Error Protected Data Bus Inversion Using Standard DRAM Components

Maurizio Skerlj
Qimonda AG
Server Memory Systems Engineering
D-81739 Munich, Germany
maurizio.skerlj@qimonda.com

Paolo Ienne
Ecole Polytechnique Fédérale de Lausanne (EPFL)
School of Computer and Communication Sciences
CH-1015 Lausanne, Switzerland
paolo.ienne@epfl.ch

Abstract

Off-chip communication consumes a significant part of main memory system power. Existing solutions imply the use of specialized memories or assume error free environments. This is either unrealistic or impractical in many industrial situations. In this paper, we propose an architecture which implements classic low power encoding but uses industry standard DRAMs. Moreover, the low power encoding is combined with error-protection in order to extend the application to noisy channels or to the presence of soft and hard failures in the memory. Parallelism between the two encoding processes avoids any latency adder. Our experimental results, based on current consumption measurements of DDR2 DRAM components in mass production, show savings up to 31% on the I/O power and 6% on the total memory energy of a single channel memory system of 4GB at practically no cost.

1. Introduction

In recent years, large data centers deploying high-end, multiprocessor systems running commercial workloads started to face energy-use constraints. As transistor density and demand for computing performance rapidly increase, the systems are approaching the limits of what can be built without introducing expensive techniques such as liquid cooling. Commercial workloads are typically memory intensive, therefore high-end servers comprise one or more high-performance processors and large amounts of dynamic random–access memory (DRAM). In fully equipped high-end systems the amount of power consumption of the memory, associated with the power consumption of cooling fans, can reach 50% of the total [12]. Moreover, during lifetime, the energy consumption cost exceeds the cost of the hardware [7]. Limiting the peak power consumption will also help to decrease associated costs such as thermal design and data center power distribution. Therefore, any proposal that reaches tangible power consumption reduction in an economically viable way is of capital importance in such applications.

This paper focuses on reducing the power consumption of high-speed off-chip data buses present in high-end memory systems. The so far proposed techniques for lowering the power spent in the interconnect assume an error-free environment, or the possibility of using specialized memory components. The actual constraints of memory systems in commercial high-end servers are such that the proposed concepts cannot be applied to main memory in a straightforward way. We propose an architecture that combines low power encoding and error protection suitable for noisy environments. Moreover, all the coding can be performed in the memory controller, without the need for additional interconnects, so that conventional DRAMs can be used. In Section 1.1 a detailed explanation of the problem is given. Section 2 gives an overview of related and previous work. Mathematical description of the proposed coding is given in Section 3 and in Section 3.1 its application to commercial memory systems is shown. Power savings results are reported in Section 4, while the achieved reliability of the proposed system is analysed in Section 5. A statistical model that takes into account hard, soft, and communication failures is developed in Section 5.1, the rationale for the parameter choice is given in Section 5.2 and its numerical simulation results for different memory systems are reported in Section 5.3.

1.1. Problem Statement

Currently, the bus-invert technique [8, 23] is implemented in memory systems according to the scheme of Fig. 1a: the encoding and decoding of the bus is performed at user end as well as at memory side and the

0-7695-3117-2/08 $25.00 © 2008 IEEE
DOI 10.1109/ISQED.2008.74

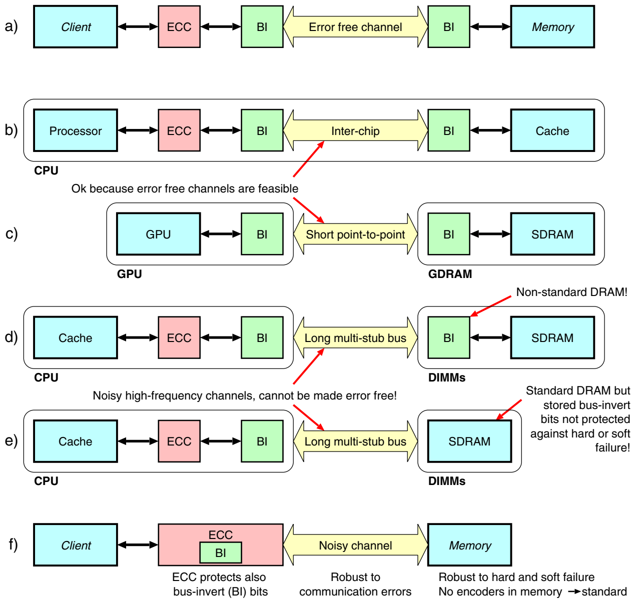
a)
Client
ECC
BI
Error free channel
BI
Memory
b)
Processor
ECC
BI
Inter-chip
BI
Cache
CPU
Ok because error free channels are feasible
c)
GPU
BI
Short point-to-point
BI
SDRAM
GPU
GDRAM
Non-standard DRAM!
d)
Cache
ECC
BI
Long multi-stub bus
BI
SDRAM
CPU
DIMMs
Noisy high-frequency channels, cannot be made error free!
Standard DRAM but stored bus-invert bits not protected against hard or soft failure!
e)
Cache
ECC
BI
Long multi-stub bus
SDRAM
CPU
DIMMs
f)
Client
ECC
BI
Noisy channel
Memory
ECC protects also bus-invert (BI) bits
Robust to communication errors
Robust to hard and soft failure
No encoders in memory → standard

