VLSI DESIGN
1998, Vol. 7, No. 1, pp. 31–57
Reprints available directly from the publisher
Photocopying permitted by license only

Automated Synthesis of Skew-Based Clock Distribution Networks

JOSÉ LUIS NEVES and EBY G. FRIEDMAN*

Department of Electrical Engineering, University of Rochester, Rochester, NY 14627

In this paper a top-down methodology is presented for synthesizing clock distribution networks based on application-dependent localized clock skew. The methodology is divided into four phases: 1) determination of an optimal clock skew schedule for improving circuit performance and reliability; 2) design of the topology of the clock tree based on the circuit hierarchy and minimum clock path delays; 3) design of circuit structures to implement the delay values associated with the branches of the clock tree; and 4) design of the geometric layout of the clock distribution network. Algorithms to determine an optimal clock skew schedule, the optimal clock delay to each register, the network topology, and the buffer circuit dimensions are presented.

The clock distribution network is implemented at the circuit level in CMOS technology and a design strategy based on this technology is presented to implement the individual branch delays. The minimum number of inverters required to implement the branch delays is determined, while preserving the polarity of the clock signal. The clock lines are transformed from distributed resistive-capacitive interconnect lines into purely capacitive interconnect lines by partitioning the RC interconnect lines with inverting repeaters. The inverters are specified by the geometric size of the transistors, the slope of the ramp shaped input/output waveform, and the output load capacitance. The branch delay model integrates an inverter delay model with an interconnect delay model. Maximum errors of less than 2.5% for the delay of the clock paths and 4% for the clock skew between any two registers belonging to the same global data path are obtained as compared with SPICE Level-3.

Keywords: Clock distribution networks, clock scheduling, clock skew, clock tree, CMOS inverters, repeaters

1. INTRODUCTION

Most existing digital systems utilize fully synchronous timing, requiring a reference signal to control the temporal sequence of operations. Globally distributed signals, such as clock signals, are used to provide this synchronous time reference. These signals can dominate and limit the performance of VLSI-based systems. This characteristic is, in part, due to the continuing reduction of feature size

*Corresponding author.

concurrent with increasing chip dimensions. Thus interconnect delay has become of increasing significance, perhaps of greater importance than active device delay. Furthermore, the design of the clock distribution network, particularly in high speed applications, requires significant amounts of time, inconsistent with the high design turnaround of the more common data flow portion of a circuit.

Several techniques have been developed to improve the performance and design efficiency of clock distribution networks, such as buffer insertion [1] to reduce propagation delay and power consumption of clock distribution networks, symmetric distribution networks [2] such as H-tree structures to ensure minimal clock skew, and zero-skew clock routing algorithms [e.g., 3, 4] to automatically layout high speed clock distribution networks in cell-based designs. A common trait of these approaches is that the clock distribution network is designed so as to minimize the clock skew between each register, not recognizing that localized clock skew [5, 6] can be used to improve synchronous circuit performance and minimize the likelihood of any race conditions. Furthermore, no known techniques exist today that can automatically synthesize high speed and robust clock distribution techniques while including distributed buffers along the clock path. A novel methodology is therefore presented in this paper for efficiently synthesizing distributed buffer tree-structured clock distribution networks that exploit non-zero localized clock skew to improve circuit performance. This methodology is represented as part of the overall IC design cycle in Figure 1.

The top-down clock distribution design methodology described in this paper is divided into four major phases. The first phase is the determination of an optimal clock skew schedule [7, 8], specifying the localized clock skew schedule which maximizes circuit performance and reliability. In the second phase, a topological design of the clock distribution network is obtained [9]. The topology is specified in terms of the structure of the clock distribution network, configured as a tree with

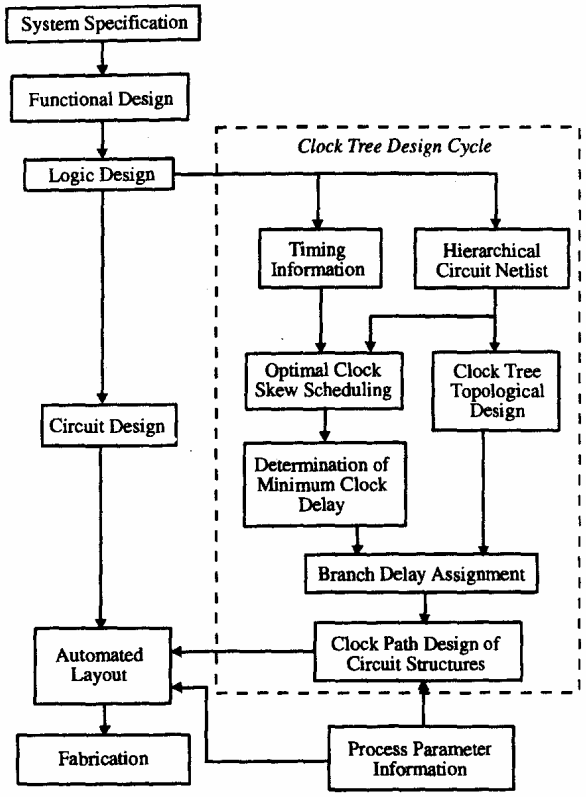

FIGURE 1 Block diagram of the clock tree design cycle integrated with standard integrated circuit design flow.

minimum delay values assigned to each branch of the network. These branch delays satisfy the clock skew specifications derived from the optimal clock skew schedule. In the third phase, circuit elements are designed to implement the delay values determined during the topological design phase. The final phase is the layout design of the clock distribution network, in which the physical layout of the clock distribution network is merged with the overall structure of the VLSI circuit.

The technology and layout affect the design of clock distribution networks in two ways. First, the accuracy of the clock distribution network requires that the design methodology consider the effects of process variations on the implementation of the circuit structures, otherwise unacceptable variations in the values of the local clock skew may be induced. Second, the layout of the clock distribu-

tion network is highly dependent upon the design methodology and the physical floorplan of the functional circuit. It is for these reasons that the layout phase is treated separately and is therefore not directly addressed in this paper. However, layout and technology related issues are discussed in the Appendix.

The first three phases of the clock distribution design methodology are addressed here. Assuming the timing information of a circuit, namely the logic, register, and interconnect delays, is known, an optimal clock skew schedule is obtained that provides the minimum clock period of the circuit. This material is presented in Section 2. Furthermore, with the localized clock skew schedule and technology related information, an optimal delay is assigned to each clock path driving each register in the circuit and is presented in Section 3. With the clock delay information, the topology of the clock distribution network can be determined with minimum delay values assigned to each branch of the network [9]. This topic is described in Section 4. Circuit structures implementing these delay values are designed based on the temporal specification derived from the topological design of the clock distribution network [10]. The branch delay values are implemented with CMOS inverters, once the minimum number of inverters required to preserve the polarity of the clock signal is determined. The inverters are described in terms of transistor geometry (width/length ratio), the slope of the input and output signals, and the output capacitive load. The use of distributed inverters placed along the clock path converts a resistive-capacitive network into a purely capacitive network, greatly simplifying the delay models of the network and increasing the accuracy of the circuit implementation. This process is described in Section 5. Delay calculations for each clock path and comparison with SPICE simulations for several circuit examples are presented in Section 6. The accuracy of this clock distribution design methodology is measured by comparing the analytically derived clock skew with the SPICE simulations and the originally specified clock skew schedule. Finally, in Section 7, the primary results of this paper are summarized. Thus, a fully specified circuit level description of a clock distribution network exploiting non-zero localized clock skew is developed using the methodology and algorithms presented in this paper.

2. OPTIMAL CLOCK SKEW SCHEDULING

The first phase in the design of a clock distribution network is the determination of the minimum local clock skews that will increase circuit performance by reducing the maximum clock period while ensuring that no race conditions exist. This phase is called *optimal clock skew scheduling* and has been extensively studied [6, 7, 8, 11]. In this section, these existing approaches are summarized and applied to the problem of clock distribution network synthesis. Assuming the timing characteristics of the circuit are known, such as the minimum and maximum delay of each combinational logic block and the register delay characteristics, it is possible to determine the local clock skew of each data path and the minimum system-wide clock period (maximum clock frequency). This process is accomplished by formulating the optimal clock scheduling problem as a linear programming problem and solving the set of linear inequalities with standard linear programming techniques [12]. In Section 2.1, a model of a synchronous circuit is presented. In Section 2.2, two types of race conditions that can occur in a synchronous circuit are presented. Furthermore, sufficient timing relationships that will prevent the occurrence of these race conditions are described, relating the delay of the registers, logic blocks, interconnect, and clock skew. Chip-to-Chip (e.g., board level) clock skew is discussed in Section 2.3. In Section 2.4, an algorithm to determine the minimum clock period and optimal clock skew schedule is presented. Finally, in Section 2.5, the process of determining an optimal clock skew schedule is illustrated with an example.

Before discussing optimal clock skew scheduling, a definition of clock skew is first presented. Clock skew is manifested by a lead/lag relationship between the clock signals that control a local data path, where a local data path is composed of two sequentially adjacent registers with, typically, combinational logic between them.

DEFINITION 1 Given two sequentially adjacent registers, R_i and R_j, the clock skew between these two registers is defined as

$$T_{\text{Skew}ij} = T_{\text{CD}_i} - T_{\text{CD}_j}, \tag{1}$$

where T_{CD_i} and T_{CD_j} are the clock delays from the clock source to the registers R_i and R_j, respectively.

Furthermore, for a given local data path, if the clock delay to the initial register is less than the clock delay to the final register, the clock skew is described in this paper as *negative*. Likewise, if the clock delay to the initial register is greater than the clock delay to the final register, the clock skew is described as *positive* [13]. Finally, a *global data path* is a data path consisting of one or more local data paths.

2.1. Model of a Synchronous Circuit

A block diagram of a multi-stage synchronous digital system is depicted in Figure 2. Between each pair of registers there is, typically, a block of combinational logic with possible feedback paths between the registers. Each register is either a multi- or single-bit register, and all the register outputs are assumed to change at the same transition point of the clock signal. Only one single-phase clock signal source is assumed. The registers are composed of edge-triggered flip-flops, each assuming a single output state for each clock cycle. The combinational logic block is described in terms of the maximum and minimum delay values, $T_{\text{Logic(max)}}$ and $T_{\text{Logic(min)}}$, respectively. These logic delay values are obtained by considering the delay of all possible input to output paths within each combinational logic block. For simplicity and without loss of generality, the block diagram in Figure 2 only considers a single input, single output system. The system is modeled by the registers R_i through R_l, the logic blocks between each of the registers, and the clock path delays, C_{di} to C_{dl}. The registers R_s and R_t model a local data path inserted as a feedback path. The registers R_{in} and R_{out} are registers placed at the input and

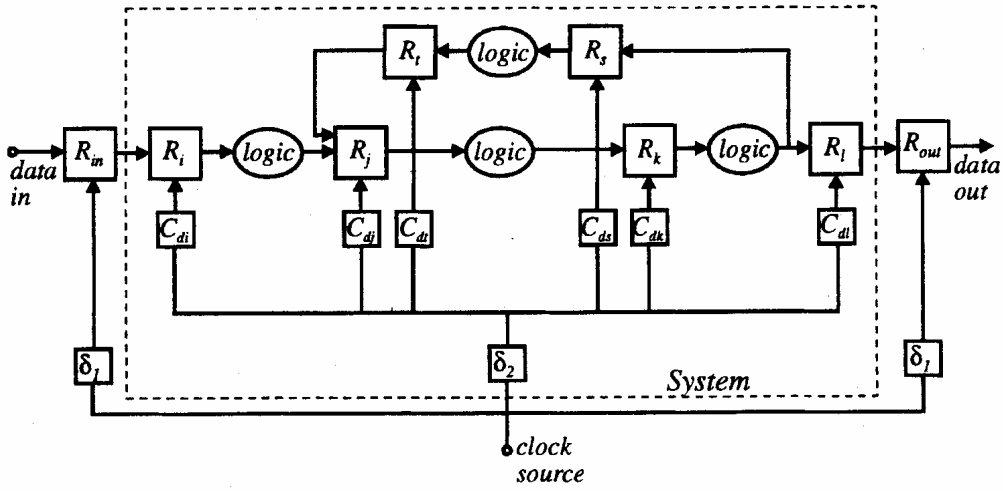

FIGURE 2 Multi-stage block diagram of a synchronous digital system.

output of the circuit, respectively. In Figure 2, δ_1 and δ_2 are intentional delays inserted into the clock paths. The delay δ_1 is the delay of the clock lines driving the system interface registers, while the delay δ_2 is the delay of the clock line driving the on-chip registers of the system. Although in Figure 2 the clock signal path delays are shown as being independent of each other, in Section 4 a design strategy is presented for determining the hierarchical topology of a tree structured clock distribution network.

