Micheloni · Crippa · Marelli

Inside NAND Flash Memories

Petitioner Micron Ex-1021, 0001

Rino Micheloni
Luca Crippa
Alessia Marelli

Inside NAND Flash Memories



Petitioner Micron Ex-1021, 0002

Inside NAND Flash Memories

Petitioner Micron Ex-1021, 0003

Rino Micheloni • Luca Crippa • Alessia Marelli

Inside NAND Flash Memories



Petitioner Micron Ex-1021, 0004

Rino Micheloni
Integrated Device Technology
Agrate Brianza
Italy
rino.micheloni@ieee.org

Luca Crippa
Forward Insights
North York
Canada
luca.crippa@ieee.org

Alessia Marelli
Integrated Device Technology
Agrate Brianza
Italy
alessiamarelli@gmail.com

ISBN 978-94-007-9834-2 ISBN 978-90-481-9431-5 (eBook)
DOI 10.1007/978-90-481-9431-5
Springer Dordrecht Heidelberg London New York

© Springer Science+Business Media B.V. 2010
Softcover re-print of the Hardcover 1st edition 2010
No part of this work may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, microfilming, recording or otherwise, without written permission from the Publisher, with the exception of any material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work.

Cover design: eStudio Calamar S.L.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Petitioner Micron Ex-1021, 0005

Preface

In the last decade Flash cards became the most important digital storage support. It is difficult to identify a single killer application (digital photography, MP3, digital video, ...) or whether the success of this media support is due to its usability in different applications. It is also difficult to state whether the recent revolution in the infotainment area has been triggered by the availability of high-performance NAND Flash memories or if the extraordinary success of Flash cards is a consequence of the establishment of new applications.

In any case, independently of the cause–effect relationship linking new digital applications and Flash cards, to realize how NAND Flash memories entered in our daily life, it is sufficient to imagine as they would change our recent habits if the NAND memories disappeared suddenly. To take a picture it would be necessary to find a film (as well as a traditional camera...), disks or even magnetic tapes would be used to record a video or to listen a song, and a cellular phone would return to be a simple mean of communication rather than a console for an extended entertainment.

The development of NAND Flash memories will not be set down on the mere evolution of these digital systems since a new killer application can trigger a further success for this memory support: the replacement of Hard Disk Drives (HDD) with Solid State Drives (SSD).

The advantages of SSD with respect to HDD are countless (higher read/write speed, higher mechanical reliability, random access, silent operation, lower power consumption, lower weight, ...). Nevertheless, the cost per gigabyte is still significantly favorable to HDD and the success of SSD will depend only on the performances that they will succeed in reaching.

The present book, the fourth authored by Rino Micheloni and coworkers after *VLSI-Design of Non-volatiles Memories*, *Memories in Wireless Systems* and *Error Correction Codes for Non-volatile Memories*, all published by Springer, tackles all the main aspects related to NAND Flash memories, from the physical, technological, and circuit issues to their use in Flash cards and SSDs.

After an extended market overview (Chap. 1), Chap. 2 has been conceived as a guide for the entire book, so that the reader can directly reach the heart of the problems that interest him.

The following chapters deepen physical and technological aspects. In particular, Chaps. 3 and 4 tackle technological and reliability issues of traditional Floating-Gate NAND memories, respectively, while the state-of-the-art of charge trapping technologies is effectively summarized in Chap. 5.

Petitioner Micron Ex-1021, 0006

The central part of the book is dedicated to circuit issues: logic (Chap. 6), sensing circuits (Chaps. 8 and 9) and high voltage blocks (Chaps. 11 and 12). Other basic topics related to circuit aspects are also analyzed, such as the implementation of Double Data Rate interfaces (Chap. 7), while multilevel storage (2 bits per cell) is presented in Chap. 10.

Techniques adopted to increase memory reliability and yield are discussed in Chaps. 13 and 14, the former dealing with redundancy, the latter with Error Correction Codes. The test flux used by NAND memories manufactures is described in Chap. 15.

Chapter 16 focus on NAND devices storing 3 and 4 bits per cell that allow reaching the lower cost per gigabyte and that will conquer, in the next few years, the market of consumer applications.

The last chapters are dedicated to the two most popular systems based on NAND memories: Flash cards (Chap. 17) and SSD (Chap. 18). The relationship between NAND memories and system performances are highlighted.

Finally, Chap. 19 describes the effects of ionizing radiations on non-volatile memories, to understand whether NAND memories can be safely used in space and military applications.

Prof. Piero Olivo,
Dean of the Engineering Faculty,
University of Ferrara, Italy

Petitioner Micron Ex-1021, 0007

Acknowledgements

After completing a project like a technical book, it is very hard to acknowledge all the people who have contributed directly or indirectly with their work and dedication.

First of all, we wish to thank all the authors of the contributed chapters.

We have to thank Mark de Jongh for giving us the possibility of publishing this work and Cindy Zitter for her continuous support.

Last but not least, we keep in mind all our present and past colleagues for their suggestions and fruitful discussions.

Rino, Luca and Alessia

Petitioner Micron Ex-1021, 0008

Table of contents

Preface v

Acknowledgements vii

1 Market and applications for NAND Flash memories 1
Gregory Wong

2 NAND overview: from memory to systems 19
R. Micheloni, A. Marelli and S. Commodaro

3 Program and erase of NAND memory arrays 55
Cristoph Friederich

4 Reliability issues of NAND Flash memories 89
C. Zambelli, A. Chimenton and P. Olivo

5 Charge trap NAND technologies 115
Alessandro Grossi

6 Control logic 131
A. Marelli, R. Micheloni and R. Ravasio

7 NAND DDR interface 161
Andrea Silvagni

8 Sensing circuits 197
L. Crippa and R. Micheloni

9 Parasitic effects and verify circuits 235
L. Crippa and R. Micheloni

10 MLC storage 261
L. Crippa and R. Micheloni

11 Charge pumps, voltage regulators and HV switches 299
R. Micheloni and L. Crippa

Petitioner Micron Ex-1021, 0009

12 High voltage overview 329
R. Micheloni and A. Marelli

13 Redundancy 353
A. Marelli and R. Micheloni

14 Error correction codes 393
T. Zhang, A. Marelli and R. Micheloni

15 NAND design for testability and testing 423
Andrea Silvagni

16 XLC storage 455
R. Micheloni and L. Crippa

17 Flash cards 483
A. Ghilardelli and S. Corno

18 Low power 3D-integrated SSD 515
K. Takeuchi

19 Radiation effects on NAND Flash memories 537
M. Bagatin, G. Cellere, S. Gerardin and A. Paccagnella

About the authors **573**

Index **575**

Petitioner Micron Ex-1021, 0010

1 Market and applications for NAND Flash memories

*Gregory Wong**

1.1 Introduction

In the year 2000, the total semiconductor memory revenues of SRAM, DRAM, NOR Flash and NAND Flash memories amounted $46 billion. As the decade closed in 2009, total revenues are projected to have declined 8.7% to $42 billion. Of the major memory segments, only NAND Flash memory revenues have grown in the past decade at a CAGR of almost 50% (see Fig. 1.1).

Standalone SRAM saw a steady decline (CAGR – 14.5%) throughout the decade from $6.5 billion in 2000 to $1.6 billion in 2009 as the L2 cache became integrated on the CPU and as low power DRAM replaces SRAM due to its lower cost structure.

After the bursting of the Internet bubble, DRAM declined by over 61% in 2001 before climbing back to $33.8 billion in 2006. The latter half of the decade saw excessive oversupply resulting in a decline in revenues to $21.2 billion in 2009.

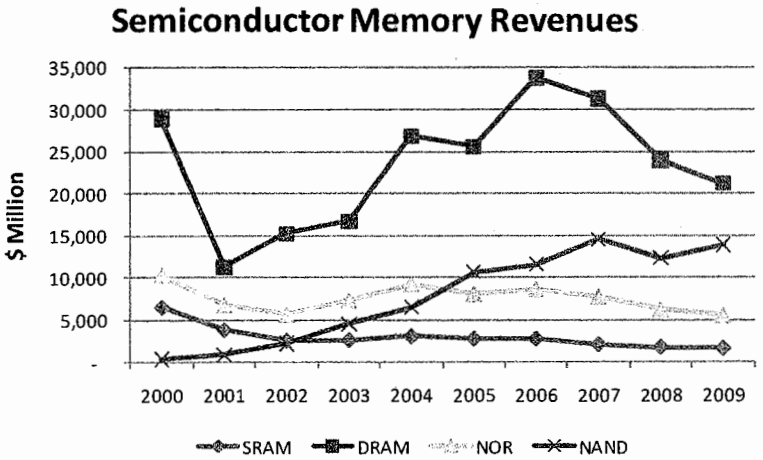


Fig. 1.1. Semiconductor memory revenues (Source: WSTS, Forward Insights)

* Forward Insights, greg@forward-insights.com

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_1,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0011

The NOR Flash market has shrunk almost 50% from $10.3 billion in 2000 to $5.5 billion in 2009 as NAND Flash penetrated the cell phone market in the latter half of the decade.

NAND Flash experienced some volatility during the decade but its extremely low cost structure has allowed it to enable new applications such as digital photography, portable storage, portable audio and video and multi-functional portable devices. NAND Flash started the decade at $370 million and surpassed NOR Flash revenues in 2005. It grew 37 times to close out the decade at $13.8 billion.

1.2 Flash memory architectures

Flash memory was invented by Dr. Fujio Masuoka of Toshiba Corp. in 1984. Based on Masuoka's invention, Intel Corp. commercialised common ground NOR Flash memory in 1988 seeing a non-volatile storage medium to storage program codes including a PC's BIOs and firmware for various consumer products. NOR Flash was also the basis for the first Flash memory cards and non-volatile solid state drives in the 1990s.

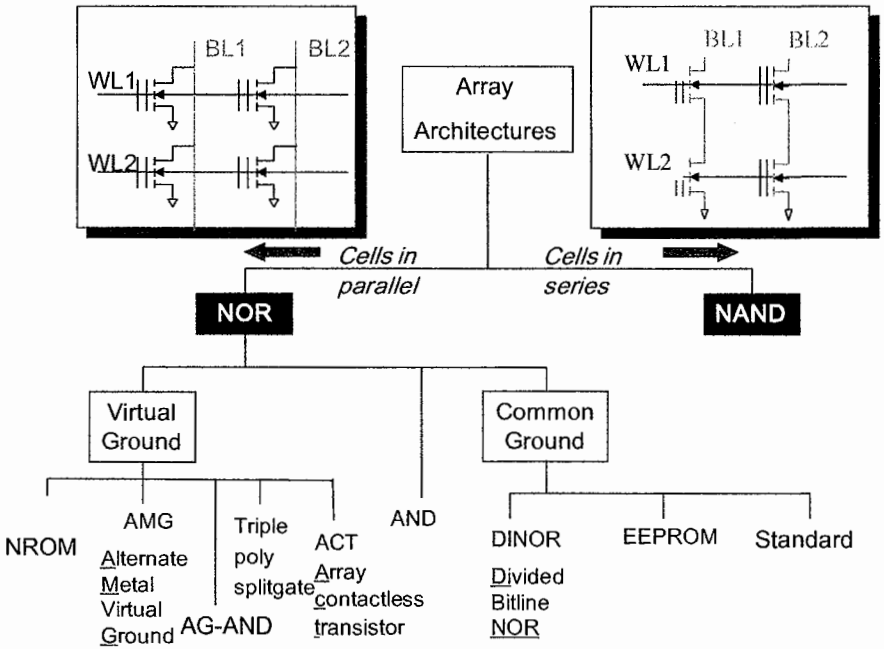


Fig. 1.2. Flash memory array architectures (Source: Forward Insights)

Petitioner Micron Ex-1021, 0012

Toshiba Corp. introduced NAND Flash memory in 1988 which promised lower cost per bit than NOR Flash and faster program and erase throughput. Unlike NOR Flash which is organized on a byte or word basis, NAND Flash is organized into pages and erased on a block basis. A block consists of 64 or more pages. The organization of the NAND Flash is conducive to lower cost per bit but not suitable for random access. NAND Flash is therefore employed as a data storage medium similar to optical media and hard disk drives.

The two main array architectures are summarized in Fig. 1.2. NOR Flash memory employs a parallel array architecture where each cell may be accessed via a contact (Fig. 1.3). The direct cell access is the reason for the superior random performance of NOR Flash. The NOR array may be classified into the virtual ground or common ground array. A variety of device variations of these two classification have been introduced by various companies such as Saifun's NROM, Sharp's ACT and Renesas' AND, AG-AND and DINOR, however the mainstream NOR Flash is the common ground implementation manufactured by the likes of Numonyx, Samsung and Winbond.

In contrast, the memory cells in NAND Flash are organized serially. Figure 1.4 shows 32 memory cells sandwiched between two large select transistors and two contacts. Random performance is slow due to the fact that there are no contacts directly accessing the memory cells. However, because there are only two contacts every 32 memory cells, the effective cell size is much smaller than for NOR Flash resulting in a smaller chip size and lower cost per bit. The cell size of NAND Flash is generally in the 4 F^2 range where F is the design rule of the chip. Due to the parallel architecture of NOR Flash, its cell size is relatively large at 10 F^2.

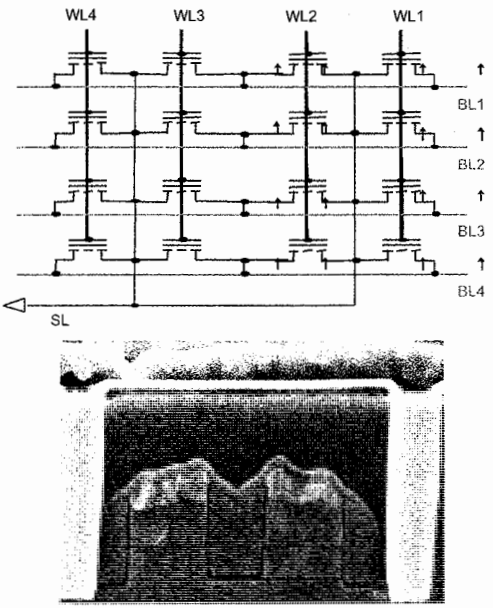


Fig. 1.3. NOR array architecture (Source: Forward Insights)

Petitioner Micron Ex-1021, 0013

The device characteristics of NAND and NOR Flash memories are compared in Fig. 1.5. NOR Flash exhibits superior random read and write performance versus NAND Flash.

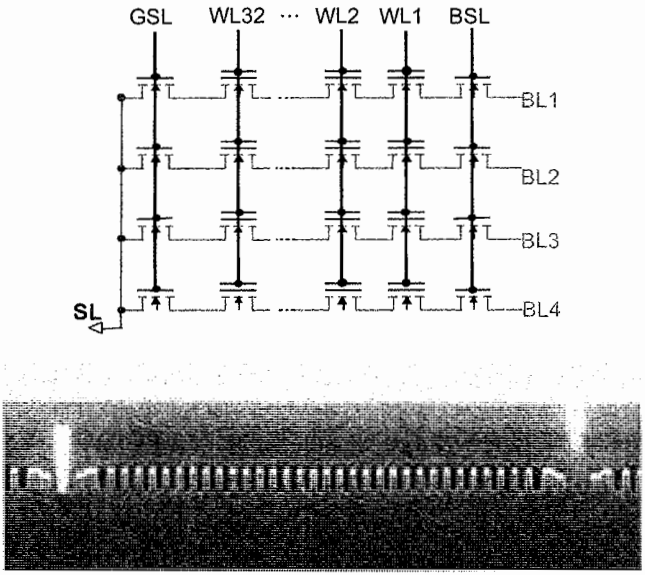


Fig. 1.4. NAND array architecture (Source: Forward Insights)

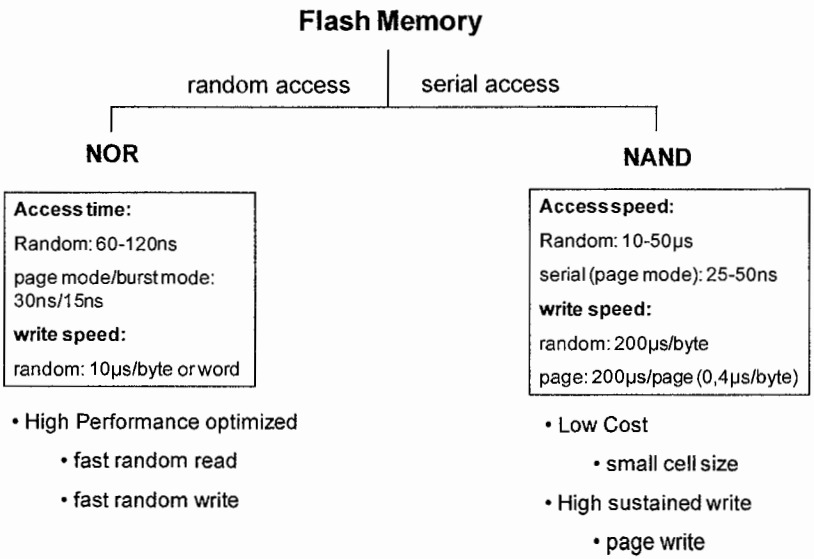


Fig. 1.5. NAND versus NOR (Source: Forward Insights)

Petitioner Micron Ex-1021, 0014

However, NAND exhibits fast page writes due to the ability to write 4–8 kB simultaneously resulting in very high sequential write throughput. The serial architecture and small cell size make NAND Flash optimized for low cost mass storage whereas NOR Flash is optimized for performance code storage and execution.

1.3 Multi-bit per cell storage

1.3.1 Memories scaling

Where other semiconductor memories were on a 2 year cadence for new process technology introduction, NAND Flash memories have historically been on a 1 year cadence. This accelerated process scaling resulted in the bit size of SLC NAND overtaking MLC NOR in 2005 as shown in Fig. 1.6. MLC NAND is by far the lowest cost semiconductor memory with none of the memory technologies even close to being cost competitive. This is mainly due to the very small cell size combined with multi-level cell capability.

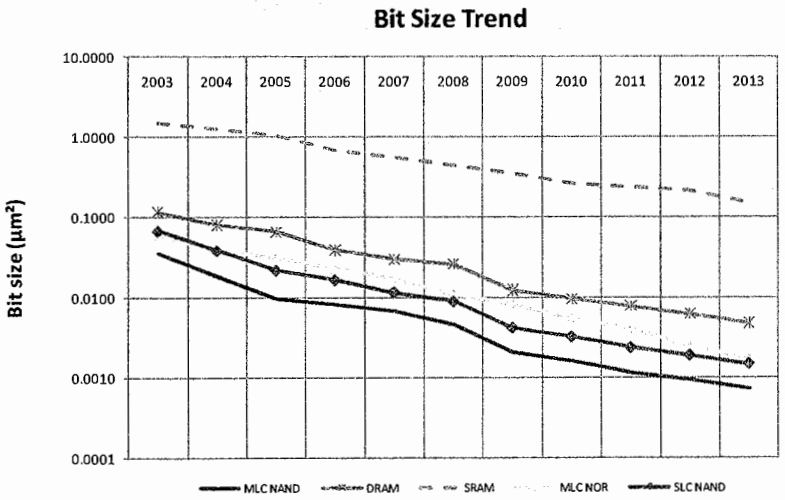


Fig. 1.6. Scaling (Source: Forward Insights)

1.3.2 Multi-level cell concept

Figure 1.7 illustrates the concept of multi-level cell storage in Flash memories. Conventional SLC or single-level cell storage distinguishes between a '1' and '0' by having no charge or charge present on the floating gate of the Flash memory cell. By increasing the number of charge or voltage threshold (Vt) levels, more

Petitioner Micron Ex-1021, 0015

than 1-bit per cell may be stored. Two bits per cell (MLC) storage is enabled by increasing the number of Vt levels to four representing 11, 10, 01 and 00. Similarly by increasing the number of voltage threshold levels to eight and 16, 3-bit per cell and 4-bit per cell storage is enabled.

The benefit of multi-level cell storage is that storage capacity may be increased without a corresponding increase in process complexity. The same fab equipment used to manufacture silicon wafers for SLC products may be used to manufacture MLC, 3-bit per cell and 4-bit per cell devices. However, multi-level cell storage requires accurate placement of the Vt charge levels so that the charge distributions don't overlap as well as accurate sensing of the different charge levels. As the number of Vt levels increases the time it takes for accurate programming and sensing increases. Additional circuitry and programming algorithms are necessary to ameliorate the degradation of the performance and endurance of such devices.

However in effect, transitioning from SLC to MLC to 3-bit to 4-bit per cell technology is equivalent to a partial shrink of the device from one process technology generation to the next without additional capital investment.

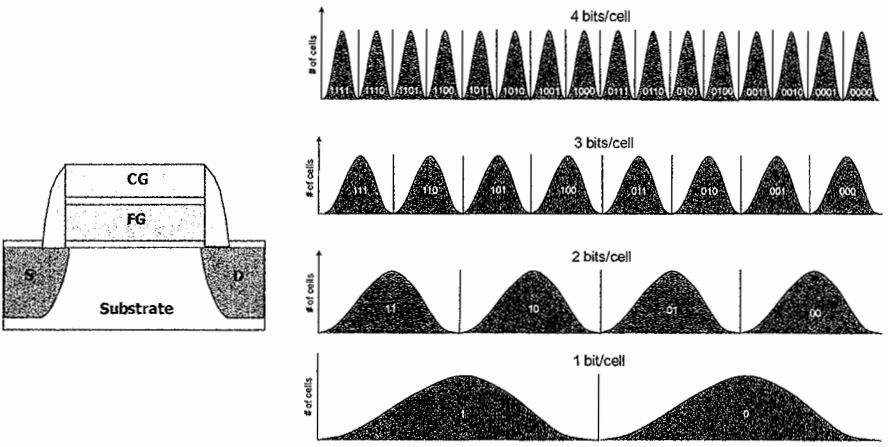


Fig. 1.7. Multi-level storage in floating gate NAND Flash memory

1.3.3 NAND scaling

Figure 1.8 shows the bit size trend for SLC, MLC, 8LC (3-bit per cell) and 16LC (4-bit per cell) technologies. The bit size is a proxy for the cost. The first MLC NAND Flash chip was introduced at the end of 2001 by SanDisk and Toshiba. It was a 1 Gb chip based on 0.16 µm process technology. Subsequent MLC generations were introduced on an almost yearly basis.

The first commercial production of 8LC began in 2008 also by SanDisk and Toshiba. The device, a 16 Gb product based on 56 nm process technology was introduced at a process technology one generation behind the mainstream MLC

Petitioner Micron Ex-1021, 0016

products. As a result, the device was short-lived since the MLC products based on 43 nm technology were more cost-competitive than the 8LC product. However in 2010, the cost benefits of 8LC technology are expected to be realized as 8LC is manufactured on the same process technology as MLC.

As with 8LC, 16LC also from SanDisk and Toshiba was introduced in 2009 on a mature process technology – 43 nm. As a result, the first product – a 64 Gb chip – is not cost competitive with the mainstream 32 nm MLC products. It is mainly being used as a learning vehicle. Only when 16LC can be manufactured on the leading edge process technology will the full cost benefits be realized.

Note that the cost per bit reduction becomes progressively smaller as one transitions from SLC to MLC to 8LC to 16LC. An approximate 40–50% reduction can be obtained moving from SLC to MLC but this figure drops to 20% for MLC to 8LC and 10% for 8LC to 16LC. As a result, the economic benefit of 16LC may not be enough to justify the additional design efforts to implement it.

One of the main scaling challenges is that the number of electrons stored in the floating gate is decreasing significantly with each process generation. This has consequences for the sensing of the data value stored and data retention. These issues combined with inter-cell interference will make 16LC a less scalable technology. To overcome these challenges, innovative programming algorithms and signal processing techniques filter the signal from the noise will be required for future generations of NAND Flash.

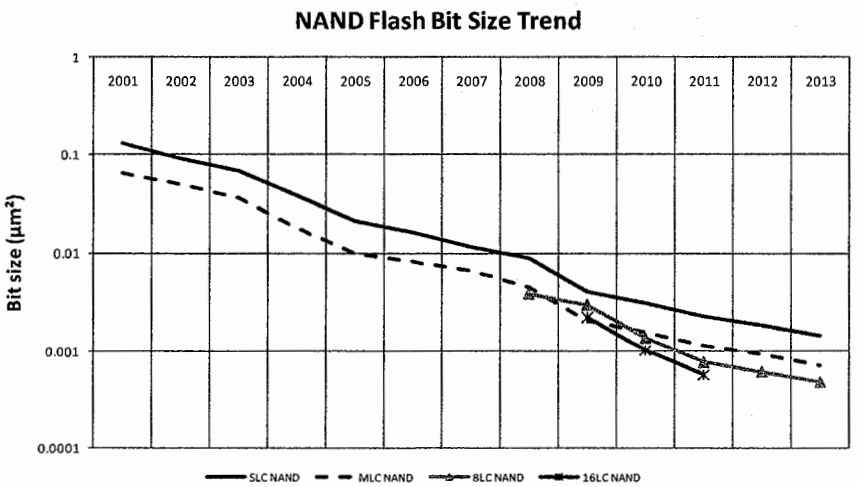


Fig. 1.8. NAND Flash bit size trend (Source: Forward Insights)

1.3.4 Capacity

Until 2006, MLC chip density was doubling every year. This doubling of chip density is being extended to 18 months to 2 years as the scaling challenges

Petitioner Micron Ex-1021, 0017

increase. As MLC became mainstream, SLC began to lag behind in process migration and density transition. For example, it will take 3 years for the transition from 8 to 16 Gb SLC chips.

The benefit of 8LC and 16LC is the ability to enable monolithic chip densities which would not be possible with SLC and MLC. For example, the first 16LC device was the first monolithic chip with a capacity of 64 Gb. It is expected that 8LC and 16LC will continue to lead the introduction of the highest capacity devices (Fig. 1.9).

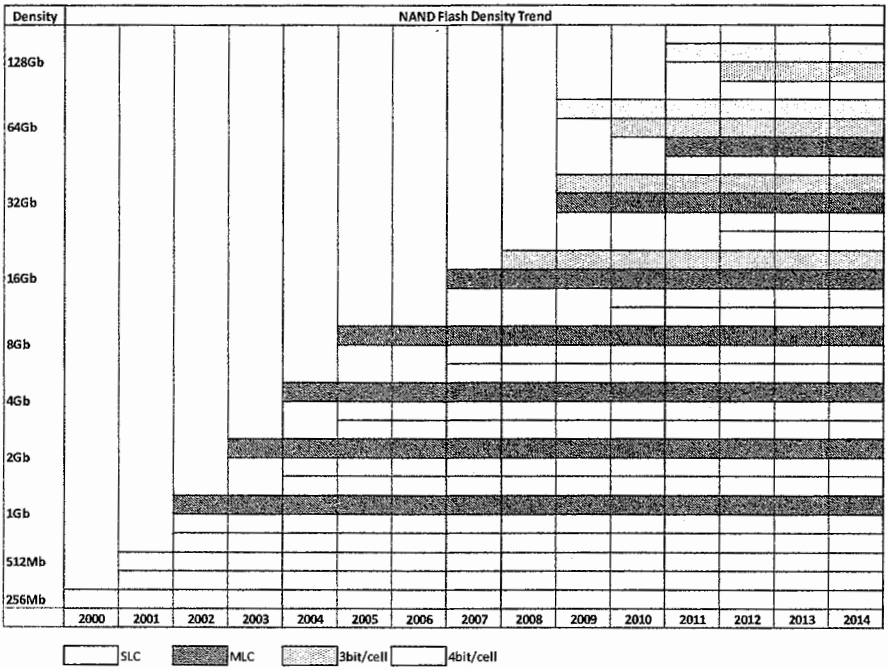


Fig. 1.9. NAND Flash density roadmap (Source: Forward Insights)

1.3.5 Device characteristics

The lower costs associated with multi-level cell technology does not come for free. The lower cost comes at the expense of reduced performance and endurance. A comparison of the device characteristics for SLC, MLC, 3- and 4-bit per cell is summarized in Fig. 1.10. Significant degradation in endurance, retention and write performance occurs with increasing number of bits per cell. Read performance and operating current are also negatively impacted.

Petitioner Micron Ex-1021, 0018

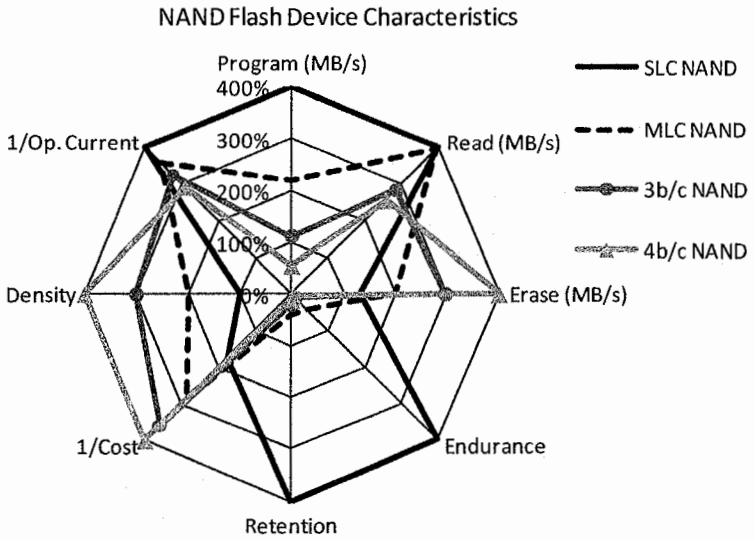


Fig. 1.10. NAND device characteristics (Source: Forward Insights)

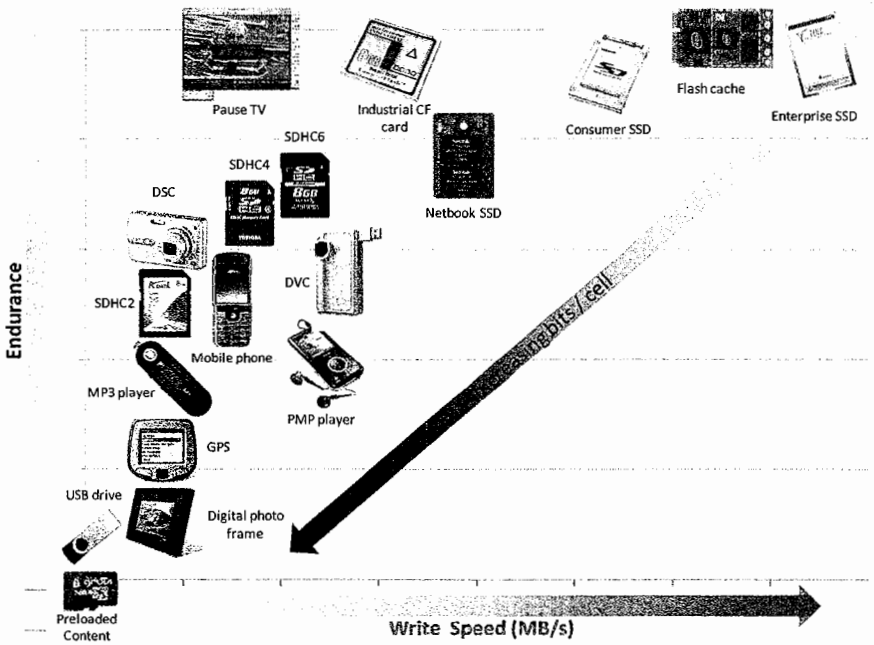


Fig. 1.11. Applications for multi-bit per cell NAND Flash memories (Source: Forward Insights)

Petitioner Micron Ex-1021, 0019

Of course, the trade-off is higher capacity and lower cost per bit. Erase performance actually improves due to the larger block size.

The implication of the disparate device characteristics is that the applications usage model will dictate the type of devices that will be used.

Figure 1.11 provides a simplified overview of the applications based on two key parameters – endurance and write speed. Applications with less demanding endurance and write performance requirements are better suited for devices with multi-bit per cell storage. Content storage and archiving may be facilitated by 4-bit per cell devices. Consumer applications that are largely read intensive such as MP3/PMP players, Flash memory cards, personal navigation devices can use 3-bit per cell devices. MLC may be found in consumer and netbook solid state drives and SLC for solid state drives in the enterprise. However, for certain enterprise applications especially read intensive with low duty cycles, MLC may be sufficient.

1.4 Market and applications

1.4.1 Removable portable storage

The application which triggered the rise of NAND Flash was digital photography and digital storage. The first Flash memory card format, Compact Flash I incorporated NOR Flash at its inception in 1994. The SmartMedia card introduced in 1995 was the first removable media format to incorporate NAND Flash followed by the MultiMedia Card in 1997, Memory Stick in 1998, Secure Digital in 1999, and xD-Picture Card in 2002. Miniaturized "mini" and "micro" versions of these cards have been subsequently introduced.

Figure 1.12 shows the plethora of Flash memory card form factors that have replaced analog film. In fact, Flash memory cards store not only images but also audio and video enabling portable transport of files from one device to another. USB Flash drives introduced in 2000 by Trek and IBM have driven the floppy disk to extinction.

1.4.2 Embedded storage

NAND Flash is employed for code/data storage or data storage in a variety of portable and mobile applications such as cellular phones, MP3/PMP players, digital video camcorders and personal navigation devices. NAND Flash may be embedded in applications in several ways.

Raw NAND – NAND Flash chips are soldered onto the PCB of the device with the host-side handling the Flash translation layer (FTL), bad block management and error correction (ECC). For example, MP3/PMP players employ a Flash controller to manage the raw NAND.

Petitioner Micron Ex-1021, 0020

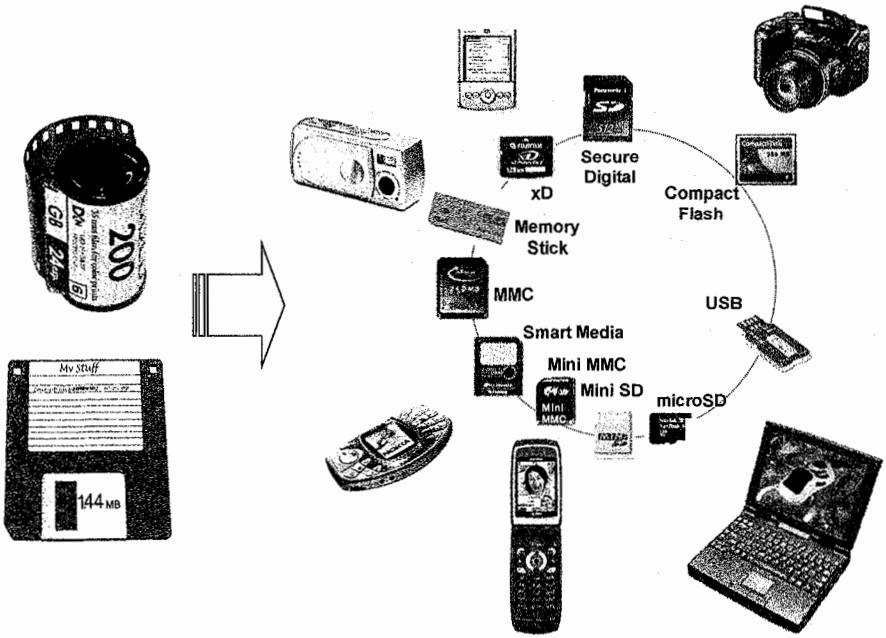


Fig. 1.12. Removable portable storage

NAND with on-chip controller – The NAND Flash devices are used for code and data storage with only the Flash translation layer managed by the host. In some instances, these devices such as ONENAND from Samsung, Toshiba and Numonyx and mDOC (mobile Disk-on-Chip) from SanDisk come with a NOR Flash interface and act as a replacement for the NOR Flash.

Multi-chip Package (MCP) – The NAND Flash is combined with both a NOR Flash and low power DRAM or just a DRAM for code and data storage. MCPs are primarily used in mobile phones.

An overview of memory architectures for different mobile phones is presented in Fig. 1.13. The architectures depend on the functionality of the phones.

The primary function of entry level mobile phones is voice communications. The phones are generally capable of SMS and basic web browsing. As a result, large memory capacity is not a requirement. Entry level mobile phones employ a XIP (execute in place) architecture whereby the program code is stored and directly executed from the NOR Flash and pseudo-SRAM is used as the working memory. This memory system is cost optimized solution for small memory densities and low power.

Feature phones provide voice, MMS, camera, audio, video and web browsing functionality. Higher capacity for data storage is therefore necessary. Feature phones combine three memories – NOR Flash for code execution, NAND Flash for data storage and RAM as a working memory. This architecture is essentially a XIP with additional NAND Flash for data storage.

Petitioner Micron Ex-1021, 0021

Smartphones have morphed into convergence devices. In addition to the feature phone capabilities, these devices are designed for multi-tasking – running more than one application at the same time, e-mailing and infotainment.

The store and download (SnD) architecture used in smartphones employs a PC-like approach. Instead of storing in NOR Flash, the code is stored in NAND Flash, copied into DRAM and executed from DRAM. In this scenario, code has to be swapped in and out of the DRAM for execution. Since the RAM has to act as both a code execution and work memory, the RAM capacity is larger than the RAM required in the NOR/NAND/RAM implementation in feature phones. Because of the very low cost per bit of NAND Flash, this architecture is cost optimized for medium/large memory capacity requirements at the expense of higher power consumption and longer boot time than the XIP implementation.

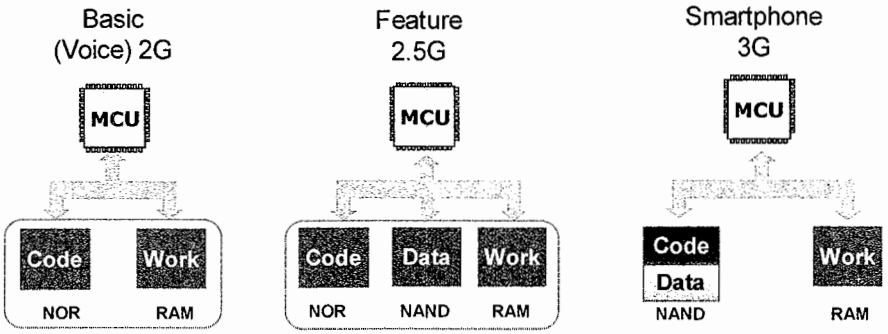


Fig. 1.13. Memory system architectures in mobile phones

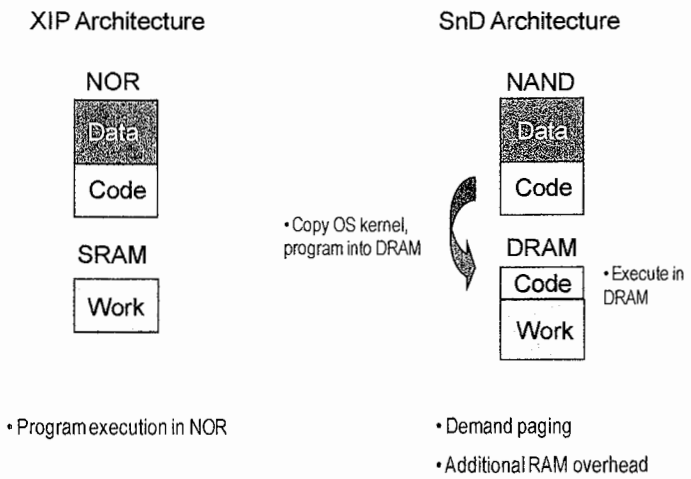


Fig. 1.14. Store and download versus XIP architecture

Petitioner Micron Ex-1021, 0022

Figure 1.14 contrasts the SnD and XIP architectures.

Managed NAND solutions – NAND Flash device ECC requirements and in some cases page sizes and block sizes are changing with each new technology generation. Instead of having to continually update the host firmware to manage these details, these changes are managed within the device by combining raw NAND and a controller in a package. All the FTL, bad block management and ECC are handled within the package allowing the host to perform only a simplified read/write to the device. The host side requires a basic driver to interface with the managed NAND device typically using a eMMC or eSD protocol. The raw NAND and managed NAND implementations are illustrated in Fig. 1.15.

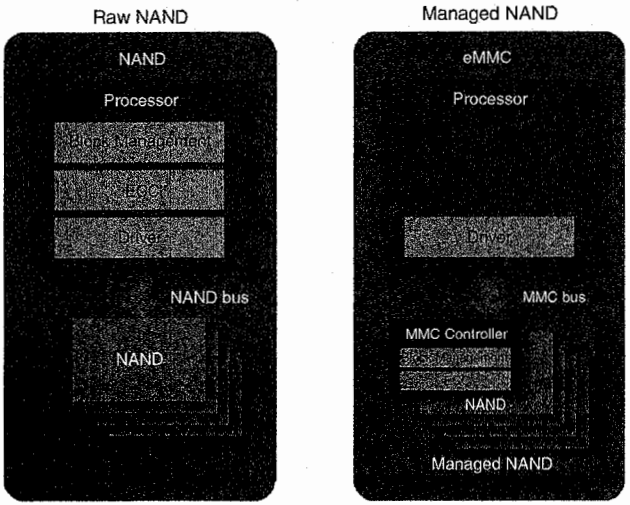


Fig. 1.15. Managed NAND versus raw NAND (Source: Micron Technology)

1.4.3 Solid state drives

The first SSDs pioneered by StorageTek in 1978 were RAM-based SSDs. It wasn't until late 1980s and early 1990s when the first Flash-based SSDs were developed. Western Digital demonstrated a 2.5″ NAND Flash SSD in 1989, however the main promoters of Flash-based SSDs, SanDisk and Intel, based their SSDs on NOR Flash technology in the early 1990s. Due to the higher cost of NOR Flash versus DRAM, Flash-based SSDs were relegated to niche markets.

However, the remarkable and rapid price decline of NAND Flash memory in recent years which has driven analog film and the floppy disk to extinction is now enabling NAND Flash memory to penetrate the realm of hard disk drives (HDDs). Solid state drives (SSDs) offer the promise of significantly better performance and reliability due to the lack of mechanical parts.

Petitioner Micron Ex-1021, 0023

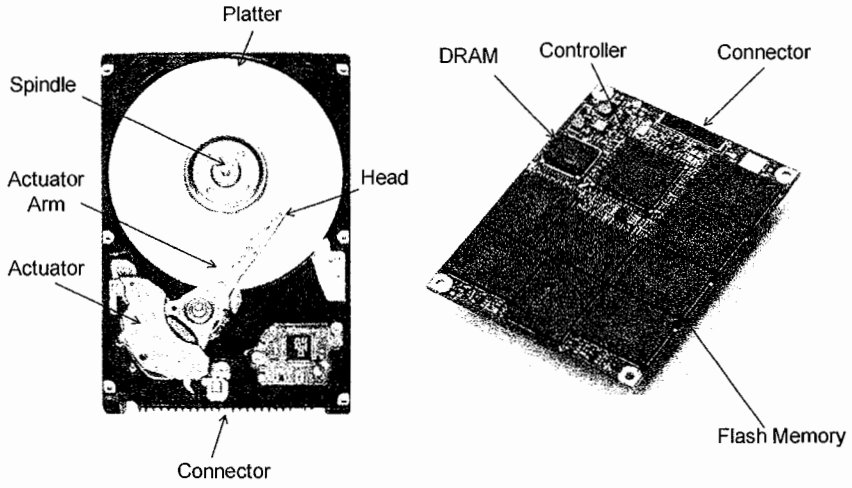


Fig. 1.16. Solid state drive versus hard disk drive (Source: Toshiba Corp., Forward Insights)

Figure 1.16 details the major components of an SSD and HDD. The HDD, being based on storage in a spinning magnetic platter, requires an actuator and actuator arm to move the head to the appropriate sector to be read or written. This movement of the read head results in extremely long latencies in the milliseconds and a DRAM buffer is used to hide the seek time. The HDD controller, particularly if it is a system-on-chip, incorporates the processor, servo control logic, interface, error correction code, disk sequencer and buffer controller.

In contrast, the SSD contains no mechanical parts and consists of a few major components: NAND Flash memory, SSD controller, connector, DRAM, PCB and passives. In addition, because of the small size of NAND Flash memory, the SSD form factor is not limited to standard 1.8, 2.5 or 3.5″ HDD form factors but can also come in module form.

NAND-based SSDs started out in industrial and military where ruggedness and reliability are a priority. In 2006, the first SSD for personal computers was introduced followed by SSDs for enterprise computing.

Table 1.1 summarizes the NAND Flash/SSD usage models in the different market segments. There are several ways NAND Flash or SSDs may utilized. Firstly, it can act as an alternate storage device, in many cases replacing a HDD for latency or reliability improvement and power consumption reduction. An optimized OS or file-system is required to obtain the full benefits of the SSD.

Secondly, the SSD may act as an I/O accelerator. I/O acceleration is primarily used in the enterprise computing environment. A software driver is required for this approach.

Thirdly, the SSD/NAND Flash may be part of the tiered system memory and replace part of the DRAM. Due to the better economics of NAND Flash versus DRAM, this allows an increase in system memory storage capacity with the

Petitioner Micron Ex-1021, 0024

benefits of reduced power consumption. To manage this tiered system memory, the kernel memory manager needs to be designed to manage the Flash memory.

A comparison of SSD and HDD in PCs (Fig. 1.17) shows that SSDs are superior to HDDs in all aspects except for cost and capacity. The cost and capacity are related since the major impediment to higher SSD capacities is the high cost per gigabyte.

Table 1.1. NAND usage model

NAND Usage Model	Consumer	Enterprise	Industrial	Military
Alternate Storage	√	√	√	√
I/O Accelerator		√		
Extended System Memory	√	√		

In the enterprise space, high throughput and low access latency are critical. Enterprise HDDs employ several techniques to improve system throughput and decrease the access latency. High performance 15 k rpm HDDs may be organized in a RAID configuration with the data being striped across multiple disks thereby increasing the bandwidth of the total system. Adding HDDs increase system bandwidth but result in increased space, power and cooling requirements.

"Short-stroking" the HDDs whereby data is placed on the outer tracks of the disk to reduce the seek time of the mechanical heads reduces system access latencies. However, this entails that only part of the HDD is being used to achieve the required performance resulting in excess storage capacity than would otherwise be necessary. As illustrated in Fig. 1.18, one SSD can replace several striped, short-stroked HDDs.

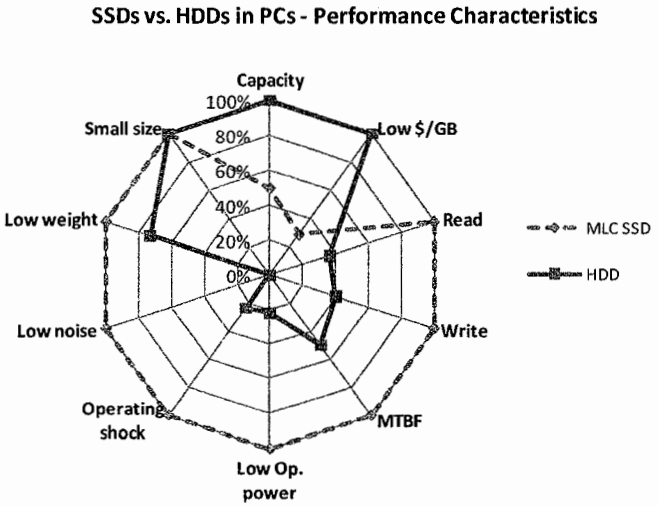


Fig. 1.17. 2.5″ SSDs versus HDDs in PCs – performance characteristics (Source: Forward Insights)

Petitioner Micron Ex-1021, 0025

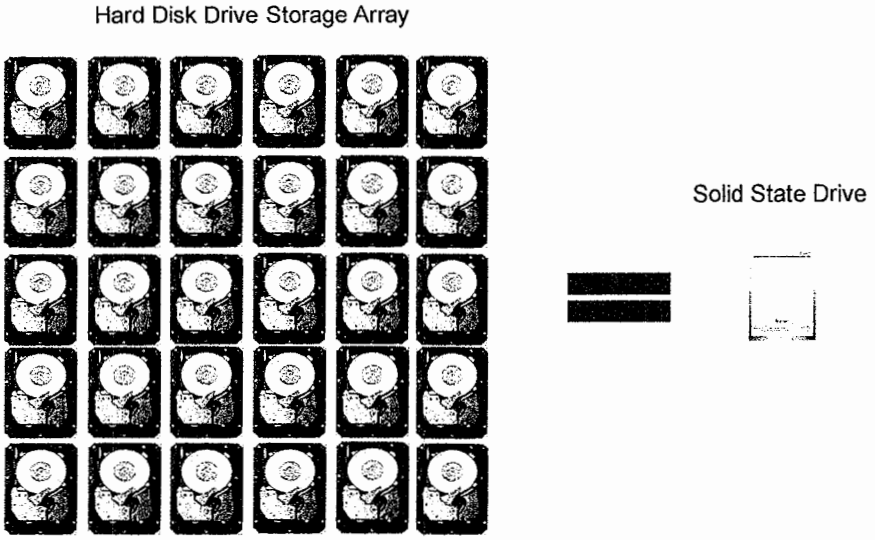


Fig. 1.18. Enterprise SSD value proposition

Figure 1.19 shows that although on a price per GB basis, HDDs are far superior to SSDs, the cost for an HDD system to obtain comparable metrics in terms of $/IOPS, IOPS/GB or IOPS/W is far higher than for SSDs. This is just the storage comparison. When the total system is considered, far fewer SSDs will be required for a given performance than HDDs. If the maintenance, power and space savings are included, the economics of using SSDs are even more attractive.

SPECIFICATION	Enterprise Class SSD[1]	Enterprise Class HDD	Delta
Capacity (GB)	146	73	100%
Price ($)	12,500	389	3113%
Performance			
Random			
Read	52,000 IOPS	300 IOPS (est)	17233%
Write	18,000 IOPS	300 IOPS (est)	5900%
Power			
Operating mode	8.4W	10-12W	-24%
$ / GB	85.62	5.33	1607%
$ / IOPS			
Read	0.24	1.30	-81%
Write	0.69	1.30	-46%
IOPS / GB			
Read	356	4	8567%
Write	123	4	2900%
IOPS / W			
Read	6,190	27	22598%
Write	2,143	27	7757%

[1]STEC ZeusIOPS

Fig. 1.19. Enterprise SSD versus HDD key metrics

Petitioner Micron Ex-1021, 0026

1.5 Market outlook

In the early years, the NAND Flash market was driven by removable storage applications such as the Flash memory card following by the USB Flash drive. In 2004, NAND Flash began to replace 1″ HDD in MP3 players led by Apple's iPod Mini. The era of high capacity embedded Flash storage in mobile phones began with the 4 and 8 GB iPhone from Apple Corp. in 2007. Prior to 2007, smartphones only had relatively low density Flash on-board and in some models, high density storage was supplemented via a slot which allowed the insertion of a Flash memory card. The iPhone is not just a phone but a convergence device incorporating camera, video, audio and gaming functionalities supported by content and applications from an online store.

Following the iPhone's lead, smartphone from other handset manufacturers are increasing the embedded Flash memory storage of their devices and mobile market is forecast to become the major consumer of NAND Flash gigabytes within the next 5 years.

The next frontier after consumer and communications is computing. NAND Flash-based SSDs are expected to become a new driver of NAND Flash bit consumption beyond 2012 driven by adoption of SSDs in mainstream consumer PCs.

The NAND Flash industry's historical triple digit bit growth ended in 2009 as a result of the global economic downturn. As technology transitions slow down, bit growth is forecast to experience a more moderate 69% CAGR from 2007 to 2016 (Figs. 1.20 and 1.21).

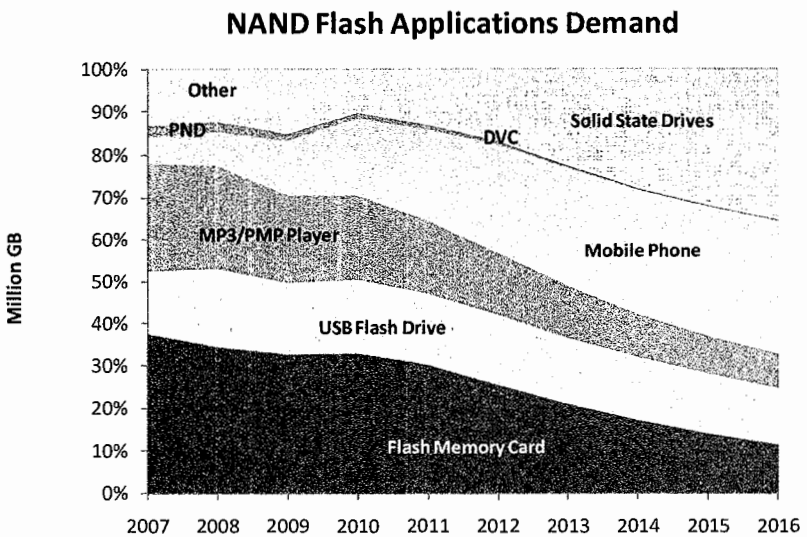


Fig. 1.20. NAND Flash applications demand (% GB) (Source: Forward Insights)

Petitioner Micron Ex-1021, 0027

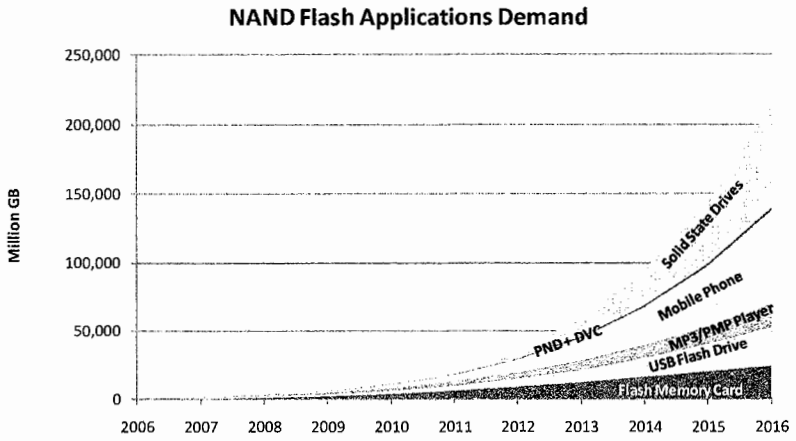


Fig. 1.21. NAND Flash applications demand (Source: Forward Insights)

NAND Flash memory is facing serious scaling issues and it is within the next 5 years, that a successor technology is expected take the reins and fuel the insatiable storage demands of emerging applications.

Petitioner Micron Ex-1021, 0028

2 NAND overview: from memory to systems

R. Micheloni[1], *A. Marelli*[2] *and S. Commodaro*[3]

2.1 Introduction

It was in 1965, just after the invention of the bipolar transistor by W. Shockley, W. Brattain and J. Bardeen, that Gordon Moore, co-founder of Intel, observed that the number of transistors per square centimeter in a microchip doubled every year. Moore thought that such trend would have proven true for the years to come as well, and indeed in the following years the density of active components in an integrated circuit kept on doubling every 18 months. For example, in the 18 months that elapsed between the Pentium processor 1.3 and the Pentium-4, the number of transistors grew from 28 to 55 million.

Today, a standard desktop PC has processors whose operating frequency is in the order of some gigahertz, while its memory can store as much information as terabytes.

In this scenario, a meaningful portion of the devices produced is represented by memories, one of the key components of any electronic systems.

Semiconductor memories can be divided into two major categories: RAM, acronym for *Random Access Memories*, and ROM, acronym for *Read Only Memories*: RAM loses its content when power supply is switched off, while ROM virtually holds it forever. A third category lies in between, i.e. NVM, acronym for *Non-Volatile Memories*, whose content can be electrically altered but it is also preserved when power supply is switched off. These are more flexible than the original ROM, whose content is defined during manufacturing and cannot be changed by the consumer anymore.

The history of non-volatile memories began in the 1970s, with the introduction of the first EPROM memory (*Erasable Programmable Read Only Memory*). Since then, non-volatile memories have always been considered one of the most important families of semiconductors and, up to the 1990s, their interest was tied up more to their role as a product of development for new technologies than to their economic value. Since the early 1990s, with the introduction of non-volatile Flash memories into portable products like mobile phones, palmtop, camcorders, digital cameras and so on, the market of these memories has experienced a stunning increase.

[1] Integrated Device Technology, rino.micheloni@ieee.org
[2] Integrated Device Technology, alessiamarelli@gmail.com
[3] Pegasus MicroDesign, stefano_commodaro@yahoo.it

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_2,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0029

The most popular Flash memory cell is based on the *Floating Gate* (FG) technology, whose cross section is shown in Fig. 2.1. A MOS transistor is built with two overlapping gates rather than a single one: the first one is completely surrounded by oxide, while the second one is contacted to form the gate terminal. The isolated gate constitutes an excellent "trap" for electrons, which guarantees charge retention for years. The operations performed to inject and remove electrons from the isolated gate are called program and erase, respectively. These operations modify the threshold voltage V_{TH} of the memory cell, which is a special type of MOS transistor. Applying a fixed voltage to cell's terminals, it is then possible to discriminate two storage levels: when the gate voltage is higher than the cell's V_{TH}, the cell is on ("1"), otherwise it is off ("0").

It is worth mentioning that, due to floating gate scalability reasons, charge trap memories are gaining more and more attention and they are described in Chap. 5, together with their 3D evolution.

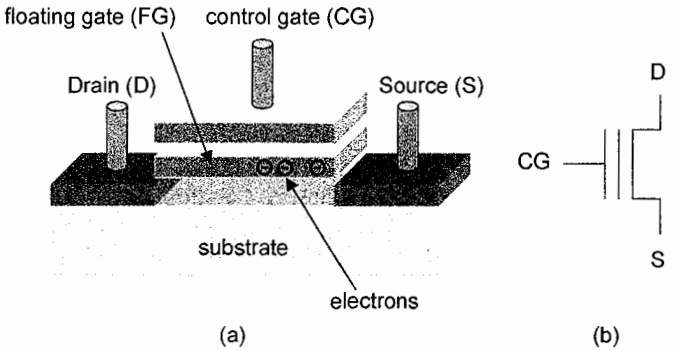


Fig. 2.1. (a) Floating gate memory cell and (b) its schematic symbol

2.2 NAND memory

2.2.1 Array

The memory cells are packed to form a matrix in order to optimize silicon area occupation. Depending on how the cells are organized in the matrix, it is possible to distinguish between NAND and NOR Flash memories. This book is about NAND memories as they are the most widespread in the storage systems. NOR architecture is described in great details in [1].

In the NAND string, the cells are connected in series, in groups of 32 or 64, as shown in Fig. 2.2. Two selection transistors are placed at the edges of the string, to ensure the connections to the source line (through M_{SL}) and to the bitline (through M_{DL}). Each NAND string shares the bitline contact with another string. Control gates are connected through wordlines (WLs).

Petitioner Micron Ex-1021, 0030

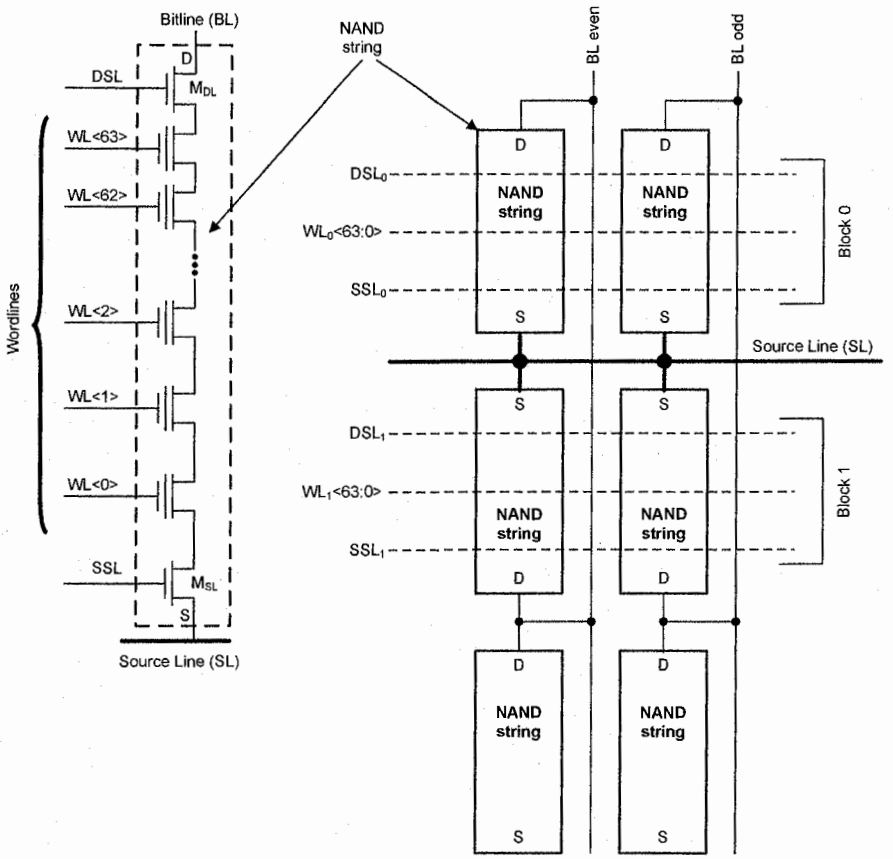


Fig. 2.2. NAND string (left) and NAND array (right)

Logical pages are made up by cells belonging to the same wordline. The number of pages per wordline is related to the storage capabilities of the memory cell. Depending on the number of storage levels, Flash memories are referred to in different ways: SLC memories store 1 bit per cell, MLC memories (Chap. 10) store 2 bits per cell, 8LC memories (Chap. 16) store 3 bits per cell and 16LC memories (Chap. 16) store 4 bits per cell.

If we consider the SLC case with interleaved architecture (Chap. 8), even and odd cells form two different pages. For example, a SLC device with 4 kB page has a wordline of 65,536 cells.

Of course, in the MLC case there are four pages as each cell stores one *Least Significant Bit* (LSB) and one *Most Significant Bit* (MSB). Therefore, we have:

- MSB and LSB pages on even bitlines
- MSB and LSB pages on odd bitlines

Petitioner Micron Ex-1021, 0031

All the NAND strings sharing the same group of wordlines are erased together, thus forming a Flash block. In Fig. 2.2 two blocks are shown: using a bus representation, one block is made up by WL_0<63:0> while the other one includes WL_1<63:0>.

NAND Flash device is mainly composed by the memory array. Anyway, in order to perform read, program, and erase additional circuits are needed. Since the NAND die must be inserted in a package with a well-defined size, it is important to organize all the circuits and the array in the early design phase, i.e. it is important to define a floorplan.

In Fig. 2.3 an example of a floorplan is given. The Memory Array can be split in different planes (two planes in Fig. 2.3). On the horizontal direction a Wordline is highlighted, while a Bitline is shown in the vertical direction.

The Row Decoder is located between the planes: this circuit has the task of properly biasing all the wordlines belonging to the selected NAND string (Sect. 2.2.2). All the bitlines are connected to sense amplifiers (Sense Amp). There could be one or more bitlines per sense amplifier; for details, please, refer to Chap. 8. The purpose of sense amplifiers is to convert the current sunk by the memory cell to a digital value. In the peripheral area there are charge pumps and voltage regulators (Chap. 11), logic circuits (Chap. 6), and redundancy structures (Chap. 13). PADs are used to communicate with the external world.

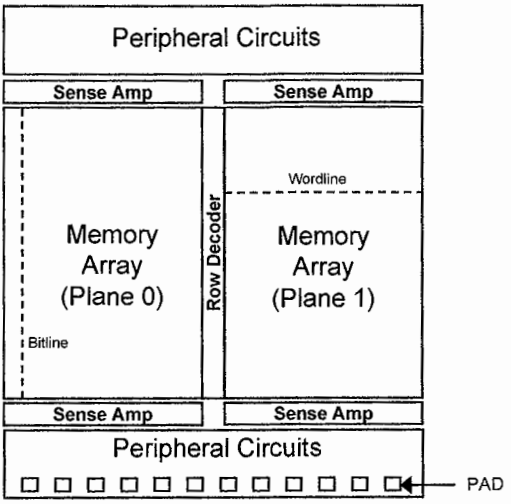


Fig. 2.3. NAND Flash memory floorplan

2.2.2 Basic operations

This section briefly describes the basic NAND functionalities: read, program, and erase.

Petitioner Micron Ex-1021, 0032

Read

When we read a cell (Fig. 2.4), its gate is driven at V_{READ} (0 V), while the other cells are biased at $V_{PASS,R}$ (usually 4–5 V), so that they can act as pass-transistors, regardless the value of their threshold voltages. In fact, an erased Flash cell has a V_{TH} smaller than 0 V; vice versa, a written cell has a positive V_{TH} but, however, smaller than 4 V. In practice, biasing the gate of the selected cell with a voltage equal to 0 V, the series of all the cells will conduct current only if the addressed cell is erased.

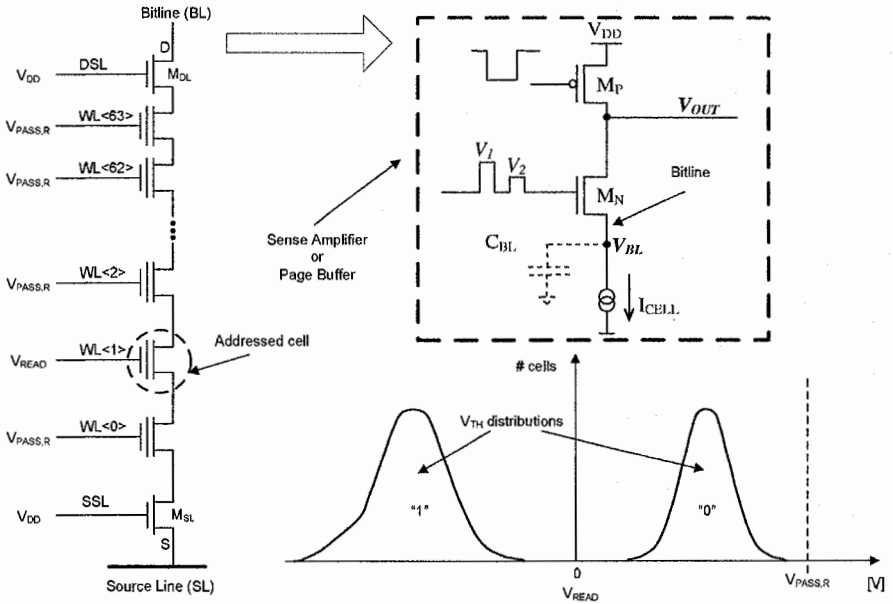


Fig. 2.4. NAND string biasing during read and SLC V_{TH} distributions

String current is usually in the range of 100–200 nA. The read technique is based on charge integration, exploiting the bitline parasitic capacitor. This capacitor is precharged at a fixed value (usually 1–1.2 V): only if the cell is erased and sinks current, then the capacitor is discharged. Several circuits exist to detect the bitline parasitic capacitor state: the structure depicted in the inset of Fig. 2.4 is present in almost all solutions. The bitline parasitic capacitor is indicated with C_{BL} while the NAND string is equivalent to a current generator.

During the charge of the bitline, the gate of the PMOS transistor M_P is kept grounded, while the gate of the NMOS transistor M_N is kept at a fixed value V_1. Typical value for V_1 is around 2 V. At the end of the charge transient the bitline will have a voltage V_{BL}:

$$V_{BL} = V_1 - V_{THN} \tag{2.1}$$

Petitioner Micron Ex-1021, 0033

where V_{THN} indicates the threshold voltage value of the NMSO M_N.

At this point, the M_N and M_P transistors are switched off. C_{BL} is free to discharge. After a time T_{VAL}, the gate of M_N is biased at $V_2 < V_1$, usually 1.6–1.4 V.

If a T_{VAL} time has elapsed long enough to discharge the bitline voltage under the value:

$$V_{BL} < V_2 - V_{THN} \tag{2.2}$$

M_N turns on and the voltage of node OUT (V_{OUT}) becomes equal to the one of the bitline. Finally, the analog voltage V_{OUT} is converted into a digital format by using simple latches.

A more detailed analysis of read techniques is presented in Chap. 8.

Program

Programming of NAND memories exploits the quantum-effect of electron tunneling in the presence of a strong electric field (Fowler–Nordheim tunneling [2]). In particular, depending on the polarity of the electric field applied, program or erase take place. Please refer to Chap. 3 for more details.

During programming, the number of electrons crossing the oxide is a function of the electric field: in fact, the greater such field is, the greater the injection probability is. Thus, in order to improve the program performances, it is essential to have high electric fields available and therefore high voltages (Chaps. 11 and 12). This requirement is one of the main drawbacks of this program method, since the oxide degradation is impacted by these voltages.

The main advantage is the current required, which is definitely low, in the range of nanoamperes per cell. This is what makes the Fowler–Nordheim mechanism suitable for a parallel programming of many cells as required by NAND page sizes.

The algorithm used to program the cells of a NAND memory is a *Program & Verify* algorithm (Chaps. 3 and 12): verify is used to check whether the cell has reached the target distribution or not.

In order to trigger the injection of electrons into the floating gate, the following voltages are applied, as shown in Fig. 2.5:

- V_{DD} on the gate of the drain selector
- $V_{PASS,P}$ (8–10 V) on the unselected gates
- V_{PGM} (20–25 V) on the selected gate (to be programmed)
- GND on the gate of the source selector
- GND on the bitlines to be programmed
- V_{DD} on other bitlines

The so-called *self-boosting* mechanism (Chap. 3) prevents the cells sharing the same gate with the programmed one from undergoing an undesired program. The basic idea is to exploit the high potentials involved during programming through

Petitioner Micron Ex-1021, 0034

the parasitic capacitors of the cell, in order to increase the potential of the region underneath the tunnel oxide (inset of Fig. 2.5).

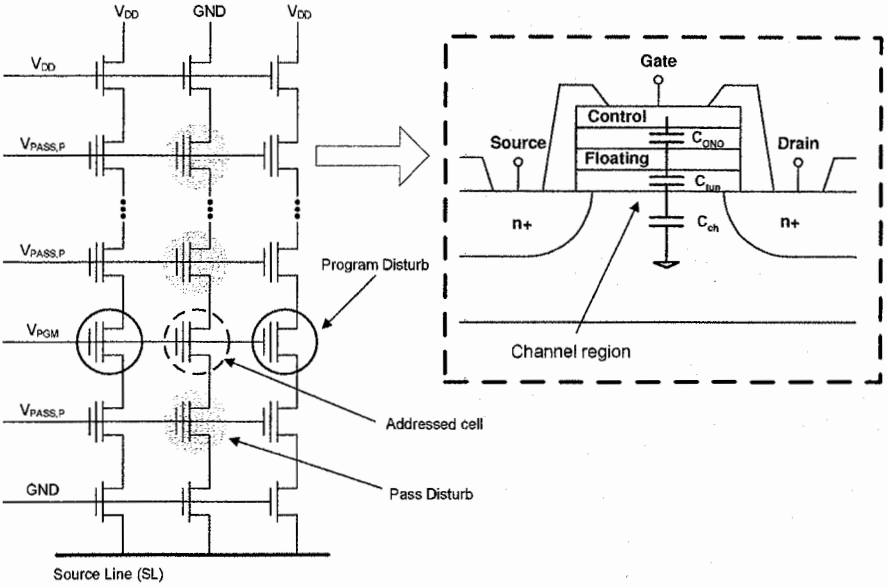


Fig. 2.5. NAND string biasing during program and parasitic capacitors contributing to the self-boosting inhibit

When the bitlines are driven to V_{DD}, drain transistors are diode-connected and the corresponding bitlines are floating. When $V_{PASS,P}$ is applied to the unselected wordlines, parasitic capacitors boost the potential of the channel, reducing the voltage drop across the tunnel oxide and, hence, inhibiting the tunneling phenomena.

As the memory cells are organized in a matrix, all the cells along the wordline are biased at the same voltage even if they are not intended to be programmed, i.e. they are "disturbed". Two important typologies of disturbs are related to the program operation: the *Pass disturb* and the *Program disturb,* as shown in Fig. 2.5: their impact on reliability is described in Chap. 4.

Erase

NAND memory is placed in a triple-well structure, as shown in Fig. 2.6a. Usually, each plane has its own triple-well. The source terminal is shared by all the blocks: in this way the matrix is more compact and the multiplicity of the structures which bias the iP-well is drastically reduced.

The electrical erase is achieved by biasing the iP-well with a high voltage and keeping grounded the wordlines of the sector to be erased (Fig. 2.6c). Therefore, NAND technologies don't need negative voltages. Again, the physical mechanism is the Fowler–Nordheim tunneling. As the iP-well is common to all the blocks,

Petitioner Micron Ex-1021, 0035

erase of unselected blocks is prevented leaving their wordlines floating. In this way, when the iP-well is charged, the potential of the floating wordlines raises thanks to the capacitive coupling between the control gates and the iP-well (Fowler–Nordheim tunneling inhibited).

Figure 2.6b sketches the erase algorithm phases. NAND specifications are quite aggressive in terms of erase time. Therefore, Flash vendors try to erase the block content in few erase steps (possibly one). As a consequence, a very high electric field is applied to the matrix during the *Electrical Erase* phase. As a matter of fact, erased distribution is deeply shifted towards negative V_{TH} values. In order to minimize floating gate coupling (Chap. 12), a *Program After Erase* (PAE) phase is introduced, with the goal of placing the distribution near the 0 V limit (of course, guaranteeing the appropriate read margin).

Typical erase time of a SLC block is about 1–1.5 ms; electrical erase pulse lasts 700–800 μs.

Technology shrink will ask for more sophisticated erase algorithms, especially for 3–4 bit/cell devices. In fact, reliability margins are usually shrinking with the technology node: a more precise, and therefore time consuming, PAE will be introduced, in order to contain the erased distribution width. In summary, erase time is going to increase to 4–5 ms in the new NAND generations.

Chapter 12 contains a detailed description of the whole erase algorithm.

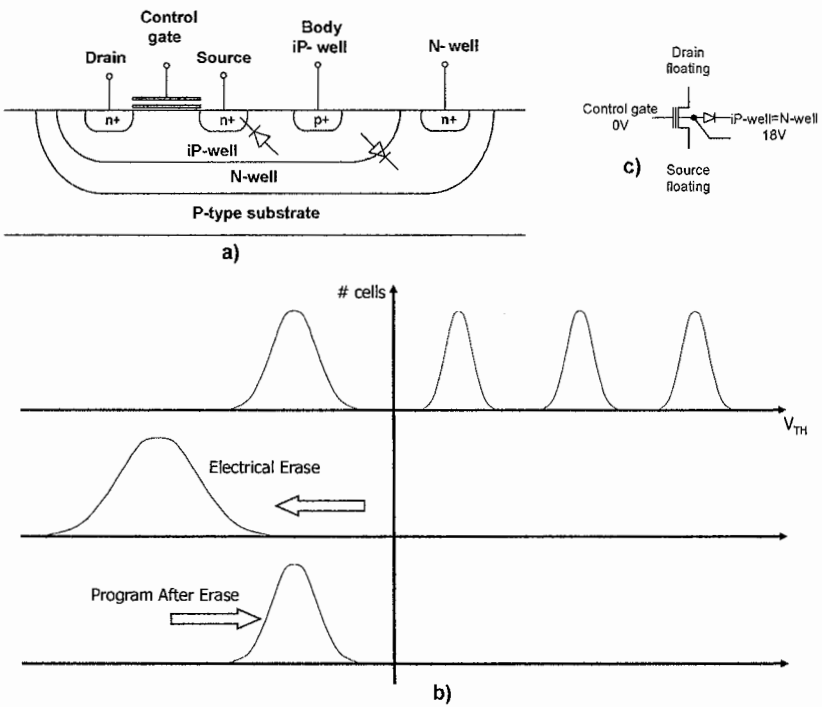


Fig. 2.6. (a) NAND matrix in triple-well; (b) erase algorithm; (c) erase biasing on the selected block

Petitioner Micron Ex-1021, 0036

2.2.3 Logic organization

NAND memory contains information organized in a specified way. Looking at Fig. 2.7, a memory is divided in pages and blocks. A block is the smallest erasable unit. Generally, there are a power of two blocks within any device. Each block contains multiple pages. The number of pages within a block is typically a multiple of 16 (e.g. 64, 128). A page is the smallest addressable unit for reading and writing. Each page is composed of main area and spare area (Fig. 2.8). Main area can range from 4 to 8 kB or even 16 kB. Spare area can be used for ECC (Chap. 14) and system pointers (Chap. 17) and it is in the order of a couple of hundreds bytes every 4 kB of main area.

Every time we want to execute an operation on a NAND device, we must issue the address where we want to act. The address is divided in row address and column address (Fig. 2.9). Row address identifies the addressed page, while column address is used to identify the bytes inside the page. When both row and column addresses are required, column address is given first, 8 bits per address cycle. The first cycle contains the least significant bits. Row and column addresses cannot share the same address cycle.

The row address identifies the block and the page involved in the operation. Page address occupies the least significant bits.

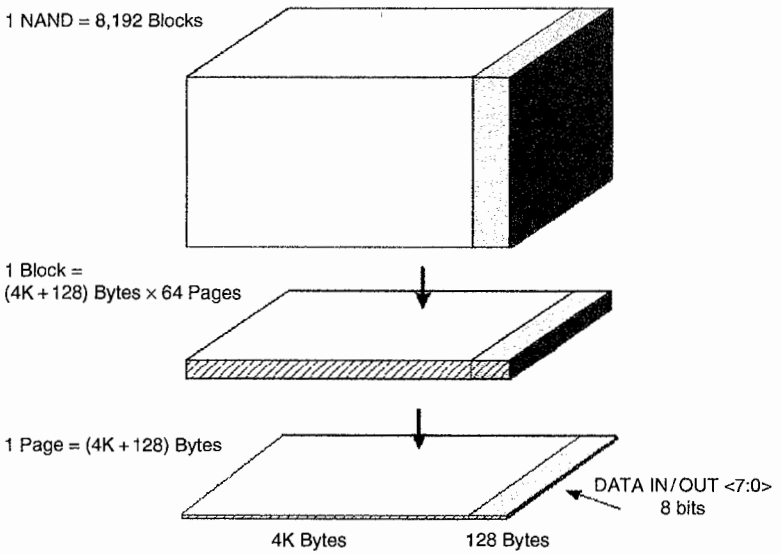


Fig. 2.7. NAND memory logic organization

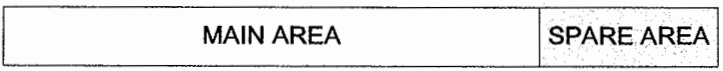


Fig. 2.8. NAND page structure

Petitioner Micron Ex-1021, 0037

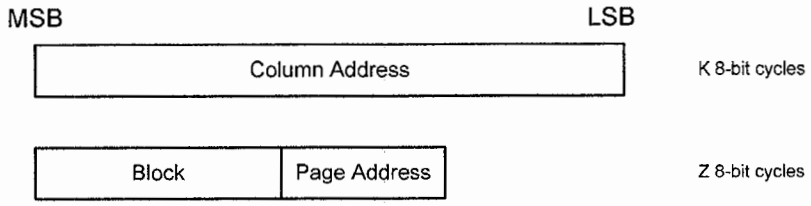


Fig. 2.9. Address structure

2.2.4 Pinout

NAND devices communicate with the external user by means of pins (Fig. 2.10). These pins are monitored by the Command Interface (Chap. 6) which has the task to understand the functionality required to the memory in that moment.

Pins shown in the Fig. 2.10 are listed below.

- CE#: it is the Chip Enable signal. This input signal is "1" when the device is in stand-by mode, otherwise it is always "0".
- R/B#: it is the Ready/Busy signal. This output signal is used to indicate the target status. When low, the target has an operation in progress.
- RE#: it is the Read Enable signal. This input signal is used to enable serial data output.
- CLE: it is the Command Latch Enable. This input is used by the host to indicate that the bus cycle is used to input the command.
- ALE: it is the Address Latch Enable. This input is used by the host to indicate that the bus cycle is used to input the addresses.
- WE#: it is the Write Enable. This input signal controls the latching of input data. Data, command and address are latched on the rising edge of WE#.
- WP#: it is the Write Protect. This input signal is used to disable Flash array program and erase operations.
- DQ<7:0> : these input/output signals represent the data bus.

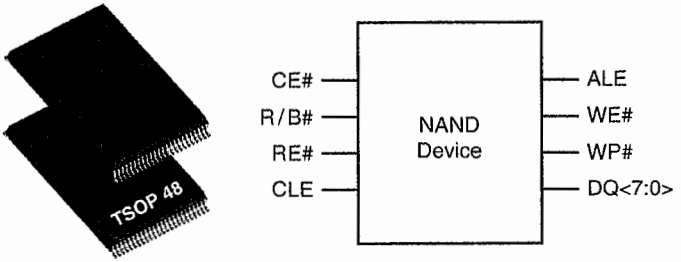


Fig. 2.10. TSOP package (left) and related pinout (right)

Petitioner Micron Ex-1021, 0038

2.3 Command set

2.3.1 Read operation

The read function reads the data stored at a specified address. In order to accomplish this goal, NAND device must recognize when a read operation is issued and the related addresses. After a busy time needed to execute the read algorithm, NAND device outputs the data sequence. Based on the device pin signals, the NAND *Command Interface* (CI, Chap. 6) is able to understand when a command is issued, when an address is issued or when it must perform data out.

Figure 2.11 shows a command cycle ("Cmd"). CI recognizes a "Cmd" cycle if CLE is high. In that case, the 8-bit value on DQs represents the command code.

Figure 2.12 shows address cycles. Generally, all the operations need the addresses where they have to act. The address length depends on the operation and on the capacity of the NAND; anyway, N cycles are needed to input column addresses and M cycles are needed to input row addresses. CI recognized an address cycle if ALE is high. In the meanwhile, all other input signals are low and the DQs value is the address.

The last command phase used by the read operation is the data out, shown in Fig. 2.13. Data out is performed by toggling signal RE#: at every cycle a new data is available on DQs.

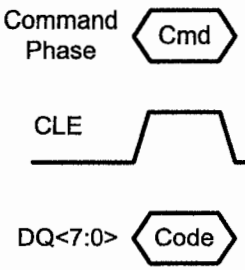


Fig. 2.11. Command cycle ("Cmd"): CLE is high, all other input signals are low

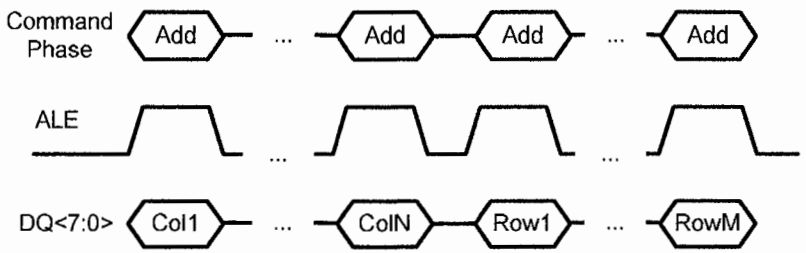


Fig. 2.12. Address cycle: ALE is high, all other inputs signals are low

Petitioner Micron Ex-1021, 0039

These basic cycles are used by the NAND to decode and perform every operation. Figure 2.14 shows the command sequence for a read operation. The first cycle is used to issue the read command "RD" (e.g. 00h). After the command cycle a number of cycles is used to provide the addresses. As described in Sect. 2.2 the column addresses are given first, followed by the row addresses. All the pins (ALE, CLE, RE#) not present in the figure must be driven as described above. Code "RDC" (Read Command Confirm, e.g. 30h) is used to confirm the read command. Finally, the device goes busy and the read operation starts. When NAND returns ready, the data output cycles start.

The above described read command outputs the entire Flash page, regardless the number of bytes we want to read. In some cases, a small move of data is required or we may want to read randomly inside a page. Command Change Read Column, also known as Random Data Output, is able to change the column address we are reading.

Figure 2.15 shows the Change Read Column sequence. After the usual read command is executed, it is possible to change the column address during data out. A command cycle "CRC" (Change Read Column, e.g. 05h) is issued, followed by the addresses of the locations we want to output from. Only fewer cycles are required with respect to the usual read command, since only the column addresses are needed. A confirm command cycle "CRCC" (Change Read Column Confirm, e.g. E0h) is used to enable the data out. It is worth noting that no additional busy time is necessary, because data are already stored in the page buffers.

Generally, the busy time in a read operation lasts for about 25–30 µs. One way to improve read throughput is the Read Cache Command (when available). With this command, it is possible to download data from the Flash memory, while page buffers are reading another page from the Flash array.

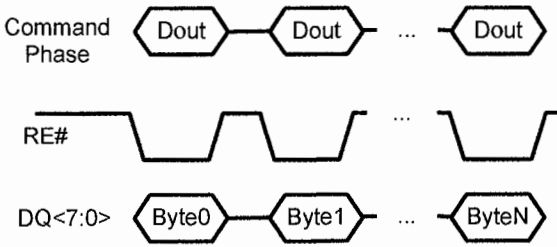


Fig. 2.13. "Dout" cycle: RE# is low, all other inputs signals are low

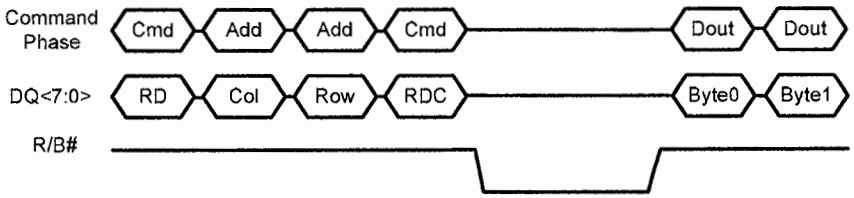


Fig. 2.14. Read command sequence

Petitioner Micron Ex-1021, 0040

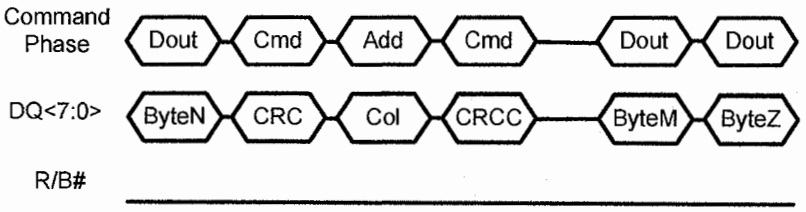


Fig. 2.15. Change Read Column sequence

Sequential Read Cache Command sequence is shown in Fig. 2.16. A Read Command must be issued before Read Cache Command. After the device returns ready, the command code "RC" (Read Cache, e.g. 31h) is used to initiate data download from the matrix to page buffers. RB# goes low for a while and then N Dout cycles are used to output first page. Since no other addresses are input, the next sequential page is automatically read inside the NAND. When we don't need to read other pages, the last page is copied into the page buffers by using command "RCE" (Read Cache End, e.g. 3Fh).

Random Cache Read sequence is shown in Fig. 2.17: with this command it is possible to select the address of the page we want to cache.

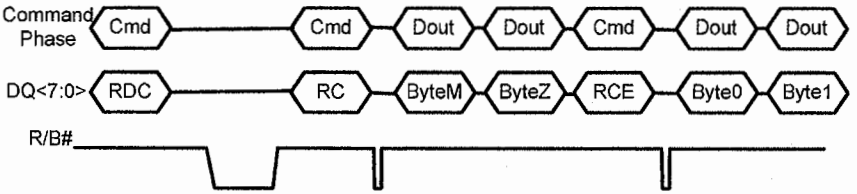


Fig. 2.16. Sequential Cache Read command

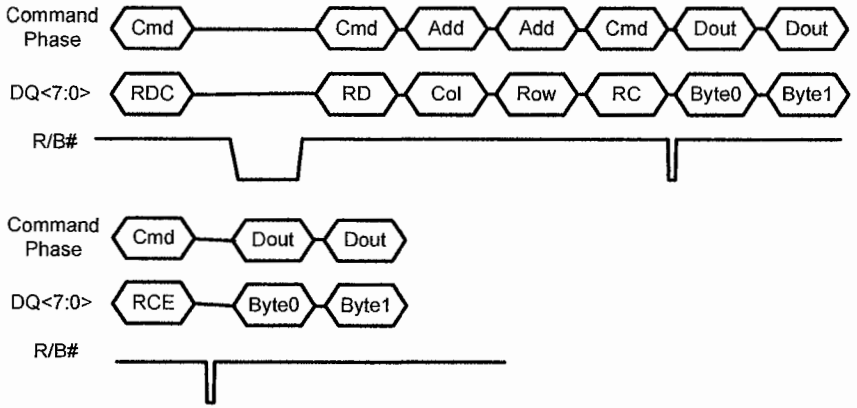


Fig. 2.17. Random Cache Read command

Petitioner Micron Ex-1021, 0041

On multi-plane device, it is possible to issue a read command on multiple planes simultaneously. Figure 2.18 shows the command sequence for Multi-plane Read.

After the standard Read Command cycle "RD", the addresses of the page we want to read on plane 0 are issued. The command code "MR" (Multi-plane read, e.g. 32h) is used in the next command cycle so that the device is ready to receive the addresses belonging to plane 1. Once the new addresses and the Read Command Confirm Code "RDC" are given, the device goes busy to perform the read algorithm on both planes simultaneously. When the device returns ready, the command cycle CC (Choose Column, e.g. 06h) is used to select the address of the page we want to output, followed by a number of address cycles. The command code "CCC" (Choose Column Confirm, e.g. E0h) is a command confirm. Finally the Dout cycles are used to output the read data.

Since both the Read Cache command and the Multi-plane read have been introduced to increase performances, it is interesting to compare them. Figure 2.19 shows a comparison among Read, Cache Read and Multi-plane Read.

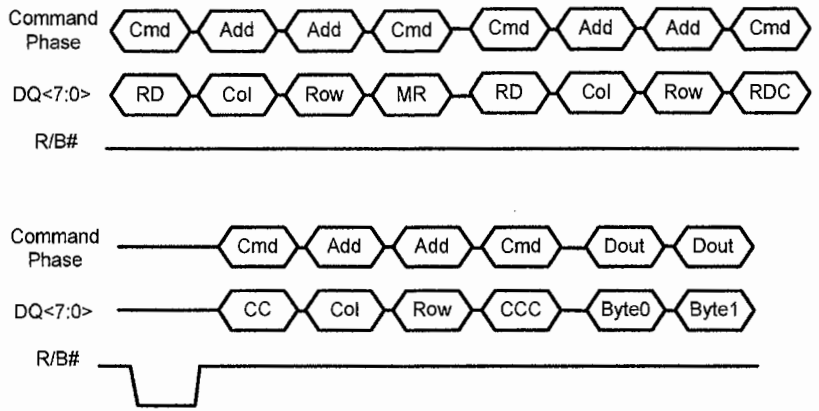


Fig. 2.18. Multi-plane read command

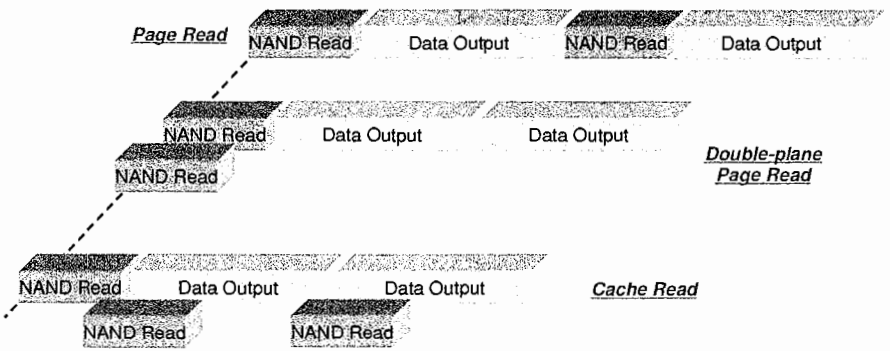


Fig. 2.19. Performance comparison among Read, Double-plane Read and Cache Read

Petitioner Micron Ex-1021, 0042

Given T_{ALGO} the time needed to read from the NAND array, and T_{OUT} the time needed to download the page, the total time to perform the read of two pages with a standard read is $T = 2\ T_{ALGO} + 2\ T_{OUT}$. If a multi-plane command is used, T_{ALGO} runs simultaneously on both planes and $T = T_{ALGO} + 2\ T_{OUT}$. Evaluating T in the Cache Read case is more difficult, because we have to consider the ratio between T_{ALGO} and T_{OUT}. If T_{OUT} is longer than T_{ALGO}, then $T = T_{ALGO} + 2\ T_{OUT}$. It follows that the performances of Cache Read and Double Plane Read are the same. On the contrary, if T_{ALGO} is longer than T_{OUT} (Chap. 16), it won't be possible anymore to mask T_{ALGO} with a single page data out (Fig. 2.20). In this case, Double-plane Read performs better than Cache Read.

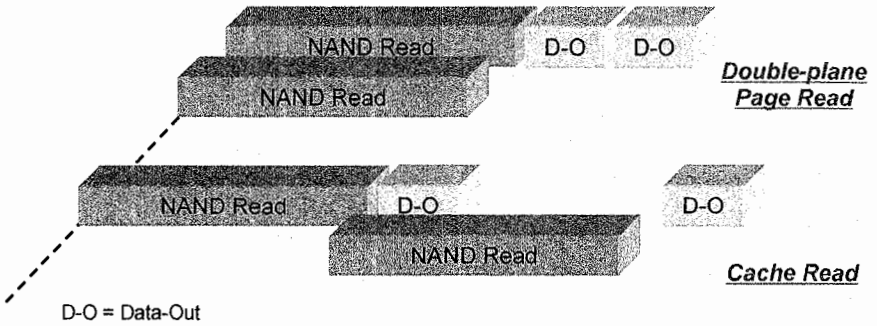


Fig. 2.20. Performance comparison among Read, Double-plane Read and Cache Read with a NAND array read time longer than page data output

2.3.2 Program operation

Purpose of program operation is to write a data sequence at a specified address. The basic cycles are those already described for read operation, such as Command cycle and Address cycle. The only added cycle is the "Din" cycle, represented in Fig. 2.21. Data in is performed by toggling signal WE#: at every cycle a new byte shall be made available on DQs.

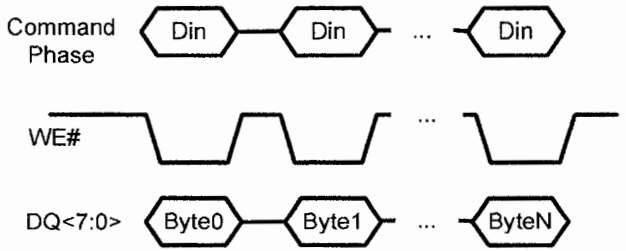


Fig. 2.21. "Din" cycle: WE# is low, all other inputs signals are low

Petitioner Micron Ex-1021, 0043

Program sequence is shown in Fig. 2.22. A command cycle to input "PM" code (Program, e.g. 80h) is followed by a number of address cycles to input the addresses where we want to write. Once the location is set, N "Din" cycles are used to input data into the page buffers. Finally a "PMC" (Program Confirm, e.g. 10h) command is issued to start the algorithm.

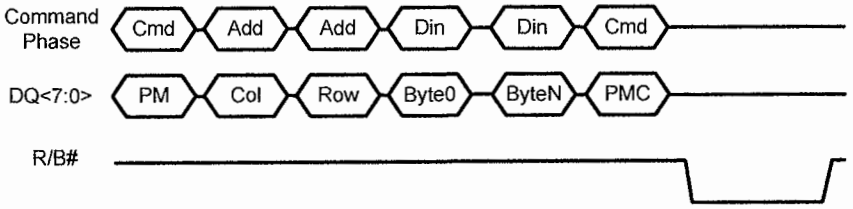


Fig. 2.22. Program command

As already described for read operation, also in the program case there could be the need to move a small amount of data. Change Write Column is used to change the column address where we want to load the data.

Program busy time can be very long: 150–200 µs. Program cache command or double-plane program are used to increase write throughput. Figure 2.23 shows the sequence for Cache Program and Double Plane Program.

The first cycles ("PM" cycle, address cycles and "Din" cycles) are the same as in the standard program. Instead of "PMC" a "C/M" command cycle is issued. "C/M" can be the Cache Program Code (e.g. 15h) or a Double Plane Command (e.g. 11h). Once another "PM" command is given, followed by the new addresses and the "PMC" command, the device goes busy and the program algorithm is performed simultaneously on both pages. It is worth noting that the above described Double plane program is generally known as Concurrent double-plane Program, because the program algorithm works simultaneously on both planes.

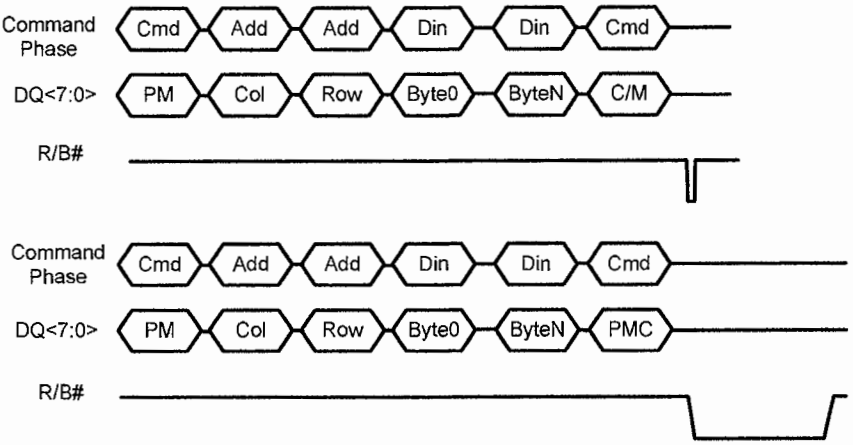


Fig. 2.23. Cache Program and Double-Plane Program commands

Petitioner Micron Ex-1021, 0044

Overlapped Double-Plane Program is also available, where the program on the first plane starts as soon as data are loaded in the page buffers. In this case the NAND architecture must be able to perform the algorithm in an independent way on both planes.

The comparison between the above mentioned program commands is shown in Fig. 2.24.

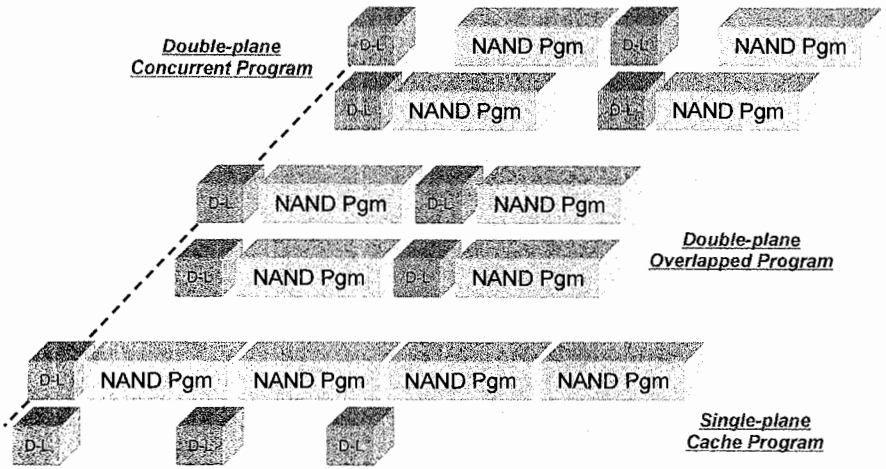


Fig. 2.24. Performance comparison among Cache Program, Overlapped Double Plane program and Concurrent Double Plane Program

2.3.3 Erase operation

The Erase operation is used to delete data from the Flash array. As already described (Sect. 2.2.3), the smallest erasable unit for the NAND memory is the block. Figure 2.25 shows the Erase Command sequence.

The Erase command is very simple: a "ER" code is issued (Erase Command, e.g. 60h), followed by the block address and the "ERC" code (Erase Command Confirm, e.g. D0h). After that, the device goes busy to perform the algorithm.

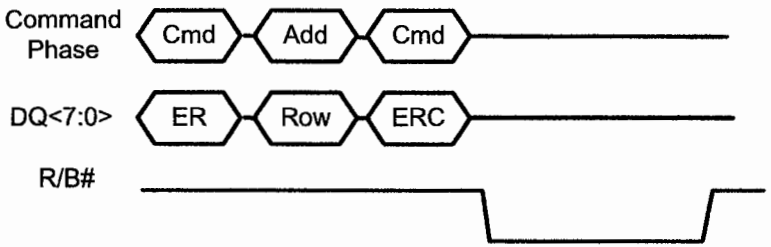


Fig. 2.25. Erase command

Petitioner Micron Ex-1021, 0045

Since erase is the longest operation (about 1 ms), the Double-Plane Erase command has been introduced to erase two blocks at the same time. Figure 2.26 shows the command sequence for the Double-Plane Erase.

The standard erase cycles ("ER" command and row address cycles) are followed by a "MER" command (Multi-plane erase, e.g. D1h). Once both the plane 1 addresses and the "ERC" code are given, the device goes busy, erasing both blocks simultaneously.

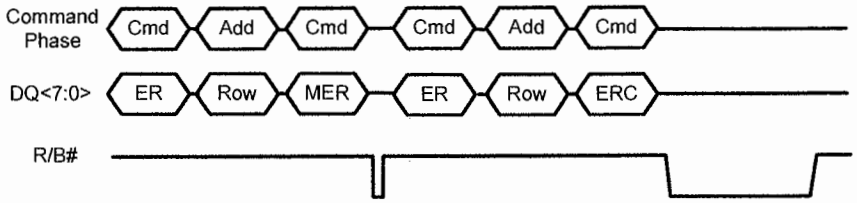


Fig. 2.26. Double-Plane Erase command

2.3.4 Synchronous operations

NAND read throughput is determined by array access time and data transfer across the DQ bus. The data transfer is limited to 40 MB/s by the asynchronous interface. As technology shrinks, page size increases and data transfer takes longer; as a consequence, NAND read throughput decreases, totally unbalancing the ratio between array access time and data transfer on the DQ bus. DDR interface (Chap. 7) has been introduced to balance this ratio.

Nowadays two possible solutions are available on the market. The first one, *Source Synchronous Interface* (SSI), is driven by the ONFI (Open NAND Flash Interface) organization established in 2006 with the purpose of standardizing the NAND interface. Other NAND vendors use the Toggle-Mode interface.

Figure 2.27 shows the NAND pinout for SSI. Compared to the *Asynchronous Interface* (ASI, Sect. 2.2), there are three main differences:

- RE# becomes W/R# which is the Write/Read direction pin.
- WE# becomes CLK which is the clock signal.
- DQS is an additional pin acting as the data strobe, i.e. it indicates the data valid window.

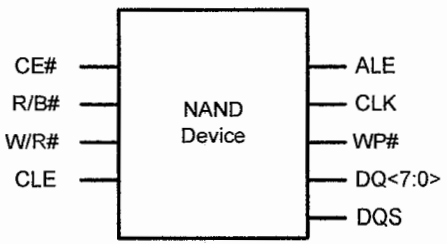


Fig. 2.27. Pinout of a NAND Flash supporting Source Synchronous Interface

Petitioner Micron Ex-1021, 0046

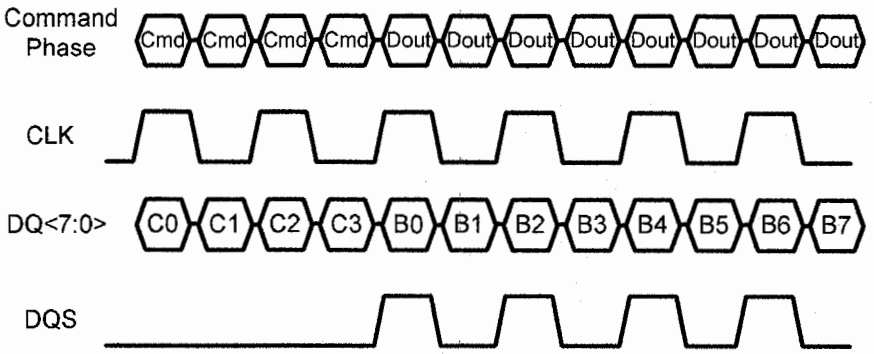


Fig. 2.28. Source Synchronous Interface DDR sequence

Hence, the clock is used to indicate where command and addresses should be latched, while a data strobe signal is used to indicate where data should be latched. DQS is a bi-directional bus and is driven with the same frequency as the clock.

Obviously, the basic command cycles described in the previous sections must be modified according to the new interface.

Figure 2.28 shows a "Cmd" sequence, followed by "Dout" cycles for SSI.

Toggle-Mode DDR interface uses the pinout shown in Fig. 2.29.

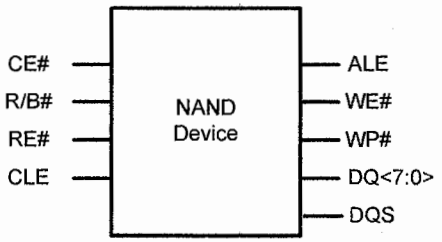


Fig. 2.29. Pinout of a NAND Flash supporting Toggle-Mode Interface

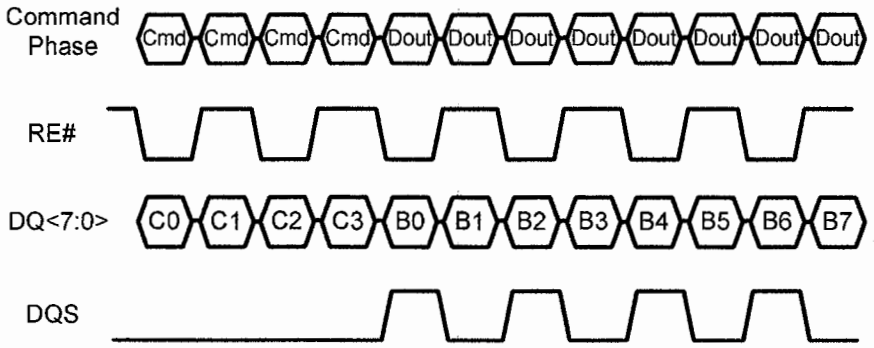


Fig. 2.30. Toggle-Mode DDR sequence

Petitioner Micron Ex-1021, 0047

It can be noted that only the DQS pin has been added to the standard ASI. In this case, higher speeds are achieved increasing the toggling frequency of RE#.

Figure 2.30 shows a "Cmd" sequence, followed by "Dout" cycles for Toggle-Mode interface.

2.4 NAND-based systems

Flash cards, USB sticks and *Solid State Disks* (SSDs) are definitely the most known examples of electronic systems based on NAND Flash.

Several types of memory cards are available on the market [3–5], with different user interfaces and form factors, depending on the needs of the target application: e.g. mobile phones need very small-sized removable media like µSD. On the other hand, digital cameras can accept larger sizes as memory capacity is more important (CF, SD, MMC). Figure 2.31 shows different types of Flash cards.

The interfaces of the Flash cards (including USB sticks) support several protocols: parallel or serial, synchronous or asynchronous. Moreover, the Flash cards support hot insertion and hot extraction procedures, which require the ability to manage sudden loss of power supply while guaranteeing the validity of stored data.

For the larger form factors, the card is a complete, small system where every component is soldered on a PCB and is independently packaged. For example, the NAND Flash memories are usually available in TSOP packages. It is also possible to include some additional components: for instance, an external DC-DC converter can be added in order to derive an internal power supply (CompactFlash cards can work at either 5 or 3.3 V), or a quartz can be used for a better clock precision. Usually, reasonable filter capacitors are inserted for stabilizing the power supply. Same considerations apply to SSDs; the main difference with Flash cards is the system capacity as shown in Fig. 2.32 where multiple NANDs are organized in different channels in order to improve performances.

For small form factors like µSD, the size of the card is comparable to that of the NAND die. Therefore, the memory chip is mounted as bare die on a small substrate. Moreover, the die thickness has to be reduced in order to comply with the thickness of µSD, considering that several dies are stacked, i.e. mounted one on top of each other.

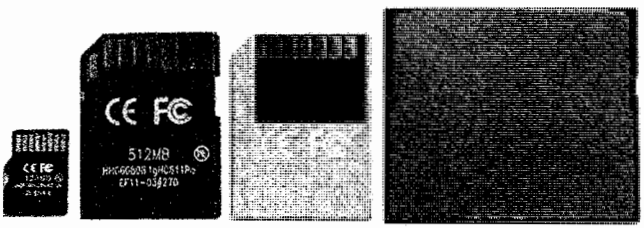


Fig. 2.31. Popular Flash card form factors

Petitioner Micron Ex-1021, 0048

All these issues cause a severe limitation to the maximum capacity of the card; in addition external components, like voltage regulators and quartz, cannot be used. In other words, the memory controller of the card has to implement all the required functions.

The assembly stress for small form factors is quite high and, therefore, system testing is at the end of the production. Hence, production cost is higher (Chap. 15).

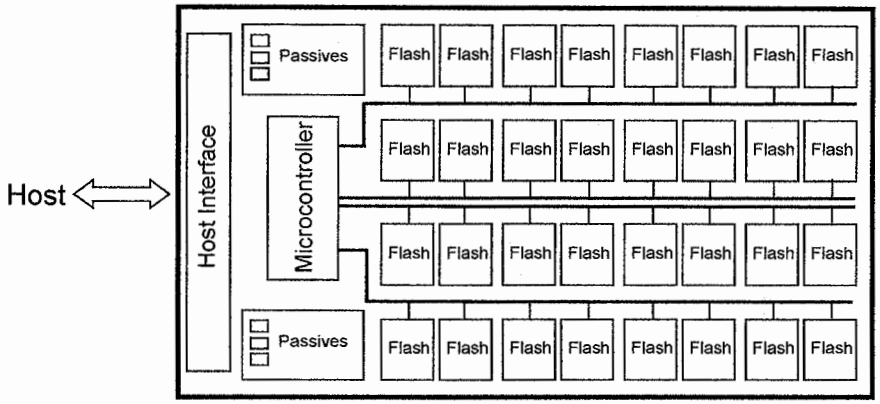


Fig. 2.32. Block diagram of a SSD

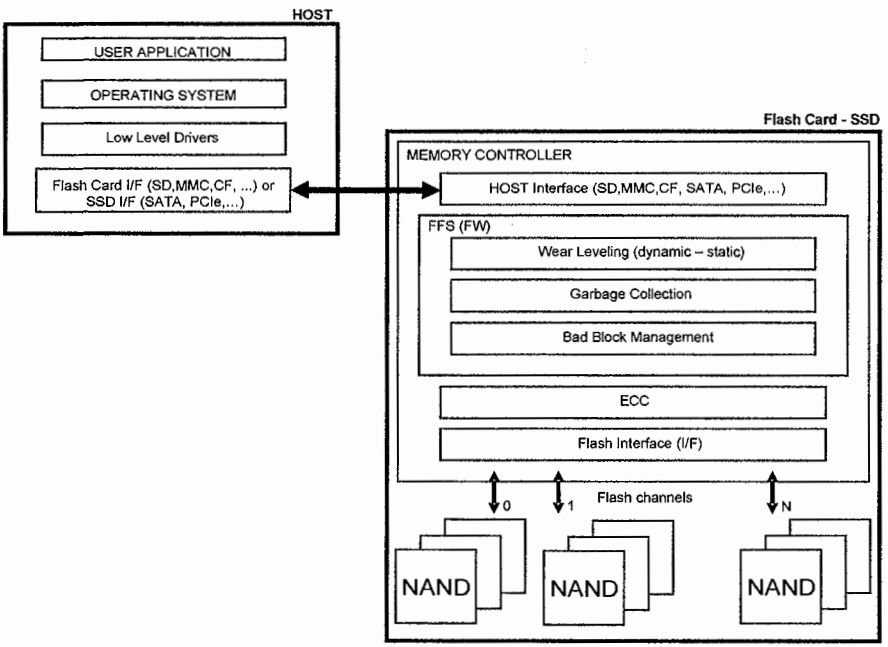


Fig. 2.33. Functional representation of a Flash card (or SSD)

Petitioner Micron Ex-1021, 0049

For a more detailed description of Flash cards, please, refer to Chap. 17. SSDs are described in Chap. 18.

Figure 2.33 shows a functional representation of a memory card or SSD: two types of components can be identified: the memory controller and the Flash memory components. Actual implementation may vary, but the functions described in the next sections are always present.

2.4.1 Memory controller

The aim of the memory controller is twofold:

1. To provide the most suitable interface and protocol towards both the host and the Flash memories
2. To efficiently handle data, maximizing transfer speed, data integrity and information retention

In order to carry out such tasks, an application specific device is designed, embedding a standard processor – usually 8–16 bits – together with dedicated hardware to handle timing-critical tasks.

For the sake of discussion, the memory controller can be divided into four parts, which are implemented either in hardware or in firmware. Proceeding from the host to the Flash, the first part is the host interface, which implements the required industry-standard protocol (MMC, SD, CF, etc.), thus ensuring both logical and electrical interoperability between Flash cards and hosts. This block is a mix of hardware – buffers, drivers, etc. – and firmware – command decoding performed by the embedded processor – which decodes the command sequence invoked by the host and handles the data flow to/from the Flash memories.

The second part is the Flash File System (FFS) [6]: that is, the file system which enables the use of Flash cards, SSDs and USB sticks like magnetic disks. For instance, sequential memory access on a multitude of sub-sectors which constitute a file is organized by linked lists (stored on the Flash card itself) which are used by the host to build the File Allocation Table (FAT).

The FFS is usually implemented in form of firmware inside the controller, each sub-layer performing a specific function. The main functions are: *Wear leveling Management*, *Garbage Collection* and *Bad Block Management*. For all these functions, tables are widely used in order to map sectors and pages from logical to physical (Flash Translation Layer or FTL) [7, 8], as shown in Fig. 2.34.

The upper block row is the logical view of the memory, while the lower row is the physical one. From the host perspective, data are transparently written and overwritten inside a given logical sector: due to Flash limitations, overwrite on the same page is not possible, therefore a new page (sector) must be allocated in the physical block and the previous one is marked as invalid. It is clear that, at some point in time, the current physical block becomes full and therefore a second one (Buffer) is assigned to the same logical block.

The required translation tables are always stored on the memory card itself, thus reducing the overall card capacity.

Petitioner Micron Ex-1021, 0050

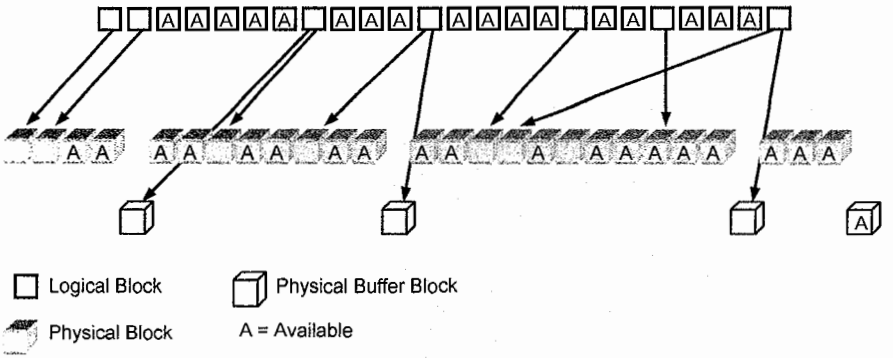


Fig. 2.34. Logical to physical block management

Wear leveling

Usually, not all the information stored within the same memory location change with the same frequency: some data are often updated while others remain always the same for a very long time – in the extreme case, for the whole life of the device. It's clear that the blocks containing frequently-updated information are stressed with a large number of write/erase cycles, while the blocks containing information updated very rarely are much less stressed.

In order to mitigate disturbs, it is important to keep the aging of each page/block as minimum and as uniform as possible: that is, the number of both read and program cycles applied to each page must be monitored. Furthermore, the maximum number of allowed program/erase cycles for a block (i.e. its endurance) should be considered: in case SLC NAND memories are used, this number is in the order of 100 k cycles, which is reduced to 10 k when MLC NAND memories are used.

Wear Leveling techniques rely on the concept of logical to physical translation: that is, each time the host application requires updates to the same (logical) sector, the memory controller dynamically maps the sector onto a different (physical) sector, keeping track of the mapping either in a specific table or with pointers. The out-of-date copy of the sector is tagged as both invalid and eligible for erase. In this way, all the physical sectors are evenly used, thus keeping the aging under a reasonable value.

Two kinds of approaches are possible: Dynamic Wear Leveling is normally used to follow up a user's request of update for a sector; Static Wear Leveling can also be implemented, where every sector, even the least modified, is eligible for re-mapping as soon as its aging deviates from the average value.

Garbage collection

Both wear leveling techniques rely on the availability of free sectors that can be filled up with the updates: as soon as the number of free sectors falls below a given threshold, sectors are "compacted" and multiple, obsolete copies are deleted.

Petitioner Micron Ex-1021, 0051

This operation is performed by the Garbage Collection module, which selects the blocks containing the invalid sectors, copies the latest valid copy into free sectors and erases such blocks (Fig. 2.35).

In order to minimize the impact on performance, garbage collection can be performed in background. The equilibrium generated by the wear leveling distributes wear out stress over the array rather than on single hot spots. Hence, the bigger the memory density, the lower the wear out per cell is.

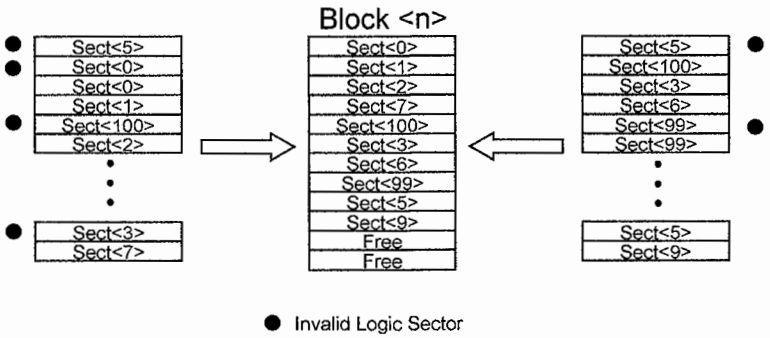


Fig. 2.35. Garbage collection

Bad block management

No matter how smart the Wear Leveling algorithm is, an intrinsic limitation of NAND Flash memories is represented by the presence of so-called *Bad Blocks* (BB), i.e. blocks which contain one or more locations whose reliability is not guaranteed.

The *Bad Block Management* (BBM) module creates and maintains a map of bad blocks, as shown in Fig. 2.36: this map is created during factory initialization of the memory card, thus containing the list of the bad blocks already present during the factory testing of the NAND Flash memory modules. Then it is updated during device lifetime whenever a block becomes bad.

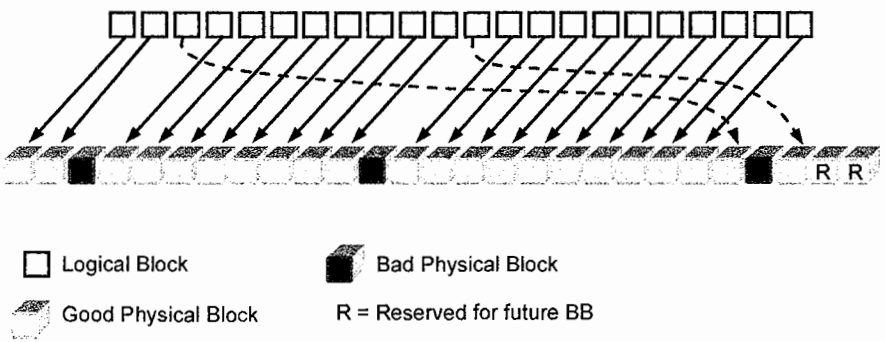


Fig. 2.36. Bad Block Management (BBM)

Petitioner Micron Ex-1021, 0052

ECC

This task is typically executed by a specific hardware inside the memory controller. Examples of memories with embedded ECC are also reported [9–11]. Most popular ECC codes, correcting more than one error, are Reed–Solomon and BCH [12]. While the encoding takes few controller cycles of latency, the decoding phase can take a large number of cycles and visibly reduce read performance as well as the memory response time at random access.

There are different reasons why the read operation may fail (with a certain probability):

- Noise (e.g. at the power rails)
- V_{TH} disturbances (read/write of neighbor cells)
- Retention (leakage problems)

The allowed probability of failed reads after correction is dependent on the use case of the application. Price sensitive consumer application, with a relative low number of read accesses during the product life time, can tolerate a higher probability of read failures as compared to high-end applications with a high number of memory accesses. The most demanding applications are cache modules for processors.

The reliability that a memory can offer is its intrinsic error probability. This probability could not be the one that the user wishes. Through ECC it is possible to fill the discrepancy between the desired error probability and the error probability offered by the memory (Chap.14).

The object of the theory of error correction codes is the addition of redundant terms to the message, such that, on reading, it is possible to detect the errors and to recover the message that has most probably been written.

Methods of error correction are applied for purpose of data restoration at read access. Block code error correction is applied on sub-sectors of data. Depending on the used error correcting schemes, different amount of redundant bits called parity bits are needed.

Between the length n of the code words, the number k of information bits and the number t of correctable errors, a relationship known as Hamming inequality exists, from which it is possible to compute the minimum number of parity bits:

$$\sum_{i=0}^{t} \binom{n}{i} \leq 2^{n-k} \tag{2.3}$$

It is not always possible to reach this minimum number: the number of parity bits for a good code must be as near as possible to this number. On the other hand, the bigger the size of the sub-sector is, the lower the relative amount of spare area (for parity bits) is. Hence, there is an impact in Flash die size.

BCH and Reed–Solomon codes have a very similar structure, but BCH codes require less parity bits and this is one of the reasons why they were preferred for an ECC embedded in the NAND memory [11].

Petitioner Micron Ex-1021, 0053

2.4.2 Multi-die systems

A typical memory system is composed by several NAND memories. Typically, an 8-bit bus, usually called channel, is used to connect different memories to the controller (Fig. 2.32). It is important to underline that multiple Flash memories in a system are both a means for increasing storage density and read/write performance [13].

Operations on a channel can be interleaved which means that another memory access can be launched on an idle memory while the first one is still busy (e.g. writing or erasing). For instance, a sequence of multiple write accesses can be directed to a channel, addressing different NANDs, as shown in Fig. 2.37: in this way, the channel utilization is maximized by pipelining data load, while the program operation takes place without requiring channel occupation. A system typically has two to eight channels operating in parallel (or even more).

As shown in Fig. 2.38, using multiple memory components is an efficient way to improve data throughput while having the same page programming time.

The memory controller is responsible for scheduling the distributed accesses at the memory channels. The controller uses dedicated engines for the low level communication protocol with the Flash.

Moreover it is clear that the data load phase is not negligible compared to the program operation (the same comment is valid for data output): therefore increasing I/O interface speed is another smart way to improve general performance: high-speed interfaces, like DDR, have already been reported [14] and they are discussed in more details in Chap. 7. Figure 2.39 shows the impact of DDR frequency on program throughput. As the speed increases, more NAND can be operated in parallel before saturating the channel. For instance, assuming a target of 30 MB/s, 2 NANDs are needed with a minimum DDR frequency of about 50 MHz. Given a page program time of 200 µs, at 50 MHz four NANDs can operate in interleaved mode, doubling the write throughput. Of course, power consumption has then to be considered.

There are also hybrid architectures which combine different types of memory. Most common is usage of DRAM as memory cache. During write access the cache is used for storing data before transfer to the Flash. The benefit is that data updating, e.g. in tables, is faster and does not wear out the Flash.

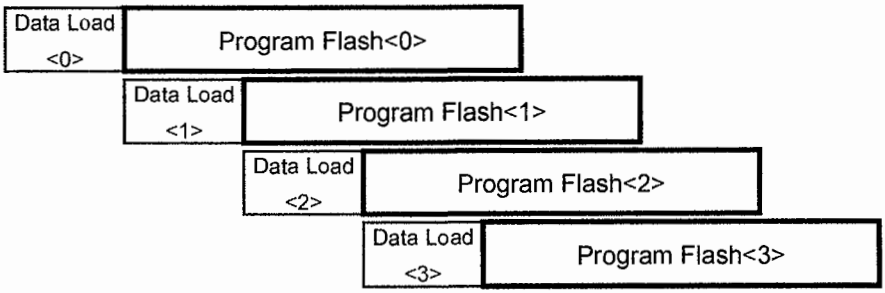


Fig. 2.37. Interleaved operations on one Flash channel

Petitioner Micron Ex-1021, 0054

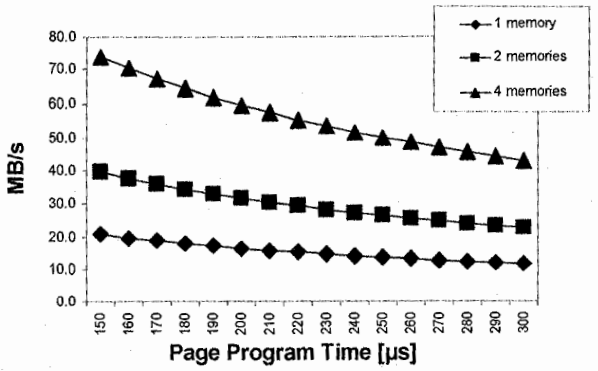


Fig. 2.38. Program throughput with an interleaved architecture as a function of the NAND page program time

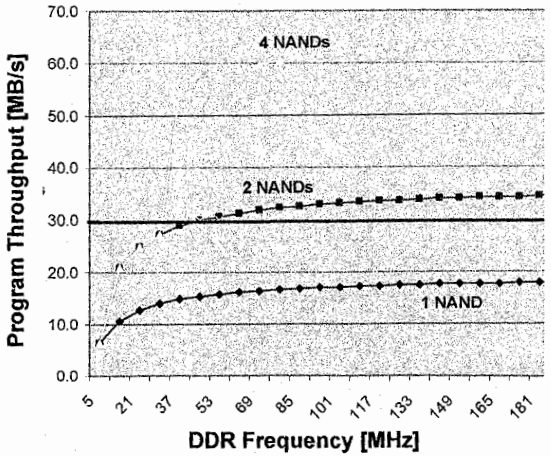


Fig. 2.39. Program throughput with an interleaved architecture as a function of the channel DDR frequency. 4 kB page program time is 200 µs

Another architecture uses a companion NOR Flash for purpose of "in-place execution" of software without pre-fetch latency. For hybrid solutions, a multi-die approach, where different memories are packaged in the same chip, is a possibility to reduce both area and power consumption.

2.4.3 Die stacking

Reduced form factor has been one of the main drivers for the success of the memory cards; on the other hand, capacity requirement has grown dramatically to the extent that standard packaging (and design) techniques are no longer able to

Petitioner Micron Ex-1021, 0055

sustain the pace. In order to solve this issue, two approaches are possible: advanced die stacking and 3D technologies.

The standard way to increase capacity is to implement a multi-chip solution, where several dies are stacked together. The advantage of this approach is that it can be applied to existing bare die, as shown in Fig. 2.40: die are separated by means of a so-called interposer, so that there is enough space for the bonding wires to be connected to the pads.

On the other hand, the use of the interposer has the immediate drawback of increasing the height of the multi-chip, and height is one of the most relevant limiting factors for memory cards.

One way to overcome this issue is to exploit both sides of the PCB, as shown in Fig. 2.41: in this way, the PCB acts as interposer, and components are evenly flipped on the two sides of the PCB. Height is reduced, but there is an additional constraint on design: in fact, since the lower die is flipped, its pads are no longer matching those of the upper die. The only way to have corresponding pads facing one another is to design the pad section in such a way that pad to signal correspondence can be scrambled: that is, when a die is used as the bottom one, it is configured with mirrored-pads. Such a solution is achievable, but chip design is more complex (signals must be multiplexed in order to perform the scramble) and chip area is increased, since it might be necessary to have additional pads to ensure symmetry when flipping.

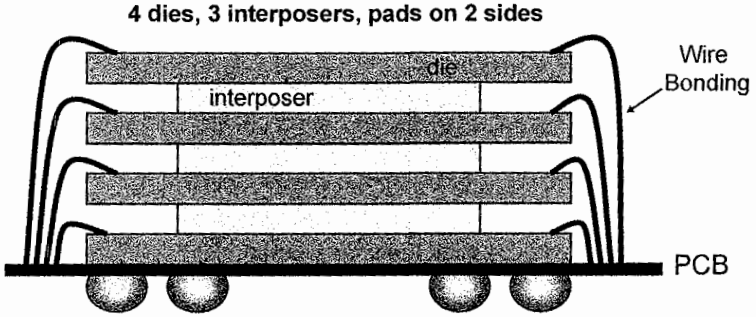


Fig. 2.40. Standard die stacking

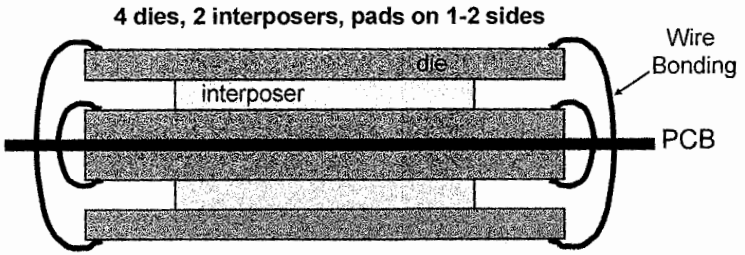


Fig. 2.41. Flipped die stacking

Petitioner Micron Ex-1021, 0056

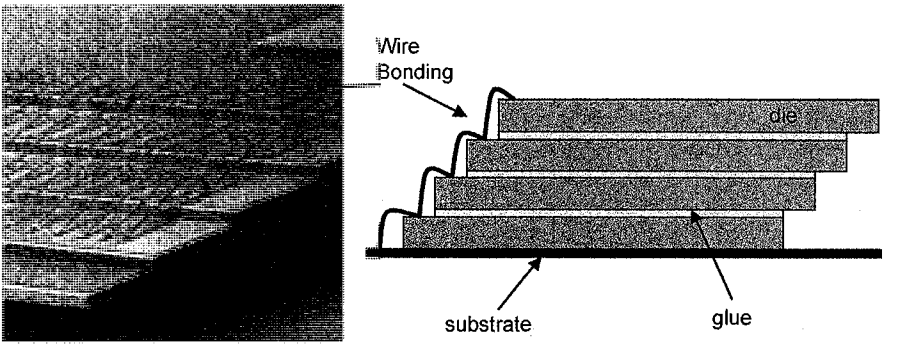


Fig. 2.42. Staircase die stacking: four dies, zero interposers, pads on one side

The real breakthrough is achieved by completely removing the interposer, thus using all the available height for silicon (apart from a minimum overhead due to the die-to-die glue). Figure 2.42 shows an implementation, where a staircase arrangement of the dies is used: any overhead is reduced to the minimum, bonding does not pose any particular issue and chip mechanical reliability is maintained (the disoverlap between dies is small compared to the die length, and therefore the overall stability is not compromised, since the upmost die does not go beyond the overall center of mass).

The drawback is that such a solution has a heavy impact on chip design, since all the pads must be located on the same side of the die. In a traditional memory component, pads are arranged along two sides of the device: circuitry is then evenly located next to the two pad rows and the array occupies the majority of the central area. Figure 2.43 shows the floorplan of a memory device whose pads lie on two opposite sides.

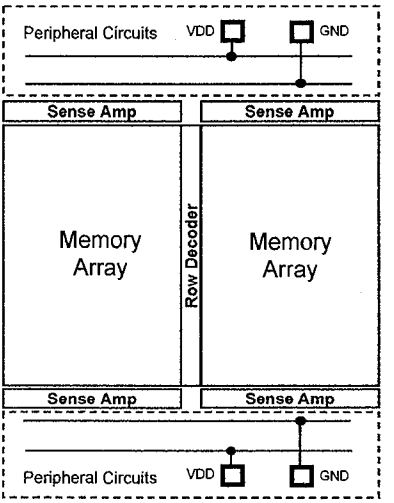


Fig. 2.43. Memory device with pads along opposite sides

Petitioner Micron Ex-1021, 0057

If all pads lie on one side, as shown in Fig. 2.44, chip floorplan is heavily impacted [15]: most of the circuits are moved next to the pads in order to minimize the length of the connections and to optimize circuit placement. But some of the circuits still reside on the opposite side of the die (for instance, part of the decoding logic of the array and part of the page buffers, i.e. the latches where data are stored, either to be written to the memory or when they are read from the memory to be provided to the external world).

Of course, such circuits must be connected to the rest of the chip, both from a functional and from a power supply point of view. Since all pads are on the opposite side, including power supply ones, it is necessary to re-design the power rail distribution inside the chip, making sure that the size and geometry of the rails is designed properly, in order to avoid IR drops issues (i.e. the voltage at the end of the rail is reduced due to the resistive nature of the metal line).

One of the main disadvantages of staircase stacking is the increased size in the direction opposite to the pad row. Of course, this fact limits the number of dies, given a specific package.

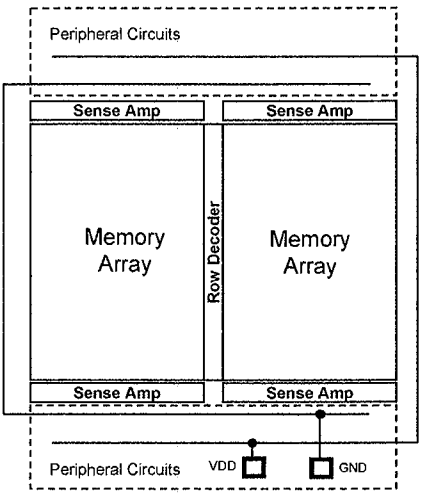


Fig. 2.44. Memory device with pads along one side

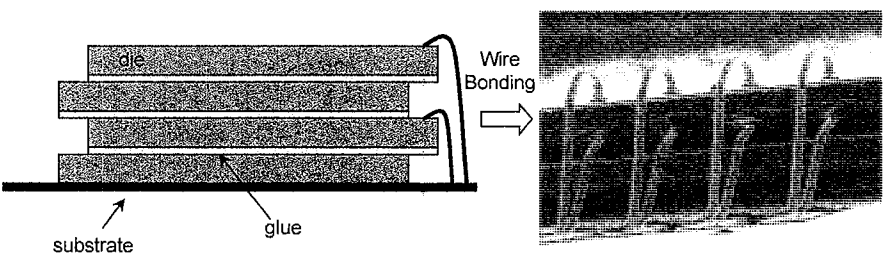


Fig. 2.45. "Snake" die stacking

Petitioner Micron Ex-1021, 0058

Once the design effort has been done, a possible alternative is the "snake" stacking as shown in Fig. 2.45. In this case, thanks to the double side bonding, the overall size can be clearly reduced.

Recently, another stacking option came into the game: *Through Silicon Via* (TSV) [16–19]. With this technology, dies are directly connected without asking for a wire bonding, as depicted in Fig. 2.46. A 3-D view of one TSV connection is shown in Fig. 2.47. One of the main advantages is the reduction of the interconnection length and the associated parasitic *RC*. As a result, data transfer rate can be definitely improved, as well as power consumption.

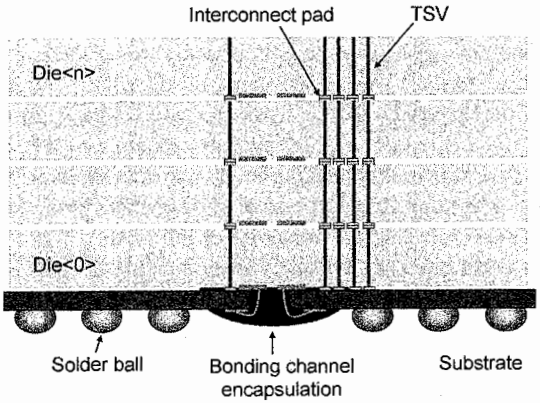


Fig. 2.46. Multiple die stacking with TSV technology

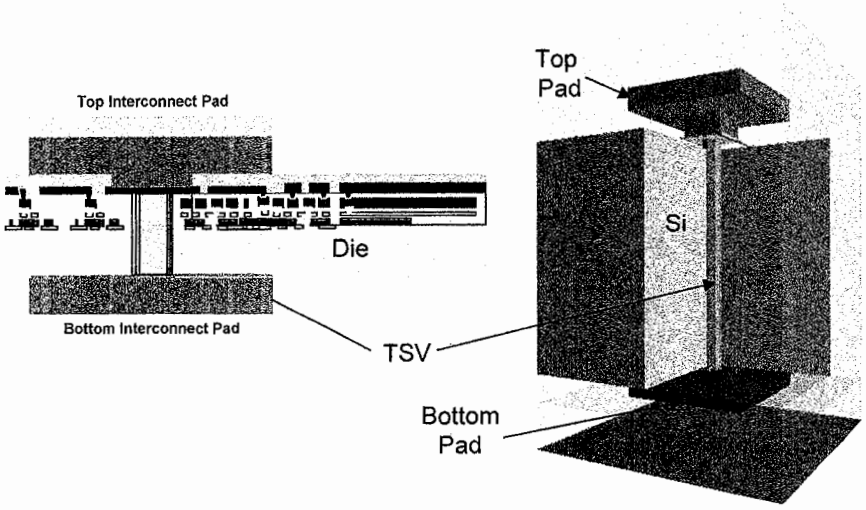


Fig. 2.47. Die with TSV (left) and TSV 3-D view (right)

Petitioner Micron Ex-1021, 0059

In fact, DRAM are the main drivers for TSV technology as, with the standard bonding technology, they cannot stack more than two dies in the same package.

The solutions presented so far exploit advances in stacking techniques, eventually requiring changes in chip design floorplan. Quite recently, advanced design and manufacturing solutions have been presented, where the 3D integration [20] is performed directly at chip level.

2.4.4 3D memories and XLC storage

The 3D concept is simple: instead of stacking several dies, each of which being a fully-functional memory component, it is possible to embed in the same silicon die more than one memory array. In this way, all the control logic, analog circuits and the pads can be shared by the different memory arrays. In order to keep the area to the minimum, the memory arrays are grown one on top of the other, exploiting the most recent breakthroughs in silicon manufacturing technology.

Two different solutions have been recently presented for NAND Flash memories: in one case [21, 22], the topology of the memory array is the usual one, and another array is diffused on top of it, as shown in Fig. 2.48, so that two layers exist. Therefore the NAND strings (i.e. the series of Flash memory cells which is the basic building block of the array) are diffused on the X–Y plane. Around the arrays, all the peripheral circuitry is placed in the first (i.e. lower) layer. The only exception is the wordline (WL) decoder. To avoid Fowler–Nordheim (FN) erasing of the unselected layer, all WLs in that layer must be floating, just like the WLs of unselected blocks in the selected layer. This function is performed by a layer-dedicated WL decoder.

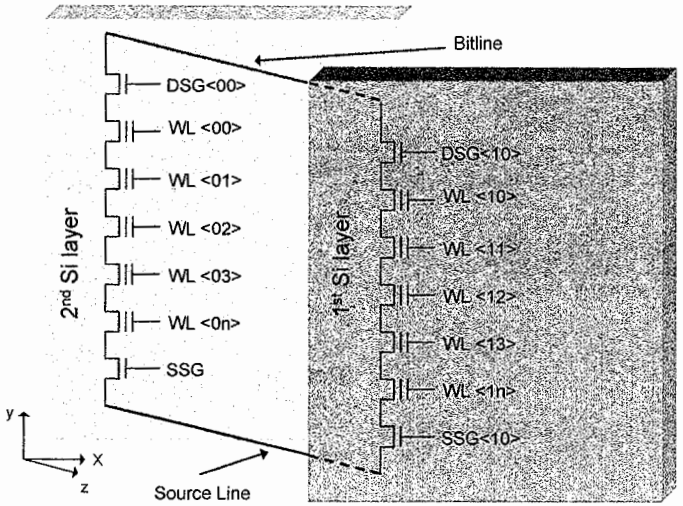


Fig. 2.48. Three-dimensional horizontal memory array

Petitioner Micron Ex-1021, 0060

The second approach [23, 24] is shown in Figs. 2.49 and 5.9: in this case the NAND strings are orthogonal to the die (along the Z direction). NAND string is on the plugs located vertically in the holes punched through whole stack of the gate plates. Each plate acts as control gate except the lowest plate which takes a role of the lower select gate. 3-D charge trap memories are described in Sect. 5.3.

The density of the memory can also be increased acting at cell level: in its simplest form, a non-volatile memory cell stores one bit of information: '1' when the cell is erased and '0' when it is programmed: Sensing techniques for measuring the amount of electrical charge stored in the floating gate are described in Chap. 8. This kind of storage is referred to as Single Level Cell (SLC).

The concept can be extended by having four different charge levels (corresponding to logic values 00, 01, 10, 11) inside the floating gate, thus leading to the so-called Multi Level Cell (MLC) approach, i.e. 2 bit/cell. Chapter 10 is entirely dedicated to MLC NAND devices.

Several devices implementing the 2 bit/cell technology are commercially available, and indeed MLC has become a synonym for 2 bits/cell. Almost all the Flash cards contain MLC devices as they are cheaper than SLC.

The circuitry required to read multiple bits out of a cell is of course more complex than in the case of single bit, but the saving in term of area (and the increase in density) is worth the complexity. The real disadvantage lies in reduced endurance and reliability.

In terms of endurance, as already mentioned previously, a SLC solution can withstand up to 100,000 program/erase cycles for each block, while a MLC solution is usually limited to 10,000. For this reason, wear leveling algorithms must be used, as already outlined in Sect. 2.4.1. In terms of reliability, it is clear that the more levels are used, the more read disturbs can happen, and therefore the ECC capability must be strengthened.

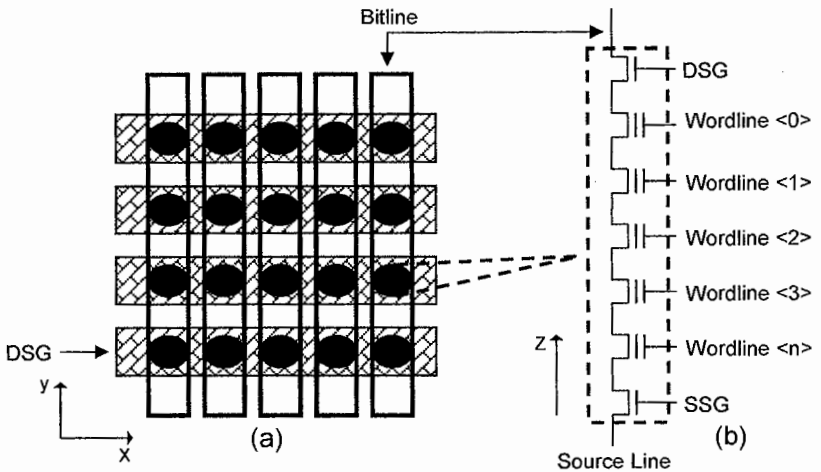


Fig. 2.49. Three dimensional vertical memory array: (a) top down view of 3-D vertical memory array; (b) equivalent circuit of the vertical NAND string

Petitioner Micron Ex-1021, 0061

The concept has been extended recently to 3 bit/cell and 4 bit/cell, where 8 and 16 different charge levels are stored inside the same cell. This storage approach is known as XLC and it is described in Chap. 16.

References

1. G. Campardo, R. Micheloni, D. Novosel, "VLSI-Design of Non-Volatile Memories", Springer-Verlag, 2005.
2. R. H. Fowler and L. Nordheim, "Electron Emission in Intense Electric Fields," Proceedings of the Royal Society of London, Vol. 119, No. 781, May 1928, pp. 173–181.
3. www.mmca.org
4. www.compactflash.org
5. www.sdcard.com
6. A. Kawaguchi, S. Nishioka, and H. Motoda. "A Flash-Memory Based File System", Proceedings of the USENIX Winter Technical Conference, pp. 155–164, 1995.
7. J. Kim, J. M. Kim, S. Noh, S. L. Min, and Y. Cho. "A Space-Efficient Flash Translation Layer for Compactflash Systems," IEEE Transactions on Consumer Electronics, Vol. 48, No. 2, May 2002, pp. 366–375.
8. S.-W. Lee, D.-J. Park, T.-S. Chung, D.-H. Lee, S.-W. Park, and H.-J. Songe. "FAST: A Log-Buffer Based FTL Scheme with Fully Associative Sector Translation", 2005 US-Korea Conference on Science, Technology, and Entrepreneurship, August 2005.
9. T. Tanzawa, T. Tanaka, K. Takekuchi, R. Shirota, S. Aritome, H. Watanabe, G. Hemink, K. Shimizu, S. Sato, Y. Takekuchi, and K. Ohuchi, "A Compact On-Chip ECC for Low Cost Flash Memories," IEEE Journal of Solid-State Circuits, Vol. 32, May 1997, pp. 662–669.
10. G. Campardo, R. Micheloni et al.,"40-mm^2 3-V-only 50-MHz 64-Mb 2-b/cell CHE NOR Flash memory," IEEE Journal of Solid-State Circuits, Vol. 35, No. 11, Nov. 2000, pp. 1655–1667.
11. R. Micheloni et al., "A 4Gb 2b/cell NAND Flash Memory with Embedded 5b BCH ECC for 36MB/s System Read Throughput", IEEE International Solid-State Circuits Conference Dig. Tech. Papers, pp. 142–143, Feb. 2006.
12. R. Micheloni, A. Marelli, R. Ravasio, "Error Correction Codes for Non-Volatile Memories", Springer-Verlag, 2008.
13. C. Park et al., "A High Performance Controller for NAND Flash-based Solid State Disk (NSSD)", IEEE Non-Volatile Semiconductor Memory Workshop NVSMW, pp. 17–20, Feb. 2006.
14. D. Nobunaga et al., "A 50nm 8Gb NAND Flash Memory with 100MB/s Program Throughput and 200MB/s DDR Interface", IEEE International Solid-State Circuits Conference Dig. Tech. Papers, pp. 426–427, Feb. 2008.
15. K. Kanda et al., "A 120mm^2 16Gb 4-MLC NAND Flash Memory with 43nm CMOS Technology", in IEEE International Solid-State Circuits Conference Dig. Tech. Papers, pp. 430–431, Feb. 2008.
16. Chang Gyu Hwang, "New Paradigms in the Silicon Industry, International Electron Device Meeting (IEDM), 2006, pp. 1–8.
17. M. Kawano et al., "A 3D Packaging Technology for a 4Gbit Stacked DRAM with 3Gbps Data Transfer, International Electron Device Meeting (IEDM), 2006, pp. 1–4.

Petitioner Micron Ex-1021, 0062

18. M. Motoyoshi, "Through-Silicon Via (TSV)", Proceedings of the IEEE, Vol. 97, Jan. 2009, pp. 43–48.
19. M. Koyanagi et al., "High-Density Through Silicon Vias for 3-D LSIs", Proceedings of the IEEE, Vol. 97, Jan. 2009, pp. 49–59.
20. G. Campardo, M. Iaculo, F. Tiziani (Eds.), "Memory Mass Storage", Chap. 6, Springer, 2010.
21. S.-M. Jung, J. Jang, W. Cho et al., "Three Dimensionally Stacked NAND Flash Memory Technology Using Stacking Single Crystal Si Layers on ILD and TANOS Structure for Beyond 30nm Node," IEDM Dig. Tech. Papers, pp. 37–40, Dec. 2006.
22. K. T. Park et al.," A 45nm 4Gb 3-Dimensional Double-Stacked Multi-Level NAND Flash Memory with Shared Bitline Structure", IEEE International Solid-State Circuits Conference Dig. Tech. Papers, pp. 510–511, Feb. 2008.
23. H. Tanaka, M. Kido, K. Yahashi et al., "Bit Cost Scalable Technology with Punch and Plug Process for Ultra High Flash Memory," Dig. Symp. VLSI Technology, pp. 14–15, June 2006.
24. R. Katsumata et al., "Pipe-shaped BiCS Flash Memory with 16 Stacked Layers and Multi-Level-Cell Operation for Ultra High Density Storage Devices", 2009 Symposium on VLSI Technology Digest of Technical Papers, pp. 136–137.

Petitioner Micron Ex-1021, 0063

3 Program and erase of NAND memory arrays

Christoph Friederich[1]

The purpose of NAND Flash memories as a non-volatile memory is to store the user data for years without requiring a supply voltage. The state of the art memory cell for this purpose in NAND Flash is the 1T floating gate memory cell, which is based on a MOSFET. In contrast to the 1T1C DRAM cell, which consists of an access transistor and a separate capacitance as charge storage node, the 1T floating gate cell is a MOSFET whose gate dielectric is split with a charge storage node in between. This charge storing node, usually made of poly-silicon, is electrically isolated completely by the surrounding dielectrics. Its stored charges represent the information, and may be altered according to the user data by the program operation.

The following chapter deals with such program and erase operations and their embodiments in the NAND Flash memory array. After presenting the single floating gate memory cell, its charge injection mechanisms and programming methods are reviewed. Then, the organization of memory cells in the NAND array and its consequences on the program inhibit as well as cell disturbs are explained. After that, a brief overview on the erase operation and the erase inhibit follows. The last section of the chapter is about the impact of variations on the memory system, or rather the impact of process, data and charge injection variations on the V_{th} distribution of the memory cells.

3.1 Floating gate cell physics

The program and erase operations alter the number of stored charges in the memory cell. Therefore they have to take into account the specific properties of the used memory cell. This section presents the structural design of the floating gate memory cell as well as a capacitive cell model. There are a lot of different floating gate cell concepts and corresponding program and erase methods in use today, each optimized and specific to their field of applications. In the following we focus on the state of the art ONO IPD floating gate cell for mainstream data Flash applications.

[1] Institute for Technical Electronics, Technische Universität München, christoph.Friederich@mytum.de

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_3,
© Springer Science+Business Media B.V. 2010

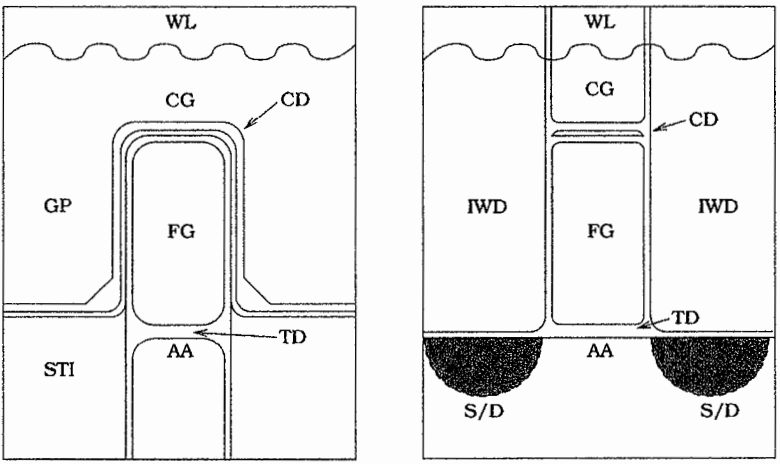
Petitioner Micron Ex-1021, 0064

3.1.1 The ONO IPD floating gate cell

The floating gate memory cell was proposed by D. Kahng and S. Sze at the Bell Labs [1]. Based on the *F*loating-Gate *E*lectron *T*unneling *MOS* (FETMOS) cell [2], it has gone through several structural innovations [3], becoming the *S*elf-*A*ligned *S*hallow *T*rench *I*solation (SA-STI) cell [4] which is widely used today and shown in Fig. 3.1. The dielectrics surrounding the floating gate are efficiently isolating the charge storage node, enabling the retention of the stored information for years without any power supply. In contrast to the *Flo*ating-Gate *T*unneling *Ox*ide (FLOTOX) cell [5], it uses the active channel oxide as tunneling medium instead of employing separate thin oxide areas or additional electrodes.

The side wall oxide and the *I*nter *W*ord line *D*ielectric (IWD) isolate the floating gates from each other. The *I*nter *P*oly *D*ielectric (IPD) separates the poly silicon *C*ontrol *G*ate (CG) from the poly silicon *F*loating *G*ate (FG). It is typically a $SiO_2 - Si_3N_4 - SiO_2$ (ONO) triple layer, but the use of a SiO_2 – high-k dual layer for future technology nodes is in discussion [6]. In the most recent technologies in production the Si_3N_4 of the triple stack is replaced by a higher-k metal-oxide based material [7].

To recognize also future developments in barrier engineering [8, 9] and gate materials [10], in the following the more general terms *T*unneling *D*ielectric (TD) and *C*oupling *D*ielectric (CD) are preferred over the specific terms tunneling oxide and inter poly dielectric and their material implications. So the tunneling dielectric describes the part of the floating gate isolation which is located between the floating gate and the silicon surface. There, the charges are injected during the program and erase operations. The coupling dielectric describes the part of the floating gate isolation which is located between the floating gate and the control gate.



(a) STI cross section (b) CG cross section

Fig. 3.1. The ONO IPD floating gate cell

Petitioner Micron Ex-1021, 0065

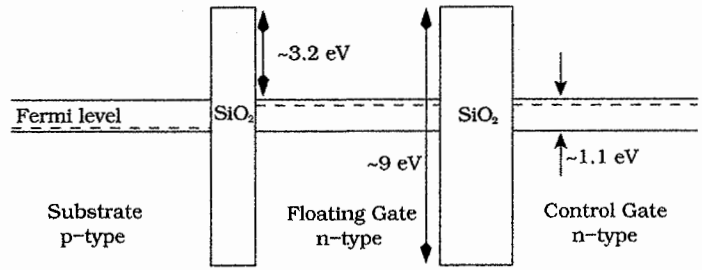
3.1.2 The band diagram of the floating gate cell

The conducting floating gate is an equipotential surface which enables homogeneous electric field conditions at the plan surface regions of the floating gate.

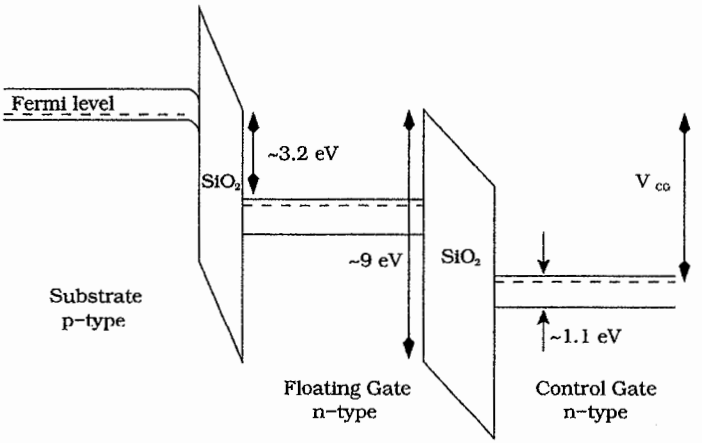
To understand the cell physics and its operation principles the band diagram must be reviewed. The floating gate builds a potential well with the large band-gap materials used for the tunneling and coupling dielectrics as shown in Fig. 3.2a.

This potential well holds the injected charges on the floating gate. The energy barrier for the electrons assuming an n-type silicon floating gate and control gate is $\Phi_B = 3.2$ eV. Often the ONO IPD is replaced by a SiO_2 placeholder with the same *E*quivalent *O*xide *T*hickness (EOT). While for the program operation (among the others) this simplification is valid, in some cases the actual CD configuration is relevant [11, 12].

The potential on the floating node is influenced by capacitive coupling with its environmental nodes as well as by the stored charges. The cell structure is chosen in a way that the floating gate node is strongly coupled to the potential of its control gate.



(a) Flat band condition for the floating gate structure



(b) The floating gate structure with a positive control gate potential applied

Fig. 3.2. Band diagrams of a floating gate cell

Petitioner Micron Ex-1021, 0066

This ensures the band bending in the tunneling dielectric to be larger than in the coupling dielectric as indicated in Fig. 3.2b. This asymmetry of the electric field across the TD and the CD is the key reason for the fact that injected charges stay on the floating gate and are not transferred further to the control gate during the program operation.

3.1.3 Capacitive cell model

The strong capacitive coupling of the floating gate to the control gate by the coupling dielectric can be modeled by a capacitive cell model. The model (see Fig. 3.3) for the floating gate cell can be derived from the definition of the capacitance:

$$C_i = \frac{\delta Q_i}{U_i} = \frac{\delta Q_i}{\psi_{FG} - \psi_i} \tag{3.1}$$

The charge Q_{FG} on the floating node ψ_{FG} is the sum of all partial charges $\sum \delta Q_i$ and with Eq. (3.1) it is

$$Q_{FG} = \sum_i \delta Q_i = \sum_i C_i \cdot (\psi_{FG} - \psi_i) = \sum_i (C_i \cdot \psi_{FG} - C_i \cdot \psi_i) \tag{3.2}$$

Therefore the potential of the floating node ψ_{FG}, Eq. (3.2) can be written as

$$\psi_{FG} = \frac{Q_{FG}}{\sum_i C_i} + \sum_i \frac{C_i}{\sum_i C_i} \cdot \psi_i = \frac{Q_{FG}}{C_{FG}} + \sum_i \frac{C_i}{C_{FG}} \cdot \psi_i \tag{3.3}$$

The electrostatic condition in the semiconductor channel of the floating gate cell is linked to the floating gate potential by the MOS capacitance like it is to the gate potential in a regular MOSFET.

However the floating gate potential can not be directly accessed like the gate in a MOSFET, but is dependent from the control gate potential, from the potential of all other surrounding nodes as well as from the stored charges as indicated by Eq. (3.3).

Figure 3.1 shows the SA-STI floating gate cell which is commonly used for data Flash memories. The coupling dielectric and the control gate (CG) with its gate plugs (GP) wrap the FG on three sides. Considering all capacitances shown in Fig. 3.1 the Eq. (3.3) can be formed to

$$\begin{aligned} \psi_{FG} \;=\; & \frac{Q_{FG}}{C_{FG}} + \frac{C_S}{C_{FG}} \cdot \psi_S + \frac{C_D}{C_{FG}} \cdot \psi_D \\ & + \frac{C_{TD}}{C_{FG}} \cdot \psi_{Ch} + \frac{C_{CD}}{C_{FG}} \cdot \psi_{CG} \end{aligned} \tag{3.4}$$

The coupling ratios C_i / C_{FG} are a measure for the influence of the specific node's potential on the potential of the floating gate. Of high importance for the

Petitioner Micron Ex-1021, 0067

program operation is the gate coupling ratio C_{CD}/C_{FG} since it correlates the control gate potential with the floating gate one.

In a MOSFET, one definition for the threshold voltage is the gate bias, beyond flat-band, just starting to induce an inversion charge sheet at the semiconductor channel. Accordingly for a floating gate cell it can be defined as the control gate bias causing a floating gate potential which induces an inversion charge sheet at the semiconductor channel. In contrast to the MOSFET it is not only dependent on the device parameters but it is also dependent on both the potential of the surrounding voltage nodes and the amount of stored charges.

Normally the environmental conditions during the read-out operation are constant and therefore do not contribute to the V_{th} shift (Chap. 8). At the threshold voltage $V_{th} = \psi_{CG,1}$ the floating node ψ_{FG} has a certain potential [13, p. 312]:

$$\psi_{Vth} = \psi_{FG} = V_{FB} + 2\psi_B + \frac{\sqrt{2\epsilon_S q N_A (2\psi_B)}}{C_{TD}}$$

V_{FB}	[V]	Flat-band voltage
ψ_B	[V]	Fermi level from intrinsic Fermi level
ε_S	$1.04 \cdot 10^{-12}$ [F/cm]	Permittivity of silicon
q	$1.60 \cdot 10^{-19}$ [C]	Elementary charge
N_A	[cm^{-3}]	Acceptor impurity concentration

If the stored charges are changed, the same potential ψ_{Vth} is established at the floating node for a different control gate potential $\psi_{CG,2}$. From Eq. (3.3) the threshold voltage shift caused by the injection of charges can be derived

$$\begin{aligned}
\psi_{Vth} &= \psi_{FG} = \frac{Q_{FG,1}}{C_{FG}} + \frac{C_{CD}}{C_{FG}} \cdot \psi_{CG,1} + \sum \frac{C_i}{C_{FG}} \cdot \psi_{i,1} \\
\psi_{Vth}(Q_{FG,1}, \psi_{CG,1}) &= \psi_{Vth}(Q_{FG,2}, \psi_{CG,2}) \\
0 &= \frac{Q_{FG,2}}{C_{FG}} - \frac{Q_{FG,1}}{C_{FG}} + \frac{C_{CD}}{C_{FG}} \cdot \psi_{CG,1} - \frac{C_{CD}}{C_{FG}} \cdot \psi_{CG,2} \\
&\quad + \underbrace{\sum \frac{C_i}{C_{FG}} \cdot \psi_{i,2} - \sum \frac{C_i}{C_{FG}} \cdot \psi_{i,1}}_{0 \quad \wedge \psi_{i,1} = \psi_{i,2}} \\
0 &= \frac{\Delta Q_{FG}}{C_{FG}} + \frac{C_{CD}}{C_{FG}} \cdot \Delta V_{th} \\
\Delta V_{th} &= -\frac{\Delta Q_{FG}}{C_{CD}}
\end{aligned} \tag{3.5}$$

It must be highlighted that the number of charges per V_{th} shift is not primarily dependent from the physical size of the floating gate nor from the overall floating gate capacitance. The only parameter which influences the number of electrons is the coupling capacitance to the control gate C_{CD}. For a high number of stored charges the capacitance of the coupling dielectric needs to be increased as much as possible, e.g. by increasing the floating gate height. This enhances the capacitance

Petitioner Micron Ex-1021, 0068

between the control gate plugs and the floating gate by area. Another strategy to serve this purpose is to decrease the electrical thickness of the coupling dielectric by the use of materials possessing a higher dielectric constant.

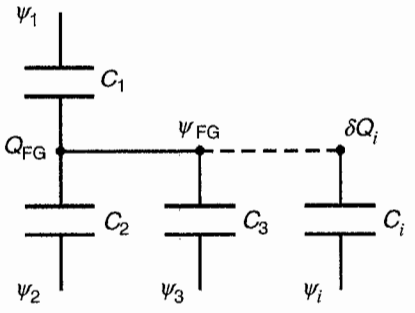


Fig. 3.3. Capacitive model of the floating gate cell

3.2 Altering the stored charges

As already outlined, the stored charges in the floating gate shift the cell's V_{th} according to the user data. The following section explains the physical mechanism of the charge transfer to the floating gate and how it is used by the programming algorithm. Finally the interaction of some cell parameters with the programming algorithm is discussed.

3.2.1 Fowler–Nordheim tunneling mechanism

By applying a high electric potential between the channel and the control gate, the band structure of the device is strongly bent. This effectively reduces the tunneling distance throughout the TD barrier and enables a substantial current into the charge storage layer for programming or erasing the stored information. The orientation of this current is given by the polarization of the applied voltages. According to Eq. (3.4) the gate coupling ratio $\alpha = C_{CD}/C_{FG}$ defines how the applied control gate potential ψ_{CG} is linked to the floating gate potential ψ_{FG}:

$$\psi_{FG} \propto \frac{C_{CD}}{C_{FG}} \cdot \psi_{CG,1} = \alpha \cdot \psi_{CG} \tag{3.6}$$

The electric field E across the tunneling dielectric, assuming a grounded channel, is given by:

$$E = \frac{\psi_{FG}}{d_{TD}} \propto \alpha \cdot \psi_{CG} \tag{3.7}$$

Petitioner Micron Ex-1021, 0069

Since ψ_{FG} directly translates into the electric field across the tunneling dielectric by its physical thickness d_{TD}, α is the main parameter for the required control gate voltages during the program and erase operations.

Field emission or *F*owler–*N*ordheim (FN) tunneling is the dominant tunneling mechanism through potential barriers of relevant thickness (≥2 nm) at high electric fields [14]. Due to the strong bending of the band structure the effective barrier thickness is lowered (see Fig. 3.4) and therefore the tunneling probability is increased.

The tunneling current density J_{FN} through a SiO_2 barrier can be described by [15]:

$$J_{FN} = \mathrm{a} \cdot \frac{E^2}{\Phi_B} \cdot exp\left(\frac{-\mathrm{b} \cdot \Phi_B^{\frac{3}{2}}}{E}\right) \tag{3.8}$$

where $\mathrm{a} = q^3 / (16\,\pi^2\,\hbar)$ and $\mathrm{b} = (4 / 3\hbar) \cdot (2\,m^* q)^{1/2}$.

E	[MV/cm]	Electric field at the injecting interface
Φ_B	[eV]	Potential barrier at the interface
$\hbar$	$1.06\ 10^{-34}$ [Js]	Reduced Planck's constant
m^*	[kg]	Electron effective mass in silicon oxide

Two second order effects can be taken into account for a more accurate description: the influence of temperature and the image force barrier lowering.

Due to the electrostatic effect of a charge near the semiconductor surface, the image force lowers the effective barrier height. It can be described by the introduction of two correction functions $\mathrm{t}(\Delta\Phi_B)$ and $\mathrm{v}(\Delta\Phi_B)$ which are both tabulated elliptic integrals as a function of the normalized barrier lowering $\Delta\Phi_B$ [15]:

$$\Delta\Phi_B = \frac{1}{\Phi_B}\sqrt{\frac{q^3 E}{4\,\pi\,\epsilon_{ox}}} \tag{3.9}$$

ε_{ox}	$3.45 \cdot 10^{-13}$ [F/cm]	Permittivity of silicon oxide

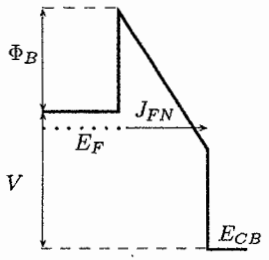


Fig. 3.4. Fowler-Nordheim tunneling through a potential barrier

Petitioner Micron Ex-1021, 0070

Although the tunneling is basically independent from temperature, the Fermi-Dirac statistics for the occupation of the conduction band by electrons is not. Taking this into account, a correction factor for the temperature T dependency described by a reciprocal sinc function is introduced:

$$f(T) = \frac{1}{\operatorname{sinc}(\pi ckT)} = \frac{\pi ckT}{\sin(\pi ckT)} \tag{3.10}$$

where

$$c = \frac{2\sqrt{2m^*}\, t(\Delta\Phi_B)}{\hbar qE}$$

k	$1.38 \cdot 10^{-23}$ [J/K]	Boltzmann's constant

In modern device technologies the electron energy quantization in the accumulation region must be also considered for an accurate modeling of the tunneling current. It results in a field dependent lowering of the oxide barrier height and an electric field reduction across the barrier [16].

Fowler–Nordheim tunneling shares with other write mechanisms the fact that the device is stressed by the injection of charges through or over the barrier. This results in a limited amount of write cycles until either the electric characteristics of the device is changed or the data retention is degraded.

3.2.2 Incremental step pulse programming ISPP

In order to control the programmed V_{th} of the NVM cell a bit-by-bit program verify algorithm is used [17]. Therefore the program operation is split in several program pulse steps with a V_{th} verification (sensing) operation (Chap. 8) in between. If the V_{th} of a cell is detected above a certain program verify level, further programming of this single cell will be stopped by setting it in a program inhibit state (Sect. 3.4). For FN tunneling used as programming mechanism the gate voltage of the program pulse V_{pp} is increased by a constant value $\Delta V_{pp} = V_{step}$ after each program step [18]. Therefore, this programming scheme is called *I*ncremental *S*tep *P*ulse *P*rogramming (ISPP). The cell's V_{th} shifts by ΔV_{th} which is equal to V_{step} as shown in Fig. 3.5a.

The graphs in Fig. 3.5 were obtained from a simple capacitor model of a floating gate cell (Fig. 3.1). By integrating the field dependent tunneling current across the tunneling dielectric according to Eq. (3.8) during each programming pulse, the cells V_{th} shift is obtained from Eq. 3.5.

The reason for the equivalence of ΔV_{th} and V_{step} is the self limitation of the tunneling current. The already tunneled charges are decreasing the electric field between the gate and the channel (Fig 3.5b) and therefore the tunneling current decreases.

Petitioner Micron Ex-1021, 0071

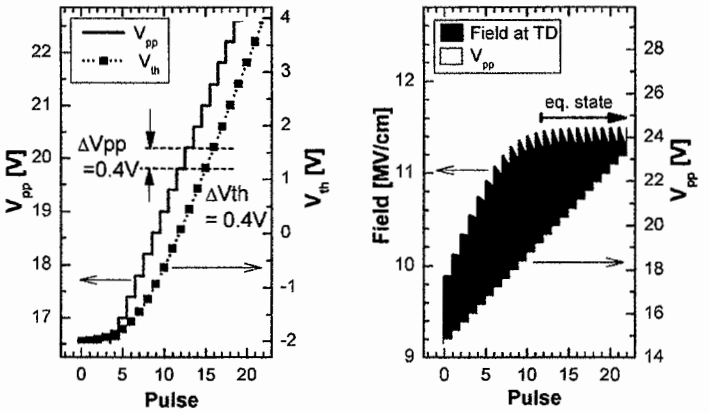
The time t dependent electric field $E(t)$ across the tunneling dielectric can be expressed using Eqs. (3.3) and (3.7) as:

$$
\begin{aligned}
E(t) &= \frac{\psi_{FG}}{d_{TD}} = \frac{1}{d_{TD}} \cdot \left[\frac{Q_{FG}(t)}{C_{FG}} + \frac{C_{CD}}{C_{FG}} \cdot \underbrace{\psi_{CG}}_{V_{pp}} + \underbrace{\sum \frac{C_i}{C_{FG}} \cdot \psi_i}_{constant} \right] \\
&= \frac{1}{d_{TD}} \cdot \left[\frac{Q_{FG}(t)}{C_{FG}} + \frac{C_{CD}}{C_{FG}} \cdot V_{pp} - \frac{C_{CD}}{C_{FG}} \cdot V_{th,Q=0} \right. \\
&\quad \left. + V_{FB} + 2\psi_B + \frac{\sqrt{2\epsilon_s q N_a (2\psi_B)}}{C_{TD}} \right] \\
&= \frac{1}{d_{TD}} \cdot \left[\underbrace{-\frac{C_{CD}}{C_{FG}}}_{\alpha} \cdot \Delta V_{th}(t) - \frac{C_{CD}}{C_{FG}} \cdot V_{th,Q=0} + \frac{C_{CD}}{C_{FG}} \cdot V_{pp} \right. \\
&\quad \left. + V_{FB} + 2\psi_B + \frac{\sqrt{2\epsilon_s q N_a (2\psi_B)}}{C_{TD}} \right] \\
&= \frac{1}{d_{TD}} \cdot \left[-\alpha \left(\overbrace{\Delta V_{th}(t) + V_{th,Q=0}}^{V_{th}(t)} \right) + \alpha \cdot V_{pp} \right. \\
&\quad \left. + V_{FB} + 2\psi_B + \frac{\sqrt{2\epsilon_s q N_a (2\psi_B)}}{C_{TD}} \right] \\
&= \frac{1}{d_{TD}} \cdot \left[\alpha \cdot [V_{pp} - V_{th}(t)] + V_{FB} + 2\psi_B + \frac{\sqrt{2\epsilon_s q N_a (2\psi_B)}}{C_{TD}} \right] \\
E(t) &\sim \frac{1}{d_{TD}} \cdot \alpha \cdot [V_{pp} - V_{th}(t)] \qquad (3.11)
\end{aligned}
$$

In the next program step s the electric field is raised again by applying the higher programming voltage $V_{pp,s} = V_{pp,s-1} + V_{step}$ to the control gate. This feedback loop enables an equilibrium state for the mean electric field and therefore also for the mean tunneling current. This process repeats at each program step and as indicated in Eq. (3.11) it results in a constant V_{th} shift ΔV_{th} with the constant increase of V_{pp} by V_{step}. For this reason incremental step pulse programming is often referred to as constant current programming scheme. However, this may be misleading since the tunneling current decreases during each program pulse as the electric field decreases, but only its mean value is constant over several consecutive program steps.

In the same way the injected charges decrease the electric field across the TD, they increase the field across the CD. How tunneled charges alter the band diagram of a floating gate structure is shown in Fig. 3.6. Compared to the neutral state in Fig. 3.6a, the tunneled charges raise the floating gate potential and therefore decrease the electric field in the TD as well as increasing the electric field in the CD as indicated in Fig. 3.6b.

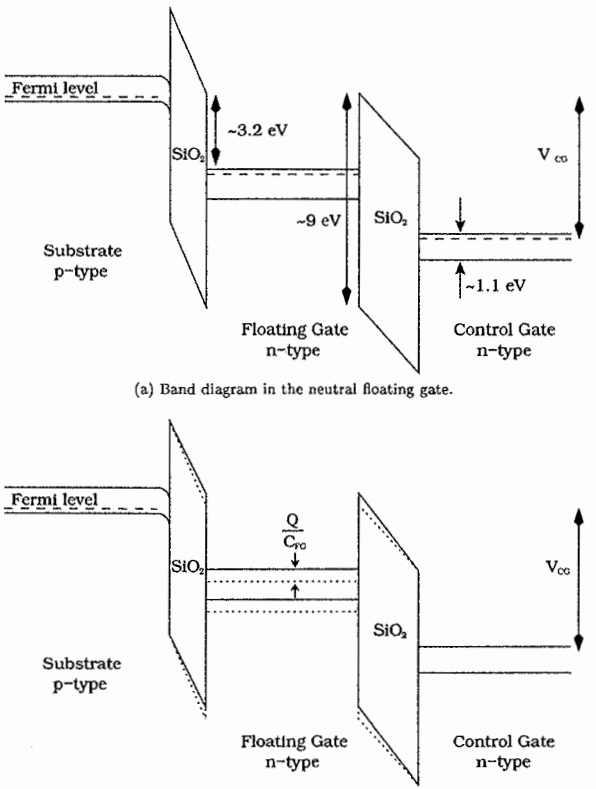
Petitioner Micron Ex-1021, 0072



(a) Constant V_{th} shift during incremental gate stepping.

(b) Electric field across tunneling dielectric over program pulses.

Fig. 3.5. Programming in incremental step pulse programming schemes



(a) Band diagram in the neutral floating gate.

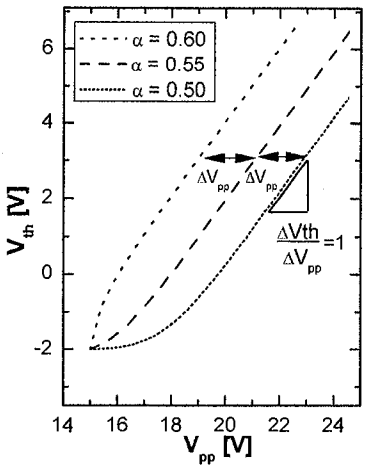
(b) Band diagram after charge injection (solid) compared to the neutral state (dashed).

Fig. 3.6. Influence of stored charges on the band diagram of a floating gate structure

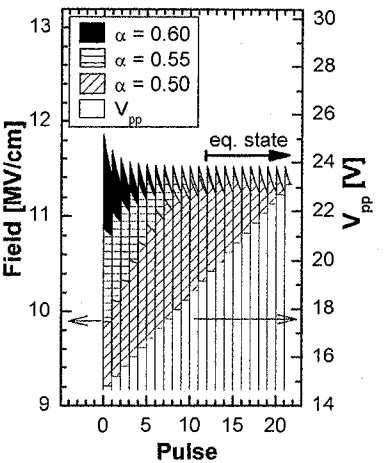
Petitioner Micron Ex-1021, 0073

3.2.3 Interaction between cell parameters and ISPP

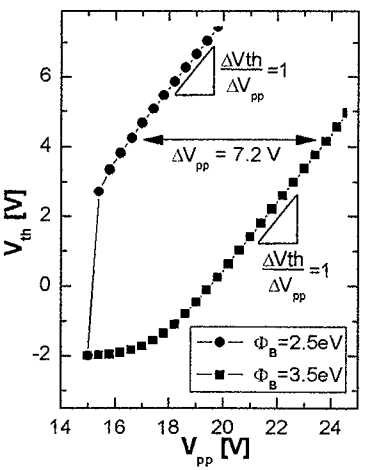
It must be highlighted that due to the feedback loop the V_{th} shift in such incremental step pulse programming schemes is independent from any device parameter.



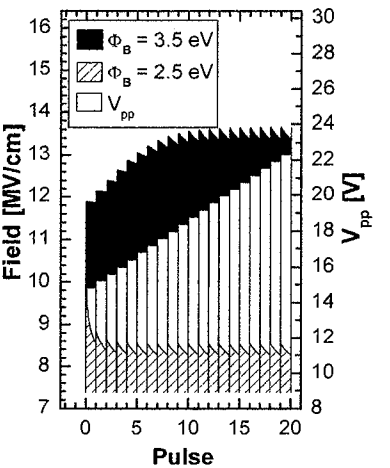
(a) Threshold voltage shift in dependence of the programming voltage for different gate coupling ratio α.



(b) Electric field across tunneling dielectric for different gate coupling ratio α.



(c) Threshold voltage shift in dependence of the programming voltage for two dielectric barrier heights Φ_B.



(d) Electric Field across tunneling dielectric for varying barrier height Φ_B.

Fig. 3.7. Interaction between incremental step pulse programming scheme and cell parameters

Petitioner Micron Ex-1021, 0074

The control gate coupling ratio α only influences the required program pulse voltages (Fig. 3.7a and b), since it changes the ratio between the control gate voltage and the floating gate potential (Eq. (3.6)). Parameters like the barrier height Φ_B or the effective electron mass m^* are influencing the control gate voltage required to program to a certain V_{th} level. Moreover they have a strong impact on the mean electric field in the equilibrium state as shown in Fig. 3.7d. Therefore they may also influence the reliability due to the field dependent charge to breakdown [19].

Although the presented ISPP algorithm is able to hide some cell parameters and compensate their variations to a certain extent, they still enable the functionality of the memory cell. In highly scaled memory cells, the required field asymmetry at the TD and CD is hard to guarantee for a sufficient V_{th} range. This results in program saturation effects [20] and it degrades the previously stated (and for the practical use highly important) feedback loop.

3.3 The NAND array

The previous sections described the memory cell and its program mechanism. However modern non-volatile memories contain millions and billions of memory cells, organized in a memory array. Row and column decoders address the cells in the memory array, while write and read circuits control the data input and output operations. This section briefly recalls the key properties of the NAND array architecture relevant for the program and erase operations.

The core components of the NAND Flash memory are shown in Fig. 3.8. In the NAND array the cells are organized in pages, blocks and planes. The page is the unit for the simultaneous program and read operation, the block is the smallest possible erase unit consisting of several pages. The top and bottom page buffers are connected to a pair of bit lines which, in a shielded bit line scheme [21], are addressed alternately. The bit lines are connected through the string select transistor to a number of memory transistors in series, which are forming the NAND string. The ground select transistor isolates the string from the *S*ource *L*ine (SL). The row decoders control the select lines and word lines in the block. The NAND strings in a block are selected by the *S*tring *S*elect *L*ine (SSL) and the *G*round *S*elect *L*ine (GSL). The blocks are connected alternately to the left and right row decoder. The two sided staggered page buffer and row decoder placement relaxes the space requirement in the layout of these circuits. However recent designs propose single sided layout for both, in order to improve synchronization between them. Moreover single sided layout improves the cell area efficiency thanks to the lack of wiring and circuits at three edges of the chip [22]. The unit sizes are given by the array layout, effectively hiding the physical and technological constraints of the NAND array from the external world.

The page size is proportional to the number of cells per word line and it is given by the word line length limited by its RC delay and the bit line pitch limited by lithography. The block size is proportional to the number of cells per NAND

Petitioner Micron Ex-1021, 0075

string and it is limited by the circuit ability of handling its series resistance during the read operation (see Chap. 9). By increasing the page size and therefore the number of cells operated in parallel the read and write performance can be increased. By increasing the block size, the cell array efficiency is increased, since more cells share both the necessary select gate and contact area.

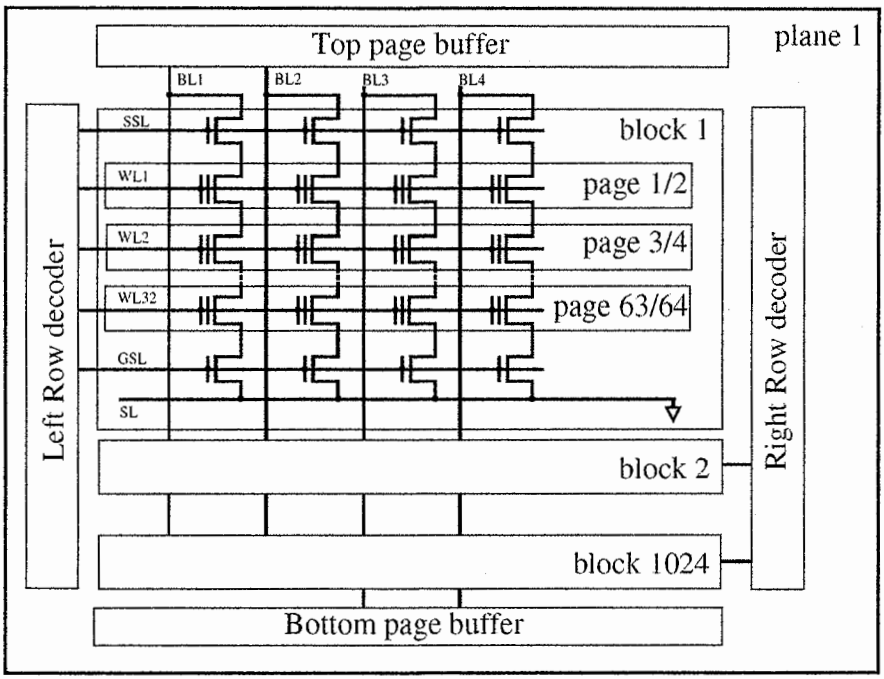


Fig. 3.8. Schematic NAND array architecture

3.4 The program operation and its side effects in the NAND array

The program operation alters the floating gate potential of the selected cells to a specified target level by injecting charges to the floating gate. With the previously described incremental step pulse programming scheme, the programming voltage V_{pp} applied to the word line of the selected page is increased by a constant increment V_{step} for each program pulse. This increases the charges on the floating gate by an amount which shifts the cells V_{th} by a constant increment ΔV_{th}. After each program pulse the page is read and the cells which either have reached the target V_{th} level or should not be programmed at all are set to program inhibit for the consecutive program pulses. The program operation has finished when either all cells' V_{th} are higher than the target V_{th} level or an overall program pulse limit is reached.

Petitioner Micron Ex-1021, 0076

The following section describes this program operation, in particular the programming algorithm. The primary task of the programming algorithm is to selectively shift the cells' V_{th} (bit by bit) to their target level as accurately as possible. Therefore it has to take into account the interference effects between the memory transistors caused by the sharing of voltage nodes in the NAND architecture. The following section focuses on the bit selectivity by means of the self-boosted program inhibit (SBPI) scheme and on the corresponding capacitive string model. Furthermore the relevant disturb effects during SBPI are presented. The last subsection is about advanced SBPI schemes which are able to reduce the disturb effects during the program operation, resulting in a more accurate V_{th} control.

3.4.1 Self-boosted program inhibit SBPI

During the program operation, the cells share the high programming voltage on the selected word line, but the program operation has to be bit selective. Therefore a high channel potential is able to reduce the voltage drop across the tunneling dielectric, and it prevents the electrons from tunneling from the channel to the floating gate as indicated by Fig. 3.9a. In the first NAND Flash devices, the channel was charged by applying a high voltage, e.g. 8 V, to the bit lines connected to the program inhibited NAND strings. But this method suffers from several disadvantages [23]:

1. The charge pump has to provide the high inhibit voltage on the high capacitive bit lines. This is power consuming and it requires a large silicon area for the charge pumps.
2. There is a high voltage drop between the bit lines connected to inhibited strings and adjacent bit lines connected to strings which have to be programmed. This limits the scalability of the memory array due to field breakdown boundary conditions of the isolating inter bit line dielectrics.
3. The page buffer size is increased due to the transistor size required to handle high voltages in the input path.

The self boost program inhibit scheme [18] is less power-hunger. By charging the string select lines and bit lines connected to inhibited cells to the voltage of the common collector V_{cc}, the select transistors are closed since the channel potential is higher than $V_{cc} - V_{th,sg}$. At this point the channel of the NAND string becomes a floating node. By raising the word line potential (selected word line to V_{pp} and unselected word lines to V_{ppass}), the channel potential is boosted by the coupled series capacitance through the control gate, floating gate, channel and bulk. The bias conditions of the lines and the corresponding timing diagram of the SBPI operation are shown in Fig. 3.9b and c, respectively. The voltages applied by the page buffers to the specific bit lines control whether the select transistors are closed or open, therefore initiating the self-boosted program inhibit. Since the page buffers also evaluate the V_{th} state of the memory cells during the verify operation, they have full control on the ISPP operation. This ensures the bit

Petitioner Micron Ex-1021, 0077

selectivity of the program algorithm, despite V_{pp} is applied to all the cells sharing the selected word line.

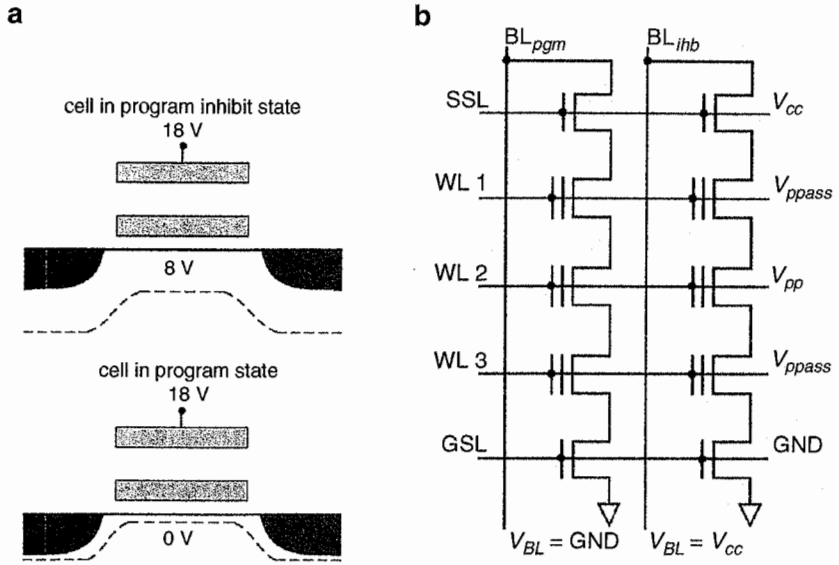


Fig. 3.9. Self-boosted program inhibit scheme

Petitioner Micron Ex-1021, 0078

3.4.2 Capacitive model of the NAND string

The self-boosted program inhibit can be described by a model of the isolated NAND string and the involved capacitances. The boosted potential ΔV_{ch} in a NAND string containing N cells and modeled by the capacitance network shown in Fig. 3.10a can be expressed by

$$\begin{aligned}\Delta V_{ch} &= \sum_i \frac{C_{cell}}{\sum_i (C_{cell}+C_{dep})} \cdot \Delta V_{wl,i} \\ &= \frac{C_{cell}}{N\,(C_{cell}+C_{dep})}(N-1)V_{ppass} \\ &\quad + \frac{C_{cell}}{N\,(C_{cell}+C_{dep})} \cdot V_{pp} \end{aligned} \tag{3.12}$$

where C_{cell} is the simplified cell capacitance in the capacitance network of the memory cell in Fig. 3.10b, as indicated by Fig. 3.10c.

The boosting efficiency $\Delta V_{ch}/V_{ppass}$ is a measure of the V_{ppass} voltage required to boost the channel potential V_{ch} by ΔV_{ch}. Since the boosted channel potential reduces the electric field across the TD during the program inhibit, a high boosting efficiency is desired.

