Fully Programmable Memory BIST for Commodity DRAMs

Ilwoong Kim, Woosik Jeong, Dongho Kang, and Sungho Kang

To accomplish a high-speed test on low-speed automatic test equipment (ATE), a new instruction-based fully programmable memory built-in self-test (BIST) is proposed. The proposed memory BIST generates a high-speed internal clock signal by multiplying an external low-speed clock signal from an ATE by a clock multiplier embedded in a DRAM. For maximum programmability and small area overhead, the proposed memory BIST stores the unique sets of instructions and corresponding test sequences that are implicit within the test algorithms that it receives from an external ATE. The proposed memory BIST is managed by an external ATE on-the-fly to perform complicated and hard-to-implement functions, such as loop operations and refresh-interrupts. Therefore, the proposed memory BIST has a simple hardware structure compared to conventional memory BIST schemes. The proposed memory BIST is a practical test solution for reducing the overall test cost for the mass production of commodity DDRx SDRAMs.

Keywords: Built-in self-test, DRAM, low cost, at-speed test.

I. Introduction

As the capacity and density of semiconductor memories have rapidly increased, maintaining acceptable yields and quality has become the most critical challenges in semiconductor memory manufacturing. Most commodity dynamic random-access memory (DRAM) is tested with a multitude of test algorithms in a mass production test, by using external automatic test equipment (ATE). Recently, to test high-density and high-speed commodity memories, such as Giga-bit DDR3 SDRAM with at-speed, a high-end ATE platform is required; however, such a platform comes at an extremely high cost. The following are some of the major requirements of a memory BIST for commodity DRAMs: high fault coverage, high test frequency, high diagnostic capability, and small area-overhead.

Most of the previous researches on memory BIST techniques have been developed for embedded memories on system on a chip (SOC) [1]–[4], but the test algorithms for embedded memories are inappropriate for the mass production testing of commodity DRAMs. Specifically, programmability is mandatory, not optional; programming is used to build various types of test algorithms for a mass production test, as well as to determine the root cause(s) when critical faults occur, to screen them out. To satisfy the major requirements of a memory BIST for commodity DRAMs (high fault coverage, high test frequency, and small area-overhead), an instruction-based programmable memory BIST is the most suitable for commodity DRAMs.

This paper proposes a new memory BIST, presents experimental data, and provides conclusions on the proposed BIST.

Manuscript received Jan. 16, 2015; revised Apr. 4, 2015; accepted Apr. 21, 2015.

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2015R1A2A1A13001751).

Ilwoong Kim (woong@soc.yonsei.ac.kr), Dongho Kang (fourier2@soc.yonsei.ac.kr), and Sungho Kang (corresponding author, shkang@yonsei.ac.kr) are with the Department of Electrical and Electronic Engineering, Yonsei University, Seoul, Rep. of Korea.

Woosik Jeong (woosik.jeong@sk.com) is with the Product Development Group, Hynix Semiconductor Inc. Icheon, Rep. of Korea.

http://dx.doi.org/10.4218/etrij.15.0115.0040

II. Proposed Memory BIST

The main concept of the proposed memory BIST is to maximize the programmability and reusability of low-performance ATEs, which are commonly used in the mass production testing of commodity DRAMs. Even if a commodity DRAM has a memory BIST, an external ATE is required to provide input signals and automatically acquire test results.

Figure 1 shows a block diagram of the proposed memory BIST, which consists of a sequence buffer, instruction buffer, and instruction decoder. In the proposed memory BIST, a built-in redundancy analysis (BIRA), which consists of fault collection from the proposed memory BIST and redundancy analysis to extract the featured repair solution, is used for wafer-level test and repair [5].

