Historical Overview of Solid-State Non-Volatile Memories

Giovanni Campardo

There are 10 different kinds of people.
Those who understand binary, and those who don't.
(From the web).

1 The Story

I can say that my career can be superimposed, for a long time, on the story of non-volatile memories. I am not so old that I witnessed the birth of non-volatile memories, but, when I started to work on integrated circuits, the memory-cell dimension was in the micron range: 1.5 micron (μm) as the minimum length and the size of the device was 512 Kb.[1] The contact size was 1.5 μm by 1.5 μum. Today the minimum size is in the range of a nanometer (nm) and the size of the device is approaching a terabit (Tb).

At that time, there were some fixed points that defined a 'good' device:

- Chip size must be around 50 mm.[2]
- Access time, in read mode, must be around 100 ns.
- All the bits must be 'good', i.e., no failure bits are permissible in the parts.
- Current consumption must not be greater than 50 mA.

We will see how these 'fixed points' changed during the years, driven by the applications.

The first commercial non-volatile memory with a solid-state technology, was an EEPROM, the Electrical, Erasable, Programmable Read Only Memory, at the end of the '70s, but, at that time, I already was in college. I was first involved in the EPROM development, Fig. 1.

[1]Memory-hardware designers always mean bit, speaking about memory size. So, for me, a memory of 1 M means one megabit. Software people always speak about Byte!.

G. Campardo (✉)
ADG Automotive Digital Division - Product Design, STM, Agrate, Italy
e-mail: giovanni.campardo@st.com

R. Gastaldi and G. Campardo (eds.), *In Search of the Next Memory*,
DOI 10.1007/978-3-319-47724-4_2

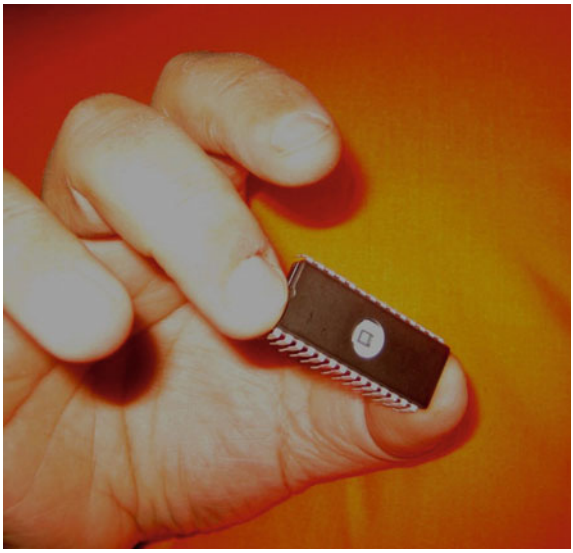

Fig. 1 EPROM device with the silicon window that enabled erasure by UV

The need to have a device able to maintain data stored inside it, also when the voltage is removed, was a fundamental requirement, as well as simple ways to store and retrieve the data and programs.

In the 1980s, there were no MP3 music players, no digital photo cameras, and neither any digital televisions, cellular phones, tablets, portable computers… so, why did we need nonvolatile memories in the 80s?

All electronics machines, like desktop computers or industrial machines like the automatic coffee maker, the dishwasher, the ice cream machine, etc., need a program to work. Perhaps hard to believe, but, in the 1980s, the washing machine, for example, had a camshaft, like a carillon. A metallic cylinder with variously geared teeth, the cylinder rotated slowly, and the teeth opened and closed various electro-mechanical relays.

Of course, the need to reduce the size, decrease the power consumption, and increase the 'intelligence' of the executive program imply having the electronics on board and, most importantly, having storage to encode the program to be executed.

In an EPROM, you can write the bytes individually but the erasing is done by removing the chip from the board and putting it under a UV light for at least 20 min. After that, you must put the EPROM into the socket of a 'dedicated machine' that is able to write the memory code and verify it. Finally, one reinserts the EPROM into the devices' application board with a new program inside.

Of course this is not acceptable today! Can you imagine doing the same now for all your your devices, e.g., for your cellular phone? Each time you want store a new phone number, you have to turn your phone off, open the case, remove the memory, employ a 'dedicated machine' where your address book is updated, and then put the memory back into the case, etc.

If you want to save an SMS, you have to do the same operation each time that you decide to write a new message. It would not be practical.

EPROM has a glass window in the package to permit erasing the device. The package is made of ceramic and not plastic to increase robustness, but the drawback is the added cost, due to the presence of the window. The mechanism to program is called a Channel Hot Electron (CHE), and the erasing was done by tunneling, produced by UV radiation.

The circuitry for an EPROM device (Fig. 2) can be rendered into these fundamental blocks: the decoders for rows and columns; the sense amplifier; the circuitry to read the status of the cell; the circuitry for the redundancy; and the registers reset by the EPROM cell called UPROM, used to store the fail addresses; at every read request, the circuitry compares the address presented to the redundancy registers. If

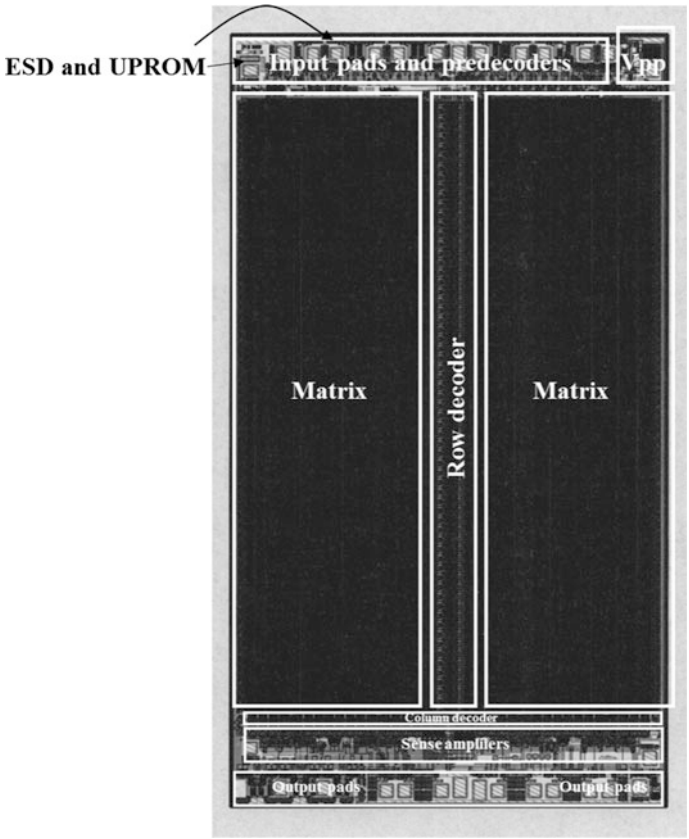

Fig. 2 The array, divided into two semi-arrays, is depicted, with the row decoder in between. The column decoder is located above the sense amplifiers to read and the output pads. In the top part are the input pads and predecoders for both rows and columns. Finally, the VPP pad, ESD, and UPROM are positioned as shown

