Applied Mathematics & Information Sciences
An International Journal

Novel Memory Access Scheduling Algorithms for a Surveillance System

Slo-Li Chu and Min-Jen Lo*

Department of Information and Computer Engineering, Chung Yuan Christian University, Chung Li, 32023, Taiwan

Received: 27 Sep. 2012; Revised 11 Dec. 2012; Accepted 28 Dec. 2012
Published online: 1 Mar. 2013

Abstract: The continuously growing functionality of digital video surveillance make the surveillance system integrate more streaming processors for serving more cameras to recoding their raw video streaming data. But the memory subsystem can not provide necessary bandwidth and become the bottleneck of whole system. Therein how to improve the performance of the accessing memory will become a major challenge of designing a modern surveillance system. This study proposes novel memory accessing scheduling algorithms, with a corresponding memory controller, called Self-Adjustable Memory System (SAMS), for a multiple-channel streaming system-on-a-chip. By integrating Access Buffers, Frontend Scheduler, Reorder Block, Backend Scheduler, and two scheduling algorithms, SAMS can provide a sufficient memory bandwidth for the streaming processors with high bandwidth requirements. The utilization of multiple DRAM banks can be improved accordingly. The experimental results illustrate that SAMS will arrange enough bandwidth for the streaming processors that have bursting transferring requirement. The enhanced speedup can achieve 3.9X than conventional memory subsystem.

Keywords: Memory access scheduling, self-adjustable memory system, surveillance system, SystemC.

1. Introduction

Due to the more functional requirement of consumers in digital video surveillance, the designers of the surveillance system need to integrate more multimedia Intelligent Property (IP) in the system, to serve more cameras to coding their raw video streaming data. The typical surveillance system is as illustrated in Fig. 1, which integrates several streaming processors, CPU, video and audio interface, and multiple DRAM banks into a single chip. Although the number of external DRAM banks is increased, the additional memory bandwidth cannot be fully utilized due to the limitation of conventional interconnection network in the surveillance system. Therein how to improve the performance of the memory subsystem will become a major challenge of designing a modern surveillance system. According to our previous study of multimedia system [2], we find an important attribute. When the multimedia system-on-chip (SOC) integrates multiple streaming processors, the achievement of the bandwidth are not only by total memory throughput but also by dynamically adjusting the bandwidth usage of multimedia processors, which can guarantee the good utilization of limited memory bandwidth. Therefore in this paper, two memory access scheduling algorithms with corresponding mechanisms, called Self-Adjustable Memory System (SAMS), is proposed for a multiple-channel surveillance system. The proposed SAMS system integrates a Frontend Scheduler, Reorder Block, and Backend Scheduler, to provide a sufficient memory bandwidth for the streaming processors with high bandwidth requirements. The performance bottleneck of the memory access in the modern surveillance system also can be solved. The utilization of multiple DRAM banks can be improved by the management of proposed mechanisms. The experimental results illustrates that SAMS will arrange enough bandwidth for the streaming processors that have bursting transferring requirement. The proposed SAMS architecture has been implemented by SystemC and Bluespec SystemVerilog. The fabrication results of SAMS are also provided.

The rest of this paper is organized as follows. Section 2 briefly discusses related works. Section 3 presents the detailed architecture and execution follow of proposed SAMS architecture. Section 4 shows the experimental re-

* Corresponding author: e-mail: slchu@cycu.edu.tw

sult of SAMS system. Finally, the concluding remark is proposed in Section 5.

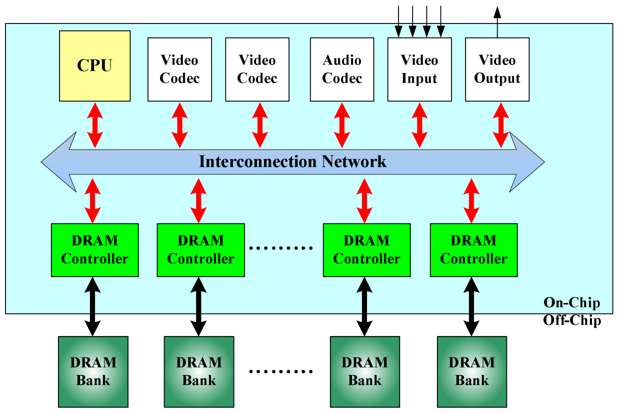

Figure 1 Architecture diagram of a typical surveillance system.

2. Related Works

2.1. Arbitration of Interconnection Network

In the conventional SOC systems, the corresponding interconnection networks are Bus architectures, which integrate the attached multimedia processor and peripheral IPs, where the bus arbiter is usually the performance bottleneck. Several bus arbitration mechanisms of the bus arbiter will be discussed as below.

