12

Study on Low-Power Image Processing for Gastrointestinal Endoscopy

Meng-Chun Lin
Department of Computer National Chengchi University
Taiwan

1. Introduction

Gastrointestinal (GI) endoscopy has been popularly applied for the diagnosis of diseases of the alimentary canal including Crohn's Disease, Celiac disease and other malabsorption disorders, benign and malignant tumors of the small intestine, vascular disorders and medication related small bowel injury. There are two classes of GI endoscopy; wired active endoscopy and wireless passive capsule endoscopy. The wired active endoscopy can enable efficient diagnosis based on real images and biopsy samples; however, it causes discomfort for the patients to push flexible, relatively bulky cables into the digestive tube. To relief the patients' discomfort, wireless passive capsule endoscopes are being developed worldwide (1)-(6).

The capsule moves passively through the internal GI tract with the aid of peristalsis and transmits images of the intestine wirelessly. Developed by Given Imaging Ltd., the PillCam capsule is a state-of-the-art commercial wireless capsule endoscope product. The PillCam capsule transmits the GI images at a resolution of 256-by-256 8-bit pixels and the frame rate of 2 frames/sec (or fps). Because of its high mobility, it has been successfully utilized to diagnose diseases of the small intestine and alleviate the discomfort and pain of patients. However, based on clinical experience; the PillCam still has some drawbacks. First, the PillCam cannot control its heading and moving direction itself. This drawback may cause image oversights and overlook a disease. Second, the resolution of demosaicked image is still low, and some interesting spots may be unintentionally omitted. Therefore, the images will be severely distorted when physicians zoom images in for detailed diagnosis. The first drawback is the nature of passive endoscopy. Some papers have presented approaches for the autonomous moving function (7; 8; 21; 22; 25). Very few papers address solutions for the second drawback. Increasing resolution may alleviate the second problem; however, it will result in significant power consumption in RF transmitter. Hence, applying image compression is necessary for saving the power dissipation of RF transmitter (9)-(14), (23), (24), (26).

Our previous work (11) has presented an ultra-low-power image compressor for wireless capsule endoscope. It helps the endoscope to deliver a compressed 512-by-512 image, while the RF transmission rate is at 1 megabits ($(256 \times 256 \times 2 \times 8)/1024^2$) per second. No any references can clearly define how much compression is allowed in capsule endoscope application. We define that the minimum compression rate is 75% according to two considerations for our capsule endoscope project. The first consideration is that the new

image resolution (512-by-512) that is four times the one (256-by-256) of the PillCam can be an assistant to promote the diagnosis of diseases for doctors. The other one is that we do not significantly increase the power consumption for the RF circuit after increasing the image resolution from the sensor. Instead of applying state-of-the-art video compression techniques, we proposed a simplified image compression algorithm, called GICam, in which the memory size and computational load can be significantly reduced. The experimental results shows that the GICam image compressor only costs 31K gates at 2 frames per second, consumes 14.92 mW, and reduces the image size by at least 75% .

In applications of capsule endoscopy, it is imperative to consider the tradeoffs between battery life and performance. To further extend the battery life of a capsule endoscope, we herein present a subsample-based GICam image compressor, called GICam-II. The proposed compression technique is motivated by the reddish feature of GI image. We have previously proposed the GICam-II image compressor in paper (20). However, the color importance of primary colors in GI images has no quantitative analysis in detail because of limited pages. Therefore, in this paper, we completely propose a series of mathematical statistics to systematically analyze the color sensitivity in GI images from the RGB color space domain to the 2-D DCT spatial frequency domain in order to make up for a deficiency in our previous work (20). This paper also refines the experimental results to analyze the performance about the compression rate, the quality degradation and the ability of power saving individually.

As per the analysis of color sensitivity, the sensitivity of GI image sharpness to red component is at the same level as the sensitivity to green component. This result shows that the GI image is cardinal and different from the general image, whose sharpness sensitivity to the green component is much higher than the sharpness sensitivity to the red component. Because the GICam-II starts compressing the GI image from the Bayer-patterned image, the GICam-II technique subsamples the green component to make the weighting of red and green components the same. Besides, since the sharpness sensitivity to the blue component is as low as 7%, the blue component is down-sampled by four. As shown in experimental results, with the compression ratio as high as 4:1, the GICam-II can significantly save the power dissipation by 38.5% when compared with previous GICam work (11) and 98.95% when compared with JPEG compression, while the average PSNRY is 40.73 dB. The rest of the paper is organized as follows. Section II introduces fundamentals of GICam compression and briefs the previous GICam work. Section III presents the sensitivity analysis of GICam image and shows the importance of red component in GI image. In Section IV, the GICam-II compression will be described in details. Then, Section V illustrates the experimental results in terms of compression ratio, image quality and power consumption. Finally, Section VI concludes our contribution and merits of this work.

Except using novel ultra-low-power compression techniques to save the power dissipation of RF transmitter in high-resolution wireless gastrointestinal endoscope systems. How to efficiently eliminate annoying impulsive noise caused by a fault sensor and enhance the sharpness is necessary for gastrointestinal (GI) images in wired/wireless gastrointestinal endoscope systems. To overcome these problems, the LUM filter is the most suitable candidate because it simultaneously has the characteristics of smoothing and sharpening. In the operational procedure of LUM filter, the mainly operational core is the rank-order filtering (ROF) and the LUM filter itself needs to use different kind of rank values to accomplish

the task of smoothing or sharpening. Therefore, we need a flexible ROF hardware to arbitrarily select wanted rank values into the operation procedure of LUM filter and we have proposed an architecture based on a maskable memory for rank-order filtering. The maskable memory structure, called dual-cell random-access memory (DCRAM), is an extended SRAM structure with maskable registers and dual cells. This dissertation is the first literature using maskable memory to realize ROF. Driving by the generic rank-order filtering algorithm, the memory-based architecture features high degree of flexibility and regularity while the cost is low and the performance is high. This architecture can be applied for arbitrary ranks and a variety of ROF applications, including recursive and non-recursive algorithms. Except efficiently eliminating annoying impulsive noises and enhance sharpness for GI images, the processing speed of ROF can also meet the real-time image applications.

2. GICam image compressor

2.1 The review of GICam image compression algorithm

Instead of applying state-of-the-art video compression techniques, we proposed a simplified image compression algorithm, called GICam. Traditional compression algorithms employ the YCbCr quantization to earn a good compression ratio while the visual distortion is minimized, based on the factors related to the sensitivity of the human visual system (HVS). However, for the sake of power saving, our compression rather uses the RGB quantization (15) to save the computation of demosaicking and color space transformation. As mentioned above, the advantage of applying RGB quantization is two-fold: saving the power dissipation on preprocessing steps and reducing the computing load of 2-D DCT and quantization. Moreover, to reduce the hardware cost and quantization power dissipation, we have modified the RGB quantization tables and the quantization multipliers are power of two's. In GICam, the Lempel-Ziv (LZ) coding (18) is employed for the entropy coding. The reason we adopted LZ coding as the entropy coding, is because the LZ encoding does not need look-up tables and complex computation. Thus, the LZ encoding consumes less power and uses smaller silicon size than the other candidates, such as the Huffman encoding and the arithmetic coding. The target compression performance of the GICam image compression is to reduce image size by at least 75%. To meet the specification, given the quantization tables, we exploited the cost-optimal LZ coding parameters to meet the compression ratio requirement by simulating with twelve tested endoscopic pictures shown in Fig.3.

When comparing the proposed image compression with the traditional one in (11), the power consumption of GICam image compressor can save 98.2% because of the reduction of memory requirement. However, extending the utilization of battery life for a capsule endoscope remains an important issue. The memory access dissipates the most power in GICam image compression. Therefore, in order to achieve the target of extending the battery life, it is necessary to consider how to efficiently reduce the memory access.

2.2 Analysis of sharpness sensitivity in gastrointestinal images

2.2.1 The distributions of primary colors in the RGB color space

In the modern color theory (16; 17), most color spaces in used today are oriented either toward hardware design or toward product applications. Among these color spaces, the

RGB(red, green, blue) space is the most commonly used in the category of digital image processing; especially, broad class of color video cameras and we consequently adopt the RGB color space to analyze the importance of primary colors in the GI images. In the RGB color space, each color appears in its primary spectral components of red, green and blue. The RGB color space is based on a Cartesian coordinate system, in which, the differ colors of pixels are points on or inside the cube based on the triplet of values (R, G, B). Due to this project was supported in part by Chung-Shan Institute of Science and Technology, Taiwan, under the project BV94G10P. The responsibility of Chung-Shan Institute of Science and Technology mainly designs a 512-by-512 raw image sensor. The block-based image data can be sequentially outputted via the proposed locally-raster-scanning mechanism for this raw image sensor. The reason for adopting a novel image sensor without using generally conventional ones is to efficiently save the size of buffer memory. Conventional raw image sensors adopt the raster-scanning mechanism to output the image pixels sequentially, but they need large buffer memory to form each block-based image data before executing the block-based compression. However, we only need a small ping-pong type memory structure to directly save the block-based image data from the proposed locally-raster-scanning raw image sensor. The structure of this raw image sensor is shown in Fig.1 (a) and the pixel sensor architecture for the proposed image sensor is shown in Fig.1 (b). In order to prove the validity for this novel image sensor before the fabrication via the Chung-Shan Institute of Science and Technology, the chip of the 32-by-32 locally-raster-scanning raw image sensor was designed by full-custom CMOS technology and this chip is submitted to Chip Implementation Center (CIC), Taiwan, for the fabrication. Fig.2 (a) and Fig.2 (b) respectively shows the chip layout and the package layout with the chip specification. The advantage of this novel CMOS image sensor can save the large area of buffer memory. The size of buffer memory can be as a simple ping-pong memory structure shown in Fig.9 while executing the proposed image algorithm, a novel block coding. Our research only focuses on developing the proposed image compressor and other components are implemented by other research department for the GICam-II capsule endocopy. Therefore, the format of the GI image used in the simulation belongs to a raw image from the 512-by-512 sensor designed by Chung-Shan Institute of Science and Technology. In this work, we applied twelve GI images captured shown in Fig.3 for testcases to evaluate the compression technique. The distribution of GI image pixels in the RGB color space is non-uniform. Obviously, the GI image is reddish and the pixels are amassed to the red region. Based on the observation in the RGB color space, the majority of red values are distributed between 0.5 and 1 while most of the green and blue values are distributed between 0 and 0.5 for all tested GI images. To further analyze the chrominance distributions and variations in the RGB color space for each tested GI images, two quantitative indexes are used to quantify these effects. The first index is to calculate the average distances between total pixels and the maximum primary colors in each GI image, and the calculations are formulated as Eq.1, Eq.2 and Eq.3. First, Eq.1 defines the the average distance between total pixels and the most red color ($\overline{R}$), in which, $R(i,j)$ means the value of red component of one GI image at (i,j) position and the value of most red color (R_{max}) is 255. In addition, M and N represent the width and length for one GI image, respectively. The M is 512 and the N is 512 for twelve tesed GI images in this work. Next, Eq.2 also defines the average distance between total pixels and the most green color ($\overline{G}$) and the value of most green one (G_{max}) is 255. Finally, Eq.3 defines the average distance between total pixels and the most blue color ($\overline{B}$) and the value of most blue color (B_{max}) is 255. Table 1 shows the statistical results of $\overline{R}$, $\overline{G}$ and $\overline{B}$

for all tested GI images. From Table 1, the results clearly show that $\overline{R}$ has the shortest average distance. Therefore, human eyes can be very sensitive to the obvious cardinal ingredient on all surfaces of tested GI images. Moreover, comparing $\overline{G}$ with $\overline{B}$, $\overline{G}$ is shorter than $\overline{B}$ because $\overline{G}$ contributes larger proportion in luminance.

$$\begin{aligned}\overline{R} &= E[(1-\frac{R(i,j)}{R_{max}})] \\ &= (\frac{1}{M\times N})\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(1-\frac{R(i,j)}{R_{max}})\end{aligned} \tag{1}$$

$$\begin{aligned}\overline{G} &= E[(1-\frac{G(i,j)}{G_{max}})] \\ &= (\frac{1}{M\times N})\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(1-\frac{G(i,j)}{G_{max}})\end{aligned} \tag{2}$$

$$\begin{aligned}\overline{B} &= E[(1-\frac{B(i,j)}{B_{max}})] \\ &= (\frac{1}{M\times N})\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(1-\frac{B(i,j)}{B_{max}})\end{aligned} \tag{3}$$

The first index has particularly quantified the chrominance distributions through the concept of average distance, and the statistical results have also shown the reason the human eyes can sense the obvious cardinal ingredient for all tested GI images. Next, the second index is to calculate the variance between total pixels and average distance, in order to further observe

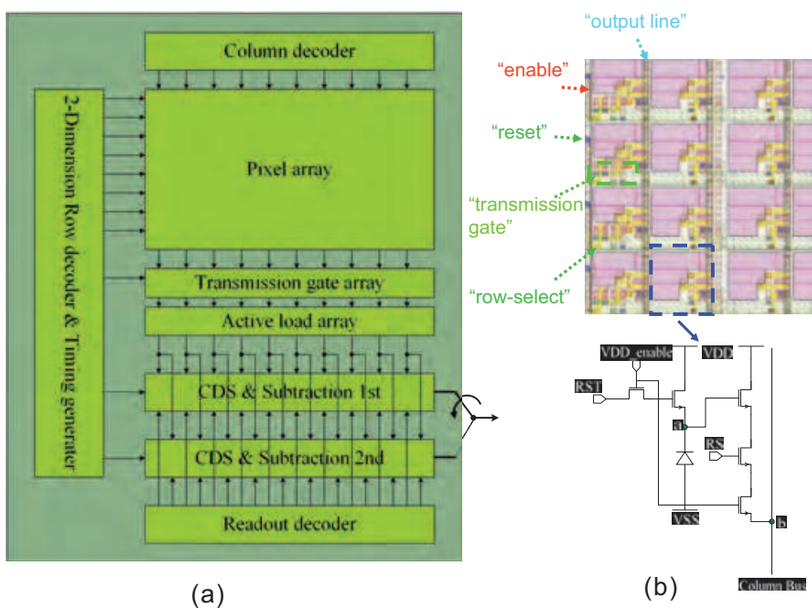
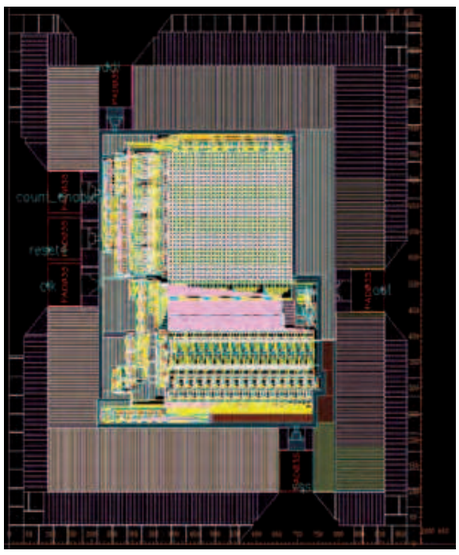
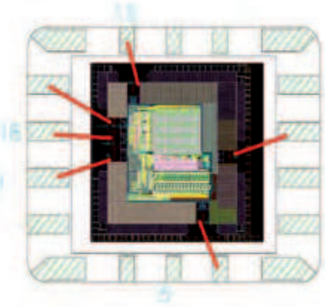


Fig. 1. (a) The structure of locally-raster-scanning raw image sensor (b)The pixel sensor architecture for the locally-raster-scanning raw image sensor.





Technology	0.35 um
Voltage	3.3 V
Sensor Array Size	32-by-32
Power Consumption	8.8586 mW
Chip Size	1.000651.01845 mm^2
Output	Analog output

(a) (b)

Fig. 2. (a)The chip layout of the locally-raster-scanning raw image sensor (b)The package layout and the chip specification of the locally-raster-scanning raw image sensor.

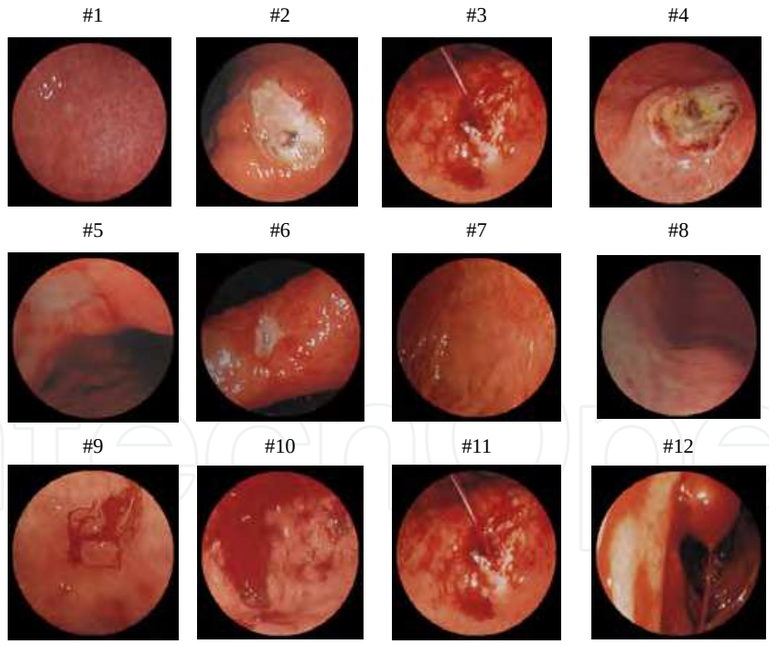


Fig. 3. The twelve tested GI images.

the color variations in GI images, and the calculations are formulated as Eq.4, Eq.7 and Eq.10. The Table 2 shows that the average variation of red signal is 0.09, the average variance of green one is 0.03, and the average variance of blue one is 0.02. It signifies that the color information of red signal must be preserved carefully more than other two primary colors; green and blue,

Average Distance			
Test Picture ID	$\overline{R}$	$\overline{G}$	$\overline{B}$
1	0.58	0.80	0.82
2	0.55	0.74	0.79
3	0.54	0.81	0.86
4	0.55	0.76	0.81
5	0.66	0.82	0.85
6	0.66	0.84	0.87
7	0.59	0.82	0.88
8	0.68	0.81	0.83
9	0.55	0.80	0.85
10	0.53	0.81	0.84
11	0.53	0.81	0.86
12	0.62	0.80	0.85
Average	0.59	0.80	0.84

Table 1. The Analysis of Average Distance.

for GI images because the dynamic range of red signal is broader than green and blue ones. In addition, the secondary is green signal and the last is blue signal.

$$VAR_R = E[(1 - \frac{R(i,j)}{R_{max}})^2] - \{E[(1 - \frac{R(i,j)}{R_{max}})]\}^2 \tag{4}$$

$$= (\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [1 - \frac{R(i,j)}{R_{max}}]^2 - \tag{5}$$

$$[(\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (1 - \frac{R(i,j)}{R_{max}})]^2 \tag{6}$$

$$VAR_G = E[(1 - \frac{G(i,j)}{G_{max}})^2] - \{E[(1 - \frac{G(i,j)}{G_{max}})]\}^2 \tag{7}$$

$$= (\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [1 - \frac{G(i,j)}{G_{max}}]^2 - \tag{8}$$

$$[(\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (1 - \frac{G(i,j)}{G_{max}})]^2 \tag{9}$$

$$VAR_B = E[(1 - \frac{B(i,j)}{B_{max}})^2] - \{E[(1 - \frac{B(i,j)}{B_{max}})]\}^2 \tag{10}$$

$$= (\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [1 - \frac{B(i,j)}{B_{max}}]^2 - \tag{11}$$

$$[(\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (1 - \frac{B(i,j)}{B_{max}})]^2 \tag{12}$$

Variance of Distance			
Test Picture ID	VAR_R	VAR_G	VAR_B
1	0.08	0.02	0.02
2	0.11	0.05	0.03
3	0.10	0.03	0.02
4	0.10	0.04	0.02
5	0.07	0.02	0.01
6	0.08	0.02	0.01
7	0.09	0.02	0.01
8	0.06	0.02	0.02
9	0.09	0.03	0.01
10	0.10	0.03	0.02
11	0.10	0.03	0.02
12	0.10	0.04	0.02
Average	0.09	0.03	0.02

Table 2. The Analysis of Variance.

