Application Report

Error Detection in SRAM

Peter Ehlig and Salvatore Pezzino

ABSTRACT

This application report presents information about how SRAM circuits fail and what can be done about it. This discussion is targeted towards the electronic system developer or integrator interested in improving the robustness of the system. The document is written from the perspective of research of failures in semiconductor SRAM instances.

Table of Contents

List of Tables

Trademarks

C2000™ are trademarks of Texas Instruments.
All trademarks are the property of their respective owners.

1 Introduction and Scope

With today's significant reliance on electronic control and management comes concern for the safety of the electronic operation. One key point of concern is the operation of the SRAM in the system. This is important and from some perspectives, the most important aspect of safety in the execution of electronics. This is because:

- SRAM is a large component in the overall device, often the largest.
  - Largest in area
  - Largest in transistor count
- SRAM is very dense circuitry and therefore susceptible to disturb or subtle defects.
- SRAM operates in reduced voltage ranges vs normal circuit logic and therefore susceptible to disturbances.

For these reasons, SRAM bit cell and array layouts are key components to any new integrated circuit process development and qualification. It is where integrated circuit fabrication experts go to find drift and variation in manufacturing. This is why SRAM Error Detection is important in safety related electronics. If you would like to skip to the end, see Section 8.

The discussion includes:

- Section 2: An overview of SRAM bit array organization
- Section 3: A summary of SRAM failure mechanism including where each is of concern
- Section 4: Methods available for managing memory failures in the electronic system
- Section 6: Specific to TMS320C2000 products

2 SRAM Bit Array

Information is stored into SRAM as words. The word length varies per the needs of the storage. 32-bit words are used for this discussion. The bits of the word are stored in an array of bits. However, the physical position of each individual bit is optimized to the specifics of the semiconductor process. The array is a matrix with rows and columns. In the example below, there are N rows of 8 words each. If N = 256 that comes to 256x8x32 = 65,536 bits or 2048 words of SRAM.

Note how the bits of an individual word are not physically adjacent to each other, but rather all the bits of 1 D-input/Q-output are adjacent. The bits of an individual word are physically separated by 7 bits from other words. This improves both the size of the physical array and the routing of the D/Q signals. You will see later how this improves the error detect-ability of multi-bit failure mechanisms.

The following shows values in three words as follows:

```
Word(0) = 1 … 0 1b     Row0, Column0
Word(1) = 1 … 1 0b     Row0, Column1
Word(7) = 1 … 0 0b     Row3, Column7
```

Table 2-1. SRAM Bit Array

			D0					D1					D31			
Row	Col															
	0	1	2	…	7	0	1	2	…	7	0	1	2	…	7	Column
N-1																
…																
5																
4																
3					0					0					1	
2																
1																
0	1	0				0	1				1	1				

The physical organization of bit cells in an array may vary for many reasons, but this figure provides a good example to discuss. This example shows a column multiplexor factor of 8, meaning there are 8 bits on a row for each Q/D. On a memory read the row is selected and then the column address provides the decode information for which specific bit will go out the Q. On a memory write, the column decodes define specifically which bits well be written to on the selected row. The sub-array (256x8) for each D/Q is called a stick.

Larger memory arrays are often organized with even higher column mux factors.

3 Sources of SRAM Failures

This section summarizes a number of sources for SRAM failures from the perspective of a system designer or integrator.

3.1 Manufacturing Defects

It is not the intention of this paper to go into details on defects generated during the manufacturing of a semiconductor device. The SRAM, being such a prevalent, dense, and sensitive circuit, is handed with significant attention in semiconductor device testing. The testing environment includes advanced and often proprietary testing algorithms. The manufacturing test environment allows for testing with voltage/temperature/frequency margining that is not possible in the end system. The semiconductor device designs include special test modes to allow the manufacturing test to include even further margin testing.