Static Priority: This arbitration [3] provides a fixed priority for each master IP. If multiple master IPs send request to the arbiter at the same time, the arbiter will grant the master IP with the highest priority. The advantage of this mechanism is simple and low hardware cost; the disadvantage is the starvation of low priority IP when the high priority IP sends request frequently.

Time Division Multiplexing (TDM): TDM [3] can be simply partitioned into two classes, one-level TDM and two-level TDM. One-level TDM divides the bus access time into several time slots, and then dispatch these time slots to the master IPs. The master IP who takes its own time slot can send request to the bus arbiter. If the master IP who owns the time slot will not send request, the arbiter will schedule next request till the current time slot is timeout. It will worst the unused time slot and lose the utilization of bus. Therefore, the two-level TDM is proposed to solve the problem of low utilization. The two-level TDM consists of two scheduling stages. The first level scheduling is the same as one-level TDM which divide access time into several time slots. If the allocated time slot is unused, the second level arbitration mechanism will schedule the next candidate master IP to use this time slot. It will obtain better bus utilization by arranging next master IP. The round-robin mechanism is usually adopted in the second level arbitration.

Lottery Scheduling: Lottery Scheduling mechanism [4] firstly assigns an unique ID to each master IP as a ”lottery ticket”. Then it generates a ”lottery number” by using its random generator. If the lottery number is the same as the lottery ticket of the master IP, this master will obtain the grant. If the lottery ticket doesn’t match any available lottery number, it will adopt a weighting function to figure out the next candidate master IP for granting.

2.2. Commercial Surveillance Systems

The typical surveillance system contains a CPU, memory controllers, peripherals, and bus interconnection network. It also contains multiple streaming IPs, multimedia codes, video and audio interfaces. All of the master and slave IPs are controlled by CPU via bus interconnection network to share the multimedia codes, access the data in the external DRAM banks, and manipulate the video and audio streaming data. Some of the systems adopted in surveillance systems are introduced as below.

Faraday FIE8180: This chip is proposed by Faraday Inc. [5], which consists of FA626 CPU, DDR SDRAM controller, video capture channels, H.264 codecs, MPEG4 codecs, JPEG codecs, and other required peripherals, and then integrates by ARM AMBA AHB bus. The fast IPs are connected by processor-side AHB bus, and the slow peripherals are connected by peripheral-side AHB and APB buses. The surveillance streaming data are retrieved from dual video capture channels, and then store into external DDR SDRAM banks. Then the video codes process the raw streaming data and convert to the video formats assigned by users. The processed streaming files can be stored into storage or transfer to internet for further play.

Mobilygen MG3500: This chip is designed by Mobilygen Inc. [6], which integrates ARM926 processor, external DDR2 SDRAM controllers, multimedia IPs, HD H.264 codecs, MPEG2 decoder, JPEG codec, audio codec, video input processors and required peripherals. The above components are integrated by multi-level AHB bus. Since the streaming codecs and general peripheral IPs are separated into two different buses, the system throughput can be increased.

TI TMS320DM365: This system [7], provided by TI, focuses on digital multimedia applications. By integrating Video Processing Subsystem (VPSS), ARM926 processor, DDR2 memory controllers, voice codec, required peripherals, and other multimedia IPs into a single chip, this architecture can performance high performance in video surveillance. It also contains a buffer to improve the actual transferring bandwidth from bus to external DDR2 DRAM banks.