2.2.2 The analysis of sharpness sensitivity to primary colors for gastrointestinal images

Based on the analysis of RGB color space, the importance of chrominance is quantitatively demonstrated for GI images. Except for the chrominance, the luminance is another important index because it can efficiently represent the sharpness of an object. Eq.13 is the formula of luminance (Y) and the parameters ; a1, a2 and a3 are 0.299, 0.587 and 0.114 respectively.

$$Y = a1 \times R + a2 \times G + a3 \times B \tag{13}$$

To efficiently analyze the importance of primary colors in the luminance, the analysis of sensitivity is applied. Through the analysis of sensitivity, the variation of luminance can actually reflect the influence of each primary colors. Eq.14, Eq.15 and Eq.16 define the sensitivity of red ($S_{Y_{i,j}}^{R_{i,j}}$), the sensitivity of green ($S_{Y_{i,j}}^{G_{i,j}}$), and the sensitivity of blue ($S_{Y_{i,j}}^{B_{i,j}}$) at position (i,j), respectively for a color pixel of a GI image.

$$S_{Y_{i,j}}^{R_{i,j}} = \frac{\Delta Y_{i,j}/Y_{i,j}}{\Delta R_{i,j}/R_{i,j}} = \frac{R_{i,j}}{Y_{i,j}} \times \frac{\Delta Y_{i,j}}{\Delta R_{i,j}} = \frac{a1 \times R_{i,j}}{Y_{i,j}} \tag{14}$$

$$S_{Y_{i,j}}^{G_{i,j}} = \frac{\Delta Y_{i,j}/Y_{i,j}}{\Delta G_{i,j}/G_{i,j}} = \frac{G_{i,j}}{Y_{i,j}} \times \frac{\Delta Y_{i}}{\Delta G_{i,j}} = \frac{a2 \times G_{i,j}}{Y_{i,j}} \tag{15}$$

$$S_{Y_{i,j}}^{B_{i,j}} = \frac{\Delta Y_{i,j}/Y_{i,j}}{\Delta B_{i,j}/B_{i,j}} = \frac{B_{i,j}}{Y_{i,j}} \times \frac{\Delta Y_{i}}{\Delta B_{i,j}} = \frac{a3 \times B_{i,j}}{Y_{i,j}} \tag{16}$$

After calculating the sensitivity of each primary colors for a GI image, the average sensitivity of red ($\overline{S_Y^R}$), the average sensitivity of green ($\overline{S_Y^G}$), and the average sensitivity of blue ($\overline{S_Y^B}$) are calculated by Eq.17, Eq.18 and Eq.19 for each GI images. M and N represent the width and length for a GI image, respectively. Table 3 shows the average sensitivities of red, green and blue for all tested GI images. From the calculational results, the sensitivity of blue is the slightest and hence the variation of luminance arising from the aliasing of blue is very

invisible. In addition to the sensitivity of blue, the sensitivity of red is close to the one of green and thus they both have a very close influence on the variation of luminance.

$$\overline{S_Y^R} = (\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} S_{Y_{i,j}}^{R_{i,j}} \tag{17}$$

$$\overline{S_Y^G} = (\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} S_{Y_{i,j}}^{G_{i,j}} \tag{18}$$

$$\overline{S_Y^B} = (\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} S_{Y_{i,j}}^{B_{i,j}} \tag{19}$$

To sum up the variance of chrominance and the sensitivity of luminance, blue is the most insensitive color in the GI images. Therefore, the blue component can be further downsampled without significant sharpness degradation. Moreover, comparing the red signal with the green signal, they both have a very close influence on the variation of luminance, because they have very close sensitivities. However, the chrominance of red varies more than the chrominance of green and hence the information completeness of red has higher priority than the green. Because the proposed compression coding belongs to the DCT-based image coding, the coding is processed in the spatial-frequency domain. To let the priority relationship between red and green also response in the spatial-frequency domain, the analysis of alternating current (AC) variance will be accomplished to demonstrate the inference mentioned above in the next subsection.

The Sensitivity of Primary Colors in Luminance			
Test Picture ID	$\overline{S_Y^R}$	$\overline{S_Y^G}$	$\overline{S_Y^B}$
1	0.49	0.43	0.08
2	0.44	0.48	0.08
3	0.55	0.39	0.06
4	0.47	0.46	0.07
5	0.45	0.47	0.08
6	0.48	0.45	0.07
7	0.52	0.42	0.06
8	0.44	0.48	0.08
9	0.51	0.43	0.06
10	0.54	0.40	0.06
11	0.55	0.39	0.06
12	0.49	0.44	0.07
Average	0.49	0.44	0.07

Table 3. The Analysis of Average Sensitivities.

2.2.3 The analysis of AC variance in the 2-D DCT spatial frequency domain for gastrointestinal images

According to the analysis results from the distributions of primary colors in the RGB color space and the proportion of primary colors in the luminance for GI images, the red signal

plays a decisive role in the raw image. The green signal plays a secondary role and the blue signal is very indecisive. To verify the validity of observation mentioned above, we first use the two-dimensional (2-D) 8×8 discrete cosine transform (DCT) to transfer the spatial domain into the spatial-frequency domain for each of the components, R, G1, G2 and B. The 2-D 8×8 DCT transformation can be perceived as the process of finding for each waveform in the 2-D 8×8 DCT basic functions and also can be formulated as Eq.20, Eq.21, Eq.22, Eq.23 and Eq.24 for each 8×8 block in R, G1, G2 and B subimages respectively. M and N represent the width and length for one GI image respectively. k, l=0, 1, ..., 7 and y_{kl} is the corresponding weight of DCT basic function in the kth row and the lth column. P represents the total number of pictures and B represents the total number of 8×8 blocks in the GI images.

$$R_{pb}(kl) = \frac{c(k)}{2}\sum_{i=0}^{7}[\frac{c(l)}{2}\sum_{j=0}^{7} r_{ij}cos(\frac{(2j+1)l\pi}{16})]cos(\frac{(2i+1)k\pi}{16}) \tag{20}$$

$$G_{pb}(kl) = \frac{c(k)}{2}\sum_{i=0}^{7}[\frac{c(l)}{2}\sum_{j=0}^{7} g_{ij}cos(\frac{(2j+1)l\pi}{16})]cos(\frac{(2i+1)k\pi}{16}) \tag{21}$$

$$B_{pb}(kl) = \frac{c(k)}{2}\sum_{i=0}^{7}[\frac{c(l)}{2}\sum_{j=0}^{7} b_{ij}cos(\frac{(2j+1)l\pi}{16})]cos(\frac{(2i+1)k\pi}{16}) \tag{22}$$

$$c\,(k) = \begin{cases} \frac{1}{\sqrt{2}}, \; if\; k = 0 \\ 1, \; otherwise. \end{cases} \tag{23}$$

$$c\,(l) = \begin{cases} \frac{1}{\sqrt{2}}, \; if\; l = 0 \\ 1, \; otherwise. \end{cases} \tag{24}$$

Next, we calculate the average energy amplitude of all alternating current (AC) coefficients of all tested GI images, in order to observe the variation of energy for each of the components R, G1, G2 and B, and the calculations are formulated as Eq.25, Eq.26, Eq.27.

$$A_R(kl) = \frac{1}{P}\sum_{p=1}^{P}[\sum_{b=0}^{B-1}|R_{pb}(kl)|] \tag{25}$$

$$A_G(kl) = \frac{1}{P}\sum_{p=1}^{P}[\sum_{b=0}^{B-1}|G_{pb}(kl)|] \tag{26}$$

$$A_B(kl) = \frac{1}{P}\sum_{p=1}^{P}[\sum_{b=0}^{B-1}|B_{pb}(kl)|] \tag{27}$$

After calculating the average energy amplitude, we convert the 2-D DCT domain into one-dimensional (1-D) signal distribution in order to conveniently observe the variation of frequency. Consequently, a tool for transforming two-dimensional signals into one dimension is needed. There are many schemes to convert 2-D into 1-D, including row-major scan, column-major scan, peano-scan, and zig-zag scan. Majority of the DCT coding schemes adopt zig-zag scan to accomplish the goal of conversion, and we use it here. The benefit of zig-zag is its property of compacting energy to low frequency regions after discrete cosine transformation. The arrangement sorts the coefficients from low to high frequency, and

Fig.4(a) shows the zig-zag scanning order for 8×8 block. Fig.4(b) shows the 1-D signal distribution after zig-zag scanning order and Fig.4(c) shows the symmetric type of frequency for the 1-D signal distribution.

Through the converting method of Fig.4, the 1-D signal distributions of each R, G1, G2, B components are shown in Fig.5. The variances of frequency are 1193, 1192, 1209 and 1244 for G1, G2, R and B respectively, and the variance of R is very close to the ones of G1 and G2 from the result. However, the datum of G are twice the datum of R based on the Bayer pattern and hence, the datum of G can be reduced to half at the most. Based

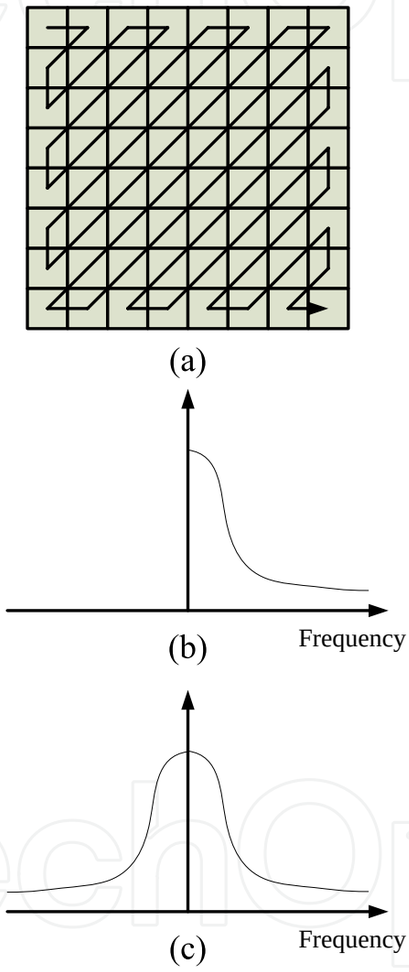


Fig. 4. (a) zigazg scanning for a 8×8 block (b) 1-D signal distribution after zigzag scanning order (c) The symmetric type of frequency for the 1-D signal distribution.

on the analysis result mentioned above, the R component is very decisive for GI images and it needs to be compressed completely. However, the G1, G2 and B components do not need to be compressed completely because they are of less than the R component. Therefore, in order to efficiently reduce the memory access to expend the battery life of capsule endoscopy, the datum of G1, G2 and B components should be appropriately decreased according to the proportion of their importance prior to the compression process. In this

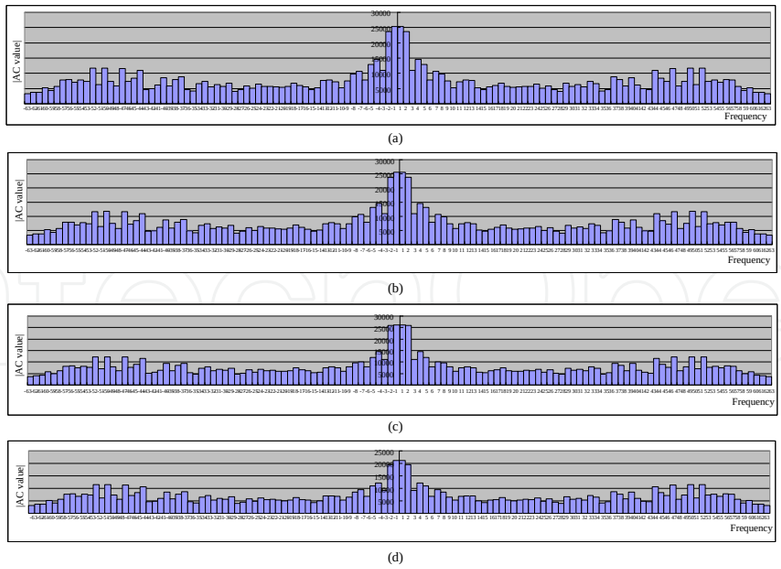


Fig. 5. (a) Spatial-frequency distribution converting into one-dimension for G1 component (b) Spatial-frequency distribution converting into one-dimension for G2 component (c) Spatial-frequency distribution converting into one-dimension for R component (d) Spatial-frequency distribution converting into one-dimension for B component.

paper, we successfully propose a subsample-based GICam image compression algorithm and the proposed algorithm firstly uses the subsample technique to reduce the incoming datum of G1, G2 and B components before the compression process. The next section will describe the proposed algorithm in detail.

2.3 The subsample-based GICam image compression algorithm

Fig.6 illustrates the GICam-II compression algorithm. For a 512×512 raw image, the raw image firstly divides into four parts, namely, R, G1, G2, and B components and each of the components has 256×256 pixels. For the R component, the incoming image size to the 2-D DCT is 256×256×8 bits, Where, the incoming image datum are completely compressed because of the importance itself in GI images. Except for the R component, the GICam-II

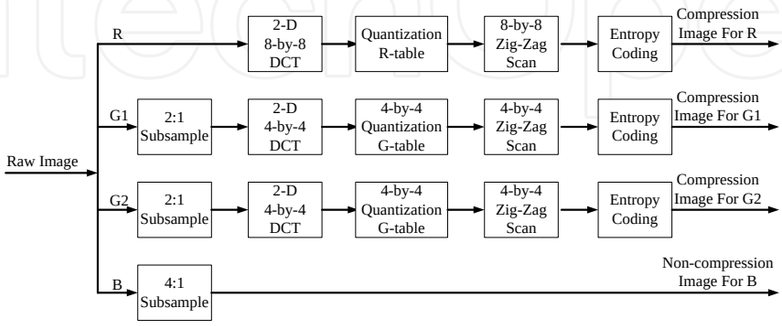


Fig. 6. The GICam-II image compression algorithm.

algorithm can use an appropriate subsample ratio to pick out the necessary image pixels into

the compression process for G1, G2 and B components, and Eq.28 and Eq.29 are formulas for the subsample technique. $SM_{16:2m}$ is the subsample mask for the subsample ratio 16-to-2m as shown in Eq.28, and the subsample mask $SM_{16:2m}$ is generated from basic mask as shown in Eq.29. The type of subample direction is block-based, when certain positions in the subsample mask are one, their pixels in the same position will be compressed, or otherwise they are not processed. For the G1 and G2 components, the low subsample ratio must be assigned, considering their secondary importance in GI images. Thus, the 2:1 subsample ratio is candidate one, and the subsample pattern is shown in Fig.7 (a). Finally, for the B component, the 4:1 subsample ratio is assigned and the subsample pattern is shown in Fig.7 (b). In the GICam-II image compression algorithm, the 8×8 2-D DCT is still used to transfer the R component. However, the 4×4 2-D DCT is used for G1 and G2 components because the incoming datum are reduced by subsample technique. Moreover, the G quantization table is also modified and shown in the Fig.8. Finally, the B component is directly transmitted; not be compressed, after extremely decreasing the incoming datum. Because of the non-compression for the B component, the 8×8 and 4×4 zig-zag scanning techniques are added into the GICam-II to further increase the compression rate for R, G1 and G2 components before entering the entropy encoding. In the GICam-II, the Lempel-Ziv (LZ) coding (18) is also employed for the entropy coding because of non-look-up tables and low complex computation.

$$SM_{16:2m}(i,j) = BM_{16:2m}(i \mod 4, j \mod 4) \quad m = 1,2,3,4,5,6,7,8. \tag{28}$$

$$BM_{16:2m}(k,l) = \begin{bmatrix} u(m-1)\ u(m-5)\ u(m-2)\ u(m-6) \\ u(m-7)\ u(m-3)\ u(m-8)\ u(m-4) \\ u(m-2)\ u(m-5)\ u(m-1)\ u(m-6) \\ u(m-7)\ u(m-3)\ u(m-8)\ u(m-4) \end{bmatrix} \tag{29}$$

where $u(n)$ is a step function, $u(n) = \begin{cases} 1, & for\ n \geq 0 \\ 0, & for\ n < 0. \end{cases}$

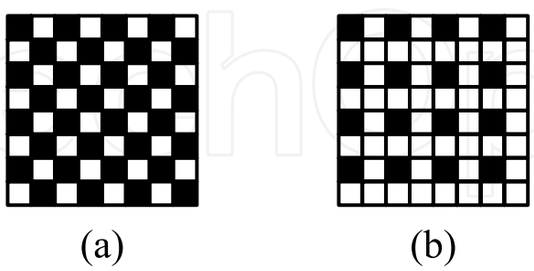


Fig. 7. (a) 2:1 subsample pattern (b) 4:1 subsample pattern.

2.4 The architecture of subsample-based GICam image compressor

Fig.9 shows the architecture of the GICam-II image compressor and it faithfully executes the proposed GICam-II image compression algorithm shown in Fig.6. The window size, w,

32	32	32	32	32	32	64	64
32	16	16	32	32	64	64	128
32	16	16	32	32	64	128	128
32	32	32	32	64	64	128	256
32	32	32	64	64	128	128	256
64	64	64	128	128	128	256	256
64	128	128	128	256	256	256	256
128	128	128	256	256	256	256	512

(a)

16	16	32	32
16	16	32	64
32	32	64	64
64	64	128	128

(b)

Fig. 8. (a)The modified R quantization table (b) The modified G quantization table.

and the maximum matching length,*l* parameters for LZ77 encoder can be loaded into the parameter register file via a serial interface after the initial setting of the hardware reset. Similarly, coefficients of 2-D DCT and parameters of initial setting for all controllers shown in Fig.9 can be also loaded into the parameter register file. The GICam-II image compressor processes the image in the block order of G1, R, G2 and B. Because the data stream from the image sensor is block-based, the GICam-II image compressor adopts the structure of ping-pong memory to hold each block of data. The advantage of using this structure is the high parallelism between the data loading and data precessing.

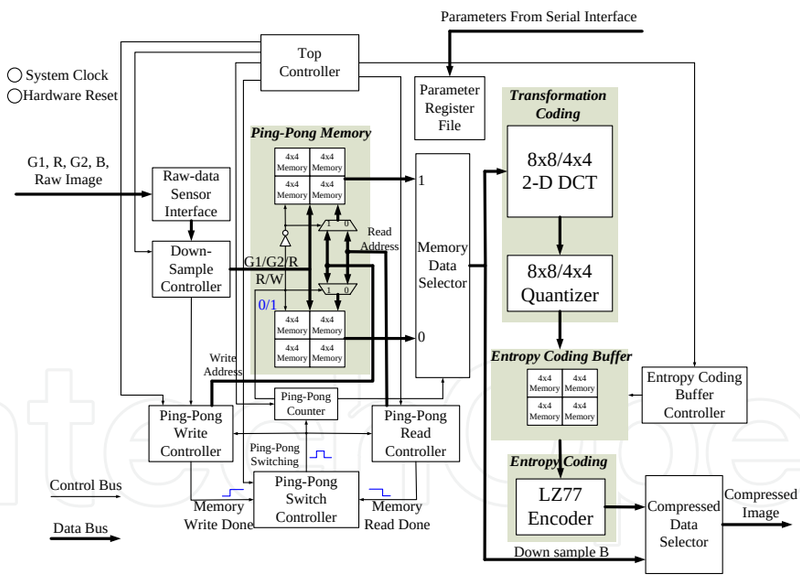


Fig. 9. The GICam-II image compressor.

