Very Low Cost Design and Implementation of Block Truncation Coding for Image Compression

Sajed Zadeh Dadashi and Sahar Zadeh Dadashi, *Member, IACSIT*

***Abstract*—One of image compression algorithms is block truncation coding that has lower efficiency in compare with some other algorithms, for example, JPEG and JPEG2000, but also has low cost and complexity and can be used in many applications. Many of image compression algorithms were implemented using FPGAs to work in fast digital systems. In this paper, BTC is implemented using some parallel microcontrollers that have acceptable speed and very low cost. AVR microcontroller has many applicable instructions, but has not the division instruction. Using the proposed division algorithm in this paper, this operation can be executed very fast.**

***Index Terms*—BTC, block average, bit map, AVR microcontrollers, division.**

I. Introduction

In many digital image systems, specially in surveillance systems, one image compression algorithm used to reduce the image size to increment the use of memory. This image compression algorithm must have suitable efficiency and high speed. JPEG and JPEG2000 are examples of these compression algorithms that have high efficiency and compression ratio, but also have high complexity and cost[1],[2].

In every application, the way of hardware implementation is selected based on acceptable speed, fault tolerancy, cost and circuit size. There are two main ways to implement hardware based systems. The advent of high-density field programmable gate arrays (FPGAs), in combination with new synthesis tools, has made it relatively easy to produce programmable custom hardware[3]. Implementations on FPGAs can provide high performance within certain design constraints, demonstrating speedups of orders of magnitude over conventional machines [4]. In FPGA based systems, circuits are designed as block diagrams and connections between them. Then, by using of hardware description languages(for example,VHDL) or prepared circuit blocks, the system is implemented. FPGAs have high speed and many facilities, but because of high cost, are not acceptable in many applications. One of these implemented algorithms is block truncation coding (BTC) [5],[6]. BTC is a simple and effective technique for image compression that applies lossy compression principles [7]. another way is microcontroller based systems that can accept instructions in their memory and execute them. One of the main problems of low cost microcontrollers is their low speed in compare with FPGAs. In some applications, using of some parallel and pipelined microcontrollers can overcome this problem.

In Section II, the BTC algorithm is explained. Section III, explains the design of the system using some processing units. Section IV, is about the new proposed division operation in AVR microcontrollers. In Section V, the system is implemented and the speed analysis is described in Section VI. Finally, section VII, explain the conclusion.

II. BTC Image Compression Algorithm

The BTC algorithm [7] divides the image into small rectangular blocks of pixels(for example, 8×8). The compression is achieved by producing a bit map for the quantized block and two 8-bits quantization levels. Increasing the block size will increase the compression ratio at the cost of reduced quality of the restored image.

For each block, the average value is calculated. A two-level quantization is performed for the entire block so that a '0' value is stored for the pixels with values smaller than the average, and the rest of the pixels are represented by the value '1'.These '0's and '1's produce the bit map of image block. Therefore, one '0' or '1' is exist in bit map instead of 8 bits in original image.

Moreover, two gray levels(low and high) are produced for each block. One of them is low average that represents the average of the gray levels for the pixels whose gray level is less than the block average. The other is high aveage that relates to the average of the gray levels for the pixels have greater value than the block average. Therefore, for each block that have 64 pixels(64 bytes), 10 bytes are stored in the compressed version, 8 bytes that relates to the bit map of the block(one bit for every pixel) and 2 bytes that represents the block low average and block high average. As a result, the compression ratio is 6.4.In image reconstruction process, '0' is replaced by block low average and '1' is replaced by block high average.

III. System Design

A. Using of Parallel Microcontrollers

Suppose that there is only one block compression unit in the system. When this unit gets the pixels of one block, must process the operations and produce the bit map, block low average and block high average. therefore, in the time of compression of this block, receiving of another block is delayed. Even if this unit can receive all blocks of this image, receiving of another image is delayed. For this reason,

Manuscript received August 15, 2012; revised October 6, 2012.

The authors are with the Department of Computer Engineering, Young Researchers Club, Roudsar and Amlash Branch, Islamic Azad University, Roudsar, Iran (e-mail: sajed.dadashi@ yahoo.com, sahar_zdp@yahoo.com).

several processing units are used to improve the system efficiency. The designed system is shown in Fig. 1.

The presented system can receive input images(for example, from a camera) consecutively. There are four compression units that act in the same way. The control unit informs four units consecutively to receive their related block with 64 bytes of data.