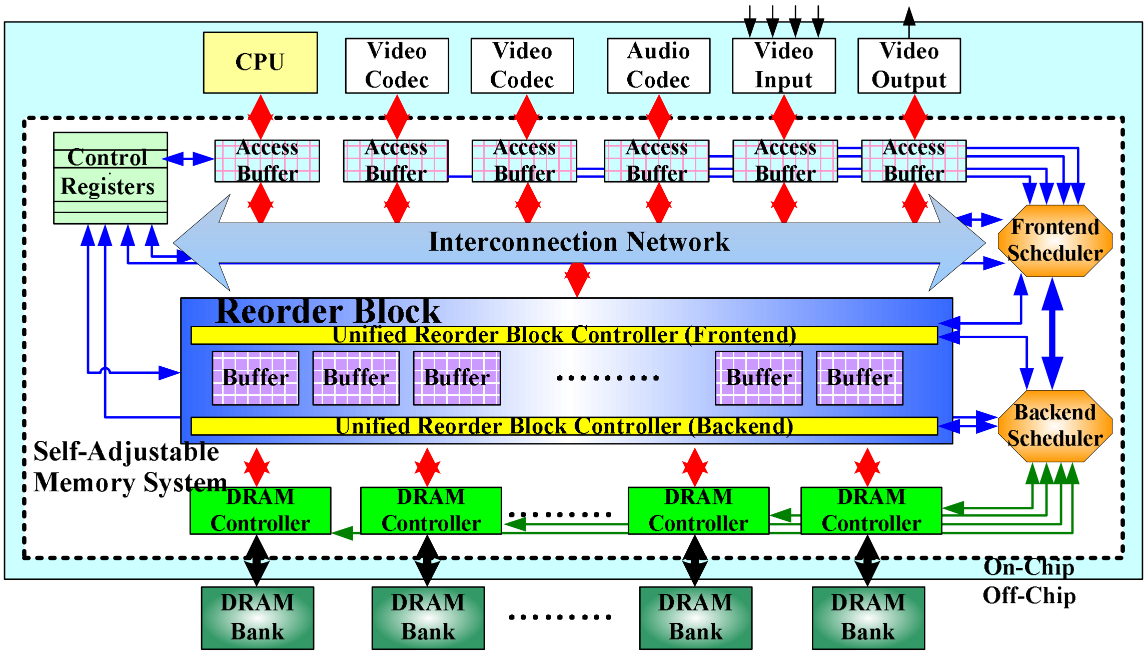

Figure 2 The architecture of proposed SAMS system.

3. The Architecture of Self-Adjustable Memory System

As the growing requirements of channel number of digital camera and high definition video stream in the modern surveillance chip, the high memory bandwidth requirements from accessing memory by these streaming IPs become the major performance bottleneck. Many architectures consist of multiple external DRAM banks to enlarge the total theoretical memory bandwidth. However, the increased external DRAM channels will increase the complexity of bus arbitration. The actually transferring bandwidth will be limited accordingly. In this paper, a new memory controller, call Self-Adjustable Memory System (SAMS) is proposed to solve the above problem. The architecture of surveillance SOC with SAMS system is as shown in Fig. 2, which consists of multiple streaming IPs, video codecs, audio code, and the proposed SAMS system. The SAMS system can utilize the memory bandwidth from additional external DRAM banks due to the cooperation of Frontend Scheduler, Backend Scheduler, and Reorder Block. The functionalities of the modules in the SAMS are mentioned below.

–**Access Buffer** serves as the buffer of streaming IPs and multimedia codecs, which can temporarily store the data and improve the transferring bandwidth. The configuration of Access Buffer can be set by Control Registers. The runtime statistical data will send to the Frontend Scheduler for further arbitration and scheduling.

–**Interconnection Network** is the communication channels between Access Buffer and Reorder Block. Since the streaming IPs and external DRAM banks increased, the conventional interconnection network, such as shown in Fig. 1, can not provide enough effective bandwidth. Compared to the conventional interconnection network, the proposed Interconnection network can dynamically self-adjust the bandwidth by cooperating the crossbar-based structure and Frontend Scheduler.

–**Frontend Scheduler**, as shown in Fig. 2, arbitrates all simultaneously requests of Access Buffers, and then decides a suitable Access Buffer to send results into Reorder Block. The scheduling mechanism will be discussed later. It also monitors and controls the current flows of Interconnect Network, and then decides a suitable arbitration results for the Interconnection Network.

–**Reorder Block** is composed by unified SRAM blocks which can dynamically reconfigure the region and size according the schedule results of Frontend Scheduler. The data from Access Buffer can be stored into a Buffer in the Reorder Block. When a transfer is completed, the Buffer will be recycled for the further usage. Therefore it can be dynamically allocated and recycled. Since the data transfer from Access Buffers may have locality, it contains a detecting mechanism for internal forwarding by adopting the packing results of Unified Reorder Block Controller (Frontend). It also can reduce the arbitration cost of Interconnection Network by collecting the runtime transferring attributes.

–**Backend Scheduler** is responsible for scheduling and arbitrating the transferring requests of Blocks in the

Reorder Block. Since the number of external DRAM banks increased, the scheduling mechanism has to be improved, instead of conventional round-robin methods. The detailed scheduling mechanism will be mentioned later. In order to improve the DRAM locality, the requests will be packed by Unified Reorder Block Controller (Backend), then schedule by Backend Scheduler to allocated suitable time slot and external DRAM banks.

–**DRAM Controller** is the typical DRAM controller to control the actually access of external DRAM banks. Since Backend Scheduler and Unified Reorder Block Controller (Backend) help to schedule and arbitrate the requests of Blocks, the hardware complexity of DRAM Controller can be reduced.

4. The Scheduling Mechanisms of SAMS System

In this section, the scheduling mechanisms of Frontend Scheduler and Backend Scheduler will be discussed, respectively.

