Power Optimal Buffered Clock Tree Design

Ashok Vittal and Malgorzata Marek-Sadowska

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

Abstract

We propose a new problem formulation for low power clock network design that takes rise time constraints imposed by the design into account. We evaluate the utility of inserting buffers into the clock route for satisfying rise time constraints and for minimizing the area of the clock net. In particular, we show that the classical H-tree is sub-optimal in terms of both area and power dissipation when buffers may be inserted into the tree. We show that the power minimization problem is NP-hard and propose a greedy heuristic for power-optimal clock network design that utilizes the opportunities provided by buffer insertion. Our algorithm inserts buffers and designs the topology simultaneously. The results we obtain on benchmarks are significantly better than previous approaches in terms of power dissipation, wire length, rise times and buffer area. Power dissipation is typically reduced by a factor of two, rise times are four times better and buffer area requirements are an order of magnitude smaller.

I. Introduction*

Clock routing is an important problem in the layout design of synchronous digital systems as it significantly influences the area, speed and power dissipation of the synthesized system. The design of the clock distribution network in a synchronous digital system determines the clock skew, thus directly affecting the maximum attainable clock frequency. It determines the rise times of the signals at the clocked elements. This also limits the maximum frequency of operation [1]. The clock net is usually one of the first nets to be routed and constitutes a blockage for nets routed subsequently. As it is one of the largest nets, the area occupied is also a concern. The clock net accounts for a significant fraction of the system power dissipation as it switches most frequently and is a large net.

The method of means and medians [15] was an algorithm proposed for clock net synthesis. The clock net was designed using purely geometric considerations. This could lead to large skews. Subsequent research has aimed to get zero skew under a better delay model [23], better algorithms for the problem [9], [17], minimizing the wire length [2], [3], path-delay optimization [5], [6] and better time complexity [10]. Reliability issues were addressed in [21] and a buffer redistribution algorithm was proposed in [4]. Planar clock net design was considered in [24]. None of these algorithms considered the power dissipated by the clock network. Besides, rise time constraints imposed by design specifications were not considered. While [25] does try to minimize the clock power dissipation, it is meant solely for MCMs. A bounded skew clock tree is synthesized and power reduction is achieved by reducing the wire length; this is not power optimal as the power dissipated by the buffers is not considered. Buffered clock tree synthesis was considered in [6]. However, buffers were inserted as a post-processing step after topology design and the power was not optimized. We take a look at each of these inadequacies.

Low power design has gained importance due to the increasing popularity of portable applications, reliability concerns associated with high operating temperatures (most failure mechanisms are accelerated at high temperatures [1]) and the costs of system cooling. The power consumed by a synchronous system is mainly due to the clock circuitry, the core functional logic and the output drivers. The clock power is typically one third of the total power dissipation in CMOS VLSI systems [20]. We address clock circuitry power dissipation in this paper. We provide a problem formulation for low power clock network design that is general enough to include many technologies like CMOS VLSI and bipolar ECL. We prove that the problem is NP-hard. However, a simple greedy heuristic returns reasonable solutions. The power consumed by our clock circuitry is typically a factor of two smaller than greedily designed topologies with buffers only at the root. This decrease in power dissipation can result in a much larger decrease in cost if it enables a cheaper packaging option. The threshold-like behavior of cooling costs [14] increases the significance of our results.

Buffers are inserted into clock trees as a post-processing step and this is typically done only at the root. Aggressive designs should use all the available degrees of freedom to obtain better performance. During clock routing, however, current design methods do not insert buffers at internal nodes of the clock tree. We show that *simultaneous topology design and buffer insertion* can do significantly better in terms of area, rise times and power dissipation. The skew remains zero under the Elmore delay model. The H-tree [13] which has been the inspiration for a lot of work on clock routing [15], [17] is demonstrated to be *sub-optimal* in terms of area and power. Besides our results on benchmarks enable the clock routes to be used at frequencies that are higher than any other known methods. This is due to the lower limit on the clock period imposed by the rise times of the clock signals, which is decreased by buffering at internal nodes.

