Análisis y evaluación de un módulo IME hardware para el codificador HEVC usando una plataforma SoC

Estefanía Alcocer, Otoniel López-Granado, Manuel P. Malumbres[1] y Roberto Gutiérrez[2]

Resumen— **En el estándar de codificación HEVC, la estimación de movimiento es una de las tareas más complejas del codificador de vídeo, necesitando un gran porcentaje del tiempo de codificación total. Esto se debe, principalmente, a los cambios introducidos en comparación con el estándar predecesor H264/AVC, siendo los siguientes: un conjunto más extenso de modos de partitición del Coding Tree Unit, la presencia de múltiples frames de referencia y el tamaño varible de los Coding Units. Además, HEVC adopta una estimación de movimiento por bloques de tamaño variable para obtener una eficiencia de codificación avanzada.**

En este trabajo, evaluamos un diseño de estimación de movimiento por enteros, IME, implementado en una plataforma SoC. En esta evaluación, mediremos el impacto en el Rate/Distortion al aplicar diferentes tamaños tanto de CTU como de áreas de búsqueda de referencia. Además, se evaluará el efecto de las transferencias del DMA requeridas en el rendimiento computacional. Esta arquitectura ha sido sintetizada e implementada en el Xilinx SoC, Zynq-7 Mini-ITX Motherboard XC7Z100 (xc7z100ffg900-2). Nuestra arquitectura IME hardware se puede configurar para trabajar con difrerentes tamaños de CTU y rangos de búsqueda del frame de referencia. Los resultados muestran que no hay diferencias significativas en términos de Rate/Distortion entre las diferentes configuraciones, pero sí existe un gran impacto en la complejidad del codificador cuando tanto el tamaño del CTU como el rango de búsqueda se incrementa.

Hemos evaluado nuestro diseño IME hardware usando diferentes configuraciones de CTU, tamaños de área de búsqueda y tamaños de ráfaga del DMA, con el fin de determinar la máxima ganancia con respecto al software de referencia del HEVC. Los resultados muestran que el tiempo de codificación se podría reducir 588 veces. Además, esta evaluación revela que las transferencias del DMA son el cuello de botella del sistema completo.

Palabras clave— **HEVC, codificación de vídeo, FPGA, estimación de movimiento.**

I. Introducción

EL estándar de codificación HEVC (High Efficiency Video Coding) [1] se presentó en Enero de 2013 por el JCT-VC (Joint Collaborative Team on Video Coding). Este nuevo estándar sustituye el actual estándar H.264/AVC [2] con el fin de tratar con las tendencias multimedia del mercado actual y futuro como los contenidos de vídeo con definiciones 4K y 8K y profundidades de color de alta calidad a 10 bits. HEVC ha mejorado significativamente la eficiencia de codificación en cuanto a su predecesor H.264/AVC en un factor de casi dos veces mientras se mantiene una calidad visual equivalente [3]. En cuanto a la complejidad, el decodificador HEVC no parece ser muy diferente al del H.264/AVC [4]. Sin embargo, se espera que el codificador HEVC sea considerablemente más complejo que el codificador de H.264/AVC [5] y será un tema candente de investigación en los próximos años, por ejemplo, codificar un segundo de un vídeo a una resolución de 1080p60 HD (High Definition) con el codificador del software de referencia puede llevar más de una hora de ejecución en un ordenador de sobremesa actual.

Como en estándares anteriores, la estimación de movimiento, ME (Motion Estimation), es la tarea más compleja del codificador, necesitando más del 90% del tiempo de codificación [6]. En HEVC, la complejidad es incluso más crítica debido a diferentes parámetros como son (a) un mayor conjunto de modos de partición del Coding Tree Unit (CTU), (b) la presencia del múltiples frames de referencia y (c) el tamaño variable de los Coding Units (CU) comparando con su predecesor H264/AVC. Además, HEVC adopta la técnica Variable Block Size Motion Estimation (VBSME) para obtener una eficiencia avanzada de codificación, lo que se logra a costa de un enorme aumento de la complejidad computacional.

Se han propuesto muchas arquitecturas hardware para acelerar el módulo del ME HEVC con el objetivo de reducir, tanto como sea posible, la complejidad total del codificador. El bloque de la estimación de movimiento por enteros, IME (Integer-pel Motion Estimation), está a cargo del ME. En la mayoría de las propuestas del estado del arte, las arquitecturas IME hardware se centran solamente en los algoritmos de búsqueda de movimiento ya que esto conlleva la mayoría del tiempo computacional del bloque IME. Generalmente, el algoritmo más popular de la búsqueda de movimiento en implementaciones hardware es el algoritmo Full Search (FS), de búsqueda completa. Este realiza una búsqueda en todos los puntos de la zona de búsqueda establecida de un frame de referencia, y como consecuencia, es capaz de proporcionar el resultado óptimo, es decir, un vector de movimento que minimiza el error residual del CTU actual.