4.1. The Scheduling Mechanism of Frontend Scheduler

Modern sophisticated multimedia DRAM controllers usually consist of complex scheduling algorithms to improve the performance of memory access, such as MediaMem [2]. Based on static priority fashion, these algorithms can reduce access latency but not handle the burst bandwidth requirement of multimedia processor, and then the low-priority channels will starvation. Also, the round-robin policy can fairly allocate bandwidth for each channel, but cannot adjust dynamically when the channel requires burst bandwidth. Therefore a dynamically adjustable scheduling mechanism of Frontend Scheduler is proposed to deal the above situations.

The execution steps of Frontend Scheduler are illustrated in Fig. 3. First, for each Access Buffer of the streaming processor, the upper bound bandwidth (Upper_bw) and the lower bound bandwidth is determined for the further scheduling steps, by Frontend Scheduler Attributes Determine module. The corresponding control registers of Frontend Scheduler is set accordingly. During execution, Frontend Scheduler records the real transfer bandwidth (Act_bw) of each Access Buffer. Then scheduler classifies the transfer priority of Access Buffer into three levels, Act_bw, Upper_bw, and Lower_bw, according to their actually transfer requirements. Three priority queues, High_queue, Normal_queue, and Low_queue, are adopted accordingly. After completing the above initialization steps, the scheduler can be executed. The executing flow of scheduler can be divided into two stages, Request Stage and Arbitration Stage.

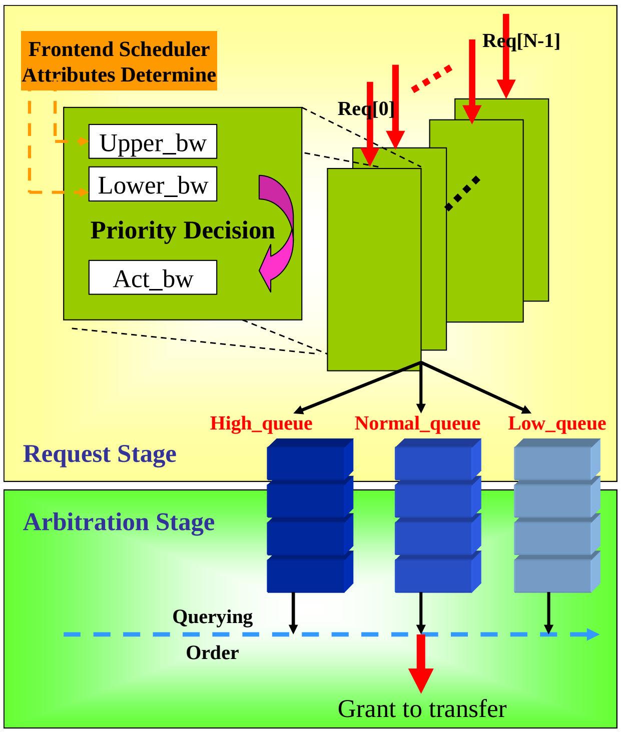

Figure 3 The arbitration flow of Frontend Scheduler.

The scheduling algorithm of Request Stage is listed in Fig. 4. For i'th Access Buffer, it adopts registers of act_bw[i], lower_bw[i], upper_bw[i], and req[i] to records the states of real transfer value, bandwidth upper bound, bandwidth lower bound, and request of the i'th Access Buffer, respectively. According to the states of corresponding act_bw, lower_bw, and upper_bw, Frontend Scheduler periodically checks requests and decides their transfer priorities. Then the request will be put into a feasible priority queue accordingly.

The arbitration algorithm of Arbitration Stage is also listed in Fig. 4. According to the pooling sequence, from High_queue to Low_queue, scheduler checks any request in three priority queues and grant the highest priority one. So the Access Buffer with higher bandwidth requirement can get more bandwidth but not induce the starvation of lower priority Access Buffer. The fairness of the requests in the same Access Buffer can be maintained.

Since the bandwidth requirement of Access Buffer may change over time, the priority and attributes can be determined via Frontend Scheduler Attributes Determine module. This module makes scheduler can not only meet the bandwidth requirement of all Access Buffers of the corresponding streaming processors, when total resource bandwidth is sufficient, but also provide ability to flexibly allot bandwidth focused on important streaming processors dynamically, when total resource bandwidth is inadequate.

Request:

```
for i = 1 to n-1
if ( req[i] == true ) begin
    if ( act_bw[i] <= lower_bw[i] )
        insert request tag i to High_queue
    else if ( act_bw[i] <= upper_bw[i] )
        insert request tag i to Normal_queue
    else if ( act_bw[i] > upper_bw[i] )
        insert request tag i to Low_queue
end
```

Arbitration:

```
if ( anyone waiting in High_queue )
    grant the head waiting tag of High_queue to transfer
else if ( anyone waiting in Normal_queue )
    grant the head waiting tag of Normal_queue to transfer
else if ( anyone waiting in Low_queue )
    grant the head waiting tag of Low_queue to transfer
```

Figure 4 The scheduling algorithms of Frontend Scheduler.

