INTEGRATION, the VLSI journal 53 (2016) 145–156

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Power-performance enhancement of two-dimensional RNS-based DWT image processor using static voltage scaling

Azadeh Safari *, Cheecottu Vayalil Niras, Yinan Kong

Department of Engineering, Faculty of Science and Engineering, Macquarie University, Australia

ARTICLE INFO

Article history:
Received 31 May 2015
Received in revised form
9 December 2015
Accepted 16 December 2015
Available online 13 January 2016

Keywords:
High-speed arithmetic
Residue number system
Discrete wavelet transform
Low-power design
Image compression
Multi-voltage processor

ABSTRACT

Digital image processing is widely used in fast and high-performance applications. The high speed and functional requirements of such applications, however, lead to increased power consumption. Hence, finding a way to solve the power-performance issues is of great importance. In this paper, we present the power-performance enhancement of a two-dimensional (2D) discrete wavelet transform (DWT) image processor using the residue number system (RNS) and the static voltage scaling (SVS) scheme. The aim of this paper is to investigate the effects of the RNS and SVS scheme on the proposed image processor. The original contributions of the proposed design include a low-complexity hardware architecture of the RNS-based filter banks, optimized transposition units and exploiting the SVS scheme to reduce the power consumption. The multiplierless scheme of the RNS-based filter banks and the binary-coded number format are used to save on hardware complexity, while modular arithmetics and 6-bit dyadic fraction filter coefficients are applied to improve the system performance. The bi-orthogonal discrete wavelet transform CDF97 is chosen to compress the images due to its multi-resolution features and its ability to localize finite signals. The proposed design has been synthesized using the generic library *SAED90nmEDK* with the Synopsys Design Compiler (DC).

1. Introduction

One of the most significant current discussions in digital image processing is about finding a fast and efficient technique for the storage and transmission of images. Various methods have been developed and introduced to optimize existing systems. Among proposed schemes, optimizing the arithmetic level of image processors has drawn more attention, since it plays an important role in satisfying the requirement for a large volume of computational operations in image processing. Replacing conventional binary systems with the Residue Number System (RNS) has been steadily rising in favor over the last 50–60 years. The RNS-based architecture allows the processing of modular channels simultaneously, and saves significant delays in arithmetic operations. Using small integers in independent channels also reduces the carry propagation and number of partial products in adders and multipliers, respectively [1,2]. There are many RNS-based designs for general and specialized processors in the literature [3,4]. The design and implementation of an RNS-based image processor using DWT filter banks and RNS arithmetic is proposed in [3]. The proposed design used 27 look-up tables (LUT) for modular arithmetic. Each LUT had an 8-bit width and 256 entries to store all the results of modular multiplication. The downside of that processor is that LUTs and RAMs are main sources of leakage power, which is the major concern of standalone applications like mobile phones and cameras. On the other hand, [4] has reported that the best RNS designs have a hardware cost about the same as or more than the binary designs. In other words, while using RNS can help to enhance the performance of an image processor, the architecture of using modular arithmetic might increase the power consumption of the system [5]. Therefore, one question that needs to be asked is what aspects of RNS processors are superior to binary designs. This paper will provide a design and optimization of a RNS-based digital image processor in detail and examine it against a binary processor. Details of proposed residue arithmetic units are explained in such a way that the novelty can be appreciated. The study will also implement the proposed processor with static voltage scaling to achieve the best power-performance trade-off in the proposed image processor.

2. Preliminaries

This section provides background information and preliminaries on the residue number system, the discrete wavelet

* Corresponding author.
E-mail addresses: azadehsafari2008@gmail.com (A. Safari), cvniras@gmail.com (C.V. Niras), yinan.kong@mq.edu.au (Y. Kong).

http://dx.doi.org/10.1016/j.vlsi.2015.12.006
0167-9260/© 2016 Elsevier B.V. All rights reserved.

transform and the static voltage scaling method that are directly relevant and useful for designing and implementing the proposed processor.

2.1. The residue number system

Each number (X) in the RNS can be represented as a set of the least positive remainders when it is divided by the set of the moduli (moduli set). The moduli set is shown as $(m_1, m_2, \ldots, m_i)$ where m_i is the ith modulus, and each pair of moduli are relatively prime. The residue set is commonly shown as $(r_1, r_2, \ldots, r_i)$ where r_i is the ith residue. The residue of (X) can be calculated based on the congruence [6,7]:

$$r_i = \begin{cases} X \bmod m_i, & X \geq 0 \\ (m_i - |X|) \bmod m_i, & X < 0 \end{cases} \tag{1}$$

Each RNS-based system has a *dynamic range* that is the total number of different values that can be represented using that set of moduli [6].

For the residue set of $X = (x_1, x_2, \ldots, x_n)$ and $Y = (y_1, y_2, \ldots, y_n)$, the arithmetic operation is performed independently on each residue, as follows:

$$|XoY|_M = (|x_1 o y_1|_{m_1}, |x_2 o y_2|_{m_2}, \ldots, |x_n o y_n|_{m_n}) \tag{2}$$

where (o) represents addition, subtraction or multiplication. Eq. (2) shows the primary advantage of using small residues instead of large numbers. It also shows how the carry-propagation problem is solved in the RNS by limiting it within a single residue [8].

