Power Saving in CMOS Processors by Optimal Wire Spacing

Abstract – **Interconnect power is a significant part of dynamic power dissipation in microprocessors. Cross-capacitance between adjacent wires is a major contributor to this power consumption, and it can be reduced by post-processing of the layout for optimal allocation of spaces between wires. Necessary and sufficient conditions for the existence of optimal space allocation are derived in this paper. At the optimum, every wire must be in equilibrium of its line-to-line capacitance density on its two sides, weighted by signal activity factors. A practical implementation which has been used to reduce power in the design of a recent commercial high-end microprocessor is presented, exhibiting an average reduction of 16.8% per layer in the upper metal layers.**

1. INTRODUCTION

The dissipation of power has become a major concern in processor design because of the drive to deliver lighter mobile computers with longer battery life and the growing awareness to environmental heating. Hence, every opportunity to save power is exploited, such that a total power saving is accumulated by adding small contributions from various design levels and approaches. Power reduction was addressed at different design levels [1] [2], from architecture and system level through RTL synthesis, signal encoding, circuit implementation, all the way down to layout implementation, which is the focus of this paper. We claim that interconnect power (the dynamic power dissipated because of charging and discharging wire capacitances, which is the dominant dynamic power component in processors [4]), can be significantly reduced by optimizing inter-wire spacing in the layout.

Commercial routing tools and manual artwork of mask designers tend to produce densely-spaced wires, while leaving empty areas ("white spaces") in nearby areas, instead of spreading the wires evenly. Tools and humans typically do not take advantage of the entire area available for layout implementation, because routing is usually a sequential process. Therefore, the more area is saved at any routing step, the better is the chance to complete all required interconnections [3]. However, this approach results in non-uniform area utilization, leaving “white areas” in the layout (Figure 1). Based on this observation, we propose to eliminate the white space by spreading-out wires, thus increasing inter-wire spaces in order to reduce line-to-line capacitances and save power. Local optimization techniques by wire spacing for power and cross-talk minimization were reported in the literature [14], [15], [16] but our technique is global and therefore more effective.

The optimization problem of minimizing the power consumed by interconnecting wires on a die by adjusting their spacing is formulated and solved in this paper. It is assumed that interconnects have already been routed (manually or by some CAD tool), so that interconnection topology cannot be changed in any metal layer. It is also assumed that wire widths have been set to satisfy signal delay and other design goals such as reliability. The optimal spacing allocation can thus be made as a post-processing step performed on the layout. The paper also demonstrates an industrial design flow which was used to implement this optimization technique on an actual processor design.

The rest of this paper is organized as follows. In the next section the circuit and layout models used throughout the discussion are presented. A necessary and sufficient condition for every wire so total power is minimized is proven in section 3. An iterative solution algorithm is described in section 4. Results obtained for a recent high-end microprocessor designed in 65 nanometers technology are presented in section 5. Delay constraints and minimal spacing rules addressed in section 6.

2. INTERCONNECT MODELING ASSUMPTIONS

The interconnecting wires in upper metal layers typically run in alternating orthogonal directions, e.g. wires residing in even layers are vertical and wires in odd layers are horizontal, as shown in Figure 2. The influence of jogs (Figure 2) on power and delay is negligible; therefore they are excluded from the optimization process. The interconnect power associated with a logic signal is proportional to it's total capacitance and to it's average amount of switching as compared to clock signal (the signal’s *activity factor*). Drivers and receivers are assumed as given, and their optimization is beyond the scope of this paper.

Spacing optimization is carried out at each layer independently of the other layers as follows: As shown in Figure 2, shifting wires in one layer doesn’t affect spacing of (the orthogonal) wires in the layers above it and below it. One may think that significant wire shift may not always be possible, since a wire movement in layer l might cause overlap of wire segments in an adjacent (orthogonal) layer below or above it. However, such cases require a large wire shift, which are typically impossible because of high layout density and statistically uniform distribution of shifted wires over the layout. For example, when our technique was implemented on a real industrial layout, the largest wire shift was only 1.6 μm. In addition, the maximum wire shift is limited by the power grid network which is kept fixed with no shift. Although the length of horizontal wires in layers $l-1$ and $l+1$ may slightly change, their variations are assumed negligible for any practical consideration.

The fundamental model we use to derive optimal spacing conditions is shown in Figure 3. There, a few wires run in parallel and the entire layout segment is shielded on both sides by wires connected to ground. Shielding wires do not make logical transitions; hence they do not consume any power. Locations of shielding wires are constant and distance between them is A.

