

Micheloni · Crippa · Marelli



Inside NAND Flash Memories

Rino Micheloni
Luca Crippa
Alessia Marelli

Inside NAND Flash Memories

 Springer

Inside NAND Flash Memories

Rino Micheloni • Luca Crippa • Alessia Marelli

Inside NAND Flash Memories

 Springer

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)

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.

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

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

Table of contents

Preface	v
Acknowledgements	vii
1 Market and applications for NAND Flash memories <i>Gregory Wong</i>	1
2 NAND overview: from memory to systems <i>R. Micheloni, A. Marelli and S. Commodaro</i>	19
3 Program and erase of NAND memory arrays <i>Cristoph Friederich</i>	55
4 Reliability issues of NAND Flash memories <i>C. Zambelli, A. Chimenton and P. Olivo</i>	89
5 Charge trap NAND technologies <i>Alessandro Grossi</i>	115
6 Control logic <i>A. Marelli, R. Micheloni and R. Ravasio</i>	131
7 NAND DDR interface <i>Andrea Silvagni</i>	161
8 Sensing circuits <i>L. Crippa and R. Micheloni</i>	197
9 Parasitic effects and verify circuits <i>L. Crippa and R. Micheloni</i>	235
10 MLC storage <i>L. Crippa and R. Micheloni</i>	261
11 Charge pumps, voltage regulators and HV switches <i>R. Micheloni and L. Crippa</i>	299
	ix

12 High voltage overview <i>R. Micheloni and A. Marelli</i>	329
13 Redundancy <i>A. Marelli and R. Micheloni</i>	353
14 Error correction codes <i>T. Zhang, A. Marelli and R. Micheloni</i>	393
15 NAND design for testability and testing <i>Andrea Silvagni</i>	423
16 XLC storage <i>R. Micheloni and L. Crippa</i>	455
17 Flash cards <i>A. Ghilardelli and S. Corno</i>	483
18 Low power 3D-integrated SSD <i>K. Takeuchi</i>	515
19 Radiation effects on NAND Flash memories <i>M. Bagatin, G. Cellere, S. Gerardin and A. Paccagnella</i>	537
About the authors	573
Index	575

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.

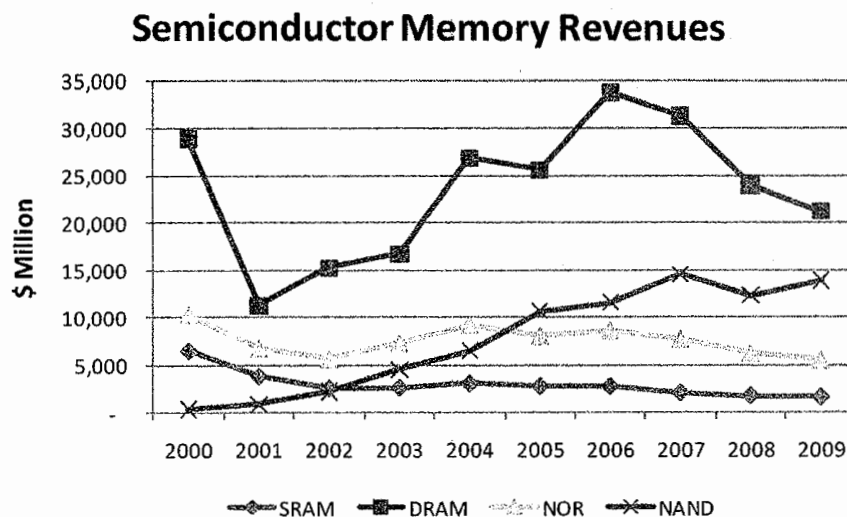


Fig. 1.1. Semiconductor memory revenues (Source: WSTS, Forward Insights)

* Forward Insights, greg@forward-insights.com

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 store 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.

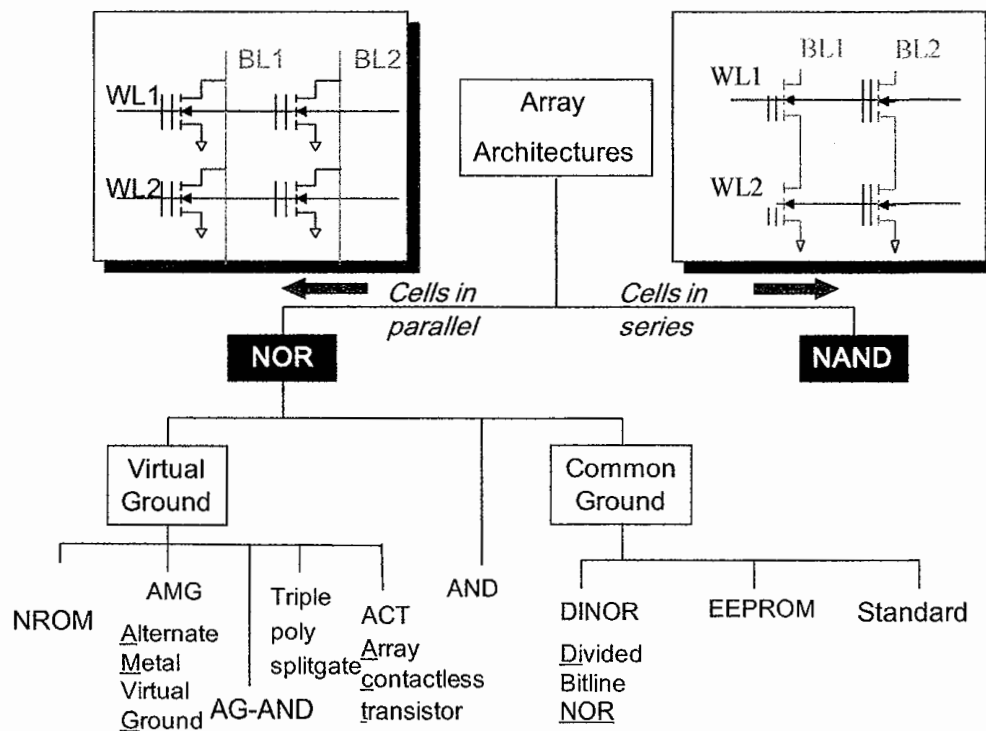


Fig. 1.2. Flash memory array architectures (Source: Forward Insights)

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 $4F^2$ 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 $10F^2$.

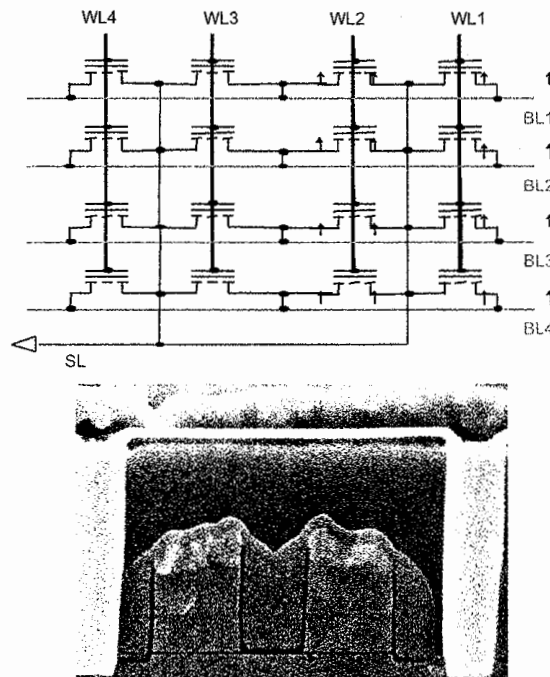


Fig. 1.3. NOR array architecture (Source: Forward Insights)

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.

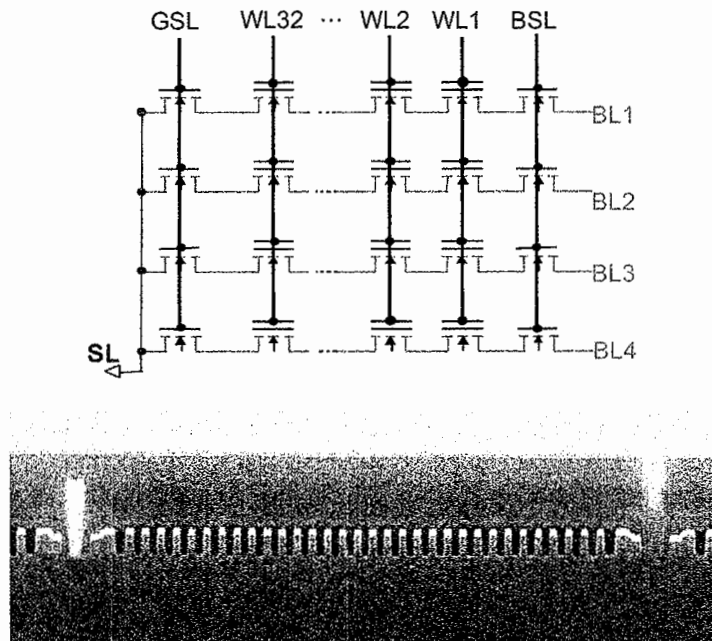


Fig. 1.4. NAND array architecture (Source: Forward Insights)

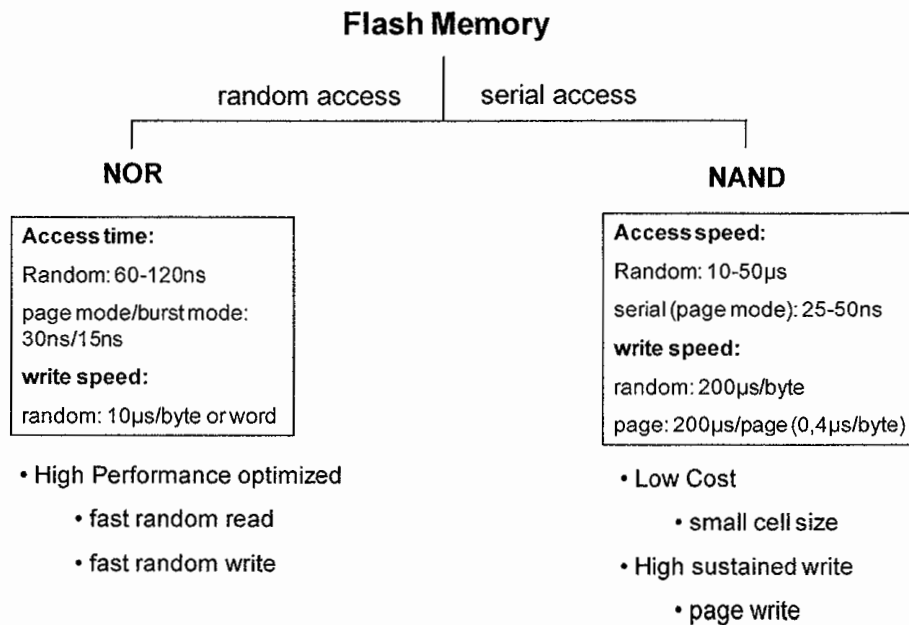


Fig. 1.5. NAND versus NOR (Source: Forward Insights)

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.

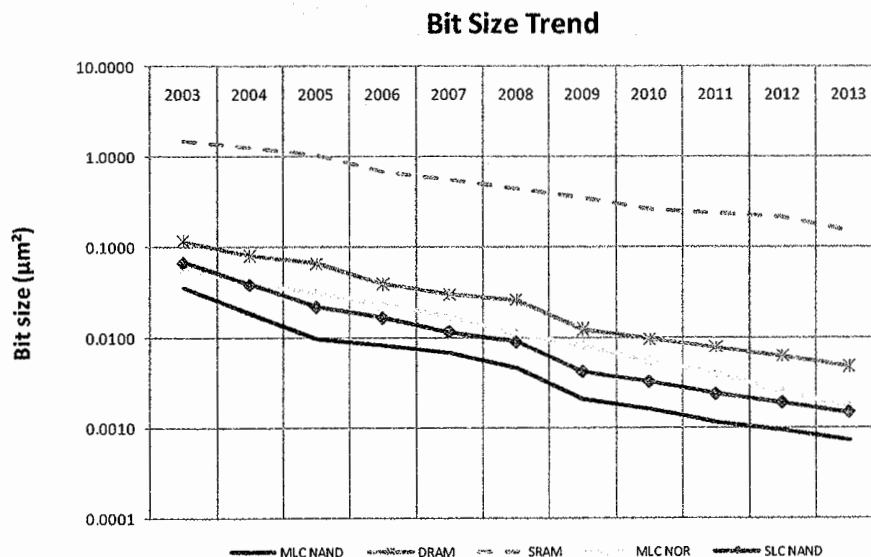


Fig. 1.6. Scaling (Source: Forward Insights)

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 (V_t) levels, more

than 1-bit per cell may be stored. Two bits per cell (MLC) storage is enabled by increasing the number of V_t 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 V_t charge levels so that the charge distributions don't overlap as well as accurate sensing of the different charge levels. As the number of V_t 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.

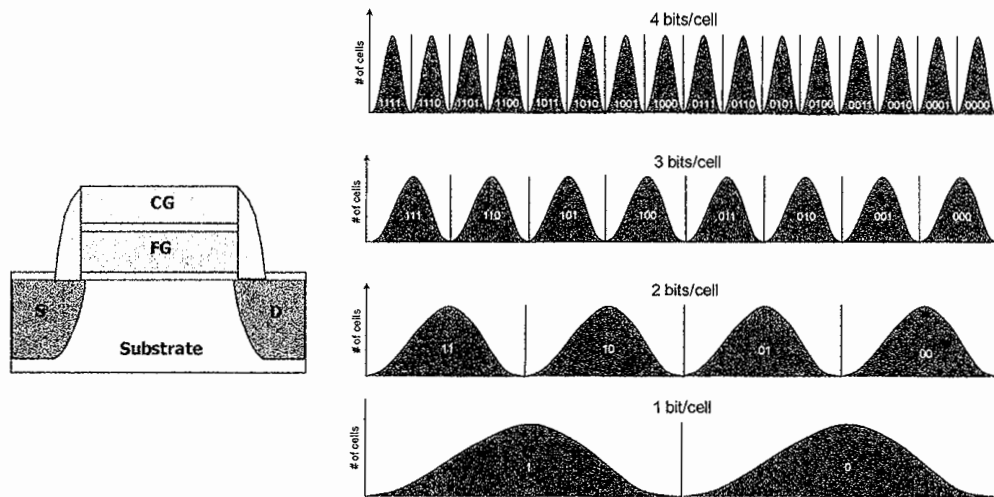


Fig. 1.7. Multi-level storage in floating gate NAND Flash memory

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

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.

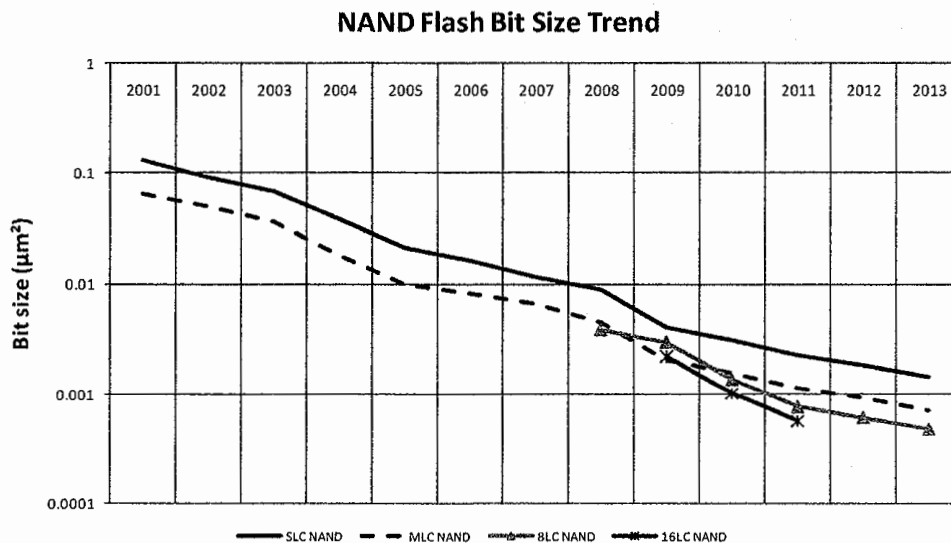


Fig. 1.8. NAND Flash bit size trend (Source: Forward Insights)

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

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).

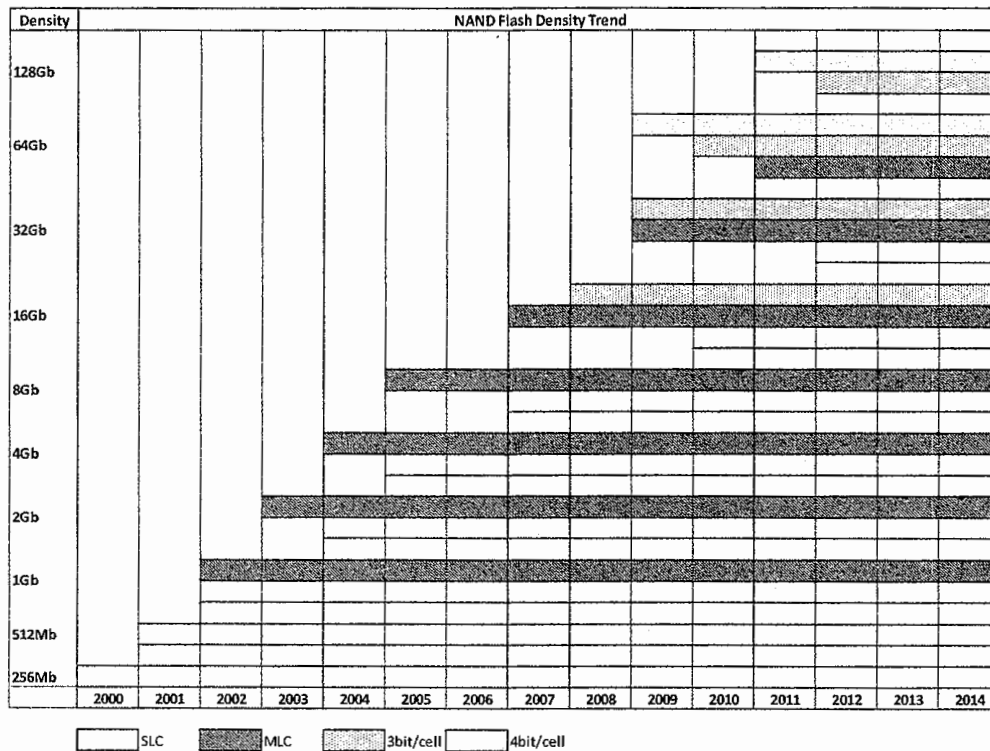


Fig. 1.9. NAND Flash density roadmap (Source: Forward Insights)

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.

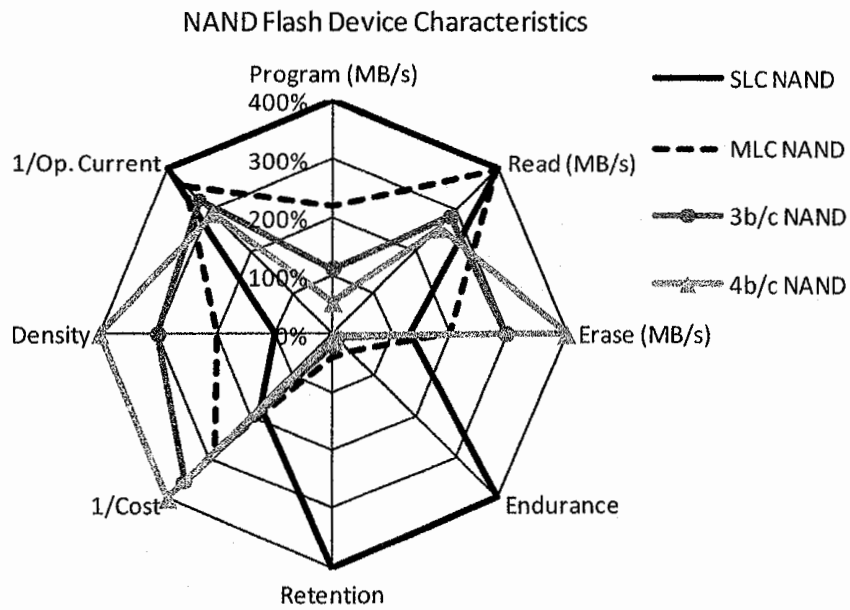


Fig. 1.10. NAND device characteristics (Source: Forward Insights)

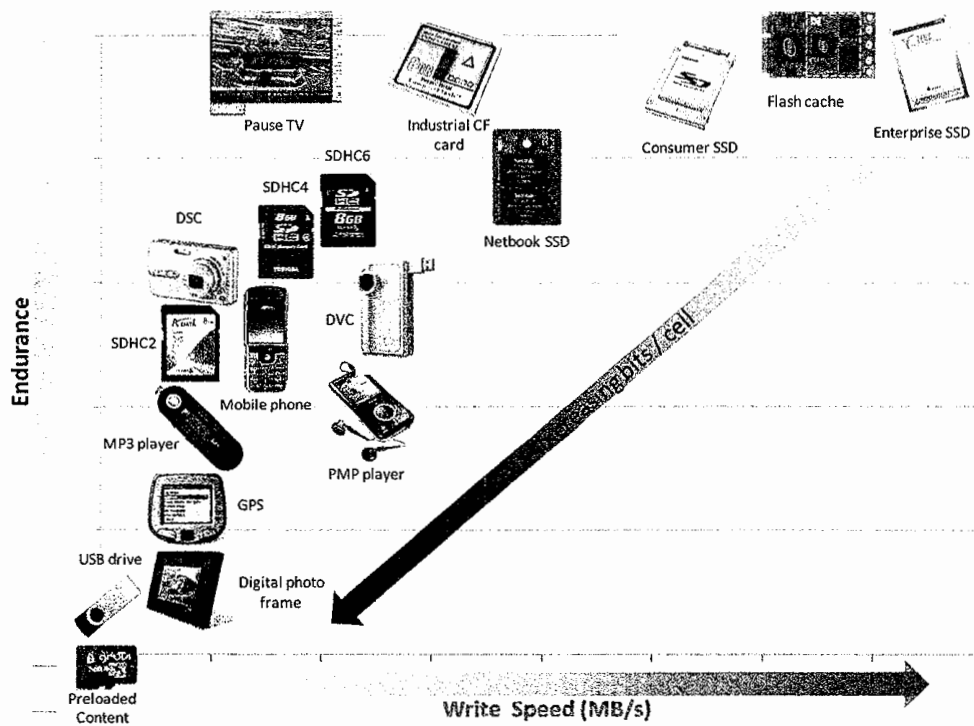


Fig. 1.11. Applications for multi-bit per cell NAND Flash memories (Source: Forward Insights)

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.

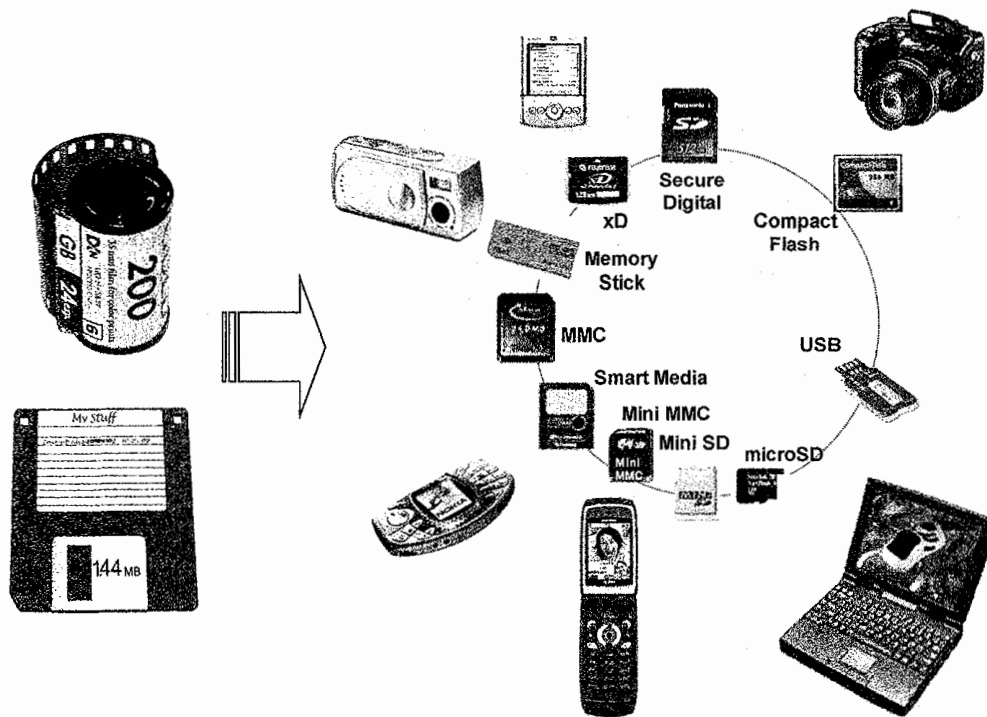


Fig. 1.12. Removable portable storage

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.

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.

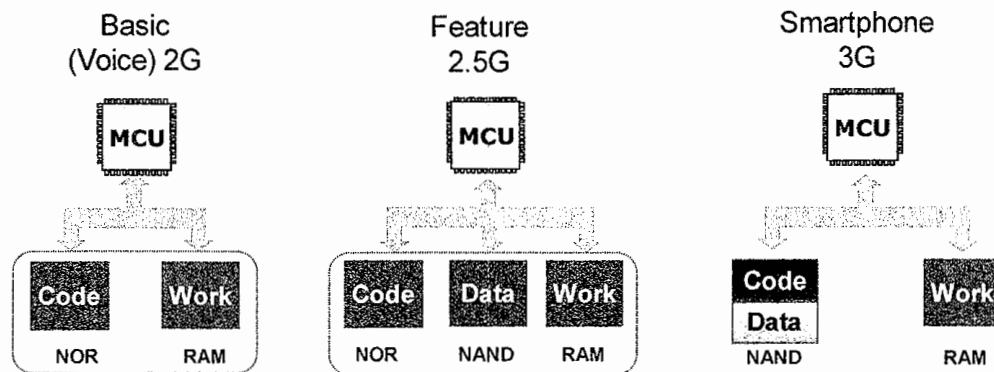


Fig. 1.13. Memory system architectures in mobile phones

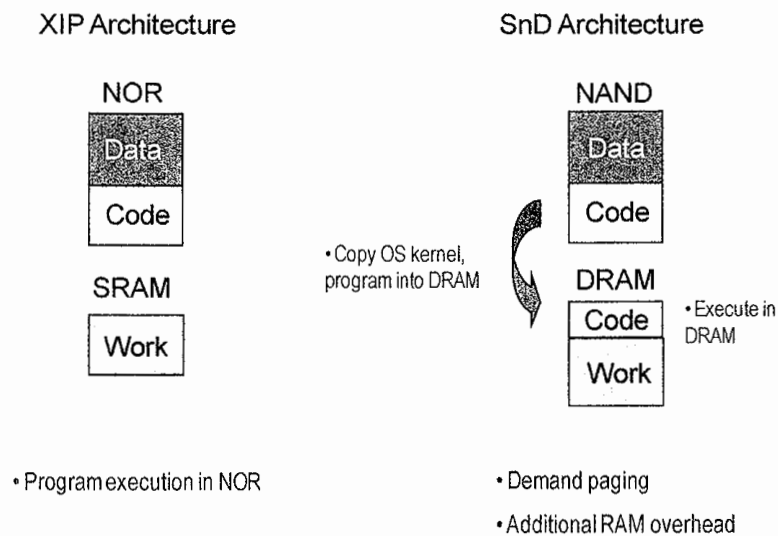


Fig. 1.14. Store and download versus XIP architecture

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.

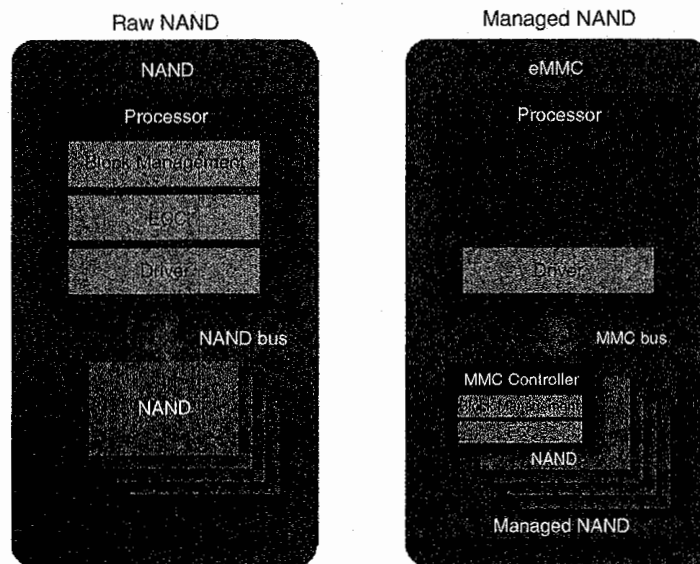


Fig. 1.15. Managed NAND versus raw NAND (Source: Micron Technology)

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.

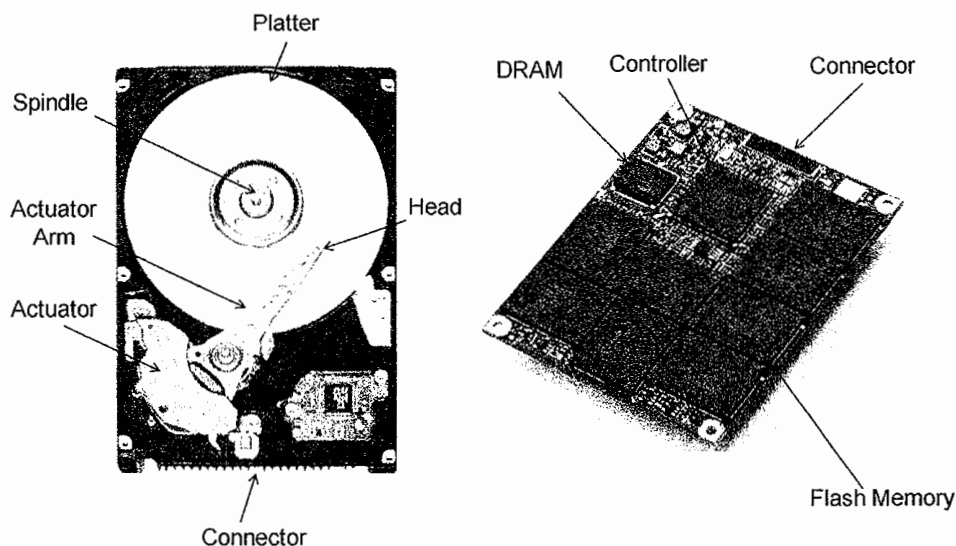


Fig. 1.16. Solid state drive versus hard disk drive (Source: Toshiba Corp., Forward Insights)

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 be 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

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.

SSDs vs. HDDs in PCs - Performance Characteristics

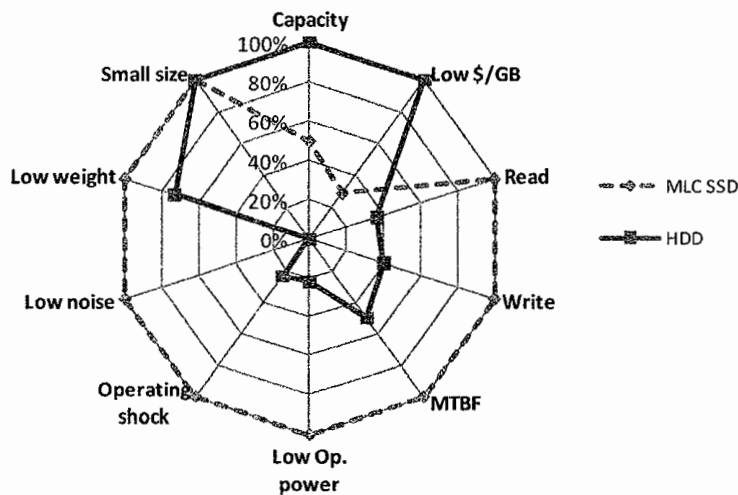


Fig. 1.17. 2.5" SSDs versus HDDs in PCs – performance characteristics (Source: Forward Insights)

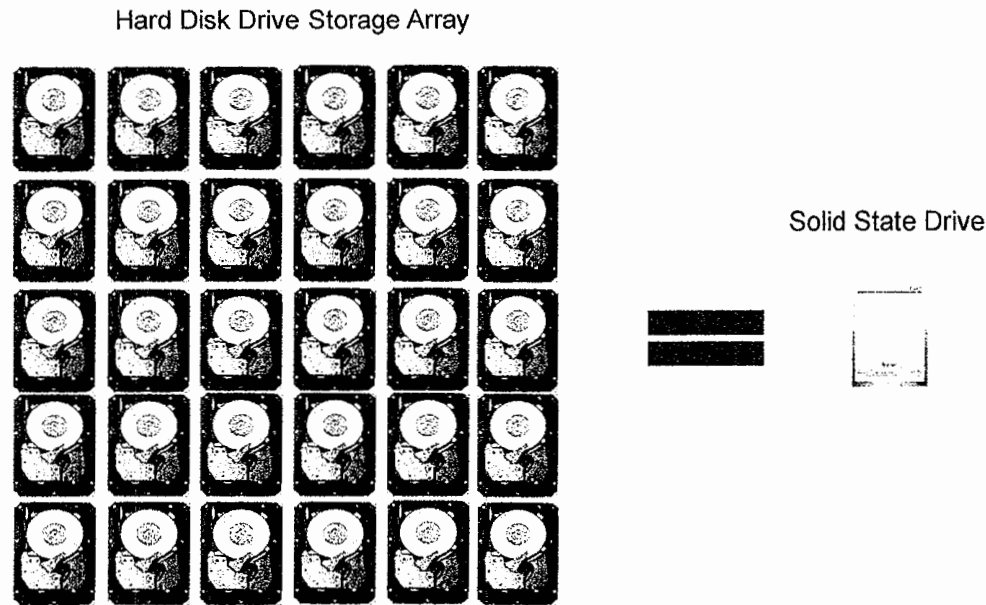


Fig. 1.18. Enterprise SSD value proposition

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%

¹STEC ZeusIOPS

Fig. 1.19. Enterprise SSD versus HDD key metrics

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).

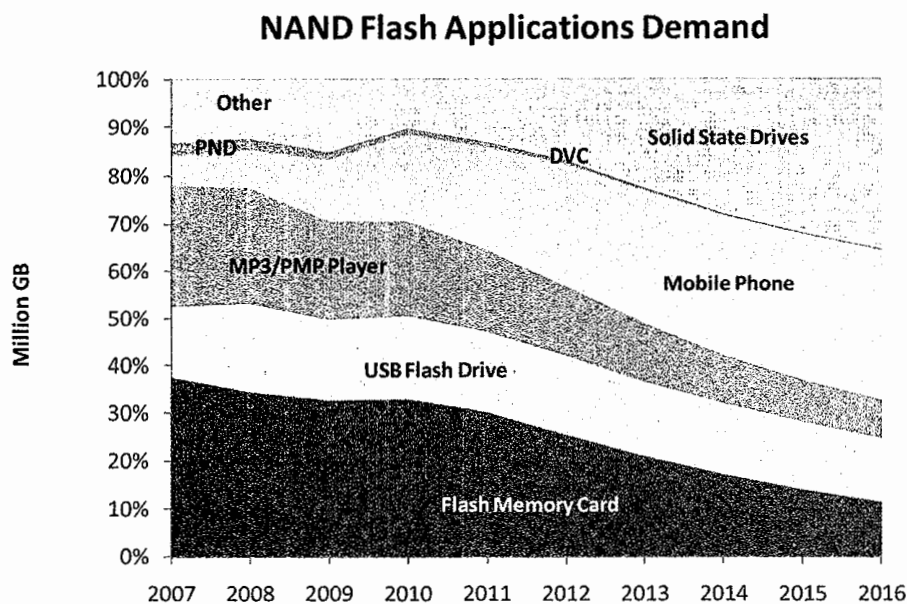


Fig. 1.20. NAND Flash applications demand (% GB) (Source: Forward Insights)

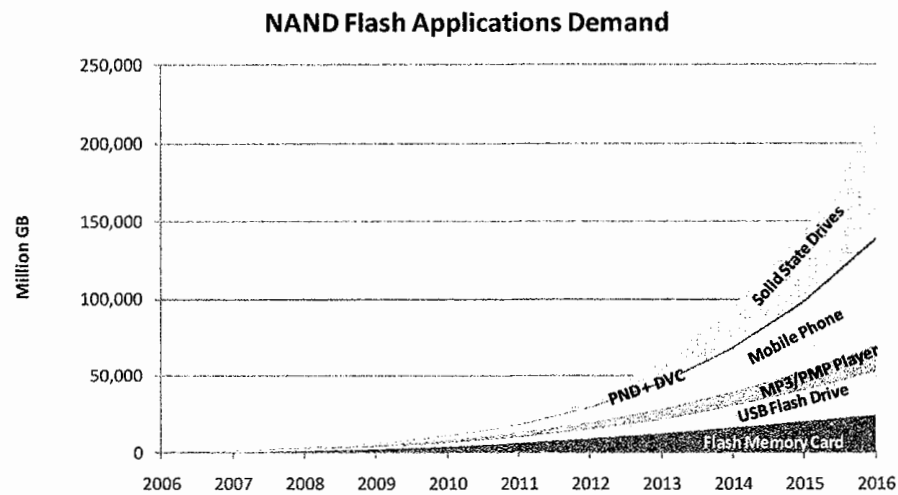


Fig. 1.21. NAND Flash applications demand (Source: Forward Insights)

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.

2 NAND overview: from memory to systems

R. Micheloni¹, A. Marelli² and S. Commodaro³

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

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.

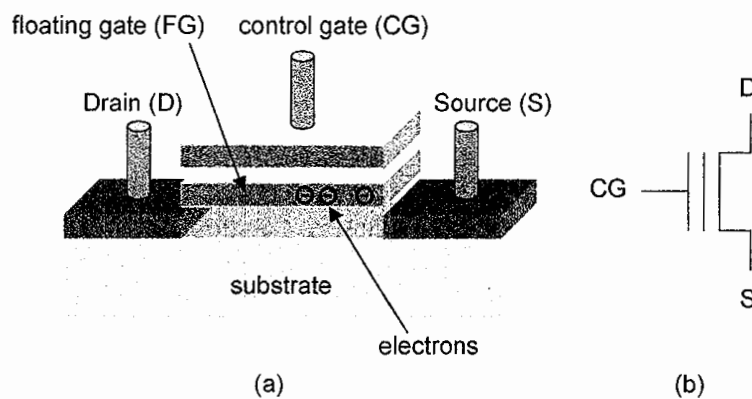


Fig. 2.1. (a) Floating gate memory cell and (b) its schematic symbol

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).

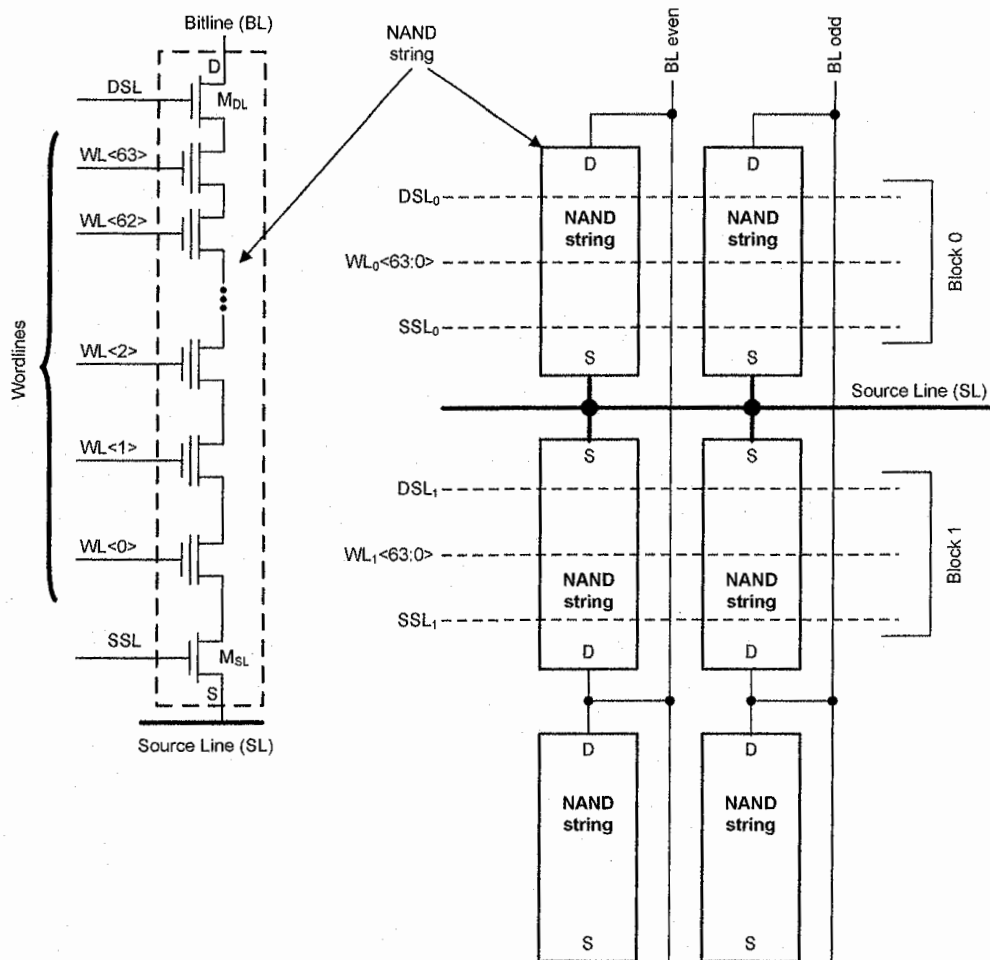


Fig. 2.2. NAND string (left) and NAND array (right)

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

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.

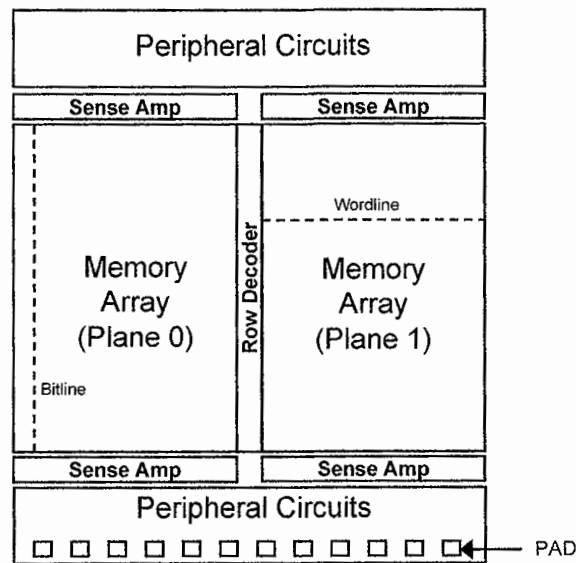


Fig. 2.3. NAND Flash memory floorplan

2.2.2 Basic operations

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

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 $V_{\text{PASS,R}}$ (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.

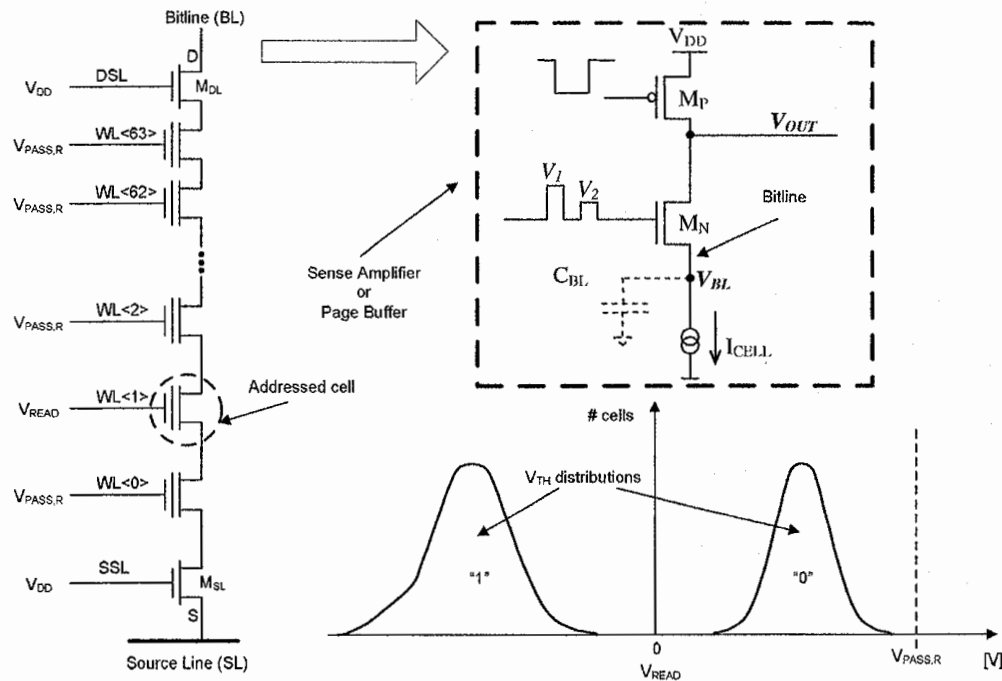


Fig. 2.4. NAND string biasing during read and SLC V_{TH} distributions

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} :

$$V_{\text{BL}} = V_1 - V_{\text{THN}} \quad (2.1)$$

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:

$$V_{BL} < V_2 - V_{THN} \quad (2.2)$$

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

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

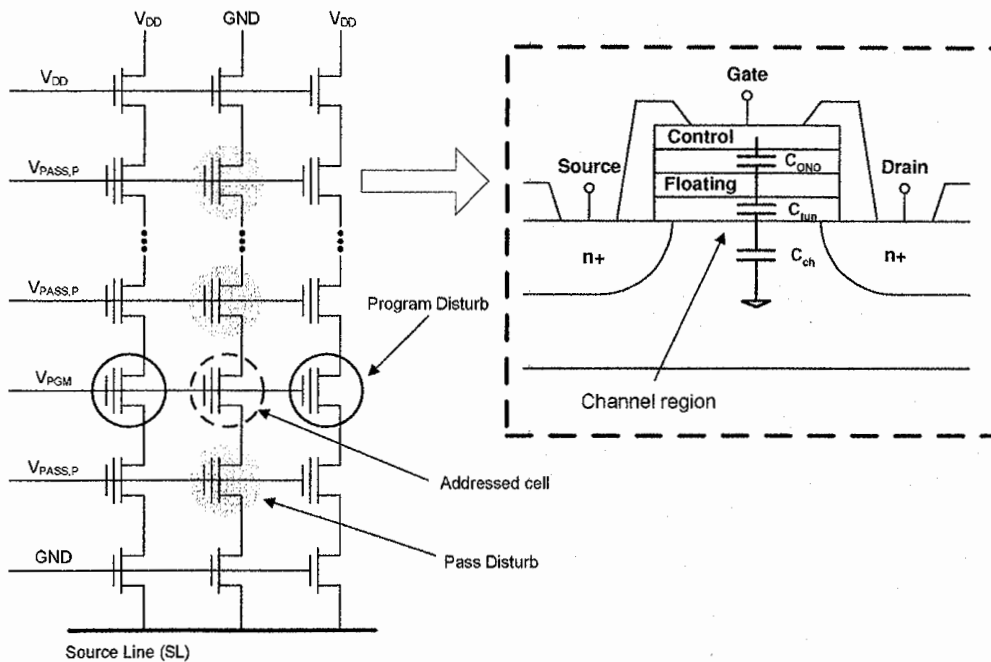


Fig. 2.5. NAND string biasing during program and parasitic capacitors contributing to the self-boosting inhibit

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,

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 μ s.

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.

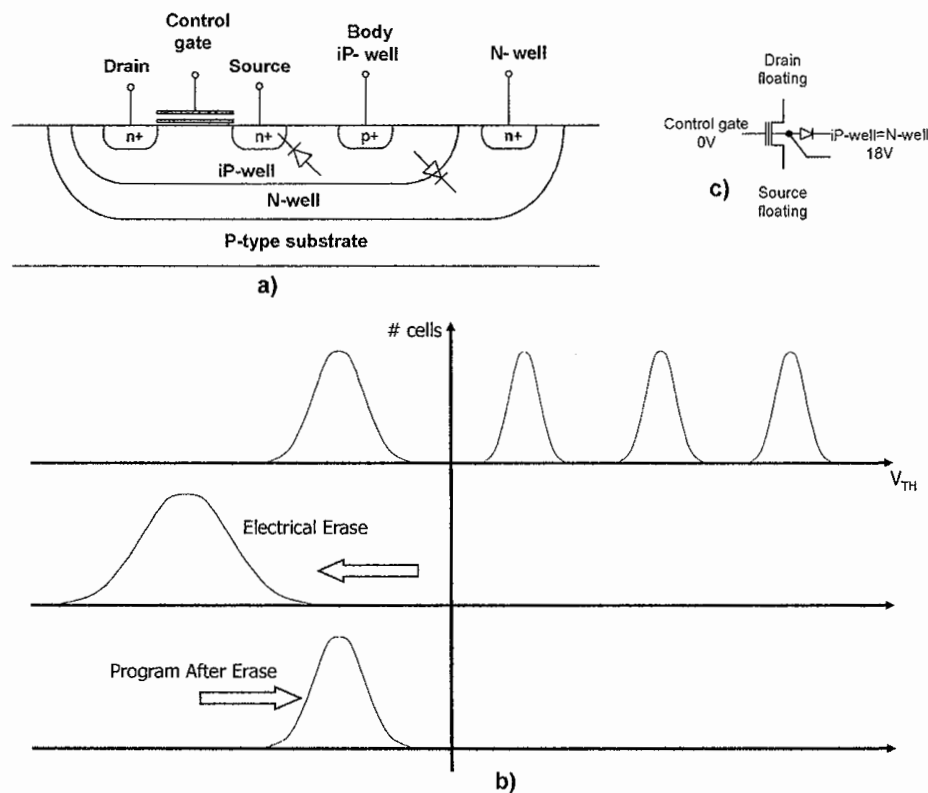


Fig. 2.6. (a) NAND matrix in triple-well; (b) erase algorithm; (c) erase biasing on the selected block

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.

