Dual-Port SDRAM Optimization with Semaphore Authority Management Controller

Jaehwan Kim and Jongwha Chong

This paper proposes the semaphore authority management (SAM) controller to optimize the dual-port SDRAM (DPSDRAM) in the mobile multimedia systems. Recently, the DPSDRAM with a shared bank enabling the exchange of data between two processors at high speed has been developed for mobile multimedia systems based on dual-processors. However, the latency of DPSDRAM caused by the semaphore for preventing the access contention at the shared bank slows down the data transfer rate and reduces the memory bandwidth. The methodology of SAM increases the data transfer rate by minimizing the semaphore latency. The SAM prevents the latency of reading the semaphore register of DPSDRAM, and reduces the latency of waiting for the authority of the shared bank to be changed. It also reduces the number of authority requests and the number of times authority changes. The experimental results using a 1 Gb DPSDRAM (OneDRAM) with the SAM controllers at 66 MHz show 1.6 times improvement of the data transfer rate between two processors compared with the traditional controller. In addition, the SAM shows bandwidth enhancement of up to 38% for port A and 31% for port B compared with the traditional controller.

Keywords: Dual-port SDRAM, memory controller, semaphore control, dual-processor.

Manuscript received May 6, 2009; revised Oct. 22, 2009; accepted Nov. 5, 2009.

This work was supported by the R&D program of Mtekvision Co., Ltd, Rep. of Korea, ETRI SoC Industry Development Center, Human Resource Development Project for IT SoC Architect and Seoul R&BD Program (10560). This work was also supported by the MKE, Rep. of Korea, under the ITRC support program supervised by the IITA (IITA-2009-C1090-0902-0019).

Jaehwan Kim (phone: +82 2 2220 0558, email: corekjh@hanyang.ac.kr) and Jongwha Chong (email: jchong@hanyang.ac.kr) are with the Department of Electronics and Computer Engineering, Hanyang University, Seoul, Rep. of Korea.

doi:10.4218/etrij.10.0109.0262

I. Introduction

Since the late 1990s, the markets for mobile multimedia systems such as cellular phones, personal digital assistants, portable multimedia players, and digital cameras have grown rapidly [1]. Nowadays, users expect mobile multimedia systems to support powerful functions like audio, video, 3D games, and even DMB. Mobile multimedia systems need to perform complicated operations and transmit data at high speed.

Up to now, mobile multimedia systems based on dual-processors have been used. They consist of a baseband processor, which performs the modem functions, and an application processor, which performs multimedia DSP functions. Figure 1(a) shows the traditional architecture of cellular systems. The baseband processor and application processor, which have local memory, use a directly connected dual-port SRAM (DPSRAM) bus to transfer data between them. When data is transferred from outside the cellular system to the baseband processor, it is saved in the local memory of the baseband processor. Next, the baseband processor reads the data from the local memory and saves it in an on-chip memory, such as a buffer or cache. Then, the baseband processor transfers the data saved in the internal memory to the application processor by using the DPSRAM bus. The procedures of exchanging data between two processors take a long time. In this architecture, it is impossible to transfer data at high speed while the system performs the application program.

To solve this problem, a new memory device, the DPSDRAM was introduced. Samsung Electronics announced the DPSDRAM called OneDRAM [2]. Because one DPSDRAM can be used as two local memories for two processors, and its shared bank can be used to transfer data between the

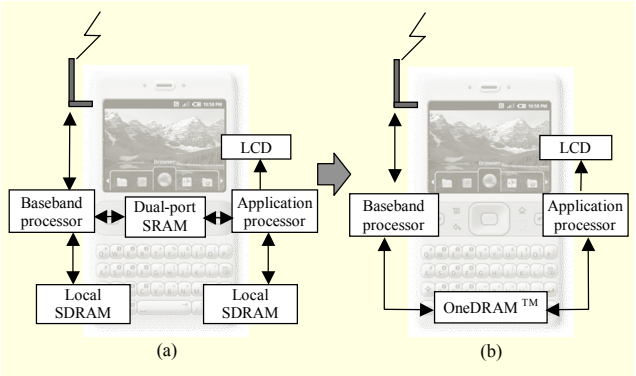

Fig. 1. Simplified architecture of cellular systems: (a) without DPSDRAM and (b) with DPSDRAM (OneDRAM).

processors, the new system based on the DPSDRAM shows performance enhancement of 5 times [2]. Samsung Electronics already announced a product based on OneDRAM. Figure 1(b) shows the architecture of the system using OneDRAM. The baseband processor and the application processor use the OneDRAM as a shared memory instead of using the individual local memory. The DPSRAM bus is unnecessary because the OneDRAM can be used as the communication channel between two processors.

When the DPSDRAM is used in mobile multimedia systems, mutual exclusion [3], [4] is needed to prevent access contention. Access contention occurs when two different processors simultaneously access the shared bank of the DPSDRAM. The OneDRAM uses semaphore mutual exclusion [4] to prevent access contention. However the semaphore causes memory access latency. Because this latency is repetitive, it is a fatal bottleneck in system performance. The semaphore latency problem must be solved to make transferring data at high speed possible in mobile multimedia systems based on the DPSDRAM. In [5], Yang and others proposed algorithms to reduce semaphore latency. They concentrated on the semaphore methods; however, a new DPSDRAM architecture is needed to use their algorithms.