Switching power consumed by a signal wire is associated with its self capacitance to ground planes (representing the adjacent metal layers) and with line-to-line capacitance to other wires of same layer as shown in Figure 3. We say that two wires are "visible" to each other if they have a common span and there are not any other wires between them along this span. ***Spacing visibility graph*** $G(U,E)$ is a directed graph whose vertices U correspond to wires and the arcs E correspond to spaces between wires visible to each other (Figure 3). Only line-to-line capacitance to visible wires is accounted. The progression of VLSI process technology has made this line-to-line term dominant over others, and its importance is expected to grow in future generations [9], [10], [11]. The line-to-line capacitance between two adjacent wires is proportional to their common span, and inversely proportional to the real power of space between them [13]. We say that $I_i \prec I_j$ if I_i and I_j satisfy: 1) the intersection of their vertical span is non empty, 2) x_i and x_j, the abscissas of I_i and I_j, respectively, satisfy $x_i < x_j$, and I_i and I_j are visible to each other.

Every signal σ_i has some activity factor α_i ranging from $\alpha_i = 0$ if it never switches (e.g., shields or power delivery network), to $\alpha_i = 1$, if it switches twice at every cycle (e.g., clocks). Signal activity factors can be derived using industrial power simulators by checking signal activity factor in different circuit switching scenarios and then averaging over all cases [1]. The power contributed by the line-to-line capacitance between σ_i and σ_j depends on α_i, α_j and Miller Coupling Factor (MCF) between σ_i and σ_j. According to Miller's theorem the simultaneous switching of two signals in identical and opposite directions yields MCF of 0 or 2, respectively, or -1 to 3 if worst-case transition slopes are assumed[17] . For calculating cumulative average power over many transitions, an average MCF of 1 is assumed for internal wires. Under this assumption the power contributed by the line-to-line capacitance between σ_i and σ_j is proportional to $\alpha_i + \alpha_j$.

A left-to-right topological order of the wires is maintained in each layer. The layout manipulations described in this paper involve only wire spacing optimizations, which preserve the order of the wires as given in the initial interconnect topology.

We assume that the widths $w_0, w_1, ..., w_{n,} w_{n+1}$ of the wires are predefined and thus are not subject to change in the optimization. This assumption agrees with VLSI design practice, where wire widths are set very early in the design flow according to signal propagation delay goals. Optimal spacing, however, is more opportunistic and is addressed late in the design. There, all interconnects are already implemented with their specified space, so the unused “white area” can be distributed among wires in order to reduce their line-to-line capacitance.

Let l_{ij} be the common span of I_i and I_j in which they are visible to each other. If I_i and I_j are not visible to each other l_{ij} is undefined, but for the mathematical discussion we set it to be identically zero. The space $x_j - x_i$ between I_i and I_j is defined if and only if $l_{ij} > 0$. It needs to satisfy the following constraint, which accounts for the predefined wire widths and the minimum wire spacing dictated by the process technology:

$$x_j - x_i - \left(w_j + w_i\right)/2 \geq S_{min}, \quad I_i \prec I_j \tag{1}$$

The line-to-line capacitance c_{ij} associated with I_i and I_j is given by

$$c_{ij} = \kappa \frac{l_{ij}}{\left(x_j - x_i - \left(w_j + w_i\right)/2\right)^a}, \tag{2}$$

where a is a real constant. For first-order analysis one may assume a=1, but a better fit to current technology is obtained with the value of a slightly greater than one [13]. The benefit of our optimization technique increases as this parameter increases.

The factor κ depends only on process technology. The total switching power $P^{cross}\left(\overline{x}\right)$ resulting from line-to-line capacitance is therefore proportional to:

$$P^{cross}\left(\overline{x}\right) \propto \sum_{0\le i\le n}\sum_{i<j\le n+1}\left(\alpha_j + \alpha_i\right)c_{ij} = \\ = \kappa \sum_{0\le i\le n}\sum_{i<j\le n+1}\frac{\left(\alpha_j + \alpha_i\right)l_{ij}}{\left(x_j - x_i - \left(w_j + w_i\right)/2\right)^a} \tag{3}$$

The goal is to find $\overline{x} = \left(x_1, ..., x_n\right)$ that minimizes (3). Recall that I_0 and I_{n+1} are fixed, hence we assume that $x_0 = 0$ and $x_{n+1} = A$ (total routing area).

3. NECESSARY AND SUFFICIENT LAYOUT CONDITION FOR MINIMAL POWER

Lemma 1: The minimum of (3) subject to (1) is global.

