Transitioning eMRAM from Pilot Project to Volume Production

Cyrille Dray[1], Khushal Gelda[2]
Benoit Nadeau-Dostie[3], Wei Zou[4], Luc Romain[3], Jongsin Yun[4], Harshitha Kodali[4], Lori Schramm[5], Martin Keim[6]

[1]Arm Sophia Antipolis, France; [2]Bengaluru, Karnataka, India
Siemens [3]Ottawa, Canada; [4]Wilsonville, USA; [5]Atlanta, USA; [6]Orlando, USA

***Abstract*—Embedded non-volatile RAM technology, and in particular Magneto-resistive RAM (MRAM), continues making great progress in read/write speed and cycling endurance, now rivaling traditional memory technologies. This advancement together with their low-energy consumption and non-volatility opens huge application markets. While stand-alone MRAM products are already being deployed, the first pilot applications for embedded MRAM are just starting to emerge. The availability of Design-for-Test (DFT) tools is key in the transition from low-volume technology exploration and pilot projects to high-volume production. In this paper we explore a new aspect of a Memory Built-In Self-Test (MBIST) tool for eMRAM, namely Error Correcting Code (ECC)-aware test and repair technology. This new MBIST capability provides the ability to make in-system, user-programmable trade-offs concerning the eMRAM repair resources. Having extra control over the repair resources in turn increases both manufacturing yield as well as the longevity of the product in its application.**

Keywords*—eMRAM, trimming, ECC, ECC tuning, ECC-aware test, memory fault injection, memory test, memory repair, memory yield, memory aging, DFT, MBIST

I. Introduction

The rapidly growing IoT market creates a huge demand for embedded non-volatile memories. For example, IoT devices [1], with AI technology [2], require high density and high performance, low power memory operation. However, scaling traditional eFlash memories has limitations going below 28nm due to its high cost. The industry offers several alternative embedded non-volatile memories such as MRAM, CBRAM (Conductive Bridge Random-Access Memory), FeRAM (Ferroelectric Random-Access Memory), ReRAM (Resistive Random-Access Memory), etc. Recent MRAM technology announcements by Samsung [3] and IBM [4] show very promising results. MRAMs manufactured at 14nm FinFET technology sustained a sub 100ns write speed with 1E14 cycles of endurance. This is very impressive compared to eFlash technology which offers micro-second to milli-second write speed with 1E5 cycling endurance [5]. It also promises downscaling capability to fill the embedded NVM market needs beyond the 28nm node. This type of high speed, high endurance, and high-density MRAM is expected to replace some of the cache memories in the near future.

In this Industrial Short Paper, we describe a new MBIST solution for eMRAM. This MBIST solution provides the same fully integrated implementation environment to build MBIST DFT logic for both SRAMs as well as MRAMs. Furthermore, much of a standard solution for SRAM transfers directly, so the same MBIST methodology can be used for testing both SRAM and MRAM. Accordingly, Section II introduces first the standard MBIST test and repair flow for SRAM. In Section III we will develop this flow into a production-ready test and repair flow for eMRAMs highlighting only the differences. For completeness, this section also summarizes eMRAM specific fault models and the necessary trimming used in the experimental setup outlined in Section IV. The focus in Section IV is however on "ECC-Aware Test and Repair". This combines the power of the ECC, which every emerging NVM requires, with the power of repair for the purpose of improving yield and durability of the memory in the product application. The data presented in Section IV clearly demonstrates the applicability of the ECC-aware test and repair technology to eMRAMs. Lastly, Section V concludes the paper.

II. Standard Memory BIST Overview

Figure 1 illustrates a widely used standard MBIST test and repair manufacturing flow. The same manufacturing flow can be used for SRAMs implementing redundancy (repair) or ECC logic. ECC is not typically used for SRAMs during manufacturing test, but it can be used during in-system testing to ensure tolerance of soft errors.