Test algorithms for a commodity DRAM generated by an ATE generally comprise five unit operations as follows: command instructions, loop operations, refresh-interrupts for managing various types of retention tests, address operations, and data operations. To guarantee a high level of programmability for test algorithms, within the confines that an ATE is able to provide, a memory BIST for commodity DRAMs must support multiple levels of loop operations and various types of timers for refresh-interrupts for testing the retention time of DRAMs. However, in the proposed memory BIST, the loop operations and refresh-interrupts are managed by an external ATE to guarantee a high level of programmability and reduce the hardware required for the two functions. Therefore, the proposed memory BIST handles only three unit operations: a command instruction, address operation, and data operation; there are also two types of buffer: an instruction buffer and a sequence buffer. An instruction buffer contains a command instruction, an address operation, and a data operation.

The proposed memory BIST is designed to perform at-speed tests with a low-frequency ATE. The external test clock frequency of the required low-frequency ATE can be multiplied by four or eight to generate a faster internal clock frequency for the proposed memory BIST by using modified DLL logic. For this reason, several internal instructions must be delivered from the external ATE in an external clock cycle so that a sequence buffer pointer, which contains pointers for instruction buffers to be executed sequentially, is transferred from the ATE to the proposed memory BIST. Subsequently, a series of instructions is able to be executed with fast internal clocks.

Figure 2 shows the bit fields for a sequence buffer and an instruction buffer. The proposed memory BIST has only 31 instruction buffers (from #0 to #30) and 32 sequential buffers. These are sufficient for storing complicated test algorithms for DRAM mass production.

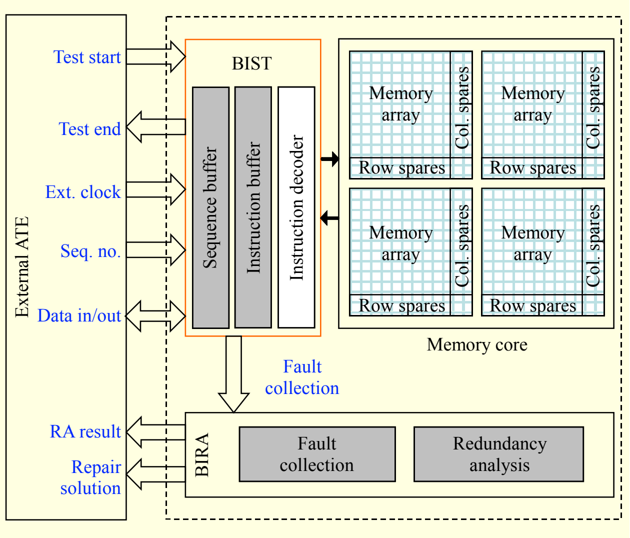


Fig. 1. Block diagram of proposed memory BIST.