Proof: Let us define $s_{ij} = x_j - x_i - \left(w_j + w_i\right)/2$ to be the spacing between two visible wires. Substitution s_{ij} into (1) and (3) yields the following minimization problem:

$$\text{minimize:} \quad \kappa \sum_{0\le i\le n}\sum_{i<j\le n+1}\frac{\left(\alpha_j + \alpha_i\right)l_{ij}}{s_{ij}^{\ a}} \tag{4}$$

$$\text{subject to:} \quad s_{ij} \geq S_{min}, \quad I_i \prec I_j, \tag{5a}$$

$$s_{ij} - x_j + x_i + \left(w_j + w_i\right)/2 = 0, \quad I_i \prec I_j, \tag{5b}$$

$$0 < x_i < A, \quad 1 \le i \le n \tag{5c}$$

The objective function (4) is convex and same are the constraints (5a) - (5c). Consequently there is one minimum which is global [5].☻

Consider now the variable abscissa x_i of a wire I_i whose width is w_i $1 \le i \le n$. Denote its left and right visible wires by $I_{i,j}^l$ and $I_{i,j}^r$, respectively, where the superscript designates left and right sides of I_i and the subscript j is varying. We use the same indexing notation for the corresponding abscissas, widths, lengths of wires overlap and activity factors.

Let us ignore for the moment the requirement (5a) of minimum spacing, and replace it by $s_{ij} > 0$, which still guarantees the partial order preservation in (5b). Although it is not feasible for VLSI layout, it simplifies the characterization of the optimal spacing yielding minimum power. We’ll return to (5a) and take it into account in the real implementation of wire spacing. Formally, (5a) is replaced by

$$s_{ij} > 0, \quad I_i \prec I_j \tag{5d}$$

Theorem 1 *(necessary and sufficient condition for minimal interconnect power):* A necessary and sufficient condition so that the switching power expression in (4) is minimized subject to the constraints (5b)-(5d) is that every wire $I_i, 1 \le i \le n$ satisfies:

Table I. Optimization results for different design blocks

Block number	*Area, mm²*	*Transistor count*	*Net count*	*Nets touched*	*Max wire shift, um*	*Initial Power, % of total*	*Power improvement, %*	*Run time ,s*
1	7.87	$3.1*10^6$	$1.2*10^5$	$1.1*10^5$	0.54	58.82	14.2	$1.35*10^3$
2	7.01	$3.5*10^6$	$7.1*10^4$	$5.7*10^4$	0.78	16.89	20.9	$0.94*10^3$
3	6.78	$4.4*10^6$	$7.1*10^4$	$6.1*10^4$	0.83	11.55	21.6	$1.06*10^3$
4	5.14	$4.2*10^6$	$4.6*10^4$	$4.1*10^4$	1.21	6.66	17.8	$0.76*10^3$
5	3.2	$1.9 * 10^6$	$4.4*10^4$	$3.7*10^4$	1.55	6.08	20.5	$0.7*10^3$
Total	**30.02**	**$17.1*10^6$**	**$3.5*10^5$**	**$3.1*10^5$**	**-**	**100**	**16.81**	**-**

$$\sum_j \frac{l^l_{i,j}\left(\alpha_i+\alpha^l_{i,j}\right)}{\left[x_i-x^l_{i,j}-\left(w_i+w^l_{i,j}\right)\right]^{a+1}}=\sum_k \frac{l^r_{i,k}\left(\alpha_i+\alpha^r_{i,k}\right)}{\left[x^r_{i,k}-x_i-\left(w_i+w^r_{i,k}\right)\right]^{a+1}} \quad (6).$$

Summation on left and right hand sides of (6) is taken on all left and right visible wires, respectively.

Proof: By substitution of (5b) into (4) it follows that the power consumed by wire I_i is proportional to:

$$\sum_j \frac{l^l_{i,j}\left(\alpha_i+\alpha^l_{i,j}\right)}{\left(x_i-x^l_{i,j}-\left(w_i+w^l_{i,j}\right)\right)^{a}}+\sum_j \frac{l^r_{i,k}\left(\alpha_i+\alpha^r_{i,k}\right)}{\left(x^r_{i,k}-x_i-\left(w_i+w^r_{i,k}\right)\right)^{a}} \quad (7)$$

The minimum of (4) is obtained at an internal point of the region $s_{ij} > 0,\ \ I_i \prec I_j$. Otherwise, there would be some $s_{ij} = 0$. This, however, would set (4) to infinity, hence not a minimum.

Since the minimum is obtained at an internal point, and by lemma 1 the minimum is global, a necessary and sufficient condition to minimize (4) is that its derivative by the abscissa of every wire is zero. Differentiation of (7) by x_i yields (6) [5]. ☻

The physical interpretation of Theorem 1 is that it is necessary and sufficient for minimum interconnect power that every wire will be in equilibrium, where the sum of its left side capacitors derivatives weighted by their corresponding activity factors is equal to that of the right side.

