Adapting the Columns of Storage Components for Lower Static Energy Dissipation

Mehmet Burak Aykenar, Muhammet Özgür, Osman Seçkin Şimşek, Oğuz Ergin
TOBB University of Economics and Technology
Ankara, Turkey
{mbaykenar, mozgur, ossimsek, oergin}@etu.edu.tr

***Abstract*—SRAM arrays are used especially in memory structures inside the processor. Static energy dissipation caused by leakage currents is increasing with every new technology and large SRAM arrays are the main source of the leakage current. We analyzed the content distribution of columns of SRAM arrays and based on the majority of the content, the body-bias of transistors are changed to reduce the static energy dissipation of these SRAM arrays. Our simulations reveal that when our technique is used in the register file of the processor, the leakage energy dissipation decreases by 39% and the total energy dissipation by 14% with an area overhead of 11%.**

Keywords-logic design; leakage current; static energy dissipation; low-power; variation

I. Introduction

On-chip memory structures such as caches at all levels, register files, branch prediction buffers, reorder buffers and other tables are constructed using static random access memory (SRAM) cells rather than dynamic RAM cells in order to achieve faster access. Today's high performance processors demand larger and deeper cache hierarchies to support faster data transfer operations and larger instruction windows to exploit instruction level parallelism (ILP). In order to satisfy this demand, resources became larger and with the help of transistor scaling, enlarging these structures became available with every new generation of technology.

Intermediate nodes of the digital circuits are charged and discharged each and every switching operation. This charging and discharging of the capacitances causes dynamic energy dissipation. However, with shrinking feature sizes of transistors and lower threshold voltages, transistors leak current, even when they do not perform any meaningful work. The power dissipation due to the leakage current is called static energy dissipation, sometimes also named as leakage energy dissipation to imply leakage current.

Storage components, which are getting larger, even with the shrinking size of the transistors, are the main source of static energy dissipation. L2 cache of AMD Opteron microarchitecture covers 42% of the all core area [16], also L2 & L3 caches of Intel Nehalem microarchitecture is nearly 20% of the all core area [17].

Several researchers observed the distribution of content values for rows of different memory structures inside processor to exploit any repetition caused by usually sign bits of the values. Observing rows of the memory arrays is a well-known and suitable approach, since for narrow values the sign bit of the number is extended to fill the memory space. Thus, significant effort have been made by observing the bit values in the rows of memory structures and numerous methods were proposed to exploit this fact to reduce static or dynamic energy dissipation of these modules. One exception to this approach tries to detect a column full of same logic values [2]. We leave the explanation of these previous methods of leakage current reduction to the related work section.

In this paper, we propose a new method to reduce the static energy dissipation of on-chip storage components by dynamically changing body-bias voltages of transistors. We show the design of a simple but effective majority detector circuit to observe the content of the columns of SRAM arrays and adapt the body bias voltage of the entire column according to the majority of the stored bit values. We also considered the effects of timing variation caused by different operating temperatures and showed that our design is capable of coping with these problems. Our simulation results show that our scheme reduces the static energy dissipation by 39% and the overall energy dissipation by 14% with 11% overhead in a register file that is built with 8T-SRAM cells.

Our scheme can be used for both observing the rows and columns of the storage elements as the same majority detection logic and control mechanism can be applied in both ways. However, we chose to observe content change in the columns instead of rows of the storage elements due to three main reasons:

- In contemporary 8T-SRAM bitcell designs VDD and other lines are located vertically instead of horizontally [18]. In the layout of the 8T-SRAM bitcell, NWELLs of the p-type transistors are connected and shared among the bits of a column, instead of rows of the storage arrays.
- The majority of the content of a row can change drastically when a write operation occurs. However, columns of the storage components are more stable than rows as one write operation affects only one bit of a column while for a row all the bits can change.
- The number of bits containing logic-0 will be very high for the upper most bits of lines in a storage element as they carry mostly positive numbers. Therefore, adapting body-bias voltage of the transistors used for the upper most bits in SRAM bitcells maintains high energy efficiency.

The remainder of the paper is organized as follows. In Section 2 8T-SRAM design is explained. Our majority detection scheme is proposed and the subparts of the scheme are detailed in Section 3. The simulation methodology is described in Section 4. Section 5 includes the results of our

978-1-4799-0524-9/13/$31.00 ©2013 IEEE

simulations and discussions. Section 6 summarizes the related work. Finally, Section 7 provides some clues to the future work and concludes the discussion.

