LINEARIZATION OF THE TIMING ANALYSIS AND OPTIMIZATION OF LEVEL-SENSITIVE SYNCHRONOUS CIRCUITS

by

Baris Taskin

B.S. in E.E., Middle East Technical University, Turkey, 2000

Submitted to the Graduate Faculty of

the School of Engineering in partial fulfillment

of the requirements for the degree of

Master of Science

University of Pittsburgh

2003

UNIVERSITY OF PITTSBURGH

SCHOOL OF ENGINEERING

This thesis was presented

by

Baris Taskin

It was defended on

March 28, 2003

and approved by

Steven P. Levitan, Professor, Electrical Engineering Department

Marlin H. Mickle, Professor, Electrical Engineering Department

Thesis Advisor: Ivan S. Kourtev, Assistant Professor, Electrical Engineering Department

ABSTRACT

LINEARIZATION OF THE TIMING ANALYSIS AND OPTIMIZATION OF LEVEL-SENSITIVE SYNCHRONOUS CIRCUITS

Baris Taskin , M.S.

University of Pittsburgh, 2003

This thesis describes a linear programming (LP) formulation applicable to the static timing analysis of large scale synchronous circuits with level-sensitive latches. The automatic timing analysis procedure presented here is composed of deriving the connectivity information, constructing the LP model and solving the clock period minimization problem of synchronous digital VLSI circuits. In synchronous circuits with level-sensitive latches, operation at a reduced clock period (higher clock frequency) is possible by taking advantage of both non-zero clock skew scheduling [12]* and time borrowing [14]. Clock skew scheduling is performed in order to exploit the benefits of nonidentical clock signal delays on circuit timing. The time borrowing property of level-sensitive circuits permits higher operating frequencies compared to edge-sensitive circuits. Considering time borrowing in the timing analysis, however, introduces non-linearity in this timing analysis. The *modified big M (MBM)* method is defined in order to transform the non-linear constraints arising in the problem formulation into solvable linear constraints. Equivalent LP model problems for single-phase clock synchronization of the ISCAS'89 benchmark circuits are generated and these problems are solved by the industrial LP solver CPLEX [10]. Through the simultaneous application of time borrowing and clock skew scheduling, up to 63% improvements are demonstrated in minimum clock period with respect to zero-skew edge-sensitive synchronous circuits. The timing constraints governing the level-sensitive synchronous circuit operation not only solve the clock period minimization problem but also

*Bracketed references placed superior to the line of text refer to the bibliography.

provide a common framework for the general timing analysis of such circuits. The inclusion of additional constraints into the problem formulation in order to meet the timing requirements imposed by specific application environments is discussed.

DESCRIPTORS

Digital synchronous VLSI circuits	Linear Programming
Optimization	Static timing analysis

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

S	A synchronous digital VLSI circuit.
G_S	The graph representation of the synchronous circuit S.
R	A register in a synchronous circuit network.
R_i	The initial register of a local data path.
R_f	The final register of a local data path.
$R_i \rightarrow R_f$	The local data path formed by the sequentially adjacent registers R_i and R_f.
C	A clock signal synchronizing a circuit network.
C_i	The clock signal driving the register R_i of a circuit network.
C_f	Clock signal driving the register R_f of a circuit network.
T	The period of the clock signal C.
C_W^L	The width of the active phase of the clock signal in a level-sensitive circuit.
C_W^F	The width of the active phase of the clock signal in an edge-sensitive circuit.
ϕ_i	The latency of the clock signal C_i with respect to common clock signal.
ϕ_f	The latency of the clock signal C_f with respect to common clock signal.
ϕ_{if}	The phase shift operator: $$\phi_{if} = \phi_i - \phi_f + kT,$$ where k is the number of clock cycles occuring between respective cycles of the clock signals C_i and C_f.
t_i	The delay of the clock signal C_i from the clock source to the register R_i.
t_f	The delay of the clock signal C_f from the clock source to the register R_f.
$T_{skew}(i,f)$	The clock skew between registers R_i and R_f: $$T_{skew}(i,f) = t_i - t_f.$$
D_P^{if}	The propagation delay of the data signal on the local data path $R_i \rightarrow R_f$.
D_{PM}^{if}	The maximum propagation delay of the data signal on $R_i \rightarrow R_f$.
D_{Pm}^{if}	The minimum propagation delay of the data signal on $R_i \rightarrow R_f$.
H_f	The hold time of the latch R_f.

S_f	The setup time of the latch R_f.
D_{CQ}	The clock-to-output delay of a latch.
D_{DQ}	The data-to-output delay of a latch.
$\max()$	The maximum function. Evaluates to the value of the variable with the highest value given in the function.
$\min()$	The minimum function. Evaluates to the value of the variable with the lowest value given in the function.
$Fan-in()$	The function to return the number of incoming local data paths of a register.
r	The number of registers in a synchronous circuit.
p	The number of local data paths in a synchronous circuit.
T_{FF}^{noskew}	The minimum clock period of an edge-sensitive synchronous circuit under zero-clock skew.
T_{FF}^{skewed}	The minimum clock period of an edge-sensitive synchronous circuit under non-zero clock skew scheduling.
T_L^{noskew}	The minimum clock period of a level-sensitive synchronous circuit under zero-clock skew.
T_L^{skewed}	The minimum clock period of a level-sensitive synchronous circuit under non-zero clock skew scheduling.
I_{FF}^{CSS}	The improvement of the minimum clock period of an edge sensitive synchronous circuit through clock skew scheduling.
I_L^{CSS}	The improvement of the minimum clock period of a level-sensitive synchronous circuit through clock skew scheduling.
I_L^{TB}	The improvement of the minimum clock period of a level-sensitive synchronous circuit through time borrowing.
I_L^{TBCSS}	The improvement of the minimum clock period of a level-sensitive synchronous circuit through time borrowing and clock skew scheduling.
T_L^r	The minimum clock period of an additionally restricted level-sensitive synchronous circuit with latches.
I_L^r	The improvement of the minimum clock period of an additionally restricted level-sensitive synchronous circuit.

ABBREVIATIONS

LP	Linear Programming
NLP	Non-Linear Programming
MBM	Modified Big M Method

1.0 INTRODUCTION

The timing analysis of a digital VLSI synchronous circuit is subject to the structural and application-specific timing constraints of the circuit. Various algorithms have been proposed to model the timing scheduling problem of synchronous circuits as both *linear* and *non-linear* continuous optimization problems [1, 2, 4, 6, 8, 11, 12, 18, 19, 24, 25]. A high-level categorization of such timing analyses is by the register type; analyses targeting *edge-sensitive* (flip-flop based) and *level-sensitive* (latch based) digital synchronous circuits. The timing scheduling problem of edge-sensitive synchronous circuits has been successfully addressed by both linear and quadratic programming approaches in [12]. Furthermore, it is known that the timing constraints for level-sensitive synchronous digital circuits contain non-linearity due to time borrowing [14]. Previous studies on level-sensitive circuits have focused on relaxing these non-linear constraints by using iterative solution methodologies [2, 18, 19, 25]. In [25], the effects of time borrowing on the circuit operation are considered using an iterative approach starting with a bound for the minimum clock period and checking for timing violations. The formulation in [25] provides a limited consideration of non-zero clock skew in the timing analysis and is modified in [8] to accommodate for local clock skew as well as global clock skew. Both of the analyses offered in [25] and [8] consider clock skew as a *fixed* numerical constant. Considering clock skew as a *variable* within a permissible range, instead of a fixed value, permits operation at higher operating frequencies (through clock skew scheduling [12]). In this work, the linear timing analysis problem generated by the *modified big M*(*MBM*) method not only accounts for time borrowing [14] but also applies clock skew scheduling [12] in order to calculate the minimum clock period of a level-sensitive synchronous circuit.

This thesis integrates the ongoing research efforts at the Electrical Engineering Department of the University of Pittsburgh in order to develop a novel timing analysis and optimization procedure for high-performance level-sensitive synchronous circuits. The formulation and solution procedures proposed herein are also introduced in [26–28]. Note that the *static timing analysis* of level-sensitive synchronous circuits is pursued both in previous and the presented work.

The objective of this work is to formulate the clock period minimization problem of level-sensitive synchronous circuits under non-zero clock skew as a Linear Programming (LP) problem. In this type of timing analysis, time borrowing is accounted for and clock skew scheduling is applied in a single-phase synchronization scheme of operation. The resulting problem for level-sensitive circuits is inherently *non-linear*. Thus, the MBM method is introduced in order to linearize the problem formulation. The proposed formulation and calculation procedures are completely automated. The linearization procedure presented here differs from the iteration-based algorithms [2, 18, 19, 25] in that it provides a stand-alone LP model with a specific objective function and distinct set of constraints that can be solved by any standard LP solver. The LP problems formulated in this work are solved using the industrial LP solver CPLEX which implements a rich set of memory and speed efficient optimization algorithms [10]. As the CPLEX solvers implement variants of the *simplex method* [30] to solve the LP model problem, the *expected* computational complexity of the procedure is polynomially bounded [3].

The rest of this document is organized as follows. The fundamentals of synchronous system operation and system modeling for computational analysis are summarized in Chapter 2. Abstract operation of system storage elements and parameterization of the relevant timing properties and problem variables are presented in Chapter 3. Chapter 4 summarizes the current state of static timing analysis of level-sensitive synchronous circuits and introduces the timing constraints. The development of the proposed problem formulation and the generated LP model problem are presented in Chapter 5. The analysis and validation of the results as the presented procedure is applied on the suite of ISCAS'89 benchmark circuits are presented in Chapter 6. Comments on the experimental results and concluding remarks are offered in Chapter 7.

2.0 SYNCHRONOUS DIGITAL VLSI SYSTEMS

VLSI is an acronym that stands for very large scale integration. This term is used to refer to a broad area of electrical and computer engineering application fields, where the focus is on the design and analysis of dense electronic integrated circuits. The investigation of high-performance computational elements, high-density memory elements, sequential and combinational control logic units, sub-micron size analog circuits, etc. are some samples among various others, in fields related to VLSI circuit design.