Solving (6) for all wires together with the constraints (5b)-(5d) involves a large number of nonlinear equations and linear inequalities. Its solution for a typical VLSI layout can be very tedious. In the following we'll present an iterative algorithm for practical problem solution.

4. ITERATIVE ALGORITHMS FOR POWER MINIMIZATION

An iterative algorithm which minimizes the total switching power is proposed. It is based on the equilibrium condition for minimum stated in Theorem 1. It has been implemented and successfully used for power reduction in the design of a commercial 65 nanometer high-end microprocessor. Some power reduction results are presented in section 5 below. This approach involves utilization of vacant area in layout, which has been exploited for enhancing manufacturing yield by tools such as [8], using an iterative algorithm to balance the white space between interconnects in VLSI layout as was proposed in [1].

The algorithm works on one wire at a time while maintaining a global view of the other wires. It repositions a wire between its left and right visible wires, such that local equilibrium is achieved. According to Theorem 1, at a non minimum point there exists at least one wire which is not in equilibrium. We then shift it to the abscissa yielding equilibrium by calculating x from equation (6). Article [1] proved that such iterations converge to a configuration where all wires are in equilibrium.

It has yet to be seen that the repositioning of a single wire indeed reduces the total power. Considering (3), the only affected terms are those which involve the shifted wire and its left and right visible ones. These terms are expressed in (7) within the proof of Theorem 1. This amount of power appears only once in (3) and its value after repositioning has been lowered, hence the net power change is negative. We can summarize in the following theorem:

Theorem 2: The iterative algorithm which positions wires one at a time in their local equilibrium abscissa converges to a limit which is the global minimum of switching power incurred by the wires.

Proof: The infinite sequence of power values obtained by the iterative algorithm is positive and monotonic decreasing, hence converging to a limit where all wires are in equilibrium. Theorem 1 ensures that this limit is indeed the global minimum. ☻

A few implementation comments follow. In order to ensure fast convergence, wires are treated in decreasing order of their imbalance of left and right weighted capacitance derivatives. Clearly, this order is dynamically changing, since the balancing of a single wire is changing the imbalance of the other wires visible to it.

Efficient handling of the dynamically changed imbalance ordering is possible by using a heap data structure, where the most imbalanced wire is stored at heap's top [6]. Wires are popped from the top of the heap one at a time, and then repositioned at their equilibrium abscissa. Such operation consumes $O(1)$ time, assuming that the number of visible wires of any wire is bounded, which is the practical situation in VLSI layout. The balanced wire is then reentered into the heap. The imbalance of the adjacent wires is then recalculated and their location in the heap is updated according to their new imbalance value. This operation takes $O(\log n)$ time [6]. The speed of convergence of such an iterative balancing procedure has been studied in [1].

5. EXPERIMENTAL RESULTS

A pictorial example of real spacing optimization is shown in Figure 5, where next to every wire its corresponding activity factor is written. As shown in Figure 5(b) the optimization algorithm distributes the spacing according to the relative weight of wires' activities. The algorithm described in the previous section is exemplified on a number of circuit blocks from a high-end microprocessor designed in 65nm process technology. Only global interconnect layers (i.e. wires routed on 5, 6, 7 and 8 metal layers) were optimized in these blocks. Signal activity factors were extracted from relevant program traces using industrial tools [1]. As follows from the results, more than 95% of the nets have activity factors less than 0.2.

Optimization results for all blocks are presented in Table 1. In Figure 7 metal breakdown and optimization results for each block are shown. As can be seen, full chip cross-capacitance interconnect power is reduced by 16.8%. According to [4] and Figure 8, which shows dynamic power breakdown for the high-end microprocessor, this amount translates to about 1.68% of the total dynamic power. In a real industrial design environment where the algorithm was deployed, such a reduction is very significant. Although today the leakage power can amount to half of the total power, in future technologies it can drop significantly because of usage of tri-gate devices, and therefore our technique will become even more advantageous.

The difference in power reduction among the various blocks is explained by differences in signal activities, metal density and existing wire spaces.

6. MAINTAINING DELAY CONSTRAINTS WHILE MINIMIZING POWER

The line-to-line capacitance obtained for power minimization may not be optimal for delay optimization. Though the improvement in total capacitance weighted by activity factors will statistically work in favor of reducing capacitance weighted by drivers' resistance, the changes may also result in min and max delay violations. Two different approaches to tackle this problem are described. The first one is preventive and avoids any delay violation. It was the practice used for the design mentioned in this paper. The other is a corrective approach which fixes violations after they have occurred.

