Slo-Li CHU, Min-Jen LO

Chung Yuan Christian University

Design a High-Performance Memory Controller for a Multimedia SOC

Abstract. *Continuously growing functionalities of modern consuming electronics make the major multimedia SOC (system-on-a-chip) chip more complex. Moreover, the integrated multimedia processors and the required memory bandwidth are increasing. Therein how to improve the performance of the memory controller will become a major challenge of designing a modern multimedia SOC. According to our previous study of multimedia SOC, to achieving the bandwidth requirements are not only by improving memory throughput but also by dynamically adjusting the bandwidth usage of multimedia processors. Therefore we develop novel memory subsystem, called Smart Memory Controller (SMC), which integrates a novel scheduling/arbitration mechanism, a unified access buffer, multi-level memory access classification/scheduling, and several corresponding hardware modules, to provide a sufficient memory bandwidth for the multimedia processors with high bandwidth requirements. The proposed SMC architecture has been implemented by SystemC/Bluespec/Verilog HDL. The experimental results from whole SMC system illustrates that SMC will arrange enough bandwidth for the channels that have bursting transferring requirement. The fabrication results of SMC are also provided.*

Streszczenie. *W artykule zaproponowano nowy system pamięci nazwany SMC (smart memory controller) przeznaczony do multimedialnych elementów typu SOC (system on a chip). System integruje mechanizm planowania i arbitrażu (scheduling/arbitration), bufor dostępu, wielopoziomowy dostęp do pamięci i wiele innych modułów. (**Projekt kontrolera pamięci o dużej wydajności w przeznaczeniu do multimedialnych elementów SOC**)*

Keywords: Memory Access Scheduling, Multimedia System-on-a-Chip, Interconnection Network, SystemC.
Słowa kluczowe: SOC – system on e chip, kontroler pamięci

Introduction

Due to the continuous improvement of semiconductor technique, the functionalities of modern consuming electronic products are increases, especially in multimedia streaming processing, such as MP3 decoding, MPEG4 encoding/decoding, and voice recording. In order to achieve these requirements, the system-on-chip (SoC) solution is usually adopted, which integrates multiple high performance multimedia processors (MP) in a single chip. These MPs access the system memory via shared on-chip bus. Even MP has the DMA (Direct Memory Access) capability to access memory individually, only one MP is granted to access memory at one time. While multiple multimedia processors process high volume streaming data, it is easily to cause memory contentions and can not afford enough bandwidths for these MPs. Consequently, the access capability of the memory subsystem is an important factor to affect the performance of whole system.

Therefore a novel memory subsystem, called Smart Memory Controller (SMC), is proposed in this paper, to improve the memory accessing capability while encountering bursting transferring requirement. The whole SMC architecture has been designed by SystemC/Bluespec/Verilog HDL in detail. We also build two SoC systems, SMC Platform and Basic Platform, to perform whole system functional verification and performance evaluation. According to the experimental results, the proposed SMC architecture is a feasible for the high-bandwidth required multimedia environment.

The rest of this paper is organized as follows. Section 2 briefly discusses related works of memory access optimization. Section 3 presents the detailed architecture and execution flow of proposed SMC architecture. Section 4 shows the experimental results of SMC system. Finally, the concluding remark is proposed in Section 5.

Related Works

Vlachos et. al. propose a Programmable Protocol Processor (PRO3) system architecture [3], perform in networking environment and aims in accelerating execution of telecom protocol. In order to provide the large requirement of memory bandwidth, Kornaros et. al. propose a Data Memory Management subsystem(DMM) [1]. The internal scheduler forwards the incoming command from the four ports to the data queue manager giving priority to each port. The data memory controller performs the low level read from data memory and writes to data memory. Then reassembly block organizes the segmented packages and output them. This mechanism is designed for network processor, and its access request and data transfer are regular. Therefore it is not suitable for irregularly access fashions of multimedia processors.

