PAPER

A Self-Timed SRAM Design for Average-Case Performance

Je-Hoon LEE[†a)], ***Member***, **Young-Jun SONG**[††], ***and*** **Sang-Choon KIM**[†b)], ***Nonmembers***

SUMMARY This paper presents a self-timed SRAM system employing new memory segment technique that divides memory cell arrays into multiple regions based on its latency, not the size of the memory cell array. This is the main difference between the proposed memory segmentation technique and the conventional method. Consequently, the proposed method provides a more efficient way to reduce the memory access time. We also proposed an architecture of dummy cell and completion signal generator for the handshaking protocol. We synthesized a 8 MB SRAM system consisting of 16 512K memory blocks using Hynix 0.35-μm CMOS process. Our implantation shows 15% higher performance compared to the other systems. Our implementation results shows a trade-off between the area overhead and the performance for the number of memory segmentation.

key words: *asynchronous circuit, SRAM, self-timed logic, memory segmentation*

1. Introduction

Recently, low-power consumption and high-performance have been the major concern in deep sub-micron ASIC designs. Most of these ASICs are synchronous and their activities are controlled by a global clock which triggers at the same time. The designers focus mostly on the data processing by assuming the existence of a global clock even if this timing assumption is based on delay models that have questionable accuracy [1]. Furthermore, the gate delay is decreased under downscaling whereas the global wires do not scale in length since they communicate signals across the chip. Increased demand for high speed devices causes many signal integrity challenges and time closure problems such as clock tree structure, clock-gating, and clock skew problem [2].

An asynchronous design is free from those problems because it employs local handshaking, not global clock. It allows a large number of clock signals globally distributed throughout the ASIC to be replaced by the local handshaking between the neighboring circuitries. An asynchronous circuit is locally controlled by handshaking signals such as request and acknowledge that are generated when and where they are needed. Asynchronous design has many potential advantages over synchronous design such as no clock skew, low-power consumption, and easy global timing. For these advantages of asynchronous design methodology, many research results have been reported in the field of asynchronous design and in particular, asynchronous processors [3]–[7].

ASRAM (asynchronous static random access memory) has been introduced in many asynchronous designs that can be classified into three methods according to their delay model, a) bounded delay, b) DI (delay insensitive), and c) SI (speed-independent) delay model. Generally, the asynchronous circuit based on DI delay model uses data encoding to generate the completion signal of all function blocks whereas the asynchronous circuit based on bounded delay model uses constant delay element that matches the worst-case latency corresponding function block. It ensures a timing relationship between data and handshaking signals, which must take account of gate and wire delay. That is, the delay in the control signal must not be less than the delay in the combinational circuitry. In SI delay model, all wire delays are negligible while gate delays are unbounded. Whereas SI and DI are well defined properties under the unbounded gate and wire delay model, asynchronous circuits whose correct operation relies on more elaborate or engineering timing assumptions are simply called self-timed [1].

An asynchronous processor should interact with an asynchronous memory by handshaking protocol. When we construct asynchronous memory, we require the completion signal of read and write operations for handshaking protocol. When we employ the bounded delay model, the matched delay should be obtained from the precise post layout simulation. This delay element is commonly used to generate the acknowledge signal for memory access that is initiated by a request signal from an asynchronous processor for read and write operation. The time to generate acknowledge signals is longer than the time to complete memory access because the latency of delay element is long enough to indicate the completion of corresponding memory access. The matched delay represents worst-case latency of each bit-line in the memory cell. Consequently, it operates with worst-case performance, not average case performance, which is significant advantage of asynchronous design.

In literature, many implementations of the asynchronous memory have been evaluated. AMULET3i processor that was presented by J. Garside [4] employed dual-ported, unified memory structure instead of separated in-

Manuscript received September 7, 2010.
Manuscript revised March 28, 2011.

[†]The authors are with Division of Electronics and Information Communication Engineering, Kangwon National University, 1 Joongang-ro Samcheok, Gangwon, 245–711, Republic of Korea.
[††]The author is with the Chungbuk BIT Research-Oriented University Consortium, Chungbuk National University, San 12, Gaeshin-dong, Cheongju, Chungbuk, 361–763, Republic of Korea.
a) E-mail: jehoon.lee@kangwon.ac.kr
b) E-mail: kimsc@kangwon.ac.kr

DOI: 10.1587/transinf.E94.D.1547

struction and data memory. It was divided into eight 1 KB blocks, each of which has two ports in order to reduce the energy cost of each memory access [8]. The speed of local RAM access varies by about 20% since it is wholly asynchronous. Furthermore, the control circuitry for handshaking is complicated. It used both a line fetch latch and a synchronization latch for each bit-line in order to generate the completion of read operation. In the asynchronous cache system included in AMULET2e, timing had been done by including extra, dummy bits within the custom logics of the datapath as in other parts of the AMULET processor [9]. V. Sit [10] presented asynchronous SRAM that can communicate with other asynchronous system based on a four-phase handshaking by generating read/write completion signals with increased average speed by the variable bit-line load concept. Thereby, the average speed performance is improved. However, this concept still suffers from the diminished performance according to the size of memory. In addition, the demand for the high-speed memory for asynchronous processors ever increased.

Even though individual memory array has a very regular structure and produces data in a constant time, each part of the memory will have its own characteristic timing. An asynchronous memory system can naturally accommodate such timing variation. However, there is much room for exploiting high-speed and low-power memory system using more subtle timing variation in asynchronous SRAM.

This paper presents an implementation of self-timed memory system. It is divided into several regions and they can generate the completion signal separately when the read/write operation is finished. V. Sit [10] presented the memory architecture that differentiates the time to access the memory cells at different location. Similarly, we divided the memory into several regions but each region has different size to improve the memory access speed without significant area overhead. The impact of the time to access the memory cell on the different location is also analyzed. Then, we proposed the trade-off between the number of regions and the hardware complexity. It is proved that the delay of the memory cell is increased exponentially according to the location of the memory cell. Finally, the proposed design contains small completion generation circuitry and dummy cell.

The remainder of this paper is organized as follows. Section 2 describes the control mechanism of self-timed memory and analyzes the delay of the bit-lines in a memory. Section 3 presents the proposed completion generation circuitry and the method for placing dummy cells for average-case performance. Section 4 presents the experimental results. Section 5 draws conclusions.

