Accelerating embedded software processing in an FPGA with PowerPC and Microblaze

Luis Pantaleone and Elias Todorovich
INTIA Institute
Universidad Nacional del Centro de la Pcia. de Bs. As.
Paraje Arrollo Seco, Tandil (B7001BBO), pcia. de Bs. As, Argentina
Email: {lpanta,etodorov}@exa.unicen.edu.ar

Abstract—**This paper presents an heterogeneous multi processor architecture for fast algorithms for image processing. Each microprocessor process an asymmetric fraction of the image. The proposed architecture uses one hardcore PowerPC microprocessor as master and multiple softcore MicroBlaze microprocessors as slaves. Several architecture configurations were utilized to get the maximum acceleration respect to a single microprocessor. Decoding image algorithms for Bayer pattern are present as study case. The systems were implemented in a Virtex-5 FXT30, with one PowerPC and multiples MicroBlaze. A 3-core system with one PowerPC and two MicroBlaze achieves 2X acceleration.**

I. Introduction

Moderns FPGA digital systems can include one or more microprocessors. The image processing could be done in the FPGA or in a microprocessor. The Xilinx FPGAs, has the MicroBlaze[1] softcore microprocessor, and in some families the hardcore PowerPC[2]. The newest family Zinq-7000 uses an ARM Cortex A9 dual core microprocessor[3]. This new family uses the architecture AMBA developed by ARM instead the CoreConnect. One advantage of processing the images using algorithms implemented in a microprocessor is the ease of implementing them against the implementation in FPGA. The two alternatives give us a tradeoff between ease of development and performance.

The software is sequentially execute. This has a main disadvantage, the performance. If the objective is raise the performance, the system can be implemented in a FPGA. Thus takes advantage of the parallelism. But, the development time is greater than the software.

One of the basic options when it comes to speeding up a single core system is to increase the frequency. But this increase has a limit. The MicroBlaze processor uses the frequency of the digital system to operate. In contrast, the PowerPC uses a core frequency independent of the digital system. It also uses another clock frequency for the processor PLB interfaces and the processor interconnect (crossbar). The clock frequency ratio between processor clock and interconnect clock can be N:1 or (2N+1):2, where N is an integer greater than 0[2]. Because of this, sometimes by increasing the core clock frequency lowers the crossbar frequency, resulting in a digital system running at lower speed.

When the frequency reaches his maximum limit, another solution is needed. One solution can consist in parallel processing. Today high-performance embedded systems consist of digital subsystems and multiple microprocessor. The software program is distributed over multiple microprocessors using sophisticated inter-connects[4].

The proposed architecture is based on multiple processors, it is called multi-core system. When a multi-core systems contains one PowerPC and one or multiples MicroBlaze, we refer as an hybrid (heterogeneous) multi-core system. Many papers[5][6][7] has been written about homogeneous multiprocessor architectures, but most of them uses only softcore processors. Another paper has been written about heterogeneous microprocessors and their interconexion [8]. The focus is not placed on the parallelization, each microprocessor performs a certain function. The focus in this work is the software parallelization using heterogeneous microprocessors. The main focus in this work is in the load balancing of the microprocessor. Each microprocessor process an asymmetric portion of the image instead symmetrical portion.

A image algorithm for Bayer image decoding is presented a case study. The target architecture has been designed with the Xilinx Platform Studio (XPS) version 14.3. The bus architecture used is the CoreConnect by IBM.

The rest of this paper is organized as follows. Section 2 describes the multi-core systems and their alternatives. In sections 3, we present results of our implementations and comparison of their performance. Section 4 concludes the paper with several major outcomes.

II. System Design

A system based on multiple processors has been designed. It is based on the master/slave pattern[9]. The system has one microprocessor as the master which synchronizes the slaves.

The system is restricted to parallelizable algorithms, that that is when there are not dependecies among the original image pixels and the processed image pixels to be processed. In general, to calculate the new value of a pixel, the neighbors data is needed. The submatrix formed by the central pixel and the neighbors is called kernel. The most frequently kernels used are 3x3 or 5x5. Exist another configurations used by some algorithms[10].