Figure 1. Bus-invert usage in memory systems. The principle of (a) is successfully implemented as in (b) or (c) for error-free communication channels (on-chip or point to point channels). Off-chip, limited reliability either requires nonstandard components (d) or limits severely reliability to memory failures (e). Our encoding scheme (f) combines low-power and reliability for commercially viable systems.

communication between user and memory is assumed error free. Commercial implementation of this scheme can be found in communication between microprocessor and on-chip caches (Fig. 1b, [9]) or in commercial DRAMs for graphic applications (Fig. 1c, [10]). In both examples the assumption about error free communications holds true due to the high quality of the interconnect. Also, due to the specialized use of such memory systems, the integration of the codec on the memory side is economically viable. This approach is hardly extensible to large memory systems.

In typical main memory subsystems for server applications there are large data buses connecting memory modules (DIMMs, Dual Inline Memory Modules), each module having a large number of DRAM components organised in ranks. In this case, integrating a codec in the DRAM components is disadvantageous as the number of inversion signalling bits will have to match the number of DRAM components per rank. Such pin-count increase is not sustainable as it would result in larger module connectors, larger routing channels on the motherboard, larger pin-count on the packages, and the impossibility of relying on economies of scale since those components will differ from the mainstream commodity memories. The system cost increase would therefore be significant. Moreover, the long stub bus connecting all the memory devices might not be error free at high frequencies (Fig. 1d).

If coding and decoding for low power is performed at the memory controller only (like in Fig. 1e), the inversion signalling bits have to be stored in the memory. The memory content can be corrupted by soft failures caused by alpha particles or cosmic rays [27, 15, 3]. As the coding for low power has to be the outermost coding, a straightforward combination with error protection coding would lead to unprotected signalling bits resulting in poor system reliability.

This paper proposes an architecture (Fig. 1f) that protects the bus-invert signalling bits in order to achieve resilience toward single-bit errors (due to either communication errors or failures in the memory). Moreover, by encoding large blocks of data, a lower level of redundancy is needed. The bits freed are used for bus-invert signalling and can be stored in the main memory without the need of additional signals and pins. We will therefore show in Section 5.3 that, with appropriate encoding on the processor side, tangible power savings can be achieved while using unmodified industry-standard DRAMs without any loss of overall system reliability.

2. State of the Art

The bus-invert method was originally described in a patent by Fletcher [8] and later further analysed by Stan and Burleson [23, 24]. It is a simple method which achieves 50% decrease in peak I/O power dissipation and 25% decrease in average I/O power dissipation in unterminated buses. In high-speed communication systems, the I/O power consumption is dominated by line termination. In this case bus-encoding techniques have to be combined with tailored termination schemes [6]. In the above mentioned works the communication channel is assumed error free. This is not always the case and, as we will later show, implementing bus-invert techniques with noisy channels leads to poor system reliability and resilience.

Memory systems for mission critical applications implement error correction or detection schemes in order to increase the communication reliability and the resilience toward soft and hard failures. The combination of low power encoding with error correction and detection was already explored by Sridhara and Shanbhag [21]. In this work it is observed that, for such a combination, the error protection has to be the outermost code. The additional latency of two sequential coding steps can be avoided using a method developed by Mulla and Tu [19], but this work does not explore the impact of noisy channels. Bertozzi showed that the lowest power consumption is achieved by error detection and retransmission as opposed to error correction [4]. This technique, originally proposed for on-chip communications, cannot be applied to memory systems, where soft errors corrupt the memory content [27]. We suggest and analyze in detail a viable combination of bus-invert coding and error correction in the presence of noisy channels and both soft and hard failures.

