Component Interconnect*), el puerto Bus serial universal (USB, *Universal Serial Bus*), PS/2, Serie ó Paralelo ó un conector RCA, entre otros. En la figura 3.1 se muestra un digitalizador de video desarrollado por la compañía Tetra Picolo, éste se comunica a la PC mediante el bus PCI y recibe video codificado en la norma RS-170 a través de SVideo, RCA o coaxial TV. La mayoría de las compañías [45] en el mercado ofrecen digitalizadores de video con precios generalmente altos, sin embargo ofrecen múltiples opciones de configuración, con la finalidad de realizar aplicaciones con grandes requerimientos especialmente destinadas a grandes empresas.

3.1.1 Clasificación de digitalizadores de video

Debido a la variedad existente en el mercado de digitalizadores de video y considerando sus funciones y/o características principales, se ha adoptado un conjunto de términos mediante los cuales se pueda describir este diseño. Los atributos más importantes de un digitalizador de video se describen en el apéndice B y son los siguientes:

- Capacidad para digitalizar marcos contra campos.
- Capacidades para la digitalización de video de marco inmóvil contra video en tiempo real.
- Resolución.
- Ancho de bits en las muestras.
- Capacidades monocromáticas contra color.
- Capacidad de procesamiento de imágenes sobre tarjeta.
- Capacidades en la salida de video.

[45] *PixelSmart, Ellips, MeB Systems GMBH, Mikrotron, Curtech, Optimumvision, inX Systems, Coreco, Martos, Epix y Scion son algunas de las empresas líderes mundialmente en el mercado de la fabricación de digitalizadores de video. Sus productos manipulan video en diferentes formatos como NTSC, PAL, YUV, RS422, RGB y LVDS, principalmente y son conectados a la PC mediante el bus PCI, ISA o PC104, además de que se pueden controlar bajo diversos sistemas operativos, desde Windows 3.1 hasta XP, OS2 y Linux.*

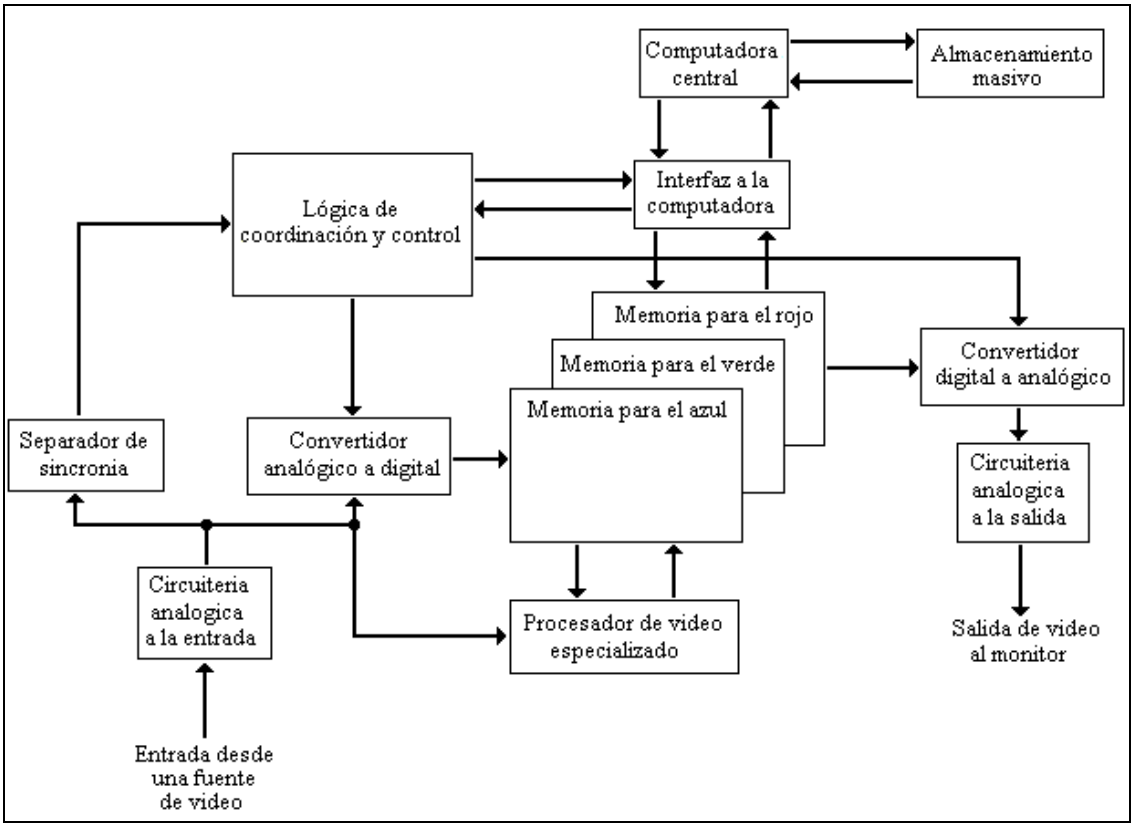

Figura 3.2. Diagrama de bloques de las funciones de un digitalizador de video.

3.1.2 Esquema general de un digitalizador de video

Considerando las propiedades mencionadas anteriormente, se puede ahora realizar un bosquejo de cómo se integra un digitalizador de video con características completas, el cual se muestra en la figura 3.2. Este diagrama a bloques representa a los digitalizadores de video comerciales, sin embargo existen múltiples características que los diferencian. En la página WEB: http://members.planeteer.com/ se muestran algunos de los fabricantes de digitalizadores de video con precios que van desde los $199 hasta $6000 dólares.

3.2 Metodología de diseño e implementación del sistema propuesto

En este apartado se detalla la implementación de un digitalizador de video de video teniendo como elemento central un FPGA y el cual será utilizado en sistemas autónomos, el FPGA será configurado con la descripción de un procesador de video para la adquisición de imágenes. Este digitalizador de video cuenta con funciones necesarias para digitalizar, almacenar y manipular la información de video.

Para el desarrollo en este proyecto de tesis se ha tomado como base la metodología de un sistema empotrado, esta metodología ha sido ajustada, mediante algunos cambios, al desarrollo del sistema propuesto.

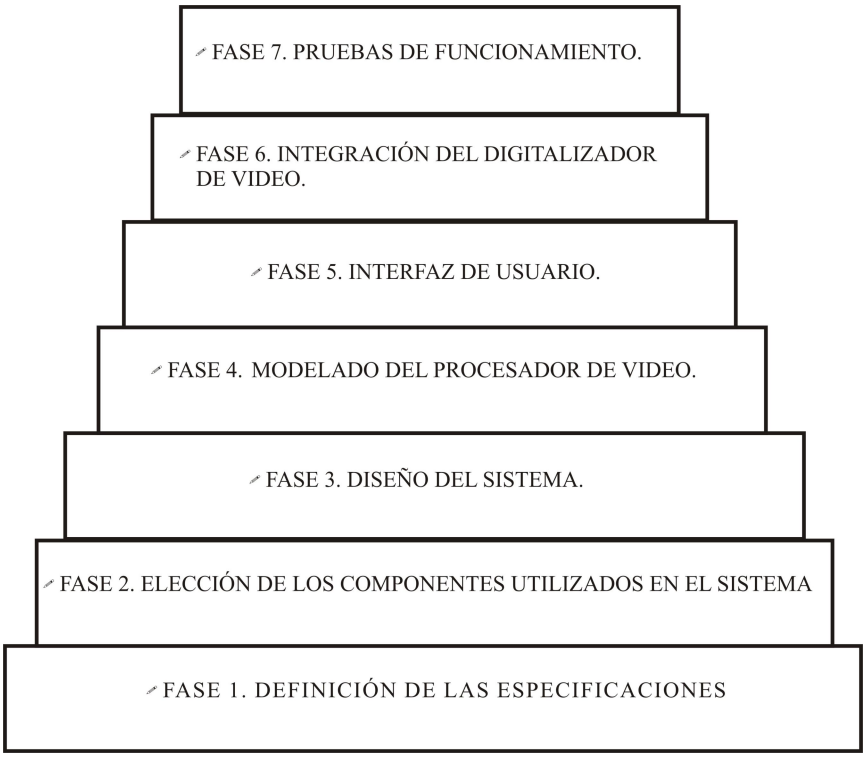

Figura 3.3. Diagrama de la metodología utilizada en el desarrollo del digitalizador de video.

La figura 3.3 muestra las fases que forman esta metodología, y son las siguientes:

Fase 1. Definición de las especificaciones. Limita las capacidades del digitalizador de video en cuanto a los sistemas de video soportado, tamaño de imagen, resolución.

Fase 2. Elección de los componentes utilizados en el sistema propuesto. Aquí se detallan brevemente los CI's más importantes empleados en el digitalizador de video.

Fase 3. Diseño del sistema. El digitalizador de video está integrado principalmente por elementos electrónicos analógicos, digitales y algunos que manejan ambas partes, las subetapas listadas a continuación muestran una forma de clasificación particular y de cómo ha sido dividido el diseño para facilitar su implementación:

Fase 3.1. Procesador de video.

Fase 3.2. Módulo analógico.

Fase 3.3. Módulo digital.

Fase 4. Modelado del procesador de video. El procesador de video implementado en un FPGA de la familia de Xilinx, mediante lenguaje VHDL, realiza la digitalización y almacenamiento de las señales de video, la coordinación y control de las señales de sincronía y la transferencia de imágenes hacia la PC. Los procesos o módulos que complementan esta fase son:

Fase 4.1. Módulo de sincronía.

Fase 4.2. Módulo de control.

Fase 4.3. Módulo de almacenamiento.

Fase 4.4. Módulo para el protocolo RS-232.

Fase 5. Interfaz de usuario. Para verificar el correcto funcionamiento del digitalizador de video, se implementa un programa para recepción de imágenes vía el puerto serie de la PC, este programa permite verificar la digitalización y procesamiento de imágenes.

Fase 6. Integración del digitalizador de video. En esta fase las partes analógica y digital son acopladas para formar al digitalizador de video, el sistema resultante se complementa con el dispositivo optoelectrónico que proporcione la señal de video RS-170 (generalmente una cámara CCD), para adquirir las imágenes enfocadas, digitalizarlas y almacenarlas en el sistema de memoria.

Fase 7. Pruebas de funcionamiento. Finalmente empleando el digitalizador de video se verifican los resultados de las imágenes adquiridas mediante la interfaz con la PC.

3.2.1 Especificaciones del digitalizador de video

Las especificaciones del digitalizador de video a implementar son las siguientes:

- Capacidad para adquirir marcos o campos de cualquier fuente de video monocromática reglamentada por la norma RS-170.
- Dimensión de la imagen a digitalizar: máximo 488 líneas horizontales por 785 líneas verticales.
- La imagen almacenada debe ser en tonos de grises (8 bits por píxel).
- Tener la capacidad de permitir aplicar algoritmos de procesamiento de imágenes sobre el cuadro adquirido.
- Habilitar una interfaz con una computadora personal, la computadora personal únicamente servirá para evaluar el funcionamiento del digitalizador de video.

3.2.2 Componentes empleados para la construcción del digitalizador de video

Los componentes principales empleados en la construcción del digitalizador de video o sujetador de marco se detallan brevemente en la tabla 3.1.

La principal característica que tienen en común estos elementos es que son dispositivos comerciales, por tanto, cuentan con un buen soporte comercial y técnico facilitando el diseño de sistemas que los emplean.

3.2.3. Diseño del sistema propuesto

Con la finalidad de facilitar el diseño e implementación del digitalizador de video, éste ha sido dividido en 3 bloques, los cuales son descritos a continuación.

Tabla 3.1. Elementos importantes del digitalizador de video desarrollado.

Elemento	*Descripción*
ADS802U	Convertidor analógico-digital proporcionado por la compañía Texas Instruments con un rango de operación de muestreo máximo de 10 MSPS y una resolución de 12 bits.
LM1881U	Separador de sincronía de video de la compañía Nacional Semiconductor, extrae información de coordinación de la señal de video incluyendo sincronización compuesta y vertical, coordinación de pórtico anterior e información de los campos par e impar desde una fuente de video NTSC, PAL ó SECAM.
KM684000B	Memoria SRAM de Samsumg fabricada mediante un proceso de tecnología CMOS. Almacena hasta 512K x 8 localidades, también puede operar con baterías de respaldo consumiendo muy poca corriente.
SP233ACP	Controlador y receptor de línea mejorado de Sipex, implementa la interfase del protocolo de comunicaciones RS-232. Ésta integrado de tres bloques de circuitos básicos, un controlador del transmisor, un receptor y un surtidor de carga.
XC4010XL	FPGA perteneciente a la familia 4000 de Xilinx, implementa hardware mediante software, que en su mayoría es lógica de coordinación y control, además del protocolo de comunicaciones RS-232 y generación de señales para la conversión de la señal de video compuesta a datos binarios. Debido a que este FPGA contiene 400 CLBs solo puede implementar los procesos correspondientes a la digitalización de una imagen.
XC2S200	FPGA perteneciente a la familia Spartan II de Xilinx, contiene 1176 CLB's y la frecuencia de operación soporta hasta 200 MHz. Empleando los recursos de este FPGA se pueden implementar los procesos de digitalización de imágenes y además los algoritmos de detección de bordes de Sobel y el de mejora de gradiente direccional de bordes.

3.2.3.1 Modelado del Procesador de video

El procesador de video implementado en un FPGA de Xilinx se divide en módulos o procesos que realizan las operaciones de sincronía y control de las señales de coordinación, implementación de la *Unidad Asíncrona de transmisión y recepción* (UART, *Unit Asynchronuos Recepcion-Transmision*) y generación de la señal de reloj para el funcionamiento del ADC.

El desarrollo detallado de estos módulos se lleva a cabo en la sección 3.2.4

3.2.3.2 Módulo analógico del digitalizador de video

La parte analógica del digitalizador de video corresponde a los dispositivos que forman parte del pre-tratamiento de la señal de video para poder ser digitalizada. Aunque el digitalizador de video contiene CI's que funcionan con partes digital y analógica, se tratará de separar a éstas para desarrollar una mejor explicación del sistema completo, este es el caso del ADC el cual necesita una parte analógica que le proporcione las tensiones de referencia para su adecuado funcionamiento. La figura 3.4 muestra un diagrama a bloques de digitalizador de video, en ella se pueden apreciar los elementos que forman parte del subsistema analógico. Las funciones realizadas por la etapa analógica incluyen:

- Acoplamiento de video con terminaciones de 75Ω.
- Corrección en CD de la señal de video.
- Rechazo a la subportadora de color.
- Almacenamiento parcial de la señal
- Generación de las tensiones de referencia para el ADC.
- Conversión analógica a digital.
- Generación de señales de sincronía.

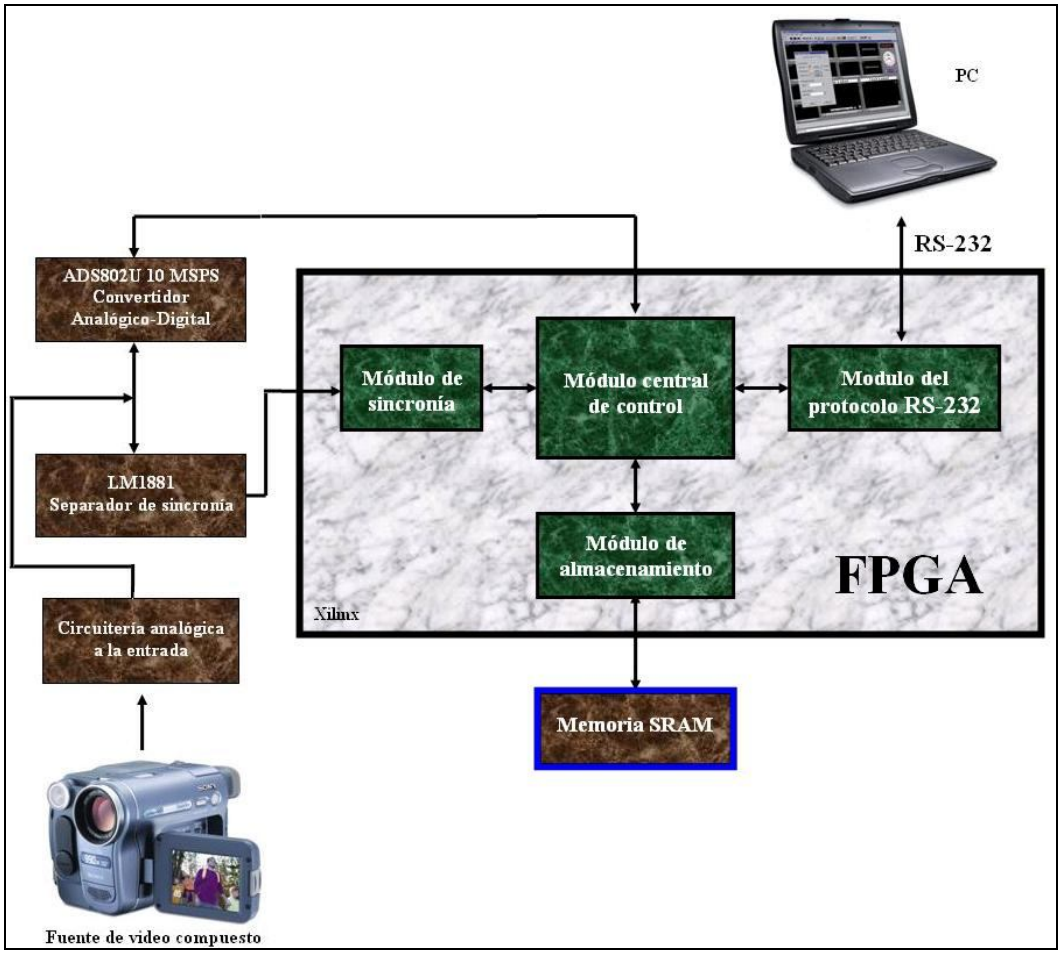

Figura 3.4. Diagrama a bloques del digitalizador de video

3.2.3.2.1 Acoplamiento de la señal de video compuesta

Para realizar un acoplamiento de impedancias apropiada del digitalizador de video con la fuente de video, es indispensable proporcionar la terminación de 75Ω requerida por la señal de video, el resistor R1 cumple con esta función. La resistencia R1 se puede quitar si se conecta la entrada del digitalizador de video en paralelo con un dispositivo de despliegue (televisión, monitor, etc.) el cual proporcionara la terminación apropiada, la figura 3.5 representa la señal[46] de video compuesta acoplada con el diseño.

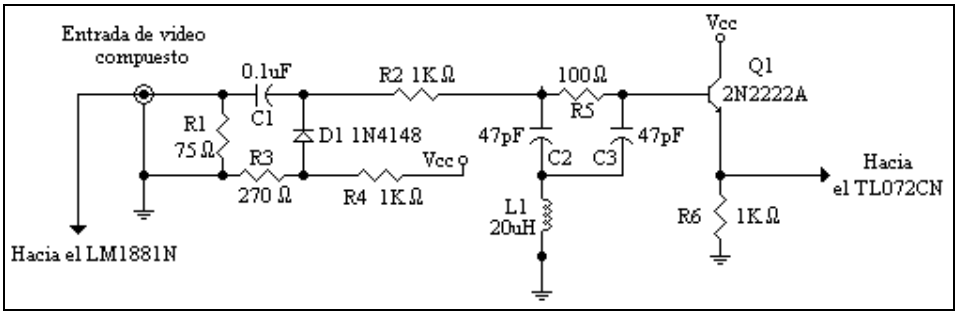

Figura 3.5. Configuración para acoplar y corregir en CD, la señal de video compuesta.

[46] *Ésta proviene de una cámara de video JVC GR-AX767UM que proporciona el estándar VHS NTSC de 1Vpp, 75Ω de impedancia y salida analógica.*

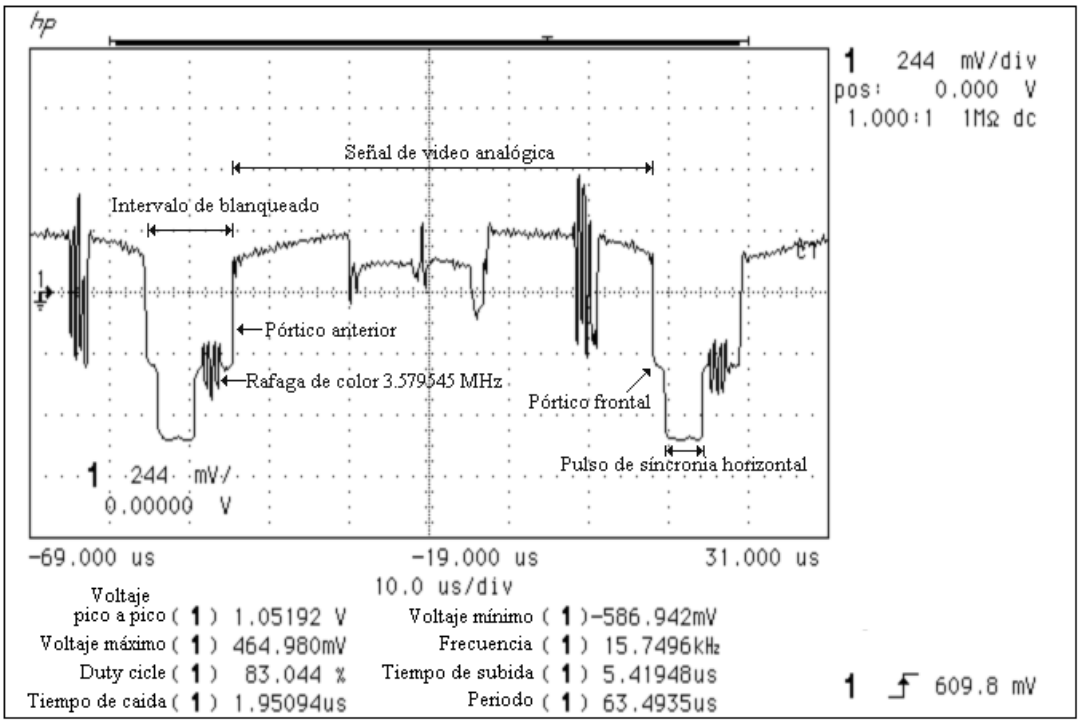

Figura 3.6. Señal de video compuesta proveniente de una cámara de video.

3.2.3.2.2 Corrección en CD de la señal de video compuesta

El Capacitor C1, los resistores R3 y R4 y el diodo D1 que se muestran en la figura 3.5 cambian el nivel de DC de la señal de video entrante, al necesario para que la parte analógica del sistema pueda realizar su tratamiento.

El acoplamiento en Corriente alterna (AC, *Altern current*) del video se realiza mediante C1, el cual es necesario para quitar cualquier offset en Corriente directa (DC, *Direct Current*) que se encuentre presente en la fuente de video.

Cuando la señal de AC está acoplada, el nivel síncrono tiene un valor aproximado a -0.286V, entonces el divisor de tensión formado por R3 y R4 produce una tensión de salida de aproximadamente 1.06V, esta tensión, acoplada con los 0.6V a 0.7V de caída a través del diodo D1 de pequeña señal, cambian la señal de video por aproximadamente 0.4V lo que lleva a las pulsos sincronos con tendencia negativa, por encima del nivel de tierra, localizándolos alrededor de 150mV, como se puede apreciar en la figura 3.6.

3.2.3.2.3 Eliminación de la ráfaga de color

Después de cambiar la señal de video al nivel apropiado se puede filtrar la ráfaga de color o la frecuencia de crominancia, resultando la señal mostrada en la en la figura 3.7. El filtrado se realiza mediante un filtro de muesca[47] integrado por los componentes C2, C3, L1 y R5, figura 3.7. Este filtro atenúa la frecuencia de la ráfaga de color del sistema NTSC localizada alrededor de 3.579545MHz para que su presencia no influya en la adquisición la imagen. Básicamente, el filtro convierte una señal de video a color a una monocromática.

[47] *Éste es un filtro pasabajas el cual permite que pasen a través de él las frecuencias bajas, está diseñado para atenuar la ráfaga de crominancia y frecuencias mayores para que éstas no afecten la señal de video, su respuesta se representa mediante diagramas de Bode.*

Este filtro es necesario solamente si se está utilizando una fuente de video a color con el digitalizador de video; si se utiliza una fuente de video monocromática, el filtro puede ser omitido del diseño.

El transistor Q1 y R6 mostrados también en la figura 3.6, implementan una configuración de seguidor de tensión y tiene por función aislar este circuito de la etapa siguiente.

3.2.3.2.4 Niveles de referencia

En esta etapa de la parte analógica se generan las tensiones de referencia que necesitan el circuito de corrección de offset del video y el convertidor analógico a digital.

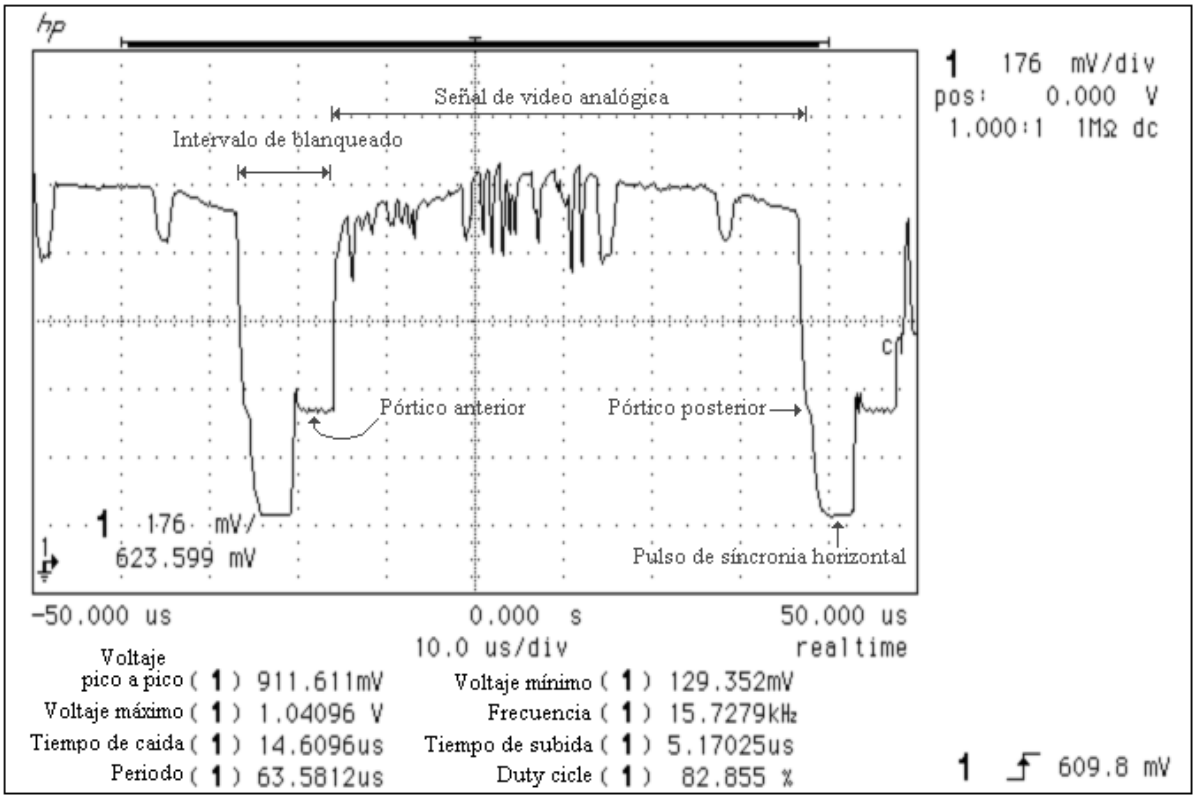

Figura 3.7. Señal de video corregido en CD.

