Clock Skew Optimization for Ground Bounce Control

Ashok Vittal, Hien Ha, Forrest Brewer and Malgorzata Marek-Sadowska

Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106

Abstract - High speed synchronous digital systems require large switching currents to facilitate rapid signal transitions. These large currents create voltage drops on the power distribution network and necessitate expensive chip packaging with a large number of supply pins. In this paper we propose a novel technique to reduce the dynamic transient current drawn from the supply pins. Our approach is based on sub-dividing the synchronous clocking into multiple sub-clocks with relative skew. This spreads the computation across the entire clock cycle instead of largely occurring at the beginning. Timing constraints must also be obeyed, so that no races or timing errors are introduced. We propose an exact algorithm based on integer linear programming to solve this problem. We have used our method in the design of a 5GHz ECL encoder chip to achieve a factor of two reduction in ground bounce, as shown by HSPICE simulations. We also obtained order-of-magnitude improvements in ground bounce on benchmarks laid out in submicron CMOS technology. The approach potentially leads to significant reductions in packaging costs.

I. Introduction*

Simultaneous switching noise in a digital system erodes noise margins of receivers, slows down drivers and adds noise to quiet lines, potentially causing logic errors [2], [9]. These problems are exacerbated by large gate counts (which cause large switched currents), higher operating frequencies (signal rise times decrease) and scaled supply voltages (leading to smaller noise margins). Power supply level fluctuations are particularly acute at the beginning of the clock cycle, when many latches and output drivers are simultaneously activated. High speed systems, typically, solve this problem by increasing the number of chip power supply pins, which raises the packaging costs. The number of supply pins in a high performance microprocessor is quite large, e.g., the number of supply pins in the DEC Alpha microprocessor is 140 [4]. The transition from a simple package with pins on the periphery to an array package is typically necessary when the pin count grows above two hundred, so the cost savings by reducing the required number of pins can be tremendous. In this paper, we propose a design technique to reduce ground bounce. This technique is potentially far cheaper to implement and does not affect circuit speed, chip area or average power dissipation. It is based on clock skew optimization to keep the inrush current within limits.

Clock skew optimization is a well-known technique for timing optimization. By skewing the clock inputs to the various flip-flops, it is possible for paths with longer delay to borrow time from shorter paths. A linear programming formulation of the skew assignment problem was proposed in [5]. The timing constraints for the skew between a pair of flip-flops with combinational logic in between are linear. The constraints enforce the absence of races and also leave enough time for the combinational logic to complete the computation. The objective function in that formulation could either be the clock period or the worst case timing margin of the circuit. Ground bounce may be incidentally reduced, but is not addressed by the technique.

Related work on clock skew optimization in [8] considers circuits controlled by k-phase clocks with arbitrary duty cycles. A linear programming formulation for minimum clock period computation is proposed. This technique also introduces the potential difficulty of generating and distributing arbitrary duty-cycle clocks. Graph-based algorithms for clock skew computation have been proposed in [3], [7], [10] and [11]. The ground bounce for these solutions cannot be controlled.

Clock skew optimization to maximize timing margins tends to group input latches into a skew regime, internal flip-flops into a second regime and finally output latches into a third class. This often helps the ground bounce because not all clocked elements are switching simultaneously. However, we can do much better by truly spreading computation over time. In effect, our method takes a synchronous design and transforms it to make it appear asynchronous as seen from the input supply pins. Essentially, our method retains the main advantage of synchronous design - the simplicity due to the global notion of time, and simultaneously achieves one of the advantages of asynchronous design - the absence of large simultaneous switching noise.

The method is practical because it uses a single clock distribution network to distribute multi-phase clocks of the same duty cycle to all the clocked elements. This is in contrast to multi-phase non-overlapping clocks which require the routing of multiple clock nets with multiple clock sources. Our method uses a degree of freedom afforded by new clock routers - the ability to meet specified skews on placed clocked elements.Finally, skew addition for ground bounce reduction was considered by [9], which considers only skewing output drivers (this might lead to timing problems), and [1], which uses skew for a particular regular design. Clearly, neither [1] nor [9] can be used for designs with many irregular timing constraints.

