Wire Placement for Crosstalk Energy Minimization in Address Buses

Luca Macchiarulo   Enrico Macii   Massimo Poncino

Politecnico di Torino
Torino, ITALY 10129

Abstract

We propose a novel approach to bus energy minimization that targets crosstalk effects. Unlike previous approaches, we try to reduce energy through capacitance optimization, by ad opting non-uniform spacing between wires. This allows reduction of power, and at the same time takes into account signal integrity. Therefore, performance is not degraded. Results show that the method saves up to 30% of total bus energy at no cost in performance or complexity of the design (no encoding-decoding circuitry is needed), and limited cost in area.

1 Introduction

In deep submicron (DSM) technologies, inter-wire (or crosstalk) capacitances are assuming a dominant effect on the total energy required by buses to transfer information across a chip. As a consequence, low-power bus encoding techniques need to be enhanced to account for this additional contribution to the capacitances that are charged and discharged during communication.
Several recent contributions have provided promising results in this direction. All these approaches tackle crosstalk bus energy by minimizing the number of simultaneous transitions on adjacent bus lines, either by modifying the data sent on the bus [1, 2, 3] through explicit encoders, or by swapping some of the bus lines during layout [4, 5].
All these solutions share a common limitation, that makes them little appealing to physical designers. In some sense, they tackle the wrong problem: They aim at reducing crosstalk by minimizing the number of simultaneous transitions. While this may be a way to reduce crosstalk *power*, it does not reduce crosstalk *by itself*. The main reason why crosstalk makes deep-submicron design hard is because it affects signal integrity. Thus, even a sensible reduction of crosstalk power is of little interest for designers, if it does not guarantee the proper functionality of the design.
Since crosstalk is mainly a capacitive effect, the only way to reduce it is that of reducing the capacitance that causes it. In other terms, in the context of buses, the minimization of crosstalk power should be obtained as a side effect of the minimization of crosstalk capacitances. This would allow the solution to be consistent with the techniques that are typically adopted to reduce crosstalk effects on timing, and use energy consumption as a side metrics.
When considering buses (or any global wire), the most intuitive solution consists of spacing the wires, which tends to reduce crosstalk at the source. Another approach consists of shielding the wires by inserting alternating V_{dd} and ground lines between existing wires [6]. Although the latter seems to be promising, the former is more well-accepted because it does not require additional capabilities from existing place and route tools, and will be used as the reference solution for our work.
The point addressed in this paper is that the conventional, uniform spacing of the bus wires has, as main objective, the minimization of crosstalk effects (e.g., signal integrity). However, crosstalk power is not considered. We introduce a new bus placement technique that targets the minimization of crosstalk energy, and that is compatible with the wire spacing paradigm. This is made possible through a non-uniform spacing of the bus lines, determined according to activity information of the various bus lines, obtained from profiling a set of typical traces. We focus on address buses because of the better predictability of address traces with respect to data traces, as shown in [4].
The proposed technique leverages an accurate capacitive model that considers all electrical effects in order to express the dependency of both the coupling and the self capacitance with respect to the inter-wire distance. The problem of finding an optimal bus spacing is solved with an heuristic algorithm, that provides up-to 40% savings in total bus energy, at no performance overhead or complexity of the design.

2 Crosstalk Effects and Power

In the case of long, global buses, crosstalk can become a serious limiting factor in the timing verification of the design. Crosstalk is mainly a capacitive effect, represented by a high *coupling capacitance* between wires. In DSM technologies, such coupling capacitance (C_X) exceeds the capacitance between the individual wire and ground (*self capacitance*, C_L). Electrical-level simulations for $0.18\mu m$ technologies have shown that C_X can even be three times larger than C_L [1].
With the purpose of quantifying the effects of crosstalk on a bus, we have simulated the structure depicted in Figure 1, that shows the electrical equivalent a simple three-wire bus, where the two outer wires (a and c) exhibit a rising transition, while wire b exhibits a falling transition. All transitions are simultaneous, because we assume that all the bus drivers are clocked. Wires a and c (the *aggressors*) will cause the transition on wire b (the *victim*) to become non-ideal. The effect of the large coupling capacitance between a and b, and c and b will be that of (i) delaying the transition on b, and (ii) causing an initial negative spike at b.