When the GICam-II image compressor begins, the proposed architecture first loads the incoming image in the block order of G1, R, G2 and B from the image sensor and passes them with the valid signal control via the Raw-Data Sensor Interface. The Raw-Data Sensor Interface is a simple register structure with one clock cycle delay. This design absolutely makes sure that no any glue-logic circuits that can affects the timing of logic synthesis exists between the raw image sensor and the the GICam-II image compressor. The Down-Sample

Controller receives the valid data and then selects the candidate subsample ratio to sample the candidate image data in the block order of G1, R, G2 and B. The Ping-Pong Write Controller can accurately receive the data loading command from the Down-Sample Controller and then pushes the downsample image data into the candidate one of the ping-pong memory. At the same time, the Ping-Pong Read Controller pushes the stored image data from another memory into the Transformation Coding. The Ping-Pong Write Controller and the Ping-Pong Read Controller will issue an announcement to the Ping-Pong Switch Controller, respectively while each data-access is finished. When all announcement arrives in turn, the Ping-Pong Switch Controller will generate a pulse-type Ping-Pong Switching signal, one clock cycle, to release each announcement signal from the high-level to zero for the Ping-Pong Write Controller and the Ping-Pong Read Controller. The Ping-Pong Switch Counter also uses the Ping-Pong Switching signal to switch the read/write polarity for each memory in the structure of the Ping-Pong Memory.

The Transformation Coding consists of the 2-D DCT and the quantizer. The goal of the transformation coding is to transform processing data from the spatial domain into the spatial frequency domain and further to shorten the range in the spatial frequency domain before entropy coding in order to increasing the compression ratio. The 2-D DCT alternatively calculates row or column 1-D DCTs. The 1-D DCT is a multiplier-less implementation using the algebraic integer encoding (11). The algebraic integer encoding can minimize the number of addition operations. As regards the RG quantizer, the GICam-II image compressor utilizes the barrel shifter for power-of-two products. The power-of-two quantization table shown in Fig.8 can reduce the cost of multiplication while quality degradation is quite little. In addition, the 8-by-8 memory array between the quantizer and the LZ77 encoder is used to synchronize the operations of quantization and LZ77 encoding. Since the frame rate of GICam-II image compressor is 2 frames/second, the 2-D DCT can be folded to trade the hardware cost with the computing speed, and the other two data processing units, quantization and LZ77 encoder, can operate at low data rate. Due to non-compression for the B component, the B component is directly transmitted from the ping-pong memory, not be compressed. Finally, the LZ77 encoder is implemented by block-matching approach and the detail of each processing element and overall architecture have been also shown in paper (11).

2.5 Experimental results

We have particularly introduced the method of efficiently decreasing the incoming datum with the subsmaple technique in the GICam-II compression algorithm. The performance of the compression rate, the quality degradation and the ability of power saving will then be experimentally analyzed using the GICamm-II compressor.

2.5.1 The analysis of compression rate for GI images

In this paper, twelve GI images are tested and shown in the Fig.3. First of all, the target compression performance of the GICam-II image compression is to reduce image size by at least 75% . To meet the specification, we have to exploit the cost-optimal LZ coding parameters. There are two parameters in the LZ coding to be determined: the window size, w, and the maximum matching length,l. The larger the parameters, the higher the compression ratio will be; however,the implementation cost will be higher. In addition, there are two kinds

of LZ codings in the GICam-II compressor, one is R(w, l) for R component and the other is G(w, l) for G1 and G2 components. We set the values of parameters by using a compression ratio of 4:1 as the threshold. Our goal is to determine the minimum R(w, l) and G(w, l) sets under the constraint of 4:1 compression ratio.

The compression ratio (CR) is defined as the ratio of the raw image size to the compressed image size and formulated as Eq.30. The measure of the compression ratio is the compression rate. The formula of the compression rate is calculated by Eq.31. The results in Fig.10 are shown by simulating the behavior model of GICam-II compressor; it is generated by MATLAB. As seen in Fig.10, simulating with twelve endoscopic pictures, (32, 32) and (16, 8) are the minimum R(w, l) and G(w, l) sets to meet the compression ratio requirement. The subsample technique of the GICam-II compressor initially reduces the input image size by 43.75% ((1-1/4-(1/4*1/2*2)-(1/4*1/4))*100%) before executing the entropy coding, LZ77 coding. Therefore, the overall compression ratio of GICam-II compressor minus 43.75% is the compression effect of LZ77 coding that combines with the quantization, and the simulation results are shown in Fig.11.

This research paper focuses to propose a subsample-based low-power image compressor for capsule gastrointestinal endoscopy. This obvious reddish characteristic is due to the slightly luminous intensity of LEDs and the formation of image in the capsule gastrointestinal endoscopy. The GICam-II compression algorithm is motivated on the basis of this reddish pattern. Therefore, we do not consider compressing other endoscopic images except for gastrointestinal images to avoid the confusion of topic for this research. However, general endoscopic images generated via a wired endoscopic takes on the yellow characteristic due to the vividly luminous intensity of LEDs. The yellow pattern mainly consists of red and green and it also complies with the color sensitivity result in this research work. Therefore, I believe that the proposed GICam-II still supports good compression ratio for general endoscopic images.

$$Compression\ Ratio\ (CR) = \frac{bits\ before\ compression}{bits\ after\ compression} \tag{30}$$

$$Compression\ Rate = (1 - CR^{-1}) \times 100\% \tag{31}$$

2.5.2 The analysis of compression quality for GI images

Using (32, 32) and (16, 8) as the parameter sets, in Table 4, we can see the performance in terms of the quality degradation and compression ratio. The measure of compression quality is the peak signal-to-noise ratio of luminance (PSNRY). The calculation of PSNRY is formulated as Eq.32. Where MSE is the mean square error of decompressed image and is formulated as Eq.33. In Eq.33, α_{ij} is the luminance value of original GI image and β_{ij} is the luminance value of decompressed GI image. The result shows that the degradation of decompressed images is quite low while the average PSNRY is 40.73 dB. Fig.12 illustrates the compression quality of decoded test pictures. The difference between the original image and the decompressed

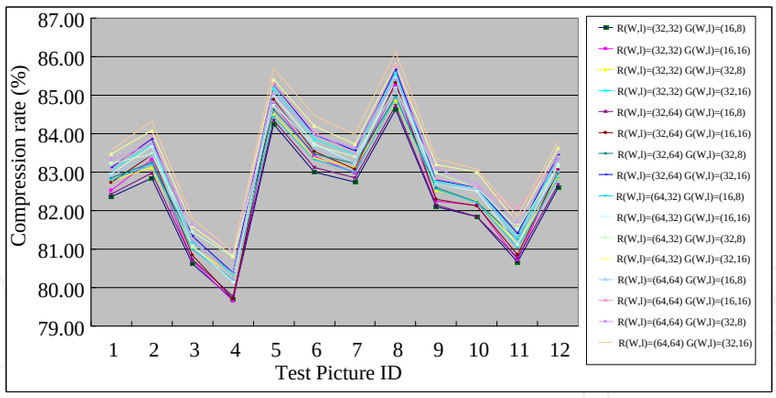


Fig. 10. The compression performance of the GICam-II image compressor.

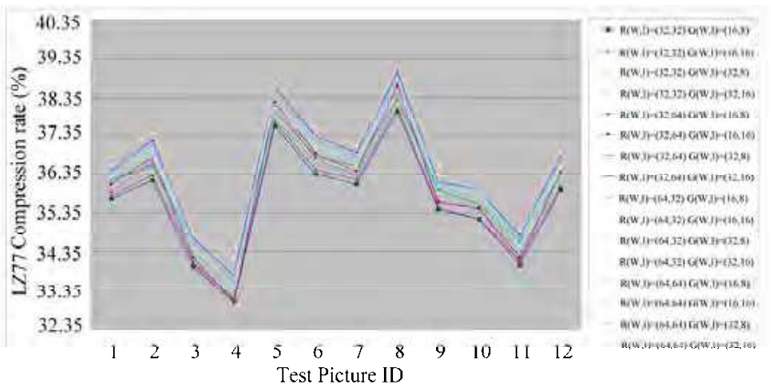


Fig. 11. The compression performance of LZ77 coding that combines with the quantization in the GICam-II image compressor .

image is invisible.

$$PSNRY = 10log_{10}(\frac{255^2}{MSE}) \tag{32}$$

$$MSE = (\frac{1}{M \times N}) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (\alpha_{ij} - \beta_{ij})^2 \tag{33}$$

To demonstrate the validity of decompressed images, five professional gastroenterology doctors from the Division of Gastroenterology, Taipei Medical University Hospital are invited to verify whether or not the decoded image quality is suitable for practical diagnosis. The criterion of evaluation is shown in Table 5. The score between 0 and 2 means that the diagnosis is affected, the score between 3 and 5 means that the diagnosis is slightly affected and the score between 6 and 9 means that the diagnosis is not affected. According to the evaluation results of Fig.13, all decoded GI images are suitable for practical diagnosis because of high evaluation score and the diagnoses are absolutely not affected, except for the 5th and 8th decoded images. The degrees of diagnoses are between no affection and extremely slight affection for the 5th and the 8th decoded images because only two doctors subjectively feel their diagnoses are slightly affected. However, these two decoded images are not mistaken in

Test Picture ID	PSNRY (dB)	Compression rate (%)
1	40.76	82.36
2	41.38	82.84
3	39.39	80.62
4	38.16	79.70
5	42.56	84.25
6	41.60	83.00
7	41.03	82.74
8	43.05	84.63
9	40.21	82.11
10	40.36	81.84
11	39.39	80.66
12	40.85	82.60
Average	40.73	82.28

Table 4. The simulation results of twelve tested pictures.

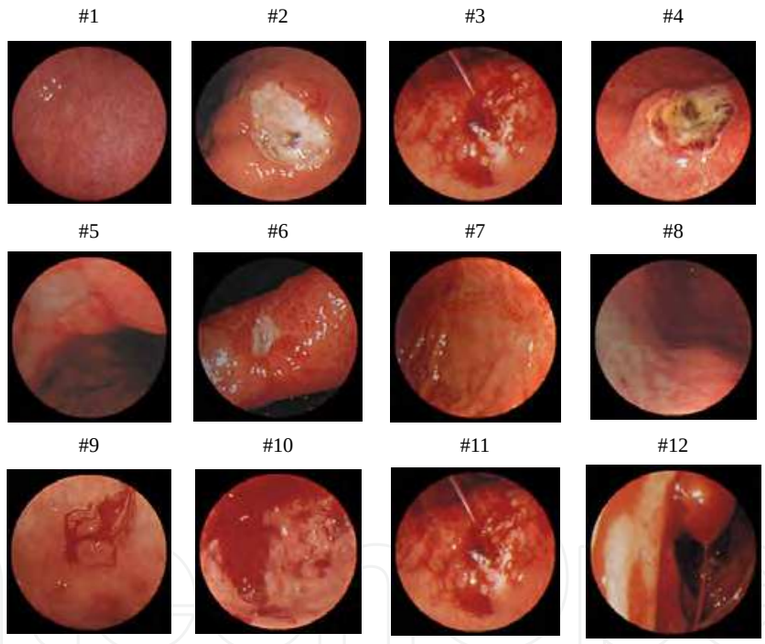


Fig. 12. Demosaicked GI images.

diagnosis for these professional gastroenterology doctors. Therefore, the PSNRY being higher than 38 dB is acceptable according to the objective criterion of gastroenterology doctors.

Score	Description
$0 \sim 2$	diagnosis is affected
$3 \sim 5$	diagnosis is slightly affected
$6 \sim 9$	diagnosis is not affected

Table 5. The criterion of evaluation.

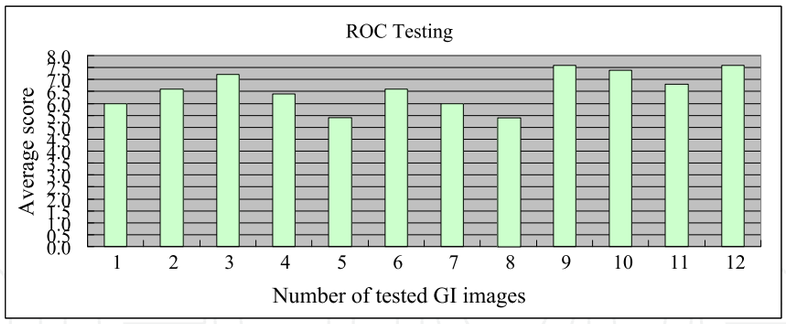


Fig. 13. The evaluation results of professional gastroenterology doctors.

2.5.3 The analysis of power saving

To validate the GICam-II image processor, we used the FPGA board of Altera APEX 2100 K to verify the function of the GICam-II image processor and the prototype is shown in Fig.14. After FPGA verification, we used the TSMC 0.18 μm 1P6M process to implement the GICam-II image compressor. When operating at 1.8 V, the power consumption of logic part is 3.88 mW, estimated by using PrimePower^{TM}. The memory blocks are generated by Artisan memory compiler and consume 5.29 mW. The total power consumption is 9.17 mW for the proposed design. When comparing the proposed GICam-II image compressor with our previous GICam one in Table 6, the power dissipation can further save 38.5% under the approximate condition of quality degradation and compression ratio because of the reduction of memory requirement for G1, G2 and B components.

The GICam-II compressor has poorer image reconstruction than JPEG and our previous GICam one because the GICam-II compressor uses the subsample scheme to down sample green and blue components according to the 2:1 and the 4:1 subsample ratios. The raw data before compression has lost some raw data information. Hence, the decoded raw data should be reconstructed (the first interpolation) before reconstructing the color images (the second interpolation). Using two level interpolations to reconstruct the color images has poorer image quality than one level interpolation. Fortunately, the decoded image quality using GICam-II compressor can be accepted and suitable for practical diagnosis and the evaluation results of professional gastroenterology doctors can be shown in the last subsection.

Finally, we compare the GICam-II image processor with other works and the comparison results are shown in the Table 7. According to the comparison results, our proposed GICam-II image compressor has lower area and lower operation frequency. It can fit into the existing designs.

3. Rank-Order Filtering (ROF) with a maskable memory

Rank-order filtering (ROF), or order-statistical filtering, has been widely applied for various speech and image processing applications [1]-[6]. Given a sequence of input samples $\{x_{i-k}, x_{i-k+1}, ..., x_i, ..., x_{i+l}\}$, the basic operation of rank order filtering is to choose the r-th largest sample as the output y_i, where r is the rank-order of the filter. This type of ROF is normally classified as *the non-recursive ROF* . Another type of ROF is called *the recursive ROF*. The difference between the recursive ROF and the non-recursive ROF is that

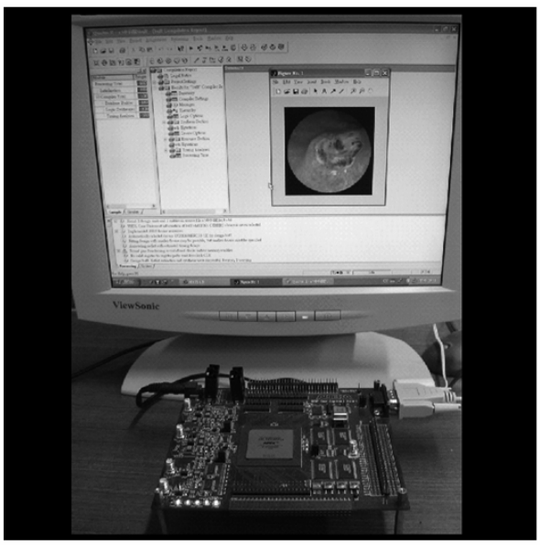


Fig. 14. The FPGA prototype of the GICam-II image compressor.

	JPEG designed by (19)	GICam image compressor (11)	Proposed GICam-II image compressor
Average PSNRY	46.37 dB	41.99 dB	40.73 dB
Average compression rate	82.20%	79.65%	82.28%
Average power dissipation	876mW	14.92 mW	9.17 mW

Table 6. The comparison Result with Previous GICam Works

	Area	Frequency (MHz)	Power (mW)	Supply Voltage (V)
GICam image compressor (11)	390k	12.58	14.92 (evaluated)	1.8
X.Xie et al.(12)*	12600k	40.0	6.2 (measured)	1.8
K.Wahid et al. (13)	325k	150	10 (evaluated)	1.8
X.Chen et al.(14)*	11200k	20	1.3 (measured)	0.95
Proposed GICam-II image compressor	318k	7.96	9.17 (evaluated)	1.8
* includes analog and transmission circuit and SRAM				

Table 7. The Comparison Results with Existing Works

the input sequence of the recursive ROF is $\{y_{i-k}, y_{i-k+1}, ..., y_{i-1}, x_i, ..., x_{i+l}\}$. Unlike linear filtering, ROF can remove sharp discontinuities of small duration without blurring the original signal; therefore, ROF becomes a key component for signal smoothing and impulsive noise elimination. To provide this key component for various signal processing applications, we intend to design a configurable rank-order filter that features low cost and high speed.

Many approaches for hardware implementation of rank-order filtering have been presented in the past decades [8, 10-24]. Many of them are based on sorting algorithm [11, 22, 23, 25-28]. They considered the operation of rank-order filtering as two steps: sorting and choosing. Papers [10, 19] have proposed systolic architectures for rank-order filtering based on sorting algorithms, such as bubble sort and bitonic sort. These architectures are fully pipelined for high throughput rate at the expense of latency, but require a large number of compare-swap units and registers. To reduce the hardware complexity, papers [8, 12, 14, 15, 29-32] present linear-array structures to maintain samples in sorted order. For a sliding window of size N, the linear-array architectures consist of N processing elements and require three steps for each iteration: finding the proper location for new coming sample, discarding the eldest one, and moving samples between the newest and eldest one position. The three-step procedure is called delete-and-insert (DI). Although the hardware complexity is reduced to $O(N)$, they require a large latency for DI steps. Paper [31] further presents a micro-programmable processor for the implementations of the median-type filters. Paper [20] presents a parallel architecture using two-phase design to improve the operating speed. In this paper, they first modified the traditional content-addressable memory (CAM) to a shiftable CAM (SCAM) processor with shiftable memory cells and comparators. Their architecture can take advantages of CAM for parallelizing the DI procedure. Then, they use two-phase design to combine delete and insert operations. Thereafter, the SCAM processor can quickly finish DI operations in parallel. Although the SCAM processor has significantly increased the speed of the linear-array architecture, it can only process a new sample at a time and cannot efficiently process 2-D data. For a window of size *n-by-n*, the SCAM processor needs n DI procedures for each filtering computation. To have an efficient 2-D rank-order filter, papers [12, 27] present solutions for 2-D rank-order filtering at the expense of area.

In addition to the sorting algorithm, the paper [35] applies the threshold decomposition technique for rank-order filtering. To simplify the VLSI complexity, the proposed approach uses three steps : decomposition, binary filtering, and recombination. The proposed approach significantly reduce the area complexity from exponential to linear. Papers [10, 13, 16-18, 21, 33, 34] employ the bit-sliced majority algorithm for median filtering, the most popular type of rank-order filtering. The bit-sliced algorithm [36, 37] bitwisely selects the ranked candidates and generates the ranked result one bit at a time. Basically, the bit-sliced algorithm for median filtering recursively executes two steps: majority calculation and polarization. The majority calculation, in general, dominates the execution time of median filtering. Papers [10], [17] and [37] present logic networks for implementation of majority calculation. However, the circuits are time-consuming and complex so that they cannot take full advantages of bit-sliced algorithm. Some papers claim that this type of algorithm is impractical for logic circuit implementation because of its exponential complexity [31]. Paper [21] uses an inverter as a voter for majority calculation. It significantly improves both cost and processing speed, but the noise margin will become narrow as the number of inputs increases. The narrow noise

margin makes the implementation impractical and limits the configurability of rank-order filtering.