2.2. Preventing Clock Hazards

To illustrate the existence of clock hazards in a local data path, consider the circuit illustrated in Figure 3a. For an input data signal to be transferred correctly from the input of R_i to the output of R_j, the input data should be transferred to the output of R_i upon arrival of a clock pulse and transferred to the output of R_j with the arrival of the following clock pulse. This situation is illustrated in Figure 3b. Race conditions in these synchronous systems may occur for two reasons. First, a race condition occurs if a data signal appearing at the output of R_i upon arrival of a clock pulse is not available at the input of R_j when the following clock pulse arrives at R_j. This case is called *zero clocking* [6] due to the excessive use of *positive clock skew* [5, 13], and is illustrated in Figure 3c. For example, the data appearing at the output of R_i, upon arrival of a clock pulse, is available at the input of R_j in $t=7$ tu (time units), while the following clock pulse arrives at R_j in $t=6$ tu. In the second race condition, the data signal appearing at the output of R_i upon arrival of a clock pulse is available at the input of R_j before the same clock pulse arrives at R_j. This case is called *double clocking* [6] due to the excessive use of *negative clock skew* [5], and is illustrated in Figure 3d, where the clock pulse that triggered R_i arrives at R_j in $t=6$ tu, while the data signal appearing at the output of R_i arrives at R_j in $t=5$ tu.

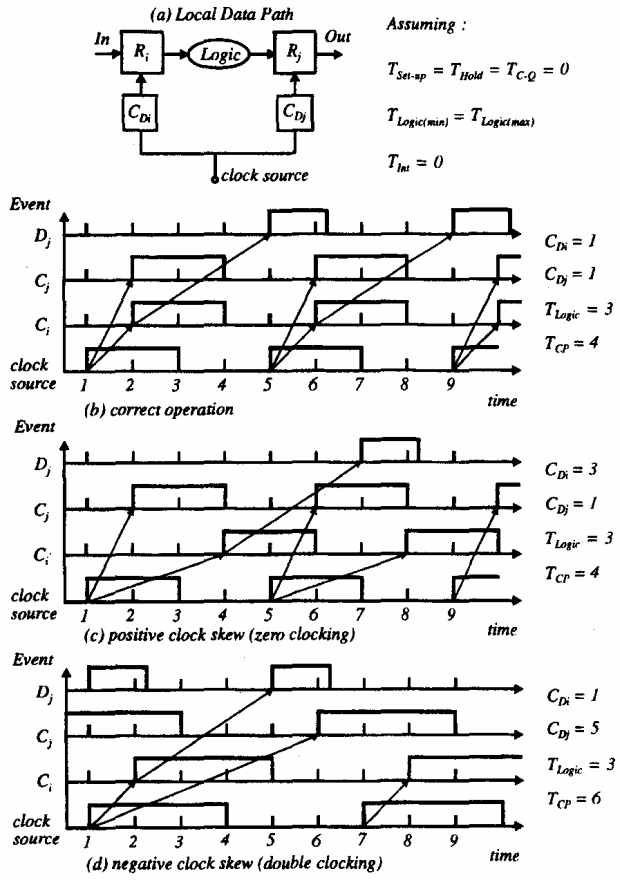

FIGURE 3 Example of clock hazards. (a) block diagram; (b) correct timing operation; (c) positive clock skew; (d) negative clock skew.

To prevent both types of clock hazards, a set of inequalities must be satisfied for each local data path. These inequalities are described in terms of the system clock period T_{CP} and the individual delay components of the local data paths [5, 6, 7, 8, 10]. To avoid negative clock skew creating a race condition between two sequentially adjacent registers, R_i and R_j, the following inequality must be satisfied,

$$T_{\text{Skew}ij} \geq T_{\text{Hold}} - T_{\text{PD(min)}}, \tag{2}$$

where

$$T_{\text{PD(min)}} = T_{C-Qi} + T_{\text{Logic(min)}} + T_{\text{Int}} + T_{\text{Set-up}xj}, \tag{3}$$

and where $T_{\text{Skew}ij}$ is the difference in delay between the clock path delays T_{CD_i} and T_{CD_j}, T_{Hold} is the

amount of time the input data must be stable after the clock signal changes state, T_{C-Q} is the time required for the data signal to leave R_i once it is triggered by a clock pulse C_i, $T_{\text{Logic(min)}}$ is the minimum propagation delay through the logic block between the registers R_i and R_j, T_{Int} is the interconnect delay, and $T_{\text{Set-up}}$ is the time required to successfully propagate to and latch the data signal within R_j. To avoid positive clock skew creating a race condition (by limiting the maximum clock frequency) between two sequentially adjacent registers, R_i and R_j, the following inequality must be satisfied

$$T_{\text{CP}} \geq T_{\text{Skew}ij} + T_{\text{PD(max)}}, \tag{4}$$

where

$$T_{\text{PD(max)}} = T_{C-Qi} + T_{\text{Logic(max)}} + T_{\text{Int}} + T_{\text{Set-up}j}, \tag{5}$$

and where $T_{\text{Logic(max)}}$ is the maximum propagation delay through the logic block between the registers R_i and R_j.

2.3. Off-Chip Zero Clock Skew

Although it is possible to have off-chip non-zero clock skew, it is desirable to ensure that the clock skew between the VLSI circuit I/O's approach zero, in order to avoid complicating the specification of the interface of the circuit with other components also controlled by the same clock source, typically at the circuit board level. To avoid race conditions and limiting clock rates, the delay of the clock path driving the output register is chosen equal to the delay of the clock path driving the input register. Consider, for example, the system illustrated in Figure 4a. In Figure 4b, the delay of the clock path driving R_{out} is less than the delay of the clock path driving R_{in}, causing a race condition since any data latched into R_{out} is also latched into R_{in} by the same clock pulse. Nevertheless, observe that a negative clock skew is permissible between registers R_{out} and R_{in} since the amount of skew is less than the clock period of the system. The operation of the system where both clock signals have the same delay is illustrated in Figure 4c.

If a system is composed of more than two circuits, all driven by the same clock source, the same approach to applying zero clock skew between the input and output registers is used. Therefore, a clock skew relationship can be established between the input and output off-chip registers along the internal global data paths of the circuit, and is expressed as

$$T_{\text{Skew in,out}} = T_{\text{Skew in,1}} + \cdots + T_{\text{Skew } i-1,i} + \cdots + T_{\text{Skew } n,\text{out}} = 0. \tag{6}$$

Note that (6) is only valid if the interface circuitry is controlled by the same clock source. The constraint does not apply to asynchronous circuits or synchronous circuits that communicate asyn-

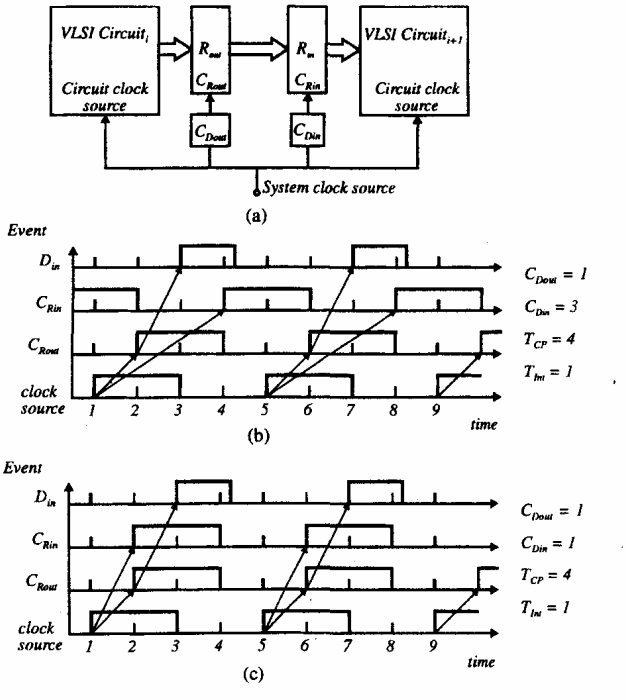

FIGURE 4 Matching of I/O clock skew between VLSI circuits, (a) block diagram; (b) data transfer without matching clock skew; (c) data transfer with matching clock skew.

chronously. Furthermore, restricting the off-chip clock skew to zero does not preclude the VLSI circuit from being optimized using intentional non-zero clock skew. The primary difference is that the performance improvement of the VLSI circuit is less than what would be obtained without this constraint. This concept of applying non-zero clock skew at the off-chip level is further exemplified in Section 2.5.

2.4. Minimum Clock Period

The system-wide clock period is minimized by finding a set of clock skew values that satisfy (2)–(6) for each local data path. Again, the timing characteristics of each local data path is assumed to be known. The minimum clock period is obtained when the problem is formalized as a *linear programming problem*, such as
minimize:

$$T_{CP}$$

subject to the local and global timing constraints:

$$\begin{aligned} T_{\text{Skew}ij} &\geq T_{\text{Hold}} - T_{\text{PD(min)}} \\ T_{CP} &\geq T_{\text{Skew}ij} + T_{\text{PD(max)}} \\ T_{\text{PD(max)}} &= T_{C-Qi} + T_{\text{Logic(max)}} + T_{\text{Int}} + T_{\text{Set-up}j} \\ T_{\text{PD(min)}} &= T_{C-Qi} + T_{\text{Logic(min)}} + T_{\text{Int}} + T_{\text{Set-up}j} \\ T_{\text{Skew in, out}} &= T_{\text{Skew in,1}} + \cdots + T_{\text{Skew } i-1,i} + \cdots \\ &\quad + T_{\text{Skew } n\text{, out}} = 0. \end{aligned}$$

2.5. Circuit Example

A circuit example for determining the minimum clock period of a multi-stage system with feedback paths is illustrated in Figure 5. The numbers inside the logic blocks are the minimum and maximum delays of each block, respectively. Without loss of generality, the register timing parameters are assumed to be constant and equal for each register. The example is intended to represent a small circuit, composed of five global data paths. These data paths illustrate different timing characteristics

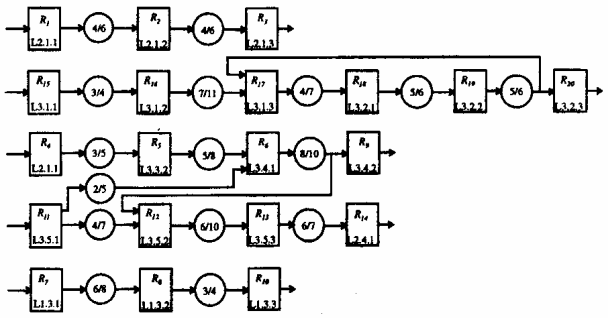

FIGURE 5 Circuit example composed of five global data paths.

of a synchronous circuit. The first data path, R_1 to R_3, illustrates the specification of zero clock skew between each register. The second data path, R_{15} to R_{20}, illustrates non-zero clock skew and the effect of feedback paths on the clock period. The third and four paths, R_4 to R_9 and R_{11} to R_{14}, illustrate non-zero clock skew and the existence of interconnections between global data paths. The last data path, R_7 to R_{10}, illustrates the effect of non-zero clock skew on reducing the clock period. In each of these paths it is assumed that the first and last registers of each global data path are off-chip registers. Therefore, the clock skew between those registers must satisfy (6). Each global data path is analyzed independently to obtain the optimal clock skew schedule as well as the minimum clock period. The minimum clock period of the example circuit is the maximum value of the clock periods obtained from each data path.

The linear relationships that determine the minimum clock period and the optimal clock skew schedule for the circuit shown in Figure 5 is described in terms of each independent global data path within the circuit. The constraints of each global data path are listed in Table I, where the numbers represent the maximum and minimum delays of the logic blocks between sequentially adjacent registers. For simplicity, the clock-to-Q and set-up times of the registers are assumed to be zero. The solution of the linear inequalities given in Table I is shown in Table II, where an optimal clock skew schedule as well as the minimum clock period is provided.