II. SRAM Bitcell Design

SRAM bitcell was previously designed with six transistors and called as the 6T-SRAM bitcell. However, with 45 nm technology in 2008, Intel switched to a new bitcell design that consists of 8 transistors and called it as the 8T-SRAM bitcell [21]. This new bitcell is maintained in order to allow lower voltage operations along with more robust design [1]. The circuits of the 6T-SRAM & 8T-SRAM bitcells are shown in Figure 1. For writing into the bitcell, both 6T-SRAM and 8T-SRAM work in the same manner: After write driver activated, word line (WL) is set to logic-1 and selects the word to be written. Finally, the data is written into the bits of the selected word. Read operation is a bit different than the write operation and due to the disturbance of the bitcell when reading the data, 8T-SRAM is preferred. For reading the data in 6T-SRAM bitcell, bitlines are precharged to VDD/2 for low-power operation. When the word line selects the word to be read, the sense amplifiers sense the voltage difference between two bitlines and quickly give the output. However, precharged bitlines can disturb the content of the bitcell and may cause stability problems if this read operation is not carefully handled. 8T-SRAM bitcell addresses this problem and improves the read noise margin by differentiating read & write lines. Although our scheme is independent of a particular SRAM bitcell design, we used the contemporary 8T-SRAM bitcell model and the layout of the bitcell is based on Noguchi's work [18].

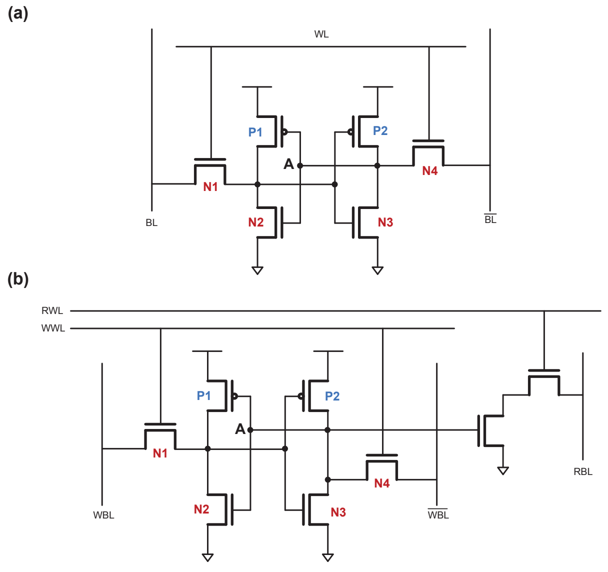

Fig. 1 : Schematic of the bitcell (a) 6T-SRAM and (b) 8T-SRAM

III. Reducing Leakage Current

Static energy dissipation caused by leakage current consumes significant portion of the total power dissipation with shrinking transistor sizes. As with 90 nm process technology the static energy dissipation became 50% of the total chip power dissipation [4]. Cache memory power dissipation is reached nearly 50% percent of the processor cores [8][9]. Register file is another major component where SRAM bitcells are used in a processor. Banking and clock gating strategies are used to reduce the power dissipation of the register file. Even with these methods, register file power dissipation can reach 15-20% of core power [10][11].

In SRAM bitcells, the leakage current follows different paths when the storage is logic-1 and logic-0. In Figure 1, when node A = 0, the standby current leaks over N2, P2 and N4, whereas when A = 1, the standby current leaks over N3, P1 and N1.

$$V_{TN} = V_{TO} + \left[\left(\frac{t_{ox}}{\epsilon_{ox}}\right)\sqrt{2q\epsilon_{si}N_A}\right]\left[\sqrt{|V_{SB} + 2\varphi_F|} - \sqrt{|2\varphi_F|}\right] \qquad (1)$$

The reason for increasing the leakage current with shrinking feature sizes is the decrease of the threshold voltage levels of the NMOS transistors and the increase of the threshold voltage level of the PMOS transistors. Threshold voltage can be adapted during manufacturing time or by changing the body-bias voltage of the transistor. As seen in (1), in order to increase the threshold voltage of an NMOS transistor, body of the transistor should be connected to a voltage level below GND; in order to decrease the threshold voltage of a PMOS transistor, body of the transistor should be connected to a voltage level greater than VDD. However, changing the threshold voltage values of the transistors may cause extra delay. Therefore, threshold voltages of the transistors should be kept at a level where the delay overhead is negligible.