From Eq. (3.12) the boosting efficiency is derived:

$$\begin{aligned}\frac{\Delta V_{ch}}{V_{ppass}} &= \frac{C_{cell}}{C_{cell}+C_{dep}} \cdot \left[\frac{(N-1)}{N} + \frac{1}{N} \cdot \frac{V_{pp}}{V_{ppass}}\right] \\ &= \frac{C_{cell}}{C_{cell}+C_{dep}} \left[1 + \frac{1}{N}\left(\frac{V_{pp}}{V_{ppass}} - 1\right)\right] \end{aligned} \tag{3.13}$$

The relation given by Eq. (3.13) contains two important facts about the boosting efficiency:

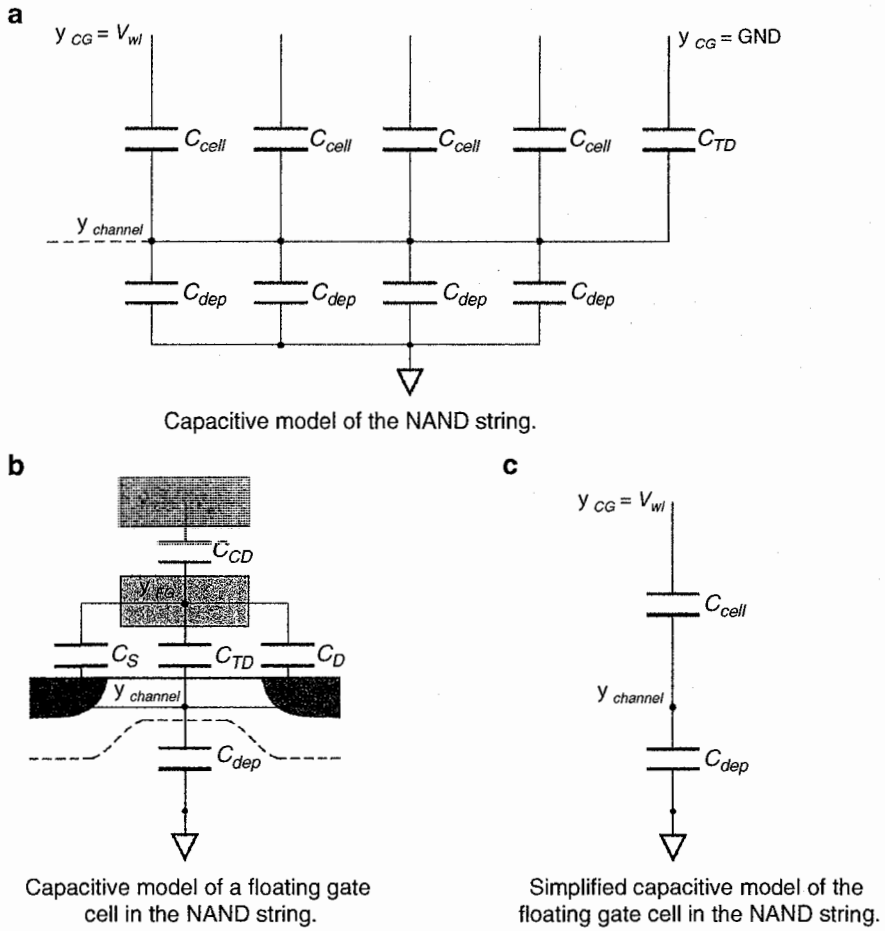
1. The boosted channel potential can be increased by minimizing the channel depletion capacitance C_{dep}. The cell isolation scheme, as well as the channel implantation dose, are therefore important technology parameters [24]. For the same reason, fully depleted SOI or gate all around cell concepts, where C_{dep} is zero, are featuring a very high boosting efficiency.
2. The boosted channel potential increases with smaller string length N. Since the selected word line self boosts the channel with the higher programming voltage V_{pp} compared to the unselected word lines with V_{ppass}, the boosting efficiency increases with

$$\frac{1}{N} \cdot \left(\frac{V_{pp}}{V_{ppass}} - 1\right)$$

For an increased boosting efficiency, the modification of the NAND array was proposed as well, which introduces an additional channel boost capacitance

Petitioner Micron Ex-1021, 0079

perpendicular to the word lines [25]. Indeed the boosted channel potential is increased, but this solution is not commonly used due to its additional process complexity and area consumption.



Capacitive model of the NAND string.

Capacitive model of a floating gate cell in the NAND string.

Simplified capacitive model of the floating gate cell in the NAND string.

Fig. 3.10. Capacitive models of the NAND string

3.4.3 Disturb effects

Disturb effects alter the memory transistors threshold voltage unintentionally during memory access operations under the influence of specific disturb conditions. Since it is essential for an efficient area consumption to arrange the storage transistors in a contact saving way, the sharing of voltage nodes can not be avoided. During the read and program operations, positive voltages are applied to the gate nodes of the memory transistors, wherein the channel is at a lower

Petitioner Micron Ex-1021, 0080

potential or even grounded. Therefore this condition is called gate disturb (also word line disturb) and it is the most common disturb mechanism in NAND Flash memory arrays. The positive voltage is able to cause Fowler–Nordheim tunneling of electrons to the floating gate. This soft programming disturbs the memory transistor's stored state albeit with a slower rate than at the intended program operation. Since the tunneling current has an exponential dependency from the voltage drop across the tunneling dielectric, the programming time required to cause the same V_{th} shift as an intended programming pulse is much longer. However this fact limits the *N*umber *O*f *P*rograms (NOP) which can be realized without influencing the stored memory state. Since the floating gate potential and therewith the electric field across the tunneling dielectric are dependent on the cells' V_{th} (Eq. (3.11)), erased cells are most vulnerable to soft programming disturb effects. For disturb effects which slightly erase cells, the highest V_{th} state is most vulnerable due to the same reason. During the program operation there are mainly three disturb effects, which all affect cells in a different location of the memory array.

3.4.3.1 Program disturb

The program disturb applies to the cells sharing the word line with the selected cells but are set to program inhibit (cell A in Fig. 3.11). Although the program inhibit boosts the channel potential, soft programming can not be avoided especially when a high number of program pulses are applied. The effective programming voltage for these cells is $V_{pp} - V_{ch}$.

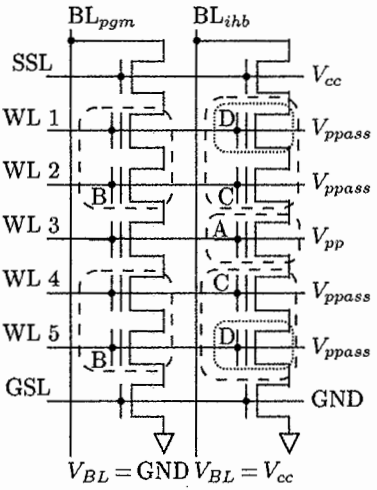


Fig. 3.11. Disturb during the program operation in NAND array architecture

Petitioner Micron Ex-1021, 0081

As previously stated, V_{ch} or rather ΔV_{ch} is proportional to the applied V_{ppass} in SBPI schemes. Therefore the program disturb can be reduced by increasing V_{ppass} at the expense of an increased pass disturb.

3.4.3.2 Pass disturb

As stated above, the program disturb can be reduced by increasing V_{ppass}. Unfortunately this increases the pass disturb, which applies to the cells located in the same NAND string as a selected cell (cell B in Fig. 3.11). In this case the channel potential is set to ground and the gate nodes are set to the V_{ppass} voltage. Therefore the effective programming voltage for these cells is V_{ppass}.

Usually there is no pass disturb on the unselected cells in program inhibited NAND strings (cell C in Fig. 3.11), since V_{ppass} and the self-boosted channel potential show only minor differences, insufficient for FN tunneling to take place. However for short NAND strings based on cell concepts with a high boosting efficiency due to the missing C_{dep} like SOI devices [26] or gate all around concepts [27], a negative pass disturb (soft erase) during the program operation can be expected and in fact was recently reported [28]. In this case the programming pulse V_{pp} boosts the channel potential V_{ch} to a much higher level (see Sect. 3.4.2) than the gate voltage V_{ppass} applied to the passing word lines. This inverse voltage drop for cells at position C causes electrons to be tunneled out of their floating gates and results in a lower cell V_{th}.

3.4.3.3 Edge disturb

Recently a new disturb mechanism affecting only the edge cells of the NAND string was reported [29]. It is caused by hot carriers which are created by the potential drop between the select transistor and the program inhibited NAND string with its boosted channel potential. The accelerated electrons can stray into the edge cells (cell D in Fig. 3.11) and there causing a V_{th} shift. The overall disturb effect is among other parameters, like the maximum voltage on the edge WL, also dependent from the distance of the select gate to the edge memory cells. This fact may be a blocking point for further scaling, but could be bypassed by the introduction of dummy word lines [30]. These additional memory cells are located between the select transistors and the other memory cells of the NAND string, but are not used for data storage and so are never set to the high V_{pp}.

3.4.4 Advanced SBPI schemes

As described in Sect. 3.4.1 the program inhibit is based on boosting the potential of the floating channel to higher voltages to reduce the electric field at the TD and therefore prevents charge injection. So SBPI schemes have to create a channel potential which is as high as possible. However, like all operations involving floating nodes, it is extremely vulnerable to leakages. Especially because of the

Petitioner Micron Ex-1021, 0082

low capacitance of the channel, even small leakage currents can degrade the channel potential and result in an increased program disturb.

One leakage path is the punch through current at the select transistors. The high potential drop between the boosted channel and the bit or source line during the program inhibit require a certain minimal length of the select transistor. Another issue at the select transistor is caused by the capacitive coupling between the select line and the adjacent word line. During the word line ramp up, capacitive bounces can slightly open the select device in the sub threshold region, therefore increasing the leakage [31]. Besides special word line voltage ramping techniques, schemes which lower the select line voltage during the word line ramp up were proposed [32, 33].

Other leakage paths comprise junction leakage in the depletion region of the source/drain junctions as well as substrate leakage in the depletion region of the channel's inversion layer.

Besides the boosted potential ΔV_{ch} expressed by Eq. (3.12), also the precharge potential in the channel $V_{ch,pre}$ sums up to the boosted channel potential $V_{ch,boost}$:

$$V_{ch,boost} = V_{ch,pre} + \Delta V_{ch} \tag{3.14}$$

Advanced SBPI schemes were proposed to maximize $V_{ch,pre}$ in the channel either from the bit line [34] or from the source line [35]. However in the case of non-selectively precharging of all NAND strings, either by the bit line or by the source line, a selective discharge is required. This must be done through the bit line and its connected page buffer since it holds both the result of the verify operation and the user data.

3.4.4.1 Local SBPI schemes

In contrast to the global SBPI schemes outlined in the previous section and in Sect. 3.4.1, local SBPI schemes [36, 37] are mostly independent from string length and V_{ppass}. With these schemes, the program inhibited cell is isolated from its NAND string by the adjacent cells. Of course, this is true for all the strings belonging to the addressed word line. This allows a higher boosting efficiency due to the exclusive coupling of the cell's localized channel to the selected word line. This is in contrast to the global SBPI wherein the whole channel is primarily coupled to the unselected word lines just because of their number. The bias conditions of the local SBPI schemes are shown in Fig. 3.12a. The V_{ppass} applied to the unselected word lines boosts the channel potential, which closes the transistors adjacent to the selected cell due to their body effect. The program inhibited cell forms a single floating node as indicated in Fig. 3.12. The boosted potential of Eq. (3.12) reduces to:

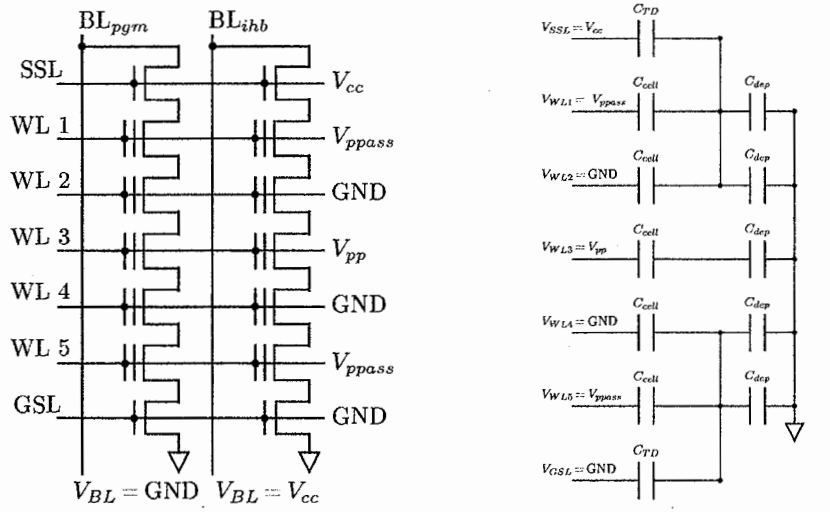
$$\Delta V_{ch} = \frac{C_{cell}}{(C_{cell} + C_{dep})} \cdot V_{pp} \tag{3.15}$$

Although the local SBPI scheme features higher boosting efficiency, it also faces some problems. Especially the low voltage applied to the shielding word line

Petitioner Micron Ex-1021, 0083

next to the selected word line and its program voltage V_{pp} is an issue. First of all, it requires a sequential page programming order from the GSL to the SSL, since the ground potential of the bit line needs to be passed to the selected cell in the case of programming. The programming order ensures the cells on the shielding word line in SSL direction to be erased and open. Second, the local field conditions cause additional hot carrier disturb effects in scaled technologies [38].

Another issue is the electric field strength in the inter word line dielectric. Because of device scaling, also the word lines are getting closer, increasing the electric field caused by the potential drop between the selected and the shielding word line. The critical field strength of the isolating IWD is a scaling limit for memories with local SBPI. However channel engineering may allow local self-boosting conditions for the program inhibited cell without the need for low voltages applied to the adjacent word lines [39].



(a) Bias conditions during the local self boosted program inhibit.

(b) Capacitive model of local self boost program inhibit.

Fig. 3.12. The classic local self boost program inhibit

3.4.4.2 Asymmetric SBPI

Asymmetric SBPI schemes are programming algorithms with a sequential page programming order and asymmetric voltage conditions for the erased part of the string and the already programmed part. The main task of asymmetric SBPI schemes is to reduce the background pattern dependency of the program inhibit. With higher V_{th} of the cells in the NAND string, the pass disturb shifts to higher V_{ppass} as shown in Fig. 3.13. This reduces the V_{ppass} window, where the program disturb for all V_{th} target levels as well as the pass disturb is below a certain level.

Petitioner Micron Ex-1021, 0084

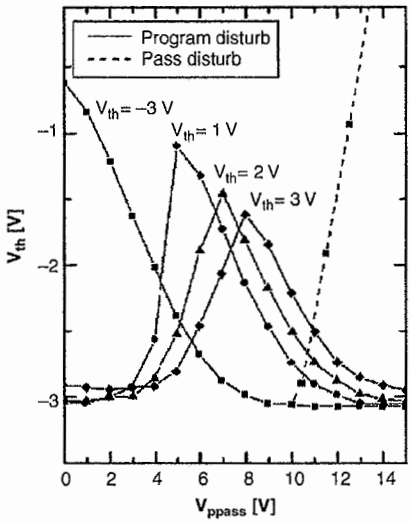


Fig. 3.13. Background pattern dependent program disturb. Data is extracted from [25]

Dependent on the V_{th} of the cells in the string, the precharged potential $V_{ch,pre}$ propagates into the string. During the self-boosted inhibit all transistors in the string are turned on, and the $V_{ch,pre}$ is reduced due to charge sharing with the whole string capacitance. This degradation of $V_{ch,boost}$ causes the background pattern dependent program disturb. Several asymmetric SBPI schemes were proposed [40–43] to prevent the charge sharing by isolating the already programmed part of the string during the program inhibit by various bias conditions as indicated by the example in Fig. 3.14.

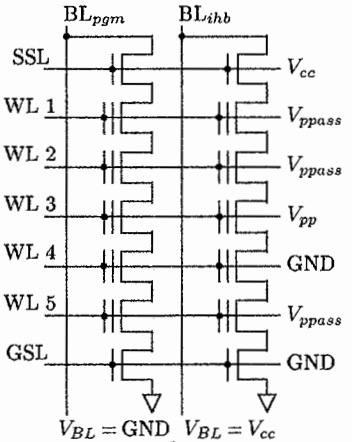


Fig. 3.14. Bias conditions during an asymmetric self-boosted program inhibit

Petitioner Micron Ex-1021, 0085

3.5 The erase operation and the NAND array

The erase operation resets the information of all cells in one block simultaneously. For this purpose the common p-well of the memory array is charged to a high positive voltage (~20 V). The word lines of the selected block are grounded. This results in a high electric field across the tunneling dielectric of the cells in the selected block and causes the stored electrons to be tunneled out of the floating gate to the substrate, lowering the cells' V_{th}. In modern technology nodes, the electric field between adjacent word lines would result in dielectric breakdown when a non-zero voltage would be applied to some of them in order to prevent the erase. For this reason the block, i.e. the smallest erase unit, contains all word lines located between the two select gates (Chap. 8). The erase operation could also be realized by applying negative voltages to all word lines of the selected block, as is the case with silicon on insulator array approaches, where no bulk is available at all. However in this case the row decoders would have to be able to switch both high negative and high positive voltages (for the programming), with an increased area consumption. After the erase pulse, an erase verify is done to ensure that all cells are well erased, followed by additional erase pulses if required. Because of the positive voltage supply design commonly used in NAND Flash systems, the challenge is to be able to detect the negative V_{th} of the memory cells with no negative voltage available.

During the erase operation the bit lines and the source line grid are at p-well potential (off the junction built-in potential), since the n-junctions at the bit line contacts are biased in forward direction by applying positive voltages to the well. Therefore the low voltage logic devices of the page buffer have to be isolated from the bit lines of the cell array during the erase operation.

3.5.1 Self-boosted erase inhibit SBEI

Like in the self-boosted program inhibit, wherein the isolated NAND string is boosted by raising the word lines to high voltages, word lines can be boosted to high voltages by the ramped potential of the shared p-well. The electrically isolated word lines of the unselected blocks are boosted according to the capacitance ratio of C_{cell} and the word line capacitance C_{wl}, while the word lines of the selected block are kept grounded. By the increased potential of the floating word lines in the unselected blocks, the tunneling of charges out of their floating gates is prevented [23].

3.5.2 Erase disturb

Erase disturb, unlike the program disturb, is not usually a big issue in NAND Flash memories. Since the erase operation affects all cells located in one block at the same time, there is no relevant intra-block interference. However, like all self-boosted inhibits which rely on the capacitive coupling of floating nodes, the SBEI

Petitioner Micron Ex-1021, 0086

is vulnerable to poor boosting efficiency and to insufficient electric isolation of the floating nodes. Leakage currents in the row decoder can degrade the boosted word line potential, resulting in a soft erase of the programmed cells in the unselected blocks. These issues are taken care of by both the row decoder design and process technology.

3.6 Stochastic effects: Impact on cell distributions

In recent years, stochastic effects in scaled Flash memories came to research interest concerning the aspects of data retention e.g. [44–46] and the read operation [47–49]. However there are also stochastic effects affecting the program operation, which are discussed in this section. These effects influence the program operation and broaden the obtained V_{th} distribution. They limit the theoretical accuracy of the program operation, and they are therefore of great interest. The following section discusses three stochastic effects in NAND Flash memories which can be addressed by the used programming algorithm.

Since process variations were present since the first Flash memories were manufactured, they were also the first stochastic effect taken into account in the program operation.

The second stochastic effect described in this section is the floating gate cross coupling. Due to its increasing effect on the cell distribution with smaller device dimensions, it requires more and more sophisticated programming algorithms to reduce its effective impact.

The third effect is the random fluctuation of the transferred charges during the program operation, due to its quantum mechanic nature. It is most prominent in few electron memories, but also will be an issue for the future scaling of state of the art floating gate memories.

The last subsection is about stochastic *M*odels for *C*ell *S*ystem *I*nteraction (MCSI) which can be used to evaluate the impact of these effects on the resulting V_{th} distribution. During the concept phase, MCSI allow both the evaluation of expected effects and the trade-off between its algorithmic, design and technology countermeasures.

3.6.1 Process variations

As stated in Sect. 3.2, the programming mechanism used in NAND Flash is FN tunneling. The most important cell parameter for FN tunneling is the electric field across the tunneling dielectric. The gate coupling ratio α defines the ratio between the applied voltage at the control gate and the electric field across the coupling dielectric (Eq. (3.11)):

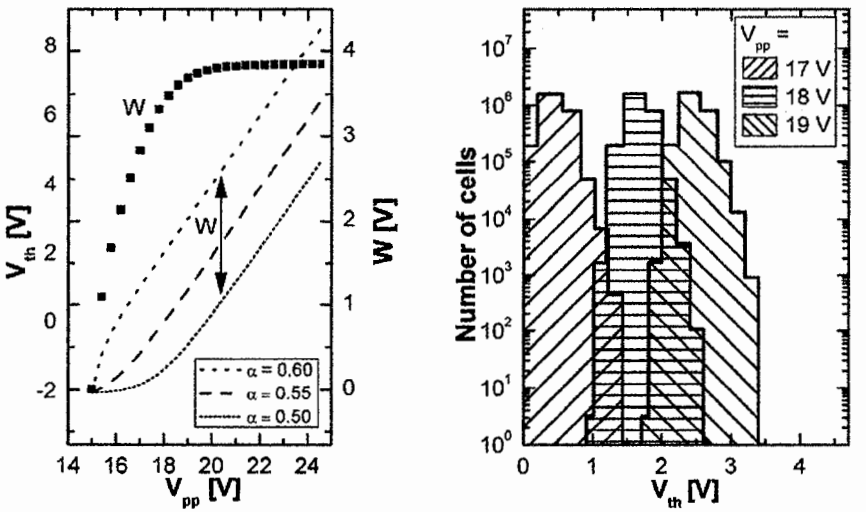
$$E = \frac{\psi_{FG}}{d_{TD}} \propto \alpha \cdot \frac{\psi_{CG}}{d_{TD}}$$

Petitioner Micron Ex-1021, 0087

Technology variations in cell dimensions, layer thicknesses and distance to the neighboring cells influence the capacitive conditions of each cell and result in a distribution of the cell parameter α and therefore also influence the electric field. When a fixed programming voltage is applied to the word line, and therefore to the control gate of several memory transistors, these variations in the electric field result in a distribution of the amount of tunneled charges. For this reason there is a native V_{th} distribution width due to the statistical spread of the cells' programmability.

The intrinsic minimal distribution width W of the memory array after one single program pulse, is the difference $W = V_{th,max} - V_{th,min}$ in the V_{th} between the fastest programmable cell and the slowest programmable one.

The influence of the gate coupling ratio on the programming behavior for three cells with different cell parameters is shown in Fig. 3.15. As already mentioned, different programming pulse voltages are required to start the programming for the cells featuring different α values. This increases the distribution width after each programming pulse until a quasi-static equilibrium state is reached. The intrinsic distribution width W is a measure of the technological ability to control cell variations. However process variations can not be avoided and therefore were subject to algorithmic countermeasure to hide their impact on the memory system. They were the main motivation for the use of a program verify in combination with a bit selective program inhibit operation [17].



(a) Threshold voltage shift in dependence of the programming voltage for different gate coupling ratio α in ISPP scheme.

(b) V_{th} distribution of cells in a 4Mb memory array without verify operation. The data is taken from [50].

Fig. 3.15. Intrinsic distribution width W due to cell to cell variations

Petitioner Micron Ex-1021, 0088

3.6.2 Floating gate cross-coupling

As derived in Sect. 3.1.3 the threshold voltage of the floating gate cell is not a device constant but it is also dependent on its environment. In fact it is sensitive to the stored charges on adjacent floating gates and therefore the V_{th} of a cell is unintentionally shifted when neighboring cells are programmed. This issue was first addressed in NOR architecture [51] but it is more severe in NAND architecture [52] because of the smaller cell size ($4F^2$) and therefore smaller distance to adjacent cells. For the same reason, device scaling increases the floating gate cross coupling effect. Figure 3.16 shows the relevant capacitances involved in the floating gate cross coupling effect in NAND array architecture.

Similarly to Eq. (3.5), the V_{th} shift caused by a shift $\Delta V_{th,i}$ on an adjacent cell can be described by

$$\begin{aligned} 0 &= \frac{C_{CD}}{C_{FG}} \cdot \Delta V_{th} + \frac{C_{cc,i}}{C_{FG}} \cdot \Delta\psi_{FG,i} \\ &= \frac{C_{CD}}{C_{FG}} \cdot \Delta V_{th} + \frac{C_{cc,i}}{C_{FG}} \cdot \frac{\Delta Q_i}{C_{FG}} \\ &= \frac{C_{CD}}{C_{FG}} \cdot \Delta V_{th} + \frac{C_{cc,i}}{C_{FG}} \cdot \frac{-C_{CD} \cdot \Delta V_{th,i}}{C_{FG}} \\ \Delta V_{th,cc} &= \frac{C_{cc,i}}{C_{FG}} \cdot \Delta V_{th,i} = \gamma_i \cdot \Delta_{V th,i} \end{aligned} \tag{3.16}$$

Considering Eq. (3.16) for each of the eight neighbor cells and their specific cross coupling ratio γ_i, the overall V_{th} shift by cross coupling $\Delta V_{th,cc}$ can be described by

$$\Delta V_{th,cc} = \sum_i \gamma_i \cdot \Delta_{V th,i} \tag{3.17}$$

The relevant coupling ratios and their contributions in each direction are indicated in Fig. 3.17. The face to face cross coupling capacitance $C_{cc,ff}$ is the largest parasitic capacitance followed by the cross coupling capacitance along the word line $C_{cc,wl}$. It must be noted that the coupling along the word line also includes the direct interference from the floating gate to the neighbor string channel [53]. The shielding effect of the control gate plugs reduces $C_{cc,wl}$ and therefore the gate plug height above the AA is a relevant parameter. The diagonal cross coupling capacitance $C_{cc,diag}$ is the smallest parasitic capacitance because of the larger distance to the relevant neighbor cells. However in today's technology nodes the impact of the diagonal cells can not be neglected anymore.

Normally floating gate cross interference is not seen as a stochastic effect due to its deterministic nature (Eq. (3.17)). However the two factors γ_i and also $\Delta_{Vth,i}$ can be seen as random variables. The first factor γ_i is due to process variations which are slightly altering the capacitive ratios. The second factor $\Delta_{Vth,i}$ is related to the user data, which in most cases can be considered as random.

Petitioner Micron Ex-1021, 0089

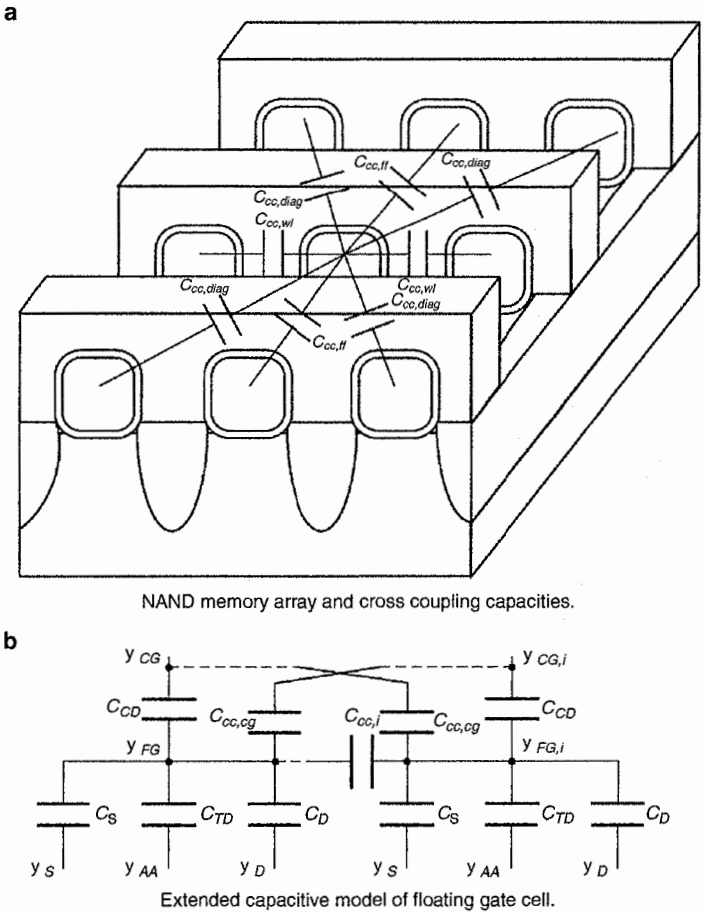


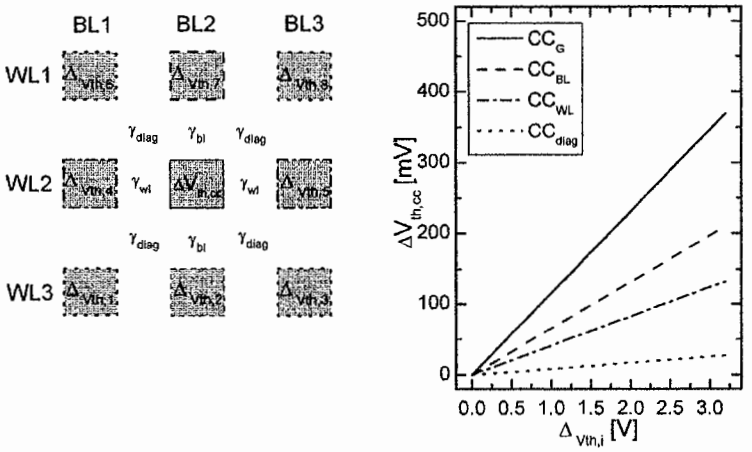
Fig. 3.16. Floating gate cross coupling in NAND architecture

Therefore also the impact of floating gate cross coupling on the cells' V_{th} distribution can be described by a stochastic formalism, especially when not a single cell, but a large cell ensemble is observed.

Due to the increasing cross coupling effect at smaller technology nodes, recently this issue is seen as a severe scaling limit for NAND Flash memories [54]. Despite technology countermeasures are under evaluation [55], very successful algorithmic methods were proposed [56] to suppress the negative impact of floating gate cross coupling. These methods compensate the shift caused by the cross coupling effect of the already programmed neighboring cells by injecting fewer charges:

$$\Delta V_{th} = \Delta V_{th,cc} - \frac{\Delta Q}{C_{CD}}$$

Petitioner Micron Ex-1021, 0090



(a) Top view on a cell array centered on the victim cell and relevant cross coupling coefficients.

(b) Cross coupling shift $\Delta V_{th,cc}$ in dependence from the shift on all surrounding cells $\Delta_{Vth,i}$ in each direction. The cross coupling coefficients γ were obtained from 3D field simulations at 48 nm.

Fig. 3.17. Maximum amount of cross coupling in NAND architecture

In a first step the random page programming order is replaced by a sequential one which can take into account the capacitive influence already in place due to three out of eight neighboring cells [31] in addition to the reduction of the back pattern dependency [36]. Considering the cross coupling effect of more neighbors require a complex mapping of the page address to the physical cell location combined with multiple programming steps. With these sophisticated program algorithms, even the feasibility of 4 bit per cell storage, requiring excellent V_{th} control, was demonstrated [57].

3.6.3 Injection statistics

In NAND Flash, Fowler–Nordheim tunneling is used as injection mechanism, which is by itself a physical effect based on quantum mechanics statistics. In previously technology generations this effect was greatly mitigated by the large number of electrons stored in the floating gate. This large number was balancing the tunneling current and it compensated tunneling fluctuations, resulting in a very conform programming operation.

However the injection statistics of electrons was subject of research in single or few electrons based memory devices [58–60]. Recently it was also experimentally demonstrated for state of the art floating gate memory cells in ISPP schemes [61]. For small $\Delta V_{th} < 1$ V it can be expressed by its standard deviation $\sigma_{\Delta Vth}$:

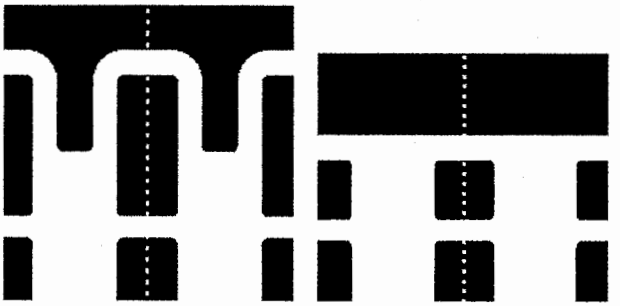
$$\sigma_{\Delta Vth} = \sqrt{\frac{q}{C_{CD}} \cdot \Delta V_{th}} \tag{3.18}$$

Petitioner Micron Ex-1021, 0091

For larger ΔV_{th} the feedback loop on the mean tunneling current – due to the electric field of already injected charges (Sect. 3.2.2) – results in a saturation of $\sigma_{\Delta Vth}$ [62]. As indicated by Eq. (3.18) the deviation can be reduced either by reducing ΔV_{th} (dependent on program algorithm parameter V_{step}) or by increasing C_{CD} (dependent on technology and the memory cell concept used). Both countermeasures are valid options for handling the intrinsic fluctuations of the tunneling process for future NAND Flash memories [63].

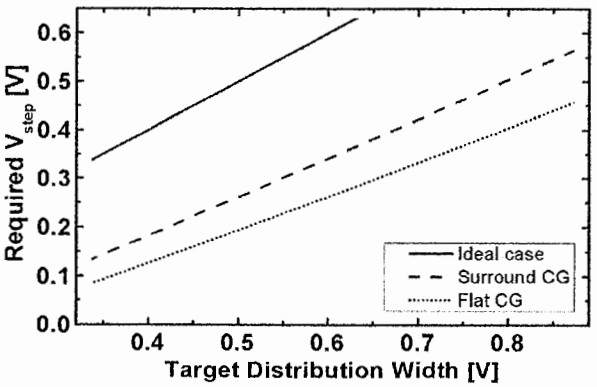
3.6.4 Models for Cell–System Interaction

As just mentioned, in most cases noise effects or other interferences on the cells' V_{th} distribution can be either controlled or managed by a variety of different approaches. Often a trade-off between these options concerning costs, time and effort as well as their effectiveness needs to be evaluated in a very early phase of the memory development.



(a) STI cross section of a surround control gate geometry.

(b) STI cross section of a flat control gate geometry.



(c) Required V_{step} for a certain target distribution width to compensate the injection fluctuations of the two cell geometries at 36 nm.

Fig. 3.18. Impact of cell geometries on the memory system

Petitioner Micron Ex-1021, 0092

*M*odels for *C*ell–*S*ystem *I*nteraction (MCSI) are able to assist this balancing activity. In real memory cells, a whole bunch of different effects are present and they all have their own influence on the resulting cell's V_{th}. On the other side they are often mostly independent from each other and additionally they can be described by a probability density function (pdf). Luckily, from a memory system designer point of view, the behavior of each specific memory cell doesn't need to be predicted. A typical NAND Flash memory system contains several billions of cells and as long as the cell ensemble as a whole behaves as required, the memory system is functional. Therefore a stochastic approach for describing the cell ensemble is chosen, which describes each contribution of the effects to the cells' V_{th} by a random variable No_i. The sum of all random variables represents the net effect on the cells' V_{th} distribution and can be described by the convolution * of their probability density functions [64]:

$$pdf_{net}(\Delta V_{th}) = pdf_{No_1}(\Delta V_{th}) * pdf_{No_2}(\Delta V_{th}) * \cdots * pdf_{No_i}(\Delta V_{th}) \quad (3.19)$$

Based on such a model, the impact of these effects on the memory system can be estimated. A MCSI for the impact of various cell geometries on the injection fluctuations, outlined in the previous subsection, was recently presented [63]. The cell geometries shown in Fig. 3.18a and b (i.e. their C_{CD}) influence the required algorithm setting. The flat control gate geometry [65] features a smaller C_{CD} due to the missing area of the GP assuming the same EOT of the coupling dielectrics. The increase in the standard deviation indicated by Eq. (3.18) can be compensated by a reduced V_{step} for a certain target distributions width as shown in Fig. 3.18c. Unfortunately, a reduced V_{step} results in a degradation of the programming speed [66] so that for high performance Flash memories the surround control gate cell (Fig. 3.18a) is favorable.

References

1. D. Kahng and S. Sze, "A floating-gate and its application to memory devices," The Bell System Technical Journal, vol. 46, no. 4, pp. 1288–1295, 1967.
2. J. Yeargain and C. Kuo, "A high density floating-gate EEPROM cell," in International Electron Devices Meeting, 1981. IEDM'81,, vol. 27, 1981, pp. 24–27, 1981.
3. G. Ginami et al., "Survey on Flash technology with specific attention to the critical process parameters related to manufacturing," Proceedings of the IEEE, vol. 91, no. 4, pp. 503–522, April 2003.
4. S. Aritome et al., "A 0.67 μm2 self-aligned shallow trench isolation cell (SA-STI cell) for 3 V-only 256 Mbit NAND EEPROMs," in International Electron Devices Meeting, 1994. IEDM '94. Technical Digest., pp. 61–64, Dec 1994.
5. W. Johnson et al., "A 16Kb electrically erasable nonvolatile memory," in Solid-State Circuits Conference. Digest of Technical Papers. 1980 IEEE International, vol. XXIII, pp. 152–153, Feb. 1980.

Petitioner Micron Ex-1021, 0093

6. B. Govoreanu, D. Brunco, and J. V. Houdt, "Scaling down the interpoly dielectric for next generation Flash memory: Challenges and opportunities," Solid-State Electronics, vol. 49, no. 11, pp. 1841–1848, Nov 2005.
7. K. Gibb and H. Gu. (2008, 11) High-k Gate Dielectrics – the Future Is Friendly for NAND Flash. Chipworks. [Online]. Available: http://www.chipworks.com/blogs/
8. K. K. Likharev, "Layered tunnel barriers for nonvolatile memory devices," Applied Physics Letters, vol. 73, no. 15, pp. 2137–2139, 1998. [Online]. Available: http://link.aip.org/link/?APL/73/2137/1
9. M. Specht, M. Städele, and F. Hofmann, "Simulation of High-K Tunnel Barriers for Nonvolatile Floating Gate Memories," in Proceeding of the 32nd European Solid-State Device Research Conference, Sept. 24–26, 2002, pp. 599–602.
10. N. Chan et al., "Metal control gate for sub-30nm floating gate NAND memory," in Non-Volatile Memory Technology Symposium, 2008. NVMTS 2008. 9th Annual, pp. 1–4, Nov. 2008.
11. S. Mori et al., "ONO inter-poly dielectric scaling for nonvolatile memory applications," IEEE Transactions on Electron Devices, vol. 38, no. 2, pp. 386–391, Feb 1991.
12. B. Govoreanu et al., "Investigation of the low-field leakage through high-k interpoly dielectric stacks and its impact on nonvolatile memory data retention," in International Electron Devices Meeting, 2006. IEDM '06, pp. 1–4, Dec. 2006.
13. S. Sze and K. K. Ng, Physics of Semiconductor Devices, 3rd ed. Wiley, Inc., Hoboken, NJ, 2007.
14. R. H. Fowler and L. Nordheim, "Electron emission in intense electric fields," Proceedings of the Royal Society of London, vol. 119, no. 781, pp. 173–181, May 1928.
15. M. Lenzlinger and E. H. Snow, "Fowler-Nordheim tunneling into thermally grown SiO," Journal of Applied Physics, vol. 40, no. 1, pp. 278–283, 1969. [Online]. Available: http://link.aip.org/link/?JAP/40/278/1
16. J. Sune, P. Olivo, and B. Ricco, "Quantum-mechanical modeling of accumulation layers in MOS structure," IEEE Transactions on Electron Devices, vol. 39, no. 7, pp. 1732–1739, July 1992.
17. T. Tanaka et al., "A 4-Mbit NAND-EEPROM with tight programmed Vt distribution," Symposium on VLSI Circuits, 1990. Digest of Technical Papers, pp. 105–106, June 1990.
18. K.-D. Suh et al., "A 3.3 V 32 Mb NAND Flash memory with incremental step pulse programming scheme," Solid-State Circuits Conference, 1995. Digest of Technical Papers. 42nd ISSCC, 1995 IEEE International, pp. 128–129, 350, Feb 1995.
19. P. Apte and K. Saraswat, "Correlation of trap generation to charge-to-breakdown (Qbd): a physical-damage model of dielectric breakdown,"IEEE Transactions on Electron Devices, vol. 41, no. 9, pp. 1595–1602, Sept 1994.
20. M. F. Beug, N. Chan, T. Hoehr, L. Mueller-Meskamp, and M. Specht, "Investigation of program saturation in scaled interpoly dielectric floating-gate memory devices," IEEE Transactions on Electron Devices, vol. 56, no. 8, pp. 1698–1704, Aug. 2009.
21. T. Tanaka et al., "A quick intelligent page-programming architecture and a shielded bitline sensing method for 3 V-only NAND Flash memory," IEEE Journal of Solid-State Circuits, vol. 29, no. 11, pp. 1366–1373, Nov 1994.
22. K. Takeuchi et al., "A 56nm CMOS 99mm2 8Gb Multi-level NAND Flash Memory with 10MB/s Program Throughput," Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International, pp. 507–516, Feb. 2006.

Petitioner Micron Ex-1021, 0094

23. K.-D. Suh et al., "A 3.3 V 32 Mb NAND Flash memory with incremental step pulse programming scheme," IEEE Journal of Solid-State Circuits, vol. 30, no. 11, pp. 1149–1156, Nov. 1995.
24. S. Satoh et al., "A novel isolation-scaling technology for NAND EEPROMs with the minimized program disturbance," International Electron Devices Meeting, 1997. IEDM '97. Technical Digest., pp. 291–294, Dec 1997.
25. S. Satoh et al., "A novel Channel Boost Capacitance (CBC) cell technology with low program disturbance suitable for fast programming 4 Gbit NAND Flash memories," in VLSI Technology, 1998. Symposium on Digest of Technical Papers, pp. 108–109, June 1998.
26. C. Friederich, M. Specht, T. Lutz, F. Hofmann, L. Dreeskornfeld, W. Weber, J. Kretz, T. Melde, W. Rösner, E. Landgraf, J. Hartwich, M. Stadele, L. Risch, and D. Richter, "Multi-level p+ tri-gate SONOS NAND string arrays," IEDM' 06. International Electron Devices Meeting Technical Digest, pp. 1–4, Dec. 2006.
27. H. Tanaka et al., "Bit Cost Scalable Technology with Punch and Plug Process for Ultra High Density Flash Memory," in IEEE Symposium on VLSI Technology, June 2007, pp. 14–15.
28. Y. Komori et al., "Disturbless Flash memory due to high boost efficiency on BiCS structure and optimal memory film stack for ultra high density storage device," in IEEE International Electron Devices Meeting (IEDM' 08), pp. 1–4, Dec. 2008.
29. J.-D. Lee et al., "A New Programming Disturbance Phenomenon in NAND Flash Memory By Source/Drain Hot-Electrons Generated By GIDL Current," Non-Volatile Semiconductor Memory Workshop, 2006. IEEE NVSMW 2006. 21st, pp. 31–33, 2006.
30. K.-T. Park et al., "Scalable Wordline Shielding Scheme using Dummy Cell beyond 40 nm NAND Flash Memory for Eliminating Abnormal Disturb of Edge Memory Cell," Japanese Journal of Applied Physics, vol. 46, no. 4B, pp. 2188–2192, 2007. [Online]. Available: http://jjap.ipap.jp/link?JJAP/46/2188/
31. T. Cho et al., "A dual-mode NAND Flash memory: 1-Gb multilevel and high-performance 512-Mb single-level modes," IEEE Journal of Solid-State Circuits, vol. 36, no. 11, pp. 1700–1706, Nov. 2001.
32. J. Lee et al., "A 1.8 V 2 Gb NAND Flash memory for mass storage applications," IEEE International Solid-State Circuits Conference, 2003. Digest of Technical Papers, ISSCC , pp. 290–494, vol.1, 2003.
33. J. Lee et al., "A 90-nm CMOS 1.8-V 2-Gb NAND Flash memory for mass storage applications," IEEE Journal of Solid-State Circuits, vol. 38, no. 11, pp. 1934–1942, Nov. 2003.
34. T.-S. Jung et al., "A 3.3-V single power supply 16-Mb nonvolatile virtual DRAM using a NAND Flash memory technology," IEEE Journal of Solid-State Circuits, vol. 32, no. 11, pp. 1748–1757, Nov. 1997.
35. K. Takeuchi, S. Satoh, K. Imamiya, and K. Sakui, "A source-line programming scheme for low-voltage operation NAND Flash memories," IEEE Journal of Solid-State Circuits, vol. 35, no. 5, pp. 672–681, May 2000.
36. T.-S. Jung et al., "A 3.3 V 128 Mb multi-level NAND Flash memory for mass storage applications," Solid-State Circuits Conference, 1996. Digest of Technical Papers. 42nd ISSCC., 1996 IEEE International, pp. 32–33, 412, Feb 1996.

Petitioner Micron Ex-1021, 0095

37. T.-S. Jung et al., "A 117-mm2 3.3-V only 128-Mb multilevel NAND Flash memory for mass storage applications," IEEE Journal of Solid-State Circuits, vol. 31, no. 11, pp. 1575–1583, Nov. 1996.
38. S. Hur et al., "Effective program inhibition beyond 90 nm NAND Flash memories," in Proceedings of IEEE 20th Non-Volatile Semiconductor Memory Workshop, pp. 44–45, 2004.
39. D. Oh et al., "A New Self-Boosting Phenomenon by Soure/Drain Depletion Cut-off in NAND Flash Memory," in Proceedings of 22nd IEEE Non-Volatile Semiconductor Memory Workshop, pp. 39–41, Aug. 2007.
40. J.-y. Jeong, J.-s. Yeom, and S.-s. Lee, "Method of programming non-volatile semiconductor memory device," Patent 20 020 118 569, August, 2002. [Online]. Available: http://www.freepatentsonline.com/y2002/0118569.html
41. J. W. Lutze, J. Chen, Y. Li, and M. Higashitani, "Source side self boosting technique for non-volatile memory," Patent 6 859 397, February, 2005. [Online]. Available: http://www.freepatentsonline.com/6859397.html
42. J.-K. Kim, H.-B. Pyeon, H. Oh, R. Schuetz, and P. Gillingham, "Low Stress Program and Single Wordline Erase Schemes for NAND Flash Memory," in Proceedings of 22nd IEEE Non-Volatile Semiconductor Memory Workshop, pp. 19–20, Aug. 2007.
43. K.-T. Park, M. Kang, S. Hwang, Y. Song, J. Lee, H. Joo, H.-S. Oh, J.-h. Kim, Y.-t. Lee, C. Kim, and W. Lee, "Dynamic Vpass ISPP scheme and optimized erase Vth control for high program inhibition in MLC NAND Flash memories," in Symposium on VLSI Circuits, pp. 24–25, June 2009.
44. D. Ielmini, A. Spinelli, A. Lacaita, and A. Modelli, "Statistical modeling of reliability and scaling projections for Flash memories," in International Electron Devices Meeting, 2001. IEDM Technical Digest, pp. 32.2.1–32.2.4, 2001.
45. L. Larcher, "Statistical simulation of leakage currents in MOS and Flash memory devices with a new multiphonon trap-assisted tunneling model," IEEE Transactions on Electron Devices, vol. 50, no. 5, pp. 1246–1253, May 2003.
46. L.-C. Hu, A.-C. Kang, J. Shih, Y.-F. Lin, K. Wu, and Y.-C. King, "Statistical modeling for postcycling data retention of split-gate Flash memories," IEEE Transactions on Device and Materials Reliability, vol. 6, no. 1, pp. 60–66, March 2006.
47. H. Kurata et al., "The Impact of Random Telegraph Signals on the Scaling of Multilevel Flash Memories," Symposium on VLSI Circuits, 2006. Digest of Technical Papers, pp. 112–113, 2006.
48. N. Tega et al., "Anomalously Large Threshold Voltage Fluctuation by Complex Random Telegraph Signal in Floating Gate Flash Memory," in International Electron Devices Meeting, 2006. IEDM '06, pp. 1–4, Dec. 2006.
49. C. M. Compagnoni et al., "Statistical Model for Random Telegraph Noise in Flash Memories," IEEE Transactions on Electron Devices, vol. 55, no. 1, pp. 388–395, Jan. 2008.
50. M. Momodomi et al., "A 4 Mb NAND EEPROM with tight programmed Vt distribution," IEEE Journal of Solid-State Circuits, vol. 26, no. 4, pp. 492–496, Apr 1991.
51. J. Chen and Y. Fong, "High density non-volatile Flash memory without adverse effects of electric field coupling between adjacent floating gates," US Patent 5 867 429, February, 1999. [Online]. Available: http://www.freepatentsonline.com/5867429.html
52. J.-D. Lee, S.-H. Hur, and J.-D. Choi, "Effects of floating-gate interference on NAND Flash memory cell operation," IEE Electron Device Letters, vol. 23, no. 5, pp. 264–266, May 2002.

Petitioner Micron Ex-1021, 0096

53. A. Ghetti, L. Bortesi, and L. Vendrame, "3D Simulation study of gate coupling and gate cross-interference in advanced floating gate non-volatile memories," Solid-State Electronics, vol. 49, no. 11, pp. 1805–1812, 2005.
54. K. Kim, "Technology for sub-50nm DRAM and NAND Flash manufacturing," in IEEE International Electron Devices Meeting, 2005. IEDM Technical Digest, pp. 323–326, Dec. 2005.
55. D. Kang et al., "Improving the Cell Characteristics Using Low-k Gate Spacer in 1Gb NAND Flash Memory," in International Electron Devices Meeting, 2006. IEDM '06, pp. 1–4, Dec. 2006.
56. K.-T. Park, "A Zeroing Cell-to-Cell Interference Page Architecture with Temporary LSB Storing Program Scheme for Sub-40nm MLC NAND Flash Memories and beyond," IEEE Symposium on VLSI Circuits, pp. 188–189, June 2007.
57. N. Shibata et al., "A 70nm 16Gb 16-level-cell NAND Flash Memory," IEEE Symposium on VLSI Circuits,pp. 190–191, June 2007.
58. K. Yano et al., "Single-electron memory for giga-to-tera bit storage," Proceedings of the IEEE, vol. 87, no. 4, pp. 633–651, Apr. 1999.
59. G. Molas et al., "Impact of few electron phenomena on floating-gate memory reliability," in IEEE International Electron Devices Meeting, 2004. IEDM Technical Digest, pp. 877–880, Dec. 2004.
60. G. Molas et al., "Degradation of floating-gate memory reliability by few electron phenomena," IEEE Transactions on Electron Devices, vol. 53, no. 10, pp. 2610–2619, Oct. 2006.
61. C. Compagnoni et al., "First evidence for injection statistics accuracy limitations in NAND Flash constant-current Fowler-Nordheim programming," in IEEE International Electron Devices Meeting, 2007. IEDM '07, pp. 165–168, Dec. 2007.
62. C. Compagnoni et al., "Analytical model for the electron-injection statistics during programming of nanoscale NAND Flash memories," IEEE Transactions on Electron Devices, vol. 55, no. 11, pp. 3192–3199, Nov. 2008.
63. C. Friederich, J. Hayek, A. Kux, T. Müller, N. Chan, G. Köbernik, M. Specht, D. Richter, and D. Schmitt-Landsiedel, "Novel model for cell – system interaction (MCSI) in NAND Flash," in International Electron Devices Meeting, Technical Digest. IEDM 08, Dec. 2008.
64. V. Sobolev, Encyclopaedia of Mathematics. Springer, 2002, ch. Convolution of functions.
65. J. Van Houdt, "High-k materials for nonvolatile memory applications," in Reliability Physics Symposium, 2005. Proceedings. 43rd Annual. 2005 IEEE International, pp. 234–239, 17–21, 2005.
66. K. Takeuchi, T. Tanaka, and T. Tanzawa, "A Multi-page Cell Architecture for High-speed Programming Multi-level NAND Flash Memories," VLSI Circuits, 1997. Symposium on Digest of Technical Papers, pp. 67–68, June 1997.

Petitioner Micron Ex-1021, 0097

4 Reliability issues of NAND Flash memories

C. Zambelli[1], *A. Chimenton*[2] *and P. Olivo*[3]

4.1 Introduction

The continuous demand for NAND Flash memories with higher performance and storage capabilities pushes the manufactures towards the limits of present technologies and to explore new solutions, both from the physical and the architectural point of view.

The memory reliability represents one of the major antagonist towards this unstoppable technological evolution, since the correct operations must be assured not only when a new product is presented, but they must be demonstrated for the entire life cycle. In particular, the minimum number of write operations and the ability of keeping unaltered the stored information for years and years must be guaranteed.

In this chapter the principal reliability mechanisms affecting the traditional floating-gate NAND Flash memories will be addressed: the physical aspects caused by charge transport and trapping in thin insulator layers as well as the incorrect behaviors related to the array architecture. It will also shown as these effects increase dramatically their impact when Multi Level Cells (MLC) are considered. New emerging mechanisms, such as Gate-Induce Drain Leakage (GIDL), Random Telegraph Noise (RTN) and temperature instabilities will also be addressed.

4.2 Basic concepts

The concept element of traditional NAND Flash memory cell is a metal oxide semiconductor device with a floating gate electrically isolated by means of a tunnel oxide and of an interpoly oxide as sketched in Fig. 4.1 [1]. The former oxide plays a basic role for the control of the device threshold voltage whose value represents, from a physical point of view, the stored information. Electrons transferred into the floating gate give a threshold voltage variation $\Delta V_t = -Q/C_{pp}$. In quiescent conditions, thanks to the two oxides, the charge stored does not leak

[1] Dip. Ing., Università degli Studi di Ferrara, cristian.zambelli@unife.it
[2] Dip. Ing., Università degli Studi di Ferrara, andrea.chimenton@unife.it
[3] Dip. Ing., Università degli Studi di Ferrara, piero.olivo@unife.it

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_4,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0098

away, thus granting the nonvolatile paradigm fulfillment. Oxides are available in different material depending on the BEOL process. The common materials are: pure silicon dioxide (SiO_2) for tunnel oxides and a stack of oxide–nitride–oxide (SiO_2–Si_3N_4–SiO_2) for interpoly oxides.

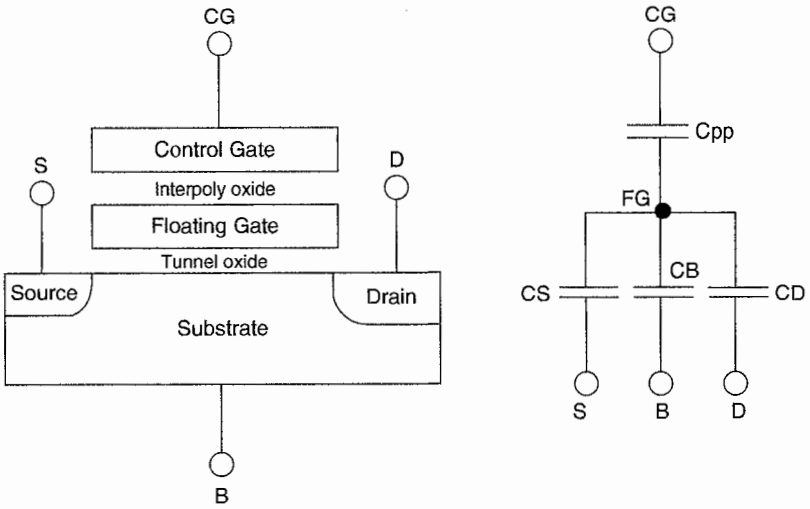


Fig. 4.1. FLOTOX device and its equivalent capacitance model

In the last decade attention was paid also to the so-called *charge trapping* NAND memories like SONOS (silicon–oxide–nitride–oxide–silicon) and TANOS (Tantalum–aluminum–nitride–oxide–silicon) as a cost-effective alternative to the traditional Flotox cell architecture. Their operation principles and physics are described on this book.

The cells are rearranged into an array organization [2] in which we can define three basic subdimensions: string, page and block. The definition of the characteristics of each dimension are well depicted on the previous chapters of this book.

The physical mechanism used for both injecting and extracting electrons to/from the floating gate is the Fowler–Nordheim (FN) tunneling [3]. High electrical field applied to the tunnel oxide ($E_{OX} \approx 10$ MV/cm) allows for electron transfer across the thin insulator to the floating gate. In NAND architectures the electronic tunneling involves the MOS channel/substrate and requires appropriate biasing of control gate and bulk terminals (see Fig. 4.2), while drain and source are left floating.

With respect to the Channel Hot Electron mechanism exploited for cell programming in NOR architectures, FN tunneling requires higher voltages, therefore much more complex charge pumps and higher programming times. In addition, the use of the same mechanism for both programming and erasing the cells exposes the tunnel oxide to a larger degradation, with possible reliability effects. These drawbacks, however, are compensated by the much lower current (orders of magnitudes) required by the writing operations significantly improving power consumption and/or programming parallelism.

Petitioner Micron Ex-1021, 0099

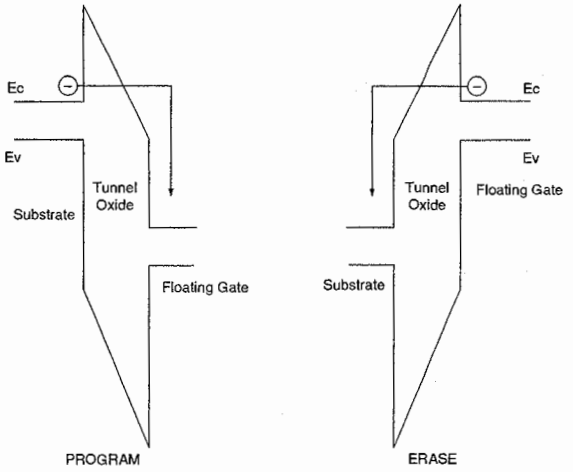


Fig. 4.2. Band diagram sketch of tunneling effect

4.3 Basic reliability effects related to tunnel oxides

During its lifetime a NAND Flash module undergoes a large number of Program/Erase cycles. Every cycle involves very high electrical fields applied to the tunnel oxide. The reliability of the entire memory requires that the tunnel oxide is able to operate correctly under stress conditions. It is obvious that huge efforts are to be spent to determine the right process for the tunnel oxide creation (in terms of thickness, material, growth, defectivity, interface, ...) in order to achieve a successful and reliable NAND technology.

In this section we will analyze the basic physical mechanisms related to the tunnel oxide and affecting both memory endurance and data retention. The thin tunnel oxide may be also responsible for other effects, such as over-programming, possibly producing read errors.

4.3.1 Endurance and intrinsic oxide degradation

The endurance of a memory module is defined as the minimum number of Program/Erase cycles that the module can withstand before leading to a failure. The erased and programmed distributions must be suitably separated, in order to correctly read the logical state of a cell. The difference between EV (Erase reference Value) and PV (Program reference Value) is defined as the "read margin window". However, keeping a correct read margin is not sufficient to guarantee a correct read operation: if during its lifetime the threshold voltage of an erased cell exceeds the EV limit and approaches 0 V, the current flowing through the cell may be not high enough to be identified as "erased" by the reading circuitry, thus producing a read error. Similarly, a programmed cell could be read as "erased" if

Petitioner Micron Ex-1021, 0100

its threshold voltage becomes lower than PV and approaches 0 V. As for the programmed distribution, it is also important that the upper threshold limit does not increase significantly with time, since a too high threshold can block the current flowing through the strings during reading operations.

FN tunneling leads intrinsically to oxide degradation [4]. As a result of consecutive electron tunneling, traps are generated into the oxide [5]. When filled by electrons, charged traps can increase the potential barrier thus reducing the tunneling current, as shown in Fig. 4.3.

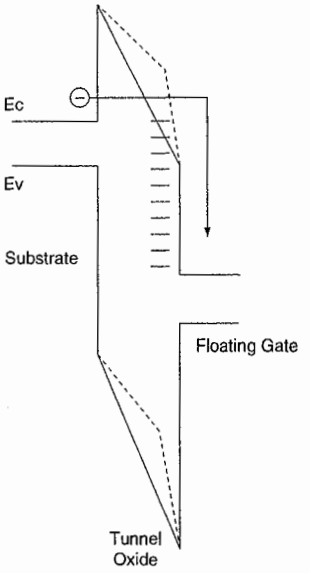


Fig. 4.3. Band diagram during a program operation: without traps (solid lines in the oxide region) and with traps (dashed lines in the oxide regions)

Since the programming and erasing pulses feature constant amplitude and duration, less charge is transferred to and from the floating gate causing an efficiency reduction of both the program and erase operations. A narrowing of the read margin window is then expected.

The charge trapped inside the oxide also produces a threshold shift ΔV_T directly proportional to its amount. The V_T shift is symmetrical and increases the threshold voltage of both erased and programmed cells (see Fig. 4.4).

Writing waveform optimization can help in limiting the trapped charge. For example, it has been shown that the window closure can be reduced by using low voltage erasing pulses able to remove the charge accumulated in the oxide [6].

As shown in Fig. 4.4, the threshold voltage shifts increase with the number of program/erase operations until an endurance failure occurs. Threshold voltage shifts could be recovered by applying specific procedures, that however result not convenient when comparing their effects with the required architectural overheads and time consumption.

Petitioner Micron Ex-1021, 0101

As evidenced in Fig. 4.4, the most critical effect is the increase towards 0 V of the erased threshold. To check whether all cells of a sector have been correctly erased (all threshold voltage must be below 0 V), an erase verify procedure is applied after any erase operation. It consists in a particular read operation performed by driving simultaneously all the word lines of the sector at 0 V: if the read current in a bitline is 0, it means that at least a cell blocked the current flow because its threshold voltage was higher than 0 V. The entire block is marked as *bad block* by the memory controller and no longer addressed [2].

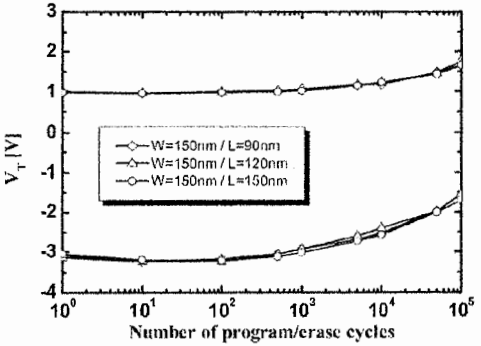


Fig. 4.4. Threshold voltage degradation during cycling of NAND Flash with different geometrical features [7]

4.3.2 Hot Hole Injection oxide degradation

The HHI (Hot Hole Injection) or AHI (Anode Hole Injection) is a breakdown process responsible of tunnel oxide degradation in Flash memories and other side effects such as erratic bits and temperature instabilities enhancement. Experiments have evidenced that HHI occurs during the high-field stress and allows the creation of interfacial states at the substrate–oxide interface. The presence of these states is responsible for a sensible degradation of the transfer characteristic of the floating gate devices by lowering its transconductance g_m, thus further increasing the threshold voltages level [2].

HHI can be straightforwardly explained by analyzing the band diagram of Fig. 4.5. For the program operation case an incident tunneling electron arriving at the Poly-Si anode with energy w_e^{in} from the top of silicon conduction band (CB) creates hole in the valence band (VB) and electron in the CB by direct ionizing collision in Poly-Si. The excess energy of the primary electron (i.e. the injected one) is assumed to be equally shared between the generated electron–hole pair. The generated hole in the Poly-Si VB with an energy w_h is now back injected into the oxide by tunneling through the oxide/anode interface barrier of energy Φ_a^h, as w_h is less than the barrier height Φ_a^h.

Petitioner Micron Ex-1021, 0102

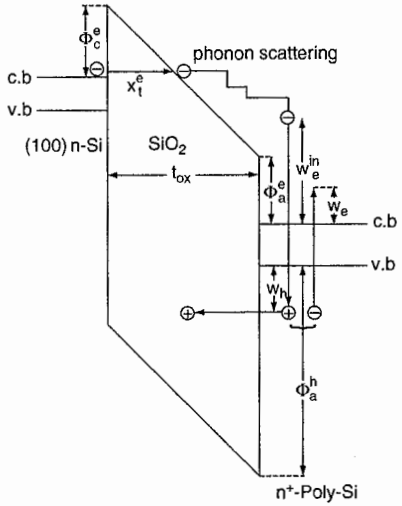


Fig. 4.5. Band diagram of HHI (Hot Hole Injection) phenomenon on a NAND Flash memory structure. The substrate is Si with orientation <100> and the floating gate is n+ doped Poly-Si [8]

During high-field electron transport, holes in the anode are generated when a significant fraction of the injected electrons lose their kinetic energy either by emitting surface plasmons or by directly exciting (via phonon scattering) the anode electrons via interband transition in the case of Poly-Si (majority of Flash memory modules) and intraband transition in the case of metal such as aluminum [8, 9].

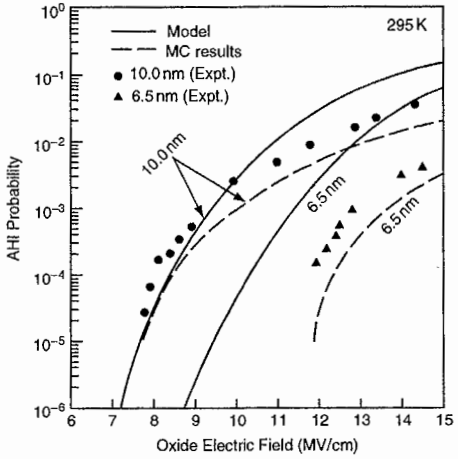


Fig. 4.6. AHI probability calculated with triangular hole-energy barrier approximation as a function of uniform oxide electric field with oxide thickness as a parameter [8]

Petitioner Micron Ex-1021, 0103

According to this physical model, the HHI probability α_h of a tunneling electron can be calculated as:

$$\alpha_h = \sum P_h \theta_h \tag{4.1}$$

where P_h is the probability of hole generation via Impact Ionization in the anode material, and θ_h is the injection probability of the generated hole from the anode VB. Figure 4.6 shows in particular the HHI probability calculated with the triangular hole energy barrier approximation as a function of applied oxide electric field [8].

4.3.3 Data retention

The retention concept is the ability of a memory to keep a stored information over time with no biases applied. Electron after electron, charge loss could slowly leading up to a read failure: a programmed cell can be read as erased if its V_T shifts below 0 V.

The intrinsic retention is mainly limited by tunneling (see Fig. 4.7) through the oxide even if thermionic emission mechanisms over the barrier can be considered. Recent studies [10] demonstrated that an oxide thickness of 4.5 nm is enough for granting theoretical intrinsic retention of 10 years, which is the minimum limit imposed by present standards.

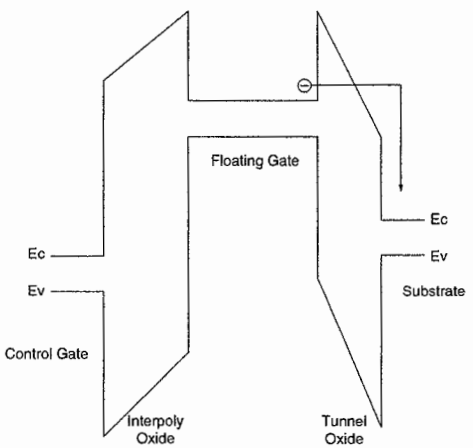


Fig. 4.7. Band diagram of the cell when programmed and not biased. The main mechanism for data loss is tunneling through the tunnel oxide

The cell retention worsen with memory cycling and this effect is appreciable as a reduction of the V_T levels as sketched in Fig. 4.8 showing the time evolution of the cumulative V_T distribution of programmed cells. Charge loss from the floating gate moves the V_T distribution towards lower values.

Petitioner Micron Ex-1021, 0104

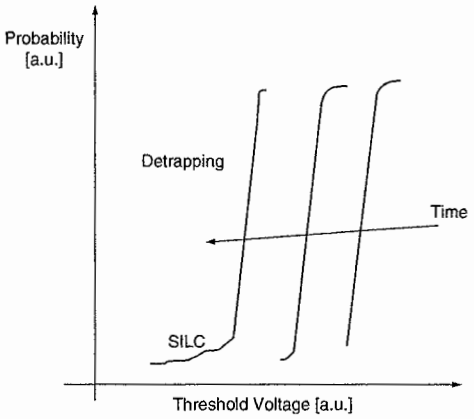


Fig. 4.8. Cumulative distribution of a NAND array on Program state. Both detrapping and SILC effects are appreciable on the time evolution

In additions, a tail in the lower part of the distribution indicates that a small percentage of cells is losing charge faster than average.

The rigid shift of the cumulative V_T distribution can be related to the oxide degradation within the oxide and at the Si–SiO_2 interface. As described in Sect. 4.3.1, successive electron tunneling leads intrinsically to oxide degradation, characterized by traps generation. These traps may be responsible for charge loss from the floating gate towards the silicon substrate. In fact, an empty trap suitably positioned within the oxide can activate trap assisted tunneling (TAT) mechanisms characterized by a significantly higher tunnel probability with respect to a triangular barrier unmodified by the trap presence. In addition, an electron trapped within the oxide during writing operations and responsible for the V_T increase leading up to endurance failures may be detrapped when the program pulse is switched off, when the cell is read or even when the cell is not addressed. As a result, the empty trap may enhance the TAT phenomenon (assuming a positive charged trap) and, in addition, it can increase the electron field at the floating gate-tunneling oxide interface thus raising the electron tunnel probability. The required activation energy for detrapping has been calculated in several experiments about 1.1 eV [11].

It is clear that these mechanisms are strongly related to the oxide degradation and therefore data retention decreases with the number of applied writing pulses.

Results of retention stresses showed that the cells contributing to the tails of the V_T distribution are characterized by a leakage current larger than the average at the same stress level [12].

The Stress Induced Leakage Current (SILC) of these cells is attributed to TAT process of carriers through the tunnel oxide traps. By modeling the behavior of the tail cells it has been shown that a single TAT model is not consistent with the observed leakage current. The charge loss of these cells is attributed to a tunneling process assisted by two traps (2TAT) – see Fig. 4.9.

Petitioner Micron Ex-1021, 0105

The characteristics of this phenomenon are the opposite with respect to detrapping: low activation energy (0.1–0.3 eV) and strong field acceleration.

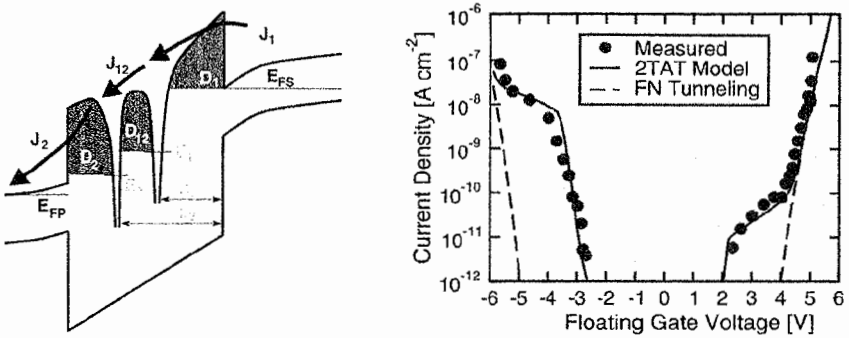


Fig. 4.9. Two traps assisted tunneling (2TAT) band diagram (left) and comparison between the SILC-2TAT model result versus the classical Fowler-Nordheim theory (right) [14]

SILC affects a larger number of cells after each writing cycle, thus confirming that also this abnormal charge loss is driven by oxide degradation. The position of leaky cells within the array, however, does not show any clusterization which could be related to a technological process. In addition, the abnormal leakage is not permanent: a cell can exhibit a SILC effect after a cycle and behave as normal when erased and reprogrammed. This result is consistent with a model requiring two traps with suitable locations within the oxide to activate a SILC effect. Some of these cells can suffer an erratic behavior [13] due to a trap annihilation/ reactivation process. It has been proved, however, that traps are annealed over 85°C so high temperatures partially mitigate this effect [14].

4.3.4 Overprogramming

As stated on the previous chapters of this book, it is important that the maximum programmed threshold voltage doesn't exceed the OP (Over Programming) limit, since a too high threshold could inhibit the correct behavior of pass transistor of an unread cell, thus blocking the current flowing through the string. Besides the intrinsic oxide wear out, a more subtle effect can move, during programming, the threshold voltage above the OP limit. This effect, related to the so-called fast moving bits and to erratic bits [15, 16], has been widely studied in NOR architectures, in which the accurate control of the erase distribution plays a fundamental role. In NOR architectures over-erase is caused by some cells exhibiting an enhancement of the tunneling current and it has been related to Anode Hole Injection and the subsequent formation of positive charge clusters in the oxides (see Fig. 4.10).

The presence of erratic behavior (that is a transient behavior in which few cells exhibit, during their normal operation, a larger tunneling current) has been observed also when FN tunneling is used for cell programming [15]. Such an

Petitioner Micron Ex-1021, 0106

effect, however, may be controlled by tailoring opportunely the pulse amplitudes and durations during programming and the V_{pass} voltage used in read operation.

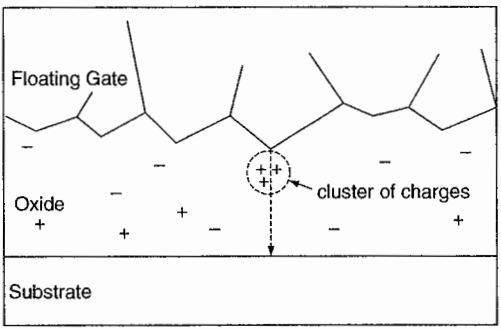


Fig. 4.10. Charge cluster model used for explanation of fast-programming bits [15]

4.3.5 General comments concerning oxide degradation

MLC architectures are more prone to reliability effects related to oxide degradation with respect to SLC (Single Level Cells) memories. In the MLC approach, the separation between two adjacent threshold levels is a fraction of the read margin typical of a SLC architecture. Therefore, even a slight V_T variation may lead to a logical error. The charge variation within the floating gate can occur when the memory is not biased as well as during the program/erase phase (because of the creation of traps within the oxide able to modify the tunneling barrier or because of erratic phenomena inducing overprogramming). As an consequence, while a SLC architecture can usually withstand 100 k program-erase cycles, the typical cycle number for a MLC memory is 10 k [2].

The impact of the reliability effects related to the oxide degradation can be significantly reduced by using appropriate management policies.

For instance, it is important to distribute the writing stress over the entire population of cells rather than on a single hot spot, thus avoiding that some blocks are updated continuously while the others keep unaltered their charge content. It is clear that blocks whose information is updated frequently are stressed with a large number of program-erase cycles. In order to keep the aging effects as uniform as possible, the number of both read and write cycles of each block must be monitored and stored by the memory controller. *Wear Leveling* techniques [2, 17] are based on a logical to physical translation for each sector. When the host application requires an update on the same logical sector, the memory controller dynamically maps the data on a different physical sector. The out-of-date copy of the sector is tagged as both invalid and eligible for erase. In this way all the physical sectors are evenly used, thus reducing the overall oxide aging.

To reduce possible errors caused by oxide aging and erraticity, Error Correcting Codes (ECC) are widely used in NAND memories and in particular in MLC architectures [18].

Petitioner Micron Ex-1021, 0107

4.4 Disturbs related to memory architecture

Architectural solutions for memory operations may also affect the overall reliability, by producing errors and even cell failures. The most common effects are the so called "disturbs", that can be interpreted as the influence of an operation performed on a cell (Read or Program) on the charge content of a different cell.

Read disturbs are the most frequent source of disturbs in NAND architectures. This kind of disturb may occur when reading many times the same cell without any erase operation. All the cells belonging to the same string of the cell to be read must be driven in a ON state, independently of their stored charge. The relatively high V_{pass} bias applied on the control gate and the sequence of V_{pass} pulses applied during successive read operation may trigger SILC effects in some cells that, therefore, may gain charge. These cells suffer a positive shift of their threshold voltage that may lead to read errors (when addressed). Since the SILC effect is not symmetrical, the cells that may be affected by SILC effects induced by read disturbs are not necessarily the same that may exhibit data retention problems. Figure 4.11a shows the typical read disturb configuration.

The probability of suffering a read disturb increases with the cycle number (i.e. towards the end of the memory useful lifetime) and it is higher in damaged cells. Read disturbs do not provoke permanent oxide damages: if erased and then reprogrammed, the correct charge content will be present within the floating gate.

Two other important typologies of disturbs are related to the program operation: the *Pass disturbs* and the *Program disturbs,* which are shown respectively on Fig. 4.11b and c. The former are similar to the read disturbs and affects cells belonging to the same string of a cell to be programmed. With respect to the read disturb, the Pass one is characterized by the higher V_{pass} voltage applied to cells that are not to be programmed (thus enhancing the electric field applied to the tunnel oxides and the probability of undesired charge transfer). On the other side, the pass disturb may be provoked, in the worst case, by a program operation on all the string cells but the one affected by the disturb (when a string has been fully programmed, an erase operation must be necessarily performed before any other reprogram): therefore the disturb duration is much shorter and the cumulative effect of successive read pulses encountered in read disturbs is not present.

The Program disturbs, on the contrary, affect cells that are not to be programmed and belong to the same wordline of those that are to be programmed. In that case the program disturb is strongly related to the voltages and pulse sequences used for the self-boosting techniques. The criticality of an effective program operation limiting program disturbs and/or possible successive errors is attested by the fact that in NAND memories the program operation should follow a precise and well defined "hierarchy": it is necessary to start from the cell nearest to the source selector and proceed along the string up to the cell nearest to the drain selector. This procedure is important, because the threshold voltage of a cell depends on the state of the cells placed between the considered cell and the source contact (the *background pattern dependency* phenomenon); the series resistance of the cells is different if they are programmed or erased.

Petitioner Micron Ex-1021, 0108

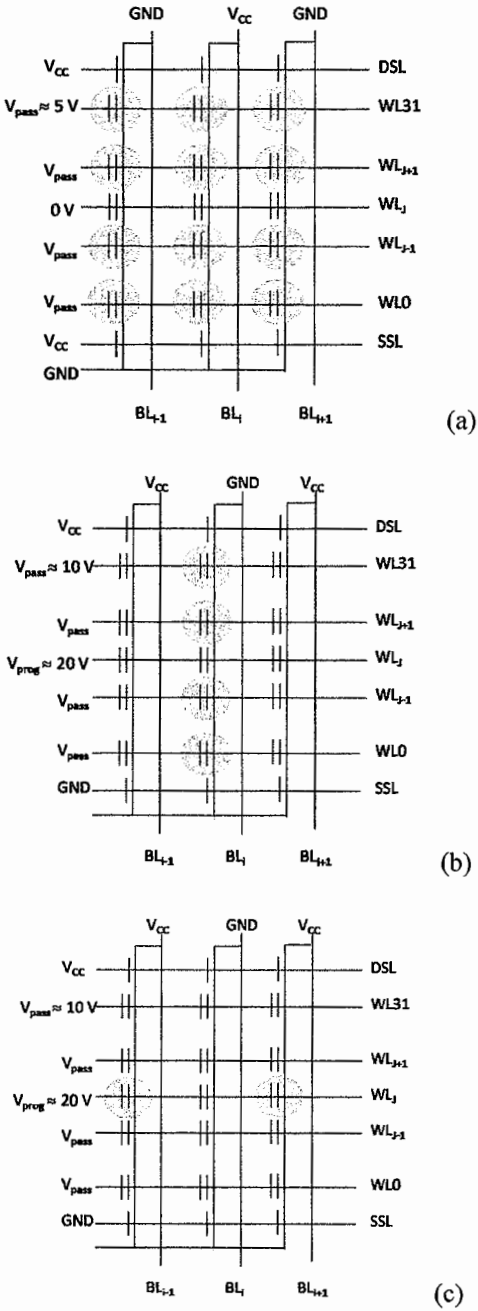


Fig. 4.11. Representation of read disturb (a), pass disturb (b) and program disturb (c) in a NAND Flash array. The cells potentially affected by the disturbs are marked in gray

Petitioner Micron Ex-1021, 0109

If this procedure is not followed, the threshold voltage of the cell may be different in the read phase, with respect to the verify phase.

4.5 Emerging reliability threats

As the technology scaling of NAND Flash proceeds, emerging reliability threats have to be considered in addition to the traditional issues presented in the previous sections. The influence of phenomena like the Gate-Induced Drain Leakage and the Random Telegraph Noise cannot be neglected during the validation phase of a technology since they introduce new disturb types during the common operations of a NAND module. This section will introduce a picture of the physical origins of these issues and describe their effects on a NAND Flash array.

4.5.1 Gate Induced Drain Leakage

The *Gate Induced Drain Leakage* (GIDL) is a major leakage mechanism occurring in OFF MOS transistors when high voltages are applied to the drain and it is attributed to tunneling taking place in the deep-depleted or even inverted region underneath the gate oxide [19]. When the gate is biased to form an accumulation layer at the silicon surface, the surface behave like a *p* region more heavily doped than the substrate. This effect causes the depletion layer of the junction at the surface to be more narrow than elsewhere. The narrowing of the depletion layer near the surface increases the local electric field thus enhancing high field effects near that region. When the gate voltage in a nMOS transistor is 0 V (or below) and the drain is biased at a high voltage, the n^+ drain region under the gate can be depleted or even inverted (see Fig. 4.12). This effect causes a peak field increase leading up to high field phenomena such as avalanche multiplication and band-to-band tunneling. The band-to-band tunneling probability can also be increased by the presence of surface traps, resulting in a band-trap-band tunneling.

As a result of these effects, minority carrier are emitted in the drain region underneath the gate and swept laterally to the substrate which is at a lower potential, thus completing a path for the leakage current.

Figure 4.13 shows the sub-threshold characteristics in an nMOS transistor for different drain voltages [20].

On NAND arrays the GIDL has been found to produce erroneous programming in specific cells [24], that is those belonging to WL0, adjacent to the SSL transistor. The SSL transistor is OFF during programming and GIDL effects may be present if its drain is driven at high voltages. This situation occurs because of the self-boosting techniques adopted to prevent programming. As shown in Fig. 4.14, if WL0 is driven at V_{pgm} to program the cell in the central column, the channel voltage of the cells sharing the WL0 line are to be raised to prevent their programming.

Petitioner Micron Ex-1021, 0110

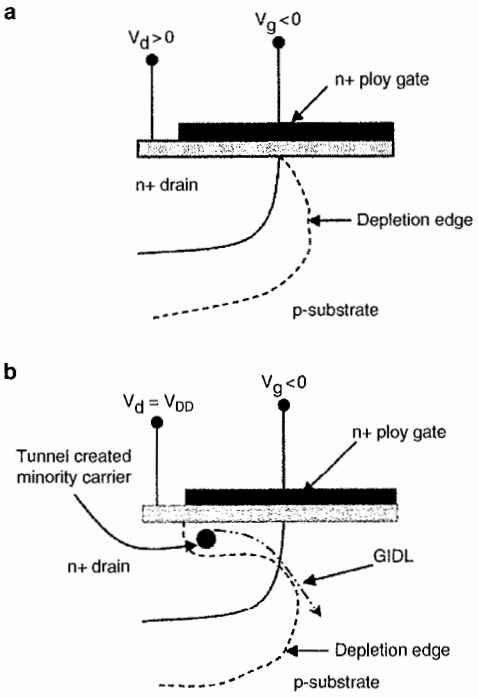


Fig. 4.12. Condition for the depletion region near the gate-drain overlap region of a nMOS transistor when the surface is accumulated with a low negative gate bias (a), and n+ region is depleted or inverted with high negative gate bias (b) [19]

Because of the self-boosting technique, the source terminal of those cells (therefore also the drain node of their adjacent SSL transistors) are raised to values higher than V_{CC}, thus leading up to bias configurations activating GIDL effects. The electron–holes pair generation follows and the generated electrons are accelerated at the SSL – WL0 space region and can be injected as hot electrons in the floating gate of WL0 cells.

Also the DSL transistors may trigger GIDL mechanisms when boosting effects take place on cells sharing WL31, but since the voltage applied to the DSL gate line is V_{CC}, the consequences suffered by cells belonging to WL31 are much lower with respect to those sharing WL0. Figure 4.15 shows experimental data for three different cells: WL0 and WL31, potentially affected by GIDL disturbs, and a reference one located in the middle of the string [21].

This unwanted programming in specific cells is supposed to burden in scaled architectures, since a reduced separation between WL0 and SSL lines will increase the accelerating fields for hot electrons.

Recently, to mitigate GIDL effects, it has been proposed to introduce two dummy world lines to separate the two select transistors and the effective string of cells [22]. To reduce the impact of these two additional word lines, longer strings of 64 cells have been proposed to improve area efficiency.

Petitioner Micron Ex-1021, 0111

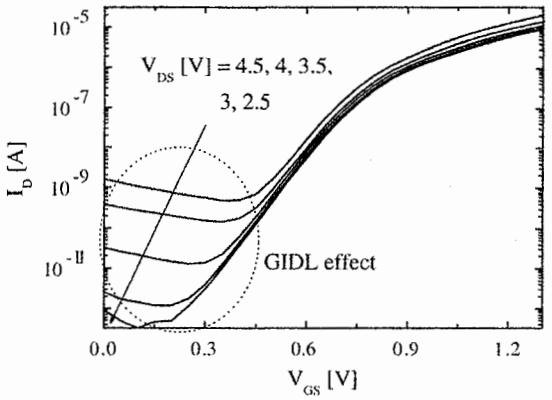


Fig. 4.13. Leakage current measured in a nMOS transistor for different drain–source voltages V_{DS} in the subthreshold regime. When V_{GS} approaches 0 V, for high V_{DS} values the leakage current is due to GIDL [20]

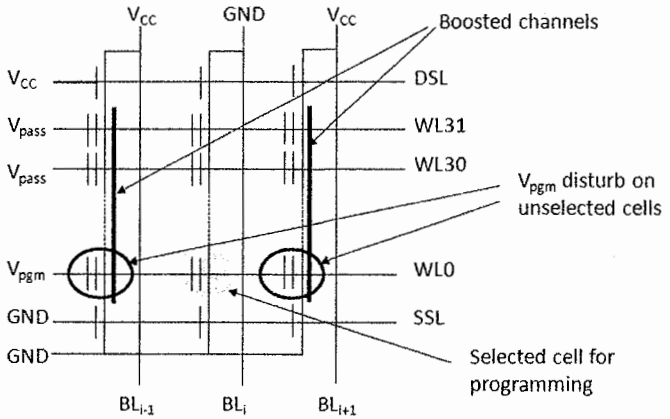


Fig. 4.14. Bias conditions possibly activating GIDL effects on SSL transistors belonging to columns BL_{i-1} and BL_{i+1}

A higher number of cells in series (64 cells vs 32 of standard architectures), however, increase the string resistance so that word-line voltage modulation are required during read and programming thus ensuring that proper voltage levels are applied depending on the cell location within the string. For instance, a higher word-line voltage is used when accessing a cell near the top of the string (close to the bit line) to compensate for the string resistance.

Petitioner Micron Ex-1021, 0112

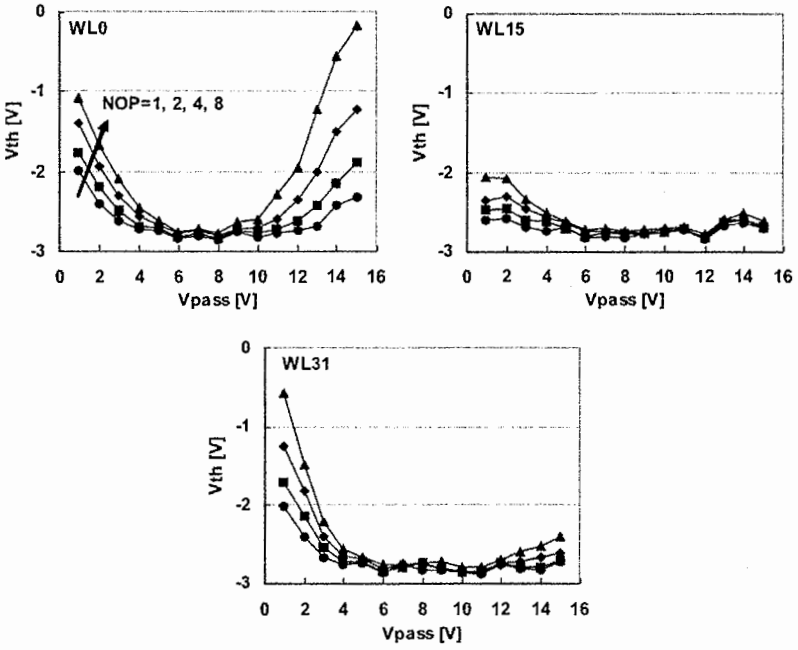


Fig. 4.15. Measured program disturbs characteristics for 3 cells belonging to WL0, WL15 and WL31, respectively [21]. NOP indicates the number of partial programming when multiple writing of a world line is allowed for specific applications. It can be observed that, for higher values of V_{pass}, the disturb on cell WL0 becomes significant

4.5.2 Random Telegraph Noise

The Random Telegraph Noise (RTN) phenomenon has been observed in semiconductor devices like bipolar transistors and junction field effect transistors since the 1950s (with others fashioned names like burst noise or popcorn noise). In MOSFET devices its study has started about 30 years later. The physical concept of RTN is attributed to a mechanism of capture/emission of single electrons by interfacial traps with time constants τ_c and τ_e, resulting in fluctuations of the transistor drain current and therefore on its threshold voltage. RTN can be analyzed both in time and frequency domain (see. Fig. 4.16): the former analysis depict the *quasi-binariety* of the phenomenon remarking an exponential distribution of the capture/emission time constants, the latter enables the definition of a power spectral density law related to τ_c and τ_e [23].

The amplitude of the RTN is often calculated as a ratio between the drain current and its variation and typical values ranges from 0.1% and 1%. From the equation of the *MOSFET* in strong inversion regime the noise amplitude is derivable as:

Petitioner Micron Ex-1021, 0113

$$I_D \propto \mu_n Q_n \Rightarrow \frac{\Delta I_D}{I_D} = \frac{\Delta Q_n}{Q_n} + \frac{\Delta \mu_n}{\mu_n}$$
$$\frac{\Delta I_D}{I_D} = \frac{q}{C_{OX}(V_G - V_T)WL} + \frac{\Delta \mu_n}{\mu_n}\frac{\Delta A}{WL} \tag{4.2}$$

where ΔA represents the area over which the mobility is strongly affected by scattering. When large fluctuations of the drain current occur (i.e. more than 20% of the nominal value) the RTN phenomena becomes *anomalous*. On ultranarrow devices (i.e. W < 40 nm) the *quasi-monodimensional* channel enhances the scattering and the multiple defect interaction, thus leading up RTN ratio values about 60%.

On floating gate cells as those used in NAND, V_T is usually quoted rather than the drain current. Due to the floating gate coupling coefficient α_G, the threshold voltage fluctuation becomes (neglecting the mobility fluctuations):

$$\Delta V_T = \frac{q}{C_{OX} WL \alpha_G} \tag{4.3}$$

which gives a typical value of 25 mV using a 45 nm technology with a tunnel oxide of 7 nm and a gate coupling ratio of 0.65.

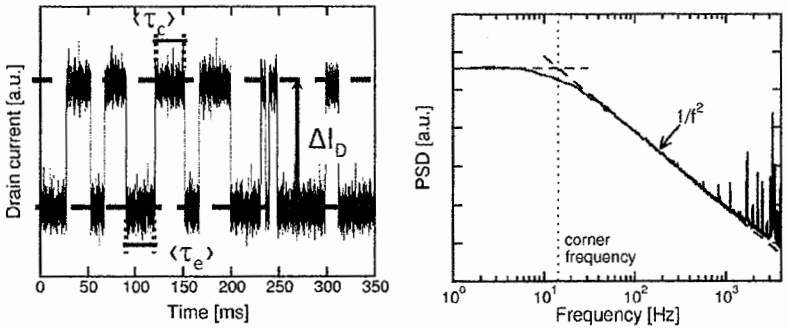


Fig. 4.16. Representation of the time domain plot (left) and power spectral density (right) of the RTN of a MOSFET device [14]

RTN has been recently studied in NOR and NAND architectures [24, 25]. Figure 4.17 shows the cumulative distributions F and 1 – F in a NAND array, where ΔV_T has been calculated as the threshold voltage difference between two consecutive read operation [26].

It can be observed a ΔV_T drift as far as the number of consecutive reading operation is increased. This RTN drift has been attributed to the presence of slow trap states within the oxide.

Petitioner Micron Ex-1021, 0114

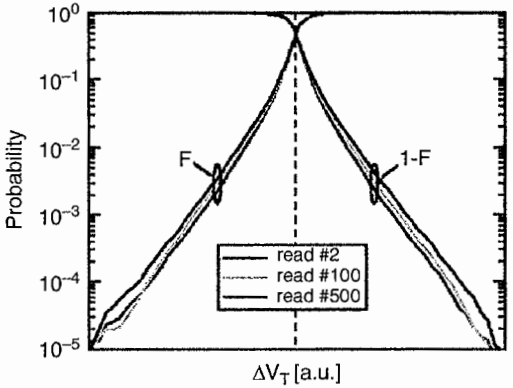


Fig. 4.17. Drift issue of the random telegraph noise on a read cycled NAND device [14]

The dependence of RTN noise in Flash memories on oxide traps has also been evidenced by the influence of memory ageing on ΔV_T cumulative distributions: when increasing the number of Program/erase cycles, new traps are created within the oxide and, therefore, V_T instabilities are found to be more and more probable, as shown in Fig. 4.18 [26].

The previous results demonstrate that RTN must be considered when designing the V_T levels in multilevel architectures since it can produce read errors, in particular for increasing P/E cycle numbers.

More recently, a new RTN effect has been observed in NOR cells and always related to the presence/creation of cluster of traps within the oxide [27]. It has been demonstrated that erratic writing (erase in NOR, program in NAND architectures) is due to a RTN behavior that is related to the presence/absence of a cluster of positive charge in the tunnel oxide close to the injecting interface which strongly affects the result of the FN operation (see Fig. 4.19).

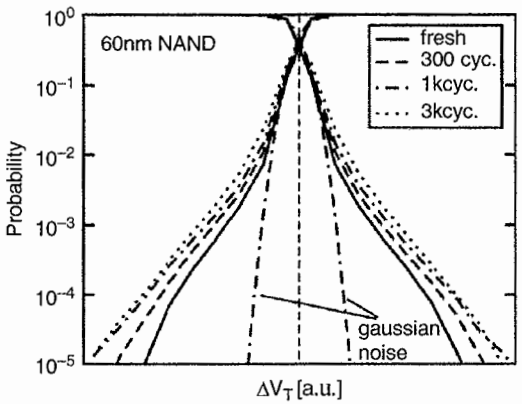


Fig. 4.18. ΔV_T distributions F and 1 – F as a function of the cycle number. The Gaussian noise contribution to the measured ΔV_T is also shown [26]

Petitioner Micron Ex-1021, 0115

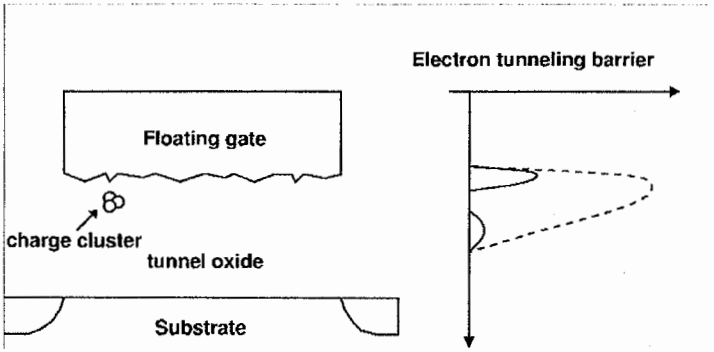


Fig. 4.19. Left: charge cluster model in which a positive charge cluster is located close to the floating gate of a cell in a NOR architecture. Right: electron tunneling barriers of silicon dioxide seen by electrons at the bottom of the floating gate conduction band in presence (solid line) or absence (dashed line) of the positive charge cluster [27]. In a NAND cell the erratic behavior is observed during programming, so that a cluster of traps influences the FN probability when it is close to the cell channel (i.e. the injecting interface)

In particular, when programming a NAND cell, the expected threshold level can be related to the absence of the positive charge, whereas its presence can enhance the tunneling current resulting in a final higher threshold level, possibly leading up to overprogramming issues and/or erroneous programming in MLC architectures.

4.5.3 Charge injection statistics

The aggressive scaling of NAND memories, necessary to improve their performances, brings attention to the discrete nature of the charge stored in the floating gate. The number of electrons determining the stored information, in fact, continuously decreases with the tunneling area dimension. When only few electrons control the cell state (i.e. setting the transistor threshold voltage), their statistical fluctuations determine a non-negligible spread. These fluctuations may be attributed to the statistics ruling the electron injection into the floating gate during program or to the electron emission from the floating gate during erase or retention [28–29], both related to the granular nature of the current flow [30]. A slight variation of the number of electrons injected during programming may produce V_T variations possibly leading to errors in MLC architectures.

Cells in NAND MLC architectures are programmed by using a staircase voltage on the control gate. This kind of waveform allows achieving a constant current Fowler-Nordheim tunneling: by increasing the wordline voltage after each short pulse it is possible to compensate the field reduction that follows the electrons injection into the floating gate (see Fig. 4.20). For sufficiently large step numbers, a linear V_T increase is obtained, with a ΔV_T per step almost equal to the applied voltage step (see Fig. 4.21).

Petitioner Micron Ex-1021, 0116

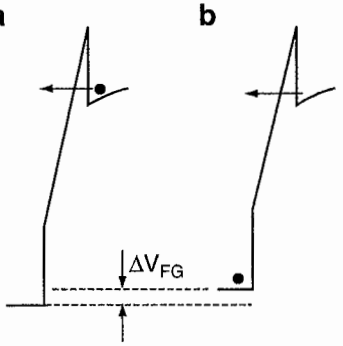


Fig. 4.20. Electrostatic effect on the injection process: the Floating Gate (FG) potential changes when an electron is injected from the substrate (a) to the FG (b), reducing the tunnel oxide field and the tunneling current [30]

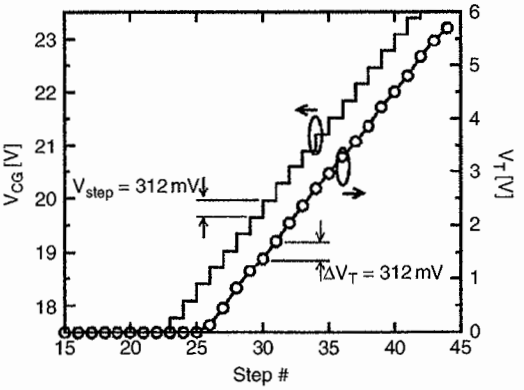


Fig. 4.21. Staircase V_{CG} waveform and corresponding V_T transient in a NAND cell under a constant-current FN program operation [30]. The step duration is 11 μs

This is due to the programming current convergence toward an equilibrium stationary value, corresponding to an average number of electrons transferred to and from the floating gate for each step [31].

The discrete nature of the charge flow introduces, for each cell, a statistical spread contribution to the resulting ΔV_T after each step.

It has been evidenced that the threshold voltage variation spread, indicated as $\sigma_{\Delta Vt}$, depends only on the parameter V_{step} of the programming waveform (i.e. on the injected charge per step qn) and not on the pulse duration and on the number of pulses to achieve ΔV_T. The spread versus the average value of the threshold voltage variation shows a nearly poissonian behavior for low ΔV_T values, while a sub-poissonian statistics clearly appears for larger ΔV_T

Since the evalution of $\sigma_{\Delta Vt}$ on a NAND cells array requires the comparison of two subsequent read of the device V_T (before and after a determined program

Petitioner Micron Ex-1021, 0117

step), the contributions of the RTN add to the injection statistics and potentially could affect the analysis. From the theoretical point of view the convolution of the RTN probability distribution (displaying exponential tails) and the gaussian probability density describing the injection statistics shows that in the lower ends of the ΔV_T distribution nearly exponential RTN tails appear, while the main part (used for $\sigma_{\Delta Vt}$ evaluation) remain unaffected, as shown in Fig. 4.22.

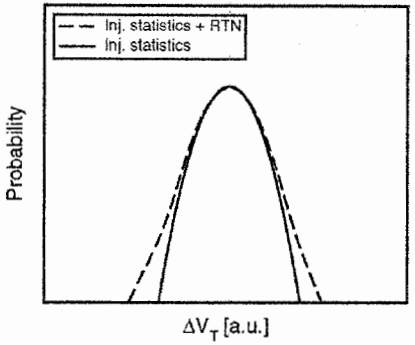


Fig. 4.22. Probability distributions of the injection statistics with and without the contribution of the RTN [30]

Furthermore, experiments shown that $\sigma_{\Delta Vt}$ does not change with cycling, revealing that the ΔV_T statistics is not the result of a change in the charge state of the tunnel oxide, as the increase of the trap density induced by cycling and revealed by the larger RTN instabilities has no effect in the extracted threshold voltage spread (see Fig. 4.23).

On the low ΔV_T regime we can express the spread of the threshold voltage variation in relation of the floating gate to control gate capacitance C_{pp}. Since $\Delta V_T = qn/C_{pp}$ we have:

$$\sigma_{\Delta V_T} = \frac{q}{C_{pp}}\sigma_n \qquad \sigma_n = \sqrt{\bar{n}} \tag{4.4}$$

considering that for small n values the injection events are rare and uncorrelated, hence being ruled by a Poisson statistics. By rewriting the first Eq. (4.4) substituting the second equation result on it and considering that the number of injected electrons is approximativly equal to the threshold voltage variation, it is possible to write the formula of the spread under low variation of the threshold voltage as:

$$\sigma_{\Delta V_T} = \frac{q}{C_{pp}}\sqrt{\bar{n}} = \sqrt{\frac{q}{C_{pp}}\overline{\Delta V_T}} \tag{4.5}$$

Petitioner Micron Ex-1021, 0118

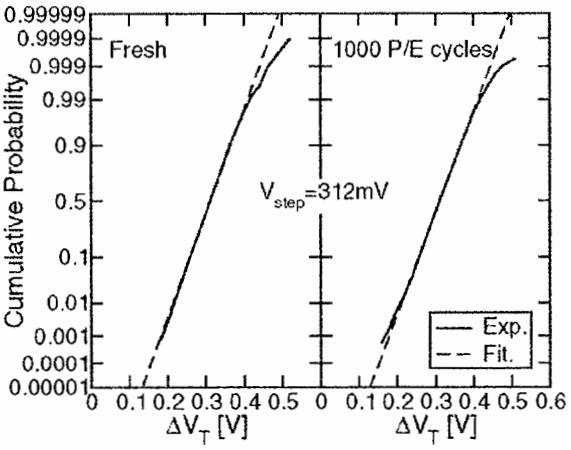


Fig. 4.23. Comparison between two ΔV_T probability distributions. The left one has been extracted on a fresh NAND device, while the right one has been extracted after 1,000 Program/Erase cycles on the same device [30]

4.5.4 Temperature instabilities

The Bias Temperature Instability (BTI) has been one the major concern of CMOS circuit reliability during the last decade, therefore indirectly affecting the NAND Flash memory as well. Since these memories rely principally on the Fowler–Nordheim tunneling effect both for program and erase operations, it is worth to point out the dependency of the tunneling induced oxide stress on the temperature. An explanation of the BTI occurrence in these memories can be accurately evaluated by monitoring the temperature dependant threshold voltage V_T evolution with respect to program/erase cycles and stress time.

Figure 4.24 shows the room temperature program/erase cycle evolutions of the sub-threshold swing (SSW) and the hysteresis voltage V_{HYS} defined as the difference between V_T in the forward and reverse V_{GS} sweeps (respectively representing a read operation with increasing/decreasing read voltage). While the SSW in the programmed cell is nearly independent of the program/erase cycling, that in the erased cell shows a severe roll up as the number of cycles increases. Therefore, the generation of N_{IT} (interface traps) during the program/erase FN stress is observed to be significant only in the erased cell. Basing on the reaction–diffusion (R-D) model of the released hydrogen species h^* in the negative-bias-temperature instability (NBTI) in p-channel MOS field-effect transistors [32], N_{IT} is repassivated by back-diffused hydrogen species to Si/SiO_2 interface, resulting in the immunity of SSW to P/E cycling in the programmed cell as shown in Fig. 4.24.

However, in an erased cell, the generation of N_{IT} due to broken ≡Si–H and ≡Si–O bonds by hot holes injected from Si substrate is followed by the induction of serious damages in both the bottom oxide and the Si/SiO_2 interface as is the case

Petitioner Micron Ex-1021, 0119

in pMOSFETs subjected to the FN hot hole stress. Moreover, a part of injected holes are captured by bulk traps in the oxide during the erase operation [32].

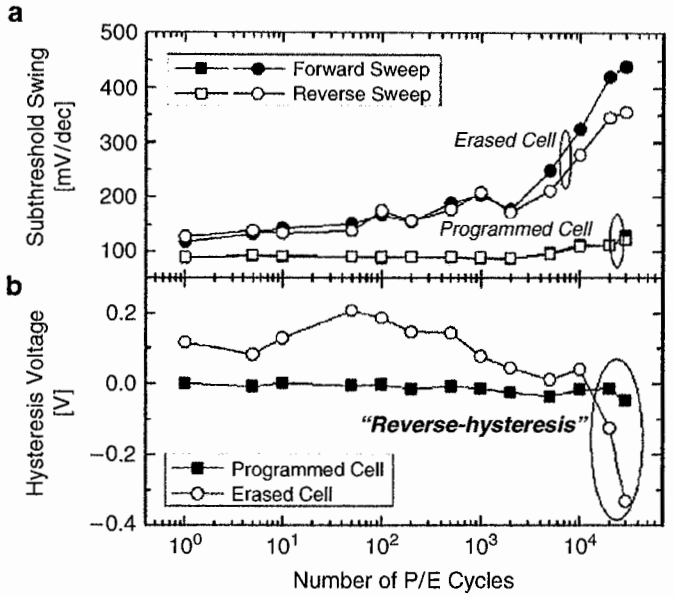


Fig. 4.24. The program/erase cycle evolution of the (a) subthreshold swing (SSW) and the (b) hysteresis characteristics of a measured NAND-type memory cell [33]

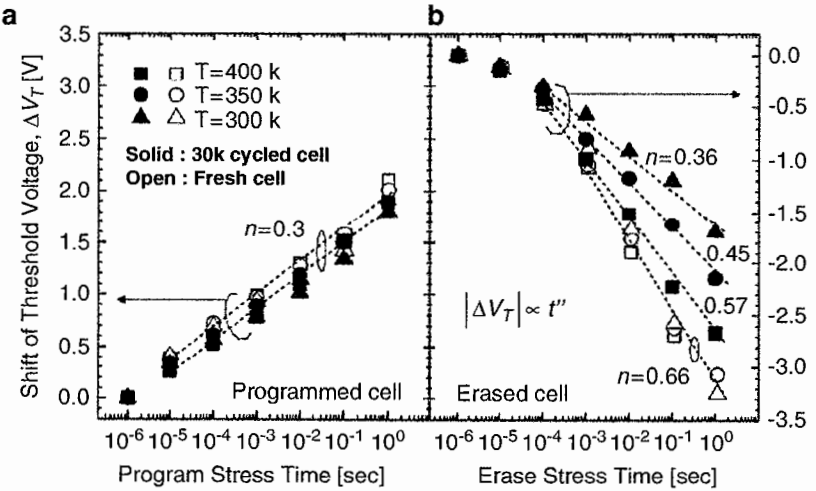


Fig. 4.25. The program/erase cycle Fowler-Nordheim stress time dependence of threshold voltage shift (ΔV_T) in the programmed cell (a) and in the erased cell (b) at various temperatures [33]

Petitioner Micron Ex-1021, 0120

As a result of the observed phenomena, the bias temperature dependence of the FN stress time evolution in the erased cell is similar to the recovery of N_{IT} that takes place in the dynamic NBTI stress time evolution in pMOSFETs.

Analyzing the data of Fig. 4.25 it is possible to extract the power-law rule for the determination of the FN temperature dependence. On programmed cells the exponent n is almost constant while on the erased cells varies significantly with temperature due to AHI-induced holes trapped in the oxide in the erased cell conditions.

References

1. F. Masuoka, M. Momodomi, Y. Iwata and R. Shirota, "New ultra high density EPROM and Flash EPROM cell with NAND structure", *IEEE Tech. Dig.*, pp. 552–555, 1987.
2. P. Cappelletti, C. Golla, P. Olivo and E. Zanoni, Eds., *Flash Memories*. Boston, MA: Kluwer, ch. 5, 1999.
3. M. Lenzlinger and E.H. Snow, "Fowler-Nordheim tunneling into thermally grown SiO_2", *IEDM Tech. Dig.*, vol. 40, pp. 273–283, 1969.
4. Y. B. Park and D. K. Schroeder, "Degradation of thin tunnel gate oxide under constant Fowler-Nordheim current stress for a Flash EEPROM," *IEEE Transactions on Electron Devices*, vol. 45, pp. 1361–1368, 1998.
5. A. Modelli, A. Visconti, and R. Bez, "Advanced Flash memory reliability," *IEEE International Conference on Integrated Circuit Design and Technology*, pp. 211–218, 2004.
6. J. H. Lee et al., "Using erase self-detrapped effect to eliminate the Flash cell program/erase cycling Vth window close," *Proceedings of the IRPS*, pp. 24–29, 1999.
7. J.D. Lee et al., "Degradation of tunnel oxide by FN current stress and its effects on data retention characteristics of 90nm NAND Flash memory",*Proceedings of the IRPS,* pp. 497–501, 2003.
8. S. Piyas, "Calculation of the probability of hole injection from polysilicon gate into silicon dioxide in MOS structures under high-field stress", *Solid-State Electronics*, vol. 43, pp. 1677–1687, 1999.
9. M. V. Fischetti, "Model for the generation of positive charge at the Si-SiO_2 interface based on hot-hole injection from the anode", *Physical Review B*, vol. 31, pp. 2099–2113, 1985.
10. D. Ielmini, A.S. Spinelli and A.L. Lacaita, "Recent developments on Flash memory reliability", *Microelectronic Engineering, 14th biennial Conference on Insulating Films on Semiconductors*, vol. 80, pp. 321–328, 2005.
11. N. Mielke, H. Belgal, I. Kalastirsky, P. Kalavade, A. Kurtz, Qingru Meng, N. Righos, Jie Wu, "Flash EEPROM threshold instabilities due to charge trapping during program/erase cycling", *IEEE Transactions on Device and Materials Reliability*, vol. 4, pp. 335–344, 2004.
12. P. Cappelletti, R. Bez, A. Modelli and A. Visconti, "What we have learned on Flash memory reliability in the last ten years", *IEEE Tech. Dig. IEDM,* 2004.
13. A. Chimenton, P. Pellati, and P. Olivo, "Erratic bits in Flash Memories under Fowler–Nordheim Programming," *Japan Journal of Applied Physics*, vol. 42, pp. 2041–2043, 2003.
14. A.S. Spinelli, *IRPS tutorial*, 2009.

Petitioner Micron Ex-1021, 0121

15. A. Chimenton, P. Pellati, and P. Olivo, "Overerase Phenomena: An Insight Into Flash Memory Reliability," *Proceedings of the IEEE*, vol. 91, pp. 617–626, 2003.
16. A. Chimenton and P. Olivo, "Impact of high tunneling electric fields on erasing instabilities in NOR Flash memories", *IEEE Transactions on Electron Devices*, vol. 53, pp. 97–102, 2006.
17. R. Micheloni, M. Picca, S. Amato, H. Schwalm, M. Scheppler and S. Commodaro, "Non-volatile memories for removable media," *Proceedings of the IEEE*, vol. 97, pp. 148–160, 2009.
18. R. Micheloni, A. Marelli, and R. Ravasio, *Error Correction Codes for Non-Volatile Memories*, Springer-Verlag, 2008.
19. K. Roy, S. Mukhopadhyay and H. Mahmoodi-Meimand, "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits", *Proceedings of the IEEE*, vol. 91, pp. 305–327, 2003.
20. L. Lopez, P. Masson, D. Nèe and R. Bouchakour, "Temperature and drain voltage dependance of gate-induced drain leakage", *Elsevier Microelectronics Engineering*, vol. 72, pp. 101–105, 2004.
21. Lee Jae-Duk et al., "A new programming disturbance phenomenon in NAND Flash memory by source/drain hot-electrons generated by GIDL current", *IEEE NVSMW*, pp. 31–33, 2006.
22. K. Kanda et al., "A 120mm^2 16Gb 4-MLC NAND Flash Memory with 43nm CMOS Technology", *IEEE ISSCC*, pp. 430–625, 2008.
23. F. Rahmoune and D. Bauza, "Si-SiO2 interface trap capture properties", *Microelectronic Engineering*, vol. 59, pp. 115–118, 2001.
24. H. Kurata et al., "Random telegraph signal in Flash memory: Its impact on scaling of multilevel Flash memory beyond the 90-nm node", *IEEE Journal of Solid-State Circuits*, vol. 42, pp. 1362–1369, 2007.
25. G. Servalli et al., "A 65 nm NOR Flash technology with 0.042 μm2 cell size for high performance multilevel application", *IEEE IEDM Tech. Dig.*, pp. 869–872, 2005.
26. C.M. Compagnoni, A.S. Spinelli, S. Beltrami, M. Bonanomi and A. Visconti, "Cycling Effect on the Random Telegraph Noise Instabilities of NOR and NAND Flash Arrays", *IEEE Electron Device Letters*, vol. 29, pp. 941–943, 2008.
27. A. Chimenton, C. Zambelli and P. Olivo, "A statistical model of erratic erase based on an automated random telegraph signal characterization technique", *Proceedings of the IRPS*, pp. 896–901, 2009.
28. K. Yano et al., "Single-electron memory for giga-to-tera bit storage", *Proceedings of the IEEE*, vol. 87, pp. 633–651, 1999.
29. G. Molas et al., "Impact of few electron phenomena on floating-gate memory reliability", *Electron Devices Meeting, 2004. IEEE IEDM Tech. Dig.*, pp. 877–880, 2004.
30. C.M. Compagnoni, R. Gusmeroli, A.S. Spinelli and A. Visconti, "Analytical model for the electron-injection statistics during programming of nanoscale NAND Flash memories", *IEEE Transactions on Electron Devices*, vol. 55, pp. 3192–3199, 2008.
31. A. Chimenton, P. Pellati, and P. Olivo, "Constant charge erasing scheme for flash memories", *IEEE Transactions on Electron Devices*, vol. 49, pp. 613–618, 2002.
32. S. Mahapatra, P. B. Kumar, and M. A. Alam, "Investigation and modeling of interface and bulk trap generation during negative bias temperature instability in p-MOSFETs", *IEEE Transactions on Electron Devices*, vol. 51, pp. 1371–1379, 2004.
33. Seung Hwan Seo et al., "Dynamic bias temperature instability-like behaviors under Fowler–Nordheim program/erase stress in nanoscale silicon-oxide-nitride-oxide-silicon memories", *Applied Physical Letters*, vol. 92, 133508, 2008.

Petitioner Micron Ex-1021, 0122

5 Charge trap NAND technologies

*Alessandro Grossi**

5.1 Introduction

In the last years a big research effort has been spent in the study of new technologies that could represent a possible alternative to conventional floating gate (FG) NAND. In fact even if FG NAND is the dominant technology and there is no advice of reduction in scaling pace, several physical roadblocks seem to limit future scalability (e.g. electrostatic interference among adjacent cells). Charge trap (CT) memories may overcome some of these limitations and represent the best candidate to substitute FG devices for future nodes [1]. Differently from floating gate cells that have a semiconductor as storage element, in CT case electrons are trapped inside a dielectric layer. The different storage material change drastically cell architecture impacting also on physical mechanisms for write operations (both program and erase) and reliability.

In this chapter an overview of most important CT characteristics will be done, describing stack composition, basic functionality and potential issues. Last section is devoted to an introduction to 3D arrays that are the natural evolution of planar CT architectures allowing a more compact organization of the cells.

5.2 Planar charge trap NAND

5.2.1 Stack description

Several options are reported in literature for charge trap memories. The reason of this assortment is caused by the manifold needs that are addressed by the layers that compose the cell (up to eight different layers have been proposed to realize a single CT cell). A more detailed insight about the proposed stacks will be done in a following section, but it could be useful to start the introduction to CT memories reporting a summary of the basic elements that constitute a CT cell. As shown in Fig. 5.1, starting from top to bottom of the stack, it is possible to list:

* Numonyx, R&D Technology Development, alessandro.grossi@numonyx.com

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_5,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0123

- Gate
- Blocking layer
- Storage layer
- Tunnel layer
- Substrate

Typically gate is a multi-layer stack in order to separately tackle the following basic requirements: low sheet resistance, correct work-function and good interface with blocking dielectric.

As in FG NAND, first proposed CT technologies used a gate structure composed by a top layer with low resistivity (typically a metal silicide) and a bottom polysilicon layer, but more advanced planar CT devices use full metal gate.

Like interpoly materials in FG process, blocking dielectric must avoid diffusion of electrons trapped in storage layer to gate, but this element has to carry out additional tasks during write operations giving to blocking layer a fundamental role in cell performance and reliability.

Many requirements must be respected by the ideal blocking material.

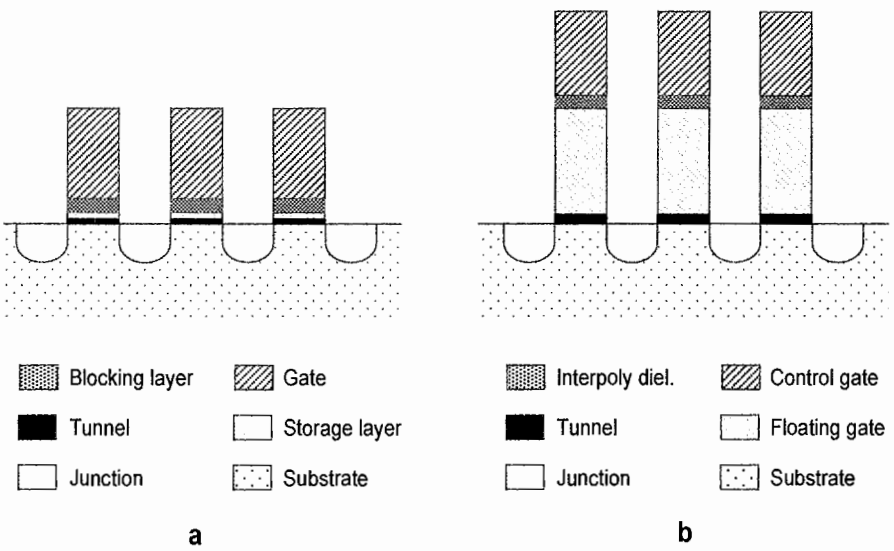


Fig. 5.1. Comparison between CT cell stack (a) and conventional FG memory (b)

One of the most important is the very high dielectric constant that allows increasing the thickness of the layer to improve retention keeping a reasonable Equivalent Oxide Thickness (EOT) of the whole stack.

Petitioner Micron Ex-1021, 0124

Electric field in tunnel dielectric during program or erase is equal to

$$E_{tun} = \frac{V_G - V_{FB}}{EOT} \tag{5.1}$$

where V_G is the bias applied to gate and V_{FB} is the flatband voltage. Looking at Eq. (5.1) it is clear that a small EOT is mandatory to have the high electric field needed for fast write operations.

A second crucial requirement is a very limited charge trapping in cycling to avoid performance degradation during product lifetime. In a scaled technology node only few hundreds of electrons are stored in trapping layer during program and this requires a very limited leakage through blocking layer during retention.

To achieve this goal, it is important to assure the highest possible band offset between the conduction bands of storage and blocking materials since this barrier plays a fundamental role in limiting charge diffusion from storage layer to gate. Unfortunately this request is in conflict with the need of high dielectric constant: in fact looking at the scatter plot between dielectric constant and energy gap reported in Fig. 5.2, it is possible to appreciate that materials with high dielectric constant are characterized by limited band gap [2].

Ideal storage dielectric must suit the following basic requirements:

- Very high trap density. A material characterized by many traps can store charge in a thin layer and this allows maintaining a low EOT for the whole dielectric stack. Moreover the use of a storage layer with high trap density is becoming more important scaling technology node since a smaller cell needs a higher trapped charge to keep the same shift during program due to fringing effects.
- Deep trap levels to limit the risk of electron detrapping through tunnel and blocking layers.
- High dielectric constant in order to reduce stack EOT.

Like in FG NAND, the role of tunnel is fundamental to realize a cell with good performance and reliability. Since tunnel thickness is scaled compared to FG NAND, it is mandatory the use of a dielectric with a very high band offset and low trap density to limit leakage during retention. Silicon oxide fits both requirements and moreover can be easily obtained in thin layers with low defectivity. For these reasons most charge trap cells proposed up to now use silicon oxide alone or in multi-layer stacks to modulate tunnel barrier shape [3].

No special requirement is needed for substrate of charge trap memories. Most technologies presented in literature use a conventional silicon substrate, only few papers present a different substrate choice like Silicon-On-Insulator (SOI) [4].

5.2.2 Cell write mechanisms

Band diagrams of SONOS and floating gate stacks are compared in Fig. 5.3. Basic concept of the two memories is the same: energy of electrons inside storage materials (typically nitride for charge storage devices and polysilicon for FG cells)

Petitioner Micron Ex-1021, 0125

is lower then bottom of conduction bands of tunnel and blocking dielectrics preventing diffusion of trapped charge.

For both cells, write operations consist in the modulation of the number of stored electrons by mean of tunneling currents obtained applying a different bias to gate and substrate.

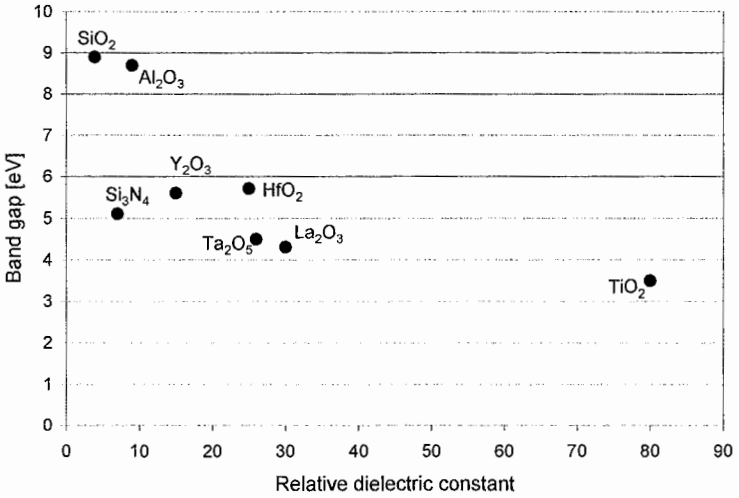


Fig. 5.2. Band gap versus relative dielectric constant for some dielectric materials

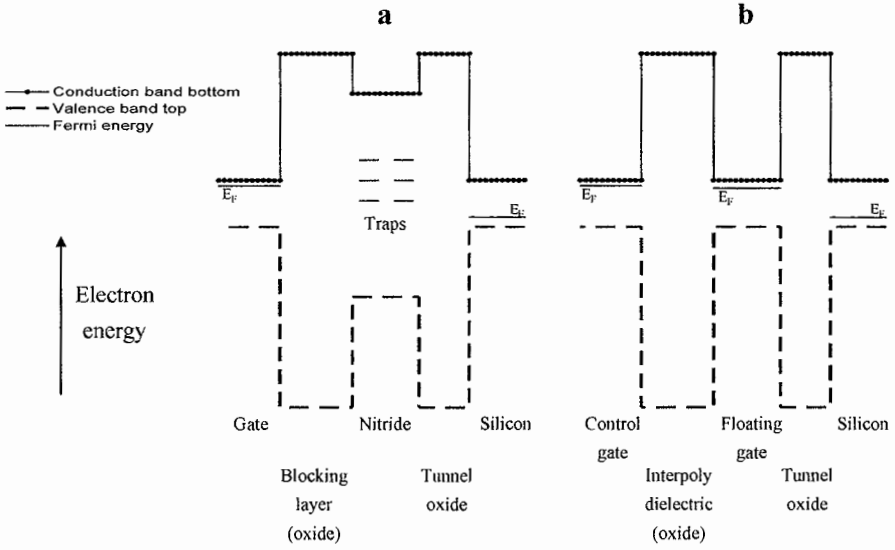


Fig. 5.3. Band diagram comparison between SONOS (a) and floating gate (b)

Petitioner Micron Ex-1021, 0126

Differently to FG devices for which Fowler-Nordheim current in tunnel oxide is the only relevant mechanism for write operations, more mechanisms contribute to program and erase behavior for CT cells. Development of simulation tools [5, 6] able to quantify the contribution of each of these phenomena and match the experimental characteristics is a fundamental research activity for optimization of cell stack. Purpose of this section is to describe physical background necessary for the comprehension of write operations of a CT cell.

Program

In Fig. 5.4 the band diagram of a typical CT cell during program is reported with a description of main active mechanisms. Electric field in cell stack injects electrons from substrate in active dielectrics. A part of this charge is captured by traps inside storage layer increasing cell threshold. The following equation describes the number of electrons trapped as a function of time:

$$\frac{dN_T(t)}{dt} = \alpha_T \cdot J_{FN} \cdot \left[N_{T,0} - N_T(t)\right] / q \tag{5.2}$$

where α_T is the capture cross section, J_{FN} is the Fowler–Nordheim current density (i.e. density of the current injected from substrate) and $N_{T,0}$ is trap density in storage layer.

Electrons that are not trapped in storage layer flow through the stack without any interaction and are collected by gate electrode. This current, which does not contribute to cell programming, is not present in FG cells. This difference explains the lower CT efficiency compared to FG that could be quantified by ramp program slope. If program is performed with a voltage ramp, cell threshold shifts with constant slope after a transient time. The ratio between threshold shift and voltage ramp is 1 for FG, whereas in CT cells this number is lower than 1 and measures the program degradation of a specific CT architecture.

Differently from FG, in CT cells a significant percentage of stored electrons detraps during all program operation through blocking dielectric due to Poole–Frenkel and tunneling emission. In case of Poole–Frenkel emission, trapped electrons are thermally excited to conduction band and then detrapped from storage layer by the electric field.

At the end of program operation efficiency drops for three reasons:

- Increased detrapping due to higher electric field in storage and blocking layers
- Reduced number of electrons injected from substrate due to lower tunnel electric field
- Less electrons captured due to limited number of free trap levels available

When number of detrapped and trapped electrons is equal, cell threshold saturates. A saturation threshold independent by program condition is typically caused by the lack of free trap levels in storage element, whereas a variable saturation threshold is obtained when the number of trapped and detrapped electrons reaches a balanced condition.

Petitioner Micron Ex-1021, 0127

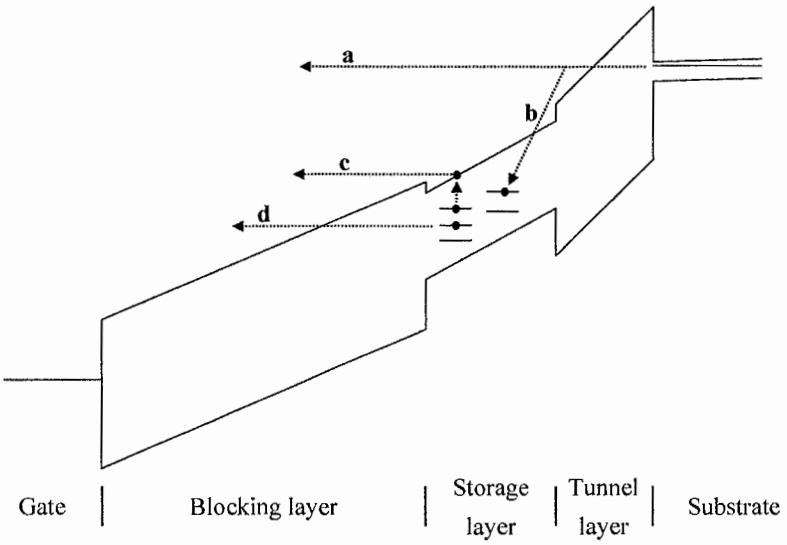


Fig. 5.4. Band structure of a typical stack during program with most important mechanisms: (a) tunneling electrons without interaction with storage layer; (b) trapped electrons; (c) Poole–Frenkel detrapping; (d) tunneling detrapping

Erase

As shown in Fig. 5.5, erase is the result of electron detrapping due to tunneling. Since tunnel probability is higher for charge trapped in shallower traps, storage materials with trap levels near conduction band are characterized by fast erase, but poor retention. *Vice versa* deep trap dielectrics have good retention properties and limited erase performance.

To overcome this trade-off, band engineered (BE) tunnel stacks activate hole injection to speed up erase. Holes usually do not have a significant impact on erase since typical thickness of a single layer tunnel oxide prevents their injection from substrate, but the use of a BE tunnel layer can increase many orders of magnitude injected hole number providing a faster erase and lower cycling degradation [7].

Electron detrapping through tunnel layer can be partially compensated by trapping in storage dielectric of electrons injected from gate. This parasitic back-tunneling current can be limited by the optimization of gate and blocking layers since the increased band offset between gate materials with high work-function and blocking layer lowers tunneling probability and the introduction of blocking layers with elevated dielectric constant reduces electric field. Like in program operation, also threshold during an erase pulse tends to reach a saturation value given by the balance of detrapping and trapping currents.

Petitioner Micron Ex-1021, 0128

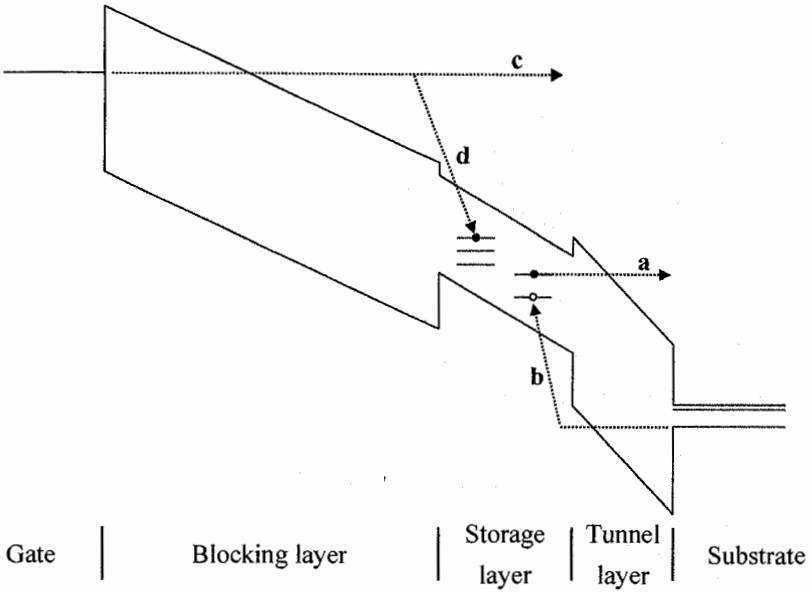


Fig. 5.5. Band structure of a typical stack during erase with most important mechanisms: (a) electrons detrapped from storage layer; (b) hole injection (minor for standard tunnel oxide); (c) back-tunneling current; (d) electrons trapped in storage layer

5.2.3 Stack options

In the last decade many CT NAND options have been proposed to continuously improve cell performance. The criterion used to depict these architectures is the description of the materials that realize the basic components of cell stack (Table 5.1). In this section a review of the most important CT options are reported.

Table 5.1. Summary of the most important acronyms used for CT memories

	SONOS	SANOS	MANOS	BE-MANOS
Gate	polySilicon	polySilicon	Metal	Metal
Blocking layer	silicon diOxide	Alumina	Alumina	Alumina
Storage layer	silicon Nitride	silicon Nitride	silicon Nitride	silicon Nitride
Tunnel layer	silicon diOxide	silicon diOxide	silicon diOxide	Band Engineered Oxide
Substrate	Silicon	Silicon	Silicon	Silicon

Petitioner Micron Ex-1021, 0129

SONOS (Silicon-Oxide-Nitride-Oxide-Silicon) [8] is the simplest and first proposed stack. The basic limitation of this architecture is related to the use of the same material for tunnel and blocking dielectric. As a result of this choice when there is no charge stored in nitride layer, the electric field is the same in tunnel and blocking oxide. As a result, erase operation is very slow and erase saturation threshold is near native threshold.

SANOS cell has been the first improvement of SONOS using Al_2O_3 as blocking layer. This change drastically reduces electric field in blocking layer due to the higher dielectric constant of alumina compared to oxide, shifting to lower values erase saturation threshold. Moreover with this stack is possible to reduce total EOT thickness of the cell stack increasing the electric field on tunnel and improving program and erase speed.

As explained in Fig. 5.6, introduction of metal as bottom gate layer gave an additional improvement compared to SANOS since use of materials characterized by work-function higher then n-doped silicon grants a faster and more effective erase [9]. In the session devoted to write operations, it has been explained that gate work-function plays a crucial role in controlling back-tunneling phenomenon that must be carefully considered and limited to guarantee good erase performance of CT cells. The acronyms used to indicate this option are MANOS (more general to indicate any metal layer) or TANOS (this latter term is used when metal is tantalum or titanium nitride). Alumina and nitride are still the most studied and promising materials for charge trap NAND devices, but many other candidates have been considered.

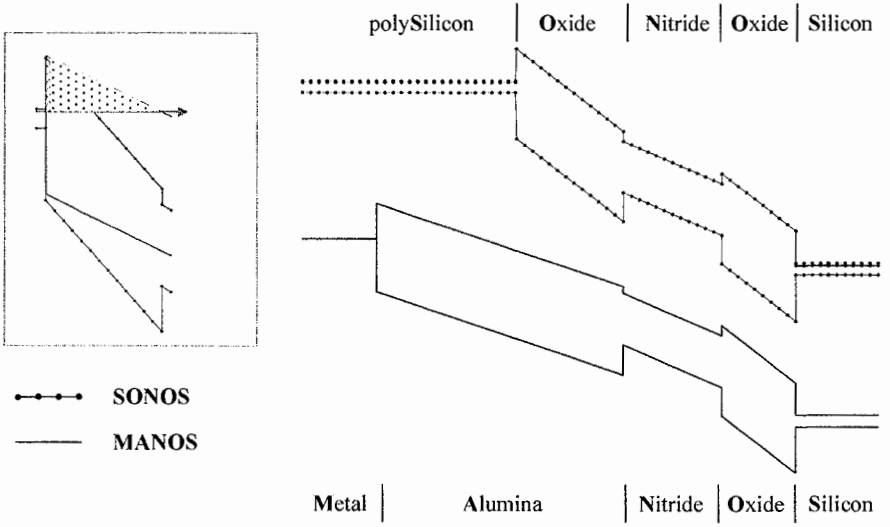


Fig. 5.6. Band structure comparison between SONOS and MANOS stacks with same EOT. In the inset on the left: gate-blocking region after alignment of Fermi level of electrons in gate and gate-blocking interface. A clear increase of barrier preventing back-tunneling injection is visible for MANOS.

Petitioner Micron Ex-1021, 0130

MHNOS (or THNOS) and MAHOS (or TAHOS) are the words typically used to define a general stack using a high k dielectric as blocking or storage layer respectively.

As explained before, recently multi-layer stacks have been proposed to obtain better blocking, storage or tunnel performances. Only two multi-layer options will be reported here: MAONOS and BE-MANOS.

MAONOS is the acronym used to describe a widely proposed double-layer blocking dielectric [10]. A thin bottom silicon oxide provides a better band offset with nitride to improve retention whereas thicker top alumina layer limits total EOT and improve cell performance.

As explained in the section about write mechanisms, BE-MANOS [11, 12] (i.e. MANOS stack with band engineered tunnel) represents a promising improvement of standard MANOS architecture. Many stacks have been proposed to obtain a BE tunnel, but the typical multi-layer structure is made by a bottom and top layers of SiO_2 including a different dielectric (e.g. Si_3N_4 or HfO_2).

Purpose of the SiO_2 layers is to get better retention preventing electron tunneling from storage layer to substrate or *vice versa*, whereas the inner dielectric modulates the tunnel barrier allowing a higher current during program and erase (Fig. 5.7).

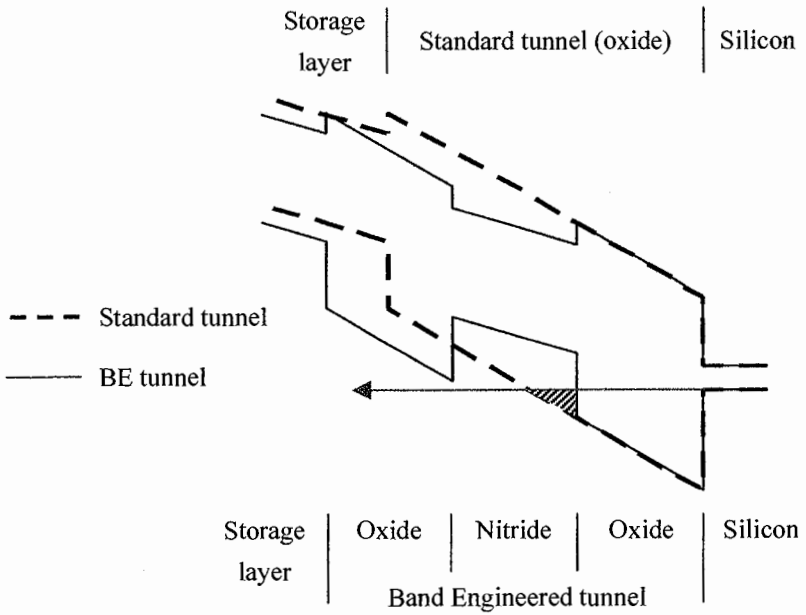


Fig. 5.7. Band structure comparison between two stacks with same EOT and different tunnel during erase. Smaller barrier for hole injection is pointed out in case of BE tunnel

Petitioner Micron Ex-1021, 0131

5.2.4 Why charge trap memories?

As reported in the introduction, charge trap NAND architecture is now deeply investigated since it could represent a viable solution for some issues that can limit future scalability of present FG NAND. Most important improvements are:

- Huge electrostatic interference reduction
- Higher immunity to active oxide defectivity
- Easier process integration

In FG NAND a lot of activity has been done on process [13, 14] and algorithms to limit fringing issues, but a clear worsening of this phenomenon is observed scaling the technology node. Even if CT NAND are not completely unaffected by this problem [15], a drastic fringing improvement could be measured comparing CT versus FG at the same technology node.

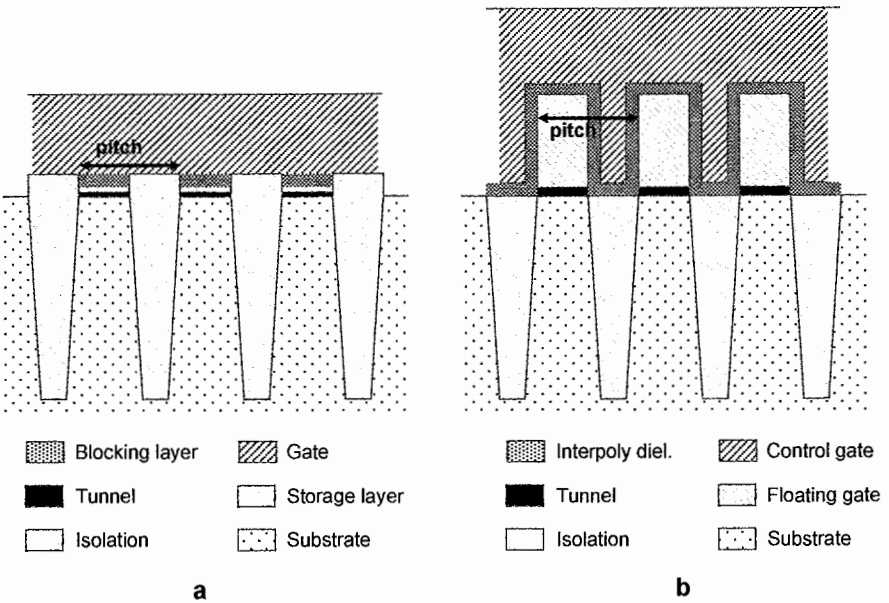


Fig. 5.8. Bitline cross-section of charge trap (a) and floating gate (b) NAND

The second big advantage of CT compared with FG NAND cells is the higher resistance to active dielectric defectivity both native or SILC. Stress Induced Leakage Current (SILC) is the defectivity caused in active oxides by electrical stress. In case of a defect in tunnel or interpoly oxide, almost all the charge inside a FG cell is suddenly lost, whereas in CT NAND only the charge stored near the defect is detrapped. Superior robustness to defectivity issues allows CT NAND to greatly shrink tunnel oxide thickness without any major retention problem.

Petitioner Micron Ex-1021, 0132

Also from process integration point of view, CT flow provides advantages compared to FG architecture resulting in higher cell scalability. As shown in Fig. 5.8, in FG technologies bitline pitch is equal to the sum of floating gate width, two interpoly layers and control gate coupling/shield layer width, and the scaling of all these parameters have a direct impact on cell performance and reliability. On the contrary, charge trap memories are characterized by a flat architecture that allows an easier shrink path.

CT process has some beneficial effects also in wordline pitch reduction since floating gate removal simplifies wordline lithography and etch. A further opportunity to scale down wordline pitch could be obtained by the introduction of junctionless approach that has been demonstrated in CT architecture [16, 17]. Due to floating gate removal, gate control of junction area is higher for CT cells. In a scaled technology node with an optimized channel doping profile engineering, the limited space between two adjacent cells could be inverted by the pass bias even without any junction and the removal of junction implant increases cell gate effective length allowing a further scaling for this specific CT architecture.

5.2.5 Planar charge trap issues

Even if CT NAND array are a promising solution to critical issues that will prevent future scalability of FG memories, they have specific marginalities that could limit performances or reliability. Two problems have been introduced in the section about write operations. As explained, both in program and in erase cell threshold saturates providing an operative working window typically smaller then standard FG one. More advanced CT architectures have greatly improved this parameter, but other progresses must be introduced to fulfill market trend to use products with many bits for each cell.

Other possible marginalities can be caused by program slope reduction explained during the description of write mechanisms. The higher bias needed to compensate limited program efficiency can induce stack degradation and enhance program disturb marginalities. Also read disturb is usually worse then FG case due to tunnel thickness reduction possible in CT arrays as a result of the superior robustness to tunnel defectivity. As reported in literature [18], improvements to disturb problems can be obtained increasing blocking layer band offset versus storage layer or introducing a band engineered tunnel dielectric.

Last critical item about CT is cycling degradation due to active oxides worsening caused by the higher current that flows through a CT stack compared with a FG one in write operations and by the increased bias in program.

5.3 3D charge trap memories

Even if planar charge trap memories seem a promising option to overcome scaling issues of FG products, more effective array organizations are continuously under

Petitioner Micron Ex-1021, 0133

development. 3D arrays could represent the most cost efficient solution for mass storage NAND products.

The most spontaneous solution to realize a 3D array consists in the multiple stacking of many planar arrays [19]. Even if this solution is able to realize products with a very small equivalent cell area, its process cost is high since many critical masks must be replicated for each vertical level. Equivalent cell area is equal to unit cell area divided by the number of vertical levels used by 3D architecture.

To overcome cost issue, recently alternative architectures characterized by vertical channel have been presented. The fundamental feature common to these arrays is the reduced number of masks since critical masks are exposed only once to etch all vertical levels at the same time. Most research activity in 3D array development seems now focused on vertical channel arrays because products realized with these technologies have a very good cost versus cell density ratio.

In Fig. 5.9 is reported a generic cross-section of a gate-all-around vertical channel technology.

Most relevant characteristics of three promising 3D organizations with vertical channel are summarized in Table 5.2.

Table 5.2. Summary of the most important characteristics of 3D vertical channel arrays

	BiCS [20]	VRAT [21]	TCAT [22]
Cell operation mode	Depletion	–	Enhancement
Gate all around	Yes	No	Yes
Gate/channel first	Gate first	Channel first	Channel first
Gate	p+ polysilicon	n+ polysilicon	Tungsten
Blocking/storage/ tunnel	Oxide/nitride/ oxide	Alumina/nitride/ oxide	Oxide/nitride/ oxide
Channel	n polysilicon	Polysilicon	p polysilicon

Cell operation mode is a critical parameter for any 3D array. For vertical channel architecture, process integration of an enhancement mode cell could be more complex since implant doping is critical due to channel orientation and in situ doping of a p type polysilicon is not easy to control. On the other side, depletion mode cells are less robust to leakage issues then enhancement mode ones causing an increased off current or a less effective boosting of unselected bitlines during program.

Gate-all-around has a double beneficial impact on cell performances. A better sub-threshold slope and a reduced leakage can be achieved if gate completely wraps around the channel.

The second advantage is that electric field is higher in tunnel oxide due to different curvature radius of active layers resulting in a significant cell performance improvement compared to planar structures.

Petitioner Micron Ex-1021, 0134

Like in planar NAND, cell electrical performances and equivalent cell area will be fundamental parameters to evaluate a new architecture, but in 3D case robustness to process marginalities will be a key parameter for a successful technology.

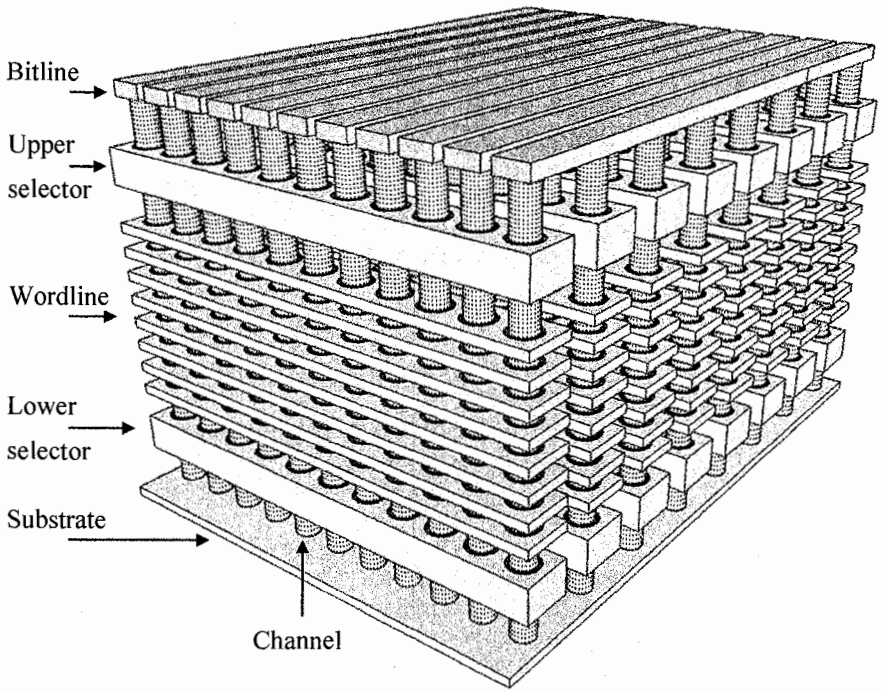


Fig. 5.9. Cross-section of a general 3D array with vertical channel

Since each process has its specific weak points, it is not possible to perform a complete review of potential issues, but a general problem that should be managed by any future 3D organization is thickness and uniformity control for tunnel, storage and blocking layers.

Acknowledgments

The author would like to thank Emilio Camerlenghi (Numonyx, R&D Technology Development) for the useful suggestions.

Petitioner Micron Ex-1021, 0135

References

1. K. Kim, "Future outlook of NAND Flash technology for 40nm node and beyond", Proceedings of NVSMW 2006.
2. J. Robertson, "Band structures and band offsets of high K dielectrics on Si", Applied Surface Science, 2002.
3. K.K. Likharev, "Layered tunnel barriers for nonvolatile memory devices", Appl. Phys. Lett., P.2137, 1998.
4. Y.K. Lee et al., "Multi-level vertical channel SONOS nonvolatile memory on SOI", Symposium on VLSI technology 2002.
5. E.S. Choi et al., "Modeling and characterization of program/erasure speed and retention of TiN-gate MANOS (Si-oxide-SiN_x-Al_2O_3-metal gate) cells for NAND Flash memory", Proceedings of NVSMW 2007.
6. A. Mauri et al., "A new physics-based model for TANOS memories program/erase", IEDM tech. digest 2008.
7. G. Ghidini et al., "Cycling degradation in TANOS stack", Proceedings of INFOS 2009 conference.
8. T.Y. Chan et al., "A true single-transistor oxide-nitride-oxide EEPROM device", IEEE Elect. Dev. Lett., vol.8, p.93, 1987.
9. C.H. Lee et al., "A novel SONOS structure of SiO_2/SiN/Al_2O_3 with TaN metal gate for multi-Giga bit Flash memories", IEDM tech. digest 2003.
10. S.C. Lai et al., "An oxide buffered BE-MANOS charge trapping device and the role of Al_2O_3", Proceedings of NVSMW 2008.
11. H.T. Lue et al., "BE-SONOS: a Bandgap Engineered SONOS with excellent performance and reliability", IEDM tech. digest 2005.
12. S.C. Lai et al., "MA BE-SONOS: a Bandgap Engineered SONOS using metal gate and Al_2O_3 blocking layer to overcome erase saturation", Proceedings of NVSMW 2007.
13. M. Park et al., "Effect of low-K dielectric material on 63nm MLC (Multi-Level Cell) NAND Flash cell arrays", Proceedings of tech. papers of VLSI-TSA 2005.
14. D. Kang et al., "The air spacer technology for improving the cell distribution in 1 Giga bit NAND Flash memory", Proceedings of NVSMW 2006.
15. Y.W. Chang et al., "A new interference phenomenon in sub-60nm nitride-based Flash memory", Proceedings of NVSMW 2007.
16. C.H. Lee et al., "Highly scalable NAND Flash memory with robust immunity to program disturbance using symmetric inversion-type source and drain structure", Symposium on VLSI technology 2008.
17. H.T. Lue et al., "A novel junction-free BE-SONOS NAND Flash", Symposium on VLSI technology 2008.
18. A. Furnemont et al., "Physical understanding of SANOS disturbs and Variot engineered barrier as a solution", Proceedings of NVSMW 2007.
19. S.M. Jung et al., "Three dimensionally stacked NAND Flash memory technology using stacking single crystal Si layers on ILD and TANOS structure for beyond 30nm node", IEDM tech. digest 2006.
20. H. Tanaka et al., "Bit Cost Scalable technology with punch and plug process for ultra high density Flash memory", Symposium on VLSI technology 2007.

Petitioner Micron Ex-1021, 0136

21. J. Kim et al., "Novel 3-D structure for ultra high density Flash memory with VRAT (Vertical-Recess-Array-Transistor) and PIPE (Planarized Integration on the same PlanE)", Symposium on VLSI technology 2008.
22. J. Jang et al., "Vertical cell array using TCAT (Terabit Cell Array Transistor) technology for ultra high density NAND Flash memory", Symposium on VLSI technology 2009.

Petitioner Micron Ex-1021, 0137

6 Control logic

A. Marelli[1], *R. Micheloni*[2] *and R. Ravasio*[3]

6.1 Logic device view

Logic is the part of the NAND device that enables the communication with the external user and organizes data inside the device and the functionalities of the device itself. It represents the interface with the user and the brain of the device.

As it is possible to see in Fig. 6.1, the logic area is very small in comparison with pumps or page buffers.

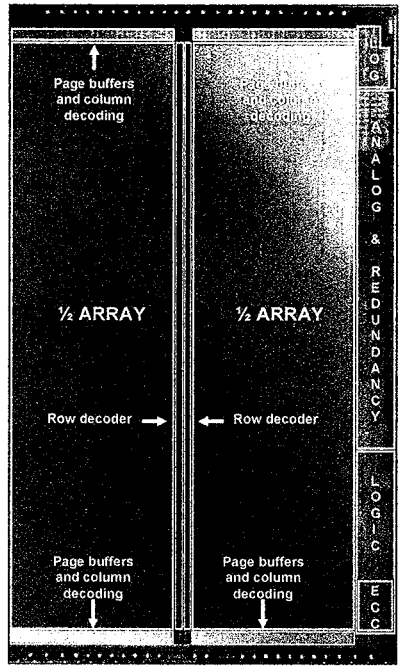


Fig. 6.1. Picture of a NAND 4 Gb MLC device, where the logic part is highlighted [1] ©IEEE

[1] Integrated Device Technology, alessiamarelli@gmail.com
[2] Integrated Device Technology, rino.micheloni@ieee.org
[3] Pegasus MicroDesign, roberto.ravasio@ieee.org

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_6, 131
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0138

The basic logic blocks are divided in six groups with different functionalities.

1. Control Interface, said CI [2, 3]. It represents the command interface between the device and the external user.
2. Microcontroller. It represents the device's "brain", because it stores and executes the algorithms the device has to perform. Therefore, microcontroller is able to move all the device parts involved in read, program or erase operations.
3. Redundancy: it can be managed by the microcontroller or it can be implemented as a finite state machine (Chap. 13). This logic is used to increase the wafer yield.
4. Error Correction Code (ECC) [4] can be embedded in the memory device (Chap. 14). This algorithm enhances the reliability of the read operation in the customer final application.
5. As we will see in Chap 15, the memory testing is a fundamental functionality. For this reason, there could be a Test Interface (TI) block on the device, that constitutes the interface with the user when device is in test mode.
6. Datapath. Basically, it is the fast link between IOs and page buffers.

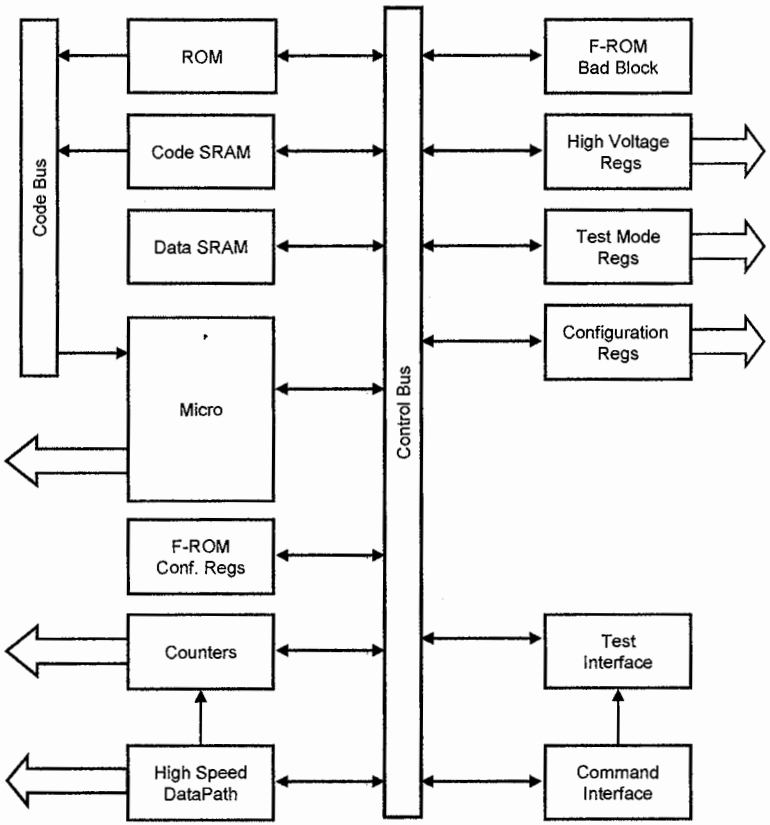


Fig. 6.2. Logic view of a NAND device

Petitioner Micron Ex-1021, 0139

In addition to these groups, there are a lot of registers in the device, storing the configurations of the device as the pumps voltages or the different ways an analog circuit can behave or an algorithm can be executed. Moreover, often an analog circuit is paired with a logic circuit able to control its functioning, e.g. row decoder. Figure 6.2 is the logic view of a NAND device.

6.2 Command interface

As already said, the first function a NAND memory must be able to do is to communicate with the external user, i.e. a memory has to understand commands, take data and output data.

The logic block implementing this functionality is basically a finite state machine and is represented by Command Interface (CI) when the device is in user mode and by Test Interface (TI) when the device is in test mode.

CI understands legal or illegal command sequences, defined in the device specification and interacts with other logic blocks as datapath or microcontroller. CI is composed by a huge finite state machine clocked by WE# and driven by all I/O signals such as ALE or CLE.

Figure 6.3 represents CI and its interaction blocks.

1. I/O is represented by all control signals toward the external user: R/B#, CLE, ALE, WP#, WE#, RE#, CE#, STACK_ID[1:0], DQ[7:0].
2. Reset Interface exchanges reset information with logic global reset.
3. Datapath interface controls datapath inputs and outputs.
4. Test interface switches between user mode and test mode and vice versa.
5. Firmware Control Interface enables microcontroller to execute the algorithms.
6. Schematically CI is sketched in Fig. 6.4.

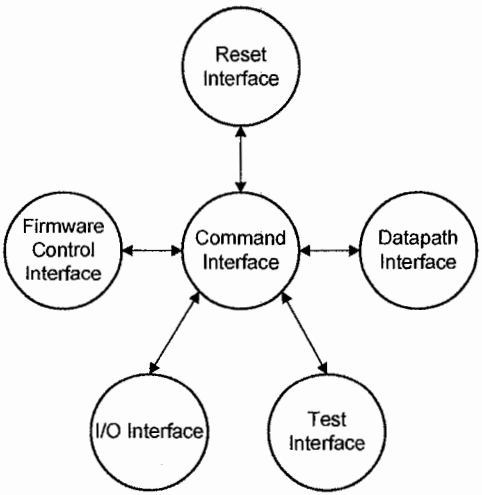


Fig. 6.3. Command interface and its interaction blocks

Petitioner Micron Ex-1021, 0140

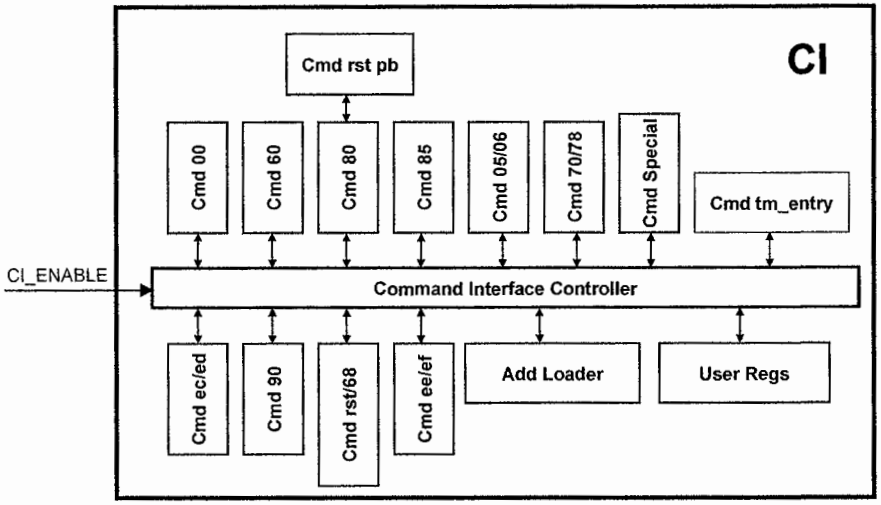


Fig. 6.4. CI basic representation

CI is constituted by a huge Finite State Machine divided in multiple finite state machines. Only one FSM works at a time.

In order to be sure that the machine works as designed, every FSM has to leave the machine in the IDLE state. An illegal command sequence has to leave the machine in the IDLE state, too. Every FSM exits from that state with a particular command sequence.

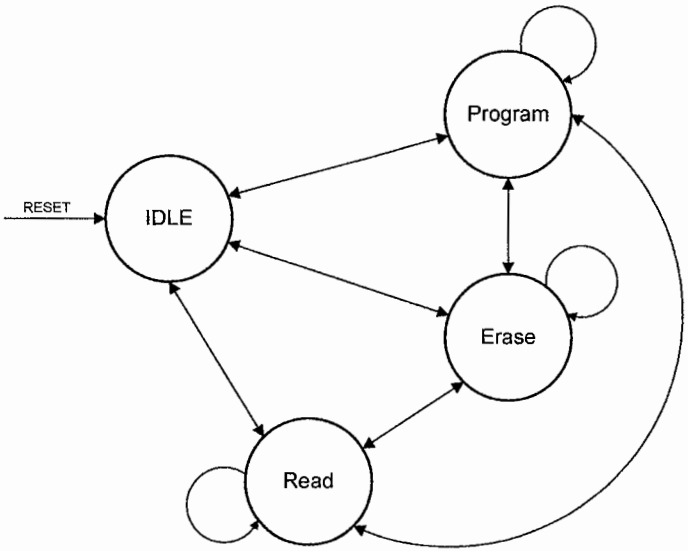


Fig. 6.5. Representation of the general FSM

Petitioner Micron Ex-1021, 0141

Command Interface Controller disables a specific FSM if the relative command is not allowed.

User Regs are common registers among all the FSMs, such as the busy flag, read allowed flag or die select.

Add Loader latches the address. It is enabled and controlled by the specific FSM active at that moment.

During power on, the CI Controller disables every commands, so that all the FSMs are disabled, too. The signal CI_ENABLE inputs CI Controller from power on logic, in order to enable the CI Controller and the commands receiving.

The commands legal for the CI are those described in the device specification (Chap. 2). There exist also some special commands, that NAND vendors reserve to key customers, handled by a FSM called Cmd Special.

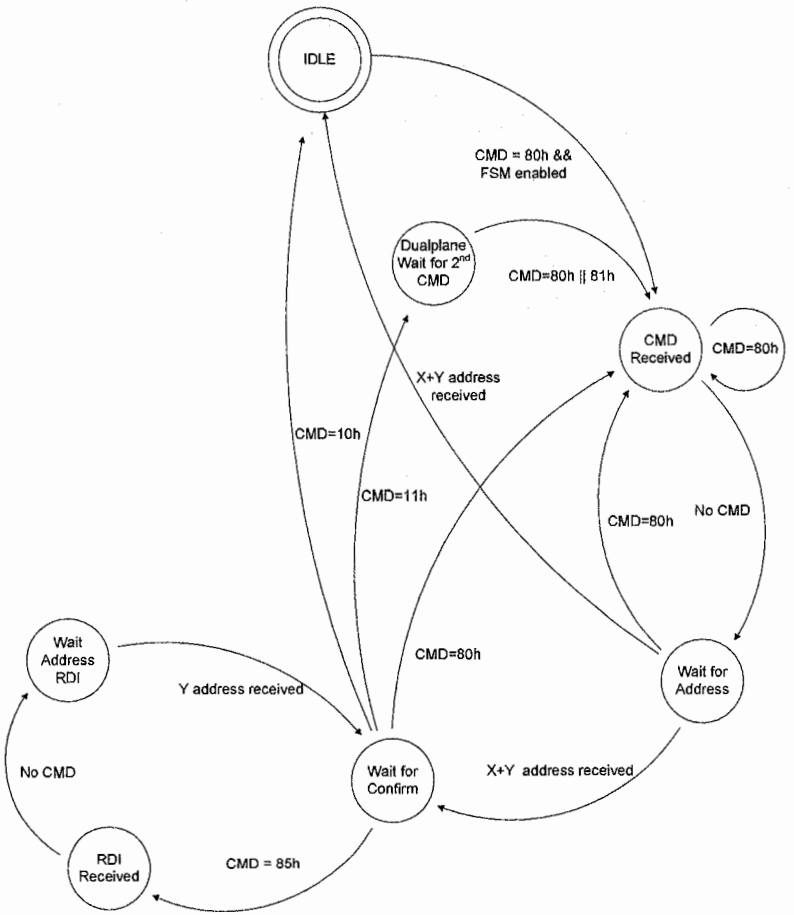


Fig. 6.6. subFSM for a program sequence

Petitioner Micron Ex-1021, 0142

Also the CI Controller is constituted by a FSM that recognizes if a specific command is a read, a program or an erase and enables the correct subFSM. Every time the Controller receives an illegal sequence, the device goes into the IDLE state as represented in Fig. 6.5.

In the following, we will show a program sequence. The subFSM enabled in this case is represented in Fig. 6.6.

Generally, a program sequence starts with the command 80h. If the correct FSM is enabled, it goes in the CMD Received state and waits for an address.

In this phase, this subFSM controls the Address Loader. The address is constituted by a specific number of bytes identifying the wordline and the bitline that must be written.

Address Loader is represented in Fig. 6.7.

If the command is a single plane program, the address is stored in both planes until the plane address is latched. At that point, only one plane is active, since the not involved plane is disabled. On the other hand, if we are dealing with a dual plane program, the address is stored in both planes until the plane address is input and the not selected plane is disabled. On the second plane, bytes are overwritten with the correct plane address. In this way both planes are active (Chap. 2).

Coming back to the flow of Fig. 6.6, the FSM is now able to receive data. A signal enables data input through datapath, at the end the subFSM waits for the confirm command.

Before confirming and starting a program, it is possible to execute a Random Data Input (RDI) command. The new command needs a new address.

It is also possible to execute a dual plane operation. In this case, the FSM waits for a dualplane command, new address and new data. If a dualplane command is issued an internal flag is set.

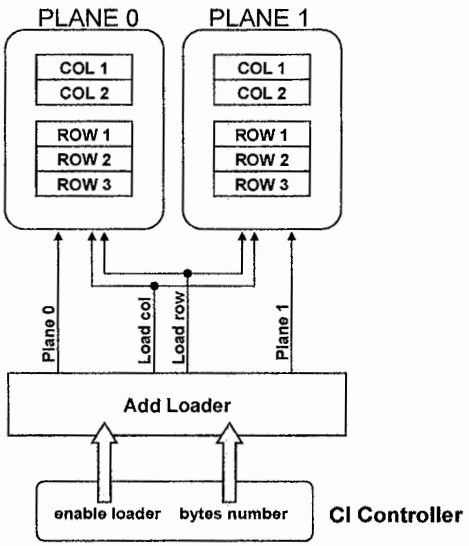


Fig. 6.7. Basic Address Loader scheme

Petitioner Micron Ex-1021, 0143

After the confirm command (10h), the device goes busy and CI enables a signal ci_program that activates microcontroller to execute the program algorithm.

If the command sequence is interrupted by illegal instructions, the FSM goes back into the IDLE state.

When microcontroller executes a specific algorithm, the device is busy. The only command CI is able to accept during device busy state is reset command and testmode entry command.

6.3 Test interface

Test Interface (TI) is used as interface, when we want to test some particular device features not accessible in usermode. Test Interface is enabled by a specific command sequence, called testmode entry. When the device is in testmode, it is possible to access Address Loader.

Generally, a device can have these modes:

- Usermode that represents the standard functionality, where commands described in the device specification are available
- Usertestmode that represents the standard functionality plus some particular commands
- Testmode that is the test operational mode

Figure 6.8 represents how it is possible to change the operational modes with proper command sequences recognized by the CI Controller.

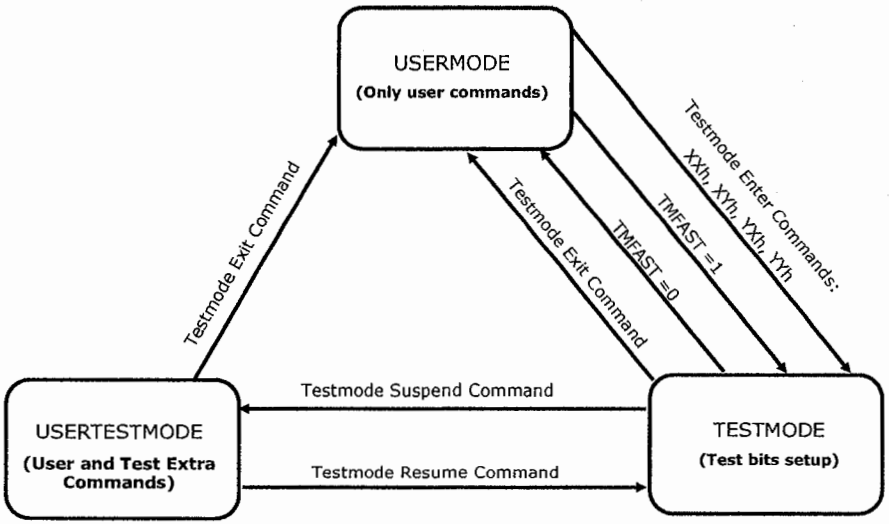


Fig. 6.8. Flow diagram used to change operational modes among usermode, usertestmode and testmode

Petitioner Micron Ex-1021, 0144

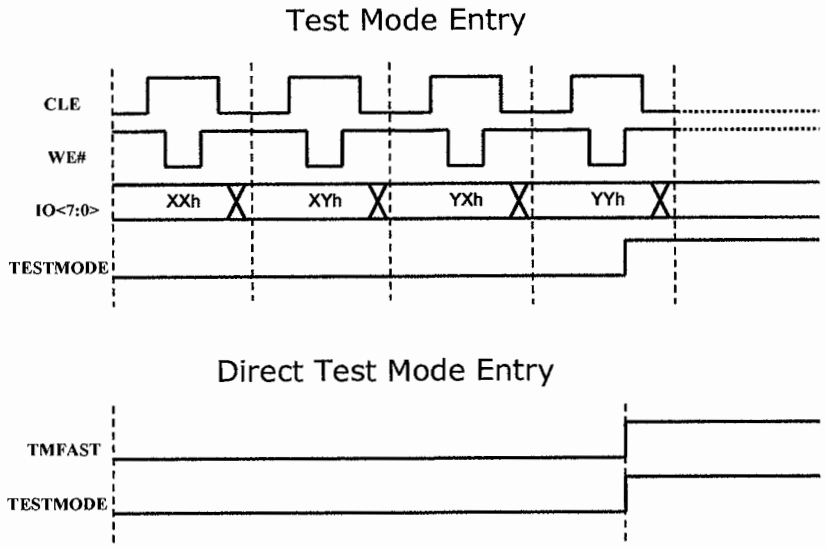


Fig. 6.9. Test Mode entry and direct Test Mode entry through the TMFAST pad

In order to have an easier and fast way to enter testmode, some devices have a special pad TMFAST that can be accessed only at wafer level. When TMFAST is forced to "1", the memory is directly in testmode as depicted in Fig. 6.9, where the two ways to enter testmode are shown.

Once TI is enabled, it substitutes CI: TI recognizes the command set and drives input and output data/address on the logic bus.

Generally, TI is allowed to access the different registers and different memory circuits without the aid of the microcontroller. A special command set is available to the TI to access registers and page buffers:

- Read PB. Data are read from page buffers.
- Read CNTRL REG. Data are read from control registers.
- Write PB. Data are written in the page buffers.
- Write CNTRL REG. Data are written in the control registers.

Read and write commands can have particular features, such as the auto-increment and the possibility to perform logical operations on the output bits. For example, it is possible to read and verify the content of all the page buffers in sequence. Obviously, TI is used also to validate the microcontroller behaviour. In order to test microcontroller, it is possible to suspend an algorithm and to re-start it from a different point to observe the exit status.

TI is built as a finite state machine in an similar way to Command Interface.

Test Interface works on Datapath through CI Controller. In particular, the only needed information is the address loader. Once the address is loaded, TI FSM must enable the selected plane and the necessary operations through testmode registers.

Petitioner Micron Ex-1021, 0145

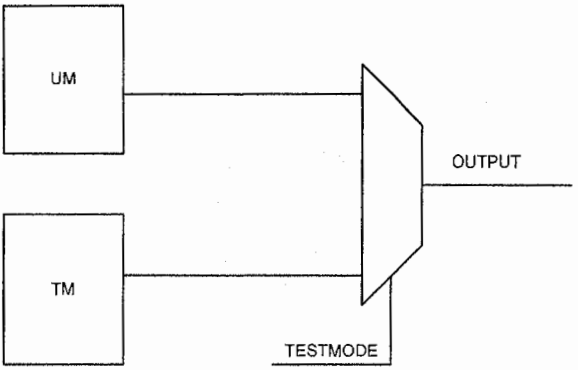


Fig. 6.10. Testmode registers

Let's now explain what testmode registers are. All the circuits added for test purposes can't influence the standard user mode functionality and can't worsen performances. The adopted solution is sketched in Fig. 6.10. A TM register is associated with a UM register: when the signal TESTMODE is high, the output takes the value contained in the register TM, influencing the behaviour of the circuitry downstream. When the signal TESTMODE goes low, the standard usermode functionality is enabled.

TI is able to work with different parallelism:

- TI × 8 where eight DQs are used as input/output pins
- TI × 4 where four DQs are used as input/output pins
- TI × 2 where two DQs are used as input/output pins

Generally, the first way is used to verify design, since TI with a register read command is able to read the configuration of the memory itself. The last two modes are used during Wafer test where a high number of devices must be tested in parallel.

The signals involved in TI × 8 are shown in Table 6.1. Generally, the pins maintain their usual functionality except WE_N, that is here used as a clock, enabling data latch on its rising edge.

Signals involved in TI × 4 and TI × 2 are shown in Table 6.2.

Table 6.1. Pins used in TI × 8 machine

Pin Name	Width	Usage	Remarks
CE_N	1	Per Chip	Separate for each chip
WE_N	1	Shared	Works as a clock
RE_N	1	Shared	Enables the output
ALE	1	Shared	Indicates a new address
CLE	1	Shared	Indicates a new command
DQ	8	Shared	Tranfers command/data/address

Petitioner Micron Ex-1021, 0146

Table 6.2. Pins used in TI × 4 and TI × 2 machines

Pin Name	Width	Usage	Remarks
CE_N	1	Per Chip	Separate for each chip
WE_N	1	Shared	Works as a clock
CLE	1	Shared	Indicates a new command
DQ	4/2	Shared	Tranfers data/command/address

As it is possible to see, the number of pins involved in TI × 4/ TI × 2 is less than in TI × 8. Also here WE_N is used as a clock to latch the data on its rising edge. There aren't any control pins used to indicate a new address or to enable the output, so the command format must be fixed. Since we have only four or two pins involved, we must double or quadruple the number of cycles in respect to TI × 8. The cycles will pass the data bytes from the most significant bit to the least significant bit.

As regarding TI × 4 and TI × 2, a logic internal to Input and Output buffer reconstructs the byte in the TI × 8 useful format (Fig. 6.11). The cycles frequency is generated with the aid of a frequency divider.

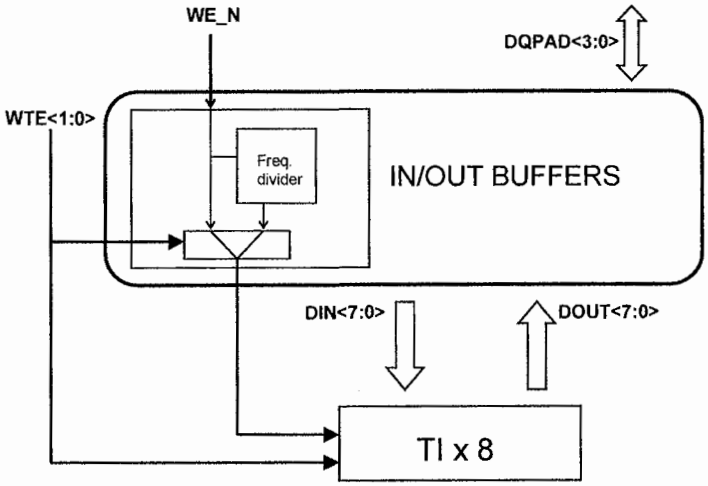


Fig. 6.11. TI × 4/2 data reconstruction for TI × 8 format

6.4 Datapath

Traditionally, NAND memories have an asynchronous interface and it is very difficult to have frequency higher that 40 MHz for the data download/upload [5]. NAND chips have linear dimensions easily higher than 10 mm so that data have to flow through a long path with an unavoidable impact on the transmission time

Petitioner Micron Ex-1021, 0147

through the chip. One of the most adopted solutions to overcame this problem is the use of a pipeline on the datapath [6].

In the following we will describe datapath structure for a NAND memory with double side architecture and with control pads on the opposite side in respect to data pads.

With reference to Fig. 6.12 the data input sequence is here described.

1. During the low-phase of WE#, input buffers on I/O PADS block and latches on DP_UP and DP_DW blocks are enabled. In this way, input data flow to the latches placed in DP_UP and DP_DW blocks.
2. On the rising edge of WE#, I/O PADS input buffers are disabled. Data are latched in DP_UP latches till the next falling edge of WE#. The counter addresses the appropriate page buffers for the following write operation.
3. On the high-phase of WE#, IO CONTROL latches are open and the COLUMN DECODER is addressing the right page buffers.

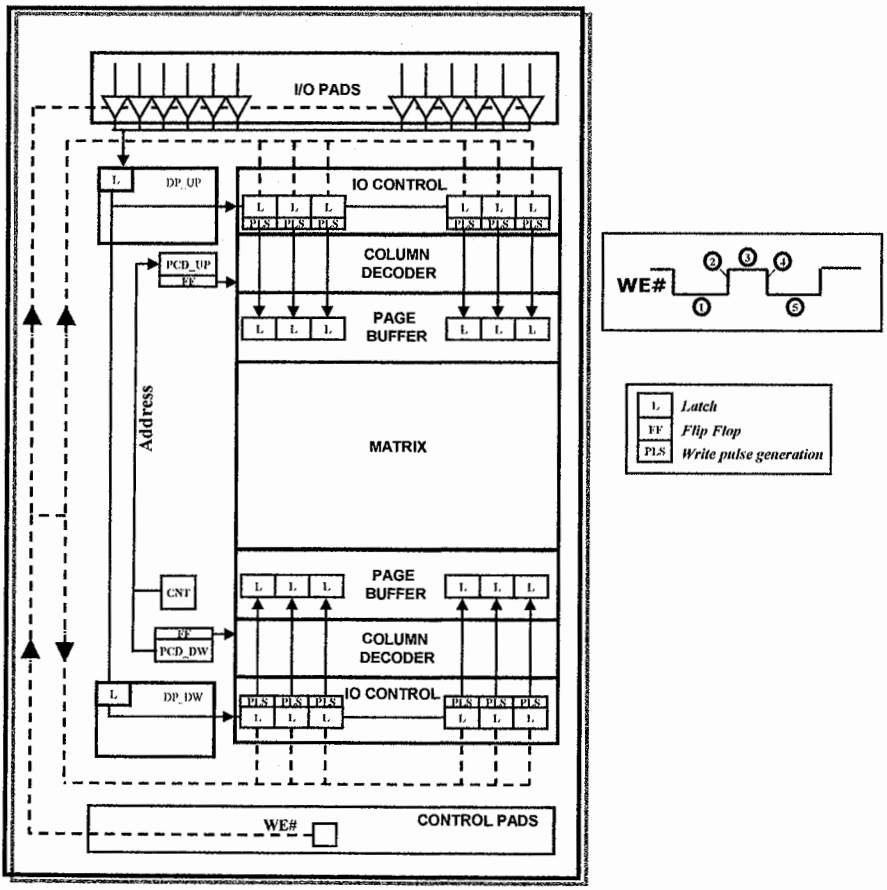


Fig. 6.12. Data input path

Petitioner Micron Ex-1021, 0148

4. On the falling edge of WE#, data are latched in the IO CONTROL latches.
5. On the next low-phase of WE#, while I/O PADS input buffers and DP_UP latches receive new data from the user (as in phase 1), IO CONTROL generates write pulses for loading the latched data into the page buffer latches.

The data output sequence involves three different phases (Fig. 6.13).

1. The address is updated by the counter on the falling edge of RE#.
2. During the entire RE# cycle, the COLUMN DECODER selects the proper page buffers, then data are sent from the page buffer latches (up and down) toward the output buffers of I/O PADS.
3. On the falling edge of RE#, flip-flop in DATA I/O PADS sample the data.

Performance driven applications like SSD are now forcing the NAND towards the adoption of the DDR interface (Chap. 7).

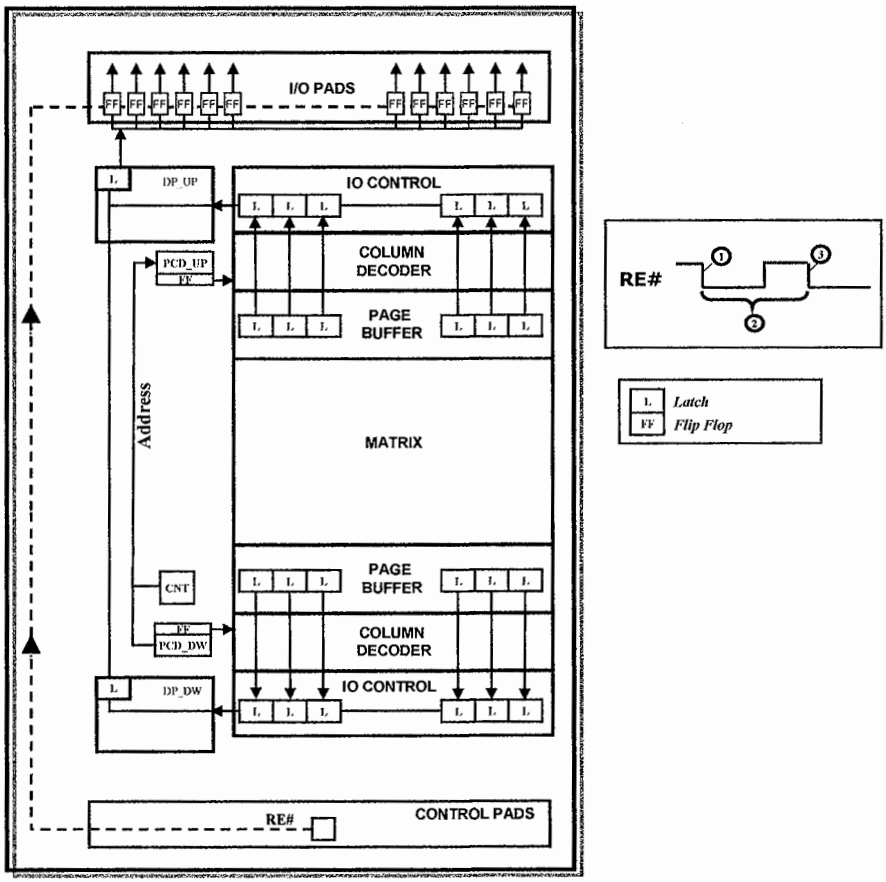


Fig. 6.13. Data output path

Petitioner Micron Ex-1021, 0149

In order to sustain the DDR throughput, designers have to put in place all the possible tricks: from pipeline architectures to differential data transmission lines as sketched in Fig. 6.14 [7] where the data coming from the page buffer enter into a cross-coupled inverter pair. Toggling the SOURCE signal, this secondary sensing circuit amplifies the small voltage difference between DATAIN and its complement, improving the overall access time.

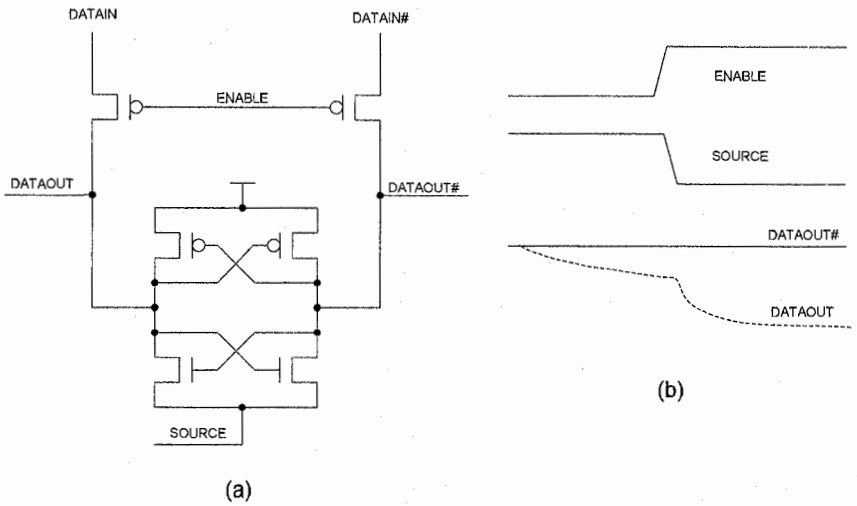


Fig. 6.14. Differential data transmission

6.5 Microcontroller

As already said, the microcontroller inside the memory is the "brain" of the device. Microcontroller implements the needed algorithms for a Flash memory. In order to be able to perform the necessary operations, these conditions must hold true:

- The sequence of operations that must be executed for a specific algorithm (read, program, erase etc.) has to be stored.
- The microcontroller has to be able to perform arithmetical, logical or ports operations.

The code storage in a ROM memory is a possible solution (Fig. 6.15). There could be also a Code RAM memory containing the specific firmware for the test and the debug of the device.

The microcontroller is constituted by different blocks necessary to the decoding and the execution of an operation.

First of all there is the *Program Counter*. It stores the address of the memory location containing the instruction that must be executed.

Petitioner Micron Ex-1021, 0150

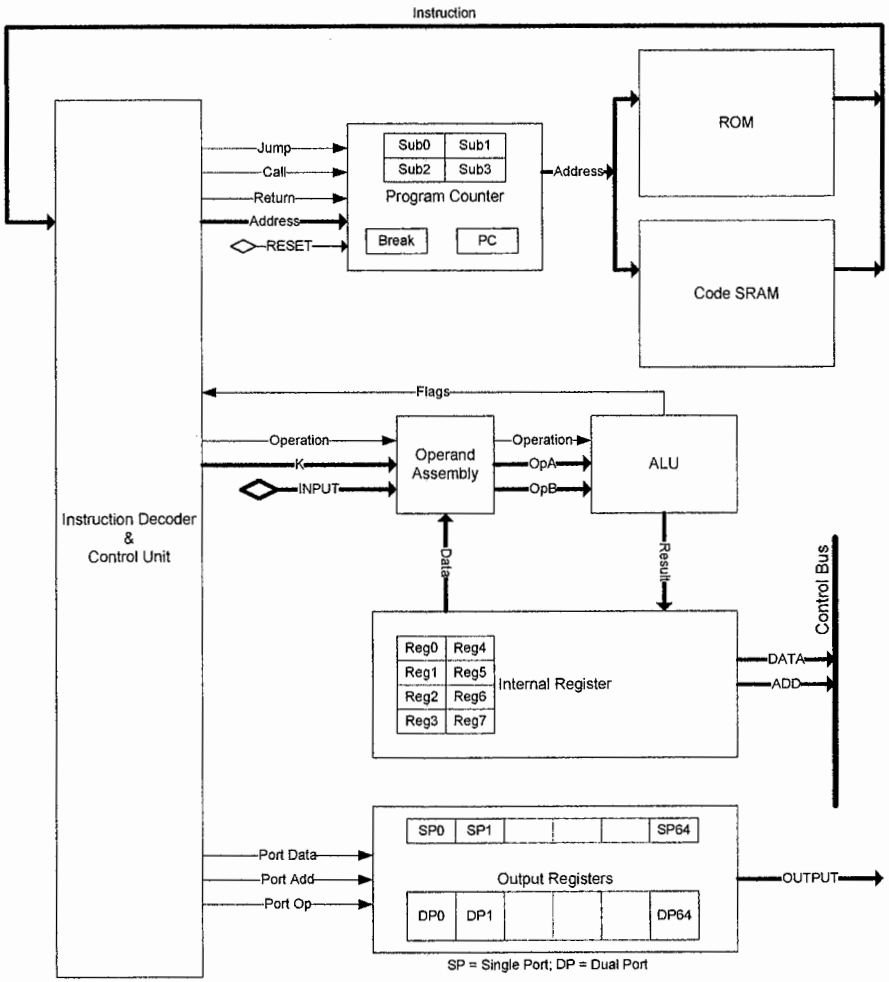


Fig. 6.15. Microcontroller structure with ROM and RAM memories

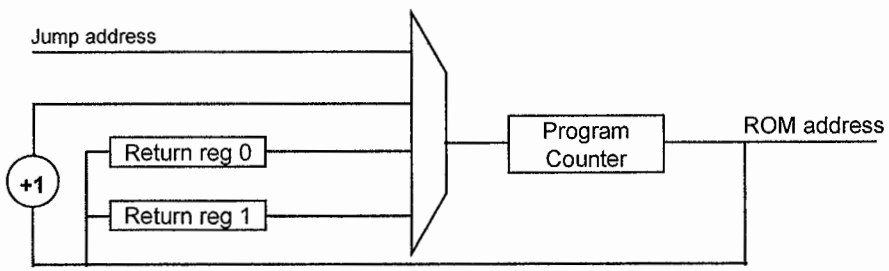


Fig. 6.16. Program counter structure

Petitioner Micron Ex-1021, 0151

It is also able to handle the address increment, the absolute or relative jumps and the calls to subroutines with different stack levels. The levels of stack indicate how a subroutine is far away from the main program.

