# Methods for increasing the performance of image processing system

Tsvetomir Nikolov Lazarov and Todor Stoianov Djamiykov

*Abstract* – **This paper presents methods for increasing the performance of image processing system. The factors that influence the process are presented and their relations with other main system parameters. Several example algorithms are presented as a illustration of these ideas.**

*Key words* – **image processing, algorithms, performance, parallel processing, embedded systems.**

## I. Introduction

Modern days design and manufacturing of semiconductor integrated circuits is at level where the implementation of complex multi-functional electronic systems is getting easier and reachable goal. Application area and the type of the problems solved are never stop expanding. This trend is valid for image processing systems too. A modern system of this type has the following main characteristics:

- It is constructed from several heterogeneous components (optic, image sensor, processing system, external communication interfaces) and all of them are mandatory for system to be operational;
- There are lots of processing components that transform the type and the quantity of the information;
- Different types of functions are executed (information processing, transfer and storage) and they have a different significance to the main system goal;
- Implemented image processing algorithms work in real time and they are complex enough in order to guarantee the required quality of the output result;
- There is possibility for firmware preprogramming or update, parameters setting, selection of functions to be utilized, mode selection;
- Relatively modest amount of internal memory and access to unlimited external memory through high-speed communication interfaces;
- Short live cycle (design, production and operation time) due to the fast new product development and the competition.

T. Lazarov is PhD student at the Department of Electronics and Electronics Technologies, Faculty of Electronic Engineering and Technologies, Technical University - Sofia, 8 Kliment Ohridski blvd.,1000 Sofia, Bulgaria, e-mail: tsvetomirlazarov@yahoo.com

T. Djamiykov is with the Department of Electronics and Electronics Technologies, Faculty of Electronic Engineering and Technologies, Technical University - Sofia, 8 Kliment Ohridski blvd., 1000 Sofia, Bulgaria, e-mail: tsd@tu-sofia.bg

Image processing is mostly sequence of operations and functions applied over every single picture element (pixel). The image is partitioned into the CMOS image sensor to a huge number of elements – starting from half a million to several millions and going more in the newest sensors. Due to the increasing number of the pixels, the time required for processing of the whole image also increases. This issue is getting bigger in real-time mode of operation with continuous stream of frames and even more difficult if the processing algorithms are using data from several consequent frames for certain calculation.

The total duration of one frame processing is critical parameter for real-time systems. If the system does not comply with this criterion it should be redesigned with applying methods for increasing system performance. When one of them or a combination of methods is utilized, the total duration for frame processing could be reduced.

Image processing system performance may be increased not only by speeding up the processes which is accomplished through decreasing the total processing time and/or the time for execution of the basic functions. Another approach to increase system performance is by executing more tasks at the same time. This feature is specific for the systems that implement parallel processing. They could achieve many times improvement through the use of more hardware resources. Pipe-line structure of processing is the typical example of parallel processing.

In accordance with the data presented by now, the methods for increasing the system performance could be grouped in two main categories – methods that has a main goal duration reduction and the second group is of methods that implement parallel processing.

## II. Reducing the processing time

The easiest way for reducing the processing time is by increasing the frequency of the main clock signal. In most cases, the relation is linear – if the clock frequency is increased N times than the processing time is decreased N times. The designer should consider the fact that all the constants that are timing related are initially calculated for the lower frequency, so they should be calculated for the newer one. This method has upper limit determined by the parameters of the hardware. Besides, increasing the clock frequency leads to excess power consumption that is reason for change in two other system parameters. The first one is the duration of one battery cycle – the time that the device works without additional battery charging. Increased consumption exhausts the battery faster – battery cycle is getting shorter. Second main issue is caused by the released heat that warms up the chip and leads to negative change of the electrical and the timing parameters. Additional efforts as cooling, thermo-control and thermo-compensation should be applied in order to neutralize this issue and to guarantee correct and safe chip functioning. There are designs in which the trade-off between consumption and speed could be reversed. If the system time requirements are met in case of work at lower clock frequency than it is decreased as much as possible in order to improve the consumption related parameters.

Another approach for processing time reduction is by decreasing the number of clock cycles needed for a certain operation. There are several ways to do this. “Intelligent design” is a term that describes the manner that the system designer should utilize during the system designing. Processing algorithms and functions should be the most appropriate for the specified task, excess and unnecessary repetitions of operation executions should be avoided, definition and usage of constants and variables should account available data types and data type conversion, etc.

It is a common case when there are several similar or alternative processing algorithms. Each one of them has its own advantages and disadvantages, requirements and quality of the end result. Figure 1 shows diagram that presents the correlation between the complexity of the implemented algorithm and the number of required processing operations needed for one frame. It is obvious that the more complex algorithms (the ones that work with more pixels) demand more execution time. The shape of the curve is exponential and confirms the fact that better quality comes at the expense of slower execution.