On-chip decoupling capacitors can be added to reduce ground bounce [2]. The charge stored in the on-chip decoupling capacitor is used to support the switching transient. However, there are limits to the decoupling capacitor which can be achieved. Clearly, reducing the current requirement is still useful. On-chip decoupling also reduces the resonant frequency of the power distribution network as seen from the switching elements. If the resonant frequency becomes equal to the clock frequency, the system can ring, i.e., large oscillations may build up on the power supply lines. This is a particularly insidious failure mode, as the best fix is to reduce

*. This work was supported in part by the Defence Advanced Research Projects Agency under contract DABT63-93-C-0039, the National Science Foundation under Grant MIP 9419119 and in part by LSI Logic & Silicon Valley Research through the California MICRO program.

ICCAD '96
1063-6757/96 $5.00 © 1996 IEEE

the effective pin inductance - a degree of freedom which is frozen once the package has been decided. With lossy on-chip capacitors the Q of the resonance can be decreased, but the magnitude of the excitation at this frequency should still be limited. Thus, our technique is an optimization method which complements on-chip decoupling and reduces the requirements on the decoupling network.

The outline of the paper is as follows: section II describes a simple example to demonstrate our method. Section III proposes our ground bounce minimization roblem formulation and discusses several issues related to our formulation. Section IV provides an example from a reasonably large scale design implemented in GaAs heterojunction bipolar ECL. Section V presents results for sub-micron CMOS, obtained using commercially available design automation tools and Section VI concludes.

II. A simple example

Our method is best illustrated by a simple example: an n-bit shift register, shown in Figure 1. The zero skew design has master latches of all n flip-flops switching at the same time when the positive clock edge arrives, and similarly all slave latches switching on the negative clock edge as shown in Figure 2. The worst case ground bounce is n times that due to one flip-flop. Assuming negligible register delay, setup and hold-times, there is no safety margin against races.

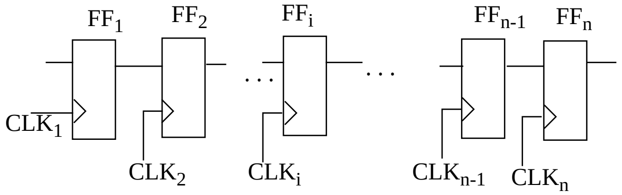

Figure 1. An n-bit shift register

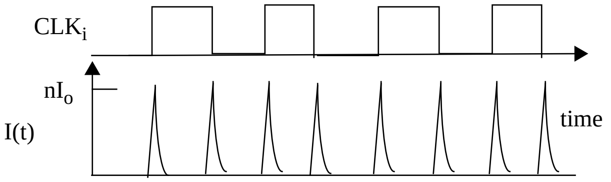

Figure 2. Zero skew design for the shift register

The skewed design for maximum safety margin in timing distributes the same clock to all the even-numbered flip-flops and the opposite phase to the odd-numbered flip-flops, as shown in Figure 3. The clock period remains the same. The worst case ground bounce is as bad as before. The safety margin against races is improved dramatically to half the clock period. Typically, a safety margin of one-fourth the clock period is considered more than adequate. The skewed design for ground bounce minimization with T/4 safety margin distributes n clock phases, each more than 90 degrees behind phase with the next (but less than 270 degrees behind), as in Figure 4.The worst case ground bounce is reduced by a factor of n, the clock period is the same and the safety margin is one-fourth the clock period. We can get a factor of n improvement in ground bounce in this way.

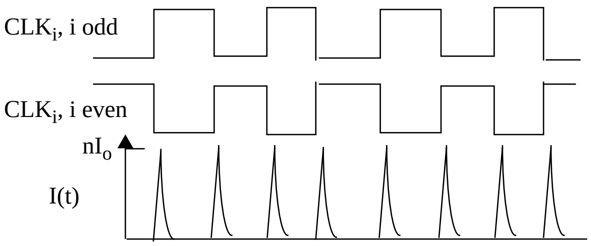

Figure 3. Skew for maximum timing margins

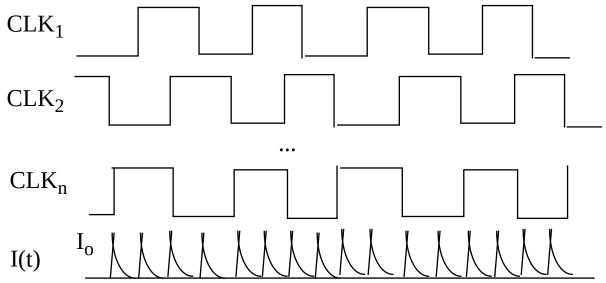

Figure 4. Clocking schemes for ground bounce reduction with safety margin T/4

The above example shows the tradeoff between ground bounce and circuit speed. It also leads to a natural figure of merit for a clock schedule: the clock period - maximum supply current derivative product. This product has the dimensions of current and must be made as small as possible.