As shown in Fig. 6.16, at each clock cycle, the new value of the program counter can be:

- The next instruction
- One of the values frozen up before the call to a subroutine (Return reg 0 and Return reg 1 in the figure)
- The address indicated by the jump if the condition is satisfied

Coming back to Fig. 6.15, another important block is composed by the *Internal Registers* (eight in the figure). They are registers internal to the microcontroller, necessary to the execution of an operation or a sequence of operations. A register can be loaded with a constant value or with a value read from the ROM or can be the result of an operation. Generally, the registers are constituted by bytes, but sometimes they can be grouped for operation having a result longer than 8 bits. We must observe that, depending on the chosen implementation, there can be constraints on the registers use, such as the division of the registers in banks so that it is not possible to use two registers of the same bank in the same operation. In addition to the registers, there are the flags, that are bits resulting from particular operations executed by the ALU.

The microcontroller computational center is the *Arithmetic Logic Unit* or *ALU*. The ALU executes an operation associated with a specific opcode and implemented in the microcontroller. The operations can be with one or two operands. The operands can be internal registers, flags or constants read from the ROM (K data). The result of the operation is stored in the internal registers, with the exception of test and compare operation. There are also some operations that set some flag bits as a result (Fig. 6.17).

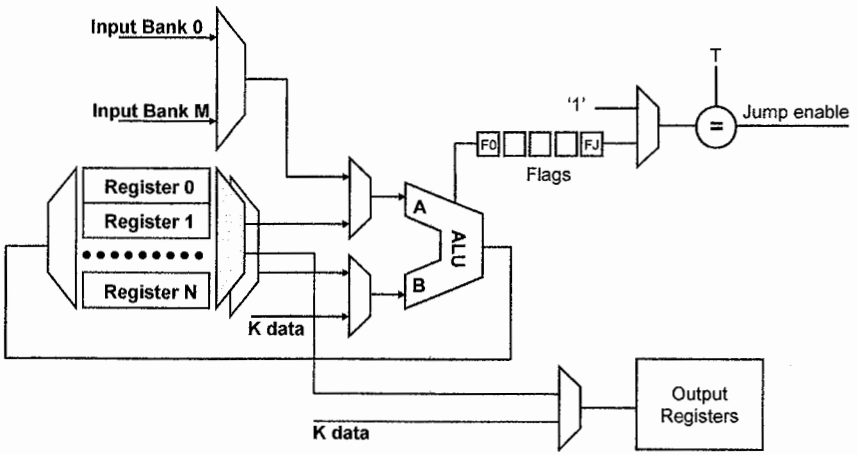


Fig. 6.17. Data manipulation inside microcontroller

Petitioner Micron Ex-1021, 0152

Generally, the used flags are listed below.

- Z: zero flag. It is used in the arithmetic operations and it goes high when the result is zero.
- C: carry flag. It is used in the arithmetic operations and it goes high when the result has a carry.
- A: and flag. It is used in the logic operations and it is the logic AND among all the bits of the result.
- O: or flag. It is used in the logic operations and it is the logic OR among all the bits of the result.

Depending on the implementation, some other flags can be added:

- N: not flag. It is used in the logic operations and it goes high when the result is a negative number.
- P: parity flag. It is used in the logic operations and it goes high when the number of bits equal to 1 in the result is even.

Based on the current instruction and on the result of the compare of the ALU flags, the Control Unit, through the Instruction Decoder, generates the control signals for the handling of the different logic units. The instructions are executed in pipeline using the clock frequency (Fig. 6.18). At the first clock cycle the instruction is read from the ROM. At the second clock cycle the instruction is stored in the instruction registers and decoded by the instruction decoder (a simple combinatorial network) [8] (Fig. 6.19).

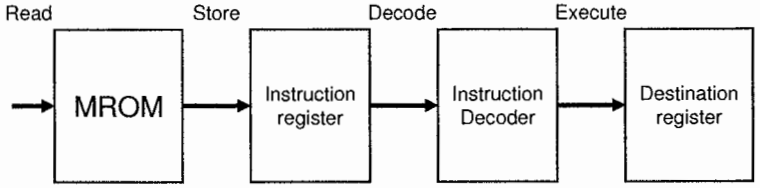


Fig. 6.18. Microcontroller timings

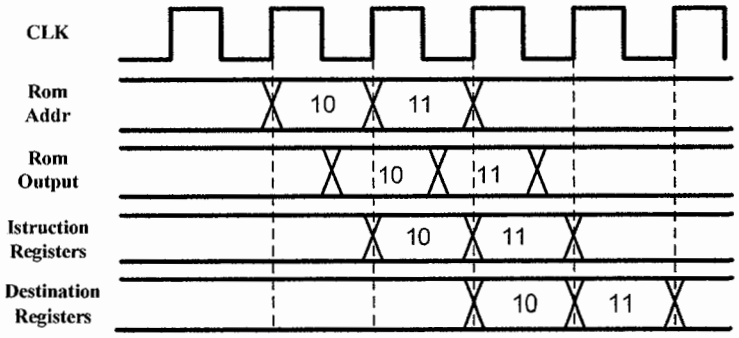


Fig. 6.19. Pipeline waveforms

Petitioner Micron Ex-1021, 0153

At the third clock cycle the instruction is executed and the result is stored in the destination register. The only un-typical operations are the jumps and the calls that need an internal NOP to synchronize the pipeline.

Finally, the last block of the microcontroller is constituted by the *Output Registers*. Each register is made up by four latches. The most advantageous structure for the output registers is based on the dual ports concept. With this structure, the registers are handled by two independent ports called port A and port B (Fig. 6.20). As it is possible to observe from the figure, port A operates over all the outputs, while port B operates only over some output registers. The dual ports structure allows the use of two different bank registers at the same time, so that it is possible to move eight control signals at each clock cycle.

There are three possible port-operations that can be executed on the outputs:

1. SET, i.e. 1, 2, 3 or 4 register bits of the output are set.
2. RESET, i.e. 1, 2, 3 or 4 register bits of the output are reset.
3. STORE, i.e. it is possible to set or reset the four output register bits independently.

In addition to these operations, we can add the PULSE operation, i.e. a positive impulse is generated.

The dual port structure is able to guarantee performances and flexibility. First of all, it is possible to execute a port operation in conjunction with a logical operation, a jump or a call. It is useful to observe that the operations on the control bus are generally slower than the port operations, because it is not possible to issue the data and the address in the same clock cycle. As a consequence, the microcontroller should use the control bus for the configuration signals and the port operations for signals outstanding for the algorithms execution.

Apart from the internal structure, the characterizing feature of a microcontroller is what it is able to do, that is its Instruction Set. Before designing a microcontroller, we have to understand the must-have operations. In fact, general purpose microcontrollers are not useful in the NAND memory environment, because they are generally bigger and slower, in order to guarantee a full flexibility not needed in the device. In other words, it is useless to implement operations not used, it is better to optimize the used ones.

As already said, the operations can be divided into logical and arithmetical operations and most of them set some flags during their execution. In Tables 6.3 and 6.4 there are the logical and arithmetical operations for a particular microcontroller ALU and the set flags. The A and B operators indicate the first and the second register, the K operator is a constant and the D operator is the destination register.

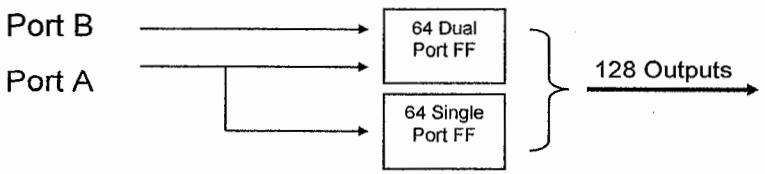


Fig. 6.20 Output registers

Petitioner Micron Ex-1021, 0154

Table 6.3. Logical operations

Name	Action	Flag 1	Flag 0
OR	A or B ≥ D	O	A
	A or K ≥ D		
AND	A and B ≥ D	O	A
	A and K ≥ D		
INV	not A(i) ≥ D	O	A
CLEAR	00h ≥ D		
SET	'1' ≥ D(i)		
RESET	'0' ≥ D(i)		
TEST	A(i) ≥ C		C
SHL			C
SHR			C
ROL			C
ROR			C

Table 6.4. Arithmetical operations

Name	Action	Flag 1	Flag 0
SUM	A + B ≥ D	Z	C
	A + K ≥ D		
SUB	A − B ≥ D	Z	C
	A − K ≥ D		
INC	A + 1 ≥ D	Z	C
DEC	A − 1 ≥ D	Z	C
CMP	A − B	Z	
	A − K		
LD	A ≥ D	Z	
	K ≥ D		

The operations described in the tables can be executed in conjunction with a jump, a call, a test or a port operation on the outputs. The contemporary execution makes the code faster and more compact, because, on one hand we can execute two operations on the same clock cycle, on the other hand, the code is shorter. For this reason, it is important to decide instruction field length. In the example depicted in Fig. 6.21 the instruction field is 32 bits long. The first 8 bits are constituted by the opcode, i.e. the identifying code of the operation. The next 8 bits stored the first operand, the used flags for the jump operation or the source register.

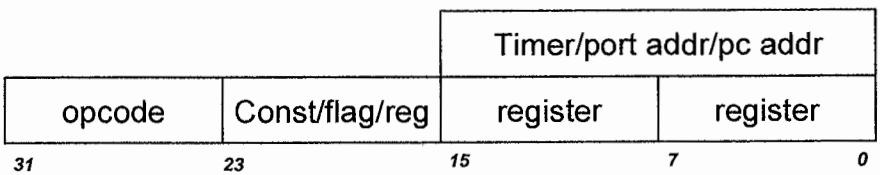


Fig. 6.21. Instruction field

Petitioner Micron Ex-1021, 0155

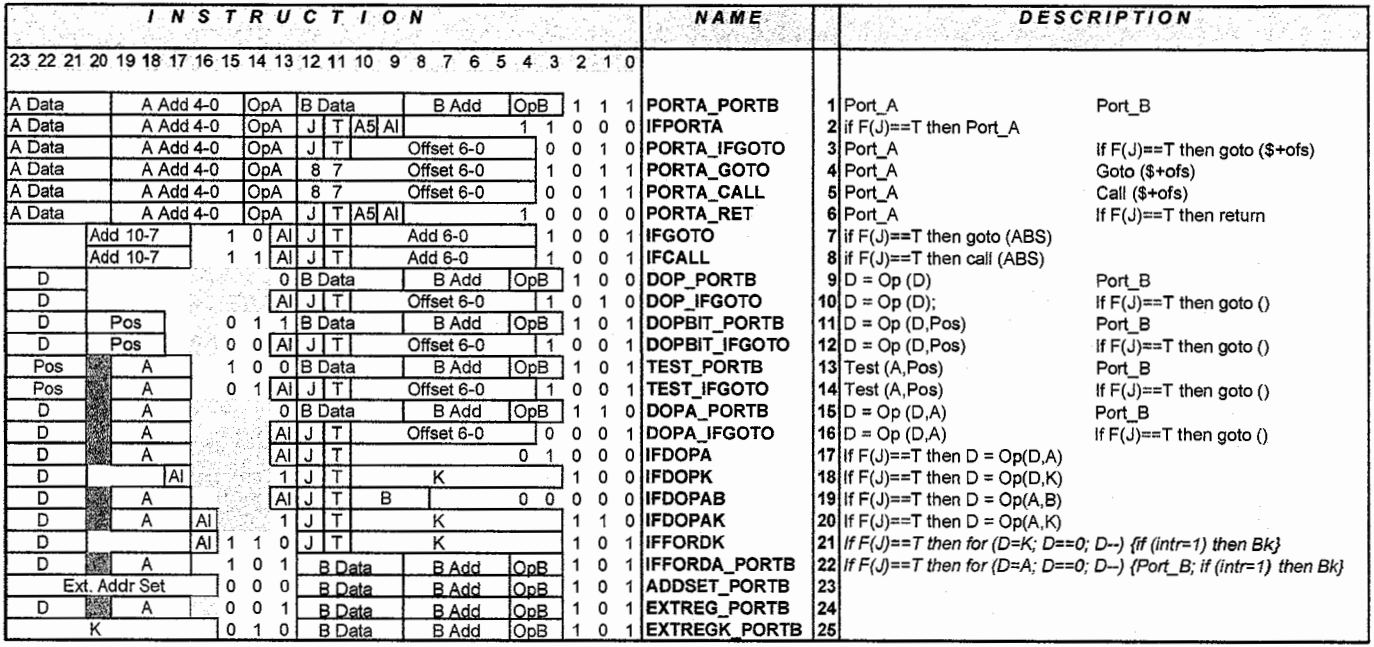


Fig. 6.22. Instruction set

Petitioner Micron Ex-1021, 0156

The last 16 bits can store the address of two registers (source and destination) or the starting point of the timer or the address of the port we want to access or the address of the jump.

Once the instruction field length is decided, every instruction must be coded following that structure. Since a smaller length is equal to a smaller area, some optimizations should be done in order to not have a rigid structure as the one indicated in Fig. 6.21, but a more flexible one with mixed fields as the one indicated in Fig. 6.22.

The instruction field presented in Fig. 6.22 is constituted by 24 bits. There are a lot of contemporary operations, such as the instruction 9 that executes an operation with a single operand in conjunction with a port operation on the outputs or the instruction 3 that executes a port operation on the outputs in conjunction with a conditional jump depending on the flag value.

The instructions 7 and 8, instead, show how the opcode field can be split and mixed with the address field.

6.6 ROM

The ROM contains the firmware necessary to the microcontroller for the algorithms execution. The ROM structure is shown in Fig. 6.23. ROM has to contain all the necessary firmware, so that its size is considerable: generally, it has to store 2k instructions (24 or 32 bits depending on the instruction field length).

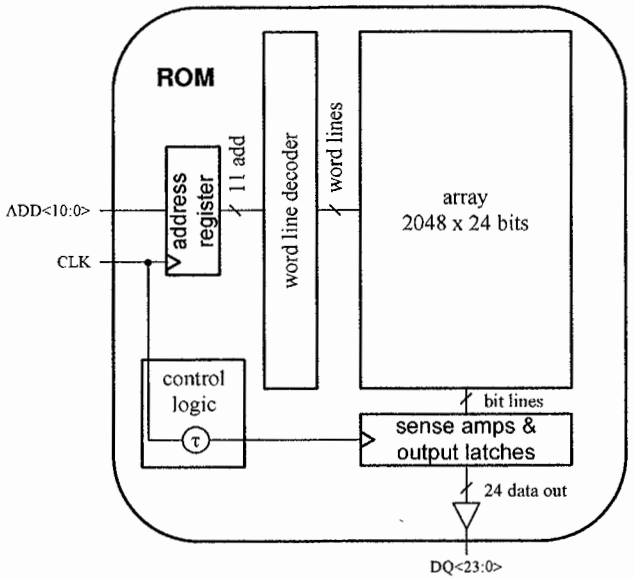


Fig. 6.23. ROM architecture

Petitioner Micron Ex-1021, 0157

ROM supports a cycle synchronous latency random read access time. Every ROM inputs are sampled with the clock raising edge, except for the reset signal. Outputs are updated when the internal asynchronous read cycle is completed and before the next clock raising edge. Typically the used clock is in the order of 20 MHz.

The interface is constituted by the signals:

- CLK i.e. the clock. At every cycle a read operation is required.
- ADD<10:0> i.e. the address.
- DQ<23:0> i.e. the output data.

Data are stored in the array as shown in Fig. 6.24.

The ROM is constituted by a right and a left array in order to minimize the wordline parasitic loads. Only one array at a time can be accessed.

Every wordline is made by data cells and dummy cells. The ROM associated with the instruction set depicted in Fig. 6.22 has 48 data cells and two dummy cells. There are 512 wordlines, one reference wordline and one dummy wordline.

Finally, every plane is made up by 48 bitlines and 512 wordlines. Since the instruction field is composed by 24 bits, every wordline contains two words of 24 bits each. The memory cell is composed by a NMOS and a metal option OPZ that can enable/disable the transistor connection with the bitline (Fig. 6.25). An open OPZ corresponds to a "1" stored, while a closed OPZ corresponds to a "0" stored. In this way, it is possible to change the ROM content with only one metallization level, which is particular important from the point of view of the costs.

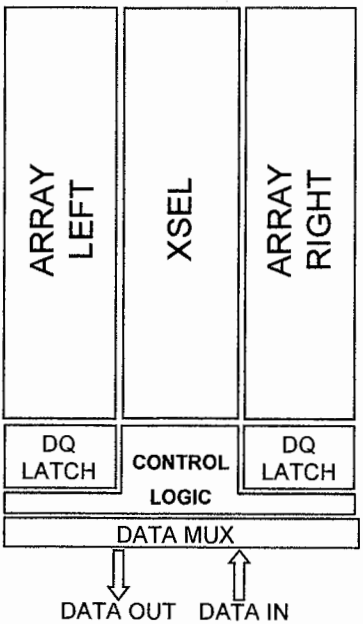


Fig. 6.24. Data arrangement inside the array

Petitioner Micron Ex-1021, 0158

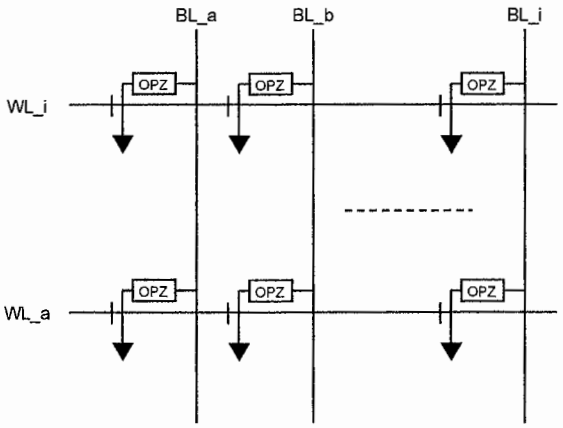


Fig. 6.25. ROM array

6.7 RAM

Also RAM is used to store data. It's not unusual to have more than one RAM with different purposes. One RAM (code RAM) is used to store microcontroller code. This code isn't the firmware needed for the device basic functioning, but debug code or test code. This is the reason why this RAM isn't very big, it could have the size of around 256 instructions (24 or 32 bits depending on the Instruction field length). Sometimes, there is also a smaller RAM, used only for data storage (data RAM) or for fuse monitoring (Chap. 13). Its dimensions can be reduced to 256 bytes. Microcontroller reads data from Code RAM on a custom bus in order to have a higher bandwidth (Fig. 6.2).

As already seen for the ROM, RAMs support a cycle synchronous latency random read access time. All RAMs inputs are sampled with the clock rising edge, except for configuration signals and testmode signal. The outputs are reloaded when the internal asynchronous read cycle is complete and before the clock next rising edge. Configuration signals and testmode signal are considered stable before and after every access.

RAMs are organized on a dual plane architecture in order to increase the performances. Only one plane at a time can be accessed. Every wordline is made up by data cells and dummy cells. For example, the wordline of the Data RAM associated with the instruction set of Fig. 6.22 is composed by eight data cells and three dummy cells. Every plane contains 128 wordlines, one reference wordline and one dummy wordline.

One plane is made up by eight bitline pairs and 128 wordlines (Fig. 6.26).

The cells belonging to the same wordline are accessed together, opening the wordline to read or to write. Every bitline pair is independently sensed and has a dedicated writing driver.

Petitioner Micron Ex-1021, 0159

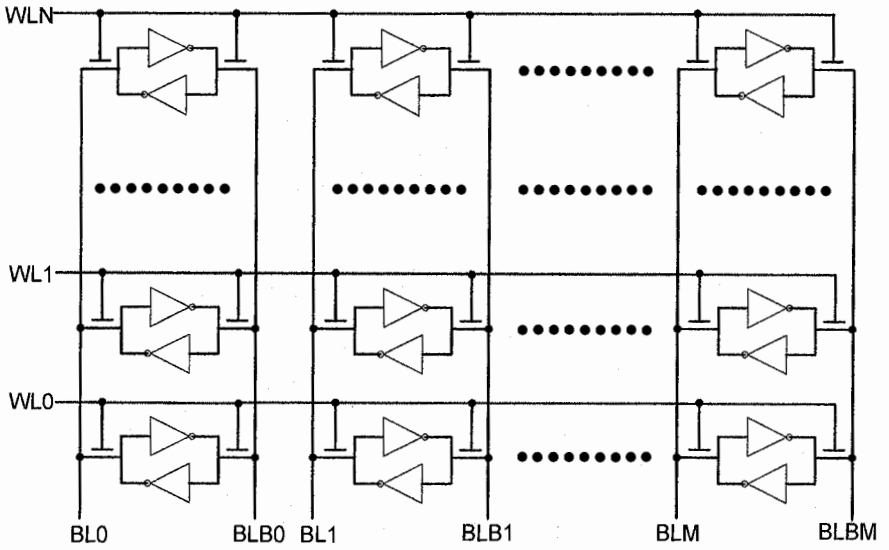


Fig. 6.26. RAM array

6.8 Meta-language

As already said, the microcontroller contains all the NAND necessary algorithms, written as an instructions sequence and decoded by the microcontroller through their opcode. These instructions operate with microcontroller inputs, outputs and internal registers in a very complex way and following an accurate timing. The instructions are presented to the microcontroller as a binary sequence. Since it is not easy to write the algorithms as binary sequences, we use the meta-language.

Meta-language allows the writing of the code in an understandable way, with some rules linked to the microcontroller instruction set. The operations sequence is written in a mnemonic way (Fig. 6.22). Then the code is compiled to control the "grammar", i.e. the variables name and addressing and the instructions properties. Once the grammar is controlled, some scripts convert the instructions sequence in a binary file, that will be used by the microcontroller.

Let's now see how we proceed. First of all, the algorithms structure can be very complex, so that it is recommended to split the structure in different subprograms. Figure 6.27 shows the code structure.

The main file is the core of the structure. The main algorithm controls the link among all the blocks interfacing microcontroller and among all the algorithms.

First of all, the main has an "include" of all the files needed to interface the microcontroller and all the data needed by the algorithms.

Petitioner Micron Ex-1021, 0160

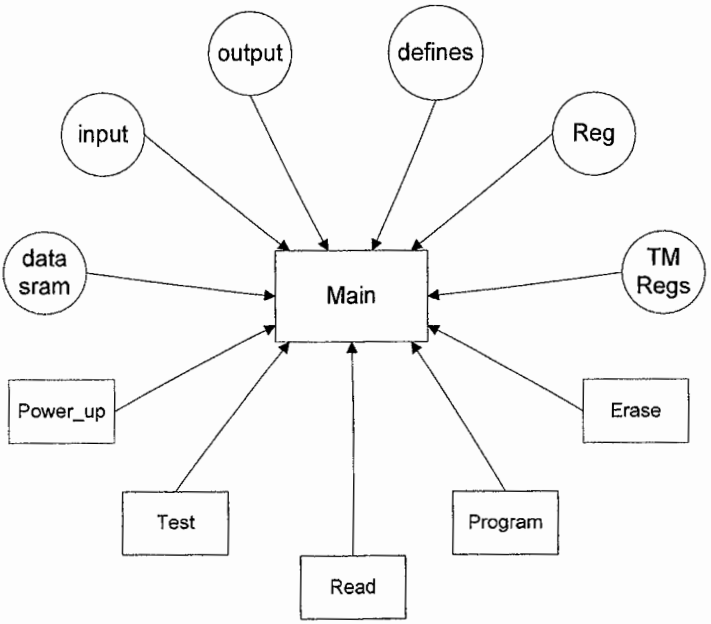


Fig. 6.27. Code structure

Those files are listed below.

- Defines: it contains all the common defines among the algorithms. They could be voltages or timings needed in different point of the algorithm execution. In case of defines used by only one algorithm, they could be put as header of that specific algorithm.
- Out: it defines the microcontroller outputs. The outputs are a fixed number of ports connected with the circuits that microcontroller has to directly move (page buffers, high voltage circuits or decoders). The outputs are divided in ports A and B as explained in Sect. 6.5. Figure 6.28 shows a section of this file. In the figure, a mnemonic name is associated with each output port. The mnemonic name is useful to remember the function of the signal associated with that output, e.g. bit 0 of group 3 is a signal used in the high voltage circuitry to ground V_{blc}. Every time we need to execute a port operation on that specific signal, it is enough to write `hv_vblctognd` and pre-processing will associate the specific output port to that name. A 4-output group is an addressable outputs word. If there are 64 port A outputs and 64 port B outputs, there are 16 port A words and 16 port B words. The grouping in word is useful if we need to execute an operation on all the 4 bits of the word. Let's now suppose that we want to reset (take to 0) all the output ports `hv_vblctognd`, `hv_vblcpepgmtognd`, `hv_gslpmptovblcpgm` and `hv_gslpmptovblcrd`. The store command enables the contemporarily writing of all the four bits, only if they belong to the same word. From this example, we can understand how a careful outputs mapping is very important to optimize the code.

Petitioner Micron Ex-1021, 0161

```
#define  io_datset          "02h 0"
#define  io_datrst          "02h 1"
#define  pb_pbfineset       "02h 2"
#define  pb_pbfinerst       "02h 3"

/*-------------------------------------------------

#define  hv_vblctognd          "03h 0"
#define  hv_vblcpepgmtognd     "03h 1"
#define  hv_gslpmptovblcpgm    "03h 2"
#define  hv_gslpmptovblcrd     "03h 3"

/*-------------------------------------------------

#define  hv_vunseltognd          "04h 0"
#define  hv_passpmptovunselrd    "04h 1"
#define  hv_vseltognd            "04h 2"
#define  hv_vreadtovsel          "04h 3"
```

Fig. 6.28. Outputs map

- In: it contains the microcontroller inputs mapping. The structure is the same as the outputs map. Inputs are less complicated than outputs, since they are not divided in single and dual ports.
- Reg: it contains the microcontroller internal registers mapping. This mapping is very easy, because the registers (for example 8) are used as a support for the operations and don't need a mnemonic name.
- TM Regs: it contains the address of all other registers around the device, such as Test Mode registers. We need to know their address to access them during some particular routines, such as testing. In addition to testmode registers, there are the external registers or some configuration registers or registers for some circuits if required.
- Data Sram: it contains RAM addresses. As mentioned earlier, the data SRAM may contain functional data (e.g. downloaded from fuses) for the algorithms. The file structure is the same as the one in Fig. 6.it contains only the address in bytes. Suppose we want to reserve byte 3 of SRAM to a particular timing `T_bitlineprec`. Then, whenever we access the SRAM during the algorithms at the location `$T_bitlineprec`, we are actually addressing byte 3 of SRAM.

Besides containing all the interfaces, the main file contains all the algorithms necessary for the device. The main code enables the appropriate algorithm based on specific read inputs (typically signals from the CI). The algorithms included in the main are:

- Power up. It is the algorithm necessary during the device's power on. It executes the minimum functional device setting, such as copying information data from the fuses to SRAM.
- Read.
- Program.
- Erase.
- Test. It contains test routines.

Petitioner Micron Ex-1021, 0162

The algorithms are presented as an instructions sequence, written in meta-language so that it is understandable and readable. In Fig. 6.excerpt of the read algorithm.

The instructions are those described in Fig. 6.22. For example, the first row of instructions performs a port A and a port B simultaneously, exploiting the dual port outputs structure. If the dual port structure weren't available, there should have been two clock cycles in order to have those five outputs properly set. The command store executes a particular setting on a whole word of outputs, while set or reset command acts on a particular output bit.

Command `cba($cr_selvprevsen)` puts the address `$cr_selvprevsen` on the control bus, in order to access that particular location that can be found in RAM or registers around the device. Simultaneously with the setting of the address, we can perform a port B operation. Since port B operations can often be performed concurrently with other operations, it is recommended that the output ports mapping connect on port B signals that move frequently and whose concurrency with other operations is very important for the algorithms.

Once the address on the control bus is set, we can perform a read or a write in that location. In the example a writing of the constant `VCLAMP1`, defined in the header of the file (or in the defines file), is executed.

Last remark concerns the instruction `ld (r1, 80h)` i.e. the load of the hexadecimal value 80 in register r1. This instruction brings forward what will come after some time. In fact the following instruction is a wait and does not use the register r1. There are points in the algorithms where we must execute a number of operations simultaneously and quickly; here the concurrent nature of some operations helps.

```
/* pass pmp disabled and BL discharged
store(hv_passhiregenhigh 0,hv_passhiregenlow 0,hv_passpmen 0,
  hv_passpmdisch 0);set(sw_dischs(0));
/* dsg and ssg disabled
reset(hv_vgsltogssl); reset(hv_vgdltogssl);
/* pass pmp discharged and WLs to gnd
set(hv_passpmdisch); reset(rd_biasselect(0));
/* dsg and ssg to gnd
set(hv_gssltognd); set(hv_gdsltognd);
/* sel and unsel switches disabled
cba($cr_selvprevsen); store(hv_vunseltognd
  1,hv_passpmptovunselrd 0,hv_seltognd 1,hv_vreadtovsel 0);
write(VCLAMP1);
/* gsl pmp disabled
/* vblc path disabled
reset(sw_bsls(0)); set(hv_vblctognd);
set(hv_gslpmpdisch);
ld(r1,80h);
wait(r2,t_gsldisch-100 ns);
```

Fig. 6.29. Algorithm in meta-language

Petitioner Micron Ex-1021, 0163

There are also points where we have to wait (typically the waiting times are related to analog circuits); here we can prepare the registers or the address in a proper way for those moments where time becomes a critical parameter.

Once the algorithms have been written in this form and the file has been compiled, the instructions must be translated in a binary file understandable for the microcontroller. Usually, this translation is made with the aid of appropriate scripts.

For example, suppose that at a specific point of the algorithm we executed a test of the second bit of the register 3 and, at the same time, we executed the store of the pattern 0101 on the 4-bit output signals of the group 7. This instruction is the number 13 of Fig. 6.22, whose opcode is shown in Fig. 30.

Here below there is a description of the opcode fields.

- Pos indicates the position of the bit in the byte that we want to test, in the example 010.
- A indicates the internal register (there are eight internal registers) whose bit we want to test, in the example 011.
- Bdata indicates the pattern that we want to write on the four outputs of the group, in the example 0101.
- Baddr indicates which group of four outputs is taken into account (there are 16 groups of four port B outputs), in the example 0111.
- OpB indicates the type of operation we want to execute on port B, i.e. store or set or reset. In the example we want to execute a store, encoded with 10.
- The gray parts in the figure indicate the bits that have no meaning and can be either 0 or 1.

Finally, the operation is encoded as

```
010X011X1000101011110101
```

where the X's indicate bits that can be 0 or 1.

All the algorithms can be translated in this way, in order to have an encoded instruction at each ROM address.

Figure 6.31 shows an example of an extract of "translation".

A file like the one in the figure is very useful during the debugging of algorithms. From the simulations we can know what is the address of the ROM at any given time and the hexadecimal encoding of the associated operation. The figure shows the contents of the ROM in terms of address, encoding operation and mnemonic instruction operation. In this way the file is readable and the debug is simplified.

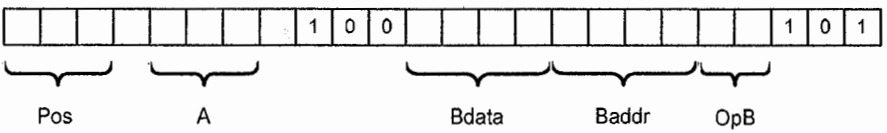


Fig. 6.30. Opcode example

Petitioner Micron Ex-1021, 0164

```
ROMADD  code       instr
-----------------------------------------------
 630    0xBB6E86   IF(OFLG==1) THEN GOTO l_BERLEL_Jdiff0;

 631    0xB87EC4   CALL sub_BEREK_RIPETO15;

 632    0xA5AE85   IF(OFLG==1) THEN GOTO l_BERLEL_ERR;

 633    0xAF3EC4   CALL sub_BEREK_RIPETO4;

 634    0xBA3EC4   CALL sub_BEREK_RIPETO16;

 635    0x600030   LD16(REG0, 0003H);
```

Fig. 6.31. Algorithm in meta-language

```
01111100111111011000101000
11100000001111001011110000
11110000001111100011001100
00111010011111110010001111
10100100011111110110011000
00000000000000000000000000
10100001101010010100000001
10100111101011101000000000
10110010101111111011001000
10110001100011101100100000
11100000001111110010010000
11100100001110000000000001
11100000001110101001111000
00001000001000110011001100
```

Fig. 6.32. ROM content binary conversion

In any case, ROM content consists in a binary file. In Fig. 6.32 there is an example.

The digital simulations use a file of this type [8], while the analog simulations need to convert the ROM content in schematic.

References

1. R. Micheloni et al., "A 4Gb 2b/cell NAND Flash Memory with Embedded 5b BCH ECC for 36MB/s System Read Throughput", IEEE Solid-State Circuits Conference, Digest of Technical Papers, pp. 497–506, 2006.
2. G. Campardo, R. Micheloni, D. Novosel, "VLSI-Design of Non-Volatile Memories". Springer, 2005.
3. P. Cappelletti, C. Golla, P. Olivo, E. Zanoni, Eds., "Flash Memories". Kluwer, 1999, ch. 5.
4. R. Micheloni, A. Marelli, R. Ravasio "Error Correction Codes for Non-Volatile Memories", Springer, 2008.

Petitioner Micron Ex-1021, 0165

5. G. Campardo et al., "An Overview of Flash Architectural Developments", IEEE Proceedings of the, vol. 91, No. 4, pp. 523–536, April 2003.
6. M. Annaratone, "Digital CMOS Circuit Design", Kluwer, 1986.
7. D. Nobunaga et al., "A 50nm 8Gb NAND Flash Memory with 100MB/s Program Throughput and 200MB/s DDR Interface", IEEE Solid-State Circuits Conference, Digest of Technical Papers, pp. 426–625, 2008.
8. D. L. Perry, "VHDL: Programming by Example", McGraw-Hill, 2002.

Petitioner Micron Ex-1021, 0166

7 NAND DDR interface

*Andrea Silvagni**

7.1 NAND Flash evolution: the need for increased bandwidth

Nowadays NAND Flash memory is pervading every type of electronic application. The availability of cheap storage memory, in combination with high densities, makes up the choice in favor of NAND Flash on board of applications traditionally linked to other types of memories (such as EEPROM and NOR) or technologies (such as Hard Disk Drives). Mobile devices, PDA, PC, camcorders, set top boxes, servers, routers, enterprise storage and many more new applications requiring a storage media will have to deal with NAND Flash in the coming years. The memories environment has changed since year 2001 when growth rate for NAND surpassed the 'Moore law' observed for processor growth and predicting a double density every eighteen months. By then, NAND devices with double density have been introduced every year. In addition, MLC devices with 2 bit/cell entered volume production.

This evolution/revolution in semiconductor industry aimed mainly to reduce the cost per MB of the memory itself in order to enable the new applications and open new markets.

In contrast to the technology evolution, if we have a look at the basic functionalities and performances of NAND Flash devices we can observe these haven't improved at a similar rate in the last decade. The new applications triggered by this wave of NAND Flash price trend are now claiming an improvement in performances too. By looking at the roadmap for some of the most common interfaces, such as USB and eMMC, we can observe a planned evolution in the 300 MB/s speed range. PCI Express will also offer a performance improvement. Every application segment (Portable, PC, Mobile etc.) is targeting higher transfer rates and this will reflect in challenges both in HW and SW for NAND based systems.

NAND Flash have already increased page size from 2 to 4 and 8 KB, making available a large amount of data internally with the same latency time but the throughput is limited at the I/O interface as discussed in Sect. 7.1.2.

* Forward Insights, andrea@forward-insights.com

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_7,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0167

7.1.1 Applications driving the NAND high speed interface evolution

Within this section the main applications calling for the evolution of High Speed NAND in the coming years will be explored.

PC applications have become a major demand driver since 2007 when NAND memories made their appearance in the PC platform plugged on PCI express bus with specific modules. Specific initiatives such as ONFI [1] are driving the adoption of NAND on the PC platform through standardization efforts in HW and SW. The Express Card Standard (created by a broad coalition of PCMCIA member companies) is the next generation of PC Card technology used in more than 95% of all notebook computers to add new hardware capabilities. Complete HW/SW are being offered by solution providers to integrate various NAND storage types to PCI-express or to typical enterprise storage interfaces.

Figure 7.1 shows the memory hierarchy level in a PC platform. As we move towards the CPU we have high speed memory used as cache while, at the opposite side, there is the need for high capacity at the lowest cost. Availability of cheap, fast and high density Flash memory is driving changes in the PC memory hierarchy. A well-implemented hierarchy allows a memory system to take advantage from the performance of the fastest component, the cost per bit of the cheapest component, and the energy consumption of the most energy-efficient component. Today, performance, capacity and cost of Flash NAND make it appealing to create a cache level for disk drives (on the platform itself or integrated in Hybrid Disk drives) filling the large existing performance gap between DRAM and Hard Disks. In some cases, NAND Solid State Storage will replace completely HDD (Laptop, enterprise PC).

In cached Non-Volatile assisted systems, special memory management SW provides efficient cache utilization based on page usage patterns over time.

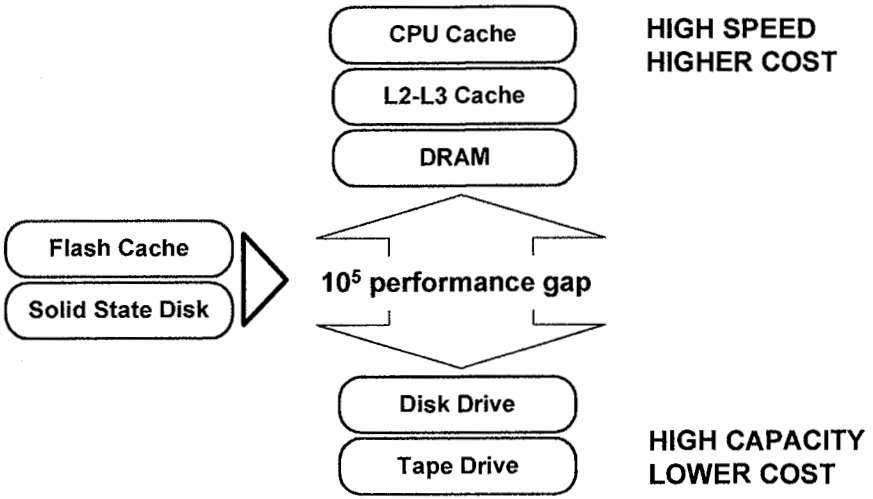


Fig. 7.1. Memory hierarchy in the PC platform and Flash opportunity

Petitioner Micron Ex-1021, 0168

Hybrid Disk drives have made their appearance adding NonVolatile cache (NV Cache) to the hard disk drive. This solution has the same advantages as in cached NV assisted systems with NV cache on the PC mainboard:

- Allows data to be read and written while platter is spun down
- Data in cache persisting when powered down

Solid state disks (SSD) are totally solid state based. The advantages of such a solution are the elimination of weakness and bottleneck of mechanical Hard Disk Drives (HDD):

- Power, HDD consumes 10–15% of the power budget in mobile solutions.
- Slow power up (Spin up times can be 2–5 s).
- Seek and rotational latency is a performance bottleneck in HDD. Latency is dominating the access time in HDD, it can be several milliseconds compared to less than 100 μs in Flash.
- HDD are Fragile, shock prone, low MTBF in mobile systems.

Mobile applications benefit from lower power consumption and hard shock immunity of Flash based storage systems. Besides, they don't need the big amount of data storage that only HDD can provide at a lower cost. Solid State Storage enterprise applications are looking into hybrid solutions offering improved performance and overall life-cycle cost reduction.

7.1.2 Limitations of the asynchronous interface

It is interesting to compare NAND density and performance evolution with that of DRAM chips. The comparison is interesting because of some similarities between the two technologies:

- DRAM drove the technology rush in the past years. NAND is now leading the technology edge.Both memories read data in chunks called pages.
- Both types of memories are part of the PC platform. Currently, DRAM is closer to the CPU serving the speed requirements of the system while NAND is supplying the storage capacity.

Figure 7.2 illustrates, in arbitrary units, the performance and density evolutions for DRAM and NAND. Density followed approximately the Moore's law for both memories. Performance for DRAM has improved 25% average year over year since 2000, while the NAND performance has remained nearly flat. The reason for that is the fact that NAND memory has been relegated for years to pure data storage and only recently it has entered the PC system. As NAND enters into new applications such as PC/mobile systems, it becomes evident that a bottleneck in the system performance is hidden in the NAND interface and must be fixed with a proper roadmap.

Petitioner Micron Ex-1021, 0169

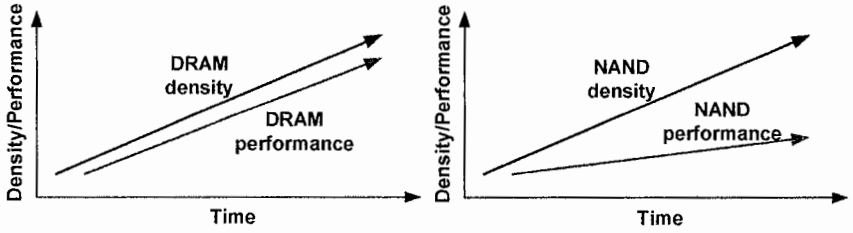


Fig. 7.2. NAND Performance trend versus DRAM

In the left part of Fig. 7.3 it is plotted the access time of NAND legacy (asynchronous) Flash devices having 1, 2 and 4 KB pages.

The access time is split into the *array access time* – that is the time necessary to transfer the data from the NAND cells into the page buffers – and the *transfer time* required to move all the data in the page buffers out of the chip through the legacy interface. It is possible to notice that in 1 KB page devices the transfer time was in balance versus the array read time. This gave the possibility of interleaving read operations either from two different devices or from two planes, hiding de facto the read array time. In 4 KB page device this balance is completely lost.

To restore the balance between read array time and transfer time it is necessary to increase the I/O throughput. On the right hand side of Fig. 7.3, it is shown the situation of 4 KB devices with a DDR interface operating at 66 MHz (DDR133): the desired balance is restored.

The DDR200 (200 MB/s) speed grade keeps the balance between the array and data transfer times even with 8 KB page size. The picture becomes evident in Fig. 7.4 where the internal throughput capability (i.e. page size/internal access time) is plotted versus the page size. It can be shown that the legacy interface throughput was in line with the internal throughput of 1 KB page NAND Flash devices.

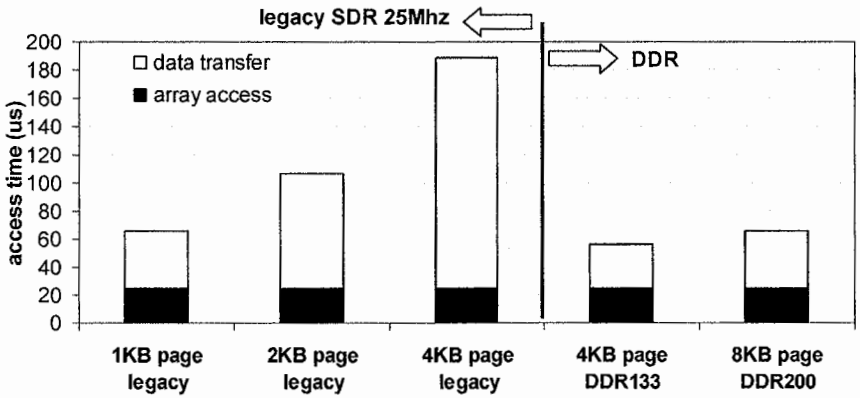


Fig. 7.3. Array time and data transfer breakdown in NAND: Legacy versus DDR

Petitioner Micron Ex-1021, 0170

SLC NAND with 4 KB pages have an internal throughput aligned with DDR133 interface. Moving to MLC 2 bit/cell with 8 KB page, the throughput still remains aligned with DDR133 because of the larger MLC internal access time. 8 KB page SLC devices (not shown) would call for DDR266 interface. DDR400 interface roadmap is aligned with the next generation having page size greater than 8 KB and greater access time. This is in line with system interfaces roadmap evolving towards 300 MB/s rates.

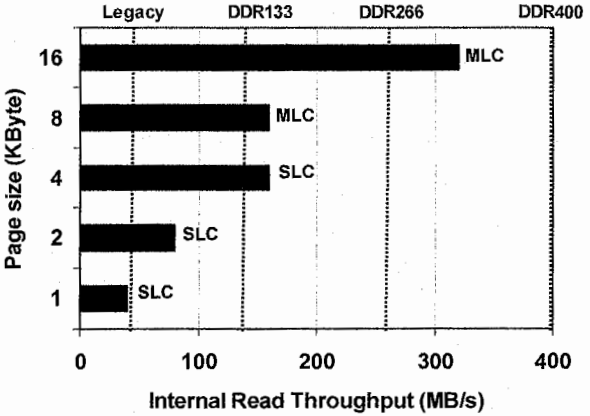


Fig. 7.4. Internal Read throughput versus page size

7.1.3 How to improve the performance of NAND based systems

As already discussed in the previous section, it is desirable to have NAND devices with a balance between the internal throughput (the array read time) and the external throughput (transfer time). This allows to use NAND devices having two internal planes working interleaved, hiding the array time. This solution is quite common because the power issues related to this kind of solution are still manageable inside the chip and optimized versus two separate NAND devices active at the same time. Nevertheless, new systems such as SSD call for an increase of the Read and Write throughput of Flash based systems.

Flash based systems like SSD are made up by several NAND memory devices and one controller. The controller has the primary function to communicate with the NANDs and conveys data from/towards the external interface.

The basic system options are fundamentally one of the following types.

1. Increase the number of dies per channel as shown in Fig. 7.5a. This solution encounters limitations from channel loading. It has the advantage of lower pin count and lower cost in hardware, especially in the controller, but it might not satisfy the requirements of write throughput for the system.

Petitioner Micron Ex-1021, 0171

2. Increase the number of channels (Fig 7.5b). This solution shift the problems inside the memory controller which has to manage the parallel data flows coming from the memory channels. It is an expensive solution because, for example, the controllers have to manage also ECC and have the need of dedicated SRAM/ register file. The advantage of this solution is that it is scalable and flexible and allows to reach very high Read/Write throughput.

In every case there is the problem to keep under control the power and the channel limitations with careful interface design considerations. Multiple channels technique, as shown in Fig. 7.5b, is common in SSD.

To overcome the already named power limitations, some other types of configurations have been proposed together with a special interface in which group of devices are connected through a ring topology with the controller and Address/ Data is passed through the ring from device to device in a point-to-point connection. Each device adds one clock cycle of latency which is not an issue in mass storage applications. Those kinds of solutions will be treated in the last section together with some power and performance considerations.

Building a successful Solid State Storage means to deal with many aspects related to NAND Flash and systems. Here, we will deal mainly with the I/O bottleneck problem which first must be solved by a proper interface roadmap.

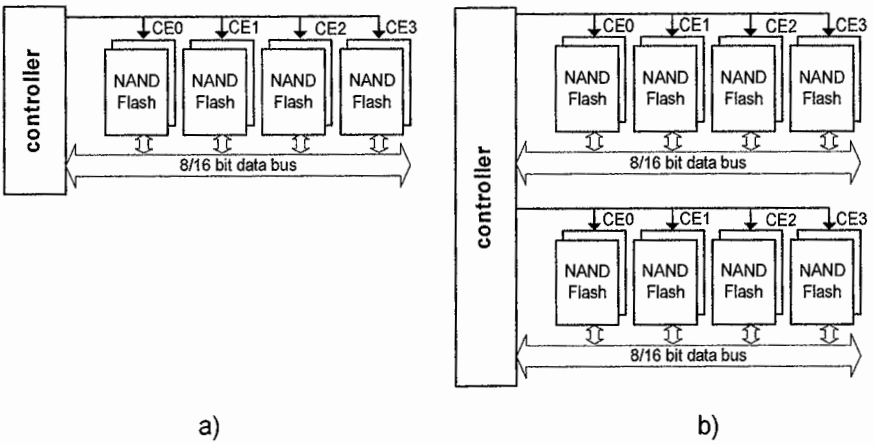


Fig. 7.5. SSD system enhancement: (a) increased number of dies per channel (b) increased number of channels

7.1.4 Adopting DDR protocol

First step toward improvement of the legacy NAND interface is Double Data Rate (DDR) interface. The same way DRAMs evolved starting from year 2000 is now followed in the NAND arena.

Petitioner Micron Ex-1021, 0172

Table 7.1 summarizes DRAM DDR interface generations along with the year of introduction [6]. Table 7.1 also lists the main architectural steps associated with interface evolution.

Table 7.1. DDR comparison table

GEN	Year	Rate Max	I/O	ODT	Strobe	Prefetch	Package
		MT/s	V				
SDR		133	3.3	NO	NO	1	TSOP
DDR	2004	400	2.5	NO	Single ended	2	TSOP/BGA
DDR2	2006	800	1.8	YES	Diff/single	4	BGA
DDR3	2008	1,600	1.5	YES	Differential	8	BGA

High speed NAND have seen the introduction of DDR interface in year 2008. The challenge in the coming years will be the standardization of the interface among the vendors. Two solutions have been proposed as first generation of DDR NAND:

1. ONFI organization [1] has proposed the introduction of a clock and data strobe, ready for an evolutionary path in the DRAM style.
2. Samsung has proposed the Toggle mode NAND [3] where only data strobe has been introduced.

All the industry claims an interface Roadmap with a first generation at 133–266 MB/s, a second generation with more than 400 MB/s throughput. The interface roadmap will be kept with LVTTL bus driving style as long as possible in order to ease integration, but some design tricks will have to be introduced in order to sustain the claimed bandwidth. This will include the proper scaling of interface voltage, the use of a specific termination type, On-Die Terminations, differential strobes and, going beyond, synchronization circuit. Finally, it will include DLL/PLL and the change to a SSTL class of terminated bus.

The DDR protocol diagrams are sketched in Fig. 7.7 along with the principal timings [6, 7]. A Synchronous clock must be provided to the memory chips (not needed in toggle-mode interface). Bidirectional Data bus DQ is driven with a frequency equal to that of the system clock i.e. every clock edge. The data throughput is therefore doubled compared to that of a Single data rate system at the same clock frequency. Data strobe signal is driven with the same timing and characteristics of the DQ lines with the purpose of using it as the data capture signal at the receiver.

Data Strobe utilization allows to build a system where the flight time skew of the data versus the clock signal is neglected as the Data Strobes act as a clock with minimum skew. Systems scalability will benefit from the approach since the DQS load will match that of the DQ lines, keeping the same timings, e.g. when changing the load by adding more dies on the channel.

Petitioner Micron Ex-1021, 0173

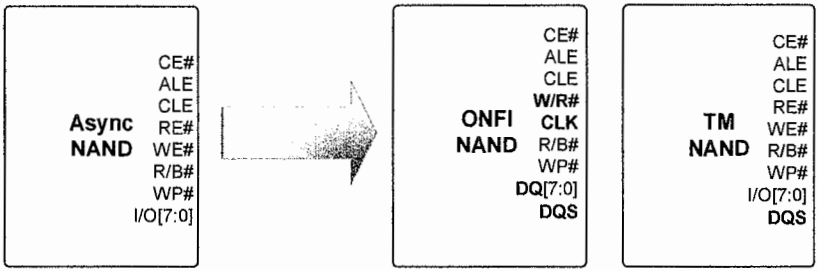


Fig. 7.6. Legacy NAND versus ONFI 2 and Toggle-Mode Synchronous NAND Interface

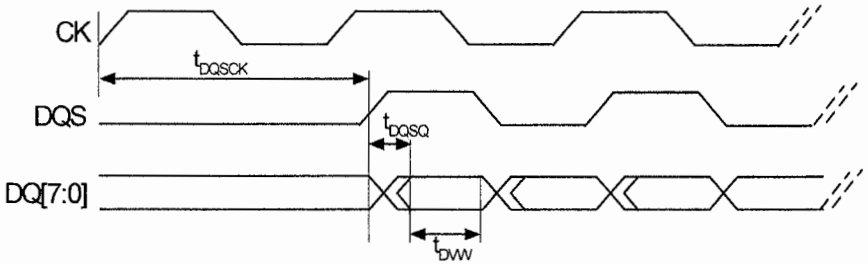


Fig. 7.7. DDR interface diagram

Open NAND Flash Interface (ONFI) was formed in 2006 to drive standards for the NAND raw devices. ONFI 2 defined the source synchronous interface to increase the bandwidth of each I/O channel while adding only one pin, DQS. The protocol is backwards compatible to asynchronous NAND, reducing or eliminating firmware changes for command set when used with asynchronous NAND interface controllers that only support the asynchronous NAND interface. The protocol does not foresee the need for DLL (Delay Locked Loop) circuits in the NAND Flash devices. ONFI 2.1 improved the I/O channel performance up to 200 MB/s. The work in ONFI 3 is targeting 400 MB/s. High throughput enhancements are achieved by leveraging on output impedance settings, short channel, on-die terminations and finally differential clocks and data strobes.

Pin out differences between legacy and ONFI 2.0 DDR NAND are sketched in Fig. 7.6:

- WE# becomes a fast CLK.
- RE# handles data direction by becoming W/R# (Write/Read#).
- I/O[7:0] renamed to DQ[7:0] (name change only, functionally identical).
- DQS, a new bi-directional signal, is enabled.

Set Features command enables the source synchronous interface and other configuration modes such as multiple output drive strength settings, so that many NAND devices can share the I/O bus while maintaining the data integrity.

Petitioner Micron Ex-1021, 0174

Toggle mode adds DQS data strobe signals; RE is used to trigger the read cycle as done in asynchronous interface; DQS is used to strobe the data on both edges.

7.1.5 High density systems: density, power and performance

Today, SSDs are offered in densities up to 64–128 GB and higher. To satisfy Enterprise requirements of endurance and quality, producers are forced to use SLC devices or MLC with trailing-edge technology. As of today, a 32 GB SSD can be built using eight packages (e.g. TSOP), each containing four 16 Gbit-sub 50 nm generation SLC NAND. The system can be built using four independent data channels. This means that two TSOP packages are connected to a single channel with a total of eight dies hanging on the same data bus which is loaded roughly by 50 pF for each data line.

Some considerations are fundamental when designing a system:

- Which is the minimum needed configuration in terms of channels number and which is the operating frequency to achieve the required throughput?
- How much power is dissipated in the system? Is it sustainable?

Due to the channel capacitance, each time a single NAND die is being written or read, the entire capacitance (that can be as high has 50 pF in the example above) of the data lines is driven.

I/O power consumption in a DDR system can be derived as [26]:

$$P = 9 \cdot \eta \cdot f \cdot C \cdot V^2 \tag{7.1}$$

where η is the bit activity ratio, f is the DDR frequency, C is the capacitance of a single line and V is the supply voltage of the interface.

For the system described $\eta = 0.5$, $C = 50$ pF and $V = 3.3$ V. We can derive the power consumption as a direct function of throughput (Fig. 7.8).

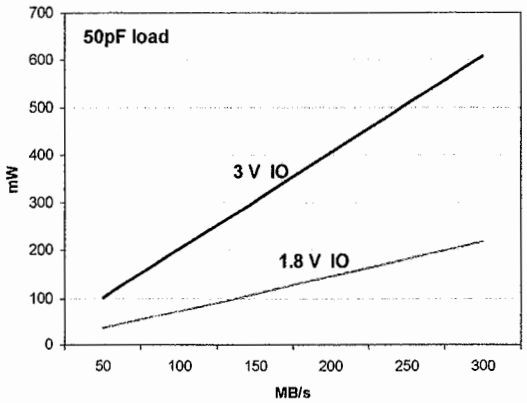


Fig. 7.8. I/O power as a function of channel throughput

Petitioner Micron Ex-1021, 0175

An immediate consideration is the possibility to scale the I/O interface voltage. This solution is fundamental for the new NAND generations and will go hand by hand with the operating frequency increase in the DRAM style. Anyhow, power considerations remain the basics for the definition of the future system architectures [6, 22].

Finally, how is it possible to increase the storage capacity? Figure 7.9 shows a typical diagram of memory system which communicates with the main system through a dedicated interconnect channel. In case of current SSD architectures, the controller communicates with memory chips by means of several channels and the controller system handles memory management locally. The High speed connection can be a SATA or a PCI-express bus.

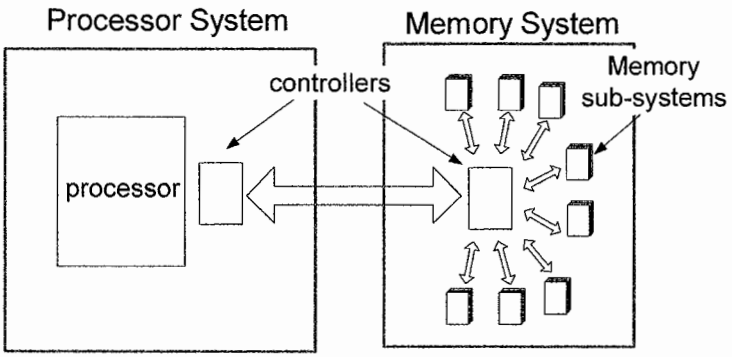


Fig. 7.9. Architectural change evolution

To increase the dimension of such a system it is possible to:

- Increase the number of channels inside the storage system
- Increase the number of Memory dies connected to a single channel
- Increase the number of Memory systems (e.g. add SSDs multiplexing the PCIe bus)

Increasing the number of channels is the primary option to increase performance of the system but it costs in terms of hardware. Increasing the number of memories, instead, encounters limitations due to the I/O capacitance. The system solution would be to add more SSDs in the system.

To partially overcome the above limitation it is possible to exploit modern System in Package technologies building memory systems where the memory die communicates internally on local (to the package) bus. The local bus can be driven by normal CMOS buffers instead of OCD ESD-compliant structures. By making use of simplified ESD structures it is possible to reduce further the bus capacitance. A special die of the stack will take care of selecting the data from the addressed die and connect it to the external nodes using standard off chip driving structures.

Figure 7.10 depicts a system in which memory chips are stacked and connected using an unspecified Local Interconnect Bus. The Interface Chip provides data translation from local interconnect bus to the external bus by means of a standard

Petitioner Micron Ex-1021, 0176

off chip driver (OCD). By means of advanced packaging technologies it is also possible to think the local interconnect bus as a slower Single Data Rate channel to be multiplexed onto external DDR bus by the Interface Chip. Interface chip can be built using a different technology from the one used for memories, thus optimizing the cost of the die. Some dedicated test pads on the memory chip will be anyway required for wafer testing.

This kind of solution can make use of advanced packaging technologies such as *Through Silicon Vias* (TSV) interconnections [23] as shown in Fig. 7.11. TSV solution creates interesting opportunities for stacking thanks to its relatively low parasitic capacitance and its flexibility.

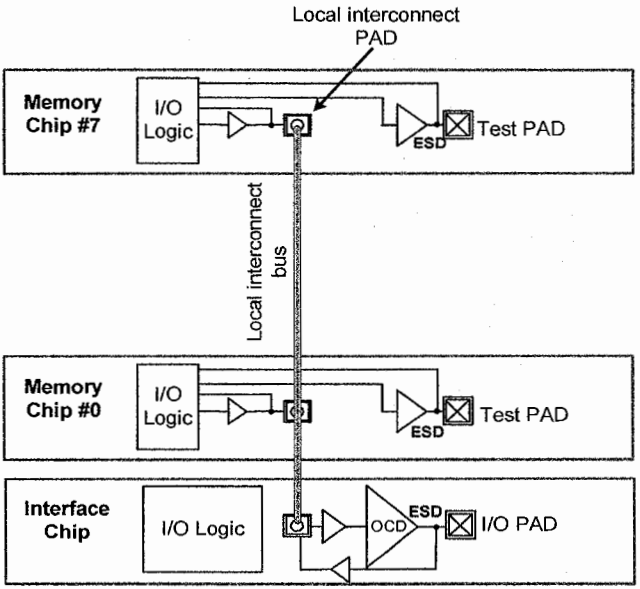


Fig. 7.10. Local interconnect bus architecture

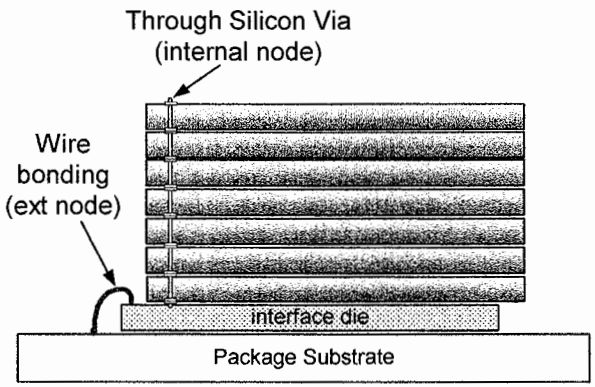


Fig. 7.11. Multiple stacking with local interconnect bus TSV Technology

Petitioner Micron Ex-1021, 0177

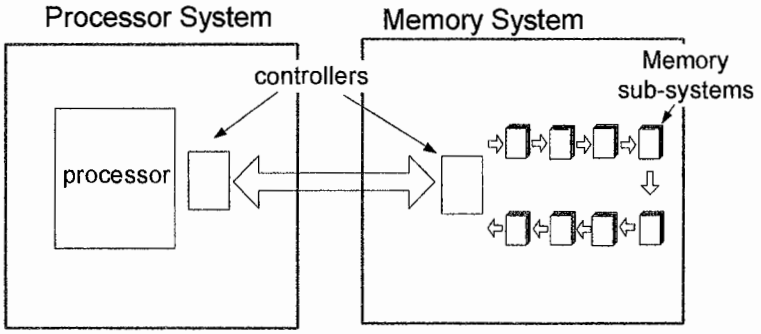


Fig. 7.12. Serial ring topology

In the direction of system density increase, another option would be adopting a serial ring link such as shown in Fig. 7.12.

By means of this kind of topology it is possible to exploit the advantages of unidirectional, point-to-point, daisy-chain cascade supporting many dies on a ring without bus signal integrity problems caused by high capacitance as in multi-drop architecture. The device address is assigned on initialization protocol and each memory package responds to that address. The advantage of this situation is that the off chip driver is optimized for a single standard load, obtaining also a reduction in die size and leakage. I/O power dissipation is related to the position of the addressed die in the ring but simpler I/O design also means lower power dissipation inside the die.

7.2 Basic input output circuit design

In this section we will explore the design issues in the NAND Flash Input/Output interface and its evolution. The section starts with a survey on the I/O interface and output buffer design problems in standard asynchronous NAND products available in the market. Next in the chapter, the design issues in output circuits design for improved high-speed NAND will be treated.

7.2.1 I/O circuits for asynchronous NAND

In the NAND Flash memory offer, the asynchronous products generally have a read cycle of 25 ns (or 40 MHz operation). This is certainly not challenging for the modern technology nodes and designers utilize output buffer circuitry with well known circuital techniques, without worrying too much about time delays and impedance matching issues. Design, instead, focuses on other aspects, such as die size or switching noise issues. Other types of non volatile memory, such as NOR Flash products, normally operate well beyond the 40 MHz frequency with a full CMOS interface. In this section we will review the commonly used techniques

Petitioner Micron Ex-1021, 0178

and tricks necessary to obtain a competitive CMOS working solution as a basis to move forward into the high speed operation.

In Table 7.2 the design challenges that must be taken into account when designing an output buffer are summarized; the area of design that is mainly impacted by each challenge is also highlighted.

Table 7.2. Output buffer design challenges

Challenge	Design area
High Output load capacitance	–Driver and pre-driver sizing –Crowbar current –Output resistance control –Slew rate control
Package electrical constraints	–SSN –Slew rate control
Small area	–Layout
ESD requirements	–Layout, technology
EMC requirements	–Slew rate control
Voltage domain change	–Level shifting
AC-DC performance and logic functions	–Propagation delay –High impedance state

7.2.2 Basic CMOS output buffer design

Let's first explore how the above points in Table 7.2 affect an Output driver stage design by reviewing the standard CMOS I/O design techniques [4, 5, 7].

Let's consider Fig. 7.13 where a simplified structure of CMOS Output Buffer is sketched.

A buffer is generally represented by a big CMOS inverter. The input node of the buffer scheme is connected to the internal data bus and the output node is connected to the output load generally represented by a capacitor. The output buffer, in its simpler form, is represented by a linear model composed by a voltage generator driving '1' or '0' plus a resistance representing the output impedance.

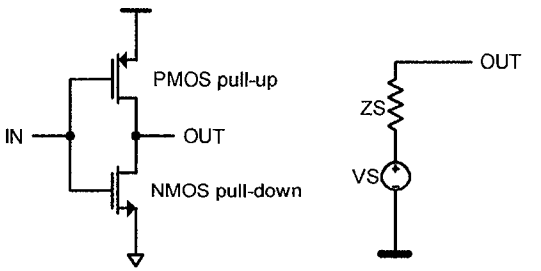


Fig. 7.13. Simple output buffer schematic

Petitioner Micron Ex-1021, 0179

Actually, the output characteristic is not linear and is also subjected to all variations of process and temperature connected with the MOS devices. In Fig. 7.14 the operation of the buffer is shown. Let's suppose that at the initial point the output is charged to VDD and the input of the buffer is switched from GND to VDD. The operating point initially moves from one curve to the other as VGS raises from 0 V to VDD and finally moves along the curve shown with VGS = VDD. The load is discharged and the operating point moves along the N-MOS characteristic curve until VDS = 0 V. It is possible to observe that the impedance varies from a very high value at the beginning, when the NMOS device operates in saturation, to a low ohmic value when the device is in the linear zone (e.g. when VDS < VDD-V_{TH}). It is important to recall that devices operate most of the time in the linear zone (where VOUT = VDD/2) and the resistance varies a lot during the switching time: a situation not desirable when the matching of impedance to a transmission line is necessary. In particular, the operation in saturation should be avoided because the output impedance would be very high.

The linearization of the output buffer is one fundamental point when dealing with high speed as we will see later in Sect. 7.3.

For asynchronous NAND, the impedance matching problem is usually shifted at the system level design by introducing proper linearization techniques (if needed). System designers have to deal with products having very different impedance values from vendor to vendor and with high spread with process voltage and temperature (PVT) variations. Nevertheless, it is important to know that a correct circuit design can ease the system level integration and make the component easily manageable.

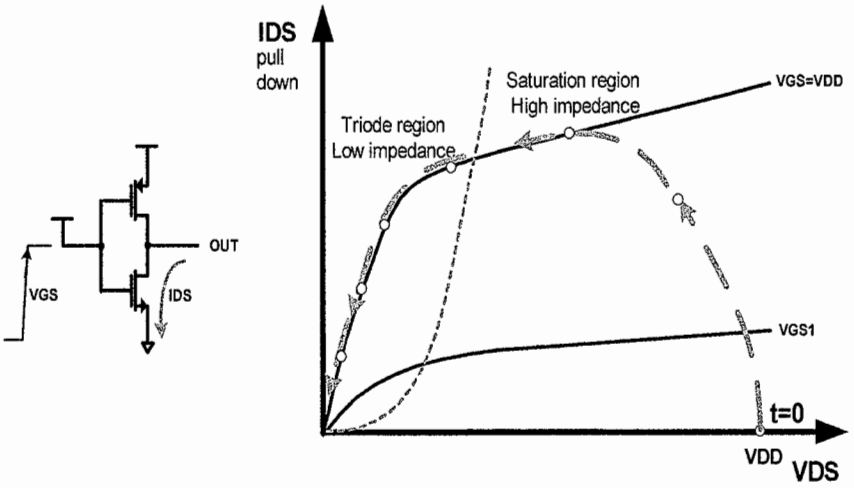


Fig. 7.14. Output buffer operation

Petitioner Micron Ex-1021, 0180

7.2.3 Driving high capacitive loads and noise: slew rate control

In typical systems the NAND output buffer is used to drive large capacitive loads, in the range of 50–100 pF. In this situation the output capacitance transition is very long compared to the buffer switching time. The buffer conductance is usually made very large to reduce the charge/discharge time and match the specifications.

The memory data bus can be byte or word wide and the capacitive loading current provided through the chip is multiplied by the number of switching data bits. The inductances of the bonding wires in the package (Fig. 7.15) – that can be as high as 5–10 nH in TSOP packages – play a negative role in noise margin because they generate bounces on internal supply lines that could affect the functionalities of other circuits in the core region. This effect is often called *Simultaneous Switching Noise* (SSN) and will be treated more in detail later.

A simple model, as shown in Fig. 7.16, can be used to understand that a fast switching of the push pull is detrimental for noise [9].

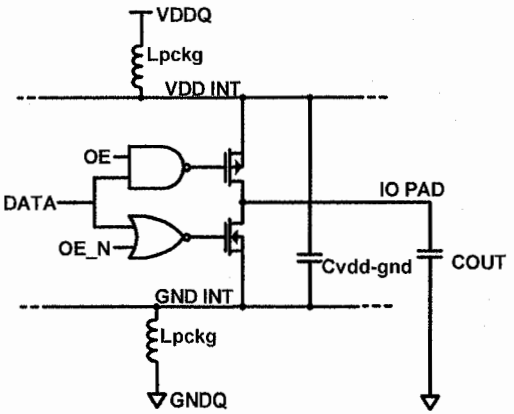


Fig. 7.15. RLC Parasitics in output buffer simple model

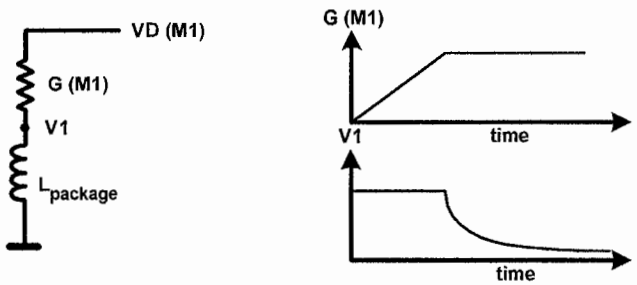


Fig. 7.16. Simple model to evaluate voltage noise

Petitioner Micron Ex-1021, 0181

The noise generated on node V1 can be calculated as:

$$VN^{\max} = VD \cdot \frac{1}{1 + \frac{t_r}{L \cdot G}} \tag{7.2}$$

where t_r is the switching time, G is the maximum conductance of the transistor (e.g. when V_{GS} = VDD for N-NMOS). For example, if we substitute L = 5 nH and G = 1/10 Ohm, t_r = 0.5 ns in the formula, we have a noise as high as ½ VD.

The conductance G can be very high and is subjected to big variations with temperature. Therefore, some control of the transistor turn-on should be done.

To slow down the turn-on of the push–pull driver, appropriate sizing of the pre-driver circuit (that also will contribute in the noise generation) is required. As a consequence, if the gate of PMOS and NMOS are slowed down, we face the crowbar current in the output stage. Crowbar occurs when both PMOS and NMOS are on at the same time similarly to what happens in CMOS digital circuits, with the difference that, here, the currents are much higher due to the devices dimensions. This situation must generally be avoided because it contributes to power dissipation and noise. To avoid this situation the buffer structure of Fig. 7.17 can be adopted [4, 5]. With this configuration the transistor, say the pull-up, is switched-off before the pull down is turned on.

NAND and NOR structures can be tuned to obtain a fast switching-off and a proper switching-on time for the output driver stage. In the figure it is also shown the output enable signal OE that is used to turn the output stage in high impedance state in order to leave the data bus available for other driving devices.

As we will see later, in high-speed circuits, a certain amount of crowbar current must be traded off in favour of speed and control of output impedance.

Another important design constraint is to control slew rate of the output driver but also of the pre-driver which is, in addition, a big contributor for noise and should not be forgotten. In asynchronous devices, the slew rate is generally controlled by acting on the pre-driver, so that the pull-up and pull-down transistors are gradually switched on/off [5, 17].

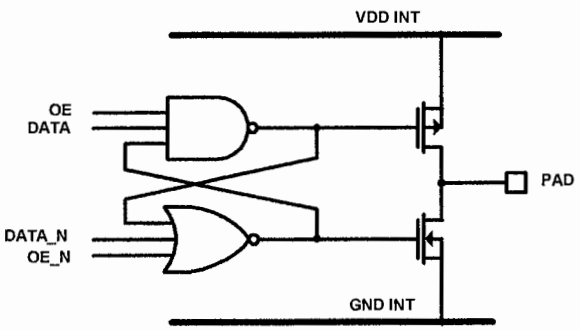


Fig. 7.17. Pre-driver to avoid crowbar in push pull stage

Petitioner Micron Ex-1021, 0182

Generally, this is optimized in the slow corner and the result is a big variation with process/voltage/temperature (PVT). The pre-driver *RC* output constant must be much smaller than the data window, otherwise there is a risk to have a data dependent jitter.

Other more sophisticated techniques to control the slew rates consist in controlling the current in the pre-driver stage. This technique allows also to control the slew rate in PVT conditions by setting appropriate current value depending on the PVT corner properly detected by dedicated circuit [18].

It must be also noted that slew rate control is beneficial to reduce electromagnetic emissions from the memory which is usually the main responsible for this kind of disturbance in the systems.

If a wide data bus is present, it could be beneficial to consider skewing the output enable by a proper small delay and consequently spreading in time the current requests.

7.2.4 Simultaneous Switching Noise (SSN)

As the data rate increases, the data valid window (i.e. the time data must be stable at the output of the memory in order to be captured by the controller) is reduced correspondingly. As a consequence, all effects contributing to reduce margin become more and more important [22]. The memory is affected by noise while generating the data; this will narrow further the available bit time. One of the main responsible for data window margins degradation is the simultaneous switching noise (SSN) [5, 7, 14–16]. SSN is one of the biggest noise effects impacting the data margin and resulting in jitter increase.

SSN is an inductive noise caused by several outputs switching at the same time. One single buffer could have a good transient behaviour, but, when all the data buffer are switching at the same time, the data AC behaviour could be corrupted. The problem is serious in output buffer memory design because of two effects:

- Jitter and signal bounces are increased and data window margin is reduced.
- The noise generated could affect other circuits, especially analog or memory sense amplifiers, reducing operating margin or creating systematic non-working windows.

With a large capacitive load, a large current is requested to charge the load and the power network must supply that current. The current flows in inductances, typically in the bonding wires or leads of the package, and the resulting noise is introduced on to the power and ground supplies. This noise is transferred to the output and the output AC characteristics are affected.

The simultaneous switching noise is determined, in principle, by the following equation:

$$V_{SSN} = N \cdot L \cdot \frac{\partial I}{\partial t} \tag{7.3}$$

where N is the number of switching outputs, L the equivalent inductance in which current must flow, and I the current per driver.

Petitioner Micron Ex-1021, 0183

Since this mechanism is dependent on the number of output switching, N, this makes the noise dependent also on the data sequence.

To deal correctly with SSN it is necessary to understand the complete signal current paths in the memory. In Fig. 7.18 a complete path is shown. Local metal resistances are omitted but they should be evaluated as possible sources of interference.

It is straightforward to understand that the problem is really connected with the package. When TSOP packages are used, very long bonding wires can be present leading to high inductance values. Moving to higher data rates requires to leave such types of package in favour of more controllable Ball Grid Arrays.

Table 7.3 summarizes the main causes of SSN and the possible design countermeasures.

Table 7.3. SSN causes and countermeasures

Cause	Implementation
Avoid crow-bar current in the output stage	Control pull-up and pull down gate
Slow down edges	Apply slew rate control techniques
Power supply limitations and RLC parasitics	Add on-die supply capacitance. The capacitance added between VDD and Ground will act providing charge, when needed, that will not flow into the power inductance.
Care of internal power lines	If it is possible, choose a proper assignment of I/O pins and power supplies separating the current paths. The degree of freedom can be restricted by specification assigned pinout but, if possible, increase the number of power supply pins and place them near the I/O pins, thus reducing also the Metal resistances. Take into account that adjacent VDD and GND will benefit from mutual inductance since the currents are generally flowing in opposite direction.
Package choices	Try to obtain the shortest bonding wires or traces. Use double bonding. Use BGA-type package

The most important point is controlling the output slew rate. Other recommendations are useful to avoid that noise to be coupled to other circuits (mainly analog) causing malfunction or latches to switch state falsely. Triple well technology could be used to separate noisy circuits [4]. Separate power supplies for the core logic and the I/Os can help limiting negative effects of SSN.

Petitioner Micron Ex-1021, 0184

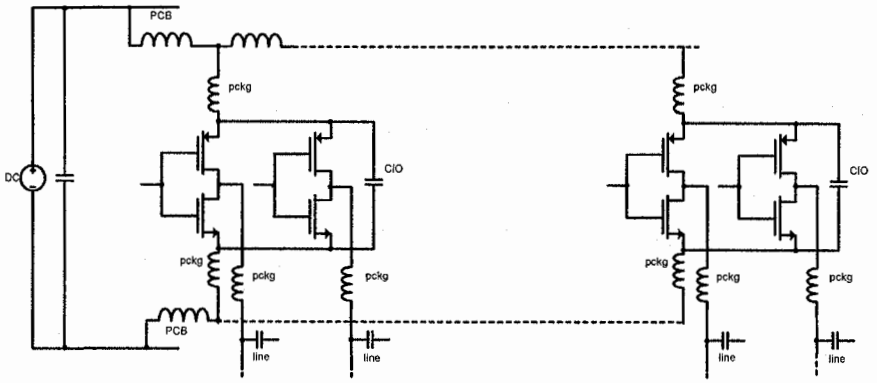


Fig. 7.18. Model example used to evaluate SSN

7.3 High speed NAND I/O design

In this section the design issues related to an output stage of a high-speed NAND are analyzed. In Table 7.4, those challenges are summarized and the related design areas are described. As performance requirements increase, some of the topics become more and more relevant.

In the following sub-sections the basic challenges listed in Table 7.4 will be analyzed and the design techniques necessary to cope with the increasing demand in performance will be treated.

Table 7.4. High Speed output stage design challenges

Challenge/constraint	Area
Operating voltage 1.8V or lower	Devices (technology) optimization
Output impedance control	Output driver design: linearization, on-die Terminations
Max allowed capacitance	Technology, topology, layout, ESD
Slew rate control	Predriver design
Duty cycle distortion, jitter	Signal chain, voltage domain change, decoupling capacitors
Fly-by delay	Data path optimization
Resonance	Package parasites, decoupling capacitors
Channel limitations	Terminations, equalization, pre-emphasis
ESD	Technology, layout
Power consumption	Interface, termination styles

Petitioner Micron Ex-1021, 0185

7.3.1 High speed output buffer circuits

Output buffer in high speed signal transmission is often named Off-Chip Driver (OCD). In addition to the task of being the interface circuit between inside and outside, OCD in high speed memories has to accomplish several additional tasks.

- Translate data flow between single data rate and DDR worlds.
- Voltage domain change. The core of the memory could operate at a different voltage level than the I/O interface and the data signals have the need to be shifted from the core level to the interface voltage.
- Provide the AC/DC requirements such as V_{OL}/V_{OH}, slew rate or impedance matching.
- Provide the On-Die-Termination (ODT).
- ESD protection.

Various types of OCD output stages are used in memory design depending on the interface type and speed, as shown in Fig. 7.19.

The single ended CMOS buffer is widely used in DDR designs and we focus mainly on this kind of structure because it is widely used in almost all commodities DRAM memories. Differential drivers are often used to produce clocks and data strobes.

The use of differential output drivers and receivers for critical signals such as strobes and clocks help to keep timing margins. A differential output consists of a pair of signals that are always switching opposite in phase (odd mode). A differential receiver is a circuit that triggers at the crossing of two signals. This eliminates common-mode noise and increase the signal quality dramatically. Differentially routed transmission lines are also more immune to coupled noise and non-ideal return paths because the odd mode sets up a virtual ground between the signals.

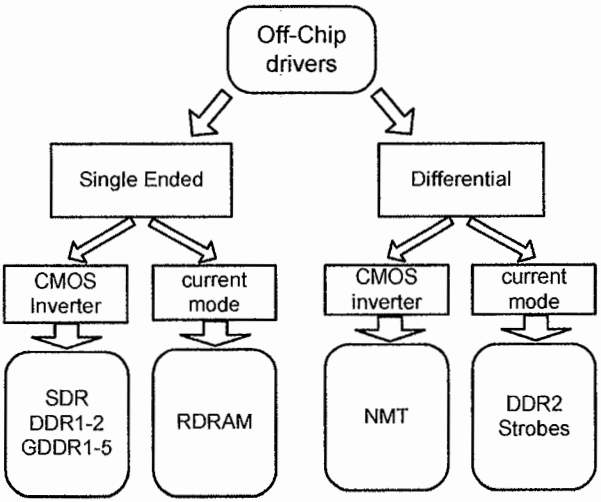


Fig. 7.19. A OCD taxonomy

Petitioner Micron Ex-1021, 0186

7.3.2 Double data rate OCD

A DDR OCD is a synchronous output buffer. In synchronous systems, OCD include a register stage used to synchronize the output with the internal databus. In DDR design a block named *serializer* is included in the buffer design as shown in Fig. 7.20. Serializer block performs the Single Data Rate (SDR) to DDR conversion.

The Serializer block receives 2n data at a given rate R (SDR) and multiplexes these data onto an internal line at a higher rate 2R (DDR). For example, two 8-bit data at a rate of 25 MB/s are multiplexed onto a 50 MB/s DDR bus by using a 50 MHz clock.

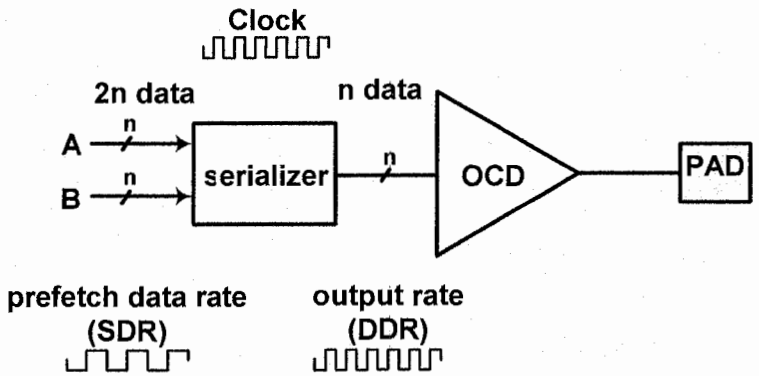


Fig. 7.20. OCD schematic block diagram

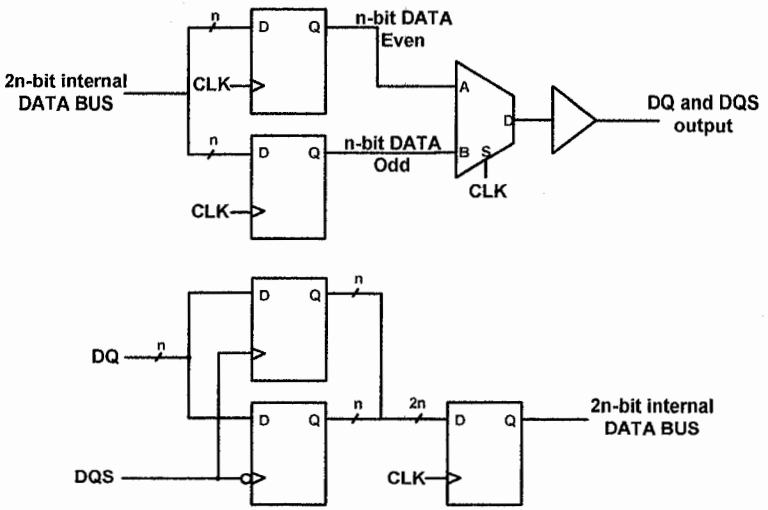


Fig. 7.21. Serializer/deserializer 2n prefetch Read and Write block diagram

Petitioner Micron Ex-1021, 0187

We should highlight that the OCD is operating at a frequency higher than in other memory blocks. For example, the data entering OCD is running at a frequency that is halved.

Serializer block converts from Single Data Rate world to the Double Data rate world by means of the structure shown in Fig. 7.21. The same figure shows the deserializer block needed to convert data from DDR bus into SDR internal bus by using data strobe DQS as clock to register the DQ data input.

In DDR design the data rate is doubled vs. a single edge design at the same system frequency. Therefore, since we have to deal with smaller delays in OCD, it is necessary to take more countermeasures in designing the block to avoid jitter eating almost all margins.

Variations to the structure can be done in order to obtain greater data rate at the output maintaining the same frequency in the internal buses. This is achieved through wider prefetch of data from the memory and multiplexing more data bus onto the DDR bus.

7.3.3 OCD linearity: push–pull and open-drain configurations

It is of primary importance to offer a linear behaviour of the output characteristics in order to obtain a good behaviour in the system signal integrity, especially in absence of special termination techniques at system level (e.g. SSTL logic). Linearity of the OCD is thus important to achieve a stable matching impedance to the external lines. Figure 7.22 illustrates the problem when interfacing a CMOS buffer structure with a transmission line.

Assuming that the transmission line is at VDD from the previous state, when the pull down is switched on, VDS makes the transition to zero. In this condition the current is limited to VDD/Z0 where Z0 is the impedance of the line seen by the driver.

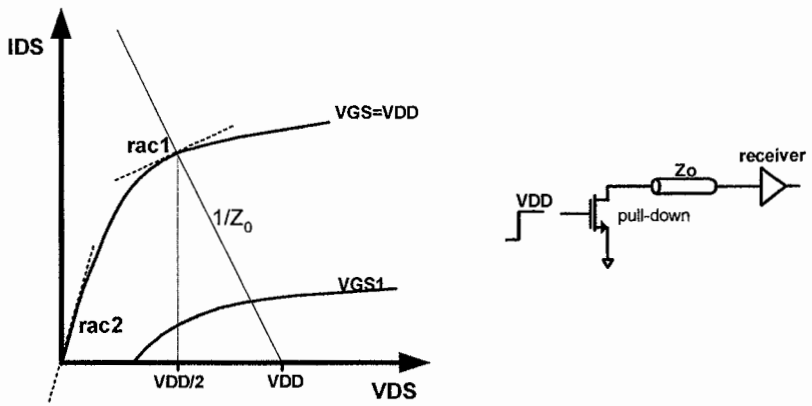


Fig. 7.22. Output impedance of driver (NMOS) and load line

Petitioner Micron Ex-1021, 0188

The result is a load line with slope 1/Z0. In order to reach the target of launching the signal with a value of VDD/2 – so that the signal at the end of line is reflected, reaching full VDD swing – it is necessary that the intersection of the transfer characteristic with VGS = VDD crosses 1/Zo at VDS = VDD/2. In this point the signal is driven, and then the wave is reflected back and returns to the output node. VDS is then decreased to a value between VDD/2 and zero. Here the resistance rac2 shown from the driver to the incoming wave is very different from rac1. The best situation is when rac2 equals Z0 to reduce reflections. Because of different data patterns it is clear that intermediate situations can occur, therefore producing a certain amount of *Inter Symbol Interference* (ISI). The best situation is reached when rac1 and rac2 are close, i.e. when the driver operates closer to linear region. In real world the characteristics of the PMOS and NMOS output transistors are given by technology and cannot be changed by designers. A solution to obtain fairly linear characteristics can be found by using linearization serial resistors as shown in Fig. 7.23. Resistors *R* added in series to both pull-up and pull-down help reaching the target output impedance stability.

Figure 7.23 shows two configurations where serial resistors are added to PMOS device and NMOS device or where a single common resistance is added to NMOS and PMOS devices. The second alternative is preferable since it reduces the parasitic capacitance at the output node, even if this solution doesn't allow optimizing PMOS versus NMOS resistance. Moreover, the best trade-off is obtained by playing with dimensions of NMOS and PMOS.

The example in Fig. 7.24a shows the effect of the linearization resistance on the pull-down output characteristics. Moreover, in Fig. 7.24b two choices for resistor pull-up and pull-down sizes are shown. In one case big MOS size is coupled with resistors close to the 50 Ω transmission line; the second case has smaller MOS and smaller resistance. It is shown that both the solutions are close to the target impedance at VDD/2 but they are very different at small VDS. It must be noticed that the first solution has a bigger capacitance than the second one and this could lead to problems at system level.

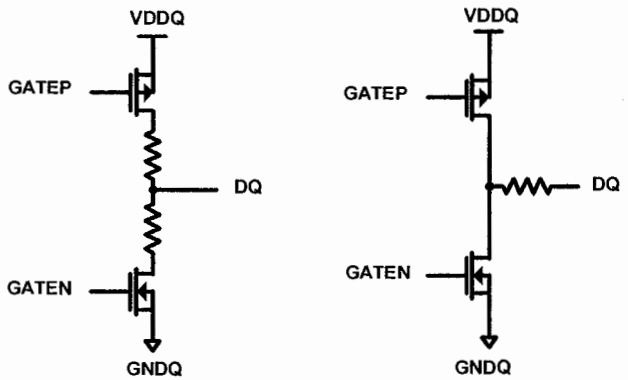


Fig. 7.23. OCD inverter stage with Linearization resistors

Petitioner Micron Ex-1021, 0189

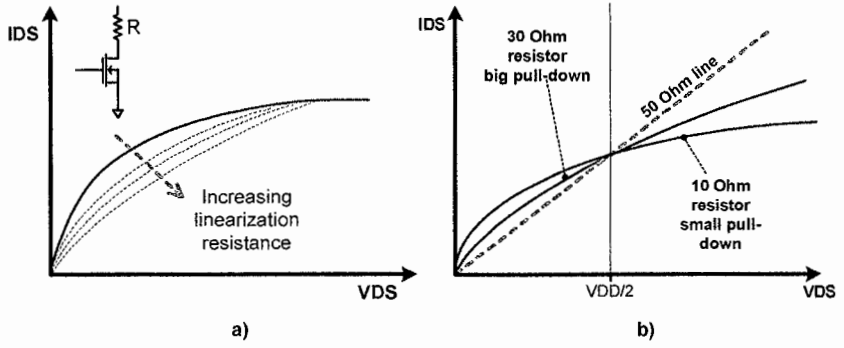


Fig. 7.24. Examples of choices for linearization resistor and driver dimensions

To design the optimal resistance value it is necessary to evaluate the MOS characteristics and find the proper trade-off amid die size and parasitic capacitance. In the end, the final output impedance is given partly by the MOS characteristic superimposed with the resistance characteristic.

It is important to highlight that the best result can be found only if we know the system in which the NAND memory will be plugged in and the best OCD configuration will be found after system simulations. Standardization efforts are in the direction to force the manufacturer to a specific solution in order to have a better interoperability between vendors. As an example, standard DDR2 protocol specifications definition approved by JEDEC defines with precision the characteristics of the output driver, including PVT variation [8].

In order to give a guiding rule to manufacturers of High Speed NAND to accommodate NAND process variations for min/max values, it is important to adopt a specification consensus to define the impedance value. ONFI consortium in combination with JEDEC aims at defining a standard for NAND DDR LVCMOS protocol.

Just for the sake of completeness, Fig. 7.25 shows an open drain configuration in which a big NMOS has a series resistor of the desired impedance (i.e. same value of the characteristic impedance of the line). With such a solution, the line is perfectly matched (without considering PVT variations) to the source at a price of termination scheme. Its advantage lies in the reduction of power drawn from the termination scheme because only the pull-down is used.

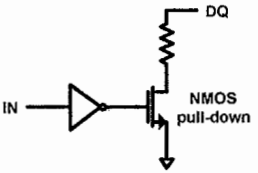


Fig. 7.25. Open drain OCD

Petitioner Micron Ex-1021, 0190

7.3.4 Slew rate control and bandwidth

In Sect. 7.2.3 it has been explained the importance of controlling the slew-rate of the output driver. Slew rate control is a key point in high speed OCD design as well. Drivers should be designed in order to avoid driving frequencies greater than the signalling rate. Simple and sophisticated methods can be used depending on the target signalling rate. In Fig. 7.26 three methods for controlling the slew rate are shown. The first introduces passive delays after the pre driver. This simply slows down the gates and, therefore, the slew rate output. Problems with this solution are related to process-voltage-temperature (PVT) variation, crowbar current and impedance mismatches. The second method is more sophisticated and uses a current control technique for the pre-driver stage. By means of such current controlled structure it is also possible to track PVT variations and operate proper corrections [19, 21]. Despite the method is going towards an ideal structure, it is worthless to adopt such a complex and area wasting solution if other design techniques are not considered or poorly implemented. The third method is much simpler and tricky. This method consists in adding feedback resistors to the power supply of the pre-driver, in order to reduce the local supply voltage when there is a current increase. This method is very simple and allows to achieve a very compact and simple structure. It also provides a sort of feedback versus the number of switching outputs: when a large number of outputs are switching, the output is slowed down helping its stabilization. Drawbacks are the pattern sensitivity and jitter increase.

When it is required to obtain a stable control of slew-rate and output impedance, a time-split method is widely used. The basic principle is to split output pull-up and pull-down devices into branches and activate them serially with proper sequential delay. This time-distributed driver can be implemented in a simple analog form suitable for relative low operating frequency or digital form [5, 13, 20, 27]. Figure 7.27 shows a basic implementation of the analog form where the pull-up/down branches are driven by a resistive line which contributes to define the *RC* delay element for each branch. Each branch can be "weighted" to obtain the best slew rate conditions.

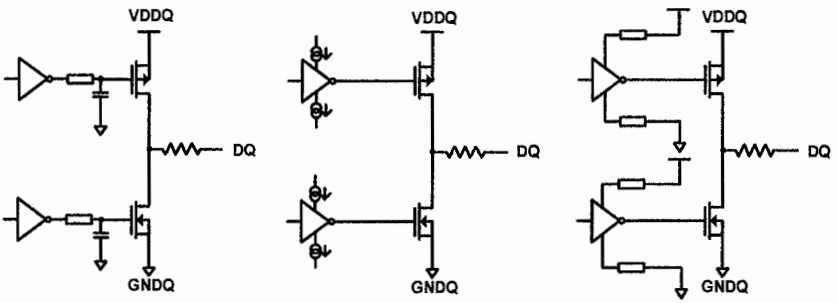


Fig. 7.26. Slew rate control by means of (a) gate slope (b) gate slope controlled (c) feedback on power supply

Petitioner Micron Ex-1021, 0191

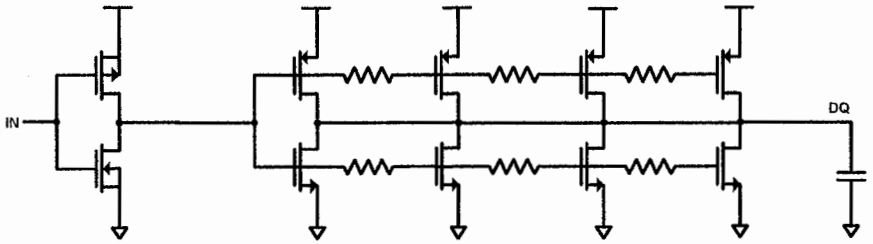


Fig. 7.27. Slew rate control by output driver time-distributed activation

The digital form of the time-distributed slew rate control is well suited for high frequency operations. Figure 7.28 shows an implementation example where pull-up and pull-down are split into four branches. Each branch is driven by a separate control signal. Each control signal is skewed by a proper delay interval generated by simple CMOS buffers properly sized. The gate of each branch is driven with a sharp edge and each branch is activated in sequence by delay intervals.

In Fig. 7.29 it is shown the sequence of activation/deactivation of the pull up/pull down branches. At Time T = 0 all the pull down are activated and pull-ups are deactivated; after delay time t_d, one pull-down branch is deactivated and the first pull up branch is switched on. At time $2 \cdot t_d$ a second pull-down branch is deactivated and a pull-up branch is activated and so on. The result is a sort of staircase whose slope depends on the t_d time delay.

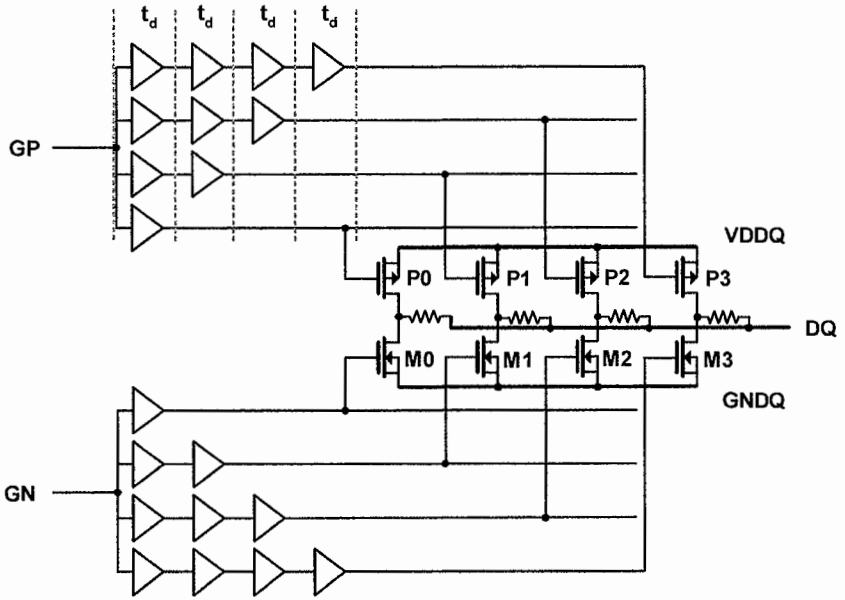


Fig. 7.28. Slew rate control by mean of digital time-domain split

Petitioner Micron Ex-1021, 0192

It is important to notice that a crowbar current is present during the whole transition but this is traded-off in change of the important fact that, during each time frame, it is always the same total number of branches to be active, thus resulting in constant output impedance during the switching.

The shown method is very effective and represents a relatively simple option for high speed operation of LVCMOS buffers well beyond 400 MHz with a good PVT and impedance matching control.

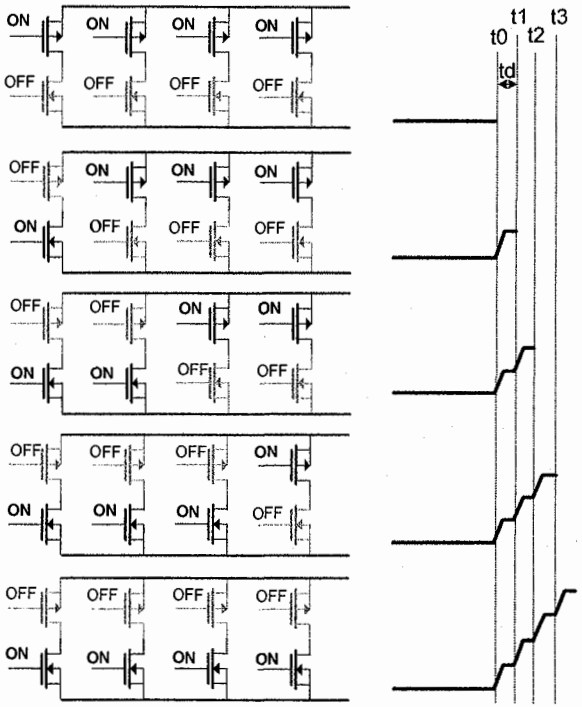


Fig. 7.29. time domain split time diagram

7.3.5 Voltage domain change: level shifting

Interface voltage usually differs from the power supply of the core of the memory. For example, the memory could internally operate at 1.5 V by means of a DC-DC down-converter, whereas the data interface needs a 3 or 1.8 V driving. Voltage domain change occurs also when the memory has different power pins for core supply voltage and I/Os. This situation allows the use of independent supply generators to separate the noise coming from data bus and from the core region. In a simpler system designs it is still possible to connect the pins to the same supply on the PCB. The OCD structure implements the level shifting function which consists in shifting the levels of the digital signals from the core voltage GND/VDD to the

Petitioner Micron Ex-1021, 0193

interface voltage GNDQ/VDDQ. Figure 7.30 shows a modified structure where NMOS transistors M5 and M6 are added in order to speed up the transition of nodes from low to high.

The level shifting circuit or, more generally, the point where the data change voltage domain, is critical in jitter generation. The two domains provide two different references for the signal detection; therefore, any disturbs on the power supply lines lead to the introduction of additional distortion.

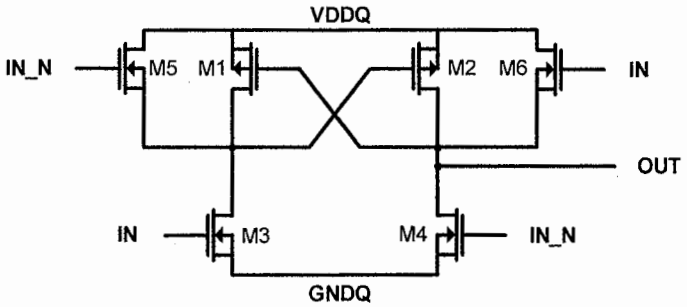


Fig. 7.30. Level shifter modified

7.3.6 Jitter sources and duty cycle distortion

off-chip driver complexity implies that data is travelling along many gates before reaching the output stage. This is due also to the fact that final driver is a big stage that needs proper buffering in order to be driven correctly and in the appropriate time delay. In addition, some logic levels are needed to implement logic functions such as test functions added in the path. Last, but not the least, a level shifter or voltage domain change is usually present in the data path. The result is a long chain of inversions the data has to travel. The design of the chain of inversions is fundamental in the control of duty cycle distortion. Duty cycle distortion occurs when:

- Positive and negative slopes are different.
- Number of inversion is odd.
- Ground or power shifts.

To reduce the jitter in a chain of inverters it is necessary to keep the same slope in the chain i.e. using the same ratio between the driver strength and the load instead of trying to minimize the inverter chain.

Another source of jitter is hidden in level shifters and voltage domain change. Level shifter presented in Fig. 7.30 introduces asymmetric positive/negative slopes detected by a receiver gate with different time delay. Voltage domain change affects jitter due to SSN that shifts the detection threshold as shown in simplified diagram of Fig. 7.31.

Petitioner Micron Ex-1021, 0194

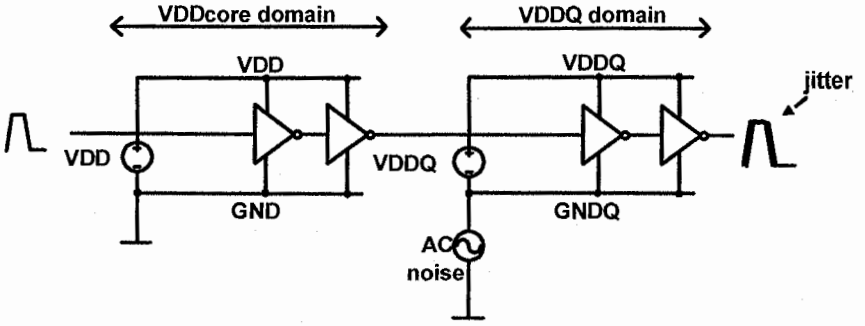


Fig. 7.31. Voltage domain change as a source of jitter

Output noise causes VDDQ and GNDQ bounces and, in turn, GNDQ to be different from GND. The result is a change of the trigger threshold in the receiving inverter, where the supply changes domain. This particular situation might happen at a particular operating frequency, when the delay from domain change to output equals the bit time. The data bit which is undergoing a transition at the output causes overshoot on the power supply. This overshoot is usually present both on VDDQ and GNDQ; therefore, both local ground and VDD bounces shift the characteristics of the inverters. At the voltage domain frontier the bit is driven by a different reference and is detected by an inverter subjected to the noise bounce, leading to a time jitter. Proper care should be taken in designing the power domain change.

7.3.7 ESD

Electrostatic Discharge (ESD) is a natural phenomenon: charged entities discharge current pulses to the IC pins [5, 12]. High speed I/O designers have to confront themselves with ESD design topics since their circuits are connected to the external world and to the ESD protection network. Usually, ESD design should start when defining the I/O floorplan of the chip. Proper definition of ESD constraints and paths at an early stage results in the optimization of I/O capacitance and size. In particular, output capacitance is a key topic for high-speed design and it is the main reason why a basic ESD overview is included in this chapter.

Three main discharge types can be modelled and are relevant to IC design: Human Body Model, Machine Model, and Charged Device Model whose characteristics are summarized in Table 7.5.

Table 7.5. Characteristics of ESD discharge types

	Ipeak (A)	Rise time (ns)	Bandwidth (MHz)
HBM	1.33	10–30	2.1
MM	3.7–7	15–30	12
CDM	10	1	1,100

Petitioner Micron Ex-1021, 0195

A proper understanding and proper definition of the ESD requirements of a product is fundamental to avoid over-engineering and obtain better performances.

ESD involves all types of pins: supplies, inputs, outputs or bidirectionals. Some pins can have special functions i.e. they can be tolerant to 12 V for some testing or manufacturing purposes or open drain configuration. We are now considering the protection of inputs and I/O pins but it is important to remark that discharge can be applied between two or more different pins whatever type they are.

Whenever an ESD discharge is applied to a couple of pins, the discharge path develops through the weakest point in order to find the less resistive path. This weaker point is generally entering a breakdown that could be destructive. For example, a parasitic bipolar can turn-on, offering low impedance path to the discharge. If we are able to identify the weak point (sometimes it is not so easy because it can be due to a little separation between biased straps in the layout) and fix it, the discharge develops through the next weak point in our layout. By iterations we reach a point where the desired discharge path is strong enough to avoid burning out. In this perspective, the best way to address ESD problems is to adopt a favourite ESD path approach. Within this approach, all possible pin combinations that can be discharged are considered by building a table with all the pins. Then, all the paths are examined and optimized to offer the preferred path to the discharge. Another advantage is that it is not necessary to exaggerate with distances and guard rings in our layout, avoiding a waste of space.

In Fig. 7.32 the main discharge devices and paths in a memory are shown. First of all, there are some important devices paths that eventually collect all types of discharges.

In the next subsections the input and I/O pads ESD concerns will be treated keeping in mind that key point in I/O speed design is capacitance and ESD protection always plays a role in capacitance contribution.

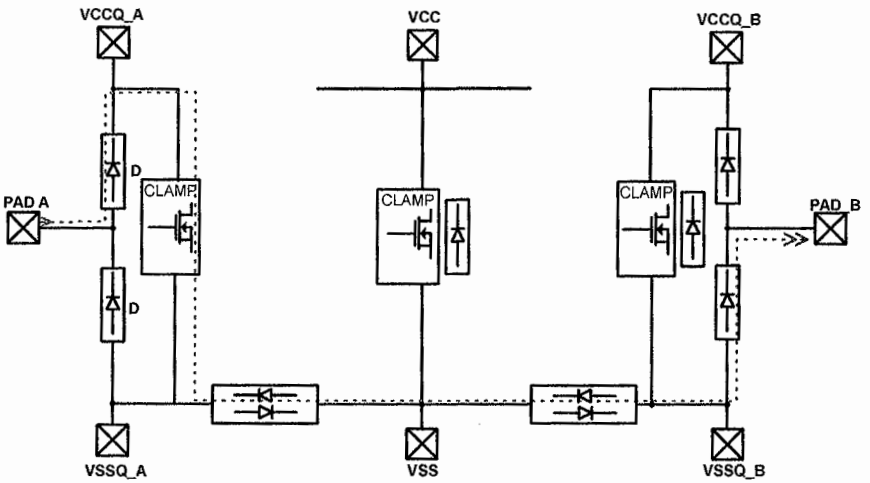


Fig. 7.32. Simplified ESD network

Petitioner Micron Ex-1021, 0196

Input and output ESD protections

Input pins are generally protected by ESD diodes as shown in Fig. 7.33a. A discharge applied to the input pins must be routed toward either VCCQ or VSSQ through diodes. A VCC-VSS clamp will eventually close the discharge loop. As we mentioned, if the protection is slow to activate, then other weak paths could turn on. To avoid that, it is necessary that the clamp is well connected to the input supply with a low ohmic path.

A secondary protection is generally present by means of a resistor, generally placed towards the gates of the input pins to offer a *RC* delay to the ESD signal, plus another diode. CDM discharge types have a current slope in the range of some nanoseconds therefore R could be in the range of kiloOhms.

Diodes, resistor and routing contribute to the total input capacitance.

Output stage is normally an inverting stage with a NMOS and a PMOS connected to the pad as shown in Fig. 7.33b. NMOS is generally the weakest point because of snapback phenomena and must be controlled. If technology (implant scheme) is not able to guarantee a safe snapback operation in the output transistor, this must be controlled by means of serial ballast resistors, typically in N-Well. Normally, in high-speed design, resistors are introduced for linearity reasons; therefore, the ballast resistors are not wanted. If the resistor has to act as a ballast resistor, then width of the resistor must be chosen large enough to avoid saturation of the resistor, and the capacitance of the structure can increase correspondently [5].

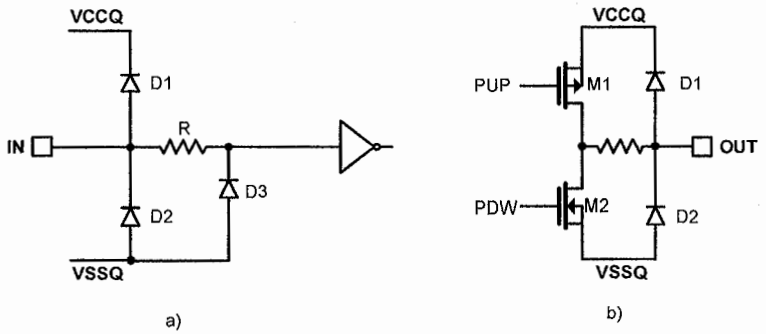


Fig. 7.33. (a) Input protection, (b) output protection

7.3.8 Layout

The structure in Fig. 7.34 shows a special example of a NMOS driver in non-salicided technology. The protection diodes are sliced and placed between fingers of the NMOS. The drain junction acts also as the protection diode anode. The contact to gate distance on the drain side is shown: contacts are placed at a non minimum distance so that $n+$ diffusion helps protect the transistor from snap-back.

Petitioner Micron Ex-1021, 0197

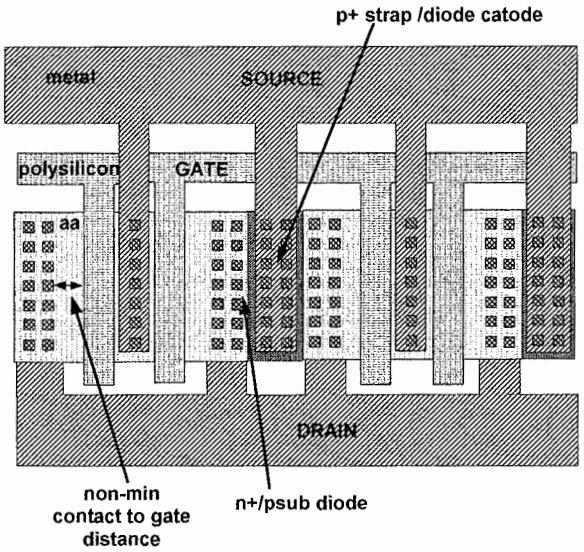


Fig. 7.34. NMOS pull-down layout example

The contact–gate distance is critical in defining the output capacitance, since the parasitic drain diode is proportional to the distance and it is an important point for output capacitance improvement.

In high speed drivers the diode is often separate from the MOS transistor and placed where more convenient.

The disposition of OCD components in the layout (called floorplan of the OCD) is very important to achieve optimal performance. As usual the OCD requirements are not the only important characteristics of the chip and some trade-offs must be done when defining OCD floorplan. Usually the OCD floorplan starts by considering the total available space for pads and ESD protections structures and the proper position of decoupling capacitors. As a matter of fact, the X and Y dimensions of the chip are determined by the core of the device, its array, and all the pads in the worst case must fit in one side of the device. In addition, the ESD structures (clamps) must be put in a proper disposition in order to obtain efficient ESD protection.

In Fig. 7.35 an ideal OCD floorplan is shown. The output PMOS and NMOS structures, together with linearization resistors are placed at both sides of the DQ Pad. With this position, the MOS transistors are easily (and low ohmic) connected to the pad and to the power supply rails which are routed horizontally above and below the PAD stripe. The pre-driver structures are placed aside the transistors. ESD primary diodes are placed on top and bottom of the pad in an advantageous position for connecting them to the power supply rails and to the DQ pad. The floorplan shown here allows to optimize the metal routing needed at the DQ node in order to obtain the smaller possible capacitance. Moreover, on top the logic circuits of the OCD are designed, including the serializer circuit and level shifting

Petitioner Micron Ex-1021, 0198

circuits. Input logic could be positioned wherever is more convenient, also inside the chip.

The proposed scheme shows a VDD/GND pad pair on the left and right of the DQ pad, thus providing short forward and return paths to the signal, independent from the paths of the other DQ pads.

The real floorplan will be the result of trade-off choices between size and performances.

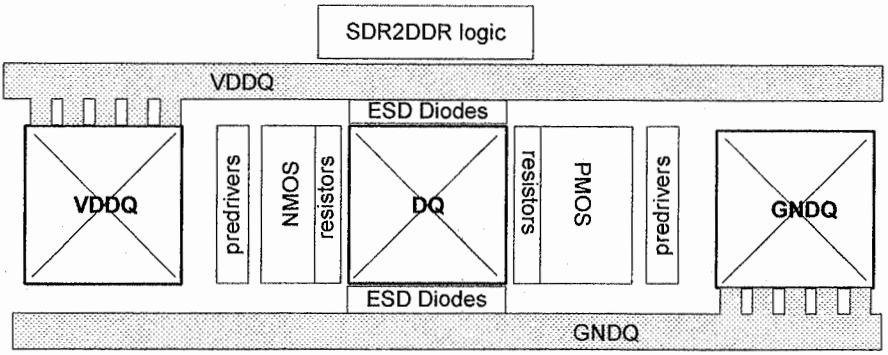


Fig. 7.35. OCD layout ideal floorplan

7.3.9 I/O capacitance problem

The capacitance of the Off-Chip Driver plays a fundamental role in high speed design for many different reasons and must be minimized. At system level, when more dies are attached on the same channel (a typical situation in many application and especially in high-speed demanding applications like SSD), the I/O capacitance sums up to very high values, compromising signal integrity and posing a limitation in dies stacking on a single channel. This reflects on the decision to have systems with multiple channels (impacting the cost of the system itself) or to adopt a termination scheme. We have seen that achieving a low capacitance at the DQ node is a difficult task which requires the best trade-off choices among design/layout, technology options, and package.

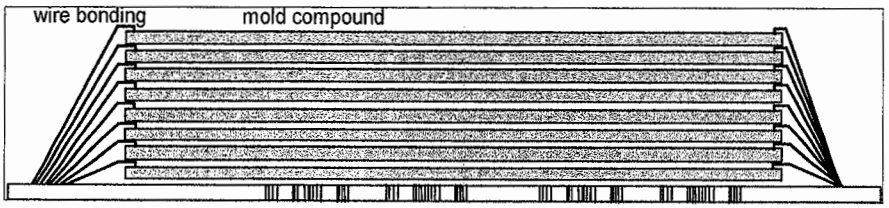


Fig. 7.36. Package with eight dies stacking on a Ball Grid Array substrate

Petitioner Micron Ex-1021, 0199

I/O input capacitance also plays a role on signal integrity of very high frequency terminated systems because the receiver capacitance acts as a shunt for the signal, creating disturbance in the signal quality.

Figure 7.36 shows an example of a chip with eight dies stacked on a BGA substrate.

In the following Table 7.6 the contributors to the I/O capacitance are summarized together with some numeric examples.

Table 7.6. Contributors in building the OCD output capacitance

	Metal routing	ESD diodes	PAD	Transistors and resistors	Total
C	1 pF	1 pF	0.2 pF	1.5 pF	4.0 pF

The first contribution is given by the transistor diffusion capacitance whose value is technology dependent and can be very high. ESD rules imposes contact to gate distances with relaxed numbers which, in turn, increase the diffusion area responsible for drain capacitance.

Moreover, if smaller output impedance is wanted in order to drive difficult overloaded bus situation, it is necessary to increase the pull-up pull-down MOS finger size, thus increasing also the capacitance contribution. For example, if the transistor is doubled in size, the associated parasitic capacitance is doubled and, in turn, it gets an increase of about 30% in the total capacitance.

Another important contribution is given by the linearization resistors. Here we recall that the choice of transistor size and linearization resistors can play a key role in the signal integrity due to the capacitance introduced. A careful trade-off must be done based on system simulations. ESD structures also have a technology-dependent capacitance.

Last but not the least, metal routing used to connect the off-chip driver components can result in a very high capacitance. This contribution is very dependent on the quality of the layout choices (Sect 7.3.8).

Finally, Table 7.7 shows that a typical system with eight memory dies on the same channel can easily reach values up to 50 pF.

Table 7.7. Total system data line capacitance in an eight die stack

	CI/O	WB	PCB	Controller	Total N = 8
C TOT	N*4.0	N*0.5	1.35 pF	4.5 pF	42 pF

Summarizing, I/O capacitance optimization is achieved by a proper analysis of the following design and technology points:

- Proper choice of transistors and linearization resistors size and arrangement
- Reduction of Drain diffusion capacitance by specific technology modification for the I/O transistors

Petitioner Micron Ex-1021, 0200

- Layout optimization
- ESD gate–contact distance reduction and ESD diodes size reduction
- Package choice

Standardization requires the NAND products from different vendors to have the same or similar loads. For example, let's consider to build a 32 GB system using sixteen 16 Gb NAND. The requirements of performances drive towards the choice of a dual channel; therefore, we have eight dies on the same channel. Today, the most cost effective solution is a TSOP package with four stacked dies. As a result, we have two packages tied to the same channel. Figure 7.37 shows the above mentioned system. Manufacturing implies a variance of production parameters and, as a consequence, the capacitance of an I/O pin, composed of several terms, can spread from a typical value in positive and negative direction. If we consider the worst case where all the dies in TSOP1 have minimum capacitance and all the dies in TSOP2 have maximum capacitance, the capacitance spread between the two channels is equal to:

$$C_{spread} = 4 \cdot (C + \Delta) - 4 \cdot (C - \Delta) \tag{7.4}$$

If the absolute variation is $\Delta = 1$ pF this leads to an imbalance of 8 pF. The smaller the imbalance is, the smaller the signal integrity issues are: these last limit system performances. Control and standardization of the NAND products will be a key point for enabling new applications based on solid state storage.

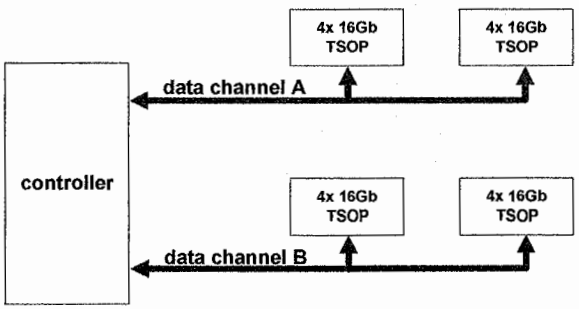


Fig. 7.37 32 GB system, dual channel using four dies TSOP

References

1. www.onfi.org
2. http://www.intel.com/standards/nvmhci/index.htm
3. https://www.denali.com/en/events/webcasts/2008/togglenand/
4. G. Campardo, R. Micheloni, D. Novosel, VLSI-Design of Non-Volatile Memories, Springer-verlag, Heidelberg, 2005.
5. S. Dabral T., Maloney Basic ESD and I/O Design, Wiley-Interscience, New York, 1998.
6. B. Jacob, Spencer W. Ng, David T. Wang, Memory Systems: Cache, DRAM, Disk, Morgan Kaufman, CA, 2007.

Petitioner Micron Ex-1021, 0201

7. S.H. HallGarrett, W. HallJames, A. McCall, High-Speed Digital System Design-A Handbook of Interconnect Theory and Design Practices, Wiley-Interscience, New York, 2000.
8. JEDEC JESD79-2C
9. T. Wada, M.E. Kenji Mami, Simple Noise Model and Low-Noise Data-Output Buffer for Ultrahigh-speed Memories, IEEE Journal of Solid-State Circuits, Vol. 25, No. 6, Dec. 1990.
10. H.B. Bakoglu, Circuits, Interconnections, and Packaging for VLSI, Addison-Wesley, Boston, MA, 1990.
11. M. Annaratone, Digital CMOS Crcuit Design, Kluwer, Boston, MA, 1986.
12. www.esda.org
13. Shyh-Jye Jou et al, Low Switching Noise and Load-Adaptive Output Buffer Design Techniques, IEEE Journal of Solid-State Circuits, Vol. 36, No. 8, pp. 1239–1249, 2001.
14. P. Heydari, M. Pedram, Ground Bounce in digital VLS circuits, IEEE Transactions on VLSI Systems, Vol. 11, Issue 2, pp. 180–193, April 2003.
15. R. Senthinathan, J.L. Prince, "Simultaneous Switching Ground Noise Calculation for Packaged CMOS Devices," IEEE Journal of Solid-State Circuits, Vol. 26, pp. 1724–1728, Nov. 1991.
16. R. Senthinathan, J.L. Prince, Simultaneous Switching Noise of CMOS Devices and Systems, Kluwer, Boston, MA, 1994.
17. E. Chioffi, F. Maloberti, and others, "High-Speed, Low-Switching Noise CMOS Memory Data Output Buffer," IEEE Journal of Solid-State Circuits, Vol. 29, No. 11, pp. 1359–1365, November 1994.
18. T. Gabara, W. Fischer, and others, "Forming Damped LRC Parasitic Circuits in Simultaneously Switched CMOS Output Buffers," IEEE Journal of Solid-State Circuits, Vol. 32, p. 407, 1997.
19. K. Asahina, S. Kato, and S. Kayano, "Output Buffer with On-Chip Compensation Circuit," Proceedings of 1993 Custom Integrated Circuit Conference, pp. 29.1.1–29.1.4, May 1993.
20. T. Sekiguchi, M. Horiguchi et al, "Low-Noise, High-Speed Data Transmission Using a Ringing-Canceling Output Buffer," IEEE Journal of Solid-State Circuits, Vol. 30, No. 12, pp. 1569–1574, Dec. 1995.
21. C.H. Lim, W.R. Daasch, "Output Buffer with Self-Adjusting Slew Rate and On-Chip Compensation IC/Package Design Integration," Proceedings of 1998 IEEE Symposium on IC/Package Design Integration, Issue 2–3, Page(s): 51–55, Feb. 1998.
22. M.-J. Edward Lee, William J. Dally, Ramin Farjad-Rad, Hiok-Tiaq Ng, Ramesh Senthinathan, John Edmondson, and John Poulton, CMOS High-Speed I/Os – Present and Future, 2003 IEEE International Conference on Computer Design (ICCD'03), Oct. 2003
23. H. Hikeda, A 3D Packaging with 4Gb Chip-Stacked DRAM and 3Gbps High-Speed Logic 3D-SIC, 2007.
24. J. Poulton, Signaling in High-Performance Memory Systems 1999 ISSCC.
25. T.J. Gabara, S.C. Knauer, Digitally adjustable resistors in CMOS for High-performance applications, IEEE JSSC, 27, p 1176, 1992.
26. A. Chandrakasan and R. Brodersen, Eds., Low Power CMOS Design, Kluwer, Boston, MA, 1995.
27. B. Deutschmann, T. Ostermann, CMOS Output Driver With Reduced Ground Bounce and Electromagnetic emission, Solid-State Circuits Conference, 2003. ESSCIRC '03.

Petitioner Micron Ex-1021, 0202

8 Sensing circuits

L. Crippa[1] *and R. Micheloni*[2]

8.1 Introduction

The reading operation is designed to address a memory cell and extrapolate the information stored therein. In the case of a NAND-type Flash memory, the memory cells are connected in series, in groups (strings) of 2^k cells, up to 64.

In Fig. 8.1 the string of a NAND Flash memory is illustrated: M_{BLS} and M_{SLS} NMOS selector transistors connect the string to bitline BL and to source line SL respectively.

As in other types of Flash memories, the stored information is associated with the cell's threshold voltage V_{TH}: in Fig. 8.2 the threshold voltage distributions of cells containing one logic bit are shown.

If the cell has a V_{TH} belonging to the erased distribution, it contains a logic "1", otherwise, if it belongs to the written distribution, it contains a logic "0". Cells containing n bit of information have 2^n different levels of V_{TH}.

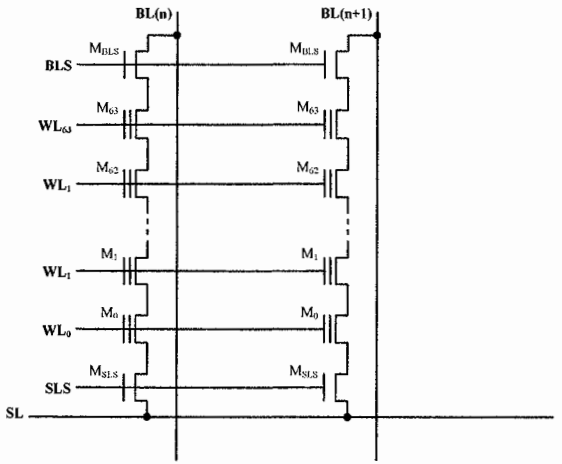


Fig. 8.1. Two strings in a NAND architecture

[1] Forward Insights, luca.crippa@ieee.org
[2] Integrated Device Technology, rino.micheloni@ieee.org

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_8,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0203

Flash cells act like usual MOS transistors. Given a fixed gate voltage, the cell current is a function of its threshold voltage. Therefore, through a current measure, it is possible to understand to which V_{TH} distribution the memory cell belongs.

The fact that a memory cell belongs to a string made up by other cells has some issues. First of all, the unselected memory cells must be biased in a way that their threshold voltages do not affect the current of the addressed cell. In other words, the unselected cells must behave as pass-transistors. As a result, their gate must be driven to a voltage (commonly known as V_{PASS}) higher than the maximum possible V_{TH}. In Fig. 8.2 V_{PASS} has to be higher than V_{THMAX}.

However, the presence of $2^n - 1$ transistor in series has a limiting effect (saturation) on the current's maximum value; this maximum current is, therefore, much lower than the one available in NOR-type Flash memories.

Figure 8.3 shows the I–V (current–voltage) characteristic of a NAND cell (string): V_{READ} is applied to the selected gate while V_{PASS} bias the unselected gates. V_{PASS} is a fixed voltage. Three main string working-regions can be highlighted.

1. Region A: the addressed cell is not in a conductive state.
2. Region B: V_{READ} makes the addressed cell more and more conductive.
3. Region C: the cell is completely on, but the series resistance of the pass transistors (unselected cells) limits the current to I_{SSAT}.

The string current in region C can be estimated as:

$$I_{SSAT} = \frac{V_{BL}}{(n-1)R_{ON}} \tag{8.1}$$

where R_{ON} is the series resistance of a single memory cell, V_{BL} is the voltage applied to the bitline and n is the number of the cells in the string. R_{ON}, at a first approximation, is the resistance of a transistor working in the ohmic region.

For a MOS transistor in ohmic region the following equation holds true:

$$I_D = k \cdot \left[(V_{GS} - V_{TH}) \cdot V_{DS} - \frac{V_{DS}^2}{2} \right] \tag{8.2}$$

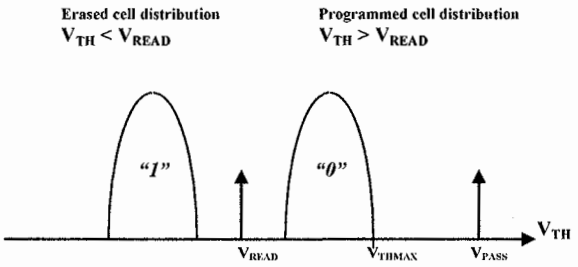


Fig. 8.2. Threshold voltage distributions for erased and programmed cells

Petitioner Micron Ex-1021, 0204

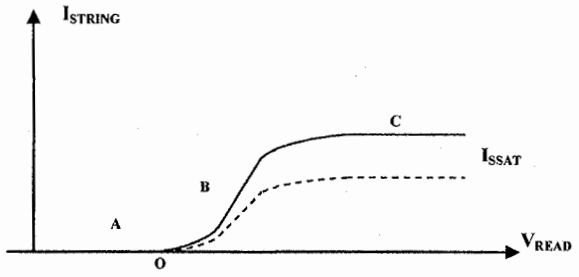


Fig. 8.3. Cell current characteristics versus gate voltage

For small V_{DS} values, as in our case, Eq. (8.2) may be simplified as:

$$I_D = k\left[\left(V_{GS} - V_{TH}\right) \cdot V_{DS}\right] \tag{8.3}$$

Therefore, R_{ON} is equivalent to

$$R_{ON} = \frac{V_{DS}}{I_D} = \frac{1}{k\left(V_{GS} - V_{TH}\right)} \tag{8.4}$$

Equation (8.4) shows that R_{ON} is a function of V_{TH}. In other words, I_{SSAT} depends on the V_{TH} values of the n cells in series. When all the cells are programmed to V_{THMAX}, R_{ON} takes its maximum value (dashed line in Fig. 8.3). R_{ON} influences the I–V characteristic also in region B but in a more negligible way.

In order to reduce the dependency from R_{ON}, the cell has to be read in region B as near as possible to point O.

The order of magnitude of the saturation current, in the state-of-the-art NAND technologies, is a few hundreds of nanoAmpere, that means a reading current of some tens of nA. It is very hard to sense such small currents with the standard techniques used in NOR-type Flash memories, where the reading current is, at least, in the order of some microAmpere. Moreover, in NAND devices, tens of thousands of strings are read in parallel. Therefore, tens of thousands of reading circuits are needed. Due to the multiplicity, a single reading circuit has to guarantee a full functionality with a very low area impact. As a matter of fact, the first memory NAND prototypes used traditional sensing methods, since the said currents were in the order of tens of microAmpere [4].

The reading method of the Flash NAND memories consists in integrating the cell current on a capacitor in a fixed time (Fig. 8.4). The voltage ΔV_C across a capacitor C, charged by a constant current I for a time period ΔT, is described by the following equation:

Petitioner Micron Ex-1021, 0205

$$\Delta V_C = \frac{I}{C}\Delta T \tag{8.5}$$

Since the cell current is related to its V_{TH}, the final voltage on the capacitor (ΔV) is a function of V_{TH} too.

This chapter deals with different reading techniques, starting from the one using the bitline parasitic capacitor, ending with the most recent sensing technique which integrates the current on a little dedicated capacitor. The above mentioned techniques can be used both in SLC and MLC NAND memories. In the MLC case, multiple basic reading operations are performed at different gate voltages (Chap. 10).

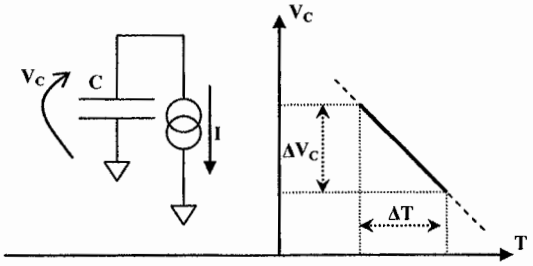


Fig. 8.4. Capacitor discharge through a constant current source

8.2 Reading techniques using the bitline capacitor

Historically, the first reading technique used the parasitic capacitor of the bitline as the element of the cell current integration [1–3].

In Fig. 8.5 the basic scheme is shown. V_{PRE} is a constant voltage. At the beginning, C_{BL} is charged up to V_{PRE} and then it is left floating (T_0). At T_1 the string is enabled to sink current (I_{CELL}) from the bitline capacitor. The cell gate is biased at V_{READ}. As shown in Fig. 8.6, if the cell is erased, the sunk current is higher than (or equal to) I_{ERAMIN}. A programmed cell sinks a current lower than I_{ERAMIN} (it can also be equal to zero). C_{BL} is connected to a sensing element (comparator) with a trigger voltage V_{THC} equal to V_{SEN}. Since I_{ERAMIN}, C_{BL}, V_{PRE} and V_{SEN} are known, it follows that the shortest time (T_{EVAL}) to discharge the bitline capacitor is equal to:

$$T_{EVAL} = C_{BL}\frac{V_{PRE} - V_{SEN}}{I_{ERAMIN}} \tag{8.6}$$

If the cell belongs to the written distribution, the bitline capacitor will not discharge below V_{SEN} during T_{EVAL}. As a result, the output node (OUT) of the voltage comparator remains at 0. Otherwise, if the cell is erased, V_{BL} drops below V_{SEN} and the OUT signal is set to 1.

Petitioner Micron Ex-1021, 0206

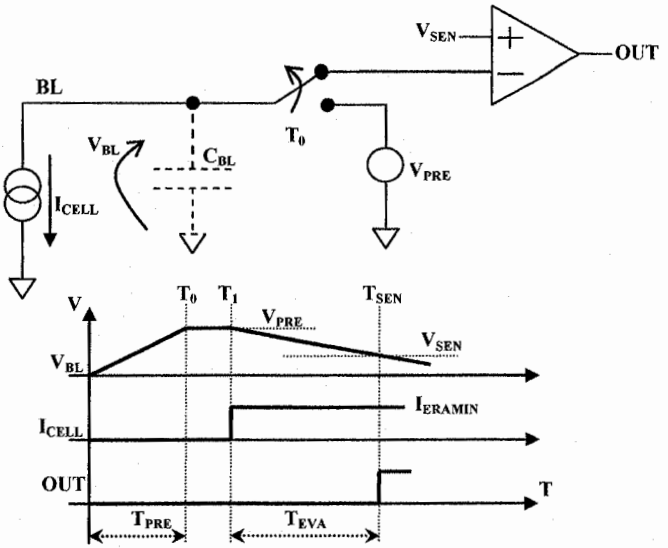


Fig. 8.5. Basic sensing scheme exploiting bitline capacitor and the related timing diagram

Definition 8.1 A *Single Read Operation (SRO)* is what has been described so far. Fixing V_{PASS}, V_{PRE}, V_{SEN} and T_{EVAL}, the single read operation establishes if the cell V_{TH} is higher or lower than the read voltage V_{READ}.

Definition 8.2 V'_{READ} is the voltage to be applied on the cell gate in order to have $V_{BL} = V_{SEN}$. In other words, the output of the comparator resulting from a SRO at $(V'_{READ} + \varepsilon)$ is inverted compared to the one resulting from a SRO at $(V'_{READ} - \varepsilon)$. V'_{READ} is defined as the *Read Cell Threshold Voltage* V_{THR}.

Figure 8.7 shows the matrix and reading circuits organization in the former NAND devices. In that structure, the cells belonging to the same wordline (WL) are read in parallel during a single read operation. Therefore, each bitline has its sensing circuit, traditionally called "Page Buffer" (PB) [1].

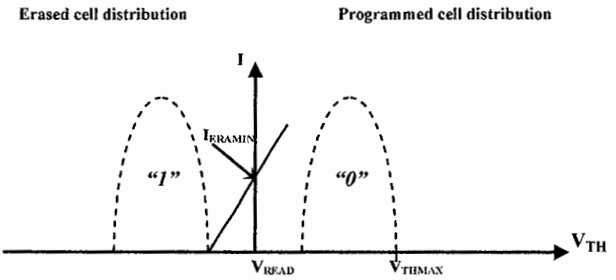


Fig. 8.6. Minimum erase current versus V_{READ}

Petitioner Micron Ex-1021, 0207

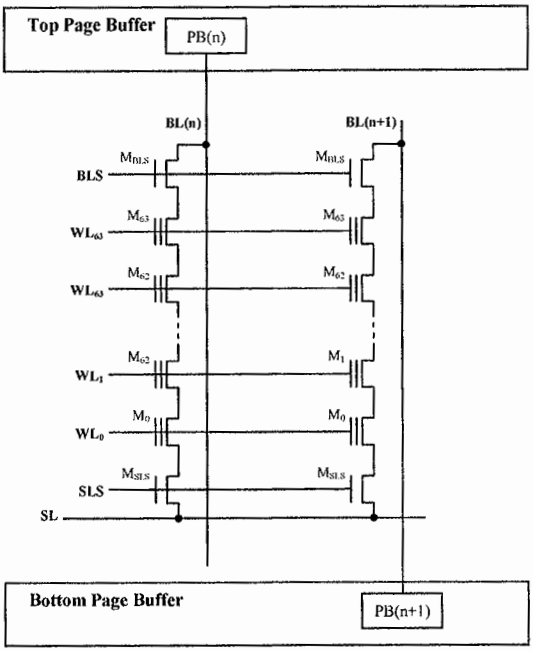


Fig. 8.7. Page buffer organization

The basic page buffer structure is sketched in Fig. 8.8. Figure 8.9 shows a timing diagram of a SRO. During the precharge phase T_{PRE}, M_{SEL} and M_{PCH} are biased to V_{PRE} and $V_{DD} + V_{THN}$ respectively. V_{THN} is the threshold voltage of a NMOS transistor and V_{DD} is the device's power supply voltage.

As a consequence, C_{BL} is charged to the following value:

$$V_{BL} = V_{PRE} - V_{THN} \tag{8.7}$$

During this phase, the SO node charges up to V_{DD}. Since V_{GS} and V_{DS} can be higher than 20–22 V, M_{HV} has to be an high voltage (HV) transistor. In fact, during the erase phase, the bitlines are at about 20 V and M_{HV} acts as a protection element for the page buffer's low voltage components. Instead, during the reading phase, M_{HV} is biased at a voltage that makes it behave as pass-transistor. Moreover, during the precharge phase, the appropriate V_{READ} and V_{PASS} are applied to the string. M_{BLS} is biased to a voltage (generally V_{DD}) that makes it work as pass transistor. Instead, M_{SLS} is turned off in order to avoid cross-current consumption through the string.

Typically, V_{BL} is around 1 V. From Eq. (8.7), V_{PRE} values approximately 1.4–1.9 V, depending on the V_{THN} (NMOS threshold voltage). The bitline precharge phase usually lasts 5–10 μs, and depends on many factors, above all the value of the distributed bitline parasitic *RC*.

Petitioner Micron Ex-1021, 0208

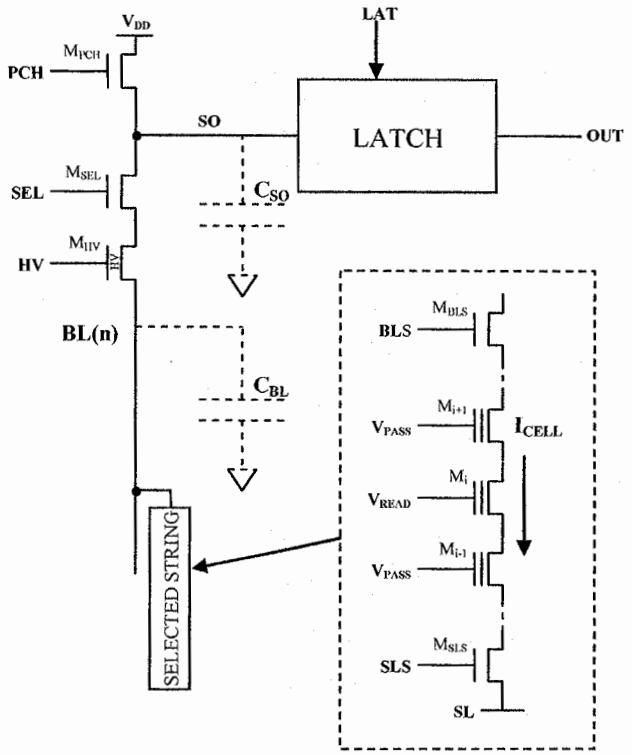


Fig. 8.8. Basic elements of the page buffer

Sometimes this precharge phase is intentionally slowed down to avoid high current peaks from V_{DD}. In order to achieve this, the M_{PCH} gate could be biased with a voltage ramp from GND to $V_{DD} + V_{THN}$.

At the end of the precharge phase, PCH and SEL are switched to 0. As a consequence, the bitline and the SO node parasitic capacitor are left floating at a voltage of $V_{PRE} - V_{THN}$ and V_{DD}, respectively. M_{SLS} is then biased in order to behave as pass transistor. In this way the string is enabled to sink (or not) current from the bitline capacitor.

At this point, the evaluation phase, indicated with T_{EVAL} in Fig. 8.9, starts. If the cell has a V_{TH} higher than V_{READ}, no current flows and the bitline capacitor maintains its precharged value. Otherwise, if the cell has a V_{TH} lower than V_{READ}, the current flows and the bitline discharges (dotted line in Fig. 8.9).

After T_{EVAL}, M_{SEL} is biased to a voltage value $V_{SEN} < V_{PRE}$. If, during T_{EVAL}, I_{CELL} discharged the bitline under the value:

$$V_{BL} < V_{SEN} - V_{THN} \tag{8.8}$$

Petitioner Micron Ex-1021, 0209

then M_{SEL} turns on, equalizing the voltage level of the SO node (V_{SO}) to V_{BL}. This is true because $C_{BL}>>C_{SO}$; in this way, the charge sharing can be neglected. In Fig. 8.9 the case that satisfies Eq. (8.8) is shown with dotted lines.

Definition 8.3 The *Average Threshold Current* I_{READTH} is defined as:

$$I_{READTH} = \frac{\Delta V \cdot C_{BL}}{T_{EVAL}} \tag{8.9}$$

where:

$$\Delta V = (V_{PRE} - V_{THN}) - (V_{SEN} - V_{THN}) = V_{PRE} - V_{SEN} \tag{8.10}$$

If, during the evaluation phase, the average cell current is higher than Eq. (8.9), the Eq. (8.8) holds true. Besides, V_{SO} is equalized to V_{BL} and, according to Definition 8.2, V_{THR} is lower than V_{READ}. On the contrary, if the average cell current is lower than Eq. (8.9), the Eq. (8.8) is false, V_{SO} remains at V_{DD} and V_{THR} is higher than V_{READ}.

Therefore, T_{EVAL} is an important parameter for both the overall time of the single reading operation and the overall time of the program operation (during the verify phase – Chap. 10).

T_{EVAL} can be easily computed from Eq. (8.9):

$$T_{EVAL} = \frac{\Delta V \cdot C_{BL}}{I_{READTH}} \tag{8.11}$$

In order to reduce as much as possible the evaluation time, we must reduce ΔV and C_{BL} and increase the average threshold current. C_{BL} depends on the technological parameters and is linked to the length and geometrical dimensions of the bitline: in the best cases it is around 2 pF. It is very difficult to have ΔV smaller than 100 mV for design reasons; this value is sized in order to hide the collateral effects during the evaluation phase, e.g. the disturbance and the noise injection on the bitline voltage. I_{READTH} depends on technological parameters and biasing conditions (pass voltages). With state-of-the-art technologies (sub-30 nm), a value greater than 50 nA is not available. From Eq. (8.11) it follows:

$$T_{EVAL} = \frac{100mV \cdot 2pF}{50nA} = 4\mu s \tag{8.12}$$

4 μs is a best case number; in reality, T_{EVAL} is in the range of 10 μs.

After the evaluation time, the following "information" is available on the SO node:

- if $V_{THR} > V_{READ}$, then $V_{SO} = V_{DD}$.
- if $V_{THR} < V_{READ}$, then $V_{SO} < V_{SEN} - V_{THN}$.

Now the V_{SO} voltage must be immediately "digitalized and frozen" in the latch structure, because of the very low charge retention of the SO node capacitor.

Petitioner Micron Ex-1021, 0210

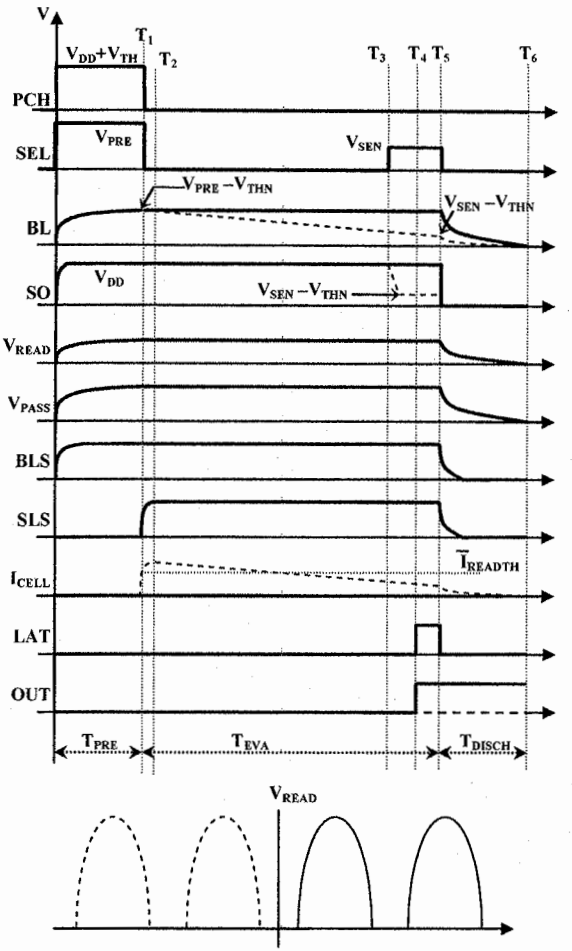


Fig. 8.9. Single read operation (SRO) timing diagram

A very often implemented latching structure is illustrated in Fig. 8.10, with its timing diagram.

Before every SRO, the OUT node of the latch is forced to ground (T_{RST} phase in Fig. 8.10). At the end of the reading operation, the LAT signal is set to V_{DD} and two cases are possible:

- if $V_{SO} = V_{DD}$, the $M_{LAT} - M_{SO}$ series is strongly conductive, forcing the OUT_N node to ground.
- if $V_{SO} = V_{SEN} - V_{THN}$, $M_{LAT} - M_{SO}$ series is not able to unbalance the latch and OUT_N remains at V_{DD}.

The result of the single read operation is finally stored into the latch. It's clear that the latch must be correctly sized in order to work as described above.

Petitioner Micron Ex-1021, 0211

Figure 8.11 shows the "latch static characteristic" V_{OUT} versus V_{SO} and "read static characteristic" V_{SO} versus V_{BL}. The point referred to as "C" corresponds to the end of the bitline precharge phase. The segment referred to as "CB" corresponds to a cell that does not sink enough current to discharge V_{BL} below the value $V_{SEN} - V_{THN}$ in T_{EVAL}. On the contrary, if the cell has discharged V_{BL} below $V_{SEN} - V_{THN}$, then SO node is shorted to the bitline (segment "OA" in Fig. 8.11b). In reality, the transition between points "A" and "B" is not so steep: the actual slope depends on the ratio between the bitline capacitance and the SO node capacitance (this ratio is in the order of 30–100).

The latch is well sized if:

- when $V_{SO} = V_{DD}$, then $V_{OUT} = V_{DD}$.
- when $V_{SO} < V_{SEN} - V_{THN}$, then $V_{OUT} = 0$ V.

Of course, these conditions must hold true PVT (process – voltage – temperature). Therefore, the threshold voltage of the latch (V_{THL}) should satisfy the following conditions:

$$V_{THL} > V_{SEN} - V_{THN};\ V_{THL} < V_{DD} \tag{8.13}$$

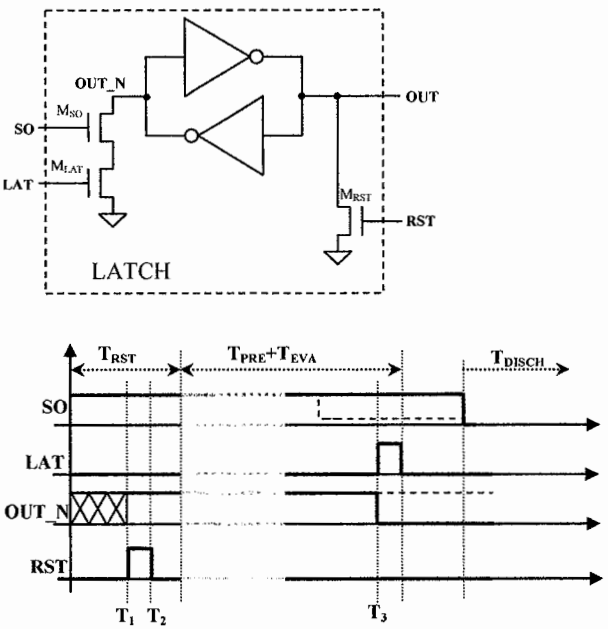


Fig. 8.10. First Latch circuital solution and its timing diagram

If Eq. (8.13) is satisfied, the latch doesn't affect the outcome of the read operation; in other words, the latch acts as a pure storage element. V_{PRE} and V_{SEN} are generated from a voltage regulator based on a well controlled reference

Petitioner Micron Ex-1021, 0212

voltage (typically a band-gap reference, Chap. 11). As a result, ΔV of Eq. (8.10) is a fixed value. If Eq. (8.13) is not satisfied, the latch influences the read operation. In this case, the term $V_{SEN} - V_{THN}$ of Eq. (8.10) has to be substituted with V_{THL}. V_{THL} PVT variations can range from 500 mV to 1 V. As a result, ΔV of Eq. (8.10) is not a fixed value.

In Fig. 8.12 another latch used in the page buffer circuits is sketched. It is made up by two three-state inverters (I1, I2) and a M_{EQ} equalization transistor. EN1 and EN2 can drive I1 and I2 into an high impedance state. In the latch of Fig. 8.10 the SO node drives the M_{SO} gate; instead, in Fig. 8.12 V_{SO} node is directly transferred on the OUT_N node through M_{LAT}. In Fig. 8.12, a possible timing diagram is also shown. During the first reading phase the latch is held in high impedance. Just before the end of the evaluation phase, there is an equalization phase (T_{EQ}): OUT_N and OUT nodes are shorted (approximately $V_{DD}/2$) through M_{EQ}. The equalization phase ends at time T_1. At time T_2, V_{SO} is transferred on OUT_N (M_{SO} is enabled). At T_3, the Inverter I1 is enabled (tri-state is released) and two cases are possible:

- if $V_{SO} = V_{DD}$, I1 sets the OUT signal to GND.
- if $V_{SO} = V_{SEN} - V_{THN}$, I1 sets the OUT signal to V_{DD}.

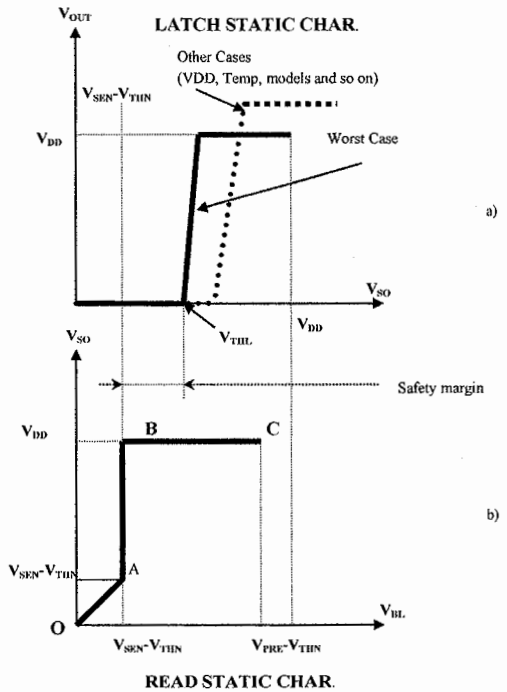


Fig. 8.11. (a) Latch characteristic; (b) read characteristic

At T_4 also I2 is enabled and the positive feedback of the latch takes place. Finally, at T_5, M_{S0} is disabled and the latch holds its state. The critical phase is

Petitioner Micron Ex-1021, 0213

definitely when I1 is enabled; also in this case, I1 inverter works correctly if its switching threshold voltage is always between V_{DD} and $V_{SEN} - V_{THN}$. In other words, it is like substituting the latch static characteristic of Fig. 8.11a with the inverter I1 static characteristic. Figure 8.13 shows a transistor level schematic of the three-state latch described in Fig. 8.12.

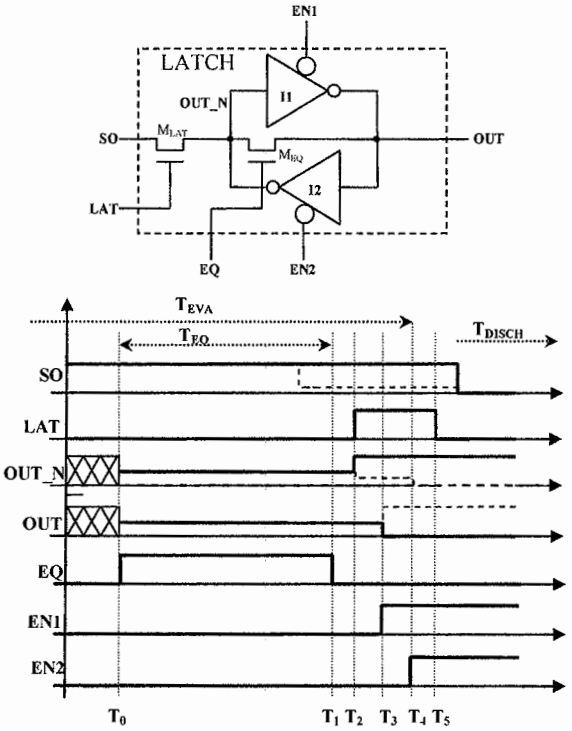


Fig. 8.12. Second latch circuital solution and its timing diagram

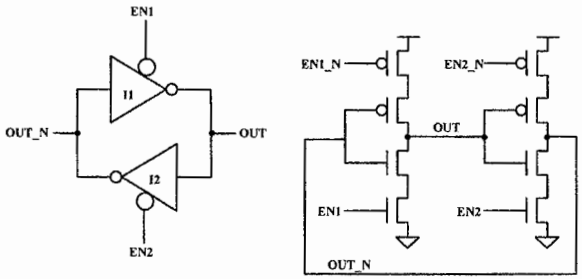


Fig. 8.13. Three-state latch

Petitioner Micron Ex-1021, 0214

8.2.1 Interleaving architecture

Given the Eq. (8.9), it is clear that the bitline capacitance has a direct influence on the average threshold current. C_{BL} must fulfill the following requirements:

- It must be a known parameter.
- It must be immune to external noise.

Figure 8.14 is a bitline cross-section showing the different contributions to C_{BL}:

- C_{AD} is the parasitic capacitor between the bitline and the lower plane (usually it is the wordline plane).
- C_{AU} is the parasitic capacitor between the bitline and the upper plane (usually it is the source-line plane).
- C_C is the parasitic capacitor between two adjacent bitlines.
- C_{C2} is the parasitic capacitor between a bitline and its second nearest bitline.

Therefore, C_{BL} can be written as:

$$C_{BL} = C_{AU} + C_{AD} + 2C_C + 2C_{C2} \tag{8.14}$$

The above mentioned contributions depend on the bitline geometrical values (width W, height H and spacing S in Fig. 8.14), on the distance between upper an lower ground levels and on the oxide thickness. These parameters are not uniform among different wafers, dice and even within the same die. However, a correct reading must be ensured. Once the maximum and minimum values of C_{BL} are known, it is possible (through Eq. (8.9)) to define a "cell working window" (Fig. 8.15) between the following I values:

$$I_{READTH\_MAX} = \frac{\Delta V \cdot C_{BL\_MAX}}{T_{EVAL}} \tag{8.15}$$

$$I_{READTH\_MIN} = \frac{\Delta V \cdot C_{BL\_MIN}}{T_{EVAL}} \tag{8.16}$$

Of course, looking at Fig. 8.15, this relation has to be satisfied:

$$I_{READTH\_MAX} < I_{SSAT} \tag{8.17}$$

If Eq. (8.17) is not true, the memory cell never discharges V_{BL} under ($V_{SEN} - V_{THN}$) and it is seen as written for any V_{READ}. The only way to solve this issue is to select a different set of V_{PRE}, V_{SEN} and T_{EVAL}.

In reality, as seen in Fig. 8.9, the bitline voltage and the cell current change over time during T_{EVAL}. It follows that the considerations on the average current and on the cell working window are only a starting point. It is necessary to relay on circuit simulators in order to verify that every parameter has been sized correctly.

Petitioner Micron Ex-1021, 0215

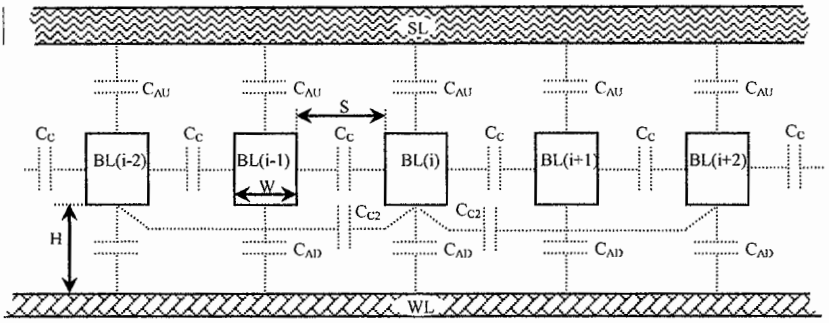


Fig. 8.14. Bitline parasitic capacitors

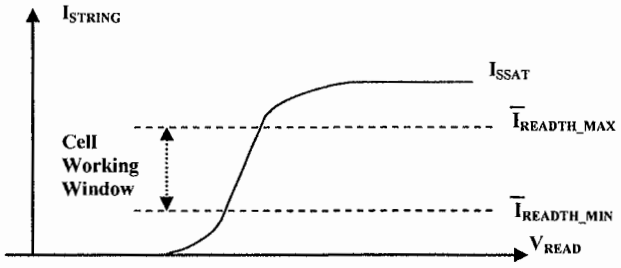


Fig. 8.15. Cell working window

In all the explained theory, another important assumption is that the bitline capacitor has one of its terminals fixed to ground. Actually, looking at Fig. 8.14, C_{BL} ground terminal is physically distributed over four nodes:

1. The upper plate, usually the source-line
2. The lower plate, usually the wordline or the source-line
3. The left bitline
4. The right bitline

During the evaluation time the first two nodes are forced at a fixed voltage. Instead, the adjacent bitlines could be discharged by the strings connected to them. Assuming a voltage noise ΔV_{BL} on the C_{BL} ground node, Eq. (8.9) becomes:

$$I_{READ\_TH} = \frac{(\Delta V + \Delta V_{BL}) \cdot C_{BL}}{T_{EVAL}}$$

and it is equivalent to inject, during the evaluation time, an average noise current equal to:

Petitioner Micron Ex-1021, 0216

$$I_{READ\_NOISE} = \frac{+\Delta V_{BL} \cdot C_{BL}}{T_{EVAL}}$$

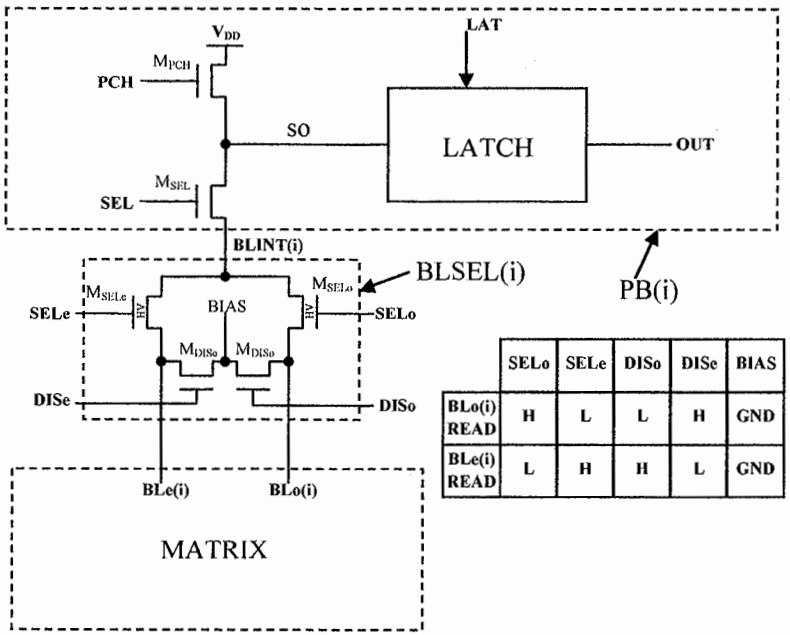
Depending on its value, ΔV_{BL} can also invalidate the read operation. In any case, ΔV_{BL} makes an apparent shift of the V_{THR}.

Let's suppose that BL(i + 1) and BL(i − 1) bitlines are discharged to $V_{PRE} - V_{SEN}$. On the coupled BL(i) the following voltage variation is induced:

$$\Delta V_{BL} = -\frac{C_C + C_C}{C_{BL}}(V_{PRE} - V_{SEN}) \tag{8.18}$$

This value is bigger if the adjacent bitlines are completely discharged to ground.

With the continuous bitline shrinking (W and S in Fig. 8.14), the coupling capacitances play an important role. In sub-40 nm NAND technologies they contribute 80–90% of the total bitline capacitance. To overcome this issue, the interleaving architecture is introduced. While the even (or odd) bitlines are read, the odd (or even) bitlines are forced to a fixed voltage (generally ground), acting as an electrical shield [5–7]. As shown in Fig. 8.16, M_{SLe} and M_{SLo} (bitline selectors) are placed between the bitlines and the page buffer. If the even bitlines BLe are read, M_{SELe} acts as a pass-transistor. Transistor M_{SELo} is turned off. The DISo signal turns on the M_{DISo} transistor, forcing the odd bitline BLo to the fixed BIAS voltage. M_{DISe} is turned off.



	SELo	SELe	DISo	DISe	BIAS
BLo(i) READ	H	L	L	H	GND
BLe(i) READ	L	H	H	L	GND

Fig. 8.16. Interleaving bitline architecture

Petitioner Micron Ex-1021, 0217

In order to minimize the power consumption, BIAS and the source line (SL) should be biased at the same voltage. In fact, these two nodes are shorted if a cell with $V_{THR} < V_{READ}$ belongs to the unselected bitlines. SL and BIAS are usually grounded during the reading operation.

With this architecture, the noise injection effects through the C_C coupling capacitors are eliminated. However, the coupling through C_{C2} (Fig. 8.14) is still in place. This contribution is not negligible: in the state-of-the-art technologies, C_{C2} contributes 5–10% of the total bitline capacitance. This problem is solved by the architecture described in Sect. 8.3.

8.2.2 Interleaving architecture: page buffer core design

There are two different ways of placing the page buffers and the bitline selectors around the matrix: in Fig. 8.17, they are located on one of the two sides (along the bitline direction) of the matrix. This architecture is called *single-side page buffer interleaving architecture*: the bitline pair BLe(i) and BLo(i) ends in the BLSEL(i) which has the output line BLINT(i) [5, 7, 10, 12, 14–17, 20, 22, 23]. Figure 8.18 shows the *double-side page buffer interleaving architecture*: the even/odd bitline pair ends alternatively in the up and down selectors [6, 8, 9, 11, 13].

The first solution is more compact, because it puts together all the page buffers in just one device's area. That gives a big advantage in terms of signal and power supply routing, IR drop and datapath signals. Instead, the second solution has the BLINT and the page buffer circuits less dense. In fact, the single-side solution needs a BLINT signal every two bitlines, while the double-side architecture needs one BLINT signal every four bitlines.

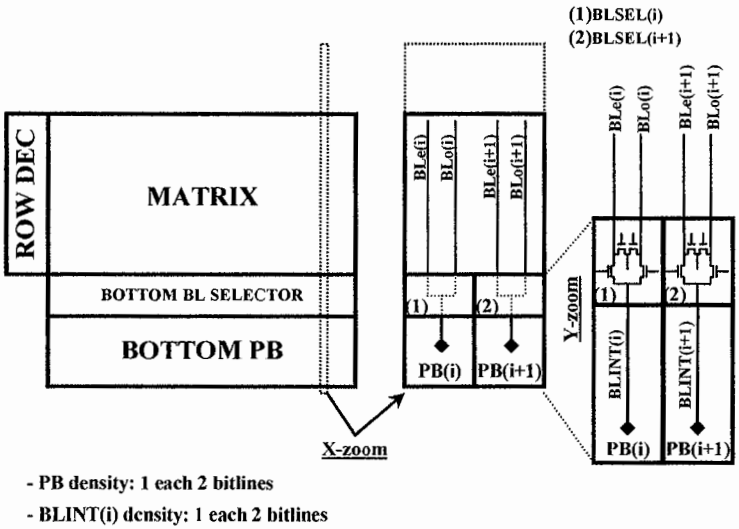


Fig. 8.17. Single side interleaving architecture

Petitioner Micron Ex-1021, 0218

It is important to take care of the layout density of the signals. In particular, the BLINT signal is one with the highest density in the NAND device, except bitlines (BLs) and wordlines (WLs). BLINT lines belong to a region usually called "Core Region" (CR), where the layout design rules are the most aggressive. Those rules can't be the ones used in the matrix, because CR is neither regular nor linear as the matrix is. For example, in the CR there are isolated curves, shapes and contacts, while in the matrix area there are only the regular BLs and WLs. Moreover, the matrix is manufactured with an aggressive and expensive lithography in comparison with the rest of the chip. Nowadays, BLs and WLs are defined with double patterning (pitch fragmentation) techniques, while single patterning lithography is used elsewhere. As the technology continues to shrink, it is very difficult to integrate a very dense BLINT region both from the lithography and the layout points of view. This is the reason why, sometimes, a double-side architecture is preferred.

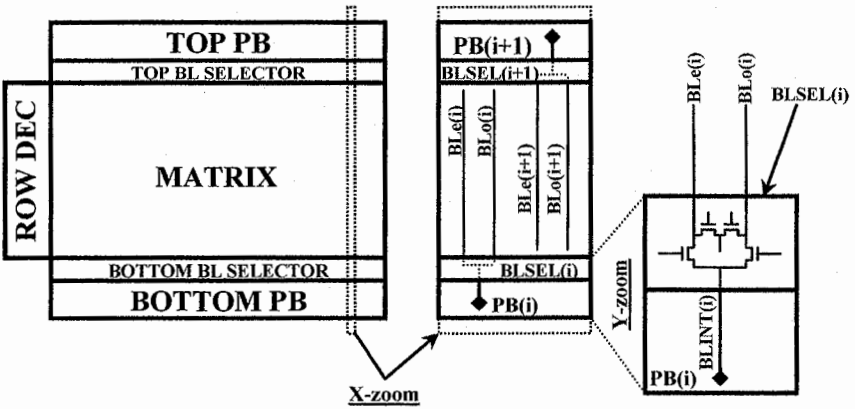


Fig. 8.18. Double side interleaving architecture

The page buffers' density can be instead reduced, using stacked architectures as sketched in Fig. 8.19. Page buffers are piled up on four rows, in order to reduce four times their density: from a page buffer every four BLs to a page buffer every 16 BLs. Obviously, the same solution can be used for the single-side architecture.

The page buffer is constituted by transistors having layout rules (strictly related to power supply voltage) that remain mostly unchanged through the different technological nodes. Therefore, as technology goes down to finer geometries, the page buffer has less space available in the orthogonal direction to the bitlines. For this reason, the number of page buffer rows increases more and more. In the state-of-the-art devices, it is usual to find structures with 8/16 page buffer rows.

Petitioner Micron Ex-1021, 0219

Figure 8.19 shows that the BLINT signals density is constant and it is not related to the number of page buffer rows. BL and BLINT densities vs. the technological node S are depicted in Fig. 8.20.

As an example, we consider a 36 nm NAND device, where an interleaving double-side architecture is adopted and the page buffers are stacked on eight rows. The page buffer is therefore drawn using a 32 bitlines pitch:

$$PBPITCH = (32+32)\cdot 36nm = 2{,}304\mu m \tag{8.19}$$

In Fig. 8.21 a layout area corresponding to the bitline selector (BLSEL) region is illustrated. In the upper part of the figure there are the bitlines and the gates of two BLSEL HVNMOS transistors. The bitlines are made with the second metal layer (metal2). In the lower part of the figure the BLINT signals, drawn with the first metal layer (metal1), are shown. Actually, the two figure overlap, but are here split to make them clearer.

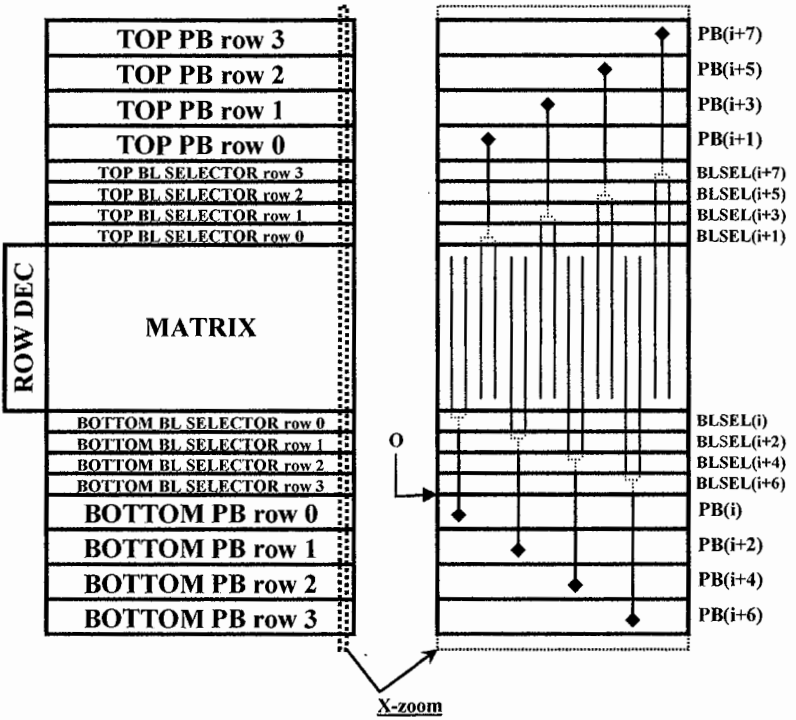


Fig. 8.19. Double-side interleaving architecture with four stacked PB rows

Petitioner Micron Ex-1021, 0220

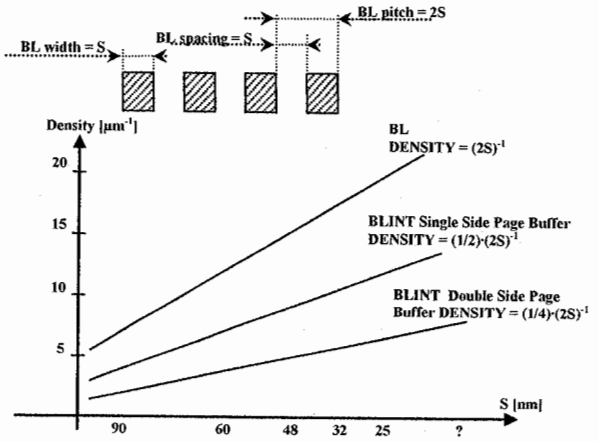


Fig. 8.20. BL and BLINT density versus the technology node S

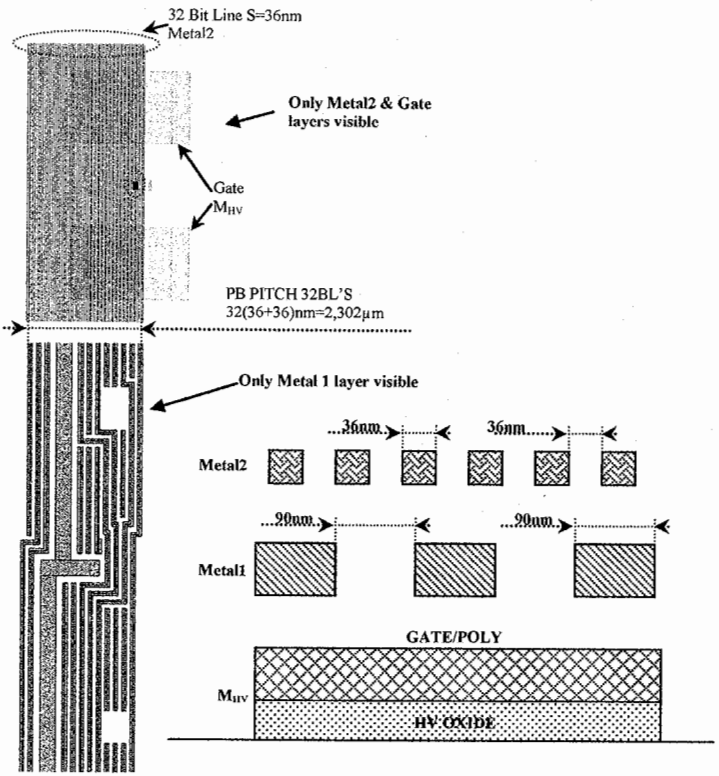


Fig. 8.21. 36 nm technology node: BL and BLINT layout in BLSEL circuitry

Petitioner Micron Ex-1021, 0221

Since we are considering a double-side architecture, the BLINT density should be four times lower than the bitlines density; therefore, BLINT could be drawn with width and spacing of 144–144 nm. Actually, also the BIAS signal of Fig. 8.16 and the connection of the bitlines to the HV transistors of the bitline decoder (the small circled rectangle in Fig. 8.21) need some space. A trade off between layout, design and technology is to choose a value of 90–90 nm as the minimum size for the metal1 signals in the Core Region. With a single-side architecture, the starting point would have been of 72–72 nm, but, for the same reasons, the minimum dimension of the BLINT signal would have been reduced further.

The BLINT and BIAS signals routing is illustrated in Fig. 8.22. The depicted region is the one with the highest BLINT density. It is located at the interface between the bitline selectors and the page buffers (the area indicated by the "O" arrow in Fig. 8.19). The dummy lines are added only for lithographic reasons. The VIRPWR signal is equivalent to the BIAS signal.

Figure 8.23 shows a page buffer drawn on a 32 bitlines pitch: gates, active areas and contacts of the NMOS/PMOS transistors are depicted. It should be noticed that, in the orthogonal direction to the bitlines, only two transistors can be placed side by side. A small portion of the metal1 layer is also shown. It is possible to observe the high density of the signals and the use of lithographic dummy lines.

Finally, Fig. 8.24 shows a portion of a page buffer's row, made up by the structures of Fig. 8.23.

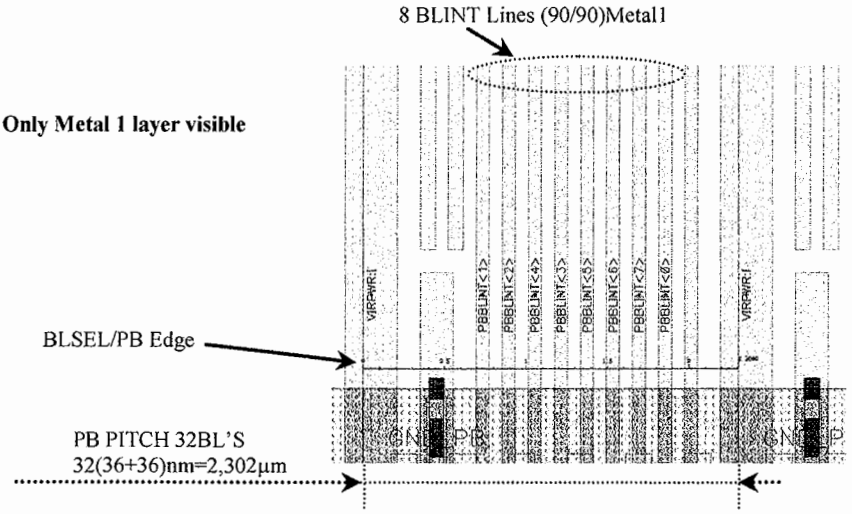


Fig. 8.22. 36 nm BLINT density in BLSEL/PB boundary area

Petitioner Micron Ex-1021, 0222

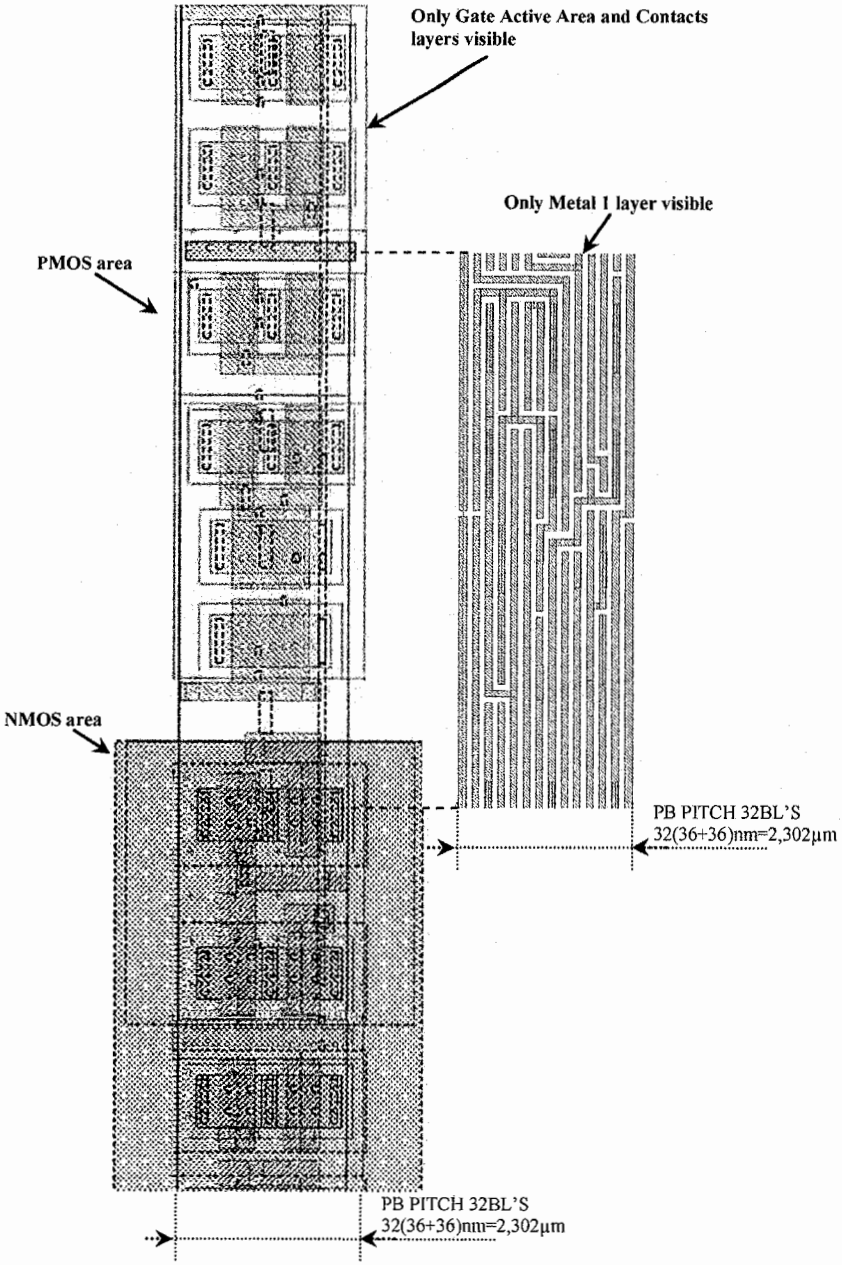


Fig. 8.23. PB layout in 36 nm NAND technology: gate, contact and metal1 layers are shown

Petitioner Micron Ex-1021, 0223

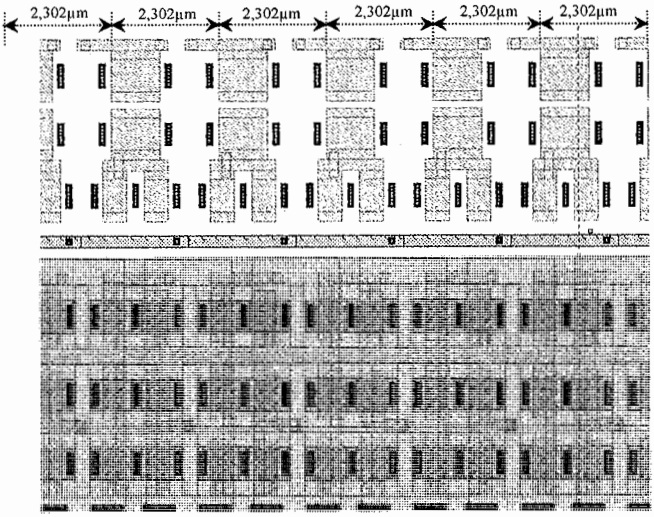


Fig. 8.24. 36 nm layout portion of a PB row: gate and metal1

8.3 Reading techniques with time-constant bitline biasing

Figure 8.25 shows the capacitive effects of four bitlines adjacent to the bitline BL_0. Each of the bitlines has its own self capacitance C_{BL00}, C_{BL11} and C_{BL22} respectively. The pair of BL_0 and BL_1 adjacent bitlines have mutual capacitance C_{BL01}. The pair of BL_0 and BL_2 adjacent bitlines have mutual capacitance C_{BL02}. Each capacitance contributes to the current flowing along BL_0. In particular, the currents due to each bitline self capacitance are:

$$i_{BLC00} = C_{BL00} \cdot \frac{\partial V_{BL0}}{\partial t}$$

$$i_{BLC11} = C_{BL11} \cdot \frac{\partial V_{BL1}}{\partial t}$$

$$i_{BLC22} = C_{BL22} \cdot \frac{\partial V_{BL2}}{\partial t}$$

Similarly, the cross currents due to the pair BL_1 and BL_2 are:

$$i_{BLC01} = C_{BL01} \cdot \frac{\partial (V_{BL0} - V_{BL1})}{\partial t}; \quad i_{BLC02} = C_{BL02} \cdot \frac{\partial (V_{BL0} - V_{BL2})}{\partial t}$$

Petitioner Micron Ex-1021, 0224

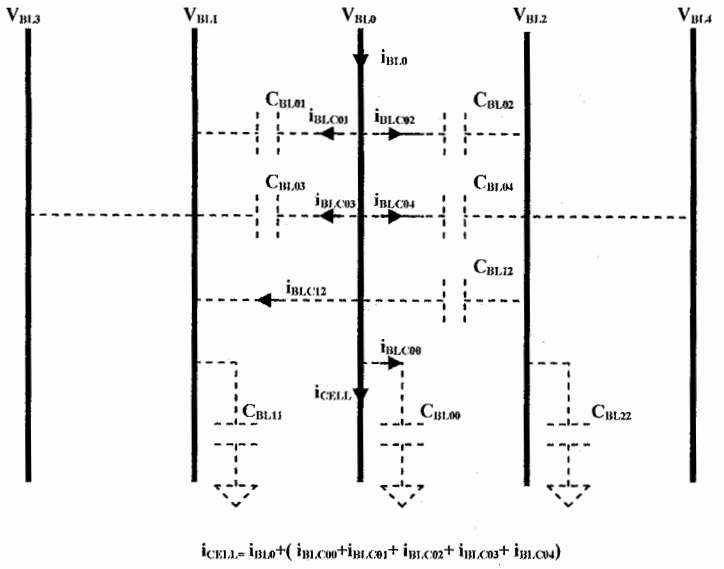


Fig. 8.25. Parasitic capacitors among adjacent bitlines

It follows that the current flowing along BL_0 is:

$$i_{BL0} \approx i_{CELL} + \left(i_{BLC00} + i_{BLC01} + i_{BLC02}\right) \tag{8.20}$$

where i_{CELL} is the cell current. Eq. (8.20) is an approximation, because it includes only the adjacent bitlines contributions. Generally speaking, considering BL_0, there will be also non adjacent bitlines contributions: C_{BL03} on the left and C_{BL04} on the right:

$$i_{BLC03} = C_{BL03} \cdot \frac{\partial(V_{BL0} - V_{BL3})}{\partial t}; \quad i_{BLC04} = C_{BL04} \cdot \frac{\partial(V_{BL0} - V_{BL4})}{\partial t}$$

Therefore, Eq. (8.20) becomes:

$$i_{BL0} \approx i_{CELL} + \left(i_{BLC00} + i_{BLC01} + i_{BLC02} + i_{BLC03} + i_{BLC04}\right) \tag{8.21}$$

Considering all the contributions given by all the non adjacent bitlines, we have:

$$i_{BL0} = i_{CELL} + \left(i_{BLC00} + i_{BLC01} + i_{BLC02} + i_{BLC03} + i_{BLC04} + \ldots\ldots\right) \tag{8.22}$$

In state-of-the-art technologies, all the non adjacent bitlines contributions can be estimated as about 10% of the adjacent bitlines contribution.

Petitioner Micron Ex-1021, 0225

As explained in Sect. 8.2, in the interleaving architecture the discharge rate of the parasitic bitline capacitor is measured. The adjacent bitlines are grounded: in this way, the crosstalk between them is eliminated. This sensing technique results in a time varying $V_{BL0} = V_{BL0}(t)$. Looking at Fig. 8.25, the whole capacitance of BL_0 has to be considered: $C_{BL00} + C_{BL01} + C_{BL02}$. Moreover, interleaving architecture does not eliminate all the currents due to non adjacent bitlines, e.g. those associated with C_{BL03} and C_{BL04}. These currents start playing a significant role at around 30 nm technologies. All the above mentioned contributions influence the transient behavior of the bitline voltage.

A possible solution to the bitline coupling issue is to bias all the bitlines to a constant voltage during the sensing phase. In this case, Eq. (8.21) becomes:

$$i_{BL0} = i_{CELL}$$

In other words, the current flowing through the bitline is equivalent to the one of the NAND string. Having the same voltage on all the bitlines allows their simultaneous read: the *All BitLine* (ABL) architecture is now introduced.

8.3.1 All BitLine (ABL) architecture

The approach is similar to the one used in NOR Flash memories, where, during the evaluation phase, the bitline voltage is forced to a constant value. Anyway, as explained in Sect. 8.1, with NAND currents of few tens of nanoAmpere, it is very hard to implement traditional NOR-like sensing methods.

The sensing technique is basically the same used in the interleaving architecture. An intentionally placed capacitor is used instead of the C_{BL} bitline parasitic capacitor. In the ABL world the page buffer is also called sense amplifier (SA).

Figure 8.26 shows the main elements of the ABL sense amplifier. The page buffer latch is replaced by a voltage comparator with a V_{THSA} trigger voltage. The other elements are those ones already described in the interleaved architecture, but here used in a different way. The capacitor C_{SO} is involved in the integration of the cell current: it can be done using either MOS gates or poly-poly capacitors.

Figure 8.27 shows the timings used in a single reading operation. The precharge phase is similar to the one described for the interleaving architecture, where M_{PCH} and M_{SEL} gates are biased to $V_{DD} + V_{THN}$ and V_{PRE} respectively. M_{HV} HVNMOS has the behavior already described and, during the single reading operation phase, works as pass transistor. The signals which drive the string gates (V_{READ}, V_{PASS} and BLS) are activated as usual. Instead SLS signal is immediately activated in order to stabilize the bitlines during the precharge phase. In fact, if the SLS had been activated during the evaluation phase, there would have been a voltage drop on those bitlines with an associated sinking current string.

Petitioner Micron Ex-1021, 0226

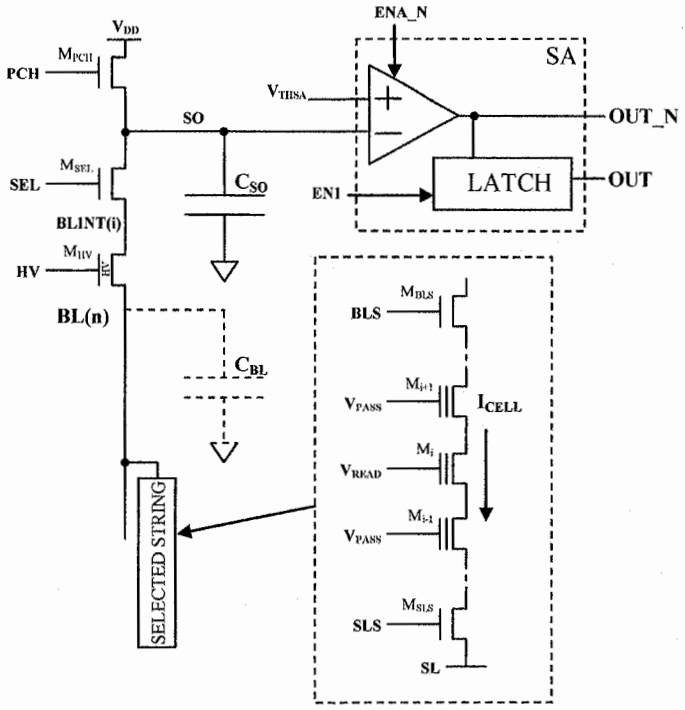


Fig. 8.26. ABL sense amplifier

The precharge final condition

$$V_{BL} = V_{PRE} - V_{THN} \tag{8.23}$$

is, therefore, valid only for the bitlines which have an associated string in a non conductive state.

Equation (8.23) should be replaced by:

$$V_{BL} = V_{PRE} - V_{THN} - \Delta \tag{8.24}$$

where Δ is the voltage drop on the bitlines resistance (typical values are in the order of hundreds of kiloOhm up to 1 MΩ). NAND strings having deep erased cells might create voltage drops of a few hundreds of milliVolt, but also cells absorbing tens of nanoAmpere generate drops of tens of milliVolt. By activating SLS during the evaluation phase, there would have been some noise on the adjacent bitlines due to the above mentioned voltage drops. In this way the voltage on the bitlines wouldn't have been constant anymore. Δ includes also the necessary M_{SEL} overdrive. On the other hand, the connection of SLS during the precharge phase determines a significant current consumption, which will be discussed in Chap. 9.