VLSI circuits can be classified according to typical field of application, details of the manufacturing process, operational characteristics and other features. In this study, digital VLSI circuit design is of particular interest and in the rest of the document, the term *VLSI design* is used to refer to this particular group of circuits unless otherwise indicated.

The term *digital VLSI circuit design* implies the design of sequential and combinational circuits. Sequential circuits consist of *logic* and *register* (or *storage*) elements while combinational circuits consist only of the former. A detailed discussion of sequential and combinational circuits is presented in [20]. All sequential circuits have a well-defined ordering of *switching* events that ensure the correct ordering of data propagation between register elements and ultimately between the input and output pins. The majority of the sequential circuits currently on the market are *synchronous* circuits, where the sequentiality in time of the data transfers are provided by a globally distributed synchronization signal. The globally distributed synchronization signal is called the *clock* signal and defines the *timing scheme* or *timing discipline* of the synchronous circuit [6]. Furthermore, the distribution of the clock signal throughout the circuit, in order to generate and distribute the time reference to each register, is accomplished through a highly specialized structure called the *clock distribution network* or *clock tree network.*

A digital synchronous circuit is a network of combinational logic elements and globally clocked registers. The combinational logic elements implement the functionality of the circuit, where the clocked registers serve to store the computation results at each clock cycle. The logical order of the computation and storage processes is synchronized by the globally distributed clock signal. An overall representation of a typical synchronous circuit is shown in Figure 2.1.

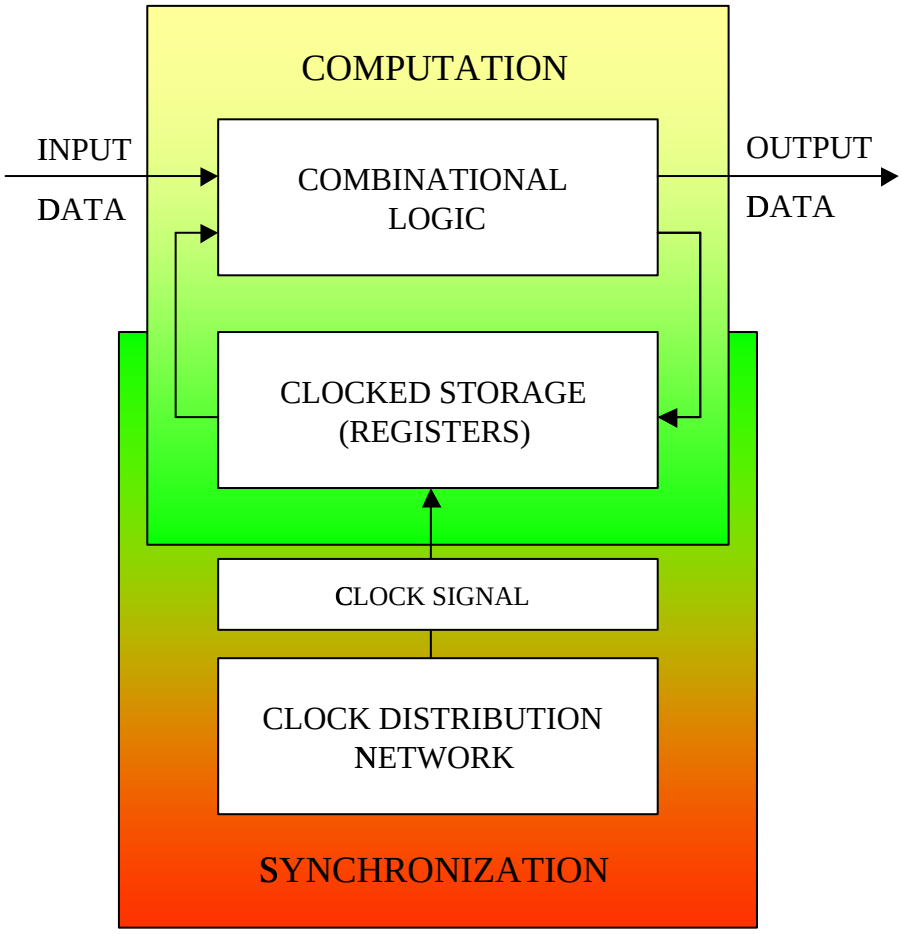


Figure 2.1 Finite state machine model of a synchronous system.

The operation of a synchronous system involves gradual propagation of data signals from the input pins towards the output pins of the circuit. The gradual advancement—and computation—of the data signal is orchestrated by the clock signal. Typically, the active level or transition of the clock signal initiates the propagation of data from the input terminals and storage elements towards the output terminals and next stages of storage elements. Each clock cycle constitutes a computational cycle, where the data signal departs from the source, is processed in the combinational logic between the registers and is finally stored in or delivered to the destination register or the output terminals of the synchronous circuit, respectively.

In order to analyze the operational characteristics of a synchronous circuit, *local data paths* are defined. The overall operation of a synchronous circuit is the sequential execution of a large set of simple computations that occur at each local data path. A local data path is the building block of a synchronous circuit and is

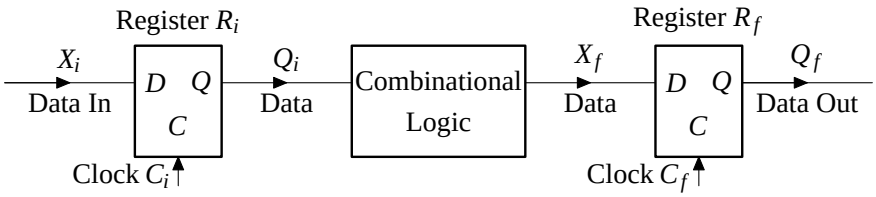


Figure 2.2 A local data path in a globally clocked synchronous circuit network.

formed by a collection of combinational logic blocks between two clocked storage elements. The data signal is processed within a local data path once for each clock signal cycle, where the data signal initiating from a storage element is processed in the combinational logic block and is stored in the next stage of storage elements, ready for the next clock cycle.

Definition 1: Local data path. Let R_i and R_f be two registers in the clocked circuit network where *i* and *f* stand for initial and final, respectively. Let the input and output terminals of these registers, and the signals present at these terminals be defined as shown in Figure 2.2. A local data path is the circuit architecture formed by a sequentially adjacent pair of registers [12] and the combinational logic block between them. The output Q_i of the initial register propagates through the combinational logic block, evaluating to the data signal X_f, before the data signal X_f arrives at the final register R_f. For proper operation of the sequentially adjacent pair of registers R_i and R_f, the data stored in R_i must be manipulated by the logic and stored into R_f during the next cycle of the clock signal C_f. Note that the clock signals C_i and C_f are representations for the two synchronization signals at the clock input terminals of registers R_i and R_f, respectively. The clock signals C_i and C_f differ by the nonidentical clock signal delays at respective registers.

The modeling and behavior of synchronous digital systems are described in the rest of this section. Specifically, the operation of a synchronous system with level-sensitive latches is presented in Section 2.1. The modeling of a sequential circuit as a graph is briefly reviewed in Section 2.2. Different synchronization schemes adopted for synchronous circuits are briefly introduced in Section 2.3.

2.1 Operation of a Synchronous System with Registers

Because of interconnect delays and process parameter variations, there may be significant time delays between the clock signals originating at the clock source and arriving at the destination registers. The algebraic difference between the delays of the synchronizing clock signals of the initial and final register of a local data path is defined as clock skew [12]:

Definition 2: Clock Skew. Let R_i and R_f be a sequentially adjacent pair of registers (only combinational delay between registers) synchronized by the clock signals C_i and C_f, respectively. The clock skew between R_i and R_f is defined as

$$T_{skew}(i, f) = t_i - t_f, \tag{2-1}$$

where t_i and t_f are delays of the clock signals, C_i and C_f, from a common clock source to the registers R_i and R_f, respectively [12].

The clock skew is an algebraic difference which may evaluate to a negative, zero or positive value depending on the values of t_i and t_f. Positive clock skew has a limiting effect on the maximum operating frequency of a synchronous circuit. Negative clock skew on the other hand may effectively improve the minimum clock period of a circuit. Precise engineering of the clock tree network enables the utilization of negative skew on critical paths and permits positive clock skew on less-critical paths in order to speed up the data propagation. This approach is called *clock skew scheduling* [12].

Definition 3: Clock skew scheduling. Clock skew scheduling is a methodology to determine the optimal values of clock signal delays t_k to each register R_k in order to obtain the maximum operating frequency. Clock skew scheduling provides shorter minimum clock periods on critical paths. Note that generally, the term *clock skew scheduling* refers to *non-zero clock skew scheduling* [12].

Excessive negative and positive clock skew may lead to timing hazards in the circuit. Negative skew may cause data to be latched into the final register R_f during an earlier clock cycle than intended, thereby overwriting data latched during the earlier clock cycle. This type of hazard is known as *double clocking* [12]. Similarly, positive skew may cause data to be lost by arriving late at the final register. This phenomenon is

known as *zero clocking* [12]. The double clocking and zero clocking hazards are also called *hold* and *setup time violations*, respectively [25].

Data propagation in level-sensitive circuits is substantially different compared to edge-sensitive synchronous circuits due to the transparency property of latches. In level-sensitive circuits, the data signal arriving at a latch in the transparent phase is immediately propagated through the latch. This fact leads to the phenomenon called *time borrowing*.

