PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

DISEÑO DE UNA ARQUITECTURA DE UN FILTRO DIGITAL DE SOBRE MUESTREO DE IMÁGENES, EN FACTOR 2, DE ACUERDO AL FORMATO H.264/SVC SOBRE FPGA

Tesis para optar el Título de **Ingeniero Electrónico,** que presenta el bachiller:

Christian Enrique Cano Salazar

ASESOR: MSc. Ing. Mario Andrés Raffo Jara

Lima, FEBRERO del 2012

RESUMEN

El presente trabajo consiste en la realización del diseño de la arquitectura en hardware de un filtro digital tipo FIR (*Respuesta al impulso finito*) para sobre muestreo de imágenes de Televisión Digital, de acuerdo al estándar japonés-brasileño H.264/SVC de codificación de video escalable, con una tasa de cuadros mayor o igual a 30 cuadros por segundo (fps) para poder operar en tiempo real en un decodificador/codificador (CODEC).

La arquitectura propuesta fue validada primero en software por medio del entorno de programación MATLAB®. La descripción en hardware de la arquitectura diseñada, es decir, la síntesis comportamental del software, se realizó por medio del lenguaje de descripción de hardware VHDL además de ser compatible con los modelos más modernos de FPGA's (Arreglo de Puertas Programables en Campo) de las familias CYCLONE de la compañía Altera.

Para la descripción del diseño realizado en el FPGA, se utilizó el Software Quartus II versión 9.1 sp2 Full Edition, haciendo posteriormente la verificación y validación de dicha descripción mediante el uso de la herramienta de simulación Testbench con el software ModelSim versión 6.5b de Altera.

Se optó por la implementación de la arquitectura en un FPGA debido a que para hacer diseños de arquitecturas que van a operar en tiempo real, el FPGA presenta ventajas como el *paralelismo* de operaciones, el bajo consumo de energía respecto a otros dispositivos además del poder personalizar los recursos del dispositivo con el que se va a trabajar. El paralelismo de operaciones permite obtener una alta velocidad de procesamiento, es decir, alcanzar un menor tiempo de operación para la arquitectura. El bajo consumo de energía es una característica fundamental para equipos portátiles, además que el personalizar los recursos del dispositivo, por ejemplo el tamaño del bus de datos, permite optimizar el uso de los recursos del mismo.

La operación fundamental de funcionamiento de la arquitectura diseñada se basa en tener una imagen en menor escala, es decir se parte de una imagen de pequeñas dimensiones, que presenta un tipo de resolución para un tipo de dispositivo A, en este caso se parte de una imagen con resolución QVGA (320 x 240), luego dicha imagen pasará a través del filtro de sobre muestreo con un factor de escala de 2, consiguiendo una imagen con dimensiones mayores la cual puede ser utilizada por un dispositivo B, la imagen obtenida luego de ser filtrada será de resolución VGA (640 x 480). Para realizar el sobre muestreo se utilizó el formato de imagen YC_BC_R, en lugar del RGB para evitar el alto grado de correlación que se tiene entre los planos en el formato RGB lo que dificulta el proceso de codificación resultando en la reducción de la eficiencia del proceso. El sobre muestreo de la imagen se realiza en forma paralela en los planos de luminancia y en los de cromaticidad, haciendo que el proceso de sobre

muestreo se lleve a cabo en el menor tiempo posible, lo cual genera una mayor eficiencia en el proceso. Se obtuvo una frecuencia máxima de operación de 221.58 MHz, con lo que se puede llegar a procesar 1036 cuadros por segundo, con lo cual se cumplió el objetivo de poder operar a una tasa mayor de 30 cuadros por segundo (requerimiento de tiempo real).

Finalmente, se efectuaron las pruebas correspondientes para la validación de la imagen sobre muestreada en el software MATLAB® respecto a hardware, analizando las matrices resultantes de las imágenes sobre muestreadas que fueron generadas tanto por software como por el hardware.

FACULTAD DE
CIENCIAS E
INGENIERÍA

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

TEMA DE TESIS PARA OPTAR EL TÍTULO DE INGENIERO ELECTRÓNICO

Título	: Diseño de una Arquitectura de un Filtro Digital de Sobre Muestreo de Imágenes, en factor 2, de acuerdo al formato H.264/SVC sobre FPGA.
Área	: Microelectrónica # 973
Asesor	: MSc. Ing. Mario Andrés Raffo Jara
Alumno	: Christian Enrique Cano Salazar
Código	: 20067027
Fecha	: 19/10/2011

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA
DECANO
15.Nov.11

Descripción y Objetivos

El presente trabajo tiene como objetivo el diseño en FPGA de una arquitectura de un filtro digital tipo FIR para sobre muestreo de imágenes, en un factor de escalabilidad de dos, de acuerdo al formato H.264/SVC de codificación de video escalable. Primero, el algoritmo del filtro será modelado en software por medio del entorno de programación MATLAB®. Luego se procederá a realizar la síntesis comportamental del algoritmo modelado en software, la cual permitirá obtener una arquitectura en hardware del filtro. Posteriormente se efectuará la descripción de la misma utilizando el lenguaje de descripción de hardware VHDL. Finalmente la arquitectura será sintetizada para un dispositivo FPGA de la familia Cyclone II de la compañía Altera. El filtro en hardware deberá cumplir con el requerimiento mínimo de procesamiento de secuencias de video de resolución QVGA (320 x 240 pixeles) en tiempo real, es decir que procese imágenes a una tasa mayor o igual a 30 cuadros por segundo. Para sintetizar la descripción de la arquitectura en FPGA, se utilizará el Software Quartus II de la compañía Altera. Para realizar la verificación y validación del funcionamiento de dicha descripción se usará la simulación por medio de Testbench en VHDL empleándose el software ModelSimAltera de la compañía Mentor Graphics.

El Objetivo Principal del presente trabajo de tesis es realizar el diseño en FPGA de la arquitectura de un filtro digital tipo FIR para sobre muestreo de imágenes, en un factor de escalabilidad de dos, de acuerdo al formato H.264/SVC.

Los Objetivos Específicos son los siguientes:

1) Diseño del algoritmo del filtro de escalabilidad en software.

2) Diseño en hardware de la arquitectura correspondiente al filtro.

3) Descripción de la arquitectura en hardware usando VHDL, para ser sintetizada sobre un dispositivo FPGA de la familia Cyclone II.

4) Verificación funcional de la arquitectura hardware, validándola con los resultados en software.

5) Optimización de la frecuencia de operación de la arquitectura, para mejorar la tasa de procesamiento de video con la finalidad de operar en tiempo real.

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
SECCIÓN ELECTRICIDAD Y ELECTRÓNICA

MÁXIMO 50 PÁGINAS

Dr. Ing. BENJAMÍN CASTAÑEDA APHAN
Coordinador de la Especialidad de Ingeniería Electrónica

FACULTAD DE
CIENCIAS E
INGENIERÍA

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

TEMA DE TESIS PARA OPTAR EL TÍTULO DE INGENIERO ELECTRÓNICO

Titulo : Diseño de una Arquitectura de un Filtro Digital de Sobre Muestreo de Imágenes, en factor 2, de acuerdo al formato H.264/SVC sobre FPGA.

Índice

Introducción

1. Marco problemático

2. Marco teórico y presentación del asunto de estudio.

3. Aplicación en software del filtro digital de sobre muestreo de imágenes en factor de dos.

4. Diseño en hardware del filtro digital de sobre muestreo de imágenes en factor de dos.

5. Resultados

Conclusiones

Recomendaciones

Bibliografía

Anexos

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
SECCIÓN ELECTRICIDAD Y ELECTRÓNICA
Dr. Ing. BENJAMÍN CASTAÑEDA APHAN
Coordinador de la Especialidad de Ingeniería Electrónica

MÁXIMO 50 PÁGINAS

ÍNDICE GENERAL

AGRADECIMIENTOS

A mis Adorados Padres y hermano…por su gran apoyo incondicional, su cariño, su preocupación y sus grandes consejos que me ayudaron a poder cumplir mis objetivos de manera exitosa.

A Nelida… mi abuela… por ser más que una abuela, por ser un ángel.

A mi estimado asesor MSc. Ing. Mario Raffo,

por brindarme su gran amistad, además de toda la confianza, el gran apoyo

y preocupación para con mi persona para desarrollar con éxito este trabajo de tesis.

A mí estimado Grupo de Microelectrónica (GuE),

por haberme dado la oportunidad de poder descubrir mi verdadera vocación.

A Henry Block, José Quenta, Cristopher Villegas, Edward Mitacc, Andres Jacoby y demás miembros del Grupo de Microelectrónica por todos los consejos y enseñanzas, por todo el apoyo, paciencia y sincera amistad.

A Jaime Reategui, Edmundo Pozo, Giancarlo Sotelo, Richard Nole, Ruth Olivera, Jonathan Diaz, Edson Yupanqui, Daniel Lanao, Juan Ugarte y demás amigos que me brindaron su amistad y apoyo a lo largo de estos últimos años.

A Rocío Espinel, Laura Salas, Claudia Achong, Rosario Renteria y demás compañeros de trabajo por brindarme su gran amistad, su gran apoyo y por lo buenos momentos que vivimos.

A mis profesores… por todos sus consejos y enseñanzas.

A todos… mil gracias!

"Sólo triunfa en el mundo quien se levanta y busca a las circunstancias, creándolas si no las encuentra."

— **George Bernard Shaw**

INTRODUCCIÓN

El resultado de la revolución tecnológica ha permitido desarrollar ampliamente el campo de la Televisión Digital Terrestre (TDT), dado que se ha generado numerosas innovaciones en distintas áreas como el procesamiento digital de señales e imágenes y microelectrónica. Además, de la evolución rápida de las telecomunicaciones tanto para transmisiones analógicas, como para transmisiones digitales, estas últimas se han desarrollado altamente, por lo que la transmisión de datos a través de esta logra utilizar anchos de banda soportados por diversas redes de transmisión, las cuales presentan robustez ante posible perdidas de datos o errores.

En el Perú, recientemente, se ha adoptado el estándar japonés-brasileño de transmisión de TV digital SBTVD-T, debido a que este es un estándar abierto a mejoras y posibles modificaciones, por lo que se podría desarrollar tecnología con fines de aportar nuevas mejoras al estándar. El estándar SBTVD-T es una evolución del estándar japonés ISDBT-T. Dentro de las modificaciones que se realizaron figura el cambio de formato de compresión de video digital MPEG-2 a H.264/AVC, el cual presenta ciertas características que permiten conseguir una mejor calidad de imagen, este estándar logra emitir una señal que puede ser captada de forma gratuita en celulares que le sean compatibles. Esto es algo muy interesante para la realidad peruana, dado que en los últimos años se ha incrementado considerablemente el número de usuarios de telefonía móvil, así como, el sistema es apropiado para nuestro país en vista de que la base de este estándar ha sido diseñado considerando la geografía accidentada de Japón, la cual presenta similitudes a la de Perú, enfatizando en que el estándar fue diseñado para cubrir esa parte territorial donde la señal común no logra ser captada.

Como una extensión del formato H.264/AVC surge el formato H.264/SVC que permite escalabilidad de video, el cual busca aumentar la resolución/calidad del video. Teniendo como objetivo transmitir un solo tamaño de paquete de datos para que pueda ser captado por cualquier dispositivo y este pueda adaptarlo a su resolución óptima, es decir escalar la imagen del video. Por lo que el presente trabajo de tesis se tiene como objetivo desarrollar un módulo de sobremuestreo de imágenes para dispositivos que manejan resolución VGA como óptima. Tomando como premisa que la imagen transmitida es de tamaño QVGA, escalándolo en un factor de 2 para llegar a una resolución VGA.

El presente trabajo de tesis está organizado de la manera descrita a continuación. En el capítulo 1 se presenta el marco problemático del tema de tesis. En el capítulo 2 se aborda con mayor detalle el marco teórico del algoritmo del filtro que se plantea desarrollar enfatizando en el método y proceso de sobremuestreo de la imagen. En el capítulo 3 se realiza una validación en software del algoritmo de sobremuestreo verificando que se obtengan resultados válidos con la imagen sobremuetreada. En el

capítulo 4 se diseña el algoritmo del filtro en hardware utilizando síntesis comportamental para transformar el algoritmo en software a hardware. En el capítulo 5 se realiza la simulación por Testbench de la arquitectura diseñada y se comparan los resultados obtenidos por software y hardware, con lo cual se valida la arquitectura es correcta. Finalmente se presentan las conclusiones y recomendaciones.

CAPITULO 1

MARCO PROBLEMÁTICO

El gran avance de la tecnología en los últimos años, respecto a la calidad de la imagen, ha tenido un crecimiento muy rápido, por lo cual el interés por el video y la calidad de la imagen ha tenido un aumento progresivo en forma paralela con el avance de la tecnología; esta tendencia ha derivado actualmente en la necesidad de diseñar dispositivos capaces de procesar video e imágenes en alta definición, lo cual permite a su vez realizar producciones de televisión con mayor calidad, logrando obtener mayor aceptación de los usuarios del mismo. Sin embargo, en el contexto de mejorar la calidad de imagen y video se presentan limitantes importantes; por un lado, para poder obtener mejores estándares en calidad de imagen y video se necesita equipos más modernos con mejores tecnologías, lo cual a su vez hace un incremento sustancial del precio de estos equipos.

Existen varios tipos de dispositivos donde se pueden hacer estos tipos de procesamiento de imagen y video. Se puede citar a los PDSP's (Procesador de procesamiento digital de señales), FPGA's (Arreglo de Puertas Programables en Campo) [1], CUDA (Arquitectura de un dispositivo de computo unificado), GPU's (Unidad de procesamiento gráfico), entre otros [1]. Cada uno de estos dispositivos presenta características específicas, ya sea de rendimiento, velocidad, frecuencias de trabajo, consumo de energía.

Hoy en día, el estudio de la factibilidad para hacer la elección de un dispositivo, depende primordialmente de la frecuencia de operación del dispositivo, así como también del consumo de energía que pueda generar, siendo este último muy importante, dado que en la actualidad los equipos portátiles están cubriendo gran parte del mercado tecnológico, por lo cual el consumo de energía tiene que ser cada vez menor, permitiendo así dar un mayor tiempo de autonomía al dispositivo portátil.

Esta primera etapa del documento de tesis se basa en establecer un esquema de trabajo para realizar el diseño de la arquitectura de un filtro tipo FIR para sobre muestreo de imágenes de Televisión digital para poder adaptarlo a la resolución óptima requerida por cada equipo que procesa Video digital. Por tal motivo se desarrolló el formato H.264/SVC, el cual permitirá escalar la imagen a la resolución óptima de cada dispositivo. Se puede mencionar algunos tipos de resolución como QVGA, el cual es manejado por los celulares, televisores de alta definición (HDTV)[1], etc. Para propósito de esta tesis, se realizará el diseño de una arquitectura de un filtro que sobre muestree imágenes de una resolución QVGA (320x240) a una resolución VGA (640x480), posteriormente se efectuará la validación en hardware de la misma.

[1] A continuación los acrónimos entre paréntesis representarán los nombres que le preceden.

CAPITULO 2

MARCO TEÓRICO Y PRESENTACIÓN DEL ASUNTO DE ESTUDIO

2.1. Estado del Arte

2.1.1. Presentación del asunto de estudio

Actualmente se ha desarrollado rápidamente gran cantidad de dispositivos capaces de manipular video digital como iPAD's, teléfonos celulares, HDTV, por ello, es poco coherente tener un único tipo de representación de video para ser utilizado por todos estos dispositivos, por lo tanto fue establecido el formato H.264/SVC[2], el mismo que es una extensión del formato H.264/AVC[3] que es usado en el estándar de televisión japonés-brasileño SBTVD, permitiendo la adaptación a los requerimientos de los equipos de manipulación de video existentes en el mercado. Para poder tener una idea más detallada de lo que se trata se puede citar por ejemplo si se transmite una señal de video digital con una resolución de HDTV y se desea reproducir dicha señal en un celular, los cuales trabajan de manera óptima con una resolución QVGA, sería poco eficiente e injustificable decodificar dicha señal para HDTV para luego sub-amostrarla con el fin de que pueda ser reproducida de manera óptima por el celular. A su vez, tampoco sería viable ni consecuente transmitir diversos tipos de paquetes de la misma señal de video para cada uno de los dispositivos existentes. A diferencia del formato H.264/AVC, el H.264/SVC permite la escalabilidad de video, compuesta por una imagen de poca resolución a una de mayor resolución. El término escalabilidad está relacionado con el aumento de la resolución/calidad de video, partiendo de una estructura base utilizando estructuras de enriquecimiento en el proceso de codificación y decodificación del video.

Se empezará por adquirir la imagen en una escala pequeña, para este caso se trabajará con imágenes con resolución QVGA (320x240), a la cual se le aplicará un sobre muestreo por medio de un filtro del tipo FIR en los planos de luminancia (Y) y cromaticidad (Cb y Cr), según el formato de imagen YC_BC_R[4] [2]. El filtro aplicado tendrá un factor de escalabilidad de 2, el cual permitirá obtener una imagen con mayor resolución, en este caso será VGA (640x480).

[2] SVC: *Scalability Video coding.*
[3] H.264/AVC es un estándar que define un codificador/decodificador de video de alta compresión.
[4] Se usa el formato YC_BC_R en lugar del RGB para evitar el alto grado de correlación que tienen entre los planos en el formato RGB lo que dificulta el proceso de codificación resultando en la reducción de la eficiencia del proceso.

En un principio, todo el proceso de sobre muestreo de la imagen QVGA, previamente mencionado, será desarrollado utilizando el entorno de programación MATLAB® [2], con lo cual se efectuará la validación respectiva de la calidad de la imagen sobre muestreada, dado que se está utilizando filtro tipo FIR de 4-taps. Si la calidad de la imagen no es óptima se procederá a realizar el mismo proceso, pero utilizando filtro tipo FIR de 6-taps [3], con lo que se estaría mejorando la calidad de la imagen procesada.

Finalmente, luego de realizar la correcta validación del proceso de sobre muestreo mediante el uso del software MATLAB®, se procederá a realizar el diseño de la arquitectura del filtro para ser implementada en un dispositivo FPGA (Arreglo de Puertas Programables en Campo) evaluando el correcto funcionamiento y la eficiencia de procesamiento de la información y así poder hacer la elección del dispositivo FPGA de las familias Cyclone de la compañía Altera que proporcione la mayor eficiencia y calidad [1] en términos de frecuencia de operación y de consumo de recursos.

2.1.2.- Estado de la investigación:

Respecto del tema antes mencionado, referente al estándar H.264/SVC [2], se puede agregar que este estándar logra explorar tres tipos de escalabilidad de video digital: temporal, espacial y cualitativa o de fidelidad. Un dispositivo que es compatible con el estándar H.264/SVC es capaz de poder decodificar una señal de video, ya sea digital o no, generando una representación de una imagen que contenga información del video decodificada con una resolución espacial, temporal o cualitativa.

Además, dicho estándar proporciona soporte a la escalabilidad espacial, la cual está basada en un concepto de decodificación por capas. Es importante mencionar que el sistema de codificación de escalabilidad espacial definido para el estándar H.264/SVC se basa en un mecanismo de predicción inter-capas, el cual está constituido por tres procesos, bien definidos, como son predicción de movimiento, predicción residual y intra-predicción.

En el presente tema de tesis el enfoque se centrará en la intra-predicción, más precisamente en lo que refiere a la implementación del hardware de sobre muestreo espacial de las imágenes de las tramas de video.

En la figura 1 se muestra la estructura típica de un codificador de H.264/SVC que presenta 2 capas: La capa 0 es la capa base, la cual cuenta con la imagen en menor escala y la capa 1 cuenta con la imagen en mayor escala.

