DCC: Double Capacity Cache Architecture for Narrow-Width Values

Mohsen Imani, Shruti Patil, Tajana Rosing CSE Department, University of California San Diego, La Jolla, CA 92093, USA {moimani, patil, Tajana}@ucsd.edu

ABSTRACT

Modern caches are designed to hold 64-bits wide data, however a proportion of data in the caches continues to be narrow width. In this paper, we propose a new cache architecture which increases the effective cache capacity up to 2X for the systems with narrow-width values, while also improving its power efficiency, bandwidth, and reliability. The proposed double capacity cache (DCC) architecture uses a fast and efficient peripheral circuitry to store two narrow-width values in a single wordline. In order to minimize the latency overhead in workloads without narrow-width data, the flag bits are added to tag store. The proposed DCC architecture decreases cache miss-rate by 50%, which results in 27% performance improvement and 30% higher dynamic energy efficiency. To improve reliability, DCC modifies the data distribution on individual bits, which results in 20% and 25% average static-noise margin (SNM) improvement in L1 and L2 caches respectively.

Keywords

Cache, SRAM; NBTI; SNM; Low power design

1. INTRODUCTION

Scaling the CMOS technology creates two new challenges: static power and device reliability. This has been a major concern for the design of integrated circuits. Due to the exponential dependency of sub-threshold current to the threshold voltage, the static power increases significantly as technology nodes shrink [1, 2]. In addition, increasing temperatures and current density of devices degrades lifetime and performance of CMOS technology due to the aging mechanisms such as bias temperature instability (BTI) and hot carrier injection (HCI) [3]. The negative bias temperature instability (NBTI) is an important factor that causes aging and reliability degradation of PMOS devices. This phenomenon increases the absolute threshold voltage of PMOS transistors and slows down circuits by up to 20% slower [4]. In memories, NBTI increases the probability of failure during the expected lifetime due to the soft-errors [5, 6]. Since embedded SRAM comprises a large portion of chip area and holds critical data for processing, it is crucial to improve performance, power, bandwidth and reliability in the face of such scaling challenges [7].

In modern microprocessors, the word length has increased to 64-bits to enable higher memory addressability, however much of the data continues to be 32-bits wide or less [8-11]. This is due to availability of the high number of small and zero values on the processor computation. Such narrow width values waste valuable memory capacity when represented as 64-bits values. Further, non-uniform data distribution of narrow-width values on upper 32-bits of the wordline decreases memory/register file lifetime due to negative bias temperature instability (NBTI) degradation [3].

In this paper, we introduce double capacity cache (DCC) architecture which stores two narrow-width data on a single wordline in order to increase the effective cache capacity up to 2X. We define four modes

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from <u>Permissions@acm.org</u>.

GLSVLSI '16, May 18-20, 2016, Boston, MA, USA © 2016 ACM. ISBN 978-1-4503-4274-2/16/05\$15.00 DOI: http://dx.doi.org/10.1145/2902961.2902990 that determine the data placement in cache line, and allow the system to write/read in half or full wordline schemes. The DCC design incurs low overheads even for workloads without narrow-width values. To meet the cache deadlines, we use different DCC implementations on L1 and L2 caches. Our evaluations on SPEC CPU 2006 benchmarks shows higher effective cache capacity in DCC decreases cache miss-rate and improves the system performance by 27% on average. Half write/read operation and lower cache miss-rate decrease the dynamic power of system by 30% and 35% on L1 and L2 caches. Further, DCC improves the average static-noise margin (SNM) degradation by 20% and 25% in L1 and L2 caches respectively.

