DEPÓSITO LEGAL ZU2020000153
ISSN 0041-8811
E-ISSN 2665-0428

Revista de la Universidad del Zulia

Fundada en 1947
por el Dr. Jesús Enrique Lossada

Ciencias del Agro Ingeniería y Tecnología

Año 12 N° 32
Enero - Abril 2021
Tercera Época
Maracaibo-Venezuela

DOI: http://dx.doi.org/10.46925//rdluz.32.22

Evaluación del rendimiento de las arquitecturas de hardware HPS y HPS+FPGA para un sistema de procesamiento de imágenes

Cesar Arturo Niño Carmonal *
Manuel-Jesús Sánchez-Chero **
Emanuel Ortiz Ortiz ***
Juan Carlos Sernaque Julca ****
Cecilia Lizeth Risco Ipanaqué *****

RESUMEN

El objetivo de este trabajo fue evaluar el rendimiento de las arquitecturas de hardware: Hard Processor System (HPS) y la unión de un HPS con una matriz de compuertas programables o FPGA (HPS + FPGA) para un sistema de procesamiento de imágenes. Se evalúan: el tiempo de ejecución de los algoritmos de procesamiento de imágenes y el consumo de energía. Para una Plataforma SoC se realiza el diseño de hardware en Verilog utilizando los núcleos de video IP del University Program (UP) de Intel - FPGA. Se desarrolla también el software para control y visualización de resultados empleando OpenCV. Se trabajó con imágenes de 320x240 pixeles. Para una aplicación en tiempo real se observó una mejora de 38.8% en el tiempo de ejecución y un consumo 6.85% mayor en la Arquitectura HPS+FPGA respecto a la Arquitectura HPS. La Arquitectura HPS+FPGA supera al HPS y mantiene bajo el consumo de energía.

PALABRAS CLAVE: Algoritmos de procesamiento, Arquitecturas de hardware, Plataforma SoC, Rendimiento, Procesamiento de imágenes.

* Docente Asociado. Universidad Nacional de Piura. Perú. https://orcid.org/0000-0002-0981-0822

** Docente Investigador. Universidad Nacional de Frontera. Perú. https://orcid.org/0000-0003-1646-3037. E-mail: manuelsanchezchero@gmail.com

*** Bachiller. Universidad Nacional de Piura. Perú. https://orcid.org/0000-0002-4222-7372

**** Bachiller. Universidad Nacional de Piura. Perú. https://orcid.org/0000-0002-3157-8935

***** Jefa (E) de la Unidad de Tecnología de Información y Comunicación. Universidad Nacional de Frontera. Perú. https://orcid.org/0000-0002-7936-1495

Recibido: 03/09/2020 Aceptado: 10/11/2020

DOI: http://dx.doi.org/10.46925//rdluz.32.22

Performance evaluation of HPS and HPS+FPGA hardware architectures for an image processing system

ABSTRACT

The objective of this work was to evaluate the performance of hardware architectures: Hard Processor System (HPS) and the union of an HPS with a programmable gate array or FPGA (HPS + FPGA) for an image processing system. The following are evaluated: the execution time of the image processing algorithms and the energy consumption. For a SoC Platform, hardware design is performed at Verilog using the IP video cores of the Intel University Program (UP) - FPGA. The software for control and visualization of results using OpenCV is also developed. We worked with 320x240 pixels images. For a real time application it was observed an improvement of 38.8% in the execution time and a 6.85% higher consumption in the HPS+FPGA Architecture with respect to the HPS Architecture. The HPS+FPGA Architecture outperforms HPS and keeps power consumption low.

KEYWORDS: Processing Algorithms, Hardware Architectures, SoC Platform, Performance, Image Processing

Introducción

La implementación de visión artificial en sistemas embebidos requiere de hardware cada vez más eficiente en términos de capacidades computacionales y consumo de energía. Para la implementación de estos sistemas se vienen utilizando plataformas SoC con algunos problemas de rendimiento como el observado en el trabajo de Espinoza (2016), quien implementó clasificadores HAAR en una laptop donde no hubo retardo; mientras que en un Raspberry PI el retardo fue de 1000 a 2000 milisegundos. Para mejorar esto, se recurre a métodos de aceleración mediante hardware; como por ejemplo, mediante los FPGA, ya que proporcionan una mayor eficiencia energética que las GPU y las CPU y un mayor rendimiento que las CPU como menciona Mittal y Vetter (2015).

Para desarrollar el presente trabajo se utilizó una Plataforma SoC (System on chip, Sistema en un chip) basada en un microcontrolador ARM (Advanced Risc Machine), y FPGA (Field Programmable Gate Array, arreglo de compuertas programables) Terasic Inc. (2018). El término SoC en el nombre de la plataforma es debido a que la tarjeta cuenta con un chip central conformado por un ARM, una FPGA, módulos de comunicación, controladores de memoria,

DOI: http://dx.doi.org/10.46925//rdluz.32.22

entre otros; de tal manera que juntos conforman casi un computador completo integrado en un solo chip, siendo necesario conectar algunos pocos periféricos fuera de la tarjeta para funcionar como una minicomputadora. El objetivo de la investigación fue implementar un sistema de procesamiento de imágenes empleando una arquitectura basada en un microcontrolador ARM y una FPGA, a la cual llamaremos en este trabajo como Arquitectura HPS+FPGA, con la finalidad de mejorar el rendimiento en términos de tiempo de ejecución de los algoritmos de procesamiento de imágenes y consumo de energía, con respecto a una arquitectura basada solo en ARM (Arquitectura HPS).

1. Materiales y métodos

En esta investigación se realizó el diseño, configuración e implementación de hardware y software para procesamiento de imágenes en una plataforma SoC de Altera utilizando Quartus como herramienta de diseño de hardware y el lenguaje Embedded C para el desarrollo del software. El diseño en el FPGA se basó en los núcleos del University Program de Intel-FPGA y algunos núcleos de diseño propio, mientras que para el desarrollo en Software se utilizó OpenCV. Se trabajó con imágenes de 320x240 pixeles. Adicionalmente se utilizó un dispositivo de medición en tiempo real de la potencia consumida por la tarjeta SoC DE10Nano como requisito para realizar el análisis del consumo de energía.