3.1.1 Time Zero Fails

Due to the special algorithmic targeting and extensive margining in the production tests, Time-Zero defect mechanisms are covered prior to device integration into the system. In a mature semiconductor process, most of the defects (>98%) are caught with a normal March algorithm, for example March13n. In more advanced process nodes (<65 nm), more advanced algorithms are necessary to get to this point.

3.1.2 Latent Fails

Latent manufacturing defects are material weakness in the semiconductor structure that might fail in the system, but might be missed with the above mentioned algorithm and margin testing. This structural weakness is targeted with manufacturing stress tests to force the failures prior to shipping the devices.

3.2 Circuit Drift With Usage

The performance of semiconductors will drift with time and eventually drift to the point of potential circuit failure. The device data sheet defines the device life time. The device design is verified with margined speed path analysis. The devices are tested with margin enough to operate correctly over the specified life time. Performance drift in the SRAM is specifically targeted with margin testing and margin design best practices.

3.3 Circuit Overstress

The data sheet clearly defines the voltage and temperature operation limits of the devices. The specified life time is shortened if the device is subjected to stresses outside those defined in the data sheet. Extreme stresses will damage the device. Sometimes the damage caused is an easily identifiable failure, but sometimes the failure is subtle. Overvoltage or High Temperature stressing specifically affects SRAM operation.

3.4 Soft Errors

In the context of this discussion soft errors refer to SRAM fails that are caused by an event external to the SRAM array and do not damage the SRAM circuitry. They are temporary in that once a new value is written to the affected word the error no longer exists. Two sources of this type failure are radioactive particles penetrating the circuit and dynamic voltage noise at the time of a read or write of a word.

Soft errors become more common as the semiconductor geometries reduce. This is both because the bits are closer together (creating a more target rich environment) and because the voltage levels are lower so that the bit cell storage is less robust.

Soft errors only affect the system if the system reads the disturbed word before this word gets written. Some SRAM contents are static in nature, for example code or tables. Other SRAM contents are dynamic, such as incoming data and variables. Also many algorithms operating on data are resilient to single bit errors especially if the errors are in the lower significant bits. So the fail rate due to soft errors is much lower in dynamic data than in static usage of SRAM. In many systems, the code and tables are stored in non-volatile memory like ROM or FLASH and the SRAM is primarily used for dynamic storage.

3.4.1 Radioactive Events

Radioactive events come from radioactive particles penetrating the semiconductor material and upsetting the voltage in the bit cell latch. These events are rare under normal circumstances, but can be higher in a radioactive medical environment and/or very high altitudes where there is less of an atmosphere to reduce the particle counts.

Depending upon the angle of the penetration and the intensity of the particle, one or more bit values can be flipped. This will happen in a straight line. If the straight line goes across rows then the multiple failing bits are guaranteed to be in different words. This works for vertically oriented penetration or for diagonal.

If the penetration orientation is down the row, then it must go beyond the mux factor to create more than one bit in the same word.

The smaller the geometries, the more bits can be disturbed by a particle penetration. At a 90 nm process node a penetration of 5 cells is extremely rare. At 65 nm a penetration of 6 cells is extremely rare. Low leakage process nodes are less susceptible to these events than high performance process nodes.

Radioactive particle penetration can, and occasionally does, physically damage SRAM cells. This is more common in process nodes with more aggressive structural geometries.

3.4.2 Dynamic Voltage Events

Any electronic circuitry that involves switching will have dynamic voltage events, even on an integrated circuit there are voltage events. For example, any time the main system clock switches, many other transistors transition and there is some noise on the power plane. While the voltage events within the semiconductor device are small, a marginal bit cell might be disturbed if it happens during a read or write to the bit cell. However, the devices are tested with enough margins to screen out such bit cells so this is not a concern at the system level. Voltage events that are large enough to disturb a margined bit cell come from off chip or operating the devices out of data sheet specification. Even so, it will happen only to the weaker bits (weaker due to normal and expected process variation). Therefore, single bit detection is adequate to address such events.