In Section II we take a look at the rationale behind buffer insertion. We discuss some examples to demonstrate the use of buffers. Section III formulates the low power design problem. We show that the problem is NP-hard by reducing the minimum Steiner tree problem to it. The reduction gives a straightforward algorithm to modify any tree into a zero skew tree by inserting buffers. A greedy heuristic for low power clock distribution is presented in Section IV. The algorithm simultaneously designs the topology and inserts buffers. Section V discusses the results and concludes.

*. This work was supported in part by the Advanced Research Projects Agency under contract DABT63-93-C-0039, NSF Grant MIP 9117328 and by AT&T Bell Labs and Xilinx through the California MICRO program.

32nd ACM/IEEE Design Automation Conference ®

II. Buffer Insertion

Buffers are inserted in clock nets to ensure that the rise times of the signals at the clock entry points are not too large. Large rise times are undesirable as they lead to clock pulse width narrowing [1], i.e., the ON time of the clock pulse is reduced. Besides, logic threshold voltage variations of the clocked elements combined with large rise times introduces timing noise which means that the frequency of operation is reduced. Thus, rise times are fundamental design constraints and are more important than delay. However, the rise times are strongly correlated to the delays (the two are linearly related in our model, which we outline later). Previous approaches have, therefore, optimized the delays [5], [6]. When circuit techniques other than buffering are used, the delay may be irrelevant but the rise times will continue to be important.

In [1] device characteristic variations over a large WSI substrate were used to justify a single buffer strategy. A mature process, however, may guarantee reproducible buffer delays. Devices on a chip tend to be closely matched though their absolute characteristics may vary over a process window. Inserting buffers at internal nodes in the clock tree may not be detrimental to system performance, especially if other gains accrue. In this section we show that buffers can be used to significantly *reduce the wiring costs*.

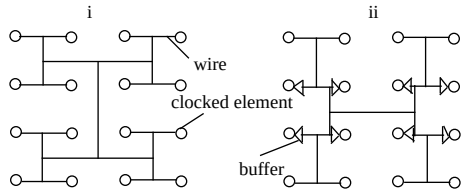

Fig. 1: Wire length optimization possible using buffers. i) The H-tree uses 18 wire length units. ii) A minimum Steiner tree with inserted buffers uses 15 units of wire length. The buffers are used as variable delay elements to make the delays to the sinks equal. Buffer sizes are different.

Consider the regular 16-point clock routing instance shown in Fig. 1. For such an instance the H-tree [13] is popular. This incurs a wiring cost of 18 units. By inserting buffers as shown, the wiring cost can be reduced to 15. The buffer sizes are chosen so that zero skew is maintained. Asymptotically, it is easy to show that the H-tree for a regular grid of clock entry points requires $1.5D\sqrt{N}$ wire length where D is the length of the die and N is the number of clocked elements. On the other hand, the minimum rectilinear Steiner tree needs only $D\sqrt{N}$ wire length. Thus, 50% savings in wire length result by just inserting buffers intelligently, while maintaining zero skew.

A natural question at this point is: How much wire length savings are possible for irregular pin distributions? The bar graph in Fig. 2 answers this question. We compare

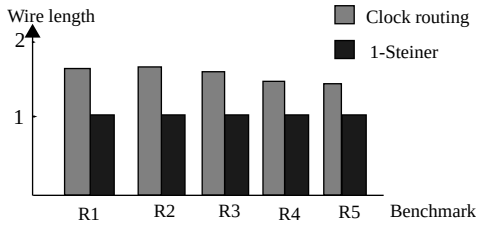

Fig. 2: Wire length savings possible on clock routing benchmarks

the best previous clock routing results with a minimum Steiner tree heuristic [18]. We see that even for irregular pin distributions the wire length savings possible range from 40% to 60%. The results are normalized with respect to the 1-Steiner [18] results.

The examples above give us *existence proofs* of solutions that are better, when buffers are inserted in the tree. Delay optimal buffering for clock trees given an initial zero skew route using dynamic programming has been proposed in [6]. For the H-tree example, however, we can never obtain the wire length optimal solution by inserting buffers after running a clock routing algorithm. Similarly, inserting buffers into a minimum Steiner tree may result in unacceptable buffer sizes. The algorithm should design the topology and insert buffers simultaneously, if we are to get good results. This is done in Sections III and IV.