Moduli-set selection has a direct impact on the performance of the residue system. Selecting an efficient moduli set for a specific dynamic range improves the bit efficiency [9]. In this paper, the moduli set $(2^n - 1, 2^n, 2^n + 1)$ is selected, being very popular when applying RNS for image processing due to its simplicity and the design efficiency of functional and conversional units [10–12]. The selected moduli set has been suggested as “the most standard and widely used” moduli set in [13]. Also, the authors in [14] have compared the selected moduli set with “general moduli sets” in terms of the speed and hardware complexity. They compared the area and speed with four general moduli sets for 8, 16, 32 and 64-bit ranges and concluded that the reverse converter of the selected moduli set “is the fastest and requires the least amount of data”.

In terms of the bit efficiency of the selected moduli set, [15] suggests that, for the medium dynamic range (21 bits or less), the most efficient moduli set is the selected moduli set; however for large dynamic ranges (22 bits or more) we cannot use a three-moduli set any more and it should be of the form $(2^{n_1}, 2^{n_1}+1, 2^{n_1}-1, 2^{n_2} \pm 1, \ldots, 2^{n_i} \pm 1)$. Later on, they cited [16] and concluded that “the upper bound of the dynamic range for using three-moduli sets is around 24 bits”. In this paper, for the selected three-moduli set and $n = 8$, the dynamic range (M) has 16,776,960 different values, or 24 bits, which is in the suggested range. For the moduli set $(2^n - 1, 2^n, 2^n + 1) = (255, 256, 257)$, the total number of bits required for the different arithmetic blocks are calculated using Eq. (3). It shows that, for the selected moduli set, each operating block should use 25 bits to avoid overflow:

$$\lceil \log_2 m_1 \rceil + \lceil \log_2 m_2 \rceil + \lceil \log_2 m_3 \rceil = 8 + 8 + 9 = 25 \tag{3}$$

2.2. Discrete wavelet transform (DWT)

The discrete wavelet transform (DWT) is capable of fast image compression with less area and low power consumption. It has shown excellent performance in digital image compression and de-noising applications such as source encoding in the JPEG2000 still-image compression standard and in FBI wavelet scalar quantization [17–19].

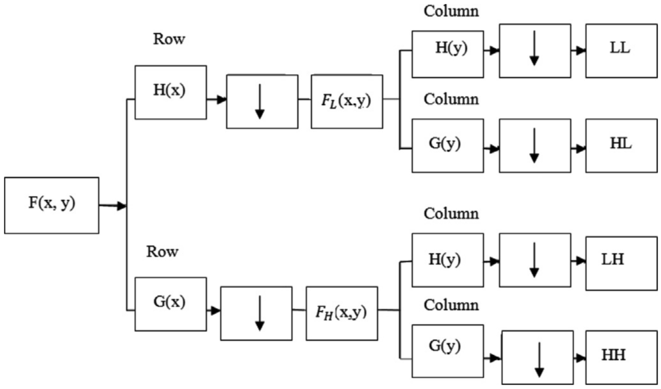

Fig. 1. One-level signal decomposition using 2D DWT.

In DWT, an image is decomposed by passing through an analysis filter bank which consists of a highpass and a lowpass filter at each level. After each level of decomposition, an approximation signal with the resolution reduced by a factor of 2 (LL) and detail signals (LH, HL and HH) are obtained. If further decomposition is required, the previous level's lower-resolution approximation signal (LL) becomes the next-level sub-sampling input, and its related detail signal is stored after filtering at each level [20,21]. Fig. 1 shows one-level signal decomposition using 2D DWT.

DWT can be implemented using the pyramid algorithm (an octave-band filter bank with j levels) in the multi-resolution analysis framework. For a signal sequence of $x(n)$, the approximation (a_n^i) and detail (d_n^i) signals at level i are defined using Eqs. (4) and (5), respectively [22]:

$$a_n^i = \sum_{k=0}^{N-1} h_k a_{2n-k}^{(i-1)} \tag{4}$$

$$d_n^i = \sum_{k=0}^{N-1} g_k a_{2n-k}^{(i-1)} \tag{5}$$

where h_k and g_k are lowpass and highpass coefficients selected based on the chosen wavelet family, n is the filter length, k is the filter coefficient length and $i = 1, 2, \ldots, j$.

Removing high-frequency components of images using DWT improves the quality of images. One reason for achieving better-quality images by removing high-frequency components is that luminance (brightness) or low-frequency components are more important than chrominance (colour difference) or high-frequency components to provide a fine-quality image. Another advantage of using DWT in image processing is that, in multi-resolution image processing, only the approximation signal (LL) will be stored and used as the next-level input. This feature helps to reduce hardware complexity. Having noted the popularity and advantages of using DWT, it is also important to choose a correct wavelet for a specific application. Among the DWT family, bi-orthogonal DWT uses a linear-phase filter, which solves the common problems of image compression such as coefficient expansion, border artifacts, image blurring or spatial dislocations. In addition, BDWT filters can be designed to have integer coefficients, and multiplications can be implemented using only register shift and addition. The capability of bi-orthogonal wavelets with symmetric extension is another reason for their superior performance [23,24]. Table 1 shows the CDF97 floating-point filter coefficients.

2.3. Static voltage scaling (SVS) method

There are various low-power design (LPD) approaches to reduce total and leakage power, from the early steps of circuit design to the very last steps of system implementation. The most