Multi-bit read or write failures can happen due to a significant dynamic voltage event. This is indicative of a voltage issue and best addressed with voltage monitoring rather than memory monitoring.

3.4.3 Summary of Error Sources

In summary, the primary concern for in-system SRAM errors is disturb failures due to soft events discussed in Section 3.4.

Manufacturing defects are screened via methods only available in the manufacturing environment. In the unlikely event of these defects getting through un-screened, the methods in Section 4 provide additional coverage. The methods described in Section 4 also address failures due to normal circuitry drift that may become a factor when systems go beyond the data sheet device life time or should there be minor overstress.

4 Methods for Managing Memory Failures in Electronic Systems

This section presents a number of methods available for managing memory failures in electronic systems. While specifically targeting SRAM, most of the information applies to other memories such as ROM and FLASH. The description is from the perspective of a system designer or integrator, but considers the previous discussions tied to integrated circuit devices.

Even within the realm of safety considerations there are different perspectives towards managing memory failures.

- Safe State: Taking the system to a safe state when an error has been identified.
- System Availability: Continuing to run in some cases when an error has been identified.
- Fail Safe: System continues properly even when an error has been identified.

Each of these three perspectives adds cost to the system. Depending upon the requirements of the market, the additional cost may be easily justified. Alternatively, these additional costs may be prohibitive in other target markets.

4.1 Start-Up Testing

Testing the SRAM at start-up does not address the three above listed perspectives. However, there is significant value added in testing the SRAM at start-up even if it is not possible to test while the system is operational.

There is a proverb in the semiconductor industry that the second worst thing you can do to a system is power it down, with the very worst thing being power up. This is because the voltage and current swings during power up and power down of a system (or system component) cannot be fully managed. The devices are designed to work with this, but should a destructive power event occur, it is more likely to occur during power up/down. Therefore, testing the SRAM at power up addresses the most likely points where the circuit will be damaged.

Additionally, SRAM is easier to test at start up because there is no context to save and restore because the system context has not yet been loaded. SRAM testing by the embedded CPU takes a lot of CPU cycles and is not as easily or as effectively done while sharing the CPU resources with the system operation.

4.2 In-System Testing

In-system testing involves testing a targeted circuit while the system is in full operation. This involves taking a time slice for the CPU's attention to test a portion of the circuitry. This time slice must be small enough to not affect the CPU's system responsibility. In a real-time control system this can be restrictive. With respect to the Safe State perspective, the complete range of the SRAM must be tested within each safety interval.

In-system testing of SRAM involves:

- Context Save of the targeted SRAM
- Running a SRAM test algorithm over the SRAM
- Restoring the previous context of the SRAM

This has to be done before the system requires access to the SRAM under test. It is difficult to do this for a full SRAM instance within a device. However, it is possible to do this on a small slice of SRAM at a time, for example 16 or 32 words. As stated in Section 3.1.1, most defect related failures in-system can be detected with a simple March algorithm (March13n or even March7). In newer process nodes (45 nm and smaller), there is a need for more advanced algorithms to get the same coverage.

This method does not detect a failure on a system read of the SRAM, but can catch errors developing in the memory before the system reads a failing location. The method is used in cases where the SRAM in devices has no other detection methods available.

4.3 Parity Detection

Parity detection identifies single bit errors in a read access. The parity circuitry sets the parity bits when the SRAM word location is written and verifies that there are no single bit errors in the word when it is read back. This is done within the read/write cycles, so no CPU overhead his involved. Should the parity circuitry identify an error, it generates a high priority interrupt to the CPU.

This detection mechanism is simple and relatively inexpensive to implement in semiconductor devices. Parity addresses the Safe State perspective for Safety. As described earlier in Section 2 and Section 4.1 virtually all SRAM failures in-system are likely to be single bit per word failures. This applies to both physical defect mechanisms as well as soft errors. Additional coverage can be provided by also protecting the memory address bits with parity.

