SOPC for Real Time Multi-Video Treatments with QoS Requirements

LAMJED TOUIL, LILIA KECHICHE BOURAOUI OUNI

Laboratory of electronics and microelectronics, University of Monastir, Tunisia

National Engineering School of Monastir. University of Monastir, Tunisia

TUNISIA
lamjedtl@yahoo.fr,

Abstract: - Today, a significant number of embedded systems focus on multimedia applications with almost insatiable demand for low-cost and high performance. Generally, the majority of video applications need to execute parallel tasks with simultaneous access to the memory. One fact is that these parallel tasks have different bandwidth requirements that have to be satisfied separately when granting access to the memory. In this paper, we propose a general purpose architecture for video applications which satisfies parallel access to the memory with different bandwidth requirements. The proposed solution is based on the Multi Port Memory Controller MPMC. The management of memory accesses is assured by using the BGPQ algorithm to guarantee QoS requirements. We demonstrate the important role of this solution in multi-video applications when multiple bandwidths are required. In fact, to successful the deployment of DRAM, it is mandatory to use a flexible and scalable interface with the appropriate arbitration algorithm. The proposed architecture is implemented using the Xilinx Virtex-5 FPGA and its available resources like embedded memory, DCM's and others. It also introduces diverse modules such as video zoom-in and out. This provides the utility of using this architecture as a universal video processing platform according to different applications requirement.

Key-Words: - BGPQ algorithm, MPMC, DDR, QoS, Image Processing.

1 Introduction

Video processing and content analysis techniques are widely used in several applications, including robotics, medical imaging manufacturing, security systems, interactive-TV and others. This type of applications is known to be resource demanding especially for real time treatments as they need a huge data transfer between memory and processing blocks with respect to defined time constraints. As a result, real time image and video processing applications require several treatment constraints to be satisfied. Hence, designing an image and video processing unit can be very complex and time consuming as the verification process can take months due to the system's complexity.

One of the most important constraints of real time video possessing is data exchange with memory. In reality, video applications have to process a huge amount of data that may present a major concern for the application's performance. This large amount of data needs to be stored in memory, transferred to the processing blocks and sent to the display unit. In addition, simultaneous implementation of various treatments on chip requires additional access to the memory. For these reasons, the DMA controller core plays an important role for the data transfer without the intervention of CPU. This solution is limited especially if a large and a parallel number of data bytes come from different video modules, in this case, the DMA cannot resolve the communication problem.

For real time video applications, traditional interfacing solutions of the SRAM and DRAM with concurrent applications present limited solutions [32-33]. In fact, most of the existing designs and SDRAM interfaces lack of generality and focus on implementing specific algorithms for domain-specific applications requirements. For example, some critical latency systems need a preemptive access while some others who are bandwidth sensitive need an equitable resource share. These competing requirements need to be satisfied as an accomplishment of a large Quality of Service (QoS) coverage. As a result, the design productivity gap is widening with every new video application. In this context, the use of an intelligent, flexible and

scalable interface to dialogue with the memory will be challenging. Such interface has to be easy for incorporation into different video processing applications and has to satisfy some antagonist QoS requirements.

In this paper, we have proposed a general-purpose platform for Image and Video Processing using Multi-Port Memory Controller (MPMC) [1] with an appropriate algorithm to grant minimal bandwidth for every connected module and to satisfy real time video constraints for a large subset of applications. The proposed architecture can be used to validate a real time video indexing and resizing applications as it satisfies the need for a general purpose hardware platform which supports and facilitates complex video and image processing applications.

The implementation of such architecture needs a good choice of hardware platform. Enabled by the latest technology advancements, a variety of hardware platforms are conceived to serve a wide range of applications. The choice of the best platforms depends on design trade-offs between efficiency, flexibility and costs.

One of the existing solutions is general-purpose processors, which are designed to be used in a particular application domain. Graphical processing units (GPU) are specialized computational units dedicated to manipulating computer graphics and data parallel computing [31]. Containing hundreds of cores, these units are known to have a high parallel structure, which allows them to process large amount of data in parallel.

Another solution is the Application-Specific Integrated Circuits (ASICS) which are conceived to perform specific tasks. In these circuits, control and computational data paths are optimized which allow them to achieve real-time performance with efficiency in energy and performance. Although their high computational performance, the mentioned platforms remains of specific purpose that limits their flexibility for any extension. Reconfigurable architectures like FPGA are one of suitable solutions as they allow modifications to the data path itself and the control flow. With this type of architecture, a run-time hardware reconfigurations is allowed which extend the application domain of the device. Hence, we have chosen the Virtex-5 FPGA reconfigurable architecture as they deliver ASIC-like density and performance, while their flexibility and operational characteristics offer distinct advantages over their ASIC counterparts. They offer a compromise between the flexibility of general purpose processors and the hardware-based speed of ASICs. Considering the requirements of such constrained applications, the use of hardware accelerators and dedicated digital systems becomes a necessity.

This paper is divided as follows: The related work and background are offered in section 2. Section 3, describes an abstraction diagram for acquisition, analysis and multi-display techniques for video processing. The proposed system is also pursued in this section. Section 4 introduces the MPMC controller solution; presents the BGPQ algorithm to satisfy the QoS of the entire system and gives the proposed video processing platform. Experimental results under different video processing are assessed in section 5. Finally, a conclusion and a discussion on future research are discussed in section 6.

