10 Memories

This chapter describes the memory cell architecture. After a general introduction, we detail the principles and implementation of static RAM, dynamic RAM, read-only memories, electrically erasable memories, and Ferro-magnetic memories.

10.1 The world of Memories

Semiconductor memories are vital components in modern integrated circuits. Stand-alone memories represent roughly 30% of the global integrated circuit market. Within system-on-chip, memory circuits usually represent more than 75% of the total number of transistors.

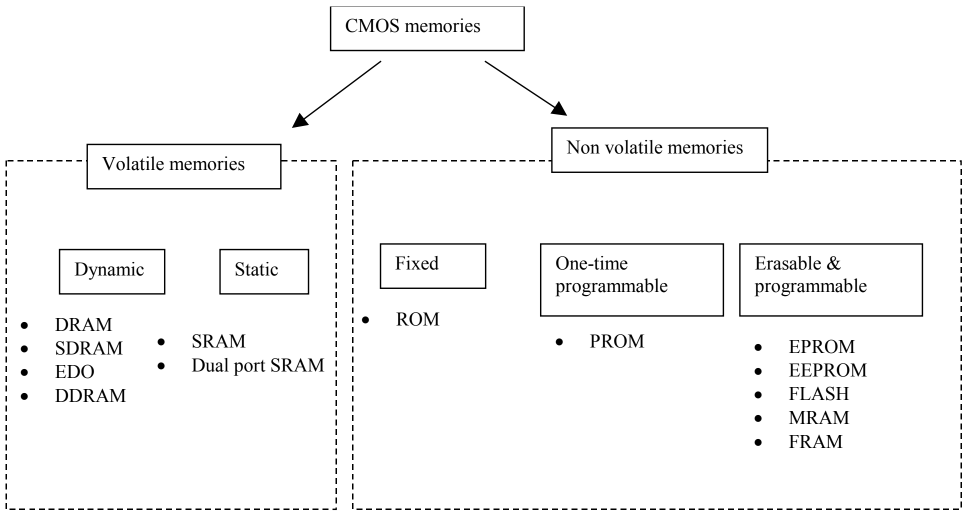


Figure 10-1 Major classes of CMOS compatible memories

Two main families of devices exist: volatile and non-volatile memories.

- In volatile circuits (Figure 10-1 left), the data is stored as long as the power is applied. The dynamic random access memory (DRAM)<gloss> is the most common memory.
- Non-volatile memories are capable of storing the information even if the power is turned off (Figure 10-1 right). The read-only memory (ROM) is the simplest type of non-volatile memory. One-time programmable memories (PROM<gloss>) are a second important family, but the most popular non volatile memories are erasable and programmable devices: the old electrically programmable ROM (EPROM), the more recent

Electrically Erasable PROM (EEPROM, FLASH), and the new magneto resistive RAM (MRAM <Gloss>) and ferroelectric RAM (FRAM<Gloss>) memories.

Millions of elementary memories are used by microprocessors for executing software. Micro-code, operating systems and low level software are usually stored in non-volatile memory. The software execution requires fast-access random access memory such as static or dynamic RAM. Memory exist as stand-alone components (Figure 10-2-a), but also as embedded blocks in system-on-chip, such as those shown in figure 10-2-b.

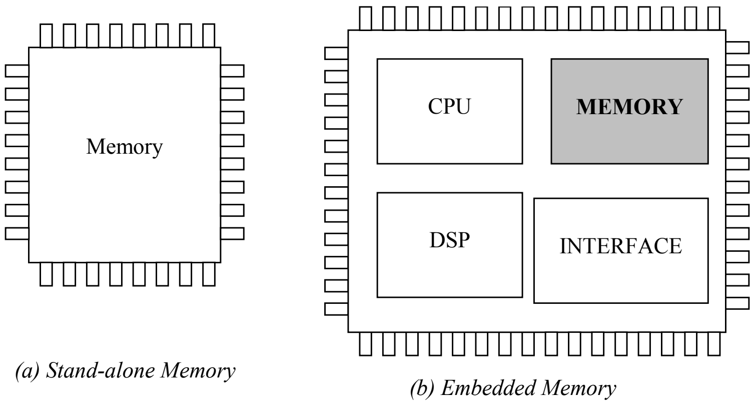


Figure 10-2: The memory exists as a stand-alone component or as embedded block

Memory organization

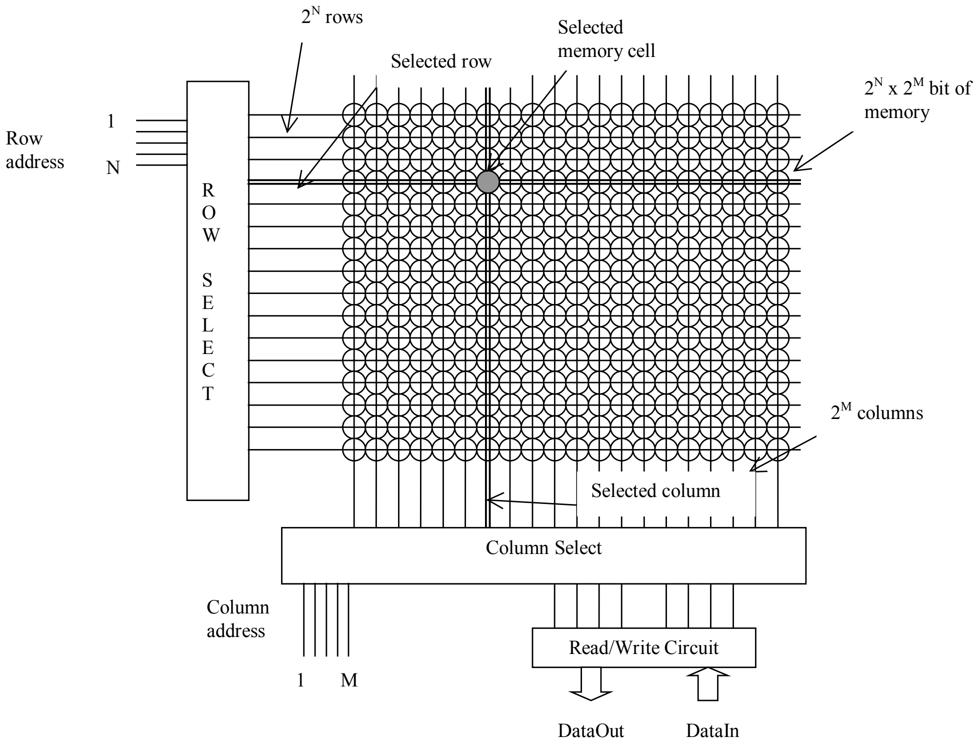


Figure 10-3 Typical memory organization

Figure 10-3 shows a typical memory organization layout. It consists of a memory array, a row decoder, a column decoder and a read/write circuit. The row decoder selects one row from 2^N, thanks to a N-bit row selection address. The column decoder selects one row from 2^M, thanks to a M-bit column selection address. The memory array is based on 2^N rows and 2^M columns of a repeated pattern, the basic memory cell. A typical value for N and M is 10, leading to 1024 rows and 1024 columns, which corresponds to 1048576 elementary memory cells (1Mega-bit). Several organizations exist: 1024x1024 bit, 128Kx8bit, 64Kx16bit, 32Kx32 bits, etc.. For example, the organization 128Kx8bit consists in selecting 8 columns in parallel. In that case, the size of *DataOut* and *DataIn* bus is 8 bit.

Access Time

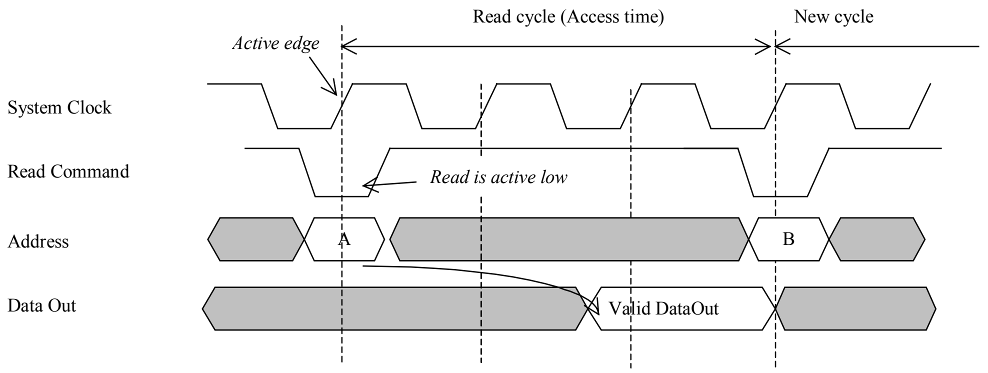


Figure 10-4: Read Access Time

The typical timing diagram of a memory block is shown in figure 10-4. A fast system clock, around 1GHz period, synchronizes the whole sequence. On an active level of the *Read* command (Usually a low level), the read cycle begins. It may take several clock cycles before the data is available. In the case of figure 10-4, two clock cycles are necessary before the valid data is proposed at the *DataOut* bus. The typical access time for Mega-bit memories ranges between 1ns and 10ns.

10.2 Static RAM Memory

Introduction

The static RAM is a very important class of memory. It consists of two cross-coupled inverters, which form a positive feedback with two possible states illustrated in figure 10-5. This cell is also the base of many sequential circuits, as detailed in chapter 8.

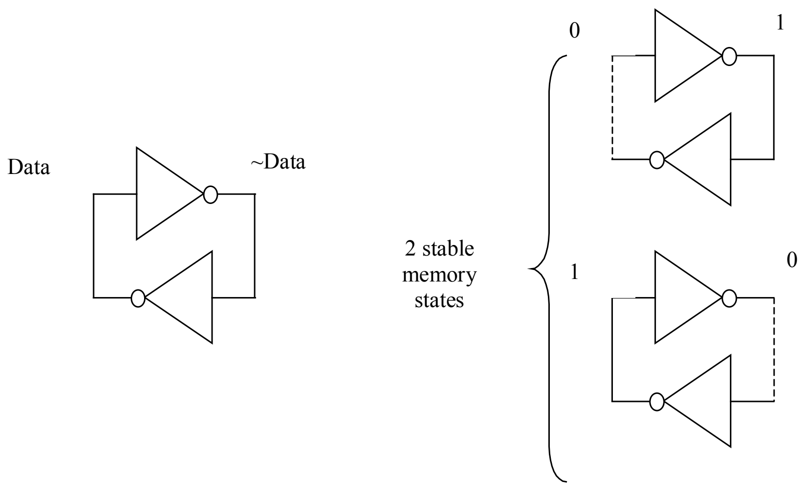


Figure 10-5: Elementary memory cell based on an inverter loop

Trying the Five Transistors Cell

The state of the memory cell can be changed by a pass transistor. In that case, the memory cell has five transistors. The signal that controls the gate of the pass transistor is usually called the *word line*, or *WL*. The data information that flows vertically is called the *Bit Line* (*BL*). The schematic diagram of the static memory array is shown in figure 10-6.

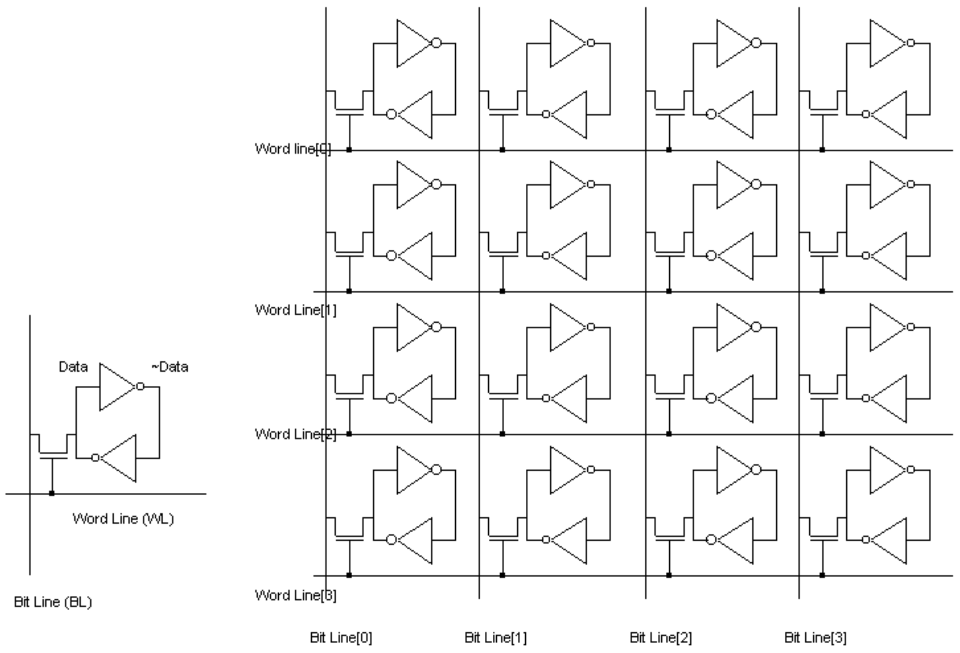


Figure 10-6: Building a memory array with five transistor static memory cell (RAMStatic5T.SCH)

The 5 transistor static RAM layout is given in Figure 10-7. Remember that the RAM cell will be copied hundreds of times in X and Y, that silicon area is money, and that up to 80% of the system-on-chip transistors may be used for memory. Consequently, the RAM layout should be as compact as possible to produce a memory array with the smallest possible silicon area. The steps to build the 5T memory cell are presented in the next figures. We start by designing one nMOS and two compiled inverters (Figure 10-7). The first action is to flip the inverter situated on the left in order to share the supply network (Figure 10-8).

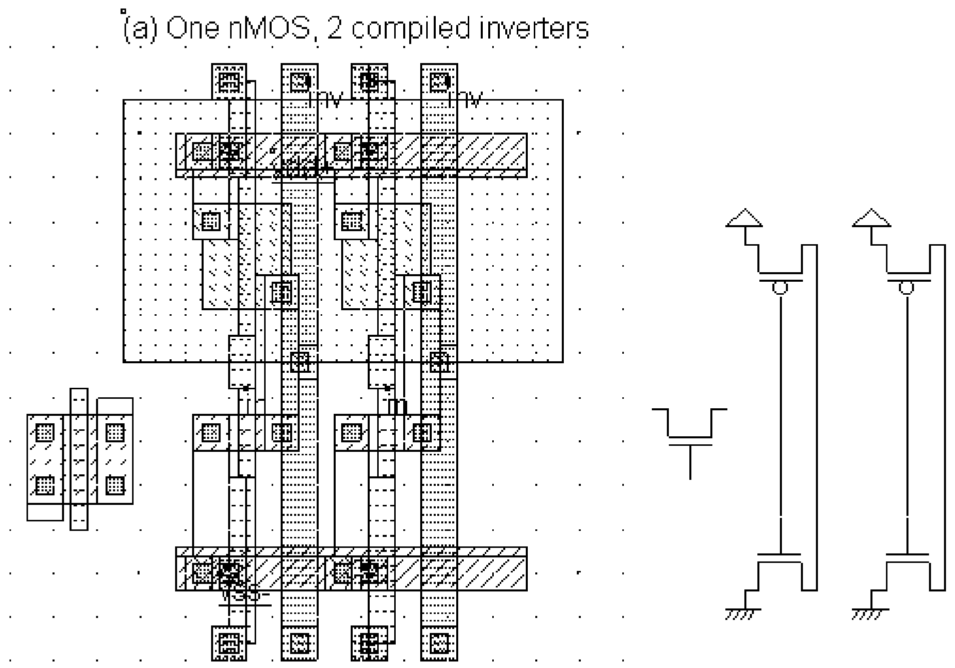


Figure 10-7: Initial steps for the design of the 5T memory cell (RAMStatic5T.MSK)

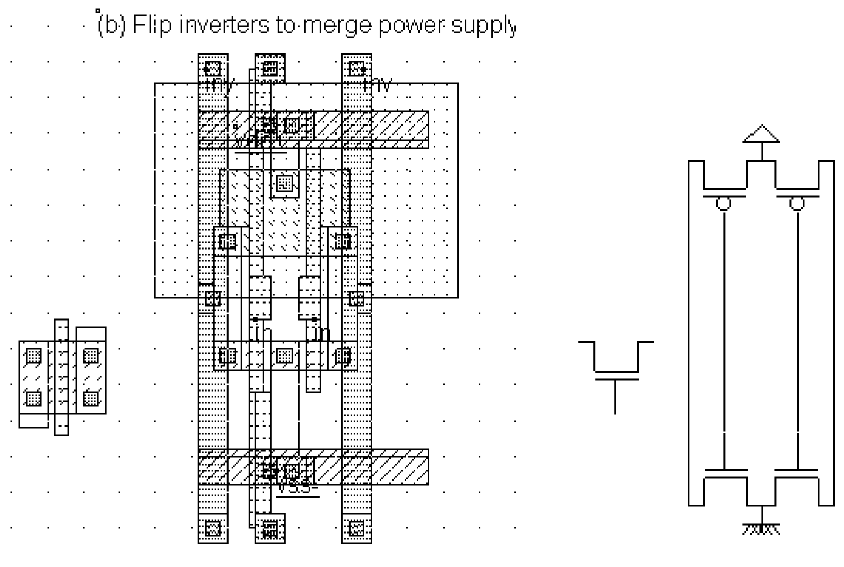


Figure 10-8: Merging the supply contacts to compact the cell design (RAMStatic5T.MSK)

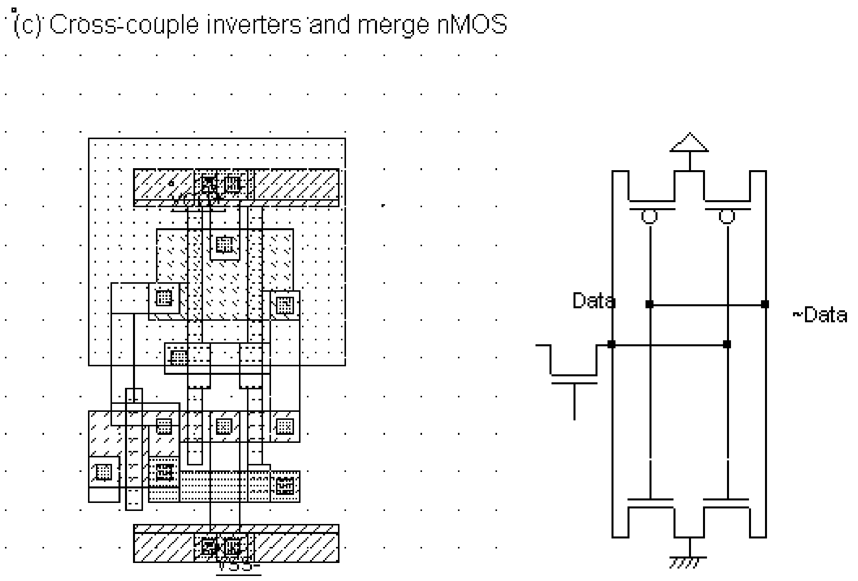


Figure 10-9: Construction of the cross-coupled interconnects (RAMStatic5T.MSK)

The next step, illustrated in figure 10-9, consists in building the cross-coupled interconnects. We need to verify that the design rules are not violated thanks to the Design Rule Checker that can be found in the **Analysis** menu. One interconnect is routed in between the nMOS and pMOS regions, while the other interconnect is routed in the lower part of the cell. Another strategy would consist in enlarging the separation between nMOS and pMOS to rout both cross-coupled interconnects in the middle of the cell.

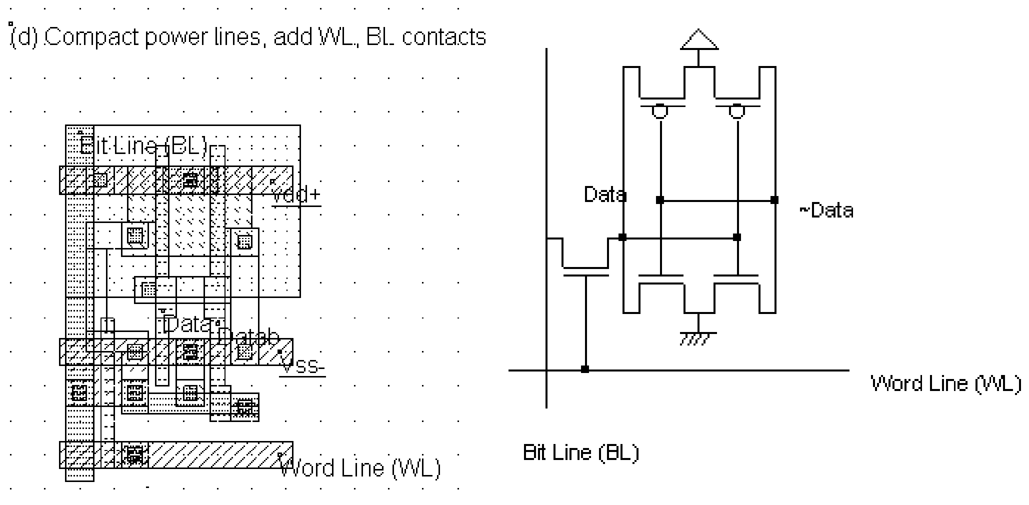
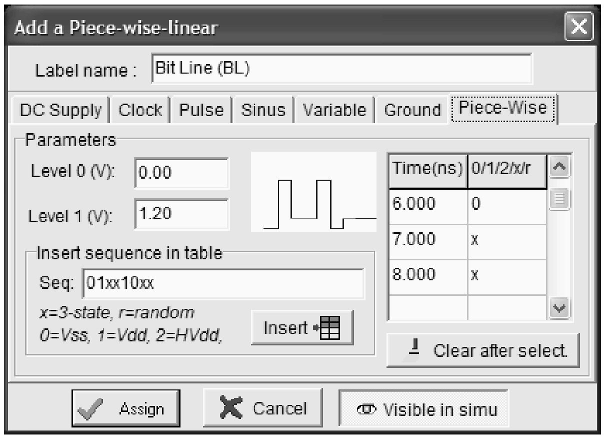


Figure 10-10: Final design of the 5T memory cell (RAMStatic5T.MSK)

The final aspect of the cell is shown in figure 10-10. The **Bit line** signal is made with metal2 and crosses the cell from top to bottom. The supply lines are horizontal, made with metal3. This allows easy matrix-style duplication of the RAM cell.

Simulation of the Five Transistor Cell

The best way to stimulate the RAM cell is to assign a pulse on the *Bit Line*. We insert the sequence 01xx10xx, where "0" means a 0V (Ground voltage), "1" 1.2V (VDD core supply), and "x" means a floating node. When the pulse is in "x" state, the node is no more controlled by the pulse, and the memory cell data may take the lead on the vertical *Bit Line*. The word line is simply assigned a clock.



Label name : Bit Line (BL)

Parameters

Level 0 (V): 0.00

Level 1 (V): 1.20

Insert sequence in table

Seq: 01xx10xx

x=3-state, r=random
0=Vss, 1=Vdd, 2=HVdd,

Time(ns)	0/1/2/x/r
6.000	0
7.000	x
8.000	x

Figure 10-11: Using a pulse to write and read the 5T memory cell (RamStatic5T.MSK)

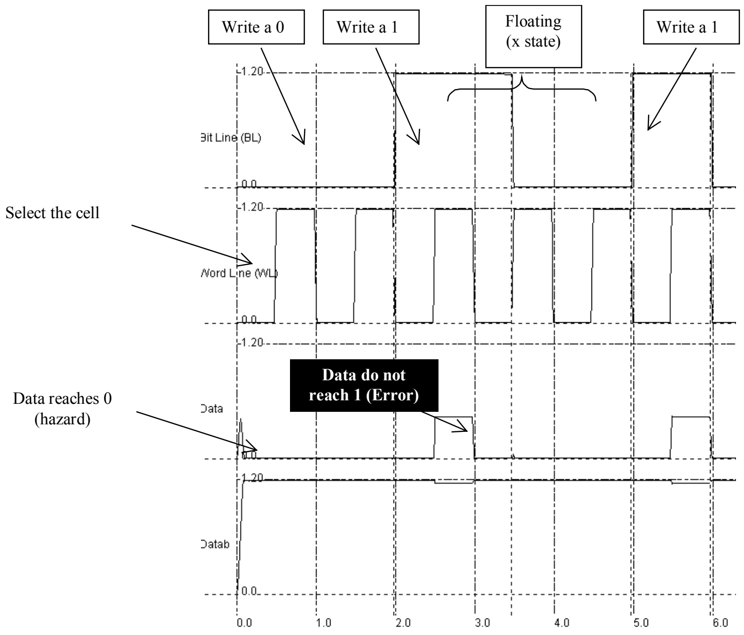


Figure 10-12: Simulation of the 5T memory cell which exhibits a design error(RamStatic5T.MSK)

The simulation of the 5T static RAM is shown in figure 10-12. We have attempted to write a "1" at time 2.0ns, but when the word line WL is asserted (time 2.5ns), we do not observe a change in the memory state, as we would expect. This is due to the wrong sizing of the n-MOS pass transistor, which cannot compete with the n-MOS device of the inverter. Although the pass nMOS width is larger than the nMOS of the inverter, the logic information "1" coming from the bit line turns down to a weak "1" once is passes through the nMOS device (Figure 10-13).

The width of the pass transistor must be enlarged to pass the "1" strongly enough to win the competition against the "0" forced by the nMOS device of the inverter. Consequently, the layout must be changed until the memory state is controlled without any hazard.

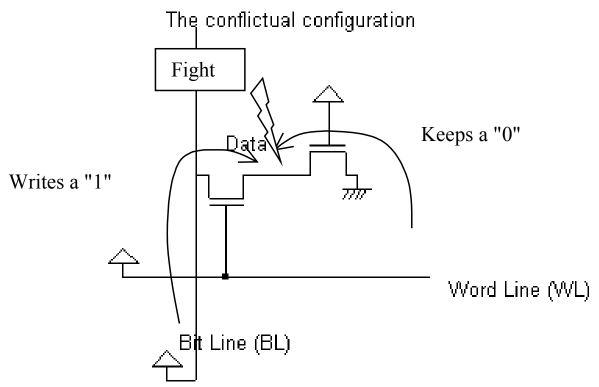


Figure 10-13: Conflict in the static memory cell which is at the origin of the design error(RamStatic5T.MSK)

Corrected Five Transistor Cell

To obtain a minimum safety margin, the following changes must be conducted (Figure 10-14):

- The switching point of the inverter must be lowered. Consequently, the pMOS width must be significantly reduced.
- The pass transistor must be stronger. In other words, the width must be increased. The drawback is a larger cell area.
- The inverter strength must be reduced. The length must be increased. The drawback is a slower cell response.