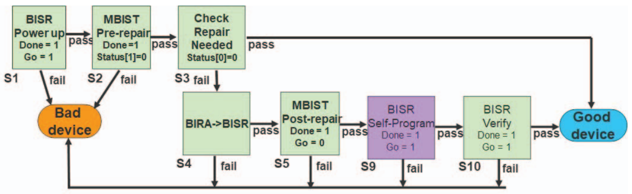


Fig. 1. MBIST test and repair manufacturing flow

The Built-in Self Repair (BISR) power-up step (S1) of the flow emulates a power-up event. The BISR chain is loaded with a repair solution previously calculated and stored in a fuse box. All spare resources of repairable memories are flagged as unused when executing the flow for the first time. The MBIST pre-repair step (S2) calculates a new repair solution based on the previous one. It tests all memories in the design and identifies failing memories, and in particular failing memories that cannot be repaired due to insufficient spares or the absence of any spare resources. If no such memory is found, the flow moves to the next step (S3); otherwise, the device is flagged as faulty.

Step S3 involves checking whether a memory with spare resources requires repair. If it does, the calculated repair solution is transferred to the BISR registers during the BIRA-> BISR step (S4). The MBIST post-repair step (S5) is then executed to confirm that the new repair solution is valid. Step S5 is similar to the pre-repair step, with the difference that no memory failures are allowed in any of the test patterns. It allows a complete test of the spares allocated during pre-repair.

If step S5 is successful, the BISR self-programming step (S9) compresses the repair data and programs it into the fuse box. The BISR Verify step (S10) involves the BISR verification function, which reads the data from the fuse box, decompresses it, and verifies that it matches the repair data stored in the BISR chain which was used as the input to step S9.

III. eMRAM Memory BIST Overview

This section contrasts the eMRAM MBIST flow later outlined in Figure 4 with the standard SRAM flow of the previous section. Here, we only focus on the essential differences. We will start with MRAM-specific defect models, then briefly describe trimming before we detail the ECC-aware test and repair technology and flow.

A. eMRAM-specific Fault Models

An MTJ (Magnetic Tunnel Junction) together with an access transistor make up the storage cell of the MRAM. The data (bit) is stored as a resistance that is determined by the bistable electron spin polarization in the free layer of an MTJ. Those polarities can switch back and forth by applying a forward or reverse bias onto the MTJ. Memory cells where resistance is stuck at a high or low resistance level are easily detectable based on traditional memory test patterns such as March tests. However, MRAMs have several unique defects that are difficult to capture with typical March tests.

The electron tunneling through the MgO layer highly depends on the quality of the oxide layer. Imperfections in the atomic layer near the interface, such as interface roughness and pin holes, are non-trivial to completely avoid. Those in turn can form a domain wall pinning site and provide a leakage path, while others can cause reliability issues [5]. Such reliability fails are difficult to capture because their fail signature will not be observable until the memory wears out. Voltage and temperature stress tests can accelerate wear-out in a reasonably short time but cannot be used to test a product without sacrificing its life span.

Repeating back-to-back operations, or so-called hammer tests, can apply moderate stress to detect some of the wear-out issues but are time-expensive tests. Hence, testing an entire memory array is a significant challenge. A solution to this problem is a new test flow, which first identifies the hammer test candidates based on their resistance level. It achieves high defect coverage without a significant test cost increase [7].

B. eMRAM Trimming

Trimming is the process of determining a reference value, which decides if the read (electrical) value must be interpreted as a logical 0 or as a logical 1. For MRAM, whose storage elements are resistance-based, trimming is done to determine a specific reference resistance. This reference value cannot be determined up-front with high confidence but must be learned, preferably on-chip by using fully autonomous circuitry for each memory and sense amplifier. This has been described for an earlier eMRAM experiment in [8]: Each row in Figure 2 indicates a memory array of the tested die with 8Mb granularity, except for the 1Mb instance in the first row; each column indicates different sample die. The color coding indicates the different computed trim values for the different samples.