the address is recognized as one of the failed, the read is redirected to the redundant cells, otherwise 'normal' cells will be activated. In the device depicted in Fig. 2, the UPROM (Unerasable Programmable Read Only Memory) was inserted into the input buffers, so the comparison was done very early in the access time of the redundant parts, which was not so different from the good one. The UPROM structure was really very clever; he who invented this, must have had a moment's glance at 'the God notebook'. The UPROM was performed by the EPROM cell but, the problem is that these cells must not be erased when the user decides to erase the device by UV light. So, this cell was covered by metal and the last layer was to connect the transistor but to avoid the possibility that the light could reach the cells, like a waveguide. To this end, the gate connection was designed not like a straight line but like a wavy line.

Other circuits present were the ATD generator, the Power On, the I/O buffer with the ESD circuitry, and control signals PAD, such as the CE# and the OE#.

Figure 3 shows the biasing voltage needed to program (or write) an EPROM cell, on the left. On the right is a photo of the real physical structure.

The ATD, Address Transition Detector (Fig. 4), was what we can call a 'pseudo clock'; EPROM had no clock but, as we can understand, a synchronization circuits is needed to scan the sequential operations, so the ATD observes the change of address and also the control signal. When a change occurs, a pulse was generated, and this pulse was used as a 'start' to drive the event sequence of the read chain blocks.

The Power On circuit, many times called POR (Power On Reset), is a very simple circuit able 'to detect' the supply voltage ramp, and, during it, to generate a pulse to set the necessary circuits into a known state, or, more simply stated, a hardware reset.

The ESD circuitry is used to prevent possible overvoltage or under, beyond the limits to avoid the possibility of destroying the circuitry. ESD circuit will turn on before the discharge will be able to reach the first transistor gate. These structures are inside the PADs.

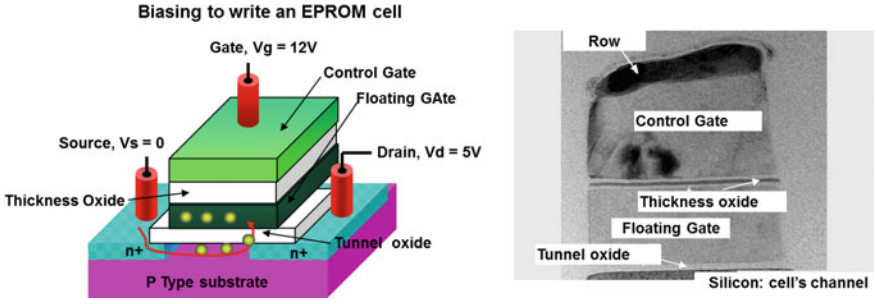

Fig. 3 EPROM-cell write biasing

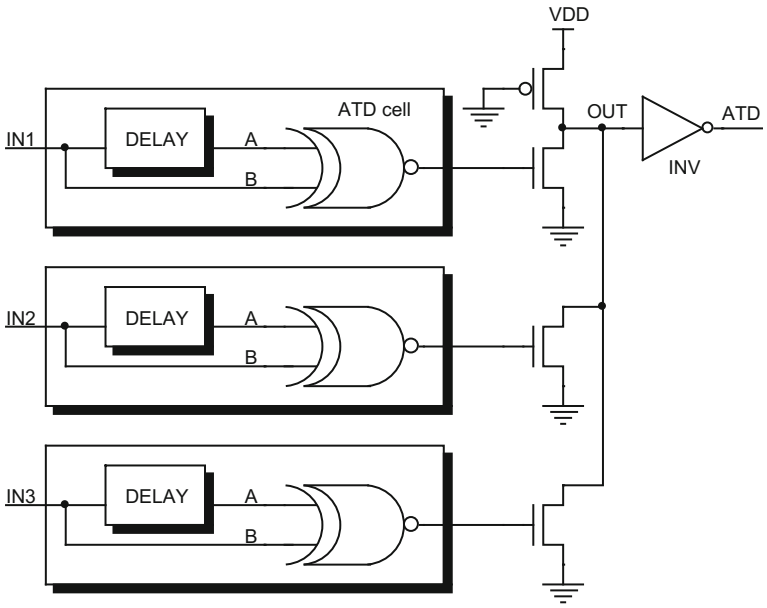

Fig. 4 The block in the box is able to generate a pulse when the IN# changes. The composition of several control blocks enables detection of the transition of the input signals. The various outputs are combined by means of a distributed NOR

The control signal is indicated by CE#, the Chip Enable#, and active low (# is the symbol to indicate that the signal is active when it is low, normal ground, 0 v). This convention, to have the active state at low-voltage value of the input, was due to the NMOS circuitry. Low input means: no consumption, the NMOS transistor is off, and a sure, full value exists of the output of the first inverter. This signal turns on the device while the OE#, Output Enable# removes the output buffer from the three state situation (or high impedance state) where the Byte value was already loaded.

So, this was really 'poor' circuitry as compared with the circuitry we have today in a non-volatile memory. In fact, in the '80s, the process was NMOS with no PCH transistor available! Simple process, only NMOS transistors, means complex circuit, if you want to have good performances.

Bootstrap techniques[2] are adopted throughout, otherwise the voltage node could easily reach the ground value. To reach the VDD, NMOS limit, i.e., the capability to reach the VDD due to the positive threshold, you need to set the gate of the generic NMOS a step above the VDD.

[2]For me the word bootstrap didn't have a clear meaning. I knew what the bootstrap is, but when I eventually understood the real meaning of this, and I started to imagine someone trying to lift up himself pulling on the boot straps, I really came to understand the meaning of this in electronics circuits.