B. Operations

Compression unit 1 receives the 64 bytes of data related to block 1 and saves them in it's internal SRAM memory. Simultaneously, the sum of block values is calculated. Then while the compression unit 1 operates next calculations(for example, division of block sum by 64 to produce the block average), the compression unit 2 receives the 64 bytes of block 2. Then while unit 1 operates next required operations and unit 2 finishes the receiving and starts the operations, unit 3 receives the 64 bytes of data related to block 3. Finally, while unit 1 starts to send compressed data to MMC memory, unit 2 operates last operations, unit 3 operates next operations and unit 4 receives the 64 bytes related to block 4. This cycle repeated again and again to receiving data with four units consecutively.

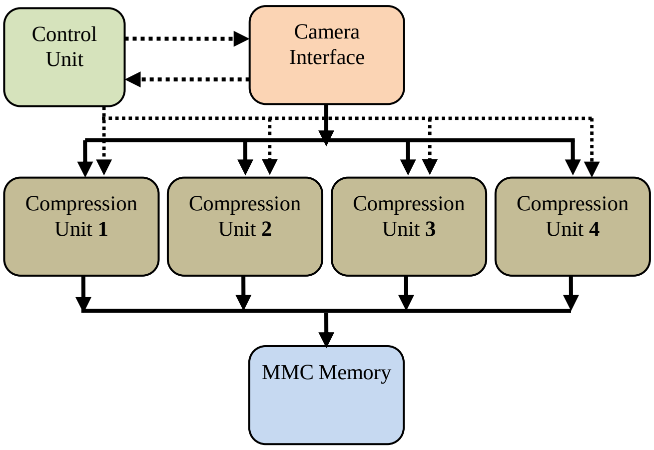

Fig. 1. The system designed for BTC image compression algorithm.

When one unit receives all data related to one block, the block sum is calculated. The sum of block must be divided by 64 to get the block average. For this operation, the sum of block is shifted to the right by 6 bits that executed very fast. Then the unit reads 64 bytes stored in internal SRAM consecutively and compare them with block average. if the pixel value is smaller than the block average, adds it to the block low sum and adds 1 to the low counter. Otherwise, adds it to the block high sum and adds 1 to the high counter. In the end of this stage, we have the sum of pixel values smaller than block average in block low sum and the sum of pixel values equal or greater than block average in block high sum. we also have the number of smaller values in low counter and the number of greater values in high counter. Low counter and high counter can have arbitrary values between 0 and 64 based on the related block. Therefore, we cannot use right shift to divide block low sum by low counter to calculate block low average and to divide block high sum by high counter to calculate block high average.

The number of compression units is configured in a way that no interruption take place in the operations of units and also no confusion exist in the receiving data from input and sending compressed data to MMC memory.

The control unit sends a signal to units 1 to 4 and inform them for receiving data from input. While one unit is receiving data, input lines of other units are in high ampedance state. Also when one unit is sending compressed data to MMC memory, output lines of other units are in high ampedance state.

IV. Proposed New Division Algorithm

We use AVR microcontrollers in the implemented system, because they have very low cost, but they cannot execute the division operation. We can use the continuous subtraction to calculate block low average and block high average, but this calculation is very time consuming. therefore we must find another way.

We propose the new division algorithm that named it hierarchical subtract based division(HSD). This new algorithm can decrement the division time and have important role to increment the number of images that can be received in one second. In HSD, initially some coefficients are selected based on the divisor that here is low counter and high counter. Because the divisor is between 0 and 64, we select 3,8,25,55 as four coefficients. We can select arbitrary coefficients but, it should be noted that the difference between them must not be small. Then, the multiplication of divisor and every coefficient are calculated. Maybe this work waste time a little, but speedups next stages. In AVR microcontrollers, the multiplication is executed in one cycle. Then, the dividend is compared with the greatest product. If the dividend is equal or greater than the greatest product, we add the greatest coefficient to the quotient and subtract the greatest product from dividend. This operation continues with comparing the decremented dividend and the greatest product again and again and when the dividend becomes smaller than the greatest product, in the next stage, the dividend is compared with the next greater product and so on Fig. 2 shows the stages of this algorithm.

If we use continuous subtraction, after every comparison and when the dividend is greater than the divisor, the quotient is incremented one unit and the number of comparisons and increments is almost equal to quotient, but if we use HSD, several comparisons and subtractions are done in one stage. For example, if the result of division(quotient) is 59, in the first stage, 55 is added to quotient and in the second stage, 3 is added to the quotient. Therefore, the processing that is done in two stages is equal to 58 stages in continuous subtraction. In the final stage, dividend compare with the divisor and 1 added to the quotient and the result is 59.