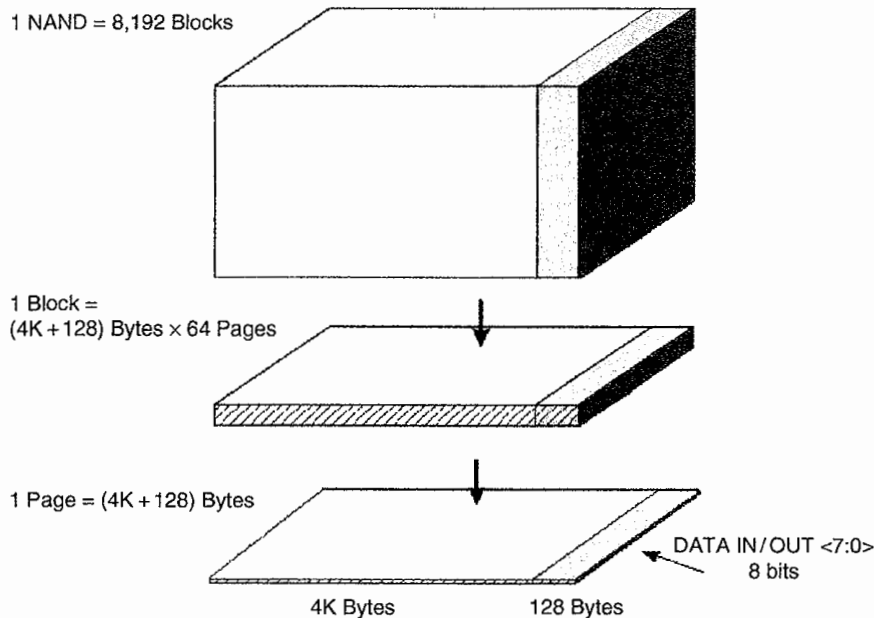


Fig. 2.7. NAND memory logic organization

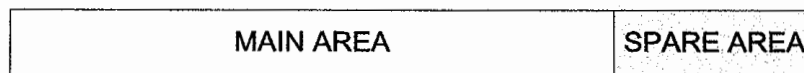


Fig. 2.8. NAND page structure

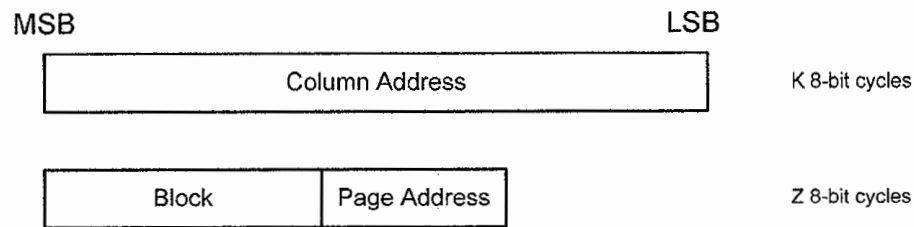


Fig. 2.9. Address structure

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.

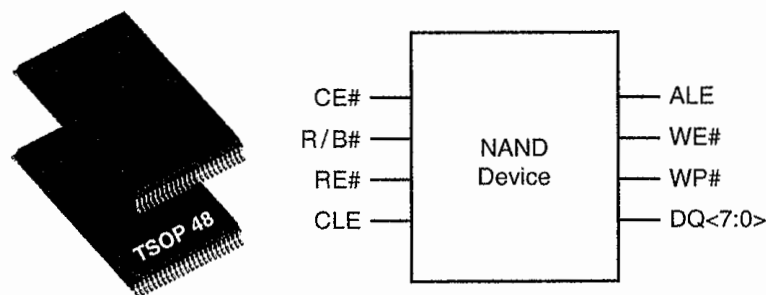


Fig. 2.10. TSOP package (left) and related pinout (right)

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.

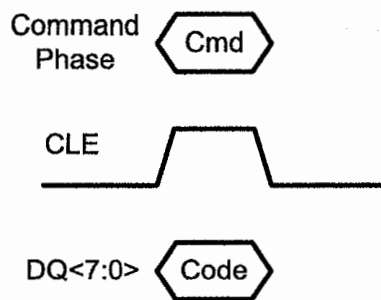


Fig. 2.11. Command cycle (“Cmd”): CLE is high, all other input signals are low

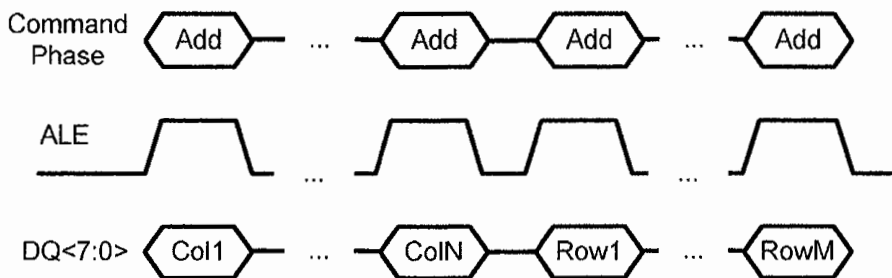


Fig. 2.12. Address cycle: ALE is high, all other inputs signals are low

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 μ s. 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.

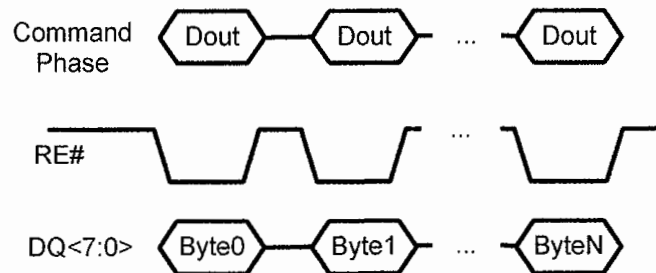


Fig. 2.13. “Dout” cycle: RE# is low, all other inputs signals are low

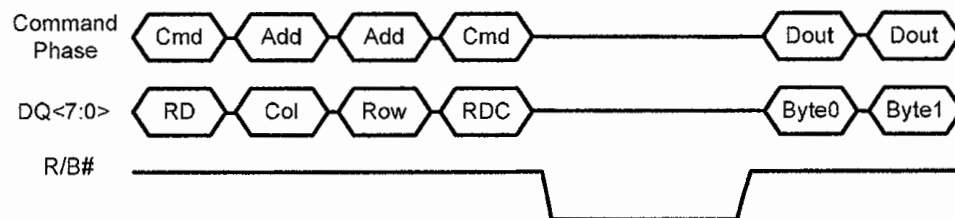


Fig. 2.14. Read command sequence

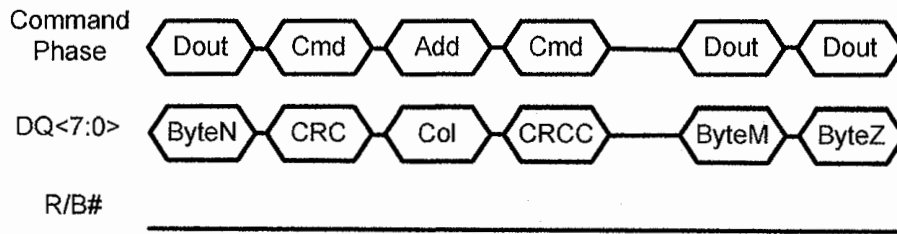


Fig. 2.15. Change Read Column sequence

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.

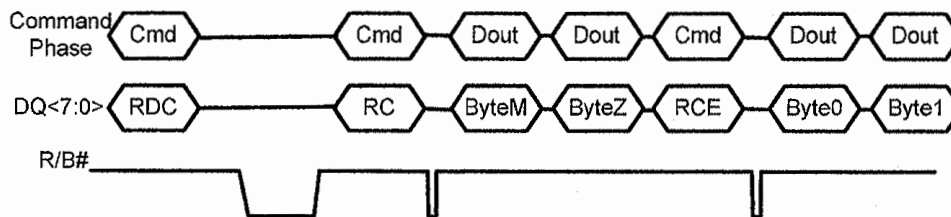


Fig. 2.16. Sequential Cache Read command

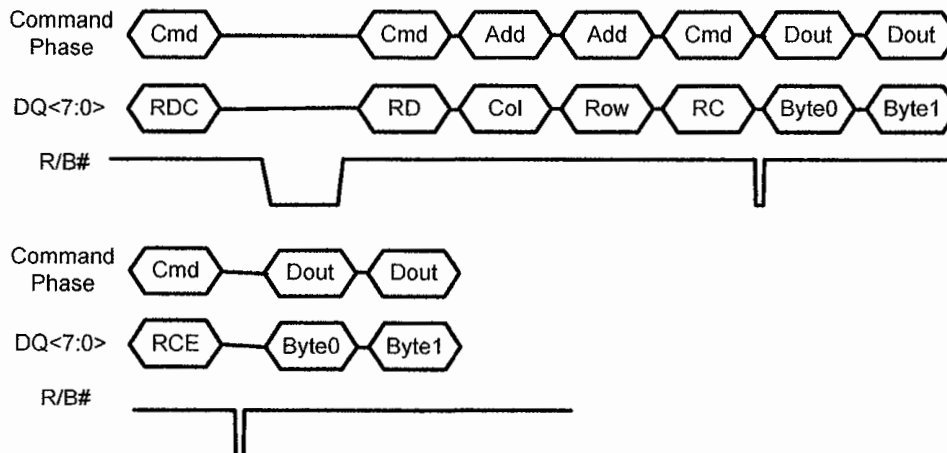


Fig. 2.17. Random Cache Read command

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.

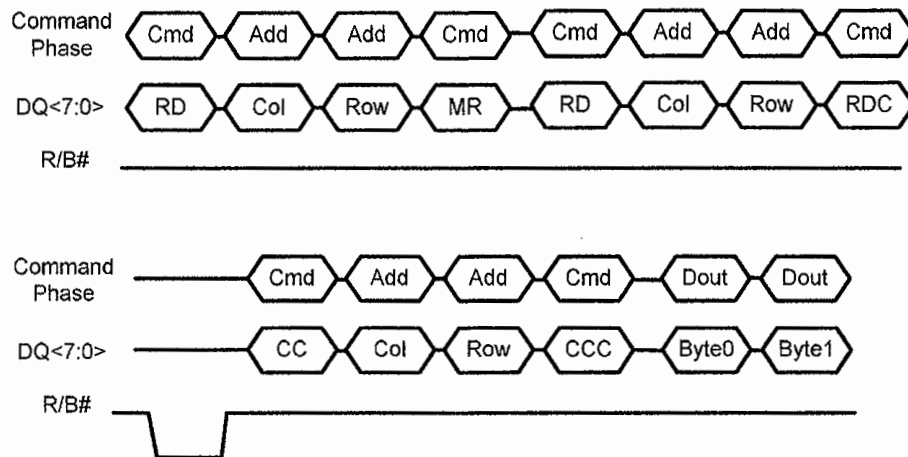


Fig. 2.18. Multi-plane read command

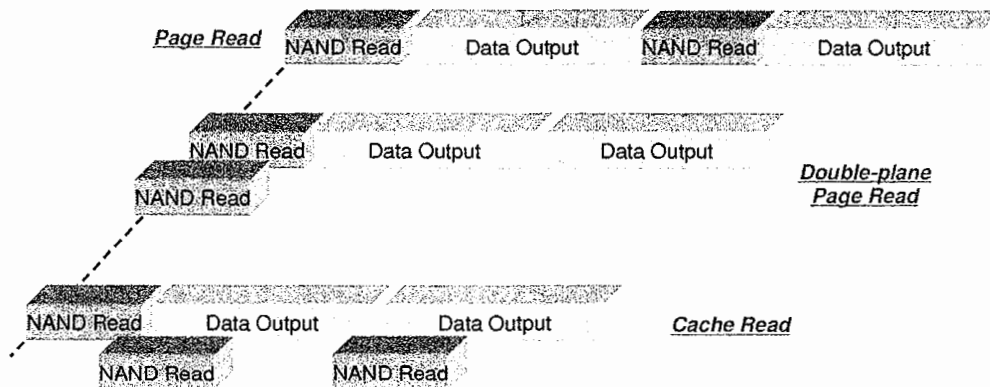


Fig. 2.19. Performance comparison among Read, Double-plane Read and Cache Read

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.

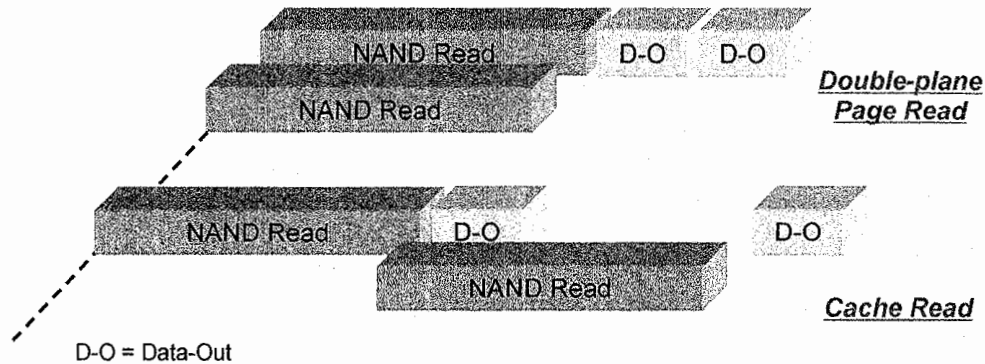


Fig. 2.20. Performance comparison among Read, Double-plane Read and Cache Read with a NAND array read time longer than page data output

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.

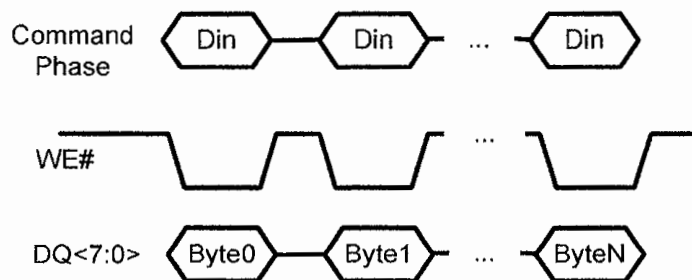


Fig. 2.21. "Din" cycle: $WE\#$ is low, all other inputs signals are low

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.

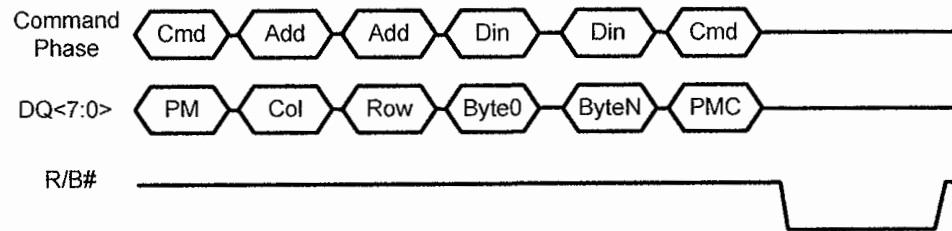


Fig. 2.22. Program command

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 μ s. 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.

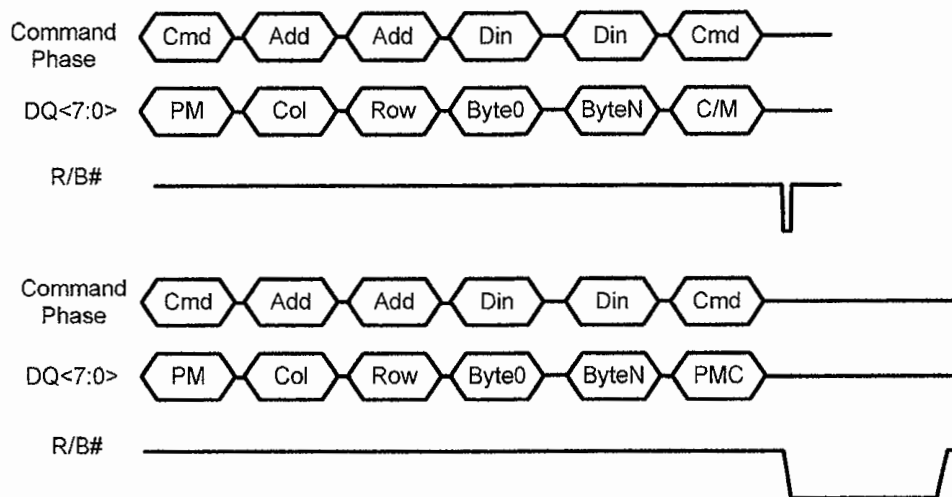


Fig. 2.23. Cache Program and Double-Plane Program commands

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.

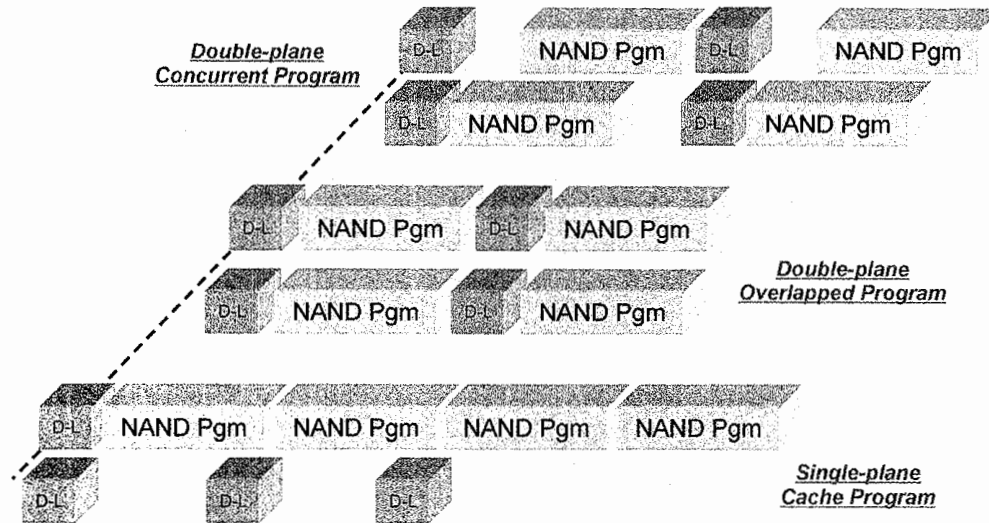


Fig. 2.24. Performance comparison among Cache Program, Overlapped Double Plane program and Concurrent Double Plane Program

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.

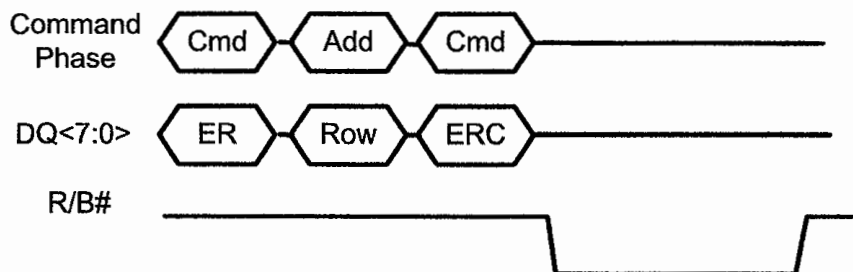


Fig. 2.25. Erase command

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.

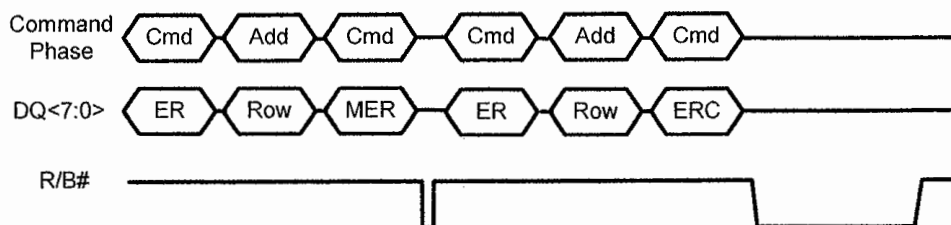


Fig. 2.26. Double-Plane Erase command

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.

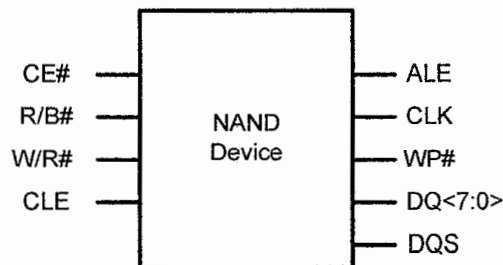


Fig. 2.27. Pinout of a NAND Flash supporting Source Synchronous Interface

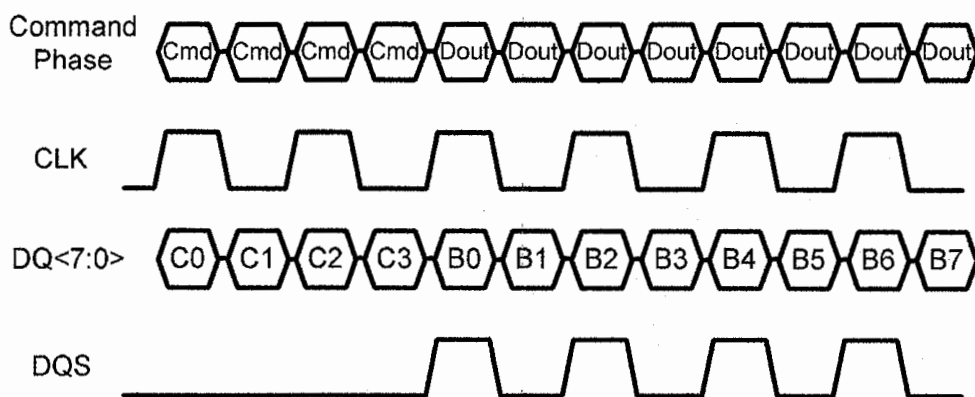


Fig. 2.28. Source Synchronous Interface DDR sequence

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.

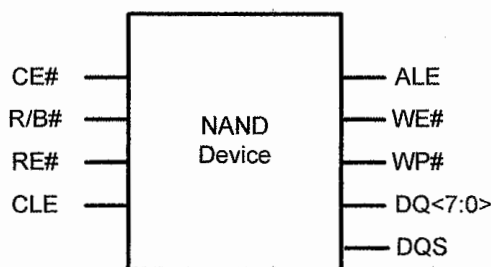


Fig. 2.29. Pinout of a NAND Flash supporting Toggle-Mode Interface

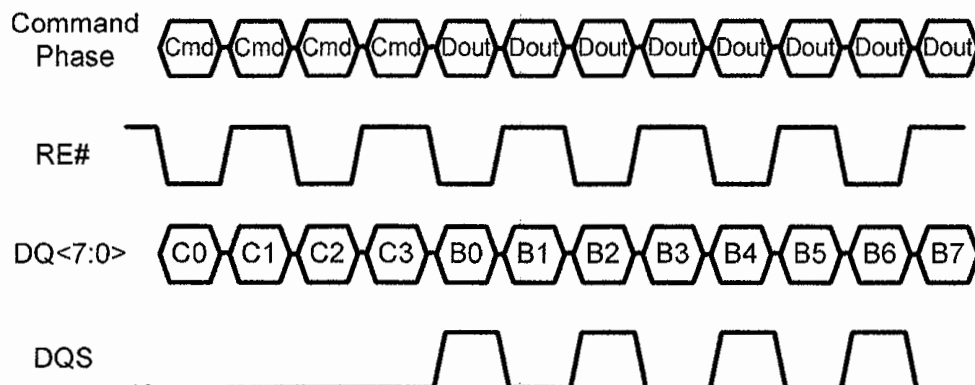


Fig. 2.30. Toggle-Mode DDR sequence

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 μ SD. 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 μ SD, 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 μ SD, considering that several dies are stacked, i.e. mounted one on top of each other.

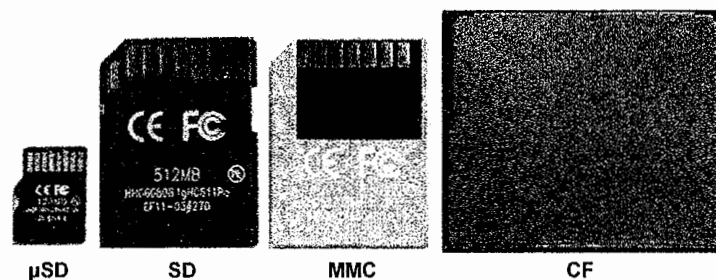


Fig. 2.31. Popular Flash card form factors

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).

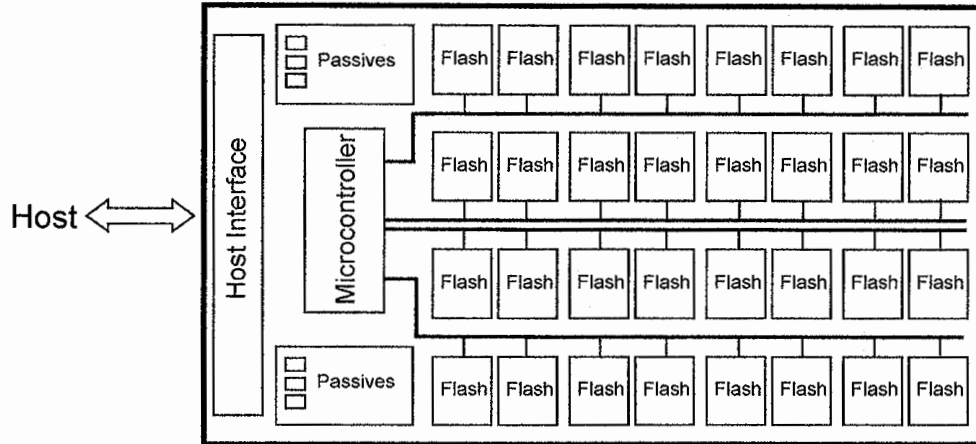


Fig. 2.32. Block diagram of a SSD

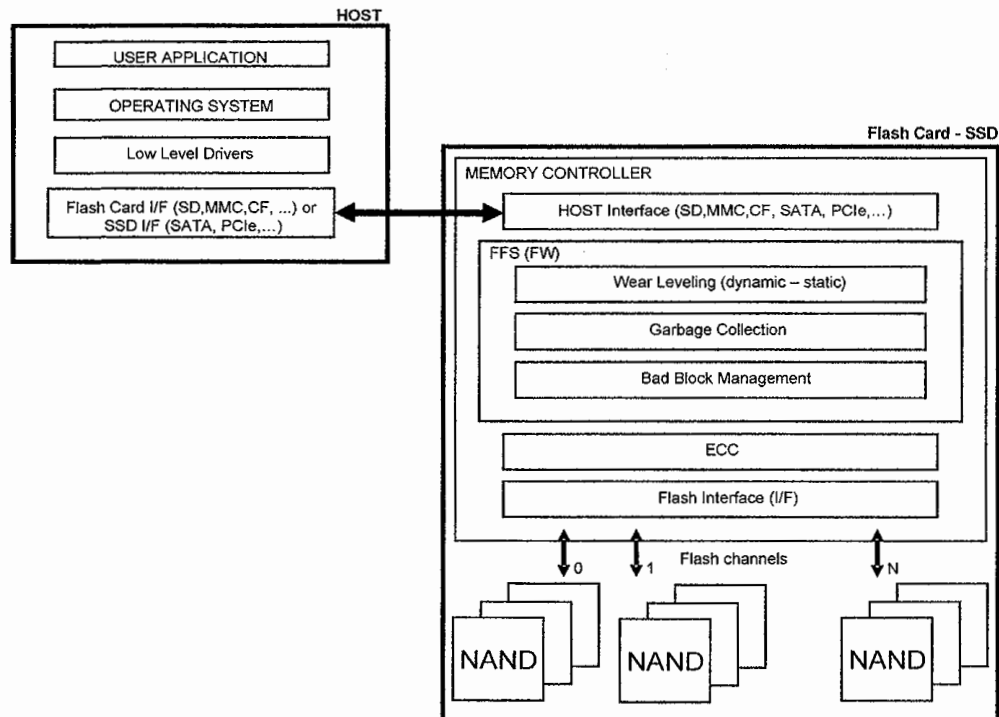


Fig. 2.33. Functional representation of a Flash card (or SSD)

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.

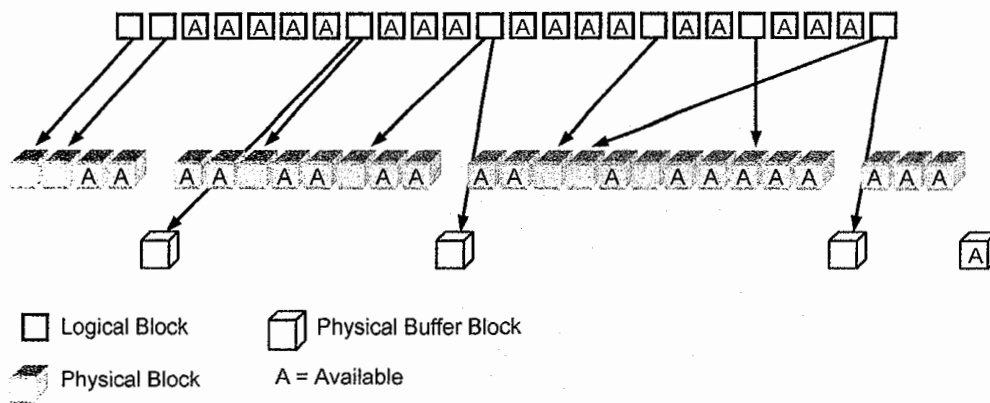


Fig. 2.34. Logical to physical block management

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.

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.

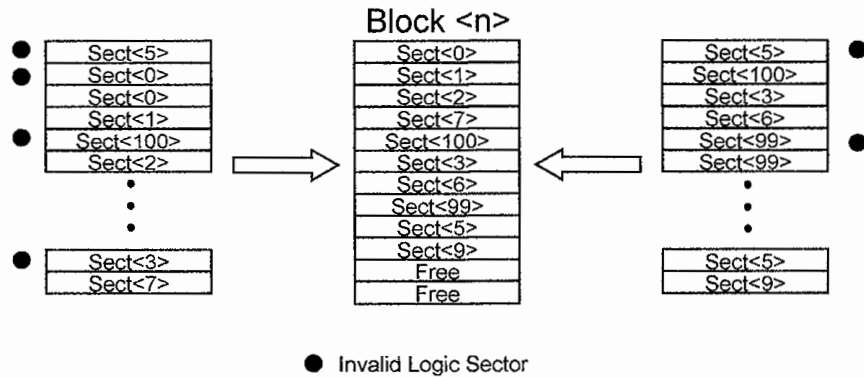


Fig. 2.35. Garbage collection

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.

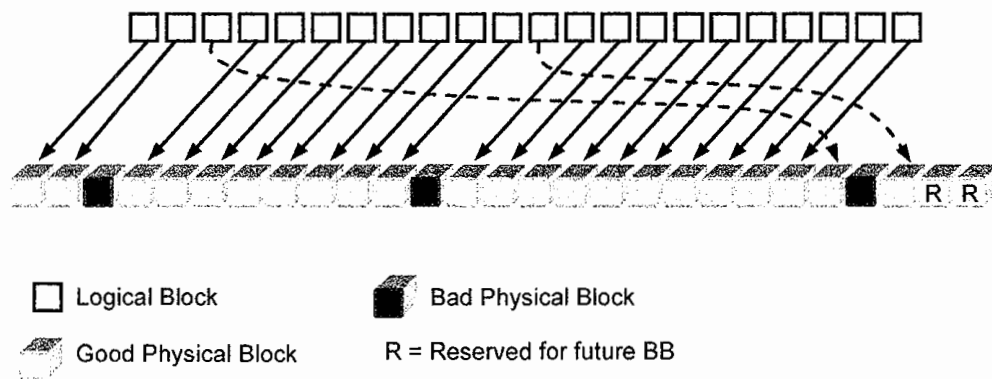


Fig. 2.36. Bad Block Management (BBM)

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:

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

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].

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 μ s, 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.

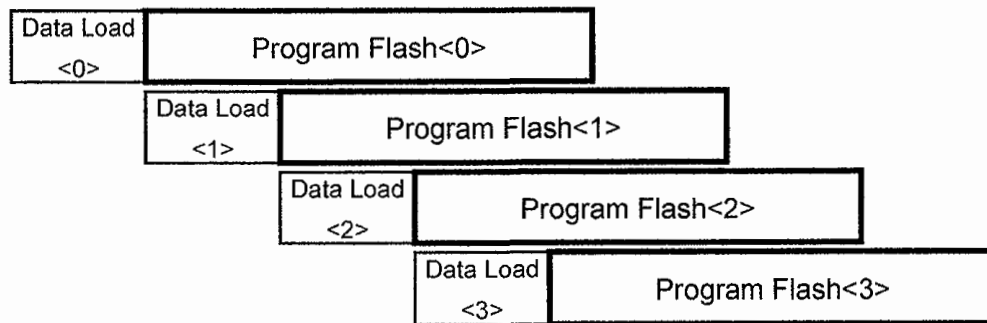


Fig. 2.37. Interleaved operations on one Flash channel

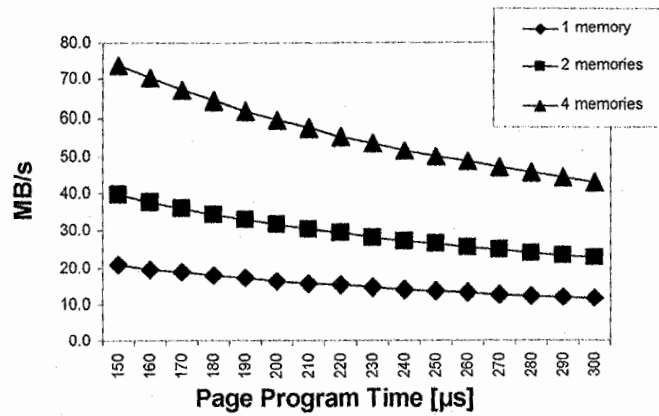


Fig. 2.38. Program throughput with an interleaved architecture as a function of the NAND page program time

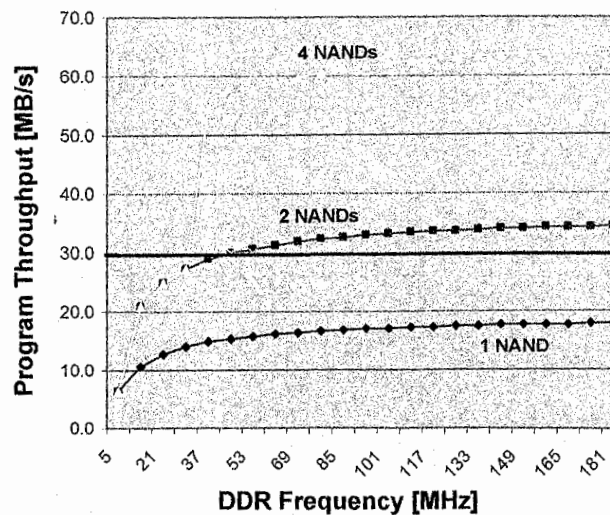


Fig. 2.39. Program throughput with an interleaved architecture as a function of the channel DDR frequency. 4 kB page program time is 200 μs

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

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.

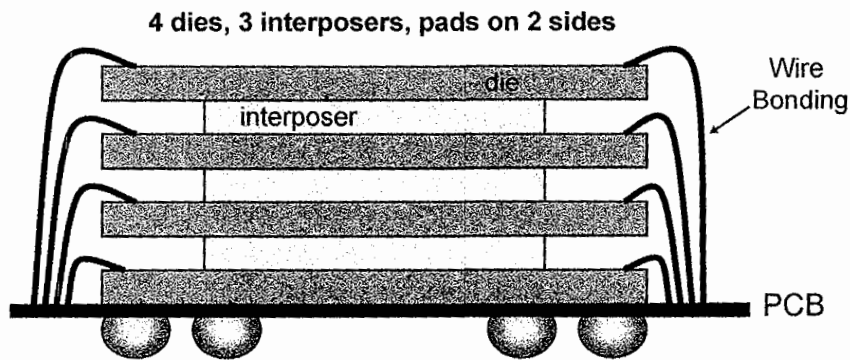


Fig. 2.40. Standard die stacking

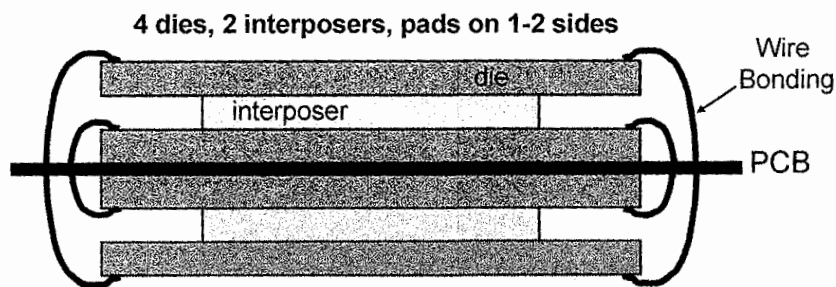


Fig. 2.41. Flipped die stacking

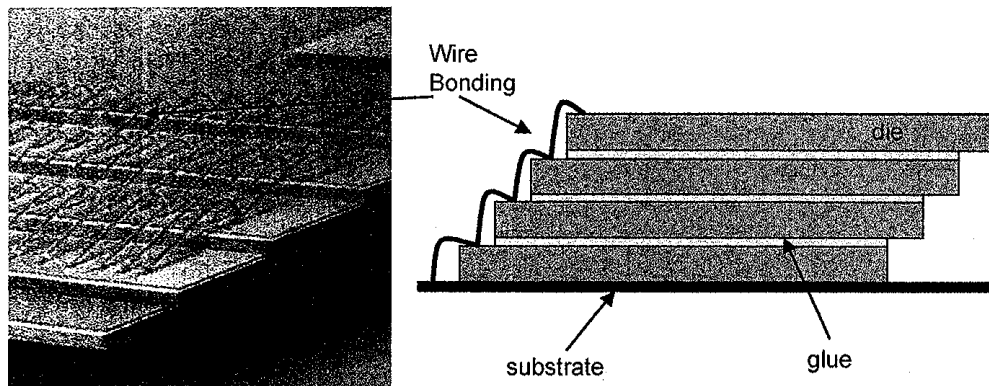


Fig. 2.42. Staircase die stacking: four dies, zero interposers, pads on one side

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.

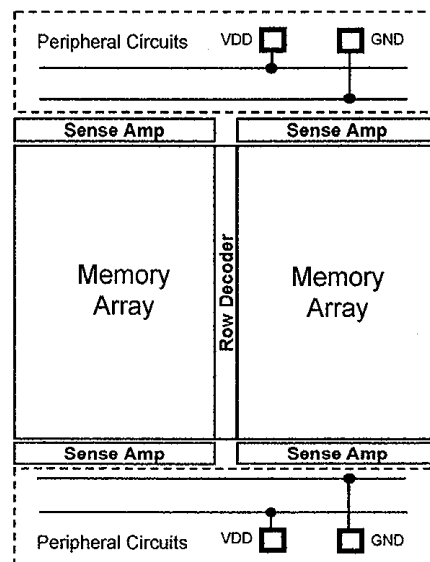


Fig. 2.43. Memory device with pads along opposite sides

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.

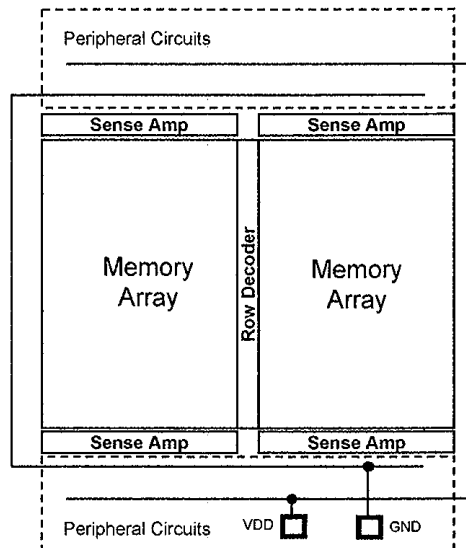


Fig. 2.44. Memory device with pads along one side

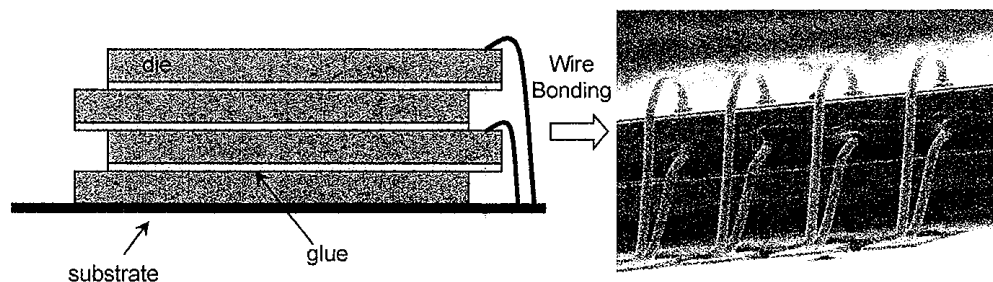


Fig. 2.45. "Snake" die stacking

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.

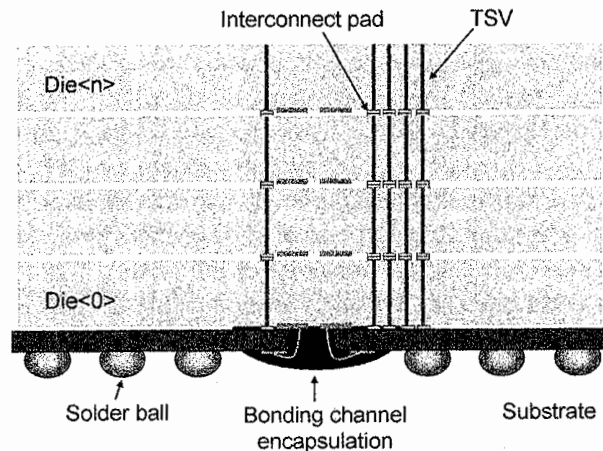


Fig. 2.46. Multiple die stacking with TSV technology

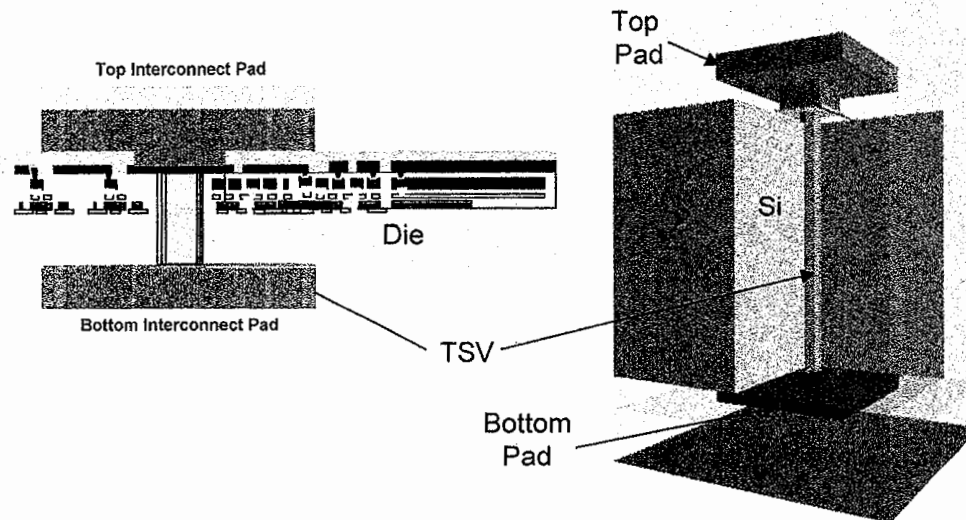


Fig. 2.47. Die with TSV (left) and TSV 3-D view (right)

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.

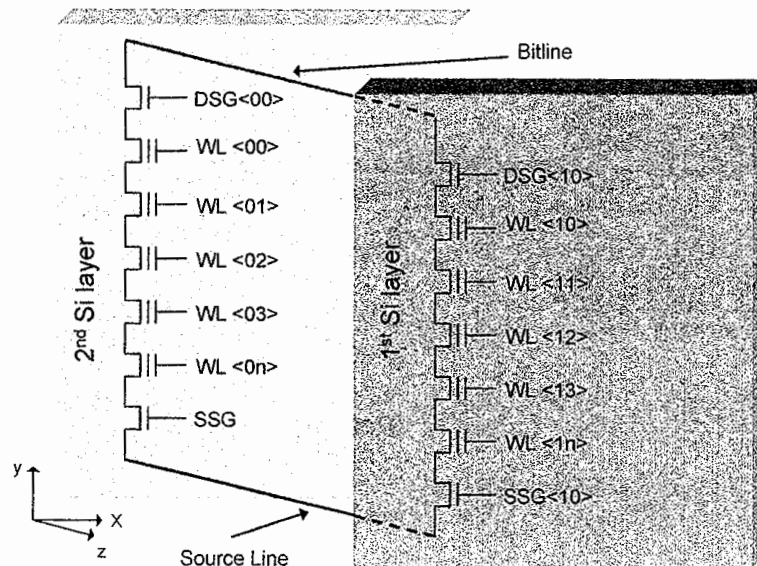


Fig. 2.48. Three-dimensional horizontal memory array

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.

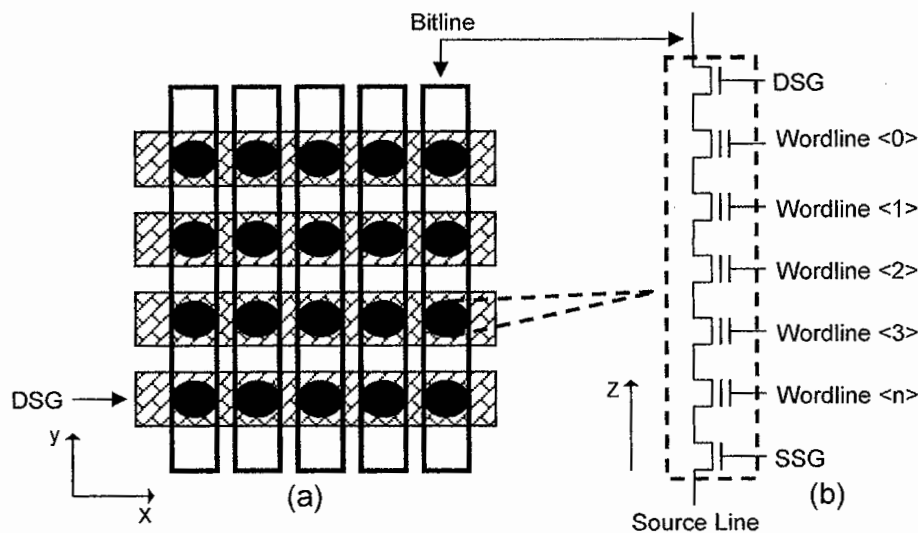


Fig. 2.49. Three dimensional vertical memory array: (a) top down view of 3-D vertical memory array; (b) equivalent circuit of the vertical NAND string

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-mm² 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 120mm² 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.

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.

3 Program and erase of NAND memory arrays

Christoph Friederich¹

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

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 *Floating-Gate Electron Tunneling MOS* (FETMOS) cell [2], it has gone through several structural innovations [3], becoming the *Self-Aligned Shallow Trench Isolation* (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 *Floating-Gate Tunneling Oxide* (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 *Inter Word line Dielectric* (IWD) isolate the floating gates from each other. The *Inter Poly Dielectric* (IPD) separates the poly silicon *Control Gate* (CG) from the poly silicon *Floating Gate* (FG). It is typically a $\text{SiO}_2 - \text{Si}_3\text{N}_4 - \text{SiO}_2$ (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 *Tunneling Dielectric* (TD) and *Coupling Dielectric* (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.

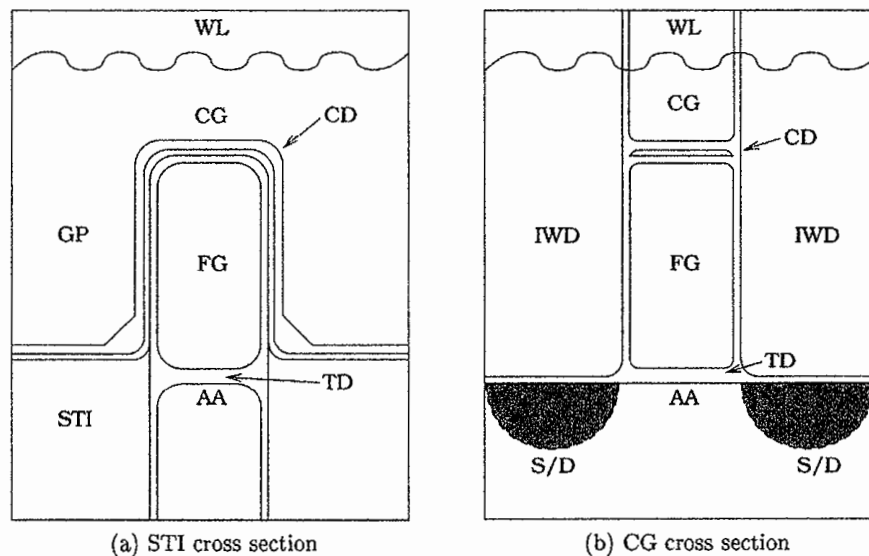


Fig. 3.1. The ONO IPD floating gate cell

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 *Equivalent Oxide Thickness* (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.