2 Related Work

Last decades, real time video applications have become a focus for many searchers as the processing of a high-definition video stream in real-time presents a challenging task for embedded systems. In the literature, many platforms for real time video processing have been implemented using DSP and FPGA solutions [15][16][17][18]. These platforms are mainly oriented for specific video applications. In [19], the authors propose a general purpose and reconfigurable platform for video and image processing. The proposed platform includes several modules which can be divided into standard modules for video applications and user specific modules. The user specific modules include video zoom in and video zoom out. The proposed system can be extendable for other modules that contain additional functionalities, but it supports only a limited image size and do not present a high execution time efficiency.

One of the disadvantages of real time video applications is that they demand a lot of memory to save data while processing. Although various efforts that have been devoted , memory performance remains one of the limiting factors in evaluating system efficiency and can be considered as the bottleneck for current multimedia processing systems. A compromise between performance, cost, power consumption and reliability has to be studied while designing and interfacing shared memories. For video applications, managing the access to the shared memory remains one of the important tasks as the concurrent subsystems present heterogeneous requirements.

In [24] authors presented a system for foreground object segmentation with shadow removal. It uses a HD video scene of a 60 frame per second and 1920x1080 as a resolution. The whole system is implemented on a single Virtex 6 FPGA device,

with different modules for image acquisition, background generation, memory transactions, segmentation and result presentation. A comparison between the hardware solution and the software implementation of the same algorithms have been made by authors to show an ability to process 60 fps compared with 1 frame per 1,7 second for the software solution. Although these advanced results, Memory requirements are among the constraints that were faced when implementing the system. Hence, implemented algorithms were chosen after an excessive analyze for memory requirements. Although being an important factor, authors didn't give indication about the percentage of the used RAM while processing.

In [26], the authors implemented a Horn and Schunk algorithm for real-time motion detection on Cyclone II FPGA. To solve the high memory demand, authors used parallel storage units to facilitate memory access and computations in a single clock. Hence, the usage of 8 RAMs to store 2 consecutive frames is proposed. Although the system presents high optical flow execution rate and efficiency, the used memory remains very high (85% of the total memory bits).

One of the proposed solutions is memory controllers which have attracted many research domains as a new off-chip memory access solution [2] [3]. In fact, classical solution interface with the external memory cannot satisfy the communication and especially in the presence of several processing modules. In the last decade a number of studies are published that deal with the problem of access to the memory [4] and the treatment acceleration [5], but only few use simultaneous memory access. Also several solutions dealing with the problem of processing video in parallel using the FPGA-based system are proposed, [6] [7], but the majority of them are dedicated to a specific application. Author in [14] proposes a reconfigurable SDRAM controller that can be used in predictable virtual platforms. The controller is configurable at the run time through a command scheduler. A predictable SDRAM resource is created using bounds at the execution time of memory requests. The controller translates each request on memory into commands with a defined execution time, called a pattern. A compostable service to memory is also offered by the means of compostable memory patterns. This method allows different patterns to be loaded into the controller to get a better compromise between memory bandwidth, power and response time for active use case.

In a previous study, (34) a preliminary FPGA-based implementations of the video system was investigated. The Proposed system focus on the multi-video treatment without Qos improvement. The work in this paper focuses on developing a fully functional hardware architecture for parallel video processing with the usage of MPMC and a custom arbitrage algorithm to guarantee different QoS requirements.

3 Dataflow diagram of the proposed model

The major purpose of the proposed system is to propose a new method for improving the quality of service in case of real time multi video processing. In this context, a hardware test architecture is designed for the acquiring; processing and video display was proposed. Using this architecture, the user can add several interactive display techniques. The proposed architecture can be divided into three main modules: a standard module for video analysis, which accepts as input a video scene, a second module which performs the necessary operations to prepare for exploitable results to be used for output and the third module reserved for additional display techniques like zoom in, zoom-out and others. The division of the system to different modules aims to separate functional blocks in order to simplify any wish to add extra blocks.

3.1 Acquisition module

The first module is responsible for acquiring video signals and its introduction to the system to be processed. The components that were included for decoding were ‘line decoder’, ‘4 : 2 : 2 to 4 : 4 :4’,‘Ycrcb to RGB’, ‘Timing generation’, ‘De-entrelacement’, these blocks performed the task of converting the data from composite format to RGB and also kept track of the VSynch and HSynch to determine new frame and new line. Figure 1 presents the acquisition module.