Instead of using logic circuits, this paper presents a novel memory architecture for rank-order filtering based on a generic rank-order filtering algorithm. The maskable memory structure, called dual-cell random-access memory (DCRAM), is an extended SRAM structure with maskable registers and dual cells. The maskable registers allow the architecture to selectively read or write bit-slices, and hence speed up "parallel read" and "parallel polarization" tasks. The control of maskable registers is driven by a long-instruction-word (LIW) instruction set. The LIW makes the proposed architecture programmable for various rank-order filtering algorithms, such as recursive and non-recursive ROFs. The proposed architecture has been implemented using TSMC 0.18um 1P6M technology and successfully applied for 1-D/2-D ROF applications. For 9-point 1-D and 3-by-3 2-D ROF applications, the core size is $356.1 \times 427.7um^2$. As shown in the post-layout simulation, the DCRAM-based processor can operate at 290 MHz for 3.3V supply and 256 MHz for 1.8V supply. For image processing, the performance of the proposed processor can process video clips of SVGA format in real-time.

3.1 The generic bit-sliced rank-order filtering algorithm

Let W_i={$x_{i-k}, x_{i-k+1}, ..., x_i, ..., x_{i+l}$} be a window of input samples. The binary code of each input x_j is denoted as $u_j^{B-1} \cdots u_j^1 u_j^0$. The output y_i of the r-th order filter is the r-th largest sample in the input window W_i, denoted as $v_i^{B-1} \cdots v_i^1 v_i^0$. The algorithm sequentially determines the r-th order value bit-by-bit starting from the most significant bit (MSB) to the least significant bit (LSB). To start with, we first count 1's from the MSB bit-slice of input samples and use Z_{B-1} to denote the result. The b-th bit-slice of input samples is defined as $u_{i-k}^b u_{i-k+1}^b \ldots u_i^b \ldots u_{i+l}^b$. If Z_{B-1} is greater than or equal to r, then v_i^{B-1} is 1; otherwise, v_i^{B-1} is 0. Any input sample whose MSB has the same value as v_i^{B-1} is considered as one of candidates of the r-th order sample. On the other hand, if the MSB of an input sample is not equal to v_i^{B-1}, the input sample will be considered as a non-candidate. Non-candidates will be then polarized to either the largest or smallest value. If the MSB of an input sample x_j is 1 and v_i^{B-1} is 0, the rest bits (or lower bits) of x_j are set to 1's. Contrarily, if the MSB of an input sample x_j is 0 and v_i^{B-1} is 1, the rest bits (or lower bits) of x_j are set to 0's. After the polarization, the algorithm counts 1's from the consecutive bit-slice and then repeats the polarization procedure. Consequently, the r-th order value can be obtained by recursively iterating the steps bit-by-bit. The following pseudo code illustrates the generic bit-sliced rank-order filtering algorithm:

Given the input samples, the window size $N=l+k+1$, the bitwidth B and the rank r, do:

Step 1: Set b=B-1.

Step 2: (Bit counting)

Calculate Z_b from $\{u_{i-k}^b, u_{i-k+1}^b, \ldots, u_i^b, \ldots, u_{i+l}^b\}$.

Step 3: (Threshold decomposition)

If $Z_b \geq r$,v_i^b =1; otherwise v_i^b =0.

Step 4: (Polarization)

If $u_j^b \neq v_i^b$, $u_j^m = u_j^b$ for $0 \leq m \leq b-1$ and $i-k \leq j \leq i+l$

Step 5: *b=b-1*.

Step 6: If $b \geq 0$ go to Step 2.

Step 7: Output y_i.

Fig.15 illustrates a bit-sliced ROF example for N=7, B=4, and r=1. Given that the input samples are $7(0111_2)$, $5(0101_2)$, $11(1011_2)$, $14(1110_2)$, $2(0010_2)$, $8(1000_2)$, and $3(0011_2)$, the generic algorithm will produce $14(1110_2)$ as the output result. At the beginning, the "Bit counting" step will calculate the number of 1's at MSBs, which is 3. Since the number of 1's is greater than r, the "Threshold decomposition" step sets the MSB of y_i to '1'. Then, the "Polarization" step will consider the inputs with $u_j^3 = 1$ as candidates of the ROF output and polarize the lower bits of the others to all 0's. After repeating the above steps with decreasing b, the output y_i will be $14(1110_2)$.

3.2 The dual-cell RAM architecture for rank-order filtering

As mentioned above, the generic rank-order filtering algorithm generates the rank-order value bit-by-bit without using complex sorting computations. The main advantage of this algorithm is that the calculation of rank-order filtering has low computational complexity and can be mapped to a highly parallel architecture. In the algorithm, there are three main tasks: bit counting, threshold decomposition, and polarization. To have these tasks efficiently implemented, this paper presents an ROF processor based on a novel maskable memory architecture, as shown in Fig.16. The memory structure is highly scalable with the window size increasing, by simply adding memory cells. Furthermore, with the instruction decoder and maskable memory, the proposed architecture is programmable and flexible for different kinds of ROFs.

The dual-cell random-access memory (DCRAM) plays a key role in the proposed ROF architecture. In the DCRAM, there are two fields for reusing the input data and pipelining the filtering process. For the one-dimensional (1-D) ROF, the proposed architecture receives one sample at a time. For the *n-by-n* two-dimensional (2-D) ROF, the architecture reads n samples into the input window within a filtering iteration. To speed up the process of rank-order filtering and pipeline the data loading and filtering calculation, the data field loads the input data while the computing field is performing bit-sliced operations. Hence, the execution of the architecture has two pipeline stages: data fetching and rank-order calculation. In each iteration, the data fetching first loads the input sample(s) into the data field and then makes copies from the data field to the computing field. After having the input window in the computing field, the rank-order calculation bitwisely accesses the computing field and executes the ROF tasks.

The computing field is in the maskable part of DCRAM. The maskable part of DCRAM performs parallel reads for bit counting and parallel writes for polarization. The read-mask register (RMR) is configured to mask unwanted bits of the computing field during read operation. The value of RMR is one-hot-encoded so that the bit-sliced values can be read from

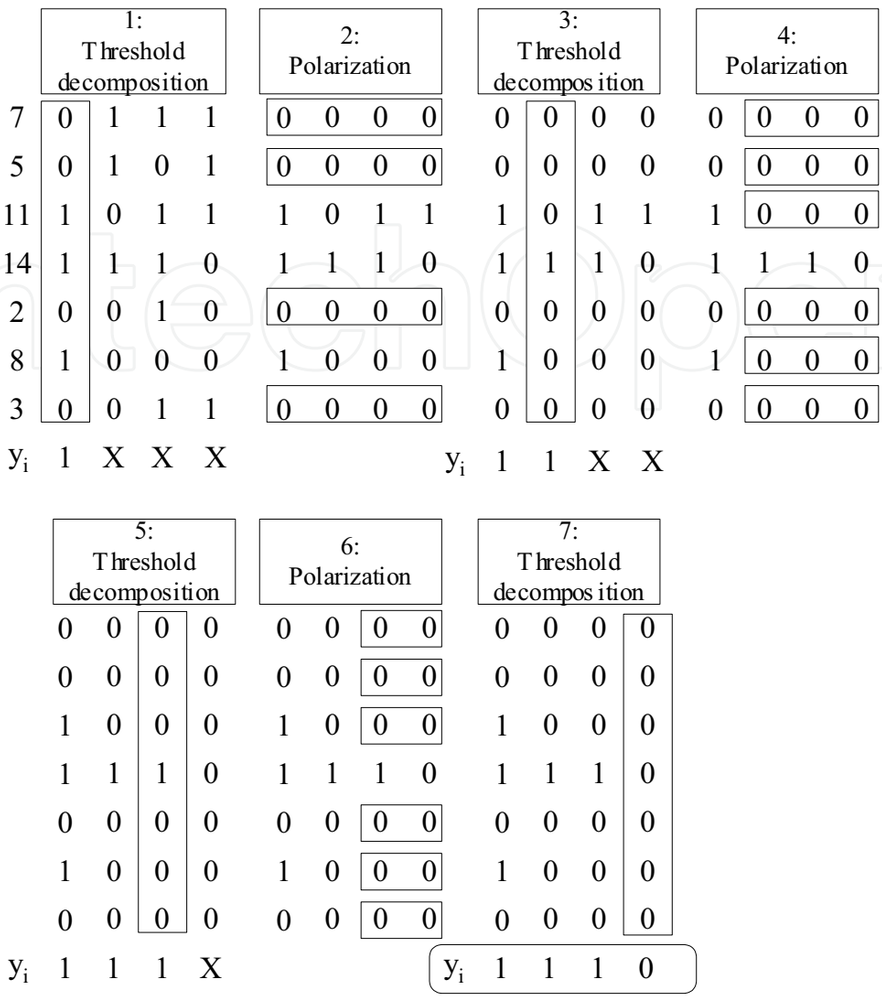


Fig. 15. An example of the generic bit-sliced ROF algorithm for N=7, B=4, and r=1.

the memory in parallel. The bit-sliced values will then go to the Level-Quantizer for threshold decomposition. When the ROF performs polarization, the write-mask register (WMR) is configured to mask untouched bits and allow the polarization selector (PS) to polar lower bits of noncandidate samples. Since the structure of memory circuits is regular and the maskable scheme provides fast logic operations, the maskable memory structure features low cost and high speed. It obviously outperforms logic networks on implementation of bit counting and polarization.

To start with the algorithm, the RMR is one-hot-masked according to the value b in the generic algorithm and then the DCRAM outputs a bit-sliced value $\{u^b_{i-k}, u^b_{i-k+1}, \ldots, u^b_i, \ldots, \ldots u^b_{i+l}\}$ on "**c_d**". The bit-sliced value will go to both the Level-Quantizer and PS. The Level-Quantizer performs the Step 2 and Step 3 by summing up bits of the bit-sliced value to Z_b and comparing Z_b with the rank value r. The rank value r is stored in the rank register (RR). The bitwidth w of Z_b is $\lceil log_2^N \rceil$. Fig.17 illustrates the block diagram of the Level-Quantizer, where FA denotes the full adder and HA denotes the half adder. The signals "S" and "C" of each FA or HA represent sum and carry, respectively. The circuit in the dash-lined box is a comparator. The

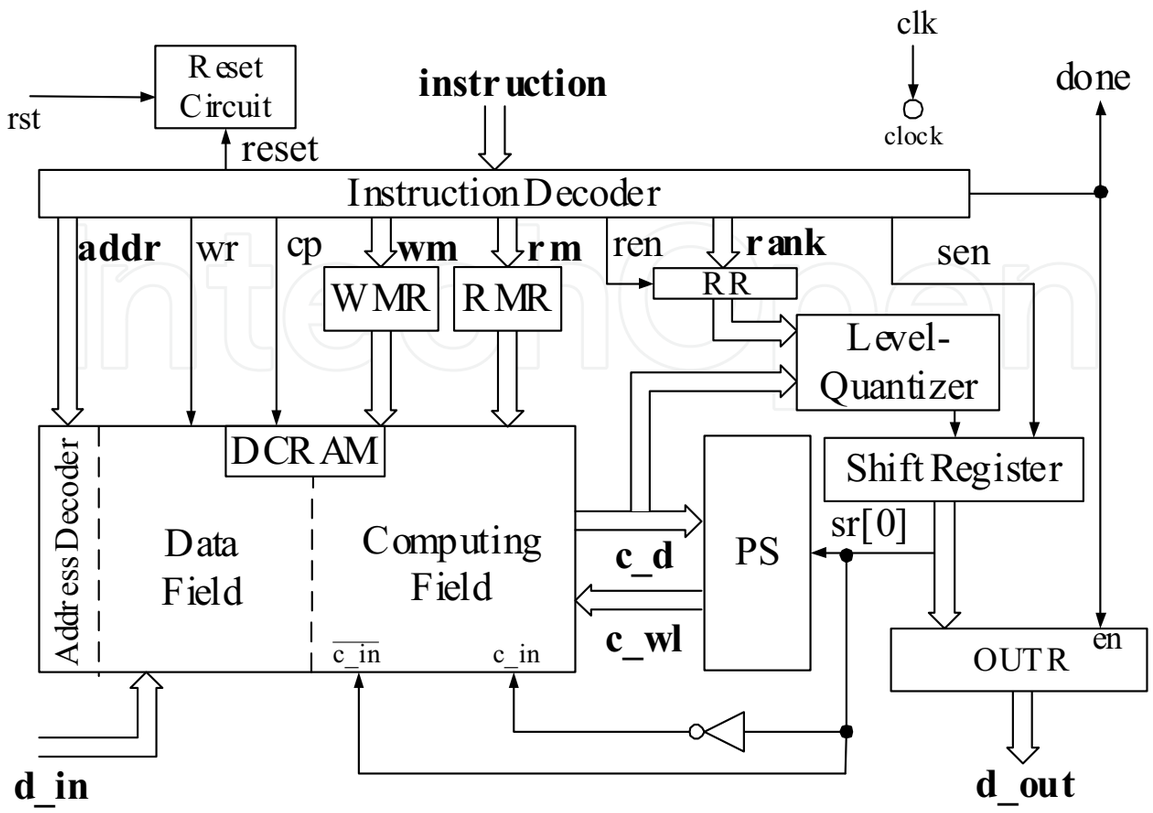


Fig. 16. The proposed rank-order filtering architecture.

comparator is implemented by a carry generator because the comparison result of Z_b and r can be obtained from the carry output of Z_b plus the two's complement of r. The carry output is the quantized value of the Level-Quantizer.

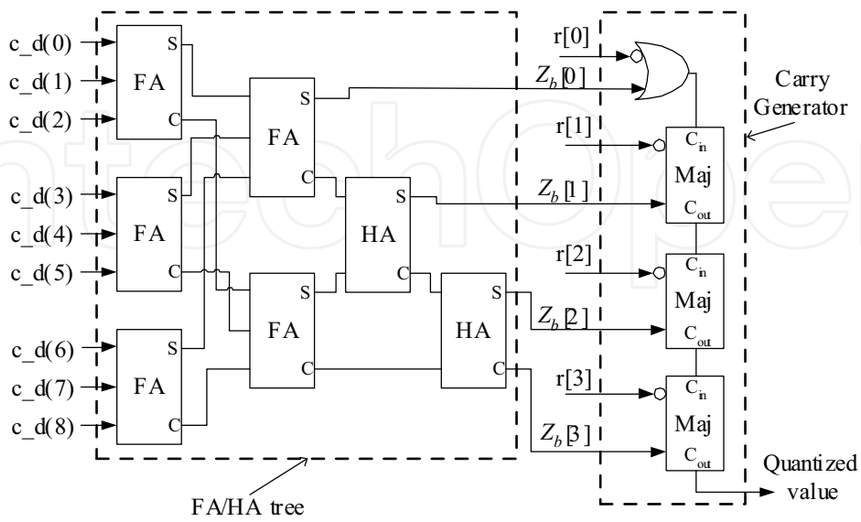


Fig. 17. The block diagram of the Level-Quantizer.

Normally, the comparison can be made by subtracting r from Z_b. Since Z_b and r are unsigned numbers, to perform the subtraction, both numbers have to reformat to two's complement numbers by adding a sign bit. In this paper, the reformated numbers of Z_b and r are expressed as $Z_{b,S}$ and r_S, respectively. Since both numbers are positive, their sign bits are equal to '0'. If $Z_{b,S}$ is less than r_S, the result of subtraction, Δ, will be negative; that is, the sign bit (or MSB) of Δ is '1'. Eq.34 shows the inequation of the comparison, where $\overline{r_S}$ denotes the one's complement of r_S and $\mathbf{1}$ denotes $(00\ldots01)_2$. Because the MSB of $Z_{b,S}$ is '0' and the MSB of $\overline{r_S}$ is '1', to satisfy Eq.34, the carry of $Z_{b,S}^{w-1} + \overline{r_S^{w-1}}$ must be equal to '0' so that the sign bit of Δ becomes '1'. To simplify the comparison circuit, instead of implementing an adder, we use the carry generator to produce the carry of $Z_{b,S}^{w-1} + \overline{r_S^{w-1}}$. Each cell of the carry generator is a majority (Maj) circuit that performs the boolean function shown in Eq.35. Furthermore, we use an OR gate at the LSB stage because of Eq.36. Thus, the dash-lined box is an optimized solution for comparison of Z_b and r without implementing the *bit-summation* and *signed-extension* parts.

$$\Delta = Z_{b,S} + \overline{r_S} + \mathbf{1} < 0. \tag{34}$$

$$Maj(A, B, C) = AB + BC + AC. \tag{35}$$

$$Z_b^0 \cdot \overline{r^0} + Z_b^0 \cdot 1 + 1 \cdot \overline{r^0} = Z_b^0 + \overline{r^0}. \tag{36}$$

After the Level-Quantizer finishes the threshold decomposition, the quantized value goes to the LSB of the shift register, "sr[0]". Then, the polarization selector (PS) uses exclusive ORs (XORs) to determine which words should be polarized, as shown in Fig.18. Obviously, the XORs can examine the condition of $u_j^b \neq v_i^b$ and select the word-under-polarization's (WUPs) accordingly. When **"c_wl"** is '1', the lower bits of selected words will be polarized; the lower bits are selected by WMR. According to the Step 4, the polarized value is u_j^b which is the inversion of v_i^b. Since v_i^b is the value of sr[0], we inverse the value of "sr[0]" to "c_in", as shown in Fig.16.

As seen in the generic algorithm, the basic ROF repeatedly executes *Bit-counting*, *Threshold decomposition*, and *Polarization* until the LSB of the ROF result being generated. Upon executing B times of three main tasks, the ROF will have the result in the Shift Register. A cycle after, the result will then go to the output register (OUTR). Doing so, the proposed architecture is able to pipeline the iterations for high-performance applications.