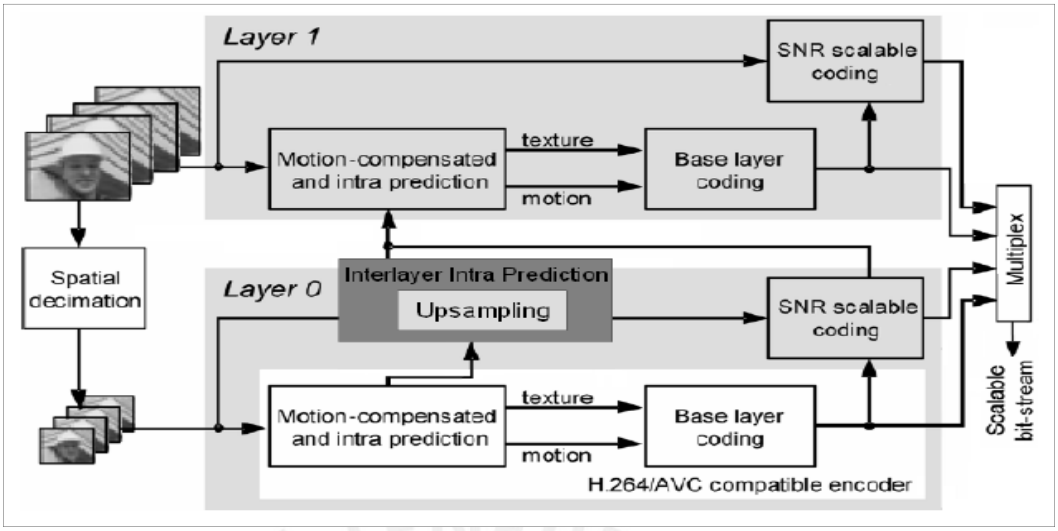


Figura N° 1. Diagrama de bloque de un codificador de H.264/SVC [4].

Como se puede observar el bloque responsable del proceso de sobre muestreo, el cual es el objeto de estudio en esta tesis, se encuentra resaltado en la figura 1.

Lo que se busca es poder partir de una imagen en pequeñas dimensiones, la cual será aplicada un dispositivo Y, para luego poder hacer el proceso de sobre muestreo de dicha imagen para que tenga dimensiones mayores y así pueda ser utilizado por el dispositivo Z.

El presente trabajo está enfocado para trabajar con una resolución QVGA (320x240) en la capa base (capa 0) y con una resolución VGA (640x480) en la capa mejorada (capa 1). Ambas capas usan el formato de imagen YCbCr. Este formato está establecido por el Sector de Radiocomunicaciones (UIT-R)[5] como norma para trabajar con video digital. Dicha norma es la *Rec. UIT-R BT.601*[6][5]. El formato YCbCr es utilizado, en lugar del formato RGB, para evitar el alto grado de correlación que se presenta entre los planos Rojo, Verde y Azul del formato RGB, lo que dificulta el proceso de codificación resultando en la reducción de la eficiencia del proceso de sobre muestreo que será planteado a continuación [6].

[5] ITU-R: Sector de Radiocomunicaciones de la UIT (UIT-R) desempeña un papel fundamental en la gestión del espectro de frecuencias radioeléctricas y de las órbitas de los satélites, recursos naturales limitados que suscitan una demanda creciente por parte de un gran número, cada vez mayor, de servicios como el servicio fijo, móvil, de radiodifusión, de radioaficionados, de investigación espacial, de telecomunicaciones de emergencia, de meteorología, de los sistemas mundiales de posicionamiento, de observación del medio ambiente y de comunicaciones que se encargan de la seguridad de la vida humana en la tierra, en el mar y en el aire. (www.itu.int)

[6] **Rec. UIT-R BT.601** - Codificación de los parámetros de estudio de la televisión digital.

2.1.2.1.- Método de Sobre Muestreo de imágenes de acuerdo al estándar H.264/SVC:

Como se observa en la figura 1 el método de sobre muestreo se encuentra explícitamente identificado en el diagrama de bloques del codificador H.264/SVC. Este bloque es el responsable de incrementar la resolución de la imagen de la capa base (capa 0) a una resolución mayor en la capa mejorada (capa 1).

Este método es aplicado cuando el modo de predicción de un bloque es inter-capas y el bloque correspondiente en la capa base ha sido decodificada usando intra-predicción. Se puede mencionar también que este trabajo de sobre muestreo es utilizado en el método de predicción residual inter-capas.

Para la realización del proceso de sobre muestreo se emplearán filtros del tipo FIR (*Finite Impulse Response*), como se empleará el formato de imagen YCbCr, se utilizará tres filtros, uno para cada plano de Luminancia (Y) y de cromaticidad (Cb y Cr). Para el plano de Luminancia (Y) es aplicado un filtro tipo FIR de 4-taps y para los planos de cromaticidad rojo y azul (Cb y Cr) se aplica un filtro Bilineal. El orden de aplicación de los filtro será de manera simultánea para los tres planos de la imagen, lo cual permitirá que el proceso de sobre muestreo se realice a una mayor velocidad, es decir a un menor tiempo. Los filtros serán aplicados, primero horizontalmente y luego verticalmente. Cabe mencionar que el uso de diferentes tipos de filtros para los planos de luminancia y cromaticidad es básicamente para incrementar la calidad del filtrado sin tener un importante incremento en su complejidad. En investigaciones previas se ha realizado el filtrado con filtros bilineales para ambos planos, resultando una importante reducción en la calidad del filtraje, según estudios realizados por el Dr. Shijun Sun y presentado en una patente de Estados Unidos (USA) [3] bajo el título de “Métodos y sistemas para el diseño de filtros de sobre muestreo”, se hacen análisis y cálculos para definir los coeficientes de los índices de los filtros que se aplican a los planos de luminancia y cromaticidad. Sin embargo, la elección del índice del filtro a escoger dependerá del factor de escala de sobre muestreo.

2.1.2.2. Proceso de Filtrado en el Plano de Luminancia y en el Plano de Cromaticidad:

Como se mencionó anteriormente se aplicará un filtro de tipo FIR de 4-taps [3] para el plano de luminancia y un filtro tipo FIR bilineal para el plano de cromaticidad. Ambos con un factor de escalabilidad de 2. En la codificación de video escalable SVC el sobre muestreo está compuesto por un grupo de 16 filtros, como se muestra en las tablas 1 y 2 de la referencia [7], donde la elección de los índices se realiza en función al factor de escala de sobre muestreo.

Para el propósito de esta tesis se utiliza un filtro de sobre muestreo con un factor de 2, por lo cual, según la referencia [6] menciona, se debe utilizar los filtros de índice 12 y índice 4 en los planos de

luminancia y de cromaticidad, Si se observa nuevamente dichos índices en las tablas 1 y 2 para poder tener los coeficientes de los filtros se tienen las siguientes fórmulas:

- Filtros para el plano de luminancia

$SL_{12} = (1*A + 8*B + 28*C + 3*D) / 32$ (1)

$SL_4 = (3*A + 24*B + 8*C + 1*D) / 32$ (2)

- Filtros para el plano de cromaticidad

$SC_{12} = (0*A + 8*B + 24*C + 0*D) / 32$ (3)

$SC_4 = (0*A + 24*B + 8*C + 0*D) / 32$ (4)

Se hace la división entre 32[7] en ambos casos para poder mantener el valor del resultado en el intervalo de 0 a 255, es decir normalizado a 1 byte (8 bits).

Tabla N° 1.- Filtro de sobre muestreo para el plano de luminancia [6].

Índice de fase del filtro	Coeficientes de interpolación del filtro			
	e[-1]	e[0]	e[1]	e[2]
0	0	32	0	0
1	-1	32	2	-1
2	-2	31	4	-1
3	-3	30	6	-1
4	-3	28	8	-1
5	-4	26	11	-1
6	-4	24	14	-2
7	-3	22	16	-3
8	-3	19	19	-3
9	-3	16	22	-3
10	-2	14	24	-4
11	-1	11	26	-4
12	-1	8	28	-3
13	-1	6	30	-3
14	-1	4	31	-2
15	-1	2	32	-1

[7] La implementación en hardware de una operación de división eleva considerablemente el consumo de recursos, por ello es importantísimo tener en cuenta que la división entre 32 puede realizarse por medio de operaciones de desplazamiento (shift) a la derecha en 5 posiciones.

Tabla N° 2.- Filtro de sobre muestreo para el plano de cromaticidad [6].

Indice de fase del filtro	Coeficientes de interpolación del filtro			
	e[-1]	e[0]	e[1]	e[2]
0	0	32	0	0
1	0	30	2	0
2	0	28	4	0
3	0	26	6	0
4	0	24	8	0
5	0	22	10	0
6	0	20	12	0
7	0	18	14	0
8	0	16	16	0
9	0	14	18	0
10	0	12	20	0
11	0	10	22	0
12	0	8	24	0
13	0	6	26	0
14	0	4	28	0
15	0	2	30	0

Los planos de luminancia y cromaticidad de la imagen QVGA (320 x 240) que ingresa a la capa base serán segmentadas en macro bloques de 4 x 4 pixeles, las cuales serán ingresadas en orden a cada filtro y como se establece un factor de escala de 2, a la salida tendremos macro bloques de 8 x 8 pixeles con los cuales se generará la nueva imagen sobre muestreada, que en este caso será de una resolución VGA (640 x 480), verificando el factor de escalabilidad de 2.

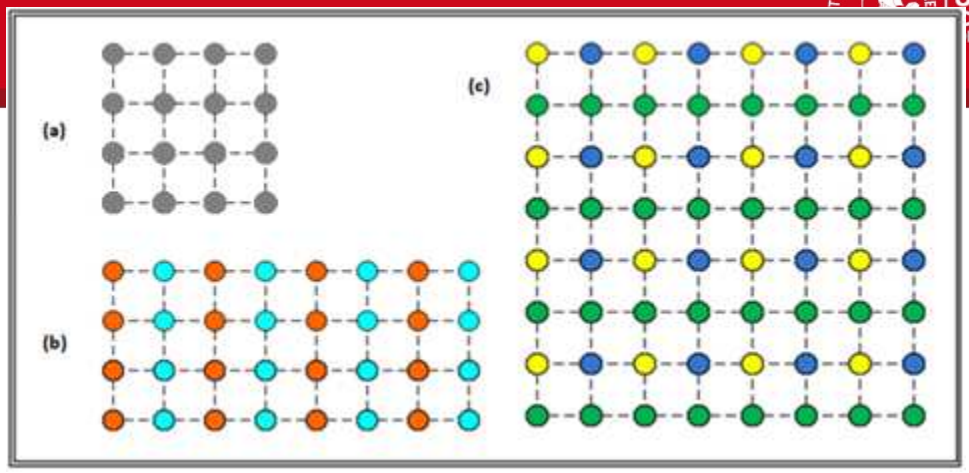


Figura N° 2.- (a) macrobloque de 4x4 de la imagen original, (b) resultado del filtro horizontal y (c) resultado del filtro vertical [4].

Para ver de una manera más gráfica el proceso de sobre muestreo se puede ver la figura 2, donde se muestra las operaciones y el resultado de los filtros ya sea de luminancia y cromaticidad partiendo de un macrobloque de 4 x 4 pixeles (figura 2a), en dicha figura los datos originales están representados gráficamente por círculos de color gris. Luego al ser filtrado horizontalmente se obtenga un macrobloque de 4 x 8 (figura 2b), donde los círculos de color naranja representan gráficamente a los datos obtenidos al aplicar el filtro de índice 12 (no son los originales son los obtenidos al procesar el macrobloque de 4 x 4 pixeles) mientras que los datos obtenidos al aplicar el filtro de índice 4 (en el macrobloque de 4 x 4) están representados gráficamente por los círculos de color celeste. Finalmente, se realiza el filtrado vertical al macrobloque de 4 x 8 pixeles, obtenido después del filtrado horizontal, obteniendo un macrobloque de 8 x 8 pixeles (figura 2c), en el cual los datos procesados por el filtro de índice 12 están representados gráficamente por los círculos de color amarillo y color azul (los círculos de color amarillo representan al resultado de filtrar los círculos de color naranja y los de color azul representan al resultado de filtrar los círculos de color celeste), mientras que los datos procesados por el filtro de índice 4 están representados gráficamente por los círculos de color verde (luego de procesar el macrobloque de 4 x 8).

CAPITULO 3

APLICACIÓN EN SOFTWARE DEL FILTRO DIGITAL DE SOBREMUESTREO DE IMÁGENES EN FACTOR DE DOS

3.1. Aspectos generales de la aplicación en software MATLAB.

Actualmente, existen diferentes opiniones y metodologías entre que debe implementarse en software y en hardware. Las implementaciones en software vienen ganando la aceptación del público debido a la gran versatilidad y las ventajas que presenta sobre lo que es implementación en hardware para aplicaciones reales. Por ejemplo, reconocimiento de expresiones faciales en imágenes de cámaras digitales modernas. Se tienen otras aplicaciones en lo referente a control de procesos, donde el tiempo de procesamiento de la información no es un requerimiento fundamental del sistema. También se puede observar aplicaciones en la medicina muy importantes como por ejemplo, diagnostico de enfermedades mediante la clasificación y detección de patrones. Todas estas aplicaciones mencionadas se realizan sobre software en un computador.

La existencia de una gran variedad de herramientas de diseño (toolbox) y las diferentes herramientas de programación de alto nivel, que permiten realizar e implementar los diseños en menor tiempo, son dos grandes ventajas que presenta la implementación en software. Sin embargo, existen algunas limitaciones importantes a tomar en cuenta, como la recarga del procesador (CPU); por ejemplo, tareas como clasificación y localización consumen gran parte de los recursos del mismo, lo que origina que se reduzca el desempeño global del sistema. Además, cabe mencionar que aplicaciones en tiempo real son poco eficientes en software, por ello es que estos tipos de aplicaciones se desarrollan en hardware.

Esta tesis está enfocada a desarrollar una aplicación en tiempo real, por lo tanto el diseño e implementación final será sobre hardware, en este caso un dispositivo de arreglos programables en campo (FPGA). Sin embargo, antes de realizar el diseño del algoritmo de sobre muestreo en hardware, se realizará la validación del mismo en software porque, como se mencionó, éste tiene herramientas de programación de alto nivel que permiten hacer la implementación en menos tiempo, para así poder verificar la obtención de los resultados deseados antes de pasar a la etapa de implementación en hardware.

El software que se utilizará para hacer la validación del algoritmo es el MATLAB® R2010a debido a la gran versatilidad que presenta en el campo del procesamiento de imágenes. Los resultados obtenidos a nivel de software permitirán realizar análisis críticos comparándolos con los resultados deseados.

3.2. Esquema de desarrollo de la aplicación en software.

El algoritmo de sobre muestreo, descrito anteriormente, es un modelo matemático que permitirá realizar un incremento de la resolución, en un factor de 2, de la imagen de entrada, la cual tiene una resolución del tipo QVGA (320x240), generando una imagen de resolución tipo VGA (640x480), la cual presenta el doble de resolución de la imagen de entrada. Todo este proceso se trabajará con una imagen en formato YCbCr, dado que es un formato establecido para trabajar con imágenes digitales[8]. En la Figura 3 se muestra el proceso a seguir en la programación en MATLAB®.

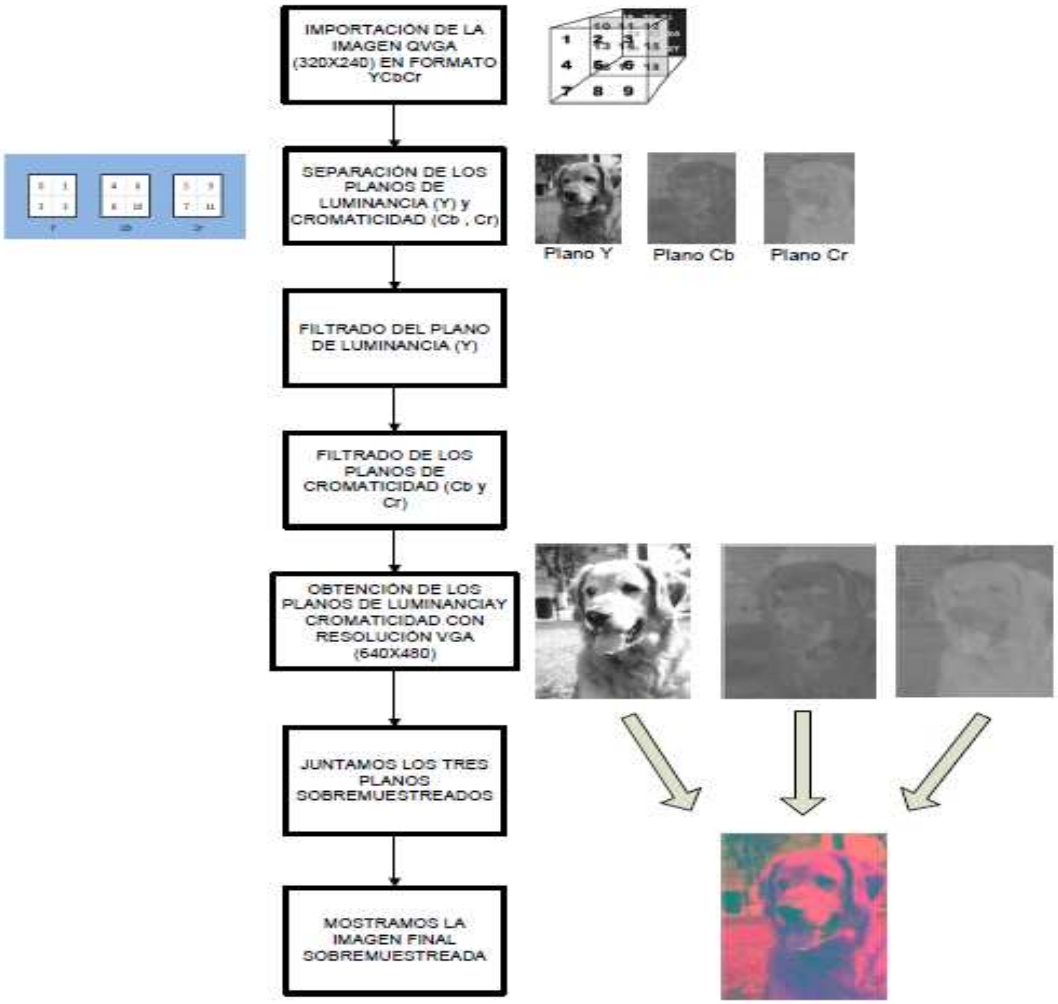


Figura N° 3.- Diagrama de flujo del algoritmo del filtro.

[8] Tema expuesto en el capitulo 2.1.2

3.3. Verificación del algoritmo del filtro de sobremuestreo por medio de una aplicación en Software.

Para verificar el funcionamiento de las características que se está tomando en la arquitectura diseñada del filtro del sobremuestreo se implementó el algoritmo en una aplicación software en el entorno de programación MATLAB®. Los resultados obtenidos a través de esta aplicación servirán para constatar los resultados obtenidos por la arquitectura hardware diseñado del filtro.

3.4. Resultados obtenidos de la aplicación en software del algoritmo del filtro

Las pruebas se realizaron con una imagen QVGA (320x240 pixeles), llamada "perrito.jpg". Entonces si el algoritmo puede funcionar para una imagen, entonces también puede ser aplicable a una secuencia de video, dado que un video es una secuencia de imágenes. Es importante poder mencionar que se realizó una prueba del algoritmo del filtro de sobremuestreo con un video de resolución QCIF, llamado "bus"[9]. Con esto se puede confirmar que el algoritmo puede ser aplicado para operar con secuencias de video y al operar a una frecuencia adecuada se puede llegar a cumplir con el requerimiento de operación en tiempo real, lo cual se busca al diseñar una arquitectura del algoritmo del filtro en hardware.

3.4.1. Resultados obtenidos para el plano de luminancia (Y):

El plano de luminancia de una imagen en formato YCbCr es el que define la forma de la imagen [2][16], por ello es que al apreciar la imagen en este plano se puede tener una idea de la forma de la imagen, pero no de los colores de la misma. En la figura N° 4 se aprecia el plano de luminancia original de la imagen "perrito.jpg" antes de ser sobre muestreada y en la figura N° 5 se aprecia el plano de luminancia sobremuestreado.

3.4.2. Resultados obtenidos para el plano de cromaticidad azul (Cb) y plano de cromaticidad rojo (Cr):

El plano de cromaticidad azul y el plano de cromaticidad rojo de una imagen en formato YCbCr son los que definen los colores de la imagen. En la figura N° 6 se puede observar el plano de cromaticidad azul de la imagen original "perrito.jpg" y el la figura N° 7 se aprecia la imagen sobre muestreada de la misma.

Asimismo, En la figura N° 8 se puede observar el plano de cromaticidad rojo de la imagen original "perrito.jpg" y en la figura N° 9 se aprecia la imagen sobre muestreada de la misma.

