A STATISTICAL ANALYSIS, FOR REDUCING THE ENERGY DISSIPATION IN A BUS-SWITCH ENCODER.

Mauro Olivieri
Department of Electronic Engineering
University of La Sapienza
Rome, Italy
Email: Olivieri@die.uniroma1.it

Francesco Pappalardo and Giuseppe Visalli
Advanced System Technology
ST Microelectronics Catania, Italy
Email: Francesco.Pappalardo@st.com
Email: Giuseppe-ast.Visalli@st.com

ABSTRACT
The Bus Switch mechanism is a recently proposed bus encoding technique for low-power off-chip data buses. The approach is based on clustering, reordering and encoding the bus input lines according to a reordering pattern and a fixed coding function. This work presents a statistical approach for reducing the hardware overhead of the bus switch technique, by operating with a sub-set of the possible reordering patterns. We demonstrate the effectiveness and robustness of the proposed approach by ANSI C simulations, measuring the average switching activity savings. Our results show a modest switching activity degradation while saving 90% computation time, thus obtaining a sub-optimal encoder configuration satisfactory for a large variety of benchmarks.

1 Introduction and Background

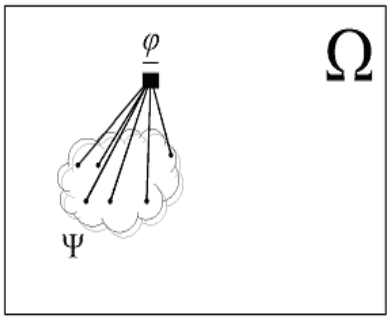
The problem of bus-encoding for low-power is mostly concerned to the switching power reduction in both address [2] [3] and data buses [1] [7], due the correlation between two consecutive words. Address buses, in particular, imply high correlation due the instruction locality during program execution. More in detail, the program address space could show different working segments, permitting memory addressing with reduced bus lines [5]. The general approach for address and data bus considers the data correlation known in advance. This property does not held in many applications, where the bus transmits a large amount of information with varying statistical properties. The Bus Switch (BS) mechanism [6] represents a recently proposed adaptive approach for low-energy data bus encoding. This approach divides a large data bus in identical clusters, applying a reordering scheme and a fixed coding function to the input lines of the bus. This combined action permits activity saving greater than previous approaches such as bus invert (BI) [9] and adaptive partial bus invert (APBI) [8]. Unfortunately, the hardware complexity reduces the BS' field of application to off-chip buses with line capacitance more than 6pF and 4pF in technology at 180nm and 130nm respectively (32-bit bus, 4-bit cluster size [6]). In particular, the main limitation for power reduction and circuit feasibility concerns the utilization of the full-set of possible reordering patterns, which factorially depends on cluster size. A practical BS realization considers a small cluster size in conjunction with a reduced pattern set. Such sub-optimal BS architecture has different energy consumption, as illustrated in the right part of Figure 1 in a 130nm CMOS library. Sizing the sub-optimal BS architecture, with minimal loss of performance, represents the actual challenge for enhancing the BS feasibility, by decreasing the minimal bus capacitance for the cost-effective BS utilization. Presently, an exact statistical profiling of the BS performance requires a huge computation load, because of the simulation of complex benchmarks. This work introduces a general purpose statistical approach for BS architecture sizing, based reducing the total reordering patterns by an effective and low cost off-line analysis. Being: Ω as the space of all possible benchmarks (fig. 1 Left), the proposed approach considers a very reduced subset of benchmarks φ which is statistically representative of an initial (big) set $\Psi \subseteq \Omega$. The off-line analysis operated with random test vectors from a Monte Carlo engine [4], measuring the most used reordered patterns. This approach dramatically reduced the computational time, addressing the design issues toward the minimum hardware complexity for the encoder. This result enhances the feasibility and usefulness of the BS approach. The paper is organized as follows: section 2 shortly introduces the BS mechanism and the design-performances issues. Section 3 defines the statistical approach to the problem, including the necessary background for a consistent theory. Section 4 illustrates the simulation results, showing the loss in performance (i.e. less switching activity saving) by means of the proposed approach. Section 5 is dedicated to the conclusion.