III. Problem formulation

3.1 Integer linear programming formulation

We wish to calculate clock skews to each of m flip-flops to satisfy timing constraints, while minimizing the ground bounce. We divide the clock period into a small number of time bins, say n. The clock skew for every flip-flop is an integer value and belongs to [0, n-1]. This discretization reflects the fact that fine grain control is not possible when there are process variations. We represent the clock skew of flip-flop i by n binary variables (S_{i0},..., S_{in-1}), where S_{ij} is 1 if and only if the clock skew of flip-flop i is equal to j. Clearly

$$\left(\sum_{j=0}^{n-1} S_{ij} = 1\right), i = 0, \ldots, m-1 \qquad (1)$$

The circuit graph leads to linear, double-sided timing constraints, as in [5]. Thus, with combinational logic between flip-flop i and flip-flop k characterized by minimum delay $T_{ik\ min}$ and maximum delay $T_{ik\ max}$, we have

$$\left(\sum_{j=0}^{n-1} j \cdot S_{kj}\right) - T_{ik,\,min} \le \left(\sum_{j=0}^{n-1} j \cdot S_{ij}\right) \le \left(\sum_{j=0}^{n-1} j \cdot S_{kj}\right) + n - T_{ik,\,max}$$

(2)

Each term in parentheses represents the time at which the corresponding flip-flop switches. The inequality on the left guards against a race condition and the inequality on the right leaves enough time for the new data to propagate from flip-flop i to flip-flop j.

Using a topological analysis of the circuit graph (given static timing analysis), we obtain the set of possible switching instants for each of the gates and latches in the circuit. For each gate, these times depend linearly on the skew of the flip-flops in its fan-in cone, as shown below. Let I_{lm} be the maximum possible current derivative for element l at time t. The total current derivative at time t is then

$$Id_t = \sum_{\forall l} I_{lt} \qquad (3)$$

We wish to minimize the maximum of all Id s. This is achieved using the constraints

$$Id_t \le I_{max}, \forall t \qquad (4)$$

The objective is

$$Minimize\left(I_{max}\right) \qquad (5)$$

The number of binary variables used in the formulation is O(m*n). There are O(e) constraints, where e is the number of edges in the circuit graph (a graph with a vertex for each flip-flop and an edge between 2 vertices if the corresponding flip-flops communicate directly through combinational logic).

3.2 Input pattern and timing dependence

The ground bounce clearly depends on the applied input pattern and the current state of the circuit. Therefore, the problem of determining the input sequence which maximizes the ground bounce, for a given clock schedule, is of interest. For instance, in the sequential ripple carry adder example of [5], an input pair which causes the input carry to ripple all the way to the output carry would maximize the number of elements switching in the clock cycle. In an n-bit shift register, 0101... is the worst case input pattern. While the worst case input pattern is not dependent on the clock skew schedule for these two specific examples, in general it could also exhibit clock skew dependence.

The times at which gates switch on chip also depend on the times at which the chip inputs change. For designs where inputs and outputs are latched, our formulation can also determine the clocking for the input and output latches. If the inputs are not latched, clock skew cannot mitigate those ground bounce problems caused by the fanout cones of switching inputs.

3.3 Design flow

Our technique is used after placement and global routing, but prior to detailed routing. Delay estimates are obtained following global routing for use in the integer linear program. Following clock schedule optimization, the clock router must guarantee the required delays. We have modified the router in [12] for this purpose, in our design flow.

3.4 Lining up input and output latches

The input latches typically have to operate with clocks of the same phase, so that the data need be available over a reasonably large time window. If the optimal clock phase assignment to the input latches spreads the input latch timing over the entire clock cycle, the time window over which the data needs to be available becomes very small and process/temperature variations may result in system malfunction. This can be avoided by adding delay into the path of data inputs to late clocks, so that the time window when data should be ready is big enough. If the layout area or power dissipation of the buffers added as delay elements is a concern, we can constrain all input latches to operate with clocks of identical phases using the constraints

$$\left(\sum_{j=0}^{n-1} j \cdot S_{ij} = \sum_{j=0}^{n-1} j \cdot S_{kj}\right), \forall inputs\,(i, k) \qquad (6)$$

3.5 Allowing for safety margins in timing

Timing margin improvement can be achieved using the same technique as in [5] - an additional slack variable (M) is added into each of the timing constraints and a constraint is added to make this slack at least as large as a required constant (C).