2. Control Mechanism and Memory Segmentation Technique

This section presents the proposed self-timed SRAM architecture and the proposed memory access method with asynchronous processor using 4-phase handshaking protocol. The proposed memory segmentation technique is also presented. This technique allows all memory segments to have same delay time of their bit-line for achieving average-case performance.

2.1 The Proposed Memory Access Method

The proposed self-timed SRAM consists of five major parts as shown in Fig. 1: memory cell array, Row decoder, Column multiplexer, Completion generator, and Handshake control. The first three parts of the proposed SRAM are same with that of the typical memory design [11]. Most SRAM designs consist of multiple cell arrays and each memory cell has general architecture that is constructed as 6 transistors. The proposed SRAM uses the general row address decoder that is broken into two pieces. The first piece, Region decoder is responsible for selecting one of all memory regions and the second part, Row decoder, is to select one of the row lines. Then, WL driver raising only one row among the all word-lines. After WL activation and the sensing are complete, Column multiplexer is used to select the addressed data from one or more of these bit-lines. Pass transistor arrays are used to achieve variable bit-line load owing to the memory segmentation technique [10]. Completion generator and Handshake control are newly introduced for the handshaking protocol between the asynchronous processor and the proposed self-timed SRAM. In addition, we divide the memory cell array into multiple memory regions from 1 to i that has different size of the memory cell array according to the delay time of memory access. We add the dummy cell to every memory region so as to detect the completion time of memory access to the corresponding memory region.

We introduce the self-timed design technique that refers the circuit whose correct operation relies on more elaborate timing assumptions comparing to the circuit based on SI delay model that operates correctly assuming ideal zero-delay wires. In addition, we introduce the dummy cell that is responsible for indicating the completion of memory access to the corresponding memory cell. However, there is time difference in completion time of memory operation between the corresponding memory cell and the dummy cell. Thus, it is inevitable to ensure the completion obtained from the dummy cell should be driven after the completion of memory operation to the corresponding memory cell. The safety margin for completion is required to solve this problem. In this paper, the dummy cells are located at the last word-line of all memory regions so that they have the longest bit-line track in every memory region and they requires more small safety margin caused by difference in delay of bit-line. A dummy cell requires preceding works such as pre-writing and pre-reading before the memory operation will be commenced. We trim this delay time to do these works will be greater than the safety margin for completion using the post-layout simulation. Consequently, the time for preceding works in dummy cell can be used to guarantee safety margin for completion of memory operation on the memory cell array instead of the matched delay in asyn-

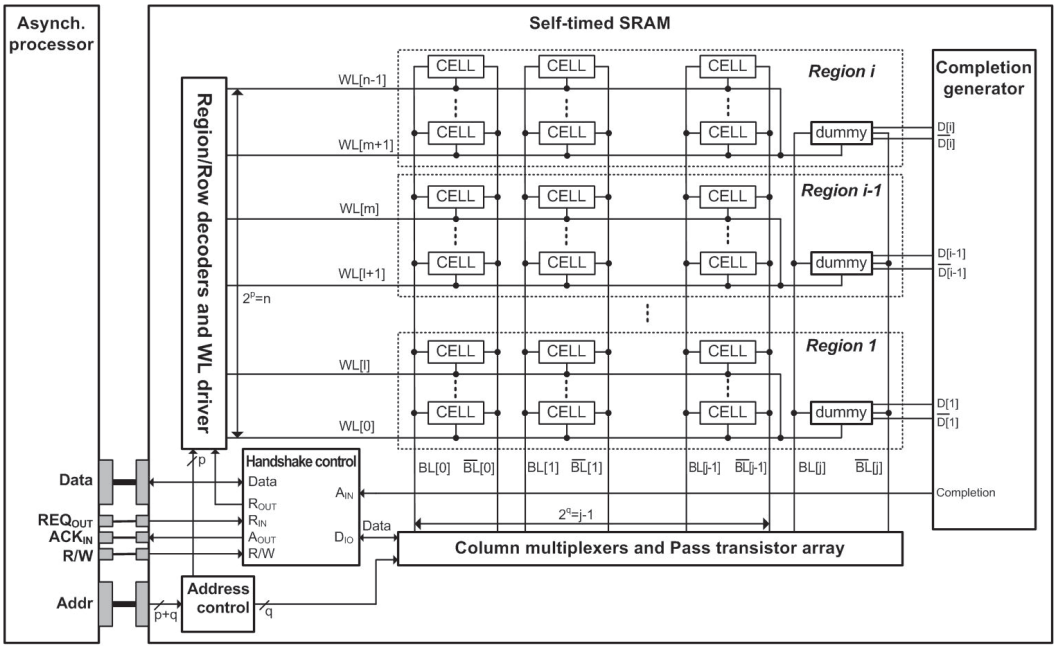

Fig. 1 The proposed SRAM architecture consisting of memory cell array, dummy cell, and completion generator.

chronous circuits employing bounded delay model.

The remaining procedure of generating a completion signal for the memory cell array can be described as follow. After the dummy cell senses the completion of the memory access, it transits the output *high* to Completion generator to indicate the given work is finished. For example, the dummy cell in memory region *i* transit its outputs D[i] and $\overline{D}[i]$ to *high* and *low*, respectively when the memory operation for any cell in the memory region *i* since the dummy cell sense the completion of that operation. As soon as Completion generator receives those signals, it issues the completion signal for acknowledgement.

The procedure for memory access can be summarized as follows. The memory operations consist of read and write operation. Both of them are triggered by the transition of request signal, REQ_{OUT} with valid input address transferred from the asynchronous processor. All bit-line pairs (BL and $\overline{BL}$) are charged to a predetermined voltage to reset bit-lines and dummy cells are also set to predetermined value according to the kind of operations. Then, either read or write operation for the memory cell is performed. Also, the given operation is performed for dummy cell in the same way to generate the completion signal. When the dummy cell generates the completion signal, it transfers the output signal pairs (D and $\overline{D}$) to Completion generator. Then, Completion generator transfers the Completion signal to the Handshake control, and then it issues the acknowledge signal, A_{OUT} to the asynchronous processor to indicate the given operation is completed. The asynchronous processor responds by taking request signal, REQ_{OUT}, low in order to indicate the given operation is no longer guaranteed to be valid. At last, a handshake control block in SRAM acknowledges this by setting the acknowledge signal, A_{OUT} in low. At this point the asynchronous processor can initiate the next memory operation.