Definition 4: Time borrowing [28]. Time borrowing (also called cycle stealing [14]) refers to the time sharing phenomenon between consecutive clock cycles of adjacent local data paths due to the transparency of level-sensitive latches. Let $R_i \rightarrow R_j$ and $R_j \rightarrow R_k$ be two local data paths in a synchronous circuit S. Let S_{FF} and S_L denote the edge-sensitive (flip-flop-based) and level-sensitive (latch-based) synchronous circuits, respectively. In the edge-sensitive synchronous circuit S_{FF}, the upper bound on data propagation time on the local data paths $R_i \rightarrow R_j$ and $R_j \rightarrow R_k$ is the minimum clock period T_{FF}. In the level-sensitive circuit S_L, the transparency property of the latch R_j permits data propagation times higher than the minimum clock period T_L on the local data path $R_i \rightarrow R_j$ by borrowing time from the propagation on the next local data path—next clock cycle—$R_j \rightarrow R_k$. It is known that unless the circuit topology of the edge-sensitive circuit S_{FF} or the level-sensitive circuit S_L is modified, the data propagation times in both circuits are identical. Therefore, a shorter minimum clock period $T_L \leq T_{FF}$ is feasible for the level-sensitive circuit S_L. This fact is illustrated in Figure 2.3, where the timing diagrams for S_L and S_{FF} are shown on the left and right, respectively. Detailed investigation of the time borrowing phenomenon is presented in [14].

2.2 Graph Model of a Synchronous Digital VLSI System

A graph model is often used for computer representation of a synchronous digital system. For convenience, the graph model representing a synchronous circuit, where each vertex represents a register and each edge represents a local data path, is called a circuit graph. A circuit graph provides a common abstract framework for the automated analysis of circuits.

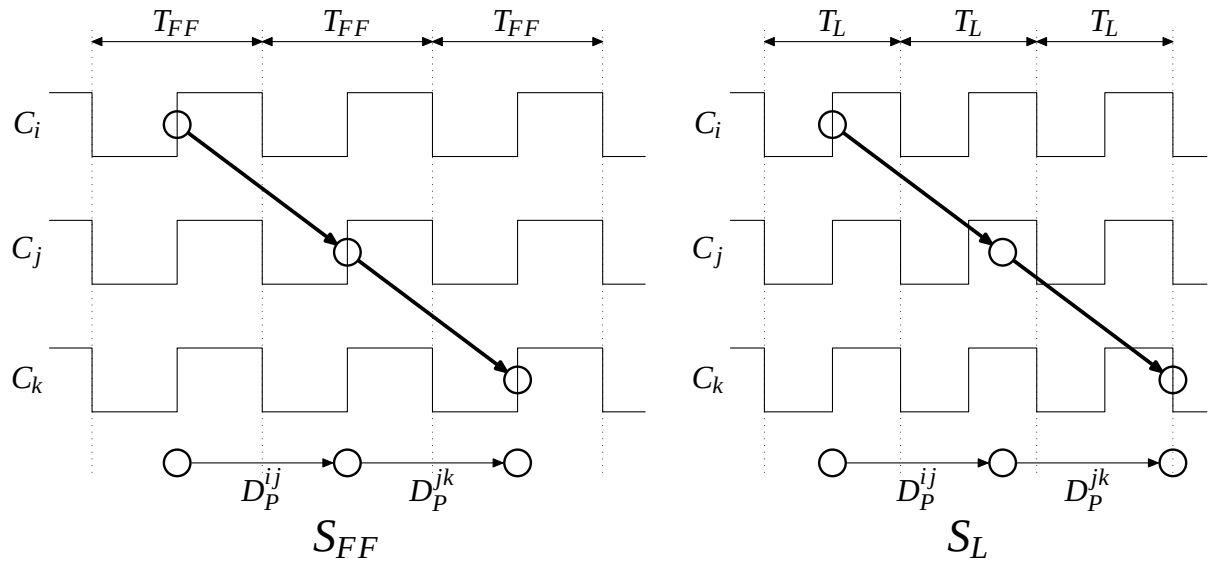


Figure 2.3 Effects of time borrowing on circuit operation. The timing diagram for the edge-sensitive circuit S_{FF} and level-sensitive circuit S_l are shown. The variables D_P^{ij} andD_P^{jk} represent data propagation times on local data paths $R_i \rightarrow R_j$ and $R_j \rightarrow R_k$, respectively. Data propagation is represented by the arrows. Note that in the local data path $R_i \rightarrow R_j$ of S_L, the data signal arrival at R_j occurs during the transparent phase of R_j, borrowing time from the adjacent local data path $R_j \rightarrow R_k$. For identical data propagation times on adjacent local data paths, a smaller clock period (higher operating frequency) is possible for S_L, that is, $T_L \leq T_{FF}$.

Definition 5: Circuit graph [12]. A fully synchronous digital circuit S is represented as the connected undirected simple graph G_S. The graph G_S is the ordered six-tuple $G_S =< V^{(S)}, E^{(S)}, A^{(S)}, h_l^{(S)}, h_u^{(S)}, h_d^{(S)} >$, where $V^{(S)} = v_1, .., v_r$ is the set of vertices of the graph G_S, $E^{(S)} = e_1, .., e_p$ is the edges of the graph G_S, $A^{(S)} = [a_{ij}^{(S)}]_{r \times r}$ is the symmetric adjacency matrix of G_S [12]. Note that r is the number of registers and p is the number of data paths in the synchronous circuit. Each vertex from $V^{(S)}$ represents a register of the circuit S. The mappings $h_l^{(S)} : E^{(S)} \rightarrow \Re$ and $h_u^{(S)} : E^{(S)} \rightarrow \Re$ to the set of real numbers $\Re$ assign the lower and upper permissible bounds D_{Pm}^k and D_{PM}^k, respectively. The lower and upper bounds of data propagation time D_{Pm}^k, D_{PM}^k are defined between the sequentially-adjacent pair of registers (on the local data path) represented by the edge $e_k \in E$. The edge labeling $h_d^{(S)}$ defines a direction of signal propagation for each edge v_x, e_z, v_y, where the subscripts x and y denote two arbitrary registers forming the local data path indicated by the subscript z.

The described undirected graph is used as the basis for a directed graph with the same vertices set V and edge set E. The topology of the graph is preserved, while the edge set is modified in order to

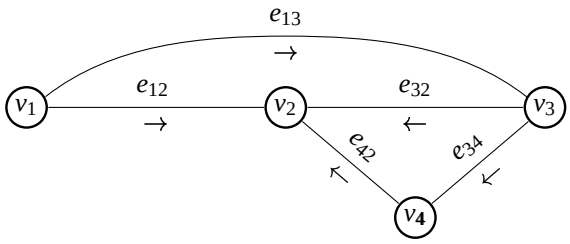


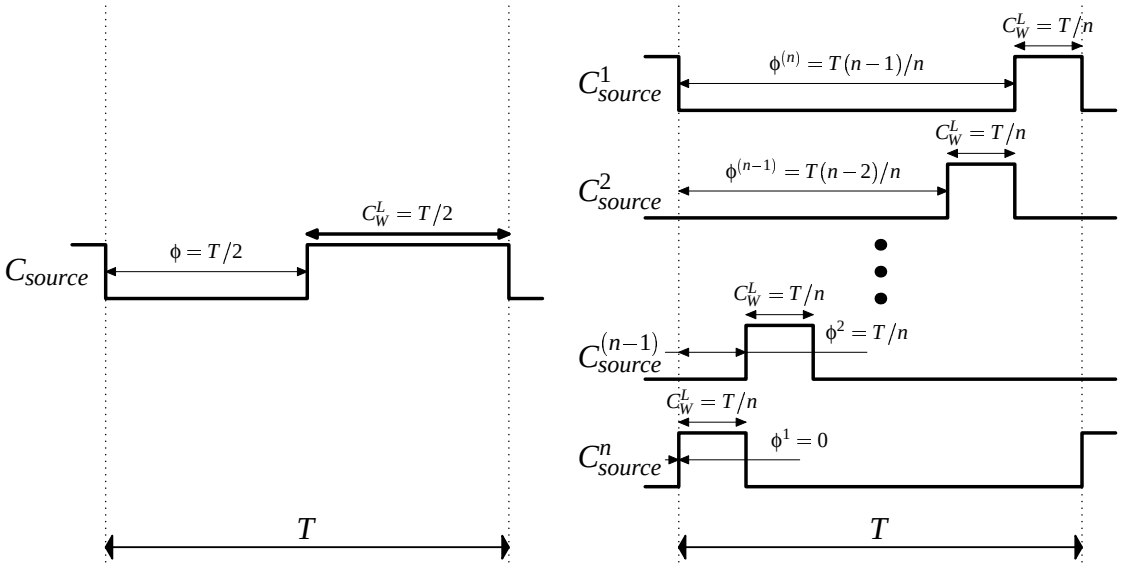
Figure 2.4 A graph representation of a synchronous system. The graph vertices are four different registers, with five local data paths.

accommodate for the direction of data flow between the vertices. The generated directed graph, where each vertex corresponds to a register and each *directed* edge corresponds to a data path, represents a sequential circuit. The directed graph representation of a sample network is presented in Figure 2.4.

2.3 Single-Phase Synchronization

As mentioned earlier, the operation of a synchronous circuit is orchestrated by a globally distributed clock signal. The clock signal ensures the correct ordering of operations on local data paths. Synchronization in such circuits can be provided by a *single* or *multi-phase* clock signal. A single phase clock signal is a periodic signal where the phases or cycles are indistinguishable instances referred to a fixed reference cycle. However, the easy-to-implement and easy-to-analyze single-phase scheme has several shortcomings in the synchronization of state-of-the-art VLSI circuits. Below nanometer feature sizes, the wire sizes shrink disproportionally with the feature size. Thus, only a certain percentage of the chip is reachable during a single clock cycle [9]. For instance, for $0.1\mu m$ feature size, only 16% of the die is predicted to be reachable within a single clock cycle [9]. A multi-clock domain approach is advantageous in terms of increasing the reachability of circuit registers, routing and creating less skew within physically neighboring local clock domains and saving power. Furthermore, individual domains can be designed and operated independently, which provides higher granularity towards frequency and voltage scaling.