This study proposes a semaphore authority management (SAM) controller to solve the latency problems which occur when the shared bank of the DPSDRAM is accessed. The SAM minimizes the semaphore latency by using three algorithms. The first algorithm, named *duplication of semaphore register*, prevents the latency of reading the semaphore register in DPSDRAM because the authority of the shared bank can be checked with the duplication of semaphore register algorithm in the SAM controller. The second algorithm, named *adaptive command prefetching*, requests the authority of the shared bank before executing read/write commands. It reduces the latency of waiting for the authority to be changed. The last algorithm, named *auto-release of authority*, reduces the numbers of requests and changes of authority of the shared bank. It does this by making one processor release authority to the other processor automatically. The reconfigurable architecture of SAM which is a combination of these three algorithms increases the bidirectional communication speed.

To evaluate the performance of the SAM controller, we used the Verilog simulation model of Samsung Electronics' 1 Gb OneDRAM. The results of experiments with the SAM controllers at 66 MHz show 1.6 times improvement of the data transfer rate between two processors compared with the traditional controller. In addition, the SAM shows bandwidth enhancement of up to 38% for SDR SDRAM (port A) and 31% for DDR SDRAM (port B) compared with the traditional controller.

The rest of this paper is organized as follows. Section II analyzes the semaphore latency of DPSDRAM. Section III describes the three algorithms of the SAM controller. Section IV evaluates SAM under realistic circumstances with the DPSDRAM. Finally, section V concludes the paper.

II. Analyses of Semaphore Latency

The DPSDRAM can be accessed by two independent ports. The DPSDRAM is composed of the dedicated banks and a shared bank. Each port can access to its own dedicated banks, and the shared bank can be accessed by both ports. Figure 2 describes the simplified internal architecture of the DPSDRAM. In the dedicated banks, access contention cannot occur, because two ports cannot access them at the same time. Access contention occurs in the shared bank because two ports can access it at the same time. To prevent access contention, semaphore mutual exclusion is needed. For example, OneDRAM has a semaphore register and two mailbox registers in the shared bank. These are shown in Fig. 2. A 1-bit semaphore register is assigned as the authority of shared bank. If its value is 0, port A can access the shared bank, but if its value is 1, then port B can access the shared bank. Changing authority is only possible if the port with the authority of the shared bank releases the authority by writing a semaphore register. During boot-up of the SDRAM, the authority of the

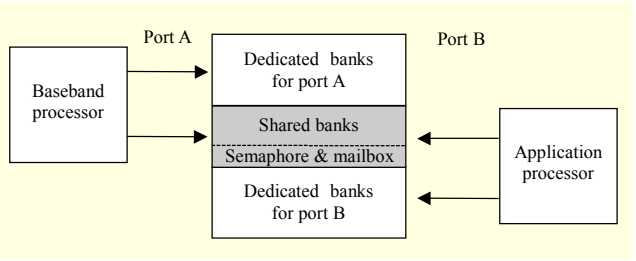

Fig. 2. Simplified architecture of the DPSDRAM.

shared bank is given to the assigned port by default.

By using 32-bit register called a mailbox, the port without the authority from the shared bank can request the authority to the port with authority. A mailbox can be used for transferring short messages or commands. The location and the size of data saved in the shared bank which should be transferred to the other port can be sent as well as messages requesting authority. Port A has the permission to write, and port B has the permission to read the contents of the mailbox (A→B), and the opposite is true for mailbox (B→A). If one port writes data in a mailbox, an interrupt signal occurs at the other port, and it is not changed until the port reads the mailbox contents.

The semaphore is a solution to prevent access contention. However, the semaphore causes the latency of reading the semaphore register, requesting authority, and waiting for the authority to be changed. Semaphore latency makes the system unable to transfer data at high speed. Figure 3 shows the simplified process by which port A accesses the shared bank when port B has the authority of the shared bank. Before accessing the shared bank, port A reads the semaphore register and checks the authority of the shared bank (①). Because port B has the authority, port A writes a message requesting authority in the mailbox (②). As a result, an interrupt signal occurs at port B, and port B perceives the interrupt signal and reads the mailbox contents (③). Then, port B writes in the semaphore register to change the authority of the shared bank (④). Port A waits until the authority is released by port B. Actually, since a port without authority does not know the time when the other port releases the authority, it needs to read the semaphore register continuously. Finally, port A can use the shared bank of the DPSDRAM after reading the semaphore register (⑤). Because each operation shown in Fig. 3 is composed of several commands, the total semaphore latency for using the shared bank is not negligible. The RTL simulation with the DPSDRAM shows that the semaphore spends about 40 clock cycles in the worst case. Because semaphore latency occurs whenever ports access the shared bank, the total latency increases rapidly if processors use the DPSDRAM infrequently.

To maximize the data transfer rate of mobile multimedia systems based on the DPSDRAM, the following semaphore latency problems must be solved:

- the latency of reading the semaphore register to check the authority of shared bank,
- the latency of waiting for the authority of the shared bank to be changed,
- the latency of using the mailbox register to request authority.

III. Proposed SAM Controller

In this section, the SAM controller to minimize the semaphore latency of DPSDRAM is described. Figure 4 shows the block diagram of the SAM controller. The functions for modules of the SAM controller are summarized in Table 1. The SAM controller follows the JEDEC [6] standard. The SAM controller is a modification of the Lattice Semiconductor's SDRAM controller IP [7]. The algorithms of SAM are described in the rest of this section.

