

Low Stress Program and Single Wordline Erase Schemes for NAND Flash Memory

Jin-Ki Kim, Hong-Beom Pyeon, HakJune Oh, Roland Schuetz and Peter Gillingham

MOSAID Technologies Incorporated, 11 Hines Road, Ottawa, Ontario, Canada K2K 2X1

Introduction

Voltage stress during programming [1] is a major factor limiting reliability in NAND Flash memory. To control programming stress several desirable features such as random page program, partial page program, and low Vcc operation are eliminated or restricted. Program stress becomes more significant as process technology is scaled down and as Single-Level Cell (SLC) gives way to Multi-Level Cell (MLC) devices. To increase device reliability a low stress program scheme for random page program in SLC devices and a sequential program scheme for MLC devices is introduced. Separately, system-level performance degrades as a function of the NAND block size [2] due to the additional operations necessitated by wear-leveling algorithms. Techniques for single wordline erase in SLC and partial block erase in MLC are introduced to minimize system overhead due to larger block size and to extend the system lifetime.

Source Precharge Program Inhibit and Asymmetrical Self Channel Boosting Program Scheme

The self boosted program inhibit scheme [1] is commonly employed to prevent unwanted programming of erased cells. However, the boosted channel voltage that inhibits programming is strongly dependant on Vcc and the pass voltage applied to unselected wordlines (Vpass). In order to reduce stress during programming, channel boosting efficiency should be maximized while minimizing the magnitude of Vpass. In addition, it is highly desirable to lower Vcc as process geometries decrease and to reduce power consumption for mobile applications. In order to facilitate lowering Vcc while simultaneously reducing stress during programming, we introduce the Source Precharge Program Inhibit (SPPI) and Asymmetrical Self Channel Boosting (ASCB) program schemes shown in Fig. 1 and Fig. 2.

At time T2 the selected NAND string is precharged to Vpass through the Segmented Common Source Line (SCSL) by applying Vpass to most wordlines and the Ground Select Line (GSL). Once precharged, at time T3 the selected cell is decoupled from the rest of cells in the selected NAND string by the Vdcp gate voltage on the upper adjacent cell, and by dropping the voltage on the lower adjacent cell to 0V. At time T4, the selected wordline rises from Vpass to Vpgm locally boosting the channel of the selected cell by $\Delta V_{pgm} (= V_{pgm} - V_{pass})$ with high boosting efficiency. Finally, at time T6, the input data on the bitline is loaded onto the selected NAND string by enabling the string select transistor. Depending on the input data (Vcc for Data '1' and 0V for Data '0'), the boosted charge on the selected NAND string is discharged to enable the programming of the selected cell. If the input data is '1', then the boosted charge remains unaffected and programming is inhibited. Fig. 3 shows the boosted channel voltage as a function of Vcc and the page program functionality. As compared to conventional Incremental Step Pulse Programming [1], the proposed SPPI and the ASCB program scheme eliminates the Vcc dependency and enables a lower and constant Vpass. Moreover, this new program scheme loads bitline data through the string select transistor in the final programming step to dramatically reduce wordline-to-SSL capacitive coupling [3]. Programming using a lower and constant Vpass level greatly reduces stress.

In MLC NAND Flash memory, sequential page programming is unavoidable if cell-to-cell interference (i.e. Vth shift) [3] with adjacent cells is to be minimized. Figure 4 shows the boosted channel voltage (Vch_boost) during a sequential page program operation as a function of Vpass and the data pattern when the new programming scheme is used. It shows that Vpass can be reduced to 7V even while maintaining a boosted channel voltage of at least 9V. As a result, the voltage stress during programming ($V_{pgm} (18V) - V_{ch_boost}$) is kept to 9V.

Single Wordline Erase

For NAND Flash memories, the large size mismatch between the amount of memory erased as a unit (a block) and the amount of memory programmed at a time (a page), causes the controller to often perform many page copy operations whenever a part of a block needs to be updated. The consequences of these additional operations include overall decreased performance, increased power consumption, and unnecessary consumption of finite erase/program cycles.

All the data in arbitrarily selected rows of cells can be erased by applying 0V to the corresponding wordlines and Vers-Vth to all the other (unselected) wordlines (Fig. 5). This erase scheme places some additional load on the Vers generator, but this is not significant since the array substrate capacitance is much greater than that of the selected wordlines in the selected block. Erasure of the unselected blocks can be prevented by a conventional self-boosting erase inhibit scheme [1]. The ability to erase only the cells under one or more word lines eliminates the complexity and performance degradation stemming from the size difference between the erase unit and the program unit in conventional NAND Flash.