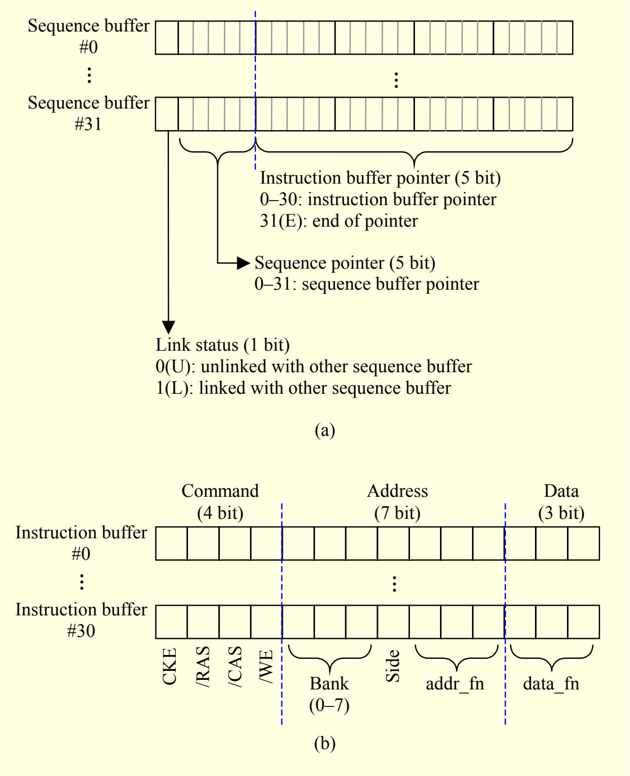
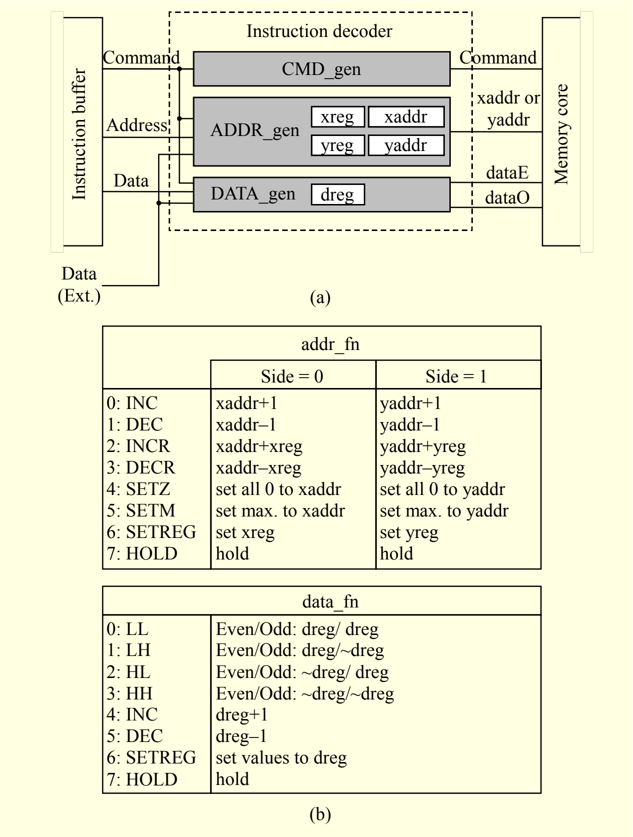


Fig. 2. Bit fields for (a) sequence buffer and (b) instruction buffer.

Each sequence buffer of Fig. 2(a) consists of twenty-six bits. The first bit of a sequence buffer represents the buffer's *link* status with one of the other remaining buffers. The next five bits represent the sequence buffer pointer. The final twenty bits represent instruction buffer pointers, which are to be executed



addr_fn		
	Side = 0	Side = 1
0: INC	xaddr+1	yaddr+1
1: DEC	xaddr–1	yaddr–1
2: INCR	xaddr+xreg	yaddr+yreg
3: DECR	xaddr–xreg	yaddr–yreg
4: SETZ	set all 0 to xaddr	set all 0 to yaddr
5: SETM	set max. to xaddr	set max. to yaddr
6: SETREG	set xreg	set yreg
7: HOLD	hold	hold

data_fn	
0: LL	Even/Odd: dreg/ dreg
1: LH	Even/Odd: dreg/~dreg
2: HL	Even/Odd: ~dreg/ dreg
3: HH	Even/Odd: ~dreg/~dreg
4: INC	dreg+1
5: DEC	dreg–1
6: SETREG	set values to dreg
7: HOLD	hold

Fig. 3. (a) Instruction decoder and (b) function descriptions of addr_fn and data_fn.

sequentially.

An instruction buffer pointer is comprised of five bits and represents a valid instruction buffer pointer when its value is from 0 to 30, whereas it represents the end of the instruction buffer when its value is 31.

If the first bit of sequence buffer A is set to “1” and the next five bits point to sequence buffer B, then this means that all of the valid instructions of sequence buffer B will be executed after completing those of sequence buffer A.

Each instruction buffer consists of fourteen bits for DDR3 SDRAM. In Fig. 2(b), the first four bits of the instruction buffer represent the command instruction for DDR3 SDRAM, such as Active, Write, Read, and Precharge. The next seven bits represent the address operations; the fifth to seventh bits represent the bank address. The eighth bit selects whether x addresses or y addresses are to be modified by the address function. The next three bits represent the address functions, and the last three bits represent the data function.