3.3 Implementation of dual-cell random-access memory

Fig.19 illustrates a basic element of DCRAM. Each element has two cells for data field and computing field, respectively. The data cell is basically an SRAM cell with a pair of bitlines. The SRAM cell is composed of INV1 and INV2 and stores a bit of input sample addressed by the wordline "d_wl[i]". The computing cell performs three tasks: *copy*, *write*, and *read*. When the copy-line "cp" is high, through INV5 and INV6, the pair of INV3 and INV4 will have the copy of the 1-bit datum in the data cell. The *copy* operation is unidirectional, and the pair of INV5 and INV6 can guarantee this directivity. When the one-bit value stored in the computing cell needs to be polarized, the "wm[j]" and "c_wl[i]" will be asserted, and the computing cell will perform the *write* operation according to the pair of bitlines "c_bl[j]" and "$\overline{\text{c\_bl[j]}}$". When the ROF reads the bit-sliced value, the computing cell uses an NMOS,

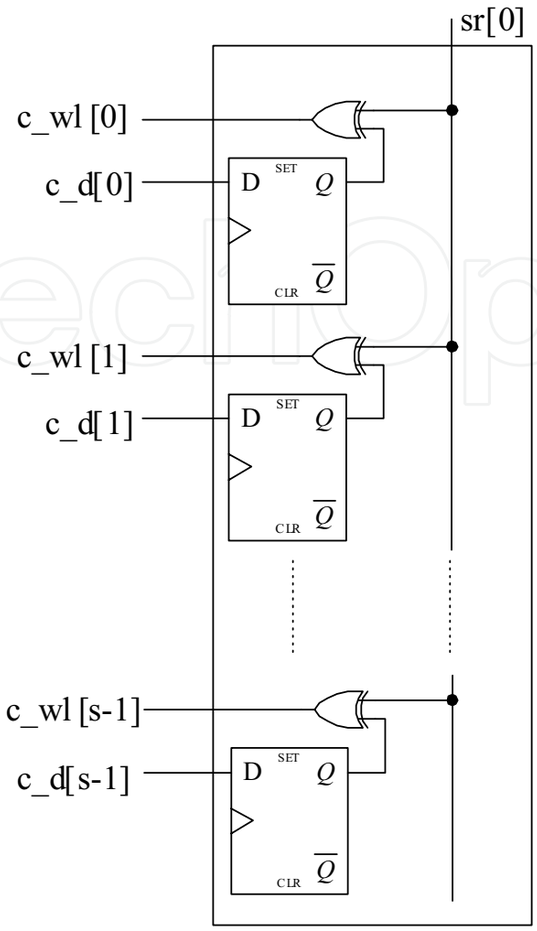


Fig. 18. The polarization selector (PS).

gated by "rm[j]", to output the complement value of the stored bit to the dataline "$\overline{\text{c\_d[i]}}$". The datalines of computing cells of each word will be then merged as a single net. Since the RMR is one-hot configured, each word has only a single bit being activated during the *read* operation.

As shown in Fig.20, the dataline "$\overline{\text{c\_d[i]}}$" finally goes to an inverter to pull up the weak '1', which is generated by the "rm[j]"-gated NMOS, and hence the signal "c_d[i]" has the value of the *i*-th bit of each bit-slice. Because the ROF algorithm polarizes the non-candidate words with either all zeros or all ones, the bitline pairs of computing cells are merged as a single pair of "c_in" and "$\overline{\text{c\_in}}$".

Fig.21 illustrates the implementation of DCRAM with the floorplan. Each $D_i - C_i$ pair is a maskable memory cell where D_i denotes D_cell(*i*) and C_i denotes C_cell(*i*). Each word is split into higher and lower parts for reducing the memory access time and power dissipation (64). The control block is an interface between control signals and address decoder. It controls wordlines and bitlines of DCRAM. When the write signal "wr" is not asserted, the control block will disassert all wordlines by the address decoder.

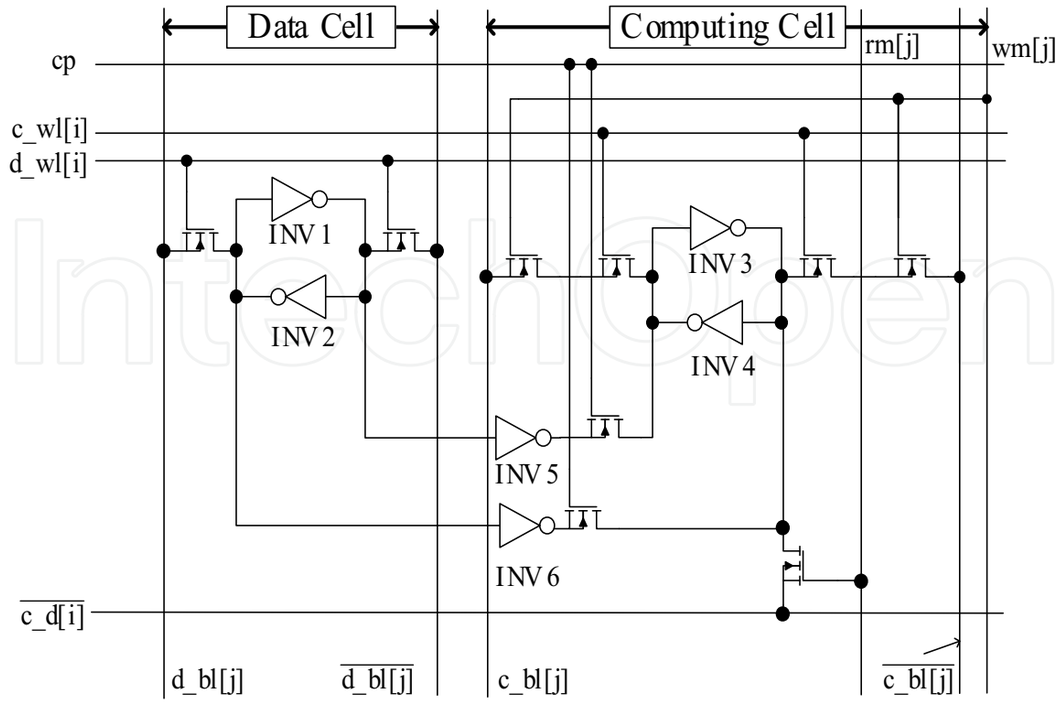


Fig. 19. A basic element of DCRAM.

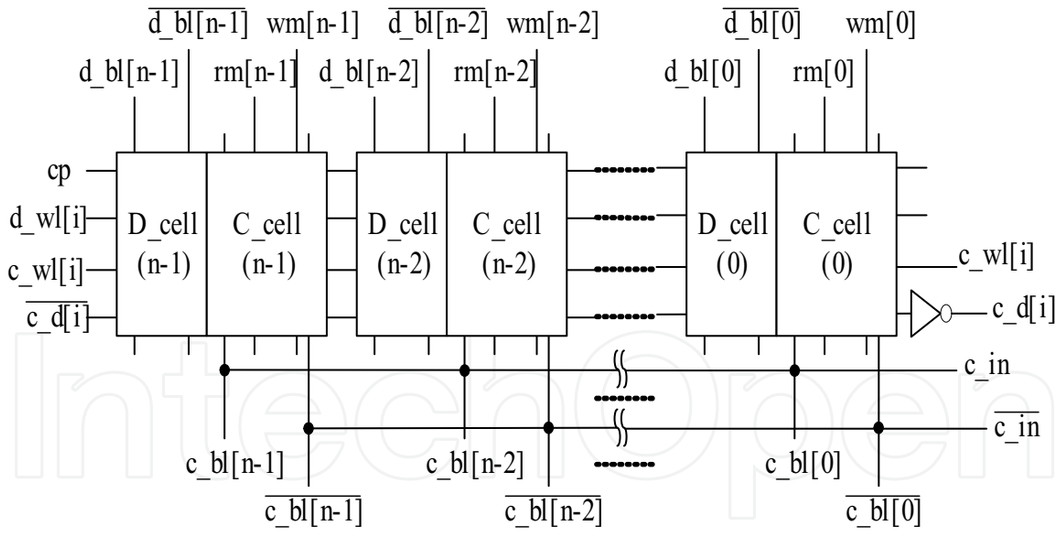


Fig. 20. A DCRAM word mixing data field and computing field. D_cell(*i*) denotes the data field of *i*-th bit and C_cell(*i*) denotes the computing field of *i*-th bit.

3.4 Instruction set of proposed ROF processor

The proposed ROF processor is a core for the impulsive noise removal and enabled by an instruction sequencer. Fig.22 illustrates the conceptual diagram of the ROF processor. The instruction sequencer is used for the generation of instruction codes and the control

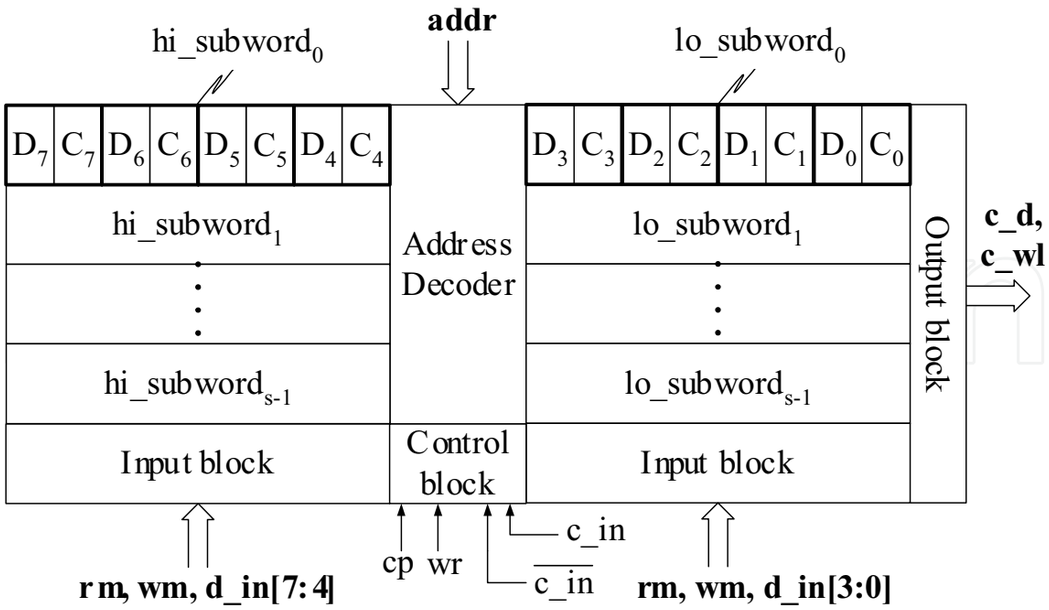


Fig. 21. The floorplan of DCRAM.

of input/output streams. The instruction sequencer can be a microprocessor or dedicated hardware.

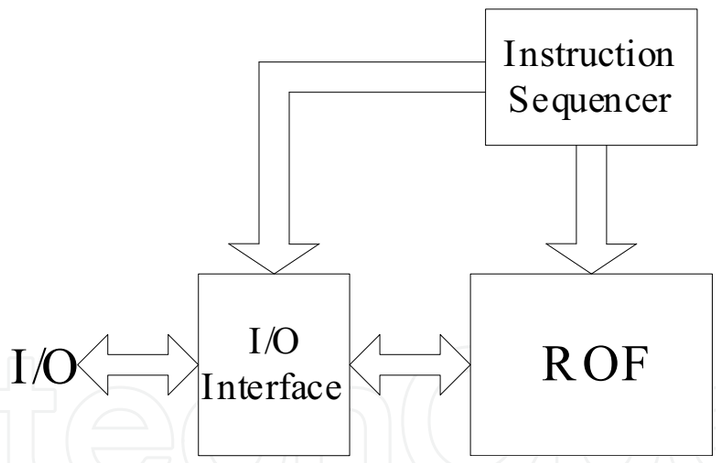


Fig. 22. The conceptual diagram of the ROF processor.

Fig.23 lists the format of the instruction set. An instruction word contains two subwords: the data field instruction and the computing field instruction. Each instruction cycle can concurrently issue two field instructions for parallelizing the data preparation and ROF execution; hence, the proposed processor can pipeline ROF iterations. When one of the field instructions performs "no operation", `DF_NULL` or `CF_NULL` will be issued. All registers in the architecture are updated a cycle after instruction issued.

The instruction `SET` resets all registers and set the rank register RR for a given rank-order *r*. The instruction `LOAD` loads data from "**d_in**" by asserting "wr" and setting "addr". The instruction `COPY/DONE` can perform the "COPY" operation or "DONE" operation. When the

bit value of *c* is '1', the DCRAM will copy a window of input samples from the data field to the computing field. When the bit value of *d* is '1', the DCRAM wraps up an iteration by asserting "en" and puts the result into OUTR.

The instruction `P_READ` is issued when the ROF algorithm executes bit-sliced operations. The field <*mask*> of `P_READ` is one-hot coded. It allows the DCRAM to send a bit-slice to the Level-Quantizer and PS for the *Threshold decomposition* task. The instruction `P_WRITE` is issued when the ROF algorithm performs the *Polarization* task. The field <*mask*> of `P_WRITE` is used to set a consecutive sequence of 1's. The sequence can mask out the higher bits for polarization.

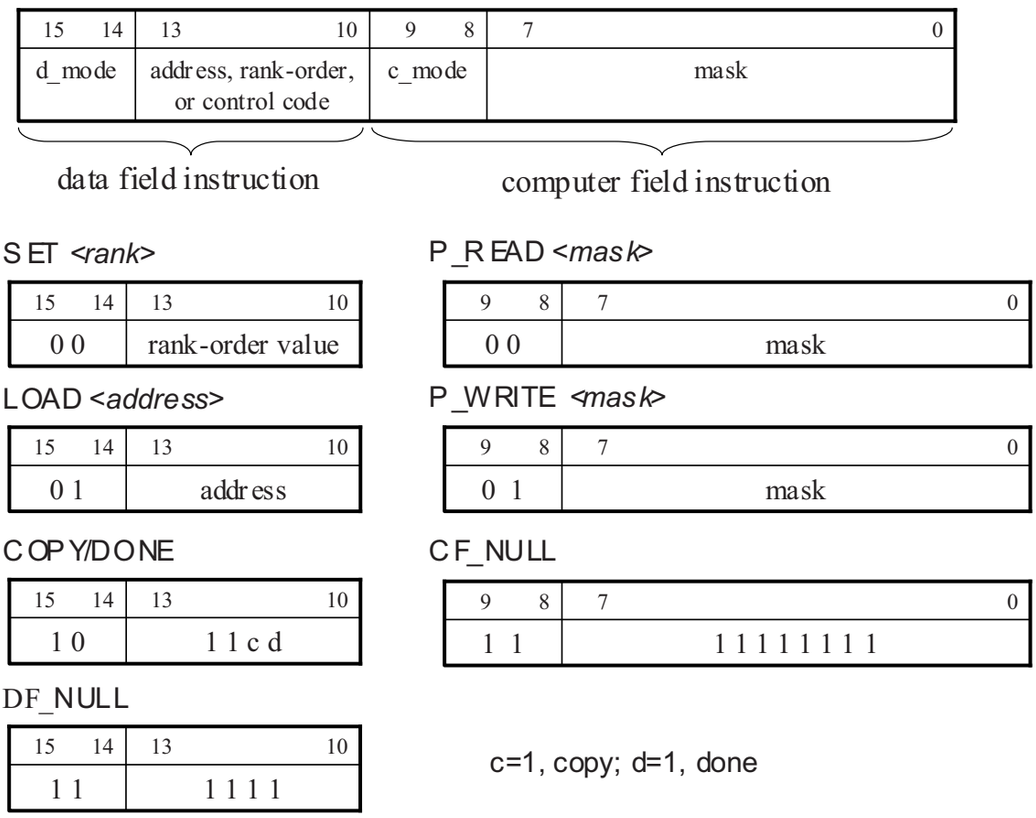


Fig. 23. The format of the instruction set.

To generate instructions to the ROF processor, the complete 1-D non-recursive ROF circuit includes an instruction sequencer, as shown in Fig.24.

Since the instruction set is in the format of long-instruction-word (LIW), the data fetching and ROF computing can be executed in parallel. So, the generated instruction stream can pipeline the ROF iterations, and the data fetching is hidden in each ROF latency. Fig.25 shows the reservation table of the 1-D ROF example. As seen in the reservation table, the first iteration and the second iteration are overlapped at the seventeenth, eighteenth and nineteenth clock steps. At the seventeenth clock step, the second iteration starts with loading a new sample while the first iteration processes the LSB bit-slice. At the eighteenth clock step, the second iteration copies samples from the data field to the computing field, and reads the MSB bit-slice.

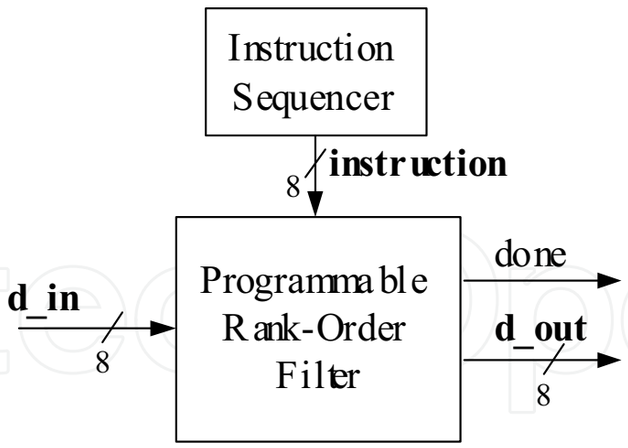


Fig. 24. Block diagram of the 1-D non-recursive ROF.

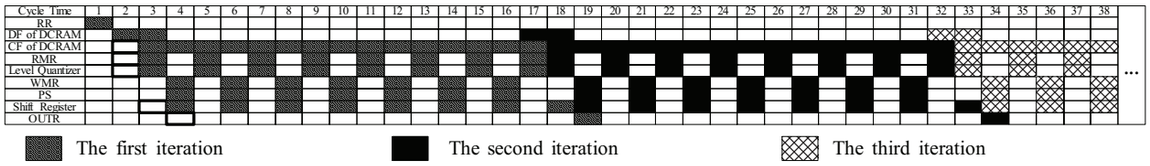


Fig. 25. Reservation table of the 1-D non-recursive ROF.

At the same time, the first iteration prepares the first ROF result for OUTR. At the nineteenth clock step, the first iteration sends the result out while the second iteration performs the first polarization. Thus, the iteration period for each iteration is 15 cycles.

3.5 Application of the proposed ROF processor

In Section 3.4, we use 1-D non-recursive ROF as an example to show the programming of the proposed ROF processor. Due to the programmable design, the proposed ROF processor can implement a variety of ROF applications. The following subsections will illustrate the optimized programs for three examples: 1-D RMF, 2-D non-recursive ROF, and 2-D RMF.

3.6 1-D recursive median filter

The recursive median filtering (RMF) has been proposed for signal smoothing and impulsive noise elimination. It can effectively remove sharp discontinuities of small duration without blurring the original signal. The RMF recursively searches for the median results from the most recent median values and input samples. So, the input window of RMF can be denoted as $\{y_{i-k}, y_{i-k+1}, ..., y_{i-1}, x_i, ..., x_{i+l}\}$, where $y_{i-k}, y_{i-k+1}, ..., y_{i-1}$ are the most recent median values and $x_i, ..., x_{i+l}$ are the input samples, and the result y_i is the $\lceil (l+k+1)/2 \rceil$-th value of the input window.

Fig.26 demonstrates the implementation of the 1-D RMF. To recursively perform RMF with previous median values, the i-th iteration of 1-D RMF loads two inputs to the DCRAM; one is x_{i+l} and the other is y_{i-1}. As shown in Fig.26, the 2-to-1 multiplexer is used to switch the input stream to the data field, controlled by the instruction sequencer; the input stream is from either "**d_in**" or "**d_out**". When the proposed ROF processor receives the input stream, the

program will arrange the data storage as shown in Fig.26. The date storage shows the data reusability of the proposed ROF processor.

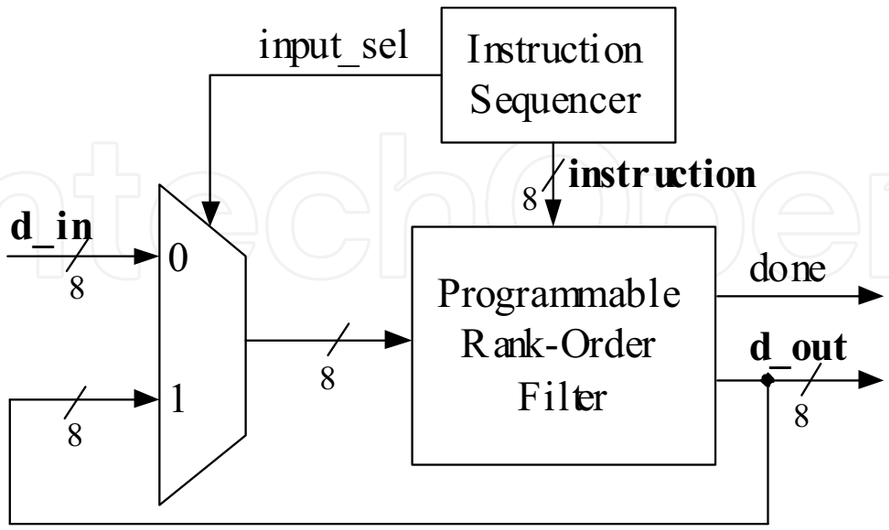


Fig. 26. Block diagram of the 1-D RMF.

