CTAN017: Read Disturb Handling

Covered Products: -503 CF cards, -900S SATA products, -806/808 SD cards, -300 USB products

1 Introduction

There are some applications that consist of mostly read operations and very few or no write operations to the flash storage device. An example of such an application is casino gaming, where the software is loaded onto a flash memory card once and is then read only in actual use.

For these applications, some have chosen to use MLC NAND products in the belief that since there is little or no write operations to the flash device, the low endurance rating of MLC NAND is not a factor and that the flash memory can be read indefinitely without issues. However, this is not the case. Actual data collected from the field shows that data stored on MLC flash can get corrupted even in a read only application. This is because flash memory is vulnerable to read disturb errors, particularly for small geometry, MLC NAND.

In this appnote, we will explore what causes read disturb error and what steps are taken in Cactus Technologies® flash products to mitigate this problem.

2 Read Disturb Errors

In a NAND flash array, the memory cells are organized in long strings, functionally equivalent to a multi-input NAND gate. During a read operation, the selected word line is biased to a read threshold voltage, which is in between the states that need to detected. The unselected word lines are biased high voltage level so that all the cells on the NAND string are fully conducting, which allows the state of the selected cell to pass through to the sense amplifier. This bias condition is illustrated in Fig. 1:

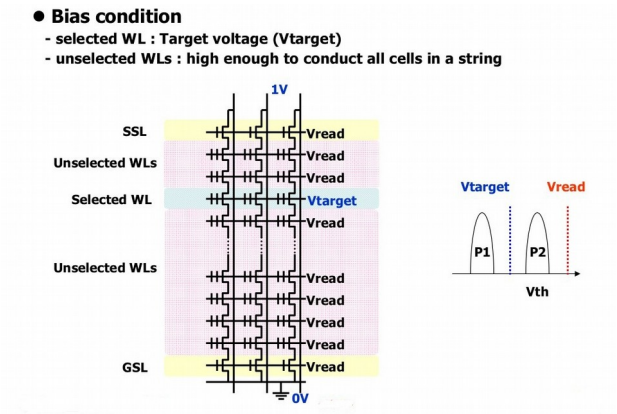

Fig. 1: Read bias condition

This biasing condition, however, can cause the cells in the unselected word lines to be slightly programmed. If this exact read biasing configuration is repeated multiple times, eventually, the cells in the unselected word lines will gradually shift towards the programmed state and lead to a sensing error. This condition is illustrated in the diagram below:

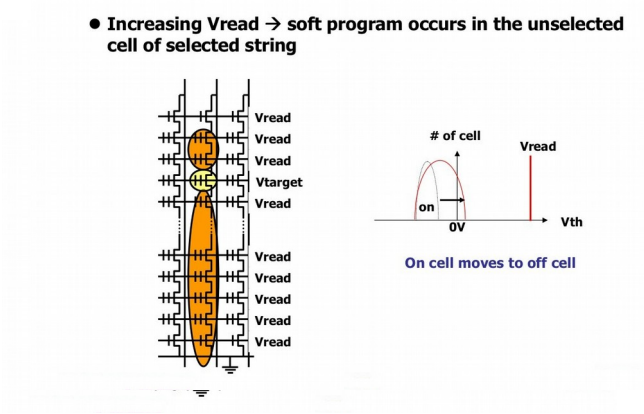

Fig 2: Read disturb

Read disturb errors can also be caused by cross coupling noise from neighbouring cells. These cross coupling noise gets progressively worse as the device geometry shrinks, as illustrated in the following chart:

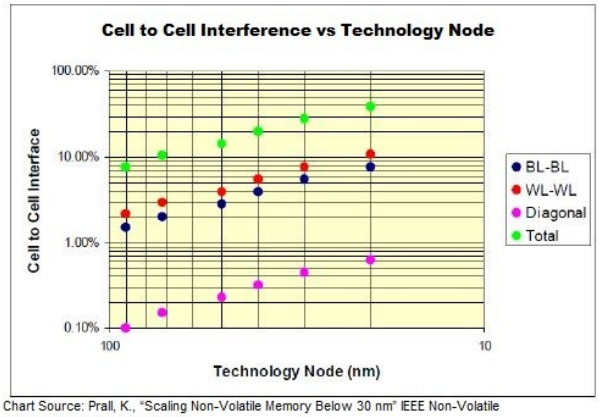

Chart Source: Prall, K., "Scaling Non-Volatile Memory Below 30 nm" IEEE Non-Volatile Semiconductor Workshop 2007

This type of cross coupling induced errors gets worse over temperature as high temperature accelerates the movement of charge to and from the memory cells.

If the read disturb errors do not get handled properly, eventually they will lead to uncorrectable ECC errors and device failure.

3 Read Disturb Handling

To combat the effects of read disturb, a couple of features have been implemented in selected Cactus Technologies® products.

3.1 Near Miss ECC

An ECC error detection threshold, which is below the max. ECC correction capability of the device, can be set. When this threshold is reached on a read operation, the block in error will be refreshed by re-writing the data. This re-write operation can take place during the next write command or immediately, depending on the particular product.

3.2 Read Wear Leveling

In Read Wear Leveling, a read disturb counter is implemented per block. This count is monitored and compared to a preset threshold. When the threshold is reached, the block is refreshed by re-writing the data to a different block.

Note that there is minimal performance impact on the product in refreshing the affected blocks as this event happens rarely and usually only when the product is near end of life.

4 Final words

As we tried to show in this appnote, a read only application can encounter read disturb errors, particularly for small geometry processes (i.e. 1X/1Y). Cactus Technologies® has implemented firmware features to combat this specific problem. If the reader is interested in this type of product, please contact our sales team for further information.

5 Version History

Version	***Date***	***Change***
1.00	Nov. 1, 2016	Initial Version