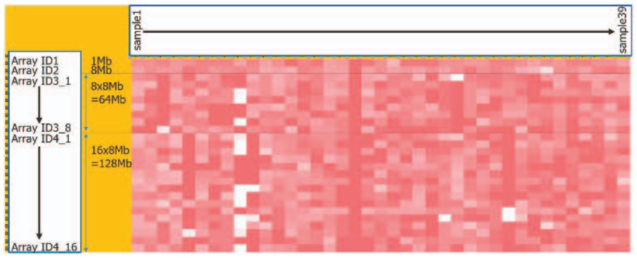


Fig. 2. Die-to-Die self-trimming setting color map

C. ECC-Aware Memory Test & Repair

ECC logic is required for MRAMs to handle their probabilistic nature. At the same time, using ECC logic can increase overall reliability by allowing a small number of defects per word to be corrected. We refer to the maximum number of ECC correctable defects within each word as N_{ECC}. MRAM memories may include redundant rows or columns which can be also used to increase yield.

In the following section we explain how ECC correction can be used in conjunction with redundant elements to repair defects during manufacturing. We refer to the number of ECC correctable bits allowed for production as $N_{ECCRepair}$ with $N_{ECCRepair} \leqslant N_{ECC}$.

During memory test with ECC enabled, a specific number of defects within a given word can be tolerated if the number of defects does not exceed $N_{ECCRepair}$. These defects can be ignored knowing they will be corrected by ECC. However, if the total number of defects in a word exceeds $N_{ECCRepair}$, we declare this as a multi-bit failure. If a compatible redundant element is available for these defects, then it is allocated; otherwise, the memory is declared non-repairable.

We pay special attention when testing memories with ECC and redundancy as to prevent ECC-based test escapes. We categorize these as Type-1 and Type-2 test escapes.

1) *Type-1 Test Escapes*

Table 1 illustrates how an MBIST test pattern might lead to an incorrect repair decision when ECC is used to repair manufacturing defects. In the example, the memory has an 8-bit data path. We assume bit D2 is stuck-at 1 and bit D6 is stuck-at 0.

The all-0 / all-1 test pattern is typically used to test memories. For this test pattern, the stuck-at 1 error is detected when reading back the all-0 pattern, and the stuck-at 0 error is detected when reading back the all-1 pattern. It (falsely) appears that a single error correction code is enough to repair the memory.

For memories without a column multiplexer, a checkerboard-like pattern and its inverse are also typically applied. Again, it would (falsely) appear that ECC can repair this memory.

However, this is not the case as any pattern containing a 0 for bit D2 and a 1 for bit D6 would cause a failure in the system. This is classified as a Type-1 test escape.

TABLE I. MULTI-BIT DEFECT DETECTED AS SINGLE-BIT DEFECTS BY TEST PATTERNS

D7	D6	D5	D4	D3	D2	D1	D0	Logical Data Word
X	S0	X	X	X	S1	X	X	Stuck-at-0 and Stuck-at-1 bits
0	0	0	0	0	0→1!	0	0	All 0s (PASS)
1	1→0!	1	1	1	1	1	1	All 1s (PASS)
0	1→0!	0	1	0	1	0	1	2x2 Checkerboard (PASS)
1	0	1	0	1	0→1!	1	0	2x2 Inverse Checkerboard (PASS)
X	1→0!	X	X	X	0→1!	X	X	UserData (FAIL!)

2) *Type-2 Test Escapes*

Conventional test patterns used in memory test algorithms do not allow complete repair of manufacturing defects in ECC bits when ECC is enabled. Figure 3 illustrates this case. For a particular ECC encoding scheme, assume the ECC code for the all-0 test pattern is 00000 and the code for the all-1 pattern is 10100. This means that three bits of every word never toggle. This is a test escape of bit line shorts when trying to apply a physical checkerboard pattern in memory.