As mentioned above, the input stream to the DCRAM comes from either "**d_in**"" or "**d_out**"". The statements of "`set input_sel, 0;`" and "`set input_sel, 1;`" can assert the signal "input_sel" to switch the input source accordingly. The statements of "`LOAD i, CF_NULL;`" and "`LOAD i, CF_NULL;`" is employed for the data stream, as per Fig.27. As seen in Fig.28, the throughput rate is limited by the recursive execution of the 1-D RMF; that is, the second iteration cannot load the newest median value until the first iteration generates the result to the output. However, we still optimized the throughput rate as much as possible. At the twentieth clock step, the program overlaps the first iteration and the second

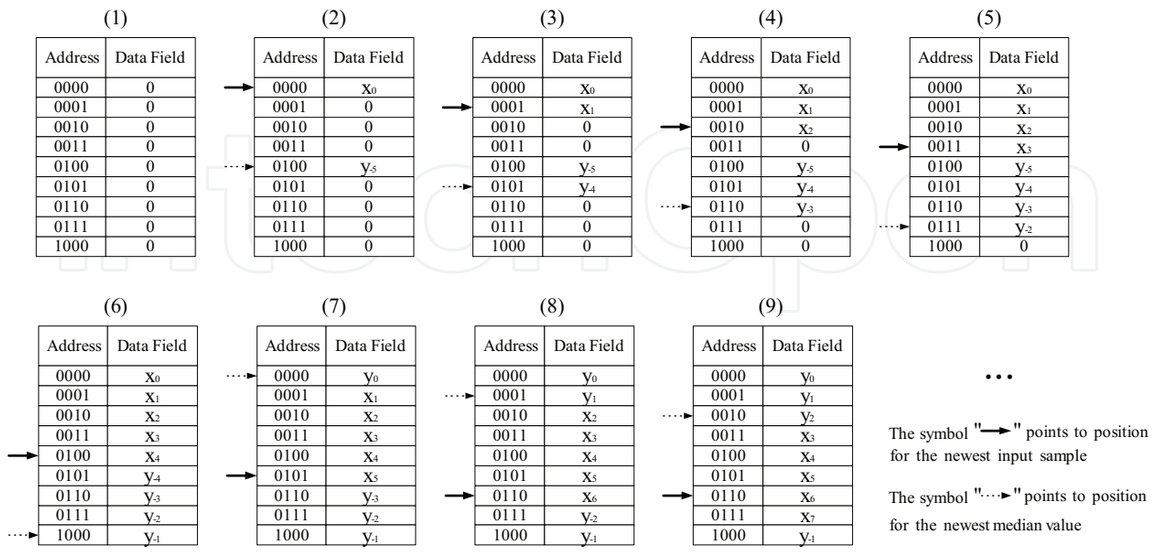


Fig. 27. The flow for data storage of the 1-D RMF.

iteration so that the data fetching and result preparing can be run at the same time. As the result, the sample period is 18 cycles.

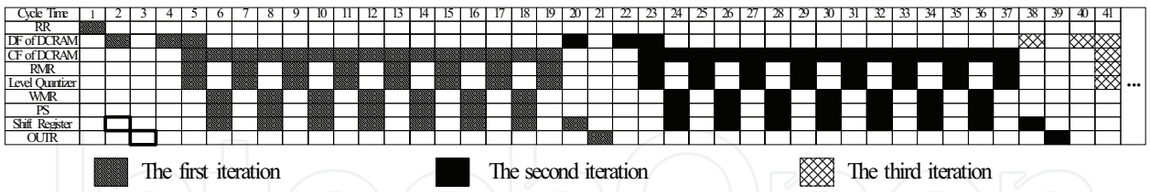


Fig. 28. Reservation table of the 1-D RMF.

3.6.1 2-D non-recursive rank-order filter

Fig.29 illustrates the block diagram for the 2-D non-recursive ROF. From Fig.30, each iteration needs to update three input samples (the pixels in the shadow region) for the 3×3 ROF; that is, only n input samples need to be updated in each iteration for the $n \times n$ ROF. To reuse the windowing data, the data storage is arranged as shown in Fig.31. So, for the 2-D ROF, the data reusability of our process is high; each iteration updates only n input samples for an $n \times n$ window. Given a 2-D $n \times n$ ROF application with n=3 and r=5, the optimized reservation table can be scheduled as Fig.32.

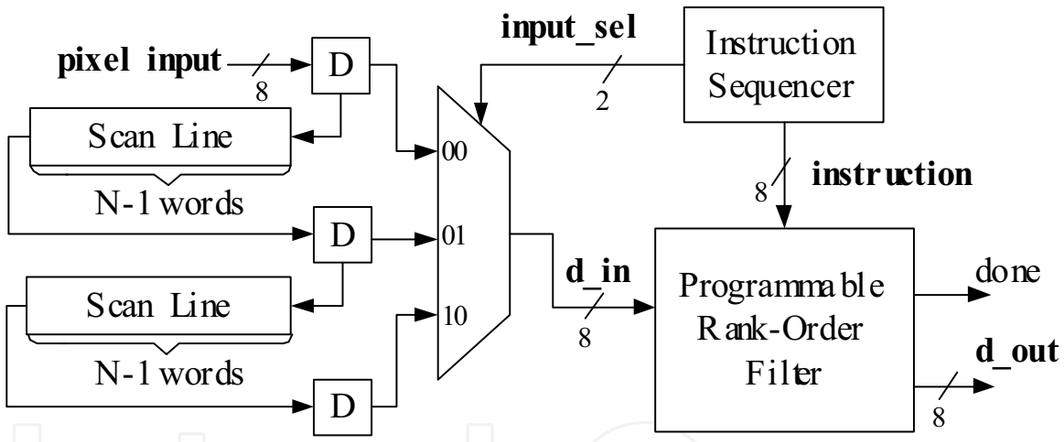


Fig. 29. Block diagram of the 2-D non-recursive ROF with 3-by-3 window.

3.6.2 2-D recursive median filter

Similar to the 1-D RMF, the two-dimensional(2-D) *n-by-n* RMF finds the median value from the window formed by some previous-calculated median values and input values. Fig.33(a) shows the content of the 3×3 window centered at (i, j). At the end of each iteration, the 2-D 3×3 RMF substitutes the central point of the current window with the median value. The renewed point will then be used in the next iteration. The windowing for 2-D RMF iterations is shown in Fig.33(b), where the triangles represent the previous-calculated median values and the pixels in the shadow region are updated at the beginning of each iteration. According to the windowing, Fig.34 illustrates the data storage for high degree of data reusability. Finally, we can implement the 2-D RMF as the block diagram illustrated in Fig.35. Given a 2-D 3×3 RMF application, the optimized reservation table can be scheduled as Fig.36.

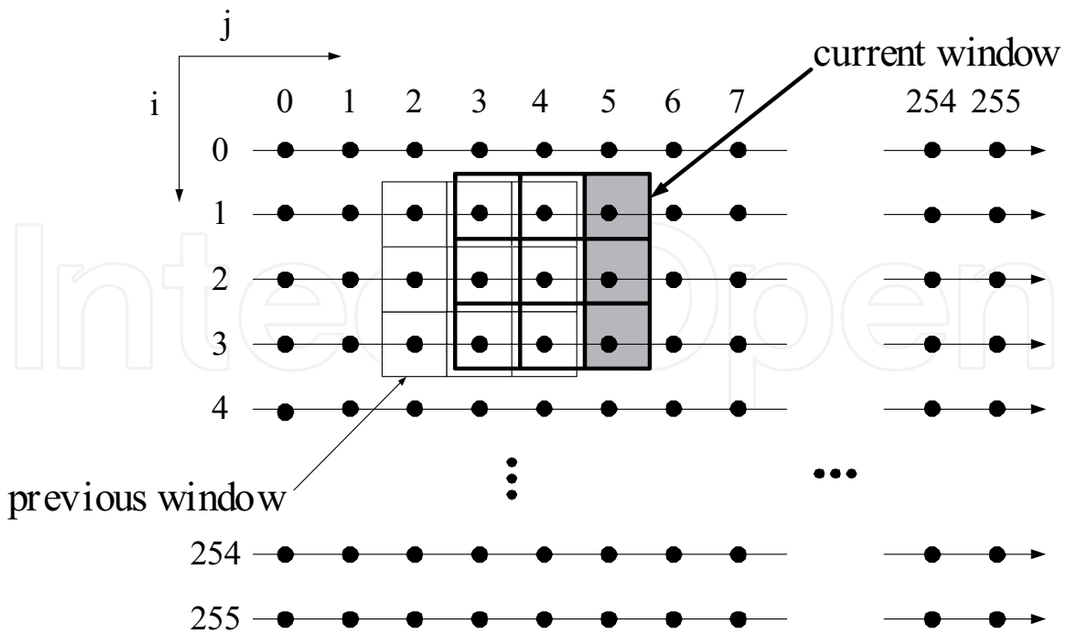


Fig. 30. The windowing of the 3 × 3 non-recursive ROF.

(1)

Address	Data Field
0000	x(3,3)
0001	x(2,3)
0010	x(1,3)
0011	x(3,1)
0100	x(2,1)
0101	x(1,1)
0110	x(3,2)
0111	x(2,2)
1000	x(1,2)

(2)

Address	Data Field
0000	x(3,3)
0001	x(2,3)
0010	x(1,3)
0011	x(3,4)
0100	x(2,4)
0101	x(1,4)
0110	x(3,2)
0111	x(2,2)
1000	x(1,2)

(3)

Address	Data Field
0000	x(3,3)
0001	x(2,3)
0010	x(1,3)
0011	x(3,4)
0100	x(2,4)
0101	x(1,4)
0110	x(3,5)
0111	x(2,5)
1000	x(1,5)

(4)

Address	Data Field
0000	x(3,6)
0001	x(2,6)
0010	x(1,6)
0011	x(3,4)
0100	x(2,4)
0101	x(1,4)
0110	x(3,5)
0111	x(2,5)
1000	x(1,5)

(5)

Address	Data Field
0000	x(3,6)
0001	x(2,6)
0010	x(1,6)
0011	x(3,7)
0100	x(2,7)
0101	x(1,7)
0110	x(3,5)
0111	x(2,5)
1000	x(1,5)

The symbols "→" point to positions for the newest input samples

Fig. 31. The data storage of the 2-D non-recursive ROF.

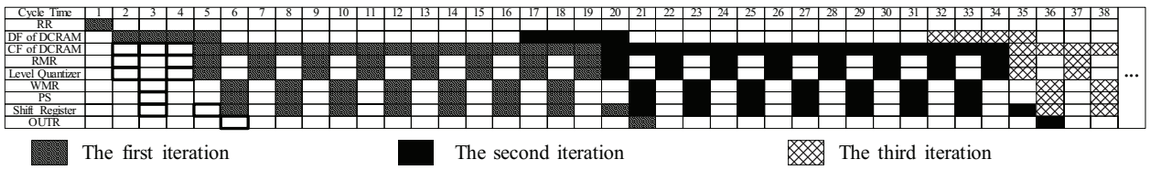
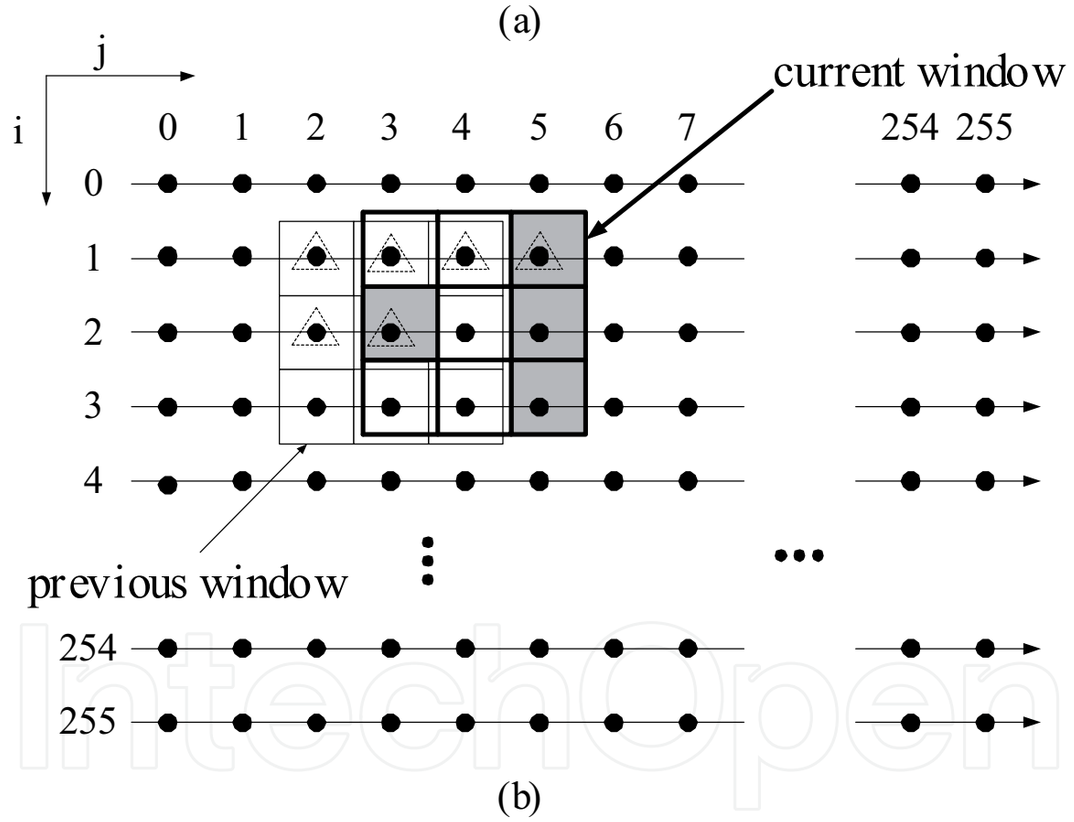


Fig. 32. Reservation table of the 2-D ROF.

y(i-1,j-1)	y(i-1,j)	y(i-1,j+1)
y(i,j-1)	x(i,j)	x(i,j+1)
x(i+1,j-1)	x(i+1,j)	x(i+1,j+1)

(a)



(b)

Fig. 33. (a) The content of the 3×3 window centered at (i, j). (b) The windowing of the 2-D RMF.

3.7 The fully-pipelined DCRAM-based ROF architecture

As seen in Section 3.5, the reservation tables are not tightly scheduled because the dependency of bit-slicing read, threshold decomposition, and polarization forms a cycle. The dependency cycle limits the schedulability of ROF tasks. To increase the schedulability, we further extended the ROF architecture to a fully-pipelined version at the expense of area. The

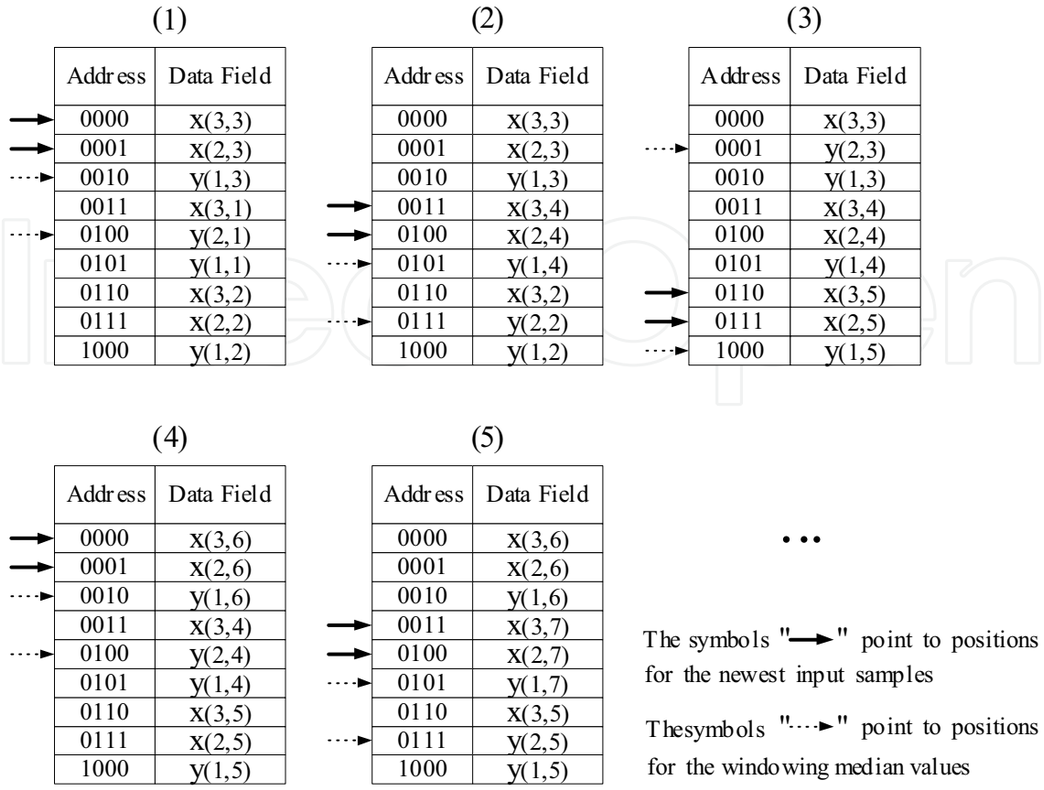


Fig. 34. The data storage of 2-D RMF.

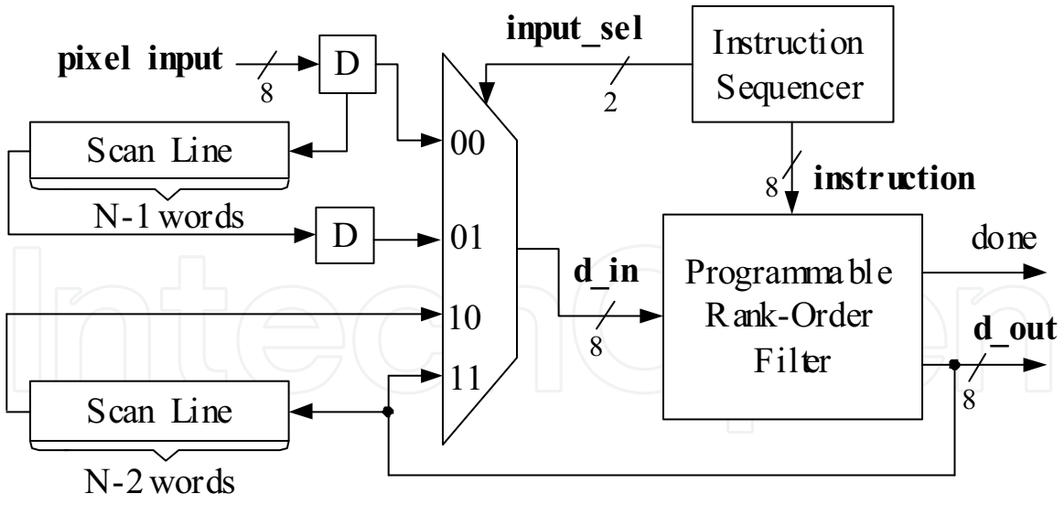


Fig. 35. Block diagram of the 2-D RMF with 3-by-3 window.

fully-pipelined ROF architecture interleaves three ROF iterations with triple computing fields. As shown in Fig. 37, there are three computing fields which process three tasks alternatively. To have the tightest schedule, we pipelined the Level-Quantizer into two stages, LQ1 and LQ2, so the loop (computing field, Level-Quantizer, Shift Register) has three pipeline stages

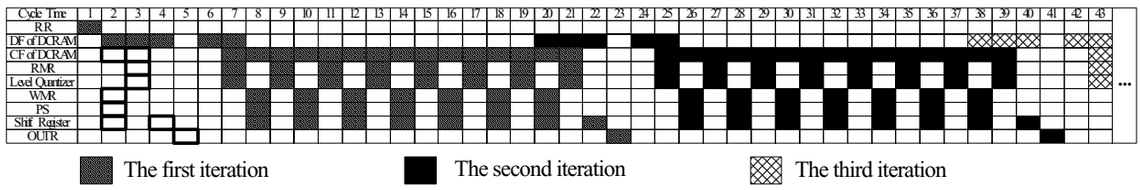


Fig. 36. Reservation table of optimal pipeline for 2-D recursive median filter.

for the highest degree of parallelism. The LQ1 is the FA/HA tree and the LQ2 is the carry generator.