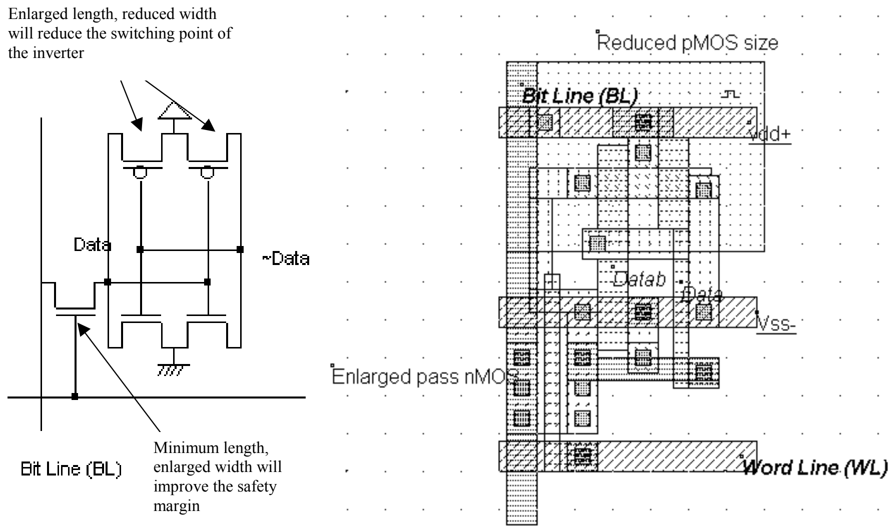

Enlarged length, reduced width will reduce the switching point of the inverter
Data
~Data
Bit Line (BL)
Minimum length, enlarged width will improve the safety margin
Reduced pMOS size
Bit Line (BL)
Vdd+
Datab
Data
Vss-
Enlarged pass nMOS
Word Line (WL)

Figure 10-14: Layout of the modified 5T memory cell (RamStatic5Tb.MSK)

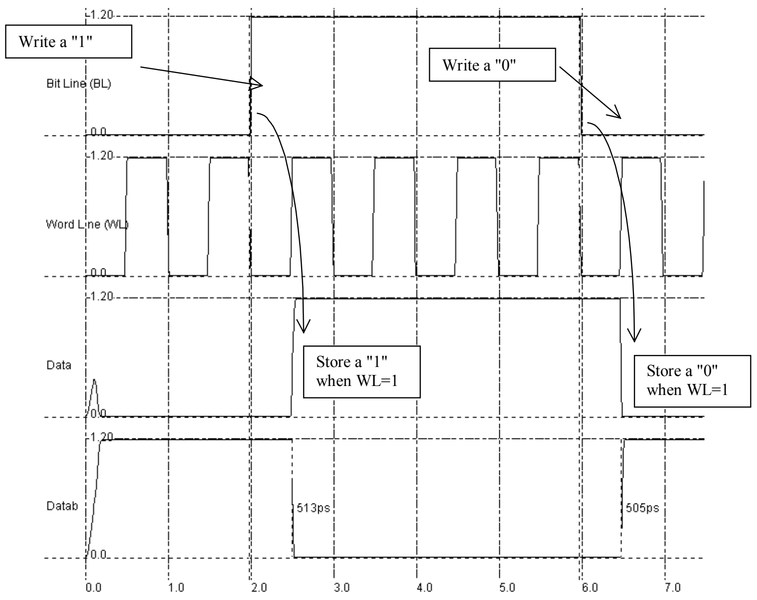


Figure 10-15: Correct simulation of the 5T memory cell (RamStatic5Tb.MSK)

The new layout is shown in figure 10-14. The simulation using model 3 still does not work. However, the simulation using BSIM4 does (Figure 10-15). How should we interpret this result? The modified design is just at the limit between a correct and a wrong design. Any variation (Process, temperature, real MOS dimension) could act on the result. This means that we must change again the design in order to obtain a safe behavior which would not be affected by a small change of parameter. In a 5T implementation, this must be done by further increasing of the pass Mos width, which enlarges the cell dimensions even more. It becomes evident that the 5T design is useless.

The 6 transistor Memory Cell

The basic cell for static memory design is based on 6 transistors, with two pass gates instead of one. The corresponding schematic diagram is given in Figure 10-16. The circuit consists again of the 2 cross-coupled inverters, but uses two pass transistors instead of one. The cell has been designed to be duplicated in X and Y in order to create a large array of cells. Usual sizes for Megabit SRAM memories are 256 column x 256 rows or higher. A modest arrangement of 4x4 RAM cells is proposed in figure 10-16. The selection lines *WL* concern all the cells of one row. The bit lines *BL* and *~BL* concern all the cells of one column.

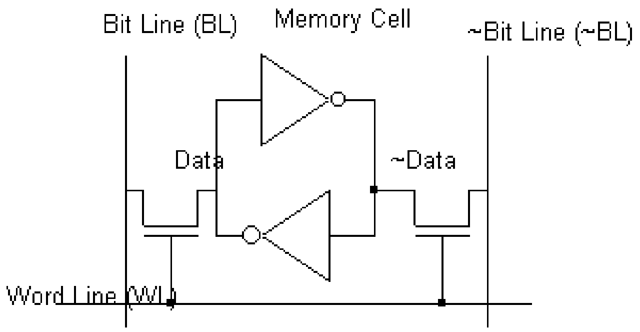


Figure 10-16: The layout of the 6 transistor static memory cell (RAM6T.SCH)

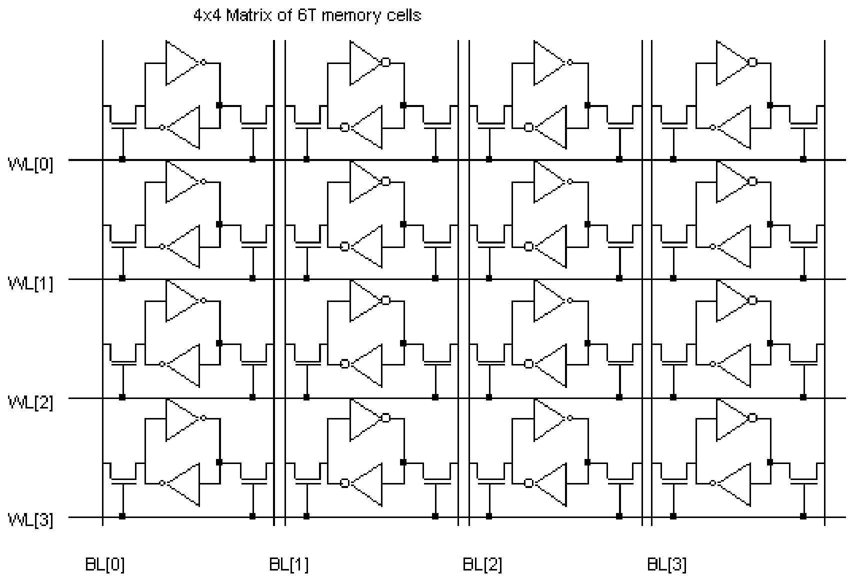


Fig. 10-17 An array of 6T memory cells, with 4 rows and 4 columns (RAM6T.SCH)

The 6T Cell Layout

The RAM layout is given in Figure 10-18. The *BL* and *~BL* signals are made with metal2 and cross the cell from top to bottom. The supply lines are horizontal, made with metal3. This allows easy matrix-style duplication of the RAM cell.

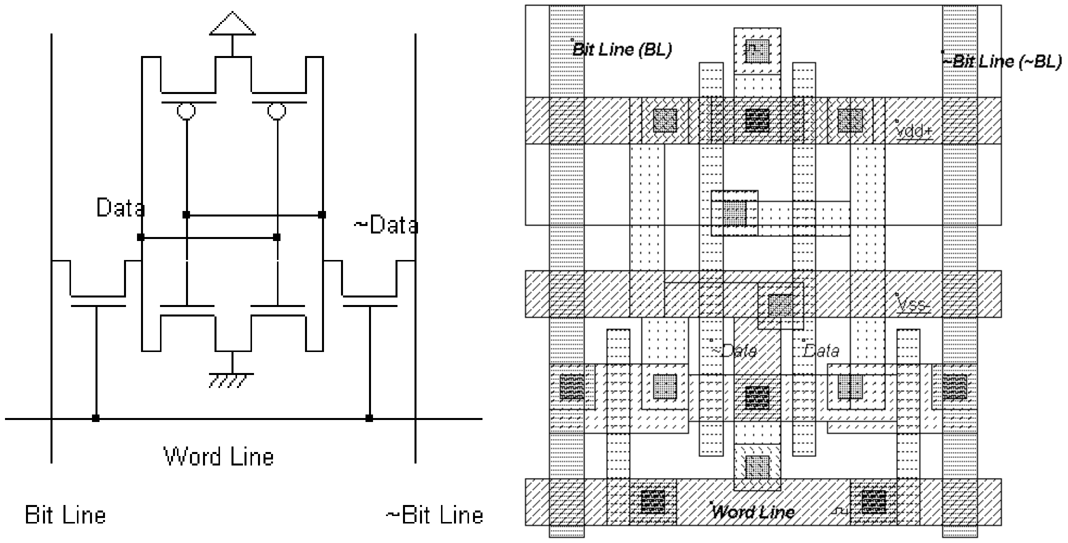


Fig. 10-18. The layout of the static RAM cell (RAM6T.MSK).

The cross-section shows the nMOS devices and the connection to VSS using metal3, situated in the middle of the cell. The *BL* and *~BL* lines, in metal2 are on both sides. The word line controls the access between the bit lines and the internal memory information.

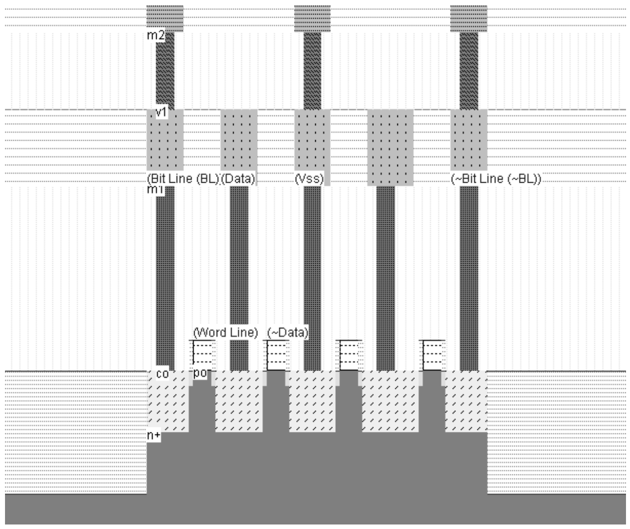


Fig. 10-19. Cross-section of the static RAM cell in the n-channel MOS region (RAM6T.MSK).

The size of the static RAM is given in the menu **Layout size**, accessible through the command **File →Properties**. As shown in figure 10-20, the layout dimensions are 41x46 lambda.

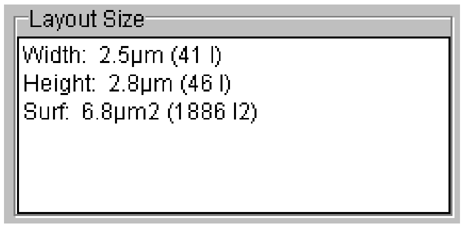

Layout Size
Width: 2.5µm (41 l)
Height: 2.8µm (46 l)
Surf: 6.8µm2 (1886 l2)

Fig. 10-20. The size of the static RAM cell can be found in the Properties menu(RAMStatic6T.MSK).

The 6T Memory Simulation

WRITE CYCLE. Values 1 or 0 must be placed on *Bit Line*, and the data inverted value on *~Bit Line*. Then the selection *Word Line* goes to 1. The two-inverter latch takes the *Bit Line* value. When the selection *Word Line* returns to 0, the RAM is in a memory state.

READ CYCLE. The selection signal *Word Line* must be asserted, but no information should be imposed on the bit lines. In that case, the stored data value propagates to *Bit Line*, and its inverted value *~Data* propagates to *~Bit Line*.

SIMULATION. The simulation parameters correspond to the read and write cycle in the RAM. The simulation steps proposed in figure 10-21 consist in writing a 0, a 1, and then reading the 1. In a second phase, we write a 1, a 0, and read the 0. The *Bit Line* and *~Bit Line* signals are controlled by pulses. The floating state is obtained by inserting the letter "x" instead of 1 or 0 in the description of the signal.

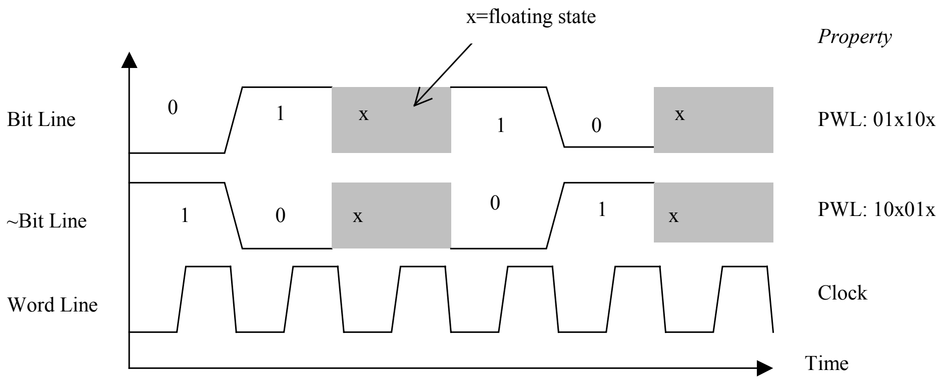
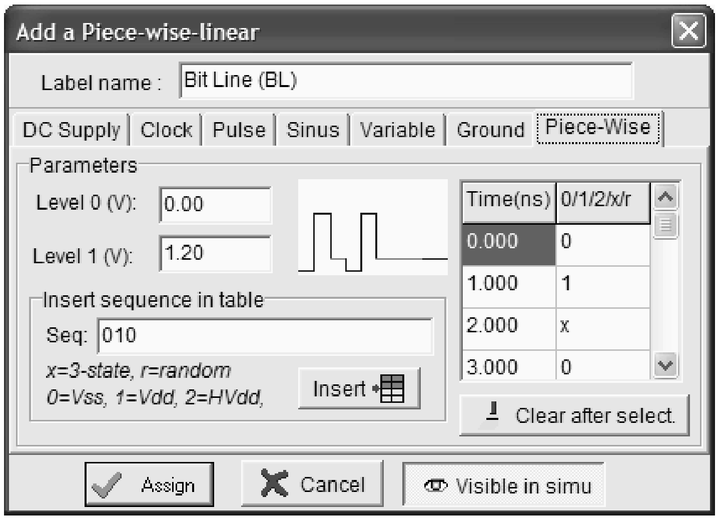


Figure 10-21. Proposed stimulation patterns for the simulation of the 6T static Ram memory (RamStatic6T.MSK)



Label name : Bit Line (BL)

Parameters

Level 0 (V): 0.00

Level 1 (V): 1.20

Insert sequence in table

Seq: 010

x=3-state, r=random
0=Vss, 1=Vdd, 2=HVdd,

Time(ns)	0/1/2/x/r
0.000	0
1.000	1
2.000	x
3.000	0

Fig. 10-22. The bit Line pulse uses the "x" floating state to enable the reading of the memory cell (RamStatic6T.MSK)

The simulation of the RAM cell is proposed in figure 10-23. At time 0.0, *Data* reaches an unpredictable value of 1 after an unstable period. Meanwhile, *~Data* reaches 0. At time 0.5ns, the memory cell is selected by a 1 on *Word Line*. As the *Bit Line* information is 0, the memory cell information *Data* goes down to 0. At time 1.5ns, the memory cell is selected again. As the *Bit Line* information is now 1, the memory cell information *Data* goes to 1. During the read cycle, in which *Bit Line* and *~Bit Line* signals are floating, the memory sets these wires respectively to 1 and 0, corresponding to the stored values.

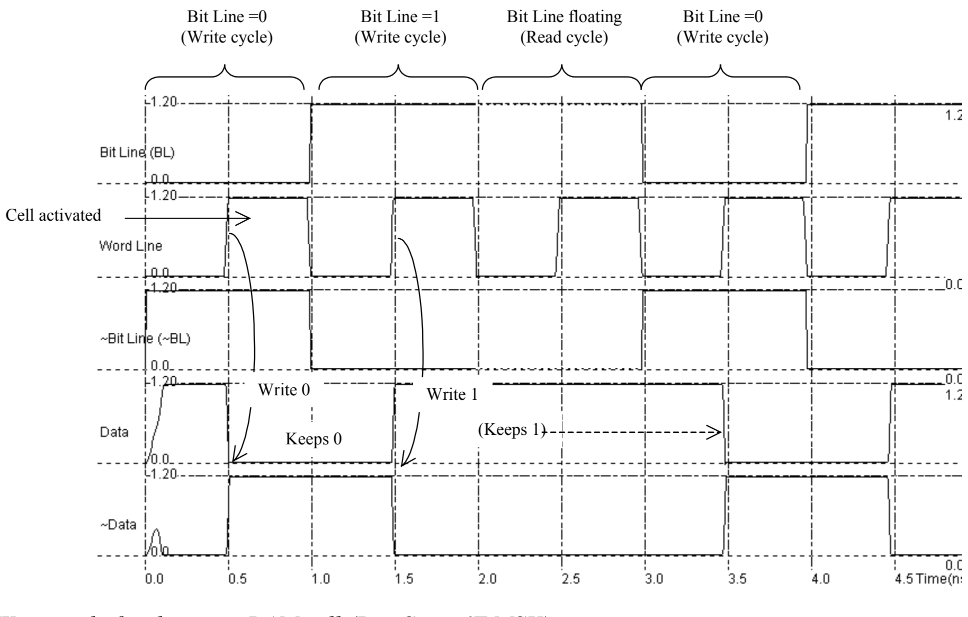
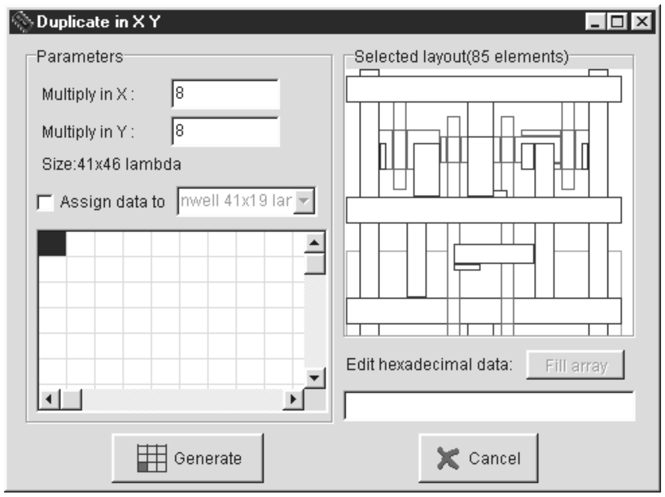


Fig. 10-23. Write cycle for the static RAM cell (RamStatic6T.MSK).

10.3 A 64 Bit Static RAM

A Poor 64 bit RAM array

We can duplicate the RAM cell into a 8x8 bit array using the command **Edit -> Duplicate XY**. Select the whole RAM cell and a new window appears. Enter the value « 8 » for X and « 8 » for Y into the menu. Click on « **Generate** ». The array of 8x8 memory cells is generated.


Duplicate in X Y

Parameters

Multiply in X : 8

Multiply in Y : 8

Size:41x46 lambda

Assign data to nwell 41x19 lar

Selected layout(85 elements)

Edit hexadecimal data:

Fig. 10-24. Duplicating the RAM Cell in X and Y (Ram8x8.MSK)

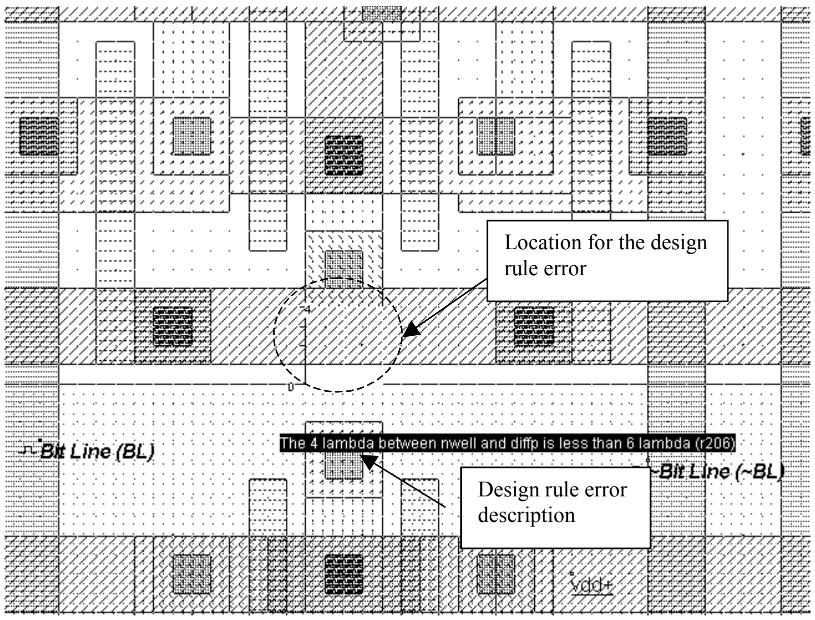

Location for the design rule error
Bit Line (BL)
~Bit Line (~BL)
Design rule error description
Vdd+
The 4 lambda between nwell and diffp is less than 6 lambda (r206)

Fig. 10-25. Design rule error between polarization and n-well (Ram8x8.MSK)

Following the duplicate command, it is recommended to verify the design rules, due to the proximity in X and Y of other cells. In figure 10-25, there is a violation of the minimum distance between the polarization contact and the next n-well zone. It can be noticed that we lose an important area at the interface between the pMOS devices and the upper nMOS devices. A good idea is to flip the cells in order to share the VDD polarization and bulk contacts.

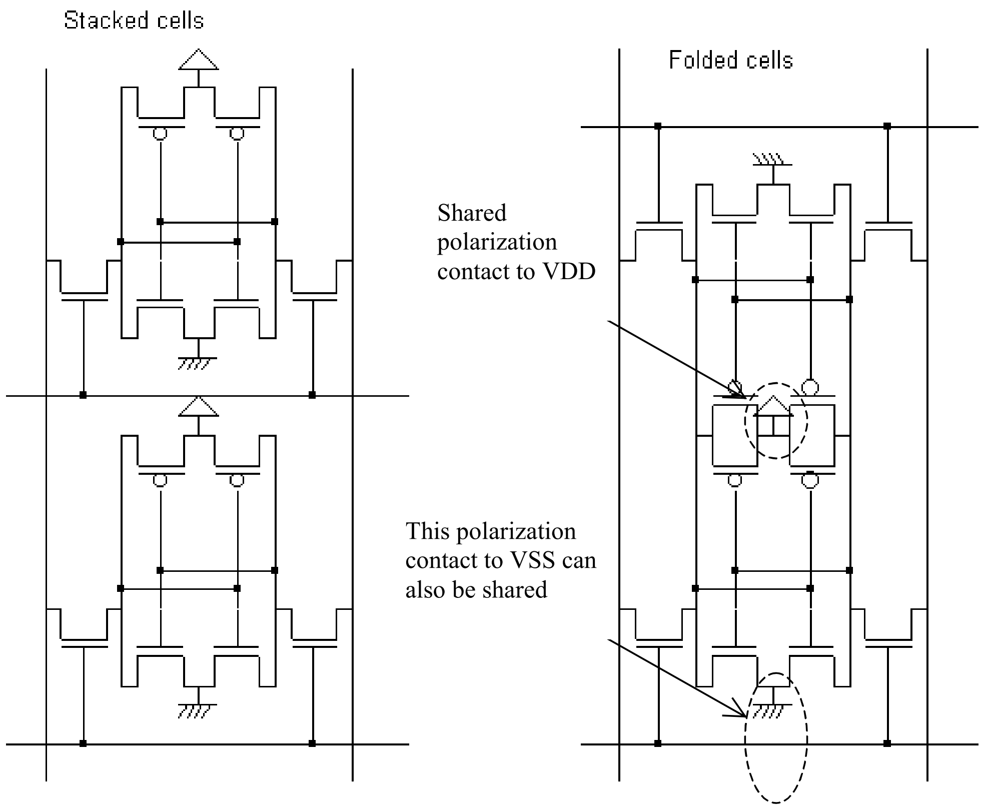

Stacked cells
Folded cells
Shared polarization contact to VDD
This polarization contact to VSS can also be shared

Figure 10-26. Sharing the VDD well and polarization helps reducing the cell area (Ram8x8.MSK)

Compared to the layout of figure 10-25, which corresponds to a 48 lambda height (Once the design rules have been complied with), the height of the same static RAM with shared VDD supply becomes equal to 43 lambda, which corresponds approximately to a 10% gain. There are many ways to further improve the 6T cell density.

A Compact 8x8 RAM array

A very interesting approach to obtain a more compact memory cell consists in sharing all possible contacts: the supply contact, the ground contact and the bit line contacts. The consequence is that the effective cell size can be significantly reduced. An example of very compact memory cell [Sharma] is given in figure 10-27.

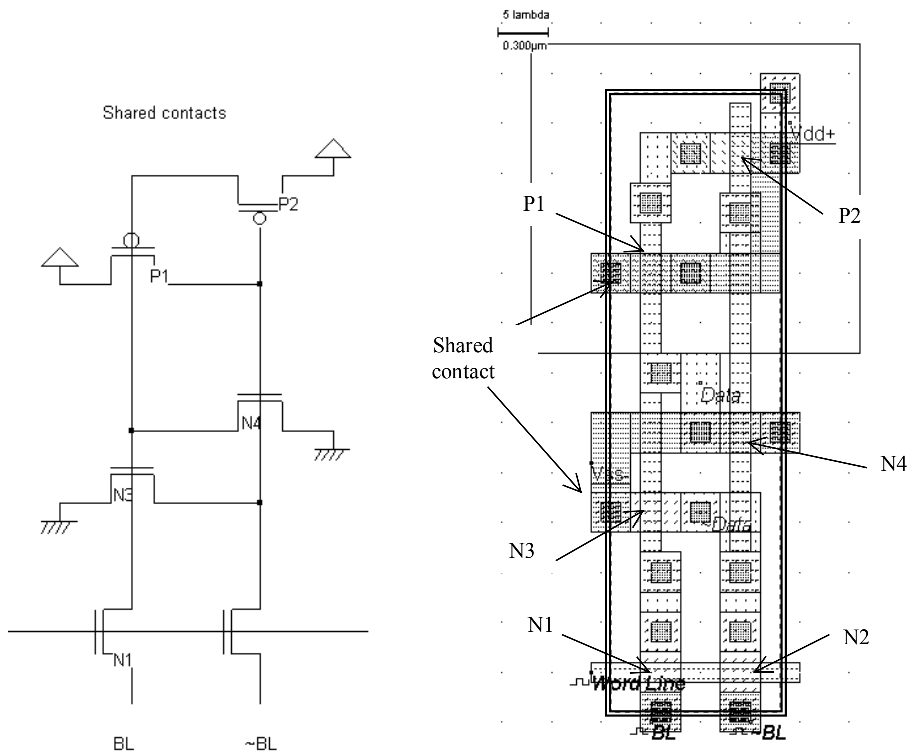


Figure 10-27. Sharing all possible contacts lead to a very compact cell design (Ram6Tcompact.MSK)

The layout is functionally identical to the previous layout. The only difference is the placement of MOS devices and contacts. We duplicate the RAM cell into a 64 bit array. The multiplication cannot be done directly by the command **Duplicate XY,** as we need to flip one cell horizontally to share lateral contacts, and flip the resulting block vertically to share vertical contacts (Figure 10-28).