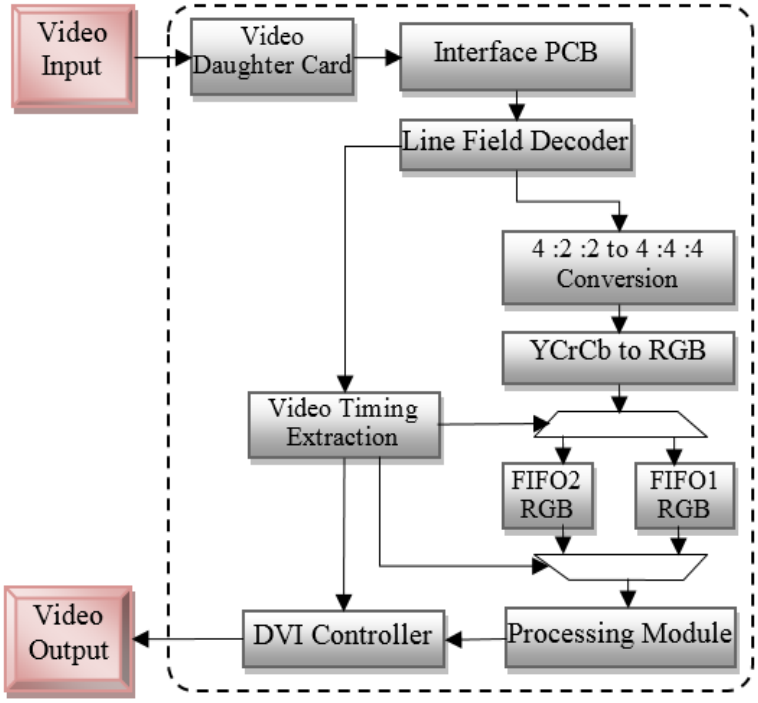

Figure 1: Acquisition module

3.2. Analyze module

The analyze block is responsible for analyzing video signals in order to extract high and semantic results. This block can be separated into 3 different levels: low-level, mid-level and high level treatment (figure 2)

.

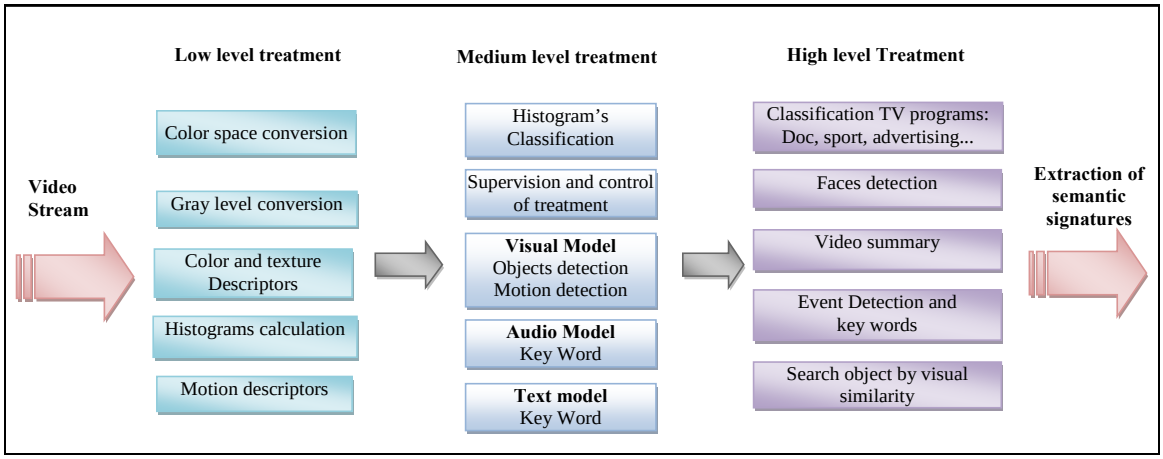

Figure 2: Analysing treatments

3.3. The display module

The display module ensures the visualization of video content with different display methods chosen by the user such as: multi-display, video zoom-in, video zoom-out. In fact, the video content analysis results are generated and then applied to the display block validation techniques which are based on Bilinear Interpolation method [9]. This block can be separated into two levels: low-level, and high level treatment. Several display technologies and exploitation of results are possible. Among the techniques we quote:

Zoom-In: this method aims to increase a part of the screen according to the wishes of viewers.
Zoom-out: this method describes the ability to reduce the size of the displayed image.
PIP: Picture In Picture, this method designs the display of two pictures (main with big size and sub) simultaneously. The main is displayed in full screen while the sub image is displayed in a corner of the screen
POP: Stands for picture-outside-picture, this method gives the ability to divide the screen into two same-size pictures.
Video Mosaic: this method aims to display many video sources at the same time in on the same TV screen.
Resizing data, either upscaling or downscaling, is an important signal processing technique due to the variety of data formats and sources used today. Several techniques has been developed [26][27][28][29][30], but in this work we choose the bilinear interpolation for its simplicity and efficacy.
The bilinear interpolation uses the four nearest pixel values that are located in diagonal direction from that specific pixel in order to find the appropriate color intensity value of a desired pixel. New pixels between line n and line n+1 are generated with a combination factor of (½), then new pixels between the two vertical pixel lines are recreated.

3.4. Design methodology

In this work, we have used Platform-based design (PBD) methodology [8] combined with Bus Based Approach (BBA). PBD uses an existing base of components and architectures to decrease design time and uses IP blocks to build system architectures. PBD separates the HW/SW partition into behavior approach, architecture approach, and mapping. This method supports IP reuse, and derivative design.
For the BBA methodology, components will be designed around an on-chip bus architecture. This approach needs the interface of the concerned bus protocol to be integrated, which leads to the creation of a logic of communication. The bus behaves as a generic component responsible for interfacing blocks and as a bridge if there is more than a bus in the system architecture. Combining both PDB and BBA approaches makes the platform more flexible for the addition of external blocks. Hence, it is possible to add other blocks with simplification over their interfacing and integration.
Based on the proposed structure, we aim to develop a model system based on reconfigurable architecture [10] [11]. In this paper, we propose a preliminary version of hardware architecture to support several video processing applications.

4 Proposed Architecture Of the Entire System

The general architecture of the application is given by figure 3. As previously explained, the entire design consists of three essential modules: acquisition module, display module and video processing module. These three blocks are connected to independent ports of the MPMC (figure4). One port can be servicing a read or write while another is servicing a different read or write. Access to the memory bus is arbitrated between all ports using the BPGQ algorithm.