[9] Secuencia de video "bus" tomada de la siguiente pagina web < http://media.xiph.org/video/derf/>

Si se juntan los tres planos sobremuestreados se tiene la imagen original sobremuestreada en formato YCbCr, la cual se aprecia en la figura N° 10, y si se desea apreciarla en sus colores reales tiene que convertirse al formato RGB [2] [16], como se muestra en la figura N° 11.

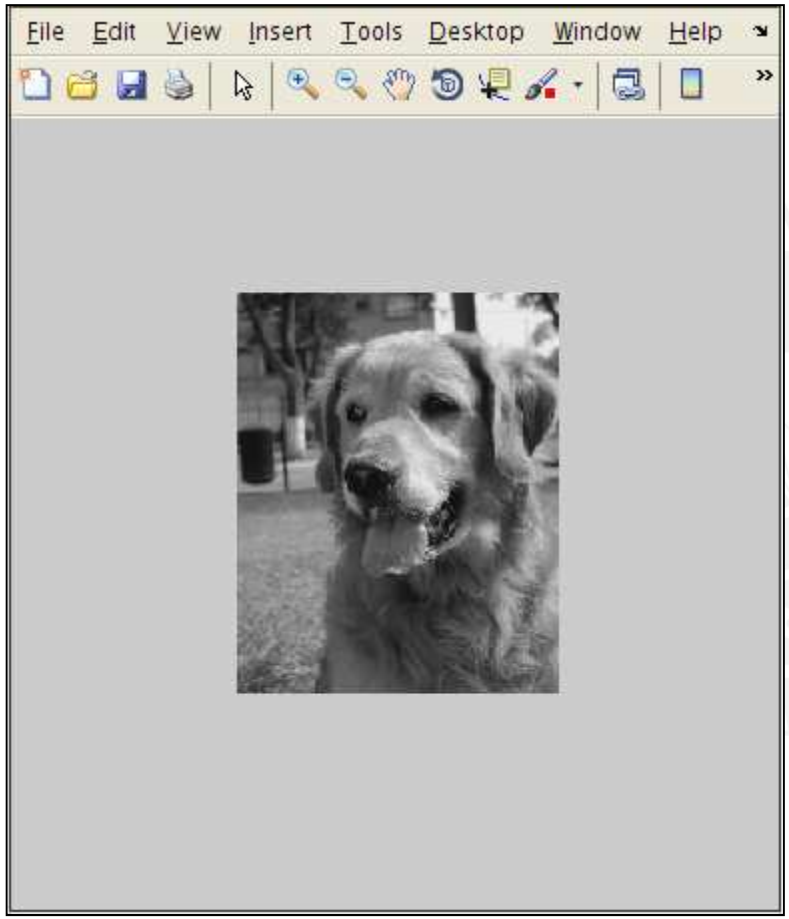


Figura N° 4.- Plano de luminancia de la imagen QVGA.

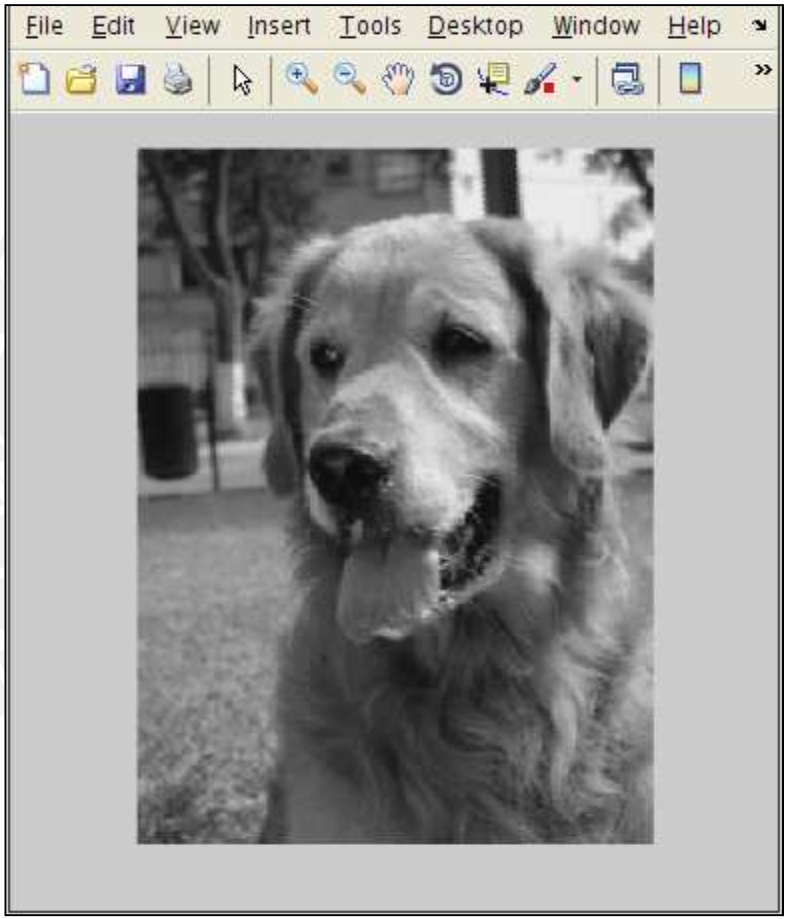


Figura N° 5.- Plano de luminancia de la imagen VGA.

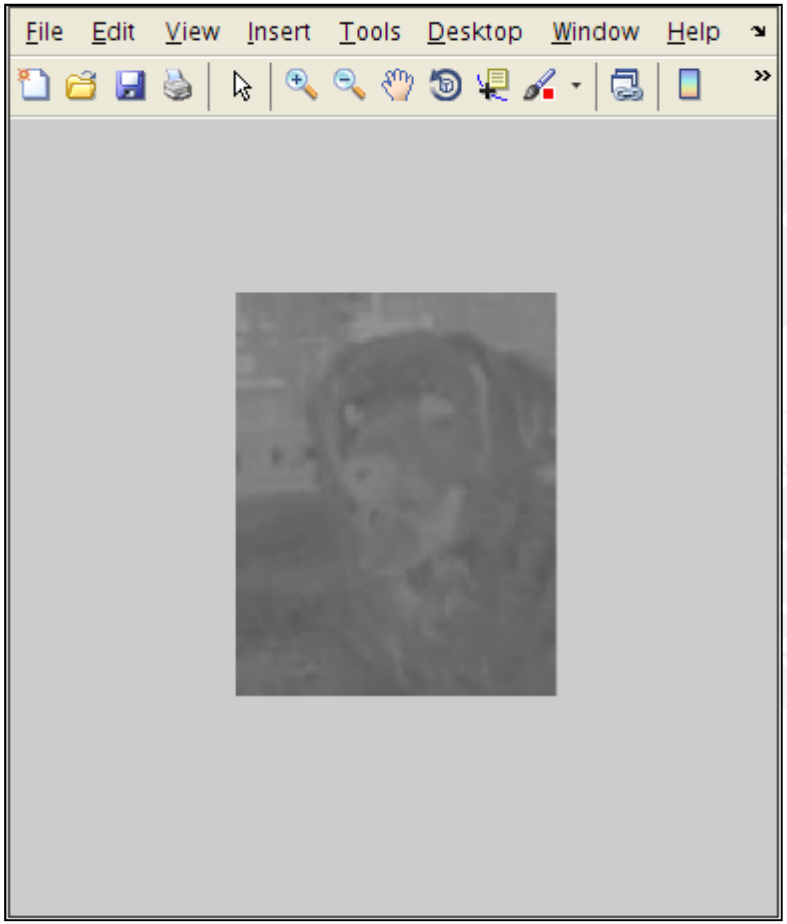


Figura N° 6.- Plano de cromaticidad azul de la imagen QVGA.

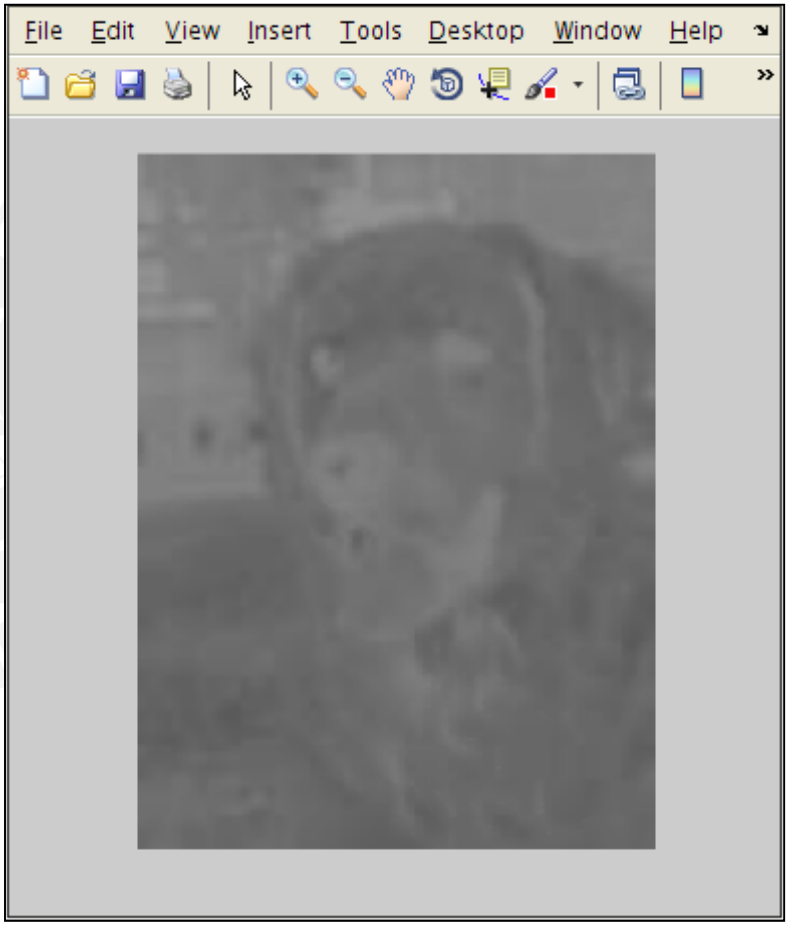


Figura N° 7.- Plano de cromaticidad azul de la imagen VGA.

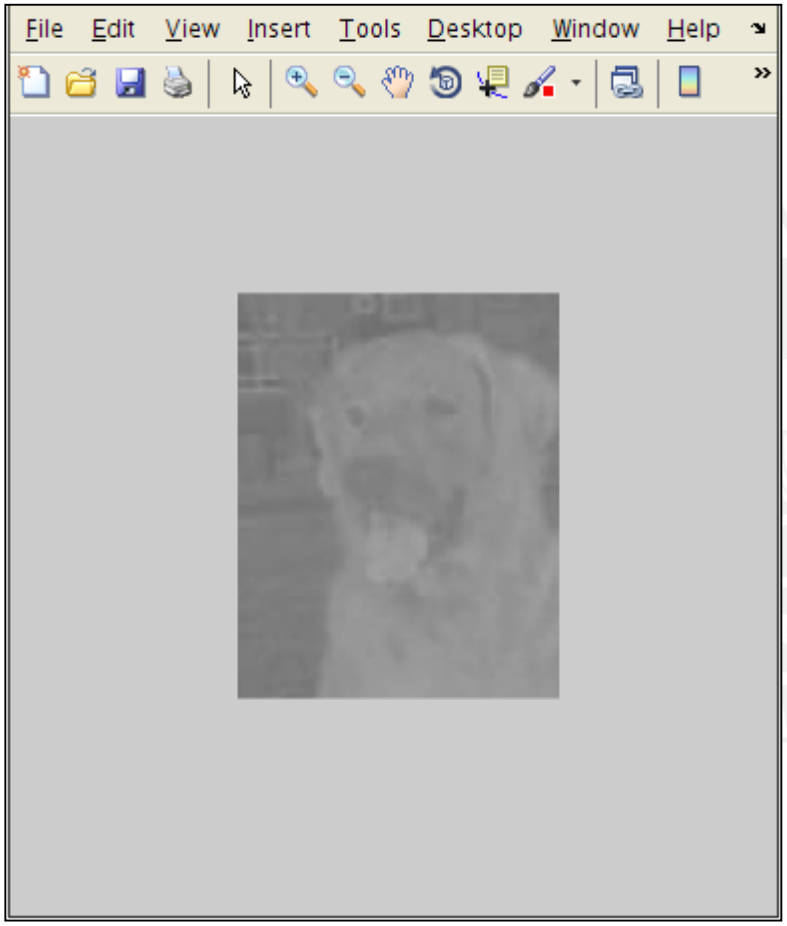


Figura N° 8.-Plano de cromaticidad rojo de la imagen QVGA.

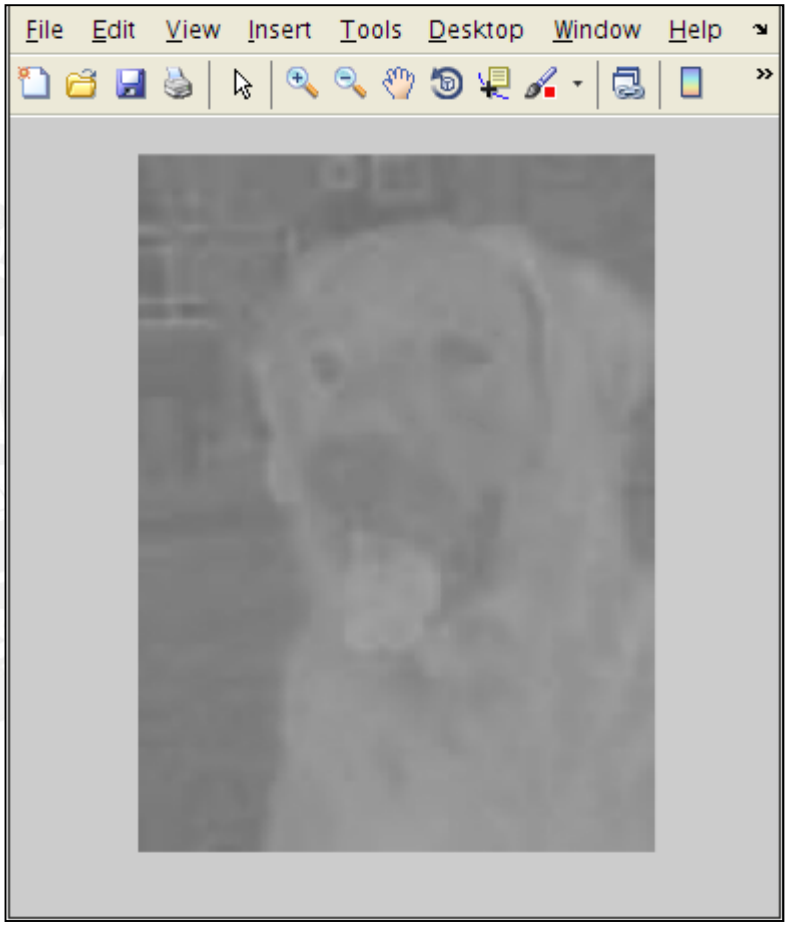


Figura N° 9.-Plano de cromaticidad rojo de la imagen VGA.

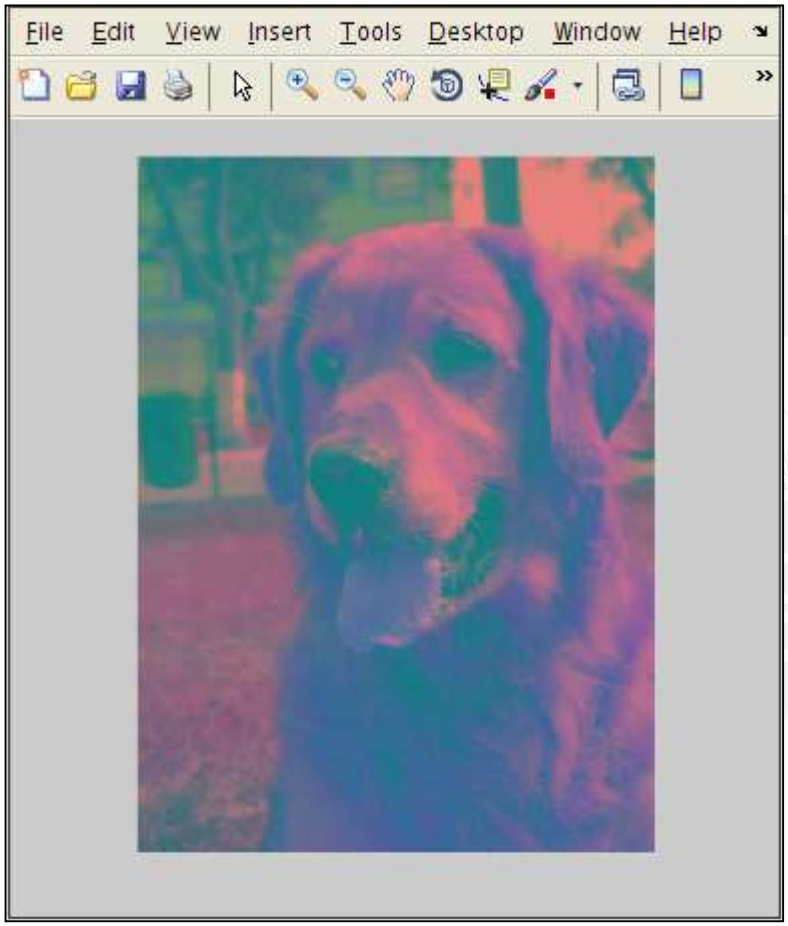


Figura N° 10.- Imagen en formato YCbCr de la imagen VGA.

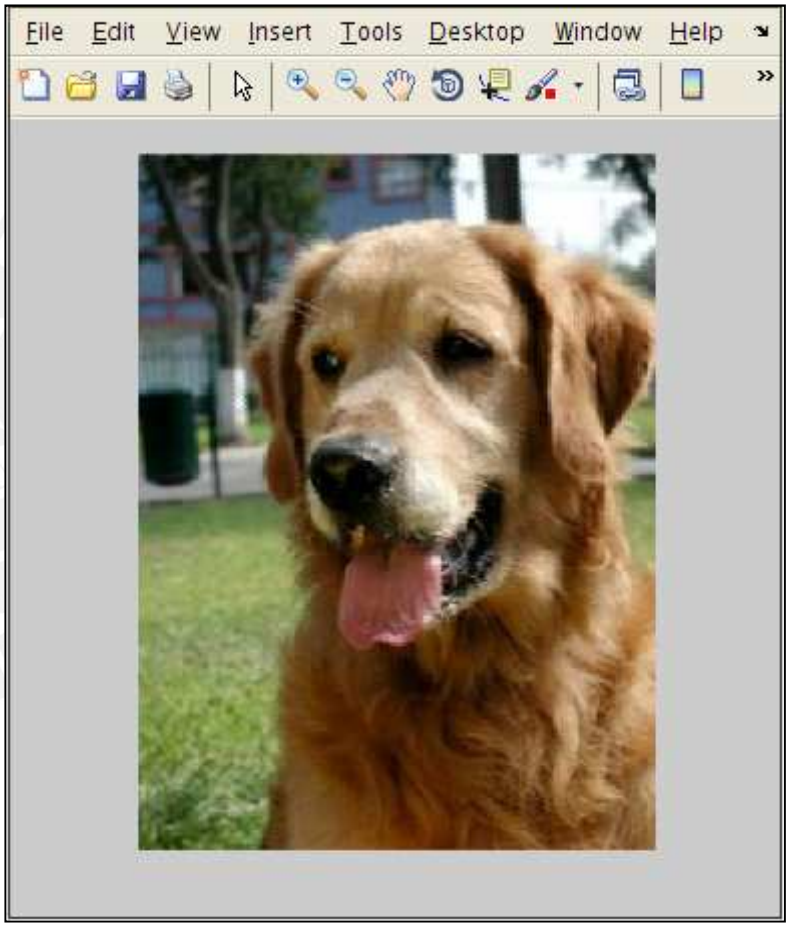


Figura N° 11.- Imagen en formato RGB de la imagen VGA.