$$M - T_{ik,\,min} + \sum_{j=0}^{n-1} j \cdot S_{kj} \le \sum_{j=0}^{n-1} j \cdot S_{ij} \le n - M - T_{ik,\,max} + \sum_{j=0}^{n-1} j \cdot S_{kj} \qquad (7)$$

$$M \ge C \qquad (8)$$

3.6 Bounded rotation

The skews introduced for ground bounce minimization could be quite large. This leads to some difficulty during manual verification - it becomes unclear on which edge the data should be ready. Further, very large skews, while improving ground bounce and maintaining timing margins at the target frequency, may lead to designs failing at high clock rates even though the devices could go faster. This is the case when there are combinational paths running from the output of a latch to the input of a flip-flop. Our optimization is for a fixed clock frequency. Alternatively, if the circuit is run at a higher frequency, skew might force failure. This can be avoided to some extent by bounding the skew spread. This is easily achieved by adding extra constraints to the integer linear program.

3.7 Interaction with the clock router

The output of the clock skew optimization is the input to the clock routing phase. The clocked elements should, ideally, be clustered so that elements with close skew phase are also placed close by and can be driven by the same buffer. In a hierarchical design system, a node is input a single phase clock and might distribute different clock phases to its children. The procedure of building the clock tree bottom up works quite well [12].

3.8 Sensitivity to process variations

Statistical process variations cause uncertainty in the exact time at which a clock edge arrives. These variations might lead to designs which are not robust - delay variations could increase the simultaneous switching noise. This is implicitly handled in our formulation by introducing coarse granularity in the times at which clock edges arrive. The arrival time is an interval. Processes which are not mature would call for a smaller number of time bins (n). On the other hand, a larger number of time bins allows the integer linear program to find better solutions.

3.9 Specifying an upper limit on the number of clock regimes

The clock router may impose restrictions on the total number of clock regimes. It is possible to include extra constraints and variables to achieve a specific limit on number of clock regimes. This is done using n binary variables R_0, R_1, ..., R_i, ..., R_{n-1} where R_i is 1 if and only if the i^{th} time bin has been chosen for some clock regime. If R_{max} is the

$$\left(m \cdot R_j \ge \sum_{i=0}^{m-1} S_{ij} \right), j = 0, \ldots, n-1 \qquad (9)$$

required maximum number of clock regimes, we specify

$$\sum_{j=0}^{n-1} R_i \le R_{max} \qquad (10)$$

IV. A 5GHz encoder example

Our method is particularly useful in the design of high speed systems where ground bounce is a significant problem and timing margins are important. We, therefore, discuss the results obtained using our approach for the design of a 5GHz encoder chip for use in a 40Gbit/s ATM network. In order to obtain such high speeds, Rockwell's baseline GaAs-AlGaAs heterojunction bipolar ECL technology is used. The number of transistors in the chip is about three thousand. The design is highly pipelined and consists of nine modules. The chip design is further described in [6].

The peak-to-peak current swing with zero skew was 42mA. The SPICE simulation results are shown in Figure 5a.The simulated circuit includes layout parasitics extracted using the commercially available parasitic extraction tool LPE, from Cadence. If the ground bounce calculated using these current requirements is to be below 30mV (one tenth the logic swing), 20 power/ground pin pairs are necessary. Each of the modules were simulated with their loads and a separate power supply feeding them with test vectors to obtain the zero skew characteristic response. This is done after placement and extraction of layout parasitics for the interconnections within sub-modules and for global wiring.