El procedimiento para hacer la implementación de la Arquitectura HPS+FPGA en una plataforma SoC basada en FPGA, fue el siguiente:

1. Definición de las consideraciones o criterios previos al diseño de hardware.
2. Diseño e interconexión de núcleos IP para la plataforma SoC DE10-Nano.
3. Desarrollo de software de control y procesamiento de imágenes.
4. Evaluación del rendimiento en función del tiempo de ejecución de algoritmos de procesamiento de imágenes
5. Evaluación del rendimiento en función del consumo de energía.

DOI: http://dx.doi.org/10.46925//rdluz.32.22

Se utilizó la técnica de recolección de datos donde se obtuvo, el tiempo de ejecución de los algoritmos de procesamiento de imágenes tanto en la Arquitectura HPS así como en la Arquitectura HPS+FPGA. La instrumentación de recolección de datos fue mediante fichas o guías de observación.

1.1. Consideraciones previas al diseño de hardware

Se han tenido en cuenta las siguientes consideraciones para el diseño de hardware:

- El diseño debe permitir futuras modificaciones por lo que es importante conservar recursos de memoria y compuertas lógicas para posibles implementaciones. Adicionalmente, esto permite conservar bajo el consumo de energía.
- Los IP's utilizados para procesar imágenes deben ser de uso libre, por lo que se emplearon los IP's del University Program de Intel-FPGA en su mayoría.
- De los diferentes tipos de memoria disponibles en la Plataforma SoC DE10Nano se consideraron la memoria RAM y la Memoria embebida (OnChip) en la FPGA. Según Martínez (2018), la memoria OnChip Tipo RAM (OCRAM) es la mejor elección cuando se emplean los IP's del University Program debido a su alta tasa de intercambio de datos. Esta memoria es de tamaño limitado por lo que la resolución de las imágenes tiene un máximo de 320x240 pixeles.

1.2. Diseño e interconexión de núcleos IP para la plataforma SoC DE10-Nano

Para el procesamiento de imágenes se evaluaron en primer lugar los núcleos IP's prediseñados de Intel-FPGA para el procesamiento de imágenes. Se decidió elegir los núcleos de video IP del Univesity Program (UP), pues son de código abierto y no requieren licencia, Intel Corporation (2018). Para evaluar el rendimiento de las arquitecturas HPS y HPS + FPGA se plantearon tareas distintas: conversión de color desde RGB a Escala de grises, detección de bordes Canny y filtro de mediana.

Para las dos primeras tareas se emplearon los núcleos IP "Color-Space-Converter" y "Edge-Detection" de la suite de video del University Program respectivamente; para el filtro de mediana se ha creado un núcleo IP llamado "video_filtro_mediana5x5" compatible con el resto de núcleos del University Program. El núcleo video_filtro_mediana5x5 está basado en la matriz

DOI: http://dx.doi.org/10.46925//rdluz.32.22

sistólica clásica de ordenamiento, Pimpale (2015), la cual fue generalizada para permitir el ordenamiento de 25 pixeles.

La OCRAM es el almacenamiento principal de las imágenes en la FPGA y el contador de marcos recibe señales de los DMA's. El bloque "Video signal 2" es una abstracción de un conjunto de núcleos encargados de acondicionar la señal de video para poder proyectarla en una pantalla.

El bloque "Linux desktop video signal" son un conjunto de núcleos IP encargados de recibir la señal de video del escritorio Linux LXDE, procedente del HPS y ya estaba originalmente en el diseño de la Arquitectura HPS.

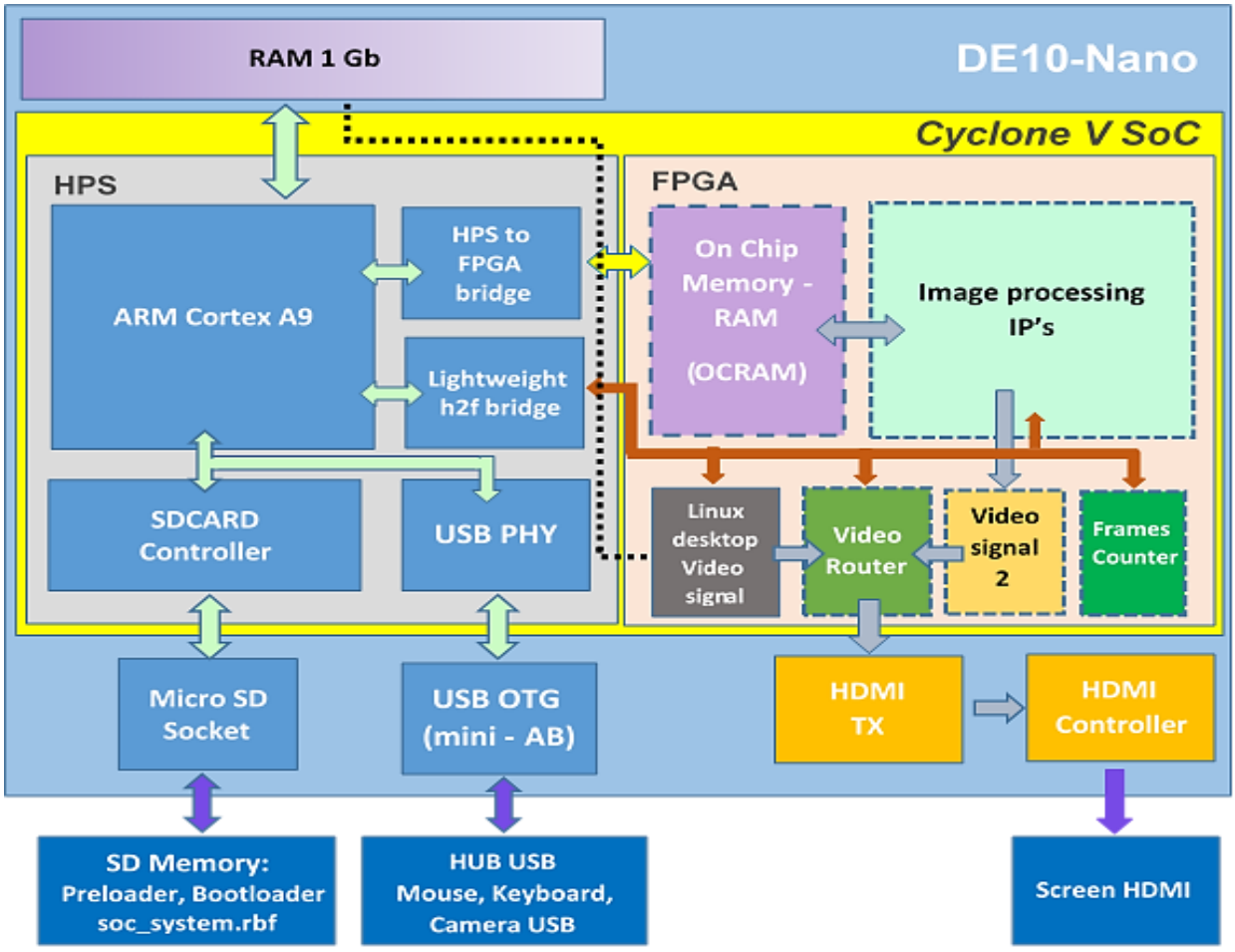


Figura 1. Hardware simplificado de la arquitectura HPS + FPGA
Fuente: Elaboración propia