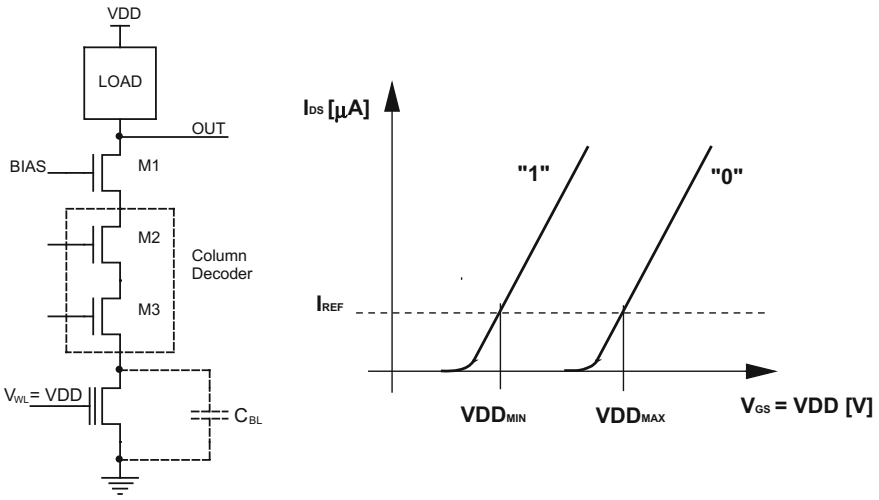

Fig. 5 The first sense amplifier made a simple comparison with the fixed current represented by the LOAD. Logic "1" and "0" differs from a threshold shift, obtained by a program. The VDD space was the voltage on the gate row. The space in between VDD_{MIN} and VDD_{MAX} defined the VDD voltage range to be read. In that space, a cell with a current greater of I_{REF} is recognized as a "1", and with a lower current as a "0"

The sense amplifier to read was, for the initial devices, to a simple 'inverter stage' (Fig. 5), written cell sink '0' current, virgin cell sink current.

The read mode circuit was like an inverter.

In the latter half of the '80s, devices started to have the first differential sense amplifiers that used a reference cell (Fig. 6).

After a while, a great revolution that has lasted 30 years began. The era of the mobile would eventually arrive, and nothing could stop it.

I remember, like it was yesterday, when the boss of the R&D (he had to be one of the contributors to this book) entered our open workspace, where the memory group was sitting, with a sample in his hands. He was very, very happy, and he said 'they erase…..'; he was actually saying that they are trying to erase an EPROM device, a test device, using an electric pulse, not like an EEPROM but to erasing the entire matrix. One of my colleagues, a test engineers, answered: "…we call these rejects…!".

I will never be able to forget the face and the look of the R&D boss, a very perturbed manager!

In any case, a story that I told to the student during some seminars at the university recounted how Flash memory was invented to make it possible to erase, in one shot, all memory content. This request came from a military application. They wanted to have the possibility to destroy the software, or firmware, should a missile or airplane or whatever contains a piece of proprietary software that could fall into the hands of the enemy.

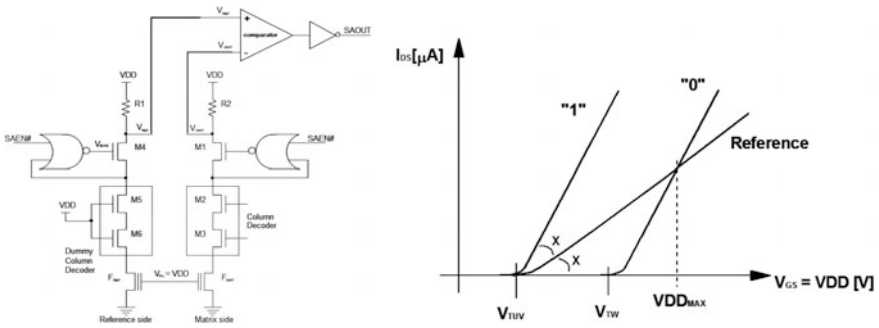

Fig. 6 A differential sense amplifier means having two branches to compare one to the other. This speeds up the read phase. In the first sense amplifier approach, the main issue was to enlarge the VDD working window. So, using a cell to trace the process and temperature variability, a sense must be able to have a reference characteristic in between the "1" and the "0". This was obtained with unbalanced loads R1 and R2

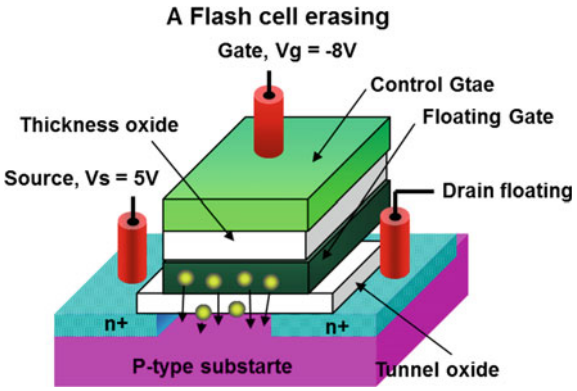

Fig. 7 Flash cell erase biasing

Whether or not the story is true, the possibility to modify the content of the memory without removing it from the application board was the first requirement. And so, why not use EEPROM? EEPROM was and is a non-volatile memory, programmable and erasable by electrical application, Fig. 7. With this, there is no need to remove it from the board.

The price to be paid was the size of the cell. Each cell, or each byte, could be addressed independently and, using high voltage, an electric-tunnel erase is achieved. While this was a great advance in term of flexibility but without the ability to have mass storage memory, only a few kilobytes are possible.

Now having the capability to modify the memory content using external commands, the market that seems to govern our lives (is it true?) Started to ask for more and more.

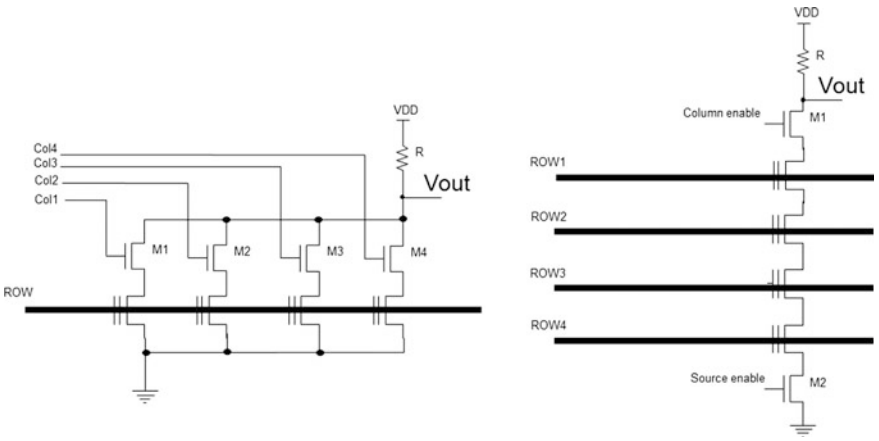

Fig. 8 Let me skip forward for a moment. On the *left side*, we have the NOR organization. Each cell connected to the same ROW will be activated. If the correspondent column decoder transistor will be ON, the node Vout will be driven to the relative voltage. NOR organization means: it is enough to have only one element turned on to produce the output. On the *right side*, the NAND organization: to drive the output, all of the cell must be activated. The different values of the voltage ROW# will define the cell that must be read and the cells that must be turned on to be a pass. NAND organization means: all the elements must be turned on

As a first step, when the memory devices were still EPROM and not yet Flash, the process, also for the memory, became CMOS. The memory designers, at least, could exploit a lot of new possibilities but also there were a lot of new things to learn.