Las arquitecturas propuestan en [6], [7], [8], [9], [10] presentan un bloque IME hardware usando un algoritmo de búsqueda FS. En [6], se propone una unidad de Suma de Diferencias Absolutas (SAD) en una FPGA (Field-Programmable Gate Array) la cual es capaz de analizar todos los modos de partición de

[1]Dpto. de Física y Arquitectura de Computadores, Univ. Miguel Hernández. Elche, e-mail: {ealcocer, otoniel, mels}@umh.es

[2]Dpto. de Ingeniería de Comunicaciones, Univ. Miguel Hernández. Elche, e-mail: roberto.gutierrez@umh.es

un CTU excepto el conjunto de modos de partición asimétricos. Los autores fijan un tamaño de área de búsqueda menor que la establecida en el estándar HEVC, siendo capaz de ejecutar hasta 30 fps con resoluciones de vídeo de 2k. En [8], el tamaño máximo de CTU se reduce a 32x32 píxeles con un rango de búsqueda de ±23 píxeles. Esta arquitectura se implementa en una FPGA y alcanza 30 fps a resoluciones 1080p. En [9], se estudian diferentes áreas de búsqueda configurables, logrando un máximo frame rate de 57 fps a 720p resolución de vídeo.

Por otro lado, en [11] y [12], se muestran diferentes implementaciones de estrategias de búsqueda de movimiento sub-óptimas, también conocidas como algortimos rápidos de ME, como los nuevos DS (Diamond Search) o TSS (Three Step Search). También han sido estudiadas arquitecturas hardware de ME similares para el anterior estándar H264/AVC en [13], [14], [15], [16], [17], los cuales son de nuestro interés debido a la gran similitud entre arquitecturas de los bloques IME en ambos estándares.

En nuestro trabajo previo [18], se presenta una nueva arquitectura hardware que realiza el cálculo del IME usando una FPGA. Los autores muestran dos técnicas innovadoras: (a) una nueva estructura del árbol de sumadores SAD, y (b) un nuevo orden de escaneo de memoria; logrando codificar a tasas de frame de hasta 116 fps y 30 fps para formatos de vídeo de 2K y 4K, respectivamente. La estructura del árbol de sumadores SAD realiza las sumas en el primer nivel del árbol, comenzando dede el tamaño máximo del CTU, y dividiendo por la mitad la cantidad de sumas de los siguientes niveles del árbol. Este enfoque es diferente al resto de los trabajos del estado del arte, los cuales dividen primero un CTU en bloques más pequeños para realizar acumulacicones consecutivas, manteniendo las mismas sumas en cada nivel y por tanto necesitando un mayor número de pasos para adquirir todos los SADs. Con esta propuesta, los autores toman ventaja de los recursos proporcionados por la FPGA, obteniendo la latencia mínima posible al calcular los SADs para todos los niveles y particiones de un CTU. De este modo, los SADs correspondientes a las particiones asimétricas se obtienen de una manera rápida y eficiente. En cuanto al nuevo orden de escaneo de memoria, una serie de registros de desplazamiento reconfigurables y elementos de procesado son los responsables de almacenar los píxeles necesarios para los frames de referencia y el actual, manteniéndolos siempre disponibles para el cálculo de los SADs y los vectores de movimentos (MVs) de un CTU. Así, se evitan los accesos a memoria externa ya que los píxeles disponibles son altamente reutilizados al reconfigurar el desplazamiento de estos de una manera más eficiente.

En este trabajo, evaluamos el diseño IME presentado en [18] al implementarlo en una placa de evaluación. En esta evaluación mediremos el imparcto en el Rate/Distortion (R/D) al aplicar diferentes tamaños de CTU y áreas de búsqueda. Además, evaluaremos el efecto de las transferencias desde memoria externa necesarias en el funcionamiento computacional.

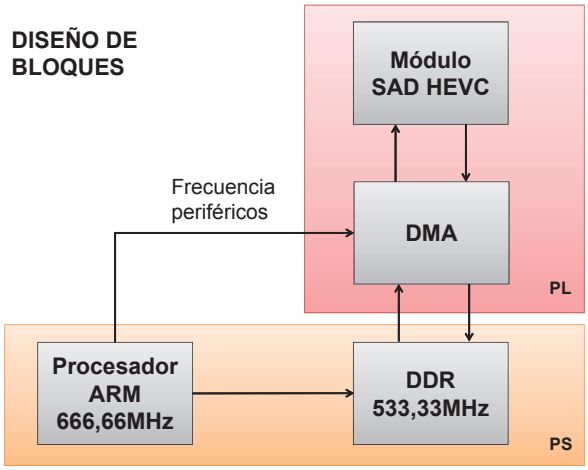


Fig. 1. Arquitectura hardware completa

El resto de este trabajo se organiza como sigue. En la Sección II se presenta una breve descripción del diseño de la arquitectura propuesta mientras que en la Sección III, se muestran los experimentos numéricos tanto software como hardware analizando los resultados de nuestro diseño hardware sobre una placa de evaluación. Finalmente, en la Sección IV se recogen algunas conclusiones y trabajo futuro.

