Implementation of the JasPer Encoder in a NIOS II processor

Jaime-Andres Arteaga
Bionanoelectronics Group
Universidad del Valle
Cali, Colombia
andreaga@univalle.edu.co

Jaime Velasco-Medina
Bionanoelectronics Group
Universidad del Valle
Cali, Colombia
jvelasco@univalle.edu.co

Abstract—This article presents the implementation of the JasPer encoder on a NIOS II processor. JasPer is a software-based implementation of the JPEG2000 codec and is included in the Part 6 of the standard. In this case, only the encoder is implemented and its performance is compared with that of a software JPEG encoder from the Independent JPEG Group. PSNR calculations for a 320x240-pixel Lena.bmp image show that the reconstructed images from the JasPer encoder have fewer distortions than those reconstructed from the JPEG encoder. Specifically, the JPEG2000 image is 4.56 dB closer to the original one than the JPEG image when the compression ratio is 106:1. A profiling analysis is performed in order to determine which parts of the JasPer encoder have to be synthesized into FPGA for a more efficient implementation. Profiling results show that the DWT and Tier-1 modules are the most time-consuming stages of the JPEG2000 encoding algorithm in both lossless and lossy compression.

Keywords: Compression, Discrete Wavelet Transform, FPGA, JasPer Software, JPEG2000 encoder, NIOS II Embedded Processor.

I. INTRODUCTION

Nowadays the JPEG standard is one of the most commonly used for image compression. It is found in almost every system: either it is a general purpose system, such as a laptop, or an embedded system, such as a digital camera. However, it is not optimum and has many weaknesses that make it not suitable in many areas, e.g. multimedia applications. One of these weaknesses is the blocking artifacts generated in the reconstructed image, as a result of processing the original data using blocks of 8x8 pixels during the DCT (Discrete Cosine Transform). These artifacts give to the reconstructed image a squared-appearance that is more visible when the JPEG is configured for a higher compression ratio.

The JPEG2000 standard was developed to address this and other drawbacks of the JPEG standard. One of its main features is the presence of a single unified architecture for both lossless and lossy compression. As a result, the same architecture is suitable for applications where exact replicas of the original images are needed (medical imagery), or where the objective is saving memory (photographic storage) [1]. JPEG2000 allows also accessing the compressed data in a random manner, giving the possibility of decoding certain areas of the image without the need of decoding the whole compressed data. This is very useful especially in multimedia applications, where users can edit the image by cropping, replacing, or scaling an area of interest. After the edition, the area of interest is newly encoded again and stored in the original compressed data. JPEG2000 is also optimized for information transmission in noisy environments and has the capacity to handle images up to 2³² x 2³² pixels, whereas JPEG goes only up to 2¹⁶ x 2¹⁶ pixels.

The advantages achieved by JPEG2000 require a more complex coding algorithm and a higher computational load than those found in JPEG. This has motivated several works that have focused to implement a JPEG2000 encoder as a coprocessor into an FPGA [2], [3], [4]. Complete software implementations of JPEG2000 standard have been developed, namely Kakadu and JasPer. JasPer is a software implementation of the JPEG2000 codec that is included in the Part 6 of the standard as reference software for test and validation of JPEG2000 systems, and unlike Kakadu, is available free of charge [5]. JasPer was designed especially for Windows and Unix-type systems. Although these software implementations are not as efficient as the hardware ones, they are more flexible and allow the designer to take full advantage of all the capabilities of the standard. Kakadu has been implemented successfully in an FPGA using hardware coprocessing with the NIOS II embedded processor [4]. JasPer has not been integrated in NIOS II systems, but is part of several image processing applications for desktops and laptops [6].

In this work, the JasPer implementation of the JPEG2000 encoder is adapted to run in a NIOS II embedded processor. This processor is synthesized in the Cyclone II EP2C70 of the DE2-70 board. The objective is to evaluate the performance of JasPer in a NIOS II processor, in order to obtain an efficient implementation of a JPEG2000 embedded system.