Petitioner Micron Ex-1021, 0227

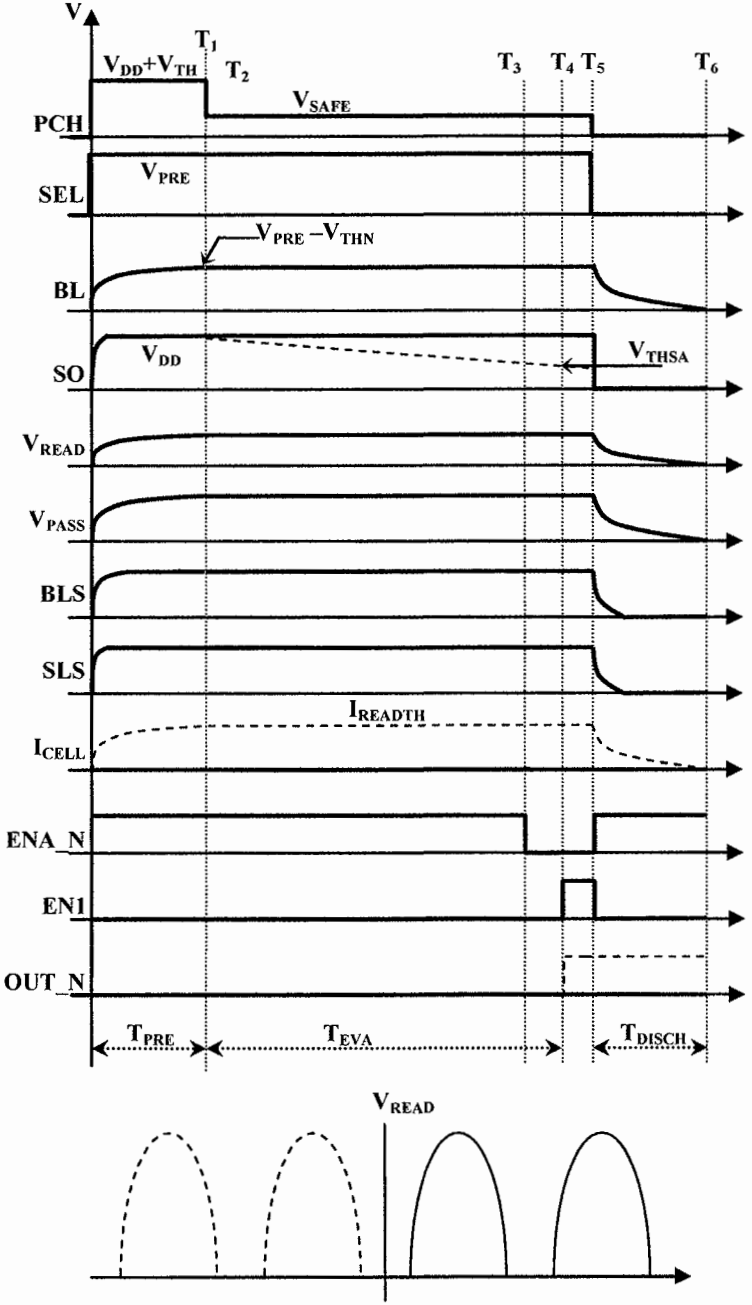


Fig. 8.27. ABL single read operation (SRO) timing diagram

Petitioner Micron Ex-1021, 0228

At the end of the precharge phase (T_1), the bitlines are biased to a constant voltage and V_{SO} is equal to V_{DD}. At this point, M_{PCH} is switched off and the evaluation phase starts. Actually, M_{PCH} is biased to a V_{SAFE} voltage value. Later on, the meaning of V_{SAFE} will result clearer. When M_{PCH} is switched off, the cell current (through M_{PRE}) discharges the C_{SO} capacitor. If, during the evaluation time, $V_{SO} < V_{THSA}$ (trigger voltage of Fig. 8.26 comparator), then OUT_N switches (dotted lines in Fig. 8.27). The "threshold current" I_{READTH} is defined as:

$$I_{READTH} = \frac{\Delta V \cdot C_{SO}}{T_{EVAL}} \tag{8.25}$$

where

$$\Delta V = V_{DD} - V_{THSA} \tag{8.26}$$

In this case, there is no need to use an average current as it is in Definition 8.3. Observe that, because the bitline is biased to a fixed voltage, a constant current flows.

Equations (8.25) and (8.26) are similar to Eqs. (8.9) and (8.10); therefore, it is possible to extrapolate the evaluation time:

$$T_{EVAL} = \frac{\Delta V \cdot C_{SO}}{I_{READTH}} \tag{8.27}$$

Given the same read currents, it follows that the ratio between Eqs. (8.11) and (8.27) is determined by the ratio between C_{BL} and C_{SO}. C_{BL} is a parasitic element and has a value of 2–4 pF. Instead, C_{SO} is a design element and has typical values around 20–40 fF, i.e. two orders of magnitude lower than C_{BL}. The reduction of the evaluation time from 10 µs to hundreds of ns is another advantage of the All Bitline architecture.

8.3.2 ABL architecture: sensing design

Before illustrating the sensing circuits, let's analyze the V_{SAFE} meaning. Suppose that the current flowing along the bitline is able to discharge V_{SO} to GND (0 V). Before that happens, M_{SEL} drain (i.e. SO node) goes down to $V_{PRE} - V_{THN}$. At this point, M_{SEL} is in the ohmic region and, therefore, it works as a pass transistor, connecting the SO node to the bitline. As a consequence, the cell current discharges both C_{SO} and C_{BL}. It follows that the bitline is not forced at a constant voltage.

A possible solution is to bias the M_{PCH} gate with a voltage V_{SAFE}, in order to make M_{PCH} behave as a clamp transistor of the SO voltage. The following relation must be valid:

$$V_{SAFE} - V_{THN} \geq V_{PRE} - V_{THN} \Rightarrow V_{SAFE} > V_{PRE} \tag{8.28}$$

Petitioner Micron Ex-1021, 0229

This clamp value must not influence the current integration on the SO capacitor, i.e. the clamping function can't take place above the V_{THSA} trigger voltage:

$$V_{SAFE} - V_{THN} \leq V_{THSA} \tag{8.29}$$

Therefore, from Eqs. (8.28) and (8.29), the following conditions must hold true:

$$V_{PRE} - V_{THN} \leq V_{SAFE} - V_{THN} \leq V_{THSA} \tag{8.30}$$

In Fig. 8.28 a possible sense amplifier implementation and its related timing is illustrated.

During the precharge phase, OUT_N is grounded through M_{SET} and then left floating, while EN1 drives I2 in a high-impedance state, until the end of the sensing operation. At the end of the evaluation time, M_{EN} is switched on forcing ENA_N to ground. At this point two cases are possible:

1. if $V_{SO} > V_{DD} - V_{THP}$, M_{SA} is off (OUT_N='0', OUT='1')
2. if $V_{SO} < V_{DD} - V_{THP}$, M_{SA} is on (OUT_N='1', OUT='0'). It corresponds to the dotted lines in the timing diagram of Fig. 8.28.

V_{THP} is the threshold voltage of the PMOS M_{SA}. The trigger voltage of this circuit is:

$$V_{THSA} = V_{DD} - V_{THP} \tag{8.31}$$

Replacing Eq. (8.31) in Eq. (8.26) we obtain:

$$\Delta V = V_{DD} - \left(V_{DD} - V_{THP}\right) = V_{THP} \tag{8.32}$$

and from Eq. (8.25):

$$I_{READ\_TH} = \frac{V_{THP} \cdot C_{SO}}{T_{EVAL}} \tag{8.33}$$

In other words, the trigger voltage of the ABL sense amplifier depends only on the threshold voltage of a transistor (V_{THP}). It is clear that the trigger voltage doesn't depend on V_{DD} as it is for Eq. (8.25).

Finally, the I2 three-state is released and the OUT_N logic value is stored in the latch. As already explained for V_{SAFE}, the SO signal should not be discharged below $V_{PRE} - V_{THN}$. It follows:

$$V_{THSA} > V_{PRE} - V_{THN} \tag{8.34}$$

This is automatically verified if Eq. (8.30) holds true.

Petitioner Micron Ex-1021, 0230

Of course, the latch I1–I2 can also be used without exploiting the three-state option. In this case, M_{EN}, M_{SA} and the latch must be correctly sized so that the latch unbalancing always takes place when V_{SO} goes below $V_{DD} - V_{THP}$.

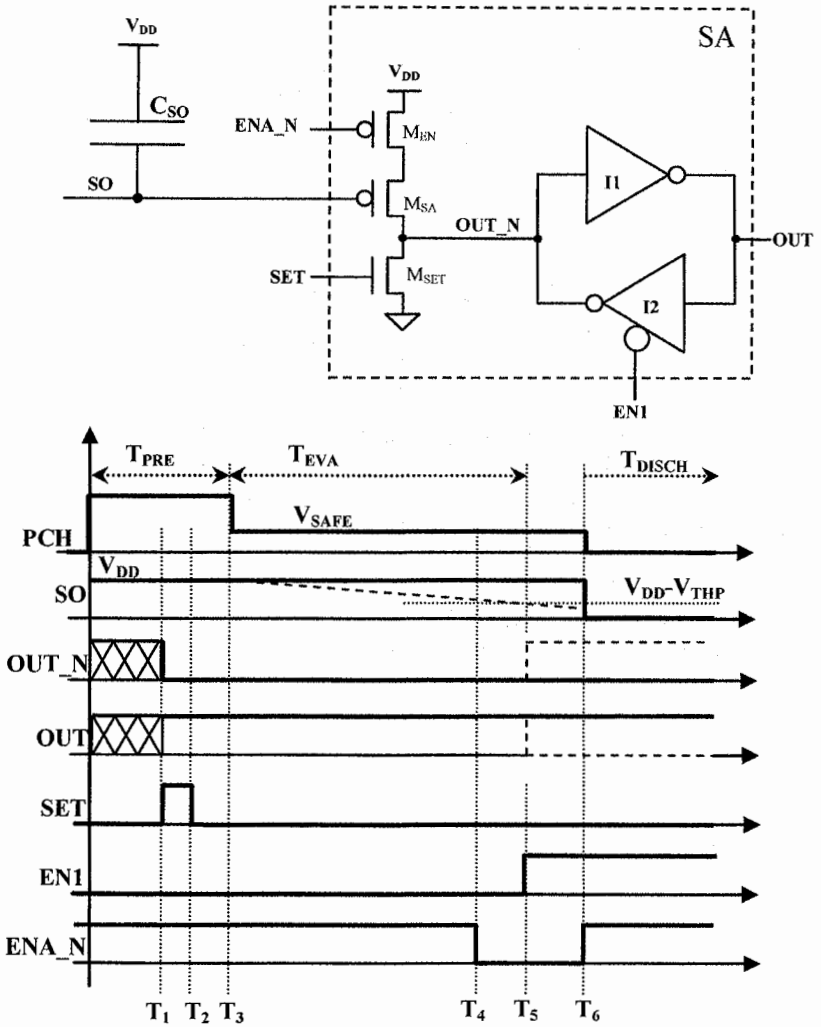


Fig. 8.28. ABL sense amplifier and its timing diagram

8.4 ABL versus interleaving architecture

In this section a comparison between ABL and the interleaving architecture is presented [25].

Petitioner Micron Ex-1021, 0231

A) Evaluation time reduction

ABL: the evaluation time is reduced from tens of microsecond to some hundreds of nanosecond. In this way, both the read time and the program time (during the verify phases) improve.

B) Bitline noise coupling reduction

ABL: during the evaluation phase, the bitlines are biased to a fixed voltage, so that the coupling disturbance between adjacent and non adjacent bitlines is eliminated.

C) All bitlines read

The bitlines are read simultaneously. Given the same number of cells belonging to a wordline (same wordline length), a SRO with ABL provides twice the number of bits compared to a SRO performed with interleaving architecture.

D) Energy saving and cell working point

During the precharge phase, ABL forces all the bitlines at the same voltage. In this way, the parasitic coupling capacitors are not charged, while the area capacitors are. In the conventional architecture, instead, both the coupling capacitors and the area capacitors are charged. As already said, the coupling capacitance constitutes 80–90% of the total capacitance. Therefore, there is an energy saving of the same percentage. Actually, also in the ABL architecture the bitlines are not all charged at the same voltage (see Eq. (8.24) and the analysis of the Δ factor). Anyway, the advantage on the interleaving architecture remains remarkable.

In the interleaving architecture, the average current consumption during the precharge phase is:

$$I_{PRE} = C_{BL} \frac{\Delta V}{T_{PRE}} n_{BL} \tag{8.35}$$

where n_{BL} represents the number of bitlines charged in parallel. In the state-of-the-art technologies this number is around 128 k.

Given C_{BL} equal to 2 pF, ΔV equal to 1 V and a precharge time of 10 µs, we obtain an average current consumption of about 20 mA. Therefore, a reduction of 80–90% has a big impact on the device's current consumption.

Another advantage in terms of energy-saving is that the ABL architecture needs a lower bitline biasing level compared to the interleaving architecture. Figure 8.29 shows that, during the evaluation time, the cell working point moves from the A characteristic to the B characteristic, because of the bitline voltage discharge. In the ABL architecture, the cell can be directly biased with a lower bitline voltage, e.g. equal to the one indicated in Fig. 8.29. In terms of current saving, it is equal to a ΔV lower level in Eq. (8.35).

Petitioner Micron Ex-1021, 0232

A third advantage in energy saving comes from point C) assuming the same wordline length. The interleaving architecture requires two single reading operations (even and odd): the bitlines are charged and discharged two times. ABL reads all the bitlines with only one bitlines charge. Figure 8.30 illustrates the bitline behavior in both architectures. The advantages associated with ABL are: lower bitline voltage, shorter evaluation time and simultaneous read of even/odd cells with consequent access time saving.

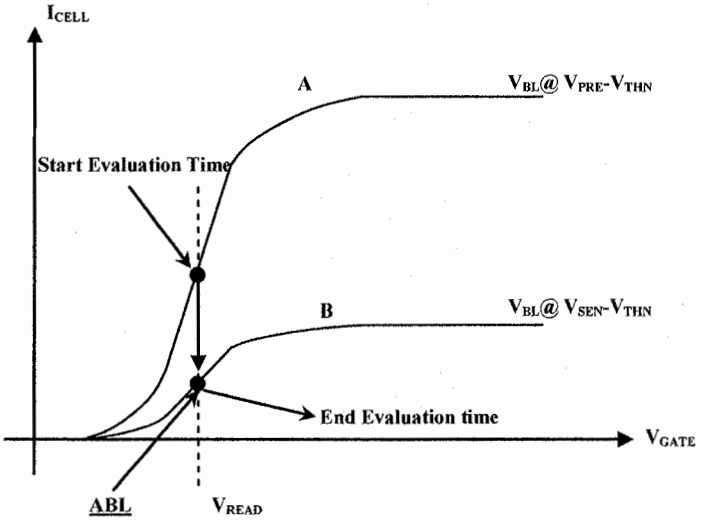


Fig. 8.29. Cell working point: ABL versus interleaving

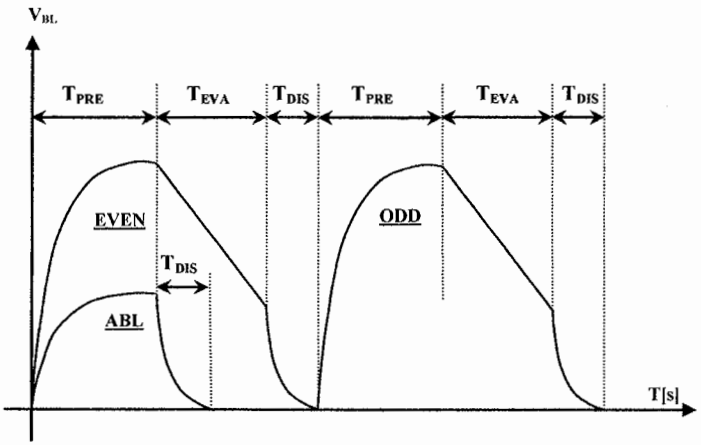


Fig. 8.30. Bitline voltage: ABL versus interleaving

Petitioner Micron Ex-1021, 0233

As previously described, the drawback of the ABL architecture is the static current consumption during the precharge phase. This is due to cells with $V_{THR} << V_{READ}$. Besides the current consumption, these cells are also responsible for the disturbance on the common source-line (see Chap. 10) [26].

E) Floating-gate-coupling reduction during program

The ABL architecture has also the advantage of programming and verifying the cells in a parallel way, minimizing the neighboring floating gate coupling (the so-called "Yupin Effect") [25, 27–29].

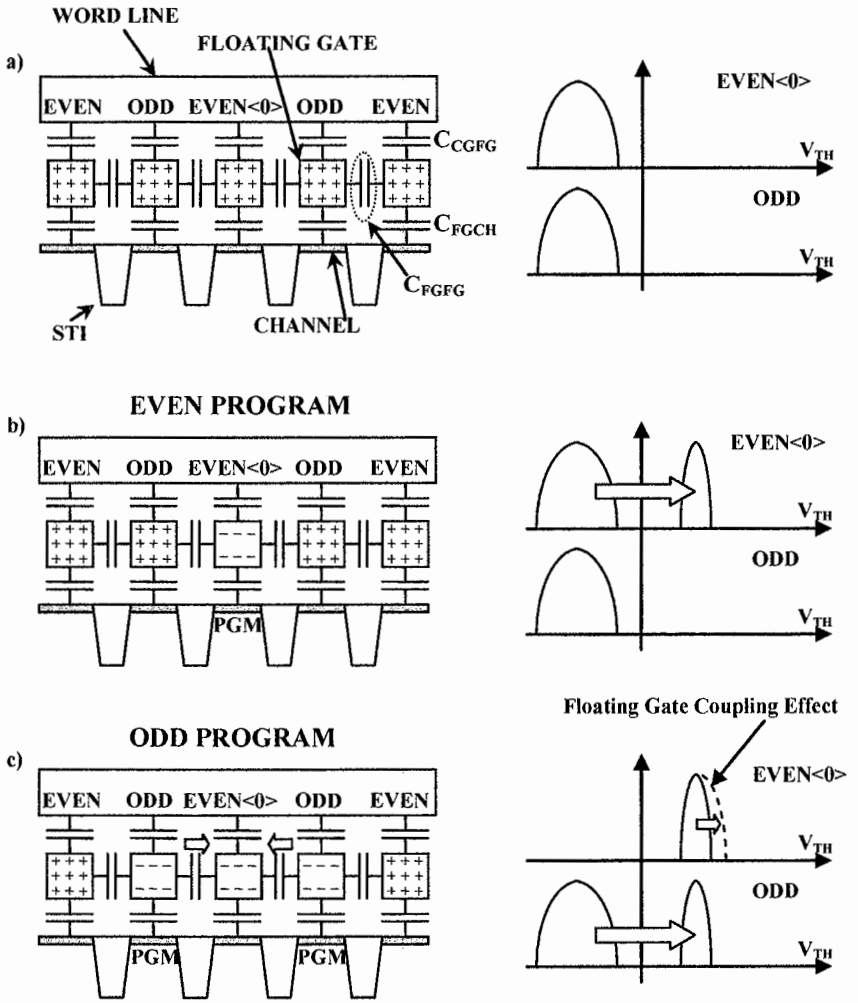


Fig. 8.31. Interleaving architecture: floating-gate-coupling effect during a program operation

Petitioner Micron Ex-1021, 0234

In Fig. 8.31a the wordline cross-section is shown. There are five erased cells and their coupling capacitors C_{FGFG}. Suppose to program the even cell in the center of Fig. 8.31b (EVEN<0>). Electrons accumulate in the programmed cell's floating gate and the cell moves from the erased distribution to the written one. In Fig. 8.31c the neighbors odd cells are programmed and, because of their negative accumulated charge, there is a change in the electric field. This is reflected as a voltage threshold shift for the middle cell that appears more programmed. The overall even cells distribution suffer an enlargement on the right side (dotted lines in Fig. 8.31c).

The above described effect is becoming more evident with the continuous technology shrink. Indeed, C_{FGFG} has a more and more important role compared to the other cell capacitors (for example C_{CGFG} and C_{FGCH}). It becomes a serious problem for the devices with more than 1 bit per cell, where the written distributions have a smaller space available. In fact, wider distributions lead to read margins reduction.

ABL sensing reduces the floating gate coupling by programming all adjacent memory cells simultaneously (Fig. 8.32). Therefore, ABL seems to be the right architecture for multibit per cell devices [25].

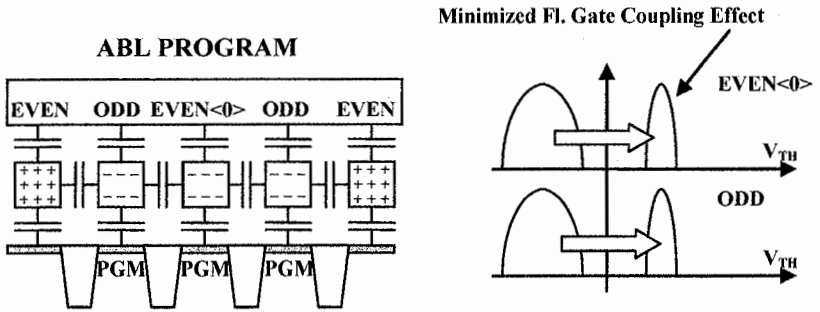


Fig. 8.32. Floating-gate-coupling effect during program with ABL architecture

F) Read/Program Disturbances

Read and program operations cause a stress to all the unselected cells because of the V_{PASS} voltage. In the interleaving architecture even and odd cells are read and programmed in two phases. In the ABL architecture the problem is reduced, because all the cells belonging to the same wordline are read and programmed in parallel.

G) Sensing circuitry density

ABL needs one sense amplifier for each bitline. Therefore, the number of reading circuits is doubled compared to the interleaving architecture. Figure 8.33 shows an ABL double side architecture, where the BLINT signals density is equal to the one of the interleaving single-side architecture of Fig. 8.17. An ABL single side

Petitioner Micron Ex-1021, 0235

architecture is difficult to implement, because it needs a BLINT signals density equal to that of the bitlines. As a matter of fact, in the state-of-the-art technologies all the devices with an ABL architecture have a double side structure [18, 19, 21, 24]. BLINT signal density is summarized in Fig. 8.34.

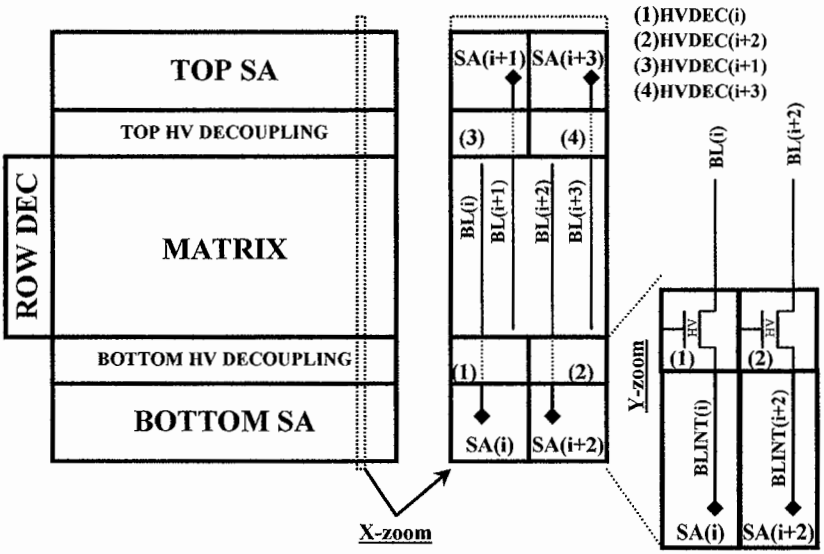


Fig. 8.33. Double-side ABL architecture

However, ABL reduces the HV transistors area. If there are four HV transistors composing BLSEL(i) for each page buffer (Fig. 8.16), the ABL needs only one. Considering that the sense amplifier's density is twice the page buffer's density, it follows that the number of HV transistors is halved. Moreover, we don't need DISCH and SEL anymore, reducing the routing complexity and the control circuits. Above all, the BIAS signal is no longer necessary and, as previously shown in Fig. 8.22, it gives more space to the BLINT routing.

H) Current integrator element

The interleaving architecture uses the parasitic capacitor C_{BL}, while ABL is based on C_{SO}, which is intentionally introduced in the sense amplifier scheme. A variation in the C_{BL} value changes the sensed current (Eq. (8.9)) and the cell working window (Eqs. (8.15)–(8.17)). The same is true for ABL, as seen in Eq. (8.25), but the capacitance spread is lower for C_{SO}. As a result, ABL exploits in a better way the available NAND string current.

Petitioner Micron Ex-1021, 0236

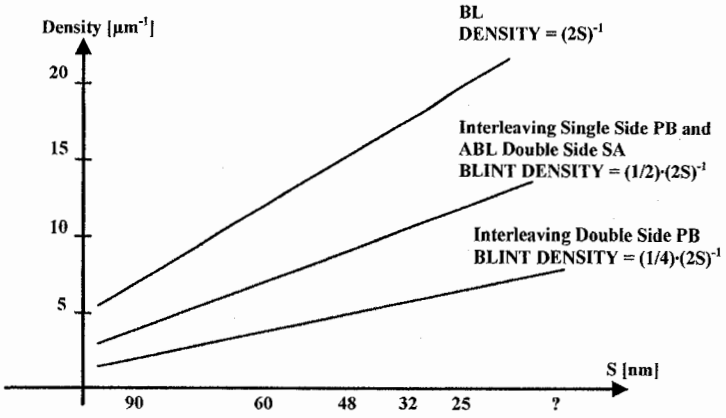


Fig. 8.34. BLINT density ABL versus interleaving

References

1. Kang-Deog Suh et al. *A 3.3 V 32 Mb NAND Flash Memory with Incremental Step Pulse Programming Scheme*, IEEE Journal of Solid-State Circuits, Vol. 30, No. 11, Nov. 1995, pp. 1149–1156.
2. Yoshihisa Iwata et al. *A 35 ns Cycle Time 3.3 V Only32 Mb NAND Flash EEPROM* IEEE IEEE Journal of Solid-State Circuits, Vol. 30, No. 11, Nov 1995, pp. 1157–1164.
3. Jin-Ki Kim et al. *A 120-mm 64-Mb NAND Flash Memory Achieving 180 ns/Byte Effective Program Speed,* IEEE Journal of Solid-State Circuits, Vol. 32, No. 5, May 1997, pp. 670–680.
4. Yasuo Itoh, et al. *An Experimental 4Mb CMOS EEPROM with a NAND Structured Cell* Solid-State Circuits Conference, 1989. Digest of Technical Papers. 36th ISSCC., 1989 IEEE International, Feb. 1989, Page(s):134–135.
5. Tanaka, T. et al. *A quick intelligent page-programming architecture and a shielded bitline sensing method for 3 V-only NAND Flash memory*, IEEE Journal of Solid-State Circuits, Vol. 29, No. 11, Nov. 1994, pp. 1366–1373.
6. Tae-Sung Jung et al. *A 3.3 V 128 Mb multi-level NAND Flash memory for mass storage applications* Solid-State Circuits Conference, 1996. Digest of Technical Papers. 43rd ISSCC.,1996 IEEE International, Feb. 1996, Page(s):32–33, 412.
7. Imamiya, K. et al. *A 130 mm² 256 Mb NAND Flash with shallow trench isolation technology* Solid-State Circuits Conference, 1999. Digest of Technical Papers. ISSCC. 1999 IEEE International, Feb. 1999, Page(s): 112–113, 412.
8. Taehee Cho et al. *A 3.3V 1Gb Multi-Level NAND Flash Memory with Non-Uniform Threshold Voltage Distribution* Solid-State Circuits Conference, 2001. Digest of Technical Papers. ISSCC. 2001 IEEE International, Feb. 2001, Page(s): 80–81, 410.
9. June Lee et al. *A 130 mm² A 1.8V 1Gb NAND Flash Memory with 0.12μm STI Process Technology,* Solid-State Circuits Conference, 2002. Digest of Technical Papers. ISSCC. 2002 IEEE International, Feb. 2002 Page(s): 104–105, 450, Vol. 1.

Petitioner Micron Ex-1021, 0237

10. Hiroshi Nakamura et al. *A 125mm² 1Gb NAND Flash Memory with 10MB/s Program Throughput* Solid-State Circuits Conference, 2002. Digest of Technical Papers. ISSCC. 2002 IEEE International, Feb. 2002 Page(s): 82–83, 411.
11. June Lee et al. *A 90-nm CMOS 1.8-V 2-Gb NAND Flash Memory for Mass Storage Applications* IEEE Journal of Solid-State Circuits, Vol. 38, No. 11, Nov 2003, pp. 1934–1942.
12. Seungjae Lee et al. *A 3.3V 4Gb Four-Level NAND Flash Memory with 90nm CMOS Technology* Solid-State Circuits Conference, 2004. Digest of Technical Papers. ISSCC. 2004 IEEE International, Feb. 2004 Page(s): 52–53, 513, Vol. 1.
13. R. Micheloni et al. *A 4Gb 2b/cell NAND Flash Memory with Embedded 5b BCH ECC for 36MB/s System Read Throughput* Solid-State Circuits Conference, 2006. Digest of Technical Papers. ISSCC. 2006 IEEE International, Feb. 2006 Page(s): 497–506.
14. Takahiko Hara et al. *A 146-mm2 8-Gb Multi-Level NAND Flash Memory With 70-nm CMOS Technology* IEEE Journal of Solid-State Circuits, Vol. 41, No. 1, Jan 2006, pp. 161–169.
15. Dae-Seok Byeon et al. *An 8Gb Multi-Level NAND Flash Memory with 63nm STI CMOS Process Technology* Solid-State Circuits Conference, 2005. Digest of Technical Papers. ISSCC. 2005 IEEE International, Vol. 1, Feb. 2005, pp. 46–47.
16. Ken Takeuchi et al. *A 56-nm CMOS 99-mm² 8-Gb Multi-Level NAND Flash Memory With 10-MB/s Program Throughput* IEEE Journal of Solid-State Circuits, Vol. 42, No. 1, Jan 2007, pp. 219–232.
17. Dean Nobunaga et al. *A 50nm 8Gb NAND Flash Memory with 100MB/s Program Throughput and 200MB/s DDR Interface* Solid-State Circuits Conference, 2008. Digest of Technical Papers. ISSCC. 2008 IEEE International, Feb. 2008, pp. 426–427, 625.
18. Raul-Adrian Cernea et al. *A 34 MB/s MLC Write Throughput 16 Gb NAND With All Bit Line Architecture on 56 nm Technology* IEEE Journal of Solid-State Circuits, Vol. 44, No. 1, Jan 2009, pp. 186–194.
19. Yan Li et al. *A 16Gb 3b/ Cell NAND Flash Memory in 56nm with 8MB/s Write Rate* Solid-State Circuits Conference, 2008. Digest of Technical Papers. ISSCC. 2008 IEEE International, Feb. 2008, pp. 506–507, 632.
20. Seung-Ho Chang et al. *A 48nm 32Gb 8-level NAND Flash memory with 5.5MB/s program throughput* Solid-State Circuits Conference, 2009. Digest of Technical Papers. ISSCC. 2009 IEEE International, Feb. 2009, pp. 240–241,241a.
21. Takuya Futatsuyama et al. *A 113mm² 32Gb 3b/cell NAND Flash memory* Solid-State Circuits Conference, 2009. Digest of Technical Papers. ISSCC. 2009 IEEE International, Feb. 2009, pp. 242–243.
22. Raymond Zeng et al. *A 172mm² 32Gb MLC NAND Flash memory in 34nm CMOS* Solid-State Circuits Conference, 2009. Digest of Technical Papers. ISSCC. 2009 IEEE International, Feb. 2009, pp. 236–237.
23. Shibata, N et al. *A 70 nm 16 Gb 16-Level-Cell NAND Flash Memory* IEEE Journal of Solid-State Circuits, Vol. 43, No. 4, April 2008, pp. 929–937.
24. C. Trinh et al. *A 5.6MB/s 64Gb 4b/Cell NAND Flash memory in 43nm CMOS* Solid-State Circuits Conference, 2009. Digest of Technical Papers. ISSCC. 2009 IEEE International, Feb. 2009, pp. 246–247.
25. Raul Adrian Cernea *Nonvolatile Erasable Flash NAND Memories*, SanDisk Corporation, Santa Clara Valley (SCV), IEEE Solid State Circuits Society, April 2008.

Petitioner Micron Ex-1021, 0238

26. Raul-Adrian Cernea et al. U.S. Patent No. 7443757 – *Non-volatile memory and method with reduced bitline crosstalk errors* Assignee: SanDisk Corporation (Sunnyvale, CA).
27. Jae-Duk Lee et al. *Effects of floating-gate interference on NAND Flash memory celloperation* Semicond. Res. & Dev. Center, Samsung Electron. Co. Ltd., Gyunggi; Electron Device Letter, IEEE May 2002, Vol. 23, No. 5, Page(s): 264–266.
28. Chen , et al *High density non-volatile Flash memory without adverse effects of electric field coupling between adjacent floating gates* U.S. Patent No. 5867429 Assignee: SanDisk Corporation (Sunnyvale, CA).
29. Raul-Adrian Cernea et al. U.S. Patent No. 6987693, 7239551 – *Non-volatile memory and method with reduced neighboring field errors* Assignee: SanDisk Corporation (Sunnyvale, CA).

Petitioner Micron Ex-1021, 0239

9 Parasitic effects and verify circuits

L. Crippa[1] *and R. Micheloni*[2]

9.1 Background pattern dependency (BPD)

As anticipated in the introduction to Chap. 8, the current–voltage (I/V) characteristic of the cell is influenced by the data being programmed afterwards in the other cells. In order to avoid the source degeneration effect (load variation between cell source terminal and SL), the cells of a string are sequentially programmed starting from the source-side cell all the way to the bitline-side cell. For instance (Fig. 9.1), cell M_0 is programmed first, followed by M_1, M_2 and so on. The drain-side load variation has a "modulation" effect on the I/V characteristic of the cell in terms of both reduction of the saturation current I_{SSAT} and variation of the slope (Fig. 8.3). Such a modulation effect is known as Background Pattern Dependency (BPD). The physical threshold voltage of M_0, V_{TH}, obviously does not change (neglecting the floating gate coupling effect $M_1 - M_0$).

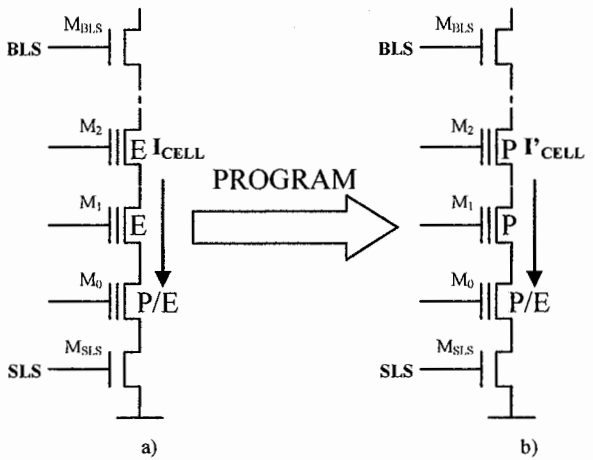


Fig. 9.1. Cell M_0 current modulation due to Background Pattern Dependency (BPD)

[1] Forward Insights, luca.crippa@ieee.org
[2] Integrated Device Technology, rino.micheloni@ieee.org

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_9,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0240

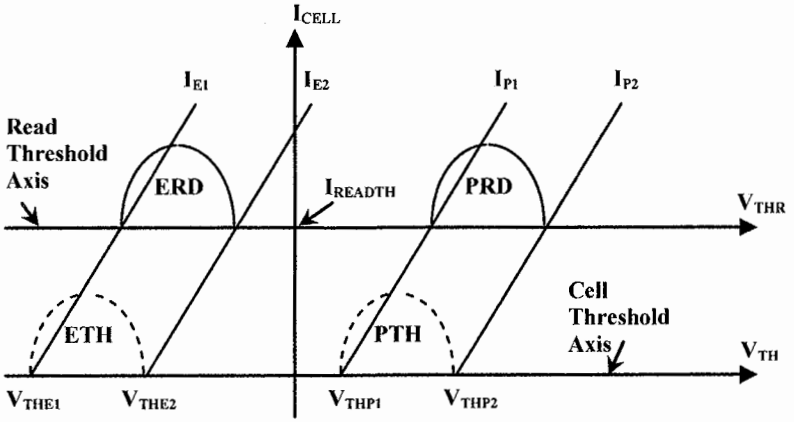


Fig. 9.2. Correspondence between V_{TH} and V_{THR}

Programmed state of cell M_0 is therefore verified using a current characteristic (I_{CELL} in Fig. 9.1a) which is different from the one available at the end of the program phase on the remaining cells of the string (I'_{CELL} in Fig. 9.1b).

Figure 9.2 shows the distributions and the I/V characteristics of cell M_0 before programming the remaining cells of the string. V_{TH} axis represents the physical threshold voltage of the cell. V_{THR} axis crosses the current axis I_{CELL} at I_{READTH}. I_{READTH} is the threshold current derived from Definition 8.3 and from Eq. (8.9) or Eq. (8.25). Therefore, V_{THR} is the axis of the *Read Cell Threshold Voltage* according to the Definition 8.2 (Sect. 8.2).

From a graphical point of view, read threshold voltage is the intersection between the I/V characteristic of the cell and V_{THR} axis (Fig. 9.2). I_{E1}, I_{E2}, I_{P1} and I_{P2} represent the I/V characteristics at the boundaries of both the erased (ETH) and programmed (PTH) distributions: V_{THE1}, V_{THE2}, V_{THP1} and V_{THP2} are the corresponding physical threshold voltages. Distributions ETH and PTH relate, respectively, to ERD and PRD, i.e. their projections on the V_{THR} axis.

Figure 9.3 shows the situation after all the remaining cells of the string have been programmed (floating gate coupling is neglected). Characteristics I_{E2} and I_{P2} gets shifted onto I'_{E2} and I'_{P2} respectively. The unwanted effect on V_{THR} axis is a widening of the final distributions of M_0, as shown by the dotted lines.

The fact that I_{E1} and I_{P1} are shifted in the same way (thus resulting in a right-bound rigid shift of the distribution, featuring no widening) should not mislead. The pattern according to which the cells are programmed is always different, therefore all the possible cases should be taken into account: for instance, if all the remaining cells of the string are left un-modified in their erased state, current characteristic of M_0 is not modulated.

All the characteristics of the cells suffer from BPD, even if the effect is less and less evident as the cell is nearer to transistor M_{BLS}: the last cell, the one next to the bitline selector, is the only one not suffering from BPD.

Petitioner Micron Ex-1021, 0241

Figure 9.4 shows that a reduction of the working point ($I'_{READTH} < I_{READTH}$) causes a reduction of the BPD effect (shown as BPD'): this is equivalent to shift the axis of the current thresholds towards the axis of the physical thresholds.

An advantage of ABL sensing (Chap. 8) is that it can operate at lower currents, thus mitigating the BPD.

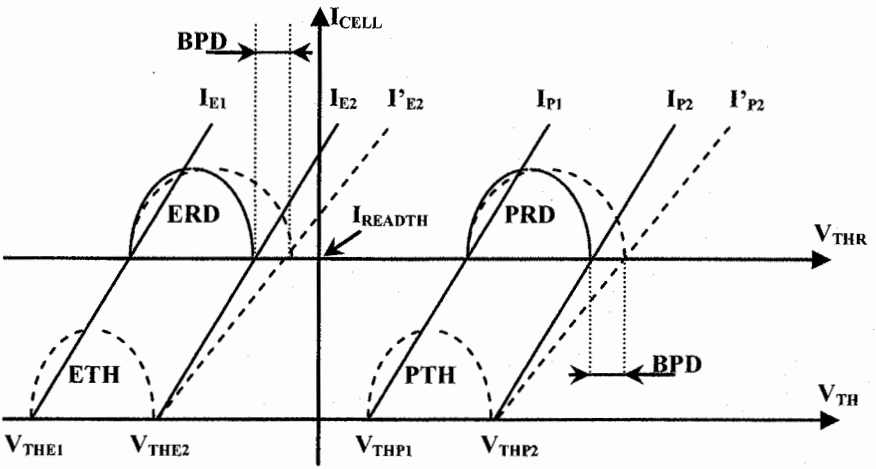


Fig. 9.3. BPD effects on Read Threshold Axis (V_{THR})

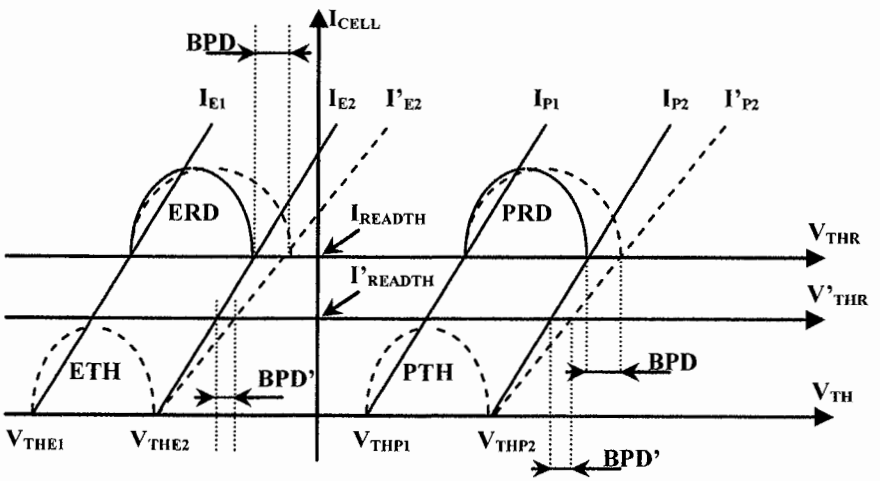


Fig. 9.4. Lower cell current working point reduce BPD effect

Petitioner Micron Ex-1021, 0242

9.2. Reading techniques for negative sensing

Every read and verify operation of the programmed distributions are carried out applying to the gate of the cell a V_{READ} voltage which is greater than (or equal to) zero. As a consequence, programmed distributions can only have positive V_{THR} (see Definition 8.2).

NAND technology does not require the generation of negative voltages inside the device. Therefore, negative V_{THR} cannot be verified using the sensing methodologies described in Chap. 8. The availability of a method to determine the position of the cells belonging to the erased distributions brings several advantages.

First of all, it is useful to verify that the string has been really erased: in fact, an additional erase pulse could be required in order to take the cell in the desired distribution. Sometimes, a successful erase is not possible due to cells with technological defects.

Because of the spread of the electrical and technological parameters, each cell reacts in a different way to the single erase pulse: the width of the erased distribution is usually in the order of some Volts. Widths of programmed distributions are usually around some hundreds of milliVolt.

The ability of controlling and reducing the width of the erase distribution (by means of a proper soft-programming algorithm) brings several benefits.

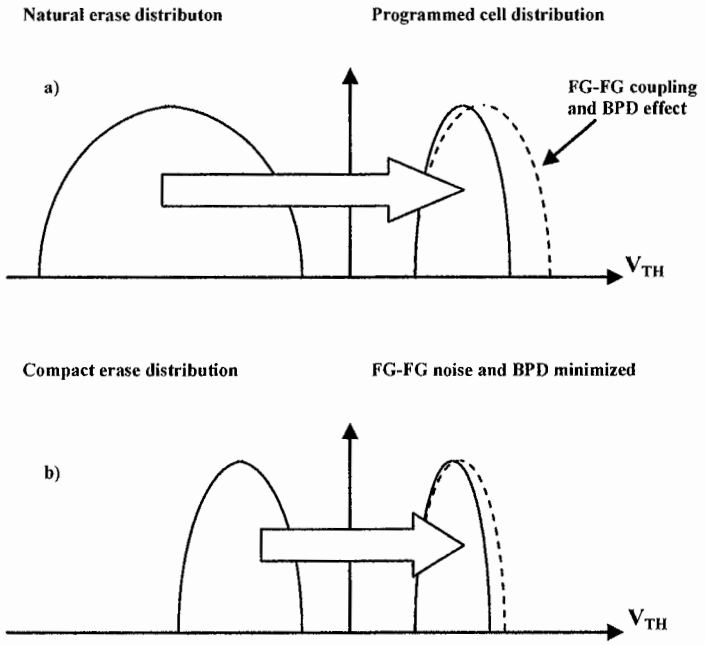


Fig. 9.5. Benefit of a compact erased distribution

Petitioner Micron Ex-1021, 0243

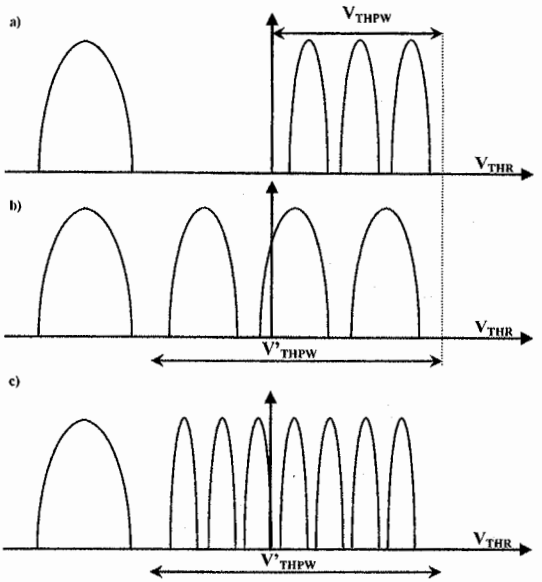


Fig. 9.6. Negative sensing expand program threshold window

1. Reduction of current consumption during the read/verify phase: cells/strings which are deeply erased sink a lot of current. Furthermore, as shown in Sect. 9.3, current injection in the resistive network of source line SL originates both noise and reduction of read margins.
2. Floating gate coupling reduction and BPD reduction: during programming, the V_{TH} of the cells shifts. Disturb to the adjacent cells is proportional to the V_{TH} shift. Deeply erased cells exhibit higher threshold voltage shifts (Fig. 9.5).
3. Extension of the programmed window V_{THPW} of Fig. 9.6a to V'_{THPW} of Fig. 9.6b. The advantage is that either the distance between distributions can be increased (thus improving read margins) or the width of the distributions can be increased (higher programming steps). Alternatively the number of programmed levels can be increased (Fig. 9.6c).

In the following section, some techniques that allow the read of negative V_{THR} are described.

9.2.1 Conventional negative verify

Figure 9.7a shows the ETH distribution after the electrical erase, seen on V_{TH} axis, and its corresponding representation on V_{THR} axis, ERD. I_E represents the I/V characteristic of the least erased string, biased with $V_{PASS} = 5$ V. Program-after-erase operation (Chap. 12) is used to avoid over-erase of the NAND cells: a series

Petitioner Micron Ex-1021, 0244

of program pulses (Chap. 3) is applied until the erase verify operation is successfully completed.

The most straightforward way of performing an erase verify is to do a program verify with $V_{READ} = 0$ V. From a graphical point of view, performing a verify operation with $V_{READ} = 0$ V and $V_{PASS} = 5$ V is equivalent to shift the I_E characteristic until it crosses axis $V_{THR} = V_{READ} = 0$ V (equivalent to axis I_{CELL}) at I_{READTH}. Therefore, ERD distribution shifts in ERD0 distribution, leaning on 0 V axis (Fig. 9.7a).

It is possible to "stop" the shift of ERD distribution to a negative voltage instead of reaching the 0 V axis: that is, a verify operation with a different V_{PASS} voltage. Biasing the string with $V_{READ} = 0$ V and $V_{PASS} = 0$ V, its I/V characteristic becomes the dotted line I'_E of Fig. 9.7b.

After electrical erase, I'_E characteristic crosses I_{CELL} axis at a current higher than I_{READTH}. Program-after-erase operation is cyclically repeated until erased distribution crosses I_{CELL} axis at I_{READTH} (Fig. 9.7c). Verify the cell with $V_{READ} = 0$ V and $V_{PASS} = 0$ V is the same as verifying it with $V_{PASS} = 5$ V and a negative V_{READ} equal to V_{WLEQ} (as shown in Fig. 9.7c).

An alternate method to emulate a read operation with a negative V_{READ} is to use a verify time T'_{EVAL} shorter than the nominal one (T_{EVAL}). When compared to standard read conditions, a reduction of T_{EVAL} determines an increase of the trigger threshold current:

$$I'_{READTH} = \frac{\Delta V \cdot C_{BL}}{T'_{EVAL}} > I_{READTH} = \frac{\Delta V \cdot C_{BL}}{T_{EVAL}} \tag{9.1}$$

Figure 9.8 shows the equivalence of a read operation with a reduced evaluation time with respect to a standard read ($V_{READ} = 0$ V and $V_{PASS} = 5$ V) performed with a negative wordline biasing at V_{WLEQ}.

Both methods can be implemented using the reading architectures described in Chap. 8. It is also possible to implement both methods at the same time ($V_{PASS} = 0$ V and T'_{EVAL}) in order to lower the equivalent negative voltage (V_{WLEQ}) further.

Unfortunately the unrelenting shrink of the technologies and the consequent decrease of the saturation current of the string (Chap. 8) have severely limited the efficiency of these methods:

- Dotted line I'_E of Fig. 9.7 could have a saturation level which is slightly greater than (or even lower than) I_{READTH}.
- Threshold current defined by Eq. (9.1) cannot be greater than saturation current of the string. Therefore it is not possible to lower the value of T_{EVAL} indefinitely.

As a consequence, V_{WLEQ} gets closer and closer to 0 V, dangerously moving the erased distribution next to the programmed one.

Petitioner Micron Ex-1021, 0245

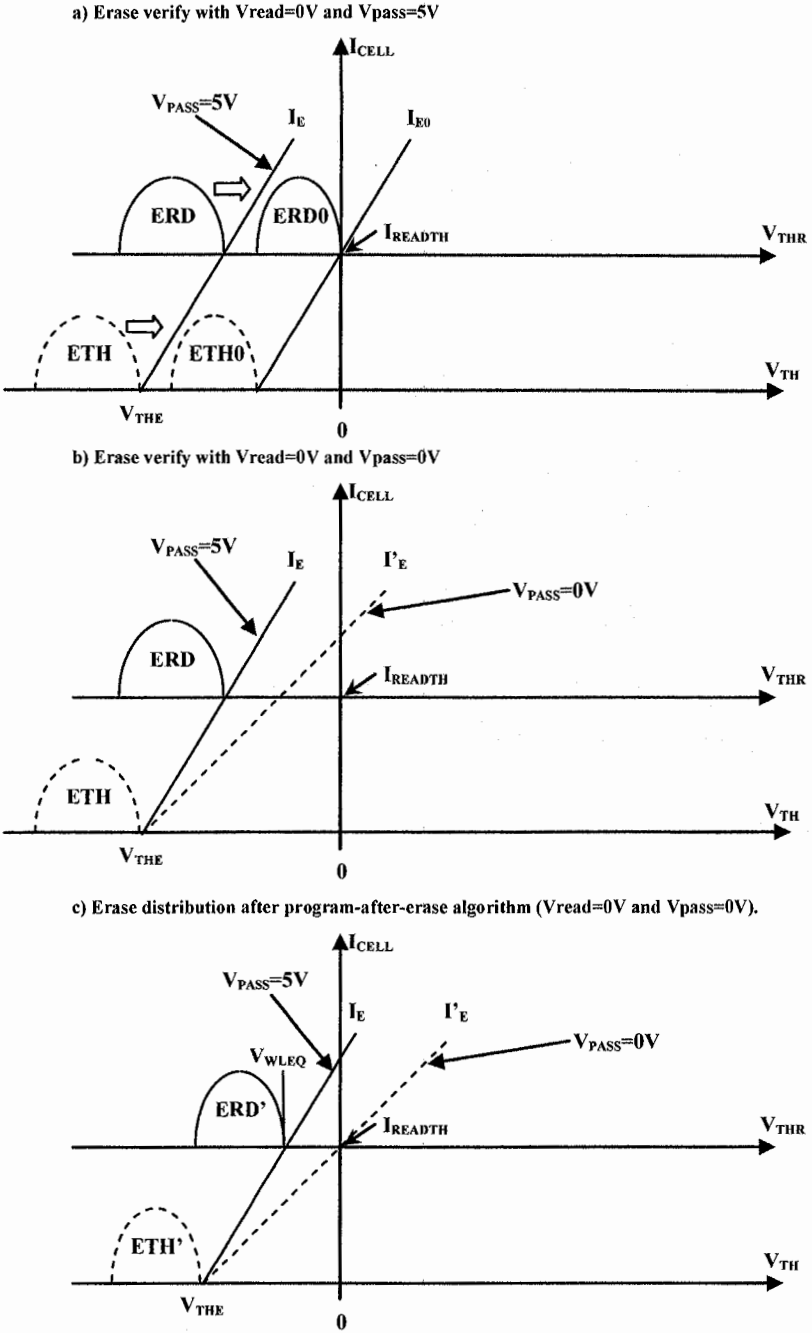


Fig. 9.7. Negative sensing with $V_{READ} = 0$ V

Petitioner Micron Ex-1021, 0246

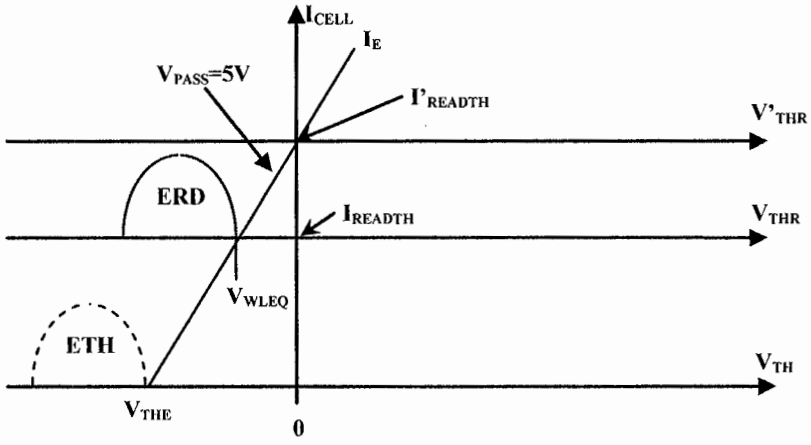


Fig. 9.8. Negative sensing with T_{EVAL} modulation

9.2.2 Absolute negative sensing

Let's consider a NMOS transistor with a negative threshold voltage V_{THN}: if its source is biased with a positive voltage and its gate is at ground, the voltage at its drain can rise up to the absolute value of V_{THN}, assuming a small current flowing (Fig. 9.9).

This statement can be easily proven by applying the equation for the NMOS in saturation region to the transistor shown in Fig. 9.9.

$$I = K \cdot (V_{GD} - V_{THN})^2 = K \cdot (V_G - V_D - V_{THN})^2 \tag{9.2}$$

It should be noted that the drain takes the place of the source in Eq. (9.2) since the two terminals are swapped (MOS transistor is a bidirectional element).

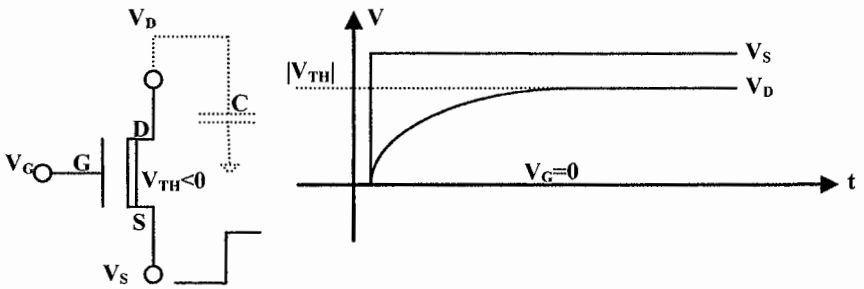


Fig. 9.9. Negative threshold voltage NMOS under positive source biasing

Petitioner Micron Ex-1021, 0247

Furthermore, under the assumption of biasing the NMOS using a gate voltage equal to zero, the transistor operates in saturation region if

$$V_S \geq V_{TH} \tag{9.3}$$

Applying a voltage step to the source, a current flow towards the drain is established. As shown in Fig. 9.9, the drain is floating and connected to a capacitor. Thus, V_D keeps growing until the current I no longer flows through the MOS:

$$I = 0 \tag{9.4}$$

Substituting Eq. (9.4) in Eq. (9.2) with $V_G = 0$ V, it follows

$$K \cdot (-V_D - V_{THN})^2 = 0 \Rightarrow V_D = -V_{THN} \tag{9.5}$$

Disregarding the body effect, the absolute value of the threshold voltage of the MOS is the voltage of the drain node.

Let's assume that the cell of a NAND string takes the place of the MOS in Fig. 9.9. The other cells of the string, which act as pass transistors, can be disregarded.

The situation is now more complex, since the cell is a floating gate MOS (Fig. 9.10).

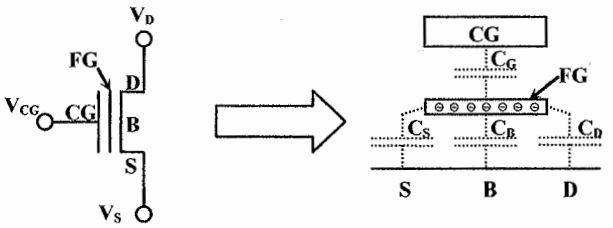


Fig. 9.10. Floating gate NMOS coupling capacitors

Current I is now equal to:

$$I = K \cdot (V_{FG} - V_S - V_{THFG})^2 \tag{9.6}$$

where V_{THFG} is the physical threshold voltage seen by the floating gate [1].

The value of the floating gate voltage V_{FG} depends on the charge (Q_{FG}) stored inside it and on the voltages of control gate V_G, bulk V_B, source V_S and drain V_D:

$$V_{FG} = \alpha_G \cdot V_G + \alpha_B \cdot V_B + \alpha_S \cdot V_S + \alpha_D \cdot V_D + \frac{Q_{FG}}{C_{TOT}} \tag{9.7}$$

Petitioner Micron Ex-1021, 0248

where

$$C_{TOT} = C_G + C_B + C_S + C_D \tag{9.8}$$

$$\alpha_G = \frac{C_G}{C_{TOT}};\ \alpha_B = \frac{C_B}{C_{TOT}};\ \alpha_S = \frac{C_S}{C_{TOT}};\ \alpha_D = \frac{C_D}{C_{TOT}} \tag{9.9}$$

To a first approximation, contribution of α_S and α_D can be neglected. Substituting Eq. (9.7) inside Eq. (9.6) with $V_G = 0$ V and $V_B = 0$ V, it follows that:

$$I = K \cdot (-V_S - V_{THFG} + \frac{Q_{FG}}{C_{TOT}}) = K \cdot (-V_S - V'_{THFG}) \tag{9.10}$$

where V'_{THFG} is the physical threshold voltage seen by the floating gate, including the contribution of Q_{FG}

$$V'_{THFG} = V_{THFG} - \frac{Q_{FG}}{C_{TOT}} \tag{9.11}$$

Similarly to the previous case, drain and source nodes are indeed swapped ($V_S \rightarrow V_D$) in Eq. (9.10). After the transient, current is zero and V_D can be calculated:

$$(-V_D - V'_{THFG}) = 0 \Rightarrow V_D = -V'_{THFG} \tag{9.12}$$

When the contributions due to the capacitive coupling with source and drain are considered, Eq. (9.12) should be re-written as follows:

$$V_D = V(D,S) - V'_{THFG} \tag{9.13}$$

where the term V(D,S) depends on α_S and α_D. It is quite complex to estimate such contributions, since the floating gate terminal is not physically accessible. Therefore, a piece of information related to the threshold voltage seen by the floating gate is available at the drain terminal (bitline) of the cell. The threshold voltage seen by the control gate V_{THCG} has the following relationship with V'_{THFG}:

$$V_{THCG} = \frac{V'_{THFG}}{\alpha_G} \tag{9.14}$$

Let's assume that we are able to measure bitline voltages: after every single program-after-erase pulse it is possible to verify that the threshold voltage seen by the floating gate does not rise above a given V_{THFGE} value, which can be used (thanks to Eq. (9.14)) to find the corresponding maximum limit for the threshold voltage seen by the control gate, V_{THCGE}.

As shown in Fig. 9.11, a value for the negative threshold V_{WLEQ} on the V_{THR} axis (corresponding to V_{THCGE}) can be found.

Petitioner Micron Ex-1021, 0249

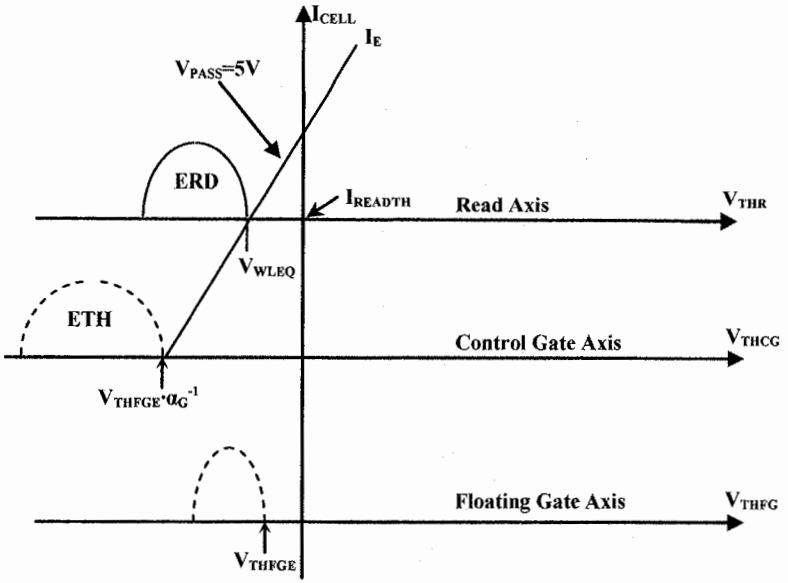


Fig. 9.11. Effect of absolute negative sensing on the different axis

Measurement of the bitline voltage is shown in Fig. 9.12. The circuit inside the dotted box is a page buffer with a well-defined trigger threshold V_{THS}. Assuming valid Eq. (9.12), when V_{BL} goes below V_{THS} (dotted lines in Fig. 9.12), that is

$$|V_{THFG}| \leq V_{THS} = |V_{THFGE}| \tag{9.15}$$

the sensing circuit switches. V_{THS} is equal to $|V_{THFGE}|$ after the transient is finished.

This sensing technique cannot be used in conjunction with All Bit Line architecture. In fact, the information contained in the bitline voltage would be perturbed by the coupling with other bitlines (Sect. 9.2.4).

An even/odd interleaving architecture with fixed potential shielding is the most suitable solution. Differently from sensing operations, the shielding to ground would cause a short circuit between source line SL and ground, through the erased strings. To overcome this issue, the only voltage that can be used for shielding is the one used for SL. Figure 9.13 shows the scheme implementing the shielding. The signals of the bitline selectors switch the same way as in the case of read (Fig. 8.16). BIAS node is short to SL through the switch driven by the EV signal.

Another issue related to this technique is that trying to measure high bitline voltages V_{BL} can cause problems to the proper operation of the circuits. For instance, let's consider the page buffer; after the evaluation phase, two values are possible on SO node:

- If $V_{BL} > |V_{THFGE}|$ then node SO is equal to V_{DD} and the latch switches.
- If $V_{BL} < |V_{THFGE}|$ then node SO is connected to V_{BL} and, in the worst case, it is exactly equal to $|V_{THFGE}|$. The latch does not switch.

Petitioner Micron Ex-1021, 0250

A sufficiently high value of $|V_{THFGE}|$ could anyway let the page buffer latch switch (see the sizing of the latch in Chap. 8).

Therefore, there is a maximum limit to the value of $|V_{THFGE}|$, i.e. it should be smaller than the trigger threshold voltage V_{THL} of the latch.

In order to overcome this issue, two valid techniques can be used.

1. One solution is to implement a sensing phase before the transient of the bitline charge has finished. This transient can last some tens of μs. In this way, it is possible to discriminate bitline voltages which, after the transient, would require a higher $|V_{THFGE}|$. As shown in Fig. 9.14 (dotted lines), shifting the sensing operation from T_1 to T_2 is equivalent to having a higher $|V_{THFGE}|$.
2. Another method is to exploit the coupling capacitor between bitline and source line. Once the transient has finished, source line is discharged (time T_1 in Fig. 9.15) either to ground or to an intermediate value. Thanks to the coupling capacitor, bitline voltages are shifted downwards (Fig. 9.15).

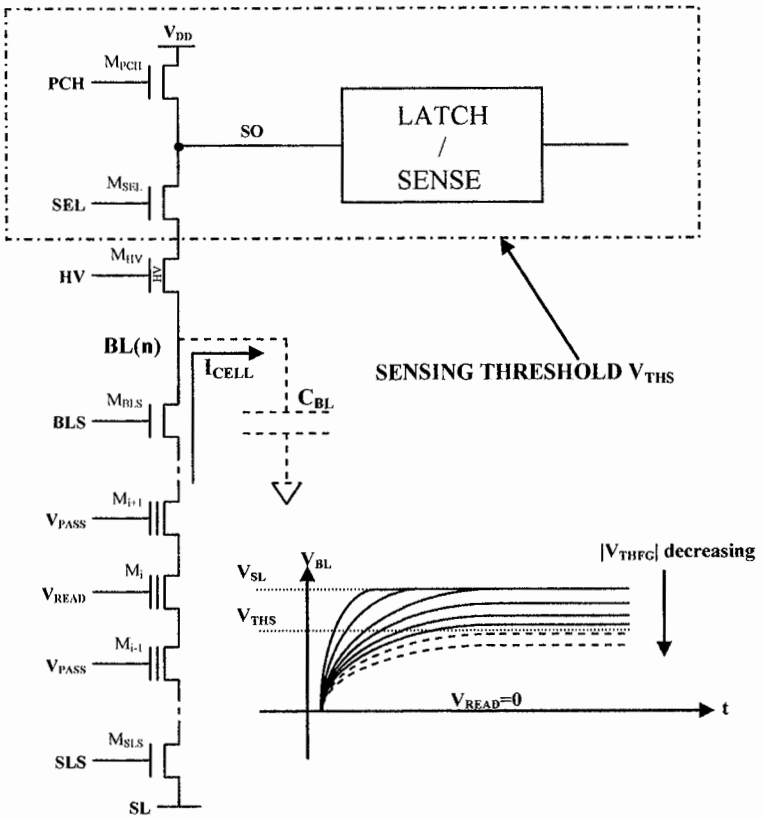


Fig. 9.12. Absolute negative threshold sensing

Petitioner Micron Ex-1021, 0251

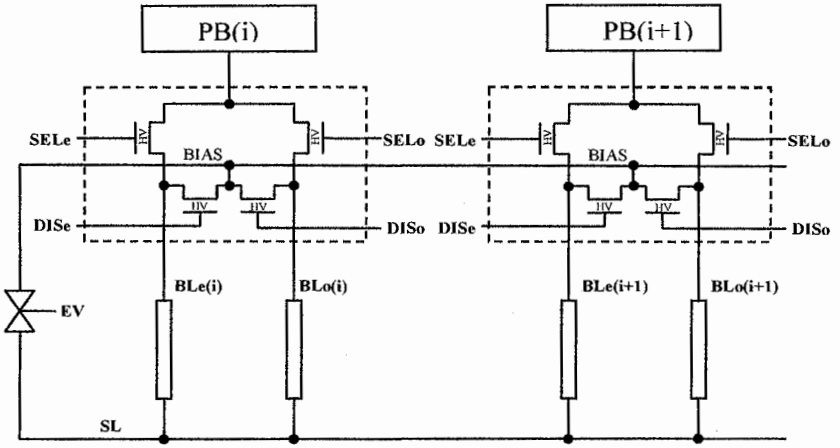


Fig. 9.13. Interleaving bitline architecture for negative sensing

Experimental results show that, once V_{THFGE} is set to −1 V, the corresponding value of V_{WLEQ} is −3 V.

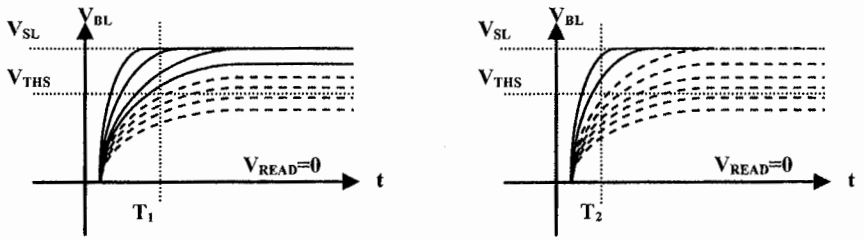


Fig. 9.14. Absolute negative threshold sensing with variable sensing time

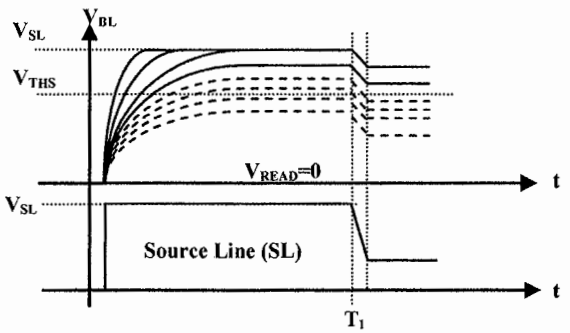


Fig. 9.15. Absolute negative threshold sensing after SL–BL boosting

Petitioner Micron Ex-1021, 0252

9.2.3 Current sensing

The current flowing in a MOS can be controlled not only by means of the gate voltage V_G, but also by means of the source voltage V_S according to the following equation:

$$I_D = K \cdot (V_G - V_S - V_{TH})^2 \tag{9.16}$$

This is true even in the case of a MOS whose threshold voltage V_{TH} is negative. If the following condition holds true

$$V_S = (V_G - V_{TH}) \tag{9.17}$$

then the current is zero.

The value of the current derived from Eq. (9.16), when V_S is greater than V_G, is the same as the one derived when $V_S = 0$ V and the (negative) gate voltage is equal to $V_G - V_S$. The body effect can be cancelled biasing the bulk node to a voltage V_S. In order to allow the current (defined in Eq. (9.16)) to flow inside the MOS, the drain potential V_D must be greater than source potential V_S.

$$V_D > V_S \tag{9.18}$$

Figure 9.16 summarizes the biasing conditions described so far for a current sensing where equivalent V_G is negative.

Since drain voltage can be set to a constant value, this approach can be applied to ABL architecture [2].

From now on, let's consider $V_G = 0$ V; the condition $V_S > V_G$ is equivalent to have a sensing operation with a negative voltage on the gate. Figure 9.17 shows the current sensing of an erased cell and the timings of the main signals involved.

A voltage $V_{READ} = 0$ V is applied to the gate of the selected cell, while the remaining cells are at V_{PASS}, which is high enough to make them act as pass transistors. Both M_{BLS} and M_{SLS} act as pass transistors as usual.

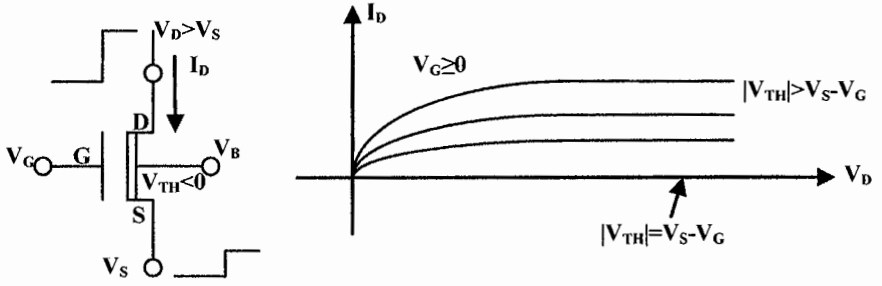


Fig. 9.16. Negative threshold NMOS biasing for current sensing

Petitioner Micron Ex-1021, 0253

At time T_0, source line SL is biased with a positive voltage. In case the V_{TH} of the selected cell is higher (as absolute value) than the voltage of the SL, the bitline gets charged at the same potential as the SL. At time T_1, transistors M_{HV} and M_{SEL} get biased according to the usual sensing conditions. At time T_2, M_{PCH} is switched on and the bitline gets to the value $V_{PRE} - V_{TH}$ and V_{SO} rise to V_{DD}. At time T_3, the bitline charge transient has finished and a constant current flows through the bitline, whose value depends on the threshold voltage of the selected cell. At this point, the usual evaluation phase can begin, that is the integration of the current on the SO capacitor.

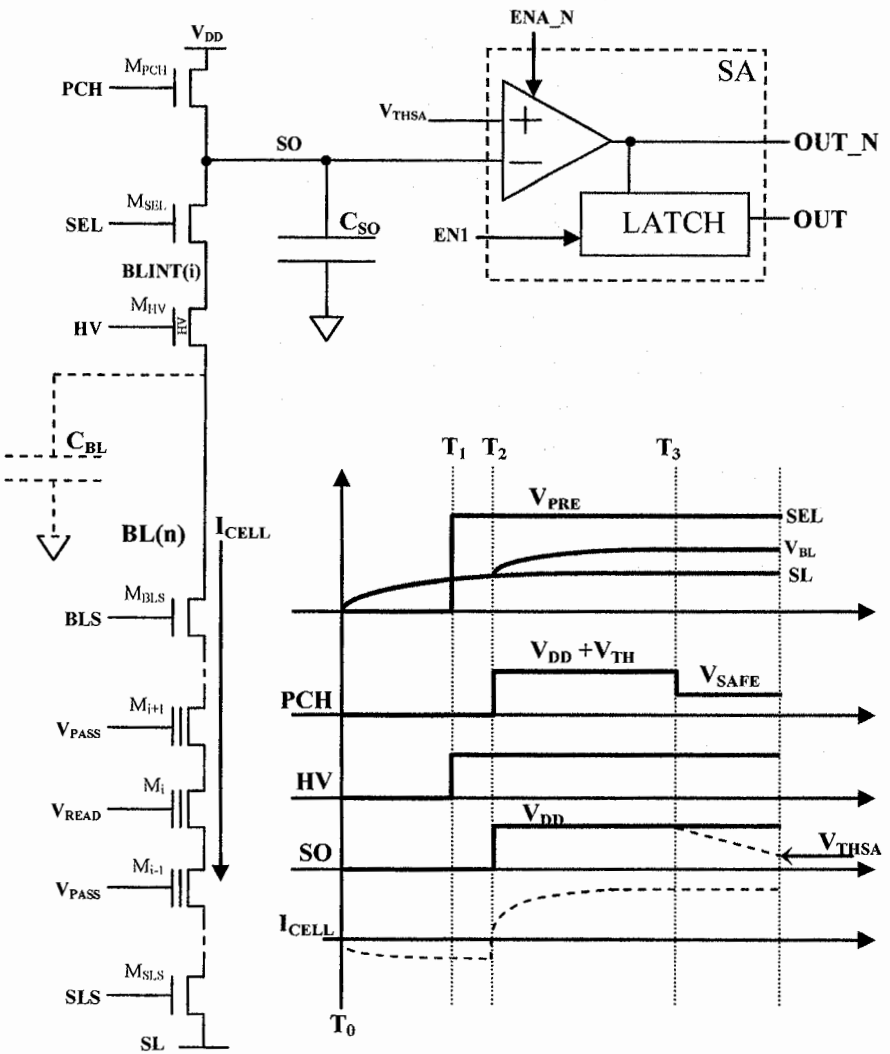


Fig. 9.17. ABL sensing for negative threshold voltages

Petitioner Micron Ex-1021, 0254

If the evaluation phase lasts exactly the same time as the read phase, this is equivalent to a verify executed with a wordline voltage equal to $-V_S$. Indeed, it is necessary to take into account the effect of modulation of the bitline voltage and therefore of the V_{DS} of the cell: in order to have a situation exactly equivalent to the read case, a voltage equal to $V_{PRE} + V_S$ should be applied to the node SEL. Such a potential would be so high that it would compromise the correct operation of the circuit. It is necessary to bias the bitline with a lower voltage and cope with the error caused by using a lower V_{DS}. The amount of error introduced is comparable to the one caused by the typical BPD.

A possible solution is to properly modulate the evaluation time, similarly to the approach described in Sect. 9.2.1, in order to set a current threshold more suitable to compensate this effect.

In order to completely solve the issue, a method could be to always keep the same source and drain potentials for every read and verify operation, varying V_{READ} only.

9.2.4 Source line read for ABL sensing

The most relevant issue related to the latter read technique is certainly the regulation of node SL to a fixed voltage: sensing operation, as shown in Sect. 9.3, is strongly dependent on V_S variation.

When hundreds of thousand strings are read in parallel, a considerable current injection (tens of milliAmpere) occurs on the source node. As a consequence, the only way to keep a constant value of V_S with a proper precision is to add a voltage regulator, whose size is not negligible. Furthermore, it is necessary to implement a low-resistivity distribution network for V_S. The resulting increment of the chip area could simply be unfeasible.

A way to avoid voltage regulators for the source line is shown in Fig. 9.18. From a circuit point of view, a M_{DISCH} transistor has been added, so that it generates a discharge current I_{DISCH} in the order of a hundred of nA. Voltage V_{DISCH} could be generated, for instance, using a current mirror common to all M_{DISCH} transistors [3].

During the sensing phase, at time T_0, transistor M_{HV} is always open and it connects the discharge transistor to the bitline. Source line SL is biased to V_{DD} and therefore it does not need a voltage regulator. Transistor M_{SEL} is off and the bitline is charged at a voltage related to the threshold voltage of the selected cell. A cell with a negative threshold voltage is conductive even if $V_{READ} = 0$ V.

The discharge path acts as a pull-down for those bitlines which experienced a pull-up of their voltages because of the coupling with adjacent bitlines: this stored extra charge is eliminated by the discharge current I_{DISCH}.

At time T_2, all the bitlines have reached their correct voltage, without coupling contributions.

Petitioner Micron Ex-1021, 0255

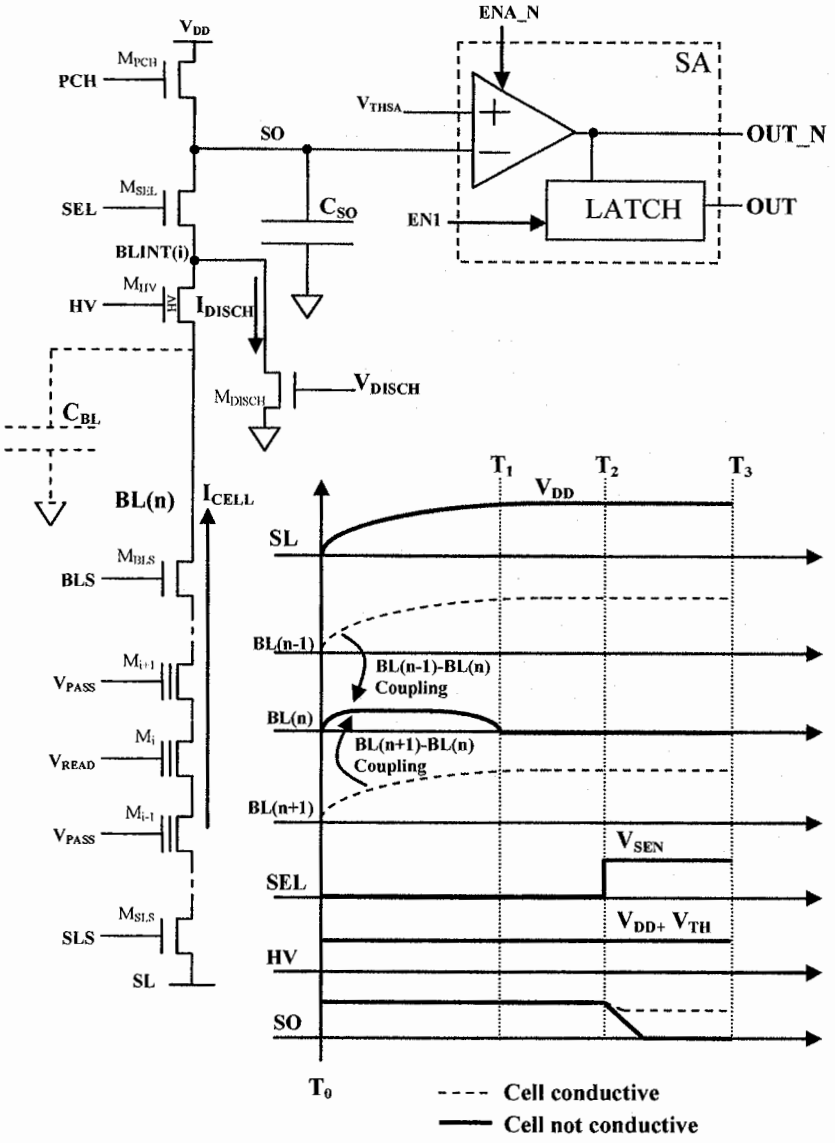


Fig. 9.18. ABL sensing for negative threshold: alternative solution

Figure 9.18 shows the waveforms for three adjacent bitlines BL(n + 1), BL(n) and BL(n – 1): a conductive cell is associated to both bitlines BL(n + 1) and BL(n – 1), while a non-conductive cell is associated to bitline BL(n). Bitline voltages of BL(n + 1) and BL(n – 1) increase as V_{SL} approaches V_{DD}. Because of the coupling

Petitioner Micron Ex-1021, 0256

capacitors towards BL(n), an undesired increase of BL(n) voltage occurs. Thanks to the discharge current I_{DISCH}, the unwanted charge on the bitline BL(n) is removed.

At time T_3, transistor M_{SEL} is enabled and a charge sharing between the bitline and the node SO is established. The usual voltage sensing is then performed through the comparator. The behavior of bitlines BL(n + 1) and BL(n − 1) is represented by the dotted line, while the behavior of BL(n) is represented by the solid line in Fig. 9.18.

This technique ($V_{SL} > 0$ V) can be, therefore, applied with the same timings and voltages, during the negative sensing and during the read and verify operations: the proper V_{READ} voltage is applied to the selected wordline, while the unselected cells are biased at V_{PASS}. The overall duration of the bitline transient completion is either theoretically calculated or derived using proper experiments and/or testing. The drawback of this technique is indeed the generation of small discharge currents: the transistors operating with these currents are biased with very low V_{GS}. Therefore, they are particularly sensitive to threshold voltage variations and to the several spikes of the common ground, which has to absorb the current coming from hundreds of thousand M_{DISCH} transistors.

9.3 Source line bias error

Source line SL is shared among all the strings of the matrix. During read and verify operations, it is always considered an ideal ground. In reality, because of its non-zero resistance and because of the currents injected into it, its value is highly variable.

The source line is built using a low-resistivity material (usually, the highest metal layer) which covers the whole matrix and terminates on the ground PAD, as shown in Fig. 9.19. A NAND Flash with two memory planes is sketched. This network is connected, through a small (high-resistivity) diffusion, to a set of *n* strings; *n* is in the order of a hundred of strings.

The source line, together with its resistive contributions, is very complex to evaluate, and Fig. 9.19 shows a simplified model:

- Inside the dotted area, both the resistive contributions of the line in diffusion ΔR_i and the voltage drops ΔV_i (caused by the cell currents I_{CELLi}) are included.
- Point D represents the end of the diffusion line and the beginning of the low-resistivity metal line, modeled with R_M, onto which I_{CELLM} current (from all the remaining cells of the matrix on the left) is injected. I_{CELLM} generates the drop ΔV_M.
- Point A collects the currents from both the matrixes, and R_G is the resistance value associated to the metal section connecting point A to the ground pad. The resulting voltage drop is represented by ΔV_G.

Petitioner Micron Ex-1021, 0257

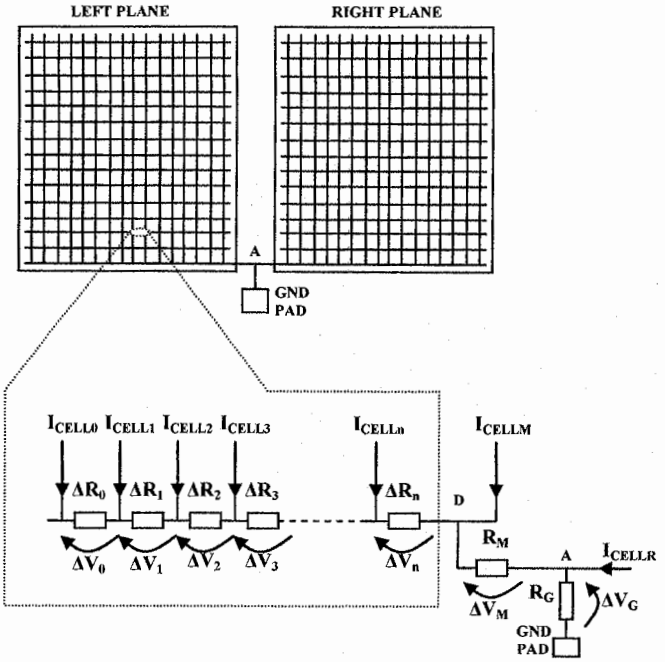


Fig. 9.19. SL routing and parasitic resistance contributes

The effect on the cell is shown in Fig. 9.20. The increase of the source line voltage ΔV_{SL} reduces the real voltage V_{GS} applied to the cell. If the read cell threshold voltage is considered, there is an apparent increase of V_{THR}. Equivalent resistance of the source line is around 10–20 Ω and the resulting currents are in the order of 10 mA. Therefore the voltage drop is usually about 200 mV.

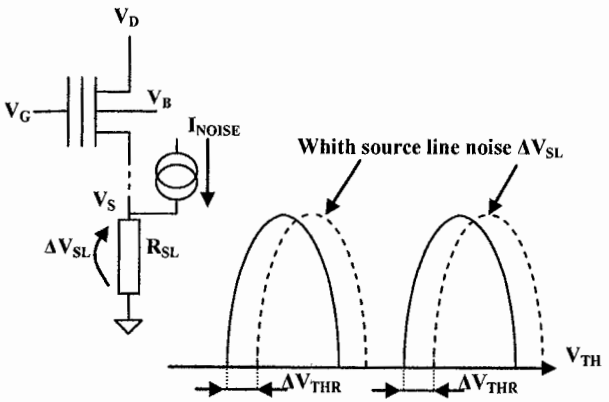


Fig. 9.20. SL noise and voltage distribution shift

Petitioner Micron Ex-1021, 0258

The shift of V_{THR} apparently has a 1:1 ratio with the voltage increase of the source line, that is:

$$\Delta V_{THR} = \Delta V_{SL} \tag{9.19}$$

In reality the shift is bigger because the rise of V_S not only impacts the V_{GS} of the cell, but also the V_{DS} of the string. Assuming a bitline biasing of 0.5 V, a 0.2 V increase of the source line would result in a voltage of 0.3 V applied at the ends of the source line, thus reducing it of almost 50% and causing a reduction of the cell current, which in turn causes an increase of V_{THR}.

A further contribution to the V_{THR} increase is due to the body effect on the cell, even if it could theoretically be eliminated by biasing the body of the cells to the source line voltage. It must be taken into account that the source line is a lumped-parameter network (therefore showing different voltages, Fig. 9.19), while a single voltage (unique and common to all the cells) can be chosen for the body.

Experimental results [4] have shown that the reduction of V_{GS}, of V_{DS} and the body effect, caused by the raise of the source line of some hundreds of millivolt, leads to a new format of Eq. (9.19):

$$\Delta V_{SHIFT} = K \cdot \Delta V_{SL} \tag{9.20}$$

where the value of K is between 2 and 3.

Therefore, a source line increase of 200 mV impacts the distribution causing a V_{THR} shift of 400–600 mV.

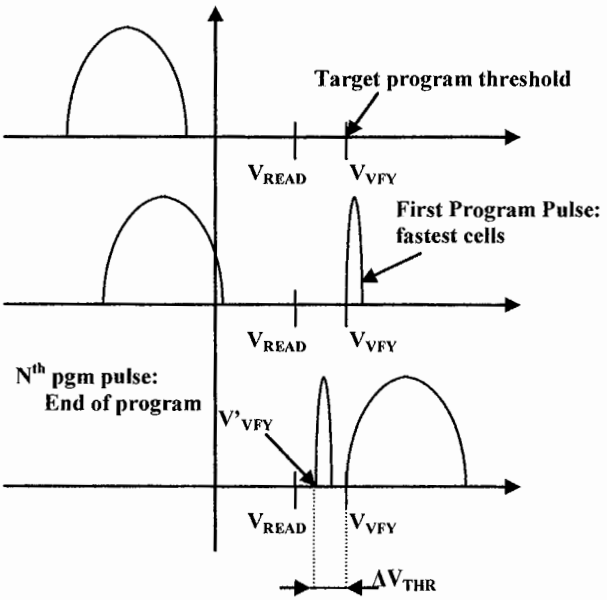


Fig. 9.21. SL noise: distribution enlargement and read margin reduction

Petitioner Micron Ex-1021, 0259

The biasing error of the source line would cause no issue if it were a constant parameter over time. Unfortunately the currents injected onto it are pattern dependent, that is they depend on the data stored inside the cells, and the number of possible combinations is really huge. On top of that, the value of the currents of the erased cells depends on several external parameters, like, for instance, the temperature, which influences the saturation value.

Frequently, one or more cells, which have been programmed and verified in an environment full of erased cells (maximum ΔV_{SL}), could be read later on in an environment full of programmed cells ($\Delta V_{SL} = 0$ V). It could also happen during the program operation: for physical reasons, some cells reach the programmed distribution faster than others. Such fast cells are verified with $\Delta V_{SL} > 0$, but when the slow cells have reached the programmed distribution, the fast ones experience an apparent shift of their V_{THR} towards left, as shown in Fig. 9.21.

The target value for the V_{THR} of the programmed cells is equal to the verify voltage V_{VFY}. At the first program step the fast cells are verified with a biasing error on the source line because of the cells which are still in the erased distribution. At the end of programming, fast cells have been verified with an apparent verify voltage equal to

$$V'_{VFY} = V_{VFY} - \Delta V_{THR} \tag{9.21}$$

and the read margin is no longer

$$\Delta V_{RM} = V_{VFY} - V_{READ} \tag{9.22}$$

but

$$\Delta V_{RM} = V_{VFY} - V_{READ} - \Delta V_{THR} \tag{9.23}$$

and, therefore, it is considerably reduced.

Another interesting case can be found with some types of multilevel memories (Chap. 10) as shown in Fig. 9.22. After a first program phase, some cells have moved from the erased distribution E to the written distribution D1. These cells are verified with source line noise. In a second program phase, all the erased cells move to distributions D2 and D3. D1 cells are shifted because of the source line noise which has now disappeared: consequences are similar to the ones above mentioned. Obviously, these shifts have a more dramatic impact in multilevel devices, because of the reduced read margins.

As said, the source line error and the apparent shift associated to the V_{THR} of the cells is absolutely unpredictable and it varies over time: it must be then considered as a real widening of the distribution, similarly to the other parasitic effects already discussed (floating gate coupling, BPD, etc.).

Different techniques have been devised and implemented in multi bit per cell devices in order to minimize this effect.

Petitioner Micron Ex-1021, 0260

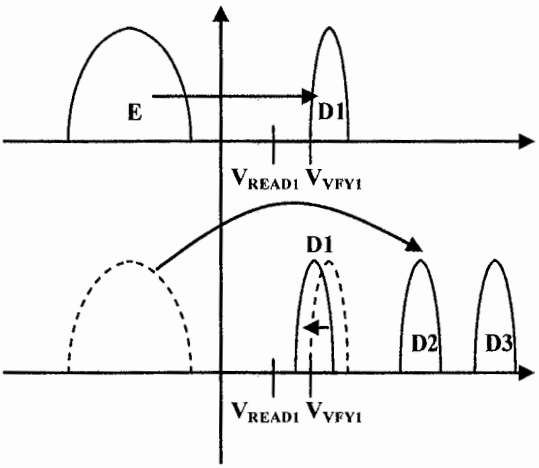


Fig. 9.22. SL noise: distribution enlargement and read margin reduction for MLC

9.3.1 Sensing with source line bias compensation

Figure 9.23a shows the usual architecture with source line drop: voltages applied to the ends of the string (V_D, V_G and V_B) are measured against a common ground. A possible countermeasure is to generate the NAND string voltages as a function of the source line voltage V_{SL}, as shown in Fig. 9.23b [4].

In the simplest implementation, the three voltages shift of a value equal to the voltage of the source line [5, 6].

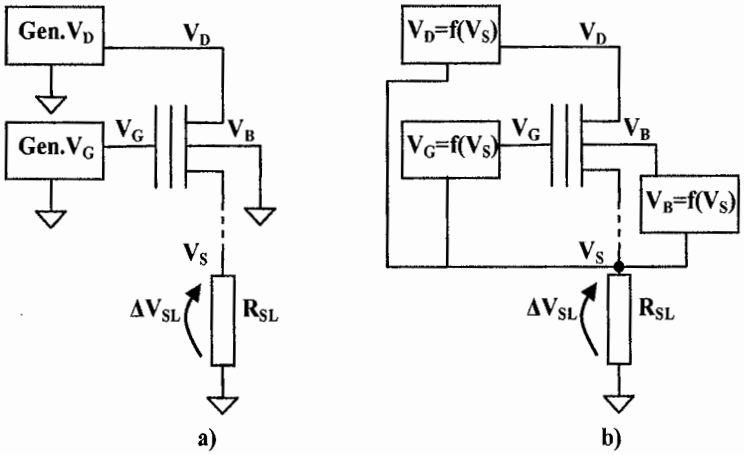


Fig. 9.23. SL noise compensation

Petitioner Micron Ex-1021, 0261

Figure 9.24 shows an implementation of the V_G voltage generator for the V_{SL} tracking. When signal TOSL is at V_{DD} (TOGND is set to ground), GNDLOCAL is driven at the same voltage as V_{SL} and V_G is equal to:

$$V_G = V_{REF}\left(1+\frac{R_2}{R_1}\right)+V_{SL} \tag{9.24}$$

where V_{REF} is generated by the BGREF circuit (band-gap reference circuit).

Therefore, cell's V_{GS} is equal to:

$$V_{GS} = V_G - V_{SL} = V_{REF}\left(1+\frac{R_2}{R_1}\right) \tag{9.25}$$

and it does not depend on V_{SL}.

On the other hand, when the signal TOGND is at V_{DD} (TOSL is at ground), the reference voltage of the circuit is GND again. In operations other than read, V_{SL} is different from GND (for instance, V_{DD} during program) and, therefore, it cannot be used as a reference.

The same technique can be used for V_D and V_B regulators: it is sufficient to replace their ground terminal with V_{SL}. Unfortunately, this method cannot compensate the drop of the source line completely, because V_{SL} is not a constant value over the whole SL network: only one voltage can be taken as a reference. For instance, voltages of nodes A and D of Fig. 9.19 can be used: V_D is the preferred choice as it compensates the voltage drops on R_M and R_G.

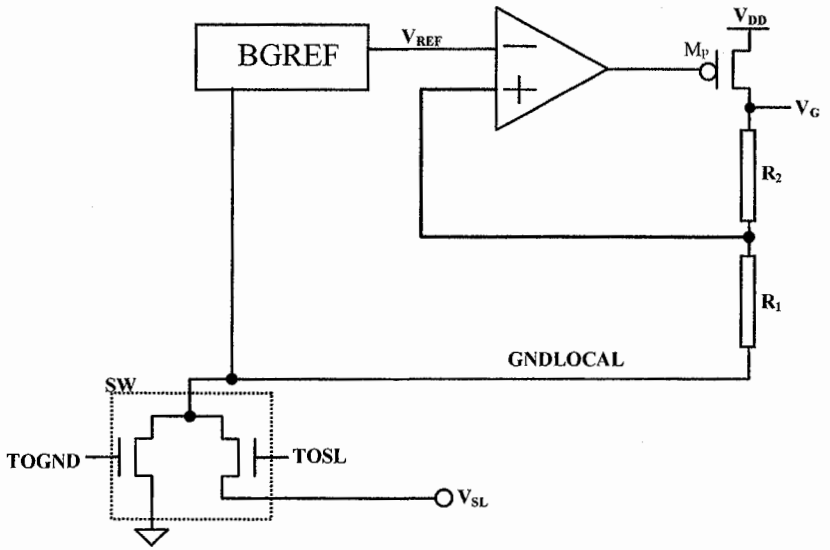


Fig. 9.24. SL noise compensation: wordline voltage generator

Petitioner Micron Ex-1021, 0262

9.3.2 Multi-pass sensing

Multi-pass sensing is a more efficient method to reduce the source line noise [7]. The basic concept is to repeat SRO using current thresholds $I_{READTHn}$ which are decreased at each iteration until the convergence to the value defined by the equation:

$$I_{READTH} = \frac{\Delta V \cdot C_{BL}}{T_{EVAL}} \tag{9.26}$$

At each pass, the cells which draw a current greater than the threshold current $I_{READTHn}$ are identified and excluded from subsequent read operations. In this way, at each pass, the read is less and less influenced by the noise due to the injection of high currents in the source line. The value of the threshold current of each single pass can be defined, for instance, by modulating the value of T_{EVAL} in Eq. (9.26).

Let's assume that the final threshold current (defined by Eq. (9.26)) is chosen to be 50 nA: the first pass could have a threshold current of 500 nA, the second one of 100 nA. The last pass is the actual read, performed with a threshold of 50 nA: the difference with the single pass read technique is that all the cells which sink more than 100 nA are shut down.

Of course, several passes could be used, but it is clear that after a couple of passes the source line current has already been greatly reduced, and the benefit introduced by adding more passes is more and more negligible. Figure 9.25 shows that the initial distribution, affected by source line noise, converges to the ideal distribution without source line noise: in the example, the first pass excluded the cells belonging to the portion of the erased distribution tagged with E1, while the second pass excluded those belonging to the portion E2.

Differently from previously discussed feedback technique of the source line node, this method can theoretically minimize the effect of the source line noise as much as desired, increasing the number of passes indefinitely. The drawback is the read time increase.

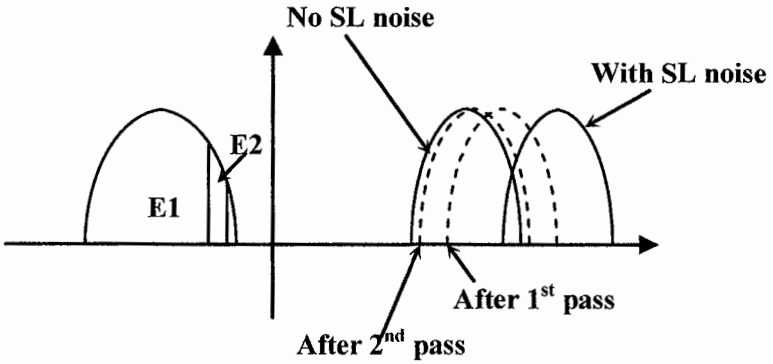


Fig. 9.25. Two pass sensing effect

Petitioner Micron Ex-1021, 0263

A way to shut down the identified cells is to force their associated bitlines to ground, so that the V_{DS} of the string is zero. This method can be implemented in the ABL sensing scheme, as shown in Fig. 9.26. Let's briefly see what happens, recalling what has been widely explained in Chap. 8.

First of all, the latch is set through M_{SET} which forces OUT_N = 0 and OUT = 1. M_{FB} acts as a pass and M_{PD} is off. The first pass is therefore a usual read operation as seen in the previous chapter, but having $I_{READTH1} > I_{READTH}$.

At the end of the first pass:

- If $I_{CELL} < I_{READTH1}$ then the state of the latch does not change (OUT = 1) and nothing changes in the next read operation.
- If $I_{CELL} > I_{READTH1}$ then the state of the latch changes (OUT=0). In the following read operation, M_{FB} is off and it excludes the bitline from the charge path, while M_{PD} is on and forces the bitline to ground.

Obviously, the latch cannot be set in the following sensing passes, otherwise the information stored in the latch would be lost.

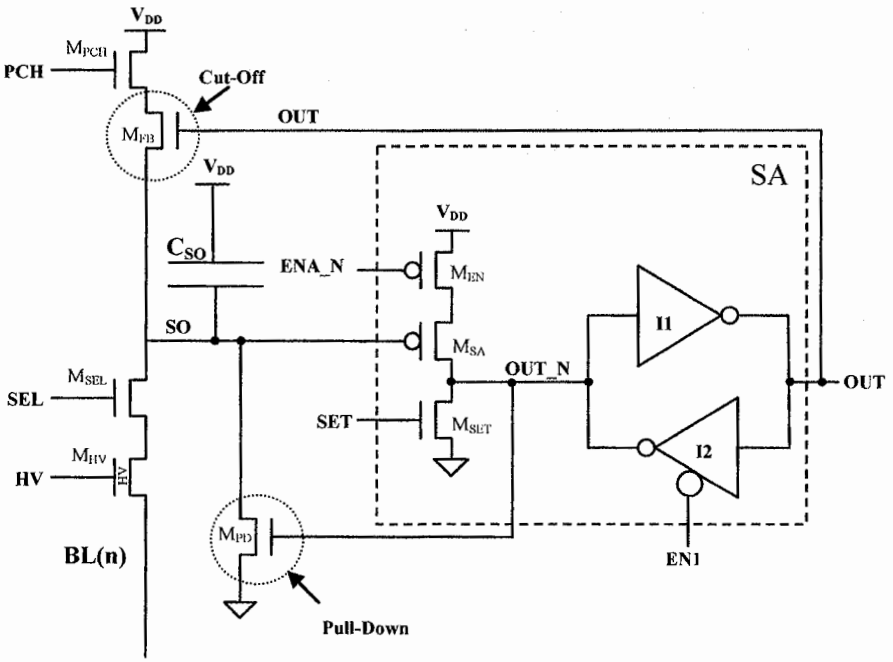


Fig. 9.26. ABL sense amplifier modified to exclude high current cells

Petitioner Micron Ex-1021, 0264

References

1. Wang, S.T *On the I-V characteristics of floating-gate MOS transistors.* Electron Devices, IEEE Transactions Sept. 1979, Volume: 26, Issue: 9, Page(s): 1292–1294.
2. Hao Thai Nguyen et al. U.S. Patent No. 7447079 – *Method for sensing negative threshold voltages in non-volatile storage using current sensing* Assignee: SanDisk Corporation (Milpitas, CA).
3. Seungpil Lee et al. U.S. Patent No. 7545678 – *Non-volatile storage with source bias all bit line sensing* Assignee: SanDisk Corporation (Milpitas, CA).
4. Deepak Chandra Sekar et al. U.S. Patent No. 7606072 – *Non-volatile storage with compensation for source voltage drop* Assignee: SanDisk Corporation (Milpitas, CA)
5. Raul-Adrian Cernea et al. U.S. Patent No. 7499324 *Non-volatile memory and method with control gate compensation for source line bias errors* Assignee: SanDisk Corporation (Milpitas, CA).
6. Yan Li et al. *A 16Gb 3b/Cell NAND Flash Memory in 56nm with 8MB/s Write Rate* Solid-State Circuits Conference, 2008. Digest of Technical Papers. ISSCC. 2008 IEEE International Feb. 2008, Page(s): 506–507, 632.
7. Raul-Adrian Cernea et al. U.S. Patent No. 7196931 - *Non-volatile memory and method with reduced source line bias errors* Assignee: SanDisk Corporation (Sunnyvale, CA).

Petitioner Micron Ex-1021, 0265

10 MLC storage

L. Crippa[1] *and R. Micheloni*[2]

The obvious advantage of a 2 bit/cell implementation (MLC) with respect to a 1 bit/cell device (SLC) is that the area occupation of the matrix is half as much; on the other hand, the area of the periphery circuits, both analog and digital, increases. This is mainly due to the fact that the multilevel approach requires higher voltages for program (and therefore bigger charge pumps), higher precision and better performance in the generation of both the analog signals and the timings, and an increase in the complexity of the algorithms.

10.1 MLC coding and programming algorithms

Figure 10.1 shows an example of how 2 bits are associated to the four read threshold distributions stored in the cell, and how the set of programmed distributions is built starting from the erased state "E". In this case the multilevel is achieved in two distinct rounds, one for each bit to be stored [1, 2].

In the first round, the so-called lower-page (associated to the *Least Significant Bit* – LSB) is programmed. If the bit is "1", the read threshold of the cell V_{THR} (see Definition 8.2) does not change and, therefore, the cell remains in the erased state, E. If the bit is "0", V_{THR} is increased until it reaches the D1 state.

V_{THR} is modified by means of the *Incremental Step Pulse Programming* (ISPP) algorithm: a voltage step (whose amplitude and duration are predefined) is applied to the gate of the cell. Afterwards, a verify operation is performed, in order to check whether V_{THR} has exceeded a predefined voltage value (in this case V_{VFY1}). If the verify operation is successful, the cell has reached the desired state and it is excluded from the following program pulses. Otherwise another cycle of ISPP is applied to the cell, where the program voltage is incremented by ΔISPP.

In the second round, the upper-page (associated to the *Most Significant Bit* – MSB) is programmed. If the bit is "1", V_{THR} does not change and, therefore, the cell remains either in the erased state, E, or in the D1 state, depending on the value of the lower-page. When MSB is "0", V_{THR} is programmed as follows:

[1] Forward Insights, luca.crippa@ieee.org
[2] Integrated Device Technology, rino.micheloni@ieee.org

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_10,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0266

- If, during the first round, the cell remained in E state, then V_{THR} is incremented to D3.
- If, during the first round, the cell was programmed to D1, then, in the second round, V_{THR} reaches D2.

Even in this case, the program operation uses ISPP, and the verify voltages are V_{VFY2} and V_{VFY3}. Lower-page programming only needs the information related to LSB, while for the upper-page it is necessary to know both the starting distribution (LSB) and the MSB.

Because of technological variations, V_{THR} is not perfectly related to the amplitude of the program pulse (during ISPP): there are "fast" cells which reach the desired distribution with few ISPP pulses, while other "slow" cells require more pulses.

The amplitude of the first program pulse ($V_{PGMLSB0}$) of the lower-page should not allow the threshold V_{THR} of the "fastest" cell to exceed V_{VFY1}. Should it happen, an undesired widening of distribution D2 occurs or, in the worst case scenario, V_{THR} might reach D2 distribution at once.

Typical $V_{PGMLSB0}$ is around 16 V. In case of program of "slow" cells from E to D1, the last programming step needs values as high as 19 V. Assuming ΔISPP equal to 250 mV, it takes 12 steps to move from 16 to 19 V.

Similarly, the starting pulse of the upper-page $V_{PGMMSB0}$ should have an amplitude such that the "fastest" cell does not go beyond V_{VFY2}.

$$V_{PGMMSB0} = V_{PGMLSB0} + (V_{VFY2} - V_{VFY1}) \tag{10.1}$$

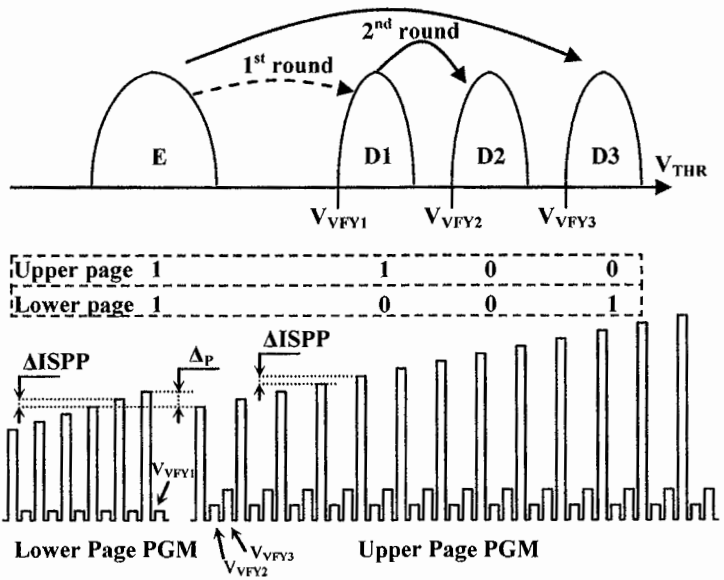


Fig.10.1. Two rounds MLC program operation

Petitioner Micron Ex-1021, 0267

The value of $V_{VFY2} - V_{VFY1}$ is typically around 1 V and, therefore, the initial voltage is about 17 V. As shown in Fig. 10.1, the upper-page ISPP does not start from the last voltage used for the lower-page programming, but it begins at a lower voltage. For example, instead of starting at 19 V, it could start at 17 V, eight steps below.

10.1.1 Full-sequence programming

Full-sequence programming [3, 4] is shown in Fig. 10.2; in this case, LSB and MSB of the same cell are programmed at the same time, and there is no need to apply the same program voltage twice.

A first drawback of the full-sequence programming is that, after each program pulse, three different verify operations are needed (at V_{VFY1}, V_{VFY2} and V_{VFY3}).

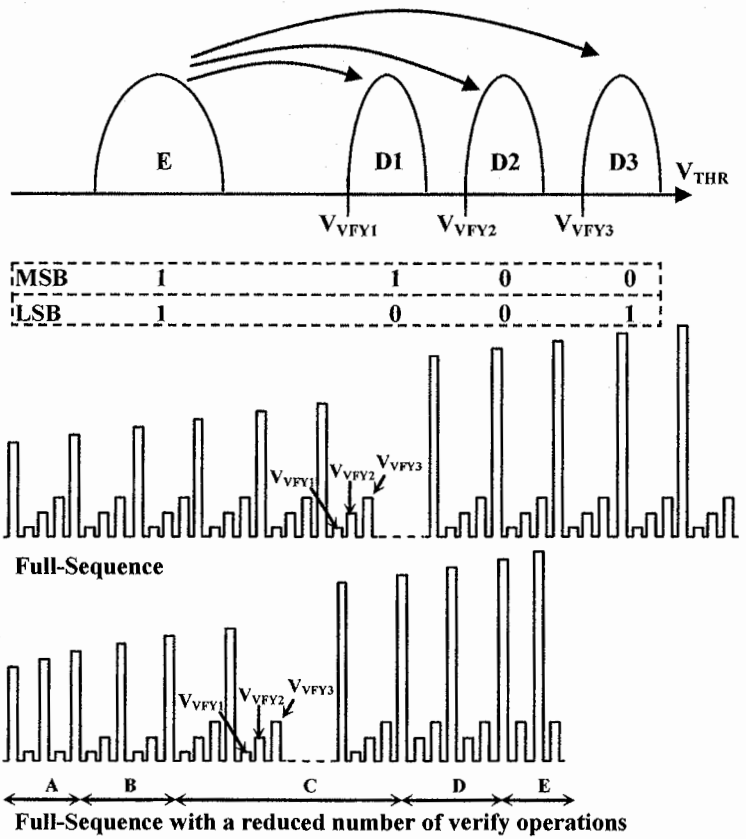


Fig. 10.2. Full-sequence program operation

Petitioner Micron Ex-1021, 0268

In order to reduce the overall number of verifications, the following algorithm can be used:

- At the beginning, only verify at V_{VFY1} is performed (A segment of Fig. 10.2).
- V_{VFY2} verify begins when at least one cell has reached V_{VFY1} (B segment of Fig. 10.2)
- V_{VFY3} verify begins when at least one cell has reached V_{VFY2} (C segment of Fig. 10.2).
- V_{VFY1} verify ends when all the cells have reached D1 (D segment of Fig. 10.2).
- V_{VFY2} verify ends when all the cells have reached D2 (E segment of Fig. 10.2).
- Only V_{VFY3} is used afterwards.

Let's now consider a NAND memory with ABL architecture. The programming of two adjacent logical pages is shown in Fig. 10.3. The first page is made up by the LSBs of all the memory cells belonging to the addressed wordline; instead, the second page is composed by the MSBs.

Another drawback of the full-sequence algorithm is evident when we consider the time needed for the *data-load* (DL) operation (Sect. 10.3); data-load is the upload of the pages to be programmed into the NAND Flash (i.e. page buffers).

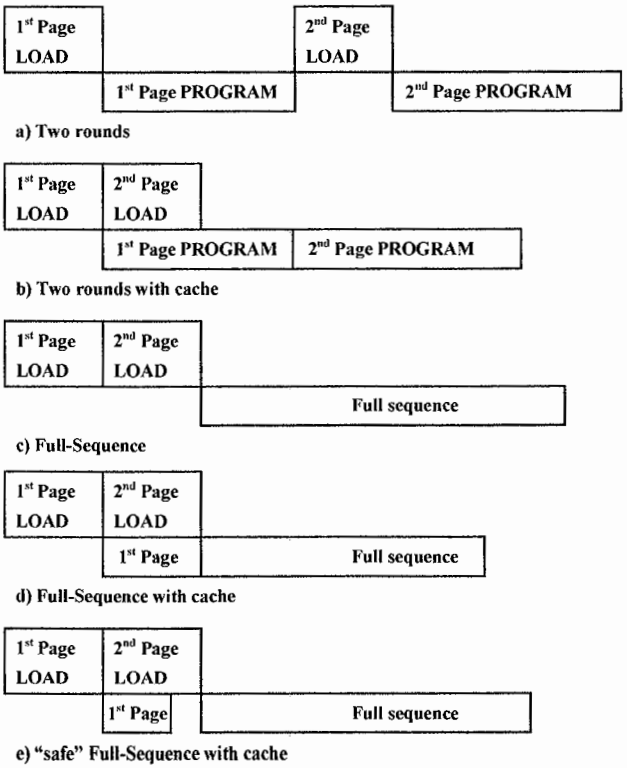


Fig. 10.3. MLC programming algorithms with ABL architecture

Petitioner Micron Ex-1021, 0269

When computing the overall program time, DL has to be definitely taken into account. Figure 10.3a shows the DL sequence for two distinct program operations. The programming time of the first page usually lasts more than the DL of the second page. A way to reduce the overall program time is to load the second page while the first page is being programmed (cache-program, as shown in Fig. 10.3b).

Full-sequence algorithm starts only if both MSB and LSB pages are available (Fig. 10.3c); this algorithm can be modified as in Fig. 10.3d. First page programming starts once the first page has been uploaded; in the meanwhile, the second page is uploaded as in the cache approach. After the full upload of the second page, first page program is interrupted and the real Full-Sequence starts. The risk associated with this technique is that the Full-Sequence might start with a too high ISPP voltage: some fast cells, whose target is D2, might directly jump to D3. To overcome this issue, program is interrupted when at least one cell has reached V_{VFY1}, and the full-sequence starts once the second page has been uploaded (Fig. 10.3e).

Programming algorithms described so far are the most commonly used, but others have been devised. It is not the aim of this chapter to describe all the possible implementations, since the reader should now be able to derive several hybrid solutions out of the ones shown in Fig. 10.3. Other interesting techniques are described in Chap. 16 (XLC storage).

10.1.2 Floating gate coupling reduction

Let's consider two cells, M_i and M_{i+1}, belonging to two adjacent wordlines within the same NAND string. When programming cell M_{i+1}, the *Floating Gate Coupling* (FGC) effect on the adjacent cell M_i depends on the final state of M_{i+1}. Figure 10.4 represents the worst case: cell M_{i+1} is programmed from state E to state D3, and cell M_i's threshold V_{THR} increases, as shown by the dotted distributions.

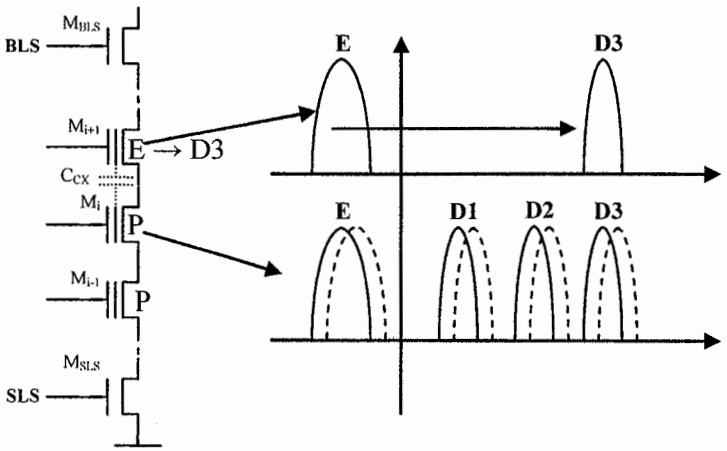


Fig. 10.4. Floating gate coupling between adjacent cells

Petitioner Micron Ex-1021, 0270

Of course, intermediate situations exist; cell M_{i+1} moves either from state E to D2 or from E to D1 and the amount of FGC gets smaller and smaller. In the case where M_{i+1} remains in state E, no disturb occurs. The amount of coupling depends depends on the target V_{THR} of the adjacent cell, which is pattern-dependent. This fact causes a widening of the distributions and a resulting reduction of read margins.

A program technique [5] which significantly reduces FGC is sketched in Fig. 10.5: also this algorithm calls for two program rounds, one for the lower-page and one for the upper-page. The first round is similar to the one already discussed. If the bit of the lower-page is "1", then V_{THR} does not change and the cell remains in the E state. If the bit is "0", then V_{THR} is incremented until it reaches D2', associated with verify level V'_{VFY2}.

State D2' is an intermediate one, which does not need the MLC accuracy and, therefore, it can be programmed using a wider ΔISPP1 step. State D2' is going to be re-programmed during the second programming round (upper-page), as shown in Fig. 10.5.

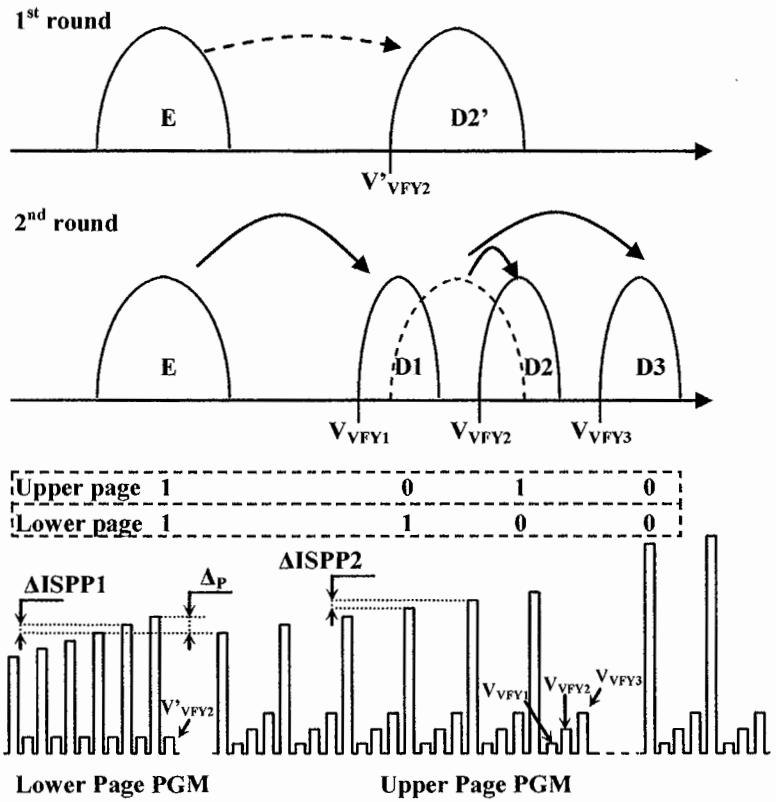


Fig. 10.5. Programming algorithm for floating gate reduction

Petitioner Micron Ex-1021, 0271

The MSB program is the following:

- If the cell is in the E state and the upper-page is "1", then the cell remains in E.
- If the cell is in E and the upper-page is "0", then the cell is programmed to D1, which corresponds to the verify voltage V_{VFY1}.
- If the cell is in the intermediate D2' state and the upper-page is "1", then the cell is programmed to state D2, which corresponds to verify voltage V_{VFY2}.
- If the cell is in D2' and the upper-page is "0", then the cell is programmed to state D3, which corresponds to verify voltage V_{VFY3}.

It should be noted that $V_{VFY2} > V'_{VFY2}$. In order to reduce the FGC effect, the above described technique has to be used in conjunction with a specific programming sequence. Usually, MLC NAND memories have a restriction on the addressing during the program operation. Within a block, the pages must be programmed consecutively, from the least significant page to the most significant page of the block. Random page address programming is prohibited.

Let's consider the ABL architecture (without full-sequence) sketched in Fig. 10.6a. Logical page 1 is programmed into the lower-page of cell M_0. In Fig. 10.6 logical pages are shown with a circled number. Unlike the previous case, logical page 2 is not programmed into the upper-page of cell M_0, but rather in the lower-page of adjacent cell M_1. If the state of M_1 goes from E to B2', the threshold voltage of cell M_0 shifts because of FGC: if the V_{THR} of M_0 belonged to B2' distribution, the shift is indeed recovered by the program of the upper-page, which corresponds to the third page. The fourth page is programmed in the lower page of cell M_2. Cell M_1 sees its threshold shifted because of the floating gate coupling with M_0 and M_2, but again the threshold shift of M_1 is recovered when the fifth page is programmed. Such last program is the only one providing the overall contribution to floating gate coupling towards cell M_0.

Therefore, the floating gate coupling is caused by the program of the upper-page only: unlike the previous case where the program of the upper-page in worst case causes a jump from E state to D3 state, this technique has a worst case where the jump is reduced to either E state to D1 or D2' state to D3.

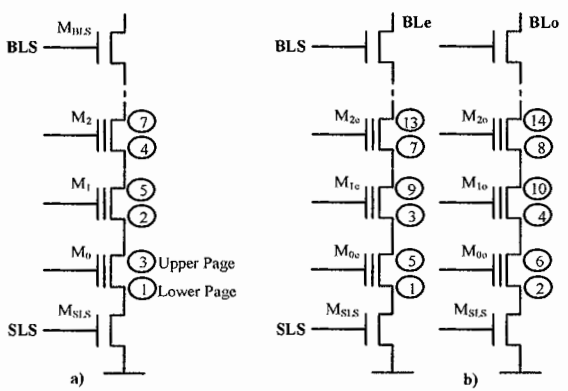


Fig. 10.6. Cell programming cross-sequence for a) ABL and b) interleaving

Petitioner Micron Ex-1021, 0272

This technique has been described for an ABL architecture, but it can be also applied to an interleaving architecture, by means of the sequence shown in Fig. 10.6b: the advantage is that floating gate coupling is reduced in BL-BL direction as well.

Another interesting type of distribution coding is shown in Fig. 10.7 [6]. Also in this case, programming of the upper-page has a worst case where the threshold voltage jump is either E-D2 or D1-D3. When applied to the program sequence shown in Fig. 10.6, floating gate coupling is statistically reduced even if no reprogram is performed.

It is worth to mention that a variable coding scheme has been proposed [12]. Depending on the data pattern to be stored, the NAND device selects one out of four available gray codes in order to minimize the number of cells to be actually programmed. A reduced number of cells to be programmed means a reduced bitline precharge current and a tighter cell's V_{TH} distribution width.

Chapter 16 deals with reprogramming techniques for XLC storage.

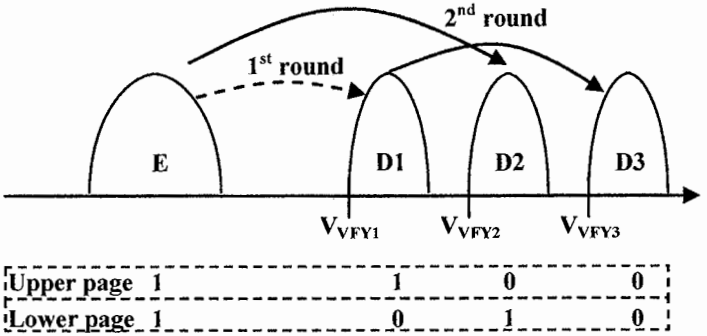


Fig. 10.7. A third option for MLC coding

10.2 MLC sensing circuit

Aim of the sensing circuit is to determine which distribution the threshold voltage V_{THR} of the cell belongs to. Several circuit solutions have been devised and described in literature. In most cases, the circuits include two elements which are able to store, either statically (latch) or dynamically (capacitor), the information derived from the SRO (see Definition 8.1) [6–9].

10.2.1 Read

Thanks to a sequence of SROs, it is possible to determine whether the cell's V_{THR} is above or below the three voltage levels V_{READ1}, V_{READ2} and V_{READ3}, which are interleaved between distributions E, D1, D2 e D3 respectively (see Fig. 10.8).

Petitioner Micron Ex-1021, 0273

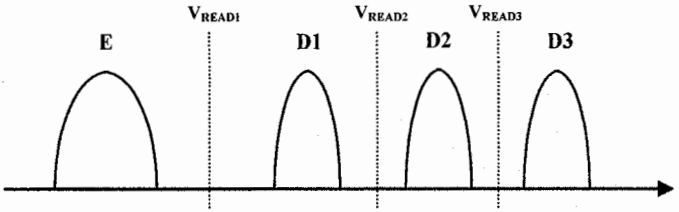


Fig. 10.8. Read voltage levels

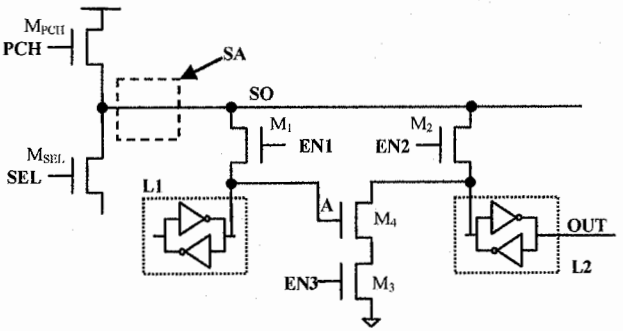


Fig. 10.9. Basic MLC sensing circuit

The sense amplifier gets the data from the SROs and provides the bit associated with either the lower-page or the upper-page on the output. Figure 10.9 shows a circuit that can be used to read the distributions sketched in Fig. 10.1. Only the elements relevant to read operation are described.

In this schematic, two static latches L1 and L2 are able to store the value of node SO by enabling NMOS M_1 and M_2. Node SO corresponds to the SO node of an interleaving architecture, or it can be considered as the output of a sense amplifier SA of an ABL architecture (see Chap. 8). Usually, a single latch provides the output data: in the case shown in Fig. 10.9, it is L2.

Figure 10.10 shows the sequence of operations needed to discriminate the upper-page. Only one SRO at V_{READ2} is required: if cell's V_{THR} is below V_{READ2}, then the upper page is "1" (no matter whether it belongs to E or D1), otherwise it is "0" (no matter whether it belongs to D2 or D3).

The logic value on the node SO is:

- "0" If V_{THR} belongs to either E or D1 distributions
- "1" If V_{THR} belongs to either D2 or D3 distributions

By enabling M_2, the SO logic data is stored inside the latch L2 and its output node OUT is equal to "1100" in case of V_{THR} belonging to distributions E, D1, D2 and D3 respectively (upper-page coding).

Petitioner Micron Ex-1021, 0274

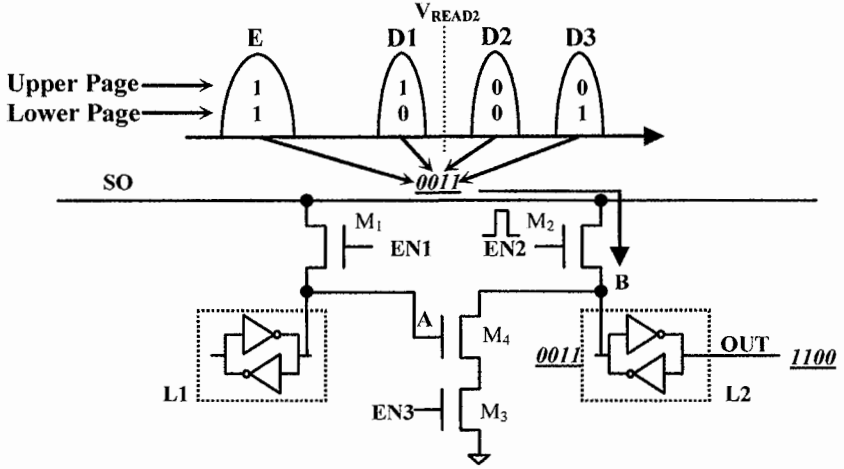


Fig. 10.10. MLC upper page read

For a proper read of the lower-page, it is necessary to have two SROs at V_{READ1} and V_{READ3}: if the distribution is between V_{READ1} and V_{READ3}, then the lower-page is "0", otherwise it is "1".

After the SRO at V_{READ1} (see Fig. 10.11a), the logic value on the node SO is:

- "0" If V_{THR} belongs to E distribution
- "1" If V_{THR} belongs to D1, D2 or D3

By enabling M_2, the SO logic data is stored inside the latch L2 and OUT is "1000". Figure 10.11b shows the next SRO at V_{READ3}. SO logic value is:

- "0" If V_{THR} belongs to E, D1 or D2
- "1" If V_{THR} belongs to D3

By enabling M_1, the SO logic data is stored inside the latch L1 and its node A is equal to "0001". The correct value of the lower-page can be derived by means of a logic OR between OUT and A nodes:

$$lower - page = A \oplus OUT \tag{10.2}$$

This operation is carried out by enabling M_3 (see Fig. 10.11c): the pair of M_3 and M_4 transistors is conductive, thus forcing node B to ground only if node A is "1" (V_{THR} in D3). Node OUT goes from "1000" to "1001", which corresponds to the coding of the lower-page.

Several different techniques exist to elaborate SROs output data in order to get the logic value stored inside the cell. It is not possible to describe all these techniques with few pages: the most popular are described in the following sections.

Petitioner Micron Ex-1021, 0275

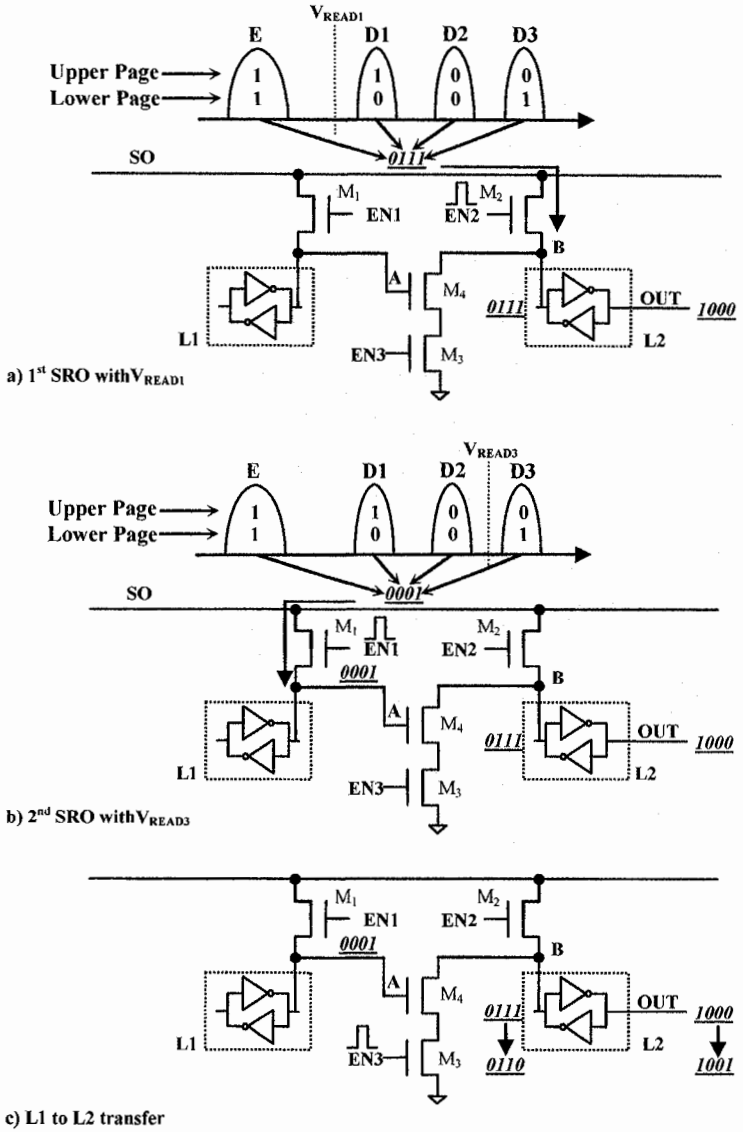


Fig. 10.11. MLC lower-page read

10.2.2 Cache read

Read operation involves a high number of cells in parallel, and each single bit is stored inside a latch, which corresponds to latch L2 in the previous example. Even if a high number of bits is available in L2 latches (4–8 KB typically), the output parallelism of the device is limited to 8–16 bits.

Petitioner Micron Ex-1021, 0276

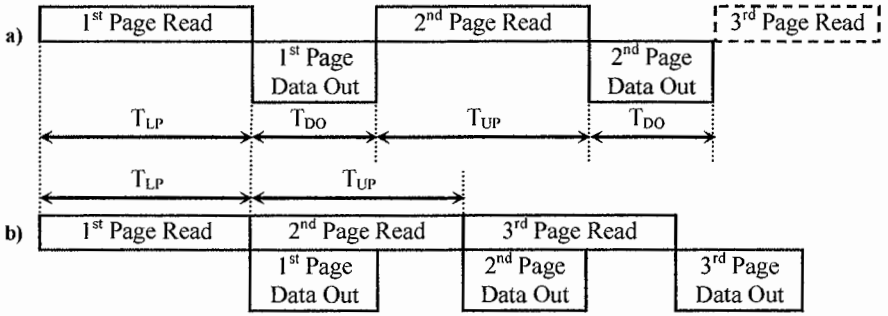


Fig. 10.12. Cache read

Since a new output data is usually available every 25–50 ns, the duration of data-out operation is in the order of tens of microseconds, i.e. comparable to a SRO.

NAND devices are mostly used for mass storage, and several pages are read in sequence. Therefore, it is advisable to make read operation independent of data-out operation, in order to reduce the overall data access time: such architecture is referred to as cache read.

Figure 10.12a shows a NAND memory device without cache read. Average page access time is equal to:

$$T_R = \frac{T_{LP} + T_{UP}}{2} + T_{DO} \tag{10.3}$$

where T_{LP} and T_{UP} represent the read time of the lower-page and of the upper-page respectively, while T_{DO} is the page data-out time.

In case of a NAND device with cache read (Fig. 10.12b), average page access time is equal to:

$$T_R = \frac{T_{LP} + T_{UP}}{2} \tag{10.4}$$

Whenever T_{DO} is greater than T_{LP} and T_{UP}, page access time is equal to T_{DO}.

Not all sensing circuits are able to manage a cache read: the solution shown in Fig. 10.9 does not work because both L1 and L2 latches are involved in the read operation of the lower-page. Of course, a third latch could be added, but it would not be acceptable because of the consequent area increase of the sensing circuits.

Figure 10.13 shows a solution able to implement a cache read architecture: latch L1 and the auxiliary circuit composed of M_1, M_2 and M_{SO} are able to perform the same operations as the sensing circuit described above, but without the need of L2. Therefore, L2 can be used as a buffer for data-out operations. Let's analyze the behavior in detail.

At the beginning, SET signal is "high" and M_5 drives the node OUT "low"; SO is "high", M_1 is on and, therefore, node A is "high". Figure 10.14a shows the sequence of operations performed to read the upper-page. A SRO is done at V_{READ2}.

Petitioner Micron Ex-1021, 0277

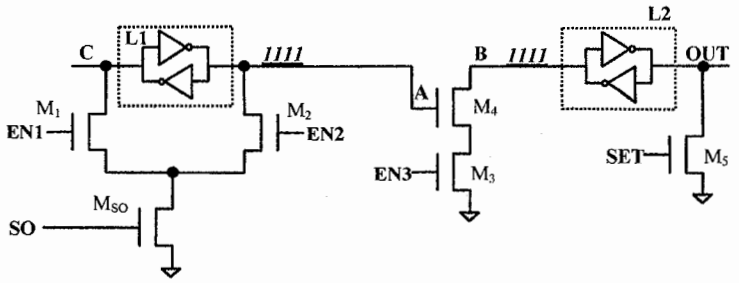


Fig. 10.13. MLC with cache read

In the four cases, SO logic value is equal to "0011". Enabling M_2, node A is forced to ground only if M_{SO} is on, that is only when node SO is "1". Therefore the final state for node A is "1100".

In order to implement cache read, the data stored in L1 is transferred to L2, by enabling M_3, as shown in Fig. 10.14b: node B toggles from "1111" to "0011", and OUT becomes "1100".

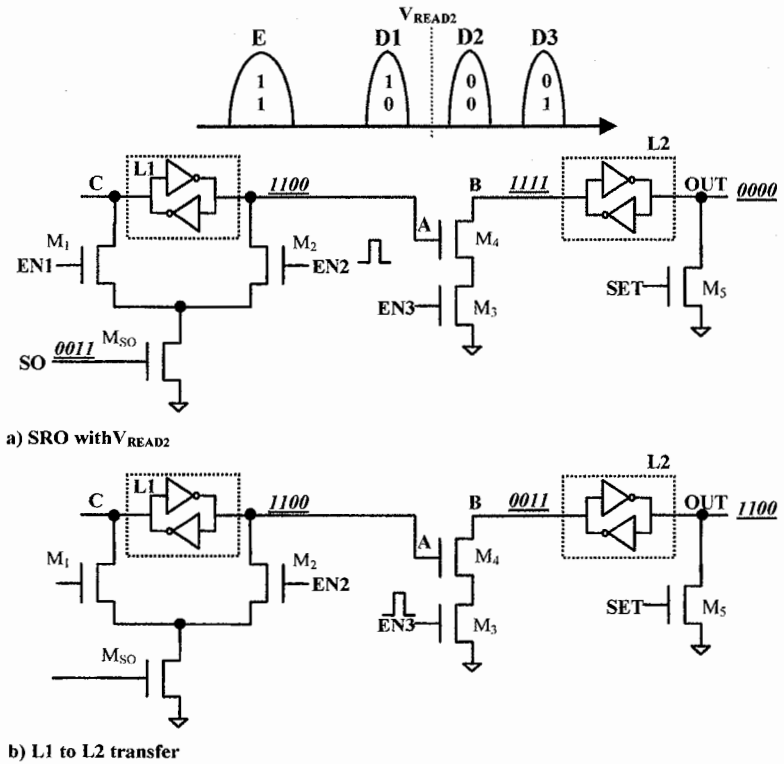


Fig. 10.14. MLC with cache read: upper-page read

Petitioner Micron Ex-1021, 0278

While L2 is used for data-output operations, latch L1 can be used to read the lower-page. Latch L1 is brought back to the conditions shown in Fig. 10.13 where node A is forced to "1" and two SROs are executed one after another, as shown in Fig. 10.15a and b.

First SRO is done at V_{READ1}: node SO is equal to "0111", M_2 is enabled, and node A changes from "1111" to "1000". Second SRO is at V_{READ3}: node SO is equal to "0001", M_1 is enabled, and node C changes from 0111 to 0110. Therefore, A is equal to "1001".

In the meanwhile, latch L2 has finished the data-out phase and it is now possible to transfer the data from L1 to L2. B is set to "1". The transfer operation takes place in the usual way, by enabling M_3. The lower-page "1001" becomes available on node OUT (Fig. 10.15c). Latch L1 is now free and can be used to read the following page; at the same time L2 is busy performing the data-out of the lower-page.

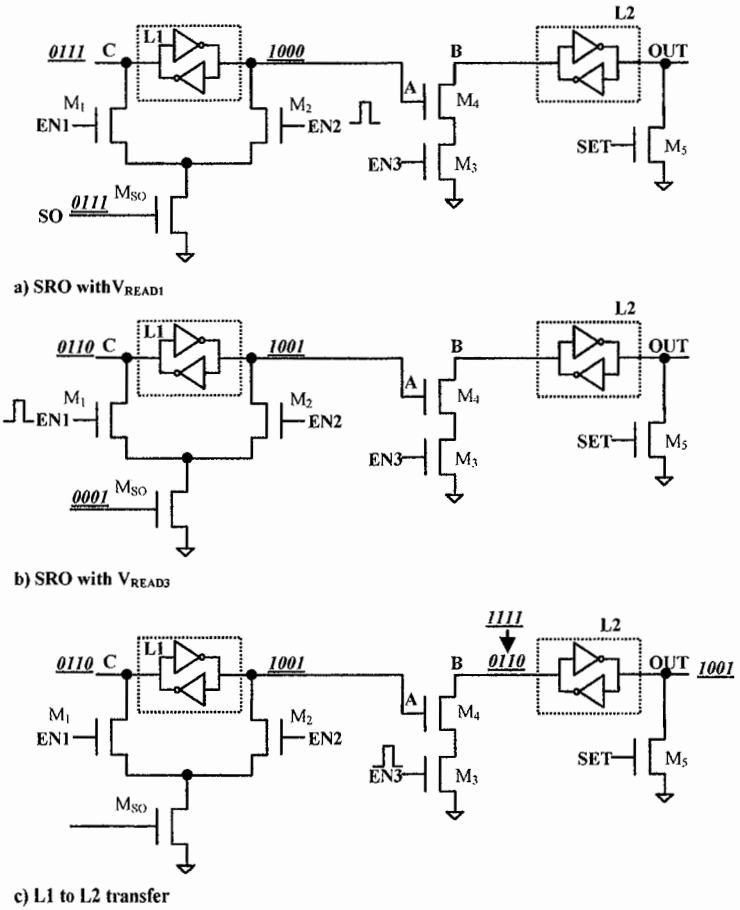
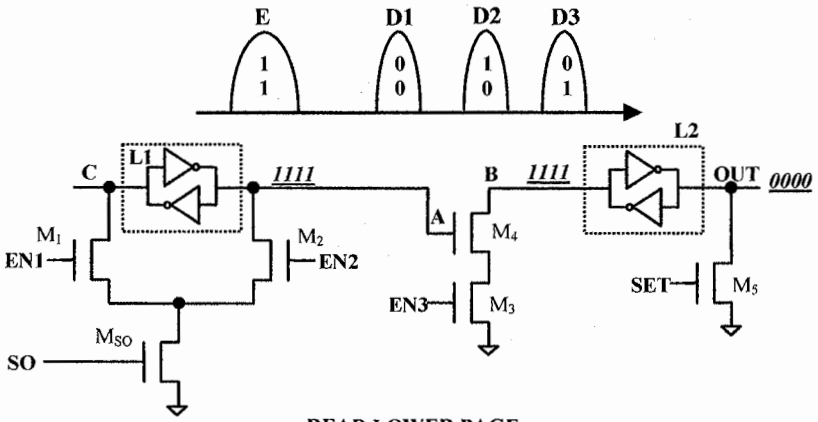


Fig. 10.15. MLC with cache read: lower page read

Petitioner Micron Ex-1021, 0279

Such sensing circuit can also be used to implement the read operation for other types of coding, like the one shown in Fig. 10.15. As described in the table of Fig. 10.16 (ENX column indicates the enabling signal), the read of the upper-page is done by two SROs: the former using V_{READ1} and the latter using V_{READ2}. Read of the lower-page needs three SROs, at V_{READ1}, V_{READ2} and V_{READ3}. In this case as well, latch L1 is busy doing the read operation while latch L2 performs the data-out, which means that even this structure belongs to the cache read category.

The implementation of the read operation for the type of coding shown in Fig. 10.7 can be easily derived: since the coding of the upper-page and the coding of the lower-page are swapped, it is sufficient to swap the labels between the upper-page and the lower-page in the table of Fig. 10.16.



READ LOWER PAGE

	SO	ENX	A	C	B	OUT
Start	xxxx	x	1111	0000	1111	0000
V_{READ1}	0111	EN2	1000	0111	1111	0000
V_{READ3}	0001	EN1	1001	0110	1111	0000
L1 TO L2 Transfer	xxxx	EN3	1001	0110	0110	1001

READ UPPER PAGE

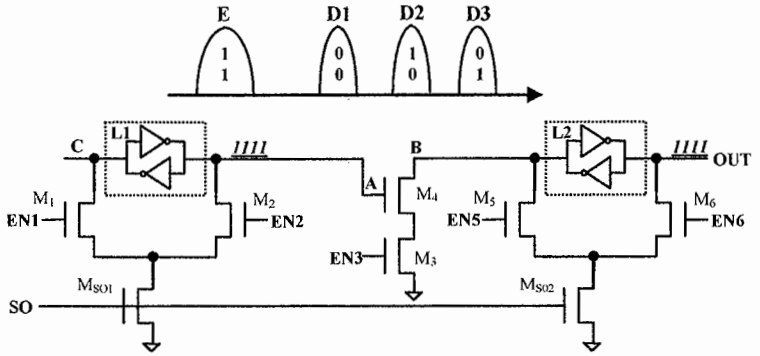
	SO	ENX	A	C	B	OUT
Start	xxxx	x	1111	0000	1111	0000
V_{READ1}	0111	EN2	1000	0111	1111	0000
V_{READ2}	0011	EN1	1011	0100	1111	0000
V_{READ3}	0001	EN2	1010	0101	1111	0000
L1 TO L2 Transfer	xxxx	EN3	1010	0101	0101	1010

Fig. 10.16. MLC with different distribution coding and cache read

Petitioner Micron Ex-1021, 0280

It should be noted that using the solution shown in Fig. 10.13 it is possible to read both pages using only three SROs, while the example shown in Fig. 10.16 requires five SROs.

It should always be possible to discriminate the position of the threshold voltage using only three SROs. For instance, the solution shown in Fig. 10.17 implements the simultaneous read of the two pages: the two latches, L1 and L2, work in parallel during SROs. At the end of three SROs, the lower-page is on node A, and the upper page is on node OUT.



READ LOWER AND UPPER PAGE

	SO	ENX	A	C	B	OUT
Start	xxxx	x	1111	0000	0000	1111
V_{READ1}	0111	EN2/EN6	1000	0111	0111	1000
V_{READ2}	0011	EN1	1011	0100	0111	1000
V_{READ3}	0001	EN2/EN5	1010	0101	0110	1001
			LOWER PAGE		UPPER PAGE	

Fig. 10.17. MLC page buffer for reading two pages

10.2.3 MLC program/verify operations

During program operation, the same program pulse (at a gate voltage equal to V_{PGM}) is applied to all the cells belonging to the same wordline. It is responsibility of the sensing circuit to verify and, if necessary, inhibit those cells which have reached the desired distribution. In particular, the cell is inhibited and it does not change its threshold V_{THR} any longer if during the program step the corresponding bitline is biased to a voltage which is high enough to switch off selection transistor M_{BLS} (Fig. 10.6). The voltage applied to the bitline is typically equal to either V_{DD} or $V_{DD}-V_{THN}$. On the other hand, the cell increments its threshold voltage if during the program step the bitline is biased to a voltage which is low enough to

Petitioner Micron Ex-1021, 0281

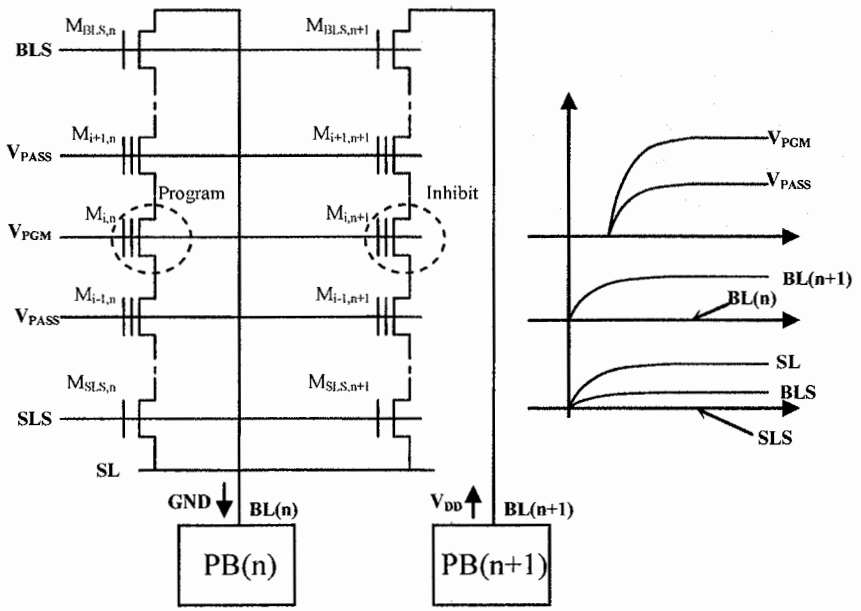
switch on M_{BSL} so that it can act as a pass transistor, Typically, the bitline is biased to ground and the situation just described is shown in Fig. 10.18.

In order to switch-off the transistor M_{BSL}, its gate is biased to the lowest possible voltage, considering that M_{BSL} has to be on when the bitline is forced to ground: typical values lie between 1 and 2 V.

Program of the upper-page is the most complex operation and its analysis allows us to understand the fundamental elements which must be present inside the sensing circuit. First of all, in all the coding cases previously seen, two information are needed in order to program the upper-page properly:

- Which is the initial distribution: the state of the lower-page must be known and stored in the sensing circuit. If the lower-page has already been programmed, the information must be recovered from the cell.
- Which is the target distribution: this information, given by the value of the bit of the upper-page, is provided by the user and it must be stored in the sensing circuit.

Since two pieces of information must be stored, it is evident that at least two latches are needed inside the sensing circuit. Furthermore, the circuit has to be interfaced to a logic-unit capable of loading the information that program needs (the data-load described in Sect. 10.3).



- PB(n) discharge BL(n) to ground: cell $M_{i,n}$ programmed

- PB(n+1) charge BL(n+1) to V_{DD}: cell $M_{i,n+1}$ inhibited

Fig. 10.18. Cells program and inhibit

Petitioner Micron Ex-1021, 0282

Since the final state of the cell depends on the combination of the data stored inside the latches, it is necessary that an information exchange takes place between L1 and L2. Finally, the sensing circuit is connected to the bitline (through node SO) not only for the SRO, but also to provide the correct biasing of the bitline during the ISPP program phase (Chap. 3). Figure 10.19 depicts the features of the sensing circuit during program/verify operations.

This reference scheme can be used to devise the algorithm and the corresponding circuit: a possible solution implementing the write operations for a MLC architecture using the coding of Fig. 10.1 is presented below.

Figure 10.20a shows the flow diagram describing how to program the lower-page: information is loaded (data-load) in the latch L1:

- L1 = 1 means that the cell must be inhibited.
- L1 = 0 means that the cell must be programmed.

In the former case, the cell is excluded from following program steps by forcing the bitline to V_{DD}: even if the ISPP program step is applied to the gate of this cell, its V_{THR} does not change.

In the latter case, the bitline is forced to ground and, during the ISPP program step, its V_{THR} is incremented. Verify operation follows, which is identical to a SRO performed at V_{VFY1}:

- If V_{THR} is lower than V_{VFY1}, then the cell has not reached D1 distribution. The latch does not change its state (L1 = 0): at the next ISPP step, the bitline is forced to ground again in order to increment V_{THR} further.
- If V_{THR} is higher than V_{VFY1}, then the cell has reached D1. The latch changes its state (L1 = 0→1): at the next ISPP step, the bitline is forced to V_{DD} and the cell is inhibited during the following program steps.

It is important to highlight that, once a cell is inhibited, it remains inhibited throughout all the following program operations of the lower-page. In other words, an inhibited cell never takes part in the following program steps.

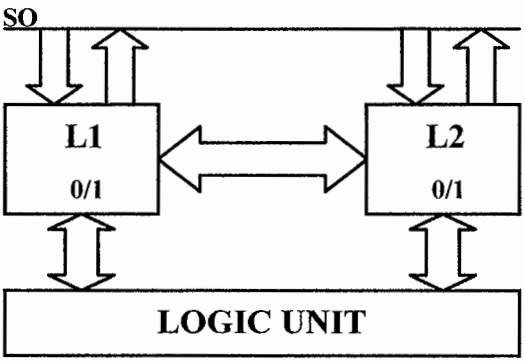


Fig. 10.19. Sensing circuit during program/verify operations

Petitioner Micron Ex-1021, 0283

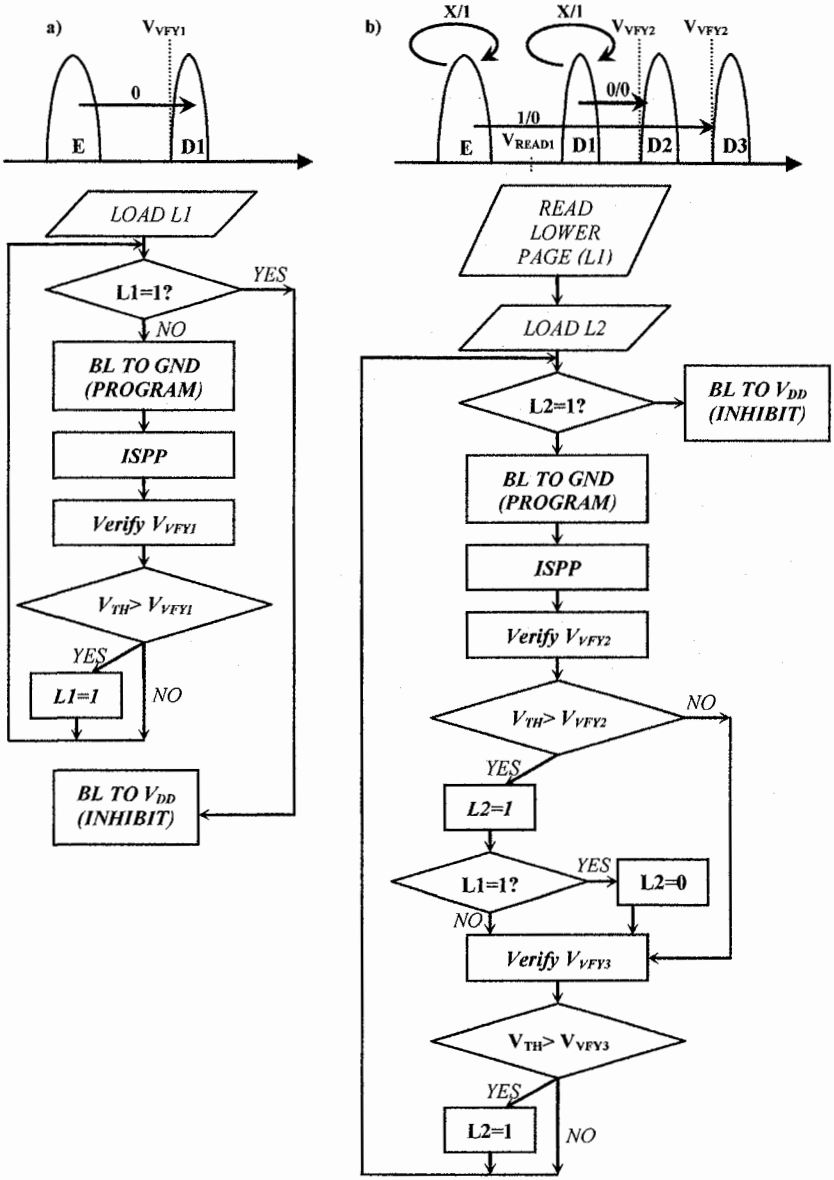


Fig. 10.20. Lower-page and upper-page program operations

Figure 10.20b shows the flow diagram describing how to program the upper-page: information is loaded (data-load) in the latch L2:

- L2 = 1 means that the cell must be inhibited (it remains in either E or D1 state).
- L2 = 0 means that the cell must be programmed to either D2 or D3 state.

Petitioner Micron Ex-1021, 0284

Meanwhile the lower-page is read and stored in latch L1.

- L1 = 1 means that the starting state is E.
- L1 = 0 means that the starting state is D1.

If L2 = 1, no program operation takes place: the bitline is forced to V_{DD} and the cell is inhibited. If L2 = 0, V_{THR} has to be incremented (bitline forced to ground).

At the end of the ISPP step, the first verify is performed at V_{VFY2}:

- If V_{THR} is lower than V_{VFY2}, then the cell has not even reached D1. The latch does not change its state (L2 = 0): at the next ISPP step, the bitline is forced to ground again in order to increment V_{THR} further;
- If V_{THR} is higher than V_{VFY2}, then the cell has reached D2 and the latch changes its state (L2 = 0→1). If the starting distribution was D1 (L1 = 0), then the cell has reached the target distribution. Otherwise, if the starting distribution was E (L1 = 0), then the content of latch L2 is reset to 0 because the cell has to reach D3.

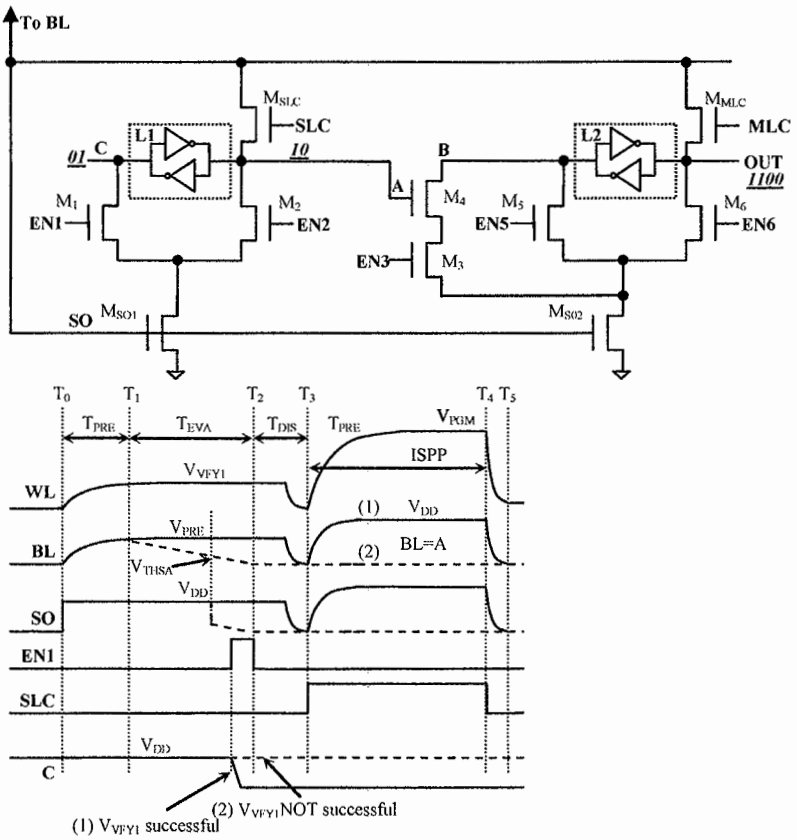


Fig. 10.21. MLC lower-page program

Petitioner Micron Ex-1021, 0285

Verify at V_{VFY3} is then performed:

- If $V_{THR} < V_{VFY3}$, then the cell has not reached D3. The latch L2 does not change its state (L2 = 0): at the next ISPP step, the bitline is forced to either ground (if L2 = 0) or V_{DD} (if L2 = 1): the latter case occurs if in the previous verify (performed at V_{VFY2}) the cell reached D2 starting from D1.
- if $V_{THR} > V_{VFY3}$, then the cell has reached D3 and the latch changes its state (L2 = 0→1).

Figure 10.21 shows the circuit implementing program operation of the lower-page and the timings of related signals. On node A of latch L1 the data-load of the lower-page occurs:

- C = 0 (A = 1) means that the cell must be inhibited (it remains in the E state).
- C = 1 (A = 0) means that the cell must be programmed to the D1 state.

At time T_0 a SRO at V_{VFY1} starts, and it ends at time T_2: the value of node SO is stored in L1, by enabling M_1:

- If $V_{THR} < V_{VFY1}$ (verify at V_{VFY1} NOT successful), then SO = 0: the series $M_{SO1} - M_1$ is not conductive and, therefore, the latch keeps its state (dotted line in Fig. 10.21).
- if $V_{THR} > V_{VFY1}$ (verify at V_{VFY1} successful), then SO = 1: the series $M_{SO1} - M_1$ is conductive and, therefore, C node is forced to ground (solid line in Fig. 10.21).

At time T_3, M_{SLC} is enabled and the bitline forcing (to the voltage of node A) takes place during the ISPP step:

- If A = 0 the bitline is forced to ground: this situation occurs when verify at V_{VFY1} is NOT successful AND data-load C = 1.
- If A = 1 the bitline is forced to V_{DD}: this situation occurs when verify at V_{VFY1} is successful OR data-load C = 0.

The circuit shown in Fig. 10.21 also implements the permanent-inhibit because during verify operation node C can only be forced to ground trough $M_{SO1} - M_1$ but never to "1": therefore, node A (and the bitline) cannot get back to "0" once forced to "1". At time T_5 the ISPP step ends, and the cycle restarts from T_0; in every $T_0 - T_5$ cycle the program voltage V_{PGM} is incremented.

The circuit shown in Fig. 10.21 is also able to implement the program operation of the upper-page, as shown in Fig. 10.22.

The lower-page is read in a SRO at V_{READ1}, by enabling M_1 (node A was previously forced to "0"). Node A remains at "0" if the lower-page is in E, while it is forced to "1" if the lower-page is in D1.

Latch L2 stores the information of the upper-page:

- B = 0 means that the cell must be inhibited (it remains in either E or D1 state).
- B = 1 means that the cell must be programmed to either D2 or D3 state.

Petitioner Micron Ex-1021, 0286

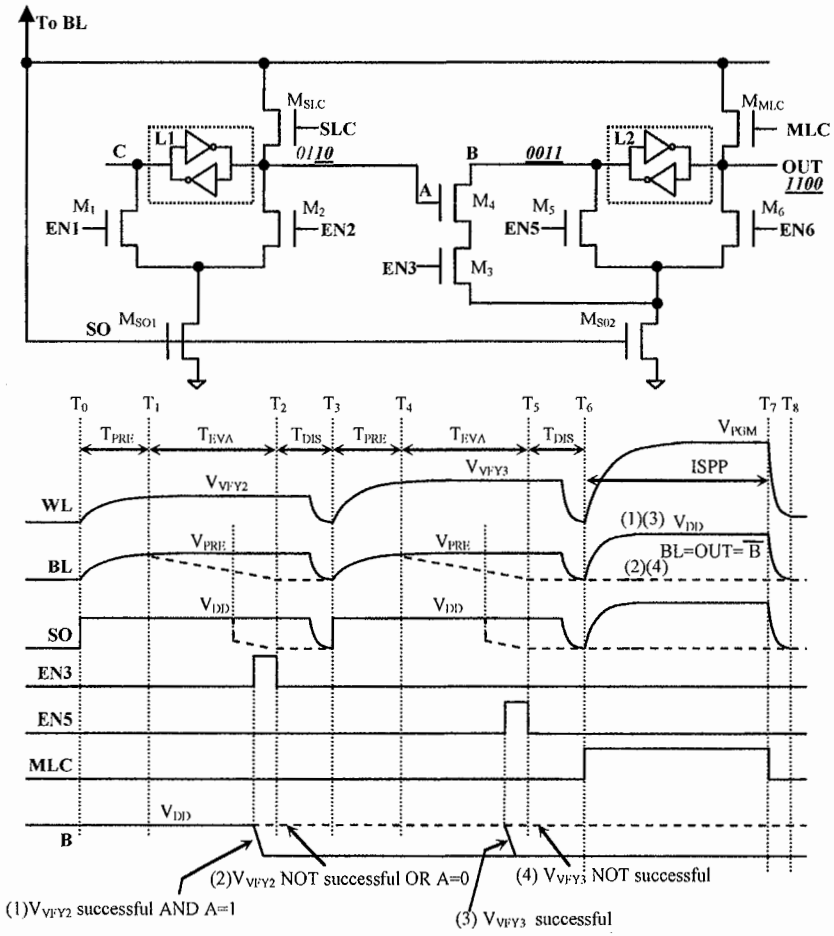


Fig. 10.22. MLC upper-page program

Latch L2 is used to drive the bitline during the ISPP through transistor M_{MLC}: node OUT is forced to "1" (forcing B to "0") if and only if the cell has reached the desired distribution. Of course, two verify operations are needed, one at V_{VFY2} and one at V_{VFY3}. If the verify at V_{VFY2} is successful, node OUT must be forced to "1" only if the cell has to move to distribution D2.

Therefore, the forcing to "0" of node B of L2 should be conditioned to the content of latch L1 by means of the series of transistors M_{SO2}, M_3 and M_4. As usual, node SO is equal to "1" if the cell has crossed V_{VFY2}, otherwise SO = 0:

- If A = 0 (the cell should be moved to D3) M_4 is always off and B cannot be forced to "0" by node SO in any way (case (2) in Fig. 10.22).
- If A = 1 (the cell should be moved to D2) M_4 is always on: by enabling M_3, node B is forced to "0" if SO = 1 (case (1) in Fig. 10.22).

Petitioner Micron Ex-1021, 0287

Verify using V_{VFY3} follows (independently from L1) and it starts enabling M_5: if the verify is successful, node SO is equal to "1" (case (3) in Fig. 10.22) and node B is forced to ground, otherwise B remains at "1" (case (4) in Fig. 10.22).

At time T_6 the MLC signal is enabled and the bitline, during the ISPP step, is forced to the value of node OUT. If the cell has reached the desired distribution, bitline BL is forced to V_{DD} (solid line in Fig. 10.22), otherwise to ground (dotted line in Fig. 10.22).

At time T_8, ISPP step ends and the cycle restarts from T_0; in every T_0 - T_8 cycle the program voltage V_{PGM} is incremented.

Such scheme can also be used for a MLC program operation featuring the coding shown in Fig. 10.7: program sequences of lower-page and upper-page are the same as the one shown in Figs. 10.21 and 10.22, respectively. The difference is that the lower-page read before upper-page program is performed in a SRO at V_{READ1}, by enabling M_2 instead of M_1 (C previously forced to "0").

Concerning the coding shown in Fig. 10.5, a third latch is needed because of the reprogram of distribution D2' at V_{VFY2}.

10.2.4 Check circuits

Sensing circuits have to report to the logic-unit of Fig. 10.19 that all the cells involved in a program operation have reached the desired state. Considering, for example, the case of the upper-page shown in Fig. 10.22, latch L2 moves its OUT node to "1" when the cell has reached the target distribution. When all the OUT nodes of all the sensing circuits go to "1", program operation has completed successfully. A simple circuit implementing this kind of control is shown in Fig. 10.23.

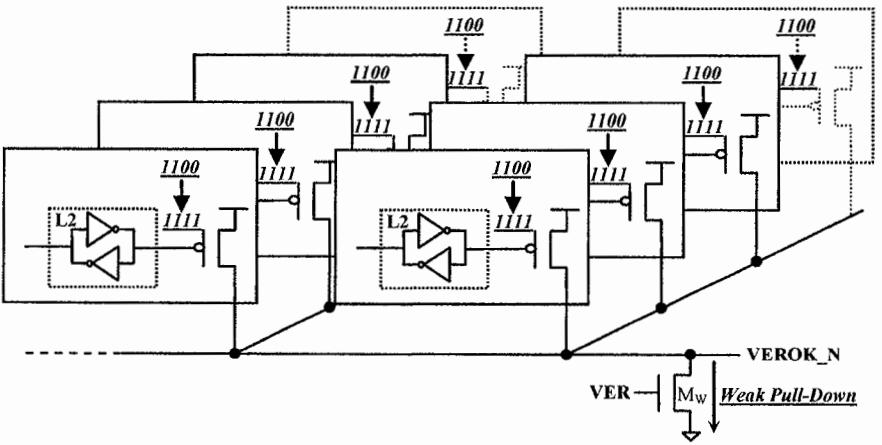


Fig. 10.23. Check circuit for verify operations

Petitioner Micron Ex-1021, 0288

Node OUT is connected to the gate of a PMOS which shares a node (VEROK_N) with all the other PMOS. Node VEROK is connected to a NMOS M_W which works as a pull down, weak enough not to defeat a PMOS potentially on (it is a wired-NAND structure).

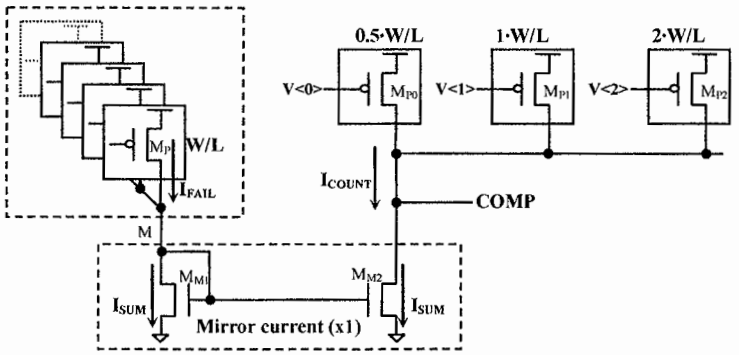
At the end of the verify phase, VER signal enables the M_W tied to the node VEROK_N:

- If at least one of the PMOS is on, then at least one of the OUT nodes is at ground: node VEROK_N is forced to "1" (program has not finished yet).
- If all the PMOS are off, then all the OUT nodes are at "1": node VEROK_N is discharged to ground by M_W. Program phase is completed because all the cells have reached the desired distribution.

It is possible that some cells don't reach the target distribution even at the last ISPP step. If the number of failing cells is small enough (1–4), it is possible to correct the page content by means of ECC algorithms (Chap. 14). Unfortunately, the above-mentioned circuit is not able to count the number of failing cells.

Figure 10.24 shows a circuit able to count the number of failing cells. Even in this case, the PMOS M_P are connected in parallel, but their common node enters into a current mirror. Each of the conductive M_P transistor provides a I_{FAIL} current which sums up on node M to generate I_{SUM}.

I_{SUM} current is mirrored on node COMP and compared with a reference current I_{COUNT}, whose value depends on the combination of "on" transistors M_{P0}, M_{P1}, M_{P2}. The size (W/L) of transistor M_{P1} is the same as M_P and M_{P2} and it is composed of two M_P in parallel, while the size of M_{P0} is chosen in such a away that when it is on, its current is exactly half as much as I_{FAIL}. Signals V<2:0> enables the different combinations.



V<0>	V<1>	V<2>	I_{COUNT}	COMP=1
1	0	0	$0.5 \cdot I_{FAIL}$	fail nr. =0
1	1	0	$1.5 \cdot I_{FAIL}$	fail nr. <2
1	0	1	$2.5 \cdot I_{FAIL}$	fail nr. <3
1	1	1	$3.5 \cdot I_{FAIL}$	fail nr. <4

Fig. 10.24. Fail counter

Petitioner Micron Ex-1021, 0289

In order to fully understand the behavior, let's consider the case when only three OUT nodes are at ground (three failing cells) and therefore I_{SUM} current is three times I_{FAIL}. When all the three PMOS are on (all V<2:0> signals to ground), I_{COUNT} is equal to:

$$I_{COUNT} = 3.5 \cdot I_{FAIL} \tag{10.5}$$

Since $I_{COUNT} > I_{SUM}$, node COMP is "1" and it indicates that the number of failing cells is less than (or equal to) three. Switching off MP1 (V<1> to V_{DD}) it follows that:

$$I_{COUNT} = 2.5 \cdot I_{FAIL} \tag{10.6}$$

Since $I_{COUNT} < I_{SUM}$, node COMP is "0" and it indicates that that the number of failing cells is greater than 2. Combining the information, it easily follows that the number of failing cells is 3.

The circuit shown in Fig. 10.24 can be used to determine whether the number of failing cells is 0, 1, 2 or 3, as shown in the table. Of course, by properly adding other PMOS to the I_{COUNT} generator, the number of failing bits that can be computed increases.

10.2.5 Coarse and fine programming

The width of written distributions is closely related to the step amplitude ΔISPP of the program voltage. The small black squares in Fig. 10.25 show the linear relationship between the increase of the threshold voltage of the cell V_{TH} and the programming step: 0.2 V increase on V_{PGM} translates into 0.2 V increase on V_{TH}. This is true if the bitline voltage is tied to a fixed potential, for instance at 0 V.

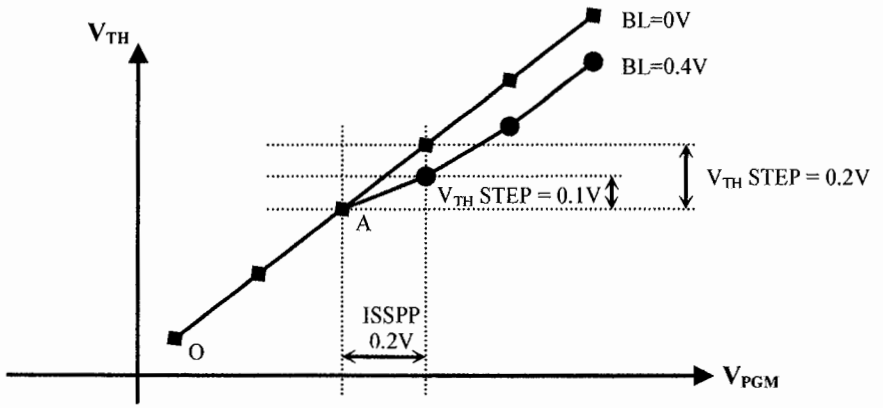


Fig. 10.25. Cell threshold voltage (V_{TH}) vs. ISPP step

Petitioner Micron Ex-1021, 0290

Because of the amplitude of the program step, the distribution's width is exactly 0.2 V. In fact, let's consider a cell which should be programmed beyond V_{VFYH}: in the best case scenario, the final step causes the cell to slightly go above V_{VFYH} (Fig. 10.26a), while in worst case the cell is slightly below V_{VFYH}, and in the following step it will get to a distance of 0.2V from V_{VFYH} (Fig. 10.26b).

Figure 10.25 also illustrates the case where the potential of the bitline in point A is changed (for instance to 0.4 V) and the characteristic with V_{BL} at 0 V shifts onto another one (i.e. the one with V_{BL} = 0.4 V) which is identified by the small black circles [10]. In the following ISPP step, the threshold voltage step is smaller than 0.2 V, being approximately 0.1 V. Then, the relationship is again a linear one, at 0.2 V/step.

By modulating the bitline voltage at 0.4 V when the cell is next to the V_{VFYH} level (as is the case shown in Fig. 10.26b), it is possible to reduce the V_{TH} step, introducing another verify level, V_{VFYF}, 0.1 V smaller than V_{VFYH}.

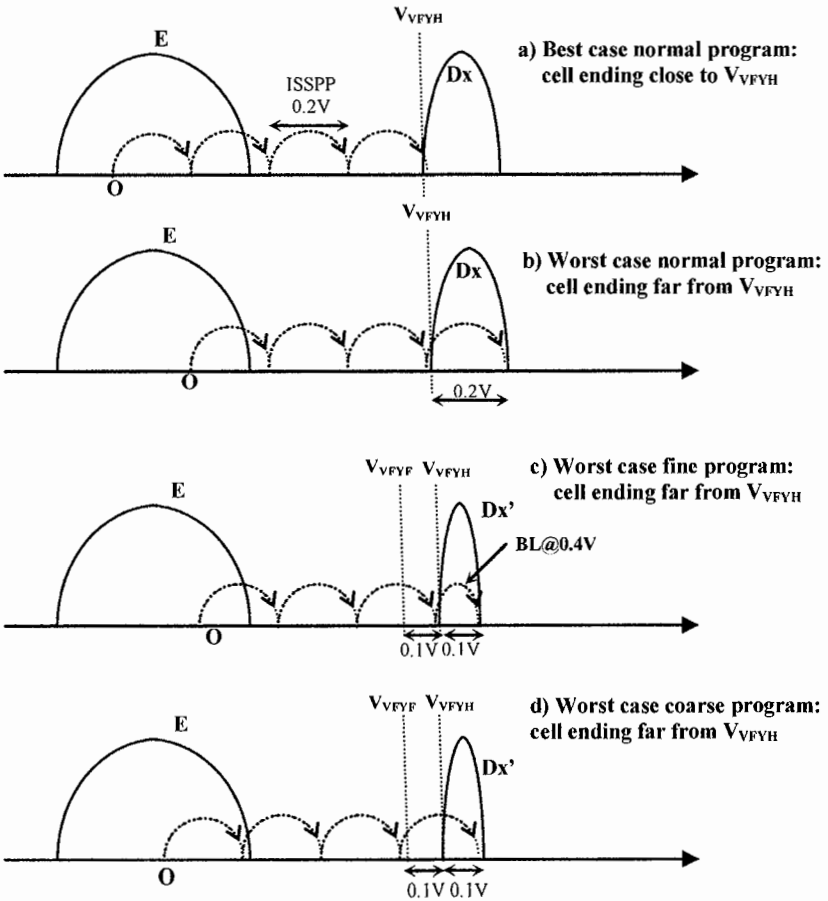


Fig. 10.26. Coarse and fine programming for distribution narrowing

Petitioner Micron Ex-1021, 0291

If the cell lies between V_{VFYF} and V_{VFYH} levels (case shown in Fig. 10.26c), the bitline is biased at 0.4 V so that its cell's V_{TH} is incremented of 0.1 V during the following program step. In any case, the cell goes above V_{VFYH} level and, in the worst case scenario, it is 0.1 V far from V_{VFYH} instead of 0.2 V. Otherwise, if the threshold voltage is lower than V_{VFYF} (case shown in Fig. 10.26d), a standard program step, with the bitline biased at 0 V, is applied. In the worst case scenario, the cell's threshold is 0.1 V higher than V_{VFYH}.

Therefore, the distribution width of the programmed cell has been decreased by 0.1 V. This coarse and fine programming technique can be used with any of the three distributions of the multilevel memory. Let's consider the case where such technique is only applied to distribution D1 of the lower-page. Figure 10.27 shows both the implemented circuit and the signal timings which allows a coarse and fine programming of distribution D1.

As previously explained, latch L1 is used to program the lower-page and it stores the indication whether the cell needs to reach D1 (A = 1) or not (A = 0). The latch L2 stores the information whether the cell has gone above V_{VFYF} (its node B is equal to "1" at the beginning). The task of the latch L1 is to bias the bitline to either GND or $V_{DD} - V_{THN}$, while the latch L2 biases the bitline to 0.4 V when the cell has gone above V_{VFYF} level. Voltage V_{FINE} is equal to:

$$V_{FINE} = 0.4 + V_{THN} \tag{10.7}$$

where V_{THN} is the threshold of M_8.

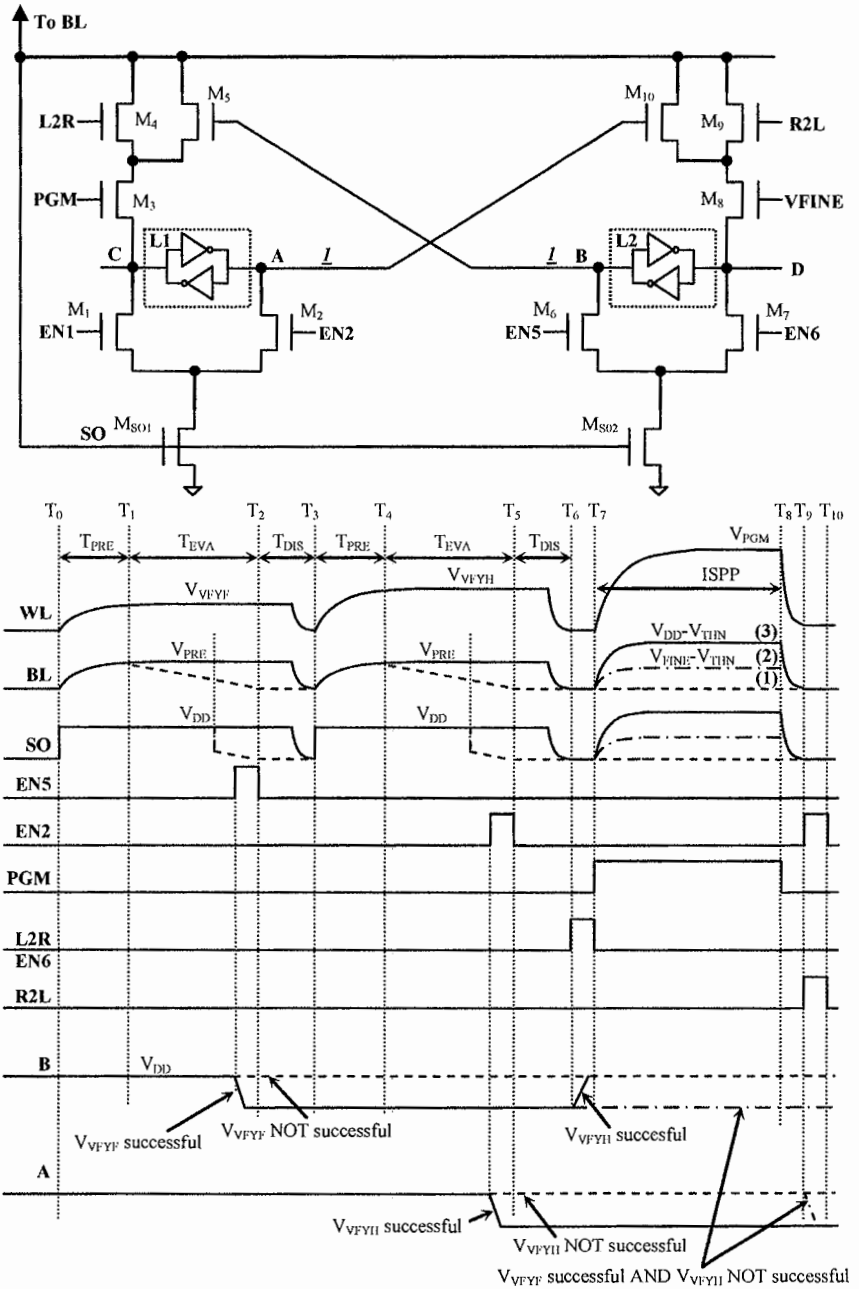
Neglecting transistors M_4 and M_9, used for data transfer between the latches, it can be seen that the path connecting node C to the bitline depends on the state of M_5, whose gate is controlled by L2, while the path connecting node D to the bitline depends on the state of M_{10}, whose gate is controlled by L1.

The situation shown in Fig. 10.27 (A = 1, B = 1) corresponds to the starting condition when the cell has to be programmed: the bitline has to be forced to ground before the first step. In fact, nodes D and C are equal to "0"; by enabling M_3, both latches L1 and L2 force the bitline to ground through the pairs of NMOS M_3–M_5 and M_8–M_{10}, respectively. If the cell does not have to be programmed (A = 0), latch L2 is excluded while M_3 is enabled: therefore, C drives the bitline to $V_{DD} - V_{THN}$.

Two consecutive SROs are performed on the cell, one using V_{VFYF} and another using V_{VFYH}, as shown by the timings in Fig. 10.27. The outcome of the verify at V_{VFYF} is stored in latch L2 (enabling M_6), while the result of the verify at V_{VFYH} is stored in L1 (enabling M_2). At the end, three different cases might occur:

1. The cell has $V_{THR} < V_{VFYF}$ (SO = 0): node B remains at "1" and nothing changes (obviously the cell is not going to succeed in the following verify at V_{VFYH} either, and therefore latch L1 is not changing its state). This situation is represented by the dotted line in Fig. 10.27.
2. The cell has $V_{THR} > V_{VFYF}$ (SO = 1) but $V_{THR} < V_{VFYH}$ (SO = 0): node B goes to "0" while node A remains at "1". Latch L1 is not driving the BL which is biased at 0.4 V by L2 through the series M_8–M_{10} (D is now a "1"). This situation is represented by the dotted/dashed line in Fig. 10.27.

Petitioner Micron Ex-1021, 0292



(1) A=1 and B=1 Any Verify successful (PROGRAM BL@GND)
(2) A=1 and B=0 V_{VFYF} successful AND V_{VFYH} NOT successful (FINE PROGRAM BL@V_{FINE}-V_{THN})
(3) A=0 and B=1 V_{VFYH} successful (PERMANENT INHIBIT BL@V_{DD}-V_{THN})

Fig. 10.27. Coarse and fine programming: a circuital solution

Petitioner Micron Ex-1021, 0293

3. The cell has $V_{THR} > V_{VFYH}$ (SO = 1) and, obviously, it is also $V_{THR} > V_{VFYF}$: both nodes A and B are at ground, and therefore both latches are excluded. In order to avoid such condition, at time T_6 there is an information transfer from L1 to L2, enabling at the same time both M_4 and M_7: node D is forced to "0" (and node B reverts to "1") if and only if A = 0. This situation is represented by the solid line in Fig. 10.27.

At time T_7 the program pulse starts and the bitline is properly biased in each of the three possible cases. At the end of the program pulse, it is necessary to let-latch L1 know whether the cell has been fine-programmed with BL at 0.4 V so that the cell is inhibited in the following steps. For this reason, at time T_{10} it is necessary to perform an information transfer from L2 to L1 by enabling both M_9 and M_2. Only if node D is "1" (case 2) node A is forced to ground, which is the permanent inhibit condition (case 3).

This technique can be used for D2 and D3 programming (upper-page) as well, adding the latch related to the information contained in the lower-page: the sensing circuit would totally require three latches [11].

10.3 Data-load

Data to be programmed are made available at the pads of the NAND memory and are sequentially loaded inside the latches of the sensing circuit SA. Usually, the number of latches to be loaded corresponds to the size of the page. Data-load operation needs a high number of cycles since a limited number of data pads (8–16) are available to upload a page whose size is in the order of some kilobyte. Some of the most widely used architectures for data-load are presented in this section. The assumption is that page size is 8 KB and the number of data pads is 8 (×8 device).

10.3.1 Data-load 1

Sensing circuits and corresponding latches are arranged in eight blocks of 1,024 rows by eight columns, as shown in Fig. 10.28.

For each data PAD, a whole block is allocated and therefore at each data-load cycle data DQ<i> is stored in one of the 1024 × 8 latches of Block<i>. 1024 × 8 cycles are needed in order to upload a whole page. A set of address bits ADD<12:0> are decoded by X-LOGIC and Y-LOGIC circuits and used to univocally identify the position of the latch inside the block:

- Sub-set ADD<12:10> identify a single row, common to all the blocks, enabling one of the LX<7:0> signals.
- Sub-set ADD<9:0> identify a single column, for each block, enabling one of the LY<1023:0> signals.

Petitioner Micron Ex-1021, 0294

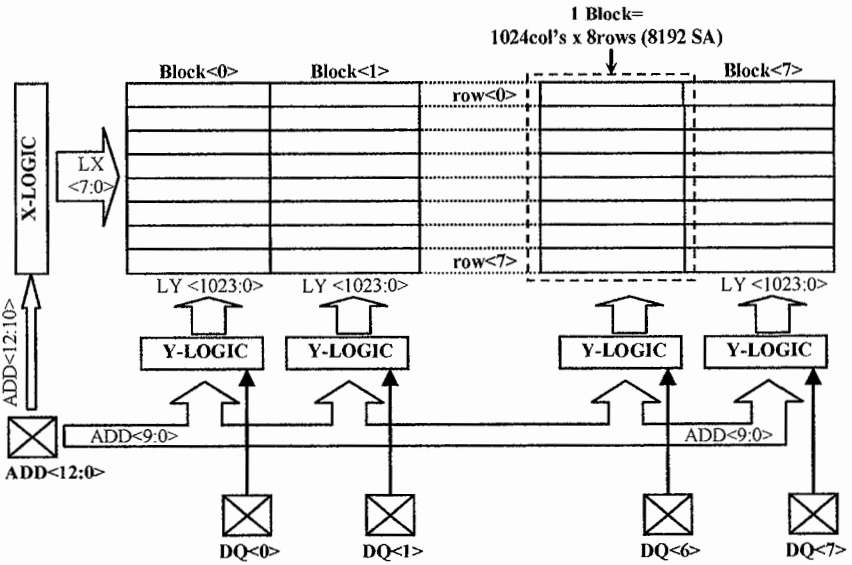


Fig. 10.28. Data-load of a 8 kB page size: first solution (data-load 1)

Figure 10.29 represents data-load process on the latches of column <k> inside block <i> (only the latches of row 7 and 0 are visible). It is possible to force to ground either node A or B inside the latch enabling the corresponding signal LX<n> and one of the signals LY<k> or LY_N<k>.