When a new image arrives, the master calls the slave microproccessors to process it. Every one executes the same algorithm, but on different images areas. Both, the master and the slaves, executes the algorithm in parallel. Every slave works on a specific memory area, reading and writing in their corresponding addresses. Required image portions are copied from global memory area to a local memory area by the slave microproccessor before the image processing. The master in addition to the image processing is responsible for other tasks, including image transmission from, to, and inside the system. The software in each processor run in stand alone mode. To run it usually an OS is not required.

Every microprocessor (master and slaves) has his local memory for data and instructions. The microprocessors uses the Harvard architecture instead the Von Neumann. The memories are implemented in block RAMs. All of these shares the main memory, an external DDR RAM. It uses the MPMC (Multi Port Memory Controller) core. The MPMC is responsible for communication with the DDR external memory. It has 8 independent ports. These ports can be connected to a MicroBlaze microprocessor, PowerPC microprocessor, Video Frame Buffer, PLB bus, etc.

The images arrive from a *camera* controller through a parallel interface. This core acquires images from the external camera and sends to the main memory, across the PLB bus. It also controls the camera. The core has to synchronize the value of pixel intensity with the control signals, also has to synchronize with the camera clock and system clock. The frequency at which work the camera is 24MHz. The camera sensor doesn't work with RGB pattern, instead, works with pattern Bayer. Due to this, only need one channel instead of the 3 channels required by RGB. The depth of color is 12 bits, but the system works with the 8-bit MSB. It is described in detail in[**?**].

The software executed in the master microproccessor has an extra logic for the communication between it and each slave. As the processing times of the master and slaves are different, a synchronization mechanism is needed. The communication between the master and each slave is trough the IP Core MailBox. It is explained in detail by Xilinx[11].

The master can be implemented in a PowerPC in the case that the FPGA have one, otherwise in a MicroBlaze. The slaves are implemented with the MicroBlaze processor. The figure 1 shows an slave. The slave is formed by the MicroBlaze microprocessor and local data and instruction memory. The Mailbox IP core for communication between the MicroBlaze and the master and others IP cores and buses is used.

A multi-core system could contain one or more slaves. The Fig. 2 shows a QuadCore (4-core) system. This configuration permits adding any number of slave units. The memory controller uses the IP core MPMC. The PLB bus the camera controller and the master microproccessor use one port each one, so the are 6 ports available. Therefore up to six slaves can be connected, if there are enough resources in the FPGA. In the case that more than six slaves need to be conected the ports access will be shared trough a PLB Bus resulting in a slower memory access.

A. Asymmetrical processing

Each microprocessor (master or slave) has an area (or portion) of the image to process (load balancing). This area in general is not equal for all. The relative area of the image of the master microprocessor is calculated in the Eq. 1, and the relative area of the image for each slave microprocessor is in the Eq. 2.

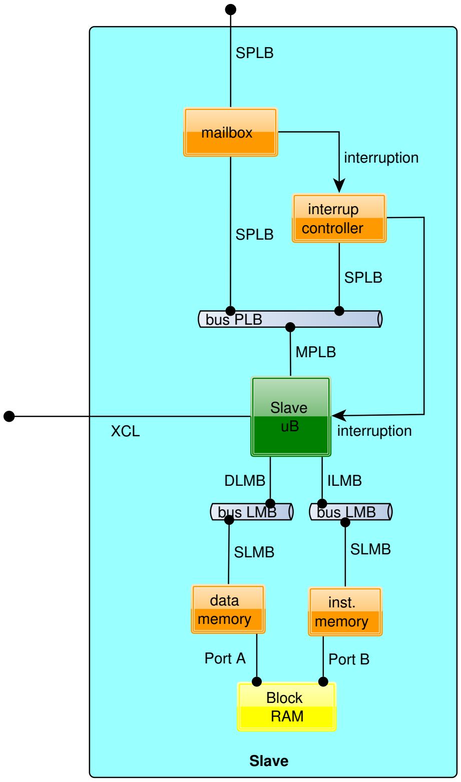

Fig. 1. Slave

$$F_m = \frac{\frac{T_s}{N-1}}{T_m + \frac{T_s}{N-1}} \qquad (1)$$

$$F_s = \frac{1 - F_m}{N - 1} \qquad (2)$$

Where F_m and F_s are the relative area of the image for master and slave respectively, and N is the number of cores (microprocessors). The variables T_m and T_s are the time that takes to process the 100% of the image by the master and slave respectively.