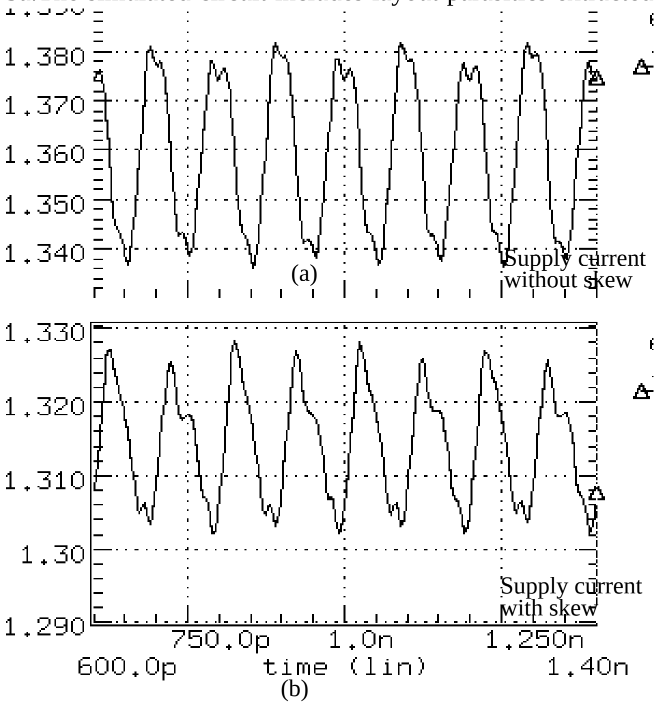

Figure 5. Supply currents a) Zero skew design b) With clock skew

The floorplan leaves a small area in the center of the chip for the global clock distribution network. The current for the design with clock skew is the convolution of the clock input and the characteristic response for the linear time-invariant system. Using these current waveforms and timing constraints extracted from the circuit configuration and functionality, the linear program was formulated and solved. A clock distribution network which realized these skews was synthesized and the entire system was simulated with skew. SPICE simulations showed reduction in current swing to 24mA, as shown in Figure 5b. The design could now be packaged.

Note that the dependence on input patterns is not large for both the skewed and non-skewed designs - the variation during different clock cycles is less than 10%. The circuit actually has three power supplies and the current waveform for only the supply with the largest swing is shown. The integer linear programming formulation is general enough to handle multiple power supplies also.

V. Sub-micron CMOS results

In this section, we present results for implementations in sub-micron CMOS using the Cascade Epoch design automation tools for placement, routing and extraction. We show ground bounce results for several circuits - a sequential ripple carry adder circuit, the ISCAS 89 benchmark S27 and a 16-bit shift register. We also study timing safety margin - ground bounce tradeoffs.

We used Cascade Epoch for automatic placement, routing and extraction with a 0.7 micron CMOS standard cell library. TACTIC, a static timing analysis tool, is used to obtain timing constraints. We extract SPICE models for the design and change the clock network, to obtain a design with smaller ground bounce.

Table 1 shows the SPICE results for benchmark circuits implemented in 0.7 micron CMOS. Among our test circuits are a shift register, where the number of timing constraints grows linearly with the number of flip-flops and a sequential ripple carry adder, which exhibits quadratic growth. S27 is included to represent irregular timing constraints. Note that the ground bounce for the shift register skewed for timing is less than half that without skew; this is due to the capacitance of the non-switching elements helping out by sharing charge.

Table 1: 0.7 micron CMOS ground bounce SPICE results

Benchmark	Clock period (ns)	Ground bounce (mV/nH)		
		Zero skew	Skewed for timing	Skewed with ILP
S27	10	8.4	8.4	4
SRCA8	10	18	15	8
Shift16	10	17	7	2.8

The input sequences for these (small) circuits were chosen to exercise the state transition which maximized the number of input & flip-flops switching. The time required for solving the integer linear program is a few minutes and is less than the SPICE simulation time in all cases.

The ground bounce is minimized by our integer linear program. Clearly, timing safety margin - ground bounce tradeoffs are possible. Figure 6 below shows these tradeoff curves for the ripple carry adder circuit described in [5], implemented in 0.7 micron CMOS. We see that significant gains in ground bounce are possible even for this simple design. Note that each point on the curve represents a valid clock skew schedule. The curve is discontinuous for the adder with input and accumulation registers (sequential ripple carry adder) because there are no valid designs for large timing margins. It is possible to obtain a 10% safety margin for a 10% differential(54% to 64%) in normalized ground bounce. The shift register exhibits very loose timing constraints - it is possible to obtain a factor of nT/δt improvement, where T is the clock period and δt is the granularity. This directly implies timing granularity - ground bounce tradeoffs. Coarser granularity is required for processes which are not stable yet and finer granularity leads to improved results. From the integer linear program, we see that if three adjacent time bins have the same Id, then a worst case delay variation equal to the time bin size could line up the three contributions, increasing the ground bounce by at most a factor of three. In practice, a choice of time bin size equal to about a tenth of the clock period should be sufficient to obtain ground bounce improvements, while maintaining the robustness of the clock skew schedule.