CAPITULO 4

DISEÑO EN HARDWARE DEL FILTRO DIGITAL DE SOBREMUESTREO DE IMÁGENES EN FACTOR DE DOS

Con la validación en software del algoritmo de sobre muestreo, al verificar los resultados deseados, se pasa a la siguiente etapa, donde mediante el método de síntesis comportamental, el algoritmo realizado en software se convierte a descripción en hardware. Lo cual permitirá tener la arquitectura del filtro de sobre muestreo basado en el código que ha sido validado en software previamente, esperando los mismos resultados obtenidos mediante software. Las pruebas respectivas de esta arquitectura se realizaran en un dispositivo FPGA Familia Cyclone II modulo DE2 de Altera.

4.1. Dispositivo Lógico Programable FPGA:

El término dispositivo lógico programable (Programmable Logic Device, PLD) se trata de un chip de propósito general para la implementación y desarrollo de circuitos lógicos. Presenta un conjunto de elementos lógicos agrupados de distintas maneras. Es decir, es un bloque que contiene compuertas lógicas e interruptores programables, los cuales conectan entre si las compuertas para así formar el circuito lógico que sea requerido [8].

Estos dispositivos se clasifican de acuerdo al grado de complejidad que presenta y a la cantidad de recursos que presenta el dispositivo. Así, existen los SPLD (Simple PLD), dentro de los cuales tenemos a los PLA(Programmable Logic Array), PAL's (Programmable Array Logic), PLA/PAL registradas (Registered PLA/PAL), GAL (Generic PAL), pasando a un plano más complejo tenemos a los CPLD's (Complex PLD) y los FPGA's (Field Programmable Gate Array) [1] [8].

De acuerdo a lo mencionado se puede notar que un FPGA (Arreglo de puertas programable en campo) es un tipo de PLD que es capaz de soportar implementaciones de circuitos lógicos y sistemas digitales relativamente grandes y además son configurables en campo, lo que quiere decir que se puede configurar en el lugar del diseño, junto al diseñador [8]. Un FPGA presenta una arquitectura virgen que puede ser configurada de acuerdo a las necesidades del diseñador, logrando así una gran diversidad de aplicaciones, ya sean simples o complejas. El almacenamiento de estos diseños se realiza por lo general en una SRAM (Static Random Access Memory), pero también presenta otras tecnologías de configuración como la EEPROM/ FLASH y la antifuse [1][8].

4.2. Justificación del uso de Dispositivos Lógicos Programables FPGA:

Existen gran variedad de dispositivos capaces de procesar información a una gran velocidad y con un alto rendimiento, pero a la vez el poder tener mayor rendimiento y eficiencia significa un mayor consumo de energía del dispositivo o el incremento del tamaño del dispositivo. Por ejemplo tenemos dispositivos como los DSP's, PDSP's, ASIC's, GPU's, CUDA's, etc. Todos estos dispositivos tienen un rendimiento muy alto, mayor velocidad, algunos son más baratos, comparado con un FPGA, pero dentro de los objetivos planteados lo que se busca es poder llegar a obtener un menor tiempo de procesamiento a la vez de ahorrar en términos de consumo de energía, por lo tanto debido al menor tiempo de procesamiento que posee, permite que se pueda operar de una manera mucho más eficiente en tiempo real.

Un dispositivo FPGA puede trabajar en tiempo real de manera muy eficiente, el tiempo de procesamiento de la información es mucho menor comparado a otros dispositivos mencionados anteriormente, debido a que se diferencia de los otros dispositivos, en que posee la característica del paralelismo y además un FPGA consume mucho menor energía.

4.3. Procesador digital de señales (DSP):

Es un dispositivo compuesto por un sistema basado en un procesador o microprocesador tipo RISC que contiene un conjunto de instrucciones, un hardware y un software optimizados, el cual está diseñado para ser empleado en aplicaciones que requieran operaciones numéricas a muy alta velocidad, por lo tanto, al poseer estas características es muy utilizado para el procesamiento y representación de señales en tiempo real. Una característica de este dispositivo, que lo diferencia de un FPGA, es que trabaja de manera secuencial, lo cual hace que el tiempo de procesamiento del DSP sea mayor a la de un FPGA, debido a que el FPGA se ejecuta de manera paralela [1].

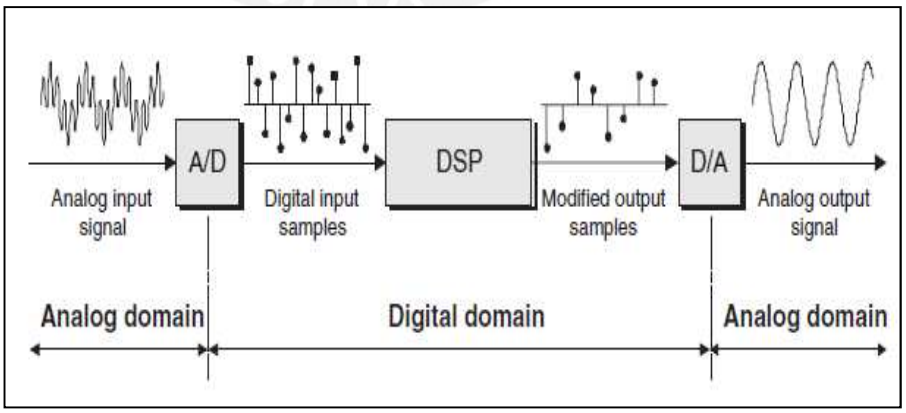


Figura N° 12.- Diagrama de bloques del funcionamiento de un DSP [1].

- **El paralelismo**: Es una característica resaltante de los FPGA's, por lo que si el algoritmo realizado se puede dividir en partes pequeñas para ejecutarlo en paralelo, entonces es más conveniente trabajar con un FPGA en aplicaciones en donde se precisa un menor tiempo de operación.

- **Precisión personalizada**: Otra característica importante de los FPGA's, es decir que si se necesita trabajar con números específicos de bits, el FPGA puede ser personalizado para la cantidad de bits requerido, lo cual permite optimizar la utilización de la cantidad de bits, sin embargo, un DSP está programado para trabajar con un número determinado de bits, el cual no se puede modificar. Esta es otra ventaja de los FPGA's respecto a los DSP's.

Por otro lado, cabe mencionar que si en caso no se requiera trabajar con mucha velocidad ni un requerimiento especial de precisión, entonces podría ser más barato y fácil el usar un PDSP.
Actualmente, los procesadores programables de señal digital (PDSP) son veloces, baratos y versátiles, pero para propósitos de esta tesis el tema de consumo de energía es muy importante, debido a que la aplicación de este bloque de sobre muestreo está definido para realizarse futuras mejoras y adaptarlo para aplicaciones sobre equipos portátiles, y al hablar de estos tipos de equipos es tratar de ahorrar la mayor cantidad de energía posible para que dicho dispositivo, ya sea un celular, iPAD, entre otros, que tenga un mayor tiempo de autonomía.

4.4. Unidad de procesamiento gráfico (GPU):

Es un tipo de dispositivo especial diseñado para aplicaciones de procesamiento de gráficos u operaciones de coma flotante, son utilizados en sistemas embebidos, teléfonos móviles, computadoras personales, estaciones de trabajo y consolas de videojuegos. Los GPU modernos tienen un alto grado de rendimiento, dado que son capaces de procesar 10 millones de polígonos por segundo [9]. Se puede ver que estos dispositivos son muy eficientes en aplicaciones gráficas, pero a la vez esto hace que el dispositivo consuma gran energía para realizar todas sus funciones de manera óptima. En el aspecto del consumo de energía, el FPGA consume menos energía que los GPU, dado que lo que se busca es diseñar un módulo para ser usado en equipos portátiles, donde el consumo de energía en un tema muy importante. Los GPU's, en especial los CUDA de NVIDIA, presentan una mayor velocidad de procesamiento y superan ampliamente en rendimiento a los FPGA's, dado que cuenta con mucho paralelismo y las frecuencias en las que opera son altas, pero consumen hasta 19 veces más energía que los FPGA's; Además, el número de operaciones son mínimas (sumas y restas), por lo tanto no se

requiere de un dispositivo altamente desarrollado para realizar operaciones básicas, por ello es que el FPGA es el dispositivo más idóneo para la implementación de esta arquitectura [10].

Cabe mencionar que los otros tipos de soluciones hardware, a parte del FPGA, como son los DSP, GPU, PDSP presentan ventajas y desventajas dependiendo de la aplicación que se desea desarrollar. Sin embargo, existen soluciones del tipo software, como la que nos brinda el MATLAB®, la cual es muy útil para el desarrollo de aplicaciones de procesamiento de imagen. Pero si se desea realizar aplicaciones que operen en tiempo real, se recomienda la utilización de soluciones hardware porque presentan mayor eficiencia y rendimiento [9]. Las soluciones software como es el caso del MATLAB® no es la más idónea para aplicaciones en tiempo real.

4.5. Arquitectura Genérica en un FPGA:

Por lo general, los dispositivos FPGA están conformados por elementos básicos como bloques lógicos, recursos o bloques de interconexión y unidades de entrada y salida [1].

Los bloques lógicos están compuestos por tablas de verdad (Look at table, LUT), las cuales son circuitos configurables que permiten realizar cualquier tipos de implementación de circuitos o funciones lógicas de un número determinado de entradas, por flip-flops y multiplexores [8].

Para poder tener control y dominio sobre las interconexiones requeridas entre los bloques lógicos, los recursos de interconexión y las unidades de entrada y salida se configuran dichas interconexiones mediante técnicas de configuración, como por ejemplo, una de la más conocida y utilizada llamada tecnología de celdas SRAM.

Adicionalmente, se puede mencionar que dependiendo del tipo de dispositivo FPGA, este puede contar como bloques adicionales, ya sea bloques PLL, multiplicadores, MAC’s, DSP’s, microprocesadores, entre otros.

4.6. Arquitectura del FPGA Cyclone II de ALTERA:

El dispositivo Cyclone II de Altera es un tipo de FPGA basado en un arreglo bidimensional de bloques lógicos (Logic Array Blocks, LAB). Además de estar formados por elementos reconfigurables SRAM de 1.2v, 90nm con un alta densidad de elementos lógicos (logic elements, LEs) entre 4608 y 68416, hasta 1.1 Mbits de memoria RAM embebida [11]. Cuenta con bloques dedicados adicionales como bloques de memoria RAM embebidas de 4K, de 13 a 150 multiplicadores embebidos de 18x18, de 2 a

4 PLL's, soporte de memoria externa de alta velocidad (DDR, DDR2, SDR, SDRAM y QDRII SRAM) y unidades de entrada/salida avanzadas [11]. La cantidad exacta de cada uno de los elementos mencionados depende del dispositivo elegido, pues estas características están presentes en toda la familia Cyclone II.

La arquitectura del FPGA Cyclone II presenta un diseño bidimensional compuesto por filas y columnas entre los cuales están los arreglos de bloques lógicos (LABs). La disposición de los componentes se aprecia en la figura N° 13, donde se puede observar bloques de memoria, multiplicadores embebidos, los PLL's, los arreglos lógicos y los bloques de entrada/salida (Input/Output Element, IOE). Por lo tanto, estos dispositivos están compuestos por LAB's, IOE's, PLL's, memorias y multiplicadores.

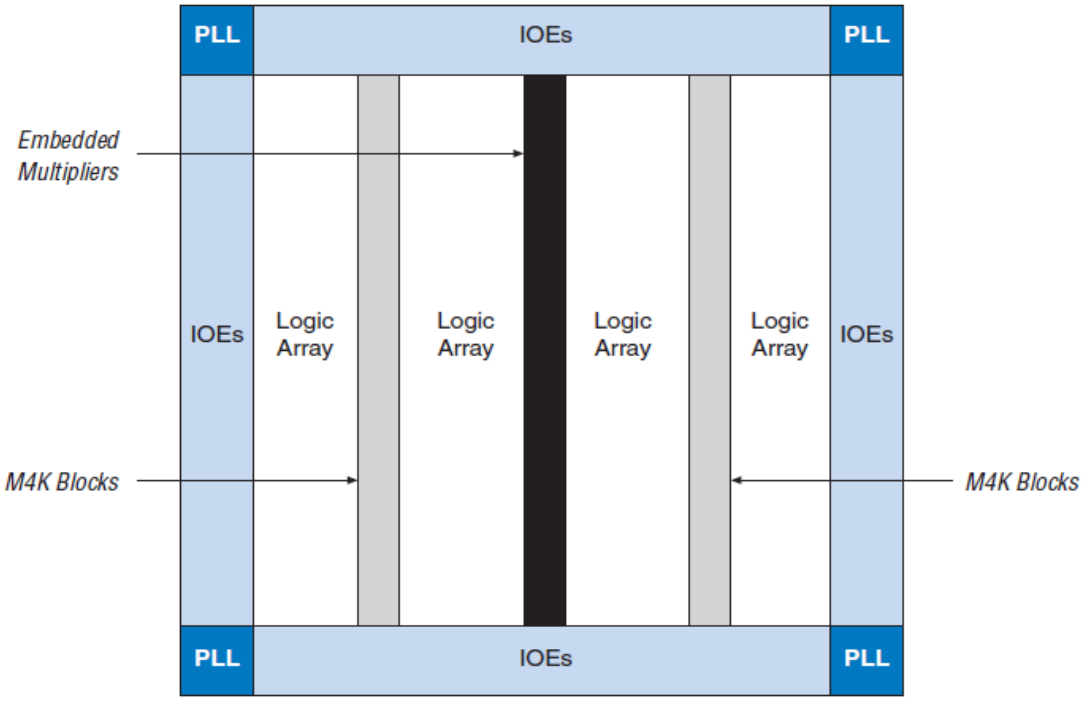


Figura N° 13.- Diagrama de bloques del FPGA Cyclone II EP2C20 [11].

Existe una estructuración entre los arreglos lógicos en base a los arreglos de bloques lógicos (LABs), que básicamente consisten en un arreglo de elementos lógicos (LE's) y recursos de interconexión locales (Local InterConnect) [11]. Además, este dispositivo está conformado por filas y columnas de interconexión para realizar interconexiones entre los elementos lógicos (LE's) de diferentes arreglos de bloques lógicos (LAB's). En la figura N° 14 se muestra la arquitectura de un arreglo de bloque lógico (LAB).

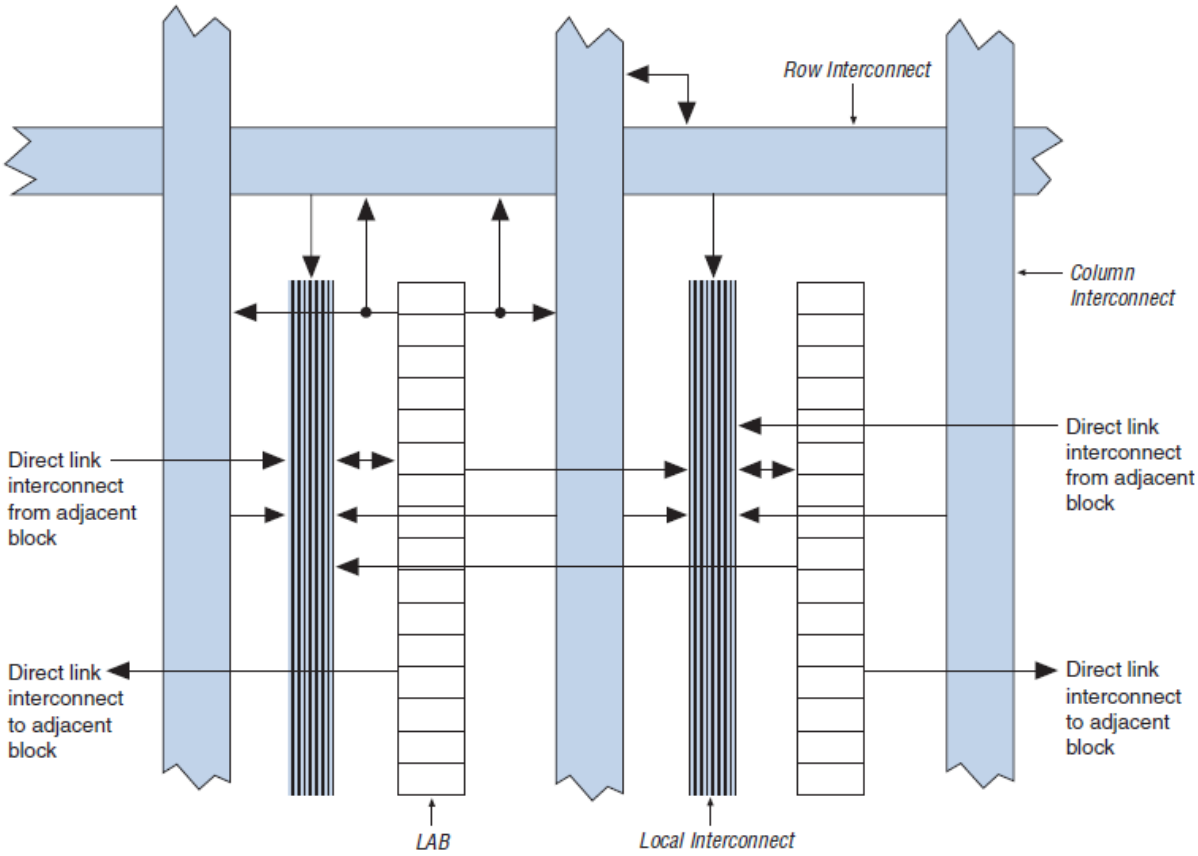


Figura N° 14.- Estructura del arreglo de bloque lógico (LAB) del FPGA Cyclone II [11].

4.7. Lenguaje de descripción de hardware VHDL:

Para realizar la descripción de hardware, es muy usado es el VHDL, el cual es un lenguaje de descripción de hardware (HDL: Hardware Description Language) que describe el comportamiento de un circuito lógico o sistema electrónico digital: cuyo sistema físico o real tiene la posibilidad de ser implementado en el dispositivo respectivo.

Mayormente es utilizado, junto con el Verilog HDL, para poder realizar la implementación de circuitos en FPGA's, CPLD's, PLD's y en el ámbito relacionado con los ASIC's, logrando que estos diseños sean portables y reutilizables, además permite tener la ventaja de que el VHDL o el Verilog HDL son independientes de la tecnología, característica que lo hace muy valorado.

No está demás mencionar que las sentencias que se utilizan en VHDL son inherentemente concurrentes, es decir, se ejecutan de manera simultánea/paralela. Sin embargo las sentencias que se encuentran dentro de procesos, funciones o procedimientos son ejecutadas de manera secuencial.

Para realizar esta labor, utiliza las denominadas herramientas EDA (Electronic Design Automation), las cuales fueron creadas exclusivamente para las tareas de síntesis, implementación y simulación de circuitos lógicos previamente descritos. Algunas son ofrecidas dentro de los entornos de desarrollo de los fabricantes (Quartus II, ISE); mientras que otros son desarrollados y ofrecidos por compañías EDA especializadas como Leonardo Spectrum (Sintetizador de Mentor Graphics), Synplify (Sintetizador de Synplicity) y ModelSim (simulador de Mentor Graphics), esta última es una herramienta muy utilizada para realizar simulaciones Testbenches [12] [13].

Respecto de los diseños lógicos se puede decir que estos han llegado a ser de alta calidad desde que los diseñadores optimizan los diseños por medio de reglas de diseño. La industrias actuales de IC's (Circuitos integrados) están empezando a tomar como base dichas reglas de diseño para poder sintetizar de una manera más óptima [14].

Todo este proceso de diseño de circuitos, fue dividido en niveles por los doctores Daniel D. Gajski y Robert H. Kuhn en el siguiente artículo de la referencia [15] presentado en 1983, quienes realizan una representación de los niveles de abstracción de un diseño en un diagrama, conocido como Diagrama "Y" o Diagrama de Gajski, el cual separa el proceso de diseño en niveles jerárquicos, los cuales son representaciones funcionales (niveles de comportamiento), representación estructural (nivel estructural) y representación física (nivel físico), tal como se muestra en la figura N° 15 [15].