Sonics Inc. proposes a memory controller, MemMax [4], to enhance the efficient of DRAM access for multimedia SOC. It includes a scheduler and a point-to-point interconnection network. When DMA and MPs request to access memory at the same time, the scheduler can arbitrate and grant a MP to access DRAM. The scheduler also can combine the suitable requests to reduce the overhead of single memory transfer. Integrated with proprietary μNetwork and scheduler, the MemMax system can provide good memory bandwidth for multiple MPs. However, it majorly considers the DRAM side scheduling and can not dynamically adjust the channel bandwidth according to MP's requirement. These drawbacks limit the performance of multimedia SOC system.

Nieh and Lam propose SMART, a scheduler for multimedia and real-Time application [2]. This scheduling algorithm decides the priority by a "value-tuple", which is a tuple with two components: priority and the biased virtual finish time (BVFT).Priority is a static quantity either assigned by the user or assigned the default value; If task A has a higher static priority or if both task A and task B has the same priority and task A has an earlier BVFT, task A has a higher value-tuple than task B. This arbitration mechanism is designed for their proprietary SOC but not suitable for generic multimedia SOC system, so we need to redesign the feasible scheduling mechanism.

The Architecture of Smart Memory Controller

In this section, we will discuss the detailed architecture of Smart Memory Controller (SMC) in former. Then the arbitration mechanism and primitive operations of SMC are discussed later, includes Read and Write operations are introduced in later.

- **The components of SMC system**

The proposed SMC architecture, as shown in Fig. 1, consists of R/W Channels, Interconnection Network, Channel Scheduler, SRAM Scheduler, SRAM System, and DRAM System. The Multimedia Processor (MP), External DRAM, and CPU are attached externally.

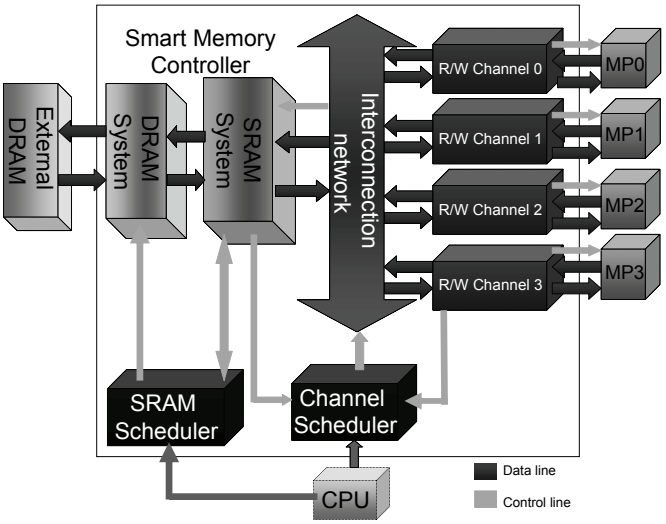


Fig.1. The architecture of proposed SMC System.

R/W Channel, as shown in Fig. 2, is the connection interface between SMC and an external MP, which includes four FIFOs and corresponding control registers, can temporarily buffering the data and notify Channel Scheduler for further arbitration and scheduling.

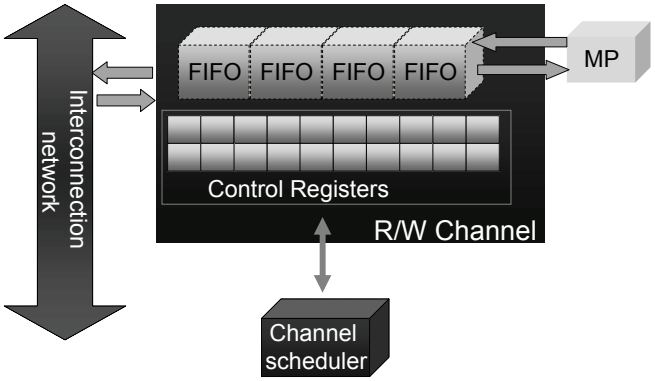


Fig.2. The architecture of R/W Channel.

Channel Scheduler arbitrates all simultaneously requests of R/W Channels, then decides one R/W Channel to get grant to transfer data via Interconnection Network. The organization of Channel Scheduler with attached devices is as shown in Fig. 3. It contains three priority queues to store R/W Channel ID number which sends request. Each channel has its dynamic weight. Then the corresponding R/W Channel ID is pushed into a specified priority queue according to its weight for further scheduling. Current states of Channel Scheduler are also updated to SRAM System, and retrieve status from SRAM System.

