VLSI Design
1995, Vol. 3, No. 1, pp. 81–92
Reprints available directly from the publisher
Photocopying permitted by license only

A Comparative Study of Synchronous Clocking Schemes for VLSI Based Systems*

AHMED EL-AMAWY and UMASANKAR MAHESHWAR
Department of Electrical and Computer Engineering
Louisiana State University, Baton Rouge, LA 70803

(Received January 6, 1993, Revised September 22, 1993)

Recently a novel clock distribution scheme called Branch-and-Combine(BaC) has been proposed. The scheme guarantees constant skew bound irrespective of the size of the clocked network. It utilizes simple nodes to process clock signals such that clock paths are adaptively selected to guarantee constant skew bound. The paper uses a VLSI model to compare the properties of the new scheme to those of the well established H-Tree approach. The H-Tree is a binary tree of simple buffers which is laid out such that leaves are at equal distances from the root.

Our study considers clocking 2-D processor meshes of arbitrary sizes. We evaluate and compare the relevant parameters of both schemes in a VLSI layout context. We utilize parameters such as *clock skew*, *link costs*, *node costs* and *area efficiency* as the basis for comparison. We show that for each BaC network, there is a certain threshold size after which it outperforms the corresponding tree network in terms of skew. We also show that except for node costs, BaC networks outperform the H-Tree, especially when the size of the clocked network is large. As an extension we show that BaC clocking does not suffer from potential pulse disappearance, no matter how large the network is.

Key Words: *Clock skew; VLSI; Layout; H-Tree; Branch-and-Combine; Link costs; Node Costs; Area utilization*

1. INTRODUCTION

In any digital system, a control mechanism to guarantee correct sequencing of events is needed. A majority of systems use a single clock source to connect the sequence of events with time. This *synchronous* scheme is the most widely used due to its simplicity and relative low cost. In some systems, an asynchronous scheme is used. In such cases, different processors, or more generally, data processing elements operate under control of unrelated clocks. Communications between different processors (or elements) must follow a strict handshaking protocol to guarantee correctness. Asynchronous schemes are more difficult and expensive to implement, are potentially slower, and could suffer from metastable failures. In this study, we will only consider synchronous schemes and confine our attention to VLSI implementation issues.

In synchronous control schemes, a single clock source is typically employed to provide a continuous stream of clocking events spaced equally in time (voltage transitions, pulses, ... etc.). Ideally, a clocking event should reach and affect all data processing units (processors or elements) at the same time. However, due to several factors such as different threshold voltages, different path lengths and buffer delays, the events usually reach the elements at different times. The difference in arrival times is called *clock skew*. Different aspects of the clock skew problem have been studied in [8], [1], [2]–[6], [13]. The problem clock skew poses is that it directly relates to the clocking rate. Larger skew requires slower clock rate for the system to operate correctly [8], [1], [3]. It has been shown that, for a synchronous system to operate properly, the minimum clock period must be extended by the amount of the largest anticipated clock skew in the system [8], [3]. In other

*Supported in part by NSF grant no. MIP 9117206 and in part by NSF/LEQSF under contract 1992-3-ADP-04.

words, if system A which is skew-free can safely operate with a minimum clock period of T, then a similar system B which suffers from skew $\leq \tau$ must use a clock period $\geq T + \tau$. Clearly the problem is more significant in larger systems. Nonetheless, skew can be a problem even on a single chip [13].

Fisher and Kung [8] studied the problem of designing clocking networks for linear arrays and 2-D meshes in a VLSI context. They suggested a binary tree clocking network, called H-Tree, to efficiently clock the arrays. The attractive feature of this clocking network is that every leaf node in the clocking network is equidistant from the source, which is connected to the root. They utilized two models namely the *summation model* and the *difference model* to study skew bounds. They showed that in the summation model (which is a weaker model), clock skew grows with the size of the clocking network due to minor fluctuations in buffer delays. The drawback is that when the size of the networks gets larger, clock skew grows without bound, thereby indicating that the clock rate must be slowed down substantially.

In a recent development, El-Amawy [3]–[5] proposed a new clock distribution scheme called Branch-and-Combine clocking. The most interesting feature of the scheme is that clock skew is guaranteed to have a constant upper bound regardless of network size. The Branch-and-Combine (BaC for short) technique relies on the existence of cycles containing finite number of nodes. The nodes perform simple operations on the clock signals to control skew within known bounds. Clock signal paths are adaptively and automatically selected such that each node is triggered via the shortest delay path from the source to the node [6]. If a pair of communicating data processors or elements are clocked from two nodes which are in the same cycle then the skew is guaranteed to be less than the delay through that cycle. Thus employing shorter cycles reduces skew bound at the expense of some extra hardware. This implies the existence of alternatives in BaC network design.

In this paper we utilize a VLSI model to evaluate and compare H-Tree and BaC clocking networks. For simplicity we only consider clocking a 2-D mesh of data cells or data processors. We evaluate the target networks based on a set of parameters such as skew, link costs, node costs and VLSI area. In Section 2 we review and state certain basic results for H-Tree and BaC clocking networks. Section 3 contains a comparative analysis of both clocking schemes. In Section 4 we discuss the problem of potential pulse disappearance and show that BaC networks are void from this problem. Section 5 contains our conclusion.

2. BASICS OF THE TWO SCHEMES

2.1 Tree clocking networks

The underlying graph of a tree clocking network is a binary tree, an example of which is shown in Figure 1 for a size of 7. The nodes in the tree are simple buffers interconnected in a binary tree fashion while the clock source is connected to the root.

In [8], Fisher and Kung introduced a model (called the summation model) in which delay variations from link to link are not ignored even if they are of the same length. They used the model to analyze a tree clocking network and proved that clock skew is $\Omega(N)$, where N is the size (number of nodes) of the tree. In [11] Kugelmass and Steiglitz developed two probabilistic models in which the propagation delay on every source to processor path is the sum of independent contributions which are identically distributed. A metric free model predicts skew upper bound to grow as $\Theta(\log N)$ where N is the number of clocked processors. They have also used a metric model assuming H-Tree layout to conclude that skew grows as $\Theta(N^{1/4}(\log N)^{1/2})$. In [2] a metric free probabilistic model is used to study clocking long linear arrays from a tree. The study shows that the clock period must be increased at the rate $\log N$ to compensate for skew, with low failure probability. In a VLSI layout context, metric-free models would not be applicable, however.