The relative area of the image indicates the fraction of the total image, and this number goes from 0 to 1. The absolute area (or "area") of the image represents the number of lines of the image to process. This is calculated by multiplying the total number of lines of the image and the relative area. The result is a real number, must be rounded to the nearest integer. Due to this, the area of each slave is not calculated from the relative area. The number of lines to process is calculated from 3. The problem is that not always the result is an integer. Due to this, the area of image of each slave must be rounded to the lower integer modulo, and the area of the image of the master are the remaining lines.

$$L_s = \frac{1 - L_m}{N - 1} \qquad (3)$$

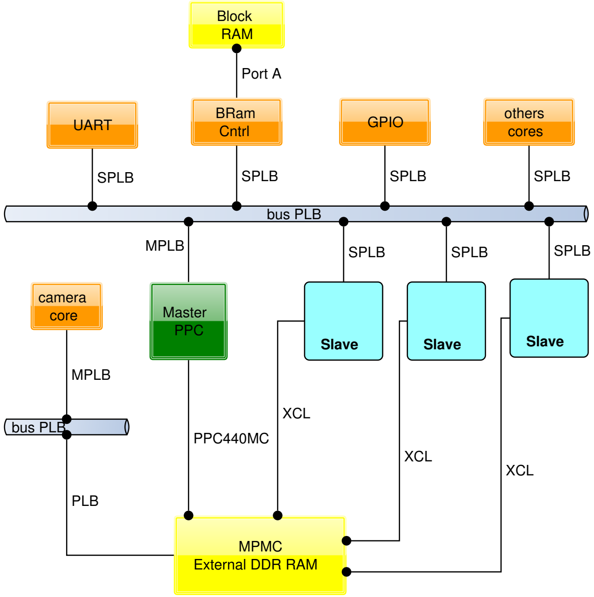

Fig. 2. MultiCore System

One problem to find the perfect fractions or load balancing to minimize the overall time is that the result is a float. These rounded results in an imbalance of areas. Another problem comes from the number of slaves. The larger the amount the slaves, less processing lines on each core. It means that one line less on each slave causes in an order of slaves number more lines in the master.

The total lines of the image to process it must include some extras image lines to perform the correct image processing. This is because the image algorithms uses a kernel. The image is stored in the memory in a row-major order.

III. Experimental Results

The experimental results were implemented on a Virtex 5 FXT Evaluation Kit. The Virtex 5 FXT system board includes the Xilinx Virtex-5 FX30T FPGA, with over 32,000 logic cells, one embedded PowerPC 440 core and 2,448 Kb of Block RAM. The system board includes 64 MB of DDR2 SDRAM. The frequency used by the system (included MicroBlaze) was 125MHz and 250 MHz by the PowerPC.

The implemented algorithms for the experiments are Bayer image decoding [12]. The Bayer image algorithms are *"lineal interpolation"*[13] (called algorithm 1), *"lineal interpolation with correlation"*[13] (called algorithm 2), and the *"Gradient Based Interpolation"* (called algorithm 3). The last one is proposed by Laroche and Prescot[14]. The last one was used in the Kodak DCS 200 digital camera system [15]. The difference between these algorithms is the resulting image quality. To achieve a better image quality more arithmetic operations between pixels and more memory access are required. Impacting on the processing time.

For run time measureents, systems with one to six Microblaze slaves and a PowerPC master were implemented. Due to the Block RAMs limitation only systems with one (dual core) and two slaves (3-core), plus the master were implemented. Table I shows the corresponding synthesis results. The number of LUTs are rounded to thousands of them.