A. ASRAM - CSRAM

Previously proposed Asymmetric SRAM (ASRAM) design aims to reduce the leakage current of SRAM bitcells by changing the threshold voltages of the transistors during manufacturing [3][4]. Because most of the bits inside the storage components are logic- 0, the threshold voltages of the transistors in the bitcell are arranged, in order to keep the leakage low when the content of the cell is logic-0. In Figure 1, the threshold voltage of P2 transistor is kept lower and the threshold voltage of N2 and N4 transistors are kept higher than the normal bitcell. Recently proposed Conscious SRAM (CSRAM) changes the body-bias voltage of the transistors dynamically by observing the content of the bitcell and adapts the threshold voltages of transistors for the bitcell and its 2, 4 or 8 neighbor bits in the same row [7].

Our scheme differs from ASRAM such that we aim to change the threshold voltages of the transistors by adapting body-bias voltages dynamically like CSRAM design. However, as opposed to the CSRAM design that observes the contents of the rows of a storage element and locally changes the body-bias voltages of the bit observed and its 2, 4 or 8 neighbors, our proposed design, observes the content of the whole column and our majority detection circuit discovers if the majority of the bits inside column are logic-1 or logic-0. We then adapt the transistors of all of the bitcells in the same column based on the majority of the bits.

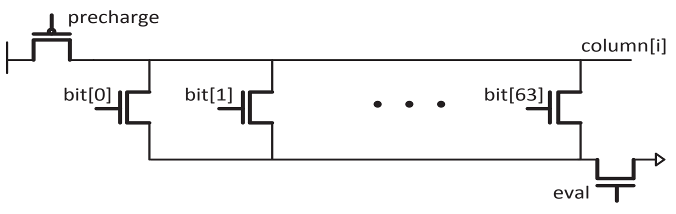

Fig. 2: Majority Detection Circuit

B. Majority Detection Logic

Majority detection logic is responsible for the detection of the number of 0s or 1s in columns of SRAM arrays and it is implemented for each of a column using dynamic CMOS logic in order to achieve faster operation. The circuit for a column (ith column) can be seen in Figure 2. Inputs to the gates of NMOS transistors are connected to the content of the each 8T-SRAM bitcell of the column. The circuit is in fact the comparator of a CAM logic which has one of its inputs always connected to the ground. In a 64-bit register file, 64 such detection circuits have to be used as there are 64 columns in the structure. Although we implemented the circuit with an evaluation transistor, this transistor can be removed in order to reduce the area overhead.

In the regular operation of the circuit in Figure 2, the column node is precharged and it is discharged if there is a 1 in any one of the inputs. Depending on the number of 1s in the input vector the amount of time required for the output node to go to zero varies and by latching the output multiple times it is possible to understand the number of 1s inside the column by looking at the delay of the circuit.

C. Adapting the Columns for Low Leakage

In order to reduce the leakage inside the columns of the storage components, we propose to use the content aware bitcell approach and change the body-bias of the transistors inside the entire column according to the number of 1s inside the columns. If the majority of the stored bits are 1, then all of the bitcells are bias adaptively to leak less assuming that all of them are holding 1. The majority detection logic is employed in order to count the number of 1s inside the column.

The decision of how often the majority detector will be used to detect the number of 1s inside the column is important for the success of the proposed scheme. If the detection logic is employed too frequently, the power dissipation of the circuit itself increases but more effective body-bias control is achieved as the contents of the columns are checked with frequent intervals. On the other hand, if the detection circuit is not employed frequently, the power dissipation of the circuit decreases but there can be inefficiency as during these long time periods the number of logic-0s and logic-1s may change in significant numbers. Therefore, a good compromise point has to be found while choosing period of majority detection.