Some investigators studied the problem of embedding trees into VLSI arrays [9], [15] and achieved 100% utilizations. These approaches emphasized reducing the interconnection distances between parents and their children in the tree [9], [15] with no

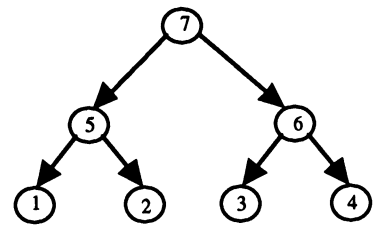

FIGURE 1 A binary tree of size 7.

concern for the locations of nodes in the same level or the distance between them. In the present context however the leaf nodes are assumed to clock the data processors which are arranged in a 2-D mesh. Consequently the leaves have to be embedded in the form of a 2-D mesh and their distances to the root must be as equal as possible. The H-Tree layout [8] was considered the most appropriate for VLSI since it satisfies the above requirements.

Here we introduce a *metric* model suitable for modeling both network types for VLSI. The layout shown in Figure 2 is that of an H-Tree with 64 leaves clocking a (square) 2-D mesh with 64 processors. All nodes in the H-tree layout are identical containing a single buffer each. The leaves of the tree are marked with circles to indicate that they directly clock the processors in the data network. The index within each leaf node indicates the processor clocked by that node. The intermediate nodes are marked by black spots in the figure, whereas grid points which have not been used in the layout are marked with x's. It can be seen from Figure 2 that the link length between a parent and its child is not constant as implied in the *metric-free* models. That explains why a metric free model could not be applied in the present context. We now model the

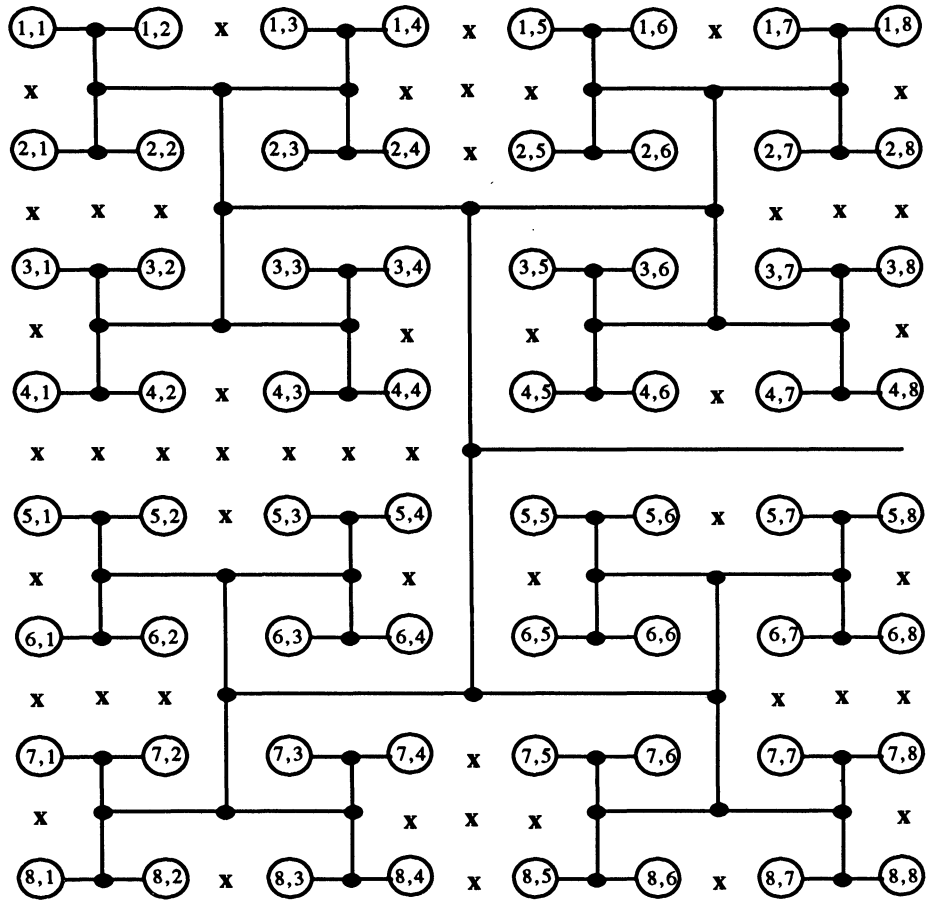

FIGURE 2 Tree clocking network for a 2D mesh of size 64 using H-Tree layout.

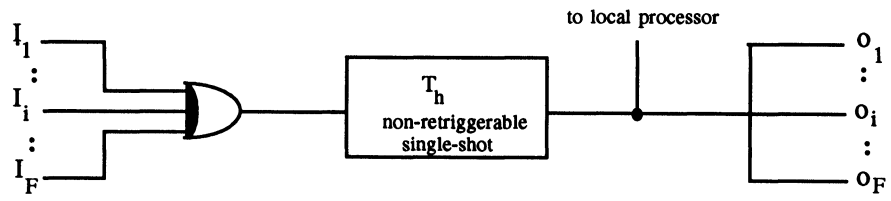

FIGURE 3 A simple node design.

network using the following metric model:

- Delay in a link between two adjacent grid points = λ (grid points may be adjacent vertically or horizontally)
- Delay in a buffer = λ (we consider the link delay to be equal to the delay in a single buffer)
- $\lambda = x \pm a$ where a is the factor due to process parameter variations
- Delay in a link is directly proportional to the length of the link. We assume the delay over a unit length is λ.

Using the above model we can estimate the clock skew for the H-Tree.

Lemma 1: *Under the above metric model, a tree clocking network using H-Tree layout, will have clock skew of* $O(\sqrt{N})$.

Proof: It can be seen from the layout in Figure 2 that the link lengths between a parent and its children doubles every two levels as we traverse up the tree from the leaves. For the sake of convenience and without loss of generality, we assume that the size of the 2-D mesh = $mn = N$ is a power of 2. That is $N = 2^k$, where k is a positive integer. Therefore the size of the tree is $2N - 1$ and the number of leaves is N. A tree with N leaves has $\log N + 1$ levels and the length of the path from the root to any leaf is $\log N$. Hence if $\log N$ is even, then the delay through this path will be given by the expression

$$\left(1 + 1 + 2 + 2 + \cdots + 2^{\log N/2} + 2^{\log N/2}\right)(\lambda) + (\log N)(\lambda) \tag{1}$$

This expression reduces to [14]

$$\left(4\sqrt{N} - 2 + \log N\right)(\lambda) \tag{2}$$

Hence the worst case clock skew is given by

$$\left(4\sqrt{N} - 2 + \log N\right)2a = \left(8\sqrt{N} - 4 + 2\log N\right)a \tag{3}$$

For the case when $\log N$ is odd, it can be easily shown that maximum skew is given as

$$\left(12\sqrt{\frac{N}{2}} - 4 + 2\log N\right)a \tag{4}$$

Hence using H-Tree layout, the clocking network has a clock skew of $O(\sqrt{N})$. □