Figure 3 shows the structure of the instruction decoder of the proposed memory BIST and provides descriptions of the address function and data function for the instructions. The instruction decoder decodes each bit of an instruction buffer and generates a command; x address and y address; and *dataE/dataO* for the memory core.

The address generator of the proposed memory BIST (that is, *ADDR_gen* of Fig. 3(a)) has two address registers, *xaddr* and *yaddr*, to hold a current x address and y address, respectively. It also has two registers, *xreg* and *yreg*, which hold address values to be added or subtracted into address registers by INCR or DECR functions of the *addr_fn* of Fig. 3(b).

The data generator of the proposed memory BIST has two data registers (*dreg*) to hold current data values, *dataE* for a clock rising edge and *dataO* for a clock falling edge.

III. Communication between Memory BIST and ATE

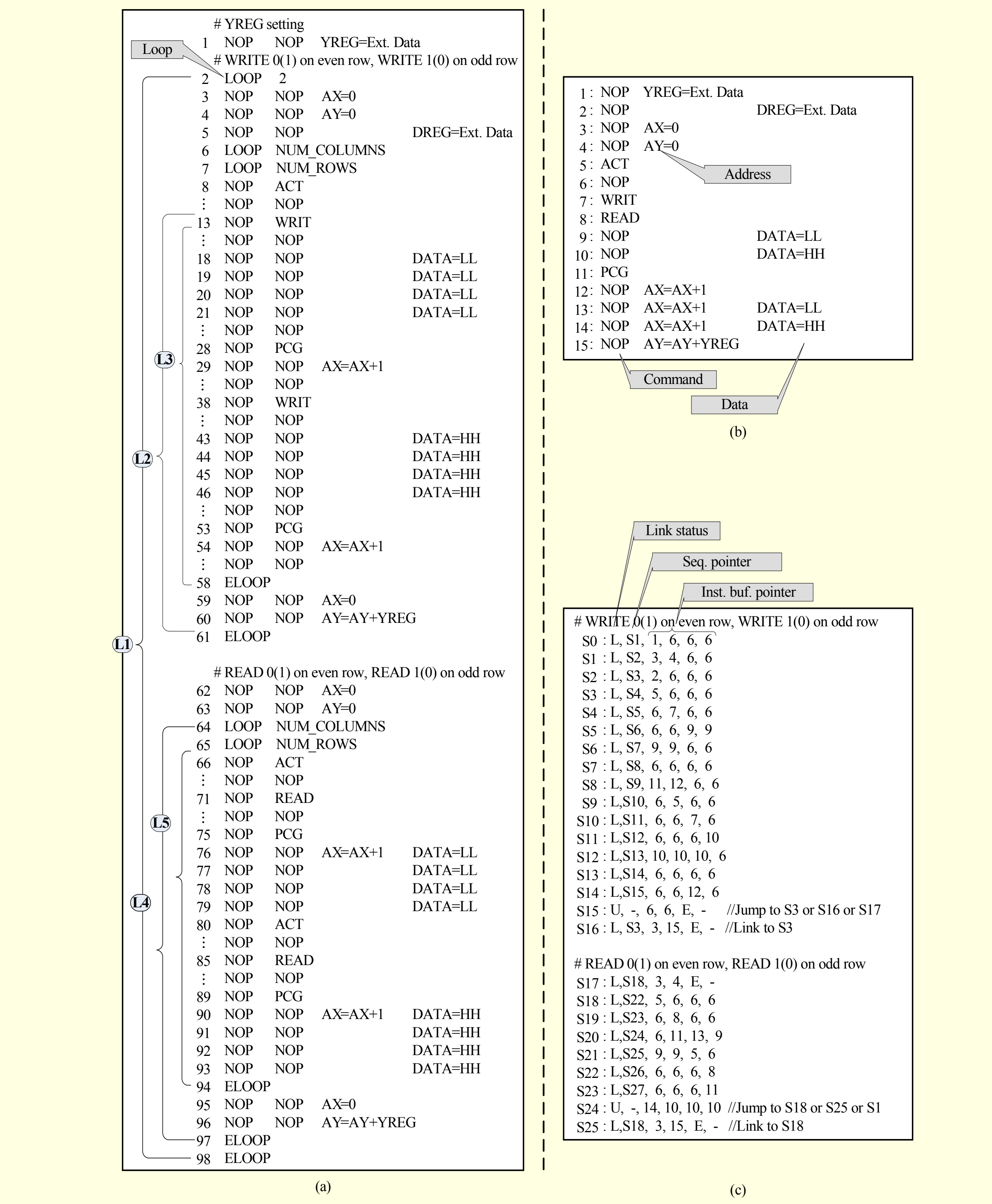
Figure 4(a) shows an example of a scan algorithm with checkerboard data background for a DDR3 SDRAM, which includes five loops (L1–L5) that collectively form a three-level loop (L1). Figures 4(b) and 4(c) represent a set of unique instructions (which are saved in the instruction buffers) and a set of sequence buffers (to construct the same test algorithm described in Fig. 4 (a)), respectively.