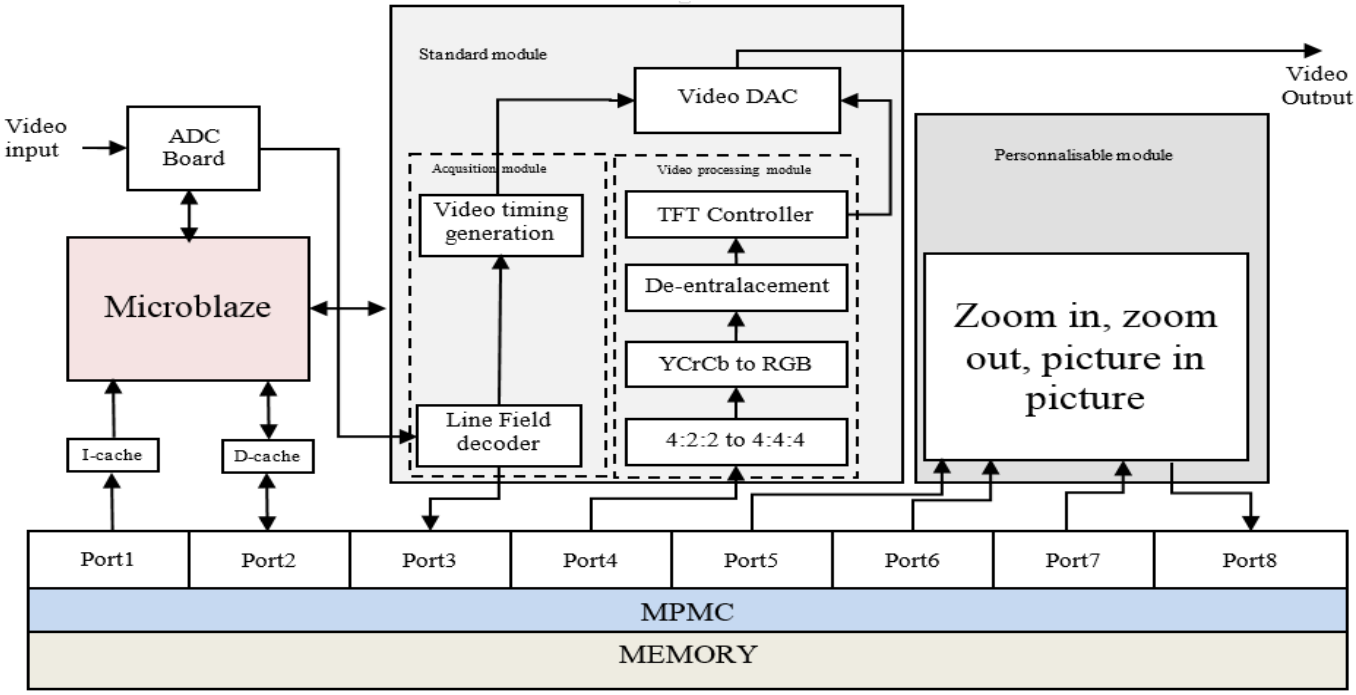

Figure 3: Data processing of the proposed design.

In the next paragraphs, we will talk about the proposed solution for memory management of the different access requests. In fact we have chosen the Multi Port Memory Access (MPMC) to manage access to the shared off-chip memory.

4.1 The MPMC

The MPMC is an eight-port DDR SDRAM memory controller, where each port can be chosen from a set of personality interface module (PIMs). The DDR transmits controls and data on the bus. The connection of the ports to the microprocessor (Hard or Soft) is assured by PLB bus and the MPMC Native Port Interface (NPI). MPMC supports the Soft Direct Memory Access (SDMA) controller that provides full-duplex high-bandwidth. A Video Frame Buffer Controller (VFBC) PIM is also available. Figure 4 depicts the architecture of MPMC.

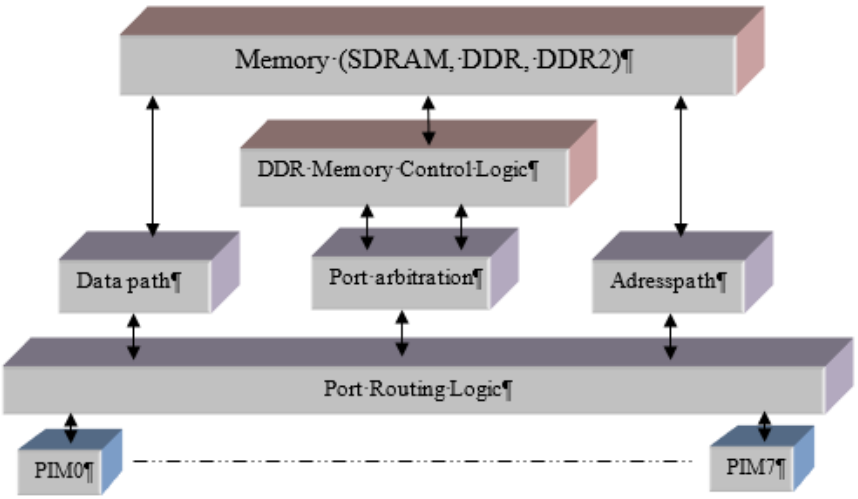

Figure 4: MPMC Architecture Block Diagram

The FPGA's internal memory is not enough to hold real time images, thus, off-chip memories are provided. For this type of memories, the bandwidth represents one of the important and key resources for digital systems. Hence, sharing it by different subsystems is a growing demand for its low cost and

pins count. Thus, an interface coordinating different blocks with the external memory is required.