DOI: http://dx.doi.org/10.46925//rdluz.32.22

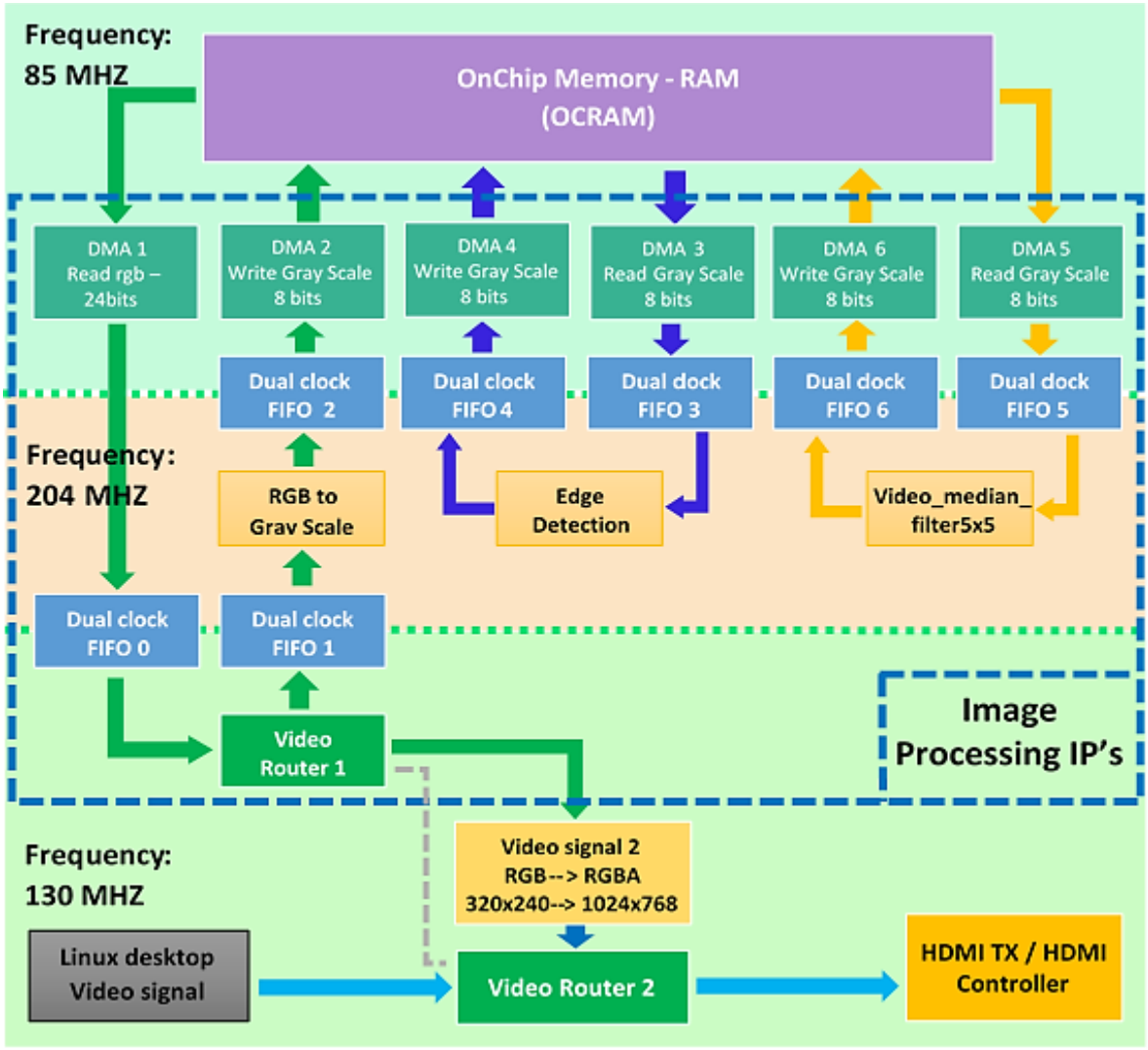


Figura 2. Configuración simplificada del FPGA en la arquitectura HPS + FPGA

Fuente: Elaboración propia

El bloque video Router es utilizado para direccionar la señal de video hacia el transmisor/controlador HDMI (HDMI TX / HDMI Controller), mediante este núcleo IP se puede seleccionar entre la señal de video de Linux y la señal de video procedente de los bloques de procesamiento de imágenes. La forma en que están interconectados los bloques de procesamiento de imágenes (Image Processing IP's) se indica de forma simplificada en la figura 2.

Los controladores DMA son utilizados para leer y escribir imágenes en la OCRAM y se han implementado dos de ellos en cada conjunto de bloques. Existen tres conjuntos de bloques

DOI: http://dx.doi.org/10.46925//rdluz.32.22

dedicados a realizar una tarea específica: conversión a escala de grises, detección de bordes y filtrado de mediana. Los bloques Dual Clock FIFO han sido utilizados para adaptar señales de video entre diferentes dominios de reloj.

1.3. Desarrollo de software para control y procesamiento de Imágenes

En la figura 3 se muestra una descripción general del software desarrollado para el control de hardware en el FPGA y visualización de resultados. Se han creado dos librerías con las funciones básicas que serán utilizadas para permitir la comunicación entre el HPS y los núcleos IP en el FPGA, así como para el control y configuración de los mismos. Adicionalmente, se ha creado una librería llamada ImageProcessing donde se implementan funciones para procesamiento de imágenes y métodos para la medición de tiempo de procesamiento utilizando la Arquitectura HPS y HPS+FPGA. Estas funciones son llamadas según la lógica planteada en una interfaz de usuario creada para facilitar la toma de datos del rendimiento. La interfaz gráfica se desarrolló con la librería CVUI, la cual es una librería de solo encabezado y depende únicamente de las primitivas de OpenCV, Dovyski.Github.Io. (2018). Todo el software se desarrolló en C/C++.