2.2 The Proposed Memory Segmentation Technique

In this section, we present the proposed memory segmentation technique that divides the memory cell array into multiple regions to generate the completion signal separately. Our segmentation technique is based on the delay time, not the size of the memory cell array. Therefore, we can construct each memory region with same delay as represented the delay of dummy cell. It ensures that different memory regions have different time to access the memory cell according to the maximum delay time of bit-line.

It is inevitable to generate the varied completion signals according to the location of memory cell array since the memory access time varies directly as the physical location of memory cell. As the growing needs for high-capacity SRAM, the worst-case delay for the longest bit-line will be increased. A dummy cell array is inserted in order to detect completion time of the corresponding memory cell. However, the number of dummy cell and completion generator significantly impact on the circuit area.

Figure 2 shows the different kinds of memory segmentation according to the layout implementation of dummy cell. Many of asynchronous SRAM implementations use the single dummy cell for the entire memory cell array as shown in Fig. 2 (a). In this case, it has the smallest overhead area because it requires single dummy cell. However, the memory access time is restricted by the maximum delay for whole memory system. Thereby it operates with worst-case performance, which is one of the drawbacks of synchronous SRAM design. Figure 2 (b) shows the example of mem-

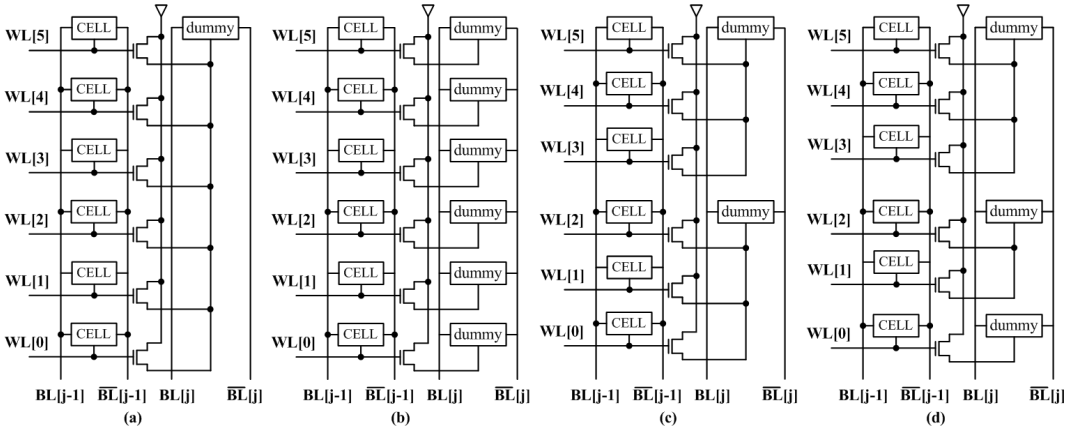

Fig. 2 Comparison of memory segmentation techniques according to (a) the maximum bit-line delay, (b) the dummy cell connected to the every word-line, (c) the dummy cell for the all blocks having fixed length, (d) the dummy cell located on the every blocks with variable length.

ory architecture that every word-line is connected with the dummy cell. Theoretically, the highest memory access time is achieved when all word-lines are connected to the dummy cell. However, it causes a significant area inefficiency due to the circuit area significantly due to the additional dummy cell for every word-line. Furthermore, the increasing complexity of completion generator has a significant impact on the circuit size of whole memory system.

To solve this problem, V. Sit [10] presented the memory segmentation technique using the variable bit-line load concept as shown in Fig. 2 (c). It divides the whole memory system into multiple regions. Each region has single dummy cell that represents the maximum delay of the corresponding memory regions. The completion signal is generated by comparing the accessed memory region and its maximum delay time obtained from the dummy cell. The simulation results showed that the segmented memory can reduce the memory access time by as much as 40% even though they are divided into four fixed-length regions without consideration of the bit-line delay in practical.

Due to the fact that the delay of bit-line in memory system depends on the overall load capacitance of word-lines, it increases as the number of word-lines increase. The slope is almost linear at the beginning, and then it changes into lower values. Therefore, it is more efficient way to divide the whole memory system into the multiple regions that has same transmission delay, not same size of the memory cell array.

In this paper, we propose the memory segmentation technique based on the bit-line delay of the memory cell. We include a dummy cell in each memory region to generate a completion signal. Each memory region has only one dummy cell as shown in Fig. 2 (d). The memory access time is same for every memory cell in the same region. V. Sit [10] divided the whole memory cell array into the multiple regions that has different delays with same size. On the contrary, we divide memory so that each memory region has same delay with different size of region. This difference is well expressed in Figs. 2 (c) and (d).

There are two different kinds of operations for memory access such as read and write operations. The memory access time for read and write operations are defined as Eqs. (1) and (2), respectively. Both of them include the bit-line delay time, T_{BL}. The total memory access time for read operation, T_R, can be defined as the sum of bit-line delay, the read operation time, and the completion generation time for read operation. Similarly, the memory access time for write operation, T_W, can be defined as the sum of bit-line delay, the write operation time, and the completion generation time for write operation.

$$T_R = T_{RD} + T_{BL} + T_{RC} \tag{1}$$

$$T_W = T_{WD} + T_{BL} + T_{WC}, \tag{2}$$

where T_{BL} represents the bit-line delay time for Eqs. (1) and (2). T_{RD} and T_{WD} represent the latency for read and write operation for the memory cell and the corresponding dummy cell, respectively. T_{RC} and T_{WC} represent the delay for generating a completion signal in the completion generator for read and write operation, respectively. Our implementation includes the memory cell arrays, dummy cells, and Completion generator. All delay times for all elements can be directly obtained from the post-layout simulation results.