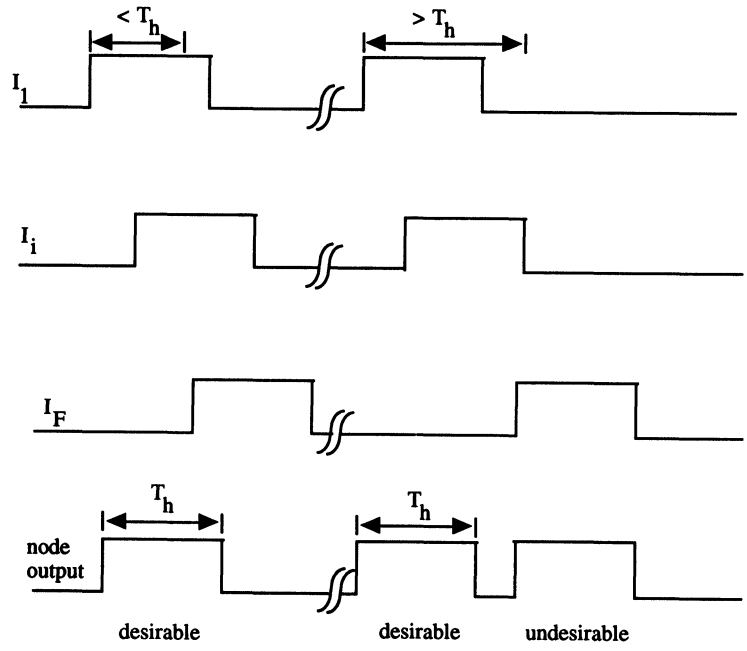

FIGURE 4 Illustrative timing diagrams.

2.2 BaC Networks

Branch and Combine (BaC) clocking has recently been introduced by El-Amawy [3]–[5], the most attractive property of BaC clocking is its ability to guarantee constant skew upper bound between any pair of directly communicating processors, regardless of network size. This is achieved by employing simple clock nodes in the clock distribution network whose main function is to process clock signals such that skew is controlled within known bounds. In the process, clock signal paths are adaptively and automatically selected such that each node is triggered via the shortest delay path from the source to that node. Although each node will generally receive multiple input pulses per clocking event, only the first pulse reaching the node will trigger it. Thus each node is guaranteed to trigger once per clocking event. This ensures the stability of the distribution network despite the existence of cycles [6]. The nodes also are responsible for clocking the data processors or cells (processors hereafter). Although a clock node could clock more than one processor, [3], [4] we assume here that each node is assigned to a unique processor and each processor is assigned a unique node. In this paper we only consider clocking a 2-D mesh of processors.

The principle on which BaC clocking is based is that the graph underlying the clock network is cyclic in nature such that each pair of adjacent nodes must be included in a cycle of finite length $\leq L$. Each node will have more than one input but only the first arriving pulse (on any input) during any particular clocking event will trigger the node. Once triggered the node outputs a pulse and enters a state in which it remains unresponsive (inert) to further inputs for a period $T_h > L\Delta$, where Δ is the delay through a node and any one of its output links, taken over the entire network. This guarantees that the node will be triggered once (by the first arriving pulse) per clocking event and that all subsequent input pulses belonging to the same event will be absorbed or ignored. This is possible to implement since they are known to arrive with T_h units of time from the first

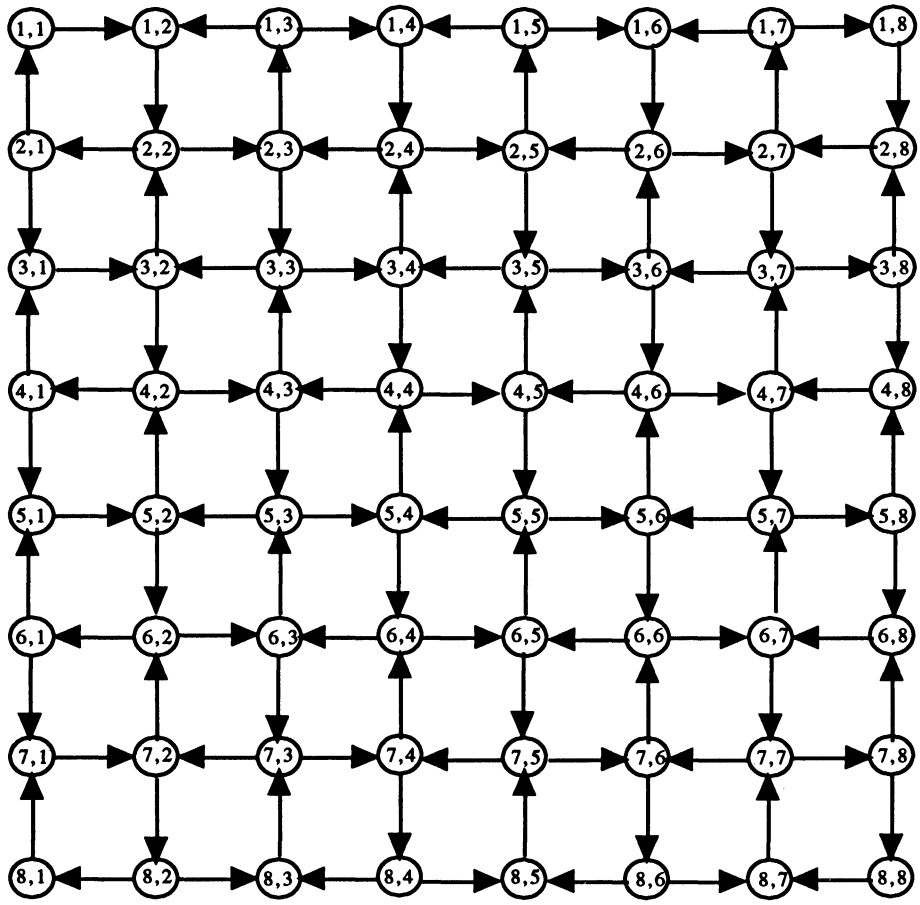

FIGURE 5 An F_2 BaC clocking network for a 2D mesh of size 64.