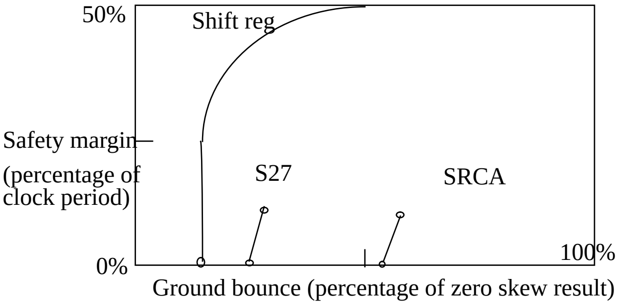

Figure 6. Safety margin - ground bounce tradeoff curve

VI. Discussion and conclusions

The number of supply pins is already quite large for high performance microprocessors today. We believe that our work is the first CAD technique which directly addresses the problem of reducing this requirement. We also believe that our work will highlight this important problem and will motivate other methods aimed at alleviating the problems of supplying power to high performance electronic systems.

Our integer linear programming formulation is primarily intended for system level design, where there are a few tens of modules. It can, clearly, also be used to adjust skews on all flip-flops of a design. This would need heuristics which give sub-optimal results, but have better run times. As such, branch-and-bound programs for integer linear programming are capable of returning intermediate solutions when stopped before a complete search of the decision tree. However, heuristics based on other intuition might still be helpful. It is easy to see that the optimal solution without timing constraints can be solved using a quadratic time complexity dynamic programming algorithm. A modification to handle timing constraints using dynamic programming is possible. Graph-based approaches also seem attractive.

Introduction of skew into synchronous designs increases the spectral content of the required current at higher frequencies. For instance, in the n-bit shift register example the current requirement was reduced in amplitude by a factor of two, but the signal frequency was multiplied by two. Clearly, the design of the on-chip decoupling capacitor network should take this into account.

Our ground bounce minimization formulation is valid for a particular clock period. While it is possible to explore tradeoffs by running the integer linear program solver for various clock period values, the problem of implicitly handling clock period minimization is open. The difficulty with handling variable clock periods is that the characteristic current waveforms of the modules change with clock period and it is not easy to characterize this change without some notion of the circuit functionality of the module. We believe that our work will also motivate such research.

Acknowledgments

We would like to thank Dr. Jose Luis Neves and Professor E.G. Friedman of the University of Rochester for providing us their clock skew optimization code. We would also like to acknowledge the use of Dr. Michel Berkelaar’s integer linear programming package, lp_solve. Finally, we would also like to thank Steve Beccue, a consultant for Rockwell, and Professor Stephen I. Long of UCSB for many useful simulation tips.

References

[1] R. Amerson, R. Carter, W. Culbertson, P. Kuekes, G. Snider, “Plasma: an FPGA for million gate systems”, Proceedings of the International Symposium on FPGAs, pp. 10-16, 1996.

[2] H.B. Bakoglu, *Circuits, Interconnections and Packaging for VLSI*, Addison-Wesley, 1990.

[3] R.B. Deokar and S. Sapatnekar, “A graph-theoretic approach to clock skew optimization”, Proceedings of the International Conference on Circuits and Systems, pp. 407-410, 1994.

[4] D.W. Dobberpuhl et al., “A 200MHz, 64-bit, dual-issue CMOS microprocessor”, IEEE Journal of Solid-State Circuits, Vol. 27, No. 11, pp. 1555-1566, 1992.

[5] J.P. Fishburn, “Clock skew optimization”, IEEE Transactions on Computers, Vol. 39, No. 7, pp. 945-951, 1990.

[6] H. Ha and F. Brewer, “Implementation of a 40Gbit/s fiber channel encoder/decoder”, Custom Integrated Circuits Conference, 1996.

[7] J.L. Neves and E.G. Friedman, “Design methodology for synthesizing clock distribution networks exploiting non-zero localized clock skew”, IEEE Transactions on VLSI Systems, to appear, 1996.

[8] K.A. Sakallah, T.N. Mudge, O.A. Olukotun, “CheckTc and minTc: timing verification and optimal clocking of synchronous digital circuits”, Proceedings of DAC, pp. 111-117, 1990.

[9] R. Senthinathan and J.L. Prince, *Simultaneous switching noise of CMOS devices and systems*, Kluwer Academic Publishers, 1994.

[10] N. Shenoy and R.K. Brayton, “Graph algorithms for clock schedule optimization”, Digest of Technical Papers of the ICCAD, pp. 132-136, 1992.

[11] T.G. Szymanski, “Computing optimal clock schedules”, Proceedings of the DAC, pp. 399-404, 1992.

[12] A. Vittal and M. Marek-Sadowska, “Power-optimal buffered clock tree design”, Proceedings of DAC, pp. 497-502, 1995.