Even though multi-phase synchronization is advantageous in many aspects, the analyses of such clocking schemes present many challenges. Identification of the clock tree and partitioning into various clock



Single-phase clock signal Multi-phase clock signal

Figure 2.5 Single and multi-phase synchronization of a synchronous circuit. In multi-phase synchronization, the non-overlapping clock phases are defined with identical on-times (C_W^L). The parameter C_{source} denotes the clock signal at the originating clock source. The superscripts $\{1, \ldots, n\}$ represent individual clock phases. Note that the multi-phase clock synchronization is defined for $n \geq 2$, where n represents the number of clock phases.

domains, identifying signals that work across different clock domains, identifying data stability and combinatorial input isolations are common concerns. In the presented work, only the *single-phase* synchronization of synchronous circuits is investigated. Figure 2.5 presents a generic representation of single-phase and multi-phase symmetric clock signals.

The generic formulation for multi-phase clocking is defined for $n > 2$. The latency of each clock cycle with respect to the common clock signal is unique and is denoted here by the parameter ϕ_i. The parameter ϕ_i is the latency of the clock signal at register R_i, which is synchronized by clock phase C^{p_i}. The symbol ϕ_{if} is called the *phase shift operator* [2] which is used to transform timing variables between different cycles of the clock signal. The phase shift operator ϕ_{if} is defined by the algebraic equation $\phi_{if} = \phi_i - \phi_f + kT$, where k is the number of clock cycles occurring between the clock phases C^{p_i} and C^{p_f}. Note that for the single-phase clock methodology, the phase shift operator evaluates to $\phi_{if} = T$. The improvement of the presented timing analysis in order to accommodate multi-phase or multi-domain clock synchronization is among the future directions of this research.

3.0 TIMING PROPERTIES OF SYNCHRONOUS DIGITAL SYSTEMS

The general structure and principles of operation of a fully-synchronous digital VLSI system were described in Chapter 2. In an abstract overview, a synchronous circuit is identified by the storage elements and the synchronization scheme of the circuit. As presented in Section 2.3, the operation of a digital VLSI circuit is heavily dependent on the synchronization scheme. Furthermore, the *registers* directly contribute to the operation of the circuits by storing the data values at the end of each computational cycle and providing stable data signals.

Registers can be classified into two categories: (edge-triggered) flip-flops are sensitive to the changes in their data input terminals when the clock signal has low-to-high or high-to-low transition while (level-sensitive) latches are sensitive when the clock signal has a certain level or value. In level-sensitive circuits, the active level of the synchronizing clock signal defines the *transparent* phase and the inactive level of the clock signal defines the *opaque* phase of latch operation. The transition of the clock signal which starts the transparent phase is called the *leading* edge, while the *trailing* edge is the transition of the clock signal which concludes the transparent phase and marks the beginning of the opaque phase. This chapter begins with the introduction of two different types of storage elements—*flip-flops* and *latches*. The principles of operation of these registers are discussed and parameters defining the operational characteristics are introduced. In particular, the operation of edge-triggered flip-flops are discussed in Section 3.1 and the operation of level-sensitive latches are discussed in Section 3.2.

3.1 Parameters of an Edge-Sensitive Synchronous Circuit

The specific circuit design or electrical implementation of an edge-triggered flip-flop need not be considered in this work. At a higher level of abstraction, the timing properties of flip-flops are encapsulated by certain timing parameters. These parameters connect the events on the input, output and clock terminals of a flip-flop.

A *flip-flop* is a type of register which is sensitive to the transition of the synchronizing clock signal. Therefore, a *flip-flop* is commonly referred to as an edge-triggered flip-flop or edge-triggered register. A typical edge-triggered flip-flop with a clock signal C, input signal D and output signal Q is shown in Figure 3.1. The operation of a flip-flop is presented in Figure 3.2. Note that the presented flip-flop is a

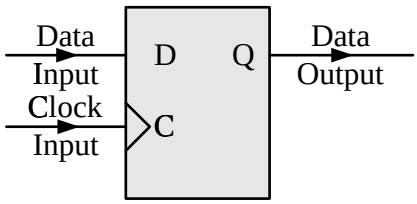


Figure 3.1 An edge-triggered flip-flop or register symbol.

positive-edge triggered flip-flop, whose data output latches the input signal when the clock signal makes a low-to-high transition. The sensitive region for a flip-flop, where the register latches the data, is indicated by the shaded region in Figure 3.2.

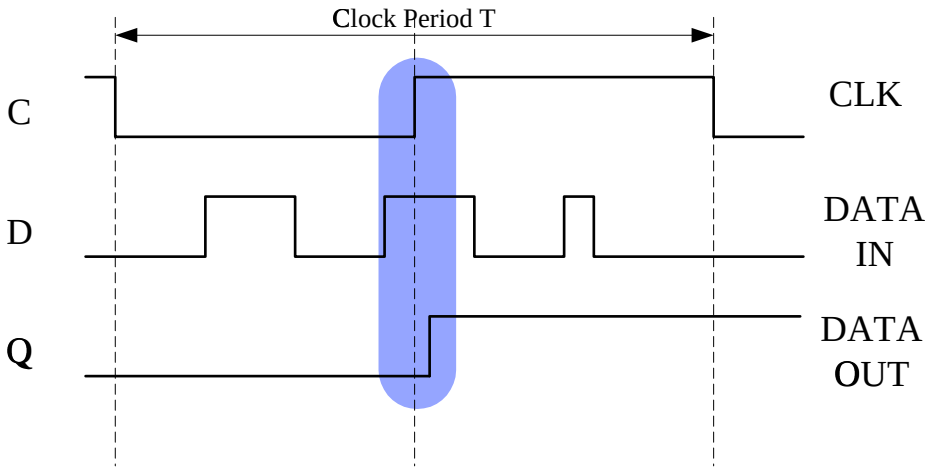


Figure 3.2 Typical operation of an edge-trigerred flip-flop shown in Figure 3.1.

Parameters H_f, S_f, D_{CQ} and C_W^L which stand for the hold time, setup time, clock-to-output delay and clock on-time, respectively are introduced briefly. Hold time is the minimum time that the data signal D must remain stable after the latching edge of the clock signal so that it is registered at the intended clock cycle. In Figure 3.3, the value $H_f = t_4 - t_3$ labels the hold time for the given clock cycle on R_f. Setup time

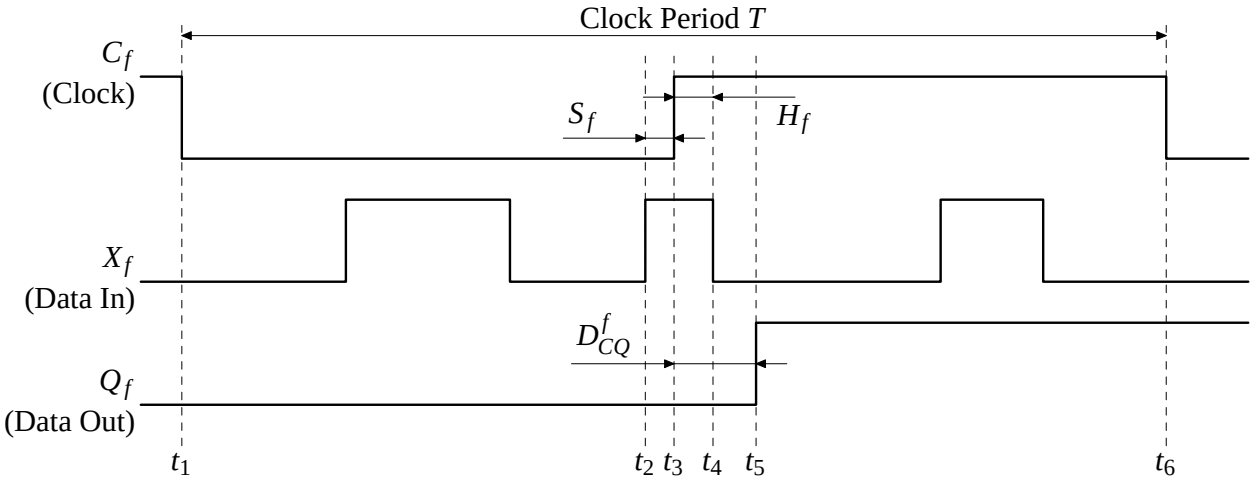


Figure 3.3 Timing properties of an edge-sensitive flip-flop in a circuit with a clock period $T = t_8 - t_1$. The operation of the final flip-flop R_f of a local data path is illustrated.

is the minimum time between the latching edge of the respective clock cycle and a change in X_f such that the new data value can be registered in the intended clock cycle. The setup time on R_f is illustrated with $S_f = t_3 - t_2$. The propagation delay of the data signal from the input terminal to the output terminal after the active transition of the clock signal—clock-to-output delay—is shown as $t_5 - t_3$. The subscripts *m* and *M* appended to the parameter D_{CQ} stand for the minimum and maximum delay values, respectively.

A typical clock cycle of a clock signal is shown in Figure 3.3. The length of the clock period is denoted by the parameter T. The minimum time interval between the leading and trailing edges of the clock signal is represented by the parameter C^L_W. In Figure 3.3, the *leading* and *trailing* edges occur at times t_3 and t_6, respectively. As pointed out in Section 2.2, the data propagation time D_P through the combinational logic block of a local data path $R_i \rightarrow R_f$ is defined $D_{Pm} \leq D_P \leq D_{PM}$. The subscripts m and M stand for the minimum and maximum values, respectively.

3.2 Parameters of a Level-Sensitive Synchronous Circuit

The specific circuit design or electrical implementation of a level-sensitive latch need not be considered in this work. At a higher level of abstraction, the timing properties of such latches are encapsulated by

certain timing parameters. These parameters connect the events on the input, output and clock terminals of a level-sensitive latch.

A *latch* is a type of register which is sensitive to the level of synchronizing clock signal. Therefore, a *latch* is commonly referred to as a level-sensitive latch or level-sensitive register. A typical level-sensitive latch with a clock signal C, input signal D and output signal Q is shown in Figure 3.4. The operation of

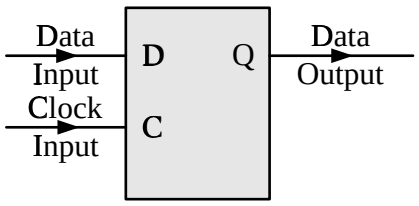


Figure 3.4 A level-sensitive latch or register symbol.