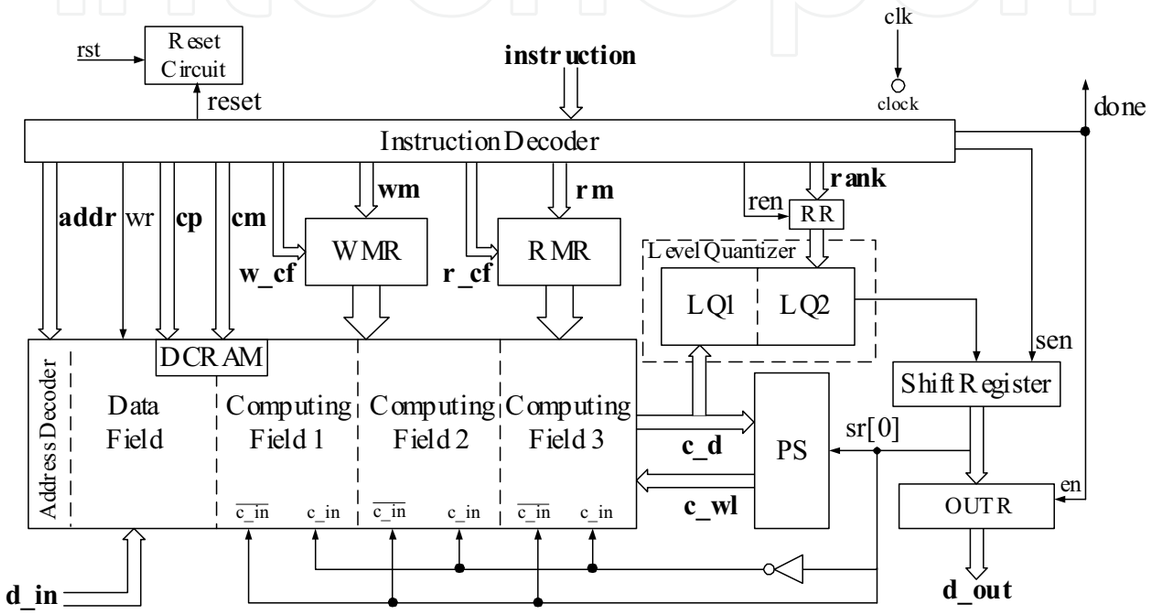


Fig. 37. The fully-pipelined ROF architecture.

Since there exists three iterations being processed simultaneously, a larger memory is required for two more iterations. Hence, we extended the DCRAM to an $(N + 2\delta)$-word memory, where N is the window size of ROF and δ is the number of updating samples for each iteration. The value of δ is 1 for 1-D ROF, and n for 2-D n-by-n ROF. To correctly access the right samples for each iteration, the signal "**cm**" is added to mask the unwanted samples during the copy operation. In each computing field, the unwanted samples are stored as all zeros. Doing so, the unwanted samples will not affect the rank-order results. Fig.38 illustrates the modified computing cell for fully-pipelined ROF. The INV5 and INV6 are replaced with GATE1 and GATE2. When "cm[i]" is '0' the computing cell will store '0'; otherwise, the computing cell will have the copy of the selected sample from the data cell. Finally, we use "**cp**", "**w_cf**", and "**r_cf**" to selectively perform *read*, *write*, or *copy* on computing fields. To efficiently program the fully-pipelined architecture, the instruction set is defined as shown in Fig.39. The fields <*c_cf*> of `COPY`,<*w_cf*> of `P_WRITE`, and <*r_cf*> of `P_READ` are used to control "**cp**", "**w_cf**", and "**r_cf**". Given a 1-D non-recursive rank order filter application with N=9 and r=3, the reservation table can be optimized as shown in Fig.40.

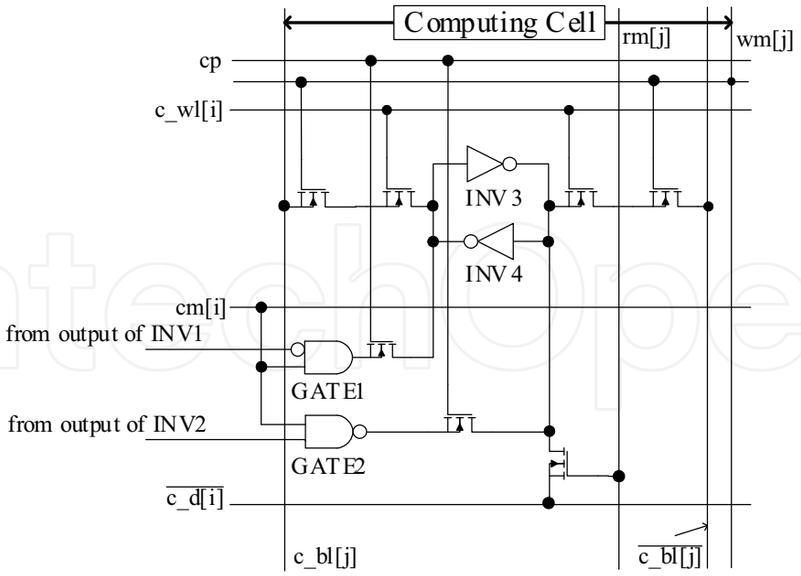


Fig. 38. A modified circuit of computing cell for fully-pipelined ROF.

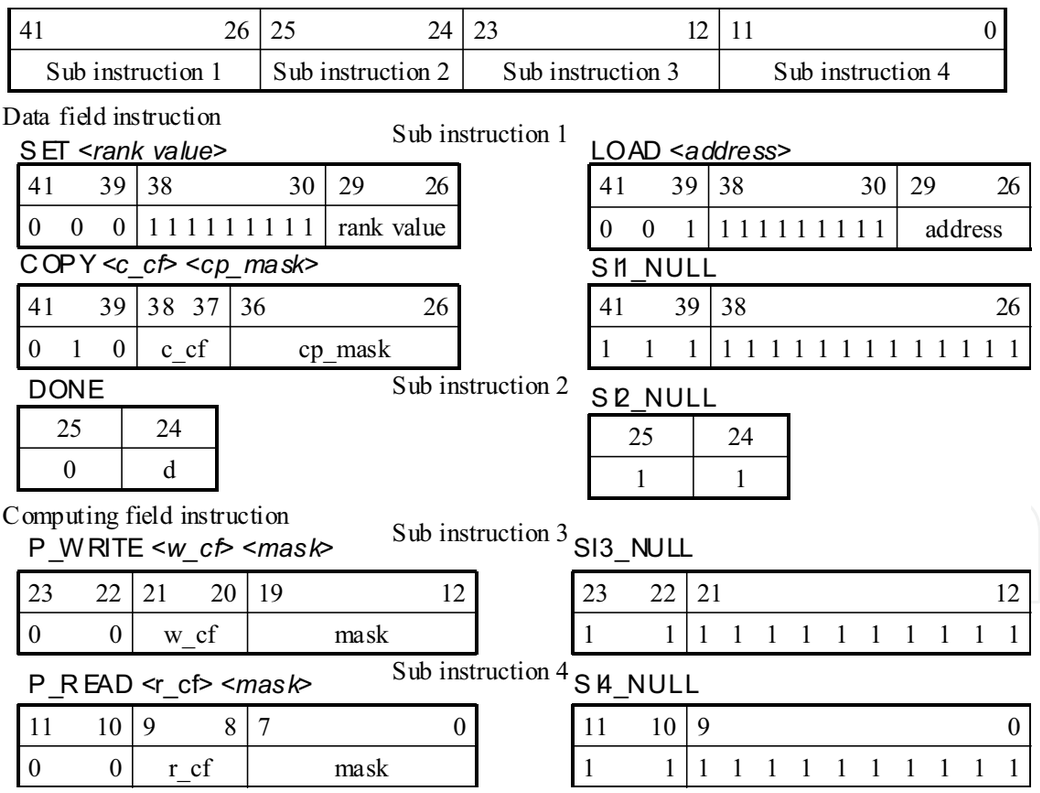


Fig. 39. The format of the extended instruction set for the fully-pipelined ROF architecture.

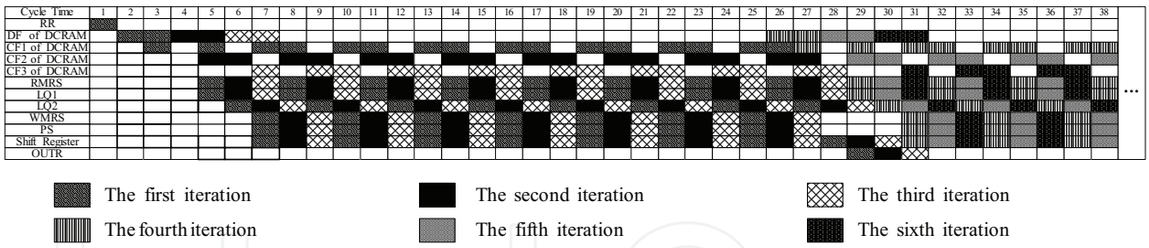


Fig. 40. Reservation table of the 1-D non-recursive ROF for fully-pipelined ROF architecture.

3.8 Chip design and simulation results

To exercise the proposed architecture, we have implemented the ROF architecture, shown in Fig.16, using TSMC 0.18 μm 1P6M technology. First, we verified the hardware in VHDL at the behavior level. The behavior VHDL model is cycle-accurate. As the result of simulation, the implementations of the above examples are valid. Fig.41 and Fig.42 demonstrate the results of VHDL simulations for the 2-D ROF and RMF, respectively. Fig.41(a) is a noisy "Lena" image corrupted by 8% of impulsive noise. After being processed by 2-D ROFs with r=4, 5, and 6, the denoise results are shown in Fig.41(b), (c), and (d), respectively. Fig.42(a) is a noisy "Lena"

Fig. 41. Simulation results of a 2-D ROF application. (a) The noisy "Lena" image corrupted by 8% of impulsive noise. (b) The "Lena" image processed by the 3×3 4th-order filtering. (c) The "Lena" image processed by the 3×3 5th-order filtering. (d) The "Lena" image processed by the 3×3 6th-order filtering.

image corrupted by 9% of impulsive noise. After being processed by the 2-D 3×3 RMF, the denoise result is shown in Fig.42(b). The results are the same as those of Matlab simulation.

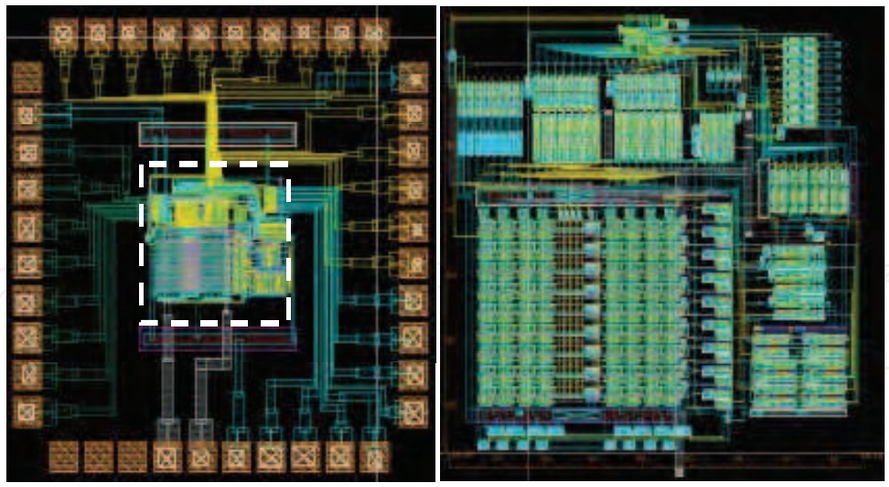
(a) (b)

Fig. 42. Simulation results of a 2-D RMF application. (a) The noisy "Lena" image corrupted by 9% of impulsive noise. (b) The "Lena" image processed by the 3×3 RMF.

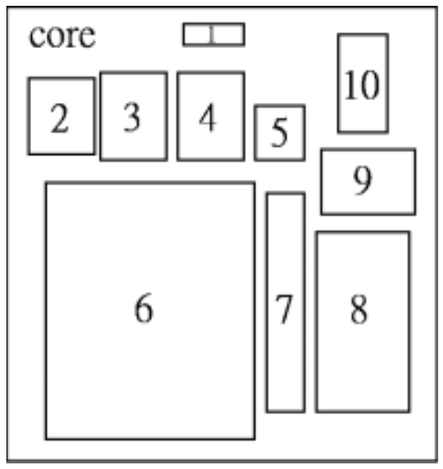
Upon verifying the proposed ROF processor using the cycle-accurate behavior model, we then implemented the processor in the fully-custom design methodology. Because of high regularity of memory, the proposed memory-based architecture saves the routing area while comparing with the logic-based solutions. Fig.43 (a) shows the overall chip layout and the dash-lined region is the core. The die size is $1063.57 \times 1069.21 \mu m^2$ and the pinout count is 40. Fig.43 (b) illustrates the detail layout of the ROF core. The core size is $356.1 \times 427.7 \mu m^2$ and the total transistor count is 3942. Fig.43 (c) illustrates the floorplan and placement. The physical implementation has been verified by the post-layout simulation. Table 8 shows the result of timing analysis, obtained from NanoSim. As seen in the table, the critical path is the path 3 and the maximum clock rate can be 290 MHz at 3.3V and 256 MHz at 1.8V. As the result of post-layout simulation, the power dissipation of the proposed ROF is quite low. For the 1-D/2-D ROFs, the average power consumption of the core is 29mW at 290MHz or 7mW at 256MHz. The performance sufficiently satisfies the real-time requirement of video applications in the formats of QCIF, CIF, VGA, and SVGA. The chip is submitting to Chip Implementation Center (CIC), Taiwan for the fabrication.

Path	Description	1.8V supply	3.3V supply
1	From the output of RR to the input of the shift register.	1.2 ns	0.78 ns
2	From the output of RMR, thru DCRAM to the input of the PS.	1.8 ns	1.1 ns
3	From the output of RMR, thru DCRAM and the Level-Quantizer, to the input of the shift register.	3.9 ns	3.44 ns
4	From the shift register, thru the inverter connected to "c_in", to the SRAM cell of the computing field.	3.02 ns	1.96 ns
5	From "**d_in**" to the SRAM cell of the data field.	3.05 ns	1.85 ns
6	From the SRAM cell of the data field to the SRAM cell of the computing field.	1.24 ns	1.09 ns

Table 8. Timing analysis of the proposed ROF processor.



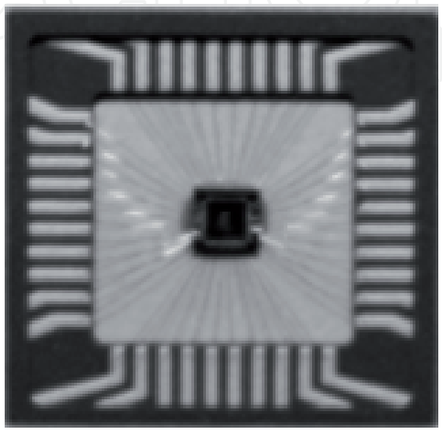
(a) (b)



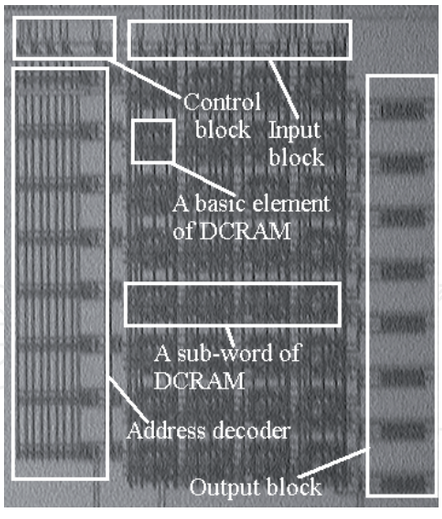
(c)

Fig. 43. The result of chip design using TSMC 0.18um 1P6M technology. (a) The chip layout of proposed rank-order filter. (b) The core of the proposed ROF processor. (c) The floorplan and placement of (b). (1: Instruction decoder; 2: Reset circuit, 3: WMR, 4: RMR, 5: RR, 6: DCRAM, 7: PS; 8: Level Quantizer; 9: Shift Register; 10: OUTR.)

Furthermore, We have successfully built a prototype which is composed of a FPGA board and DCRAM chips to validate the proposed architecture before fabricating the custom designed chip. The FPGA board is made by Altera and the FPGA type is APEX EP20K. The FPGA board can operate at 60 MHz at the maximum. The DCRAM chip was designed by full-custom CMOS technology. Fig.44(a) shows the micrograph of the DCRAM chip. The chip implements a subword part of DCRAM and the Fig.44(b) illustrates the chip layout. The fabricated DCRAM chip was layouted by full-custom design flow using TSMC 0.35 2P4M technology. As shown in Fig.45, with the supply voltage of 3.3V, the DCRAM chip can operate at 25 MHz. Finally, we successfully integrated the FPGA board and the DCRAM chips into a prototype as shown in Fig.46 The prototype was validated with ROF algorithms mentioned above.



(a)



(b)

Fig. 44. (a) The micrograph of DCRAM chip. (b) The layout of the DCRAM chip.

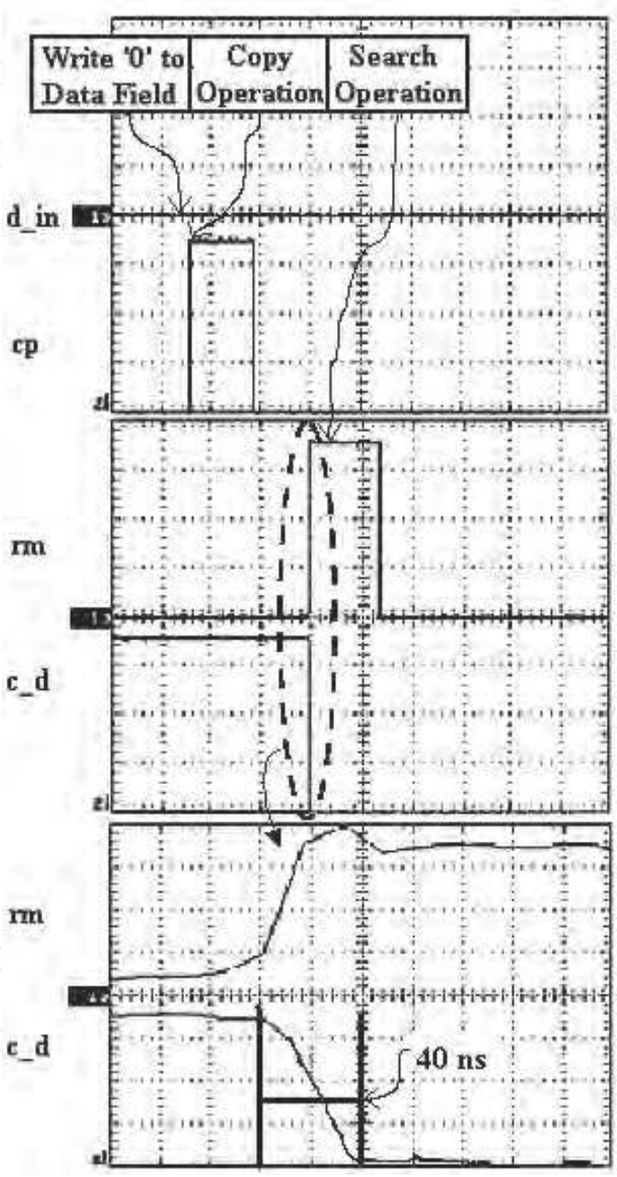


Fig. 45. Measured waveform of the DCRAM chip.