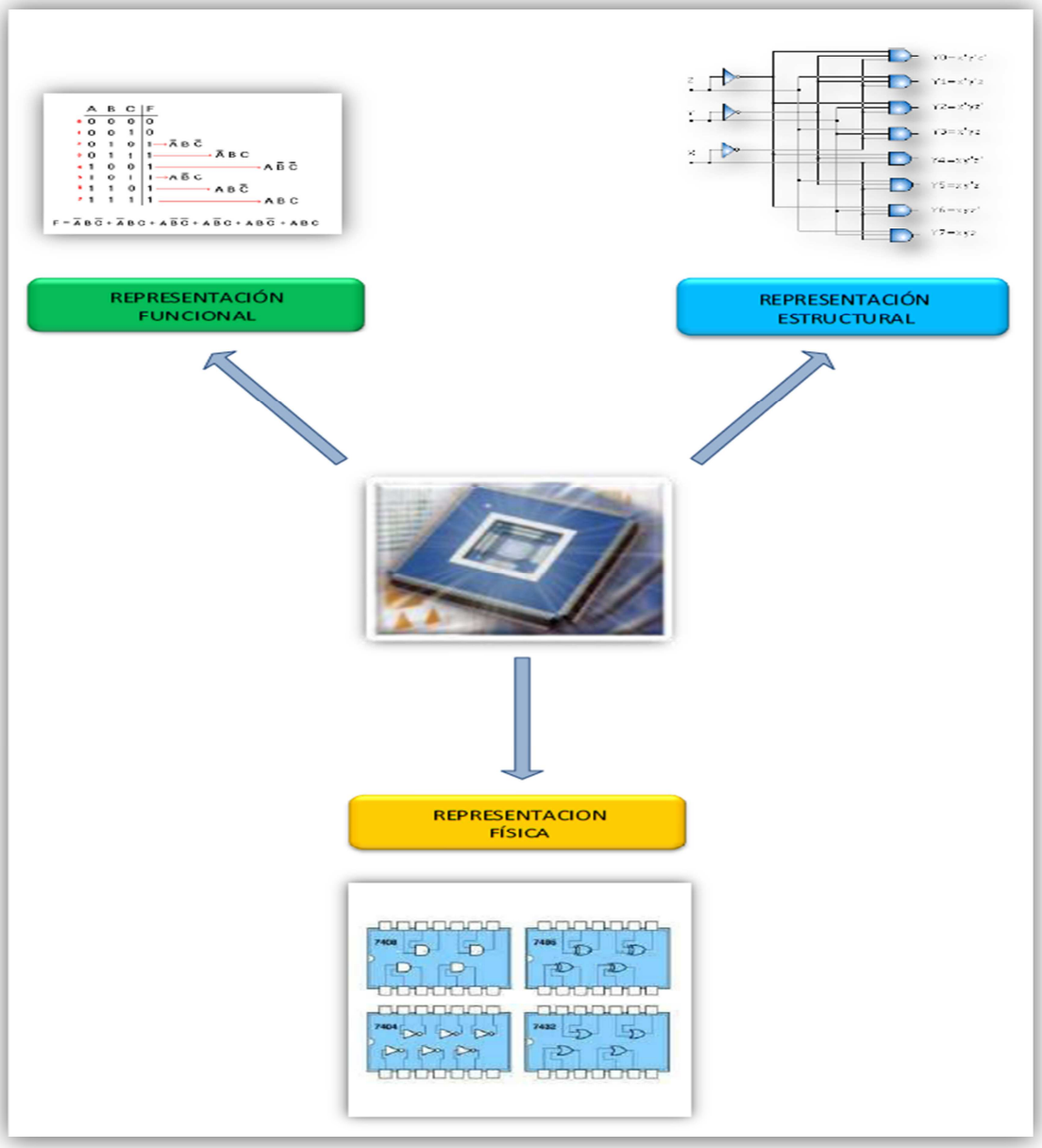


Figura N° 15.- Diagrama "Y" o Diagrama de Gajski.

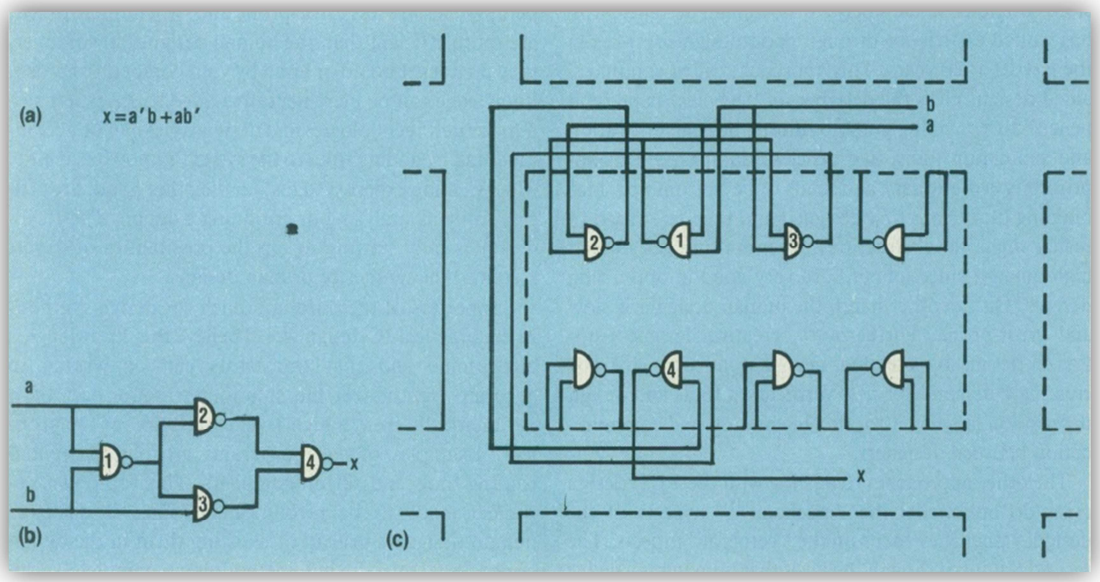


Figura N° 16.- Representación funcional (a), estructural (b) y física (c) de una expresión booleana [15].

Representación funcional: Los consumidores, por lo general, están interesados en la funcionalidad del circuito lógico o chip, así como también de cómo está constituido. En la figura 16a, se muestra una expresión booleana x= a'b + ab', donde sus entradas son a y b, y su salida es 'x'. Este tipo de representación no dice nada referente a la estructura del diseño, donde se puede aclarar que la representación funcional está enfocado primordialmente al comportamiento del diseño, por ello es que también es conocido como un nivel jerárquico a nivel de comportamiento [15].

Representación Estructural: Esta representación es el puente o enlace entre la representación funcional y la representación física. Es un desarrollo esquemático o mapeado de la representación funcional a través de compuertas y conectores lógicos, donde se tiene como limitaciones temas como el costo, área y tiempo de desarrollo. Por ejemplo, teniendo como referencia la expresión booleana antes mencionada (Figura N° 16a), la cual puede ser representada con cuatro compuertas lógicas tipo NAND de dos entradas, tal como se muestra en la figura N° 16b.

Cabe mencionar que este tipo de representación no especifica ningún parámetro físico, como la posición de las compuertas lógicas en un circuito impreso o en un chip de silicio. Sin embargo, algunas veces la representación estructural sirve como representación funcional [15].

Representación Física: En una etapa final de un proceso de diseño lógico, se tiene a la representación a nivel físico, el cual se caracteriza por ignorar el funcionamiento del diseño lógico desarrollado

previamente. Por ejemplo, si se usa un arreglo de compuertas lógicas tipo NAND de dos entradas, entonces es posible enlazar la representación a nivel estructural de la figura 16b, dando una ubicación física de las conexiones y compuertas lógicas dentro de un circuito impreso o chip de silicio, tal como se muestra en la figura N° 16c [15].

4.8. Diseño de la arquitectura del filtro digital hardware de sobre muestreo sobre el FPGA CYCLONE II de ALTERA

4.8.1. Hipótesis de solución del diseño de la arquitectura

Con la base teórica mencionada tanto en el marco problemático, como en el marco teórico, el poder hacer un diseño de una solución en hardware permite aprovechar de manera mucho más óptima el paralelismo de operaciones, respecto a una implementación en software. La utilización óptima del paralelismo, característica principal de los FPGA's, permite incrementar la frecuencia de operación de una arquitectura, con lo que permite realizar una mayor cantidad de operaciones en intervalos de tiempo menores.

Lo que se buscó tener muy en cuenta al realizar el diseño de la arquitectura es el tema relacionado con el paralelismo, entonces si se obtiene un diseño lo más paralelo posible se puede obtener frecuencias de operación muy altas, lo cual a su vez genera que se tenga una arquitectura de mayor tamaño. Esta desventaja hoy en día ya no genera muchos problemas debido a que con el avance de la tecnología los circuitos integrados cada vez se están haciendo más pequeños. Por lo anteriormente descrito, es que el diseño es lo más paralelo posible, más que nada en la etapa de filtrado (horizontal y vertical).

Implementar una arquitectura hardware sobre un FPGA permite obtener grandes ventajas, dado que dicho dispositivo tiene los recursos lógicos suficientes como para poder desarrollar una aplicación de alta complejidad operando en tiempo real. Además, al implementar la arquitectura del filtro, luego de realizar la síntesis comportamental del algoritmo en software, se optimiza la frecuencia de operación aprovechando el alto grado de paralelismo del flujo del algoritmo desarrollado en hardware.

4.8.2. Diseño de la arquitectura a partir de la síntesis comportamental del algoritmo implementado en software.

Luego de validar el algoritmo del filtro de sobre muestreo de imágenes, en factor dos, por medio de una aplicación en el entorno de programación MATLAB® (el código fuente figura en el anexo A). Dicho algoritmo en software, fue implementado de manera propia sin tomar como referencia algún otro algoritmo en MATLAB®. Luego, al realizar la síntesis comportamental de la misma se diseño una arquitectura hardware del algoritmo aprovechando de la manera más optima el alto grado de paralelismo del mismo con la finalidad de optimizar la frecuencia de operación para cumplir con requerimiento del tiempo de operación (mayor o igual a 30 cuadros por segundo).

Cabe mencionar, que el presente trabajo de tesis tiene como base teórica el trabajo citado en la referencia [4]. Asimismo, el trabajo citado en la referencia [6] permitió tener un conocimiento más amplio en la elección de los coeficientes del filtro para tomarla en cuenta en el diseño de la arquitectura.

Los criterios para realizar un óptimo diseño de la arquitectura son tres:

- La frecuencia de operación de la arquitectura.
- Cantidad de recursos hardware utilizados.
- Cantidad de ciclos de operación.

Es bueno mencionar que la potencia consumida de la arquitectura debe tomarse en cuenta si se tiene como objetivo, a largo plazo, la utilización de esta arquitectura en dispositivos portátiles, los cuales dependen primordialmente del tiempo de duración de sus baterías, las cuales a su vez dependen del consumo de potencia del dispositivo en general.

4.8.3. Arquitectura del algoritmo del filtro de sobre muestreo de imágenes en factor dos.

El diseño de la arquitectura completa del filtro de sobremuestreo se dividirá en tres sub-arquitecturas[10], una para cada plano de la imagen en formato YCbCr (Plano de Luminancia (Y), Plano de cromaticidad Azul (Cb), Plano de cromaticidad rojo (Cr)). Las cuales funcionarán de forma paralela para asi poder alcanzar un tiempo de procesamiento de la imagen que nos permita cumplir con el objetivo de operar en tiempo real. En la figura N° 17 se muestra un esquema de funcionamiento de la arquitectura completa.

[10] En el capítulo 2 se explicaron cada una de las sub-arquitecturas que conforman la arquitectura del filtro de sobremuestreo.

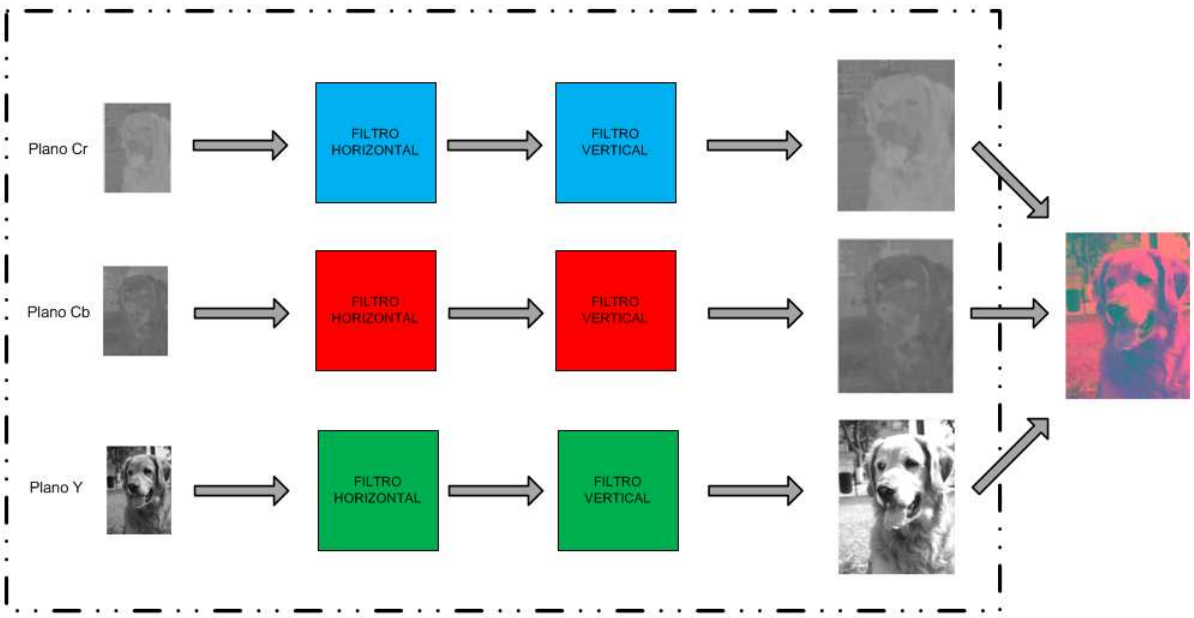


Figura N° 17.- Esquema general del funcionamiento de la arquitectura completa.

Como se observa en la figura N° 17 la entrada a la arquitectura son los tres planos (Y, Cb y Cr) de la imagen, donde cada plano se procesa por separado y en paralelo e independientemente uno de otro, generando cada uno su imagen sobre muestreada del plano correspondiente, tal como se muestra en la figura 9.

La arquitectura del algoritmo del filtro para los tres planos tiene el mismo esquema de desarrollo, y están compuestos por los siguientes bloques principales:

- Bloque Conversor Serie-Paralelo
- Bloque Contador de pixeles
- Bloque Contador de filas
- Bloque de Registro de 4 entradas de 8 bits
- Bloque de Registro de 8 entradas de 8bits
- Bloque de Registro de 16 entradas de 8 bits
- Bloque de Filtrado Horizontal
- Bloque de Filtrado Vertical

Las arquitecturas desarrolladas para cada plano tienen todos los bloques mencionados previamente. Las diferencias entre la arquitectura del plano de luminancia y de los planos de cromaticidad, están en el bloque de filtrado horizontal y en el bloque de filtrado vertical. Estos dos bloques presentan diseños

diferentes en su arquitectura debido a que los coeficientes, a usar, de los filtros son distintos tal como se menciona en el capítulo 1. Por consiguiente, se procede a explicar cada bloque, haciendo énfasis en los bloques de filtrado horizontal y el bloque de filtrado vertical.

4.8.3.1. Arquitectura del bloque conversor Serie-Paralelo:

Este bloque está conformado por un arreglo de Flip-Flops tipo D interconectados de forma serial, tal como se muestra en la figura N° 18. La función de este bloque dentro de esta arquitectura es hacer ingresar cada pixel de forma serial, el cual se realiza con cada flanco de subida, hasta que ingresen los primeros 4 pixeles y así poder almacenarlos en un registro para que luego pasen a ser procesados.

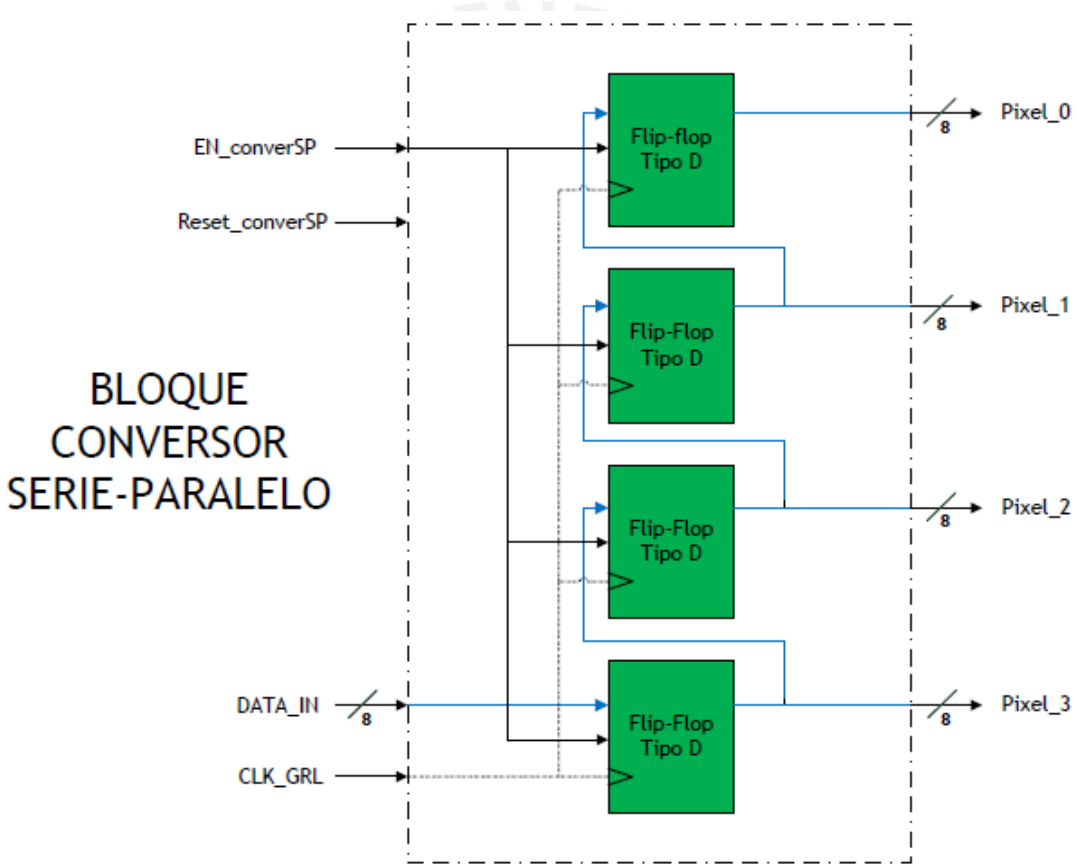


Figura N° 18 Bloque conversor Serie-Paralelo.

4.8.3.2. Bloque Contador de pixeles:

Este bloque está compuesto por un Flip-Flop tipo D y un comparador. La función principal de este bloque es llevar la cuenta de que ingresen los 4 primeros pixeles. Entonces, conforme va ingresando cada pixel el contador, se va incrementando y comparándose con el valor de 4.

Si se cumple esta igualdad la salida del contador será '1', caso contrario será '0', dicha señal de salida irá a la máquina de estados (FSM) de la arquitectura general, para que este deshabilite el conversor Serie-Paralelo y no deje ingresar ningún pixel más hasta que los 4 pixeles que ingresaron pasen a otra etapa de procesamiento de los mismos. Además, dicha salida funciona como habilitador del bloque contador de filas, el cual se explicará más adelante. Entonces, una vez que los cuatro primeros pixeles pasan a otra etapa, el conversor serie-paralelo se vuelve a habilitar para proseguir con el ingreso de los siguientes 4 pixeles. El diagrama del bloque en mención se puede observar en la figura N° 19.

4.8.3.3. Bloque Contador de filas:

Este bloque está compuesto, al igual que el contador de pixeles, de un Flip-Flop tipo D y de un comparador, como se presenta en la figura N° 19. Este bloque incrementa su cuenta cada vez que se complete el ingreso de los 4 pixeles del macrobloque de 4x4, con lo cual permite llevar la cuenta del número de filas de los macrobloques que están ordenados de manera vertical en el archivo de texto que se ingresa a la arquitectura cuando se realiza la simulación por Testbench. Entonces cuando se completa la lectura de todas las filas de los macrobloques de un cuadro (imagen o un plano de imagen de un video), el contador indicará con una señal en alta que se ha completado la lectura de toda la imagen, dicha señal ingresa a la máquina de estados (FSM), la cual emite una orden de reiniciar, tanto el contador de pixeles como el de filas para empezar a leer una nueva imagen y procesarla. Este contador permitirá que la arquitectura se pueda aplicar al procesamiento de video.