The MPMC plays the role of coordinator for the access to the shared DDRAM from different blocks. An MPMC scheduler arbitrates requests and roots them from and to the off-chip memory at every clock cycle. One of the challenges that face the MPMC is to optimize the competing requirement of these blocks. These requirements essentially concern the satisfaction of an optimized latency and a guaranteed bandwidth. In the proposed architecture, the MPMC design has to take in consideration the following requirements:

The MPMC has to satisfy the various set of bandwidth requirement of the connected ports. For example the line field decoder (figure2) connected to port 3 requires 94 band width while the zoom in connected to port 6 requires 34 bandwidth.

The MPMC has to satisfy latency requirements for all blocks without violation. A latency sensitive block has to be serviced as early as possible.

The MPMC must simultaneously satisfy a diverse set of bandwidth requirements from all ports. Table I shows the bandwidth requirements for some blocks connected to the MPMC ports. According to this table, the bandwidth requirement differs from 6.2 to 94.

Suppose that we have N ports, the MPMC is responsible for the arbitration of requests and the grant of access to the memory for these N ports. Each port has a queue $i \epsilon [0 \dots N-1]$ and an associated bandwidth guarantee noted Si with $\sum_i S_i \leq 1$.

To achieve a guaranteed bandwidth some algorithms try to establish a fair access to the memory for all users such as Weighted Fair Queue (WFQ) and Round Robin. Although WFQ can improve latency by granting access to high priority request, the combination of such algorithms do not allow to attain both prior access and bandwidth fairness. Another drawback is the absence of a real management over the residual bandwidth. For these reasons we have chosen to work with the Bandwidth Guarantee Priority Access (BGPQ) algorithm.

TABLE1: BANDWIDTH VALUE FOR SOME BLOCKS.

Port	Block	Bandwidth
1	I-cache	6.2
2	D-cache	9.3
3	Line Field Decoder	94
4	4:2:2 to 4:4:4	72.5
5	Zoom In	53
6	Zoom Out	34
8	Output	62.1

4.2.BGPQ algorithm

The BGPQ [20] [21] is an algorithm which main objectives are to provide the minimal necessary bandwidth for different ports and allow prior classes to use the residual bandwidth. For every scheduling cycle, the BGPQ computes the dynamic bandwidth utilization of every port. Suppose that we have N ports, for each one we have: a queue for arriving requests, an empty flag Ei , an active flag Ai and a bandwidth guarantee Si where $\sum_{i=0}^{N-1} S_i \leq 1$.

A dynamic credit is computed at the beginning of the (i+1)th scheduling cycle as follows:

$$D_i(n+1) = D_i(n) + A_i(n).S_i + K_i(n), i \in \{0, \dots, N-1\} \quad (1)$$

$$A_i(n) = !\,E_i(n)$$

$$k_i(n) = \begin{cases} -1 \; if \; D_i(n) = max_{j \in \{0,\dots,n-1\}} \{ D_j(n) | A_j(n) \} \\ 0 \; otherwise. \end{cases} \quad (2)$$

The dynamic credit Di(n) of the current cycle equals to the dynamic credit of previous cycle Di(n-1). The credit $A_i(n).S_i + K_i(n)$ is added for active queues, for the others, the dynamic credit remains the same and as a result they lose access to scheduling resources.

If $\sum_{i=0}^{N-1}$ Ai(n).Si≤1, the granted bandwidth is not wholly used and as a result we have a residual bandwidth that can allocated for the most prior class. The total residual bandwidth is computed as follows:

$$d(n) = 1 - \sum_{i=0}^{N-1} A_i(n).S_i \quad (3)$$

The dynamic credit becomes:

$$D_i(n+1) = D_i(n) + A_i(n).S_i + K_i(n) + R_i(n), i \in \{0, \dots, N-1\} \quad (4)$$

With:

$$R_i(n) = \begin{cases} d(n) \text{ if } min_{j \in \{0,\dots,n-1\}}\{j | A_j(n)\} \\ 0 \text{ otherwise.} \end{cases} \quad (5)$$

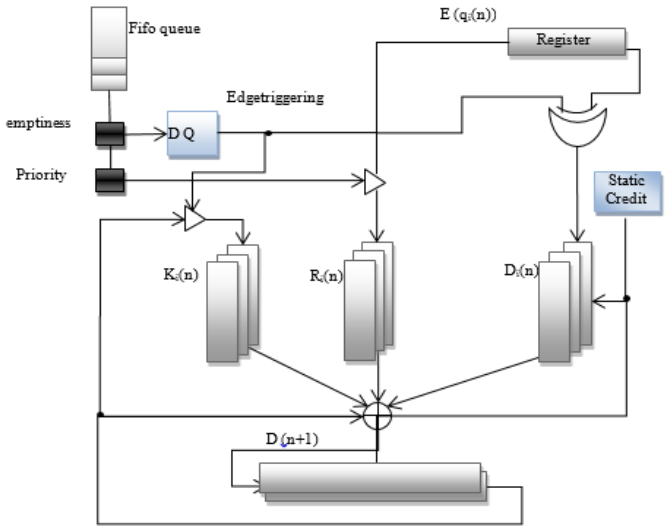

Figure 5: A block diagram of the BGPQ algorithm

By default, the MPMC uses the Round Robin arbitration algorithm, which gives an equal overall priority for each port. Compared to this last, the BGPQ algorithm has the following advantages:

- A better control over residual bandwidth is provided by granting access to this critical resource for tasks with the highest priority calculated at the run time.
- An easy implementation due to its low complexity.