II. Descripción de la arquitectura hardware

En esta sección, presentamos un resumen de un diseño IME completo en una plataforma SoC (System-On-Chip) usando el módulo SAD HEVC propuesto en [18]. El SoC consta de dos partes bien definidas, un sistema de procesado, PS (Processing System), basado en un procesador ARM y varios periféricos como Ethernet, USB, etc., y una lógica programable, PL (Programmable Logic), formada por una FPGA. Nuestra arquitectura ha sido modelada en VHDL, y ha sido sintetizada, simulada, implementada y testada en el Xilinx SoC, Zynq-7 Mini-ITX Motherboard XC7Z100 (xc7z100ffg900-2). La corrección de nuestro diseño se ha probado y verificado con el modelo de referencia del HEVC, HM 14 [19].

En la arquitectura propuesta, el procesador ARM controla la transferencia entre el módulo IME SAD y la memoria DDR (Double Data Rate) la cual almacena los frames de referencia y el frame actual, mediante un módulo DMA (Direct Memory Access). El sistema completo se muestra en la Figura 1.

El procesador ARM trabaja a 666.66 MHz, y la DDR a 533.33 MHz, mientras que la frecuencia de reloj del PL está limitada a la frecuencia máxima del módulo SAD HEVC el cual es el responsable del cálculo del IME.

En cuanto al proceso IME, cada frame de vídeo se subdivide y particiona en unidades de codificación básicas llamadas CUs. La estructura de codificación

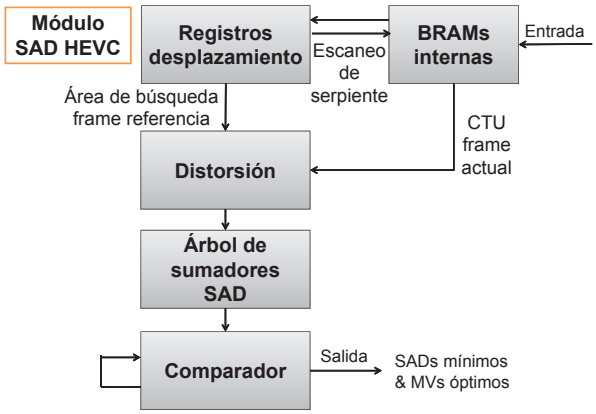


Fig. 2. Módulo SAD HEVC hardware

en HEVC consta de CUs con un tamaño máximo de 64x64 píxeles, tan grande como un CTU (Coding Tree Unit), los cuales pueden ser divididos recursivamente en bloques cuadrados hasta un tamaño de 8x8 píxeles. Cada CU consta de unidades de predicción PUs (Prediction Units) cuyo tamaño puede variar desde el tamaño máximo del CU hasta 4x8 o 8x4 en este caso de predicción Inter, soportando 8 modos de partición. En nuestra propuesta, el módulo SAD HEVC responsable del cálculo IME se puede configurar para trabajar con tamaños de CTU de 64x64 y 32x32. En el caso de un CTU de 64x64 píxeles, el PL puede trabajar a 220MHz mientras que con un tamaño de CTU de 32x32 la frecuencia de reloj del PL se fija al máximo de la placa de evaluación usada, 250 MHz. Sin embargo, dicho módulo por separado podría trabajar a una frecuencia máxima de 333 MHz. Por lo tanto, la frecuencia máxima está limitada a la frecuencia de reloj máxima del PL, la cual es 250 MHz para la plataforma de evaluación utilizada.

Nuestro módulo consta de (a) áreas de memoria interna para almacenar píxeles del CU del frame actual y los píxeles pertenecientes al área de búsqueda de los frames de referencia, (b) un bloque de distorsión donde los píxeles de ambos frames se restan, (c) un árbol de sumadores SAD y (d) un comparador acumulativo que guarda los valores mínimos de SAD y sus MVs correspondientes para todas las particiones de CU, como se muestra en la Figura 2. Para más detalles de este módulo, ver el trabajo previo de los autores [18].

En la Tabla I, se muestran los recursos usados para la implementación de nuestro módulo SAD HEVC para unos tamaños de CTU máximos de 64x64 y 32x32, respectivamente, en una Zynq-7 Mini-ITX Motherboard XC7Z100 FPGA. Como se observa, nuestro módulo SAD HEVC require un 63% y un 16% del total de área usada para unos tamaños de CTU de 64x64 y 32x32, respectivamente.