When we divide the whole memory system into multiple regions, all delays except the bit-line delay are almost constant. Since the load capacitance of bit-line shall be increased as the increasing number of word-lines, the memory region containing the long memory cell arrays has a long bit-line delay. The maximum bit-line delay for each memory region will be varied according to the size of the memory region. In order to divide the memory system into multiple regions that have same latency, the bit-line delays for all memory regions should be same. After we should estimate the precise bit-line delay for the whole memory cell arrays, we determine the number of memory regions which is obtained from the trade-off between the circuit size and the performance. We make a bit-line load model for the whole memory cell arrays using R-C model as shown in Fig. 3,

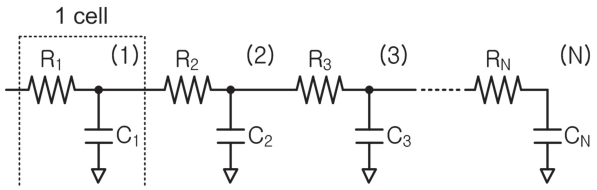

Fig. 3 R-C modeling for bit-line delay.

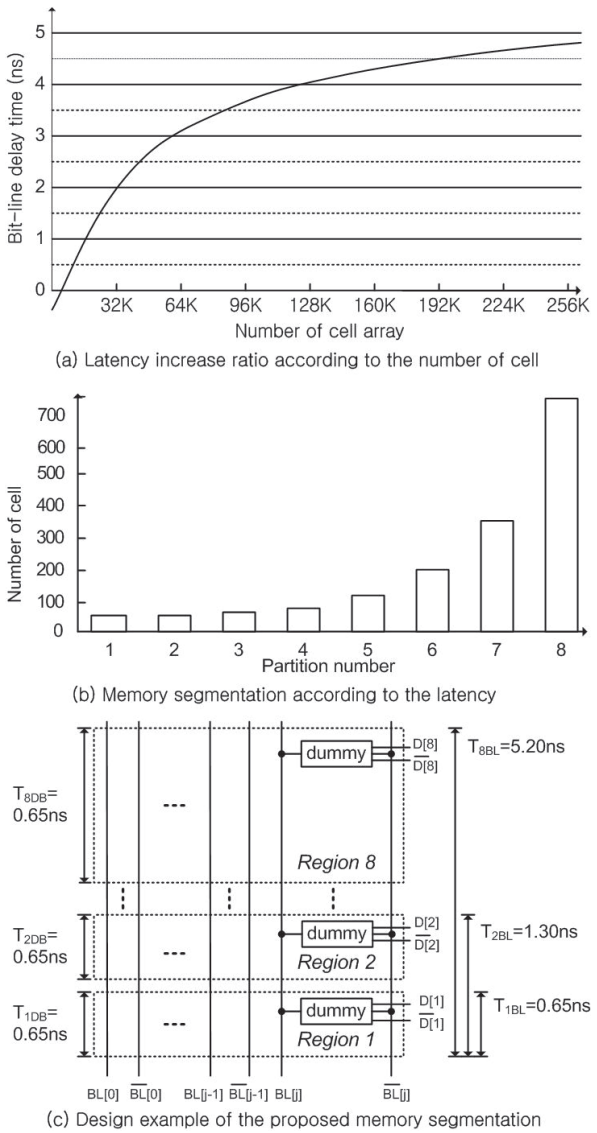

Fig. 4 The proposed memory segmentation technique, (a) the bit-line delay increasing ratio according to the number of cell, (b) memory segmentation according to the bit-line delay, (c) design example of the proposed memory segmentation.

which is similar way presented by Elmore [12]. The delay owing to this model can be also extracted by the post-layout simulation of memory system. The parameters that are used in the simulation obtained from Hynix 0.35-μm CMOS process [13].

The simulation results for the bit-line delay are shown in Fig. 4 (a). The size of cell array is increased from 0 to 256 KB. This bit-line delay is fit to the logarithm function. This figure suggests an important feature of the bit-line delay, which monotonically increases. However, the slope is not linear at the overall axis of the increasing size of memory cell array. Note that the graph in Fig. 4 (a) is indeed increasing slowly and that it is concaving down. Consequently, it has more benefit to divide memory system based on its bit-line delay instead of the size of memory segment since it can exploit high-speed memory system using more subtle timing variation in the ASRAM.

For example, we divide 256K subarrays into 8 regions that have same bit-line delay. Total bit-line delay for 256K subarrays is 4.8 ns as shown in Fig. 4 (a). Every memory region has same delay bound, 0.6 ns since we divide the whole memory cell arrays into 8 regions. That is to say, the bit-line delay bound of every memory region can be obtained when total bit-line delay is divided into the number of region. However, every bit-line delay for the different memory regions is different. Therefore, all memory regions have different size of memory array as shown in Fig. 4 (b). The bit-line delay for upper memory region is bigger than that of lower memory region as shown in Fig. 4 (c). The more the memory region is close to the Column multiplexes, the more it has the shortened bit-line delay. Consequently, the bit-line delay for the k-th memory region, T_{kBL}, can be obtained from Eq. (3).

$$T_{kBL} = \sum_{n=1}^{k} Max(T_{nDB}) = k \times Max(T_{kDB}), \tag{3}$$

where k is the number of memory region. $Max(T_{kDB})$ represents the maximum delay bound for k-th memory region. Every delay bound for different memory regions is same in the proposed self-timed SRAM system owing to the proposed memory segmentation method. However, all bit-line delays for different memory regions are different as shown in Fig. 4 (c). Therefore, the memory access time is varied according to the location of memory region.

3. The Proposed Dummy Cell and Completion Generator Circuit

When the self-timed SRAM employing a completion generator circuitry with dummy cells is compared to the self-timed SRAM employing bounded delay model, two significant advantages stand out. The one is reusability and the other is performance. In particular, the reusability is one of the crucial design concerns. The work for generating the delay element is the most time-consuming process since it can be obtained from the precise post-layout simulation. In addition, this work should be repeated to construct the delay element whenever designers change the target fabrication process. In order to avoid this problem, we employ the dummy cell and Completion generator circuit rather than the matched delay as shown in Fig. 1 so as to indicate the completion of the given memory operation.