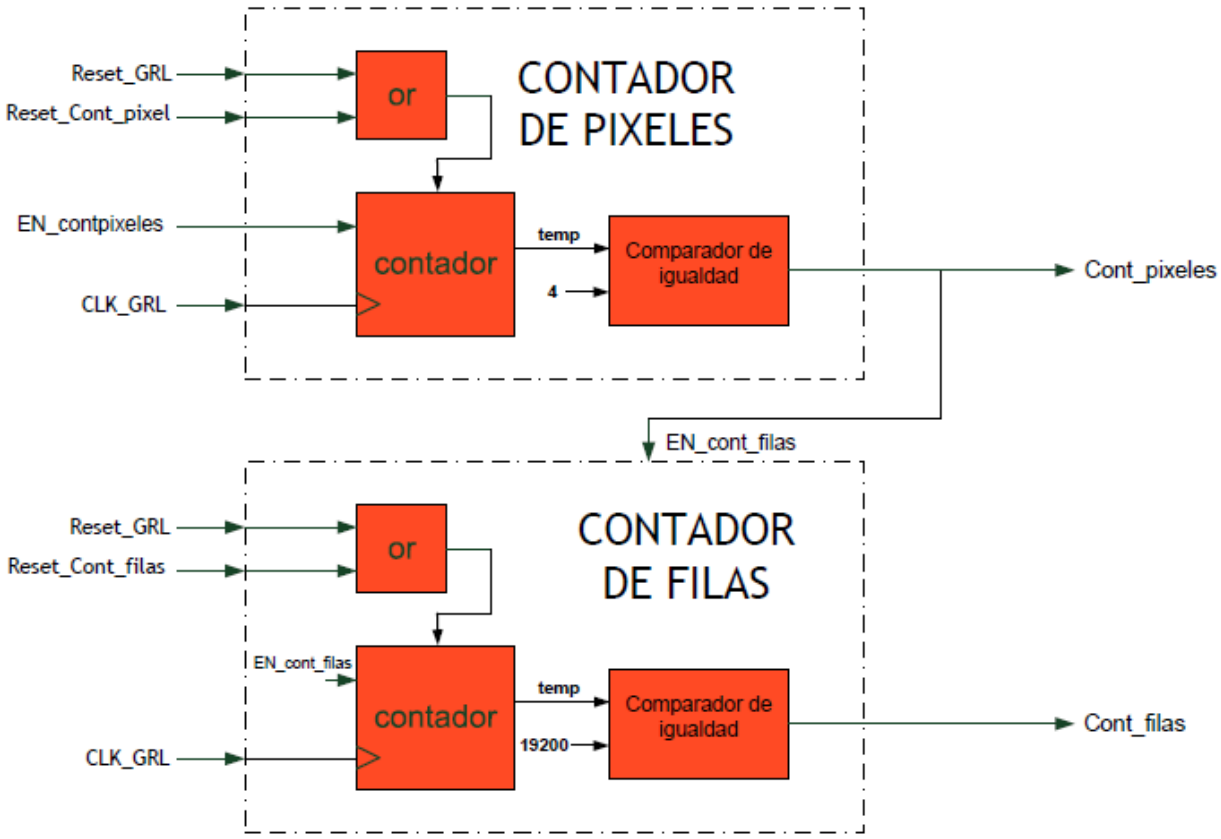


Figura N° 19. Bloques del contador de pixeles y contador de filas de la imagen a procesar.

4.8.3.4. Bloque de registro de Modulo 4:

Este bloque está compuesto por un arreglo de cuatro Flip-Flop's tipo D en paralelo, que tiene 4 entradas de 8 bits cada una, tal como se muestra en la figura N° 20. Este bloque es el encargado de almacenar los 4 pixeles de entrada de los macrobloques, una vez que estos se hayan completado, para que luego dichos pixeles ingresen al bloque de filtrado horizontal.

4.8.3.5. Bloque de registro de Modulo 8:

Este bloque está compuesto por un arreglo de 8 Flip-Flop's tipo D en paralelo, que tiene 8 entradas de 8 bits cada una, tal como se muestra en la figura N° 21. La función de este bloque es el encargado de almacenar los 8 pixeles que se generan cuando los 4 pixeles originales pasan por el bloque de filtrado horizontal.

4.8.3.6. Bloque de registro de Modulo 16:

Este bloque está compuesto por un arreglo de 16 Flip-Flops's tipo D en paralelo, que tiene 16 entradas de 8 bits cada una, tal como se muestra en la figura N° 22. Este bloque se encarga de almacenar la

respuesta final del bloque de filtrado vertical, el cual recibe como entradas los 8 pixeles de respuesta del bloque de registro de modulo 8.

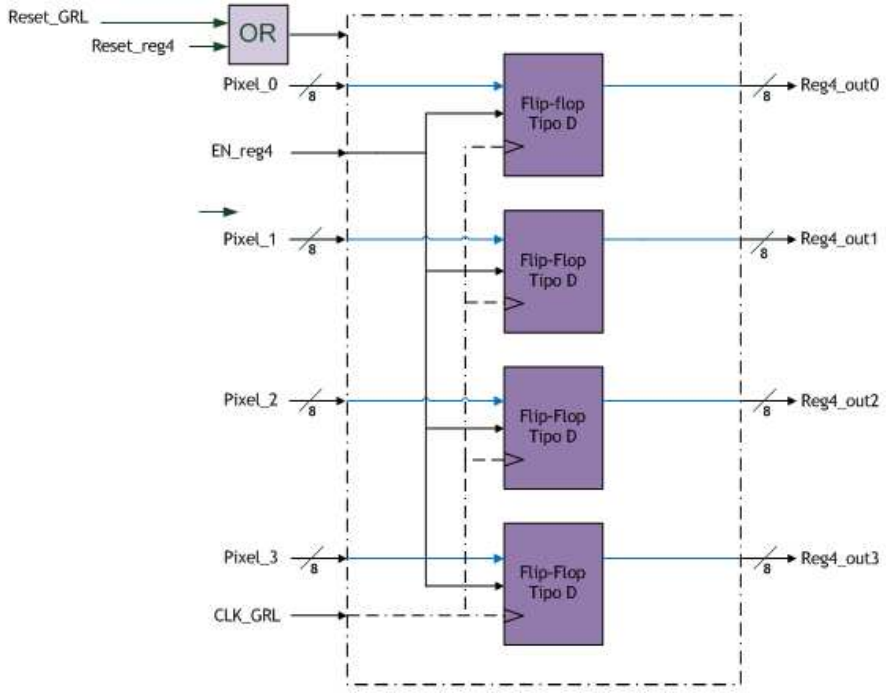


Figura N° 20. Diagrama del bloque de registro de modulo 4.

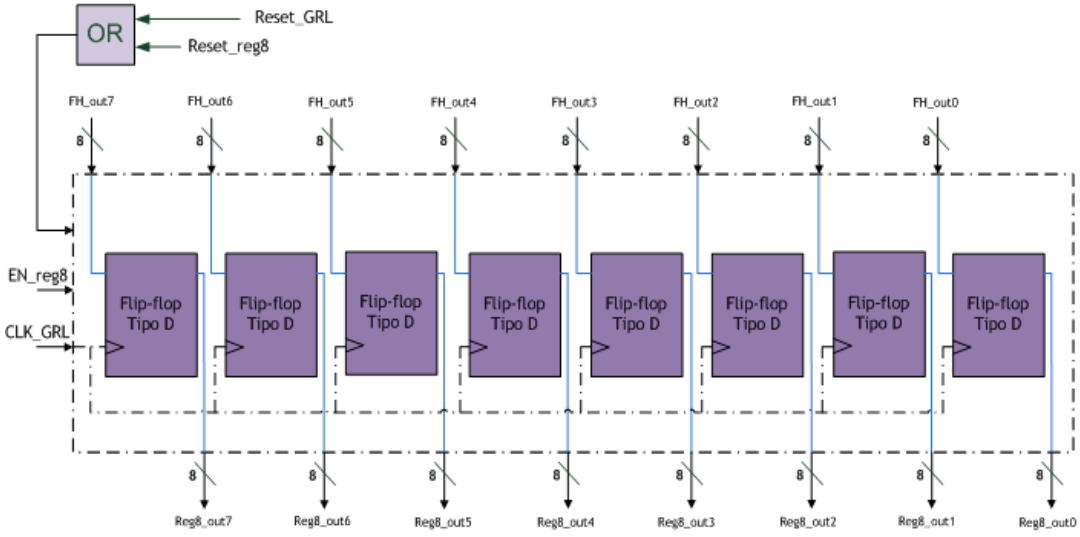


Figura N° 21. Diagrama del bloque de registro de modulo 8.

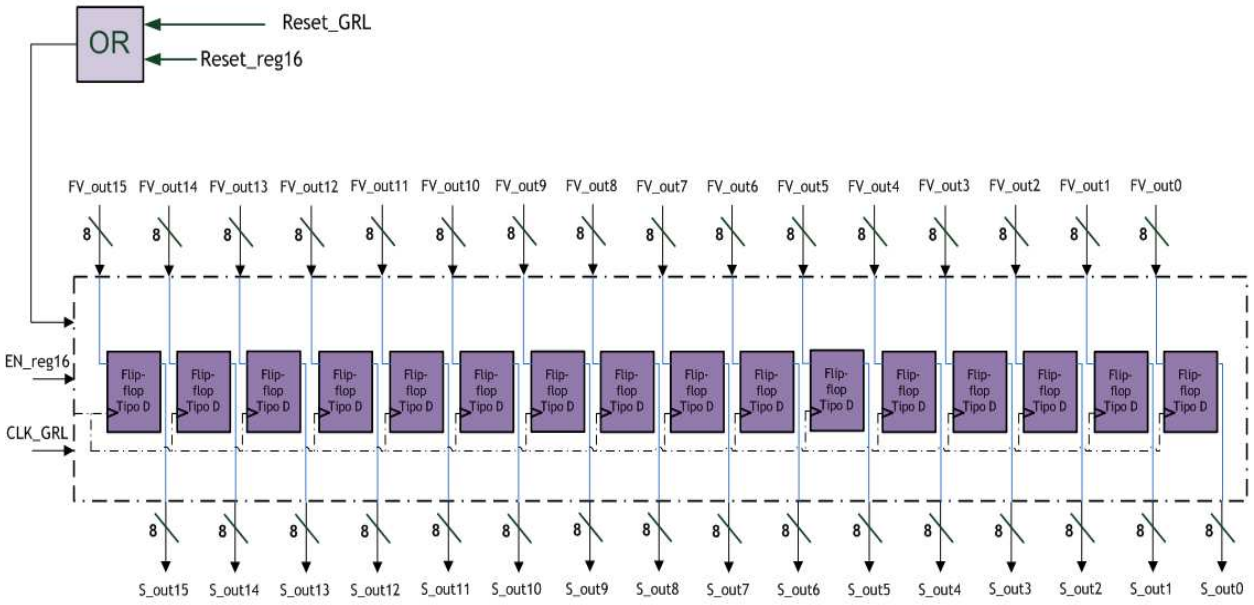


Figura N° 22. Diagrama del bloque de registro de modulo 16.

4.8.3.7. Bloque del Filtro Horizontal:

El filtrado horizontal consiste en generar un macrobloque de 4x8 pixeles, a partir de un macrobloque de 4x4 pixeles. Este bloque está conformado por 8 bloques internos, 4 bloques contienen el algoritmo del filtro de orden 4 y los otros 4 bloques son del filtro de orden 12. Este bloque recibe como entrada cada fila de 4 pixeles del los macrobloques de 4x4 para así generar un macrobloque de 4x8. El bloque de filtro horizontal realiza operaciones diferentes para el plano de luminancia y plano de cromaticidad. A continuación se mostrará a detalle cada uno.

a) Bloque de filtrado horizontal para el plano de luminancia:

Como se mencionó en el capitulo anterior, este bloque está compuesto por 4 bloques del filtro de orden 4 y otros 4 del filtro de orden 12, donde cada uno realiza la operación correspondiente con los coeficientes determinados para cada uno. (Ver Anexo B)

- Filtros para el plano de luminancia

$$SL_{12} = (1*A + 8*B + 28*C + 3*D) / 32$$

$$SL_{4} = (3*A + 24*B + 8*C + 1*D) / 32$$

Para implementar el filtro se utilizará desplazamientos de bits (sentido izquierdo para las multiplicaciones y derecho para las divisiones considerando potencias de 2) debido a que la implementación en hardware de operaciones de multiplicación y división incrementa considerablemente el consumo de recursos, lo que a su vez genera un incremento en el consumo de energía.

➔ **Filtro de orden 12**[11]**:**

En el caso de la multiplicación por el coeficiente "-1" se realizó un complemento A2 del pixel A, para el caso de multiplicación del pixel B por el coeficiente "8" se empleó tres desplazamientos de un bit hacia la izquierda. Para el caso del pixel C, lo que se realizó fue separar el coeficiente "28" de la siguiente manera:

$$S12 = (-A + 8.B + 16.C + 8.C + 4.C - 2.D - D)$$

Donde se realizó cuatro desplazamientos de 1 bit a la izquierda para el caso de la multiplicación por 16; tres desplazamientos de 1 bit a la izquierda para el caso de la multiplicación por 8; y dos desplazamientos de 1 bit a la izquierda para la multiplicación por 2. Luego para la multiplicación del pixel D por el coeficiente "-3" se realizó un desplazamiento de 1 bit a la izquierda, al resultado se le aplicó el complemento A2, con lo cual se obtuvo el pixel D multiplicado por "-2", pero como se necesita que este multiplicado por "-3", a dicho resultado se le sumó el complemento A2 del pixel D, con lo que se obtendrá el pixel D multiplicado por "-3". Finalmente, para obtener el resultado final en el intervalo de 0 a 255, es decir normalizado a 1 byte, se tiene que efectuar la división en un factor de 32, lo cual se realiza efectuando cinco desplazamientos de 1 bit a la derecha, obteniendo el resultado final normalizado a 1 byte. En caso que la operación de exceda el valor de 255, se realiza un multiplexado, realizando una previa comparación para saber si el resultado excedió 255, si se da este último, se trunca el valor a 255.

➔ **Filtro de orden 4**[12]**:**

Este filtro presenta casi los mismos coeficientes que el filtro de orden 12. Para la multiplicación del pixel A por el coeficiente "-3" se realizó un desplazamiento de 1 bit a la izquierda, al resultado se le aplicó el complemento A2, con lo cual se obtuvo el pixel A multiplicado por "-2", pero como se necesita que este multiplicado por "-3", a dicho resultado se le sumó el complemento A2 del pixel A, con lo que se obtendrá el pixel A multiplicado por "-3". Para el caso del pixel B, lo que se realizó fue separar el coeficiente "28" de la siguiente manera:

$$S4 = (-2.A - A + 16.B + 8.B + 4.B + 8.C - D)$$

Donde se realizó cuatro desplazamientos de 1 bit a la izquierda para el caso de la multiplicación por 16; tres desplazamientos de 1 bit a la izquierda para el caso de la multiplicación por 8; y dos desplazamientos de 1 bit a la izquierda para la multiplicación por 2. Para el caso de multiplicación del pixel C por el coeficiente "8" se realizó tres desplazamientos de un bit hacia la izquierda. Luego en el caso de la multiplicación del

[11] Toda la operación de manera grafica está explicada en el ANEXO B.
[12] Toda la operación de manera grafica está explicada en el ANEXO B.

pixel D por el coeficiente "-1" se realizó un complemento A2 del pixel D. Finalmente, para conseguir el resultado final en el intervalo de 0 a 255, es decir normalizado a 1 byte, se tiene que efectuar la división en un factor de 32, lo cual se realiza efectuando cinco desplazamientos de 1 bits a la derecha, con lo cual se obtiene el resultado final normalizado a 1 byte. En caso que la operación de exceda el valor de 255, se realiza un multiplexado, realizando una previa comparación para saber si el resultado excedió 255, si se da este último, se trunca el valor a 255.

b) Bloque de filtrado horizontal para el plano de Cromaticidad:

Como se mencionó en el capitulo anterior, este bloque está compuesto por 4 bloques del filtro de orden 4 y otros 4 del filtro de orden 12, donde cada uno realiza la operación correspondiente con los coeficientes determinados para cada uno.

- Filtros para el plano de cromaticidad

$SC_{12} = (0*A + 8*B + 24*C + 0*D) / 32$

$SC_{4} = (0*A + 24*B + 8*C + 0*D) / 32$

Se utilizarán desplazamientos de bits para la implementación de las multiplicaciones y divisiones del filtro horizontal para el plano de Cromaticidad siguiendo el criterio explicado en la implementación Bloque de filtrado horizontal para el plano de luminancia.

➔ **Filtro de orden 12[13]:**

En el caso de la multiplicación por el coeficiente "0" del pixel A y el pixel D sólo se descarta el valor de este pixel y continúa con el valor de cero. Para el caso de multiplicación del pixel B por el coeficiente "8" se realizó tres desplazamientos de un bit hacia la izquierda. Para el caso del pixel C, lo que se realizó fue separar el coeficiente "24" de la siguiente manera:

S12 = (0.A + 8.B + 16.C + 8.C + 0. D)

Donde se realizó cuatro desplazamientos de 1 bit a la izquierda para el caso de la multiplicación por 16; y tres desplazamientos de 1 bit a la izquierda para el caso de la multiplicación por 8. Finalmente, al igual que en el filtro horizontal de luminancia, para obtener el resultado final en el intervalo de 0 a 255, es decir normalizado a 1 byte, se tiene que efectuar la división en un factor de 32, lo cual se realiza efectuando cinco desplazamientos de 1 bits a la derecha, obteniendo el resultado final normalizado a 1 byte. En caso que la operación de exceda el valor de 255, se realiza un multiplexado, realizando una previa comparación para saber si el resultado excedió 255., si se da este último, se trunca el valor a 255.

[13] Toda la operación de manera grafica está explicada en el ANEXO C y D.

→ **Filtro de orden 4**[14]**:**

Este filtro presenta casi los mismos coeficientes que el filtro de orden 12. Para la multiplicación por el coeficiente "0" del pixel A y el pixel D sólo se descarta el valor de este pixel y continúa con el valor de cero. Para el caso del pixel B, lo que se realizó fue separar el coeficiente "24" de la siguiente manera:

$$S4 = (0.A + 8.B + 16.B + 8.C + 0.D)$$

Donde se realizó cuatro desplazamientos de 1 bit a la izquierda para el caso de la multiplicación por 16 y tres desplazamientos de 1 bit a la izquierda para el caso de la multiplicación por 8. Para el caso de multiplicación del pixel C por el coeficiente "8" se realizó tres desplazamientos de un bit hacia la izquierda. Finalmente, para conseguir el resultado final en el intervalo de 0 a 255, es decir normalizado a 1 byte, se tiene que efectuar la división en un factor de 32, lo cual se realiza efectuando cinco desplazamientos de 1 bits a la derecha, con lo cual se obtiene el resultado final normalizado a 1 byte. En caso que la operación de exceda el valor de 255, se realiza un multiplexado, realizando una previa comparación para saber si el resultado excedió 255, si se da este último, se trunca el valor a 255.

4.8.3.8. Bloque del Filtro Vertical[15]:

El filtrado horizontal consiste en generar un macrobloque de 8x8 pixeles, a partir de un macrobloque de 4x8 pixeles, generado por el bloque de filtro horizontal. Este bloque de filtrado vertical está conformado por dieciséis bloques internos, ocho bloques contienen el algoritmo del filtro de orden 4 y los otros ocho bloques son del filtro de orden 12. Este bloque recibe como entrada cada fila de 8 pixeles del los macrobloques de 4x8 para así generar un macrobloque de 8x8. El bloque de filtro vertical realiza operaciones diferentes para el plano de luminancia y plano de cromaticidad.

Tanto el filtro vertical del plano de luminancia, como el de cromaticidad están compuestos por 8 bloques de filtros de orden 12 y 8 bloques de filtros de orden 4, dichos bloques utilizan los mismos coeficientes y el mismo algoritmo que se utiliza en la etapa del filtro horizontal.