Firstly Flash, NOR architecture, is not an acronym, like for EPROM or EEPROM; Flash means 'in a flash', like a lighting. NOR is based on the matrix organization, see Fig. 8.

The first Flash devices had a structure similar to the EPROM: very simple circuitry, no internal algorithms machine, programmed with CHE, and erasure by tunneling.

Now came the era of the ASM, Application Specific Memory or memory design for a special application. For example, a digital filter, an FIR, constructed with an EPROM, used like a look-up table to speed up the calculus and was used in the first digital television, Fig. 9a; and an ASIC, with a macrocell Flash, was used for automotive application, Fig. 9b.

Another ASM was an I/O port expander for automotive application, composed of a Flash and a SRAM memory on the same die, see Fig. 10. Each time the car was turned off, the parameters written in the SRAM were stored in the Flash, ready for the next ignition. At that time, for many different types of applications, a software program was introduced to recover the depleted cells.

We can say that the Flash story is the story of the erase mode and the way to recover the problematic information caused by the erase. For an EPROM, the distribution of the virgin cell is tight. The light erasing is able to set the floating gate

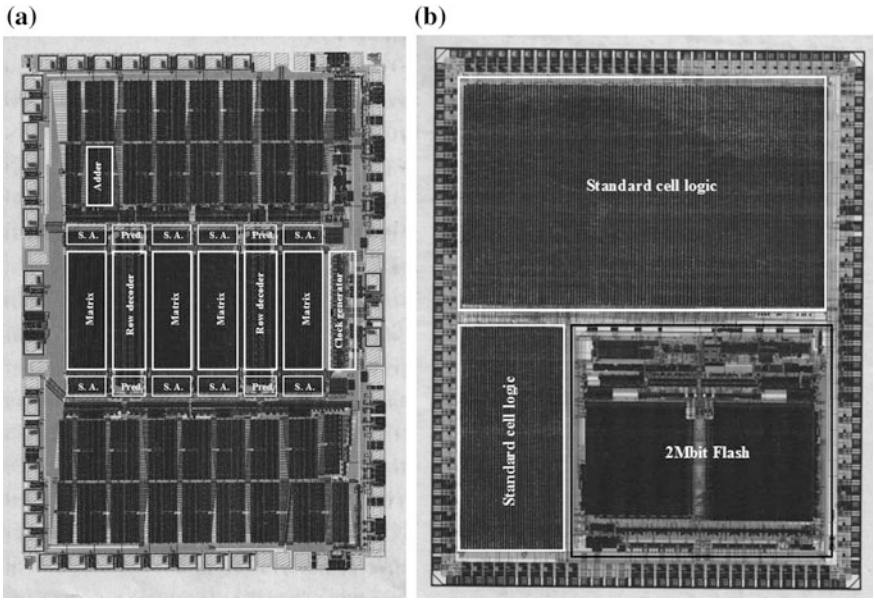

Fig. 9 **a** A FIR filter using an EPROM memory, divided into four semi-arrays to speed up read. The content of the memory is a look-up table whole outputs are added by a fast adder to produce the digital processing that realized the required filter. In this device, the column decoder was not present to speed up read. **b** ASM memory chip. 2 Mbit Flash memory inside a pad limited device with logic, designed with the standard cell technique. Note the regularity of the logic cells compared with the part of memory realized by designing custom transistors

to the original situation of charge neutrality, so that the distribution of the erased cells is only dependent on the structural differences between the cells. For a Flash, the erasing is a collective action done by electrical pulses. The results is a large distribution of the threshold voltage, Fig. 11.

Another of the points that distinguish EPROM from Flash is the reference cell and also the matrix organization that is specifically related.

In an EPROM device, the matrix is in a continuum with the reference cell, inserted in the matrix as a reference column for each output.

This solution is feasible because the erasing is done by light and results in a tight distribution. Since there is a cell for each output, if the matrix is organized with eight outputs, we will have eight reference cells for each row. This structure will be repeated for each column. This structure has the highest similarity to the cells matrix in terms of process and temperature variation, leaving the best solution for the differential read mode. This solution, for a Flash, is not more likely, because, if the reference cells were inserted inside the matrix when the erase is performed, the reference cells will also be erased and the distribution is also be larger in respect to the UV distribution. It will be possible to reserve 'special' blocks for the reference cells, but a lot of space will be consumed. The final solution was to remove the

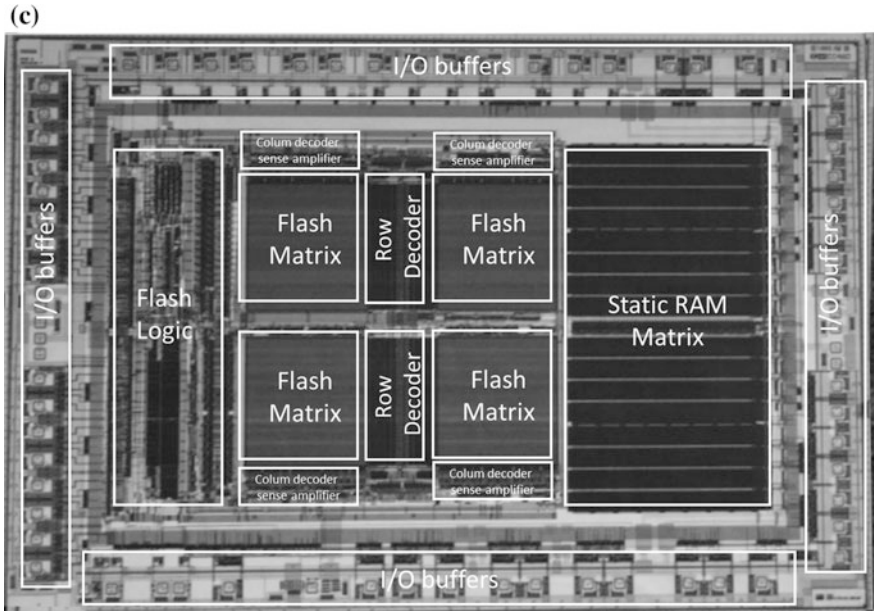

Fig. 10 An I/O Port Expander done with a 1Megabit Flash, plus a 16Kbit SRAM, to store the program and data for the automotive application

reference cells from the matrix and create another little matrix, outside the matrix space, while inside all the reference cells remain that one might need.