2. RELATED WORK

Several previous efforts improve soft-error protection or power/performance efficiency of the narrow-width values in register files and caches [8-11]. Brooks et al. [8] implemented clock gating technique to turn off non-active part of narrow-width values in register file. Using register file with different sizes is another method to decrease the power and area overhead in narrow-width values [12]. However, finding the exact size of narrow-width data and assigning it to an appropriate register file consumes significant power. Hu et al, in [13] used narrow-width characteristics to make a duplicate of the value within the same data item. This in-register duplication (IRD) eliminates the requirement for additional copy registers and improves the architectural vulnerability factor of register files. The work in [9] introduced a thermal behavior of a low-power value-aware register file (VARF) which relates to internal hotspots on register file. They proposed three thermal control techniques and added two flag bits to balance the access activity across partitions of VRAF by shifting the narrow-width data between the upper and lower 32-bits halves of register file. Kong et al, [14] designed a process tolerant last level cache in 3D microprocessors. They divide data values into four groups with 16-bits, 32-bits, 48-bits and 64-bits narrow-width granularity which allows them to store more data in cache with the same storage capacity. Wang et al, in [10] utilized an extra flag bit to keep track of the register file lines with narrow-widths. They considered values of lower 34-bits as narrow-width data implement power gating method for the other 30-bits to reduce register file power consumption.

Although power/clock gating of non-active part of narrow-width data improves power efficiency, it is not an efficient solution since it generally wastes about half of register file capacity or cache area. In addition the aging-aware techniques used extra flag bits to balance the data distribution. An efficient method should be able to use the inactive part of narrow-width data to increase the cache utilization or balance the data with no performance overhead.

3. DOUBLE CACHE CAPACITY

In this paper we leverage the narrow-width value observation to improve the power efficiency for caches with narrow-width values. Our proposed architecture, double capacity cache (DCC), adds two flag bits to cache architecture to detect the narrow-width values and stores their valuable lower 32 bits. We use the upper 32-bit memory available to store another narrow-width value. Modifying data placement in such a way improves power, performance, bandwidth and lifetime of caches in systems with narrow-width values. In the best case, this technique improves the effective cache capacity and memory bandwidth by up to 1.78X depending on the percentage of narrow-width values used by the workload.

To implement DCC, we add flag bits (HalfW & UpW) to the tag store. This increases the effective cache size (cache associativity) by making the tag store larger. Flag HalfW indicates whether the data read/write should be performed with half or full length. Hal/W=1 shows that only one of the data entry is valid, and only half of the wordline is necessary for the read or write. We therefore activate partial read/write circuitry as required. This significantly improves the power efficiency of the memory access. In contrast, Hal/W=0 indicates that valid data uses full cache line length (64-bits mode). In this case, the data can be either 64bits wide or a combination of two 32-bits values. The second flag bit (UpW) provides information about the data placement or width based on the first flag bit. When the system is in half mode (Hal/W=1), UpWindicates whether data is placed on upper (UpW=1) or lower 32-bits (UpW=0) of cache line. In full-length mode, this flag determines whether the value in cache line is wide-width data (UpW=0) or the combination of two narrow-width data (UpW=1). Using the two flag bit values, the proposed cache has four different modes as follows (see Figure 1):

(*i*) *HalfW*=0, *UpW*=0:

In read mode, this bit combination shows that the value in this wordline is wide-width. The system puts the original data {upper half, lower half} on the output bus. In write mode if input data is not narrow-width, we reset both flags to zero.

(*ii*) *HalfW*=1, *UpW*=0:

This combination shows that one value in cache line is narrowwidth and it saved on the lower 32-bits of cache line. In read mode the peripheral circuitry puts $\{32 b 0, Lower 32-bits\}$ on output bus. In write mode, the system writes the lower 32-bits of this data on the least significant 32-bits of cache line.

(iii)HalfW=1, UpW=1:

This mode indicates that the data stored data on cache is narrow-width and it is saved in the upper 32-bits of cache entry. In read mode, the upper 32-bits of cache line are written on the lower 32-bits of the output bus as $\{32'b\ 0,\ Upper\ 32-bits\}$. During write operation, the system writes the first 32-bits of data in upper half of cache line.

(iv) HalfW=0, UpW=1:

Both lower and upper halves of wordline cache contain narrow-width data and in read mode, the peripheral circuitry reads the entire 64-bits wordline {*Upper 32-bits*, *lower 32bits*}. If the read is from processor, only the desired word is put on the output bus, while if it is from a lower level cache, both words are put on the bus for higher bandwidth. In write mode, the system writes the whole wordline on cache in full length to increase the cache bandwidth utilization (double narrow-width write).