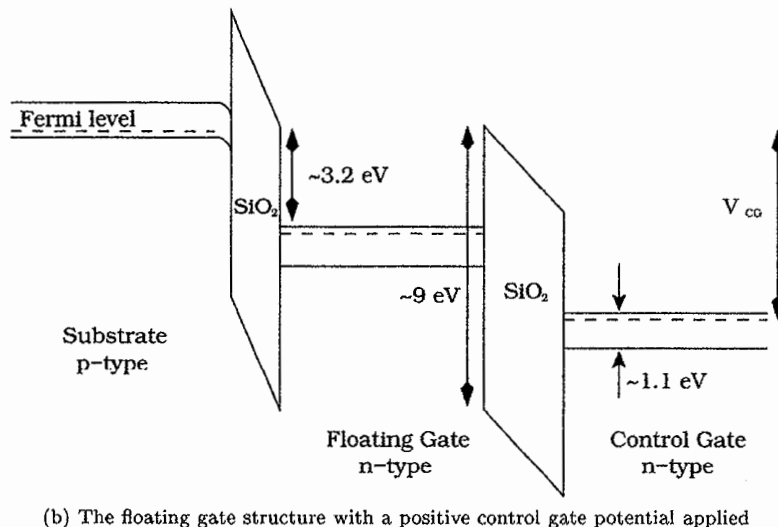
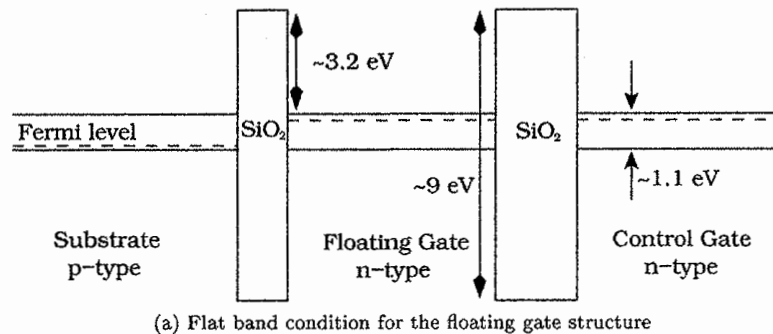


Fig. 3.2. Band diagrams of a floating gate cell

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:

$$C_i = \frac{\delta Q_i}{U_i} = \frac{\delta Q_i}{\psi_{FG} - \psi_i} \quad (3.1)$$

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

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

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

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

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

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

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

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

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

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

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.

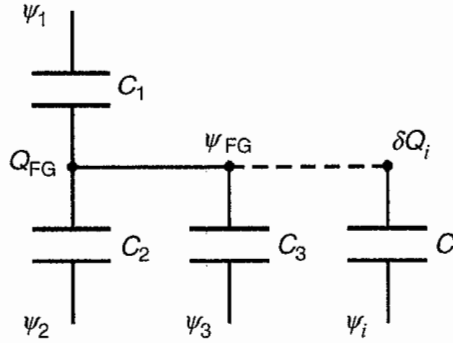


Fig. 3.3. Capacitive model of the floating gate cell

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} :

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

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

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

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 Fowler–Nordheim (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]:

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

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

E	[MV/cm]	Electric field at the injecting interface
Φ_B	[eV]	Potential barrier at the interface
$\hbar$	$1.06 \cdot 10^{-34}$ [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 $t(\Delta\Phi_B)$ and $v(\Delta\Phi_B)$ which are both tabulated elliptic integrals as a function of the normalized barrier lowering $\Delta\Phi_B$ [15]:

$$\Delta\Phi_B = \frac{1}{\Phi_B} \sqrt{\frac{q^3 E}{4 \pi \epsilon_{ox}}} \quad (3.9)$$

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

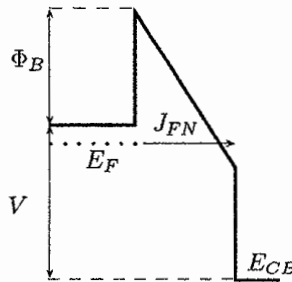


Fig. 3.4. Fowler-Nordheim tunneling through a potential barrier

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:

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

where

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

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

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 *Incremental Step Pulse Programming* (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.

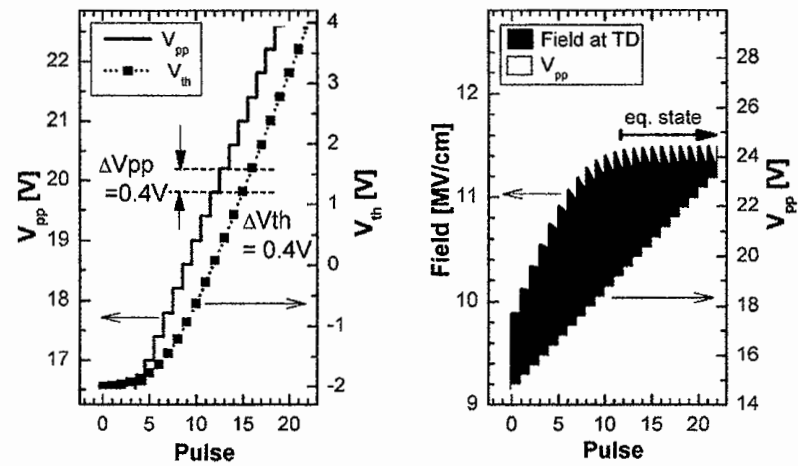
The reason for the equivalence of ΔV_{th} and V_{step} is the self limitation of the tunneling current. The already tunneled charges are decreasing the electric field between the gate and the channel (Fig 3.5b) and therefore the tunneling current decreases.

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

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

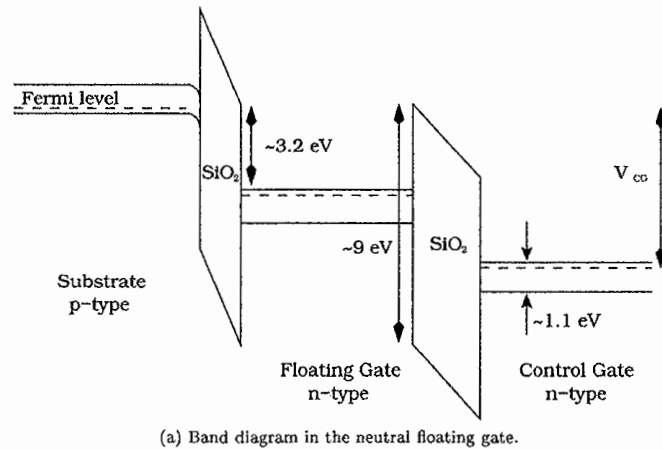
In the next program step s the electric field is raised again by applying the higher programming voltage $V_{pp,s} = V_{pp,s-1} + V_{step}$ to the control gate. This feedback loop enables an equilibrium state for the mean electric field and therefore also for the mean tunneling current. This process 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.

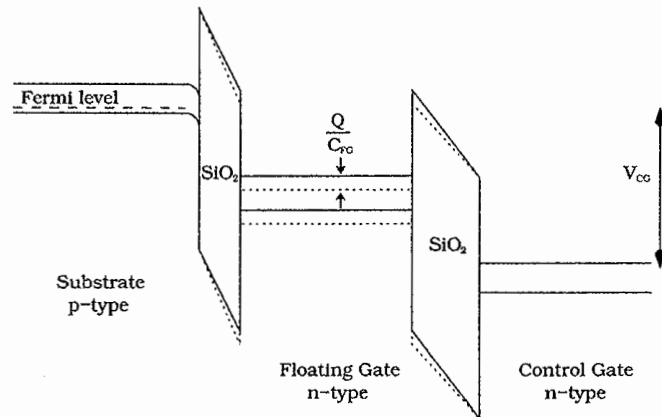


(a) Constant V_{th} shift during incremental gate stepping. (b) Electric field across tunneling dielectric over program pulses.

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

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.

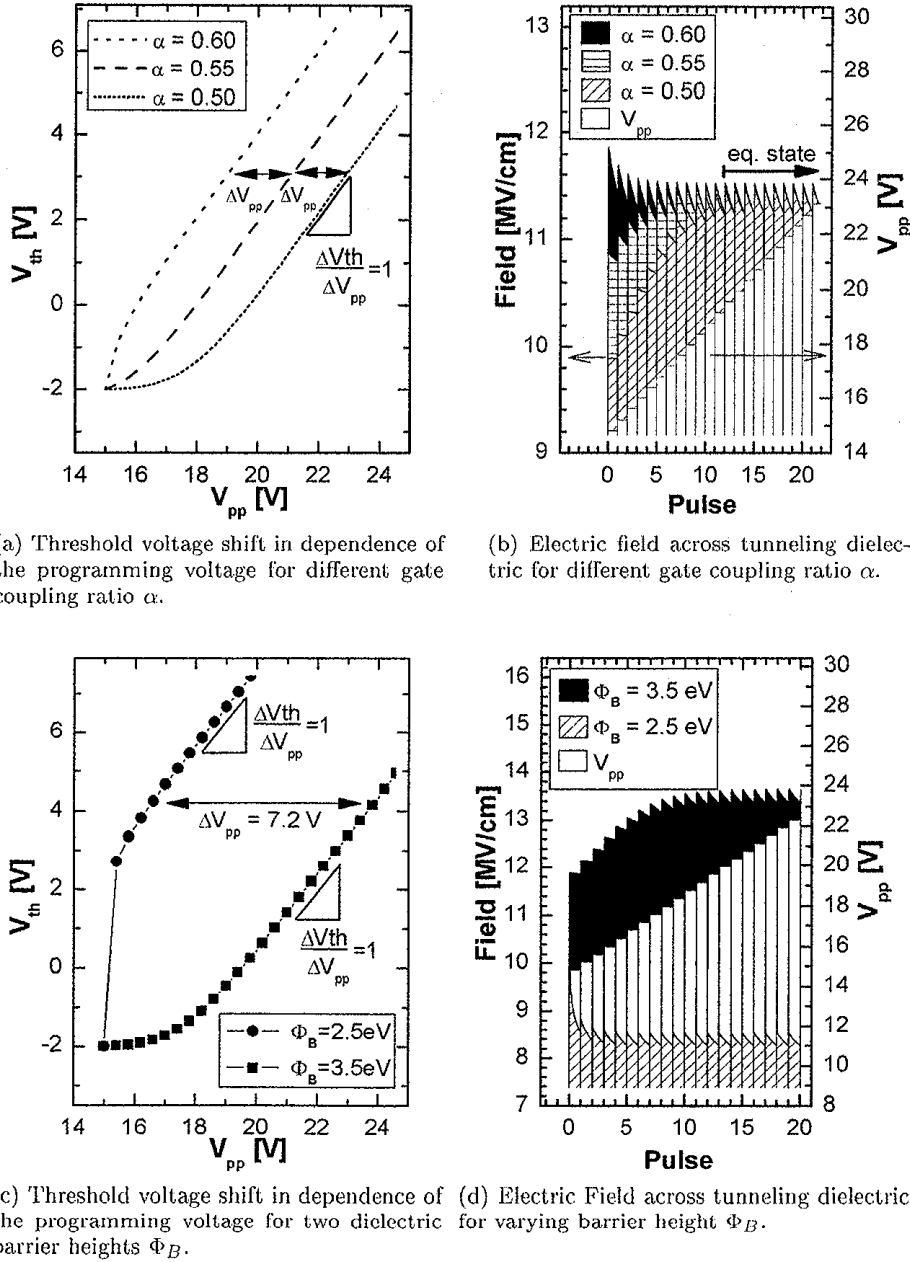


Fig. 3.7. Interaction between incremental step pulse programming scheme and cell parameters

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 Source Line (SL). The row decoders control the select lines and word lines in the block. The NAND strings in a block are selected by the String Select Line (SSL) and the Ground Select Line (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

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.

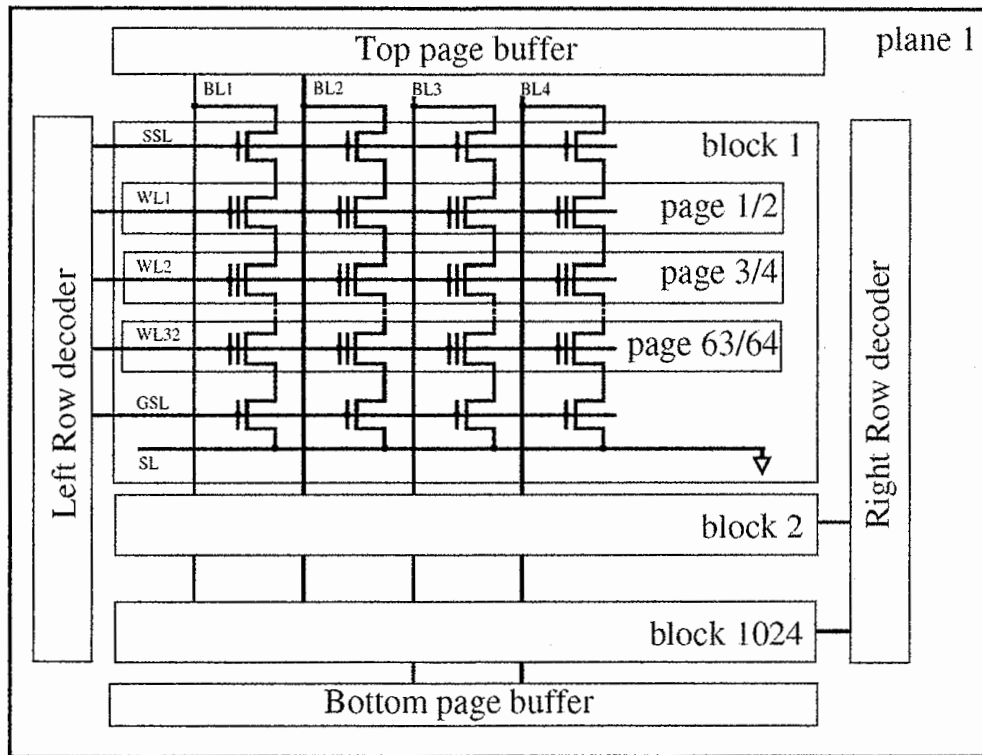


Fig. 3.8. Schematic NAND array architecture

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.

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-hungry. 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

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

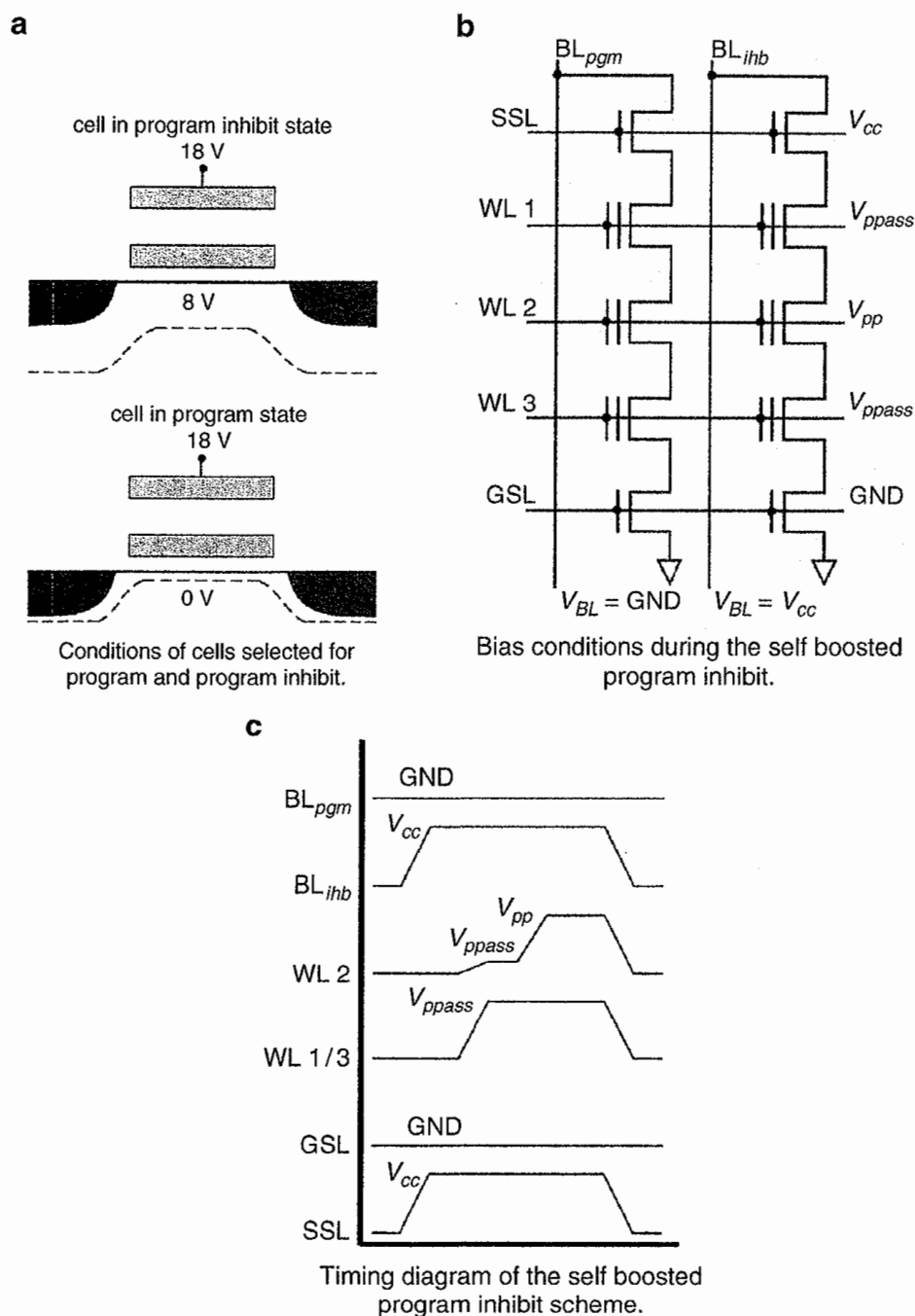


Fig. 3.9. Self-boasted program inhibit scheme

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

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

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:

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

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

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.

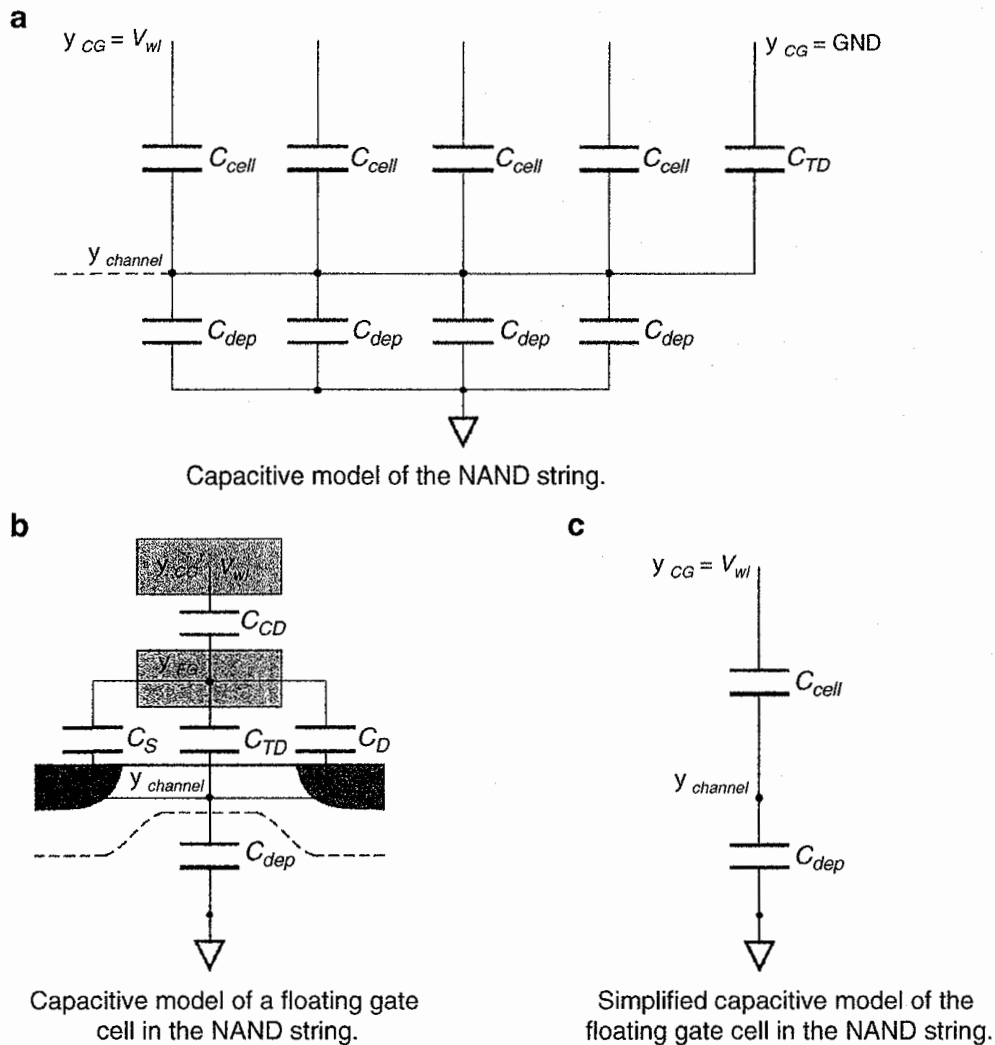


Fig. 3.10. Capacitive models of the NAND string

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

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 Number Of Programs (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 $V_{pp} - V_{ch}$.

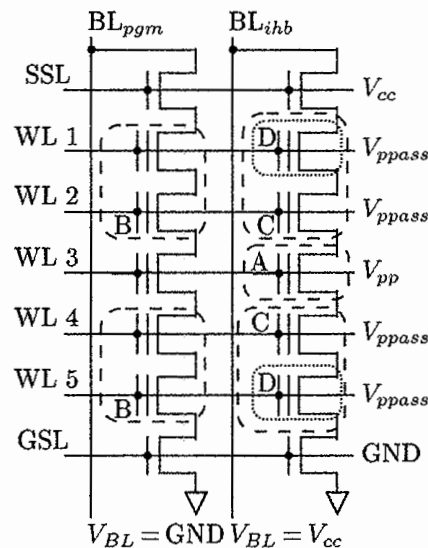


Fig. 3.11. Disturb during the program operation in NAND array architecture

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

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 $V_{ch,boost}$:

$$V_{ch,boost} = V_{ch,pre} + \Delta V_{ch} \quad (3.14)$$

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

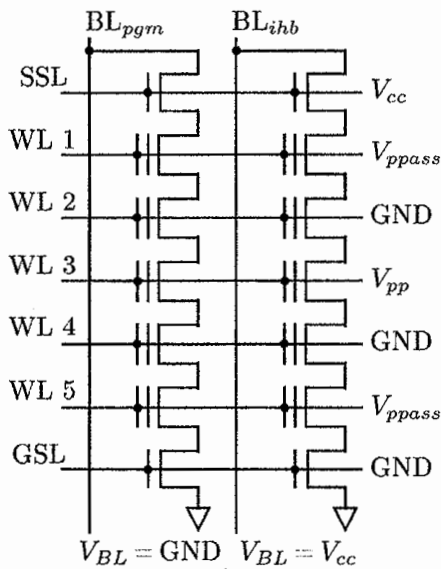
In contrast to the global SBPI schemes outlined in the previous section and in Sect. 3.4.1, local SBPI schemes [36, 37] are mostly independent from string length and V_{ppass} . With these schemes, the program inhibited cell is isolated from its NAND string by the adjacent cells. Of course, this is true 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:

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

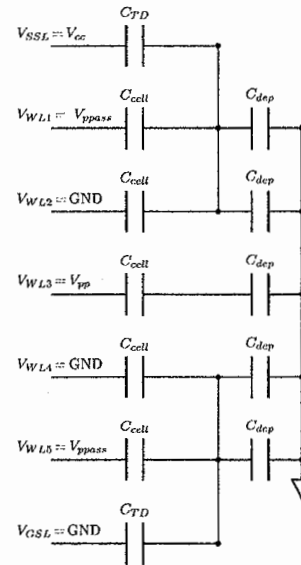
Although the local SBPI scheme features higher boosting efficiency, it also faces some problems. Especially the low voltage applied to the shielding word line

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.

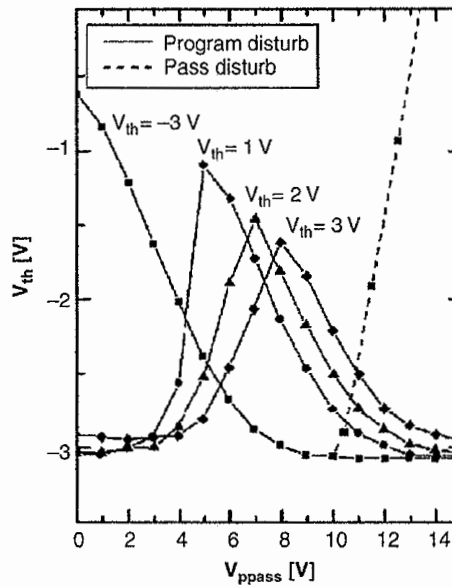


Fig. 3.13. Background pattern dependent program disturb. Data is extracted from [25]

Dependent on the V_{th} of the cells in the string, the precharged potential $V_{ch,pre}$ propagates into the string. During the self-booted 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.

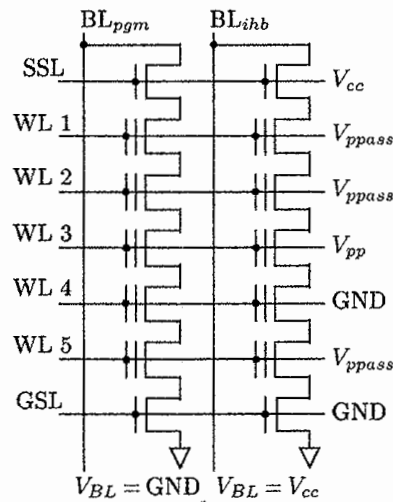


Fig. 3.14. Bias conditions during an asymmetric self-booted program inhibit

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

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 *Models for Cell System Interaction* (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

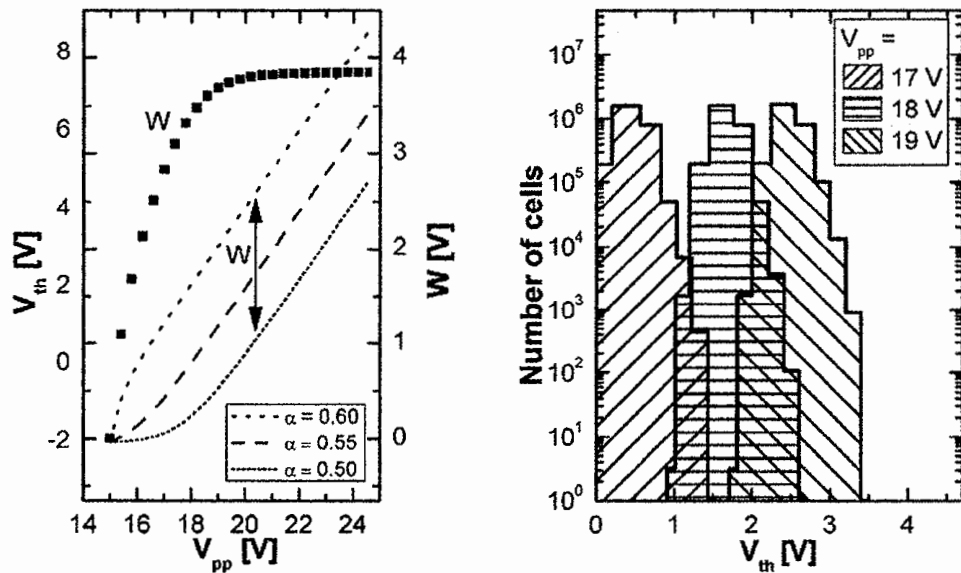
As stated in Sect. 3.2, the programming mechanism used in NAND Flash is FN tunneling. The most important cell parameter for FN tunneling is the electric field across the tunneling dielectric. The gate coupling ratio α defines the ratio between the applied voltage at the control gate and the electric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

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

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

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 $\Delta V_{th,i}$ 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 $\Delta V_{th,i}$ is related to the user data, which in most cases can be considered as random.

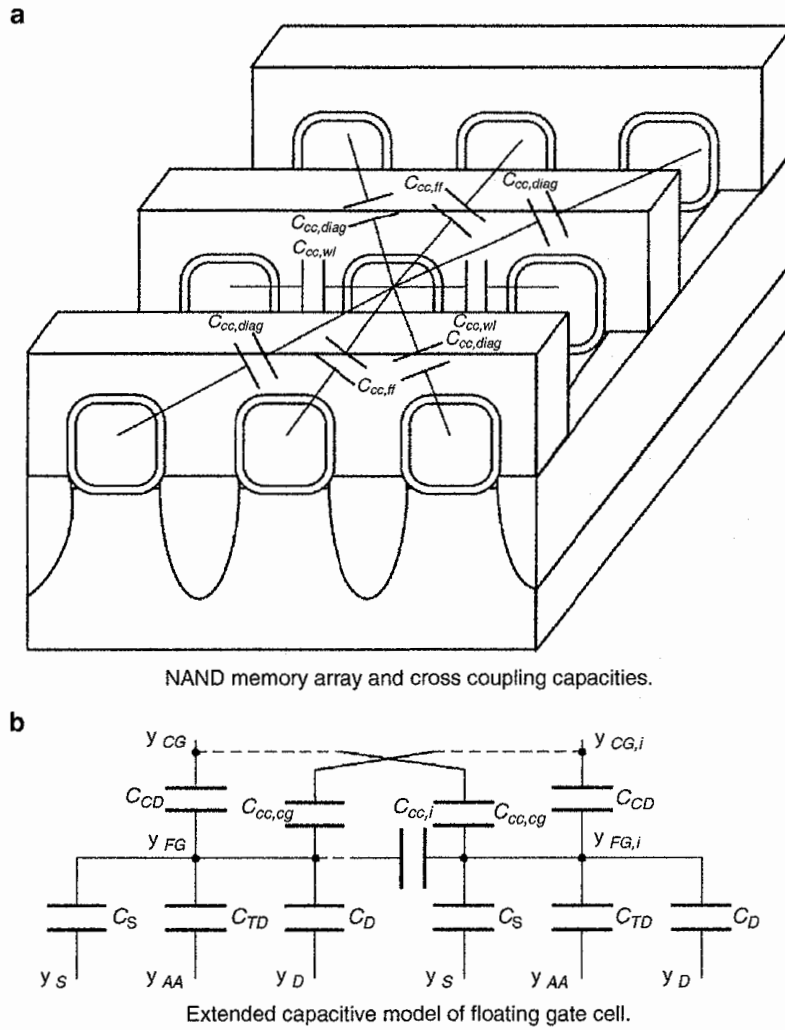


Fig. 3.16. Floating gate cross coupling in NAND architecture

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:

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

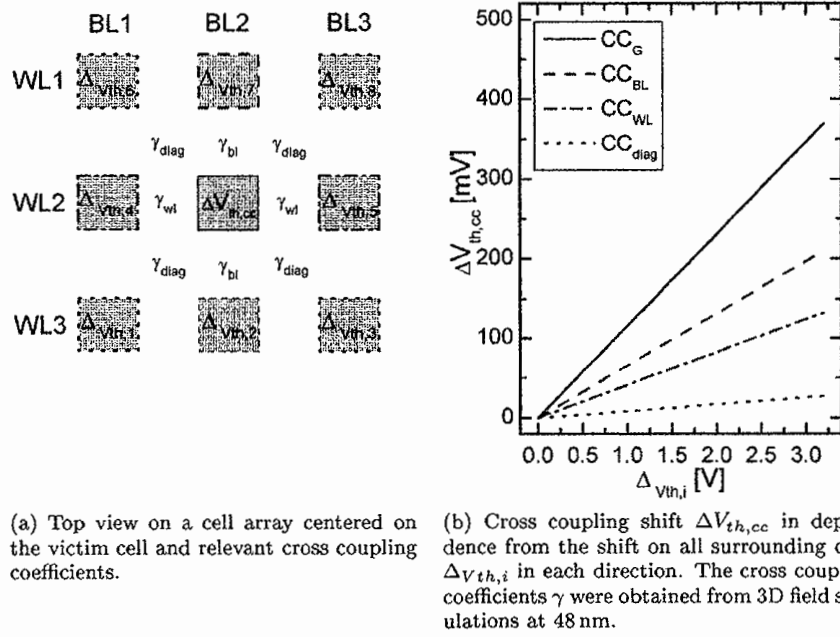


Fig. 3.17. Maximum amount of cross coupling in NAND architecture

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 $\sigma_{\Delta V_{th}}$:

$$\sigma_{\Delta V_{th}} = \sqrt{\frac{q}{C_{CD}}} \cdot \Delta V_{th} \quad (3.18)$$

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 $\sigma_{\Delta V_{th}}$ [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.

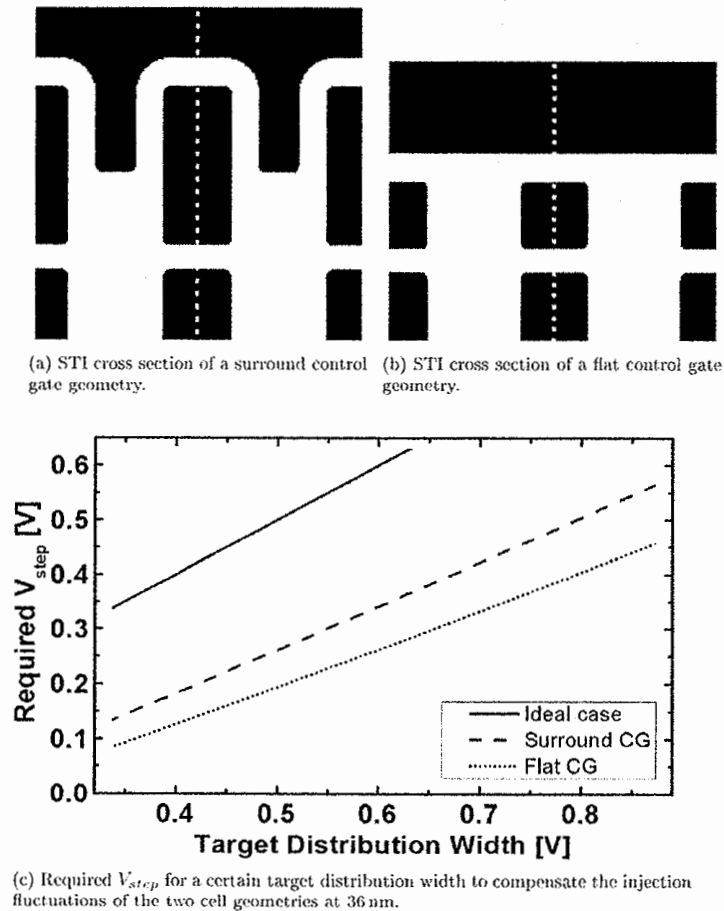


Fig. 3.18. Impact of cell geometries on the memory system

Models for Cell–System Interaction (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]:

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

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 μm^2 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.

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 SiO₂," *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 V_t 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 (Q_{bd}): 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.

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.

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 Source/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," *IEEE Electron Device Letters*, vol. 23, no. 5, pp. 264–266, May 2002.

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.

4 Reliability issues of NAND Flash memories

C. Zambelli¹, A. Chimenton² and P. Olivo³

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

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 ($\text{SiO}_2\text{--Si}_3\text{N}_4\text{--SiO}_2$) for interpoly oxides.

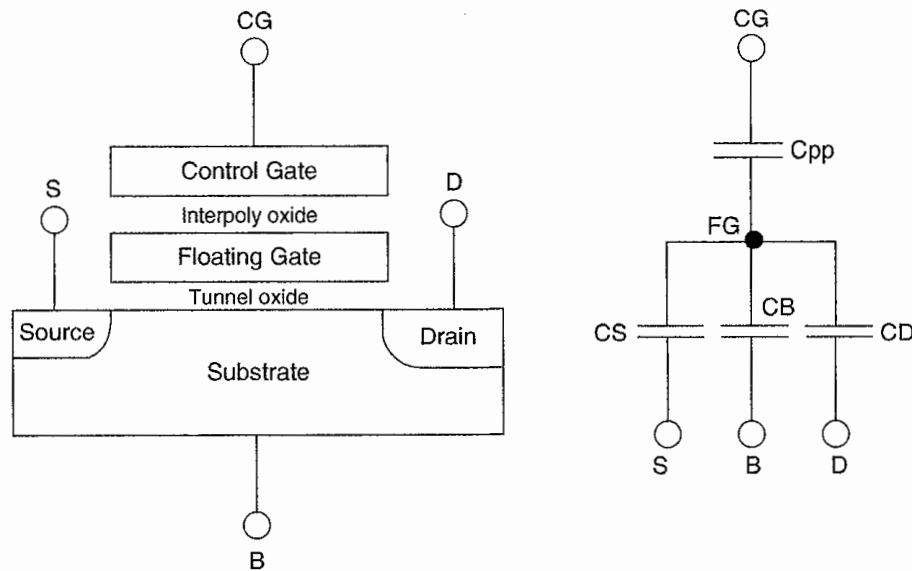


Fig. 4.1. FLOTOX device and its equivalent capacitance model

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 ($E_{\text{OX}} \approx 10 \text{ MV/cm}$) 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.

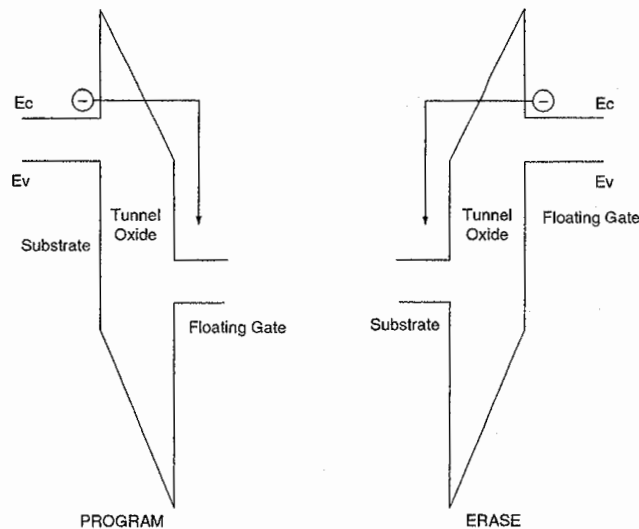


Fig. 4.2. Band diagram sketch of tunneling effect

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

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.

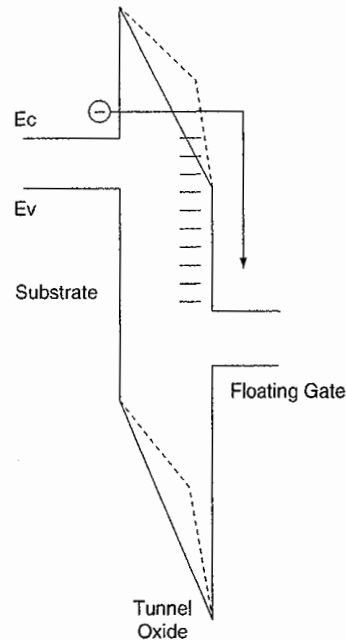


Fig. 4.3. Band diagram during a program operation: without traps (solid lines in the oxide region) and with traps (dashed lines in the oxide regions)

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.

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].

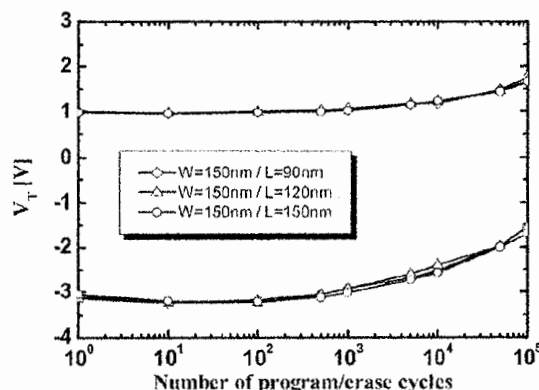


Fig. 4.4. Threshold voltage degradation during cycling of NAND Flash with different geometrical features [7]

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 .

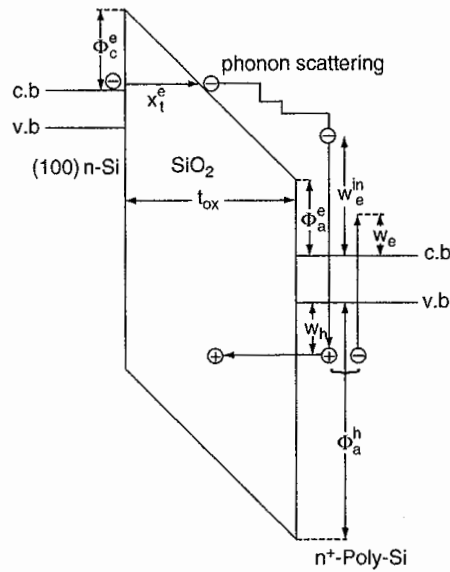


Fig. 4.5. Band diagram of HHI (Hot Hole Injection) phenomenon on a NAND Flash memory structure. The substrate is Si with orientation $\langle 100 \rangle$ and the floating gate is n+ doped Poly-Si [8]

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].

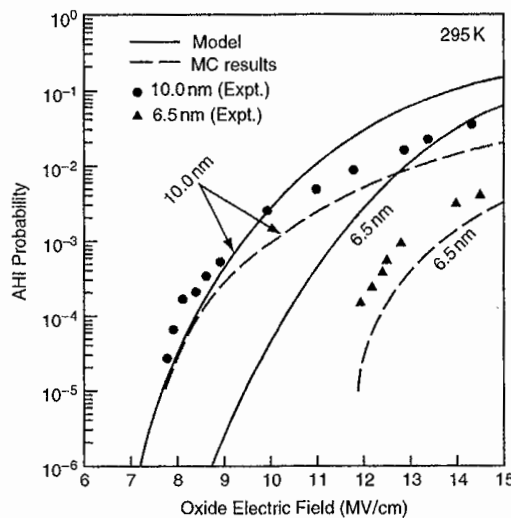


Fig. 4.6. AHI probability calculated with triangular hole-energy barrier approximation as a function of uniform oxide electric field with oxide thickness as a parameter [8]

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

$$\alpha_h = \sum P_h \theta_h \quad (4.1)$$

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.

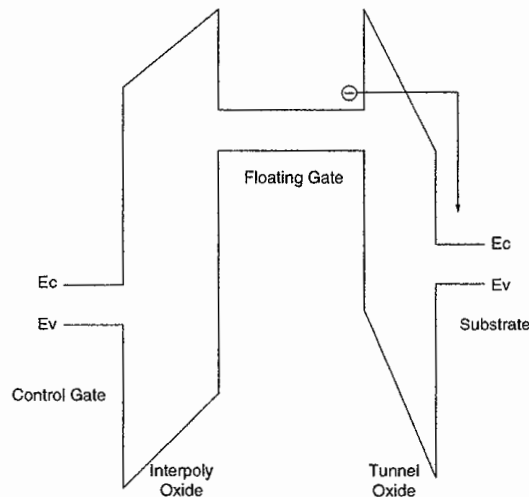


Fig. 4.7. Band diagram of the cell when programmed and not biased. The main mechanism for data loss is tunneling through the tunnel oxide

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.

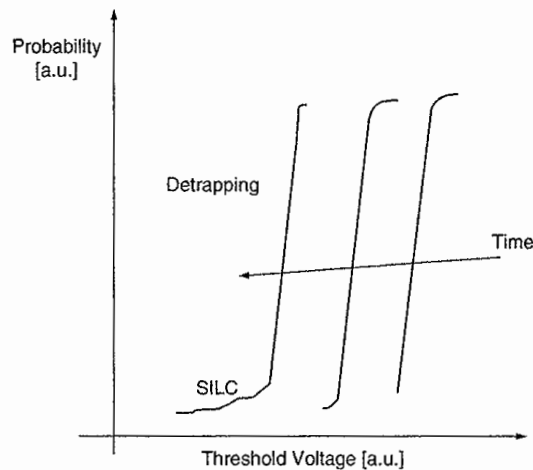


Fig. 4.8. Cumulative distribution of a NAND array on Program state. Both detrapping and SILC effects are appreciable on the time evolution

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-SiO₂ 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.

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

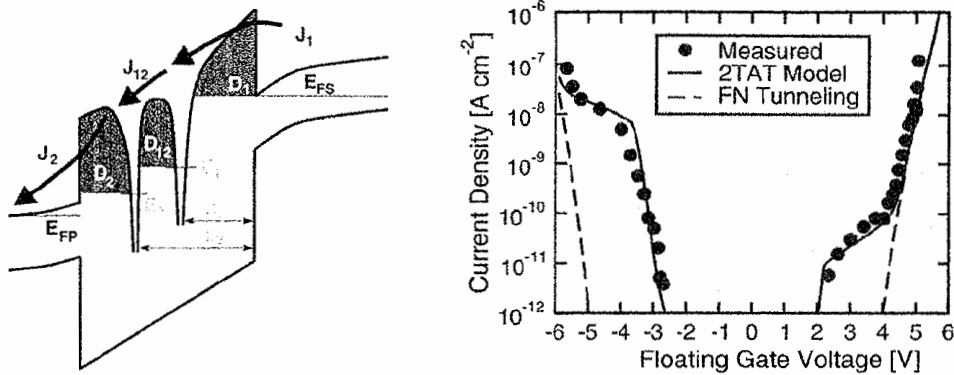


Fig. 4.9. Two traps assisted tunneling (2TAT) band diagram (left) and comparison between the SILC-2TAT model result versus the classical Fowler-Nordheim theory (right) [14]

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

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

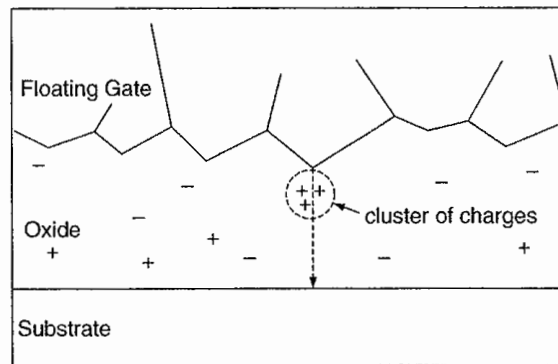


Fig. 4.10. Charge cluster model used for explanation of fast-programming bits [15]

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 a 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].

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.

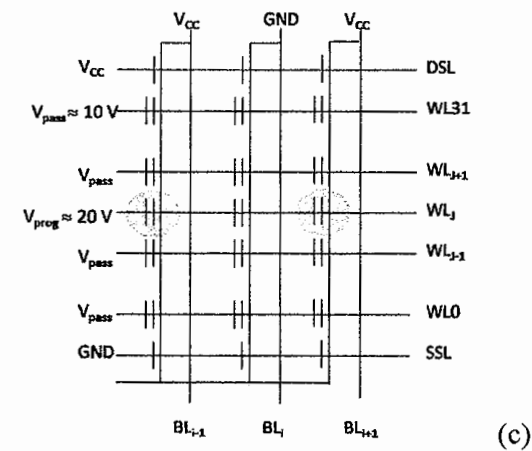
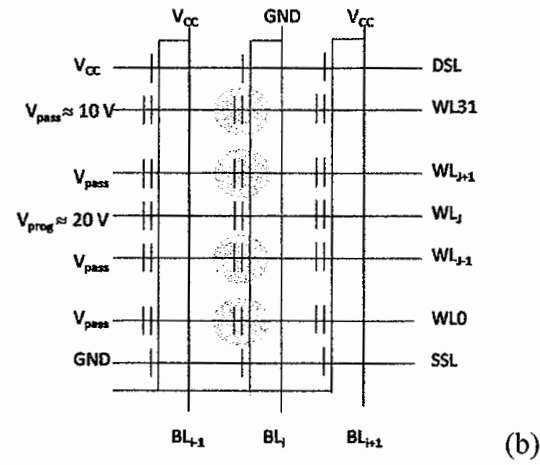
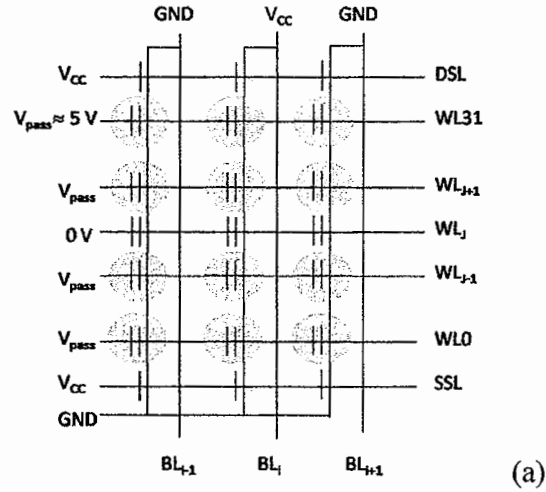


Fig. 4.11. Representation of read disturb (a), pass disturb (b) and program disturb (c) in a NAND Flash array. The cells potentially affected by the disturbs are marked in gray

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 behaves 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 carriers 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.