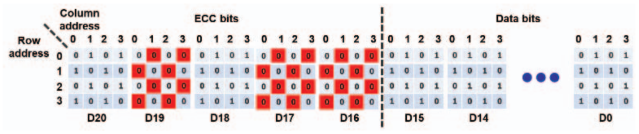


Fig. 3. Error Correction Code bits for test patterns

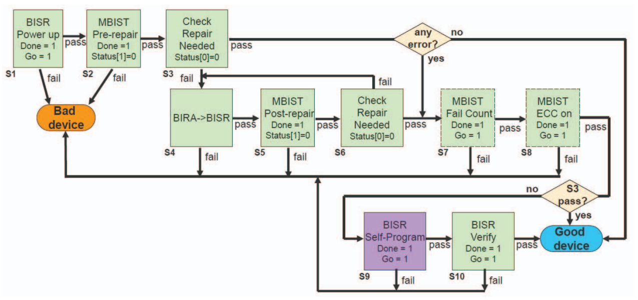
3) *ECC-Specific Memory Test Requirements*

To address Type-1 test escapes, the memory test logic must accumulate the results of all test patterns applied on the same address in order to identify multi-bit defects.

For Type-2 test escapes, additional data patterns are required which are not necessarily the same as the ones used for the detection of multiple errors in the data bits. Further, the additional test patterns depend on the actual ECC logic encoding scheme. Section III.D describes more information on how to avoid Type-2 test escapes.

D. ECC-Aware Test & Repair Flow

Figure 4 illustrates the ECC-aware memory test and repair flow. It comprises ten steps (S1 to S10), each corresponding to a test pattern. Several steps (S1, S3, S4, S6, S9, and S10) are identical to the standard MBIST flow discussed in Section II. However, the MBIST pre-repair (S2) and post-repair (S5) test steps differ significantly from the standard MBIST flow. The optional steps (S7 and S8) are also discussed below.



BISR Power up Done = 1 Go = 1
S1
pass
fail
MBIST Pre-repair Done =1 Status[1]=0
S2
pass
fail
Check Repair Needed Status[0]=0
S3
pass
fail
any error?
no
yes
Bad device
BIRA->BISR
S4
MBIST Post-repair Done =1 Status[1]=0
S5
Check Repair Needed Status[0]=0
S6
MBIST Fail Count Done =1 Go = 1
S7
MBIST ECC on Done =1 Go = 1
S8
S3 pass?
BISR Self-Program Done = 1 Go = 1
S9
BISR Verify Done = 1 Go = 1
S10
Good device

Fig. 4. ECC-aware test and repair manufacturing flow

During the MBIST pre-repair test (S2), ECC is disabled to avoid Type-2 escapes. The test algorithms must also be designed to avoid Type-1 escapes. This is done with a test including sequences targeting a reference address and accumulating the results of several read operations for an address before deciding how to repair the address: If the number of defects within one word is less than or equal to $N_{ECCRepair}$, then these defects are ignored, as ECC will correct them. However, if the total number of defects in a word exceeds $N_{ECCRepair}$, then a compatible spare element is allocated. The pre-repair test fails if there are not enough compatible spare resources available.

The MBIST post-repair test (S5) is the same as the pre-repair (S2). ECC is still disabled, and the algorithms used during pre-repair are applied again. This is necessary for a complete test of the spares allocated during pre-repair, including the ECC bits. It is also necessary to run in repair mode since single and sometimes double-bit errors must still be tolerated.

Optionally, an MBIST fail count test (S7) can be performed to verify that the number of errors repaired by ECC across the whole memory space does not exceed a certain threshold. This test is particularly useful for high-reliability applications.

Furthermore, an additional test with ECC enabled (S8), called MBIST ECC on, can also be applied. During this test, no errors are permissible on data bits post-ECC. However, it is challenging to evaluate the quality of this test due to Type-1 and Type-2 escapes.

E. Application of ECC-Aware Test and Repair Technology

The technology can be used in a wide variety of applications and use cases, covering various memory technologies, most notably SRAM and emerging NVM.