4.2. The Scheduling Mechanism of Backend Scheduler

The main functionality of Backend Scheduler is to monitor and select the suitable Blocks to access multiple External DRAM Banks. Since the transferring requirements of Blocks are varied, the Backend Scheduler has to determine the access order of Blocks and External DRAM Banks to improve the effective DRAM bandwidth by reducing the idle time of External DRAM Banks. The scheduling mechanism is as described below.

- **–Step 1:** The current transferring statistical data of all alive Buffers in the Reorder Block is monitored by Backend Scheduler for the further scheduling stages.
- **–Step 2:** The weights of alive Buffers which send access request are calculated by using the required transferring size, actually transferring size, and assigned priority of the corresponding Access Buffer, which is provided by Frontend Scheduler. The determined weights are sorted and pop the highest ones.
- **–Step 3:** Determine the suitable External DRAM Banks according to the alive Buffers which have the highest weights. If multiple Buffers access the same External DRAM Bank, the fair arbitration mechanism such as Round-Robin policy is adopted to select a suitable Buffer to transfer.
- **–Step 4:** Control the DRAM controllers and apply the memory transfer by using the packing requests which are generated by Unified Reorder Block Controller. If the memory transfer of the Buffer is completed, the Buffer will be recycled for the further usage.
- **–Step 5:** If all of the alive Buffers complete the transfers, the Backend Scheduler will return to Step 1.

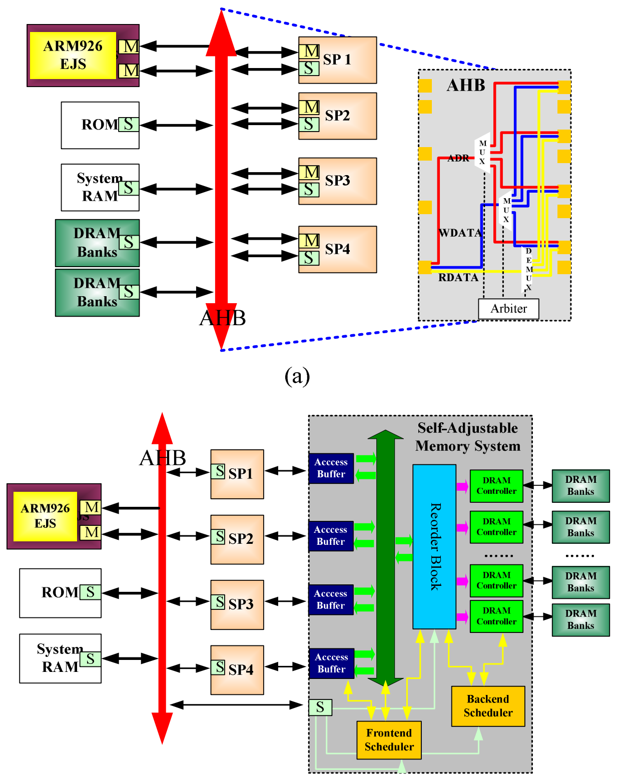

Figure 5 The system organizations of (a) Reference Surveillance System (b) SAMS System that are implemented by CoWare Platform Architect and SystemC HDL.

5. Experimental Results

The real-time surveillance applications and multimedia processing are widely adopted in many application domains [8]. In order to simulate the above application scenarios, the Reference Surveillance System is proposed. The evaluation platforms of the Reference Surveillance System and proposed SAMS System are created by CoWare Platform Architect and SystemC HDL, to implement behaviors of streaming processors and a surveillance system, as illustrated in Fig. 5 (a) and (b), respectively. Both of them are composed by the same ARM 926EJS, AHB bus, External DRAM Banks, ROM, System RAM, and four streaming processors (SP), as shown in Fig. 5. The difference between these two platforms is Direct Memory Access (DMA) Controller (Reference Surveillance System) and SAMS (SAMS System), to exam the performance enhancement of the proposed mechanisms. The proposed four streaming processors, included a MPEG4 Encoder, a MPEG4 Decoder, a MP3 Encoder, and a MP3 Decoder, are configured by several resolutions and bit rates. The MPEG4 Encoder and Decoder can configure by resolutions of 1080p, 720p, 720x480, and 320x200, in 30 frames per second. Meanwhile, the MP3 Encoder and Decoder can configure by the bit rate of 320k.