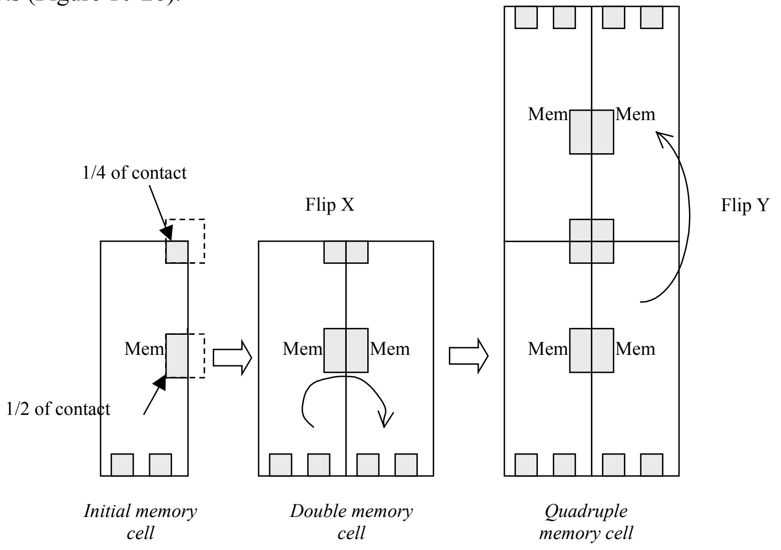


Figure 10-28. The 4 cell memory pattern with shared contacts (Ram6Tcompact.MSK)

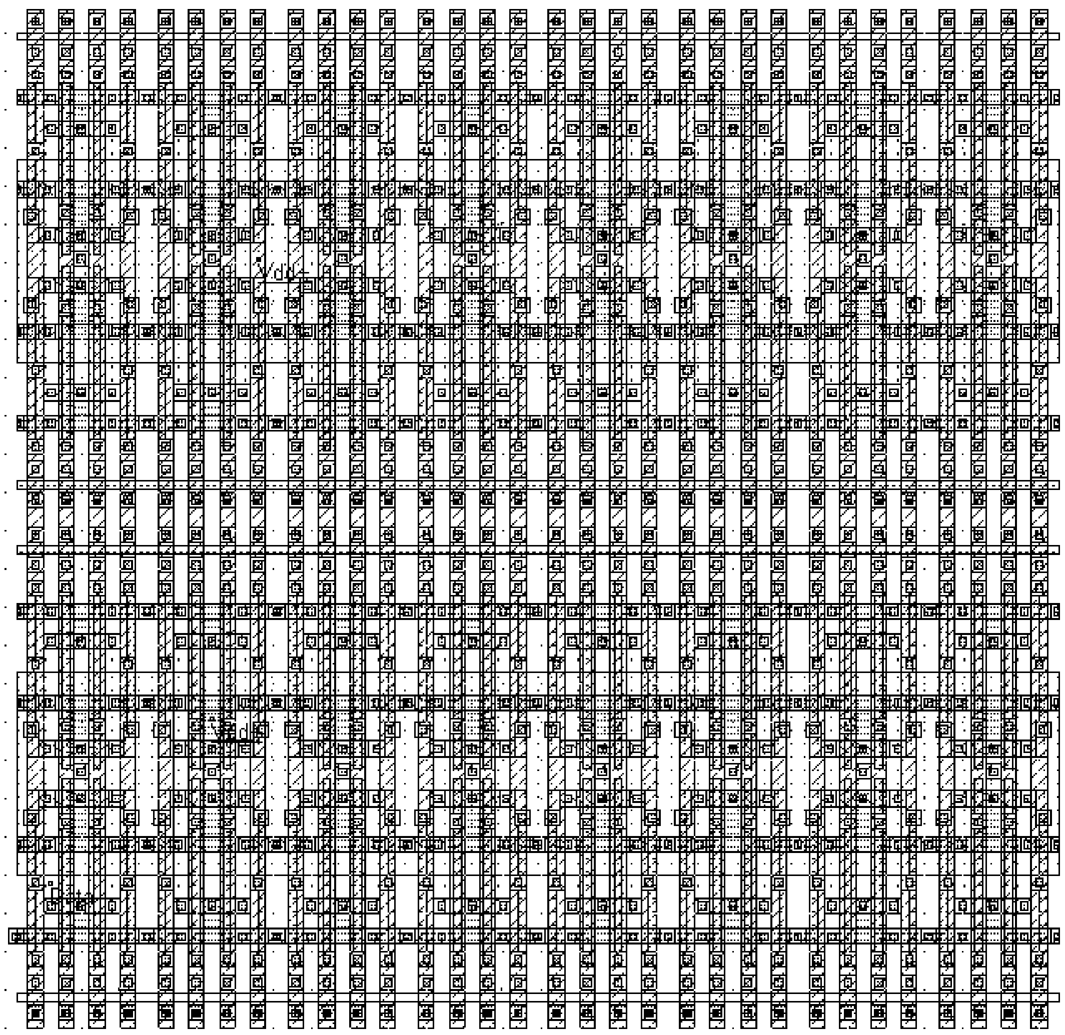


Figure 10-29. Compact 16x4 array of memory cells with shared contacts (Ram16x4Compact.MSK)

As the cell is very narrow, we organize the layout in a 16x4 bit arrangement (16 cells in X, 4 cells in Y). The Ram8x8 block based on the cell "RamStatic6T" leads to a 434µm2 layout (0.12µm), while the Ram16x4 block based on "Ram6Tcompact" leads to a 262 µm2 layout, that is a 45% gain. The result is very dense, as shown in figure 10-29. However, specific technological features are developed (45° layout, direct diffusion to poly contact, pseudo pMOS, local rule violation) to further reduce the cell area by a factor of 2 and more [Sharma].

Checking the Fault Margin

Although the simulation of the compact memory cell works fine, the question remains whether we are close to an erroneous writing, or not. In other words, do we have a sufficient margin? One answer consists in changing the sizing of the critical transistors and run the simulation. We consider that a 20% margin is sufficient to warranty a correct behavior. Clearly, the pass transistor should be designed large for both fast write and quick read operations [Harzasti]. However, for the objective of a small silicon area and low power consumption, the pass transistor should be kept small. There is clearly a compromise to find.

Secondly, for a safe operation, the pMOS device of the inverter should be small, to lower the commutation point. Meanwhile, a poor pMOS drive would slow down the read operation. We investigate the role of the pMOS width in figure 10-30. The reference size, called (a) in the figure, is the one provided in RAM6Tcompact.MSK. The reference pMOS size is *Wp*=4 lambda, *Lp*=2 lambda. The reference pass transistor has a 2 lambda gate length and 4 lambda gate width.

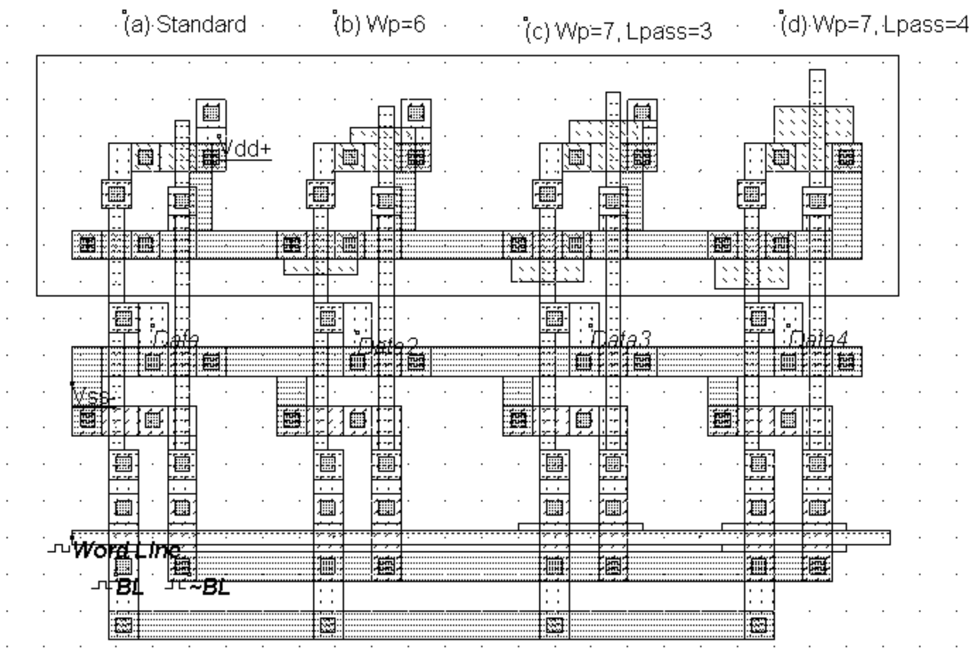


Figure 10-30: Changing the design sizing of critical MOS devices to investigate the fault margin (Ram6Tmargin.MSK)

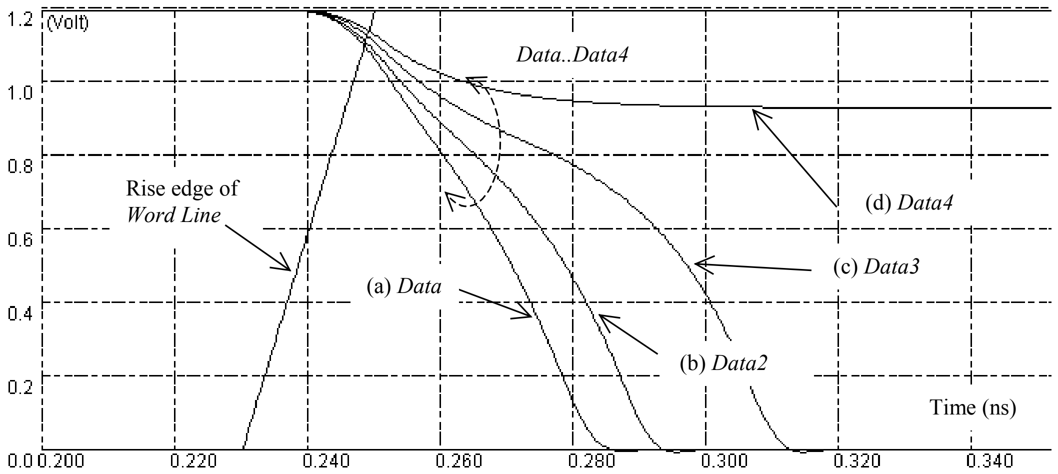
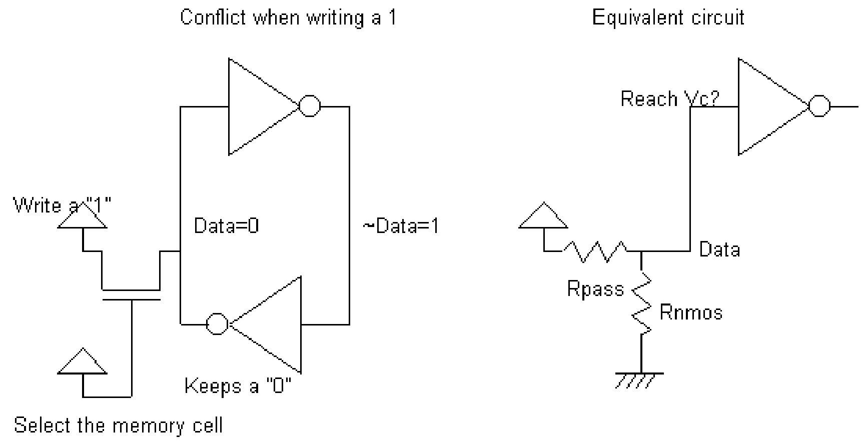


Figure 10-31: Simulation of the four designs when writing a "0" (Ram6Tmargin.MSK)

In design (b) of figure 10-30, the commutation point of the inverter is raised by enlarging the pMOS device, getting closer to a write failure. In (c) the commutation point is increased even more, and the drive of the pass transistor is reduced by enlarging the channel. In (d) the pass transistor channel is altered even more.

The simulation of figure 10-31 consists in writing a "0" in the memory cell. This corresponds to a conflict between the pass transistor which is connected to "0" and the lower inverter which keeps a "1", because ~*Data*=1. The conflict is solved once the *Data* voltage reaches the commutation point *Vc* of the upper inverter, which changes the state of the loop. We note a correct waveform for designs (a) and (b), a slow change for design (c) and an erroneous state "1" for design (d).



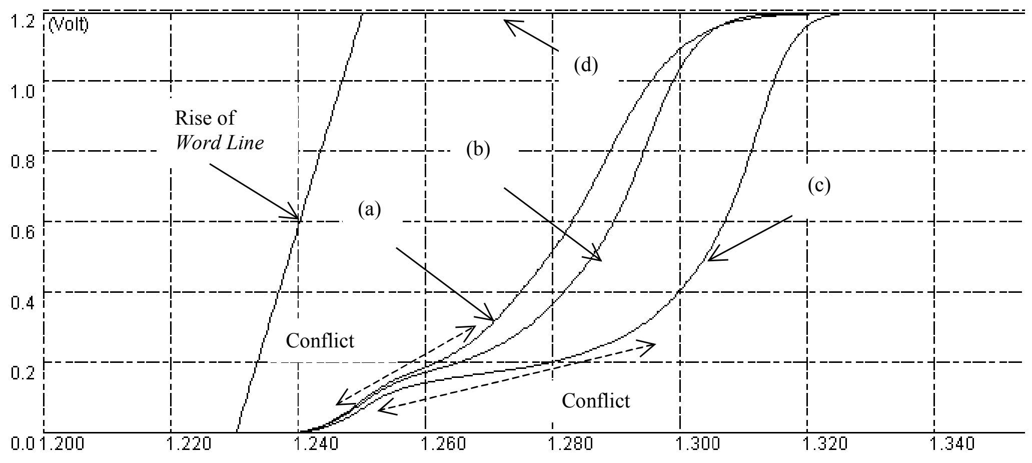


Figure 10-32: Simulation of the four designs when writing a "1" (Ram6Tmargin.MSK)

The simulation of figure 10-32 consists in writing a "1" in the memory cell. This operation is critical as the pass transistors compets with the inverter and the memory may remain at "0", which was fatal for the 5-transistor design of figure 10-10. Again, we note a correct waveform for designs (a) and (b), a slow change for design (c). Design (d) is stuck at "1". The margin between design (a) and (c-d) is sufficient to consider the standard design as a safe design.

Row Selection Circuit

The row selection circuit decodes the row address and activates one single row. This row is shared by all word line signals of the row. The row selection circuit is based on a multiplexor circuit. One line is asserted while all the other lines are at zero.

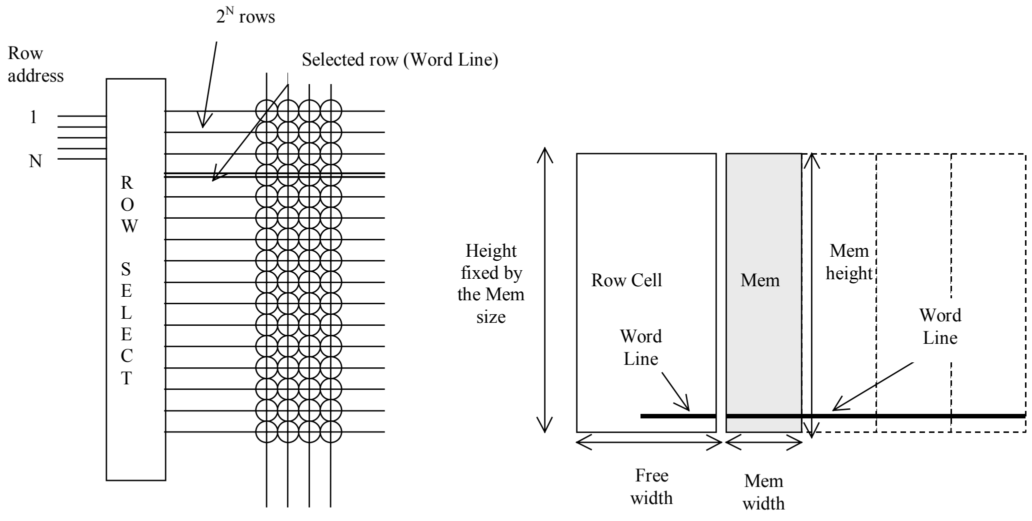


Fig. 10-33 The row selection circuit

In the row selection circuit for the 16x4 array, we simply need to decode a two-bit address. Using AND gates is one simple solution. In figure 10-34, we present the schematic diagram of 2-to-4 and 3-to-8 decoders. In the case of a very large number of address lines, the decoder is split into sub-decoders which handle a reduced number of address lines.

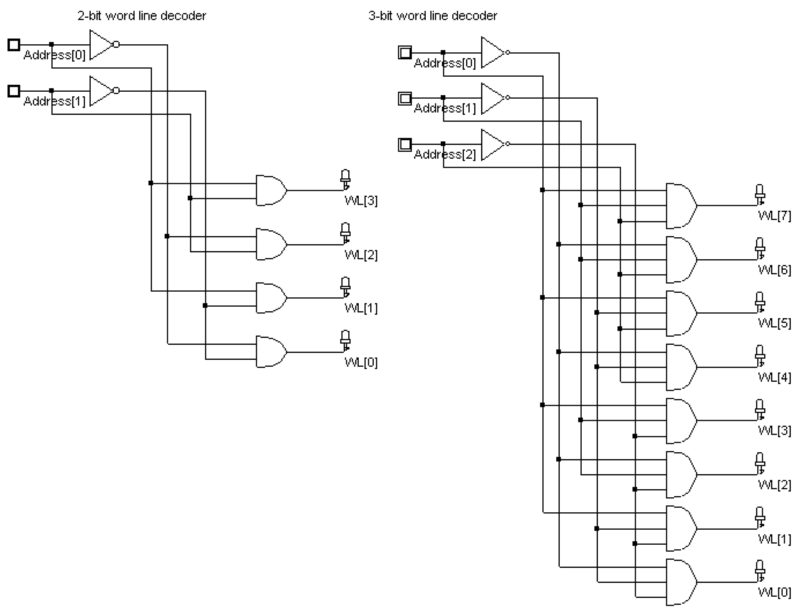


Fig. 10-34. The row selection circuit in 2 bit and 3 bit configuration (RamWordLine.SCH)

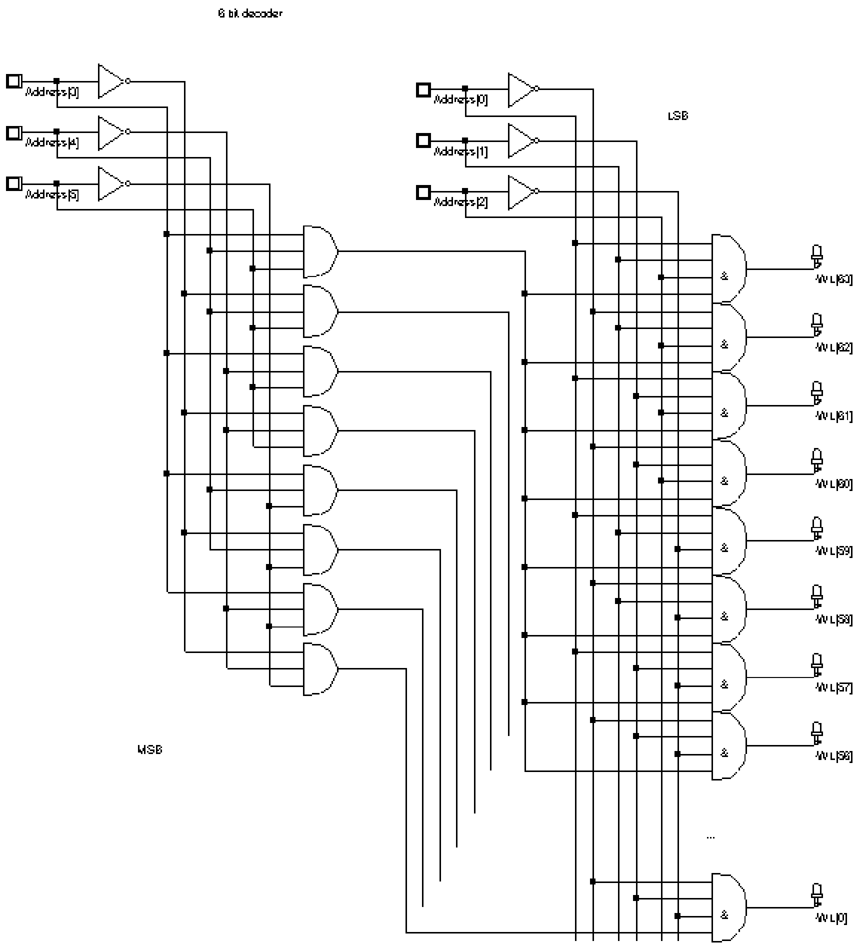


Fig. 10-35. The 6-bit row selection circuit using groups of 3 bits (RamWordLine.SCH)

The 6-bit decoder of figure 10-35 is built from two stages of 3-bit decoders. A total of 64 word lines are generated using this circuit. All word lines have not been shown for clarity's sake. The row selection circuit loads a very significant capacitance which is the sum of *Word Bit* of each elementary memory cell. Consequently, the AND gate is designed using a NAND gate followed by a strong buffer (Figure 10-36).

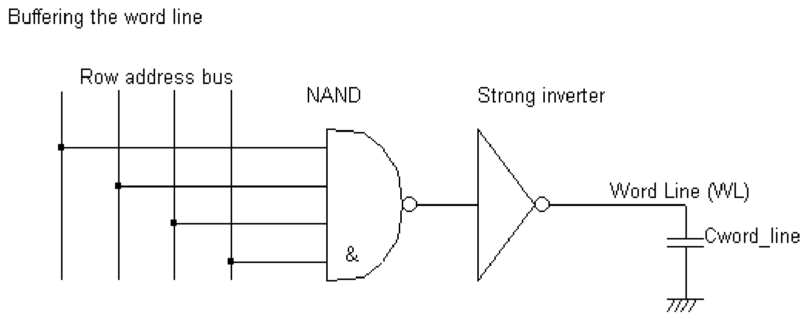


Figure 10-36. Buffering the word line command (RamWordLine.SCH)

The Row Selection circuit height should be adjusted to that of the RAM cell height. When making the final assembly between blocks, the command **Edit → Move Step by Step** is very useful. This command helps to move a selected block with a lambda step (Figure 10-37).

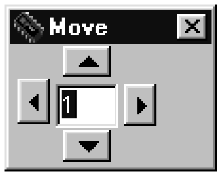


Figure 10-37 Moving a portion of layout step by step

The row selection layout has the particularity to be very regular. It encrypts a binary pattern through the addition of contacts, as illustrated in figure 10-38. The binary encoding can be realized in a semi-automatic way in Microwind. The idea is to create a pattern consisting of a vertical metal box (Address lines), crossing a horizontal metal2 box, and a via to build an electrical link between these two nodes. The basic pattern is duplicated 6 times in X and 3 times in Y for each *word line* circuit. The via layer is copied only if a physical layer is needed at the intersection between vertical address lines and horizontal interconnects. The copying of the via is controlled by a Boolean table sown in figure 10-39.

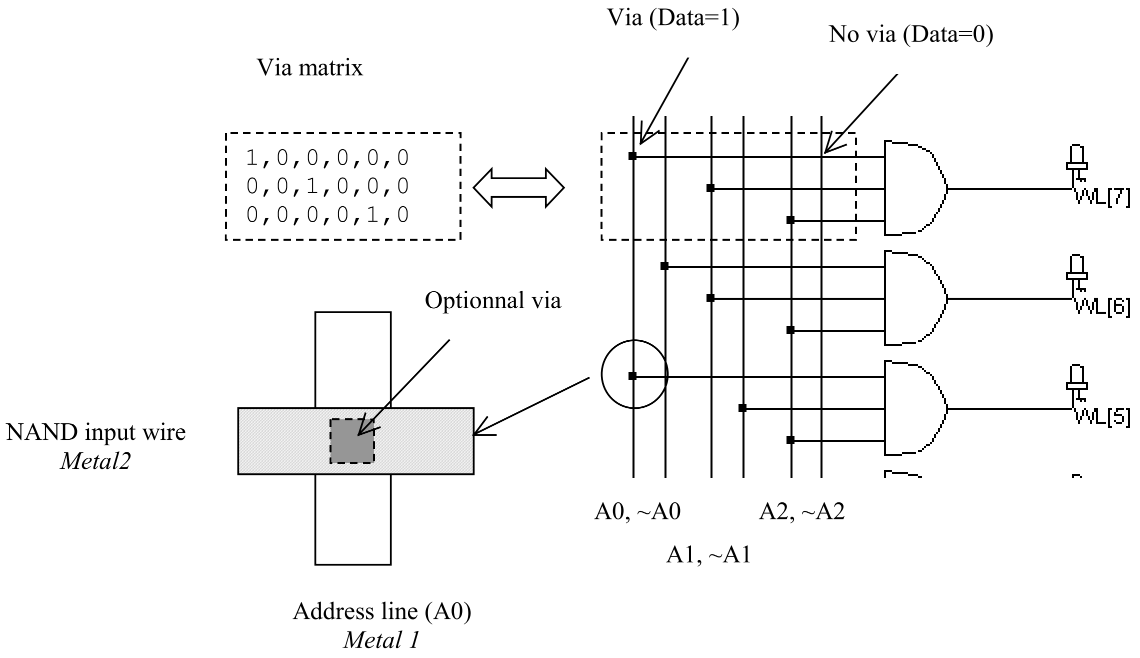
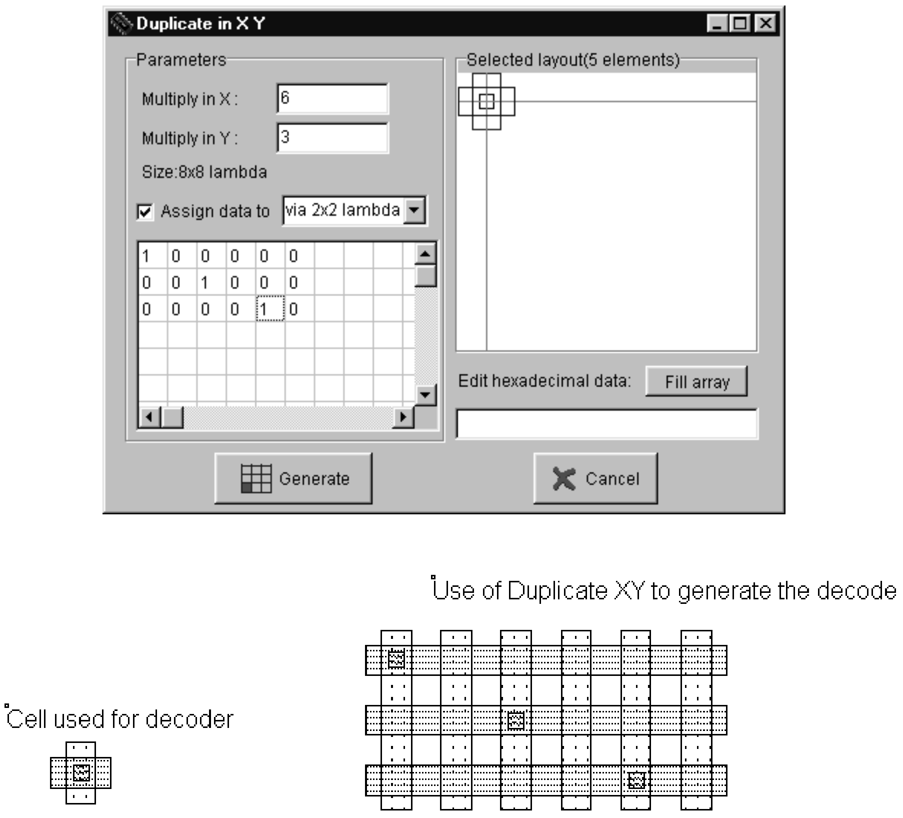


Figure 10-38 The address uses contacts on selected address lines to program the word line according to each address.