Beyond the conventional usage of MBIST, the technology enables less commonly used testing capabilities, such as online testing. Indeed, ECC-protected SRAM-based systems could perform online tests with a varying $N_{ECCRepair}$ budget to detect correctable errors before they accumulate and become uncorrectable. Another unconventional usage is for IP qualification of ECC-protected memories. For instance, given the budget $N_{ECCRepair}=1$, $N_{ECC}=2$, the initial screening of the QA samples can be done with $N_{ECCRepair}=1$, whereas the QA final test can use $N_{ECCRepair}=2$.

IV. Experimental Results

To verify overall the eMRAM automated self-trimming, test, and repair flow, we built a Verilog testcase comprising an 8Mb eMRAM instance of 128k addresses and 79 bits. It is coupled to a 2bit error correction and 3bit error detection ECC logic that can be bypassed. Conventional IO and row repair is also implemented. The MRAM DFT is inserted on top, in a fully automated way using the integrated flow with the MRAM MBIST tool. It is configured to use ECC to repair up to one bit error per word, leaving at least one error correction capability for in-field corrections. The memory instance behavioral model comes from our silicon proven eMRAM compiler ([3], [9], [10]), with faults injected as per figure 5 and table 2. The first class of injected faults extracted from actual silicon measurements is to demonstrate the automated self-trimming feature. This yields the optimal Sense Amplifier TRIM (SATRIM) setting that minimizes the Read0 and Read1 Fail Bit Count (FBC).

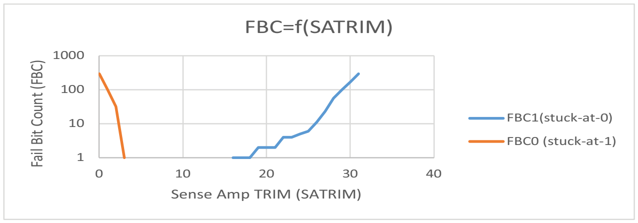


Fig. 5. First class of faults injection meant to demonstrate automated self-trimming of Read operations.

The second class of faults injection adds stuck-at bits, cumulatively, to demonstrate the ECC-aware test and repair.

TABLE II. Second class of faults injection

Bit position	Address	fault	Repair involved in 1bit ECC repair mode
Bit0	0	Stuck at 0	IO repair
Bit1	256	Stuck at 0	ECC
Bit2	512	Stuck at 1	ECC
Bit3	768	Stuck at 0	Row repair, thanks to error accumulation in BIRA
Bit4	768	Stuck at 1	

Next is to define the overall test flow as simulated. Table 3 lists the 30 test patterns that are run sequentially and exemplify the automated Read self-trimming, followed by the pre-repair, post-repair, and finally post-repair with ECC logic enabled. This latter step also involves a bit-masking feature available from the MBIST tool.

TABLE III. Overall simulated test flow

pattern #	test steps	details
1	clear_bisr	set and apply minimum trim (SATRIM) value to MRAM
2	pre_repair_i<0>	load trim value to MRAM, determine Write1 FBC at minimum trim value
3	BIRAtoBISR_i<0>	apply next median trim value to MRAM
[4-11]	pre_repair_i<i>, BIRAtoBISR_i<i>	trim Read1 (binary search), 1≤i<5
12	pre_repair_i<5>	trim Read1 (final iteration), determine R1_bound
13	clear_bisr	set and apply maximum trim (SATRIM) value to MRAM
14	pre_repair_b<0>	load trim value to MRAM, determine Write0 FBC at maximum trim value
15	BIRAtoBISR_b<0>	apply next median trim value to MRAM
[16-23]	pre_repair_b<i>, BIRAtoBISR_b<i>	trim Read0 (binary search), 1≤i<5
24	pre_repair_b<5>	trim Read0 (final iteration), determine R0_bound
25	BIRAtoBISR_b<5>	set and apply average of R1_bound and R0_bound to MRAM
26	load_trim	load final trim value to MRAM
27 (S2)	pre_repair	run ECC-aware pre-repair, ECC logic bypassed
28 (S4)	BIRAtoBISR_REPAIR	transfer BIRA to BISR
29 (S5)	PostRepair	run ECC-aware post-repair, ECC logic bypassed
30 (S8)	PostRepair_withECC	run post-repair, ECC logic and bit masking enabled