Since MLC Flash memories must be sequentially programmed, a different partial block erase scheme is needed. Fig. 6 shows a new partial block erase scheme for MLC memories in which all the cells under a contiguous collection of wordlines, beginning at any point within the block and continuing to the highest wordline in the block (the one closest to the String Select Line (SSL)), are all erased simultaneously. For example, in Fig. 6b, all the cells under wordlines 27 through 31 are erased together; in Fig. 6c, all the cells under wordlines 2 through 31 are erased. This technique minimizes the cell-to-cell interference to which MLC storage cells are susceptible.

A conventional block erase verify (i.e. all wordlines in the selected block biased to 0V) cannot be done following a single or multiple wordline erasure because some cells in the selected NAND strings may still be in a programmed state. Instead, as shown in Fig. 7, an arbitrary number of cells per string can be verified by varying the Vcs that is applied to the bottom of the string. Adequate sensing margins to verify any number of cells can be guaranteed with only a few different Vcs values. The specific value of Vcs depends on the number of wordlines erased.

Conclusion

To address the critical issue of program stress and enable low Vcc (1.8V) operation in NAND Flash memory, the novel SPPI and ASCB program schemes are introduced. Single and multiple wordline erase, and partial block erase schemes are also introduced. These address the traditional mismatch between erase and program granularity, and thereby improve system performance while reducing power consumption and extending effective device lifetime. The new program and erase schemes do not require process & cell changes. The chip size increase to implement the new program and erase schemes is less than 0.3%.

References

- [1] K-D Suh et al., "A 3.3V 32Mb NAND flash memory with incremental step pulse programming scheme", *ISSCC Tech. Dig.*, pp. 128-129, 1995.
- [2] K. Takeuchi et al., "A 56nm CMOS 99mm² 8Gb Multi-level NAND Flash Memory with 10Mbyte/sec Program Throughput", *ISSCC Tech. Dig.*, pp. 144-145, 2006.
- [3] June Lee et al., "A 1.8V 2Gb NAND Flash Memory for Mass Storage Applications", *ISSCC Tech. Dig.*, pp. 290-291, 2003.
- [4] J-D Lee et al., "Effects of Floating-gate interference on NAND Flash Memory Cell Operation", *IEEE Electron Device Letter*, Vol. 23, No. 5, pp. 264-266, May 2002.