TABLE I. CONFIGURATIONS SYNTHESIS RESULTS

	dual core	three core
Block Rams (in kb)	1,296 (52%)	2,160 (88%)
LUTs (in thousands)	8 (38%)	13 (63%)

TABLE II. ACCELERATION BETWEEN ASYMECTRIC AND SIMETRIC FRACTIONS

Master	Alg. 1	Alg. 2	Alg. 3
PowerPC	211ms	248ms	322ms
MicroBlaze	372ms	361ms	520ms

The time to process the complete image, obtained with only one PowerPC (as master) and only one MicroBlaze (as master) were taken for each algorithm. Table II shows the time to process. In all the three algorithms the *PowerPC* is faster than *MicroBlaze*, achieving an acceleration between 1.46X and 1.76X. Thus, the *PowerPC* can process a bigger area of the image than the *MicroBlaze*. According to Xilinx [16][17], the PowerPC 440 gets 2*DMIPS/MHz* in the Dhrystone benchmark, while the MicroBlaze gets 1.19*DMIPS/MHz* (synthesized with the performance configuration). It results the PowerPC is 1.68X times faster than MicroBlaze.

In a context of heterogeneous multi-core microprocessors is interesting to study the case with processing times when the image is divided into a symmetrical or asymmetrical fractions of the image. The time to run the algorithms with symmetric and asymmetric fractions of image on each core were taken for timing analysis. The times for systems with more than three slaves were taken with the dual core system. But in this case, the master and the slave, process the fraction of image corresponding to the number of microprocessors cores to simulate. The run time for systems from 7 to 10 cores were taken only as analysis measure. The systems were simulated on a 3-core system, ignoring the ports limitation of the mpmc. The times for image processing with the approach of asymmetric and symmetric fractions of the image shown in Fig. 3, 4 and 5. The asymmetrical approach is faster than the symmetrical, but this advantage decreases as the number of cores increases. In the case of the algorithm 2 and 3, the execution times between the approaches of asymmetric and symmetric fractions of image for systems 2 and 3 over 8 cores were the same.

The accelerations between a system with asymmetric and symmetric fractions of image are show in the Fig. 6 . These acceleration decreases as the numbers of microprocessor cores increase. The three algorithms show a similar acceleration pattern. The acceleration of the algorithm 3 with 8 or more cores is almost 1. These means that the fractions of the image tends to be almost symmetric. The other algorithms tend to converge to a 1x acceleration in a similar way. The acceleration of the algorithm 2 in a 9-core system is less than 1 (0.97X). Due to the inbalance of the image fractions, the time processing of the master microprocessor is bigger than the slaves microprocessor. In this case the processing time of the asymmetric fraction is more than the symmetric fraction.