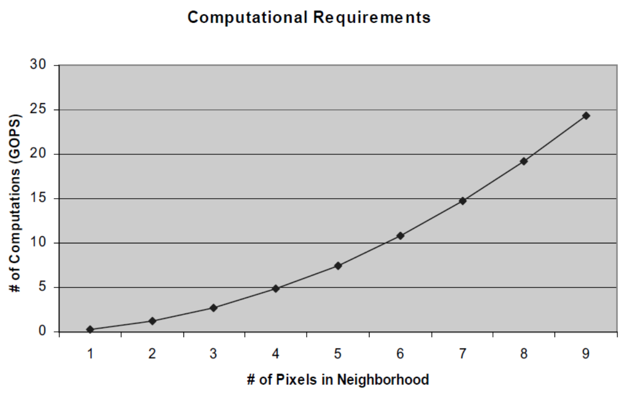

Figure 1. Total number of operations for one frame processing in dependence of implemented algorithm [4].

The designer can select adequate algorithm depending on the parameters of the project and required end result. The goal is to get the result with optimal usage of the available resources in keeping the restrictions. Different algorithms get to the final result with a different quality and could generate or hide some additional effects/defects. Designer’s choice depends on his practical experience, the results from research end experiments and on how the rules and the restrictions for the current system design are kept.

Another possible way for decreasing the number of clock cycles needed for a certain operation is by changing the balance software and hardware task implementation. There are specialized hardware modules (accelerators) that could execute some operations for a few clock cycles. They use dedicated modules and logic that is specifically designed for certain task. There is possibility for some configuration and parameters setting in some modules

Design process of the system hardware structure and the choice of its main components is in great importance for the system performance. The processor is the most important element – its parameters and instruction set, its architecture and memory organization, the availability of tool for speeding-up the processing (pipe-line instruction execution, cash memory and others). The memory block should contain enough memory space with small access and data transfer times. Memory organization and usage depend on the current application but this is also a factor that could be system bottle neck. High-speed serial or N-bit parallel communication channels are required for information transfer.

There are three basic operations that are executed during the image processing – multiplication, adding and comparison. These three basic operations are the elementary functions for construction of all the processing algorithms. The way they are implemented and used determines in a high degree how the available resources are utilized and defines the system performance. The multiplication uses a pixel value as multiplicand and specific coefficient is used as a multiplier. The coefficient value depends on the applied algorithm and the current step that is executed. It could be integer or fractional number and it could be constant for every pixel or variable according to some function. The adding is used for a number of calculations – for finding the average value, for calculation of certain variable that is constructed from several components, for adding and/or subtracting of some value (when the deviation from the average value is calculated for example), for eliminating the noise component and etc. The comparison checks the value of the pixel or variable and answers to the question if this value is above/below certain threshold, if it into or outside a defined interval, determines the position of the pixel in the sensor matrix, compares pixel value against other pixels’ values, compares same type variables and so on.

## III. PARALLEL PROCESSING

Parallel processing is defined as simultaneous processing of two or more pieces of information by the same number of functional processing modules. There are two basic types of parallel processing depending on the types of the modules and the functions that they execute.

The first type – task-parallelism uses different functional modules for the different tasks. Processed information flows sequentially through all of them. Pipe-line instruction execution is the typical example while instruction N is executed, the next one (N+1) is decoded and the one after it (N+2) is fetched from the program memory and in the same time the result from the previous one (N-1) is stored into the data memory.

Data-parallelism utilizes multiple same type modules for executing simultaneously the same operations over different parts of the data array. Figure 2 presents a example of data-parallelism. It is very important the correct distribution of the pixels into the separate streams and the correct merging and arrangement of the pixels in the common array at the end. Every processing stream could contain multiple stages but they should be the same for all streams.

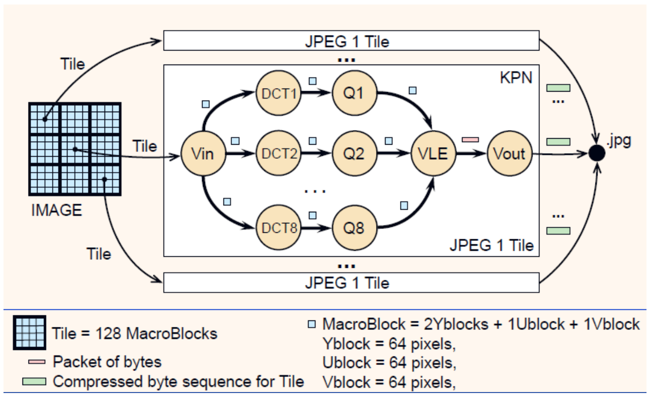

Figure 2. Data-parallelism example [2].

Multiprocessor image processing system is the one that has embedded multiple processing elements (processors that execute complex and/or many functions) into the system structure. In this case we have multiprocessor information processing that could include task-parallelism and/or data-parallelism. Some main issues should be solved in order to guarantee the correct execution of the multiple functions. One of them is the task distribution among the processors. Other issue is the separation of operation with many repetitions over the set of same type modules. Another task is defining the number and the size of the memory blocks used by the different applications. May be, the most important task is the synchronizing and management of the data streams and the control of the task execution. All of these issues are solved by using specialized software for system level design – for example DAEDALUS design flow. It generates adequate output by applying proper models and algorithms for analysis and synthesis.