SRAM System is composed by four SRAM Partitions, SRAM Controller, Address Generator, and SRAM Divider, as presented in Fig. 4. SRAM Partitions, used to store the transferring data, are logically divided from unified SRAM buffer. It can improve the performance of data transferring by data locality. Address Generator generates the absolute DRAM addresses for the requests of R/W Channels. SRAM Controller, managed by SRAM Scheduler, is used to control the data transfer. SRAM Divider partitions logical address and keeps the addressing boundaries of each SRAM Partition.

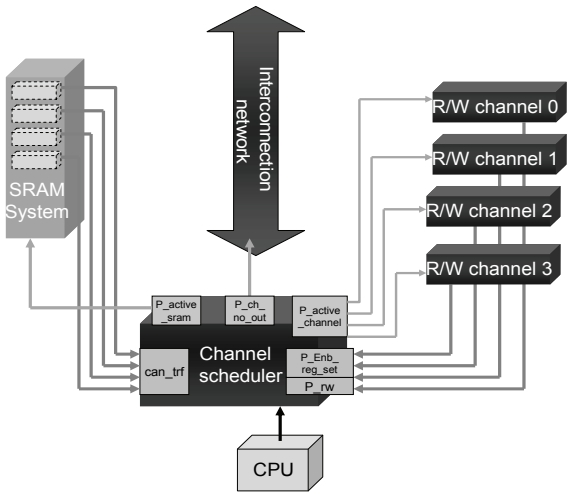


Fig.3. The organization of Channel Scheduler with attached devices.

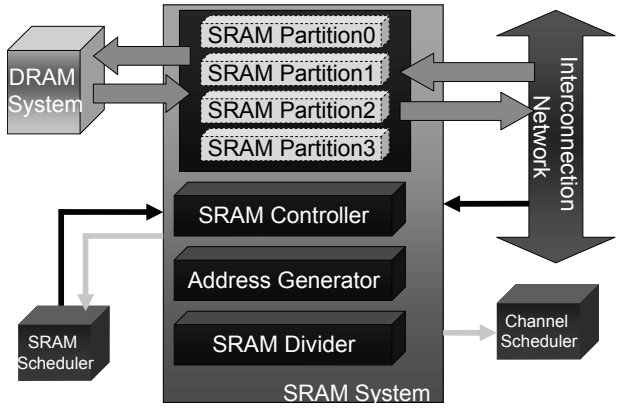


Fig.4. The architecture of SRAM System.

SRAM Scheduler is responsible for scheduling and arbitrating one SRAM Partition to access to the off-chip DRAM. It also controls DRAM System to transfer data to/from off-chip DRAM according to the logical address. The organization of SRAM Scheduler and attached devices are proposed in Fig. 5.

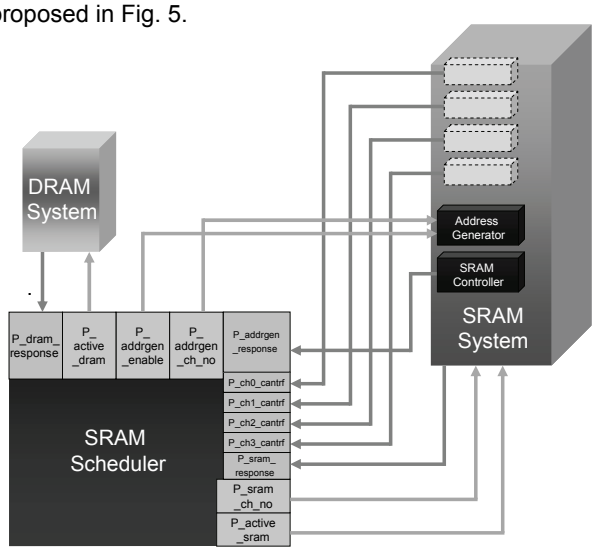


Fig.5. The organization of SRAM Scheduler with attached devices.