III. Experimentos numéricos

ME es una tarea integrada en la predicción Inter del HEVC. Por esta razón, se ha considerado un modo de configuración Low Delay B (LB). En LB,

TABLA I
Área utilizada en Mini-ITX

Recursos	CTU 64x64	CTU 32x32
LUTs	166383	40911
Flip-flops	190159	50028
Block-RAMs	32	16

el primer frame se codifica como una imagen intra y el resto se codifican como frames bidireccionales generalizados (tipo inter). Esta estructura de codificación es la más popular para el streameing de vídeo, diseñado principalmente para comunicaciones interactivas en tiempo real.

Dada la configuración previa, se han realizado diferentes experimentos del IME de HEVC con el fin de observar como ambos parámetros, el tamaño del CTU y el tamaño del área de búsqueda, impactan en el R/D y en la complejidad del codificador HEVC. Se han elegido unos tamaños de CTU de 64x64 y 32x32 píxeles y sus correspondientes rangos de búsqueda, SR (Search Range), cuyos tamaños son el 100% del CTU, el 80% del CTU, y/o el 50% del CTU. Además, se han seleccionado dos secuencias de vídeo del conjunto de vídeos de las *common conditions* del HEVC: *ParkScene* con una resolución de 1920x1080 (24 fps) y *Traffic* a 2560x1600 (30 fps). Para realizar estos tests, hemos usado el módelo de referencia del HEVC, HM 14 [19]. El software de referencia HEVC se ha compilado con Visual Studio 2010 y ejecutado sobre un PC con las siguientes características: Intel Core i7-3770 CPU 3.40GHz con 8GB RAM.

Se ha medido el tiempo del módulo IME software usando un algoritmo de búsqueda FS cuando se codifica una secuencia de vídeo. Nuestro trabajo se ha centrado en el diseño e implementación hardware de un algoritmo FS que es capaz de acelerar significativamente el proceso de estimación de movimento del codificador HEVC sin perder rendimiento R/D, ya que el algoritmo FS es la estrategia más adecuada para procesos hardware, el cual busca el movimiento exhaustivamente para todos los PUs en cada uno de los puntos del área de búsqueda establecida. Por lo tanto, debido a la regularidad computacional y la excelente calidad de vídeo de salida, la estimación de movimiento con FS es comunmente empleada en implementaciones hardware [20].

En al Figura 3, se muestra el porcentaje del tiempo total de codificación que requiere el IME en software usando el algoritmo de búsqueda FS, sin contar con el procedimiento de optimización R/D. Este porcentaje de tiempo es similar para todas las secuencias analizadas. Como se puede observar en la Figura 3, este tiempo depende de los tamaños de CTU y SR. Normalmente, el módulo IME del codificador requiere más tiempo cuanto mayor sea el tamaño tanto del CTU como del SR, como es de esperar. El tiempo que el codificador HEVC necesita para realizar la es-

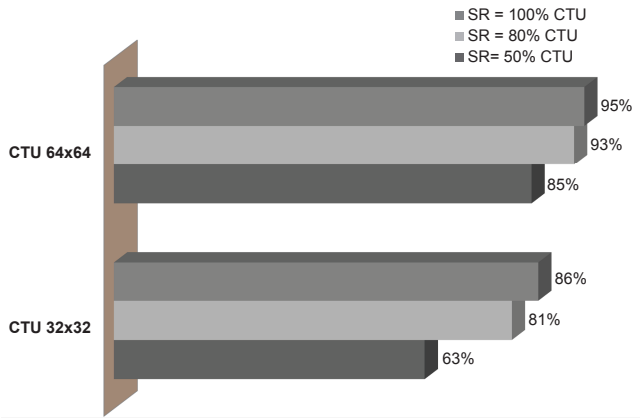
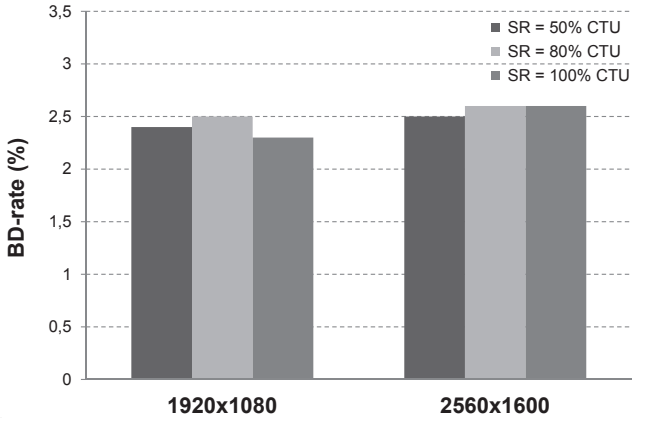
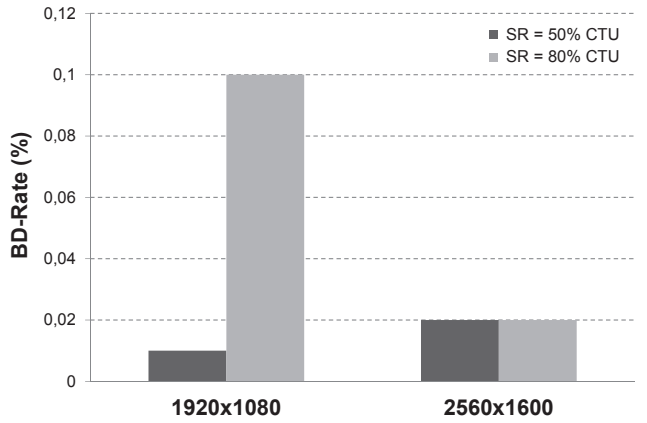