A flow which prevents delay violations is illustrated on Figure 4. The spacing algorithm is executed first and all parasitics are modified accordingly. A timing analysis is then performed in order to discover degraded signals that have become critical. The spacing algorithm is executed again on the original input data, but wires of degraded signals together with their visible wires are held fixed. Another timing analysis is performed in order to check whether other timing degradation has occurred. This iteration continues until convergence. Usually two iterations suffice.

A smarter but more difficult to implement approach is to restore all original delays by post-resizing drivers in order to fix max and min delay violations. In what follows we will be more pessimistic and consider the impact of fixing all delay changes rather than just max and min delay violations.

As a first step we need to express driver size sensitivity to delay change. A simplified Elmore delay model of the driver-receiver pair is given by [18]: $D=\left(R+aL/W\right)\left(C+bLW+cL\left(1/S'+1/S''\right)\right)$ where R is driver's resistance, L is wire length and W is its width, C is the capacitive load of the receiver, S' and S'' are the spaces on the two sides of the interconnecting wire, and a, b and c are process technology parameters. The sensitivity is then given by $\left(dR/dD\right)/\left(R/D\right)=(1+aL/WR)$.

The sensitivity depends therefore on wire length and width, process technology sheet resistance and driver's resistance. Figure 9 plots the change in percents that needs to take place by driver size in order to restore the delay for one percent of delay change, as a function of driver size. We simulated minimum width wires of several top-level metal layers with appropriate sheet resistance of 65 nanometer process technology. Several lengths $L=500\mu m$, $1000\mu m$ and $3000\mu m$ were measured for driver's resistance varying from 50Ω to $1.5k\Omega$. Figure 9 shows the results for the worst metal layer. As shown in the plot, driver size is more sensitive in longer\ interconnect, and strong (low resistance) drivers are more sensitive than weak (high resistance) ones. As an example, a change of 10% of delay incurred at a signal with a driver of 100Ω and wire length of $1000\mu m$ is recovered by a change of 20% in driver size.

The histogram in Figure 6 illustrates the distribution of delay change incurred in the top-level interconnects as a result of spacing optimization. As can be clearly seen, for about 80% of the interconnects the amount of change is negligible and falls in the range of simulation accuracy. We have therefore to restore the delays of 20% of the top-level interconnects. Recall that this is still worst case analysis since the delay change of majority of those doesn't result in max or min delay violation.

In order to calculate the amount of driver size changes implied by delay restoration, the histogram in Figure 6 is combined with the driver size sensitivity in Figure 9, thus yielding a distribution of driver size change which is not shown due to paper size limitations. This data is further used to calculate the amount of power growth resulting from resizing (both upsizing and downsizing), which eventually yielded 0.1% of the total chip power consumption. Recalling that Table 1 yielded 1.68% power save, we are left with 1.58% net power saving.

7. CONCLUSION

The industrial practice of low-power design requires exploitation of every possible contribution to saving in the power budget. In this paper we have presented a mathematical analysis of interconnect power minimization, and described a method to reduce interconnect power, which is the dominant component of dynamic power on chip, by reallocation of white space between wires according to the proven optimality conditions. The technique is employed as a post-processing step on the layout while keeping the power grid fixed. It is easy to incorporate into a complete design flow. Implementation of this method on a state-of-the-art processor design in 65nm technology has yielded a reduction of about 17% in interconnect power. Although this corresponds to only a couple of percentage points in the total power, this result is industrially significant. Further relative gains are expected from this technique in future technologies.

8. ACKNOWLEDGEMENTS

The authors would like to thank to Julian Pogerov, Amit Erez and Wael Hendawi for their helpful comments.

9. REFERENCES

[1] S. Borkar, "Low power design challenges for the decade," *Proceedings of the 2001 conference on Asia South Pacific design automation*, pp. 293 – 296.

[2] S. Devadas and S. Malik, "A survey of optimization techniques targeting low power VLSI circuits*," Proceedings of the 32nd ACM/IEEE conference on Design automation*, 1995, pp. 242 – 247.

[3] C. Li, M. Xie, C-K Koh, J. Cong and P. H. Madden, "Routability-Driven Placement and White Space Allocation", *ICCAD 2004*, pp. 394 – 401, November 2004.

[4] N. Magen, A. Kolodny, U. Weiser and N. Shamir, "Interconnect-power dissipation in a microprocessor*", Int. Workshop on System-level interconnect prediction*, pp. 7-13, Paris, 2004

[5] D. G. Luenberger, Linear and Nonlinear Programming, Chapter 6.5*, Addison Wesley*, 1984.

[6] T. H. Cormen, C. H. Leiserson and R. L. Rivest, Introduction to Algorithms, *MIT Press*, 2nd Edition. 2001.

[7] T. C. Hu, Integer Programming and Network Flows, *Addison Wesley*, 1969.