This paper is organized as follows: Section II introduces the JPEG2000 algorithm and explains its main features; Section III presents some of the JPEG2000 implementations designed for FPGAs; Section IV describes the system architecture used to evaluate the performance of JasPer in a NIOS II processor; Section V presents the experimental results using the image Lena.bmp; Conclusions and Future Work are presented in Section VI.
A JPEG2000 encoder compresses images in a lossless or a lossy manner using the same architecture, presented in Figure 1 [1]. The encoding process is divided basically in five stages.

A. Forward Multicomponent Transformation

If the image is too large, every one of its components is first partitioned into a number of non-overlapping rectangular blocks called tiles. Typical tile sizes are 256x256 or 512x512 pixels. This partition produces blocking artifacts in the reconstructed image at the tile boundaries as in JPEG compressed images. These artifacts are avoided if the image is not partitioned into tiles, but this requires a large amount of memory to store the processed data, especially during the DWT (Discrete Wavelet Transformation) calculation.

After tiling the image, the samples of every component are converted into two’s complement representation with a DC level shifting. The objective is to guarantee that the dynamic range of the samples is centered on zero. If desired, a multicomponent transformation is applied to reduce correlations amongst color components. For lossless and lossy compression, the RCT (Reversible Color Transformation) and the ICT (Irreversible Color Transformation) are used respectively. The latter is the RGB-YCbCr transformation used in the baseline JPEG algorithm.

B. Forward Wavelet Transform

To obtain the image frequency coefficients, the JPEG2000 standard utilizes the DWT instead of the DCT used in the JPEG standard. The DWT decomposes an image or a tile into four sub-bands (LL1, LH1, HL1, and HH1) at different levels of resolution. The sub-bands LH1, HL1, and HH1 contain the high frequency information of the image in the horizontal, vertical, and diagonal orientation respectively. The sub-band LL1 is a downsampled version of the original image and is further decomposed into four new sub-bands LL2, LH2, HL2, and HH2. The process can continue decomposing the sub-band LL2 into higher levels. For lossless compression, the wavelet filter bank used is the Le Gall (5, 3), whereas for lossy compression, the filter bank is the Daubechies (9, 7). The numbers indicate the size of the low-pass and high-pass FIR filters: The Daubechies bank is composed of a 9-tap low-pass filter and a 7-tap high-pass filter.

C. Quantization

In the lossy compression, the DWT coefficients are quantized using a uniform scalar quantization with dead-zone on the origin according to (1)

$$q_b(i,j) = \text{sign}(y_b(i,j)) \frac{|y_b(i,j)|}{\Delta_b}$$  \hspace{1cm} (1)

where $y_b(i,j)$ is a DWT coefficient in the sub-band $b$ and $\Delta_b$ is the quantization step size for sub-band $b$. The standard supports different quantization step sizes for each sub-band [1].

D. Tier-1 Encoder

The quantized coefficients are subsequently partitioned into blocks of size 32x32 or 64x64. Then, each block goes through a bit-plane encoding to reduce the number of bits needed to represent the compressed data. Every bit-plane is encoded using a fraction bit-plane coding (BPC) to produce a context value and a binary decision value for each position in three coding passes called the Significance Pass, the Refinement Pass, and the Clean-up Pass. The context value and the binary decision are used to generate the compressed codes corresponding to each block.

E. Tier-2 Encoder

The encoded data generated by the Tier-1 encoder is evaluated to find the optimal point of truncation in order to achieve a selected rate of compression. Thereby, only a subset of the encoded data from Tier-1 is included in the final code stream. If lossless compression is performed, all the encoded data is included [5].

F. Rate Control

The adjusting of the quantization step sizes and the selection of the truncation points for the code-block encoded data are known as the Rate Control process of the JPEG2000 encoder. This control allows getting the desired compression ratio for the final code stream achieving the higher PSNR (Peak-Signal-to-Noise Ratio) possible.