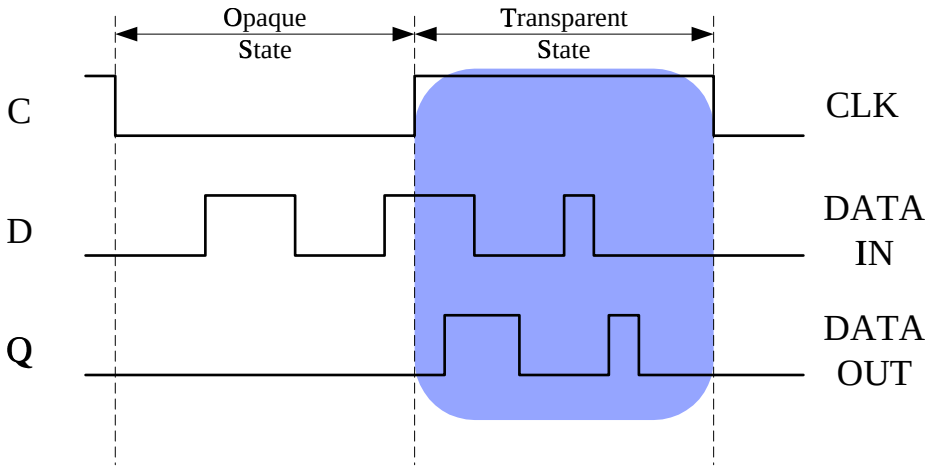


Figure 3.5 Typical operation of a level-sensitive latch shown in Figure 3.4.

the level-sensitive latch is presented in Figure 3.5. Note that the presented latch is a positive-level sensitive latch, whose data output follows any change in the input signal when the clock signal remains at its positive value or level. As stated before, the state of the level-sensitive latch, where any change in the input signal is propagated to the output is called the *transparent* state or phase. In Figure 3.5, the transparent state is indicated by the shaded region on the timing diagram.

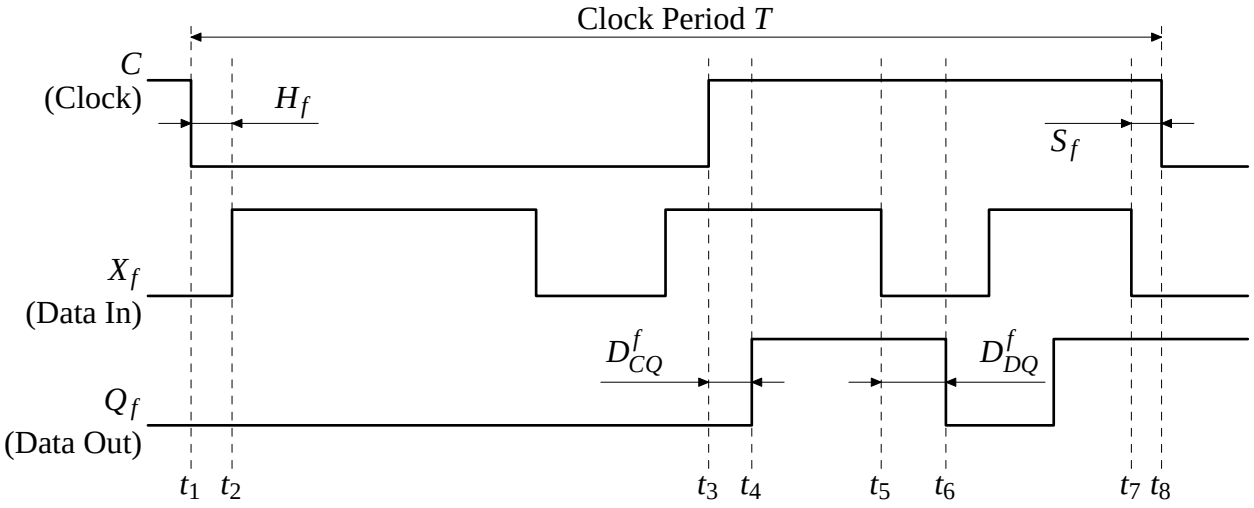


Figure 3.6 Timing properties of a level-sensitive latch in a circuit with a clock period $T = t_8 - t_1$. The operation of the final latch R_f of a local data path is illustrated.

Parameters H_f, S_f, D_{DQ}, D_{CQ} and C^L_W which stand for the hold time, setup time, data-to-output delay, clock-to-output delay and clock on-time, respectively are introduced briefly. Hold time is the minimum time that the data signal D must remain stable after the trailing edge of the clock signal so that it is latched during the intended clock cycle. In Figure 3.6, the value $H_f = t_2 - t_1$ labels the hold time for the given clock cycle on the final register R_f of a local data path. Setup time is the minimum time between the trailing edge of the respective clock cycle and a change in X_f such that the new data value can be latched in the intended clock cycle. The setup time on R_f is illustrated with $S_f = t_8 - t_7$. The propagation delay of the latch from the data input terminal to the output terminal—data-to-output delay—on R_f is shown as $t_6 - t_5$. The propagation delay of the latch from the clock input terminal to the output terminal—clock-to-output delay—is shown as $t_4 - t_3$. The subscripts *m* and *M* appended to the parameters D_{DQ} and D_{CQ} stand for the minimum and maximum delay values, respectively.

Without affecting the generality of the presented work, the formulation of the timing constraints is derived for a specific reference clock cycle. The reference clock cycle can be selected as starting with the inactive value of the clock signal followed by the active value (opaque-phase-first) or vice versa (transparent-phase-first). In this work, the timing constraints are formulated considering an opaque-phase-first clock signal driving positive level-sensitive latches. A typical clock cycle of such a clock signal is shown in

Figure 3.6. The length of the clock period is denoted by the parameter T. The minimum time interval between the leading and trailing edges of the clock signal—commonly called the on-time and defining the transparent phase—is represented by the parameter C_W^L. In Figure 3.6, the *leading* and *trailing* edges occur at times t_3 and t_8, respectively.

The data propagation time D_P through the combinational logic block of a local data path $R_i \rightarrow R_f$ was defined in Section 2.2 (Recall that $D_{Pm} \leq D_P \leq D_{PM}$). The propagation of the data signal through the data path $R_i \rightarrow R_f$ is formulated during two consecutive clock cycles, generically called the k-th and $(k+1)$-th. The data signal departs from R_i during the k-th clock cycle, is processed in the combinational logic block and arrives at the destination register R_f during the $(k+1)$-th clock cycle. Note that the departure of the data signal from R_i occurs during the transparent phase of the k-th cycle. The arrival of the data signal at R_f can occur both during the transparent and opaque phases of the $(k+1)$-th cycle. If the data signal arrives during the transparent phase, it is immediately propagated through the latch R_f. If the data signal arrives during the opaque phase of R_f, the data signal has to remain stable until the latching edge of the clock signal (beginning of the transparent phase) to propagate through the latch R_f.

4.0 TIMING ANALYSIS OF LEVEL-SENSITIVE SYNCHRONOUS CIRCUITS

Digital VLSI synchronous circuits are subject to different types of timing analyses. The presence of a globally distributed synchronization signal constitutes a merit of comparison between any given sets of synchronous circuits. This fact has been widely used in the electronics industry. General timing analysis of such circuits, however, span more detailed analyses of the circuit performance and have been utilized for various engineering purposes. Timing analysis of synchronous circuits have traditionally been studied on three different problems: clock period minimization [1, 2, 4, 11, 12, 18, 25–27], clock period verification [2, 24] and clock retiming [15]. Clock period minimization is the analysis of a synchronous circuit in order to solve for the minimum clock period, in other words for the maximum operating frequency, of a synchronous circuit. Clock period verification is the analysis to ensure that a synchronous circuit is fully-operational for a given clock period. Clock period retiming—also called *circuit retiming*—is the analysis of a synchronous circuit aiming to achieve higher operating frequencies by modifying the circuit architecture.

Even though there are different types of timing analysis problems, the operation of the synchronous circuit under scrutiny is generally identical in all cases (possibly except for retiming problems). Thus, in the formulation of the timing analysis problem, a framework of constraints identifying synchronous circuit operation is essential. For instance the setup and hold time requirements of each register element in a synchronous circuit pose certain constraints on circuit operation, thus they are represented in all types of timing analysis problems.

The generation of a general framework for the timing analysis of level-sensitive circuits is discussed in this chapter. The clock period minimization problem is modeled and the generated problem is later solved in Chapter 5. In Section 4.1, the *operational constraints* [27] governing level-sensitive operation are introduced. In Section 4.2, a brief overview of previously offered algorithms for the clock period minimization problem is described. The *constructional constraints* defined for the novel linear programming model of the clock period minimization problem are presented in Section 4.3.

4.1 Problem Formulation

Certain conditions must be satisfied for every sequentially adjacent pair of registers in a synchronous circuit in order to prevent timing hazards. These conditions are encapsulated by four sets of operational constraints and two sets of constructional constraints. The *operational constraints* [27] are the constraints that model the operation of a synchronous circuit. The *constructional constraints* [27] are defined to ensure the correctness and completeness of the proposed model of te optimization problem. The definitions for the first three sets of operational constraints—called *latching, synchronization* and *propagation* constraints, respectively—are borrowed from [2]. The fourth set of operational constraints—called the *skew* constraints [26]—are derived from the skew definitions for edge-sensitive synchronous circuits presented in [12]. The latching, synchronization, propagation and skew constraints are described in Sections 4.1.1 , 4.1.2 , 4.1.3 and 4.1.4, respectively.