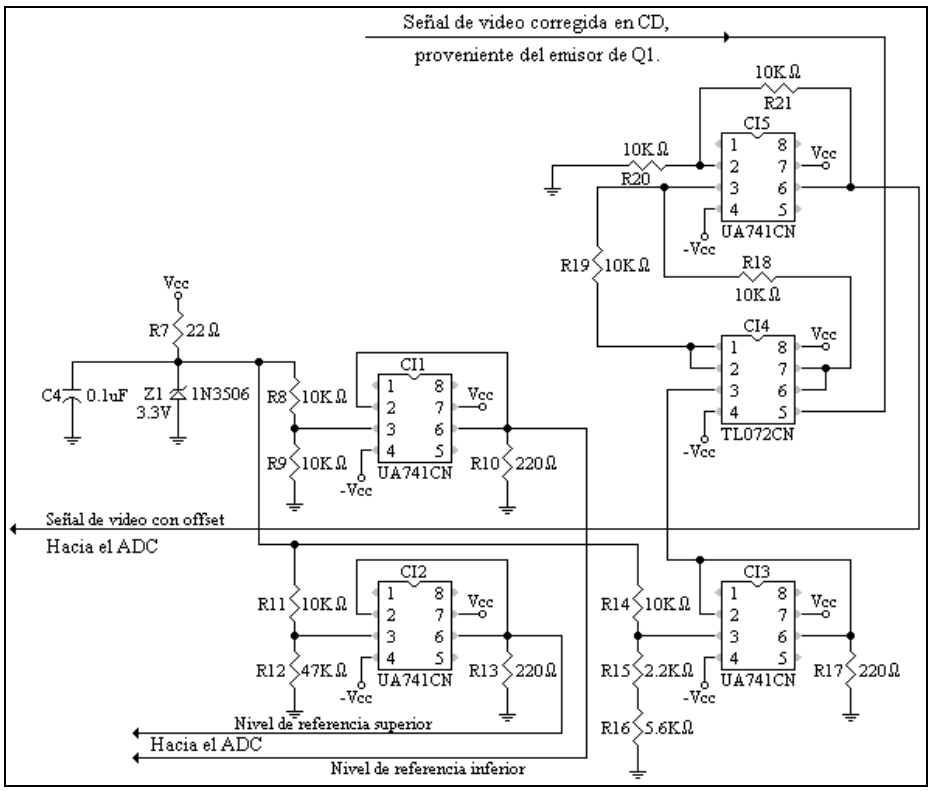

Figura 3.8. Generación de los niveles de referencia para el ADC y la señal de video con offset.

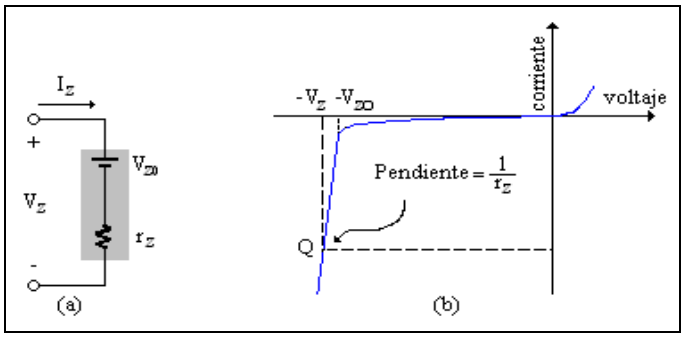

Figura 3.9. Diodo Zener. (a) Modelo con resistencia interna, (b) Curva característica.

La figura 3.8 ilustra el diagrama que genera los niveles de referencia. El diodo zener Z1, de 3.3V, en conjunto con C4 y R7 implementan una configuración de regulador de tensión que alimenta a los amplificadores encargados de generar los niveles de referencia necesarios.

El valor de R7 se obtiene a partir del modelo con resistencia interna del diodo Zener ilustrado en la figura 3.9. Partiendo de las ecuaciones obtenidas del análisis del modelo Zener:

$$V_Z = V_{Z0} + r_Z I_Z$$ **Ecuación (3.1).**

$$I_Z = I = \frac{V^+ - V_{Z0}}{R + r_Z}$$ **(Ecuación 3.2).**

donde: V_Z = Tensión nominal del diodo Zener.
V_{Z0} = Tensión de la fuente interna del diodo Zener.
V^+ = 5 V_{DC}.
r_Z = Resistencia incremental del diodo Zener en el punto de operación Q.
I_Z = Corriente de operación.

La ecuación 3.1 proporciona el valor de V_{Z0} ya que los parámetros r_Z e I_Z son proporcionados por el fabricante[48] y V_Z es la tensión nominal del diodo Zener por tanto V_{Z0} = 1.7V. En la ecuación 3.2 V^+ = Vcc = 5V, con los valores anteriores se puede obtener el valor de R que es el resistor R7 de aproximadamente 22Ω [44, 45].

La circuiteria que sigue es totalmente dependiente del diodo Zener ya que a partir de este regulador de tensión se generan las tensiones de referencia para el ADC y también el nivel de offset agregado a la señal de video.

El CI1, un UA741CN, implementa una configuración de seguidor de tensión, también conocido como seguidor de fuente, amplificador de ganancia unitaria o amplificador de aislamiento, en esta configuración la tensión de entrada se aplica a la entrada no inversora proveniente de un divisor de tensiones formado por R8 Y R9, proporcionando de esta forma la tensión necesitada para el nivel de referencia inferior, el cual se encuentra alrededor de 1.45V. El CI2, también un UA741CN, genera el nivel de referencia superior en conjunto con R11 y R12, implementando la misma configuración que el CI1, esta tensión está alrededor de 2.46V.

Aunque el CI3 implementa la misma configuración que CI1 y CI2 no genera una tensión de referencia, sino un nivel de offset agregado a la señal de video en conjunto con R14, R15 y R16, éste es de aproximadamente 0.8V, el cual se agrega a la señal de video para obtener la señal ilustrada en la figura 3.10.

El CI4, un TL072CN, contiene dos amplificadores operacionales, éstos implementan configuraciones de seguidor de tensión para el nivel de offset y para la señal de video proveniente del emisor de Q1. Las salidas del CI4 entran a un amplificador operacional en configuración de sumador no inversor[49] que implementa CI5 en conjunto con los resistores R18, R19, R20 y R21, encargado de proporcionar el offset a la señal de video.

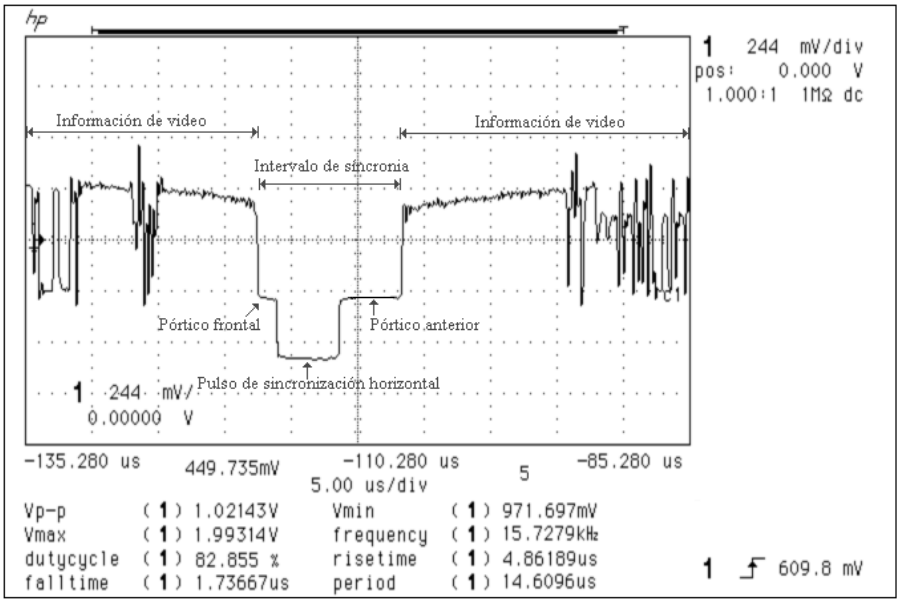

Figura 3.10. Señal de video con offset de 0.8V.

[48] *Fairchild Semiconductor es la empresa que desarrolla este tipo de diodos Zener, mas información se puede encontrar en sus hojas de especificaciones (BZX85C10.pdf) o en el sitio Web www.fairchild.com*

[49] *Esta configuración consta de dos partes, un promediador pasivo en este caso formado por R18 y R19, cuya salida alimenta a la segunda parte que es un seguidor de tensión, su finalidad es sumar las tensiones a la entrada del promediador pasivo.*

3.2.3.2.5 Separador de sincronía

El separador de sincronía de la señal de video, figura 3.11, tiene como elemento principal al LM1881, CI6. Este CI extrae la información de coordinación de la señal de video, incluyendo sincronización compuesta y vertical, coordinación de pórtico anterior e información de los campos par e impar desde una fuente de video NTSC, PAL ó SECAM con amplitudes desde 0.5Vpp hasta 2.0Vpp.

La salida vertical se produce en el rizo de subida del primer pulso en el periodo de sincronía vertical. El LM1881 fue diseñado para operar con una fuente de 5V a 12V, los únicos componentes que necesita aunados a la fuente de alimentación, son un capacitor de acoplamiento de corriente conectado al pín 6 (C5) en paralelo con un resistor (R14) que limita los niveles de corriente interna y un capacitor (C4) para el acoplamiento de la entrada compuesta.

El resistor R14 permite que el LM1881 se ajuste a fuentes de señal con frecuencias diferentes de 15.734 Khz (Frecuencia utilizada por el sistema NTSC).

El CI7, un LM311P, es un comparador de tensiones y es utilizado para convertir las salidas del LM1881 a nivel TTL, valores reconocidos por el FPGA. La referencia a la entrada del comparador está ajustada en aproximadamente 85mV, generada por el divisor de tensiones formado por R23, R24 y R25.

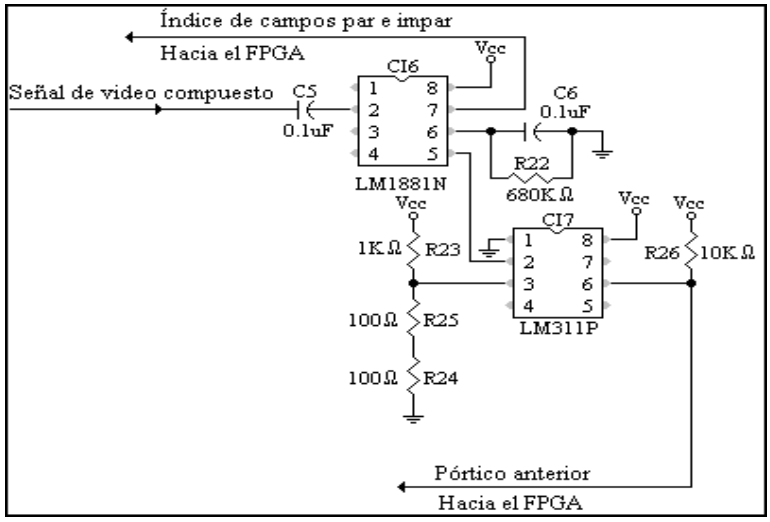

Figura 3.11. Esquemático del circuito separador de sincronía.

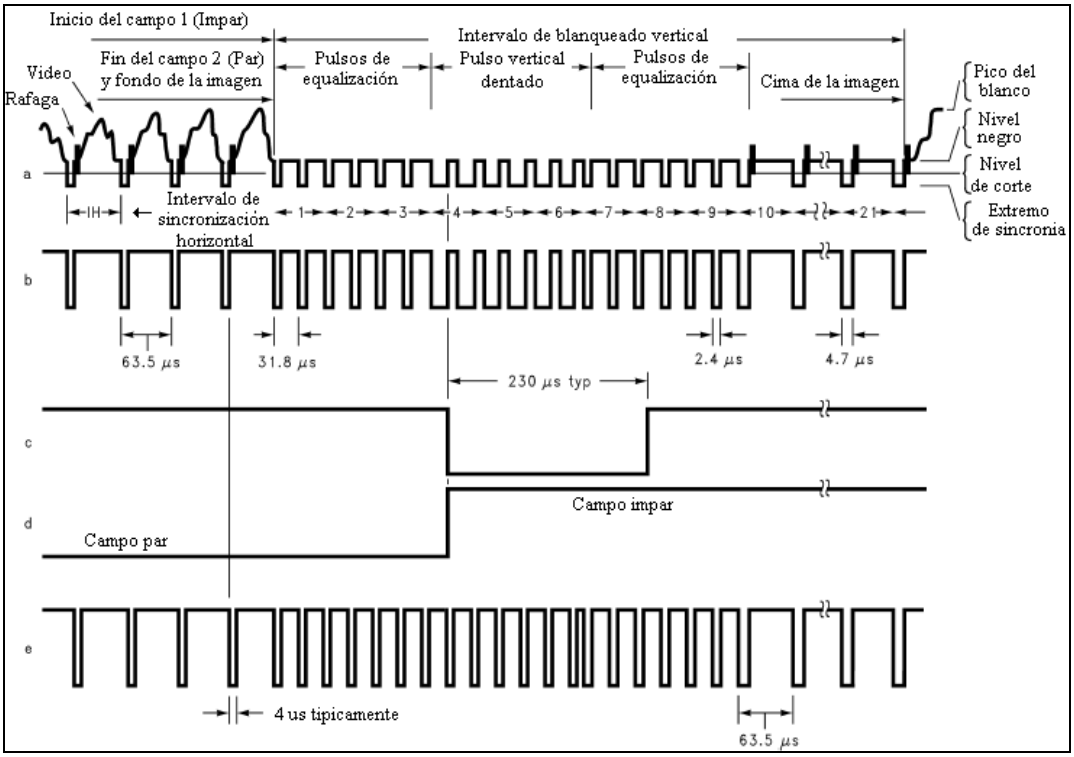

Figura 3.12. Señales controladas por el LM1881. (a) Video compuesta. (b) Sincronización compuesta. (c) Pulso de salida vertical. (d) Índice de campos par e impar. (e) Señal de pórtico anterior.

Aunque el LM1881 proporciona cuatro señales de sincronía especializadas, en este diseño solo se utilizaran la salida de campos par e impar y el pulso de pórtico anterior. Este par de salidas generan señales de coordinación que identifican los campos de video, recuperan señales de sincronización contaminadas u omitidas y también proporcionan referencias de coordinación para la extracción de datos codificados o no codificados en líneas de video específicas. Para entender mejor la información de coordinación y los tipos de señales del LM1881 éstas se ilustran en la figura 3.12 y se explican a continuación.

Sincronización compuesta

La salida de sincronización compuesta, figura 3.12b, es una reproducción de la señal de video compuesta bajo el nivel de negro. Esta señal se obtiene sujetando el extremo síncrono de la señal de video a 1.5V y usando un comparador de umbral fijo por arriba de esta tensión para quitar la sincronización de la señal, esta señal está disponible en el pin 1 del LM1881. El umbral de separación del extremo síncrono se encuentra nominalmente en los 70mV lo que significa, que para el nivel de entrada mínimo de 0.5Vpp el nivel de recorte estará cerca del punto medio sobre la amplitud del pulso síncrono.

Este umbral de separación es independiente de la amplitud de la señal, para una entrada de 2Vpp el nivel de corte se localiza en el 11% de la amplitud del pulso síncrono y la corriente de carga es de 0.8mA. Normalmente se asume que la fuente de señal se encuentra relativamente limpia y libre de ruido, sin embargo algunas fuentes como las señales originadas a partir de películas VHS tienen picos excesivos, provocando que la alta frecuencia del video y los componentes de crominancia se extiendan por abajo del nivel de referencia negro. Algunos discos de video guardan el pulso de ráfaga de crominancia presente durante el periodo de blanqueado vertical, para que la ráfaga aparezca sobre los extremos sincronos de tres periodos de línea en lugar de un nivel negro.

Cuando las impedancias de la fuente de señal son bajas, es decir, 75Ω a 620Ω, un resistor en serie con la fuente y un capacitor de 510pF a tierra formaran un filtro pasabajas con frecuencia de corte de 500Khz. Este ancho de banda es más que suficiente para dejar pasar la parte de los pulsos sincronos de la forma de onda, sin embargo, cualquier subportadora incluida en la señal se atenuara por lo menos en 18dB. Además de que el filtrado elimina el ruido térmico de la fuente. La salida tendrá un retardo de entre 40ns hasta 200ns debido a este filtro.

Identificación de los campos par e impar

Esta característica se puede utilizar para aplicaciones en las que es necesario almacenar marcos en memoria o en la extracción de señales de evaluación de campos alternados. Para una señal de video compuesta entrelazada uno de los dos campos de video que la integran debe contener la mitad de una línea horizontal al final de la exploración vertical, es decir, la última línea del campo impar contiene la mitad de la última línea de la imagen y la otra mitad se encuentra en la primera línea del campo dos. El campo impar inicia con el borde del primer pulso de ecualización mientras que el campo impar inicia en el borde del segundo pulso del intervalo de retrazo vertical.

Para detectar los campos par e impar el LM1881 integra la forma de onda de sincronía compuesta. Un capacitor se carga durante el periodo entre pulsos sincronos y se descarga cuando el pulso de sincronización está presente. El periodo entre pulsos de sincronización horizontal es suficiente para que la tensión del capacitor alcance el nivel de umbral de un comparador que limpia a un flip-flop cuya señal de reloj es la forma de onda síncrona.

Cuando se alcanza el intervalo vertical, el tiempo de integración más pequeño entre pulsos de ecualización previene que se alcance el umbral y también que la salida Q del flip-flop cambie con cada pulso. Ya que el periodo de la media línea al final del campo impar tiene el mismo efecto que un periodo del pulso de ecualización, la salida Q tendrá polaridad diferente sobre los subsecuentes campos. Así que comparando la polaridad de Q con el pulso de salida vertical, se genera un índice del campo par-impar. El pín 7, indicador de esta característica, permanece en bajo durante el campo par y en alto durante el campo impar como se puede observar en la figura 3.12(d).

Pulsos de salida del pórtico anterior y de ráfaga portal

En una señal de video compuesta, la ráfaga de crominancia se localiza en el pórtico anterior del periodo de blanqueado horizontal, su longitud es aproximadamente de 4.8us y también es el nivel negro para las subsecuentes líneas exploradas de video. El LM18881 genera un pulso en su pín 5 que puede ser utilizado para recuperar la ráfaga de crominancia de la señal de video compuesta o como una señal para la restauración en DC de la forma de onda de la señal. Esta señal que se muestra en la figura 3.12(e), es obtenida de forma simple luego de cargar un capacitor interno a partir del nivel de referencia de los pulsos de sincronización horizontal.

Simultáneamente la salida del pín 5 cambia al nivel bajo y se mantiene hasta que se carga el capacitor, 4 μs después.

Un pulso de salida de ráfaga portal se puede obtener luego de diferenciar la ráfaga de salida utilizando una red C-R[50], necesaria en aplicaciones que requieren rangos de exploración horizontal altos en combinación con rangos de exploración vertical (60Hz-120Hz).

[50] *Una red C-R es un circuito que obtiene su respuesta natural con la ausencia de fuentes independientes mientras se libera la energía de C, de otra forma el circuito se estará cargando de energía.*

3.2.3.2.6. Convertidor Analógico-Digital

El convertidor analógico digital (ADC, *Analog Digital Converter*) es el elemento que tiene por función la digitalización de la señal de video, por tanto se encarga de enlazar el tratamiento analógico con el tratamiento digital de la señal de video.

El ADC elegido es el ADS802U, un ADC monolítico de 12 bits cuyo circuito de interfaz se ilustra en la figura 3.13, éste opera a baja potencia y con una frecuencia de muestreo máxima de 10 millones de muestras por segundo.

Algunas de sus ventajas son que opera con una sola fuente de alimentación de 5V y se puede configurar para aceptar señales diferenciales o referenciadas a tierra. Emplea corrección de errores para proveer una excelente ejecución lineal, baja distorsión, alta relación señal a ruido y sobremuestreo, todo esto hace posible que se utilice en aplicaciones de telecomunicaciones, instrumentación y video.

En la tabla 3.2 se describen brevemente los pines que integran el ADS802U. En función de las especificaciones del sistema, apartado 3.2.1, de los 12 bits de resolución que tiene el ADC únicamente se usaran los 8 bits más significativos.

Tabla 3.2 Breve descripción de los pines del ADS802.

Pín	*Nombre*	*Descripción*
1,14,25,26	GND	Tierra
2…13	B1…B12	Bus de datos: LSB…MSB
15,17,20,24	+Vs	Fuente de alimentación de +5V
16	CLK	Entrada de reloj para la conversión, 50% Duty cycle
18	OE	Habilitación de salida.
19	MSBI	Inversión del bít más significativo. Utilizada para obtener una salida binaria en complemento a dos
21	REFB	Nivel de referencia inferior. Mínimo +1.1V
22	CM	Tensión en modo común. Es derivado de (REFT + REFB) / 2
23	REFT	Nivel de referencia superior. Máximo +3.4V
26	IN	Entrada Analógica
27	$\overline{IN}$	Entrada complementaria.

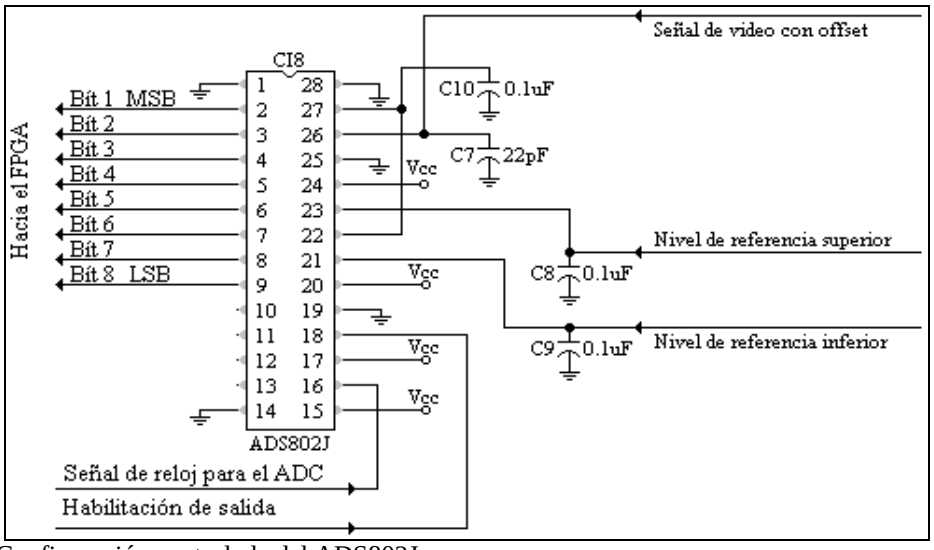

Figura 3.13. Configuración controlada del ADS802J.

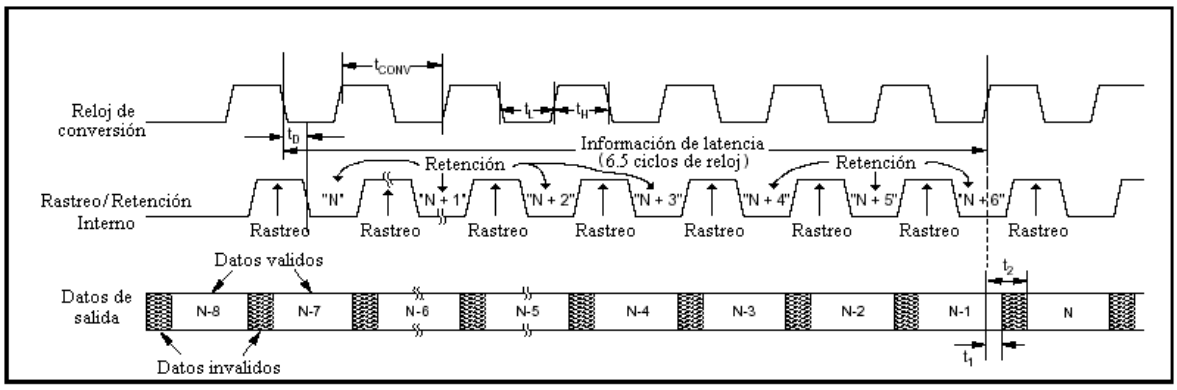

Figura 3.14. Diagrama de inicialización para el ADS802.

La señal de reloj (CLK) del ADC es generada por el FPGA, ésta debe tener un valor menor o igual a 10MHz. En la figura 3.14 se muestra el diagrama de inicialización para obtener resultados correctos, es decir al conectar la tensión de alimentación se necesitan 7.5 ciclos del reloj de conversión para que el ADC se configure después de esto los datos digitalizados corresponderán a la señal a la entrada, en la tabla 3.3 se describen los tiempos correspondientes a las señales antes mencionadas y mostradas en la figura 3.14.