La librería fpgaT implementa todas las funciones necesarias para establecer comunicación con los núcleos IP implementados en la FPGA. Estas funciones permiten leer y escribir datos. La librería HardwareControl.h es la segunda Capa de software, incorpora funciones específicas que permiten configurar, enviar o leer información desde algún núcleo periférico especifico como por ejemplo el video_router o algún DMA_controller. Contiene funciones que permiten controlar el funcionamiento de los núcleos implementados en FPGA. Esta librería depende de fpgaT.h.

La librería ImageProcessing es la tercera capa de software. En esta capa de software se implementan funciones para procesar imágenes considerando las tareas de conversión a escala de grises, filtro de mediana y detección de bordes utilizando las Arquitecturas HPS y HPS+´FPGA. Esta librería hace uso de las secuencias implementadas en HardwareControl.h. La escritura de la imagen en la OCRAM se realiza de manera directa mediante el comando memcpy() tomando el trabajo de Frazer (2017).

DOI: http://dx.doi.org/10.46925//rdluz.32.22

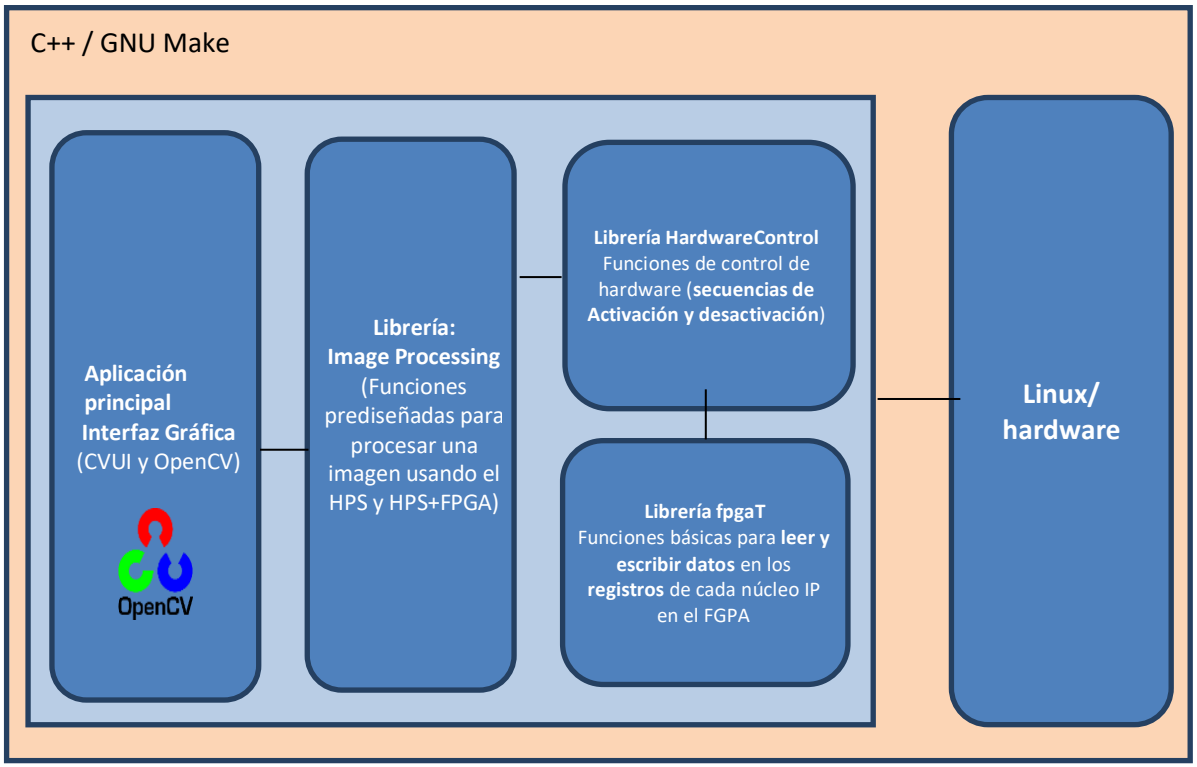


Figura 3. Interfaz de usuario para el control de hardware y procesamiento de imágenes
Fuente: Elaboración propia

La interfaz gráfica de la figura 4, junto a las librerías ImageProcessing.h y fpgaT.h se unieron en un único programa encargado en realizar el procesamiento de imágenes empleando las arquitecturas HPS y HPS+FPGA dependiendo de las opciones seleccionadas. Además de procesar las imágenes también se realiza la medición del tiempo promedio requerido para ejecutar los algoritmos de procesamiento, la medición del tiempo requerido para la escritura y lectura de la imagen procesada desde el HPS hacia la OCRAM.

En la figura 4 se muestra la interfaz corriendo en la tarjeta DE10Nano. El objetivo de la interfaz es controlar el sistema de procesamiento de imágenes de una manera más interactiva y facilitar la toma de datos respecto al rendimiento de las arquitecturas HPS y HPS + FPGA.

2. Resultados y discusión

Para comprobar si la nueva Arquitectura (HPS+FPGA) realiza un correcto procesamiento sobre las imágenes se procedió a evaluar la respuesta por medio de dos imágenes de prueba