TABLE I Constraint list for the data paths in the example circuit shown in Figure 5

Path R_1-R_3	Path $R_{15}-R_{20}$	Path R_4-R_9; $R_{11}-R_{14}$	Path R_7-R_{10}
R_1-R_2: $C_{d1}-C_{d2}$ $T_{Skew1-2} \geq -4$ $T_{Skew1-2} - T_{CP} \leq -6$	$R_{15}-R_{16}$: $C_{d15}-C_{d16}$ $T_{Skew15-16} \geq -3$ $T_{Skew15-16} - T_{CP} \leq -4$	R_4-R_5: $C_{d4}-C_{d5}$ $T_{Skew4-5} \geq -3$ $T_{Skew4-5} - T_{CP} \leq -5$	R_7-R_8: $C_{d7}-C_{d8}$ $T_{Skew7-8} \geq -6$ $T_{Skew7-8} - T_{CP} \leq -8$
R_2-R_3: $C_{d2}-C_{d3}$ $T_{Skew2-3} \geq -4$ $T_{Skew2-3} - T_{CP} \leq -6$	$R_{16}-R_{17}$: $C_{d16}-C_{d17}$ $T_{Skew16-17} \geq -7$ $T_{Skew16-17} - T_{CP} \leq -11$	R_5-R_6: $C_{d5}-C_{d6}$ $T_{Skew5-6} \geq -5$ $T_{Skew5-6} - T_{CP} \leq -8$	R_8-R_{10}: $C_{d8}-C_{d10}$ $T_{Skew8-10} \geq -3$ $T_{Skew8-10} - T_{CP} \leq -4$
$T_{Skew1-2} + T_{Skew2-3} = 0$	$R_{17}-R_{18}$: $C_{d17}-C_{d18}$ $T_{Skew17-18} \geq -4$ $T_{Skew17-18} \geq -7$ $R_{18}-R_{19}$: $C_{d18}-C_{d19}$ $T_{Skew18-19} \geq -5$ $T_{Skew18-19} - T_{CP} \leq -6$	R_6-R_9: $C_{d6}-C_{d9}$ $T_{Skew6-9} \geq -8$ $T_{Skew6-9} - T_{CP} \geq -10$ $T_{Skew4-5} + T_{Skew5-6} + T_{Skew6-9} = 0$ $T_{Skew4-5} + T_{Skew5-6} + T_{Skew6-12} + T_{Skew12-13} + T_{Skew13-14} = 0$ $T_{Skew11-6} + T_{Skew6-12} = 0$	$T_{Skew7-8} + T_{Skew8-10} = 0$
	$R_{19}-R_{20}$: $C_{d19}-C_{d20}$ $T_{Skew19-20} \geq -5$ $T_{Skew19-20} - T_{CP} \leq -6$	$R_{11}-R_{12}$: $C_{d11}-C_{d12}$ $T_{Skew11-12} \geq -4$ $T_{Skew11-12} - T_{CP} \leq -7$	
	$R_{19}-R_{17}$: $C_{d19}-C_{d17}$ $T_{Skew19-17} \geq -5$ $T_{Skew19-17} - T_{CP} \leq -6$	$R_{12}-R_{13}$: $C_{d12}-C_{d13}$ $T_{Skew12-13} \geq -6$ $T_{Skew19-20} - T_{CP} \leq -10$	
	$T_{Skew15-16} + T_{Skew16-17} + T_{Skew17-18}\ T_{Skew18-19} + T_{Skew19-20} = 0$	$R_{13}-R_{14}$: $C_{d13}-C_{d14}$ $T_{Skew13-14} \geq -6$ $T_{Skew13-14} - T_{CP} \leq -7$	
	$T_{Skew17-18} + T_{Skew18-19} = -T_{Skew19-17}$	$R_{11}-R_6$: $C_{d11}-C_{d6}$ $T_{Skew11-6} \geq -2$ $T_{Skew11-6} - T_{CP} \leq -5$	
		R_6-R_{12}: $C_{d6}-C_{d12}$ $T_{Skew6-12} \geq -8$ $T_{Skew6-12} - T_{CP} \leq -10$ $T_{Skew11-12} + T_{Skew12-13} + T_{Skew13-14} = 0$	

TABLE II Optimal clock skew schedule and clock period for the example circuit shown in Figure 5

$T_{Skew1-2} = 0$	$T_{Skew16-17} = -4$	$T_{Skew19-20} = 0$	$T_{Skew5-6} = 0$	$T_{Skew12-13} = -2$	$T_{Skew6-12} = -2$
$T_{Skew2-3} = 0$	$T_{Skew17-18} = 0$	$T_{Skew19-17} = -1$	$T_{Skew6-9} = -3$	$T_{Skew13-14} = 1$	$T_{Skew7-8} = -2$
$T_{Skew15-16} = 3$	$T_{Skew18-19} = 1$	$T_{Skew4-5} = 3$	$T_{Skew11-12} = 1$	$T_{Skew11-6} = 3$	$T_{Skew8-10} = 2$
$T_{CP} = 8$					

If zero clock skew between off-chip registers (and on-chip registers) is not required, the minimum clock period is $T_{CP} = 7$ tu. Although the restriction of zero clock skew between system I/O increases the minimum clock period to 8 tu (as shown in Table II), a performance improvement is still obtained by applying localized non-zero clock skew within the circuit. If zero clock skew is assumed throughout the circuit, the minimum clock period becomes $T_{CP} = 11$ tu, due to the worst case path delay of the local data path between registers R_{16} and R_{17}. Thus, a 27% improvement in the minimum clock period is achieved by optimally scheduling the on-chip localized clock skews while retaining zero off-chip clock skew.

3. DETERMINATION OF MINIMUM CLOCK PATH DELAY

Synthesizing a clock distribution network requires a description of the circuit at the register transfer

level (RTL) and the local clock skew between each pair of sequentially adjacent registers. The local non-zero clock skew is obtained from the optimal clock scheduling process described in Section 2. In order to determine the minimum delay of each clock path, information describing both the local clock skew and the relationship between the clock skew of each local data path belonging to the same global data path is required. In Section 3.1, this relationship for global data paths with forward and feedback connections is formalized. In Section 3.2, an algorithm is presented for calculating the minimum clock path delays.

3.1. Theoretical Background

To determine the minimum clock delay from the clock source to each register, it is necessary to note any relationships that may exist among the clock skews of sequentially adjacent registers occurring within a global data path. Specifically, the effects of feedback paths within global data paths on clock skew must be considered. The relationships among the clock skews of sequentially adjacent registers in a global data path is called conservation of clock skew. These relationships are formalized below:

THEOREM 1 *For any given global data path, clock skew is conserved. Alternatively, the clock skew between any two registers which are not necessarily sequentially adjacent is the sum of the clock skews between each pair of registers along the global data path between those two same registers.*

Proof: This theorem is proved by induction. For a global data path with only two registers, the clock skew is defined by Definition 1. Now add an extra register to the global data path as illustrated in Figure 6. The clock skew between registers R_i and R_j is $T_{\text{Skew}ij} = T_{\text{CD}_i} - T_{\text{CD}_j}$, and the clock skew between registers R_j and R_k is $T_{\text{Skew}ik} = T_{\text{CD}_j} - T_{\text{CD}_k}$. Substituting $T_{\text{Skew}ik}$ into $T_{\text{Skew}ij}$, $T_{\text{Skew}ij} + T_{\text{Skew}ik} = T_{\text{CD}_i} - T_{\text{CD}_k}$ is obtained, which is the clock skew between registers R_i and R_k. By adding n registers to the global data path, the clock skew between the first and the last register is $T_{\text{Skew}in} = T_{\text{Skew}ij} + \cdots + T_{\text{Skew}in-1,n} = T_{\text{CD}_i} - T_{\text{CD}_n}$. If one register is added to the global data path, the clock skew between registers R_n and R_{n+1} is $T_{\text{Skew}n,n+1} = T_{\text{CD}_n} - T_{\text{CD}}n + 1$. Substituting $T_{\text{Skew}n,n+1}$ into $T_{\text{Skew}i,n}$, $T_{\text{Skew}i,n+1} = T_{\text{Skew}ij} + \cdots + T_{\text{Skew}n,n+1}$ is obtained. Since n can be any value greater than two, this theorem is proved. □

Although clock skew is defined between two sequentially adjacent registers, Theorem 1 shows that clock skew can exist between any two registers in a global data path. Therefore, it extends the definition of clock skew introduced by Definition 1 to any two non-sequentially adjacent registers belonging to the same global data path. Theorem 1 also illustrates that the clock skew between any two non-sequentially adjacent registers which do not belong to the same global data path has no physical meaning.

A typical sequential circuit may contain sequential feedback paths, as illustrated in Figure 7. It is possible to establish a relationship between the clock skew in the forward path and the clock skew in the feedback path, because the initial and final registers in the feedback path are also registers in the forward path. As shown in Figure 7, the initial and final registers in the feedback path R_l–R_j are the final and initial registers of the forward path R_j–R_k–R_l. This relationship is formalized below:

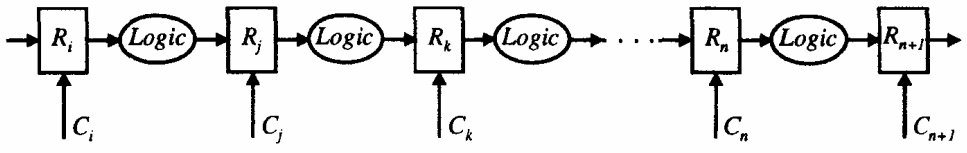

FIGURE 6 Example of a global data path.

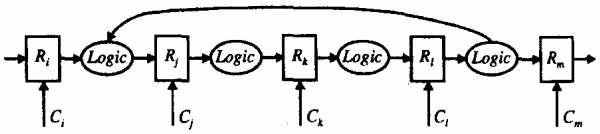

FIGURE 7 Data path with feedback paths.

THEOREM 2 *For any given global data path containing feedback paths, the clock skew in a feedback path between any two registers, say R_l and R_j, is related to the clock skew of the forward path by the following relationship,*

$$T_{\text{Skew}_{\text{feedback},lj}} = -T_{\text{Skew}_{\text{forward},jl}}. \tag{7}$$

Proof The theorem will be proved by contradiction by first showing that (7) is valid, and then by showing that if (7) is not valid, Theorem 1 and Definition 1 are not satisfied. Applying Theorem 1 to the forward path between registers R_l and R_j, $T_{\text{Skew}lj} = T_{\text{CD}_l} - T_{\text{CD}_j}$ is obtained. Likewise, applying Theorem 1 to the feedback path between registers R_j and R_l, $T_{\text{Skew}jl} = T_{\text{CD}_j} - T_{\text{CD}_l}$ is obtained, which is the negative of $T_{\text{Skew}ij}$. Consider that the clock skew in the feedback path, $T_{\text{Skew}ij} \neq T_{\text{CD}_j} - T_{\text{CD}_l}$. However, this assumption contradicts Theorem 1, and therefore the assumption is false, validating the theorem. □

3.2. Clock Path Delay Algorithm

A synchronous circuit is formed by one or more global data paths as shown in the example illustrated in Figure 5. For each global data path, the clock delay of the registers is calculated by first choosing the local data path with the largest clock skew. If the clock skew is positive (negative), the clock delay of the final (initial) register of the local data path is assigned a constant K, and the clock delay of the initial (final) register is assigned the constant K plus (minus) the clock skew of the final (initial) register, where the constant K is the minimum clock delay of the circuit. The clock delay of the following register connected to the final register of the local data path is the clock delay of the final register minus the clock skew between them. Similarly, the clock delay of the previous register connected to the initial register is the clock delay of the initial register plus the clock skew between them. Proceeding in this fashion for the remaining registers, the clock delay to each of the registers in the global data path is calculated in terms of K. If the clock skew is zero for every local data path in the global data path, the clock path delay is the same for each register.