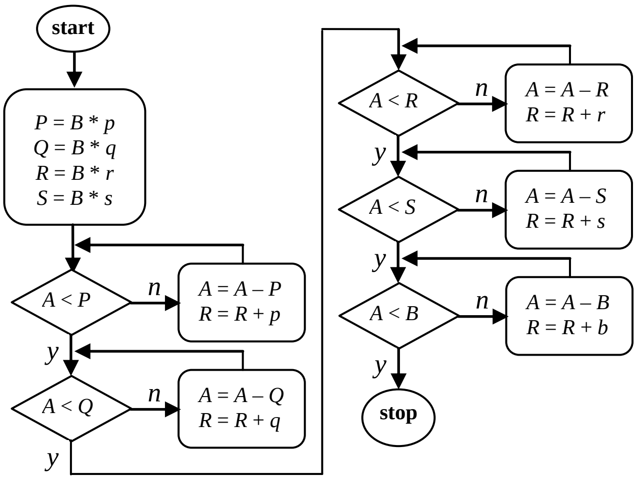

Fig. 2. HSD algorithm four coefficients: *p*, *q*, *r*, *s* dividend: *A* divisor: *B* quotient: *R* four products: *P*, *Q*, *R*, *S*.

V. System Implementation

A. Microcontrollers and Connections

In this sysem, we use one ATmega 8 for control unit and four Atmega 32 for compression units. Pin configuration of these microcontrollers are shown in Fig. 3.

When the data ready line that is connected to portc pin 0 of control uint, becomes high, control unit informs one of the compression units to receive 8 bit data related to one pixel of image block. When control unit informs one unit for 64 times to receive 64 bytes, in the next stage, informs another unit to receive 64 bytes and so on. At first, compression unit 1 is busy. Then, compression units 1 and 2 are busy for different stages of compression algorithm. This process continues until compression unit 4 starts to receive data. In this time and until the system is on, all units are busy and therefore, only at small time in the system start, maximum use of units is not possible.

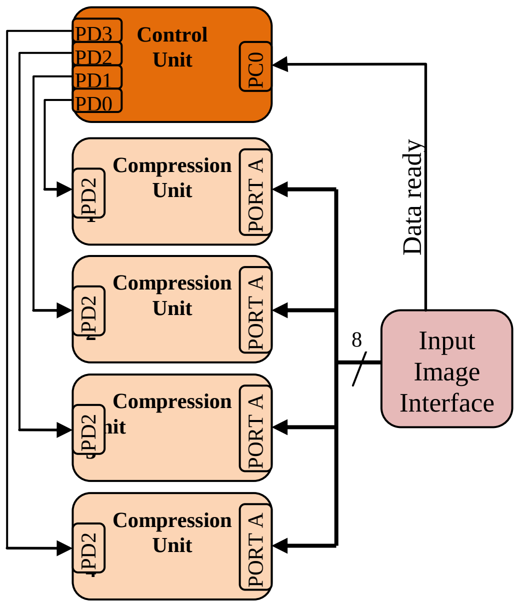

Fig. 3. Pin configuration of control unit and compression units.

B. MMC Memory

We decided to save compressed images in MMC memory that requires 3.3 volts to work, while AVR microcontrollers require 5 volts. The data must be sent from output port of microcontrollers to MMC, has 5 volts and must be decrement to 3.3 volts to connect to MMC input lines. This problem is solved by resistant network that is shown in Fig. 4.

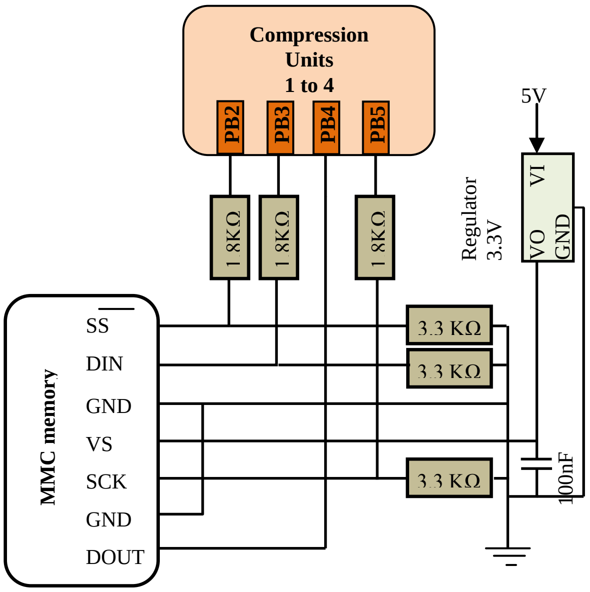

Fig. 4. Connection between microcontrollers and MMC.

AVR microcontrollers have the SPI interface and can connects to the MMC memory. Microcontrolle is master and the MMC memory is slave. Four lines are required for SPI connection as shown in Fig. 4.

The implemented system is shown in Fig. 5. It should be noted that the large size of circuit is because of using the microcontrollers in the PDIP package and placing the system in the 1000 holes board. If we use the microcontrollers in the TQFP package and place them on the printed circuit board, the circuit size will be very small.

