Capítulo 5

Análisis de la Señal

Lenguajes de Descripción de Hardware. [9] [10]

Los lenguajes de descripción de hardware HDL por sus siglas en inglés (*Hardware Description Languages*) son utilizados a partir de los años 70 en los ciclos de diseño de sistemas digitales asistidos por herramientas de CAD (Computer-Aided Design) electrónico. En los años 80 aparecen los lenguajes Verilog y VHDL que, aprovechando la carrera tecnológica de la época y los adelantos en las herramientas de fabricación de circuitos integrados, logran imponerse como lenguajes imprescindibles en el desarrollo de nuevos sistemas.

Los estudios para la creación del lenguaje conocido como VHDL (VHSIC HDL) comenzaron en el año de 1981, bajo la cobertura de un programa para el desarrollo de circuitos Integrados de alta velocidad (VHSIC), por el Departamento de Defensa de los Estados Unidos. Dos años después las compañías Intermetrics, IBM y Texas Instruments obtuvieron la concesión de un proyecto para la realización del lenguaje y de un conjunto de herramientas que fueran de apoyo para su implementación, y años después se convierte en un estándar IEEE-1076 (1987), en sus inicios este lenguaje era utilizado simplemente para la simulación y recreación de circuitos, para eventos en tiempos definidos por el usuario de forma virtual y fue en 1993 que sufrió leves modificaciones.

5.1.1 Características del lenguaje. [10]

El lenguaje VHDL fue creado con el propósito de documentar sistemas y circuitos digitales con ayuda de un lenguaje formal.

El lenguaje VHDL es un lenguaje normalizado el cual describe los circuitos electrónicos en un formato adecuado para ser interpretado tanto por máquinas como personas. Es un lenguaje ejecutable, lo cual permite que la descripción textual del hardware se materialice en una representación del mismo utilizable por herramientas auxiliares tales como simuladores y sintetizadores lógicos, compiladores, simuladores de tiempo, herramientas de diseño físico, entre otros.

El lenguaje VHDL da posibilidades de adaptarse de manera precisa a las distintas fases que se dan en el desarrollo de los diseños, o bien se podría decir que el VHDL tiene un amplio rango de capacidad descriptiva. Esto quiere decir que dicho lenguaje es adaptable a distintas metodologías de diseño y es independiente de la tecnología, lo que permite que distintas instituciones, compañías y organizaciones se puedan relacionar entre si con la electrónica digital, y además facilita la actualización y adaptación de los diseños a los avances de la tecnología en cada momento. También cabe señalar su destacada capacidad del lenguaje para el manejo de proyectos de grandes dimensiones, y para la sustitución de componentes o modificación de circuitos.

5.1.2 Análisis de Imagen. [11]

Una vez que se ha entendido el concepto del lenguaje VHDL y sus características, a continuación se describirá cómo fue creado y diseñado el análisis de la imagen adquirida por la cámara CCD en el lenguaje VHDL.

A partir de las señales que se obtuvieron en la interfaz del capítulo anterior (Dato, Línea y Cuadro), y una vez que se analizaron sus características se definió una serie de módulos dentro de la arquitectura del programa.

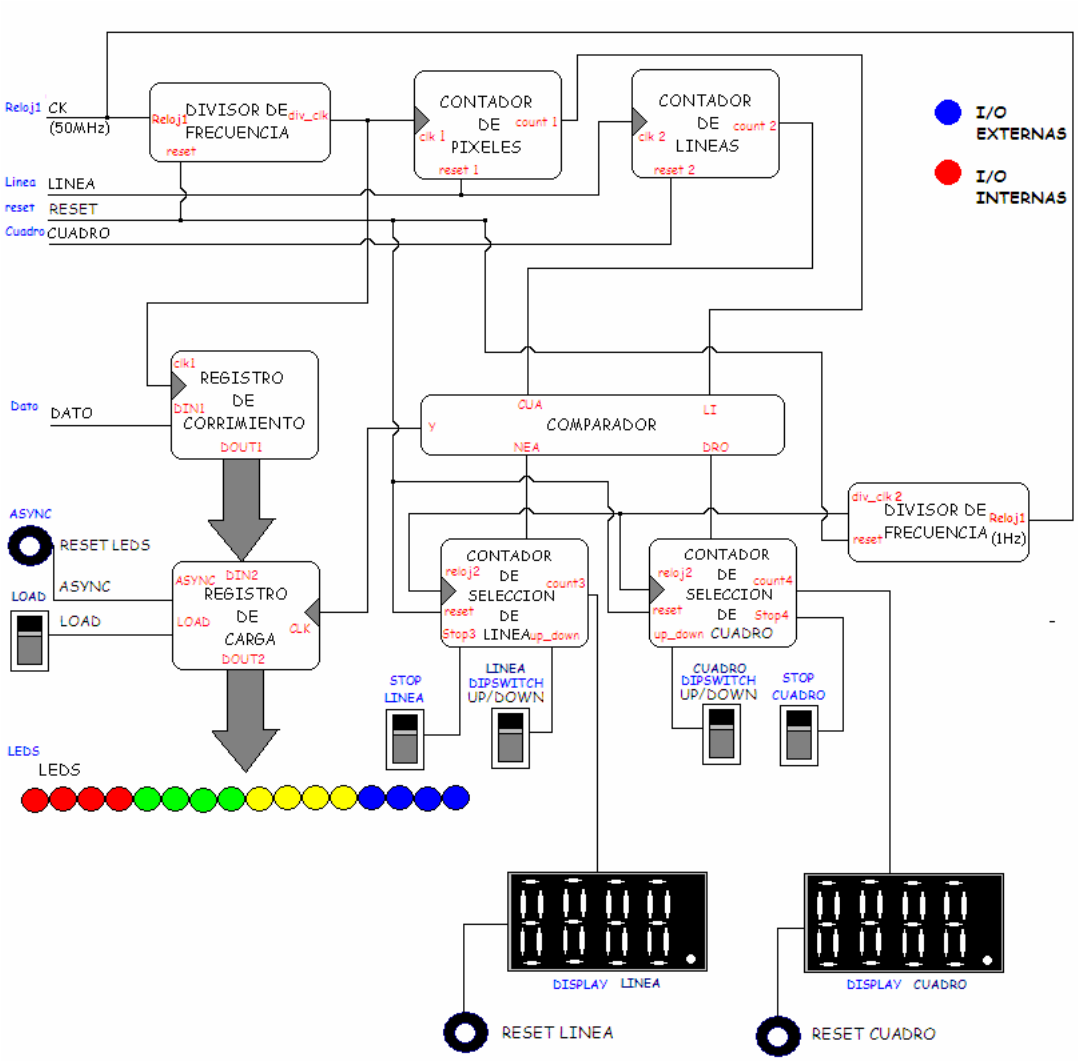

Figura 5.1 Esquema general del programa "Adquisición de Imagen".