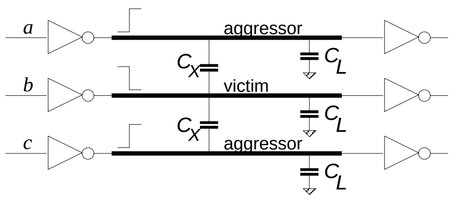

Figure 1. **Crosstalk on a 3-Line Bus.**

From SPICE simulations, it is apparent that delay increase can be harmful in true critical paths: The case of $C_X = C_L$ (corresponding to twice the minimum spacing for a 0.25μm technology) already introduces a delay of 1.0 ns in a 4mm bus, comparable to a few stages of logic. This explains why spacing constraints are needed to enforce high-performance in critical parts of the circuit. However, due to area constraints, not all wires can be spaced as safe signal integrity considerations would suggest. For example, it has been reported (see [7]) that in the design of a last generation microprocessor a fairly large number of wires longer than 2mm have a crosstalk capacitance greater than or equal to their self-capacitance.

These considerations explain the general approach to wiring that we decided to adopt in this paper. While the use of a widely spaced bus is beneficial for both delay and power, it comes at a cost in area. To take into account performance constraints, typically more stringent than area or power, we assume that the designer has given some minimal spacing between wires that guarantees all timing and signal integrity requirements to be satisfied. This is equivalent to giving a new minimal spacing rule that overcomes the technological design rule. However, if the area that has been allocated to the bus during floorplan is larger than the minimum one, we can exploit the extra spacing to reduce the dissipated power.

In order to assess the best wire spacing in terms of energy, we first need a power model. The energy model per cycle for a bus line will thus include the two capacitive effects:

$$E_{bus} = (\alpha_L C_L + \alpha_X C_X) V_{dd}^2, \qquad (1)$$

where α_L and α_X denote the rates at which each capacitance is switched: α_L represents the conventional switching activity of the line, and α_X is related to the simultaneous switching of two adjacent lines. When the transitions on two adjacent lines a and b are aligned in time, there are only two transition pairs that cause C_X to switch: (i) When both a and b switch to different final values, and (ii) when one of the two lines switches, while the other does not, and their final values are different.

Table 1 shows the normalized energy consumption for a two-line bus, when all capacitive effects are considered [1]. Value λ denotes the ratio C_X/C_S. In the table, only $0 \rightarrow 1$ transitions are counted as power dissipating transitions on C_L. This is because they are the ones that actually charge the capacitance, drawing energy from the power supply. In practice, this distinction replaces the $\frac{1}{2}$ factor in the conventional power dissipation model.

The table clearly shows that increasingly larger values of λ will tend to emphasize the importance of the energy due to switching of the coupling capacitances, as opposed to that of the self capacitances. Conventional bus encoding schemes target the minimization of the energy due to C_L. By considering spacing we can modify the values of both C_L and C_X, for each individual bus line and for each pair of bus lines, respectively. In fact, as the next section details, both C_S and C_X are dependent on wire spacing. Such a solution has the advantage of not requiring an explicit codec, nor any substantial layout modifications, and therefore comes for free, given an admissible slack for bus routing space. Furthermore, this does not prevent the use of methods that target α_X and α_C minimizations.