4.1.1 Latching Constraints

Latching constraints bound the arrival time of the data signal X_f (recall the local data path in Figure 2.2) in order to ensure that X_f is latched during the intended clock cycle. The earliest arrival time of X_f at the data input terminal of R_f is denoted by the parameter a_f. Similarly, the latest arrival time of X_f is denoted by A_f. Both parameters are defined in the frame of reference of the native clock cycle, that is, relative to the beginning of the current clock cycle. In Figure 4.1 for instance, the earliest data arrival at R_f occurs at time $t_f + kT + a_f$ with respect to global time zero ($t_f + kT$ is the beginning of the current clock cycle). The interval for the data arrival time is characterized by the hold time and the setup time requirements of R_f as follows:

$$H_f \leq a_f \tag{4-1}$$

$$A_f \leq T - S_f. \tag{4-2}$$

Eq. (4-1) above constrains the earliest arrival of X_f at R_f. The earliest data arrival time must be no earlier than *hold* time after the trailing edge (t_2 in Figure 3.6) of the previous clock cycle. Suppose the $(k+1)$-th

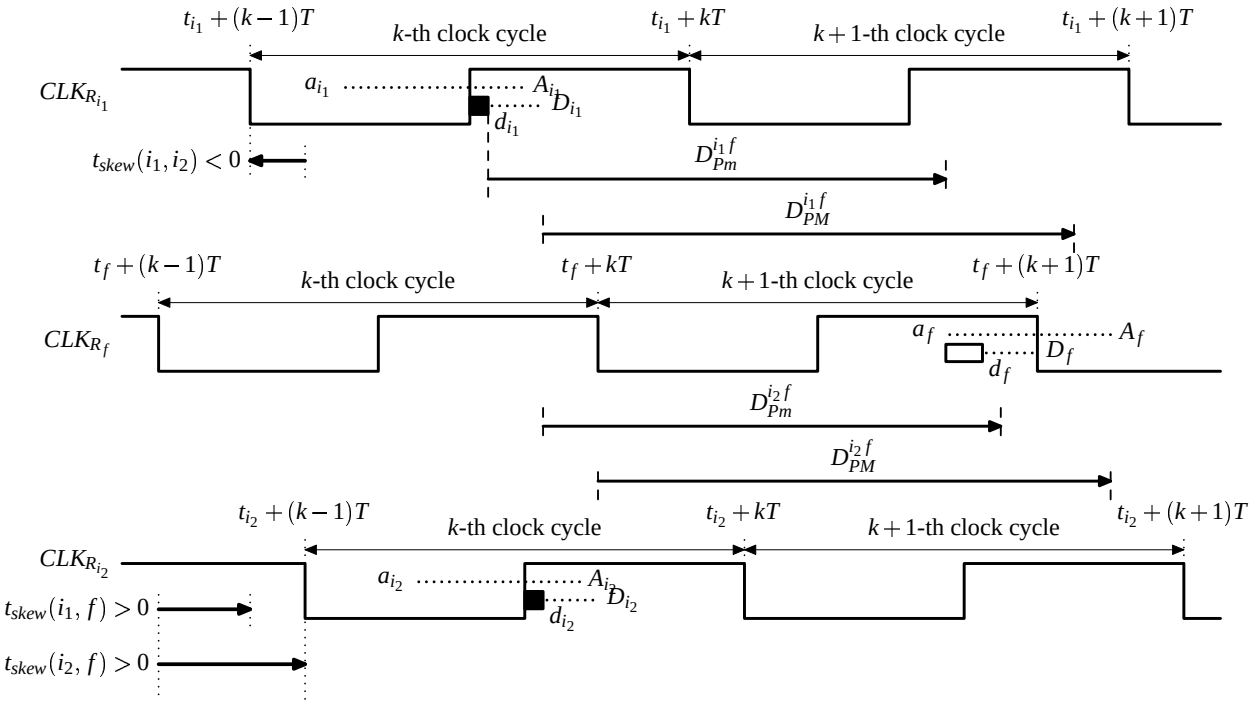


Figure 4.1 Propagation of the data signal in a simple circuit. Note that two local data paths starting at the latches R_{i_1} and R_{i_2} and ending at R_f are considered. The time intervals for the arrival and departure times of the data signal are illustrated by the upper and lower parallel dotted lines, respectively. The lengths of the white and black rectangular boxes correspond to the clock-to-output and data-to-output latch delays, respectively.

clock cycle at latch R_f is illustrated in Figure 3.6, where $t_1 = t_f + kT$. The hold time is defined by the difference $t_2 - t_1$. If data arrives at R_f earlier than the hold time, a *double-clocking* hazard occurs [12].

Similarly, Eq. (4-2) represents the setup constraint on R_f. As shown in Figure 3.6, the data must arrive at the final latch at least *setup* time prior to the trailing edge of the clock cycle. Assuming the $(k+1)$-th clock cycle is illustrated in Figure 3.6, the trailing edge of the clock cycle occurs at $t_8 = t_f + (k+1)T$. Thus, data cannot be latched into R_f during the $(k+1)$-th cycle if the data arrives later than $t_7 = t_f + (k+1)T - S_f$. Late arrival of the data signal results in a *zero clocking* hazard [12] as previously explained.

4.1.2 Synchronization Constraints

Synchronization constraints define the departure time of the data signal Q_i from the initial latch of a local data path as illustrated in Figure 4.2. The departure time from a latch depends on the state of the latch—transparent or opaque. Implementation-specific register internal delays, D_{DQ} and D_{CQ}, affect the

departure times in transparent and opaque states of operation, respectively. The earliest departure time d_i of Q_i from R_i is defined in Eq. (4-3). The latest departure time D_i is defined by Eq. (4-4):

$$d_i = \max\left(a_i + D^i_{DQm}, T - C^L_W + D^i_{CQm}\right), \tag{4-3}$$

$$D_i = \max\left(A_i + D^i_{DQM}, T - C^L_W + D^i_{CQM}\right). \tag{4-4}$$

An exhaustive inspection of all possible cases of earliest and latest departure times during the k-th clock cycle is shown in Figure 4.2.

Consider Eq. (4-3), which describes the earliest departure time of the data signal Q_i from latch R_i. The first term of the max function, $\left(a_i + D^i_{DQm}\right)$, describes the time instant when the input data arrival occurs at its earliest time during the active phase of the clock signal C_i. The data signal immediately propagates through the latch (as illustrated in cases I and VIII of Figure 4.2). In these cases, the earliest departure time d_i from R_i depends on the earliest arrival time a_i of the data signal and the time D^i_{DQ} it takes for the data to appear at the output terminal of R_i.

The second term of the max function, $\left(T - C^L_W + D^i_{CQm}\right)$, refers to the case when the earliest data arrival time occurs during the opaque phase of R_i. In the opaque phase of operation, the departure time of the data signal from the initial latch occurs clock-to-output delay D^i_{CQ} later than the leading edge of the clock signal. Such data propagation is illustrated in cases II-VII of Figure 4.2. The max function is used to combine these cases, and to define the earliest departure time d_i from the initial latch R_i. Similar reasoning applies to the derivation of the latest departure time D_i defined by Eq. (4-4).

4.1.3 Propagation Constraints

Propagation constraints define the arrival time of the data signal X_f at the final latch R_f of a local data path. These constraints are as follows:

$$a_f = \left(\min_i \left[d_i + D^{if}_{Pm} + T_{skew}(i,f)\right]\right) - T \tag{4-5}$$

$$A_f = \left(\max_i \left[D_i + D^{if}_{PM} + T_{skew}(i,f)\right]\right) - T. \tag{4-6}$$

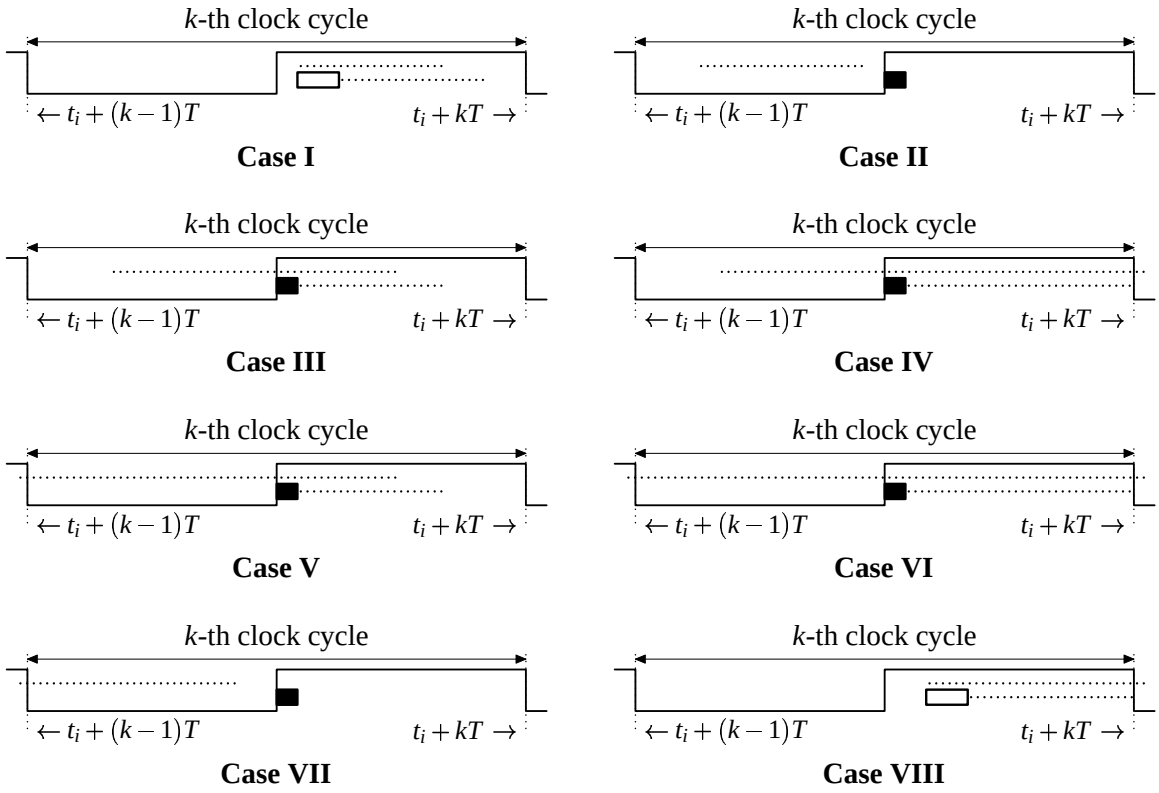


Figure 4.2 Possible cases for the timing relationships among arrival and departure times for the data signal at the latch R_i. The time intervals for the arrival and departure times are illustrated by the upper and lower parallel dotted lines, respectively (The left and right ends of these dotted lines correspond to earliest and latest times, respectively.). The lengths of the white and black rectangular boxes correspond to the clock-to-output and data-to-output latch delays, respectively. Note that cases V through VIII may exhibit clocking hazards as explained in the text.