El nivel de tensión máximo que el ADC puede convertir está limitado por el nivel de referencia superior (REFT), que en la señal de video con offset representa el nivel blanco, y el nivel de tensión mínimo que el ADC puede convertir estará limitado por el nivel de referencia inferior (REFB), éste representa el nivel negro en la señal de video compuesta con el offset agregado.

El nivel síncrono de la señal con offset se localiza alrededor de los 950mV, el cual no es el mismo que el nivel negro que se localiza alrededor de 1.45V, por lo tanto, la señal de video se digitalizara en un rango de tensiones que va desde 1.45V hasta 2.45V, se mantiene entonces un rango de 1V para digitalizar la información de video, esto se debe a que frecuentemente las señales contienen picos que exceden la señal de video compuesta, es decir, pueden ser mayores a 1Vpp y estos picos son información de video importante en muchos casos. Estos niveles de referencia proporcionan un incremento binario por cada 3.9mV de la señal de video con offset.

Tabla 3.3. Descripción de los símbolos utilizados en la coordinación del ADS802.

Símbolo	*Descripción*	*Mínimo*	*Típicamente*	*Máximo*	*Unidades*
t_{CONV}	Periodo de conversión del reloj	100		100000	ns
t_L	Pulso bajo del reloj	48	50		ns
t_H	Pulso alto del reloj	48	50		ns
t_D	Retardo de apertura		2		ns
t_1	Tiempo de retención de los datos	3.9			ns
t_2	Tiempo de retardo del nuevo dato,			12.5	ns

3.2.3.3. Módulo digital del digitalizador de video

La parte digital del digitalizador de video, ilustrada en la figura 3.4, digitaliza y manipula la información de la señal de video compuesta, previamente procesada por la parte analógica del sistema. El FPGA mostrado en la figura 3.15 es la parte medular de este diseño, ya que controla todo el módulo digital y modela los algoritmos de digitalización del video.

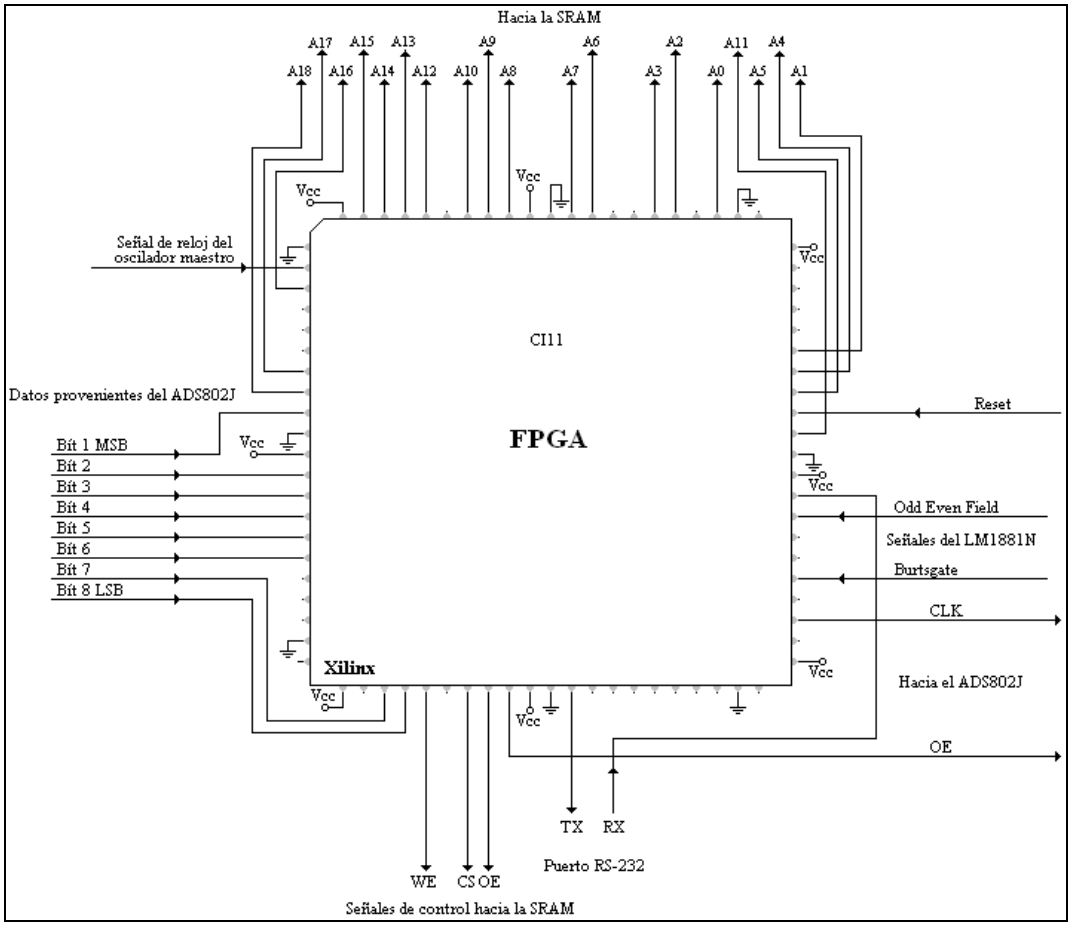

Figura 3.15. Conexiones del FPGA.

Las funciones realizadas por la circuiteria digital son:

- Hardware del procesador de video.
- Sistema de memoria.
- Convertidor de niveles RS232-TTL.
- Control del ADS802U. Apartado 3.2.3.2.6.

3.2.3.3.1. Hardware del procesador de video

La implementación del procesador de video será hecha en un FPGA XC4000X y un XC2S200 de la compañía Xilinx, este dispositivo combina versatilidad arquitectónica, memoria RAM sobre el chip seleccionada mediante bordes de activación y modos de doble puerto, incremento en la velocidad, abundantes recursos de ruteo y un software sofisticado para llevar a cabo la implementación automatizada completamente de diseños complejos de alto rendimiento y densidad. La figura 3.15 muestra las señales que interviene en la conexión del FPGA con los elementos digitales y analógicos que conforman el diseño del digitalizador de video, la tabla 3.4 recopila el nombre y una breve descripción de estas señales. Las características más importantes del FPGA XC4000 se describen brevemente en el capitulo 2, sin embargo se puede consultar el archivo 4000.pdf del sitio www.xilinx.com para obtener información completa de este dispositivo, en cuanto al FPGA XC2S200 se pueden consultar sus características en el archivo ds001_2.pdf del mismo sitio WEB.

Tabla 3.4. Señales manipuladas por el XC4010XL.

Nombre	*Tipo de señal*	*Pines*	*Descripción*
Clock	Entrada	13	Proviene del oscilador, su función principal es poner en operación al FPGA con cada flanco de subida.
Reset	Entrada	66	Le indica al FPGA que sus variables deben tomar valores iniciales.
OddEvenF	Entrada	61	Esta señal informa que campo de video está transcurriendo.
BurstGate	Entrada	58	Con un flanco de subida marca el inicio de captura de la información de video contenida en cada línea.
RX	Entrada	62	Recibe los comandos enviados desde la PC.
TX	Salida	44	Transmite los datos de las imágenes hacia la PC.
A	Salida	77,65,79,80,67, 68,83,84,3-5,69, 7-10,14,18,19	Direcciones de la SRAM.
O	Entrada/salida	20,23-27,35,36	Datos de la SRAM.
CS	Salida	39	Selecciona la SRAM utilizada, en este diseño solo se utiliza una, asi que puede permanecer en bajo asegurando el funcionamiento.
OE	Salida	40	Controla el envió de datos de la SRAM al FPGA ó el estado de alta impedancia de los pines de datos de la SRAM.
WE	Salida	37	Permite almacenar los datos de la imagen en la SRAM
G	Salida	41	Controla el envió de datos o el estado de alta impedancia del ADC.
Convert	Salida	56	Genera la señal de reloj para el ADC.

3.2.3.3.2 Sistema de memoria

El sistema de memoria está formado por una SRAM KM684000 de la compañía Samsung, las características relevantes de esta memoria son: organizada como 512K x 8 localidades, fabricada mediante un proceso de tecnología CMOS, trabaja en un rango de temperatura de 0°C a 70°C, con una fuente de alimentación de 4.5V a 5.5V y con un tiempo de acceso de 70ns, también puede operar con baterías de respaldo consumiendo muy poca corriente. La interfaz del sistema de memoria con el FPGA puede ser apreciada en la figura 3.16.

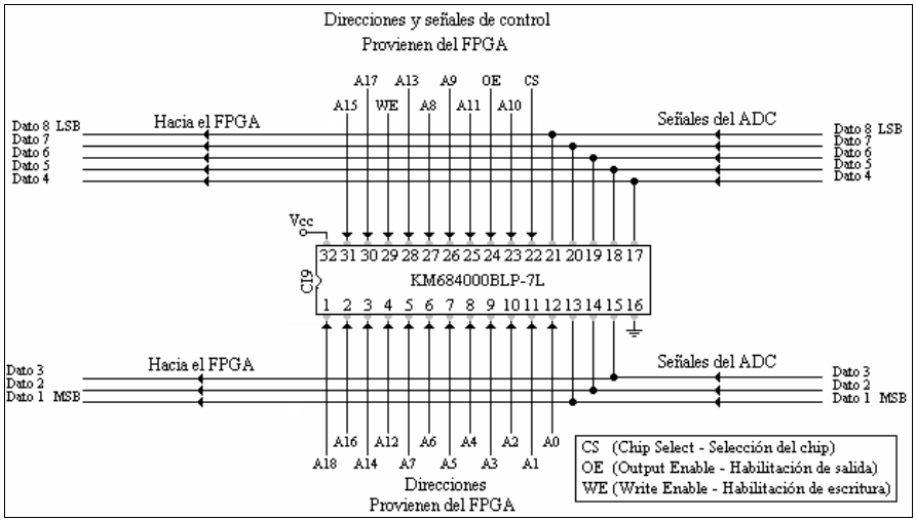

Figura 3.16. Diagrama de conexiones de la SRAM con el FPGA.

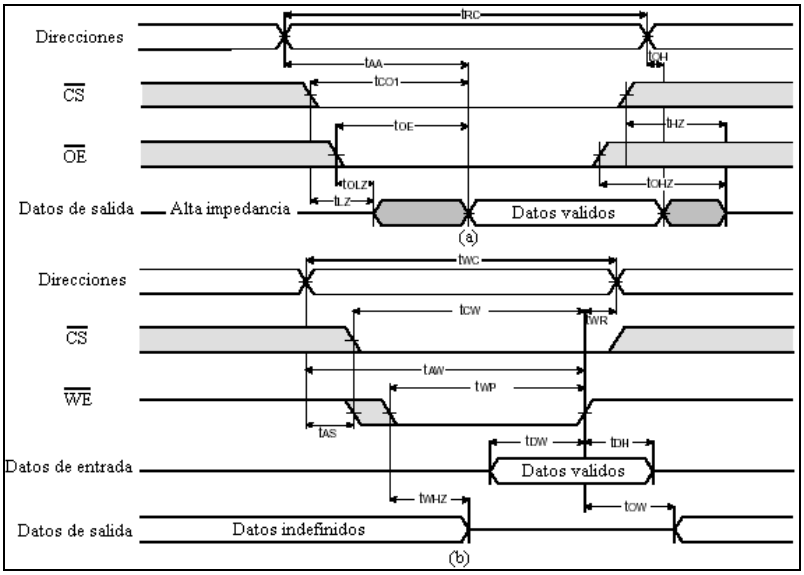

Figura 3.17. Diagramas de tiempos de la SRAM. (a) Lectura, (b) Escritura.

Tabla 3.5. Tiempos para ejecutar funciones de lectura y escritura de la SRAM.

	Lista de parámetros	*Símbolo*	*Mínimo*	*Máximo*	*Unidades*
Lectura	Tiempo del ciclo de lectura	t_{RC}	70	-	ns
	Tiempo de acceso a la dirección	t_{AA}	-	70	ns
	Tiempo del CS para obtener datos	t_{CO}	-	70	ns
	Tiempo del OE para obtener datos	t_{CE}	-	35	ns
	Salida del CS a baja impedancia	t_{LZ}	10	-	ns
	Salida del OE a baja impedancia	t_{OLZ}	5	-	ns
	Salida del CS a alta impedancia	t_{HZ}	0	25	ns
	Salida del OE a alta impedancia	t_{OHZ}	0	25	ns
	Salida desde el cambio de dirección	t_{CH}	10	-	ns
Escritura	Tiempo del ciclo de escritura	t_{WC}	70	-	ns
	CS al final de la escritura	t_{CW}	60	-	ns
	Tiempo de estructuración de las direcciones	t_{AS}	0	-	ns
	Direcciones validas al final de la escritura	t_{AW}	60	-	ns
	Ancho del pulso de escritura	t_{WP}	50	-	ns
	Tiempo de restauración de escritura	t_{WR}	0	-	ns
	Escritura a alta impedancia	t_{WHZ}	0	25	ns
	Datos sobrepuestos al tiempo de escritura	t_{DW}	30	-	ns
	Tiempo de escritura desde retención de datos	t_{DH}	0	-	ns
	Final de escritura a baja impedancia	t_{OW}	5	-	ns

Los ciclos de acceso, escritura y lectura, de la memoria KM684000 involucra al bus de direcciones, al bus de datos y a las señales de control OE, WR y CS, tal como se puede apreciar en la figura 3.17; los tiempos que deben de cumplir estas señales están disponibles en la tabla 3.5.

3.2.3.3.3 Convertidor de niveles RS232-TTL

Los niveles de tensión manejados por el puerto serie de una PC se encuentran en el rango de ±7 a ±13 Volts, los cuales resultan incompatible con la lógica TTL que maneja un sistemas digitales, por tanto, esta parte del sistema tiene por función la conversión de niveles RS232 a TTL y viceversa.

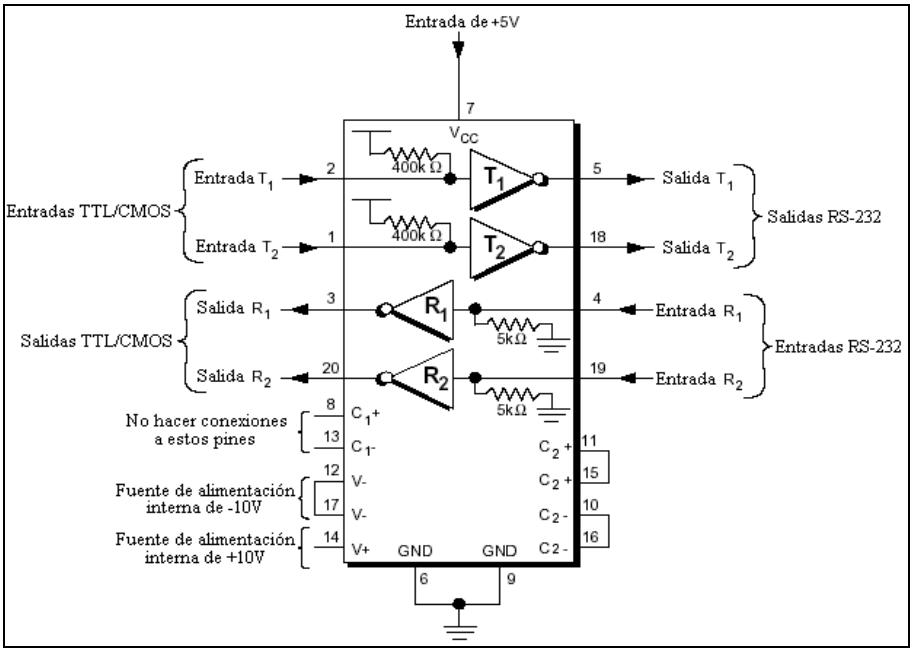

Figura 3.18 Configuración interna y externa del SP232ACP.

Esta conversión entre niveles se implementa mediante el CI SP233ACP, figura 3.17. Algunas de sus características son:

- 5V de alimentación.
- Cumple con las especificaciones establecidas en el estándar RS-232.
- Puede implementar 2 transmisores / receptores.
- Velocidad de transmisión ó recepción hasta de 120Kbps.
- Requiere solamente de un capacitor de tantalio conectado a los pines 7 y 9.
- Operación CMOS de baja potencia.

El SP232ACP está integrado por tres bloques básicos, transmisores, receptores y un surtidor de carga. Los transmisores, denotados con T1 y T2 en la figura 3.18, aceptan señales de entrada con niveles TTL o CMOS y emite señales RS-232 invertidas con relación a los niveles lógicos de entrada. Los receptores, R1 y R2 en la figura 3.18, convierten señales de entrada RS-232 a señales TTL invertidas.

La entrada proviene de una línea de transmisión, donde la longitud de los cables largos y la interferencia pueden degradar la señal, entonces las entradas tendrán un margen de histéresis típico de 500mV. Esto asegura que el receptor sea virtualmente inmune a líneas de transmisión ruidosas. Los umbrales son de 0.8V mínimo y 2.4V máximo, cumpliendo así con los requerimientos establecidos por el estándar RS-232 de ± 3V.

La sección del surtidor de carga de este dispositivo permite que el circuito opere con una sola fuente de alimentación de +5V, ésta genera las tensiones de operación internos requeridos por el dispositivo. El surtidor de carga está formado por dos secciones:

- Un doblador de tensión.
- Un inversor de tensión.

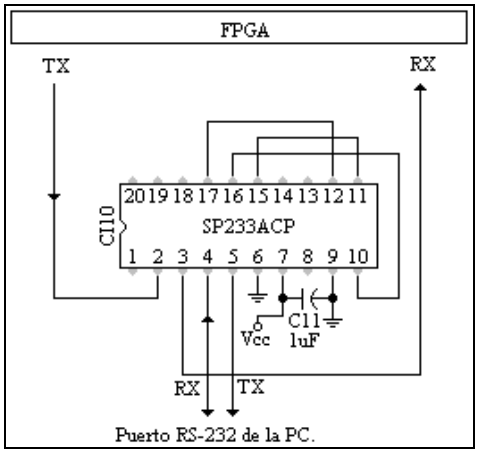

Figura 3.19. Conexiones del CI SP233ACP.

C_1 y C_2 son capacitores internos que en conjunto con un oscilador también interno controlan la acumulación de carga y la inversión de la tensión, en las hojas de datos del SP232ACP se pueden analizar las configuraciones internas. Por otra parte se puede apreciar que la ventaja de utilizar el CI10 es que no requiere componentes externos.

La figura 3.19 muestra la interfaz entre el FPGA y el puerto serie de una PC, a través del convertidor de niveles SP233ACP.

3.2.4 Procesador de Video

Para el modelado del procesador de video se hace uso de herramientas para la Automatización del Diseño Electrónico (EDA, *Electronic Design Automation*), éstas están integradas por el conjunto de herramientas, tanto hardware como software, que son empleadas en el diseño de sistemas electrónicos [48]. Las Herramientas EDA software, llamadas Herramientas de Diseño Asistido por Computadora (CAD, *Computer Aided Design*), se utilizan en el diseño de hardware mediante software.

Las Herramientas EDA hardware están formadas principalmente por tarjetas de evaluación de dispositivos electrónicos. La finalidad de las herramientas EDA es volver al diseño electrónico un proceso más simple, aumentar la calidad del producto final y reducir los costos de producción.

El modelado o descripción del procesador de video es hecho con el lenguaje descriptor de hardware VHDL utilizando la herramienta EDA-CAD Xilinx Foundation versión 4.1i de la compañía Xilinx. Para la implementación del diseño se hizo uso de las herramientas EDA hardware XS-40 de Xess Company.

Antes de iniciar con el modelado de un sistema digital mediante un HDL, es necesario definir el tipo de modelado a utilizar y la metodología de diseño a seguir. Consideraciones relevantes en la definición de estos puntos son: el nivel de abstracción que se desee manejar y la complejidad del sistema a modelar.

Dentro de un HDL existen en general dos tipos de modelado o niveles de abstracción:

- **Estructural**. Describe el sistema como un grupo de compuertas o bloques que se interconectan. La descripción estructural se asemeja a un esquemático pines de entrada-salida.

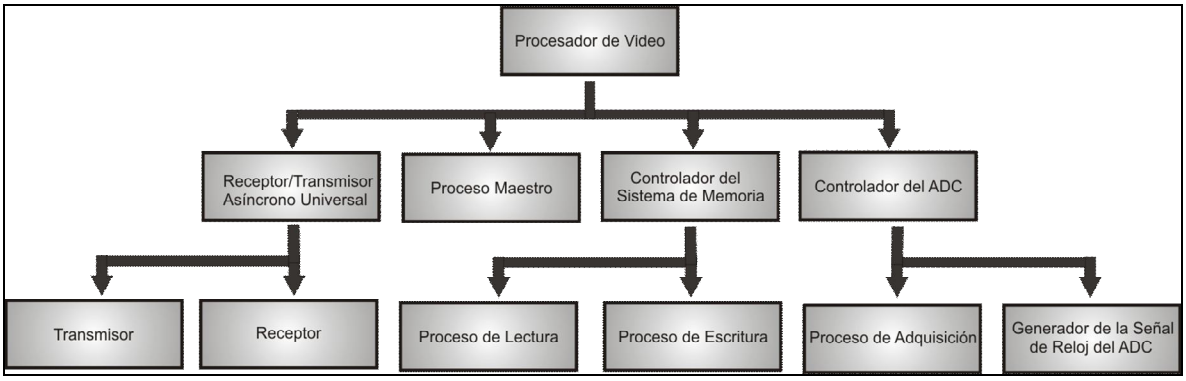

Figura 3.20. División del sistema siguiendo la metodología descendente.

- **Comportamental**. Describe el comportamiento de las salidas en función de las entradas (Expresión booleana, máquinas de estado, etc.).

Por otra parte, existen también dos metodologías de diseño principales, ascendente (*bottom-up)* y descendente (*top-down)*, estas metodologías son estrategias de procesamiento de información y ampliamente usadas en las herramientas EDA-CAD.

- En el diseño ***descendente*** se captura una idea en un nivel de abstracción alto y se implementa a partir de ésta descripción, en un proceso hacia abajo incrementando el nivel de detalle según lo requerido. Cada parte nueva es entonces redefinida, cada vez con mayor detalle hasta que la especificación completa es lo suficientemente detallada para validar el modelo. El diseño descendente se diseña con frecuencia con la ayuda de "cajas negras" que hacen más fácil llenar requerimientos aunque estas cajas negras no expliquen en detalle los componentes individuales.
- En contraste, en el diseño ***ascendente*** las partes individuales se diseñan con detalle y luego se enlazan para formar componentes más grandes, que a su vez se enlazan hasta que se forma el sistema completo. Las estrategias basadas en el flujo de información ascendente parecen potencialmente necesarias y suficientes porque se basan en el conocimiento de todas las variables que pueden afectar los elementos del sistema.

Para el modelado del procesador de video se ha elegido utilizar una descripción comportamental y seguir la metodología de diseño descendente. Siguiendo los fundamentos de la metodología descendente, el procesador de video es dividido en los módulos ilustrados en la figura 3.20.

3.2.4.1 Unidad central de proceso o proceso maestro

En el diseño se declaran 33 objetos de tipo *signal* para respaldo de la información proveniente de las señales de entrada y también para control, además de implementar rutinas principalmente de conteo o comparación.

Todos los procesos realizan cada instrucción cuando ocurren los flancos de subida del oscilador maestro que controla el modulo digital para que se ejecute sincronizadamente, el diagrama de flujo de la figura 3.21 muestra el proceso maestro.

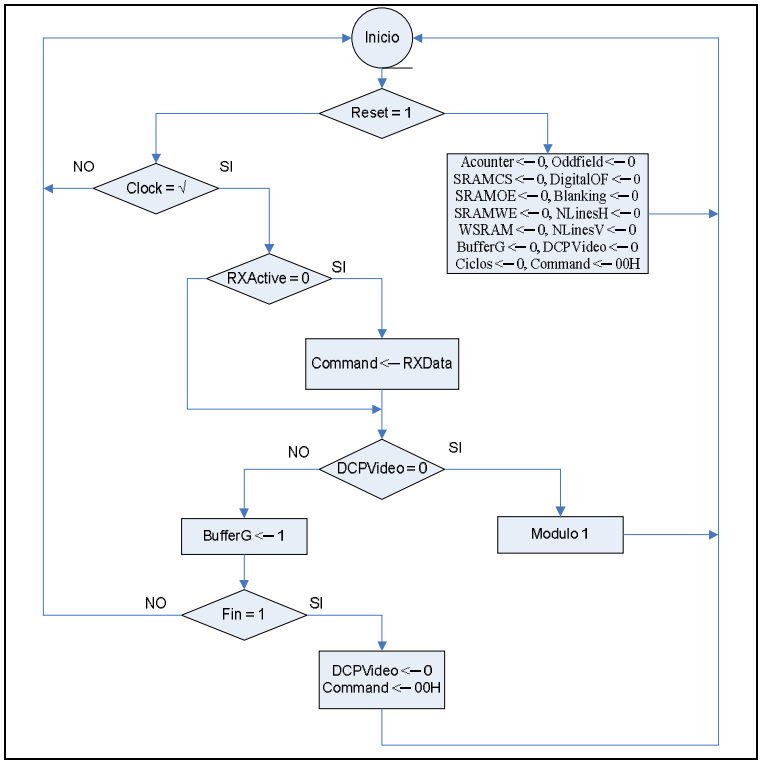

Figura 3.21 Proceso maestro.

La figura 3.22 que corresponde al modulo 1 es parte del proceso principal el proceso principal donde se lleva a cabo la sincronía del proceso de digitalización. La descripción de estos diagramas de flujo es la siguiente:

Cuando **Reset** recibe el valor de 1, las variables del proceso reciben valores iniciales, como se puede ver en la figura 3.23. **ACounter** controla el número de localidad de la SRAM accesada por el FPGA, **SRAMCS** selecciona la SRAM utilizada, **SRAMOE** habilita la salida de datos de la SRAM hacia el FPGA, **WSRAM** controla el proceso de escritura a la SRAM, **BufferG** permite que los datos digitalizados por el ADC sean enviados hacia el FPGA, **Ciclos** corresponde a los ciclos de reloj que se tarda el FPGA en acceder una localidad de la SRAM ya sea para lectura o escritura, **Oddfield** se refiere al campo de video actual, **DigitalOF** indica cuando existe información de la imagen y no de sincronía, **Blanking** controla el numero de pulsos de blanqueado vertical, **NLinesH** cuenta el número de líneas de video capturadas, **NLinesV** cuenta el número de píxeles por cada línea de video, **DCPVideo** controla el proceso de digitalización y transmisión de video, **Command** indica al FPGA cuando debe terminar de capturar la imagen en turno e iniciar la transmisión de la imagen almacenada.