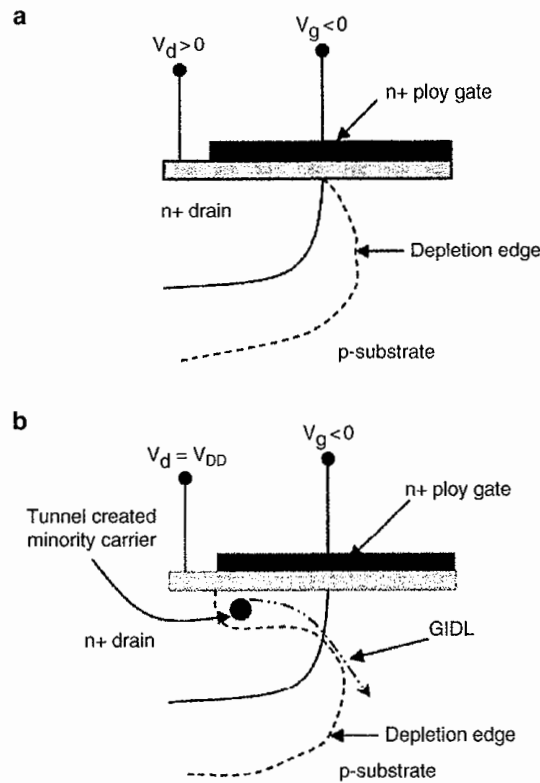


Fig. 4.12. Condition for the depletion region near the gate-drain overlap region of a nMOS transistor when the surface is accumulated with a low negative gate bias (a), and n+ region is depleted or inverted with high negative gate bias (b) [19]

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.

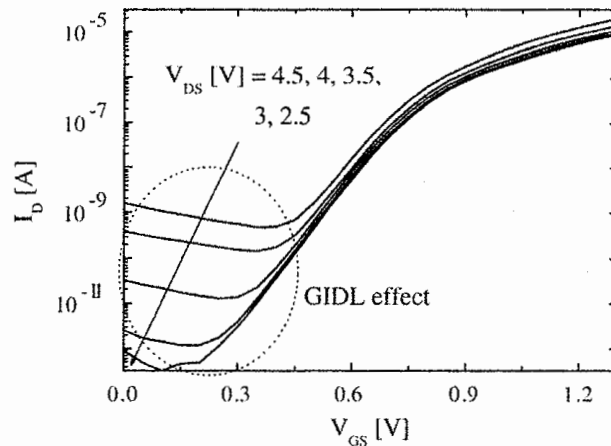


Fig. 4.13. Leakage current measured in a nMOS transistor for different drain-source voltages V_{DS} in the subthreshold regime. When V_{GS} approaches 0 V, for high V_{DS} values the leakage current is due to GIDL [20]

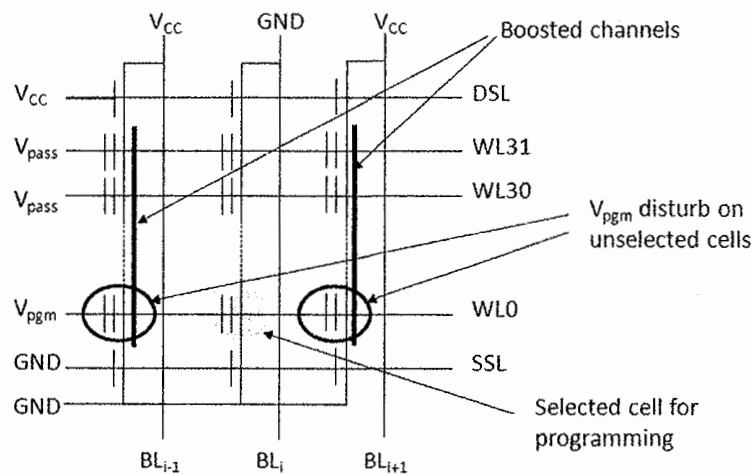


Fig. 4.14. Bias conditions possibly activating GIDL effects on SSL transistors belonging to columns BL_{i-1} and BL_{i+1}

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.

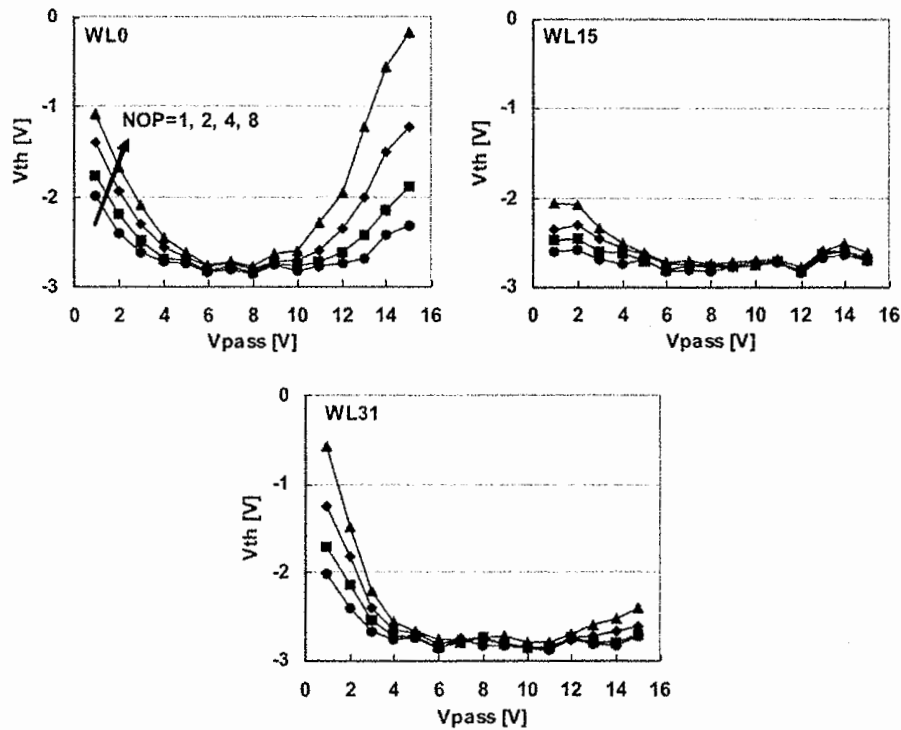


Fig. 4.15. Measured program disturbs characteristics for 3 cells belonging to WL0, WL15 and WL31, respectively [21]. NOP indicates the number of partial programming when multiple writing of a word line is allowed for specific applications. It can be observed that, for higher values of V_{pass} , the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

$$I_D \propto \mu_n Q_n \Rightarrow \frac{\Delta I_D}{I_D} = \frac{\Delta Q_n}{Q_n} + \frac{\Delta \mu_n}{\mu_n} \quad (4.2)$$

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

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):

$$\Delta V_T = \frac{q}{C_{OX}WL\alpha_G} \quad (4.3)$$

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.

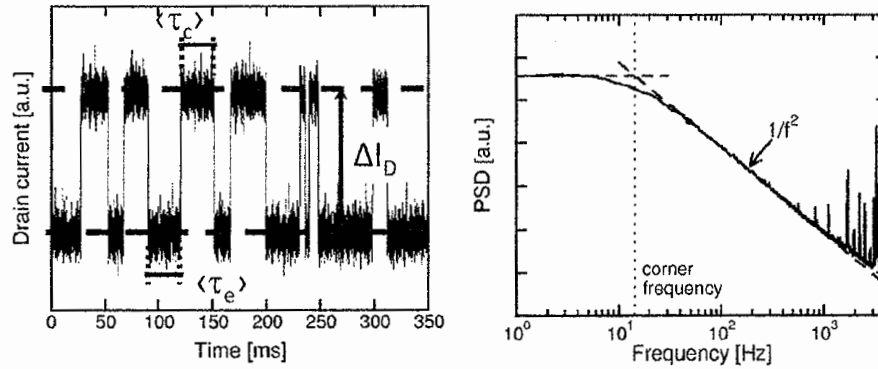


Fig. 4.16. Representation of the time domain plot (left) and power spectral density (right) of the RTN of a MOSFET device [14]

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.

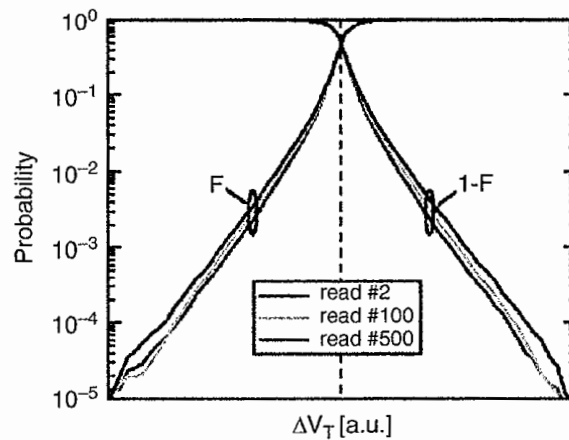


Fig. 4.17. Drift issue of the random telegraph noise on a read cycled NAND device [14]

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).

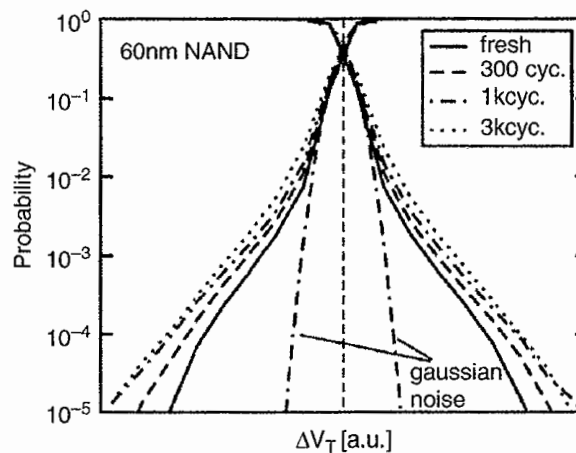


Fig. 4.18. ΔV_T distributions F and $1 - F$ as a function of the cycle number. The Gaussian noise contribution to the measured ΔV_T is also shown [26]

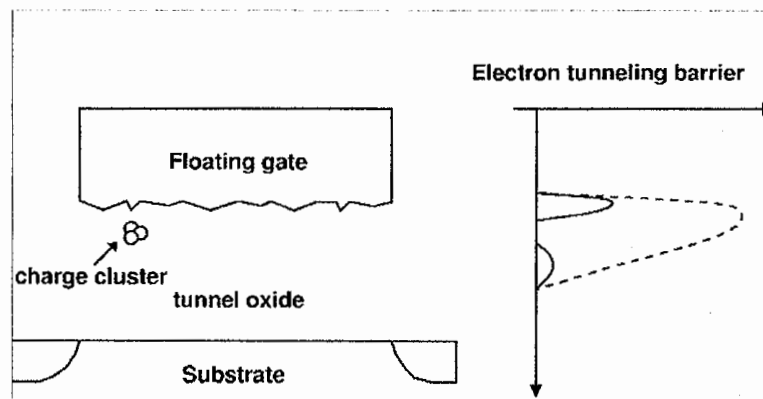


Fig. 4.19. Left: charge cluster model in which a positive charge cluster is located close to the floating gate of a cell in a NOR architecture. Right: electron tunneling barriers of silicon dioxide seen by electrons at the bottom of the floating gate conduction band in presence (solid line) or absence (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).

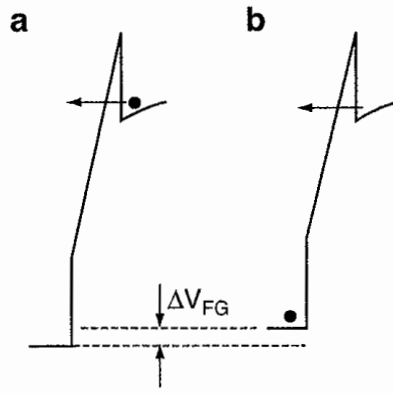


Fig. 4.20. Electrostatic effect on the injection process: the Floating Gate (FG) potential changes when an electron is injected from the substrate (a) to the FG (b), reducing the tunnel oxide field and the tunneling current [30]

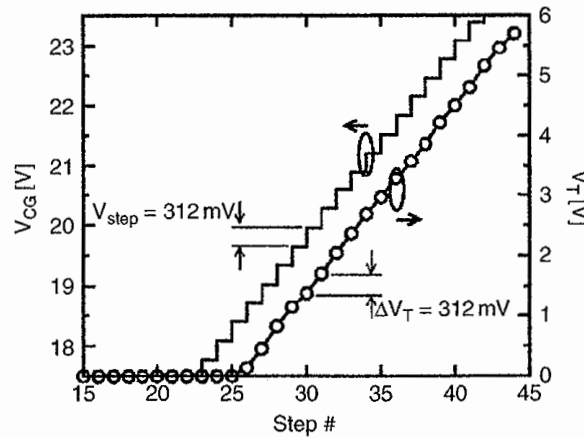


Fig. 4.21. Staircase V_{CG} waveform and corresponding V_T transient in a NAND cell under a constant-current FN program operation [30]. The step duration is 11 μ s

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

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 $\sigma_{\Delta V_T}$ evaluation) remain unaffected, as shown in Fig. 4.22.

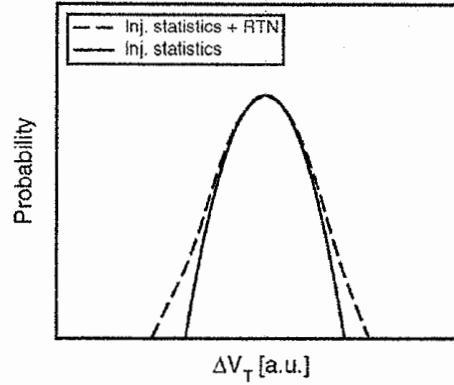


Fig. 4.22. Probability distributions of the injection statistics with and without the contribution of the RTN [30]

Furthermore, experiments shown that $\sigma_{\Delta V_T}$ 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:

$$\sigma_{\Delta V_T} = \frac{q}{C_{pp}} \sigma_n \quad (4.4)$$

$$\sigma_n = \sqrt{n}$$

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 approximatively equal to the threshold voltage variation, it is possible to write the formula of the spread under low variation of the threshold voltage as:

$$\sigma_{\Delta V_T} = \frac{q}{C_{pp}} \sqrt{n} = \sqrt{\frac{q}{C_{pp}} \overline{\Delta V_T}} \quad (4.5)$$

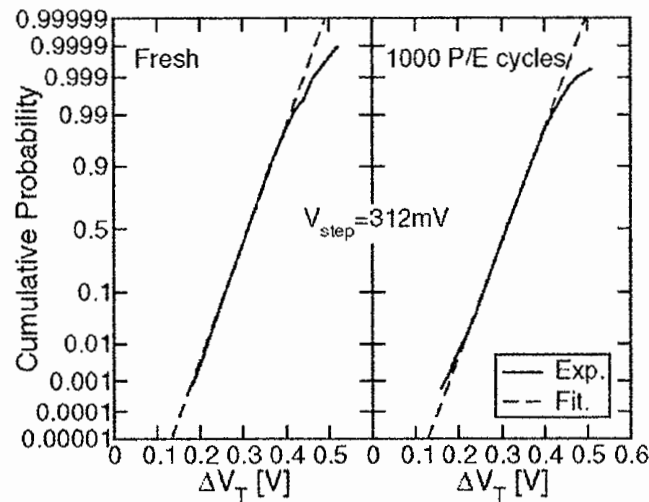


Fig. 4.23. Comparison between two ΔV_T probability distributions. The left one has been extracted on a fresh NAND device, while the right one has been extracted after 1,000 Program/Erase cycles on the same device [30]

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/SiO₂ 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 $\equiv\text{Si-H}$ and $\equiv\text{Si-O}$ bonds by hot holes injected from Si substrate is followed by the induction of serious damages in both the bottom oxide and the Si/SiO₂ interface as is the case

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].

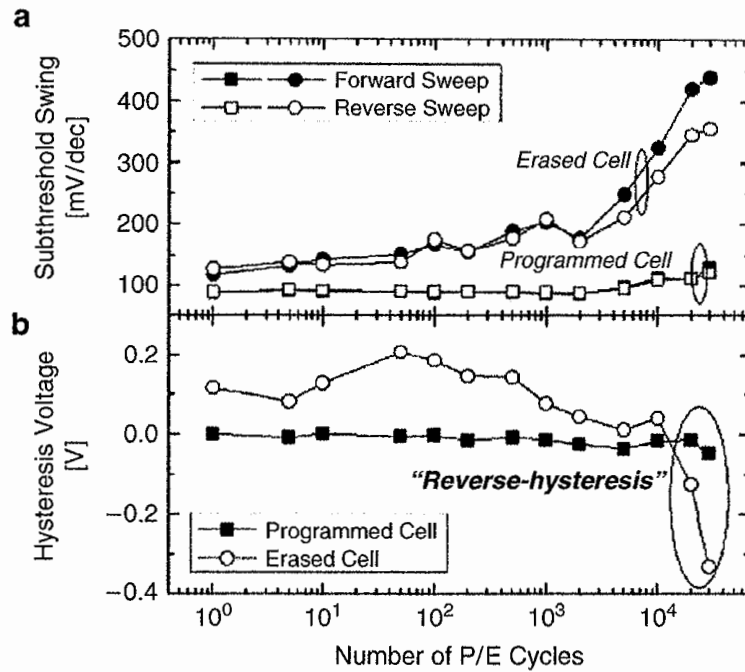


Fig. 4.24. The program/erase cycle evolution of the (a) subthreshold swing (SSW) and the (b) hysteresis characteristics of a measured NAND-type memory cell [33]

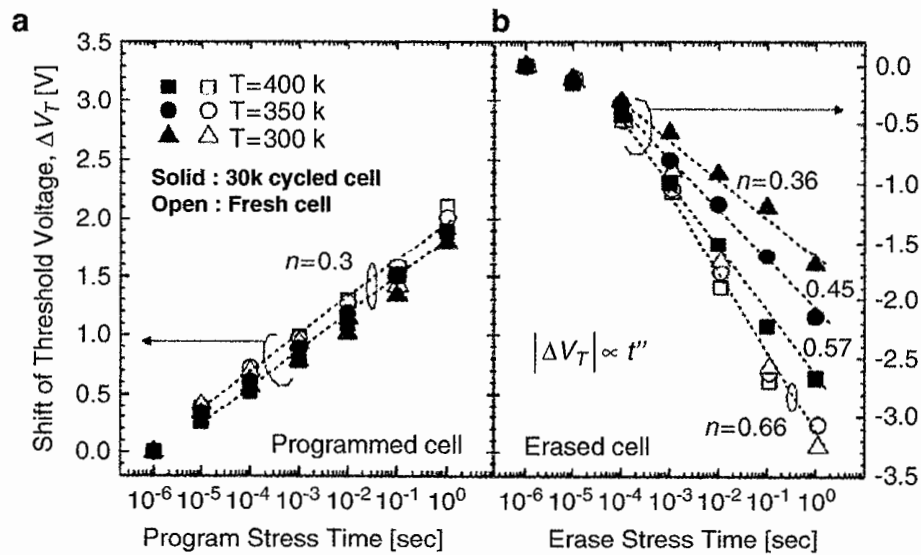


Fig. 4.25. The program/erase cycle Fowler-Nordheim stress time dependence of threshold voltage shift (ΔV_T) in the programmed cell (a) and in the erased cell (b) at various temperatures [33]

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 V_{th} 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-SiO₂ 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.

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 120mm² 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 μm^2 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.

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

- 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.

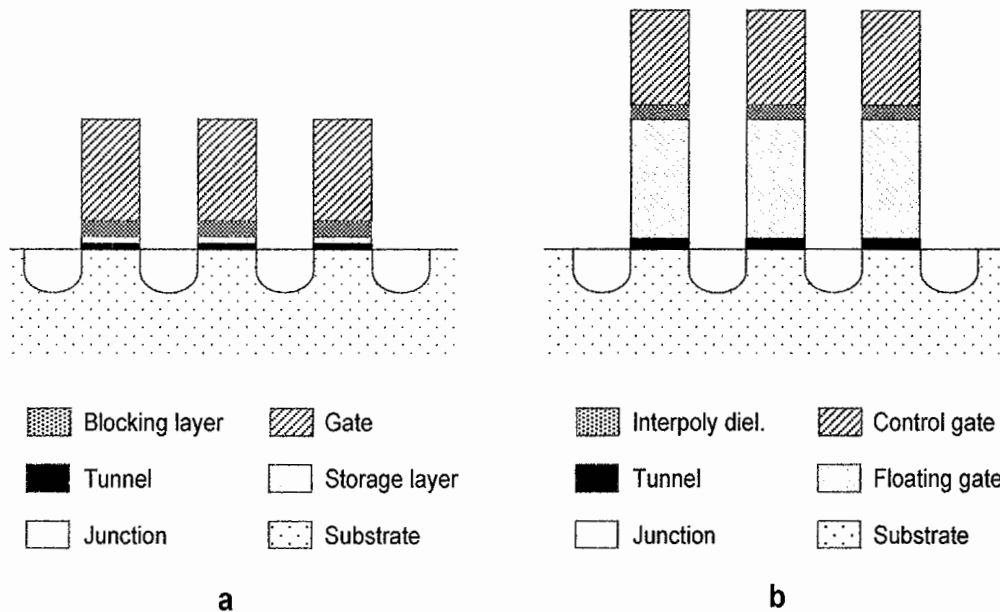


Fig. 5.1. Comparison between CT cell stack (a) and conventional FG memory (b)

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.

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

$$E_{tun} = \frac{V_G - V_{FB}}{EOT} \quad (5.1)$$

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)

is lower than 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 means of tunneling currents obtained applying a different bias to gate and substrate.

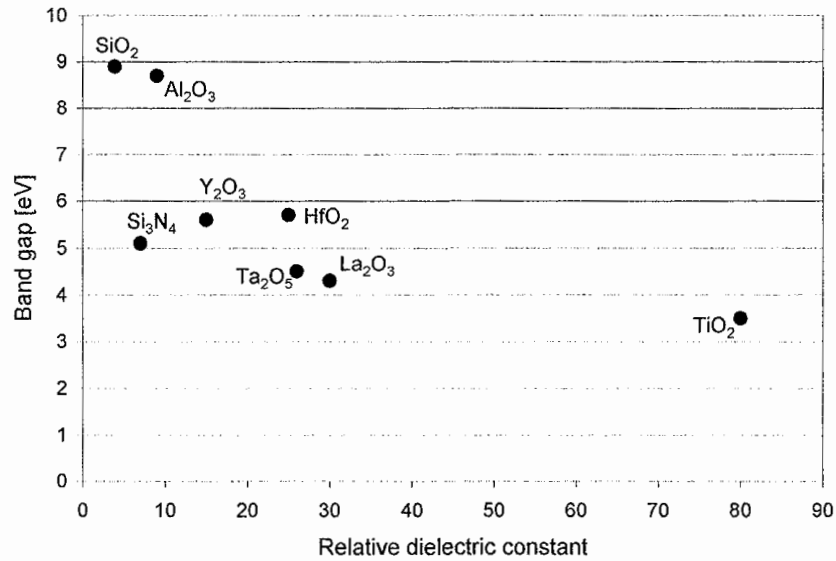


Fig. 5.2. Band gap versus relative dielectric constant for some dielectric materials

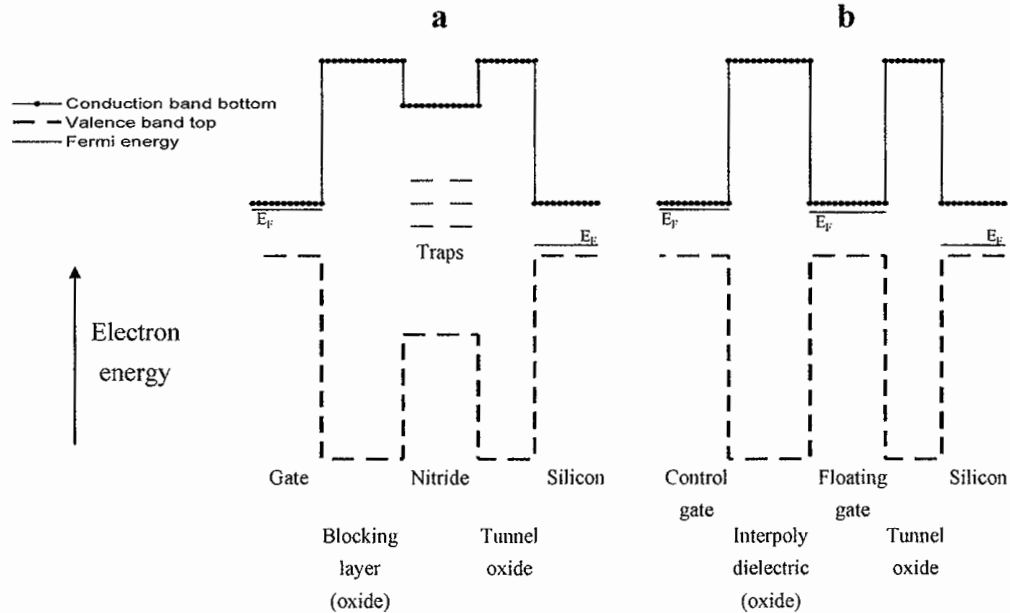


Fig. 5.3. Band diagram comparison between SONOS (a) and floating gate (b)

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:

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

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.

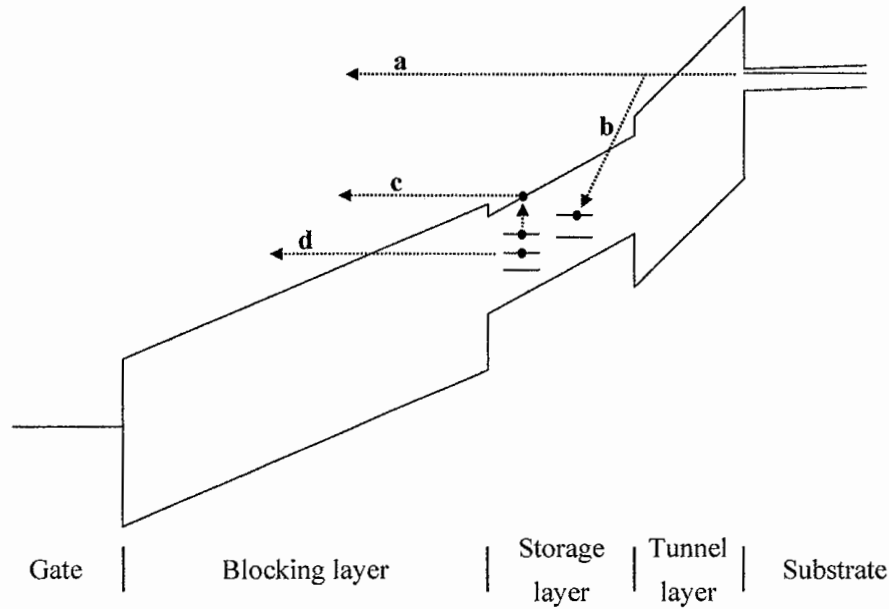


Fig. 5.4. Band structure of a typical stack during program with most important mechanisms: (a) tunneling electrons without interaction with storage layer; (b) trapped electrons; (c) Poole-Frenkel detrapping; (d) tunneling detrapping

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.

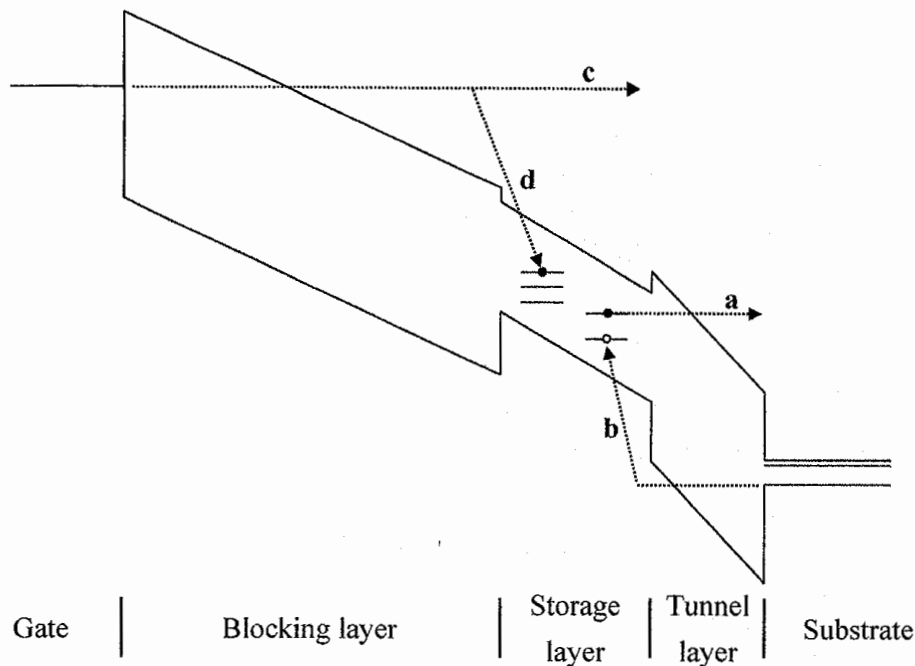


Fig. 5.5. Band structure of a typical stack during erase with most important mechanisms: (a) electrons detrapped from storage layer; (b) hole injection (minor for standard tunnel oxide); (c) back-tunneling current; (d) electrons trapped in storage layer

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 Band
Tunnel layer	silicon diOxide	silicon diOxide	silicon diOxide	Engineered Oxide
Substrate	Silicon	Silicon	Silicon	Silicon

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 than 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.

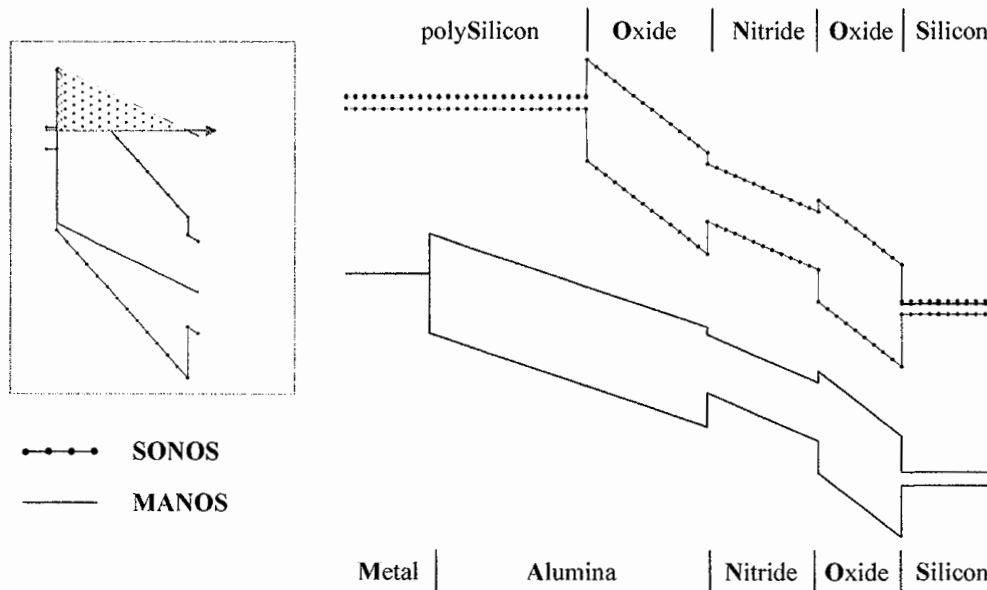


Fig. 5.6. Band structure comparison between SONOS and MANOS stacks with same EOT. In the inset on the left: gate-blocking region after alignment of Fermi level of electrons in gate and gate-blocking interface. A clear increase of barrier preventing back-tunneling injection is visible for MANOS.

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).

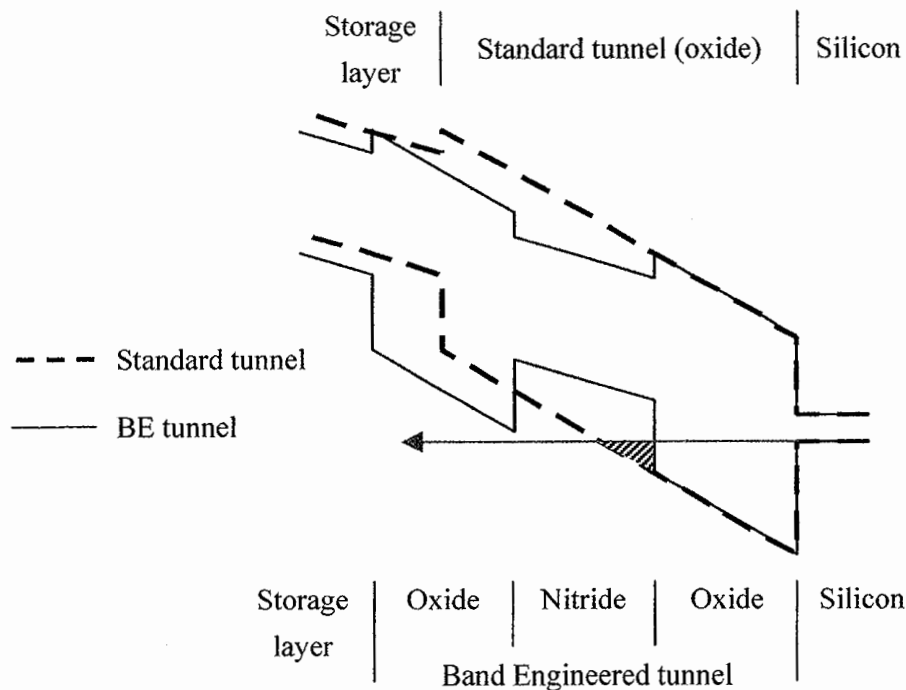


Fig. 5.7. Band structure comparison between two stacks with same EOT and different tunnel during erase. Smaller barrier for hole injection is pointed out in case of BE tunnel

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.

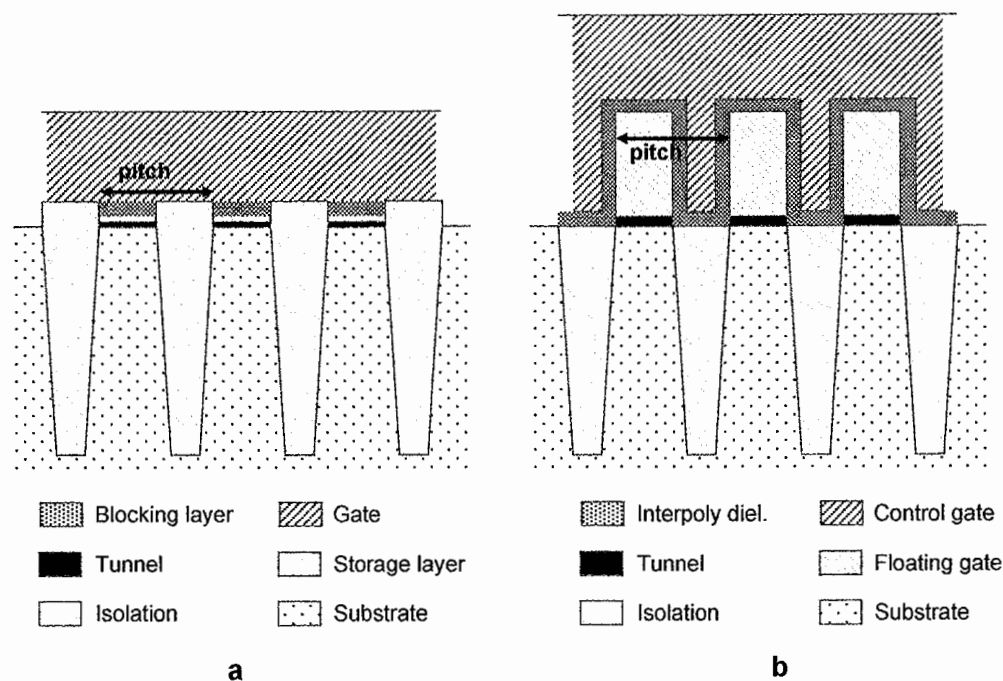


Fig. 5.8. Bitline cross-section of charge trap (a) and floating gate (b) NAND

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.

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 than 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 than 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

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.

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.

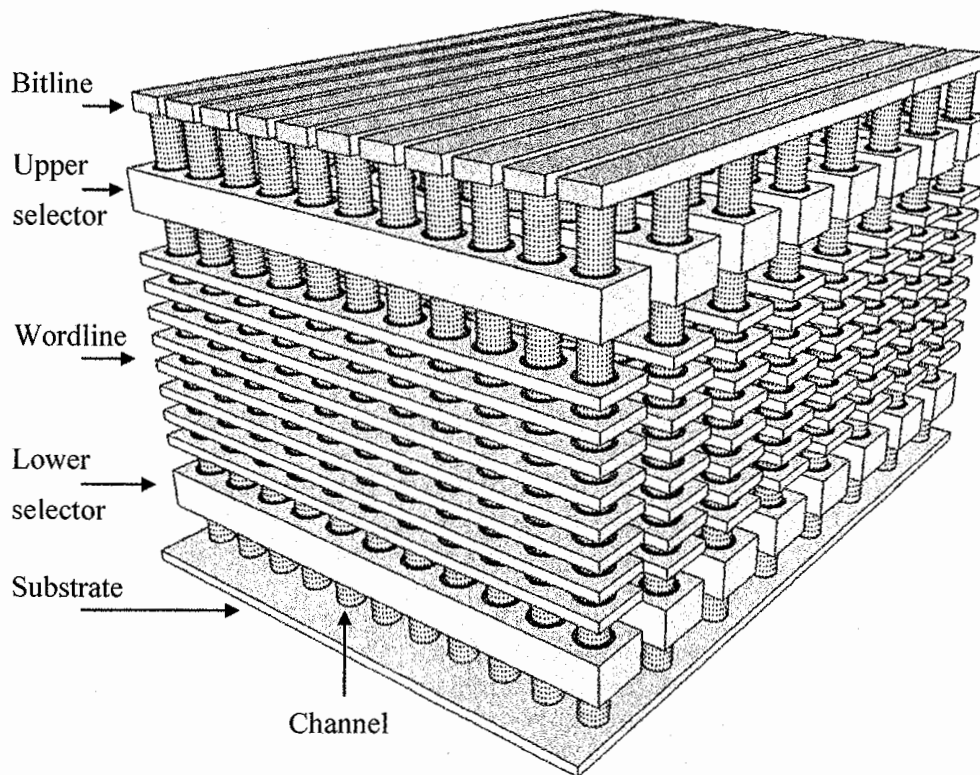


Fig. 5.9. Cross-section of a general 3D array with vertical channel

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.

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/erase speed and retention of TiN-gate MANOS (Si-oxide-SiN_x-Al₂O₃-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 SiO₂/SiN/Al₂O₃ 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 Al₂O₃", 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 Al₂O₃ 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.

-
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.

6 Control logic

A. Marelli¹, R. Micheloni² and R. Ravasio³

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.

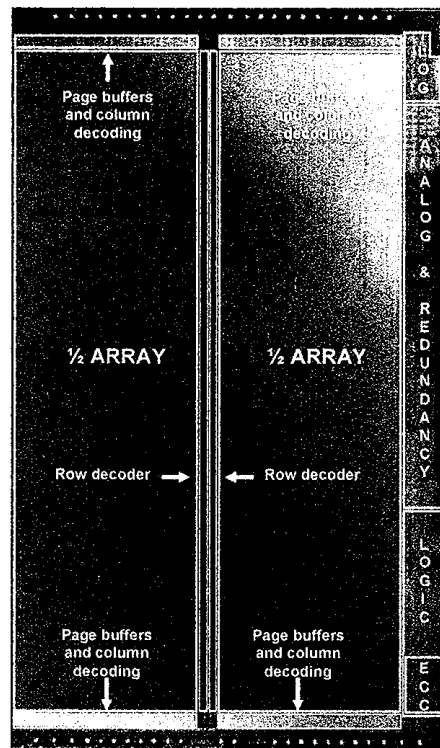


Fig. 6.1. Picture of a NAND 4 Gb MLC device, where the logic part is highlighted [1]
©IEEE

¹ Integrated Device Technology, alessiamarelli@gmail.com

² Integrated Device Technology, rino.micheloni@ieee.org

³ Pegasus MicroDesign, roberto.ravasio@ieee.org

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.

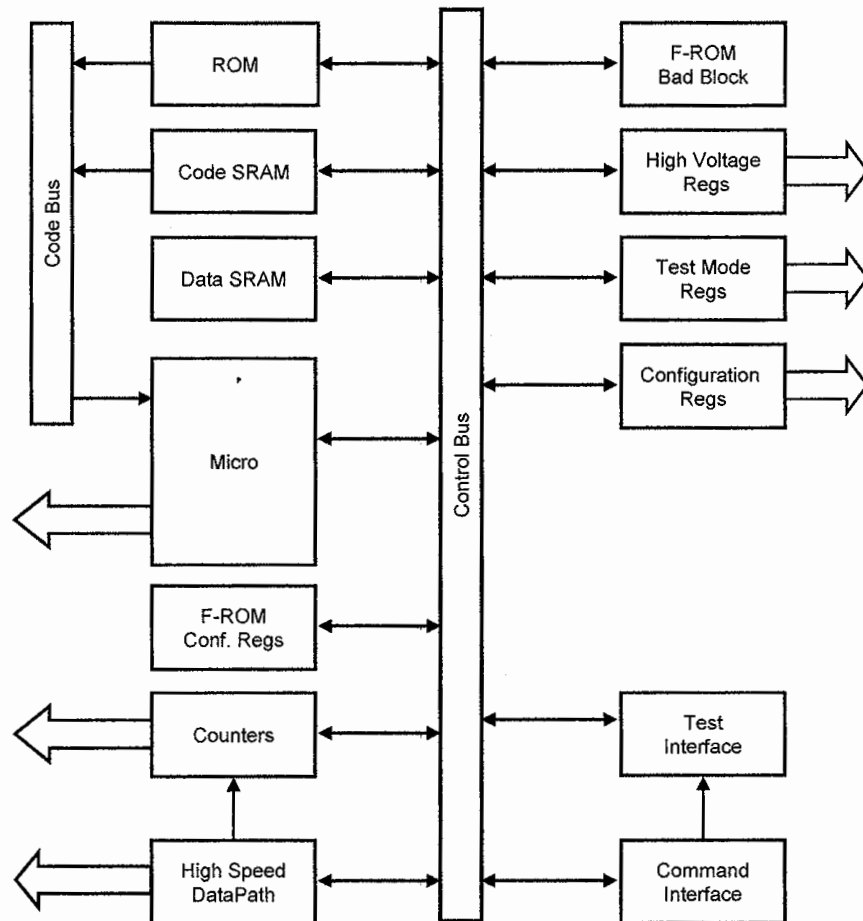


Fig. 6.2. Logic view of a NAND device

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.

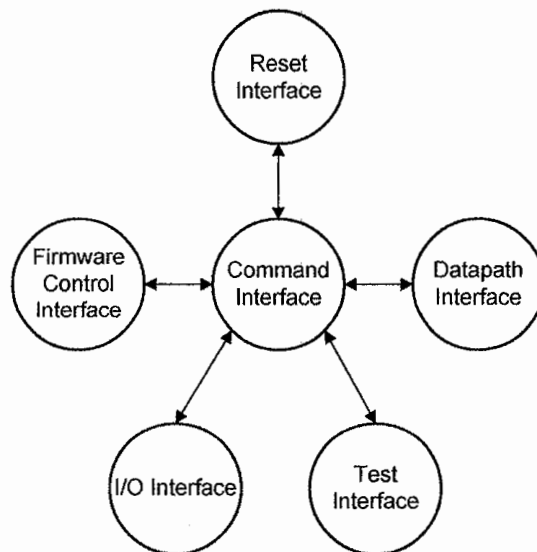


Fig. 6.3. Command interface and its interaction blocks

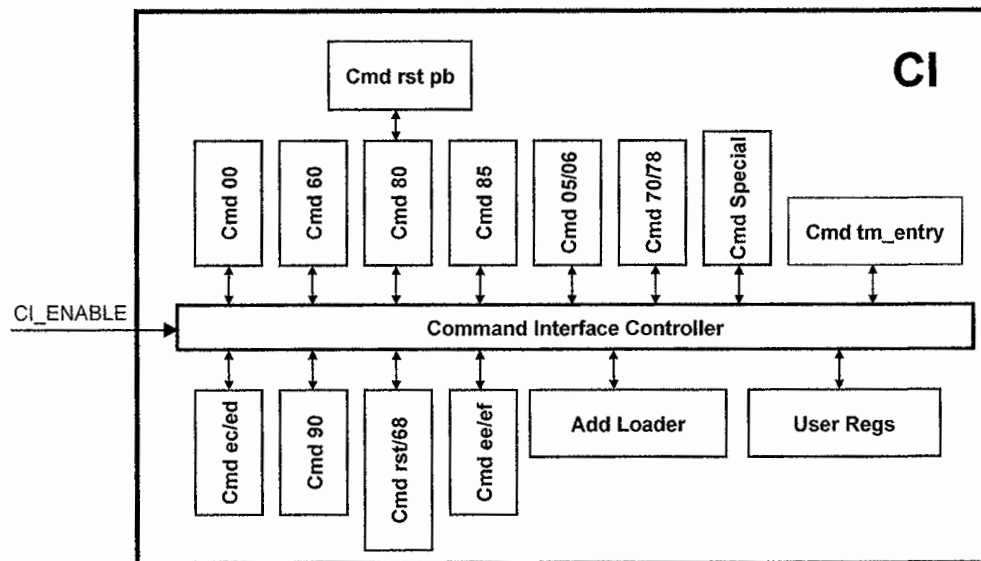


Fig. 6.4. CI basic representation

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.

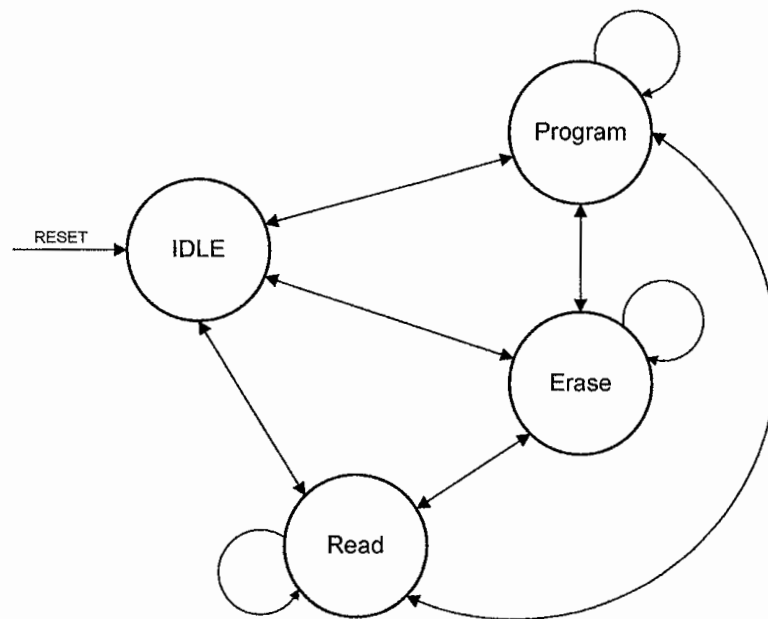


Fig. 6.5. Representation of the general FSM

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`.

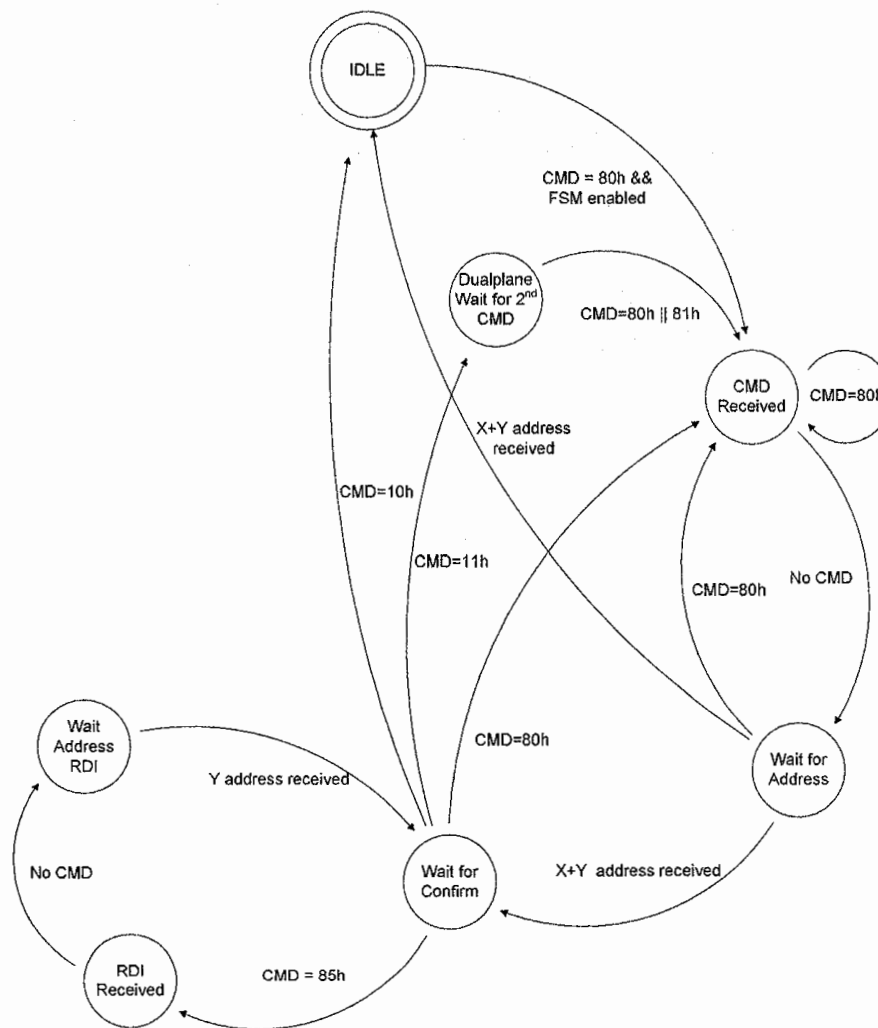


Fig. 6.6. subFSM for a program sequence

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.