En la figura 5.1 claramente se puede observar que hay 9 módulos principales, los cuales serán explicados uno a uno detalladamente y cual es su funcionamiento para el análisis de la imagen.

1. Divisores de Frecuencia o Preescaladores
2. Contadores Internos
    - Contador de Líneas
    - Contador de pixeles
3. Contadores Externos
    - Selección de Línea
    - Selección de Cuadro
4. Registro de corrimiento
5. Comparador
6. Registro de Carga

Se recomienda revisar los módulos teniendo a la mano el esquema general mostrado en la Figura 5.1, para ir entendiendo su desarrollo.

Divisores de Frecuencia.

Estos dispositivos fueron creados con el propósito de preescalar una frecuencia dada básicamente por la tarjeta de evaluación Digilab2 la cual será explicada con detalle más adelante, el reloj interno que ofrece dicha tarjeta es de un valor predeterminado con 50MHz de frecuencia, a partir de aquí se definió que dicha frecuencia dividida entre la frecuencia de la señal de Línea, y ésta multiplicada a su vez por el numero de muestras que se deseaba analizar y visualizar da por resultado un cierto numero de elementos.

$$Elementos = \frac{\left(\dfrac{50MHz}{(FrecuanciadeLínea)(NúmerodeMuestras)}\right)}{2(Nivel0\,y\,Nivel1)}$$

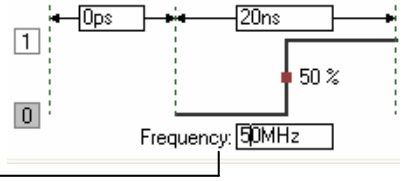

Figura 5.2 Ecuación de cálculo de elementos para divisor de frecuencia.

En la figura 5.2 se puede observar el cálculo del número de elementos para introducir en el programa que gobierna al módulo del divisor de frecuencia. La división entre dos que se muestra en la figura se da por que la frecuencia calculada tiene dos niveles lógicos por periodo. En el programa fuente de cada divisor se explica más detalladamente cual es el funcionamiento interno, éste se encuentra en el apéndice A.

Contadores Internos.

El contador de detección interna de pixeles usa como reloj la señal de salida generada por el divisor de frecuencia, su reinicio o reset para el conteo que mantiene en proceso viene de la señal de Línea (Interfaz), y su salida o bien cuenta va directamente a un comparador, el cual será explicado más adelante.

El contador de detección interna de Líneas usa como reloj la señal de Línea, y el reset o reinicio del contador viene de la señal de Cuadro, obviamente ambas señales que gobiernan a este modulo son entregadas por la interfaz explicada en el capítulo anterior, y su salida también va conectada a un comparador.

Los contadores de detección interna de pixeles y Líneas son dos módulos que funcionan utilizando números binarios y tienen como tarea mantenerse contando de acuerdo a las condiciones de las señales que se les asignaron, las salidas de ambos van conectadas a un comparador y no se debe olvidar que cada modulo contiene su señal de inicialización para evitar que su funcionamiento permanezca indeterminado.

Contadores Externos.

El segundo divisor de frecuencia que se muestra del lado derecho en la Figura 4.1, fue programado de la misma forma que el primero, la única variación es que se necesitó únicamente convertir o preescalar de 50 MHz a 1Hz, lo cual da un total de 25,000,000 de elementos, esto como resultado de la aplicación de la operación mostrada en la figura 5.2, se puede apreciar a detalle su funcionamiento interno en el programa fuente ubicado en el apéndice A.

Los contadores externos de Línea y Cuadro son parecidos a los internos, ya que éstos también cuentan en forma binaria e incrementan su cuenta de acuerdo a la señal de reloj, pero como lo indica su nombre, los contadores externos son gobernados por señales que vienen de un usuario o persona fuera del diseño digital, y debido a esto es que se le conecta como señal de reloj el segundo divisor de frecuencia (1Hz) para hacer visible al ojo humano el conteo que se genera en los módulos, además de que se puede incrementar o decrementar su cuenta, inclusive detener a voluntad dicha cuenta en cualquier momento.

La razón por la que se mencionó la construcción del segundo divisor de frecuencia fue para que se entendiera en cuales dispositivos se implementaron y la facilitación de manejo para el usuario, ya que estos dispositivos son creados con el propósito de ir indicando (a voluntad del usuario) en forma digital cual es la ubicación de la Línea y el Píxel de la imagen tomada que se desea analizar.

La salida de los dos contadores externos van conectadas a un comparador, y su finalidad se explicará más adelante, de igual manera van conectadas a displays para visualización del usuario plenamente identificables en el diagrama general de la figura 5.1

Registro de corrimiento.

Como lo indica su nombre el registro de corrimiento va desplazando dato por dato de acuerdo a la señal reloj que tiene, dicha señal es la misma que utiliza el contador interno de pixeles, la cual viene del primer divisor de frecuencia. Su entrada está conectada a la señal de Dato (Interfaz) ya que la entrada del registro es de un bit y su salida es de 16 bits, se debe recordar que la señal Dato es el resultado de la comparación de voltajes de la señal de video para detectar el color blanco (1 lógico) y el negro (0 lógico).