Figure 5 shows an example of a timing diagram between an ATE and the proposed memory BIST; it is assumed that the internal clock frequency is eight times faster than the external clock frequency from the ATE. To construct the same test algorithm as the one shown in Fig. 4, the external ATE generates sequence pointers in every clock cycle, such as is shown in Fig. 5. If a generated sequence number is synchronized with the external low frequency clock signal, then the internal instructions for the sequence buffer are executed to the memory core. On every rising edge of the external clock, the sequence buffer pointer value is stored into a sequence register of the memory BIST. At the same time, external data is stored in an external data register for setting the *xreg*, *yreg*, or *dreg* of the memory BIST.

When the Link status is set to “0” and an instruction pointer of a sequence buffer reaches the “End of pointer” or the fourth instruction pointer, the memory BIST retrieves the next sequence buffer pointer value from the sequence register; otherwise, the sequence pointer value is ignored.

In this case, the external ATE generates a SEQ0 once and consecutive SEQ3s. SEQ0 indicates S0 in the first sequence buffer of Fig. 4(c). S0 is linked with S1, which is the start point of L1. S1–S14 are sequentially linked, and S14 is linked with S15. However, S15 is not linked with any other sequence buffer. Therefore, the proposed memory BIST retrieves SEQ3 from the sequence register and jumps to S3, which is the start point of L2 and L3.

The external ATE manages the sequence buffer pointers and internal register setting; it is sufficiently knowledgeable with the *loop control* and *refresh-interrupt control* to build



```
     # YREG setting
 1   NOP    NOP    YREG=Ext. Data
     # WRITE 0(1) on even row, WRITE 1(0) on odd row
 2   LOOP   2
 3   NOP    NOP    AX=0
 4   NOP    NOP    AY=0
 5   NOP    NOP                   DREG=Ext. Data
 6   LOOP   NUM_COLUMNS
 7   LOOP   NUM_ROWS
 8   NOP    ACT
 ⋮   NOP    NOP
13   NOP    WRIT
 ⋮   NOP    NOP
18   NOP    NOP                   DATA=LL
19   NOP    NOP                   DATA=LL
20   NOP    NOP                   DATA=LL
21   NOP    NOP                   DATA=LL
 ⋮   NOP    NOP
28   NOP    PCG
29   NOP    NOP    AX=AX+1
 ⋮   NOP    NOP
38   NOP    WRIT
 ⋮   NOP    NOP
43   NOP    NOP                   DATA=HH
44   NOP    NOP                   DATA=HH
45   NOP    NOP                   DATA=HH
46   NOP    NOP                   DATA=HH
 ⋮   NOP    NOP
53   NOP    PCG
54   NOP    NOP    AX=AX+1
 ⋮   NOP    NOP
58   ELOOP
59   NOP    NOP    AX=0
60   NOP    NOP    AY=AY+YREG
61   ELOOP

     # READ 0(1) on even row, READ 1(0) on odd row
62   NOP    NOP    AX=0
63   NOP    NOP    AY=0
64   LOOP   NUM_COLUMNS
65   LOOP   NUM_ROWS
66   NOP    ACT
 ⋮   NOP    NOP
71   NOP    READ
 ⋮   NOP    NOP
75   NOP    PCG
76   NOP    NOP    AX=AX+1        DATA=LL
77   NOP    NOP                   DATA=LL
78   NOP    NOP                   DATA=LL
79   NOP    NOP                   DATA=LL
80   NOP    ACT
 ⋮   NOP    NOP
85   NOP    READ
 ⋮   NOP    NOP
89   NOP    PCG
90   NOP    NOP    AX=AX+1        DATA=HH
91   NOP    NOP                   DATA=HH
92   NOP    NOP                   DATA=HH
93   NOP    NOP                   DATA=HH
94   ELOOP
95   NOP    NOP    AX=0
96   NOP    NOP    AY=AY+YREG
97   ELOOP
98   ELOOP
```