Nevertheless, the self-timed SRAM system employing bounded delay model is widely used since it has advan-

tages in the circuit area and robustness in operation. In general, the conventional asynchronous circuit employing the bounded delay model has a drawback in terms of performance since it requires the delay element that matches the worst-case latency for memory read or write operation. However, the proposed SRAM system can overcome this drawback regardless of which delay model to adopt. It is achieved by the proposed memory segmentation technique that can provide different bit-line delays according to the corresponding memory region. Since the maximum delay bound of memory regions is obtained from the memory segmentation process, we replace the dummy cell and completion generator circuit to the fixed delay and we add the other matched delay that has fairly large safety margin for the worst-case latency of the memory read or write operation. In this section, we present the details of the dummy cell and completion generator circuitry.

3.1 The Proposed Dummy Cell

The proposed self-timed SRAM consists of multiple regions having different size of the memory cell array. A dummy cell is connected to the last word-line of each memory region. When the word-line transits to *high*, the corresponding word-line is selected for the memory operation. Then the bit-line is used either to read data from the memory cell or to write data to it. The last bit-line for the last memory cell in the word-line is also used for the relevant dummy cell. Consequently, the completion of memory operation for the dummy cell indicates the completion of memory operation for the memory cell that is connected with the same word-line.

Memory operation can be categorized into read and write operations. Two different preceding works such as pre-writing and pre-reading are needed for the dummy cell so as to detect when the read and write operations are completed. First, pre-writing is to write data into the dummy cell. This has to be proceed before the memory read operation will be commenced. Second, pre-reading is to reset the dummy cell. This operation must take precedence in order for dummy cell to write data. In memory read operation, the completion signal will be generated as soon as the dummy cell outputs the pre-written data. The memory write operation is performed in the similar way. The write completion for dummy cell will be generated when the dummy cell outputs the valid data.

The proposed dummy cell includes the PW (pre-writing) and PR (pre-reading) circuitries as shown in Fig. 5. A PW circuit is responsible for storing ‘1’ to the dummy cell before the memory read operation. A PR circuit is responsible for resetting the dummy cell before the memory write operation. Completion generator is responsible for checking the outputs of dummy cell during memory read operation. It is also responsible for the internal state value of dummy cell during memory write operation. If the dummy cell does not complete the given operation, it outputs all zeros on two output D and $\overline{\text{D}}$. If not, it outputs the data and the

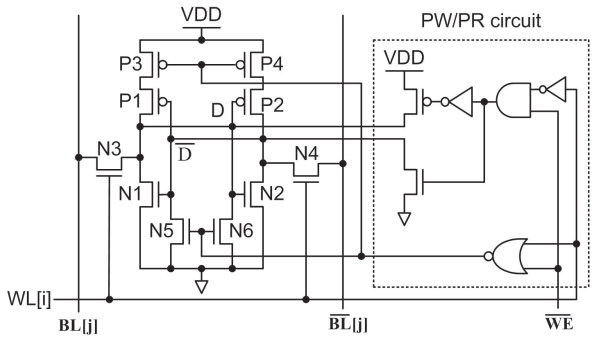

Fig. 5 The proposed dummy cell architecture including PW (Pre-Write) and PR (Pre-Read).

logical complementary data on two output D and $\overline{\text{D}}$. Therefore, the Completion generator connects to the dummy cell for each memory region. It generates the completion signal for Handshake control unit when it received the complement data from the dummy cell during memory read operation. Also, Completion generator compares the input data onto the bit-line for the dummy cell to the data stored in dummy cell. It generate the completion signal when both of them are logical complementary.

The procedure for generating a completion of memory write operation can be summarized as follows. A write enable signal, $\overline{\text{WE}}$ is used to control the internal value of dummy cell. First, the dummy cell stores logical ‘1’ when write enable signal, $\overline{\text{WE}}$ is high and word-line, WL[i] is low. Then, the write enable signal, $\overline{\text{WE}}$ is transit to low, N5 and N6 in dummy cell are turn on. Thus, the internal values of dummy cell, D and $\overline{\text{D}}$ are reset to all zeros. If the word-line, WL[i] is transit to high, N5 and N6 are turn off and P3 and P4 are turn on. At this time, the reset operation for dummy cell is stopped and the dummy cell acts as conventional SRAM. Thus, the transferring data on the bit-line, BL[j] is stored to the dummy cell. After the data transfer is finished, the word-line, WL[i] and the write enable signal, $\overline{\text{WE}}$ transit into low and high, respectively. The internal value of dummy cell, D and $\overline{\text{D}}$ are connected to the power supply and the ground, respectively. Thus, the dummy cell is recovered and it stores data ‘1’. The procedure for the memory read operation is similar with that of memory write operation. Only word-line is transit to high. The process for pre-writing operation is skipped. As the conventional SRAM, the dummy cell outputs the internally stored value, ‘1’ onto the bit-line.

The proposed SRAM requires pre-writing operation in dummy cell before the write operation so as to detect the completion of write operation by setting the internal value of dummy cell to ‘1’. However, the pre-charging on DRAM is to avoid the voltage difference of the bit-line pairs caused by destructive reads. This operation turns off active word-line and pre-charge bit-line pairs to ready state. This fact is a main difference between pre-write operation in the proposed SRAM and the pre-charge operation in the conventional DRAM.

3.2 The Completion Generator and the Operation Procedure

The dummy cell and Completion generator are used to detect the completion time of given memory operation and they allow to Handshake control to generate acknowledge signal. The architecture of the Completion generator is illustrated in Fig. 6. In this figure, D and $\overline{D}$ represent the stored data and its complementary, respectively, in dummy cell for detecting the completion. SD and SDb are generated after D and $\overline{D}$ are passed in the sense amplifier, respectively. Di and $\overline{Di}$ are the new data inputting into the dummy cell and Si and Sib are obtained from the sense amplifier input, Di and $\overline{Di}$, respectively. SB and SBb are obtained from the sense amplifying the internal data in dummy cell. They are driven when word-line is chosen and write enable, $\overline{WE}$, is high. The detailed architectures of the completion signal generator for read and write operations are shown in Figs. 7 (a) and (b), respectively.

Before the read or write operation for memory system, all dummy cells should be discharged in order to prevent the erroneous detection of completion time, in particular the write operation. Then, it precedes pre-writing or pre-reading processes with the reading or writing operations for the dummy cell, respectively. The dummy cell can commence to detect the exact time when it completes the given operation. The outputs and internal data of dummy cell are amplified using the sense amplifier and they are transferred to Completion generator. Completion generator compares both the outputs and the internal value of dummy cell in order to generate the completion signal for handshaking protocol.