## IV. EXAMPLES

Reducing three times the duration of processing one frame is possible just by rearrangement of the function execution sequence. Example for this method is described in [5] – repeated multiplication (dominating factor that determines execution duration) is moved to be the very first operation and this simple change is enough to get the significant improvement.

One of the edge detection algorithms works with Sobel operator that is defined as follows:

$$Ax = \begin{vmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{vmatrix} \quad Ay = \begin{vmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix}$$

The real implementation of this algorithm omits multiplication with coefficient „0” – this component is not added to the sum. Multiplication with coefficient „±1” is not made too – the corresponding components are just added to or subtracted from the sum. Multiplication with coefficient „±2” also could be replaced with two times addition or subtraction.

To adopt the method described in the first example for speeding-up the edge detection algorithm the following should be done. If there is enough memory space, multiplication by 2 of every pixel should be executed once in order to generate “pixels multiplied by 2” array. When the main procedure requires sum component that is pixel multiplied by “±2”, the corresponding element from the prepared array is fetched and added to the sum. This modification of the edge detection algorithm could be used for other similar algorithms too. For example, edge detection algorithm that utilizes the Kirsch operator works with coefficients „0”, „±3” и „±5”.

If certain conditions are met, some operators could be replaced with two simpler operators without adding significant error. This method – “singular-value decomposition combined with small generation kernel decomposition”, is describe in [3]. It allows work with fewer and simpler multiplication coefficients. Hence the execution is speeded-up, functional processing modules are simpler and memory buffers are smaller. Figure 3 presents substitution of operator H that is 5x5 by two simpler operators F and G that are 3x3.

$$H = \begin{vmatrix} 1 & 3 & 4 & 3 & 1 \\ 3 & 9 & 12 & 9 & 3 \\ 4 & 12 & 16 & 12 & 4 \\ 3 & 9 & 12 & 9 & 3 \\ 1 & 3 & 4 & 3 & 1 \end{vmatrix} = F * G$$

$$F = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix} \qquad G = \begin{vmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{vmatrix}$$

Figure 3. Kernel decomposition

Some of the embedded functions of the CMOS image sensors allow processing of subset of frame elements. “Skip” function makes sub-sampling – certain number of rows and/or columns are omitted during the sensing matrix scanning. “Bin” function automatically sums 2x or 3x rows and/or columns neighboring pixels and outputs the sum as a element of frame with lower resolution. “Window Size” function allows defining the size and the position of the area of interest. The user could implement algorithm with dynamic control of the sensor output stream by combining the functions mentioned above. At step one the whole sensor matrix is scanned at low resolution. Resulting frame is processed in order to get the object’s position and dimensions. These data is used for defining the sensor area that will be scanned at maximum resolution. The pixels from that area contain the information needed for final result calculation with enough accuracy.

## V. Conclusion

The methods presented in the current paper are only the theoretical basis. Quantitative evaluations can be made after system design and implementation. There are some examples of image processing systems analyzed in [6].

In order to get good results in terms of both speed and quality, the right balance between required end result and utilization of the available resources should be reached.

Embedded functions of the CMOS sensors if used in the proper way could lead to simplified and improved image processing system.

Presented methods for performance increasing and the requirements to the hardware and software lead us to electronic elements that are quite suitable for image processing system implementation. These are the FPGA chips – they have enormous amount of hardware resources that allow implementation of multiple elements for parallel data processing, prebuilt specialized processing and communication modules, a lot of internal memory with flexible organization, high clock frequency and many other features that make them preferable choice as a core of the system processing part. This is especially valid at design stage and/or if few devices are to be produced.

## Acknowledgement

This work is supported by the Technical University – Sofia under Contract 122ПД0049-03.

## References

[1] Nikolov H., *System-level Design Methodology for Streaming Multi-Processor Embedded systems*. Leiden Institute of Advanced Computer Science, Leiden University, The Netherlands, 2009.

[2] Stefanov T., *Embedded Systems and Software.* Leiden Institute of Advanced Computer Science (LIACS), Leiden University, The Netherlands, 2010.

[3] Castleman K.R., *Digital Image Processing*, Prentis Hall, 1996.

[4] Nakamura, J. *Image Sensors and Signal Processing for Digital Still Cameras*, CRC Press, 2006.

[5] Ц.Лазаров, Т.Джамийков, *Модифициран алгоритъм за трансформиране на RGB Bayer матрица в монохромен кадър*, Сборник на МИИО, София, 2011.

[6] Ц.Лазаров, Т.Джамийков, *Анализ на ефективността на многопроцесорна вградена система за обработка на образи* , Сборник „Електроника 2012”, София, 2012.