To illustrate how the clock delay is calculated, consider the synchronous circuit illustrated in Figure 5. This circuit is represented in Figure 8 by a graph where a vertex represents a register instance, a directed edge represents a physical path between a pair of registers, and an edge weight represents the clock skew of a local data path. Consider, for example, the global data path, R_{15} to R_{20}. The maximum absolute clock skew is 4 tu for the local data path (R_{16}, R_{17}) in Figure 8. Therefore, the clock delay to R_{16} from the clock source is the constant K, and the remaining clock delays are given as a function of K plus the local

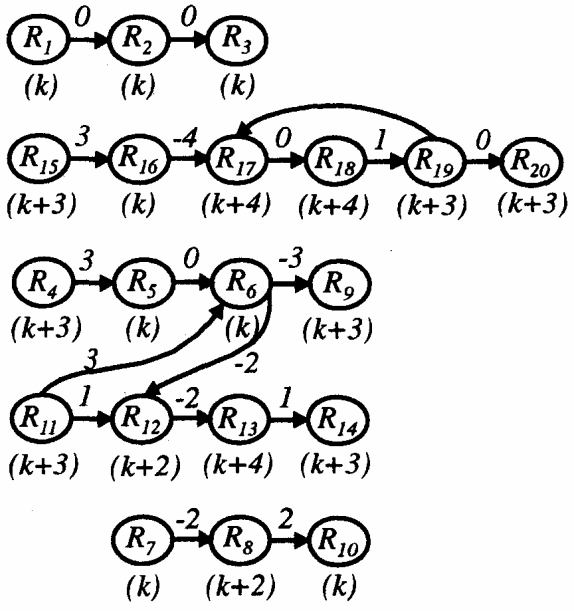

FIGURE 8 Optimal delay assignment for the example circuit in Figure 5.

values of clock skew. Observe that the constant value for the global data path composed of registers R_1 to R_3 may be different from the constant value for the global data path composed of registers R_{15} to R_{20}, since there are no common registers (i.e., connections) between these two paths.

The procedure to calculate the minimum clock delay is formalized in an algorithm called *Path_Delay*, which is summarized in Figure 9. After determining the local data path with the greatest clock skew, each of the clock delays to the other registers is calculated by traversing the global data path and attributing to each node the clock delay value that enforces the desired clock skew specification. If the initial clock skew specification satisfies Definition 1 and Theorems 1 and 2, each node is visited only once and the time complexity of the algorithm is $O(n)$, where n is the total number of nodes in the graph. The clock delay of each register obtained with this algorithm is represented by an expression in terms of K attached to each vertex, as illustrated in Figure 8.

Assuming a single clock source and a clock distribution network that produces the delay values found from the previous procedure, the clock skew requirements of the circuit are satisfied. Such a network is illustrated in Figure 10 for the example circuit shown in Figure 5, where the numerals inside the rectangles represent clock delays and the value of K is assumed to be 1 tu.

```
Algorithm Path_Delay()
begin
  Circuit = G(v,e);
  for each Data path D in Circuit G do
  find MAX(TSkew_ij);
  if ( TSkew_ij < 0 )
     v_i = K;
     v_j = K - TSkew_ij ;
  else
     v_j = K;
     v_i = K + TSkew_ij ;
  while (v_i,v_j not visited) and (v_i,v_j in Data path D) do
     Delay v_j+1 = delay v_j + TSkew_j,j+1;
     Delay v_i-1 = delay v_i + TSkew_i-1,i;
     Mark v_i-1, v_j+1 as visited;
end Path_Delay
```

FIGURE 9 Algorithm to find the optimal clock delay to each register.

Providing independent clock path delays for each register in a circuit, as is done in Figure 10, is impractical, due to the unnecessarily large capacitive load placed on the clock source and the inefficient use of die area. It is preferable to reduce this load by allowing the paths of the faster clock nets to be built from the paths of the slower clock nets of neighboring clock paths. This approach suggests a tree structure for the clock distribution network, where the branching points are selected according to the delay at each individual node and the relative physical location of the clocked registers. Such an approach for determining the structure of a clock distribution network is described in the following section.

4. TOPOLOGY OF CLOCK DISTRIBUTION NETWORK

In this section, a methodology is presented for designing the topology of a clock distribution network that implements the delay values obtained from the algorithm *Path_Delay* introduced in Section 3. A reasonable initial strategy is to provide an independent path delay for each register, as shown in Figure 10. This approach has the advantage of isolating each clock signal and requires a simple circuit composed of cascaded inverters and interconnect sections, for which delay models have been studied in great detail [e.g., 1, 13, 15]. However, independent clock paths for each register are impractical because a typical VLSI circuit may contain many thousands of registers, expending excessive area.

In this paper the clock distribution network for a specific circuit is designed in a tree structure using the hierarchical information of the original circuit description and the clock delay values obtained from the algorithm *Path_Delay*. In Section 4.1,

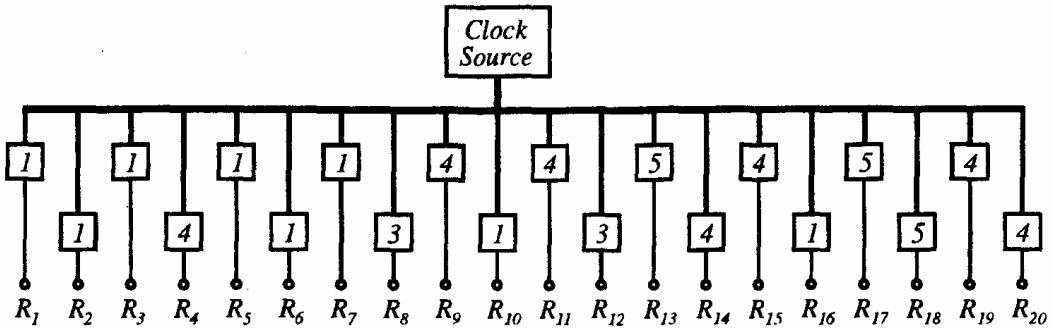

FIGURE 10 Clock distribution network for the example circuit shown in Figure 5.

the construction of the clock tree based on the hierarchical description of the circuit is described. Techniques to calculate the branch delays are presented in Section 4.2. In Section 4.3, a methodology for constructing clock distribution networks with minimal or no hierarchical information is described.

4.1. Clock Network Topology

Almost any large VLSI circuit is composed of several levels of hierarchy, where the number of levels is defined by the size and organization of the system and related factors such as the complexity of the circuit, the number of transistors, and the design methodology. During circuit placement, the elements of each hierarchical module are typically placed physically close to each other. Based on this reasoning, the hierarchical description of the circuit can be used to constrain the structure of the clock distribution network. In Section 4.3, the approach described herein to develop the topology of a clock distribution network is extended to handle flat non-hierarchical databases.

To build the clock distribution network, the hierarchy of the circuit is extracted from the circuit netlist and represented as a tree structure. The root vertex is the clock source, the internal vertices are the branching points derived from the hierarchy, and the leaf vertices are the registers. The hierarchical tree structure of the circuit example shown in Figure 5 is illustrated in Figure 11. For the purpose of calculating the individual clock delays, the branches of the tree are classified as either *external or internal.* The external branches are the branches connected directly to the registers. All other branches are classified as internal.

4.2. Calculation of Branch Delay

The minimum delay values obtained from the algorithm *Path_Delay* are insufficient to completely determine the delay of each branch of the clock distribution network due to two characteristics of the hierarchical representation:

1) The clock skew between sequentially adjacent registers may not depend on the delay of the internal branches. Consider, for example, the global data path R_1 to R_3 in Figure 5. These registers are driven by the same branching point, as depicted in Figure 11, meaning that the clock skew specifications are satisfied solely by the delay of the individual external branches driving R_1, R_2 and R_3.
2) The clock skew specifications can only be satisfied if the delay of some of the internal branches is known. As an example, the global data path formed by R_{15} to R_{20} contains three registers, R_{15} to R_{17}, driven by one branching point, while the remaining three registers, R_{18} to R_{20}, are driven by a second branching point. The clock skew between registers R_{17} and R_{18} can only be satisfied if the delay of the internal and external branches driving these

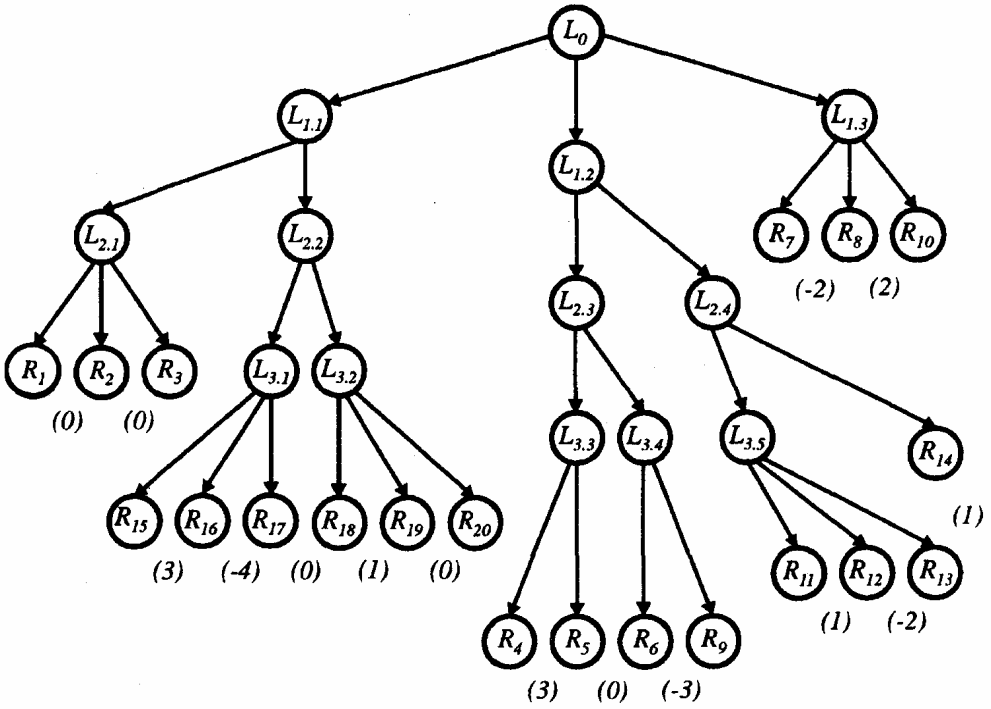

FIGURE 11 Hierarchical representation of the example circuit shown in Figure 5.

two registers is known. A similar situation exists between dependent global data paths belonging to separate hierarchical levels, such as the clock skew between registers R_6 and R_{12} (see Fig. 8)

Note that the first characteristic does not require that the delay of the internal branches be known, while the second case requires this information. Therefore, the individual delay of each of the external branches must be known, although the delay of the internal branches may not be known. Without placement information, the precise location of the registers is indeterminate, therefore it is not possible to know the precise delay of some of the internal branches. Since it is necessary to assign delay values to some internal branches, a constant unit value, called Δ, is chosen for this purpose. Once an estimate of the layout induced impedances are determined, more realistic delay values for the internal branches can be calculated to generate a more precise estimate of the path delays throughout the clock distribution network.

Delay of External Branches – When both registers within a local data path are driven by the same branching point, the clock skew specifications are completely satisfied by the delay values assigned to the external branches. These delay values are calculated using the algorithm *Path_Delay* and are expressed in absolute terms, provided that a value is assigned to the constant K. Since the delay of a branch cannot be zero, unit delay is assigned to K without loss of generality.

Delay of Internal Branches – It is possible, however, to have a global data path driven by more than one branching point. For example, the global data path R_{15} to R_{20} is driven by two separate branches of the clock distribution network. Starting with the common vertex that drives the data path (vertex $V_{L2.2}$), variables are assigned to each of the branches, as illustrated in Figure 12(a).

Clock delay equations are chosen to satisfy the clock skew specifications. The set of equations are

$$c - d = 3,$$
$$d - e = -4,$$
$$a + e = b + f,$$
$$f - g = 1,$$
$$g = h.$$

This system of equations has multiple solutions, since more unknowns exist than constraints. To obtain a solution, the delay of the external branches is initially calculated, providing a solution for variables c to h, respectively. Rewriting these equations, it is found that $a = b$. Attributing unit delay to these variables, a solution is found, symbolized by "[x]" and shown in Figure 12(b), which satisfies the original clock skew specifications.