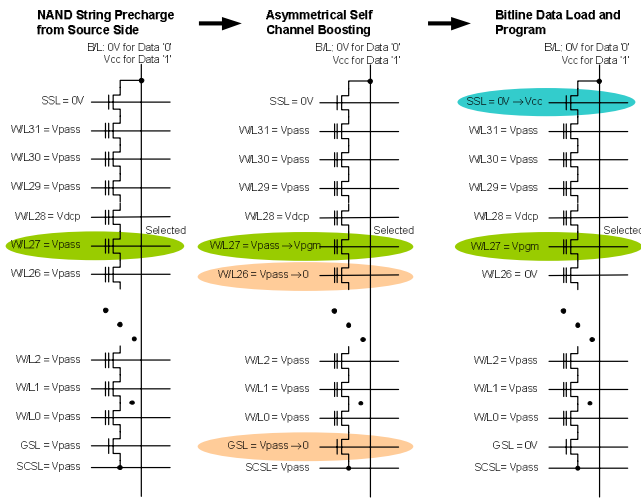


Fig. 1 Program Sequence

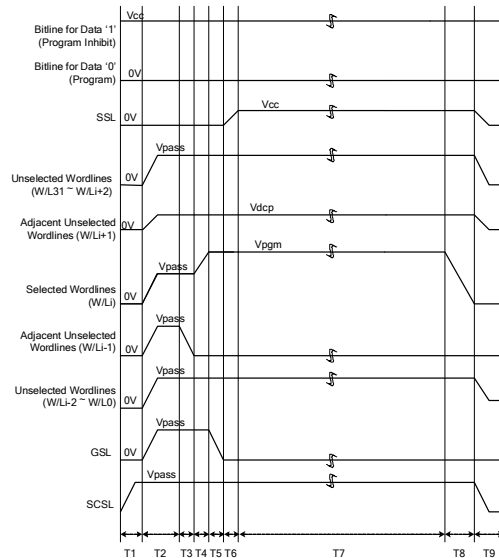


Fig. 2 Program Timing

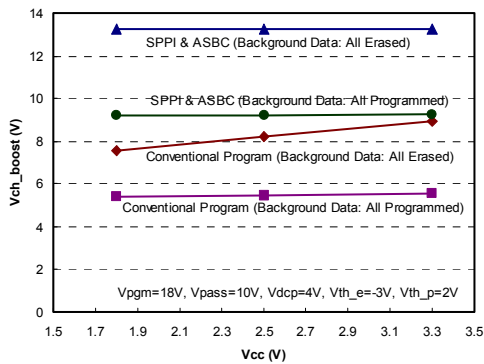


Fig. 3 Boosted Channel Voltage with Random Page Program

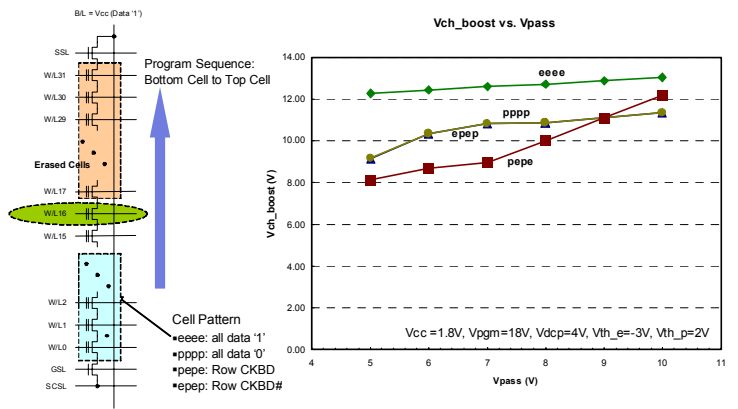


Fig. 4 Boosted Channel Voltage with Sequential Page Program for MLC

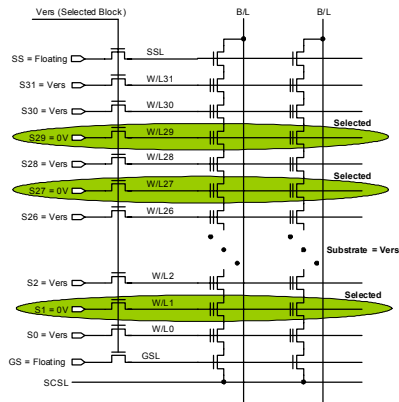


Fig. 5 Wordline Basis Erase for SLC

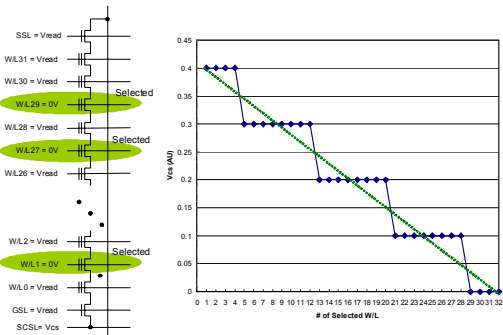


Fig. 7 Erase Verify - Vcs Characteristic

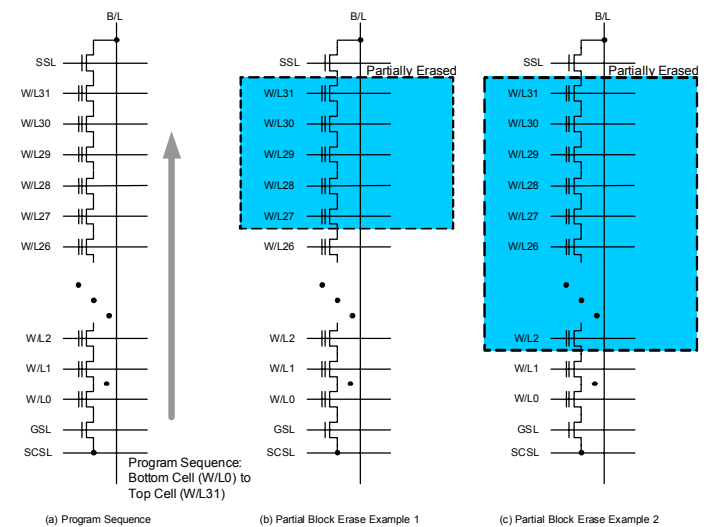


Fig. 6 Partial Block Erase Corresponding to Sequential Program for MLC