- **The arbitration mechanism of Channel Scheduler**

In order to arbitrate the requests of R/W Channels for better utilization and fairness, a novel arbitration mechanism is required for Channel Scheduler. The proposed arbitration algorithm for Channel Scheduler is as listed in Fig. 6. At first, Channel Scheduler creates three priority queues: High_queue, Normal_queue, and Low_queue. The scheduling order of these three priority queues are High_queue, Normal_queue, and Low_queue, respectively. The attached R/W Channels can obtain their corresponding weights according to their actual requirement. The lower_bw and the upper_bw represent the lowest bandwidth and the highest bandwidth which the bus can provide. The actual_bw is the actual bandwidth which the corresponding R/W Channel needs. All of the transferring requests of R/W Channels can be arbitrated according to their actual requirements, assigned priorities, and memory bandwidth.

```
// Arbitration Algorithm

//Ti: transferring request from R/W Channels.
// Priority(Ti): the assigned priority of the request Ti

if (transferring request coming) //initial
  push request to " Normal_queue";

sort  "High_queue, Normal_queue, Low_queue" in order;
find top Ti;

if(time_slice != 0)
    Pi is granted to access memory;
else
{
    if ((Priority(Ti)=="High_queue")&&(actual_bw<upper_bw))
        Priority(Ti)="High_queue" ;
    else if ((Priority(Ti)=="High_queue")&&(actual_bw>upper_bw))
        Priority(Ti)="Low_queue" ;
    else if ((Priority(Ti)=="Low_queue")&&(actual_bw<upper_bw))
        Priority(Ti)="High_queue" ;
    else if ((Priority(Ti)=="Low_queue")&&(actual_bw>upper_bw))
        Priority(Ti)="Low_queue" ;
        Ti suspend;
    else if ((Priority(Ti)=="Normal_queue")&&(actual_bw > upper_bw))
        Priority(Ti)="Low_queue" ;
    else if ((Priority(Ti)=="Normal_queue")&&(actual_bw < lower_bw))
        Priority(Ti)="High_queue" ;
    else
        Priority(Ti)="Normal_queue" ;
}
```

Fig. 6. The scheduling algorithm of Channel Scheduler.

- **The Primitive Operations of SMC System**

In this subsection, the execution flows of two primitive operations, Read and Write of SMC system, are introduced.

Read operation:

Read operation denotes that MP requests to read data from external DRAM. MP firstly gets the required data from R/W Channel if the required data is in the R/W Channel. Otherwise, R/W Channels will send the requests to Channel Scheduler for read data from the corresponding SRAM Partition. When multiple R/W Channels try to access SRAM Partitions concurrently, Channel Scheduler grants one R/W Channel to access SRAM Partition according to the proposed scheduling mechanism. If the required data is in the SRAM Partition, R/W Channel receive data directly from the SRAM Partition.

If this SRAM Partition doesn't contain the required data, SRAM Partition will request to access external DRAM via DRAM System. Then SRAM Scheduler will schedule all of the requested SRAM Partition. Once this SRAM Partition gets grant, DRAM System will transfer data from external DRAM to this SRAM Partition according to the logical address.

Write operation:

Write operation means MP write data to external DRAM. At beginning, if any free space is in the R/W Channel, the attached MP transfers data to the R/W Channels directly. Otherwise, if the space of R/W Channel is full, R/W Channel will send the requests to Channel Scheduler for transferring data to the corresponding SRAM Partition. When multiple R/W Channels try to access SRAM Partition concurrently, Channel Scheduler grants one R/W Channel to write data into a SRAM Partition.

When the SRAM Partition is full, SRAM Partition will request to access external DRAM through DRAM System. Then SRAM Scheduler will arbitrate all of the SRAM Partitions that requests to access at the same time. When one SRAM Partition gets grant, DRAM System will transfer data from SRAM Partition to external DRAM according to the logical address