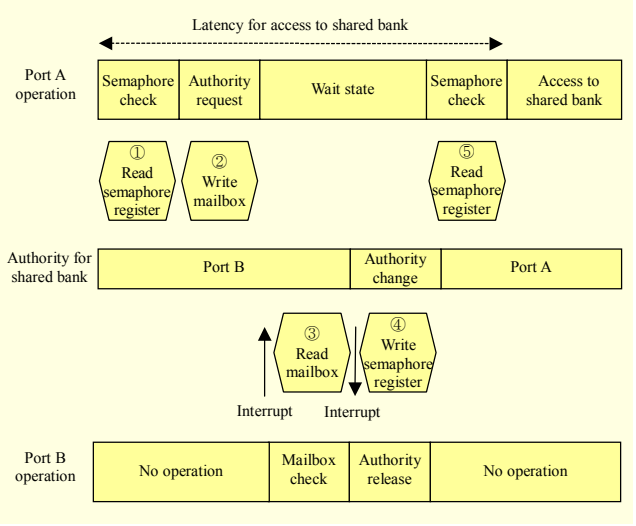

Fig. 3. Simplified timing diagram of semaphore latency.

1. Duplication of Semaphore Register

Only the port with the authority can access the shared bank of the DPSDRAM. To check the authority, the processors which are connected to both ports, need to read the semaphore register. A command which reads the semaphore register is added whenever the processors use the shared bank of the DPSDRAM. The latency of repeatedly reading the semaphore register creates a bottleneck which makes the system unable to transfer data at high speed. Especially if the processor uses the shared bank frequently, it becomes a serious problem.

The SAM controller reduces the number of times the

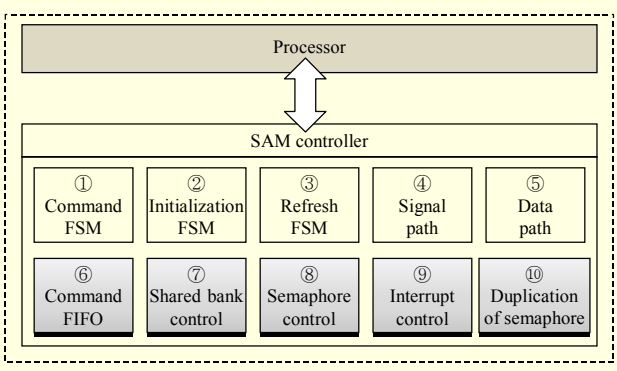

Fig. 4. Block diagram of the SAM controller for DPSDRAM.

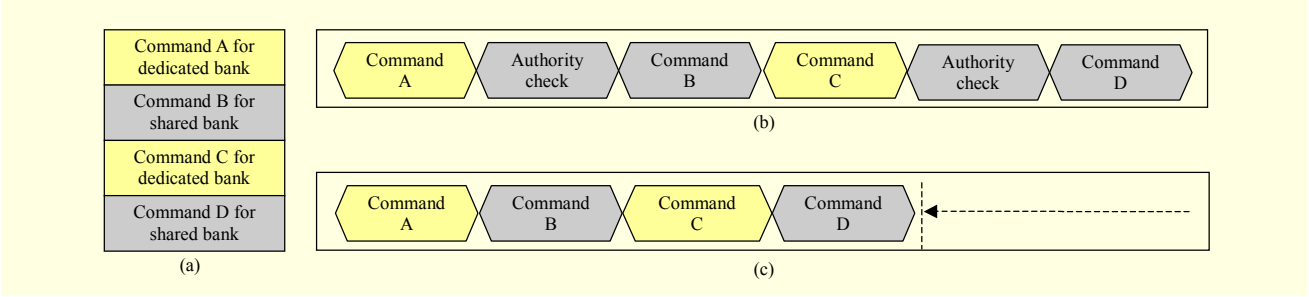

Fig. 5. (a) Command list of the processor with authority of the shared bank and a simplified diagram of commands executing (b) by using a traditional controller and (c) by using an SAM controller.

Table 1. Functions for modules of SAM controller.

Module	Function
(a) Command FSM	Handle DPSDRAM initialization
(b) Initialization FSM	Handle read/write commands
(c) Refresh FSM	Perform a periodic refresh of DPSDRAM
(d) Signal path	Generate signals such as RAS, CAS, Address, and so on.
(e) Data path	Act as interface for transferring data between the system and the DPSDRAM.
(f) Command FIFO	Store commands delivered from the system.
(g) Shared bank control	Handle commands for accessing the shared bank.
(h) Semaphore control	Perform the procedures of semaphore.
(i) Interrupt control	Handle interrupt signals delivered from the DPSDRAM
(j) Duplication of semaphore register	Save the authority of the shared bank

semaphore register is read by using the algorithm named *duplication of semaphore register*. The SAM controller includes the duplication of semaphore register (sem_reg), ⑩ in Fig. 4. The sem_reg can save the same value as the semaphore register. If the command for accessing the shared bank is delivered from the processor to the traditional controller, the value of the semaphore register must be read to check the authority. On the other hand, if the command for accessing the shared bank is delivered from the processor to the proposed SAM controller, there is no need to access the shared bank because the authority can be checked with sem_reg.