2 The Bus Switch Mechanism

In principle, the Bus Switch technique can be logically expressed as a four-step process:

1. A large bus is divided into several identical clusters of M (cluster depth) lines each.

2. Each M-line bus is coded by swapping the input lines using a particular *reordering pattern*



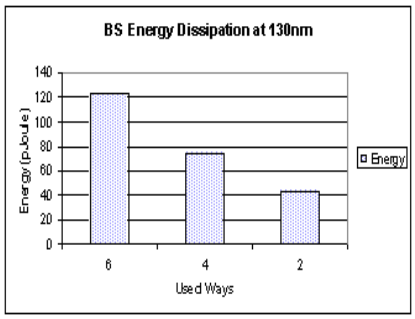


Figure 1. Left: The set of possible benchmarks. Right: BS' energy estimation at different ways (130nm technology)

3. A tentative data encoding is obtained by applying to the swapped M lines a fixed *coding function*.
4. The process is repeated M! times from step 2 until the optimal reordering pattern is found, that minimizes the output switching activity in the encoded data of the whole bus.

In [6], the authors found the minimal bus capacitance for BS convenient utilization be:

$$C_{min} = \frac{2}{\alpha \cdot V_{dd}^2} \cdot E_{Encoder} \cdot \frac{1}{N + EL} \qquad (1)$$

The N-line bus operates at frequency 1/T with a signal level Vdd. EL represents the BS' extra lines, α the average switching activity reduction and $E_{Encoder}$ the estimated BS energy consumption. Experimental results in [6] indicated that the activity saving does not significantly improves at cluster depth grater than six. Additionally, a reduced and optimized reordering pattern set does not significantly improve the activity saving [6]. The proposed approach improves C_{min}, reducing $E_{Encoder}$.

3 The proposed BS statistical optimization

The effectiveness of a BS reordering pattern depends on the preceding data values and *reordering patterns*. For this reason, the optimal reordering patterns reduction is hard to accomplish. Experimental results in [6] indicated the pattern set reduction, based on measurement of the most used reordering schemes, sometimes even improves the switching activity savings. More often, it causes a slight saving degradation. This section introduces the proposed approach for selecting the best reordering patterns for minimal performance degradation, valid for a large variety of benchmarks.

3.1 Problem Formulation

Let: Ω be the space of possible benchmarks of file transfer over the bus, organized by file type (Bin, PDF, AVI, etc.). Table 1 illustrates the BS activity savings (SA) statistics,

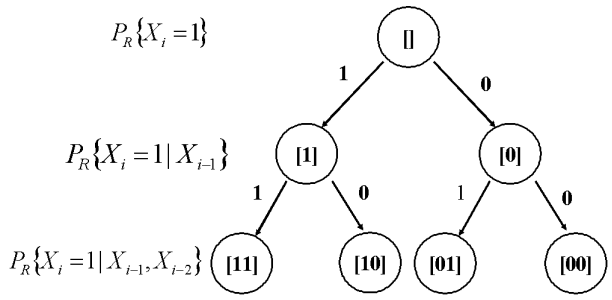


Figure 2. N-order model using binary tree

for the whole set of benchmarks (32-bit bus). Each category introduces a different dispersion index, which justifies the choice of typology-based test benches collection.

$$\rho = \frac{E\{SA\}}{\sigma_{SA}^2} \qquad (2)$$

We consider a much reduced subset of benchmarks φ which is statistically representative of an initial big set $\Psi \subseteq \Omega$. In this work, Ψ represents 60 different benchmarks divided in six categories. We measured the most used reordering patterns by using a test bench vector, which has the statistical properties of φ by means of a Monte Carlo engine.