[8] Sagantec, Xtreme – a wire spacing tool for manufacturing yield enhancement.

[9] R. Ho, K. Mai and M. Horowitz, "The future of wires", *Proceedings of the IEEE*, Vol. 89, no. 4, Apr. 2001.

[10] D. Sylvester and K. Keutzer, "Getting to the bottom of deep submicron", *in Proc. ICCAD*, pp. 203-211, 1998.

[11] 2005 ITRS report, available online http://www.itrs.net/reports.html

[12] A. Abou-Seido, B. Nowak and C. Chu, " Fitted Elmore Delay: A Simple and Accurate Interconnect Delay Model*", IEEE ransactions on VLSI Systems*, vol. 12, no. 7, pp. 691-696, July 2004.

[13] S. Wong, G.W. Lee, D. J. Ma, "Modeling of Interconnect Capacitance, Delay and Crosstalk in VLSI", *IEEE Transactions on Semiconductor Manufacturing*, Vol.13, No.1, Feb 2000.

[14] E. Macii, M. Poncino and S. Salerno, "Combining Wire Swapping and Spacing for Low-Power Deep-Submicron Buses", *Proc.of the 13th ACM Great Lakes symposium on VLSI*, pp. 198-202, 2003

[15] K. Chaudhary, A. Onozawa and E. Kuh, "A spacing algorithm for performance enhancement and cross-talk reduction," *Proc. of ICCAD-93*, pp. 697 - 702.

[16] P. Saxena and C. L. Liu, "An algorithm for crosstalk driven wire perturbation", *IEEE Trans. on CAD*, Vol. 19, No. 6, 2000, pp.691-702

[17] P. Chen, D.A. Kirkpatrik and K. Keutzer, "Miller factor for gate-level coupling delay calculation", *Proc. IEEE/ASM Intl. Conf. on CAD*, 2000, pp. 68-75

[18] H. Bakoglu, "Circuits, Interconnects, and Packaging for VLSI", Addison-Wesley,1990

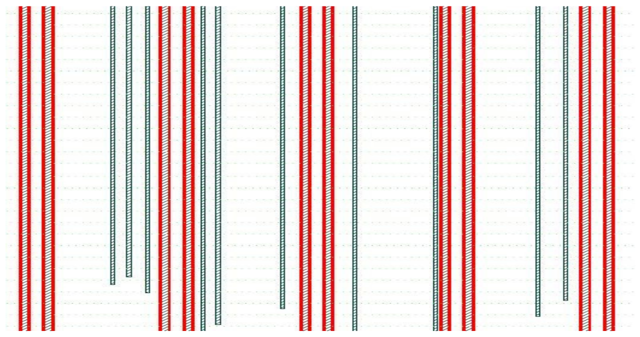

Figure 1 Section of layer 7 layout. Power grid wires are emphasized. White spaces can be well seen on the picture

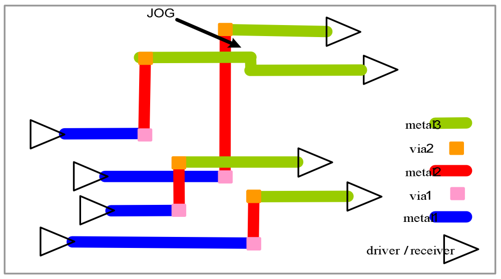

Figure 2: Typical interconnect patterns: A drivers transmits a signal which propagates through interconnecting wires of various layers. Consecutive layers route wires in alternating orthogonal directions. Connections from layer to layer are made by vias. Some wires may have jogs.

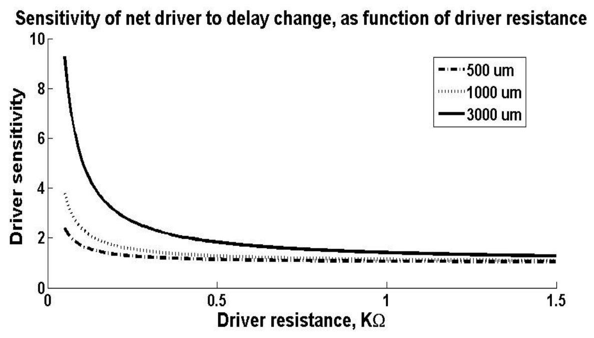

Figure 9: Sensitivity of driver change to spacing change for typical values of spacing from S_{min}=0.14 um up to $3S_{min}$=0.42 um