RXActive coordina el proceso principal y el de recepción, ya que cuando recibe el valor de 1, **Command** recibe el valor del carácter almacenado en **RXData** capturado vía RS-232 en el proceso de recepción, esta asignación se ilustra en la figura 3.23.

Si **DCPVideo** recibe el valor de 0 se ejecuta el diagrama de flujo correspondiente al modulo 1, éste se ilustra en la figura 3.22. La primer operación de este modulo le indica a **SRAMCS**, **SRAMOE** y **BufferG** que reciban valores iniciales, así se habilitan la SRAM y el ADC para su utilización posterior.

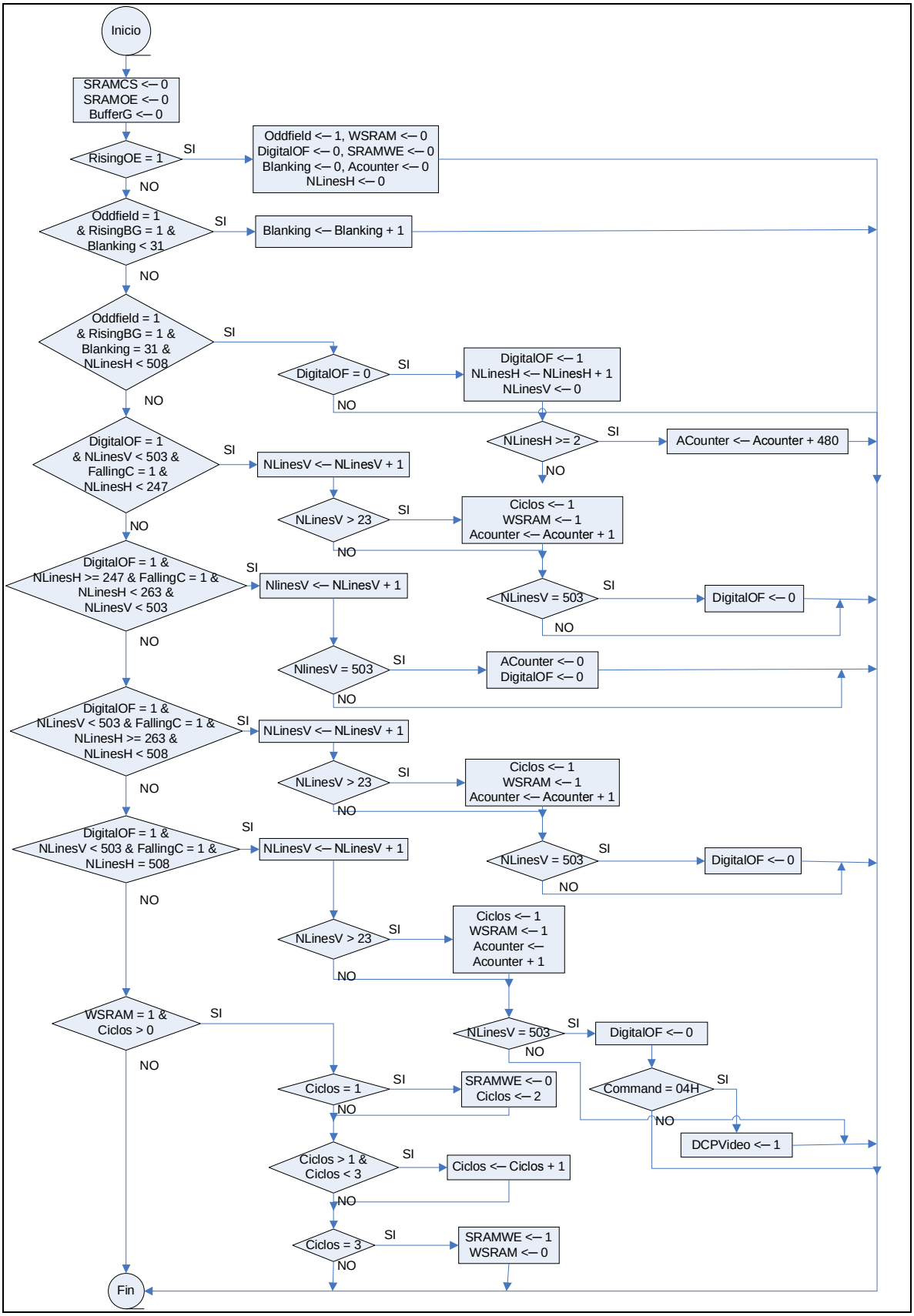

Figura 3.22. Módulo de digitalización.

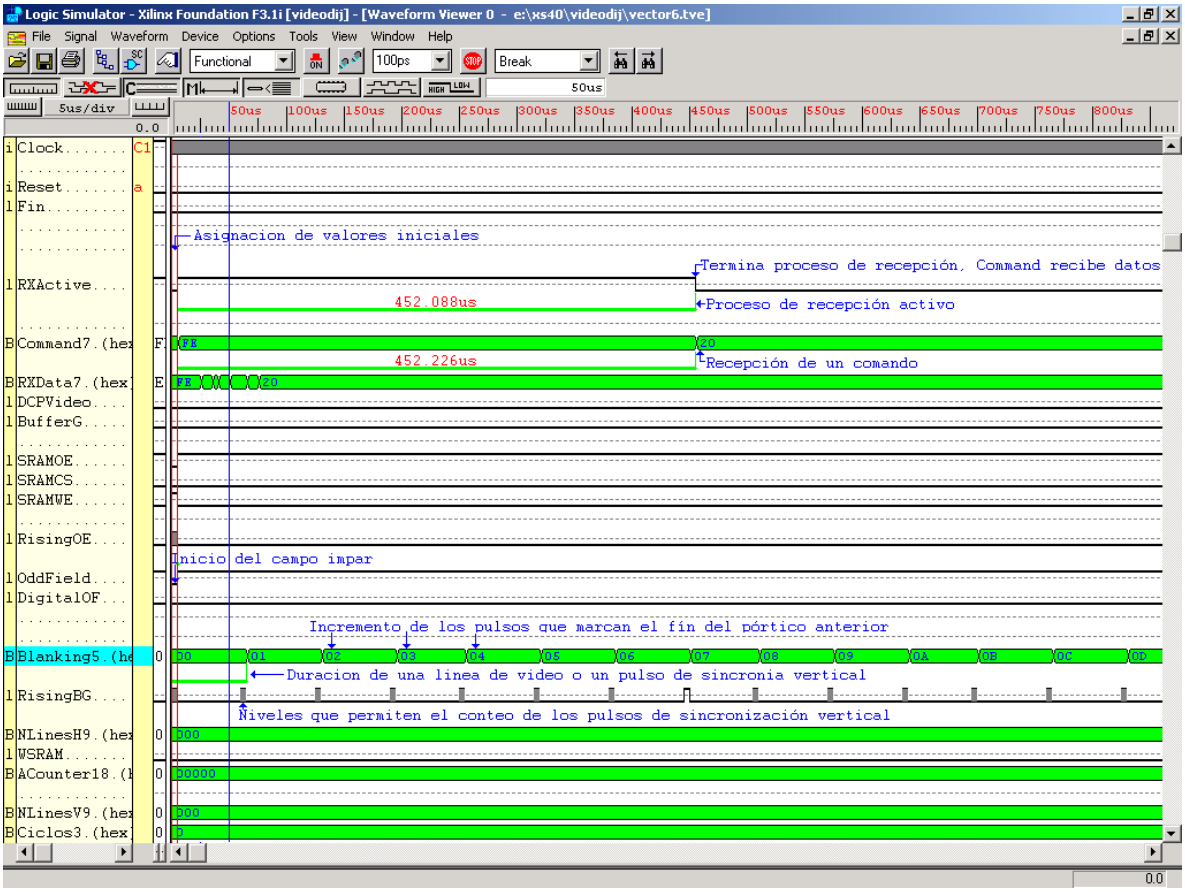

Figura 3.23. Detección de los pulsos de sincronía.

La siguiente condición se cumple cuando **RisingOE**, recibe el valor de 1, esto sucede cuando se genera un flanco de subida en la señal índice de campos par e impar de la figura 3.12(d), el valor de 1 solo se mantiene en un ciclo de reloj, como se ilustra en la figura 3.23, si en ese tiempo **DCPVideo** está inhabilitando el proceso de captura de una imagen completa se esperara hasta que ocurra un nuevo flanco de subida y que **DCPVideo** esté habilitando el proceso, para iniciar una nueva captura de imágenes.

Cuando **RisingOE** tiene el valor de 1 **Oddfield** recibe el valor de 1, de esta forma se sabrá que el campo impar o uno está activo mientras mantenga este valor. En este tiempo **DigitalOF** recibe valor de 0, el cual lo pone en espera hasta el momento de empezar a capturar los píxeles que componen una imagen. **Blanking** recibe el valor de 0 en espera del intervalo de pulsos de sincronía vertical. **NLinesH**, **WSRAM**, **SRAMWE** y **ACounter** también reciben valores iniciales hasta que inicie la información de video.

La siguiente condición se cumple cuando el campo impar está activo, es decir, que **Oddfield** tenga el valor de 1, también es necesario que ocurra un flanco de subida en la señal de pórtico anterior ilustrada en la figura 3.12(e), el cual es detectado por **RisingBG** y que **Blanking** sea menor que 31, éste ultimo es el número de pulsos[51] de blanqueado vertical que deberán transcurrir para llegar a la información de la imagen. Mientras que las condiciones anteriores se cumplan el proceso solo esperara hasta cumplir con los 31 pulsos para iniciar la digitalización de la imagen en turno, sumando 1 a **Blanking** en cada flanco de subida de la señal de pórtico anterior detectado por **RisingBG** como se ilustra en la figura 3.23.

[51] *La cantidad de pulsos varía de acuerdo al estándar utilizado, éste es un parámetro a modificar si se desea utilizar una fuente de video que controle un estándar de video diferente tal como PAL, SECAM o alguna variación.*

Cuando **Oddfield** tiene el valor de 1 y **Blanking** ha conseguido el valor de 31, además de que el número de líneas horizontales contadas mediante **NLinesH** sea menor de 508, es necesario esperar a que ocurra un nuevo flanco de subida en la señal de pórtico anterior para que **RisingBG** lo detecte y reciba el valor de 1. Entonces **DigitalOF** tomara el valor de 1 permitiendo así iniciar la digitalización de video, sin embargo antes de asignar el valor es necesario que **DigitalOF** sea igual con 0.

Cuando las señales anteriores reciben los valores mencionados, **NLinesH** incrementa su valor en una unidad y **NLinesV** recibe 0, para que la siguiente línea de video compuesta digitalizada se almacene en SRAM, como se ilustra en la figura 3.24. En el capitulo 1 se estableció que en la norma RS-170 la información de un marco de video está almacenada en 2 campos, que entrelazados forman un marco.

Por lo tanto después de la primera línea de video almacenada, las subsecuentes dejaran el espacio para almacenar las líneas contenidas en el campo de video par como se muestra en la figura 1.4. Esta condición se cumple cuando **NLinesH** es mayor que 1, para dejar el espacio en SRAM se suma 480 a **ACounter** que representa el número de dirección de la SRAM en cuestión, como se ilustra en la figura 3.24.

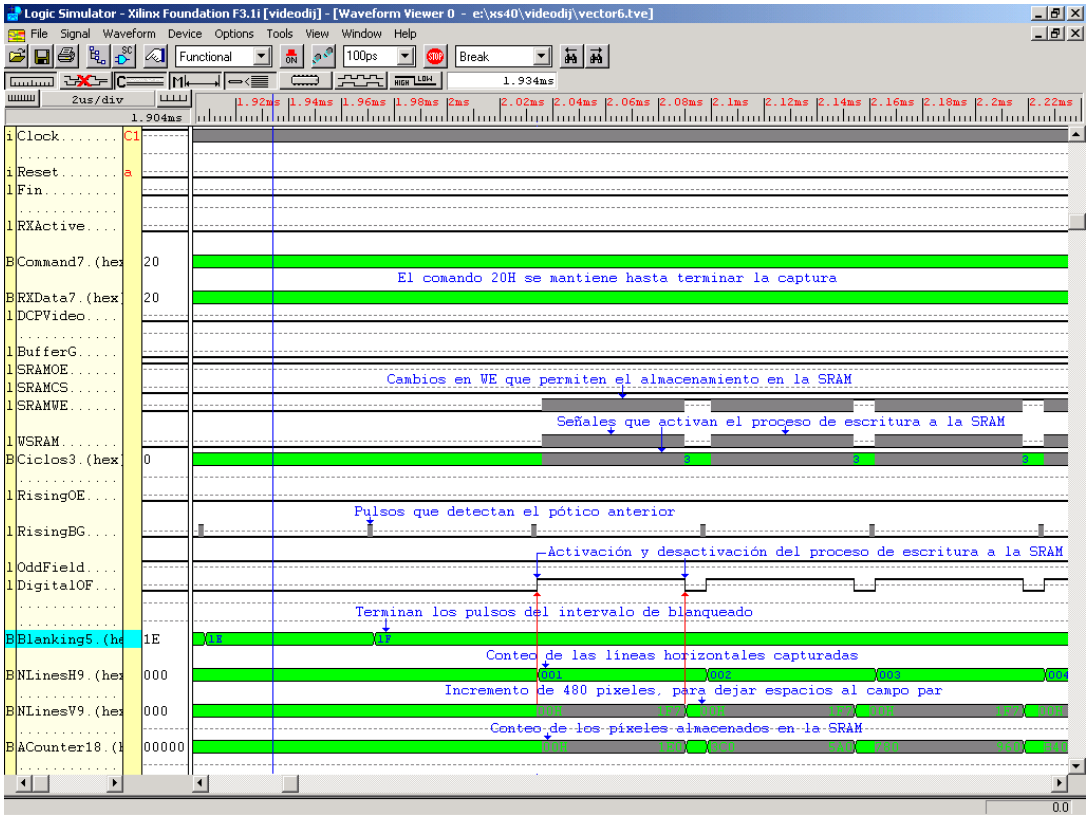

Figura 3.24. Diagrama de tiempos del almacenamiento para el inicio de un marco impar.

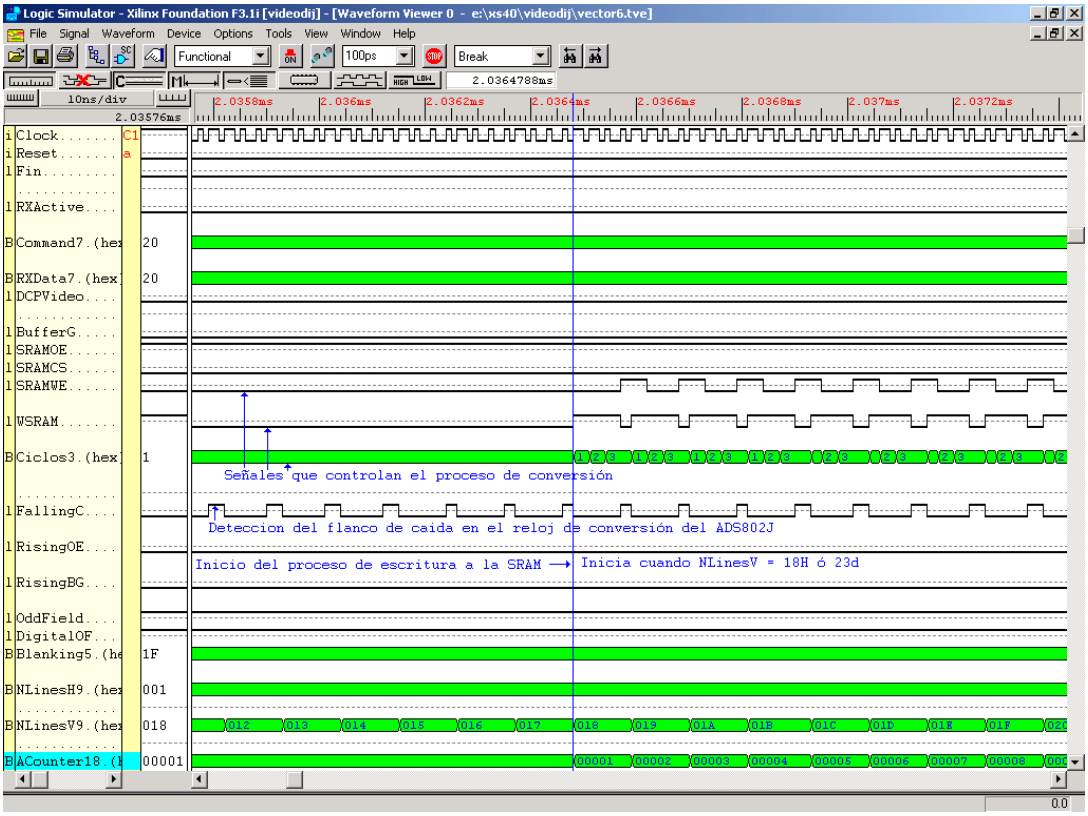

Figura 3.25. Diagrama de tiempos del almacenamiento para un píxel.

El primer campo de video se captura cuando **DigitalOF**, **NLinesV**, **NLinesH** y **FallingC** actuan de la siguiente manera; **DigitalOF** debe tener el valor de 1, **NLinesV** debe ser menor que 503, es decir, que la cantidad de píxeles por línea digitalizados sea menor que 503 y que **NLinesH** sea menor que 247 (este número asegura que se está almacenando el campo de video impar). Si las señales anteriores reciben los valores mencionados, es necesario que ocurra un flanco de caída del reloj de conversión del ADC para que **FallingC** reciba el valor de 1 y así la condición se ejecute.

La digitalización inicia con incrementos de 1 unidad en **NLinesV**, sin embargo el almacenamiento en SRAM se inicia con la digitalización del píxel número 24, al llegar a este valor se activa el proceso de escritura a la SRAM, como se muestra en la figura 3.25. Este proceso se realiza en cuatro ciclos de reloj los que son contabilizados por **Ciclos**, para que éste se lleve a cabo el objeto **WSRAM** debe recibir el valor de 1. **ACounter** se incrementa cada vez que un píxel se almacena en SRAM hasta que el **NLinesV** alcanza el valor de 503 ó 1F7H, entonces este proceso de almacenamiento se suspende asignando 0 a **DigitalOF** como se ilustra en la figura 3.26.

La cantidad de líneas almacenadas del campo impar son 246[52]. El almacenamiento de una línea de video inicia después de digitalizar el píxel número 23 de la propia línea, la diferencia con 503 proporciona 480 que es el número de píxeles por línea que contiene la imagen capturada, este número está en función de la frecuencia utilizada por el ADC.

[52] *En total se almacenan 492 líneas de video por imagen, este número varia de acuerdo al estándar y la cantidad de campos entrelazados.*

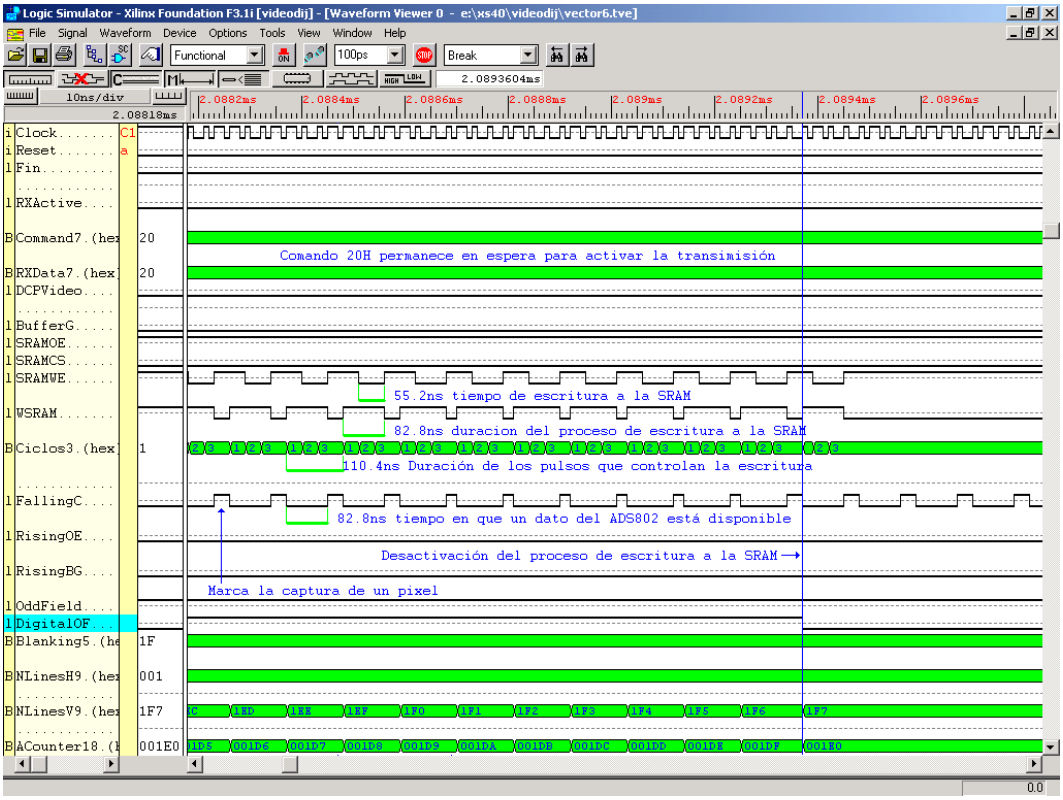

Figura 3.26. Diagrama de tiempos del almacenamiento de los últimos bytes de una línea de video.

Para cambiar la resolución de la imagen es necesario cambiar esta frecuencia de reloj y la proporción se encuentra de la siguiente forma:

$$55us * Frecuencia\_de\_reloj\_del\_ADC = Número\_de\_pixeles\_horizontales \qquad \textbf{Ecuación (3.3)}$$

En donde 55µs representa el tiempo activo en una línea de video, por lo cual al utilizar la formula de la ecuación 3.3 con una frecuencia de 9MHz, el resultado es de 495 píxeles de los cuales se emplean 480 en las imágenes almacenadas ya que al inicio y final de una línea puede haber franjas de negro. Los 480 píxeles contabilizados por **ACounter** se muestran en la figura 3.26 como 1E0.

La siguiente condición permite detectar el intervalo de blanqueado vertical localizado al final del campo impar y antes del campo par, se cumple cuando; **DigitalOF** tiene el valor de 1, **NLinesH** se encuentra entre las líneas 246 y 263 que pertenecen al intervalo de la información de video y **NLinesV** es menor que 503 además de que **FallingC** detecte un nuevo flanco de caída en la señal de reloj del ADC.

Cuando las señales anteriores reciben los valores mencionados entonces es posible realizar la digitalización de las líneas de video horizontal, sin embargo éstas no se almacenan en memoria, como se muestra en la figura 3.27. Finalmente **ACounter** toma el valor de 0 cada vez que se cumple esta condición y **DigitalOF** recibe el valor de 0 inhabilitando el proceso de digitalización del intervalo de blanqueado vertical.

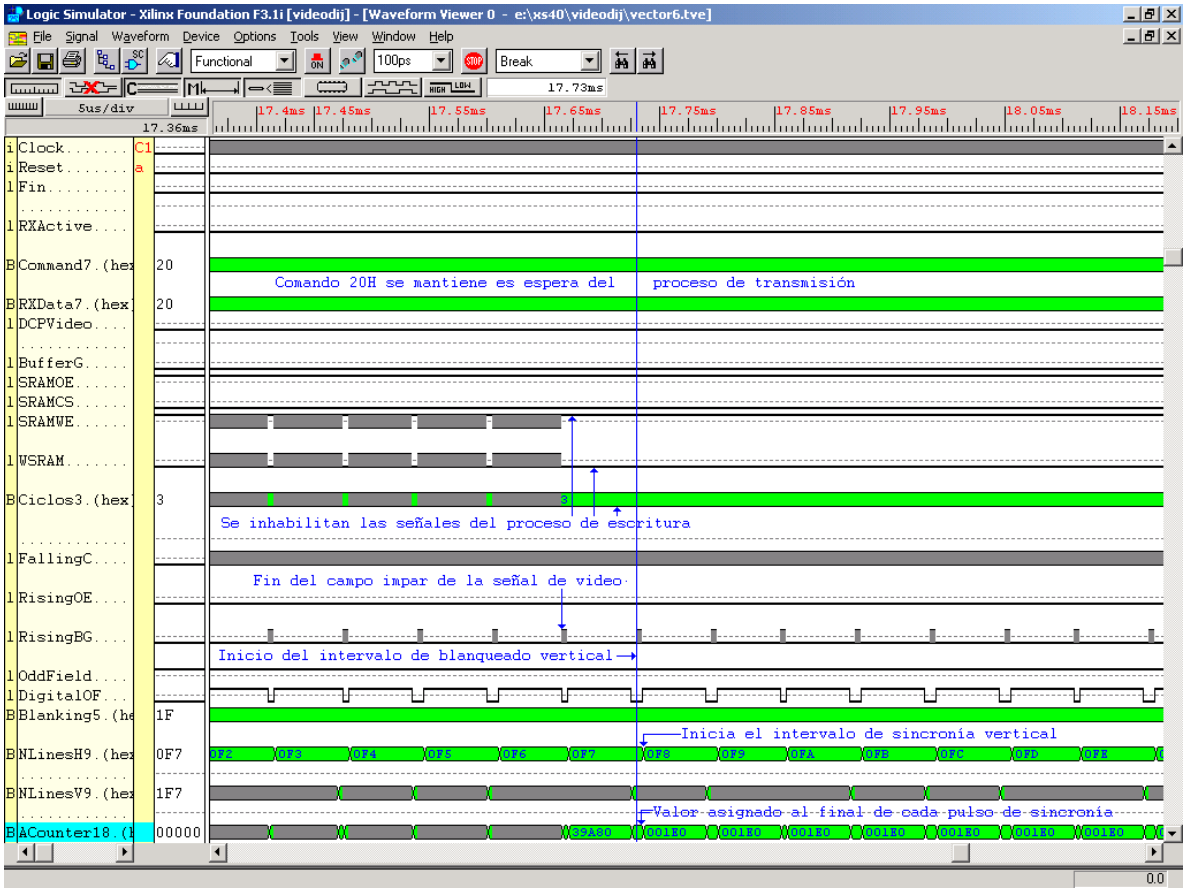

Figura 3.27. Diagrama de tiempos al final de un marco impar.