The opportunities provided by buffer insertion for reducing the wire length can be expressed in terms of the *deferred merge embedding* technique [11], [3], [2]. This method builds a topology bottom-up, starting from a set of clock points and successively merging till a zero skew clock tree results. During each merge of sub-trees to form a larger sub-tree, the locus of possible Elmore zero-skew merge points on the Manhattan plane is determined. The delays from the sinks to the root of the new sub-tree must be equal (for zero skew) and the wire length used must be as small as possible (equal to the distance between the two sub-tree roots). The set of possible zero skew merge points under these constraints is a line segment.

Consider the possibilities if we can add a buffer into either branch of the tree when merging sub-trees T_1 and T_2 shown in Fig. 3. The position of the roots on the Manhattan

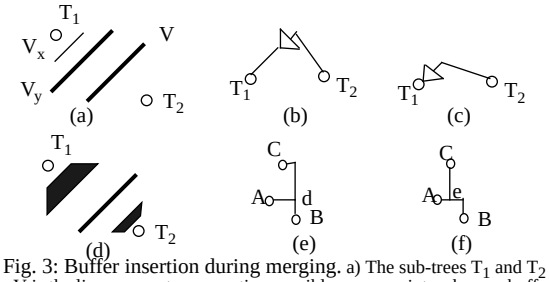

Fig. 3: Buffer insertion during merging. a) The sub-trees T_1 and T_2 are merged. V is the line segment representing possible merge points when no buffer is inserted. V_x and V_y are the line segments for configurations in (b) and (c). (d) represents the final set of possible merge points. (e): clock route when no buffers are inserted. (f): The clock route when a buffer is inserted just before A (smaller wire length)

plane is shown in (a) and (d) as circles. When no buffer is added, as in classical deferred merge embedding, the locus of zero-skew merge points is the line segment V. With a fixed size buffer at the start of the left sub-tree as shown in (b), we get the line V_x as the set of possible merge points. The zero skew merge point is closer to T_1 as the buffer has added delay in the left branch. With a buffer just before T_1, as shown in (c), we get the merging segment V_y. We get similar line segments when a buffer is inserted into the right sub-tree. The complete set of possible merge points is shown in (d). The regions correspond to buffers being added at points in between the start and end of the two sub-trees. (e) and (f) show how the clock route is improved by this freedom. When clock points A and B are merged without buffers, the closest merge point to C is d. With buffers, the closest merge point to C is e. Wire length savings have resulted.

We now explain how the positions of the line segments V_x and V_y can be calculated. Let the capacitances of the sub-trees T_1 and T_2 be C_1 and C_2 and the corresponding delays be D_1 and D_2. Let the distances of the zero-skew merge point from the roots of T_1 and T_2 be L_1 and L_2 respectively. For zero skew, the delay from any sink in the new sub-tree to the root must be equal, after merging. Consider configuration (b). The equivalent circuit is shown in Fig. 4. The driver

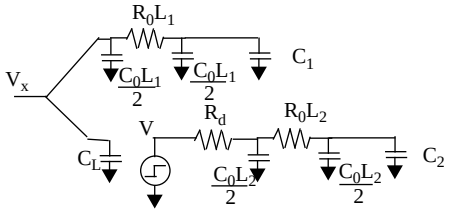

Fig. 4: Equivalent circuit for finding zero skew merge point.

resistance is R_d, the buffer input capacitance is C_L, the resistance per unit length of interconnect is R_0 and the capacitance per unit length of interconnect is C_0. The zero skew equations are

$$D = D_1 + R_0 L_1 (C_1 + \frac{C_0 L_1}{2}) \qquad (1)$$

$$D = D_2 + R_0 L_2 (C_2 + \frac{C_0 L_2}{2}) + R_d (C_2 + C_0 L_2) \qquad (2)$$

where D is the delay to any sink from the root of the new tree.

As we do not want any detour wiring, the merge point should be within the rectangle having the two sub-tree roots as corners (the "bounding box"). We therefore have

$$L = L_1 + L_2 \qquad (3)$$

The new tree capacitance is

$$C = C_L + C_2 + C_0 L_2 \qquad (4)$$

In these equations, L_1, L_2, D and C are the unknowns. Their values are easily determined from (1) through (4).

Solving similar equations, we get the other delimiting segment V_y. Sliding the position of the buffer along the sub-tree moves the merge point from one delimiting segment to the other. As the merge point has to be within the bounding box, the zero skew merge region with left sub-tree buffering is a polygon as shown in Fig. 3d. The other polygon in Fig. 3d corresponds to right sub-tree buffering. The two polygons may overlap and may also include the segment V.