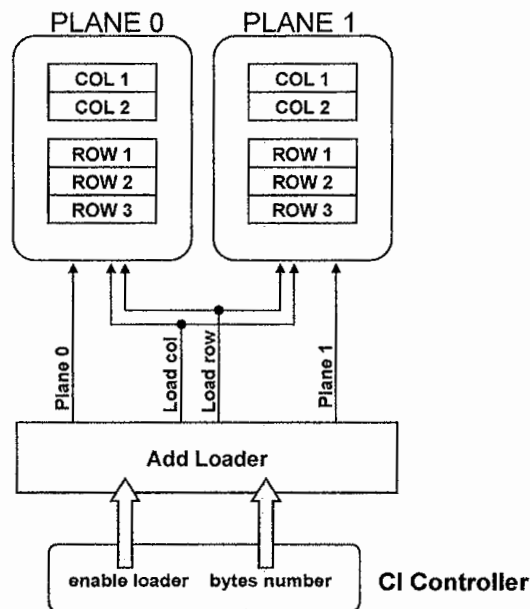


Fig. 6.7. Basic Address Loader scheme

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.

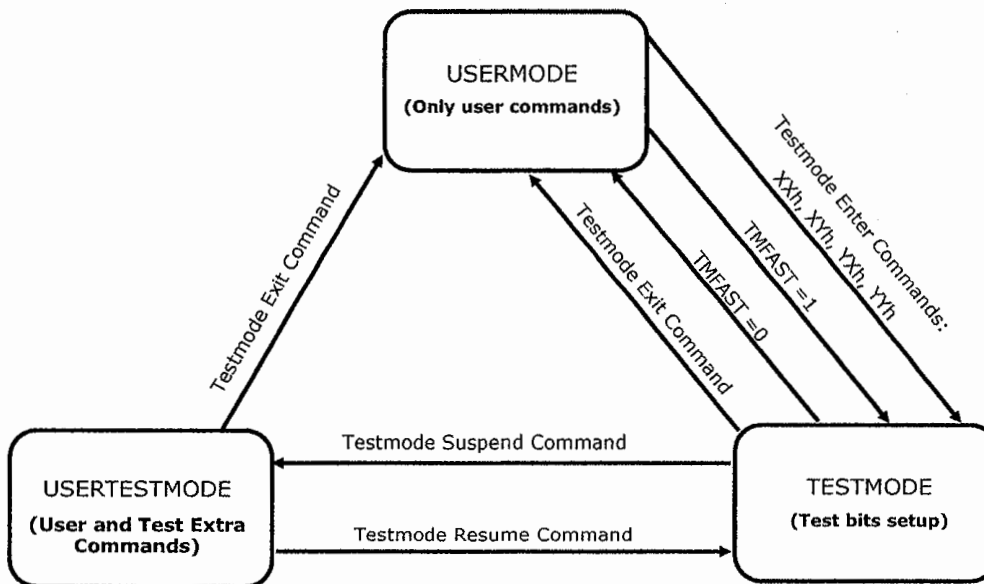


Fig. 6.8. Flow diagram used to change operational modes among usermode, usertestmode and testmode

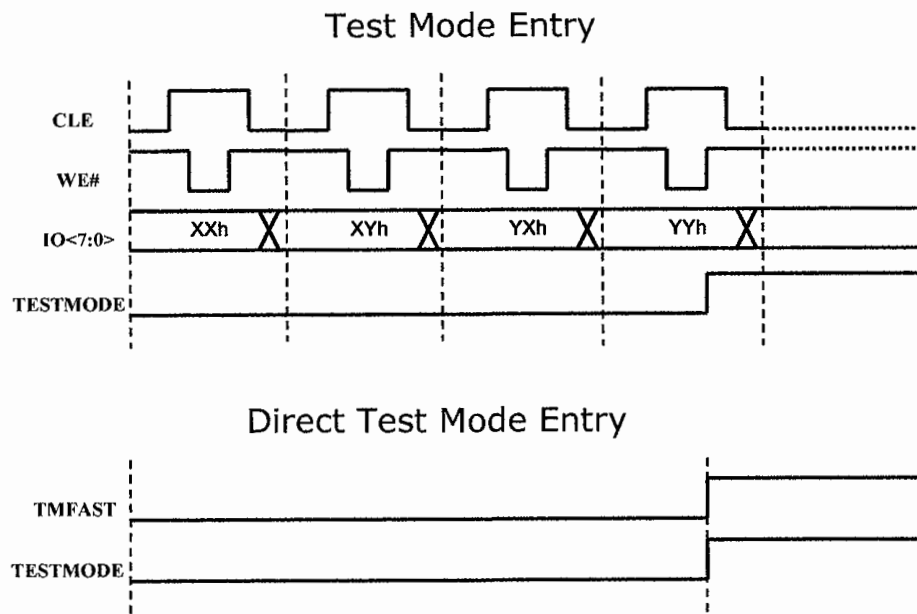


Fig. 6.9. Test Mode entry and direct Test Mode entry through the TMFAST pad

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.

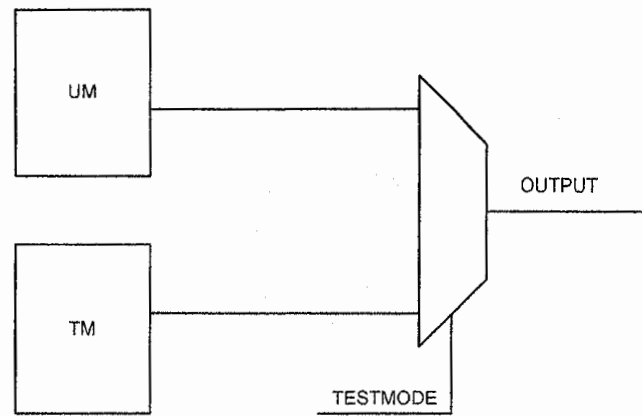


Fig. 6.10. Testmode registers

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:

- $TI \times 8$ where eight DQs are used as input/output pins
- $TI \times 4$ where four DQs are used as input/output pins
- $TI \times 2$ 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 $TI \times 8$ 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 $TI \times 4$ and $TI \times 2$ are shown in Table 6.2.

Table 6.1. Pins used in $TI \times 8$ machine

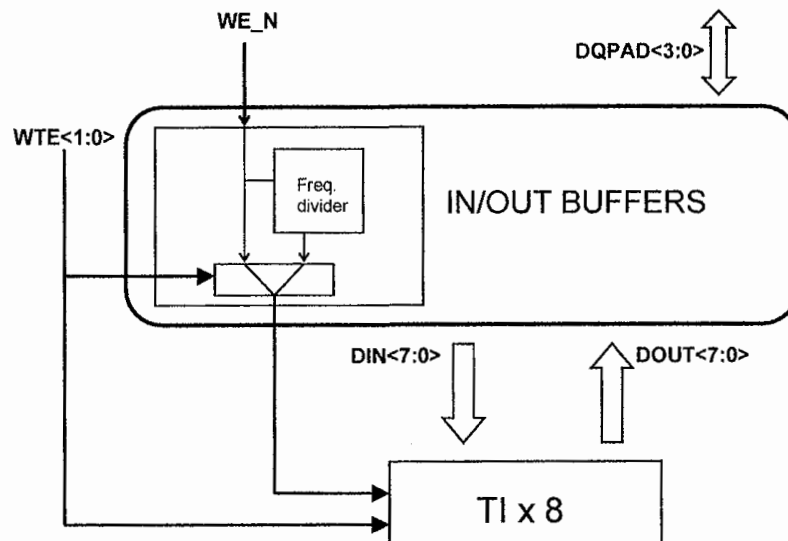
Pin Name	Width	Usage	Remarks
CE_N	1	Per Chip	Separate for each chip
WE_N	1	Shared	Works as a clock
RE_N	1	Shared	Enables the output
ALE	1	Shared	Indicates a new address
CLE	1	Shared	Indicates a new command
DQ	8	Shared	Transfers command/data/address

Table 6.2. Pins used in $TI \times 4$ and $TI \times 2$ 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 $TI \times 4/ TI \times 2$ is less than in $TI \times 8$. 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 $TI \times 8$. The cycles will pass the data bytes from the most significant bit to the least significant bit.

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

**Fig. 6.11.** $TI \times 4/2$ data reconstruction for $TI \times 8$ format

6.4 Datapath

Traditionally, NAND memories have an asynchronous interface and it is very difficult to have frequency higher than 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

through the chip. One of the most adopted solutions to overcome 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.

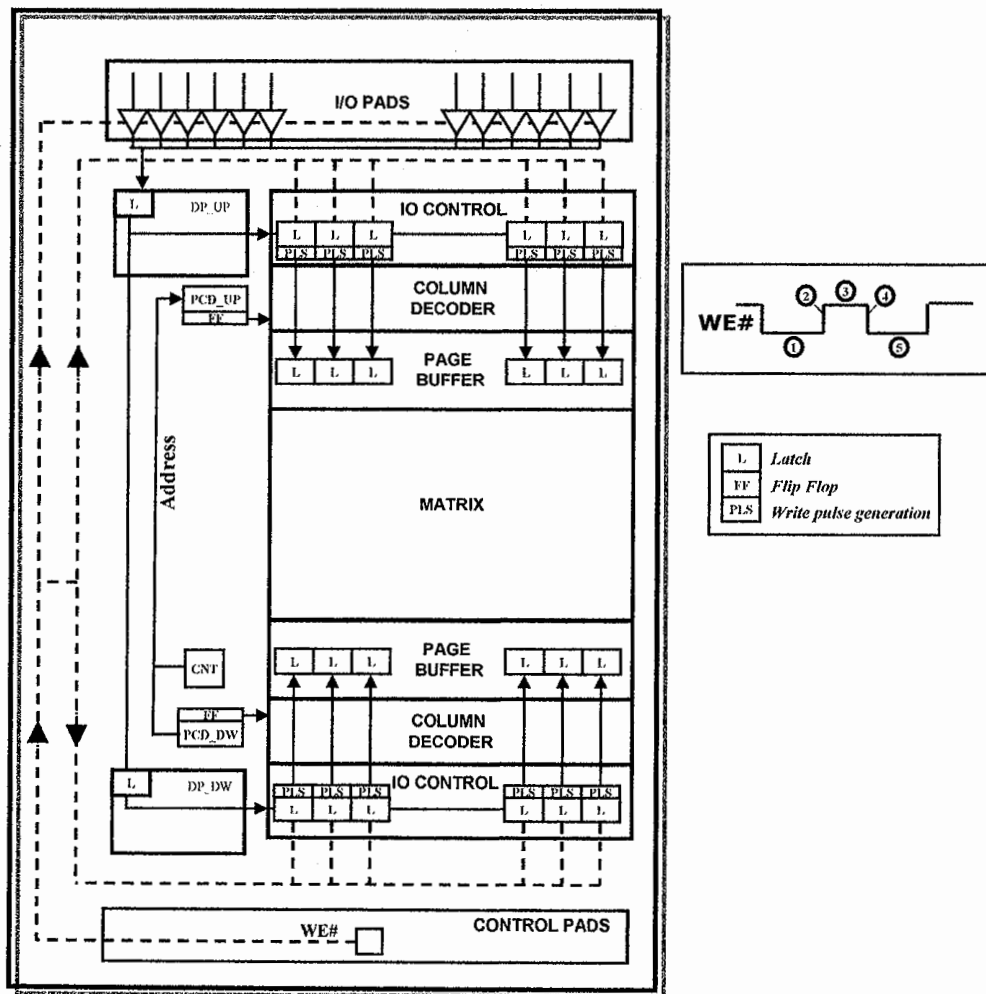


Fig. 6.12. Data input path

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).

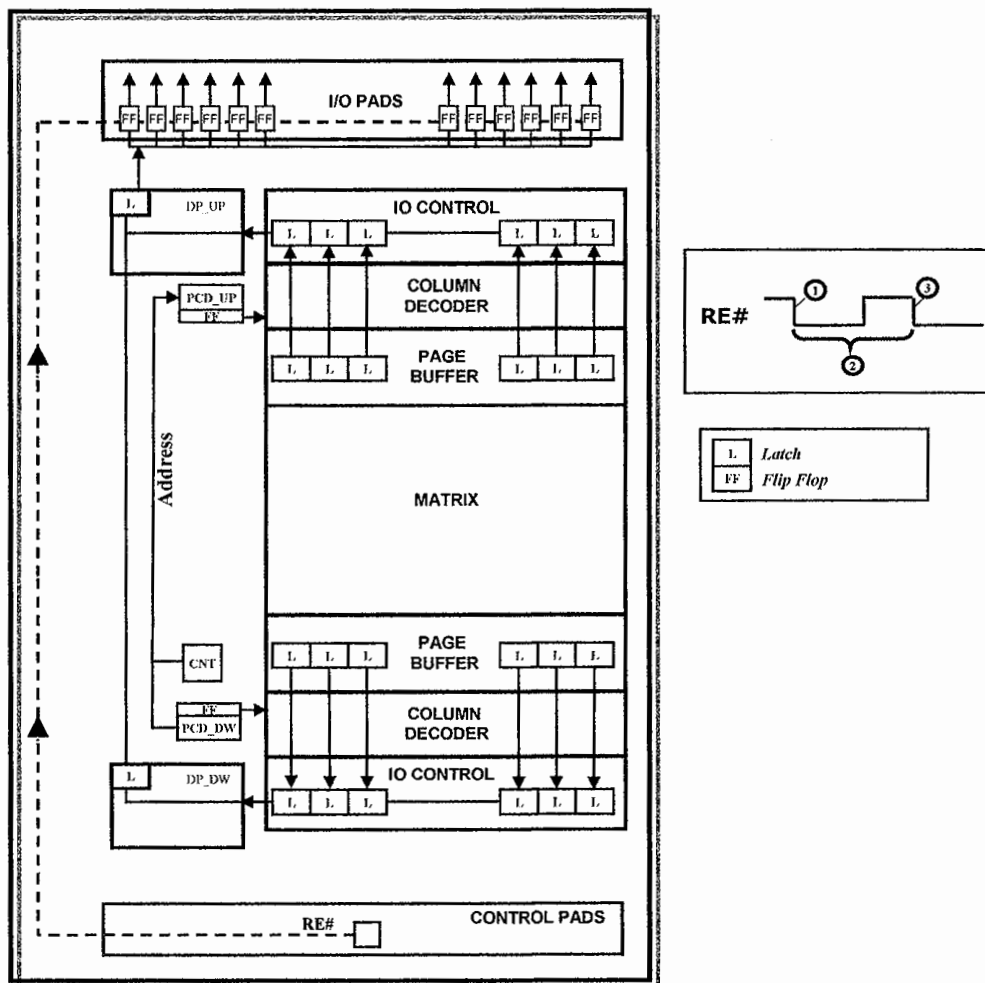


Fig. 6.13. Data output path

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.

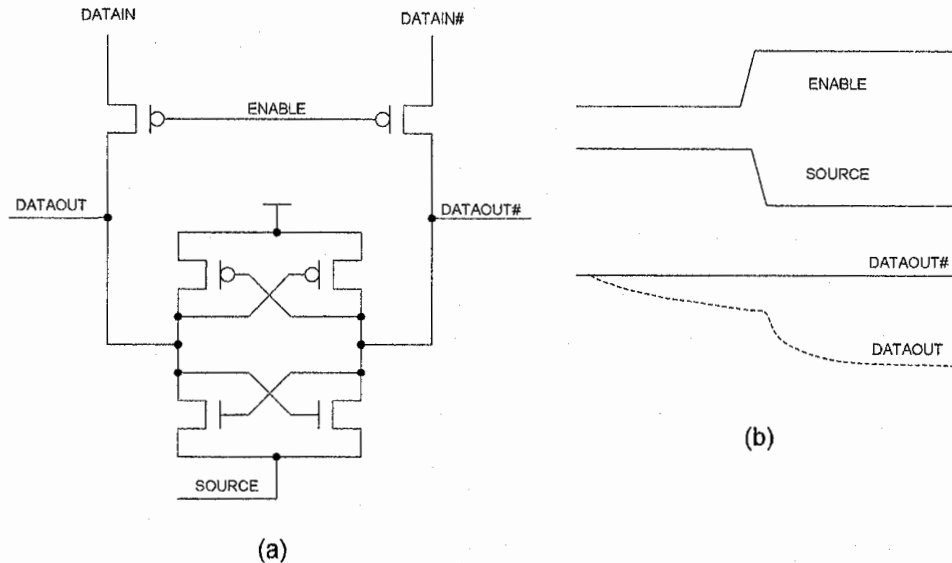


Fig. 6.14. Differential data transmission

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.

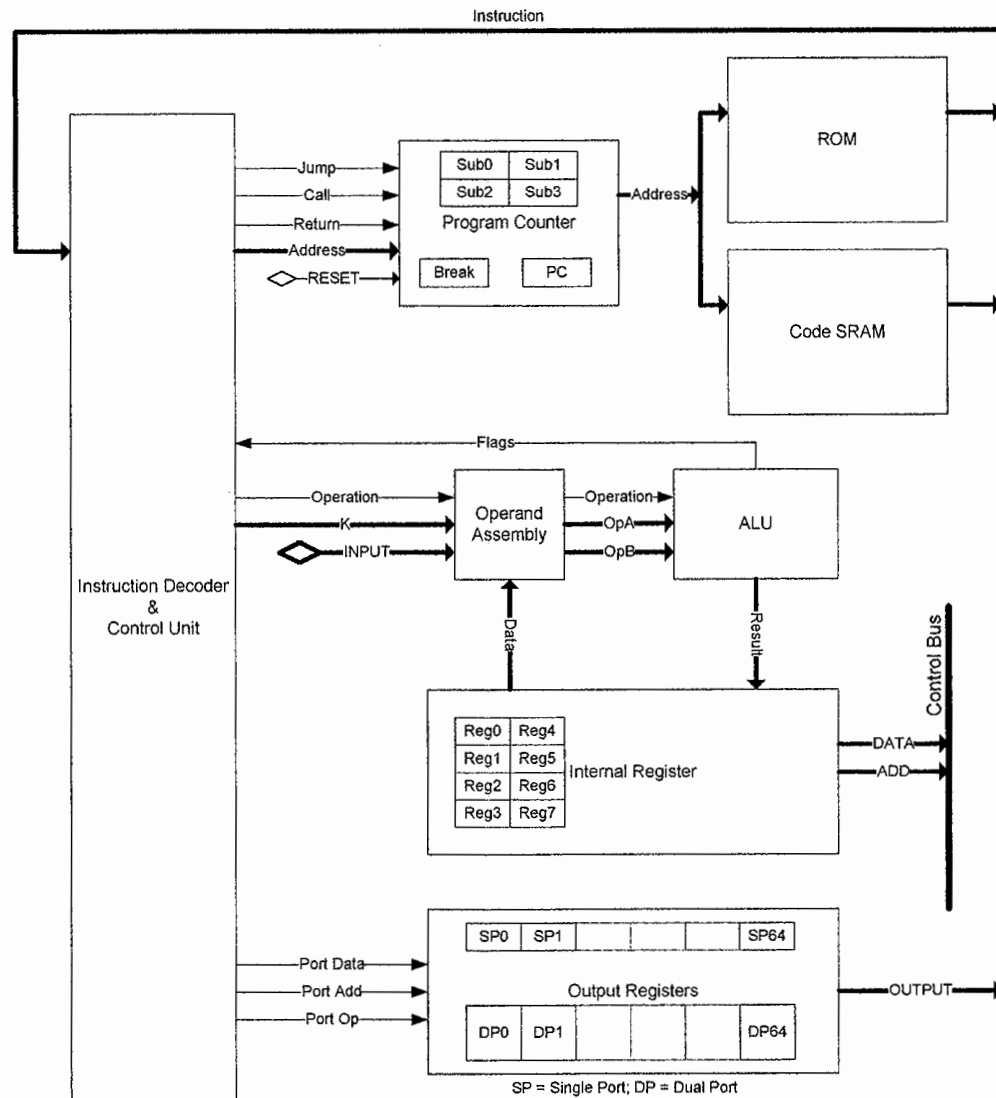


Fig. 6.15. Microcontroller structure with ROM and RAM memories

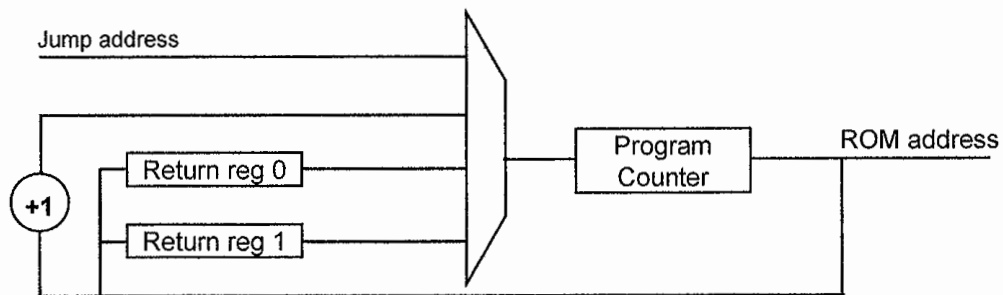


Fig. 6.16. Program counter structure

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).

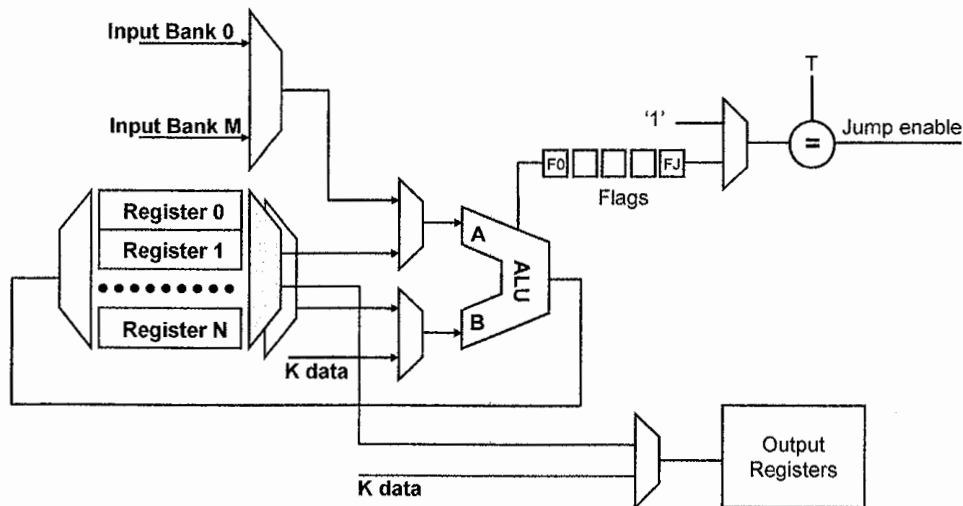


Fig. 6.17. Data manipulation inside microcontroller

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).

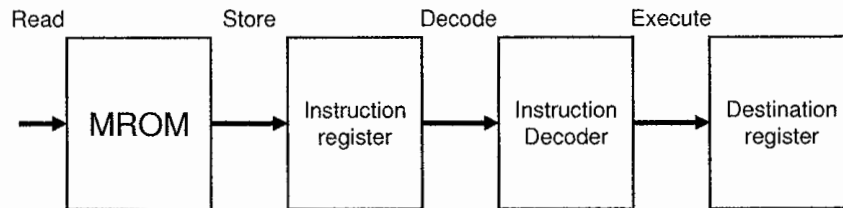


Fig. 6.18. Microcontroller timings

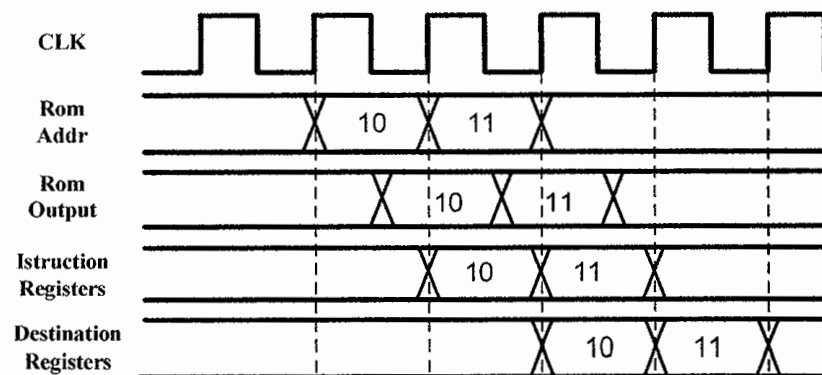


Fig. 6.19. Pipeline waveforms

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.

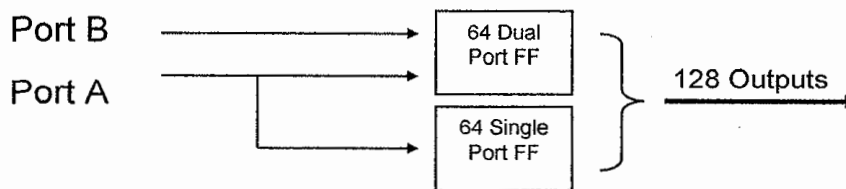


Fig. 6.20 Output registers

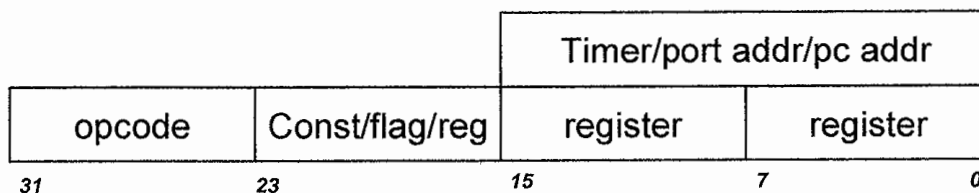
Table 6.3. Logical operations

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

Table 6.4. Arithmetical operations

Name	Action	Flag 1	Flag 0
SUM	A + B $\geq$ D A + K $\geq$ D	Z	C
SUB	A - B $\geq$ D A - K $\geq$ D	Z	C
INC	A + 1 $\geq$ D	Z	C
DEC	A - 1 $\geq$ D	Z	C
CMP	A - B A - K	Z	
LD	A $\geq$ D K $\geq$ D	Z	

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

I N S T R U C T I O N																								NAME				DESCRIPTION			
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
A Data		A Add 4-0		OpA		B Data		B Add		OpB		1	1	1	PORTA_PORTB									1	Port_A	Port_B					
A Data		A Add 4-0		OpA		J	T	A5		AI		1	1	0	0	0	IFPORTA									2	if F(J)==T then Port_A				
A Data		A Add 4-0		OpA		J	T	Offset 6-0				0	0	1	0	PORTA_IFGOTO									3	Port_A	if F(J)==T then goto (\$+ofs)				
A Data		A Add 4-0		OpA		8	7	Offset 6-0				1	0	1	1	PORTA_GOTO									4	Port_A	Goto (\$+ofs)				
A Data		A Add 4-0		OpA		8	7	Offset 6-0				0	0	1	1	PORTA_CALL									5	Port_A	Call (\$+ofs)				
A Data		A Add 4-0		OpA		J	T	A5		AI		1	0	0	0	PORTA_RET									6	Port_A	if F(J)==T then return				
		Add 10-7		1	0	AI	J	T	Add 6-0				1	0	0	1	IFGOTO									7	if F(J)==T then goto (ABS)				
		Add 10-7		1	1	AI	J	T	Add 6-0				1	0	0	1	IFCALL									8	if F(J)==T then call (ABS)				
D						0	B Data	B Add		OpB		1	0	0	DOP_PORTB									9	D = Op (D) Port_B						
D						AI	J	T	Offset 6-0				1	0	1	0	DOP_IFGOTO									10	D = Op (D); If F(J)==T then goto ()				
D		Pos		0	1	1	B Data	B Add		OpB		1	0	1	1	DOPBIT_PORTB									11	D = Op (D,Pos) Port_B					
D		Pos		0	0	AI	J	T	Offset 6-0				1	0	0	1	DOPBIT_IFGOTO									12	D = Op (D,Pos) If F(J)==T then goto ()				
Pos		A		1	0	0	B Data	B Add		OpB		1	0	1	0	TEST_PORTB									13	Test (A,Pos) Port_B					
Pos		A		0	1	AI	J	T	Offset 6-0				1	0	0	1	TEST_IFGOTO									14	Test (A,Pos) If F(J)==T then goto ()				
D		A				0	B Data	B Add		OpB		1	1	0	DOPA_PORTB									15	D = Op (D,A) Port_B						
D		A				AI	J	T	Offset 6-0				0	0	0	1	DOPA_IFGOTO									16	D = Op (D,A) If F(J)==T then goto ()				
D		A				AI	J	T					0	1	0	0	IFDOPA									17	If F(J)==T then D = Op(D,A)				
D				AI		1	J	T	K				1	0	0	0	IFDOPK									18	If F(J)==T then D = Op(D,K)				
D		A		AI		AI	J	T	B				0	0	0	0	IFDOPAB									19	If F(J)==T then D = Op(A,B)				
D		A		AI		1	J	T	K				1	1	0	IFDOPAK									20	If F(J)==T then D = Op(A,K)					
D				AI		1	1	0	J		T		K				1	0	1	IFFORDK									21	If F(J)==T then for (D=K; D=0; D--) {if (intr=1) then Bk}	
D		A		1	0	1	B Data		B Add		OpB		1	0	1	1	IFFORDA_PORTB									22	If F(J)==T then for (D=A; D=0; D--) {Port_B; if (intr=1) then Bk}				
Ext. Addr Set				0	0	0	B Data		B Add		OpB		1	0	1	ADDSET_PORTB									23						
D		A		0	0	1	B Data		B Add		OpB		1	0	1	1	EXTREG_PORTB									24					
K				0	1	0	B Data		B Add		OpB		1	0	1	1	EXTREGK_PORTB									25					

Fig. 6.22. Instruction set

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).

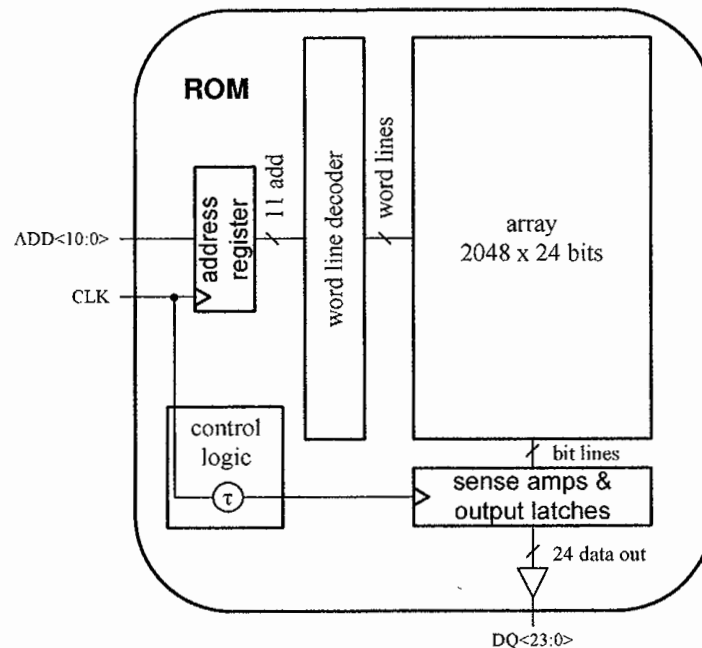


Fig. 6.23. ROM architecture

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.

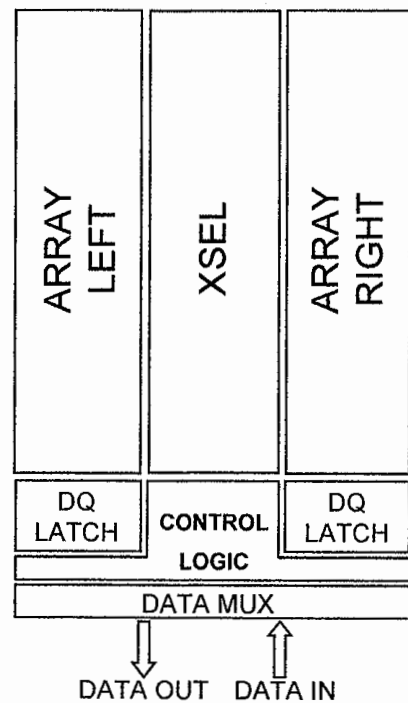


Fig. 6.24. Data arrangement inside the array

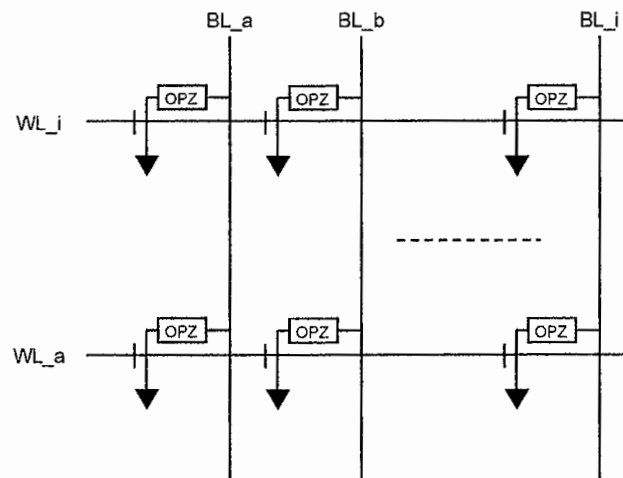


Fig. 6.25. ROM array

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.

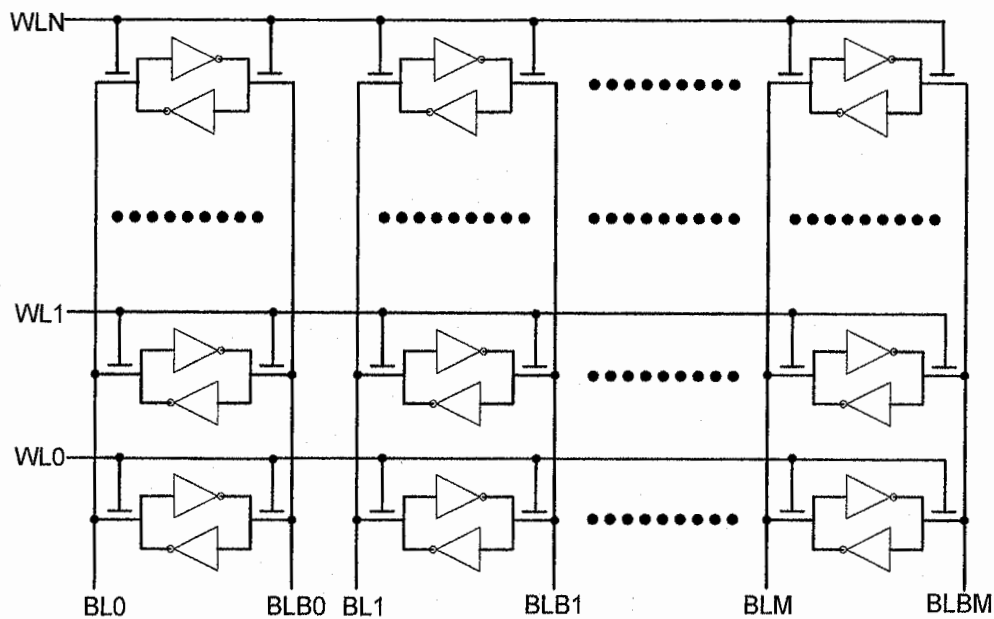


Fig. 6.26. RAM array

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.