El siguiente paso tiene como función digitalizar las 245 líneas de video del campo par, se cumple cuando **DigitalOF** tiene el valor de 1, **NLinesH** se encuentra entre 262 y 508, además de que cada píxel se digitalizara si ocurre un flanco de caída del reloj de conversión del ADC, es decir que **FallingC** reciba el valor de 1 y si **NLinesV** es menor a 503. Cuando esta condición se cumple han transcurrido alrededor de 30 pulsos del intervalo de sincronía vertical, este número no es exacto debido a lo siguiente, el método de digitalización empleado consiste en esperar el tiempo correspondiente a 17 líneas de video y no pulsos de sincronía vertical ya que estos últimos no duran el mismo tiempo, por eso son 17 pulsos los que transcurren en el intervalo de sincronía vertical y no 30, como se ilustra en la figura 3.28. El procedimiento de escritura a la SRAM ocurre de igual forma como se explico para el campo impar, éste se muestra en la figura 3.26.

La siguiente condición se cumple de forma similar a la anterior, la diferencia es que ésta ocurre solo cuando **NLinesH** es igual a 508, además de que **DigitalOF** debe tener el valor de 1, **NLinesV** sea menor que 503 y **FallingC** tenga el valor de 1. Esta condición permite que se digitalice la última línea de video del campo par.

Cuando se ha terminado de digitalizar el ultimo píxel de la ultima línea de video capturada se habilita el proceso de escritura a la SRAM para almacenar este píxel como se ilustra en la figura 3.29 y también se compara **Command** = 20H si esto es verdadero significa que la PC ha enviado este carácter para iniciar la transmisión de una imagen completa, el proceso de transmisión se habilita asignando el valor de 1 a **DCPVideo.**

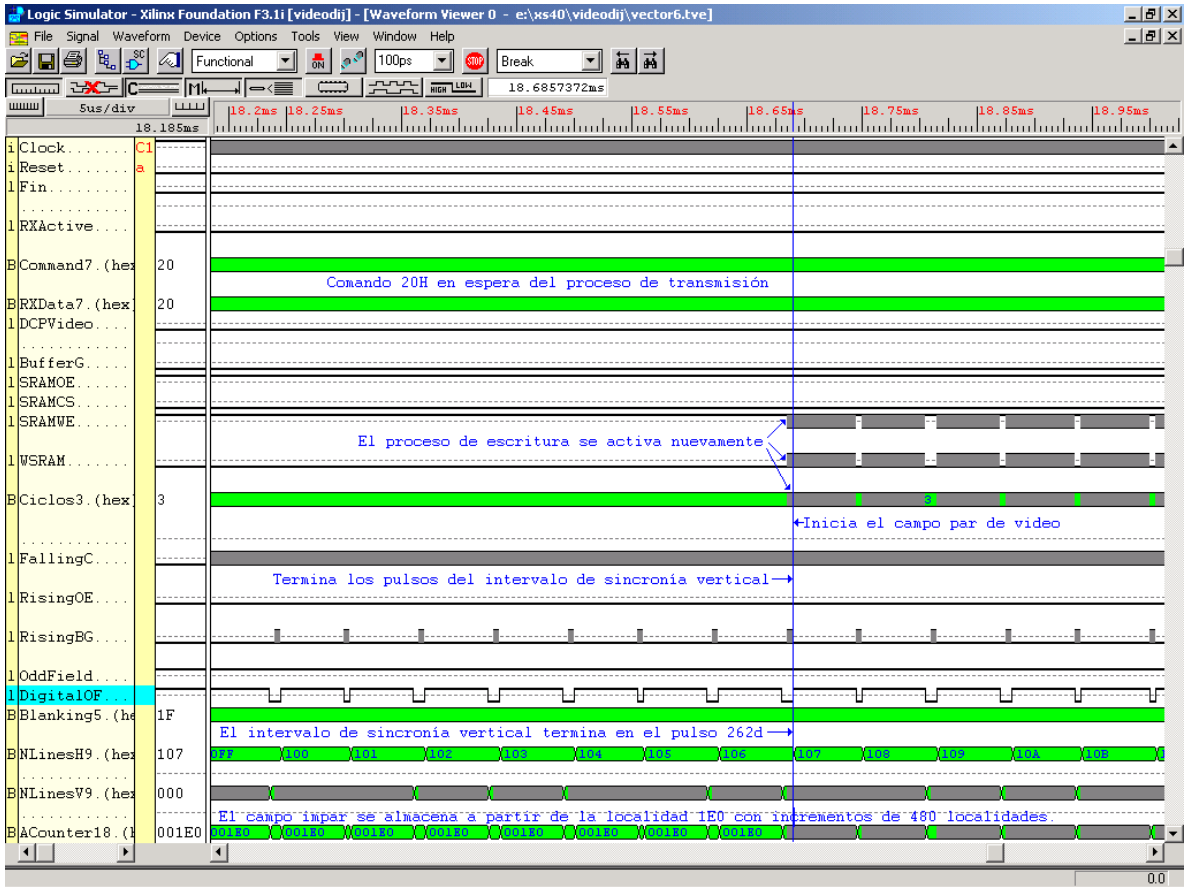

Figura 3.28. Diagrama de tiempos del campo de video par.

La siguiente condición activa la escritura a la SRAM, se habilita cuando **WSRAM** tiene el valor de 1 y **Ciclos** es mayor que 0. Este se completa en aproximadamente 110ns siendo el limite 70ns debido al tiempo de acceso de lectura o escritura a la SRAM utilizada en esta tesis, cuando transcurre el primer ciclo de reloj **SRAMWE** recibe el valor de 0 y de esta forma se marca el inicio del proceso de escritura a la SRAM como se muestra en la figura 3.26, cuando **Ciclos** tiene el valor de 2 **SRAMWE** mantiene su valor en 0 y finalmente cuando **Ciclos** tiene el valor de 3, **SRAMWE** recibe 1, indicando que el dato ha sido almacenado.

Si **DCPVideo** tiene el valor 1, **BufferG** pondrá los pines de datos correspondientes al ADC en un estado de alta impedancia, esto es porque el proceso de transmisión estará activo lo que implica utilizar un bus entre el FPGA y la SRAM, por lo cual los pines de datos utilizados por el ADC entrarían en conflicto con las conexiones de datos de los CI's antes mencionados.

Si **Fin** es igual 1 el proceso de transmisión ha terminado y entonces **DCPVideo** recibe el valor de 0, así puede volver a capturar imágenes de video, **Command** por su parte tomara el valor de 00H y esperara un nuevo comando que le indique transmitir imágenes completas.

3.2.4.2 Unidad Asíncrona de Recepción-Transmisión.

La transferencia de imágenes digitalizadas y posteriormente procesadas hace uso del protocolo de comunicaciones RS-232 a través del puerto serie de la PC, empleando el código escrito en VHDL es posible cambiar los rangos de transferencia de baudios, el bít de parada e inicio, y los bits de paridad que son los elementos más importantes en este protocolo de comunicación.

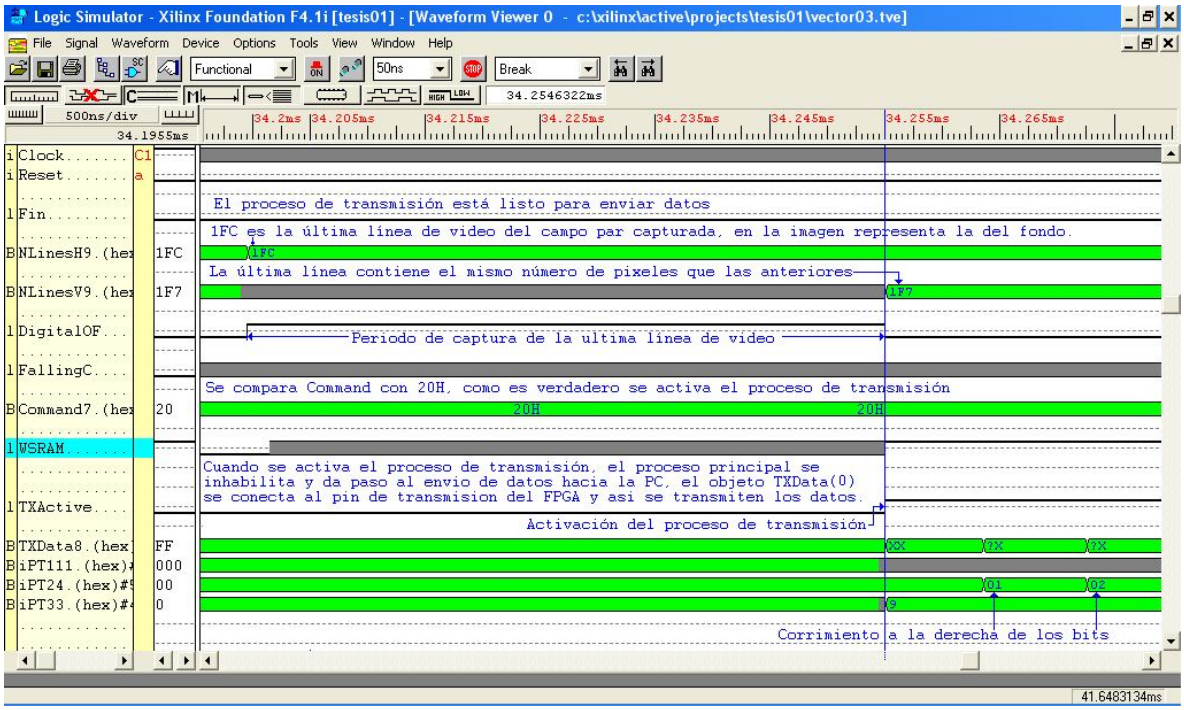

Figura 3.29. Diagrama de tiempos para el protocolo RS-232.

3.2.4.2.1 Proceso de recepción RS-232

Al igual que el proceso principal, ejecuta una instrucción cada vez que ocurre un flanco de subida en el oscilador maestro, éste proceso se ilustra en la figura 3.30.

La primera decisión que toma éste es si **Reset** es igual a 1 si esto sucede, **iPR1** recibe 0, este objeto cuenta cada ciclo del reloj maestro, **iPR2** toma el valor de 0, se incrementa cada vez que transcurre el tiempo de duración de un bít en el protocolo RS-232, **RXData** recibe 0FFH, este objeto almacenara el comando enviado por la PC. **RXActive** también recibe 0 y sirve para indicar que se está recibiendo un carácter de la PC. **RXDelay** recibe 0 y sirve para tomar una muestra a la mitad del valor[53] del bít en turno.

[53] *La forma de digitalizar un bít en una transmisión RS-232 es dividirlo en 16 muestras, de las muestras 8, 9 y 10 el bít recibe el valor de 1 si al menos dos muestras tienen ese valor de otra forma el bít recibirá 0.*

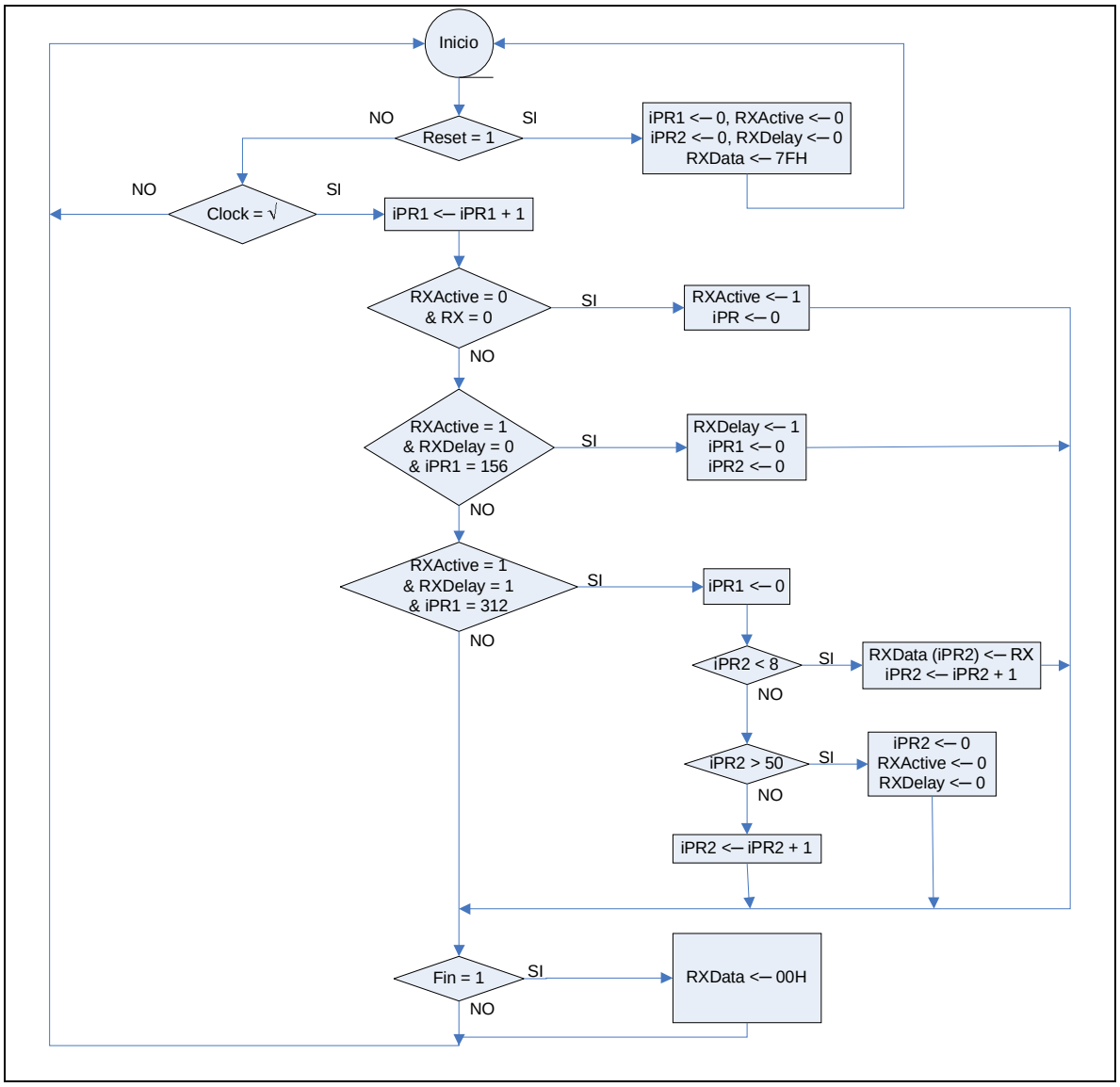

Figura 3.30. Proceso de recepción RS-232.

Figura 3.31. Diagrama de recepción.

La siguiente instrucción se cumple cuando se inicia la transmisión de un carácter desde la PC, las condiciones que debe cumplir son que **RXActive** y **RX** sean igual a 0 esto se cumplirá con el inicio de la transmisión ya que la línea de transmisión de la PC hacia el FPGA se encuentra siempre en 1 y el inicio de transmisión envía como primer dato un 0 que es el bít de inicio como se muestra en la figura 3.31.

Cuando esta instrucción se cumple se activa la bandera de recepción de un carácter asignando a **RXActive** el valor de 1 como se ilustra en las figuras 3.23 y 3.31.

La siguiente condición se utiliza para posicionarse a la mitad de la duración de 1 bít para tomar una muestra como se había mencionado, se cumple cuando **RXActive** se ha activado, **RXDelay** esta desactivado ya que no ha transcurrido el tiempo de espera e **iPR1** es igual 156. 156 corresponde a la mitad del tiempo que dura un bít en una transmisión con velocidad de 115200 bps (bits por segundo), la forma de obtener este valor es dividiendo la frecuencia del reloj maestro entre la velocidad de transmisión deseada y luego dividirlo entre 2 ya que corresponde a la mitad de un bít como se muestra en la ecuación 3.4. La condición termina activando a **RXDelay** y asignando valores iniciales a los objetos **iPR1** e **iPR2**.

$$\frac{Frecuencia\_del\ reloj\_maestro}{Velocidad\_de\_transmisión * 2} = Ciclos\_de\_retardo \qquad \textbf{Ecuación (3.4)}$$

Cuando se ha activado la recepción de un carácter y ha transcurrido el tiempo de retardo, la posición que se ha alcanzado es la mitad del tiempo que dura el bít de inicio, es necesario esperar 312 ciclos de reloj más, para almacenar la muestra del bít menos significativo, después de transcurrir los siguientes 312 ciclos es posible tomar la muestra del segundo bít y así sucesivamente. Los 8 bits se almacenan en **RXData** teniendo en su localidad 0 el bít LSB y en su localidad 7 el bít MSB, después de que se han capturado los 8 bits el proceso de recepción esperara aproximadamente 370µs aunque podría esperarse menos tiempo, sin embargo aquí se ha utilizado este valor para asegurar el dato, como se ilustra en la figura 3.31. El diagrama de flujo solo funciona para una transmisión de 115200, 8 bits, 1 bít de inicio y 1 bít de parada, se pueden agregar mas características a la transmisión modificando este diagrama.

Fin ayuda al proceso de recepción a saber cuando el proceso de transmisión ha terminado de enviar a la PC una imagen completa, ya que cuando recibe el valor de 1 se asigna a **RXData** el valor inicial de 00H con el fin de que **Command** no active una segunda transmisión de la misma imagen, debido a que **Command** después de un pulso del reloj maestro adquiere el valor de **RXData** como se puede ver en la figura 3.14.

3.2.4.2.2 Proceso de transmisión RS-232

Al igual que los procesos anteriores este ejecuta una instrucción por cada flanco de subida del reloj maestro, este proceso se ilustra en la figura 3.32.

Si **Reset** es igual a 1 los objetos controlados por este proceso toman valores iniciales, **ACounterT** se posiciona en la dirección 0, **SRAMCST** se habilita recibiendo el valor de 0, **SRAMOET** se deshabilita, **SRAMWET** se deshabilita ya que solamente se realizara lectura a la SRAM, **BufferGT** se deshabilita debido a que las líneas de datos provenientes del ADC en este momento no se utilizaran.

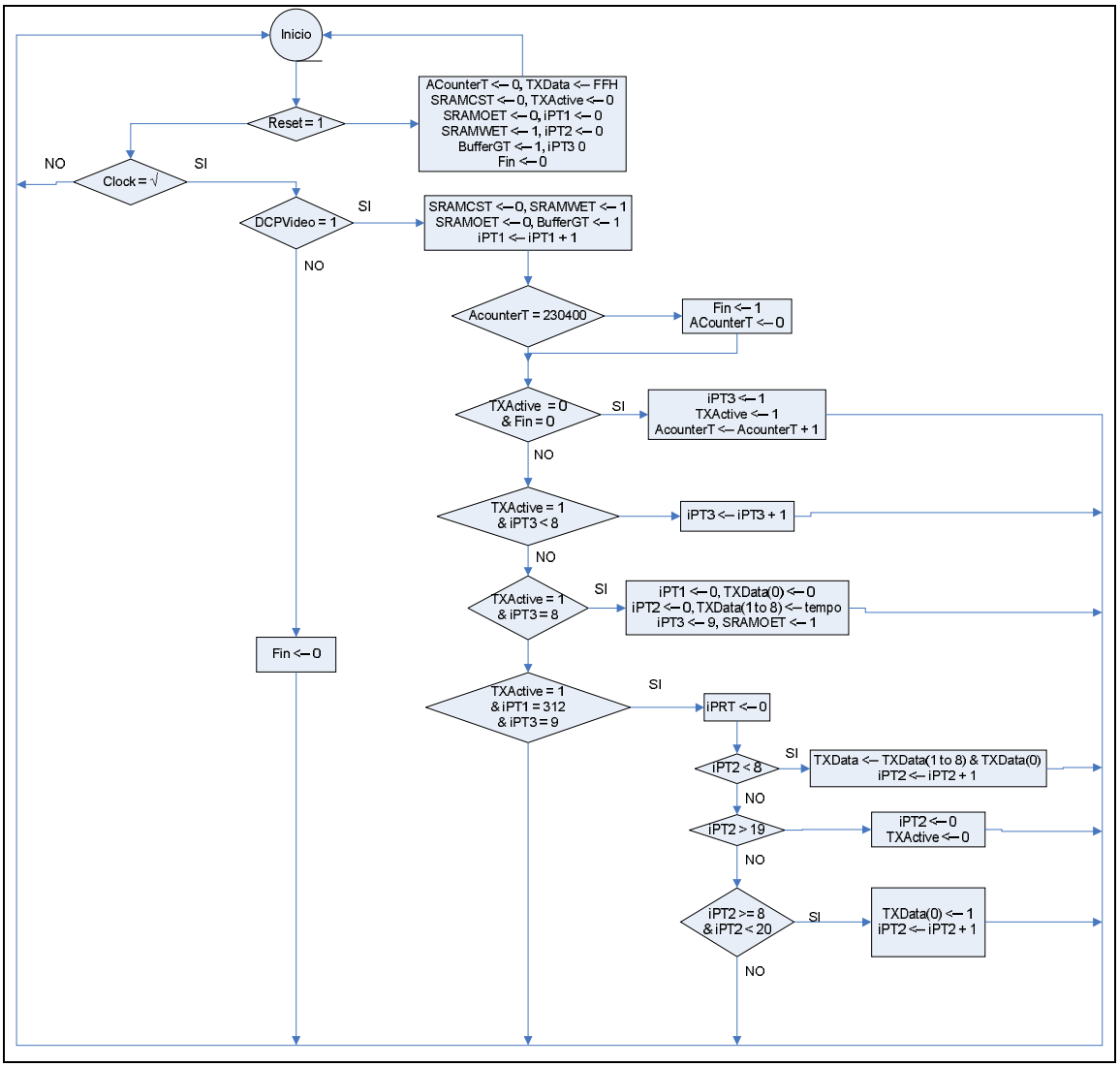

Figura 3.32. Proceso de transmisión RS-232.

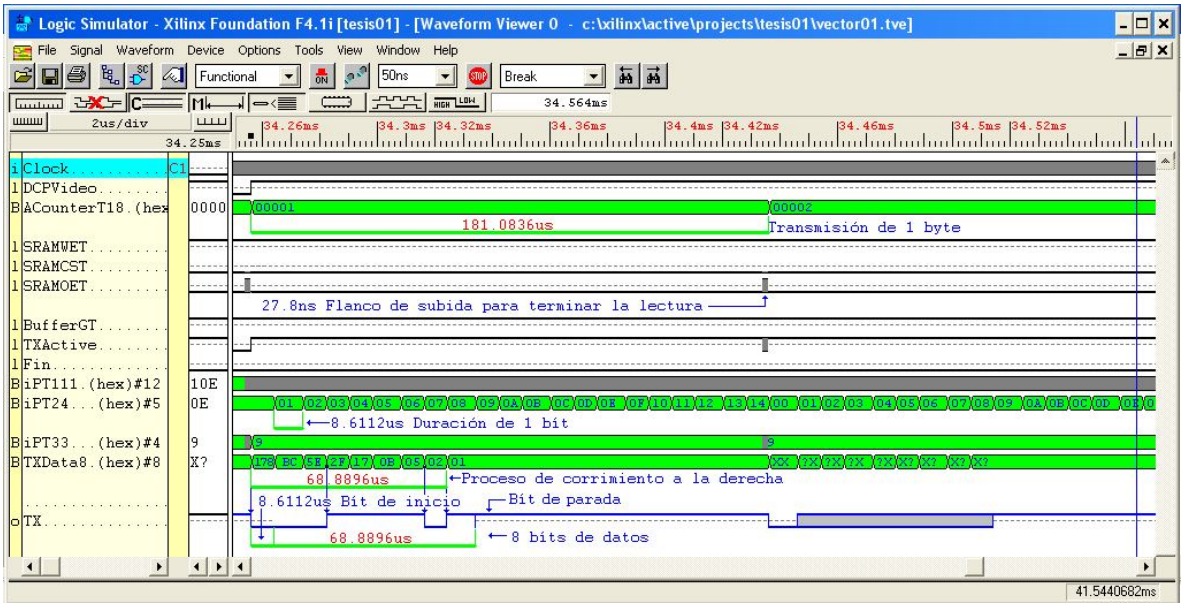

Figura 3.33. Proceso de transmisión.

TXData que acomoda los bíts para ser transmitidos recibe 1FFH y es que por cualquier motivo la línea de transmisión del FPGA se debe mantener con 1, **TXActive** se inhabilita y así el proceso de transmisión no está enviando datos, los objetos **iPT1**, **iPT2** e **iPT3** reciben 0 para iniciar una nueva cuenta en caso de ser requeridos. **Fin** recibe 0 lo que permite al proceso de transmisión estar en espera de un comando que solicite una transmisión de imagen, como se ilustra en la figura 3.33.

Si **DCPVideo** recibe 1 el proceso principal detiene la captura de imágenes completas debido a que se ha recibido un comando de transmisión y entonces se habilita este proceso, como se ilustra en la figura 3.33. Se prepara entonces la lectura a la SRAM asignando los valores apropiados a **SRAMWET**, **SRAMCST**, **SRAMOET** y **BufferGT**, además de que **iPT1** inicia su conteo.

La primer comparación se hace para saber si ha terminado de enviar el píxel 230400 que es la cantidad de píxeles contenidos en una imagen, si esta condición se cumple **Fin** recibe 1 y **ACounterT** recibe el valor de 0 para iniciar el almacenamiento de una nueva imagen.

La transmisión se activa cuando **TXActive** y **Fin** son iguales a 0, asignando 1 a **TXActive**, **iPT3** cuenta 9 ciclos del reloj maestro para asegurar que la localidad requerida envié los datos hacia el FPGA cuando **SRAMOET** cambie su valor de 0 a 1.