(a)

```
 1 : NOP   YREG=Ext. Data
 2 : NOP                    DREG=Ext. Data
 3 : NOP   AX=0
 4 : NOP   AY=0
 5 : ACT
 6 : NOP
 7 : WRIT
 8 : READ
 9 : NOP                    DATA=LL
10 : NOP                    DATA=HH
11 : PCG
12 : NOP   AX=AX+1
13 : NOP   AX=AX+1          DATA=LL
14 : NOP   AX=AX+1          DATA=HH
15 : NOP   AY=AY+YREG
```

(b)

```
# WRITE 0(1) on even row, WRITE 1(0) on odd row
 S0 : L, S1,  1,  6,  6,  6
 S1 : L, S2,  3,  4,  6,  6
 S2 : L, S3,  2,  6,  6,  6
 S3 : L, S4,  5,  6,  6,  6
 S4 : L, S5,  6,  7,  6,  6
 S5 : L, S6,  6,  6,  9,  9
 S6 : L, S7,  9,  9,  6,  6
 S7 : L, S8,  6,  6,  6,  6
 S8 : L, S9, 11, 12,  6,  6
 S9 : L,S10,  6,  5,  6,  6
S10 : L,S11,  6,  6,  7,  6
S11 : L,S12,  6,  6,  6, 10
S12 : L,S13, 10, 10, 10,  6
S13 : L,S14,  6,  6,  6,  6
S14 : L,S15,  6,  6, 12,  6
S15 : U,  -,  6,  6,  E,  -     //Jump to S3 or S16 or S17
S16 : L, S3,  3, 15,  E,  -   //Link to S3

# READ 0(1) on even row, READ 1(0) on odd row
S17 : L,S18,  3,  4,  E,  -
S18 : L,S22,  5,  6,  6,  6
S19 : L,S23,  6,  8,  6,  6
S20 : L,S24,  6, 11, 13,  9
S21 : L,S25,  9,  9,  5,  6
S22 : L,S26,  6,  6,  6,  8
S23 : L,S27,  6,  6,  6, 11
S24 : U,  -, 14, 10, 10, 10  //Jump to S18 or S25 or S1
S25 : L,S18,  3, 15,  E,  -   //Link to S18
```

(c)

Fig. 4. Example of test algorithm for DDR3 SDRAM: (a) scan algorithm with checkerboard data backgraound, (b) instruction buffer, and (c) sequence buffer.

complicated test algorithms. Therefore, complicated control functions are not required in the proposed memory BIST, which makes the structure of the proposed memory BIST very simple. In addition, the required area overhead for saving test