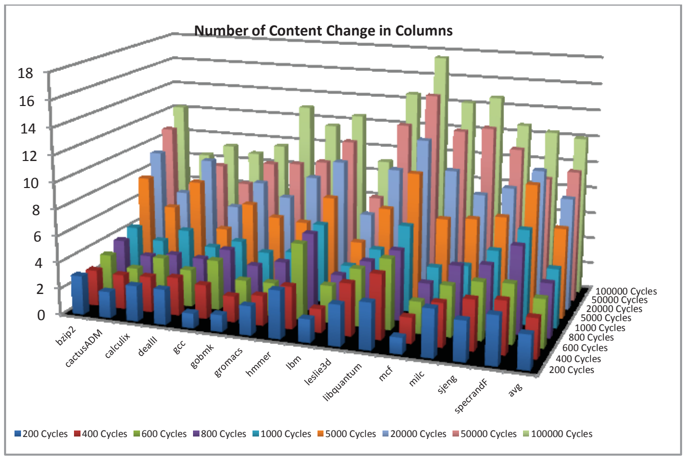

Fig. 3 : Number of Content Change in Columns

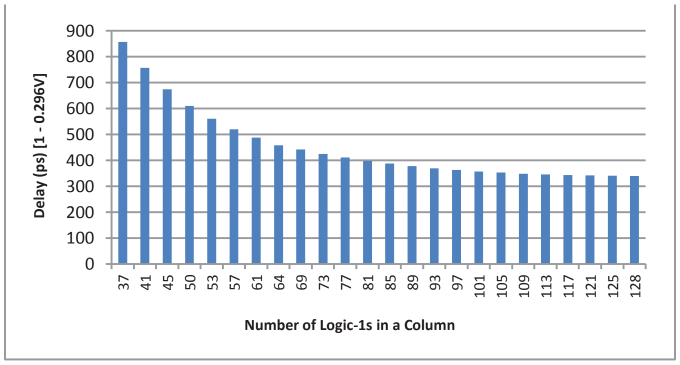

Fig. 4: Time to Set "column_out" Signals to Zero

Figure 3 shows the change of the number of logic-1s (or logic-0s) in SRAM array columns, where different time periods are used to apply majority voting. The figure shows that the number of changes in content of the columns increases drastically after 1000 cycles (light blue bar), yet the difference between 1000 and the lower intervals is not very large. The figure also reveals that, on the average across all benchmarks, the number of 1s inside the columns do not change by just 2 when the period is set as 200 cycles. In choosing the sampling period, it is desirable to have a significant change over the previous state but not a large difference that will lead to lost power-saving opportunities. Thus, we chose the operation frequency of the detection circuit as 1000 cycles in order to optimize the power efficiency vs. accuracy.

In our design, the "eval" signal of the majority detector is activated (set to logic-1) at every 1000 cycles. This signal must be logic-1 for 2 clock cycles (1000 ps for a 2 GHz machine) as we used 2-bit saturating counters which use twice the clock rate of the system (4 GHz) as it was done with the double-pumped ALU of Pentium IV [20]. When the "eval" signal is set to logic-1, the transistors connected to the 8T-SRAM bitcells containing logic-1 start to discharge the capacitance of the output and the time needed to set the *column[i]* signal to logic-0 is related the number of logic-1s that the SRAM array contains. Enable signals of the 2-bit saturating counters are activated (active-high) when the related *column[i]* signal is set to logic-1. Counting operation stops when *column[i]* signal is set to logic-0 and the system clock coming to counters are gated with the *column[i]* signal, which results in the deactivation of the enable inputs of the counters. This way counters do not dissipate any dynamic power until the beginning of the next 1000 cycle period. By adding a sleep transistor to kill the counters and the majority detection logic during the 1000 cycle period, leakage current is also prevented. Majority detection logic also does not dissipate dynamic power as well when it is not in use.

The counter results should be interpreted by a logic structure to control the body-bias voltage of the transistors inside the SRAM array column. The time required to discharge the *column[i]* signal with respect to the number of logic-1 in a column is given in Figure 4. The discharging time for a column having less than 37 logic-1 bits is not shown as the delay in that case mandates the use of a 3-bit saturating counter with a 2 GHz clock. The contents of the counters are checked with control logic after the output node is totally discharged. If the resulting logic-1 count indicated by the saturating counters is more than 64 than the body bias of the transistors are arranged such that they leak less when they hold a value of 1. Since it is not possible to know which cells contain the value of 1 or 0 in this scheme, the cells that contain logic-0 will not be able to provide and power savings.

D. Variation-Aware Control Mechanism

The delay of the majority detection logic varies with different temperatures. Since the saturating counters rely on the discharge time of the *column[i]* signals, their contents can become corrupted with varying temperatures. Therefore, the variation effects must be taken into consideration in order to adapt body-bias voltages of the transistors in the column of storage elements successfully. Table 1 shows the discharge time of the majority detector circuit for different temperatures when there are 50, 64 and 77 1s inside a column. The table also shows the value of the corresponding counter in each case. As shown in Table-1, the count number can be the same if there are 50 or 77 logic-1 values in a column with different temperatures, so one does not simply decide the body-bias configuration of a column by just looking the count number of the counters.