The flag bits can be accessed in parallel with address decoding, without adding extra latency to the system. In L1, tag store and data store are also often accessed together to reduce latency, while in L2, tag store and data store are typically accessed in serial order. Thus, in case of narrow-width data in L2, the system only needs to read half of the data line once tag is read.



Figure 1. Different modes of proposed DCC based on flag bits.



Figure 2. Proposed DCC architecture.

4. CACHE RELAIBILITY

Our simulation results show that non-uniform data activity (0% or 100%) has the most effect on SNM degradation of cell. We use NBTI model in [15] to calculate the shift in threshold voltage of PMOS transistor. In order to find the SNM degradation after three years, we used HSPICE to simulate a conventional 6T-SRAM cell with 45nm predictive technology model. For pull-up PMOS transistors we changed the threshold voltage of transistor based on different data activity (see Table 1). Our evaluation shows that after three years with zero data activity (one PMOS always biased with zero) the threshold voltage decreases the SNM stability by 14.9% with respect to baseline 6T cell. Having ones and zeros 50% of the time on SRAM storage nodes results in lowest SNM degradation. In this case, the SNM degradation is 2.9%. Table 1 shows the SNM degradation in SRAM cells for a range of data activity.

Table 1. SNM degradation on SRAM at different data activity.

Data activity	0%	10%	20%	30%	40%	50%
SNM degradation	14.9%	11.3%	8.6%	6.3%	4.1%	2.9%

5. EXPERIMENTAL SETUP

We use GEM5 simulator [16] to extract data traces from L1 and L2 caches when executing SPEC 2006 benchmarks. We fast-forward 500 million instructions and then simulate the next 1 billion instructions in detail. The processor configuration is shown in Table 2. Our evaluation with SPEC CPU 2006 benchmarks shows that more than 87% and 85% of data in L1 and L2 caches respectively can be represented by using only the lower 32-bits. We use *Synopsys Design Compiler* tool in 45nm TSMC technology to evaluate the effect of power, delay and area of peripheral circuitry. The extracted data is added to the number of cycles in L2 cache in GEM5 simulator.

- T 1 1	<u> </u>	D I			<u>~</u>	
Tabla	•	Raca	ina	nrococcor	continurati	on
	1.	Dase		DIDUCESSOL	COMPRESSION	
1 4010		20000		p10000000	• • • · · · · · · · · · · · · ·	· · · ·

	¥ ¥	
Frequency	2 GHz	
L1 Cache	32KB, 2-way association, 64B cache line	
L2 Cache	512KB, 8-way association, 64B cache line	
Memory	DRAM, DDR3, 4GB	

5.1. Dynamic energy

Based on cache traces from the conventional cache and the DCC cache, and using hit/miss access energy from NVSIM tool, the normalized energy consumption of L1 and L2 caches is depicted on

Figure **3**. The first bar in every axis represents the normalized read, write and miss energy of the conventional cache. Total dynamic energy of proposed L1 and L2 cache improves by 30% and 35% on average.

This improvement is due to:



Figure 3. Dynamic energy consumption in conventional and DCC caches.

Effective cache capacity (Miss rate): Higher effective cache capacity on DCC decreases the average miss rate in L1 and L2 caches. This reduction directly improves the energy consumption of the system.

Half read/write of narrow-width data: In DCC, the number of narrow-width reads/writes is high. Each consumes approximately half the energy of a full read/write in L2. This is because we activate only half of the cache wordline for read/write on half values. However merging the data after a read operation adds energy overheads. In L2 cache this overhead is the energy of multiplexers, while in L1 cache, this comes from input/output buffer circuitry that is controlled by *HalfW* and *UpW* signals. This energy overhead is shown as a red bar in

Figure 3.

Write back bandwidth: When the data misses in L1 cache, the system sends a request to L2 cache to find that address. In case of a hit, a part of L2 data is transferred to L1 to improve data locality. In DCC, this operation can be performed at up to $2\mathbf{X}$ higher rate with lower energy consumption. The average time that the system is in 2X bandwidth mode is directly proportional to the percentage of narrow-width values, since each wordline now carries two data values and handles two write operations in the lower level of cache. In addition, in a write-back operation that updates the cache data in higher memory levels, we have a similar advantage in terms of energy improvement.