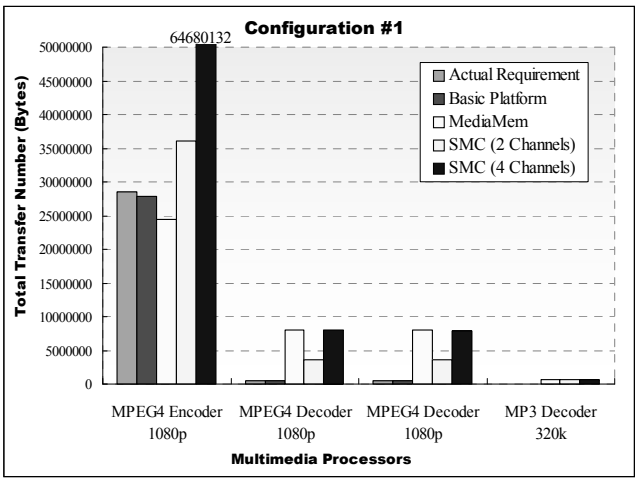
Experimental Result

The evaluation platforms of the generic Basic Platform and proposed SMC System are created by CoWare Platform Architect and SystemC HDL, to implements functionalities of multimedia processors and SOC systems, respectively. Both of them are consisted of the same ARM 926EJS, AHB bus, Dual-channel External DRAM, ROM, System RAM, and four multimedia processors/MPs. The difference between these two platforms is Direct Memory Access (DMA) Controller (Basic Platform) and SMC Module (SMC System), to exam the performance enhancement of the proposed mechanisms. The proposed four multimedia processors (MPs), 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.

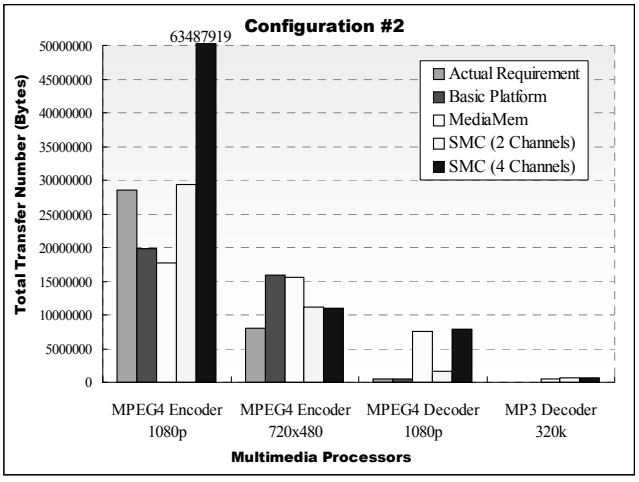
The experimental results are shown in Fig. 7. "Actual Requirement" denotes the actually transferring requirement of the MP. "Basic Platform" represents the transfer amount of Basic Platform. "MediaMem" denotes the experimental results from our previous MediaMem [5] system to demonstrate the characteristics of static-priority memory access scheduling. "SMC (2 Channels)" and "SMC (4 Channels)" denote the proposed SMC systems whose external DRAM systems are consisted of dual and quad channels, respectively.

The following experiments are configured as two models. "Configuration #1" adopts four MPs by using MPEG4 Encoder (1080p), MPEG4 Decoder (1080p), MPEG4 Decoder (1080p), and MP3 Decoder (320k). The results are shown in Fig. 7 (a). The heavy-gray bars show the actual requirements of four MPs. The MP of MPEG4 Encoder 1080p consumes largest amount of total transfer that requires near thirty times than the sum of the transferring amounts of other three MPs. The dark-gray bar represents the scheduling results of Basic Platform by using DMA controller and fair bus arbitration. Due to the burst transferring requirement of MPEG4 Encoder 1080p, the transferring amount of dark-gray bar is less than required. In contrast, the medium-gray bar denotes the transferring number of MediaMem [5] system by using static-priority scheduling and simple two-level bus arbitration. Although the above mechanisms can prevent the starvations of other three MPs, it still can not deal with the burst transferring requirement of MPEG4 Encoder 1080p. These problems can be solved by SMC (2 Channels), represented by light-gray bar. The external DRAM systems of heavy-gray, dark-gray, medium-gray, and light-gray bars are dual channels,

but SMC (2 Channels) can dynamically arrange more bandwidth to cover the requirement of burst transferring. Its novel scheduling policy still can achieve the requirements of other three MPs. The cyan bar denotes that SMC system is constructed by quad external DRAM channels. The proposed SMC 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 bars in Fig. 7 (b), the external DRAM of SMC that are configured as quad channels achieves near two times transferring amount than dual channels. It also illustrates the scalability of SMC system.