Comparador.

Este dispositivo reúne cuatro entradas, lo cual incluye tanto las salidas de los contadores internos como la de los contadores externos y dentro de este dispositivo se generan las condiciones que aparecen en la figura 5.3

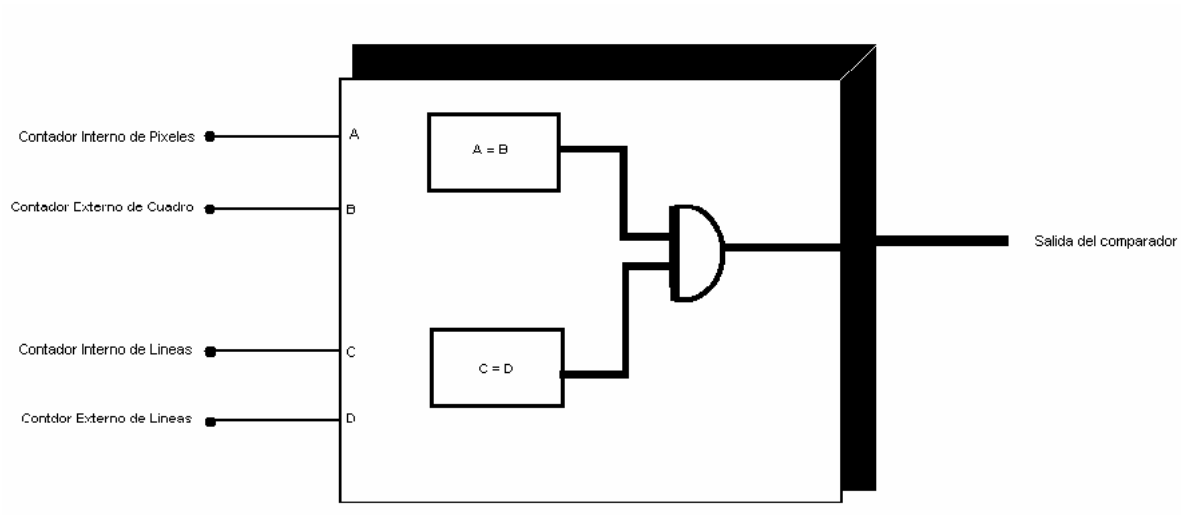

Figura 5.3 Diagrama interno de comparador.

Una vez que se cumplen las condiciones internas en el módulo de comparación, este envía a la salida un 1 lógico o un 0 lógico dependiendo del resultado del proceso interno del dispositivo.

Registro de carga.

Este módulo es un simple registro que permite que los datos almacenados que se encuentran en el registro de corrimiento pasen finalmente, gobernado por la señal que viene de la salida del modulo de comparación, dicha salida actúa como señal de reloj para que pasen los datos que se fueron acumulando durante los procesos anteriores, y posteriormente sean visualizados en 16 LEDS para poder identificar píxeles Negros (0) o Blancos (1) por cada led.

La traducción de este programa principal al lenguaje VHDL es la base del análisis de la imagen, en el capitulo 6 (pruebas y resultados) el entendimiento y funcionamiento de este diseño es fundamental para desarrollar los algoritmos de apoyo.

5.2 FPGA. [11] [12]

El arreglo de compuertas de campo programables FPGA por sus siglas en inglés (Field Programmable Gate Array), es un arreglo de bloques lógicos programables colocados en una infraestructura de interconexiones programable; es posible programar la funcionalidad de los bloques lógicos, las interconexiones entre bloques y las conexiones entre entradas y salidas. Un FPGA es programable a nivel hardware, además proporciona las ventajas de un procesador de propósito general y un circuito especializado que puede reconfigurarse las veces que sea necesario para depurar su funcionalidad.

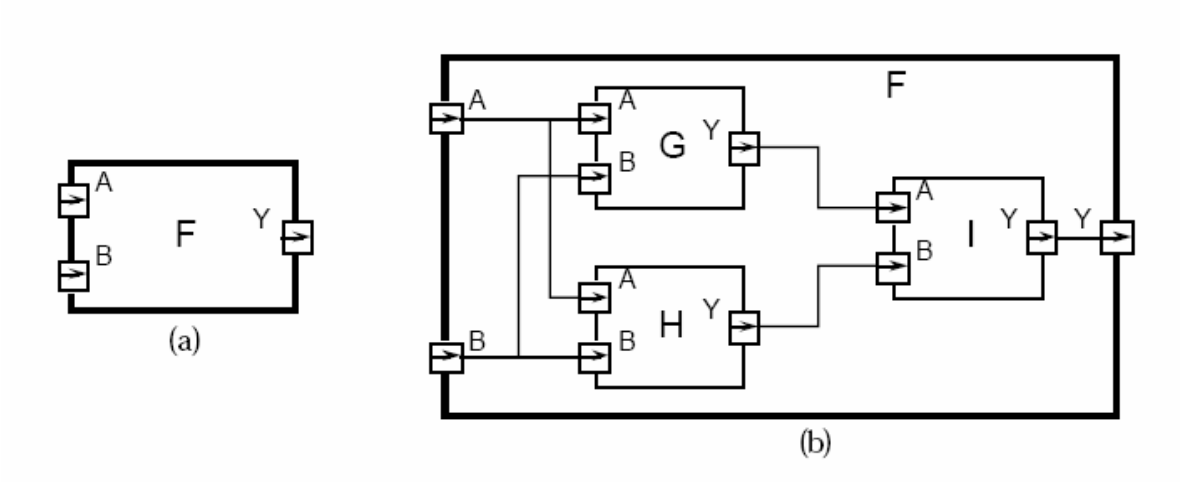

Figura 5.4 Ejemplo de Descripción Estructural (Interconexión de Bloques Lógicos).

Las aplicaciones de los FPGA´s van más allá de la simple implementación de lógica digital.