For each incoming path to latch R_f, the lower bound for a_f is individually calculated using the expression $\left[d_i + D_{Pm}^{if} + T_{skew}(i,f) - T\right]$. The minimum of the arrival times among the incoming data paths is assigned as the earliest arrival time at R_f. The latest arrival time A_f for the data signal is defined similarly. In case of multiple data paths fanning into R_f, the maximum of the arrival times among the incoming data paths is the latest arrival time of the data signal at R_f. These two facts are implied in the formulation by the inclusion of the $\min$ and $\max$ functions in Eqs. (4-5) and (4-6), respectively.

The propagation constraints are illustrated on a sample synchronous circuit in Figure 4.1. The earliest arrival time is illustrated on the data path $R_{i_1} \rightarrow R_f$. The data signal departs from R_{i_1} at time d_{i_1} and propagates on the data path $R_{i_1} \rightarrow R_f$ for a time period of $D_{Pm}^{i_1 f}$. On this data path, the recorded earliest data arrival time $\left[d_{i_1} + D_{Pm}^{i_1 f} + T_{skew}(i_1,f) - T\right]$ is earlier than the arrival time $\left[d_{i_2} + D_{Pm}^{i_2 f} + T_{skew}(i_2,f) - T\right]$

recorded on the only other incoming path to R_f, $R_{i_2} \rightarrow R_f$. Hence, the earliest data arrival time a_f at R_f is defined by the propagation on the $R_{i_1} \rightarrow R_f$ data path. Similarly, on the data path $R_{i_2} \rightarrow R_f$, a maximum data propagation time of $D_{PM}^{i_2 f}$ elapses conferring the latest data arrival time at R_f, $\left[A_f = D_{i_2} + D_{PM}^{i_2 f} + T_{skew}(i_2, f) - T\right]$.

The departure of Q_i and the arrival of X_f must occur during two consecutive clock cycles for proper circuit operation. The phase shift operator $\phi_{if} = T$ is subtracted from the calculated arrival time in order to change the point of reference of the data arrival time at R_f to the beginning of the previous clock cycle.

4.1.4 Skew Constraints

Skew constraints [26] introduce lower and upper bounds on clock skew on a local data path:

$$A_i + D_{DQM}^i + D_{PM}^{if} \leq 2T - T_{skew}(i, f) - S_f \tag{4-7}$$

$$D_{CQM}^i + D_{PM}^{if} \leq T + C_W^L - T_{skew}(i, f) - S_f \tag{4-8}$$

$$\max\left[a_i + D_{DQm}^i, T - C_W^L + D_{CQm}^i\right] + D_{Pm}^{if} \geq T - T_{skew}(i, f) + H_f. \tag{4-9}$$

Presence of clock skew in level-sensitive synchronous circuits significantly affects the system timing. The latching [Eqs. (4-1) and (4-2)], synchronization [Eqs. (4-3) and (4-4)] and propagation [Eqs. (4-5) and (4-6)] constraints presented previously are derived considering the presence of non-zero clock skew in the clock tree network. These three sets of constraints naturally impose lower and upper bounds on clock skew. Thus, the skew constraints are redundant if a typical minimization of the clock period problem is pursued. However, if implementation-specific constraints modify or suppress any of the given constraints, such that, the bounds on clock skew are invalidated, the skew constraints are essential to the correct analysis of the circuit. The skew constraints are important and complete provide a timing analysis framework for level sensitive circuits.

The effects of clock skew on synchronous circuit operation can be derived from the latching [Eqs. (4-1) and (4-2)], synchronization [Eqs. (4-3) and (4-4)] and propagation [Eqs. (4-5) and (4-6)] constraints. Note that the variable A_f described in Eq. (4-2) can be expressed as follows:

$$A_f = D_i + D_{PM}^{if} + T_{skew}(i, f) - T. \tag{4-10}$$

Substituting Eq. (4-4) in Eq. (4-10), then substituting the result into Eq. (4-2) leads to,

$$\max\left[A_i + D^i_{DQm}, T - C^L_W + D^i_{CQm}\right] + D^{if}_{PM} + T_{skew}(i,f) - T \leq T - S_f, \tag{4-11}$$

conveniently represented by the first two sets of skew constraints [Eqs. (4-7) and (4-8)].

Eq. (4-1) must hold to prevent the early arrival of data signal, where a_f depends on d_i as implied by Eq. (4-5):

$$a_f = d_i + D^{if}_{Pm} + T_{skew}(i,f) - T. \tag{4-12}$$

Eq. (4-12) also depends upon whether the data signal arrives before or during the transparent state of the latch. Substituting Eq. (4-3) into Eq. (4-12), Eq. (4-12) into Eq. (4-1) and rearranging the terms lead to the last set of the skew constraints, Eq. (4-9). Eq. (4-9), re-written in Eq. (4-13), is a non-linear skew constraint, as the elimination of the max function is not straightforward:

$$H_f \leq \max\left[a_i + D^i_{DQm}, T - C^L_W + D^i_{CQm}\right] + D^{if}_{Pm} + T_{skew}(i,f) - T. \tag{4-13}$$

As stated before, the skew constraints [Eqs. (4-7), (4-8) and (4-9)] are redundant in the formulation of a typical clock period minimization problem, as these constraints are derived from the existing set of constraints [Eqs. (4-1)–(4-6)]. The skew constraints are not included in the LP model presented in Section 5.2, but are used in the verification of the proposed solution method in Section 6.3.

4.2 Iterative Solution Approach

The operational constraints (Section 4.1) provide a system of equations defining the abstract operation of a level-sensitive synchronous circuit. Different versions of the constraints presented in Section 4.1 have been used by designers in order to develop a timing analysis model for the level-sensitive circuits.

The most significantly-adapted and studied timing analysis approach is presented in [1, 18, 19, 25]. The timing analysis approach presented in this series of papers involve several algorithms targeting clock period verification and minimization problems, all based on the framework of equations described in Section 4.1.

```
//Initialize the latch arrival times
for i = 1 to |r| {
    A_i^prev = a_i^prev = −∞;

    // iterate the evaluation of the departure and arrival time equations
    // until convergence or a maximum of |r| iterations
    iter = 0;
    repeat
        iter = iter + 1;
        // update the latch departure times based on the latch arrival times
        // computed in the previous iteration
        for i = 1 to |r| {
            D_i = max ( A_i^prev, φ_i + D_i );
            d_i = max ( a_i^Prev, φ_i + d_i );
        };
        // update the latch arrival times based on the just-computed
        // latch departure times
        for i = 1 to |r| {
            A_i = max_j ( D_j + D_PM );
            a_i = min_j ( d_j + D_Pm );
        };
    until ( ( ( A_i = A_i^prev ) && ( a_i = a_i^prev ) ) || ( iter + 1 > |r| ) ) )
;

};

// check and record setup and hold violations
for i = 1 to |r| {
    SetupVio[i] = A_i > T - S_i + d_i;
    HoldVio[i] = a_i < H_i + D_i;
};
```

Figure 4.3 The iterative algorithm offered in [18]. Note that r is the number of registers in the synchronous circuit. The a, d, A and D vectors are the earliest arrival/departure and latest arrival/departure times, respectively, where the superscript $prev$ identifies the value of a variable in the previous clock cycle. The variables $SetupVio$ and $HoldVio$ hold the timing violation information for each register.

The algorithms proposed in these papers are iterative algorithms. In particular, very small values are assigned to the timing variables of a circuit and the circuit is investigated for timing violations by iteratively incrementing the values of the timing variables. The iterative algorithm offered in [18] for the clock period minimization problem of level-sensitive circuits is presented in Figure 4.3. In this algorithm, the arrival times are initialized to $a_i = A_i = -\infty$, where the algorithm simulates the start-up timing of the circuit. At each iteration step, the execution of the circuit at a clock cycle is simulated. Finally, once the arrival and departure times of the latches are determined, the algorithm checks for potential setup and hold time violations.

The algorithm presented in Figure 4.3 has been shown to converge to solutions quite rapidly [18]. The algorithm complexity is reported as $O(|r||p|)$, where $|r|$ is the number of latches in a circuit and $|p|$ is the number of edges of a circuit graph (Recall from Section 2.2 that the number of edges of a circuit graph is the number of local data paths). However, it has been proven in [25] that in case of data-path loops (sequential feedback) in the synchronous circuit, the arrival and departure times might increase without bound. This leads to a setup violation and the described algorithm fails to provide reasonable run-times. In [1], a fix is offered to the algorithm. The fix is based on the assumption that, a data path loop in the circuit can be detected in $|r|$ iterations. Thus, the algorithm is modified to artificially limit the number of iteration steps by $|r|$. In this algorithm, the worst case complexity of the resulting algorithm is cubic in the number of registers r, as each iteration involves examining up to $|p|$ edges, and p is at most $|r|^2$ [1].

The iterative algorithm presented in Figure 4.3 is later modified in [8] and [31] in order to account for multiple clock domains and crosstalk, respectively. Briefly, even though the iterative algorithm provides an initial and useful formulation for the timing analysis of level-sensitive circuits, the algorithm has fallacies in presence of data path loops and is insufficient to provide a common framework for general timing analysis. In the presented work, a novel model for the timing analysis of level-sensitive synchronous circuits is developed. The developed model constitutes a well-defined framework for general timing analysis problems. Furthermore, the integration of clock skew scheduling into the timing analysis problem is introduced, which permits operation at higher operating frequencies. Section 4.3 describes the necessary set of constraints in order to derive the LP model formulation of the clock period minimization problem. The entirety of the

constraints presented in Section 4.1 (operational constraints) and Section 4.3 (constructional constraints) form the necessary set of constraints for the derivation of the LP model problem.