Figure 5 shows the proposed hardware architecture to implement the BGPQ algorithm

4.3.Functioning of the system

Using the MPMC was crucial to the success of the design as previously explained and because of bottlenecks which present a problem in transferring the video data to d from memory. This problem was solved by using available ports and additional buses. In the system, we used MPMC with “CoreConnect” bus and especially Processor Local Bus “PLB” which is obtained from the Xilinx IP library under the name of “plb_v64”. The PLB bus offers a connection between a PLB masters and slaves. It presents the essential element for transport data from one block to another.

In every memory write, data and control synchronization signals are accessible at an MPMC port. When the write data buffer (burst buffer) is not empty, the Multi Port Memory Controller transfers the data into the burst buffer after acknowledges the request. The data persists in this buffer until the memory bus is accessible. MPMC ports are delivered as follows: port 1 and 2 are assigned for Microblaze Data/Instruction cash. The Microblaze microprocessor core is a 32-bit RISC soft core processor using Harvard architecture optimizing for embedded applications. It has a memory management unit, floating point unit, instructions and data cashes and many other optimization functions. The microprocessor Data/instruction cash bus interface to the multi-port memory controller. The Microblaze module provides direct memory access to the processor IXCL and DXCL interfaces; it also can be connected directly to the PLB bus attached to the PLB PIM. Port 3 is assigned for video input. Port 4 is assigned for video output display. Ports 5, 6, 7 and 8 are reserved for the multi-video techniques block. Hence, the MPMC is

managing two kinds of connected blocks: ports 1,2,3,4 are reserved for standard blocks that are the main modules in every video application: from the input to the display. Ports 5,6,7,8 are reserved for customizable blocks that can be add by user.

In this architecture, we have proposed a combination of hardware and software solution to get both rapidity and flexibility. We used the hardware solution to implement the blocks which are multiple slaves IP cores based on Multi Port Memory Access. The software solution is adopted for making the choice of one of the wanted display techniques more flexible. The user has the possibility to choose one of the available display possibilities through push bottoms. Hence, the Microblaze assures the following roles:

The configuration of the ADC board [22].

The management of the choice of the user (choice of the display technique).

The management of the hardware modules by activating the module to be executed.

The operating frequencies of DDR memory, MPMC and the videooutput are respectively 133 MHz, 100MHz and 27MHz. every one of the three main modules has functioning blocks. For example,there are 8 functioning blocks in the acquisition module composed as follow: line decoder, 4:2:2 to 4:4:4 conversion, YCrCb to RGB transformation, Timing video, Decoder, de-interlacement (figure 2).

Using only one bus is insufficient to handle transactions. In fact, word cannot be transferred on every period of the bus, since the buffers are not all the time ready. It takes some cycles to setup a burst write/read. To remedy this situation, we formed two modules for communicate processing reading and writing. Each module has its own PLB bus linking to the memory. This gives the system more flexibility and bandwidth and made the design easier to understand handles and debug. PLB is based on IBM's 64-bits CoreConnect technology. It uses an arbitration policy to control the slaves devices attached to the bus. It has 64 bits wide data bus, 32 bits wide address bus, and eight words cache line transfers.

The XPS_TFT controller is an IP core from the EDK library. It is an IP core provided by the Xilinx core library [23]. It connects to the PLB_v4.6 bus and act as a PLB master. The XPS_TFT reads video frames from MPMC and generates signals to DVI port.

5 Experimental Results and Discussion

In this work, we have compared results found by the solution based on the MPMC to the conventional solution based on Xilinx CoreConnect bus with multiple masters and a processor instruction/data caches with key high-speed devices connected by a slave port to the SDRAM. We also conducted an evaluation by constructing a virtual prototype in Mirabilis Design's VisualSim for a performance modeling and architecture exploration solution. Table 3 summarizes the major resource utilization characteristics of the system, from which one see the final design utilizes about 30% of memory on chip and 16% of logic resource.

TABLE 2: RESOURCE UTILIZATION OF THE ENTIRE SYSTEM

Hardware resources	Available	Used	Utilization
Logic slices	17,280	5,896	30%
Bonded IOBs	800	80	10%
CLB	69,120	11,754	16%
Logic Cells	110,592	14,252	13%
DSP-48E Slices	64	10	16%
DCM	12	2	17%
IO Banks	23	5	22%
Block RAM	4608	1314	28.5%

In [19] the system is implemented on the Xilinx Virtex-II FPGA using the PowerPc processor and the internal memory. The provided experimental results show a moderated resource consumption as it consumes 50% of the memory and 20% of the logical resources. Although these results claim the

resource consumption to be low, adding extra modules will be relatively limited due to the use of internal memory and a hardware processor.

In our proposed study, we have shown that the use of a purely material using the "MPMC" with an appropriate algorithm combined with CoreConnect solution give better results compared to the "coreConnect" based solution. Table 3 shows the instruction and data cache statistics for the virtual prototype model. We ran the simulation on a 2.4 GHz Microsoft Windows XP (SP3 and Standard Edition) with 512 Mb of cache.