Delay Equalization – Once the clock path delay has been determined for each register in the circuit, it is necessary to match the clock path delays between all the input and output registers. This step is performed by identifying the input/output register with the largest clock delay and equating

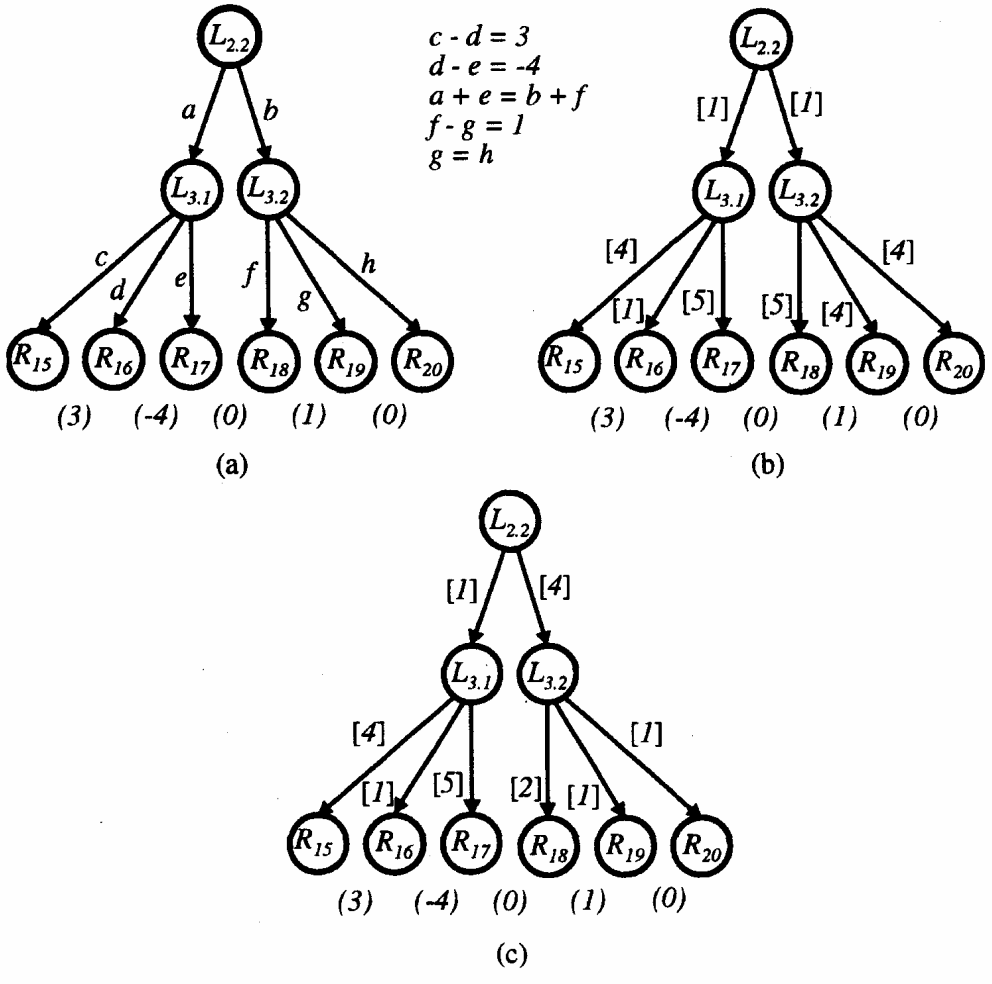

FIGURE 12 Calculation of the internal branch delays.

the clock delay of the other input/output registers with this value, without invalidating the optimal clock skew assignment determined previously. For the circuit example, it was assumed that the initial and final registers of each global data path are also the input and output registers of the circuit. In order to satisfy (6), the clock path delay driving the initial and final registers must be the same and equal to the largest clock path delay driving an I/O register. After the delay of the internal branches are determined, the clock path delay driving the I/O registers R_{15}, R_{20}, R_4, R_9, R_{11} and R_{14} is 7 tu, while the clock path delay driving the I/O registers R_1, R_3 is 3 tu and the I/O registers R_7, R_{10} is 2 tu. Therefore, the clock path delay of the registers R_1, R_3, R_7, and R_{10} must be increased to 7 tu.

Delay Shifting – It is possible to reorganize the delay of the external branches to reduce the total number of delay units needed to build the clock distribution network, thereby reducing die area. Consider, for example, Figure 12(b). The external branches connected to registers R_{18} to R_{20} may each have their branch delay reduced by three, if the delay of the internal branch driving each of these registers is increased by three. Another advantage of shifting the delay is the increased flexibility of the circuit implementation, since the delay can be shifted among branches to accommodate variations in the layout placement. An example of delay shifting is illustrated in Figure 12(c). Extending this procedure to the remaining branches of the clock tree depicted in Figure 11, it is possible to determine the minimum delay value of each branch of the clock distribution network, as illustrated in Figure 13.

4.3. Reorganization of the Clock Tree

The methodology for constructing the clock distribution network presented in this section assumes that the circuit netlist is described hierarchically. It is possible, however, to have

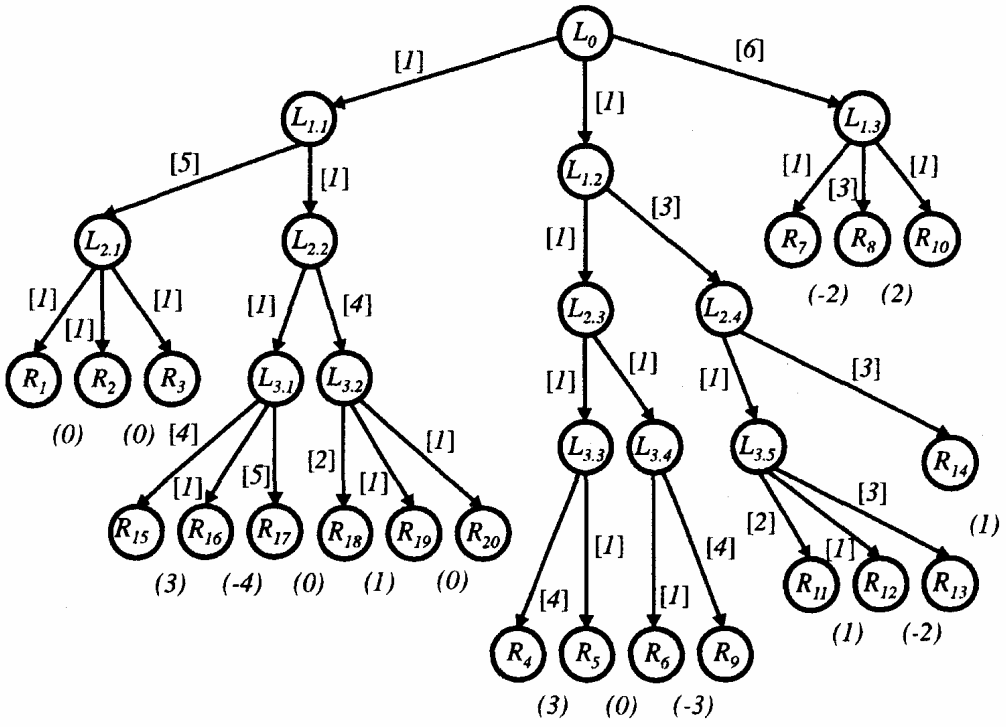

FIGURE 13 Delay assignment for each branch of the clock distribution tree.

circuit descriptions which are completely or partially flat. For those circuits described non-hierarchically, the result of Section 4.2 is a clock distribution network with independent clock paths driving each register, similar to that shown in Figure 10. In this section, a strategy for transforming these types of inefficient clock distribution networks into tree structures is presented. As an example, the clock distribution network for the flat representation of the circuit in Figure 5 is transformed into a clock tree with several levels of hierarchy, as illustrated in Figure 14.

Without regard to placement information, the clock distribution tree can be built by placing the shorter delay clock paths in branches close to the clock source and the longer delay clock paths in branches farther from the clock source. For this purpose, the branch with the longest delay is partitioned into a series of segments, where each segment emulates a precise quantified delay value, Δ. Between any two segments, there is a branching point to other registers or sub-trees of the clock tree, where several segments with delays greater or equal to Δ are cascaded to provide the appropriate final delay at each leaf node. The result of this approach is illustrated in Figure 14, where a significant saving of segment delays, as compared to Figure 13, is obtained (from 67 Δ to 31Δ).

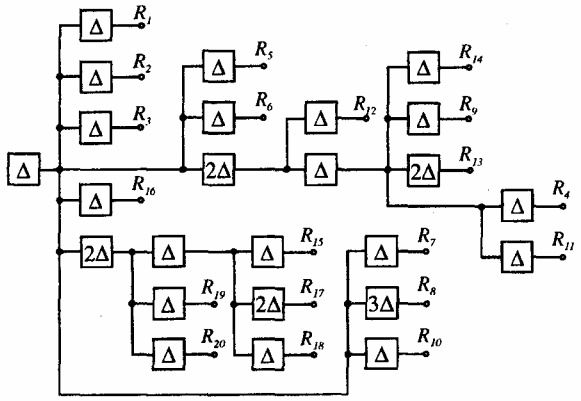

FIGURE 14 Delay assignment of non-hierarchically defined clock distribution network.

A further area improvement can be obtained by reorganizing the clock distribution network with the assignment of a separate sub-tree for each global data path that does not contain any register in common with any other global data path. For example, the circuit in Figure 8 has four global data paths with this characteristic, two of them being the global data path formed by the registers R_1 to R_3 and the global data path formed by the registers R_{15} to R_{20}. The algorithm *Form_Units*, summarized in Figure 15, is used to determine those global data paths that contain common registers. This result is accomplished by comparing the set of registers of two global data paths. If the result of the comparison is that two global data paths are not independent, both sets are grouped together into a new set, formed by the union of the registers of the two global data paths. For a module with n global data paths, the algorithm is $O(n)$ in the best case, where all the global data paths have registers in common, and is $O(n^2)$ in the worst case, where all the global data paths have no registers in common.

5. DESIGN OF CIRCUIT DELAY ELEMENTS

The delay of the circuit structures that emulates the delay values associated with each branch of the network requires high precision, because variations in the delays of the internal branches are

```
Algorithm Form_Units(M)
  gdp_set = all global data paths module M;
  unit_set = ∅;
  while gdp_set not ∅ do
    unit_i = first gdp in gdp_set;
    for each gpd_j in gdp_set do
      if( register of gdp_j ∈ unit_i )
        unit_i = unit_i ∪ gpd_j;
        gdp_set = gdp_set - gdp_j;
end Form_Units
```

FIGURE 15 Algorithm to group global data paths.

propagated throughout the network, causing unacceptable variations in the desired clock skew. It is important to note that it is much more difficult and important to satisfy the clock skew *between* any two clock paths rather than to satisfy each individual clock path delay.

Delay elements can either be composed of passive or active circuit elements. Implementing the delay elements within the clock distribution network as a passive RC network is unacceptable for several reasons; 1) the delay of each branch is highly dependent on the delay of every other branch, 2) the clock signal waveform would degrade, limiting system performance and reliability, 3) an accurate delay model of a passive clock distribution network for a circuit with thousands of registers is difficult to obtain, and 4) the layout of the passive RC network is highly sensitive to small variations in position or length of the clock lines, producing unacceptable variations in the localized clock skew. Therefore, two criteria must be met to successfully design a clock distribution network. First, the delay of each branch must be implemented such that each branch is independent of the delay of the other branches. Second, the clock branches must be designed such that there are no physical layout constraints difficult to implement.

To satisfy both criteria, the strategy adopted is to implement the delay segments with active elements, specifically CMOS inverters. Due to the high input impedance of a CMOS inverter, the inverter effectively isolates each clock branch from each other. Additionally, the interconnect lines are constrained to behave as purely capacitive lines by properly inserting these distributed CMOS inverters as repeaters along the clock signal path [15]. The insertion points are chosen such that the output impedance of each inverter is much greater than the resistance of that portion of the interconnect section being driven. This strategy permits the length of a single interconnect line to be accurately modeled as a lumped capacitance with negligible resistance. However, the strategy also places a maximum constraint on the length of an individual portion of an interconnect line between inverting repeaters, thereby limiting the physical placement of a clock branch within the circuit layout. The maximum interconnect length for two CMOS processes is illustrated in Appendix A, describing the limiting length a distributed RC line can be accurately modeled as a lumped capacitor.