Moreover, activity on the communication bus can be reduced by storing the most frequently transmitted messages in a cache and transmitting the relative cache indexes instead [1, 13], or by using frequent value data-bus encoding [25]. Our scope are main memory systems where large data buses connect memory modules, each module having a large number of DRAM components. Such data bus is divided in bytes or nibbles connected in parallel to all memory ranks. In this case the use of the aforementioned methods will prove disadvantageous, as the overhead will be present in each and every DRAM component, which contributes to the data bus with a small number of bits. Moreover, in multi-rank memory systems, only the DRAMs in the referred rank contribute to the data bus, further lowering the efficiency of such schemes.

3. Exploiting Code Linearity

In bus-invert coding, data bits are conditionally inverted based on some metric—e.g., bus activity or frequency of symbols that consume more power in line terminations. An additional bit, signalling the inversion, makes possible to retrieve the original message. The bus-invert encoding shows poor resilience against errors, as single errors hitting the bus-invert signalling bit will result in multiple errors (all the data bits will be incorrectly inverted by the decoder—hence all bits will be wrong).

In order to avoid that an erroneous inversion signalling bit leads to wrong decoding, also the inversion signalling bit should be protected with ECC (Error Correction Code). By using the linearity of ECC codes, the inversion signalling bit can be error-protected and the ECC calculation on the original word x can still be performed in parallel. For the sake of simplicity let us now assume that the bus inversion is performed on the whole word x—the case of inverting chunks of x will lead to more inversion signalling bits and the described method can be easily generalised.

First, let us extend x with as many zeros as the number of bus-invert signalling bits, leading to $[x, 0]$ in case of whole word inversion. Let G be the generator matrix of the code. Let $y = [x, 0] \cdot G$ be the encoded word. Inverting x is equivalent to adding modulo 2 the all-ones vector $\vec{1}$, or $\bar{x} = (x \oplus \vec{1})$. Using the linearity,

$$[\bar{x}, 1] \cdot G = ([x, 0] \oplus [\vec{1}, 1]) \cdot G = y \oplus [\vec{1}, 1] \cdot G. \qquad (1)$$

If we express G in canonical form, then $[x, 0] \cdot G = [x, 0, c]$, with c being the check-bits. Then $[\vec{1}, 1] \cdot G = [\vec{1}, 1, c']$. Therefore, in case of bus inversion, the word to be transmitted is

$$[\bar{x}, 1, c \oplus c']. \qquad (2)$$

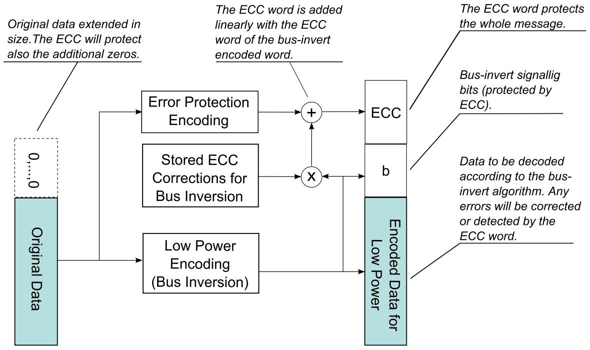
Original data extended in size.The ECC will protect also the additional zeros.
The ECC word is added linearly with the ECC word of the bus-invert encoded word.
The ECC word protects the whole message.
Bus-invert signallig bits (protected by ECC).
Data to be decoded according to the bus-invert algorithm. Any errors will be corrected or detected by the ECC word.
0,...,0
Original Data
Error Protection Encoding
Stored ECC Corrections for Bus Inversion
Low Power Encoding (Bus Inversion)
+
x
ECC
b
Encoded Data for Low Power

Figure 2. Exploiting code linearity. By using the linearity of algebraic codes, the ECC word is corrected in order to protect the message also after the encoding for low power.

One should note that c' is a fixed value for a given G and can thus be precomputed and stored. This means that effectively the ECC encoding and the bus inversion can still be done in parallel and nevertheless the inversion signalling bits will be error protected. The parallelism between the ECC and the low power encoding make it possible to avoid any latency adder. Also the modulo 2 sum between c and c' can be achieved by substituting already present buffers, which act as pre-drivers for the off-chip transmitters, with xor gates. The encoding process is represented graphically in Fig. 2.