Using the 200-MHz MPMC, the end-to-end latency for the execution of the application benchmark was 89.254 µs, while the 400-MHz CoreConnect was 88.13 µs. Both matched our real-time threshold. We found that the MPMC Memory Controller typically finished its tasks faster than the CoreConnect bus. The MPMC Memory Controller had a significantly better average hit ratio (93.72% versus 89.35%) for the Icache (instruction). At certain times during the simulation, the CoreConnect bus did get to a 93% hit ratio, but the duration was very short. This indicated that the MPMC arbitrated SDRAM with an appropriate arbiter system quests better than the classic solution based only on CoreConnect bus. On the other hand, the design of the application was based on the creation of set of IP and the reuse of others. The VHDL language is proposed to describe the different modules of the proposed system and also EDK Embedded Development Kit and ISE tools are used in the implementation of our application.

TABLE 3: INSTRUCTION AND DATA CACHE STATISTICS

Statistic Name	**MPMC(classic)**		**MPMC(with BGPQ)**	
	Inst. Cache	Data cache	Inst. Cache	Data cache
Hit_Ratio_Max	94.03	95.2	93.72	96.23
Hit_Ratio_Mean	89.35	89.43	93.72	96.23
Throughput_MIPs_Max	72.33	2.13	8.54	0.22
Throughput_MIPs_Mean	7.13	0.12	8.54	0.22

With:

Hit_Ratio_Max: Is the maximum percentage of memory accesses satisfied by the cache.

Throughput_MIPs_Max: is the average rate of successful message delivery over a communication port.

The BGPQ architecture is proposed for increasing the efficiency of hardware utilization and decreasing hardware cost. To examine the design quality, the Hardware Utilization Rate (HUR) was used as a performance index. The HUR can be presented as follows:

$$HUR = \frac{\sum_1^n \alpha_i}{n * \alpha_{st}}$$

The HURs of the proposed Hardware Architecture without BGPQ algorithm and with BGPQ algorithm are reported in Table 4. In regard to the video sequence with image size 1280*720, the hardware utilization rates of the proposed architecture without BGPQ algorithm and with BGPQ algorithm are 42.23% and 69.48%. For the image size of 1920*1080, the Hardware Utilization rate with The BGPQ algorithm and without BGPQ algorithm are 37.52% and 61.35%. It shows that the HUR of the proposed hardware architecture with the BGPQ algorithm is 1.17 times that the classic solution.

The classic method based on MPMC module achieves an average frame rate of only 9.2 fps at high definition resolution 1920 × 1080. We have proven that the proposed method with MPMC module and BGPQ algorithm can achieve an average frame rate of 34.56 fps at high definition resolution 1920 × 1080.

TABLE 4: HARDWARE UTILISATION(%)

Image resolution	**1280*720**	**1920*1080**
Without BGPQ	42.48	69.48
With BGPQ	37.52	61.35

Ratio	1.17	1.17

6 Conclusion

In this paper, we have proposed a preliminary version of hardware architecture to support several video processing applications. One of the problems faced while implementing such model is the management of competing access to the shared memory with respect to different QoS requirements like bandwidth and latency. Accessing the same memory simultaneously generates memory conflicts which increases data latency and reduces the memory port bandwidth. In this context, we have tried to give a suitable solution to easily and efficiently manage the shared memory and give the possibility for the system to add other modules with respect to the QoS requirements. In this paper, we have demonstrated the important role of the MPMC controlled by a suitable algorithm. The MPMC interface combined with the BGPQ algorithm ensures the respect of different QoS requirements for real time video processing. Also, we have demonstrated that the MPMC with BGPQ algorithm based method combined with CoreConnect bus solution is more efficient compared with the classic method based on bus MPMC with Connect only to ensure the parallelism of several communication treatments with external memory. The proposed architecture presents a preliminary prototype system for the multi-task video and the image processing.

The final system was implemented using the Xilinx Virtex–5 development system. This system provides a scalable and real-time reconfigurable platform to meet the requirements for many video processing applications. Furthermore, the reconfigurable and extendable characteristics of this system allow it to be easily modified to embed into different video and image processing scenarios experimental results. In this paper we have presented a solution based on reconfigurable technology for the validation of processing applications for real-time video.

In the future work, it would be interesting to integrate more complicated video processing modules [12] into this platform.

References

[1] LogiCORE IP Multi-Port Memory Controller (MPMC), (v6.03.a), DS643, March, 1, 2011.

[2] M. Bojnordi and E. Ipek. PARDIS: A programmable memory controller for the DDRx interfacing standards. In Proc. ISCA, 2012.

[3] J. Reineke et al. PRET DRAM Controller: Bank Privatization for Predictability and Temporal Isolation. In Proc. CODES+ISSS, 2011.

[4] C. Pilato, F. Ferrandi, D. Sciuto, "A design methodology to implement memory accesses in High-Level Synthesis", Published in: Hardware/Software Codesign and System Synthesis, Proceedings of the 9th International Conference on, Page(s): 49 – 58, Taipei, Oct. 2011.

[5] A. Canis, J. Choi, M. Aldham, V. Zhang, A. Kammoona, J. Anderson, S. Brown, T. Czajkowski, "LegUp: high-level synthesis for FPGA-based processor/accelerator systems" in ACM/SIGDA, FPGA, Volume 3 Issue 3, Article No. 41 ACM, May, 2012.

[6] C. He, A. Papakonstantinou,D. Chen, "A novel SoC architecture on FPGA for ultra fast face detection", Published in: Computer Design, ICCD, IEEE International Conference on 2009.