3.2 Statistical Model

Let $X_i(t)$ the generic i-th input bus line. It is a random process at two distinct values (0,1), non necessarily strict

	BIN	Jpeg	AVI	ZIP	PDF	MP3
$E\{SA\}$	17.94	17.37	17.22	17.01	12.89	18.06
σ_{SA}^2	2.75	0.24	0.43	0.53	4.23	0.30
ρ	6.5	69.78	39.66	31.93	3.04	59.39

Table 1. Activity saving' statistics

sense stationary (SSS). Since a benchmark represents a particular record of bus process, the static probabilities P0 and P1 have been measured by the relative frequency. If we dispose of M different samples then

$$P_0 = P_R\{X_i(t) = 0\} \approx \frac{1}{M} \cdot \sum_{t}^{M} \big(X_i(t) == 0)?1 : 0 \tag{3}$$

$$P_1 = P_R\{X_i(t) = 1\} \approx \frac{1}{M} \cdot \sum_{t}^{M} \big(X_i(t) == 1)?1 : 0 \tag{4}$$

Additionally, we can define the N-order statistic by means of the following probability:

$$P_R\{X_i(t) = 1 | X_i(t-1), X_i(t-2), \ldots X_i(t-N)\} \tag{5}$$

This probability represents a sufficient statistics for the complete acknowledgment of the process' behavior. The probability can be measured in a similar fashion of static probability, creating an N-depth binary tree (fig. 2).

3.3 Proposed Approach

The proposed approach considers the N-order model of the considered benchmarks set φ providing its statistical distribution. The approach can be expressed as 5 step process:

1. Let R be the wished number of reduced pattern set ($R < M!$)
2. A Monte Carlo engine generates a random input stream accordingly to φ 's N-order model.
3. The full BS model (M!) encoded the stream, measuring the frequency of the used reordering patterns.
4. The full pattern set has been ordered by frequency of use.
5. The process is repeated from step 2 until we have R *stable* most used patterns (*stopping criterion*).

The Monte Carlo simulator permits repeated (stochastic) simulations using a randomly sampled region of model space. MC-SIM operates with continue-time variables and systems. This tool models a system defining continue states and outputs and assigning the states' dynamics. The tool does not support directly discrete-time simulations. However, the discrete-time analysis can be pursued including the explicit time dynamics (dt(t)=1), temporarily variables (e.g. $state\_tmp$) used in the discrete-time state assignments:

State = (t == time)? New_value: $state\_tmp$

The model parameters can change during Monte Carlo Simulations, with their exact values.

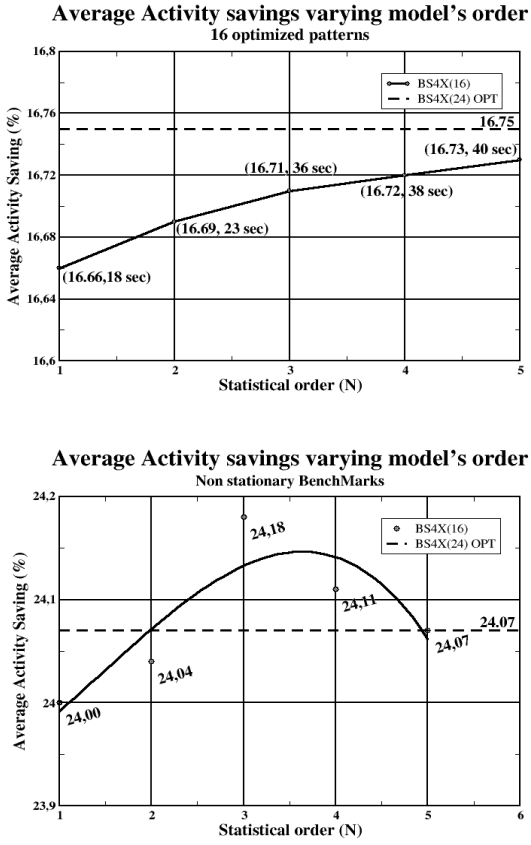


Figure 3. Up: Statistical Optimizations varying model's order. Down: Statistical Optimizations of non-stationary benchmarks (including polynomial regression).