Signals LY_N<k> and LY<k> are shared among the eight latches of column <k> and the polarity depends on the value of data DQ<i> according to the logic depicted in Fig. 10.29. Neglecting signal SYNC, the result of the decoding of ADD<9:0> enables signals LY_N<k> and LY<k> only for column <k> by enabling the signal COL<k>=1. In all the columns <j> not equal to <k>, signals LY_N<j> and LY<j> are thus forced to ground because COL<j> = 0.

Therefore, inside each block <i> only the latch belonging to row <n> and column <k> is allowed to change its state depending on the polarity of DQ<i>. Figure 10.29 shows the timings of a single cycle of data-load. At time T_0 addresses ADD<12:0> and data DQ<7:0> are stable. Decoding logic identifies column <k> (COL<k> = 1) and row <n> (LX<n> = 1) of the latch, but no load operation occurs until the synchronization signal SYNC is equal to "1". This signal is mainly needed to mask both the delays associated with signals ADD<12:0> and data DQ<7:0> (fly time to reach the decoding logic) and the decoding time itself. The time margin $T_1 - T_0$ must ensure the stability of all the logic signals before enabling the actual data-load by means of signals LY_N<k> and LY<k>.

According to the example shown in Fig. 10.29:

- If DQ<i> = 0, then LY<k> = 0, LY_N<k> = 1, A = 0, B = 1 (dotted line).
- If DQ<i> = 1, then LY<k> = 1, LY_N<k> = 0, A = 1, B = 0 (solid line).

Petitioner Micron Ex-1021, 0295

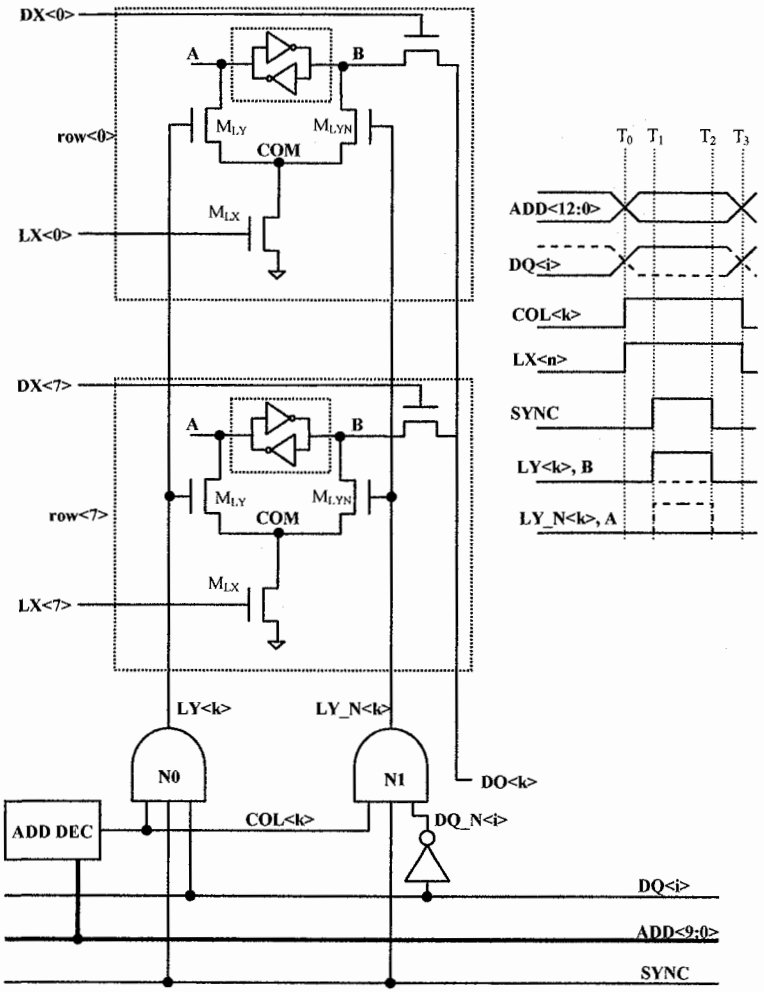


Fig. 10.29. Data-load on column COL<k> (data-load 1)

At time T_2 only signal SYNC is set to ground and at time T_3 a new load cycle starts again, the same way as it happens at time T_0. Cycle time usually lasts some tens of nanosecond. For the sake of completeness, Fig. 10.29 also shows the data-output path, enabled by one of the DX<n> signals. The value of node B of all the latches of row <n> is brought on node DO<k> towards column decoding, not shown in the picture.

Signal lines LX<n> are some millimeters long and they also have a high gate load (8192 M_{LX} transistors), while lines LY<k> and LY_N<k> are relatively short (< 0.5 mm) and their gate load is only 8 M_{LY}. In order to limit power consumption, it is better to switch as less as possible the lines whose capacitive

Petitioner Micron Ex-1021, 0296

load is higher. Therefore, it is advisable to load data in sequence along the rows and not along the columns so that the number of switching events of the LX signals is minimized. For instance, all the latches of row <n> are loaded first, scanning all the 1,024 column one after another, and then the same is done on row <n + 1> and so on.

10.3.2 Data-load 2

In this implementation, the 8 DL<7:0> signals are conditioned by the value of the data DQ<7:0> coming from the pads. All the remaining address signals identify one of the 8,192 columns (see Fig. 10.30) thanks to 8,192 CSEL signals.

In order to increase the data-load speed and to reduce power consumption, the use of differential signals (DL<n> and DL_N<n>) is recommended, as shown in Fig. 10.31 (only the latch of row <n> is depicted).

At time T_0, lines DL<n> and DL_N<n> are equalized and forced to ground enabling the NMOS transistors M_{DEQ} and M_{DIS} by means of signals DEQ and DIS. Once the signals of the decoder are stable, at time T_1 the synchronism signal SYNC0 is set, enabling the three-state of all the latches of column <k> (setting the signal LEN<k>) and the corresponding equalization and forcing to ground of nodes A and B (through signals RST<k> and DIS<k>).

SYNC0 signal also ends the equalization of DL<n> and DL_N<n> and enables their differential load. In the example shown in Fig. 10.31, DQ<n> = 1 and therefore only line DL<n> is loaded (dotted line) through the signal D<n> = 1 which enables the transistor M_D. On the other hand, since D_N<n> = 0, transistor M_{DN} is off and the line DL_N<n> remains at ground (solid line).

At time T_2, signal SYNC1 ends the equalization and the forcing of nodes A and B to ground for all the latches of column <k> and, at the same time, it enables the corresponding MOS M_{DL} and M_{DLN} by setting CSEL<k> = 1.

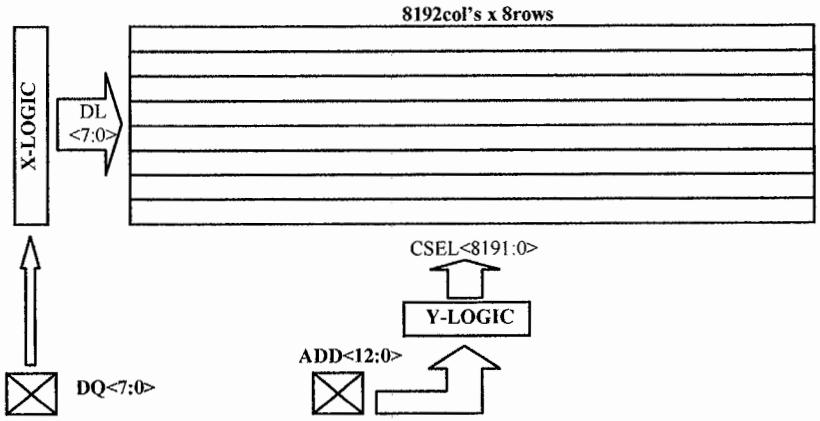


Fig. 10.30. Data-load of a 8 kB page size: second solution (data-load 2)

Petitioner Micron Ex-1021, 0297

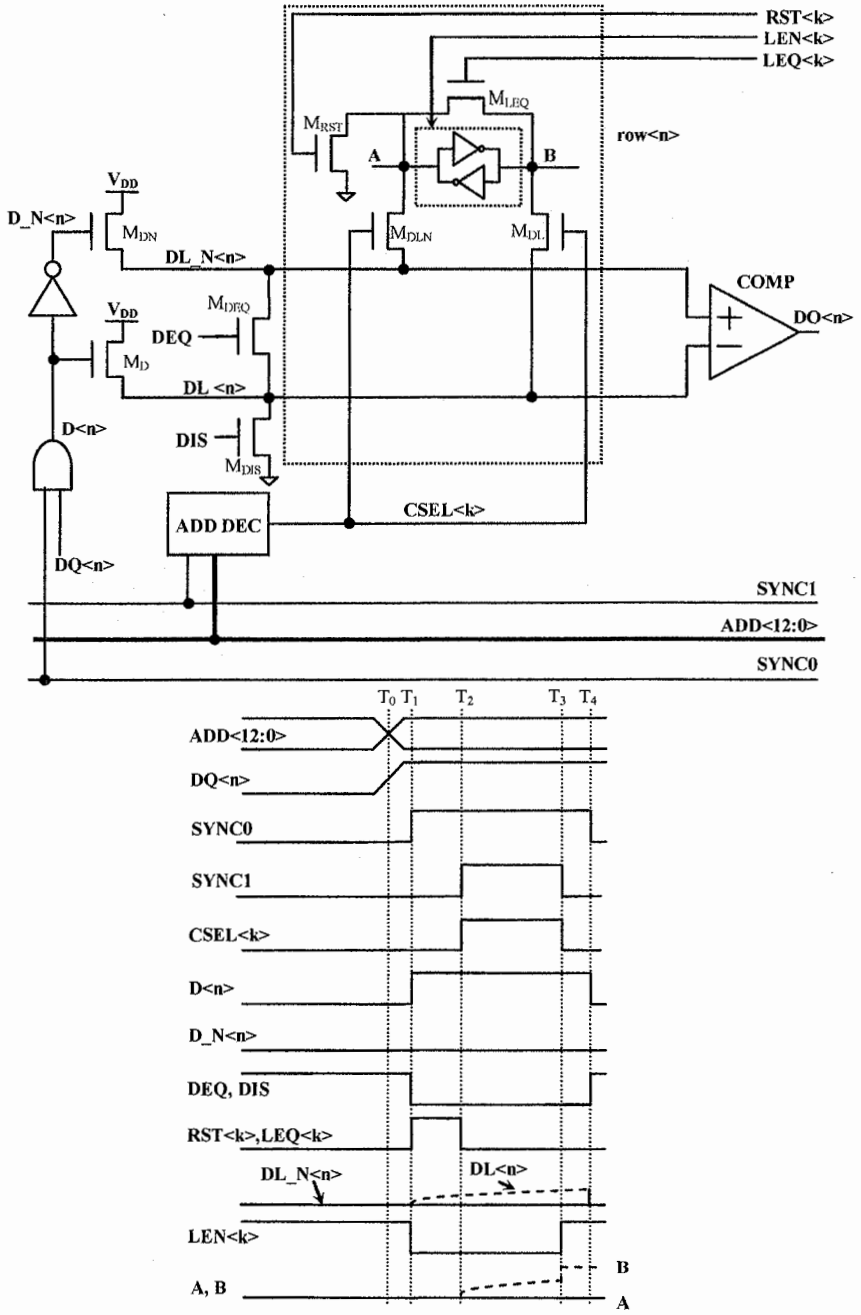


Fig. 10.31. Data-load on column CSEL<k> (data-load 2)

Petitioner Micron Ex-1021, 0298

At this point, nodes A and B are connected to the corresponding lines DL_N <n> and DL <n> and they move accordingly. At time T_3, signal SYNC1 is set to ground, CSEL<k> = 0 and NMOS M_{DL} and M_{DLN} are switched off, thus isolating nodes A and B. At the same time the three-state of the latch is released and the data is regenerated and stored inside the latch. At time T_4, a new load cycle starts, as it happened at T_0, moving on to the next column <k + 1>.

Figure 10.31 also shows the data-output path, which is a differential path enabled by CSEL<k>: at the end of the DL_N<n> and DL<n> lines, a comparator COMP is used to generate the signal DO<n>.

10.3.3 Data-load 3

Similarly to the first example, also in this case the sensing circuits and related latches are arranged as eight blocks of 1,024 rows × eight columns, as shown in Fig. 10.32, and for each data PAD DQ<i> a whole block is allocated. Addresses ADD<12:10> are used to identify one of the eight rows.

In this implementation of the data-load architecture, data DQ<i> influences the signals DX<i> and DX_N<i> which are common to all the latches of a block. Furthermore, the existing column decoding of data-out is used to identify column k. In Fig. 10.32, block DEC enables a single NMOS M_S<k> through the signal S<k> which connects node DO<k> to node DOUT.

As represented in Fig. 10.33, during the data-out phase, the IOCTRL circuit forces node DOUT to ground (enabling M_{DIS}) and handles the data flow DQ<i> (signal DOUTEN disconnects node DOUT from pad DQ<i>). Only two latches of column k are depicted.

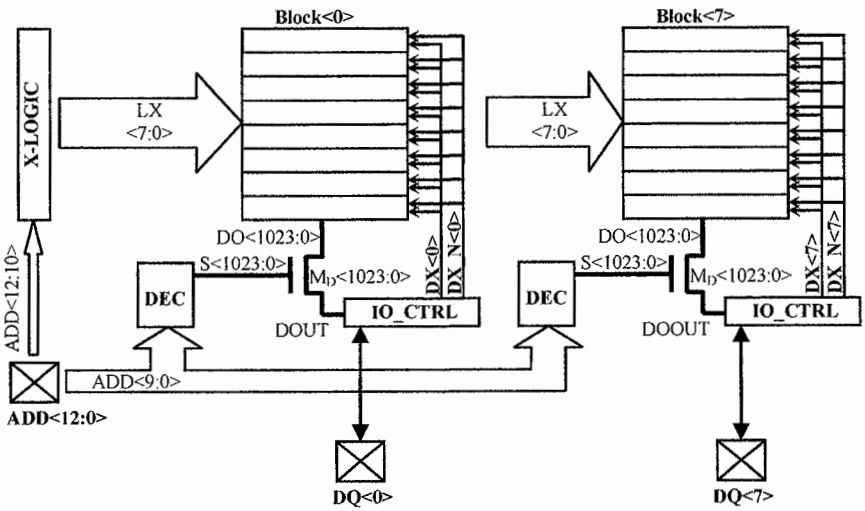


Fig. 10.32. Data-load of a 8 kB page: third solution (data-load 3)

Petitioner Micron Ex-1021, 0299

Data-load occurs only in the latch belonging to column <k> (DO<k> forced to ground) and to row <n> (COM connected to DO<k> through LX<n>). The value of the data depends on the polarity of signals DX_N<k> and DX<k>.

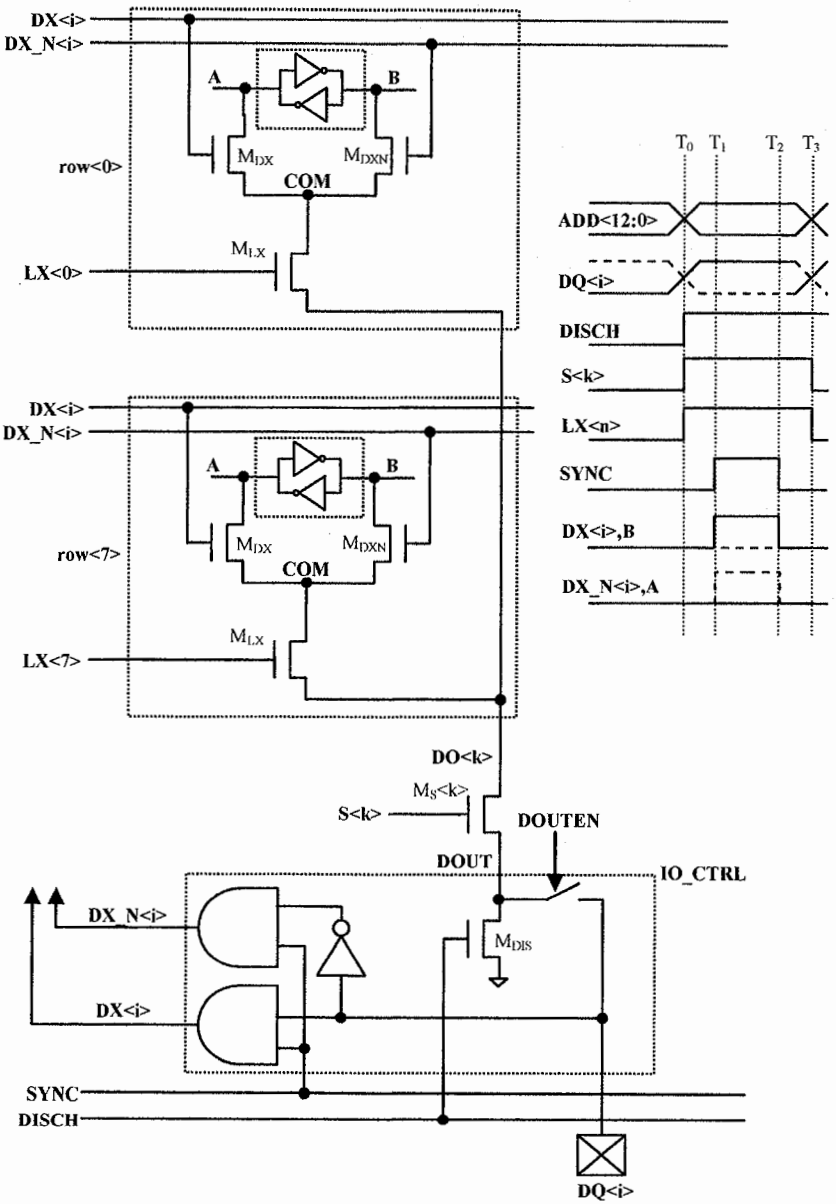


Fig. 10.33. Data-load on DO<k> line (data-load 3)

Petitioner Micron Ex-1021, 0300

At time T_0, column (S<k>) and row (LX<n>) are selected, but it is only at time T_1, thanks again to the signal SYNC, that the toggling of signals DX_N<k> and DX<k> occurs, as shown in the timings of Fig. 10.33:

- If DQ<i> = 0 then DX<k> = 0, DX_N<k> = 1, A = 0, B = 1 (dotted line).
- If DQ<i> = 1 then DX<k> = 1, DX_N<k> = 0, A = 1, B = 0 (solid line).

During data-output operations, signal DATAEN connects node DOUT to pad DQ<i>; furthermore, the forcing to ground is disabled by setting DISCH = 0. By properly controlling signals LX<n>, DX<i> and S<k>, it is possible to output all nodes A of all the latches of a block <i> on DOUT, one at a time.

10.4 Moving read voltages

With the continuous shrink of the NAND technology, retention (Sect. 4.3.3) is becoming more and more important, especially for multi-bit per cell storage. Retention is defined as the ability of the memory cell to retain its information over time; in this case, cell's terminals are not biased. Root cause of the retention issues is the charge loss from the floating gate, which is particularly sensitive to the storage temperature and to the number of program/erase cycles. As a result, V_{THR} distributions shift towards lower values as shown in Fig. 10.34a. Distribution D3 is the most impacted by the retention shift.

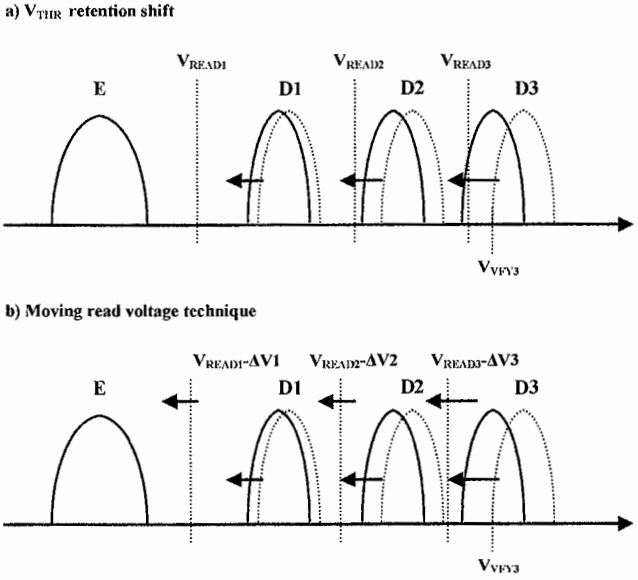


Fig. 10.34. (a) V_{THR} retention shift, (b) moving read voltages technique

Petitioner Micron Ex-1021, 0301

Usually, read voltages are fixed and, therefore, retention reduces read margins. In order to cope with retention, a moving read voltages technique has been proposed [12]. The basic concept is described in Fig. 10.34b: read voltages V_{READ1}, V_{READ2} and V_{READ3} are trimmed on the fly (through $\Delta V1$, $\Delta V2$ and $\Delta V3$, respectively) to compensate the V_{THR} shift.

Program and read algorithms manage together the above mentioned read voltages adjustment. During the program operation, the embedded logic unit counts the number NP_{D3} of cells to be placed in D3: each logic page has extra cells to store NP_{D3}, as shown in Fig. 10.35. Every following read operation at V_{READ3} gives out the actual number NR_{D3} of cells belonging to D3. The internal microcontroller compares NP_{D3} and NR_{D3}: if $NR_{D3} < NP_{D3}$, then depending on the difference, a proper set of $\Delta V1$, $\Delta V2$ and $\Delta V3$ is selected and a new read is performed. This algorithm monitors only distribution D3 as the related V_{THR} retention shift is supposed to be the highest. Data integrity of NP_{D3} is guaranteed by ECC (Chap. 14).

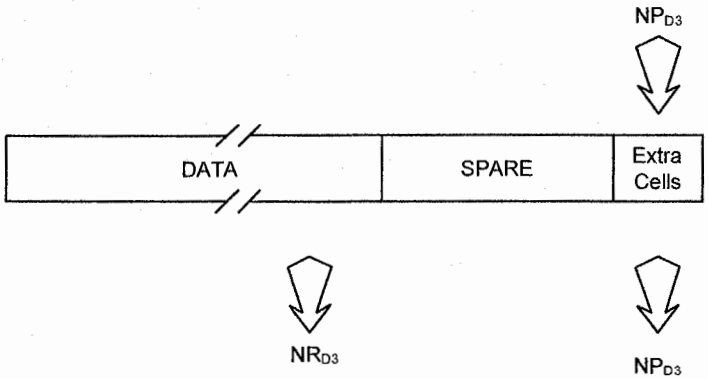


Fig. 10.35. Each page has extra cells to store the number NP_{D3} of cells that should be on distribution D3 after reading

References

1. Seungjae Lee et al. *A 3.3V 4Gb Four-Level NAND Flash Memory with 90nm CMOS Technology* IEEE International Solid-State Circuits Conference, ISSCC, Digest of Technical Papers, Feb. 2004, pp. 52–53, 513, Vol. 1.
2. Dae-Seok Byeon et al. *An 8Gb Multi-Level NAND Flash Memory with 63nm STI CMOS Process Technology* Solid-State Circuits Conference, ISSCC, Digest of Technical Papers, Feb. 2005, pp. 46–47, Vol. 1.
3. Raul-Adrian Cernea et al. *A 34 MB/s MLC Write Throughput 16 Gb NAND With All Bit Line Architecture on 56 nm Technology* IEEE Journal of Solid-State Circuits, Vol. 44, No. 1, Jan. 2009, pp. 186–194.
4. Rino Micheloni et al. U.S. Patent No. 7366014 - *Double page programming system and method* Assignee: STMicroelectronics S.r.l. (Agrate Brianza (MI), IT).

Petitioner Micron Ex-1021, 0302

5. Noboru Shibata et al. U.S. Patent No. 7443724 – *Semiconductor memory device for storing multivalued data* Assignee: Kabushiki Kaisha Toshiba (Tokyo, JP).
6. Ken Takeuchi, Tomoharu Tanaka, and Toru Tanzawa *A Multipage Cell Architecture for High-Speed Programming Multilevel NAND Flash Memories* IEEE Journal of Solid-State Circuits, Vol. 33, No. 8, Aug. 1998, pp. 1228–1238.
7. Tac Sung Junget al. "*A 117mm^2 Only 128-Mb Multilevel NAND FLASH Memory for Mass Storage Applications*" IEEE Journal of Solid-State Circuits, Vol. 31, No. 11, Nov. 1996, pp. 1575–1583.
8. Tomoharu Tanaka et al. U.S. Patent No. 6545909 – *Nonvolatile semiconductor memory device* Assignee: Kabushiki Kaisha Toshiba (Kawasaki, JP).
9. Luca Crippa, Rino Micheloni et al. U.S. Patent No. 7336538 – *Page buffer circuit and method for multi-level NAND programmable memories*: STMicroelectronics S.r.l. (Agrate Brianza (MI), IT).
10. Tomoharu Tanaka et al. U.S. Patent No. 6643188 – *Non-volatile semiconductor memory device adapted to store a multivalued data in a single memory cell* Assignee: Kabushiki Kaisha Toshiba (Minato-ku, JP).
11. Shih-Chung Lee et al. U.S. Patent No. 7508715 – *Coarse/fine program verification in non-volatile memory using different reference levels for improved sensing* Assignee: SanDisk Corporation (Milpitas, CA).
12. Changhyuk Lee et al. *A 32Gb MLC NAND-Flash Memory with Vth – Endurance – Enhancing Schemes in 32 nm CMOS* IEEE International Solid-State Circuits Conference, ISSCC, Digest of Technical Papers, Feb. 2010, pp. 446–447.

Petitioner Micron Ex-1021, 0303

11 Charge pumps, voltage regulators and HV switches

R. Micheloni[1] *and L. Crippa*[2]

Modifying or reading the number of electrons stored into the floating gate requires a big set of voltages. The high voltage (HV) system has to provide all these voltages with the desired precision, timing and granularity. On top of that, many voltages have a value greater than the NAND power supply VDD, asking for an on-chip charge pump. This chapter deals with the HV basic building blocks.

11.1 Charge pumps

In the NAND environment, one of the most used type of charge pumps is the voltage doubler [1]. The basic stage is shown in Fig. 11.1. It is a feedback system that can duplicate the input voltage and, essentially, it is made up by two n-channel transistors (MN1, MN2), two p-channel transistors (MP1, MP2) and two capacitors (C1, C2) of the same size.

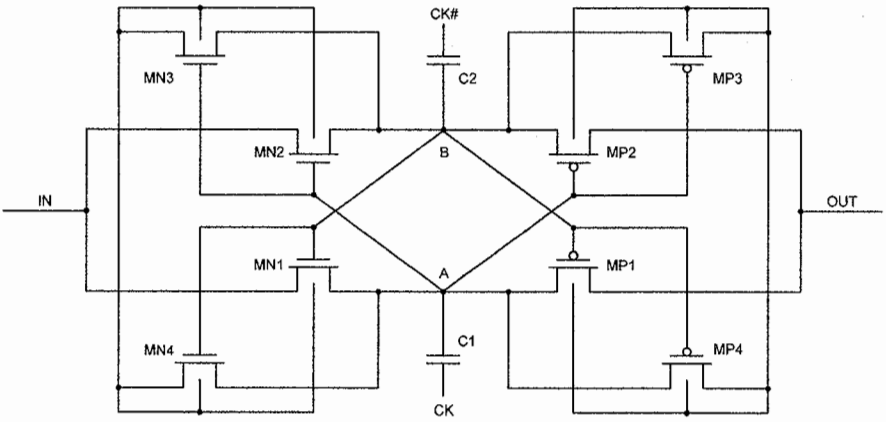


Fig. 11.1. Basic stage of a voltage doubler

[1] Integrated Device Technology, rino.micheloni@ieee.org
[2] Forward Insights, luca.crippa@ieee.org

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_11,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0304

In order to understand the principle of operation of this circuit, it can be assumed that, at the beginning, nodes A and B, as well as CK (pump clock) and its complement (CK#), are at GND. In this way, both transistors MN1 and MN2 are off. Voltage on the node IN (V_{IN}) is set to VDD (i.e. the chip power supply).

As soon as CK toggles from GND to VDD, V_A becomes VDD, activating transistor MN2. Since CK# remains at GND, the charge starts flowing from power supply to capacitor C2 until V_B reaches a value equal to VDD − $V_{TH,MN2}$. When CK goes to GND, transistor MN2 turns off. At the same time, CK# gets to VDD and, therefore, V_B becomes (VDD − $V_{TH,MN2}$ + VDD), turning on transistor MN1. As a result, C1 is charged up to VDD. Of course, when CK# goes to GND again, V_B is, in principle, equal to VDD − $V_{TH,MN2}$. Since the signal CK is used as a clock, each capacitance is continuously charged and discharged between VDD and 2·VDD. In other words, during each period of the clock either V_A or V_B is at 2·VDD.

At this point, in order to build a real charge pump, voltages on nodes A and B have to be transferred to the next pump stage. Now MP1 and MP2 come into the game. When CK is at VDD, V_A is 2·VDD and V_B is VDD. Transistor MN1 is, therefore, turned off while MP1 is active, transferring the voltage of node A to node OUT. In the meanwhile MP2 is off, MN2 is on and the capacitor C2 is charged up. When CK goes back to GND and CK# becomes VDD, then the circuit behaves in the opposite way: MN1 and MP2 are active (the former charges capacitor C1, the latter transfers the voltage of node B to the output) while MN2 and MP1 are turned off. It is worth to note that no active direct paths between IN and OUT are allowed: these paths would result in a loss of charge and, therefore, in a reduced output voltage.

As usual, when designing a charge pump, one issue to cope with is the biasing of the transistor body terminals. The easiest solution is to connect the body of the n-channel transistor to the power supply and the body of the p-channel transistor to the output node.

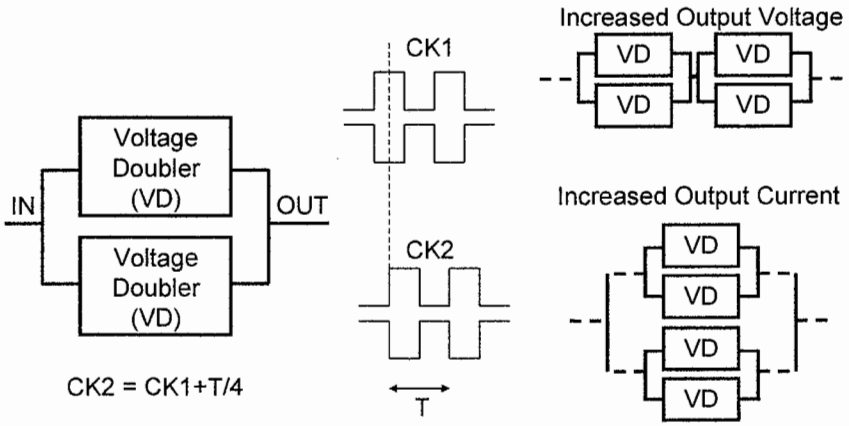


Fig. 11.2. Charge pump as a cascade of basic voltage doubler stages

Petitioner Micron Ex-1021, 0305

The drawback of this solution is that the output voltage is considerably reduced by the body-effect of the transistors itself. In Fig. 11.1 a "dynamic biasing" has been chosen: bodies are continuously switched between V_A and V_B. As a result, the body of the NMOS transistors is always kept at the lowest voltage (through MN3 and MN4) while the body of the PMOS transistors is always at the highest voltage (through MP3 and MP4).

The basic stage of Fig. 11.1 can be used to build up more complex structures as depicted in Fig. 11.2. Usually, two stages are used in parallel in order to decrease the ripple of the output voltage. In fact, due to the internal switching activity of the capacitors, the output of the pump can be more or less noisy. When talking about ripple, we generally refer to the height of the "peaks" that can be found in the output node waveform.

In order to properly control the output voltage, voltage doubler stages are inserted in a feedback loop as described in Fig. 11.3. A block called "Hireg" is used to limit the output voltage. Thanks to a resistive divider (it could also be made by CMOS diodes), the output voltage is compared with V_{REF} (usually a band-gap reference voltage). CK drivers are then enabled/disabled depending on the comparison result.

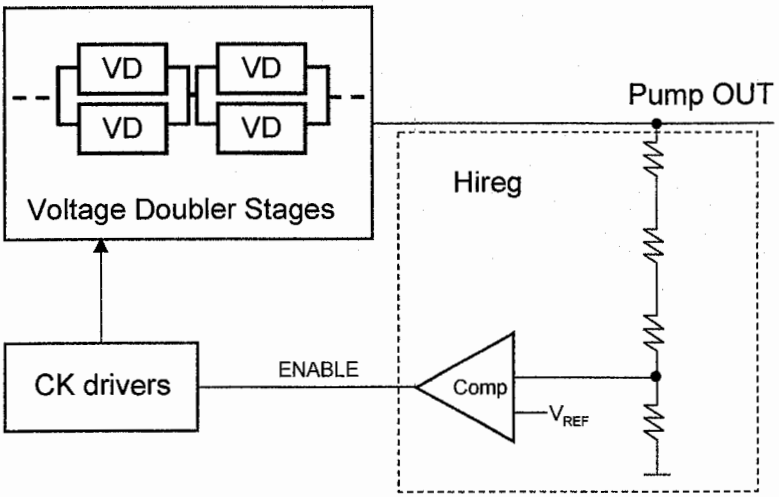


Fig. 11.3. Charge pump architecture

Compared with a Dickson charge pump [1], the voltage doubler has several advantages:

- The possibility of driving the capacitances using only two complemented phases. This implies a relatively simple phase generator circuit with respect to the one used in the Dickson pumps.
- The possibility of using only "low voltage" transistors (i.e. transistors that can sustain a voltage difference at their terminal lower than power supply) since the voltage difference at the terminals of each transistor does not go above VDD in any operating phase of the circuit.

Petitioner Micron Ex-1021, 0306

- No boost capacitors are required for the pass transistors (since we can use the one implemented for the charge transfer), thus achieving a reduction of the area occupation and limiting the dissipated power.
- Reduced ripple of the output voltage.

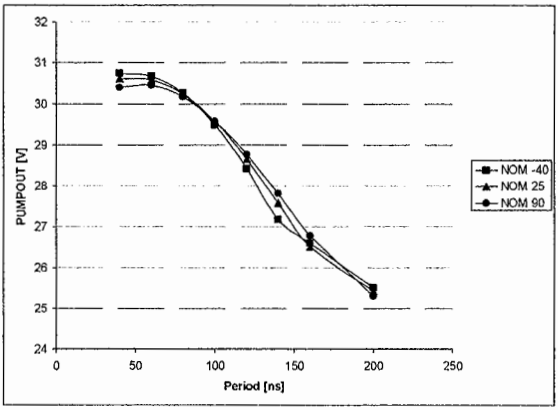


Fig. 11.4. Charge pump output voltage as a function of the clock period

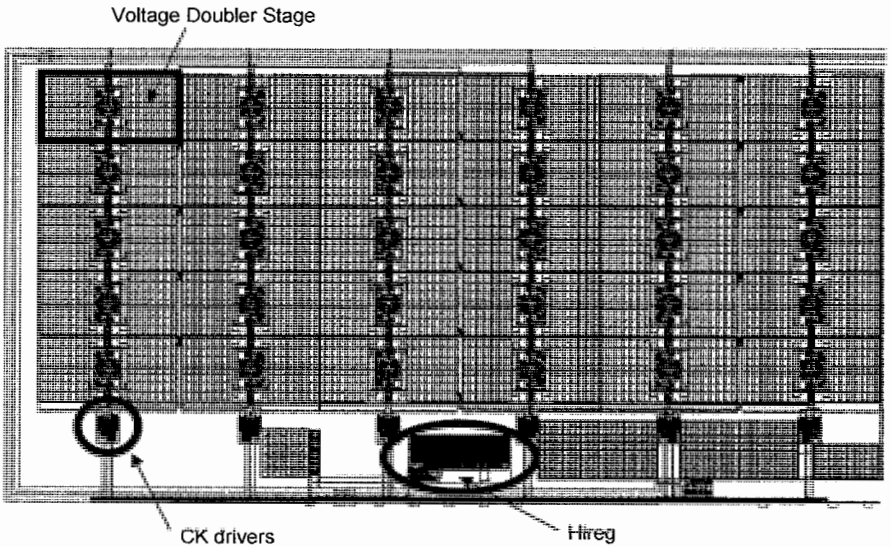


Fig. 11.5. Layout of a voltage doubler-based charge pump

In order to find the best configuration, the output voltage of the charge pump is measured varying the CK period, as it is shown in Fig. 11.4 where it is assumed a DC current load of 150 μA. Optimum CK period is usually in the range of

Petitioner Micron Ex-1021, 0307

60–80 ns considering an output resistance of around 10 kΩ. The voltage doubler pump can easily achieve voltages above 25 V starting from the chip VDD of 2.5 V. Power efficiency η_P can be as high as 20–30% if the current load remains in the range of few hundreds microAmpere.

$$\eta_P = \frac{V_{OUT} \cdot I_{OUT}}{V_{IN} \cdot I_{IN}} \tag{11.1}$$

From Fig. 11.4 it is clear that the clock period has to be as short as possible. Faster clock means bigger area of the CK drivers. The right trade-off has to be found considering that, in most of the NAND applications, silicon cost is the main driver.

The modular structure of the voltage doubler described above results in a compact layout of the whole charge pump as sketched in Fig. 11.5 where the main blocks are highlighted.

11.2 Read regulator

In a 2 bit-per-cell multilevel Flash NAND memory, four different threshold voltage (V_{TH}) distributions exist, as shown in Fig. 11.6. All the cells are in the 11 state after electrical erase. During programming phase, the threshold voltage of the cells is incremented in small steps until the desired value is reached. At the end of each program step, a verify operation is performed, in order to evaluate whether V_{TH} has gone above one of the verify voltages, V_{FY1}, V_{FY2} or V_{FY3}. Of course, verify voltage depends on which bits have to be stored in a given cell. For instance, in order to reach "00" logic value, threshold voltage has to go above V_{FY2}. Once target distribution is reached, further program pulses are not applied to that cell.

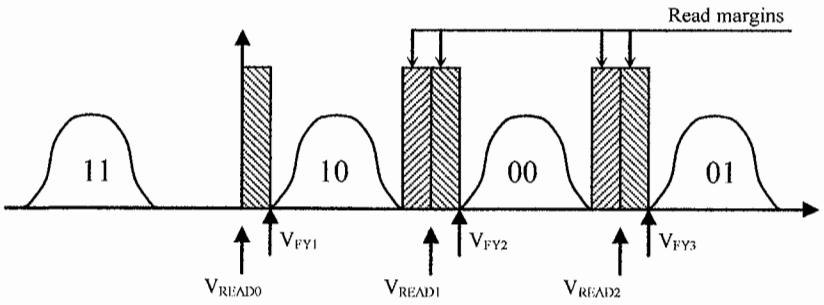


Fig. 11.6. Cell V_{TH} distributions in a 2bit/cell NAND memory

Petitioner Micron Ex-1021, 0308

In order to univocally determine the logic value stored in the selected cell, read operation uses three voltage values, V_{READ0}, V_{READ1}, and V_{READ2} as shown in Fig.11.6. Each read voltage is centered between two adjacent distributions so that read margins are maximized. For instance, the distance between V_{READ1} and the rightmost side of 10 distribution should be equal to the distance of the leftmost side of 00 distribution. With multilevel memories, the typical value for such distances is 300 mV.

In order to achieve the required precision, voltages to be applied to the cells are generated by means of voltage regulators which exploit band-gap techniques to generate a precise reference voltage. In this way, the voltages generated on-chip are independent from temperature, at least to a first approximation. On the other hand, the V_{TH} distributions of the memory cells are highly sensitive to temperature variations: as temperature increases, V_{TH} decreases and vice versa (see Fig. 11.7).

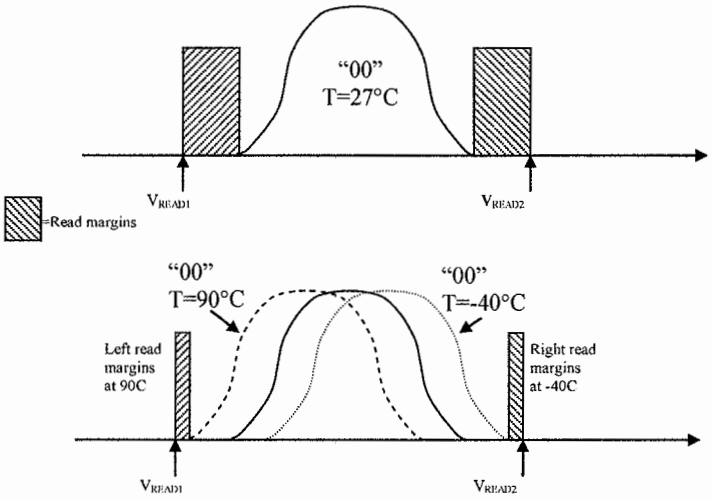


Fig. 11.7. V_{TH} variations with temperature

As a result, read margins are reduced when temperature varies, because the tails of the distributions get nearer and nearer to read voltages. For instance, as shown in Fig. 11.7, 00 distribution gets nearer to V_{READ2} at low temperature, while it gets nearer to V_{READ1} at high temperature. The same is true for each distribution. Threshold voltage of the cell typically shifts of −1.5 mV/°C. As a consequence, overall variation is approximately 200 mV if a temperature range of −40°C to 90°C is considered.

Therefore, a specific type of read voltage regulator is needed [2–4]: that is, the thermal coefficient of its output voltage has to be as similar as possible to the coefficient of the cell's V_{TH}. In this way, read voltages rigidly shift with distributions, keeping the margins unaltered. (Fig 11.8). A similar constraint is true for verify voltages.

Petitioner Micron Ex-1021, 0309

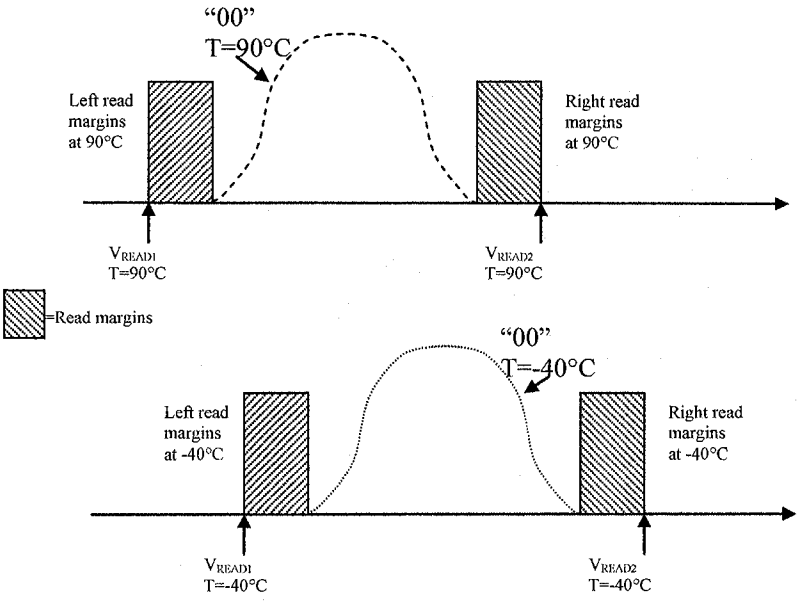


Fig. 11.8. V_{READ} tracking of V_{TH} variations with temperature

NAND technological process generally utilizes tunnel oxide to build low-voltage transistors. The idea is, therefore, to exploit the temperature profile of the threshold voltage of a NMOS transistor (V_{THN}). A voltage generator whose output is independent from temperature could be devised, to which the V_{GS} of the NMOS biased at low current is summed, but such a solution would have some limitations:

- V_{GS} is equal to 0.7 V at 27°C, while V_{FY1} is typically 0.3 V. A simple voltage adder would therefore call for the generation of a negative voltage of −0.4 V.
- It is not possible to make a partition of the V_{GS}, because the thermal coefficient must be the same as the coefficient of V_{THN}.
- Subtraction of V_{GS} from a temperature-independent voltage cannot be achieved as well, since resulting thermal coefficient would be +1.5 mV/°C.

A possible solution to the problem is shown in Fig. 11. 9. V_{READIN} voltage is generated by a voltage regulator whose input is the band-gap voltage V_{BG} (which is temperature-independent). V_{READIN} can result in different values, depending on the selected switch (SW1-SW5). V'_{GT} voltage is equal to V_{GT}, since it is derived through a buffer. V_{GT} is equal to $V_{GATE} - V_{GS}$, where V_{GS} is the gate–source voltage of NMOS M0. This transistor is biased with a current of few microAmperes which is low enough to let V_{GS} be almost the same as its threshold voltage. By means of a proper selection of RSOURCE, it is easy to bias this transistor as needed. In order to get such a small current, the value of RSOURCE shall be in the order of hundreds of kiloOhms. V_{GATE} is temperature-independent as well.

Petitioner Micron Ex-1021, 0310

Resulting equations are as follows:

$$V_{READ} = V_{READIN} + \left(V_{READIN} - V'_{GT}\right) \cdot \frac{R_1}{R_2} \tag{11.2}$$

$$V'_{GT} = V_{GT} = V_{GATE} - V_{GS} \tag{11.3}$$

By combining Eqs. (11.2) and (11.3) and assuming that $R_1 = R_2$:

$$V_{READ} = 2 \cdot V_{READIN} - V_{GATE} + V_{GS} \tag{11.4}$$

It is worth noting that inside Eq. (11.4) the only temperature-dependent term is V_{GS}. The assumption of $R_1 = R_2$ is needed in order to allow a correct tracking of the temperature coefficient of the threshold voltage of the cell.

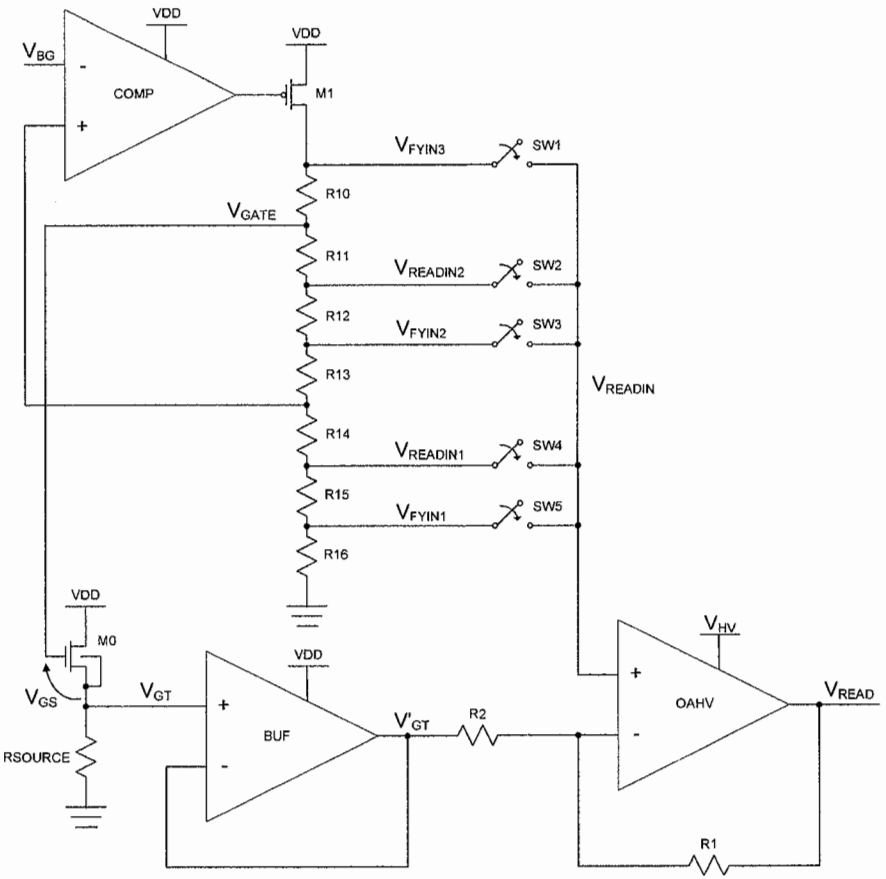


Fig. 11.9. Temperature compensated read regulator

Petitioner Micron Ex-1021, 0311

Next thing to do is to choose the proper values for V_{READIN} and V_{GATE} to generate the required V_{READ} values. Typical value for V_{FY1} is 0.3 V while V_{FY3} is usually in the order of 3 V.

Assuming V_{GATE} = 1.8 V and a threshold voltage of M0 at 27°C equal to 0.7 V, an output value of 0.3 V requires, according to Eq. (11.4), a V_{READIN} equal to 0.7 V. Similarly, an output value of 3 V requires a V_{READIN} equal to 2.05 V. Every intermediate value required for either read or verify obviously need a V_{READIN} value between 0.7 and 2.05 V.

This example highlights another important topic: assuming a device VDD between 2.7 and 3.6 V, V_{READIN}, V_{GATE} and V'_{GT} voltages can be generated without using charge pumps. A charge pump is only needed to provide supply to the operational amplifier OAHV (shown as V_{HV} in Fig. 11.9).

Figures 11.10 and 11.11 show the result of some simulations of the described circuit designed with a 51 nm NAND process. The threshold voltage of the NMOS is equal to 0.45 V at 27°C. In Fig. 11.10 V_{READ} is plotted against time at different V_{READIN} values.

Applying the V_{READ} shown in Fig. 11.10 to the gate of the written cell, the characteristic of the NAND string becomes the one shown in Fig. 11.11b, while Fig. 11.11a shows the graph in case the temperature compensation is not applied. It is worth noting that the characteristics get tighter around the characteristic at 27°C. Thanks to the compensation, threshold voltage spread decreases from 200 mV to less than 20 mV!

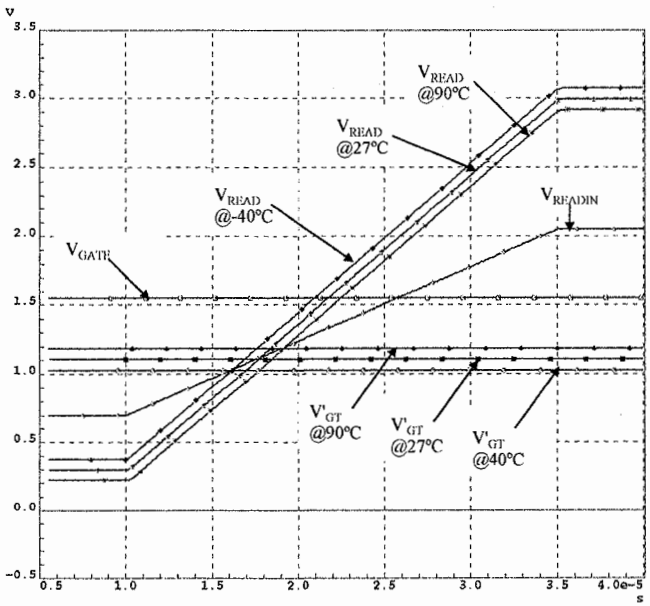


Fig. 11.10. Simulation of V_{READ} variations

Petitioner Micron Ex-1021, 0312

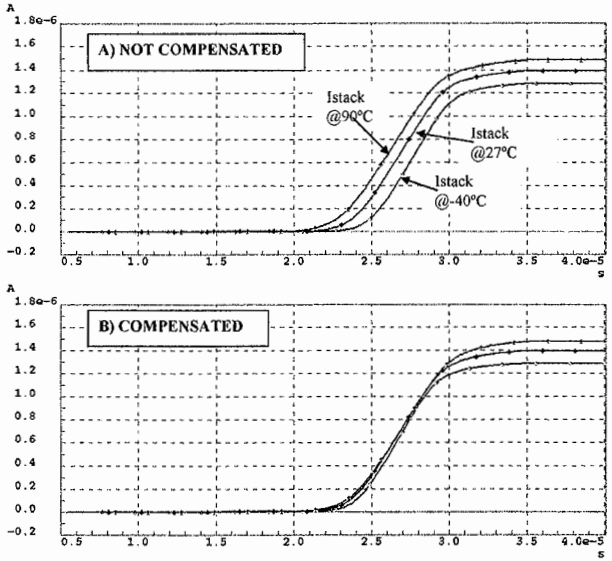


Fig. 11.11. NAND string current without (a) and with (b) compensation of the read voltage

11.3 Double-supply voltage regulator

Both program and erase operations require voltages higher than VDD. For instance, the programming staircase voltage starts at 14–15 V and arrives at 25 V and beyond. High voltages are generated by a charge pump and filtered by a proper voltage regulator: in this way it is possible to reduce the ripple and obtain the desired output voltage value.

In 1 bit/cell Flash memories, voltage regulator is omitted and the output voltage of the pump is directly used, regulated by means of an on-off type of control. Typical ripple values are in the order of 1–2 V. In case of multilevel memories, the target voltage precision cannot be achieved without a voltage regulator.

NAND technology does not usually provide PMOS HV (High Voltage) transistor, therefore it is not possible to implement traditional voltage regulators like the one shown in Fig. 11.12. In fact, the use of a low-voltage transistor for M_{POUT} would mean that the voltage drop across its terminals must be guaranteed not to exceed 4–5 V. This must be true both in static and in transient conditions. On top of that, all the required values for the staircase program pulse must be generated out of the pump output voltage (~30 V), beginning at 15 V: that is, M_{POUT} must be a HV transistor.

In order to solve the issue it is possible to design a voltage regulator [5] whose first differential stage is supplied by VDD, while the second one is supplied by a charge pump so that the HV value can be provided at the output (Fig. 11.13).

Petitioner Micron Ex-1021, 0313

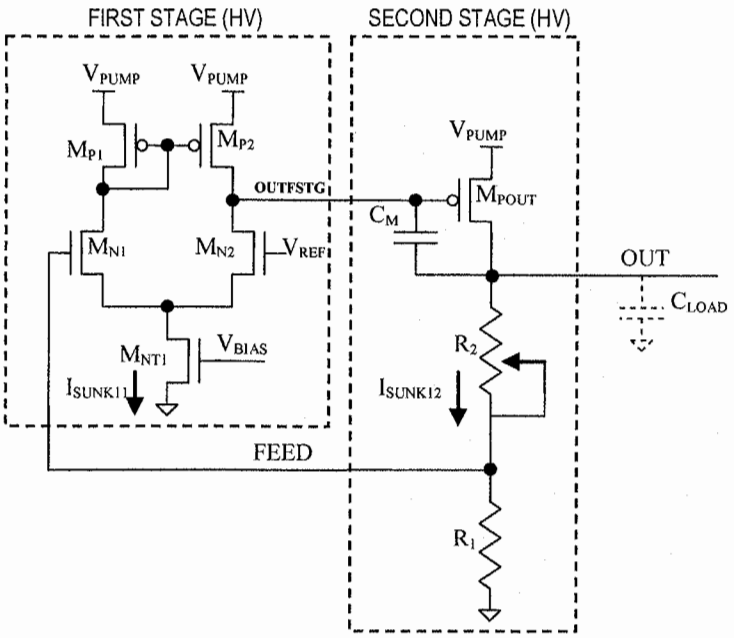


Fig. 11.12. Voltage regulator with high voltage PMOS

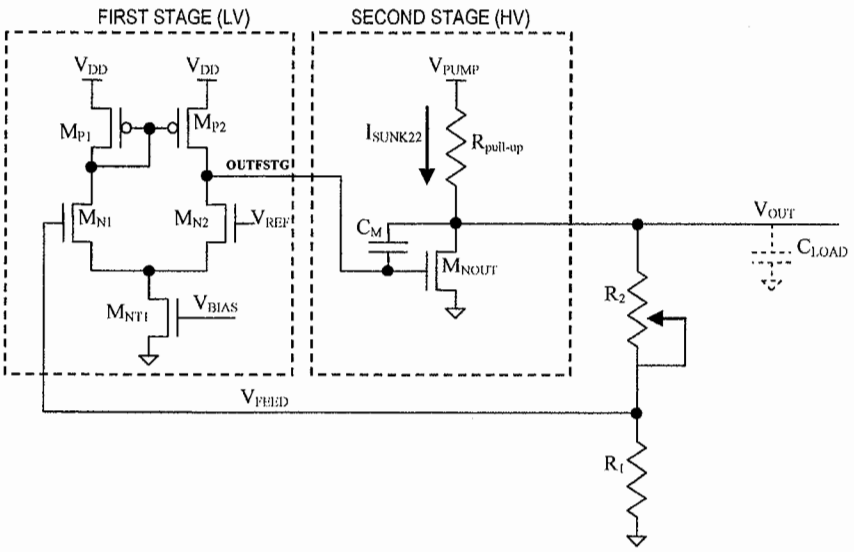


Fig. 11.13. Double-supply voltage regulator

Petitioner Micron Ex-1021, 0314

By supplying the first stage with VDD, PMOS LV transistors can be used to realize the current mirror ($M_{P1} - M_{P2}$). The second stage is instead designed using an NMOS HV (M_{NOUT}) together with a resistive pull-up ($R_{pull\text{-}up}$).

One of the main drawbacks of this structure is the current consumption in case of high capacitive loads (C_{LOAD}). The charge time T_{RISE} of C_{LOAD} can be approximated by:

$$T_{RISE} = \alpha \cdot C_{LOAD} \cdot R_{pull-up} \tag{11.5}$$

where α lies between 3 and 5 depending on the needed precision. Considering a load in the order of 100 pF, $\alpha = 3$ and $T_{RISE} = 1$ µs, $R_{pull\text{-}up}$ is approximately 3 kΩ.

Assuming that $V_{OUT} = 10$ V and $V_{PUMP} = 24$ V, static current sunk from the pump I_{SUNK22} is equal to:

$$I_{SUNK\,22} = \frac{(24-10)V}{3K\Omega} \cong 5mA \tag{11.6}$$

In order to make the current consumption independent from the load, the circuit of Fig. 11.13 can be modified as shown in Fig. 11.14, i.e. adding a follower stage which acts as a current buffer.

At the end of the transient, the consumption of the branch which includes the follower M_{NFOLL} is equal to:

$$I_{SUNK3F} = I_{SUNK12} = \frac{V_{OUT}}{R_1 + R_2} \tag{11.7}$$

i.e. it is the same as the one featured in the circuit of Fig. 11.12.

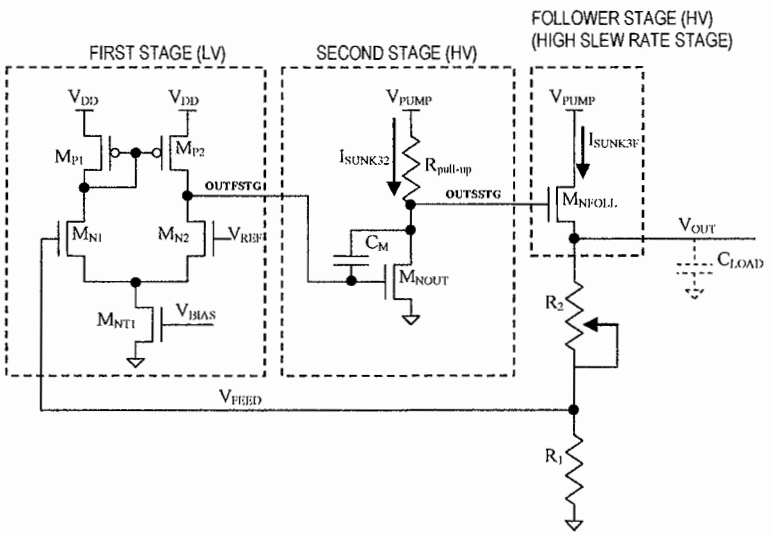


Fig. 11.14. Double-supply voltage regulator with follower stage

Petitioner Micron Ex-1021, 0315

At this point $R_{pull\text{-}up}$ is completely independent from the load and it can be sized properly to reduce the current consumption from the pump. Such resistance must be anyway able to charge the gate of M_{NFOLL} in a reasonable amount of time. The main difference with the circuit shown in Fig. 11.13 is that the parasitic load represented by the gate of the follower is many orders of magnitude smaller than the capacitive load of 100 pF. It is therefore possible to size $R_{pull\text{-}up}$ up to values in the order of a hundred of kiloOhm. For instance, with a pull-up resistance of 200 kΩ, assuming a voltage drop between gate and source of M_{NFOLL} of about 1 V (threshold voltage) and V_{OUT} = 10 V, the current sunk from the pump is equal to:

$$I_{SUNK32} = \frac{(24-11)V}{200K\Omega} \cong 65\mu A \tag{11.8}$$

Such a current consumption is comparable to the current consumption of the first stage of the circuit in Fig. 11.12. Of course, I_{SUNK32} decreases if the output voltage is higher than 10 V.

The circuit shown in Fig. 11.14 can be improved in terms of maximum output voltage by using triple well transistors (in order to eliminate the body effect on the follower) and low-V_{TH} transistors (in order to reduce the voltage drop between gate and source due to the threshold voltage of the follower).

Using the circuits shown above, it is possible to design the regulator which generates the voltage to be applied to the gate of the cells during program operations.

In order to perform effective and precise program operations, the best method for biasing the wordlines is with a linear "staircase" voltage; in such a way, a 1:1 relation exists between gate voltage increment, ΔV_P, and voltage shift after each program pulse, ΔV_T.

To obtain "programmable" gate voltage values, we can split the feedback resistive divider into a series of smaller resistances, each one having a switch in parallel (Fig. 11.15). When adopting this solution, two major problems have to be taken into account: the first concerns the differential linearity of the output voltage as it impacts the V_{TH} distribution widths of the memory cells.

The second problem concerns the charge injection related to the commutation of the switches, which leads to voltage overshoot on the feedback node (FEED). In order to have a fast settling time on the program voltage, these overshoots have to be reduced as much as possible.

In Fig. 11.15a, where R_2 is made programmable by means of a series of binary weighted resistors, each one with a selection switch in parallel; the number of switches simultaneously open depends on the selected program voltage level, giving arise to serious linearity problems (as much as ±15% of the voltage step). Furthermore, when many switches commute simultaneously, the node FEED is subject to high charge injection, especially if the switches are designed with high aspect ratio to minimize their parasitic resistance. From the linearity point of view, a better solution is sketched in Fig. 11.15b where R_2 is split in a series of w equal resistors of value ΔR. In both solutions, R_1 is kept constant.

Petitioner Micron Ex-1021, 0316

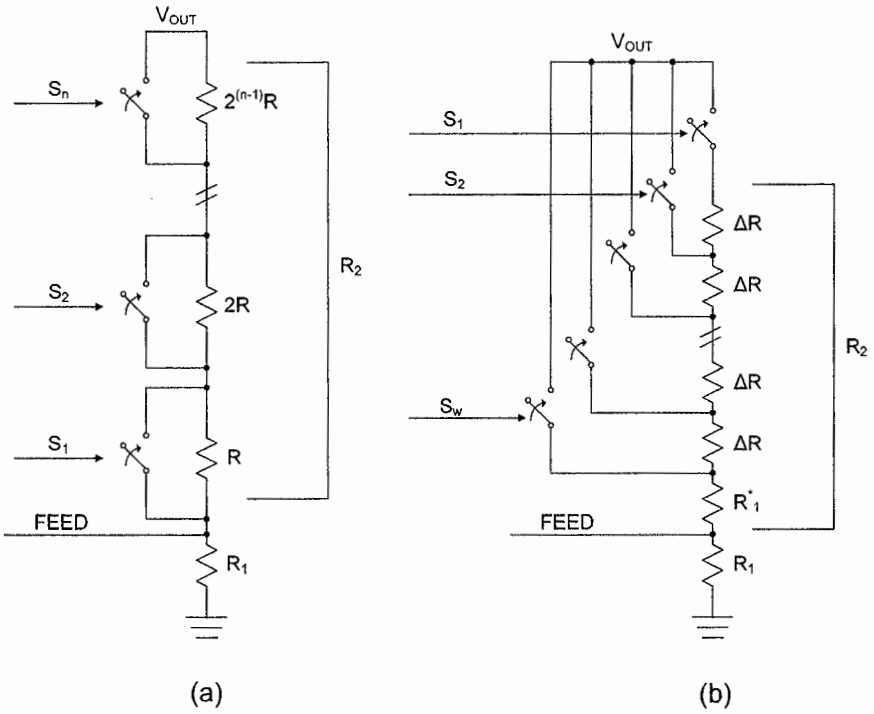


Fig. 11.15. Two ways of implementing a programmable resistive feedback divider

The voltage step ΔV_P is achieved at each staircase step by increasing the value of R_2 by a fixed amount ΔR; at the p-th program step the gate program voltage V_P is equal to:

$$V_P = V_{BG}\left(1+\frac{R^*_2 + p\Delta R}{R_1}\right) \tag{11.9}$$

where R^*_2 is the value that R_2 assumes at the first program step. The program step voltage ΔV_P is:

$$\Delta V_P = V_{BG}\frac{\Delta R}{R_1} \tag{11.10}$$

At each p-th program pulse, a suitable value of resistor ΔR is shorted by means of the control logic that produces the w signals to drive the switches from S_1 to S_w, thus achieving $R_2 = R^*_2 + p\Delta R$. To avoid introducing parasitic resistances that may vary from one step to the next, only one switch is active during each program pulse, while the others are off.

Figure 11.16 shows voltage V_{OUT} at the output of the program regulator. It is worth noting that the output voltage of the pump V_{PUMP} changes (through the high regulator of the pump itself) in order to minimize the stress of the HV transistors.

Petitioner Micron Ex-1021, 0317

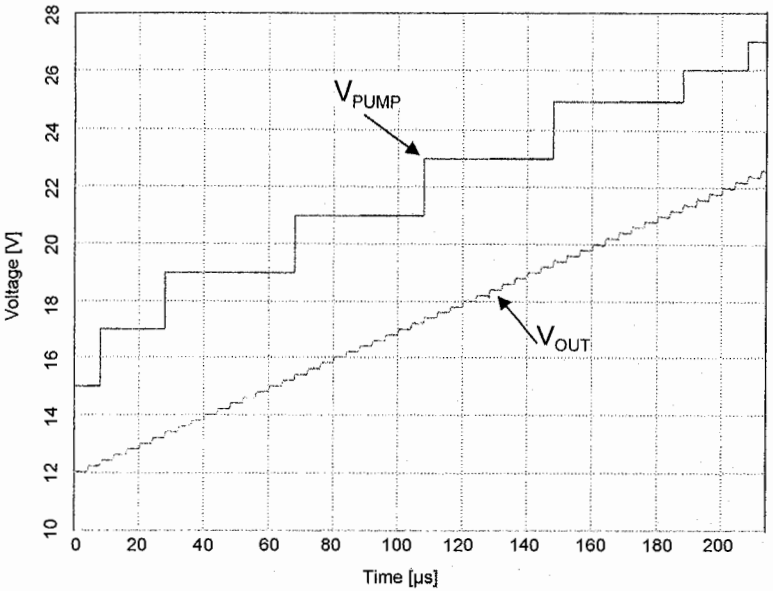


Fig. 11.16. Staircase voltage for the program operation

The concept of separated power supplies can be extended to other circuits as well. An example can be found in Fig. 11.17 where a circuit for the controlled current discharge of a C_{LOAD} capacitor is shown. For instance, this type of circuits are used inside the Flash to discharge the isolated p-well of the matrix at the end of an erase operation (p-well is at 22 V).

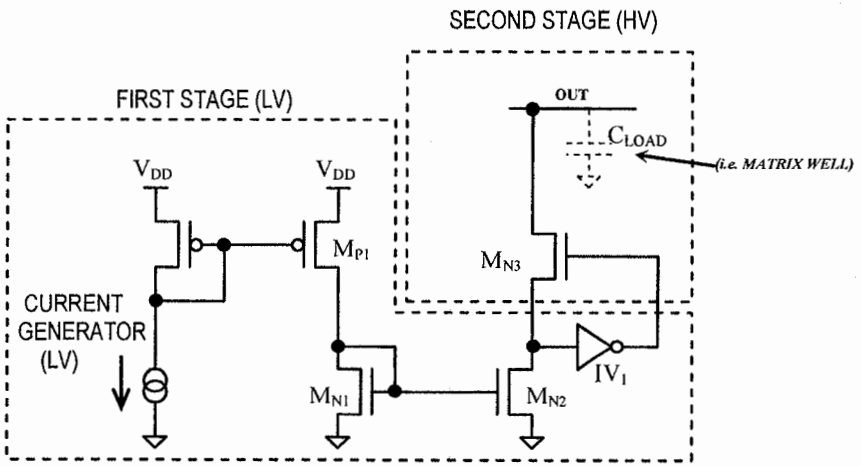


Fig. 11.17. Current-controlled discharge circuit for high capacitive loads

Petitioner Micron Ex-1021, 0318

LV section is composed of a LV current generator, a PMOS LV current mirror and another NMOS LV mirror.

Since the drain of MOS M_{N2} cannot be directly connected to the high voltage node, a standard drain limiter stage is introduced, composed of the inverter IV_1 and the MOS HV M_{N3}. In this way the high voltage node OUT is decoupled from the drain of M_{N2} which, thanks to the voltage limiter, has a voltage value lower than VDD.

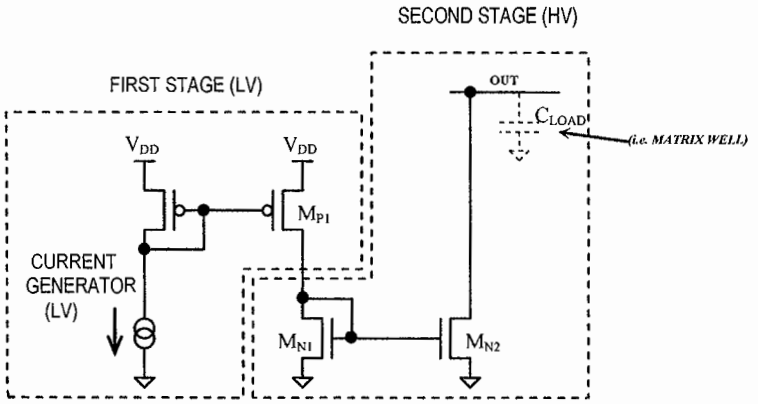


Fig. 11.18. Discharge circuit without drain limiter

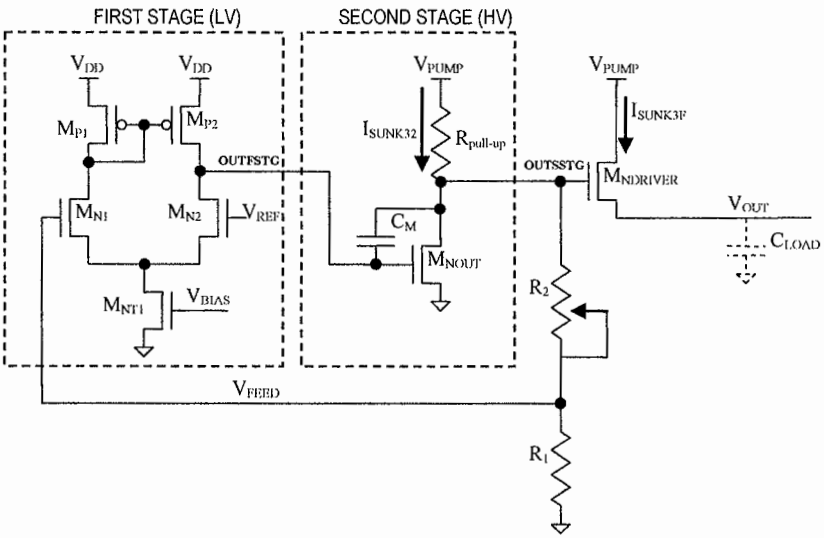


Fig. 11.19. Charge circuit for high capacitive loads

Petitioner Micron Ex-1021, 0319

The resulting circuit is able to discharge a high voltage node by means of a constant current; all the control section is designed using LV transistors, which are a better choice in terms of reliability, performance, area, consumption, etc. A slightly different implementation, where M_{N1} and M_{N2} are HV, is shown in Fig. 11.18.

Another example of the double-supply approach is the circuit shown in Fig. 11.19: it is used to charge the output capacitor C_{LOAD} by means of a voltage ramp. VREF is a ramp-shaped voltage (< VDD) which is amplified by the circuit on the node OUTSSTG. V_{OUT} is, therefore, a ramp voltage equal to $V_{OUTSSTG}$ minus the threshold voltage of the transistor $M_{NDRIVER}$. This is a open-loop regulation useful to drive huge capacitive loads without stability issues in the feedback circuit. A typical application example for such a circuit is the charge of the p-well of the matrix when electrical erase is performed.

11.4 Voltage references

The band-gap reference circuit [1] is widely used in NAND to control the voltages during program and erase operations. The main feature of this circuit (Fig. 11.20) is to produce an output voltage that is stable with respect to both VDD and temperature variations. The temperature compensation technique is extremely simplified because only the first order thermal coefficient of V_{BE} is compensated.

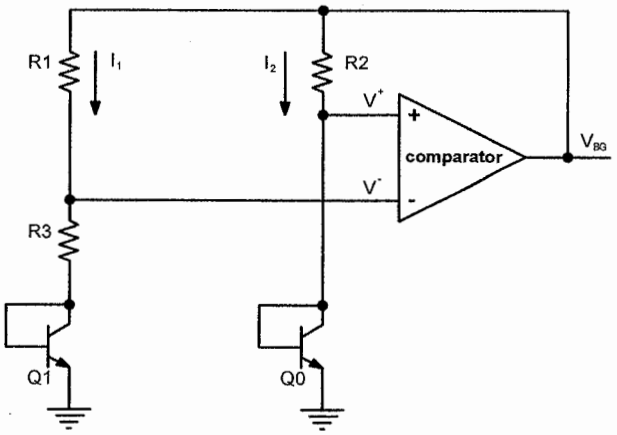


Fig. 11.20. Basic schematic of the band-gap reference circuit

It can be shown that

$$V_{BG} = V_{BE0} + V_t \frac{R1}{R3} \ln \frac{R1 \cdot A_1}{R2 \cdot A_0}, \quad V_t = \frac{kT}{q} \tag{11.11}$$

Petitioner Micron Ex-1021, 0320

where k is Boltzmann's constant, q is the charge of the electron and T is the absolute temperature. A_0 and A_1 are the emitter areas for the transistors Q0 and Q1.

Temperature coefficient of V_{BE} can, therefore, be compensated acting on the values of resistors R1, R2 and R3. The typical plot of V_{BG} over temperature is shown in Fig. 11.21. It is important to note that V_{BG} depends on technological parameters and, therefore, proper values of compensation resistors must be chosen in order to eliminate dependency from temperature, and not to act on the absolute value.

On the other hand, voltage regulators multiply V_{BG} by a specific factor: therefore, the output voltage depends on the absolute value of V_{BG}. In order to satisfy this opposite requirements, the architecture shown in Fig. 11.22 is used. After the circuit of Fig. 11.20 an amplifier is inserted, which generates V_{BG1} voltage to be distributed to the voltage regulators.

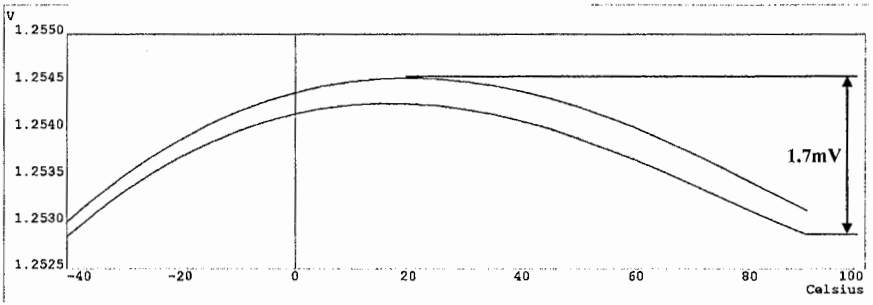


Fig. 11.21. Band-gap reference voltage versus temperature

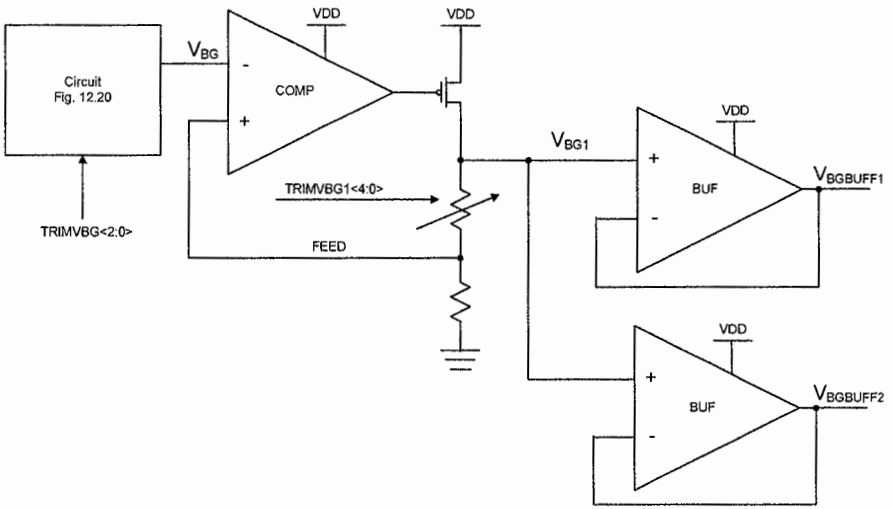


Fig. 11.22. Band-gap reference architecture

Petitioner Micron Ex-1021, 0321

At this point two separate sets of trimming bits (fuses) are placed:

- TRIMVBG<2:0> used for temperature compensation of the band-gap circuit
- TRIMVBG1<4:0> used to provide the proper reference voltage to the voltage regulators

Since the gain of the amplifier which generates V_{BG1} from V_{BG} cannot be smaller than 1, the circuit of Fig. 11.22 does not allow V_{BG1} to be smaller than V_{BG}: this can represent an issue since V_{BG} varies from 1 V to 1.5 V while V_{BG1} is usually centered on 1.25 V. The trick is to modify the resistive divider as shown in Fig. 11.23. In this case the FEED node is tapped in different points depending on the trimming bits configuration.

Since V_{BG} is used by several circuits inside the chip, it is distributed by means of different buffers so that potential noise injected on a given V_{BGBUFF} voltage is not impacting the other circuits.

The choice of the trimming values is performed during factory testing (Chap. 15). V_{BGBUFF} voltages are fed to a device pad and measured by the test machine. It is important to note that, in this way, it is possible to compensate offsets, if any, of the upstream amplifiers.

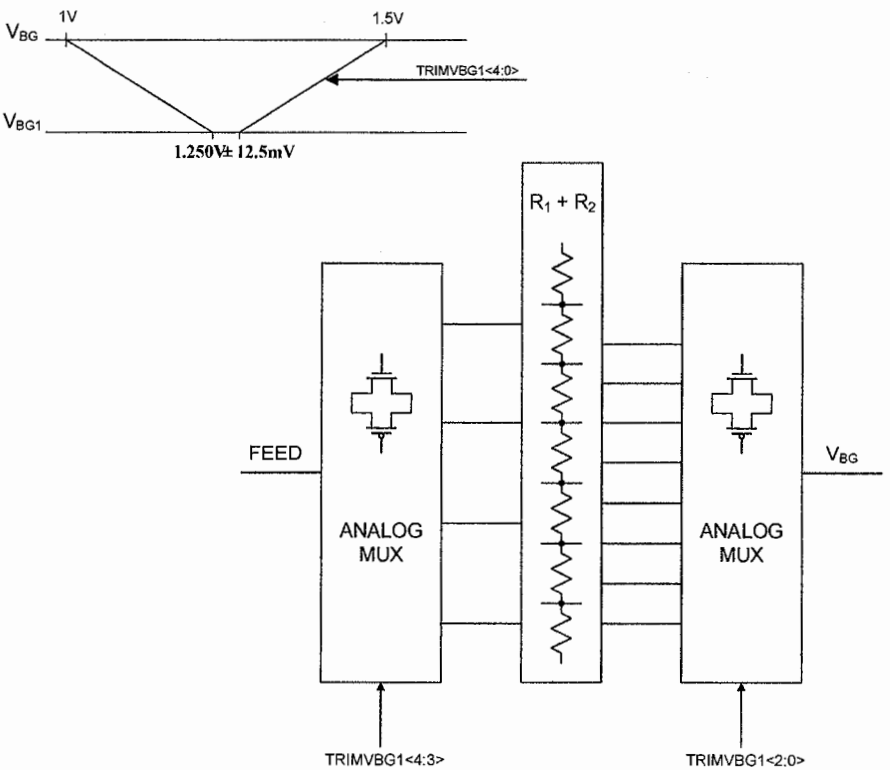


Fig. 11.23. Modified resistor divider for band-gap trimming

Petitioner Micron Ex-1021, 0322

11.5 Internal supply voltage regulator

In many NAND devices, external supply voltage VDD is not directly applied to all the circuits [6, 7]. Some of them are powered by an internal supply (V_{INT}) filtered by a proper voltage regulator and this solution brings several advantages. For instance, in case of devices supplied at 3.6 V, a V_{INT} equal to 2 V allows the use of transistors whose oxide thickness is reduced, which are smaller and better performing. In the case of page buffers, by using V_{INT} it is possible to mitigate the dependency of the triggering threshold from VDD (i.e. several tens of millivolt), which in turn would bring a benefit to the width of the distributions. Of course inside the NAND memory there can be more than one V_{INT} regulators, depending on the design constraints (noise, power consumption, precision required by the circuits).

V_{INT} regulator is therefore a DC–DC converter. Its conceptual scheme is shown in Fig. 11.24.

For the sake of simplicity, it can be assumed that the power supply is needed by logic ports. Voltage drop of V_{INT} during inverter switching depends on the value of the filtering capacitance C_{FILTER}, on the value of fan out capacitance (gates, routing, junctions) and on the cross-conduction current, which in turn depends on the slope of the signal driving the inverters. Beyond a given maximum switching frequency of the logic, V_{INT} dramatically drops. This frequency is directly related to the cutoff frequency of the regulator. Since the DC–DC converter is designed using the same technology of the inverters, its cutoff frequency cannot be higher than the one of the plain inverter.

Let's now see an example, assuming that three ideal voltage regulators are available, each of which featuring a different cutoff frequency: 1 GHz, 100 MHz and 10 MHz. Our system includes 35,000 inverters whose average parasitic load is 20 fF. C_{FILTER} is equal to 10 nF and only one fourth of the inverters is switching. The result of the simulation is shown in Fig. 11.25: it can be noted that the voltage drop is independent from the cutoff frequency of the regulator. The regulator which exhibits the best behavior is the one whose cutoff frequency is 1 GHz, which is comparable with the one of the single inverter (2 GHz).

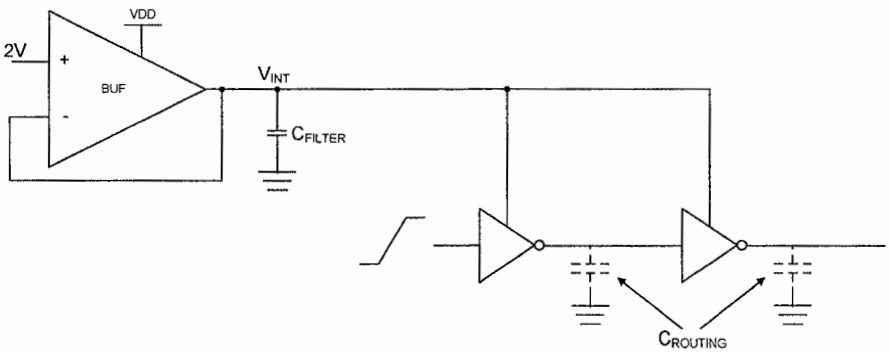


Fig. 11.24. Conceptual scheme of a DC–DC converter

Petitioner Micron Ex-1021, 0323

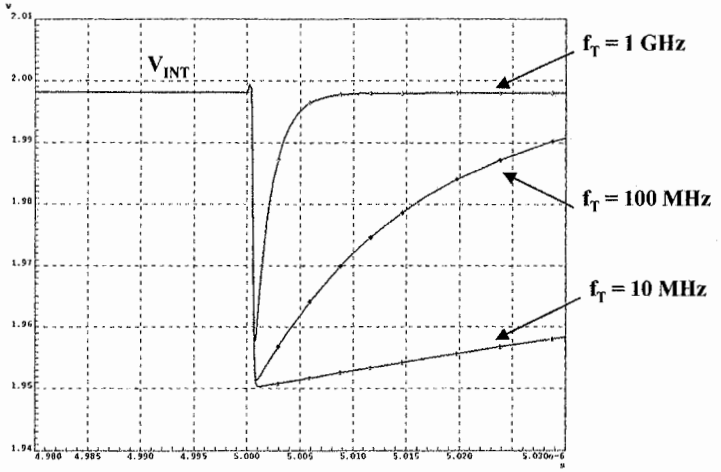


Fig. 11.25. V_{INT} step response for different cutoff frequencies (f_T) of the voltage regulator

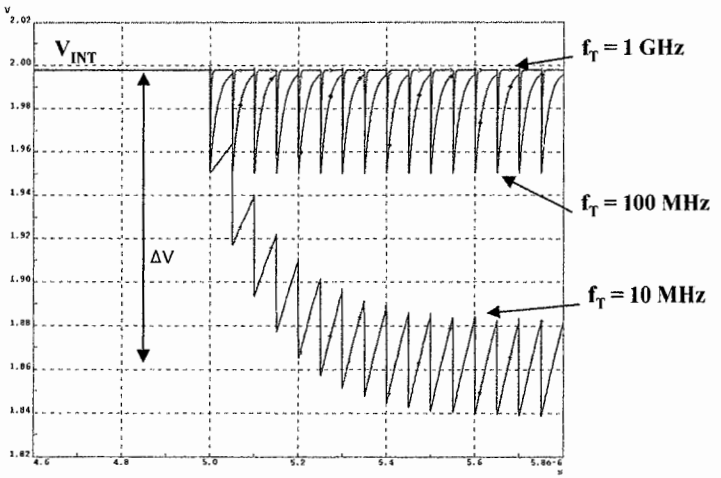


Fig. 11.26. V_{INT} behaviour when logic gates are switching at 20 MHz

Let's now see what happens when the inverters switch at a 20 MHz frequency. Figure 11.26 clearly shows the limit of the regulator with 10 MHz cutoff frequency, which cannot recover in time between consecutive switching events. The resulting ΔV depends linearly, in first approximation, on the switching frequency of the inverters.

Let's now see which are the most common DC–DC converter architectures used in NAND memories. Figure 11.27 shows two different configurations which differ in the output transistor: PMOS PM1 for solution (a) and NMOS PN1 for solution (b).

Petitioner Micron Ex-1021, 0324

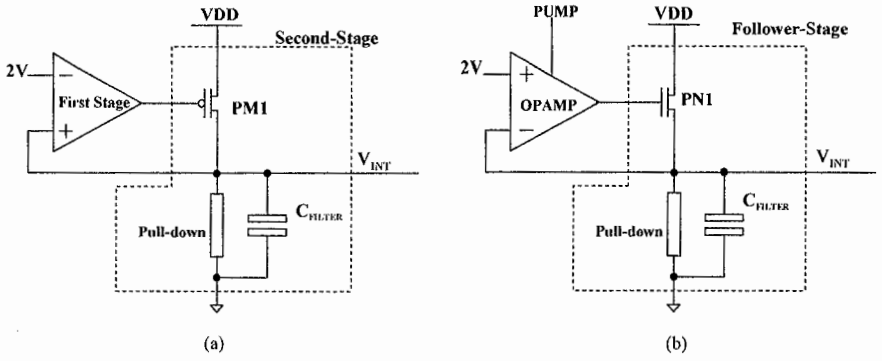


Fig. 11.27. DC–DC converter topologies

The latter needs a charge pump which supplies OPAMP, in order to recover the threshold voltage of PN1; the advantage is that PN1, being of NMOS type, can be smaller than PM1. In both cases, the pull-down stage can be either resistive or an active load.

Owing to the above-mentioned reasons, solution shown in Fig. 11.27a is usually the preferred one. Size of PM1 must be chosen in such a way that the current needed to ensure a proper operation of the connected circuits is guaranteed both in static and transient conditions. On top of that, it is worth noting that PM1 is also responsible of the charge of the filter capacitor during NAND power-up. In order to satisfy all these requirements, PM1 can be some millimeters long.

In case of mobile applications, where power supply is provided by a battery, it is fundamental to limit consumption as much as possible. Of course, it is possible to power down the whole device, but the drawback is that a complete power on sequence is needed to wake it up again, which can take as long as 1 ms to complete. Therefore a stand-by mode is embedded in the NAND memory, like in many other devices, which greatly reduces consumption with respect to active mode. Generally speaking, it is not possible to completely switch off all the internal circuits, since the transition from stand-by to active mode is specified to be quite fast. In other words, there is not enough time to switch on circuits like the DC–DC converter, because of the long time it takes to re-charge the filer capacitor. Figure 11.28 shows a possible solution for stand-by. A main active DC–DC converter is designed, which is operational in active mode. When stand-by mode is entered, the main DC–DC converter is switched off and a secondary one is switched on, whose task is to refresh the charge lost by C_{FILTER}, because of the leakage, and to supply the band-gap circuit.

The size ratio between M1 and M2 is approximately 100:1. The structure shown for the V_{INT} regulator needs a pull-down stage composed of a current generator (indeed a NMOS transistor whose gate is biased at a proper reference voltage).

It is important to note that the transient from the main to the alternate regulator should not cause significant variations on V_{INT} node: in fact, even the first operations after the recovery from stand-by must be fully functional. Therefore the management of SW1 and SW2 switches is particularly critical.

Petitioner Micron Ex-1021, 0325

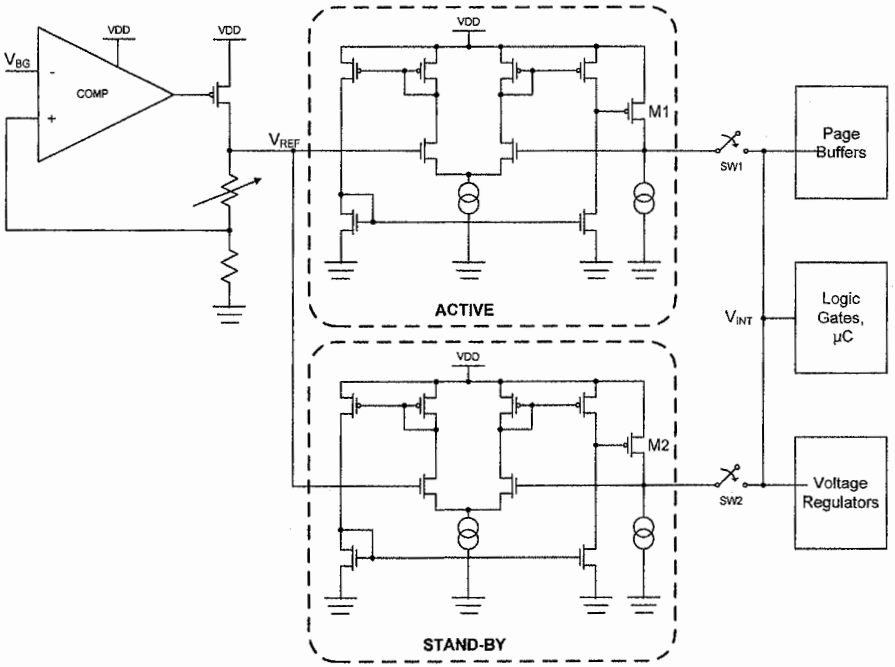


Fig. 11.28. Two V_{INT} generators are used to decrease the power consumption in stand-by mode

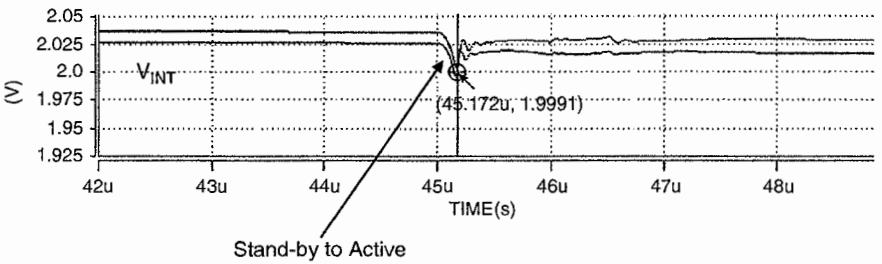


Fig. 11.29. V_{INT} during the recovery from stand-by

Figure 11.29 shows the simulation of the recovery from stand-by for two different values of V_{INT}. Transient lasts some hundreds of nanoseconds, but voltage drop is limited to 25 mV.

Petitioner Micron Ex-1021, 0326

11.6 High voltage switches

As explained in Chap. 12, NAND memories need several high voltages, i.e. voltages higher than the device power supply, VDD. Many circuits, like for instance the row decoder, use HV voltages generated by different circuits. It is therefore necessary to have analog switches which allow the transfer of such voltages whenever required.

If the available technology features both NMOS and PMOS HV, the easiest structure able to transfer a HV voltage is shown in Fig. 11.30. The voltage swing of the IN signal is in the VDD/GND range. Transistors M1, M2, M3 and M4 are placed in a positive feedback loop. When M2 is turned on, the voltage V_{SW} is ground and, therefore, V_{HV} and V_{OUT} are equal. Once M2 is turned off (M1 is on), thanks to the positive feedback, V_{SW} becomes equal to V_{HV} and V_{OUT} is independent from V_{HV}: M8 is off if $V_{OUT} < V_{HV} + V_{TH,M8}$. Scope of transistors M5 and M6 is to reduce V_{DG} on M1 and M2 thus increasing the reliability of the whole structure. Transistor M7 is used to limit the current consumption when the positive feedback takes place.

This circuit needs a PMOS HV able to switch when $V_{DS} = V_{HV}$. Therefore such a solution is not feasible when it is necessary to manage the maximum voltages which are used in a NAND memory (~30 V).

With some NAND processes, a HV PMOS is available: this PMOS transistor has a thick gate oxide (like HV NMOS) and features limits of 4–5 V on the operational V_{DS}. In these cases it is possible to implement the circuit shown in Fig. 11.31 [8].

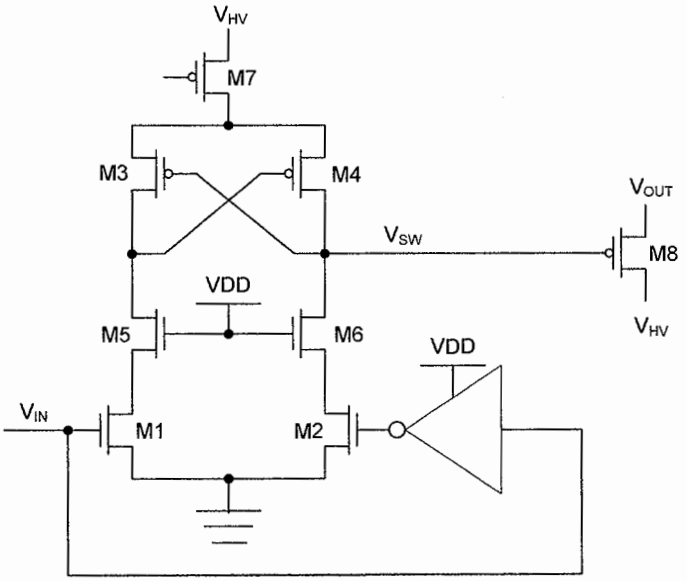


Fig. 11.30. CMOS HV switch

Petitioner Micron Ex-1021, 0327

Behavior is as follows. M3 is a native (depletion) NMOS. Threshold voltages of M2 and M3 are equal to −1 to −0.8 V respectively.

To begin with, let's consider the case where IN is low and V_{SW} = GND. M2 is off and V_{SB} is equal to 0.8 V (M3 has a negative V_{TH}). M4 is off and V_{OUT} is independent from V_{HV}. When V_{IN} approaches VDD, M2 turns on rising the voltage of node SW. Transistor M1 as well contributes to the rise of V_{SW} until when the latter equals to VDD − $V_{TH,M1}$. It is important to note that, at this point, M1 is diode-connected and it isolates the low-voltage section of the circuit. On top of that, the positive feedback between nodes SB and SW (through M2 and M3) takes voltage V_{SW} to the value V_{HV}. One of the main limits of this circuit with respect to the one of Fig. 11.30 lies in the fact that the switch transistor between V_{HV} and V_{OUT} is now a NMOS type. It means that V_{OUT} voltage is (V_{SW} - $V_{TH,M4}$). It is important to note that, due to the considerably high body effect, voltage drop can be as high as 2.5 V. Obviously such a loss must be compensated by rising V_{HV} (i.e. exploiting the charge pumps).

Opposite transition, i.e. V_{SW} going from a high value to GND, is not so easy since it is not possible to switch off M2 using a low voltage signal. It is therefore necessary either to lower V_{HV} supply or to insert an all-NMOS type of HV switch before M3.

Of course switches exclusively based on NMOS HV are the only one that can be used in all those NAND processes where no PMOS HV is available.

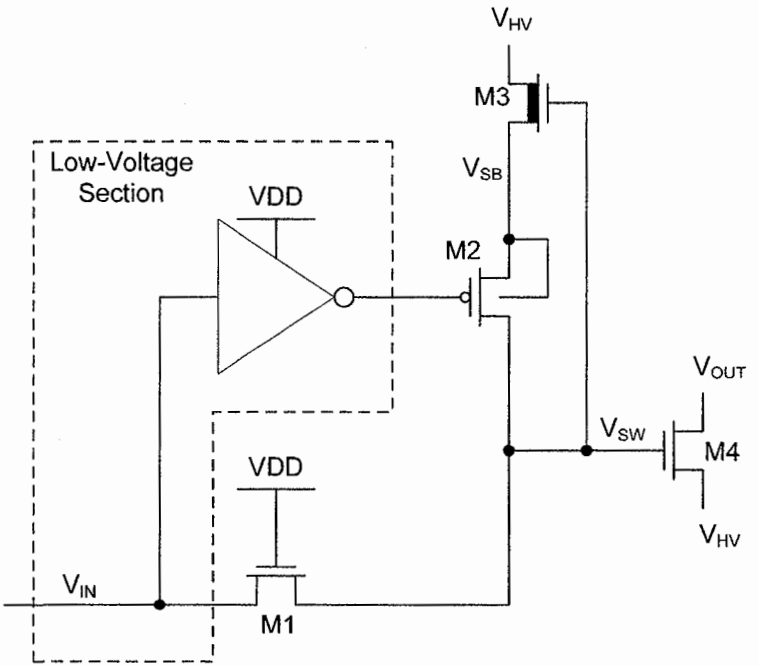


Fig. 11.31. Modified CMOS HV switch

Petitioner Micron Ex-1021, 0328

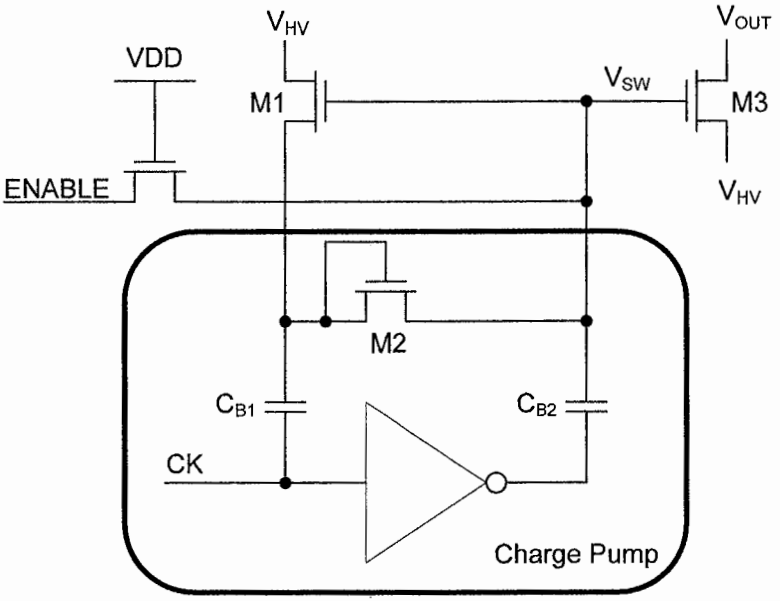


Fig. 11.32. All NMOS HV switch

Let's see the circuit in Fig. 11.32 [9]. Capacitors C_{B1} and C_{B2} and transistor M2 constitutes a Dickson charge pump driven by the clock signal CK.

When the pump is enabled by toggling ENABLE signal to high, feedback transistor M1 allows a growing charge on the capacitors, until the condition $V_{SW} > V_{HV} + V_{TH,M3}$ is met.

Figure 11.33 shows a simulation of the circuit of Fig. 11.32. It can be seen that output voltage V_{OUT} features a ripple of approximately 600 mVpp (peak–peak), which is due only to the clocking of the signal CK, under the assumption of a ripple-free HV input voltage. The existence of such ripple on V_{OUT} is particularly critical in case of multilevel NAND memories. In fact, it must be reminded that such circuits are also used inside the row-decoder, i.e. next to the gate of the memory cells. An excessive ripple on the wordlines would result in a widening of memory cell's V_{TH} distributions and a consequent reduction of operating margins.

The circuit shown in Fig. 11.32 can be modified replacing the Dickson pump with the stage of the voltage doubler described in Sect. 12.1. Basic scheme is shown in Fig. 11.34 [10].

When the switch is turned off, node SW is kept to ground by M_{NH3} and node IN is isolated from V_{HV} by means of transistor M1.

Assuming that all the internal nodes at GND when the circuit starts operating, the internal node IN gets to VDD − V_{THN} through transistor M_{NH2}. As soon as the clock signal CK starts, voltage of node SW grows up to V_{IN} + VDD. Such voltage, applied on the gate of M1, increases the voltage of the node IN until when V_{OUT} equals V_{HV} thanks to the complete switch on of pass transistor M2.

Petitioner Micron Ex-1021, 0329

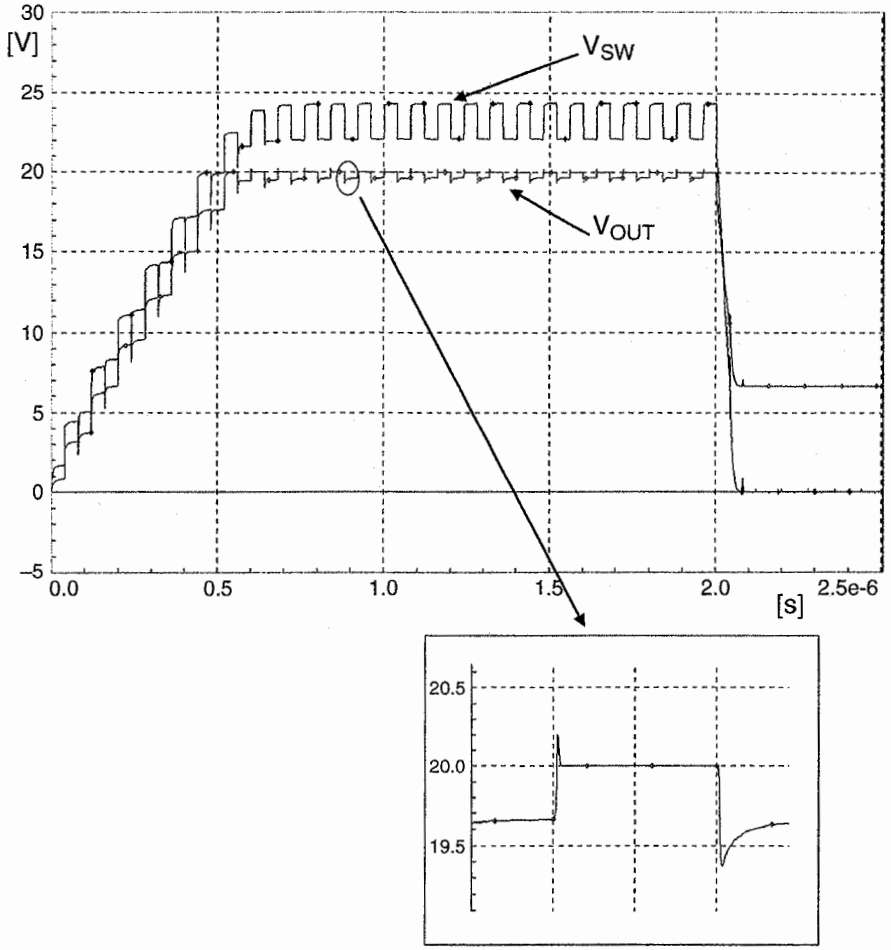


Fig. 11.33. Simulation of all-NMOS HV switch

Transistors M_{NH1} and M_{NH3} are needed only to discharge to 0 V the nodes IN and SW when the switch is inactive.

Figure 11.35 shows a simulation of the circuit of Fig. 11.34. When compared with the scheme of Fig. 11.32, rise timings have not changed, but the peak–peak ripple has decreased to 200 mVpp.

It is important to highlight that voltage differences at the terminals of the transistors inside the circuit never go above VDD; the transistors which have to be high voltage are identified by the "HV" label.

Petitioner Micron Ex-1021, 0330

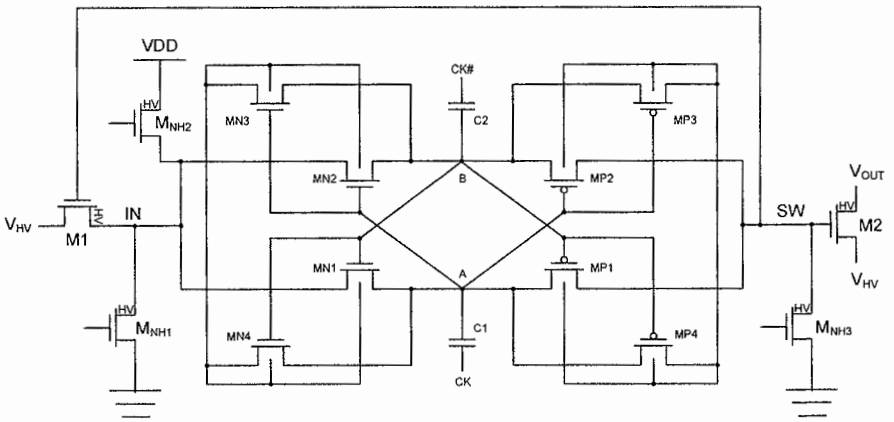


Fig. 11.34. Voltage Doubler HV switch

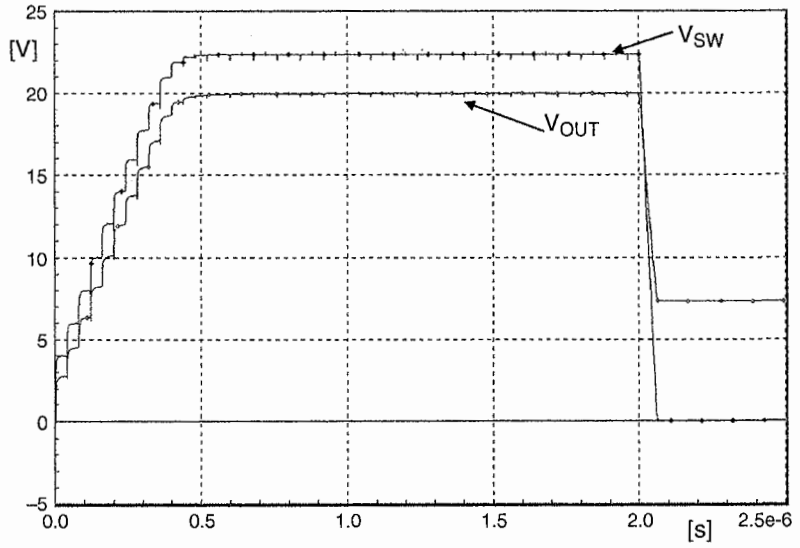


Fig. 11.35. Simulation of voltage doubler HV switch

References

1. G. Campardo, R. Micheloni, D. Novosel, *VLSI-Design of Non-Volatile Memories*, Springer-Verlag, 2005.
2. L. Crippa, G. Ragone, M. Sangalli, R. Micheloni, U.S. Patent No.7474577 – *Circuit and method for retrieving data stored in semiconductor memory cells*. Assignee: STMicroelectronics/Hynix Semiconductor.

Petitioner Micron Ex-1021, 0331

3. T. Tanzawa, T. Tanaka, K. Takeuchi, U.S. Patent No.5864504 – *Nonvolatile semiconductor memory with temperature compensation for read-verify referencing scheme* Assignee: Kabushiki Kaisha Toshiba (Kawasaki, JP).
4. T.-H. Cho, Y.-T. Lee, U.S. Patent No.6870766 – *Multi-level Flash memory with temperature compensation* Assignee: Samsung Electronics Co., Ltd. (Suwon-si, KR).
5. L. Crippa, M. Sangalli, G. Ragone, R. Micheloni, U.S Patent App. 20070164811 – *Multistage regulator for charge-pump boosted voltage applications, not requiring integration of dedicated high voltage high side transistors.* Assignee: STMicroelectronics/ Hynix Semiconductor.
6. K. Takeuchi et al., *A 56-nm CMOS 99-mm^2 8-Gb Multi-Level NAND Flash Memory With 10-MB/s Program Throughput.* IEEE Journal of Solid-State Circuits, Vol. 42, No. 1, Jan 2007 pp. 219–232.
7. G.A. Rincon-Mora, *Analog IC Design with Low-Dropout Regulators,* McGraw-Hill, Electronic Engineering, 2009.
8. T. Tanzawa, U.S. Patent No.7272046 – *High voltage switch suitable for non-volatile memories Assignee:* Micron Technology, Inc. (Boise, ID).
9. J. H. Park et al., U.S. Patent No. 6549461 – *Driving circuits for a memory cell array in a NAND-type Flash memory device. KR* Assignee: Samsung Electronics Co., Ltd. (Kyunggi-Do, KR).
10. G. Ragone, L. Crippa, M. Sangalli, R. Micheloni, U.S. Patent No.7521983 – *High-voltage switch with low output ripple for non-volatile floating-gate memories.* Assignee: STMicroelectronics/Hynix Semiconductor.

Petitioner Micron Ex-1021, 0332

12 High voltage overview

R. Micheloni[1] *and A. Marelli*[2]

12.1 Program algorithm

Program and erase are the only two operations able to modify the content of the cells. Since the content of the cells is strictly related to the number v of electrons inside the floating gate, these operations involve high electric fields (i.e. high voltages) to exploit the Fowler–Nordheim phenomena (Chap. 3) and change v. Particular attention must be taken when dealing with high voltages, since a little variation could have dramatic consequences.

The program algorithm flow is described in Fig. 12.1: programming is achieved through an incremental staircase voltage V_{SEL} (ISPP, Chap. 10) applied to the selected wordline [1]. In this way, the width of the cell's threshold voltage (V_{THR}, Chap. 8) distribution, moving from the erased state to the programmed one, is controlled. For this reason there is a counter K initialized with 1 that counts the number of program pulses performed; $V_{START,P}$ is the ISPP starting voltage and can be around 14 V.

After each program pulse a *Program Verify* (PV) follows. It is basically a read operation to verify that the cell's threshold voltage is beyond $V_{THR,PV}$. If all the cells supposed to be programmed have a V_{THR} higher than $V_{THR,PV}$, than the program operation stops. Otherwise, there are some cells that still need program pulses. When the number of program pulses is equal to K_{MAX}, then the maximum possible number of pulses has been reached and the program exits with a fail. K_{MAX} can be 10 pulses or more depending on the chosen ISPP voltage step ΔV_{PP}. A program fail means that either some cells are too "slow" or they are defective and can't leave the erased state. If K_{MAX} has not been reached, then another program pulse must be issued. For this reason, K is increased and the V_{SEL} applied to the selected wordline is incremented by ΔV_{PP}. The small ΔV_{PP} is, the small the distribution width is, at the cost of a higher number of program pulses and, therefore, a longer program time.

Since the SLC program time is usually around 200 μs, the target number of pulses is generally around 6, ΔV_{PP} is 1 V and V_{START} is 14 V.

The two main phases, Program Pulse (PP) and Program Verify (PV) have a very complex behaviour in terms of timings and signals involved.

[1] Integrated Device Technology, rino.micheloni@ieee.org
[2] Integrated Device Technology, alessiamarelli@gmail.com

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_12,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0333

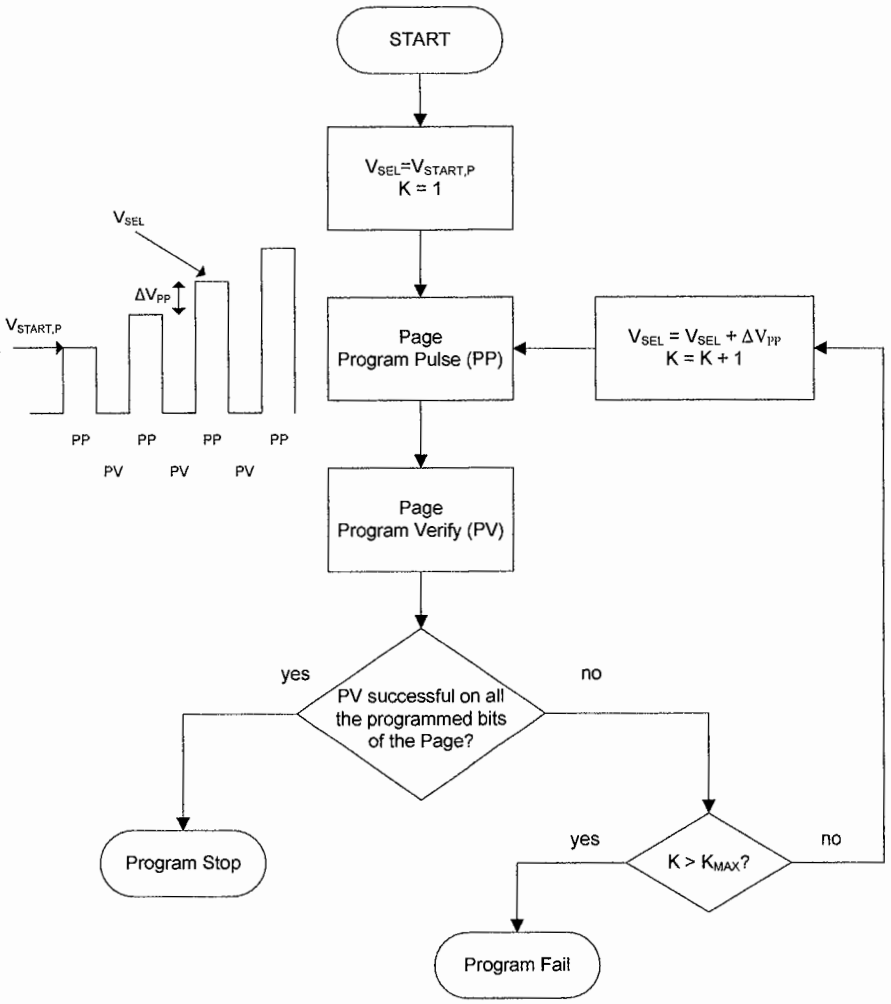


Fig. 12.1. Program flow. Incremental Step Programming Pulse (ISPP) technique is used

Since PV is similar to read and has been already described (Chap. 8), in the following we will focus on the PP phase.

As described in Chap. 3, the main topic that the program pulse must cover is the *Program Inhibit* (PIH) scheme. Different schemes can be used and two possible combinations (Sect. 3.4.4.2), options 1 and 2, are presented in Tables 12.1 and 12.2. In those tables, given a selected WL, each column contains the corresponding NAND string biasing during the ISPP program pulse. Wordlines are indicated with the bus WL<63:0>, dummy wordlines with DWL<1:0>.

PIH option 1 needs higher voltages from the power system: the wordline to be programmed can have its gate biased up to a maximum of 23 V while option 2 is

Petitioner Micron Ex-1021, 0334

limited to 20 V. If the NAND array includes dummy wordlines, those are biased at 10 V with PIH option 1 and at 8 V with PIH option 2.

In order to easy the description of the program pulse, the selected wordline is referred to as WL_{SEL}, the next one on the DSL side is WL_{+1} while the next one on the SSL side is WL_{-1} (Fig. 12.2). For example, if we want to program wordline 60 we have: $WL_{SEL} = 60$, $WL_{+1} = 61$ and $WL_{-1} = 59$. Both the above mentioned inhibit schemes have the same voltages for WL_{+1} and WL_{-1}. Wordlines WL_{-2} and WL_{-3} are biased with the same voltages in both schemes, i.e. 2 and 0 V, respectively. All the other wordlines WL_{UNSEL} are biased with 9 V in the first scheme and 8 V in the second one. Observe that the first scheme require a higher number of different voltage values.

Table 12.1. Program Inhibit option 1. Given a selected WL, the corresponding table column contains the voltages to be applied on the NAND string. V_{SEL} is limited to 23 V. All voltages are in volt

V_{WL} [V]	Selected WL<63>	Selected WL<62>	Selected WL<61>	Selected WL<60>	Selected WL<59>	Selected WL<58>
DWL<1>	10	10	10	10	10	10
WL<63>	$\mathbf{V_{SEL}}$	10	9	9	9	9
WL<62>	10	$\mathbf{V_{SEL}}$	10	9	9	9
WL<61>	2	10	$\mathbf{V_{SEL}}$	10	9	9
WL<60>	0	2	10	$\mathbf{V_{SEL}}$	10	9
WL<59>	9	0	2	10	$\mathbf{V_{SEL}}$	10
WL<58>	9	9	0	2	10	$\mathbf{V_{SEL}}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮
WL<0>	9	9	9	9	9	9
DWL<0>	9	9	9	9	9	9

Table 12.2. Program Inhibit option 2. Given a selected WL, the corresponding table column contains the voltages to be applied on the NAND string. V_{SEL} is limited to 20 V. All voltages are in volt

V_{WL} [V]	Selected WL<63>	Selected WL<62>	Selected WL<61>	Selected WL<60>	Selected WL<59>	Selected WL<58>
DWL<1>	8	8	8	8	8	8
WL<63>	$\mathbf{V_{SEL}}$	8	8	8	8	8
WL<62>	8	$\mathbf{V_{SEL}}$	8	8	8	8
WL<61>	2	8	$\mathbf{V_{SEL}}$	8	8	8
WL<60>	0	2	8	$\mathbf{V_{SEL}}$	8	8
WL<59>	8	0	2	8	$\mathbf{V_{SEL}}$	8
WL<58>	8	8	0	2	8	$\mathbf{V_{SEL}}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮
WL<0>	8	8	8	8	8	8
DWL<0>	8	8	8	8	8	8

Petitioner Micron Ex-1021, 0335

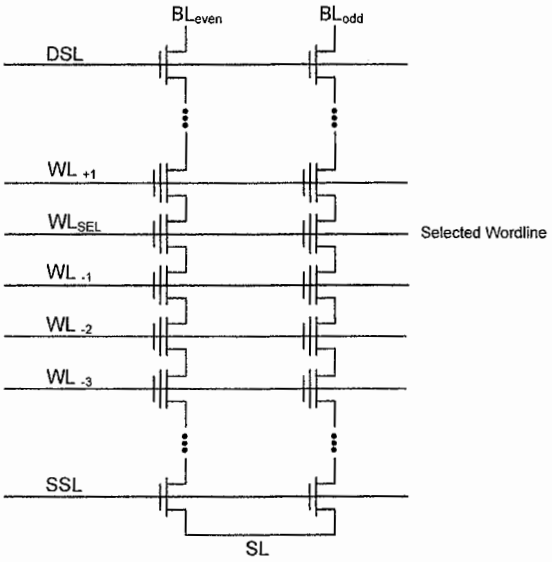


Fig. 12.2. NAND string

In the following we will describe the program pulse using the inhibit scheme of Table 12.2. The program pulse can last approximately 25 μs and is divided in the different phases described in Table 12.3. Table 12.4 represents the voltage master table for the different phases.

During the first phase T_0, the gate of the DSL transistor is biased at 4.5 V. Bitline precharge is selective, since only the bitlines that need inhibit are precharged. Also the source line SL and dummy wordlines DWL are precharged to 1.5 and 8 V, respectively. During T_1, DSL voltage is reduced to 2 V in order to completely shut off the DSL transistor. During T_2, inhibit begins to take place. The unselected wordlines are biased to 8 V, WL_{-3} stays at 0 V as the inhibit scheme requires, while all the others (WL_{+1}, WL_{SEL}, WL_{-1} and WL_{-2}) are biased at 2 V. During T_3, the complete inhibit scheme is enabled: WL_{+1}, WL_{-1} and WL_{-2} are biased at 8 V, while WL_{SEL} is at V_{SEL}. The WLs rise together to minimize coupling issues. This is the longer phase, since the voltage on WL_{SEL} must rise to V_{SEL} (20 V maximum).

Table 12.3. ISPP program pulse timings

Phase	Time (μs)	Operation
T_0	5	BL precharge
T_1	1.5	DSL shut off
T_2	3.5	Inhibit
T_3	11	WL_{SEL} up
T_4	1.1	WL_{SEL} down
T_5	3.1	Shut down

Petitioner Micron Ex-1021, 0336

Table 12.4. ISPP program pulse voltages

	T_0	T_1	T_2	T_3	T_4	T_5
BLeven (V)	0.7	0.7	0.7	0.7	0.7	0
BLodd (V)	2.5	2.5	2.5	2.5	2.5	0
DSL (V)	4.5	2	2	2	2	0
DWL (V)	8	8	8	8	8	0
WL_{UNSEL} (V)	0	0	8	8	8	0
WL_{+1} (V)	0	0	2	8	8	0
WL_{SEL} (V)	0	0	2	20	0	0
WL_{-1} (V)	0	0	2	8	8	0
WL_{-2} (V)	0	0	2	2	2	0
WL_{-3} (V)	0	0	0	0	0	0
SSL (V)	0	0	0	0	0	0
SL (V)	1.5	1.5	1.5	1.5	1.5	0

After that, the program pulse is finished and the last two phases are used to bring back the NAND string to the initial conditions. During T_4, the voltage on WL_{SEL} decreases to 0 V. Finally, during T_5, all the remaining voltages are forced to 0 V.

12.2 Erase algorithm

The erase flow is represented in Figs. 12.3 and 12.4. The erase algorithm is composed by different phases that can be enabled or not, depending on the NAND device. The first one is *Program Before Erase* (PBE). The purpose of this phase is to have all the cells in the same state, so that they can start the electrical erase from the same point (even aging).

If PBE is enabled, then the device performs a program over all the wordlines belonging to the block that must be erased. This program does not require a verify after every pulse, nor the inhibit scheme, since all the WLs must be programmed (except dummy WLs). However, also in this case the program follows a voltage staircase starting from the lowest programming voltage (e.g. 12 V) and incrementing of 1 V at each pulse, up to 17 V. Usually, this phase lasts 150 μs.

After PBE the electrical erase can start: it is a sequence of erase pulses and hard erase verifies (HEV) that pushes cell's V_{THR} towards the erased distribution. The electrical erase is the longest operation and can last 1,000 μs.

Timings and voltages of the erase pulse are shown in Tables 12.5–12.7. During the erase pulse, all the wordlines belonging to the selected block are kept at ground, the matrix ip-well must rise (through a staircase) to 23 V and all the other nodes are floating. V_{ERASE} bias the matrix ip-well for a long period (800 μs), labeled as "erase voltage plateau", during which the cells are erased.

Petitioner Micron Ex-1021, 0337

Since the matrix ip-well (as well as the surrounding n-well) is common to all the blocks, it reaches high voltages also for the unselected blocks. In order to prevent an unintentional erase on those blocks (Sect. 3.5.2), the WLs are left floating; in this way, their voltage can rise thanks to the capacitive coupling between the wordline layer and the underneath matrix layer. Of course, the voltage difference between wordlines and ip-well should be low enough to avoid Fowler–Nordheim tunnelling (Chap. 3).

After each erase pulse a hard erase verify (HEV) follows. During this phase all the wordlines are kept at ground. The purpose is verifying if there are some cells that have a V_{THR} higher than 0 V, so that another erase pulse can be applied. If HEV isn't successful for some columns of the block, there are some columns too programmed. If the maximum number (Z_{MAX}) of erase pulses is reached (typically 4), than the erase exits with a fail. Otherwise, the V_{ERASE} applied to the matrix ip-well is incremented by ΔV_E and another erase pulse follows.

If HEV is successful for all the columns of the block, then the electrical erase is finished and the V_{THR} of all the cells is below 0 V. Anyway, we don't know how far away from 0 V the V_{THR} of the erased cells is. In other words, the erased distribution width and its starting V_{THR} are unknown.

Table 12.5. Electrical erase pulse timings

Phase	Time μs	Operation
T_0	5	Let nodes floating
T_1	200	Ramp ip-well to V_{ERASE}
T_2	800	Erase voltage plateau
T_3	10	ip-well down
T_4	10	Shut down

Table 12.6. Electrical erase pulse voltages for the selected block

	T_0	T_1	T_2	T_3	T_4
BLeven	Float	Float	Float	Float	Float
BLodd	Float	Float	Float	Float	Float
DSL	Float	Float	Float	Float	Float
WLs (V)	0	0	0	0	0
SSL	Float	Float	Float	Float	Float
SL	Float	Float	Float	Float	Float
ip-well (V)	0	V_{ERASE}	V_{ERASE}	0	0

Table 12.7. Electrical erase pulse voltages for unselected blocks

	T_0	T_1	T_2	T_3	T_4
BLeven	Float	Float	Float	Float	Float
BLodd	Float	Float	Float	Float	Float
DSL	Float	Float	Float	Float	Float
WLs	Float	Float	Float	Float	Float
SSL	Float	Float	Float	Float	Float
SL	Float	Float	Float	Float	Float
ip-well (V)	0	V_{ERASE}	V_{ERASE}	0	0

Petitioner Micron Ex-1021, 0338

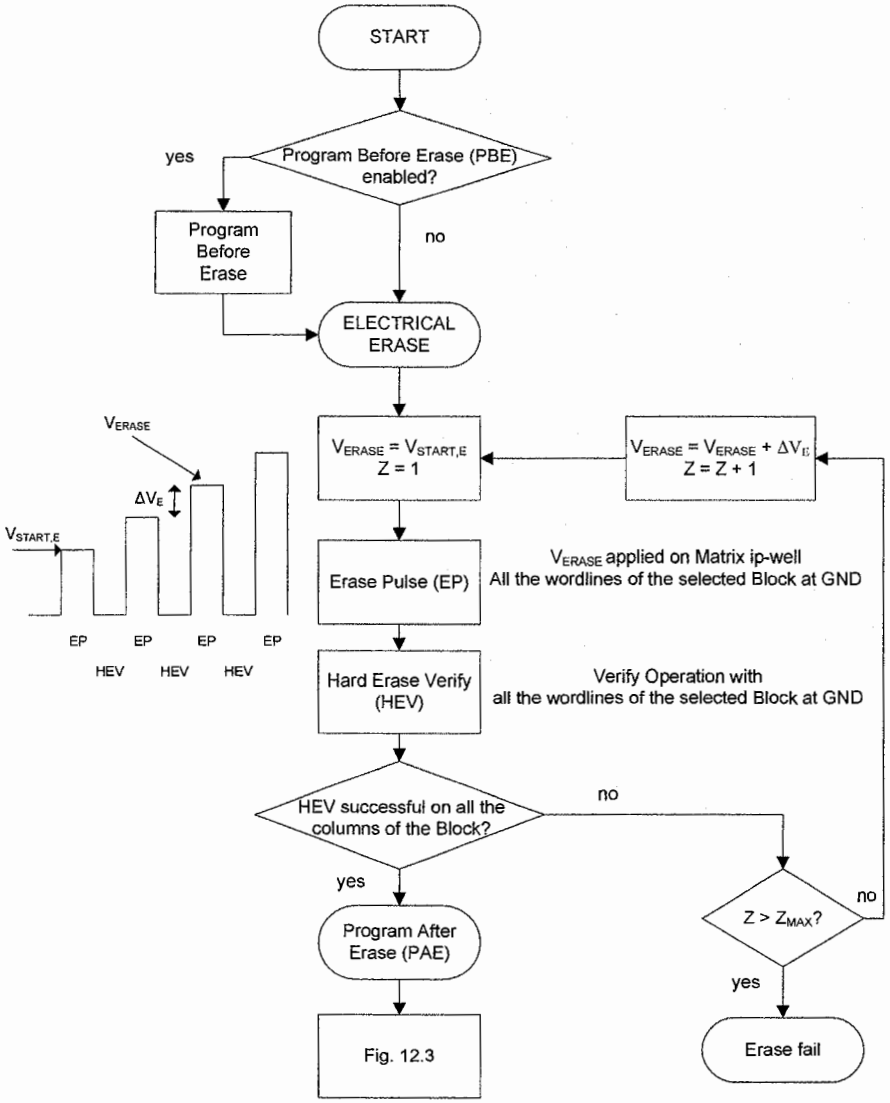


Fig. 12.3. Erase flow: Program Before Erase and electrical erase

Since we know that strongly erased cells can be an issue because of the induced floating gate coupling disturb (Chap. 8), we want to compact the distribution and push it nearer to 0 V. *Program After Erase* (PAE) is the phase that have the goal of compacting the erased distribution and programming the strongly erased cells.

Since PAE is a program operation, it is made up by program pulses (ISPP) followed by *Soft Erase Verifies* (SEV).

As seen in Sect. 9.2, negative-V_{THR} sensing can be done in different ways. With the classic solution (Sect. 9.2.1), all the wordlines are biased at the same voltage

Petitioner Micron Ex-1021, 0339

(e.g. 0.4 V); therefore, the verify operation can only measure the overall current of the whole NAND string. In other words, the above mentioned verify technique is not selective at the memory cell level. As a result, PAE programs all the wordlines in parallel, and when the first string reaches the SEV level, then PAE stops. This PAE verify, done at the sector level, can last around 15–20 µs.

Absolute negative verify (Sect. 9.2.2) is cell selective. In fact, the addressed wordline is biased at ground while the unselected WLs are at V_{PASS}. In this case, PAE can look for and program only those cells which really contribute to the string leakage. Of course, this second verify technique is more time consuming and it can be longer than 30 µs.

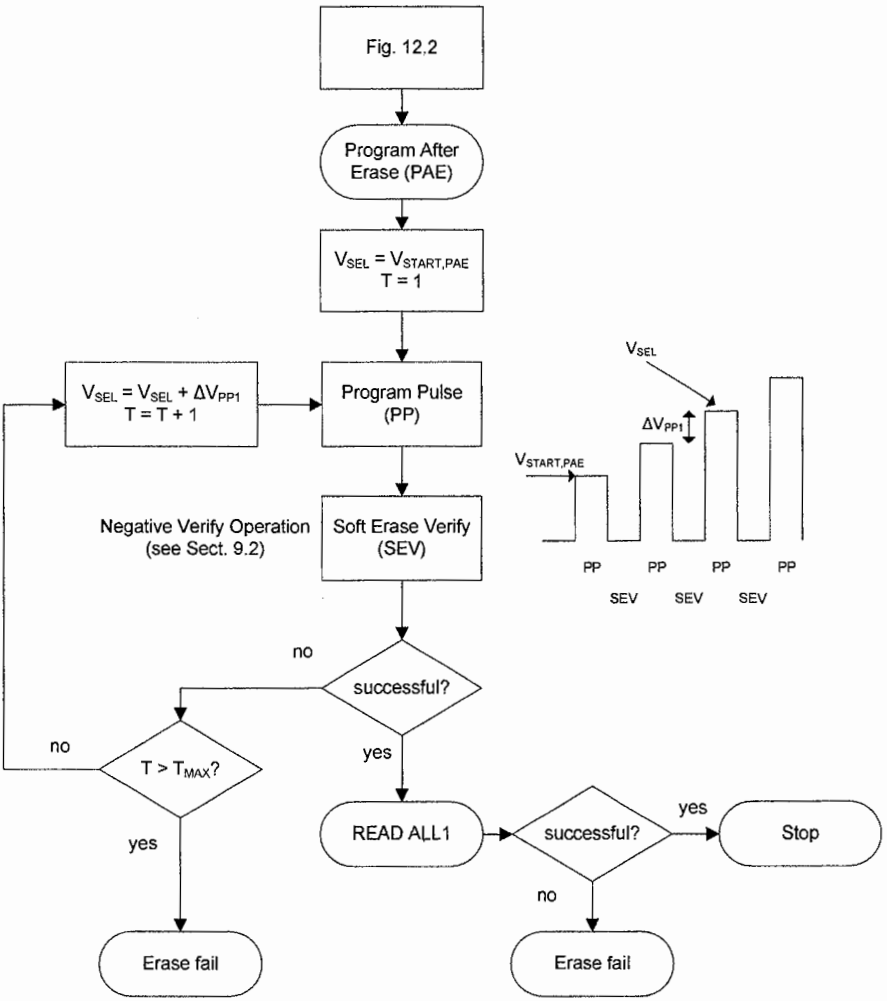


Fig. 12.4. Erase flow: Program After Erase (PAE)

Petitioner Micron Ex-1021, 0340

If the maximum number of PAE pulses is reached and SEV is not successful, then erase operation exits with a fail. Otherwise, a *Read All 1* operation is performed. In this way, we are sure that the erased distribution hasn't been too programmed and the cells can be read as erased. If this last operation is successful, then the block is correctly erased.

12.3 HV system

The high voltage system is shown in Figs. 12.5 and 12.6: those two figures must be analyzed together.

On one side there are the charge pumps and voltage regulators able to generate the voltages described in Tables 12.4 and 12.6; on the other side the HV switches to carry those voltages on wordlines and selectors.

Let's start with the selectors DSL and SSL. DSL selector is connected to the global GDSL through a pass transistor and it can be biased at three different voltages (Table 12.4) depending on the operation and on the specific phase of that operation [2, 3]. In order to select the right voltage for DSL, there is the SW DSL block (Fig. 12.6), whose inputs are: GND, V_{GSLPMP} used to bias the switch itself, V_{GSL} and V_{GDL}. V_{GSLPMP} is generated by the pump GSLPMP, while V_{GSL} is the output of the high voltage regulator GSL REG. V_{GDL} is generated by the low voltage regulator GDL REG. For a detailed description of charge pumps and voltage regulators, please, refer to Chap. 11.

SW SSL block plays the same role, as SW DSL, for SSL: in this case, 2 V is not necessary and the block inputs are V_{GSLPMP}, V_{GSL} and GND.

All the pumps and the regulators use a band-gap reference voltage V_{BG}. V_{INT} is the output of the internal power supply regulator (Sect. 11.5). V_{INT} is generated by two different block: one is used when the NAND device is on (active) while the other is activated during stand-by in order to reduce the overall power consumption.

A more complex system is needed to bias the wordlines. Looking at the example shown in Table 12.2, the more sophisticated WL biasing scheme is required during the inhibit phase of programming when five different voltages are used.

The addressed wordline WL_{SEL} is biased at V_{SEL} which has different values depending on the algorithm. The associated HV switch, sketched in Fig. 12.7, has the following inputs:

- $V_{PASSPMP1}$ generated by PASSPMP1. It is a high voltage, regulated by an internal HV regulator. V_{SEL} has this value during algorithms that require high voltage on the selected wordlines, typically program operation.
- V_{GSLPMP} generated by GSLPMP.
- V_{PEREG} generated by regulator PE REG. This regulator is used to control the voltage V_{PEPMP} coming from pump PEPMP. It is used during program and PBE. $V_{PASSPMP1}$ and V_{PEREG} can be used together, at least for certain voltages, for speeding up the WL_{SEL} charge transient.
- V_{READ} is a low voltage, output of READ REG. It is used during the read algorithm.

Petitioner Micron Ex-1021, 0341

The last two input voltages of SW SEL are $V_{PASSPMP2}$ and V_{PEPMP}. The latter one is produced by PEPMP, as already said, while $V_{PASSPMP2}$ is produced by PASSPMP2 pump; these values are used to bias the HV switches themselves.

During inhibit, WL_{+1} and WL_{-1} (Fig. 12.2) are biased with the same voltage V_{UNSEL1}: it can be chosen among V_{READ}, $V_{PASSPMP2}$ and GND. V_{GSLPMP} and V_{PEPMP} are required by the HV switch for its internal functioning.

WL_{-2} is biased with V_{UNSEL2}. In the inhibit scheme of Table 12.2 this value is 0 V. SW VUNSEL2 can choose between two low voltage values: GND and V_{READ}. As usual, V_{GSLPMP} is used internally by the switch.

WL_{-3} is biased with V_{UNSEL3}: it can be either V_{GSL} or GND. V_{GSLPMP} is used again for the switch.

All the other wordlines inside the NAND string are named WL_{UNSEL} and biased with V_{UNSEL0} which can be chosen among $V_{PASSPMP1}$, V_{SEV} and GND. V_{SEV} is the voltage needed on wordlines during the SEV phase in the erase algorithm. The voltage is generated by the regulator SEV REG. Other inputs of SW VUNSEL0 are V_{PEPMP} and V_{GSLPMP}.

From the above description, it is clear how the wordlines are biased in the inhibit phase of the program operation. In addition it can be easily understood how the complexity of the HV system mostly derive from the different voltages needed during programming [4]. Anyway, there are phases and operations where we don't need to distinguish among WL_{+1}, WL_{-1}, WL_{-2} and WL_{-3}. Let's take, for example, the read operation when we need 0 V on WL_{SEL} and V_{PASS} (generated by PASSPMP1) on all the other wordlines. In that case, SW VSEL outputs V_{READ} while SW VUNSEL0 outputs $V_{PASSPMP1}$. All other HV switches keep their outputs at GND. The row decoder transfers V_{SEL} on WL_{SEL} and V_{UNSEL0} on all the unselected wordlines.

During PBE all the wordlines need to be biased at the same voltage V_{PASS}. In this case there are two possible solutions:

1. V_{SEL}, equal to $V_{PASSPMP1}$, is used for all WLs.
2. V_{UNSEL0}, equal to $V_{PASSPMP1}$, is used for all WLs.

A dedicated switch is used for the dummy wordlines. DWL<0> is biased with $V_{UNSELD0}$. This voltage can be chosen among GND, $V_{PASSPMP1}$ and $V_{PDUMPMP}$. The latter voltage is generated by a specific pump PDUMPMP. DWL<1> is biased at $V_{UNSELD1}$.

The row decoder enables all these voltages on the wordlines and selectors by mean of the voltage V_{BLC}. Also this voltage can be chosen, through a switch, among V_{GSLPMP}, V_{PEPMP}, $V_{PASSPMP2}$ and GND. When V_{BLC} is high enough it opens all the transistors in the row decoder to bias the wordlines in the proper way.

Finally, we can have a look at the NAND matrix. Source is biased with V_{MTSRC}, whose value is chosen by a switch among V_{GSL}, V_{SOURCE} and GND. V_{SOURCE} is generated by the regulator SOURCE REG. Also V_{GSLPMP} inputs that switch.

Petitioner Micron Ex-1021, 0342

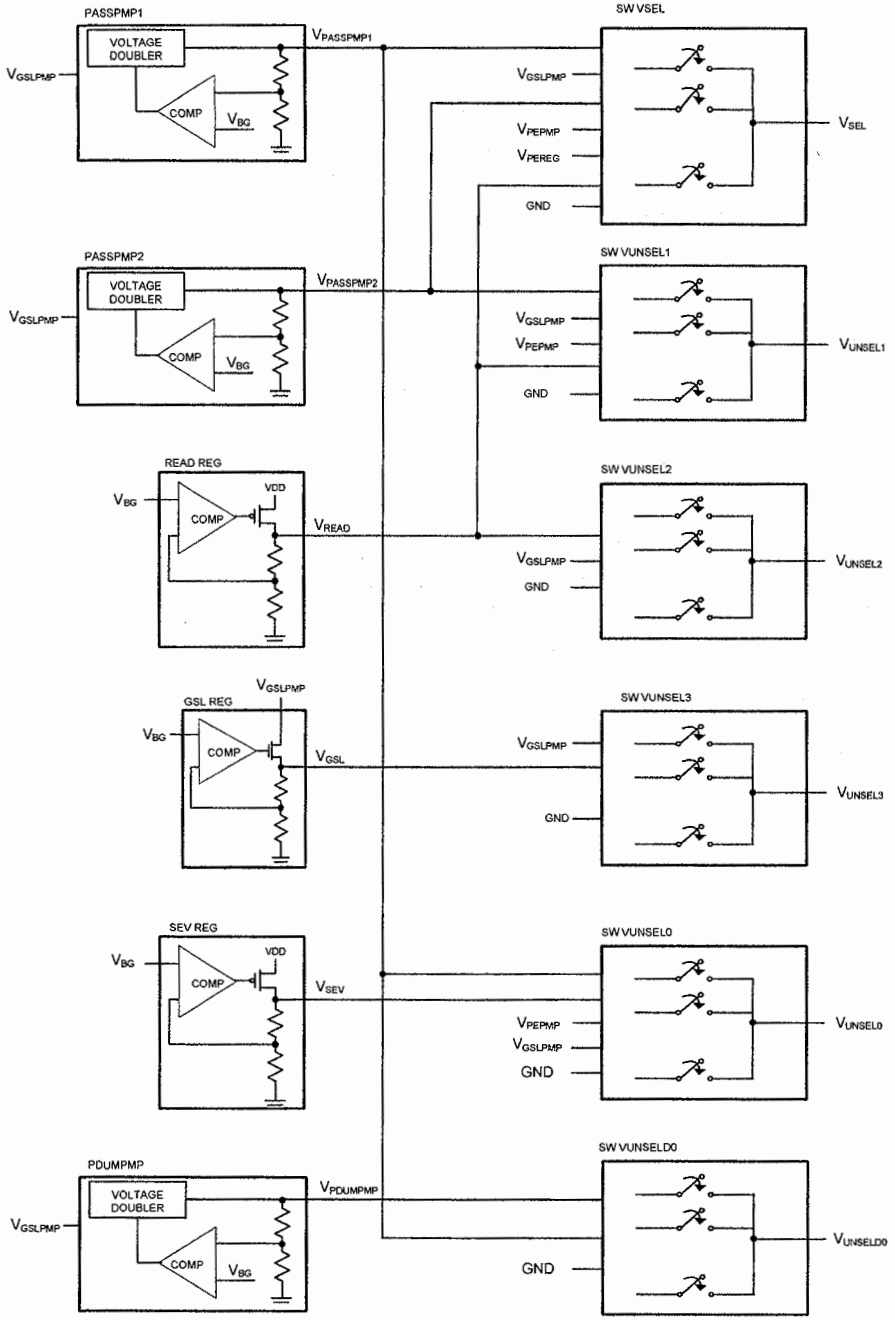


Fig. 12.5. HV system part 1/2

Petitioner Micron Ex-1021, 0343

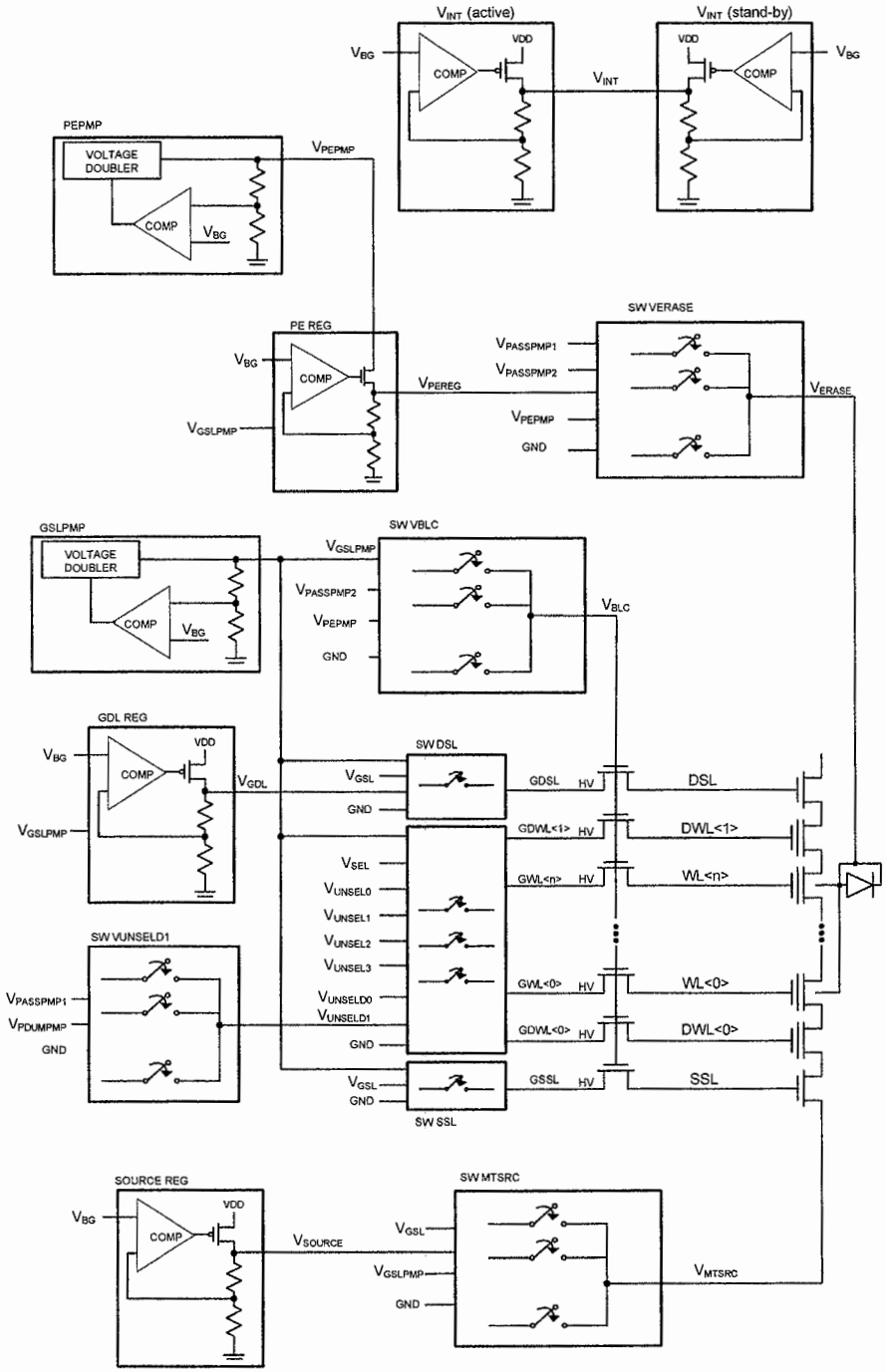


Fig. 12.6. HV system part 2/2

Petitioner Micron Ex-1021, 0344

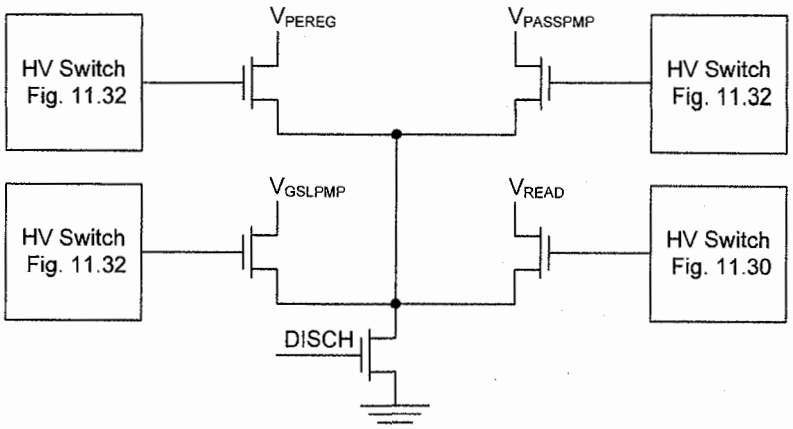


Fig. 12.7. V_{SEL} HV Switch

V_{ERASE} is used to bias matrix wells (ip-well and n-well): it is chosen through a switch among $V_{PASSPMP1}$, V_{PEREG} and GND. Also $V_{PASSPMP2}$ and V_{PEPMP} input the switch.

As just described, HV system is very complex since a lot of pumps and regulators are needed to generate all the voltages. The above description regards a system with the voltages summarized in Tables 12.4 and 12.6. As we will see in Sect. 12.6, there could be other voltage tables with a corresponding different HV system. Moreover, this is not the only way to design the HV system, since there could be more or less pumps depending on the design constraints. For example, in a dual plane device, it can be necessary to have two PASSPMP1 because of the capacitive parasitic load when driving all the wordlines at V_{UNSEL0}.

12.4 Wordline decoder

One of the most critical circuits of the *High Voltage* (HV) system is the one used to bias the *WordLine* (WL). Actually, when it comes to NAND memories, a single wordline is not enough: all the wordlines belonging to the same NAND string must be properly biased at the same time. As a result, the Row Decoder, also called Wordline Decoder or Wordline Driver [5], has to provide a set of voltages: these values are defined by the algorithms described in Sects. 12.1 and 12.2 (see Chap. 8 for the read operation).

The wordline driver comprises:

- A *Pass-Transistor* (PT) for each wordline. These transistors are used to transfer voltages from the *Global WordLines* (GWLs) to the wordlines.
- A circuit to bias the gates of the above mentioned pass-transistors.

The sizing of the components inside the row decoder is particularly critical. On one hand, the size of the pass-transistors (i.e. their series resistance) must not slow

Petitioner Micron Ex-1021, 0345

down the wordline charge/discharge transient too much; on the other hand, all these transistors must be placed nearby the matrix, where the physical wordline begins and, therefore, they must occupy as less area as possible.

Table 12.14 contains an example of the voltages needed by the NAND string during programming; Tables 12.6 and 12.7 do the same for erasing.When NAND technology provides only NMOS-type HV transistors, a possible implementation of the wordline driver is shown in Fig 12.8.

The biasing circuit of the gate of PTs consists of only one high voltage NMOS (M1). At first, all the gates are biased at a high voltage V_{PRECH} through M1. Then, M1 is switched off and, thanks to the gate-drain parasitic capacitance C_{GD} (Fig. 12.9), the rising transient of GWL performs a boost of V_{BLC}, as shown in the simulation of Fig 12.10.

However, there are several critical aspects to consider. First of all, the designer has to deal with a precharge phase of the PT gates: this phase must occur before biasing the global wordlines, otherwise the boost effect would be lost.

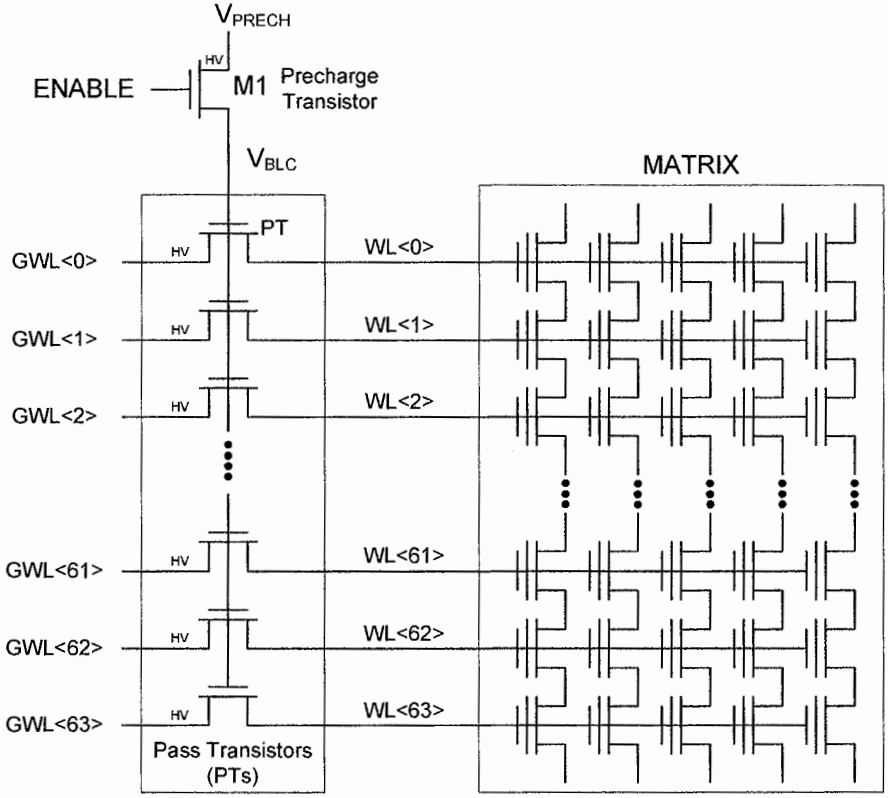


Fig. 12.8. All-NMOS wordline driver

Petitioner Micron Ex-1021, 0346

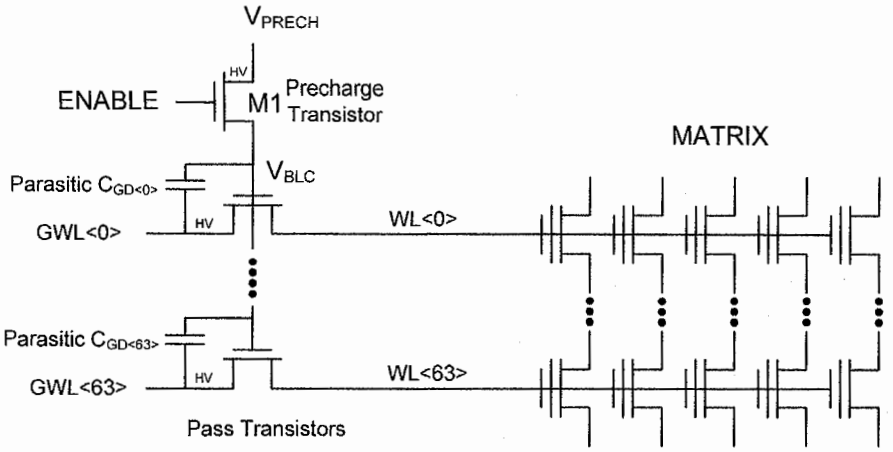


Fig. 12.9. Parasitic capacitors C_{GD} are used to boost V_{BLC}

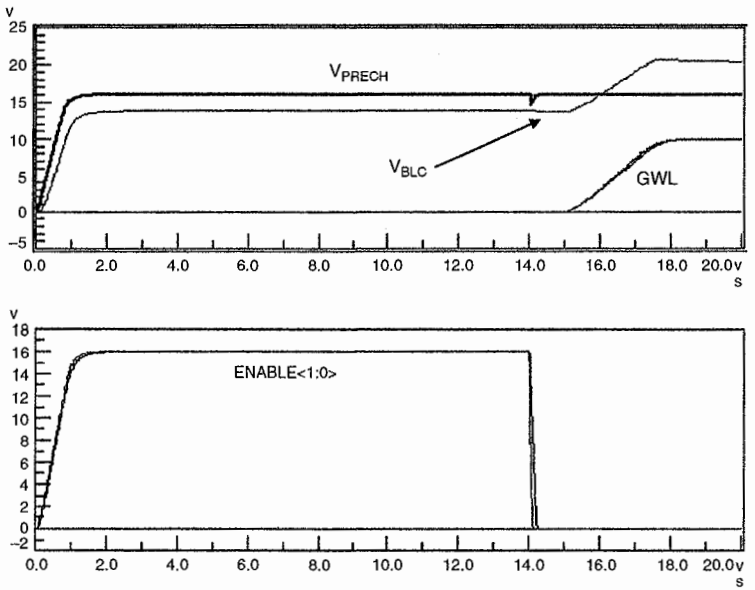


Fig. 12.10. Simulation of the circuit described in Fig. 12.8

The precharge voltage V_{PRECH} has to match V_{MAX}, which is the maximum voltage required during each algorithm. V_{MAX} is not an issue during the read operation, when the voltages are relatively low, but it ends up being close to the breakdown voltage during the program operation. The duration of the precharge phase must be calibrated to allow V_{PRECH} reaching V_{MAX}: this time increases the overall operation time, especially during programming.

Petitioner Micron Ex-1021, 0347

With reference to the circuit of Fig 12.8, precharge is driven by the ENABLE signal. In order to fully exploit the precharge benefit, ENABLE has to be biased with a voltage greater than V_{PRECH}, in order to recover the threshold voltage $V_{TH,M1}$ of transistor M1.

Particular attention deserves the boost operation. Once the boost has occurred, V_{BLC} must guarantee that, even varying temperature and technological parameters, each GWL and its corresponding WL are biased with the same voltage. Unfortunately, process and temperature variations mean that the V_{TH} of the pass transistors can vary as much as 100%. Therefore, the risk is to overcome the breakdown voltage of the oxide in some PVT (Process Voltage Temperature) corners allowed by the electrical specification of the NAND Flash memory.

A possible solution to this problem is shown in Fig. 12.11 where a diode-connected transistor M2 has been added to the circuit of Fig. 12.8. M2 acts as a voltage clamp. When the boost takes place, if the voltage reached by the PT gates exceeds $V_{TH,M2}$, then M2-diode starts sinking current and, therefore, it clamps V_{BLC}. As M2 is a MOS transistor, also its threshold voltage $V_{TH,M2}$ varies with process and temperature. In order to reduce the difference between $V_{TH,M2}$ and the V_{TH} of pass-transistors, M2 is a NMOS transistor with the same size of PTs. Of course, more diodes can be placed in series if there is the need of a bigger voltage drop between V_{BLC} and V_{PRECH}.

During the design activity, special care is taken when charging up the GWLs to their precharge voltage (i.e., GWLs charge transient). Usually, the global wordlines are very long lines that cross all the matrix without interruption, and they are connected to a pass transistor in each block.

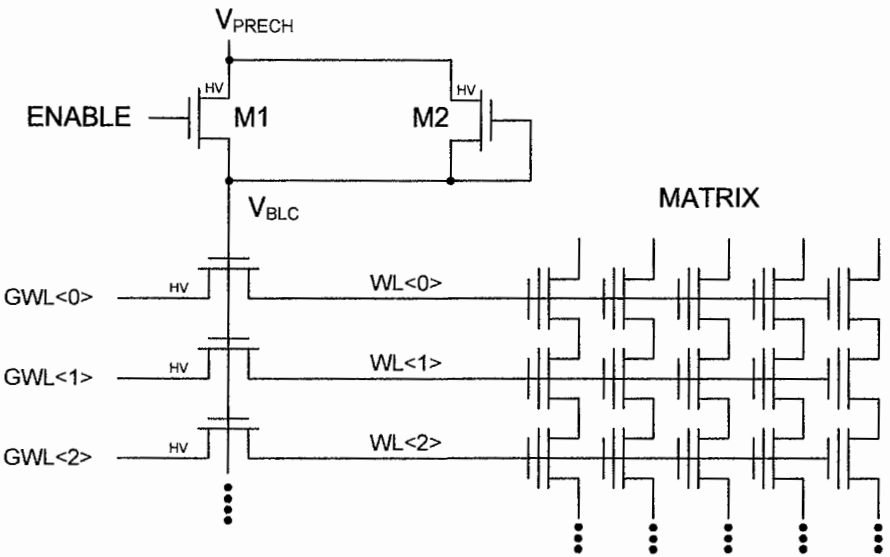


Fig. 12.11. Diode-connected transistor M2 is added to the circuit of Fig. 12.8

Petitioner Micron Ex-1021, 0348

As a result, GWLs are lines with a quite significant parasitic *RC*: a too slow charge transient prevents the proper functioning of the boost operation. It is also appropriate that this charge transient does not vary too much, changing block, process and temperature, so as to make the entire operation more reliable.

It should be noted that the NAND block is selected through the ENABLE signal. As already explained, ENABLE must be of type HV. Since, in a typical NAND device, there are hundreds of blocks, hundreds of ENABLE signals should be generated (through HV circuits), and then routed from the periphery of the NAND device to the row decoder. Of course, this straightforward approach cannot be adopted as it is extremely area consuming.

In order to solve this problem, the circuit of Fig. 12.11 is modified as in Fig. 12.12. M4 plays the role of discharge transistor while M3 is placed in series with M1. In practice, PT gates are precharged in more than one block. Later on, V_{BLC} of non selected blocks are discharged through M4.

The number of precharged blocks is a trade off between the number of HV transistors in series with M1 and the number of HV signals to be routed within the row decoder. Of course, the number of precharged blocks impacts the overall power consumption of the HV system.

When the NAND process offers a HV PMOS, the HV switch of Fig. 11.31 can be reused inside the row decoder. Once again, there is a distinction between the pass transistors PTs and the circuit used to bias their gates as depicted in Fig. 12.13. M2 features a limit of 4–5 V on the operational V_{DS}. M3 is a native (depletion) NMOS. Threshold voltages of M2 and M3 are equal to −1 and −0.8 V respectively. For detailed explanation of the positive feedback loop M2–M3, please, refer to Sect. 11.6.

One of the main advantages of this solution is the fact that V_{HV} is equal to V_{BLC} without asking for a boost operation. On top of that, the precharge phase can be removed with the consequent improvement of the program time.

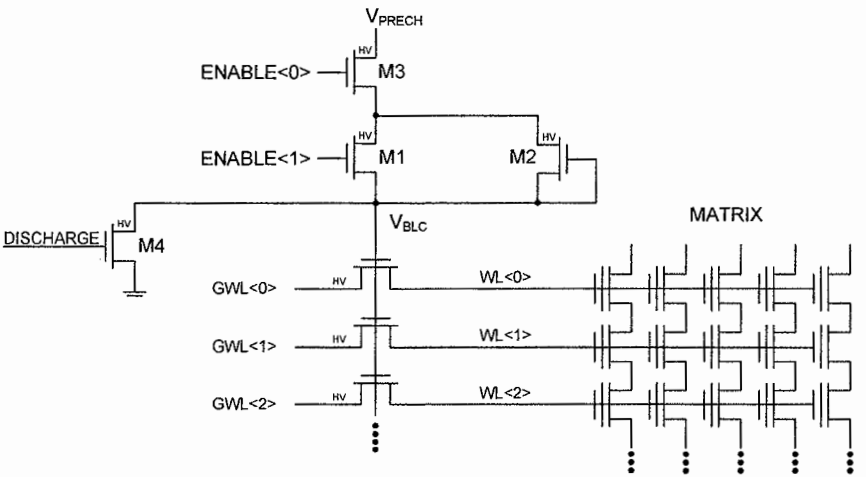


Fig. 12.12. Circuit of Fig. 12.11 is modified adding the discharge transistor M4

Petitioner Micron Ex-1021, 0349

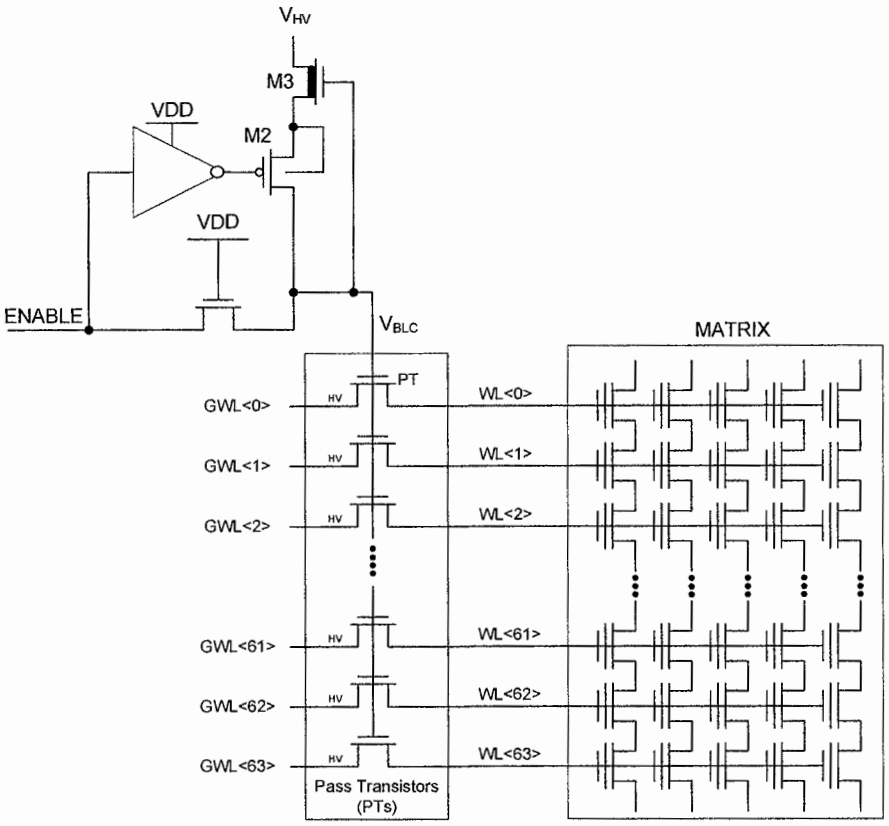


Fig. 12.13. Row decoder with a HV PMOS

12.5 Hierarchical GWL decoder

As we have seen in the previous section, each GWL drives one wordline (through a pass transistor) per block. In state-of-the-art NAND devices, there are 66 wordlines per NAND string: 64 data wordlines, plus two dummy wordlines to cope with GIDL on edge wordlines (Sect. 4.5.1). In multi plane memories, like the one shown in Fig. 12.14, it is common to have a complete set of GWL for each plane: as explained in Sect. 12.4, this is required in order to decrease the parasitic *RC* of the GWL, speeding up the rising transient.

At this point, it is worth noting that each GWL has to properly bias the associated wordline during all the algorithms. As a result, the GWL driver is made up by a number of HV switches (Sect. 11.6) equal to the maximum number of independent voltages used at the same time. Assuming, for example, five independent voltages required during programming, 66 wordlines and two planes, the total number of HV switches for GWLs driving is 660. The area impact of GWL drivers is shown

Petitioner Micron Ex-1021, 0350

in Fig. 12.15 which is a zoom of the bottom part of Fig. 12.14. It must also be underlined that adding one more voltage immediately costs 128 HV switches. Especially with multi-bit per cell technologies, the algorithms (Chap. 16) are becoming really complex with a lot of different voltages required.

A possible solution for reducing the GWL driver impact has been proposed in [6]. The basic idea is to exploit all the benefits of a hierarchical architecture [7, 8].

A *Super Global Wordline* (SGWL) concept is introduced, as sketched in Fig. 12.16. Each SGWL is tied to a subset of GWL. In this particular example, there are 24 SGWL. Dummy wordlines have their own specific SGWL. The issue related to Fig. 12.16 is that more than one GWLs are driven together, while there is the need of addressing each GWL independently as in the previous solution. Therefore, the architecture is modified as in Fig. 12.17 by using a Global Unit Switch (GSU) and a specific SGWL_UN for unselected GWLs.

The SGWL driver is an analog multiplexer among all the independent voltages required by the algorithms: in Fig. 12.18, for instance, each SGWL can carry as much as five different voltages in order to be compliant with the block diagrams shown in Figs. 12.5 and 12.6.

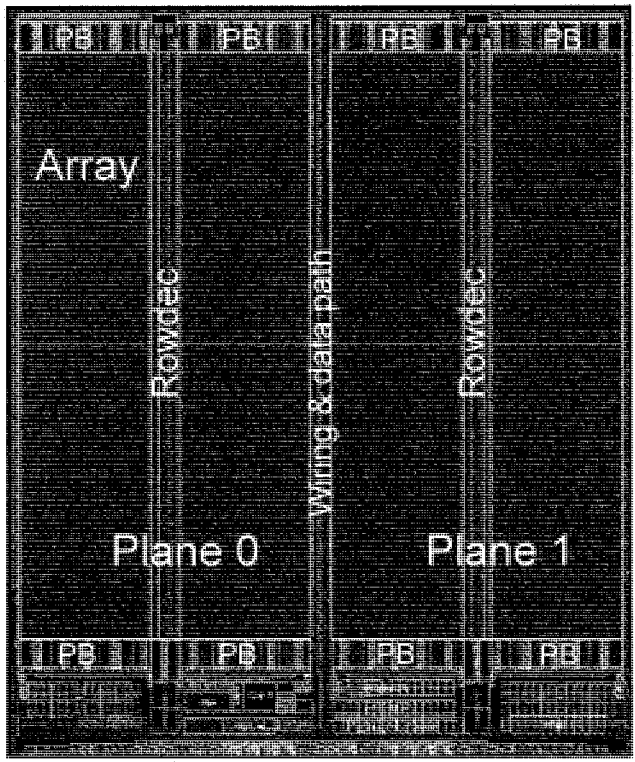


Fig. 12.14. Dual plane NAND Flash memory

Petitioner Micron Ex-1021, 0351

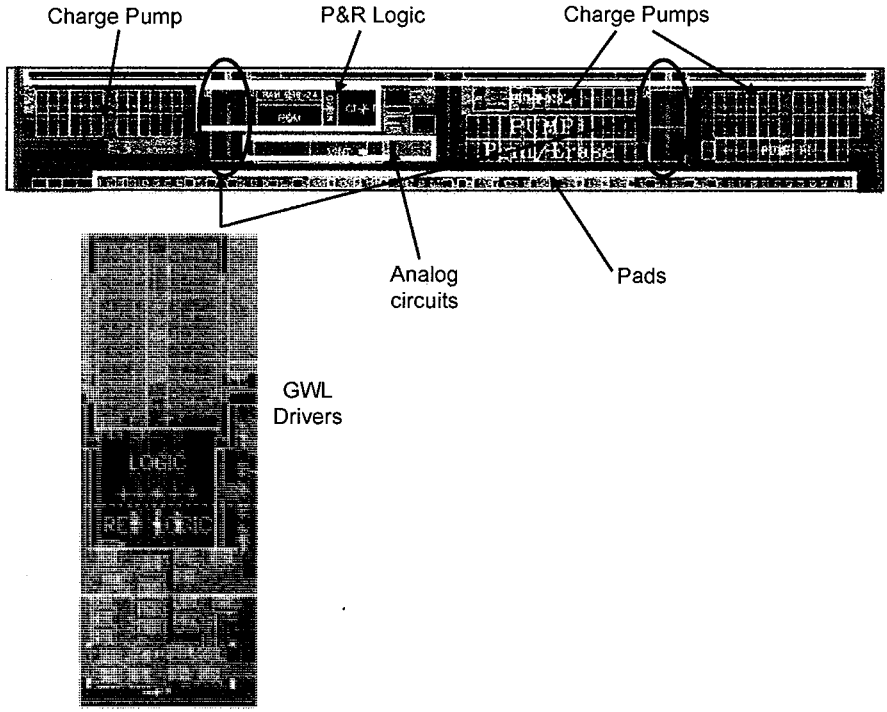


Fig. 12.15. Zoom of the bottom part of Fig. 12.12

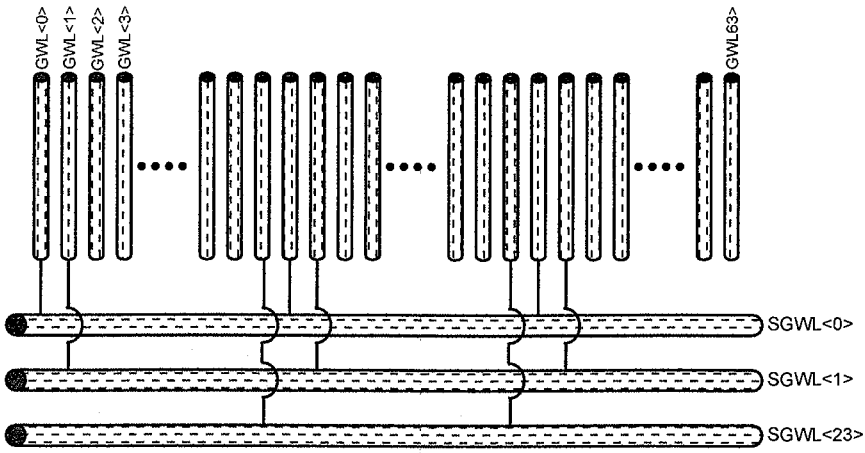


Fig. 12.16. Super Global Wordline (SGWL)

Petitioner Micron Ex-1021, 0352

Let's consider, for example, a program operation. Based on the selected inhibit scheme (Sect. 12.1) and on the addressed wordline, we assume that GWL has to be driven by the voltage available on the SGWL<0> line. In this case, GSU<0> transfers the voltage of SGWL<0> to GWL<0>, while GSU<24> transfers the voltage of the SGWL_UN line to GWL<24>.

As a matter of fact, 24 independent voltages can be applied to the NAND string using only 25 SGWL driver. With the standard approach, as in the HV system of Figs. 12.5 and 12.6, there are just five independent voltages (V_{SEL}, V_{UNSEL0}, V_{UNSEL1}, V_{UNSEL2} and V_{UNSEL3}).

Indeed, with the above described hierarchical approach, adding one more independent voltage during programming (i.e. in the block of Fig. 12.18) asks only 24 HV switches more (instead of 128 of the previous solution).

Figure 12.17 can be modified adding another SGWL_UN line and a third switch per GSU. This is the base for developing an inhibit algorithm in which the unselected WLs on the drain side are biased with a voltage different from the one of the unselected WLs on the source side.

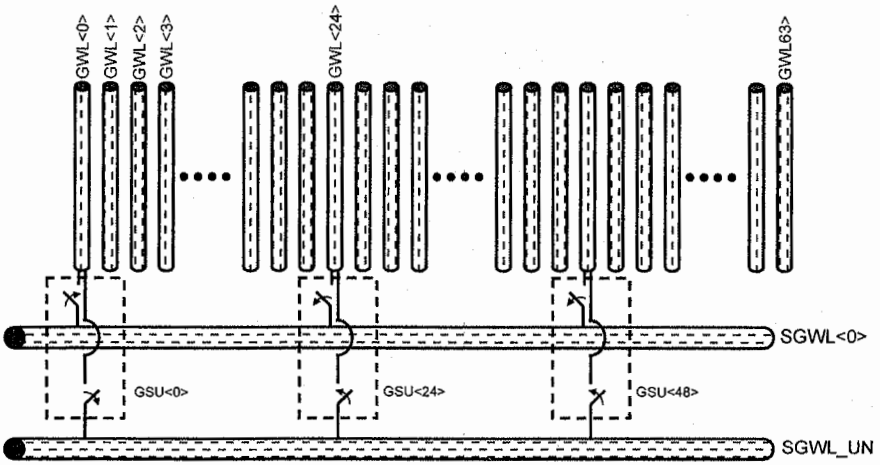


Fig. 12.17. Global Switch Units (GSU) inside the hierarchical GWL decoder

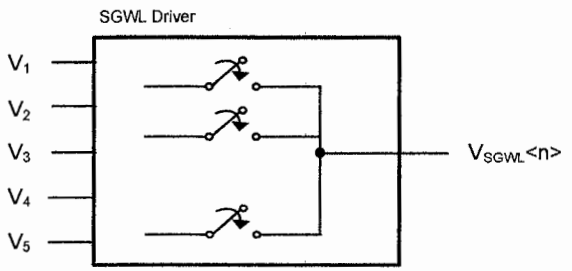


Fig. 12.18. SGWL driver

Petitioner Micron Ex-1021, 0353

The resulting inhibit scheme is very flexible as it is shown in Fig. 12.19. Underlined numbers refer to the available 24 voltages: it is worth remembering that each of these voltages can independently be one out of the available voltages (e.g. 5, Fig. 12.18) in the SGWL driver.

Therefore, the above described approach is particularly suitable for the complex inhibit algorithms required by the XLC storage (Chap. 16).

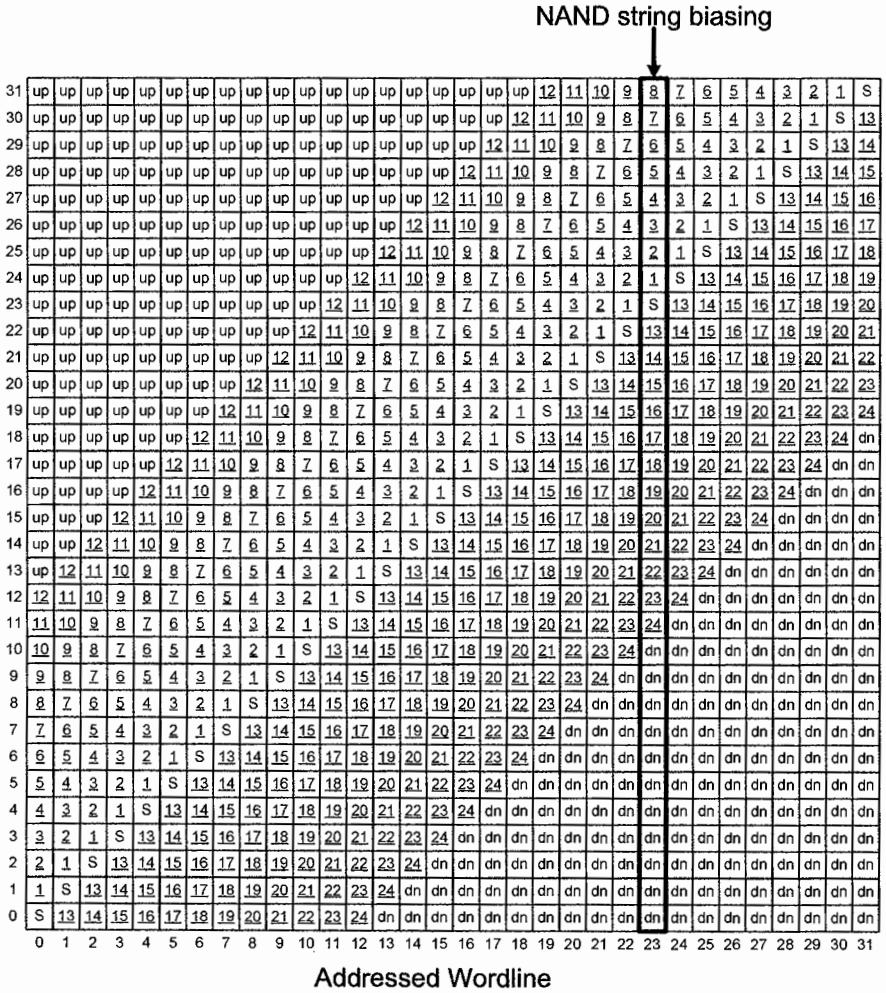


Fig. 12.19. Program inhibit scheme with 24 independent voltages (they are indicated with underlined numbers): "s" is the voltage for the addressed wordline, "up" is the voltage for the unselected wordlines on the drain side while "dn" is the voltage for the unselected WLs on the source side

Petitioner Micron Ex-1021, 0354

References

1. K.-D. Suh et al., *A 3.3 V 32 Mb NAND Flash memory with incremental step pulse programming scheme*, 1995 IEEE Solid-State Circuits Conference (ISSCC), Digest of Technical Papers, pp. 128–129, 350, Feb 1995.
2. Tachee Cho et al., *A Dual-Mode NAND Flash Memory: 1-Gb Multilevel and High-Performance 512-Mb Single-Level Modes*, IEEE Journal of Solid State Circuits, Vol. 36, No. 11, pp. 1700–1706, Nov. 2001.
3. June Lee et al., *A 90-nm CMOS 1.8-V 2-Gb NAND Flash Memory fro Mass Storage Applications*, IEEE Journal of Solid State Circuits, Vol. 43, No. 2, pp. 507–517, Feb. 2008.
4. Y. H. Kang et al., *High-Voltage Analog System for a Mobile NAND Flash*, IEEE Journal of Solid State Circuits, Vol. 38, No. 11, pp. 1934–1942, Nov 2003.
5. P. Cappelletti, C. Golla, P. Olivo, E. Zanoni (Eds.), *Flash Memories*, Kluwer, Boston, MA, 2004.
6. K. Kanda et al., *A 120mm^2 16 Gb 4-MLC NAND with 43nm CMOS Technology*, 2008 IEEE International Solid-State Circuits Conference (ISSCC), Digest of Technical Papers, pp. 430–431, San Francisco, CA, Feb. 2008.
7. G. Campardo, R. Micheloni, D. Novosel, *VLSI-Design of Non-Volatile Memories*, Springer, Berlin, Germany, 2005.
8. G. Campardo, R. Micheloni et al., *A 40mm^2 3V 50MHz 64Mb 4-level cell NOR-type Flash memory*, 2000 IEEE International Solid-State Circuits Conference (ISSCC), Digest of Technical Papers, pp. 274–275, San Francisco, CA, Feb. 2000.

Petitioner Micron Ex-1021, 0355

13 Redundancy

A. Marelli[1] *and R. Micheloni*[2]

13.1 Redundancy concept

Redundancy is part of the NAND circuits which take care of its reliability. In addition to redundancy, modern NAND Flash use error correction codes to improve device reliability.

Redundancy and ECC have the same goal, but they try to achieve a better reliability from a very different point of view. Different sources described in preceding chapters (Chap. 4) produce errors during device's life. In other words, the device is error-free when sold, but program-erase cycles or consecutive reads generate failures in the memory. ECC takes care of errors during device's life; those errors depend on the technology used and on the use of the NAND itself [1, 4].

Anyway, it is also usual to have errors due to defects in the devices production process. The technology used to produce a NAND memory is very complex, so that it is possible that some bitline contacts are not produced as wished, causing bitline shorts, or it can happen that some cells cannot move from the erase state. High productive costs don't allow the discard of every defective memory and, therefore, designers must find a way to correct these hard fails occurring in the production phase. Physical redundancy allows the correction of defective memory parts.

Physical redundancy was the only way to recover defective parts in former Flash Memories, when technology was not so complex and ECC use was not considered.

Conceptually it is very easy: the defect is recognized during Electrical Wafer Sort (EWS); if it is a repairable error, a redundancy element is set in a permanent way, in order to substitute the defective element (Chap. 15). It follows that redundancy requires additional circuits, such as spare bitlines or spare blocks, in order to be able to substitute the defective parts.

The rest of this chapter will explain how it is possible to understand the necessary redundancy resources and how the substitution can take place.

Generally speaking, redundancy is divided into row redundancy, block redundancy and column redundancy.

[1] Integrated Device Technology, alessiamarelli@gmail.com
[2] Integrated Device Technology, rino.micheloni@ieee.org

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_13,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0356

Block redundancy is often applied to block 0, because the device specifications require block 0 to be correct. Every time the defective block is addressed, the device will read the substitutive block instead.

Due to their particular structure, NAND memories haven't row redundancy. When a wordlines short occurs, the block containing those wordlines is marked as bad. The number of bad blocks in a device is defined in the memory specification and represents the number of block containing one or more erroneous bits. Blocks are marked as bad during the production phase and can't be programmed or erased. On the contrary, it is necessary to have a mask preventing their write or read.

The topic of this chapter is column redundancy, i.e. the substitution of bitline shorts or bitlines containing defective cells. Anyway, before going on, it is necessary to remind the memory architecture and how redundancy is inserted in it.

13.2 NAND architecture and redundancy

As we will see in the following sections, there could be a lot of redundancy schemes: some choices are made on probabilistic computation basis, some others on design complexity, but most of them depend on the device architecture. This is the reason why it is important to remind the blocks involved and their impact on redundancy architecture [2, 4, 9, 15, 16].

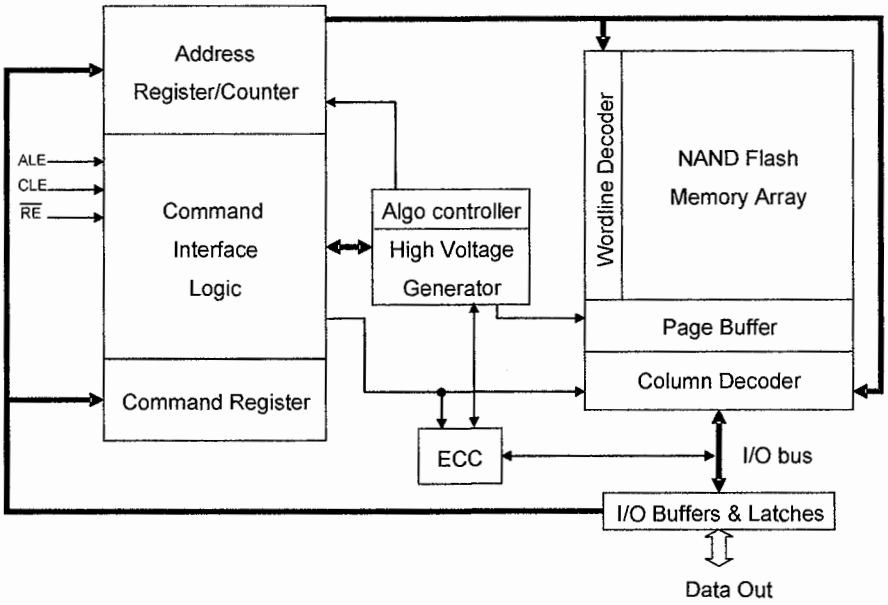


Fig. 13.1. Device logic view

Petitioner Micron Ex-1021, 0357

Figure 13.1 is a device logic view. Data input and data output from device go through I/O and represent read data or data to be written or addresses or commands [6-8].

Command Interface Logic receives some control signals in order to understand what the device can or must do at a particular moment and communicates with Microcontroller (Chap. 6). This one is able to drive signals to high voltage circuits or page buffers.

Every time a read or a program or an erase operation is issued to the matrix, the address must be indicated. This is the reason why I/O are directly linked with Address Registers. The address bus goes to wordline decoder and column decoder. Row and column decoders identify the block, the row and the bitlines involved in the operation.

Let's now see how the array is composed. The matrix is made up by all the cells representing the device size. In other words, 1 Gb device matrix is composed by 1 giga cells if the device is SLC or half giga cells if the device is MLC (every cell stores 2 bits). The matrix cells can be organized in a single or multi-plane architecture.

Figure 13.2 shows a dual plane device. Wordline decoder is in the middle of the two planes (in some devices there could be one row decoder per plane), while, at the bottom, there are page buffers and column decoder (in some devices there could be page buffers also in the upper part).

On the horizontal direction the cells are organized in wordlines and on vertical direction in bitlines. A block is made up by a group of wordlines (typically 32 or 64) and that is the basic unit that can be erased.

Every time an information is read or written in matrix, it is contained in page buffers before going to the I/O. Figure 13.3 shows a NAND block.

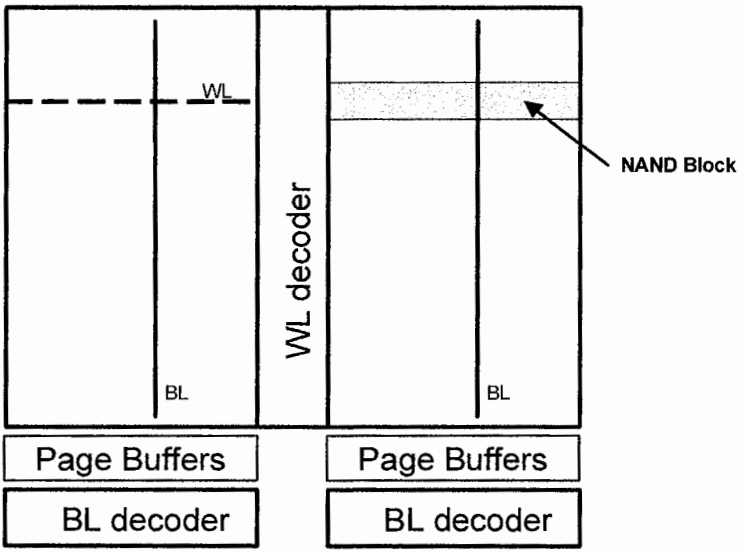


Fig. 13.2. Dual plane device

Petitioner Micron Ex-1021, 0358

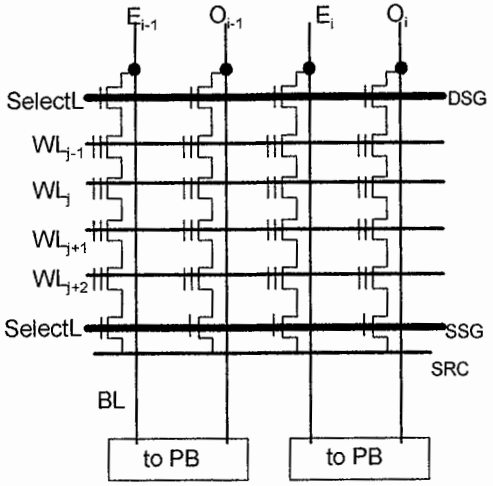


Fig. 13.3. NAND basic block

Wordlines stack (32 or 64) is bordered by Drain Select Gate (DSG) and Source Select Gate (SSG) contacted with that block. Bitlines (typically 8 or 16 KB) contact the whole plane and end in page buffers or sense amplifiers.