		(b^t, b^{t+1})			
		$0 \rightarrow 0$	$0 \rightarrow 1$	$1 \rightarrow 0$	$1 \rightarrow 1$
(a^t, a^{t+1})	$0 \rightarrow 0$	0	$1+\lambda$	0	0
	$0 \rightarrow 1$	$1+\lambda$	2	$1+2\lambda$	1
	$1 \rightarrow 0$	0	$1+2\lambda$	0	λ
	$1 \rightarrow 1$	0	1	λ	0

Table 1. **Normalized Energy on Two Adjacent Bus Lines.**

3 Capacitance Model

The discussion of Section 2 is mainly focused on time-related crosstalk effects. To consider energy, however, a more accurate energy consumption model is required. Furthermore, since the energy minimization paradigm we adopt is based on spacing the wires, the energy model should be parameterized with respect to the inter-wire distance. We therefore developed a model that considers the effect of spacing on capacitance values and, at the same time, has an analytical expression which is simple enough to be employed inside an optimization engine. In the following, we refer to a specific technology, but a similar approach can be extended to other technologies as well. We first obtained accurate values of capacitances through the use of a well-known 3D capacitance extractor (FastCap [8]), on a 0.25 micron technology. We focused on the evaluation of capacitances for wires laid out in metal 4, with minimal width, and various spacings (from 0.4 to 3.0 microns), under the assumption that metal 3 provides a complete ground plane (thus maximizing the effect of self-capacitances). The extracted capacitances are taken from the arrangement of three equally spaced wires: The values for the cross-capacitance of the middle wire (the plus marks in Figure 2) exhibit the well-known inverse proportionality relation. However, due to the non-uniformity of fringe effects and the presence of the ground plane, no exact inverse proportionality can be inferred, and we resorted to a different functional dependency obtained by interpolation of the data (solid curve).

More interesting is the behavior of the self-capacitance (the crosses in Figure 2). The influence of adjacent wire spacing on self-capacitance was known (see for example [9]), but it has often been regarded as a second-order effect. Mesured values, however, show that the dependency is far from being negligible: For example, capacitance more than doubles in the distance range considered. The trend is clearly asymptotic, where the value of the limit is given by the capacitance of an isolated wire, but the convergence is quite slow. Therefore, we approximated the curve with a logarithm of the distance (dashed curve), which fits reasonably well the results in the considered range. The physical reasons for this behavior derives from the fact that, by increasing inter-wire distance, a larger part of the wire is electrically closer to the ground plane than to the adjacent wires, thus increasing the self-

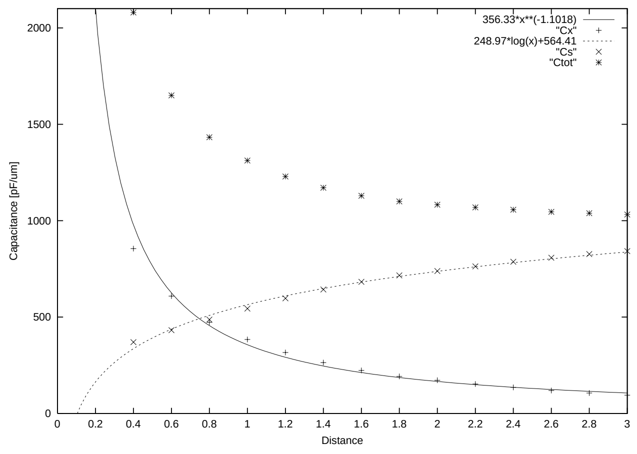

Figure 2. **Crosstalk and Self Capacitance vs. Wire Distance.**

capacitance.

Another interesting consideration concerns the effect of wire distance on *total* capacitance, obtained by summing the self-capacitance with twice the cross-capacitance with the two neighboring wires (stars in Figure 2). The obtained curve is monotonically decreasing (that is, increasing the spacing reduces total capacitance), and converges to the asymptotic value of the isolated wire.

Another important issue for our purposes is the behavior of capacitances with asymmetrical wire disposition (i.e., whose distance from right and left adjacent wire is different). Extraction with various non-symmetrical configurations shows that the self capacitance can be calculated as the average between the capacitance of the distance to the left wire and the capacitance of the distance to the right one. This greatly simplifies the modeling of asymmetrical buses: The self-capacitance can be decoupled by two contributions of the left and right adjacent wires, thus depending only on one parameter.