III. PREVIOUS WORKS

Several JPEG2000 encoders have been designed targeting FPGAs devices, taking advantage of their concurrent capabilities for intense computational tasks such as DWT and code block encoding.

A. Altera’s JPEG2000 Encoder

In this work, the DWT, quantization, and Tier-1 processes are executed in an FPGA, whereas the Tier-2 encoder runs as a software routine in a CPU. The design accelerates the code-block encoding process by using several entropy encoders in parallel whose inputs are buffered in the M-RAM blocks of an Altera’s Stratix II [2]. With eight entropy channels, this design processes a 3M-pixel image in 32 ms in a lossy manner, using 83% of the FPGA.
B. FPGA Co-processing in a JPEG2000 Implementation

In this thesis, the Tier-1 stage of Figure 1 is designed in VHDL targeting a Xilinx’s VirtexE 300K, leaving the other parts of the algorithm on a main processor such as a DSP. The JasPer software was taken as the framework of the JPEG2000 algorithm. Simulation results of the Tier-1 encoder reported a maximum frequency 54.9 MHz [3].

C. NIOS II Processor-Based Hardware/Software Co-Design of the JPEG2000 Standard

This design implements the BPC coding and the DWT calculations as dedicated hardware. The rest of the JPEG2000 encoder is executed as software on a NIOS II processor. The DWT is implemented using the lifting scheme, and DMA controllers are used to feed the code-block data into the BPC modules. This system uses the Kakadu software as framework.

IV. SYSTEM ARCHITECTURE

The system used for the evaluation of the JasPer software was implemented in a DE2-70 board. It is composed of a NIOS II fast core running at 100 MHz, an 8MB-SDRAM for data and program memory, an LCD controller for the Terasic’s LCD Panel, a 2MB-SSRAM for image-buffering, and a SD-Card interface (Figure 2).

The system reads a BMP file from a SD-Card using a library adapted from Terasic’s design examples for managing FAT16 files [7].

The RGB data extracted from this file is passed to the JPEG encoder implemented as a software library on the NIOS II processor. The library used is the jpeglib library, distributed by the Independent JPEG Group [8]. The same RGB data is also passed to the JasPer library, which generates the JPEG2000 encoded file.

The code streams generated by both encoders are passed to a host computer using the JTAG-UART module. This module is the same used to program and debug the FPGA. Additionally, the RGB data is received by the LCD controller from the NIOS II processor. This controller stores this data in the SSRAM using a back-buffering mode to display it in the Terasic’s 320x240 LCD panel. The LCD controller is written in VHDL and is adapted from the Terasic’s design examples [9].

The system uses 6350/68416 (9.3%) logic elements and 65427/1152000 (5.7%) memory bits of a Cyclone II EP2C70. The NIOS II core has a 4kB instruction cache, 2kB data cache, and a floating-point unit implemented as custom instructions inside its ALU.

V. EXPERIMENTAL RESULTS

To evaluate the performance of the JasPer software in the NIOS II system previously presented, the Lena.bmp image was used (Figure 3). This image had a 320x240 pixel resolution in order to be adjusted to the LCD panel resolution of the system.

The JasPer software was configured for six decomposition levels and code blocks of 64x64 samples. The performance was evaluated for different compression ratios. In the JasPer software, this was made by adjusting the rate parameter of the Rate Control. In the jpeglib library, the quality factor of the quantization tables had to be adjusted. For each compression ratio, the corresponding PSNR was evaluated for the image compressed with the JasPer software, as well as for the image compressed with the jpeglib library. The PSNR is one of the most commonly used quality metrics for compression techniques: The bigger the PSNR, the closer the reconstructed image is to the original one. The mathematical definition of PSNR for an 8-bit image is presented in (2).
\[
\text{PSNR} = 20 \log_{10} \left( \frac{255}{\sqrt{\sum_{i=1}^{M} \sum_{j=1}^{N} (I(i,j) - I'(i,j))^2}} \right)
\]

(2)

where \(M \times N\) is the dimension of the images, \(I\) is the original one, and \(I'\) is that reconstructed after the compression. To calculate the PSNR, the JasPer compressed image in the NIOS II was first converted to a BMP file using the command-line utility program \texttt{jasper} running on a computer desktop. Subsequently, this BMP file and the JPEG file compressed with the jpeglib in the NIOS II were converted to an YCbCr representation using Matlab. Finally, the luminance components were used to determine the PSNR of each image using Simulink blocks. Figure 4 summarizes the results.