Los FPGA´s pueden ser utilizados para la implementación de arquitecturas específicas o para acelerar algún algoritmo. Los sistemas basados en FPGA´s proporcionan un mejor desempeño que sus correspondientes implementaciones software. Una arquitectura específica para un algoritmo puede tener un rendimiento de 10 a 1000 veces superior al de una implementación en un DSP. Las aplicaciones que requieren de un gran número de operaciones simples son adecuadas para su implementación en FPGA puesto que un elemento de reconocimiento puede diseñarse para efectuar esta operación y varias instancias de éste pueden reproducirse para llevar a cabo procesamiento en paralelo. Los algoritmos de bajo nivel en visión por computadora y análisis de imágenes son candidatos ideales para explorar arquitecturas específicas que aceleren su rendimiento.

5.2.1 Programación del FPGA.

La tarjeta que se utilizó en este proyecto contiene un FPGA del tipo XILINX SPARTAN, esta tarjeta es parte de un paquete o Kit de evaluación que el laboratorio de la Universidad de las Américas Puebla proporcionó a los estudiantes y profesores de electrónica y mecatrónica, la información de la tarjeta Digilab2 y de sus dos tarjetas de expansión DIO1 así como sus referencias técnicas son bastante claras y pueden ser consultadas con toda facilidad en el apéndice B.

El software XILINX ISE6 y los datos de apoyo que se consultaron para la programación de la tarjeta de evaluación fueron proporcionados por la misma compañía en su página Web, esto incluye actualizaciones y tutoriales.

Figura 5.5. Tarjeta Digilab2.

Una vez que está listo el programa principal de adquisición y análisis de imagen en lenguaje VHDL entonces se procede a implementarlo físicamente.

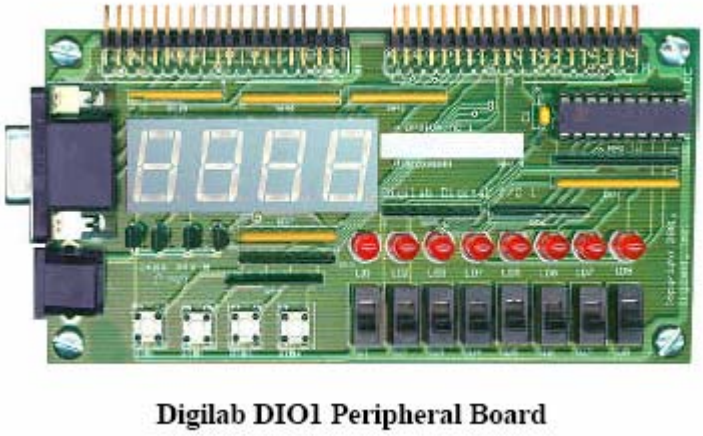

Figura 5.6 Tarjeta de expansión DIO1.

El primer paso para programar la tarjeta de evaluación fue introducir el programa de adquisición de imagen en el software de XILINX ISE versión 6.1.03i e indicar cual es era el tipo de FPGA al que se le iba a cargar el programa, que en este caso el FPGA tiene numero XC2S200 – PQ208.

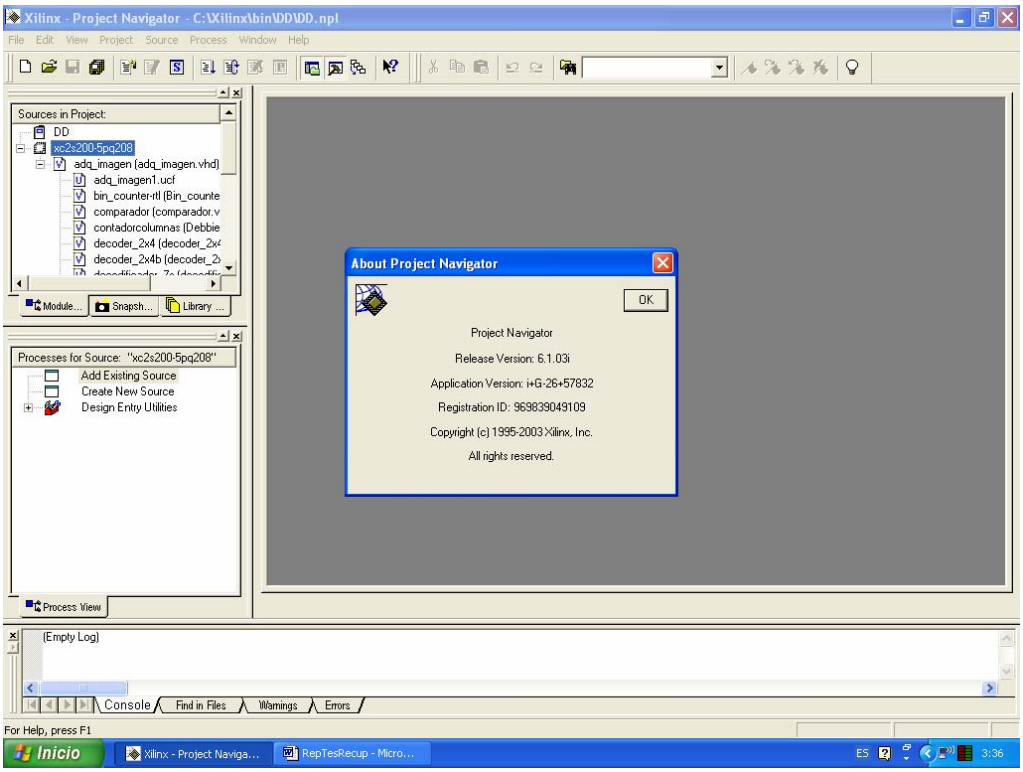

Figura 5.7 Inicio con Project Navigator.

Una vez que ya fueron aceptados y reconocidos los códigos como módulos VHDL dentro de la ventana de "Fuentes del Proyecto", se procedió a sintetizar dichas fuentes, con una opción que se encuentra en la ventana de "Procesamiento de Fuentes", el código fue compilado sin ningún problema, es decir, sin errores de sintaxis o bien errores de asignación de señales, acto seguido se creó una fuente o tabla que contenía la asignación de los pines para las entradas y las salidas entre el FPGA y las tarjetas de expansión DIO1.