In the experimental results, ”Actual Requirement” denotes the actually transferring requirement of the SP. ”Reference System” represents the transfer amount of Reference Surveillance System. ”MediaMem” denotes the experimental results from our previous MediaMem [2] system to demonstrate the characteristics of static-priority memory access scheduling. ”SAMS (2 DRAM Banks)”, ”SAMS (4 DRAM Banks)”, ”SAMS (8 DRAM Banks)”, and ”SAMS (16 DRAM Banks)” denote the proposed SAMS systems whose DRAM Banks are consisted of 2, 4, 8, 16 External DRAM Banks, respectively.

The following experiments are configured as four models.

Configuration #1, as shown in Fig. 6, adopts four SPs by using MPEG4 Encoder (1080p), MPEG4 Decoder (1080p), MPEG4 Decoder (1080p), and MP3 Decoder (320k). The gray bars show the actual requirements. The SP of MPEG4 Encoder 1080p consumes largest amount of total transfer that requires near 10X than the sum of the transferring amounts of other three SPs. The yellow bar represents the scheduling results of Reference Surveillance System by using DMA controller and fair bus arbitration. Due to the burst transferring requirement of MPEG4 Encoder 1080p, the transferring amount of yellow bar is less than required. In contrast, the blue bar denotes the transferring number of MediaMem [2] system by using static-priority scheduling and simple two-level bus arbitration. Although the above mechanisms can prevent the starvations of other three SPs, it still can not deal with the burst transferring requirement of MPEG4 Encoder 1080p. These problems can be solved by SAMS (2 DRAM Banks), which is represented by orange bar. The external DRAM banks of gray yellow, blue, and orange bars are dual banks, but SAMS (2 DRAM Banks) can dynamically arrange more bandwidth to cover the requirement of burst transferring. Its novel scheduling policy still can achieve the requirements of other three SPs. The cyan bar denotes that SAMS system is constructed by quad External DRAM Banks. Also, the pink and purple bars denoted the SAMS (8 DRAM Banks) and SAMS (16 DRAM Banks), respectively. The proposed SAMS system can easily to extend the channel amount to enlarge the total affordable transferring bandwidth without modifying the bus architecture and memory controller. According to the cyan, pink, and purple bars in Fig. 6, the external DRAM banks of SAMS that are configured as 4 banks, 8 banks, and 16 banks, can achieve near 1.8X, 2.6X, 2.8X times transferring amount than dual banks. It also demonstrates the scalability of SAMS system.

The second experimental setting, Configuration #2, acquires MPEG4 Encoder (1080p), MPEG4 Encoder (720p), MPEG4 Decoder (1080p), and MP3 Decoder (320k), as shown in Fig. 7. Due to total required transferring amount is larger than Fig. 6, the insufficient transferring amount of MPEG4 Encoder (1080p) in Reference Surveillance System is increased due to the fair arbitration can not handle full-loaded burst transferring requirements. The insufficient bandwidth of MediaMem is larger then Configuration #1 due to the 1.5X bandwidth requirements. Since MPEG Encoder (720P) consumes more bandwidth, the MediaMem and SAMS (2 DRAM banks) can not fulfill the bandwidth requirements. However, the SAMS (4 DRAM banks), SAMS (8 DRAM banks), and SAMS (16 DRAM banks) can completely fulfill the requirements. The transfer amounts of SAMS (4 DRAM banks), SAMS (8 DRAM banks), and SAMS (16 DRAM banks) are 2X, 3.6X, and 3.9X larger than SAMS (2 DRAM banks), respectively.

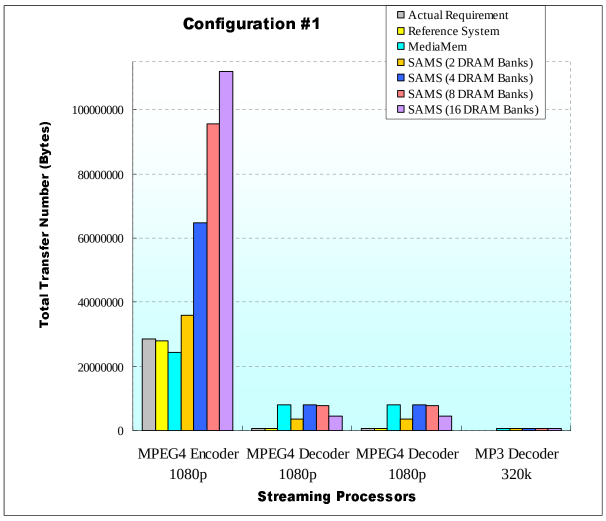

Figure 6 Performance comparison of Configuration #1.