4 Non-Uniform Wire Placement

Let W be the width of the die area that is available for spacing, that is, the distance between the wires that surround the bus, (the *neighbor* wires), minus the width of the wires. The problem we are addressing consists of placing N bus wires within the allowable width W. The solution can be represented by an array of distances $\mathbf{D} = [d_0, \ldots, d_N]$; the elements d_i identify the distance between wire i and $i+1$. The elements d_0 and d_N denote the distance between the boundary wires of the bus (line 1 and N), and the neighbor wires. Obviously, $\sum_{i=0}^{N} d_i \equiv W$.

All the d_i's are integer multiples of a technology-dependent quantity s, that represents the step of the layout grid. In the following, without loss of generality, we will assume $s = 1$; therefore, array $\mathbf{D}$ stores integer values that represent multiples of s. The actual values in μm can be obtained by multiplying d_i's by the actual value of s (0.1μ in the technology used in the experiments). This fact implies that finding the best distance configuration implies searching a discrete space.

Figure 3 shows an instance of the problem for the case of $N = 4$; the picture shows a distance assignment $\mathbf{D} = [4, 5, 4, 6, 4]$.

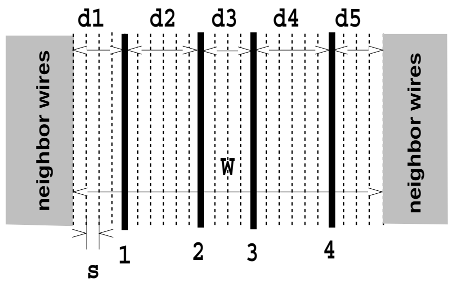

Figure 3. **Bus Placement Example.**

Due to the constraints imposed by the presence of crosstalk, there is a minimum distance between any two wires that has to be enforced. Let this quantity be d_{min}. All distancvalues must be greater than d_{min}.

In order to express the total bus energy, the knowledge of the coupling and of the switching activity (the equivalent of α_X and α_C in Equation 1) are also required. The former is represented by an array $\mathbf{C} = [c_0, \ldots, c_N]$ that expresses the coupling activity between each wire pair, including the neighbor wires. c_0 and c_N are actually fixed, because neighbor wires cannot be profiled, and therefore depend exclusively on the activity of the bus boundary wires. The latter is represented by array $\mathbf{T} = [t_0, \ldots, t_{N-1}]$, that expresses the switching activity of each wire. All the c_i's and t_i's are quantities between 0 and 1.

With reference to the base energy model of Equation 1, the problem can be formulated as follows: *Given N, W,* $\mathbf{C}$ *and* $\mathbf{T}$*, find a distance assignment* $\mathbf{D}$*, such that* $E_{bus}(\mathbf{D})$ *is minimized.*

The solution of this problem presents two main difficulties. First, the non-linearity of the cost function (energy) with respect to independent variable (the wire distances). Second, the size of the search space, that prevents any exhaustive search.

Intuitively, for the problem to have a solution, some slack must be available for the placement of the bus. This happens if the allowed width W is greater than the smallest possible bus ($(N+1) \cdot d_{min}$). The size of the space to be explored is related to the available slack $\sigma = W - (N+1) \cdot d_{min}$, and is roughly proportional to the factorial of $\sigma \cdot N$. This value prevents any exhaustive search of the solution, and requires the use of some heuristic approach. Among the several alternatives available in the literature to solve generic search problems, we have chosen an algorithm that belongs to the class of *local search* methods. It is based on the perturbation of a set of initial solutions, with the objective of exploring as many local minima as possible.

A pseudocode of the search algorithm is shown in Figure 4. The input parameters are the width W, and the profiled activity data, $\mathbf{C}$ and $\mathbf{T}$. The algorithm explores the space starting from *Nruns* different randomly generated initial solutions (Lines 2–3). For each iteration, a bus line j is selected as starting point for the local search (Line 4), and starting from the initial solution $\mathbf{D}_{init}$ (Line 5), the actual local search is started. All the N bus lines are selected as starting points.