Fig. 3. Porcentaje de tiempo de codificación SW del módulo SAD con una estrategia Full-Search



(a) BD-Rate de curvas R/D con un tamaño de CTU 32x32 y SR de 16, 26 y 32



(b) BD-Rate de curvas R/D con un tamaño de CTU 64x64 y SR de 32 y 56

Fig. 4. BD-Rate respecto a una curva R/D de referencia con un tamaño de CTU y SR de 64

timación de movimiento varía entre el 63% y el 95% del tiempo total para codificar una secuencia de vídeo completa. Por tanto, cobra sentido el diseño de una arquitectura hardware que realice el proceso IME de una manera más rápida y precisa con el fin de reducir tanto la complejidad del módulo IME como el tiempo total de codificación en la medida de lo posible.

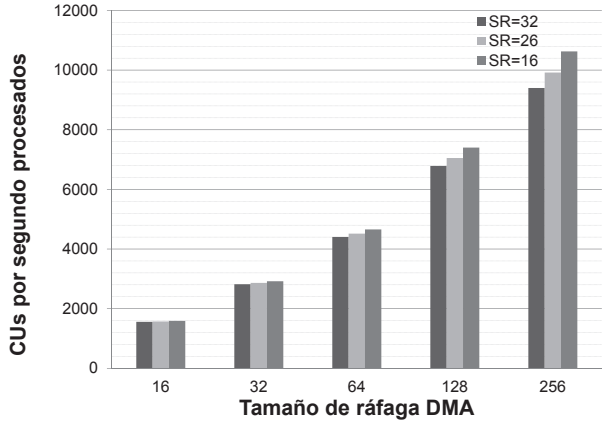
Además, se ha analizado el impacto de los parámetros previos en el R/D mediante el cálculo de la métrica de Bjontegaard (BD-rate). La métrica de Bjontegaard calcula el ahorro en tanto por ciento del bit-rate en media entre dos curvas de Rate/Distortion. Las curvas R/D se han obtenido para los siguientes niveles de compresión (valores QP): 22, 27, 32 y 37, teniendo en cuenta una curva de referencia con la configuración típica dada por el modelo software HEVC, HM-14, con un tamaño de CTU de 64x64 píxeles y un rango de búsqueda, SR, de 64 (tamaño de área de búsqueda de 128x128).

En la Figura 4, se puede observar el porcentaje de BD-Rate obtenido con diferentes tamaños de CTU y SR para vídeos de resoluciones 1920x1080 y 2560x1600. Como se muestra, existen ligeras diferencias entre los tamaños de SR para un tamaño de CTU dado, especialmente cuando el tamañao de CTU es 64x64, donde la diferencia es, como máximo, 0.1% aproximadamente (ver Figura 4(b)). De cualquier modo, las diferencias entre las posibles configuraciones son despreciables, siendo un 2.7% el bitrate incrementado, como máximo, para una calidad (PSNR) fija dada, cuando el tamaño del CTU es de 32x32 (ver Figura 4(a)). Aunque las diferencias del BD-Rate podrían depender del tipo de contenido de vídeo, se han obtenido resultados similares con otras secuencias de vídeo.

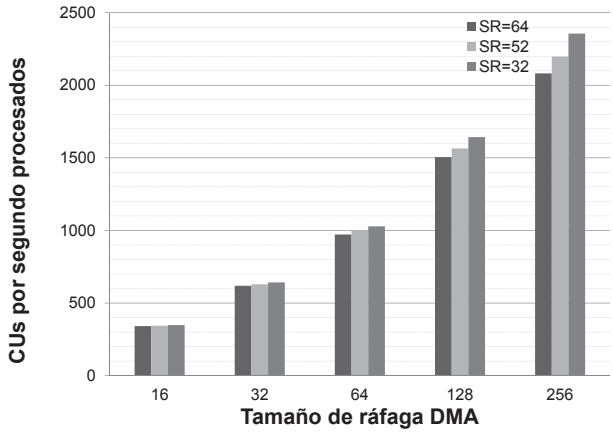
Por tanto, con el fin de reducir la complejidad permitiendo versiones más rápidas con un consumo reducido, los tamaños de CTU y SR pueden reducirse tanto como sea posible debido al insignificante impacto de estos parámetros en el resultado anterior en cuanto al BD-rate.