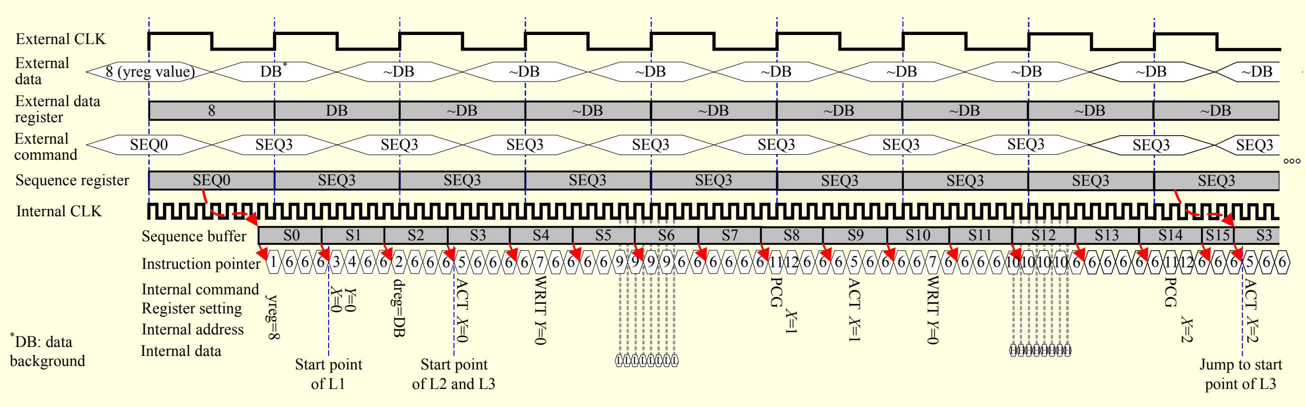


Fig. 5. Example of timing diagram between ATE and memory BIST.

algorithms into the memory BIST is very small, all the while guaranteeing full programmability and high-speed testing with a low-speed ATE.

IV. Experimental Results

The proposed memory BIST was designed with Verilog HDL and synthesized with 0.13 μm CMOS technology. Table 1 shows a performance comparison of the proposed memory BIST with previous researches [1]–[4]. In Table 1, the second row represents the area overhead of each memory BIST. There is a trade-off between the area overhead and the programmability. In a memory BIST for a commodity DRAM, a very high programmability has to be provided so as to be robust to the ever-changing conditions of mass production testing. Although the area overhead of the proposed memory BIST is larger than those of [1]–[3], it shows an approximately three times smaller area overhead as compared with that of [4] and provides a very high programmability as shown by the third row of Table 1. Unlike the previous researches [2]–[4], the timing control for a retention test can be managed by an ATE without extra hardware and additional test channels. This fully enables various kinds of retention testing algorithms. Therefore, typical mass-production test algorithms (for example, Scan, MATS+, Moving Inversion, GalRow, GalCol, March LAd [6], and so on) for a commodity DRAM can be supported. Using dual instruction decoders, the clock speed of the proposed memory BIST is 1.25 ns, and it can perform at-speed tests for DDR3 SDRAM. Most current wafer-level ATEs, such as MT6060 (YOKOGAWA) and T5377 (Advantest), are able to generate clock frequencies above 100 MHz. If a DDR3 SDRAM adopts the proposed memory BIST and generates an eight times faster internal clock frequency than the external clock speed, then the memory is able be tested at up to 800 MHz (1.6 Gbps). Without high-speed ATEs, the proposed memory BIST can provide a practical solution for testing DRAM with at-speed.

Table 1. Performance comparison.

Feature	[1]	[2]	[3]	[4]	Proposed
Area (gates)	~2.5 K	> 300 K	7.9 K	50 K	15 K
Programmability	Medium	Medium	High	High	Very high
Retention test support	N/A	Partial	Partial	Partial	Full
Testing freq.	N/A	N/A	400 MHz	972 MHz	800 MHz

V. Conclusion

In this paper, an instruction-based fully programmable high-speed memory BIST solution for commodity DRAM mass production tests with low-speed ATEs is proposed. The proposed memory BIST provides very high programmability with reasonable area overhead, making it is possible to encompass typical mass-production test algorithms for DRAMs. The proposed memory BIST can be a viable solution for maintaining yield and quality of commodity DDRx SDRAMs without using high-end ATEs.