3.1. Application to Main Memory

Substantial modifications of mainstream DRAM components can be avoided by reusing the already present redundancy for error protection and detection. In main memories for servers, ECC bits are added. For example, in a 64-bit interface, 8 additional ECC bits are sufficient to implement an extended Hamming code capable of correcting single errors and detecting double errors. By encoding larger blocks of data, the bits freed can be reused for bus inversion signalling. Encoding and decoding for bus inversion can be performed at the memory controller side, making the use of industry standard DRAMs possible.

Communication to main memory usually occurs in bursts of 4 or 8 memory words of 72 bits each, 64 bits of data and 8 bits for ECC. In order to combine data bus inversion with error protection, without additional pins for bus inversion signalling, the burst structure must be modified. Memory words are logically grouped in pairs and the ECC word is computed over 128 data bits. The ECC bits also protect the inversion signalling bits as described in Section 3. In order to have single error correction, a Hamming(255,247) code is used. Therefore, 8 redundancy bits are sufficient to correct single errors on 247-bit long messages. Same error detection performance as the standard system is achieved with a parity bit for each of the 72 bits. With the proposed data structure, each 72-bit memory word carries 64 data bits, 4 bits of ECC (half of the 8-bit ECC word), 3 bits for bus-invert signalling, and 1 parity bit.

In order to compute the ECC word, the original 128-bit message $x = (x_1, x_2)$, where x_1, x_2 are the two 64-bit words to be transmitted, is first augmented with zeros in order to reach the correct size (i.e., 247 bits). Let G be the generator matrix of the Hamming(255,247), and let us consider G in the canonical form.

$$x = (x_1, x_2, \vec{0}) \cdot G = (x_1, x_2, \vec{0}, c), \tag{3}$$

with c being the 8 check bits.

The bus inversion is performed on the vectors (x_1, x_2) in three chunks of respectively 22, 22 and 20 bits, as 64 is not a multiple of 3 (the rationale behind this partition is given in Section 4). To each inversion signalling bit is associated the vector obtained by calculating the check-bits of the vector having ones in the positions that have to be inverted, a one in the relative signalling position, and zeros elsewhere. For instance, if the first inversion signalling bit is related to the first 22 bits of x_1, then the relative check bits g_1 can be obtained as:

$$[\vec{1}_{(22)}, \vec{0}_{(42)}, 1, \vec{0}_{(182)}] \cdot G = [\vec{1}_{(22)}, \vec{0}_{(42)}, 1, \vec{0}_{(182)}, g_1], \tag{4}$$

where $\vec{1}_{(k)}$ is a k-bit long all-ones vector and $\vec{0}_{(k)}$ is a k-bit long all-zeros vector. This value has to be added modulo 2 (xor-ed) to the word $(x_1, x_2, \vec{0}_{(3)}, c)$ in order to obtain the actual message including bus inversion and single-error recovery. Double error detection is achieved by means of extra parity bits. This can be done on each column of the packet comprising 64 bits of data x_1, 4 bits of the check bits (the actual 8-bit checksum is split among two columns), and the 3 inversion signalling bits

$$p_1 = \sum(x_1, c_1) \oplus \sum_{i=1}^{i=3} b_{1,i} \sum g_{1,i}, \tag{5}$$

where c_1 is the vector comprising half of the check-bits c, $g_{1,i}$ is the vector comprising half of the check-bits g_i, and $b_{i,j} \in \{0, 1\}$ are the actual values of the bus inversion signalling bits. All sums are intended modulo 2. Again, note that the values $\sum g_{j,i}$ can be pre-calculated and the adjustment can be implemented with xor gates, as the effect will be just inverting or not the parity value $\sum(x_1, c_1)$.

4. Power Consumption

Description	Value
I/O Power supply	1.8 Volt
Transmitter impedance for control bus	16 Ω
Termination impedance for control bus	20 Ω
Transmitter impedance for DQ, DQS	40 Ω
Termination impedance for DQ, DQS	60 Ω

Table 1. Memory system electrical parameters.

No. of groups	Partition	Activity	Savings
1	64	29.27	8.53%
2	2 × 32	28.38	11.31%
3	21,21,22	27.87	12.92%
3	**22,22,20**	**27.78**	**13.18%**
4	4 × 16	27.32	14.62%

Table 2. Average number of events per bus cycle.