5.2. Performance

The second DCC opportunity is using the higher cache capacity to improve the system performance. The effect of this capacity increase is significant, especially for data-intensive workloads with higher data availability/locality and reduced number of cache misses. When both L1 and L2 caches are implemented as DCC, the system also benefits from double rates of data transfer between caches, or cache to register file. This advantage is lost when only L1 or L2 is implemented as DCC. Figure 4 compares the performance of the proposed architecture with a conventional cache for SPEC CPU2006 benchmarks. For single L1 and L2 DCC architectures, performance improves by 19% and 16% respectively. There are two reasons why implementing DCC in L2 cache has higher overhead than DCC in L1; first, the system performance is mostly sensitive to L1 effective size rather than L2 cache size. Second, the access to data and tag stores is done in parallel in L1 cache to decrease the system latency while in L2 cache these accesses are serial. As Figure 4 shows, implementing both L1 and L2 as DCC architectures enhances the system performance by 27% on average compared to baseline.



Figure 4. Normalized performance of implementing DCC technique on L1, L2 and both L1 & L2 caches.



5.3. Bandwidth

The ability of DCC to read and write double narrow-width data (on same line) can increase the data cache bandwidth up to 2X during transfer of narrow-width values between caches. When flag bits are $\{0, 1\}$, both upper and lower halves of wordline store narrow-width data. In this case, both data values are read simultaneously. This increases the cache bandwidth up to 1.78X for narrow data. This maximum cache bandwidth improvement is less than double rate since during the miss or write back operation the tag store uses 12-bits as tag values. For workloads with high data locality this increment has major impact on system performance. This advantage is available only when both levels of cache are implemented as DCC. Figure 5 shows normalized total bandwidth to/from the caches for different benchmarks when both L1 and L2 are utilized DCC architecture. The results show that the DCC L1 and DCC L2 architecture can achieve 1.45X and 1.61X higher bandwidth (# of bits per second) with respect to conventional L1 and L2 caches respectively. This reflects the number of dense transfers that are accomplished during execution of the benchmarks. Note that the additional bandwidth may also be used to transfer up to cache blocks per read (instead of a single cache block as is done typically) if narrow-width values are present. However, in this paper, we instead experiment with transferring 64-bit sets of data stored in the cache lines.

5.4. Area and Static Power

Figure 6 shows the leakage power and area of L1 and L2 cache in conventional and DCC implementations. We estimate area and power of caches with the NV*sim* [17] tool in 45nm technology. We consider the impact of the peripherals on power and area of both L1 and L2 caches. The area overhead of DCC L2 cache due to the added multiplexers and flag bits is less than 14% of conventional L2 cache area. As Figure 6 shows, double tag size in L1 cache increases the area and leakage power, but this increment is less than 16% and 9% total cache overhead respectively.



Figure 6. Leakage power and area of conventional and DCC caches. As we noted before, DCC incurs very low overhead for workloads without narrow-width values. Our evaluations show that the power overhead of proposed system (due to extra flag bits in tag store) is close to zero when the caches must store 12% or more narrow-width values during execution in the benchmark. These proportions are observed in most CPU/GPU workloads since they work with many zeros and small values. The other advantage of DCC is that power gating can be applied to a larger area of cache. Compressing narrow-width data and saving two of them in single word line, increases the effective cache capacity by up to 2X. This provides more idle space in the cache to implement power gating. This is true especially in large memory units such as L2/LLC, where the leakage power is critical. In theory, if all data were narrow-width, this technique could achieve up to 2X power saving. However, this is difficult to achieve in L1 cache since the L1 cache has higher activity and is typically nearly full with data cached from higher level caches/memory.

5.5. NBTI improvement

DCC cache not only improves the effective cache capacity, power, performance and bandwidth but also has significant effect on NBTI degradation of cache structure. As Figure 2 shows, our DCC architecture does not change the data distribution of critical bits. For example the bit indices 1 to 4, 30 to 36 and 62 to 64 have still high percentage of zeroes in proposed architecture. Thus, in the worst case SNM does not change with respect to conventional cache. However, improving the average SNM in cache structures results in lower cell soft-error rate. This is because the soft-error rate in a cache is a function of the average SNM degradation. We analyze the impact of BTI on average SNM degradation. Figure 7 shows the average SNM improvement of the proposed architecture for L1 and L2 cache after three years. The modified data distribution in DCC L1 and L2 caches improves the average SNM degradation by 25% and 29% with respect to conventional L1 and L2 caches.