4 Simulation Results

The effectiveness of the proposed approach has been demonstrated providing the ANSI C models of Bus Switch encoder/decoder system. The process of pattern selection includes an automated flow, by means of PERL scripts, including also a free and public Monte Carlo engine: the MC-SIM simulator. Our analysis considers a 4-bit clusters with a 32-bit bus, measuring the activity savings degradation, performing the pattern set reduction to 16 and 8 respectively. The ANSI C programs for statistical profiling, employees an N-depth binary tree, measuring the occurrence of logic 1 under the assigned state (see eq. 5). Starting from an initial random bench mark set φ (which includes one benchmark for each typology) , Table 3 and 4 illustrate the performance of the proposed approach, which reduces the average activity saving of only 0.06 % and 0.15% in 24-to-16 and 24-to-8 reduction respectively. Additionally, our results indicate a modest performance enhancement when increasing the traffic N-order statistic (fig. 3 top). This result indicates the benchmark represents a quasi- stationary process. The robustness of the proposed approach has been demonstrated starting from one thousand of random

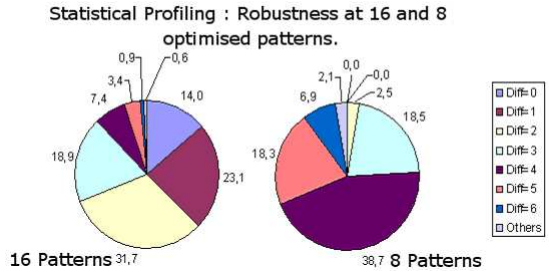


Figure 4. Robustness of the proposed optimizations

initial benchmark set, comparing the result to the reference patterns set employed in Table 3 and 4. Results indicated modest changes compared to the reference set, operating with a light reordering scheme reduction (fig. 4). Additionally, we simulated the proposed optimization strategy including non-stationary benchmarks. Explicit time-dependence represented the simplest way to generate non-stationary process. We considered the logic bus line Bi as:

$$B_i = A_i\, AND\, (X_i\, XOR\, (G_i\, OR\, N_i));$$

The logic variables A_i, X_i, and G_i explicitly depend on simulation time; N_i represents a uniformly distributed logic value. In this case, our optimization strategy permitted a better average activity saving than the complete profiling, increasing the statistical order (fig 3 Bottom). Figure 5 showed the minimal bus capacitance for cost-effective BS implementation, according to Eq. 1 for a 130nm CMOS implementation. Finally, table 2 illustrates the saving in computational time, using the proposed approach, against a complete profiling.

5 Conclusion

This work introduced the basic principle for reducing the set of reordering schemes, based on the frequency of use, in a Bus Switch encoder architecture. Such sub-optimal approach strongly reduces the computation time needed to validate the BS efficiency of coding in a large variety of benchmarks. The approach includes a probabilistic profiling, by means of the basic momentum measurement, the statistical simulation, by means of a Monte Carlo engine, and finally the reordering schemes classification. Experimental result indicate an average loss of performance below 0.06% while saving 90% of the computation time used for statistical profiling. We demonstrated how this approach is consistent and robust for a large variety of benchmarks. This approach implies the best performance efforts in a non-stationary traffic, where an accurate statistical model has been proposed. The results enhance the possibility to utilize the Bus Switch mechanism in off-chip wide data buses.

Model's order (N)	Time Saved (%)
1	94.80%
2	93.40%
3	89.71%
4	89.14%
5	88.57%

Table 2. Savings in computational time, using the proposed optimization strategy

References

[1] L. Benini, A. Macii, E. Macii, M. Poncino, and R. Scarsi. Architectures and synthesis algorithms for power-efficient bus interfaces. *IEEE Transaction on Computer-Aided Design*, 19:969–980, Sept. 2000.

[2] L. Benini, G. D. Micheli, E. Macii, D. Sciuto, and C. Silvano. Asymptotic zero-transition activity encoding for address busses in low power microprocessor-based systems. In *ACM/IEEE Great Lake Symposium on VLSI*, pages 77–82, Urbana, IL, Mar. 1997.

[3] L. Benini, G. D. Micheli, E. Macii, D. Sciuto, and C. Silvano. Address bus encoding techniques for system-level power optimization. In *IEEE Design Automation and Test in Europe*, pages 861–866, Paris, France, Feb. 1998.

[4] F. Y. Bois and D. Maszle. MCSim monte carlo simulator program in ftp://sparky.berkeley.edu/pub/mcsim.