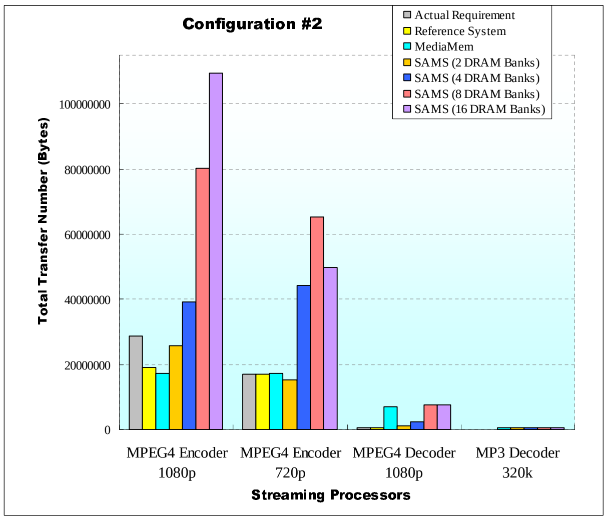

Figure 7 Performance comparison of Configuration #2.

The third experimental model, Configuration #3 acquires MPEG4 Encoder (1080p), MPEG4 Encoder (720x480), MPEG4 Decoder (1080p), and MP3 Decoder (320k), as shown in Fig. 8. Since the total required transferring amount is less than Configuration #2, the insufficient transferring amount of MPEG4 Encoder (1080p) in Reference Surveillance System is similar but the transferring amount of MPEG4 Encoder (720x480) is enough, due to the side-effect of fair arbitration. The MediaMem and SAMS (2 DRAM banks) still can not fulfill the bandwidth requirements. But SAMS (4 DRAM banks), SAMS (8 DRAM banks), and SAMS (16 DRAM banks) can completely fulfill the requirements. Due to its fewer total transferring amount, the transfer amounts of SAMS (4 DRAM banks), SAMS (8 DRAM banks), and SAMS (16 DRAM banks) are 1.9X, 3.2X, and 3.5X larger than SAMS (2 DRAM banks), respectively.

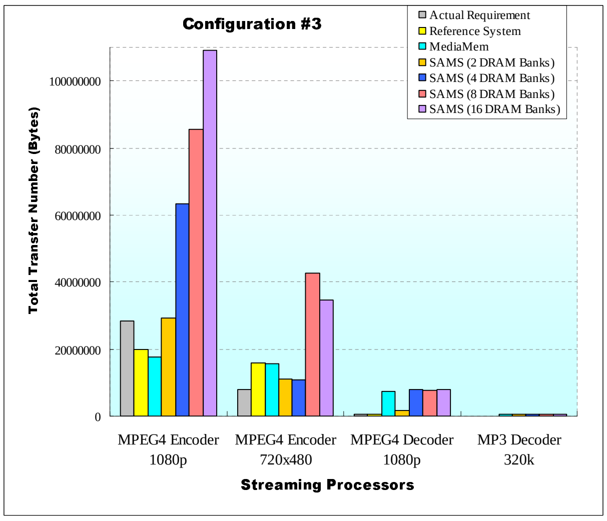

Figure 8 Performance comparison of Configuration #3.

The fourth experimental, Configuration #4 acquires MPEG4 Encoder (1080p), MPEG4 Encoder (320x200), MPEG4 Decoder (1080p), and MP3 Decoder (320k), as shown in Fig. 9. Because the total required transferring amount is less than Configuration #3, the insufficient transferring amount of MPEG4 Encoder (1080p) in Reference Surveillance System is similar but the transferring amount of MPEG4 Encoder (320x200) is enough. Although MediaMem can not fulfill the bandwidth requirements, the transferring amount of SAMS (2 DRAM banks) is enough, due to its self-adjust scheduling mechanism. Accordingly, SAMS (4 DRAM banks), SAMS (8 DRAM banks), and SAMS (16 DRAM banks) can completely fulfill the requirements and obtain 1.9X, 2.8X, and 3.2X speedup than SAMS (2 DRAM banks), respectively.

The proposed SAMS system is implemented by Bluespec SystemVerilog and then synthesized by Synopsys Design Compiler and TSMC 90 nm cell library. The working

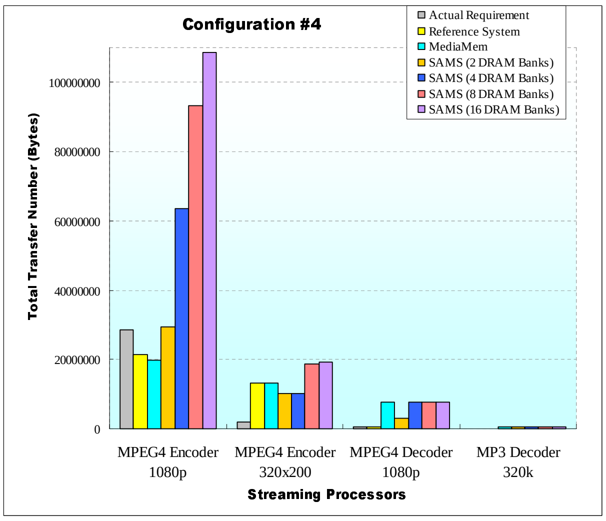

Figure 9 Performance comparison of Configuration #4.

frequency can achieve 403 MHz and consume 6679991μm^2. It confirms that SAMS system doesn't delay the overall performance of the conventional streaming SOC chip. The gate-level netlist of the SAMS system is implemented by Magma Talus and generated the layout, as shown in Fig. 10.