Al crear una nueva fuente que contenía datos que afectaban el hardware de la tarjeta de evaluación y la tarjeta de expansión, se consultaron los manuales de referencia y de acuerdo a las tablas de asignación se introdujeron los datos en este módulo.

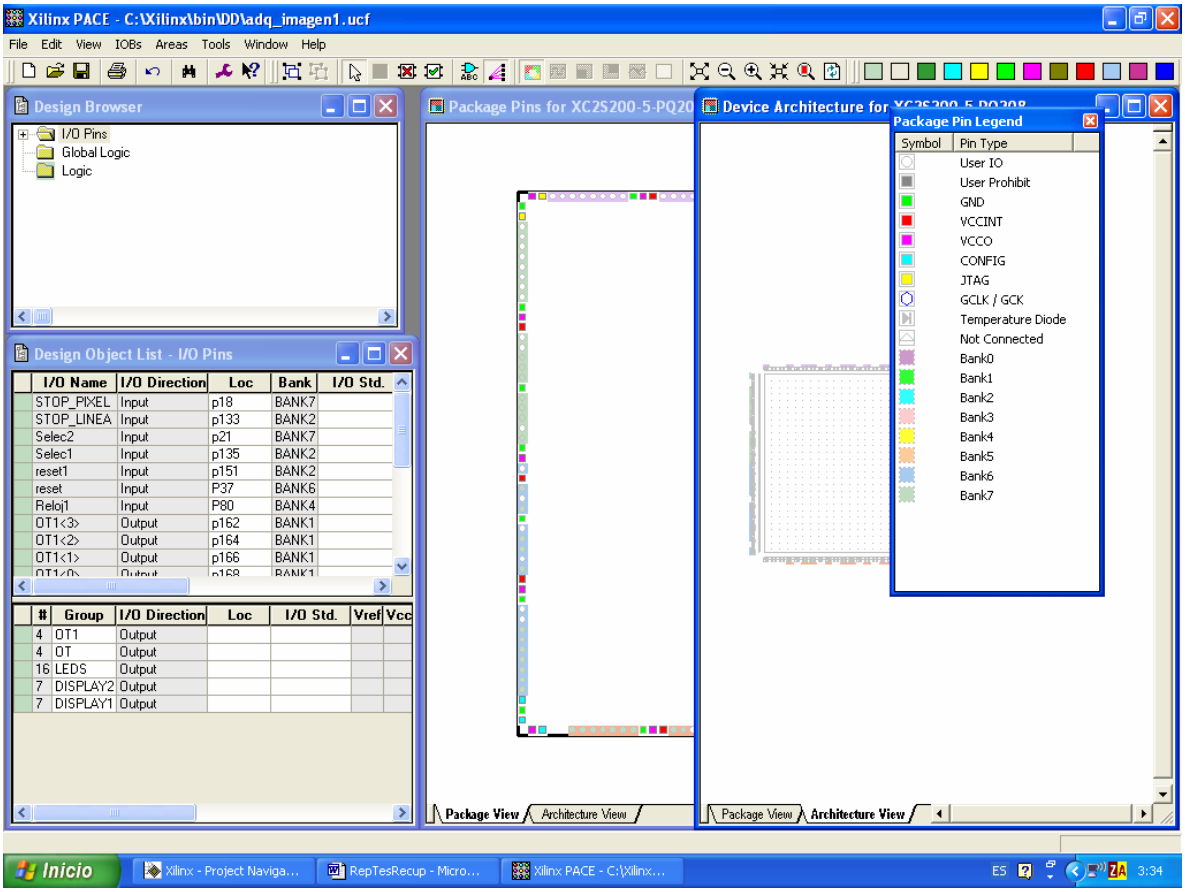

Figura 5.8 Asignación de Pines.

Cabe mencionar que para este paso se crearon tres buffers, ya que dentro del programa en VHDL había señales que controlaban a más de un componente. Esto va de acuerdo a las especificaciones técnicas de la misma tarjeta, ya que la solicitud de creación de dichos buffers al parecer esta predeterminada en el núcleo del FPGA tal vez con el propósito de evitar problemas de interferencia.

La siguiente tabla muestra un resumen de como se hizo la búsqueda adecuada de cada uno de los pines, ya que de acuerdo a los manuales se requieren tres tablas para identificar por completo cual es el conector de cada uno de los componentes del Kit de evaluación.

Adq_imagen	Conector J2(B)		Conector A & E	
Señal de Entrada	Conector	Pin en DIO1	Espejo	Pin en D2
Reloj	clk	predeterminado	predeterminado	**80**
Línea	NC	29	11	**60**
Cuadro	NC	31	9	**62**
Dato	NC	33	7	**67**
Selec1	SW3	6	36	**135**
Selec10	SW4	8	34	**138**
Selec2	SW3	6	36	**21**
Selec20	SW4	8	34	**23**
Reset	BTN1	18	24	**37**
Reset1	BTN1	18	24	**151**
ASYNC	BTN4	19	21	**43**
DIP_PIXEL	SW1	2	40	**16**
DIP_LINEA	SW1	2	40	**129**
DIP_RESET	SW8	16	26	**35**
STOP_LINEA	SW2	4	38	**133**
STOP_PIXEL	SW2	4	38	**18**
LEDS<0>	LD8	36	6	**179**
LEDS<1>	LD7	34	8	**176**
LEDS<2>	LD6	32	10	**174**
LEDS<3>	LD5	30	12	**172**
LEDS<4>	LD4	28	14	**167**
LEDS<5>	LD3	26	16	**165**
LEDS<6>	LD2	24	18	**163**
LEDS<7>	LD1	22	20	**161**
LEDS<8>	LD0	36	6	**68**
LEDS<9>	LD1	34	8	**63**
LEDS<10>	LD2	32	10	**61**
LEDS<11>	LD3	30	12	**59**
LEDS<12>	LD4	28	14	**57**
LEDS<13>	LD5	26	16	**48**
LEDS<14>	LD6	24	18	**46**
LEDS<15>	LD7	22	20	**44**
DISPLAY1<0>	CG	13	27	**148**
DISPLAY1<1>	CF	11	29	**146**
DISPLAY1<2>	CE	9	31	**141**
DISPLAY1<3>	CD	7	33	**139**
DISPLAY1<4>	CC	5	35	**136**
DISPLAY1<5>	CB	3	37	**134**
DISPLAY1<6>	CA	1	39	**132**
DISPLAY2<0>	CG	13	27	**34**