Figure 7. SNM degradation on different benchmarks on L1 and L2 conventional caches and DCC.

6. CONCLUSION

This paper introduces a new double capacity cache (DCC) architecture which can store up to $2\mathbf{X}$ data by exploiting the presence of narrow-width values. The proposed cache has four modes of operation for each cache line defined using two added flag bits. These determine data placement and allow the system to save two narrow-width values on a single wordline in order to increase the effective cache capacity. Implementing DCC architecture for all levels of cache allows us to have double cache bandwidth for data transfers between caches. This, along with larger effective cache capacity, improves cache miss rate and cache performance. In addition, storing narrow data on upper and lower 32-bits of a cache entry modifies the cache data distribution and improves the average SNM degradation.

7. ACKNOWLEDGMENTS

This work was sponsored by NSF grant #1527034 and UC San Diego Jacobs School Focht-Powell Fellowship.

8. REFERENCES

 N. S. Kim, et al., "Leakage current: Moore's law meets static power," *IEEE computer*, vol. 36, pp. 68-75, 2003.
 M. Jafari, et al., "Design of an ultra-low power 32-bit adder operating at subthreshold voltages in 45-nm FinFET," *IEEE DTIS*, pp. 167-168, 2013.

[3] S. Borkar, "Electronics beyond nano-scale CMOS," IEE/ACM DAC, pp. 807-808, 2006.

[4] M. A. Alam, et al., "A comprehensive model of PMOS NBTI degradation," *Microelectronics Reliability Elsevier*, vol. 45, pp. 71-81, 2005.

[5] M. Imani, et al, "Low Power Data-Aware STT-RAM based Hybrid Cache Architecture," *IEEE ISQED*, 2016.

[6] M. Imani, et al., "Hierarchical design of robust and low data dependent FinFET based SRAM array," *IEEE/ACM NANOARCH*, pp. 63-68, 2015.

[7] M. Ansari, et al., "Estimation of joint probability density function of delay and leakage power with variable skewness," IEEE ICECCO, pp. 251-254, 2013.

[8] D. Brooks, et al., "Dynamically exploiting narrow width operands to improve processor power and performance," *IEEE HPCA*, pp. 13-22, 1999.

[9] S. Wang, et al., "Exploiting narrow-width values for thermal-aware register file designs," *IEEE/ACM DATE*, pp. 1422-1427, 2009.

[10] S. Wang, et al., "Low power aging-aware register file design by duty cycle balancing," *IEEE/ACM DATE*, pp. 546-549, 2012.

[11] G. Duan, et al., "Exploiting narrow-width values for improving non-volatile cache lifetime," *IEEE/ACM DATE*, 2014.

[12] A. Aggarwal, et al., "Energy efficient asymmetrically ported register files," *IEEE ICCD*, pp. 2-7, 2003.

[13] J. Hu, et al., "On the exploitation of narrow-width values for improving register file reliability," *IEEE TVLSI*, vol. 17, pp. 953-963, 2009.

[14] J. Kong, et al., "Exploiting narrow-width values for process variation-tolerant 3-D microprocessors," *IEEE/ACM DAC*, 2012, pp. 1197-1206.

[15] T. Grasser, et al., "The paradigm shift in understanding the bias temperature instability: from reaction–diffusion to switching oxide traps," *IEEE Electron Devices*, vol. 58, pp. 3652-3666, 2011.

[16] N. Binkert, et al., "The gem5 simulator," ACM SIGARCH Computer Architecture News, vol. 39, pp. 1-7, 2011.
[17] X. Dong, et al., "Nvsim: A circuit-level performance, one ray, and area model for omerging neurolatile memory."

energy, and area model for emerging nonvolatile memory," IEEE ICCAD, vol. 31, pp. 994-1007, 2012.