It should be noted that the test algorithms used for Read operation trimming intentionally use a reduced address space. Consequently, some of the faults from Figure 5 are not addressed by these algorithms; nevertheless, final SATRIM still yields no Read errors as shown in Figures 6 and 7.

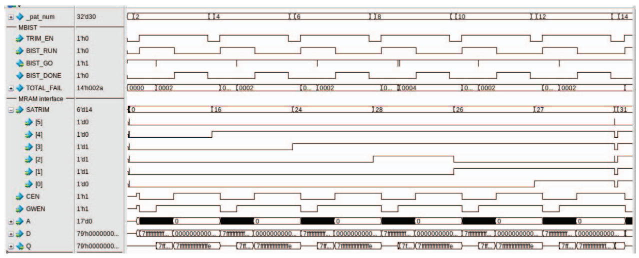


Fig. 6. Close-up of Read1 operation self-trimming test steps [1-12]

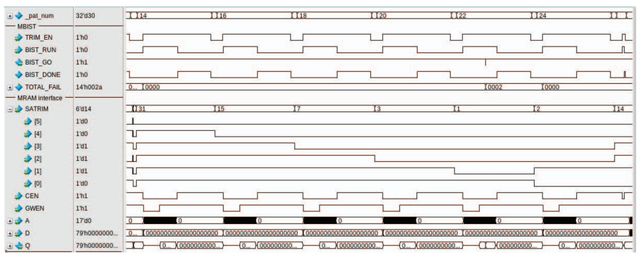


Fig. 7. Close-up of Read0 operation self-trimming test steps [13-26]

Once the MRAM instance is trimmed, the test sequence proceeds with the pre-repair test step S2, where BIRA is enabled and ECC logic is bypassed, as shown in Figure 8. This test step leads to an IO and a row repair in accordance with Table 2.

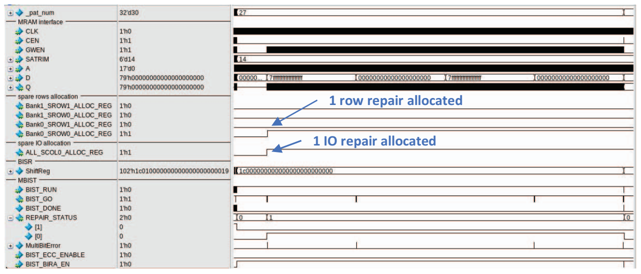


Fig. 8. Pre-repair with BIRA enabled, ECC bypassed, test step 27 (S2)

Next, comes the post-repair test step S5, where BIRA is still enabled and ECC logic remains bypassed, as shown in Figure 9. This test step completes this time with no multi-bit error thanks to the spare element's activation.

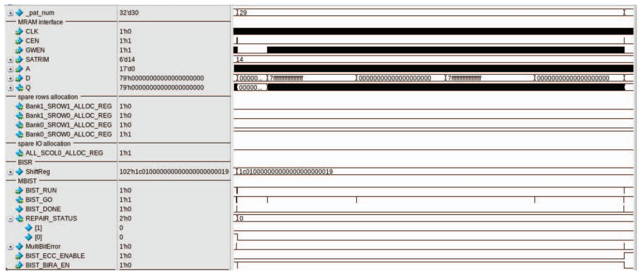


Fig. 9. Post-repair with BIRA still enabled, ECC bypassed, test step 29 (S5)

Finally, the optional post-repair test step S8, where BIRA is now disabled while ECC logic is no longer bypassed completes successfully (no errors), as shown in Figure 10.