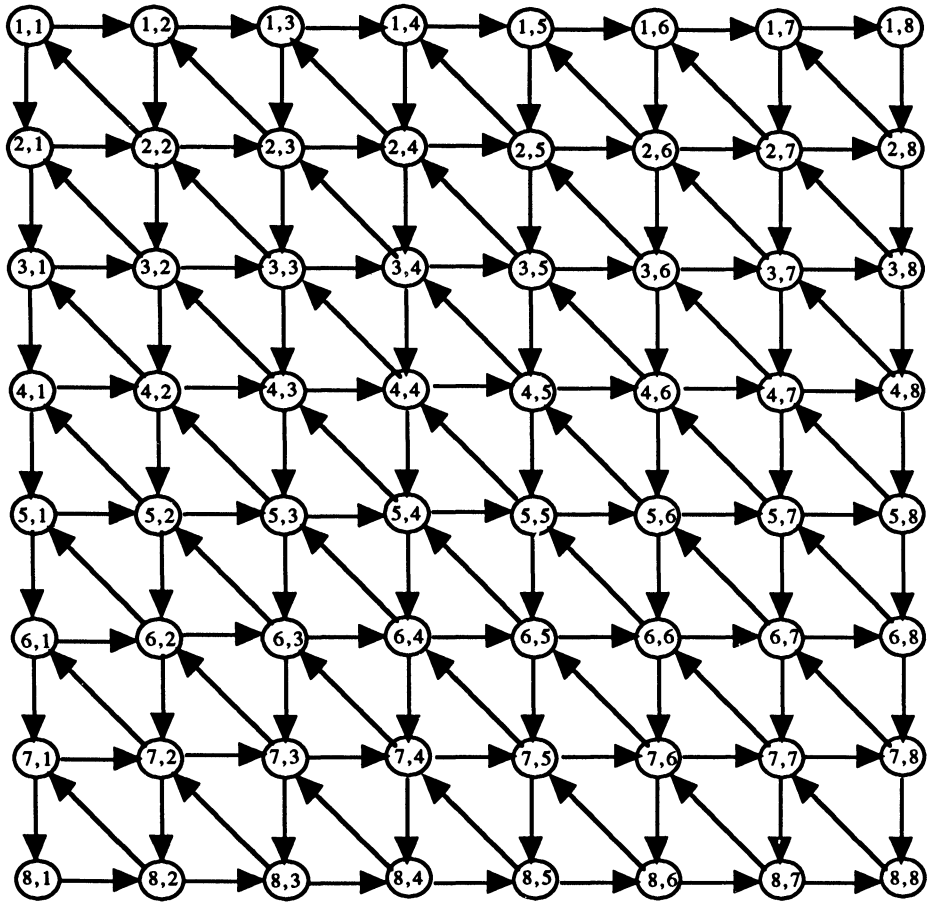

FIGURE 6 An F_3 BaC clocking network for a 2D mesh of size 64.

[6], [3]. The above also guarantees that the skew between outputs of any two nodes in the same cycle is $\leq (L - 1)\Delta$. Figures 3 and 4 illustrate a simple node design and the corresponding timing. It is to be noted that better node designs have been reported in [3]. We use the one in Figure 3 for clearer illustration.

In [3] three different networks for clocking a 2-D mesh under BaC principles with varying cost and performance levels have been described. These networks, called F_2, F_3, and F_4, are shown in Figures 5, 6, and 7, respectively where F_x refers to a network employing nodes with Fan-in = Fan-out = x. Only the F_3 network was analyzed in detail in [3] and shown to be stable under a certain set of node timing constraints. Later, El-Amawy and Kulasinghe [6] developed a general graph theoretic model for BaC networks and derived necessary and sufficient conditions for network stability. They have shown that a BaC(n) network can guarantee a constant skew upper bound of $(n - 1)\Delta$, where Δ is as defined earlier and n is the length of the longest shortest cycle (called *feature cycle length*) containing any pair of adjacent nodes. In [12] and [7], algorithms for systematic synthesis of different BaC(n) networks for hypercubes, meshes and tori have been described and proved correct. Interested readers can find more details on BaC clocking in references [3], and [6].

In this paper we compare the three networks F_2, F_3, and F_4, described in [3] to the H-Tree in the context of VLSI implementation. Any of the networks will be assumed to clock a 2-D mesh of processing cells (processors). It has been shown in [3] that the maximum skew between the outputs of two adjacent nodes is 3Δ, 2Δ and Δ for the F_2, F_3, and F_4, networks, respectively, where Δ is as defined above. For known BaC node designs, node delay amounts to two gate delays [3], [6]. Hence for BaC networks $\Delta = 4\lambda$ since each link is assumed to span a distance of two grid points and node delay is 2λ. Thus for the F_2, F_3, and F_4, networks maximum skew between two adjacent nodes (processors) is 12λ, 8λ and 4λ, respectively.

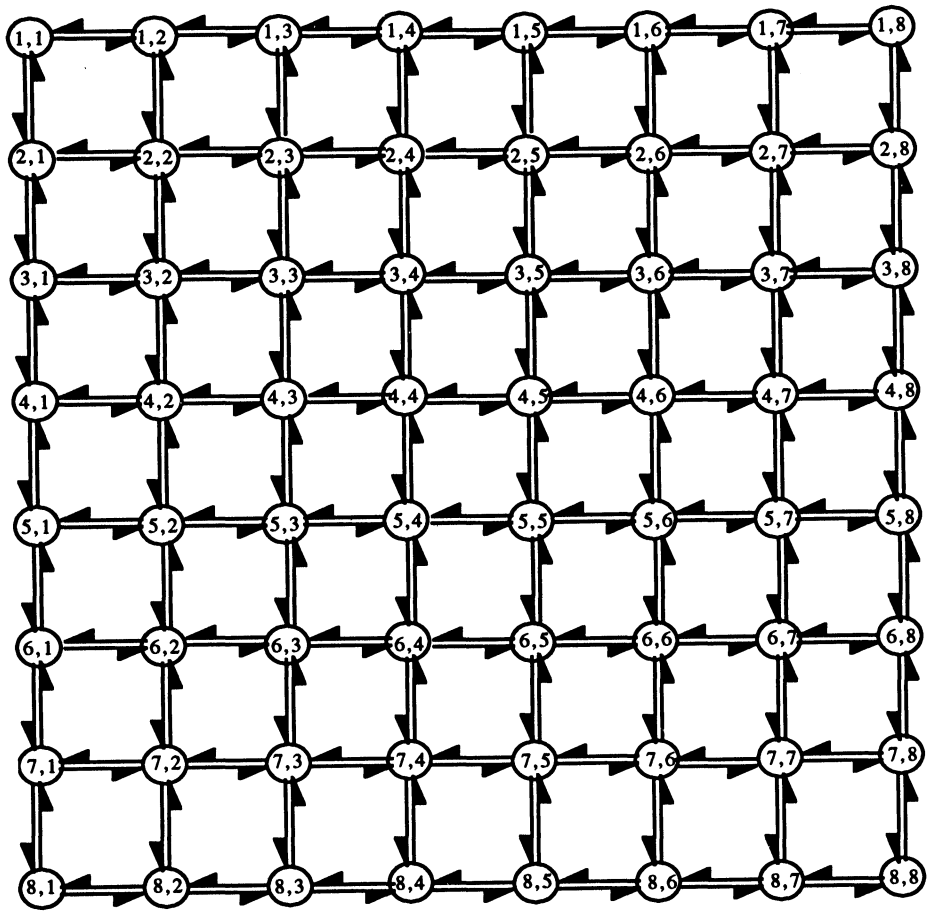

FIGURE 7 An F_4 BaC clocking network for a 2D mesh of size 64.

