Quality Memory Blocks - Balancing the Trade-Offs

Betty Prince, Ph.D.
Memory Strategies International

Abstract

Memory blocks have the basic quality requirements shared by all IP blocks. These include transferability between manufacturing areas, transferability from the original technology to the next generation technology, compatibility with available design tools, and qualified manufacturability in available wafer fabs. In addition to these general quality requirements, issues specific to memory blocks need to be considered. These include: memory type and cell for the specific implementation; memory technology generation to be used; cost issues such as requirements for special process modules; design issues such as choice of array compiler or use of predefined memory blocks; yield improvement issues such as redundancy type and implementation; test issues including BIST or direct memory access, special memory test requirements such as bit mapping, and availability of memory testers; reliability issues such as disturb problems, burn-in requirements and soft error considerations; architectural issues such as on-chip bandwidth access, pitch matching of array logic, and refresh implementation. This paper discusses these memory specific quality issues and the trade-offs involved.

I. Overview:

A quality memory core is defined here as one that is selected to be adequate for the requirements of the application including cost, reliability, and performance characteristics. Due to the level of complexity of the selection process, choosing the best memory core for the application requires careful analysis of the various options.

II. Memory Type and Cell for the Specific Implementation:

System characteristics determine the choice of memory type. System requirements include: density, volatility, performance, power consumption, and noise immunity. These characteristics are determined by both the architecture of the memory periphery and the cell type of the memory.

Memory density required varies from system to system and tends to increase for a given system type over time. Projected main memory densities of various systems over time are shown in Figure 1. Also shown is the density of the production standalone DRAM in the given time period.

Figure 1
Memory Densities of Various Systems Over Time

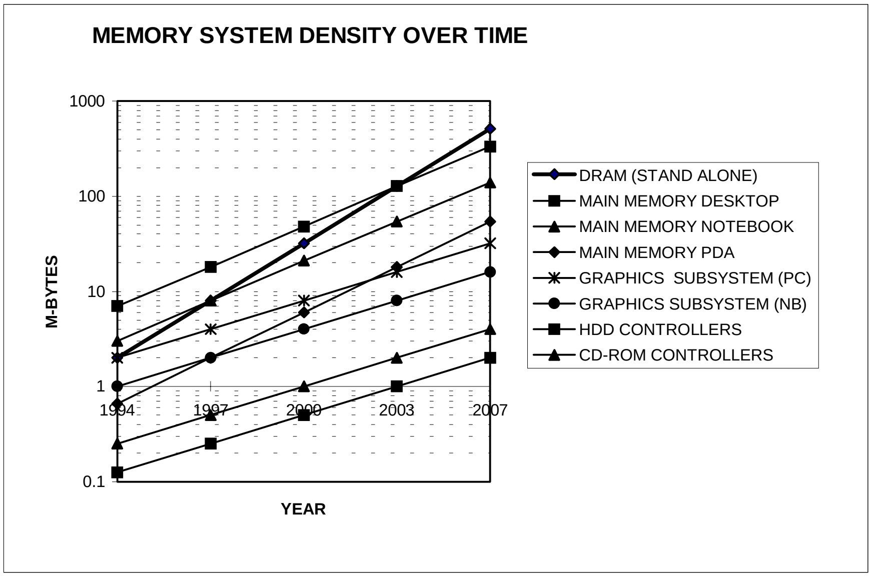


Figure 2
CELL TYPE AND DENSITY OF VARIOUS MEMORIES

	DRAM	SRAM	ROM	EEPROM	FLASH
Transistors	1.5	6	1	2	1-1.5
Density (bits)	128M	4M	256M	256K	64M
RAM Cycle Read (ns)	80	25	25	80	80
RAM Cycle Write(ns)	80	25	25	30000	30000
Overhead Circuitry	Yes	No	No	No	Yes
Relative Density	2	6	1	2	1.5

Memory cell types in general include: DRAM, SRAM, ROM, and E(E)PROM and Flash EPROM. They are differentiated by both properties and cost. The cell type of each is shown in Figure 2

Factors effecting density include both the relative cell size and the amount of support logic in the periphery required for operation of the memory type. DRAMs, for example, require refresh logic to periodically restore the charge to the storage capacitor as it leaks away. They also require circuitry to boost the wordline and precharge the bit lines. Flash memories with single external power supplies require circuitry for the high voltage program and erase operations and the flash memories with high density stacked cells require additional circuitry to control a complex algorithm for erase.

There can also be a choice of cell within a given type of memory macro. Both DRAMs and E(E)PROM/Flash memories can be built in optimized memory technologies which add additional processing steps above those used for the standard CMOS logic process. The extra steps in these optimized technologies can increase the cost of the chip by reducing the wafer sort yield and process yield. The trade-off is that the cell size is significantly reduced which, for a large memory macro, permits a significantly denser memory and smaller overall chip size.

The alternative is to use a memory cell made in a pure logic technology or which has minimal additional process steps. Cells with these properties are available both for DRAMS and for Flash memories and EEPROMs. An example for DRAMs is the four and three transistor cell as well as a simple planar one transistor cell.

For EEPROMs, there are also examples of optimized and logic compatible cells. An optimized floating gate stacked Flash memory cell might be made in double or even triple polysilicon technology while several examples have been shown of logic compatible Flash cells made in single polysilicon technology. The difference is in the cell size in a given technology. For example, an optimized stacked cell in 0.5um technology can have a .25 um2 cell size, while a logic compatible single poly cell in the same 0.5um technology would have a cell size of about 20 um2. The issue is the complexity of the process vs. the size of the memory macro.

The Flash memories have a wide selection of cell types: stacked, split, thin oxide, thick oxide, NAND, NOR, and DiNOR among others . For higher density circuits a stacked cell may be used. The stacked cell is a single minimal sized transistor that can be scaled to produce a dense array, but it requires additional control circuitry in the periphery. The NAND cell is even denser than a NOR cell, but it is slower. A DiNOR is faster than a NAND but not as dense. A split gate cell is larger than a stacked cell, but it doesn't require the additional peripheral control circuitry.

Different memory types and configurations can provide different performance levels. For example, given an SRAM of a specific density and a DRAM with a similar density and array configuration, the SRAM random read and write cycle time will be shorter since the DRAM must restore the charge to the cell, then precharge the bit lines before opening another wordline.

If, however, the DRAM is divided up into multiple arrays, then the speed may be increased due to the shorter word lines and bit lines. The macro size, however, will increase. If the speed is adequate and the divided array DRAM macro is not as large as for an SRAM, then the DRAM macro may be used as a fast memory in place of an SRAM.

Another potential method for increasing the access time of a macro is to use some of the techniques used to make standalone memories faster. One method is to make the memory synchronous. Additional speed can be obtained by pipelining the address and data path and/or using a wide bus from the array to prefetch multiple words on one clock cycle into a fast register with individual words sent out at the same multiple of speed. This technique can be used with any of the underlying memory types including DRAM or E(E)PROM/Flash.

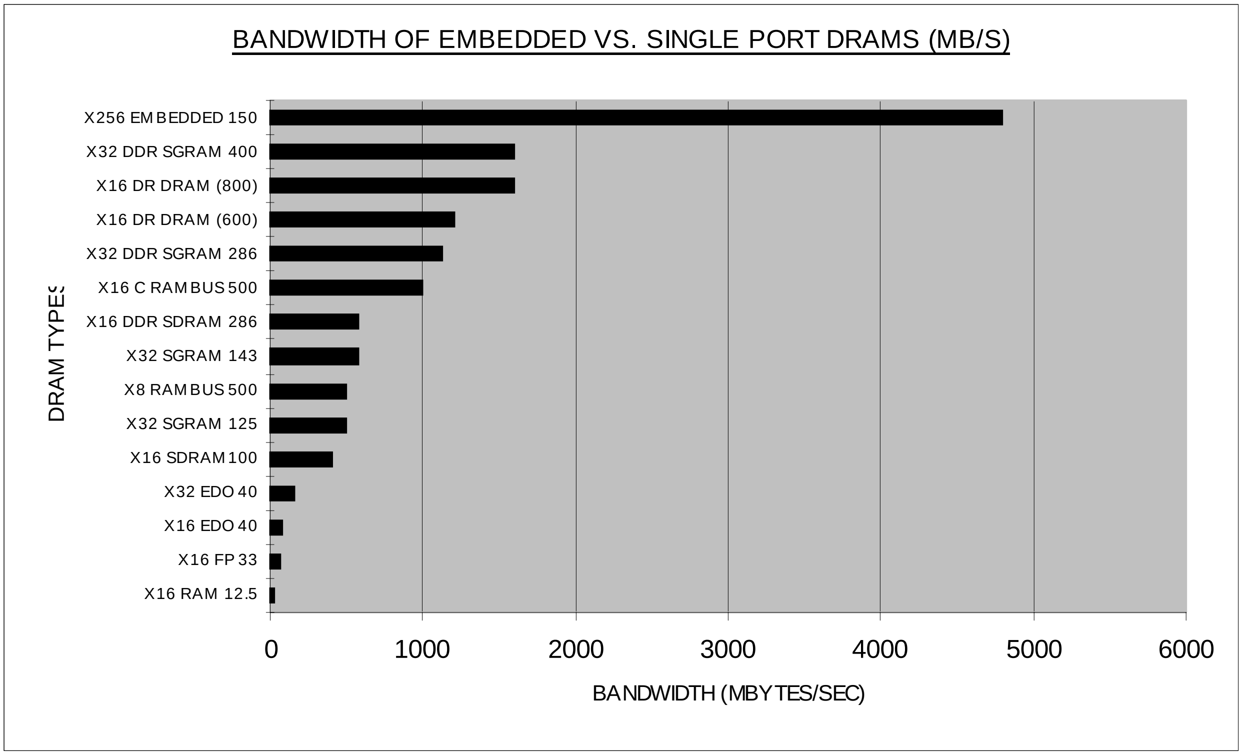


Figure 3. Bandwidth for an Embedded DRAM vs. Various Standalone DRAMs

An expansion of this method is to use a larger SRAM as a cache between a denser DRAM or EEPROM and the on-chip processor.

Both the techniques of dividing the array and of using a prefetch are used in various memory macro's today. Special configurations of memory are also possible in embedded applications that can tailor the memory to the requirements of the processor and thereby enhance the performance. Embedded CAMs and dual port SRAM macros are good examples. Embedded CAMs have been shown in both SRAM and EEPROM technologies. SRAM is faster, but the EEPROM provides a non-volatile storage medium.

Another performance issue is power consumption. Power can often be traded off against bandwidth requirements, bandwidth being the width of the bus times the speed of the bus. Potential bandwidth of embedded memories can be quite high. An example is shown in Figure 3 where the 4.8GB/s bandwidth for a DRAM macro using a 256 bit wide bus running at 150 MHz is considerably higher than the 1.6GB/sec potentially available in the most advanced standalone DRAMs today.

If, however, only 1.2 GB/s bandwidth is required for the application, then the speed of the embedded DRAM can be reduced to 37 MHz. This in turn reduces the power dissipation on the chip and also in the output buffers in addition to reducing the ground bounce and transmission line effects of going off chip resulting in a lower cost package, and lower cost printed circuit board.

It is also possible to have almost the same effect using the higher 150 MHz speed on the chip and then going to a quarter of the speed going off chip. Power consumption can also be reduced in a DRAM by dividing the word lines and only opening the segment of the wordline that is required, the trade-off is an increase in the size of the array.

III. Memory Technology Generation:

There are also trade-offs involved in deciding which technology generation to use. An established technology such as 0.25 embedded memory and logic will have a higher yield today than an 0.18um technology that is still in early production. Say, for example, that in some fab at some point in time, the yield in the 0.25um process is four times that in the 0.18um process. However, a chip will have about 0.52 the area in the former technology that it has in the latter resulting in about twice as many potential good chips per wafer. The combined result in this case is that the yield of a chip redesigned from 0.25um to 0.18um technology would fall to half the chips per wafer. Later, as the more aggressive technology reaches stability, the yields would go up and the benefit of the change would be realized.

EFFECT OF CHIP SIZE INCREASE ON YIELD OF STABLE (FICTITIOUS) WAFER FAB

		STABLE MEMORY PROCESS				DIE SIZE INCREASE				PROCESS COMPLEXITY			(+10%)	
	3Q99	4Q99	1Q00	2Q00	3Q00	4Q00	1Q01	2Q01	3Q01	4Q01	1Q02	2Q02	3Q02	4Q02
AVE WAFER/MO(K)	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00	45.00
WAFERS IN/QTR	135.00	135.00	135.00	135.00	135.00	135.00	135.00	135.00	135.00	135.00	135.00	135.00	135.00	135.00
LINE YIELD[2]	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.87	0.87	0.87	0.87	0.87
WAFERS OUT			125.55	125.55	125.55	125.55	125.55	125.55	125.55	125.55	125.55	117.45	117.45	117.45
PGDPW			295.00	295.00	295.00	200.00	200.00	200.00	200.00	200.00	200.00	200.00	200.00	200.00
PGD OUT			37037.25	37037.25	37037.25	25110.00	25110.00	25110.00	25110.00	25110.00	25110.00	23490.00	23490.00	23490.00
PROBE YIELD			0.67	0.67	0.67	0.67	0.61	0.61	0.61	0.61	0.55	0.55	0.55	0.55
ASSEMBLY YIELD			0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92
F. TEST YIELD			0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94
TOTAL YIELD			0.58	0.58	0.58	0.58	0.53	0.53	0.53	0.53	0.48	0.48	0.48	0.48
POT UNITS OUT(KU)			21459.98	21459.98	21459.98	14549.14	13246.23	13246.23	13246.23	13246.23	11943.32	11172.78	11172.78	11172.78
TECHNOLOGY(UM)			0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
UNITS OUT/WAFER IN			158.96	158.96	158.96	107.77	98.12	98.12	98.12	98.12	88.47	82.76	82.76	82.76
				STABLE										
				OUTPUT										

Figure 4. Memory Process Showing Effect of Increasing Chip Size and Process complexity

IV. Cost Issues:

System chips with optimized memory macros tend to be more expensive than a chip in pure CMOS logic because of the added number of process steps.. Adding process steps increases the cost of a chip for several reasons. The process yield can decrease due to the additional handling and the wafer sort yield can decrease with additional process steps since the defect density will go up.

A hypothetical example of a model of a stable memory wafer process is shown in Figure 4. The effect of adding process steps on the number of units out is shown first and then the effect of increasing the chip size.

The increase in chip size and process complexity can significantly decrease the potential chips per wafer which increases the cost per chip since the basis of manufacturing cost is the wafer.

V. Design Issues:

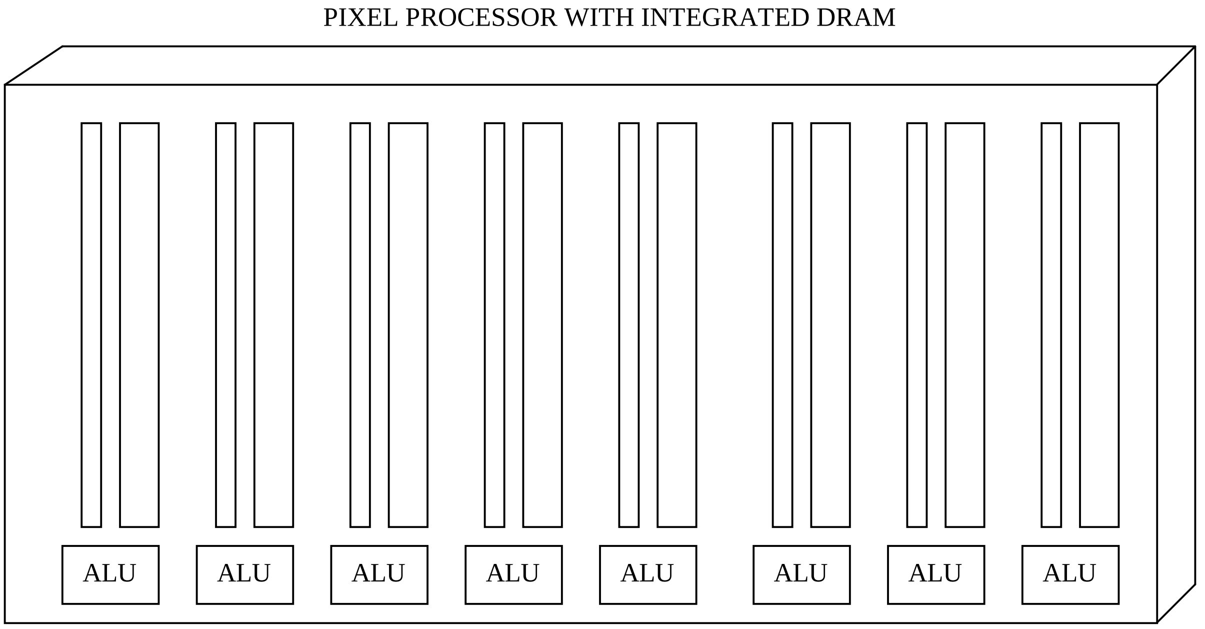
Another issue to consider is whether to use a predefined memory macro, a block compiler constructing an array out of smaller blocks, or a cell compiler. Cell compilers are commonly used for SRAMs and various SRAM cell compilers are commercially available. DRAMs with cells configured with three or four transistors can also use cell compilers.

Optimized DRAMs tend to use block compilers since even small blocks tend to maintain some of the high-density optimization that is the only justification for using DRAMs. Block compilers can either be very simple or can attempt to construct space saving features such as shared sense amplifiers.

There are also design issues for the required control circuitry for the different memory types. The control circuitry of an embedded SRAM tends to be very simple consisting of row and column address decoders, drivers, and sense amplifiers. The control circuitry of both DRAMs and EEPROMs is more complex.

DRAM cells require refresh since the data is stored in the form of charge on a capacitor. A single refresh cycle restores full charge to the storage capacitors in one row of the DRAM. A subsequent refresh restores charge to the next row. This means that row address counters and a clock are required to do refresh transparently on an embedded DRAM. DRAMs also normally use word lines boosted above VDD and circuitry to precharge and equalize the bit lines when a row is closed.

EEPROMs/Flash memories require a boosted voltage level for the program and erase. Flash memories with a stacked cell require a state machine to control the program erase algorithm due to the potential for over-erasing the floating gate.



Source: MOSAID

Figure 5. Pixel Processor Column Architecture

VI. Architectural Issues:

An architectural issue specific to memories is pitch matching any logic used in the array with the columns so that the space used is minimized. This issue arises in DRAMs with fitting the sense amplifiers into the pitch of the bit lines. The trade-off is between the density of the array and the complexity required for the sense amplifier.

An example of an architectural solution for this problem is the use of folded bit line with shared sense amplifiers so that each sense amplifier fits in the space of four bit lines. Folded bit lines permit more sensitive sense amplifiers can then be used, or various computing functions can be added to the column function. An example of a computing function added to the column on a DRAM is shown in Figure 5 which illustrates a pixel processor column architecture where four columns are associated with a single pixel processor

VII. Yield improvement Issues and Redundancy:

The redundancy that is always used on standalone DRAMs and used most frequently for embedded DRAMs is laser fuse redundancy in which fuses are blown by a laser operation directly after wafer sort. This type of fuse is smaller than an on-chip fuse even with the required guardrings to protect the rest of the chip during the laser procedure.

It is also possible to implement redundancy using fuses on the chip. The drawback of this technique is that the fuse can increase the chip size. Advantages include: field correctability, and ability to make corrections at final test. EEPROMS and Flash memory macros, for example, can implement redundancy using spare memory cells for the fuses.

Normally a few rows and columns of redundancy are offered on a large DRAM macro. For smaller macros less redundancy may be necessary. The reason for requiring redundancy is that the high density of a memory array means that any wafer processing defect in the array area will fall on an active area. The lower density of the active area in the logic sections of the chip mean that there is a higher probability of any defect being on an active area.

Since with the use of redundancy, the yield may be higher in the RAM part of the chip than the logic part of the chip, it would be useful if redundancy could be implemented to some extent in the logic part of the circuits. Several techniques have been explored for replacing logic circuits. To implement a spare decoder, for example, switches can be used that shift all decoder connections by one to bypass the faulty decoder. Another option, when logic, such as a comparitor, is added to the column structure of a memory is to replace the logic with its associated column.

VIII. Test Issues for Embedded Memories:

There are several approaches for testing embedded memory. The most straightforward is a direct memory access. This method is used frequently with large embedded DRAM arrays and consists of multiplexing the pins of the integrated chip so that in test mode the memory can be directly accessed. The drawback of this methodology is that the logic integrated on the chip must be tested separately necessitating, in many cases, two insertions into a test machine. It also can mean purchasing both a memory and a logic tester, although there are now logic testers available that can also test embedded DRAMs.

A method often used to test embedded SRAM is built in self test (BIST). The BIST circuitry can consist of a pattern generator and a data comparitor for checking the pattern from the RAM against the expected pattern. The result is a go-no go test. The BIST generator can be activated by a scan chain. The trade-off here is the added silicon used for the BIST generator and the number of test patterns to adequately test the SRAM block. The patterns used for testing can either be stored in a microcoded ROM on the chip or downloaded from the system or tester. The trade-off is the size of the ROM vs. the convenience of being able to test simply off-line from the tester.

BIST has also been shown for embedded DRAM although few circuits with embedded DRAM and BIST have reached the market. There are several considerations here that have motivated many companies making embedded DRAM to use direct memory access. The first consideration is the test coverage which is more difficult to implement for DRAMs using only a pattern generator. Next is the difficulty in producing a bit-map with BIST so that redundancy can be implemented on the DRAM. Most embedded DRAMs require redundancy to improve the chip yield. In one example, the failing addresses were collected in a register on chip that could then be polled for implementing the redundant rows and columns.

Another DRAM implemented only two long tests in the BIST which was turned on during burn-in otherwise it used direct access test.. The purpose of the BIST was to reduce the amount of time on the tester and hence reduce test cost.

It is also possible to implement build in self-repair (BISR) on a memory using on-chip fuses or off chip laser blown fuses.

IX. Reliability Issues:

DRAMs and Flash memories normally are burned-in prior to usage in a system to remove early hard fails. This entails running them at higher temperatures and/or higher voltages for a designated number of hours then testing to remove the failures. Burn-in can be done with a system chip as well as with a stand-alone memory. Stress testing is sometimes used as a substitute on standalone memories but would require care on embedded memories to avoid undue stress on logic circuitry.

DRAMs tend to suffer disturb problems from the noise of high-speed on-chip logic which can lead to soft errors. Both DRAMs and SRAMs can suffer soft errors from both alpha radiation due to low levels of natural radioactive contamination in the packaging and process materials or from natural background neutron radiation which is commonly referred to as the Cosmic Radiation problem. The signature of this background neutron radiation effect is that it is altitude and latitude dependent and also depends on the 11-year solar cycle.

X. Market issues:

Finally there are market issues which can effect some of the above technology choices. Market window may be a deciding factor in process or manufacturing selection since an available adequate process may need to be selected over a more optimum but less available process in the interest of hitting the market window.

Projected manufacturing volume may be a factor in design and technology selection. Revenue and margin potential also play a role since a larger burget may permit a more expensive technology or manufacturing option which optimizes some other parameter to be chosen.

Expected life cycle can also affect the choice of design and process technology. For example a mask ROM can be used more cost effectively than a EEPROM core in a microcontroller if the life cycle is long enough, the volume high enough to cover the initial mask charges, and no changes in the stored data are expected. In the debug stage, however, or where code changes are expected, the reprogrammable core can be more cost effective and therefore a better choice. If the expected redesign or upgrade window is short, then a reprogrammable technology may be chosen.

XI. Summary

In summary, there are many considerations both technical and market related in choosing a quality memory technology for an embedded application. Most of the trade-offs are well understood, but it is easy to overlook an important consideration without careful study of the trade-offs before beginning to design.