The locus in Fig. 3d is for a buffer of given size. A different buffer size gives a similar region of possible merge segments with different delimiting segments. Similarly, wire sizing also changes the positions of the two polygons. We now formulate the power minimization problem for clock nets.

III. Problem Formulation

For calculation of the delays, rise times and power dissipation, we replace a buffer by the equivalent circuit shown in Fig. 5 and wire segments by the equivalent circuit shown in Fig. 6. As before, R_d & C_L are the buffer driver resistance & input capacitance, L is the interconnect length and R_0 & C_0 are the interconnect resistance & capacitance per unit length. A delay element is added in series with the driver resistance to model the internal delay of the buffer. The internal delay of buffers is typically tens of picoseconds. The error in using the Elmore delay model could easily be comparable to this.

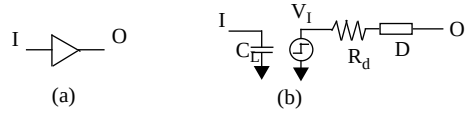

Fig. 5. Buffer modeling: a) Buffer b) Equivalent circuit

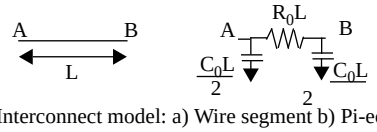

Fig. 6. Interconnect model: a) Wire segment b) Pi-equivalent

The delay is now defined to be the Elmore delay [12], [22], a model that makes the synthesis problem tractable without sacrificing too much accuracy. The expression for the Elmore delay is

$$D_i = \sum_{k \in P_i} R_k C_k$$

where D_i is the delay to the i^{th} sink, P_i is the set of resistances on the unique path from the root to the i^{th} sink, R_k is the resistance of any resistor on the path and C_k is the downstream resistance seen by R_k. The Elmore delay essentially makes a *dominant time constant approximation* to the circuit step response at any sink. Thus, if we make the same approximation to calculate the rise times, we get the 10%-to-90% rise time to be

$$T_i = 2.2 \cdot D_i$$

where R_i is the rise time of the signal at the i^{th} sink. For inputs with finite rise times, the time constants of sink responses are affected equally. Besides, the rise times at the sinks of a zero skew clock tree are equal.

The power dissipated by a clock net can be attributed to the charging and discharging of the wiring and load capacitances through the interconnect & driver resistance and to the static power dissipated, if any, by the buffers.

$$P = P_{static} + P_{dynamic}$$
$$= P_0 \cdot \sum_{buffers} k_i + f \cdot C_0 \cdot V^2 \cdot L(T)$$

Here P_0 is the sum of the static and dynamic power dissipated by a minimum size buffer, k_i is the size of the i^{th} buffer, *f* is the frequency of operation, *V* is the voltage swing, C_0 is the capacitance per unit length and L(T) is the tree length. Wire widening increases the power dissipation and we do not consider it. The techniques of [21] which widen wires for reliability as a post-processing step can be used if necessary.

The clock network design determines the buffer sizes, their locations and the interconnect topology. It therefore affects the summation in the first term and the wire length in the second term. For CMOS VLSI, the static power consumed by the buffers is negligible, so that the problem reduces to one of minimizing the total capacitance - wiring and buffer capacitances. In ECL, the static power dissipated by the buffers dominates. For multi-chip modules, both dynamic and static power consumption may be equally important. Guided by these considerations, we formulate the problem as one of finding the interconnect topology, buffer

locations and sizes to

$$Minimize\left(L(T) + \alpha\left(\sum_{i=1}^{B} K_i\right)\right) \qquad (5)$$

where α is a technology-dependent constant (given), L(T) is the wire length of the synthesized buffered tree T, B is the number of inserted buffers (determined by the design, not given) and K_i is the size of the i^{th} buffer. Note that buffer sizing is allowed by our formulation as K_i and B are not pre-specified.

The skew constraints are

$$(\forall (i,j))\, D_i = D_j \qquad (6)$$

where D_i and D_j are the Elmore delays to the i^{th} and j^{th} sinks, as before.