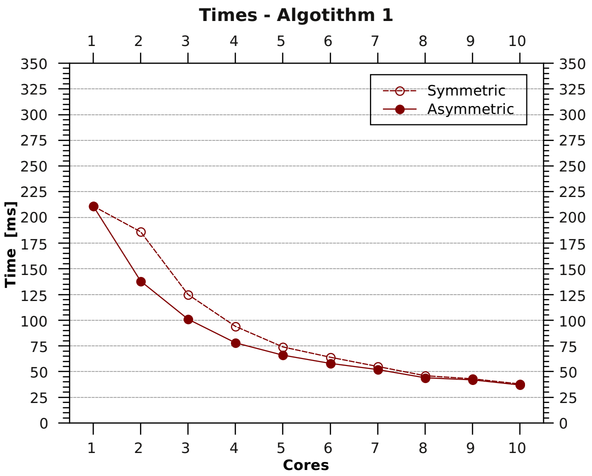

Fig. 3. Times for Algorithm 1

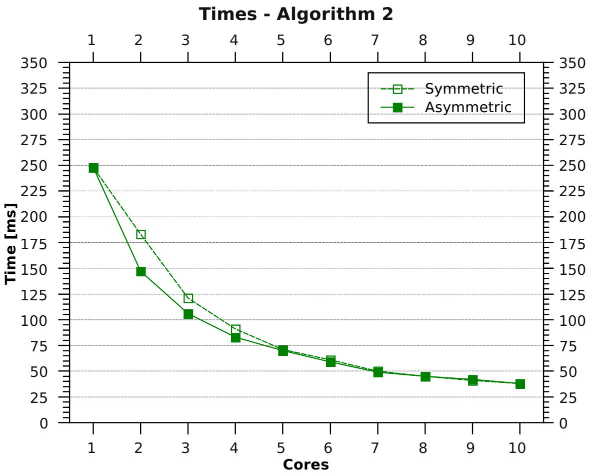

Fig. 4. Times for Algorithm 2

This behavior can happen when the system has 4 or more cores. The size of the image affects the convergence to 1 of the acceleration of an algorithm. When the image are smaller, asymmetric and symmetric fractions tend to be similar with fewer cores than if they were larger images.

The acceleration with asymmetric fractions of image respect with the one PowerPC core system is show in the Fig. 7. It means, the more *slave* processors the better will be the acceleration, but there is a limit. This limit is determined by the resources of the FPGA and the available ports of the MPMC. Such limit is the physical type. The acceleration is below the ideal acceleration, and the distance from the ideal acceleration is increased with the increased of cores. The ideal acceleration is the numbers of cores of the system, e.g. in a 4-core system the ideal acceleration is 4X. Due to limitation of memory ports, the maximum acceleration which can be implemented is a 7-core system, its acceleration is between 4X and 5X. The 7-core system has one PowerPC master and 6 MicroBlaze slaves. However due to fisical limitation a 3-core system was implemented, achieving an acceleration between 2.1X and 2.3X. The difference between the acceleration of the three algoriths is given by the memory accesses and Arithmetic operations.

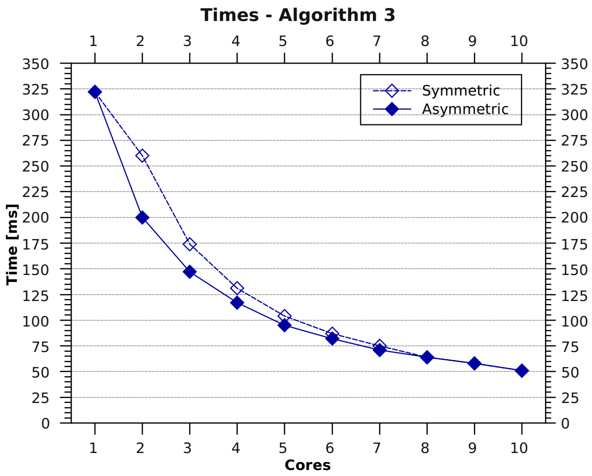

Fig. 5. Times for Algorithm 3

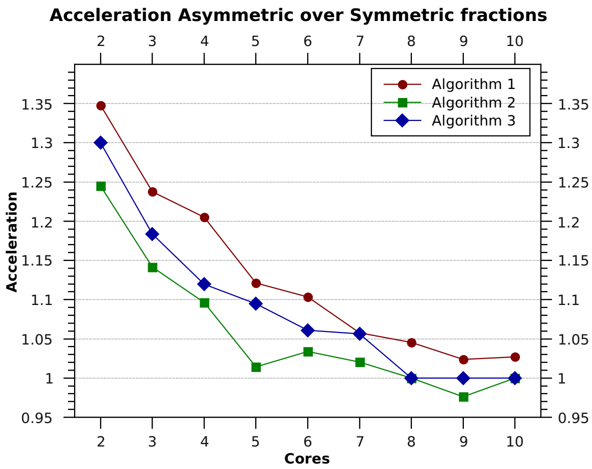

Fig. 6. Acceleration Assymmetric over symmetric fractions

IV. Conclusion

In this work an asymmetric multi-core heterogeneous microprocessors architecture on FPGA is proposed. It is aimed for fast image processing software. The acceleration is given by parallel processing. It uses a master multi-slave architecture. The master and the slaves process an asymmetric fraction of the total image. This architecture offers the following three main benefits.

First, it reduces the design time and effort by implementing the image processing in the microprocessors instead of the FPGA hardware logic.

Second, the multi-core microprocessors architecture reduce the execution time respect to a single microprocessor. This master-multi slave architecture can be used with image processing algorithms running in parallel.