DOI: http://dx.doi.org/10.46925//rdluz.32.22

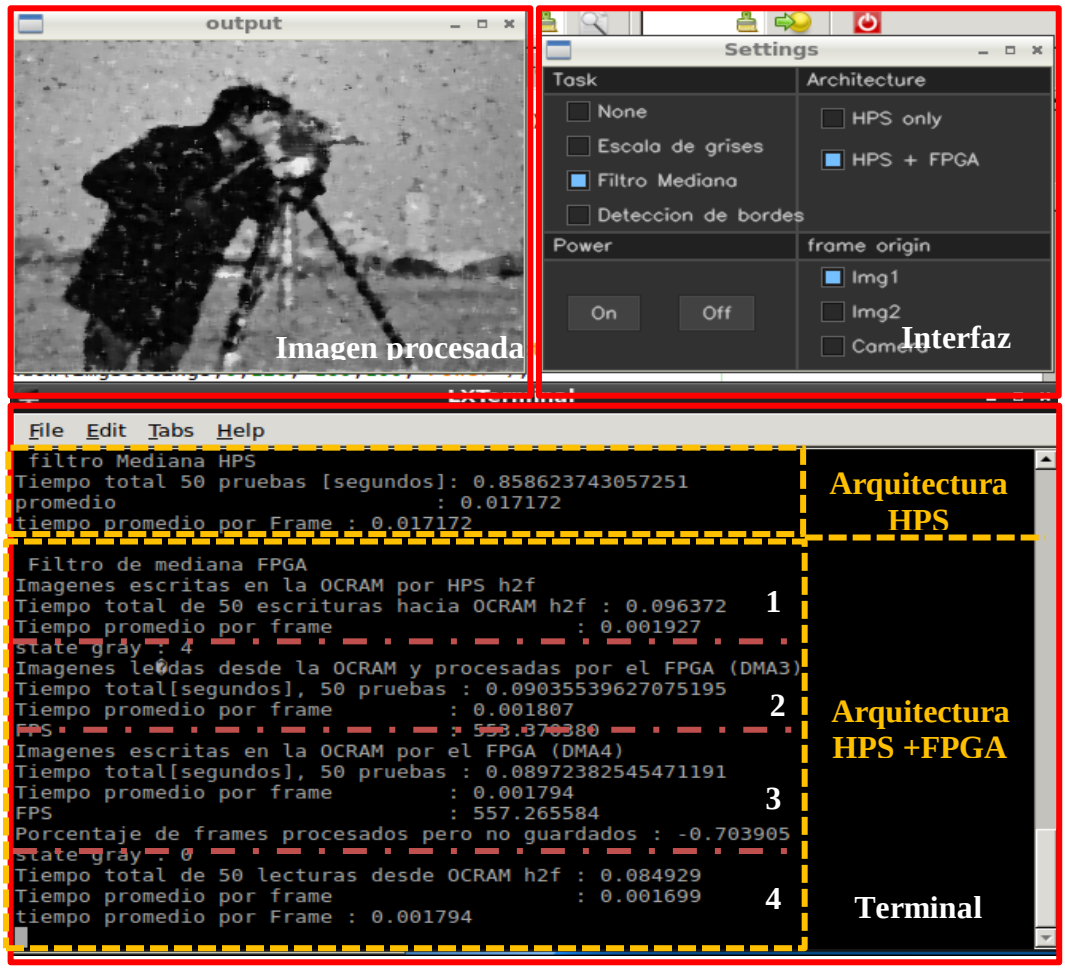
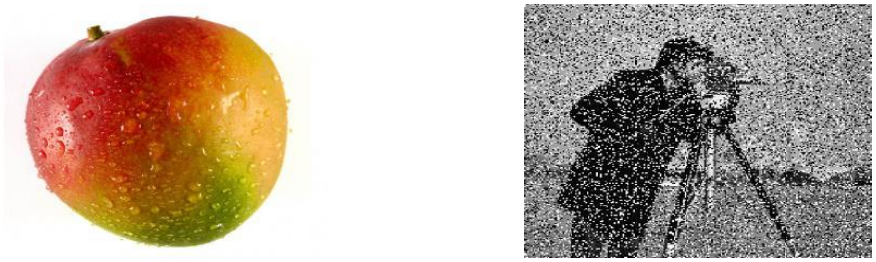


Figura 4. Interfaz gráfica
Fuente: Elaboración propia



(a) Imagen a color en formato RGB.

(b) Cameraman en escala de grises contaminada con ruido de tipo "Sal y pimienta."

Figura 5. Imágenes de prueba para el procesamiento de imágenes
Fuente: Elaboración propia

DOI: http://dx.doi.org/10.46925//rdluz.32.22

Para el caso de la Arquitectura HPS se utilizó la librería OpenCV para el procesamiento de imágenes y en el caso de la Arquitectura HPS+FPGA se utilizó la suite de video University Program de Intel-FPGA, obteniendo los siguientes resultados:

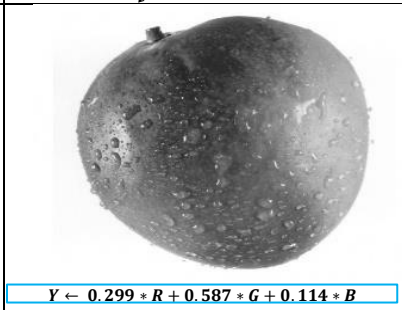
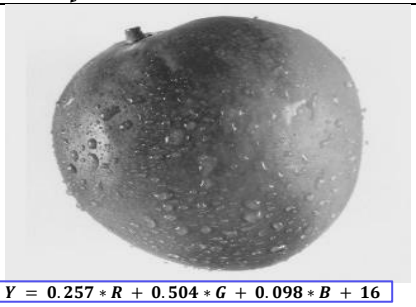
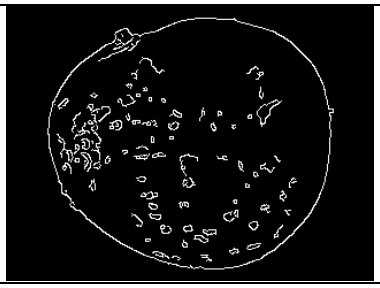
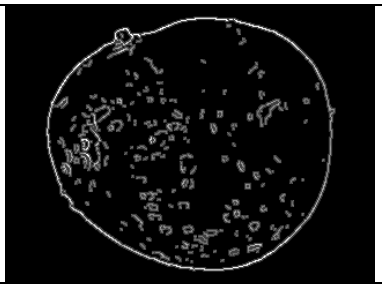
Algoritmo	Arquitectura HPS	Arquitectura HPS + FPGA
Conversión de RGB a escala de grises	 $Y \leftarrow 0.299 * R + 0.587 * G + 0.114 * B$	 $Y = 0.257 * R + 0.504 * G + 0.098 * B + 16$
Filtro de Mediana (5x5 px)		
Detección de Bordes		

Figura 6. Resultado del procesamiento de Imágenes.
Fuente: Elaboración propia

Para el caso de la conversión a escala de grises es lógico que el resultado no sea exactamente igual cuando es utilizada la arquitectura HPS+FPGA pues para realizar la conversión de color, el núcleo IP SpaceColourConverter, Intel Corporation (2018), trabaja con una ecuación diferente a la utilizada por la función ctvColor()de OpenCV, Docs.Opencv.Org. (2020), en la Arquitectura HPS.