The rise time constraints are

$$(\forall i)\ (T_i \le T_{max}) \qquad (7)$$

where T_{max} is a specified maximum allowed rise time and T_i is the rise time of the signal at the i^{th} sink. For our model the rise time constraints also impose delay constraints as the delay is proportional to the rise time. The rise time constraint is extremely important as any recognizable clock must have a clock period of at least thrice the 10%-to-90% rise time. The rise time of classically designed clock nets imposes a limit on the frequency of operation, even if logic delays are small.

We call the problem (5) with (6) and (7) the *constrained power optimal buffered clock network design* (CPOBCND) problem. We will show that the decision version of this problem is NP-complete by reducing the minimum Steiner tree problem to it. The problem is in NP as finding the delays of a given tree can be done in linear time[22]. We need the following lemma to prove the reduction.

Lemma1: Any tree can be transformed into a zero skew tree by inserting buffers into the tree.

Proof outline: Our proof is constructive and is given by the algorithm in Fig. 7. It involves a linear time traversal that

```
Algorithm Equalize_Delays
Input: A non-zero skew tree T
Output: A zero skew buffered tree T*
 begin
   if isnt_a_leaf(T)
     begin
       Equalize_Delays (left_sub_tree(T))
       Equalize_Delays (right_sub_tree(T))
       if(not_zero_skew(T))
         Insert_Buffer (T)
     end
 end
```

Fig. 7. Getting zero skew using buffers

inserts buffers in the tree to equalize delays. The correctness of the algorithm can be proved by induction. The procedure Insert_Buffer involves choosing the correct size buffer at the root of the smaller delay sub-tree using the equations (1) through (4) in Section II. For arbitrary delays and capacitances at the two sub-trees, a driver resistance can always be found to satisfy the zero skew equations (1) through (4). The buffer size changes the value of R_d. Any required driver resistance can be synthesized by varying the buffer size. All we need to do is to solve the zero skew equations for the value of R_d and find the appropriate size.

We now reduce the minimum Steiner tree problem to the CPOBCND problem.

[Rectilinear Minimum Steiner tree problem]
Instance: Point set P, number L.
Question: Is there a Steiner tree whose length is less than L?

Given an arbitrary instance (P,L) of the rectilinear minimum Steiner tree problem, we construct the CPOBCND problem instance (point set P, $\alpha = 0$, $R_{max} = \infty$). The two instances are equivalent. The correctness of the reduction follows from the following theorem.

Theorem 1: P has a Steiner tree of length less than L if and only if the CPOBCND problem has a solution of cost less than L.

Proof: If P has a Steiner tree of length less than L, the same topology with buffers inserted using EQUALIZE_DELAYS has zero skew. The cost is L as α=0. The rise time constraints are trivially satisfied. This proves that if the Steiner tree problem has a solution then our CPOBCND instance also has a solution.

If the CPOBCND instance has a solution, there is a solution whose cost is less than L. If we remove the inserted buffers, we are left with a Steiner tree whose length is less than L. □

Theorem 1 means that it is unlikely that there is a polynomial time exact algorithm for the CPOBCND problem. In the next section, we propose a heuristic for our problem. We will see that the results obtained are encouraging.

IV. A Heuristic

The greedy clustering algorithm of [9] returns unbuffered clock trees with the smallest wire length of the algorithms compared in [5]. It is therefore reasonable to expect a greedy algorithm to work well for our problem as well. Some important changes have to be made, however, as we have rise time constraints and the possibility of adding buffers. We first look at the new issues that arise.

- The greedy algorithm involves choosing the move that is locally optimal. During the zero skew merge operations we have to choose the merge that results in minimum cost increase. The cost is defined in the problem formulation.
- When two sub-trees are merged, the locus of possible merge points which do not involve detour routing is a line segment, when buffers are not allowed. As shown in Section II, with buffers we have to store two polygons and a line segment as the set of possible merge points. When a merge point corresponding to buffering is chosen, a buffer is being added for wire length savings.
- After having merged two sub-trees, we have the possibility of inserting a buffer for future rise time savings. This move should be made only if the rise time constraint is very stringent, else we end up adding too many buffers. Besides, the move should be made only if there is going to be rise time savings due to the move. This move is therefore made only if the normalized cost increase entailed by the buffer addition is less than the normalized rise time constraint saving and if rise time savings result.
- The rise time constraint should guide the topology design as well, especially if it is difficult to satisfy. When merging sub-trees, the optimum cost increase may involve a topology decision that makes the rise time constraint difficult to satisfy. We have the necessary rise time information during the

merge. We therefore adaptively modify the cost to reflect the rise time constraint: the penalty function approach.