En cuanto a la arquitectura hardware del IME propuesta, se han medido los CUs por segundo que es capaz de procesar dependiendo de los tamaños de CTU, SR y ráfaga de DMA. En nuestro diseño IME completo descrito en la Sección II, el tamaño de palabra (o datos) del DMA se ha definido como 32 bits y el tamaño de ráfaga (palabras a transmitir en una transferencia) se puede configurar desde 16 a 256 palabras. Por ejemplo, en el caso de un tamaño de CTU de 32x32 píxeles donde la frecuencia de operación es 250 MHz y con un tamaño de ráfaga del DMA de 256, el sistema propuesto puede transferir con una tasa de 2Mbps.

En la Figura 5 se muestra la cantidad de CUs por segundo procesados para cada tamaño de ráfaga del DMA. Las Figuras 5(a) y 5(b) muestran como influye el SR en los CUs por segundo cuando el tamaño del CTU es 32x32 y 64x64, respectivamente. Como se puede observar, el número de CUs por segundo procesados aumenta cuando el tamaño de ráfaga del DMA también lo hace, siendo 10626 y 2356 los máximos CUs por segundo para los tamaños de CTU 32x32 y 64x64, respectivamente. Ese incremento es exponencial debido a que el tiempo que necesita la inicialización del módulo DMA es constante e independiente del tamaño de ráfaga del DMA. Por tanto, la configuración compuesta por un tamaño máximo de ráfaga de 256, un tamaño de CTU de 32x32 y un SR de 16 es la configuración más apropiada para con-



(a) CUs por segundo procesados para un tamaño de CTU de 32x32



(b) CUs por segundo procesados para un tamaño de CTU de 64x64

Fig. 5. CUs por segundo procesados para cada tamaño de ráfaga DMA con nuestro módulo SAD HEVC propuesto

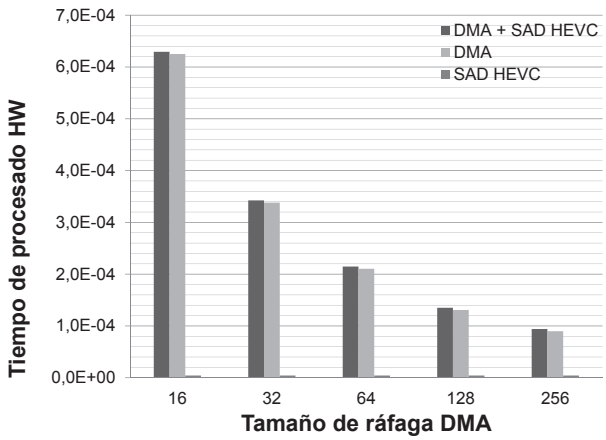


Fig. 6. Tiempo de procesado hardware para diferentes tamaños de ráfaga del DMA

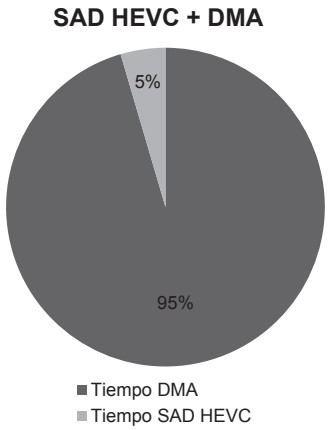


Fig. 7. Porcentaje de tiempo de las transferencias DMA y el módulo SAD para un tamaño de ráfaga de 256

seguir el mayor número de CUs por segundo usando nuestro sistema hardware.

Sin embargo, el número de CUs por segundo alcanzados previamente depende del tiempo completo de procesado de nuestro sistema hardware, el cual consta de las tranferencias del DMA y el módulo SAD HEVC. Así, en la Figura 6 mostramos el tiempo del DMA, el tiempo del módulo SAD HEVC y el tiempo total de procesar un CU para cada tamaño de ráfaga del DMA cuando establecemos la configuración de un CTU de 32x32 y un tamaño de SR del 50% CTU (±16). Como se esperaba, el tamaño máximo de ráfaga del DMA proporciona los mejores resultados, necesitando el menor tiempo de procesado de un CU. Además, dependiendo de la ráfaga de DMA elegida, las diferencias entre los tiempos de las transferencias del DMA y del módulo SAD HEVC varían, siendo el módulo SAD HEVC 21 veces más rápido para un tamaño de ráfaga de 256 y 146 veces más rápido para una ráfaga de 16 palabras.