[14] Toda la operación de manera grafica está explicada en el ANEXO C y D.
[15] Toda la operación de manera grafica está explicada en el ANEXO C y D.

4.8.3.9. Maquina de estados finita (FSM):

La máquina de estados permitirá controlar el funcionamiento de la arquitectura completa de sobre muestreo de imágenes. Está conformado por cuatro estados cada uno conforma una etapa de toda la arquitectura. El estado 1, es la etapa en la que ingresan los 4 pixeles de cada fila del macrobloque de entrada de 4x4, el ingreso de dichos pixeles es controlado por el contador de pixeles. Cuando entran los 4 pixeles, el valor de la señal del contador se pone a '1', lo cual activa el pase al siguiente estado (estado 2). En el estado 2, se habilita el registro de módulo 4 para que los 4 pixeles ingresen al bloque de filtrado horizontal, se permanece en dicho estado por tres ciclos de reloj, debido a que en el diseño de la arquitectura del bloque del filtro horizontal se emplearon etapas de pipeline. Al cumplirse los tres ciclos de reloj la señal FF3_out se pone a '1' y activa el pase al siguiente estado (estado 3). En el estado 3, se habilita el registro de modulo 8 para que los 8 pixeles que salen del bloque de filtrado horizontal ingresen al bloque de filtrado vertical, se permanece en dicho estado por tres ciclos de reloj[16]. Al cumplirse los tres ciclos de reloj la señal FF2_out se pone a '1' y activa el pase al siguiente estado (estado 4). En esta última etapa se vuelve a habilitar el bloque conversor serie paralelo para que ingresen los siguientes 4 pixeles del macrobloque de 4x4. Asimismo, se habilita el registro de modulo 16 para permitir la salida del bits procesados por la arquitectura para generar el macrobloque de 8x8 de salida. Se permanece en este estado durante un ciclo de reloj para luego reiniciar con el proceso hasta que se termine de procesar todos los macrobloques de 4x4 de la imagen QVGA. En la figura N° 23 se ilustra el diagrama de estados de la máquina de estados finitos. En la Tabla N° 3 se muestran los valores de las señales de salida en cada uno de los cuatro estados, habiendo sido utilizada la codificación de tipo Moore, debido a que permite mayor estabilidad en el circuito ya que los valores de las salidas dependen del estado actual, lo que evita que estén afectadas por transitorios en las entradas.

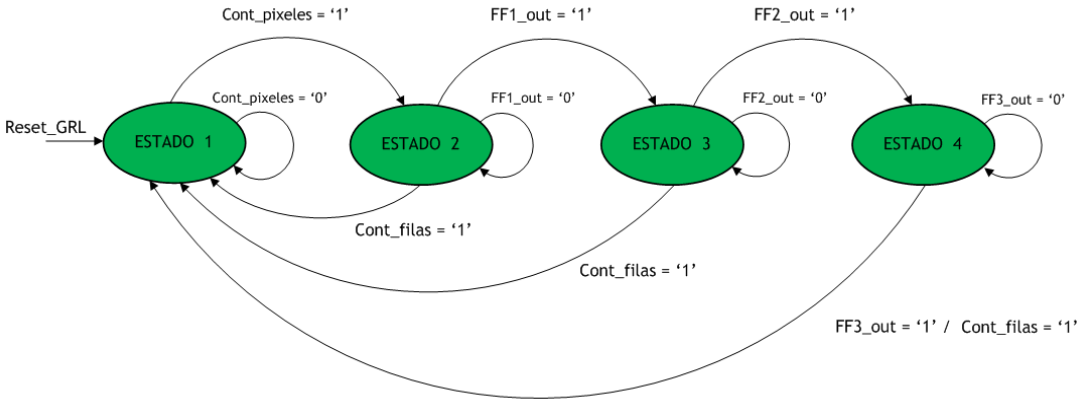


Figura N° 23.- Diagrama de estados de la máquina de estados finitos.

[16] Esta situación fue justificada en el estado 2.

Tabla N° 3.- Tabla de control de la arquitectura del filtro de sobre muestreo.

CONTROLADOR PRINCIPAL DE LA ARQUITECTURA DEL FILTRO																
ESTADO	EN_reg4	Reset_reg4	EN_reg8	Reset_reg8	EN_reg16	Reset_reg16	EN_conversorSP	Reset_conversorSP	RESET_FF1	RESET_FF2	RESET_FF3	EN_FF1	EN_FF2	EN_FF3	Reset_cont_pixel	Reset_cont_filas
UNO	0	0	0	0	0	0	1	0	0	0	1	1	1	1	0	0
DOS	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
TRES	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1	0
CUATRO	0	0	1	0	1	0	1	0	1	1	0	1	1	1	0	0

El control total de la arquitectura[17] se realiza básicamente como se muestra en la figura N° 24, de manera conjunta para los planos de luminancia (Y) y cromaticidad (Cb y Cr), donde los datos(pixeles), de cada macrobloque de 4x4, ingresan de manera serial a cada una de las arquitecturas de los planos y se obtienen, de manera paralela los datos procesados que forman los macrobloques de 8x8, con los cuales se forma la imagen sobre muestreada que presenta una resolución VGA. El sistema está controlado de manera similar para la arquitectura en el plano de luminancia y los planos de cromaticidad. Por ello, es que se obtienen los datos procesados al mismo tiempo.

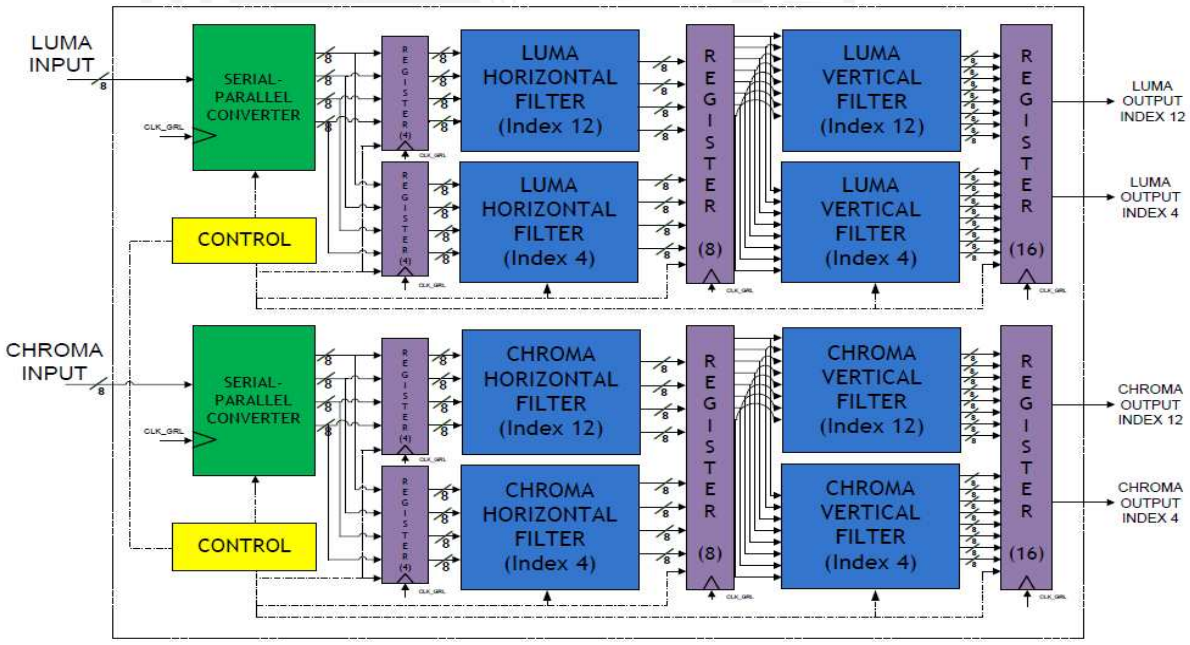


Figura N° 24.- Diagrama completo de la arquitectura del filtro de sobremuestreo.

[17] Toda la operación de manera grafica está explicada en el ANEXO B, C y D.

CAPITULO 5

RESULTADOS

5.1. Simulación de los módulos de la arquitectura diseñada:

Para poder verificar cada una de las etapas y bloques de la arquitectura se tomará como referencia la imagen “perrito.jpg” con una resolución QVGA (320x240). Los resultados de la simulación de cada plano (Luminancia Y, Cromaticidad Cb y Cr) se adjuntan en el Anexo E.

Asimismo se adjuntan los archivos del proyecto de la arquitectura, obtenidos del software Quartus II v.9.2 de la compañía ALTERA® (Ver Anexo F), junto con los resultados obtenidos por el software de simulación ModelSim v6.5b, utilizando simulación por Testbench, los cuales se podrá revisar en el Anexo E. Los nombres de los archivos obtenidos por Testbench son “Macroblokones_Y.txt” para el resultado de sobremuestreo del plano de luminancia de la imagen “perrito.jpg”; “Macroblokones_Cb.txt” y “Macroblokones_Cr.txt” como resultado de procesar los planos de cromaticidad azul y rojo. Dichos archivos de texto contienen una matriz de números, los cuales conforman la imagen sobre muestreada de cada plano. Estas matrices serán comparadas con las matrices resultantes de la imagen mediante la implementación en software.

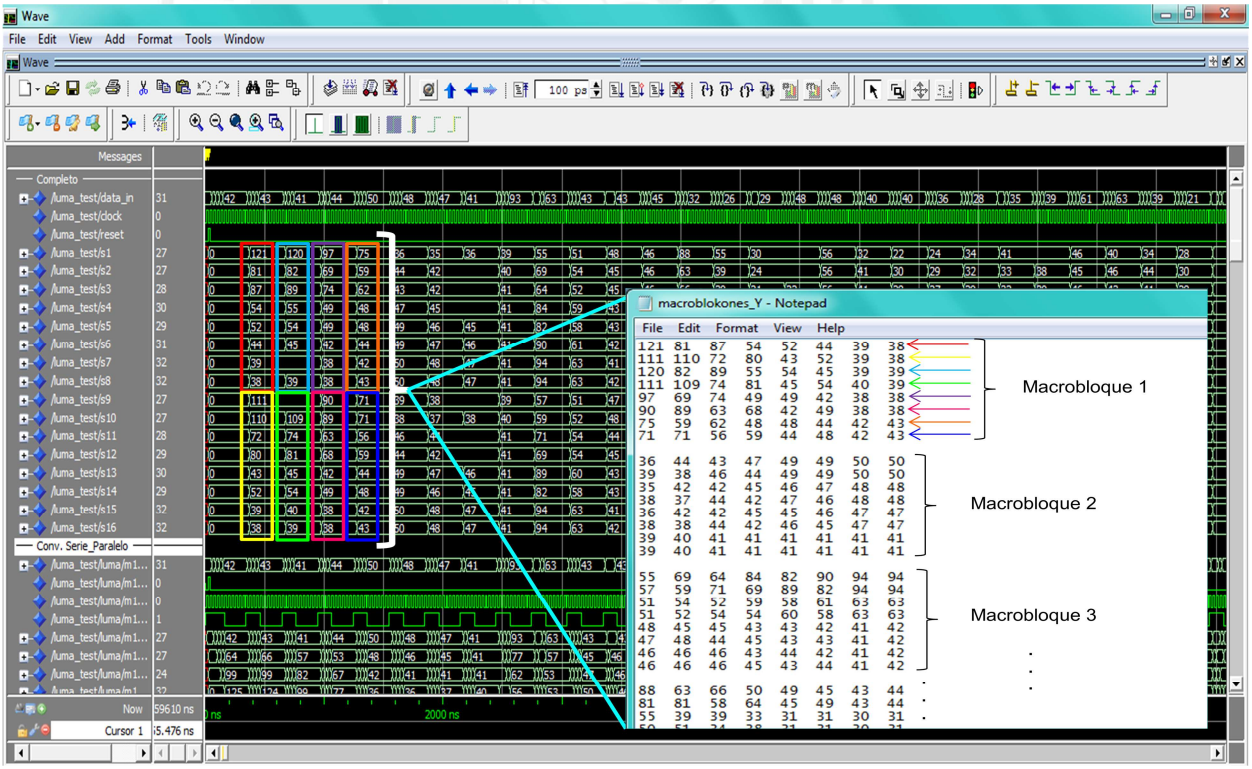


Figura N° 25.- Simulación de resultados obtenidos por el software ModelSim por medio de TestBench.

En la figura N° 25 se observa los resultados de la aplicación del algoritmo en la arquitectura diseñada para el plano de luminancia, dicho valores se muestran en la ventana de simulación. Asimismo, tales valores son grabados en un archivo de texto formando los macrobloques de 8x8, tal como se muestra en la imagen. De la misma forma se obtienen los resultados para los planos de cromaticidad rojo y azul. Luego de obtener dichos archivos de texto con los macrobloques de 8x8 se procede a realizar la comparación con la matriz obtenida por el software. En las figuras 26, 27 y 28 se puede observar las comparaciones, entre los resultados obtenidos por software y los obtenidos por hardware, respectivas para cada plano de la imagen, siendo iguales.

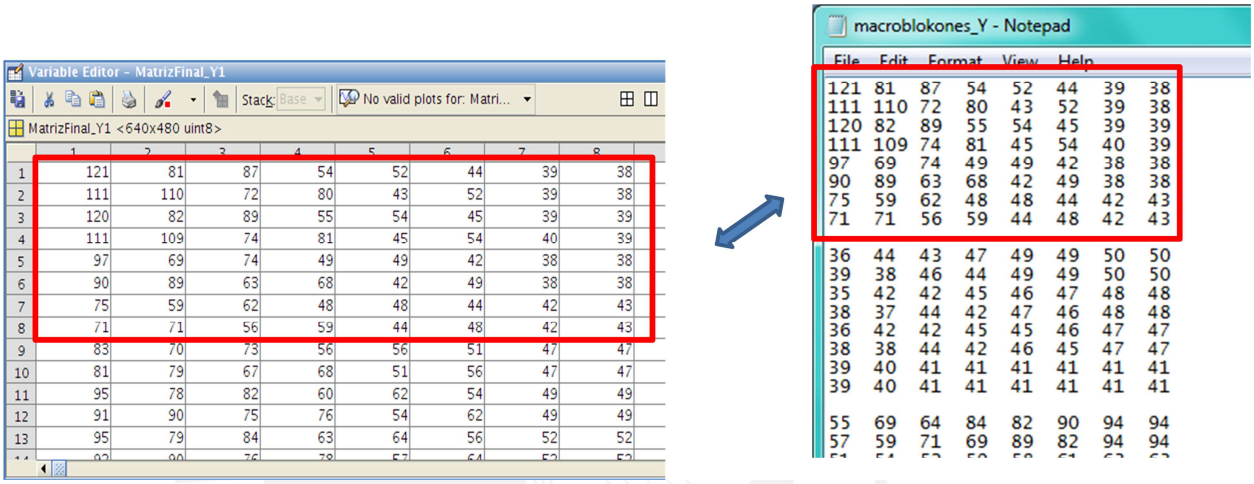


Figura N° 26. Comparación de valores entre el obtenido por software y hardware para el plano de luminancia Y.

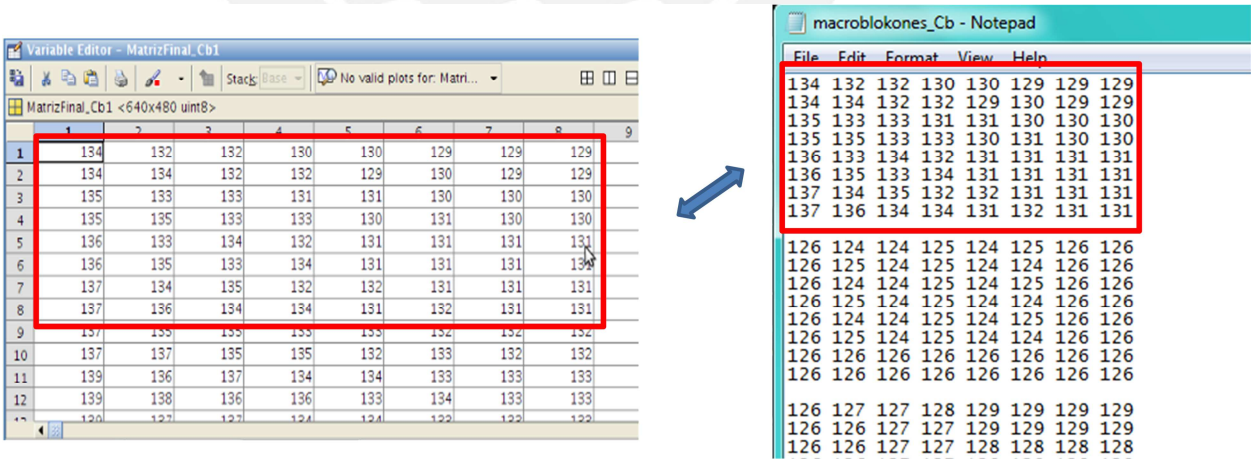


Figura N° 27. Comparación de valores entre el obtenido por software y hardware para el plano de cromaticidad azul Cb.

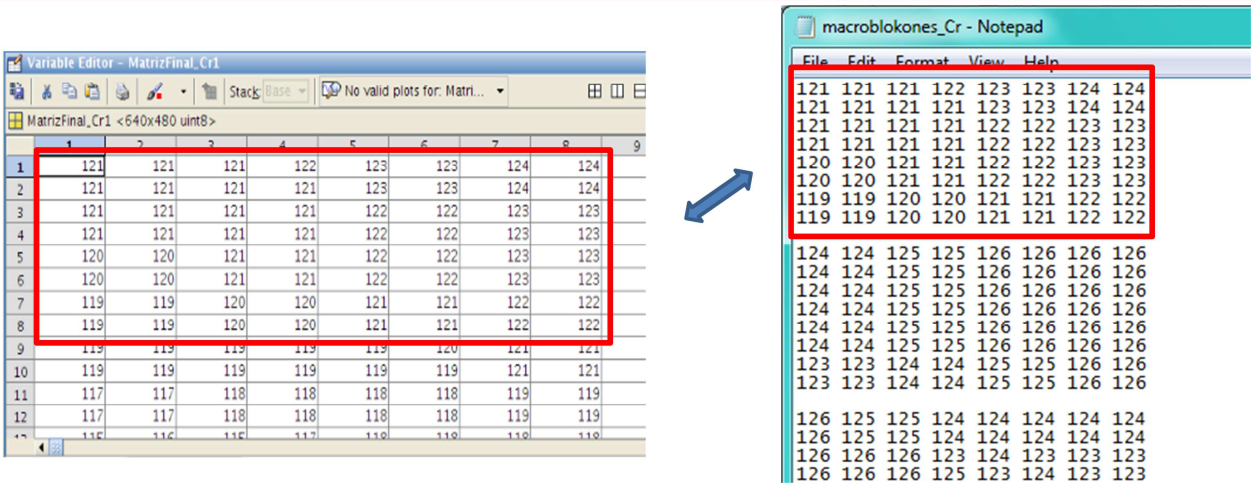


Figura N° 28. Comparación de valores entre el obtenido por software y hardware para el plano de cromaticidad rojo Cr.

De los resultados obtenidos, se observó que la arquitectura que procesa el plano de luminancia emplea 192013 ciclos de reloj en procesar cada uno de los planos[18] Y, Cb y Cr. Se concluye que mientras la frecuencia de operación de la arquitectura sea mayor se podrán procesar mayor cantidad de cuadros por segundo.