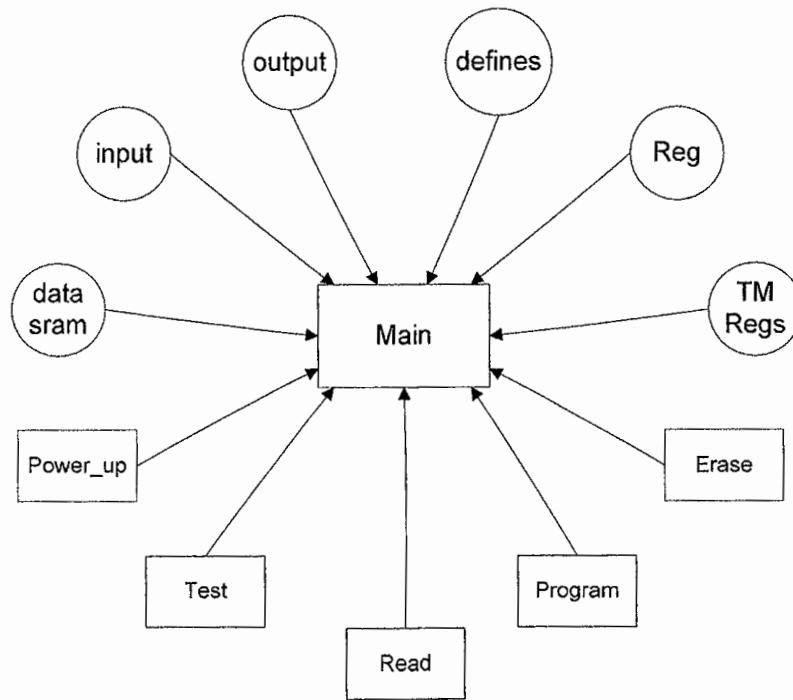


Fig. 6.27. Code structure

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_gslpmpptovblcpgm` and `hv_gslpmpptovblcrd`. 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.

```

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

/*-----

#define hv_vblctognd   "03h 0"
#define hv_vblcpepgmtognd "03h 1"
#define hv_gslpmpptovblcpgm "03h 2"
#define hv_gslpmpptovblcrd "03h 3"

/*-----

#define hv_vunseltognd   "04h 0"
#define hv_passpmpptovunselrd "04h 1"
#define hv_vselctognd    "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.

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_biasseselect(0));
/* dsg and ssg to gnd
set(hv_gssltognd); set(hv_gdsltognd);
/* sel and unsel switches disabled
cba($cr_selvprevsen); store(hv_vunseltoognd
1,hv_passpmpptovunselrld 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

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.

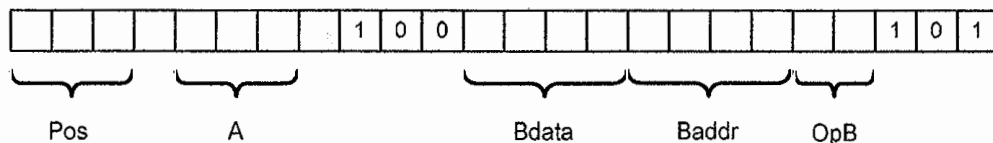


Fig. 6.30. Opcode example

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

01111100111101100010100
111000000011100101111000
111100000011110001100110
001110100111111001000111
101001000111111011001100
000000000000000000000000
101000011010100101000001
101001111010111010000000
101100101011111011001000
101100011000111011001000
111000000011111001001000
111001000011100000000001
111000000011101010011110
000010000010001100110011

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.

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.

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

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.

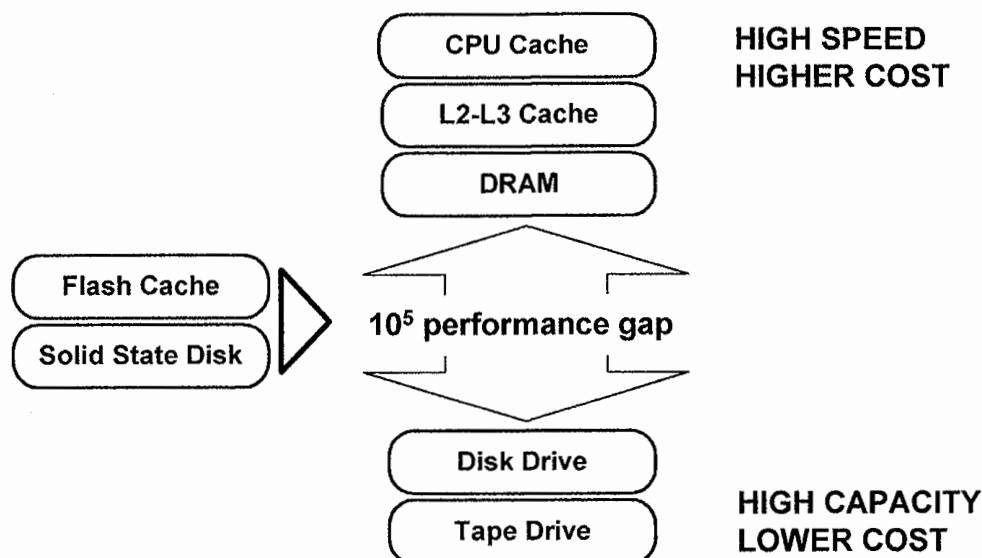


Fig. 7.1. Memory hierarchy in the PC platform and Flash opportunity

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 μ s 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.

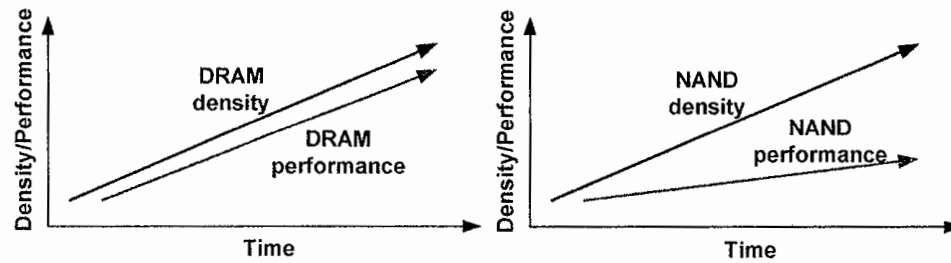


Fig. 7.2. NAND Performance trend versus DRAM

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.

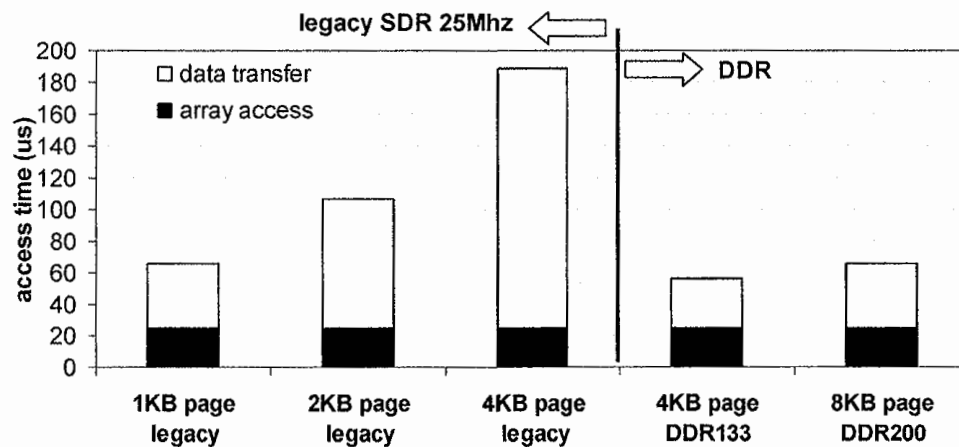


Fig. 7.3. Array time and data transfer breakdown in NAND: Legacy versus DDR

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.

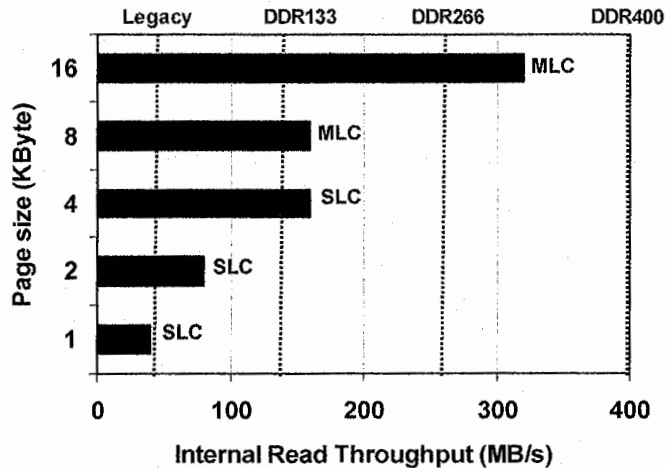


Fig. 7.4. Internal Read throughput versus page size

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.

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.

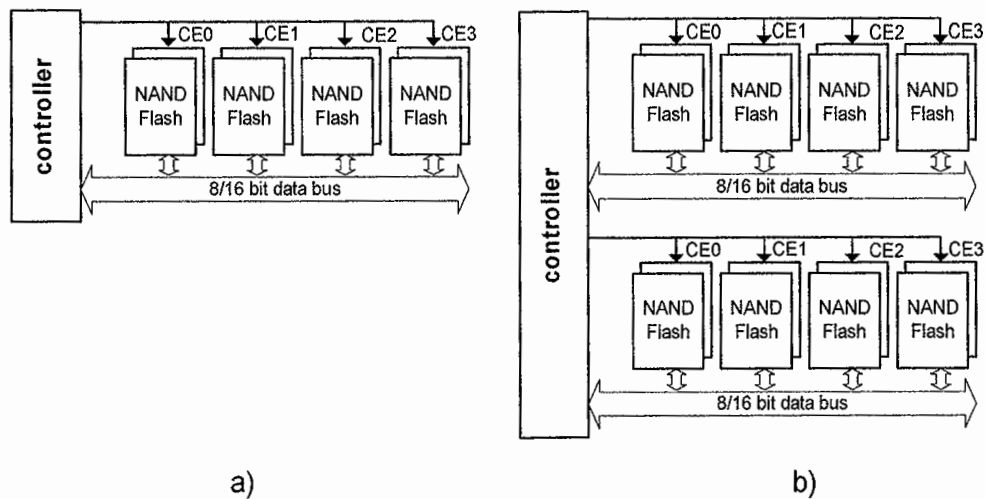


Fig. 7.5. SSD system enhancement: (a) increased number of dies per channel (b) increased number of channels

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.

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	I/O	ODT	Strobe	Prefetch	Package
		Max 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 LVTTTL 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.

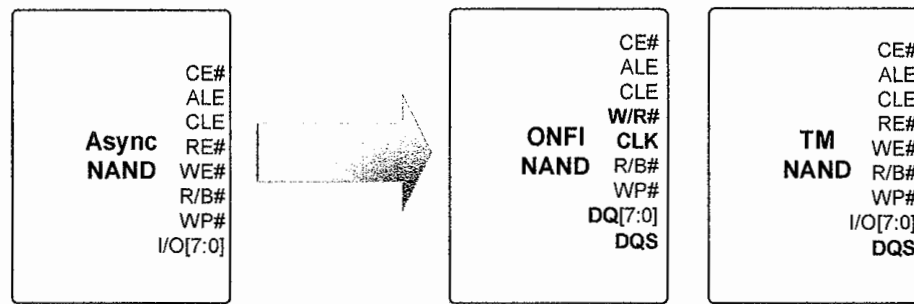


Fig. 7.6. Legacy NAND versus ONFI 2 and Toggle-Mode Synchronous NAND Interface

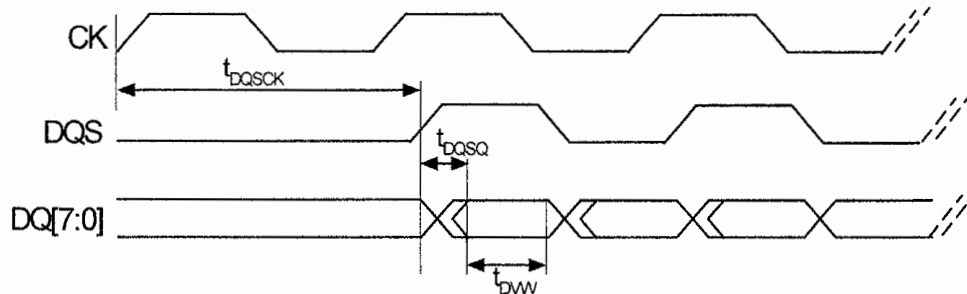


Fig. 7.7. DDR interface diagram

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.

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 as 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]:

$$P = 9 \cdot \eta \cdot f \cdot C \cdot V^2 \quad (7.1)$$

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).

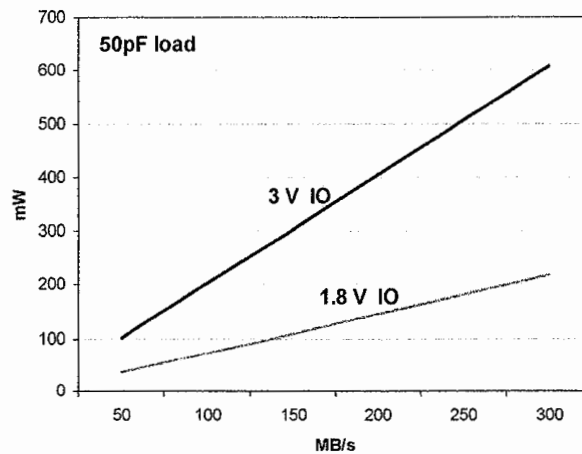


Fig. 7.8. I/O power as a function of channel throughput

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.

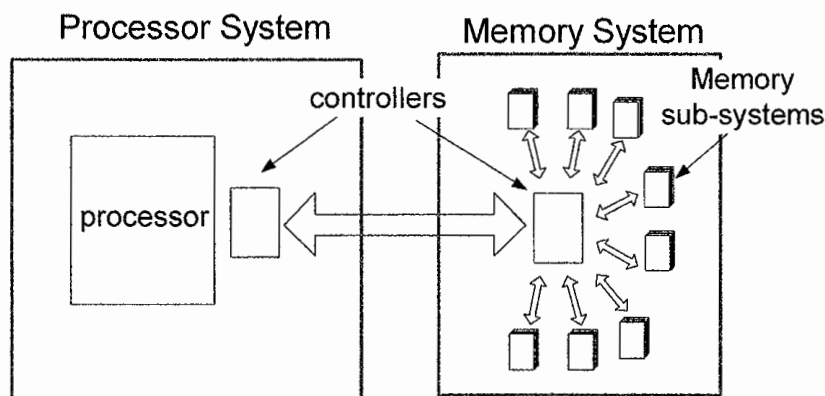


Fig. 7.9. Architectural change evolution

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

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.

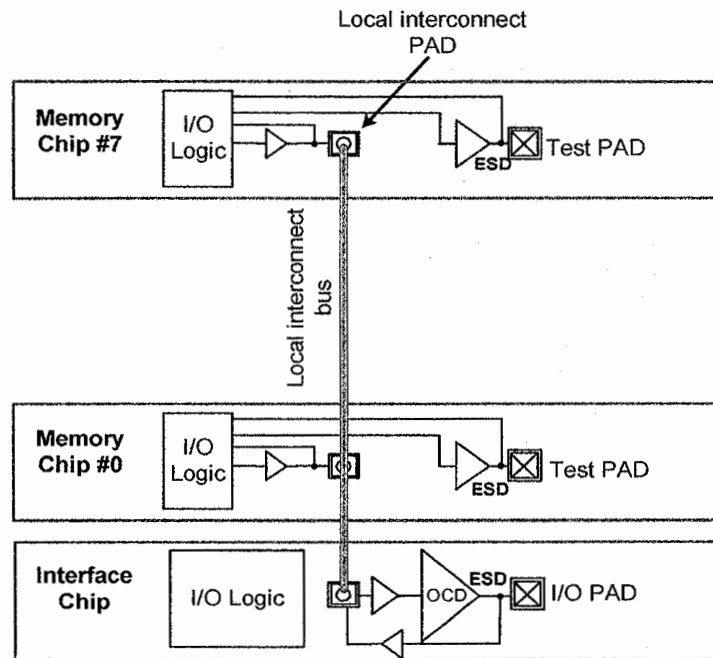


Fig. 7.10. Local interconnect bus architecture

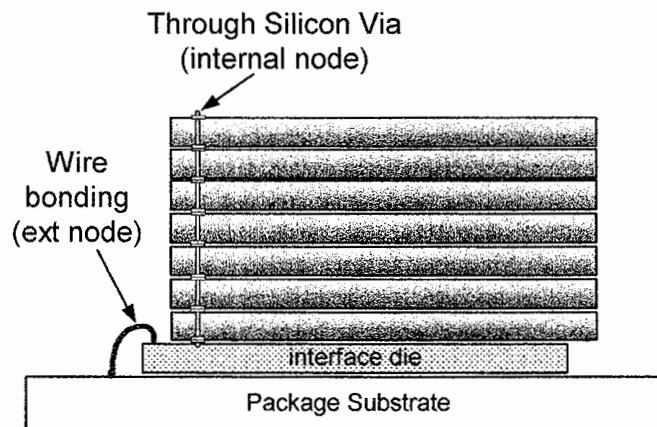


Fig. 7.11. Multiple stacking with local interconnect bus TSV Technology

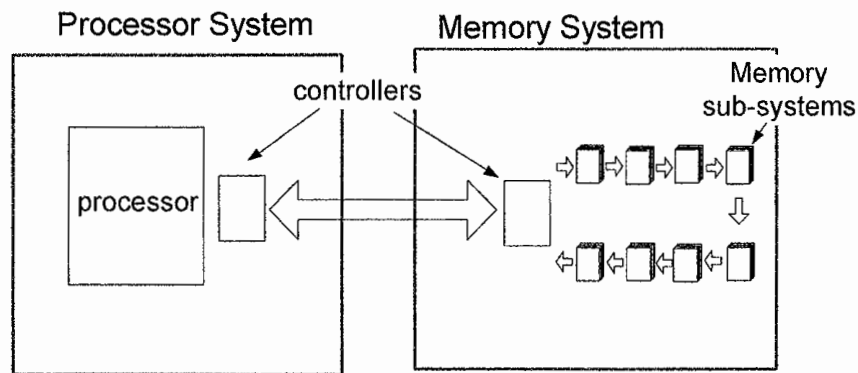


Fig. 7.12. Serial ring topology

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 circuitual 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

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.

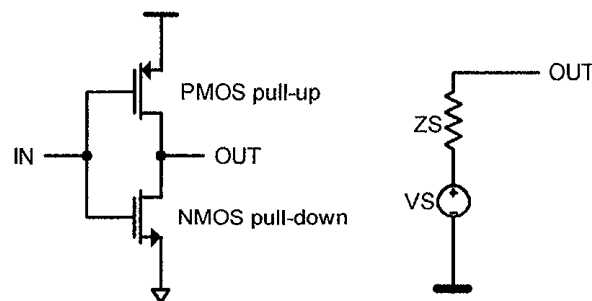


Fig. 7.13. Simple output buffer schematic

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 V_{DD} and the input of the buffer is switched from GND to V_{DD} . The operating point initially moves from one curve to the other as V_{GS} raises from 0 V to V_{DD} and finally moves along the curve shown with $V_{GS} = V_{DD}$. The load is discharged and the operating point moves along the N-MOS characteristic curve until $V_{DS} = 0$ 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 $V_{DS} < V_{DD} - V_{TH}$). It is important to recall that devices operate most of the time in the linear zone (where $V_{OUT} = V_{DD}/2$) 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.

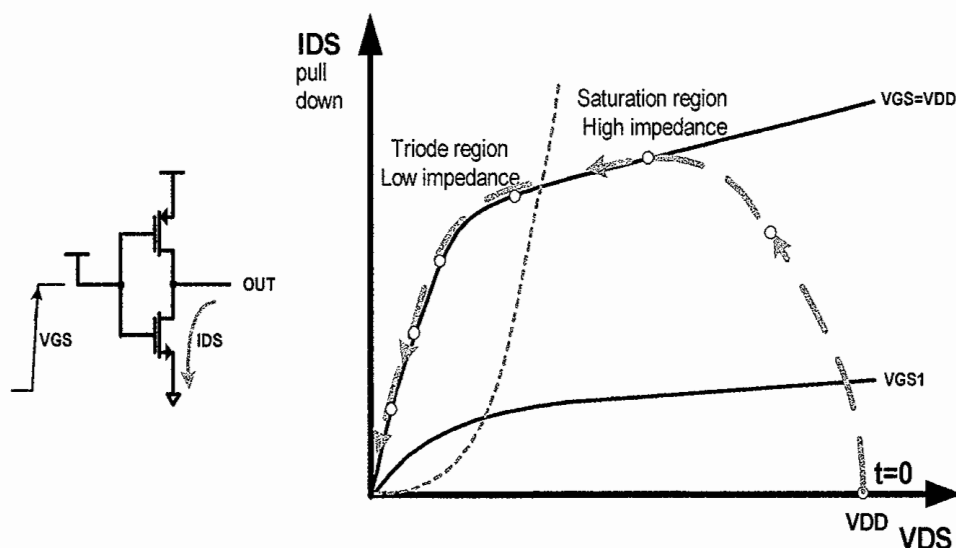


Fig. 7.14. Output buffer operation

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].

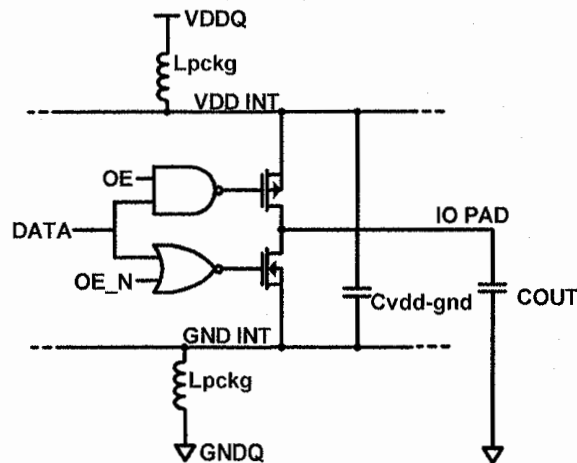


Fig. 7.15. RLC Parasitics in output buffer simple model

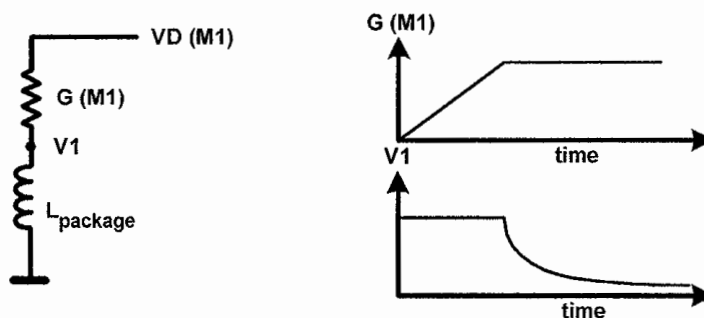


Fig. 7.16. Simple model to evaluate voltage noise

The noise generated on node V1 can be calculated as:

$$V_N^{\max} = VD \cdot \frac{1}{1 + \frac{t_r}{L \cdot G}} \quad (7.2)$$

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 $\frac{1}{2} VD$.

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].

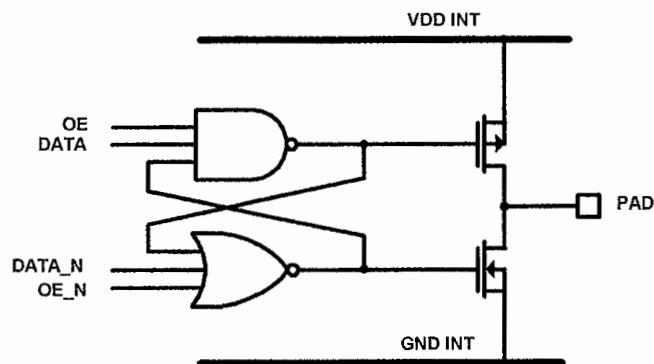


Fig. 7.17. Pre-driver to avoid crowbar in push pull stage

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:

$$V_{SSN} = N \cdot L \cdot \frac{\partial I}{\partial t} \quad (7.3)$$

where N is the number of switching outputs, L the equivalent inductance in which current must flow, and I the current per driver.

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.

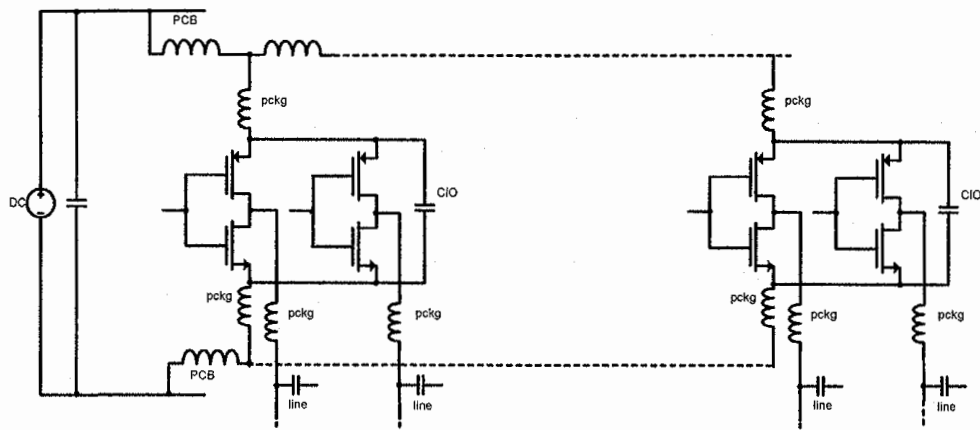


Fig. 7.18. Model example used to evaluate SSN

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

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.

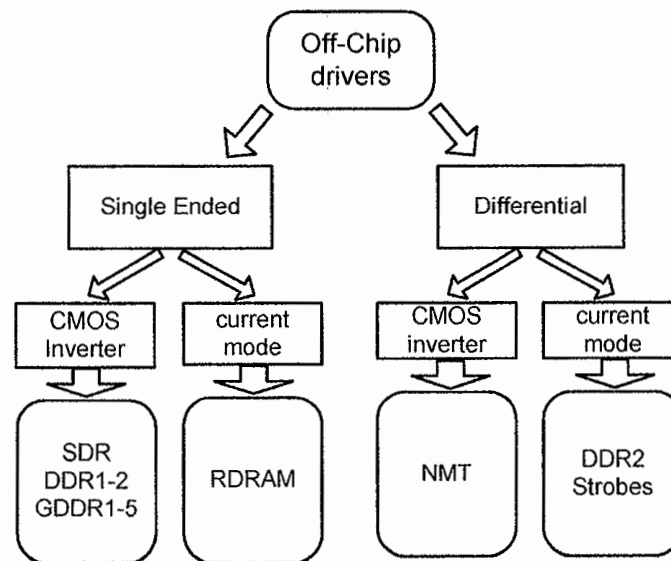


Fig. 7.19. A OCD taxonomy

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.

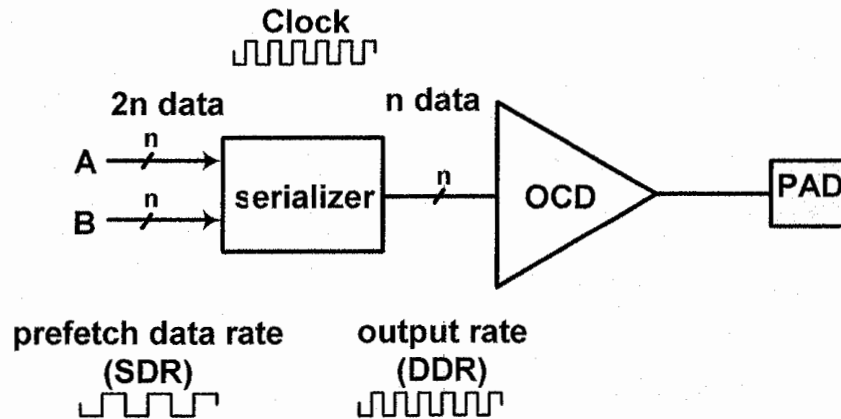


Fig. 7.20. OCD schematic block diagram

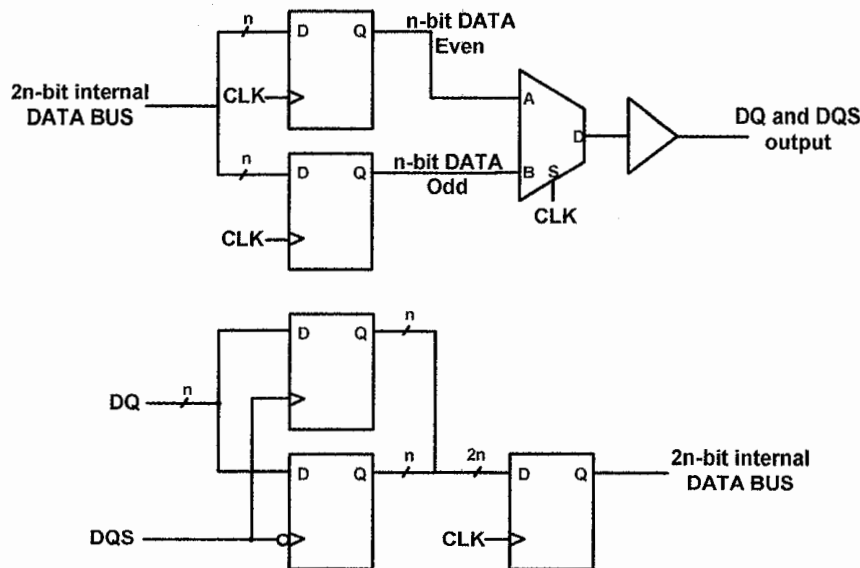


Fig. 7.21. Serializer/deserializer $2n$ prefetch Read and Write block diagram

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/Z_0 where Z_0 is the impedance of the line seen by the driver.

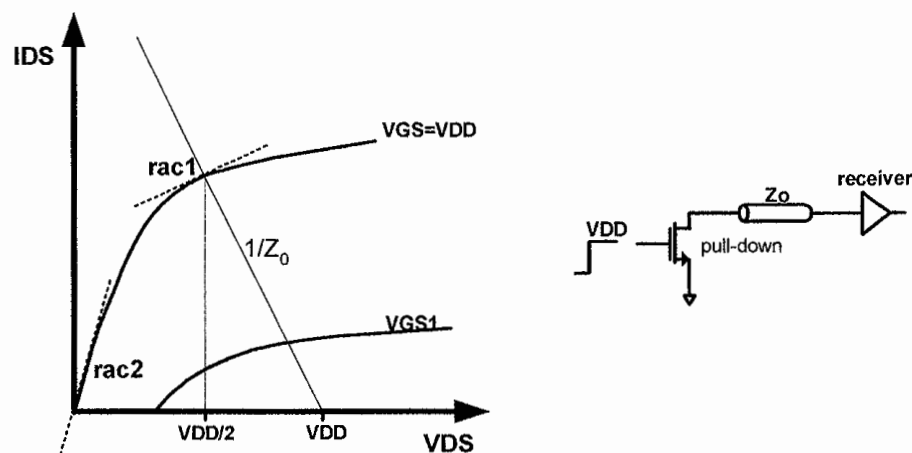


Fig. 7.22. Output impedance of driver (NMOS) and load line

The result is a load line with slope $1/Z_0$. In order to reach the target of launching the signal with a value of $V_{DD}/2$ – so that the signal at the end of line is reflected, reaching full V_{DD} swing – it is necessary that the intersection of the transfer characteristic with $V_{GS} = V_{DD}$ crosses $1/Z_0$ at $V_{DS} = V_{DD}/2$. In this point the signal is driven, and then the wave is reflected back and returns to the output node. V_{DS} is then decreased to a value between $V_{DD}/2$ 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 Z_0 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 $50\ \Omega$ 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 $V_{DD}/2$ but they are very different at small V_{DS} . 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.

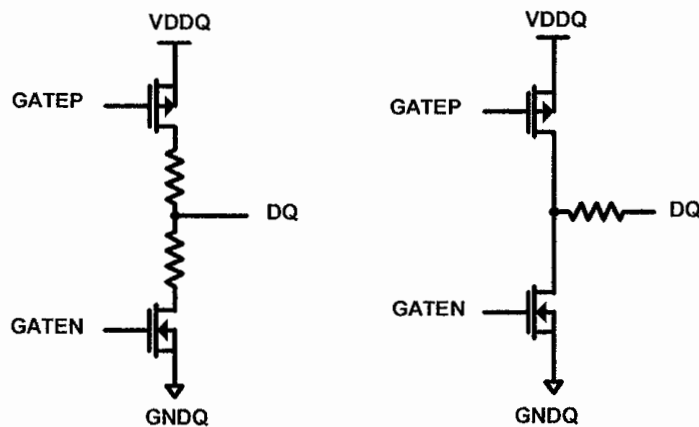


Fig. 7.23. OCD inverter stage with Linearization resistors

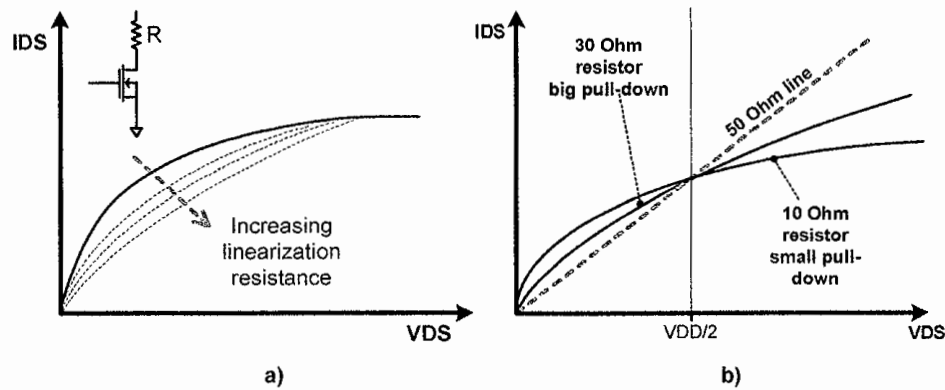


Fig. 7.24. Examples of choices for linearization resistor and driver dimensions

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 LVC MOS 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.

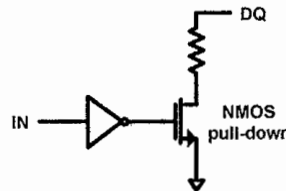


Fig. 7.25. Open drain OCD

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.

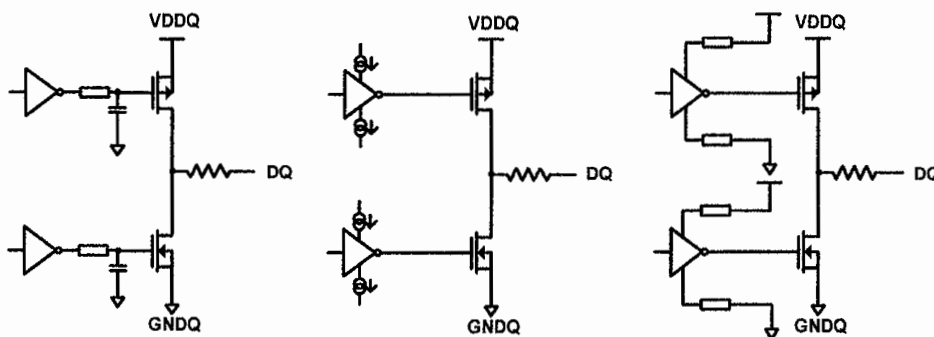


Fig. 7.26. Slew rate control by means of (a) gate slope (b) gate slope controlled (c) feedback on power supply

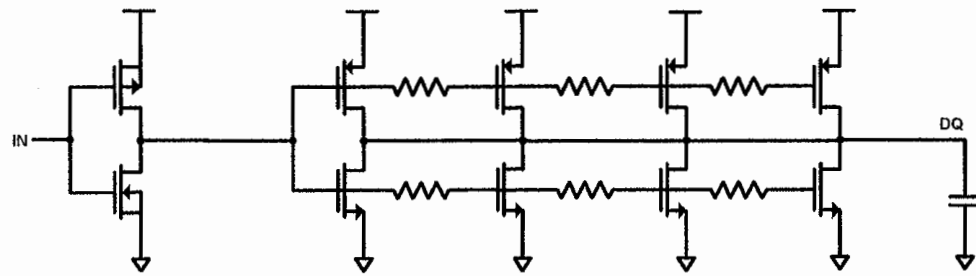


Fig. 7.27. Slew rate control by output driver time-distributed activation

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.

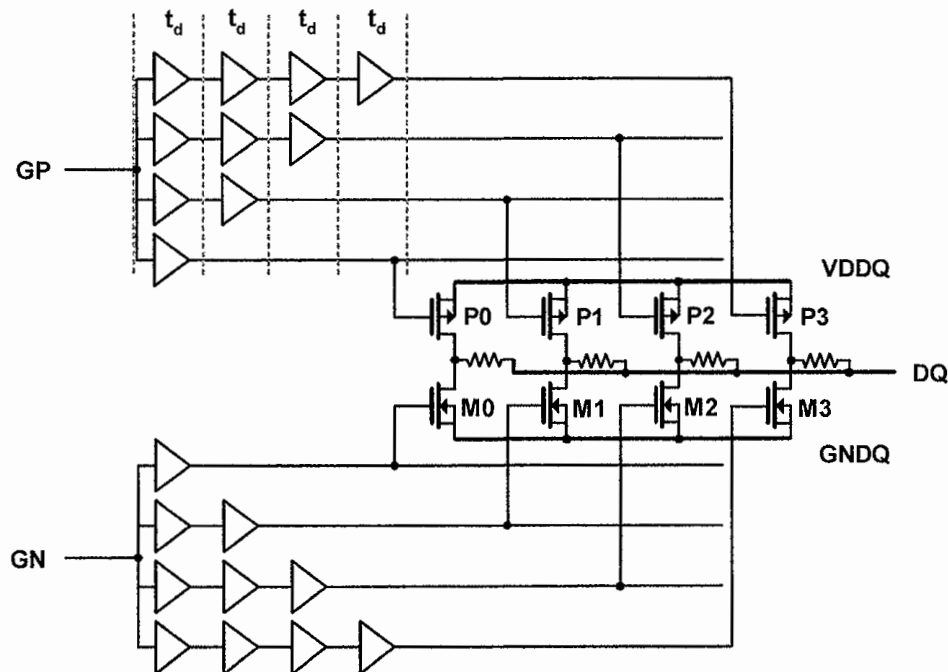


Fig. 7.28. Slew rate control by mean of digital time-domain split

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.

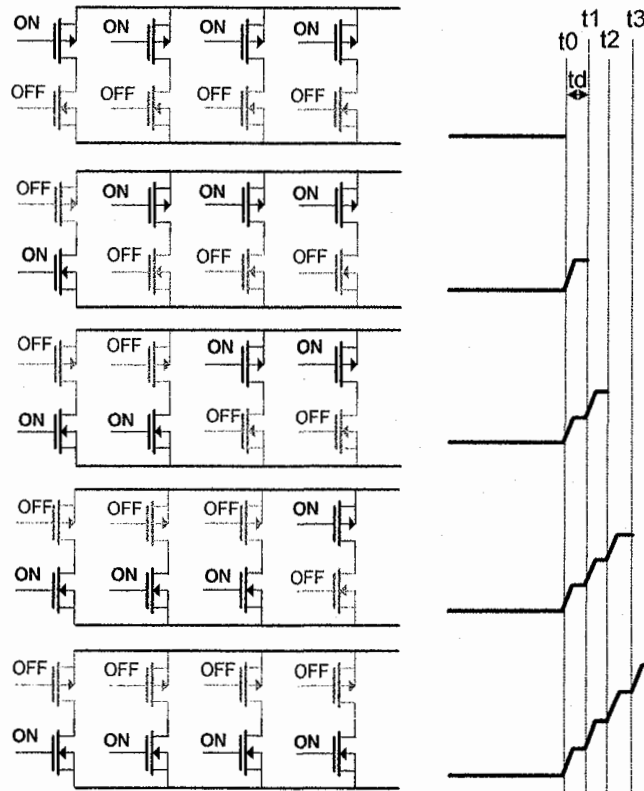


Fig. 7.29. time domain split time diagram

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

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.

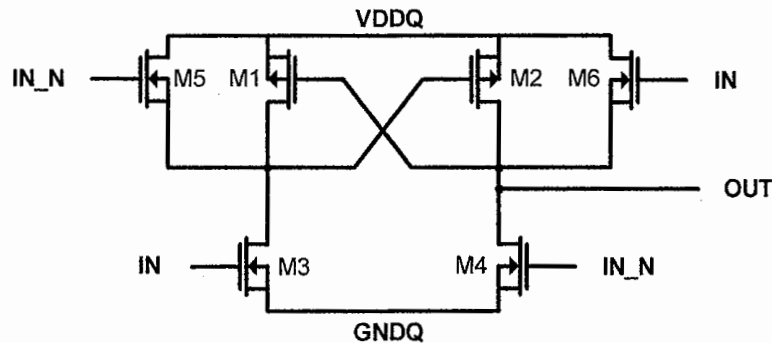


Fig. 7.30. Level shifter modified

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.

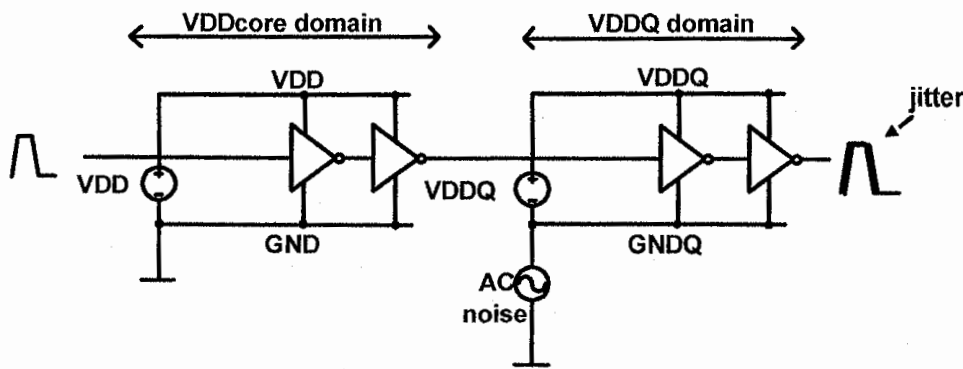


Fig. 7.31. Voltage domain change as a source of jitter

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

	I _{peak} (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

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.

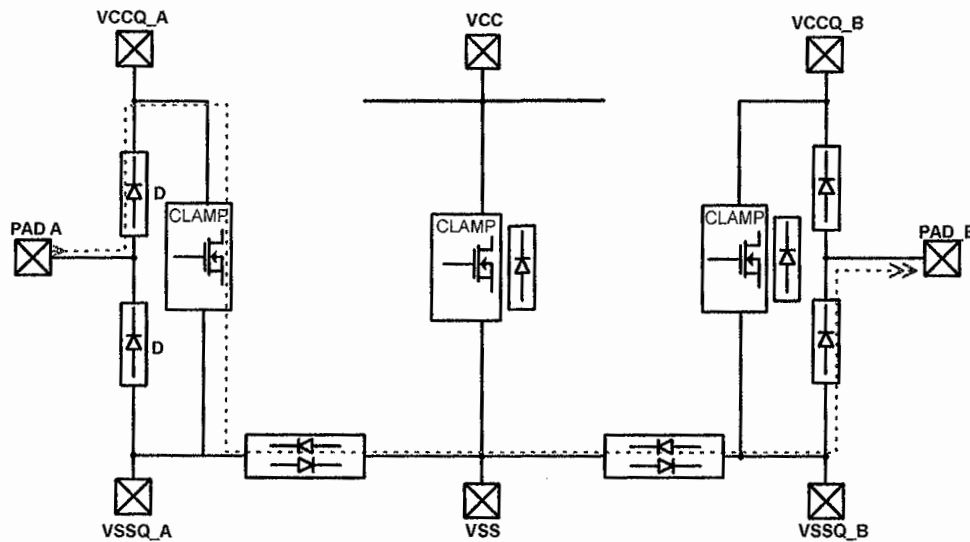


Fig. 7.32. Simplified ESD network

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].

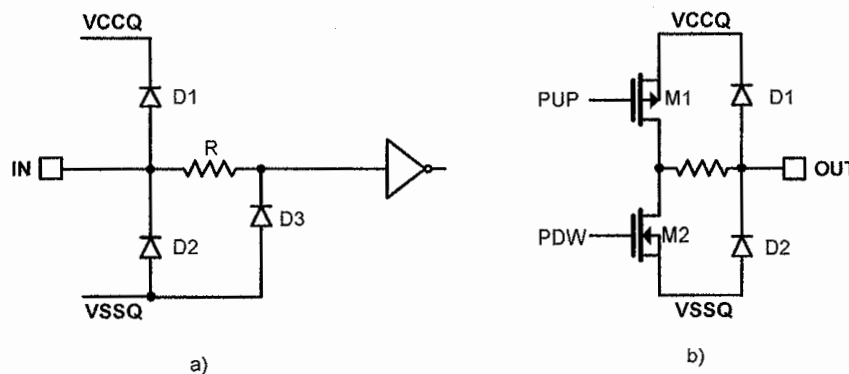


Fig. 7.33. (a) Input protection, (b) output protection

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.

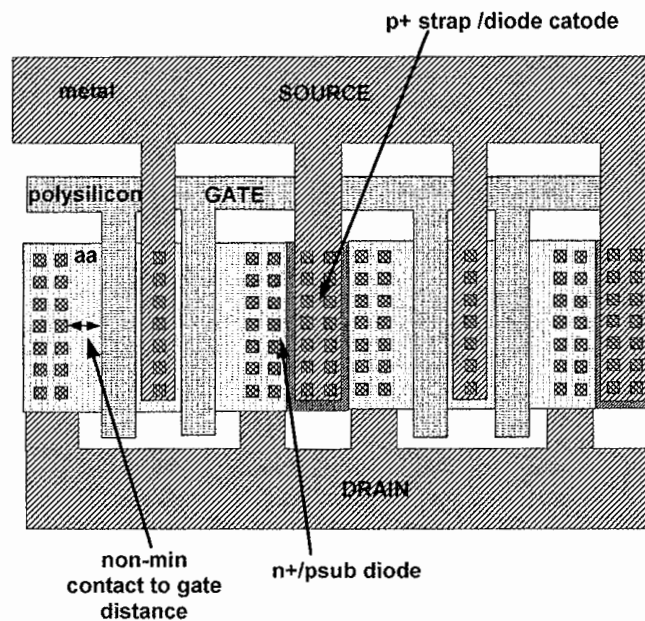


Fig. 7.34. NMOS pull-down layout example

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

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.

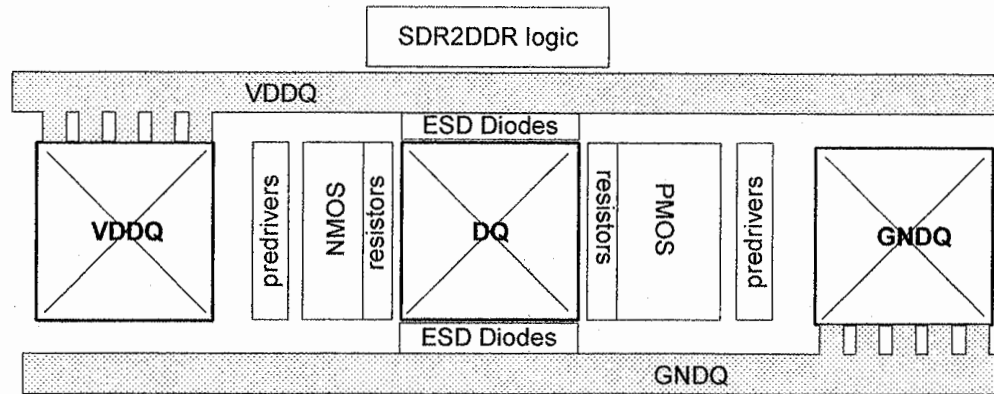


Fig. 7.35. OCD layout ideal floorplan

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.

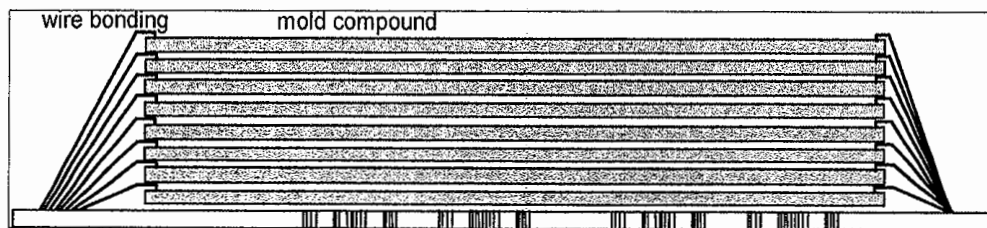


Fig. 7.36. Package with eight dies stacking on a Ball Grid Array substrate

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

- 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:

$$C_{spread} = 4 \cdot (C + \Delta) - 4 \cdot (C - \Delta) \quad (7.4)$$

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.

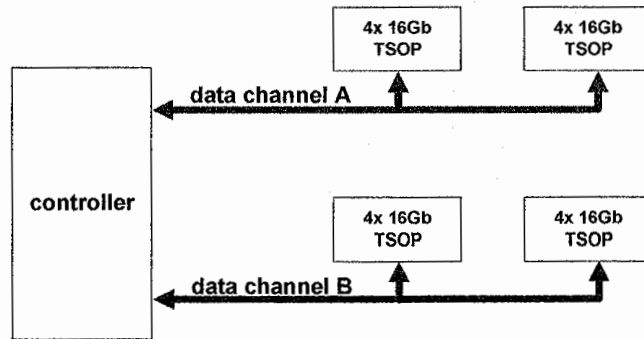


Fig. 7.37 32 GB system, dual channel using four dies TSOP

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.

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 Circuit 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. Hikedada, 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.

8 Sensing circuits

L. Crippa¹ and R. Micheloni²

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} .

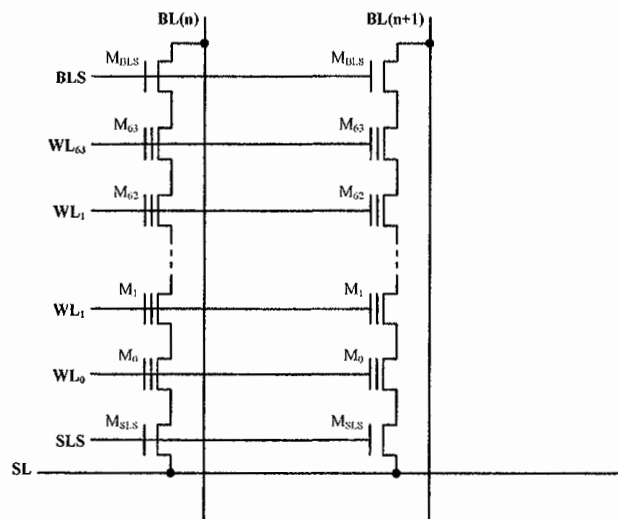


Fig. 8.1. Two strings in a NAND architecture