The second point was the fact that the EPROM was a 'single' matrix, a single monolithic block because it is not possible to 'hide' from the light a portion of the matrix and to unveil it when we want.

Anyway, the first Flash device did not have a matrix divided into sectors. The matrix was a single block, too, like an EPROM, but with the possibility of erasure by electrical signal. This was a great achievement, having the possibility to erase by electrical signals without also remove memory from the application board.

For all these reasons, the design of a Flash-memory array is more difficult compared to the design of an EPROM-memory array.

For the EPROM, the matrix structure is designed focusing on the access time, which placed constraints on the row and column length. This justifies splitting the matrix into different arrays with shorter rows and columns, by multiplying row and column decoders, called sectors.

At that time, there occurred no great circuitry revolution. The erasing was achieved by driving all the rows to 0 V, all the bit lines floating, and the source connected to a high voltage, typically 12 V.

The only further change to be accomplished was due to the fact that, in an EPROM, the row decoder that addresses logic always guaranteed a row at read voltage; this was one of the problems for the reliability of the gate stress generated;

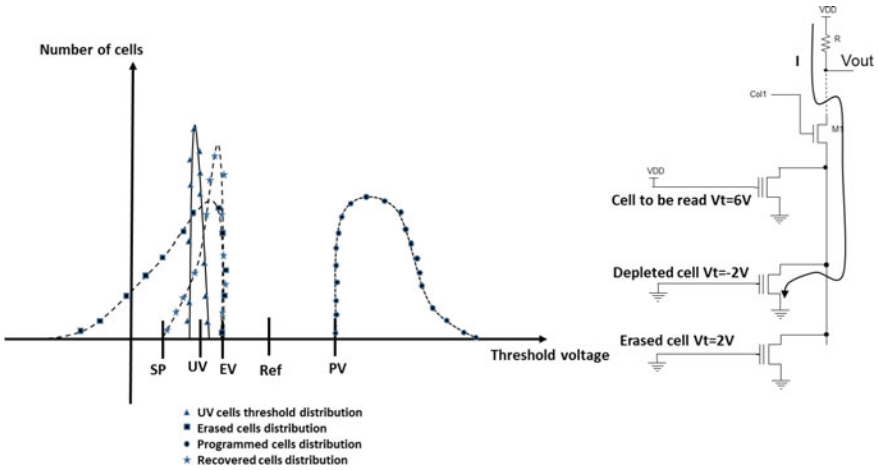

Fig. 11 The electrical erase produces a large distribution. Starting from a written cells distribution, all the cells have a threshold voltage greater than a defined value, called Program Verify (PV). The erase is done by an electrical pulse to remove electrons from the floating gate, so that the erased cells will have a threshold less than the Erase-Verify voltage (EV). The final distribution could also have cells with a threshold voltage less than than 0. These cells, called depleted cells, can produce a false read. The scheme on the right represents a bit line. We would like to read the first cell, a programmed cell with a 6 V of threshold. No current must flow from the load to the ground, but the cell with a threshold of −2 V sinks current also if it is not addressed. The results, for the sense amplifier, will be the same as if a logic '1' is present. Soft Program algorithm is able to detect a cell with a threshold less than SP voltage, and, with well-defined pulses, to recover these cells with a well-controlled program. It is fundamental not to move the other cells, already erased. The final erased distribution will be between the SP and EV voltages. The Ref value is the reference voltage threshold of the reference cell used to read. The UV distribution is graphed, which has very tight distribution around the UV threshold voltage

in any event, to erase we need to have the gate at 0 V. So a new signal was introduced to force all the gates to ground, Fig. 12.

Figure 13 shows two possible sector architectures. The first, on the left, shows different i.e., a portion of the matrix with different sources nodes to be erased separately, with a common row. On the right, are again different sectors with different sources, but the row is not common to the sectors, and the columns are inserted separately with a local column decoder between them. This solution can avoid any stress on a sector not involved during the modification, program, and erase modes.

The main problem was the spurious consumption during the erase. The erase is performed by a tunneling effect, so the current used is really very low, but the high voltage applied implies a spurious current, the Band-to-Band current, which is an order of magnitude greater than the tunneling current. The market pushes forcefully to design mobile devices that have the characteristics of low-power consumption and long battery duration, so that means a low supply voltage.

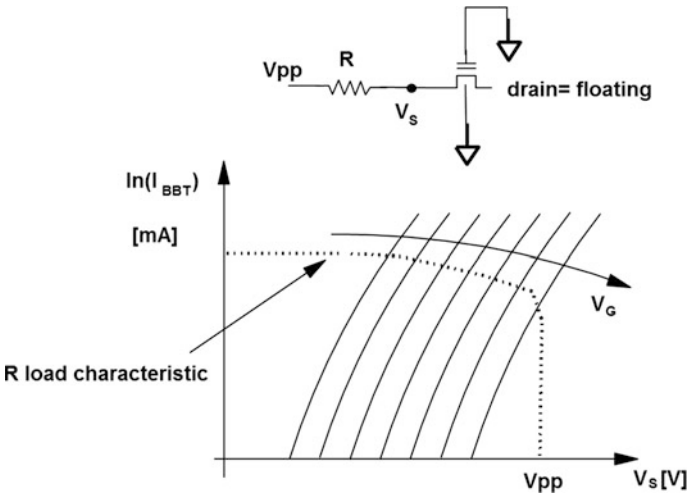

Fig. 12 The first electrical erase was performed by a high voltage, typically 12 V, on the source, which is the common node of the sector. To decrease the spurious effect, the Band-to-Band (BBT) current, a resistor, R, is connected between the external voltage Vpp and the source matrix to exercise a negative feedback and limit the spurious current. The graph represents the BBT current for various gate and source voltages V_G, V_S. As we can see, the R characteristic ranges from a high current to 0 as the erase proceeds

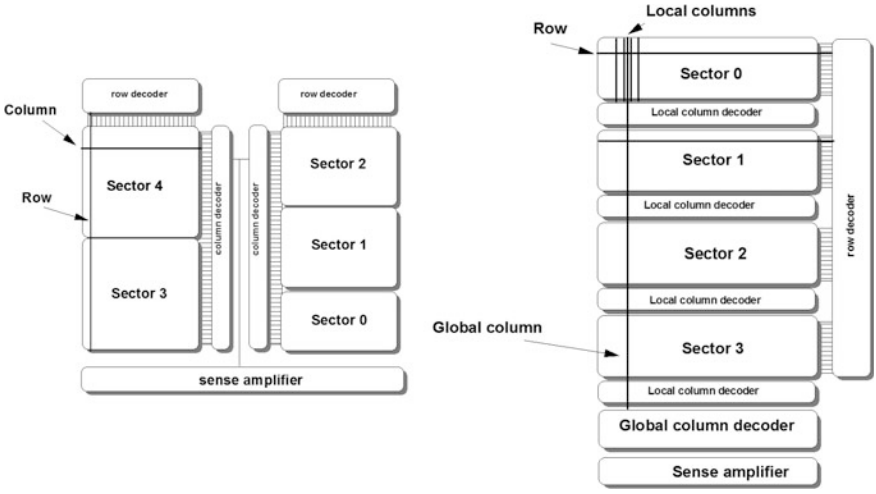

Fig. 13 Two examples of sectorization matrix array

It was not possible any more to have a dual-voltage device with a core voltage equal to 5 V and a modify voltage equal to 12 V, exactly like the EPROM.