The data matrix is filled by zeros and ones. A zero indicates that no via will be generated, a one indicates that the via will be copied. The tick *Assign data* must be asserted, and the via box must be chosen in this case. In the Boolean matrix, the "0" may be changed into a "1" by a double click at the desired location.



Multiply in X : 6

Multiply in Y : 3

Size:8x8 lambda

[✓] Assign data to via 2x2 lambda

1	0	0	0	0	0
0	0	1	0	0	0
0	0	0	0	1	0

Edit hexadecimal data:

Figure 10-39 Generating the matrix of contacts using the Duplicate XY command (RamDecoder.MSK)

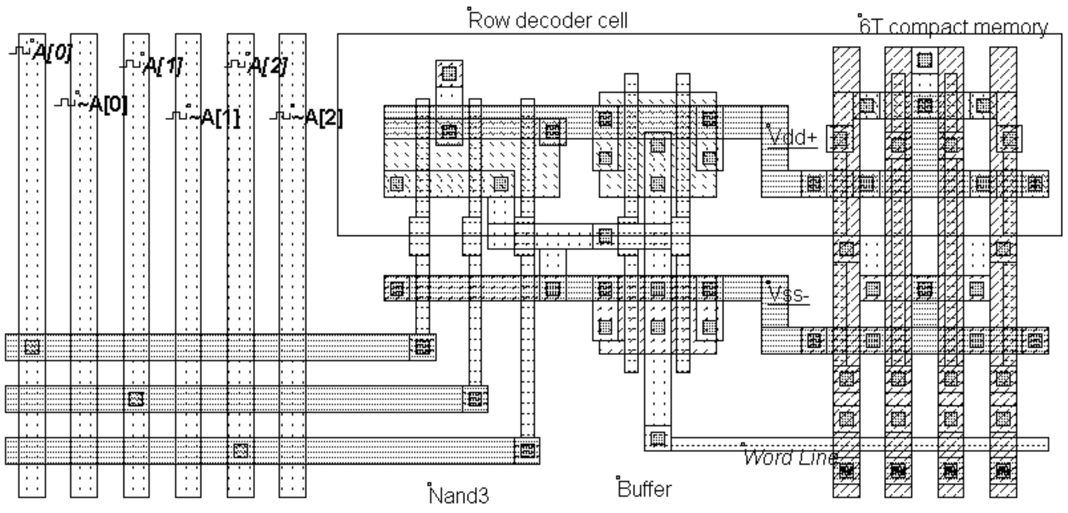


Figure 10-40 The decoder circuit and its link to the memory array (RamDecoder.MSK)

The aspect of the row decoder circuit is shown in figure 10-40. On the left side, we recognize the regular interconnect matrix and its associated via. In the middle, the 3-input NAND gate and a buffer are designed, with the appropriate whirring and supply connections to fit exactly with the rigid layout of the static RAM cell.

Column Selection Circuit

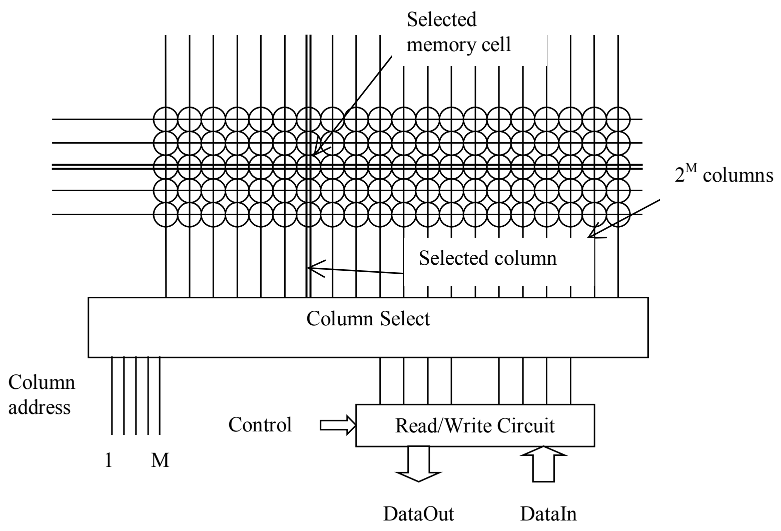


Figure 10-41. The column selection circuit principles

The column decoder selects a particular column in the memory array to read the contents of the selected memory cell (Figure 10-41) or to modify its contents. The column selector is based on the same principles as those of the row decoder. The major modification is that the data flows both ways, that is either from the memory cell to the *DataOu*t signal (Read cycle), or from the *DataIn* signal to the cell (Write cycle).

Figure 10-42 proposes an architecture based on n-channel MOS pass transistors. We consider here 4 columns of memory cells, which requires 2 address signals *Address_Col[0]* and *Address_Col[1].* The n-channel MOS device is used as a switch controlled by the column selection. When the nMOS is on and *Write* is asserted, (Figure 10-42) the *DataIn* is amplified by the buffer, flows from the bottom to the top and reaches the memory through *BL* and ***~BL***. If *Write* is off, the 3-state inverter is in high impedance, which allows the information to be read on *DataOut*.

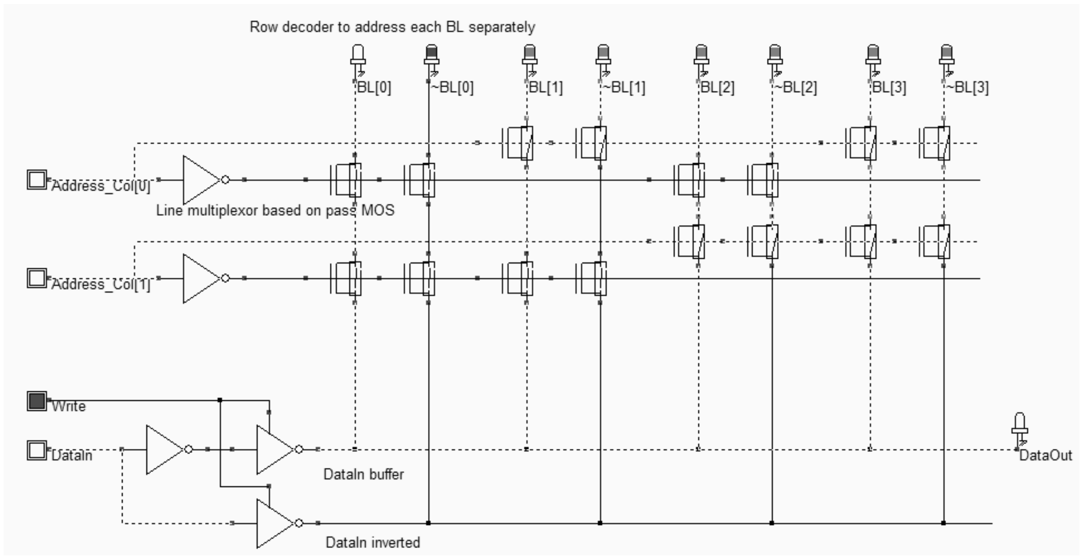


Figure 10-42. Row selection and Read/Write circuit (RamColumn.SCH)

In many cases, the *DataIn* and *DataOut* signals have a wide format, usually 8 or 16 bits. In the schematic diagram of figure 10-43, the *DataIn* and *DataOut* bus is 2-bit wide. Only one address line *Address_Col[0]* is required to select the appropriate columns. From a layout point of view, the nMOS transistors should fit the narrow width of the memory cell. This is usually done by stacking *BL* and *~BL* pass MOS devices on top of each other. Furthermore, the pass transistors should be designed with a large width to avoid any bad surprise at the write cycle.

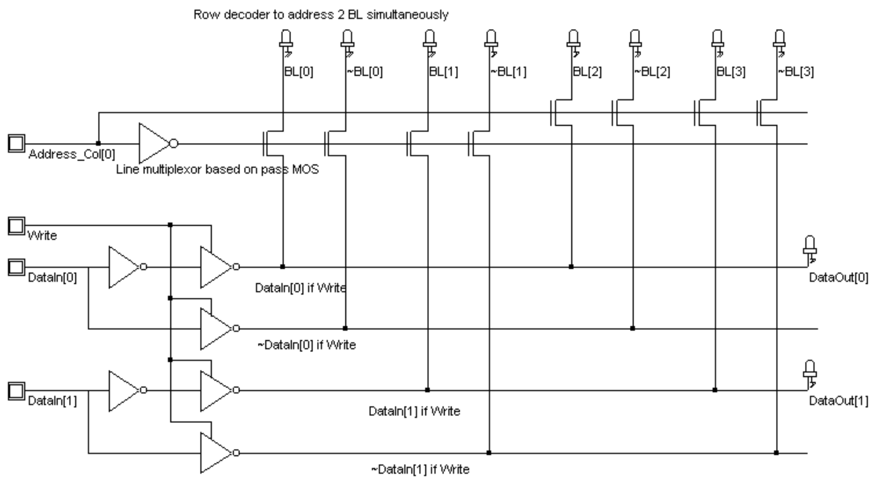


Figure 10-43. Row selection and Read/Write circuit with a 2-bit DataIn and DataOut information (RamColumn.SCH)

A Complete 64 bit SRAM

The 64 bit SRAM memory interface is shown in figure 10-44. The 64 bits of memory are organized in words of 4 bits, meaning that *DataIn* and *DataOut* have a 4 bit width. Each data *D0..D15* occupies 4 contiguous memory cells in the array. Four address lines are necessary to decode one address among 16. The memory structure shown in figure 10-44 requires two address lines *A0* and *A1* for the word lines *WL[0]..WL[3]* and two address lines *A2* and *A3* for the bit line selection. The final layout of the 64 bit static RAM is proposed in Figure 10-45.

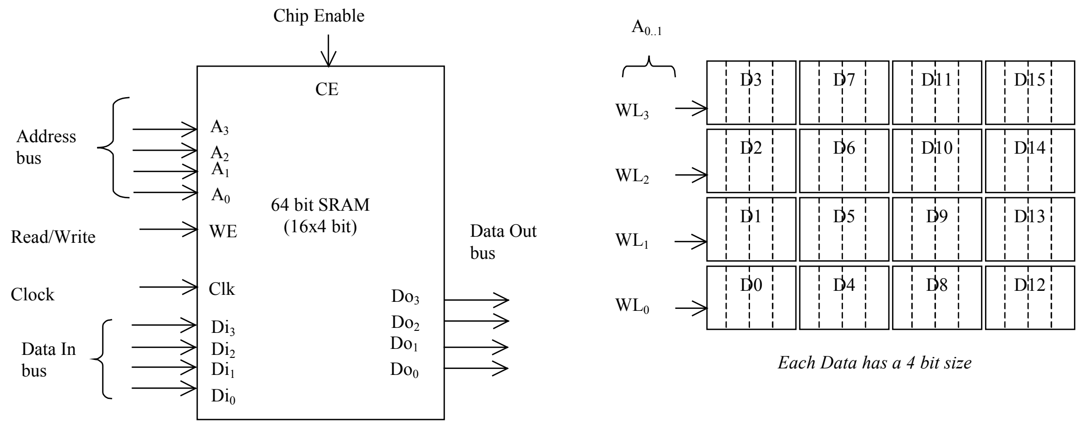


Figure 10-44. The architecture of the 64 bit RAM (RAM64.MSK)

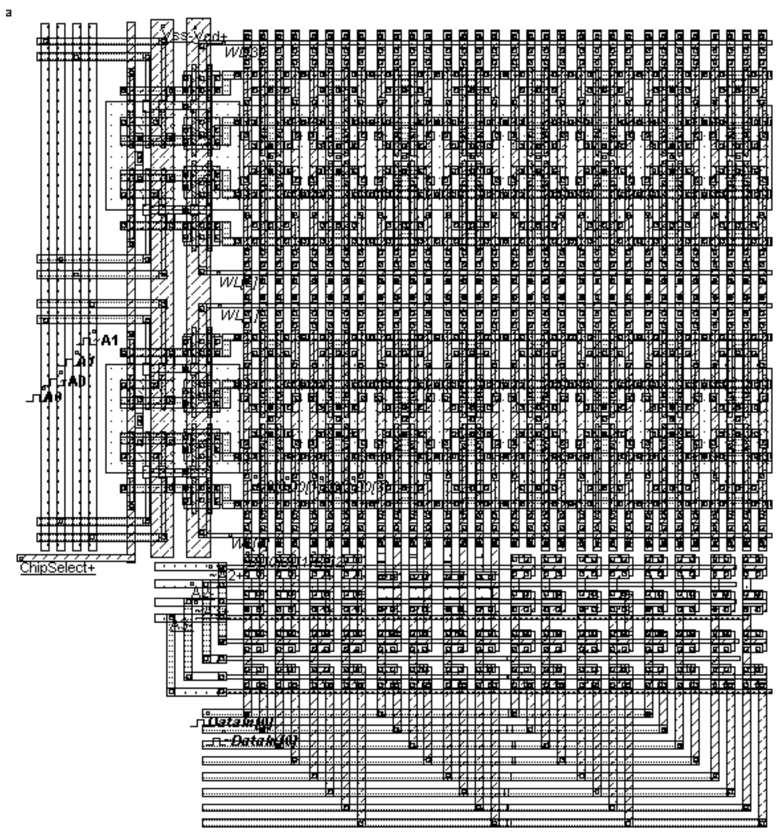
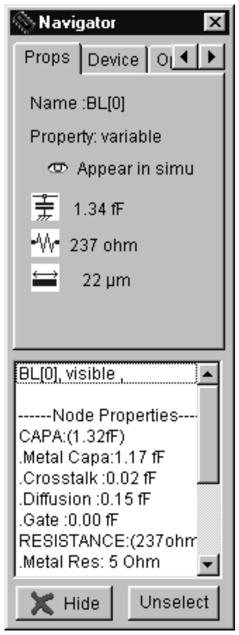


Figure 10-45. The complete RAM layout (RAM64.MSK)

Emulating a 1024x1024 memory array

The 64 bit static RAM undoubtedly has a very short access time, which is not representative of the real case memory circuits. In reality, 1Mb memory arrays are often created by designing a large matrix of 1024 cells in X and 1024 in Y. The major consequence in terms of read access performances is the addition of a very huge parasitic load on each vertical bit line, which is shared by 1024 cells. Notice that the word line is also shared by 1024 cells, which slows down considerably the line selection process.

Unfortunately, Microwind is not able to handle large complexity designs such as the 1024x1024 static RAM array. An error message would appear, and neither the extraction nor the simulation would run. The idea is to emulate the behavior of the 6T cell in a reduced configuration but with an equivalent parasitic load corresponding to the 1 mega-bit array, by using a virtual capacitor. You may find the virtual capacitor symbol in the palette. The question is to evaluate the approximated value of the total bit line and word line capacitance created by a 1Mb array. This can be done by evaluating the parasitic capacitance of one column in the 64bit RAM array, then by dividing by the number of cells, and finally by placing a virtual capacitance equivalent to the cell value multiplied by 1023.



Navigator

Props | Device | O

Name :BL[0]

Property: variable

Appear in simu

1.34 fF

237 ohm

22 µm

```
BL[0], visible ,

------Node Properties---
CAPA:(1.32fF)
.Metal Capa:1.17 fF
.Crosstalk :0.02 fF
.Diffusion :0.15 fF
.Gate :0.00 fF
RESISTANCE:(237ohm
.Metal Res: 5 Ohm
```

Hide | Unselect

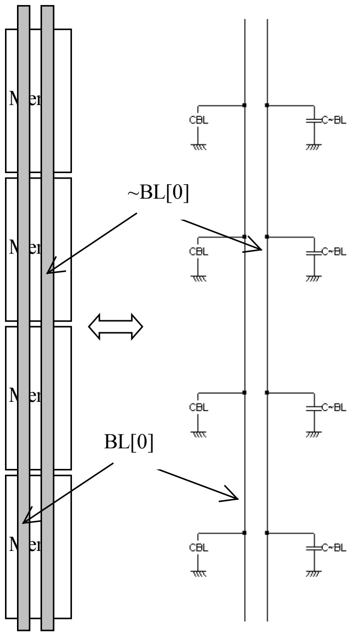


Figure 10-46 Information on the BitLine parasitic capacitance (Ram64.MSK)

The parasitic capacitance of the *BitLine* [0] is about 1.2fF (0.12µm technology). This means that each cell loads the *BitLine* by 0.3fF. The equivalent parasitic capacitance corresponding to 1024 cells would be around 300fF. In the file Ram64Loaded, not only is the simulation slow, but the 300fF load is almost equivalent to a voltage source, and may provoke a logical error, as shown in figure 10-47. First, we write a zero on *Data0* (t=0ns), and a one on *Data1* (t=1ns). When we read *Data0*, the bit lines are still at a high level on *BL[0]* and a low level on *~BL[0]* which is a heritage of the previous write cycle. At time t=2ns, the bit line levels corrupt the reading of the zero, and force the cell to an erroneous one.

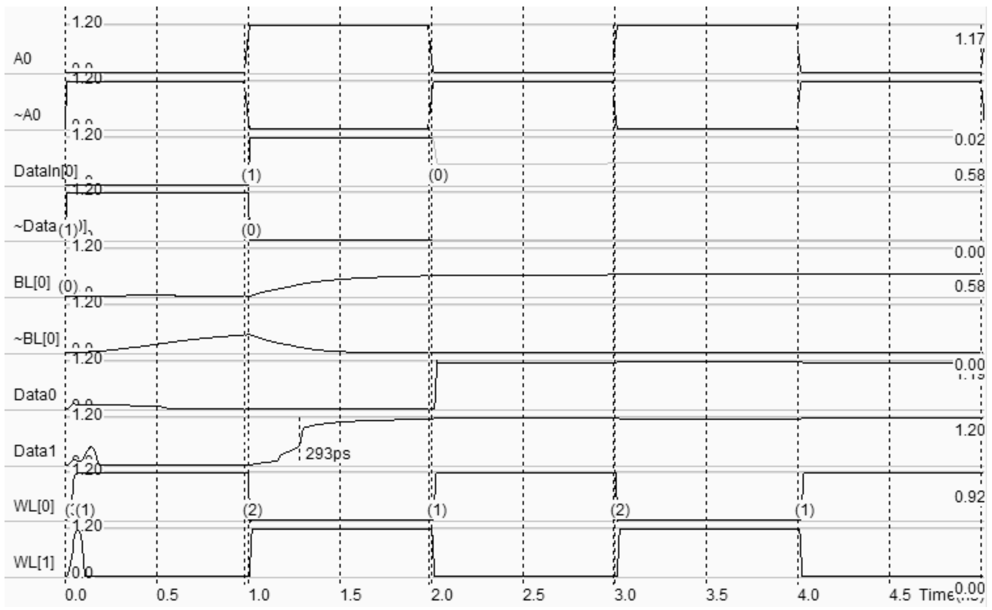


Figure 10-47 The 300fF loading provokes a reading fault at time 2.0ns (Ram64Loaded.MSK)

Precharge Circuit

To avoid logical errors due to the Bit line parasitic load capacitance and ensure safe read and write operations, a modification of the memory array and timing sequence are required. The usual voltage of precharge is VDD/2. Before reading or writing into the memory, the bit lines are tied to VDD/2 using appropriate pass gates. This represents a precharge operation. When reading a 0, the *BL* and *~BL* diverge from VDD/2 (Figure 10-48) and reach the "1" and "0" levels after a short time. When reading a 1, *BL* reaches "1" while *~BL* reaches "0". As the static RAM cells are based on active devices (Two ring inverters), the SRAM memories usually provide the fastest reading and writing access times.

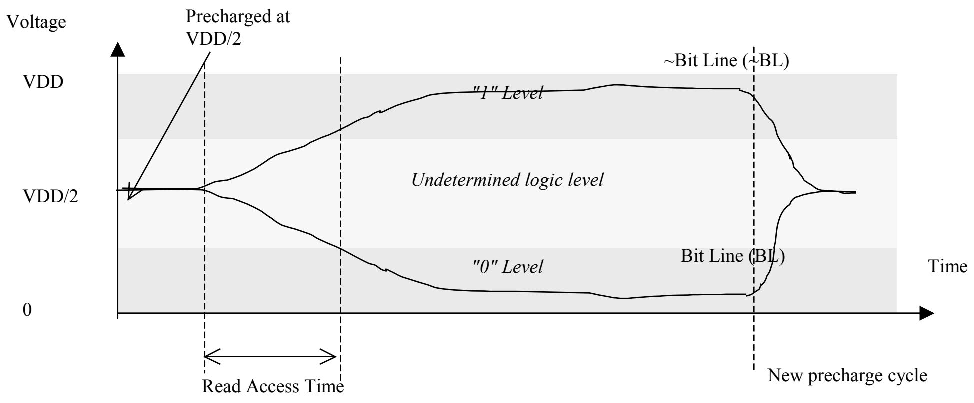


Figure 10-48: Read cycle using a precharge circuit

A simple precharge circuit consists of an n-channel MOS or p-channel MOS (Both switch the voltage VDD/2 without degradation). The drain is connected to VDD/2, the source to the bit line (Figure 10-49).

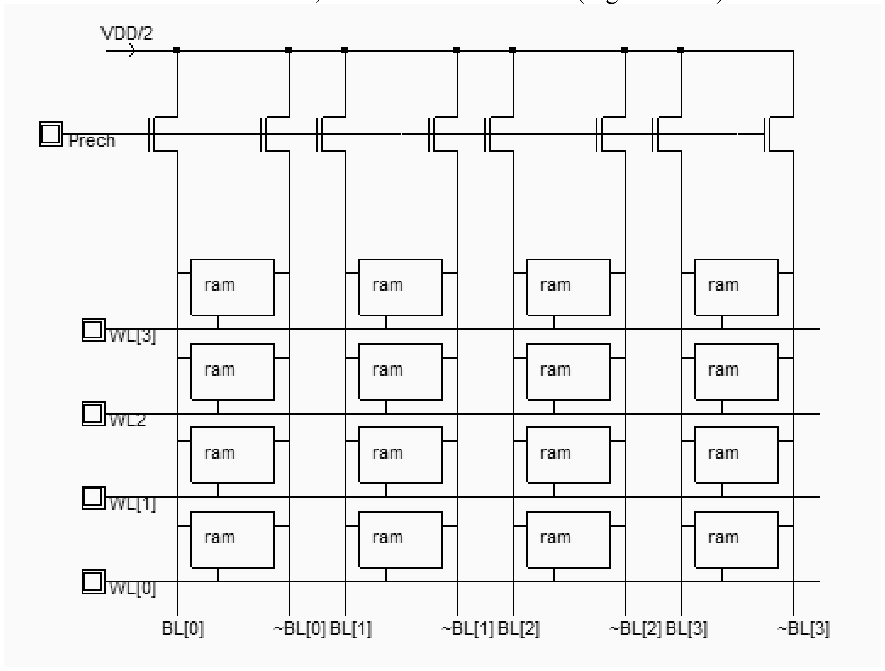
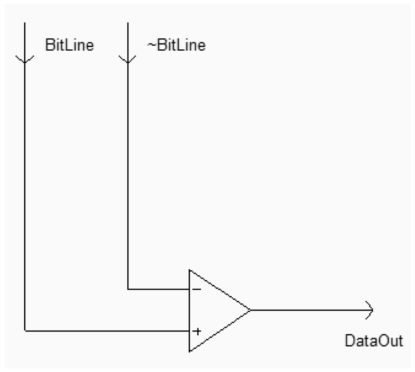


Figure 10-49: Connecting a precharge circuit to all bit lines (RamColumn.SCH)

Analog Amplifier

To further speed up the read process, analog amplifiers are used. The tiny difference is rapidly converted into a logic level, without waiting until *BL* and *~BL* reach their final voltage (Figure 10-50).



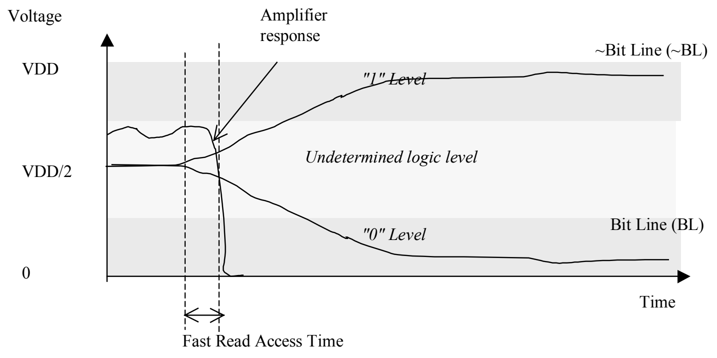


Figure 10-50: Shorter read access time thanks to a precharge circuit

The two commonly used operational amplifier designs are shown in figure 10-51. The first amplifier is a current mirror amplifier (See chapter 11 for more details on this circuit). When *Enable* is on, the *DataOut* signal saturates either to a low or high level, depending on the voltage difference V_{BL}-$V_{\sim BL}$. An alternative design for the operational amplifier is also proposed. The positive feedback in the amplifier, that is the cross-coupled link between *DataOut* and the pMOS device, permits faster sense operation than the basic circuit.