Figure 5 shows that the sem_reg of the SAM controller reduces the number of redundant authority checks. The processor with the authority of the shared bank is assumed to execute 4 commands as shown in Fig. 5(a). Command B and command D accessing the shared bank are executed after the semaphore register is read to check the authority as shown in Fig. 5(b). However, the SAM controller can directly execute the B and D commands without reading the semaphore register, as shown in Fig. 5(c), because it can check the authority by using sem_reg.

The sem_reg of the SAM controller that connects to baseband processor at port A and the sem_reg of the SAM controller that connects to the application processor at port B must be synchronized to the value of the semaphore register in the shared bank of the DPSDRAM. During boot-up, the semaphore register in the shared bank is synchronized to the default value, and sem_reg of the SAM controller is synchronized to the same value. For example, if the semaphore register is synchronized to the value of 1, then sem_reg is also synchronized to the value of 1. That is, the application processor at port B has the authority of the shared bank. The application processor can execute a command for accessing the shared bank immediately after checking the sem_reg. The baseband processor needs to request the authority to use shared bank.

If the value of the semaphore register in the shared bank of the DPSDRAM is changed, the two sem_regs of the SAM controllers which are connected to the two ports must be changed as well. The value of the semaphore register is changed only if the processor with the authority of the shared bank changes it by executing a write command. This operation is called *semaphore release*, and it occurs when the processor without the authority of the shared bank requests the semaphore. The value of sem_reg in the side of the processor with authority can be changed immediately after semaphore release. The processor without authority needs to read the semaphore register continuously after it requests the authority because it does not know when the other port releases the authority. The value of sem_reg in the side of processor without authority can be changed after the processor reads the changed value of the semaphore register.

Figure 6 shows the synchronizing sequences of the semaphore register and sem_reg of the SAM controller. The SAM controller at port A without the authority of the shared bank requests authority (①). Then, the SAM controller at port

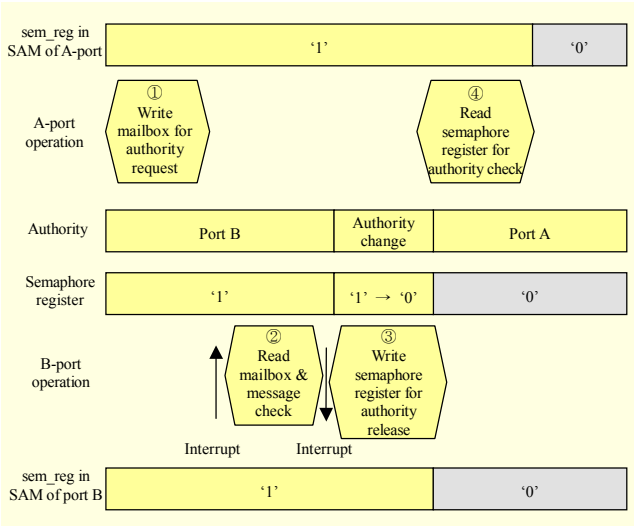

Fig. 6. Synchronizing sequences of the semaphore register and sem_reg.

B reads the mailbox contents (②) and it releases the authority of the shared bank (③). When this operation is completed, the semaphore register of the shared bank is changed to 0 from 1. The SAM controller at port B changes the value of its sem_reg to 0 from 1 immediately. However, the SAM controller at port A needs to read the semaphore register of the shared bank (④) in order to change the value of its sem_reg to 0 from 1.

2. Adaptive Command Prefetching

The processor without the authority of the shared bank needs to wait for the authority to be changed. As in [5], if the processor uses the shared bank infrequently, the latency of waiting for the authority to be changed is the major proportion of the total semaphore latency. To reduce the latency of waiting, SAM uses an adaptive prefetch [8] of a command to access the shared bank. The command FIFO, ⑥ in Fig. 4, queues the commands delivered from the processor and executes them in regular sequence.

The command FIFO selects the command for accessing the shared bank and informs the SAM controller in real-time. The SAM checks the authority of the shared bank by reading sem_reg. If the SAM has the authority of the shared bank, it sends a message to instruct the command FIFO to execute the commands in regular sequence. However, if the SAM does not have the authority of the shared bank, it instructs the command FIFO to execute the command to request authority before the execution of the next command. Because the authority of the shared bank changes before the SAM executes the command to access the shared bank, the latency of waiting for the authority to be changed is reduced.

Figure 7 shows that adaptive command prefetching reduces the latency of waiting for the authority to be changed at the port without the authority of the shared bank. The command FIFO queues 4 commands, and the current command is the command A as shown in Fig. 7(a). The traditional controller executes command A, B, and C and requests the authority of the shared bank before executing command D, which accesses the shared bank as shown in Fig. 7(b). The traditional controller needs to wait for the authority to be changed. The SAM controller requests the authority of the shared bank after executing the current command A and then executes command B and C as shown in Fig. 7(c). The SAM does not need to wait for the authority to be changed, it just needs to read the semaphore register to check the authority, and then it executes the final command D.