TABLE I. LATENCY VARIATION WITH RESPECT TO TEMPERATURE

	Number of Logic-1s in the Column					
	50		64		77	
Temp(°C)	Delay(ps)	Count	Delay(ps)	Count	Delay(ps)	Count
27	609,72	2	457,4	1	410,52	1
42	658,21	2	480,62	1	443,83	1
57	716,9	2	524,18	2	483,52	1
72	773,56	3	570,26	2	521,2	2
87	829,33	3	624,57	2	566,28	2
102	906,92	3	687,61	2	619,27	2

We observed the latency variation of the majority detection logic and three extra columns are inserted to the structure, to control the body-bias voltages of the transistors in the storage elements, containing: (1) 50 logic-1 bits, (2) 64 logic-1 bits and (3) 77 logic- 1 bits. Bitcells inside these three columns are hardwired to the GND or VDD voltage levels according to the number of logic-1s and logic-0s needed to construct them. All of the counter results related to the different columns of the storage elements are compared with the counter results of these three different hardwired columns: *count_50, count_64, count_77*. Table 1 shows that the count results of these three columns are never the same at the same time.

The comparison circuit works as follows:

- When *count_64* and *count_77* are equal: If the count of *column[i]* is smaller or equal to these two, then the majority of the bits inside the column are logic-1, if the count of *column[i]* is greater than these reference counters, then the majority of the bits inside the column are logic-0.
- When *count_64* and *count_50* are equal: If the count of *column[i]* is greater or equal to these two, then the majority of the bits inside the column are logic-0, if the count of *column[i]* is smaller than them, then the majority of the bits inside the column are logic-1.

The system architecture of our scheme can be seen in Figure 5. Total system is composed of four subunits: Majority detection logic, 2-bit saturating counters, comparison circuits and body-bias voltage controller.

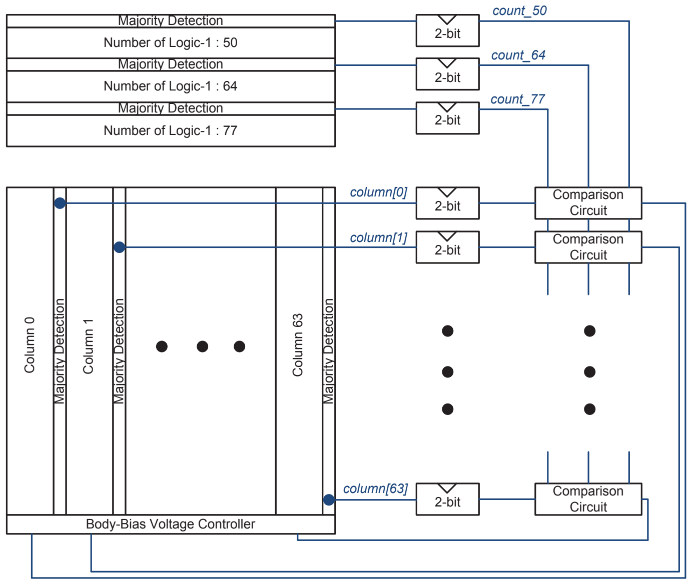

Fig. 5: System Architecture

IV. SIMULATION METHODOLOGY

We used the GEM5 cycle-accurate simulator [12] that is assembled and constructed using two popular simulator infrastructures GEMS [13] and M5 [14], for our simulations. For the workload, several benchmark programs from SPEC2006 [15] benchmark suite that we could compile and run on our simulation infrastructure are executed. We simulated 200 million instructions for warming up caches and other necessary structures for branch prediction and took the results after running another 200 million instructions. Simulation configurations are given in Table 2. Register file is chosen as the storage element to show the simulation output graphics.

Cadence design tools along with the UMC 90 nm technology is used for drawing the layouts and performing power calculations for 8T-SRAM arrays and other circuit components. For changing the body-bias voltage of the transistors we choose two different approaches: First we only adaptively biased PMOS transistors inside the cells as it was done in the previous work [7]. Second, we adaptively changed the body-bias voltages of both PMOS and NMOS transistors. For biasing body-voltage of both NMOS and PMOS transistors twin-tub process technology must be used [19].

TABLE II. SIMULATION CONFIGURATIONS

Machine Width	4-wide fetch, 4-wide issue, 4-wide commit
Window Size	32-entry issue queue, 32-entry load/store queue, 256-entry ROB
Function Units	4 Int-ALU, 4 FP-ALU, 2 Int-Mul/Div, 2 FP-Mul/Div, 4 SIMD unit
Physical Registers	128 registers INT, 128 registers FP
L1 D-cache	32 kB, 8-way set assoc., 64 byte blocks, 1 cycle hit time
L1 I-cache	32 kB, 4-way set assoc., 64 byte blocks, 1 cycle hit time
L2 Unified Cache	2 MB, 8-way set assoc., 64 byte blocks, 10 cycle hit time
Main Memory	4 GB, 64 byte blocks, 150 cycle hit time
Branch Predictor	2 kB local, 8 kB global history table
BTB	4096 entry, 16 tag size
TLBs	128 entry I-TLB, 256 entry D-TLB