The system taken into account is a 4GB plus ECC memory system, using 512Mbit 70nm DDR2 (Double Data Rate 2) at $667 MTs$ I/O speed and 5-5-5 latency DRAM components [11]. The system has 4 ranks and uses by-4 components resulting in 72 DRAM dice organized on 2 DIMMs. The power of the memory system (excluding I/Os) is calculated on the basis of DRAM typical current consumption measurement results taken during mass production, according to JEDEC standard [11]. Detailed explanation on how to calculate memory system power consumption can be found in technical notes from DRAM vendors [18]. The workload assumes a 70% data bus occupation with a write to read ratio of 1:2 with closed page policy (typical in servers). In order to achieve such a high utilization, a burst length of 8 is used.

Electrical parameters of the memory system are reported in Table 1. Activity on the data bus with bus-invert technique was calculated using the model presented by Lin and Tsai [14]. In Table 2 the effect of different partitions of the data bus is shown. The partition chosen for our design is highlighted in bold. In Table 3 a breakdown of the power consumption of a standard and of the proposed system is shown. As can be expected, the power savings on

Contributor	Pins	Power [mW]		Savings
		Standard	Proposed	
Cmd/Addr/Ctlr	39	567	567	-
DQ/DQS	**108**	**1633**	**959**	**41%**
Total I/O	147	2200	1526	31%
Total DIMM	147	10800	10126	6%

Table 3. Memory system power breakdown.

the data bus (DQ/DQS) due to the combination of the bus-invert technique with a V_{dd} termination scheme are quite high (41%). The power saving is diluted to 6%, when the total power consumption of the memory subsystem is taken into account. This has to be regarded as a lower bound on power savings as higher memory densities (less memory ranks, or by-8 organization) will result in a lower total power, while the power consumption of the I/O interface will stay constant. Moreover, power savings on the total memory subsystem will be higher if the same amount of memory is, for example, organised in 2 memory channels, each having half the number of ranks. Such configurations improve performance, as doubling the number of memory channels increases the communication bandwith at the price of higher I/O power consumption—hence, the importance of this work.

5. System Reliability

In this section, system reliability for different systems will be presented. First a mathematical model is developed, then parameters for the model are selected, and, finally, results of numerical simulations are shown.

5.1. Statistical Model

Noorlag et al. already proposed a statistical model that makes it possible to calculate the time to failure due to both hard and soft errors of a memory system with single error correction [20]. We will now extend this model to take into account also communication errors. The system is described as a discrete-time stochastic process. Let us assume that all the failure mechanisms are Poissonian processes [20, 22]. Let λ_H, λ_S, and λ_b be the rates of occurrence of respectively hard, soft, and communication fails. Let T_0 be the time period with which the word is purged from soft errors, i.e., the word is read, any single error is corrected, then is written back. A bit can experience a communication error during a write operation. For the sake of simplicity, communication errors during read operations are ignored since the word can be retransmitted. Let T_b be the bit-unit interval. Since all N_b bits are transmitted in parallel, the write operation lasts T_b. The probability of a single bit being affected by one or more communication errors is therefore

$$P(\textit{wrong transmission}) = 1 - e^{-\lambda_b T_b}. \qquad (6)$$

At any time, each memory word can be in one of the following states $\{S, H, \mathit{0}, \mathit{0}', W\}$. The word is in state S if exactly one bit is wrong due to a soft error and there are no hard fails. The word is in state H if exactly one hard failure happened. The states $\mathit{0}$ and $\mathit{0}'$ are the states where respectively N_b or $N_b - 1$ bits are correct. W is the state where the word has one bit wrong due to a communication error happened during a write operation. The word will change state with the following probabilities:

$$P_0 = e^{-N_b(\lambda_H T_0 + \lambda_S T_0 + \lambda_b T_b)}, \qquad (7)$$

$$P_{0'} = e^{-(N_b-1)(\lambda_H T_0 + \lambda_S T_0 + \lambda_b T_b)}, \qquad (8)$$

$$P_S = P_0 N_b (e^{\lambda_S T_0} - 1), \qquad (9)$$

$$P_H = N_b(1 - e^{\lambda_H T_0}) P_{0'}, \qquad (10)$$

$$P_W = N_b(1 - e^{\lambda_b T_b}) P_{0'}. \qquad (11)$$

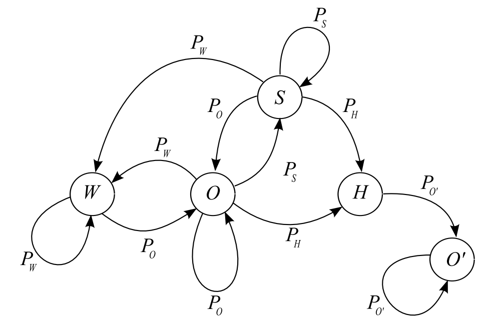

Figure 3. Stochastical model. S, H and W are the states with one bit wrong due to respectively soft, hard, or communication errors. $\mathit{0}$ and $\mathit{0}'$ are the states with N_b or $N_b - 1$ correct bits.