ACounterT se incrementa, obsérvese que la primera localidad de memoria no almacena información de video, como se muestra en la figura 3.33, las direcciones utilizadas para el almacenamiento iniciarán de 00001H hasta 38400H. La siguiente condición solamente contabiliza los ciclos antes mencionados incrementando 1 en cada ciclo de reloj a **iPT3**.

Cuando se han cumplido los 9 ciclos de **iPT3** se pueden leer los datos de la SRAM asignando 1 a **SRAMOET,** para cumplir con lo requerido por el diagrama de lectura de la SRAM en la figura 3.17(b). Los contadores **iPT1** e **iPT2** toman valores iniciales mientras que **iPT3** se inhabilita asignándole 9. **TXData** recibe en su localidad 0 el valor de 1 que representara el bít de inicio y las localidades del 1 al 9 reciben el valor leído de la SRAM.

La última condición se cumplirá hasta que el contador **iPT1** deje transcurrir 312 ciclos de reloj que corresponde al tiempo de duración de 1 bít en una transmisión de 115200 bps, este número de ciclos se obtiene de la ecuación 3.4 sin la división entre 2.

Cuando la condición anterior resulta verdadera **iPT1** recibe 0 para estar preparado en un nuevo conteo, si **iPT2** tiene valores del 0 al 7, entonces **TXData** realiza un corrimiento hacia la izquierda de sus localidades 1 al 8 asignándolos a la localidad 0 que se conecta con el pín TX del FPGA y por cada corrimiento de bít realizado **iPT2** se incrementa en 1, después de haber transmitido los 8 bits y el bít de inicio (primeros 312 ciclos de reloj para el caso de transmisión a 115200bps), se transmiten 12 bits con valor de 1 para asegurar una buena transmisión, es decir que la PC no se confunda al recibir datos enviados desde el FPGA y finalmente después de haber enviado los 20 bits **iPT2** y **TXActive** reciben 0 preparándose para la transmisión de un nuevo dato o para finalizar la transmisión.

3.2.4.3 Controlador del ADC

Este módulo genera la señal de reloj para que el ADC realice la digitalización de la señal analógica de video a la entrada, también proporciona la señal de habilitación de salida la cual sincroniza el uso de los datos a la salida del ADC para almacenarlos en la SRAM o la salida de datos de la SRAM para enviarlos hacia la PC.

3.2.4.3.1 Generación de la señal de reloj para el ADC

El proceso que genera la señal de reloj del ADC se ilustra en la figura 3.34. Esta señal de reloj se genera mediante un contador llamado **Counter1,** éste cuenta 2 pulsos del ciclo de reloj maestro y genera medio ciclo de la señal de reloj del ADC, el otro se genera con los siguientes 2 ciclos del reloj maestro. Para generar los cambios de nivel de 0 a 1, luego a 0 y así sucesivamente **TConvert** realiza una operación XOR de su valor actual con 1, de esta forma si el valor actual es 0 el resultado será 1 y si su valor actual es 1 el resultado será 0 como se muestra en la figura 3.35.

Tabla 3.6. Counter1 en función de la resolución de imagen.

Valor de Counter1	*Frecuencia de reloj del ADC (MHz)*	*Resolución horizontal de la imagen*
2	9.0	495
3	6.0	330
4	4.5	247
5	3.6	198

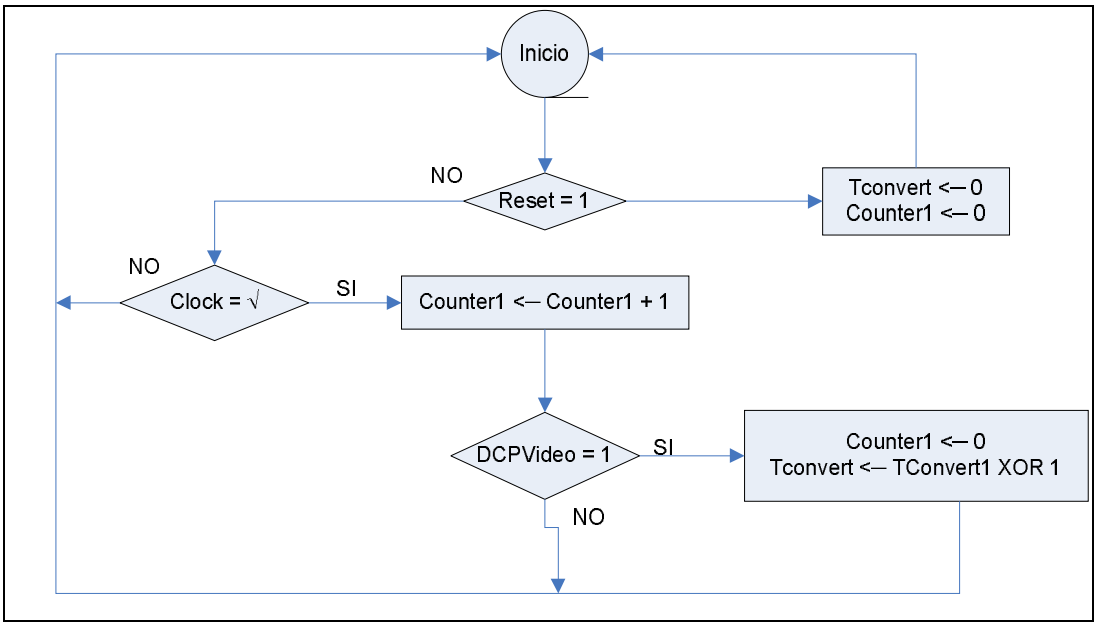

Figura 3.34. Proceso de la señal de reloj para el ADC.

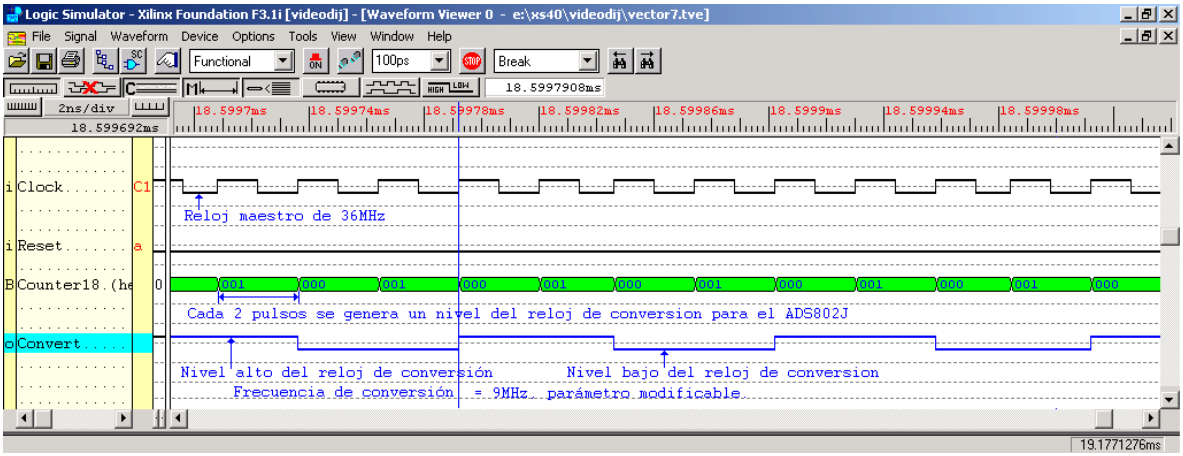

Figura 3.35. Señal de reloj del ADC.

Al cambiar el valor de comparación con **Counter1** se pueden generar otras frecuencias y así cambiar la resolución de la imagen como muestran los resultados de la tabla 3.5 obtenidos mediante el empleo de la ecuación 3.3.

3.2.4.3.2 Proceso de adquisición

La señal **BufferG** se pone a 0 mientras se realice la digitalización y almacenamiento de la señal como se muestra en el diagrama de flujo ilustrado en la figura 3.21, **BufferG** corresponde a la señal de habilitación de salida del ADC, en las figuras 3.23 a 3.28 se puede observar que la señal **BufferG** se mantiene en 0 para todas las actividades del proceso maestro.

La figura 3.24 muestra los diagramas de tiempo para la adquisición del inicio de un marco de video impar y la figura 3.27 muestra los diagramas de tiempo para el final del campo de video impar, mientras que en la figura 3.28 se ilustran los diagramas de tiempos para el inicio del campo de video par, en los cuales la señal **BufferG** se mantiene en 0. Por otra parte en las figuras 3.25 y 3.26 se ilustran los tiempos en ciclos de reloj para la escritura a la SRAM para el inicio y fin de una línea de video horizontal.

3.2.4.4 Controlador del sistema de memoria

En este proceso se generan los diagramas de tiempos ilustrados en la figura 3.17 de la sección 3.2.3.3.2 que corresponde a los procesos de escritura y lectura a la SRAM, como se especifica en la tabla 3.5 un ciclo de escritura o lectura se realiza en mínimo 70ns, durante este intervalo de tiempo las señales **SRAMCS**, **SRAMOE** y **SRAMWE** se combinan de acuerdo a los tiempos de la figura 3.17 para implementar los módulos en lenguaje VHDL de escritura y lectura a la SRAM.

3.2.4.4.1 Proceso de escritura

El proceso de escritura de acuerdo a los tiempos establecidos en la figura 3.17(b) se implementa en código VHDL de la forma ilustrada en la figura 3.26, **SRAMCS** tiene asignado el valor 0 durante todo el proceso de escritura para que la SRAM este activa, **SRAMOE** recibe el valor de 1 inhabilitando los datos a la salida y permitiendo que los datos provenientes del ADC puedan almacenarse, finalmente **SRAMWE** toma el valor de 0 durante 55ns que es mayor al tiempo necesario para habilitar la escritura de una localidad en la SRAM y luego toma el valor de 1 durante 55ns.

Para una frecuencia de oscilador de 36MHz se necesitan dos ciclos de reloj para obtener los 55ns, sin embargo para una frecuencia de operación de 50MHz son necesarios 3 ciclos de reloj que proporcionan una duración de 60ns, en ambos casos el tiempo mínimo t_{pw} se cumple permitiendo así la escritura a la SRAM.

3.2.4.4.2 Proceso de lectura

El proceso de lectura implementado en código VHDL se ilustra en la figura 3.33, la señal **SRAMCST** recibe el valor de 0 y lo mantiene durante todo el proceso de lectura de la SRAM para que el CI este activo, **SRAMWET** recibe 1 inhabilitando el proceso de escritura y la señal **SRAMOET** permite que los datos de cada localidad de la SRAM estén disponibles durante el tiempo de transferencia de bytes, es decir, para una velocidad de transmisión de 115200 y una frecuencia de operación del FPGA de 36MHz cada dato está disponible durante 8.68μs después de este tiempo durante 27.8ns la señal toma el valor de 1 dando por terminada la lectura de una localidad de la SRAM.

3.2.5 Interfaz de usuario

El puerto serie RS-232 que se emplea en las computadoras, PC's, módems, conmutadores, terminales, impresoras, fue desarrollado en los años 60´s por la EIA, creado para ofrecer una interfase entre aparatos que requieren comunicación de datos, como los anteriormente mencionados. Durante los últimos 30 años que este estándar ha estado en uso, los equipos han evolucionado tremendamente, sin embargo, la norma inicial RS-232 ha cambiado muy poco y estos cambios son normalmente debidos a la interpretación propia de algunos fabricantes.

Para establecer una comunicación mediante serie de datos ó información, se requiere un mínimo de dos alambres. Si se desea tener una comunicación bidireccional por un par de hilos y está consiste en una serie de bits de información, se requieren otras terminales que indiquen a la interfase, cuál de los aparatos interconectados transmite y cuál recibe, que tipo de información es, cuando el aparato receptor está listo para recibir, cuando el transmisor esta listo para transmitir, a que velocidad va ser la comunicación, etc., esto hace que el puerto serie tenga otras terminales que se usan para coordinar la comunicación entre los equipos.

El puerto RS-232 opera con tensiones entre +12V y -12V, ya que un cero lógico es cuando la terminal este entre +9 y +12V, y un uno lógico cuando está entre 9V y 12V, de manera que un puerto serie que no está transmitiendo, mantiene la terminal de transmisión en un 1 lógico es decir entre -9 y -12V. El conector estándar RS-232 sea este hembra ó macho, es el DB-25.

3.2.5.1 La estructura de datos del puerto serie RS-232

La comunicación de datos en un puerto serial, se usa normalmente para efectuar comunicaciones asíncronas, es decir, sin tiempo preestablecido para iniciarse, en donde los datos llegan en ráfagas ó paquetes de información, normalmente cada paquete es de 8 bits =1 byte (equivalente a un carácter en código ASCII), algunos equipos envían carácter por carácter, otros guardan muchos caracteres en la memoria y cuando les toca enviarlos, los envían uno tras otro.

Uno de los parámetros más importantes en la comunicación serie, es la velocidad con la que los datos se transmiten, para el caso del RS-232, pueden transmitir de los 300 Baudios (1 Baudio=1 bit/seg) hasta 115,200 Baudios, la velocidad depende de los equipos conectados en el puerto serie y la calidad y longitud de los cables. Otro de los parámetros de trascendencia es el bit de inicio, es decir, el bit que le indica al puerto receptor que va a llegar un Byte de información.

Hay dos tipos de paridad adicional que se usan y estos son: Mark (marca) es el bit de paridad que se intercala y siempre es un uno, space (espacio). El bit de paridad que se intercala siempre es un cero y el número de bits que se emplean para cada paquete, pueden ser 5, 6, 7 u 8.

Es así como la comunicación serie RS-232 es la comunicación de Datos más empleada en el mundo, ya que utiliza pocos alambres ó cables para lograrlo y mediante los módems, es la forma de intercomunicar computadoras, comunicarse a través de Internet, control a distancia y muchas otras aplicaciones. El estándar RS-232-EIA, es equivalente al V.24 del Comité internacional consultivo de telégrafo y teléfono (CCITT, *Consultive Commitee Internacional Telegraph and Telephone*), este comité es internacional y hace recomendaciones de carácter mundial.

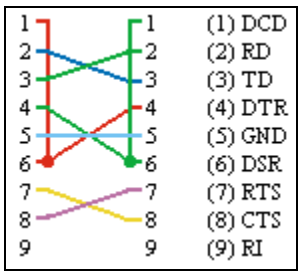

Figura 3.36. Conexiones para la interfase del puerto RS-232.

3.2.5.2 Interfase RS-232 de la PC con el digitalizador de video

La conexión del sistema a la PC se realiza a través de la configuración ilustrada en la figura 3.36. En esta figura se ilustran los pines correspondientes al puerto RS-232 de la PC. La interfaz RS-232 dispone de hasta 25 líneas que están orientadas a la comunicación de dos equipos PC. En la figura 3.36 solo se ilustra un diagrama de conexiones de 9 pines, estos se describen de la siguiente manera:

- **Línea de transmisión de datos (TD).** Línea por la que la PC envía los datos.
- **Línea de recepción de datos (RD).** Línea por la que la PC recibe los datos.
- **PC preparada (DTR y DCD).** Línea por la que la PC indica al sistema que está activo para comunicarse con él.
- **Sistema preparado (DSR).** Línea por la que el sistema indica a la PC que está activo para establecer la comunicación.
- **Petición de envío (RTS).** Con esta línea, la PC indica al sistema que está preparado para transmitir datos.
- **Preparado para enviar (CTS).-** Tras un RTS, el sistema pone esta línea en 1 lógico, tan pronto como está preparado para recibir datos.
- **Masa (GND).** Necesaria para que tenga lugar la transmisión.

Estas líneas son controladas mediante la programación de los registros del UART, que es un chip especial para la entrada y salida de caracteres y sobre todo, para la conversión de palabras de datos en las correspondientes señales del puerto serie.

Sin embargo es posible implementar el protocolo de comunicaciones RS-232 con solo tres líneas de las antes mencionadas, éstas son: TD, RD y GND. El diagrama de tiempos que estas tres líneas deben de cumplir para realizar la comunicación se ilustra en la figura 3.37.

El diagrama para la transmisión de un carácter desde el FPGA es un poco diferente del anterior, ya que la tensión que representa al 1 lógico es de +5V y el que representa al 0 lógico es 0V. El proceso de conversión de tensiones se realiza a través del CI SIP233ACP como se había explicado en el apartado 3.4.2.2.

3.2.5.3 Programa de comunicación serial

El envío de comandos para activar la transmisión de datos de una imagen completa se realiza a través de un programa realizado en BorlandC, se eligió este lenguaje por ser de más bajo nivel que algún lenguaje visual como Visual Basic u otro, con la inconveniencia de perder presentación.

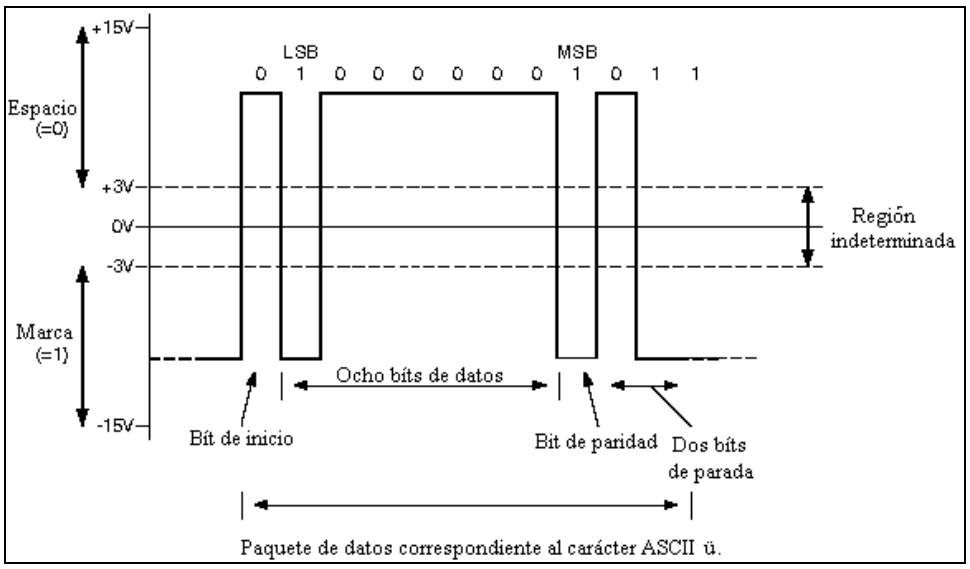

Figura 3.37. Diagrama de un carácter enviado desde la PC vía RS-232.

Figura 3.38. Propiedades de la comunicación serial.

Generalmente una PC tiene uno o dos puertos RS-232. El puerto 1, también conocido como COM1 tiene la dirección base 3F8 y el COM2 la 2F8, la configuración de un puerto se realiza a través de la programación de sus registros de control. Cada uno de los puertos COM tiene 7 registros.

La pantalla de configuración del puerto se muestra en la figura 3.38, la selección de características en esta pantalla debe coincidir con el programa en VHDL del FPGA.

3.2.6 Integración del digitalizador de video

El digitalizador de video complementado con el dispositivo optoelectrónico se ilustra en la figura 3.39. El dispositivo CCD empleado proporciona el formato de video NTSC mediante la norma RS-170A a través del cable RCA y comúnmente conocido como señal de video compuesta. Las imágenes digitalizadas y almacenadas en la SRAM se manipulan en formato Mapa de grises portátil (PGM, *Portable Gray Map*). El formato PGM es un formato de archivo muy básico para intercambiarse por archivos de mapas de bits a color. Sirve como un común

denominador para convertir archivos de mapas de bits entre diferentes plataformas. Es usado en el paquete de archivos de conversión GNU *netpbm* como el formato de archivo central. Este formato es mucho más simple que otros con la inconveniencia de que es menos flexible y potente, en el apéndice C se describen brevemente detalles del formato pgm.

En la figura 3.40 se muestra una imagen obtenida a través del sistema desarrollado en trabajo de tesis; estas imágenes se muestran en el formato PGM, previamente descrito; las imágenes son enviadas por el digitalizados hacia una PC a través de un puerto RS-232 con una velocidad de 38400 baudios, 8bits, 1 bít de inicio, 1 bít de parada y sin paridad. Además tiene una definición de 480x480 píxeles.

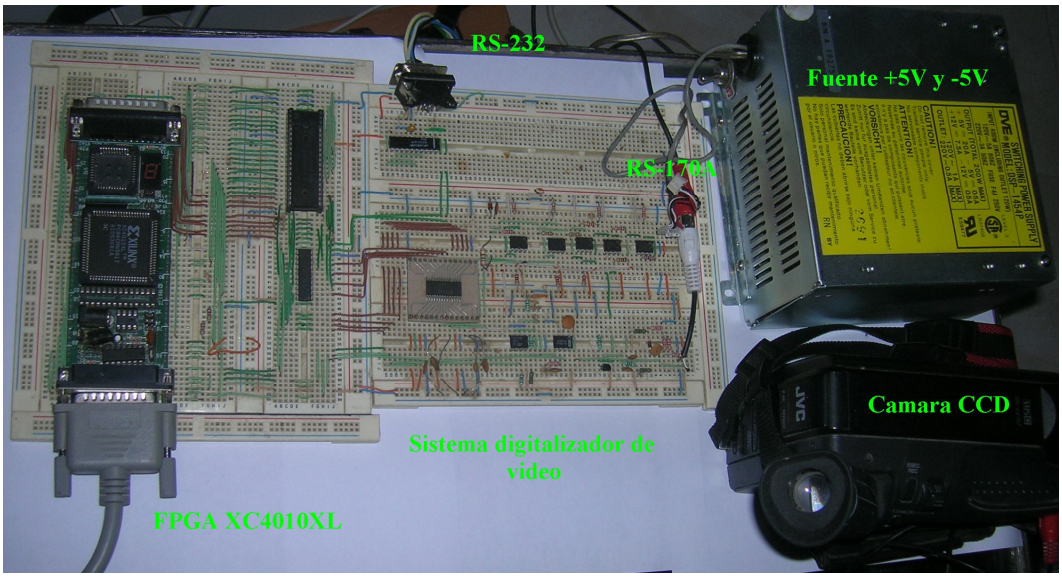

Figura 3.39 Digitalizador de video completo.

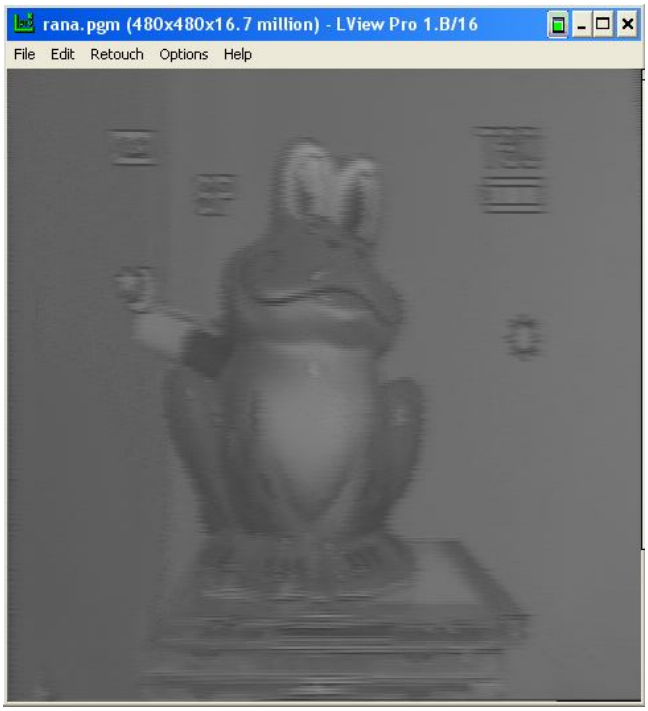

Figura 3.40. Imagen de 480x480 píxeles.

CAPÍTULO 4.

Resultados.

Este capítulo contiene los resultados de la última fase de la metodología descrita en el capítulo anterior, “Pruebas de funcionamiento del sistema”. Además se muestran posibles problemas que pueden surgir en la operación del sistema en una aplicación real. Cabe mencionar que la solución a dichos problemas no está contemplada en el presente trabajo, esto debido a que estas soluciones normalmente incluyen algoritmos de procesamiento de imágenes y el sistema propuesto limita su operación a la adquisición de las mismas. Como se ha mencionado este trabajo de tesis forma parte de un sistema más complejo, el procesamiento y análisis de la imagen es parte de etapas posteriores a la desarrollada aquí.

4.1 Enfoque de objetos.

Objetos como una cerradura dorada ilustrada en la figura 4.1, generan resplandores lo que provoca que sea difícil de definir en su forma, ya que sus lados pierdan la *continuidad de la figura* y se mezclan con colores claros, haciendo difícil de ver donde terminan los bordes de la cerradura u objeto.