According to Figure 4, the JPEG2000 compressed images have better PSNR than the images compressed using JPEG. This improvement is more evident at high compression ratios, where the difference is 4.56 dB when the ratio is 106:1, compared to a difference of 0.15 dB when the ratio is 8:1. The superiority of the JPEG2000 encoder over its counterpart JPEG is also clear when the reconstructed images from these encoders are evaluated visually. Figure 5 presents a region of these images for a compression ratio of 106:1. In this case, the JPEG2000 file is 2154 bytes long, whereas the JPEG file size is 2163 bytes. It is evident that at this compression ratio, the JPEG2000 image is still being similar to the original one. However, the JPEG image has lost a lot of information and the figure of Lena is almost vanished. Even more, the blocking artifacts in the JPEG image has become more visible, unlike the JPEG2000 image, where no blocking artifacts are present. This is because JPEG2000 uses a tile-transform (DWT), while on the contrary, JPEG uses an 8x8 pixel block transform (DCT).

The difference between the two encoders is not just noticeable at high compression ratios. Figure 6 presents the Lena’s eyes region for a compression of 16:1. In this case, the eyes (especially the right eyelashes) in the JPEG image become to be distorted in comparison with the eyes in the original image. In the JPEG2000 image, the eyes do not exhibit distortions, but some regions have a smoother appearance than they have in the original image, such as the right cheek.

In order to have a profile of the JasPer performance in the NIOS II system for future improvements, the elapsed time in the main stages was measured using the NIOS II timestamp timer. The measurements were made for lossy compression (compression ratio=8:1) and for lossless compression for five levels of decomposition (6 to 2). Table 1 presents the average times.

For the lossless compression, the encoder spends 0.964 s during the DWT calculation; at the contrary, this operation lasts 4.137 s in the lossy compression. This difference is because the wavelet filters used in lossless compression are smaller than those used in lossy compression: For lossless compression, the wavelet filter bank are a 5-tap low pass filter and a 3-tap high-pass, whereas for lossy compression, they are a 9-tap low pass filter and a 7-tap high-pass filter.
As expected, the quantization process is longer for the lossy compression than it is for the lossless compression. However, the time in the latter is not equal to zero because the quantization calculation is indeed performed but with a quantization step size equals to one.

In the case of the Tier-1 and the Tier-2 stages, the lossy compression spends less time in these calculations than in the case of lossless compression. This is understandable since these stages are entropy encoders and in lossless compression, the quantization does not reduce the entropy of the data since the DWT coefficients are only divided by one.

VI. CONCLUSIONS AND FUTURE WORK

The implementation of the JasPer software on an NIOS II processor has been presented. Its performance has been evaluated against the software JPEG encoder included in the jpeglib from the Independent JPEG Group. The PSNR calculated for the resultant images from both encoders using Lena.bmp showed that JPEG2000 images have fewer distortions than its counterpart, even at very high compression ratios. In particular, at a compression ratio of 106:1, the JPEG2000 image do not loss most of the shapes or colors of the original image, whereas the JPEG image have noticeable distortions including the blocking artifacts from the block-based DCT.

Time measurements showed that DWT and Tier-1 are the more time-consuming stages for both lossless and lossy compression using the JasPer encoder; therefore, they are the first stages that must be translated into hardware in order to optimize the encoder performance. In the case of the DWT, the best implementation is the lifting scheme according to previous works. Due to its bit-serial characteristics, the Tier-1 stage is more complex and entropy encoders in parallel must be used to take advantage of the concurrent capabilities of the FPGA.

REFERENCES