VI. Speed Analysis

In this system, the maximum frequency(16MHZ) is used for microcontrollers via external oscillator. All instructions are written in assembly language using AVR Studio Synthesis tool. The number of required cycles for any instruction is obvious. Our calculation shows that receiving 64 bytes of data related to one block takes 1040 cycles. In each second, 16 million cycles exist and therefore 15384 blocks can be received in one second. We test the system with 256×256 images that have 1024 blocks. If we divide the number of received blocks in a second by 1024, the speed of 15 images per second can be achived that is practically observable.

VII. Conclusion

As shown in this paper, we can use several parallel microcontrollers and divide jobs between them to create different systems with acceptable speed and very low cost that can be used instead of expensive FPGAs in applications that the time of processing is not very critical.

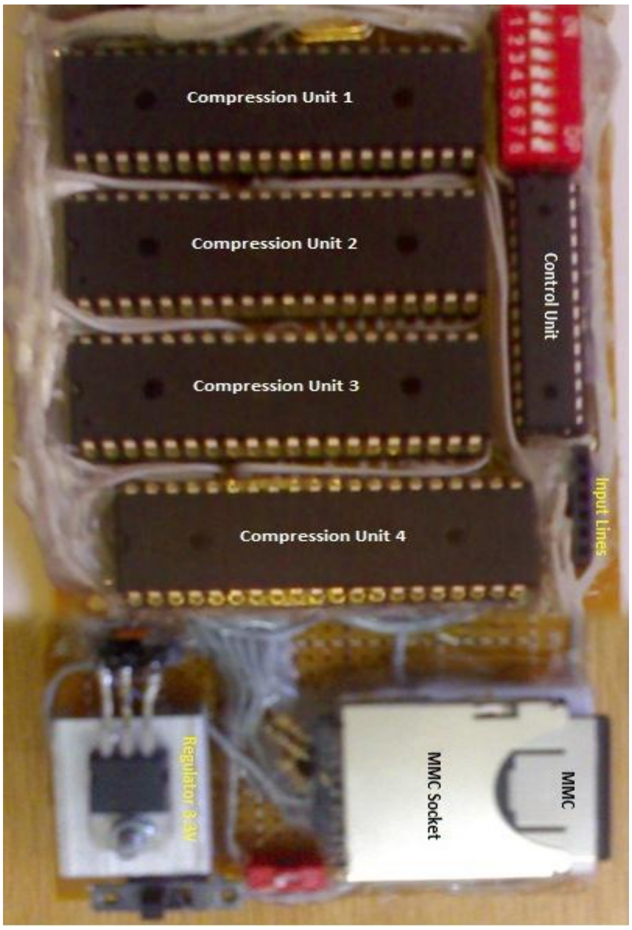

Fig. 5. The implemented system.

ACKNOWLEDGMENT

This paper is a result of the research plan "Design a hardware system for the JPEG2000 image compression algorithm". In fact, while working on this research plan, we decided to use some parallel AVR microcontrollers to have suitable speed and very low cost. At first, we used this approach for the simpler image compression algorithm "BTC" and motion detection to test our way.

REFERENCES

[1] B. O. Hanen and M. Wisan, *JPEG Compression*, 2005.

[2] *JPEG 2000: Image Compression Fundamentals, Standards and Practice (The International Series in Engineering and Computer Science)*, Springer.

[3] F. Vahid, "Digital design with RTL design," *VHDL and Verilog*, Second Edition, 2007, J ohn Wiley and Sons

[4] D. Abramson, P. Logothetis, A. Pstula, and M. Randall, "FPGA based custom computing machines for irregular problems," in *Proc. Fourth Int. Symp. on High-Performance Computer Architecture*, Las Vegas, Nevada, 1998, pp. 324-333

[5] S. M. Saif, S. Nassar, and H. M. Abbas, "FPGA implementation of block truncation coding algorithm for gray scale and color images," in *Proc. IEEE Canadian Conf. Electrical and Computer Engineering*, *(CCECE 2003)*, vol. 1, pp. 23–26, 2003.

[6] S. M. Saif, S. Nassar, and H. M. Abbas, "An FPGA implementaion of a neural optimization of block truncation coding for image/video compression," *Elsevier Microprocessors and Microsystems*, 2006

[7] E. Delp and O. Mitchell, "Image compression using block truncation coding," *IEEE Transactions on Communications* vol. 27, no. 9, pp. 329-336, 1979.

Sajed Zadeh Dadashi received his bachelor degree in Computer Engineering(Hardware) from the Department of Computer Engineering, Shahid Bahonar University of Kerman in 2005 and M.Sc. Degree in Computer Engineering(Computer Architecture) from the Department of Computer Engineering, Arak Branch, Islamic Azad University in 2009. His research interests include computer architecture, digital design, image processing, fault tolerant systems.