Third, the asymmetric fractions of images aproach reduce the idle time on the master, thus obtaining a reduction of the

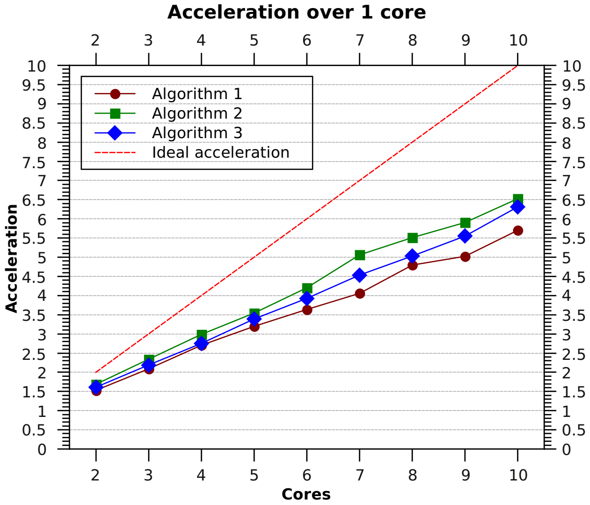

Fig. 7. Acceleration

time execution. This can be done by partitioning the image in several asymmetric fractions, and then each microprocessor processes its fraction.

Three algorithms were studied, successfully applying this architecture for image processing. This architecture was implemented using a Xilinx Virtex 5 FPGA with a PowerPC 440 embedded microprocessor.

Acknowledgment

This work was supported in part by the Agencia Nacional de Promoción Científica y Tecnológica, Argentina, under Project PICT-2009-0041.

References

[1] Xilinx, *MicroBlaze Processor Reference Guide*, 2011.

[2] ——, *Embedded Processor Block in Virtex-5 FPGAs (ug200)*, 2010.

[3] ——, “Extensible processing platform ideal solution for a wide range of embedded systems,” 2010.

[4] A. Jerraya and W. W., “Hardware/software interface codesign for embedded systems,” *Computer*, 2005.

[5] P. Huerta, J. Castillo, C. Pedraza, J. Cano, and J. I. Martine, “Symmetric multiprocessor systems on fpga,” *International Conference on Reconfigurable Computing and FPGAs*, 2009.

[6] P. Huerta, J. Castillo, J. I. Martinez, and C. Pedraza, “Exploring fpga capabilities for building symmetric multiprocessor systems,” *3rd Southern Conference on Programmable Logic*, 2007.

[7] A. Hung, W. Bishop, and A. Kennings, “Symmetric multiprocessing on programmable chips made easy,” *Proceedings Design Automation and Test in Europe*, 2005.

[8] S. Xu, C. Microsyst., O. Kingston, and H. Pollitt-Smith, “A multi-microblaze based soc system: From systemc modeling to fpga prototyping,” *Rapid System Prototyping, 2008. RSP ’08. The 19th IEEE/IFIP International Symposium on*, 2008.

[9] J. Xing, W. Zhao, and H. Hu, “An fpga-based experiment platform for multi-core system,” *Young Computer Scientists, ICYCS.*, 2008.

[10] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2010.

[11] Xilinx, *Dual processor Reference Design Suite (xpapp996)*, 2008.

[12] X. Li, B. Gunturk, and L. Zhang, “Imagedemosaicing: A systematic survey,” *Proc. SPIE, Visual Commun ImageProcess*, 2008.

[13] S. Imaging, *RGB Bayer Color and MicroLenses*, 2010.

[14] C. A. Laroche and M. A. Prescott, “Apparatus and method for adaptively interpolating a full color image utilizing chrominance gradients,” U.S. Patent 5,373,322, 1994.

[15] R. Ramanath, W. E. Snyder, G. L. Bilbro, and W. A. S. III, “Demosaiking methods for bayer color arrays,” *Journal of Electronic Imaging*, vol. 11, pp. 306–315, 2002.

[16] Xilinx, *Virtex 5 Family Brochure*, 2009. [Online]. Available: xilinx.com

[17] ——, *MicroBlaze Soft Processor Core*, 2013. [Online]. Available: xilinx.com