5.1. Preserving Clock Signal Polarity

Using a single inverter to produce a specific branch delay may invert the polarity of the clock signal for those clock paths consisting of an odd number of branches. To maintain the proper signal polarity, the tree structured graph representing the topology of the clock distribution network is searched to identify those branches that require two inverters, ensuring that the number of inverters from the clock source to every register remains even (or odd) while utilizing a minimum number of inverters.

The number of internal and external branches in a clock path is called the *depth* of the path, where the depth of each clock path is obtained by traversing the graph representing the clock tree [17]. If the depth is even, the polarity is preserved and a single inverter is assigned to each branch of the clock path. If the depth is odd, an extra inverter must be inserted to preserve the signal polarity of the clock signal. The choice of which branch should receive the extra inverter is important in determining the minimum number of inverters and for preserving the polarity of the clock paths with an even number of branches. The example circuit illustrated in Figure 4 is used in Figure 16, where the number inside each leaf node represents the depth of the clock signal path at that branching point. As shown in Figure 16, the clock paths that require an extra inverter are those branches which drive R_1, R_2, R_3 and R_{14}, since these paths have an odd number of branches. The procedure to obtain the minimum number of inverters to implement the branch delay of the clock distribution network is presented below and illustrated in Figure 16.

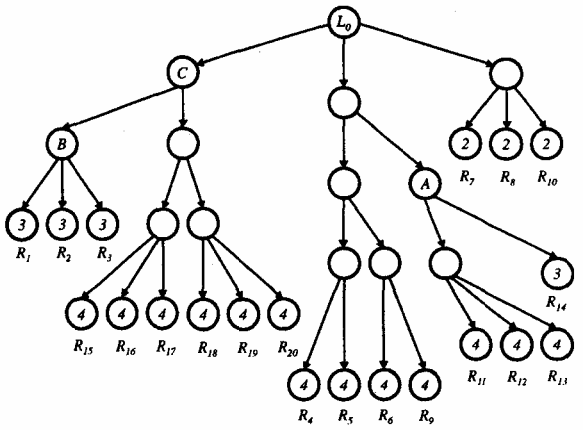

FIGURE 16 Inverter assignment of the branches of the clock distribution network.

Buffer Assignment Procedure – The assignment procedure begins at the leaf node of the clock path with odd depth and proceeds backwards, one node at a time towards the root node, until the branch that is assigned the extra inverter has been determined or the root node is reached. Starting at a leaf node and moving back one node in the path, the depth of each clock path connected to that node is determined, excluding the path being analyzed. If at least one depth value is even, two inverters are assigned to the branch in the present path. If all the depths are odd, only one inverter is assigned to the branch, and another node in the path is chosen. The inverter assignment procedure continues until every clock path with odd depth has been searched and an extra inverter is allocated to that clock signal path.

Consider, for example, the clock path driving register R_{14} in Figure 16. Going backwards one node in the clock path, node A is reached. Determining the depth of all the leaf nodes directly or indirectly driven by node A requires determining the depth of the clock paths driving registers R_{11}, R_{12} and R_{13}. Since the depth of these nodes is even, two inverters are assigned to the branch between node A and the leaf node driving register R_{14}. Observe that at this point the procedure terminates searching the clock path driving register R_{14}, because the number of buffers driving the path is now even, and begins searching another clock path with odd depth. Consider the clock path driving register R_1, also illustrated in Figure 16. Moving backwards one node in the clock path, node B is reached. The depth of the leaf nodes directly or indirectly connected to node B is odd. Therefore, one inverter is assigned to the branch driving register R_1 and the search advances one node in the clock path, moving to node C. Evaluating the depth of all the nodes connected to node C, it is determined that the depth of the nodes driving registers R_{15} through R_{20} is even, and therefore two inverters are assigned to the branch between nodes C and B. Observe that solving the inverter assignment problem for the clock path driving register R_1 immediately solves the signal polarity problem for the clock paths driving registers R_2 and R_3.

5.2. Implementation of Clock Path Delay

A clock tree is composed of many clock signal paths (branches) each driving a storage element (leaf). Clock signal paths may have branches in common, depending on the topology of the clock tree. The total delay of a clock path t_{cpd} is the summation of the delays of each individual branch along the clock path τ_{bi},

$$t_{cpd} = \sum_{i=1}^{n} \tau_{bi}. \tag{8}$$

In order to accurately sum the individual delay components along a clock signal path, three issues are of importance: 1) isolation of each branch delay; 2) integration of the inverter and interconnect delay models used to calculate the delay of each clock path, and 3) integration of the waveform shape into the inverter delay model.

The capacitive load at the output of the inverting buffer includes both the capacitance of a single interconnect line and the fanout of each node of the clock distribution network. Therefore, the output capacitive load is composed of the

capacitance of an interconnect line plus the input capacitance of each inverter driven by that interconnect line. Multiple inverters being driven by the same inverter occur at the branching points within the clock distribution network. The input gate capacitance of an inverter is modeled as $C_g = C_{ox}WL$, where C_{ox} is the oxide capacitance per unit area and is dependent on the thickness and permitivity of the oxide. Therefore, the load capacitance seen by an inverter is given by

$$C_L = C_{\text{line}} + \#\text{ of branches} * C_g, \qquad (9)$$

assuming that the inverters are of equal size. If the inverters are not of equal size, the second term in (9) becomes a summation among all the branches connected to the driving point.

Due to the high input impedance of a CMOS inverter, the inverter effectively isolates each clock branch from the following branch. Furthermore, each branch can be implemented with one or more repeaters, such that the output impedance of each repeater is much greater than the resistance of the driven interconnect section [15], as illustrated in Figure 17. Under this constraint, each inverter drives a load that can be modeled as a lumped capacitor composed of the capacitance of an interconnect line plus the input capacitance of each inverter connected to that interconnect line, as described in Appendix A. As a consequence of the output impedance of the repeater buffer being much greater than the resistance of the interconnect section, the slope of the input signal of an inverter connected to a branching point can be approximated as the slope of the output signal of the inverter connected to that branching point [10].

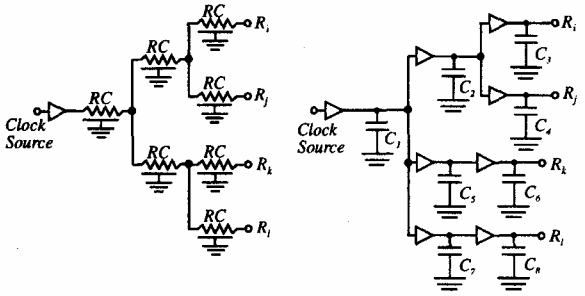

FIGURE 17 Modeling an RC network with buffers and lumped capacitors.

Under these conditions, the slope of the output waveform of the driving inverter is equal to the slope of the input waveform of the driven inverter(s). The slope of the input/output waveform can be characterized by two parameters, the geometry and the capacitive output of the driving inverter. Since the capacitive load includes the interconnect line capacitance and the input gate capacitance of each driven inverter, the transistor size and output load of each branch are each determined such that the branch delay and the total delay of a clock path are both satisfied.

5.3. Branch Delay Modeling

The delay value assigned to each branch of the clock distribution network during the topological design phase is implemented with one or two CMOS inverters, according to the assignment procedure described in Section 5.1. The delay equations describing an inverting repeater, shown in (10)–(12), are used to determine the geometric dimensions of the transistors and are based on the MOSFET I-V α-power law I-V model developed by Sakurai and Newton [18].

$$I_{DS} = \begin{cases} 0 & (V_{GS} \le V_{th} \quad : \text{cutoff}) \\ \frac{I'_{DS}}{V'_{DS}} & (V_{DS} < V'_{DS} \quad : \text{linear}) \\ I'_{DS} & (V_{DS} \ge V'_{DS} \quad : \text{saturation}), \end{cases} \qquad (10)$$

where

$$I'_{DS} = I_{DS}\left(\frac{V_{DS} - V_{th}}{V_{DD} - V_{th}}^{\alpha}\right) = \frac{W}{L_{off}} P_C (V_{GS} - V_{th})^{\alpha}, \qquad (11)$$

$$I'_{DS} = V_{DS}\left(\frac{V_{DS} - V_{th}}{V_{DD} - V_{th}}^{\alpha/2}\right) = \frac{W}{L_{off}} P_V (V_{GS} - V_{th})^{\alpha/2}, \qquad (12)$$

and where I_{DO} is the drain current at $V_{GS}=V_{DS}=V_{DD}$, V_{DO} is the drain saturation voltage at $V_{GS}=V_{DD}$, V_{th} is the threshold voltage, α is the velocity saturation index, and V_{DD} is the power supply. The parameters α, V_{DO}, I_{DO}, and V_{th} are determined as explained in [18]. The inverter is driven by a ramp signal with rising and falling slopes, s_r and s_f, respectively,

$$V_{in}=\begin{cases} s_r t & \text{rising input} \\ V_{DD}-s_f t & \text{falling input.} \end{cases} \tag{13}$$

The slope $s_r(s_f)$ is selected such that during discharge (charge), the effects of the PMOS (NMOS) transistor can be neglected [19]. This assumption is possible for fast input ramps, which occur when the crossing point between the input and output waveforms is approximately $|V_{DD}-V_{th}|$ [19]. The structure of the delay element is illustrated in Figure 18a, with the shape of the input and output waveforms illustrated in Figure 18b.

The time t_d from the 50% point of the input waveform to the 50% point of the output waveform is defined as the delay of the circuit element. Equation (14) is derived from (10)–(12) [18] and describes the load capacitance C_L of a CMOS inverter in terms of its delay,

$$C_L=\frac{2I_{DO}}{V_{DD}}\left[t_d-\left(\frac{1}{2}-\frac{1-\nu_T}{1+\alpha}\right)s_r\right], \quad \text{where} \quad \nu_T=\frac{V_{th}}{V_{DD}} \tag{14}$$

The output waveform of the driving inverter is the input signal to all branches connected to this inverter and is approximated by a ramp shaped waveform. This approximation is achieved by linearizing the output waveform (from $0.1V_{DD}$ to $0.9V_{DD}$) and is accurate as long as the interconnect resistance is negligible as compared to the inverter output impedance. The slope of the output waveform is expressed as

$$s_r=\frac{t_{0.9}-t_{0.1}}{0.8}=\frac{C_L V_{DD}}{I_{DO}}\left(\frac{0.9}{0.8}+\frac{V_{DO}}{0.8V_{DD}}\ln\frac{10V_{DO}}{eV_{DD}}\right). \tag{15}$$

Equations (14) and (15) provide the necessary relationships to design the circuit elements of a clock signal path, as explained below:

Design of clock signal path – For each clock path within the clock tree, the procedure to design the CMOS inverters is as follows: 1) the load of the initial trunk of the clock tree is determined from

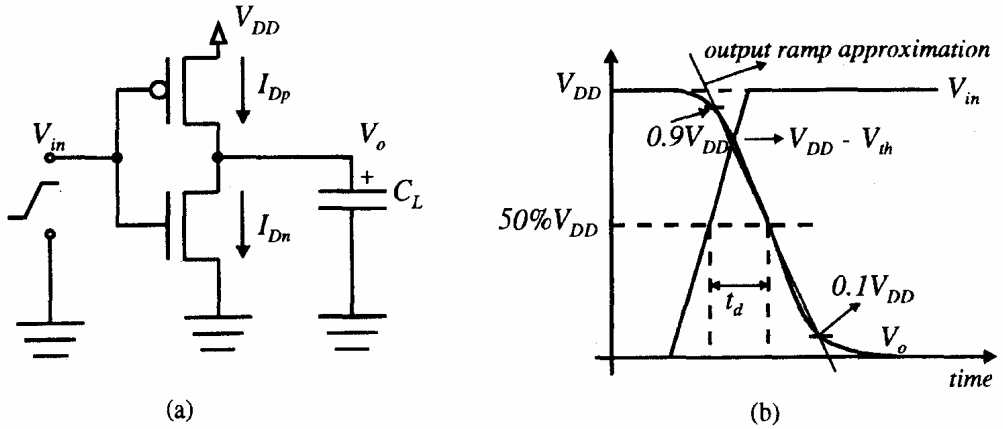

FIGURE 18 (a) Delay element; (b) Input/output waveforms of the delay element.