3. COMPARATIVE ANALYSIS

To judge the merits/demerits of different clocking schemes in a VLSI context, it is necessary to evaluate their characteristic parameters and to work out cost-performance analysis under exactly the same set of conditions. In our study we assume that both clocking networks will clock the same data network. Therefore the locations of the nodes which directly clock the processors are fixed. That is, if we superimpose a BaC network layout on the H-Tree layout, then the nodes of the BaC network will exactly lie on the nodes which directly clock the processors in the H-Tree layout (these are the circled nodes in Figure 2). We will perform simple comparative analysis based on *clock skew*, *link costs*, *nodes costs*, *area efficiency*, *and maximum edge length*. Before we get into the analysis, we define a relationship between the values of x and a where $\lambda = x \pm a$. We introduce a new parameter k, called the *variation ratio*, where $k = x/a$. We will see shortly in the analysis that this factor plays an important role in network skew. Notice that k represents the ratio of delay variation (a) to the nominal delay value (x).

3.1 Clock Skew

We call into focus the equations for clock skew given earlier. For the H-Tree clocking network

clock skew

$$= \begin{cases} \left(8\sqrt{N} - 4 + 2\log N\right)a; & \log N = \text{even} \\ \left(12\sqrt{\dfrac{N}{2}} - 4 + 2\log N\right)a; & \log N = \text{odd} \end{cases}$$

For the three BaC networks we are considering:

max. skew

$$= \begin{cases} 3\Delta = 12\lambda = 12(k+1)a; & \text{for } F_2 \text{ Network} \\ 2\Delta = 8\lambda = 8(k+1)a; & \text{for } F_3 \text{ Network} \\ \Delta = 4\lambda = 4(k+1)a; & \text{for } F_4 \text{ Network} \end{cases}$$

TABLE 1
Max skew for H-Tree Vs. F_2

(Clock skew values are in units of "a")

H-Tree / F_2

$\downarrow k$ \ $N \rightarrow$	1	2	4	8	16	32	64	128	256
1	4	10	14	24	36	54	72	106	140
	24	24	24	24	24	24	24	24	24
2	4	10	14	24	36	54	72	106	140
	36	36	36	36	36	36	36	36	36
3	4	10	14	24	36	54	72	106	140
	48	48	48	48	48	48	48	48	48
4	4	10	14	24	36	54	72	106	140
	60	60	60	60	60	60	60	60	60
5	4	10	14	24	36	54	72	106	140
	72	72	72	72	72	72	72	72	72
6	4	10	14	24	36	54	72	106	140
	84	84	84	84	84	84	84	84	84
7	4	10	14	24	36	54	72	106	140
	96	96	96	96	96	96	96	96	96
8	4	10	14	24	36	54	72	106	140
	108	108	108	108	108	108	108	108	108
9	4	10	14	24	36	54	72	106	140
	120	120	120	120	120	120	120	120	120
10	4	10	14	24	36	54	72	106	140
	132	132	132	132	132	132	132	132	132

TABLE 2
Max skew for H-Tree Vs. F_3

(Clock skew values are in units of "a")

H-Tree / F_3

$\downarrow k$ \ $N \rightarrow$	1	2	4	8	16	32	64	128	256
1	4	10	14	24	36	54	72	106	140
	16	16	16	16	16	16	16	16	16
2	4	10	14	24	36	54	72	106	140
	24	24	24	24	24	24	24	24	24
3	4	10	14	24	36	54	72	106	140
	32	32	32	32	32	32	32	32	32
4	4	10	14	24	36	54	72	106	140
	40	40	40	40	40	40	40	40	40
5	4	10	14	24	36	54	72	106	140
	48	48	48	48	48	48	48	48	48
6	4	10	14	24	36	54	72	106	140
	56	56	56	56	56	56	56	56	56
7	4	10	14	24	36	54	72	106	140
	64	64	64	64	64	64	64	64	64
8	4	10	14	24	36	54	72	106	140
	72	72	72	72	72	72	72	72	72
9	4	10	14	24	36	54	72	106	140
	80	80	80	80	80	80	80	80	80
10	4	10	14	24	36	54	72	106	140
	88	88	88	88	88	88	88	88	88

TABLE 3
Max skew for H-Tree Vs. F_4

(Clock skew values are in units of "a")

H-Tree / F_4

$\downarrow k$ \ $N \rightarrow$	1	2	4	8	16	32	64	128	256
1	4	10	14	24	36	54	72	106	140
	8	8	8	8	8	8	8	8	8
2	4	10	14	24	36	54	72	106	140
	12	12	12	12	12	12	12	12	12
3	4	10	14	24	36	54	72	106	140
	16	16	16	16	16	16	16	16	16
4	4	10	14	24	36	54	72	106	140
	20	20	20	20	20	20	20	20	20
5	4	10	14	24	36	54	72	106	140
	24	24	24	24	24	24	24	24	24
6	4	10	14	24	36	54	72	106	140
	28	28	28	28	28	28	28	28	28
7	4	10	14	24	36	54	72	106	140
	32	32	32	32	32	32	32	32	32
8	4	10	14	24	36	54	72	106	140
	36	36	36	36	36	36	36	36	36
9	4	10	14	24	36	54	72	106	140
	40	40	40	40	40	40	40	40	40
10	4	10	14	24	36	54	72	106	140
	44	44	44	44	44	44	44	44	44

Skew comparisons between these BaC networks and the H-Tree are listed in Tables 1–3. For each table the value of clock skew is calculated for different values of k and N. $k = 1$ implies that delay variation is 100% and $k = 10$ implies that variance is 10%. Actual VLSI implementations limit the delay variance due to process parameter fluctuations. We consider the range $1 \leq k \leq 5$ to represent a valid and complete interval for all practical purposes. However, we list table entries for $1 \leq k \leq 10$ for better illustration.

From the tables we observe that for each mesh size, the skew associated with the H-Tree is constant, whereas skew associated with a certain BaC network increases with k. Conversely for each value of k, the skew for a given BaC network is constant while that of the H-Tree increases with the size.

Our aim now is to identify the threshold level at which clock skew of the H-Tree clocking network becomes consistently greater than that of a BaC network. We call this size the *threshold size* indicated by $N_{cs}(k, f)$ which depends on both k, the variation ratio and f, the fan-out (fan-in) of the specific BaC network.

Let $N_{cs}(k, f)$ be that value of N such that $\forall N \geq N_{cs}(k, f)$, clock skew(tree) $\geq$ clock skew $(F = f)$ and $\forall N < N_{cs}(k, f)$, clock skew(tree) < clock skew $(F = f)$.