4.4 Error Detection and Correction (EDAC)

Error detection and correction (often referred to as ECC) goes beyond parity in that it corrects single bit errors. The EDAC circuitry can also detect a 2-bit uncorrectable error. It is possible to implement 2-bit correction, but per the discussions in Section 2 and Section 4.1, this does not provide a good return for the added cost and timing overhead.

EDAC addresses the system availability perspective for safety since the system will continue to run unabated in the presences of a single bit error.

However, EDAC:

- Adds significant cost to the memory portion of the device
- Slows down the CPU due to the added SRAM access time necessary to make corrections on the fly
- Requires more system power

For example consider the following:

- SRAM on a device is about 1/3 the cost
- EDAC cost adder to the SRAM is 30%
- EDAC requires the addition of a wait state to the memory access time
- The price of the device goes up ~40%

Not all the SRAM necessarily requires EDAC protection so this can vary with device designs. Likewise there are ways to reduce access time to less than a wait state.

4.5 Redundancy

Redundancy is the most expensive option, but also provides the best path to a Fail Safe solution. Redundancy can be implemented at varying degrees:

- Redundant logic on the device
- Multiple processors running in lock step
- Multiple processors with voting

This path rapidly expands in complexity and will not be further covered in this discussion.

5 Comparisons and Conclusions

Table 5-1 provides a summary of which detection mechanisms provide what protection level.

Table 5-1. Detection Mechanisms and Protection Levels

SRAM Coverage	Start-Up	In-System	Parity	EDaC	Redundant
Single bit per word hard failures	Y	Y	Y	Y	Y
In-system degradation single bit per word failures	N	Maybe 1	Y	Y	Y
Soft errors	N	N	Y	Y	Y
Multi-bit per word	N	N	Maybe	Y	Y
Safe state	N	N	Y	Y	Y
System availability	N	N	N	Y	Y
Fail safe	N	N	N	N	Y

1. In-system testing only catches this failure if the in-system tests check the failing word before the system does. While this is likely, it is not guaranteed.

6 C2000™ Memory Types Example

This appendix discusses what level of protection is available for the TMS320F2837xD Dual-Core Microcontrollers. The purpose is to aid the reader in finding the proper information in the documentation provided for a particular device. This is an example for one device.

6.1 TMS320F2837xD

The TMS320F2837xD memory type information can be found in the *TMS320F2837xD Dual-Core Microcontrollers Data Manual*. The *Detailed Description* section of the data manual contains information about the device. Section 7 contains the information from the *Memory Type* section of the data manual regarding which memories contain what level of protection. Subsequent sections detail each memory type.

7 Memory Types

Table 7-1 provides more information about each memory type.

Table 7-1. Memory Types

Memory Type	ECC-Capable	Parity	Security	Hibernate Retention	Access Protection
M0, M1	Yes	–	–	Yes	–
D0, D1	Yes	–	Yes	–	Yes
LSx	–	Yes	Yes	–	Yes
GSx	–	Yes	–	–	Yes
CPU/CLA MSGRAM	–	Yes	Yes	–	Yes
CPU1/CPU2 MSGRAM	–	Yes	–	–	Yes
Boot ROM	–	–	–	N/A	–
Secure ROM	–	–	Yes	N/A	–
Flash	Yes	–	Yes	N/A	N/A
User-configurable DCSM OTP	Yes	–	Yes	N/A	N/A

7.1 Dedicated RAM (Mx and Dx RAM)

The CPU subsystem has four dedicated ECC-capable RAM blocks: M0, M1, D0, and D1. M0/M1 memories are small nonsecure blocks that are tightly coupled with the CPU (that is, only the CPU has access to them). D0/D1 memories are secure blocks and also have the access-protection feature (CPU write/CPU fetch protection).