Fig. 3 shows the finite state machine with relative transition probabilities. Let A be the transition matrix:

$$A = \begin{bmatrix} P_s & P_H & P_0 & 0 & P_W \\ 0 & 0 & 0 & P_{0'} & 0 \\ P_s & P_H & P_0 & 0 & P_W \\ 0 & 0 & 0 & P_{0'} & 0 \\ 0 & 0 & P_0 & 0 & P_W \end{bmatrix}.$$

Initially the memory word is free of errors and in state $\mathit{0}$; therefore, the transition matrix A_0, which describes the first transition, is the matrix that has the third row equal to the third row of A and all other rows null. Assuming independent words, the probability that a system with N_W memory words remains operational after nT_0 time intervals is

$$P(\textit{system good at}\ t = nT_0) = \left(\sum_{i,j} r[n]_{i,j} \right)^{N_W} \qquad (12)$$

where $r[n]_{i,j}$ are the elements of the matrix

$$R[n] = A_0 A^{n-1}. \qquad (13)$$

Obvious modifications to Equations 7-11 have to be made in case of unprotected bus-invert signalling bits.

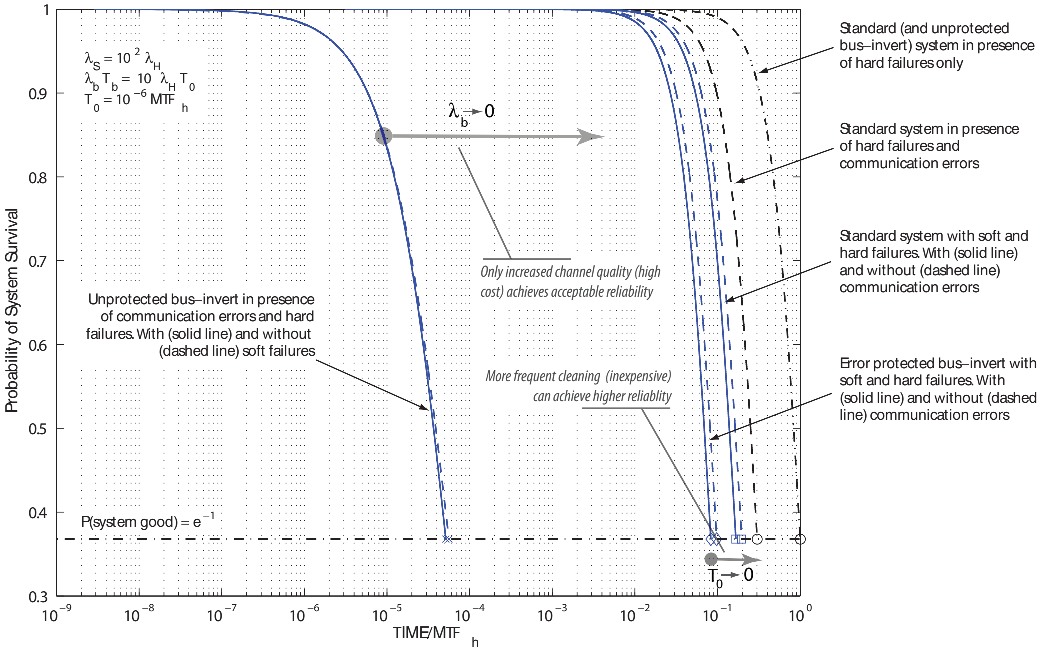

Figure 4. Survival probability for different memory systems. Unprotected bus inversion leads to unacceptable low system reliability. Error protected bus inversion makes possible to reuse redundant bits for bus inversion signalling without loss of system reliability. The small reliability loss due to lower redundancy can be easily regained with more frequent purge of errors.

5.2. Model Parameters

The memory system taken into account has 10^8 words of 72 bits for a total capacity of 800 MB excluding redundancy. The memory spaces currently used in high end servers are much larger, as there are on the market memory modules in the range of gigabytes. However a larger memory space will exacerbate the problem and the size chosen for the experiments is large enough to expose the issues of not protecting the signalling inversion bits from communication errors.