3. Auto-release of Authority

The processors at port A and port B request authority whenever they want to access the shared bank of the DPSDRAM. The following procedures need to be executed. First, the processor without the authority of the shared bank requests authority by writing a message in the mailbox, and an interrupt signal occurs at the other processor with authority. Then, the processor with authority needs to read the message in the mailbox and release authority. Frequent requests for authority by processors lead to additional latency caused by

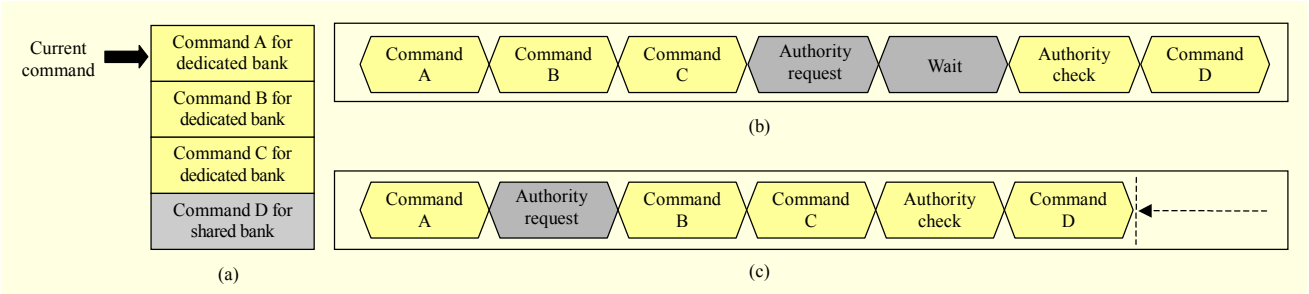

Fig. 7. (a) Example of command list in command FIFO and the diagram of their executing (b) without using the adaptive command prefetching and (c) using the adaptive command prefetching.

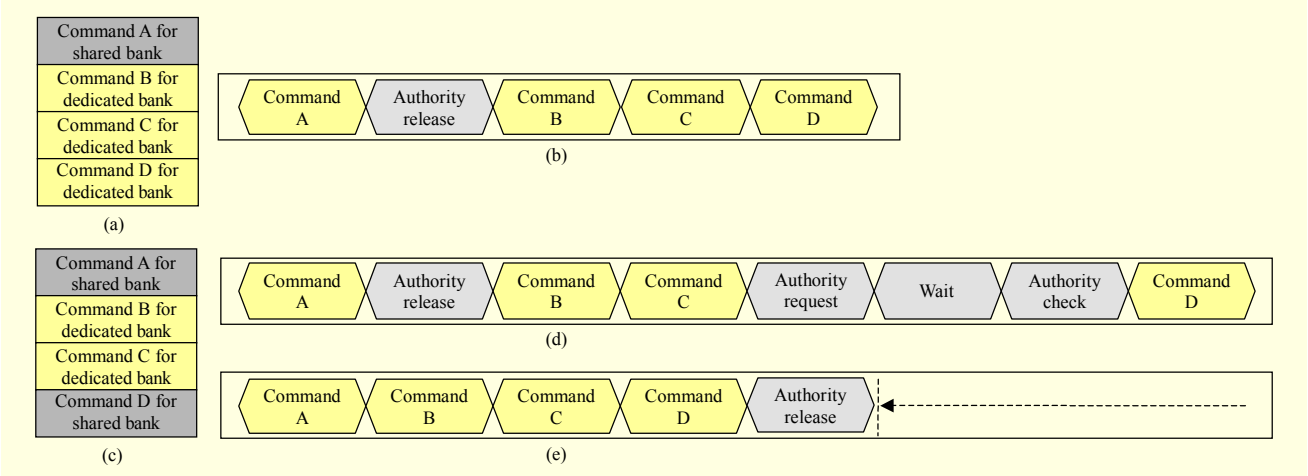

Fig. 8. Two cases of adapting the auto-release of authority.

using the mailbox register. To minimize this latency, interruption needs to be minimized by reducing the number of requests by the processors. If the processor with the authority of the shared bank releases authority before the other processor requests authority, the number of requests can be reduced. The auto-release of authority algorithm of SAM releases authority automatically without the other processor making a request. The command FIFO of the SAM controller is assumed to save one command for accessing the shared bank and three commands for accessing the dedicated banks as shown in Fig. 8(a). After executing the first command for access to the shared bank, the authority of the shared bank is released by the auto-release of authority algorithm. A diagram showing the execution of the commands is shown in Fig. 8(d). The command FIFO of the SAM controller saves two commands for accessing the shared bank as shown in Fig. 8(c). The additional operations of requesting, waiting, and checking the authority of the shared bank are executed because the authority is released by auto-release of authority algorithm after the first command for accessing the shared bank is executed.

4. Reconfigurable Architecture of SAM Controller

The DPSDRAM is connected to the two processors by two SAM controllers. The authority of the shared bank is changed whenever both SAM controllers request authority. The frequent changing of the authority leads to additional semaphore latency. The two processors have different patterns of DPSDRAM access. To avoid frequent changing of authority and to optimize the access pattern of the processors, the SAM controller is designed using the concepts of master and slave.