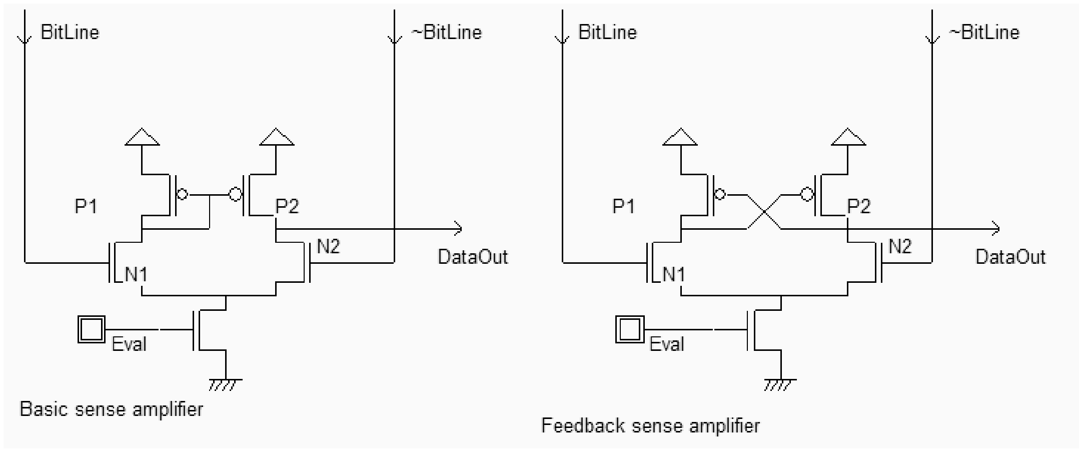


Figure 10-51: Very short read access time thanks to an operational amplifier (RamSenseAmpli.SCH)

The complete logic circuit including the *Write/Read* control, the precharge and the sense amplifier is shown in figure 10-52. When the precharge is active, all bit lines are charged to VDD/2, while all word lines are low. When the precharge is turned off, one of the world lines (*WL*) is active. A write operation (*Write/Read*=1) forces *BL* and *~BL* to the desired value given by *DataIn*. A read operation (*Write/Read*=0) turns the write buffers off and turns the sense amplifier on. The open-loop differential amplifier compares the value of *BL* and *~BL* and gives the logic result on *ReadData[0]*.

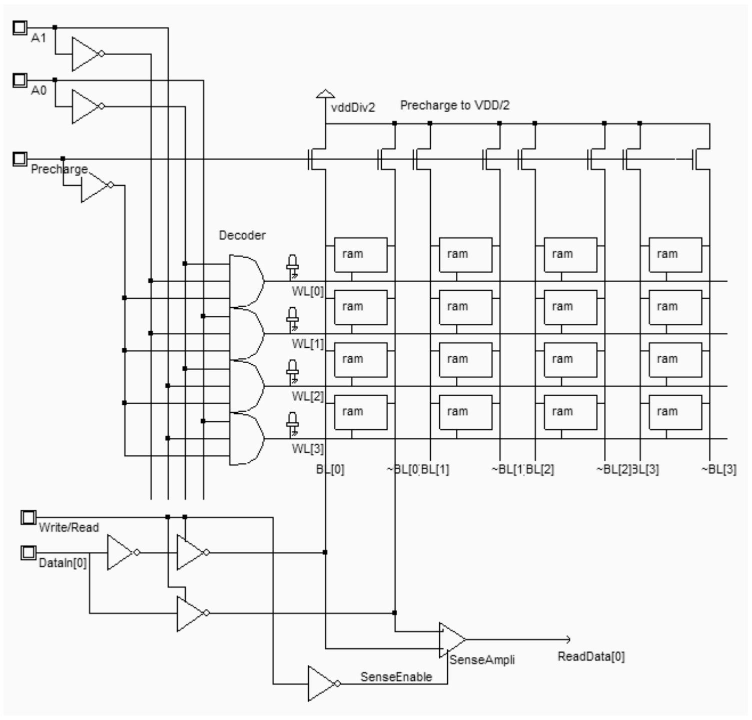


Figure 10-52: The sense amplifier used for read operation (Ram16Sense.SCH)

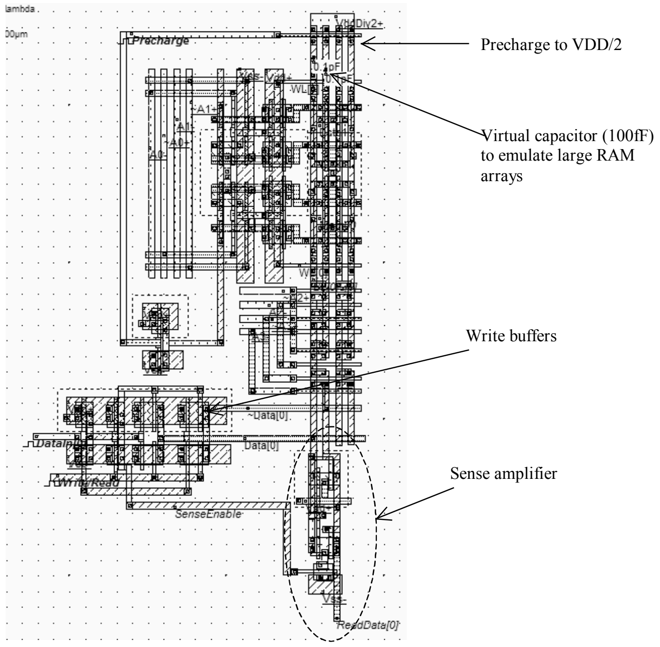


Figure 10-53: Layout of a portion of RAM with the control logic and the sense amplifier (RamSenseAmpli.MSK)

An implementation of the feedback sense amplifier is proposed in figure 10-54. The layout includes a portion of the 64 bit RAM, and the control logic. Parasitic loads corresponding to a 1Mb implementation are added to the vertical bit lines using a virtual capacitor of 0.3pF.

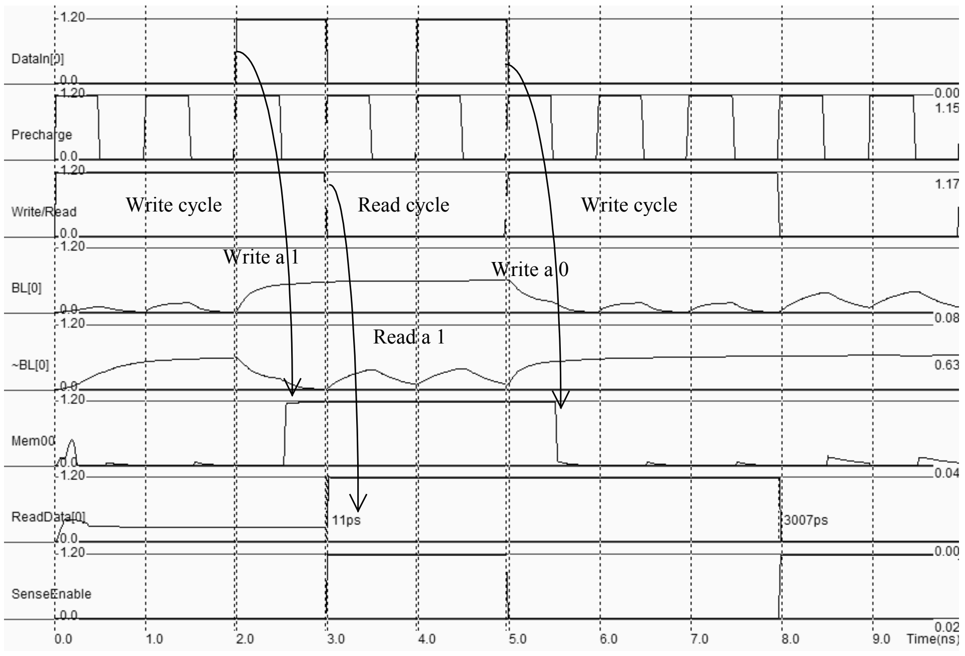


Figure 10-54: Simulation of the sense amplifier (RamSenseAmpli.MSK)

The simulation shown in figure 10-54 includes two *Write* and two *Read* cycles. The first write cycle (0-3ns) writes a 1 in the desired memory cell (*Mem00*). The reading operation (3-5ns) confirms that the memory value is "1" (*ReadData[0]*). The second write cycle (5-8ns) writes a 0 in the desired memory cell (*Mem00*). The reading operation (8-10ns) confirms that the memory value is "0" (*ReadData[0]*). Notice that the write operation (*Write/Read*=1) forces *BL* and *~BL* to the desired value given by *DataIn*. The precharge effect is clearly seen during the read operation.

Standby Current

The static power consumption of the embedded SRAM is quite high. The origin of this current is the leakage path in the active inverters, due to non negligible off currents illustrated in the schematic diagram Figure 10-55. In the 64 bit memory, the cumulated standby currents reach around 10µA. A direct extrapolation to a 1 mega-bit memory would lead to around 100mA, which is incompatible with lower power operations. In practice, Static RAM use low leakage MOS devices whenever possible, or even use MOS devices with enlarged channel length to limit the leakage current.

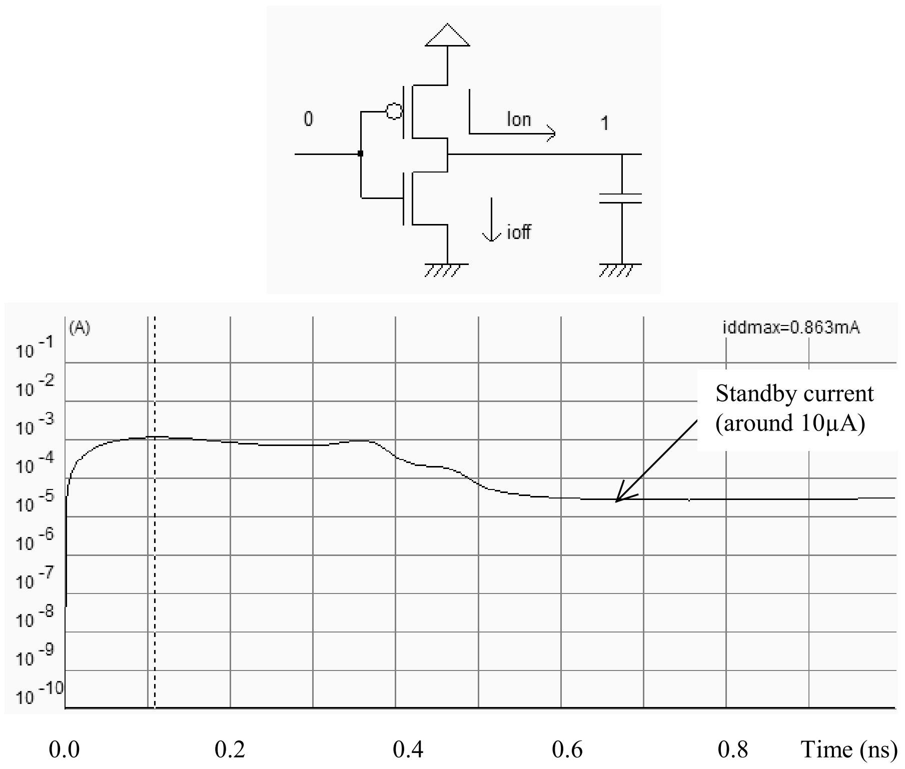


Figure 10-55: Evaluation of the standby current in a 64 bit SRAM memory

10.4 Dynamic RAM

The dynamic RAM memory has only one transistor, in order to improve the memory matrix density by almost one order of magnitude. The storage element is no longer the stable inverter loop, as for the static RAM, but only a capacitor *Cs*, also called the storage capacitor. The DRAM cell architecture is shown in figure 10-56.

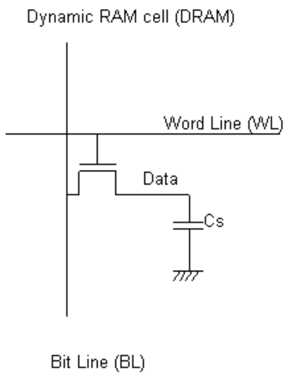


Figure 10-56: The 1 transistor dynamic RAM cell (RAM1T.SCH)

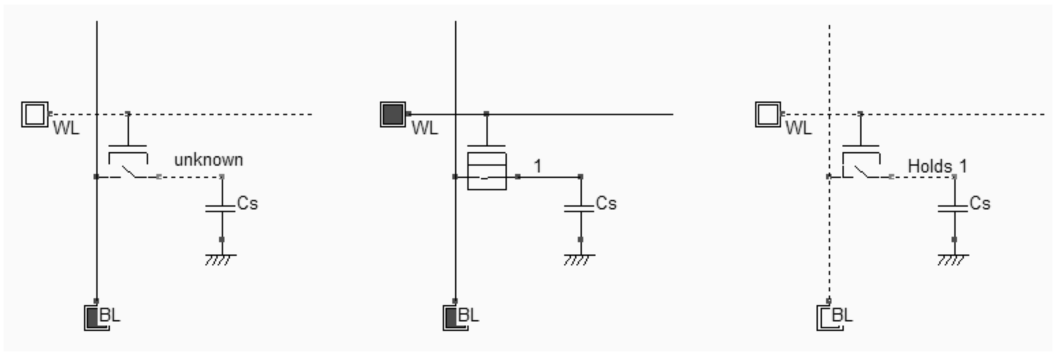


Figure 10-57: Simulation of the Write cycle for the 1 transistor dynamic RAM cell (RAM1T.SCH)

The write and hold operation for a "1" is shown in figure 10-57. The data is set on the bit line *BL*, then the word line *WL* is activated, and *Cs* is charged. As the pass transistor is n-type, the analog value reaches VDD-Vt. When WL is inactive, the storage capacitor *Cs* holds the "1".

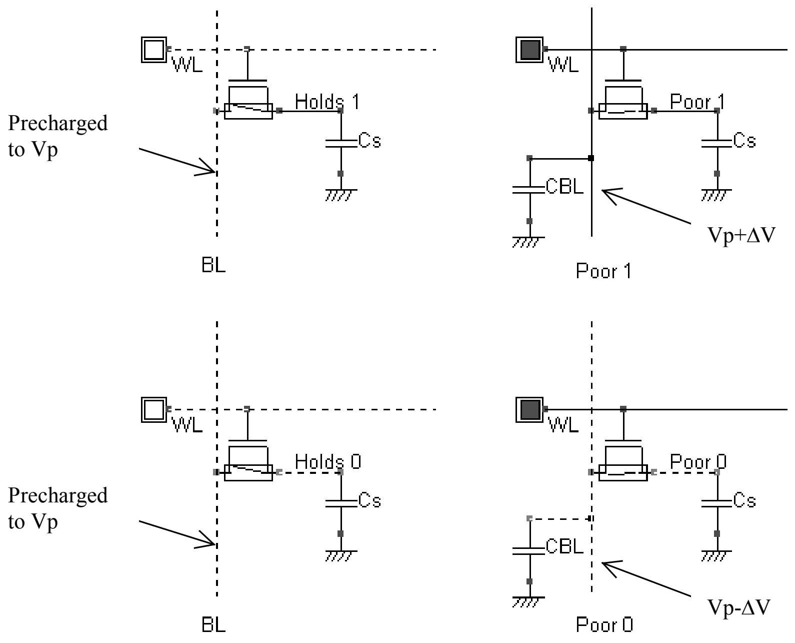


Figure 10-58: Simulation of the Read cycle for the 1 transistor dynamic RAM cell (RAM1T.SCH)

The reading cycle is destructive for the stored information. Suppose (Figure 10-58) that *Cs* holds a 1. The bit line is precharged to a voltage *Vp* (Usually around VDD/2). When the word line is active, a communication is established between the bit line, loaded by capacitor *CBL*, and the memory, loaded by capacitor *CS*. The charges are shared between these nodes, and the result is a small increase of the voltage *Vp* by *ΔV*, thanks to the injection of some charges from the memory. The formulation for this voltage difference can be evaluated in the case of a "1" stored in the memory by putting into equations the conservation of charges.

$$C_{BL}.V_p + C_s.Vdd = (C_{BL} + C_s)V_{final} \qquad \text{(Equ. 10-2)}$$

The left term of equation 10-2 corresponds to the sum of the charges stored in the bit line precharged at *Vp*, and the charges stored in the memory cell. The right term is the charge shared once the two capacitances (The bit line and the memory) are charged with the final voltage. The voltage increase can be rewritten as follows:

$$\Delta V = V_{final} - V_p = \frac{C_s}{C_{BL} + C_s}(V_{dd} - V_p) \qquad \text{(Equ. 10-3)}$$

Notice that *Cs* is usually much smaller than C_{BL}, meaning that ΔV is very small. Now, if the *Cs* was holding a zero, the activation of the word line would result in a small decrease of the precharge voltage, to *Vp -ΔV*.

In summary, the bit line voltage *Vp+ΔV* means that the memory state was 1, the voltage *Vp-ΔV* means that the memory state was 0. We say "was" because the memory information is destroyed by the read cycle. What the DRAM memory must do is to convert the +/-ΔV into 1/0, and rewrite the memory for a future read cycle.

DRAM Memory Cell

The DRAM memory cell should be as small as possible, but with the highest possible value for the storage capacitor *Cs*. The first idea, shown in figure 10-59, consists in using the parasitic junction capacitance as the storage capacitor *Cs*. The polysilicon gate is shared by all word lines in the same row, and the metal interconnect is shared by all bit lines in the same column. The capacitor *Cs* is around 0.1fF in 0.12µm technology. Notice that the bit line contact may be shared by two memory cells to improve the density.

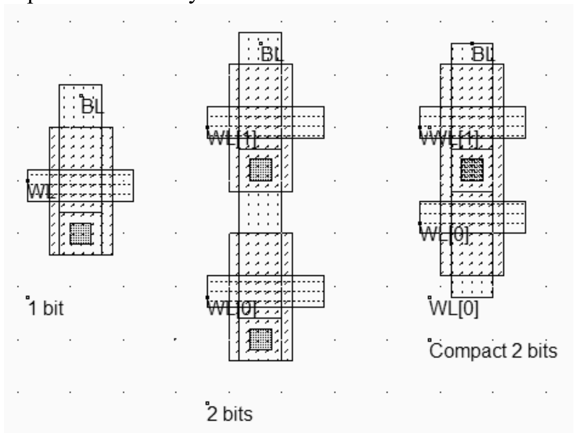
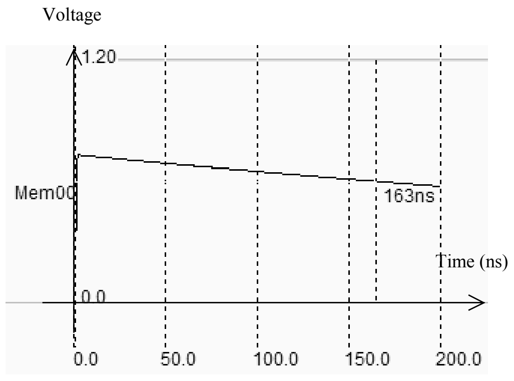


Figure 10-59: A DRAM memory design using parasitic junction capacitance (DramJunction.MSK)

There are two main problems with this design: first, the capacitance is very small, because the junction capacitance does not have a very high value, secondly, a leakage exists between the capacitor *Cs* and the bit line, through the access transistor, even when a zero voltage is applied on the bit line. Consequently, the charges stored in the capacitor tend to evacuate, partly to the ground through the parasitic diode (*Idiode*), partially to the bit line through the channel current *ioff*, which means that the memory information is retained only for less than one µsecond (Figure 10-60).



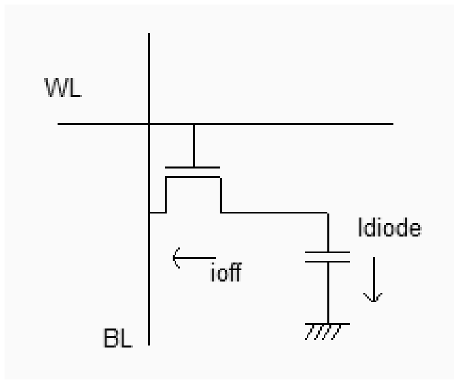
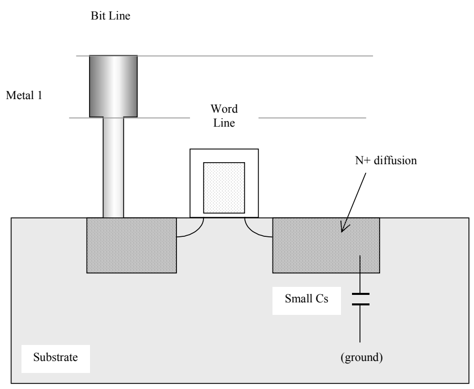


Figure 10-60: Leakage current in the dynamic RAM based on a junction capacitance (DramJunction.MSK)

The leakage current may be reduced by using low leakage MOS devices with non-minimal channel length, but the best technique is to increase by 2 or 3 orders of magnitude the storage capacitor. Commercial dynamic RAM memories use storage capacitors with a value between 10fF and 50fF. This is done by creating a specific capacitor for the storage node appearing in figure 10-61 left thanks to the following technological advances: the use of specific metal layers to create the lower plate and external walls of the RAM capacitor, an increased height between the substrate surface and metal1, and the use of high permittivity dielectric oxide. The silicon dioxide SiO2 has a relative permittivity ε_r of 3.9. Other oxides, compatible with the CMOS process have a higher permittivity (Higher "K") : Si_3N_4 with ε_r equal to 7, and Ta_2O_5 with ε_r equal to 23.



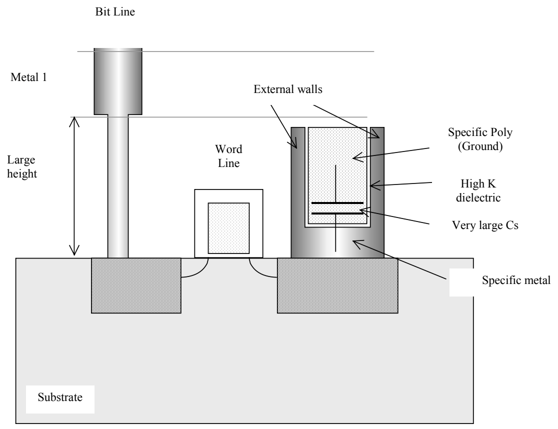
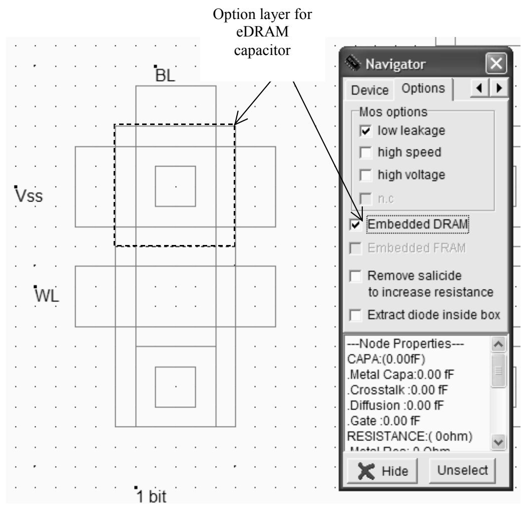


Figure 10-61: Increasing the storage capacitance (Left: junction capacitor, right, embedded capacitor)

The drawback of these methods is the addition of specific process steps to build the 3D capacitor, including delicate fabrication of high dielectric materials. The additional processing steps for the embedded DRAM represent approximately a 25% cost over the basic process. In Microwind, the high capacitance memory can be generated using the option layer, as shown in figure 10-62. The option layer is placed at the intersection between the n-diffusion area and the VSS metal line.



---Node Properties---
CAPA:(0.00fF)
.Metal Capa:0.00 fF
.Crosstalk :0.00 fF
.Diffusion :0.00 fF
.Gate :0.00 fF
RESISTANCE:(0ohm)

Figure 10-62: The option layer configured for the embedded capacitor (DramCell.MSK)

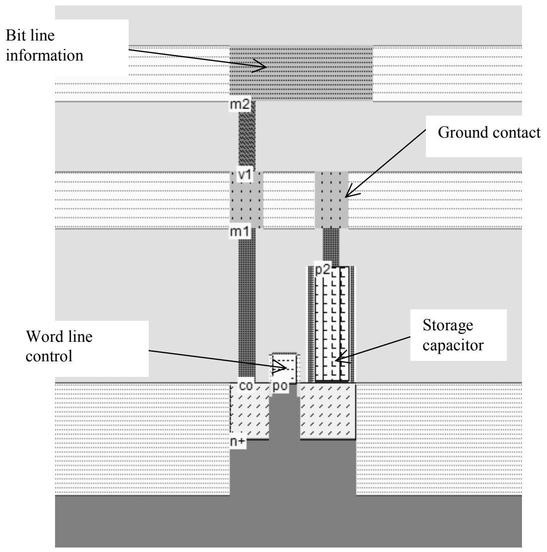


Figure 10-63: Cross-section of the DRAM cell with an embedded capacitor (DramCell.MSK)

The cross-section of the DRAM capacitor is given in figure 10-63. The bit line is routed in metal2, and is connected to the cell through a metal1 and diffusion contact. The word line is the polysilicon gate. On the right side, the storage capacitor is a sandwich of conductor material connected to the diffusion, a thin oxide (SiO2 in this case) and a second conductor that fills the capacitor and is connected to ground by a contact to the first level of metal. The capacitance is around 20fF in this design. Higher capacitance values may be obtained using larger option layer areas, at the price of a lower cell density. A DRAM array is shown in figure 10-64, together with a vertical cross-section at the capacitor location.

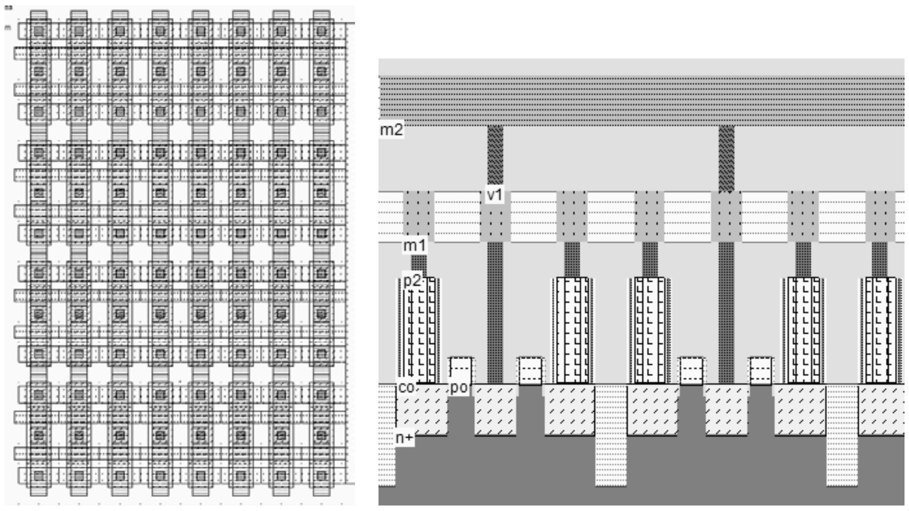


Figure 10-64: The stacked capacitor cell compared to the diffusion capacitor cell (DramEdram.MSK)

The charges stored in the capacitance of the dynamic RAM do not keep intact for very long, because of the leakage current (*Ioff*) of the pass transistor. However, the time during which the stored voltage may be considered as constant is more than two orders of magnitude higher for an embedded capacitor than for a simple diffusion.