The actual values for soft and hard failure rates for a given technology are difficult to estimate or measure and there is a high variance in reported results [2, 5, 26, 16, 17]. For low failure rates, the relationship between the mean time to failure in the presence of soft error rates and the mean time to failure due to only hard failures is determined by the ratio between soft and hard error rates, $\frac{\lambda_S}{\lambda_H}$. Although infrequent, soft errors have rates which are orders of magnitude greater than the rates of hard failures (mean times to single event due to soft errors are in the order of weeks, whereas mean times to single event due to hard errors are in the order of years). Nevertheless, the probability of failure due to soft errors can be made arbitrarily small by reducing the erasure interval T_0 sufficiently [20], provided that the row cycle time allows to read-modify-write all the memory words in the system within the erasure interval. The mean time to failure (MTF) is defined as the probability of survival of $e^{-1} \approx 36.8\%$. The target of a system designer is to select T_0 such that the mean time to failure due to hard and soft errors is comparable (i.e., not smaller than one order of magnitude) to the mean time to failure due to hard failures only (MTF_h). The clearing of soft errors through the normal data flow is usually adequate.

As opposed to soft errors due to alpha particles or cosmic rays, communication errors are usually much more frequent. In high speed interfaces, bit error rate (BER) values might range from 10^{-12} for noisy interfaces to less than 10^{-16} for highly reliable communication channels. As a reference, at a rate of 1 Gbps, a BER of 10^{-14} would result in 1 error per day on one bit line. The problem becomes worse in wide parallel interfaces, common in memory systems, forcing the design of highly reliable channels. Although small, the probability of communication errors cannot be assumed zero and, as it will appear, making this assumption during system design might lead to poor reliability. In our simulations a BER such that $\lambda_b T_b = 10 \cdot \lambda_H T_0$ was assumed.

5.3. Numerical Results

In Fig. 4 an overview of the reliability of the different systems taken into consideration is given. All measurements have been normalised over MTF_h, as the probability of survival in the presence of hard failures only (dash-dotted curve terminating with a circle) is an upper bound for systems with additional failure mechanisms. In the presence of communication errors the upper bound is lower and is represented by the survival probability curve of the standard system in the presence of both hard failures and communication errors (rightmost dashed curve terminating with a circle). The reliability performance of the standard system when also soft errors are taken into account is represented by the survival probability curves ending with squares. More frequent clearing of soft errors will increase the survival probability also in presence of communication errors, reaching the relative upper bound.

As one expects, communication errors in systems with unprotected bus inversion significantly degrade the reliability of the system. As shown in Fig. 4, with all parameters being equal, the mean time to failure (curves ending with the x-mark) decreases by several orders of magnitude. In the presence of communication errors, it is not possible to improve the system reliability by lowering the erasure period T_0, as the upper bound is represented by the same system in the absence of soft failures (dotted curve ending with the x-mark). For this system, although infrequent, communication errors are the main failure mechanism and the reliability can only be improved by increased quality of the communication media, which might not be feasible, or alternatively, by lowering the data rate on the channel, which results in decreased performance of the memory system.

Conversely, the survival probabilities of a system with the proposed error-protected bus inversion (curves ending with diamonds) are comparable to those of the standard system. Additionally, the upper bound limits (with and without communication errors) that can be reached by lowering T_0 are the same. Therefore, the proposed scheme has a very limited cost, saves tangible energy, and has practically no impact on the correct operation of the system.

6. Conclusions

We presented a novel combination of encoding for low power and error protection, which makes it possible to use the bus-invert technique in the presence of errors without any compromise on overall system reliability. Moreover, as opposed to current schemes of bus inversion in commercial applications, the proposed architecture allows the use of industry standard memories avoiding the system cost increase due to specialized DRAMs.

References

[1] K. Basu, A. Choudhary, J. Pisharath, and M. Kandemir. Power protocol: reducing power dissipation on off-chip data buses. In *MICRO 35: Proceedings of the 35th Annual ACM/IEEE International Symposium on Microarchitecture*, pages 345–355, 2002.

[2] R. Baumann. Soft error characterization and modeling methodologies at TI: Past, present, and future. In *4th Annual Topical Conference on Reliability*, 2000.

[3] R. Baumann. Soft errors in advanced computer systems. *IEEE Des. Test*, 22(3):258–266, 2005.