Table 2 summarizes the features of master and slave of the SAM controller. Figure 9 shows the access patterns of the two processors. The baseband processor which is connected to port A uses the shared bank for long durations but not frequently as shown in Fig. 9(a). In this pattern, the semaphore read and wait operations are critical to latency. The slave SAM is optimized to this pattern. The slave SAM can reduce the waiting time by using adaptive command prefetching, and the semaphore reading time can be reduced by using the duplication of semaphore register algorithm. After accessing the shared bank, the slave SAM can release authority by using auto-release of authority. Because the number of accesses to the shared bank is small, the default authority is not given. Figure 9(b) shows that the application processor, which is connected to port B, uses the shared bank frequently. The semaphore request, semaphore release, and semaphore read operations are critical to latency in this pattern. The master SAM is connected to this application processor. Default authority is needed by the master SAM to reduce the number of semaphore requests. The master SAM releases authority only when the slave SAM requests authority. Duplication of semaphore register of the master SAM can reduce semaphore read operations.

Table 2. Features of master and slave of SAM controller.

	SAM controller	
	Master	Slave
Default authority	O	X
Duplication of semaphore register	O	O
Adaptive command prefetching	X	O
Auto-release of authority	X	O

Table 3 shows the combination of master and slave controller and the evaluation results of data transfer time with two combination options. We measured the data transfer time from the baseband processor to the application processor with

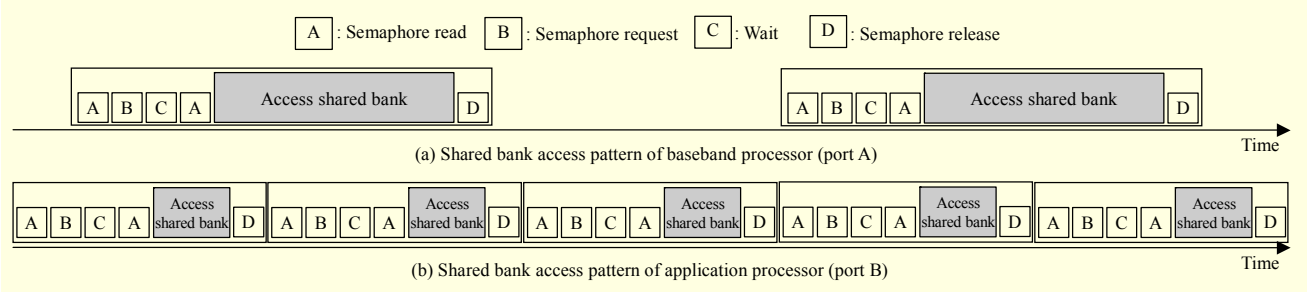

Fig. 9. Access patterns of two processors.

Table 3. Data transfer results of combinations of master and slave.

Option	Port	Processor	Controller	Data transfer time
1	A	Baseband	Slave	160 kB data: 9.138 ms
	B	Application	Master	320 kB data: 18.27 ms
2	A	Baseband	Master	160 kB data: 11.526 ms
	B	Application	Slave	320 kB data: 22.051 ms

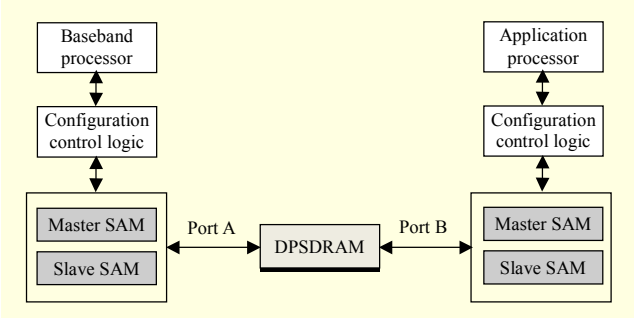

Fig. 10. Reconfigurable architecture of the SAM controller.

160 kB and 320 kB data.

Option 1, which is composed of the slave SAM controller at port A and the master SAM controller at port B, is the optimal solution for directional communication from the baseband processor to the application processor. In cellular systems, some applications, such as video phone call, require bidirectional communication between two processors. Figure 10 shows the reconfigurable architecture of the SAM controller. The reconfigurable logic is composed of the master and slave controller, and the configuration control logic chooses either the master or slave to use for communication between processors. The reconfigurable SAM increases the bidirectional communication speed.

IV. Simulation Results

In this section, the environment and the result of simulations to evaluate the SAM controller are presented. The data transfer rate and the memory bandwidth are the most important factors in high resolution image processing in mobile multimedia systems. We measured the data transfer rate between two processors and the bandwidth. We used the simulation model of Samsung Electronics' 1 Gb OneDRAM. The 1 Gb OneDRAM is composed of six banks. Port A has two dedicated banks and port B has three dedicated banks. The ports have one shared bank. Table 4 shows the specifications for both ports of OneDRAM. The data rate of port A is set to SDR, and that of port B is set to DDR. We used Cadence's NC-Verilog for RTL simulations.

We designed a traditional SDRAM controller named shared bank management (SBM) to compare with an SAM controller. The SBM controller checks the authority of the shared bank whenever a processor sends a command to access the shared bank. The SBM controller with the authority of the shared bank can execute the commands to access the shared bank. However, the SBM controller without the authority of the shared bank requests the authority and waits until the other controller releases the authority. The SBM controller with the authority of the shared bank needs to release the authority, when the other controller requests the authority. These operations are same as those shown in Fig. 3.

Through the shared bank of DPSDRAM, the processors can exchange the data. We measured the data transfer rate between two processors under two kinds of cellular system environments. The Nova H.264 decoder [9] was used as the application processor. It sends the data to the viewer, which shows the image of data. Figure 11(a) shows the environment of the traditional

Table 4. Specifications for port A and port B of the OneDRAM simulation model.