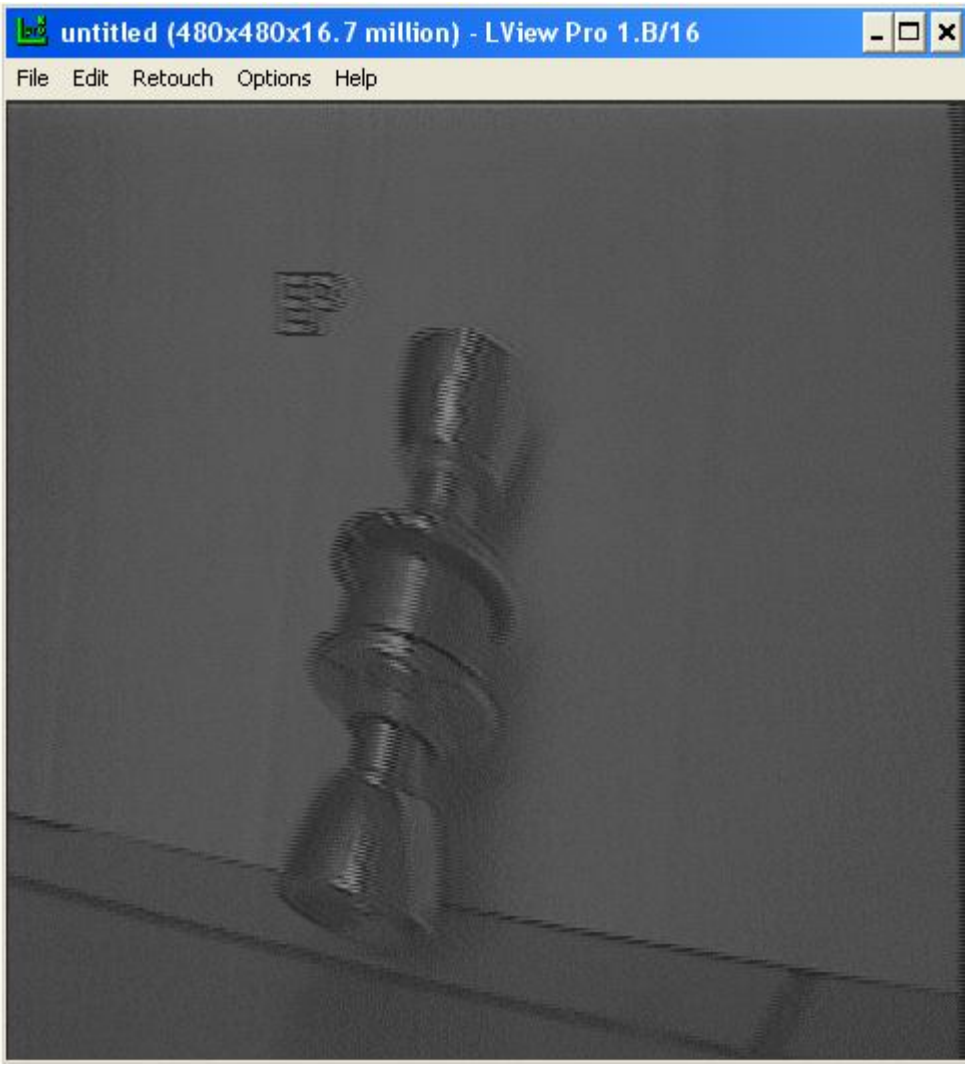

Figura 4.1 Cerradura dorada sobre fondo blanco.

Por otra parte, en ciertas aplicaciones puede resultar deseable que elementos muy pequeños, como las letras del Multimetro de la imagen 4.2, puedan ser apreciadas debido a la información de interés que pudieran contener, esta característica representa un umbral en la *definición o calidad de imagen* que se ha resuelto mediante el uso de arreglos de lentes conocidos como ZOOM. Lo que permite enfocar partes de interés en una imagen.

Figura 4.2. Multimetro.

La *profundidad de localización* de un objeto es un factor importante en la toma de decisiones de un móvil, el cual tendrá que resolver para tomar la mejor ruta. En la imagen 4.3 se puede apreciar una piedra a 30cm, una caja de software a 1.50m y a una distancia de 3m se localiza un portón, el móvil debe ser capaz de resolver que los objetos no se encuentran encimados. Esta y otras características determinaran el grado de complejidad de un móvil, al ser empleado en campo abierto.

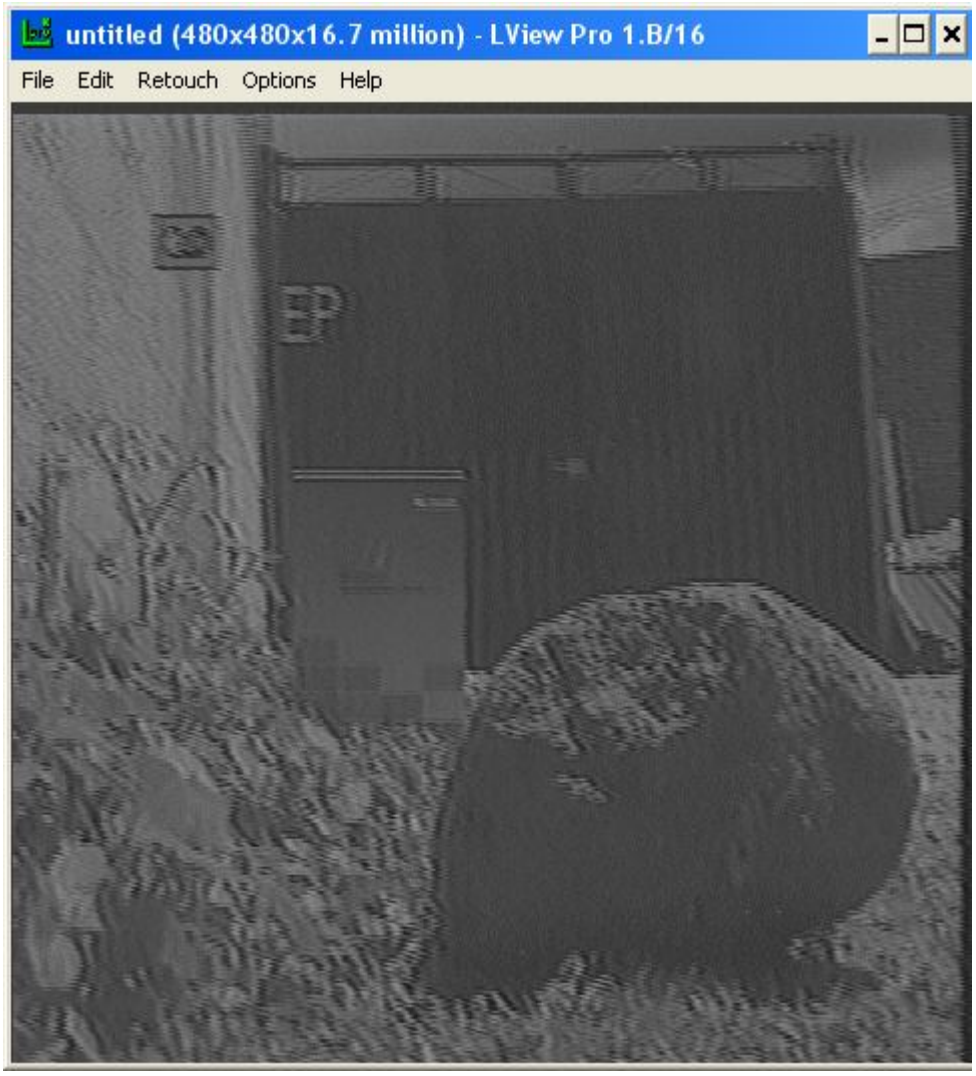

Figura 4.3 Campo abierto.

4.2 Adversidades en la digitalización.

En la captura de imágenes se pueden presentar factores adversos como el *ruido* que generalmente no provienen de la imagen si no que son causados por interferencias eléctricas en el sistema de adquisición. En la figura 4.4 se pueden observar puntos blancos en la imagen ocasionados por la interferencia de una punta de osciloscopio. Y aunque se pueden implementar algoritmos de eliminación de ruido es preferible adquirir imágenes libres de éste.

Figura 4.4. Winnie the pooh.

En la figura 4.5 se puede observar que el enfoque de objetos en primer plano genera *desenfocamiento* en otras partes de la imagen, en este caso se enfoca el centro de la imagen pero del lado izquierdo se pueden ver objetos desenfocados lo que puede confundir a los algoritmos de procesamiento en la forma del objeto.

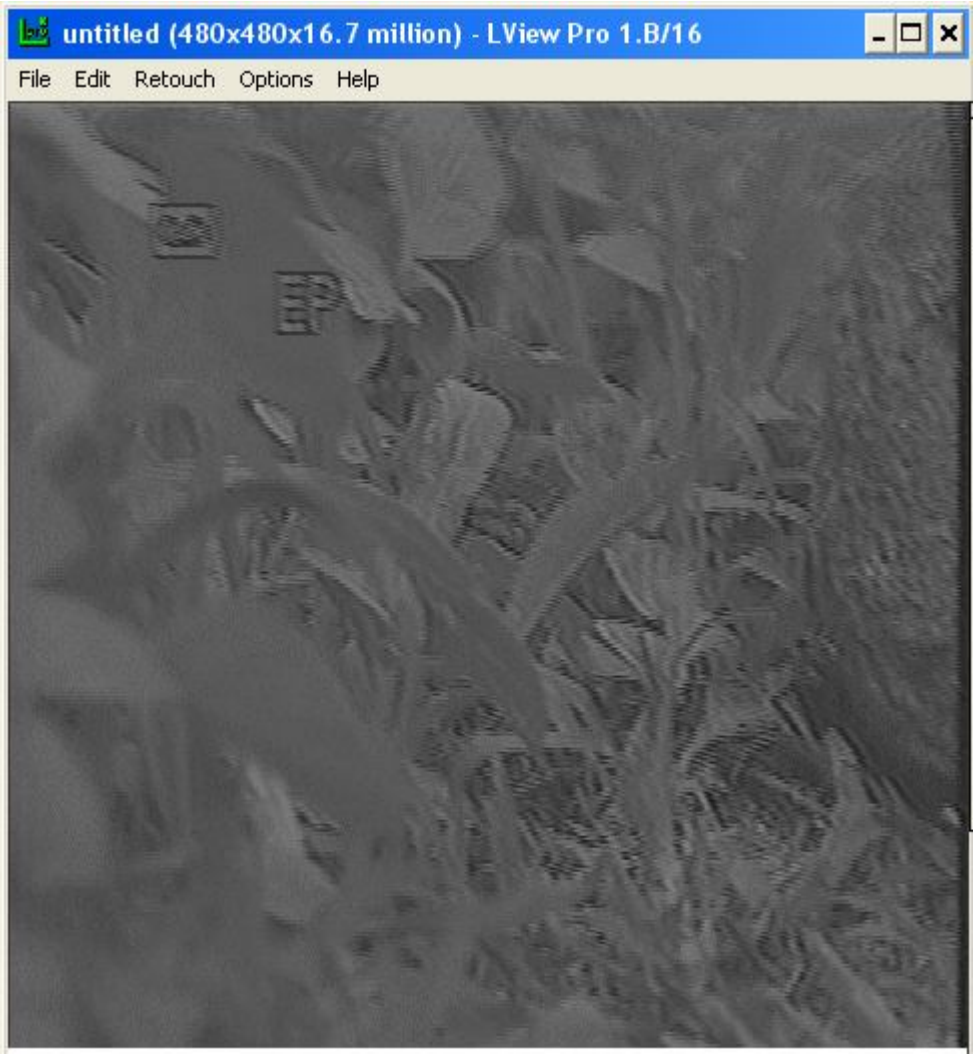

Figura 4.5 Plantas enfocadas.

La *iluminación* obviamente es un factor importante, ya que define la forma de los objetos y permite establecer umbrales en algoritmos de procesamiento como el de detección de bordes. La iluminación a bordo de sistemas móviles soluciona gran parte de problemas que tardarían en resolver los algoritmos de mejora de contraste o brillo.

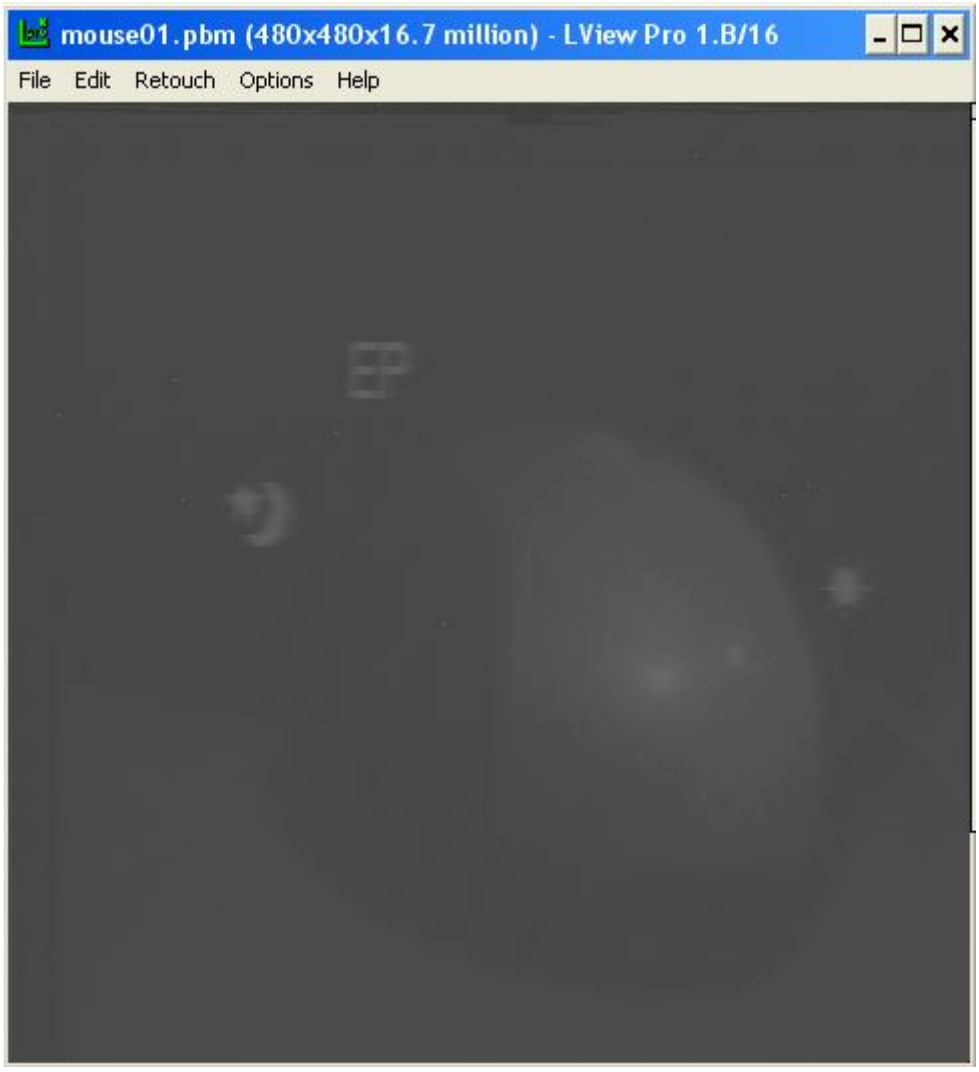

Figura 4.6. Mouse.

CÁPITULO 5

Conclusiones y trabajos futuros

5.1 Conclusiones

En este trabajo de tesis se diseñó e implementó un digitalizador de video conocido comercialmente como un *frame grabber*, el cual cumple con la norma RS-170, la entrada al sistema emplea una señal de video compuesta proporcionada por un gran número de fuentes de video con conexiones RCA.

Mediante programación en lenguaje HDL se diseñan los procesos correspondientes al protocolo RS-232, los cuáles son: digitalización de la señal de video, almacenamiento temporal en la SRAM, sincronización y procesamiento de la información de video.

Se digitalizan, almacenan y procesan marcos de video de 480 líneas horizontales y hasta de 700 líneas verticales dependiendo de la velocidad de muestreo establecida en el ADC, el tiempo de digitalización de un marco de video de 480 líneas verticales es 0.017s mientras que el tiempo de aplicación del algoritmo de Sobel resulta de alrededor de 0.45s.

La digitalización y procesamiento de la información de video se comprueba visualmente en la PC después de transferir las imágenes del digitalizador de video empleando el protocolo RS232, la transmisión de una imagen de 480x480 consume un tiempo de 20s a una velocidad de 115200 bps. Se ilustran factores que intervienen en el enfoque de los objetos como los son la continuidad de los bordes de un objeto, resolución, profundidad de localización y factores adversos como el ruido, desenfocamiento y la iluminación.

Al utilizar un dispositivo reconfigurable, FPGA, seguir una metodología de diseño y utilizar un lenguaje estándar en el modelado del sistema, el digitalizador de video diseñado fácilmente puede ser adaptado a diversas aplicaciones.

Aunque el sistema soportado por el digitalizador de video es el NTSC, también puede aceptar los formatos PAL y SECAM realizando modificaciones en la programación del FPGA y no en el hardware a diferencia de muchos sistemas comerciales.

El sistema diseñado es apto para soportar algoritmos de procesamiento y análisis digital sobre los cuadros adquiridos y almacenados en la memoria del sistema.

5.2 Trabajos futuros

El presente trabajo tiene amplias áreas de aplicación, por tanto, los trabajos futuros son bastantes; sin embargo, sólo se mencionan posibles aplicaciones en sistemas autónomos desarrollados por el grupo de colaboración.

Como se ha mencionado este trabajo de tesis forma parte de un sistema más complejo, el cual está enfocado al área de visión artificial en sistemas autónomos; surge entonces como primer trabajo, dándole continuación del presente, modelar e implementar las etapas faltantes del proyecto global, siendo estas la segmentación de imágenes y el reconocimiento de "*landmarks*".

Como posible mejora al sistema se propone sustituir la interfaz RS232 entre el digitalizador de video y la PC por una interfaz USB, esto brindará al sistema robustez, altas velocidades de transferencia y compatibilidad con equipos de cómputo actuales.

APÉNDICE A

Familia XC4000 de Xilinx

Xilinx introdujo al mercado la primer familia de FPGA's, en su serie XC2000, alrededor de 1985 y posteriormente tres nuevas generaciones: XC3000, XC4000 y XC5000. Aunque los dispositivos XC3000 son ampliamente utilizados, este diseño se enfocara a la familia XC4000 más popular. La familia XC5000 es similar a la XC4000, sin embargo esta última ha sido fabricada para ofrecer características similares a un precio más atractivo, con algunas penalidades en cuanto a la velocidad.

Xilinx ha introducido recientemente una familia basada en tecnología antifusible llamada XC9500 [12]. La familia de dispositivos XC4000 fue introducida al mercado en 1992, en su tercera generación de dispositivos, ésta se bosqueja en la figura A.1, el rango en capacidad va desde 3,000 hasta 125,000 compuertas, con retardos entre compuertas desde 2ns hasta 6ns y un costo aproximado de $20 hasta $1000 por cada dispositivo [33].

La familia XC4000 ofrece CLB's que están basados en LUT's[A.1]. Una LUT con k entradas corresponderá a una memoria de 2k-1 bíts y puede realizar cualquier función lógica de sus k entradas programando las funciones lógicas por tablas de verdad directamente dentro de la memoria, como se muestra en la figura A.2. Por otra parte, la capacidad está limitada por el número de entradas, no por la complejidad [12], [33].

A.1 *Una LUT es un arreglo de memoria con ancho de un bít, donde las líneas de dirección para la memoria son entradas de los bloques lógicos y la salida del primer bít de la memoria es la salida de la LUT.*

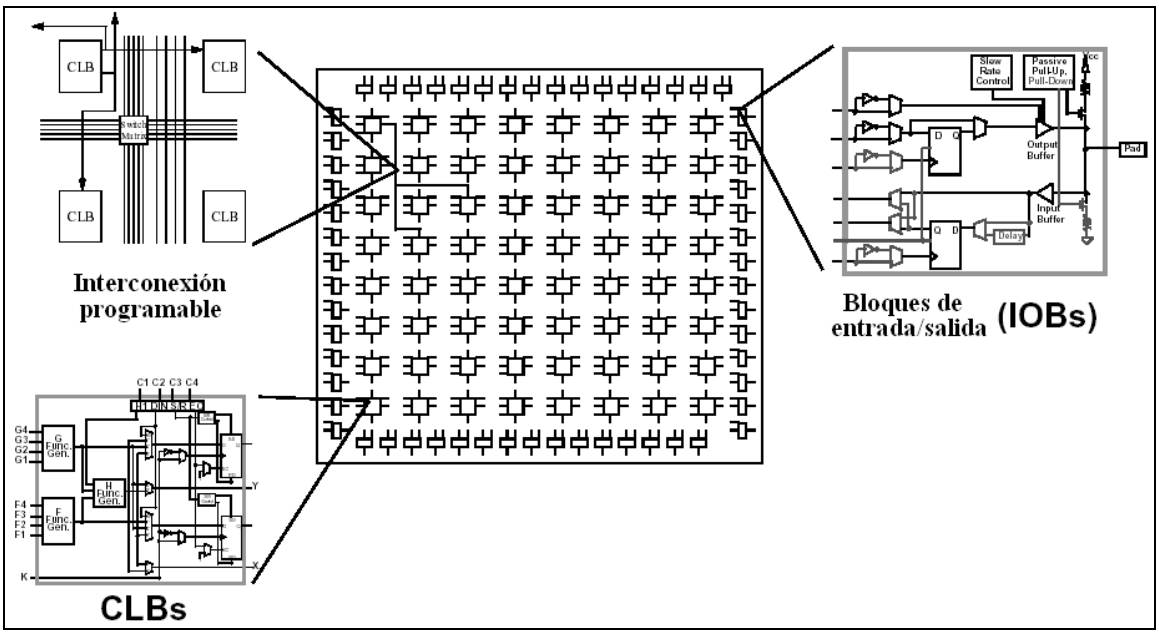

Figura A.1. Arquitectura básica de un XC4000.

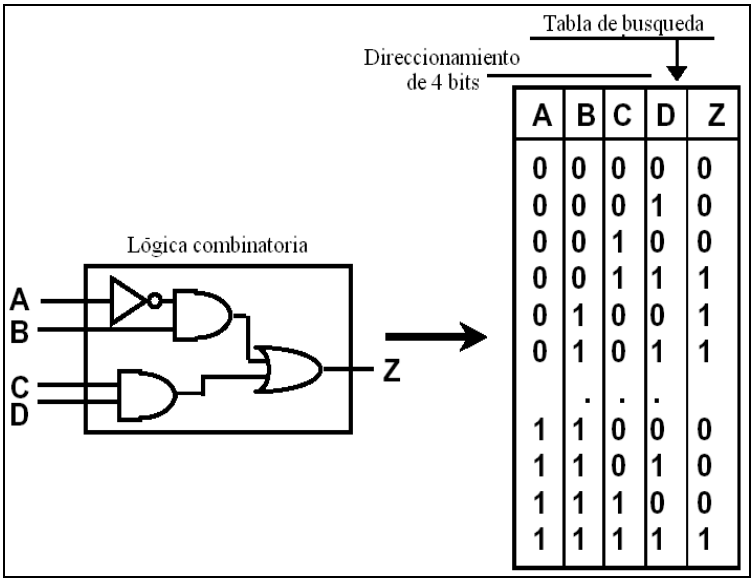

Figura A.2. Tablas de búsqueda.

Un CLB de la XC4000 contiene 3 LUT's separadas. Hay dos LUT's de 4 entradas que son alimentadas por medio de entradas CLB y la tercer LUT puede ser utilizada en combinación con las otras dos. Este arreglo permite a los CLB's implementar un amplio rango de funciones lógicas de hasta nueve entradas, dos funciones separadas de cuatro entradas u otras posibilidades.

Cada CLB contiene además dos flip-flops que pueden ser configurados como registros o latches como se ve en la figura A.3. La polaridad del reloj es independiente y puede ser utilizado de forma síncrona o asíncrona [33].

En la búsqueda de dispositivos de alta densidad que soporten la integración de sistemas enteros, los chips XC4000 adquieren características de sistemas orientados.

Por ejemplo, cada CLB contiene circuiteria que permite la ejecución eficiente de aritmética tal como un circuito que implementa un operación de acarreo lógico (*carry logic*) para circuitos sumadores típicos y además las LUT's dentro de un CLB pueden ser configuradas como celdas RAM de lectura/escritura.

La familia 4000E, tiene características adicionales en donde la RAM puede ser configurada como un puerto dual RAM con un solo puerto de escritura y dos puertos de lectura. Dentro de la 4000E, los bloques de RAM pueden ser RAM's síncronas. Además, cada chip XC4000 incluye planos AND muy amplios alrededor de la periferia del arreglo de bloques lógicos, para facilitar la implementación de bloques de circuitos tales como amplios decodificadores.

Otra característica clave de un FPGA es su estructura de interconexión además de la lógica que utiliza. La interconexión del XC4000 esta ordenada en canales horizontales y verticales. Cada canal contiene algún número de segmentos[54] de cable cortos que enlazan un CLB, segmentos más grandes que enlazan dos CLB's y segmentos muy largos que enlazan completamente el chip a lo largo y ancho. Los switches programables están disponibles para conectar las entradas y salidas de los CLB's a los segmentos de cable ó para conectar un segmento de cable a otro, como se muestra en la figura A.4 [33].

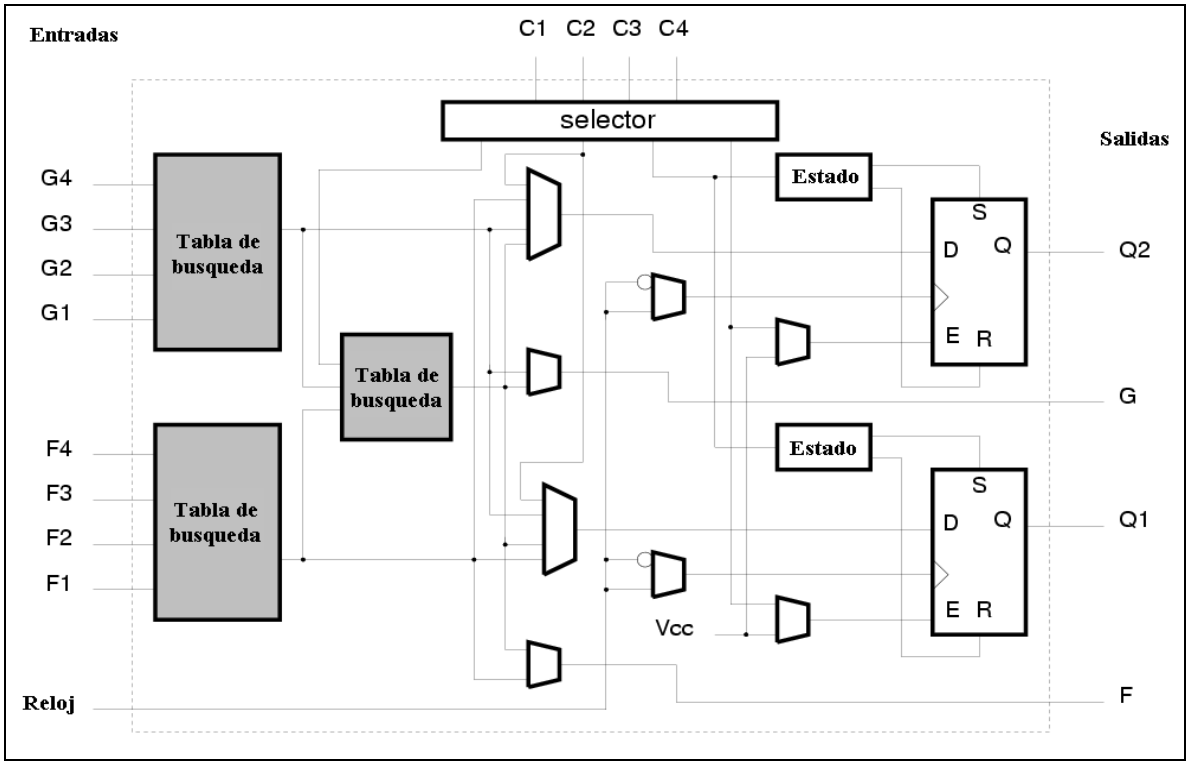

Figura A.3. CLB de la familia XC4000 de Xilinx.