[7] S. McBader, P. Lee, "An FPGA implementation of a flexible, parallel image processing architecture suitable for embedded vision systems", Parallel and Distributed Processing Symposium, Proceedings. International April 2003.

[8] A. Sangiovanni-Vincentelli, L. Carloni, F. D. Bernardinis, M. Sgroi, “Benefits and Challenges for Platform-Based Design,” Proceedings of Design Automation Conference, pp.409-414, 2004.

[9] K. T. Gribbon, D.G Bailey, "A Novel Approach to Real-time Bilinear Interpolation": Proceedings of the Second IEEE International Workshop on Electronic Design, Test and Applications, IEEE, 0-7695-2081-2/04 2004.

[10] XUPV5-LX110T MIG Design Creation Using ISE 12.2, MIG 3.5 and ChipScope TM August, 2010.

[11] Xilinx University Program XUPV5-LX110TDevelopment System.

[12] L. Pantaleone and E. Todorovich, "Accelerating embedded software processing in an FPGA with PowerPC and Microblaze" , PICAT project, 2011.

[13] L. Touil, A. ben Abdelali, A. Mtibaa, Elbey Bourennane. "Towards Hardware implementation of video applications in new telecommunications devices", J T, 2(1):75-85, 04/2010

[14] Sven Goossens, Jasper Kuijsten, Benny Akesson, Kees Goossens, a recongigurable real time SDRAM controller for mixed-time criticality system, IEEE conference on hardware/ software codesign and system synthesis, 2013.

[15] C. Desmouliers et al., HW/SW Co-design Platform for Image and Video Processing Applications on Virtex-5 FPGA Using PICO, IEEE International Conference onElectro/Information Technology (EIT), 2010.

[16] Sudeep K C, Jharna Majumdar, A Novel Architecture for Real Time Implementation of Edge Detectors on FPGA, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 1, 2011.

[17] Wajdi Elhamzi et al.,FPGA Based Real Time Wavelet Video Coding, International Review on Computers and Software (I.RE.CO.S.), 2013

[18] J. Batlle, J. Martı, P. Ridao , J. Amat, A New FPGA/DSP-Based Parallel Architecture for Real-Time Image Processing, Real-Time Imaging, Elsevier, 2002.

[19] Jie Li, Haibo He, Hong Man, Sachi Desai, A General-Purpose FPGA-Based Reconfigurable Platform for Video and Image Processing,Advances in Neural Networks ISNN, Springer, 2009.

[20] Zefu Dai, Mark Jarvin, Jianwen Zhu, credit borrow and repay: sharing DRAM with minimal latency and bandwidth guarantees, International Conference on Computer-Aided Design (ICCAD) IEEE/ACM, 2010.

[21] KL Eddie Law, the bandwidth guaranteed priotirized queuing and its implementation, Global Telecommunications Conference, IEEE, 1997.

[22] Diligent Video Decoder Board (VDEC1) Reference Manual, Revision: 4 December, 2005,

[23] The XPS Thin Film Transistor (TFT) controller, http://www.xilinx.com/support/documentation/ip_documentation/xps_tft.pdf , DS695, 2009.

[24] Tomasz Kryjak, Mateusz Komorkiewicz, Marek Gorgon, Real-time background generation and foreground object segmentation for high-definition colour video stream in FPGA device, J Real-Time Image Proc, 2012.

[25] Mateusz Komorkiewicz , Tomasz Kryjak, Marek Gorgon, Efficient Hardware Implementation of the Horn-Schunck Algorithm for High-Resolution Real-Time Dense Optical Flow Sensor, Sensors, volume 14 issue 12, 2014.

[26] A. Munoz, T. Blu, and M. Unser, “Least-Squares Image Resizing Using Finite Differences,” IEEE Transactions on Image Processing, Vol. 10, No. 9, September 2001.

[27] C. Hentschel, “Generic Method for 2D Image Resizing with Non-Separable Filters,” IEEE International Conference on Image Processing, 2004.

[28] Y. Park and H. Park, “Design and Analysis of an Image Resizing Filter in the Block-DCT Domain,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 14, No. 2, February 2004.

[29] J. Hwang and H. Lee, “Adaptive Image Interpolation Based on Local Gradient Features,” IEEE Signal Processing Letters, Vol. 11, No. 3, March 2004.

[30] L. Chen and K. Yap, “Regularized Interpolation Using Kronecker Product for Still Images,” IEEE International Conference on Image Processing, Vol. 2, September 2005.

[31] Shinpei Kato, Karthik Lakshmanan , Ragunathan Rajkumar, Yutaka Ishikawa, TimeGraph: GPU Scheduling for Real-TimeMulti-Tasking Environments, USENIX Annual Technical Conference 17 , 2011.

[32] Alexsandro,B., Altamiro S., Memory Subsystem Architecture Design for Multimedia Applications,IEEE Computer Society Annual Symposium on VLSI, 2013.

[33] Joonseok, P., Pedro D.,Synthesis of Pipelined Memory Access Controllers for Streamed Data Applications on FPGA-based Computing Engines , Proceedings of the 14th international symposium on Systems synthesis, 2001.

[34] Touil, L., Kechiche, L., Ouni., B., "Generic SOPC Platform for Video Interactive System with MPMC Controller", International Journal of Embedded Systems and Applications (IJESA) Vol.4, No. 1, March 2014.