	Port A	Port B
Data rate	SDR	DDR
Frequency	Max. 133 MHz	Max. 133 MHz
Bit organization	X32	X32

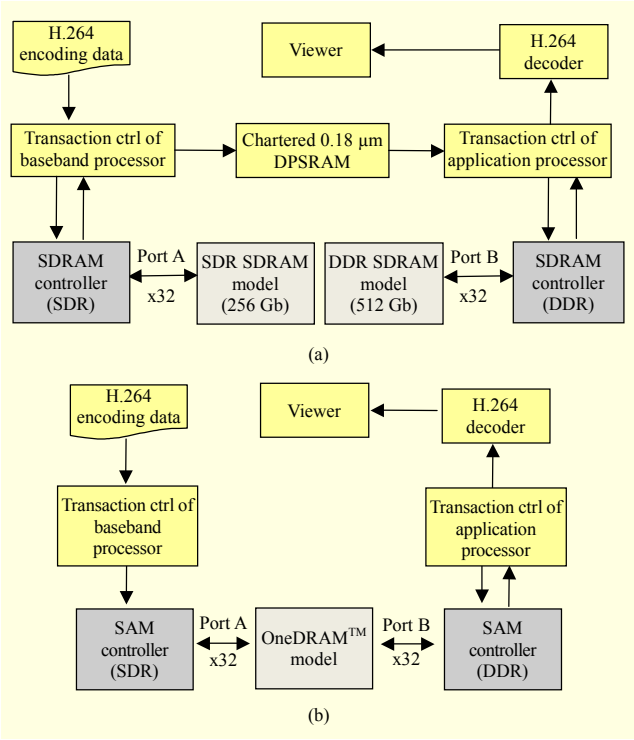

Fig. 11. Environments for measuring the data transfer rate: (a) with the DPSRAM interface and (b) with DPSDRAM.

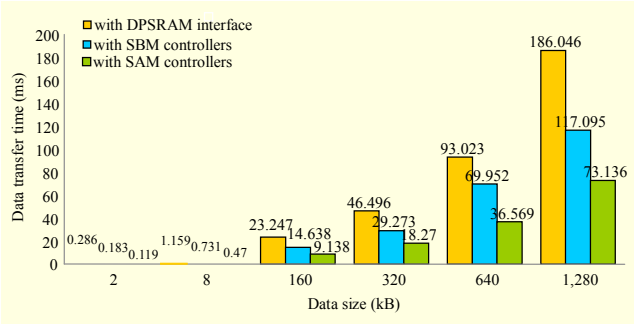

Fig. 12. Simulated data transfer rates between two processors.

system without the DPSDRAM. The Chartered Semiconductor's DPSRAM is used as a channel to exchange data between two processors. The environment of the new system with the DPSDRAM is shown in Fig. 11(b). The data transfer rate was measured using SBM controllers and using SAM controllers. We used qcif format files among YUV 4:2:0 video sequences of VTRG [10] as the benchmark data to transfer from the baseband processor to the application processor. Figure 12 shows the results of data transfer time between the two processors with the DPSRAM using SBM controllers or SAM controllers. The SAM controllers improve the data transfer rate by 1.5 to 1.6 times between two ports of DPSDRAM compared with the traditional controllers.

Figure 13 shows the architecture of the testbench for

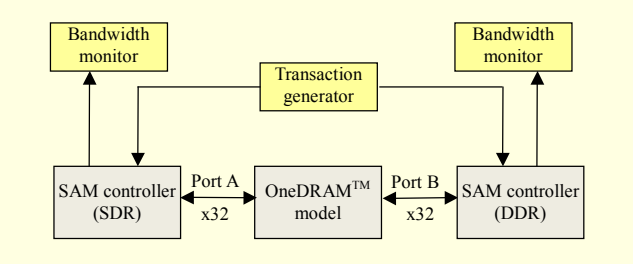

Fig. 13. Testbench for bandwidth measurement of SAM controller.

bandwidth measurement of port A and port B of the DPSDRAM. The SAM controllers of both ports executed random transactions of 10, 100, 1,000, and 10,000. One transaction consists of one write and one read for the same address and bank. Each transaction had an irregular ratio of commands accessing the shared bank to commands accessing the dedicated banks because the transactions were generated randomly. The commands accessing the shared bank included semaphore operations such as semaphore read, semaphore request, semaphore release, and wait. Any transaction which includes many commands accessing the shared bank has a high possibility of leading to a low bandwidth result. The bandwidth monitors calculated the bandwidth of both ports after finishing the execution of the transactions under the following conditions: OneDRAM as port A SDR x32 with 66 MHz, port B DDR x32 with 66 MHz, and the SAM controllers with 66 MHz. The burst length of SDR was set to 1 and DDR was set to 2.

The measured bandwidth from both ports of the OneDRAM is shown in Fig. 14 according to the numbers of transactions. The results of 10 and 1,000 random transactions show low bandwidth because transactions have commands accessing the shared bank at a high rate. The bandwidth of the SAM controller shows an enhancement of up to 38% for port A, as shown in Fig. 14(a), and up to 31% for port B as shown in Fig. 14(b). Additional performance enhancement can be achieved if the operating frequency of the environment is increased to 133 MHz and the burst length of the OneDRAM is changed to 2, 4, 8, 16, and so on.