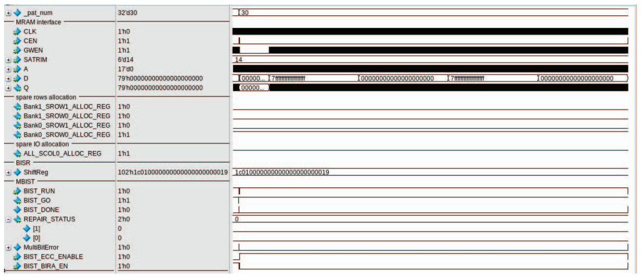


Fig. 10. Post-repair with BIRA disabled, ECC enabled, test step 30 (S8)

V. Conclusions

This paper describes a fully automated and integrated MBIST solution for embedded MRAMs (eMRAM). Such a solution is necessary for transitioning eMRAM from the technology exploration and low-volume pilot application stage to a high-volume, standard flow production line. The eMRAM MBIST solution builds on a standard MBIST solution, adding on-chip, fully automated trimming, as well as ECC-aware test and repair.

We demonstrated through hardware-derived defect insertion the effectiveness of the ECC-aware test and repair technology. Defects were derived from our production eMRAM [3] and simulated using files generated by our MRAM compiler. The ECC-aware technology allows us to trade-off simple error correction using ECC versus using repair resources during manufacturing test as well as in-system repair requirements.

In addition, the ECC-aware technology is user-programmable in the system. This enables ECC-tuning following the product's application need throughout its complete life cycle. We can now adjust things as manufacturing technology improves as well as when the eMRAM ages in the application, increasing manufacturing yield as well as the longevity of the product in its application.

References

[1] "Apollo4 Plus Low Power System-on-Chip", https://ambiq.com/apollo4-plus/

[2] "Fusion processors & microcontrollers for high-performance embedded processing", https://alifsemi.com/ensemble/

[3] T. Y. Lee et al., "World-most energy-efficient MRAM technology for non-volatile RAM applications," 2022 International Electron Devices Meeting (IEDM), San Francisco, CA, USA, 2022, pp. 10.7.1-10.7.4, doi: 10.1109/IEDM45625.2022.10019430.

[4] G. Hu et al., "Double spin-torque magnetic tunnel junction devices for last-level cache applications," 2022 International Electron Devices Meeting (IEDM), San Francisco, CA, USA, 2022, pp. 10.2.1-10.2.4, doi: 10.1109/IEDM45625.2022.10019402.

[5] Hyunjin Shin et al., "A 32Mb embeded flash memory based on 28nm with the best cell efficiency and robust design achievement featuring 13.48Mb/mm2 at 0.85V" VLSI symposium 2022 C15-2

[6] J. . -H. Park et al., "Highly reliable STT-MRAM adopting advanced MTJs with controlled domain wall pinning," 2022 International Electron Devices Meeting (IEDM), San Francisco, CA, USA, 2022, pp. 10.6.1-10.6.4, doi: 10.1109/IEDM45625.2022.10019352.

[7] Sina Bakhtavari Mamaghani1, Christopher Muench, Jongsin Yun, Martin Keim, Mehdi Baradaran Tahoori, "Smart Hammering: A Practical Method of Pinhole Detection in MRAM Memories", 2023 Design and Test in Europe (DATE).

[8] J. Yun, B. Nadeau-Dostie, M. Keim, L. Schramm, C. Dray, M. Boujamaa, K. Gelda "MBIST Supported Multi Step Trim for Reliable eMRAM Sensing," 2020 IEEE International Test Conference (ITC), Washington, DC, USA, 2020, pp. 1-5, doi: 10.1109/ITC44778.2020.9325218.

[9] Netsol nvRAM standalone products, http://netsol.co.kr

[10] Tae Young Lee, "Status and Outlook of eMRAM Technology", Samsung, MRAM Global Innovation Forum 2022