V. Results & Discussions

In order to present the benefits of the proposed scheme, we analyzed the content distribution of the register file columns. In Figure 6 (a), the percentage of logic-1 values in columns can be seen. We show the average percentage of the number of logic-1s in all columns for the benchmarks. Moshovos [4] and other researchers previously observed that most of the bits inside the storage components are usually logic-0 values and used this motivation to arrange the threshold voltages of the transistors as they all store logic-0. However, as seen in Figure 6 (a) 35% of the time the majority of the bits is logic-1 inside the columns and if ASRAM [4] is used, the advantage will be lost for the time when majority of the bits inside the columns are logic-1.

In our proposal, we observe the majority of the bits and arrange the body-bias voltages of the transistors inside the column according to the majority of the bit; therefore, it is not important if the majority of the bits are logic-0 or logic-1 such that our scheme decreases static energy dissipation in both ways. In Figure 6 (b), the average percentage of the majority values can be seen. On the average across all spec 2006 benchmarks, for 40% of the time, the logic values inside the columns are 90% logic-0 or 90% logic-1. By looking at this figure our scheme will achieve energy reduction for 90% of the bits inside the columns for 40% of the execution time. From Figure 6 (b) we understand that in columns one logic value is usually much more than the others. For example, the average percentage of the time the logic-1 and logic- 0 numbers are nearly equal (>50) is very low (11%). Therefore, columns usually have a strong majority (>90 or >80), which increases the effectiveness of the proposed scheme.

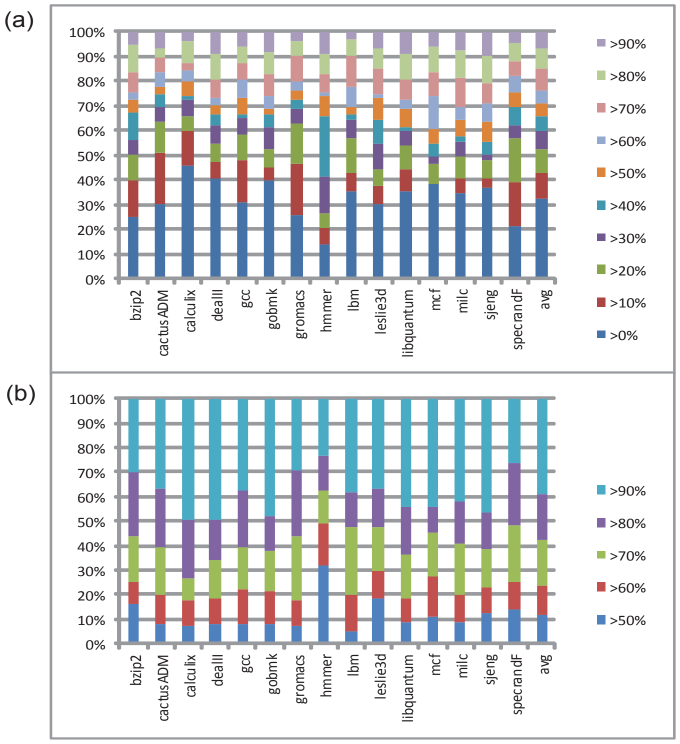

Fig. 6 : (a) Average Percentage of Logic-1s in Columns (b) Average Percentage of Majority of the Value ('0' or '1') in Columns

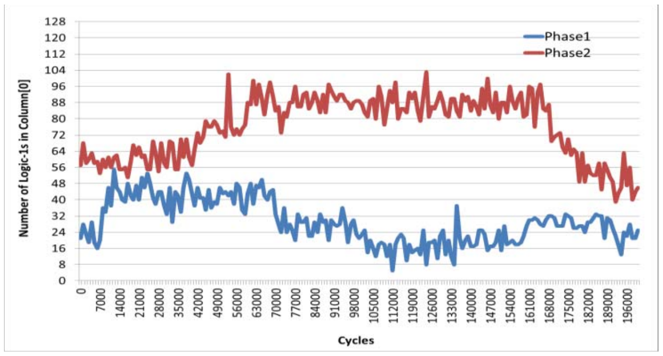

Fig. 7: Observation of two Different Phases

We also observed different phases during execution of the benchmark programs in order to gain insight about the change in the content of columns of SRAM arrays. Since showing all of the results of different phases during the execution for each benchmark is infeasible due to space limitations, we chose to show two different execution phases from the *hmmer* benchmark. Figure 7 shows the number of logic-1s in column[0] which stores the least significant bits of all lines of the register file. We choose two different 200_000 cycle periods for showing the data, one phase is from cycles 60_000_000 to 60_200_000 and the other phase is from cycles 100_000_000 to 100_200_000. As the figure reveals, although the number of ones inside the columns show some instantaneous movements in general there is a continuous progress where the number of 1s change slowly between two sampling periods. Also the figure shows that the same program can show different characteristics in various parts of the execution time.