[4] D. Bertozzi, L. Benini, and G. de Micheli. Low power error resilient encoding for on-chip data buses. In *DATE '02: Proceedings of the Conference on Design, Automation and Test in Europe*, page 102, 2002.

[5] A. Cataldo. IBM moves to protect DRAM from cosmic invaders. *EETimes*, Jun 1998.

[6] N. Chang, K. Kim, and J. Cho. Bus encoding for low-power high-performance memory systems. In *DAC '00: Proceedings of the 37th Conference on Design Automation*, pages 800–805, 2000.

[7] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *Proceedings of the 34th Annual International Symposium on Computer Architecture*, San Diego, CA, June 2007.

[8] R. J. Fletcher. Integrated circuit having outputs configured for reduced state changes. *United States Patent 4667337*, 1987.

[9] T. Givargis, F. Vahid, and J. Henkel. Evaluating power consumption of parameterized cache and bus architectures in system-on-a-chip designs. *IEEE Transactions on Very Large Scale Integration Systems*, Aug 2001.

[10] J.-D. Ihm et al. An 80nm 4Gb/s/pin 32b 512Mbit GDDR4 Graphics DRAM with Low-Power and Low-Noise Data-Bus Inversion. In *Solid-State Circuits Conference, ISSCC 2007. Digest of Technical Papers*, pages 492–493,617, Feb 2007.

[11] DDR2 SDRAM specifcation. JEDEC standard, JEDEC, www.jedec.org, May 2006.

[12] C. Lefurgy, K. Rajamani, F. Rawson, W. Felter, M. Kistler, and T. W. Keller. Energy management for commercial servers. *Computer*, 36(12):39–48, 2003.

[13] C.-H. Lin, C.-L. Yang, and K.-J. King. Hierarchical value cache encoding for off-chip data bus. In *ISLPED '06: Proceedings of the 2006 International Symposium on Low Power Electronics and Design*, pages 143–146, 2006.

[14] R.-B. Lin and C.-M. Tsai. Theoretical analysis of bus-invert coding. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 10(6):929–934, Dec 2002.

[15] W. McKee et al. Cosmic ray neutron induced upsets as a major contributor to the soft error rate of current and future generation DRAMs. In *Reliability Physics Symposium, 1996. 34th Annual Proceedings*, pages 1–6, 1996.

[16] DRAM Module Mean Time Between Failures (MTBF). Technical note, Micron Technology, www.micron.com, 1997.

[17] DRAM Soft Error Rate Calculations. Technical note, Micron Technology, www.micron.com, 1997.

[18] Calculating memory system power for DDR2. Technical note, Micron Technology, www.micron.com, 2004.

[19] D. A. Mulla and D. L. Tu. Fast approximate DINV calculation in parallel with coupled ECC generation or correction. *United States Patent Application 20050289435*, 2005.

[20] D. Noorlag, L. Terman, and A. Konheim. The effect of alpha-particle-induced soft errors on memory systems with error correction. *IEEE Journal of Solid-State Circuits*, 15(3):319–325, 1980.

[21] S. R. Sridhara and N. R. Shanbhag. Coding for system-on-chip networks: a unified framework. In *DAC '04: Proceedings of the 41st Annual Conference on Design Automation*, pages 103–106, 2004.

[22] G. R. Srinvasan. Modeling the cosmic-ray-induced soft-error rate in integrated circuits: An overview. *IBM J. Res. Dev.*, 40(1):77–89, 1996.

[23] M. R. Stan and W. P. Burleson. Bus-invert coding for low-power I/O. *IEEE Trans. Very Large Scale Integr. Syst.*, 3(1):49–58, 1995.

[24] M. R. Stan and W. P. Burleson. Low-power encodings for global communication in CMOS VLSI. *IEEE Trans. Very Large Scale Integr. Syst.*, 5(4):444–455, 1997.

[25] D. C. Suresh, B. Agrawal, J. Yang, W. Najjar, and L. Bhuyan. Power efficient encoding techniques for off-chip data buses. In *CASES '03: Proceedings of the 2003 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, pages 267–275, 2003.

[26] J. Ziegler, M. Nelson, J. Shell, R. Peterson, C. Gelderloos, H. Muhlfeld, and C. Montrose. Cosmic ray soft error rates of 16-Mb DRAM memory chips. *IEEE Journal of Solid-State Circuits*, 33(2):246–252, 1998.

[27] J. F. Ziegler and W. A. Lanford. Effect of cosmic rays on computer memories. *Science, vol. 206*, page 776, 1979.