The bitline length is a critical parameter to take in account, when dealing with parasitic effects. Therefore, sometimes bitline is cut in half parts, contacting in this way only half plane. Bitlines are divided into even and odd. Even bitlines contain information related to a page, while odd bitlines contain information related to the consecutive page. In this way every wordline contains two logic pages for an SLC device and four logic pages for a MLC device [10-14]. Supposing to have an SLC device with a stack made up by 64 wordlines: each block contains 128 logic pages.

It follows that every time a read or a program operation is issued, only a number equal to half bitlines (or a quarter in case of bitline cut) is involved at the same time. Therefore, it is possible to have a number of page buffers equal to half the number of bitlines, so that one bitline even and one odd end in the same page buffer. This won't be true anymore with the All Bitline Architecture (see Chap. 8).

Every time we operate on the array, the address is composed by different fields (Fig. 13.4):

- One data field is the block address. It is typically composed by at least 10 bits and is decoded by row decoder.
- One data field is the logic page address. This address involves both row decoder, that must identify the wordline and column decoder, that must enable even or odd bitlines. It is typically composed by 5 or 6 bits depending on the stack size. WL address length is one bit shorter compared with page address length. The non-involved bit is used by column decoder to understand if even or odd bitlines are involved in the operation.
- One data field identifies the bytes and regards only column decoder. Usually we operate with 4 or 8 kB pages, so that it is composed by 16 or 17 bits.

Petitioner Micron Ex-1021, 0359

Block Address	Page Address	BL Address

Fig. 13.4. Different data fields composing the address

Suppose we have a single plane device without bitline cut option (Fig. 13.5). The array is composed by only two blocks made up by four wordlines and 2 bytes per page.

Suppose we want to address block 1, page 3 and byte 0.

The address generator generates 5 bits going to row and column decoder.

Row decoder uses 3 bits in order to recognized the addressed wordline, in the figure WL 1 of block 1. Column decoder uses the remaining 2 bits to identify the byte and even or odd bitlines. In the figure, page buffers of byte 0 and odd bitlines are enabled.

Generally speaking, column decoder involves much more bits in respect to row decoder and is designed in a hierarchical way. In the figure the enabled bitlines are in bold. We must observe that, in case of a device with bitline cut architecture, we would have doubled the number of page buffers and only half bitlines would have been enabled.

Column decoder makes also an electrical-topological scrambling so that contiguous data won't be in contiguous places in matrix. In the figure it is possible to see that data, composing the same byte, are not consecutive. The major effect of the scrambling is to redistribute noise, in order to decrease the probability of having catastrophic events that could destroy the whole information of a byte. This effect is useful both for redundancy that substitutes defective bitlines in the matrix and for ECC that deals with bytes composing a page.

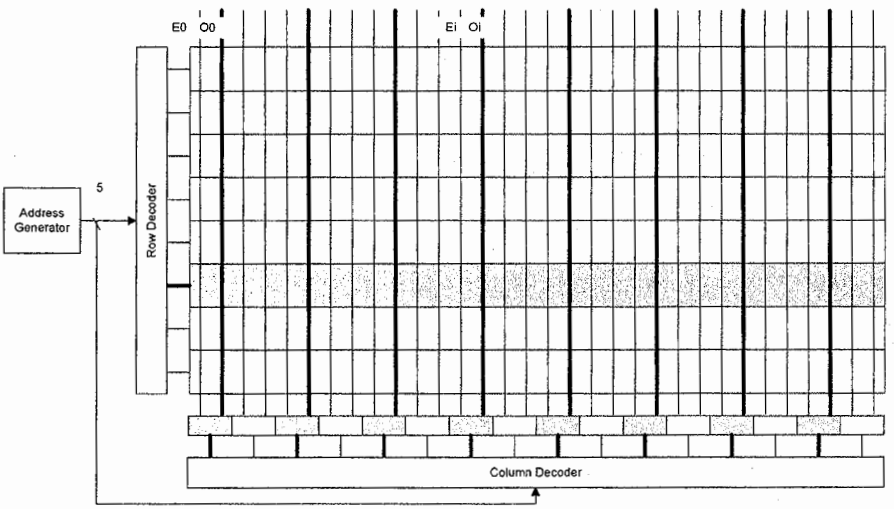


Fig. 13.5. Single plane device composed by two blocks, four wordlines and 2 bytes per page

Petitioner Micron Ex-1021, 0360

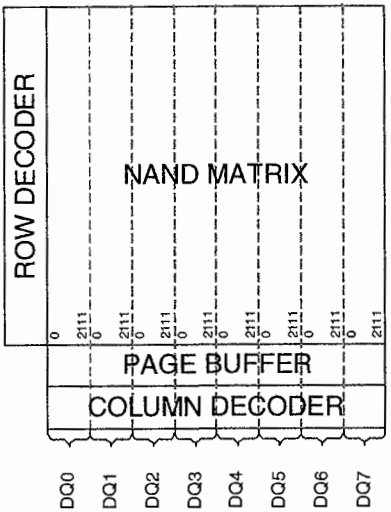


Fig. 13.6. Electrical–topological scrambling in a NAND array

In this way matrix is divided in eight different areas: the first area will be composed by all bit 0 of all bytes, the second one by all bit 1 and so on (Fig. 13.6).

A short among two or more bitlines in the first area involve bit 0 of two or more different bytes (and not consecutive).

In the following, we will see how redundancy substitutes defective bitlines. Anyway, it is worth to anticipate that the substitution can involve one or more bitlines at the same time, but the number of bits necessary to identify them is different. For the example of Fig. 13.5 the substitution of an even and odd pair needs 4 bits (there are 16 bitline pairs), while the substitution of a single bitline needs 5 bits (there are 32 bitlines).

13.3 Process data failure analysis and redundancy requirements

It's not easy to understand how many redundancy resources are necessary for a given device and how the substitution can be carried out, since the choice involves many different design aspects. On one hand, the substitution can't impact over read and program operations, this requires circuits to be optimized in order to reduce area and time overhead. On the other hand, we don't want a too high number of redundancy resources, because they introduce an area overhead. Therefore, for a given technology, it is worth to have a probabilistic analysis of fails, in order to minimize the chip failure probability due to redundancy.

Petitioner Micron Ex-1021, 0361

13.3.1 Failure analysis concept

The chip failure probability due to redundancy allows the comparison of different solutions, before designing them. In the next sections some particular architectures with their probabilistic analysis will be discussed. Here we will introduce some basic concepts.

Usually, NAND memories require a redundancy resource in two fail cases: bitline short and single cell fail. In single cell fails the whole bitline is substituted, even if only one cell is in error. On the other hand, bitline shorts are due to problems with metallization deposition and affect all the cells belonging to two or more bitline.

Figure 13.7 shows a bitline short for a given NAND device.

Let Nc be the cell number in a matrix, Nb the bitline number and p the cell error probability; it is possible to compute the probability P_f of a device failure due to lack of redundancy resources.

$$P_f = 1 - \left[(1-p)^{\frac{Nc}{Nb}} \right]^{Nb} \tag{13.1}$$

Obviously, this formula is no more valid in case of redundancy substitution. Given a technology, it is possible to understand which is the cell error probability and than compute the chip failure probability due to redundancy.

As already explained above, bitlines are divided in even and odd. Every time a bitline needs to be identified an address x plus a single bit to discriminate between even and odd is used. Every time a redundancy substitution is performed, a redundancy address y unable to discriminate between even and odd is associated with failure address x. In this way one bitline even and one bitline odd is involved in every substitution.

It is although possible the substitution of only one bitline, with advantages and disadvantages compared with the bitline-pair choice.

First of all, the pair substitution requires an address shorter of one bit. Suppose having a device with 32.768 bitlines: 16.384 even and 16.384 odd. Every time the address indicates a bitline fail, both even and odd bitlines are substituted, so that the address must be able to identify only 16.384 bitlines.

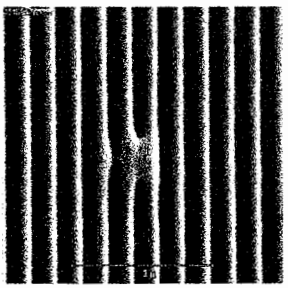


Fig. 13.7. Picture of a bitline short

Petitioner Micron Ex-1021, 0362

Note that, in this case, one redundancy resource is composed by one even bitline and one odd bitline. It follows that a device with 512 redundancy resources has 1,024 bitlines dedicated to redundancy.

Then, the substitution table is halved in bitline-pair substitution. For the example taken into consideration, bitline-pair substitution needs 512 matches before ending resources, while single bitline substitution needs 1,024 matches before ending resources. In other words, single bitline substitution has a table of 1,024 rows (number of substitutions) and 15 columns (address to identify 32,768 bitlines). Instead, bitline-pair substitution has a table of 512 rows and 14 columns, with an area gain (in terms of fuses as will be explained after) of more than 50%.

Every constraint increases chip failure probability due to redundancy, so that this area gain could reflect a reliability decrease.

As already said, redundancy takes care of two error sources: single cell fails and bitline shorts.

If we analyze single cell fails, the substitution of 512 pairs over 16,384 or 1,024 over 32,768 is very different. Figure 13.8 shows chip failure probability due to single cell fails for a device (or a part of it) that substitutes 64 bitline pairs over 4,096 pairs or 128 bitlines over 8,192 bitlines.

As it is possible to see, the bitline-pair substitution is worst than single bitline substitution. The real gap depends on the number of cells belonging to the same bitline, so that the difference highlighted in the figure have only an explicative purpose. Anyway, the substitution of single bitline is better for all the probabilities, because there are less constraints and less resources are "wasted" in comparison with bitline pairs substitution.

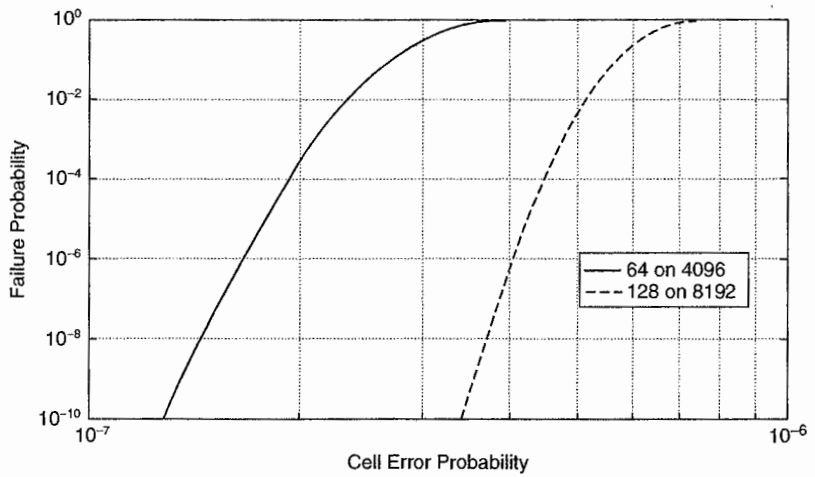


Fig. 13.8. Failure probability due to single cell errors for a device applying bitline pairs substitution (continuous line) or single bitlines substitution (dashed line)

Petitioner Micron Ex-1021, 0363

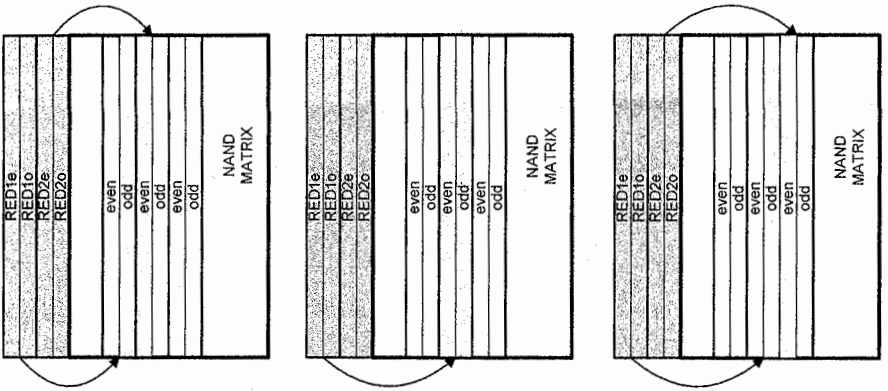


Fig. 13.9. Possible position of short between two bitlines in matrix

The situation changes if we analyze failure probability due to bitline shorts. Supposing a bitline short could affect only two bitlines, cases presented in Fig. 13.9 can occur.

For cases (a) and (c) the short involves two bitlines (light grey in the figure) associated to different redundancy resources, so that two resources must be used. Instead, in case (b) the short is between two bitlines associated to the same redundancy resource, so that a single substitution takes place.

In case of single bitline substitution, every time two resources are used, while with bitline-pair substitution a single resource is used in 33% of cases.

It's not possible to state what is better between a gain in area or a decrease in failure probability, since it depends on the number of resources needed, on available area and on failure probability targeted.

Anyway, the trend shown between single bitline or bitline-pair substitution can be worked up to the substitution of four bitlines at the same time. In that case, there will be another 50% gain in area, in comparison with bitline-pair substitution.

Figures 13.10 and 13.11 show failure probabilities for these different substitutions in single cell errors case or bitline shorts case.

The two figures show different behaviour in case of single cell fails or bitline shorts.

For single cell error source, the substitution of single bitlines is always better compared to bitline-pair or four bitlines substitution. If the error source is the bitline short, the substitution of a single bitline is better only for small probabilities. When bitline short probability grows, we gain more with the substitution of two or four bitlines, because there are cases in which it is possible to use only one resource.

In the following we will treat methods in which one substitution involves one even bitline and one odd bitline.

Petitioner Micron Ex-1021, 0364

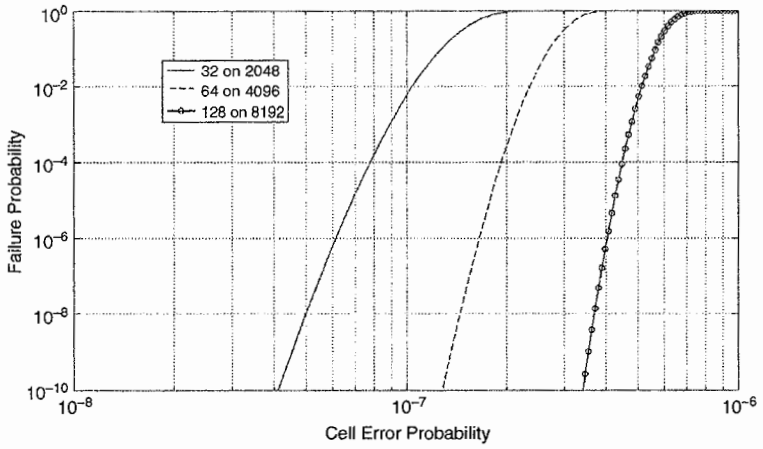


Fig. 13.10. Failure probability comparison for single cell fails in case of single bitline substitution, bitline-pair substitution and four-bitlines substitution

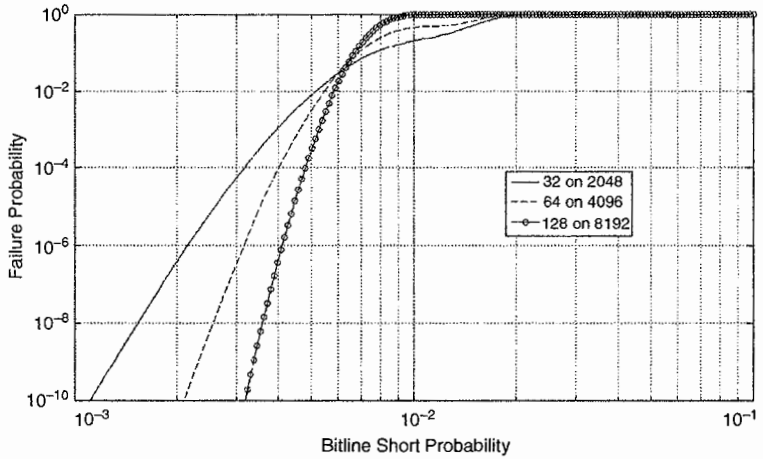


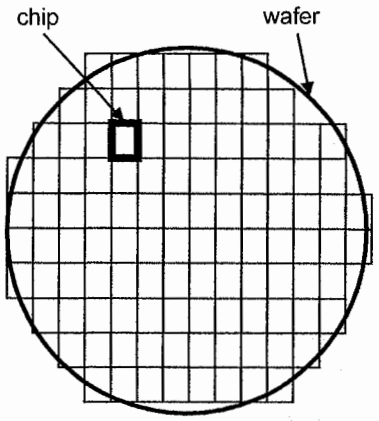
Fig. 13.11. Failure probability comparison for bitline shorts in case of single bitline substitution, bitline-pair substitution and four-bitlines substitution

13.3.2 EWS substitution

Once a device is in production and under test, it is possible to understand how many redundancy resources are used.

Tests list is very long (Chap. 15) and for every single test executed, a redundancy substitution is afterwards performed. At the end of the EWS flow we know how many redundancy resources have been used and we can obtain a figure like the one depicted in Fig. 13.12b.

Petitioner Micron Ex-1021, 0365



(a)

######### REPAIRED COLUMN (FINAL) MAP #########

[FLAT ZONE] : LEFT

11					---	---	---				
12					---	160	---	---			
13			---	---	115	168	---	---			
14		---	---	188	---	84	141	143	---	---	
15		---	---	---	93	---	87	139	121	120	
16	---	175	143	134	132	90	---	166	96	97	---
17	---	---	118	---	119	123	---	107	159	---	---
18	---	---	110	---	78	---	171	114	99	123	---
19	---	---	---	---	143	---	152	127	176	128	134
20	---	---	151	---	---	---	---	143	---	---	111
21	---	---	86	---	---	---	160	---	---	152	104
22	---	142	---	136	202	173	95	141	63	108	82
23	---	---	---	104	114	122	125	---	107	237	106
24	---	186	---	133	101	121	152	---	---	125	127
25	---	---	122	135	---	134	146	---	---	---	---
26		---	125	134	133	80	192	---	---	107	
27		---	149	179	127	133	---	---	---	---	
28			---	109	112	99	123	---	---		
29				---	---	---	---	---			
30					---	160	---				

|---|---|---|---|---|---|---|---|---|---|---|
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |

(b)

Fig. 13.12. (a) A wafer containing N chips and (b) a wafer where in every chip position the number of redundancy resources is indicated

Figure 13.12a represents a wafer, where every square is a chip. Figure 13.12b is the same wafer as Fig. 13.12a, where the numbers indicate the redundancy resources used by the chip in that position.

It is also possible to perform more accurate analysis, for example we can understand how many devices are failed due to lack of redundancy resources, after a specific test.

Figure 13.13 shows the failure probability for a given device, after some tests indicated as bins. Suppose that bin 140 and bin 120 are redundancy tests; it follows that the failure probability due to redundancy for that device is 1.77% + 1.56% = 3.33%.

EWS output result can be very complex with details regarding all tests. Even redundancy structure can be very complex: resources can be split in independent blocks.

We can have up resources located at the top of the device or down resources located at the bottom. Otherwise, the resources can be split in right and left redundancy blocks.

As a consequence, an accurate analysis of redundancy failures can lead to specific topological tests.

Figure 13.14 represents redundancy analysis for a device having the resources split in up and down blocks. The redundancy test is called 40 times during EWS flow. The graph represents a lot, therefore, it is the sum of a high number of dice. One line represents the number of dice surviving a specific test. For example, tests 5–25 haven't an impact on the number of surviving dice (around 120), while test 30 generates the fail of 67% of dice.

Petitioner Micron Ex-1021, 0366

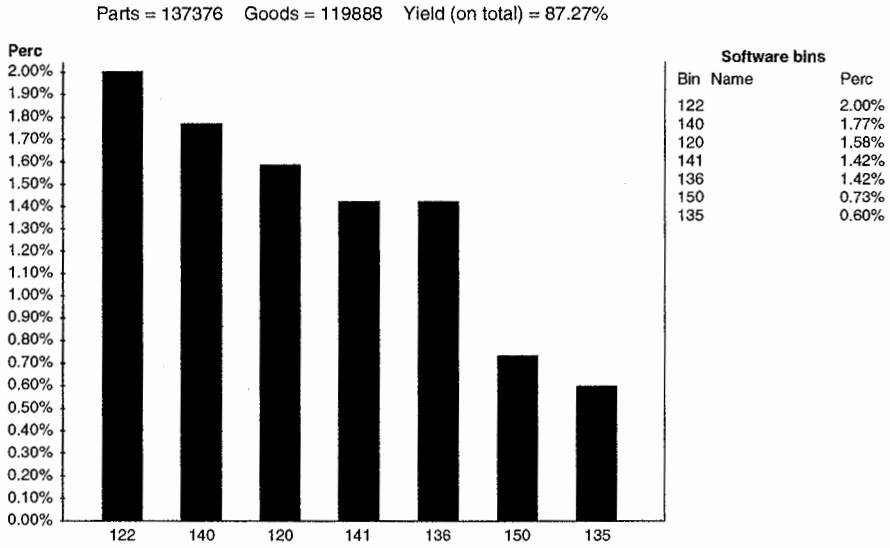


Fig. 13.13. Failure probability for a device subjected to a test flow indicated as a sequence of bins

There are two lines showing the average number of resources used after a specific test, split in up resources and down resources. Tests 5–25 haven't an impact on the number of surviving dice, because redundancy is substituting all the defective bitlines.

For example, test 25 has the same number of surviving dice compared to test 24, but it requires 10 more up resources and 10 more down resources. Test 30 has a big impact on the number of surviving dice, but it also requires a significant increase in the number of redundancy substitutions. It also introduces a topological difference between up resources that are more used and down resources. Up and down resources had the same behaviour until test 30.

Finally, this kind of graph underlines some aspects not evident during design phase, leading to more accurate behavioural and topological analysis.

In the following we will see how the substitution takes place during EWS and how fails are recognized.

For every device of every wafer the fail map is drawn. The map is basically a matrix where the defective position is indicated.

Now, we must understand if the defective bitline can be substituted, depending on redundancy constraints architecture or on redundancy resources number. If failures can be substituted, we must associate the defective position with the chosen resource.

In other words, if we want to substitute the fail bitline 3 with the resource 5, we have to force the device to read/write resource 5 every time the bitline 3 is addressed. This association is made only one time during the production phase and is valid for all the device life.

Petitioner Micron Ex-1021, 0367

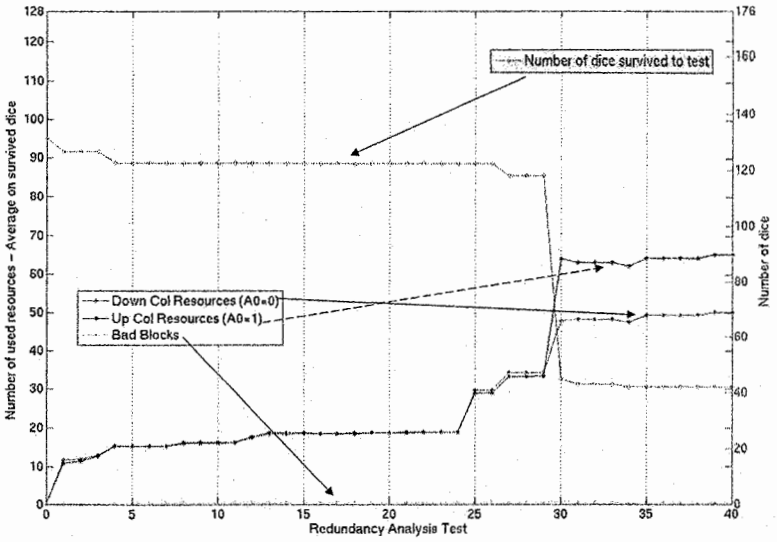


Fig. 13.14. Redundancy fails analysis during EWS

Basically, fuses, associated to redundancy circuitry, (see Sect. 13.5) are burnt in a permanent way, so that the association is never lost.

Once the fuses are burnt, previous tests must not have any fails, because now the device is able to associate redundancy resources to defective bitlines.

13.4 Redundancy architectures

There isn't a unique method to perform redundancy substitutions. The most suitable method depends on the memory architecture, on failure error rate and on microcontroller structure. Then, some area constraints can be imposed to redundancy circuitry. There could also be speed constraints to perform one substitution; this time can be very different in SLC or MLC case.

In multiplane devices usually substitutions are independent from plane to plane. In the following two different methods, one hardware and one software, will be described.

13.4.1 Realtime substitution

This substitution method is hardware based. Redundancy resources are not contiguous inside the matrix and they are mixed with data (Fig. 13.15). Therefore, there aren't bitlines more immune to topological defects and the failure probability

Petitioner Micron Ex-1021, 0368

is uniform. This is an important fact, because there aren't substitutions for the defective redundancy columns.

Redundancy resources can be split in different independent groups, depending on the matrix structure.

For example, suppose we have an architecture where row decoder is in the middle of the matrix, in this case the plane is divided into left half-plane and right half-plane. Redundancy resources, as well, would be divided in left and right groups. We can design these groups as completely independent or we can remove this constraint and use right resources for left half-plane or vice versa.

Redundancy resources can be divided into up and down groups, too. Also in this case, it is possible to choose between independent groups or shared groups.

It is important to know if the device reads/writes one byte or one word per time, since the substitution is realtime during read or program and we must know how many bits are addressed at the same time. Hardware-based method substitutes n bits for every word addressed. Given:

- *Lword* the addressed word length (8 or 16 bits)
- b the number of bits substituted for every word
- *nRS* the number of available resources

the method is able to substitute n bits every *Lword* bits, till a maximum number of *nRS* substitutions.

Let's take Fig. 13.16 as an example.

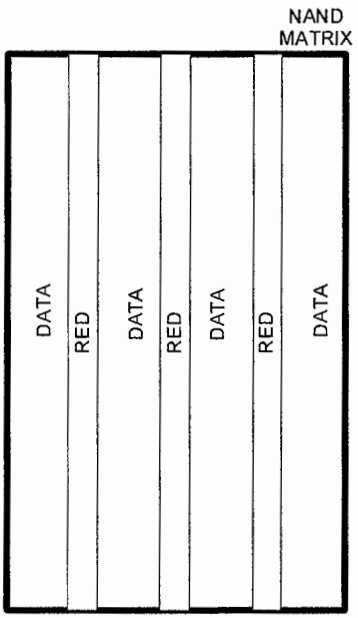


Fig. 13.15. Redundancy resources location inside matrix

Petitioner Micron Ex-1021, 0369

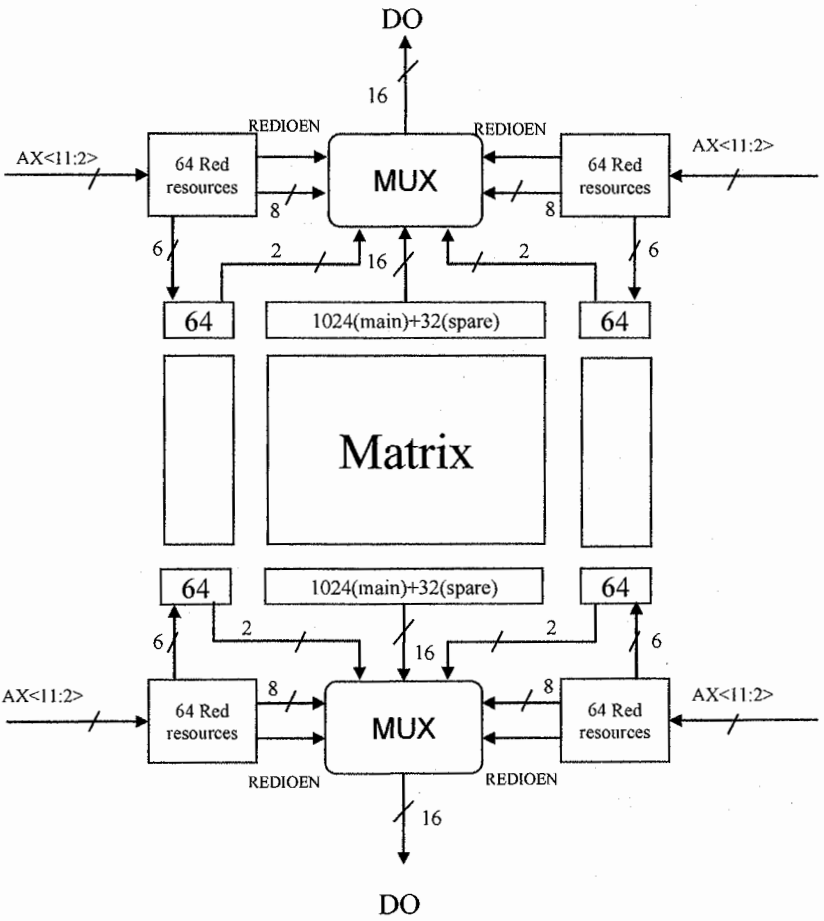


Fig. 13.16. Architecture for a hardware-based redundancy substitution

In this example the matrix is made up by 2,048 bitline pairs. There are 256 redundancy resources split in four groups of 64 resources each. There are only two MUX, one in the upper part and one in the lower part, so that left and right resources are shared, while up and down resources are independent.

The four blocks in the corners contain the addresses of the columns that must be substituted. Bitline address is an input of these blocks. During every read or program, we verify if the input address is equal to one of the addresses stored in these blocks. A match means that that bitline must be substituted and a signal REDIOEN is enabled.

We can suppose that the address of the bitline 100 is the one stored at the fifth row of the block in the up-left corner. Once we read or write bitline 100, REDIOEN signal is enabled and the fifth resource in the up left group is addressed. An address can enable more than one redundancy resource, depending on how many

Petitioner Micron Ex-1021, 0370

bits can be substituted for every word. In the example, it is possible to substitute 4 bits (two from left group and two from right group) for every word composed by 16 bits.

Given that structure, it follows that, before data out or before writing in matrix, the inputs of the MUX are the 16 bits addressed by the operation, 2 bits from left redundancy resources and 2 bits from right redundancy resources. Other inputs are the REDIOEN signal, that indicates if the redundancy substitution is enabled and two signals (one from left and one from right) that indicates the bits that must be substituted. The multiplexer makes the substitution and data out is composed by 16 bits.

The substitution of b bits for each word is a constraint that has a big impact on the failure probability due to lack of redundancy resources, in particular when we analyze single cell fails. In order to understand the mathematical computation of this probability, we must define the parameters involved:

- *NB*, i.e. the number of bitline pairs in the device
- *NC*, i.e. the number of cells in the device

The first computed quantity is n, that is the number of cells in a bitline pairs, i.e. the number of cells substituted during each redundancy replacement. This variable is computed as

$$n = \frac{NC}{NB} \tag{13.2}$$

Given x the single cell error probability, the error probability p of a bitline is

$$p = 1 - (1 - x)^n \tag{13.3}$$

Other fundamentals parameters are:

- *Lword*, i.e. the addressed word length, i.e. 8 or 16 or 32 bits.
- *Nblock*, i.e. the number of the independent groups in which redundancy resources are split. In the example of Fig. 13.17, *Nblock* is equal to two, since left and right resources are shared.
- *Nword*, i.e. the number of words in every group.
- *nRS*, i.e. the number of redundancy resources in every group.

Given those parameters, it is possible to compute the probability p_0 that a word doesn't need substitution:

$$p_0 = (1 - p)^{Lword} \tag{13.4}$$

The probability p_1 that a word contains one fail is

$$p_1 = Lword \cdot p \cdot (1 - p)^{Lword-1} \tag{13.5}$$

Petitioner Micron Ex-1021, 0371

The probability p_2 that a word contains two fails is

$$p_2 = \binom{Lword}{2} \cdot p^2 \cdot (1-p)^{Lword-2} \tag{13.6}$$

and so on, until the probability that a word contains the b errors that we want substitute.

Now, it is possible to compute the Chip Error Probability after redundancy for a device that substitutes 1 bit per word:

$$CEP_1 = 1 - \left(\sum_{i=0}^{nRS} \binom{nword}{i} p_1^i p_0^{nword-i} \right)^{nblocks} \tag{13.7}$$

while the Chip Error Probability after redundancy for a device that substitutes 2 bits per word is computed as:

$$CEP_2 = 1 - \left(\sum_{h=0}^{\left[\frac{nRS}{2}\right]} \binom{nword}{h} p_2^h \left(\sum_{i=0}^{nRS-2h} \binom{nword-h}{i} p_1^i p_0^{nword-h-i} \right) \right)^{nblocks} \tag{13.8}$$

and so on, until the substitution of b bits per word.

Figure 13.17 shows the different Chip Error Probabilities with different parameters.

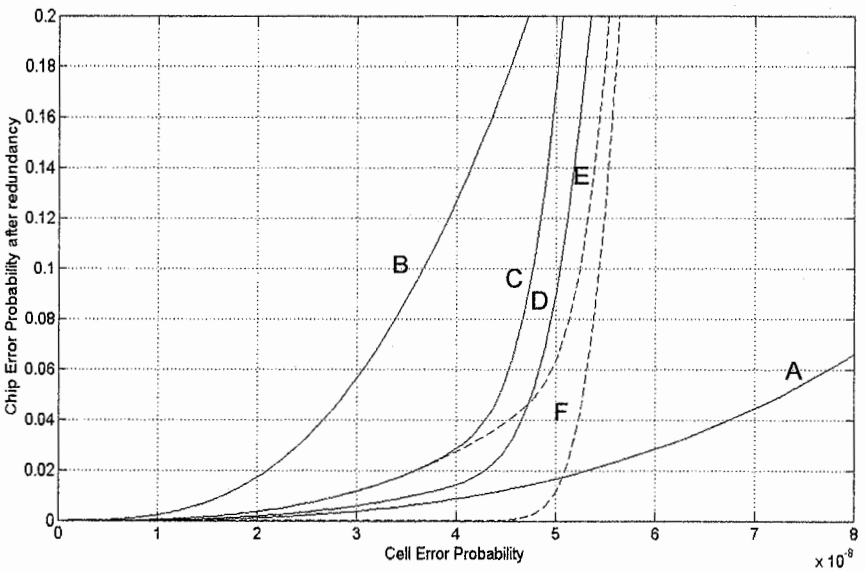


Fig. 13.17. Chip Error Probability after redundancy with different parameters

Petitioner Micron Ex-1021, 0372

The devices taken into account have the same size, except for the device whose CEP is labelled as "D", which has a size four times smaller and the device whose CEP is labelled as "A", which has a size twice smaller.

It is possible to see the effect of size on CEP, by comparing the solution "D" and the solution "B" that use the same substitution scheme, i.e. 2 bits on a word. The number of resources of the "B" scheme is doubled, but the failure probability is much higher.

Even the lines "A" and "E" have the same substitution method (2 bits on a byte), but the scheme "E", corresponding to a bigger device, underlines a higher failure probability.

However, in order to better understand the parameters effect on CEP after redundancy, it is more interesting to analyze the probabilities referring to the same size device (i.e. lines "B", "C", "E" and "F").

First observation regards the number of bits on which we perform substitution: as *lword* gets smaller, failure probability gets lower. In order to see this effect, it is possible to compare schemes "E" and "B", i.e. the substitution of 2 bits on a byte and 2 bits on a word respectively. Scheme "E" is much better, even if the number of redundancy resources is the same. Probability "E" shows another interesting behaviour: it seems composed by two different parts with different slopes. We have this behaviour because, for smaller probabilities, the fail is mainly due to the chosen scheme (2 bits on a byte or 2 bits on a word), while the number of resources doesn't count since the substitutions aren't frequent. At one point, the graph changes its slope: we have reached the point where the number of resources, instead the used scheme, has a big impact on probability and we begin to feel the lack of them.

The increase in the number of resources would move the slope-changing-point towards higher probabilities, while the introduction of a new constraint would move that point toward smaller probabilities. Solution "C" uses the same scheme as solution "E", but there is the constraint of not sharing left and right resources. The lines have the same behaviour for small probability but the scheme "C" begins to feel the lack of resources before the scheme "E".

Summarizing, from a mere probabilistic point of view, the best scheme is the one used for the solution "E" where *Lword* is equal to 8 and we don't have sharing constraint.

Anyway, it is not possible to use for redundancy an *Lword* different from the bus width the device uses. How is it possible to re-gain the lost failure probability if *Lword* is 16 bits long?

First of all, it is possible to increase the number of bits substituted per word as the scheme "F" shows. It represents the CEP after redundancy for a method that substitutes 4 bits per word. In case of smaller probabilities the CEP is much better, since it is possible to substitute defects that other schemes weren't able to substitute. However, also here we find the point of changing slopes. We can move this point doubling the number of resources. Anyway, the main problem is that the second part of the graph, where the lack of resources becomes dominant, is almost vertical and it is not possible to work on this.

Petitioner Micron Ex-1021, 0373

Finally, it is not possible to decide which is the best solution. Obviously, it is necessary to have an accurate analysis, since there are a lot of parameters to take into account. Surely, doubling the number of resources as the device size doubles is not the right choice. On the contrary, if we don't choose the right scheme, doubling the number of resources takes to doubling the area occupied by resources, without any benefits on the failure probability after redundancy.

There are memory architectures where bitlines are cut. In those architectures even redundancy resources are cut. From a probabilistic point of view, this is equivalent to doubling the bitlines and doubling the resources as well. Even in these case we can't conclude on the best solution and an accurate analysis of all the constraints is a must. Anyway, we can state that here we "waste" less redundancy cells when the fail is due to single defective cells, because we substitute only half bitline pairs, instead of the whole bitline pairs.

The idea of substituting only some part of bitline leads us to the approach described in the next section, which is not hardware-based anymore, but software.

13.4.2 Software substitution

The previous method is not convenient in single cell fails case, because it uses a bitline-pair resource to substitute a defective cell. In that case, it should be better substituting only the bitline portion the defective cell belongs to.

The following method gives the possibility to divide matrix in many strips (Fig. 13.18) so that each strip has independent substitutions.

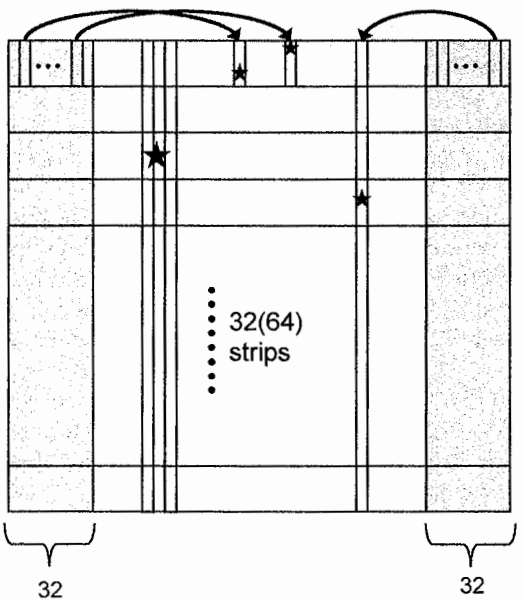


Fig. 13.18. Matrix divided in 32 strips useful for software-based redundancy architecture

Petitioner Micron Ex-1021, 0374

Obviously, a bitline short needs the substitution of the whole bitline, while single cell fail uses only a part of the redundancy resource.

Suppose Fig. 13.18 represents a matrix split in 32 strips with 64 redundancy resources. Every strip has 64 resources and can execute substitutions independently from other strips. It follows that it is possible to have $64 \times 32 = 2{,}048$ substitutions for every plane, which is a higher number compared to previous method, with less redundancy resources.

By the way, it is not possible to have a simple hardware structure (based on MUX) and a realtime substitution during data in and data out is no more available.

This approach is software-based and there isn't an impact on data in and data out: substitutions are executed when data is already stored in page buffers. Every substitutions is stored in fuses or in a non-volatile memory and is performed during reading and writing busy time.

In this method the fuses structure is ROM-like. During every reading or writing, the address y that has to be substituted is read from this ROM. This information is used to remap data written by the user in redundancy area. This approach increases busy time in reading or writing and needs to be careful analyzed.

Anyway, it has a higher number of substitutions available and more flexibility. A device with 32 strips and 64 redundancy resources can be easily reconfigured in 64 strips and 32 redundancy resources. In this way we can have a substitution scheme during ramp up production phase, where there is a higher number of fails and a substitution scheme during maturity phase, where busy time is a critical parameter, especially in SLC devices.

The number of independent redundancy resources block depends on the architecture chosen and on the way software is handled. If the substitutions are performed by a central block like microcontroller, it is possible to have not constraints at all. In that case, it should be possible to share redundancy resources among planes, even if this is not easy from circuital and architectural point of view. Anyway, if microcontroller has to perform redundancy substitutions, it has to handle a big amount of information and that could be very hard. First of all, it must to be able to address every redundancy resources, then it must have some custom operations to read from fuses and execute substitutions. As explained in Chap. 7, the introduction of new microcontroller operations is not easy, because we could be obliged to increase opcode length. Additionally, the information amount the microcontroller must be able to handle increases as the device size increases.

In multiplane devices with bitline cut structure it is easier to split redundancy resources in independent blocks. Substitutions are performed by finite state machines. With this structure, we introduce a constraint that worsens failure probability due to redundancy.

A structure composed by four different machines is able to perform four substitutions at the same time. This is particularly useful, if dealing with a device with busy time as critical parameter (typically NAND SLC). By using this approach, busy time is reduced by four times compared to a device with redundancy performed by microcontroller.

Petitioner Micron Ex-1021, 0375

Before going on, we have to underline that less constraints enables us to reduce redundancy resources (and gaining area) but the number of substitutions and so the fuses (with their area) increase substantially.

The redundancy architecture depicted in Fig. 13.19 always substitutes even and odd bitlines, it has 128 bitline resources and a fuses table composed by 2,048 rows and 17 columns. In case of byte substitution, i.e. 16 bitlines at the same time, the table is reduced to 2,048 rows and 14 columns but the number of resources needed is 1,024.

In the following we will describe the structure of Fig. 13.19, where redundancy resources are split in four independent groups and substitutions are performed by finite state machines.

First of all, we must understand if we want to substitute bits (one bitline even and one odd) or bytes.

All the involved parameters are analysed in Tables 13.1 and 13.2.

The two solutions are equivalent with the exception that the first substitutes the whole byte, while the second one only the bit.

The number of independent modules nB is the same as the number of strips nS. The number of bitlines substituted at the same time is 16 on a byte-based architecture and 2 if the substitution is performed one bit at a time.

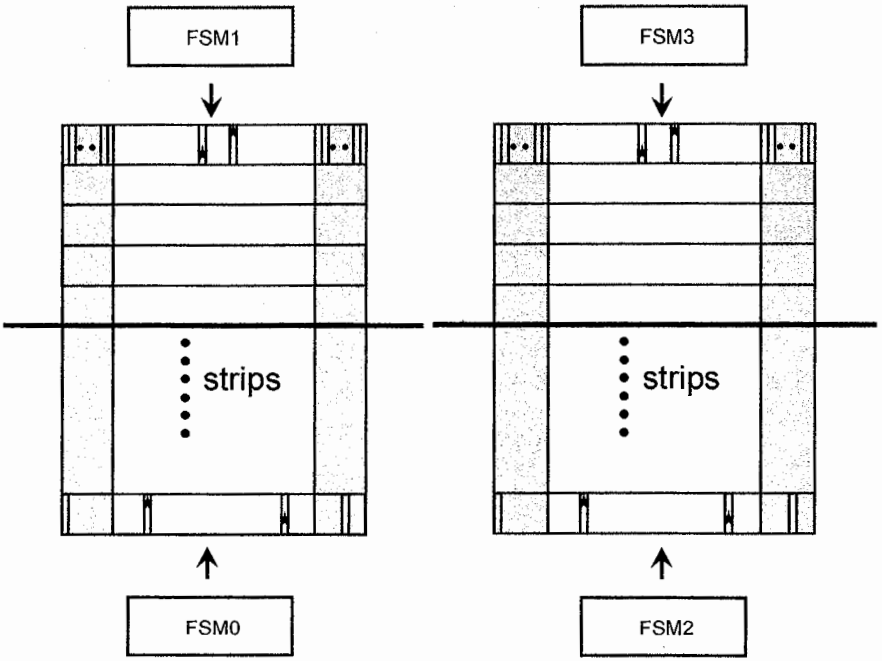


Fig. 13.19. Dual plane device with bitline cut structure. Redundancy resources are split in four independent blocks and substitutions are performed by finite state machines

Petitioner Micron Ex-1021, 0376

Table 13.1. Bitline resources and BL substituted as a function of bit-based or byte-based substitution

Solution	N modules	BL resources	Strips	BLs substituted
Byte	*nB*	8 *nBL*	*nS*	16
Bit	*nB*	*nBL*	*nS*	2

Table 13.2. Failure probability and fuses number as a function of bit-based or byte-based substitution

Solution	CEP BL shorts	CEP single cell fail	Time for substitution	N fuses
Byte	*y*	*x*	*t*	$Add \cdot nB \cdot nS \cdot nBL/2$
Bit	*y*	*x*	*t*	$(Add+3) \cdot nB \cdot nS \cdot nBL/2$

The total number of bitline resources is *nBL* on a bit-based structure (*nBL* = 128 if 64 substitutions are available) and eight times *nBL* on a byte-based structure. Therefore, from this point of view the substitution of a byte leads to a eight times bigger resources area compared to bit-based architecture.

The failure error probability is the same in the two cases, either if we consider single cell fails or bitline short fails.

The time per substitution is the number of clock pulses needed to execute a replacement and depends on device architecture and on how substitution blocks are designed. Anyway, it is certainly easier to substitute a whole byte, because the device operates with a bus whose width is a multiple of byte. Instead, a bit substitution requires more time or more area in the logic block that executes the replacements. Summarizing, from this point of view, byte substitution is more convenient.

Another parameter that involves area is the number of fuses needed, equal to:

$$N_f = Add \cdot nB \cdot nS \cdot \frac{nBL}{2} \tag{13.9}$$

in byte-based structure, where:

- *Add* is the address length needed, in order to address every byte, e.g. if the device is composed by 2,048 bytes, *Add* is equal to 11.
- *nB* is the number of independent blocks in which redundancy resources are split.
- *nS* is the number of strips in which the matrix is divided.
- *nBL*/2 is the number of available substitutions for every strip (division by 2 is due to the fact that we substitute an even and an odd bitline at the same time).

Petitioner Micron Ex-1021, 0377

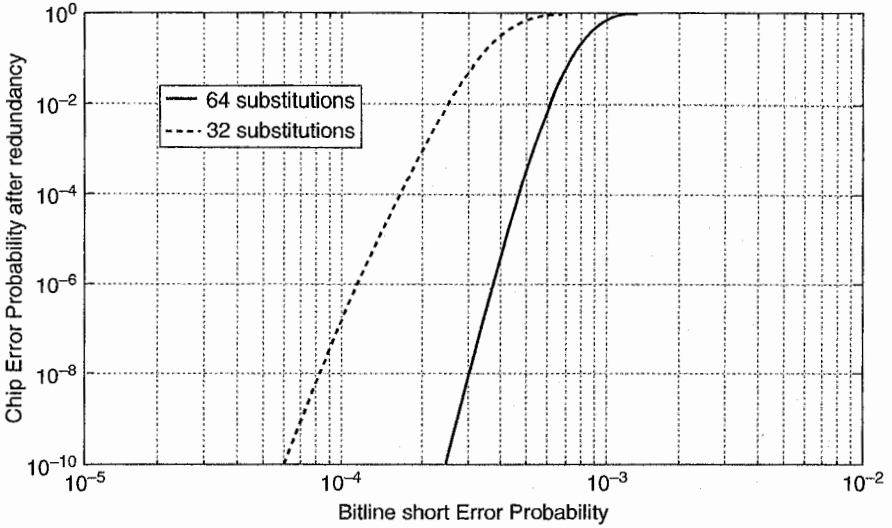


Fig. 13.20. Chip error probability due to bitline shorts varying the number of substitutions

The only difference in case of bit-based substitution is on *Add*, because now we must address the bits, e.g. for a device composed by 2,048 bytes, *Add* is equal to 14.

It follows that the area occupied by fuses is higher in case of bit substitution. It is not possible to state how much bigger is, because it depends on the device size, although it can reasonably be around 20–30%.

Finally, also in this case, the choice must be based on the device we are dealing with.

We have just seen that bit-based instead of byte-based substitutions haven't an impact on failure probability, but the change of other parameters can drastically affect this probability.

The number of independent resources blocks is a constraint that can increase or decrease failure probability as seen in the previous section.

Figure 13.20 shows different failure probability depending on the number of strips or on the number of substitutions per strip. The device taken into account is a dual plane 16 Gbit with bitline cut structure, where redundancy resources are split in four independent blocks. The page is composed by 4,096 bytes and the substitution is bit-based (two bitlines: one even and one odd).

Figure 13.21 shows the failure probability due to single cell fails for the same device. The number of strips is essential in this case, so that there are represented the lines regarding 64 and 32 substitutions with 64 and 32 strips.

- Fixing the number of resources, we observe that the increase in the number of strips guarantees a better probability. Anyway, it must be noted that the increase in the number of strips leads to a higher number of fuses necessary to store the substitutions.

Petitioner Micron Ex-1021, 0378

- As shown by comparing the failure probability of 32 strips and 64 resources with the failure probability of 64 strips and 32 resources, fixing the number of fuses, a higher number of resources gives a better probability. However, it must be noted that the difference between these two lines is not so high.

Even in this case, we must make an accurate analysis of the specific device, because the shown probabilities are not parallel lines.

Once we have decided the number of strips and the number of substitutions per strip, we must design the finite state machines that perform substitutions. Before this, it is necessary to clarify how data are stored in fuses.

As already said in Sect. 13.2.2, defective data are recognized in EWS and a corresponding fuse is burnt, so that the information is stored in a permanent way in the device. In the next section we will see the structure of the fuses and how they are arranged in the device. At this point, it is important to understand that fuses contain a table with the addresses of defective bitlines. Let's now see how these tables are organized.

First of all, observe how the substitution due to bitline shorts is handled. A bitline short involves the whole bitline pair (or half in bitline cut case). Therefore, its address should be repeated for every strip, wasting a big amount of fuses. In order to avoid this waste, we split the table so that the first part contains bitline shorts substitutions, while the second part contains single cell fail substitutions.

General structure is represented in Fig. 13.22.

The device in figure is dual plane with bitline cut structure. There are four redundancy resources independent blocks with four finite state machines.

Every machine has associated two fuses groups containing redundancy information regarding the corresponding half plane.

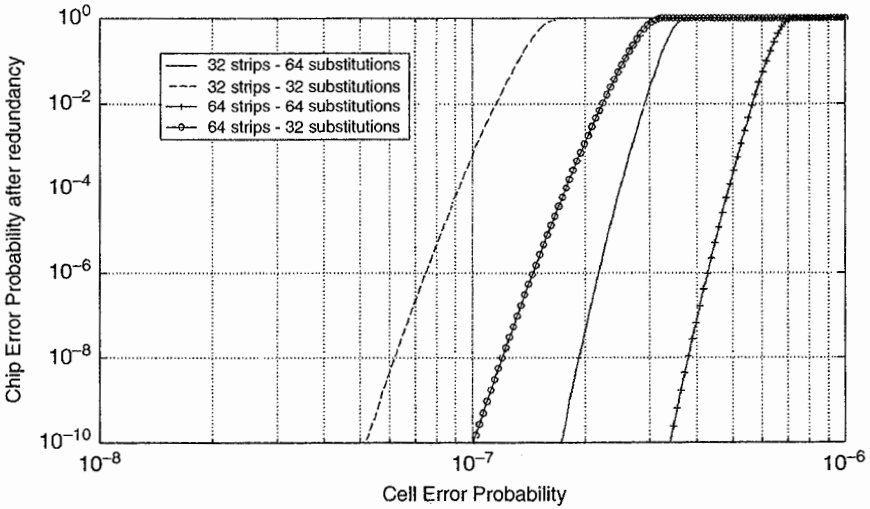


Fig. 13.21. Chip error probability due to single cell fails varying the number of strips and the number of substitutions per strip

Petitioner Micron Ex-1021, 0379

Fuses tables contain:

- Pointers table
- Bitline shorts table
- Strip tables: one for every strip. It has the information of the strip itself
- Bad block table
- Configuration table

Redundancy needs the first three tables.

The use of three tables instead of one guarantees a more compact information writing. As already said, the first gain is in storing information of bitline shorts. Then, observe that not all the strips require the maximum number of available substitutions. On the contrary, most of the strips require less substitutions and some strips doesn't require any substitution at all. For this reason, it is not convenient to keep 32 rows (if 32 is the maximum number of available substitutions) for every strip.

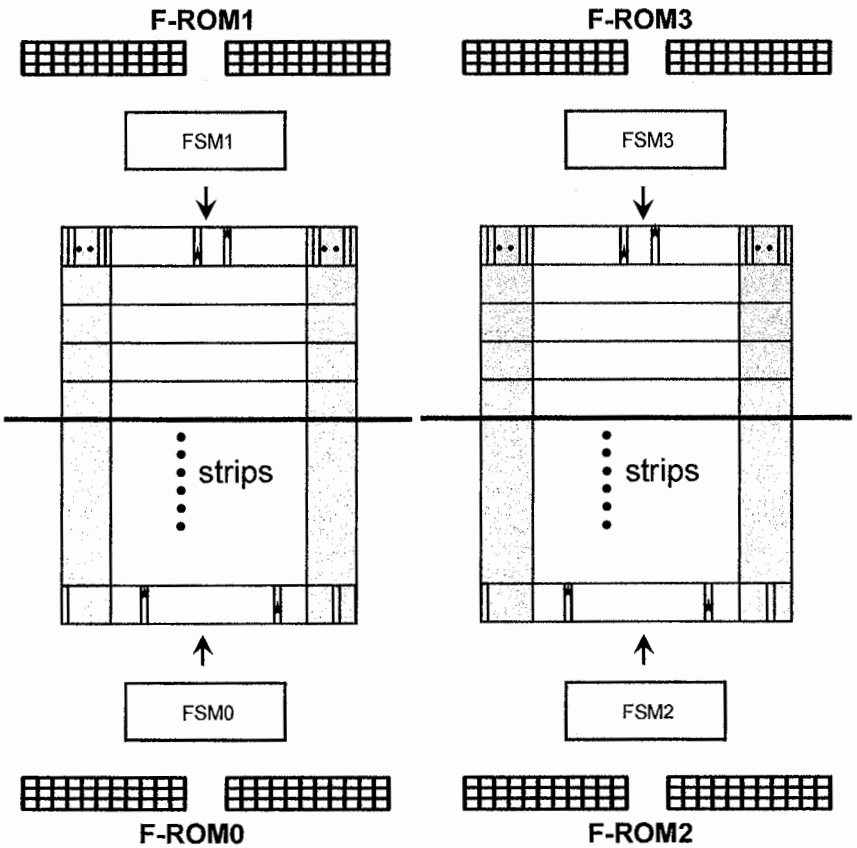


Fig. 13.22. Dual plane device with bitline cut structure: it is highlighted the fuses place

Petitioner Micron Ex-1021, 0380

Looking at Fig. 13.23 the tables are described below.

- Pointer table: it contains the starting address of the bitline table and the starting address of each strip table.
- Bitline table: it contains information about bitline shorts belonging the half plane.
- Strip table: it is composed by a number of tables equal to the number of strips.
- Every table contains information on strip substitutions.

Every row of the bitline and strip tables is a binary string that indicates the defective address and the position (right or left) of the defective bit.

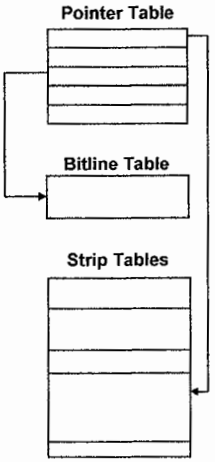


Fig. 13.23. Redundancy tables structure

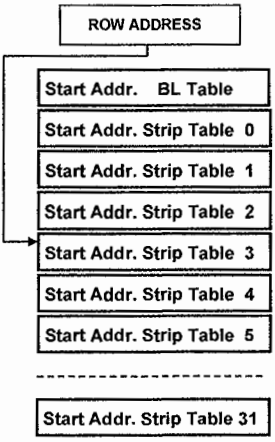


Fig. 13.24. Pointer table organization

Petitioner Micron Ex-1021, 0381

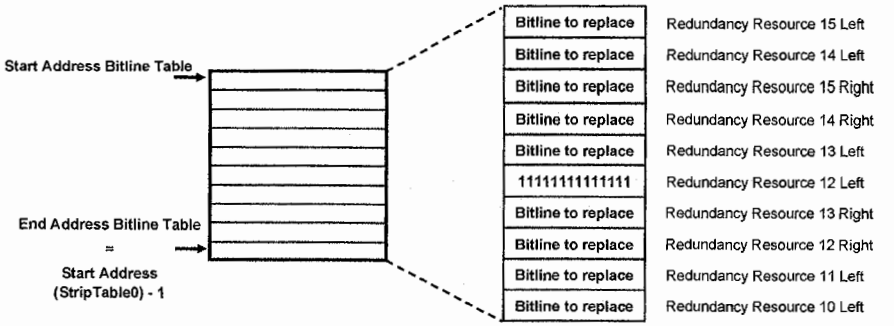


Fig. 13.25. Bitline table structure

With the help of Figs. 13.24–13.26 we will now describe how tables are handled and how finite state machines work.

Row address selects the address of the row to be read. The first row of the table is the starting address of the bitline table, while all other rows are the starting address of every table containing information related to strip.

Every time we execute substitutions on strip i, we begin reading the table containing information related to that strip, whose address is indicated in Pointer Table (*Start. Addr. Strip Table i*). The last substitution of strip i is the one indicated in row, whose address is (*Start. Addr. Strip Table* $i+1$)-1, i.e. it is the preceding row of the address of the successive strip.

Since the maximum available substitutions are not used by every strip, it could be possible having strips not using redundancy resources at all. In that case, the address of strip i is the same of the address of the strip $i + 1$.

Bitline Table contains substitutions due to bitline shorts. Since bitline shorts regard all the half-plane (or plane if bitline cut structure is not implemented), it is useful to group all the data in a single table. In this way, we don't have to rewrite the same data for all the strips.

Bitline Table is the first table, so that the first substitutions performed are the ones related to BL shorts. If we are dealing with a device with 32 redundancy resources, bitline table will have a maximum number of 32 rows. Assuming that all the substitutions are due to bitline shorts, there won't be any resources in case of single cell fails. The balance between bitline short substitutions and single cell fail substitutions can be estimated on used technology and on the maturity of the product. In that case, it is possible to reduce the rows number of the bitline table in respect to the maximum number of the half-plane available resources.

The starting address of the Bitline Table is indicated in the Pointer Table, while the last address of the Bitline Table is equal to the preceding address of the first strip.

Every row of the bitline table is the address of a pair of bitlines (one even and one odd) that needs substitution. A possible association between defective bitline-pair and redundancy resource is indicated in Fig. 13.25. Bitlines pair indicated in address 0 is substituted with the 15th resource on the left, while bitlines pair

Petitioner Micron Ex-1021, 0382

indicated in address 1 is substituted with the 14th resource on the left and so on, until the end of the table.

When a redundancy resource is defective, it is associated with bitline address 111...11. Since the all 1 address is generally not valid in the matrix, the substitution with a defective resource never occurs.

The structure of strip tables is very similar to the one of bitline table. As already said, the starting address of a strip table is indicated in the pointer table, while the last address of a strip table is the preceding one the starting address of the next strip table. The association is the one already shown and the same implementation is taken, in case of defective resources.

Now that it is clear how redundancy assignment is based on different tables, we can explain how area can be further decreased (in addition to bit/byte schemes).

Increasing the number of rows containing substitutions increases the area needed to store that information in fuses. In order to reduce the number of rows, we need to reduce the number of resources or the number of strips, taking us once again to the issues depicted in Tables 13.1 and 13.2.

By looking at Fig. 13.27, we try to find out the better scheme between one compact but with a not-so-good yield and one with a good yield but a higher area.

Virtual table concept enables us to have the yield obtained by *Sol1* with the area needed in *Sol2*.

The scheme uses the number of resources of *Sol1*, but the table is cut to a number of rows inferior to the maximum number of available substitutions and equal to the number of rows of *Sol2* (Fig. 13.28).

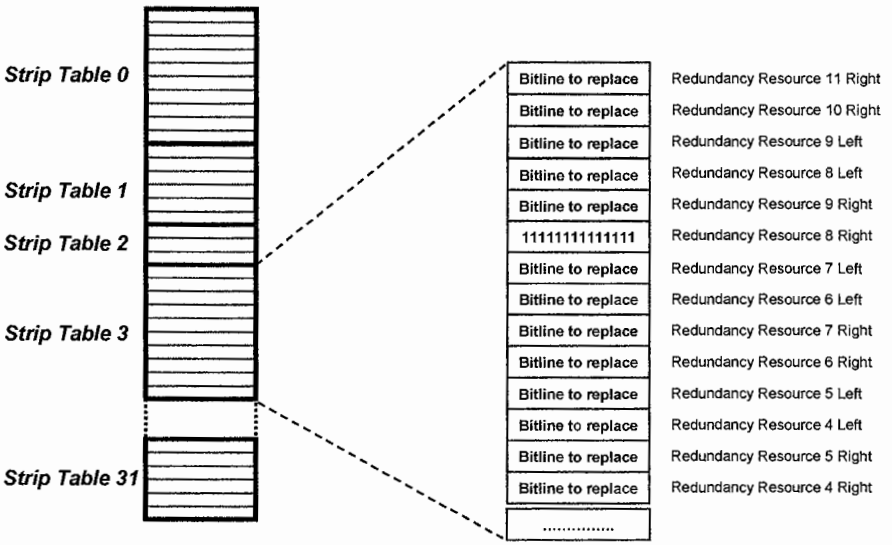


Fig. 13.26. Strip tables structure

Petitioner Micron Ex-1021, 0383

It is worth noting that the same yield of *Sol1* is to be intended in the product maturity phase when we don't use all the resources.

Substitutions during data out are shown in Fig. 13.29.

Row address enters the pointer table. This one indicates the table that must be read (bitline table or a strip table).

Let's now suppose that Strip Table contains the address of a defective bitlines pair. This address is associated with a fixed redundancy resource, e.g. the second on the right.

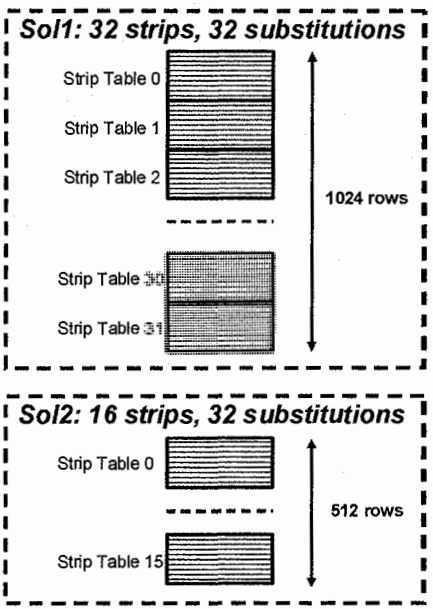


Fig. 13.27. Comparison between two schemes in terms of strips, resources and number of rows

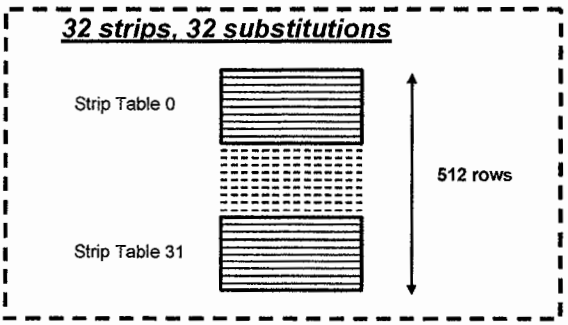


Fig. 13.28 Virtual table concept

Petitioner Micron Ex-1021, 0384

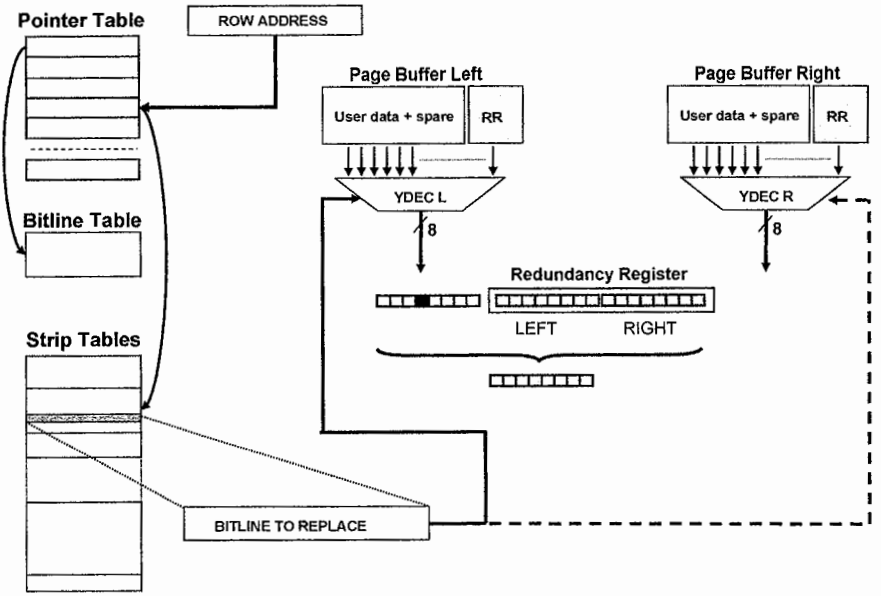


Fig. 13.29. Redundancy substitutions during data out

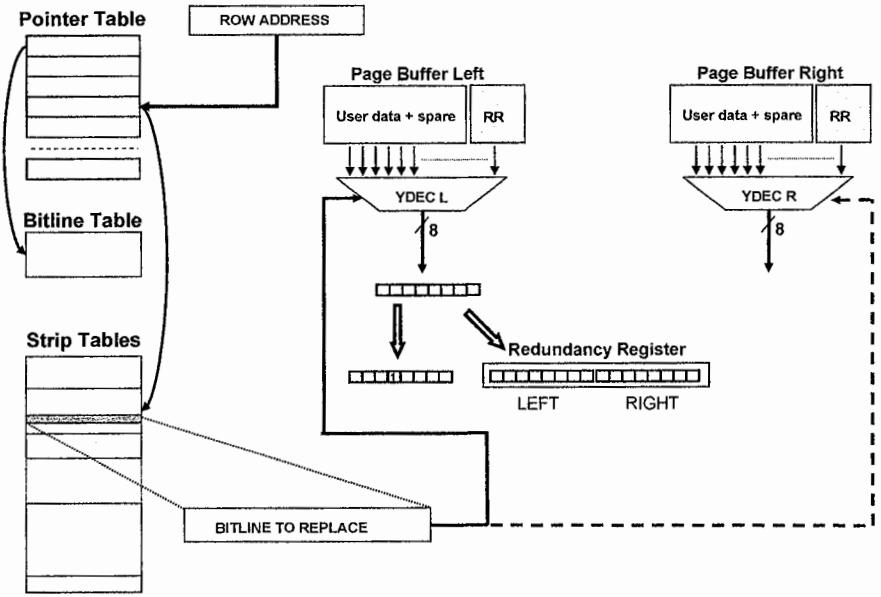


Fig. 13.30. Redundancy substitutions during data in

Petitioner Micron Ex-1021, 0385

The internal registers of finite state machine corresponding to that half-plane contain already the address of eight right redundancy resources and eight left redundancy resources. Bitlines pair address to be replaced inputs right YDEC. This MUX has already downloaded eight right data bits.

At this point, the redundancy machine substitutes the fourth bit of data register with the second bit of right redundancy resources.

We must observe that this finite state machine needs an update of the internal registers containing redundancy resources every 16 substitutions.

Redundancy substitutions during data in are shown in Fig. 13.30.

As already said for data out, defective bitline pairs are indicated in strip table beginning with the address read in the pointer table. The bitline to replace address inputs left YDEC if that bitlines pair must be replaced by a left resource. Let's now suppose that the third data bit must be substituted by first left redundancy resource. In the internal registers of the FSM it is stored the association, so that every time the third data bit is addressed, the first left redundancy resource is read instead.

13.5 Fuse ROM

Fuses contain information for the correct device functioning. FROM is a read-only memory, programmed via laser-beam during the device production phase. The information is stored in a non-volatile way and is permanent during the entire device's life. This information is composed by:

- Band-gap or circuit settings
- System settings
- Bad Block or Column Redundancy information

The column redundancy information reduction has a big impact on the reduction of fuses area, because 90% of fuses area is occupied by column redundancy information. The basic cell of a F-ROM (Fig. 13.31) is constituted by a fuse and a selector transistor. The writing operation consists in destroying the fuse in a permanent way. In this way, a F-ROM can be programmed only one time (in factory).

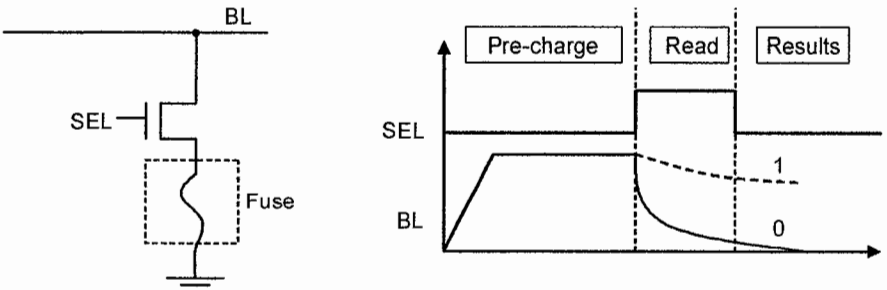


Fig. 13.31. Basic F-ROM cell and Read waveform

Petitioner Micron Ex-1021, 0386

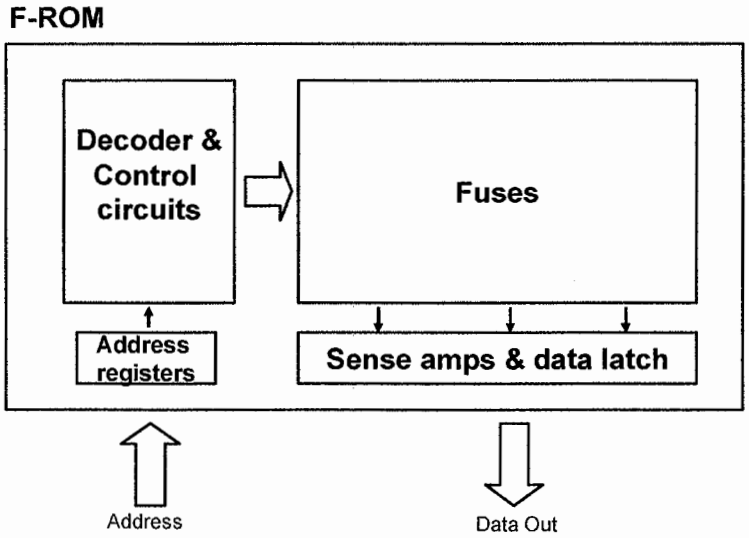


Fig. 13.32. Fuse ROM architecture

The only operation a F-ROM can execute during the device's life is a read operation. This operation consists in connecting the fuse to the precharged bitline (Fig. 13.31). A not-burnt fuse discharges the bitline to 0. A burnt fuse has a high impedance that keeps the voltage bitline very near to the precharged value, so that bitline is read as 1.

A F-ROM block scheme is sketched in Fig. 13.32. F-ROM is organized in the classical matrix way. Based on the selected address, decoders address the proper fuses. Sense amplifiers read fuses content, then digitalized as F-ROM data out.

There isn't a univocal way to design fuses. Figure 13.33 represents a polysilicon fuse and Fig. 13.34 is its section AA. Surrounding metal layers are used to isolate burning area, in order to avoid polysilicon parts going in neighbourhood zones.

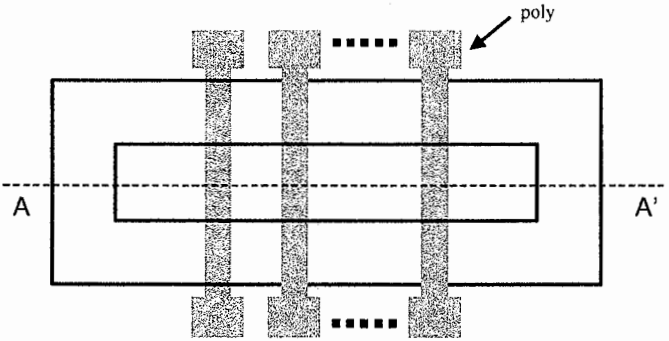


Fig. 13.33. Layout view of a polysilicon fuse

Petitioner Micron Ex-1021, 0387

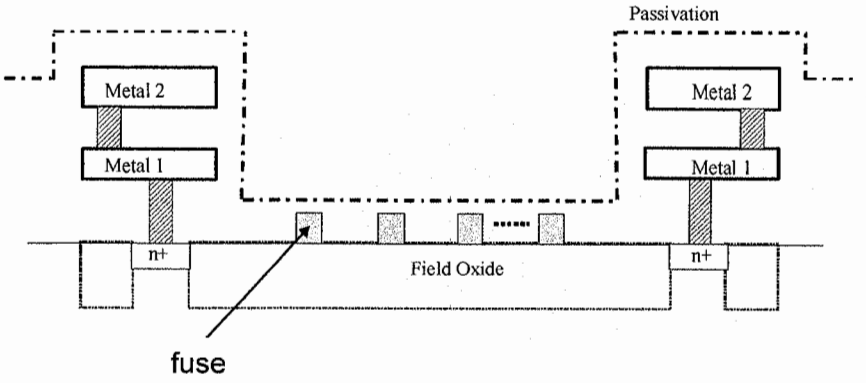


Fig. 13.34. Cross section of fuse in Fig. 13.33

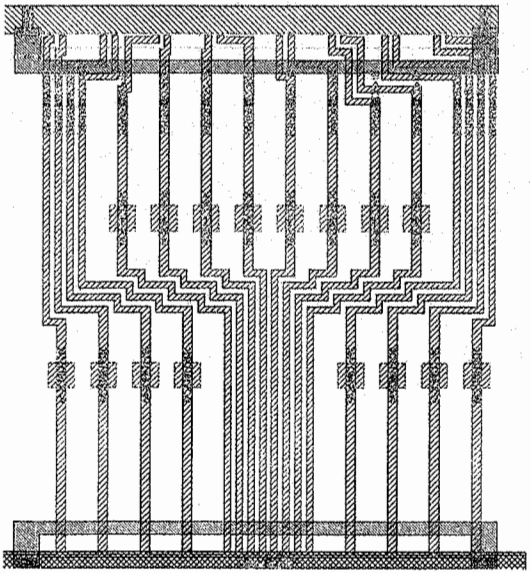


Fig. 13.35. Layout view of 16 metal fuses

Fuses can be also be realized in metal. Figure 13.35 represents 16 metal fuses organized in such a way to optimize area. The basic fuse word can be chosen and generally it is an easily addressable number. The basic cell in figure is made by 4 bits up, 8 bits down and 4 bits up and it is repeated N times. A number of blocks of Fig. 13.35 can be organized in a matrix as depicted in Fig. 13.36.

Fuses cannot be placed and filled up with information in a random way. First of all, we must be careful with their positioning inside the device, since it is not possible to have wires over F-ROM.

Petitioner Micron Ex-1021, 0388

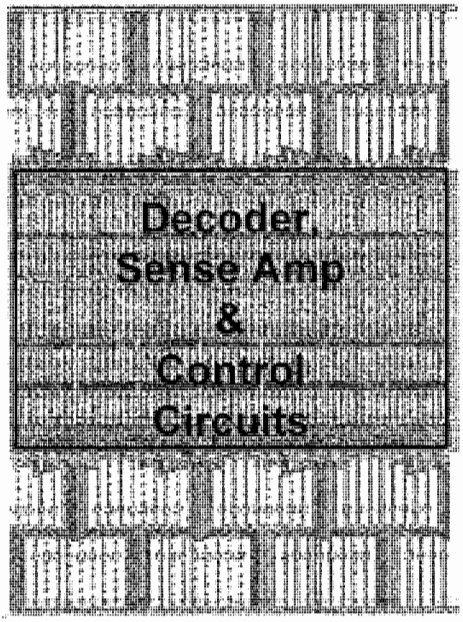


Fig. 13.36. Array of metal fuses (F-ROM)

Supposing to have the floor plan represented in Fig. 13.37 and the software redundancy structure explained in Sect. 13.4.2, it is useful to split fuses in four independent blocks.

These blocks can't be a continuous line in the near of the matrix, otherwise, they will avoid the addressing of the top and bottom parts of the device.

Therefore, every block is slip in right and left part and the four FSMs handling the half-plane substitutions are placed in the center.

If the number of substitutions concerning an half-plane cannot be contained in only one right line and one left line we have to double the number of lines.

Fuses are logically seen as an information table (Fig. 13.38).

The figure represents the half-plane fuses table for a device adopting the software redundancy substitutions explained in Sect. 13.4.2, with 32 strips, 32 redundancy resources and the virtual table option. The table has 16 columns. The first part is made up by 64 rows containing bad blocks (BB) information. In other half-plane tables these rows can contain the Parameter Page instead of the bad block information. The pointer table is contiguous to the bad block table and has the features already depicted. Rows indicated as CNF contain data needed for the device settings or circuitry settings, the number of CNF rows depends on the device itself. The last rows contain bitline table and strip tables constituted by 32 and 512-32 rows respectively. Observe that, organized in this way, the top part of the table is always full of data, while the bottom part, containing redundancy information, can be partially empty.

Petitioner Micron Ex-1021, 0389

Anyway, fuses are not organized as a table, so we need to translate this logical organization in a physical way (Fig. 13.39).

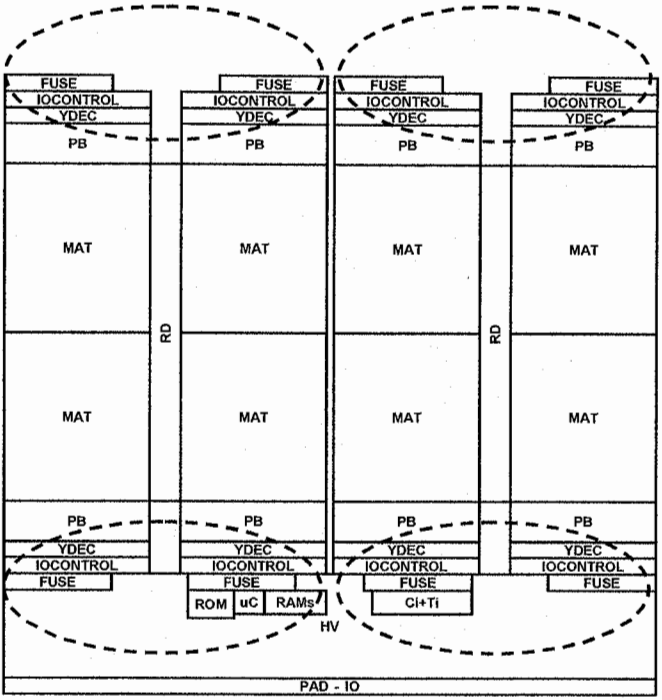


Fig. 13.37. A device floor plan, the dashed circles indicate where F-ROMs are located

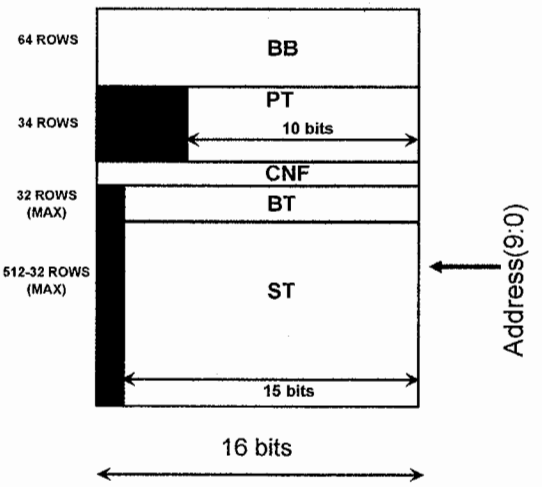


Fig. 13.38. F-ROM logical view

Petitioner Micron Ex-1021, 0390

The information concerning one half-plane are organized in four right rows and four left rows. The control logic is placed between the first rows pair and the second rows pair. This organization follows the way the laser machine works. When the laser machine doesn't have to burn a fuse, it passes over fuse without performing any operation. Besides, in order to reduce fusing time, we can group together a number of fuses that don't have to be burnt. The laser machine proceeds by passing over rows 00L-00R-01R-01L-10L-10R-11R-11L so the proposed organization groups all the potentially empty fuses together.

The empty fuses can be used to store some algorithm parts or some particular features required during the device maturity life, when most of the redundancy resources won't be used.

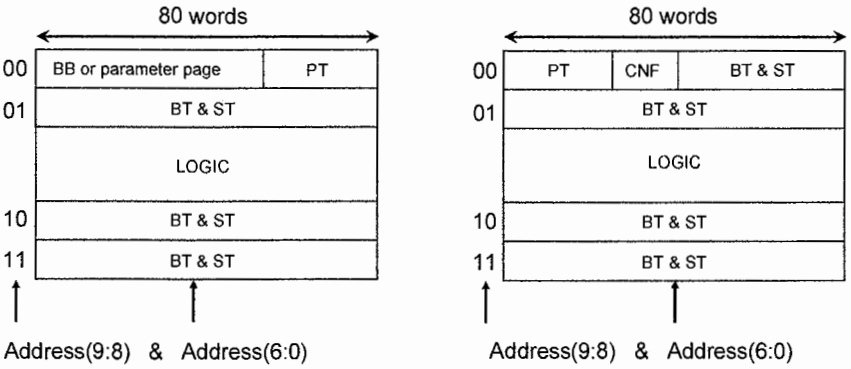


Fig. 13.39. F-ROM organization for an half plane

13.6 NAND block as OTP for redundancy data concept

The fails substituted by redundancy are mainly due to technological issues. Therefore, as technology shrinks, the number of fails increase. Besides, as the device size grows and as the technology shrinks, the number of redundancy resources and the number of fuses needed to store the information increase a lot. Unfortunately, the fuse size doesn't decrease with technology, so that the ratio between matrix area and device area will became more and more disadvantageous.

We can overcame this limitation by storing redundancy information in an OTP-like block instead of fuses [3]. OTP structure is based on non-volatile memories, so it decreases with the matrix as technology shrinks. During device power on, all the information, already stored in the OTP block, are transferred in the circuitry configuration latches or in the device RAM (redundancy information). Theoretically, this transfer is not necessary and the information can be read from the OTP block every time is needed. Due to the NAND matrix structure, every block suffers from disturbs coming from all other blocks erase. OTP block can't be reprogrammed, so, during all the device's life, it will suffers erase disturb. It follows that, at some

Petitioner Micron Ex-1021, 0391

point, the information won't be correct anymore and couldn't be used. This is the reason why it is useful to transfer the information during power on.

We must carefully choose OTP block inside the matrix, because it contains essential information for the device functioning. First of all, OTP block can't be a fixed block, e.g. block 0, because that block could be bad. The best block, in terms of parameters, can be chosen as the OTP block. Once the OTP block is chosen during EWS phase, it is programmed with all the necessary information and it is marked as bad in order to avoid its addressing for a write or an erase operation.

The address of the OTP block, needed by the memory, can be stored in fuses. In this way, the amount of necessary fuses is only the ones containing the OTP block address. During power on, the device reads the OTP address from fuses and then reads the OTP block in order to transfer and store all the parameters.

Observe that we must read the OTP block without the read parameters, since they are stored in the block itself.

Anyway, the read operation must be performed every time in the same way, without the reading parameters stored inside the OTP block and the information must be the most reliable possible. For these reasons, also the write operation (performed during EWS) must be executed with proper parameters. Threshold distributions must be as far away as possible, in order to be able to perform the reading with relaxed voltages and timings. In addition, it is worth to write in a SLC way the OTP block, if we are dealing with a MLC device.

Moreover, redundancy substitution is disabled, because column redundancy information isn't available during this read.

In order to be able to guarantee the reliability of the information stored in the OTP block, single cell fails or bitline shorts inside the block itself must be corrected with an error correction scheme. This particular ECC handles only this block. Therefore, we require a low circuital complexity even at the cost of a high number of parity bits. During EWS screening, the possible OTP blocks can only be those ones with a number of errors less than the ECC correction capability.

Figure 13.40 shows different ECC schemes with different error correction capabilities, different circuital complexity and different number of parity bits.

Let's now suppose to have 6,144 bits of information that need ECC protection, some of the possible schemes are:

- A majority code 1 over 3. Every bit is written three times and the correction capability is 1 bit over 3 bits. If we suppose to have a 16-bit word, the code requires 19,664 bits, it has rate 5/16 and a minimal computational complexity.
- A majority code 2 over 5. Every bit is written five times and the correction capability is 2 bits over 5 bits. If we suppose to have a 16-bit word, the code requires 32,768 bits, it has rate 3/16 and a minimal computational complexity.
- A majority code 3 over 7. Every bit is written seven times and the correction capability is 3 bit over 7 bits. If we suppose to have a 16-bit word, the code requires 49,152 bits, it has rate 2/16 and a minimal computational complexity.
- A majority code 7 over 15. Every bit is written 15 times and the correction capability is 7 bit over 15 bits. If we suppose to have a 16-bit word, the code requires 98,304 bits, it has rate 1/16 and a minimal computational complexity.

Petitioner Micron Ex-1021, 0392

- An Hamming code corrector of 1 bit over 15. Every word contains 11 information bits and four parity bits. If we suppose to have a 16-bit word, the code requires 8,944 bits, it has rate 11/16 and a higher computational complexity.
- A linear block code corrector of 2 bits over 15. Every word contains seven information bits and eight parity bits. If we suppose to have a 16-bit word, the code requires 14,048 bits, it has rate 7/16 and an even higher computational complexity. In this case, the decoding operation is represented by a matrix containing all the possible syndromes for one or two errors.
- A linear code corrector of 3 bits over 15. Every word contains three information bits and 12 parity bits. If we suppose to have a 16-bit word, the code requires 32,768 bits, it has rate 3/16 and a very high computational complexity.

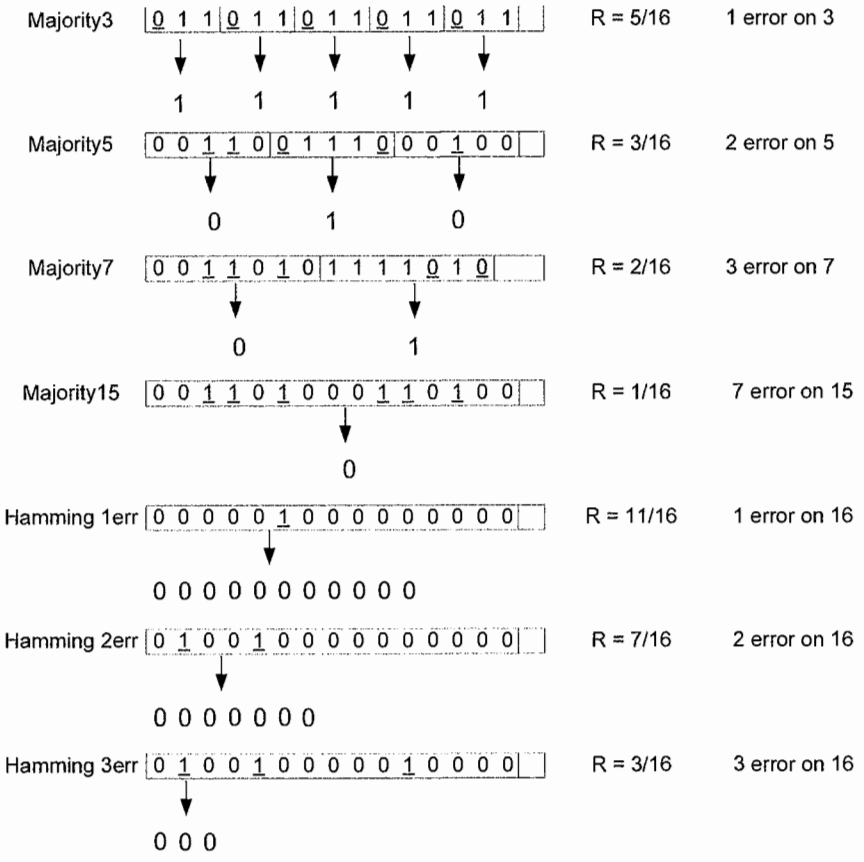


Fig. 13.40. Possible ECC schemes for the OTP block

Petitioner Micron Ex-1021, 0393

Figure 13.41 represents the probability of error for the proposed schemes.

If the cell error probability is near 10^{-3}, the single error correction (Hamming scheme or majority scheme) is not enough. If the cell error probability is less than 10^{-2}, the seven error correction code is too much.

As regarding all the other schemes, it is not possible to state which code is better. The choice depends on the number of parity bits involved (higher in majority codes) and on the computational complexity (higher in linear block codes).

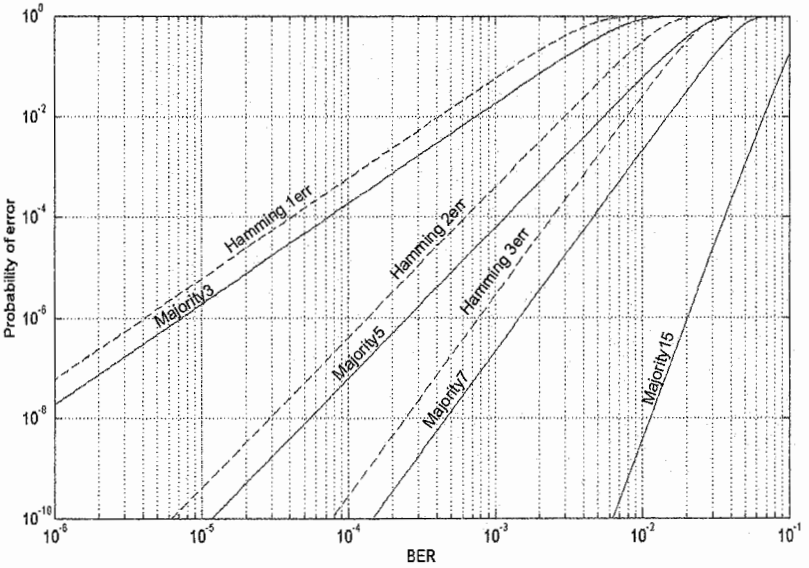


Fig. 13.41. Probability of error for the schemes proposed in Fig. 13.40

References

1. R. Micheloni, A. Marelli, R. Ravasio "Error Correction Codes for Non-Volatile Memories", Springer, 2008.
2. G. Campardo, R. Micheloni, D. Novosel "VLSI-Design of Non-Volatile Memories", Springer, 2005.
3. R. Micheloni, A. Marelli, R. Ravasio "Nand Flash Memory Device with ECC protected reserved area for Non-Volatile Storage of Redundancy Data", United States Patents US2008/0065937 A1, Mar 13, 2008.
4. Koji Sakui, Kang-Deog Suh "Nand Flash Memory Technology", Wiley 2008.
5. R. Micheloni et al. "A 4Gb 2b/cell NAND Flash Memory with Embedded 5b BCH ECC for 36MB/s System Read Throughput Solid-State Circuits Conference", Digest of Technical Papers ISSCC 2006 IEEE International, Feb. 2006, pp. 497–506.
6. R. Adrian Cernea, "Nonvolatile Erasable Flash NAND Memories", SanDisk Corporation, Santa Clara Valley (SCV), IEEE Solid State Circuits Society, April 2008.

Petitioner Micron Ex-1021, 0394

7. K. Takeuchi, T. Tanaka, T. Tanzawa "A Multipage cell Architecture for High-Sped Programming Multilevel NAND Flash memories", IEEE Journal of Solid state circuits, Vol. 33, No. 8, August 1997.
8. Lu Chih-Yuan, Lu Tao-Cheng, Liu Rich, "Non-Volatile Memory Technology-Today and Tomorrow", Physical and Failure Analysis of Integrated Circuits, 2006. 13th International Symposium on the , pp.18–23, 3–7 July 2006.
9. P. Cappelletti, C. Golla, P. Olivo, E. Zanoni, Eds., "Flash Memories". Boston, MA: Kluwer, 1999, ch. 5.
10. P. Cappelletti, R. Bez, A. Modelli, A. Visconti, "What we have learned on Flash memory reliability in the last ten years", Electron Devices Meeting, 2004. IEDM Technical Digest. IEEE.
11. M. Annaratone, "Digital CMOS Circuit Design". Boston, MA: Kluwer.
12. H. Arakawa, "Address Decoder Circuit for Non-Volatile Memory", United States Patents 5,039,882, August 13, 1991.
13. S. Aritome, "Advanced Flash Memory Technology and Trends for File Storage Application", IEDM Technical Digest, pp. 763–766, 2000.
14. S. Atsumi et al., "A 3.3V-only 16 Mb Flash Memory with Row-Decoding scheme", 1996 IEEE Int. Solid-State Circuits Conference. Digest of Technical Papers, pp. 42–43, February 1996.
15. R. Bez et al., "Introduction to Flash Memory", Proceedings of the IEEE, vol.91, pp. 554–568, 2003.
16. G. Campardo et al., "An overview of Flash Architectural Developments", IEEE Proceedings of the, vol. 91, No. 4, pp. 523–536, April 2003.

Petitioner Micron Ex-1021, 0395

14 Error correction codes

T. Zhang[1], *A. Marelli*[2] *and R. Micheloni*[3]

14.1 Introduction

As we have seen in Chap. 13, redundancy covers manufacturing defects while ECC takes care of the failures during the life of the device. This chapter deals with error correction codes applied to NAND Flash memories. In fact, when the memory is placed in its final application, different reasons for errors (see Chap. 4) can damage the written information so that it could happen that the read message is not equal to the original anymore [1].

Before proceeding on, it is necessary to introduce some basic definitions.

The error probability p, also known as raw bit error rate (BER), is defined as:

$$p = \frac{Number\ of\ bit\ errors}{Total\ number\ of\ bits} \tag{14.1}$$

The Chip Error Probability (CEP) is defined as:

$$CEP(p) = \frac{Number\ of\ chip\ errors(p)}{Total\ number\ of\ chips} \tag{14.2}$$

Considering a memory chip formed by B blocks of A bits, the CEP before ECC (CEP_{in}) is given by:

$$CEP_{in}(p) = 1-(1-p)^{AB} \tag{14.3}$$

Assuming to use a block correction code of t errors, the CEP_{out} is

$$CEP_{out}(p) = 1-(1-b)^{B} \tag{14.4}$$

where b is the failure probability of the block to which the ECC is applied

$$b = 1-[P_0 + P_1 + P_2 + ... + P_t] \tag{14.5}$$

[1] Rensselaer Polytechnic Institute, tzhang@ecse.rpi.edu
[2] Integrated Device Technology, alessiamarelli@gmail.com
[3] Integrated Device Technology, rino.micheloni@ieee.org

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_14,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0396

P_i is the probability of i errors in a block of A bits

$$P_i = \binom{A}{i} p^i (1-p)^{A-i} \tag{14.6}$$

In conclusion we have that

$$CEP_{out}(p) = 1 - \left[(1-p)^A + \binom{A}{1} p(1-p)^{A-1} + \ldots + \binom{A}{t} p^t (1-p)^{A-t} \right]^B \tag{14.7}$$

Figure 14.1 shows the graphs of CEP_{in} (indicated as "No ECC") and CEP_{out} as a function of p for a system with 4 Gbit memory, 2 kB block and ECC able to correct one, two, three, four or five errors. Assuming a raw BER of 10^{-9}, 1-error ECC gives a Chip Error Probability of 100 ppm, while a 2-errors ECC recovers five orders of magnitude.

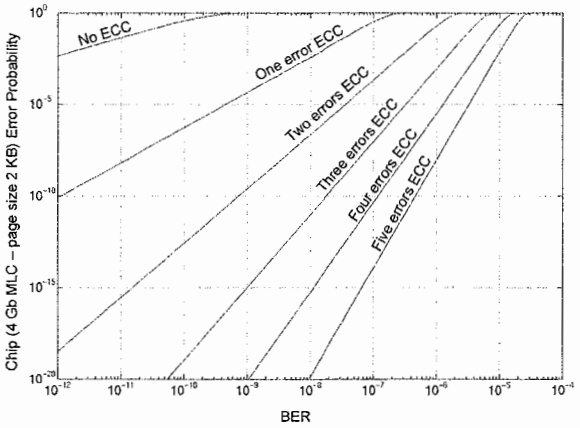


Fig. 14.1. Graph of the Chip Error Probability for a system with 4 Gbit memory. The *CEP* parameters for correction codes of one, two, three, four and five errors are given

14.2 Mathematical background

Error correction codes add redundant terms to the message, such that, on reception, it is possible to detect the errors and to recover the message that has most probably been transmitted.

Correction codes are divided into two well separated groups: block codes and convolutional codes. The first ones are so defined because they treat the information as separated blocks of fixed length, in contrast with convolutional codes which work on a continuous flow of information.

Petitioner Micron Ex-1021, 0397

Another important distinction is between hard decision codes and soft decision codes. This distinction is not based on the structure of the code itself but on the way the information is treated by the code. A binary hard decision code treats all the data in a digital way, i.e. "0" or "1"; in other words, the analog information is converted into digital format using one fixed reference level. On the other hand, a soft decision code uses also reliability information about the decision, e. g. a "0" is read with a 90% reliability or a "1" is read with a 10% reliability.

14.2.1 Block codes

Schematically, a block m of k symbols is encoded in a block c of n symbols ($n > k$) and written in a memory. Inside the memory, different sources may generate errors e (see Chap. 4), so that the block message r is read. The block r is then decoded in d by using the maximum likelihood decoding strategy, so that d is the message that has most probably been written [2–4].

A *Code C* (Fig. 14.2) is the set of codewords obtained by associating the q^k messages of length k of the space A to q^k words of length n of the space B in an univocal way. A code is defined as *linear* if, given two codewords, also their sum is a codeword. When a code is linear, encoding and decoding can be described with matrix operations.

Definition 14.1 G is called *generator matrix of a code C* when all the codewords are obtainable as a combination of the rows of G.

Each code has more than one generator matrix, that is all its linear combinations. Each code has infinite equivalent codes, i.e. all those obtained by permutation or linear combination of the matrix G.

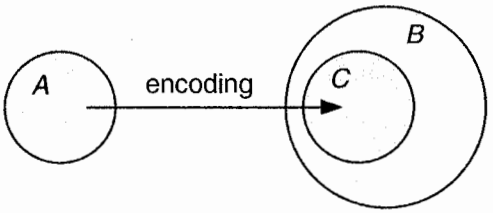


Fig. 14.2. Representation of the space generated by a code

Therefore, encoding a data message m consists in multiplying the message m by the code generator matrix G, according to Eq. (14.8).

$$c = m \cdot G \tag{14.8}$$

Definition 14.2 G is called in *standard form* or in *systematic form* if $G = (I_k, P)$, where I_k is the identity matrix $k \times k$ and P is a matrix $k \times (n - k)$. If G is in standard form then the first k symbols of a word are called information symbols.

Petitioner Micron Ex-1021, 0398

Theorem 14.1 If a code $C[n,k]$ has a matrix $G = (I_k, P)$ in standard form, then there is a *parity matrix* $H = (-P^T, I_{n-k})$ where P^T is the transpose of P and it is a matrix $(n - k)\ x\ k$ and I_{n-k} is the identity matrix $(n - k)\ x\ (n - k)$.

Systematic codes have the advantage that the data message is visible in the codeword and can be read before decoding. For codes in non-systematic form the message is no more recognizable in the encoded sequence and it is necessary to have the inverse encoding function to recognize the data sequence.

Definition 14.3 The *code rate* is defined as the ratio between the number of information bits and the codeword length.

Definition 14.4 If C is a linear code with parity matrix H, then $x \cdot H^T$ is called *syndrome* of x.

All the codewords are characterized by a syndrome equal to 0.

The syndrome is the main actor of the decoding. Having received or read the message r, first of all it is necessary to understand if it is corrupted by calculating:

$$s = x \cdot H^T \tag{14.9}$$

There are two possibilities:

- $s = 0$ => the message r is recognized as correct;
- $s \neq 0$ => the received message contains some errors.

In this second case we use the maximum likelihood decoding procedure, that is we list all the codewords and we calculate the distance between r and the codewords. Therefore the vector c that has most likely been sent is the one which has the smallest distance from r.

Definition 14.5 It is called *minimum distance* or *Hamming distance d* of a code, the minimum number of different symbols between any two codewords.

We can see that for a linear code the minimum distance is equivalent to the minimum distance between all the codewords and the codeword 0.

Definition 14.6 A code has *detection capability v* if it is able to recognize all the messages, containing v errors at the most, as corrupted.

The detection capability is related to the minimum distance as described in Eq. (14.10).

$$v = d - 1 \tag{14.10}$$

Definition 14.7 A code has *correction capability t* if it is able to correct each combination of a number of errors equal to t at the most. The correction capability is calculated from the minimum distance d by the relation:

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor \tag{14.11}$$

where the square brackets mean the floor function.

Petitioner Micron Ex-1021, 0399

Between the length n of the codewords, the number k of information symbols and the number t of correctable errors, there is a relationship known as Hamming inequality:

$$\sum_{i=0}^{t} \binom{n}{i} \cdot (q-1)^i \leq q^{n-k} \tag{14.12}$$

Substantially, the number of parity symbols $(n - k)$ has to be sufficient to represent all the possible errors. A code is defined as *perfect* if the Eq. (14.12) is an equality. In this case all the vectors of the space are contained in circles of radius

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor \tag{14.13}$$

around the codewords. Perfect codes have the highest information rate of all the codes with the same length and the same correction capability.

Codes are not fixed and they can be manipulated or combined depending on applications. The possible operation to increase the minimum distance of a code is the extension.

Definition 14.8 A code $C[n,k]$ is *extended* to a code $C'[n + 1,k]$ by adding one more parity symbol.

Generally, for binary codes, the additional parity bit is the total parity of the message. This is calculated as *sum modulo* 2 (XOR) of all the bits of the message. We can observe that, by itself, this type of code, called parity code, is enough to detect, but not to correct, an odd number of errors.

In a lot of applications there are external factors not subject to error check which determine the length permitted to an error correction code. Non volatile memories, for example, operate on codewords that have a length power of 2.

When the "natural" length of the code is not suitable it is possible to change it with the shortening operation.

Definition 14.9 A $C[n,k]$ is *shortened* into a code $C'[n - j,k - j]$ by erasing j columns of the parity matrix.

Observe that both the shortening operation and the extension are applicable only to linear codes.

Another operation that can be performed with codes is the concatenation (Fig. 14.3). This operation is based on the combination of two different codes called inner code (C_{IN}) and outer code (C_{OUT}). Concatenated codes were first developed in 1966 by Dave Forney.

Here is the concept. C_{IN} is decoded using a maximum-likelihood approach; in this way, the error probability decreases with increasing length. C_{OUT} is a block code of length N and, therefore, it can be decoded in polynomial time of N. The resulting concatenated code $C_{IN} \circ C_{OUT}$ combines together the error probability property of the inner code and the decoding time property of the outer code.

Petitioner Micron Ex-1021, 0400

Typically, the inner code is a soft-decision convolutional code with a short length and the outer code is a longer hard-decision block code (e.g. BCH or Reed Solomon). As the erroneous output of the convolutional code is bursty, the outer code must be more robust to burst errors. Additionally, an interleaving layer may be used that spreads burst errors across a wider range.

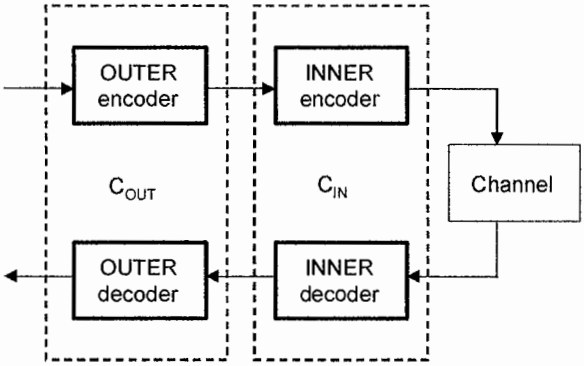


Fig. 14.3. Schematic diagram of the codes concatenation

14.2.2 Convolutional codes

Convolutional codes have been widely used in applications like satellite communications. The greatest difference in comparison with block codes is that the encoder has a memory and this means that, at every clock pulse, the bits encoded depend not only on the kth information bit but also on the preceding m ones. Besides, a convolutional encoder converts the whole information string, without taking into account the length, into a codeword. In this way not all the codewords have the same length.

The construction of the convolutional codes is not based on the distance because it is not possible to define a distance for them. For this reason, there aren't closed formulas describing the error correcting capability of the code and it must be simulated. In Fig. 14.4 there is a comparison between the correction capabilities of convolutional and block codes.

In Fig. 14.5 an example of a convolutional code is represented. The squares are memory elements. The system is ruled by an external clock that produces a signal every t_0 seconds, the effect is that the signals move in the direction of the arrows toward the following element. The output therefore depends on the current input and on the two preceding ones.

Given an input $c = (..., c_{-1}, c_0, c_1, ..., c_l, ...)$ where l is a second of time, the outputs are two sequences v^1 and v^2: $v^1 = (..., v^1_{-1}, v^1_0, v^1_1, ..., v^1_l, ...)$, $v^2 = (..., v^2_{-1}, v^2_0, v^2_1, ..., v^2_l, ...)$. At time l the input is c_l and the output is $v^l = (v^1_l, v^2_l)$ with:

$$v^1_l = c_l + c_{l-2} \tag{14.14}$$

Petitioner Micron Ex-1021, 0401

$$v_l^2 = c_l + c_{l-1} + c_{l-2} \tag{14.15}$$

where the sum is meant as binary addition.

Consequently the code is linear and with memory 2. The codeword is formed as:

$$v = \left(\cdots, v_{-1}^1, v_{-1}^2, v_0^1, v_0^2, v_1^1, v_1^2, \ldots, v_l^1, v_l^2, \cdots\right) \tag{14.16}$$

As for the block codes, the encoding is calculated as $v=c\cdot G$. Called D the delay operator, v and c can be written as:

$$c(D) = \cdots + c_{-1}D^{-1} + c_0 + c_1 D + c_2 D^2 + \cdots \tag{14.17}$$

$$v^i(D) = \cdots + v_{-1}^i D^{-1} + v_0^i + v_1^i D + v_2^i D^2 + \cdots \tag{14.18}$$

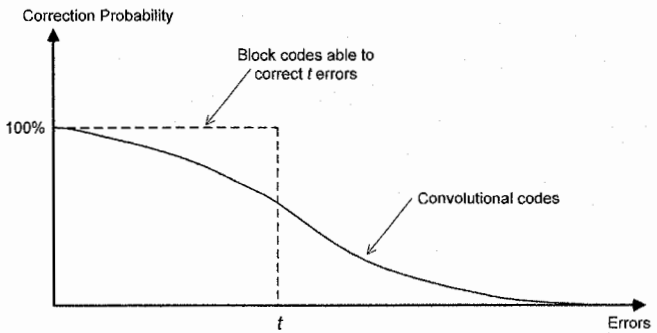


Fig. 14.4. Probability of error occurrence and error correction

Sequence generators, also known as *generator polynomials*, can be written as polynomials using the same approach.

Now consider the code of Fig. 14.5. The generator polynomials are:

$$g_1 = (101) = 1 + D^2 \qquad g_2 = (111) = 1 + D + D^2 \tag{14.19}$$

The decoding algorithm used in this kind of codes is the algorithm of maximum likelihood. The discoverer of this algorithm is Andrea Viterbi. The state of the encoder is defined as the content of memory cells. If K is the number of memory elements, the number of states is 2^K. The state diagram consists of knots, representing the states, and lines, representing the state transitions. Figure 14.6a shows the state diagram of the code introduced in Fig. 14.5.

Petitioner Micron Ex-1021, 0402

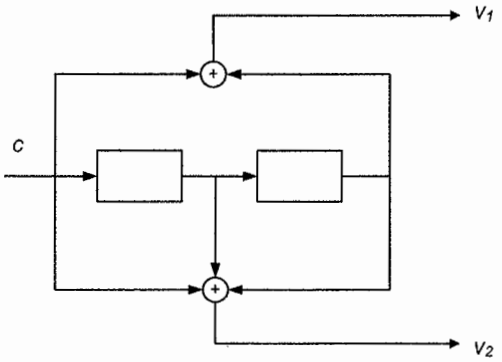


Fig. 14.5. Example of convolutional code (2,1,2)

For example, if we are in the state S_2 and the input is a 0, we reach the state S_1, meanwhile the output, following Eqs. (14.14) and (14.15), will be 01. On the other hand, if the input is a 1, we can reach the state S_3 and the output will be 10.

A Trellis diagram [2, 3] can be deduced from the state diagram by tracing all the possible input/output sequences and the state transitions (Fig. 14.6b).

The Viterbi decoding algorithm makes use of the Trellis diagram and describes a procedure of maximum likelihood decoding. The likelihood is described through a *metric* like the Hamming distance (Definition 14.5). Of course, different metrics are available and the designer has to choose the most suitable depending on the transmission channel.

Viterbi Algorithm is made up by five steps as describe below.

- Step 1: since it is not possible to wait for the whole word which could be endless, truncate the word at time b.
- Step 2: beginning at time $t = 0$, compute the partial metric for the single path entering each state. Store the survivor path and its metric for each state.
- Step 3: increase t. Compute the partial metric for all the 2^K paths entering a state by adding the branch metric entering that state to the metric of the connecting survivor at the previous unit.
- Step 4: for each state, compare the metrics of all the 2^K paths entering that state, select the path with the smallest metric (the survivor) and delete all other paths.
- Step 5: if $t < b$ repeat steps 3 and 4, otherwise stop.

The decoding complexity of this algorithm grows exponentially with the number of memory elements used in the convolutional encoder. The disadvantage in the use of convolutional codes is that a constructive method to generate them does not exist; for values K of practical interest the search for good codes must be carried out in an exhaustive way by trying out all the possible connections, with some rules to reject the bad solutions as fast as possible.

Petitioner Micron Ex-1021, 0403

Besides, despite the sure advantage of a high hardware parallelism, the decoding requires a lot of memory to store the surviving paths and it is necessary to wait for the end of the transmission to decide on all the information bits. Their greatest use is in concatenated codes and in turbo codes, generally combined with a cyclic code or with a non-symmetric error channel.

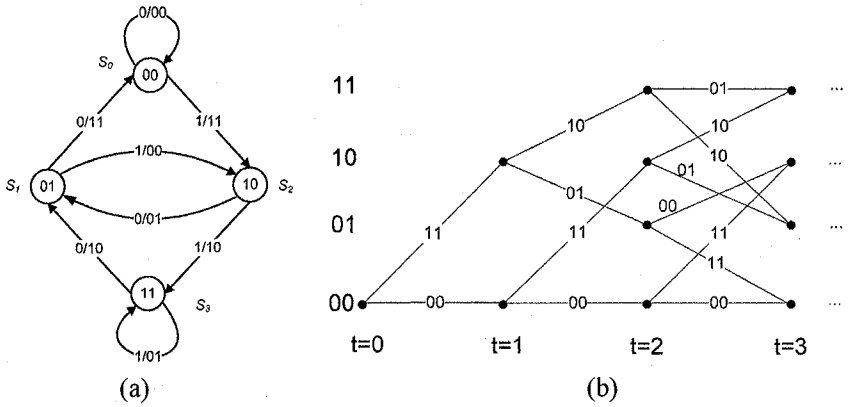


Fig. 14.6. (a) State Diagram and (b) Trellis diagram of circuit in Fig. 14.5

14.3 BCH codes

BCH codes belong to the most important class of cyclic algebraic codes. They were found through independent researches by Hocquenghem in 1959 and by Bose and Ray-Chauduri in 1960 [5, 6].

For BCH codes the minimum distance can be ensured during construction. Generally speaking, in order to know the minimum distance for a linear code with generator polynomial $g(x)$, it is necessary to compute the distance between all the possible codewords. BCH codes, by imposing some constraints on the generator polynomial, are able to ensure a "designed distance".

Definition 14.10 Let β be an element of the Galois Field $GF(q^m)$. Let b be a non-negative integer. A BCH code with "designed" distance d is generated by the polynomial $g(x)$ of minimal degree that has $d - 1$ consecutive powers of β: β^b, β^{b+1},..., β^{b+d-2} as roots. Given Ψ_i the minimal polynomial of β^{b+i} for $0 \le i < d - 1$, $g(x)$ is computed as:

$$g(x) = LCM\{\psi_0(x), \psi_1(x), \ldots, \psi_{d-2}(x)\} \tag{14.20}$$

and the data carried by the code is $k = n\text{-}deg(g(x))$.

It is possible to show that the designed d is at least $2t + 1$, hence the code is able to correct t errors.

Petitioner Micron Ex-1021, 0404

If we assume b=1, and β a primitive element of $GF(q^m)$ the code becomes a *narrow-sense* and *primitive* BCH code of length $q^m - 1$ able to correct t errors. We shall now consider narrow-sense primitive BCH codes.

In general, the decoding of a BCH code is at least 10 times more complicated than the encoding. In the following we will deal only with binary BCH codes, whose structure is presented in Fig. 14.7.

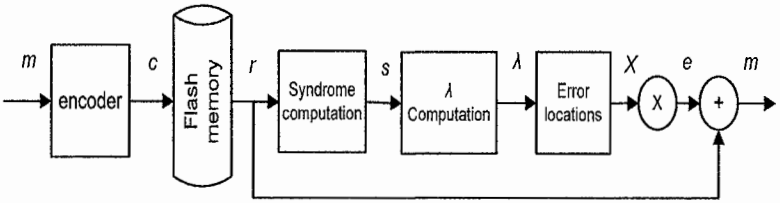


Fig. 14.7. General structure of a binary BCH code

14.3.1 Encoding

Suppose we have a BCH code *[n,k]* with generator polynomial $g(x)$ and a message $m(x)$ to encode, written as a polynomial of degree $k - 1$.

First of all, the message $m(x)$ is multiplied by x^{n-k} and subsequently divided by $g(x)$, obtaining a quotient $q(x)$ and a remainder $r(x)$ in accordance with Eqs. (14.21) and (14.22).

$$\frac{m(x)\cdot x^{n-k}}{g(x)} = q(x) + \frac{r(x)}{g(x)} \tag{14.21}$$

$$m(x)\cdot x^{n-k} + r(x) = q(x)\cdot g(x) \tag{14.22}$$

The multiplication of the message $m(x)$ by x^{n-k} produces, as a result, a polynomial of degree $n - 1$ where the first $n - k$ coefficients, now null, will then be occupied by parity bits.

Therefore the encoded word $c(x)$ is calculated as:

$$c(x) = m(x)\cdot x^{n-k} + r(x) \tag{14.23}$$

Practical implementation of Eq. (14.23) is depicted in Fig. 14.8. Note that, since we are considering binary BCH codes, the sum is actually a XOR, while the product is an AND.

The structure is formed by $n - k$ registers and there can be a maximum number of n-k multiplications by the coefficients of the generator polynomial. The content of the registers has to be summed to the incoming bit if the corresponding

Petitioner Micron Ex-1021, 0405

coefficient of the generator polynomial is 1, otherwise there will not be any wire indicating the sum.

For the first k steps, the message inputs the structure and contemporarily goes out to form the codeword. At the end, the $n - k$ registers contain the parity bits. The parity bits output in $n - k$ clock pulses. Note that the latency of this structure is n.

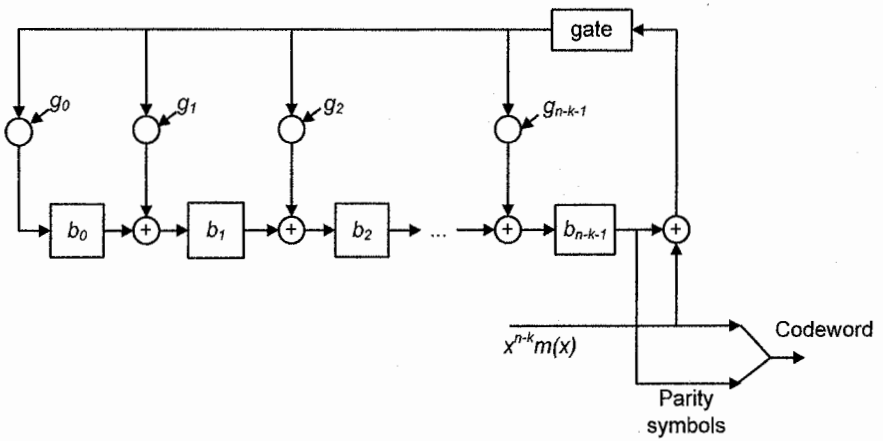


Fig. 14.8. Binary BCH encoder

14.3.2 Decoding

The decoding operation follows three fundamental steps, as shown in Fig. 14.7.

- Calculation of the syndromes
- Calculation of the coefficients of the error locator polynomial (usually done with Berlekamp–Massey algorithm [1, 6–8])
- Calculation of the roots of the error locator polynomial (usually done with Chien algorithm [1, 2, 4])

During the transmission or the reading of the encoded message some errors may occur. Also the errors can be represented by a polynomial that has coefficient 1 in correspondence with every error's position:

$$E(x) = E_0 + E_1 x + \ldots + E_{n-1} x^{n-1} \tag{14.24}$$

Observe that, in order for the code to be corrector of t errors, at most t non-null coefficients are allowed in Eq. (14.24).

The read vector $R(x)$ is therefore:

$$R(x) = c(x) + E(x) \tag{14.25}$$

Petitioner Micron Ex-1021, 0406

The first decoding step consists in calculating the $2t$ syndromes for the read message:

$$\frac{R(x)}{\psi_i(x)} = Q_i(x) + \frac{S_i(x)}{\psi_i(x)} \quad \text{with} \quad 1 \le i \le 2t \tag{14.26}$$

$$S_i(x) = Q_i(x) \cdot \psi_i(x) + R(x) \quad \text{with} \quad 1 \le i \le 2t \tag{14.27}$$

In accordance with Eqs. (14.26) and (14.27), the received vector is divided by each minimal polynomial Ψ_i forming the generator polynomial, thus getting a quotient $Q_i(x)$ and a remainder $S_i(x)$ called *syndrome*.

At this point the $2t$ syndromes must be evaluated into the elements β, β^2, β^3,..., β^{2t} whose Ψ_i are the minimal polynomials. With reference to Eq. (14.28) this evaluation is reduced to an evaluation of the message received in β, β^2, β^3,..., β^{2t}, since $\Psi_i(\beta^i) = 0$ (for $1 \le i \le 2t$) by definition of minimal polynomial.

$$S_i(\beta^i) = S_i = Q_i(\beta^i) \cdot \psi_i(\beta^i) + R(\beta^i) = R(\beta^i) \tag{14.28}$$

Consequently, the ith syndrome can be calculated either as a remainder of the division between the received message and the minimal polynomial Ψ_i, then evaluated in β^i, or as the evaluation in β^i of the received message.

Observe that, in case no errors occur, the polynomial received is a codeword: therefore the remainder of the division of Eq. (14.26) is null and all the syndromes are identically null. On the other hand, verifying if the syndromes are identically null is a necessary and sufficient condition to understand if the read message is a codeword or if some errors occurred.

For binary codes we use the property:

$$S_{2i} = S_i^2 \tag{14.29}$$

It is therefore necessary to calculate only t syndromes. The syndromes are linked to the error positions, in fact:

$$S_i = R(\alpha^i) = c(\alpha^i) + E(\alpha^i) = E(\alpha^i) \tag{14.30}$$

By indicating the error positions with X and the number of errors that have really occurred with v (for a code corrector of t errors it must be $v \le t$), we get Eq. (14.31).

$$\begin{aligned} S_i &= E(\alpha^i) = x^{e_1} + x^{e_2} + \ldots + x^{e_v} \\ &= X_1^i + X_2^i + \ldots + X_v^i = \sum_{l=1}^{v} X_l^i \end{aligned} \tag{14.31}$$

Petitioner Micron Ex-1021, 0407

From Eq. (14.31) we get Eqs. (14.32) called *power-sum symmetric functions*:

$$\begin{aligned} S_1 &= X_1 + X_2 + \ldots + X_\nu \\ S_3 &= X_1^3 + X_2^3 + \ldots + X_\nu^3 \\ &\vdots \\ S_{2t-1} &= X_1^{2t-1} + X_2^{2t-1} + \ldots + X_\nu^{2t-1} \end{aligned} \tag{14.32}$$

Each method for solving these equations is a BCH decoding procedure.

Definition 14.11 It is defined as *error locator polynomial* $\Lambda(x)$ the polynomial whose roots are the inverse of the error positions.

From the definition we have:

$$\Lambda(x) = \prod_{i=1}^{\nu} (1 - xX_i) \tag{14.33}$$

Observe that the degree of the error locator polynomial gives the number of errors that occurred. The degree of $\Lambda(x)$ is t at most, so, in the case more than t errors occur, the polynomial $\Lambda(x)$ could erroneously indicate t or less errors.

For each error X_j it must be:

$$1 + \Lambda_1 X_j^{-1} + \ldots + \Lambda_{\nu-1} X_j^{-\nu+1} + \Lambda_\nu X_j^{-\nu} = 0 \tag{14.34}$$

Multiplying by $X_j^{i+\nu}$ we get

$$X_j^{i+\nu} + \Lambda_1 X_j^{i+\nu-1} + \ldots + \Lambda_{\nu-1} X_j^{i+1} + \Lambda_\nu X_j^i = 0 \tag{14.35}$$

then, by summing over j,

$$\sum_{j=1}^{\nu} X_j^{i+\nu} + \Lambda_1 \sum_{j=1}^{\nu} X_j^{i+\nu-1} + \ldots + \Lambda_\nu \sum_{j=1}^{\nu} X_j^i = 0 \tag{14.36}$$

$$S_{i+\nu} + \Lambda_1 S_{i+\nu-1} + \ldots + \Lambda_\nu S_i = 0 \tag{14.37}$$

which rewritten in matricial form becomes:

$$\begin{pmatrix} S_{\nu+1} \\ S_{\nu+2} \\ S_{\nu+3} \\ \vdots \\ S_{2\nu-1} \end{pmatrix} = \begin{pmatrix} S_1 & S_2 & S_3 & \cdots & S_\nu \\ S_2 & S_3 & S_4 & \cdots & S_{\nu+1} \\ S_3 & S_4 & \cdots & \cdots & S_{\nu+2} \\ \vdots & \vdots & \vdots & & \vdots \\ S_\nu & S_{\nu+1} & S_{\nu+2} & \cdots & S_{2\nu-1} \end{pmatrix} \cdot \begin{pmatrix} \Lambda_\nu \\ \Lambda_{\nu-1} \\ \Lambda_{\nu-2} \\ \vdots \\ \Lambda_1 \end{pmatrix} \tag{14.38}$$

Consequently, once the syndromes are known, it is possible to find the coefficients of the error locator polynomial.

Petitioner Micron Ex-1021, 0408

The last step of the decoding process consists in searching for the roots of the error locator polynomial. If the roots are separate and they are in the field, then it is enough to calculate their inverse to have the error positions. If they are not separate or they are not in the correct field, it means that the word received has a distance from a codeword greater than t. In this case an incorrigible error pattern occurred and the decoding process fails.

14.4 Reed–Solomon codes

Reed–Solomon codes were described for the first time in 1960, in the article "Polynomial codes over certain finite fields" by Reed and Solomon [9]. It is only later that the close link with BCH codes was understood.

There are a lot of ways to present Reed–Solomon codes; in the following we will prefer the one that more closely links them to BCH codes. In this way, we will be able to underline that Reed–Solomon codes are a subset of BCH codes q-ari.

Looking back to the Definition 14.10 of BCH codes we can affirm:

Definition 14.12 Reed–Solomon codes are BCH codes q^m-ari with $n=q^m-1$.

The construction of the generator polynomial $g(x)$ results to be very simple, as the minimal polynomial in $GF(q^m)$ of an element β in $GF(q^m)$ is simply $x-\beta$. Generally Reed–Solomon codes correct symbols instead of bits.

Therefore, a generator polynomial of a Reed–Solomon code has the form

$$g(x)=\left(x-\beta^{b}\right)\cdot\left(x-\beta^{b+1}\right)\cdots\left(x-\beta^{b+2t-1}\right) \tag{14.39}$$

and the code has $k=q^m-1-2t$ information symbols.

Similarly to BCH codes, assuming $b=1$ we get narrow-sense codes, even though, in applications, the choice $b=0$ often simplifies the computational complexity.

A particular case, and surely the most used in applications, concerns Reed–Solomon codes in $GF(2^m)$, where each symbol is composed by m bits.

Suppose, for example, we search for a code whose symbol is a byte, namely 8 bits. By definition of Reed–Solomon codes, the codeword length is fixed at $2^8-1=255$ bytes. In the particular case of a code able to correct two symbols, we need four parity bytes.

Now, suppose we wish to construct a Reed–Solomon code in $GF(8)$ corrector of two error symbols. The primitive polynomial of the field is x^3+x+1. The code is seven symbols long, three of them are information symbols. The generator polynomial must have four roots in $GF(8)$:

$$\begin{aligned} g(x) &= (x-\alpha)\cdot\left(x-\alpha^2\right)\cdot\left(x-\alpha^3\right)\cdot\left(x-\alpha^4\right) \\ &= x^4+\alpha^3x^3+x^2+\alpha x+\alpha^3 \end{aligned} \tag{14.40}$$

The code just described corrects two symbols, each composed by 3 bits. It is however wrong to deduce that such code corrects 6 bits, since, if errors occur in

Petitioner Micron Ex-1021, 0409

different symbols, the code is unable to correct them. Thus, the code is able to correct a number of bits varying from 2 to 6 depending on the position of the errors inside the symbols. Due to the fact that they were born to correct symbols, Reed–Solomon codes are particularly suitable when burst type errors occur.

The general structure of a Reed–Solomon code is that of Fig. 14.9. Reed–Solomon codes require a further step (Forney algorithm [1–4]) during the decoding process, compared to binary BCH codes, since it is not enough to know the error positions but it is also necessary to know their magnitude.

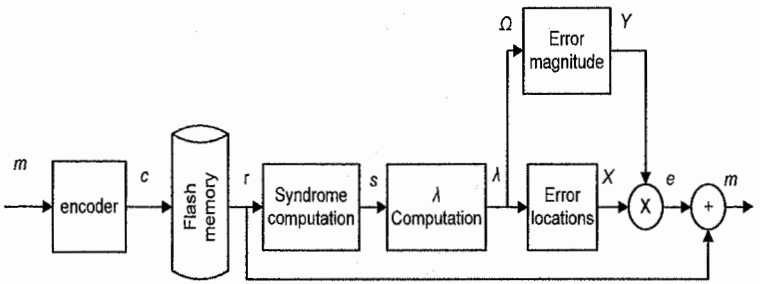


Fig. 14.9. Structure of a Reed–Solomon code

14.5 BCH versus Reed–Solomon

The choice of the more suitable cyclic code for a particular application is not simple and it would be useful to know the distribution and the generation of the errors in details.

For this reason, it is not possible to say whether a Reed–Solomon code is superior or inferior to a binary BCH code. The greatest difference between the two codes is the correction basic unit, as BCH code corrects bits, while the Reed–Solomon code corrects symbols. Therefore, a BCH code corrects t bit errors, while the Reed–Solomon code corrects t symbols of s bit errors, which can be a variable number of bits: from t bits if errors occur in different symbols up to st bits in case errors occur consecutively. At a first glance this may seem an advantage, which however is fully exploited only if the error source is of burst type which is not the case for NAND memories (Chaps. 13 and 4).

The choice of a code can also be based on the occupied area. In this case we have to note that both hardware implementation and the number of parity bits contribute to increase area. Regarding this last aspect, fixing the same correction capability, Reed–Solomon codes require a greater number of parity bits, as shown in Table 14.1.

For what concerns the encoding implementation, the difference is surely that binary codes require binary operators (AND and XOR), while Reed–Solomon codes require operators in $GF(2^m)$ which have a greater computational complexity.

Petitioner Micron Ex-1021, 0410

Besides, since Reed–Solomon codes also require a greater number of parity bits, also the generator polynomial has a higher degree, and consequently the number of operators results to be higher. The remainders of the divisions must be stored into registers. Given the structure of the two codes, while BCH codes require the storage of bits, for Reed–Solomon codes it is necessary to store symbols.

Table 14.1. Number of parity bits required by Hamming inequality (minimum), BCH and RS codes

Data bits	*Min* $T=1$	*BCH* $t=1$	*RS* $t=1$	*Min* $t=2$	*BCH* $t=2$	*RS* $t=2$	*Min* $t=3$	*BCH* $t=3$	*RS* $t=3$	*Min* $t=4$	*BCH* $t=4$	*RS* $t=4$
2,048	12	12	18	22	24	36	31	36	54	40	48	72
4,096	13	13	18	24	26	36	34	39	54	44	52	72
8,192	14	14	20	26	28	40	37	42	60	48	56	80
16,384	15	15	22	28	30	44	40	45	66	52	60	88
32,768	16	16	24	30	32	48	43	48	72	56	64	96

The hardware structure of the syndromes calculation involves the same considerations made for the encoding: the Reed–Solomon codes require a greater number of operators with a greater complexity. Besides, Reed–Solomon codes require the calculation of $2t$ syndromes, a number doubled if compared to binary BCH codes which exploit the binary property:

$$S_{2i} = S_i^2 \tag{14.41}$$

Consequently also the inputs to the Berlekamp algorithm are doubled in comparison to a BCH code. If the iterative Berlekamp algorithm [1–4] is chosen, BCH codes require t iterations instead of the $2t$ required by Reed–Solomon codes.

The same considerations made for the encoding implementation apply to the Chien algorithm. Despite the fact that the number of operators is not actually greater, their complexity, as well as the registers for the storage, are more onerous in the case of Reed–Solomon codes compared to BCH ones.

Finally, note that Reed–Solomon codes require a further step in the decoding, represented by the Forney algorithm. The most onerous operation in this step is the inversion, which is not simple and it requires an additional ROM in order to be efficient.

For what said so far, it is evident that if the choice criterion is the occupied area, BCH codes are by far superior to Reed–Solomon codes.

When the choice criterion becomes the latency then the situation changes and the BCH codes turn out to have inferior performances compared to Reed–Solomon codes. The difference lies in the fact that BCH codes are serial and process 1 bit at a time, while Reed–Solomon codes, operating on symbols, are defined with a parallel structure.

In conclusion, Reed–Solomon codes generally have a greater correction capability for burst type errors, but they require a bigger area. On the contrary, BCH codes

Petitioner Micron Ex-1021, 0411

have a smaller area but a higher latency time. It is also possible to upgrade the BCH performances using a parallel architecture as shown in the next section.

Usually, NAND memories don't show a burst behavior of the failed cells; therefore, BCH is the preferred choice by the NAND users.

14.6 Parallel BCH

In this section we assume to have a system formed by a control unit which provides and receives data from a NAND memory on an 8-bit bus.

The BCH coding and decoding operations are described in the vast majority of the cases as a process which operates in a serial way on a sequence of bits. Hence the difficulty to adopt it in a system that operates on words. The solution consists in defining machines able to elaborate the received data in a parallel way [10–13].

The serial structure of the parity computation machine can be turned into a parallel machine following some simple steps. The iterative operation carried out by the serial divisor to elaborate a word, can be unrolled as shown in Fig. 14.10. In this way, the following state of the registers is calculated on the basis of all the operations needed to elaborate a whole message word.

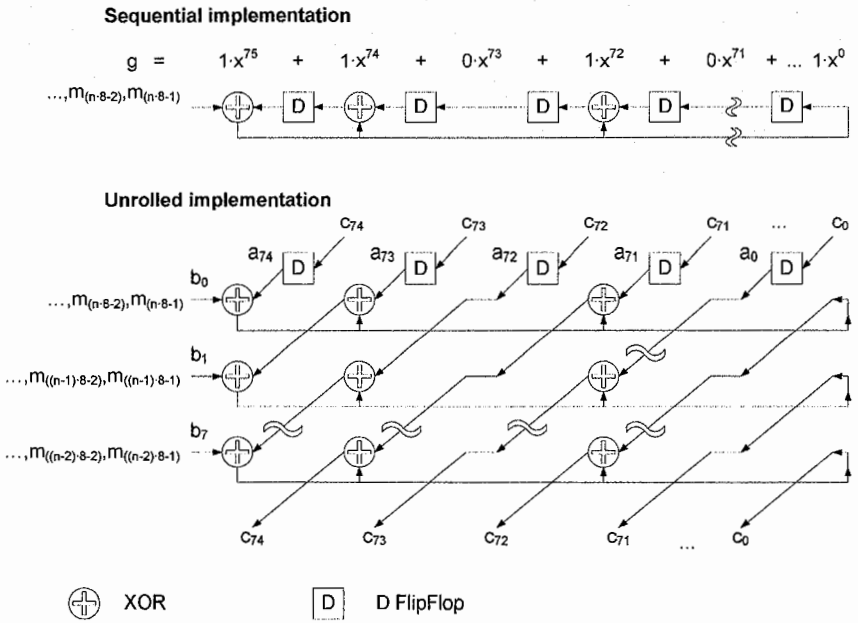


Fig. 14.10. Description of the sequential implementation of the divider and of how it could be unrolled to get a parallel implementation

Petitioner Micron Ex-1021, 0412

The direct synthesis of the unrolled implementation of the divisor risks causing some complications to the synthesis tools due to the elevated number of inputs of the combinatorial logic and the depth of the paths.

The whole combinatorial part of the unrolled implementation can be expressed by describing each of its outputs as a function of the previous state a_w and of its inputs b_k. Given *mt* the number of parity bits and *z* the number of bits processed in parallel, we can write:

$$c_j = \sum_{k=0}^{z-1} d_{kj} \cdot b_k + \sum_{w=0}^{mt-1} l_{wj} \cdot a_w \; ; \qquad d_{kj}, l_{wj} \in \{0,1\} \quad \forall \; j \in \{0,\ldots, mt-1\} \tag{14.42}$$

The same approach can be used for syndrome computation.

As Berlekamp takes only *t* iterations, there is no need to extra design effort.

Once the coefficients of the error locator polynomial have been calculated through the Berlekamp machine, the Chien machine searches all its roots. The inverses of such roots indicate the error positions.

$$\Lambda(x) = 1 + \Lambda_1 x + \ldots + \Lambda_t x^t \tag{14.43}$$

To determine the roots of the polynomial, the Chien machine orderly evaluates $\Lambda(x)$ in all the elements of the field α^0, α^1, α^2, α^3,... α^N. For each element *i* of the field for which the polynomial is null, the corresponding position $(2^m - 1 - i)$ is an error position. A possible implementation of the Chien machine is represented in Fig. 14.11.

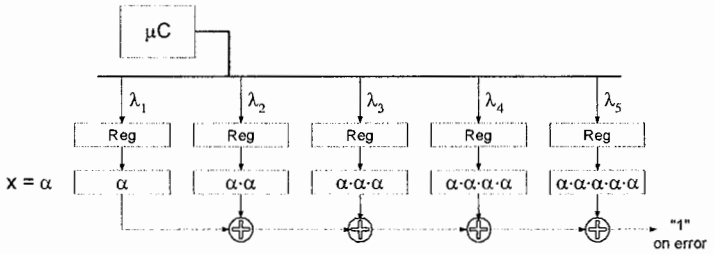


Fig. 14.11. Chien machine for a 5-error BCH: sequential implementation

The Chien machine, thus realized, searches the roots one at a time. In this phase the system is not executing any operation, neither read nor write, towards the memory but it is waiting for the signal of the error positions. Such operation, carried out for all the bits of the message, results to be a very onerous operation in

Petitioner Micron Ex-1021, 0413

terms of time. Also in this case it is possible to use a parallel architecture. Unlike the parity and the syndromes computation machines, which have to operate with a parallelism equal to the input data parallelism, the Chien machine does not have particular limits other than the complexity, the area occupied and the consumption. In the parallel implementation, at each cycle more error positions are contemporarily evaluated.

The execution time of the Chien algorithm is usually seen by the system as an additional time to the information transfer time. If the probability to have one or more errors becomes considerable, the time of search of error positions significantly weights on the system performance. We have seen how a parallel structure of the Chien machine can increase its parallelism and therefore its speed. The increase of the Chien machine parallelism inevitably affects the area used for the BCH machine (Fig. 14.12).

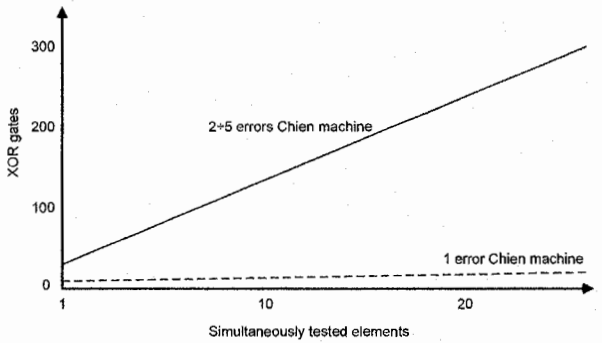


Fig. 14.12. Area impact of the Chien parallelism

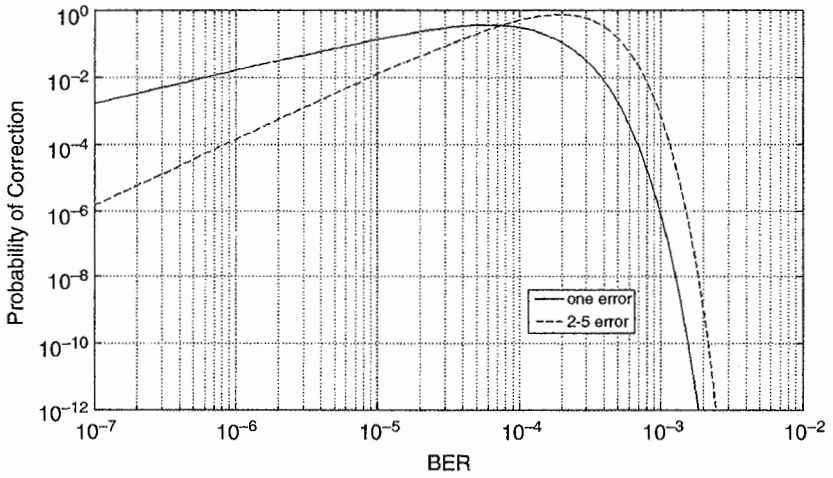


Fig. 14.13. Probability of correction for a 2,112-Byte page: single error versus 2–5 errors

Petitioner Micron Ex-1021, 0414

Figure 14.13 shows, for a 2,112-Byte page, the probability of correcting only one error and the probability of correcting 2–5 errors. Assuming a BER of 10^{-6}, we have that the probability of a single error is equal to $1.7 \cdot 10^{-2}$ and the probability of 2–5 errors is equal to $1.5 \cdot 10^{-4}$ respectively. The probability of a single error is definitely more significant and since the Berlekamp algorithm exactly indicates the number of errors to correct, it may be useful to exploit this information.

The system derived is therefore composed of a couple of Chien machines with different parallelisms, one for the correction of the single error and the second for the correction of 2–5 errors (Fig. 14.14) [11].

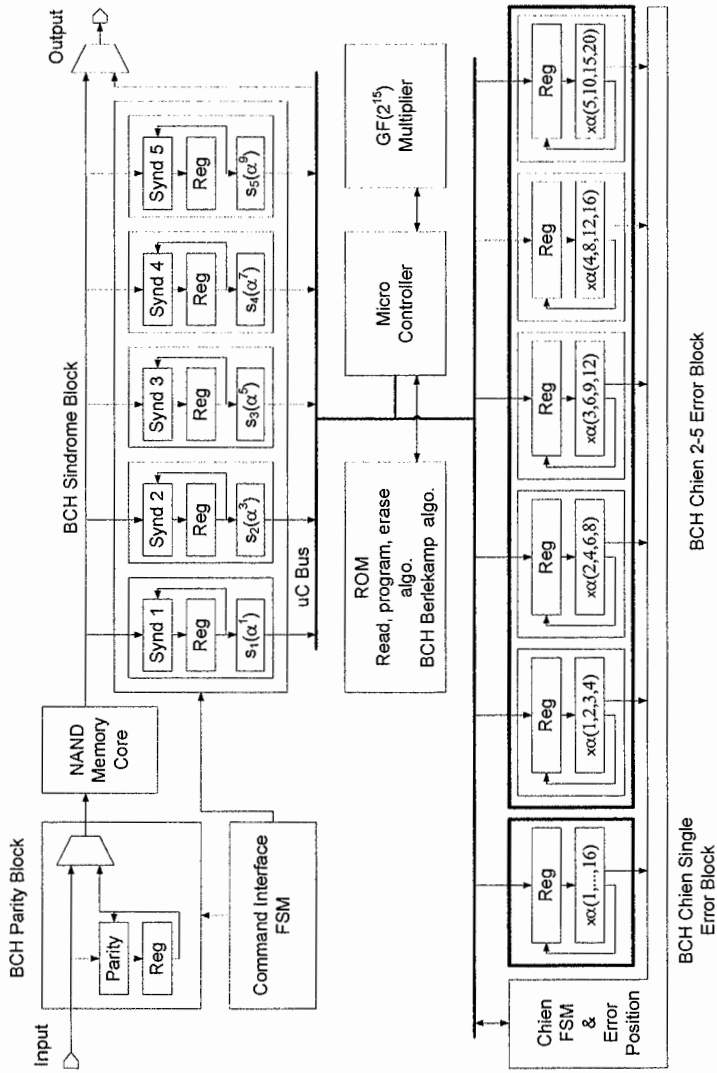


Fig. 14.14. Example of BCH engine including two parallel Chien machines

Petitioner Micron Ex-1021, 0415

14.7 Low-Density Parity-Check (LDPC) code

Since its re-discovery in late 1990s, LDPC code has received a tremendous amount of attentions because of the excellent error-correction capability and experienced a widespread use in many real-life data communication and storage applications.

In 1960s Dr. Gallager invented LDPC codes [14], in which two innovative ideas were exploited: iterative decoding and constrained random code construction. However, except for the papers of Zyablov and Pinsker [15], Tanner [16] and Margulis [17], Gallager's work was neglected by the majority of the scientific community for the next 30 years. Few years after the birth of Turbo code [18] in which both the above ideas are explored, LDPC codes were independently rediscovered by Wiberg [19] and MacKay and Neal [20].

An LDPC code is defined as the null space of a very sparse $M \times N$ parity check matrix, and typically is represented by a bipartite graph (notice that bipartite graph is one in which the nodes can be partitioned into two disjoint sets) between N variable (or message) nodes in one set and M check (or constraint) nodes in another set, as illustrated in Fig. 14.15. The LDPC code is called (j, k)-regular if each variable node has a degree of j and each check node has a degree of k. LDPC codes can be effectively decoded by the iterative belief-propagation (BP) (also known as sum-product) algorithm.

The structure of BP decoding algorithm directly matches the underlying code bipartite graph: decoding message is computed on each variable node and check node and iteratively exchanged through the edges between the neighboring nodes. It is well known that BP decoding algorithm works well if the underlying code bipartite graph does not contain too many short cycles. Thus, it is typically required that the graph should be 4-cycle free, which is easy to achieve, but the construction of graph with higher order cycle free is typically nontrivial.

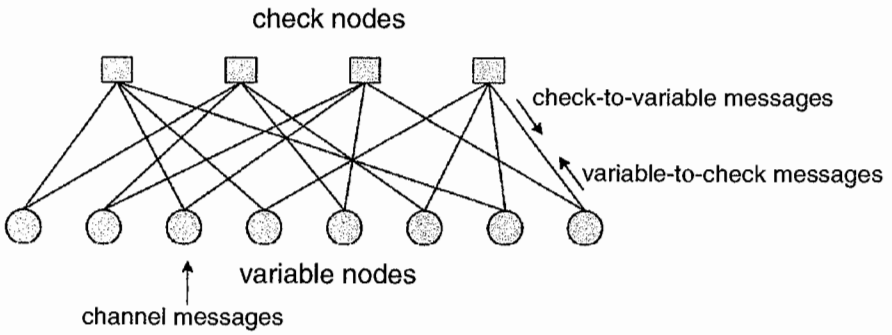


Fig. 14.15. Illustration of LDPC code bipartite graph

Petitioner Micron Ex-1021, 0416

14.7.1 LDPC code decoding algorithm

LDPC codes can be effectively decoded using BP algorithm. Since the direct implementation of BP algorithm results in computational complexity due to a large number of multiplications, it is a convention to introduce certain logarithmic quantities to convert these multiplications into additions, which leads to the Log-BP algorithm. Before presenting the Log-BP decoding algorithm, we need to introduce some definitions as follows: Let H denote the $M \times N$ parity check matrix and $H_{i,j}$ denote the entry of matrix H at the position (i, j). We define the set of bits n that participate in parity check m as $N(m) = \{n : H_{m,n} = 1\}$, and the set of parity checks m in which bit n participates as $M(n) = \{m : H_{m,n} = 1\}$. We denote the set $N(m)$ with bit n excluded by $N(m) \setminus n$, and the set $M(n)$ with parity check m excluded by $M(n) \setminus m$.

Algorithm 14.7.1 *Iterative Log-BP Decoding Algorithm*
Input: the log-likelihood ratio (LLR)

$$\gamma_n = \log \frac{p_n^0}{p_n^1} \text{ for } n = 1,2,..., N$$

Output: hard decision

$$\hat{X} = \{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_N\}$$

Procedure
1. Initialization: for each $(m, n) \in \{(i, j) \mid H_{i,j} = 1\}$, compute

$$\alpha_{m,n} = sign(\gamma_n) \log\left(\frac{1 + e^{-|\gamma_n|}}{1 - e^{-|\gamma_n|}}\right), \; sign(\gamma_n) = \begin{cases} +1 & \gamma_n \geq 0 \\ -1 & \gamma_n < 0 \end{cases}$$

2. Iterative Decoding
- Horizontal (or check node processing) step: for each m, n, compute

$$\beta_{m,n} = \log\left(\frac{1 + e^{-|\alpha|}}{1 - e^{-|\alpha|}}\right) \prod_{n' \in N(m) \setminus n} sign(\alpha_{m,n'}), \quad \alpha = \sum_{n' \in N(m) \setminus n} |\alpha_{m,n'}| \tag{14.44}$$

Vertical (or variable node processing) step: for each m, n, compute

$$\alpha_{m,n} = sign(\gamma_{m,n}) \log\left(\frac{1 + e^{-|\gamma_{m,n}|}}{1 - e^{-|\gamma_{m,n}|}}\right), \quad \gamma_{m,n} = \gamma_n + \sum_{m' \in M(n) \setminus m} |\beta_{m',n}| \tag{14.45}$$

For each n, update the "pseudo-posteriori log-likelihood ratio (LLR)" λ_n as:

$$\lambda_n = \gamma_n + \sum_{m \in M(n)} \beta_{m,n}. \tag{14.46}$$

Petitioner Micron Ex-1021, 0417

- Decision Step:
  (a) Perform hard decision on $\{\lambda_1, \ldots, \lambda_N\}$ to obtain $\hat{X} = \{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_N\}$ such that

$$\hat{x}_n = 0 \text{ if } \lambda_n > 0 \text{ and } \hat{x}_n = 1 \text{ if } \lambda_n \le 0$$

  (b) If the hard decision satisfy the code parity check constraint, then algorithm terminates, else go to Horizontal step. A failure will be declared if pre-set maximum number of iterations occurs without successfully decoding. ■

The variables $\alpha_{m,n}$ and $\beta_{m,n}$ in the above algorithm are called variable-to-check extrinsic information and check-to-variable extrinsic information, respectively. Notice that the function

$$f(x) = \log\left(\frac{1+e^{-|x|}}{1-e^{-|x|}}\right)$$

in the above algorithm is typically implemented as a Look-Up Table (LUT).

In practice, the above BP algorithm may be simplified to a so-called min-sum decoding algorithm in order to largely reduce the computational complexity at small decoding performance degradation. The main difference between BP and min-sum algorithm lies in the check node processing, i.e., the check node processing in BP algorithm is realized as

$$\beta_{m,n} = \Phi\left(\sum_{n' \in N(m)\backslash n} \Phi(|\alpha_{m,n'}|)\right) \prod_{n' \in N(m)\backslash n} sign(\alpha_{m,n'}), \quad \Phi(x) \equiv -\log\left[\tanh\left(\frac{x}{2}\right)\right]$$

The check node processing in min-sum algorithm is approximated as

$$\beta_{m,n} = \min_{n' \in N(m)\backslash n} (\alpha_{m,n'}) \prod_{n' \in N(m)\backslash n} sign(\alpha_{m,n'}).$$

Therefore, the function $\Phi(x)$, which is typically implemented as LUT, is eliminated in the min-sum decoding algorithm.

Regardless to which specific decoding algorithm is being used, it is clear that each LDPC code decoding iteration can be performed with a full parallelism by mapping each check or variable node to one decoding processor as illustrated in Fig. 14.15: each check node is realized by check node processing unit to compute the check-to-variable extrinsic information $\beta_{m,n}$, each variable node is realized by variable node processing unit to compute the variable-to-check extrinsic information $\alpha_{m,n}$, pseudo-posterior LLR λ_n, and generate $\hat{x}_n$ by performing hard decision on λ_n. By also delivering the hard decision from each variable node to its neighboring check nodes, the parity check operation after each decoding iteration can be incorporated into Check Node computation Units (CNU). Clearly, under reasonable codeword lengths, such fully parallel decoders may incur very high and even prohibitive silicon area cost, although extremely high decoding throughput can be

Petitioner Micron Ex-1021, 0418

achieved. Hence, partially parallel decoder architectures are typically preferable for practical LDPC code decoder implementation, seeking appropriate trade-off between silicon cost and decoding throughput.

14.7.2 LDPC code construction and encoder/decoder design

To be a practically promising candidate for Flash memories, LDPC codes must not only achieve very low decoding error rate with a high code rate, but also be suitable for high-speed VLSI implementation to meet the high data rate requirements of Flash memory with minimal silicon and energy cost. It has been well demonstrated that quasi-cyclic (QC) LDPC codes are one family of such implementation-oriented LDPC codes. The parity check matrix of a QC-LDPC code consists of arrays of circulants. A circulant is a square matrix in which each row is the cyclic shift of the row above it, and the first row is the cyclic shift of the last row. The parity check matrix H of a QC-LDPC code can be written as

$$H = \begin{bmatrix} H_{1,1} & H_{1,2} & \cdots & H_{1,n} \\ H_{2,1} & H_{2,2} & \cdots & H_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ H_{m,1} & H_{m,2} & \cdots & H_{m,n} \end{bmatrix},$$

where each sub-matrix $H_{i,j}$ is a binary circulant. Data storage systems such as Flash memory and hard disk drive typically demand very high code rate (e.g., 8/9 and higher). Under such high code rates, regular QC-LDPC codes are typically used, i.e., all the rows have the same number of 1s and all the columns have the same number of 1s, and all the sub-matrices $H_{i,j}$ have the same column weight of 1 or 2. Since LDPC codes are subject to error floor, the code parity check matrix column weight is typically 4 or even higher in order to ensure a sufficiently low error floor (e.g., error floor only occurs below the decoding failure rate of 10^{-12}). The regular and cyclic structure of QC-LDPC code parity check matrix can be leveraged to largely improve its encoder and decoder implementation efficiency as described below.

In the context of LDPC encoder design, the straightforward method is to multiply the information bits with the dense generator matrix derived from the sparse parity check matrix. The denseness of the generator matrix and typically large code length make the parallel implementation of generator matrix-vector multiplication impractical due to very high implementation complexity. Hence, a partially parallel encoder implementation is indispensable. However, for general non-QC LDPC codes constructed randomly, their dense generator matrices may not have any structural regularity that can be used to develop efficient partially parallel encoder architecture. For QC-LDPC codes, partially parallel encoder design becomes much more tractable. Assume the QC-LDPC code parity check matrix is $m \times n$ array of circulants and each circulant is $p \times p$, let us consider the

Petitioner Micron Ex-1021, 0419

simplest scenario, i.e., the matrix has a full rank of $m \cdot p$. We assume that code parity check matrix can be column-wise permuted so that the following sub-array has a full rank of $m \cdot p$:

$$\begin{bmatrix} H_{1,n-m+1} & H_{1,n-m+2} & \cdots & H_{1,n} \\ H_{2,n-m+1} & H_{2,n-m+2} & \cdots & H_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ H_{m,n-m+1} & H_{m,n-m+2} & \cdots & H_{m,n} \end{bmatrix}.$$

Suppose the encoding is carried out in the systematic way, i.e., the first n-m bits in each codeword are the information bits, and the first $(n - m) \cdot p$. columns of the parity check matrix correspond to the $(n - m) \cdot p$ information bits. Hence, the corresponding generator matrix can have the following form:

$$G = \begin{bmatrix} I & O & \cdots & O & G_{1,1} & G_{1,2} & \cdots & G_{1,m} \\ O & I & \cdots & O & G_{2,1} & G_{2,2} & \cdots & G_{2,m} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ O & O & \cdots & I & G_{n-m,1} & G_{n-m,2} & \cdots & G_{n-m,m} \end{bmatrix},$$

where I and O represent identity $p \times p$ matrix and zero $p \times p$ matrix. For G being the generator matrix, it must satisfies $H \cdot G^T = 0$, which clearly suggests that each $G_{i,j}$ should also be a $p \times p$ circulant. It is evident that the generator matrix-vector multiplication for QC-LDPC encoding can be carried out in a partially parallel manner by leveraging the inherent cyclic structure of the generator matrix. For more detailed discussions on QC-LDPC code encoder design, readers are referred to [21].

In the context of QC-LDPC code decoder, Fig. 14.16 shows the most straightforward decoder architecture that leverages the code parity check matrix structural properties to realize efficient partially parallel decoding. It contains m check node computation units (CNUs) and n variable node computation units (VNUs), where each CNU_i and VNU_j perform the computations, respectively, for the p consecutive rows (or check nodes) and p consecutive columns (or variable nodes) in time-division multiplexing fashion. It contains $2n$ channel message memory blocks (CMMBs), each CMMB stores the messages associated with p consecutive columns. Two sets of n CMMBs alternatively store the channel messages for current decoding and receive the channel messages of the next block to be decoded.

Let define the set $P = \{(i, j) \mid \forall\ H_{i,j} \text{ is non-zero}\}$ and $|P|$ denote the total number of non-zero circulants in the parity check matrix. The $|P| \cdot p$ decoding messages are stored in $|P|$ decoding message memory blocks (DMMB), each $\text{DMMB}_{i,j}$ stores the p messages associated with the p 1's in $H_{i,j}$. The decoder contains n p-bit hard decision memory blocks (HDMB) to store the hard decision bits. The access address of each CMMB_j, HDMB_j or $\text{DMMB}_{i,j}$ is simply generated by an individual binary counter. Each CNU_i connects with all the $\text{DMMB}_{i,j}$s with the same index i, and each VNU_j connects with all the CMMB_j, HDMB_j and $\text{DMMB}_{i,j}$

Petitioner Micron Ex-1021, 0420

with the same index j. The decoding process consists of an initialization phase and an iterative decoding phase as described in the following.

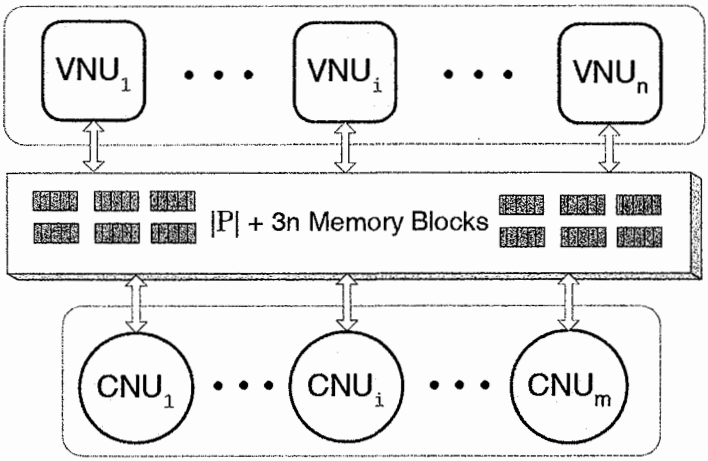


Fig. 14.16. A straightforward realization of partially parallel QC-LDPC code decoder

Initialization: Upon the received data, CMMBs are initialized to store the channel messages associated with all the variable nodes. The content of each CMMB_j is copied to all the $\text{DMMB}_{i,j}$s with the same index j as the initial variable-to-check messages.

Iterative Decoding: Each decoding iteration is completed in $2p$ clock cycles, and the decoder works in check node processing mode and variable node processing mode during the first and second p clock cycles, respectively.

- During the **check node processing**, the decoder performs the computations of all the check nodes and realizes the message passing between neighboring nodes in the code bipartite graph. In each clock cycle, each CNU retrieves one variable-to-check message from each connected DMMB, converts it to one check-to-variable message, and sends the check-to-variable message back to the same DMMB. The memory access address of each $\text{DMMB}_{i,j}$ is generated by a binary counter that is initialized to the right cyclic shift value of the non-zero circulant $H_{i,j}$ at the beginning of check node processing.
- During the **variable node processing**, the decoder performs the computations of all the variable nodes. In each clock cycle, each VNU retrieves one check-to-variable message from each connected DMMB and one channel message from the connected CMMB, converts each check-to-variable message to variable-to-check message, and sends it back to the same DMMB. The memory access addresses of each memory block are generated by the counters that are set to zero at the beginning of variable node processing.

Petitioner Micron Ex-1021, 0421

Many other partially parallel QC-LDPC code decoder architectures have been developed and interested readers are referred to [22] and references therein.

14.7.3 QC-LDPC code performance evaluation

To demonstrate the error correcting performance of QC-LDPC code as ECC for NAND Flash memory, this section presents a case study for 2 bits/cell NAND Flash memory with 4 kB page size. We construct a regular rate-15/16 (34976, 32790) QC-LDPC code with the parity check matrix column weight of 4. The code parity check matrix contains an array of 2 × 32 circulants, where all the circulants have a column weight of 2 and are constructed randomly subject to the 4-cycle free constraint.

Throughout the discussion, we will present corresponding evaluation results using the following Flash memory cell threshold voltage distribution model. The erase state has a Gaussian-like distribution, i.e., its probability density function (PDF) of the threshold voltage distribution can be approximated as

$$p_0(x) = \frac{1}{\sigma_0\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma_0^2}}$$

where σ_0 is the standard deviation and μ is the mean threshold voltage of the erase state. All the other states tend to have the same threshold voltage distribution, as illustrated in Fig. 14.17. It consists of two parts, including a uniform distribution in the middle and Gaussian distribution tail on both sides. The width of the uniform distribution equals to the program step voltage ΔV_{pp}, and the standard deviation of the Gaussian tail is denoted as σ. The Gaussian distribution on both sides models the overall effect of noises in Flash memory.

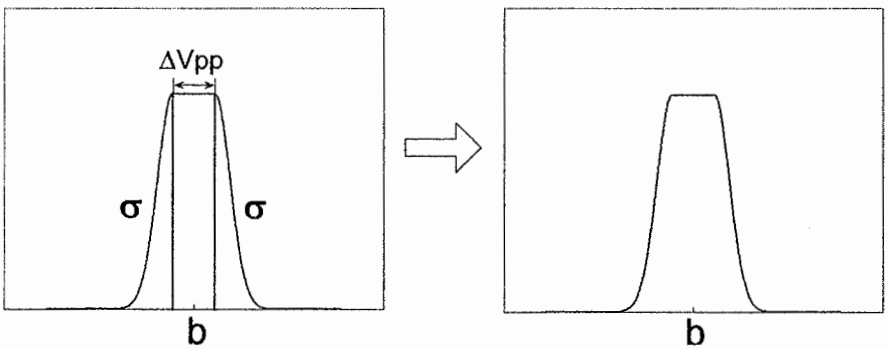


Fig. 14.17. Threshold voltage distribution model NAND Flash memory (except the erase state)

Petitioner Micron Ex-1021, 0422

Let P_0 and P_1 denote the probabilities of the uniform distribution and Gaussian distribution, respectively. The overall probability density function (PDF) $f_{pr}(x)$ can be written as

$$f_{pr}(x)=\begin{cases}\dfrac{c}{\sigma\sqrt{2\pi}}, & b-0.5\Delta V_{pp}\le x\le b+0.5\Delta V_{pp}\\ \dfrac{c}{\sigma\sqrt{2\pi}}e^{-\frac{(x-b-0.5\Delta V_{pp})^2}{2\sigma^2}}, & x>b+0.5\Delta V_{pp},\\ \dfrac{c}{\sigma\sqrt{2\pi}}e^{-\frac{(x-b+0.5\Delta V_{pp})^2}{2\sigma^2}}, & x<b-0.5\Delta V_{pp}\end{cases}$$

where b is the mean of the threshold voltage (i.e., the center of the distribution as shown in Fig. 14.18), and the constant c can be solved based on

$$P_0+P_1=\int_{-\infty}^{+\infty}f_{pr}(x)dx=1.$$

Since LDPC code decoding requires soft-input, we assume that each NAND Flash memory cell is sensed with a 16-level uniform quantization scheme as illustrated in Fig. 14.18, where the quantization thresholds are represented by the dashed lines.

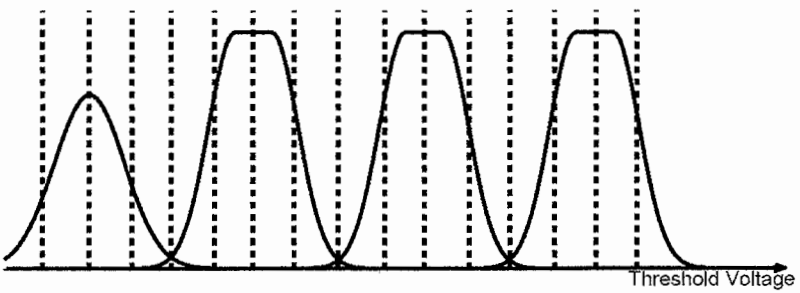


Fig. 14.18. The sensing quantization schemes for LDPC coded system

As pointed out earlier, LDPC code decoder requests the log-likelihood ratio (LLR) of each bit as the decoding input. Hence, soft-decision sensing result of each multi-level memory cell has to be translated to the LLR value for each individual bit. Let us consider n-bit/cell (or 2^n-level/cell) Flash memory. Let $\boldsymbol{s}^k$ denote the n-bit binary vector represented by the threshold voltage level k and s_i^k represent the ith bit within the vector $\boldsymbol{s}^k$, and let $f_{pr}(x)$ denote the overall PDF of Flash memory cell threshold voltage distribution, then given the sensed threshold voltage R we can calculate the LLR as

$$\ln\frac{P(R\mid x_i=0)}{P(R\mid x_i=1)}=\ln P(R\mid x_i=0)-\ln P(R\mid x_i=1)$$

Petitioner Micron Ex-1021, 0423

where we have

$$P(R \mid x_i = 0) = \frac{1}{2^n} \sum_{s_i^k = 0, k \in [1, 2^n]} f_{pr}(R), \quad P(R \mid x_i = 1) = \frac{1}{2^n} \sum_{s_i^k = 1, k \in [1, 2^n]} f_{pr}(R)$$

To carry out the simulations, we set the program step voltage ΔV_{pp} as 0.16 while normalizing the distance between the mean of two adjacent threshold voltage windows as 1. Given the value of program step voltage ΔV_{pp}, the LDPC code decoding failure rate (i.e., the sector error rate) will depend on the standard deviations of the erased state (i.e., σ_0) and the other three programmed states (i.e., σ). By fixing the normalized value of σ_0 as 0.2, simulations are carried out to evaluate the sector error rate versus normalized σ. Figure 14.19 shows the simulation results. For the purpose of comparison, a binary (34976, 32790, 136) BCH code over GF(2^{16}) is also considered, where each BCH codeword contains 34976 bits and can correct up to 136-bit errors. The results clearly show that QC-LDPC codes can largely outperform BCH codes that are being used in current design practice.

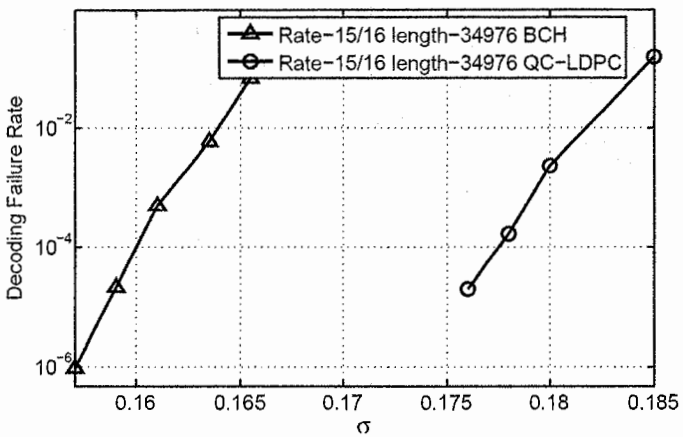


Fig. 14.19. Performance evaluation and comparison between QC-LDPC and BCH for 2-bit/cell NAND Flash memory

References

1. R. Micheloni, A. Marelli, R. Ravasio "Error Correction Codes for Non-Volatile Memories", Springer, 2008.
2. S. Lin, D. J. Costello, "Error Control Coding", Prentice Hall, June 2004.
3. T. K. Moon, "Error Correcting Coding – Mathematical Methods and Algorithms", Wiley, 2005.
4. S. B. Wicker, "Error Control for Digital Communication and Storage", Prentice Hall, 1995.

Petitioner Micron Ex-1021, 0424

5. R. C. Bose, D. K. Ray-Chaudhuri, "On a Class of Error-Correcting Binary Group Codes", Information and Contribution, vol. 3, March 1960.
6. E. R. Berlekamp, "Algebraic Coding Theory", McGraw-Hill, 1968.
7. J. Massey, "Shift-Register Synthesis and BCH Decoding", IEEE Transactions on Informaton Theory, vol. 15, Jan. 1969.
8. I. S: Reed, M. T. Shih, T. K. Truong, "VLSI Design of Inverse-Free Berlekamp-Massey Algorithm", IEEE Proceedings, vol. 138, Sept. 1991.
9. I. S. Reed, G. Solomon, "Polynomial Codes over Certain Finite Fields", Journal of SIAM, vol. 8, June 1960.
10. R. Micheloni, A. Marelli, R. Ravasio, "Method and System for Correcting Low Latency Errors in Read and Write Non Volatile Memories, Particularly of the Flash Type", US patents, US 2006/0010363 A1.
11. R. Micheloni et al.,"A 4Gb 2b/cell NAND Flash Memory with Embedded 5b BCH ECC for 36MB/s System Read Throughput Solid-State Circuits Conference", 2006. Digest of Technical Papers. ISSCC. 2006 IEEE International, Feb. 2006 Page(s): 497–506.
12. R. Micheloni, A. Marelli, R. Ravasio, "Method and System for Correcting Errors in Electronic Memory Device", US patents US 2006/0005109 A1.
13. R. Micheloni, A. Marelli, R. Ravasio, "Reading Method of a Memory Device With Embedded Errors Correcting Code and Memory Device With Embedded Error Correcting Code", US patents US 2007/0234164 A1.
14. R. G. Gallager, "Low-density parity-check codes," IRE Transactions on Information Theory, vol. IT-8, pp. 21–28, Jan. 1962.
15. V. Zyablov and M. Pinsker, "Estimation of the error-crrection complexity of Gallager low-density codes," Problemy Peredachi Informatsii, vol. 11, pp. 23–26, Jan. 1975.
16. R. M. Tanner, "A recursive approach to low complexity codes," IEEE Transactions on Information Theory, vol. IT-27, no. 5, pp. 533–547, Sept. 1981.
17. G. A. Margulis, "Explicit constructions of graphs without short cycles and low density codes," Combinatorica, vol. 2, no. 1, pp. 71–78, 1982.
18. C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in Proceedings of ICC'93, Geneve, Switzerland, May 1993, pp. 1064–1070.
19. N. Wiberg, "Codes and decoding on general graphs," Ph.D. Dissertation, Linkoping University, Sweden, 1996.
20. D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," Electronics Letters, vol. 32, pp. 1645–1646, Aug. 1996.
21. Z. Li, L. Chen, S. Lin, W. Fong, and P.-S. Yeh, "Efficient encoding of quasi-cyclic low-density parity-check codes," IEEE Transactions on Communications, vol. 54, pp. 71–81, Jan. 2006.
22. F. Guilloud, E. Boutillon, J. Tousch, and J.-L. Danger, "Generic description and synthesis of LDPC decoders," IEEE Transactions on Communications, vol. 55, pp. 2084–2091, Nov. 2007.

Petitioner Micron Ex-1021, 0425

15 NAND design for testability and testing

*Andrea Silvagni**

NAND business requires huge investments in technology developments and manufacturing. Moreover, leading edge NAND Flash memory costs and yield issues increase with every new technology node. Addressing test issues is mandatory for NAND Flash manufacturers to accelerate yield learning and enhancement, maintaining a competitive cost structure.

This chapter explores NAND Flash testing, with focus on Design for Testability (DFT) to improve test coverage and cost.

15.1 NAND architecture and testing

Seventy percent of the die size of a leading edge NAND Flash memory device is occupied by the memory array; row decoders and page buffers are the next most significant portion of the die area representing the core circuits 1. Most of the efforts in NAND Flash memory testing are related to screening out defects in the array, row decoder and page buffers where we have the most aggressive use of design rules and operating conditions. Nevertheless, the remaining 10% of the device should be tested as well, in order to verify that the peripheral circuits comply with the design specifications and guarantee the correct array operation.

Figure 15.1 shows a high level diagram of a NAND Flash memory chip. The block diagram details the X decoders which are used to select wordlines. X drivers must deal with the highest voltages in the chip and are subjected to strong voltage stress during their lifetime. Moreover, wordline drivers must be connected with the wordlines whose pitch has the tightest design rules.

Page buffers or sense amplifiers are connected to the bitlines. This block makes use of the most aggressive design rules for metal lines. Generally, there is no specific sense amplifier redundancy in the product: the only way to correct a defect in the circuitry is to use column redundancy and the sense amplifier associated with the spare columns. For a 4 kB page Flash memory, there are 4,096 sensing circuits connected to the bitlines working in parallel, plus the additional spare/ECC sensing elements. Y multiplexer selects a data bus out of the 4 k page buffer outputs.

* Forward Insights, andrea@forward-insights.com

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_15,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0426

The high voltage section of the device (comprising charge pumps and regulators) has to provide precise voltages necessary for reliable array operations during the whole device's life.

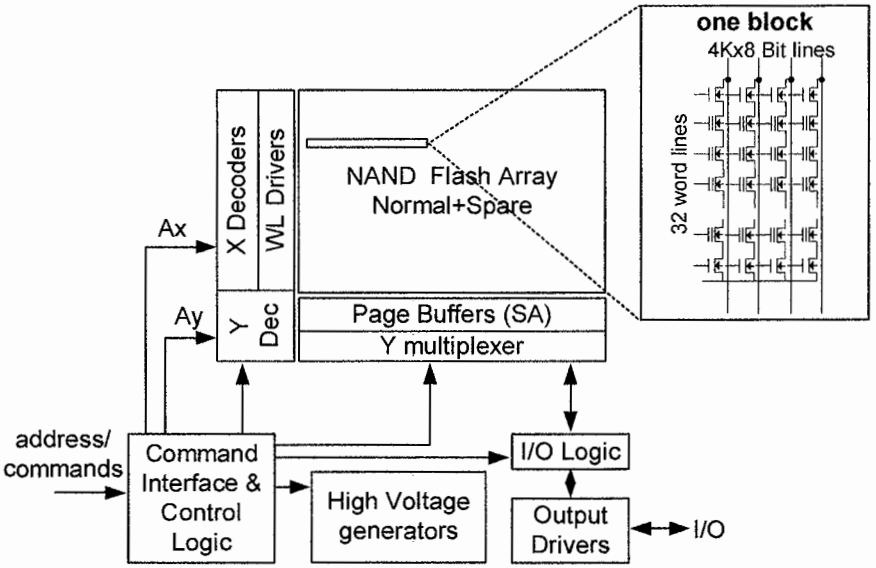


Fig. 15.1. NAND Flash memory block diagram

If we consider a write/erase cycling specification for one block of the array equal to 10 k cycles, we must be aware that the charge pumps, providing the voltages necessary for the cycling, operate in the worst (unlikely) case for 10 k cycles multiplied by the number of blocks in the array. Even if those reliability aspects are taken into account during the design phase, the manufacturer must guarantee that the production is able to maintain the reliability targets from lot to lot.

Moreover, a control logic section is used to manage all the circuits involved in memory operations. This section is frequently made up by a microcontroller plus a mask ROM and a RAM for executing code and storing data. Other implementations can be based on programmable logic arrays [3]. The digital circuitry in the chip has the same testability issues as in digital ASICs; testability can be secured by proper testing tools and accessibility such as digital scan chains in order to enable the highest quality levels and minimum testing time.

The aim of the following sections is to provide an introductory view of the test issues related to array architecture and peripheral circuits. For detailed descriptions of the circuits, it is recommended to refer to specific chapters in this book and bibliography. Specific test mode descriptions will be given in Sect. 15.4.

Petitioner Micron Ex-1021, 0427

15.1.1 Array testing

The first goal of array testing is to detect and repair array defects by means of redundancy and bad blocks.

Figure 15.2 shows a picture of a portion of NAND array where it is possible to identify, in the bitline direction, a 32 cells string ended by the bitline contact.

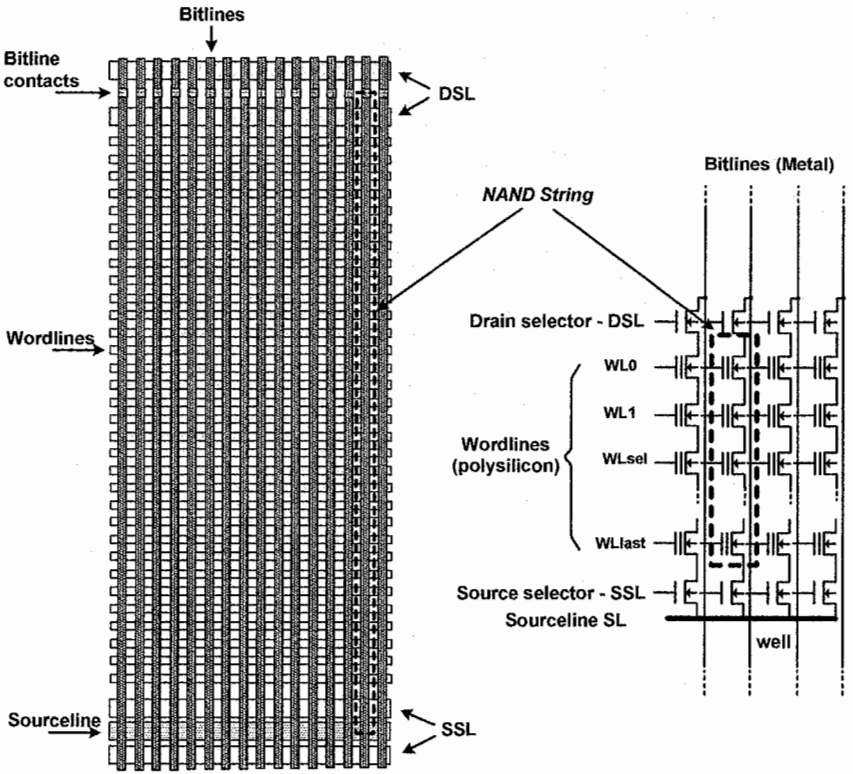


Fig. 15.2. NAND array layout and schematic

The bitline contact connects the string to the common bitline metal. Every thirty-two polysilicon wordlines (string boundary), the bigger select lines for drain and source selectors (DSL, SSL), near the bitline contacts, are shown.

It is worth to note that different array choices can be done in terms of layer types, number and arrangement used to improve the NAND operations [14–17]. However, they are strictly related to process capabilities and costs. A three metal level array is common in leading edge MLC devices. New materials for bitlines, wordlines and dielectrics as well as the already mentioned metal arrangements for the lines are being researched in order to improve performance. Those new concepts, when adopted, cause new challenges in the ramp-up and production of the memory.

Petitioner Micron Ex-1021, 0428

The main causes that reduce the production yield are summarized below.

- Shorts between bitlines
  - Caused by metal shorts and bitline contact shorts
  - Repairable by column redundancy
- Shorts between wordlines
  - Caused by poly line shorts
  - Repairable by Bad Block Management
- String open
  - Caused by bitline contacts open, bitline metal open, AA open
  - Repairable by column redundancy
- Bitline wordline shorts
  - Caused by interlayer defects
  - Repairable by Bad Block Management

Secondary factors are related to capacitive coupling structure in the cell leading to anomalous cross coupling and low cell gain. Those effects are more severe in multilevel operations and are treated in combination with error correction codes.

Effective defect detection tests are mandatory to have fast technology learning, yield enhancement and test cost reduction. Some details in array screening test modes will be described in Sect. 15.4.

15.1.2 High voltage pumps testing

Figure 15.3 sketches a block diagram with the main voltage generators inside the memory and the related signal path.

All the voltage signals used to bias the wordlines are connected to a switch block which routes the correct value to the proper wordline in the string. The outputs of the switch block are Global Wordline (GWL) signals with cardinality equal to the number of wordlines in a string (e.g. 32 or 64).

The switch block is controlled by the addressing logic. For testing purposes, it should be possible to apply proper voltage patterns to the GWL in order to perform specific tests such as activation of even and odd wordlines; therefore, maximum flexibility is required in the switch block for testing.

Most of the charge pumps and regulators have trimming options and the voltage outputs are wired (through the switch block in Fig. 15.3) to special test pads in order to verify and correct their voltage value. In the following Table 15.1 there is an example of the voltages used inside a NAND memory:

The output values of the charge pumps must be tested and, in some cases, can be trimmed by means of non-volatile registers (Fuse ROM).

Charge pumps are designed to be able to charge/discharge a capacitive load within a specified time. The stability of such timings is of fundamental importance in the total program/erase time: those timings are a significant portion of the program algorithm because the voltages must be stable before performing write pulses and verify operations. Therefore, appropriate stress acceleration can be useful to screen out the degradation of the performance of the charge pump circuits.

Petitioner Micron Ex-1021, 0429

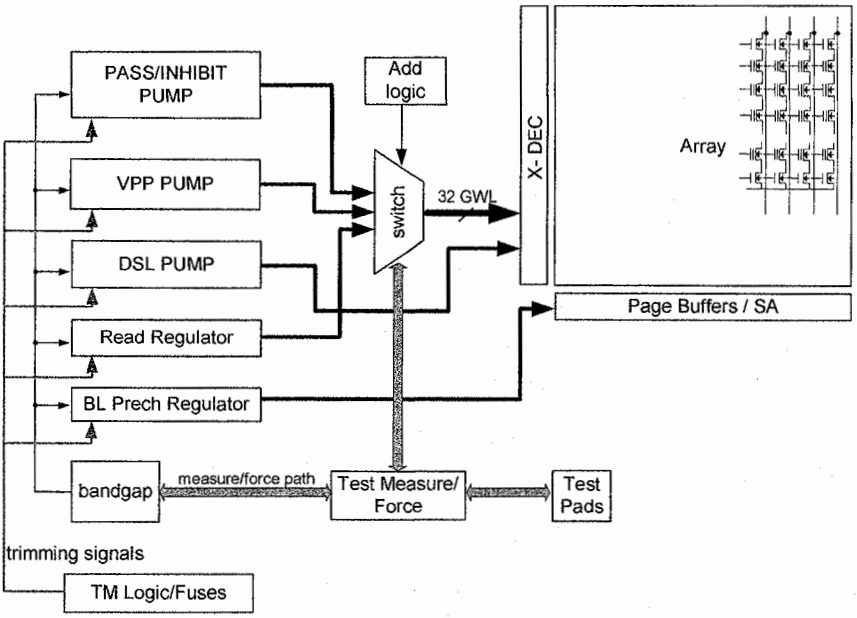


Fig. 15.3. Voltage generation and distribution block diagram

Table 15.1. NAND voltages

	Value [V]	Phase			Use	Characteristics
		Read or verify	Prog	Erase		
Program	16–24		X	X	WL Sel (prog) Well (erase)	High voltage
Inhibit	4–11	X	X		WL Unsel	High capacitive load
DSL/SSL	2–5	X	X	X	DSL SSL	
SSL	3–5	X	X	X		
V_{READ}	0– (SLC) 0–4 (MLC)	X			Selected WL	Precision, temp dependence
Vprech	1.3	X			Selected BL	Very high load

15.1.3 Read circuitry testing

Few words must be spent on the read circuitry called Page Buffer or Sense amplifier, because its functionality is fundamental in many testing operations inside the array. Page buffers are covered in Chap. 8.

Petitioner Micron Ex-1021, 0430

As a result of back-end density in the page buffer, metal shorts and open defects are the most important yield detractors in this part of the device. Detection of stuck-at faults in the page buffers is a primary concern; therefore, special tests should be designed to load and read back the page-buffer latches with dedicated patterns. In MLC devices the sensing threshold is also fundamental and should be tested.

Typically, we can have either a SLC page buffer built with one latch without cache or two latches with a cache function. In MLC at least two latches are needed, but usually a third one is adopted.

It is important to highlight, from a test perspective, that a double latch structure can be used efficiently in testing the memory for data loading and error accumulation. Some features - such as the ability to move data from one latch to the other – are fundamental to optimize special test modes (Sect. 15.4).

15.2 NAND Flash memory testing introduction

In this section the NAND Flash testing aspects are analyzed in a design for testability perspective. Testing is referred to as the group of activities which are performed on the NAND Flash based products, during the whole productive cycle. This includes the first silicon testing and the mass production testing, passing through the ramp-up/yield-enhancement pre-production phase. Test flow design consists of the definition of testing sequences necessary to screen the defective or poorly performing devices from wafer level up to the single component level or even module level. Test flow design aims at obtaining the best coverage to avoid failures in the final application, improving productivity and reducing the test time and more generally the test cost. Design for Testability (DFT) efforts are finally oriented to obtain a better test coverage and lower production test-costs and will be treated in Sect. 15.3.

15.2.1 Test phases: first silicon, ramp-up, production

Generally, the testing of a new product is divided into three phases.

1. First silicon phase. During this phase, a test flow aims at obtaining initial process/product assessments; massive use of special test modes can be applied to perform preliminary characterization of array and circuits and to collect data. In this phase the testing time is not so important, whereas the accessibility to the memory is.
2. Ramp up phase. During this phase, the test parallelism is increased, test flow has full coverage, redundancy is implemented and dedicated testers are introduced.
3. Production phase. During this phase, test time reduction and yield enhancement are addressed.

Petitioner Micron Ex-1021, 0431

The best execution of the three phases is measured by the payback curve of the product. The three phases have different weights, depending on the three cases described below.

A. The first case is when a new technology node is developed. Usually in this case an established architecture or test-chip is used to develop the lead driver for the technology. Some new DFT features could be added to the memory but, generally, the product architecture is very close to one already in production whose test environment can be reused. In this way the test flow debug time is reduced as much as possible and information on the technology are gathered earlier.
B. The second case is when a new product architecture is introduced, generally based on established technology. In this case, newly designed test mode features are used, during the first silicon phase, to make the device working and to gather early functionality information.
C. The third case is when a cut-down or derivative product is produced. In this case the minimum risk should be pursued in order to have first silicon success and a fast ramp up phase, using the established test environment from the parent device.

15.2.2 NAND Flash test flow introduction

Test flow is the term we use for a sequence of tests, in order to screen the material coming from the production and select the devices which are compliant to the electrical and reliability specifications [2].

NAND test flow is divided into Front End or Wafer Level (also named EWS – Electrical Wafer Sorting) and Back End or component flow (Final Test). Some applications require module testing when the memory chips are mounted on specific modules. A burn-in step is required in order to screen out early life defects. The basic NAND Flash test flow (Fig. 15.4) will be described in more details in the following sub-sections.

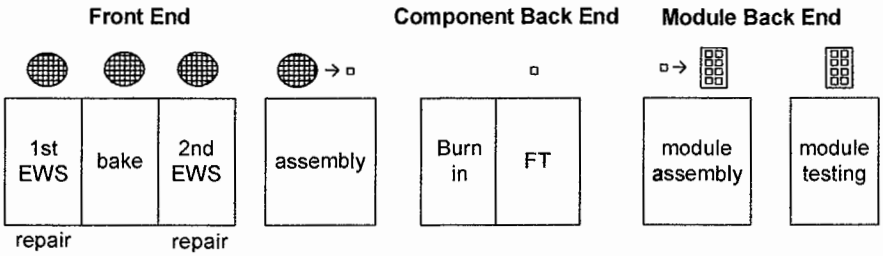


Fig. 15.4. Basic NAND Flash test flow

Petitioner Micron Ex-1021, 0432

Wafer level flow

NAND wafer level test flow or *Electrical Wafer Sort* (EWS) is generally divided in two steps with retention bake in between. First EWS is a test flow used to detect early defects, to test cell/blocks functionality and to apply stress tests. Repairable array defects, such as column shorts, will be repaired by column redundancy whereas other defects, such as wordline shorts, will be handled with bad blocks.

The basic first level flow is summarized below.

1. DC and contact test
2. Page Buffer functionality tests
3. Early cycling
4. Erase pre conditioning and read
5. Early defect detection tests
6. Programmability test and read
7. V_{PASS} test
8. Stress tests to activate defects
9. Checkerboard/Inverse Checkerboard program and verify
10. Redundancy summary and fusing
11. Precondition for bake

Since many tests involve array – and test time is proportional to array capacity – special test modes must be designed in order to reduce the test time. Such tests will be treated in Sect. 15.4: e.g. parallel program or erase test modes, fast read test modes, defect detection test modes and fail accumulation.

Good wafers, i.e. wafers with redundancy and bad block number below set limit, are sent to the bake phase: a proper activation temperature, between 150°C and 250°C, is used to identify retention failures due to mechanisms with activation energy higher then 0.6 eV [4, 19, 20].

Afterward, the second electrical wafer sort (EWS2) takes place. It aims at screening out the retention capability of the cells in the array and performing additional tests.

The basic second level test flow is summarized below.

1. DC and contact tests
2. Bake readout
3. Diagonal pattern to check for decoder defects
4. Basic User Mode test
5. Redundancy summary and fusing

Back-end flow

The main focus for back-end test or final test is to guarantee that device parameters are within the specified margins. Moreover, the final test detects assembly induced failures. Measurements are performed at low temperature and high temperature according to the specification grade.

Petitioner Micron Ex-1021, 0433

The main steps in component level testing are listed below.

1. Measurement of leakage, standby and operating currents. Chip ID readout.
2. Functionality and retention tests under extended power supply voltage conditions.
3. Dedicated tests to cover design and process stability.
4. At high temperature, the AC parameters are tested and the speed grade of the components is determined.
5. Readout of chip ID for single chip traceability and storing of measurement results for yield analysis.

Burn-in

Burn-in is a special test section used to induce and detect early life defects and improve the customer field failure rate (Fig 15.5). Burn-in is usually run at high temperature, i.e. 125°C, in special ovens with the highest possible parallelism. It is generally composed of multiple program-verify-erase cycles.

Applying physical acceleration means using the device in stress conditions (i.e., higher temperature or voltage or humidity or duty cycle, etc.) in order to observe the degradation effects of the process at an earlier time [10, 19, 20]. Stresses commonly used are voltage and temperature. The acceleration factor AF is a constant value that multiplies stress time t_{stress} returning the equivalent use time t_{use} in the standard conditions as shown in Eq. (15.1). Voltage acceleration AF_V is the most important f actor, but thermal acceleration AF_T is important to activate some defects and to accelerate the test time.

$$AF = AF_V(V) \cdot AF_T(T); \quad t_{use} = AF_V(V) \cdot AF_T(T) \cdot t_{stress} \tag{15.1}$$

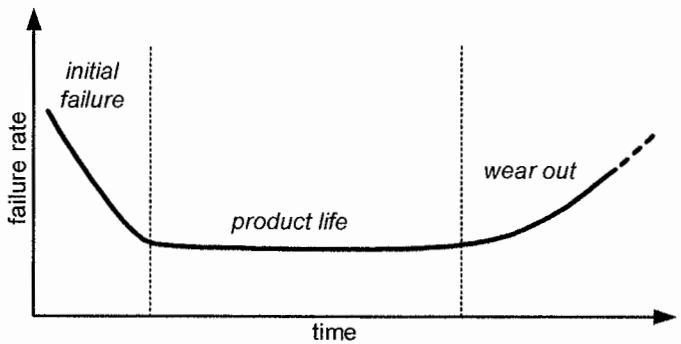


Fig. 15.5. Screening initial life defects by burn-in

Petitioner Micron Ex-1021, 0434

Table 15.2 shows an example of voltage and temperature acceleration factors.