We can thus state that $N_{cs}(k, *)$ is that size such that $\forall N \geq N_{cs}(k, *)$, clock skew(tree) > clock skew $(F = *)$. $N_{cs}(k, *)$ is the smallest value of N of which all the three BaC networks have lower or equal clock skew bound compared to that of the tree clocking network. From Tables 1, 2 and 3 we can see that $N_{cs}(10, 2) = 256$, $N_{cs}(10, 3) = 128$, and $N_{cs}(10, 4) = 32$. Note that $N = 256$ corresponds to a

2-D mesh of six 16 × 16 which is a reasonably small size. It can also be observed that $N_{cs}(k, 4) \leq N_{cs}(k, 3) \leq N_{cs}(k, 2)$ and therefore $N_{cs}(k, *) = \max(N_{cs}(k, 2), N_{cs}(k, 3), N_{cs}(k, 4))$. If $k_1 \leq k_2$, then $N_{cs}(k_2, f) \leq N_{cs}(k_1, f)$.

When we examine all threshold sizes, we notice that they are not large and therefore, it is reasonable to assume that to clock medium to large data networks, BaC networks always outperform the H-Tree in terms of clock skew. On the other hand the H-Tree is clearly superior for small networks.

3.2 Link costs

The next important parameter is the cost of links in VLSI layouts. If the underlying technology used in the VLSI implementation is the same for both networks, we can assume that the cost of a link is proportional to its length. Let the cost of one link (one unit of the grid) be l_c. Let log N be even. Then let $l_c \times$ Total.link.length will give the cost of the network. In the F_4 network [3], it can be seen that there are 4 nodes with node degree 4, $4(\sqrt{N} - 2)$ nodes with node degree 6 and the rest have a node degree of 8. Therefore the total number of links $= (4 \times 4 + 4 \times (\sqrt{N} - 2) \times 6 + (N - 4 - (4 \times (\sqrt{N} - 2))) \times 4)/2$. This works out to $4 \times (N - \sqrt{N})$. Therefore, for the F_4 network Total cost $= 4 \times ((N - \sqrt{N}) \times l_c$. Notice that we assume that each link spans two adjacent grid points. For the F_2 network Total cost $= 2 \times (N - \sqrt{N}) \times l_c$. For the F_3 network, Total cost = (Total cost for F_2) + cost of diagonal links. The cost of diagonal links $= l_c \times (\sqrt{N} - 1)^2$. Therefore Total cost $= (3N - 4(\sqrt{N} + 1)l_c$.

In case of H-Tree layout we have already observed that link length between a parent and a child doubles every two levels as we traverse up the tree. But it is also true that the number of links reduces by a factor of 2 every level. Therefore if we write down the values in the form of a series, starting from the leaves and going up level by level the picture looks like

$$\text{Link length} \rightarrow 1\,1\,2\,2\,4\,4\,8\,8\ldots$$

$$\text{No. of links} \rightarrow N\frac{N}{2}\frac{N}{4}\frac{N}{8}\frac{N}{16}\frac{N}{32}\frac{N}{64}\frac{N}{128}\cdots.$$

As the cost of the links is the summation of the cost of links at each level, we get

$$\text{Link costs} = \left(N + \frac{N}{2} + \frac{N}{2} + \frac{N}{4} + \frac{N}{4} + \frac{N}{8} + \frac{N}{8} + \frac{N}{16} \ldots + \sqrt{\frac{N}{2}} + \sqrt{N}\right).$$

The expression reduces to $3(N - \sqrt{N})$ [14].

When log N is odd the 2-D mesh will be represented by $\sqrt{2N} \times \sqrt{N/2}$. The Total cost for the F_4 network works out to $2(2N - \sqrt{N/2} - \sqrt{2N})l_c$. Hence for the F_2 network, Total cost $= (2N - \sqrt{N/2} - \sqrt{2N} + 1) \times l_c$. For the F_3 network, Total costs $(3N - 2\sqrt{N/2} - 2\sqrt{2N}) \times l_c$. For the H-Tree layout, the total cost $= (3N - 4\sqrt{N/2})$. It can be seen from the plots shown in Figure 8 that the function for the Link cost for the tree clocking network grows faster than that for the F_2 network but slower than that for the F_4 network. Link costs for the H-Tree and F_3 networks are about the same and in fact they coincide in Figure 8. It can also be observed that the Link cost for all the networks are of $\Theta(N)$.

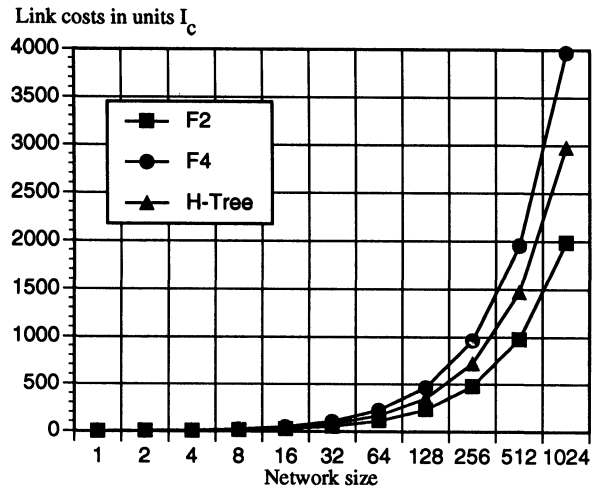

FIGURE 8 H-Tree Vs. BaC networks based on Link cost. (F_3 and H-Tree plots coincide).

3.3 Node costs

In the tree clocking network, each node consists of a single buffer. In BaC networks, each node consists of a few gates and 2-3 flip-flops [3], [6], depending on the specific design. From the structure of the two networks it can be seen that the total number of nodes in a tree clocking network $= 2N - 1$ and

total number of nodes in a BaC network = N. Though the number of nodes in the tree clocking network is double that in a BaC network, the cost of a node in a BaC network offsets this factor. Hence we can say that the total node cost of the tree clocking network will be less than that of BaC networks, perhaps by a constant factor of 10. However, from a practical point of view if we associate a node with each processor, the addition of 20 gates or so to the logic of the processor (which usually consists of thousands of gates) may have little or no cost consequences. This is particularly true in a VLSI context where logic costs are considered far less significant than link (communication) costs.

3.4 Area efficiency

Conventionally, area efficiency of a layout is termed as the ratio of the number of grid points utilized by the topology of the network being embedded into the VLSI grid [15]. Here, the total number of grid points = $(2\sqrt{N} - 1)2 = (4N - 4\sqrt{N} + 1)$. Therefore for the H-Tree clocking network, Area efficiency = $2N - 1/4N - 4\sqrt{N} - 1 \cong 50\%$ when $N \gg$. For the F_2, F_3, and F_4 networks, all the grid points are utilized as either nodes or as connection points. Therefore, for these BaC networks Area efficiency = 1; which means 100% utilization. Clearly BaC networks are much more area efficient compared to the H-Tree clocking network.