The core of the search is shown in Lines 7–28. Starting from bus line j, all wires are sequentially visited modulo N (Line 7). The currently selected line k undergoes the actual perturbation step (Lines 9–26), that consists of a progressive shift of line k of one

```
1   Search($W$, $\mathbf{C}$, $\mathbf{T}$) {
2       for ($i = 1$ to Nruns) {
3           $\mathbf{D}_{init}$ = SelectRandomAssignment();
4           for ($j = 1$ to $N$) {
5               $\mathbf{D} = \mathbf{D}_{init}$;
6               unmark all wires;
7               while ($\exists$ some wire $k$ not marked) {
8                   mark $k$;
9                   while (IsValid($\mathbf{D}$)) {
10                      $\mathbf{D}[k] - -$;
11                      $\mathbf{D}[k+1] + +$;
12                      if (Cost($\mathbf{D}, \mathbf{C}, \mathbf{T}$) < MinCost) {
13                          MinCost = Cost($\mathbf{D}$,$\mathbf{C}$, $\mathbf{T}$);
14                          $\mathbf{D}_{min} = \mathbf{D}$;
15                      }
16                      k++;
17                  }
18                  $\mathbf{D} = \mathbf{D}_{min}$;
19                  while (IsValid($\mathbf{D}$)) {
20                      $\mathbf{D}[k] + +$;
21                      $\mathbf{D}[k+1] - -$;
22                      if (Cost($\mathbf{D}, \mathbf{C}, \mathbf{T}$) < MinCost) {
23                          MinCost = Cost($\mathbf{D}, \mathbf{C}, \mathbf{T}$);
24                          $\mathbf{D}_{min} = \mathbf{D}$;
25                      }
26                  }
28              } /* end while */
            } /* end for
        } /* end for
    }
```

Figure 4. **Pseudocode of the Search Algorithm.**

position to the left (i.e., towards lines $k-1$) as shown in Lines 10–11 and to the right (i.e., towards lines $k+1$) as shown in Lines 20–21. The magnitude of the shift is always s. Any solution better than the current minimum is stored. The shift (to the left and to the right) continues until the current solution is valid (Lines 9 and 19), that is, until all distances in $\mathbf{D}$ are greater than d_{min}. After the local search has terminated, the solution $\mathbf{D}_{init}$ is restored (Line 24), and the next starting point j is chosen.

The algorithm has complexity $O(Nruns \cdot N^2)$. In spite of the relatively small number of solutions that are tried, the algorithm has always found the global miniumum on smaller examples for which an exhaustive search is feasible. In our implementation, we have used *Nruns*=500. With this value, the running time is in the order of a few minutes on an Alpha Station 433 for the case of a 32-bit wide bus.

5 Experimental Results

To assess the effectiveness of the proposed bus placement technique, we have considered address traces for some typical DSP functions, collected using two different code profilers (namely, `Armulator` for an ARM processor and `pixie` for a MIPS processor). Two traces that cover corner cases typical of address traces (i.e., highly sequential streams) have also been used: `Counter` represents a perfect counter that starts from a random address; `CountSkip` is a counter sequence intermixed with random jumps to new locations.

For each set of traces (i.e., ARM and MIPS traces) we have determined the average coupling and switching activities (α_X and α_C in Equation 1), and we have used these data as inputs to the algorithm of Section 4.

The use of average activity information instead of application-specific ones allows the use of the proposed non-uniform spacing paradigm as a general-purpose method for reducing the energy of an address bus, regardless of the trace that is actually transmitted. This averaging process will obviously decrease the efficiency of the encoding scheme, and will yield smaller energy savings than a solution obtained from custom statistics. However, we expect this penalty (whose precise quantification will be given later in the section) to be small, thanks to the similarity of the statistical profiles of typical address traces [4].

Energy results are shown in Tables 2 and 3, for the cases of $d_{min} = 0.4\mu m$ and $0.8\mu m$, respectively. Energy values are determined assuming a supply voltage of $2.5V$, and a bus length of $4mm$. For each value of d_{min}, three different values of slack σ are reported, namely, 10%, 20%, and 30%.