The solution was to decrease the supply voltage and remove the 12 V from the board. Thus, the single supply-voltage devices was created, 5 V for all.

But, how it is possible to erase having only an external 5 V supply?

This approach is feasible because it is true that the power consumption, per cell, was around, during Program, 1 mA but we need high voltage on the row, typically 10V with no power consumption because it is an RC charge while, for the drain, 5V is good enough to program. The same for read, but, for the erase pump circuit, diode and capacitance typology could be designed although the output resistance of these circuits cannot reach the value we need to sustain 12 V at a few mA of current consumption. To do this, we would need larger capacitances that consume too much space.

Technology helped to find the solution. While erasing needs to have an electric field between the control gate and the source, this could be reached not only by putting all the voltage on the source, but also by splitting the voltage between the gate and the source nodes.

The solution was to put a negative voltage on the gate and a positive voltage on the source, typically −10 V on the control gate and 5 V on the source, to achieve the electric field needed for the tunneling. Always driven by the mobile-phone requirement, sectorization explored other solutions. Of course, the sector, the part of the matrix erasable without interference with the other sectors, must have the highest granularity as possible, but this conflicts with the die area, the real parameter to be verified to produce a salable device. Another point was the problem related to the time needed to erase. Typical time-to-program is in the order of microseconds (μs), while to erase the order of magnitude is second (s). Of course, while the memory is replacing its content, with a programming or an erasing or both, the microcontroller in the system will be stopped, waiting for the memory-activity conclusion; this could be not acceptable.

The ‘read-while-write’ and ‘read-while-erase’ concepts were born to solve the problem. This was one of the most important concepts for the cellular mobile application during the first years after the introduction of cellular phoned.

Let’s come back now to the new erase voltage distribution with the negative voltage applied to the gate.

The principal issue was the incompatibility of the negative voltage with the NMOS transistors.

With the final row driver as a CMOS inverter, the PCH transistor will switch the positive voltage on the row and the NCH transistor on the ground. A ‘normal’ NMOS cannot switch a negative voltage because it has the substrate biased to ground, and the reverse diode, between the source (and drain) and the substrate, cannot sustain more than a threshold voltage before starting to conduct current.

So, the first solution was not to change the process, but to invent a circuital solution to overcome the problem, Fig. 14, which depicts the solutions.

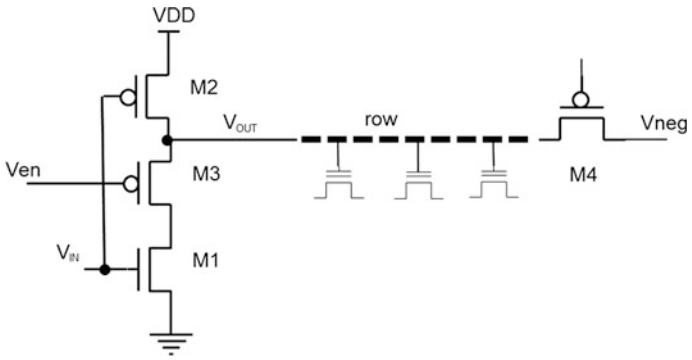

Fig. 14 The circuital solution to solve the problem related to the impossibility of putting a negative voltage on the NMOS channel terminals

The negative voltage will be switched on the matrix row from a PMOS put on the other end of the row, in respect to the row-decoder position. The row contacts never see a NMOS transistor but always PMOS. In this way, no direct diode current will be activated.

This solution was a great one, but not so economical. The row-decoder volume will greatly increase, and the necessity to use only PMOS transistors to manage negative voltage implies also other limitations. In the single voltage device, we need to generate internally the voltages greater or smaller than the supply voltage, using the pump circuits.

To generate negative voltage, as for the row decoder, we cannot manage negative voltage, so the negative voltage pump must be composed of only a PMOS transistor. This choice limits the pump efficiency because PMOS charge, the holes, are heavier and slower with respect to the NMOS charge, the electrons.

The next step was the introduction of the triple well, Fig. 15.

This process step enables having an isolated p-type substrate, with a buried layer and well on the lateral side. The NMOS transistor diffused inside this 'isolated' well could bias its relative ip-substrate and avoid any diode direct current.

Also for the read operation, a low-voltage supply introduced many diverse solutions. For the early EPROM. the voltage put on the matrix row, the cell control gate, was the VDD. With the low-voltage solution down to 5 V, and then to 1.8–1.6 V, it was mandatory to generate internally the voltage to read, using again a pump circuit.

A simple comment readily becomes clear: the Flash story correspond to the erase story, and this it is true, of course, that Flash was born to enable the electrical erasing and thus the requirements of the mobile market.

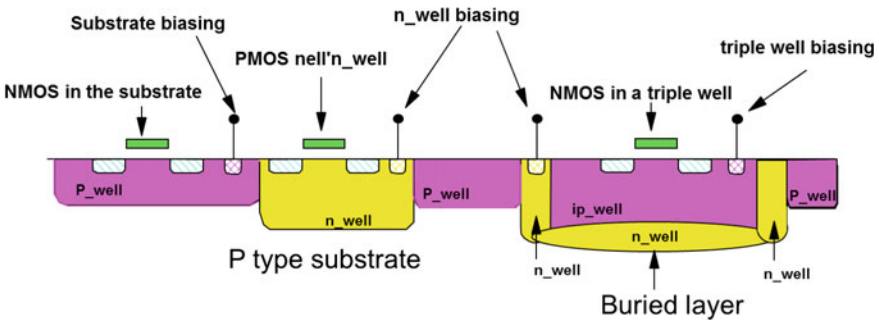

Fig. 15 The triple well technology

The technology has followed the well-known Moore law,[3] and the cell shrink path, too. Over 30 years, the minimum-size dimension has evolved from micron to nanometer.