3.5 Maximum edge length

In a VLSI layout, it is very important to determine the maximum edge length in the structure. The reason is that, VLSI design is limited by the fact that two pulses cannot physically exist on the same wire at the same time (equipotential clocking) [8]. Due to this limitation, the clock rate of this network becomes dependent on the maximum edge length.

A plot is shown in Figure 9 which illustrates this and compares the maximum clock rate for both the schemes based solely on maximum edge length. For the H-Tree, when network size becomes very large, the maximum clocking rate has to be lowered substantially to ensure correct operation of the network. This makes BaC networks more efficient in that regard since for BaC networks, the maximum edge length is independent of network size.

In [1] the authors state that using a partitioned line instead of a continuous line in the VLSI layout could allow pipelining of multiple clock signals on the same link. This can be achieved by using repeaters within the link. As a result the maximum clock frequency will only depend on the delay between two successive repeaters. But process parameter variations will cause variations in delays associated with these repeaters which may significantly add to clock skew. Therefore, partitioned lines do not completely offset the problem of maximum edge length in tree clocking networks, when the network is large.

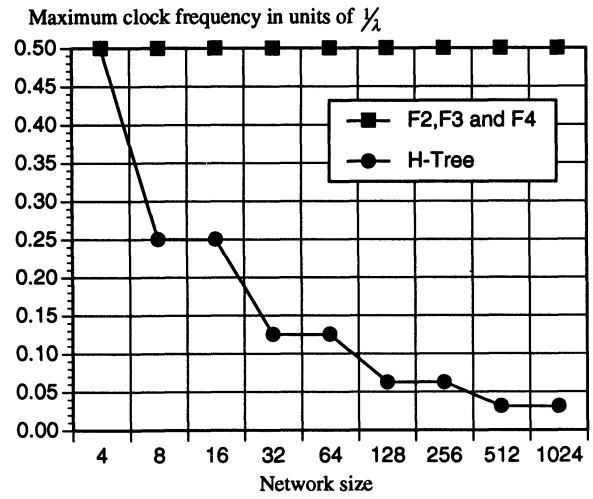

FIGURE 9 H-Tree Vs. BaC networks based on maximum clock frequency.

Remark: In [6] El-Amawy and Kulasinghe proved that no tree of any kind can be used in 2 or higher dimensions to bound skew by a constant. Thus, it might seem to the reader that introduction of cycles into the tree clocking network will enhance its performance by reducing clock skew. We have investigated this scenario and found that it is not possible. The reason for this is that, in the H-Tree layout, the distance between nodes in the same level increases at the rate of $\sqrt{N}$, where N is the size of the tree clocking network. Therefore, introducing cycles between two nodes at the same level will cause clock skew to grow at the rate of $\sqrt{N}$. Clearly this does not improve on earlier results. Hence, introduction of cycles into the tree clocking network is futile.

4. POTENTIAL PULSE DISAPPEARANCE

We now touch on a problem which has been addressed in [11]. The problem is the potential disappearance of clock pulses when clocking long linear

arrays. Although the study in [11] explicitly addresses the problem in the context of linear array clocking, the problem can also exist when clocking 2-D or higher dimensional structures. The source of the problem is the lack of uniformity of clocking buffers in passing rising and falling edges [11]. This nonuniformity could cause successive reduction in the duty cycle (pulse width) as the pulse travels through a long path of buffers and links. This could lead to the complete disappearance of the pulse. This problem can definitely exist in large size H-Trees since each leaf node would be reachable from the root via a long path consisting alternately of links and buffers.

In BaC clocking, however, the problem of pulse disappearance does not exist. The reason is that, in a sense, each node functions as timing generator. When a node is triggered, it generates a pulse with certain timing properties. The node enforces constraints on the width of the output pulses it produces. Thus each node regenerates the pulse rather then simply buffer it as with buffered networks. Hence irrespective of the size of the BaC clocking network, there is no potential for pulse disappearance. Notice that this conclusion does not preclude delay or threshold variations from node to node.

The above discussion clearly applies to clocking a 2-D mesh of processors using any "valid" BaC network such as those in Figures 5, 6 and 7. The next question then is: how can we clock a long linear array such that the problem of pulse disappearance is avoided? It has been shown [8] that a linear array of size mn can be embedded in a 2-D mesh of size $m \times n$ with dilation 1. This can be achieved by embedding the linear array in a snake like fashion in the 2-D mesh as shown in Figure 10. Since any two adjacent nodes in the linear array are mapped to two adjacent mesh nodes, our earlier discussion applies equally here. This implies that any of the three BaC networks considered in this paper can be utilized to safely clock a long linear array with no potential for pulse disappearance. Alternatively, one can use a linear string of one-input nodes, in manner analogous to *straight line clocking* described in