[5] E. Musoll, T. Lang, and J. Cortadella. Working-zone encoding for reducing the energy in microprocessor address buses. *IEEE Transaction on VLSI Systems*, 6:568–572, Dec. 1998.

[6] M. Olivieri, F. Pappalardo, and G. Visalli. Bus-switch coding for reducing power dissipation in off-chip buses. *IEEE Transaction on VLSI Systems*, 12, Dec. 2004.

[7] S. Ramprasad, N. Shanbhag, and I. Hajj. Signal coding for low power: Fundamental limits and practical realizations. *IEEE Transaction on Circuit and Systems II: Analog and Digital Signal Processing*, 46:923–929, 1999.

[8] R. Siegmund, C. Kretzchmar, and D. Muller. Adaptive bus encoding technique for switching activity reduced data transfer over wide system buses. In *Proc. of International Workshop on Power And Timing Modeling Optimization and Simulation*, Gottingen, Germany, 2000.

[9] M. Stan and W. Burleson. Bus-invert coding for low power I/O. *IEEE Transaction on VLSI Systems*, 3:49–58, Mar. 1995.

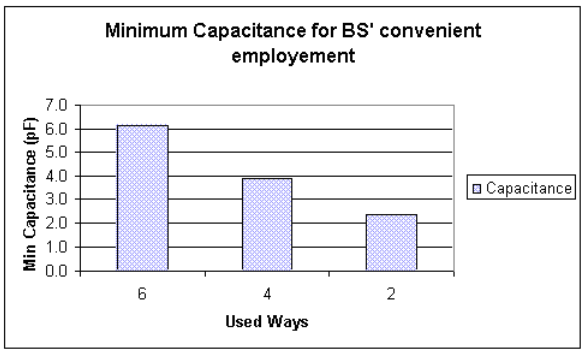


Figure 5. Minimum bus capacitance for convenient BS at 130nm (Vdd=2.5V)

Bench mark	Size (Bytes)	BS4X(24) Activity Saving	BS4X(16) Activity Saving	Bench mark	Size (Bytes)	BS4X(24) Activity Saving	BS4X(16) Activity Saving
bin0	300544	17.04%	16.77%	Zip0	871278	15.59%	15.57%
bin1	471232	16.89%	16.77%	Zip1	396378	17.39%	17.26%
bin2	323320	18.32%	18.12%	Zip2	355474	17.26%	17.21%
bin3	66008	18.17%	17.86%	Zip3	209082	17.25%	17.11%
bin4	1763708	20.51%	20.18%	Zip4	356454	17.03%	16.95%
bin5	321900	18.72%	18.19%	Zip5	8581936	16.99%	16.95%
bin6	1170872	17.47%	17.21%	Zip6	220938	17.63%	17.57%
bin7	7040	23.03%	23.56%	Zip7	170007	17.30%	17.21%
bin8	282680	17.73%	17.14%	Zip8	2111474	16.78%	16.65%
bin9	29432	11.58%	12.00%	Zip9	584184	16.90%	16.83%
jpeg0	1207581	16.83%	16.71%	Pdf0	1122283	11.56%	11.94%
jpeg1	1112023	17.37%	17.22%	Pdf1	366296	16.75%	16.77%
jpeg2	1132849	17.31%	17.20%	Pdf2	758082	12.91%	13.06%
jpeg3	1184111	17.55%	17.43%	Pdf3	748726	12.36%	12.48%
jpeg4	1109252	17.79%	17.69%	Pdf4	927832	12.62%	12.67%
jpeg5	1335984	17.14%	17.09%	Pdf5	322862	9.25%	9.73%
jpeg6	1252012	17.37%	17.28%	Pdf6	373698	15.53%	15.64%
jpeg7	1227500	17.33%	17.26%	Pdf7	1048576	12.21%	12.27%
jpeg8	1161942	17.60%	17.47%	Pdf8	485396	11.32%	11.45%
jpeg9	1236074	17.45%	17.36%	Pdf9	1124000	14.40%	14.51%
Avi0	936358	16.60%	16.54%	MP3-0	3892272	17.24%	17.14%
Avi1	1624708	16.55%	16.52%	MP3-1	5376068	18.07%	18.00%
Avi2	1569058	16.76%	16.67%	MP3-2	3423342	18.27%	18.21%
Avi3	942526	17.71%	17.59%	MP3-3	4139135	18.02%	17.94%
Avi4	1026186	17.38%	17.32%	MP3-4	4331796	18.44%	18.32%
Avi5	909982	17.83%	17.70%	MP3-5	3921572	18.24%	18.14%
Avi6	1212000	17.13%	17.08%	MP3-6	4075520	18.23%	18.13%
Avi7	1579568	17.19%	17.15%	MP3-7	8236801	18.01%	17.94%
Avi8	1236472	17.54%	17.42%	MP3-8	5728433	18.06%	17.97%
Avi9	1153354	17.51%	17.44%	MP3-9	6957257	18.01%	17.93%