(14), assuming a step input clock signal; 2) the slope of the output signal is calculated from (15); 3) this value is applied in (14) to determine the load of the following branch, and (15) is used again to determine the slope of the output signal; and 4) step 3 is repeated for each subsequent branch of the clock path. Steps 1–4 are applied to the remaining clock paths within the clock tree. Observe that if the slope of the output signal of branch b_i does not satisfy

$$s_{ri} \leq \frac{1}{\left(\frac{1}{2} - \frac{1-\nu_T}{1+\alpha}\right)} \left(t_{di+1} - \frac{V_{DD} C_{Li+1}}{2 I_{DO}} \right), \qquad (16)$$

(14) is no longer valid. The slope s_{ri} can be reduced by increasing the output current drive of the inverter in branch b_i. However, from (14), increasing I_{DOi} would increase the capacitive load C_{Li} in order to obtain the desired propagation delay t_{di} for branch b_i with a single inverter. Therefore, if the propagation delay t_{di} of the branch b_i is too large, the slope associated with branch b_i can only be reduced if the branch b_i is implemented with more than one inverter. The number of inverters required to implement the propagation delay t_{di} is chosen such that (16) is satisfied for each inverter stage and the polarity of the clock signal driving branch b_{i+1} does not change.

Equations (10) and (14)–(16) are sufficient to determine the buffer geometry and load capacitance of each inverter along every branch of the clock distribution network. As an example, the clock path driving registers R_6 and R_{12} are illustrated in Figure 19, assuming CMOS inverters with a W_P/W_N ratio equal to two. The per cent difference

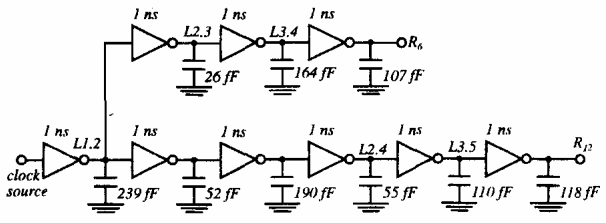

FIGURE 19 Example of clock signal paths driving registers R_6 and R_{12}.

between the analytically derived delay obtained from the topological design and implemented as circuit delay elements and the simulated delay derived from SPICE is 1.5% for the path driving R_6 and 2.3% for the path driving R_{12}. Furthermore, the per cent difference for the clock skew between the clock signals driving both registers is 4.0%.

6. SIMULATION RESULTS

The efficiency of this top-down design methodology has been verified by applying the first three phases of the methodology to a number of example circuits. In the clock skew scheduling and topological phases, the clock tree is determined in terms of the minimum clock path delay and the number of delay units for both the hierarchical and non-hierarchical implementations. In the circuit design phase, the accuracy of implementing the clock path delays and clock skew is compared with SPICE simulations.

Certain features of the topological phase are illustrated in Table III for several circuit examples. The circuit illustrated in Figure 8 is example *cir*1, cited in Table III. The second example, *cir*2, illustrates a clock distribution network in which each of the registers of a global data path are interconnected. The final circuits, *cir*3 and *cir*4, exemplify the effects of having a large number of registers driven by the same branching point.

The second column in Table III shows the number of registers within each circuit. The third and fourth columns depict the maximum clock delay of the clock distribution network based on a non-zero clock skew schedule, without and with hierarchy. The final two columns describe the total number of delay units required to implement the circuit without and with using the hierarchical information of the circuit. Although the minimum clock delay is always less for a clock distribution network with individualized clock paths driving each register, it does not reflect the reality of current VLSI circuits with many thousands of registers, requiring excessive area to implement

TABLE III Minimum clock delay and total number of delay units for several example circuits

Circuit	Registers	Minimum clock delay		Total delay units	
		No hierarcy	Hierarchy	No hierarchy	Hierarchy
cir 1	20	5	8	56	57
cir 2	15	7.2	9.2	57	53
cir 3	56	11.1	13.2	184	115
cir 4	37	8	10.6	163	124

such a clock network. On the other hand, due to branch sharing between common clock signal paths, less layout area is required in tree structured clock distribution networks since the branch delays are smaller.

The difference between the calculated and simulated clock path delays for the circuit shown in Figure 13 is compared in Table IV. The second column depicts the target delay obtained from the topological and circuit design of the clock distribution network. The third column shows the delay values of each clock path derived from SPICE circuit simulation using Level-3 device models, while the fourth column depicts the per cent error between the calculated and the numerically derived delay. Note that the maximum error for these examples is 2.5%.

The delays of the clock paths in Table IV corroborate two very important characteristics of the delay model presented herein. Keeping the interconnect resistance small as compared with the output impedance of a buffer guarantees that each branch delay can be modeled independently. Also, the accuracy of the total delay of a clock path is greatly enhanced by integrating an interconnect delay model into an inverter delay model by including the effects of input/output waveform shape and the input capacitance of all the branches connected to the output of an inverter. For example, the clock path driving register R_{15} shares branches with the clock paths driving registers R_1 and R_{19}. Nevertheless, the maximum error between the calculated and simulated delays is 2.5%. Likewise, the clock path that drives R_6 has branches in common with the clock paths that drive registers R_{12}, R_{14} and R_4, exhibiting an error less than 2%.

A more significant measure of the effectiveness of the clock distribution network design methodology presented in this paper is to guarantee that the clock skew between any pair of registers in the same global data path is accurately implemented, rather than the delay of each individual clock path. The clock skew between registers for the circuit illustrated in Figure 4 is listed in Table V. The scheduled clock skew implemented with the design methodology described in this paper for the pair of registers presented in column one is shown in column two. The values obtained from SPICE circuit simulation are listed in column three. While

TABLE IV Comparison between calculated and simulated clock path delay

Clock Path	Delay (ns)	SPICE (ns)	Error
R_1, R_2, R_3	7.0	6.84	2.3%
R_{15}, R_4, R_9	7.0	7.11	1.6%
R_{18}	8.0	8.10	1.3%
R_{19}, R_{20}	7.0	7.04	0.6%
R_5, R_6, R_{16}	4.0	4.06	1.5%
R_{17}	8.0	8.06	0.8%
R_{14}	7.0	7.17	2.4%
R_{11}	7.0	7.16	2.3%
R_{12}	6.0	6.14	2.3%
R_{13}	8.0	8.20	2.5%
R_7R_{10}	7.0	6.90	1.4%
R_8	9.0	8.90	1.1%

TABLE V Comparison between specified and simulated clock skew values

Registers	Specified Skew (ns)	Simulated (ns)	Error
R_1-R_3	0.0	0.0	0.0%
$R_{15}-R_{16}$	3.0	3.0	0.0%
$R_{12}-R_{13}$	−2.0	−2.6	3.0%
$R_{17}-R_{19}$	1.0	1.03	3.0%
R_4-R_{14}	0.0	0.06	–
R_6-R_{12}	−2.0	−2.8	4.0%
R_5-R_{13}	−4.0	−4.14	3.5%

the per cent error between the targeted and simulated values are shown in column four. Note that the maximum error is 4%, a number well within practical and useful limits.

The individual data paths have been selected to illustrate several types of clock skew situations, such as non-zero clock skew between registers in the same data path or in separate data paths. More specifically, zero clock skew between registers in the same data path is illustrated by the path between registers R_1 and R_3. The path between registers R_{15} and R_{16} illustrates positive clock skew for registers in the same data path, while the path between registers R_{12} and R_{13} illustrates negative clock skew for registers in the same data path. In these three examples, the clock skew is only dependent upon the delay of the external branches, and therefore these clock paths are independent of the delay variations of the internal branches of the clock distribution network. The following examples in Table V are more illustrative of the possible effects of variations of the internal branch delays within the clock path. The path between registers R_{17} and R_{19} illustrate non-zero clock skew in a data path with feedback which is dependent on the delay of its internal branches. The last three examples in Table V illustrate the clock skew between two registers belonging to interconnected data paths. In the first example, a new path is formed between registers R_4 and R_{14} due to the connection between registers R_6 and R_{12}. The final example also illustrates the effect of negative clock skew on two registers, R_5 and R_{13}, in this path. Observe that in the last examples, the error tolerance of the clock skew is still within acceptable margins, exhibiting good accuracy, even for those paths in which the clock paths driving two sequentially adjacent registers do not share a significant portion of the clock distribution network.

7. CONCLUSIONS

VLSI/ULSI-based synchronous systems require the efficient synthesis of high speed clock distribution networks in order to obtain higher levels of circuit performance and improved design turnaround. In this paper, circuit performance and reliability are improved by using non-zero localized clock skew to reduce the minimum clock period and ensure that no race conditions occur. An integrated top-down methodology is presented for synthesizing clock distribution networks. This methodology is composed of four phases, 1) optimal clock scheduling, 2) topological design of the clock distribution network, 3) design and modeling of the circuit delay elements, and 4) layout implementation. In this paper, clock scheduling, topological design, and the design and modeling of the circuit delay elements are presented. The fourth phase, layout implementation, is a well developed design technology and is therefore not discussed in this paper.

The optimal clock schedule, described in terms of the non-zero clock skew between each pair of sequentially adjacent registers, is derived from the circuit timing information, namely, the delay of the combinational logic, interconnect, and registers. The topology of the clock distribution network is a tree structure derived from the hierarchy of the original circuit description. Minimum delay values are assigned to each branch of the clock network, satisfying a specific clock skew schedule. An approach for developing the topology of those clock distribution networks for circuits with minimal or no hierarchical information is also described.

A strategy for choosing and implementing the branch delay values of the clock distribution network is presented, using CMOS inverters for the delay elements. The minimum number of inverters to satisfy the branch delay is obtained, while preserving the polarity of the signal driving the clock input of each register. Delay equations of an inverter, derived from the α-power law I-V model, are described. The inverter delay model accurately determines the delay of each clock path by considering the fanout, interconnect capacitance, and the slope of the input and output waveforms of each branch along the clock path. Comparisons between expected and simulated

delays of each individual clock signal path produces circuits with a maximum error of less than 2.5%. Furthermore, the maximum error between scheduled and simulated clock skew for any two registers belonging to the same global data path is 4%.

Thus, an integrated methodology for synthesizing tree-structured clock distribution networks for high performance VLSI circuits is presented. This methodology, based on inserted delay elements, accurately synthesizes localized clock skews. Furthermore, this methodology utilizes non-zero clock skew to improve overall system performance and reliability.

Acknowledgement

This research was supported in part by Grant 200484/89.3 from CNP_q (Conselho Nacional de Desenvolvimento Cientifico e Tecnológico-Brasil), by the National Science Foundation under Grant No. MIP-9208165 and Grant No. MIP-9423886, by the U. S. Army Research Office under Grant No. DAAH04-93-G-0323, and by a grant from the Xerox Corporation.

References

[1] Pullela, S., Menezes, N., Omar, J. and Pillage, L. T. (November 1993). "Skew and Delay Optimization for Reliable Buffered Clock Trees," *Proceedings of the IEEE International Conference on Computer-Aided Design*, pp. 556–562.

[2] Bakoglu, H. B., Walker, J. T. and Meindl, J. D. (October 1986). "A Symmetric Clock-Distribution Tree and Optimized High-Speed Interconnections for Reduced Clock Skew in ULSI and WSI Circuits,"*Proceedings of the IEEE International Conference on Computer Design*, pp. 118–122.

[3] Chao, T.-H, Hsu, Y.-C, Ho, J.-M, Boese, K. D. and Kahng, A. B. (November 1992). "Zero Skew Clock Routing with Minimum Wirelength,"*IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, Vol. CAS-**39**(11), pp. 799–814.

[4] Tsay, R.-S. (February 1993). "An Exact Zero-Skew Clock Routing Algorithm," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. CAD-**12**(2), pp. 242–249.

[5] Friedman, E. G. (June 1989). *Performance Limitations in Synchronous Digital Systems*, Ph. D. Dissertation, University of California, Irvine, California.

[6] Fishburn, J. P. (July 1990). "Clock Skew Optimization," *IEEE Transactions on Computers*, Vol. C-**39**(7), pp. 945–951.

[7] Sakallah, K. A., Mudge, T. N. and Olukotun, O. A. (June 1990). "*checkTc and minTc:* Timing Verification and Optimal Clocking of Synchronous Digital Circuits," *Proceedings of the IEEE/ACM Design Automation Conference* pp. 111–117.