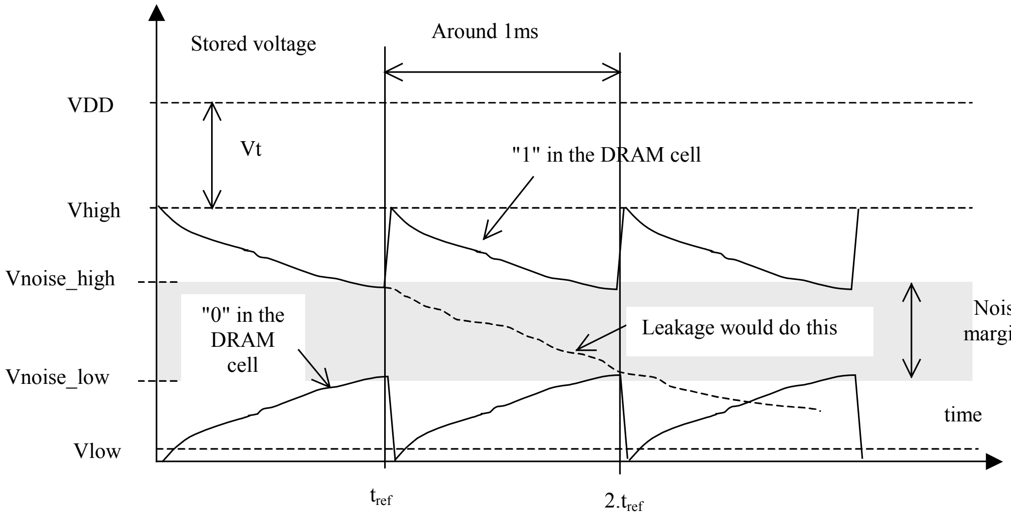


Figure 10-65: The refresh cycle for the DRAM

The capacitor discharge is the order of 1ms, as illustrated in figure 10-65. Before the memory voltage enters the undetermined zone (*Vnoise_high* when the memory contents was 1), the memory needs to be refreshed. The periodical refreshment of the memory is part of the DRAM memory control circuit.

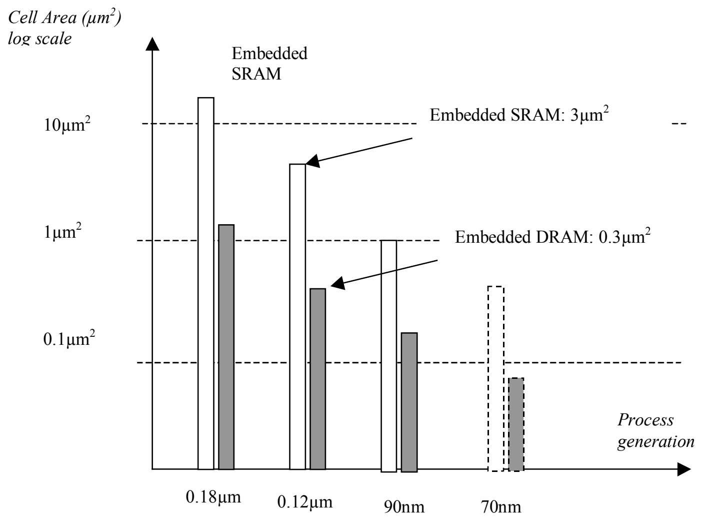
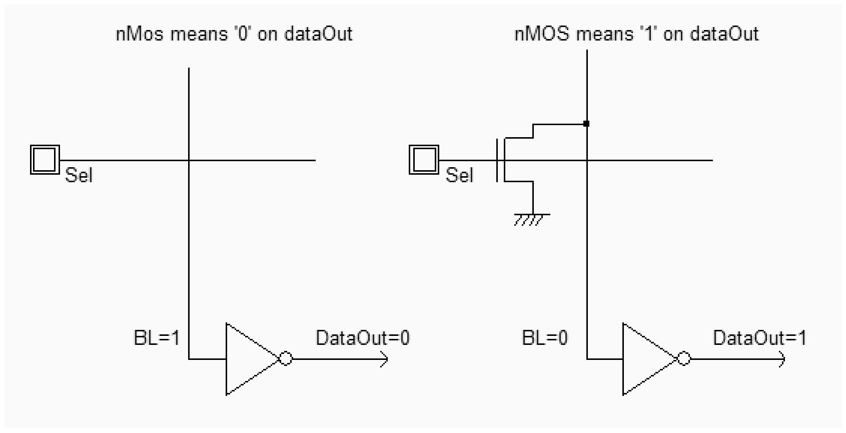


Figure 10-66: The DRAM size compared to the SRAM size

The size of the dynamic memory point is always significantly smaller than the size of the static memory point. This is of key advantage when large memory arrays are needed. However, the dynamic memories are slower to read because of the passive structure of the memory point. Static RAM memories are used for high speed data exchange while DRAM memories are used for mass memory storage.

10.5 ROM memory

The most simple non-volatile memory consists of a layout in which the data is permanently written through a specific layout arrangement and cannot be changed by the user once fabricated. The two logic states are shown in figure 10-67. The basic patterns for a "0" consists of an open circuit, while the "1" corresponds to the existence of a n-channel MOS. We assume that the vertical bit lines are precharged at 1 by default. In the case of an open circuit, the bit line remains at 1, and *dataOut* gives 0. If an nMOS device exists, the *BL* is tied to ground and *dataOut* gives 1. The logic programming is part of the fabrication process. ROM memories are used for storing microprocessor programs, mathematical values such as a sampled sinusoidal data for waveform generators, etc.. The size of ROM memories is usually small as compared to other embedded memories, because the contents cannot be changed.



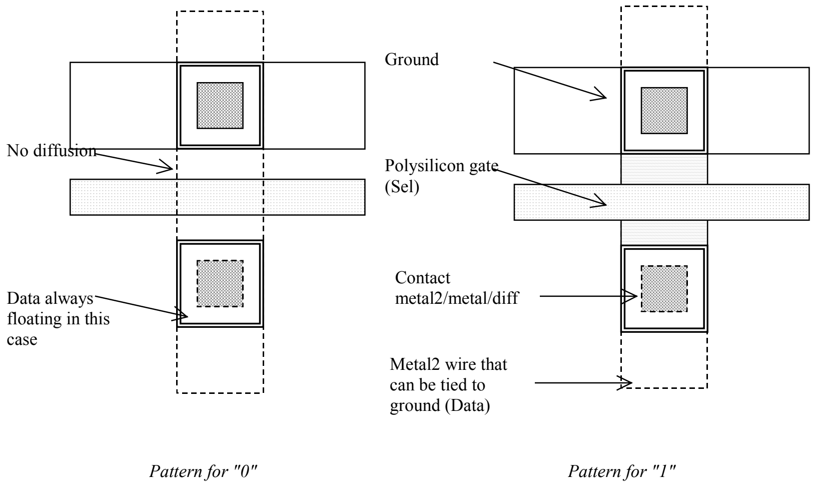

Ground
No diffusion
Polysilicon gate (Sel)
Contact metal2/metal/diff
Data always floating in this case
Metal2 wire that can be tied to ground (Data)
Pattern for "0"
Pattern for "1"

Figure 10-67 Changing the status of the memory point by a diffusion layer

We may create an array of ROM memory that stores the string "Hello!", which corresponds to the following binary data stream (Table 10-2).

Address	Character	ASCII code in hexadecimal	Equivalent in binary
00	" "	20	00100000
00	"h"	68	01101000
01	"e"	65	01100101
02	"l"	6C	01101100
03	"l"	6c	01101100
04	"o"	6f	01101111
05	"!"	21	00100001

Table 10-2 Encoding the string "Hello" in a ROM memory

The ROM architecture proposed in figure 10-68 is a NOR-like circuit [Haraszti]. The upper PMOS transistor precharges the vertical bit lines to VDD. Depending on the address, a high voltage is applied on one word line (*WL[1]* in the example), while all other word lines are kept at a low potential. The selected word line turns all programmed transistors on, which result in a discharge towards VSS, while bit lines with unprogrammed transistors remain on high voltage. The inverters situated on the lower part of the ROM array refresh and invert the bit line information which is sent to the display. Notice that the precharge effect is not simulated at logic level. When the precharge is off, all bit line nodes are considered in 3-state, without any consideration of a "low 3-state" level or "high 3-state" level which is taken into account in industrial logic simulators.

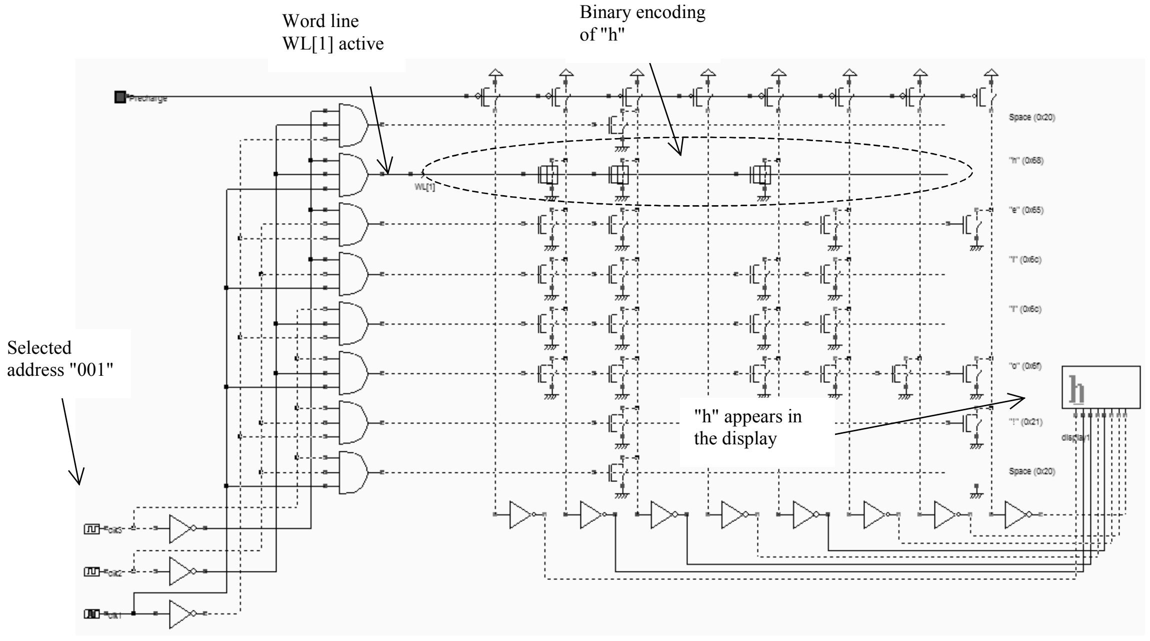
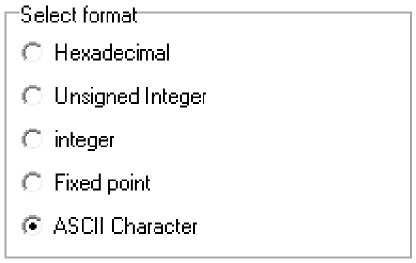


Figure 10-68 Delivering the contents of the ROM memory (Rom8x8.SCH)

In DSCH, the hexadecimal display has been configured to display the character corresponding to the ASCII input information (Figure 10-69).


Select format

- Hexadecimal
- Unsigned Integer
- integer
- Fixed point
- [✓ ASCII Character]

Figure 10-69 The hexadecimal display configured in ASCII mode (Rom8x8.SCH)

Layout considerations

The basic cells are shown in figure 10-70. The transistor situated on the left ("1") creates a low resistance path between the bit line and the ground when the word line is high. The layout corresponding to "0" is not a transistor, as the diffusion has been removed. The polysilicon ensures the continuity of the word line information, while the metal 2 ensures the continuity of the bit line information.

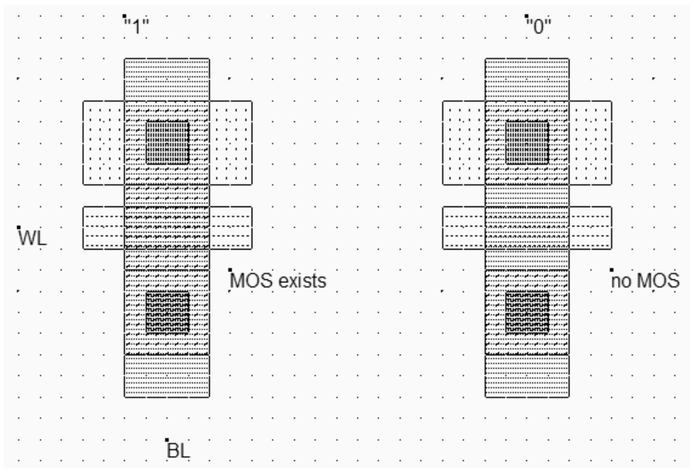
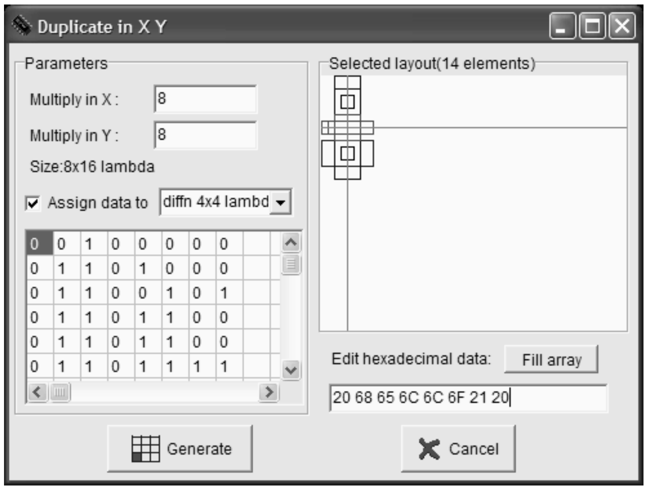


Figure 10-70 The basic patterns of the ROM memory (Rom.MSK)



Multiply in X : 8

Multiply in Y : 8

Size:8x16 lambda

☑ Assign data to diffn 4x4 lambd

0	0	1	0	0	0	0	0
0	1	1	0	1	0	0	0
0	1	1	0	0	1	0	1
0	1	1	0	1	1	0	0
0	1	1	0	1	1	0	0
0	1	1	0	1	1	1	1

Edit hexadecimal data: 20 68 65 6C 6C 6F 21 20

Figure 10-71 Programming the ROM memory during the duplication phase (Rom.MSK)

The duplication in X and Y can be programmed according to a binary information, through the menu shown in figure 10-71 (**Edit → Duplicate X,Y** in Microwind). The first step consists in giving the desired X and Y size, 8 in this example. Secondly, we choose the layout box that will be affected by the logic information. In our case, we select the n-diffusion box used for the channel. The selected layout box is marked by the cross in the layout window. Finally, we enter the desired information in hexadecimal format, and we click **Fill Array** to transform the string of data into elementary binary information. After a click on **Generate**, the regular ROM layout appears as shown in figure 10-72, which contains the binary version of the string " Hello! ".

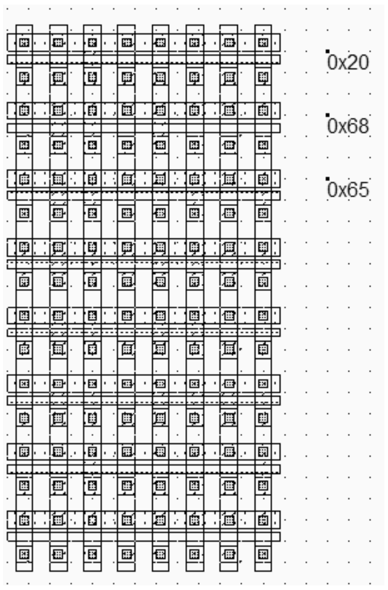


Figure 10-72 The ROM memory stores the string " Hello ! " in binary format (Rom.MSK)

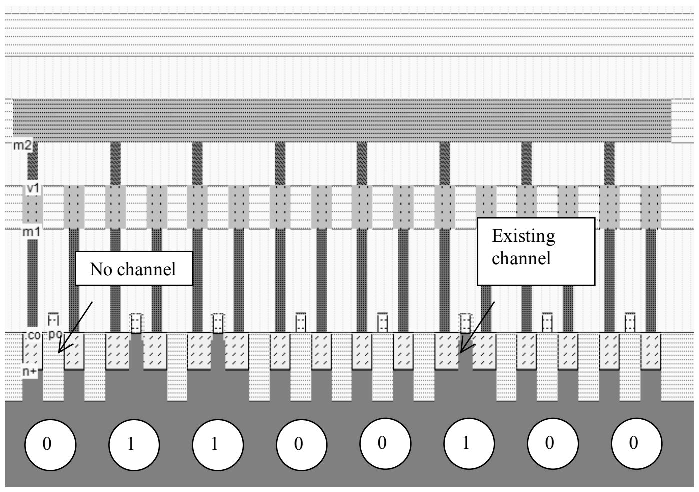


Figure 10-73 The ROM memory cross-section (Rom.MSK)

The cross-section (Figure 10-73) reveals the diffusion programming which creates or not the path to ground. If no channel is fabricated, the memory cell is equal to a zero. When the channel exists, the memory cell is equal to a one.

10.6 EEPROM memory

Double-Gate MOS

The basic element of an EEPROM (Electrically Erasable PROM) memory is the floating-gate transistor. The concept was introduced several years ago for the EPROM (Erasable PROM). It is based on the possibility of trapping electrons in an isolated polysilicon layer placed between the channel and the controlled gate. The charges have a direct impact on the threshold voltage of a double-gate device. When there is no charge in the floating gate (Figure 10-74, upper part), the threshold voltage is low, meaning that a significant current may flow between the source and the drain, if a high voltage is applied on the gate. However, the channel is small as compared to a regular MOS, and the Ion current is 3 to 5 times lower, for the same channel size.

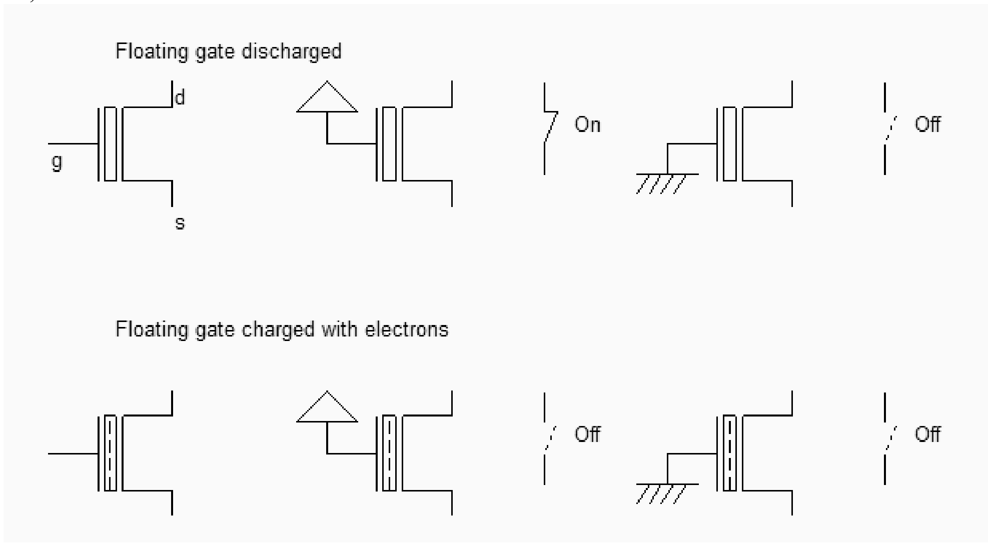
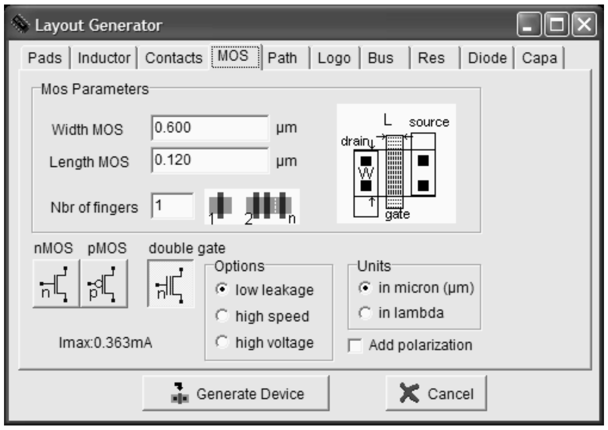


Fig. 10-74: The two states of the double gate MOS (EepromExplain.SCH)

When charges are trapped in the floating polysilicon layer (Figure 10-74, lower part), the threshold voltage is high, almost no current flows through the device, independently of the gate value. As a matter of fact, the electrons trapped in the floating gate prevent the creation of the channel by repealing channel electrons. Data retention is a key feature of EEPROM, as it must be guaranteed for a wide range of temperatures and operating conditions. Optimum electrical properties of the ultra thin gate oxide and inter-gate oxide are critical for data retention. The typical data retention of an EEPROM is 10 years.

Double-gate MOS Layout

The double gate MOS layout is shown in figure 10-75. The structure is very similar to the n-channel MOS device, except for the supplementary *poly2* layer on top of the polysilicon. The lower polysilicon is unconnected, resulting in a floating node. Only the *poly2* upper gate is connected to a metal layer through a *poly2/metal* contact situated on at the top. The cross-section of figure 10-76 reveals the stacked *poly/poly2* structure, with a thin oxide in between.



Layout Generator

Pads | Inductor | Contacts | MOS | Path | Logo | Bus | Res | Diode | Capa

Mos Parameters

Width MOS 0.600 µm

Length MOS 0.120 µm

Nbr of fingers 1

nMOS pMOS double gate

Options: [✓ low leakage] high speed high voltage

Units: [✓ in micron (µm)] in lambda

☐ Add polarization

Imax:0.363mA

Generate Device Cancel

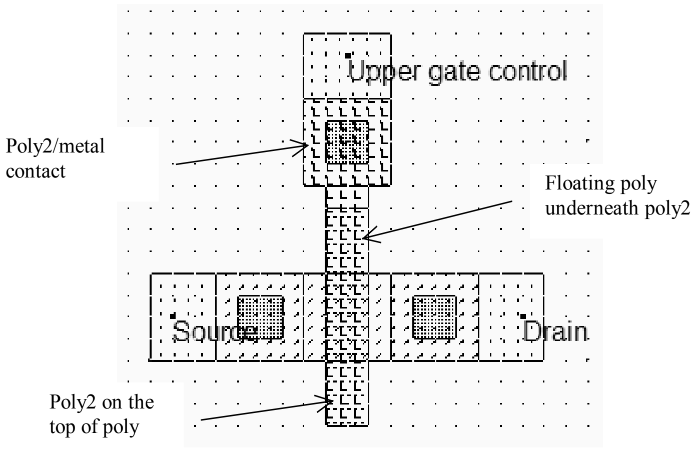


Fig. 10-75: The double gate MOS generated by Microwind (Eeprom.MSK)

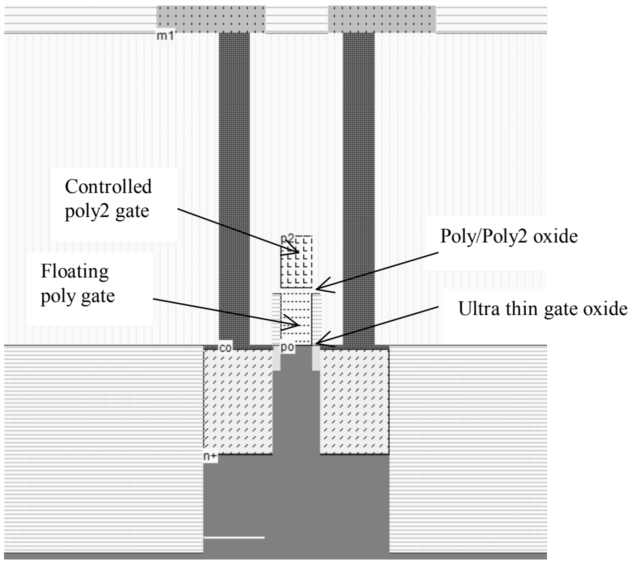
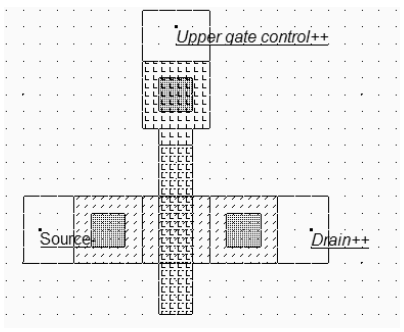


Fig. 10-76: Cross-section of the double gate MOS (Eeprom.MSK)

Double-gate MOS Charge

The programming of a double-poly transistor involves the transfer of electrons from the source to the floating gate through the thin oxide (Figure 10-78). Notice the high drain voltage (3V) which is necessary to transfer enough temperature to some electrons to become "hot" electrons, and the very high gate control to attract some of these hot electrons to the floating poly through the ultra thin gate oxide. The very high voltage varies from 7V to 12V, depending on the technology. Notice the "++" symbols attached to the upper gate and drain regions which indicate that a voltage higher than the nominal supply is used.



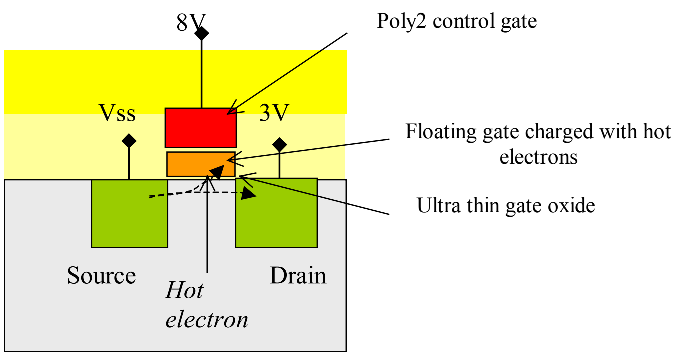
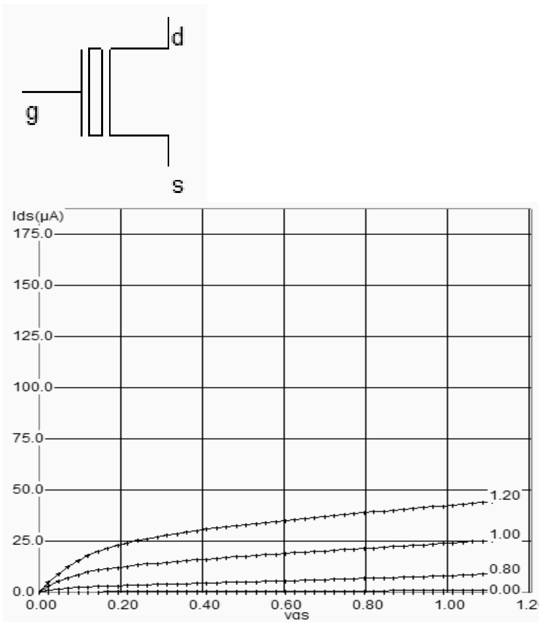
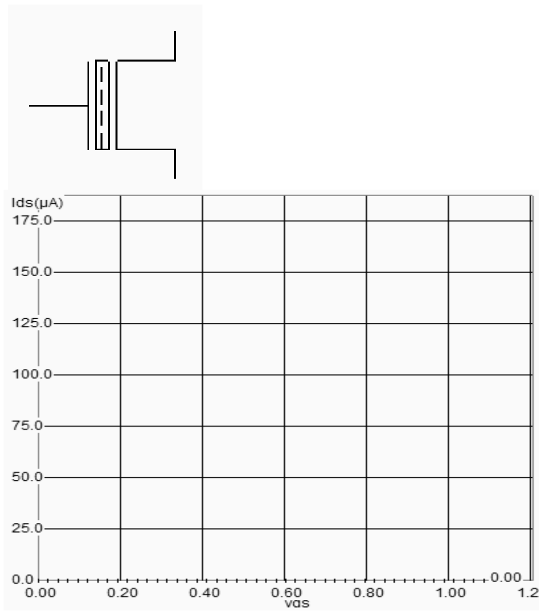


Fig. 10-78: The floating gate is charged with hot electrons thanks to a tunneling effect through the ultra-thin gate oxide (EepromCharge.MSK)



(a) Initial state: floating gate discharged



(b) Final state: floating gate charged

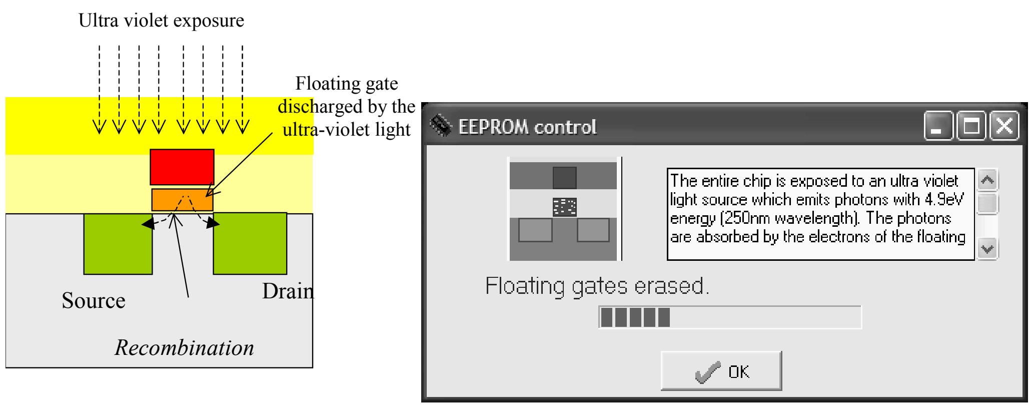
Fig. 10-79: Double-gate MOS characteristics without (a) and with charges (EepromCharge.MSK)

At initialization (Figure 10-79-a) no charge exists in the floating gate, resulting in a possibility of current when the poly2 gate voltage is high. However, the device is much less efficient than the standard n-channel MOS due to an indirect control of the channel. The maximum current is small but significant. The programming operation is performed using a very high gate voltage on poly2, here 8V.

The mechanism for electron transfer from the grounded source to the floating polysilicon gate, called tunneling, is a slow process. In Microwind, around 1000ns are required. With a sufficiently positive voltage on the poly2 gate, the voltage difference between poly and source is high enough to enable electrons to pass through the thin oxide. The electrons trapped on the floating gate increase the threshold voltage of the device, thus rapidly decreasing the channel current. When the gate is completely charged, no more current appears in the Id/Vd characteristics (Figure 10-79-b).

Double-gate MOS Discharge

The floating gate may be discharged by ultra violet light exposure or by electrical erasure. The U.V. technique is a heritage of the EPROM, which requires a specific package with a window to expose the memory bank to the specific light. The process is very slow (Around 20mn). After the U.V exposure, the threshold voltage of the double gate MOS returns to its low value, which enables the current to flow again (Figure 10-80).



The entire chip is exposed to an ultra violet light source which emits photons with 4.9eV energy (250nm wavelength). The photons are absorbed by the electrons of the floating

Floating gates erased.

Fig. 10-80: The floating gate may be discharged by U.V light exposure (EepromCharge.MSK)

In Microwind, the command **Simulate → U.V exposure to discharge floating gates** simulates the exposure of all double gate MOS to an ultra violet light source. At the end of the simulation (Which takes 10 seconds instead of 20 minutes in reality!), all floating gates of the layout are discharged. Alternatively, the charge contained in the floating gate can be accessed individually using the command **Simulate→Mos characteristics**. On the right lower corner of the device characteristics, a cursor named **Charge** appears, representing the amount of electrons stored in the floating gate. Changing the cursor position (Which corresponds in figure 10-81 to the minimum charge of electrons) modifies dynamically the MOS characteristics.

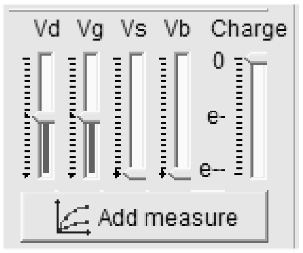


Fig. 10-81. Access to the double gate electron charge (EepromCharge.MSK)

For the electrical erase operation, the poly2 gate is grounded and a high voltage (Around 8V) is applied to the source. Electrons are pulled off the floating gate thanks to the high electrical field between the source and the floating gate. This charge transfer is called Fowler-Nordheim electron tunneling (Figure 10-82).

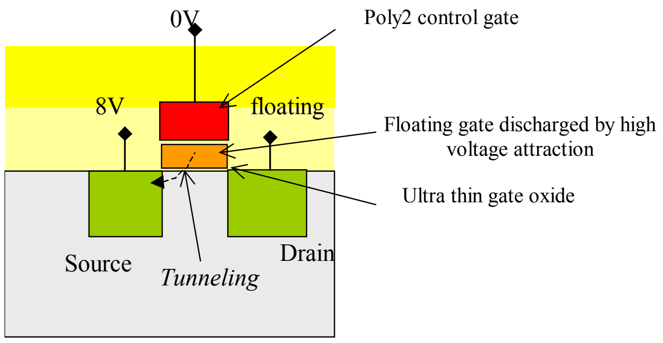


Fig. 10-82. Discharging the double gate MOS device (EepromDischarge.MSK)

EEPROM Array

EEPROM memories can be programmed electrically bit-by-bit and erased electrically bit-by-bit. EEPROM memory cells are based on one double-poly MOS devices, used for storage, and one n-channel MOS device for row selection (WL), as shown in figure 10-83.

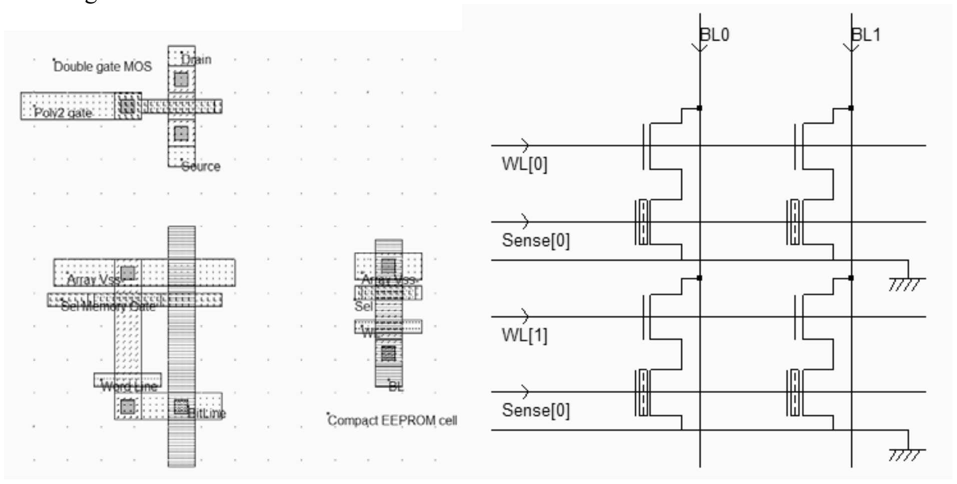


Fig. 10-83. Building an EEPROM memory cell (Eeprom.MSK)

The design requires two MOS devices per memory point, a memory transistor and a select transistor. This results in a large cell size, because of the two devices in series. A compact 2x2 arrangement is proposed in figure 10-83. The two devices are placed as close as possible (The usual design rule is 3 lambda), and the bit line and ground contacts are shared with other cells.

The basic structure for reading the EEPROM information is described in the schematic diagram of figure 10-84. After a precharge to VDD, and once *WL* is asserted, the bit line may either drop to VSS if the floating gate is empty of charges, or keep in a high voltage if the gate is charged, which disables the path between *BL* and the ground through the EEPROM device. In the case of figure 10-84 left, the floating gate has no charge, so *BL* is tied to ground after the precharge, meaning that *DataOut* is 1. The write operation consists in applying a very high voltage on the gate (8V), and to inject a high or low state on *BL*. A zero on *DataIn* is equivalent to a high voltage on *BL,* which provokes the hot electron effect and charges the floating gate. In contrast, a one on *DataIn* keeps *BL* low, and no current flows on the EEPROM channel. In that case, the floating gate remains discharged.

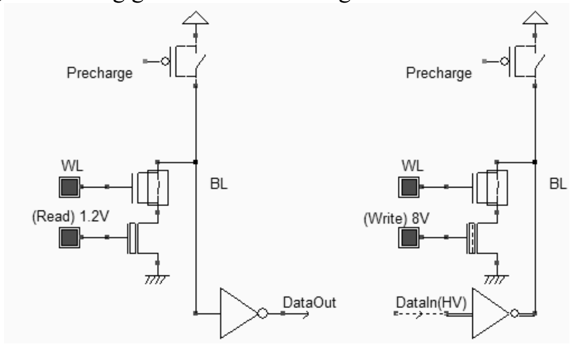


Fig. 10-84 Reading and writing in the EEPROM (Eeprom.MSK)

The 8x8 bit array using this basic EEPROM memory cell is reported in figure 10-84. The need for two devices per memory cell means a large memory array, and a relatively low level of integration.

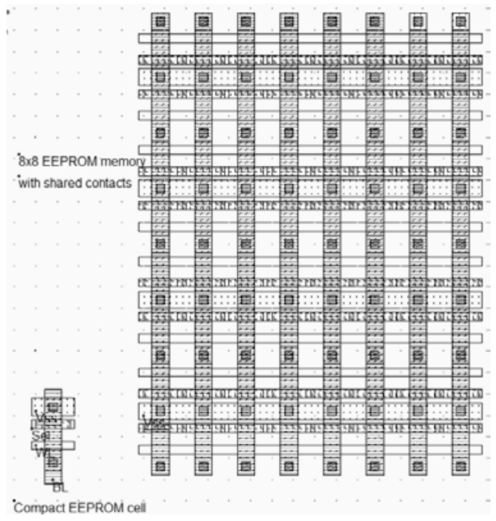


Fig. 10-84. The EEPROM memory array (Eeprom8x8.MSK)

10.7 Flash Memories

Flash memories are a variation of EEPROM memories. Flash arrays can be programmed electrically bit-by-bit but can only be erased by blocks. Flash memories are based on a single double poly MOS device, without any selection transistor (Figure 10-85). The immediate consequence is a more simple design, which leads to a more compact memory array and more dense structures. Flash memories are commonly used in micro-controllers for the storage of application code, which gives the advantage of non volatile memories and the possibility of reconfiguring and updating the code many times.

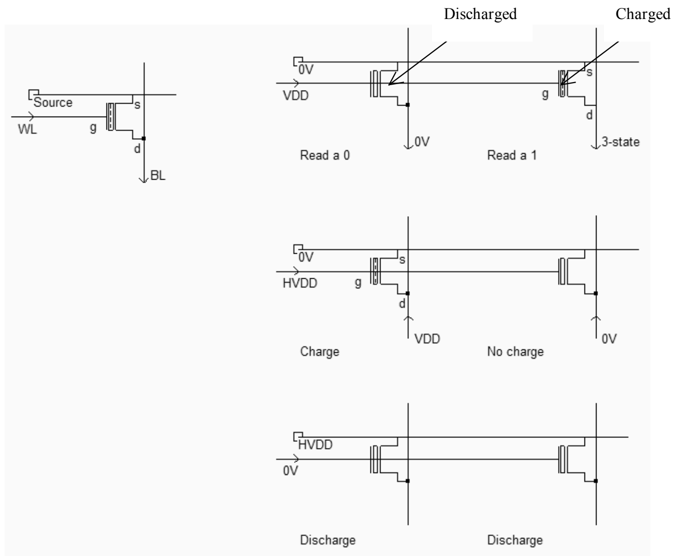


Fig. 10-85. The flash memory point and the principles for charge/discharge (FlashMemory.SCH)

The main characteristics of the Flash memory are given in figure 10-85. Assuming that the floating gate may be charged or discharged, the reading operation consists in applying a VDD voltage on the control gate, and a ground on the source (WL). The bit line drops to 0 if the gate is discharged, or remains in high impedance if the gate is charged. The charge is selective as it depends on the applied information on the vertical bit line: a VDD value provokes a charge injection, while a VSS value disables hot electron effect. The charge effect requires a high voltage HVDD on the control gates. Finally, the discharge is common to all double-gate MOS devices as soon as a high voltage HVDD is applied to the source. This is the main difference with EEPROM cells, where the high voltage was applied or not to the double-gate device, depending on the pass transistor.

Flash Memory layout

The Flash memory point usually has a "T-shape", due to an increased size of the source for optimum tunneling effect [Sharma]. The horizontal polysilicon2 is the bit line, the vertical metal2 is the word line which links all drain regions together. The horizontal metal line links all sources together.

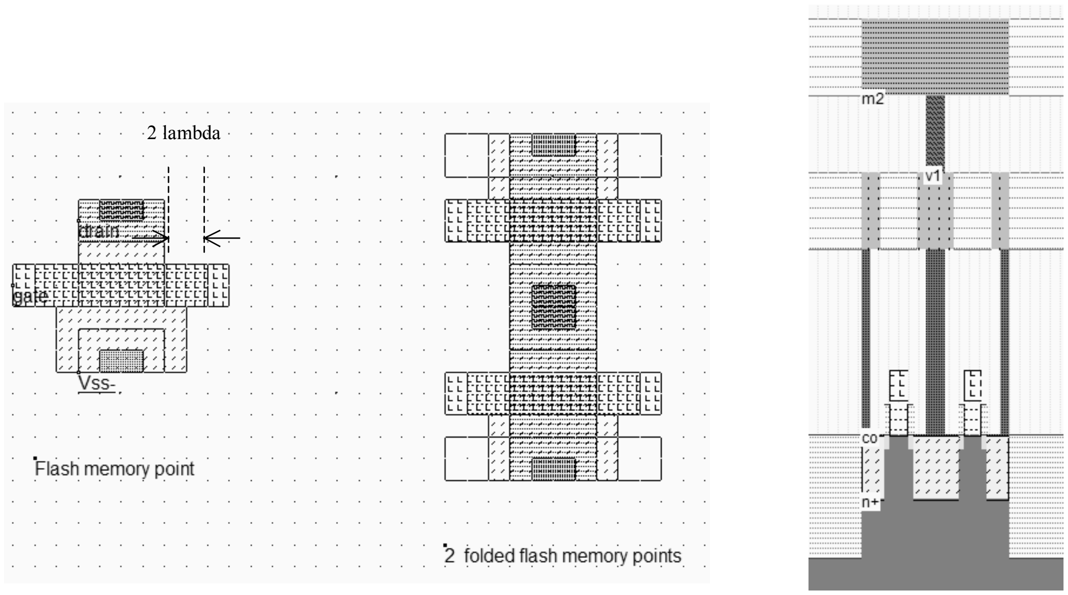


Fig. 10-86. The flash memory point and the associated cross-section (Flash8x8.MSK)

It is not an uncommon practice to violate usual design rules, in order to achieve more compact layout. In the case of figure 10-86, the poly extension is reduced from 3 lambda to 2 lambda. The cell density is increased, but the probability of an erroneous MOS behavior is also increased. Such a compromise between integration and reliability is justified by yield analysis with and without complying to design rules. This kind of information is confidentially kept within the IC company. An example of 8x8 bit Flash memory array is shown in figure 10-87.

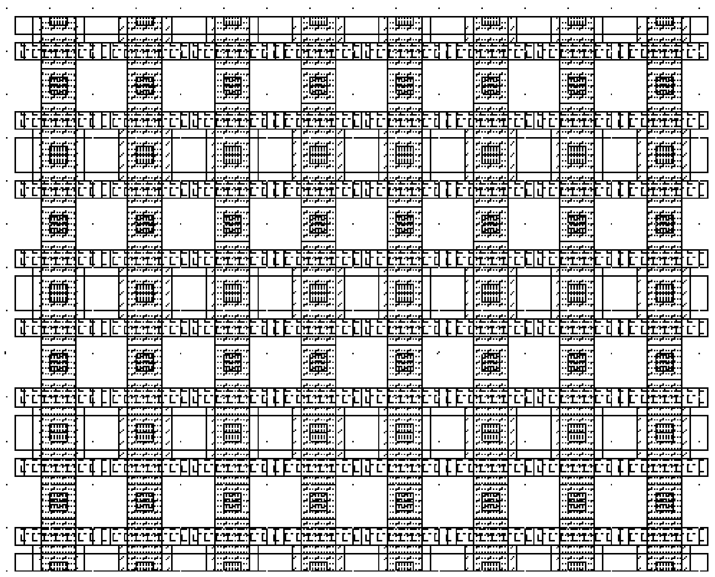


Fig. 10-87. The flash memory bank consisting of 8x8 memory cells (Flash8x8.MSK)

10.8 Ferro-electric RAM

Ferroelectric RAM memories are the most advanced of the Flash memory challengers [Geppert]. The FRAM is exactly like the DRAM except that the FRAM memory point is based on a two-state ferroelectric insulator, while the DRAM relies on a silicon dioxide capacitor. Mega-bit FRAM are already available as stand alone products. However, FRAM embedded memories have been made compatible since the 90nm CMOS technology. The Microwind software should first be configured in 90nm to access the FRAM properties using the command **File → Select Foundry**.

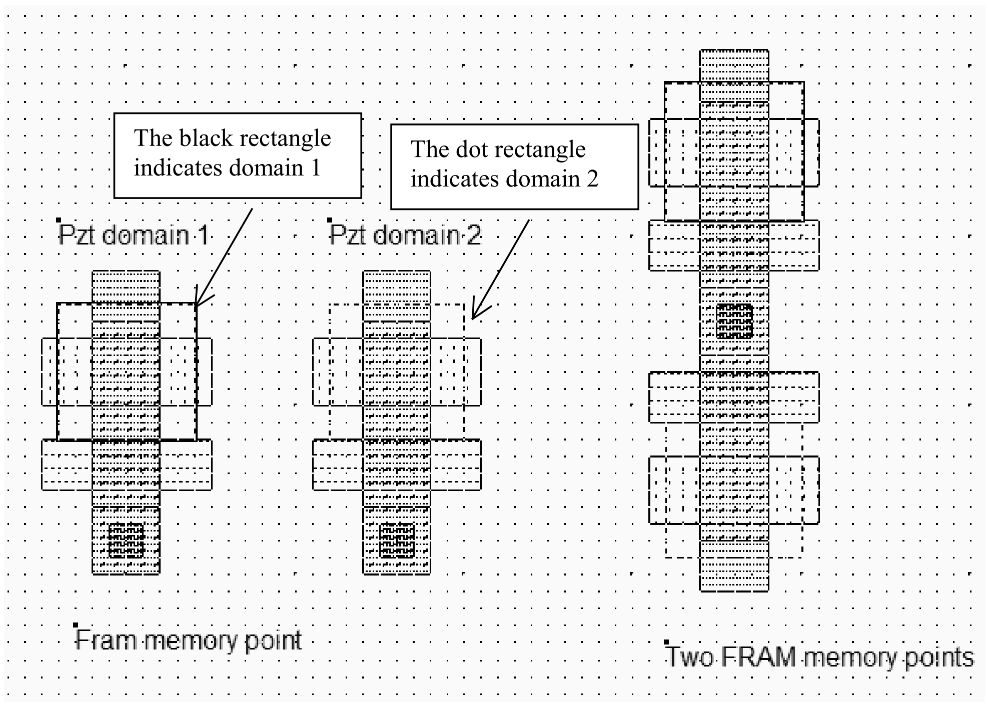

The black rectangle indicates domain 1
The dot rectangle indicates domain 2
Pzt domain 1
Pzt domain 2
Fram memory point
Two FRAM memory points

Fig. 10-88. The two domains of the FRAM memory (FramCell.MSK)

The 2D cross-section (Figure 10-89) shows the ferroelectric crystalline material made from a compound of lead, zirconium and titanium (PZT). The chemical formulation of PZT is $PbZr_{1-x}Ti_xO3$. Adjusting the proportion of zirconium and titanium changes the electrical properties of the material.

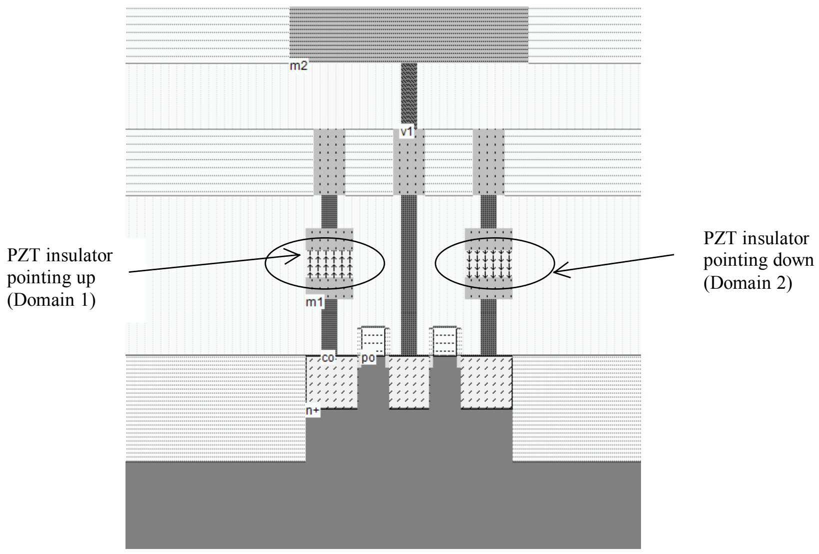

m2
v1
m1
co
po
n+
PZT insulator pointing up (Domain 1)
PZT insulator pointing down (Domain 2)

Fig. 10-89. The two domains of the FRAM memory (FramCell.MSK)

We describe here the PbTiO3 material, which is a ferroelectric insulator with a dielectric constant higher than 100, and has two stable positions, called *domains*. In the first domain, the molecular dipoles point up, in the second domain, the molecular dipoles point down. This property is handled by Microwind using arrows in the insulator material, as shown in figure 10-89. The PbTiO3 insulator is in between two metal electrodes, fabricated in Iridium.

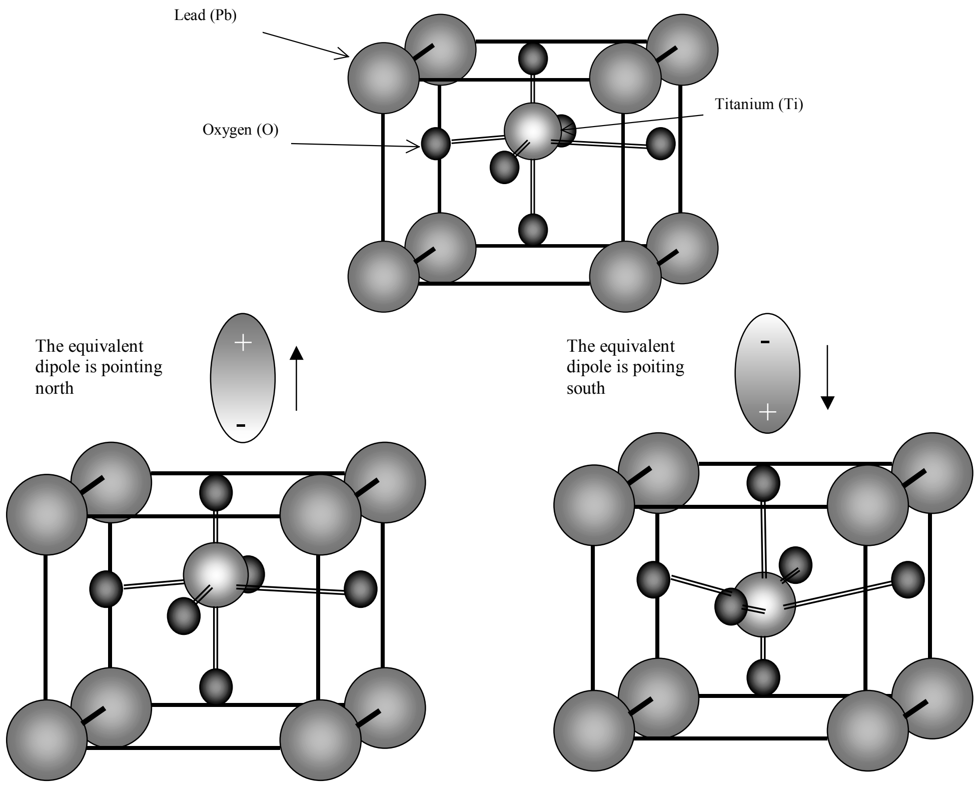

Lead (Pb)
Oxygen (O)
Titanium (Ti)
The equivalent dipole is pointing north
+
-
The equivalent dipole is poiting south
-
+

Fig. 10-90. The two domains of the structure which change the orientation of the equivalent dipole

The PbTiO3 molecular structure is given in figure 10-90. It is equivalent to a cube, where each of the eight corners is an atom of lead (Pb). In the center of the cube is a titanium atom, which is a class IVb element (See the table of elements in chapter 1), with oxygen atoms at its ends, shared with neighbors. The two stable states of the molecule are shown in figure 10-90. The titanium atom may be moved inside the cell by applying an electrical field. The remarkable properties of this insulator material are: the stable state of the titanium atom even without any electrical field, the low electrical field required to move the atom, and its very high dielectric constant (Around 100).