The proposed procedure for generating a completion signal can be described as follows. A dummy cell internally stores '1' by setting dummy cell before the read operation commence, that is, pre-writing operation. On starting the read operation, the access transistor of the dummy cell is activated and the stored data and its complementary data, D and $\overline{D}$, in dummy cell are transferred to the write completion generator. When the word-line is chosen and the write enable $\overline{WE}$ is high, Di and $\overline{Di}$ are amplified and finally, SB and SBb are obtained. The read completion signal generator compares two input signals, SB and SBb. Figure 7 shows the read completion signal generator. A read completion signal generator generates the completion signal when received data signals from the dummy cell, SB and SBb are logical complement.

In the case of write operation, the procedure of generating a completion signal as follows. The internal value of dummy cell is reset to zero before the write operation. When the memory write operation commences, the input data signals, Di and $\overline{Di}$ are transferred on the bit-lines to dummy cell and they are simultaneously transferred to a write completion signal generator. They are stored in the dummy cell and the outputs, D and $\overline{D}$ are transferred to the input of sense amplifiers to generate SD and SDb. The input signals, Di and $\overline{Di}$ and the outputs of dummy cell, D and $\overline{D}$ are compared to detect the completion time of write operation. A write completion generator generates the write completion signal when the amplified internal data signals and the transferred input signals are matched. The sum of latencies for dummy cell and completion signal generator is larger than that of the memory cell. The completion signal is generated at the end of memory operation. Then, it forces the handshaking controller to transit the acknowledge signal to *high*. The asynchronous processor indicates the completion of memory access by transiting the request signal to *low*. Handshake control transits acknowledge signal to *low* in order to return to initial state. Consequently, the timing of handshak-

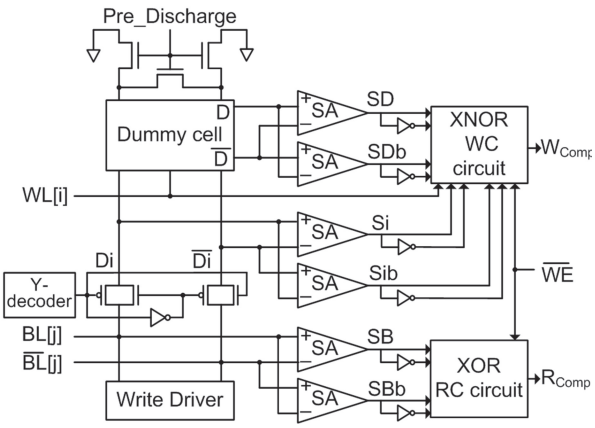

Fig. 6 The connection of between the dummy cell and the read/write completion signal generator.

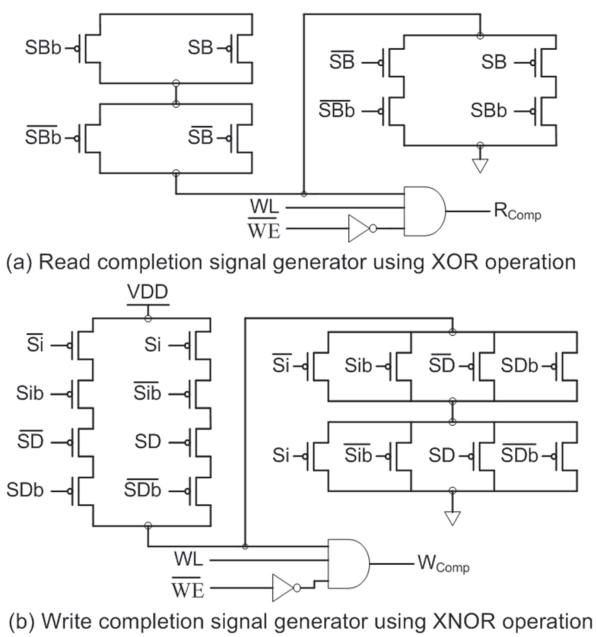

Fig. 7 Completion signal generator including (a) Read completion generator using XOR operation, (b) Write completion generator using XNOR operation.

ing protocol is satisfied using the proposed dummy cell and Completion generator.

The proposed method has another advantage to construct different types of asynchronous memory systems employing bounded delay model without significant decrease in performance. We divide the whole memory cell arrays into the multiple regions that have same maximum delay bound that is referenced as a delay element for each memory region. The bit-line delay for the corresponding memory cell array is obtained using the R-C modeling of bit-line delay that is described in Sect. 2.2. Then, the self-timed SRAM based on bounded delay model can be easily realized by replacing the proposed completion generator circuitry including the dummy cell to the matched delay. It also requires the addition work to add the other matched delay that has fairly large safety margin for the worst-case latency of the memory read or write operation. Even though this self-timed SRAM based on bounded delay model should trim the size of the matched delay for different fabrication processes, it has advantage in circuit area. However, the performance of this self-timed SRAM can be maintained as an average-case performance owing to the proposed memory segmentation technique.

4. Simulation Results

We construct memory system based on the proposed memory segmentation technique and the proposed completion generating circuitry. We make a bit-line load model for the memory cell array using R-C modeling presented by Elmore [12] for simulation. The parameters of R-C model are extracted using target process, Hynix 1P5M CMOS technology [13]. Then, we measure the bit-line delay of the memory cell arrays. This model provides efficient way to divide memory system according to the bit-line delay. We perform the layout of the proposed circuitry including dummy cell and completion generator using Cadence tool. We perform LPE (layout parasitic extraction) to extract the load capacitance using Synopsys StarRCXT. Then, we obtained the post-layout simulation results from the segmented memory region according to the bit-line delay time and handshaking circuitry for the asynchronous memory system. The simulation result completely meets the timing assumption of handshaking protocol with regardless of the given test sequences. We divide memory array into multiple blocks that have same size of memory cell array as shown in Fig. 8. Row decoder contains the multiplexers for selecting these memory blocks. The number of memory blocks depends on the size of the memory capacity. Then, we apply the proposed memory segmentation technique to each memory block. Each memory region has single dummy cell to generate the completion signal. We synthesize 8 MB SRAM module consists of sixteen separate 512K memory blocks. Each memory block comprises multiple memory region that has different size of the memory cell array. One byte of memory is accessed by supplying a 4-bit region number, a 10-bit row address, and a 9-bit column address. The process parameters that are used for the modeling for bit-line delay time are extracted from Hynix 0.35-μm CMOS technology [13]. Then, the proposed memory system is synthesized using same standard CMOS library.