DOI: http://dx.doi.org/10.46925//rdluz.32.22

Con respecto al filtro de mediana, si se obtiene un resultado muy similar para ambas arquitecturas, pues el filtro de mediana del núcleo video_median_filter5x5 y la función medianBlur() de OpenCV están basados en el mismo principio; es decir, se recorre una ventana de 5x5 pixeles por toda la imagen, estos pixeles son ordenados de forma ascendente o descendente según su valor numérico (0-255), y se toma el valor medio como resultado del filtroBradski y Kaehler(2008).

Para el Caso del algoritmo de detección de bordes se ha configurado en el HPS teniendo en cuenta las recomendaciones de OpenCV, Docs.Opencv.Org. (2020) y se ha tratado de ajustar el umbral del filtro Canny para que coincida con el resultado obtenido mediante la Arquitectura HPS+FPGA, la cual tiene un umbral definido por defecto, Intel Corporation (2018), solo con la finalidad de evaluar la similitud, pues este umbral depende de necesidades específicas de cada aplicación.

2.1. Evaluación del tiempo de ejecución

Con respecto a la Arquitectura HPS+FPGA se consideran tres etapas en el procesamiento de la imagen. La primera comprende la escritura de la imagen en la OCRAM desde el HPS; la segunda es el procesamiento de dicha imagen mediante los núcleos IP implementados en el FPGA; y la tercera etapa es la lectura de la imagen ya procesada mediante el HPS.

Luego de sumar los tiempos promedio para cada etapa y comparar el tiempo total de ejecución de la Arquitectura HPS+FPGA con el obtenido en la Arquitectura HPS se obtienen los datos de la figura 7.

En base a los resultados de la figura 7 se decide asignar las tareas al HPS y al FPGA según el tiempo de ejecución con la finalidad de obtener el mejor desempeño posible en la Arquitectura HPS+FPGA y compararlo con la Arquitectura HPS mediante una aplicación de procesamiento de imágenes basado en tres algoritmos básicos. La aplicación consiste en la adquisición de imágenes por medio de la cámara USB; esta imagen está originalmente en formato de color RGB, pasa por proceso de conversión a escala de grises, luego por un filtro de mediana, después por un filtro gaussiano y un detector de bordes para finalmente mostrar el resultado en la pantalla.

DOI: http://dx.doi.org/10.46925//rdluz.32.22

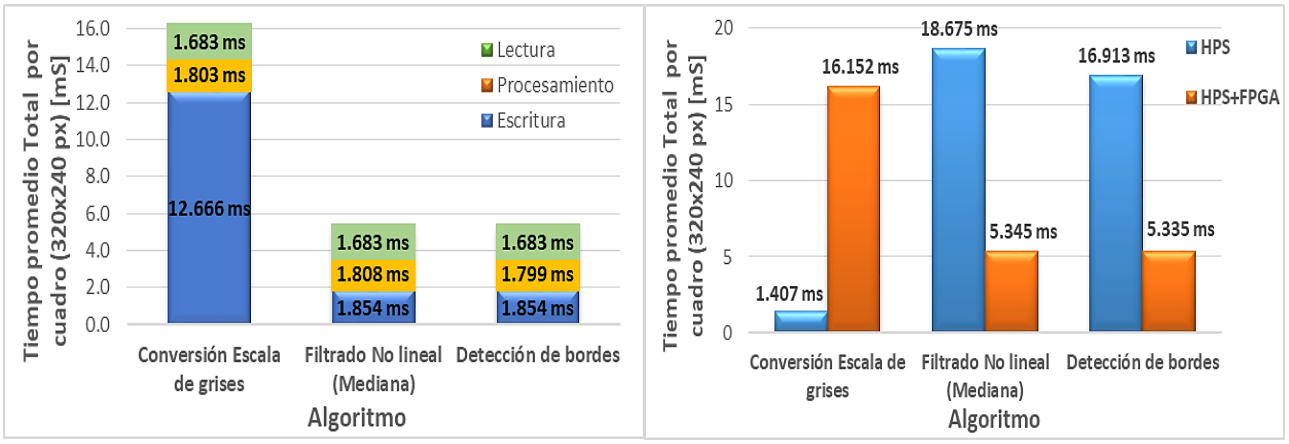


Figura 7. Tempo promedio total de ejecución por imagen (320x240px) en la Arquitectura HPS y HPS + FPGA
Fuente: Elaboración propia

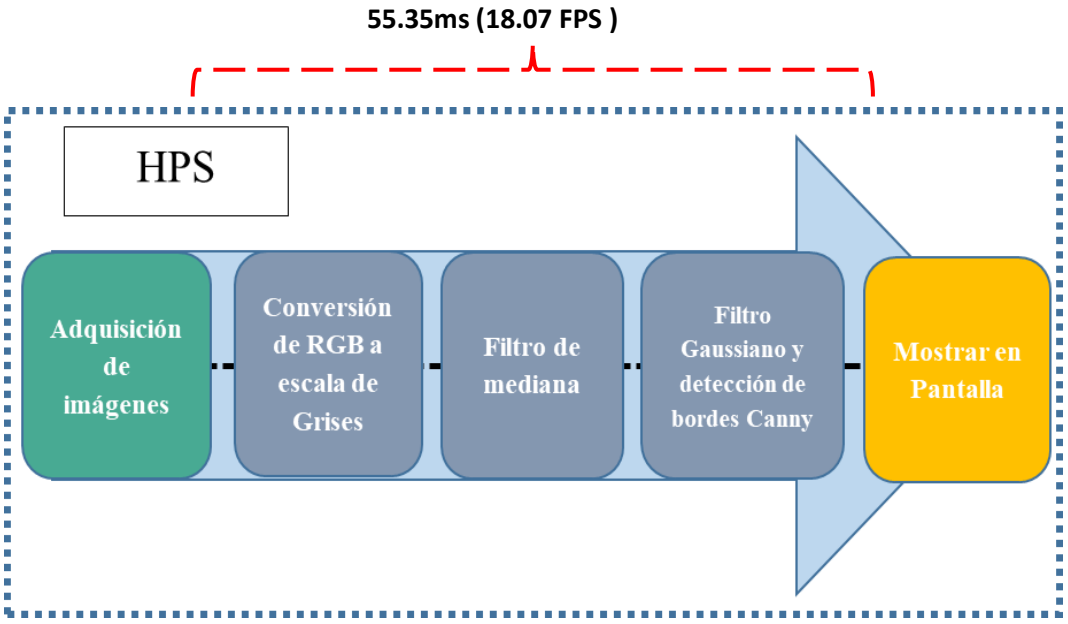


Figura 8. Propuesta de aplicación de procesamiento de imágenes usando Arquitectura HPS.
Fuente: Elaboración propia

Como se observa en la figura anterior, en la Arquitectura HPS, las tareas solo pueden ser asignadas al HPS, que luego de la experimentación se obtiene un tiempo promedio de 55.35 ms requerido para la ejecución de todo el algoritmo.