Figure 8 shows the energy difference related to our baseline model in which the body-bias voltages of the transistors in 8T-SRAM array are kept as normal operation mode (GND for NMOS & VDD for PMOS). For energy simulations, we tried two different configurations: (1) Only body-bias voltages of PMOS transistors were modified according to the majority content, (2) body-bias voltages of both PMOS and NMOS transistors were modified using twin-tub process. For the PMOS transistors, when their body-bias voltages are modified due to content, an adapted bias voltage of 2 Volts is used. For NMOS transistors the adapted body bias voltage level is -1 Volts. When these voltage levels are applied no significant delay overhead occurs for reading or writing to the bitcells as it was also observed in [7]. The temperature is set to 107°C. For calculating the total energy savings of the system, energy dissipation of 2-bit saturating counters, majority detection logic, extra columns for variation awareness and the comparator circuits are added and the contents of the columns are considered.

The total energy savings of the system reaches 8% when only body-bias voltages of PMOS transistors are modified and 14% when body-bias voltages of both PMOS and NMOS transistors are modified by using twin-tub process. The area overhead of the new 8T-SRAM array with column adaptation is 11%.

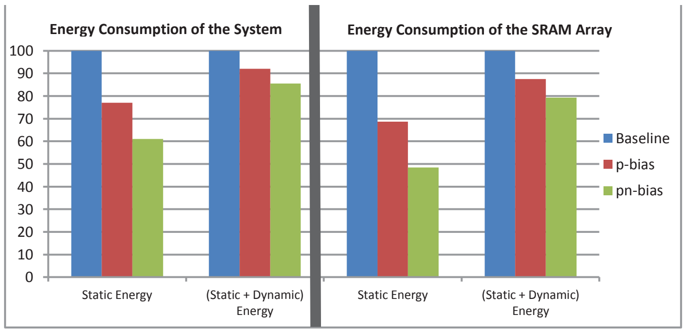

Fig. 8: Normalized Energy Consumptions

VI. Related Work

In their paper, Ahsan et al. proposed that same-content-cell-columns (SCC-columns) occur when all the bits in a column are the same [2]. However, if any of the residing bits in a column is different than the others after a write operation, this scheme is not effective on reducing the energy when writing or reading data. Also, the goal of this research is to reduce the dynamic energy dissipation of the SRAM arrays during write and read operations on the array. Our scheme aims at reducing static energy dissipation of the SRAM arrays, which is independent of read or write operation. Therefore our schemes can be used in conjunction with the scheme of Ahsan et al. to reduce the static and dynamic energy dissipation further.

In asymmetric SRAM (ASRAM) design the threshold voltages of the transistors are fixed during the manufacturing time such that the bitcell dissipates less static power due to leakage when the content of the bitcell is 0 (or 1). However, when majority of the bits are 1 in an SRAM array there is little reduction in leakage energy dissipation and even when all of the bits are 1 in a column there is no reduction at all [3][4]. Our scheme dynamically changes the body-bias voltages of transistors by observing the majority bits of columns. In order to reduce the leakage current of caches, *drowsy cache* scheme is proposed [5]. In drowsy cache, two different supply voltages are used: High-power and low-power. In low power mode, cache blocks cannot be written or read. If there are limited number of blocks frequently accessed in short time period this methods extends its potential. The difference between gated-Vdd [6] control and drowsy cache is the allowance of accessing cache blocks in drowsy cache while one cannot do this for a cache using gated-Vdd control.

Dynamically changing body-bias voltage of transistors based on replication of neighboring bitcell contents' is another implementation to reduce static energy dissipation of SRAM bitcell [7]. In the proposed scheme, additional 8 transistors are used to regulate the body-bias voltage of transistors constructing bitcell and the neighbor bits, whereas in our scheme we observe the majority of the content of a column and change the body-bias voltage of the transistors inside the whole column.