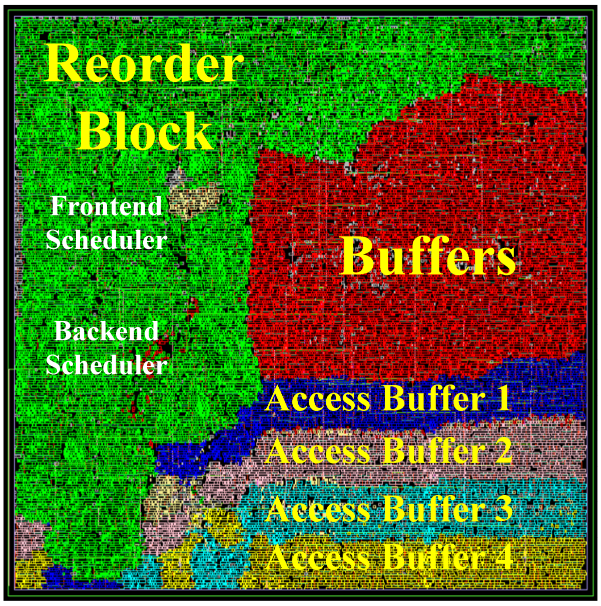

Figure 10 Layout of SAMS system.

6. Conclusions

In this paper, a novel memory, Self-Adjustable Memory System (SAMS) is proposed to overcome the performance bottleneck of modern surveillance system, by integrating Access Buffer, Frontend Scheduler, Reorder Block, and Backend Scheduler into a single memory system. The detailed mechanisms of above modules have mentioned. The proposed SAMS architecture has been simulated by CoWare Platform Architect and SystemC HDL. The experimental results illustrate that SAMS can arrange enough bandwidth for the streaming processors that have bursting transferring requirement, and can manage multiple External DRAM Banks. The fabrication results present that SAMS can achieve 403 MHz under TSMC 90 nm cell library and consume 6679991μm^2. The comprehensive whole system simulation show that proposed SAMS can utilize total memory bandwidth efficiently, better than DMA-based reference surveillance system and our previous MediaMem system, by using novel self-adjustable mechanisms of Frontend Scheduler, Reorder Block, and Backend Scheduler.

Acknowledgement

This work is supported in part by the National Science Council of Republic of China, Taiwan under Grant NSC 101-2221-E-033-049.

References

[1] G. Kornaros, I. Papaefstathiou, A. Nikologiannis, and N. Zervos, Proc. 40th Conference on Design Automation, 54-59 (2003).

[2] S. L. Chu, M. J. Lo, and H. W. Yang, Proc. 13th Asia-Pacific Computer Systems Architecture Conference, 1-8 (2008).

[3] C. H. Chen, G. W. Lee, J. D. Huang, and J. Y. Jou, Proc. 11th Asia and South Pacific Design Automation Conference, 600-605 (2006).

[4] C. A. Waldspurger and W.E. Weihl, Proc. Symposim on Operating System Design and Implementation, 1-11 (1994).

[5] Faraday Technology Corporation. http://www.faraday-tech.com/html/ solutions/8180.html.

[6] Mobilygen Corporation. MG3500 Datasheet. http://www.maxim-ic.com/ .

[7] Texas Instruments Inc. http://www.ti.com/.

[8] P. Pesout, and O. Matustik, Applied Mathematics & Information Sciences, **6**, 437-440 (2012).

Slo-Li Chu received his PhD degree in Electrical Engineering from National Sun Yat-sen University in 2002. He is currently an assistant professor of Department of Information and Computer Engineering, Chung Yuan Christian University, Taiwan. His research interests include computer architectures, parallelizing compilers, system level modeling, system-on-chip design, GPU architectures, and embedded system.

Min-Jen Lo received his BS degree in Information and Computer Engineering from Chung Yuan Christian University, Taiwan, in 2005. He is currently pursuing for his PhD degree in Electronic Engineering at Chung Yuan Christian University, Taiwan. His research interests include computer architectures, system level modeling, and system-on-chip design.