DOI: http://dx.doi.org/10.46925//rdluz.32.22

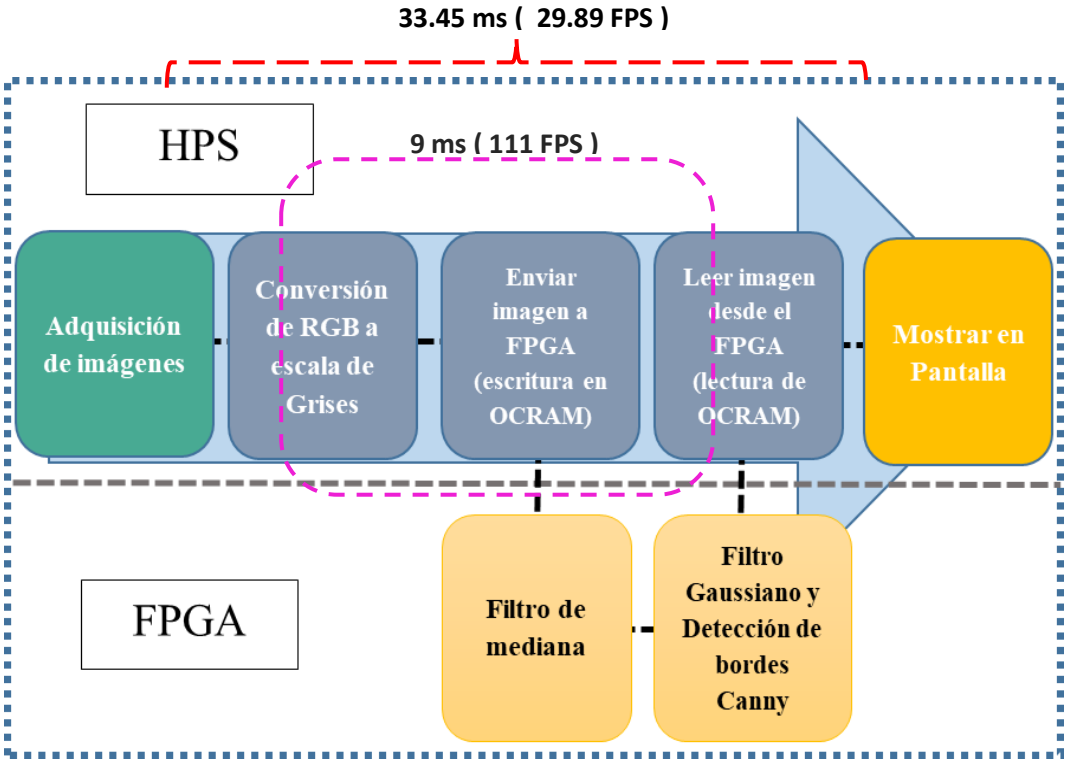


Figura 9. Propuesta de aplicación de procesamiento de imágenes usando Arquitectura HPS+FPGA
Fuente: Elaboración propia

En el Caso de la Arquitectura HPS+FPGA es posible asignar las tareas de procesamiento al HPS o al FPGA según sea conveniente, es por ello que en base a los resultados de la figura 7 se planteó asignar el filtrado de mediana y la detección de bordes al FPGA, obteniendo un promedio de 33.45ms como tiempo para la ejecución de todas las tareas (una mejora del 39%), tiempo del cual sólo 9 ms son necesarios para procesar la imagen proveniente de la cámara.

2.2. Evaluación del consumo de energía

Al hacer el análisis del consumo de energía por separado, considerando las tres tareas de procesamiento, se obtuvo que ambas arquitecturas presentan un consumo de energía similar en cada una de las tareas.

DOI: http://dx.doi.org/10.46925//rdluz.32.22

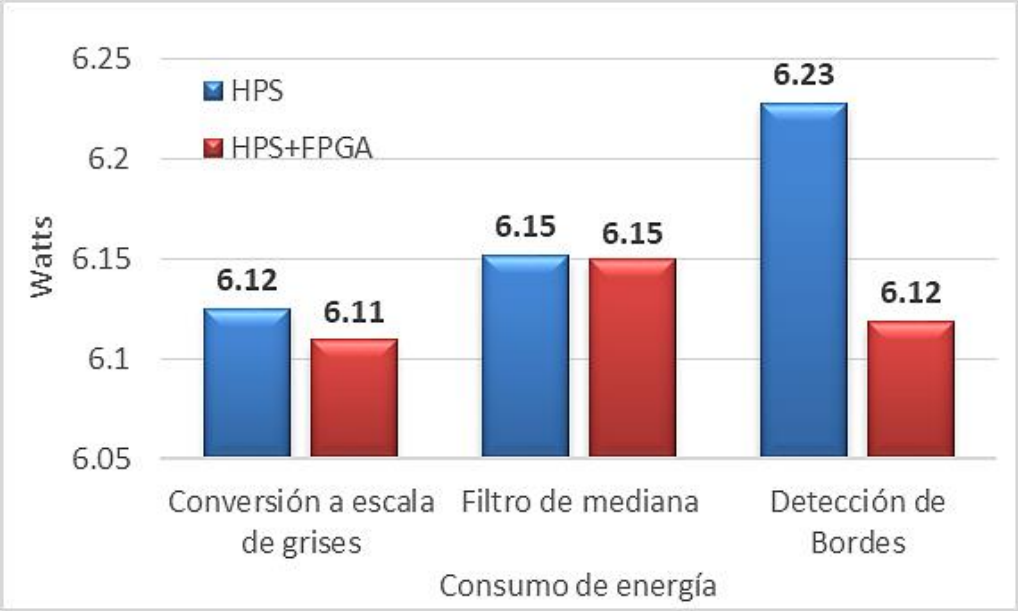


Figura 10. Consumo de energía presentado por la tarjeta DE10-Nano por cada tarea
Fuente: Elaboración propia

Sin embargo, al evaluar el consumo de energía considerando tareas combinadas como las que se presentan en la figura 8 y figura 9, se observó 6.85% más consumo de energía por unidad de tiempo en la Arquitectura HPS+FPGA que en la Arquitectura basada en HPS sólo.