VII. Conclusions & Future Work

In this paper we propose a lightweight, simple but effective method to reduce the static energy dissipation of SRAM arrays by observing the amount of bits which constitutes majority inside a column. Based on the result of the majority detection logic, body-bias voltages of the transistors inside the column are changed to reduce the leakage. We selected 8T-SRAM layouts to calculate the energy savings and achieved up to 39% energy savings for static energy dissipation and up to 14% for total energy dissipation of the SRAM arrays with an area overhead of 11%. It is possible to apply the proposed technique to all SRAM based structures (such as the TLBs, Caches, reorder buffer etc.) inside the processor. Such an analysis is left for future work.

In our phase analysis in this paper we find that some of the columns do not show a huge variance in content numbers. For example the upper most bits of the SRAM array are usually logic-0 for integer benchmarks especially. For such columns, instead of dynamically detecting the number of logic-0s or logic-1s, the body-bias voltage of transistors may be statically arranged and thus no control logic and extra counters may be needed for such columns. Such hybrid schemes are also left for future work.

References

[1] R. Kumar and G. Hinton, "A family of 45nm IA processors," IEEE International Solid-State Circuits Conference - Digest of Technical Papers, ISSCC, May 2009.

[2] B. Ahsan, et al., "Eliminating energy of Same-Content-Cell-Columns of On-Chip SRAM arrays," International Symposium on Low Power Electronics and Design, ISLPED, August 2011.

[3] N. Azizi, F. Najm, and A. Moshovos, "Low-leakage asymmetric-cell SRAM," IEEE Transactions on VLSI Systems 2003.

[4] A. Moshovos, B. Falsafi, F. Najm and N. Azizi "A Case for Asymmetric-Cell Cache Memories," IEEE Transactions on VLSI Systems 2005.

[5] K. Flautner, Nam Sung Kim, S. Blaauw and T. Mudge "Drowsy caches: simple techniques for reducing leakage power," ISCA, August 2002.

[6] M. Powell, et al. "Gated-Vdd: a circuit technique to reduce leakage in deep-submicron cache memories," International Symposiım on Low Power Electronics and Design, ISLPED, 2000.

[7] F. Koç, O. Simsek and O. Ergin "Using Content-Aware bitcells to reduce static energy dissipation" International Conference on Computer Design, ICCD, November 2011.

[8] A. Malik, B. Moyer and D. Cermak "A low power unified cache architecture providing power and performance flexibility," ISLPED 2000.

[9] K. Tanaka "Cache memory architecture for leakage energy reduction," Innovative Architecture for Future Generation High-Performance Processors and Systems, IWIA, 2007.

[10] X. Guan and Y. Fei "Reducing power consumption of embedded processors through register file partitioning and compiler support," Intl. Conf. on Application-Specific Systems, Architectures and Processors, Jul 2008.

[11] V. Zyuban, et al. "Power optimization methodology for the IBM POWER7 microprocessor," IBM Journal of Research and Development, 55(3), May-Jun 2011.

[12] N. Binkert, et al. "The gem5 simulator," SIGARCH Computer Architecture News, 39:17, Aug 2011.

[13] M. Martin, et al. "Multifacet's general execution driven multiprocessor simulator (GEMS) toolset," Computer Architecture News, Sept 2005.

[14] N. Binkert, et al. "The M5 simulator: Modeling Networked Systems," IEEE Micro, 26(4), Jul/Aug 2006.

[15] Standard Performance Evaluation Corporation, The SPEC CPU 2006 Benchmark Suite, http://www.specbench.org

[16] C. Keltcher, K. McGrath, A. Ahmed and P. Conway "The AMD Opteron processor for multiprocessor servers," IEEE Micro, Mar/Apr 2003.

[17] R. Singhal "Inside Intel® next generation Nehalem microarchitecture," Hot Chips 20, Stanford, CA, 2008.

[18] H. Noguchi, et al. "Which is the best dual-port SRAM in 45-nm process technology? – 8T, 10T single end, and 10Y differential," IEEE International Conference on Integrated Circuit Design and Technology and Tutorial, Jun 2008.

[19] L. Parillo, et al. "Twin-tub CMOS – A technology for VLSI circuits," International Electron Devices Meeting, 1980.

[20] G. Hinton, et al. "A 0.18-µm CMOS IA-32 processor with a 4-GHz integer execution unit," IEEE Journal of Solid-State Circuits, Nov 2001.

[21] K. Mistry, et al. "A 45nm logic technology with high-k+metal gate transistors, strained silicon, 9 Cu interconnect layers, 193nm dry patterning, and 100% Pb-free packaging," IEEE International Electron Devices Meeting, Dec 2007.