These key ideas lead to the algorithm Grin (GReedy INternal buffering). The time complexity of the algorithm is $O(n^3)$. This is not too expensive as the design has to be verified using an accurate simulator that solves a system of O(n) equations and takes about the same amount of time. While process variation sensitivity has not been explicitly considered as in [21], note that the approach in [21] typically changes only a few of the top levels in the tree, so that a variant of our algorithm which considers reliability during the last two or three moves would suffice.

Algorithm Grin

Input: Point set P, α, technology parameters, max rise time T_{max}
Output: Buffered Elmore zero-skew power-optimal clock tree T
begin
  repeat |P| -1 times
    begin
      minimum_cost_merge();
      if buffering_improves_delay() and cost_decreases()
        add_buffer();
    end
  Tapered_buffers_at_root()
end
minimum_cost_merge()
find a pair such that the cost is minimized

$$Cost = \Delta L + \alpha \cdot \Delta K_n + \frac{\lambda \cdot \Delta T}{T_{max} - T}$$

where K is the buffer size, L is the length and T is the rise time.
For the two nodes get the polygons representing possible merge points

V. Results and Conclusions

We implemented our algorithm in C on a SPARC 10. We ran four sets of experiments. The first set of experiments compared the results of our approach with buffer insertion at internal nodes (Grin) with greedy clustering and buffer insertion only at the root (Root) using the techniques of [19], [16]. This experiment verifies the use of internal buffering on the de facto benchmarks from [23] for three CMOS technologies. The second set of experiments tests if simultaneous topology design and buffer insertion is a viable approach for rise time improvement. We compare our delays with that of the *optimal* buffered tree synthesis of [6]. The other experiments show how the use of our method for bipolar ECL instances and explore buffer area-wire length trade-offs.

In order to demonstrate the improvement possible using buffering, we compare our algorithm with buffers at internal nodes to one with buffers only at the root. The topologies are designed by the same greedy algorithm. We compare the performance of the two algorithms on the clock routing benchmarks r1-r4 [23] for 1, 0.8 and 0.5 micron CMOS technologies. The results are shown in Tables 1,2 and 3. We have used the classical tapered drivers with exponentially increasing sizes [19],[16],[1] at the root of both trees. Buffers at internal nodes in the tree are of a single size only. The buffer area is expressed with this buffer area as the unit. The number of buffers inserted by Grin grows only linearly with the number of clock pins. The tree with buffers only at the root has to drive a large capacitance and the buffer sizes required are tremendous. By spreading the buffers over the entire tree, especially in the branches close to the root, much smaller buffer area is needed.

The power is calculated for a frequency of 100 MHz. This frequency of operation cannot be achieved by the unbuffered clock tree for some instances as the rise time is too large. We see that the power and rise times are considerably smaller for Grin. The results get better as technology scales down. This is due to the fact that the buffer RC time constant to interconnect RC time constant ratio decreases as technology scales down. This means that using buffers becomes a more attractive proposition. As the die size increases as well, we need to pay more attention to the larger examples for the finer feature size technologies.

Table 1: 1 micron CMOS technology results

Bench mark	Power(mW)		Rise time(ns)		Buffer area		# buffers	
	Grin	Root	Grin	Root	Grin	Root	Grin	Root
R1	13.6	35.5	0.466	1.31	34	235	6	6
R2	29.1	44.9	0.916	2.01	88	235	7	6
R3	35.1	94.7	0.716	3.29	90	638	9	7
R4	60.6	120.2	1.51	6.02	91	638	10	7

Table 2: 0.8 micron CMOS results

Bench mark	Power(mW)		Rise time(ns)		Buffer area		# buffers	
	Grin	Root	Grin	Root	Grin	Root	Grin	Root
R1	11.4	35.5	0.44	1.28	15	235	6	6
R2	29.3	44.9	0.73	1.98	90	235	9	6
R3	35.1	94.7	0.91	3.26	90	638	9	7
R4	60.6	120	1.49	5.76	91	638	10	7