[8] Szymanski, T. G. (June 1992). "Computing Optimal Clock Schedules," *Proceedings of the IEEE/ACM Design Automation Conference*, pp. 399–404.

[9] Neves, J. L. and Friedman, E. G. (August 1993). "Topological Design of Clock Distribution Networks Based on Non-Zero Clock Skew Specifications," *Proceedings of the IEEE Midwest Symposium on Circuits and Systems*, pp. 461–471.

[10] Neves, J. L. and Friedman, E. G. (May 1994). "Circuit Synthesis of Clock Distribution Networks based on Non-Zero Clock Skew," *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 4.175–4.178.

[11] Gray, C. T. (July 1993). *Optimal Clocking of Wave Pipelined Systems and CMOS Applications*, Ph. D. Dissertation, North Caroline State University, Raleigh, North Carolina.

[12] Papadimitriou, C. H. and Steiglitz, K. (1982). *Combinatorial Optimization Algorithms and Complexity*, Prentice-Hall.

[13] Friedman, E. G. (1995). *Clock Distribution Networks in VLSI Circuits and Systems*, IEEE Press.

[14] Friedman, E. G., Mulligan, J. H. Jr. (April 1991). "Clock Frequency and Latency in Synchronous Systems," *IEEE Transactions on Signal Processing*, Vol. **SP-39**, pp. 930–934.

[15] Dhar, S. and Franklin, M. A. (January 1991). "Optimum Buffer Circuits for Driving Long Uniform Lines," *IEEE Journal of Solid State Circuits*, Vol. SC-**26**(1), pp. 32–40.

[16] Sakurai, T. (January 1993). "Closed-Form Expressions for Interconnection Delay, Coupling, and Crosstalk in VLSI's," *IEEE Transactions on Electron Devices*, Vol. ED-**40**(1), pp. 118–124.

[17] Tremblay, J.-P. and Sorenson, P. G. (1986). *An Introduction to Data Structures with Applications*, 2nd Edition, McGraw-Hill.

[18] Sakurai, T. and Newton, A. R. (April 1990). "Alpha-Power Law MOSFET Model and its Applications to CMOS Inverter Delay and Other Formulas," *IEEE Journal of Solid State Circuits*, Vol. SC-**25**(2), pp. 584–594.

[19] Hedenstierna, N. and Jeppson, K. O. (March 1987). "CMOS Circuit Speed and Buffer Optimization", *IEEE Transactions on Computer-Aided Design*, Vol. CAD-**6**(2), pp. 270–281.

[20] Wyatt, J. L. Jr. (1987). "Signal Propagation Delay in RC Models for Interconnect," in *Circut Analysis, Simulation and Design*, Chapter 11, Elsevier Publishers.

[21] Chern, J.-H., Huang, J., Arledge, L., Li, P.-C. and Yang, P. (January 1992). "Multilevel Metal Capacitance Models for CAD Design Synthesis Systems," *IEEE Electron Device Letters*, Vol. EDL-**13**(1), pp. 32–34,

[22] Liu, S. and Nagel, L. W. (December 1982). "Small-Signal MOSFET Models for Analog Circuit Design," *IEEE Journal of Solid State Circuits*, Vol. SC-**17**(6), pp. 983–998.

[23] Sakurai, T. (August 1983). "Approximation of Wiring Delay in MOSFET LSI," *IEEE Journal of Solid State Circuits*, Vol. SC-18(4), pp. 418–426.

APPENDIX A

An interconnect line can be treated as a lumped capacitance only when the resistive impedance of the buffer driving the interconnect line is much greater than the resistance of the interconnect line [15]. Since the resistance of an interconnect line is proportional to the length of the line, a restriction is imposed on the maximum line length in order to ensure the line resistance is of negligible magnitude.

To exemplify how the resistive impedance of an interconnect line is maintained smaller than the output impedance of the inverting buffer driving that node, the interconnect length for several capacitive loads, given a target fabrication process, is analyzed. Consider, for simplicity, that the delay model is a single pole model, as given in [20], and the delay is measured at the 50% point of the output signal. The delay is

$$T_{di} = 0.693\left(\sum_{m=1}^{i}\sum_{n=m}^{i} R_m, C_n\right), \qquad \text{(A.1)}$$

where R_m and C_n are the parasitic resistance and capacitance of the interconnect line, respectively. The interconnect resistance is

$$R_{\text{int}} = R * \#\text{ of }\square\text{'s}. \qquad \text{(A.2)}$$

where $R_\square$ is the resistance per square of the interconnect line and # of □'s is the number of squares of interconnect. The capacitance of an isolated interconnect line consists primarily of two components, the parallel plate capacitance and the sidewall fringing capacitance. In submicrometer technology, the fringing capacitance cannot be neglected, and therefore the total capacitance can be modeled as [21]

$$C_{\text{int}} = \frac{C}{A} * W_{\text{int}} L_{\text{int}} + C_{\text{fr}} * 2(W_{\text{int}} + L_{\text{int}}), \qquad \text{(A.3)}$$

where C/A is the parallel plate capacitance per unit area, C_{fr} is the fringing capacitance per unit length, and W_{int} and L_{int} are the width and length of the interconnect line, respectively. For the purpose of calculating the interconnect length, given a desired interconnect capacitance, (A.3) can be rewritten as

$$L_{\text{int}} = \frac{C_{\text{int}} - 2C_{\text{fr}} * W_{\text{int}}}{\frac{C}{A} * W_{\text{int}} + C_{\text{fr}} * 2}. \qquad \text{(A.4)}$$

The output impedance of an inverter is [22]

$$R_{\text{eff}} = \frac{2L_{\text{TR}}}{W_{\text{TR}}\mu C_{\text{ox}}(V_{dd} - V_{\text{th}})}, \qquad \text{(A.5)}$$

where W_{TR} and L_{TR} are the width and length of the MOS transistor, respectively, C_{ox} is the gate oxide capacitance, μ is the channel mobility, V_{th} is the threshold voltage, V_{DD} is the power supply, and the transistor is assumed to operate in the saturation region.

Equations (A.1), (A.2) and (A.4) are used in Table A.I to estimate the interconnect length and resistance for several values of interconnect capacitance. For a value of interconnect capacitance (Column two in Tab. A.I), the length of a minimum width (W_{Int}) interconnect line is obtained from (A.4) and shown in column three. In column four, the interconnect resistance is calculated from (A.2), and the percentage ratio between the interconnect resistance and the output impedance of the buffer is presented in column five.

TABLE A.I Determination of the length of a single interconnect line

	C_{int}(fF)	L_{int}(μm)	R_{int} (Ω)	$R_{\text{int}}/R_{\text{eff}}$
2 μm CMOS	50	520	8.3	0.14%
R_{eff} = 5.8 K	100	1041	16.6	0.29%
$R_\square$ = 0.48 Ω	150	1562	25.0	0.43%
C = 0.32 fF/m^2	250	2604	41.6	0.72%
C_{fr} = 0 fF	350	3645	58.3	1.01%
W_{Int} = 3.0 μm	500	5208	83.3	1.44%
1 μm CMOS	50	349	12.8	0.43%
R_{eff} = 3.0 K	100	700	25.7	0.86%
$R_\square$ = .055 Ω	150	1051	38.5	1.28%
C = .035 fF/m^2	250	1753	64.3	2.14%
C_{fr} = 0.45 fF	350	2455	90.0	3.0%
W_{Int} = 1.5 μm	500	3562	128.6	4.27%

TABLE A.II Propagation delay of a single interconnect line with variable load

	C_{int}(fF)	D_1(ns)	D_2(ns)	Error
2 μm CMOS	50	0.29	0.29	0.0%
R_{eff} = 5.8 K	100	0.58	0.58	0.0%
$R_{\square}$ = 0.48 ω	150	0.86	0.87	<1%
C = 0.32 fF	250	1.44	1.45	<1%
C_{fr} = 0 fF	350	2.02	2.04	1.0%
W_{Int} = 3.0μm	500	2.88	2.92	1.5%
1 μm CMOS	50	0.15	0.15	0.0%
R_{eff} = 3.0 K	100	0.30	0.30	0.0%
$R_{\square}$ = .055 ω	150	0.45	0.46	2.1%
C = .035 fF	250	0.75	0.77	2.6%
C_{fr} = .045 fF	350	1.05	1.08	2.8%
W_{Int} = 1.5 μm	500	1.50	1.56	3.8%

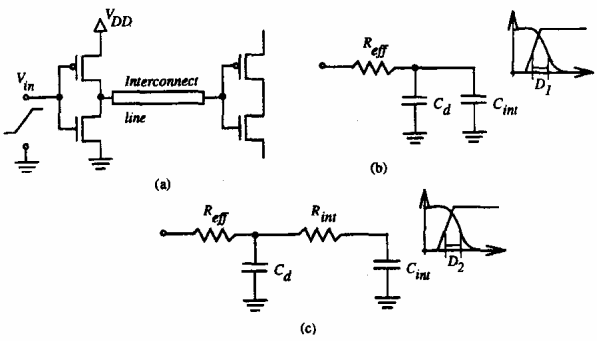

FIGURE A.1 Propagation delay of a single interconnect line. (a) Schematic diagram; (b) delay model without interconnect resistance: (c) delay model considering interconnect resistance.

From Table A.I it is noted that by including fringing capacitance in the calculation of the interconnect capacitance, the maximum interconnect length of the 1 μm process as compared to the 2 μm process is significantly reduced. In Table A.I, it is also shown that as the feature size is reduced, the interconnect resistance increases, reducing the length of interconnect which can be accurately modeled as a lumped capacitor.

The values of resistance and capacitance, illustrated in Table A.I, are used in Table A.II to obtain the signal propagation delay of an inverter driving an interconnect line. In Column two, the calculated propagation delay of an inverter, (D_1), derived from (A.1), driving an interconnect line without considering interconnect resistance is described, (illustrated in Fig. A.1b). The effective resistance of a minimum size inverter, R_{eff}, is calculated from (A.5). Using the same values of interconnect capacitance that are used in Table A.I, the propagation delay through an interconnect line considering interconnect resistance (D_2) is illustrated in Figure A.1c and presented in column four. The per cent error between both calculations is listed in column five.

As shown in column five in Table A.II, the interconnect resistance begins to affect the propagation delay for load capacitance values of approximately 150 fF, which correspond to interconnect lengths of approximately 1500 μm for a 2 μm process and 1000 μm for a 1 μm process, as shown in column three of Table A.I. Interconnect line lengths significantly greater than these values require the insertion of additional inverting buffers in order to preserve the accuracy of the branch delay and to minimize any significant signal degradation.

Authors' Biographies

José Luis Neves received the B.S. degree in Electrical Engineering and the M.S. degree in Computer Science from the Federal University of Minas Gerais, Brazil in 1986 and 1990, respectively. He also received the M.S. degree in Electrical Engineering from the University of Rochester in 1991. He completed the Ph.D. degree at the University of Rochester in 1995. Dr. Neves is currently employed at IBM microelectronic, in East Fishkill, New York working on the development of high performance VLSI-based design methodologies and CAD tools.

He received a doctorate fellowship from CNP_q (Conselho Nacional de Desenvolvimento Cientfico-Brasil) from 1990 to 1994 and has been a research assistant since 1994. His research interests and publications are in the areas of high speed and low power digital CMOS IC design techniques; clock distribution design; synchronous and asynchronous timing issues; high-level synthesis; and algorithm design for the development of CAD tools targeted for high performance systems.

Eby G. Friedman received the B.S. degree from Lafayette College in 1979, and the M.S. and Ph.D.

degrees from the University of California, Irvine, in 1981 and 1989, respectively, all in electrical engineering.

From 1979 to 1991, he was with Hughes Aircraft Company, rising to the position of manager of the Signal Processing Design and Test Department, responsible for high performance VLSI CMOS digital and analog IC's and the development of supporting design and test methodologies and CAD tools. He has been in the Department of Electrical Engineering at the University of Rochester since 1991, where he is an Associate Professor and Director of the High Performance VLSI/IC Design and Analysis Laboratory. His current research and teaching interests are in the areas of high performance VLSI/IC design and analysis and related system specifications.

He has published in the fields of high speed and low power CMOS design techniques, pipelining and retiming, and the theory and application of synchronous clock distribution networks, and has edited three books on high performance VLSI-based circuits and systems.