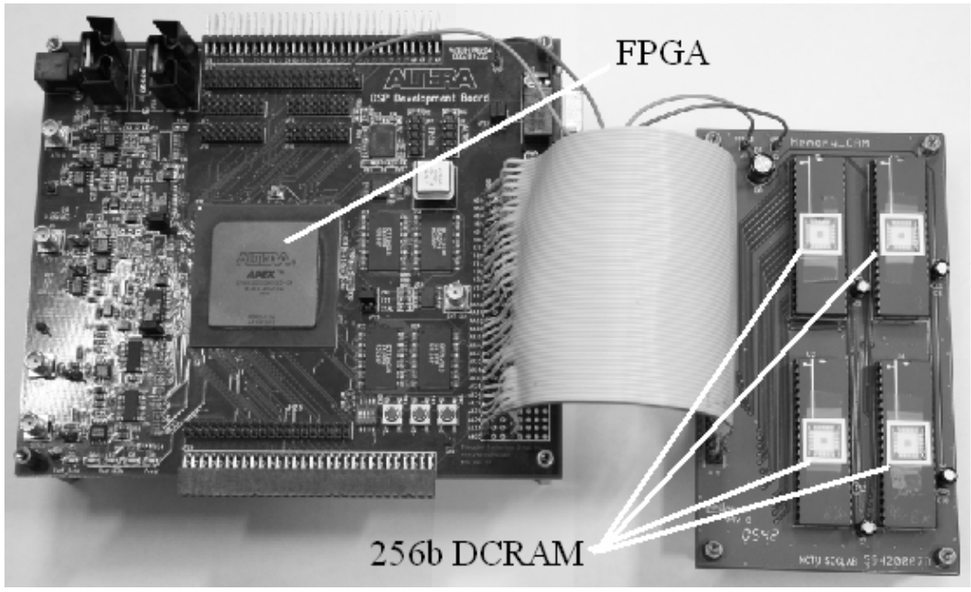


Fig. 46. The system prototype of rank-order filtering processor.

4. Conclusion

In order to further extend the battery life of capsule endoscope, this paper mainly focus on a series of mathematical statistics to systematically analyze the color sensitivity in GI images from the RGB color space domain to the 2-D DCT spatial frequency domain. According to the analysis results, an improved ultra-low-power subsample-based GICam image compression processor are proposed for capsule endoscope or swallowable imaging capsules. we make use of the subsample technique to reduce the memory requirements of G1, G2 and B components according to the analysis results of DC/AC coefficients in 2-D DCT domain. As shown in the simulation result, the proposed image compressor can efficiently save 38.5% more power consumption than previous GICam one (11), and can efficiently reduce image size by 75% at least for each sampled gastrointestinal image. Therefore, video sequences totally reduce size by 75% at least. Furthermore, the proposed image compressor has lower area and lower operation frequency according to the comparison results. It can fit into the existing designs.

Forthemore, we have proposed an architecture based on a maskable memory for rank-order filtering. This paper is the first literature using maskable memory to realize ROF. Driving by the generic rank-order filtering algorithm, the memory-based architecture features high degree of flexibility and regularity while the cost is low and the performance is high. With the LIW instruction set, this architecture can be applied for arbitrary ranks and a variety of ROF applications, including recursive and non-recursive algorithms. As shown in the implementation results, the core of the processor has high performance and low cost. The post-layout simulation shows that the power consumption can be as low as 7 mW at 256 MHz. The processing speed can meet the real-time requirement of image applications in the QCIF, CIF, VGA, or SVGA formats.

5. References

[1] G. Iddan, G. Meron, A. Glukhovsky, and P. Swain, “Wireless capsule endoscopy," Nature, vol. 405, pp. 417-418, May 25, 2000.

[2] Shinya Itoha, Shoji Kawahitob, Tomoyuki Akahoric, and Susumu Terakawad, “Design and implementation of a one-chip wireless camera device for a capsule endoscope," SPIE, vol. 5677, pp. 108-118, 2005.

[3] F. Gong. P. Swain, and T. Mills, “Wireless endoscopy," Gastrointestinal Endoscopy, vol.51, no. 6, pp. 725-729, June 2000.

[4] H. J.Park, H.W. Nam, B.S. Song, J.L. Choi, H.C. Choi, J.C. Park, M.N. Kim, J.T. Lee, and J.H. Cho, “Design of bi-directional and multi-channel miniaturized telemetry module for wireless endoscopy," in Proc. of the 2nd Annual Intl IEEE-EMBS Special Topic Conference on Microtechnologies in Medicine and Biology, May 2-4, 2002, Madison, USA, pp. 273-276.

[5] http://www.givenimaging.com/Cultures/en-US/given/english

[6] http://www.rfsystemlab.com/

[7] M. Sendoh, k. Ishiyama, and K.-I. Arai, “Fabrication of Magnetic Actuator for Use in a Capsule Endoscope," IEEE Trans. on Magnetics, vol. 39, no. 5, pp. 3232-3234, September 2003.

[8] Louis Phee, Dino Accoto, Arianna Menciassi*, Cesare Stefanini, Maria Chiara Carrozza, and Paolo Dario, “Analysis and Development of Locomotion Devices for the Gastrointestinal Tract," IEEE Trans. on Biomedical Engineering, vol. 49, no. 6, JUNE 2002.

[9] Shaou-Gang Miaou, Shih-Tse Chen, and Fu-Sheng Ke, “Capsule Endoscopy Image Coding Using Wavelet-Based Adaptive Vector Quantization without Codebook Training," International Conference on Information Technology and Applications (ICITA), vol. 2, pp. 634-637, July 2005.

[10] Shaou-Gang Miaou, Shih-Tse Chen, and Chih-Hong Hsiao, “A wavelet-based compression method with fast quality controlling capability for long sequence of capsule endoscopy images," IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing (NSIP), pp. 34-34, 2005.

[11] M. Lin, L. Dung, and P. Weng, “An Ultra Low Power Image Compressor for capsule Endoscope," BioMedical Engineering Online 2006, vol. 5:14.

[12] X. Xie, G. Li, X. Chen, X. Li, and Z. Wang, “A Low Power Digital IC Design Inside the Wireless Endoscopic Capsule", IEEE Journal of Solid-State Circuits, vol. 41, no. 11, pp. 2390-2400, November 2006.

[13] K. Wahid, S-B. Ko, and D. Teng, “Efficient Hardware Implementation of an Image Compressor for Wireless Capsule Endoscopy Applications", Proceedings of the IEEE International Joint Conference on Neural Networks pp. 2762-2766, 2008;

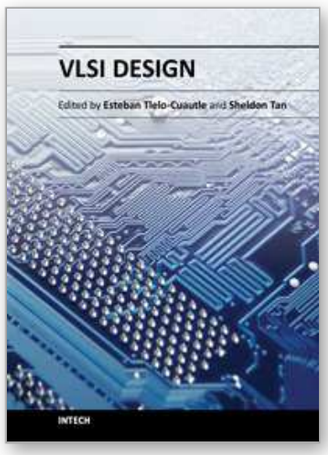
[14] Xinkai Chen; Xiaoyu Zhang; Linwei Zhang; Xiaowen Li; Nan Qi; Hanjun Jiang; Zhihua Wang, “A Wireless Capsule Endoscope System With Low-Power Controlling and Processing ASIC", IEEE Trans. on Biomedical Circuits and Systems, vol. 3, no. 1, pp.11-22, Feb. 2009.

[15] H.A. Peterson, H. Peng, J. H. Morgan, and W. B. Pennebaker, “Quantization of color image components in the DCT domain" SPIE , Human Vision, Visual Processing, and Digital Display II, vol.1453, 1991.

[16] Dr. R.W.G. Hunt, “Measuring Colour," Fountain Press, 1998.
[17] Henry R. Kang “Color Technology For Electronic Imaging Devices," SPIE Optical Engineering Press, 1997.
[18] J.Ziv and A. Lempel, “A universal algorithm for sequential data compression," IEEE Trans. on Inform. Theory, vol. 23, pp. 337-343, May 1977.
[19] Vasudev Bhaskaran, and Konstantinos Kon stantinides “Images and Video Compression Standards : Alogorithms and Architectures, Second edition," Kliwer Academic Publishers.
[20] Meng-Chun Lin, Lan-Rong Dung, and Ping-Kuo Weng “An Improved Ultra-Low-Power Subsample based Image Compressor for Capsule Endoscope," Medical Informatics Symposium in Taiwan (MIST), 2006.
[21] Gi-Shih Lien, Chih-Wen Liu, Ming-Tsung Teng, and Yan-Min Huang, “ Integration of Two Optical Image Modules and Locomotion Functions in Capsule Endoscope Applications," The 13th IEEE International Symposium on Consumer Electronics, pp.828-829, 2009.
[22] Mao Li, Chao Hu, Shuang Song, Houde Dai, and Max Q.-H. Meng, “Detection of Weak Magnetic Signal for Magnetic Localization and Orientation in Capsule Endoscope," Proceedings of the IEEE International Conference on Automation and Logistics Shenyang, China, pp.900-905, August 2009.
[23] Chao Hu, Max Q.-H. Meng, Li Liu, Yinzi Pan, and Zhiyong Liu “Image Representation and Compression for Capsule Endoscope Robot," Proceedings of the 2009 IEEE International Conference on Information and Automation, pp.506-511, June 2009.
[24] Jing Wu, and Ye Li, “Low-complexity Video Compression for Capsule Endoscope Based on Compressed Sensing Theory," 31st Annual International Conference of the IEEE EMBS Minneapolis, pp.3727-3730, Sep. 2009.
[25] Jinlong Hou, Yongxin Zhu, Le Zhang, Yuzhuo Fu, Feng Zhao, Li Yang, and Guoguang Rong, “Design and Implementation of a High Resolution Localization System for In-vivo Capsule Endoscopy," 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing, pp.209-214, 2009.
[26] Chang Cheng, Zhiyong Liu and Chao Hu, and Max Q.-H. Meng “A Novel Wireless Capsule Endoscope With JPEG Compression Engine," Proceedings of the 2010 IEEE International Conference on Automation and Logistics, pp.553-558, Aug. 2010.
[27] D.H. Kang, J.H. Choi, Y.H. Lee, and C. Lee, “Applications of a DPCM system with median predictors for image coding,” IEEE Trans. Consumer Electronics, vol.38, no.3, pp.429-435, Aug. 1992.
[28] H. Rantanen, M. Karlsson, P. Pohjala, and S. Kalli, “Color video signal processing with median filters,” IEEE Trans. Consumer Electron., vol.38, no.3, pp.157-161, Apr. 1992.
[29] T. Viero, K. Oistamo, and Y. Neuvo, “Three-dimensional median-related filters for color image sequence filtering,” IEEE Trans. Circuits Syst. Video Technol., vol.4, no.2, pp.129-142, Apr. 1994.
[30] X. Song, L. Yin, and Y. Neuvo, “Image sequence coding using adaptive weighted median prediction,” Signal Processing VI, EUSIPCO-92, Brussels, pp.1307-1310, Aug. 1992.
[31] K. Oistamo and Y. Neuvo, “A motion insensitive method for scan rate conversion and cross error cancellation,” IEEE Trans. Consumer Electron., vol.37, pp.296-302, Aug. 1991.

[32] P. Zamperoni, "Variation on the rank-order filtering theme for grey-tone and binary image enhancement," IEEE Int. Conf. Acoust., Speech, Signal Processing, pp.1401-1404, 1989.
[33] C.T. Chen and L.G. Chen, "A self-adjusting weighted median filter for removing impulse noise in images," Int. Conf. Image Processing, pp.16-19, Sept. 1996.
[34] D. Yang and C. Chen, "Data dependence analysis and bit-level systolic arrays of the median filter," IEEE Trans. Circuits and Systems for Video Technology, vol.8, no.8, pp.1015-1024, Dec. 1998.
[35] T. Ikenaga and T. Ogura, "CAM2: A highly-parallel two-dimensional cellular automation architecture," IEEE Trans. Computers, vol.47, no.7, pp.788-801, July 1998.
[36] L. Breveglieri and V. Piuri, "Digital median filter," Journal of VLSI Signal Processing, vol.31, pp.191-206, 2002.
[37] C. Chakrabarti, "Sorting network based architectures for median filters," IEEE Trans. Circuits ans Systems II: Analog and Digital Signal Processing, vol.40, pp.723-727, Nov. 1993.
[38] C. Chakrabarti, "High sample rate architectures for median filters," IEEE Trans. Signal Processing, vol.42, no.3, pp.707-712, March 1994.
[39] L. Chang and J. Lin, "Bit-level systolic array for median filter," IEEE Trans. Signal Processing, vol.40, no.8, pp.2079-2083, Aug. 1992.
[40] C. Chen, L. Chen, and J. Hsiao, "VLSI implementation of a selective median filter," IEEE Trans. Consumer Electronics, vol.42, no.1, pp.33-42, Feb. 1996.
[41] M.R. Hakami, P.J. Warter, and C.C. Boncelet, Jr., "A new VLSI architecture suitable for multidimensional order statistic filtering," IEEE Trans. Signal Processing, vol.42, pp.991-993, April 1994.
[42] Hatirnaz, F.K. Gurkaynak, and Y. Leblebici, "A compact modular architecture for the realization of high-speed binary sorting engines based on rank ordering," IEEE Inter. Symp. Circuits and Syst., Geneva, Switzerland, pp.685-688, May 2000.
[43] A.A. Hiasat, M.M. Al-lbrahim, and K.M. Gharailbeh, "Design and implementation of a new efficient median filtering algorithm," IEE Proc. Image Signal Processing, vol.146, no.5, pp.273-278, Oct. 1999.
[44] R.T. Hoctor and S.A. Kassam, "An algorithm and a pipelined architecture for order-statistic determination and L-filtering," IEEE Trans. Circuits and Systems, vol.36, no.3, pp.344-352, March 1989.
[45] M. Karaman, L. Onural, and A. Atalar, "Design and implementation of a general-purpose median filter unit in CMOS VLSI," IEEE Journal of Solid State Circuits, vol.25, no.2, pp.505-513, April 1990.
[46] C. Lee, P. Hsieh, and J, Tsai, "High-speed median filter designs using shiftable content-addressable memory," IEEE Trans. Circuits and Systems for Video Technology, vol.4, pp.544-549, Dec. 1994.
[47] C.L. Lee and C. Jen, "Bit-sliced median filter design based on majority gate," IEE Proc.-G Circuits, Devices and Systems, vol.139, no.1, pp.63-71, Feb. 1992.
[48] L.E. Lucke and K.K. Parchi, "Parallel processing architecture for rank order and stack filter," IEEE Trans. Signal Processing, vol.42, no.5, pp.1178-1189, May 1994.
[49] K. Oazer, "Design and implementation of a single-chip 1-D median filter," IEEE Trans. Acoust., Speech, Signal Processing, vol.ASSP-31, no.4, pp.1164-1168, Oct. 1983.

[50] D.S. Richards, "VLSI median filters," IEEE Trans. Acoust., Speech, and Signal Processing, vol.38, pp.145-153, January 1990.
[51] G.G. Boncelet, Jr., "Recursive algorithm and VLSI implementation for median filtering," IEEE Int. Sym. on Circuits and Systems, pp.1745-1747, June 1988.
[52] C. Henning and T.G. Noll, "Architecture and implementation of a bitserial sorter for weighted median filtering," IEEE Custom Integrated Circuits Conference, pp.189-192, May 1998.
[53] C.C Lin and C.J. Kuo, "Fast response 2-D rank order filter by using max-min sorting network," Int. Conf. Image Processing, pp.403-406, Sept. 1996.
[54] M. Karaman, L. Onural, and A. Atalar, "Design and implementaion of a general purpose VLSI median filter unit and its application," IEEE Int. Conf. Acoustics, Speech, and Signal Processing, pp.2548-2551, May 1989.
[55] J. Hwang and J. Jong, "Systolic architecture for 2-D rank order filtering," Int. Conf. Application-Specific Array Processors, pp.90-99, Sept. 1990.
[56] I. Pitas, "Fast algorithms for running ordering and max/min calculation," IEEE Trans. Circuits and Systems, vol.36, no.6, pp.795-804, June 1989.
[57] O. Vainio, Y. Neuvo, and S.E. Butner, "A signal processor for median-based algorithm," IEEE Trans. Acoustics, Speech, and Signal Processing, vol.37, no.9, pp.1406-1414, Sept. 1989.
[58] H. Yu, J. Lee, and J, Cho, "A fast VLSI implementation of sorting algorithm for standard median filters," IEEE Int. ASIC/SOC Conference, pp.387-390, Sptember 1999.
[59] J.P. Fitch, "Software and VLSI algorithm for generalized renked order filtering," IEEE Trans. Circuits and Systems, vol.CAS-34, no.5, pp.553-559, May 1987.
[60] M.Karaman and L. Onural, "New radix-2-based algorithm for fast median filtering," Electron. Lett., vol.25, pp.723-724, May 1989.
[61] J.P. Fitch, "Software and VLSI Algorithms for Generalized Ranked Order Filtering," IEEE Trans. Circuits and Syst., vol.CAS-34, no.5, pp.553-559, May 1987.
[62] B.K. Kar and D.K. Pradhan, "A new algorithm for order statistic and sorting," IEEE Trans. Signal Processing, vol.41, no.8, pp.2688-2694, Aug. 1993.
[63] V.A. Pedroni, "Compact hamming-comparator-based rank order filter for digital VLSI and FPGA implementations," IEEE Int. Sym. on Circuits and Systems, vol.2, pp.585-588, May 2004.
[64] Shobha Singh, Shamsi Azmi, Nutan Agrawal, Penaka Phani and Ansuman Rout, "Architecture and design of a high performance SRAM for SOC design," IEEE Int. Sym. on VLSI Design, pp.447-451, Jan 2002.



VLSI Design

Edited by Dr. Esteban Tlelo-Cuautle

ISBN 978-953-307-884-7
Hard cover, 290 pages
Publisher InTech
Published online 20, January, 2012
Published in print edition January, 2012

This book provides some recent advances in design nanometer VLSI chips. The selected topics try to present some open problems and challenges with important topics ranging from design tools, new post-silicon devices, GPU-based parallel computing, emerging 3D integration, and antenna design. The book consists of two parts, with chapters such as: VLSI design for multi-sensor smart systems on a chip, Three-dimensional integrated circuits design for thousand-core processors, Parallel symbolic analysis of large analog circuits on GPU platforms, Algorithms for CAD tools VLSI design, A multilevel memetic algorithm for large SAT-encoded problems, etc.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen