Towards Generic Low-Power Area-Efficient Standard Cell Based Memory Architectures

P. Meinerzhagen, C. Roth, and A. Burg
Integrated Systems Laboratory, ETH Zurich, Switzerland
{meinerzhagen,rothc,apburg}@iis.ee.ethz.ch

***Abstract*—Digital IC designers often use SRAM macrocells to implement on-chip memory functionality. In this paper we argue that in several situations, standard cell based memories (SCMs) can have advantages over SRAM macrocells. Various ways to implement SCMs are presented and compared to each other for different CMOS technologies and standard cell libraries and to corresponding macrocells, aiming for finding the most adequate memory option for each application. The benefits and drawbacks of SCMs compared to macrocells are illustrated with the example of a low-power low-density parity check (LDPC) decoder.**

I. Introduction

Virtually any CMOS chip manufactured nowadays includes some form of memory. The major options for on-chip memory implementations are: (1) registers built from flip-flops or latches, and (2) SRAM macrocells [1]. SRAM macrocells are compatible with standard CMOS technologies and for most technologies SRAM macrocell generators are available. In the following, we shall refer to flip-flop/latch arrays and to SRAM macrocells as *standard cell based memories* (SCMs) and *macro memories* (MMs), respectively, and we shall argue that for reasonably small storage capacities, SCMs might be an interesting alternative to MMs in order to improve area and energy efficiency, amongst others.

The use of SCMs described in a hardware description language eases the portability of a design to other technologies. Macrocells need to be created again for each new technology, using a dedicated memory compiler which might generate cells not fully compatible with the original design. Also, SCMs can be described in a generic way, which renders it easy to modify the *number of words* or the *number of bits per word* at design time; also, any desired numbers can be chosen. Furthermore, designs comprising SCMs can be placed automatically using the standard placement tool, whereas MMs need to be placed manually. Consequently, SCMs can be merged with logic blocks, which improves data locality and thus can reduce routing.

The one-bit storage cell of SCMs (flip-flop or latch) is clearly bigger than the one of MMs (SRAM cell). However, MMs require more peripheral circuitry such as precharge circuitry and sense amplifiers [2] than SCMs. For MMs with small storage capacity, the area overhead due to peripheral circuitry can be significant. Hence, SCMs can outperform MMs in terms of area for small storage capacities, but become much bigger for large storage capacities.

In general, it is not clear if SCMs have better energy efficiency than MMs. However, the use of SCMs can reduce routing, which leads to a reduction in power consumption. Also, traditional 6T SRAM would not work near the threshold voltage [3], [4], while SCMs can also be operated in the subthreshold regime without the need for fullcustom design.

SCMs can share the power and ground rings with the rest of the chip, while MMs typically have extra rings. For reconfigurable designs targeting low power consumption, memories are preferably organized in many small blocks which can be turned on and off separately. In the context of such fine-granular memory organizations, SCMs provide more flexibility at design time, might result in smaller overall area due to the lack of separate rings and less peripheral circuitry, and are more adequate to reduce the overall power consumption.

Contribution: In this paper, various possibilities to implement SCMs are presented and compared to each other and to corresponding MMs in terms of area and power. Also, the benefits and drawbacks of SCMs are shown through the application example of a low-power low-density parity check (LDPC) decoder.

Outline: Section II describes the various SCM architectures and gives synthesis and post-layout power analysis results. Section III shows how the concept of SCM is applied to the low-power LDPC decoder. Section IV concludes the paper.

II. Comparison of Standard Cell Based Memory Architectures

The remainder of this paper assumes memories with a separate read and write port, a word access scheme, and read and write latency one. Any such SCM has the following building blocks: (1) write logic, (2) read logic, and (3) array of storage cells. Different ways to implement the write and read logic are presented in Sections II-A and II-B, respectively, assuming flip-flops as storage cells. The use of latches instead of flip-flops as storage cells is discussed in Section II-C.

A. Write logic

Consider an array of $R \times C$ flip-flops, where R and C denote the number of rows (words) and the number of columns (bits per word), respectively. Assuming a word access scheme and a write latency one, the write logic needs to select one out of R words, according to the given write address, and update the content of the corresponding flip-flops on the next active clock edge. To accomplish this, the *write address decoder* (WAD) produces one-hot encoded row select signals, which select one row of the flip-flop array. Next, the flip-flops in the selected row need to update their state according to the data to be written. One possibility consists in using flip-flops with an enable feature or with a corresponding logic (*FFE* architecture); all flip-flops in one row are enabled by the same row select signal. Another possibility consists in using basic flip-flops in conjunction with clock gates (*CG* architecture): a separate clock signal is generated for each row, and only the currently selected row receives a clock pulse, thereby sampling the provided data, while all other rows receive a silenced clock, thereby keeping their previous data.

Synthesis results using different CMOS technologies and different standard cell libraries show that the CG architecture yields smaller SCMs than the FFE architecture for $C \geq 4$ in most cases, and $C \geq 2$ in few cases. This result is almost always independent of R.

It is clear that the CG architecture consumes less power than the FFE architecture, as the latter distributes the clock signal to each storage cell, while the former silences the clock signal of all but the selected rows. Furthermore, the 2-to-1 multiplexer inside the enable flip-flop consumes additional power which can be avoided.

B. Read logic

The read logic can be purely combinational or contain sequential elements, which leads to a read latency. Assuming a word access scheme, one out of R words must be routed to the data output, according to the read address. The typical one-cycle latency is obtained by inserting flip-flops either at the read address input or at the data output. The former and latter case require $\mathrm{ceil}(\log_2(R))$ and C additional flip-flops, impose gentle and hard read address setup time requirements, and cause considerable and negligible output delays, respectively. The task of routing one out of R words to the output can be done using either tri-state buffers or multiplexers.

1) Tri-state buffer based read logic: This approach asks for a *read address decoder* (RAD) to produce one-hot encoded row select signals, and $R \cdot C$ tri-state buffers, i.e. exactly one per storage cell. R tri-state buffer outputs connect to one bit line, which has a large lumped capacitance if R is big. Tri-state buffers with high driving capability are required to drive this capacitance. If R increases, stronger buffers are required, which further increases the lumped capacitance and thus requires even stronger buffers. Hence, employing tri-state buffers is expected to result in a large overall area and a high power consumption. Furthermore, it is difficult to buffer tri-state buses [5], which might be necessary to maintain reasonable slew rates if these buses are routed over long distances. Also, if two or more row select signals accidentally overlap, DC paths from VDD to GND can arise and short-circuit power is consumed.

2) Multiplexer based read logic: C parallel R-to-1 multiplexers are required to route an entire word to the output. The R-to-1 multiplexer itself can be implemented in many ways. Most multiplexer architectures do not require one-hot encoded row select signals and can therefore save the RAD. However, there is an energy efficient multiplexer architecture which accepts one-hot encoded row select signals, performs a logic AND operation between each row select signal and the corresponding data bit, and finally performs a logic OR operation on all AND-gate outputs. For this particular multiplexer architecture, and assuming a proper, i.e. a non-overlapping one-hot code at the selection inputs, any glitch or activity on an unselected data input will die out after the first logic stage. As opposed to this, glitches or activity on unselected data inputs of a binary selection tree multiplexer can propagate until the input of the last stage. Most logic synthesizers would yield multiplexers similar to the AND-then-OR multiplexer, typically employing dedicated multiplexer cells in the back-most logic stages.

3) Simulation results: Flip-flop based SCMs using clock gates for the write logic, and using either multiplexers or tri-state buffers for the read logic are synthesized, placed, and routed for different memory dimensions $R \times C$ (cf. Table I) as well as for different technologies and different standard cell libraries (cf. Table II). For the VCD-based post-layout power analyses, random data is written to random addresses, while data is read from random addresses, for 1’000 cycles at a clock frequency of 100 MHz. All inputs of the SCMs can be driven by buffers of standard driving strength; highly capacitive nets such as the bit lines are buffered inside the SCMs.

The multiplexer based SCMs always have smaller area and lower power consumption than the tri-state buffer based SCMs. However, the power estimation of the tri-state buffer based SCMs is rather optimistic as short-circuit power due to DC paths through tri-state buffers is not accounted for in the simulations.

Table I
FLIP-FLOP BASED SCM, CG WRITE LOGIC, 0.13-μm CMOS: AREA AND POWER FOR MULTIPLEXER AND 3-STATE READ LOGIC FOR DIFFERENT CONFIGURATIONS $R \times C$.

		Area [μm^2]		Power [mW]	
R	C	MUX	3-state	MUX	3-state
16	8	6k	6k	0.8	0.8
16	128	67k	76k	5.3	6.9
32	8	10k	11k	1.0	1.3
32	128	135k	170k	8.1	14.1
64	8	20k	28k	2.4	4.2
64	128	274k	397k	19.5	38.4
128	8	39k	56k	4.5	9.1
128	128	557k	850k	38.0	93.8

Table II
FLIP-FLOP BASED SCM, CG WRITE LOGIC, R = 16, C = 128: AREA AND POWER FOR MULTIPLEXER AND 3-STATE READ LOGIC FOR DIFFERENT TECHNOLOGIES AND STANDARD CELL LIBRARIES.

	Area [μm^2]		Power [mW]	
Tech. & lib.	MUX	3-state	MUX	3-state
180nm i)	132k	170k	11.0	19.8
180nm ii)	126k	160k	12.5	17.0
130nm i)	67k	76k	5.3	6.9
130nm ii)	72k	83k	4.1	4.9
90nm	36k	41k	1.9	3.5

C. Array of storage cells

Instead of flip-flops, latches can be used as storage cells. The previous discussions on the write and read logic remain valid when latches are used as storage cells. However, setup and hold time requirements on both write and read ports can change considerably. Sticking to a single-edge-triggered one-phase clocking discipline and a duty cycle of 50%, the WAD together with the clock gates get the first half of a clock period to generate one clock pulse and $R-1$ silenced clocks, which will make—during the second half of the clock period—the latches in one out of R rows transparent and keep the latches in all other rows non-transparent, respectively. Those latches which have received a clock pulse store the applied input data on the next active clock edge.

If the currently transparent latches are also selected by the output multiplexers, the SCM becomes transparent from its data input to its data output, and combinatorial loops through external logic can arise. To avoid this problem, a restriction on the choice of read and write addresses must be imposed. If such a restriction is not desired, latches which are non-transparent during the second half of the clock period must be added at either the SCM's data input or output, or alternatively, registers must be inserted into any path feeding the SCM's data output back to the data input.

Averaging across different technologies and standard cell libraries, we find that the area of a basic latch with given drive strength is 77% of the area of a corresponding flip-flop. Even though the total storage cell area can be reduced by 23% on average when replacing flip-flops with latches, the total SCM area shrinks less, as write and read logic remain the same. In fact, for a 0.13-µm technology, averaging over 49 samples corresponding to $R = 2^3, 2^4, \ldots, 2^9$, $C = 2^1, 2^2, \ldots, 2^7$, latch based SCMs are only 13% smaller than flip-flop based SCMs.

In latch based SCMs, the WAD together with the clock gates get only half a clock period to select one out of R words, while in flip-flop based SCMs, they get a full clock period. This is why flip-flop based SCMs qualify better for high-speed applications where address generation involves a long combinational path which cannot be pipelined.

Fig. 1 shows the schematic of the proposed standard cell based memory, which uses latches without enable feature as storage cells, clock gates for the write logic, and flip-flops at the read address input in conjunction with multiplexers for the read logic.

For the smallest flip-flop and latch based SCM architecture, as well as for the SRAM macrocell, 49 samples corresponding to $R = 2^3, 2^4, \ldots, 2^9$, $C = 2^1, 2^2, \ldots, 2^7$ have been synthesized in a 0.13-µm CMOS technology. Fig. 2 shows all points in the $C \times R$ plane—using a log-log scale—for which the SCMs are smaller than the corresponding MMs. The sampled data points are interpolated in the least squares sense, and the intersection lines $SCM = MM$ of the resulting surfaces are plotted. Those intersection lines show the border up to which the SCMs are smaller than MMs. Of course, changing from flip-flop based to latch based SCMs pushes the intersection line toward slightly bigger storage capacities $R \cdot C$. The gray lines show all memory configurations $C \times R$ with constant storage capacity $R \cdot C$. Flip-flop and latch based SCMs are smaller than MMs for storage capacities of up to around 512 and 1024 bits, respectively, considering rather high but still very applicable C/R ratios.

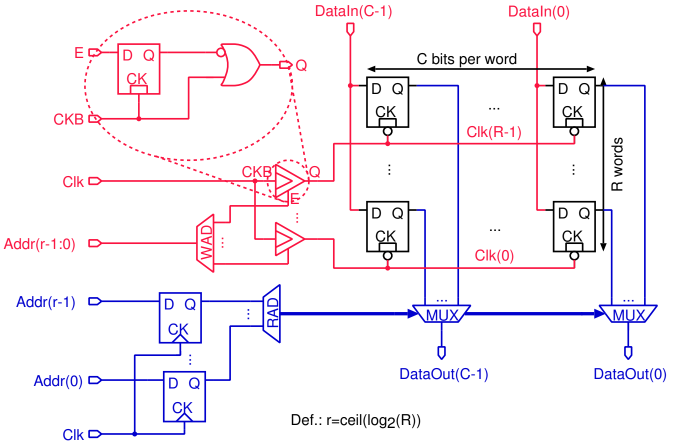

Figure 1. Schematic of latch based SCM with clock gates for the write logic and multiplexers for the read logic.

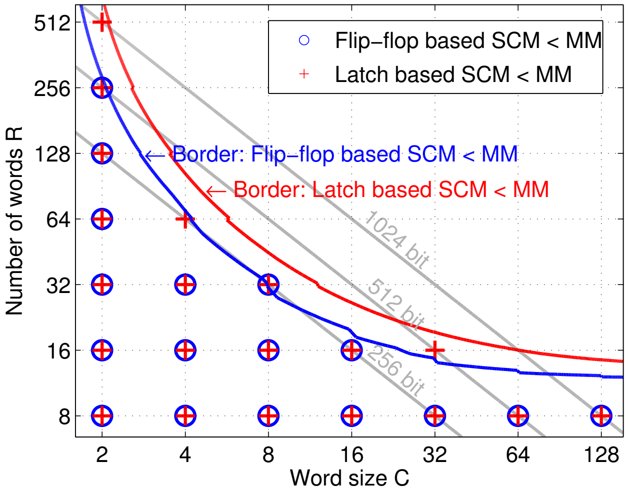

Figure 2. Flip-flop and latch based SCMs versus MMs: sampled data points and intersection lines of regression functions.

III. Application Example: Low-Power LDPC Decoder

LDPC decoders used in modern communication systems require a considerable amount of memories, which often consume a major part of the total power. Furthermore, most wireless communication standards define several operating modes, which asks for a fine-granular memory organization if low power consumption is targeted. The employment of SCMs is thus a promising way for designing portable low-power LDPC decoder intellectual properties.

In the following, two versions of an IEEE 802.11n-compliant low-power LDPC decoder based on [6] are compared. The first version uses SRAM macrocells and the second one uses several instances of the proposed latch based SCM (cf. Fig. 1). Both decoders contain three separate memories, named Q-, T-, and R-memory, and some combinational blocks between them. The R-memory is divided into an $(R,C) = (88,135)$ always-on block and two $(R,C) = (88,135)$ blocks which can be turned off separately, depending on the decoder's operating mode. Similarly, both Q- and T-memories are divided into three $(R,C) = (24,135)$ blocks.

Considering that $R < C$ for all employed SCMs, flip-flops are inserted at the read address input rather than at the data output. Each multiplexer selection signal has a fan-out of $C = 135$, which requires buffering and causes a non-negligible delay. In fact, it turns out that the paths through the output multiplexers are the most timing critical paths of the LDPC decoder design.

The two decoder versions are placed and routed in 0.13-µm CMOS technology for a target clock period of 6 ns, which is required to achieve the throughput demanded by the IEEE 802.11n standard. Table III shows the core area and the VCD-based post-layout power analysis results for both decoder implementations.

Table III
AREA AND POWER OF SCM VS. MM BASED DECODER

	Dec. w/ MM	Dec. w/ SCM	SCM gain/penalty
Power [mW]	144.32	91.58	−36.54%
Area [mm^2]	1.37	2.06	+49.97%

The power analyses show that the SCM based decoder consumes 37% less power than the MM based decoder. The main part of the decoder's power reduction can be attributed directly to the lower power consumption of the employed SCMs as compared to the MMs. Furthermore, power analyses and placement results show that SCMs enable a more local placement and routing, which leads to lower switching power. Fig. 3 for example shows that the T-memory in the SCM based decoder is completely merged into the main combinational block by the placement tool. This high data locality enables the routing tool to use shorter and lower-layer wires at these locations. Also, the fact that SCMs are not limited to a rectangular shape allows the placement tool to wrap the memories around the connected logic (cf. left part of Fig. 3), thereby minimizing wire lengths at the interfaces, which leads to a further reduction in switching power. For both decoder implementations, the leakage power is less than 1% of the total power.

All memory sub-blocks resulting from dividing the Q-, T-, and R-memory have a capacity $>$ 3kbit, which is too high for SCMs to outperform MMs also in terms of area (cf. Fig. 2). However, an increased silicon area is acceptable for the benefit of lower power consumption.

IV. Conclusions

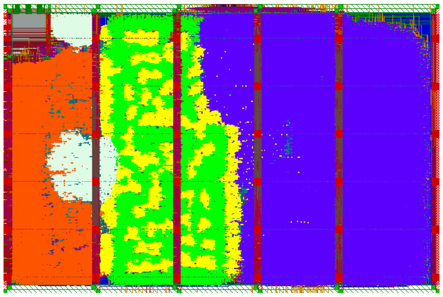

Figure 3. Layout of SCM based low-power LDPC decoder. The Q- and the R-memory are located on the left-hand and right-hand side, respectively, while the T-memory is located in the middle, merged with and surrounded by combinational logic blocks.

It was shown that SCMs can bring various benefits compared to SRAM macrocells, such as ease of portability, modifications at design time, ability to merge storage with logic, potentially less routing, lack of separate voltage supply rings, and more flexibility for fine-granular memory organizations. As for the write logic of SCMs, using basic flip-flops or latches as storage cells in conjunction with clock gates leads to smaller area and lower power consumption than using flip-flops or latches with enable feature. As for the read logic, multiplexer based implementations lead to smaller area and lower power consumption than tri-state buffer based implementations. Latch based SCMs are only slightly smaller than flip-flop based SCMs. Flip-flop based SCMs, however, are more convenient for high-speed applications than latch based SCMs. In our case study, a low-power LDPC decoder, which has 9 memory blocks with capacity $>$ 3kbit, becomes bigger when replacing MMs with SCMs, but its power consumption is significantly reduced. For applications requiring memories with storage capacity $<$ 1kbit, replacing MMs by SCMs can be profitable for both area and power.

Acknowledgment

The authors would like to thank Christoph Studer for providing the original LDPC decoder design. This work was kindly supported by the Swiss National Science Foundation under the project number PP002-119057.

References

[1] H. Kaeslin, *Digital Integrated Circuit Design: From VLSI Architectures to CMOS Fabrication*, 1st ed. Cambridge University Press, 2008.

[2] K.-S. Yeo and K. Roy, *Low-Voltage, Low-Power VLSI Subsystems*. McGraw-Hill, 2005.

[3] B. Zhai, S. Pant, L. Nazhandali, S. Hanson, J. Olson, A. Reeves, M. Minuth, R. Helfand, T. Austin, D. Sylvester, and D. Blaauw, "Energy-efficient subthreshold processor design," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 17, no. 8, pp. 1127–1137, Aug. 2009.

[4] A. Wang and A. Chandrakasan, "A 180-mV FFT processor using subthreshold circuit techniques," in *Solid-State Circuits Conference, 2004. Digest of Technical Papers. ISSCC. 2004 IEEE International*, Feb. 2004, pp. 292–529 Vol.1.

[5] J. Lillis and C.-K. Cheng, "Timing optimization for multisource nets: characterization and optimal repeater insertion," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 18, no. 3, pp. 322–331, Mar 1999.

[6] C. Studer, N. Preyss, C. Roth, and A. Burg, "Configurable high-throughput decoder architecture for quasi-cyclic LDPC codes," in *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, Oct. 2008, pp. 1137–1142.