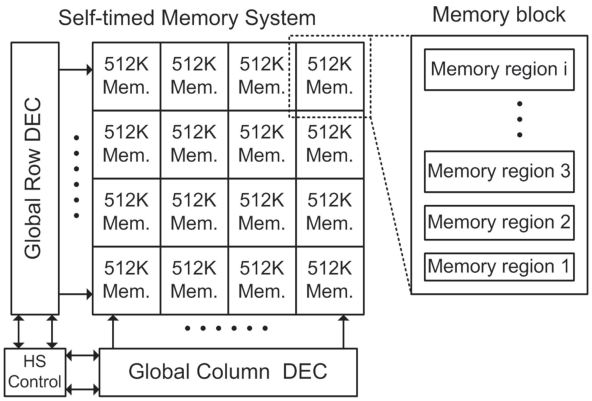

Fig. 8 The proposed 8M SRAM architecture having 16 self-timed memory regions.

Table 1 Comparison results between SRAMs employing different dummy cell placement and memory segment techniques.

Memory segmentation	Num. of region	Maximum write completion time	Area overhead
Max. bit-line delay	1	14.1 ns	0.2%
Min. bit-line delay	1	8.2 ns	37%
same region size [10]	4	10.1 ns	1.6%
	8	9.4 ns	2.1%
Proposed method	4	8.8 ns	1.0%
	8	8.5 ns	1.4%

For the fair comparison, we implement four different kinds of memory system as depicted in Fig. 2. The first is based on maximum bit-line delay and it has only one dummy cell to detect completion time as shown in Fig. 2 (a). On the contrary, every word-line in second one has dummy cell as shown in Fig. 2 (b). In addition, there are two memory systems employing memory segmentation techniques. The one is based on the size of the memory array and the other is based on its latency of the memory cell. The proposed method consists of multiple memory region having same bit-line delay, not the size of memory cell array. Table 1 shows the post-layout simulation results for these four different memory systems. It shows the write completion time and area overhead according to the dummy cell configuration techniques.

The first one is the smallest among all counterparts and the second one shows the fastest performance. As shown Table 1, the first provides the area efficient way, however, it suffers from the longest latency, 14.1 ns. Therefore, it is not applicable for the high-throughput embedded processors. The second one shows that the minimum latency for generating a write completion signal is 8.2 ns. However, the circuit area is significantly increased by the number of word-lines.

On the contrary, the two memory systems employing

the memory segmentation technique shows the better trade-off between the throughput and the area overhead. The one was presented by V. Sit [10] and the other is the proposed implementation in this paper. However, there is significant difference with respect to performance although both of them are divided into same number of memory regions.

When we apply dividing factor 4 and 8 to the memory system proposed by V. Sit, the average time for write completion are 10.1 ns and 9.4 ns, respectively. On the contrary, they are reduced to 8.8 ns and 8.5 ns in the proposed memory system having same dividing factor, respectively. This results indicates that the proposed memory segmentation method shorten the average memory access time over 15% comparing to the work presented by V. Sit [10]. Indeed, this result when memory system consists of 8 segmented memory region is close to the result for second one. The difference is only 0.3 ns. It notes that small number of segmentation is enough to increase average speed by the variable bit-line load concept. From the benchmark simulation, we obtained the minimum and maximum memory access times including handshaking timing that are 4.8 ns and 11.2 ns when we divide the memory cell array to 8 regions. The time difference between the regions is high to prove why we divide the memory cell array into multiple regions according to the delay time, not the size of memory cell array. In addition, this result shows that the memory access time of the proposed self-timed SRAM system significantly relies on how the data is allocated in different memory regions.

The number of memory regions is another important factor to evaluate the performance of segmented memory system. Figure 9 shows the relationship between the number of memory regions and the decreasing rate of write completion time, T_W. This result notes that the write completion time is decreasing as increasing number of memory segmentation. However, the completion time is indeed decreasing slowly. As shown in Fig. 9, the write completion time is saturated when the 512K memory system is divided into more than eight regions in two cases.

The other important thing is the trade-off between the memory access time and the circuit area. A dummy cell including pre-writing and pre-reading consists of 25 transistors. As increasing the number of memory regions, the area significantly increased. For the same reason, the latencies of the memory systems shall be decreased. As shown in Fig. 9, the latencies of two different memory systems getting close as the number of memory regions increases as increasing number of memory regions. The reason is that the boundary of memory regions is getting closed as dividing memory system into smaller piece.

Figure 10 shows the delay reduction rates on two memories divided into from 2 to 100 memory regions. However, the delay is saturated after 16 memory regions. If the number of memory segment is increased by more than 32, the memory access time does not decrease, rather increased owing to the increased dummy cell arrays. Therefore, it becomes trade-off between circuit area and performance to construct 8 MB SRAM module with sixteen separate 512K memory blocks.

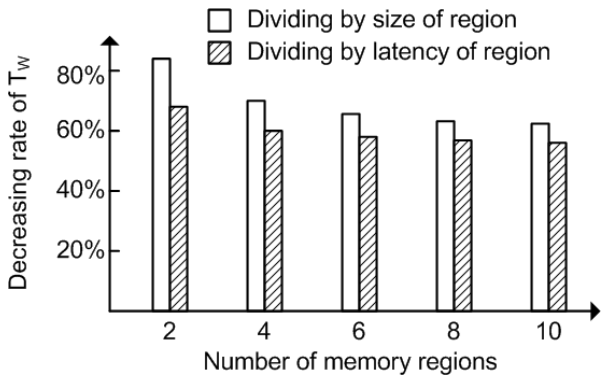

Fig. 9 The comparison results for the write completion time according to the number of memory region between the conventional memory segmentation and the proposed one.