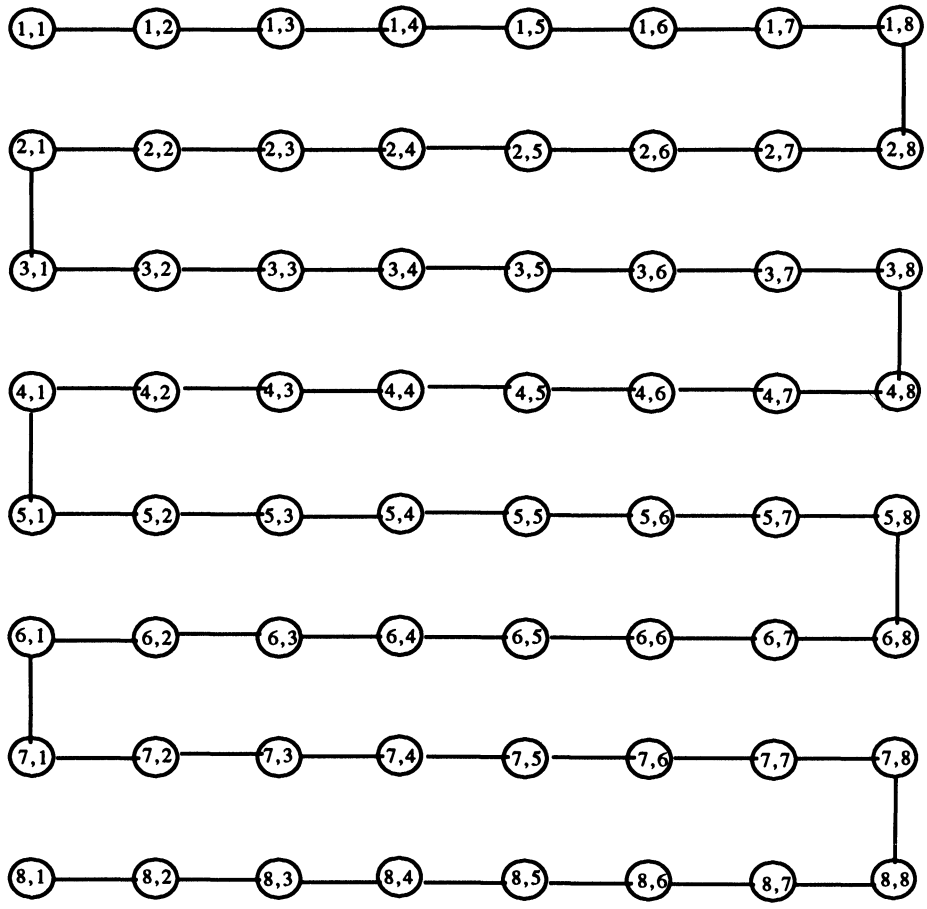

FIGURE 10 Embedding a linear array into a 2D mesh of size 64.

[11], to clock the linear array safely. In this case we simply replace buffers with nodes soley to overcome the potential clock pulse disappearance problem.

5. CONCLUSION

We have performed a comparative study of the H-Tree and three BaC networks in a VLSI layout context. We evaluated the two network types with respect to maximum clock skew, link costs, node costs, area efficiency and maximum edge length. We have shown that insofar as skew is concerned each of the BaC networks will out perform the H-Tree after a certain threshold size. We have also shown that for BaC networks link costs can be comparable (F_3), more (F_4), or less (F_2) than that for the H-Tree. In terms of node costs, BaC networks are more costly by a constant factor (of about 10) than the H-Tree. It has also been demonstrated that BaC networks are superior in terms of maximum link length. Finally we have shown that BaC networks do not suffer from potential pulse disappearance as buffered clock networks, including the H-Tree, do.

References

[1] M. Afghani and C. Svenson, "An upper bound on expected clock skew in synchronous systems," *IEEE Transactions on Computers*, vol. 41, no. 7, pp. 858–872, July 1991.

[2] M.D. Dikaiakos and K. Steiglitz, "Comparison of tree and straight-line clocking for long systolic arrays," *ASSP Conference*, vol. 3.4, 1991, pp. 1177–1180.

[3] A. El-Amawy, "Clocking arbitrarily large computing structures under constant skew bound," *IEEE Transactions on Parallel and Distributed Systems*, vol. 4, no. 3, pp. 241–255, March 1993.

[4] A. El-Amawy, "Branch-and-combine clocking of arbitrarily large computing networks," in *Proc. Intl. Conf. on Parallel Processing*, (St. Charles, IL, Aug. 1991), pp. I-409–417.

[5] A. El-Amawy, "Arbitrarily large clock networks with constant skew bound," U.S. Patent No. 5, 163, 068, 1992.

[6] A. El-Amawy and P. Kulasinghe, "Properties of generalized branch and combine clock networks," *IEEE Trans. on Parallel and Distributed Systems*, to appear.

[7] A. El-Amawy and U. Maheshwar, "Generalized algorithms for systematic synthesis of branch-and-combine clock networks of meshes and Tori," *IEEE Trans. on Parallel and Distributed Systems*, submitted.

[8] A.L. Fisher and H.T. Kung, "Synchronizing large VLSI processor arrays," *IEEE Transactions on Computers*, vol. c-34, pp. 734–740, August 1985.

[9] D. Gordon, "Efficient embeddings of binary trees in VLSI arrays," *IEEE Transactions on Computers*, vol. c-36, no. 9, September 1987.

[10] E. Horowitz and A. Zorat, "The binary tree as an interconnection network: Applications to multiprocessor systems and VLSI," *IEEE Transactions on Computers*, vol. c-30, pp. 247–253, April 1981.

[11] S. Kugelmass and K. Steiglitz, "An upper bound on expected clock skew in synchronous systems," *IEEE Transactions on Computers*, vol. 39, no. 12, pp. 1475–1477, December 1990.

[12] U. Maheshwar and A. El-Amawy, "Algorithms for systematic synthesis of branch-and-combine clock networks for hypercubes," *IEEE Trans. on Parallel and Distributed Systems*, submitted.

[13] C.L. Seitz, "System timing," in C. Mead and L. Conway, *Introduction to VLSI Systems*, Chapter 7, Addison Wesley, Reading, MA, 1980.

[14] M.R. Spiegel, "Mathematical handbook of formulas and tables," McGraw Hill, N.Y., 1986.

[15] H.Y. Youn and A.D. Singh, "On implementing large binary tree architecture in VLSI and WSI," *IEEE Transactions on Computers*, vol. 38, no. 4, April 1989.

Biographies

AHMED EL-AMAWY is an Associate Professor of Electrical and Computer Engineering at Louisiana State University. He joined LSU in 1983 at the rank of Assistant Professor.

Dr. El-Amawy obtained his B.S. and M.S. degrees with distinction in Electrical Engineering from Assiut University, Egypt in 1971 and 1978, respectively. From 1972 to 1975 he worked as a Research Engineer at the National Academy of Science, Cairo, Egypt. From 1979 to 1983 he was a graduate student and Research Assistant at Iowa State University, Ames, IA. He earned his Ph.D. in 1983 from Iowa State University in Computer Engineering.

Dr. El-Amawy's current research interests include parallel computing, VLSI algorithms, computer architecture, fault-tolerant computing, and synchronization schemes for very large systems. He owns two U.S. patents one of which describes the first clock distribution scheme capable of bounding the skew by a constant irrespective of network size. He has served on the program committees for the 6th and 7th International Conference on Computing and Information, Peterborough, Canada, 1994. He has also served on the committee or as session chairman or organizer in several IEEE conferences.

Dr. El-Amawy is a senior member of the IEEE, Computer Society, and Phi Kappa Phi.

MAHESHWAR UMASANKAR received his Bachelor's Degree from Birla Institute of Technology and Science, Pilani, India in June 1989. He then worked as an R & D Engineer with Continental Devices India Ltd., Delhi, India (July 89–May 91). He was involved in the design of Public Switched Telephone Networks (PSTN) and tester development for Integrated Services Digital Networks (ISDN). He has worked as a Research Assistant (from Jan. 92 till Dec. 93) in the Department of Electrical and Computer Engineering, Louisiana State University, Baton Rouge and worked on Synthesis of Synchronous Clock Networks for Multiprocessor Architectures. He has received his Master's Degree in Electrical Engineering from LSU in December 93. His areas of interest include Computer Architecture, Parallel Processing and VLSI Design.