DISPLAY2<1>	CF	11	29	**31**
DISPLAY2<2>	CE	9	31	**29**
DISPLAY2<3>	CD	7	33	**24**
DISPLAY2<4>	CC	5	35	**22**
DISPLAY2<5>	CB	3	37	**20**
DISPLAY2<6>	CA	1	39	**17**
OT<0>	A4	27	13	**58**
OT<1>	A3	25	15	**49**
OT<2>	A2	23	17	**47**
OT<3>	A1	21	19	**45**
OT1<0>	A4	27	13	**168**
OT1<1>	A3	25	15	**166**
OT1<2>	A2	23	17	**164**
OT1<3>	A1	21	19	**162**

Una vez que se terminó la implementación del diseño y fueron aceptadas las asignaciones de los pines, se ejecutó por último el generador del programa de extensión ***.bit*** el cual fue cargado a la tarjeta D2 a través del puerto paralelo de la computadora.

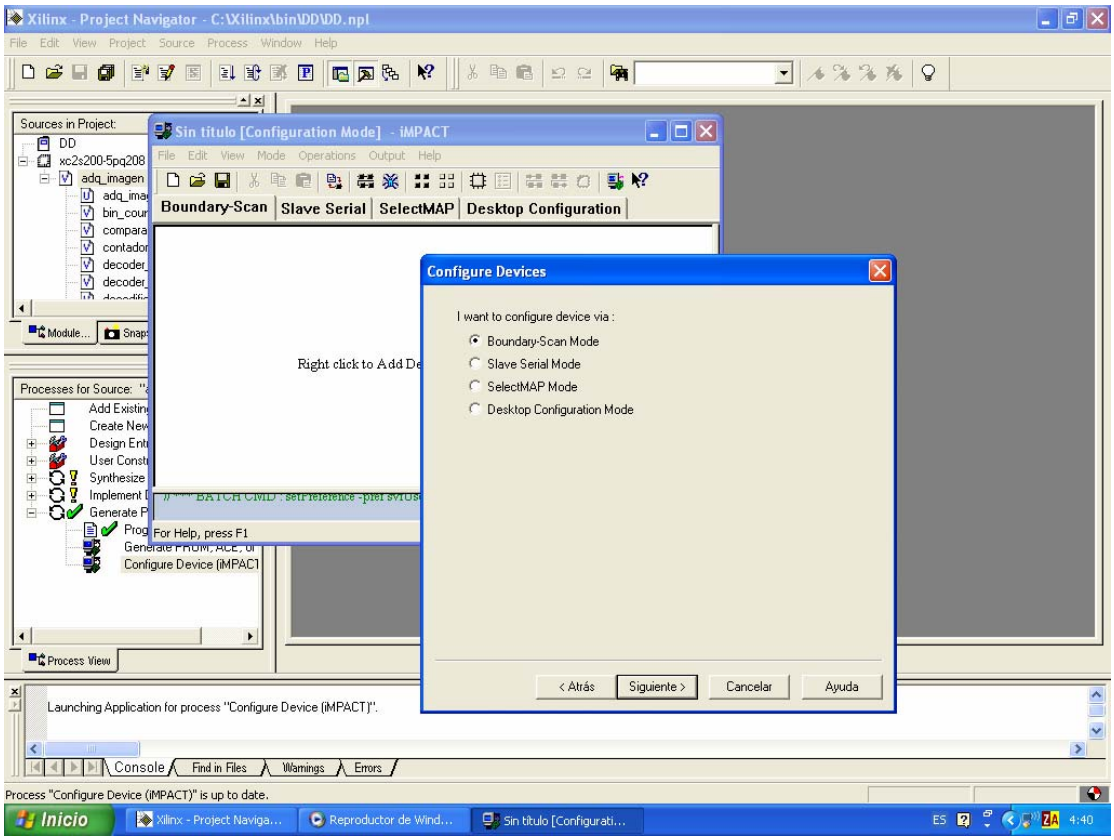

Figura 5.9 Programa Impact para cargar programa.

Durante el proceso en que se llevó a cabo la programación del FPGA se debieron hacer algunas observaciones:

- La tarjeta sólo cuenta con un número limitado de entradas externas, ya que para introducir las señales de Línea, Cuadro y Dato (indispensable para la ejecución adecuada de la adquisición y análisis de imagen) que vienen de la interfaz se tuvieron que utilizar pines "no autorizados".

- Los tres buffers que se declararon formaron cajas negras, ya que no fueron definidos como entidades sino como componentes y siempre aparecen dichas cajas como alerta en la sintetización del programa principal.

- Hay alertas que en realidad resultaron ser errores graves que no permitieron el correcto funcionamiento de la tarjeta al ejecutar las operaciones requeridas.

- El despliegue de los displays, tuvo que ser un diseño que solo convierte de binario a hexadecimal, y su funcionamiento interno se encuentra en la arquitectura del programa principal de adquisición de imagen.

- De acuerdo al manual de referencia los componentes utilizados en la interfaz para introducir las señales utilizadas por el FPGA para la ejecución del programa, fueron exitosamente compatibles, ya que el FPGA tiene voltajes de alimentación desde 0 a 3.3 V. máximo y las señales de los componente externos utilizan desde 0 a 5 V.