¹ Forward Insights, luca.crippa@ieee.org

² Integrated Device Technology, rino.micheloni@ieee.org

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:

$$I_{SSAT} = \frac{V_{BL}}{(n-1)R_{ON}} \quad (8.1)$$

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:

$$I_D = k \cdot \left[(V_{GS} - V_{TH}) \cdot V_{DS} - \frac{V_{DS}^2}{2} \right] \quad (8.2)$$

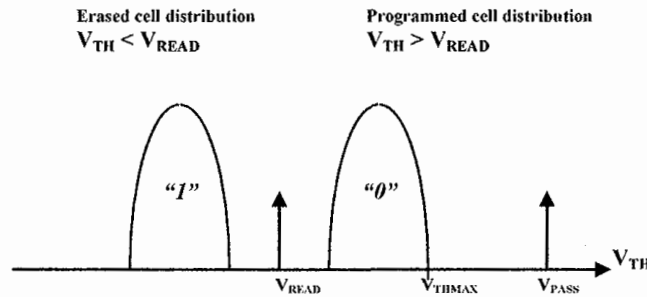


Fig. 8.2. Threshold voltage distributions for erased and programmed cells

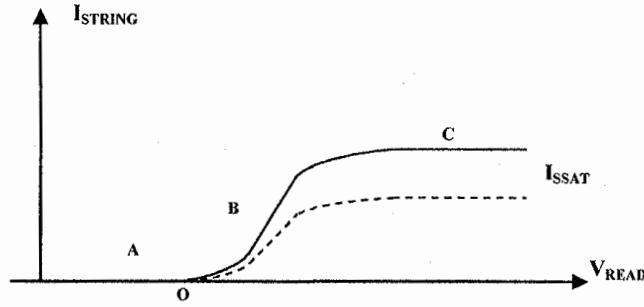


Fig. 8.3. Cell current characteristics versus gate voltage

For small V_{DS} values, as in our case, Eq. (8.2) may be simplified as:

$$I_D = k[(V_{GS} - V_{TH}) \cdot V_{DS}] \quad (8.3)$$

Therefore, R_{ON} is equivalent to

$$R_{ON} = \frac{V_{DS}}{I_D} = \frac{1}{k(V_{GS} - V_{TH})} \quad (8.4)$$

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:

$$\Delta V_C = \frac{I}{C} \Delta T \quad (8.5)$$

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).

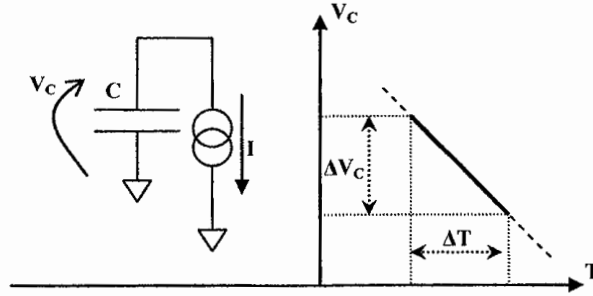


Fig. 8.4. Capacitor discharge through a constant current source

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:

$$T_{EVAL} = C_{BL} \frac{V_{PRE} - V_{SEN}}{I_{ERAMIN}} \quad (8.6)$$

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.

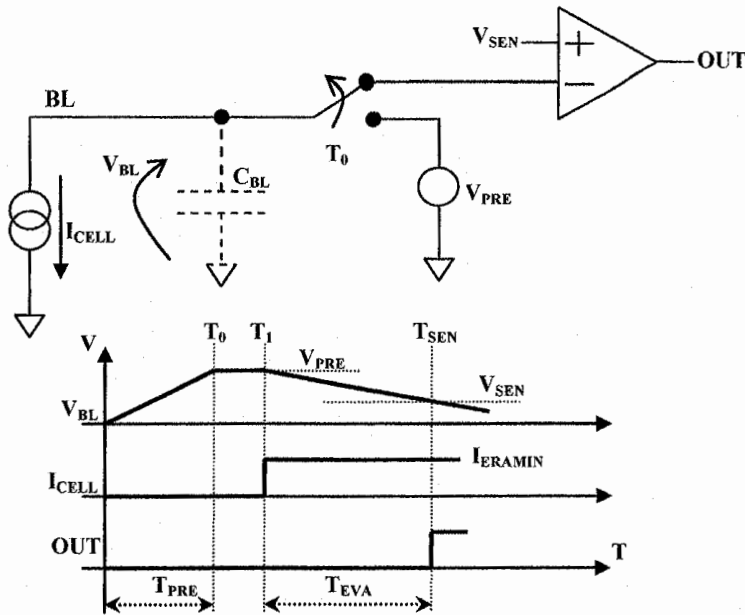


Fig. 8.5. Basic sensing scheme exploiting bitline capacitor and the related timing diagram

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 $(V'_{READ} + \epsilon)$ is inverted compared to the one resulting from a SRO at $(V'_{READ} - \epsilon)$. 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].

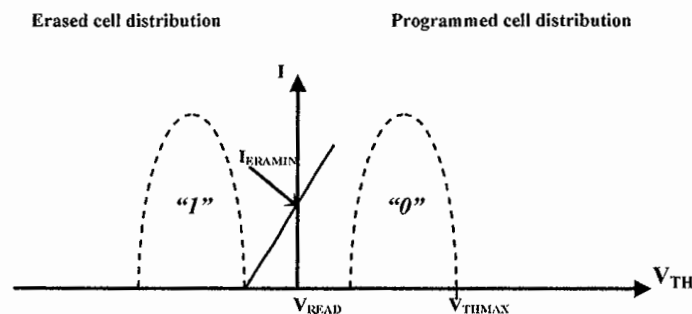


Fig. 8.6. Minimum erase current versus V_{READ}

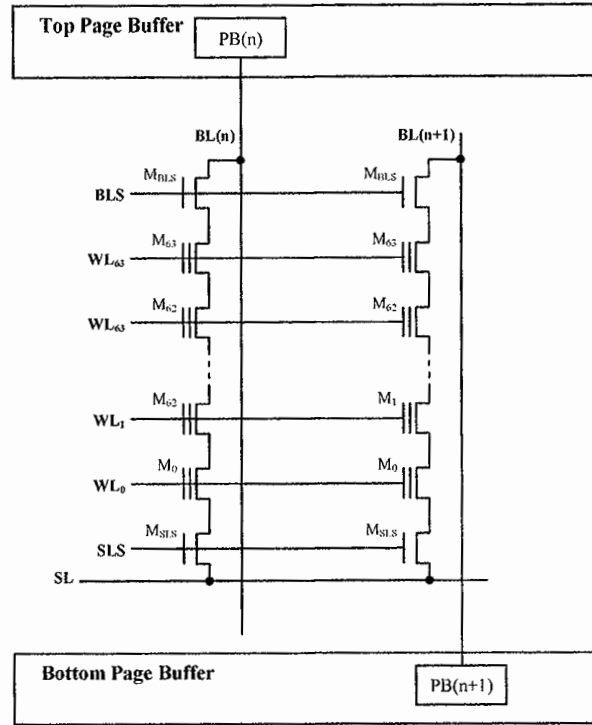


Fig. 8.7. Page buffer organization

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:

$$V_{BL} = V_{PRE} - V_{THN} \quad (8.7)$$

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 a 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 .

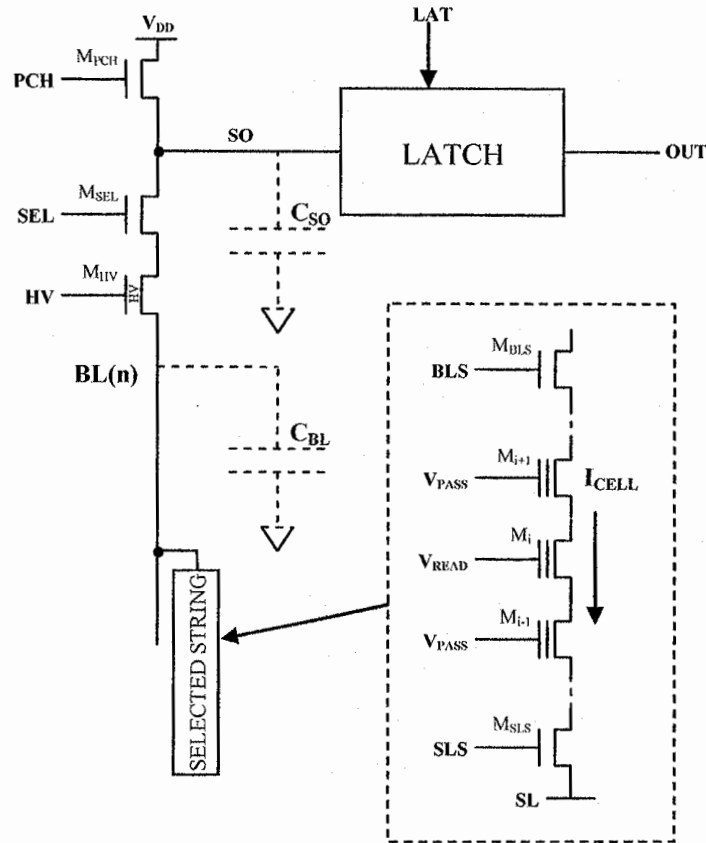


Fig. 8.8. Basic elements of the page buffer

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:

$$V_{BL} < V_{SEN} - V_{THN} \quad (8.8)$$

then M_{SEL} turns on, equalizing the voltage level of the SO node (V_{SO}) to V_{BL} . This is true because $C_{BL} \gg C_{SO}$; 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:

$$I_{READTH} = \frac{\Delta V \cdot C_{BL}}{T_{EVAL}} \quad (8.9)$$

where:

$$\Delta V = (V_{PRE} - V_{THN}) - (V_{SEN} - V_{THN}) = V_{PRE} - V_{SEN} \quad (8.10)$$

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):

$$T_{EVAL} = \frac{\Delta V \cdot C_{BL}}{I_{READTH}} \quad (8.11)$$

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:

$$T_{EVAL} = \frac{100mV \cdot 2pF}{50nA} = 4\mu s \quad (8.12)$$

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.

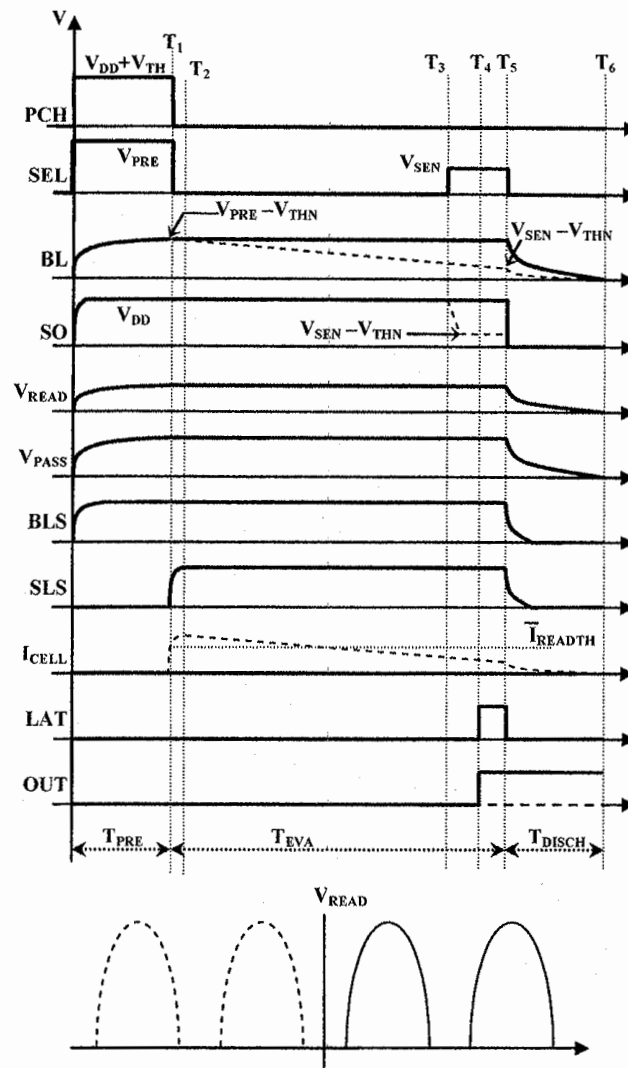


Fig. 8.9. Single read operation (SRO) timing diagram

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.

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:

$$V_{THL} > V_{SEN} - V_{THN}; V_{THL} < V_{DD} \quad (8.13)$$

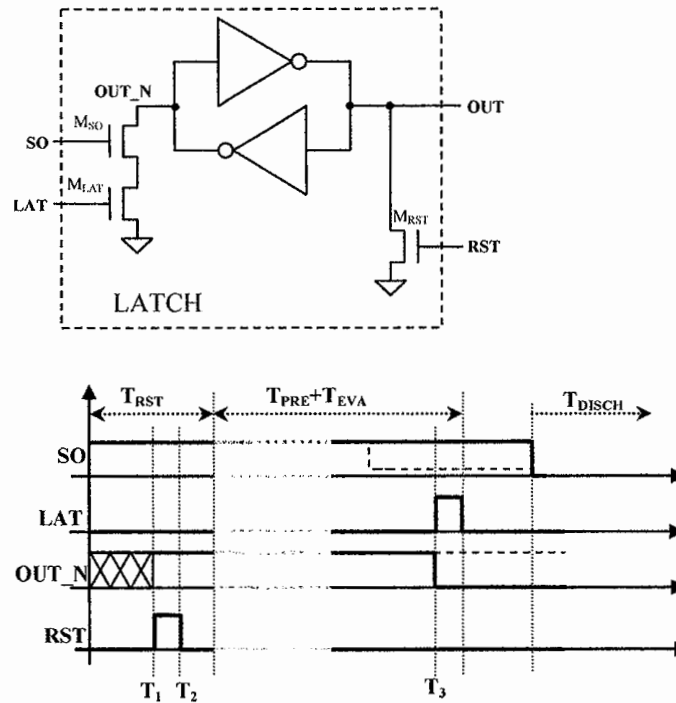


Fig. 8.10. First Latch circuitual solution and its timing diagram

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

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 a 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} .

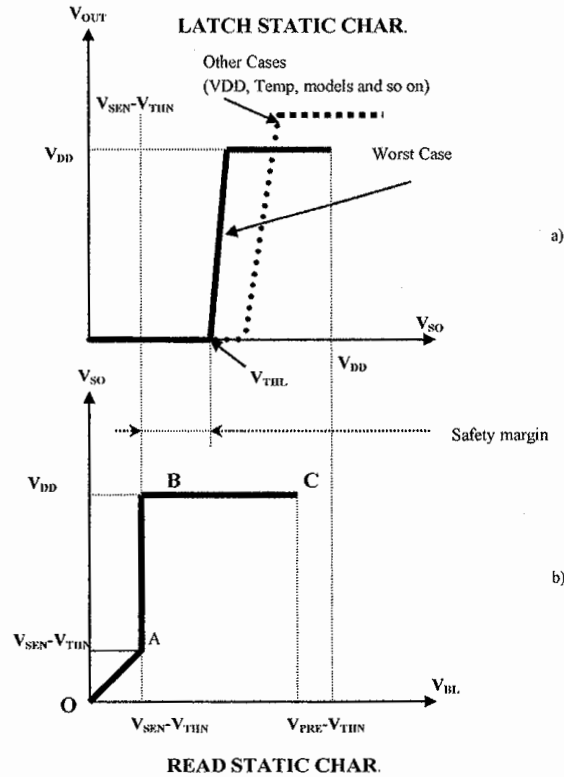


Fig. 8.11. (a) Latch characteristic; (b) read characteristic

At T_4 also I2 is enabled and the positive feedback of the latch takes place. Finally, at T_5 , M_{SO} is disabled and the latch holds its state. The critical phase is

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.

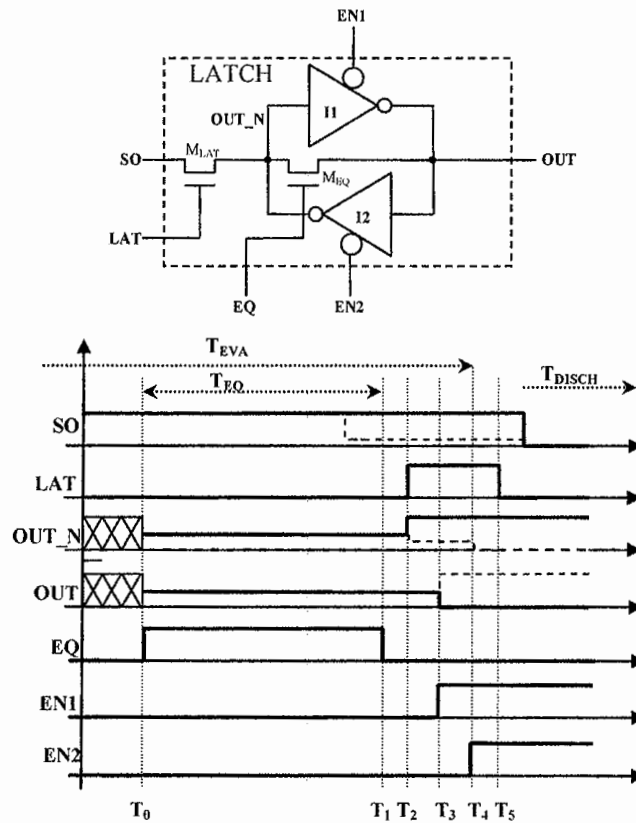


Fig. 8.12. Second latch circuitual solution and its timing diagram

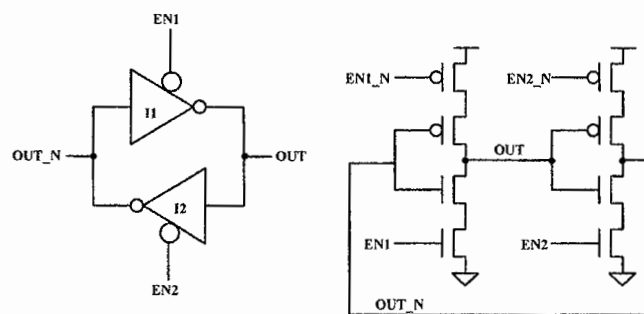


Fig. 8.13. Three-state latch

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:

$$C_{BL} = C_{AU} + C_{AD} + 2C_C + 2C_{C2} \quad (8.14)$$

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 and 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:

$$I_{READTH_MAX} = \frac{\Delta V \cdot C_{BL_MAX}}{T_{EVAL}} \quad (8.15)$$

$$I_{READTH_MIN} = \frac{\Delta V \cdot C_{BL_MIN}}{T_{EVAL}} \quad (8.16)$$

Of course, looking at Fig. 8.15, this relation has to be satisfied:

$$I_{READTH_MAX} < I_{SSAT} \quad (8.17)$$

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 rely on circuit simulators in order to verify that every parameter has been sized correctly.

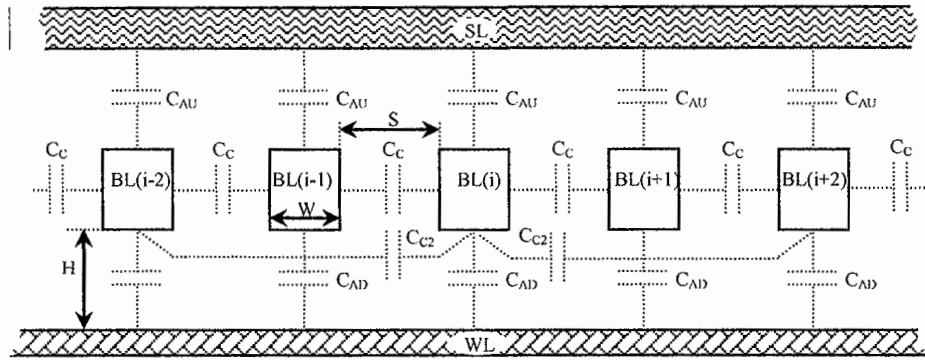


Fig. 8.14. Bitline parasitic capacitors

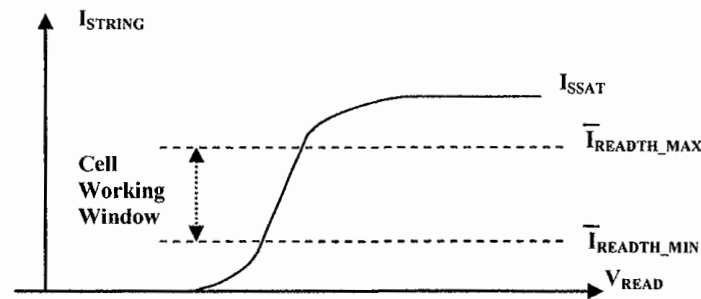


Fig. 8.15. Cell working window

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:

$$I_{READ_NOISE} = \frac{+\Delta V_{BL} \cdot C_{BL}}{T_{EVAL}}$$

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:

$$\Delta V_{BL} = -\frac{C_C + C_{C'}}{C_{BL}}(V_{PRE} - V_{SEN}) \quad (8.18)$$

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 BL_e are read, M_{SLe} acts as a pass-transistor. Transistor M_{SELo} is turned off. The $DISo$ signal turns on the M_{DISo} transistor, forcing the odd bitline BL_o to the fixed BIAS voltage. M_{DSe} is turned off.

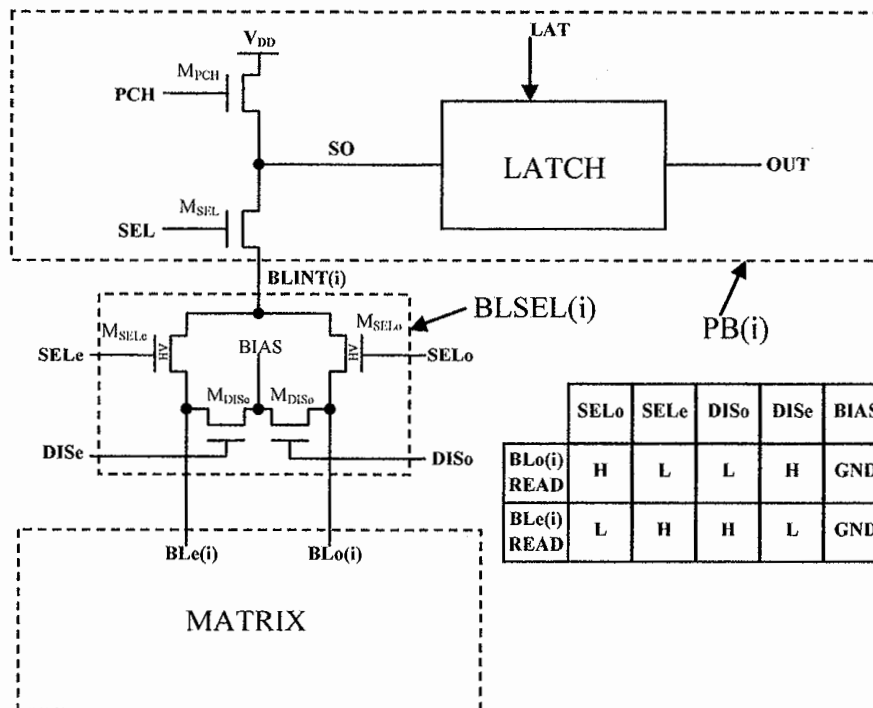


Fig. 8.16. Interleaving bitline architecture

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 $V_{\text{THR}} < V_{\text{READ}}$ 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 $\text{BL}_e(i)$ and $\text{BL}_o(i)$ ends in the $\text{BLSEL}(i)$ which has the output line $\text{BLINT}(i)$ [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.

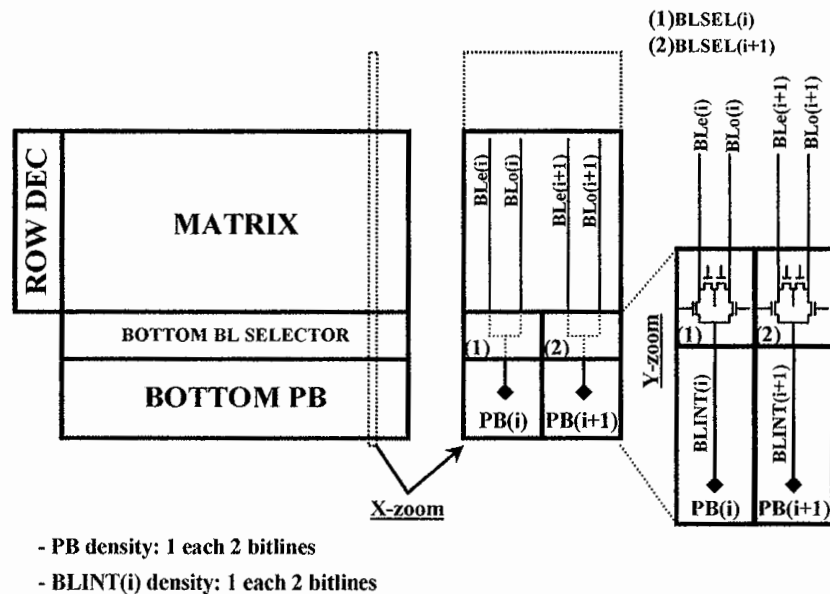


Fig. 8.17. Single side interleaving architecture

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.

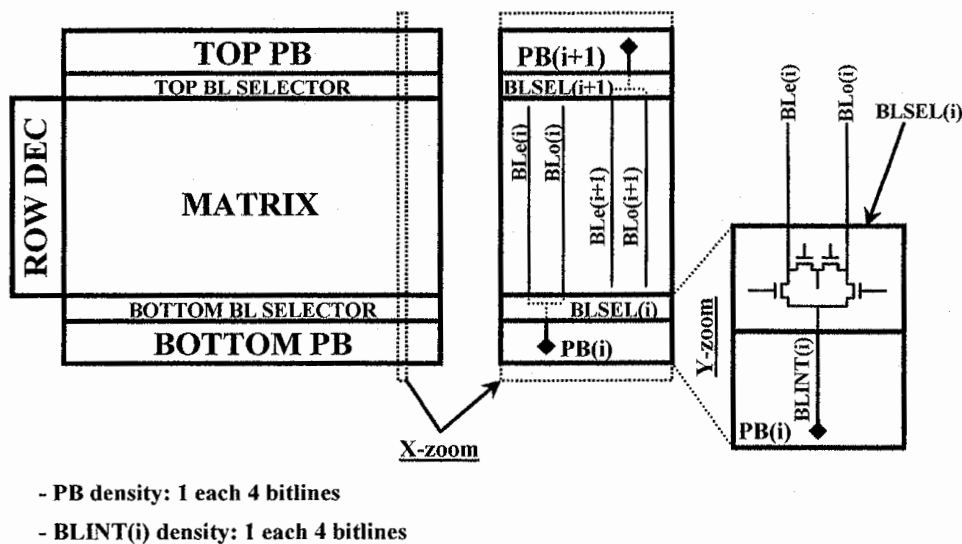


Fig. 8.18. Double side interleaving architecture

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.

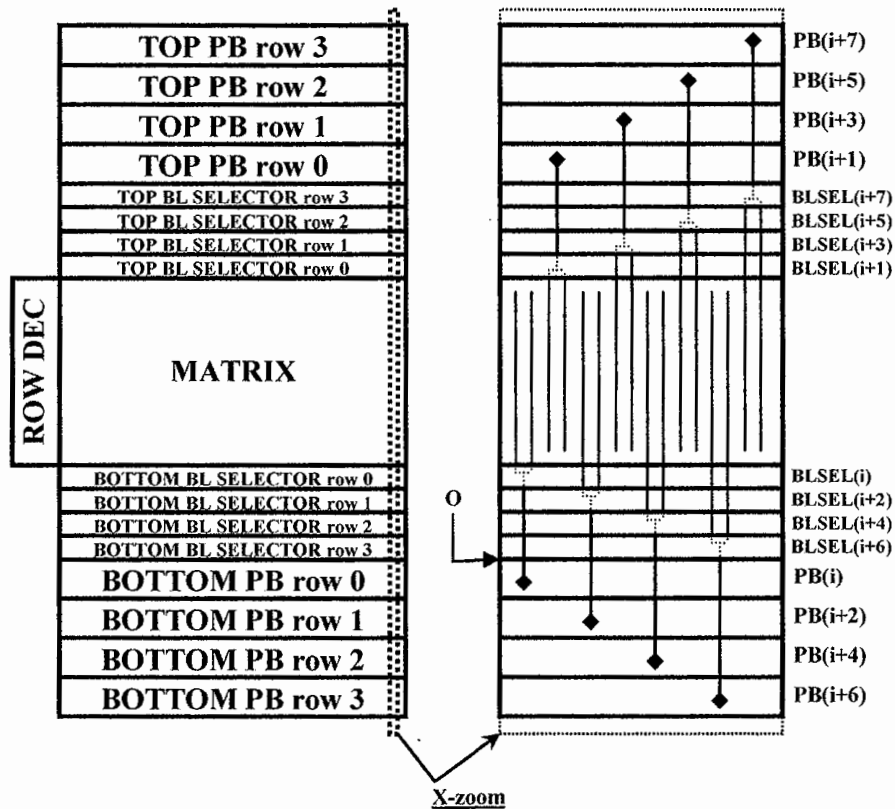
The page buffer is constituted by transistors having layout rules (strictly related to power supply voltage) that remain mostly unchanged through the different technological nodes. Therefore, as technology goes down to finer geometries, the page buffer has less space available in the orthogonal direction 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.

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:

$$PBPITCH = (32 + 32) \cdot 36nm = 2,304 \mu m \quad (8.19)$$

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.



- PB density: 1 each 16 bitlines
- BLINT(i) density: 1 each 4 bitlines

Fig. 8.19. Double-side interleaving architecture with four stacked PB rows

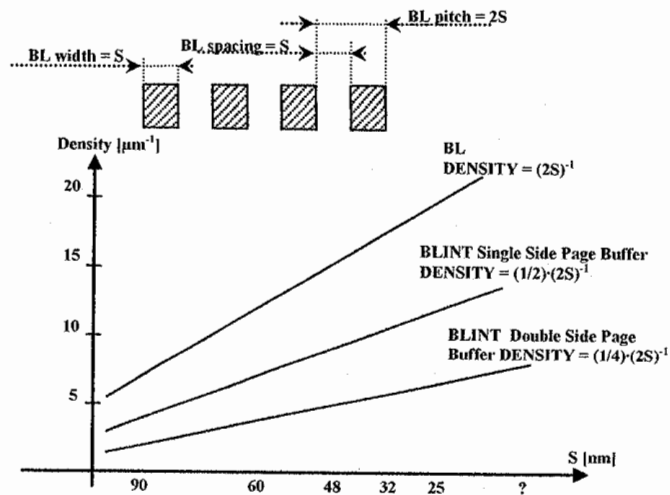


Fig. 8.20. BL and BLINT density versus the technology node S

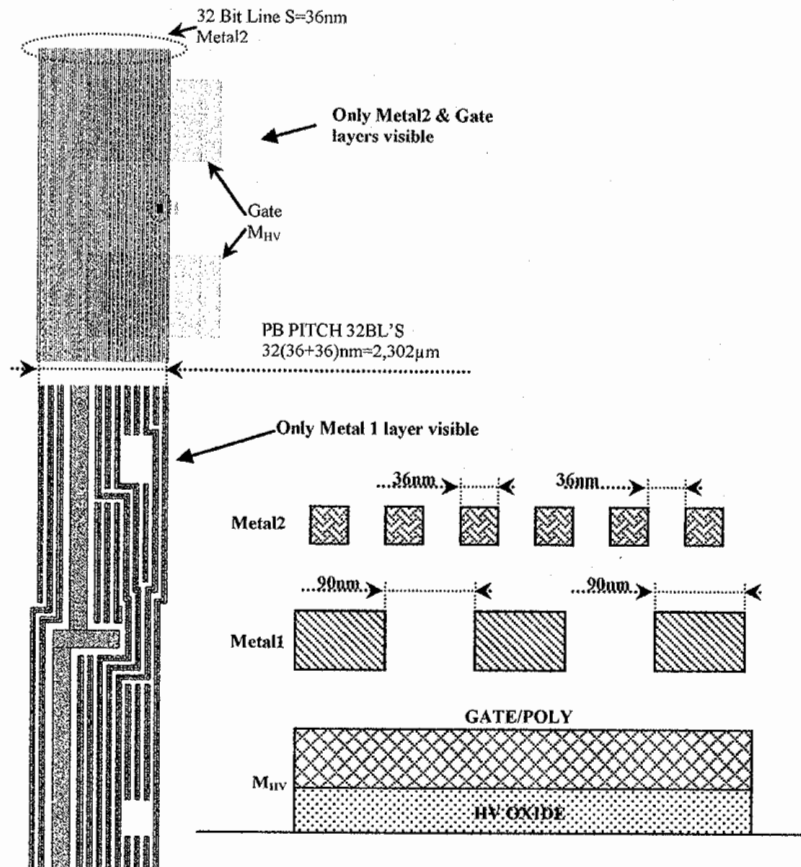


Fig. 8.21. 36 nm technology node: BL and BLINT layout in BLSEL circuitry

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 metal 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 metal 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.

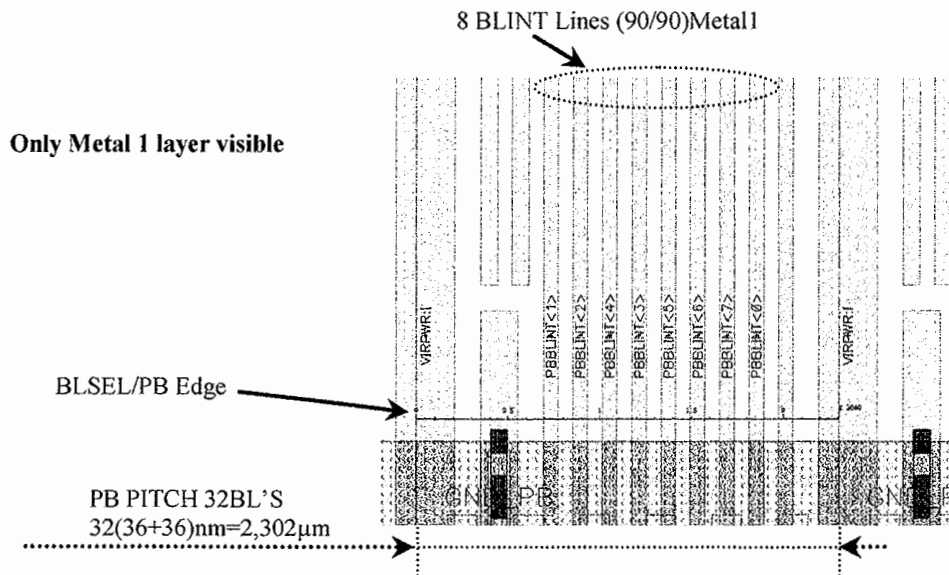


Fig. 8.22. 36 nm BLINT density in BLSEL/PB boundary area

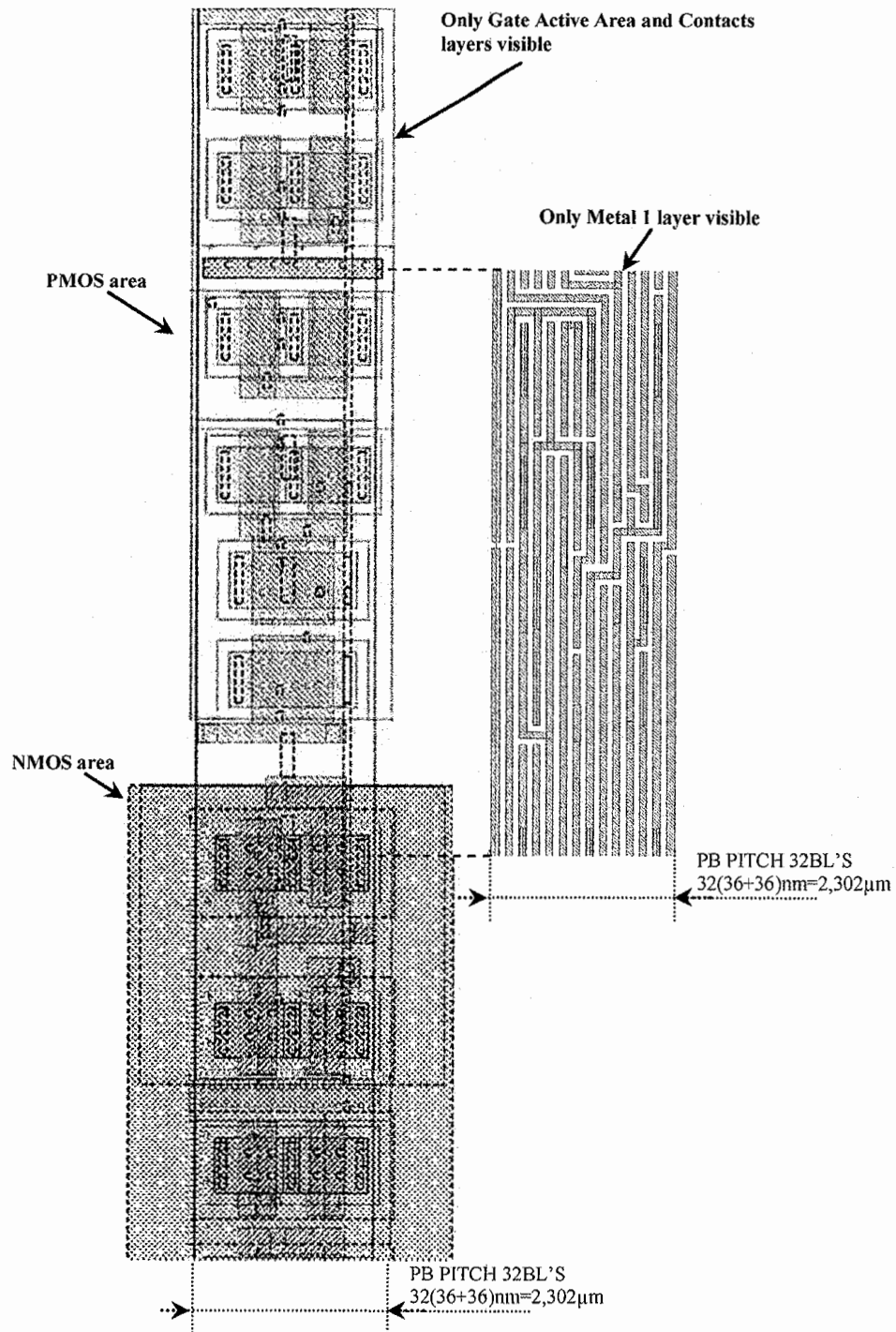


Fig. 8.23. PB layout in 36 nm NAND technology: gate, contact and metall layers are shown

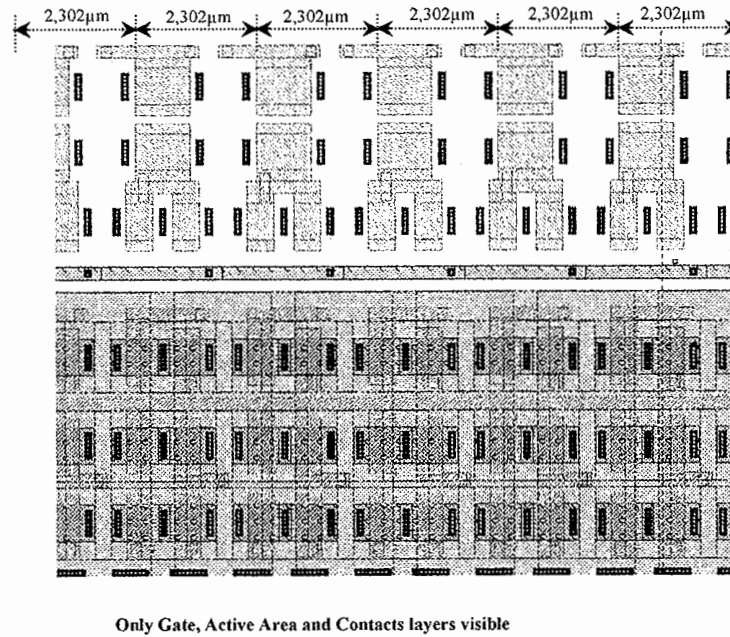


Fig. 8.24. 36 nm layout portion of a PB row: gate and metal1

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:

$$i_{BLC01} = C_{BL01} \cdot \frac{\partial (V_{BL0} - V_{BL1})}{\partial t}; \quad i_{BLC02} = C_{BL02} \cdot \frac{\partial (V_{BL0} - V_{BL2})}{\partial t}$$

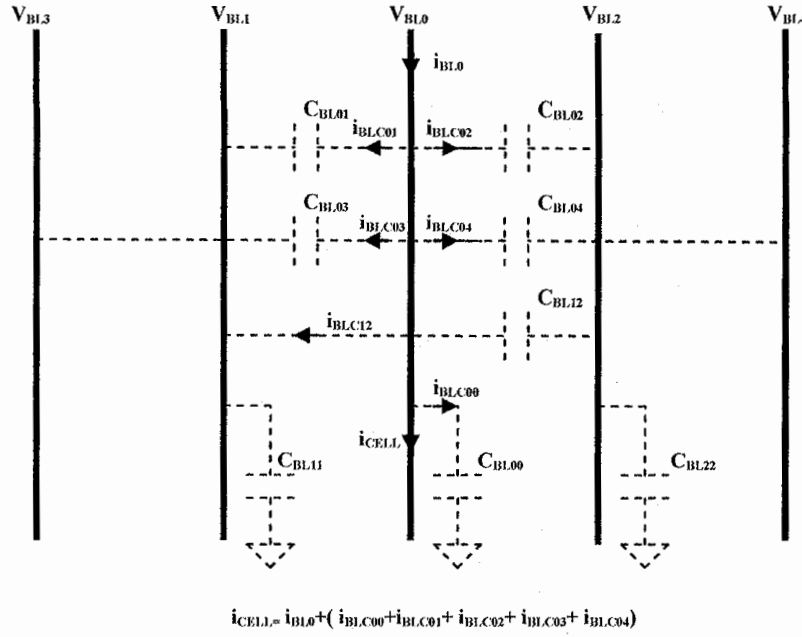


Fig. 8.25. Parasitic capacitors among adjacent bitlines

It follows that the current flowing along BL_0 is:

$$i_{BL0} \approx i_{CELL} + (i_{BLC00} + i_{BLC01} + i_{BLC02}) \quad (8.20)$$

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:

$$i_{BL0} \approx i_{CELL} + (i_{BLC00} + i_{BLC01} + i_{BLC02} + i_{BLC03} + i_{BLC04}) \quad (8.21)$$

Considering all the contributions given by all the non adjacent bitlines, we have:

$$i_{BL0} = i_{CELL} + (i_{BLC00} + i_{BLC01} + i_{BLC02} + i_{BLC03} + i_{BLC04} + \dots) \quad (8.22)$$

In state-of-the-art technologies, all the non adjacent bitlines contributions can be estimated as about 10% of the adjacent bitlines contribution.

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} HVN MOS 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.

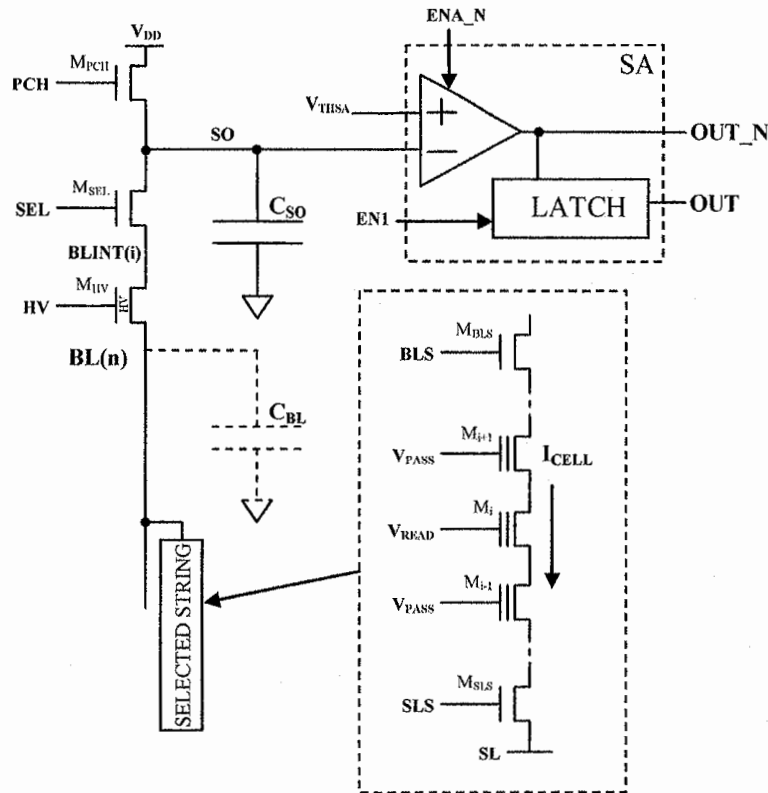


Fig. 8.26. ABL sense amplifier

The precharge final condition

$$V_{BL} = V_{PRE} - V_{THN} \quad (8.23)$$

is, therefore, valid only for the bitlines which have an associated string in a non conductive state.

Equation (8.23) should be replaced by:

$$V_{BL} = V_{PRE} - V_{THN} - \Delta \quad (8.24)$$

where Δ is the voltage drop on the bitlines resistance (typical values are in the order of hundreds of kiloOhm up to 1 M Ω). 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.

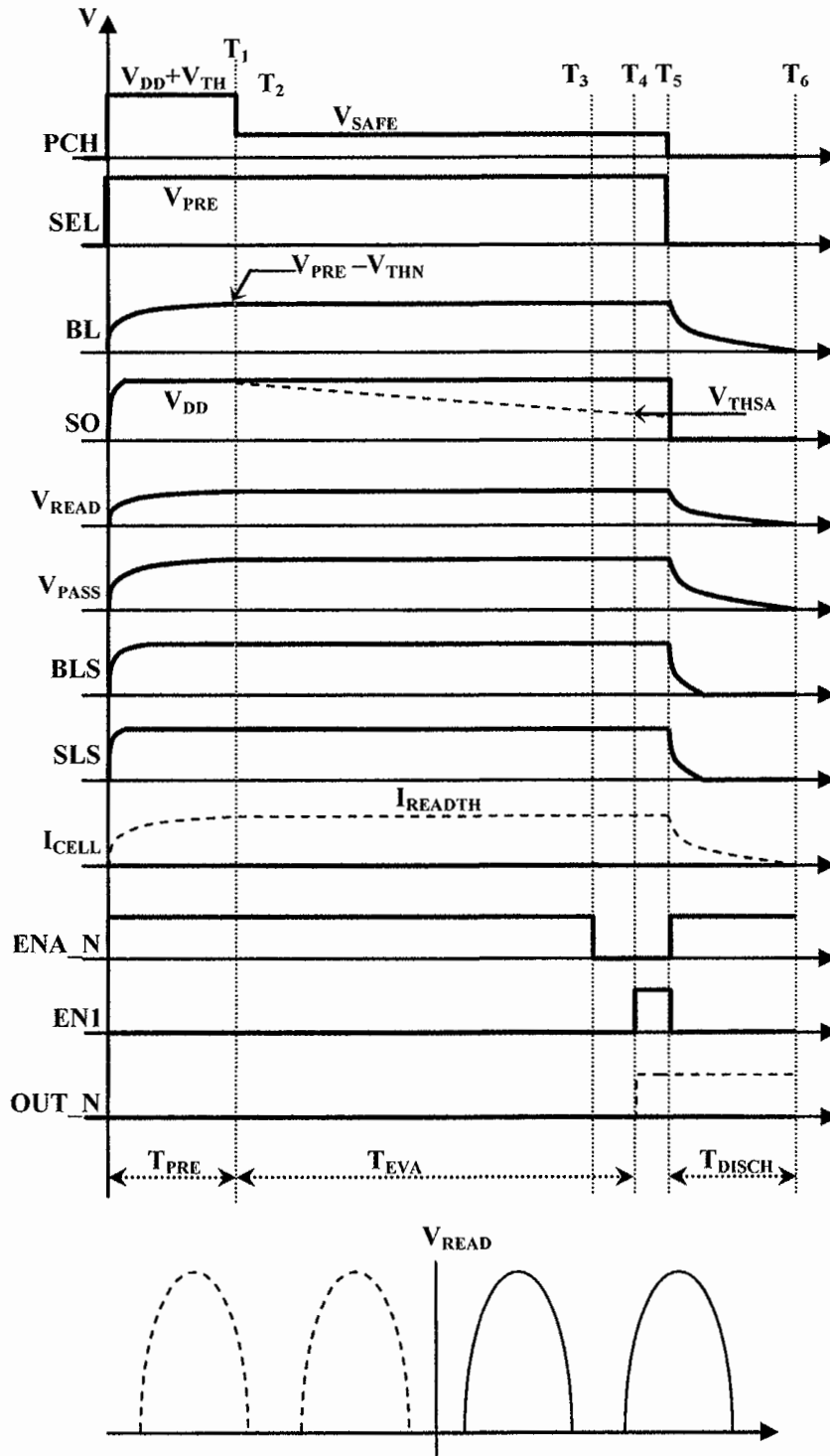


Fig. 8.27. ABL single read operation (SRO) timing diagram

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:

$$I_{READTH} = \frac{\Delta V \cdot C_{SO}}{T_{EVAL}} \quad (8.25)$$

where

$$\Delta V = V_{DD} - V_{THSA} \quad (8.26)$$

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:

$$T_{EVAL} = \frac{\Delta V \cdot C_{SO}}{I_{READTH}} \quad (8.27)$$

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 μ s 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:

$$V_{SAFE} - V_{THN} \geq V_{PRE} - V_{THN} \Rightarrow V_{SAFE} > V_{PRE} \quad (8.28)$$

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:

$$V_{SAFE} - V_{THN} \leq V_{THSA} \quad (8.29)$$

Therefore, from Eqs. (8.28) and (8.29), the following conditions must hold true:

$$V_{PRE} - V_{THN} \leq V_{SAFE} - V_{THN} \leq V_{THSA} \quad (8.30)$$

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:

$$V_{THSA} = V_{DD} - V_{THP} \quad (8.31)$$

Replacing Eq. (8.31) in Eq. (8.26) we obtain:

$$\Delta V = V_{DD} - (V_{DD} - V_{THP}) = V_{THP} \quad (8.32)$$

and from Eq. (8.25):

$$I_{READ_TH} = \frac{V_{THP} \cdot C_{SO}}{T_{EVAL}} \quad (8.33)$$

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:

$$V_{THSA} > V_{PRE} - V_{THN} \quad (8.34)$$

This is automatically verified if Eq. (8.30) holds true.

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 $V_{DD} - V_{THP}$.

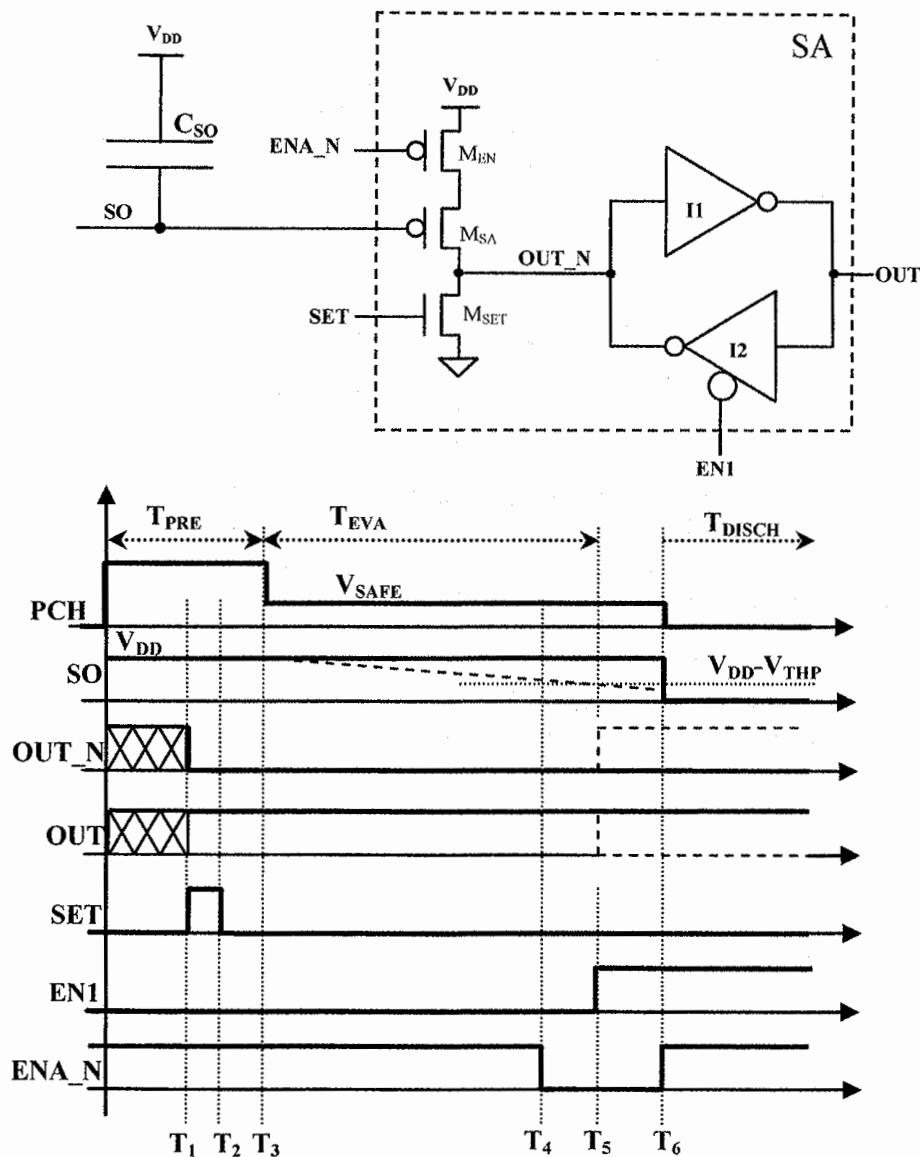