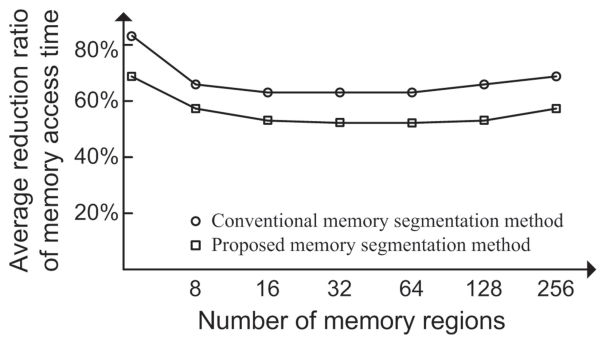

Fig. 10 The comparison results for delay reduction rates according to the increased number of memory segmentations.

5. Conclusions

This paper presents a new self-timed static RAM system. It supports read and write operations by communicating with other asynchronous processor. Unlike the conventional SRAM design, it consists of multiple memory regions having different size of the memory cell array. Since the proposed memory segmentation technique is based on the true completion time obtained from the R-C modeling of bit-line delay for memory cell array, it shows higher performance than its counterpart. In addition, we proposed the completion generation circuitry for generating a completion signal for read and write operation. The proposed techniques apply to 8 MB SRAM that consists of 16 512 KB memory regions to evaluate the performance. We analyzed the impact of the number of segmentation on the performance and area overhead. The simulation results shows that the proposed techniques delivers better trade-off between performance and area overhead according to the number of memory segments. It can reduce the memory access delay by 40% and 15% compared with conventional SRAM employing maximum delay without memory segmentation and the one employing memory segmentation with same number of mem-

ory regions. The proposed 8 MB SRAM and the proposed techniques can be easily applied to various asynchronous systems. In addition, the proposed SRAM can be changed to different types of asynchronous SRAM with bounded delay model by replacing the dummy cell to the fixed delay element since we construct memory region based on the size of latency, not the size of the memory cell array. The proposed technique can be applicable for the design of an asynchronous cache memory.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (20009-0077491). This work was also supported by the grant of the Korean Ministry of Education, Science and Technology (The Regional Core Research Program/Chungbuk BIT Research-Oriented University Consortium).

References

[1] J. Sparso and S. Furber, Principles of Asynchronous Circuit Design: A Systems Perspective, Kluwer Academic Publishers, 2001.
[2] J.H. Lee, Y.H. Kim, and K.R. Cho, "Design of a fast asynchronous embedded CISC microprocessor, A8051," IEICE Trans. Electron., vol.E87-C, no.4, pp.527–534, April 2004.
[3] A.J. Martin, A. Lines, R. Manohar, M. Nystrom, P. Penzes, R. Southworth, U. Cummings, and T.K. Lee, "The design of an asynchronous MIPS R3000 microprocessor," Proc. 7th Conf. Advance Research in VLSI, pp.164–181, Sept. 1997.
[4] J.D. Garside, W.J. Bainbridge, A. Bardsley, D.M. Clark, D.A. Edwards, S.B. Furber, J. Liu, D.W. Lloyd, S. Mohammadi, J.S. Pepper, O. Petlin, S. Temple, and J.V. Woods, "AMULET3i — An asynchronous system-on-chip," Proc. ASYNC 2000, pp.162–175, April 2000.
[5] M.C. Chang and D.S. Shiau, "Design of an asynchronous pipelined processor," Proc. ICCCAS 2008, pp.1093–1096, May 2008.
[6] J.H. Lee and K.R. Cho, "Design of a high performance self-timed ARM9 processor," IEICE Electronics Express, vol.5, no.3. pp.87–93, Jan. 2008.
[7] J. Dama and A. Lines, "GHz asynchronous SRAM in 65 nm," Proc. ASYNC 2009, pp.85–94, 2009.
[8] D. Hormdee and J.D. Garside, "AMULET3i cache architecture," Proc. ASYNC 2001, pp.152–161, March 2001.
[9] J.D. Garside, S. Temple, and R. Mehra, "The AMULET2e chche system," Proc. 2nd ASYNC, pp.208–217, March 1996.
[10] V. Sit, C.S. Choy, and C.F. Chan, "A four-phase handshaking asynchronous static RAM design for self-timed system," IEEE J. Solid-State Circuits, vol.34, no.1, pp.90–96, Jan. 1999.
[11] B. Keeth, B. Johnson, R.J. Baker, and F. Lin, DRAM Circuit Design: Fundamentals and High-speed Topics, John Wiley & Sons, 2007.
[12] Z. Quming and K. Mohanram, "Elmore model for energy estimation in RC trees," Proc. DAC 2006, pp.965–970, July 2006.
[13] Hynix Co., Hynix 0.35-μm Logic Technology: Spice model Document, April 2006.

Je-Hoon Lee received the B.S. and M.S. degrees in Computer and Communication Engineering from Chungbuk National University, Cheongju, Korea in 1999 and 2001, respectively. He received Ph.D. in Electrical Engineering from Chungbuk National University in 2005. From 2005 to 2006, he was a visiting scholar at Univ. of Southern California, USA and from 2007 to 2008, he was a visiting scholar at Murdoch University, Australia. From 2006 to 2009, he was an assistant Professor in Chungbuk National University. He is currently an assistant Professor in Div. of Electronics and Information Communication Engineering of Kangwon National University, Korea. His research interests in the digital circuit design for high-speed and low power and the computer architecture. He is a member of Institute of Electrical and Electronics Engineer (IEEE).

Young-Jun Song received the B.S. and M.S. in computer and communication engineering from Chungbuk National University, Korea in 1994, 1996 respectively and also received Ph.D. in computer and communication engineering from Chungbuk National University, Korea in 2004. Since then, he has worked as an invited vice professor in the Chungbuk BIT Research-Oriented University Consortium, Chungbuk National University, Korea. His main research interests include face recognition, pattern recognition, and computer vision.

Sang-Choon Kim received his B.S. degree in computer science from Hanvat National University in 1986. He received M.S. degree in computer science from Cheongju University in 1989. He received Ph.D. degree in computer science from Chungbuk National University in 1999. From 1983 to 2001, he worked in ETRI. Since 2001, he has been on the faculty of the Kangwon National University at department of Information and Communication Engineering. His research interests include application security, security evaluation, IPTV security and network security.