Por lo tanto, el cuello de botella del tiempo total de nuestro sistema hardware se debe a las transferencias DMA, como también se puede observar en la Figura 7. Habiendo escogido un tamaño de ráfaga de DMA de 256, las transferencias representan el 95% del tiempo total de procesado de un CU, mientras que el módulo SAD HECV solamente necesita el 5% del tiempo. Este cuello de botella puede mitigarse en trabajos futuros, por ejemplo, transfiriendo solamente las partes de los frames de referencia que difieren de los que ya están almacenados en las BRAMs internas. De esta manera, las tranferencias DMA se podrán reducir y por consiguiente, el tiempo total de procesado requerido.

Tras realizar el análisis completo, se puede determinar que la configuración hardware que mejor se adapta a los requerimientos de la aplicación (bajo consumo de potencia, tiempo de codificación, calidad de vídeo comprimido) es la establecida por un tamaño de CTU de 32x32, un SR de 16 y un tamaño de ráfaga DMA de 256.

Aunque el algoritmo de búsqueda FS es el más ampliamente usado en las implementaciones hardware del ME, el software de referencia del HEVC utiliza por defecto el algoritmo Diamond Search. Así, en la Figura 8 se muestra la ganancia que se obtiene utilizando nuestra propuesta hardware del IME, con el conjunto de configuranción anteriormente establecido, en comparación con el módulo IME software utilizando tanto FS como DS. Por tanto, observando la información proporcionada por la Figura 8, para una resolución de vídeo de 2560x1600, la integración de nuestro módulo hardware acelerará la

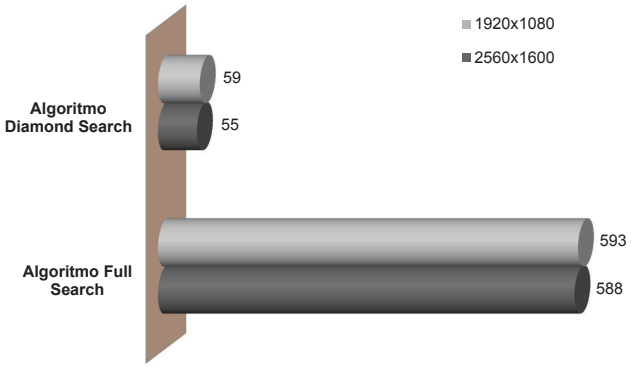


Fig. 8. Ganancia hardware frente a los algoritmos software Full Search y Diamond Search

computación del IME en 55 veces en el caso de una estrategia de búsqueda DS y 588 veces con la estrategia FS.

IV. Conclusiones

En este trabajo, se ha presentado una arquitectura hardware completa del IME, la cual incluye un módulo DMA. Se ha analizado cómo el tamaño de CTU y el rango de búsqueda afectan al funcionamiento del IME del HEVC en términos de Rate/Distortion y tiempo de procesado. Como se ha mostrado, las diferencias en R/D son despreciables para cualquier configuración, siendo un 2.7% el bitrate incrementado cuando el tamaño del CTU es de 32x32 y 0.1% para un CTU de 64x64. En cuanto a la complejidad, tanto el tamaño de CTU como el SR impactan en el tiempo total de codificación, obteniendo un tiempo de procesado menor con la configuración formada por un CTU de 32x32 y un SR del 50% del tamaño de CTU. Remarcar que el módulo ME requiere entre el 63% y el 95% del tiempo total de codificación.

Respecto a la evaluación de la propuesta hardware del IME, se han medido los máximos CUs por segundo que pueden ser procesados en función de los tamaños del CTU, SR y ráfaga de DMA. Los resultados muestran que el número de CUs procesados se incrementa cuando el tamaño de ráfaga lo hace, siendo 10626 el máximo número de CUs por segundo procesados para un tamaño de CTU de 32x32, un SR del 50% del CTU y una ráfaga de DMA de 256. Analizando estos resultados, se confirma que el cuello de botella del sistema reside en las transferencias DMA, necesitando éstas el 95% del tiempo total de procesado.

Por tanto, teniendo en cuenta tanto el R/D como la complejidad de codificación, el mejor conjunto de parámetros de configuración para nuestra arquitectura hardware del IME es el conformado por un CTU de 32x32 píxeles , un SR del 50% del tamaño de CTU ($\pm$16) y un tamaño de ráfaga del DMA de 256. En resumen, con la inclusión de nuestra propuesta hardware del IME, el tiempo de codificación se puede acelerar 588 veces.

Como posible trabajo futuro, se pretende reducir el tiempo de las transferencias DMA, reusando parte de las áreas de búsqueda de los frames de referencia para el cálculo del IME en CUs contiguos, transfiriendo solamente los nuevos píxeles de referencia requeridos en cada momento. Además, puesto que solamente se utiliza un 16% de los recursos totales de la plataforma, sería posible añadir más módulos SAD HEVC en nuestro sistema para acelerar todavía más la computación del ME.

Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Economía y Competitividad a través del proyecto I+D con referencia TIN2015-66972-C5-4-R y co-financiado mediante fondos FEDER.

Referencias

[1] B. Bross, W.J. Han, J.R. Ohm, G.J. Sullivan, Y-K Wang, and T. Wiegand, "High efficiency video coding (HEVC) text specification draft 10," *Document JCTVC-L1003 of JCT-VC, Geneva*, January 2013.

[2] ITU-T and ISO/IEC JTC 1, "Advanced video coding for generic audiovisual services," *ITU-T Rec. H.264 and ISO/IEC 14496-10 (AVC) version 16, 2012*, 2012.

[3] G.J. Sullivan, J.R. Ohm, W.J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *Circuits and systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1648 –1667, December 2012.

[4] J. Ohm, G.J. Sullivan, H. Schwarz, Thiow Keng Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards - including high efficiency video coding (hevc)," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1669–1684, 2012.

[5] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC complexity and implementation analysis," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1685–1696, 2012.

[6] Ahmed Medhat, Ahmed Shalaby, Mohammed S Sayed, and Maha Elsabrouty, "A highly parallel sad architecture for motion estimation in hevc encoder," in *IEEE Asia Pacific Conference on Circuits and Systems (APCCAS'14)*, Ishigaki, Nov. 2014, pp. 280–283.

[7] J. Byun, Y. Jung, and J. Kim, "Design of integer motion estimator of hevc for asymmetric motion-partitioning mode and 4k-uhd," *Electronics Letters*, vol. 49, no. 18, pp. 1142–1143, 2013.

[8] Xu Yuan, Liu Jinsong, Gong Liwei, Zhang Zhi, and Robert K.F. Teng, "A high performance vlsi architecture for integer motion estimation in hevc," in *IEEE 10th International Conference on ASIC (ASICON'13)*, Shenzhen, Oct. 2013, pp. 1–4.

[9] Thomas D'huys, "Reconfigurable data flow engine for hevc motion estimation," in *IEEE International Conference on Image Processing (ICIP'14)*, Paris, Oct. 2014, pp. 1223–1227.

[10] Antonio Navarro Purnachand Nalluri, Luis Nero Alves, "High speed sad architectures for variable block size motion estimation in hevc video coding," in *IEEE International Conference on Image Processing (ICIP'14)*, Paris, Oct. 2014, pp. 1233–1237.

[11] Vidyalekshmi VG, Deepa Yagain, and Ganesh Rao K, "Motion estimation block for hevc encoder on fpga," in *IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE'14)*, Jaipur, India, May 2014, pp. 1–5.

[12] P. Davis and Marikkannan Sangeetha, "Implementation of motion estimation algorithm for h.265/hevc," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 3, no. 3, pp. 122–126, 2014.

[13] Ching-Yeh Chen, Shao-Yi Chien, Yu-Wen Huang, Tung-Chien Chen, Tu-Chih Wang, and Liang-Gee Chen, "Analysis and architecture design of variable block-size motion estimation for h.264/avc," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 3, pp. 578–593, 2006.

[14] Wajdi Elhamzi, Julien Dubois, and Johel Miteran, "An efficient low-cost fpga implementation of a configurable motion estiation for h.264 video coding," *Springer Journal of Real-Time Processing*, vol. 9, no. 1, pp. 19–30, 2014.

[15] Theepan Moorthy and Andy Ye, "A scalable architecture for variable block size motion estimation on field-programmable gate arrays," in *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE'08)*, Niagara Falls, May 2008, pp. 1303–1308.

[16] M Kthiri, P Kadionik, H Levi, H Loukil, Ben Atitallah, and N Masmoudi, "An fpga implementation of motion estimation algorithm for h.264/avc," in *IEEE 5th Internation Symposium on I/V Communications and Mobile Network (ISVC'10)*, Rabat, Sept. 2010, pp. 1–4.

[17] Grzegorz Pastuszak and Mariusz Jakubowski, "Adaptive computationally scalable motion estimation for the hardware h.264/avc encoder," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 5, pp. 802–812, 2013.

[18] Estefania Alcocer, Roberto Gutierrez, Otoniel Lopez-Granado, and Manuel P. Malumbres, "Design and implementation of an efficient hardware integer motion estimator for an hevc video encoder," *Journal of Real-Time Image Processing*, pp. 1–11, 2016.

[19] HEVC software repository HM–14.0 reference model, https://hevc.hhi.fraunhofer.de/trac/hevc/browser/tags/HM-14.0, ," .

[20] Youn-Long Steve Lin, Chao-Yang Kao, Hung-Chih Kuo, and Jian-Wen hen, *VLSI Design for Video Coding - H.264/AVC Encoding from Standard Specification to Chip*, Springer, New York, NY, 2010.