Fig. 8.28. ABL sense amplifier and its timing diagram

8.4 ABL versus interleaving architecture

In this section a comparison between ABL and the interleaving architecture is presented [25].

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:

$$I_{PRE} = C_{BL} \frac{\Delta V}{T_{PRE}} n_{BL} \quad (8.35)$$

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 μ s, 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).

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.

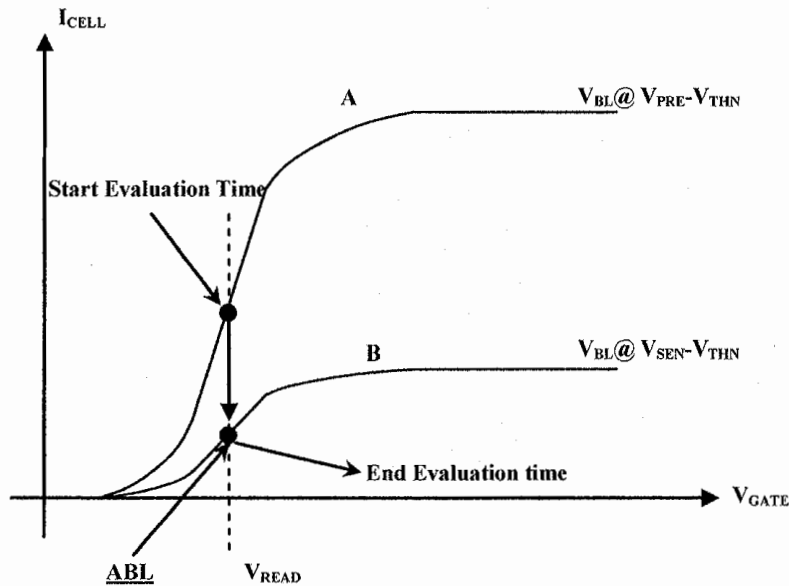


Fig. 8.29. Cell working point: ABL versus interleaving

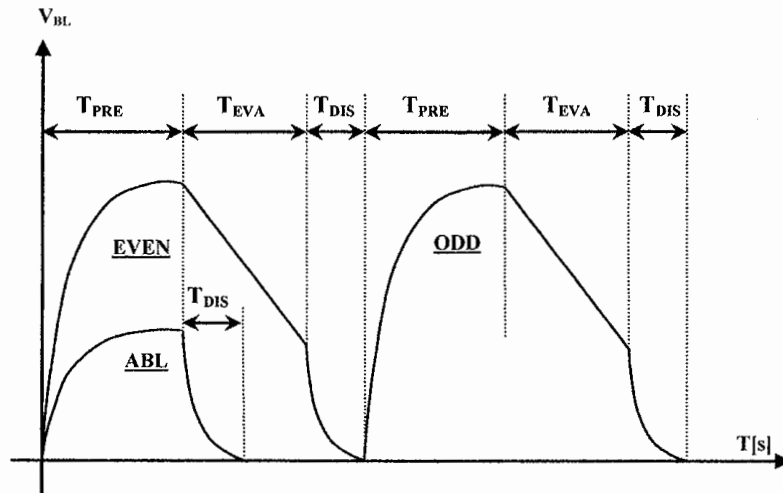


Fig. 8.30. Bitline voltage: ABL versus interleaving

As previously described, the drawback of the ABL architecture is the static current consumption during the precharge phase. This is due to cells with $V_{THR} \ll V_{READ}$. 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].

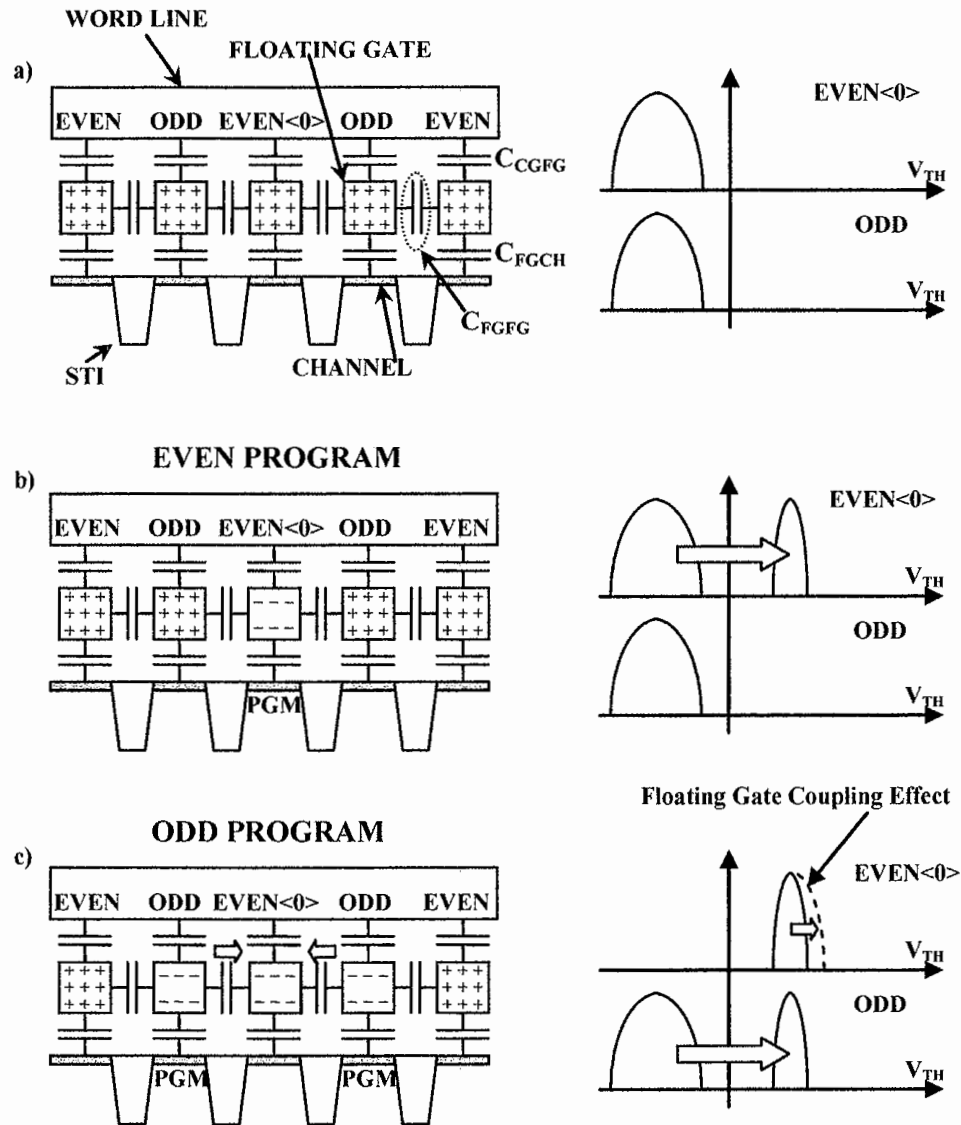


Fig. 8.31. Interleaving architecture: floating-gate-coupling effect during a program operation

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].

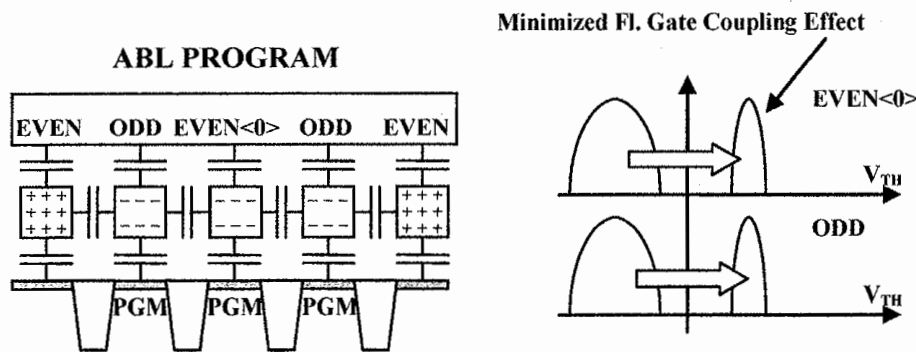


Fig. 8.32. Floating-gate-coupling effect during program with ABL architecture

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

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.

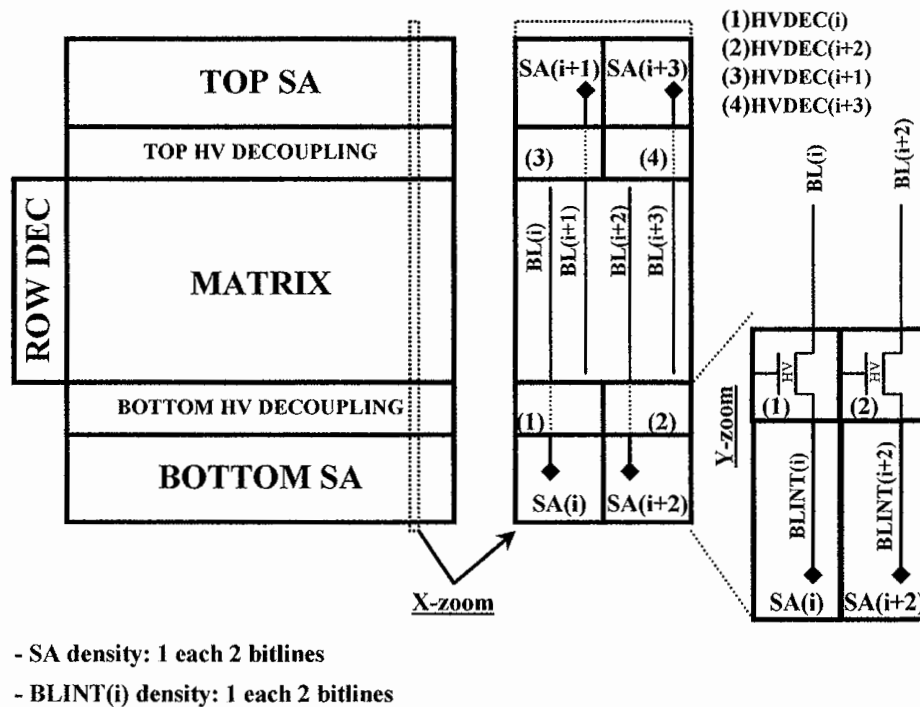


Fig. 8.33. Double-side ABL architecture

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.

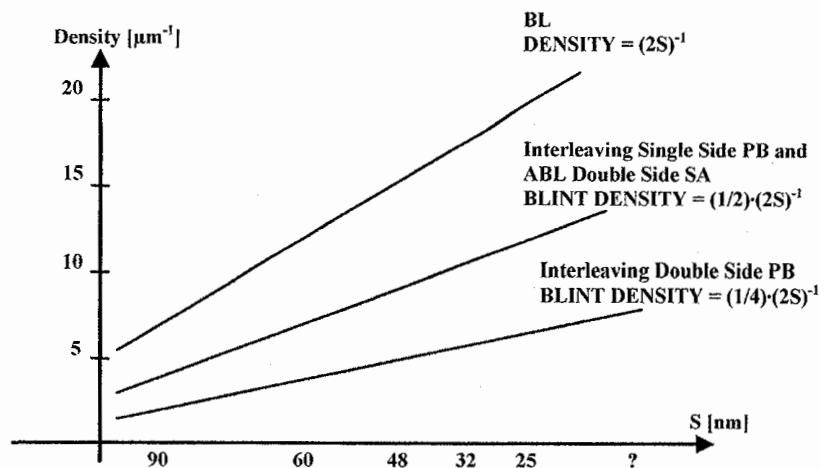


Fig. 8.34. BLINT density ABL versus interleaving

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 Only 32 Mb NAND Flash EEPROM* 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 0.12 μm 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.

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.

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).

9 Parasitic effects and verify circuits

L. Crippa¹ and R. Micheloni²

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 $M_1 - M_0$).

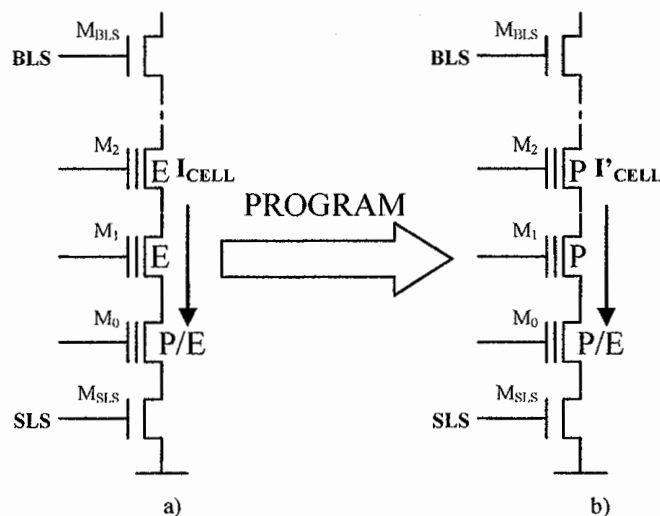


Fig. 9.1. Cell M_0 current modulation due to Background Pattern Dependency (BPD)

¹ Forward Insights, luca.crippa@ieee.org

² Integrated Device Technology, rino.micheloni@ieee.org

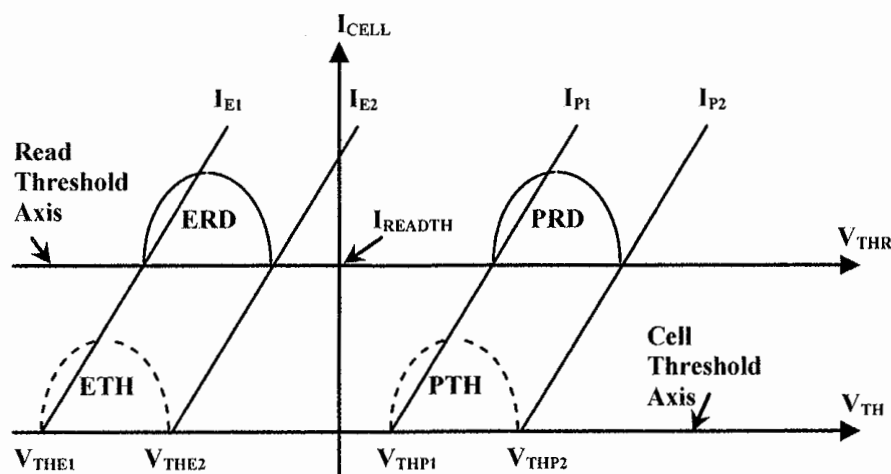


Fig. 9.2. Correspondence between V_{TH} and V_{THR}

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.

Figure 9.4 shows that a reduction of the working point ($I'_{\text{READTH}} < I_{\text{READTH}}$) 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.

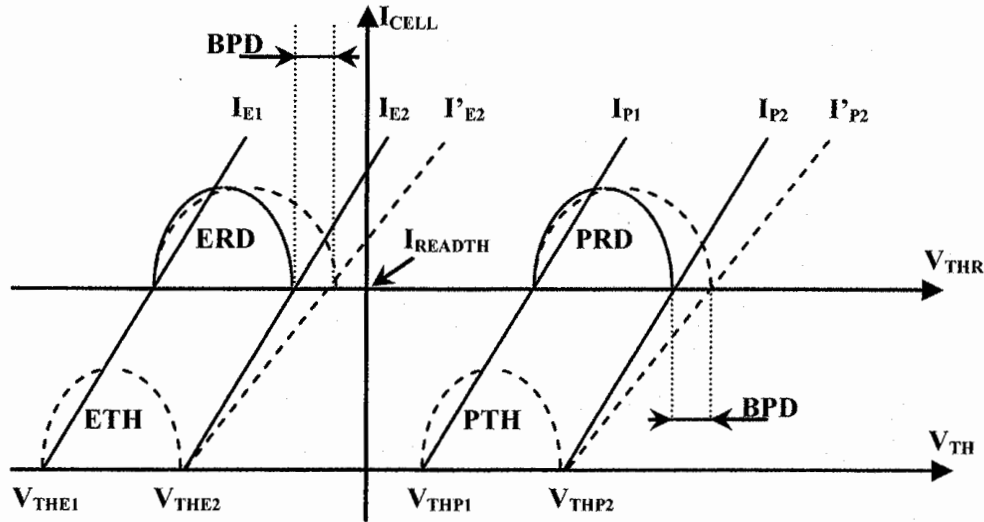


Fig. 9.3. BPD effects on Read Threshold Axis (V_{THR})

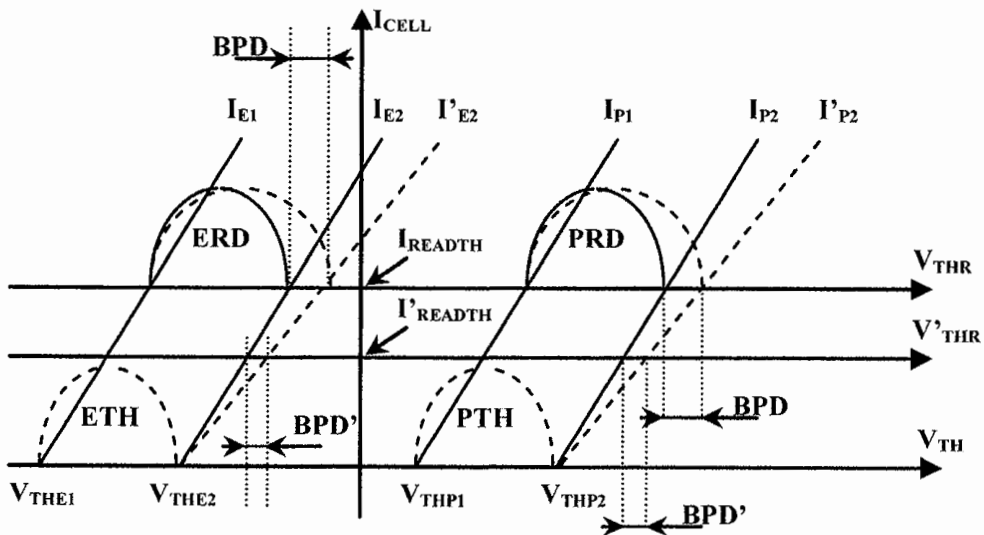


Fig. 9.4. Lower cell current working point reduce BPD effect

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.

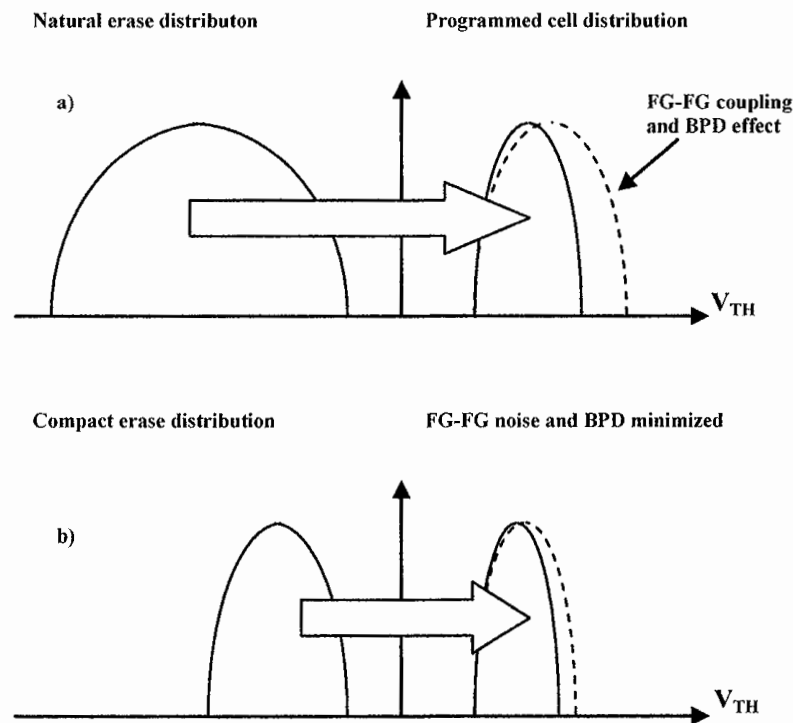


Fig. 9.5. Benefit of a compact erased distribution

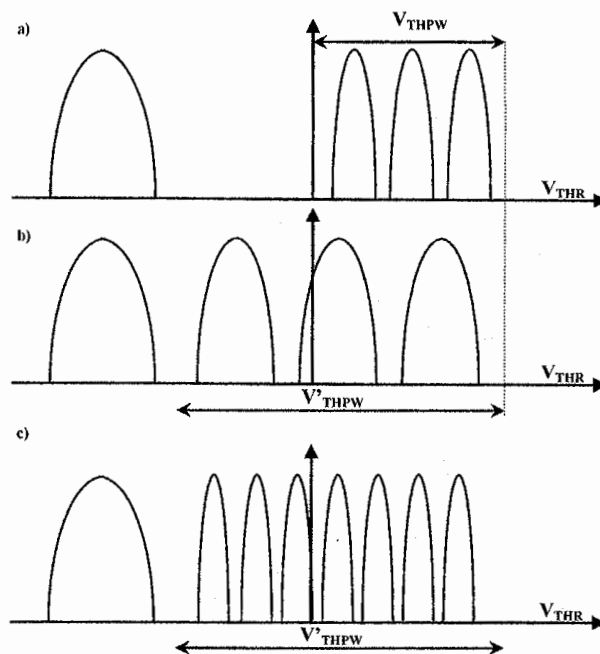


Fig. 9.6. Negative sensing expand program threshold window

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

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 $V_{\text{READ}} = 0$ V. From a graphical point of view, performing a verify operation with $V_{\text{READ}} = 0$ V and $V_{\text{PASS}} = 5$ V is equivalent to shift the I_E characteristic until it crosses axis $V_{\text{THR}} = V_{\text{READ}} = 0$ 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 $V_{\text{READ}} = 0$ V and $V_{\text{PASS}} = 0$ 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 $V_{\text{READ}} = 0$ V and $V_{\text{PASS}} = 0$ V is the same as verifying it with $V_{\text{PASS}} = 5$ 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:

$$I'_{\text{READTH}} = \frac{\Delta V \cdot C_{\text{BL}}}{T'_{\text{EVAL}}} > I_{\text{READTH}} = \frac{\Delta V \cdot C_{\text{BL}}}{T_{\text{EVAL}}} \quad (9.1)$$

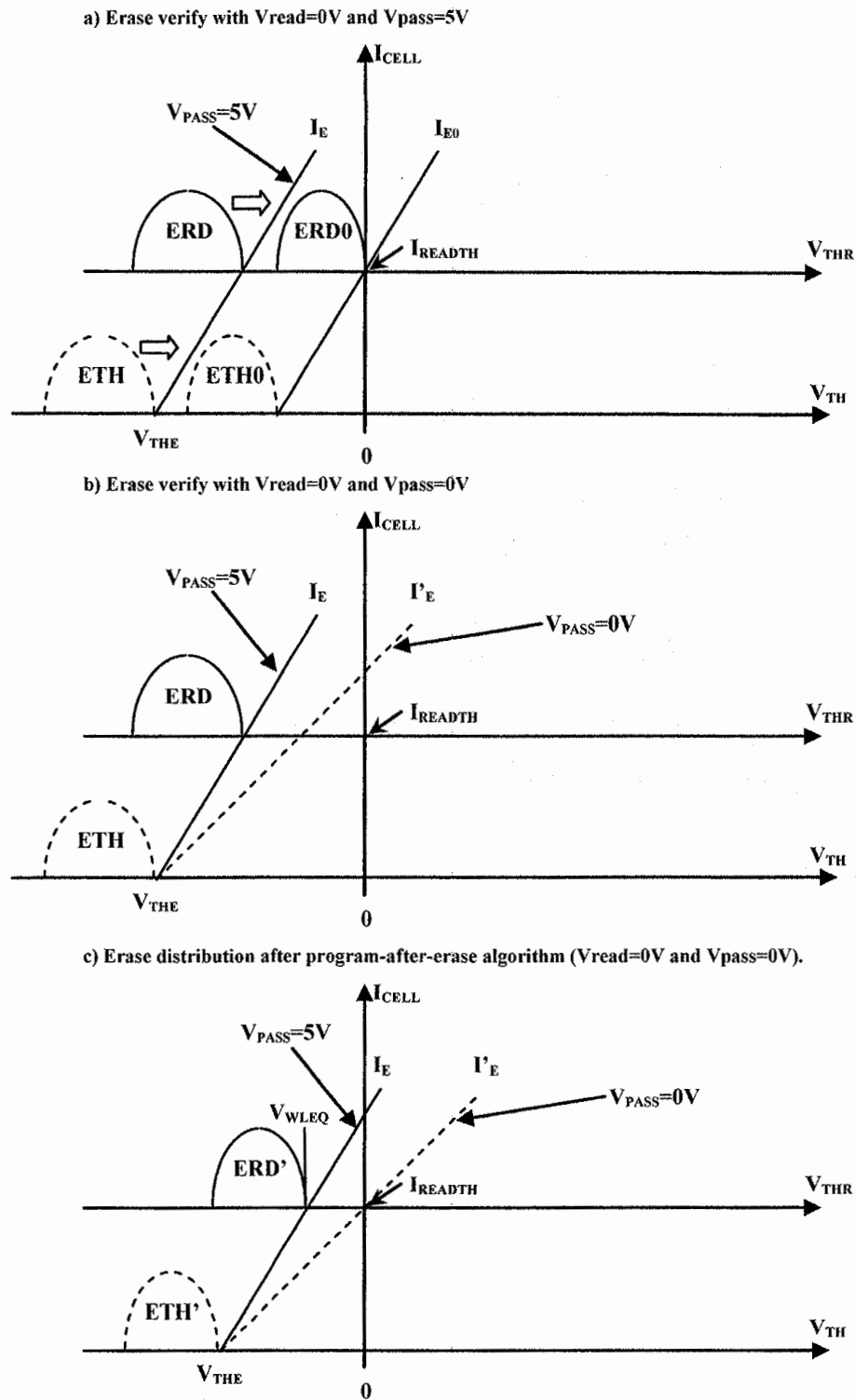
Figure 9.8 shows the equivalence of a read operation with a reduced evaluation time with respect to a standard read ($V_{\text{READ}} = 0$ V and $V_{\text{PASS}} = 5$ V) performed with a negative wordline biasing at V_{WLEQ} .

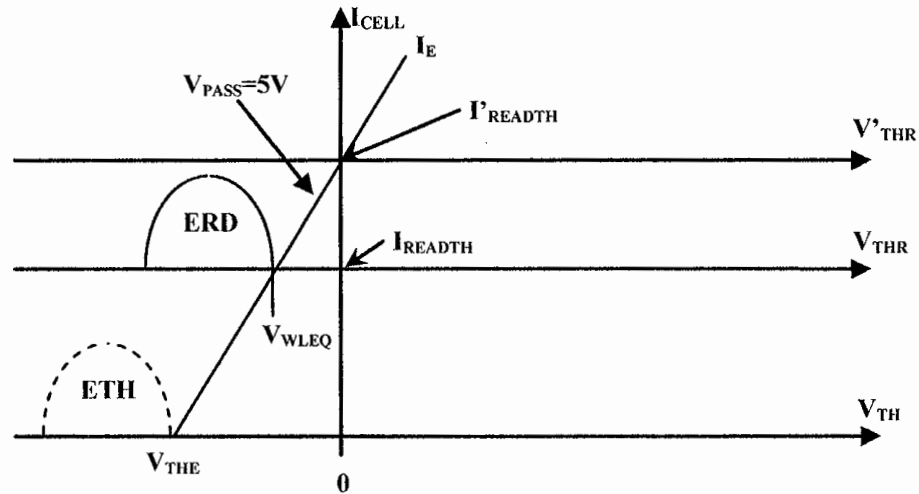
Both methods can be implemented using the reading architectures described in Chap. 8. It is also possible to implement both methods at the same time ($V_{\text{PASS}} = 0$ V and T'_{EVAL}) in order to lower the equivalent negative voltage (V_{WLEQ}) further.

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.

Fig. 9.7. Negative sensing with $V_{\text{READ}} = 0\text{ V}$

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.

$$I = K \cdot (V_{GD} - V_{THN})^2 = K \cdot (V_G - V_D - V_{THN})^2 \quad (9.2)$$

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).

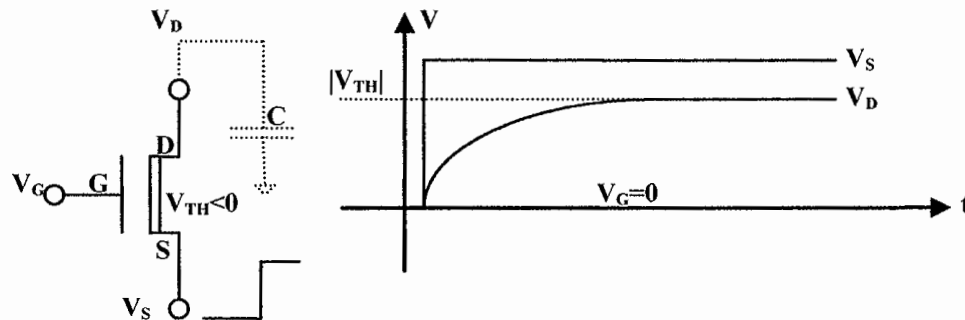


Fig. 9.9. Negative threshold voltage NMOS under positive source biasing

Furthermore, under the assumption of biasing the NMOS using a gate voltage equal to zero, the transistor operates in saturation region if

$$V_S \geq V_{TH} \quad (9.3)$$

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:

$$I = 0 \quad (9.4)$$

Substituting Eq. (9.4) in Eq. (9.2) with $V_G = 0$ V, it follows

$$K \cdot (-V_D - V_{THN})^2 = 0 \Rightarrow V_D = -V_{THN} \quad (9.5)$$

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).

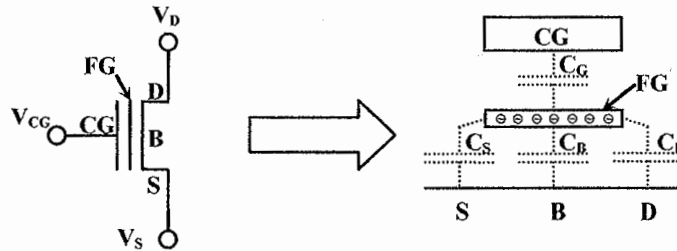


Fig. 9.10. Floating gate NMOS coupling capacitors

Current I is now equal to:

$$I = K \cdot (V_{FG} - V_S - V_{THFG})^2 \quad (9.6)$$

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 :

$$V_{FG} = \alpha_G \cdot V_G + \alpha_B \cdot V_B + \alpha_S \cdot V_S + \alpha_D \cdot V_D + \frac{Q_{FG}}{C_{TOT}} \quad (9.7)$$

where

$$C_{TOT} = C_G + C_B + C_S + C_D \quad (9.8)$$

$$\alpha_G = \frac{C_G}{C_{TOT}}; \alpha_B = \frac{C_B}{C_{TOT}}; \alpha_S = \frac{C_S}{C_{TOT}}; \alpha_D = \frac{C_D}{C_{TOT}} \quad (9.9)$$

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:

$$I = K \cdot (-V_S - V_{THFG} + \frac{Q_{FG}}{C_{TOT}}) = K \cdot (-V_S - V'_{THFG}) \quad (9.10)$$

where V'_{THFG} is the physical threshold voltage seen by the floating gate, including the contribution of Q_{FG}

$$V'_{THFG} = V_{THFG} - \frac{Q_{FG}}{C_{TOT}} \quad (9.11)$$

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:

$$(-V_D - V'_{THFG}) = 0 \Rightarrow V_D = -V'_{THFG} \quad (9.12)$$

When the contributions due to the capacitive coupling with source and drain are considered, Eq. (9.12) should be re-written as follows:

$$V_D = V(D,S) - V'_{THFG} \quad (9.13)$$

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} :

$$V_{THCG} = \frac{V'_{THFG}}{\alpha_G} \quad (9.14)$$

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.

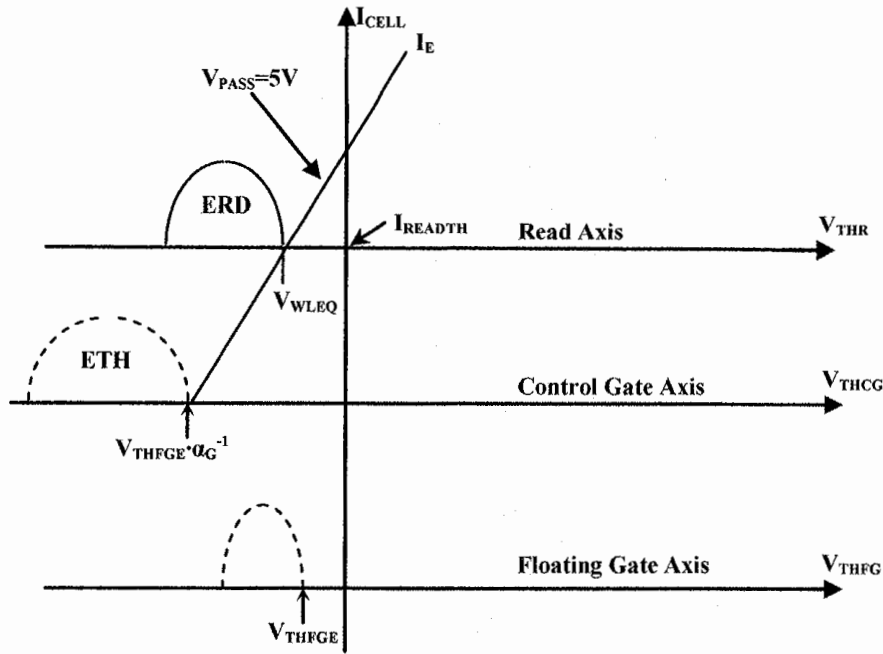


Fig. 9.11. Effect of absolute negative sensing on the different axis

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

$$|V_{THFG}| \leq V_{THS} = |V_{THFGE}| \quad (9.15)$$

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.

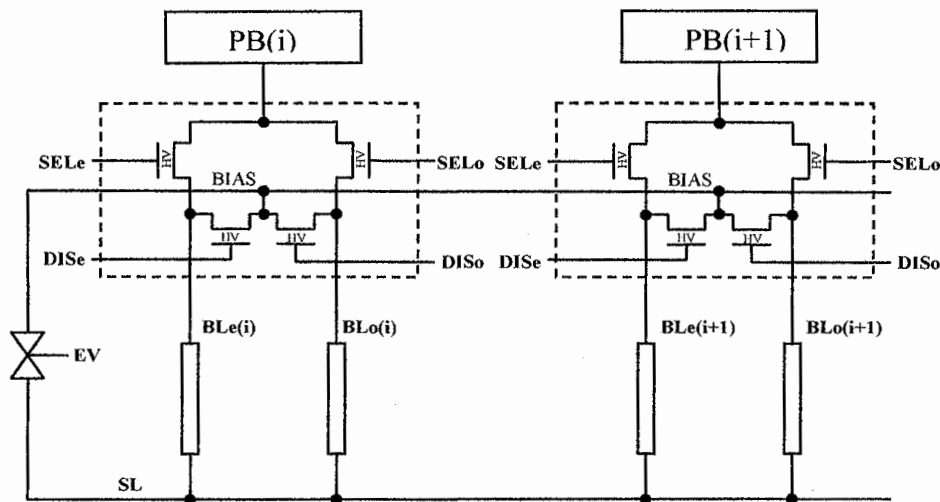


Fig. 9.13. Interleaving bitline architecture for negative sensing

Experimental results show that, once V_{THFG} is set to -1 V, the corresponding value of V_{WLEQ} is -3 V.

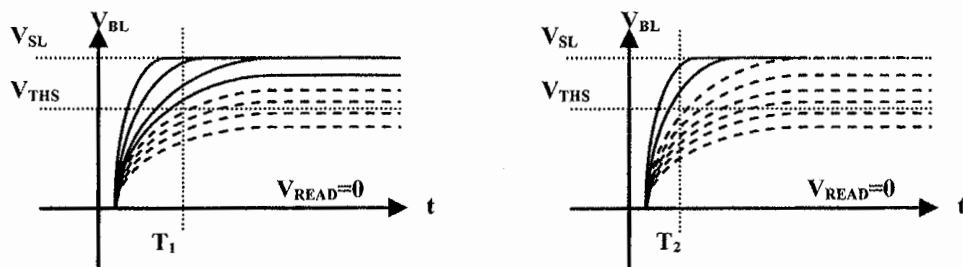


Fig. 9.14. Absolute negative threshold sensing with variable sensing time

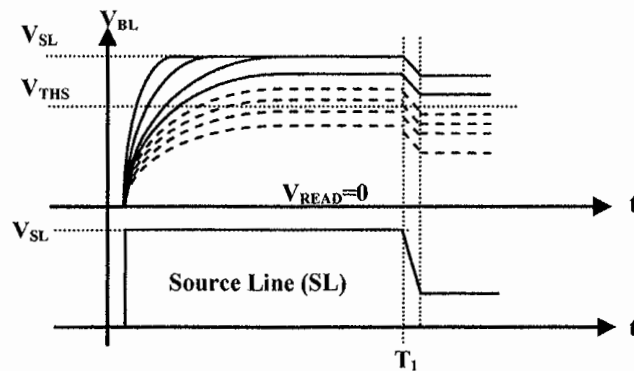


Fig. 9.15. Absolute negative threshold sensing after SL-BL boosting

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:

$$I_D = K \cdot (V_G - V_S - V_{TH})^2 \quad (9.16)$$

This is true even in the case of a MOS whose threshold voltage V_{TH} is negative. If the following condition holds true

$$V_S = (V_G - V_{TH}) \quad (9.17)$$

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 .

$$V_D > V_S \quad (9.18)$$

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.

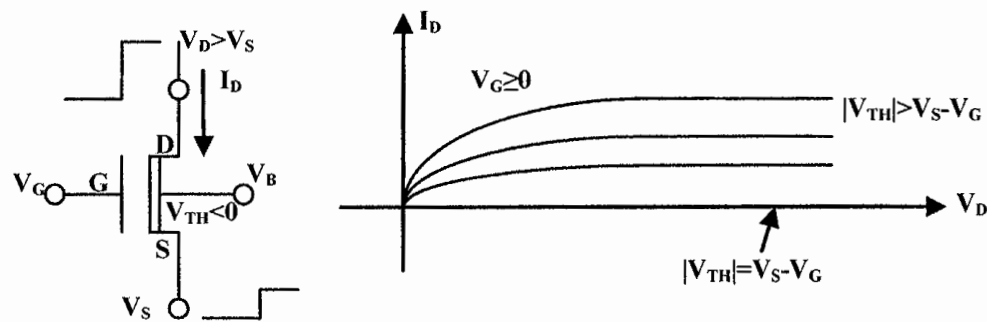


Fig. 9.16. Negative threshold NMOS biasing for current sensing

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.

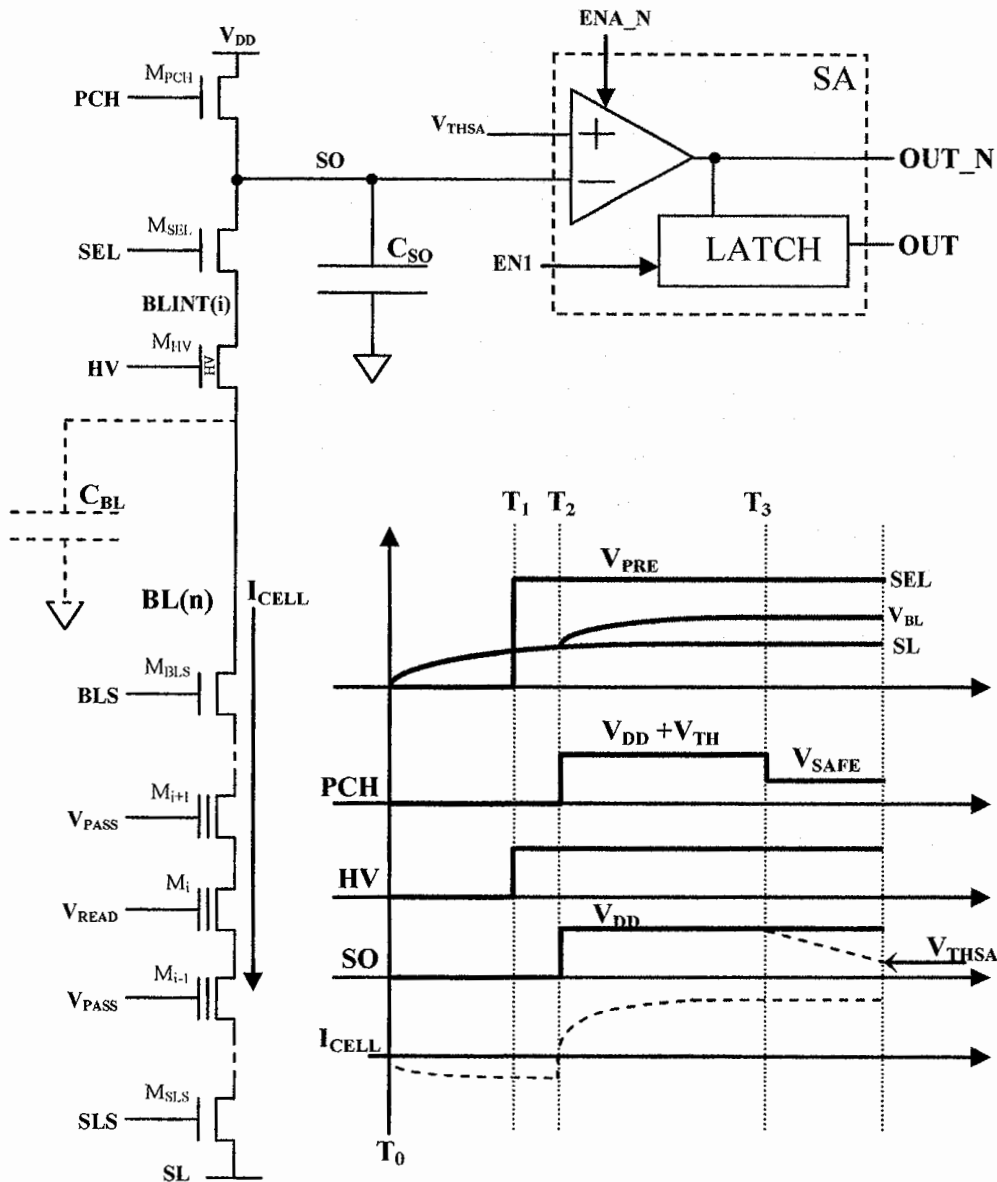


Fig. 9.17. ABL sensing for negative threshold voltages

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.

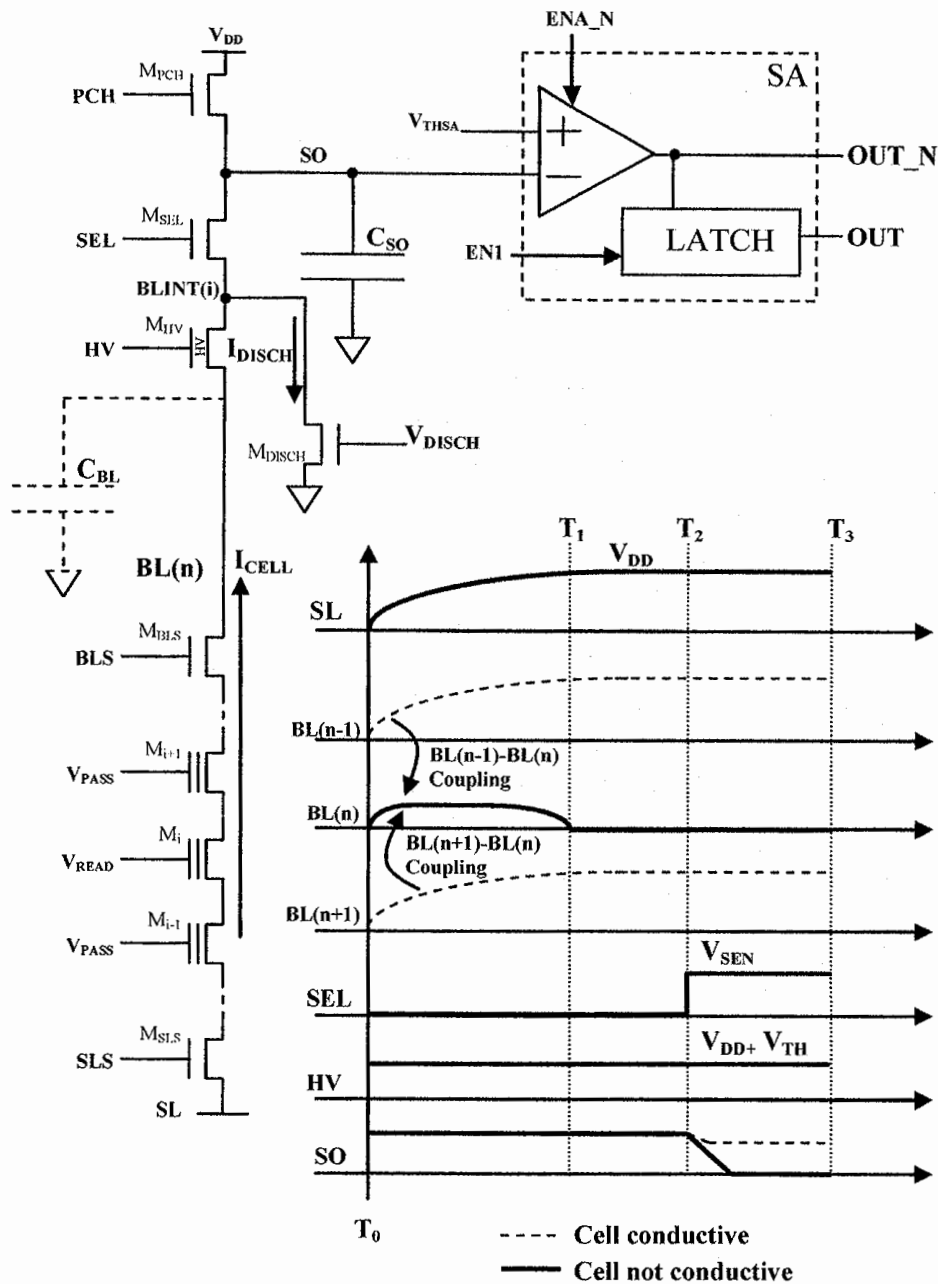


Fig. 9.18. ABL sensing for negative threshold: alternative solution

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

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 .

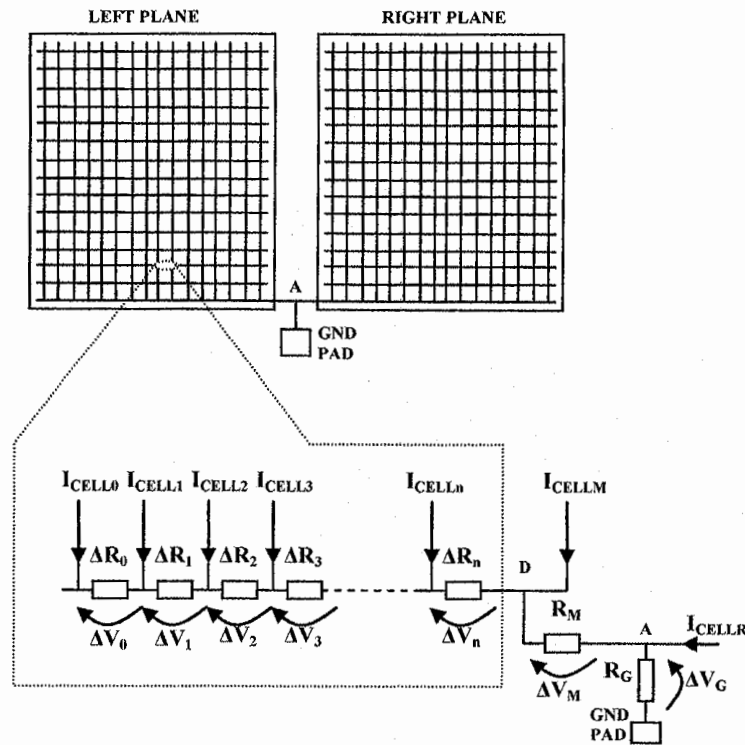


Fig. 9.19. SL routing and parasitic resistance contributes

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_{OS} 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.

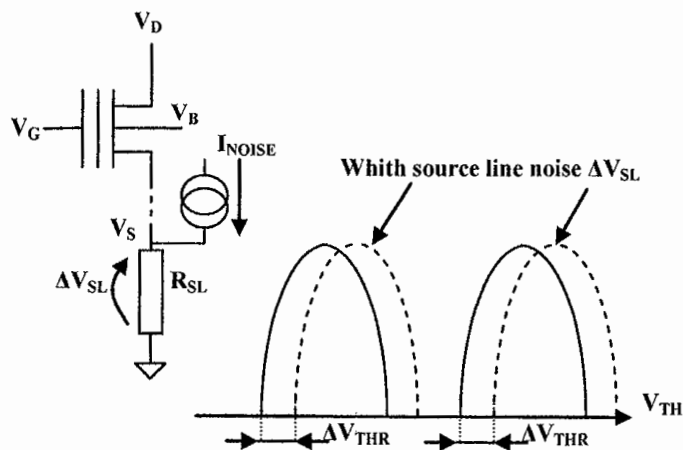


Fig. 9.20. SL noise and voltage distribution shift

The shift of V_{THR} apparently has a 1:1 ratio with the voltage increase of the source line, that is:

$$\Delta V_{THR} = \Delta V_{SL} \quad (9.19)$$

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):

$$\Delta V_{SHIFT} = K \cdot \Delta V_{SL} \quad (9.20)$$

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.

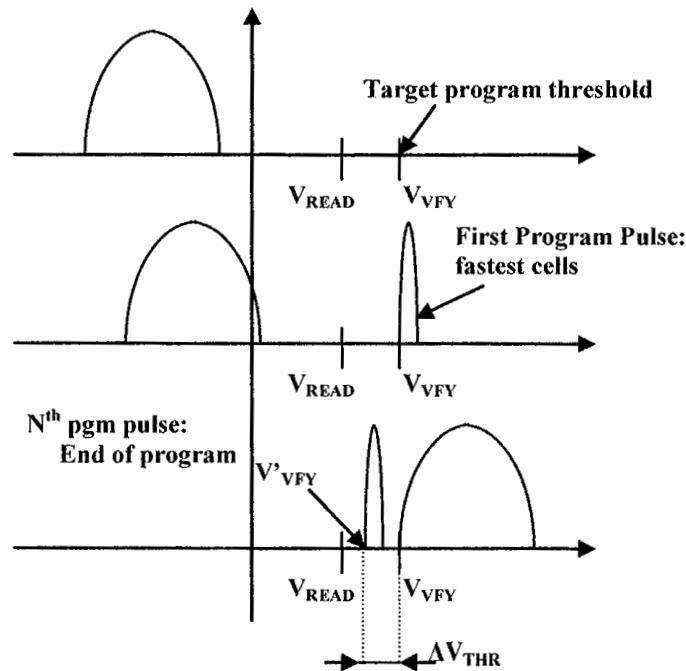


Fig. 9.21. SL noise: distribution enlargement and read margin reduction

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

$$V'_{VFY} = V_{VFY} - \Delta V_{THR} \quad (9.21)$$

and the read margin is no longer

$$\Delta V_{RM} = V_{VFY} - V_{READ} \quad (9.22)$$

but

$$\Delta V_{RM} = V_{VFY} - V_{READ} - \Delta V_{THR} \quad (9.23)$$

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.

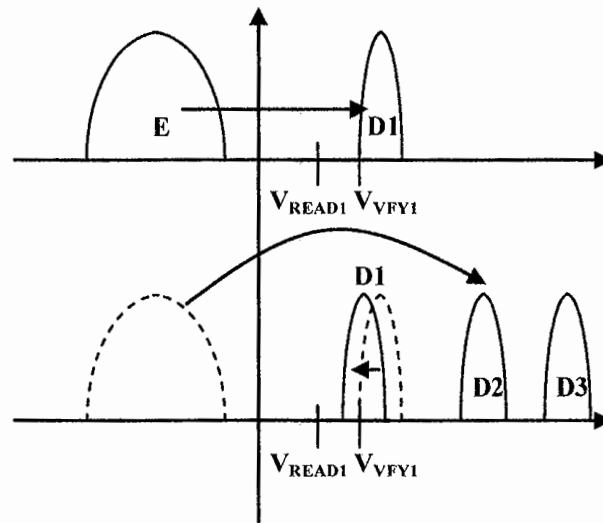


Fig. 9.22. SL noise: distribution enlargement and read margin reduction for MLC

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].

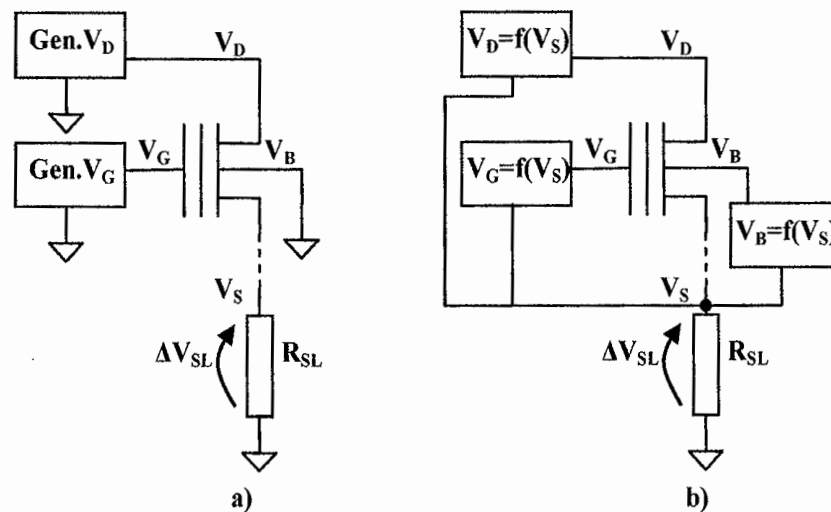


Fig. 9.23. SL noise compensation

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:

$$V_G = V_{REF} \left(1 + \frac{R_2}{R_1} \right) + V_{SL} \quad (9.24)$$

where V_{REF} is generated by the BGREF circuit (band-gap reference circuit).

Therefore, cell's V_{GS} is equal to:

$$V_{GS} = V_G - V_{SL} = V_{REF} \left(1 + \frac{R_2}{R_1} \right) \quad (9.25)$$

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 .

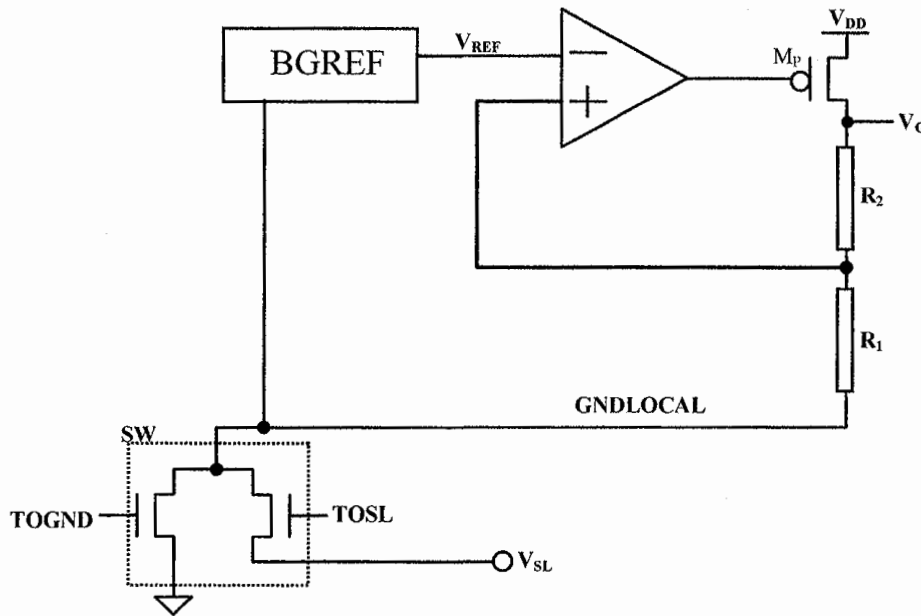


Fig. 9.24. SL noise compensation: wordline voltage generator

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:

$$I_{READTH} = \frac{\Delta V \cdot C_{BL}}{T_{EVAL}} \quad (9.26)$$

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.

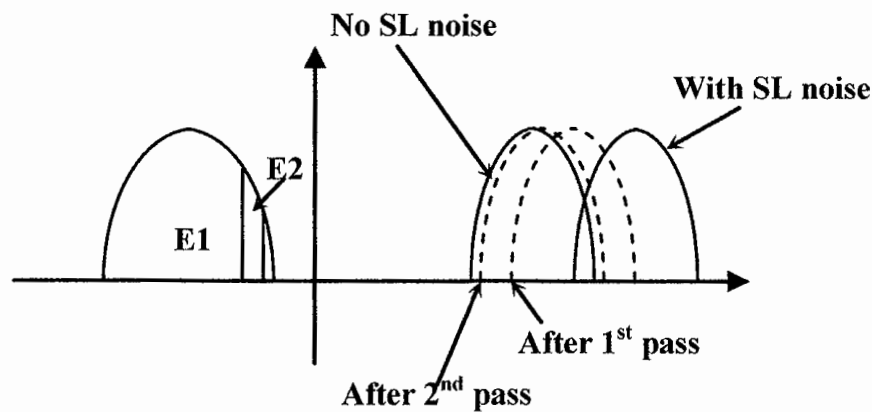


Fig. 9.25. Two pass sensing effect

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).