The PZT capacitor behavior is usually represented by the hysteresis curve shown in figure 10-91. In the X Axis, the electrical field applied to the electrodes is displayed. The Y axis represents the dipole orientation for each molecule. It can be seen that if a minimum field is applied on the capacitor, the polarization changes. An inverted electrical field is required to change the state of the material.

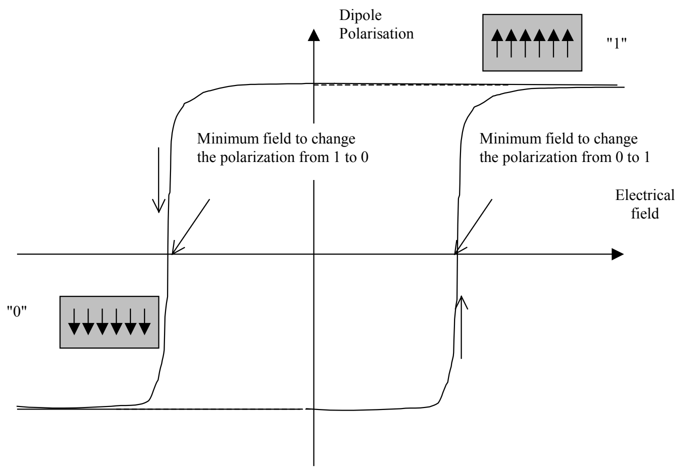


Fig. 10-91. The hysteresis curve of the PZT insulator

Consequently, the write cycle simply consists, for a 1, in applying a large positive step on which orients the dipoles north, and for a zero in applying a negative voltage step, which orients the dipoles south (Figure 10-92).

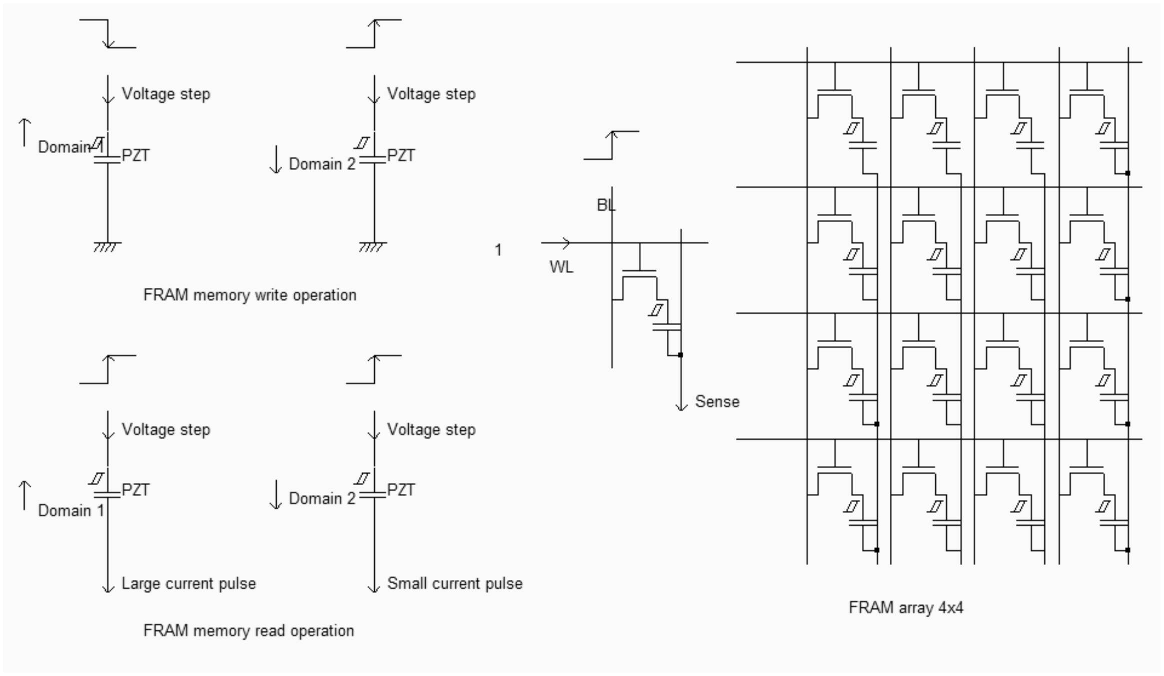


Fig. 10-92. The FRAM circuit principles and architecture (Fram4x4.SCH)

To read the domain information, an electrical field is applied to the PZT capacitor, through a voltage pulse. If the electric field is oriented in the opposite direction of the elementary dipole and is strong enough, the inner atom orientation is changed, which creates a significant current, which is amplified and considered as a 1 (Figure 10-92). If the electric field is oriented in the same direction as the elementary dipole, only a small current pulse is observed which is considered as a 0. Reading the logical information is equivalent to observing the current peak and deciding whether the current peak is small (0), or large (1). Notice that the read operation destroys the data stored in the PZT material, as for the DRAM. Just after the memory information is read, the logic information must be written back to the memory cell.

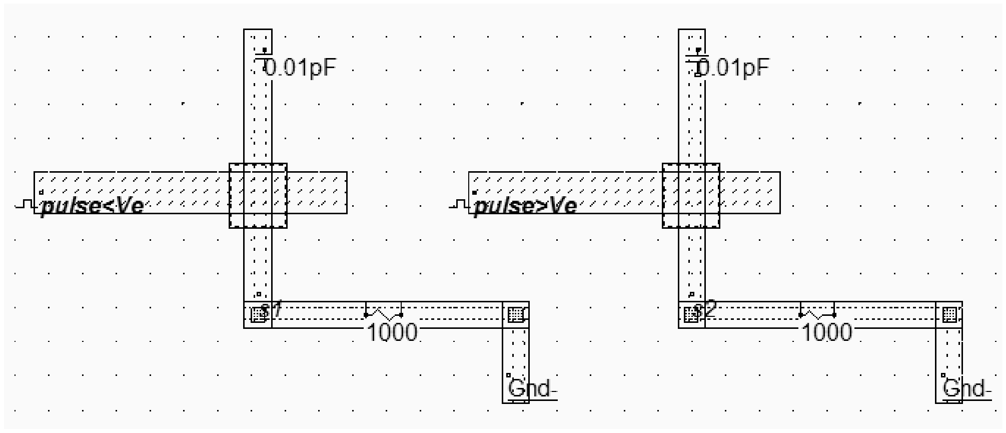


Fig. 10-93. Layout of the two PZT capacitor with initial domains oriented down (FramSimu.MSK)

In figure 10-93, two PZT capacitors are designed at the interface of metal1 and n-diffusion. The programming of the domain may be done by a double click in the option layer, and a tick in the "Embedded FRAM" option. Up-to-down and down-to-up domains are accessible.

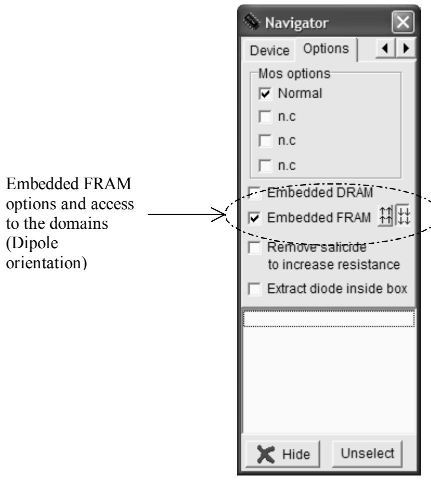


Fig. 10-94. Manual access to the domains of the ferroelectric material used in the FRAM (FramSimu.MSK)

The PZT capacitor domains are oriented down at initialization. A sufficient electrical field, created by a high voltage difference between the diffusion and the metal gives the conditions for domain change (Figure 10-95).

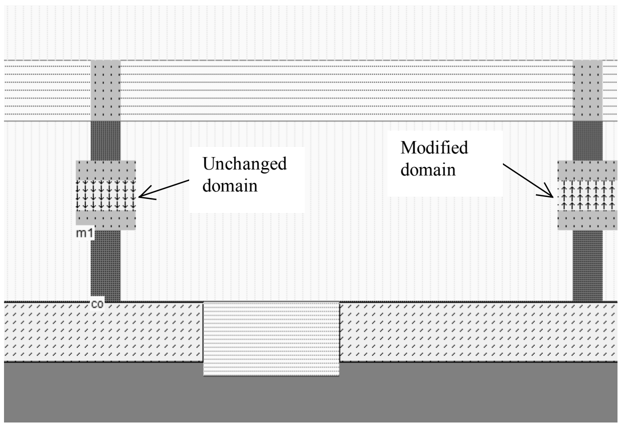


Fig. 10-95. A sufficient voltage provokes the domain change on the PZT capacitor situated on the right side (FramSimu.MSK)

The simulation must be performed using the option **With crosstalk**, which is user-accessible in the simulation menu. Consequently, the PZT capacitor coupling is taken into account. At a sufficient voltage difference *Ve* between the two plates (*Ve* is 600mV in this case), the domain is changed. This is the case on the right side as the voltage difference reaches almost 1V, but it is not the case on the left side, where the diffusion rises to a lower voltage. The final state is shown in the 2D cross-section of figure 10-96.

The ferroelectric RAM has the potential of becoming the ideal non-volatile memory for a wide range of applications, as it only requires two additional masks (Instead of 4-6 for embedded DRAM with stacked capacitor). The FRAM module has a cell size two to three times smaller than the static RAM memory cell, with the advantage of a zero standby power consumption. However, the read/write cycles are not unlimited, due to fatigue in the PZT material. Heavy read/write cycles could still be supported by SRAM memories.

10.9 Classification

A summary of CMOS embedded memory performances is given in table 10-2. The typical memory bank capacity gives an advantage to the ROM, EPROM, EEPROM and Flash memories, because of a small cell area. The reading and writing performances vary very significantly, as well as the retention of data. Dynamic RAM (DRAM) are slow but compact. Static RAM (SRAM) are fast but large. Reading the information from a passive capacitor, such as in DRAM, is much slower than reading the information from the active inverter-based memory such as in SRAM. In contrast, a single trench of stacked capacitor requires much less silicon surface than a 2-inverter memory structure, at the cost of 8 supplementary process steps. The FLASH memories combine a small area, an acceptable reading cycle and interesting non-volatile capabilities, at the price of a slow writing process (1µs). Promising performances are achieved by ferroelectric RAM (FRAM) which are the most advanced of non-volatile challengers. FRAM have endurance writing/erasing cycles comparable to the best memories, with fast reading and writing cycles, and require only two additional process masks.

Memory type	Typical Capacity	Cell area	Reading	Writing	Cycles	Retention	Process complexity	High voltage
ROM	32Mb	Very small	Medium	Impossible	0	No limit	0	no
EPROM	16Mb	Very small	Slow	Extremely slow	1-10	>30 YEARS	3	yes
E2PROM	1Mb	Large	Slow	Very slow	1E5-1E7	>10 YEARS	4	no
FLASH	16Mb	Very small	Medium	Very slow	1E4-1E5	>10 YEARS	4	yes
FRAM	4Mb	Small	Fast	Fast	1E12-1E15	>10 YEARS	2	No
eDRAM	32Mb	Small	Slow	Fast	>1E15	Volatile, needs to refresh	8	no
SRAM	4Mb	Large	Very fast	Very fast	>1E15	Volatile	0	No

Table 10-2: A classification of embedded memories according to their performances

10.10 Interfacing

The signals used in old-style RAM memory circuits are shown in figure 10-96. The memory is enabled by *Chip Enable* signal (*CE*), usually active low. When *CE* is high, the memory is in stand-by mode, usually consuming the minimum possible power. When active (*CE*=0), the memory is controlled by the signals signal *Read/write* (*WE*), *Row Address Selection* (*RAS*) and *Column Address selection* (*CAS*). The output is enabled on the n-bit Data bus thanks to *Output Enable* (*OE*).

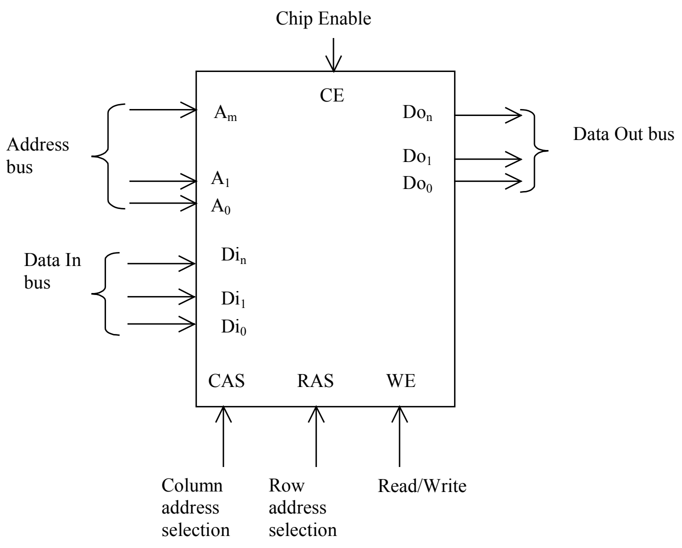


Figure 10-96: Typical RAM interfacing

Read & Write Cycles

The operating conditions of RAM memories involve a specific timing of control signal and data, as shown in figure 10-97. We consider here an address bus where the row and column address information is available with a time domain multiplexing. The row address selection (RAS) and Column address selection (CAS) provide the demultiplexor signals for the address bus. The total read cycle duration is t_{RC}. The delay between the active edge of the row address selection (RAS) and the valid data out is called the row access cycle t_{RAC}.

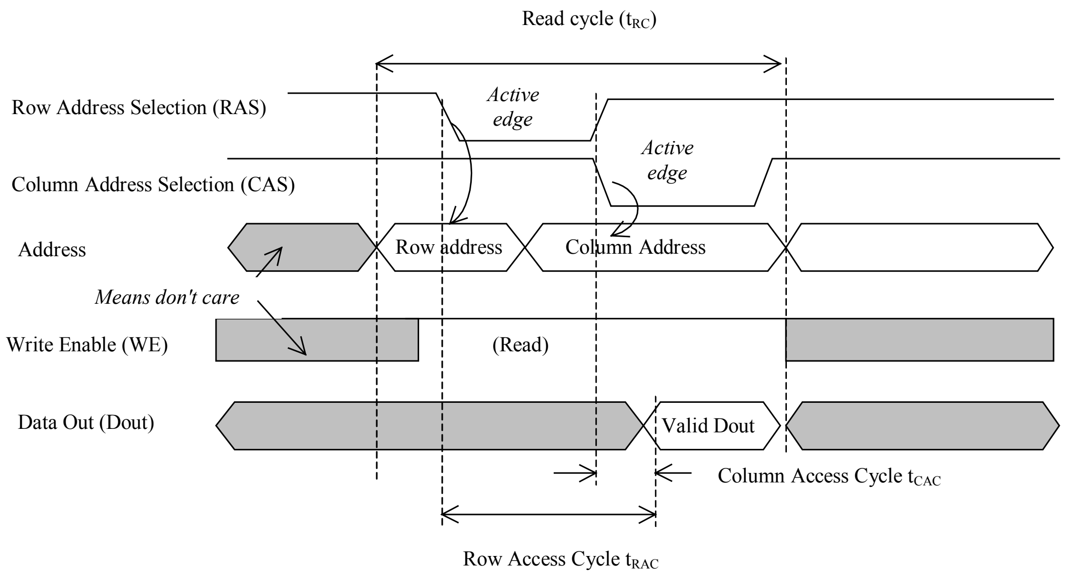


Figure 10-97: RAM timing parameters (Read cyle)

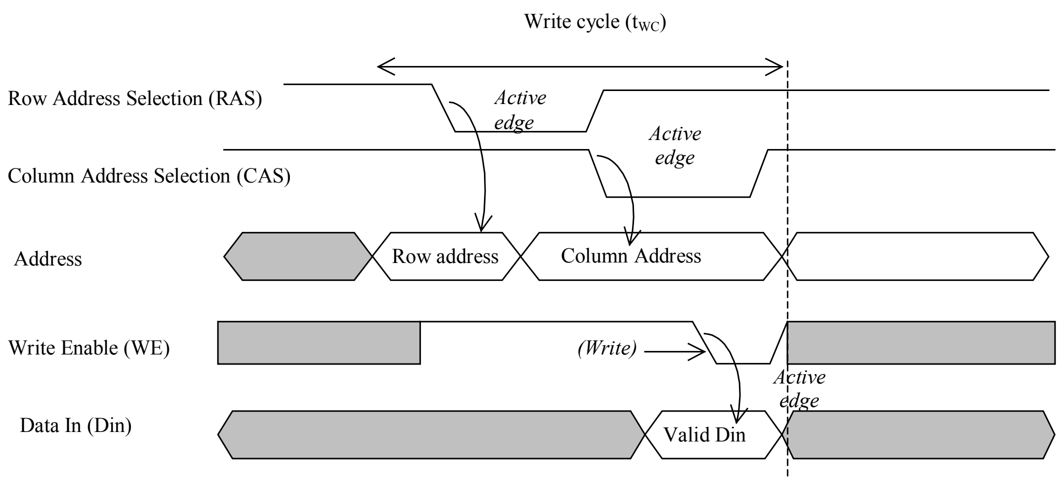


Figure 10-98: RAM timing parameters (Write cyle)

The total write cycle duration is t_{WC} (Figure 10-98). The switching performances differ considerably whether the memory is a stand-alone integrated circuit or an embedded block. Typically the read and write cycle time is 10-20ns for a stand-alone product, but around 2-5 ns for an embedded memory. This very important gain in performance is due to the short and low-capacitance interconnection between the processor and the memory, with very simple and fast block interfacing, as compared to the connection between one die and an other die through the packaging and printed circuit board.

Synchronous RAM

The synchronous RAM differs from the conventional dynamic RAM in two main points [Sharma]: all inputs and outputs are synchronized to the rise edge of the clock, and more than one word can be read or written in sequence. The typical chronograms of a synchronous RAM are shown in figure 10-99. The active edge of the clock is usually the rise edge. One read cycle includes 3 active clock edges in the example shown in figure 10-99. The row address selection is active at the first rise edge, followed by the column address selection. The data is valid at the third fall edge of the system clock.

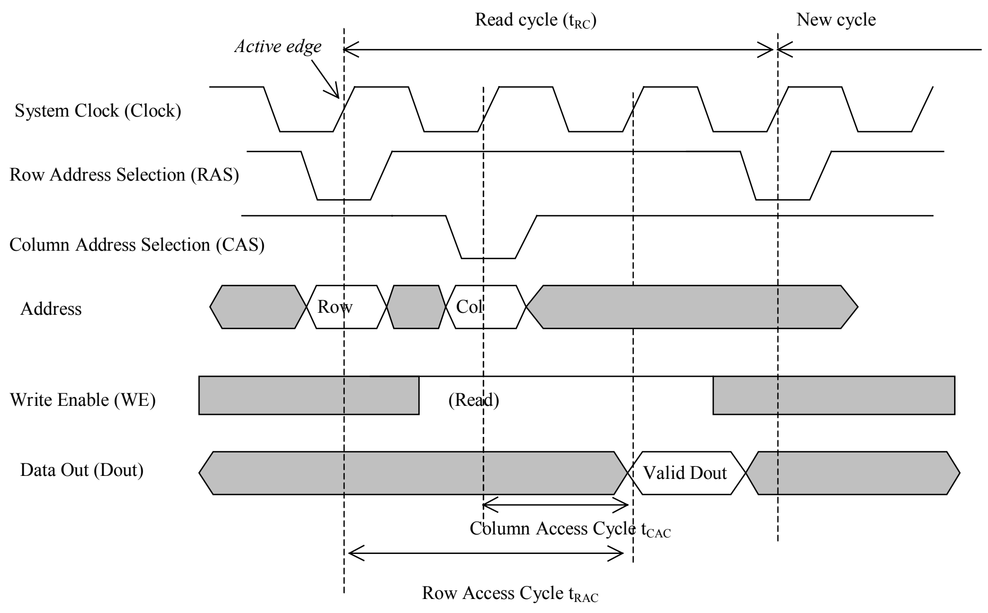


Figure 10-99: Synchronous RAM timing diagram

Double Data Rate

Double data Rate memories involve both the rise and fall edge of the clock [Sharma]. Furthermore, a series of data from adjacent memories may be sent in series on the data bus. Two contiguous data are sent, one on the rise edge of the clock, the other on the fall edge of the clock. This technique is called "burst-of-two". An example of double data rate and burst-of-two data in/out is proposed in figure 10-100. Notice that *Data In* and *Data Out* work almost independently.

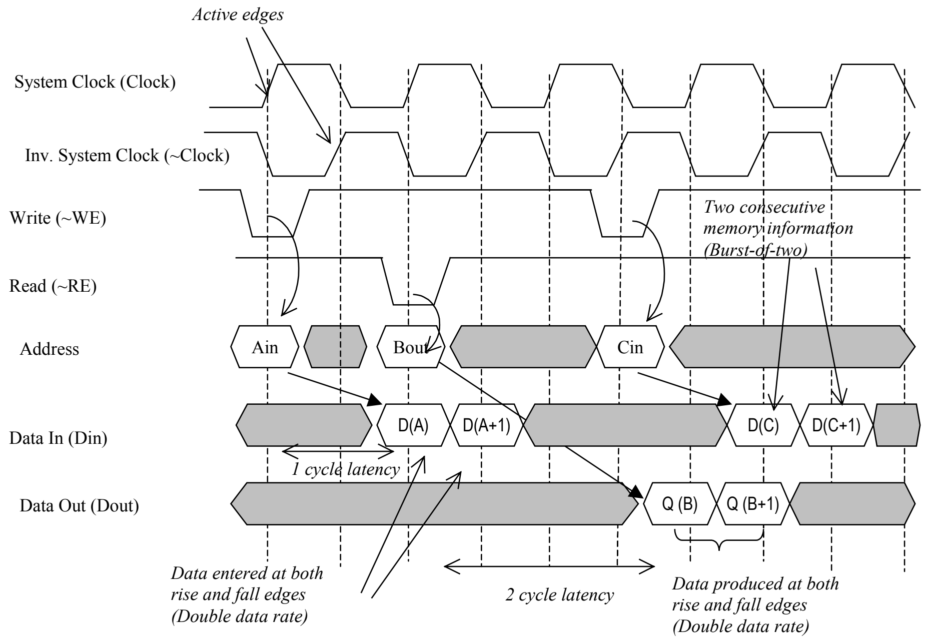

Active edges
System Clock (Clock)
Inv. System Clock (~Clock)
Write (~WE)
Read (~RE)
Address
Data In (Din)
Data Out (Dout)
Ain
Bout
Cin
D(A)
D(A+1)
D(C)
D(C+1)
Q (B)
Q (B+1)
Two consecutive memory information (Burst-of-two)
1 cycle latency
2 cycle latency
Data entered at both rise and fall edges (Double data rate)
Data produced at both rise and fall edges (Double data rate)

Figure 10-100: Illustration of the double data rate, and the use of burst-of-two memory

10.11 Conclusion

This chapter has focused on memories, which are a very important part of modern integrated circuits. The static memory has been described first, with an illustration of design challenges and dangers in the case of five-transistor architecture. The 6-transistor cell has been presented together with some layout optimization techniques to achieve the most compact memory design. The column and row selection circuits have been rapidly described, as well as the sense amplifiers used to speed up the read cycle. The principles and technological challenges related to the dynamic RAM have been introduced, with focus on the embedded stacked capacitor. The ROM memory has also been introduced. Extensive details on the EEPROM memory and its FLASH derivative have been provided in this chapter. Finally, we have introduced the principles and layout implementation of the ferro-electric RAM memory (FRAM), and concluded by a description of the asynchronous and synchronous memory interfaces.

REFERENCES

[Sharma] A. K. Sharma "Semiconductor Memories", IEEE Press, 1997, ISBN 0-7803-1000-4
[Kalter] Kalter Y. "Embedded DRAM, technological opportunities and challenges", IEEE spectrum, April 99, pp56-64
[Haraszti] Tegze P. Haraszti "CMOS memory Circuits", Kluwer Academic Publishers, 2001, ISBN 0-7923-7950-0
[Geppert] Linda Geppert "The New Indelible Memories", IEEE spectrum, March 2003, page 49
[Song] Yoon-Jong Song "Ferroelectric Thin Films for High desnsity Non vonlatile Memories", PhD dissertation, 1998, Digital Library and Archives, http://scholar.lib.vt.edu/theses

EXERCISES

Exercise 10-1
Compare the leakage current on a DRAM cell for the following technologies : 0.35µm, 0.12µm and 90nm.

Exercise 10-2
Given a 4x4 EEPROM memory array, create the chronograms to write the words 0001, 0010, 0100 and 1000, and then to read these values.

Exercise 10-3
The dual-port RAM is used as a memory shared by two blocs. The first bloc is the microprocessor unit which addresses the contents of the memory for write and read operations. The second bloc is usually a video memory or a slave processor which can only access the memory for reading. The basic dual-port cell is proposed in figure 10-101. Design and implement a 4x4 dual port RAM and perform the asynchronous reading of the DRAM contents by the slave memory circuit, while writing on the master memory circuit.

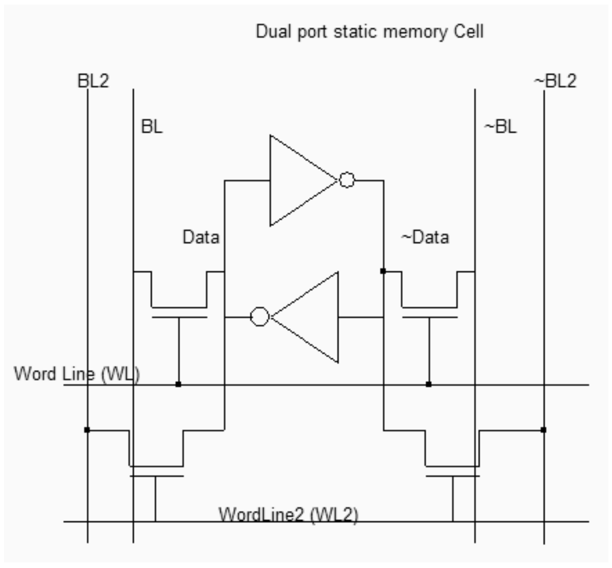


Figure 10-100: Illustration of the dual port SRAM cell

Exercise 10-4

Modify the ROM array to write the word "Welcome".

Exercise 10-5

Build a 4x4 MRAM array with complete read/write circuits.