Over those 30 years, a nonvolatile memory-chip size, in term of cells number, evolved from 64 Kb to 128 Gb, a factor of 2,000,000 greater.

This change reduced the cell size, but also the number of electrons stored in the floating gate. The first EPROM had, in the floating gate, around 50,000 electrons (this is an estimate based on the capacitance calculation and the voltage related), while today we have a few tens of electrons.

This, with the complex technological steps and the large number of cells in each device, has resulted in a very high possibility to have fail cells in the array. Moreover, with the usage, the cycle of program and erase also generate aged cells that must be recovered or substituted for during normal life of device applications.

That was the time of the Error Correction Code (ECC), and let me make a few comments about this concept.

During our normal, everyday life, we use many devices that increase the quality of life in this society, i.e., the information society. Our cellular phones, with a memory capacity and a working frequency comparable to a computer, are used not only to make calls, but also to listen to music and the radio, to watch movies, and to send and receive other messages. Moreover, we play electronic games, we maintains a calendar, an agenda, and an address and phone book, etc.

At every turn, we are exploiting the laws of quantum mechanics. Every time we record an address or a cellular phone number or we save an SMS, we use the tunnel effect to erase and write data (depending on the type of nonvolatile memory we are using), and, if we want to find a street using our GPS, we are using relativistic

[3]In 1965, just a couple of years after Jack Kilby patented the bipolar transistor, Intel's co-founder Gordon Moore observed that the number of transistor per square inch in an integrated circuit doubled every year. Moore foresaw that this trend would be valid for the years to come: indeed, the doubling of the density of the active components in an integrated circuit proved to occur every 18 months, and this is the commonly accepted value for the definition of Moore's law today.

mechanics to optimize the information. All these operations require use a large data exchange, and this exchange must work without error.

I tried to explain to my fried what the ECC is: it is exactly when a wife send her husband to the market to shop. While first she explains clearly what you have to buy, she always will give you a shopping list in a written form. In more synthetic form, this is redundant information. Although I always fail to successfully complete the errand, a transmission cannot fail. Redundant information, associated with the initial one, serves to verify the accuracy of the transmission. ECC needs more cells inside the matrix, increasing the die area.

Then came a really fundamental change, the multilevel introduction, Fig. 16. The possibility to be able 'to count' the electrons displaced, during the program, in the floating gate.

Of course, we are not able literally to count, but we are able to discriminate the value of the cell current and, knowing that this current is related to its threshold, to detect 'where' is the threshold of the cell. In Fig. 17, we see on the left the distribution for a single-level device. The thresholds are divided in two distributions, the "1" and the "0" logic. A reference put in the middle is able to understand cell thresholds and to compare the reference current and the matrix cell current.

If we are to be able, through the program action, to program a cell at the threshold level we want, as in the right-hand side of Fig. 17, we can have more than two distributions. For a 2bit/cell, we will have 4 distributions with logic values "11", "10", "01", and "00". Three different references can enable comparisons between cell current and the references that define the distributions to which the cells belongs.

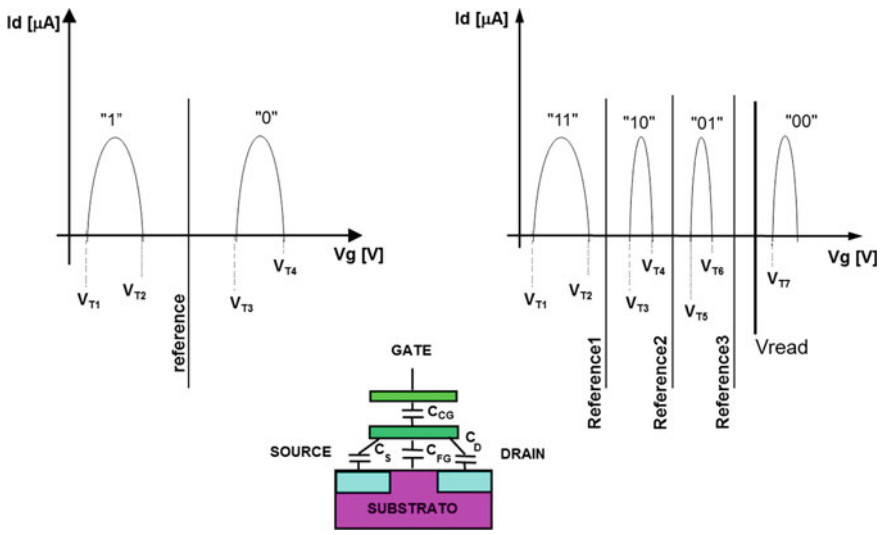

Fig. 16 The threshold distribution (*left-hand side* for a single level and *right-hand* for a 2bit/cell)

The shrinking dimensions of the technology continues, but the market requirements are ever more aggressive. Our times require tons of memory to record all the moments of our lives. The next trick to satisfy the customers was the use of the z-dimension.

A way called 'more than Moore' enables the placement, in the same package, of more dice, one piled on top of another, as shown in Fig. 17. Today, the stack up can reach up to 16 levels.

In Fig. 17, on the left is a wafer that was ground with a mixed process—chemical and mechanical—to obtain a 70-μm thickness starting with 800 μm for the original wafer. With this thickness, the wafer became elastic. The dice grinded are attached in a stack and wired to a substrate that works like a board to connect the wire bonding to the external connections, composed of metallic balls.

All USB pen drives, the Solid-State Disk (SSD), and memory used in the automotive dashboard, or inside a cellular phone, are done with this technology.[4]

While the customers want an increased complexity of the system, the management of the complexity must be handled within the system and not inflicted on the user, of course.

The next generation was the Managed-Memory generation. We already noted that the technology shrink implies an increase of defects, and how the solution was the introduction of ECC. But the user wants system compatibility. When you connect your external device to the PC, it is recognize as a peripheral and, on your screen, you will see a folder, like for all the other files you have on your internal hard disk.