References

[1] B. Ismet, O. Caty, and Y. Wong, “Highly Configurable Programmable Built-In Self Test Architecture for High-Speed Memories,” *IEEE VLSI Test Symp.*, Palm Springs, CA, USA, May 1–5, 2005, pp. 21–26.

[2] S. Boutobza et al., “Programmable Memory BIST,” *IEEE Int. Test Conf.*, Austin, TX, USA, Nov. 8–10, 2005, pp. 1155–1164.

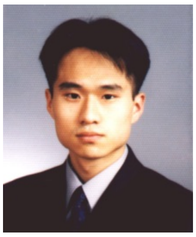
[3] Y. Park et al., “A Flexible Programmable Memory BIST for Embedded Single-Port Memory and Dual-Port Memory,” *ETRI*

J., vol. 35, no. 5, Oct. 2013, pp. 808–818.

[4] M. Kume et al., “Programmable At-Speed Array and Functional BIST for Embedded DRAM LSI,” *IEEE Int. Test Conf*., Charlotte, NC, USA, Oct. 26–28, 2004, pp. 988–996.

[5] T. Han et al., “High Repair Efficiency BIRA Algorithm with a Line Fault Scheme,” *ETRI J*., vol. 32, no. 4, Aug. 2010, pp. 642–644.

[6] Z. Al-Ars and A.J. van de Goor, “Static and Dynamic Behavior of Memory Cell Array Spot Defects in Embedded DRAMs,” *IEEE Trans. Comput*., vol. 52, no. 3, Mar. 2003, pp. 293–309.



Ilwoong Kim received his BS degree in control and instrumentation engineering from Kwangwoon University, Seoul, Rep. of Korea, in 2001 and his MS degree in electrical and electronic engineering from Yonsei University, Seoul, Rep. of Korea, in 2007, respectively. He is currently pursuing his PhD degree in electrical and electronics engineering at Yonsei University. His current research interests include built-in self-test and low-cost fault management for commodity DRAMs; TSV repair schemes and algorithms; VLSI/SOC design and testing; and design for testability.

Woosik Jeong received his BS degree in control and instrumentation engineering and his MS degree in electronics engineering from Korea University, Seoul, Rep. of Korea, in 1997 and 1999, respectively. He received his PhD degree in electrical and electronics engineering from Yonsei University, Seoul, Rep. of Korea, in 2011. Since 1999, he has been a product engineer with the DRAM Technology Development Division, SK hynix Semiconductor, Inc., Icheon, Rep. of Korea. His current research interests include memory testing, built-in self-testing, built-in self-repair, reliability, and very-large-scale integration design.

Dongho Kang received his BS degree in information and communication engineering from Myongji University, Yongin, Rep. of Korea, in 2013. He is currently pursuing his MS degree in electrical and electronics engineering at Yonsei University, Seoul, Rep. of Korea. His current research interests include reliable storage systems, built-in self-testing, TSV repair, redundancy analysis algorithms, reliability, and VLSI design.

Sungho Kang received his BS degree in control and instrumentation engineering from Seoul National University, Rep. of Korea, in 1986 and his MS and PhD degrees in electrical and computer engineering from the University of Texas at Austin, USA, in 1988 and 1992, respectively. From 1989 to 1992, he was a research scientist with the Schlumberger Laboratory for Computer Science, Schlumberger, Inc., USA. From 1992 to 1994, he was a senior staff engineer with Semiconductor Systems Design Technology, Motorola, Inc., USA. Since 1994, he has been a professor with the Department of Electrical and Electronic Engineering, Yonsei University, Seoul, Rep. of Korea. His current research interests include VLSI/SOC design and testing; design for testability; and design for manufacturability.

http://dx.doi.org/10.4218/etrij.15.0115.0040