The upper part of each table refers to ARM traces, while bottom rows concern MIPS applications. Column *Initial* shows the bus energy for the case of a minimally spaced bus, i.e., where all wire distances are equal to d_{min}. Then, for each value of σ, the energy value for two bus configurations that exploit the corresponding slacks are reported. Column *Uniform* represents the case of equally spaced bus lines according to the available slack. This arrangement represents what it is typically given by an automatic tool. Column *Non-Uniform* shows the results of our profile-based, non-uniform spacing resulting from the algorithm of Section 4. Savings are referred to the energy of column *Initial*.

The results show that the non-uniform spacing provides sizable energy savings with respect to the case of a minimally spaced bus (40.4% for $d_{min} = 0.4$, and 21.5% for $d_{min} = 0.8$ and $\sigma = 30\%$, on average). More important, our approach outperforms other manual solutions that use the available slack in a non-systematic way. For example, for $\sigma = 30\%$ the straightforward uniform spacing saves only 17.8% (for $d_{min} = 0.4$), and 9.7% (for $d_{min} = 0.8$), on average. The advantage of the non-uniform spacing is more evident for smaller values of σ.

As expected, the magnitude of the savings decreases as d_{min} increases, because the effect of crosstalk becomes smaller.

We have also validated the above results by extracting the capacitances from the actual layout of the bus, for some of the configurations reported in the tables, and re-computed their energy consumption. On average, the difference between the energy values obtained from the capacitance model of Section 3 and the re-computed energy is within 7%, comparable to the accuracy of the models.

Finally, in order to assess the loss in optimization potential due to the use of "average" statistics instead of trace-specific ones, we have evaluated the difference between the savings obtained with a custom spacing (i.e., derived from a specific trace – column *Custom*), and those of Tables 2– 3 (column *General*). Results are shown in Table 4, and are relative to the case of $d_{min} = 0.4$, and $\sigma = 10\%$. The average (over all traces) loss in optimization potential is just 1.7%, that demonstrates the applicability of a fixed non-uniform spacing to a generic address buses.

Trace	*Initial*	$\sigma = 10\%$				$\sigma = 20\%$				$\sigma = 30\%$			
		Uniform		*Non-Uniform*		*Uniform*		*Non-Uniform*		*Uniform*		*Non-Uniform*	
	E [nJ]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]
`AdaptFilter`	923	869	5.8	703	23.8	824	10.7	642	30.4	774	16.1	603	34.6
`Butterfly`	812	763	6.1	610	24.8	723	10.9	552	31.9	679	16.3	519	36.1
`IirDemo`	549	517	5.9	419	23.7	491	10.7	382	30.4	461	16.1	350	34.5
`Integrator`	387	364	6.0	294	24.0	345	10.8	268	30.8	325	15.9	252	34.9
`Count`	562	523	6.9	378	32.7	496	11.7	323	42.4	456	19.0	296	47.3
`CountSkip`	68	64	6.9	46	32.7	60	11.7	39	42.4	55	19.0	36	47.3
`DCT`	33	31	7.2	23	29.6	29	11.7	20	38.2	27	18.9	19	43.1
`DashBoard`	514	479	6.9	370	28.1	451	12.2	329	35.9	417	18.9	303	41.1
`FFT`	71	67	6.9	50	29.4	63	12.1	44	37.8	58	18.7	40	43.1
`MatMult`	77	72	6.3	55	29.1	69	11.1	49	36.3	62	19.4	45	42.0
Average			6.5		27.8		11.3		35.6		17.8		40.4

Table 2. **Energy Results for the Uniform and Non-Uniform Spacing ($d_{min} = 0.4$).**