Table 3: 0.5 micron CMOS results

Bench mark	Power mW)		Rise time (ns)		Buffer area		# buffers	
	Grin	Root	Grin	Root	Grin	Root	Grin	Root
R1	11.3	33.0	0.42	1.24	15	235	6	6
R2	22.9	42.3	1.03	1.93	35	235	7	6
R3	33.7	88.7	0.38	3.21	88	638	7	7
R4	59.2	113	1.24	5.44	90	638	9	7
R5	97.8	244	12.3	19.3	239	1735	10	8

We note the implications of these results for the clock net design of a microprocessor such as the DEC Alpha [8]. The tapered buffers at the root occupy significant die area and consumes a quarter of the total power dissipated by the chip. If we extrapolate our results, the buffer area required by our scheme will be only about a tenth. The cost of a microprocessor is proportional to the cube of the die area and our clock net would result in considerable cost savings. The overall power savings will be about 15%.

Table 4 shows the delay reduction possible using simultaneous topology design and buffer insertion. The OBTS results are from [6] and we use the same technology parameters. We weight the delay term in our cost heavily, so that the comparison is not biased against Grin. OBTS is an optimal algorithm which starts from a zero skew route and inserts

buffers from a library; we insert buffers of only one size. The OBTS delay results also include wire sizing optimizations. Our delay results are better though we do not use two dimensions available to OBTS - wire sizing and buffer sizing. The buffer area and power dissipation of the OBTS clock nets were not available[7] and could not be compared.

Table 4: Delay reduction possible by simultaneous topology design and buffer insertion

Benchmark	OBTS delay(ns)	GRIN delay (ns)
R1	0.481	0.333
R2	0.667	0.609
R3	0.685	0.630
R4	1.042	0.869
R5	1.442	1.112

The parameter α in Grin allows us to vary the "cost" of buffers. This enables us to explore buffer area - wire length trade-offs, which is useful in technologies with scarce routing resources. Fig. 8 shows the trade-off for R1-5 in 0.5 μm CMOS technology. About 10% of wire length can be saved with a 3x increase in buffer area typically. Our buffer areas are still less than that used by Root. To put these results in perspective, note that many wire length minimization techniques in clock routing have typically reported around 10% improvement in wire length using better algorithms.

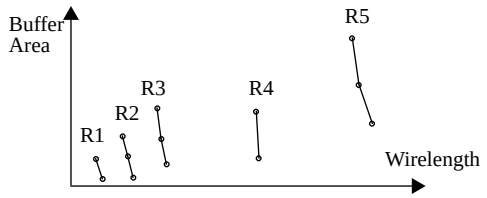

Fig. 8. Buffer area - wirelength trade-offs for 0.5μm CMOS

Our formulation is general enough to handle technologies other than CMOS as well. We used Grin for clock routing in a bipolar ECL technology. We compare Grin with Root, as in our first set of experiments. The root in this case has only one buffer (user-specified size) in both cases. The results are shown in Table 5 below. The rise times for Root are too large to allow its use in any practical system. As bipolar integration levels are smaller than CMOS, we have used only the smallest two benchmarks for comparison.

Table 5: Clock routing results for bipolar ECL

Bench mark	Rise time (ns)		Power (mW)		Buffer area		# buffers	
	Grin	Root	Grin	Root	Grin	Root	Grin	Root
R1	2.23	11.2	65.9	-	6	-	6	-
R2	0.94	32.1	112	-	10	-	10	-

In conclusion, our unified approach to clock routing is attractive in comparison to previous approaches. Our method is general and can be used for many technologies. We address many important design concerns such as rise time constraints and power dissipation. We have shown that inserting buffers into a clock route can be useful. We effectively explore a new design space that was missed by previous methods. The results show the power of our method.

Acknowledgments

We thank Dr. Ren-Song Tsay of Arcsys, Inc. for the benchmarks and Dr. Gabriel Robins and Ms. T. Zhang of the University of Virginia, Charlottesville for their 1-Steiner code.