5.2.2 Análisis del Camino Trazado para el Robot Rastrero.

Una vez que se tiene cargado el programa de adquisición de imagen en el FPGA, se comenzaron a utilizar los contadores externos que se encuentran en la tarjeta de expansión DIO1, los cuales se encuentran conectados al comparador interno, para de esta manera elegir la columna y la fila que conforman a la imagen (con una resolución de 16 pixeles), como puede apreciarse en la figura 5.10, en dicha figura se pude ver claramente una matriz virtual generada por el diseño digital implementado.

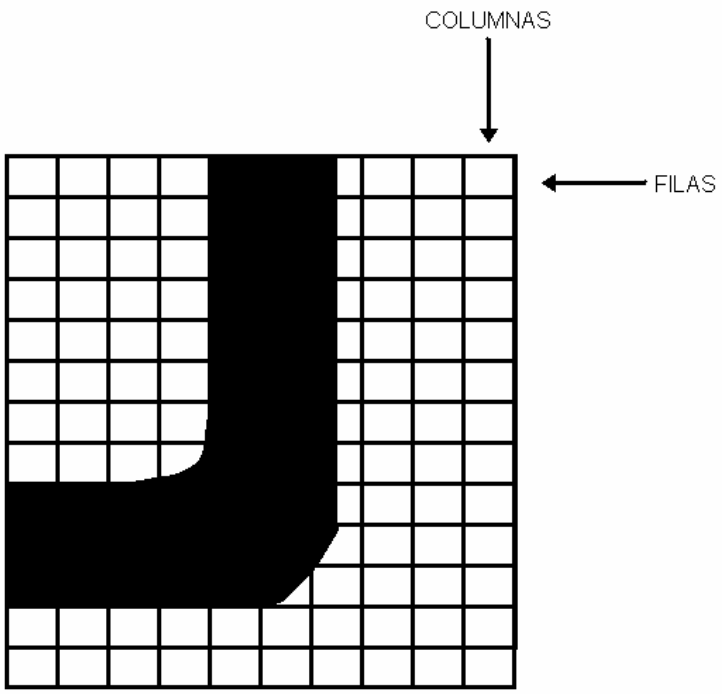

Figura 5.10 Matriz virtual.

Se debe aclarar que los contadores muestran en los displays el conteo en sistema hexadecimal, por lo tanto en el capítulo 6 de pruebas y resultados algunos de los datos tomados fueron expresados en dicho sistema.

Después de ubicar los píxeles blancos y negros fila por fila, columna por columna haciendo uso de los 16 bits (figura 5.11) proporcionados por la tarjeta de expansión (DIO1), se pudo definir la ubicación y cambios del camino trazado, de igual forma se pudo detectar en que sectores de la matriz virtual había cambios parciales (curvas) y completos (vueltas) en la trayectoria de la línea trazada.

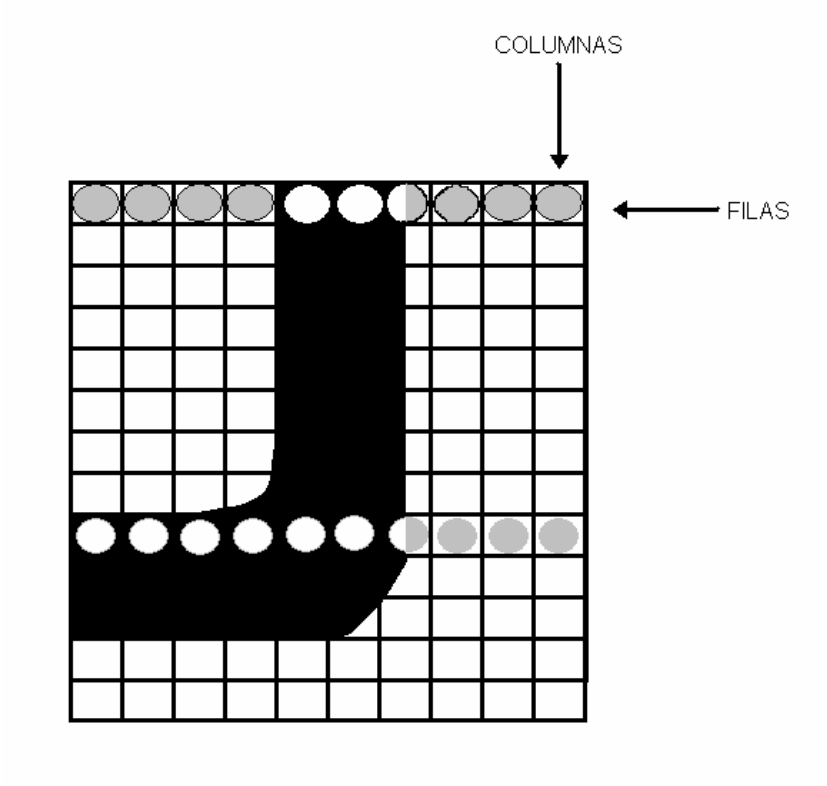

Figura 5.11 Matriz virtual detectada con los leds.

Ya que se comprobó que el sistema de análisis de imágenes funcionó adecuadamente en el FPGA, se creó un programa que se enfocó en contar el número píxeles (columnas) de color blanco y de color negro, por medio de los dos contadores gobernados por señales que involucran solamente al contador interno de píxeles, a la salida de la

comparación de contador interno de línea con el externo de línea, y las señales de Dato y Línea (siendo estas últimas dos las que vienen de la interfaz).