V. Conclusion

In this paper, an SAM controller to maximize the data transfer rate between two processors used for mobile multimedia systems based on a DPSDRAM has been proposed and tested. The semaphore latency of the DPSDRAM slows down the data transfer rate and reduces the memory bandwidth. The SAM can minimize semaphore latency of reading the semaphore register of the DPSDRAM, waiting for the authority of the shared bank to be changed, and

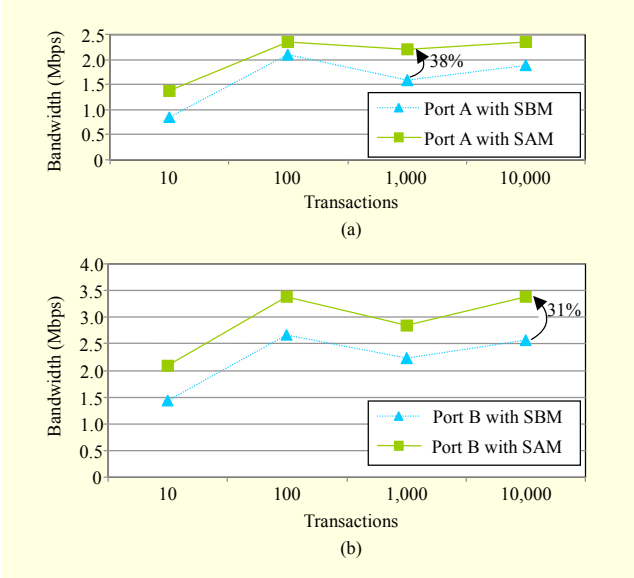

Fig. 14. Measured bandwidth results of SAM controller and SBM controller (a) at port A and (b) at port B.

requesting and changing the authority. The results of simulations show that the SAM controllers increase the data transfer rate between two processors and enhance the bandwidth of the DPSDRAM compared with traditional controllers. The SAM can be used for real-time high-resolution image processing, such as HD video streaming in mobile multimedia systems based on DPSDRAM, because it guarantees a fast data transfer rate and huge bandwidth.

Acknowledgment

The first author would like to thank S.M. Jung for helpful comments on this paper.

References

[1] C.G. Hwang, "New Paradigms in the Silicon Industry," *IEDM Tech. Dig.*, 2006, pp.1-8.

[2] J. Kim et al., "A 512 Mb Two-Channel Mobile DRAM (OneDRAM) with Shared Memory Array," *IEEE J. Solid-State Circuits*, vol. 43, no. 11, Nov. 2008, pp. 2381-2389.

[3] X. Zhang et al., "Evaluating and Designing Software Mutual Exclusion Algorithms on Shared-Memory Multiprocessors," *IEEE Parallel and Distributed Technology: Systems & Applications*, vol. 4, Mar. 1996, pp. 25-42.

[4] R.T. Jacob and I.P. Page, "Synthesis of Mutual Exclusion Solutions Based on Binary Semaphores," *IEEE Trans. Software Engineering*, vol. 15, May 1989, pp. 560-568.

[5] H. Yang et al., "Performance Evaluation and Optimization of Dual-Port SDRAM Architecture for Mobile Embedded Systems," *CASES'07*, Sept. 30, 2007, pp. 53-57.

[6] JEDEC Standard, Double Data Rate (DDR) SDRAM Specification, 2005.

[7] The SDRAM controller IP of Lattice Semiconductor Corporation. Available: http://www.latticesemi.com

[8] M. Dasygenis et al., "A Combined DMA and Application-Specific Prefetching Approach for Tackling the Memory Latency Bottleneck," *IEEE Trans. VLSI Systems,* vol. 14, Mar. 2006, pp. 279-291.

[9] NOVA: a H.264/AVC Baseline Decoder Specification. Available: http://www.opencores.org

[10] YUV 4:2:0 Video Sequences of Video Traces Research Group. Available: http://trace.eas.asu.edu

Jaehwan Kim received the BS degree in computer science from Hanyang University, Seoul, Korea, in 2006. He is currently pursuing a unified MS and PhD course in electronics and computer engineering at Hanyang University. His current research interests are VLSI and CAD as well as SoC design methodology, especially memory centric design methodology.

Jongwha Chong received the BS and MS degrees in electronics engineering from Hanyang University, Seoul, Korea, in 1975 and 1979, respectively, and the PhD in electronics and communication engineering from Waseda University, Japan, in 1981. Since 1981, he has been a professor of the Department of Electronics Engineering, Hanyang University. From 1979 to 1980, he was a researcher with C&C Research Center of Nippon Electronic Company (NEC). From 1983 to 1984, he was a visiting researcher with the Korean Institute of Electronics and Technology (KIET). In 1986 and 2008, he was a visiting professor at the University of California, Berkeley, USA. He was the chairman of the CAD and VLSI Society in 1993. In 2007, he was president of the Institute of the Electronic Engineers of KOREA (IEEK). He was the director of the Institute of Information and the Communication Center in 1997 and dean of Graduate and Undergraduate School of Information and Communications in 1999 at Hanyang University. He is currently the president of the Korean Institute of Electrical and Electronics Engineers (KIEEE) and the chairman of Fusion SOC Forum. His current research interests are in SoC design methodology, including memory centric design, indoor wireless communication SOC design for ranging and location, video systems, and power IT systems.