Trace	*Initial*	$\sigma = 10\%$				$\sigma = 20\%$				$\sigma = 30\%$			
		Uniform		*Non-Uniform*		*Uniform*		*Non-Uniform*		*Uniform*		*Non-Uniform*	
	E [nJ]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]	*E* [nJ]	Δ [%]
`AdaptFilter`	626	606	3.1	547	12.6	590	5.8	526	15.9	578	7.6	514	17.8
`Butterfly`	545	528	3.2	473	13.2	513	6.0	454	16.8	502	8.0	443	18.7
`IirDemo`	372	361	3.1	325	12.6	351	5.8	313	16.0	344	7.7	306	17.8
`Integrator`	263	255	3.1	230	12.6	248	5.7	221	15.9	243	7.6	216	17.9
`Count`	348	333	4.2	281	19.1	319	8.2	265	23.7	310	10.9	261	25.0
`CountSkip`	42	41	4.2	34	19.1	39	8.2	32	23.7	38	10.9	32	25.0
`DCT`	20	19	4.2	17	17.2	19	8.1	16	21.6	18	10.9	15	23.4
`DashBoard`	316	302	4.4	264	16.6	290	8.4	250	20.9	280	11.3	243	23.2
`FFT`	44	42	4.3	36	16.9	41	8.2	35	21.4	39	11.0	34	23.5
`MatMult`	48	46	3.9	40	16.7	44	8.4	38	20.8	42	11.1	37	23.2
Average			3.7		15.6		7.2		19.7		9.7		21.5

Table 3. **Energy Results for the Uniform and Non-Uniform Spacing ($d_{min} = 0.8$).**

Trace	*General* *E* [nJ]	*Custom* *E* [nJ]	Δ [%]
`AdaptFilter`	703	693	-1.4
`Butterfly`	610	603	-1.1
`IirDemo`	419	416	-0.7
`Integrator`	294	290	-1.4
`Count`	378	378	0.0
`CountSkip`	46	46	0.0
`DCT`	23	22	-4.5
`DashBoard`	370	362	-2.2
`FFT`	50	49	-2.0
`MatMult`	55	53	-3.8
Average			-1.7

Table 4. **Energy Results for General and Custom Spacings.**

6 Conclusions

Crosstalk effects on power can be significant in deep submicron design, but their minimization needs to take into account timing issues. We have shown that through adequate usage of non-uniform wire spacing savings of up to 40% can be obtained at no complexity or performance cost, with modifications that are essentially transparent to the designer.

References

[1] P. P. Sotiriadis, A. Chandrakasan, “Bus Energy Minimization by Transition Pattern Coding (TPC) in Deep SubMicron Technologies,” *ICCAD’00*, pp. 322-327, Nov. 2000.

[2] J. Henker, H. Lekatsas, “A^{2}BC: Adaptive Address Bus Coding for Low-Power Deep Sub-Micron Designs,” *38th DAC*, pp. 744-749, Jun. 2001.

[3] K.-W. Kim, K.-H. Baek, N. Shanbag, C.L. Liu, S.-M. Kang, “Coupling-Driven Signal Encoding Scheme for Low-Power Interface Design”, *ICCAD’00*, pp. 317-321, Nov. 2000.

[4] L. Macchiarulo, E. Macii, M. Poncino, “Low-Energy Encoding for Deep-Submicron Address Buses,” *ISLPED’01*, pp. 176–181, Aug. 2001.

[5] Y. Shin, T. Sakurai, “Coupling-Driven Bus Design for Low-Power Application-Specific Systems”, *38th DAC*, pp. 750–753, Jun. 2001.

[6] S.P. Khatri, A. Mehrotra, R.K. Brayton, R.H.J.M. Otten, A. Sangiovanni-Vincentelli “A novel VLSI layout fabric for deep submicron applications” *36th DAC*, pp. 491-496, Jun. 1999.

[7] R. Kumar, “Inerconnect and Noise Immunity Design for the Pentium 4 Processor”, Intel Technology Journal Q1, 2001.

[8] K. Nahors, J. White, “FastCap: a Multipole Accellerated 3-D Capacitance Extraction Program”, *IEEE Transactions on CAD*, 10(11), pp. 1447-1459, Nov. 1991.

[9] G. Servel et al.“Inductive Effects on Crosstalk Evaluation”, *4th IEEE International Interconnect Technology Conference*, San Francisco, Jun. 2001.