References

[1] H.B. Bakoglu, "*Circuits, Interconnections and Packaging for VLSI*", Addison-Wesley, 1990.
[2] K.D. Boese and A.B. Kahng, "Zero skew clock net routing with minimum wire length", Proceeding of the IEEE International Conference on ASIC, 1992, pp. 1.1.1 - 1.1.5
[3] T.-H. Chao, Y.-C. Hsu and J.-M. Ho, "Zero skew clock net routing", Proceedings of the Design Automation Conference, 1992, pp. 518-523.
[4] J.-D Cho and M. Sarrafzadeh, "A buffer redistribution algorithm for high speed packaging", Proceedings of the Design Automation Conference, 1993, pp. 537-542.
[5] N.-C. Chou and C.-K. Cheng, "Wire length and delay minimization in general clock net routing", Digest of Technical Papers of the IEEE International Conference on Computer Aided Design, 1993, pp. 552-553
[6] J. Chung and C.-K. Cheng, "Optimal buffered clock tree synthesis", Proceedings of the IEEE ASIC Conference, 1994, to appear.
[7] J. Chung, private communication, August 1994.
[8] D.W. Dobberpuhl et al., "A 200 MHz, 64-bit, dual-issue CMOS microprocessor", IEEE Journal of Solid State Circuits, Vol. 27, No. 11, November 1992, pp.1555-1566.
[9] M. Edahiro, "A clustering based optimization algorithm in zero skew routings", Proceeding of the Design Automation Conference, 1993, pp. 612-616.
[10] M. Edahiro, "An efficient zero skew routing algorithm", Proceeding of the Design Automation Conference, 1994, pp. 375-380.
[11] M. Edahiro, "Minimum skew and minimum path length routing in VLSI layout design", NEC Research and Development 32(4), October 1991, pp. 569-575.
[12] W.C. Elmore, "The transient response of damped linear networks with particular regard to wide-band amplifiers", Journal of Applied Physics, Vol. 19, No.1, 1948, pp. 55-63.
[13] A.L. Fisher and H.T. Kung, "Synchronizing large VLSI processor arrays", IEEE Transactions on Computers, Vol. c-34, pp. 734-740, August 1985.
[14] L. A. Glasser and D. W. Dobberpuhl, "*The Design and Analysis of VLSI Circuits*", Addison-Wesley 1985.
[15] M.A.B. Jackson, A. Srinivasan and E.S. Kuh, "Clock routing for high performance ICs", Proceedings of the Design Automation Conference, 1990, pp. 573-579.
[16] R.C. Jaeger, "Comments on 'An optimized output stage for MOS integrated circuits'", IEEE Journal of Solid State Circuits, Vol. 10, June 1975, pp. 185-186.
[17] A.B. Kahng, J. Cong and G. Robins, "High performance clock routing based on recursive geometric matching", Proceeding of the Design Automation Conference, 1991, pp. 322-327
[18] A.B. Kahng and G. Robins, "A new class of Steiner tree heuristics with good performance: the iterated 1-Steiner approach", Digest of Technical Papers of the IEEE International Conference on Computer Aided Design, 1990, pp. 428-431.
[19] H.C. Lin and L.W. Linholm, "An optimized output stage for MOS integrated circuits", IEEE Journal of Solid State Circuits, Vol. 10, No. 2, April 1975, pp. 106-109.
[20] D. Liu and C. Svensson, "Power consumption estimation in CMOS VLSI circuits", IEEE Journal of Solid State Circuits, Vol. 29, No. 6, June 1994, pp. 663-670.
[21] S. Pullela, N. Menezes and L.T. Pillage, "Reliable non-zero skew clock trees using wire width optimization", Proceedings of the Design Automation Conference, 1993, pp. 165-170.
[22] J. Rubinstein, P.Penfield and M.A. Horowitz, "Signal delay in RC tree networks", IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, Vol. 2, No.3, 1983, pp. 202-210.
[23] R.-S. Tsay, "An exact zero skew clock routing algorithm", IEEE Transactions on CAD, Vol. 12, 1993, pp. 336-339.
[24] Q. Zhu and W. W.-M. Dai, "Perfect balance planar clock net routing with minimal path length", Digest of Technical Papers of the IEEE International Conference on Computer Aided Design, 1992, pp. 473-476.
[25] Q. Zhu, J. G. Xi, W. W.-M. Dai and R. Shukla, "Low power clock distribution based on area pad interconnect for multichip modules", Proceedings of the International Workshop on Low Power Design, 1994.