Table 15.2. Stress acceleration factors example (Arrhenius accelerating temperature model with Activation Energy 0.8 eV, exponential accelerating voltage model with factor $\beta = 9$)

Stress temp	Used temp	AF_T	Stress V	Use V	AF_V	$AF_T * AF_V$
125°C	70°C	42	3.6 V	2.8 V	1.340	56,280

Voltage acceleration is much more effective especially in the activation of high voltage points. Burn-in is fundamental to activate package-related stresses such as the chemical or mechanical ones.

Burn-in is very expensive; therefore, an effective and correct burn-in strategy is fundamental in the cost containment of the test equipment and test time.

- Stress voltages and temperatures must be carefully (individually) controlled to avoid overstress damaging of the circuits and to target the right stress level achievable considering activation energy for the different goals.
- The voltage stress that obtains sufficient voltage acceleration can be moved to wafer level test. Burn-in should focus on the packaging process (inducing mechanical and chemical stresses).
- When voltage/temperature stress levels can only marginally increase from the nominal values and increased burn-in times are necessary, DFT should focus on removing those limitations.
- Wafer level burn-in should be considered by leveraging advanced wafer prober and introducing a third wafer level test step.

Some burn-in addressed failures are associated with electrical opens or shorts as summarized below:

- Metal shorts (line width fluctuations)
- Masking of metal lines
- Intrinsic defect density
- Contact shorts

In addition, there are assembly-related fails due to filler imprints:

- Mechanical stress (Temperature cycling)
- Damage of upper metal levels due to filler particles

Burn-in requirement in the test flow is strictly related to the application. For example, for Flash cards the target defect part per million (dppm) is not so stringent, and burn-in could be omitted. In Flash for SSDs the dppm is much lower and burn-in is necessary.

Burn-in flow components are:

- Proper sequence of read, program and erase operations
- Stress on power supply

Petitioner Micron Ex-1021, 0435

- Stress on wordlines
- Stress on bitlines

Since burn-in is usually done with very high parallelism, it is useful to adopt a *Built In Self Test* (BIST) strategy so that each device runs the burn-in flow independently, and there is only the need to perform the final readout of the test for all the loaded devices after a timeout period. BIST could be easily implemented and customizable if the NAND product adopts a microcontroller architecture to control the operations (Sect. 15.3.7 and Chap. 6). In this case, special test flows can be stored in the ROM, customized by parameters and executed by special test modes or test commands. Some pre-defined test sequences can also be embedded in a programmable logic array structure.

Finally, the correct strategy depends on the target application requirements. For example, defect target for USB/SD Cards is low (e.g. 2,500 dppm) so that ECC and Bad Blocks can give a sufficient tolerance to "small" defects. Moreover, use case of application is not so demanding (e.g. Flash cards in Digital Still Cameras experience some program/erase cycles at room temperature). The above mentioned considerations can make the delivery without burn-in possible. For SSDs instead, the high number of dies and the higher MTBF make the target defect lower and the burn-in strategy is recommended.

15.2.3 Test flow and test time optimization

In the first decade of 2000 the NAND Flash market experienced an acceleration in the available memory density and number of bits delivered to the market. A big issue in cost structure was that test costs increased as chip density increased, while Average Selling Price (ASP) of NAND decreased. Leading edge NAND devices take days to be tested and test time increases with capacity. Multi Level Cell (MLC) devices aim at reducing the memory cost by increasing the bit density but encounter limitations in the reliability and require a more severe and effective test screening.

Addressing the test time issue is mandatory for NAND manufacturers which lag behind logic chips manufacturers in terms of Built In Self Test hardware and standard protocols [7]. In the latest years, NAND manufacturers have been forced to increase the research in Design for Testability (DFT) in order to improve the test costs and manufacturability, but those efforts are generally proprietary. Automated Test Equipments (ATE) systems are customized to support non-volatile memory testing requirements, leading to the increase of the ATE test costs.

Many steps are necessary to reduce test costs and increase productivity. A big step in cost reduction comes from the parallelism of testing. High parallelism in wafer level and component level testing is the primary research field for manufacturers of ATE suppliers and DFT. Despite probe card manufacturers offer the possibility to test an entire 300 mm wafer in a few or even single touchdown [21], the limitations of parallel memory test come from tester resources that have to keep track of redundancy repair requirements. In addition, the more non-volatile

Petitioner Micron Ex-1021, 0436

memory chips are tested in parallel, the more non-deterministic behaviour of memory chips becomes important. The programming time varies from page to page or chip to chip. In fact, the memory testers have to wait for the slowest memory before proceeding with the next steps [9]. In order to exploit the full potential of parallel testing, memory manufacturers are increasing the level of DFT by increasing the internal parallelism (i.e. the ability to perform write/erase operation on more pages/blocks) and utilization of BIST functions in order to reduce interaction with the tester (Sect. 15.3.7).

The tester limitations also come from resources and, in this case, the DFT efforts are allocated to reduce the tester resources needs in terms of electronic pins. Low Pin Count testing (LPCT) is a testing practice used to enable parallel die testing. To enable the maximum test parallelism it is necessary to reduce the number of I/Os to be tested down, at least, to four I/Os from a byte- or word-standard user interface (Sect. 15.3.2). Power supply units remain a hot topic in limiting the further increase of test parallelism. These themes will be described in next sections.

Test flow analysis is always under research for optimization as well as to reduce total production costs and many possibilities are offered by new tester technologies and DFT techniques. The adoption of such technologies can lead to test flow modification. For example, a way pursued in order to improve cost, is to move the highest possible number of tests at wafer level. Tests traditionally performed on packaged parts, if moved to wafer level, would allow early detection of failures and save further test costs and – more important – the packaging costs. Stacked memories benefit from this strategy because the yield of a stacked device is the product of the yield of each die in the package. High temperature tests, once relegated to packaged devices, are now possible at wafer level with modern probing technologies. Wafer level burn-in operations can help in saving burn-in costs and packaging of bad dies.

To reduce the test time further, it is necessary to optimize two principal components of the test time:

1. Write/erase/read time
2. Data-in time and fail map acquisition time

As far as the first point is concerned, write and erase times are fixed by the nature of the write/erase mechanism, the minimum write/erase chunks (i.e. page size, sector size) and the device architecture (double/single plane). By means of special DFT features and test modes, it is possible to optimize the write/erase times in several points of the test flow as described in Sects. 15.2.2 and 15.4. Parallel write/erase and, in some cases, read operations over multiple pages or blocks are used for this purpose.

As regards the second point, it is possible to use DFT for reducing the test time by means of Data/Fail compression in order to save most of the data readout time from the DUT. In Sect. 15.4 it is described a test mode useful to accumulate fails during verify operations. In this case the compressed fail information is downloaded to the tester fail memory instead of the whole data page, saving data-out time.

Petitioner Micron Ex-1021, 0437

15.2.4 KGD testing

Known Good Die (KGD) Testing is an increasing practice; manufacturers deliver the entire wafer to customers together with the fail map information. The target of KGD is to deliver memories with greater coverage than generally done at EWS testing by performing on wafer additional tests, usually done at a later stage. The first problem with KGD testing is to guarantee speed performances usually tested at package level, facing issues in signal integrity and measurements. The second problem is the need for accurate thermal tests on wafer. Both the described issues require advanced probe technologies and investments but they can be useful in the perspective of reducing costs, by moving tests as much as possible from the back end of line, as discussed in Sect. 15.2.3.

15.2.5 BAD Blocks Management

NAND memory devices make use of the so called Bad Blocks Management (BBM) to obtain the highest silicon yield. When the defects in array are not repairable by column redundancy (for example, when shorts occur between two adjacent wordlines or the defect is larger than the column redundancy capability) then, the entire Block is marked as Bad and managed appropriately. A first approach in NAND redundancy strategy is to replace the bad block with extra spare blocks available on chip in a transparent way to the user, adopting a block redundancy strategy (Fig. 15.6a). This strategy obviously has an area impact because of the added spare blocks, but it can payoff in certain cases.

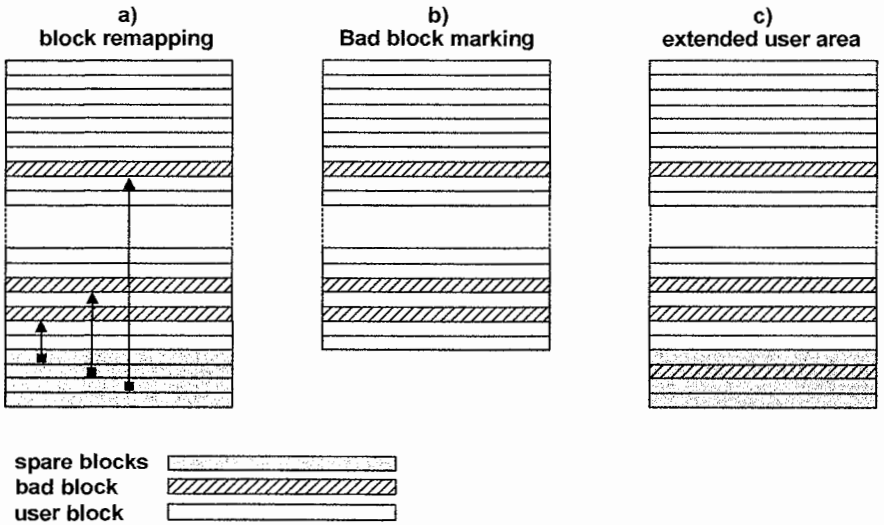


Fig. 15.6. Bad Block Management (BBM)

Petitioner Micron Ex-1021, 0438

A common approach in NAND products for data storage application is to classify the memory devices into classes, according to the number of failing blocks, and sell them with defective blocks. In this case, a maximum number of Bad Blocks is guaranteed to the user, and the File System software has to handle those blocks properly. The last option is to offer additional spare blocks to extend the addressable memory (Fig. 15.6c). Those added spare blocks can be treated as standard blocks by the File System and are offered as extra memory to the customer.

Bad blocks can be recognized by the system software because they have a special marking programmed at the factory, manufacturer dependent. For example, if the first byte of the first or the last page in the block is 00h, then the block is recognised as bad and properly treated by the File System. Industry efforts aim at standardizing the Bad Block identification [5].

Bad Blocks (BB) can be identified during the test flow and require proper management by test programs. Typically, most of the Bad Blocks are detected in early tests during the first wafer level touch down, when bigger defects are found. During testing, the bad block table is collected by the tester fail memory in order to apply the redundancy strategy at the end of the flow and finally mark the bad blocks. The strategy for BB management in testing is not straightforward because, for example, it is not possible to mark bad block information in a page when we don't know if the page can retain the information. For this reason, the simplest and most effective strategy must be put in place taking into account the available hardware on chip and the tester limitations.

During the first wafer level test, the BB map is collected in the tester. At the end of the flow the tester provides fusing information to be stored permanently in the device in the following fusing process.

With the second wafer level testing after fusing – when the most important defects have been screened and first retention tests have been done – the testing flow proceeds and uses internal memory volatile registers (or SRAM) to update the bad blocks map that will continue to accumulate new-found bad block fails. This approach avoids using the tester resources to store information. The ultimate goal is to avoid multiple fusing processes in order to reduce the test time and to take advantage of the non-volatile nature of the memory (which is not possible with DRAM). For this approach it is possible to use a good spare block or a standard block as support for storing bad block table and possibly other test information that will be used in the successive test steps. The block identified for this purpose can be marked as Bad Block and will never be used by the customer. By using this strategy all the successive testing steps will update the bad block table.

15.3 NAND DFT

Design for Testability (DFT) concerns all design efforts done to ease testing. Main objectives of Design for Testability are:

Petitioner Micron Ex-1021, 0439

- Improve productivity and gain competitiveness (test time reduction = test cost reduction)
- Increase test parallelism
- Yield enhancement
- Maximize test coverage to avoid failures at the customer
- Production process monitoring and optimization

DFT guarantees the appropriate hardware necessary to enable special test modes in order to:

- Evaluate array functionality
- Evaluate general product functionality
- Accelerate tests and stress patterns
- Modify electrical parameters
- Identify and investigate product fails
- Quantify operational windows
- Characterize internal sub-circuit margins
- Set burn-in operation conditions
- Generally reduce test costs

In the following sub-sections the basic hardware requirements in order to enable the test flows (Sect. 15.2) and test modes (Sect. 15.4) are described.

15.3.1 Special test pads

In addition to standard pads, special test pads are necessary to enable test mode features. For example, some pads could be added to enable low pin count testing at wafer level. At least one extra pad could be used to select between user mode and test mode without the need to provide special test sequence on the byte wide interface. When test mode is enabled, special commands can be given to the test interface to enable the test features.

A fundamental usage of the test mode selection is to enable the low pin count features (Sect. 15.3.2) directly at wafer level. For example, by setting the x2 I/O mode, only two pin tester resources are needed instead of eight to probe the DQ data, thus enabling a greater parallelism and saving touch down time. Table 15.3 shows an example of test pads (TE1, TE0) used to enable test mode in low pin count configuration.

Table 15.3. Test enable pins example

TE1	TE0	
0	0	Test mode x8 enabled
0	1	Test mode x4 enabled
1	0	Test mode x2 enabled
1	1	User mode

Petitioner Micron Ex-1021, 0440

At least one additional pad is required to measure internal voltages with the purpose of calibration or test coverage. The same pad can also be used to force some internal analog signals from the tester. This is useful in order to have a controlled voltage during special tests or to enhance test speed when internal generators are not able to provide enough current loads. Figures 15.7 and 15.8 show drawings with two test pads: one used to test internal low voltages and the second one to test high voltages. In some cases it could be convenient to share the pads with some standard NAND interface pads in order to enable the features on the packaged parts. The primary function of the low voltage pad is to access the internal band-gap circuit (Sect. 11.4). The band-gap trimming is a fundamental operation since the band-gap reference voltage is used to generate all the internal voltages. The same pad can be used with the purpose of measuring other reference signals or the output of voltage generators such as the V_{INT} down-converter (Sect. 11.5) [13]. Many signals (V1, V2, V3 ...) can be monitored for the purpose of trimming, coverage or debug.

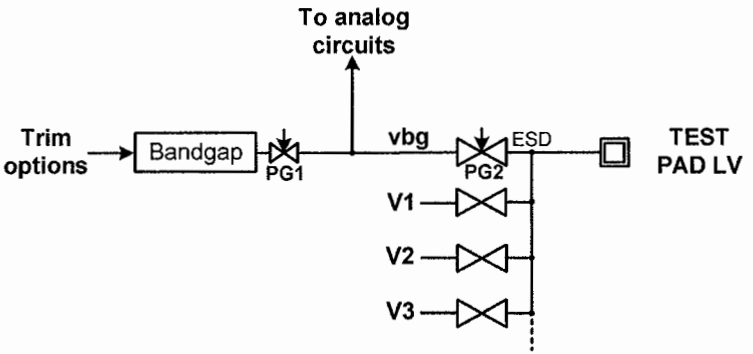


Fig. 15.7. Low voltage test pad for measurements

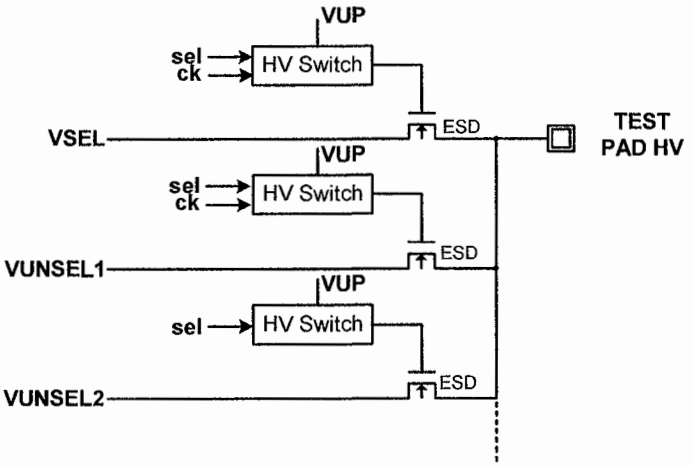


Fig. 15.8. High voltage test pads for measurements

Petitioner Micron Ex-1021, 0441

The function of the high voltage pad is to monitor and trim internal voltage regulators related to the array operation such as the read gate voltage, especially in MLC devices.

The test pad operation can be reversed in order to force the voltages from external test equipment in test/debug mode. For example, the internal band-gap voltage can be replaced by the external pad voltage by disabling pass gate PG1 and enabling pass gate PG2 (Fig. 15.7).

Table 15.4 summarizes the main internal voltages that could be connected to the external pad with the purposes of monitoring/forcing, trimming or debugging.

Table 15.4. Example of high and low voltage signals measurable with external pad

Name	Range [V]	Description
V_{BG}	1.25	Band-gap voltage
V_{PP}	0–25	Selected WL in Program , nwell/pwell in Erase
V_{INT}	1.7–2.0	Internal logic supply voltage generated from DC–DC converter
$V_{INHIBIT}$	0–15	Unselected WLs in Program (selected string)
V_{RD}	0–5	Selected WL in Read or Verify
V_{PASS}	0–8	Unselected WLs in Read
V_{SSL}	0–7	Gate of Source switch
V_{DSL}	0–7	Gate of Drain switch
V_{SA_PRECH}	0–3.6	Gate of the follower for bitline precharge in Read
V_{BL_PRECH}	0–5.5	Bitline pre-charging in Program during inhibit phase

Moreover, the most important regulated voltage signals (which have an impact in precision and performance) can be measured and trimmed through the pad. Tester limitations should be taken into account (i.e. if a maximum 15 V can be measured by the tester equipment then, higher voltage values, such as VPP, cannot be brought directly to the test pin, but only a fraction could be tested).

15.3.2 Low Pin Count Testing (LPCT)

Low Pin Count Testing (LPCT) is a testing practice widely used in ASICs and logic chips; it is gaining more and more importance in memories in order to reduce the test cost by achieving higher testing parallelism. LPCT can be used at wafer level in combination with high parallelism capability of advanced tester and probe equipments. Today, advanced memory testers are capable of a single touchdown testing on 300 mm wafers. To enable this parallelism it is necessary to reduce the number of I/Os to be tested.

Petitioner Micron Ex-1021, 0442

The low-pin count interface is enabled by means of the test interface or special test pads (Sect. 15.3.1). The data interface can therefore be reduced to single bit, double bits or nibble I/O interface. The data or commands transfers are executed in more cycles. Figure 15.9 illustrates an example of x4 I/O interface where two write cycles are needed in order to get the command word/data-in in a byte-organized NAND Flash memory (Table 15.5). The DIN are then internally re-aligned with a proper internal clock signal.

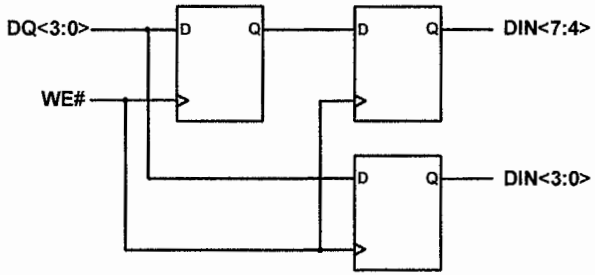


Fig. 15.9. LPCT x4 interface

Table 15.5. Data in sequence in LPCT x4

WE# cycle	DQ<3:0>	DIN <7:4>	DIN<3:0>
Cycle 0	'7''6''5''4'	'x''x''x''x'	'7''6''5''4'
Cycle 1	'3''2''1''0'	'7''6''5''4'	'3''2''1''0'

While reducing the number of I/Os, the time to transfer data from the memory to the tester and vice versa increases (doubles when switching from byte-wide to nibble-wide). This is an additional reason to avoid data transfer as much as possible during the test flow and to increase the use of data compression and BIST techniques that will be analyzed in the next sections.

15.3.3 Voltage regulators and trimming

Many voltage regulators and charge pumps in the NAND product require proper testing. Section 15.3.1 showed that the principal internal analog signals can be accessed through a test pad. The trimming procedure is fundamental for the band-gap circuit, providing a reference voltage for almost every other analog signal in the NAND memory. Moreover, every other charge pump or important voltage regulator inside the chip has the possibility to be further adjusted. The example in Fig. 15.10 shows a charge pump whose regulator is based on a V_{REF} (band-gap) reference voltage.

Petitioner Micron Ex-1021, 0443

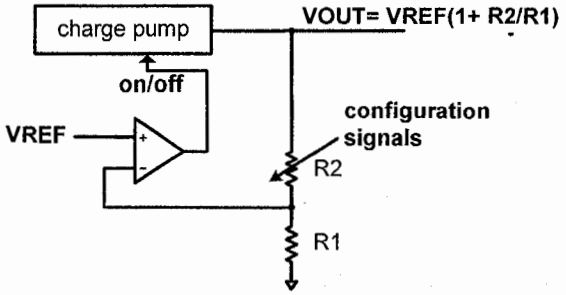


Fig. 15.10. Charge pump trimming

Besides, by means of configuration signals it is possible to adjust the resistor ladder to fine tune the pump output.

Band-gap reference has the capability to provide an excellent voltage reference which is voltage independent with a precision of few parts per million in the temperature range of interest. The importance of this circuit is so high that it is essential that the circuit is verified and trimmed at wafer level testing and that a Parametric Measurement tester Unit (PMU) is used.

The primary goal of the trimming procedure is to compress the native Gaussian distribution of the band-gap voltage circuit output into a narrow distribution. The purpose is to reduce the error when generating the highest voltage signals which apply a high multiplication factor to the band-gap reference voltage as shown in Fig. 15.11.

Specific configuration bits are available to properly set the band-gap circuit with a measure and trim procedure as shown in Fig. 15.12, making use of the configuration bits as shown.

Multilevel cell architectures require more and more precision as the number of levels increase. As a consequence, band-gap trimming is required to be more accurate and finer.

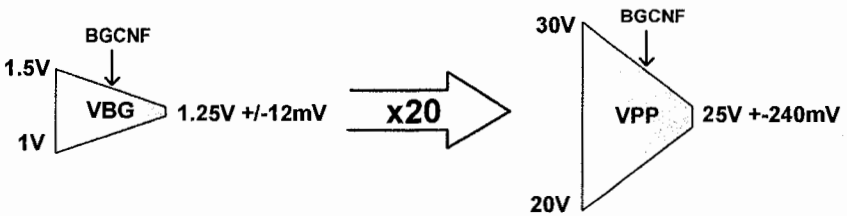
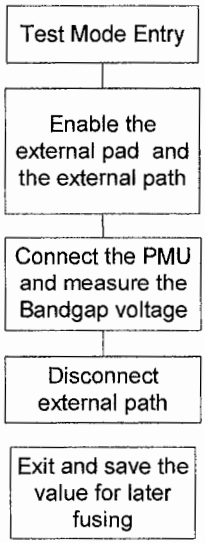


Fig. 15.11. Band-gap trimming

Petitioner Micron Ex-1021, 0444



BGCNF Coarse	Offset	BGCNF Fine	Offset
000	Default	000	Default
010	+100mV	001	+5mV
001	+150mV	010	+10mV
011	-100mV	011	+15mV
100	-150mV	100	+20mV
		101	-5mV
		110	-10mV
		111	-15mV

Fig. 15.12. Band-gap trimming flow and example of configuration signals

15.3.4 Fuse ROM

In previous sections the necessity to trim the band-gap and other circuits has been described. After having determined the correct trimming code for the circuit, it is necessary to store the value in some non-volatile registers in order to restore the proper setting at the next power-on. This kind of trimming codes, together with other parameters, can be stored permanently into the fuse ROM (Sect. 13.5). Fuses are also used to store permanently the redundancy information collected during EWS. Because of the necessity to optimize the fusing process and test coverage, the Fuse ROM must be made testable; i.e. it should be possible to read out its content. An additional DFT feature, helpful in debugging the fusing process, is the possibility to access some of the fuse elements directly from the test pads (through DMA).

15.3.5 OTP Blocks

One Time Programmable (OTP) blocks are essential as a temporary (non-volatile) storage memory during the testing or to store important "fingerprint" information for the device. Differently from DRAM, where the storage memory is volatile, NAND Flash is non-volatile and could be used during the test flow to store information. NAND storing capability suffers error rate which is recovered by external ECC; therefore, the choice to use or not NAND storage instead of fuses must be carefully analyzed. A way to cope with those named limitations is to adopt some proper data correction, for example using majority codes. Anyhow,

Petitioner Micron Ex-1021, 0445

OTP blocks can be used efficiently in many cases when the information is temporary, i.e. during the test flow. An OTP block can be either added to the memory architecture or a standard block can be used as OTP by marking it as a bad block, depending on the chosen BB strategy.

15.3.6 Test interface

Test interface (TI) is a special interface to be used during testing in order to select test mode features, in addition to the standard *User Mode* (UM) commands (Sect. 15.3).

When a Low Pin Count interface is enabled, the TI manages this option and also manages the data in/out sequencing (Sect. 15.3.2). Since *Low Pin Count Testing* (LPCT) is used at wafer level to increase the parallelism, it is necessary to configure the TI to LPCT without using byte-wide commands. This is achieved by the introduction of a special test pin that, driven by the tester, communicates to the TI that LPCT has been enabled (Sect. 15.3.2).

15.3.7 Microcontroller as Built In Self Test (BIST) hardware

NAND operations (write, erase and read) are done through complex algorithms developed to obtain the correct functionality of the memory, coping with array issues. Algorithms are internally implemented by several macro sequences involving the circuits and the array. They are, as well, controlled by a finite state machine FSM or, more generally, by an internal microcontroller [3] (Sect. 6.5). An internal microcontroller (μC) is a solution which can enable flexibility to testing. Among the possibilities offered, Built In Self Test can be easily implemented without the need of extra hardware. A simple μC hardware for the implementation of BIST will cost less than 0.5 mm^2 including a ROM and a SRAM.

Some examples of having a microcontroller for testing are:

- By means of TI, it is possible to access the control signals in the same way the microcontroller does. This means that the test program can control internal circuitry with maximum flexibility. It is a useful feature in debugging.
- It is possible to load code from the tester into the SRAM in order to perform special tests or to patch internal algorithms.
- It is possible to use features such as breakpoints.
- It is possible to access memory and logic circuits (as BIST routines).
- It is possible to implement special BIST routines to enhance test time and parallelism (EWS, burn-in).

15.3.8 Fail counter

It is known that NAND Flash memories can be delivered with some failures in the array that will be managed by external Error Correction Codes (ECC) [6].

Petitioner Micron Ex-1021, 0446

Therefore, depending on technology reliability and application, manufacturers can tolerate some single bit errors that will be corrected by ECC. An analog fast fail counter can be present in the NAND device to implement Pseudo Pass Program scheme [11]. The same circuit can be used in error tolerant testing routines. A circuital example is given in Fig. 15.13: the circuit evaluates in one step if the fail number exceeds a certain value. The fails in the page buffers build up the current *Isum* equal to *N* times *Ifail*, where *N* is the fail counting. The current *Ifail* is compared with a reference value properly set by signals RF1 and RF2 according to the desired fail limit. If the fail current exceeds the reference value, then the fail signal is set. A fail flag can be made accessible through test registers.

A fail counter can be also implemented through a digital circuit (e.g. micro-controller) that counts the fails in a page and sets a pass/fail flag. This method is slower than the analog counterpart but it is more accurate especially when high fail count number is needed (e.g. in Vth distribution search tests).

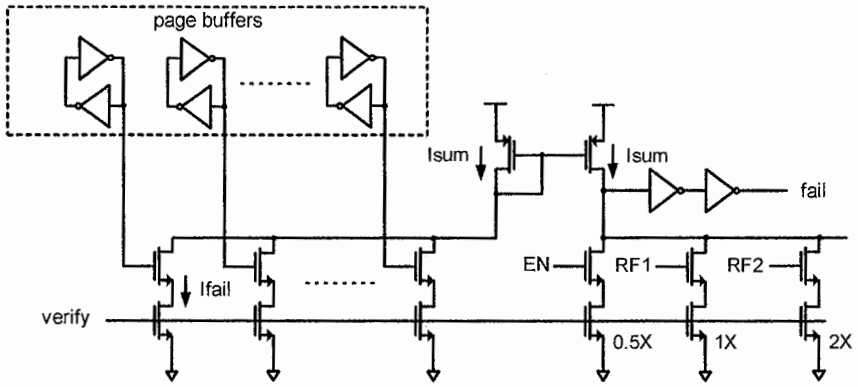


Fig. 15.13. Analog fail counter circuit

15.4 Fundamental NAND test modes

Array testing is the most time consuming and the most important part in NAND testing. As NAND memory devices increase their capacity, efficient array testing is more and more important because test time is proportional to the array size. DFT should focus particularly on test time reduction using the following guidelines:

- Write, erase operations and read/verify operations should be executed in parallel at string level, block level or multiple blocks level.
- When testing array, test data-in/data-out transfers should be avoided as much as possible.
- Test fail information should be provided to the tester in a compressed form in order to simplify redundancy analysis.

Petitioner Micron Ex-1021, 0447

In this section an overview of the most important test features, to be used particularly at wafer level, is given.

15.4.1 Parallel tests

Parallel wordline Program

By means of this test, it is possible to activate multiple wordlines in parallel in order to simultaneously program the entire block. This type of test can be used in many steps of the test flow, when it is necessary to obtain a preconditioning of the array with all the cells programmed; in this case, a good control of the programmed distribution is not required (i.e. program verify is not used). It can be used to setup a solid pattern (all wordlines programmed) or checkerboard patterns by applying programming voltage to even wordlines and V_{PASS} voltage to odd wordlines.

The capability of internal charge pumps must be checked (in terms of settling time) to sustain parallel programming, because they are designed to drive a single wordline during user mode operations. Therefore, the test should be properly characterized, especially at high temperature. When it is not possible to use the internal pumps, a possible option is to use an external voltage generator, using the test pads described in Sect 15.3.1. Limitations in this case derive from the maximum voltage/current capability during parallel tests.

The Parallel Wordline Program flow is summarized below:

1. Load program pattern into the page buffer
2. Select the wordlines to be programmed (even/odd, 0:31/32:63, 0:15/16:31/.....) according to the chosen data pattern
3. Connect the external voltage to the internal V_{PP} or the internal V_{PASS} according to the WL programming scheme
4. Apply the programming pulse
5. Exit

Parallel erase

Similarly to parallel program, parallel erase is a test feature that is useful not only for test time reduction at wafer level but also on packaged parts. For example, the first step in EWS is an erase operation used to precondition the array which comes from the factory in an unknown state. Moreover, it is effective in setup of cycling patterns.

The Parallel Erase flow is:

1. Select the blocks to be erased by means of the appropriate test registers
2. Connect the external voltage to the internal V_{ERASE}
3. Apply the erasing pulse
4. Exit

Petitioner Micron Ex-1021, 0448

Parallel read

This special test is used to speed up verification of patterns in the array. To implement the test, the read voltage must be applied to all the wordlines in a NAND string by means of special test mode setting. In this condition it is possible to understand if all the cells are below the verify level, similarly to the erase verify condition. In fact, if only one cell is above the verify level, it causes the NAND string to shut-off.

It is possible to discriminate even (odd) wordlines if odd (even) are biased to V_{PASS} instead of V_{READ}.

Differently from a standard read, in a parallel read all the cells in the string have the same voltage applied to the control gate, so that the read current depends strongly on the state of the cells (in standard read V_{PASS} is used to reduce this effect on the unselected cells). As a consequence, the test tells that all the cells are below the verify level, but not how far.

Parallel program of array and redundancy

With the purpose of reducing testing time, it is mandatory a test feature that allows programming standard and redundancy columns in parallel. Redundancy columns, in fact, must always have the same test coverage of the standard array and undergo the same test flow. This feature is useful at wafer level testing.

15.4.2 Margin read

The purpose of this test mode is to apply more severe conditions during read. This test is achieved by applying a higher (when reading 0s) or lower (when testing 1s) wordline voltage. The read voltage can be selected through test registers or it can be forced through test pins.

15.4.3 Data fail compression

Data compression in digital IC testing is related to compression of the I/O data stream in order to reduce the number of cycles needed to transfer a test sequence or a test result [7, 8]. For NAND Flash memories, the goal is to reduce the number of data transactions with the tester and this requirement is becoming more important as the memory capacity increases. The compression required in the NAND case does not aim at transfer all the data read from a page or a block in a compressed form. Instead, it aims to compress, for example, fails occurred in a bitline into a single bit. This information allows the tester to know that a bitline is failing and needs a replacement, without reading out all the pages.

To obtain a simple and effective fail compression, the page buffer structure must have at least two latches for each bit as in Fig. 15.14. This condition is usually satisfied in leading edge NAND devices [12]. Simpler SLC devices may use a single latch, but usually a second latch is available to perform pipelined

Petitioner Micron Ex-1021, 0449

read/write operations. The first latch is used to perform the standard verify/read operation; the second one is used as an error flag to accumulate the failures found scanning all the columns.

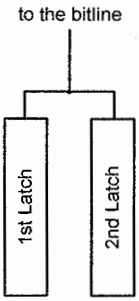


Fig. 15.14. Double latch page buffer schematic

In order to use the two latches for error compression it is necessary:

- To set or reset the latches independently
- To transfer data from the first to the second latch
- To read out the second latch output (or transfer the data into the first latch and read it out)

15.4.4 Internal Vth search

Vth search test mode is fundamental in the analysis of the cell population behaviour. The test basically analyzes the cell threshold distribution and, in particular, it finds out the edges of the distribution. This feature is very important for MLC devices analysis, where compact distributions are fundamental. It is often used in:

- Programmability tests to find out programming characteristics of the array
- Retention check tests
- Array behaviour analysis after stress or coupling tests
- Debug analysis

The Vth search is based on multiple read/verify operations where the V_{READ} voltage is swept through the search interval. To find out a distribution or a distribution tail, sometimes the test is run on a subset of the array, in order to obtain a sample of the device without scanning the entire array.

The algorithm for Vth tail search is summarized below (search of the lower edge of the distribution).

1. Set the V_{READ} voltage level inside the expected distribution (Fig. 15.15).
2. For each page within the specified address range, perform read operation versus expected data, count fails and update the fail counter. Exit the read routine if the fail counter exceeds the specified limit.

Petitioner Micron Ex-1021, 0450

3. If the fail counter is lower than the limit, exit the test and save V_{READ} as the Vth distribution edge.
4. Decrease the read voltage level and repeat the scan.

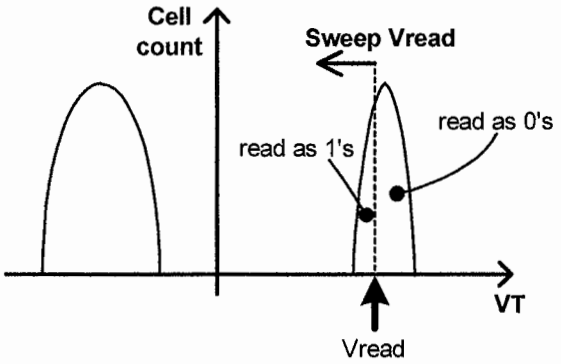


Fig. 15.15. Vth distribution edges search algorithm

The algorithms should be executed internally to save time. In particular, the fail counting should be performed internally with proper hardware (Sect. 15.3.8).

15.4.5 ROM, RAM, Fuse ROM testing

When the microcontroller structure is present, ROM and RAM must be tested. Memories Read/Write accesses are provided by the test interface.

Information fused inside Fuse ROM must be accessible for readout: this is mandatory when the fuse blowing step must be verified and optimized. A fused checksum value can be used to speed up the verification in mass production.

15.4.6 Internal clock measurement

Internal clock oscillator is one of the analog circuits whose functionality is fundamental for the correct NAND operations.

The internal oscillator can be tested in many indirect ways by measuring execution time of some basic operations, but it could be worth to add some hardware structure dedicated to perform this task with parallelism at wafer level. For example, an internal frequency measurement block can be designed by means of a digital counter fed by the internal clock, to count the time periods. The counter must be reset and driven by means of the test interface. After the testing time is elapsed, the internal counter is stopped and the value is read out. The period of the internal oscillator can be easily calculated by dividing the time elapsed with the counted cycles.

Petitioner Micron Ex-1021, 0451

15.4.7 Tests for defect detection and yield enhancement

One of the early phases of wafer level testing aims at monitoring the production and early scrapping of bad wafers.

Effective defect detection tests are of great importance also in the yield learning phase of the technology and repair strategies.

Figure 15.16 shows the localization of typical defect types in the backend process for a double metal level array with bitlines on the first level. Typically, the defect detection phase is composed of a set of tests aiming at finding out shorts and opens in the array with limited use of circuitry. Defect detection tests can be combined with static stress among layers to activate defects at wafer level.

The requirements list for defect detection tests starts by analyzing the array architecture and the process architecture and by identifying possible defect sources.

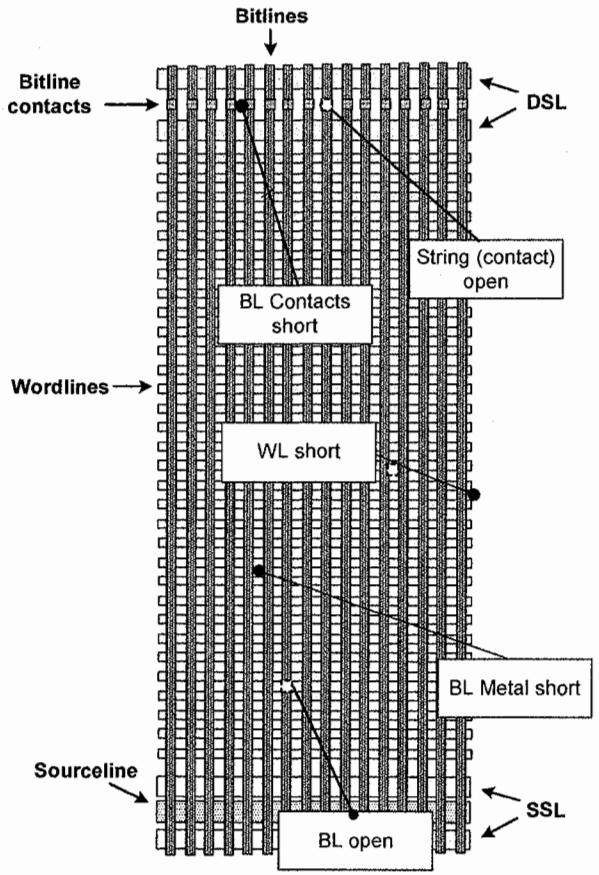


Fig. 15.16. Typical array back-end defect types

Petitioner Micron Ex-1021, 0452

In the following sub-sections a description of the test modes for defect detection in the back-end will be given. Test mode design leverage on the available NAND structures such as page buffers or ad hoc designed test hardware.

A generic test for defects detection in the back-end makes use of page buffers in an effective way to detect an improper discharge of the tested bitlines versus adjacent bitlines or a secondary layer such as wordlines. The test is generally composed by the steps listed below.

1. Precharge even (odd) bitlines and clamp odd (even) bitlines to GND.
2. Wait for an appropriate time.
3. Read page buffers and check the result. In case of shorts, the bitlines are discharged and read as 1; otherwise they are read as 0 having deselected the string by means of DSL.

Using the basic scheme described, it is possible to design the required tests.

The following sub-sections give implementation examples of some fundamental tests.

Shorts among bit lines detection

Shorts are caused by metal shorts and bitline contacts shorts.
Test flow

1. Precharge even bitlines, clamp to GND odd bitlines
2. Clamp to zero DSL
3. Wait proper time
4. Latch the value in the page buffer
5. Read out fails: 0s readout indicates a short with adjacent bitlines

Bit lines opens, string open

Opens are caused by bitline contact open, AA opens, Source line opens.
Test flow

1. Select a block preconditioned to 0s of 1s
2. Pre-charge bitlines
3. Apply pass voltage to all wordlines
4. Select DSL of a string or group of strings
5. Wait proper time
6. Latch the value in the page buffer
7. Read out fails: 0s readout indicates that the string is not able to discharge the bitline because of opens at the drain and/or source side

Shorts between wordlines

Shorts are caused by poly lines shorts.
Test flow

Petitioner Micron Ex-1021, 0453

1. Select one block preconditioned to all 0
2. Apply V_{PASS} to unselected WLs and V_{READ} = 0V to selected WL
3. Float the selected WL
4. Wait
5. Latch the page buffer
6. Read out fails: all 1s indicated that the selected wordline is shorted with adjacent wordline and charged up to the V_{PASS} level causing the bitline to discharge

The test can be applied to even/odd wordlines in order to detect shorts in one group of wordlines and proceed with bad block management without having identified exactly the failed wordline.

15.4.8 Stress modes

Defect activation tests (or stress tests) make use of voltage and/or temperature acceleration (Sect. 15.2.3) in order to activate defects that will occur later in the early-life of the product.

First, stress tests are performed at wafer level in order to early detect the biggest amount of defects and save testing, packaging and fixed costs. Stress tests at wafer level aim at screening:

- Tunnel oxide defects
- Other oxides defects
- Back-end of line defects
- Active area defects

Voltage acceleration is able to achieve high acceleration factors and it is suitable to be applied on oxides and back-end in the array.

Stress tests are time consuming and therefore, it is common to apply them separately from the defect detection tests for the reason of test cost optimization.

It follows a list of main tests for defect activation at wafer level usually performed at high temperature.

- ***Erase stress***
  - Perform some blind program/erase cycling (in the order of 100 cycles) using worst case conditions. Use a BIST self cycling routine
  - Use parallelism as much as possible
- ***V_{PASS} stress***
  - Apply V_{PASS} voltage plus over voltage to simulate the V_{PASS}/inhibit stress
  - Perform all-1 verify
- ***Pump stress***
  - Activation of charge pumps forcing their maximum operating output plus acceleration delta to simulate cycling.
  - Reduce/eliminate burn-in time

Petitioner Micron Ex-1021, 0454

- ***Wordline stress***
  - Static activation of odd wordlines versus even wordlines
  - Calculate proper acceleration delta to avoid overstress
- ***Bitline stress***
  - Static activation of all odd bitlines to VDD (or other internal voltage available) versus even wordlines to GND.
  - The stress should accelerate the stress during program inhibit.
- ***High VDD stress***

References

1. K. Sakui and K-D Suh NAND Flash Memory Technology in the book Non Volatile Memories with Emphasis on Flash, p. 223, Wiley, 2008.
2. G. Casagrande, Flash Memories, Kluwer, 1999.
3. A. Silvagni, G. Fusillo, R. Ravasio, M. Picca, S. Zanardi, An overview of logic architecture inside Flash memory devices, Proceedings of the IEEE, pp. 569–580, April 2003, Vol. 91.
4. P. Cappelletti, A. Modelli, Flash Memory Reliability, in the book Flash Memories, Kluwer, 1999.
5. ONFI organization, www.onfi.org.
6. R. Micheloni, A. Marelli, R. Ravasio, Error Correction Codes for Non-Volatile Memories, Springer, 2008.
7. M. Abramovici, M. A. Breuer, A. D. Friedman, Digital System testing and Testable Design, IEEE Press, 1990.
8. M.L. Buschnell, V.D. Agrawal, Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI, Kluwer, 2000.
9. P. Nicosia, F. Nava, Test Strategies on Non Volatile Memories, NVSMW, 2007. 22nd IEEE Volume, Issue, 26–30 Aug. 2007 Page(s):11–18.
10. NIST/SEMATECH e-Handbook of Statistical Methods, http://www.itl.nist.gov/div898/handbook/.
11. K. Hosono, T. Tanaka, K. Imamiya, K. Sakui, "A High speed Failure bit Counter for the Pseudo Pass Scheme in Program operation for Giga Bit NAND Flash", Non-Volatile semiconductor Memory Workshop, 2003.
12. K. Takeuchi, T. Tanaka, T. Tanzawa, A Multipage cell Architecture for High-Speed Programming Multilevel NAND Flash memories. IEEE Journal of Solid state circuits, Vol. 33, NO. 8, Aug.
13. K. Imamiya, Y. Sugiura, Nakamura, T. Himeno, K. Takeuchi, T. Ikehashi, K. Kanda, K. Hosono, R. Shirota, S. Aritome, K. Shimizu, K. Hatakeyama, and K. Sakui, A 130-mm2 256-Mbit NAND Flash with shallow trench isolation technology. IEEE Journal of Solid state circuits, Vol. 34, NO. 11, Nov. 1999.
14. K. Takeuchi and T. Tanaka A Dual-Page Programming Scheme for High-Speed Multigigabit-Scale NAND Flash Memories. IEEE Journal of Solid state circuits, Vol. 36, NO. 5, May 2001.

Petitioner Micron Ex-1021, 0455

15. K. Imamiya et al, A 125-mm2 1-Gb NAND Flash Memory With 10-MByte/s Program Speed. IEEE Journal of Solid state circuits, Vol. 37, NO. 11, Nov 2002.
16. T. Hara et al., A 146-mm2 8-Gb Multi-Level NAND Flash Memory With 70-nm CMOS Technology IEEE Journal of Solid state circuits, Vol. 41, NO. 1, Jan 2006.
17. K. Park, J. Choi, J. Sel, V. Kim, Y. Shin and K. Kim, Scalable WL Shielding Scheme using Dummy Gates in NAND Flash Memory for Eliminating Abnormal Disturb of Edge Memory Cell, 2006 International Conference on Solid State Devices and Materials, C-6-4L, 2006, pp. 298–299.
18. T. Tanzawa, T. Tanaka, K. Takeuchi, and H. Nakamura, Circuit Techniques for a 1.8-V-Only NAND Flash Memory. IEEE Journal of Solid state circuits, Vol. 37, NO. 1, Jan 2002.
19. Jedec standard JESD91A-JEP122 Method for Developing Acceleration Models for Electronic Component Failure Mechanisms, www.jedec.org.
20. Jedec standard JESD22a117b Electrically Erasable Programmable ROM (EEPROM) Program/Erase Endurance and Data Retention Stress Test, www.jedec.org
21. ITRS Test and Test Equipment 2009, http://www.itrs.net/.

Petitioner Micron Ex-1021, 0456

16 XLC storage

R. Micheloni[1] *and L. Crippa*[2]

16.1 Introduction

The obvious advantage of designing NAND devices capable of storing *n*-bit/cell (where *n* is currently 2, 3, and 4) is the resulting reduction in area occupation of the matrix. However, the benefits of 3-bit/cell (or 8-Level-Cell, 8LC) and 4-bit/cell (or 16-Level-Cell, 16LC) technologies don't come for free.

Programmed levels are limited to the same *program threshold window* V_{THPW} (see Fig. 16.1) as any MLC device (referred to as 4LC throughout this chapter).

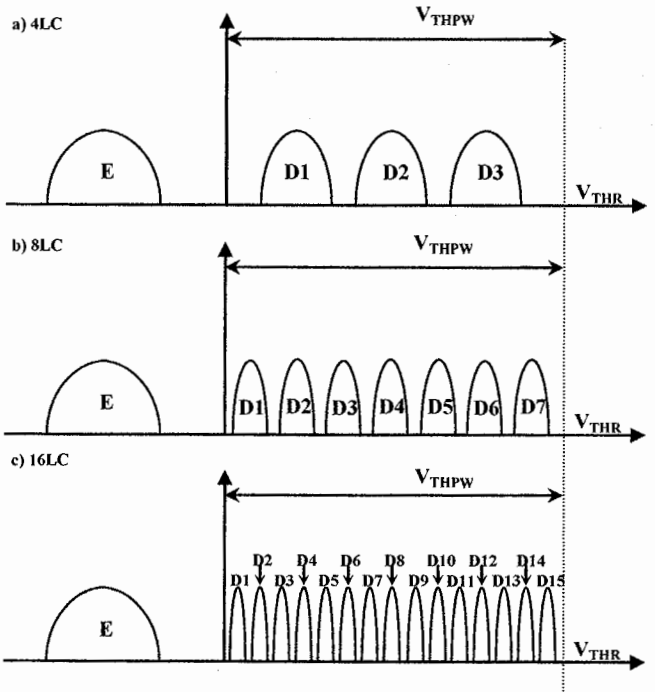


Fig. 16.1. XLC program window V_{THPW} and distribution width

[1] Integrated Device Technology, rino.micheloni@ieee.org
[2] Forward Insights, luca.crippa@ieee.org

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_16,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0457

In fact, as is the case for a 4LC device, the highest verification level must be low enough to prevent bit failures such as program disturb and read disturb. The more states a memory cell must be able to store, the more finely divided its program threshold window is. Therefore, it is mandatory to reduce further the width of programmed distributions (Sect. 16.2).

A reduction in area occupation of the matrix has a drawback in an increment of the area occupied by the peripheral circuits: in particular, the area of the sensing circuits. In fact, the number of latches inside a sensing circuit is (as a minimum) equal to the number n of bits stored inside the cell (Sects. 16.3 and 16.4).

A number n of bits corresponds to a number $(2^n - 1)$ of levels of read and verify (see Fig. 16.2). Therefore, a device with n bits per cell needs $(2^n - 1)$ SROs to read the content of the cell. Moreover, the number of verify operations after a single program pulse is equal to $(2^n - 1)$.

Taking into account the last item, we could come to the conclusion that a read operation of a 16LC device is 15 times slower than a SLC device. Indeed, 15 SROs can read four pages, and therefore the *average page access time* T_R

$$T_R = \frac{(2^n - 1)}{n} \cdot T_{SRO} \tag{16.1}$$

is equal to 3.75 T_{SRO}, where T_{SRO} is the time spent by one SRO.

Therefore, T_R ratio between a 16LC and a SLC device is 3.75 and not 15. This ratio is as low as 2.5 if a 4LC device is considered ($T_R = 1.5 \cdot T_{SRO}$).

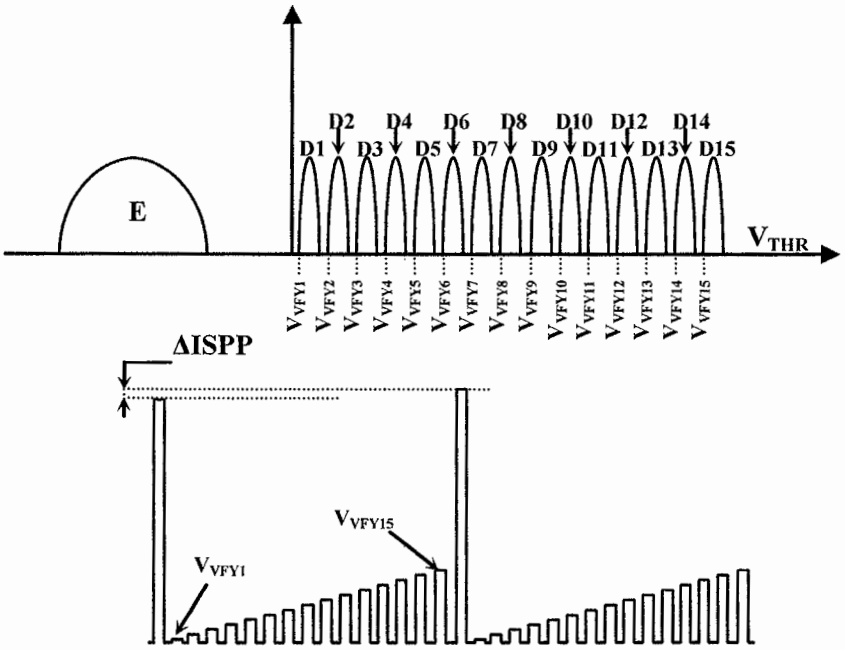


Fig. 16.2. 16LC ISPP and its 15 verify levels

Petitioner Micron Ex-1021, 0458

It should also be taken into account that NAND devices are mainly used for mass-storage and that a considerable number of pages is involved in a read operation. If the device features cache read (see Chap. 10) and data-output time of a page T_{DO} is longer than T_R, then there is no real difference in read performance between a SLC, 4LC or 16LC device (see Fig. 16.3). Similar considerations apply to interleaved architectures (e.g. systems where a controller drives several NANDs in parallel, as described in Chap. 2).

In any case a XLC device is affected by the issue of the exponentially growing number of verify operations to be performed after each program pulse. The consequent dramatic drop in program speed is the most relevant weak point of XLC devices. Both the reduction of overall verify time and other interesting solutions to increase program speed are discussed in following sections.

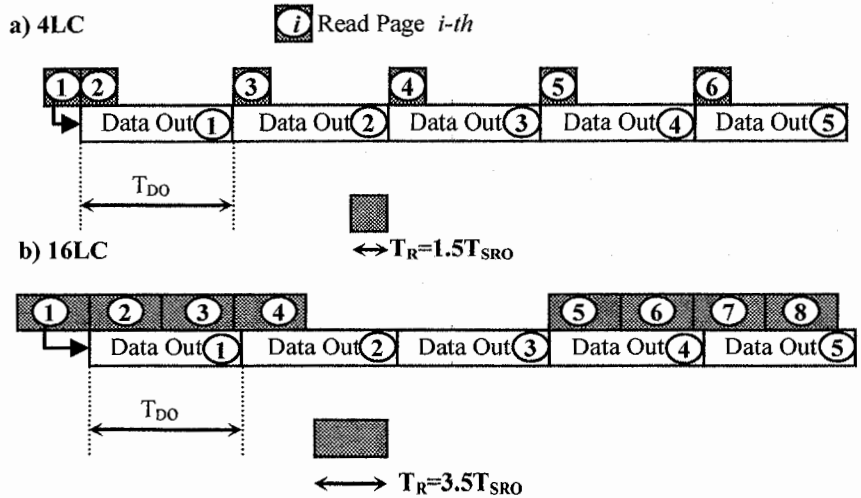


Fig. 16.3. 4LC versus 16LC with cache read: data-out streaming

16.2 V_{TH} distribution width

Figure 16.4 shows the main parasitic effects responsible for the widening of the cell's threshold voltage (V_{TH} and V_{THR}, Definition 8.2) distributions, as discussed in Chaps. 8–10. Table 16.1 summarizes the countermeasures described in the above mentioned chapters.

Most of those effects can be freely minimized: for instance, effect (1) by reducing ΔISPP, and effect (4) by using an appropriate number of multi-passes. The obvious drawback is a drastic increase in both program and read times.

Effect (3) can be reduced as well by the constant Bit-Line (BL) bias. Effects (5) and (6) are already minimized enough when the solutions above mentioned are put in place.

Petitioner Micron Ex-1021, 0459

Table 16.1. Countermeasures adopted to minimize distribution widening

Parasitic effect	Solution	Reference
(1) ΔISPP	Reduce ΔISPP	
(2) FG–FG coupling	ABL program	Sect. 8.4 E
	Erase distribution compacting	Sect. 9.2
	Program sequence	Sect. 10.1.2
(3) BL–BL coupling	ABL sensing	Sect. 8.3
(4) SL bias error	SL tracking	Sect. 9.3.1
	Multi-pass sensing	Sect. 9.3.2
(5) BPD	Low-current bias	Sects. 8.4 D–9.1
	Erase distribution compacting	Sects. 9.1 and 9.2
(6) PGM & Read Disturb	ABL	Sect. 8.4 F

The big hurdle to the narrowing of the distributions in XLC devices is the Floating Gate (FG) coupling with adjacent cells, which becomes more and more relevant as the shrink of design rules goes on. Figure 16.5 shows the shift of the programmed distribution D_X of cell $M_{i,n}$ due to the FG coupling with neighbor cells.

It should be noted that cells M_{i-1} belonging to Word-Line (WL) $i-1$ in Fig. 16.5 are not taken into account, since they have already been previously programmed.

The following equation can be used to approximately estimate the shift ΔV_{Mi} of the distribution D_X due to floating gate coupling:

$$\Delta V_{Mi} = \frac{C_{FGNC}}{C_{FGTOT}} \cdot \Delta V_{NC} \tag{16.2}$$

where:

$$C_{FGNC} = 2 \cdot C_X + C_Y + 2 \cdot C_{XY} \tag{16.3}$$

ΔV_{NC} is the average shift of threshold voltages of neighbor cells, programmed in state D_{NC}. C_X is the contribution along bitline direction, C_Y along wordline direction and C_{XY} is the contribution along diagonal direction. C_{FGTOT} is the overall floating gate capacitance which includes all contributions.

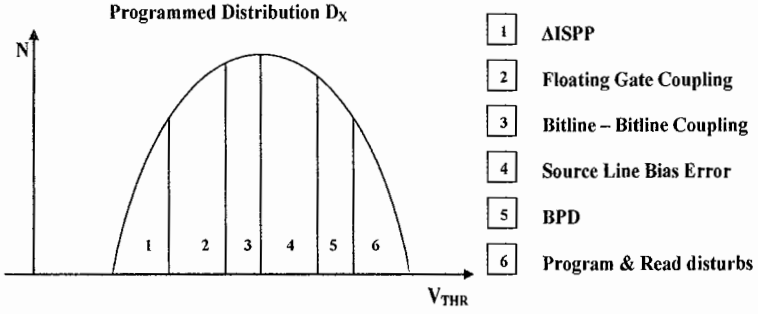


Fig. 16.4. Main parasitic contributes to the programmed distribution width

Petitioner Micron Ex-1021, 0460

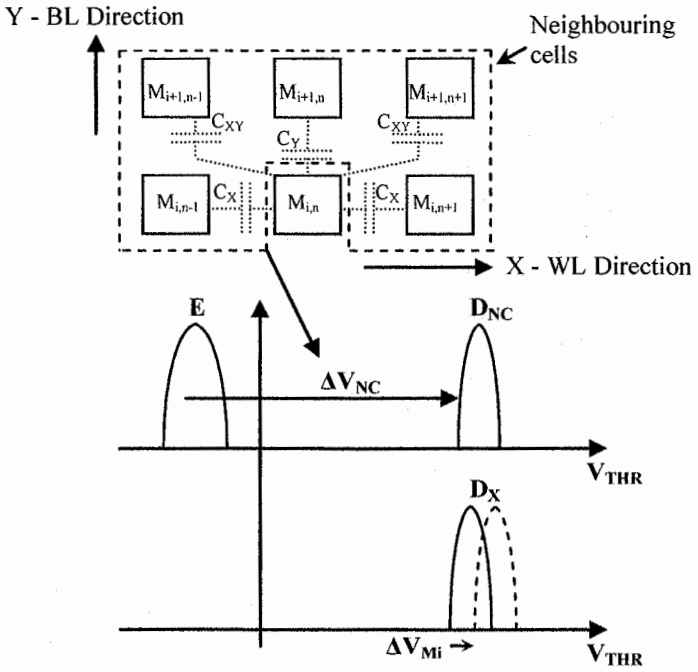


Fig. 16.5. Coupling Capacitance contribute and FG coupling

Because of continuous technology shrink, the ratio between C_{FGNC} and C_{FGTOT} grows bigger and bigger. Such ratio is solely due to technology, and it cannot be mitigated by design whatsoever. Aim of the design is instead to try and reduce ΔV_{NC} as much as possible, as already discussed in Sect. 10.1.2 for the case of 4LC technologies.

Next two sections describe the solutions implemented to reduce FG coupling in 8LC and 16LC devices, together with the techniques used to increase program speed.

16.3 8LC

16.3.1 Program sequence

In order to reduce FG coupling effect, the same solution devised for a 4LC device (Sect. 10.1.2) can be applied to an 8LC one. In particular, the page program sequence described in Fig.10.6 can be adapted to an 8LC device as shown in Fig. 16.6 [1]. To this end, a middle-page is introduced, interposed between the upper-page and the lower-page. Program sequence for the first 15 pages is shown in Fig. 16.6a.

Petitioner Micron Ex-1021, 0461

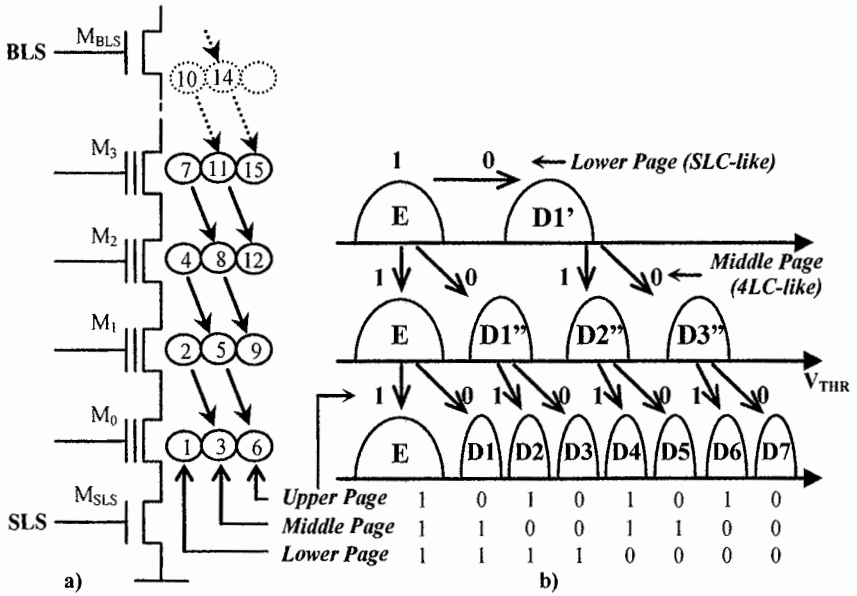


Fig. 16.6. Page program sequence for an 8LC device

Programming of the seven levels D_X (X = 1,2,...,7) is carried out in three separate rounds, as shown in Fig. 16.6b. Lower-page is programmed during the first round, middle-page during the second one and upper-page during the third one.

Distributions D1', D1", D2", and D3" are *Temporary Distributions* (TDs), which are then re-programmed until they reach their final state D_X [5, 6]. The biggest advantages of this approach are the following:

1. Program-speed increase
2. FG coupling reduction

The first advantage should be evident recalling the discussion developed in Sect. 10.1.2 (Fig. 10.5): D1', D1", D2", and D3" (TDs) do not need the eight levels precision; therefore, the lower-page can be quickly programmed using the voltage step ΔISPP1 of a SLC device (similarly to distribution D2' of Fig. 10.5). Middle-page is programmed using the same precision of a 4LC device, whose ΔISPP2 is smaller than the step ΔISPP3 required by the program operation of the upper-page.

As far as the second advantage is concerned, TDs greatly reduce floating gate coupling. An example without TDs can help the explanation. Let's assume we want to store "011" (distribution D4, Fig. 16.6) in cell M_0 and "000" (distribution D7) in cell M_1. Without re-programming, no intermediate distribution exists and, therefore, during the first round of the lower-page programming, cell M_0 is directly shift to distribution D4 using the precision of an 8LC device. During its following two program rounds, cell M_0 remains in D4, according to the decoding shown in Fig. 16.6.

Petitioner Micron Ex-1021, 0462

On the other hand, during its three rounds, cell M_1 moves from distribution E to D7 following the steps E→D4→D6→D7. Final result is shown in Fig. 16.7: the whole V_{THR} leap ΔV_{M1} from E to D7 is coupled with M_0, determining maximum shift ΔV_{M0} (see Eq. (16.2)):

$$\Delta V_{M0} = \frac{C_{FGNC}}{C_{FGTOT}} \cdot \Delta V_{M1} \tag{16.4}$$

Let's see what happens to FG coupling in the same case, but using TDs and program sequence of Fig. 16.6a. Figure 16.8 shows the reprogramming technique applied to cell M_0: The algorithm is described here below.

- First round (programming of page 0): cell M_0 reaches distribution D1' above temporary verify level V'_{VFY1}.
- Second round (programming of page 3): cell M_0 reaches distribution D2" above temporary verify level V''_{VFY2}. Cells within dotted area of distribution D1' are reprogrammed to D2".
- Third round (programming of page 6): cell M_0 reaches distribution D4 above final verify level V_{VFY4}. Cells within dotted area of distribution D2" are reprogrammed to D4.

Indeed, the first programming round of cell M_0 is followed by the first programming round of cell M_1 (E→D1'), i.e. of page 2, as shown in the sequence of Fig. 16.6a. Because of FG coupling, distribution D1' shifts (from dotted to solid distribution) as shown in Fig. 16.9. Thanks to re-programming technique, this shift can be completely eliminated by the second program round of cell M_0. In the same way, the second program round of cell M_1 (D1'→D3"), that is of page 5, is recovered by the third program round of cell M_0. The third round of program of cell M_1 (D3"→D7), that is of page 9, is the only one which cannot be recovered by M_0.

The amount of shift ΔV_{M0} is in any case much smaller than the one for the case where no re-programming is used.

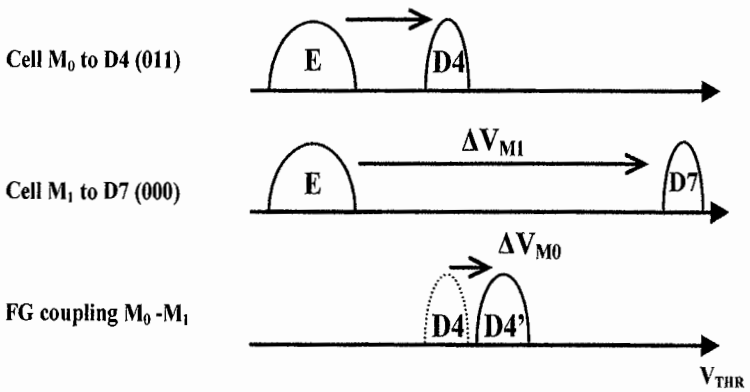


Fig. 16.7. Worst-case FG coupling without temporary distributions

Petitioner Micron Ex-1021, 0463

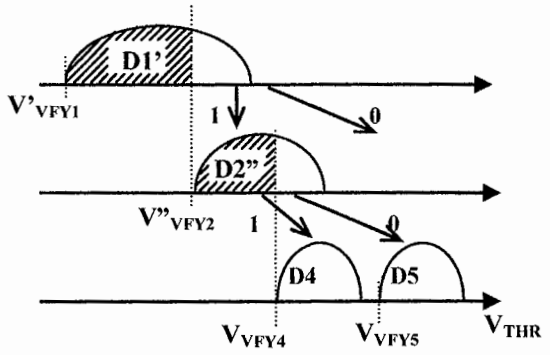


Fig. 16.8. Program rounds on cell M_0 (distribution overview)

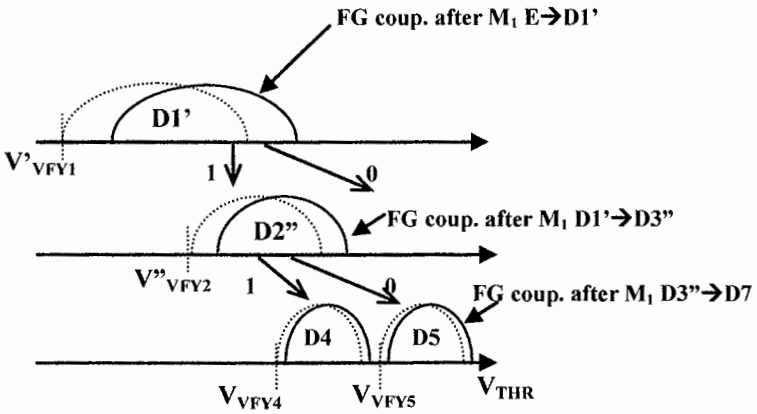


Fig. 16.9. M_0 FG coupling recovery during re-programming

Worst case occurs when M_1 shifts from level $V"_{VFY3}$ to the rightmost level of distribution D7 (V_{D7MAX}), as shown in Fig. 16.10. Without re-programming, ΔV_{M1} is in the order of 6–7 V, while in this case it is approximately 1 V. Overall mitigation of the FG coupling is therefore in the order of 70–80%.

Obviously a complete recovery of the intermediate FG coupling requires that all the verify levels are correctly set. The rule of the thumb is that the right boundary of the reprogrammed distribution should be greater than the right boundary of the initial distribution (see Sect. 16.3).

Indeed, cell M_1 could statistically occupy any of the eight possible levels, and therefore all the possible combinations should be considered: beginning with the case where M_1 remains in E (no shift of M_0), all the way through the worst case seen so far (M_1 in D7). Therefore, it is not a rigid shift of the distributions, but rather a real widening. Figure 16.11 shows the widening scenarios for all the distributions in any possible situation.

Petitioner Micron Ex-1021, 0464

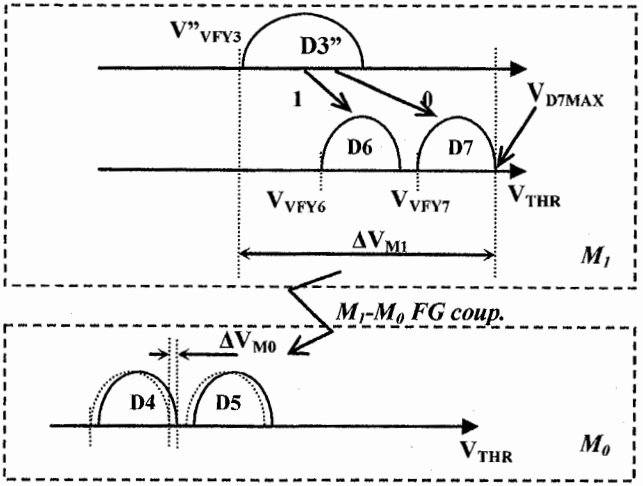


Fig. 16.10. Worst-case FG coupling with distribution re-programming

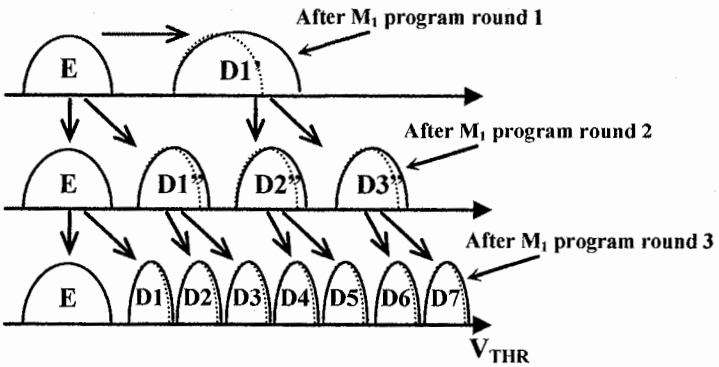


Fig. 16.11. Overall view of FG coupling with distribution re-programming

Re-programming technique discussed so far applies to ABL architecture [5] but it can be easily extended to an interleaving architecture [6]. This technique is the most efficient in terms of FG coupling recovery, but indeed other methods have been devised over time: for instance, special sequences where both lower-page and middle-page are programmed in a single 4LC round (equivalent to a Full-Sequence MLC, Sect.10.1.1) and the upper-page in a reprogramming round of 8LC, with either ABL or interleaving architecture [4].

Full-Sequence 8LC reprogramming techniques (therefore without re-programming) have been even used in the first 8LC devices at 56 nm with ABL architecture [2, 3]. More information can be found in the referenced documents at the end of this chapter.

Petitioner Micron Ex-1021, 0465

16.3.2 Program: circuits and cache operation

As discussed in Chap. 10, two latches are needed for 4LC programming. Such two latches not only contain information about target distribution, but they are also involved in both verify and program/inhibit operations. Therefore, transfers and logic operations are implemented in order to correctly process the bits inside the latches.

Of course, the minimum number of latches to program an 8LC device is three. The combination of bits stored in each latch identifies target distribution. It is not so easy and immediate to manipulate the content of the three latches in order to ensure a proper 8LC programming. In general, another latch is added in both 8LC and 16LC devices [3], which is used for read-verify operations: in this way, the sensing circuit is more flexible and easier to manage. In ABL architecture, the additional latch corresponds to block SA of Fig. 8.26.

Figure 16.12 shows the architecture of the sensing circuit of an 8LC device. Latch LRV is used for read-verify operations. The three data latches L1, L2, and L3 hold the 3 bits of data for 8LC operations. Latch L1 is used for Data-Load and Data-Output operations, managed by the Logic Unit. Information exchange among the four latches occurs across node SO.

Figure 16.13 shows a possible implementation of the sensing circuit.

Each target distribution D_X (X = 1,2,...,7), is coded inside latches L1, L2, and L3, as shown in Table 16.2. A1, A2, and A3 represent the logic output of latches L1, L2, and L3, respectively. Node ARV is the output of latch LRV and it is connected to the bitline through transistor M_{PROG} during program pulses. ARV is equal to "1" in case of cell inhibit, and therefore it is initially set to "1" only if the cell must remain in distribution E.

Table 16.12 is valid both in case of Full-Sequence program and in case of program of the upper-page using re-programming (Fig. 16.6). In the latter case, latches L2 and L1 are loaded after reading both the lower-page and the middle-page stored inside the cell.

As usual, the programming algorithm is ISPP based and it is made up by program pulsed followed by verify operations.

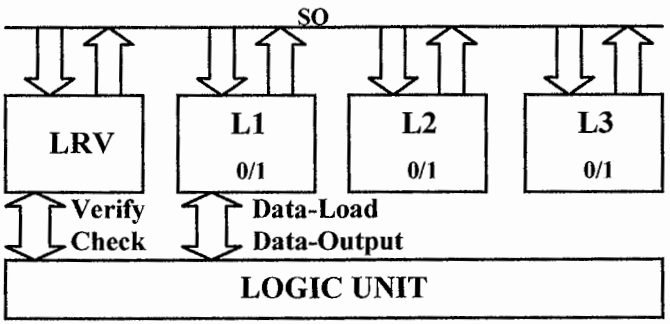


Fig. 16.12. 8LC sensing architecture

Petitioner Micron Ex-1021, 0466

Table 16.2. Logic content of latches L1, L2, L3 and LRV of Fig. 16.13

	E	D1	D2	D3	D4	D5	D6	D7
ARV	1	0	0	0	0	0	0	0
A3	1	0	1	0	1	0	1	0
A2	1	1	0	0	1	1	0	0
A1	1	1	1	1	0	0	0	0

Figure 16.14 shows the verify sequence following an ISPP program pulse. As for every SRO (see Chap. 8), after a verify operation performed at level V_{VFYi}, node SO in Fig. 16.13 can get two possible voltage values:

- if $V_{THR} > V_{VFYi}$ (V_{VFYi} successful) then SO = "1" (V_{DD}).
- if $V_{THR} < V_{VFYi}$ (V_{VFYi} fail) then SO = "0".

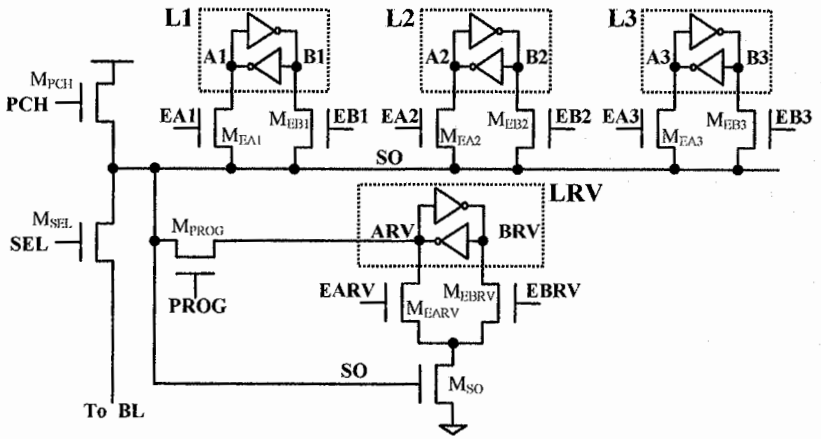


Fig. 16.13. 8LC sensing circuit for program/verify operations

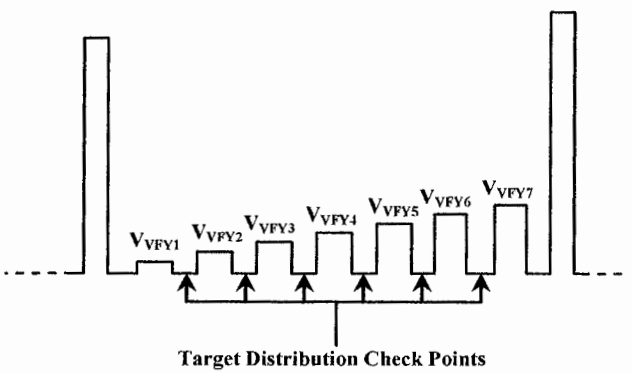


Fig. 16.14. 8LC verify sequence following a ISPP program step

Petitioner Micron Ex-1021, 0467

Before storing the value of node SO inside latch LRV, it must be checked (Check Points of Fig. 16.14) whether the target level of the cell is either V_{VFYi} or higher levels. For instance, a cell whose target is D7 goes above levels V_{VFY1}, V_{VFY2} etc, but only the overcoming of V_{VFY7} should be considered. If the target is not V_{VFYi}, then the corresponding verify success should not be considered (case *false-success*). If the target is V_{VFYi}, then the corresponding verify success should be considered (case *true-success*). At this point, latches L1, L2, and L3 come in.

From a functional point of view, in case of false-success it is sufficient to force SO voltage to ground. On the other hand, in case of true-success (or in case of failed verify as well) node SO should not be conditioned whatsoever.

A first solution which can be applied to the circuit of Fig. 16.13 is to enable one of the two transistors M_{EAj} or M_{EBj} (j = 1,2,3), according to the sequence shown in Table 16.3 and according to the timing shown in Fig. 16.15. Transistors are enabled biasing their gates to a voltage V_L.

The value of V_L is slightly greater than the threshold voltage V_{THN} of a NMOS. For example, let's assume that V_L is 200 mV higher than V_{THN}.

After a verify operation at V_{VFYi}, transistors of the corresponding column in Table 16.3 are enabled. If the cell has overcome V_{VFYi} but under a condition of false-success (SO is "1"), node SO is discharged to ground by at least one of the three latches (case 1, 2 or 3 in Fig. 16.15). If the cell has overcome V_{VFYi} in a condition of true-success, no latch discharges node SO, which remains at V_{DD}. If node SO is "0" (verify fail), it is charged at 200 mV by at least one of the three latches. As far as the trigger threshold of latch LRV is concerned, 200 mV is equivalent to "0".

For example, let's assume that the cell should reach V_{VFY6}: combination "111" is present on nodes A3, B2, and B1 (see Table 16.2). The only combination which connects A3, B2, and B1 to SO is indeed the one of column V_{VFY6} in Table 16.3. In all the other cases, V_{VFYi} ($i \neq 6$), at least one node at "0" is connected to SO, and the corresponding verify is ignored, no matter if it is a fail or a false-success. Finally, in case a fail occurs at V_{VFY6}, node SO is charged from zero to 200 mV.

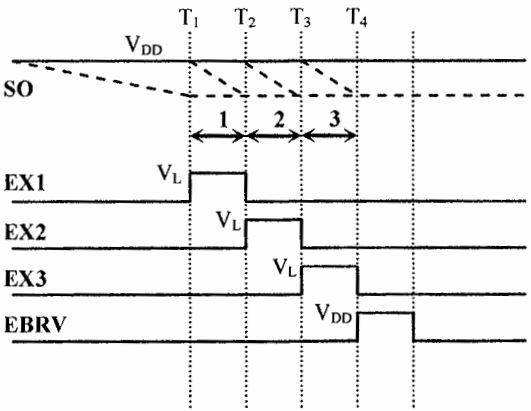


Fig. 16.15. Timing diagram related to Table 16.3

Petitioner Micron Ex-1021, 0468

Table 16.3. Transistors biased (on the gate) to V_L for each verify operation

	V_{VFY1}	V_{VFY2}	V_{VFY3}	V_{VFY4}	V_{VFY5}	V_{VFY6}	V_{VFY7}
M_{EX3}	M_{EB3}	M_{EA3}	M_{EB3}	M_{EA3}	M_{EB3}	M_{EA3}	
M_{EX2}	M_{EA2}	M_{EB2}	M_{EB2}	M_{EA2}	M_{EA2}	M_{EB2}	
M_{EX1}	M_{EA1}	M_{EA1}	M_{EA1}	M_{EB1}	M_{EB1}	M_{EB1}	

The last operation is the storing of node SO logic value inside latch ERV, by enabling M_{EBRV} (Figs. 16.13 and 16.15). If SO equals "1", node ARV goes to "1" and the cell is inhibited (permanent inhibit), otherwise the state of the latch LRV does not change.

The use of the analog voltage V_L (instead of V_{DD}) prevents the case where one of the three latches charges node SO to $V_{DD} - V_{THN}$, which would be interpreted as a "1" by the latch ERV. A modified circuit which only uses digital signals is shown in Fig. 16.16. Node SO can only be discharged to ground, by enabling transistors M_{EB} or M_{EA}, using a sequence similar to the one shown in Table 16.3. In this implementation, the signals which enable either M_{EB} or M_{EA} are digital, and they are equal to V_{DD}. On the contrary, it is not possible to force node SO to V_{DD}. Transistor M_T is used to transfer the data inside the latch.

Once D1, D2, and D3 have been done, there are only other four distributions to be programmed. Latches L2 and L3 are used for this purpose. Therefore latch L1 remains unused, and it can be used to load the following page (cache program). For the same reason, once levels D4 and D5 have been programmed, latch L2 can be used to load another page, and latch L3 alone is used to program distributions D6 and D7. After distribution D6 has been programmed, latch L3 is free as well, and distribution D7 is programmed using latch LRV only [3].

Figure 16.17 shows the cache program operation: the first 8LC program starts after data-load of pages 1, 2, and 3. Once the cells to be programmed have gone above V_{VFY3}, latch L1 can be used to load page 4. Similarly, once V_{VFY5} is overcome, latch L2 can be used to load page 5. Finally, page 6 is loaded in L3 once V_{VFY6} is overcome.

The circuit which controls the successful overcome of the different verify levels is shown in Fig. 16.18 and it is composed of two PMOS in series, M_{PSO} and M_{PARV}.

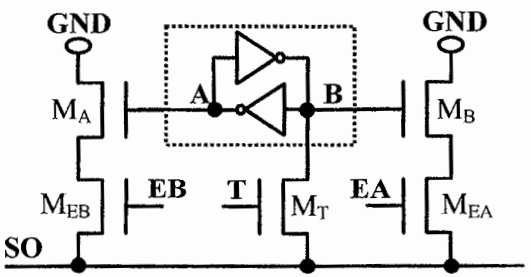


Fig. 16.16. Modified latch scheme using only digital signals

Petitioner Micron Ex-1021, 0469

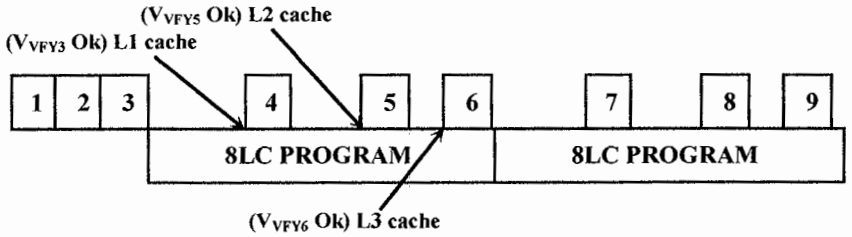


Fig. 16.17. 8LC cache program operation

The concept is similar to the one discussed in Sect. 10.2.4. Node VOK_N is common to all sensing circuits and it is weakly pulled down by M_{CHK}.

Let's neglect for a while transistor M_{PARV}: if all the cells go above level V_{VFYi}, all SO nodes are at V_{DD} (before time T_1 in Fig. 16.17). Therefore, all PMOS M_{PSO} are turned off. By enabling M_{CHK}, node VOK_N is forced to ground. Otherwise, if at least one cell has not overcome level V_{VFYi}, the corresponding M_{PSO} is on and node VOK_N goes to V_{DD}.

- If VOK_N = "0" then all the cells have gone above level V_{VFYi}.
- If VOK_N = "1" then at least one cell has gone above level V_{VFYi}.

MOS transistor M_{PARV} is necessary for those cells which have already been programmed and therefore would discharge node SO during verify phases. In fact, during a verify operation at V_{VFYi}, the cells programmed up to the D_{i-1} would discharge node SO anyway. The node ARV of the corresponding latch LRV of these cells, which are in permanent inhibit, is set to "1". By activating M_{PARV} as shown in Fig. 16.18, gate-controlled by node ARV, the cells which are already inhibited do not influence VOK_N (M_{PARV} off). From a logic point of view, VOK_N is the wired-nor of nodes ARV and SO of all the sense amplifiers.

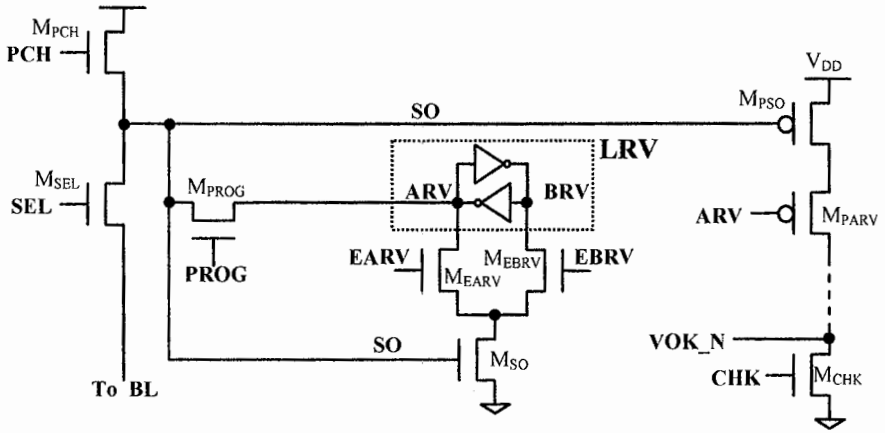


Fig. 16.18. 8LC Verify check circuit

Petitioner Micron Ex-1021, 0470

This check is useful not only for the cache program, but also to exclude useless V_{VFYi} shown in Fig. 16.14. For each level of V_{VFYi} the check of VOK_N is performed. If VOK_N = "0", then the corresponding verify is excluded from the following ISPP steps. In this way, the overall verify duration is greatly reduced, thus speeding-up program operations.

In a complementary fashion, at the beginning of the ISPP staircase the highest V_{VFYi} are useless, since the cells have a V_{THR} which is around distribution E. Specifically, it is useless to perform a verify at V_{VFYi} when no cell has overcome the previous level, V_{VFYi-1}. A smart strategy is to add the verification of level V_{VFYi} when at least one cell has overcome level V_{VFYi-1} [4]. This check is carried out adding the MOS M_{NSO} shown in Fig. 16.19.

The circuit behaves complementarily to the circuit shown in Fig. 16.18. Node FOK is common to all the sensing circuits. M_{PCHK} is a weak pull-up PMOS. During the verify at V_{VFYi-1}, if at least one node SO is at V_{DD}, the corresponding MOS M_{NSO} is on and node FOK is forced to ground (M_{NSO} is more conductive than M_{PCHK}). V_{VFYi} will be added to the next verify sequence. Vice versa, if no cell has overcome level V_{VFYi-1}, all nodes SO are at ground and all M_{NSO} are off: node FOK is forced to V_{DD} by M_{PCHK}. V_{VFYi} should not be added to the next verify sequence.

Conclusion: combining the circuits shown in Figs. 16.18 and 16.19, the verify levels are added only when really needed and are eliminated when they become useless: an ISPP with an optimized number of verify operations is achieved, and therefore program speed is maximized. Moreover, cache program operations are implemented.

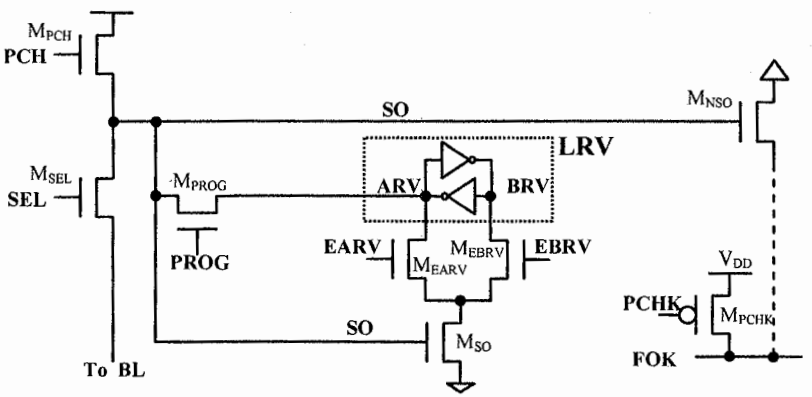


Fig. 16.19. 8LC Circuit for a minimized number of verify operations

16.3.3 Read: circuits and cache operation

Figure 16.20 shows a circuit which can be used for read operation. Of course, seven SROs are needed in order to read the content of a cell; such SROs are carried out using the seven read levels of V_{READi} shown in Fig. 16.21.

Petitioner Micron Ex-1021, 0471

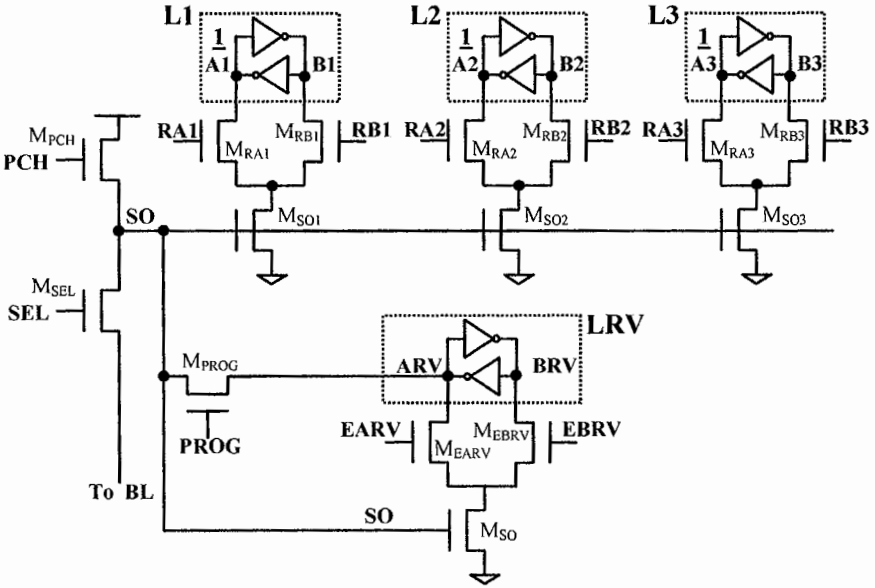


Fig. 16.20. 8LC read circuit

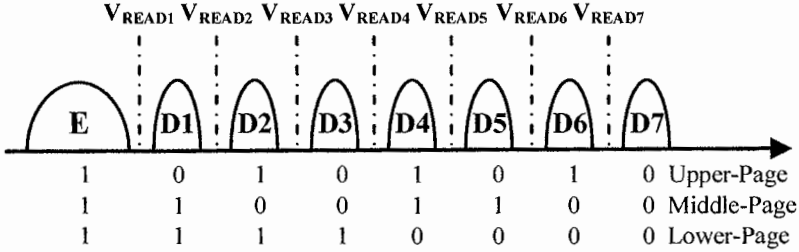


Fig. 16.21. 8LC read levels

Let's assume that the eight levels are coded as shown in Fig. 16.21. The three latches start with the corresponding nodes A1, A2, and A3 set to "1". At the end of each SRO, transistors are enabled according to Table 16.4. As usual, node SO is equal to "0" if $V_{THR} < V_{READi}$ and to "1" if $V_{THR} > V_{READi}$.

Table 16.4. Enabled transistors after the SRO at V_{READi}

	V_{READ1}	V_{READ2}	V_{READ3}	V_{READ4}	V_{READ5}	V_{READ6}	V_{READ7}
M_{RX3}	M_{RA3}	M_{RB3}	M_{RA3}	M_{RB3}	M_{RA3}	M_{RB3}	M_{RA3}
M_{RX2}		M_{RA2}		M_{RB2}		M_{RA2}	
M_{RX1}				M_{RA1}			

Petitioner Micron Ex-1021, 0472

At the end of the seven SRO operations, nodes A1, A2, and A3 are equal to lower-, middle- and upper-page, respectively. It is worth noting that a single SRO at V_{READ4} is sufficient to read the lower-page.

- If $V_{THR} < V_{READ4}$ then SO = "0": enabling M_{RA1} the latch keeps its state (A1 = "1").
- If $V_{THR} > V_{READ4}$ then SO = "1": enabling M_{RA1} the latch changes its state (A1 = "0").

Read operation is not necessarily sequential: the read of the lower-page could be done immediately in a SRO, followed by its data output. In this way, latency timings in an 8LC device can be minimized, and they become equal to those of a SLC device (see Fig. 16.3).

For the middle-page, three SRO are enough, at V_{READ2}, V_{READ4}, and V_{READ6}.

- If $V_{THR} < V_{READ2}$ then A2 = "1".
- If $V_{READ2} < V_{THR} < V_{READ4}$ then A2 = "0".
- If $V_{READ4} < V_{THR} < V_{READ6}$ then A2 = "1".
- If $V_{THR} > V_{READ6}$ then A2 = "0".

For the upper-page all the SROs are needed. Combining read circuit shown in Fig. 16.20 with program circuit shown in Fig. 16.13, the complete scheme of a latch can be derived, as shown in Fig. 16.22. Of course, this scheme can be used for L1, L2, and L3.

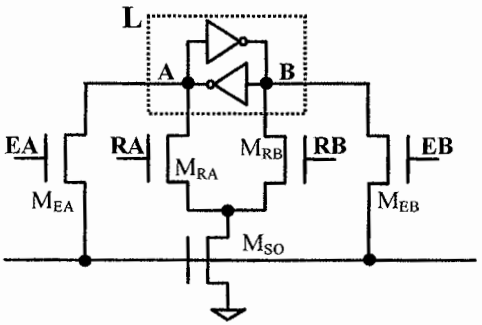


Fig. 16.22. 8LC complete latch schematic

16.4 16LC

In order to produce a 16LC device, distributions must be narrowed even further. Taking Fig. 16.1 as a reference, a programming technique which greatly reduces FG coupling is shown in Fig. 16.23 [7]. Let's consider again two adjacent cells, M_0 and M_1. Let's assume that target distribution for M_0 is a generic distribution D_i at level V_{VFYi} (i = 1,2,...,15) of Fig. 16.1, while the target for cell M_1 is D15 (worst-case FG coupling for M_0).

Petitioner Micron Ex-1021, 0473

Cell M_0 is roughly programmed at a level V_{VFYTj}, which is lower than the target level V_{VFYi}, Fig. 16.23a. This initial program round is fast, since it is performed using a wide ΔISPP.

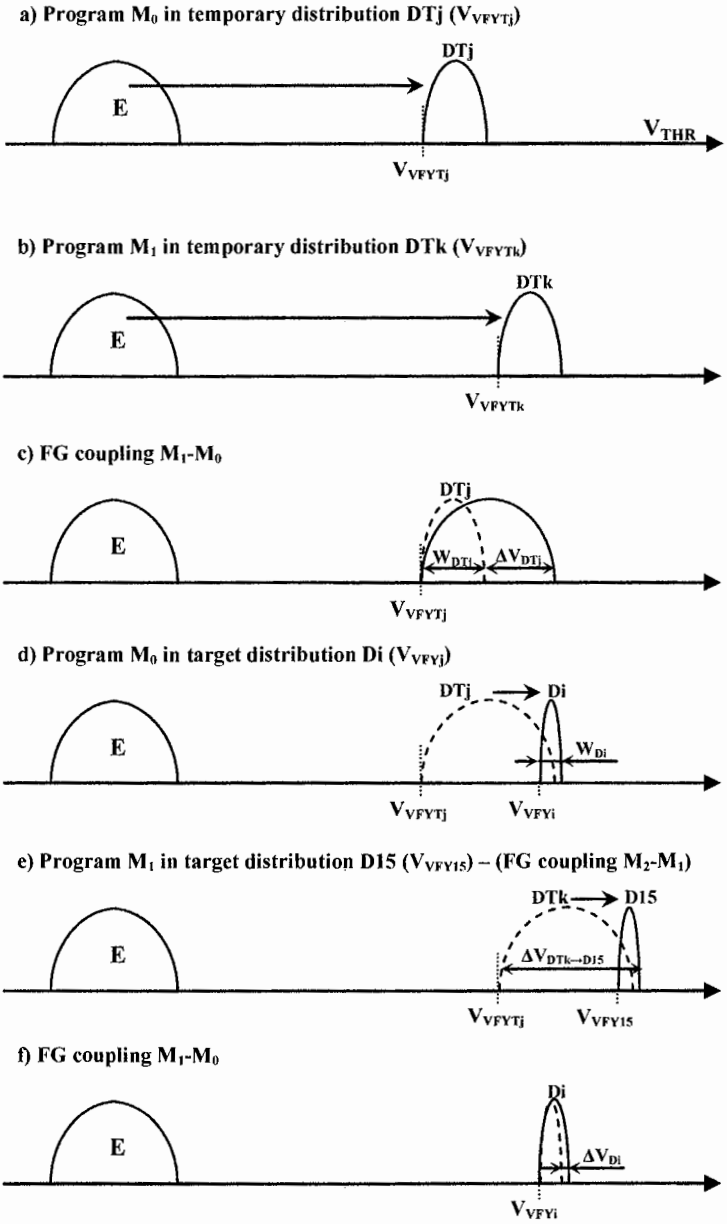


Fig. 16.23. 16LC programming scheme

Petitioner Micron Ex-1021, 0474

Afterwards cell M_1 is roughly programmed as well, at a level V_{VFYTk} which is lower than the target level V_{VFY15}, Fig. 16.23b. Therefore, the distribution of cell M_0 (DTj) gets wider because of FG coupling M_1–M_0 (Fig. 16.23c).

It should be noted that the figure also shows the case where the target of M_1 is distribution E, on top of the worst case whose target is D15: if the target of M_1 is E, there is no FG coupling effect (no shift of distributions) between M_0 and M_1. Distribution DTj (solid line) includes such two extreme cases (and therefore also the intermediate cases M_1 with target D1–D14).

At this point, cell M_0 is programmed to target distribution D_i by means of a ΔISPP which is appropriate for a 16LC programming (Fig. 16.23d). Finally (Fig. 16.23e) cell M_1 is also programmed to target level V_{VFY15} in the same way. Figure 16.23e also shows the widening of distribution DTk due to FG coupling M_2–M_1 because of roughly-programming of M_2. Roughly-programming of cell M_2 takes place after programming cell M_0 to its target distribution D_i.

The widening of distribution D_i (ΔV_{Di} in Fig. 16.23f) is only influenced by the second programming round of M_1, and it is equal to (according to Eq. (16.2)):

$$\Delta V_{Di} = \frac{C_{FGNC}}{C_{FGTOT}} \cdot \Delta V_{DTK \rightarrow D15} \tag{16.5}$$

where $\Delta V_{DTk \rightarrow D15}$ is the maximum threshold leap of M_1, in the step shown in Fig. 16.23e. The reduction of FG coupling M_1–M_0 with respect to a program where M_1 is programmed in a single 16LC round from E to D15, is approximately 90%.

The more the temporary levels V_{VFYTj} are close to the target level V_{VFYi}, the more the FG coupling given by Eq. (16.5) decreases.

Indeed a lower limit exists for the distance between these two levels, given by the fact that the rightmost limit of D_i must be greater than the rightmost limit of distribution DTj (Fig. 16.23d). Without this condition, programming shown in Fig. 16.23d would generate distributions similar to the one drawn with solid line in Fig. 16.24.

Therefore, following condition must be valid:

$$V_{VFYi} + W_{Di} \geq V_{VFYTj} + W_{DTj} + \Delta V_{DTj} \tag{16.6}$$

where W_{Di} is the width of target distribution D_i and W_{DTj} is the width of temporary distribution DTj.

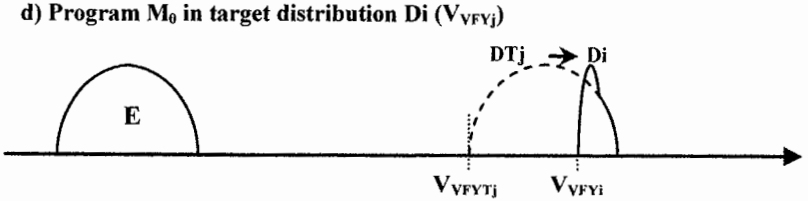


Fig. 16.24. Temporary level V_{VFYTj} and target level V_{VFYi} too near

Petitioner Micron Ex-1021, 0475

The value ΔV_{TDJ} is the FG coupling contribution due to the program leap of cell M_1 from E to D15 ($\Delta V_{E\rightarrow D15}$), which can be computed using the following equation:

$$\Delta V_{DTj} = \frac{C_{FGNC}}{C_{FGTOT}} \cdot \Delta V_{E\rightarrow D15} \tag{16.7}$$

Once target widths W_{Di} and W_{DTj} are set, the distance between levels V_{VFYTj} and V_{VFYi} is strictly related to the value given by Eq. (16.7). Because of the ongoing technology shrink, this value is due to increase and therefore the distance between V_{VFYTj} and V_{VFYi} increases as well. The drawback is that if the distance between V_{VFYTj} and V_{VFYi} increases, the final contribution of FG coupling (given by Eq. (16.5)) increases as well. Even if this method is highly efficient, its limits are more and more highlighted as shrink is performed.

Another disadvantage is that this programming sequence needs a corresponding data upload sequence which is quite complex:

- Upload (to the sensing circuit) of pages 1–4 to perform rough programming of M_0
- Upload of pages 5–8 to perform rough programming of M_1
- Reload of pages 1–4 to perform 16LC programming of M_0
- Upload of pages 9–12 to perform rough programming of M_2
- Reload of pages 5–8 to perform 16LC programming of M_1
- Upload of pages 13–16 to perform rough programming of M_3
  Reload of pages 9–12 to perform 16LC programming of M_2 and so on… so on.

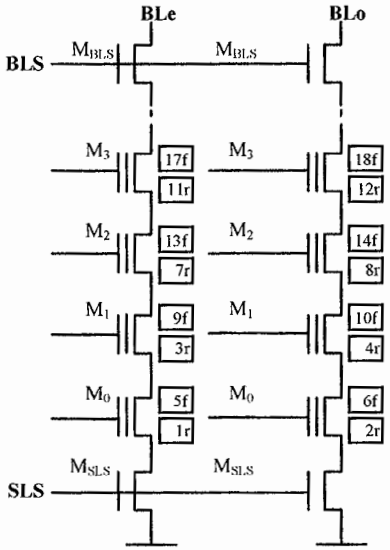


Fig. 16.25. 16LC programming sequence for an even/odd interleaved architecture

Petitioner Micron Ex-1021, 0476

Unless a sensing circuit including eight latches is developed (four for programming and four for buffering), this page load sequence should be managed by the system microcontroller, which is able to act as a buffer for the four latches of the sensing circuit.

The sequence described above is valid for ABL architecture, but it can also be used in an interleaved architecture [7]. In this case, the cells belonging to the odd bitline should be roughly-programmed before performing 16LC programming on even cell M_0.

The resulting programming sequence is shown in Fig. 16.25: the number represents programming order, while the letter "r" represents the "round of rough-programming" and "f" represents the "round of 16LC programming".

16.4.1 Three rounds re-programming sequence

A more efficient technique for 16LC programming, which programs the cells in three rounds, is shown in Fig. 16.26 [8].

In the first round of M_0, the cell is programmed 4LC-like (Fig. 16.26a) in distributions DTj.

Then the adjacent cell M_1 is programmed 4LC-like as well (first round of M_1). FG coupling effect on M_0 due to programming of M_1 is shown in Fig. 16.26b.

In the second round, cell M_0 is roughly-programmed 16LC-like reprogramming distributions DTj into distributions D_i' (Fig. 16.26c).

Similarly to what we have seen in the previous 16LC programming, reprogramming of DTj in D_i' should not generate distributions like the ones shown in Fig. 16.24.

Therefore, the rightmost values V_{RDT1}, V_{RDT2}, and V_{RDT3} of distributions DT1, DT2, and DT3 should be lower than the rightmost values of D'4, D'8, and D'12. For the same reason, distributions D_i' should be programmed to a level which is lower than the final target D_i.

Afterwards the cells M_2 and M_1 are programmed 4LC-like (first round of M_2) and rough-16LC-like (second round of M_1) respectively. Figure 16.26d shows FG coupling effect on M_0 due to these two program operations.

At this point the third round of reprogramming of cell M_0 (Fig. 16.26e) is performed at target levels D_i. Such final program is performed using a fine ΔISPP, 16LC-like.

Then cells M_3, M_2, and M_1 are reprogrammed 4LC-like (first round of M_3), rough-16LC-like (second round of M_2) and 16LC-like (third round of M_1) respectively.

Figure 16.26f shows FG coupling effect on M_0 due to these three program operations.

FG coupling effect on M_0 is mainly due to the third round of reprogramming of the distributions of M_1 from D_i' to D_i: FG coupling on M_0 is minimized.

Figure 16.27 shows program sequence for ABL architecture.

Petitioner Micron Ex-1021, 0477

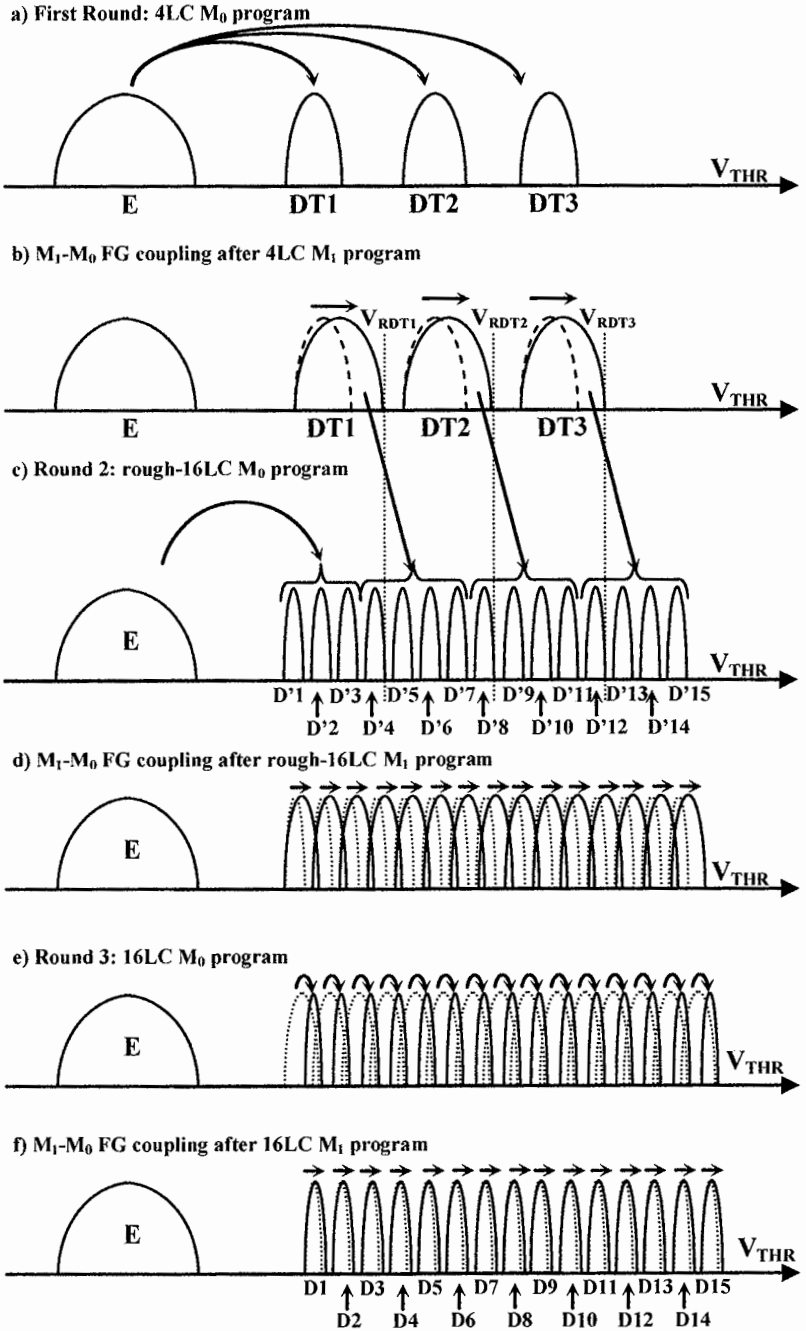


Fig. 16.26. 16LC three round programming method

Petitioner Micron Ex-1021, 0478

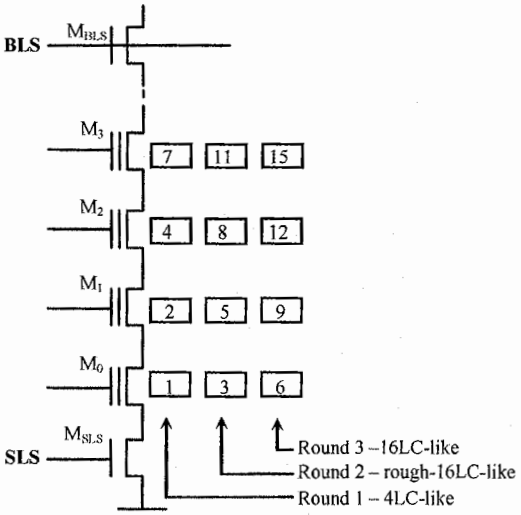


Fig. 16.27. 16LC three rounds programming sequence

16.4.2 Sequential sensing

The sequence of 15 verify steps (Fig. 16.2) significantly impacts overall program time. Figure 16.28 (upper part) shows the signals applied to the NAND string terminals in this phase. It is composed of a sequence of 15 SROs performed at levels V_{VFYi}.

As can be noted (see also Chap. 8), most of this time is wasted waiting for the operation of both charge transient T_{PRE} and discharge transient T_{DIS} of the WL. In addition to that, the biasing of the cells gates at V_{PASS} generates a coupling disturb (V_{WLCC}) on the WL of cell M_i biased at V_{VFYi}. The recovery of such disturb lasts quite a relevant amount of time.

Figure 16.28b is an alternate solution which minimizes these effects [8]. V_{PASS} has an initial transient, and then it remains constant. Therefore, both charge and discharge transients of V_{PASS} and the coupling with V_{VFYi} are eliminated. In the overall duration of a SRO, the only relevant contributions are given by charge transients T'_{PRE} (smaller than T_{PRE}) of V_{VFYi} and evaluation timings T_{EVA}. It is important to note that in case of ABL architecture T_{EVA} becomes negligible.

Settling time of the SL (source line) impacts the duration of a SRO as well. One way to reduce SL noise, (similar to the one described in Sect. 9.3.2) is to exclude from the verify operation at V_{VFYi} the cells which are below the previous level V_{VFYi-2} [8].

In this way the current injected into the source is greatly reduced, together with the peak of SL noise.

The same technique can be applied to the read sequence as well, starting from level V_{READ1} up to the level V_{READ15}.

Petitioner Micron Ex-1021, 0479

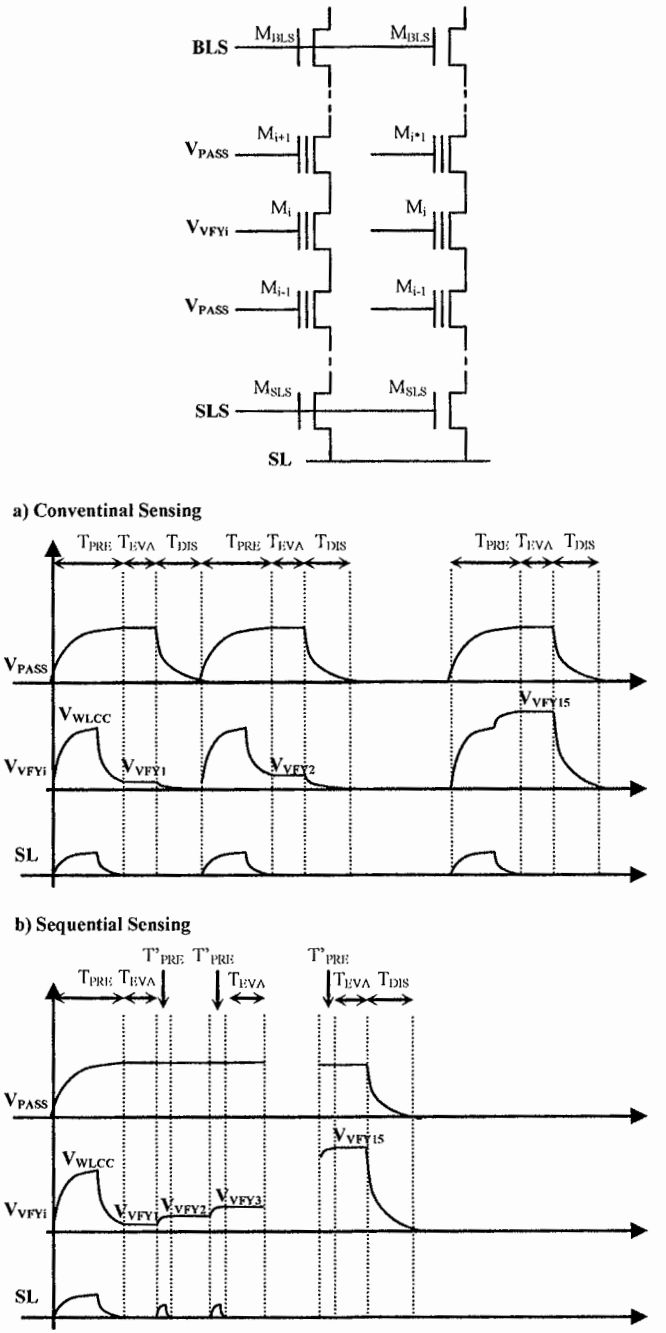


Fig. 16.28. 16LC Conventional sensing vs. sequential sensing

Petitioner Micron Ex-1021, 0480

Read and program circuits used in 16LC are not described in details, since they are a logical extension of the ones used in 8LC: in the following, the schematics are sketched.

Figure 16.29 shows the sensing circuit for program-verify (the additional fourth latch L4 can be noted). Tables 16.5 and 16.6 show the initial state of the latches. After a verify operation at V_{VFYi}, transistors of the corresponding column in Tables 16.7 and 16.8 are enabled.

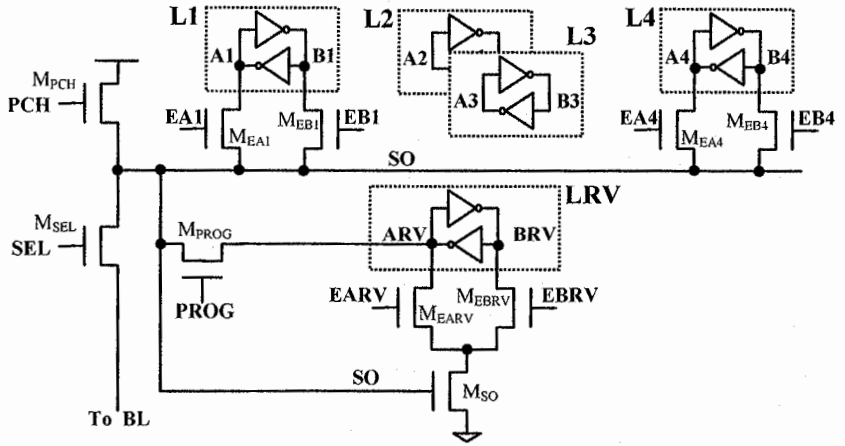


Fig. 16.29. 16LC program circuit

Table 16.5. Logic content of latches of Fig. 16.29 for 16LC distributions

	E	D1	D2	D3	D4	D5	D6	D7
ARV	1	0	0	0	0	0	0	0
A4	1	0	1	0	1	0	1	0
A3	1	1	0	0	1	1	0	0
A2	1	1	1	1	0	0	0	0
A1	1	1	1	1	1	1	1	1

Table 16.6. Logic content of latches of Fig. 16.29 for 16LC distributions

	D8	D9	D10	D11	D12	D13	D14	D15
ARV	0	0	0	0	0	0	0	0
A4	1	0	1	0	1	0	1	0
A3	1	1	0	0	1	1	0	0
A2	1	1	1	1	0	0	0	0
A1	0	0	0	0	0	0	0	0

Petitioner Micron Ex-1021, 0481

Table 16.7. Transistors biased (on the gate) to V_L for each verify operation

	V_{VFY1}	V_{VFY2}	V_{VFY3}	V_{VFY4}	V_{VFY5}	V_{VFY6}	V_{VFY7}
M_{EX4}	M_{EB4}	M_{EA4}	M_{EB4}	M_{EA4}	M_{EB4}	M_{EA4}	M_{EB4}
M_{EX3}	M_{EA3}	M_{EB3}	M_{EB3}	M_{EA3}	M_{EA3}	M_{EB3}	M_{EB3}
M_{EX2}	M_{EA2}	M_{EA2}	M_{EA2}	M_{EB2}	M_{EB2}	M_{EB2}	M_{EB2}
M_{EX1}	M_{EA1}	M_{EA1}	M_{EA1}	M_{EA1}	M_{EA1}	M_{EA1}	M_{EA1}

Table 16.8. Transistors biased (on the gate) to V_L for each verify operation

	V_{VFY8}	V_{VFY9}	V_{VFY10}	V_{VFY11}	V_{VFY12}	V_{VFY13}	V_{VFY14}
M_{EX4}	M_{EA4}	M_{EB4}	M_{EA4}	M_{EB4}	M_{EA4}	M_{EB4}	M_{EA4}
M_{EX3}	M_{EA3}	M_{EA3}	M_{EB3}	M_{EB3}	M_{EA3}	M_{EA3}	M_{EB3}
M_{EX2}	M_{EA2}	M_{EA2}	M_{EA2}	M_{EA2}	M_{EB2}	M_{EB2}	M_{EB2}
M_{EX1}	M_{EB1}	M_{EB1}	M_{EB1}	M_{EB1}	M_{EB1}	M_{EB1}	M_{EB1}

Figure 16.30 shows the timings of the enable signals of the transistors during a verify operation. The circuits of Figs. 16.18 and 16.19 are obviously valid for a 16LC as well.

As far as read is concerned, the corresponding circuit is shown in Fig. 16.31. Assuming that the coding shown in Fig.16.32 is used, Table 16.7 shows the enable sequence of the transistors associated with a SRO at V_{READi}.

Due to the fact that high storage capacity applications are becoming ubiquitous, Flash vendors are spending more and more effort on XLC memories. In fact, technology shrink is facing its physical limits while 3D options (Chap. 5) are still at R&D level. Given the effort, in the near future we expect some improvements of performances and reliability to satisfy the needs of the consumer market.

Table 16.9. Enabled transistor after the relative SRO at V_{READi}

	V_{READ1}	V_{READ2}	V_{READ3}	V_{READ4}	V_{READ5}	V_{READ6}	V_{READ7}	V_{READ8}
M_{RX4}	M_{RA4}	M_{RB4}	M_{RA4}	M_{RB4}	M_{RA4}	M_{RB4}	M_{RA4}	M_{RB4}
M_{RX3}		M_{RA3}		M_{RB3}		M_{RA3}		M_{RB3}
M_{RX2}				M_{RA2}				M_{RB2}
M_{RX1}								M_{RA1}

Table 16.10. Enabled transistor after the relative SRO at V_{READi}

	V_{READ9}	V_{READ10}	V_{READ11}	V_{READ12}	V_{READ13}	V_{READ14}	V_{READ15}
M_{RX4}	M_{RA4}	M_{RB4}	M_{RA4}	M_{RB4}	M_{RA4}	M_{RB4}	M_{RA4}
M_{RX3}		M_{RA3}		M_{RB3}		M_{RA3}	
M_{RX2}				M_{RA2}			
M_{RX1}							

Petitioner Micron Ex-1021, 0482

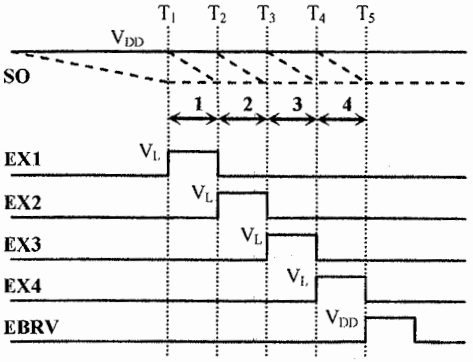


Fig. 16.30. Timing diagram related to Tables 16.7 and 16.8

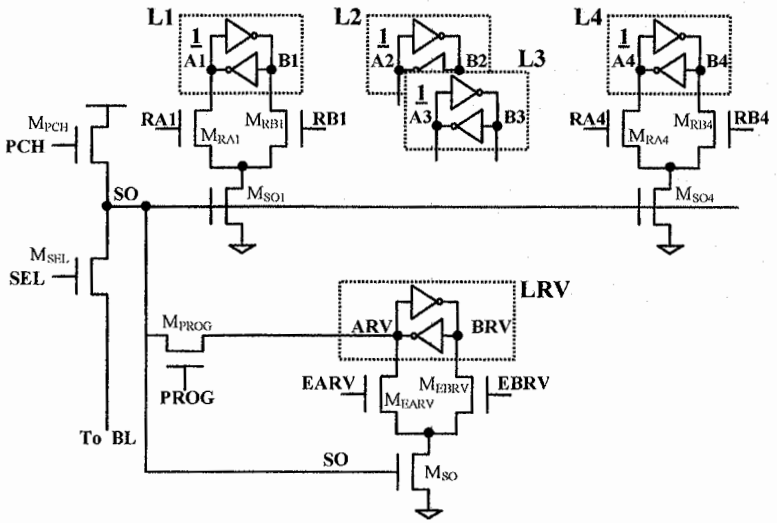


Fig. 16.31. 16LC read circuit

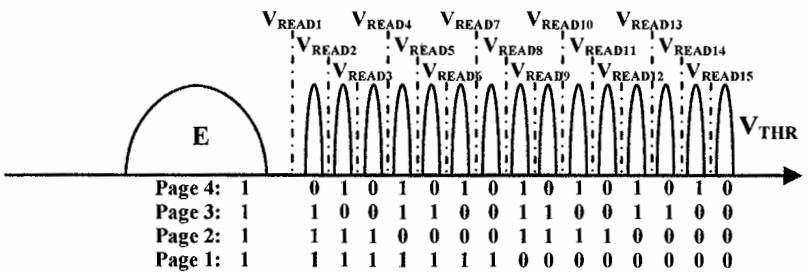


Fig. 16.32. 16LC coding

Petitioner Micron Ex-1021, 0483

References

1. Noboru Shibata et al. U.S. Patent No. 7443724 – *Semiconductor memory device for storing multivalued data* Assignee: Kabushiki Kaisha Toshiba (Tokyo, JP).
2. Yan Li et al. *A 16Gb 3b/Cell NAND Flash Memory in 56nm with 8MB/s Write Rate* Solid-State Circuits Conference, 2008. Digest of Technical Papers. ISSCC 2008 IEEE International Feb. 2008 Page(s): 506–507, 632.
3. Yan Li et al. *A 16 Gb 3-Bit Per Cell (X3) NAND Flash Memory on 56 nm Technology With 8 MB/s Write Rate,* Solid-State Circuits, IEEE Journal of Volume 44, Issue 1, Jan. 2009 Page(s):195–207.
4. Seung-Ho Chang et al. *A 48nm 32Gb 8-level NAND Flash memory with 5.5MB/s program throughput* Solid-State Circuits Conference, 2009. Digest of Technical Papers. ISSCC. 2009 IEEE International Feb 2009 Page(s): 240–241,241a.
5. Takuya Futatsuyama et al. *A 113mm^2 32Gb 3b/cell NAND Flash memory* Solid-State Circuits Conference, 2009. Digest of Technical Papers. ISSCC 2009 IEEE International Feb. 2009 Page(s): 242–243.
6. Marotta G. G. et al. *A 3bit/Cell 32Gb NAND Flash Memory at 34nm with 6MB/s Program Throughput and with Dynamic 2b/Cell Blocks Configuration Mode for a Program Throughput Increase up to 13MB/s* Solid-State Circuits Conference, 2010. Digest of Technical Papers. ISSCC 2010 IEEE International Feb 2010 Page(s): 444–445.
7. Shibata, N. et al. *A 70 nm 16 Gb 16-Level-Cell NAND Flash Memory* IEEE Journal of Solid-Stare Circuits, Vol. 43, No. 4, April 2008 Page(s):929–937.
8. Trinh, C. et al. *A 5.6MB/s 64Gb 4b/Cell NAND Flash memory in 43nm CMOS* Solid-State Circuits Conference, 2009. Digest of Technical Papers. ISSCC 2009 IEEE International Feb 2009 Page(s): 246–247.

Petitioner Micron Ex-1021, 0484

17 Flash cards

A. Ghilardelli[1] *and S. Corno*[2]

17.1 Introduction

Memory cards are solid-state devices engineered to store digital information. They come in several forms and shapes but nonetheless they have many common characteristics. In this chapter, we will describe memory cards from a user standpoint, their internal architecture and the algorithms operating within, the difficulties with relevant counter-stratagems inherent in their design.

Each of us has certainly already dealt with memory cards in our life. Almost every portable digital device has a slot for housing one of them. Digital still cameras, handheld computers, MP3 players, mobile phones, digital camcorders and video game consoles all make use of memory cards. In digital still cameras they store pictures, in MP3 players music files, in digital camcorders videos, and their applications are literally endless. Moreover, every computer nowadays has a card reader slot by means of which the information stored in the memory card can readily be transferred to it, without the use of cable connections or installation of dedicated software.

Add to that other indisputable advantages like reliability, affordability, high capacity, high speed, small size, ruggedness and it is clear why memory cards are so popular today. Memory cards are in the astounding range of gigabytes nowadays and they cost just a few dollars. They are tremendously fast, tens of megabytes transferred per second is a very common capability and sometimes even over 100 Mbyte/s. They are small, sometimes to the point to make their handling a challenging task, and light, with a weight in the range of the gram. They do not require particular care, the user can manhandle them, drop them and even subject them to extreme temperatures and still they will do their job efficiently and reliably. As they do not have internal moving parts, contrary to hard disk drives for instance, they can operate trustworthily even when hard shaken. They can be overwritten over and over again, probably never reaching their cycling life limit under normal usage.

The market is replete with different kinds of memory cards. The most common are undoubtedly the MultiMediaCard (MMC) and the Secure Digital (SD) with all

[1] Micron, aghilard@micron.com
[2] Micron, scorno@micron.com

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_17,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0485

their variations. As in all microelectronics industry, however, this is a fast changing and evolving condition. Formerly very widespread cards were the CompactFlash and the SmartMedia, although now rather superseded. More vendor specific cards are Memory Stick and xD. USB Flash Drive are yet another type of card very popular today because they have no restrictions on their size like all the aforementioned cards and can thus be very sizeable in available memory.

The following Table 17.1 summarizes the dimensions of the most widespread memory cards.

Table 17.1. Dimensions of selected memory cards

Card	Common varieties	Dimensions (mm)
CompactFlash	I	36 × 43 × 3.3
	II	36 × 43 × 5.0
SmartMedia		37 × 45 × 0.76
MultiMediaCard	MMC	24 × 32 × 1.4
	Reduced size	24 × 18 × 1.4
Secure Digital	SD	24 × 32 × 2.1
	microSD	11 × 15 × 1.0
Memory Stick	Standard	21.5 × 50 × 2.8
	Pro	21.5 × 50 × 2.8
	Duo	20 × 31 × 1.6
xD	Standard	25 × 20 × 1.7
	Type M	25 × 20 × 1.7
	Type H	25 × 20 × 1.7
USB Flash Drive		Arbitrary

Host manufacturers have a multitude of cards to pick from for their applications. Each card may have its own advantages or disadvantages in terms of form factor, size, capacity, speed, protocol complexity or pin count. Sometimes, however, more mundane motives underlie the card choice. These might include licenses and royalties that come with some cards' types.

Security is also an increasing decisive factor that may tip the scale toward one card rather than another. Some cards implement protocol features specifically devised to limit somehow the access to the card content. These features are known as *digital rights management* (*DRM*) and are mostly used by copyright owners.

17.2 Memory card architecture and assembly

One of the foremost common characteristics of memory cards is their use of NAND Flash memories as storage devices, although this is not compulsory in their definition. Indeed, memory cards with NOR Flash memories have been designed and in the near future even other futuristic memories like the Phase Change Memories (PCM) might be used, too.

Petitioner Micron Ex-1021, 0486

All memory cards have a similar architecture. A simple block diagram is represented in Fig. 17.1. The core of the card is a Micro-Controller, which masters all the card's activities. A stack of NAND memories constitutes the memory space where data are actually stored. The Micro-Controller communicates with the stack of all the NAND memories through a *Memory Interface*. The Memory Interface implements the NAND protocol to exchange information with the memory stack. It must comply with the specifications dictated by the NAND manufacturer, so a different NAND type usually necessitates a different Memory Interface.

The Micro-Controller must also communicate with the external world. This task is mandated to the *Protocol Interface*. Each type of card (MMC, SD, Compact Flash...) has to obey to its own protocol.

The three blocks Protocol Interface, Micro-Controller and Memory Interface constitute a single die, the Controller. The NAND memories are separate dice. All these dice must be assembled in the same package. This is challenging because the thickness of the card package is very thin (1–3 mm depending on the card). Different strategies can be applied to the package assembly.

One method is to assemble the Flash, typically in a TSOP or WSOP package, and the Controller, typically in an LGA package, together through surface mount technology (SMT). A simple schematic diagram is shown in Fig. 17.2.

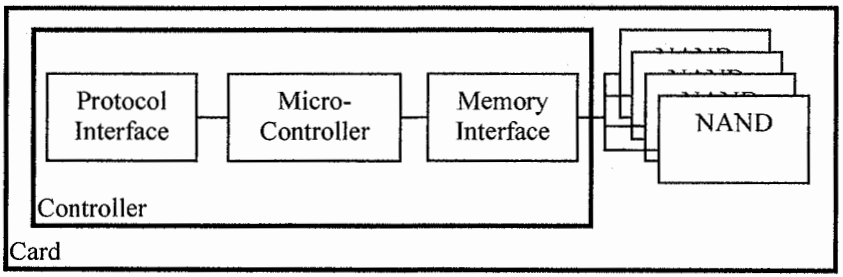


Fig. 17.1. Memory card's architecture

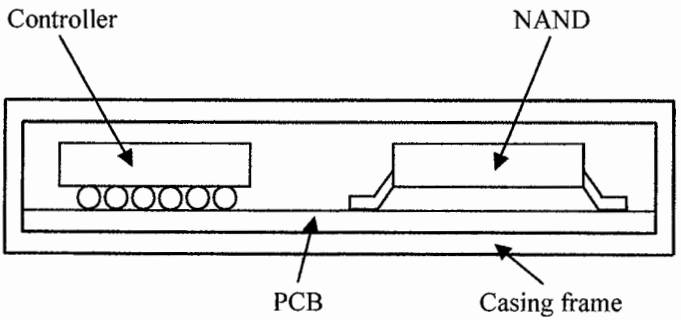


Fig. 17.2. Surface mount technology (SMT)

Petitioner Micron Ex-1021, 0487

A *Multi-Chip Package* (*MCP*) is a package containing several dice of the same type stacked vertically and connected in such a way to form a single bigger device. In Fig. 17.2, the NAND can be a single die or an MCP of stacked NAND dice. A *System-in-Package* (*SiP*) is a package combining different devices (digital, analog, passives...) forming a system. The whole card package is thus a SiP.

Another possible approach is to bond the bare chips directly to the substrate, without putting them in their own package first. This technique is dubbed Chip on Board (COB) and has the obvious advantage of letting more room within the card package.

17.3 Memory card specifications

As an example of card protocol and specifications, we can look in deeper details at a particular card type, the Embedded MultiMediaCard (eMMC). In particular, we will refer to the JESD84-A44 datasheet (MMCA 4.4), but all that will be described here still holds true with minor changes for any other version of MMC. Secure Digital cards are very similar to MMC cards, too.

17.3.1 Pinout

The pin out of the eMMC card is very simple. Apart from the supply voltage pins there is a reset pin, the ever-present clock (*CLK*), one command line (*CMD*) and eight data lines (*DAT<7:0>*). The CMD line is used by the host to issue commands to the card and by the card to send responses to the host. The DAT lines are used to exchange data between the host and the card.

17.3.2 Commands and responses

Commands are sent by the use of command tokens, whose coding scheme is 48 bit long and sent via the CMD line one bit at a time:

	Start Bit	Transmission Bit	Command Index	Argument	CRC7	End Bit
Width (bits)	1	1	6	32	7	1

The Start Bit, the Transmission Bit and the End Bit have a fixed pattern and their purpose is simply to signal the start and the end of the command token. The *Command Index* is 6 bit long, thus providing 64 possible commands. Further commands are made available through a special sequence of two command tokens. The *Argument* field has a different meaning according to the command issued.

Petitioner Micron Ex-1021, 0488

The *CRC* (*Cyclic Redundancy Check*) is an error detection code. It checks the integrity of the command, which could be received incorrectly because of transmission errors. The CRC is computed over the command token. Upon reception of the command it is computed independently by the card and compared with the one received in the command. If they match, the command is executed otherwise an error is signaled.

Typical examples of commands are the Read and Write Single Block commands. Through these commands, a Block (usually 512 byte long) is read from or written to the card. Please note that the card write command can write both 0 and 1 and not just 0 as in Flash memories. The Command Index for the Read Single Block command is 17, while the one for the Write Single Block is 24. In both cases, the Argument is the data address to be read or written. The Argument field of just 32 bits should not mislead into believing that only $2^{32} = 4$ Gbyte could be addressed, because for card with a capacity greater than 2 Gbytes the addressing is on a sector basis. Consequently, any card density can be accessed with a 32-bit field argument.

The Read and Write Single Block commands are obviously followed by the data transfer, from the host to the card for the Write Block and from the card to the host for the Read Block.

Each command is responded to by the card. The response serves the purpose for the host to have feedback information about the command just issued, like whether the card has correctly received the command, has executed or aborted it etc. Sometimes the response itself contains the information requested by the host through the command. There are a few type of responses, the most common having a code length of 48 bits and the following structure:

	Start Bit	Transmission Bit	Command Index	Card Status	CRC7	End Bit
Width (bits)	1	1	6	32	7	1

The fields are clearly similar to the command token. Here the Command Index is the replica of the command being responded to. The Card Status is a card register stashing a lot of useful information about the card status and the outcome of the command being executed.

We can now delineate the whole behavior of the host-card communication during a Read and Write operation.

In a Read Single Block, the host issues the command on the CMD line (Fig. 17.3). The card sends its response indicating in the Status Register whether the command has been accepted or declined for any reason (address out of range, CRC failed...).

Upon reception of the command, the card starts retrieving the requested data. As soon as the data are fetched and ready to be transmitted, the card sends them via the DAT line.

Petitioner Micron Ex-1021, 0489

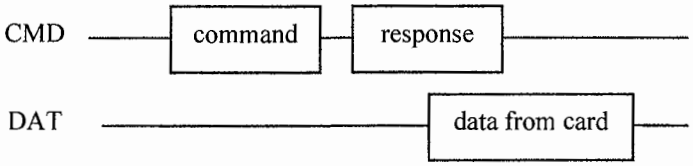


Fig. 17.3. Read Single Block

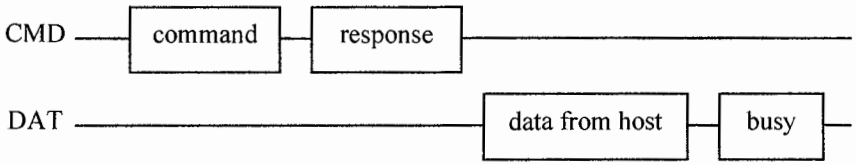


Fig. 17.4. Write Single Block

The Write Single Block command is very similar (Fig. 17.4).

This time the data to be programmed are sent by the host to the card. For this to happen, the host must wait for the card response to ensure the command has been accepted. After the data have been transmitted, the card signals on the DAT line that it is busy programming the data, until the programming is over.

We have described just two among the commonest commands, but many others are available. We report a brief list of the most significant ones.

Command Index	Description
0	Reset
5	Sleep
7	Select card
13	Send Status Register
17	Read Single Block
24	Write Single Block
28	Write Protection
35, 36, 38	Erase
42	Lock

Command 0 is self-explanatory: it is a software reset of the card.

In *Sleep* state, the power consumption is minimized. The Sleep command (CMD 5) toggles between this state and the standard consumption state.

More than one card could be connected on the same bus. Command 7 selects which one the host wants to work with out of the many connected to the same bus.

If the host wishes to know the card status at any moment, it can send the *Send Status Register* command (CMD 13). This is particularly useful at the end of some commands, like the Read or Write Single Block, to be informed about the outcome

Petitioner Micron Ex-1021, 0490

of the operation. For instance, the Write command takes a lot of time to complete and if the card encounters any problem in programming the data, this will not appear in the card response to the command (see Fig. 17.4). Instead, an error will develop during the execution of the command (busy phase in Fig. 17.4). The host can acknowledge that, only at the end of the command through a Send Status Register command.

The Read and Write Single Block commands have already been described.

An important feature of a card is the capability to protect data against inadvertent modifications (write and erase). For this reason, a *Write Protection* command is provided. The whole memory is divided into *Write Protect Groups* (whose size is much greater then a Write block, see Fig. 17.5). Each of these groups can be independently protected against accidental writing or erasing through the Write Protection command.

The memory can be explicitly erased by means of the *Erase* commands. The command operates on portions of the memory called *Erase Group* (Fig. 17.5). A contiguous range of Erase Groups can be erased at a time.

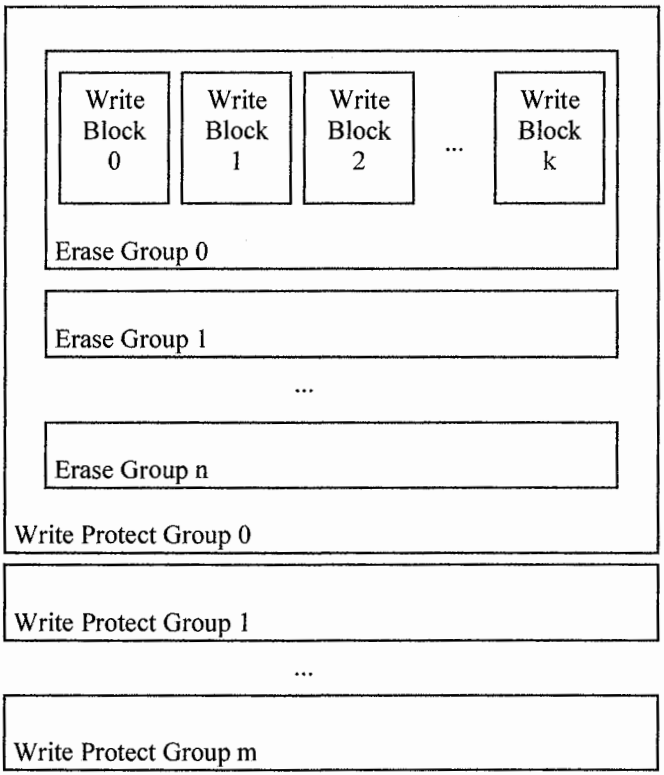


Fig. 17.5. Write Protect Groups

Petitioner Micron Ex-1021, 0491

The erase process is a three steps sequence. First, the host states the start address of the range through command 35, and then it defines the last address of the range through command 36. Finally, the actual erase process is initiated issuing the command 38.

Recently, a *Secure Erase* operation has been introduced. Applications with high security requirements may benefit from this feature. The Secure Erase command differs from the standard Erase command in two respects.

Firstly, it requires that the actual physical erase be executed when the command is issued. All other operations are halted until the Secure Erase has been completed. On the contrary, in the standard erase usually the memory is not physically erased upon reception of the command, but just marked in the file system or by the Flash Translation Layer as erased. In the standard erase, the actual erase will take place when the memory deems it appropriate, at a convenient time, typically when it needs free space for new incoming data.

Secondly, the Secure Erase command also requires the card to perform a purge operation on the groups being erased and on any copies of those groups. The definition of *secure purge* is technology dependent, for a NAND Flash being the process of overwriting all the memory locations being erased with a single character before performing the actual erase.

The purging operation is a sanitization process protecting confidential information against a laboratory attack (i.e. data recovery attempts on media outside their normal operating environment).

Command 42 is the *Lock* command. A locked card cannot be accessed for retrieving information from it or for writing new data. The Lock command locks the card and defines a password, which can be later used to unlock the card. Once unlocked, the card can be accessed normally. This command protects the card from unintentional or malicious misuse.

17.3.3 Registers

Each card is endowed with a few registers. Some of these registers contain information about the card useful to the host. Others permit the setting of some parameters.

An abbreviated list of the internal card registers follows:

Register	Description
Card Status	Card Status
CID	Card Identification
CSD	Card Specific Data
EXT_CSD	Extended CSD
DSR	Driver Stage Register

We have already dealt with the 32-bit *Card Status* register, the one sent back from the card in its response. We can now peek at its most significant fields:

Petitioner Micron Ex-1021, 0492

Bit	Identifier	Description
32	ADDRESS_OUT_OF_RANGE	The address in the command was out of the allowed range.
28	ERASE_SEQ_ERROR	The three-step erase sequence was not carried out correctly.
26	WP_VIOLATION	Attempt to program a write protected block.
25	CARD_IS_LOCKED	Indicates that the card is locked.
24	LOCK_UNLOCK_FAILED	E.g. attempting to unlock a card with the wrong password.
23	COM_CRC_ERROR	Failed CRC check.
19	ERROR	Generic error.

The *CID* register is 128 bits long and stashes a number unique for each card. No two cards will ever happen to have the same identification number. To ensure this uniqueness, each card manufacturer is assigned a manufacturer ID, 8-bit long, which is an integral part of the CID. Other fields in the register define the product name, the product revision, the serial number and the manufacturing date.

The *CSD* register stores information on how to access the card. It is 128-bit wide and somewhat complicated. Let us just list a few of its most meaningful fields.

Field	Description
SPEC_VERS	Specification version supported
TAAC, NSAC	Read access time
TRAN_SPEED	Maximum clock frequency when not in high speed mode
READ_BL_LEN	Maximum read data block length
C_SIZE C_SIZE_MULT	Device size for cards up to 2 GB
VDD_R_CURR_MAX	Maximum read current at maximum VDD
VDD_W_CURR_MAX	Maximum write current at maximum VDD
ERASE_GRP_SIZE ERASE_GRP_MULT	Erase Group size
WP_GRP_SIZE	Write Protect Group size
R2W_FACTOR	Write speed factor
WRITE_BL_LEN	Maximum write data block length

The Extended CSD register contains information about advanced card features. Such features include Double Data Rate (DDR) mode, high-speed mode, high capacity, security feature support, boot operation mode. The Extended CSD register is 512 bytes long. Part of it (the Properties segment) defines the card capabilities, while the other part (the Modes segment) defines the configuration the card is working in. We will omit a detailed description of the Extended CSD.

Petitioner Micron Ex-1021, 0493

The *Driver Stage Register* selects the power strength of the output buffer. This permits to adapt the card's output stage to the transfer rate and to the bus capacitance (which varies for instance according to the bus length and the number of connected cards).

17.4 Flash translation layer

Operating systems usually write data in sectors (e.g. 512 bytes in size). Unfortunately, Flash memories lack the so-called *update-in-place* capability. This means they cannot be altered on a sector (page in NAND nomenclature) basis, because the erase operation involves a whole block (made up of several sectors/pages). Therefore, file systems cannot write ones back to zeros in a single addressed sector. This is awkward for a high-level file system to manage.

In order to alleviate the workload of the operating system, software routines can be implemented to emulate a NAND Flash behavior where single sectors (or NAND pages) can be modified independently. Such software is called *Flash Translation Layer* (*FTL*), which is an intermediate layer between the file system and the storage media (the NAND Flash). The operating system can then write the NAND memory on a page basis without worrying about the details of its physical implementation. Usually it is the Controller in Fig. 17.1 taking care of the FTL.

The FTL achieves its task through a *logical to physical re-mapping*. Hence, when the file system sends the command to write or update a certain logical page, the FTL actually writes it in a different available physical page and updates a table storing the logical to physical mapping (*mapping table*). The page containing the older data is marked as invalid; Fig. 17.6 shows an example.

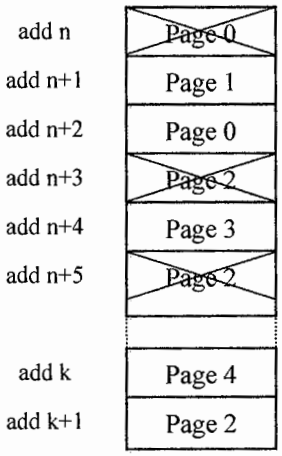


Fig. 17.6. Logical to physical re-mapping

Petitioner Micron Ex-1021, 0494

Here page 0 has been written first at address n, followed by page 1. Then page 0 has been updated, but instead of overwriting it at address n, the FTL writes it in another location (n + 2) and marks address n as invalid.

As it is apparent from Fig. 17.6, after a while a great portion of the memory will be filled with invalid data taking precious unusable space. Eventually, all these locations will have to be erased to make this memory space available to new data. This procedure is not trivial, because the erase operation involves a whole block, comprising several pages, some with valid data, while others with invalid data. Therefore, valid data must be copied first to a new location, before the erase operation takes place. All this procedure is called *garbage collection.* An example is shown in Fig. 17.7.

All valid pages are copied into new locations (the order does not matter because the mapping table takes care of the correct correspondence between the logical and physical address). Then all the memory locations containing the old data (valid and invalid) are erased, thus creating space accessible to new data.

Garbage collection is very critical for many reasons. Let us start to better illustrate this point by introducing the notion of garbage collection efficiency.

Garbage collection efficiency is defined as the ratio of the number of invalid pages in a block to the total number of pages in that block. The higher the invalid pages within a block, the higher the efficiency. Indeed, when the number of invalid pages within a block is high we have a twofold benefit. First, only a few pages must be copied into new locations, thus limiting the program time and reducing the cycling efforts (greater lifetime) of the memory cells. Moreover, when erasing such block, a lot of space will be available for new data, because only few data were valid and still taking memory space.

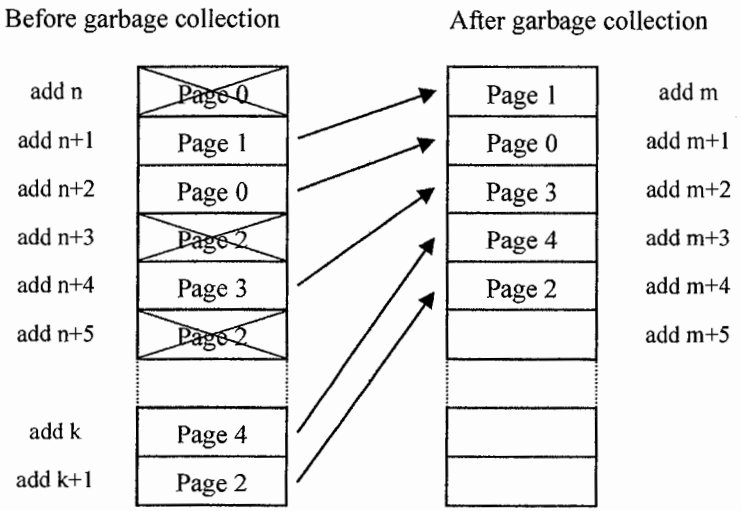


Fig. 17.7. Garbage collection

Petitioner Micron Ex-1021, 0495

When should the garbage collection be run? A high program throughput calls for a lot of free space. If the host requires more space than readily available, it will have to halt its operation and wait for the garbage collection to kick in and erase enough invalid pages. This will limit the sustainable program throughput. So apparently, the garbage collection should be run as often as possible. However, this in turn will limit the garbage collection efficiency. Therefore, an accurate trade-off will have to be determined to establish the best performance of garbage collection.

Flash memories' blocks can only be programmed and erased a limited number of times. This is the reason why a good FTL should also implement a *wear leveling* algorithm. A wear leveling software module makes sure all NAND blocks are cycled an identical number of times, avoiding some blocks ending their cycle life while others being cycled only a few times. Were not a wear leveling algorithm present, blocks with frequently changed data would endure much more cycles than blocks containing long-lived data.

There are two levels of wear leveling. In the *first level wear leveling* new data is programmed into a free block with the fewest write/erase cycles. Two techniques to achieve this are the chain method and the array method.

In the *chain method,* the pool of available blocks is organized in a chain (Fig. 17.8).

A simple round robin algorithm selects one block after another. When a block is erased, it is added to the chain.

In the *array method* an age vector stores the number of each block's program/erase cycles. This information can be used by the wear leveling algorithm to ensure a uniform exploitation of all the blocks.

In the *second level wear leveling* from time to time blocks containing long-lived data are moved (copied) into other blocks even though they never received an update command. The original block can then be used to store new incoming data. The second level wear leveling is triggered when the difference between the maximum and the minimum number of cycles per block reaches a predetermined threshold.

Devising an efficient FTL is a taxing job. Two main concerns are the clustering and the cleaning policy. A few examples of both will be reported, but let start off by introducing the concepts of hot and cold data.

Hot data are the most frequently updated data. *Cold data* (or non-hot data) are seldom updated data. In a given block, it is advantageous to gather (cluster) all hot data or all cold data, because blocks with hot data will soon become garbage. Cleaning such blocks is beneficial because they have high garbage collection efficiency.

How to effectively cluster hot data then? One method is the *Dynamic dAta Clustering* (*DAC*) depicted in Fig. 17.9.



Fig. 17.8. Chain method

Petitioner Micron Ex-1021, 0496

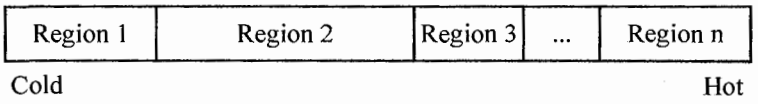


Fig. 17.9. Dynamic dAta Clustering

In the DAC method, the Flash memory is divided into *n* regions, whose size is not necessarily identical. Each region is not even necessarily made up of contiguous memory locations. Region 1 contains the coldest data, Region n the hottest. The hotter a block becomes, the more it shifts towards Region n. Conversely, the colder a block becomes, the more it shifts towards Region 1. When a block is written for the first time, it is put in Region 1. When a block is updated, if it has been into its Region k for a time shorter than a $t_{threshold}$ it is written in Region k + 1, otherwise it is written again in Region k. The $t_{threshold}$ is introduced because a block's hotness degrades as it ages. Upon the cleaning request of a block, if its valid data have been in their Region k longer than $t_{threshold-2}$ they are copied in Region k − 1, otherwise they are still copied in Region k. Each block has associated with it the number of the Region it belongs to.

The real advantage of this technique is twofold. Firstly, the algorithm does not require complex calculations and secondly the classification is fine grained (which in turn improves the performance).

Another clustering method is the *Dynamic Striping*, illustrated in Fig. 17.10. In the Dynamic Striping methodology, two registers called LRU (least recently used) are utilized. Only logical block addresses belonging to the hot list are hot, all the others are cold. The candidate list contains the logical block addresses recently written. If a logical block address belongs to the candidate list and is updated after a short while, it is promoted to the hot list. The last element of the hot list is demoted to the candidate list.

As a further improvement of the Dynamic Striping, hot data are written in the bank with the lowest number of erase cycles to enhance the wear leveling.

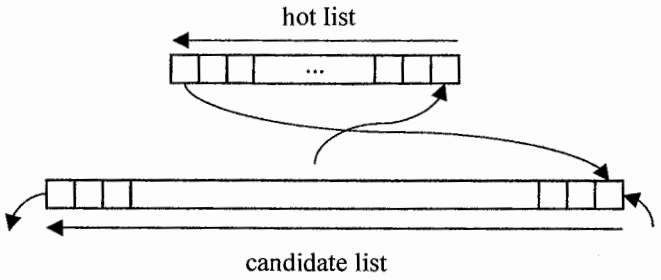


Fig. 17.10. Dynamic striping

Petitioner Micron Ex-1021, 0497

Cold data are written in the bank with the lowest capacity utilization (i.e. the ratio of the number of live pages to the number of total pages) in order to make the capacity utilization more uniform (cold data will remain longer in their locations).

Cleaning policies relate to the criterion used to pick up the block to be erased for garbage collection purposes.

The *greedy policy* is one of the simplest cleaning policies, in that it selects for erasing the block with the greatest number of invalid data. Though simple, this policy is not very efficient because it does not take into account the cycling the blocks have been subjected to. It has also been proved ineffective in case of localities of reference (i.e. the fact that if a certain memory location is referenced at a given time, it is probable that the neighboring memory locations will be referenced in the near future).

A better cleaning policy is the *cost-benefit policy*. The block chosen for erase is the one with the greatest value of the following expression:

$$\frac{age \cdot (1 - valid)}{2 \cdot valid} \tag{17.1}$$

where *valid* is the percentage of valid data in the block, and *age* is the time elapsed since the last modification. Note that (1-*valid*) is the percentage of invalid data, i.e. the space that ultimately will be freed. The term 2·*valid* can be conceptually regarded as the cost of reading the valid data and to copy them in another location (2·*valid* = *valid* + *valid* and the first term is proportional to the time required to read the valid data while the second term is proportional to the time required to copy them in a new location). Therefore, the ratio (1-*valid*)/2·*valid* in the above expression can be considered as a benefit over cost ratio. This policy is better than greedy's because it takes into account the age of the block.

Yet another cleaning policy is the *Cost Age Time* (*CAT*) *policy*. The block chosen for erase is the one with the minimum value of the following expression:

$$\frac{valid}{1 - valid} \cdot \frac{1}{age} \cdot n_C \tag{17.2}$$

where *valid* and *age* are defined as in the previous case and n_C is the number of cleaning, i.e. the number of times a block has been erased. This cleaning policy is endowed even with a sort of wear leveling thanks to the term n_C.

A good FTL must also take care of *bad blocks*. Bad blocks contain not-working or unreliable bits and therefore cannot be used. They may be present since the manufacturing of the device or may develop during its lifetime. The FTL must store all the necessary information about which blocks can be used and which cannot. When a program operation in a certain block fails, the FTL acknowledges that through the NAND Status Register and acts copying the valid data of the block into a new good block; the block where the fail occurred is then be marked as bad.

Finally, a FTL must ensure the reliability of data in the event of sudden power loss. In order to guarantee this, the FTL must wait for the program operation to

Petitioner Micron Ex-1021, 0498

complete successfully before marking the old page as invalid or before erasing a block. In such a way, if the new data is lost due to a power failure, the old data can still be recovered.

17.5 Cryptography

Security has become an important part of cards' characteristics. New protocols implement commands handling confidential data with the utmost caution. Let us mention the standard JESD84-A44 (eMMC), which states that every card complying with it must implement a special memory area called *Replay Protected Memory Block* (*RPMB*), which must be protected against replay attacks. In a *replay attack* a valid data transmission is intercepted first, recorded and then played back again later on by an attacker. Hence, cryptography has become fundamental even in card design and usage. To protect a portion of the memory against replay attacks the eMMC protocol employs a key (MAC) and a nonce. To better understand these concepts, let us briefly review a few pivotal definitions and notions about cryptography.

A *cipher* is an algorithm to *encrypt* and *decrypt* a *plaintext* into and from a *ciphertext* (Fig. 17.11).

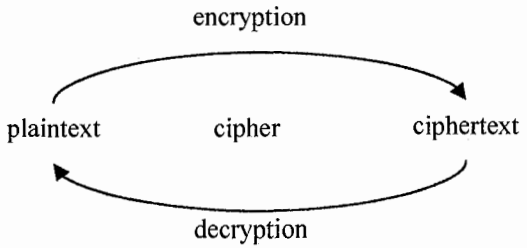


Fig. 17.11. Cipher

Note that in the everyday vernacular language sometimes the terms cipher and code are used interchangeably. However, in cryptography a *code* has a *codebook* associating each word or phrase of the plaintext with one or more codewords. Therefore, the plaintext set is somewhat limited and must be anticipated a priori. Codes operate according to the meaning while ciphers operate according to a single letter or bit.

Block ciphers work on blocks whereas *stream ciphers* work on continuous streams of data. Each cipher needs a *key* or *cryptovariable*. *Symmetric key algorithms* have the same key for encryption and decryption. *Asymmetric key algorithms* have different keys for encryption and decryption. In symmetric key algorithms, the key must be known only by the sender and by the recipient.

A *cryptographic hash function* is a function mapping a *message* into a message *digest* (a.k.a. *hash value*) as depicted in Fig. 17.12.

Petitioner Micron Ex-1021, 0499

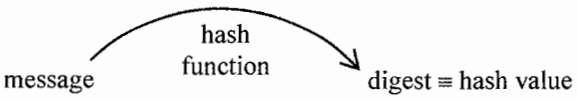


Fig. 17.12. Cryptographic hash function

The digest is a fixed-size bit string (the above-mentioned eMMC protocol employs a 256 bit digest size). Obviously, the hash function is not injective, but is a many-to-one mapping (the domain is usually greater than the codomain). A hash function must possess the following properties:

- *Preimage resistance*. Given a hash value h, it must be hard to find a message m such that h = hash(m). This property is required to not be vulnerable to *preimage attacks*.
- *Second preimage resistance* (a.k.a. *weak collision resistance*). Given a message m_1 it must be hard to find another message m_2 such that $\text{hash}(m_1) = \text{hash}(m_2)$. This is necessary to not be vulnerable to *second preimage attacks*.
- *Collision resistance* (a.k.a. *strong collision resistance*). It must be hard to find two messages $m_1 \neq m_2$ such that $\text{hash}(m_1) = \text{hash}(m_2)$. This is needed against *birthday attacks*.

A *MAC* (message authentication code) is a cryptographic hash function taking as inputs a secret key and an arbitrary-length message. The output is the MAC (a.k.a. *tag*). *HMAC* is the acronym of hash MAC. The cryptographic strength of a *HMAC* depends on its hash function, its key and the number of its output bits (which equals the digest size of its hash function). An HMAC is mathematically defined as:

$$HMAC(k,m) = H\big((k\,XOR\,opad)\,\&\,H\big((k\,XOR\,ipad)\,\&\,m\big)\big) \tag{17.3}$$

where:

- k is the secret key.
- m is the message.
- H is the hash function.
- *ipad* and *opad* are constants.
- & is the concatenation operator.

A *nonce* (number used once) is a random number used to avert a replay attack that must never be used more than once.

We are now ready to get an insight to the essence of the host and eMMC card communication protocol regarding the Replay Protected Memory Block. First of all, a 32-byte long authentication key is programmed inside the card, typically during the production testing phase. This key is stashed away and cannot be overwritten, erased or read.

Petitioner Micron Ex-1021, 0500

When the host needs to read or write data from or to this protected area, it sends a specific command through a dedicated data frame (Fig. 17.13). This data frame is composed of all the relevant information for the operation under execution (e.g. address and data) and it also includes a 32-byte-long MAC and a 16-byte-long nonce.

	Stuff bytes	Key/ MAC	Data	Nonce	Write counter	Add	Block count	Result	Req/ resp	CRC
Length (bytes) →	196	32	256	16	4	2	2	2	2	2

Fig. 17.13. Data frame for accessing the Replay Protected Memory Block

The message authentication code (MAC) is calculated using the HMAC SHA-256, which is an HMAC applying a Secure Hash Algorithm (SHA) whose digest size is 256 bit long. The MAC is calculated over the sent data frame by means of the secret authentication key.

When the host needs to write data into the write protected memory block, it sends the data frame with the write relevant information (address, data...) together with the MAC. The eMMC card calculates the MAC over the received data frame and compares it with the MAC sent over by the host. If the MAC comparison is successful then the write request is considered authenticated and executed.

Conversely, when the host needs to read data from the card, it sends the data frame with the pertinent tokens (address...) and the nonce. Then the card responds with the MAC calculated over the data frame received (and hence also on the nonce) using the authentication key. The host compares the MAC sent by the card with the one calculated by itself. If they match, the host is sure that the response is coming from the card and not from a replay attack.

As a further security improvement, a write counter is also implemented. This counter keeps track of the total amount of the successfully authenticated data write requests made by the host. The value of this register will be automatically incremented by one at each successful programming access. Only an authenticated host can be privy to this number. At each read or write request the host must send this number to the card, too. Only upon a perfect matching between the write counter values of both the host and the card, the command is executed.

17.6 Execute in place

It is sometimes useful to run a software program directly from the Flash memory instead of RAM. This capability is called *Execute in Place (XIP)* and it is used mainly in cell phones. In a XIP application, typically both the code and the data are stored in the Flash memory.

The alternative usually employed is the so-called *Store and Download* (*SnD*). In such architecture, the code is stored in a non-volatile memory (e.g. a Flash) and

Petitioner Micron Ex-1021, 0501

it is downloaded into a RAM and then executed from the RAM. This approach has two main disadvantages. The first is high power consumption due to the refresh operation required by RAM memories. The second drawback is the time lost for the downloading of the content from the Flash to the RAM, which slows down all the system.

In the Store and Download methodology, if all the code in the non-volatile memory is downloaded to the RAM, the architecture is termed *Fully Shadowed*. In this case, the boot phase is very sluggish. Conversely, in a *Demand Paged* architecture, only a subset of the code is copied from the Flash to the RAM, not all of it. Obviously, the part copied is the one needed at that specific moment.

Sometimes, in the Demand Paged architecture, the code is also stored in the Flash in a compress form and it is uncompressed when copied into RAM. On the contrary, compression is used rarely in Fully Shadowed solutions. Evidently, compression is incompatible with an Execute in Place technique.

Running the code of an application directly from the Flash memory eludes all the shortcomings of the Store and Download methodology listed thus far. Even in this case, however, a small amount of RAM is still needed for variables and the stack. Only the code and read-only data (constants) are read from the Flash. The RAM size is thus considerably reduced. This implies both a smaller footprint and lower power consumption (the refresh takes place in a much smaller memory). Another contribution to lower power consumption is given by the fact that a Flash memory can be put in stand-by state when not in use without losing its content, while this is impossible for a RAM. Usually, the Flash is divided in two areas, one for the code and the other for the data.

Unfortunately, all that glitters is not gold. In order to implement the Execute in Place, the Flash must possess a few definite characteristics. The foremost attribute is random read timing. This holds true in particular for the system code. For this reason, NOR type memories are ideal for this application. NAND type memories, on the contrary, are not very apt to it because of their high latency and for this reason are mostly used with the Store and Download method. A high-speed buffer, though, can still render NAND memories liable to Execute in Place.

Furthermore, Flash memories are slower than RAM. However, if a microprocessor's cache is present it helps to counteract this effect.

17.7 Managed memory

Until few years ago, NAND Flash memories increased their capacity size by means of technology shrink, even more than Moore's law could foresee in 1965. Today, we are getting closer to lithography limits, supposed to be close to 20 nm. To maintain the current trend and double the number of bit stored per square millimeter, in an 18–24 months timeframe, a different solution must be adopted to pack more bits into the same cell. The choice is a 2-bit/cell device, so called MLC (Multi Level Cell), a 3-bit/cell device, so called TLC (Triple Level Cell), or more in general, an n-bit per cell.

Petitioner Micron Ex-1021, 0502

17.7.1 Multibit and shrink technology issues

The increase in the number of bits per cell involves a corresponding increase in the issue related to the cell size reduction.

A first aspect to be taken into account is that we need to put a greater number of threshold distributions almost in the same voltage range; as an example, in a 2-bit/cell device we need to put three distributions in a range of about 6 V (cells erased are on the left of 0 V), while in a 3-bit/cell seven distributions are inside the same range. In a MLC device, using all the available range between 0 and 6 V, we have 6 V/3 = 2 V for each distribution, while if the device is a TLC the space available for each distribution is less than one half of the MLC: in fact 6 V/7 = 0.86 V (see Fig. 17.14).

The distributions shown in Fig. 17.14 are ideal, because they are still distinct entities, each of them taking up a well defined space; we can therefore associate every cell, whatever its Vth value is, with the logic value of the information contained inside.

In reality, due to multiple phenomena that will be discussed below, distributions tend to enlarge and to move, producing an overlap of the distributions themselves (see Fig. 17.15).

These phenomena, as we have seen, are becoming more evident with the technology shrink and especially with the increase of the number of bits stored within each cell, as it involves an exponential increase in the number of distributions to be placed on the axis of the threshold voltages.

This uncertainty about the value contained into some cells involve NAND memory devices usage; a solution has to be put in place, aimed at eliminating, or at least minimizing, the causes responsible for the Vth change in respect of the value assigned during the programming process. The same approach should also ensure that, even with a Vth change, at least for a limited number of cells, it can be possible to retrieve the information stored during the program operation.

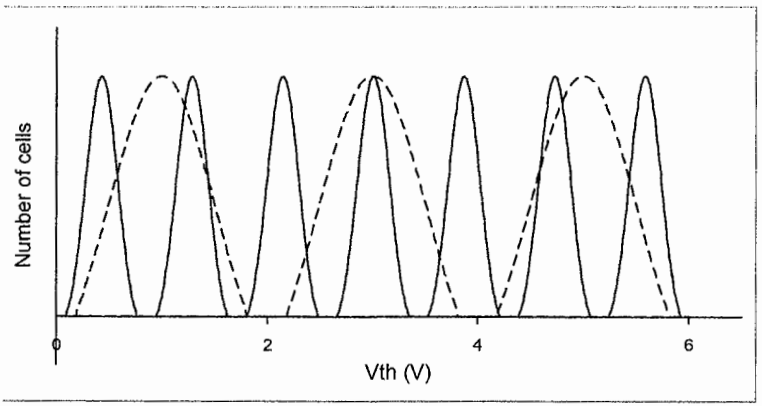


Fig. 17.14. Ideal distribution of positive Vth in 2 bits/cell case (dotted line) and 3 bits/cell (solid line)

Petitioner Micron Ex-1021, 0503

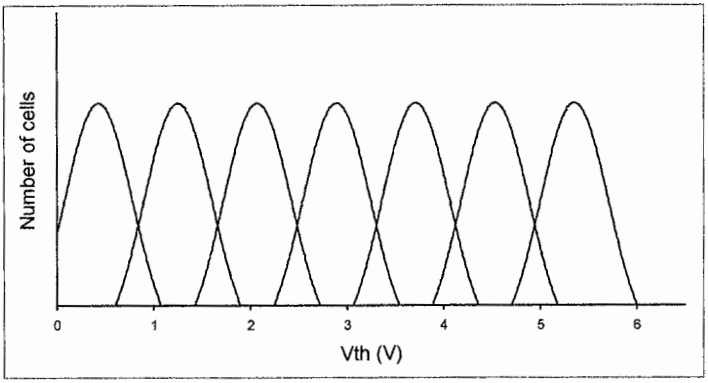


Fig. 17.15. Real distributions of positive Vth in 3 bits/cell case

Distributions shown in this chapter are only examples and they don't refer to a specific lithographic process.

Read disturb

The read disturb phenomenon (Chap. 4) occurs during reading operations and affects those cells belonging to all the word-lines (WL) of the block except the one containing the cells which must be read (see Fig. 17.16). To the unselected WLs is applied a Vpass voltage, higher than Vread, for a time in the range of tens of microseconds. In this phase, an electric field is applied to the cells and an unintended movement of charges to the floating gate occurs.

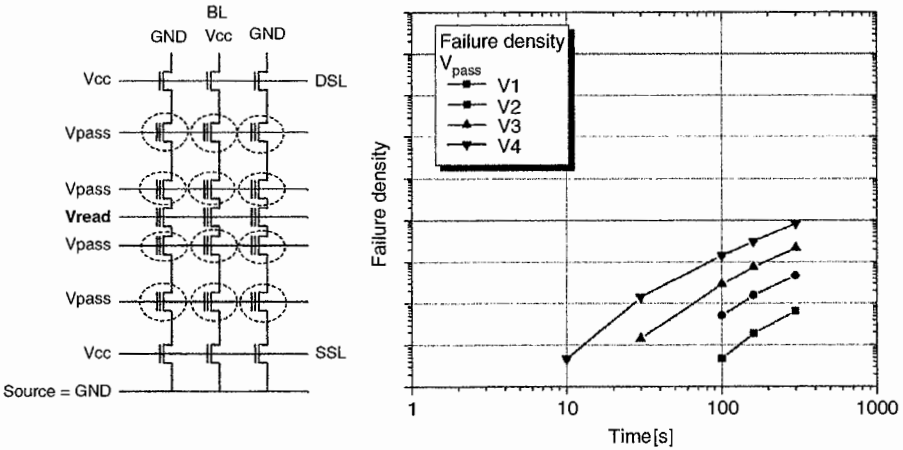


Fig. 17.16 Read disturb: involved cells (circled) and Failure Density diagram as a function of stress time and varying applied Vpass

Petitioner Micron Ex-1021, 0504

Every single cell undergoes this phenomenon whenever a read operation is made within the same block, on cells belonging to different word-lines. So, the maximum total stress time *MaxST(j)* affecting the word-line *j* in the block *k* is

$$MaxST(j) = Tread \cdot \sum_{i=0, i\neq j}^{NWL-1} Nread(i) \qquad (17.4)$$

- *NWL* is the number of WLs in the block;
- *Nread(i)* is the number of reads done in the block k on WL *i*;
- *Tread* is the duration of the read phase with WLs at Vpass.

The cells most affected by this phenomenon are the erased ones, which have negative threshold voltage; the phenomenon is additionally accentuated by the number of W/E cycles and by the increase of Vpass, as shown in the Failure Density graph of Fig. 17.16.

The final read noise effect is a broadening of the distributions towards higher Vth. This fact can be mitigated by modifying the process, the Vth window (i.e. acting on the margins between each distribution and its neighbors) and, finally, using an Error Correction Code (ECC, Chap. 14).

Pass disturb

As sketched in Fig. 17.17, the pass disturb phenomenon (Chap. 4) involves, during the program operation, the cells belonging to the same bit line of the cells to be programmed. In this case, the disturb is caused by the potential drop across the tunnel oxide, having a Vpass greater than that applied during the read operation (the channel is connected to GND).

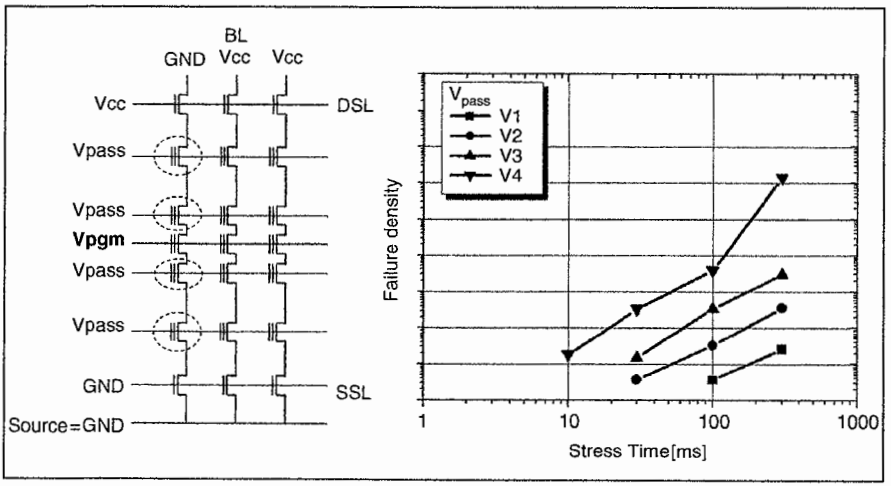


Fig. 17.17. Pass disturb: involved cells (circled) and Failure Density diagram as a function of stress time with different applied Vpass

Petitioner Micron Ex-1021, 0505

The maximum total stress time (*MaxST*) is:

$$MaxST = (NWL - 1) \times Npulse \times Tpulse \tag{17.5}$$

where *NWL* is the number of WLs in the block; *Npulse* is the number of programming pulses applied to those cells belonging to the same BL; *Tpulse* is the duration of each pulse.

Pass disturb phenomenon gets even worse with the number of W/E cycles. As previously seen for the read disturb, the most affected cells are the erased ones. Also in this case, recovery actions are process modification, Vth window adjustment and ECC.

Program disturb

Program disturb phenomenon (Chap. 4) happens during the program phase and it affects those cells belonging to the WL to be programmed (see Fig. 17.18).

Program disturbs effects are the known ones: Vth increase and distributions broadening. This disturb depends on the number of program pulses, on the maximum voltage applied to the control gate and on the data pattern. Furthermore, the disturb increases with the target Vth of the cell to be programmed.

Program disturb failure density trend is similar to the one already seen for the pass disturb; it increases with program/erase cycling count.

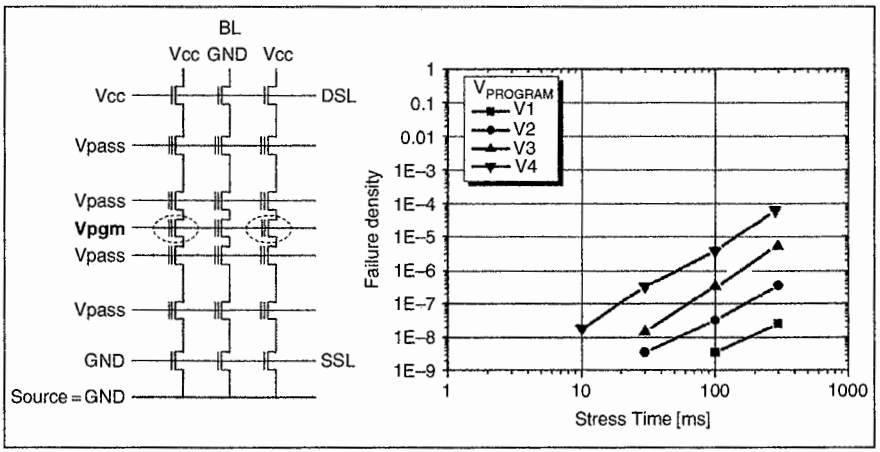


Fig. 17.18. Program disturb: involved cells are circumscribed and Failure Density

Trapping and Detrapping

During program and erase phases a trap generation can happen in the tunnel oxide between the floating gate and the substrate (Chap. 4). The root cause is the strong electric field applied to the oxide, having as effect some holes generation. Rate

Petitioner Micron Ex-1021, 0506

depends on the applied field and the total charge stocked into the oxide is functional to the square root of the cycles; average time occurring among further cycles and device function temperature also impact the phenomenon.

During program operation, electrons may be captured by the oxide traps, accumulating a negative charge and therefore modifying the cell Vth (see Fig. 17.19). This is the so called trapping phenomenon. Because of thermal heating, some electrons entrapped into the oxide can reach an activation energy high enough to overcome the barrier and move away from the oxide itself. With this charge release, a negative variation of the Vth happens. The total charge lost because of detrapping is a function of temperature, time elapsed among further program cycles and number of P/E cycles.

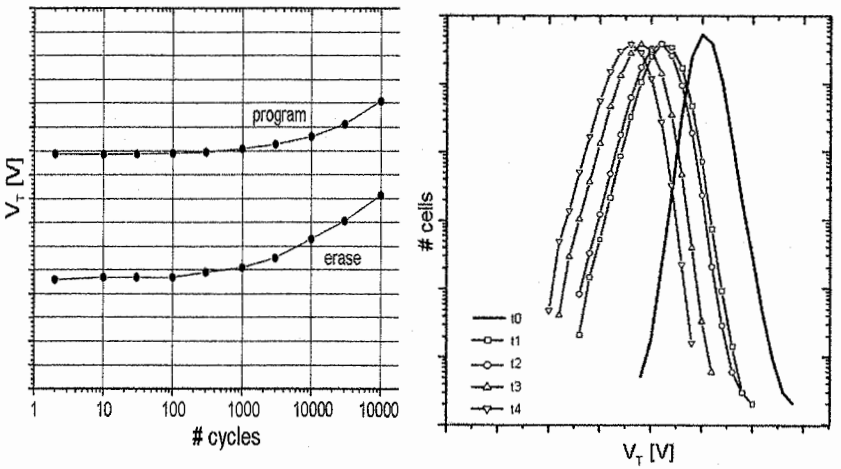


Fig. 17.19. Threshold voltage variation as a function of cycles in case of program and erase operations (left) and distribution variation due to detrapping (right)

Coupling

All disturbing phenomena previously described are linked to the technology shrink: the smaller the cell size is, the higher the negative effects on the cell themselves are.

In the past years and with previous technologies, threshold difference between two adjacent distributions corresponded to a difference in the number of electrons into the floating gate of thousands; nowadays in current devices, this difference is in the range of hundreds electrons and it is expected in the near future, at the foreseen technology limits, to be in the range of few tens. The other phenomenon reducing the difference among various distributions is the number of levels asked to coexist, as we saw exponentially linked to the number of bits to be included in a cell.

The effects of the capacitance coupling among the cell's floating gates (Chap. 4) contribute as well to the threshold modifications.

Petitioner Micron Ex-1021, 0507

To minimize the negative effects of the technology shrink and to increase the number of bits stored in each cell, the NAND technology needs to be modified and Error Correction Codes can help in recovering erroneous data. ECC solutions used so far are no longer enough; correction capability needs to be increased and a wider portion of data area is required for parity bits (Chap. 14). Moreover, an increase in decoding time, due to the complexity of the code, has to be taken into account.

Other solutions are therefore needed to grant NAND devices evolution in terms of memory capacity, at the same die size, without compromising too much performances. An option can be to couple the memory with a microcontroller.

17.7.2 Microcontroller solutions

The microcontroller can internally manage complex calculations in a most effective way with respect to the internal circuitry of the NAND Flash memory. The microcontroller can drive multiple Flash memories, connected on the same data bus with different Chip Enable or, in case it is required by the performances, on multiple buses. Additional advantage, on top of speed, is therefore the reduced amount of silicon needed.

Main task of the microcontroller is to elaborate the pattern to be programmed. As previously seen, disturbs are higher when the threshold value to be programmed is increasingly far from the erased level. A good solution to reduce disturbs is the encoding of the page aimed at minimizing the number of cells belonging to the farther distributions. In the simple SLC case, where cells own logic value 1 for erased and 0 for programmed, the number of zero to be programmed is computed, and if it is higher than the number of 1, a complement of data is applied to obtain a higher number of 1. This complement information needs to be saved so that, during reading operation, these data can be correctly retrieved and decoded following the opposite procedure. In the MLC and TLC cases the elaborations are more complex, depending on associations among distributions and multiple encodings.

To minimize the number of ambiguities in assigning different logic values, in MLC and TLC devices the microcontroller can elaborate the data to be programmed and encode the pages by prioritizing some of the farther distributions.

As already argued, those disturbs affecting a correct evaluation of the logic value inside the cells are related to the distance achievable among various distributions and to the number of block W/E cycles. Coupling capacitance between neighbor cells is responsible for the distribution modifications. Through algorithms and settings, the microcontroller is able to minimize these negative effects in order to not exceed the ECC usage.

The microcontroller is additionally able, using special commands, to send to the NAND device logic values to be used during programming, reading and erasing operations. During the device life, the amplitude of the program pulses and the number of W/E cycles affect the size and the number of the distributions. By using the method we are going to analyze, knowing the number of cycles per block, the pulse amplitude and initial voltage values can be properly set; Program Verify

Petitioner Micron Ex-1021, 0508

(PV) levels are modified accordingly. As far as the reading operation is concerned, it is possible to modify the value of the reference threshold voltage (Chap. 8) to Read Verify (RV) in order to adapt it to the average distribution shift due to cycling.

For the erase operation as well, key actions are linked to pulse voltage value and step height. All operations described above reduce the number of wrong bits and increase the device life, having the same amount of ECC.

As seen, with the technology advance and the increase numbers of bits stored in a single cell, the required ECC code must have an ever increasing correction capability. A first possible solution, achievable with a microcontroller, is the usage of two concatenated correction codes (Chap. 14): a inner block code, like BCH or Reed–Solomon, and a outer code like a convolutional one.

As shown in Fig. 17.20, having the same signal-to-noise ratio, representing the quantity of distributions overlap, there is a dramatic reduction in the Bit Error Rate (BER) using a concatenated approach.

To achieve these goals, according to the current and the future applications time to market, a strong calculation power is needed, only reachable by means of a microcontroller, able to get calculations at higher clock frequencies and, if needed, with a higher parallelism.

Another option, avoiding the usage of a concatenated code, is to use a single block code with a higher correction capability in respect of the ones used so far.

The basic idea, allowing the improvement in memory reliability, is to increase the number of bits in respect of those already used and stored in the spare area, aimed at correcting a higher number of errors. This implies that the spare area, which is limited, is no longer enough; in this specific case, as we will see later on, it needs to have specific characteristics. By using an external memory, it is instead possible to select the most suitable size.

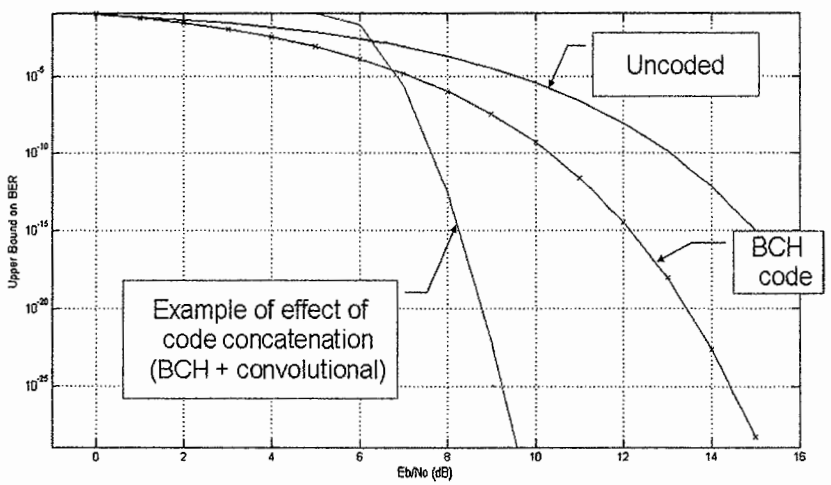


Fig. 17.20. Uncoded, BCH and code concatenation code comparison

Petitioner Micron Ex-1021, 0509

The memory, in which user data can be stored and the added memory necessary to implement the code, can be managed through the microcontroller, as shown in Fig. 17.21.

The microcontroller, apart from managing the two types of memories (they can be of different types and therefore having dedicated interfaces), is also requested to manage coding and decoding of the selected code.

The memory which stores the added information of the code, "Auxiliary Memory" in Fig. 17.21, can be used with different methods. A first approach is to evaluate the error correction capability needed for the total size of the memory we need to protect, the minimum codeword length (chunk), type and capability of the error correction code. Once the choices are made, it is possible to find the minimal additional memory compatible with the requests.

Using this method it is possible to link a chunk of the memory to be protected to a portion of the additional memory; this solution allows a fixed error correction capability for every chunk.

A second approach is based on a "dynamic" error correction capability (i.e. the error correction capability for a single chunk is increased when an error occurs); in this way it is possible to read correctly even in the case of further errors.

Specification to be fixed for implementation are, beyond of the size of the additional memory, the minimum and the maximum error correction capabilities (Nmin and Nmax).

At the beginning, every chunk of the primary memory is linked to a portion of the "Auxiliary Memory" needed to correct a minimum number of errors. Multiplying the total number of chunks by the parity bits needed to correct Nmin errors, we can calculate the required portion of the memory.

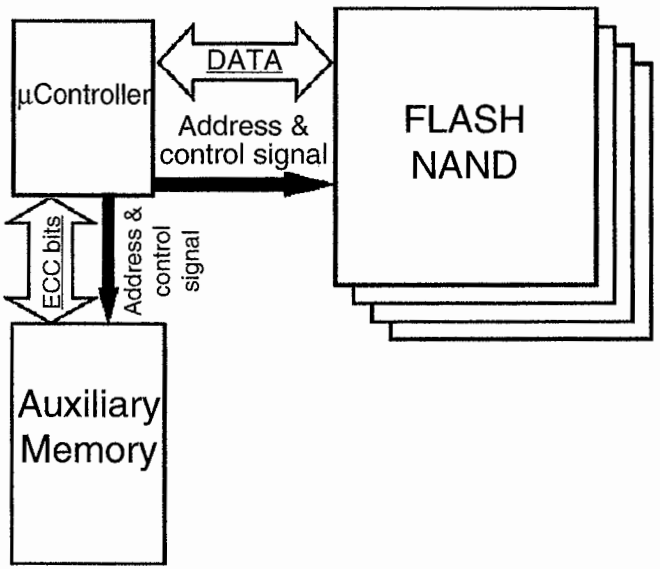


Fig. 17.21. Auxiliary memory scheme used to store code bits

Petitioner Micron Ex-1021, 0510

The remaining portion of the memory is then split in two parts (they can be equal or not), one of them is used to correct one or more additional error; knowing the size of this portion and the number of bits needed for the new error correction capability (at which must be subtract the number of bits of the previous error correction code, because they can be used for the new code), the number of chunks can be immediately calculated. Now, it can go ahead in the same manner of the previous step until the maximum value of the error correction capability (Nmax) is reached.

An aspect to consider is the fact that an error correction code greater than minimum does not cover all chunks of the primary memory, but only a portion; this fact forces to store which chunks have an error correction capability greater than minimum and at which address the additional bits of the code are stored.

Second method has the advantage, compared to the first, to reach a correction capability greater, when the "Auxiliary memory" is equal, but only a fraction of the total number of chunks can be corrected with this capability. Otherwise the first method has an error correction capability greater than minimum of the second method for all chunk of the NAND.

The choice of which method must be used can be done after an analysis of the defectiveness characteristics and their distribution within the primary memory's matrix.

Another aspect to increase the reliability of the Flash memory involves "wear leveling" algorithms (Sect. 17.4). Wear leveling, in general, comprise a function that allows a uniform usage of the blocks of a memory device.

As depicted in Fig. 17.22, NAND memory is composed by a plurality of blocks. Auxiliary Memory for this example embodiment contains a plurality of storage locations for storing wear-leveling information. The blocks of wear-leveling information are associated with the groups of blocks of NAND memory. Wear-leveling information means the number of times a block of NAND memory has been erased. If a block is erased, that block's age counter will be incremented. The increment may be made by writing a bit of information to the memory. Note that with Auxiliary memory, there is no need to erase a block before writing it. By maintaining the wear-leveling information in a proper external memory, the wear-leveling information is not subject to the long-term reliability issues of the NAND memory, and more space may be available on the NAND itself. Furthermore, the use of a different memory to store wear-leveling information related to the NAND memory avoids the need of storing the wear-leveling information in the NAND, which would result in a loss of primary storage locations and would result in an increase in overhead. Using the techniques described herein, the mean age of the physical NAND blocks may be maintained approximately constant.

Error correction information and wear-leveling information related to a NAND Flash memory may both be stored on a different external memory. For this embodiment, wear-leveling operations may take advantage of the information on failed bits to increase a block's age, thereby extending the useful life of the NAND Flash memory.

Petitioner Micron Ex-1021, 0511

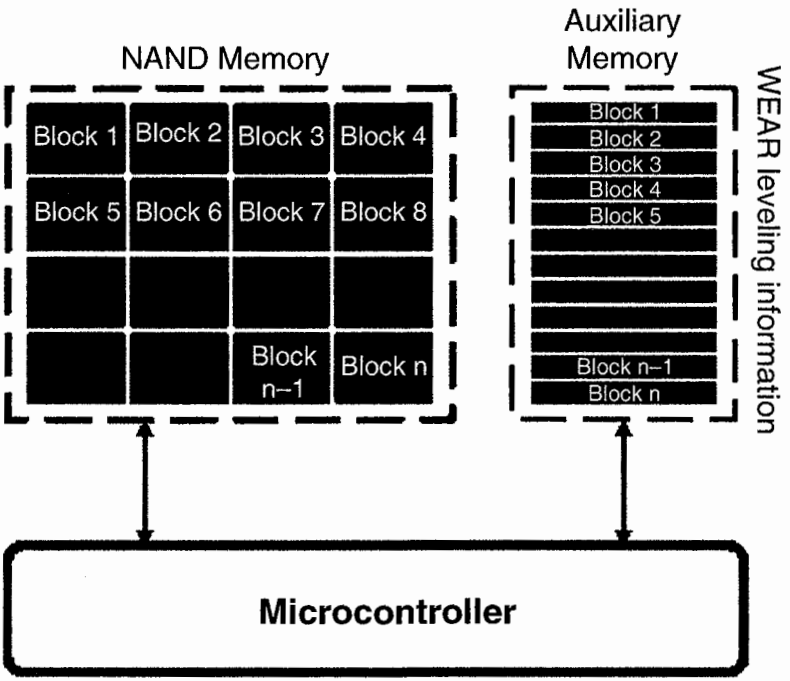


Fig. 17.22. Auxiliary memory for wear-leveling

17.7.3 Flexible solution

In non volatile memory environment, NOR and NAND addressed so far different application segments, where different performances and memory capacity were requested.