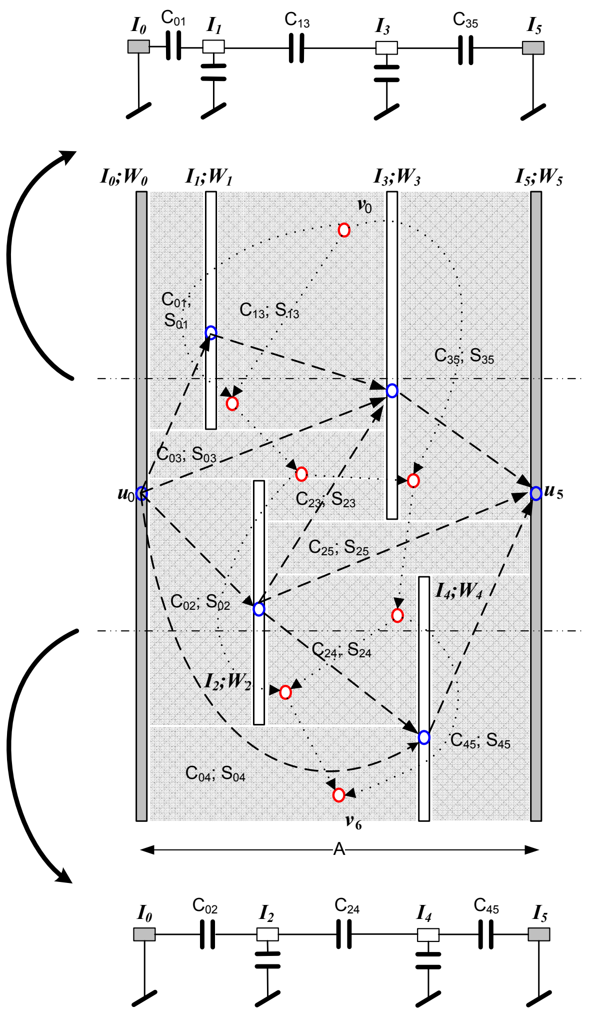

Figure 3: An example layout model of a metal layer, it's corresponding spacing visibility graph (shown in dashed lines), and it's weighted capacitance derivative graph (shown in dotted lines), Two cross sections of layout are shown in order to demonstrate wire cross-capacitances

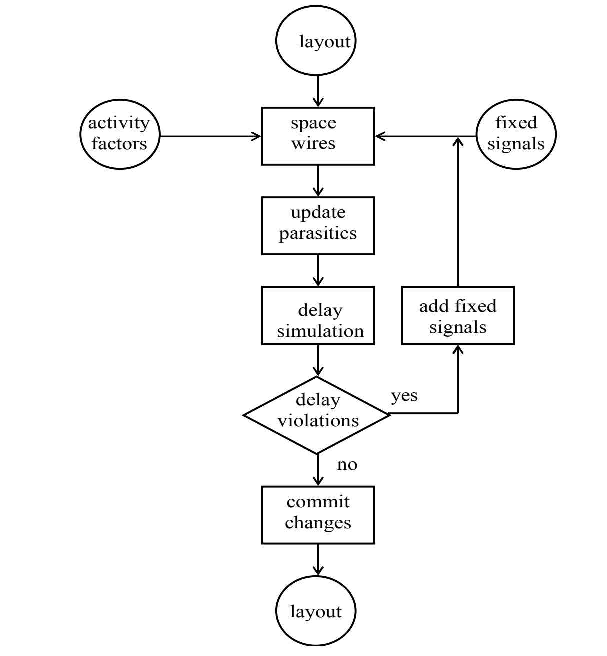

Figure 4: Proposed flow for spacing optimization

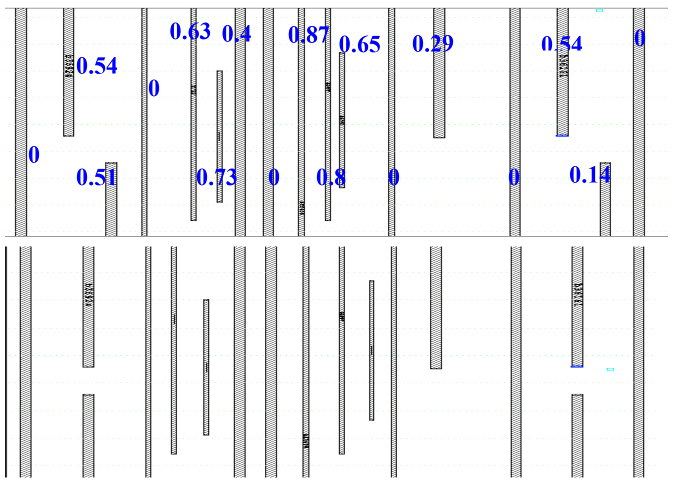

Figure 5: A small clip of 5th metal layer is presented. Snapshots of before (top) and after optimization (bottom) are shown. Activity factors for each wire are shown. Notice that the 5th wire from the right with activity of 0.29 is blocked (by other wirers than are not shown on the clip) and therefore isn't placed after optimization in the middle in between two wires with activity 0.