7.2 Local Shared RAM (LSx RAM)

RAM blocks which are dedicated to each subsystem and are accessible to its CPU and CLA only, are called local shared RAMs (LSx RAMs).

All LSx RAM blocks have parity. These memories are secure and have the access protection (CPU write/CPU fetch) feature.

By default, these memories are dedicated to the CPU only, and the user could choose to share these memories with the CLA by configuring the MSEL_LSx bit field in the LSxMSEL registers appropriately.

Table 7-2 shows the master access for the LSx RAM.

Table 7-2. Master Access for LSx RAM (With Assumption That all Other Access Protections are Disabled)

MSEL_LSx	CLAPGM_LSx	CPU Allowed Access	CLA Allowed Access	Comment
00	X	All	–	LSx memory is configured as CPU dedicated RAM.
01	0	All	Data Read Data Write	LSx memory is shared between CPU and CLA1.
01	1	Emulation Read Emulation Write	Fetch Only	LSx memory is CLA1 program memory.

7.3 Global Shared RAM (GSx RAM)

RAM blocks which are accessible from both the CPU and DMA are called global shared RAMs (GSx RAMs). Each shared RAM block can be owned by either CPU subsystem based on the configuration of respective bits in the GSxMSEL register.

All GSx RAM blocks have parity.

When a GSx RAM block is owned by a CPU subsystem, the CPUx and CPUx.DMA will have full access to that RAM block whereas the other CPUy and CPUy.DMA will only have read access (no fetch/write access).

Table 7-3 shows the master access for the GSx RAM.

Table 7-3. Master Access for GSx RAM (With Assumption That all Other Access Protections are Disabled)

GSxMSEL	CPU	Instruction Fetch	Read	Write	CPUx.DMA Read	CPUx.DMA Write
0	CPU1	Yes	Yes	Yes	Yes	Yes
	CPU2	–	Yes	–	Yes	–
1	CPU1	–	Yes	–	Yes	–
	CPU2	Yes	Yes	Yes	Yes	Yes

The GSx RAMs have access protection (CPU write/CPU fetch/DMA write).

7.4 CPU Message RAM (CPU MSGRAM)

These RAM blocks can be used to share data between CPU1 and CPU2. Since these RAMs are used for interprocessor communication, they are also called IPC RAMs. The CPU MSGRAMs have CPU/DMA read/write access from its own CPU subsystem, and CPU/DMA read only access from the other subsystem.

This RAM has parity.

7.5 CLA Message RAM (CLA MSGRAM)

These RAM blocks can be used to share data between the CPU and CLA. The CLA has read and write access to the "CLA to CPU MSGRAM." The CPU has read and write access to the "CPU to CLA MSGRAM." The CPU and CLA both have read access to both MSGRAMs.

This RAM has parity.

8 Summary

In closing, here are a couple more points of clarification.

Many newer chips in the newest semiconductor process nodes make more extensive use of EDAC in their SRAMs than earlier devices in older process nodes. While this does provide better coverage than older devices with parity or no automated detection in the memory, it may be due more to process requirements than safety. The newer process nodes, with more aggressive structural geometries, are more susceptible to both soft errors and process degradation and it is necessary to add EDAC to meet reasonable device life times. This is not a concern for the newer devices so long as they include the EDAC. However, the need for EDAC is greater in devices making use of these new processes.

A second point to consider is the available history for the process node. If the targeted market requires a 10 year life time, then from a safety perspective, it is advantageous to have greater than 10 years of volume production in the process node that services markets while carefully monitoring field failures. This is because the advancements in each new process bring with them new and unique problems.

9 References

- Texas Instruments: *TMS320F2837xD Dual-Core Microcontrollers Data Manual*

10 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision * (November 2017) to Revision A (November 2020)	Page
• Updates were made in the Abstract of this document	1
• Updated the numbering format for tables, figures and cross-references throughout the document	2

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265