Moreover we also need to build all the algorithms needed to extend the lifetime of the device, like the Bad Block Management, which is a way to discover if a block is beginning to fail and then to substitute some redundant block for it, without

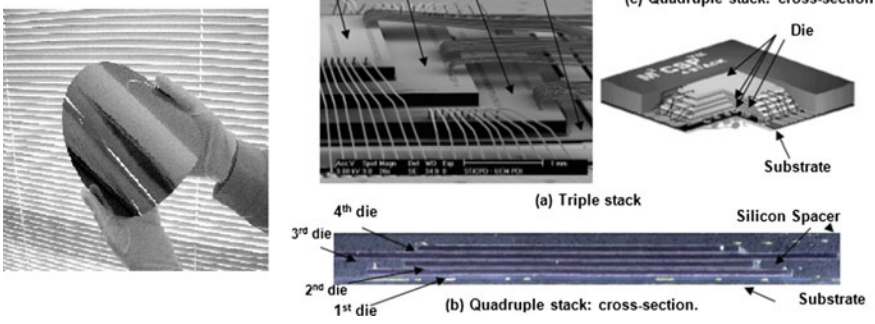

Fig. 17 Stacked technology

[4]My first PC, a few years ago, after the Z80 and some other machines, was a 16 MHz clock PC with a 16 MB hard disk. Today, in a cellular phone, the nonvolatile memory is measured in GB.

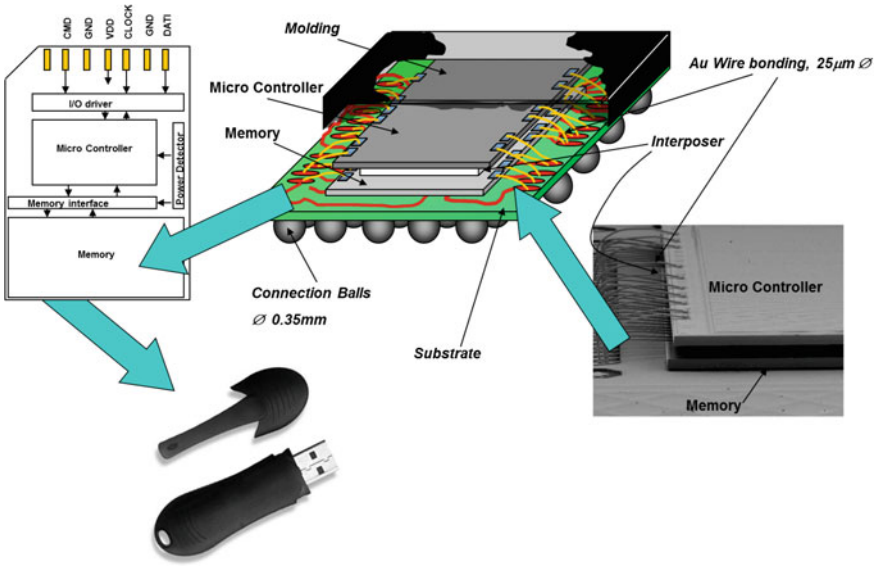

Fig. 18 A managed memory, a CARD

any loss of information. To do this, a microcontroller is embedded inside the stack, to relieve users from the having to control the memory, Fig. 18.

Up until the last ten years, the floating-gate memory was Flash NOR, but memory Flash NAND then started to become the leader. The compactness of the matrix layout, but above all the choice to use the tunneling effect also for the program,[5] provided the possibility to write to many cells as compared to the NOR solution that uses CHE. This means low-current operation, but also the possibility to increase the communication speed between the memory and system.

NAND is intrinsically slower than NOR, especially in read. We can state an order of magnitude, nanosecond for the NOR and microsecond for the NAND but, again, for a power-consumption reason, NOR reads 256–512 cells at the same time and NAND reaches 16,000.

So, NAND enhanced the performance of the system and became the real mass-storage memory.

The size of the NAND die became greater over the years, now more than 1 cm^2.

Coming back to the initial assumptions:

- Chip size must be around 50 mm^2
- Memory device is today greater than 1 cm^2
- Access time, in read mode, must be around 100 ns

[5]This was not a 'choice' but, due to the array organization in a NAND structure, it is not possible to apply a current to program with the CHE effect.

- Memory device access time is in the range of microseconds. The system 'solves' the problems
  - All the bits must be 'good', no failure bits are permitted inside the parts
- Memory devices are sold with a maximum value of the failed blocks permitted
  - Current consumption must be not greater than 50 mA.
- System power consumption could be very high, especially for the interface working at high frequency.

And now? People are exploring the z-dimension, memory, NAND memory, with the cells stacked one over the other, to save space and increase memory availability.

What are we likely to see in the next future?

References

1. G. Campardo et al. "*A 40 mm^2 3 V 50 MHz 64 Mb 4-level Cell NOR Type Flash Memory,* IEEE Journal of Solid State Circuits, Vol. 35, N0 11, November 2000.
2. G. Campardo and R. Micheloni *"Architecture of nonvolatile memory with multi-bit cells"* Elsevier Science, Microelectronic Engineering, Volume 59, Issue 1-4, November 2001, pp. 173–181.
3. G. Campardo, R. Micheloni *"Scanning the Issue",* Proceeding of the IEEE, VOL. 91, No. 4, April 2003, Special Issue on the Flash Memories.
4. G. Campardo et al. "*An Overview of Flash Architectural Developments"* Proceeding of the IEEE, April 2003.
5. R. Micheloni et al. *"The Flash Memory Read Path: building blocks and critical aspects"* Proceeding of the IEEE, April 2003.
6. G. Campardo, R. Micheloni, D. Novosel "*VLSI-Design of Non-Volatile Memories*", Springer Series in ADVANCED MICROELECTRONICS, 2005.
7. P. Pulici, G. Campardo, G. P. Vanalli, P. P. Stoppino, T. Lessio, A. Vigilante, A. Losavio, G. Ripamonti "*1 V NOR Flash memory employing inductor merged within package"* Electronics Letters, vol. 43, no. 10, Page(s): 566–567, 2007.
8. G. Campardo, R. Micheloni *"La rivoluzione delle memorie," Le Scienze, italian editon of* SCIENTIFIC AMERICAN, number 468- August 2007, pagg. 104–110.
9. R. Micheloni, G. Campardo, P. Olivo, "*Memories in Wireless Systems*", Springer Verlag, 2008.
10. A. Maurelli, D. Beloit, G. Campardo, "*SoC and SiP, the Yin and Yang of the Tao for the new Electronic Era",* Proceeding of the IEEE, Vol. 97, number 1, January 2009, Special Issue on the 3D Integration Technology.
11. G. Campardo, G. Ripamonti, R. Micheloni *"Scanning the Issue",* Proceeding of the IEEE, Vol. 97, number 1, January 2009, Special Issue on the 3D Integration Technology.
12. G. Campardo, F. Tiziani, M. Iaculo "*Memory Mass Storage*", Springer Verlag, March 2011.

http://www.springer.com/978-3-319-47722-0

In Search of the Next Memory
Inside the Circuitry from the Oldest to the Emerging Non-Volatile Memories
Gastaldi, R.; Campardo, G. (Eds.)
2017, XVIII, 247 p. 185 illus., Hardcover
ISBN: 978-3-319-47722-0