Table 3. Activity Savings, after $24- > 16$ reordering patterns reduction (Average 16.69%).

Benchmark	Size (Bytes)	BS4X(24) Activity Saving	BS4X(8) Activity Saving	Benchmark	Size (Bytes)	BS4X(24) Activity Saving	BS4X(8) Activity Saving
bin0	300544	17.04%	16.87%	Zip0	871278	15.59%	15.47%
bin1	471232	16.89%	16.71%	Zip1	396378	17.39%	17.17%
bin2	323320	18.32%	18.23%	Zip2	355474	17.26%	17.07%
bin3	66008	18.17%	18.21%	Zip3	209082	17.25%	17.07%
bin4	1763708	20.51%	20.90%	Zip4	356454	17.03%	16.91%
bin5	321900	18.72%	18.88%	Zip5	8581936	16.99%	16.88%
bin6	1170872	17.47%	17.72%	Zip6	220938	17.63%	17.41%
bin7	7040	23.03%	23.70%	Zip7	170007	17.30%	17.07%
bin8	282680	17.73%	18.04%	Zip8	2111474	16.78%	16.62%
bin9	29432	11.58%	12.38%	Zip9	584184	16.90%	16.65%
jpeg0	1207581	16.83%	16.69%	Pdf0	1122283	11.56%	10.85%
jpeg1	1112023	17.37%	17.19%	Pdf1	366296	16.75%	16.27%
jpeg2	1132849	17.31%	17.17%	Pdf2	758082	12.91%	12.36%
jpeg3	1184111	17.55%	17.43%	Pdf3	748726	12.36%	12.30%
jpeg4	1109252	17.79%	17.70%	Pdf4	927832	12.62%	12.51%
jpeg5	1335984	17.14%	17.02%	Pdf5	322862	9.25%	7.77%
jpeg6	1252012	17.37%	17.22%	Pdf6	373698	15.53%	15.14%
jpeg7	1227500	17.33%	17.19%	Pdf7	1048576	12.21%	12.05%
jpeg8	1161942	17.60%	17.42%	Pdf8	485396	11.32%	10.70%
jpeg9	1236074	17.45%	17.30%	Pdf9	1124000	14.40%	13.72%
Avi0	936358	16.60%	16.49%	MP3-0	3892272	17.24%	17.09%
Avi1	1624708	16.55%	16.42%	MP3-1	5376068	18.07%	17.95%
Avi2	1569058	16.76%	16.59%	MP3-2	3423342	18.27%	18.11%
Avi3	942526	17.71%	17.58%	MP3-3	4139135	18.02%	17.86%
Avi4	1026186	17.38%	17.22%	MP3-4	4331796	18.44%	18.30%
Avi5	909982	17.83%	17.70%	MP3-5	3921572	18.24%	18.09%
Avi6	1212000	17.13%	17.00%	MP3-6	4075520	18.23%	18.04%
Avi7	1579568	17.19%	17.07%	MP3-7	8236801	18.01%	17.88%
Avi8	1236472	17.54%	17.40%	MP3-8	5728433	18.06%	17.92%
Avi9	1153354	17.51%	17.43%	MP3-9	6957257	18.01%	17.84%

Table 4. Activity Savings, after $24-> 8$ reordering patterns reduction (Average 16.60%).