(a)



(b)

Fig. 7. Performance comparisons of two set of system configurations, (a) Configuration #1 (b) Configuration #2, for Basic Platform, MediaMem, and two SMC systems.

The second experimental setting, "Configuration #2" acquires MPEG4 Encoder (1080p), MPEG4 Encoder (720x480), MPEG4 Decoder (1080p), and MP3 Decoder (320k), as shown in Fig. 7 (b). Due to total required transferring amount (four heavy-gray bars) is larger than Fig. 7 (a), the insufficient transferring amount of MPEG4 Encoder (1080p) in Basic Platform (dark-gray bar) will be enlarged due to the fair arbitration can not handle full-loaded burst transferring requirements. The insufficient bandwidth is more serious in MediaMem due to static-priority scheduling mechanisms arranges more bandwidth, from MPEG4 Encoder (1080p) to MPEG4 Encoder (720x480), to fulfil the ratio of relative priority of these two MPs. Hence the dynamically adjustable bandwidth requirements can deal with this problem. It adjusts the transferring amount according to real situations. So the wasted bandwidth from MPEG4 Encoder (720x480) can help to satisfy the bandwidth requirements of MPEG4 Encoder (1080p) by using SMC (2 Channels) and SMC (4 Channels).

The proposed SMC system is designed by Bluespec and Verilog HDL, and then synthesized by Synopsys Design Compiler and TSMC 0.13 μm cell library. The working frequency can achieve 568 MHz and consume 564933μm2. It confirms that SMC system doesn't delay the overall performance of the multimedia SOC chip. The gate-level netlist of the SMC system is implemented by Magma Talus and generated the layout.

Conclusions

In this paper, we develop novel memory subsystem, called Smart Memory Controller (SMC), which integrates a novel scheduling/arbitration mechanism, a unified access buffer, and a multiple channel memory, to provide a sufficient memory bandwidth for the multimedia processors with high bandwidth requirements. The proposed SMC architecture has been implemented by SystemC/Bluespec/Verilog HDL. The experimental results from whole SMC system illustrates that SMC can arrange enough bandwidth for the channels that have bursting transferring requirement. The fabrication results present that SMC can achieve 568 MHz under TSMC 0.13 μm cell library and consumes 564933μm^2. The comprehensive whole system simulation show that proposed SMC can utilize total memory bandwidth efficiently, better than DMA-based Basic Platform and our previous MediaMem system, by using novel dynamic-priority Channel Scheduler and SRAM Scheduler.

Acknowledgments. This work is supported in part by the National Science Council of Republic of China, Taiwan under Grant NSC 100-2221-E-033-043.

REFERENCES

[1] Kornaros G., Papaefstathiou I., Nikologiannis A., and Zervos N, A Fully-Programmable Memory Management System Optimization Queue Handling at Multi Gigabit Rate. In: 40th Conference on Design Automation, 2003.

[2] Nieh J., and Lam M. S., The Design, Implementation and Evaluation of SMART: A Scheduler for Multimedia Applications. In: ACM symposium on Operating Systems Principles, 1997.

[3] Vlachos K., N., Nikolaou T. Orphanoudakis, S. Perissakis, Pnevmatikatos D., Kornaros G., Sanchez J. A., and Konstantoulakis G., Processing and Scheduling Components in an Innovative Network Processor Architecture. In: 16th International Conference on VLSI Design, 2003.

[4] Sonics Ltd., MemMax Memory Controller. http://www.sonicsinc.com/.

[5] Chu S. L., Lo M. J., and Yang H. W., MediaMem: A Dynamically Adjustable Memory Subsystem for High-bandwidth Required Multimedia SoC Systems. In: 13th Asia-Pacific Computer Systems Architecture Conference, 2008.

***Authors**: Prof. Slo-Li Chu and Mr. Min-Jen Lo are with Department of Information and Computer Engineering, Chung Yuan Christian University, 200, Chung Pei Rd., Chung Li, 32023 Taiwan. E-Mail: slchu@cycu.edu.tw and g9477010@cycu.edu.tw.*