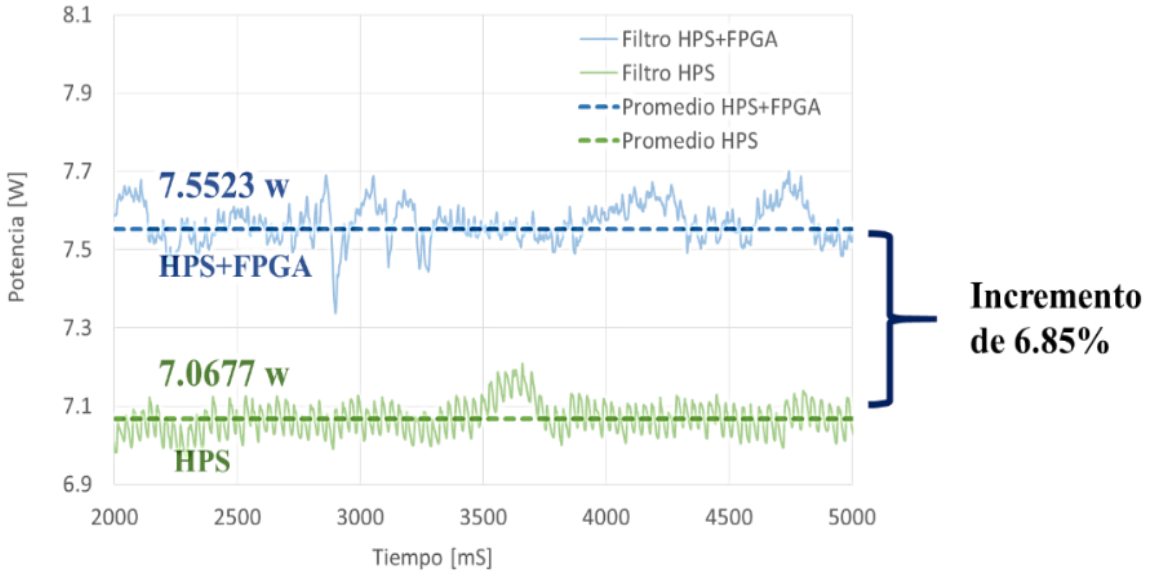


Figura 11. Consumo de energía presentado por la tarjeta DE10-Nano durante la ejecución de propuesta de aplicación.
Fuente: Elaboración propia

DOI: http://dx.doi.org/10.46925//rdluz.32.22

Conclusiones

- Con el diseño del hardware para desarrollar la Arquitectura HPS+FPGA se logró satisfactoriamente el intercambio de imágenes entre el HPS y el FPGA, las imágenes procesadas se muestran en una pantalla HDMI usando un sistema operativo basado en Linux que se ejecuta en el HPS de la misma tarjeta. Así mismo, los resultados muestran una mejora en el tiempo de ejecución de los algoritmos de procesamiento de imágenes al utilizar la Arquitectura HPS+FPGA con respecto a la Arquitectura HPS.
- Durante la ejecución de una aplicación de procesamiento de imágenes en tiempo real se observó un consumo de energía de 6.85% más al utilizar la Arquitectura HPS+FPGA. De esta manera se demostró que a más núcleos IP activos a la vez, mayor es el consumo total de energía. Sin embargo, los resultados cubren las expectativas, pues el incremento en el consumo de energía es relativamente bajo mientras que el tiempo de ejecución de los algoritmos mejora notablemente con el uso de la Arquitectura HPS+FPGA en lugar de la Arquitectura HPS.
- Para sistemas embebidos, el desarrollo de la Arquitectura HPS+FPGA puede ser utilizado en tareas o aplicaciones que requieran el uso de visión artificial, pues el tiempo de ejecución es lo suficientemente corto como para permitir aplicaciones en tiempo real. Es posible además aumentar la frecuencia de trabajo de los núcleos de procesamiento de video y de la memoria OnChip en la FPGA, e incorporar otros núcleos de procesamiento de video con la finalidad de poder realizar tareas más complejas en el procesamiento de imágenes.

Referencias

Bradski, G. y Kaehler, A. (2008). Learning OpenCV. Primera Edición. Sebastopol: O'Reilly Media, Inc.

Docs.Opencv.Org. (2020). Conversiones de espacio de color. Recuperado de https://docs.opencv.org/3.4/d8/d01/group__imgproc__color__conversions.html#ga4e0972be5de079fed4e3a10e24ef5ef0.

Docs.Opencv.Org. (2020). Detector Canny Edge. Recuperado de https://docs.opencv.org/3.4/da/d5c/tutorial_canny_detector.html.

DOI: http://dx.doi.org/10.46925//rdluz.32.22

Dovyski .Github.Io. (2018). Cvui. Recuperado de https://dovyski.github.io/cvui/.

Espinoza, H. (2016). Diseño e implementación de un sistema de seguridad y alerta para vehículos, basado en reconocimiento facial y localización gps, en una Raspberry pi b plus. Proyecto previo a la obtención del Título de Ingeniero en Electrónica y Control. Escuela Politécnica Nacional.

Frazer, R. (2017). Release tutorials-v1.0.0. Recuperado de https://github.com/intel-iot-devkit/terasic-de10-nano-kit/releases.

Intel Corporation. (2018). Fpga University Program. Video IP Cores for Intel® DE-Series Boards. University Program.

Martínez, O. (2018). Diseño de un SOPC (system on programmable chip) para el control de una cámara de 5MP con pantalla táctil en el entorno de trabajo de la tarjeta DE2-115 de Altera. Memoria TFM. Universitat Politècnica De València.

Mittal, S. y Vetter, J. A. (2015). Survey of Methods for Analyzing and Improving GPU Energy Efficiency, ACM Computing Surveys.

Pimpale, A. (2015). Optimized Systolic Array Design For Median Filter In Image Filtration. Tesis para optar el título de Master of Technology In Digital Communication. Patel College of Science & Technology, Bhopal.

Terasic Inc. (2018). DE10-Nano Cyclon V SoC with Dual-core ARM Cortex A9 - User Manual. Primera Edicion. Terasic Inc.