[54] *El número de segmentos en cada canal depende del número de parte especificado.*

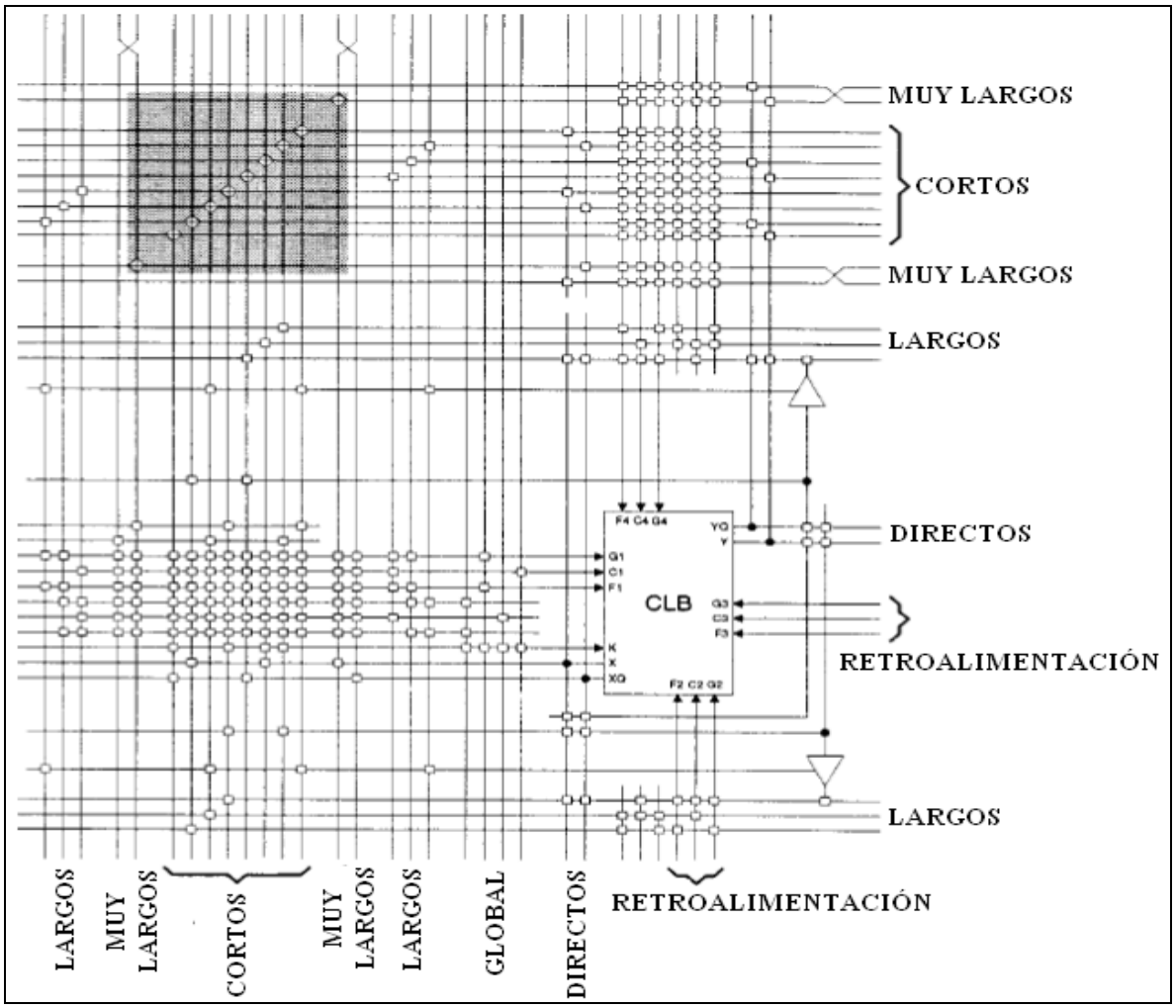

Figura A.4. Segmentos de cables de la familia XC4000 de Xilinx.

Además existen cables de conexión globales, de retroalimentación y directos, los cuales cumplen funciones diferentes en cada tipo de dispositivo. Los cables que rutean a cada CLB generalmente son:

- Interconexión local de 8 horizontales + 8 verticales.
- 8 cables muy largos: 4 horizontales + 4 verticales, que interconectan CLB's.
- 6 horizontales + 6 verticales largos conectan a los CLB's en un renglón o columna: estos son los buses y líneas de control.
- 4 globales verticales: controlan el reloj inclinado.
- 2 verticales de acarreo: comparadores/sumadores de amplia velocidad

Las entradas y salidas tienen las características mostradas en la figura A.5:

- Se localizan al final de cada renglón y columna.
- Contiene un registro/latch de entrada y un registro/latch de salida.
- Tiene dos entradas al chip: directo y registro.
- El pín de entrada/salida asociado puede ser utilizado como entrada, salida ó bidireccional.

Un punto importante en la interconexión de los dispositivos de Xilinx es que las señales deben pasar a través de switches para lograr conexiones entre CLB's y el número total de switches recorridos depende del grupo de segmentos de cable utilizados. Así, la velocidad de ejecución de un circuito implementado depende en parte de cómo se asignan los segmentos de cable a las señales, mediante herramientas CAD [12].

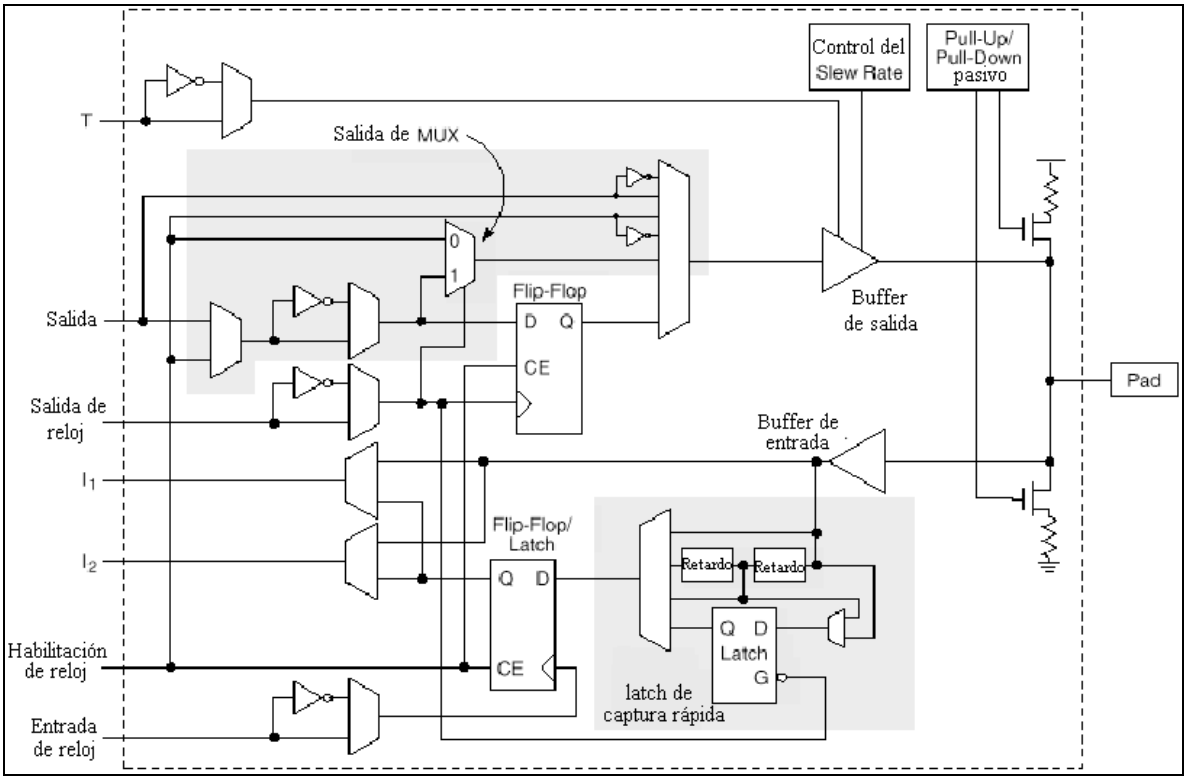

Figura A.5. Diagrama de un bloque de entrada/salida de la familia XC4000.

APÉNDICE B

Métodos de clasificación para los sujetadores de marco

B.1 Marcos contra campos.

Los digitalizadores de marco pueden adquirir video entrelazado digitalizando ambos campos[55]. Por otra parte los digitalizadores de campo son similares, excepto que ignoran uno de los campos de video, por lo cual producen imágenes con menos líneas horizontales, es decir, con una proporción de aspecto menor.

Así pues un digitalizador de marco que utiliza el sistema NTSC de 525 líneas, digitalizara imágenes hasta de 488 líneas horizontales, mientras que un digitalizador de campo solo alcanzara las 244 líneas horizontales máximo. Por otra parte en el sistema PAL y SECAM de 625 líneas un digitalizador de marco digitalizara hasta 588 líneas y un digitalizador de campo alcanzara las 294 líneas máximo.

[55] *El estándar NTSC de video entrelazado 2:1 contiene 2 campos, sin embargo existen variaciones de los estándares NTSC, PAL y SECAM que contienen 4, 8 ó hasta 16 campos, principalmente en sistemas de video profesional.*

B.2 Campos inmóviles contra video en tiempo real.

La digitalización en tiempo real permite al objeto moverse cuando está siendo capturado por una cámara de video, aunque de forma muy lenta, sin que la imagen digitalizada contenga defectos como distorsiones o desplazamientos, principalmente. Esto es posible si el marco de video se digitaliza en menos de 1/30 de segundo y durante este periodo el objeto no se mueve apreciablemente.

La digitalización con una fuente de video como la televisión o una videograbadora se puede llevar a cabo con un digitalizador de video de tiempo real ya que éste puede almacenar a la velocidad de refresco de la imagen de video. Los digitalizadores de marco inmóvil cuentan con el hecho de que la imagen de video está fija durante el proceso de digitalización.

B.3 Resolución.

La resolución define el número de píxeles que el digitalizador de video es capaz de producir por imagen. Los digitalizadores de video de tiempo real típicamente utilizan una matriz cuadrada de píxeles, por ejemplo de 256x256 o 512x512, la imagen resultante es de 256 o 512 píxeles horizontales por 256 ó 512 píxeles verticales respectivamente.

La cantidad de píxeles también puede estar determinado por la memoria[56] utilizada en el grabador, ya que los CI's de memoria tienen capacidades de almacenamiento en potencias de dos, sin embargo, hoy en día existen memorias con capacidades para almacenar millones de píxeles, lo que hace posible guardar no sólo una imagen con gran definición sino varias.

B.4 Ancho de bits.

Los digitalizadores de video así como los adaptadores de despliegue tienen un número fijo de bits usados para representar cada muestra o píxel de la imagen. Cada muestra representa la intensidad de la imagen en la posición donde fue digitalizado el píxel. Utilizar la mayor cantidad de bits en un muestreo hace más eficiente la representación digital de una imagen, sin embargo la cantidad de bits soportada por el digitalizador de video determina el precio del mismo, ya que el número de bits incrementa el ancho de la memoria requerida y del convertidor analógico a digital (ADC, *Analog to Digital Convert*). La cantidad de bits utilizada en el muestreo típicamente es de 8, 10, 12, 14 ó 16 bits, por lo cual se obtienen 256, 1024, 4096, 16384 ó 65536 diferentes valores de intensidad respectivamente.

Es obvio que si cada píxel puede representar cualquiera de los 256 valores, representara una imagen más limitada que al utilizar 65536 valores, con lo cual la representación de la imagen original será mas estrecha, esto es que se podrán apreciar detalles minúsculos y objetos más pequeños.

B.5 Color contra monocromático.

Los digitalizadores de video pueden ser monocromáticos o con capacidades para digitalizar imágenes a color. Si un digitalizador de video soporta color, generalmente tiene tres grupos de CI's ADC's y memorias, uno para cada componente de la imagen a color rojo, verde y azul, ya que es necesario tener tres grupos de bits para componer un píxel de la imagen.

[56] *Algunos grabadores de video utilizan memoria de la computadora central, por lo tanto no están restringidos en cuanto a la cantidad de memoria en la tarjeta.*

Por otra parte los digitalizadores de video monocromático tienen un ADC y posiblemente una memoria para almacenamiento de la imagen digitalizada, debido a lo anterior las imágenes resultantes se reproducen utilizando tonos de grises[57], de donde cada tono de gris representa un valor de intensidad en un píxel. La digitalización de una imagen puede ser en paralelo o en serie de acuerdo al ADC utilizado.

B.6 Procesamiento sobre tarjeta.

El procesamiento de imágenes es otra característica que puede distinguir un digitalizador de video, ya que si es capaz de ejecutar algoritmos sobre la tarjeta ofrecerá al usuario capacidad para ejecutar aplicaciones de mayor complejidad.

Con un procesador de imágenes sobre tarjeta, se pueden ejecutar varios algoritmos sobre el digitalizador de video sin saturar a la computadora central con tareas computacionalmente intensivas, ya que estos algoritmos de procesamiento de imágenes son implementados en hardware y no en software, por lo cual se puede ahorrar tiempo en el procesamiento de las imágenes ya que no es necesario tener una computadora conectada y tampoco es necesario realizar un protocolo de comunicaciones, lo cual consume recursos de la tarjeta.

B.7 Salida de video.

Muchos digitalizadores de video tienen una salida de video disponible para controlar un monitor de video directamente, está puede ser utilizada para ver los efectos de un algoritmo de procesamiento de imágenes con la finalidad de verificar la exactitud con la que el sistema opera.

La conexión puede realizarse mediante diferentes puertos, tales como IEEE 1394, USB, PS/2, serie o paralelo, de acuerdo a las capacidades del sistema y a los dispositivos utilizados sobre la tarjeta.

[57] *Estas imágenes se conocen como imágenes de tono continuo porque utilizan el rango completo de tonos de grises desde negro hasta blanco para despliegue.*

APÉNDICE C

Formato de imagen pgm

El formato PGM está formado por los siguientes campos:

- Un número característico para identificar el tipo de fichero. El número característico de un fichero PGM incluye dos caracteres "P5".
- Espacio blanco (blancos tabuladores, retornos de carro o saltos de línea).
- Una anchura, formateada como caracteres ASCII en decimal.
- Espacio blanco.
- Una altura, de nuevo en ASCII decimal.
- Espacio blanco.
- El máximo nivel de gris (Maxval), de nuevo en ASCII decimal. Debe ser menor de 65536.
- Un carácter de nueva línea u otro carácter blanco.
- Una serie de anchura * Altura niveles de gris, recorriendo la imagen de izquierda a derecha.
- Cada nivel de gris es un número entre 0 y Maxval, con 0 correspondiendo al negro y Maxval, al blanco.
- Cada nivel de gris se representa en binario mediante 1 ó 2 bytes.
- Si Maxval es menor de 256, se emplea 1 byte. En otro caso, se usan 2 bytes.
- El byte más significativo es el primero.
- Cada nivel de gris es un número proporcional a la intensidad del píxel.

- Los caracteres situados entre un "#" hasta el final de la línea, antes de la línea que contiene Maxval, son comentarios y se ignoran.
- En realidad hay otra versión del formato PGM que es bastante poco frecuente: el sencillo (plain) PGM. El formato anterior, generalmente considerado el normal, se conoce como el de datos en bruto (raw) PGM.
- La diferencia en el formato "plain" es:
- El "número mágico" es "P2", en lugar de "P5".
- Cada píxel se representa mediante un número decimal ASCII (de tamaño arbitrario).
- Cada píxel tiene espacio blanco delante y detrás de él. Debe haber al menos un carácter blanco entre dos píxeles, pero no hay un máximo.
- Ninguna línea debería tener más de 70 caracteres.

Referencias

[1] Charles Poynton. “A technical introduction to digital video”. Ed. John Wiley & Sons. 1996

[2] Charles Poynton. “Motion portrayal, eye tracking and emerging display technology” 30th SMPTE Advanced motion imaging conference, 192–202. SMPTE, 1996.

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

[4] José M. Valiente y Gabriela Andreu. “Adquisición de imágenes” Departamento de Informática, Sistemas y Computadores (DISCA). Grupo de Visión por Computador. UPV (Universidad Politécnica de Valencia)

[5] Craig A. Lindley “Practical image processing in C” Jonh Wiley & Sons, Inc USA.1991.

[6] Kelin J. Kuhn “Conventional Analog Television” - An Introduction EE 498

[7] Donald G. Fink. “The Forces at Work Behind the NTSC Standards SMPTE” JUNE 1981

[8] Director Emeritus of the IEEE and Chairman of the SMPTE Study Group on High-Definition Television. “122nd annual SMPTE Technical Conference” November 9-14, 1980, New York, N.Y.

[9] Carlos Javier Mendoza Solana. “Diseño conceptual de un sistema de adquisición de video de alta resolución en ambiente de computadora personal”. Tonantzintla, Puebla. 20 de Marzo del 2000.

[10] CONRAC Division, CONRAC Corp., “Raster Graphics Handbook”, CA 1980.

[11] John G. Proakis Dimitris G. Manolakis “Tratamiento digital de señales, principios algoritmos y aplicaciones”. Ed. Prentice Hall. España 1998.

[12] Stephen Brown and Jonathan Rose. “Architecture of FPGAs and CPLDs: A Tutorial” Department of Electrical and Computer Engineering University of Toronto.

[13] “A New Architecture For a Field Programmable Logic Device”, Tutorial.

[14] Read, J W. “Gate Arrays: Design and Applications”. 1985.

[15] Fernando Rincón y Lluis Terés. “Reconfigurable Hardware Systems” Instituto de Microelectrónica de Barcelona, Campus UAB. Bellatera, Barcelona, España.

[16] Alison Carter. "Using Dynamically Reconfigurable Hardware in Real-Time Communications. Systems" Real Time Systems Group Department of Computer Science University of York. Nov 2001.

[17] D. Gaasterland, "Computer Systems Architecture" University of Maryland August, 2001.

[18] "Diseño con PLD" Dpto. de Ingeniería Electrónica y Comunicaciones Universidad de Zaragoza.

[19] Enrique Mandado, L. Jacobo Álvarez y Maria Dolores Valdés, "Dispositivos Lógicos Programables y sus aplicaciones". Editorial Thomson Editores, Paraninfo, S.A. España 2002.

[20] Stephan M. Trimberger. "Field-Programmable Gate Array Technology". Kluwer Academic Publishers, 1994.

[21] S. Brown, R. J. Francis, J. Rose and Z. G. Vranesi, "Field-Programmable Gate Arrays", Kluwer Academic Publishers, 1992.

[22] S. Brown and J. Rose, "FPGA and CPLD Architectures: A Tutorial", Proceedings of the IEEE Design & Test of Computers, vol. 13, no. 2, p. 42-57, 1996.

[23] John F. Wakerly, "Diseño digital principios y prácticas". Editorial Prentice-Hall Hispanoamericana, S.A. Mexico 1992.

[24] Victor P. Nelson, H. Troy Nagle, Bill D. Carrol, J. David Irwin, "Analisis y Diseño de Circuitos Lógicos Digitales". Editorial Prentice Hall Inc, Mexico 1996.

[25] M. Morris Mano. "Lógica Digital y Diseño de Computadores". Editorial Prentice Hall. Hispanoamericana, S.A. Mexico 1982.

[26] M. Morris Mano. "Digital Design". Editorial Prentice Hall Hispanoamericana, S.A.

[27] Zoran Salcic and Asim Smailagic, "Digital Systems Design and Prototyping: Using Field Programmable Logic and Hardware Description Languages", second edition, Kluwer Academic Publishers, 2000.

[28] Moisés Pérez Gutiérrez. "Introducción a la Tecnología FPGA". Coordinación de Ciencias Computacionales, INAOE, Puebla México. 2002

[29] Barry Fagin, Cyril Renard. "Field Programmable Gate Arrays and Floating Point" Arithmetic. Dept. of Computer Science. U.S. Air Force Academy.

[30] Stephen M. Trimberger "Field-Programmable Gate Array Technology". Kluwer Academic Publishers, Boston, MA, 1994.

[31] Richard Larry Ukeiley. "Field Programmable Gate Arrays (FPGAs)", PTR Prentice Hall, Englewood Cliffs, NJ, 1993.

[32] Juan Carlos Herrera Lozada. "Implementación de un controlador difuso en un FPGA, utilizando lenguajes de descripción de hardware". Centro de Investigación en Computación, IPN México, DF. Marzo 2002

[33] Yann-Hang Lee. "FPGA:"Field Programmable Gate Array. Real Time Systems" Lab. Computer Science and Engineering ASU

[34] Craig D. Ulmer. “Configurable Computing: Practical Use of FPGA”. Georgia Institute of technology. Georgia. January 5, 1999.

[35] J. Faura, M. A. Aguirre, M. Moreno, P. Paddan, P. van Duong and J. Mª Insenser, FIPSOC. “A Field Programmable System On Chip, Proceedings of the Design of Circuits and Integrated Systems Conference (DCIS'97)”, Sevilla, 1997, 597-602.

[36] Zoran Salcic and Asim Smailagic, “Digital Systems Design and Prototyping: Using Field Programmable Logic and Hardware Description Languages”, second edition, Kluwer Academic Publishers, 2000.

[37] Maxfield, Clive "Max," "LPMs: High-level design uses low-level techniques," EDN, May 9, 1996.

[38] Barr, Michael. "Programmable Logic: What's it to Ya?" Embedded Systems Programming, June 1999, pp. 75-84

[39] K. C. Chang, “Digital Design and Modeling with VHDL and Synthesis”. IEEE Computer Society Press, USA,1997

[40] “Handel-C Language Reference Manual”. Version 2.1. Celoxica. 2001.

[41] Serafin Alfonso Pérez López, Enrique Soto Campos, Santiago Fernández Gómez. “Diseño de sistemas digitales con VHDL”. THOMSON Editores. España 2002.

[42] David Pellerin, Douglas Taylor. “VHDL Made easy”. Editorial Prentice Hall. USA 1997.

[43] Peter J. Ashenden “The VHDL Cookbook”, Dept. Computer Science University of Adelaide July, 1990 South Australia

[44] Adel S. Sedra, Kenneth C. Smith. “Circuitos Microelectrónicas”. Oxford University Press. Mexico 1999.

[45] Robert F. Coughlin, Frederick F. Driscoll. “Amplificadores Operacionales y Circuitos Integrados Lineales”. Editorial Prentice Hall. Mexico 1993.

[46] Donald L. Schilling, Charles Belove. “Circuitos electronicos, discretos e integrados” Alfaomega marcombo 2ª. Edición 1985.

[47] Peter Marwedel, “Embebbed System Design”, Boston : Kluwer Academic Publishers, 2003.

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

[49] Rafael C. González, Richard E. Woods. “Tratamiento digital de imágenes” Addison-Wesley Iberoamericana E.U.A 1992

[50] David Hiriart, Jorge Valdez, Fernando Quirós “SiMoN Sistema Monitor de Nubes diurno” Manual técnico, Instituto de Astronomía. Universidad Nacional Autónoma de México 2006

[51] Amilcar Y. Rodriguez Martines, “Vision and inspection systems for manufacturing cardiac pacing and defibrillation leads”, Tesis de Maestria¸ Universidad de Puerto Rico 2004

[52] Christopher R. Baxter, Mark A. Massie, "MIRIADS, Miniature Infrared Imaging Applications Development System Description and Operation" Nova Research, Inc., 2007

[53] J. Arias, "Circuitos Analógicos con inversores CMOS" UNAM, julio de 2008

[54] J.C. Moreno, M. Berenguel, F. Rodríguez, J.F. Sarabia, R. Garrote, J.L. Guzmán, O. López "Proyecto de aplicación de telerrobótica a un minirobot movil" Universidad de Almería.

[55] M.V. Aguannob, F. Lakestani, M.P. Whelan, M.J. Connelly "Speckle interferometry using a CMOS-DSP camera for static and dynamic deformation measurements", University of Limerick, Limerick, 2004

[56] Rafael Muñoz Salinas, "Soft computing and computer vision techniques applied to autonomous robot navigation and human robot interaction", tesís doctoral, Universidad de Granada.

[57] Juan Carrizo, Germán García, Humberto Secchi, Vicente Mut, "Sistema de vision 2D1/2. Una aplicación de robótica móvil" Universidad Nacional de San Juan Argentina.

[58] Luis Miguel Bergasa Pascual, "Seguimiento facial, mediante visión artificial, orientado a la ayuda de la movilidad" tesis doctoral, Universidad de Alcala. 1999.

[59] M. Sánchez Raya, R. Rodríguez Macías y J.M. Andújar Márquez, "Sistema portable de adquisición de imágenes de diagnostico medico" Escuela Politécnica Superior. Universidad de Huelva

[60] Madaín Pérez, François Cabestaing y Jack-Gérard Postaire, "STREAM, Un procesador basado en FPGA para el tratamiento de secuencias de imágenes: Aplicación a la estéreo visión densa" Laboratoire d' Automatique I3D, Université des Sciences et Technologies de Lille.

Referencias URL

[URL1] http:// ntsc-tv.com/ “Sistemas de televisión”.[Julio 2008]
[URL2] http://www.ee.surrey.ac.uk/ [Julio 2008]
[URL3] http://www.vac-brick.com/ “Frame grabbers” [Julio 2008]
[URL4] www.optimagic.com “Sistemas ópticos” [Agosto 2008]
[URL5] www.wikipedia.org “Enciclopedia libre” [Agosto 2008]
[URL6] www.xess.com “Fabricante de FPGA’s” [Agosto 2008]