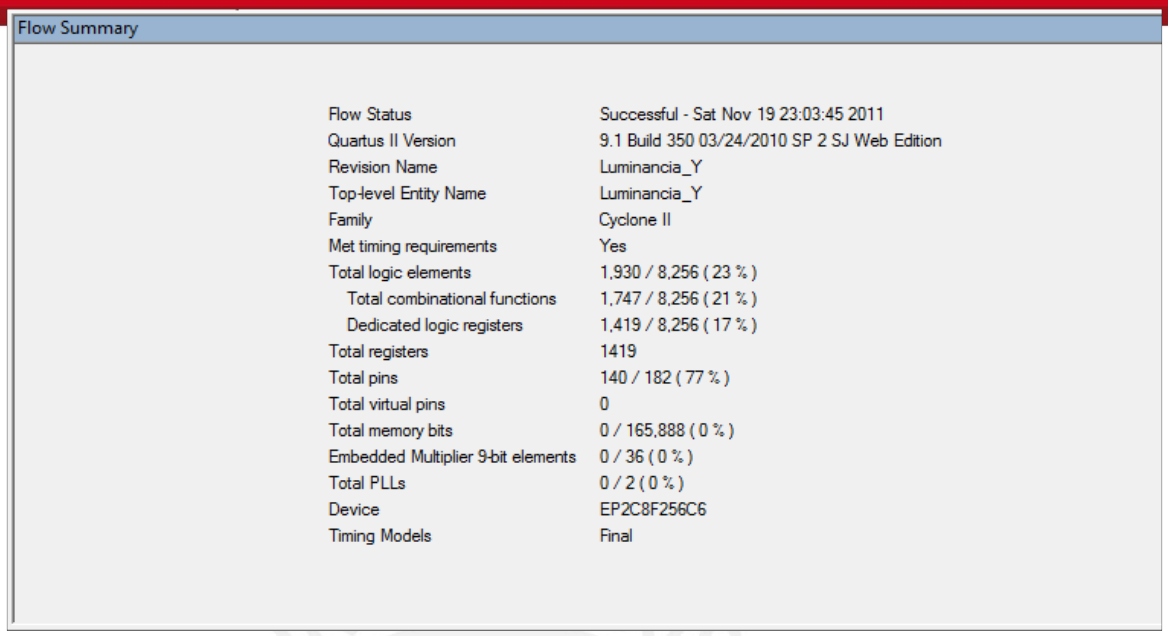
5.2. Resultados de la síntesis de la arquitectura diseñada:

La arquitectura diseñada del filtro de sobremuestreo fue sintetizada utilizando el software Quartus II v9.2 para el FPGA Cyclone II EP2C35F6F2C6 de la compañía ALTERA® [12]. Los resultados obtenidos serán visualizados a continuación, dado que se ha desarrollado una arquitectura de manera independiente para procesar cada plano por separado.

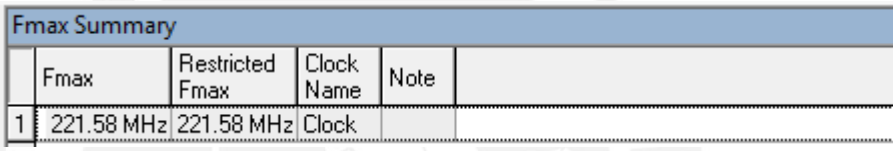
i).- Síntesis de la arquitectura del plano de luminancia (Y)

En la figura N° 29 se muestra el resultado de la síntesis realizada a la arquitectura del plano de luminancia Y.

[18] Los cuales presentan una resolución de 320x240 pixeles.



(a)



(b)

Figura N° 29. (a) Resultado de la síntesis de la arquitectura del plano de luminancia. (b)Frecuencia máxima de operación.

Se puede observar que se obtuvo una frecuencia de operación de la arquitectura, considerablemente mayor respecto al obtenida en la referencia [4], gracias a las etapas de pipeline que insertaron en algunos bloques de la arquitectura, las cuales fueron analizadas con mucho criterio para hacer la inserción y al eficiente aprovechamiento del paralelismo tomado en cuenta al realizar el diseño de la arquitectura del filtro.

ii).- Síntesis de la arquitectura del plano de cromaticidad (Cb y Cr)

La diferencia de esta arquitectura respecto a la realizada para el plano de luminancia se sitúa en el algoritmo del filtro, debido a los coeficientes que presenta son distintos a los utilizados en el plano de luminancia. En la figura N° 30, se muestra el resultado de la síntesis de la arquitectura del plano de cromaticidad Cb y en la figura N° 31 del plano de cromaticidad Cr.

Flow Summary

Flow Status	Successful - Sun Nov 20 08:07:33 2011
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	Cromaticidad_Cb
Top-level Entity Name	Cromaticidad_Cb
Family	Cyclone II
Met timing requirements	Yes
Total logic elements	673 / 8,256 (8 %)
Total combinational functions	516 / 8,256 (6 %)
Dedicated logic registers	644 / 8,256 (8 %)
Total registers	644
Total pins	140 / 182 (77 %)
Total virtual pins	0
Total memory bits	0 / 165,888 (0 %)
Embedded Multiplier 9-bit elements	0 / 36 (0 %)
Total PLLs	0 / 2 (0 %)
Device	EP2C8F256C6
Timing Models	Final

(a)

Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	303.86 MHz	303.86 MHz	Clock	

(b)

Figura N° 30.- (a) Resultado de la síntesis de la arquitectura del plano de cromaticidad Cb. (b)Frecuencia máxima de operación.

Flow Summary

Flow Status	Successful - Sun Nov 20 08:15:01 2011
Quartus II Version	9.1 Build 350 03/24/2010 SP 2 SJ Web Edition
Revision Name	Cromaticidad_Cr
Top-level Entity Name	Cromaticidad_Cr
Family	Cyclone II
Met timing requirements	Yes
Total logic elements	673 / 8,256 (8 %)
Total combinational functions	516 / 8,256 (6 %)
Dedicated logic registers	644 / 8,256 (8 %)
Total registers	644
Total pins	140 / 182 (77 %)
Total virtual pins	0
Total memory bits	0 / 165,888 (0 %)
Embedded Multiplier 9-bit elements	0 / 36 (0 %)
Total PLLs	0 / 2 (0 %)
Device	EP2C8F256C6
Timing Models	Final

(a)

Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	303.86 MHz	303.86 MHz	Clock	

(b)

Figura N° 31.- (a) Resultado de la síntesis de la arquitectura del plano de cromaticidad Cr. (b)Frecuencia máxima de operación.

Al operar simultáneamente las arquitecturas del los tres planos (Y, Cb y Cr) la frecuencia de operación máxima estará regida por la menor frecuencia de las tres arquitecturas, en este caso será la frecuencia de 221.58 MHz. Luego a partir de este resultado obtenido se puede deducir si la arquitectura es capaz de procesar secuencias QVGA con el requerimiento de 30 cuadros por segundo. Considerando las variaciones de proceso en la implementación de la arquitectura diseñada en el FPGA se tomará un valor de 90% de la frecuencia máxima de operación para determinar la tasa de procesamiento de secuencias de video por parte de la arquitectura. Tomado los datos mostrados a continuación se podrá hacer el cálculo correspondiente.

- La arquitectura tarda 192013 ciclos de reloj en procesar una imagen QVGA

- Frecuencia de Reloj Máxima: (221.58 MHz)* 90% = 199.42 MHz

(192013 Ciclos de reloj/Cuadro QVGA)*(1/199.42MHz) = 9.62 mseg/Cuadro QVGA

(1/9.62 mseg/Cuadro QVGA) = 1039 cuadros/seg (fps)

De este resultado se puede concluir que se está cumpliendo con el objetivo principal del presente trabajo de tesis: diseñar una arquitectura hardware del algoritmo de sobremuestreo de imágenes para secuencias QVGA que opere a una tasa mayor o igual a 30 cuadros por segundo.

5.3. Comparación con un trabajo previo de la bibliografía.

Para hacer la comparación en las mismas condiciones con el trabajo previo realizado y que se explica en la referencia [4], hay que aclarar que dicho trabajo fue sintetizado sobre un FPGA Stratix IV EP45SGX530HH35C3 de ALTERA®, el cual presenta una tecnología de 40 nm. A comparación del presente trabajo que fue sintetizado sobre un FPGA cyclone II de ALTERA® que presenta una tecnología de 90 nm. Por lo tanto, la tecnología utilizada por la referencia [4] es mucho más moderna que la utilizada por el presente trabajo. A pesar de ello, la frecuencia máxima de operación obtenida por dicho trabajo es menor a la obtenida por el presente trabajo. En la referencia [4] se obtuvo una tasa de procesamiento de 384 cuadros por segundo, en cambio, en este trabajo se llego a obtener una tasa de procesamiento de 1039 cuadros por segundo, a pesar de que se utilizó un FPGA de una tecnología más antigua. La tecnología utilizada por la referencia [4] es un dispositivo FPGA Stratix IV, el cual presenta una tecnología de 40 nm [17], superando la tecnología de 90 nm del FPGA Cyclone II [11]. El hecho de ser una tecnología más moderna, lo hace más pequeña comparado con tecnologías

anteriores, entonces si se tiene un menor tamaño, las distancias de ruteamiento de los circuitos son menores en una Stratix IV que en una Cyclone II. Por lo tanto, los retardos son menores lo que implica que la frecuencia de operación debe de ser mayor en un dispositivo de 40nm respecto a una tecnología de 90 nm. El diseñar una arquitectura sobre un FPGA permite validar rápidamente un diseño, con lo que se consigue un "*time to market*" mucho menor permitiendo llegar al mercado rápidamente. El validar un diseño sobre FPGA puede presentar dos etapas siguientes de implementación, la primera de ellas es continuar con el diseño sobre un FPGA, pero realizando la validación para poder operar con reconfiguración dinámica[19] [18]. La segunda es implementar la arquitectura diseñada sobre un dispositivo ASIC, el cual presenta algunas desventajas[20] si lo que se desea es llevar un dispositivo al mercado.

En la Tabla N° 4 se muestra un cuadro comparativo entre el presente trabajo y el trabajo realizado en la referencia [4]. Donde los resultados obtenidos son favorables respecto a los obtenidos por el trabajo previo. Según lo justificado en el párrafo anterior, la tecnología de 90 nm presenta mayores retrasos en la transmisión de señales comparada con una tecnología de 40 nm utilizada en la referencia [4]. Además, cabe mencionar que al revisar el trabajo realizado en la referencia [7] se pudo verificar que existe un error en la utilización de los coeficientes del filtro de orden 12 y orden 4 para el plano de cromaticidad. Los coeficientes utilizados en el plano de cromaticidad de orden 4 y orden 12 deben ser como el que se ha utilizado en el presente trabajo. En la referencia [7] utilizan los coeficientes de cromaticidad de la siguiente manera:

$$SC_{12} = (8*A + 24*B + 0*C + 0*D) / 32$$

$$SC_{4} = (24*A + 8*B + 0*C + 0*D) / 32$$

Aplicando estos coeficientes se obtiene una imagen sobremuestreada con una grilla azul, lo cual fue verificado al implementarlo en software. En cambio, si se usa los coeficientes como se menciona en el capítulo 2 se realiza un correcto sobremuestreo de la imagen.

[19] Flexibilidad para cambiar el diseño de una arquitectura sin necesidad de reiniciar o reconfigurar completamente el dispositivo. El hecho de auto-reconfigurar la arquitectura para que utilice sólo los componentes necesarios para una aplicación en un momento dado, hace que se optimice el sistema y pueda conseguir un ahorro considerable de energía [10] [19].

[20] En el capítulo 2 se explica las desventajas de un dispositivo ASIC respecto a un FPGA en lo que refiere *time to market.* Si la implementación sobre dispositivos ASIC es para una producción a corta escala esta se vuelve muy cara, comparado con lo que sería realizarlo sobre un FPGA [19].

Tabla N° 4.

	Dr. Luciano Volcan Agostini [1]	El presente trabajo de tesis
Frecuencia máxima obtenida (Mhz)	119.5 Mhz	**221.58 Mhz**
Número de ciclos de reloj por cuadro	311040 ciclos	**192013 ciclos**
Resolución	QVGA → VGA	**QVGA → VGA**
Dispositivo FPGA	Stratix IV de Altera (40 nm)	**Cyclone II de Altera (90 nm)**
Tasa máxima de cuadros por segundo (fps)	384 cuadros	**1039 cuadros**

La utilización de macrobloques genera una ligera pérdida de resolución de la imagen al sobre muestrearla a dicho problema se lo conoce como el *"Efecto bloque"*, esto se genera debido a la partición de la imagen en macrobloques, ya sea de 4x4, 4x8 o 8x8 según lo que menciona el estándar. Para tratar de corregir es que se han realizado trabajos de investigación como el que figura en la referencia [20] relacionado a un filtro, tipo adaptativo, llamado *"deblocking filter"* o "filtro desbloqueante". La técnica de filtrado consiste en determinar automáticamente la cantidad de energía del bloque así como su distribución y preservando al máximo los detalles. Se realiza una transformación para suavizar las discontinuidades entre los bloques vecinos. De esta manera, los filtros procesaran las secuencias de imágenes o video consiguiendo un mejor efecto de suavizado [20]. Con este filtro desbloqueante se procesa la imagen tratando de corregir la degradación de la resolución producida por el sobre muestreo, para que así la resolución de la imagen sobre muestreada tenga una calidad aproximada a la imagen original.

CONCLUSIONES

- El aprovechar de manera óptima la característica del alto grado de paralelismo de operaciones al realizar el diseño de la arquitectura y la inserción adecuada de etapas de pipeline, permitieron diseñar una arquitectura hardware que alcanzó una frecuencia máxima de operación de 221.58 MHz para el dispositivo FPGA Cyclone II de la compañía ALTERA®. Por temas de seguridad en el funcionamiento de la arquitectura ante posibles variaciones del proceso al momento de realizar una futura implementación se restringe a un 90% la frecuencia máxima de operación, lo cual permite procesar una secuencia de imágenes QVGA a una tasa de 1036 cuadros por segundo, con lo cual se cumple objetivo principal del trabajo de tesis, de operar a una tasa mayor o igual a 30 cuadros por segundo para cumplir con el requerimiento de operación en tiempo real de video.

- La validación del algoritmo del filtro de sobre muestreo sobre el entorno de programación MATLAB® permitió verificar que los resultados obtenidos por medio de la arquitectura hardware diseñada son válidos, cumpliendo con el objetivo de diseñar correctamente la arquitectura cumpliendo los requerimientos planteados.

- El desarrollo de equipos basados en FPGA’s para un CODEC bajo el estándar H.264/SVC con la finalidad de procesar secuencias de imágenes en tiempo real es perfectamente factible, como se ha demostrado. Asimismo, no es necesario utilizar dispositivos de última generación del mercado para poder realizar diseños eficientes. Por lo que, realizando un eficiente diseño aprovechando con criterio la característica de paralelismo de operaciones y el correcto uso de etapas de pipeline se pueden obtener altas frecuencias de operación.

- La máxima frecuencia obtenida por un único trabajo anterior relacionado con el presente trabajo de tesis obtuvo una frecuencia máxima de operación de 119.5 MHz, mientras que el presente obtuvo una de 221.58MHz, es decir, se obtuvo una mayor frecuencia de operación. Por ello, se puede afirmar que el diseño de la arquitectura realizada es más eficiente que el trabajo previo realizado en la referencia [4] en términos de tasa de procesamiento de cuadros de video.

RECOMENDACIONES

- El estándar H.264 propone el uso de macrobloques para evitar complicaciones al realizar el procesamiento, dado que una imagen no es estacionaria [20]. Por ello, es que se divide en pequeños macrobloques, ya sea de 4x4, 4x8 o 8x8. Estos macrobloques si presentan un comportamiento estacionario, sin embargo, la utilización de macrobloques genera una consecuencia directa a efecto visual y es que se pierde resolución debido al uso de macrobloques, a esto se llama "el efecto bloque". Por esto se recomienda utilizar, después del filtro de sobremuestreo, un filtro desbloqueante que tiene la finalidad de reducir el efecto bloque, controlando el ancho del filtro y sin afectar la nitidez de la imagen [20].

- El diseño planteado en el presente trabajo puede implementarse en un dispositivo FPGA de la compañía ALTERA® o XILINX® para validar y verificar el funcionamiento de toda la arquitectura desarrollada ante variaciones. Para lo cual se utilizan herramientas software, conocidos como analizadores lógicos como es el caso del Signal Tap [21] de la compañía ALTERA® y del ChipScope [22] para el caso de la compañía XILINX®.

- El uso de metodología de diseño y verificación empleado a nivel industrial es lo más recomendable cuando se hace un diseño en hardware. A nivel industrial se conoce como DUV (Design Under Verification), por ello se sugiere aplicar esta metodología en futuras mejoras o trabajos a nivel de diseño en hardware. En este trabajo se realizóo la verificación funcional por Testbench, siendo esta un forma particular de la metodología DUV.

- Se propone realizar el diseño de los demás módulos hardware que conforman la arquitectura de un decodificador o un codificador según el formato H.264/SVC planteado en el capítulo 1, así como el filtro desbloqueante, para que a futuro se pueda realizar la implementación de un prototipo sobre un FPGA, para luego, mediante la utilización de la herramienta de diseño CADENCE bajo una tecnología de 90 nm o más reciente, se pueda realizar un ASIC. También se considera que el prototipo debe ser analizado en términos de etapas de operación para evaluar la factibilidad de emplear la técnica de reconfiguración dinámica para implementarlo en un FPGA que admita este tipo de operación, teniendo como gran finalidad su utilización en equipos portátiles ya que como justificado en la referencia [10] esta técnica permite obtener implementaciones de bajo consumo de energía.

BIBLIOGRAFIA

[1]. MAXFIELD, C. The Design Warriors Guide to FPGAs. Burlington: Elsevier, 2004. 542 p.

[2]. GONZALEZ; WOODS; EDDINS Digital Image Processing Using Matlab. 2004. p.

[3]. SUN, SHIJUN Methods and systems for Upsampling Filter Design – Patent Aplication Publication Pub. N° US 2007/31065 A1

[4]. SILVA, T.; CRUZ, L.; AGOSTINI, L. A Novel Macroblock-Level Filtering Upsampling Architecture for H.264/AVC Scalable Extension. Brazilian Symposium on Integrated Circuits and Systems Design (SBCCI), 2010, p. 163-167.

[5]. www.itu.int Ultima revisión 18/05/11

[6]. SEGALL, A.; SULLIVAN, G. Spatial Scalability Within the H.264/AVC Scalable Video Coding Extension. In: IEEE Transactions on Circuits and Systems for Video Technology, v. 17, n. 9, pp 1121-1135, 2007.

[7]. DA SILVA, THAÍSA LEAL; VOLCAN AGOSTINI, LUCIANO Efficient Hardware Design for the Upsampling in the H.264/SVC Scalable Video Coding Extension

[8]. Stephen Brown, Zvonko Vranesic 2005 “Fundamentals of Digital Logic with VHDL Design”, McGraw-Hill

[9]. www.nvidia.com / Unidad de procesamiento grafico.

[10]. LLAMOCA D.; CARRANZA C.; PATTICHIS M. “Separable FIR filtering in FPGA and GPU implementations: Energy, Performance, and accuracy considerations” in Proceedings of the 2011 International Conference on Field Programmable Logic and Applications (FPL'11), Chania, Greece, September 2011.

[11]. Altera’s Corporation 2005 “ Cyclone II Device Handbook ”, Volumen 1

[12]. ALTERA Quartus II Development Software Handbook v. 10.0, 2010

[13]. Using ModelSim to Simulate Logic Circuits. Disponible en ftp://ftp.altera.com/up/pub/Altera_Material/10.1/Tutorials/Using_ModelSim.pdf>.

[14]. SUDHAKAR YALAMANCHILI. “VHDL starter’s guide”.

[15]. Daniel D. GAJSKI; Robert H. KUHN. “Guest Editor’s Introduction New VLSI Tools”. Cap. 1, 1983.

[16]. GONZALEZ; WOODS Tratamiento digital de imagenes. 1996. p.

[17]. Altera’s Corporation 2010 “ Stratix IV Device Handbook ”, Volumen 1

[18]. VAHID, F. Embedded System Design: A Unified Hardware/Software Introduction. New York: John Wiley & Sons, 2001. 324 p.

[19]. RAFFO, M. Desenvolvimento de um sistema dinamicamente reconfigurável baseado em redes intra-chip e ferramenta para posicionamento de módulos. 2010. 107 p. Dissertação (Mestrado) - Escola Politécnica, Universidade de São Paulo, São Paulo, 2010.

[20]. CORREA, G.; CRUZ, L.; AGOSTINI, L. Filtro redutor de efeito de bloco entre camadas do padrão H.264/AVC Escalavel. XV Workshop IBERCHIP 2009 , pp 561 - 566.

[21]. Signal Tap II Logic Analyzer. Disponible en <ftp://ftp.altera.com/up/pub/Altera_Material/10.1/Tutorials/VHDL/SignalTap.pdf>.

[22]. ChipScope Tool de Xilinx. Disponible en <http://www.xilinx.com/tools/cspro.htm>.