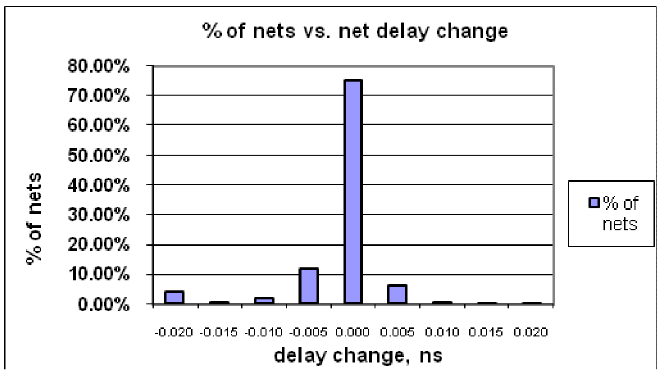

Figure 6: Distribution of delay changes incurred by power minimization. The right tail corresponds to delay increase which may cause max delay violations. The left tail corresponds to delay decrease which may cause min delay violations.

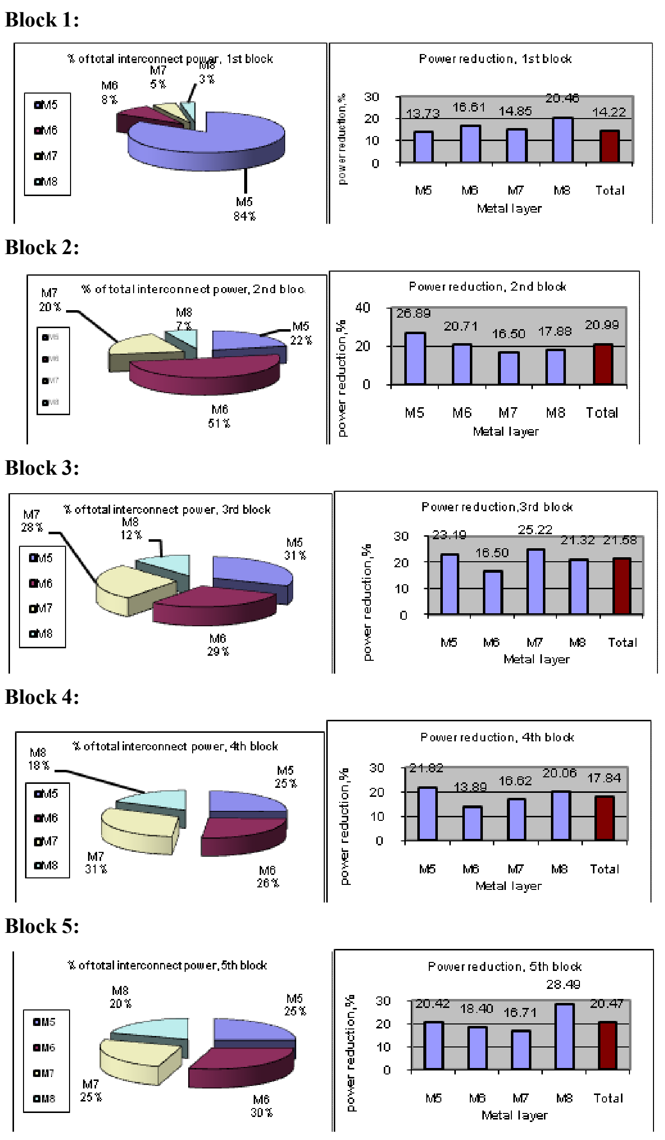

Figure 7: Optimization results for different design blocks

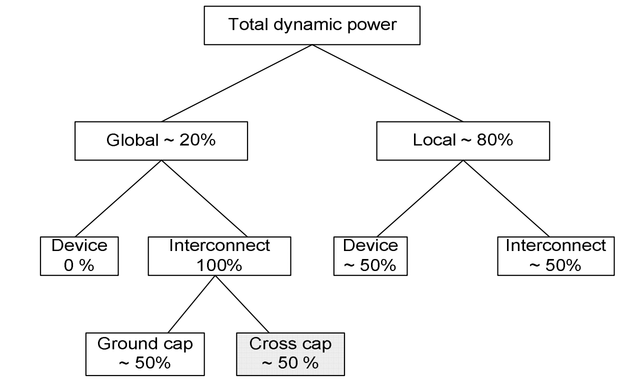

Figure 8: High-end microprocessor dynamic power breakdown