Random Read performances of NOR Flash are particularly useful to implement code Execute In Place (XiP): this kind of memory is used in a system as execution memory of code or as boot memory.

NAND write throughputs and high densities are the best solution for Mass Storage applications but the range of applicability has progressively increased covering also code storage: a NAND device is used to guarantee code (operating system code, software applications) persistence that is usually downloaded in an execution memory (typically a RAM) more often in a multichip solution.

As we saw before, NAND technology is classified mainly into two sub-types: Single Level Cell and Multi Level Cell. Table 17.2 summarizes some differences between the two classes related to the same cell lithography.

The basic differentiation points highlight that SLC is more suitable to store and download code thanks to the higher read performances and endurance, while MLC NAND is an optimized solution for all applications requiring high densities of storage.

Petitioner Micron Ex-1021, 0512

Table 17.2. NAND Flash: SLC versus MLC

	SLC	MLC
Sequential read access time	25 ns	25 ns
Random read access time	25 µs	60 µs
Program time	200 µs	800 µs
Erase time	2 ms	2.5 ms
Power supply – Vdd	1.8/3.0 V	3.0 V
Program/Erase cycles	100 k	10 k
Data retention	10 years	10 years

Many market applications require both SLC features to reliably store system code and data and MLC features, to save high density data while requiring lower reliability

A new solution, we'll call it flexible, is realized thanks to the microcontroller and consists on the partitioning of the single NAND cell in two sub sections, the first of them to be used as SLC, the other one as MLC (see Fig. 17.23).

The partitioning allows the microcontroller to use different algorithms and settings in memory usage depending on the features of the selected portion. The partition is dynamic; the space assigned to the code can be modified by the microcontroller. In the SLC partition program (PV) and read (RV) references are more relaxed and the pulses can have a wider size due to a single distribution in the positive side of the threshold voltage values. This minor precision in Vth values allows speeding up program and read operations, getting closer to native SLC memories. Additionally, in SLC partition, there is not the need of a high ECC capability, affecting as well performances and data area to store parity bits.

The same approach can be applied to 3 or more bits per cell devices, where a portion is reserved for a SLC-mode usage to get better performances in terms of throughput and reliability.

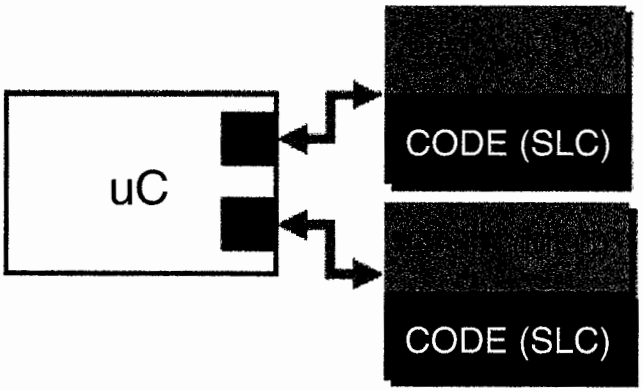


Fig. 17.23. Flexible solution scheme

Petitioner Micron Ex-1021, 0513

References

AQ1

T.G. Lenihan, E. Jan Vardaman, Worldwide Perspectives on SiP Markets: Technology Trends and Challenges, 7th International Conference on Electronics Packaging Technology, IEEE.

R.R. Tummala, V.K. Madisetti, System on chip or system on package?, Design & Test of Computers, IEEE.

L. Boettcher, D. Manessis, A. Ostmann, S. Karaszkiewicz, H. Reichl, Embedding of Chips for System in Package realization – Technology and Applications, Microsystems, Packaging, Assembly & Circuits Technology Conference, 2008. IMPACT 2008. 3rd International.

V.J. Falgiano, W.R. Newberry, Entry level MCM/MCP trade-offs and considerations, Electronic Components and Technology Conference, 1994. Proceedings., 44th.

Wei Koh, Digital Memory Card Market and Technology, IEEE.

JESD84-A44, Embedded MultiMediaCard(e•MMC) e•MMC/Card Product Standard, High Capacity, including Reliable Write, Boot, Sleep Modes, Dual Data Rate, Multiple Partitions Supports and Security Enhancement, JEDEC.

Todd K. Moon, Error Correction Coding, Mathematical Methods and Algorithms, Wiley.

SD Specifications, Physical Layer, Version 2.00, SD Card Association.

D. Eastlake, T. Hansen, "US Secure Hash Algorithms (SHA and HMAC-SHA)", RFC 4634, July 2006.

Standard DoD 5220.22-M, US DoD 5220.22-M (ECE).

NIST Special Publication 800-88, Guidelines for Media Sanitization Recommendations of the National Institute of Standards and Technology.

AN1820, How to Use the FTL and HAL Software Modules to Manage Data in Single Level Cell NAND Flash Memories, Application Note, STMicroelectronics.

AN1819, Bad Block Management in Single Level Cell NAND Flash Memories, Application Note, STMicroelectronics.

AN1821, Garbage Collection in Single Level Cell NAND Flash memories, Application Note, STMicroelectronics.

AN1822, Wear Leveling in Single Level Cell NAND Flash Memories, Application Note, STMicroelectronics.

M.L. Chiang, Paul C. H. Lee, R.C. Chang, "Using Data Clustering to Improve Cleaning Performance for Flash Memory", Software Practice & Experience.

Li-Pin Chang and Tei-Wei Kuo; An Adaptive Striping Architecture for Flash Memory Storage Systems of Embedded Systems; Proceedings of the Eighth IEEE Real-Time and Embedded Technology and Applications Symposium, IEEE.

Mei-Ling Chiang Chen-Lon Cheng Chun-Hung Wu, A New FTL-based Flash Memory Management Scheme with Fast Cleaning Mechanism, The 2008 International Conference on Embedded Software and Systems, IEEE.

L.P. Chang, T.W. Kuo, "A Real-time Garbage Collection Mechanism for Flash Memory Storage System in Embedded Systems," The 8th International Conference on Real-Time Computing Systems and Applications, IEEE.

M.L. Chiang, R.C. Chang, Cleaning Policies in Mobile Computers Using Flash Memory, Journal of Systems and Software, IEEE.

Petitioner Micron Ex-1021, 0514

Yongsoo Joo; Yongseok Choi; Chanik Park; Sung Woo Chung; Eui-Young Chung; Naehyuck Chang; Demand paging for OneNAND¿ Flash eXecute-in-place; Hardware/software codesign and system synthesis, 2006. CODES+ISSS '06. Proceedings of the 4th international conference, IEEE.

C. Park, Jaeyu Seo; Sunghwan Bae; Hyojun Kim; Shinhan Kim; Bumsoo Kim; A low-cost memory architecture with NAND XIP for mobile embedded systems; Hardware/ Software Codesign and System Synthesis, IEEE.

T. Benavides, J. Treon, J. Hulbert, W. Chang, T. Benavides, J. Treon, J. Hulbert, W. Chang, The Implementation of a Hybrid-Execute-In-Place Architecture to Reduce the Embedded System Memory Footprint and Minimize Boot Time; Information Reuse and Integration, 2007; IEEE.

K. Kim, J. Choi, "Future outlook of NAND Flash technology for 40nm node and beyond", Proc. NVSMW,2006. pp. 9–11.

J.D. Lee et al.,"A New Programming Disturbance Phenomenon in NAND Flash Memory by Source/Drain Hot-Electrons Generated by GIDL Current," NVSMW, pp. 31–33, 2006.

H. Miki, T. Osabe, N. Tega, A. Kotabe, H. Kurata, K. Tokami, Y. Ikeda, S. Kamohara, R. Yamada, "Quantitative analysis of random telegraph signals as fluctuations of threshold voltages in scaled Flash memory cells", Proc. 45th IRPS, 2007. pp. 29–35.

C.M. Compagnoni, R. Gusmeroli, A.S. Spinelli, A.L. Lacaita, M. Bonanomi, A. Visconti, "Statistical investigation of random telegraph noise Id instabilities in Flash cells at different initial trap-filling conditions", Proc. 45th IRPS, 2007. pp. 161–166.

Kinam Kim, Jung Hyuk Choi, Jungdal Choi, Hong-Sik Jeong, "The Future Prospect of Nonvolatile Memory", 2005 IEEE VLSI-TSA International Symposium, pp. 88–94, 2005.

K. Kim, J. Choi, "Future outlook of NAND Flash technology for 40nm node and beyond", 21st Non-Volatile Semiconductor Memory Workshop (NVSMW), 2006, pp. 34–35.

N.N. Mielke et al., "Bit Error Rate in NAND Flash Memories", IEEE International Reliability Physics Symposium, pp. 9–19. 2008.

S. Aritome, "Advanced Flash memory technology and trends for file storage application," in IEDM Technical Digest, 2000, pp. 763–766.

S. Li and T. Zhang, "Approaching the Information Theoretical Bound of Multi-Level NAND Flash Memory Storage Efficiency", International Memory Workshop, May 2009.

S. Li, T. Zhang, "Improving Multi-Level NAND Flash Memory Storage Reliability Using Concatenated TCM-BCH Coding", ACM Great Lakes Symposium on VLSI, May 2009.

S. Corno, M. Scognamiglio, "Memory: On the Origin of Species, from non volatile concept to the Broceliande", pending for publication.

D. Caraccio, F. Tiziani, "Managed Memory Solution: a path to move intelligence into memory systems", Elektronik Praxis, June 2009.

Petitioner Micron Ex-1021, 0515

18 Low power 3D-integrated SSD

*K. Takeuchi**

18.1 Introduction

With highly scaled 40 or 30 nm technologies, the memory capacity increases to as much as 32 Gbit as shown in Fig. 18.1. By using gigabit-capacity NAND Flash memories, SSD, Solid-State Drive that uses NAND as a mass storage of personal computers and enterprise servers is expected as a next killer application of NAND Flash memories.

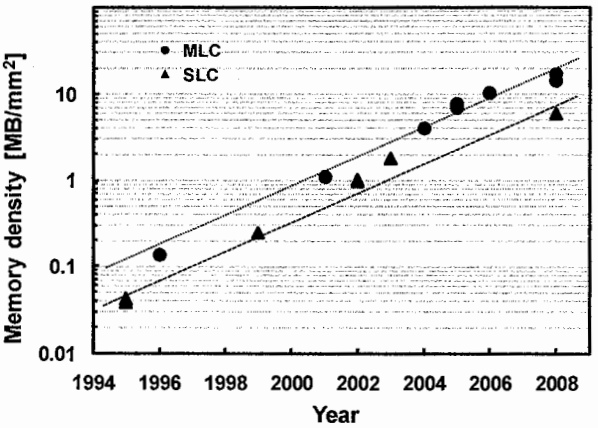


Fig. 18.1. Memory density trend of a NAND Flash memory

The hardware architecture of SSD is shown in Fig. 18.2. SSD is composed of NAND Flash memories, DRAMs and a NAND controller. To realize a low power high speed SSD, the overall performance of the NAND Flash memory and the NAND controller should be maximized by co-designing NAND Flash memory and NAND controller circuits [1–3]. Furthermore, an intelligent 3D-integration of various circuits in SSD such as the NAND controller, the NAND Flash memory and voltage generators are essential [4].

* Department of Electrical Engineering and Information Systems, Graduate School of Engineering, University of Tokyo, takeuchi@lsi.t.u-tokyo.ac.jp

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_18,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0516

18.2 Analysis of SSD performance

As shown in Fig. 18.2, SSD contains more than 8 NAND chips. For example, in Fig. 18.2, each NAND Flash memory is assumed to have 16 Gbit or 2 GByte capacity. One SSD is composed of 64 NAND chips and the total capacity is 128 GBytes, which is most suitable for the lap top computers. In Fig. 18.2, 4-channel configuration is adopted. The NAND controller can operate four channels independently.

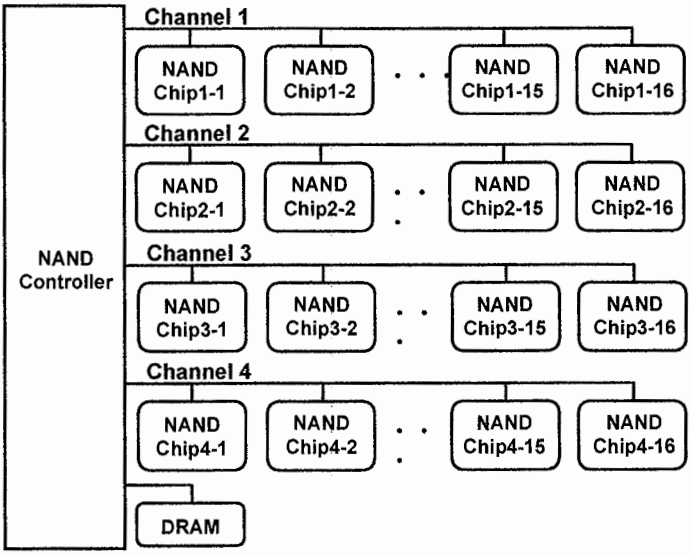


Fig. 18.2. Hardware architecture of SSD

The NAND controller can issue a write command to four channels, writing four NAND chips at the same time and improving the system-level write performance. In a channel, 16 NAND Flash memory chips are connected where I/O signals and control signals such as the WE (Write Enable)-signal, the RE (Read Enable)-signal, the CLE (Command Latch Enable)-signal, and the ALE (Address Latch Enable)-signal are shared to save area for the bonding and the packaging. To select one NAND chip in a channel, different CE (Chip Enable)-signals and R/B (Ready/Busy)-signals are assigned to the NAND chips in a channel.

Figure 18.3 shows the software architecture of SSD. The NAND controller is composed of the host interface, the Flash translation layer (FTL) and the NAND interface. FTL is the central part of the NAND controller and perform a bad block management, a logical-physical address translation, a wear-leveling, ECC and an interleaving. The intelligent write algorithm described in Sects. 18.5 and 18.6 is also controlled with FTL.

Petitioner Micron Ex-1021, 0517

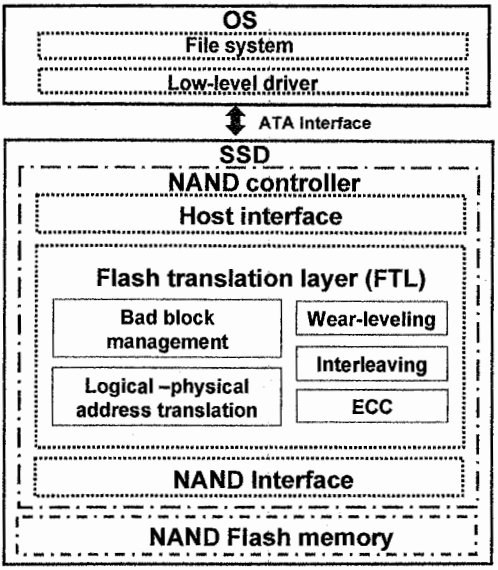


Fig. 18.3. Software architecture of SSD

In a NAND Flash memory, the programming is slower than the read by one order of magnitude [1]. The typical random read time of a 2 bit/cell NAND Flash memory is 50 μs while the typical random program time is 800 μs. Considering the read/write access time of HDD is a few milliseconds, the key design challenge of SSD is to improve the write performance. To accelerate the write speed of SSD, an interleaving is proposed where multiple NAND chips in SSD are programmed at the same time [5]. The write speed of SSD with an interleaving is expressed as follows:

$$Performance\_SSD = N \cdot Performance\_NAND \tag{18.1}$$

N is the number of NAND Flash memories operated simultaneously. *Performace_NAND* is the program speed of one NAND flash memory. As a memory cell is scaled down or more bits are stored in a memory cell such as 3 or 4 bit/cell, a more precise control of the memory cell V_{TH} is required and accelerating the NAND speed, *Performace_NAND*, becomes difficult. Actually, the program time of the 56 nm 3 bits/cell is 1.2 ms [6] and that of the 70 nm 4 bit/cell is 6 ms [7] which are much longer than that of the 2 bits/cell. Thus, the best strategy to improve the SSD speed is to increase the number of NAND chips in parallel, N. However, if N is maximized and all NAND chips in SSD, for example 64 NAND chips, are programmed simultaneously, 64 times as much current flows and the current consumption increases to an unacceptable high value, two Amperes. Therefore, in an actual SSD operation, the maximum number of N is restricted by the current consumption constraint.

Petitioner Micron Ex-1021, 0518

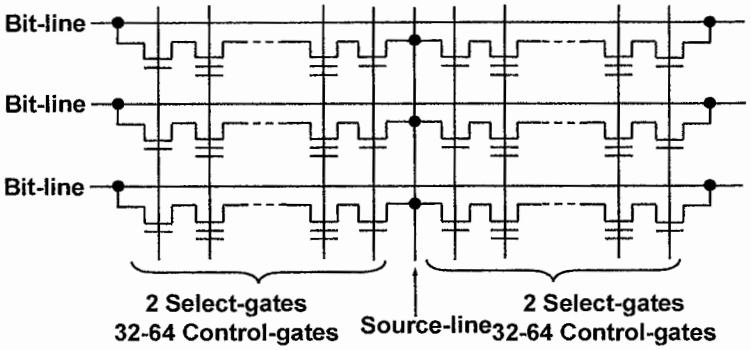


Fig. 18.4. Schematic diagram of a NAND Flash memory cell array

As the NAND cell is scaled down, the bit-line capacitance drastically increases [8] and consequently the current consumption increases. Figure 18.4 shows the schematic diagram of the NAND Flash memory cell array. The bit-line is arranged with the minimum feature size. Figure 18.5 describes the cross sectional view of the bit-line and the memory cell. As the design rule is decreased, the space between the bit-lines is also decreased. On the other hand, the height of the bit-line is not decreased to keep the low resistance of the bit-line. As a result, the inter bit-line capacitance, $C_{BL\text{-}BL}$ shown in Fig. 18.5a increases and the bit-line capacitance also increases as shown in Fig. 18.6. The total bit-line capacitance in a chip exceeds 200 nF and the current consumption to precharge the huge bit-line capacitance increases as shown in Fig. 18.7.

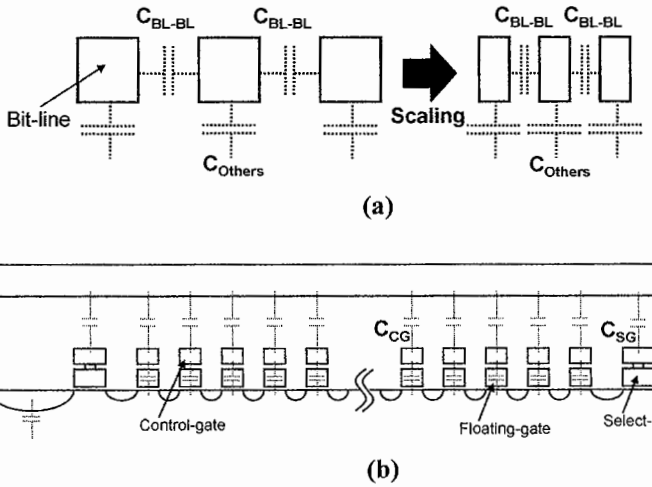


Fig. 18.5. Cross sectional view of (a) a bit-line and (b) a NAND Flash memory cell

Petitioner Micron Ex-1021, 0519

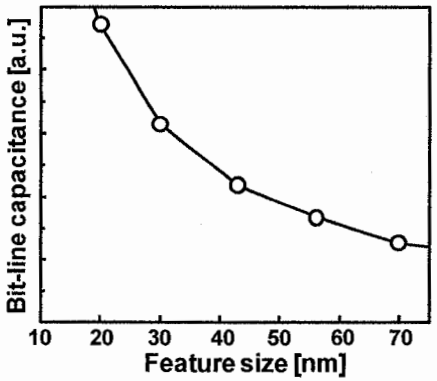


Fig. 18.6. Bit-line capacitance trend of a NAND Flash memory

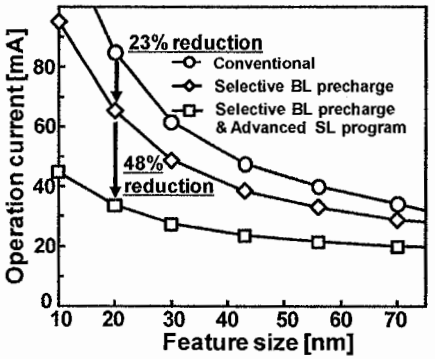


Fig. 18.7. Operation current trend of a NAND Flash memory

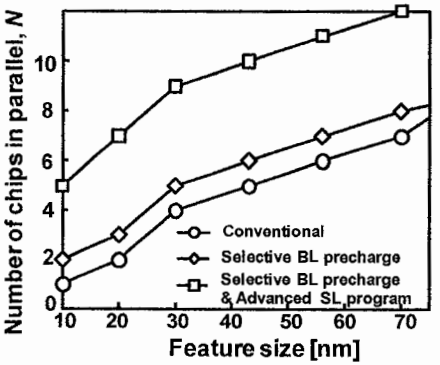


Fig. 18.8. Trend of the number of NAND chips operated in parallel, N

Petitioner Micron Ex-1021, 0520

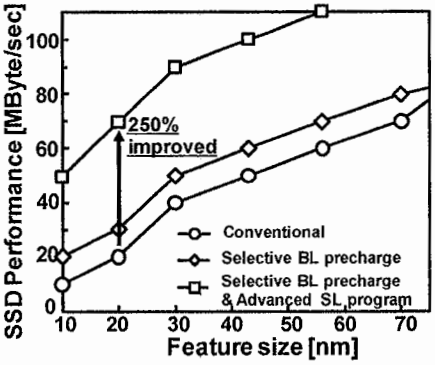


Fig. 18.9. System-level performance trend of SSD

Since the current consumption increases for sub-30nm generation, the number of chips in parallel, *N* should be smaller as shown in Fig. 18.8 and the SSD speed drastically degrades as shown in Fig. 18.9. To overcome this power consumption problem, three new circuit technologies are described in the following sections.

18.3 Selective bit-line precharge scheme

A bit-by-bit program algorithm of the NAND Flash memory [9] is shown in Fig. 18.10a.

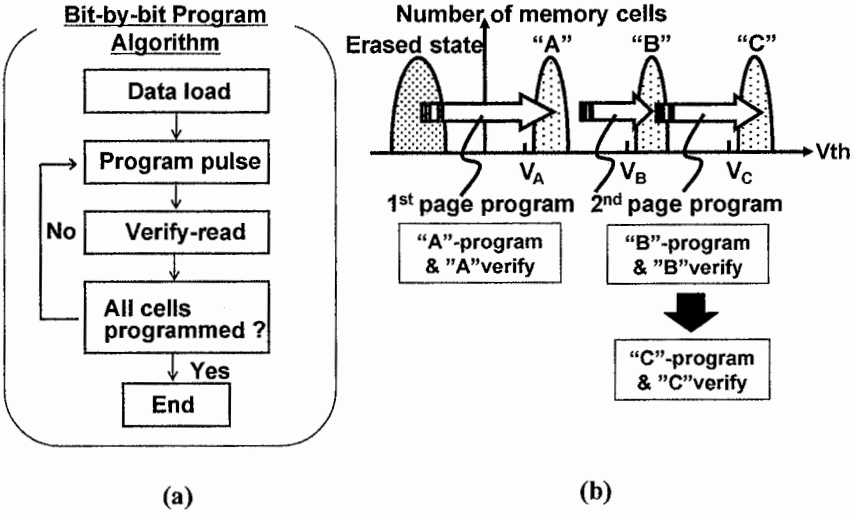


Fig. 18.10. (a) Program algorithm of a NAND Flash memory, (b) 1st page and 2nd page program of a multi-level cell

Petitioner Micron Ex-1021, 0521

First, the program data is input to the page buffers. Next, the program pulse is applied to the memory cells. Then, the verify-read to check if the memory cells are successfully programmed is performed. The re-program data in the page buffers are modified so that the re-program pulse is applied only to memory cells which are insufficiently programmed. The re-program pulse and the verify-read are repeated until all memory cells are successfully programmed.

In case of the multi-level cell, two bits in a memory cell are assigned to the 1st and 2nd pages and the two bits are programmed at different operations [10]. In the 1st page program, the memory cell is programmed from the erased state to the "A"-state shown in Fig. 18.10b. In the 2nd page program, the memory cell is programmed to the "B"- or "C"-state. The selective bit-line precharge scheme decreases the current consumption during the verify-read.

Figure 18.11 compares the conventional verify-read and the selective bit-line precharge scheme. In the conventional verify-read, all bit-lines are precharged irrespective of the program data and a lot of current flows. In the selective bit-line precharge scheme, bit-lines are selectively precharged based on the program data in the page buffer. By eliminating the unnecessary bit-line precharge, a low current operation is achieved.

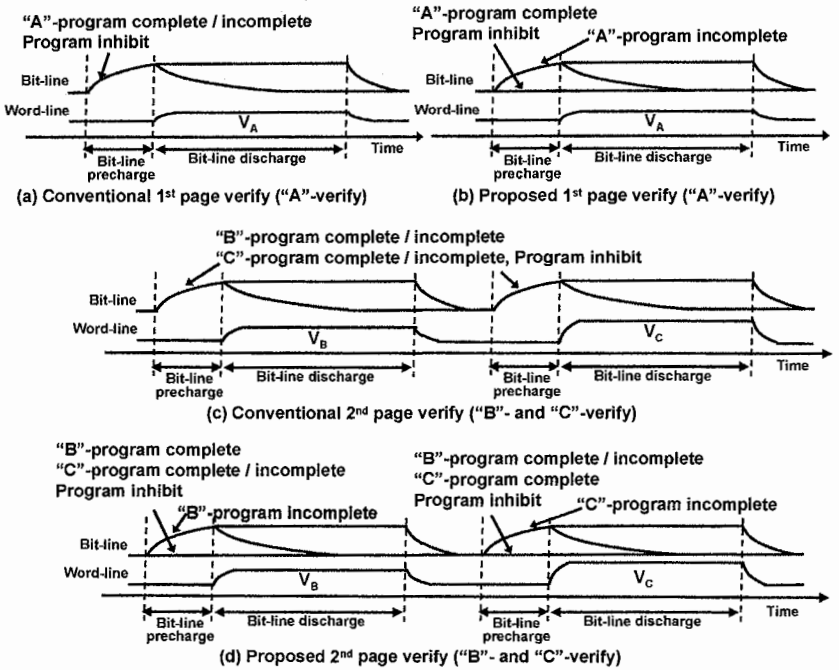


Fig. 18.11. Conventional and selective bit-line precharge scheme: (a) Conventional 1st page verify ("A"-verify); (b) Proposed 1st page verify ("A"-verify); (c) Conventional 2nd page verify ("B"- and "C"-verify); (d) Proposed 2nd page verify ("B"- and "C"-verify)

Petitioner Micron Ex-1021, 0522

This novel operation is realized by adding five transistors shown in Fig. 18.12 to the conventional page buffer [11]. The die size penalty is less than 1% of the chip size.

In the 1st page verify-read, the data in the page buffer is the program inhibit, the "A"-program complete or the "A"-program incomplete. The program data stored in a page buffer for the "A"-program incomplete is updated so that the program pulse is applied only to the memory cells which are insufficiently "A"-programmed. In the conventional "A"-verify read, all bit-lines are precharged to 1 V as shown in Fig. 18.11a. On the other hand, in the proposed "A"-verify read, in case of the "A"-program complete and the program inhibit, the data in the page buffer does not change and thus the bit-line precharge can be removed. The bias condition of the proposed page buffer is summarized in Table 18.1. By activating PRE1- and PRE2-signals in Fig. 18.12, only bit-lines of the "A"-program incomplete are precharged to 1 V as shown in Fig. 18.11b. Then, the control gate is raised to V_A in Fig. 18.10b and the bit-line is discharged.

Table 18.1. Bias condition of the selective bit-line precharge

	PRE1	PRE2
"A"-verify	"High"	"High"
"B"-verify	"High"	"Low"
"C"-verify	"Low"	"High"

In the 2nd page verify-read, "B"- and "C"-verify read are sequentially performed. In the conventional "B"-verify read, all bit-lines are precharged to 1 V, irrespective of the program data as shown in Fig. 18.11c. In the proposed "B"-verify read shown in Fig. 18.11d, the PRE1-signal is turned-on and only bit-lines of the "B"-program incomplete are precharged.

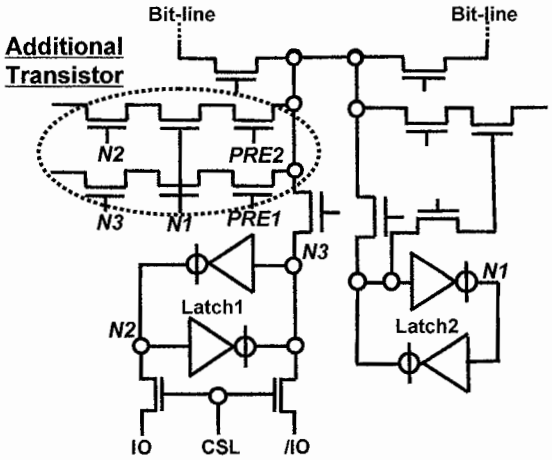


Fig. 18.12. Intelligent page buffer

Petitioner Micron Ex-1021, 0523

The bit-lines of the "C"-program, the "B"-program complete and the program inhibit are not precharged to save current. Similarly, in the proposed "C"-verify read shown in Fig. 18.11d, the PRE2-signal is activated and only bit-lines of the "C"-program incomplete are precharged.

By removing an unnecessary bit-line precharge, the current consumption decreases by 23% for sub-30 nm NAND as shown in Fig. 18.7.

18.4 Advanced source-line program

The advanced source-line program reduces the current during the program pulse. Figure 18.13 compares the conventional program [12] and the source-line program [13]. In the conventional program shown in Fig. 18.13a, to realize a program inhibit operation a higher program inhibit voltage, 2.5 V, is applied from the high capacitance bit-line to the memory cell, consuming a lot of current. Contrarily, in the source-line program shown in Fig. 18.13b, the current consumption is reduced by applying a higher voltage, 2.5 V, from the low capacitance source-line.

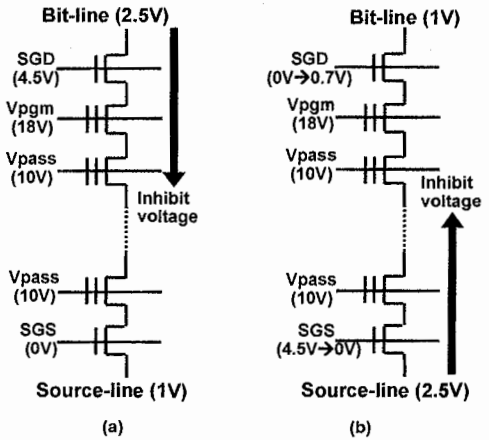


Fig. 18.13. (a) Conventional program [12], (b) Source-line program [13]

By using Fig. 18.5, the capacitance of the bit-line and the source-line is compared. The bit-line capacitance, $C_{Bit\text{-}line}$, and the source-line capacitance, $C_{Source\text{-}line}$, are expressed as follows:

$$C_{Bit-line} = C_{BL-BL} + C_{junction} \tag{18.2}$$

$$C_{Source-line} = C_{Source-line,wire} + C_{junction} \tag{18.3}$$

Petitioner Micron Ex-1021, 0524

where $C_{BL\text{-}BL}$, $C_{Source\text{-}line,wire}$ and $C_{junction}$ are the inter bit-line wiring capacitance, the source-line wiring capacitance and the junction capacitance, respectively as shown in Fig. 18.5. The junction capacitance of the bit-line is the same as that of the source-line.

In the NAND Flash memory, the inter bit-line wiring capacitance, $C_{BL\text{-}BL}$, is much larger than the source-line wiring capacitance, $C_{Source\text{-}line,wire}$, or the junction capacitance, $C_{junction}$. As shown in Fig. 18.4, the bit-lines cover all memory cells. On the other hand, one source-line is arranged every 64 or 128 control gates and four select gates. Thus, the source-line wiring capacitance, $C_{Source\text{-}line,wire}$, is much smaller than the inter bit-line capacitance, $C_{BL\text{-}BL}$. Similarly, the junction capacitance, $C_{junction}$, is much smaller than the inter bit-line wiring capacitance, $C_{BL\text{-}BL}$, because one contact is also shared by 64 or 128 memory cells and four select transistors. As a result, the source-line capacitance, $C_{Source\text{-}line}$, is as small as one tenth of the bit-line capacitance, $C_{Bit\text{-}line}$.

The source-line program was demonstrated with a 0.25 μm NAND Flash memory [13]. Yet, in the sub-50 nm gigabit-capacity NAND Flash memory since the number of source-lines increases, the total source-line capacitance in a chip exceeds 20 nF and the current consumption to charge a source-line capacitance becomes significant.

To solve this problem, the advanced source-line program is introduced [1]. In this scheme, the source-line is divided to reduce the load capacitance during the program pulse. As shown in Fig. 18.14, one memory cell array is divided to 16 sub-arrays. The source-line has a hierarchy structure where a local source-line in a sub-array connects to the global source-line through the source-line decoders.

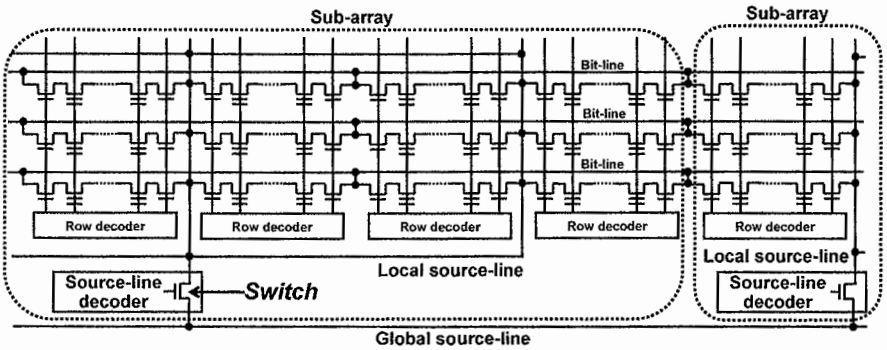


Fig. 18.14. Memory cell array architecture of the advanced source-line program

Note that only the source-line is divided and that the active region, bit-lines, control gates and select gates are not divided. As the hierarchical source-line structure is realized by changing the layout of the metals and contacts in the memory cell array, there is no cell area overhead with this new architecture. The additional source-line decoders increase the area by less than 1% of the chip size.

Petitioner Micron Ex-1021, 0525

During the program pulse, the source-line decoder of the selected sub-array is turned-on and the source-line decoders of the unselected sub-arrays are turned off as shown in Table 18.2.

Table 18.2. Operation principle of the switch in the source-line decoder in the advanced source-line program

	Selected sub-array	Unselected sub-array
Read	ON	ON
Program	ON	OFF
Erase	ON	ON

As a result, only the local source-line of the selected sub-array is biased to 2.5 V and the remaining local source-lines of unselected sub-arrays are not charged. Since the load capacitance of the source-line is reduced by 90%, the total current decreases by 48% as shown in Fig. 18.7.

On the other hand, during the read and the verify-read operations, all source-line decoders are turned on and thus all local source-lines are connected to the global source-line so that the source-line resistance is minimized. Consequently, the circuit noise caused by the source-line bounce [14] is suppressed.

By using the selective bit-line precharge scheme and the advanced source-line program, the current consumption reduces by 60% for sub-30 nm generation SSD. Assuming the same current budget of 250 mA, 2.5 times as many NAND chips operate simultaneously as shown in Fig. 18.8 and the SSD speed improves by 150% as shown in Fig. 18.9.

18.5 Intelligent interleaving

Figure 18.15 illustrates the current waveform of the NAND Flash memory. A current peak appears during the bit-line precharge and the charge pump ramp-up [8]. In the interleaving operation, if the current peak of two or more NAND chips occurs at the same time, huge current flows in SSD and the power supply drops by more than 0.3 V. To avoid this power supply noise and realize a both highly-reliable and high-speed program, an intelligent interleaving is introduced.

In this scheme, the PD (Power Detect)-signal is added. The PD-signal is connected with wired-or configuration among NAND Flash memories and the NAND controller as shown in Fig. 18.16. In Fig. 18.16, the NAND chips can belong to the same channel or the different channels in Fig. 18.2. If one of the NAND chips starts a bit-line precharge or a charge pump ramp-up that causes a current peak, the NAND chip pulls down the PD-signal. When PD is low, the NAND controller does not issue a write command to avoid the power supply noise.

Petitioner Micron Ex-1021, 0526

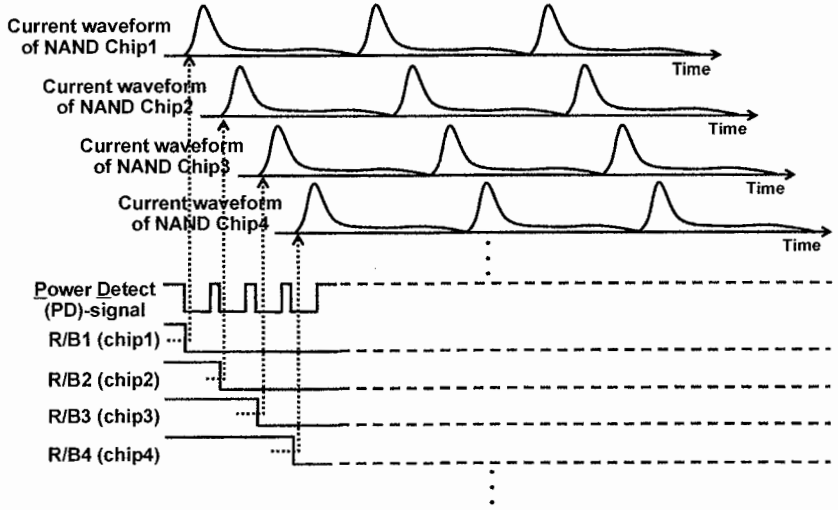


Fig. 18.15. Current waveform and operation principle of the intelligent interleaving

To monitor the status of each NAND chip, the R/B (Ready/Busy)-signal is connected between the NAND controller and each NAND Flash memory chip. The R/B-signal is low if the NAND chip operates a read, program or erase and therefore is in the busy state. When both the PD- and the R/B-signals are high, Program_Enable-signal shown in Fig. 18.17 in the controller becomes low. In that case, there is no current peak and the NAND chip is ready. Then, the NAND controller issues a write command to the NAND chip and the program starts.

By using the intelligent interleaving, multiple NAND chips are programmed at the same time without causing a power supply noise as shown in Fig. 18.15. Therefore, a highly reliable and high speed operation of SSD is achieved.

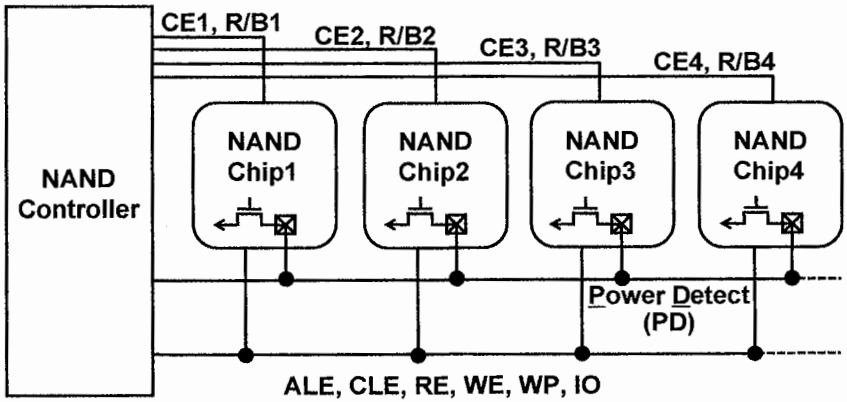


Fig. 18.16. Block diagram of the NAND Flash memory and the NAND controller with the intelligent interleaving

Petitioner Micron Ex-1021, 0527

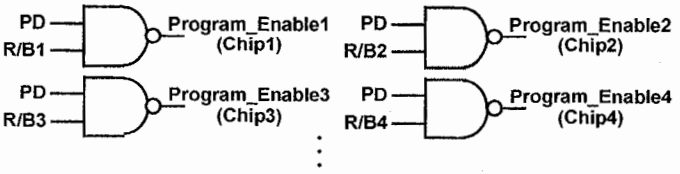


Fig. 18.17. Intelligent interleaving control circuit in the NAND controller

18.6 Sector size optimization

To further improve the write performance of the NAND Flash memory, it is essential to optimize the sector size. The sector size is the minimum file unit for the computer system. The data is transferred from the OS, operating system, to the memory storage such as SSD and HDD with the unit of the sector. With current operating systems such as Windows, the sector size is optimized for the magnetic drives and is typically 512 Bytes. The 512 Bytes sector size is much smaller than the page size of the NAND Flash memory, typically 4–8 kBytes. The discrepancy between the sector size and the page size significantly degrades the performance of SSD.

In a NAND Flash memory, all memory cells connected to the same control-gate belong to the same page and are programmed at the same time as shown in Fig. 18.18a.

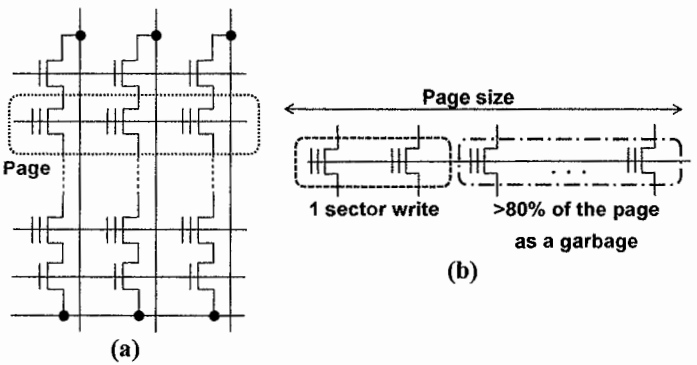


Fig. 18.18. (a) Page of a NAND Flash memory; (b) one sector write operation of a NAND Flash memory

A page can be written only once to avoid a program disturbance shown in Fig. 18.19. In Fig. 18.19, Cell A is programmed. Cell B and Cell C are unselected and are under a weak program bias condition [12], which is called the program disturbance. If a page is programmed more than once, the stress time of the program disturbance becomes longer and the unselected Cell B or Cell C are programmed, causing a data failure.

Petitioner Micron Ex-1021, 0528

Considering that a page can be written only once, if one sector write is performed, only 512 Bytes of the page is programmed and the remaining more than 80% of the page are wasted as a garbage as shown in Fig. 18.18b. Since the NAND Flash memory in SSD is seriously fragmented, in order to efficiently use SSD, the garbage collection frequently happens. During the garbage collection, the block copy operation of the NAND Flash memory is performed [8]. The block copy is composed of the cell-read, cell-program, data transfer between the NAND Flash memory and the NAND controller, and the ECC calculation. As the block copy takes as long as 100 ms, if the block copy happens frequently, the system-level write performance degrades drastically [1].

To avoid a block copy operation and maximize the system-level write performance, it is crucial to minimize the data fragmentation by optimizing the sector size. The sector size should be the same as or multiples of the page size so that the garbage would not happen in case of the one sector write operation. Figure 18.20 shows the page size trend of the NAND Flash memory. As the memory capacity increases, the number of memory cells connected to the same control gate increases and as a result the page size also increases. Furthermore, in case of the interleaving operation, the effective page size is increased to the number of chips in parallel, N times the page size of one NAND Flash memory. Therefore, the optimal sector size should be 128–256 kBytes.

The most straightforward method to increase the sector size is to change the file system of the OS [1]. However, the software size of the OS is very large and it needs huge engineering efforts to change the OS. The OS is usually changed every about 5 years and it is almost impossible to change the OS in accordance with the frequent page size change of the NAND Flash memory. Thus, optimizing the sector size by efficiently using a write buffer in the NAND controller is essential.

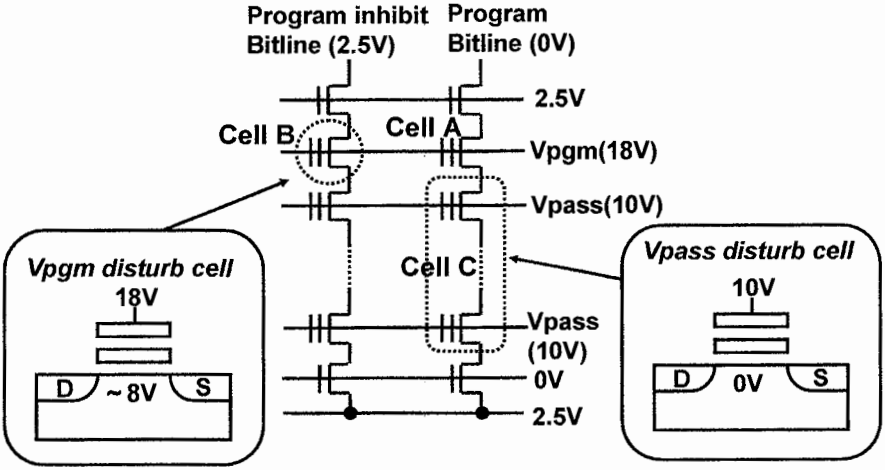


Fig. 18.19. Program disturbance of a NAND Flash memory

Petitioner Micron Ex-1021, 0529

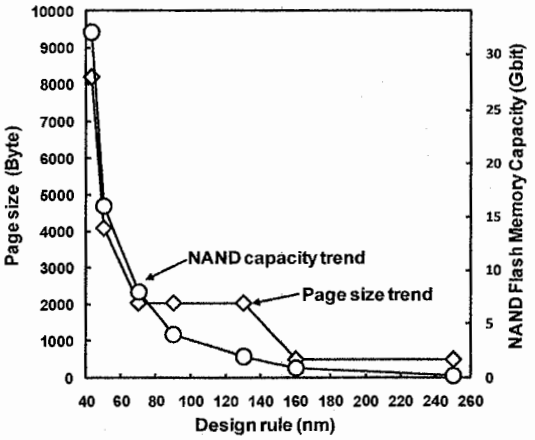


Fig. 18.20. Page size and memory capacity trend of a NAND Flash memory

In case of the one sector write operation, one sector program data is transferred from the OS to the NAND controller. The NAND controller temporarily stores the program data in the write buffer and the data is not programmed to the NAND Flash memory. Then, multiple program data are transferred from the OS to the NAND controller and are accumulated in the write buffer. When the program data size in the write buffer of the NAND controller becomes comparable with the page size, the write operation to the NAND Flash memory is performed. By efficiently using the write buffer in the NAND controller and optimizing the sector size, the fragmentation of SSD is eliminated and the SSD performance is maximized.

18.7 Adaptive program-voltage generator for 3D-SSD

As mentioned in the previous sections, decreasing power consumption is the key design issue of SSDs. The best strategy to decrease the power is lowering the supply voltage, V_{DD} from 3.3 to 1.8 V. Yet, at 1.8 V V_{DD} the power consumption of the conventional charge pumps to generate the program voltage, V_{PGM} (20 V), drastically increases and the total power consumption of the NAND only slightly decreases as shown in Fig. 18.21a. What's worse, the area of the charge pump more than doubles, which increases the NAND chip area by 5–10%. To overcome this dilemma, a low power program voltage generator (PVG) using a boost converter with an adaptive frequency and duty cycle (AFD) controller is proposed [4].

The 3D-integrated SSD proposed in [4] is shown in Fig. 18.21b. NAND chips, DRAM, a NAND controller and the proposed PVG are integrated with SiP, System in Package. The PVG consists of an inductor in an interposer, the high voltage MOS (HVMOS) and the AFD controller. In the system, the cost is also minimized. An inductor is available with no area penalty by using wiring in the interposer connecting the NAND controller, NAND Flash memories and DRAMs.

Petitioner Micron Ex-1021, 0530

The die size of the NAND decreases by 5–10% because no charge pump is needed. The HVMOS is fabricated with a matured NAND process and its area is just 15% of the conventional charge pump. Since the die size of the AFD controller is only 0.188 mm^2 with a 0.18 μm CMOS process, it can be integrated in a NAND controller with a negligible area increase.

A PVG using a boost converter was reported for a NOR Flash memory [15]. A comparison between the PVG for a NOR Flash memory and for a NAND Flash memory is summarized in Fig. 18.22a. In a NOR Flash memory, the load of the PVG is resistive. The PVG continuously supplies load current of 20 mA at an output voltage, 5 V. In such a resistive load and a low output voltage condition, a conventional PWM is employed [15]. Conversely, in a NAND Flash memory, the load is capacitive and V_{PGM} is 20 V. During the program, V_{PGM} is applied to the word-line and a DC load current of 20 μA flows. Also, a PVG for a NAND Flash memory should operate on-and-off to save power. In this condition both switching frequency and duty cycle must be dynamically optimized and the conventional PWM changing only the duty cycle cannot be used.

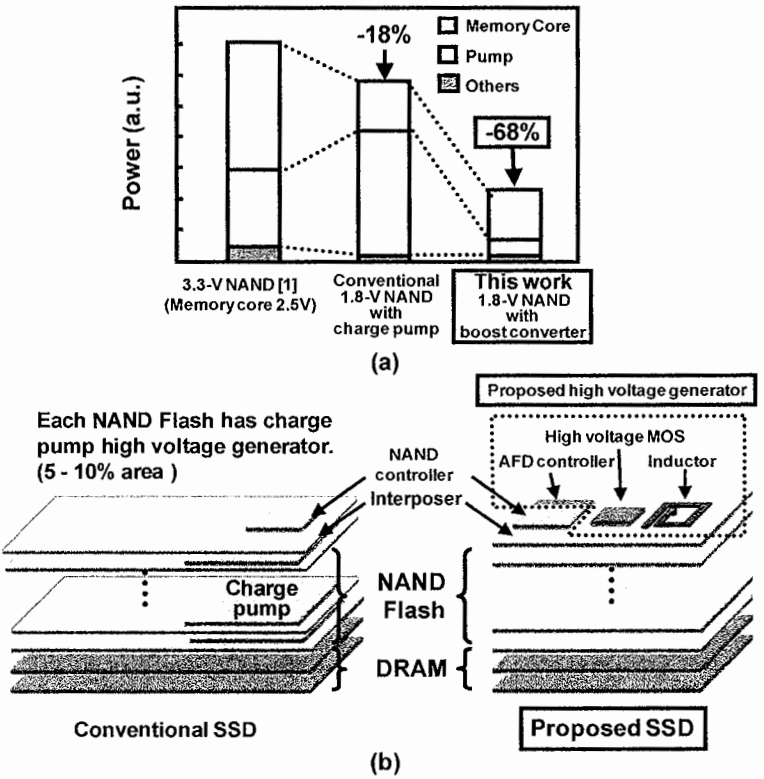
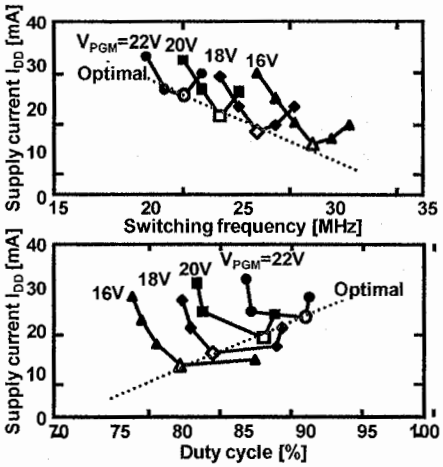


Fig. 18.21. (a) Comparison of the power consumption of NAND Flash memories; (b) structure of the 3D-integrated SSD

Petitioner Micron Ex-1021, 0531

	PWM [4]	This work
Load	Resistive	Capacitive
V_{PGM}	5V	20V
I_{LOAD}	20mA	20μA
Operation	Continuously on	On-and-off
Control	Duty cycle only	Frequency and duty cycle

(a)



(b)

Fig. 18.22. (a) Comparison of the conventional boost converter [15] and adaptive program voltage generator [4]; (b) measured optimal switching frequencies and duty cycles for various V_{PGM}

To identify the most power efficient frequency and duty cycle, an input supply current, I_{DD} is measured with the PVG. As shown in Fig. 18.22b, each V_{PGM} has different optimal frequency and duty cycle minimizing I_{DD}. In other words, the power efficiency is a function of V_{PGM}, a switching frequency and a duty cycle since the PVG operates in a discontinuous mode with a capacitive load. With a bit-by-bit program verifying scheme, in each program cycle, V_{PGM} is incremented by 0.5 V from 15 to 25 V [16, 17]. For each V_{PGM}, the AFD controller adaptively manages the switching frequency and the duty cycle simultaneously so that the energy loss is minimized.

Figure 18.23 shows the schematic diagram of the PVG with the AFD controller. V_{PGM} is monitored with three comparators and the control logic selects the proper switching frequency and duty cycle from the register sets, $Reg._L$, $Reg._M$, and $Reg._H$. These registers store a table of the frequency and the duty cycle which minimize both the power and the output voltage fluctuation. The digital controlled oscillator (DCO) is depicted in Fig. 18.24. The DCO consists of current reference circuits and a couple of capacitor arrays. The DCO enables the clock shape to be only determined by the resistor and the capacitor since the reference current I_{REF} is copied to the node V_{CAPA} and V_{CAPB} with the current mirror [18]. The frequency and the duty cycle are robust against V_{DD} fluctuation, transistor global V_{TH} variation and temperature variation. Switching time of DCO, T_{ON} and T_{OFF} are expressed as $R \cdot C_{A1\text{-}An}$ and $R \cdot C_{B1\text{-}Bn}$, respectively. Since $C_{A1\text{-}An}$ and $C_{B1\text{-}Bn}$ are independently selected according to the data in the registers, T_{ON} and T_{OFF}, that is, the switching frequency and the duty cycle are independently controlled.

To suppress the V_{PGM} fluctuation, the AFD controller dynamically changes the frequency and the duty cycle in three steps as shown in Fig. 18.25.

Petitioner Micron Ex-1021, 0532

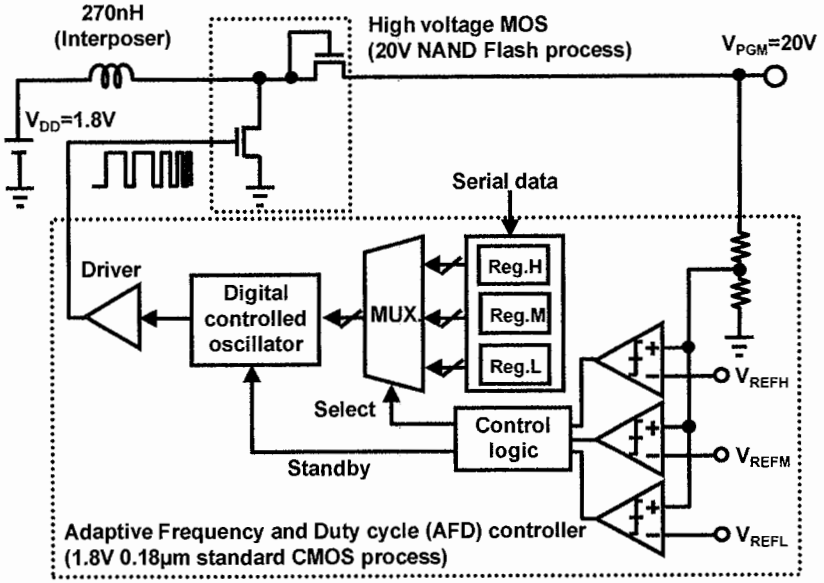


Fig. 18.23. Circuit diagram of the adaptive program voltage generator

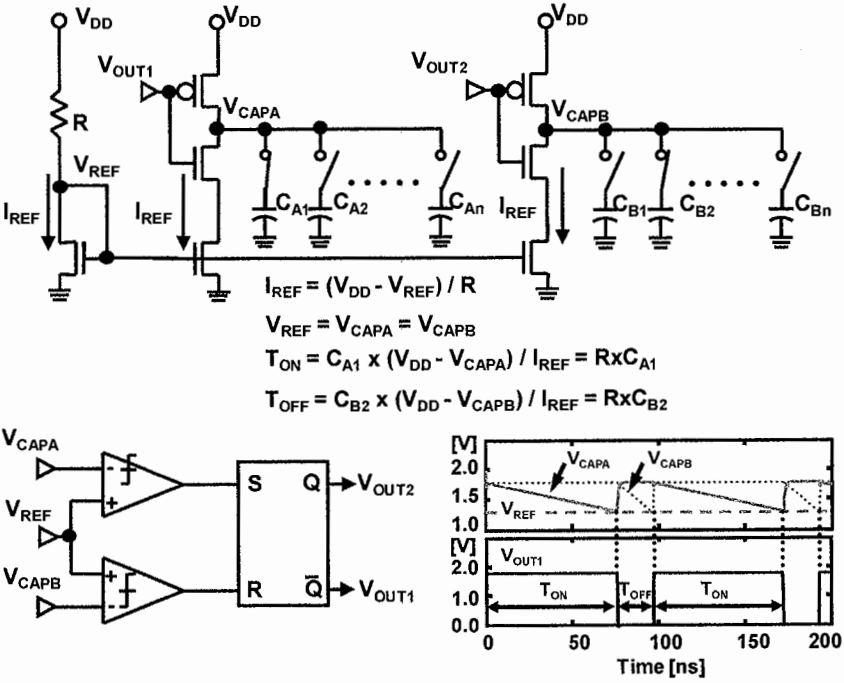


Fig. 18.24. Digital controlled oscillator (DCO)

Petitioner Micron Ex-1021, 0533

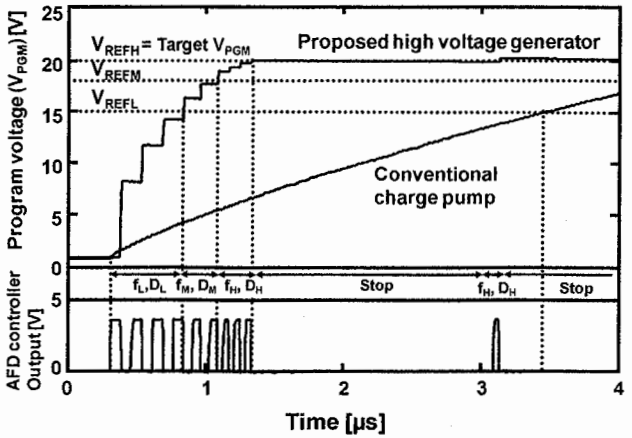


Fig. 18.25. Simulation result of the adaptive program voltage generator

In the first step, the power efficient lower frequency is chosen. The AFD controller outputs pulses with the switching frequency and duty cycle set, f_L/D_L determined by $Reg._L$. V_{PGM} rises coarsely and rapidly until V_{PGM} reaches V_{REFL}. With f_L/D_L, the voltage increment for each pulse is 5 V. The frequency becomes higher in the second and the third steps. In the second step, the AFD controller changes the switching pulse from f_L/D_L to f_M/D_M determined by $Reg._M$.

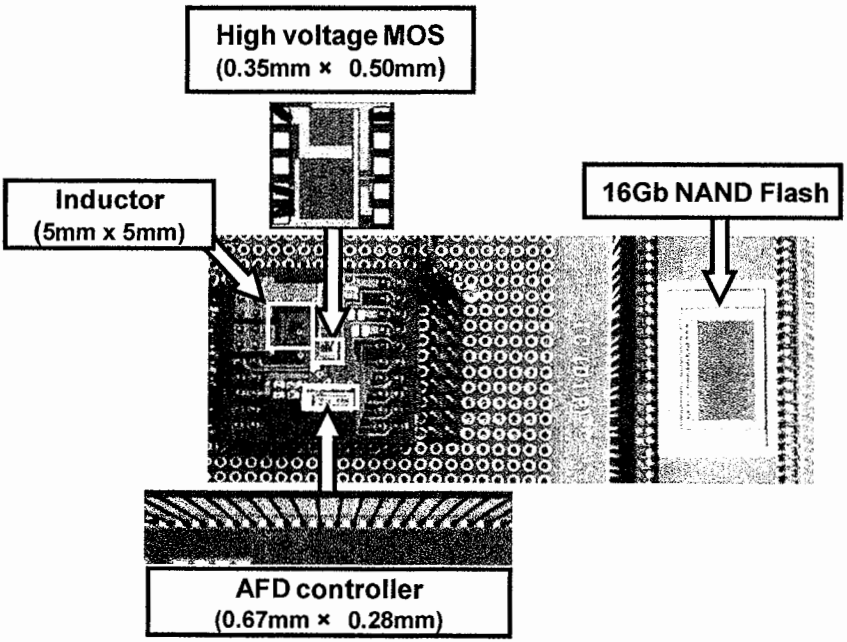
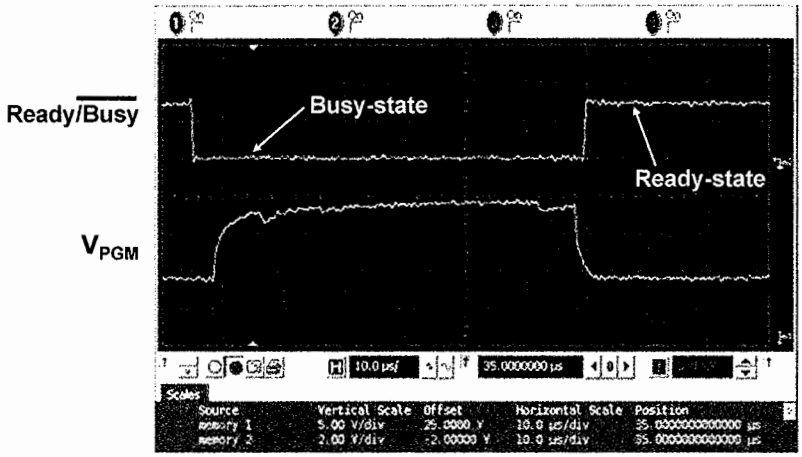


Fig. 18.26. Die microphotograph of 3D-integrated SSD

Petitioner Micron Ex-1021, 0534



	This work (Measured)	Conventional Charge Pump (Simulated)
Power consumption	30nJ (12%)	253nJ (100%)
Chip Area (high voltage MOS)	0.175mm² (15%)	1.19mm² (100%)
Chip Area (AFD controller)	0.188mm²	-----
Rising time	0.92μs (27%)	3.45μs (100%)
Technology (high voltage MOS)	20V NAND Flash process	--------
Technology (AFD controller)	1.8V 0.18μm standard CMOS	--------
Supply voltage	1.8V	1.8V

Fig. 18.27. Experimental results and key features of 3D-integrated SSD with the adaptive program voltage generator

Finally, the AFD controller finely raises V_{PGM} with f_H/D_H toward the target voltage. When V_{PGM} reaches the target voltage, the AFD controller stops switching pulses to save power. As a result, the PVG raises V_{PGM} more than three times faster than the conventional charge pump with a minimum power. V_{PGM} is precisely controlled with less than 0.3 V fluctuation, which enables a tight memory cell V_{TH} distribution.

Figure 18.26 shows the microphotograph of the bread board model of SSD consisting of the HVMOS chip (0.35 mm × 0.50 mm), the AFD controller chip (0.67 mm × 0.28 mm), a 270 nH, 0.5 Ω inductor in an interposer (5 mm × 5 mm), and a 56 nm 16 Gb NAND Flash memory chip. Key features are summarized in Fig. 18.27. The measured waveforms during the program of a 56 nm 16 Gb NAND Flash memory with the PVG is shown in Fig. 18.27. When a write command inputs to the NAND, the Ready/Busy-signal turns to low and the NAND goes into the busy state. The program voltage is supplied from the PVG and the program pulse is applied to the memory cells. Then, the verify-read detects that all memory cells are successfully programmed and the Ready/Busy-signal returns to high.

Petitioner Micron Ex-1021, 0535

The measured power consumption of the PVG is 30 nJ, which is only 12% of the conventional charge pump. Measured rising time of the PVG is 0.92 μs (at V_{DD}, 1.8 V and V_{PGM}, 15 V), while that of the charge pump is 3.45 μs. As the rising time of the V_{PGM} decreases by 2.53 μs, the program pulse width can be shortened by 2.53 μs. As a result, the total program time of a NAND Flash memory, a sum of the program pulse width and the verify-read time is 7.8% shorter than the conventional 1.8 V NAND Flash memory. The area of the HVMOS chip is just 15% of the charge pump without a control circuit or an oscillator. By decreasing V_{DD} from 3.3 to 1.8 V, the total power consumption of a NAND Flash memory decreases by 68% as shown in Fig. 18.21a.

18.8 Conclusions

In this chapter, circuit technologies co-designing the NAND Flash memory and the NAND controller are described to best optimize both the NAND Flash memory and the NAND controller of SSD. As a result, a highly reliable and high speed operation of SSD is achieved. Two novel low power circuit technologies, the selective bit-line precharge scheme and the advanced source-line program are discussed. By eliminating the unnecessary bit-line precharge during the verify-read and reducing the load capacitance during the program pulse, the operation current of sub-30 nm NAND Flash memories is reduced by 60%. Moreover, a low noise circuit technology, the intelligent interleaving, is discussed. By using the intelligent interleaving, multiple NAND chips are programmed simultaneously without a power supply noise. With these new circuit technologies, the performance of sub-30 nm generation SSD improves by 150% without a cost penalty or a circuit noise. Furthermore, an intelligent 3D-integration of SSD is described. A PVG using a single-stage boost converter for a NAND Flash memory is introduced. The PVG with the AFD controller brings a voltage scaling merit for a NAND Flash memory and realizes a drastic power reduction of the 3D-integrated SSD.

References

1. K. Takeuchi, "NAND successful as a media for SSD", ISSCC, Tutorial T7, 2008.
2. K. Takeuchi, "Novel Co-design of NAND Flash Memory and NAND Flash Controller Circuits for Sub-30nm Low-Power High-Speed Solid-State Drives (SSD)", Symposium on VLSI Circuits Tech. Dig., pp. 124–125, 2008.
3. K. Takeuchi, "Novel Co-design of NAND Flash Memory and NAND Flash Controller Circuits for sub-30nm Low-Power High-Speed Solid-State Drives (SSD)," IEEE Journal of Solid-State Circuits, vol. 44, no. 4, pp. 1227–1234, April 2009.
4. K. Ishida, et. al., "A 1.8V 30nJ Adaptive Program-Voltage (20V) Generator for 3D-Integrated NAND Flash SSD," ISSCC Tech. Dig., pp. 238–239, 2009.

Petitioner Micron Ex-1021, 0536

5. C. Park, et. al., "A High Performance Controller for NAND Flash-Based Solid State Disk (NSSD)", NVSMW Tech. Dig., pp. 17–20, 2006.
6. Y. Li, et. al., "A 16Gb 3b/cell NAND Flash Memory in 56nm with 8MB/s Write Rate", ISSCC Tech. Dig., pp. 506–507, 2008.
7. N. Shibata, et. al., "A 70nm 16Gb 16-level-cell NAND Flash Memory", Symposium on VLSI Circuits Tech. Dig., pp. 190–191, 2007.
8. K. Takeuchi, et al., "A 56nm CMOS 99mm2 8Gbit Multi-level NAND Flash Memory with 10Mbyte/sec Program Throughput", IEEE Journal of Solid-State Circuits, vol. 42, pp. 219–232, 2007.
9. T. Tanaka, et. al., "A Quick Intelligent Program Architecture for 3V-Only NAND EEPROMs", Symposium on VLSI Circuits Tech. Dig., pp. 20–21, 1992.
10. K. Takeuchi, et. al., "A Multipage Cell Architecture for High-Speed Programming Multilevel NAND Flash Memories", Symposium on VLSI Circuits Tech. Dig., pp. 67–68, 1997.
11. T. Hara, et. al., "A 146mm2 8Gb NAND Flash Memory with 70nm COMS Technology", ISSCC Tech. Dig., pp. 44–45, 2005.
12. 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.
13. K. Takeuchi, et. al., "A Source-line Programming Scheme for Low Voltage Operation NAND Flash Memories", Symposium on VLSI Circuits Tech. Dig., pp. 37–38, 1999.
14. K. Takeuchi, et. al., "A Double-Level-Vth Select Gate Array Architecture for Multi-level NAND Flash Memories", Symposium on VLSI Circuits Tech. Dig., pp. 69–70, 1995.
15. R. Sundaram et al., "A 128Mb NOR Flash Memory with 3MB/s Program Time and Low-Power Write Using an In-Package Inductor Chrage-Pump," ISSCC Dig. Tech. Papers, pp. 50–51, 2005.
16. K. Takeuchi et al., "A 56nm CMOS 99mm2 8Gb Multi-level NAND Flash Memory with 10MB/s Program Throughput," ISSCC Dig. Tech. Papers, pp. 144–145, 2006.
17. K. Kanda, et. al., "A 120mm2 16Gb 4-MLC NAND Flash Memory with 43nm CMOS Technology," ISSCC Dig. Tech. Papers, pp. 430–431, 2008.
18. T. Tanzawa and T. Tanaka, "A Stable Programming Pulse Generator for Single Power Supply Flash Memories," IEEE Journal of Solid-State Circuits, vol.32, no. 6, pp. 845–851, 1997.

Petitioner Micron Ex-1021, 0537

19 Radiation effects on NAND Flash memories

M. Bagatin[1], G. Cellere[2], S. Gerardin[3] and A. Paccagnella[4]

Electronic chips operating at sea level are constantly bombarded by a shower of high-energy neutrons, which originate from the interactions of cosmic rays with the outer layers of the atmosphere. The neutron flux changes with altitude, reaching a peak very close to the cruise altitude of airplanes, posing an even more serious threat to avionics. In addition, inevitable radioactive contaminants in the chip materials emit alpha particles, which may reach sensitive device areas and produce errors. Spacecraft and satellite electronics must operate reliably in a much harsher environment, characterized by a significant presence of ionizing radiation, in the form of protons, electrons, and heavy-ions coming from various sources. Ionizing radiation can cause either permanent or temporary damage to electronic chips, generating a plethora of effects, from flipping an SRAM memory bit from 1 to 0 or vice versa, to burning-out a power MOSFET.

NAND Flash memories are not immune to radiation effects. On the contrary, due to their complexity and large number of diverse building blocks, they exhibit quite complex failure signatures when exposed to ionizing particles. This is especially true for the space environment, where mitigation strategies are mandatory due to the severity of the effects. Yet, ionizing radiation issues are becoming a growing concern also at ground level, due to the ever-decreasing feature size, which is making floating gate cells sensitive even to small external disturbances, such as those caused by atmospheric neutrons and alpha particles.

This chapter will first present a brief overview of radiation effects, focusing on CMOS technology, and then analyze in detail the immediate and long-term effects of ionizing radiation on NAND Flash memories, both on the floating gate array and on the peripheral circuitry. Emphasis will be put on effects peculiar of the space environment, but recent results on the sensitivity to the terrestrial environment will be presented as well.

[1] University of Padova, Italy, bagatinm@dei.unipd.it
[2] University of Padova, Italy, simone.gerardin@dei.unipd.it
[3] University of Padova, Italy, giorgio.cellere@ieee.org
[4] University of Padova, Italy, alessandro.paccagnella@dei.unipd.it

R. Micheloni et al., *Inside NAND Flash Memories*, DOI 10.1007/978-90-481-9431-5_19,
© Springer Science+Business Media B.V. 2010

Petitioner Micron Ex-1021, 0538

19.1 Introduction to radiation effects in CMOS circuits

This section provides the reader with the necessary background to understand ionizing radiation-related issues in integrated circuits, focusing first on the sources of radiation in the operating environments of electronic chips, and then analyzing the interactions of radiation with chip materials and structures.

19.1.1 Environments

Before showing the effects on electronic chips, we illustrate the main sources of ionizing radiation as a function of the target environment. We start with the terrestrial environment, we then analyze the space one, which is far more severe from the radiation standpoint. Shielding is impractical in many situations, because the amount of material required would be just too large or heavy due to the very high particle energies.

Radiation-harsh man-made environments, such as high-energy physics experiments and nuclear power plants are also of interest, but are not covered here. The reader is referred to [1] for a discussion.

19.1.1.1 Terrestrial environment

Neutrons, originating from the interaction of cosmic rays with the outer layers of the atmosphere, and alpha particles, emitted by radioactive contaminants in the chip materials are the main sources of radiation-induced effects at sea level.

Cosmic rays were discovered by Victor Hess in 1912, using a balloon. Primary cosmic rays are coming from outside of our solar system. They mainly consist of protons (85%) and Helium nuclei (12%), but may also contain heavier elements and electrons [1]. The energy can be extremely high, making them very penetrating. As cosmic rays reach the atmosphere, they interact with nitrogen and oxygen atoms, generating a cascade of secondary particles (secondary cosmic rays). Several reactions are possible, giving rise to many different particles (protons, pions, muons, neutrons) and an electromagnetic component. In turn, the generated elements can have enough energy to create further particles. As a result, as cosmic rays penetrate into the atmosphere, the number of particles initially increases and then decreases, when the shielding effect of the atmosphere becomes dominant.

From the standpoint of electronic components, atmospheric neutrons are the most important particles, because, even though they come in a smaller number than muons, they are able to trigger nuclear reactions inside the chips, giving rise to charged secondary by-products, that deposit charge and disturb the operation of integrated circuits.

The neutron flux has some peculiar characteristics. It increases with altitude, peaking (more than two orders of magnitude larger than at sea level) very close to the one of commercial airplanes. Thus, avionics is much more threatened by atmospheric neutrons, than electronics on the ground. The energy spectrum features a

Petitioner Micron Ex-1021, 0539

characteristic 1/energy dependence. The energy range of interest for radiation effects includes thermal neutrons (around 25 meV) and neutrons with energy above about 10 MeV. Thermal neutrons are important, because of their large interaction cross section with ^{10}B, which is used for intermetal isolation, even though many semiconductor manufactures (but not all) have eliminated it because of this issue. High-energy neutrons (>10 MeV) are capable of triggering nuclear reactions, for example with silicon atoms, giving rise to charged secondary by-products. The neutron flux also changes with latitude, atmospheric pressure, and solar activity. Tables, models, and software tools are available to estimate the neutron flux for a given location [2]. As a reference, the high-energy atmospheric neutron flux at New York City is about 13 neutrons/cm^{-2}/hour^{-1}.

Radioactive contaminants inside chip materials are the second important source of radiation effects at sea level. They tend to be less important at high altitudes (where the neutron flux is much higher and dominates the error rate), but can be responsible for a large part of radiation-induced errors on the ground, especially on deeply scaled technologies. Typical emission levels in a chip are on the order of 10^{-3} alphas/cm^{-2}/hour^{-1} [2].

19.1.1.2 Space

There are three main sources of radiation in space [3], schematically depicted in Fig. 19.1:

- Galactic cosmic rays coming from outside our solar system
- Particles originating from the Sun during solar particle events
- Particles trapped in planets' magnetospheres

We already discussed cosmic rays in the previous section. Of course, there is no shielding from the atmosphere in space, and cosmic rays may directly hit electronic chips.

The Sun is both a source and a modulator of radiation. Its activity follows a characteristic 11-year cycle related to its magnetic field, which changes polarity at every cycle, due to the gaseous nature of the star. There are 7 years of high activity and 4 years of low activity. One of the most evident manifestation of the solar cycle is the number of sunspots, recorded since the seventeenth century, that decreases and increases with the same period. Solar Particle Events (SPE) occur at higher frequency during the declining phase of the solar maximum. SPE includes two types of events: solar flares and Coronal Mass Ejections (CME). Solar flares originate when the coronal magnetic field becomes too large, causing an energy burst release. CMEs are eruptions of plasma, which give rise to a shock wave and the outward release and acceleration of particles. The particles emitted by solar events comprise all naturally-occurring elements of the periodic table, from protons to uranium. In addition to SPE, a progressive loss of mass from the Sun, in the form of protons and electrons, continuously takes place, because of the high temperature of the corona, which provides some particles with enough energy to escape the gravitational pull. These particles have an embedded magnetic field, which, as we shall see, interacts with the planets' magnetic fields.

Petitioner Micron Ex-1021, 0540

Besides being a source of radiation, the Sun also has an effect on the other sources of radiation in space. The solar activity impacts on the galactic cosmic ray flux in an anticorrelated way: the higher the solar activity, the smaller the cosmic ray flux, because of the shielding effects produced by the particles emitted by the Sun. The Sun interacts also with the magnetosphere of the planets (of course, for those planets which have one). In the following we will discuss the Earth, but similar considerations can be drawn for other planets as well. The Earth's magnetosphere has two components, an internal one, which is approximately a dipole, and one deriving from the solar wind.

The Earth's magnetic field is strong enough to capture charged particles. Once captured, the particles spiral around the field lines, moving from one pole to the other, where they bounce back. A slower longitudinal movement takes also place, in opposite directions for positively- and negatively-charged particles. The particles trapped around the Earth form two belts: the outer belt, consisting mainly of electrons, and the inner belt, which includes protons and electrons.

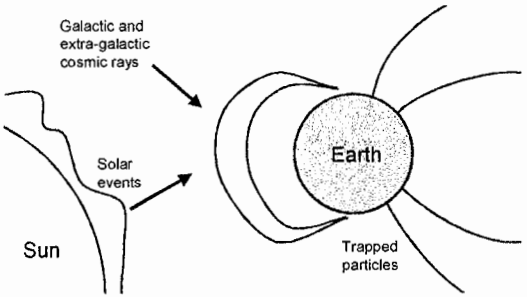


Fig. 19.1. Sources of radiation in space

The magnetic field axis forms an angle of about 11° with respect to the North–South axis, and its center is displaced by more than 500 km from the Earth's center. This causes a dip in the magnetic field over the South Atlantic, which is called the South Atlantic Anomaly (SAA) and is responsible for many problems in low orbits.

As can be understood from this description of the space weather, the amount of radiation which hits a spacecraft or a satellite is highly dependent on the orbit, the Sun activity, and other factors. In addition, the dose received by a specific component is a function of its position inside a satellite and of the materials around it.

19.1.2 Overview of radiation effects

19.1.2.1 Radiation–matter interaction

Ionizing particles (electrons, protons, heavy ions) impinging on a semiconductor or insulating material deposit energy as they are stopped by the target material.

Petitioner Micron Ex-1021, 0541

Different energy loss mechanisms exist: Coulomb interactions, either with electrons or with nuclei, and nuclear reactions [1]. As a result, electron–hole pairs are created (ionizing component) and lattice atoms are displaced from their lattice position (non-ionizing energy loss). The ionizing loss is by far the most important in CMOS circuits and will be treated here in more detail. It can be characterized by the Linear Energy Transfer (LET), which is the energy deposited per unit length by the impinging particle in the target material, through ionization processes. 3.6 eV are needed to produce an e–h pair in silicon, while 17–18 eV are required for an e–h pair in silicon dioxide [4, 5]. The LET is customarily normalized by the density of the target material, and is measured in units of $MeV \cdot cm^2 \cdot mg^{-1}$. The LET depends on the particle energy, displaying a maximum for intermediate energies, called the Bragg peak.

Multiplying the LET by the particle fluence of a monochromatic beam, one obtains the total amount of energy deposited per unit of mass through ionization, which is called Total Ionizing Dose (TID), and is measured in radian, 1 rad = 100 erg/g or Gray, 1 Gy = 1 J/kg in SI units. LET and TID are the fundamental metrics in the study of radiation effects in CMOS circuits.

A large part of the produced electron–hole pairs quickly recombine, both in semiconductors and insulators, soon after generation. The percentage of carriers that survive recombination, called charge yield, depends on the characteristics of the impinging particle, on the target material, and on the magnitude of the local electric field [5–7]. The charge yield decreases for increasing LET, because the pairs are more dense and escape mutual recombination with more difficulty with higher LET. On the contrary, increasing the electric field separates electrons from holes more efficiently and reduces recombination.

Neutral particles such as neutrons, or lowly-charged penetrating ones such as protons, can trigger nuclear reactions, generating secondary charged by-products (heavy ions), thus indirectly ionizing the target material.

Energetic heavy particles (atomic number ≥ 2) follow a straight trajectory in matter, being only rarely deflected. For a given energy and species, the distance traveled in the target material is well defined and almost constant for nominally identical ions, and is called range. The release of energy associated with the passage of an ion gives rise to a cylindrical electron–hole pair track, whose size is on the order of tens of nanometers in silicon, and considerably less in silicon dioxide [8].

For heavy energetic particles, the loss of energy occurs mostly through ionization, but as the particles slow down, the contribution of non-ionizing processes becomes increasingly important, and it is the dominant contribution near the end of the particle range (i.e., when the particle is about to stop). Light particles, such as electrons, experience frequent scattering events and describe a zig-zag trajectory. No range can be defined for such particles.

Photons are not a major concern in space, but can be used for x-ray PCB inspections at significant levels (doses used at airports check points are very low), and in addition gamma- and x- rays sources are routinely used for ground TID testing of electronic components. Photons interact in different ways with matter depending on their energy. For low-energy photons (such as those produced by

Petitioner Micron Ex-1021, 0542

10-keV x-rays; a standard test source) the photoelectric effect is the dominant mechanism. For higher-energy photons (such as those produced by Co^{60} gamma rays; another standard test source), Compton scattering is the dominant mechanism.

19.1.2.2 Categories of radiation effects

The electron–hole pairs generated by the interaction of radiation and chip materials may induce a variety of effects. They cause charge trapping and defect generation in insulators. They generate spurious currents in reverse-biased pn junctions, which may upset memory elements or give rise to voltage transients in combinatorial logic and analog circuits. As we will see, in floating gate cells they generate discharge currents that may corrupt the stored information.

Depending on the type and characteristics of the impinging radiation, both irreversible (i.e., causing permanent damage to the exposed device) and reversible (loss of information) effects may arise.

In some cases, a single particle, with highly ionizing power, deposits enough charge to cause malfunctions, which range from the corruption of the information stored in a single memory bit, to the burn-out of power MOSFETs. This type of events are called Single Event Effects (SEE). SEEs affect memory, combinatorial, and analog circuits with destructive or non-destructive effects.

In other cases, damage builds up strike after strike, during a prolonged exposure to ionizing radiation, that generates drifts in components parameters (such as threshold voltage shift, power consumption increase, gain decrease, etc.). This second type of effects can involve damage to both dielectrics layers (gate oxide, lateral isolation, passivation layers, etc.), resulting in so-called Total Ionizing Dose effects (TID), and to bulk semiconductor materials, generating Displacement Damage (DD). TID is responsible for parametric degradation and functional failure of MOSFETs (e.g., threshold voltage shift) and BJTs (e.g., current gain decrease). DD effects are due to the generation of point or extended defects in the crystalline semiconductor lattice, degrading devices which rely on bulk semiconductor conduction, such as BJTs, solar cells, etc., but not MOSFETs. Annealing effects can take place, even at room temperature, causing TID and DD effects to change over time, resulting in more or less severe degradation.

Low-LET particles, such as electrons, generate TID and DD. Protons give rise to TID, DD, and SEE. Protons SEEs are usually indirectly generated, i.e., a charged secondary product, coming from a proton nuclear reaction, is responsible for the SEE. Recently, bit-flips resulting from proton direct ionization (without the generation of a secondary particle) have been observed in very scaled SRAMs as well [9]. Alpha particles (He ions) cause TID and SEE. Heavier ions ($Z > 2$) generate SEEs and microdose effects, i.e., effects similar to TID, but localized to the ion track.

Given the fluxes and types of particles at play (see Sect. 19.1.1), TID and DD effects occur primarily in space, HEP experiments, and nuclear power plants, whereas SEEs take place also at sea level. TID effects can degrade circuit parameters also when high doses of radiation are used for inspection purposes on the ground.

Petitioner Micron Ex-1021, 0543

19.1.3 Total ionizing dose effects

Total ionizing dose effects in CMOS devices are due to charge trapping and defect generation in insulating layers. Historically, the main problems of MOSFETs with respect to radiation were shifts in the threshold voltage and degradation in trasconductance. Thanks to the thinning of the gate oxide, radiation effects in the gate stack are now much less severe. Nevertheless, damage may still occur in the thick isolation oxides, which surround the conductive channel.

19.1.3.1 Basic mechanisms

Figure 19.2 depicts the fundamental mechanisms underlying charge trapping and defect generation in an oxide layer. Radiation generates defects in insulating layers through indirect processes, i.e., by releasing species, such as holes and hydrogen ions, which in turn are responsible for the radiation response of the exposed devices. The first step in the process is the generation of energetic electron–hole pairs. After thermalization, the electrons, which have a higher mobility, are quickly swept towards the anode by the applied bias, leaving the heavier and slower holes to move inside the oxide in the opposite direction [10]. But before they do that, a large part of the e–h pairs recombine as a function of the electric field.

The surviving holes migrate towards the cathode under the influence of the applied field. Holes arriving at the Si/SiO_2 interface, where the density of defect sites is higher, can be trapped. The amount of trapped charge is very dependent on the quality of the oxide, with "hardened" ones showing orders of magnitude less radiation-induced charge trapping than "soft" oxides.

During the transport and trapping of holes, hydrogen ions (protons) are likely released. Hydrogen ions arriving at the interface can generate interface traps by reacting with hydrogen-passivated dangling bonds at the interface. Interface traps may readily exchange carriers with the channel, and are full or empty depending on the position of the relevant quasi Fermi level. Interface states are usually donor (positive when empty, neutral when charged) when their energy position is above midgap or acceptor (neutral when empty, negative when charged) when below. The creation of interface traps is much slower (up to thousands of seconds) than the build-up of trapped charge [11].

Annealing of charge in oxide-traps may start immediately and occurs due to tunneling or thermal processes. Indeed, the trapped charge can be neutralized by electrons, either being thermally excited from the valence band, or tunneling through the oxide barrier 10. In the first case, annealing increases with higher temperatures and for shallower traps; in the second case, it depends on the tunneling distance and trap energy spatial position. In contrast, high temperatures are needed to recover interface traps [12]. As a result, in low-dose rate environments (such as space), interface traps may play a predominant role.

Petitioner Micron Ex-1021, 0544

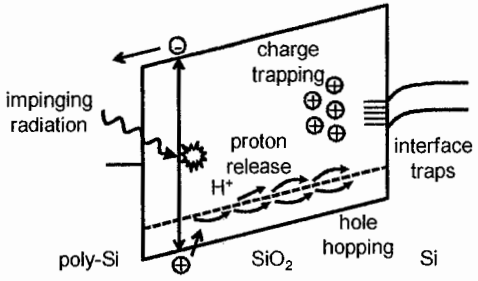


Fig. 19.2. Basic mechanisms of total ionizing dose effects

The build-up and anneal mechanisms give rise to a strong time dependence of TID effects, leading to "apparent" dose-rate effects [13]. For instance, at high dose rates and short times, anneal of trapped charge has just begun, as well as interface trap build-up: as a result, charge trapping dominates over interface state generation. However, given the same time to the trapped charge to anneal and to the interface traps to build-up, the effects do not depend on the dose rate [14].

19.1.3.2 Effects on MOSFETs

Figure 19.3 shows the effects of radiation exposure on the drain current vs gate voltage curves for n-channel and p-channel MOSFETs with a thick gate oxide.

I_d–V_{gs} curves are rigidly shifted towards lower gate voltages by positive fixed trapped charge inside the gate oxide. Radiation-induced interface traps affect the subthreshold swing, hence the threshold voltage, and degrade mobility, because they increase the scattering rate. The contributions of trapped charge and interface traps in p-channel MOSFETs are both positive when the transistor is on, and add up. On the contrary, they tend to cancel out in n-channel MOSFETs, leading to rebound effects, where the threshold voltage first decreases and then increases with dose [15]. Annealing reduces the impact of oxide charge trapping on the device characteristics, making the contribution of interface traps more important for space applications than for high dose rate applications.

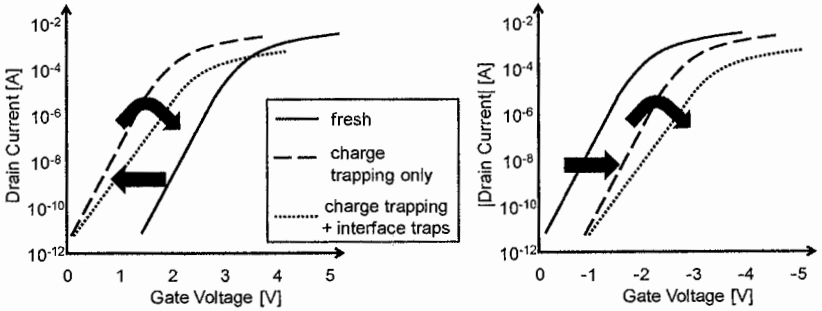


Fig. 19.3. Total ionizing dose effects in n-channel (a) and p-channel (b) MOSFETs

Petitioner Micron Ex-1021, 0545

19.1.3.3 Scaling

TID issues in CMOS devices have been strongly mitigated by scaling, and the associated thinning of the gate oxide. A simple model (valid for low doses and relatively thick oxides) shows that the amount of radiation-induced threshold voltage shift quadratically decreases with decreasing oxide thickness [16]. This decrease is even sharper for ultra-thin gate oxides (<10 nm), due to the short tunneling distance from the gate and the channel to the oxide bulk [17]. State-of-the-art silicon oxides for low-voltage MOSFETs are only 1–2 nm thick, and neither the build-up of oxide trap charge, nor the interface trap generation is significant even after high radiation doses.

Thinning of the gate oxide has brought about also detrimental effects, in particular Radiation-Induced Leakage Current (RILC) [18, 19]. This is an increase in the leakage current of gate dielectrics after exposure to low-LET particles, that has been explained through inelastic trap assisted tunneling, similar to the well-known stress-induced leakage current produced by electrical stress [20]. RILC has been observed after exposure to Co-60 gamma rays, 8-MeV electrons, 10-keV x-rays, and has been found to increase linearly with dose and depend on the applied field during irradiation. RILC is not much of a concern in logic circuits, since it only leads to negligible increases in power consumption. Yet, RILC can be a show-stopper for those technologies (such as Flash memories) which rely on the storage of data in an isolated electrode, as we shall see in the last part of this chapter.

Even though the gate oxide exhibits minimal degradation in deeply scaled CMOS technology, thick lateral isolation oxides, such as LOCOS in older technologies and STI in more recent ones [21], are still prone to charge trapping and interface state generation. Charge trapped in the isolation causes the activation of parasitic MOSFETs, which act in parallel to the drawn device. The drain current can be thought of as the superposition of three MOSFETs. Normally, the parasitic transistors are off and they do not interfere with the operation of the main transistor, thanks to their high V_{TH}. However, when radiation-induced charge builds up in the STI, the threshold voltage of the parasitic devices may strongly decrease. Since oxide charge trapping is predominantly positive, this effect is visible especially in n-channel devices.

Leakage can also occur between adjacent devices [21], due to the inversion of the region below the STI and the consequent formation of conducting paths between neighbor MOSFETs. As a result, IC static power consumption can greatly rise, possibly leading to circuit parametric failure.

To overcome the scaling limitations of conventional devices, a number of innovations have been introduced. First, the gate dielectric has been engineered through the use of nitridation, to prevent boron penetration from the polysilicon gate in pMOSFETs. This type of dielectric has been reported to exhibit superior performance in terms of hot carrier degradation, and also radiation hardness, when compared to conventional SiO_2 layers [22].

More recently, the gate stack has been completely revolutionized with the introduction of high-k materials, such as hafnium oxide, which has been

Petitioner Micron Ex-1021, 0546

commercially employed starting from the 45-nm node. The larger physical thickness of these oxides coupled with the more complex manufacturing steps may signify a return to radiation issues in the gate oxide. However, reported results are quite encouraging [23].

19.1.4 Single event effects

A single ionizing particle crossing a sensitive device area can trigger a spurious response in the circuit or damage it. These events are classified as soft errors, if they do not induce any physical damage, but only loss of information (e.g., a bit flip in a memory array); or as hard errors if they do induce permanent damage (e.g., the gate oxide rupture following the strike of a heavy ion). Some other effects may be destructive or not, depending on the intervention of protection structures, such as the latch-up induced by ionizing particles. The most significant SEEs are listed in the following [24, 25].

Non-destructive (soft) effects are listed below.

- Single Event Upset, SEU, the corruption of a single bit in a memory array
- Multiple Bit Upset, MBU, the corruption of multiple bits due to a single particle
- Single Event Transient, SET, a voltage transient induced by an ionizing particle in a combinatorial or analog part of a circuit
- Single Event Functional Interrupt, SEFI, the corruption in the controlling state machine of a chip, leading to functional interruption
- Destructive (hard) effects
- Single Event Gate Rupture, SEGR, rupture of the gate oxide occurring especially in power MOSFETs
- Single Event Burnout, SEB, burnout of a power device (IGBT, MOSFET, etc.)

Effects that may be destructive or not:

- Single Event Latch-up, SEL, the activation of parasitic bipolar structures, leading to a sudden increase in the supply current
- Single Event Snapback, a regenerative feedback mechanism sustained by impact ionization in SOI devices

The cross section is used to express the sensitivity of a chip to a particular type of event, and is defined as:

$$\sigma_{SEE} = \frac{number\,of\,events}{particle\,fluence}$$

The cross section increases for increasing LET, and the cross section versus LET curve is usually fitted with a Weibull cumulative probability distribution. The threshold LET is defined as the minimum LET required to experimentally observe an event, and the saturation LET as the LET above which the cross section does not increase anymore (or at least is considered to saturate).

Petitioner Micron Ex-1021, 0547

The concept of critical charge is often introduced in the study of single event effects. The critical charge is the minimum charge that must be deposited to trigger a certain event, and it is in close relation with the threshold LET.

The SEU in static RAMs is probably the best known SEE and occurs both in space and on Earth. To cause disturbance in a circuit, the charge generated by a heavy ion strike must be collected by a sensitive node. Reverse-biased pn junctions feature a large depletion region and a relatively strong electric field, so they are one of the most likely candidate to collect charge [24]. The process leading to a bit-flip in an SRAM is schematically depicted in Fig. 19.4: an ionizing particle strikes the drain of the off NMOSFET in the cross-coupled inverters, which form an SRAM cell. The particle generates a cascade of electron–hole pairs both inside and outside the depletion region of the reverse-biased drain pn junction. The released charge is collected by the junction, both by drift and diffusion, giving rise to a transient current. This current is sourced by the on PMOSFET in the same inverter, which tends to restore the initial state. Unfortunately, since the PMOSFET has a finite output conductance, the voltage at the struck node decreases, turning the radiation-induced current into a voltage transient. The current decreases the potential of the drain node, possibly below the cell switching voltage. If the radiation-induced transient is long enough, the feedback may cause the cell to flip, changing its initial state.

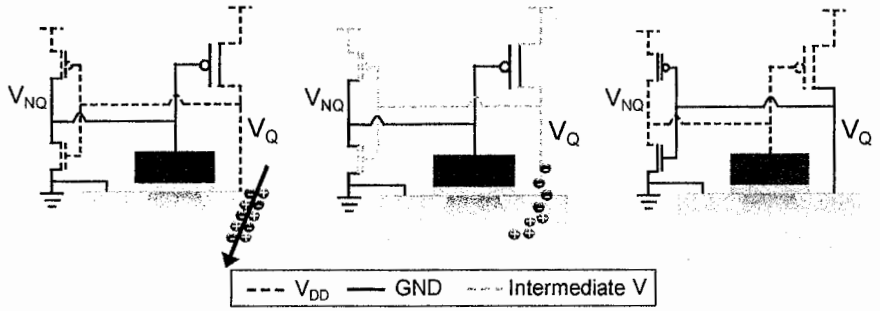


Fig. 19.4. Single event upset in an SRAM cell

Many factors come into play to determine the probability of an upset. The higher the LET of the impinging particle, the larger the deposited charge. Charge collection is also of fundamental importance. The drift component is given by the e–h pairs separated by the electric field in the junction of the depletion region and collected by the drain node. In addition, particles generated in the neutral region may diffuse towards the junction. Simulations have shown an even more complex picture, involving the distortion of the junction potential by the ion track (funnel effect). The funnel effect greatly extends the region over which charge can be collected by drift, and increases SEU sensitivity, even though its impact may be minor in circuits, like SRAM cells, where the junction bias is allowed to change. Indeed, a junction connected to a PMOSFET may go from reverse bias to zero bias during a strike, limiting the influence of drift collection.

Petitioner Micron Ex-1021, 0548

The timescale of these phenomena is different: drift is responsible for the shape of the radiation-induced transient at earlier times; whereas, the slow process of diffusion determines the response at later times.

Charge collection can be even more complicated than that described so far. An important mechanism emerging in scaled CMOS nodes and fundamental in SOI devices is the bipolar amplification of the charge deposited by a heavy-ion strike. This happens when the carriers released by radiation are confined in a well and alter the potential of that well, turning on the parasitic bipolar transistor.

The circuit response is fundamental as well. A faster cell will be easier to upset, since the voltage transient needs to be shorter to trigger the feedback. On the other hand, the characteristics of the restoring PMOSFET may change the voltage drops that develop as a consequence of the ion strike: the higher the restoring PMOS conductance, the lower the SEU sensitivity of the hit node.

Besides memory elements, combinatorial logic can be affected by radiation. Indeed, reverse-biased junctions inside logic circuits can be struck by ionizing particles and give rise to transients [26]. These transients, if not masked, can reach memory elements and be permanently latched.

There are three types of masking: logical, temporal, and electrical [27, 28]. Logical masking occurs when the inputs of a logic gate are set in such a manner that a glitch to one of its input does not result in a change at the output, because of the logic function implemented by the gate. For instance, an AND gate with both inputs low logically masks any transients at its inputs. Temporal masking occurs when the radiation-induced transient reaches the memory elements at a time when these are not sensitive (e.g., far from the edges for an edge-triggered flip flop). Temporal masking is responsible for the linear frequency dependence of SETs. Finally, electrical masking occurs when the logic gate attenuates the incoming SET, because of insufficient bandwidth. Due to ever increasing operating frequency, SETs are expected to become one of the dominant radiation effects in the coming years [29].

Single event effects have become more complex to study, as the feature size has been scaled into the deep submicron realm. Indeed, nowadays the feature size of modern chips has become comparable to the size of the ion track. Phenomena that were once confined to a single circuit node, can now involve multiple nodes [30] and charge sharing can occur. This has caused a significant rise in the number of multiple bit upsets as technologies have been scaled.

19.2 Radiation effects on NAND Flash memories

Commercial nonvolatile Flash memories are very attractive for space engineers, due to their high density and low cost. In fact, Components Off The Shelf (COTS) are arousing increasing interest for radiation harsh environments, due to cost, performance, and availability reasons. As a result, several studies have been performed on the radiation sensitivity of NAND Flash, concerning both TID and SEE. Unfortunately, these devices turned out to be quite sensitive to radiation and only few commercial NAND memories have been actually flown in space.

Petitioner Micron Ex-1021, 0549

As discussed in this book, NAND Flash memories feature several peripheral circuits (charge pumps, microcontroller, row decoder, page buffer) that are transparent to the final user, but critical for the device radiation sensitivity. In the 1990s, the control circuitry was recognized as the most radiation sensitive part of commercial (i.e., non-radiation hardened) devices [31–34], whereas floating gate cells were considered a minor problem. Since then, technology evolution has brought about more and more advanced devices, whose architectural complexity increased together with memory density and performance. Nowadays, as proven in several works, floating gate sensitivity is no more a secondary issue.

For instance, concerning SEE, reports in the literature on recent devices show that prompt [35] and anneal effects [36–38] in the floating gate cells, and issues in the control circuitry [36] (including transient errors [39], SEFIs [40] and sudden spikes in the current supply [41]) all combine together, forming an intricate scenario. A similar trend is observed also for TID: both the peripheral circuitry and the floating gate cells are sensitive, and, depending on the type of memory and the operating conditions during exposure, one or the other can cause failure under exposure to ionizing radiation.

Accurate assessment of radiation sensitivity and identification of root causes have become increasingly difficult tasks in state-of-the-art Flash memories. Quite often it is not easy to identify the exact failure mode during radiation testing, especially when the manufacturer is not willing to provide detailed information. Specific test procedures have been developed to address these issues. In particular, radiation exposure under different operating conditions (i.e., during standby, read loops, program loops, erase/program/read loops) can greatly help to recognize the failure mode. In addition to irradiations performed with different test loops, selective exposure to radiation of only some functional block can ease the identification of the part that is responsible for a given error or malfunction.

It is quite difficult to draw general conclusions and the reader should consider the following discussion as a catalog of possible issues encountered when these devices are exposed to radiation. Radiation sensitivity can exhibit large variations from manufacturer to manufacturer, even when nominally identical features are specified. In the following we analyze the sensitivity of NAND Flash to TID and SEE, pinpointing the role of each block, including the FG array (which will treated in detail from the physical point of view in Sect. 19.3), charge pumps, page buffer, microcontroller, and row decoder, in determining the device sensitivity.

19.2.1 Total ionizing dose effects

Total ionizing dose effects are peculiar of the space environment, or of some applications where important sources of man-made radiation exist (e.g., nuclear power plants), and are of less concern for the terrestrial environment. The failure dose of a NAND Flash memory can be defined in different ways, that are symptoms of radiation damage to a specific part of the memory: ability to program and erase, time to program, number of read errors, standby supply current, access time, etc. The maximum total dose that can be withstood by a Flash memory not

Petitioner Micron Ex-1021, 0550

only changes from device to device [39, 42], but also depends on the failure criterion and on the operating conditions during exposure. After TID exposure, Flash memories may draw excessive supply current [32], fail to read, program and/or erase [33, 39, 43], or become slower in these operations [32].

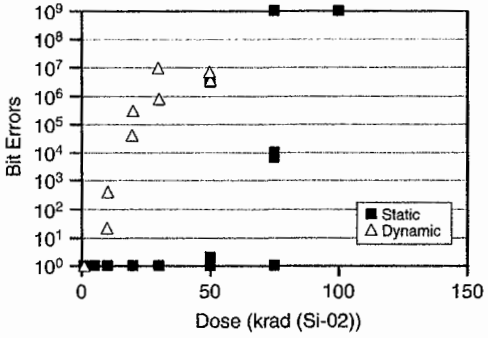


Fig. 19.5. Static and dynamic bit errors as a function of total dose [39] © IEEE, 2006

Figure 19.5 is just an example showing that under dynamic conditions (i.e., with parts exercised during exposure) the radiation sensitivity can be much larger than in static conditions (not exercised parts). The figure reports the results of static and dynamic TID tests for a 90-nm SLC NAND memory, showing that read errors in dynamic conditions start at much lower doses than in static ones [39].

In the following sections, radiation effects on the different parts of a NAND memory will be illustrated through some examples taken from the literature, showing the peculiarity of the specific damage and the conditions under which it can be observed. The list is not exhaustive, other examples can be found in the literature.

19.2.1.1 Floating gate cells

In 43 selective irradiation of the various device blocks (FG array alone, FG array together with charge pumps or row decoder) was used to understand the peculiar features of the failure mode of each block. Irradiations were performed in either 1 or 5 krad(Si) steps on idle biased samples. After each step, the devices were measured performing either a read operation or an erase/program/read loop on one or more memory blocks (previously programmed or erased). If the device was exposed to x-rays shielding all the peripheral circuit, errors were detected only in programmed cells, i.e., filled with electrons (on the contrary, the erased cells, filled with holes, do not show any errors). Figure 19.6 depicts the evolution of FG errors in programmed cells as a function of dose. Starting from 55 krad(Si), read errors with random physical locations are observed. The number of errors increases monotonically and saturates around 200 krad(Si) to the number of cells exposed to radiation; in other words, after 200 krad(Si) all exposed FGs are read as erased. These errors happen when the V_{TH} of a programmed FG changes

Petitioner Micron Ex-1021, 0551

enough to cross the user mode read level and to enter the erased cells distribution. The V_{TH} distribution moves during irradiation because of the progressive discharge of the FGs, which will be explained in details in Sect. 19.3.

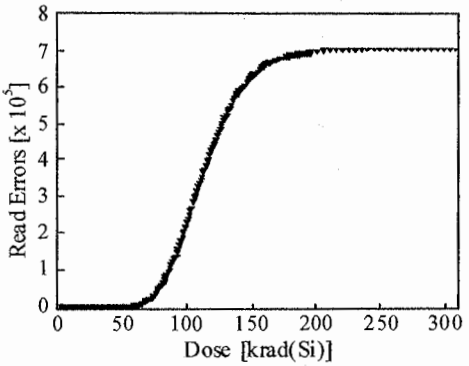


Fig. 19.6. Bit errors during progressive built-up of x-ray dose, shielding the peripheral circuitry (only FG array exposed) for a programmed block in a SLC NAND [43] © IEEE, 2009

19.2.1.2 Charge pumps

Since about 10 years ago, when the first studies on the radiation sensitivity of Flash memories were carried out, charge pumps were discovered to be the weakest part of the device, concerning TID sensitivity. In 1998, experiments on now obsolete memories, where charge pump circuitry could be disabled through an external voltage supply, showed that charge pumps were highly sensitive to TID [32, 33]. Figure 19.7 shows the time required to erase 32 blocks: in case the charge pump is active during the irradiation, after a dose of 5 krad(Si), the time to erase rapidly increases; on the contrary, disabling the charge pump, this time stays constant even after 30 krad(Si). The authors attribute the effect to radiation-induced V_{TH} shifts in the PMOS transistors used in the charge pump circuitry, causing a progressive lowering of the output voltage of the pumps.

This conclusion has been later confirmed for more scaled devices [44], where charge pumps are reported to fail in providing the sufficient power for the program voltage at doses lower than 10 krad(Si).

In more recent reports [39, 43], charge pumps are again recognized as a susceptible part during TID experiments. In [43] irradiations were performed in 10 s steps on idle samples under bias. After each step, corresponding to 1 krad(Si), the devices were measured performing either a read operation or an erase/program/read loop on one or more memory blocks (previously programmed or erased). Also Test Mode routines are used in this work to measure the output voltage generated by the program pump.

Petitioner Micron Ex-1021, 0552

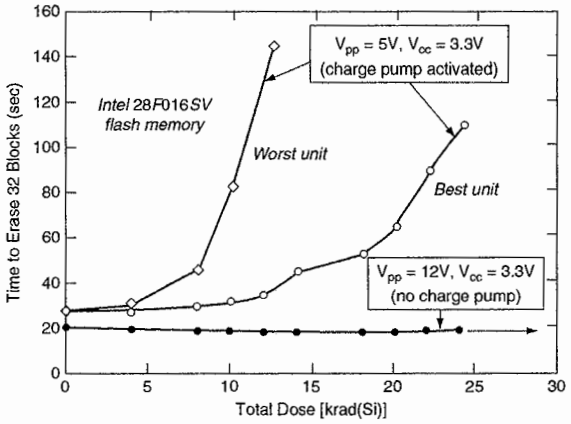


Fig. 19.7. Time to erase 32 blocks in two different conditions: device with charge pump activated and device with charge pump disabled [32] © IEEE, 1998

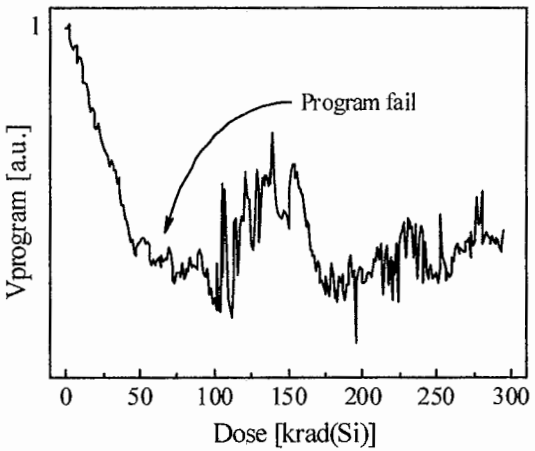


Fig. 19.8. Average voltage generated by the program charge pump during TID exposure (normalized to $V_{program}$ before irradiation) 43 © IEEE, 2009

From 100 krad(Si) on, all the memory cells are read as programmed, regardless of their initial program value. A progressive decrease of the output voltage of the program pump is reported, as shown in Fig. 19.8, together with a failure of program operation after 50 krad(Si).

This explains why read errors are observed during read loops, as a consequence of the radiation-induced voltage decrease: in fact, if the voltage provided by the charge pump is lower than designed (i.e., it is not sufficient to turn on even a single transistor in one bitline, including Source Selection Line, SSL, and Drain Selection Line, DSL, selectors), all the cells in the bitline are read as programmed (no current flows), regardless of their actual program status.

Petitioner Micron Ex-1021, 0553

The reason for the large radiation sensitivity of the charge pump is linked to the thick gate oxide and the high voltages that these circuits must use. As illustrated in Sect. 19.1, the damage introduced by TID strongly reduces when oxide layers are thinned. In addition, higher voltages cause in general more TID degradation, because a higher electric field increases the amount of charge yield, that is the amount of charge which can potentially alter the characteristic of the oxide layers.

19.2.1.3 Decoders

Another example of building block that can fail under exposure is the Row Decoder (RD). In [43], the memory was irradiated with x-rays shielding the microcontroller and charge pumps and leaving the RD exposed. Figure 19.9 illustrates the number of read errors found in programmed and erased blocks, respectively, during x-ray irradiation, exposing to radiation only the RD and a portion of the FG array. Read operations were performed, as described in previous paragraphs, after 1 or 5 krad(Si) steps on idle samples under bias. At the beginning of the irradiation and until 150 krad(Si), a gradual increase in the number of raw bit errors is observed. This behavior is due to the progressive discharging of FG programmed cells, and is similar to that observed exposing to radiation the charge pumps and the FG array (Sect. 19.2.1.1). Then, around 150 krad(Si), a sudden drop in the number of errors is observed for programmed cells, as reported in Fig. 19.9a.

Concerning the erased block, until 150 krad(Si) the situation is similar to what happens when all the control circuitry is shielded (no errors are detected). On the contrary, an increase in the number of read errors is observed for erased cells after 150 krad(Si). The abrupt fall and rise of the errors observed in programmed and erased cells, respectively, is due to damage of the RD. The fact that all the cells are read as programmed indicates that some transistors that should be turned on are actually off, or that leakage too rapidly discharges some nodes that were pulled high, thus impeding current to flow through the bitline.

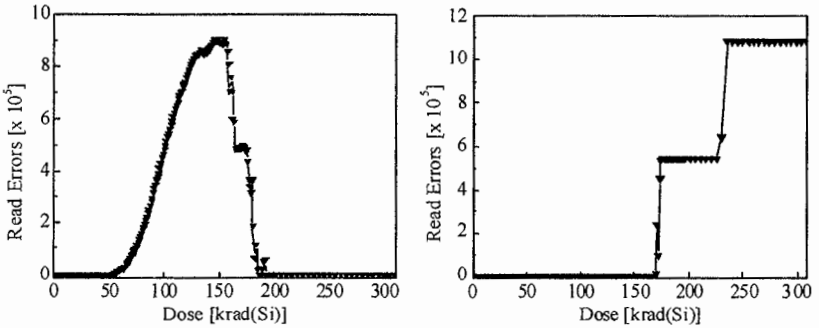


Fig. 19.9. Number of read errors during x-ray exposure of FG array and row decoder as a function of x-ray dose, for a block that was programmed (a) and a block that was erased (b) prior to irradiation 43 © IEEE, 2009

Petitioner Micron Ex-1021, 0554

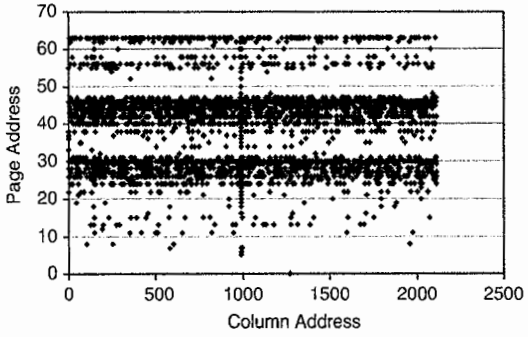


Fig. 19.10. Physical map of bit errors after exposure to 75 krad($Si0_2$) Co^{60} gamma source [39] © IEEE, 2006

Also the observed failure in one half-plane and then in the other (step-like behavior in Fig. 19.9b) is highly indicative of damage in the local RD controlling each of the two FG matrix half planes.

Other examples can be found in the literature describing radiation damage in parts of the control circuitry that may be identified with the RD. In [39], where the memories were irradiated without shields, rows of bad bits have been observed and attributed to damage in the addressing control circuitry (see Fig. 19.10).

19.2.2 Single event effects

Single event effects occur with high frequency in space, but are possible also at sea level (although to a lesser extent), due to atmospheric neutrons and alpha particles. The study of SEEs can be even more complex than TID characterization. In fact, a plethora of different events can be triggered by the passage of a single particle: charge loss from FG cells, SEUs in the page buffer, SEFIs due to strikes in the microcontroller, generating inconsistent states, etc. Each of them can be characterized by a specific cross section, which expresses the probability of occurrence of a particular type of effect. Like with TID, large sensitivity variations are possible from manufacturer to manufacturer, etc.

19.2.2.1 Floating gate cells and page buffer

When a wandering particle hits the floating gate cells or the page buffer latches, single bit errors can occur. Figure 19.11 illustrates an experiment where a memory is periodically read under irradiation, after having been written with a given program pattern (typically all zeroes or checkerboard) prior to exposure. Different errors are reported [36, 39, 41, 42, 44] in these conditions. Initially raw bit errors are observed. Some of them appear only in one read cycle and they disappear in the following one (these events are usually called dynamic errors); some others

Petitioner Micron Ex-1021, 0555

(called static errors), on the contrary, are present from a given read on and they remain even after a reset or a power cycle, disappearing only after reprogramming the device. In a few cases, sudden bursts of errors (SEFI) are detected that usually end up with a non operating device, until a reset or a power cycle is performed [36]. Dynamic errors can be traced back to upsets in the Page Buffer (PB) latches, that are susceptible to SEU, as extensively reported in literature for SRAM cells and discussed in Sect. 19.1.4. We recall that the PB is a temporary location where data are stored after being read from a page of the FG array and before being output to the device pins. To measure the PB sensitivity, one can introduce a delay between the "read" command and the transfer of the data from the memory to the output pins. In this way the data are "frozen" in the PB for an arbitrarily long time and so they are susceptible to corruption.

Figure 19.11 shows an example of the scenario we described above for a NAND irradiated with heavy-ions. Static errors (FG errors) and dynamic errors (PB errors) are observed, their rate depending on many factors, among which the speed at which the memory is read, the pattern stored in the cells (PB latches may be asymmetric), the LET of the impinging ions, etc. A SEFI is also observed in the graph, which gives rise to a burst of errors as we describe in detail in Sect. 19.2.2.2.

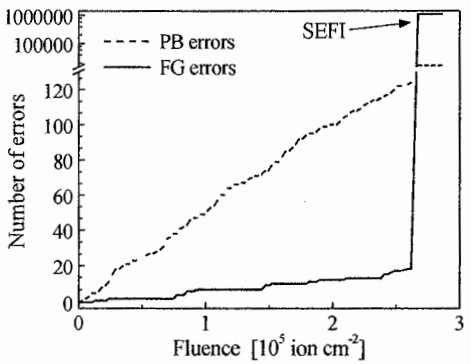


Fig. 19.11. Example of page buffer and floating gate errors build-up as function of fluence during periodic read of a memory block under Bromine irradiation [36] © IEEE, 2009

19.2.2.2 Single event functional interruptions

In a Flash memory, one of the elements that are susceptible to SEE is the microcontroller that manages the operations of the device. A SEFI occurs when an ion hit causes the microcontroller to enter a possibly inconsistent state, bringing the device to an indefinite condition. This may cause the memory to experience read errors in whole pages or blocks, to suddenly halt, to unexpectedly go in a test mode status, and many others. With the lighter ions, the greatest part of SEFIs usually occurs during program and erase operations, when the device micro-controller is used more heavily, and the applied voltages reach their maximum

Petitioner Micron Ex-1021, 0556

values. On the other hand, during irradiation with the heavier ions, functional interruptions are observed during read operations as well [36].

SEFIs can also be classified depending on the way the functionality can be restored after the event: in some cases, a power cycle is necessary to recover the device from the error condition, in some other cases a reset command is enough, and in others proper functioning may even be spontaneously restored after a few seconds without any intervention. Table 19.1 gives an idea of these aspects. Figure 19.12 is an example of how the occurrence of SEFIs varies depending on the ion LET for a NAND Flash irradiated with heavy ions.

Table 19.1. Functional interruptions observed during heavy ion irradiation for a 16-Mb Flash Samsung NAND [31] © IEEE, 1997.

Error type	Description	Recovery method
Row or column flips	Same values appear in multiple locations for the same bit position	Reinitialize and rewrite
Lockups (self clearing)	Inability to progress beyond read, write, or block clear modes	Reprogramming (erase/write/read): power cycle not required
Lockups (non-clearing)	Inability to exit read, write, or block clear modes	Power cycle followed by reprogramming
Stuck bits	Small number of bits permanently altered	None (permanent effect)

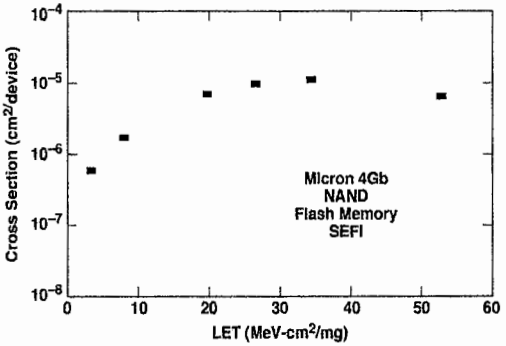


Fig. 19.12. Example of SEFI cross section as a function of LET for a NAND Flash irradiated with heavy ions [41] © IEEE, 2007

19.2.2.3 Power supply current spikes

Another phenomenon that was recently observed in some NAND samples during heavy-ion exposure is the occurrence of spikes in the power supply current. An example of this is depicted in Fig. 19.13, which refers to a device irradiated with Gold in stand-by mode. As shown in the graph, current spikes in the order of hundreds of milliampere are detected during exposure. These spikes are usually

Petitioner Micron Ex-1021, 0557

spontaneously restored with no intervention. Interestingly, this phenomenon may be observed only for some manufacturers and it may be destructive or not, even in devices produced by the same manufacturer with the same part number. The physical mechanisms underlying these current spikes has not been precisely identified, even though some evidence exists on the role of charge pumps circuitry in determining it.

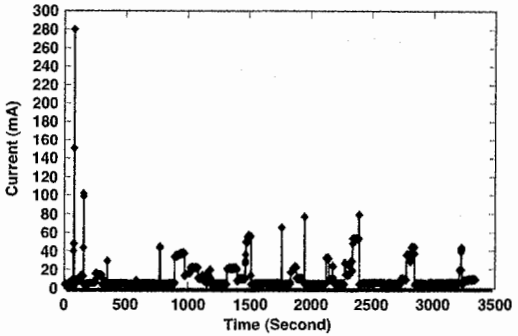


Fig. 19.13. Standby current detected in a MLC NAND memory during irradiation with Au [41] © IEEE, 2007

19.2.2.4 Overall cross section

Now that we have presented the main single event effects that a NAND memory may experience during irradiation, we will show an example of how the different contributions, such as the page buffer and the floating gate array, can determine the overall device sensitivity. As briefly mentioned before, the relative importance of each contribution strongly depends on the operating condition during exposure. Figure 19.14 compares the floating gate cross section with the page buffer cross section, in a 90-nm NAND irradiated with heavy ions, as a function of LET. Both PB and FG cross section feature the typical Weibull behavior. As seen in the figure, the FG array has a larger threshold LET and a lower saturation cross section per bit than the PB, but we have to remark that it contains many more bits compared to PB latches.

To assess the overall device cross section, we have to weigh these two contributions (neglecting SEFIs). Depending on the ion LET, time spent reading the device, and pattern, the sensitivity can be dominated either by the FG array or by the PB. Large variations (up to several orders of magnitude) can occur as a function of these parameters.

Just to give a few examples, let us consider the best-case scenario in terms of overall heavy-ion cross section. If the memory is fully erased and sporadically read, the FG array is not sensitive (being erased) and the susceptibility of the PB is very low (being scaled by the read activity).

Petitioner Micron Ex-1021, 0558

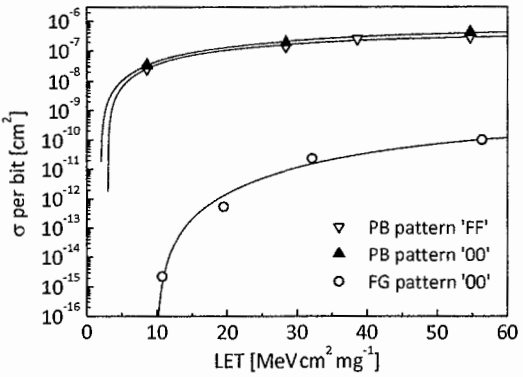


Fig. 19.14. Weibull curve illustrating the heavy-ion bit cross section for page buffer and floating gate array in a 90-nm NAND Flash [36] © IEEE, 2009

On the other hand, we can imagine the worst-case condition as the memory fully programmed, continuously and entirely accessed: in this case, the sensitivity of both the FG array and the PB latches reach their maximum value, and which of the two dominates strongly depends on the ion LET.

19.3 Radiation effects on floating gate cells

In this section we will cover the effects of ionizing radiation on the FG array. For its largest part, this section refers to the work of our research group and is based on data not usually available to the user. In fact, "user mode" routines give the information stored in a FG cell in digital terms, that is, as '0'or '1'in a SLC device or in a slightly more complex way for MLC devices. Of course, the stored information by itself is the threshold voltage V_{TH}, and is almost analog in nature, being its granularity linked to the electron charge [45]. Results presented in this section were obtained by working in close collaboration with the manufacturers and by accessing the V_{TH} of the FGs in the array. Other than for the basic physics aspects, such results are essential to understand the complex failure behaviors found in more recent technologies and outlined in this section, and also to forecast what's behind the corner. Note that part of the results presented here were obtained with NOR memories; however, they adapt with small or no modification to the case of NAND since the device architecture is not involved [46, 47].

19.3.1 Total ionizing dose

The effect of a TID irradiation on an FG is its progressive discharge as dose accumulates. This is illustrated in Fig. 19.15, reporting the V_{TH} probability densities for FG arrays programmed in the '0' and '1' state before and after

Petitioner Micron Ex-1021, 0559

different 10-keV x-ray TID levels [48–50]. Before irradiation, V_{TH} distributions closely resemble the expected Gaussian shape. After irradiation, the threshold voltage of all FGs programmed in the '0' state ('high' V_{TH}, corresponding to electrons stored in the FG) uniformly moves toward lower V_{TH}: the FGs progressively lose electrons because of irradiation. Conversely, devices programmed in the '1' state ("low" V_{TH}, holes stored in the FG) experience a V_{TH} increase, that is, a positive charge loss from the FG. The V_{TH} shift (ΔV_{TH}) is lower for the '1' than that for the '0' state, because of the much lower net charge stored in the FG in that condition. Distributions rigidly shift during irradiation since TID radiation effects are uniformly distributed over the chip area [10].

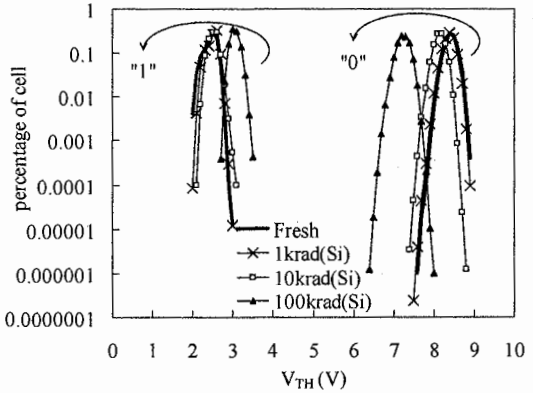


Fig. 19.15. Probability densities of FG MOSFET threshold voltage V_{TH} before and after irradiation at 1, 10, and 100 krad(Si), with 10-keV x-rays 51 © IEEE, 2004

A physics model to account for these effects has been first developed by Snyder et al. in 1989 [48] and further developed in recent times to account for the changes in the technology happened in almost 20 years [51, 52]. Briefly, charge is generated in all oxides surrounding the FG. Depending on the oxide electric field, part of these carriers recombine; after recombination, carriers thermalize, then electrons are quickly swept away from the oxide thanks to their high mobility, while holes slowly move toward the FG and may either reach the FG, where they recombine part of the stored negative charge (mechanism (1) in Fig. 19.16), or they are trapped in the oxide (mechanism (2) in Fig. 19.16). In both cases, the net effect is a reduction of the cell V_{TH}. Further, electrons may acquire enough energy from the incoming radiation [53] to jump over the oxide barrier (mechanism (3) in Fig. 19.16). There are several aspects which need to be considered when adapting this model to modern devices, including:

– The above description of different phenomena occurring in a strict sequential order is not realistic in the tunnel oxide of FG devices, since recombination, thermalization, and electron transit times are of the same order of magnitude in the range of the tunnel oxide thickness (≤10 nm) proper of modern devices.

Petitioner Micron Ex-1021, 0560

- The interpoly dielectric is an ONO sandwich in all modern devices, not a SiO_2 layer as in [48], thus the different dynamics of charge generation and trapping in the nitride layer has to be carefully considered [54, 55].
- The lateral dimensions of the FGs cannot be neglected.

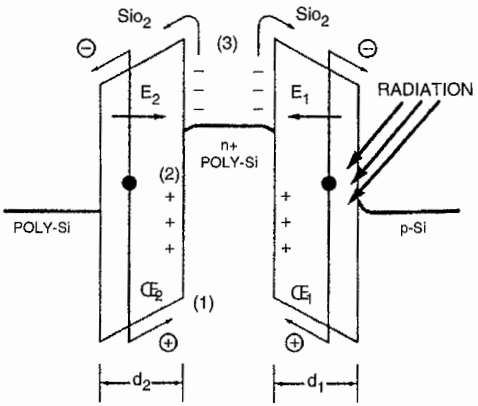


Fig. 19.16. Main contributions to threshold voltage degradation in a FG cell during TID experiments [48] © IEEE, 1989

When these aspects are taken into account, proper modeling can accurately describe the average V_{TH} shift as a function of dose, as shown in Fig. 19.17 [47, 52]. In this figure, two data sets are reported. The first one was obtained with ^{60}Co -γ rays, and the second with 10-keV x-rays. It is evident that the degradation induced by x-rays is by far larger, likely due to dose enhancements effects. In fact, 1.25 MeV photons generate high energy Compton electrons, whereas low energy photons (<50 keV) generated by the x-ray source interact with the material through the photoelectric effect, which is enhanced at the Si/SiO_2 interface [10], [56, 57].

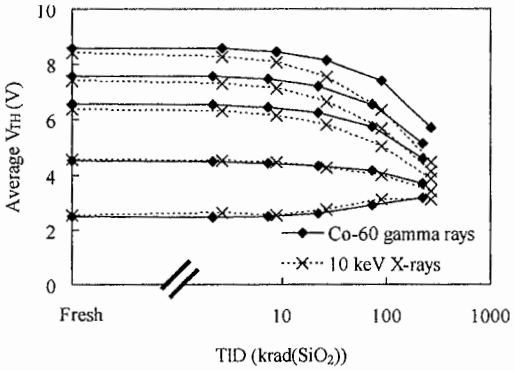


Fig. 19.17. Average V_{TH} of sectors of FG devices irradiated with 10-keV x-rays and with ^{60}Co γ-rays as a function of total dose [47] © IEEE, 2007

Petitioner Micron Ex-1021, 0561

This effect is particularly effective in presence of interfaces between high-Z and low-Z materials, which is what often happens in NAND memories, using refractory materials such as titanium, tungsten, or cobalt. On the other side, discharges obtained after irradiation with protons with low-to-average energies perfectly superimpose to results from ^{60}Co [47, 52]. Yet, very high energy protons can trigger nuclear reactions, resulting in a behavior which is somewhat intermediate between the TID and the SEE [58].

19.3.2 Single event effects

The effect of single ions on the FG array is, as expected, different from that of TID since the energy is released by each ion in a small volume. Hence, every single ion can result in a large but localized effect (SEE). As an example, a chip with all FG programmed at '1' was irradiated with $2x10^7$ I ions/cm^2 [59]. Figure 19.18 shows the V_{TH} variation over a small part of the irradiated array. Each square represents a single FG, featuring a gray level related to the ΔV_{TH} shift: the darker the square, the larger the threshold variation, which is always negative (V_{TH} decreases). The FGs with corrupted information are randomly distributed over the chip surface, and their number roughly corresponds to the number of FGs being hit by ions, meaning that every time a single FG is hit by a single ion, it experiences a net charge loss, resulting in a V_{TH} shift [35, 59–61].

A more informative picture of these same data can be obtained by looking at the statistical description of V_{TH} (Fig. 19.19): before irradiation, V_{TH} distribution resembles the expected Gaussian shape, but after irradiation, a secondary peak appears, corresponding to the FGs hit by ions. Hence, the distance between the main and the secondary peak can be taken as the average V_{TH} shift experienced by hit cells, $<\Delta V_{TH}>$. This parameter is reported as a function of the pre-rad tunnel oxide electric field and of the ion LET in Fig. 19.20. At least for ions having a relatively low energy, such as those shown in Fig. 19.20, both relationships are almost perfectly linear.

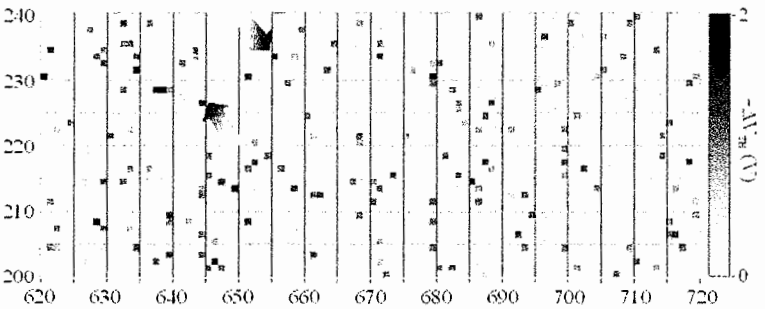


Fig. 19.18. Spatial distribution of V_{TH} for a small subset of cells from a chip after 2×10^7 iodine ions/cm^2. The gray scale (right) indicates the amount of ΔV_{TH}; the two cells indicated by arrows have $\Delta V_{TH} > 2$ V. Each square represents a FG [59] © IEEE, 2001

Petitioner Micron Ex-1021, 0562

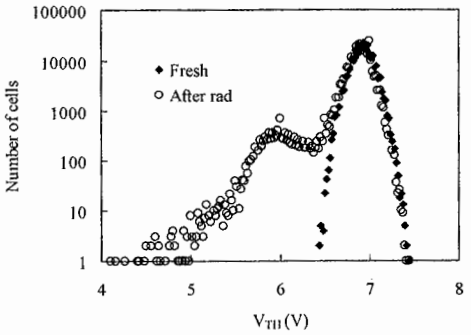


Fig. 19.19. Probability density of threshold voltages before and after irradiation with 2×10^7 iodine ions/cm^2 [59] © IEEE, 2001

Such a linear relationship allows excluding a model similar to that developed for TID effects, that is, one based on charge generation, recombination, and transport. In fact, all these phenomena have a strongly non-linear dependence on the electric field [35]. Further, the amount of charge lost for FGs in Fig. 19.20 exceeds that calculated for those same ions with a generation/recombination/transport model by a factor of more than 20 [35, 61].

One of the model currently available to explain these results is based on the idea that the dense electron/hole plasma generated in the tunnel oxide by the incoming ion acts like a resistance, thus promptly discharging the FG [35, 61]. While the value of this resistance is linked to the ion LET (heavier ions result in denser track, hence in lower resistance and in larger discharge), the duration of the path is linked to the presence of electrons in the FG. In fact, since $<\Delta V_{TH}>$ linearly depends on the number of generated electrons/holes, not on those surviving recombination, it follows that charge loss happens before recombination.

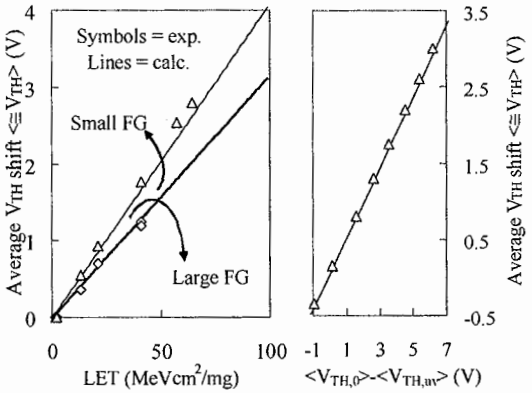


Fig. 19.20. Dependence of $<\Delta V_{TH}>$ on the LET (left) and electric field across the tunnel oxide (right). Experimental data are reported as symbols and calculations as lines [61] © IEEE, 2004

Petitioner Micron Ex-1021, 0563

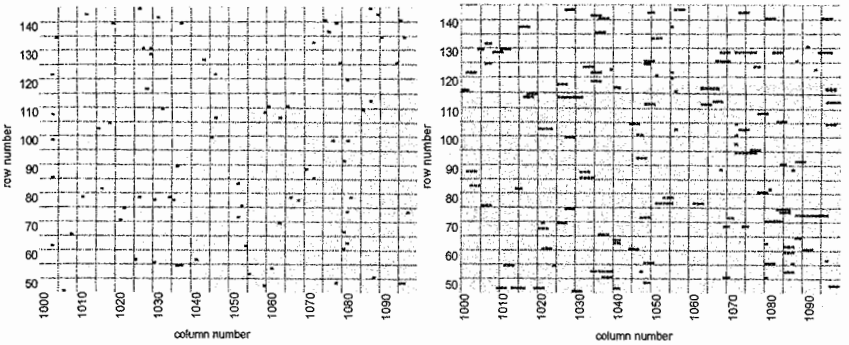


Fig. 19.21. Maps of the distribution of "yellow" and "red" FGs above small chip areas, for devices irradiated with 1217-MeV Xe at different angles: (a) normal incidence vs. (b) 75°. See text and [63] for details on color code © IEEE, 2007.

The time scale of the phenomena involved actually allows going one step further, that is, to relate the "on time" of conductive path to the thermalization of electrons [35]. Thus, the FG discharge is believed to happen in about 10 fs [35]. Lines in Fig. 19.20 are obtained with this model. Note in particular that the average $<\Delta V_{TH}>$ depends on the FG area, being larger for smaller FGs.

As will be clearer later on, this explains why user mode errors in the array were not detected in old generation Flash memories ($<\Delta V_{TH}>$ was too small to result in errors). Things get even more complicated when dealing with aggressively scaled devices. For example, in most modern technologies, some charge loss happens even from FGs which are just grazed, not hit by ions [62]. This phenomenon is not covered in details here since a full model is not yet available; however, it is worth noting that these "grazed" FGs have peculiar characteristics which allow us to separate them from their "hit" counterparts. Also, in any real world environment (be it space, high energy physics, or simply the atmosphere) particles do not cross the devices normally to the silicon surface. It is not totally surprising but of great interest the fact that, at grazing angles, a single ion can lead to the formation of a "tail" of FGs with corrupted information (Fig. 19.21) [63]. This has major implications from the reliability standpoint, for basic physics (it allows studying the ion track structure [63]), and for more applied physics (the widely used cosine law [64, 65] does not apply to Fig. 19.21).

19.3.3 Long-term effects

The discussion carried on so far leaves open an important point, that is, if the FGs hit by ions are permanently damaged or not [66–69]. FGs being hit by single ions can be easily identified thanks to their large post-irradiation ΔV_{TH}. If the hit cells are programmed again, one obtains the results shown in Fig. 19.22 [67]. Just one hour and half after programming the memory array, a tail appears extending down

Petitioner Micron Ex-1021, 0564

to less than 7 V due to FGs changing their V_{TH} and leaking charge. This charge loss should be related to the ion impact, since no similar effect was observed in non-hit FGs [66, 67]. The tail increases with time, due to the progressive discharge of those FGs which were already leaking charge 90 min after programming, and to new FGs featuring lower leakage current, which progressively add to the tail of the distribution.

This phenomenon derives from the onset of the RILC [19, 66, 68] and is a multi-Trap Assisted Tunneling (m-TAT) conduction mechanism through electrically active defects created, directly or indirectly, by the ion impact. While most studies agree that the electrically active defects are the product of positive charges generated by the irradiation [10, 70–73], in this case the damage does not appear to scale with the pre-irradiation electric field [67], so that the actual physical mechanism leading to the creation of defects is not clear. A possible explanation, linked to the non-ionizing energy loss, has been recently proposed [74]. On the other hand, the resulting defects have a close affinity with those generated by electrical stress, such as those responsible for SILC, as evidenced by their recovery after being exposed to Forming Gas Anneal (FGA) after irradiation and before the electron retention experiment [67]. In fact, neutral traps responsible for parasitic conduction phenomena such as SILC and RILC across thin oxide layers [19] are known to anneal at relatively high temperatures [75, 76].

As expected, the resulting localized conductive path becomes more and more conductive when increasing the impinging ion LET. Despite the large effect on the FG cell V_{TH}, the average current due to this defect distribution is too small to be directly measured. In fact, with only few thousands electrons in a programmed FG, a leakage current of just 1 fA would completely discharge the FG in about 1 s!

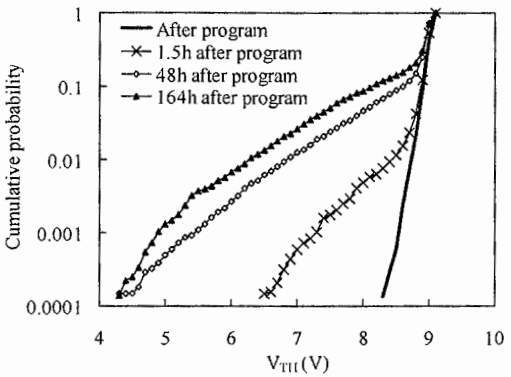


Fig. 19.22. Cumulative distribution of V_{TH} for FGs hit by iodine ions and re-programmed [67] © IEEE, 2005

19.3.4 Atmospheric neutrons

Atmospheric neutrons are a known source of soft errors in static [77] and dynamic [78] CMOS memories. Several memories have been tested with both monoenergetic

Petitioner Micron Ex-1021, 0565

[79] and spallation sources [80, 81]. The latter type of source mimics the energy spectrum of atmospheric neutrons [2]. In both cases, several single bit errors were found in the arrays, indicating that the number of excess electrons/holes stored in the FG can be corrupted by neutrons. This aspect is particularly important, despite the relatively low failure probability, because atmospheric neutrons are everywhere, so one has to take care of the huge amount of extremely dense devices used in everyday applications, instead than of a few (expensive) devices used in a niche application. A closer look at the V_{TH} distribution gives precious insights into the physical mechanism leading to such errors (Fig. 19.23). Since the main distribution does not move due to neutron irradiation, but a tail forms, from the previous discussion it is clear that neutron interactions with the device nuclei leads to recoils or nuclear reaction ionizing by-products which can cross a FG and partially discharge it.

Based on previous discussion on heavy ion effects, several peculiar aspects of neutron-induced errors can be explained, including the larger sensitivity of MLC devices with respect to SLC [80, 81], the larger sensitivity of program states corresponding to larger V_{TH}, and the increasing sensitivity with technology scaling. For this reason, neutron effects need to be taken into consideration when forecasting reliability of future devices, even if the failure rate is still within the correcting capabilities of ECC algorithms.

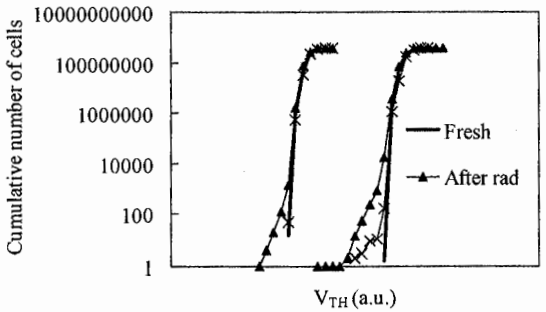


Fig. 19.23. V_{TH} cumulative distribution for devices irradiated with neutrons mimicking the atmospheric spectrum [81] © IEEE, 2008

19.4 Conclusions

Ionizing radiation effects are a serious threat to the reliability of electronic devices, especially in space, but also on Earth. There are two broad categories of radiation effects in CMOS circuits that can cause either parametric degradation (total ionizing dose effects), related to the build-up of defects and trapped charge in insulating layers; or stochastic errors, connected with the passage of a single particle (single event effects) in the wrong place, at the wrong time.

Petitioner Micron Ex-1021, 0566

NAND Flash memories are multifaceted devices, whose response to ionizing radiation is complex and has not been completely understood. In older technologies, the sensitivity (both to total ionizing dose and single event effect) mainly aroused from the peripheral circuitry, whereas in more modern devices, the floating gate array also plays a significant role.

Total ionizing dose induces parametric degradation especially in the charge pumps, where thick oxides and high voltages are used, but can also cause the discharge of floating gate cells at doses lower than those required for the failure of the peripheral circuitry. Flash memory radiation sensitivity has not benefited from Moore's law, as much as low-voltage CMOS logic did, because the program and erase voltages can be scaled with much more difficulty (or not at all). As long as this will remain true, commercial NAND Flash memories will hardly reach failure doses higher than a few tens/hundreds of krads.

Single event effects may result in the corruption of the data stored in the array, due to the discharge of floating gate cells; dynamic errors, because of bit-flips in the page buffer; functional interruptions, when ionizing particles strike the embedded microcontroller forcing it in an inconsistent state; and excessive current draw. Scaling of the feature size gate exacerbates the effects of charge loss from the floating gate due to impinging particles, decreasing more and more the minimum linear energy transfer needed to produce an error. In the past, these effects were a concern only in harsh environments, such as space; however, due to scaling, NAND floating gate cells have become so tiny, that even the indirect charge deposition caused by atmospheric neutrons can corrupt the stored information. Luckily, the raw bit error probability at sea level is small enough that it can be easily handled by current error correction code schemes.

References

1. A. Holmes-Siedle, L. Adams, "Handbook of radiation effects" 2nd edition, Oxford University Press, Oxford, 2002
2. JEDEC standard JESD-89A, available on line at: www.jedec.org/download/search/JESD89A.pdf
3. J.L. Barth, C.S. Dyer, E.G. Stassinopoulos, "Space, atmospheric, and terrestrial radiation environments," IEEE Transactions on Nuclear Science, vol.50, no.3, p.466, 2003
4. G.A. Ausman, F.B. McLean, "Electron-hole pair creation energy in SiO2," Applied Physics Letters, vol.26, p.173, 1975
5. J.M. Benedetto, H.E. Boesch, Jr., "The relationship between Co60 and 10 keV X-ray damage in MOS devices," IEEE Transactions on Nuclear Science, vol.33, p.1318, 1986
6. C.M. Dozier, D.B. Brown, "Effects of photon energy on the response of MOS devices," IEEE Transactions on Nuclear Science, vol.28, no.6, p.4137, 1981
7. T.R. Oldham, J.M. McGarrity, "Comparison of Co and 10 keV X-ray response in MOS capacitors," IEEE Transactions on Nuclear Science, vol.30, p.4377, 1983
8. T.R. Oldham, "Recombination along the tracks of heavy charged particles in SiO films," Journal of Applied Physics, vol.57, p.2695, 1985

Petitioner Micron Ex-1021, 0567

9. K.P. Rodbell, D.F. Heidel, H.H.K. Tang, M.S. Gordon, P. Oldiges, C.E. Murray," Low-energy proton-induced single-event-upsets in 65 nm node, silicon-on-insulator, latches and memory cells," IEEE Transactions on Nuclear Science, vol.54, no.6, p.2474, 2007
10. T.P. Ma, P.V. Dressendorfer, "Ionizing radiation effects in MOS devices and circuits," Wiley, New York, 1989
11. M.R. Shaneyfelt, J.R. Schwank, D.M. Fleetwood, P.S. Winokur, K.L. Hughes, F.W. Sexton, "Field dependence of interface trap buildup in polysilicon and metal gate MOS devices," IEEE Transactions on Nuclear Science, vol.37, no.6, p.1632, 1990
12. N.S. Saks, C.M. Dozier, D.B. Brown, "Time dependence of interface trap formation in MOSFET's following pulsed irradiation," IEEE Transactions on Nuclear Science, vol.35, p.1168, 1988
13. A.H. Johnston, "Super recovery of total dose damage in MOS devices," IEEE Transactions on Nuclear Science, vol.31, p.1427, 1984
14. D.M. Fleetwood, F.V. Thome, S.S. Tsao, P.V. Dressendorfer, V.J. Dandini, J.R. Schwank, "High temperature silicon on insulator electronics for space nuclear power systems: Requirements and feasibility," IEEE Transactions on Nuclear Science, vol.35, no.5, p.1099, 1988
15. J.R. Schwank, P.S. Winokur, P.J. McWhorter, F.W. Sexton, P.V. Dressendorfer, D.C. Turpin, "Physical mechanisms contributing to device rebound," IEEE Transactions on Nuclear Science, vol.31, no.6, p.1434, 1984
16. J.M. McGarrity, "Considerations for hardening MOS devices and circuits for low radiation doses," IEEE Transactions on Nuclear Science, vol.27, p.1739, 1980
17. N.S. Saks, M.G. Ancona, J.A. Modolo, "Radiation effects in most capacitors with very thin oxides at 80 K," IEEE Transactions on Nuclear Science, vol.31, no.6, p.1249, 1984
18. A. Scarpa, A. Paccagnella, F. Montera, G. Ghibaudo, G. Pananakakis, G. Ghidini, P.G. Fuochi, "Ionizing radiation induced leakage current on ultra-thin gate oxides," IEEE Transactions on Nuclear Science, vol.44, no.6, p.1818, 1997
19. M. Ceschia, A. Paccagnella, A. Cester, A. Scarpa, G. Ghidini, "Radiation induced leakage current and stress induced leakage current in ultra-thin gate oxides," IEEE Transactions on Nuclear Science, vol.45, p.2375, 1998
20. L. Larcher, A. Paccagnella, M. Ceschia, G. Ghidini, "A model of Radiation Induced Leakage Current (RILC) in ultra-thin gate oxides," IEEE Transactions on Nuclear Science, vol.46, p.1553, 1999
21. M.R. Shaneyfelt, P.E. Dodd, B.L. Draper, and R.S. Flores, "Challenges in hardening technologies using shallow-trench isolation," IEEE Transactions on Nuclear Science, vol.45, no.6, p.2584, 1998
22. N. Bhat, J. Vasi, "Interface-state generation under radiation and high-field stressing in reoxidized nitrided oxide MOS capacitors," IEEE Transactions on Nuclear Science, vol.39, no.6, p.2230, 1992
23. H. Park, S.K. Dixit, Y.S. Choi, R.D. Schrimpf, D.M. Fleetwood, T. Nishida, S.E. Thompson, "Total ionizing dose effects on strained HfO_2-based nMOSFETs," IEEE Transactions on Nuclear Science, vol.55, p.2981, 2008
24. P.E. Dodd, L.W. Massengill, "Basic mechanisms and modeling of single-event upset in digital microelectronics," IEEE Transactions on Nuclear Science, vol.50, no.3, p.583, 2003

Petitioner Micron Ex-1021, 0568

25. F.W. Sexton, "Destructive single-event effects in semiconductor devices and ICs," IEEE Transactions on Nuclear Science, vol.50, no.3, p.603, 2003
26. L.W. Massengill, M.S. Reza, B.L. Bhuva, T.L. Turflinger, "Single-event upset cross-section modeling in combinational CMOS logic circuits," Journal of Radiation Effects, Research and Engineering, vol.16, no.1, 1998
27. S.E. Diehl, J.E.Vinson, B.D. Shafer, and T.M. Mnich, "Considerations for single event immune VLSI logic," IEEE Transactions on Nuclear Science, vol.30, p.4501, 1983
28. P.E. Dodd, M.R. Shaneyfelt, J.A. Felix and J.R. Schwank, "Production and propagation of single-event transients in high-speed digital logic ICs," IEEE Transactions on Nuclear Science, vol.51, no.6, p.3278, 2004
29. R.C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies", IEEE Transactions on Device and Materials Reliability, vol.5, no.3, p. 305, 2005
30. D. Radaelli, H. Puchner, S. Wong and S. Daniel, "Investigation of multi-bit upsets in a 150 nm technology SRAM device," IEEE Transactions on Nuclear Science, vol.52, p.2433, 2005
31. H.R. Schwartz, D.K. Nichols, A.H. Johnston, "Single-event upset in Flash memories," IEEE Transactions on Nuclear Science, vol.44, no.6, p.2315, 1997
32. D.N. Nguyen, C.I. Lee, A.H. Johnston, "Total ionizing dose effects on Flash memories," IEEE Radiation Effect Data Workshop 1998, p.100
33. D.N. Nguyen, S.M. Guertin, G.M. Swift, A.H. Johnston, "Radiation effects on advanced Flash memories," IEEE Transactions on Nuclear Science, vol.46, no.6, p. 1744, 1999
34. D.R. Roth, J.D. Kinninson, B.G. Karlhuff, L.R. Lander, G.S. Bognaski, K. Chao, G.M. Swift, "SEU and TID testing of the Samsung 128 Mbit and the Toshiba 256 Mbit Flash memory," IEEE Radiation Effect Data Workshop 2000, p.96
35. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, "Sub-picosecond conduction through thin SiO_2 layers triggered by heavy ions," Journal of Applied Physics, vol.99, no.7, p.074101, 2006
36. M. Bagatin, S. Gerardin, G. Cellere, A. Paccagnella, A. Visconti, S. Beltrami, R. Harboe-Sørensen, A. Virtanen, "Key Contributions to the Cross Section of NAND Flash Memories Irradiated with Heavy Ions," IEEE Transactions on Nuclear Science, vol.55, no.6, p.3302, 2009
37. S. Guertin, D. Nguyen, J. Patterson, "Microdose Induced Data Loss on Floating Gate Memories," IEEE Transactions on Nuclear Science, vol.53, no.6, p.3518, 2006
38. H. Schmidt, D. Walter, M. Brüggerman, F. Gliem, R. Harboe-Sørensen, P. Roos, "Annealing of static data errors in NAND-Flash memories," Proceedings of RADECS 2007, p.224
39. T.R. Oldham, R. Ladbury, M. Friendlich, H. Kim, M. Berg, T. Irwin, C. Seidleck, K. LaBel, "SEE and TID Characterization of an Advanced Commercial 2Gbit NAND Flash Nonvolatile Memory," IEEE Transactions on Nuclear Science, vol.53, no.6, p.3217, 2006
40. H. Schmidt, D. Walter, M. Brüggerman, F. Gliem, R. Harboe-Sørensen, A. Virtanen, "Heavy ion SEE studies on 4-Gbit NAND-Flash memories," Proceedings of RADECS 2007, p.632
41. F. Irom, D. Nguyen, "Single Event Effect Characterization of High Density Commercial NAND and NOR Nonvolatile Flash Memories," IEEE Transactions on Nuclear Science, vol.54, no.6, p.2547, 2007
42. T. Langley, P. Murray, "SEE and TID test results of 1Gb Flash memory," IEEE Radiation Effect Data Workshop 2005, p.58

Petitioner Micron Ex-1021, 0569

43. M. Bagatin, G. Cellere, S. Gerardin, A. Paccagnella, A. Visconti, S. Beltrami, "TID sensitivity of NAND Flash memory building blocks," IEEE Transactions on Nuclear Science, vol.56, no.4, p.1909, 2007
44. D.N. Nguyen, L.F. Scheick, "TID, SEE and radiation induced failures in advanced Flash memories," IEEE Radiation Effect Data Workshop 2003, p.18
45. C. Compagnoni, A. Spinelli, R. Gusmeroli, A. Lacaita, S. Beltrami, A. Ghetti, A. Visconti, "First evidence for injection statistics accuracy limitations in NAND Flash constant-current Fowler-Nordheim programming," IEEE International Electron Device Meeting 2007, p.165
46. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, S. Beltrami, "Single event effects in NAND Flash memory arrays," IEEE Transactions on Nuclear Science, vol.53, no.4, p.1813, 2006
47. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, S. Beltrami, J. Schwank, M. Shaneyfelt, Philippe Paillet, "Total ionizing dose effects in NOR and NAND Flash memories," IEEE Transactions on Nuclear Science, vol.54, no.4, p.1066, 2007
48. E.S. Snyder, P.J. McWhirter, T.A. Dellin, J.D. Sweetman, "Radiation response of floating gate EEPROM memory cells," IEEE Transactions on Nuclear Science, vol.36, no.6, p.2131, 1989
49. G. Cellere, A. Paccagnella, S.Lora, A. Pozza, G. Tao, A. Scarpa, "Charge loss after ^{60}Co irradiation on Flash arrays," IEEE Transactions on Nuclear Science, vol.51, no.5, p.2912, 2004
50. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, "Ionizing radiation effects on floating gates," Applied Physics Letters, vol.85, no.3, p.485, 2004
51. G. Cellere, A. Paccagnella, A. Visconti, P. Caprara, M. Bonanomi, "A model for TID effects on Floating Gate memory cells," IEEE Transactions on Nuclear Science, vol. 6, no. 49, p. 3753, 2004
52. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, A. Candelori, S. Lora, "Effect of different TID sources on charge loss from programmed FG cells," IEEE Transactions on Nuclear Science, vol.52, no.6, p.2372, 2005
53. R.D. Katznelson, D. Frohman-Bentchkowsky, "An erase mode for FAMOS EPROM devices," IEEE Transactions on Electron Devices, vol.27, no.9, p.1744, 1980
54. V.A.K. Raparla, S.C. Lee, R.D. Scrimpf, D.M. Fleetwood, K.F. Galloway, "A model of radiation effects on nitride-oxide films for power MOSFET applications," Solid State Electr., vol.47, p.775, 2003
55. H. Aozasa, I. Fujiwara, A. Nakamura, Y. Komatsu, "Analysis of carrier traps in Si3N4 in oxide/nitride/oxide for Metal/Oxide/Nitride/Oxide/Silicon nonvolatile memories," Journal of Applied Physics, 38, p.1441, 1999
56. D.M. Fleetwood, P.S. Wikonur, L.J. Lorence, W. Beezhold, P.V. Dressendorfer, J.R. Schwank, "The response of MOS devices to dose-enhanced low-energy radiation," IEEE Transactions on Nuclear Science, vol.33, no.6, p.1245, 1986
57. D.M. Fleetwood, D.E. Beutler, L.J. Lorence, D.B. Brown, B.L. Draper, L.C. Rieve, H.B. Rosenstock, D.P. Knott, "Comparison of enhanced device response and predicted X-ray dose enhancement effects in MOS oxides," IEEE Transactions on Nuclear Science, vol.35, no.6, p.1265, 1988
58. G. Cellere, A. Paccagnella, A. Visconti, S. Beltrami, J. Schwank, M. Shaneyfelt, L. Damien, P. Paillet, V. Ferlet-Cavrois, J. Baggio, R. Harboe-Sørensen, E. Blackmore, A. Virtanen, P. Fuochi, "Direct evidence of secondary events induced by high energy protons," IEEE Transactions on Nuclear Science, vol.55, no.6, p.2904, 2008

Petitioner Micron Ex-1021, 0570

59. G. Cellere, P. Pellati, A. Chimenton, A. Modelli, L. Larcher, J. Wyss, A. Paccagnella, "Radiation effects on floating-gate memory cells," IEEE Transactions on Nuclear Science, vol.48, no.6, p.2222, 2001
60. G. Cellere, A. Paccagnella, L. Larcher, A. Chimenton, J. Wyss, A. Candelori, A. Modelli, "Anomalous charge loss from Floating-Gate memory cells due to heavy ions irradiation," IEEE Transactions on Nuclear Science, vol.49, no.6, p.3051, 2002
61. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, A. Candelori, "Transient conductive path induced by a single ion in 10nm SiO_2 layers," IEEE Transactions on Nuclear Science, vol.49, no.6, p.3304, 2004
62. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, "Secondary effects of single ions on Floating Gate memory cells," IEEE Transactions on Nuclear Science, vol.53, no.6, p.3291, 2006
63. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, R. Harboe Sørensen, A. Virtanen, "Angular dependence of heavy ion effects in Floating Gate memory arrays," IEEE Transactions on Nuclear Science, vol.54, no.6, p.2371, 2007
64. P. McNulty, W. Beauvais, D. Roth, J. Lynch, A. Knudson, W. Stapor, "Microbeam analysis of MOS circuits," IEEE Transactions on Nuclear Science, vol.39, no.3, p.431, 1992
65. L. Edmonds, "A method for correcting cosine-law errors in SEU test data," IEEE Transactions on Nuclear Science, vol.49, no.3, p.1522, 2002
66. L. Larcher, G. Cellere, A. Paccagnella, A. Chimenton, A. Candelori, A. Modelli, "Data retention after heavy ion exposure of Floating Gate memories: analysis and simulation," IEEE Transactions on Nuclear Science, vol.50, no.6, p.2176, 2003
67. G. Cellere, L. Larcher, A. Paccagnella, A. Visconti, M. Bonanomi, "RILC in 10 nm SiO_2 layers," IEEE Transactions on Nuclear Science, vol.52, no.6, p.2144, 2005
68. G. Cellere, A. Paccagnella, L. Larcher, A. Visconti, M. Bonanomi, "Sub-attoampere current induced by single ions in silicon oxide layers of nonvolatile memory cells," Applied Physics Letters, 88, 192909, 2006
69. G. Cellere, A. Paccagnella, A. Visconti, M. Bonanomi, "Variability in FG memories performance after irradiation," IEEE Transactions on Nuclear Science, vol.53, no.6, p.3349, 2006
70. A.J. Lelis, H.E. Boesch, Jr., T.R. Oldham, F.B. McLean, "Reversibility of trapped hole charge," IEEE Transactions on Nuclear Science, vol.35, no.6, p.1186, 1988
71. A.J. Lelis, T.R. Oldham, H.E. Boesch, Jr., F.B. McLean, "The nature of the trapped hole annealing process," IEEE Transactions on Nuclear Science, vol.36, no.6, p.1808, 1989
72. J.H. Stathis, D.J. DiMaria, "Reliability projection for ultra-thin oxides at low voltage," IEEE International Electron Device Meeting 1998, p.167
73. J.R. Schwank, D.M. Fleetwood, P.S. Winokur, P.V. Dressendorfer, D.C. Turpin, D.T. Sanders, "The role of hydrogen in radiation-induced defect formation in polysilicon gate MOS devices," IEEE Transactions on Nuclear Science, vol.34, no.6, p.1152, 1987
74. M. Beck, B. Tuttle, R. Schrimpf, D. Fleetwood, S. Pantelides, "Atomic displacement effects in single-event gate rupture," IEEE Transactions on Nuclear Science, vol.55, no.6, p.3025, 2008
75. G. Cellere, M.G. Valentini, L.Pantisano, K.P. Cheung, A. Paccagnella, "Different nature of process-induced and stress-induced defects in thin SiO_2 layers," IEEE Electron Device Letters, vol.24, no.6, p.393, 2003

Petitioner Micron Ex-1021, 0571

76. L. Pantisano, K.P. Cheung, "Stress induce leakage current (SILC) and oxide breakdown: are they from the same oxide traps?," IEEE Transactions on Device and Material Reliability, p.109, 2001
77. J.L. Autran, P. Roche, S. Sauze, G. Gasiot, D. Munteanu, P. Loaiza, M. Zampaolo, J. Borel, "Real-time neutron and alpha soft-error rate testing of CMOS 130nm SRAM: Altitude versus underground measurements," IEEE International Conference on Integrated Circuit Design and Technology 2008, p.233
78. A. Silburt, A. Evans, A. Burghelea, W. Shi-Jie, D. Ward, R. Norrish, D. Hogle, "Building a reliable internet core using soft error prone electronics," IEEE International Conference on Integrated Circuit Design and Technology 2008, p.227
79. F. Irom, T. Miyahira, D. Nguyen, J. Insoo, E. Normand, "Results of Recent 14 MeV Neutron single event effects measurements conducted by the Jet Propulsion Laboratory," IEEE Radiation Effects Data Workshop 2007, p.141
80. G. Cellere, S. Gerardin, M. Bagatin, A. Paccagnella, A. Visconti, M. Bonanomi, S. Beltrami, R. Harboe- Sørensen, A. Virtanen, P. Roche, "Can atmospheric neutrons induce soft errors in NAND Floating Gate memories?," IEEE Electron Device Letters, vol.30, no.2, p.178, 2009
81. G. Cellere, S. Gerardin, M. Bagatin, A. Paccagnella, A. Visconti, M. Bonanomi, S. Beltrami, P. Roche, G. Gasiot, R. Harboe Sørensen, A. Virtanen, C. Frost, P. Fuochi, C. Andreani, G. Gorini, A. Pietropaolo, S. Platt, "Neutron-induced soft errors in advanced Flash memories," IEEE International Electron Devices Meeting 2008, p.357.

Petitioner Micron Ex-1021, 0572

About the authors

Rino Micheloni (rino.micheloni@ieee.org) is Lead Flash Technologist at IDT (Integrated Device Technology). He has 15 years experience in NAND/NOR Flash memory design, architecture and algorithms as well as the related intellectual property. Before IDT, he was senior principal for Flash and Director of Qimonda's design center in Italy, developing 36 and 48 nm NAND memories. From 2001 to 2006 he managed the Napoli design center of STMicroelectronics focusing on the development of 90 and 60 nm MLC NAND Flash. Before that, he led the development of MLC NOR Flash. He is co-author of 95 U.S. patents and four Springer books on NOR/NAND/ECC. He is IEEE Senior Member and received the STMicroelectronics Exceptional Patent in 2003 and 2004, and the Qimonda IP Award in 2007.

Luca Crippa is Senior Technical Analyst for Design Architecture at Forward-Insights (www.forward-insights.com) with more than 10 years of experience in MLC Flash memory design. Previously, he was Senior Designer for 48 nm floating gate and 36 nm floating gate NAND Flash memories at Qimonda as well as 90 and 60 nm MLC NAND Flash products at STMicroelectronics. He was instrumental in the development of 64, 128 and 256 Mb MLC NOR Flash products at STMicroelectronics and is the author/co-author of 20 U.S. patents and the book *Memories in Wireless Systems* (Springer, 2008). Luca received his Masters degree in Electronic Engineering at the Politecnico of Milan in 1999.

Alessia Marelli was born in Bergamo, Italy in 1980. She received her degree in Mathematical Science from "Università degli Studi di Milano – Bicocca", Italy in 2003 with a thesis about ECC applied to Flash Memories. In 2003 she joined STMicroelectronics, Agrate B., Italy. She was involved in digital design of Multi-level NAND Memories, especially redundancy, ECC and algorithms. In 2007, she joined Qimonda as senior digital designer. In 2009 she joined Integrated Device Technology (IDT) as senior digital designer, where she takes care of ECC applied to SSD. She is co-author of some patents regarding Redundancy and ECC applied to Flash Memories. She is co-author of *Memories in Wireless Systems* (Springer, 2008) and *Error Correction Codes for Non-Volatile Memories* (Springer, 2008).

Petitioner Micron Ex-1021, 0573

Index

A

Adaptive frequency and duty cycle (AFD), 529–531, 533–535
Address cycles, 27, 29, 32–34, 36
Address latch enable (ALE), 28–30, 133, 516
Address loader, 136–138
ALE. *See* Address latch enable
Alpha particles, 537, 538, 542
ALU. *See* Arithmetic logic unit
Architecture
  all bit line (ABL), 220–223, 225–231, 245, 356, 463
  double-side, 212–214, 216, 230
  interleaving, 209–218, 220, 225–231, 245, 268, 269, 463
  single-side, 212, 213, 216, 229
Arithmetic logic unit (ALU), 145–147
Array access time, 36, 164
Array method, 494
Asynchronous interface, 140, 163–165, 169
ATE. *See* Automated test equipments
Automated test equipments (ATE), 433

B

Back end, 428–431, 449–451
Back ground pattern dependency (BPD), 75, 99, 235–239, 250, 255
Bad block management (BBM), 10, 13, 40, 42, 426, 435–436, 451, 516
Bad blocks (BB), 42, 93, 354, 377, 383, 386, 425, 430, 433, 435, 436, 443, 496
Bake, 430
Ball grid array (BGA), 178, 193, 194
Band engineered (BE), 120, 123, 125
Band-gap, 117, 118, 257, 304, 305, 315–317, 320, 383, 438, 439, 441, 442
Band-gap reference voltage, 207, 301, 316, 337, 438, 440, 441
BCH codes, 43, 401–408, 421
Berlekamp–Massey, 403
Bias temperature instability (BTI), 110
Bin, 363
Bitline
  coupling, 220
  cut, 356, 357, 372, 373, 375–377, 379
  discharges, 203
  noise, 226
  pairs, 152, 212, 358–362, 367, 368, 371, 376, 379, 383
  parasitic capacitance, 23, 200, 210, 220
  precharge, 202, 206, 268, 332
  shorts, 353, 359–362, 372, 374–379, 389
  short table, 377
Bitline selectors (BLSEL), 212, 214–216, 230, 236, 245
Block codes, 43, 390, 391, 394–399, 507
Board, 534
Boosted channel potential, 70, 71, 73, 74
Boosts, 25, 68, 70, 72–75, 302, 342–345, 529–531, 535
Built in self test (BIST), 433, 434, 440, 443, 451
Bulk terminals, 90
Burn in, 429, 431–434, 437, 443, 451

C

Cache, 1, 31, 34, 43, 44, 162, 163, 265, 428, 500
  program, 34, 35, 265, 464–469
  read, 32, 33, 271–276, 457, 469–471

Petitioner Micron Ex-1021, 0574

Carriers, 73, 75, 96, 101, 541, 543, 545, 548, 559
Cell current characteristic, 199
Cell error probability, 359, 360, 362, 368, 376, 391
Cell working point, 226–228
Cell working window, 209, 210
Chain method, 494
Channel hot electron, 90
Charge trap, 20, 51, 115–127, 545
Chien, 403, 408, 410–412
Chip enable (CE), 28, 133, 506, 516
Chip error probability (CEP), 369, 370, 375, 376, 393, 394
Chip on Board (COB), 486
Circulant, 416–418
CLE. *See* Command latch enable
Cleaning policy, 494, 496
CLK. *See* Clock
Clock (CLK), 36–38, 139, 140, 145–148, 151, 152, 156, 166–168, 180–182, 300, 302, 303, 324, 374, 398, 403, 418, 440, 448, 486, 507, 531
Coarse/fine programming, 285–289
Code RAM, 143, 152
Code rates, 396, 416
Coding, 8LC, 470
Coding, 16LC, 481
Cold data, 494, 496
Collision resistance, 498
Command cycles, 29, 30, 32, 34, 37
Command latch enable (CLE), 28–30, 133, 516
Concatenation, 397, 398, 498, 507
Control gate (CG), 26, 51, 56–61, 63, 66, 68, 78–80, 84, 90, 99, 107, 109, 125, 243, 244, 446, 504, 522, 524, 527, 528
Control interface (CI), 29, 132–138, 155
Controller, 10, 11, 13, 14, 39–44, 93, 98, 135–138, 147, 165, 166, 170, 177, 444, 457, 485, 492, 515, 516, 525–531, 533, 534, 555
Control unit, 146, 409
Convolutional codes, 394, 398–401
Core region (CR), 175, 187, 213, 216
Coronal mass ejection (CME), 539
Correction capability, 389, 396, 397, 407, 408, 413, 506–509
Cosmic rays, 537–540
Cost age time (CAT), 496
Coupling, 26, 56–60, 66, 74, 77–82, 84, 105, 125, 211, 212, 220, 226, 228–229, 235, 236, 239, 243–246, 250, 251, 255, 265–268, 332, 334, 335, 426, 447, 458–463, 471–477, 505–506
Coupling dielectric (CD), 56–60, 63, 66, 84
CPU, 1, 162, 163
Cryptography, 497–499
Current
  average consumption, 226
  average threshold, 204, 209
  integration, 200, 224
  mirror, 250, 284, 310, 314, 531
  saturation (I_{SSAT}), 198, 199, 235, 240
  spikes, 556–557
  string, 23, 198, 220, 230, 308
  threshold, 223, 236, 237, 240, 250, 258
Cyclic redundancy check (CRC), 487
Cipher, 497

D

3D arrays, 115, 126, 127
Data-load (DL), 44, 264, 265, 277–279, 281, 289–296, 464, 467
Datapath, 132, 133, 136, 138, 140–143, 212
Data RAM, 152
Data rate, 177, 178, 182, 416
DAT lines, 486–488
Debug, 143, 152, 157, 429, 438, 439, 447
Decrypt, 497
Density
  bitline, 213, 216
  BLINT signal, 214, 229, 230
  layout, 213
  page buffer (PB), 213
  sense amplifier (SA), 230
  signal, 214, 229, 230
Design for testability (DFT), 423–452
Detection capability, 396
Dielectric, 55, 56, 60, 77, 115–120, 122–124, 425, 542, 545, 560
Die stacking, 45–50, 193

Petitioner Micron Ex-1021, 0575

Digital controlled oscillator (DCO), 531, 532
Digital rights management (DRM), 484
Din cycles, 33, 34
Displacement damage (DD), 542
Distribution
  erased, 26, 197, 229, 238, 240, 255, 258, 333, 335, 337
  4LC, 464, 475
  8LC, 460
  16LC, 479
  MLC encoding, 6
  programmed, 91, 92, 238, 255, 261, 456, 458, 462
  reprogrammed, 462
  temporary, 460, 461, 472
3-D memory, 51
Double data rate (DDR), 36, 37, 44, 45, 142, 143, 161–195, 491
Dout cycle, 30, 31, 37, 38
DQ, 36, 167, 168, 182, 192–194, 289, 290, 292, 294, 296, 437
DQS, 29, 33, 36–38, 139, 167–169, 182
DRAM, 1, 11, 13, 14, 44, 50, 55, 162–164, 167, 170, 442
DSL/DSG, 102, 331, 332, 337, 356, 425, 450
Dual port, 147, 155, 156
DUT, 434
Dynamic data clustering (DAC), 494, 495
Dynamic striping, 495

E

Edge disturb, 73
Electrical erase, 25, 26, 239, 240, 315, 333–335
Electrical-topological scrambling, 357, 358
Electrical wafer sort (EWS), 353, 362–365, 376, 389, 429, 430, 435, 442, 445
Embedded MultiMedia Card (eMMC), 13, 161, 486, 497, 498
Encrypt, 497
Endurance, 6, 8, 10, 41, 51, 91–93, 96, 169, 510
Equivalent oxide thickness (EOT), 57, 84, 116, 117, 122, 123
Erase pulse, 26. 77, 120, 238, 244, 333, 334
Erratic bit, 93, 97
Error locator polynomial, 403, 405, 406, 410
ESD, 170, 180, 189–192, 194, 195
Evaluation phase, 203, 204, 207, 220, 221, 223, 226, 245, 249, 250
Evaluation time (t_{EVAL}), 200, 201, 203, 204, 206, 209, 240, 258
Execute in place (XIP), 11–13, 499–500, 510

F

Fail compression, 434, 446–447
Fail counter, 443–444, 447, 448
Failure analysis, 358–365
Failure probability, 358–365, 368, 370, 371, 374–376, 393, 565
Fermi level, 59, 122, 543
File allocation table (FAT), 40
File system, 14, 40, 436, 492
Final test, 429, 430
Finite state machine, 132–134, 138, 343, 373, 376, 379, 383
Flash card, 38–40, 51, 432, 433, 483–511
Flash transition layer (FTL), 10, 13, 40, 492–494, 496, 516
Flipped die stacking, 46
Floating gate coupling, 26, 105, 228–229, 235, 239, 265–268, 335, 458
Floating gate tunneling oxide (FLOTOX), 56, 90
Floorplan, 22, 47, 48, 50, 189, 192, 193
Fowler-nordheim, 24–26, 60–62, 72, 82, 90, 97, 107, 111, 119, 329
Funnel effect, 547
Fuse ROM, 383–388, 426, 442, 448
Fuses, 155, 317, 360, 365, 372–377, 380, 383–389, 442
Fusing, 338, 430, 436, 442

G

Galois field, 401
Gamma-ray, 541, 542, 545, 554
Garbage collection, 40–42, 494, 496, 528
Gate-induced drain leackage (GIDL), 89, 101–103, 304, 346

Petitioner Micron Ex-1021, 0576

Global wordline, 341, 342, 348, 426
Greedy policy, 496

H

Hamming distance, 396, 400
Hard decision, 395, 398, 414, 415, 417
Hard disk drives (HDD), 14–16, 162, 163, 517, 527
Hard erase verify, 333, 334
Hash, 497–499
Helium nuclei, 538
Hierarchical GWL decoder, 346–350
High voltage (HV) switch, 299–326
Hot data, 494, 495
Hot hole injection (HHI), 93–95

I

Incremental step pulse programming (ISPP), 62–64, 261–263, 265, 278, 280, 281, 283–286, 329, 330, 332, 333, 335, 456, 469
Instruction set, 147, 149, 151–153
Interleaving, 164, 209–218, 220, 225–231, 245, 247, 267–269, 378, 463, 516, 525–528, 535
Interpoly, 89, 90, 116, 124, 125, 560
Inter poly dielectric (IPD), 55–57
Interposer, 46, 47, 529, 534
Inter symbol interference (ISI), 183
Inter wordline dielectric (IWD), 56, 75
I/O, 14, 44, 133, 141, 142, 161, 164, 166–170, 172–173, 179–195, 355, 437, 440, 446, 516
Ionization, 95, 541, 542, 546
Iterative log-BP decoding algorithm, 414

J

Jitter, 177, 182, 185, 188–189
Junctions, 74, 77, 101, 104, 125, 191, 318, 524, 542, 547, 548

K

Known good die (KGD) testing, 435

L

Latch static characteristic, 206, 208
Layout, 66, 190–195, 213–218, 302, 303, 384, 385, 425, 524
LDPC code. *See* Low-density parity-check code
Leakage, 43, 73, 74, 78, 89, 96, 97, 101–104, 117, 124, 126, 172, 320, 336, 431, 545, 553, 564
Least significant bit (LSB), 21, 27, 261–265, 267
8-Level-cell (8LC), 6–8, 21, 455, 459–471, 479
16-Level-cell (16LC), 6–8, 21, 455–457, 459, 464, 471–481
Linear energy transfer (LET), 541, 546, 547, 555–558, 561, 562, 564, 566
Logical to physical remapping, 492
Low-density parity-check (LDPC) code, 413–421
Low pin count testing (LPCT), 434, 437, 439–440, 443

M

MAC. *See* Message authentication code
Memory interface, 485
Message authentication code (MAC), 497–499
Meta-language, 153–158
Metal fuses, 385, 386
Microcontroller, 132, 133, 137, 138, 143–150, 152–155, 157, 297, 355, 365, 372, 424, 433, 443, 448, 475, 506–511, 549, 553–555, 566
Microdose effects, 543
MLC. *See* Multi level cell
Models for cell system interaction (MCSI), 78, 83–84
Most significant bit (MSB), 21, 140, 261–265, 267
Moving read, 296–297
MTBF, 163, 433
Multi-chip package (MCP), 11, 486
Multi-die, 44–45
Multi-level, 5–6, 8, 420, 520, 521

Petitioner Micron Ex-1021, 0577

Multi level cell (MLC), 5–8, 10, 21, 41, 51, 89, 98, 107, 131, 161, 165, 169, 200, 256, 261–297, 303, 355, 356, 365, 389, 425, 428, 433, 439, 441, 447, 455, 463, 500, 501, 506, 510, 511, 520, 521, 557, 558, 565
MultiMediaCard (MMC), 38, 40, 483, 485, 486
Multi-plane, 32, 33, 36, 355
Multi trap assisted tunneling (mTAT), 564
Muons, 538

N

Neutrons, 537–539, 541, 554, 564–566
Node processing, 414, 415, 418
NOR, 1–5, 10–13, 20, 45, 80, 90, 97, 105–107, 161, 172, 176, 198, 199, 213, 220, 333, 484, 500, 510, 530, 558
Nuclear reactions, 538, 539, 541, 542, 561, 565

O

Off chip driver (OCD), 170–172, 180–185, 187, 188, 192–194
On die termination (ODT), 167, 168, 180
One time programmable (OTP), 388–391, 442–443
Opcode, 145, 148, 150, 153, 157, 372
Open NAND flash interface (ONFI), 36, 162, 167, 168, 184
Operating system (OS), 492, 527–529
OTP. *See* One time programmable
Output buffer, 140, 142, 172–175, 177, 180, 181, 492
Overprogramming, 97–98, 107
Oxide degradation, 24, 91–98

P

Package, 13, 22, 28, 38, 45, 48, 50, 169, 170, 172, 175, 177, 178, 193, 195, 432, 434, 435, 438, 445, 485, 486, 529
PAD, 192, 252, 289, 294
Page access time, average, 272, 456
Page buffer (PB), 30, 31, 34, 35, 48, 66, 68, 74, 77, 131, 132, 138, 141–143, 154, 164, 201–203, 207, 211–218, 220, 230, 245, 246, 264, 276, 318, 355–357, 372, 423, 427, 428, 430, 444–447, 450, 451, 521, 522, 549, 554–555, 557, 558, 566
Page sizes, 13, 36, 66, 67, 161, 164, 165, 289, 290, 292, 419, 527–529
Parametric measurement tester unit (PMU), 441
Parity bits, 43, 389–391, 402, 403, 407, 408, 410, 506, 508, 511
Pass disturbs, 25, 73, 75, 99, 100, 503–504
Pass voltage (V_{PASS}), 98, 99, 204, 229, 338, 445, 450, 451, 502
PCB, 10, 14, 38, 46, 187, 541
P/E cycles, 106, 111, 505
Perfect code, 397
Permanent inhibit, 281, 467, 468
Photons, 541, 542, 560
Pin, 29, 36, 38, 165, 168, 190, 195, 437, 439, 443, 484, 486
Pinout, 28, 36, 37, 486
Plaintext, 497
Plane, 25, 32, 35, 36, 50, 136, 138, 151, 152, 209, 341, 346, 347, 355–357, 365, 366, 372, 373, 375–379, 383, 386, 388, 434, 554
Pointer table, 378–381, 383, 386
Polysilicon fuse, 384
Power detect (PD), 525, 526
Precharge phase, 202, 203, 206, 220, 223, 224, 226, 228, 342, 345
Precondition, 430, 445, 450, 451
Preimage resistance, 498
Production phase, 353, 354, 364, 372, 383, 428
Program
  counter, 143, 144
  disturbs, 25, 72–74, 76, 77, 99, 100, 104, 125, 229, 456, 504, 527, 528
  inhibit, 55, 67–70, 72–77, 79, 330, 331, 350, 452, 464, 522, 523
  4LC, 464, 475
  8LC, 460, 464, 467, 468
  16LC, 472, 474–477, 479

Petitioner Micron Ex-1021, 0578

pulses, 62–67, 72, 79, 96, 240, 261, 262, 276, 289, 303, 308, 311, 312, 329–333, 335, 456, 457, 464, 465, 504, 506, 521–525, 534, 535
rounds, 266, 460, 462, 477
sequence, 34, 135, 136, 268, 283, 459–463, 475
speed, 457, 460, 469, 517
verify, 62, 79, 276–283, 329, 431, 445, 465, 479, 506, 531
window, 455
Program after erase (PAE), 26, 240, 241, 244, 335–337
Program before erase (PBE), 333, 335, 337, 338
Program voltage generator (PVG), 529–535
Protocol interface, 485
Protons, 537–543, 561

Q

Quasi cyclic LDPC code, 416

R

Radiation induced leackage current (RILC), 545, 564
Ramp up phase, 428, 429
Random access, 3, 19, 43
Random access memories (ROM), 19, 143–146, 150–152, 157, 158, 372, 383–388, 408, 424, 426, 433, 442, 443, 448
Random telgraph noise (RTN), 89, 101, 104–107, 109
Raw bit error rate (BER), 393
Raw NAND, 10, 13, 168
RE#. *See* Read enable signal
Read all 1, 337
Read disturbs, 51, 99, 100, 125, 456, 502–504
Read enable signal (RE#), 28–30, 36, 38, 133, 142, 168
Read, from source line, 250–252
Read, 4LC, 457
Read, 8LC, 470
Read, 16LC, 457
Read margins, 229, 239, 255, 266, 297, 304
Read, single operation (SRO), 201, 202, 205, 222, 226, 258, 268–276, 278, 281, 283, 287, 456, 465, 469–471, 477, 480
Read voltage (V_{READ}), 110, 201, 238, 248, 252, 269, 296–297, 304, 308, 446–448
Ready/Busy (R/B), 28, 133, 516, 526
Realtime substitution, 365–371
Redundancy resources, 353, 358–368, 370–376, 379–381, 383, 386, 388
Reed–Solomon, 43, 398, 406–409, 507
Replay protected memory data (RPMD), 498, 499
Retention, 7–9, 20, 40, 43, 56, 62, 78, 91, 95–97, 99, 107, 117, 118, 120, 123, 124, 204, 296, 297, 430, 431, 436, 447, 564

S

Secure digital (SD), 10, 40, 433, 483, 485, 486
Secure erase, 490
Secure hash algorithm (SHA), 499
Secure purge, 490
Self aligned shallow trench isolation (SA-STI), 56, 58
Self boosted erase inhibit (SBEI), 77
Self boosted program inhibit (SBPI), 68–70, 73–77
Self-boosting, 24, 25, 100–102
Sense amplifier (SA), 22, 56, 58, 177, 220, 221, 224, 225, 229, 230, 259, 269, 289, 356, 384, 423, 464, 468
Sensing
circuits, 197–231, 272, 283, 289, 294, 468, 469
current, 248–250
multipass, 258–259
negative, 238–252
Serial access, 4
Serializer, 181, 182, 192
Shortening, 397
Silicon–oxide–nitride–oxide–silicon (SONOS), 90, 117, 122, 188

Petitioner Micron Ex-1021, 0579

Silicon phase, 428, 429
Simultaneous switching noise (SSN), 175, 177–179, 188
Single event effect (SEE), 542, 545–549, 554–558, 561–566
Single event functional interrupt (SEFI), 546, 555–556
Single event upset (SEU), 546–548, 555
Single level cell (SLC), 5–10, 21, 23, 26, 41, 51, 98, 165, 169, 200, 261, 281, 329, 355, 356, 365, 372, 389, 428, 446, 460, 471, 506, 510, 511, 550, 551, 556–558, 565
Slew rate, 175–178, 180, 185–187
Snake die stacking, 48
Soft decision, 395, 398, 420
Soft erase verify (SEV), 335–338
Software substitution, 371–383
Solar activity, 539, 540
Solar flare, 539
Solar particle event (SPE), 539
Solid state drives (SSD), 10, 13–16, 39, 40, 142, 163, 165, 166, 169, 170, 193, 515–535
Source line (SL), 20, 66, 74, 77, 197, 209, 210, 212, 228, 235, 239, 245–247, 249–258, 332, 450, 477, 523–525, 535
Source line noise, 255, 258
Source line noise compensation, 256, 257
Source select gate (SSG), 356
Source synchronous, 36, 37, 168
South atlantic anomaly (SAA), 540
SSG. *See* Source select gate
SSL. *See* String select line
Staircase die stacking, 47
Store and download (SnD), 12, 13, 499, 500, 510
Stress acceleration factors, 432
Stress induced leakage current (SILC), 96, 545
String select line (SSL), 66, 75, 101–103, 331, 337, 425, 552
Strip table, 377–381, 383, 386
Sunspot, 539
Super global wordline (SGWL), 347–350
Syndromes, 390, 396, 403–405, 408, 410, 411
Systematic form, 395
System in package (SiP), 170, 486, 529

T

Test flow, 364, 428–436, 440, 442, 443, 446, 450
Test interface (TI), 132, 133, 137–140, 440, 443, 448
Thermalization, 543, 559, 563
Threshold voltage, 5, 6, 20, 23, 24, 59, 71, 80, 89, 91–93, 96, 97, 99, 101, 104, 105, 107–111, 197, 198, 201, 202, 206, 224, 229, 235, 236, 239, 242–244, 246, 248–253, 268, 276, 285–287, 303–307, 311, 315, 320, 323, 329, 344, 345, 419–421, 457, 458, 466, 501, 503, 505, 507, 511, 542–545, 558–560, 562
Through silicon vias (TSV), 49, 50, 171
Toggle mode, 36, 37, 167–169
Total ionizing dose (TID), 541–546, 549–554, 565, 566
Trimming, 317, 426, 438–442
TSOP, 28, 38, 169, 175, 178, 195, 485
Tunneling dielectric (TD), 56, 58, 60, 62, 63, 66, 70, 72, 73, 77, 78
Tunnel oxide, 25, 89–91, 93, 95, 96, 99, 106, 108, 109, 119–121, 124, 126, 305, 451, 503, 504, 559, 562

U

Usermode, 137, 139
Usertestmode, 137

V

Valence band (VB), 93, 95, 543
Verify
  check circuits, 468
  operations, 238, 240, 250, 252, 261, 263, 276–283, 303, 386, 426, 434, 444, 447, 456, 457, 464–469, 477, 479, 480
  temporary level, 473
Virtual table, 380, 381, 386
Viterbi algorithm, 400

Petitioner Micron Ex-1021, 0580

Voltage
distribution, 197, 198, 253, 419, 420
doubler, 299–303, 324, 326
regulators, 39, 206, 250, 299–326, 337, 440–442
double supply, 308–315
internal dc-dc, 318–321
program, 312
read, 304
read temperature compensation, 315
stand-by, 320, 321
trimming, 440–442
verify level, 286

W

WE#. *See* Write enable
Wear leveling, 40–42, 51, 98, 494–496, 509, 510, 516
Wordline decoder, 341–346, 355
WP. *See* Write protect
W/R#. *See* Write/Read
Write enable (WE#), 28, 33, 36, 133, 141, 142
Write protect (WP), 28, 29, 133, 489, 499
Write/Read (W/R#), 36, 168
Write speed, 10, 517

X

X-ray, 541, 542, 545, 550, 551, 553, 559, 560

Y

Yield, 132, 380, 381, 423, 426, 428, 431, 434, 435, 437, 449–451, 453

Petitioner Micron Ex-1021, 0581

Made in the USA
Middletown, DE
23 July 2021



Rino Micheloni · Luca Crippa · Alessia Marelli

Inside NAND Flash Memories

Digital photography, MP3, digital video, etc. make extensive use of NAND-based Flash cards as storage media.

To realize how much NAND Flash memories pervade every aspect of our life, just imagine how our recent habits would change if the NAND memories suddenly disappeared. To take a picture it would be necessary to find a film (as well as a traditional camera…), disks or even magnetic tapes would be used to record a video or to listen a song, and a cellular phone would return to be a simple mean of communication rather than a multimedia console. The development of NAND Flash memories will not be set down on the mere evolution of personal entertainment systems since a new killer application can trigger a further success: the replacement of Hard Disk Drives (HDDs) with Solid State Drives (SSDs).

SSD is made up by a microcontroller and several NANDs. As NAND is the technology driver for IC circuits, Flash designers and technologists have to deal with a lot of challenges. Therefore, SSD (system) developers must understand Flash technology in order to exploit its benefits and countermeasure its weaknesses.

Inside NAND Flash Memories is a comprehensive guide of the NAND world: from circuits design (analog and digital) to Flash reliability (including radiation effects), from testing issues to high-performance (DDR) interface, from error correction codes to NAND applications like Flash cards and SSDs.

ISBN 978-94-007-9834-2

9 789400 798342

springer.com

Petitioner Micron Ex-1021, 0583