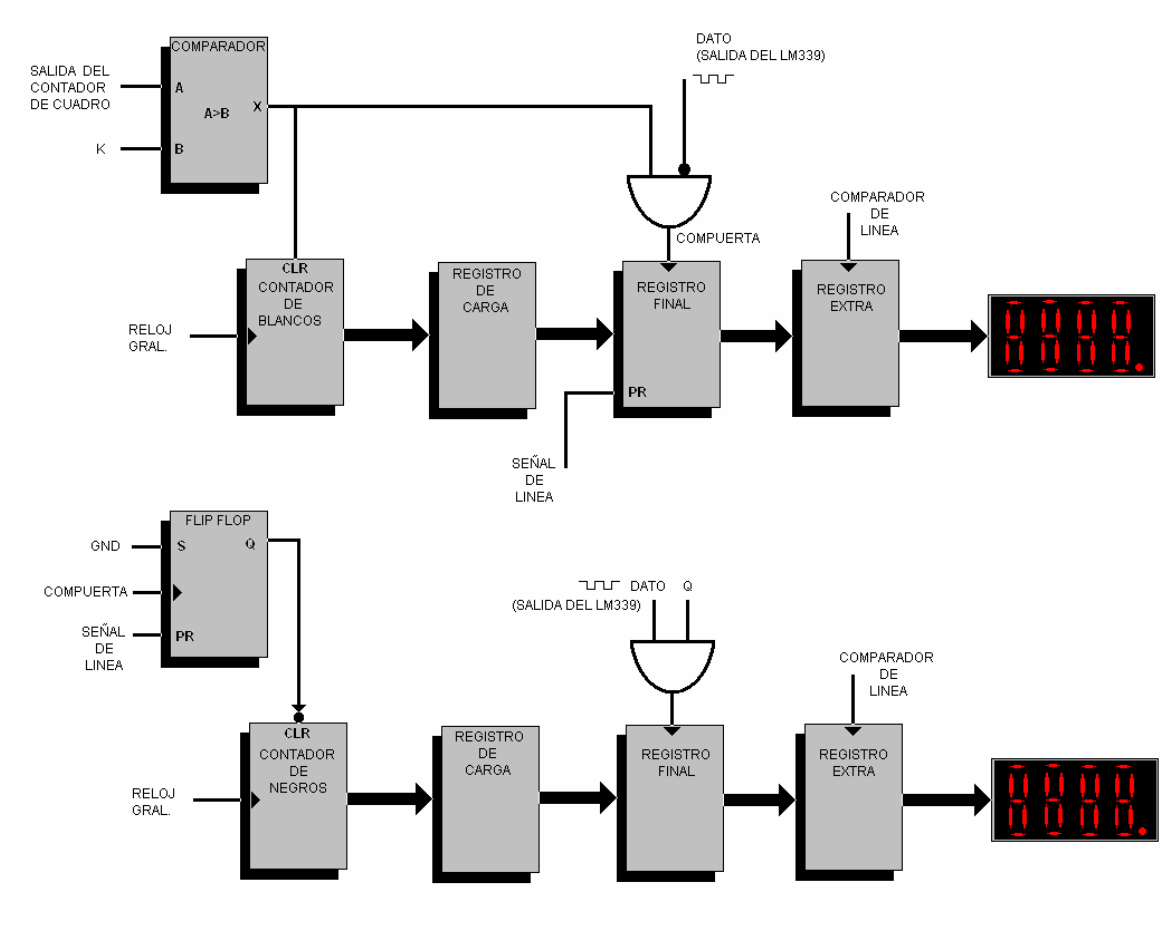

Figura 5.12 Contadores de píxeles negros y píxeles blancos.

En la figura 5.12 se puede apreciar con detalle cuales son los módulos y procesos necesarios para atrapar los datos para obtener el número de columnas con píxeles blancos y los píxeles negros, obviamente el resultado de tal operación se despliega en los displays pero puede ser difícil observarlos debido a que el análisis es bastante rápido y los cambios en la información desplegada ocurre en fracciones de segundo.

Con la información obtenida con el conteo de columnas de los dos tipos de pixeles que se están analizando se pueden realizar dos tipos de operaciones, una de las cuales puede indicar cual es el centro de la línea, con lo que se podría generar un control para la dirección del robot.

Después de haber encontrado

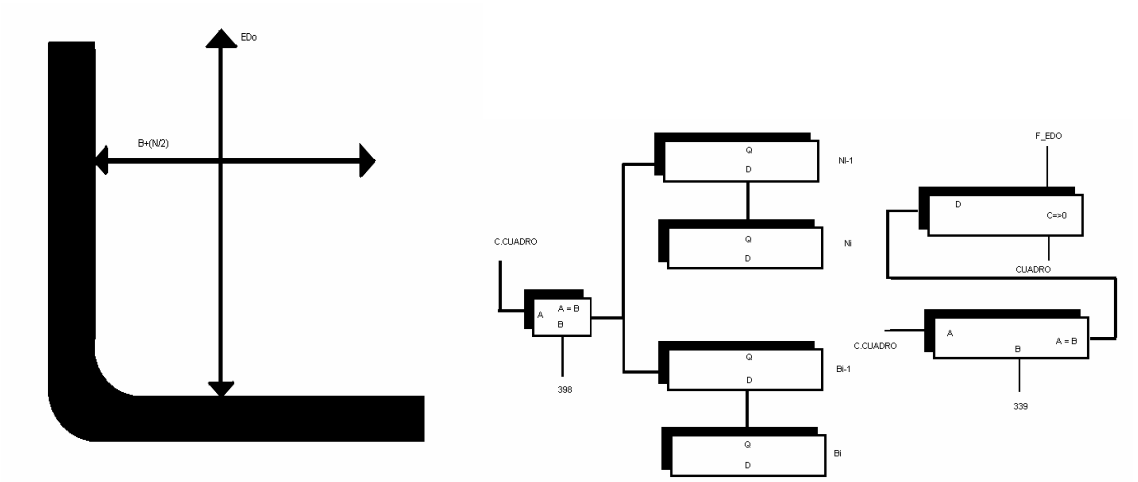

Figura 5.13 Detección de coordenadas.

Para la segunda operación simplemente se administraron otra serie de registros para manipular los datos que nos entregaron los contadores de columnas, y con esto se obtuvo una nueva secuencia de valores que indicaron las coordenadas (figura 6.13) para averiguar cual es la forma principal del camino trazado, es decir, con esta información desplegada se puede auxiliar al control de los motores para saber si se encuentra en un camino recto (acelerar) o se aproxima una curva (disminuir velocidad), teniendo las coordenadas adecuadas y estabilizadas se pueden generar los seudo códigos o algoritmos necesarios para controlar la velocidad.