4.3 LP Problem Approach

As mentioned in Section 4.1, certain conditions must be satisfied for every sequentially adjacent pair of registers in a synchronous circuit in order to prevent timing hazards. These conditions are encapsulated by the four sets of operational constraints and two sets of constructional constraints. The operational constraints defined in Section 4.1 build an introductory model for the timing analysis problem of level-sensitive circuits. The iterative solution methodology discussed in Section 4.2 is built using this model.

In the presented work, a novel timing framework for level-sensitive circuits is proposed. Furthermore, a novel linear programming model for the clock period minimization problem is derived using the referred framework. This section introduces the constructional constraints, which are introduced to fulfill the completeness of the timing framework. The constructional constraints, called *validity* and *initialization* constraints, are required to ensure the correctness of the proposed LP formulation. The first type of constructional constraints, the validity constraints, are presented in Section 4.3.1. The second type of constructional constraints, the initialization constraints, are presented in Section 4.3.2.

4.3.1 Validity Constraints

The definitions of the parameters a_f, A_f, d_f and D_f require the value of a_f (d_f) to be smaller than or equal to the value of A_f (D_f):

$$A_f \geq a_f \tag{4-14}$$

$$D_f \geq d_f. \tag{4-15}$$

While the four sets of operational constraints introduced in the preceding sections summarize the timing properties of the circuit, the required *sequentiality in time* of the referred variables is not explicitly enforced. Consistency in the definitions of a_f, A_f, d_f and D_f, must be maintained through post-solution checks or

by including additional constraints. A solution leading to a result where $a_f > A_f$, for instance, is incorrect and must be disregarded.

Introducing the validity constraints [Eqs. (4-14) and (4-15)] in the LP model is preferred over performing post-solution checks for two significant reasons. The first reason is to gain the ability to easily detect the feasibility of the problem. The second reason is to preserve the automation of the solution procedure.

4.3.2 Initialization Constraints

Recall that the procedure proposed in this work is developed in order to minimize the clock period of a synchronous circuit. Besides the minimum clock period, it may prove essential to accurately calculate the nominal data arrival and departure times for each register. The initialization constraints are introduced in order to generate a consistent timing schedule for the data signal propagation in a synchronous circuit.

The sensitivity ranges of the parameters included in the LP model are not crucially heeded in the given formulation. Due to the slack [30] on data propagation times, the feasible solution set for some variables can be a range of values rather than a specific value. For instance, suppose that the earliest arrival time of a data signal at an arbitrary latch R_k can get any value in the interval $1.8 \leq a_k \leq 2.3$ without changing the minimum clock period of the circuit. For consistency, it is preferable to assign the smallest value to the earliest arrival time $(a_k = 1.8)$. In general, it is better to assign the smallest possible values to the earliest arrival and departure time variables and the largest possible values to the latest arrival and departure time variables (where applicable). Identification of such sensitivity information is essential for the consistency of the generated timing schedule for any given circuit.

Note that, the earliest and latest data arrival times at all except for the *input* registers, are set to their lowest and highest possible values, respectively, by the propagation constraints [Eqs. (4-5) and (4-6)]. The values assigned to the earliest and latest data arrival times (a, A) at the input registers do not affect the minimum clock period unless the assigned values cause the departure times to change. It may even be considered unimportant to define earliest and latest arrival time variables (a, A) at the input registers as the non-local data paths do not affect the circuit timing directly. For consistency and completeness of

the generated timing schedule, the data arrival times at the input registers are defined and the following constraints are included in the LP formulation for each input register R_l:

$$A_l = d_l - \left(D^l_{CQm} \text{ or } D^l_{DQm} \right) \ \forall R_l : |Fan - in(R_l)| = 0. \tag{4-16}$$

Note that Eq. (4-16) is only valid for input registers.

5.0 PROBLEM FORMULATION AND THE PROPOSED SOLUTION PROCEDURE

The *non-linear* $\max$ and $\min$ functions in the constraints shown in Eqs. (4-3), (4-4), (4-5) and (4-6) present a major challenge in solving the problem of minimizing the clock period. The MBM method introduced in this work is used replace the non-linear constraints with equivalent linear constraints. The equivalence between the non-linear programming (NLP) model formulation and the re-formulated linear programming (LP) model problem is preserved.

The proposed linearization method is described in Section 5.1, and the LP model is offered in Section 5.2.

5.1 Modified Big M (MBM) Method

The linearization of the constraints which exhibit non-linear behavior is a commonly applied procedure in operations research. Non-linear constraints are manipulated to derive equivalent linear constraints, which are inherently easier to solve. In this work, a collection of known linearization procedures are applied on the non-linear constraints of the timing analysis problem. The collection of these procedures is named the *Modified big M (MBM) method* [26]. It has been considered reasonable to denominate the collection of linearization procedures the *MBM method*, as the research is developed by an inspiration from the big M method [30]. The big M method is a special case of the simplex algorithm [30] which has applications in a completely distinct set of problems with respect to the MBM method. The only similarity between the big M method and the MBM method is the use of the constant M in both methods. The constant M symbolically represents a *very large* positive number used to assign an overwhelmingly large penalty to a variable in the objective function in order to increase the priority of the variable in the optimization process.

The collection of linearization procedures composing the MBM method is presented in Table 5.1. For a minimization type LP problem—subject to constraints that have $\min$ and $\max$ functions—the transformations listed in Table 5.1 are applied to replace *non-linear* constraints with linear constraints. Note that only relevant constraints and relevant portions of the objective function are included in Table 5.1.

Define a finite set N, consisting of the variables $N = \{a, b, c, \ldots, n\}$. Consider all variables in the finite set N to be elements of the real numbers set $N = \{a, b, c, \ldots, n\} \subset \Re$. The objective function Z is a linear function of the variables $\{a, b, c, \ldots, n\}$ and is defined $Z : \Re^{|N|} \rightarrow \Re$. There are no limitations on variables being inclusive, provided the linearity of the constraints is preserved.

Two different linearization scenarios are presented in Table 5.1. In the first scenario [linearization of $a = \max(b, c)$ expression], the variable a is constrained to be the greater of the variables b and c. The constraint is replaced with two new constraints, explicitly requiring the variable a to be greater than or equal to the variables b and c. The initial constraint and the relaxed constraints are equivalent if either of the following conditions holds:

1. Equality condition is observed for at least one of the inequalities, while the other inequality operation returns true.

2. Equality condition is observed for both inequalities.

The cost function denoted by the product Ma is *added* to the objective function. The product Ma is overwhelmingly large with respect to other cost functions in the objective function as a result of the highly-weighed cost figure (recall the very large coefficient M). Thus, Ma is given the highest priority in the minimization process. As a result, the greater of the variables b and c is assigned to variable a.

The relaxation method in the second scenario [linearization of $a = \min(b, c)$ expression] is also presented in Table 5.1. In this case, the cost function Ma is *subtracted* from the objective function in order to exploit the maximum value to be assigned to the variable a.

Table 5.1 Modified *Big M* transformations

$\min\ Z$	$\rightarrow$	$\min\ (Z + Ma)$
$a = \max(b, c)$	$\rightarrow$	$a \geq b$ $a \geq c$
$\min Z$	$\rightarrow$	$\min\ (Z - Ma)$
$a = \min(b, c)$	$\rightarrow$	$a \leq b$ $a \leq c$

Similar to its implementation in the big M method, the constant M is defined *sufficiently* large to ensure the equivalence of the linear and non-linear problems. The selection of a value for the constant M depends on the solution space of a specific problem (problem constraints) and the objective function Z. Typically, the number M must be chosen significantly larger than the values of any parameter in the problem. However selection of an *extremely* large M may cause the LP solver to fail drastically [3]. A value of $M = 5000$ was experimentally found to be sufficiently large for the analysis of circuits with arrival and departure times up to 100 (time units), number of registers up to 2000, and number of data paths up to 30000. The interpretation of value assignment and the derivation of a lower bound on the constant M fall outside the scope of this thesis and will not be discussed. However, verification of the equivalence between the non-linear problem and the MBM method-transformed linear problem is a straight-forward post-solution check.

5.2 LP Model

An equivalent LP model of the clock period minimization problem is generated through the application of the MBM method. There are five sets of constraints in the finalized equivalent LP model. As explained in Chapter 4, these sets are the latching [Eqs. (4-1) and (4-2)], synchronization [Eqs. (4-3) and (4-4)], propagation [Eqs. (4-5) and (4-6)], validity [Eqs. (4-14) and (4-15)] and initialization [Eq. (4-16)] constraints. Note that, for simplicity, the skew constraints [Eqs. (4-7), (4-8) and (4-9)] are not included in the LP model. The finalized LP model for the clock period minimization problem is shown in Table 5.2.

The latching, validity and initialization constraints exhibit linear behavior. Therefore, these constraints remain unchanged in both the LP and NLP models as shown in constraints (*i-ii , vii-ix*) of the formulation. The synchronization constraints, however, are formed by the $\max$ function and exhibit non-linear behavior. The MBM method is used on the synchronization constraints in order to generate equivalent linear constraints for the LP model problem (constraints *iii* and *iv*). For instance, (*iii*) depicts the replacement of the non-linear constraint presented in Eq. (4-3) with two linear constraints, where d_i is greater than or equal to both operands of the $\max$ function, $\left(a_i + D^i_{DQm}\right)$ and $\left(T - C^L_W + D^i_{CQm}\right)$. Note that the cost function Md_i is added to the objective function. Propagation constraint on the latest data arrival time (Eq. (4-6)), exhibits similar non-linearity with the synchronization constraints such that the $\max$ function is used. The

Table 5.2 The transformed constraints for the 'Modified big M' method.

	LP Model
min	$T + M[\sum_{\forall R_j}(d_j + D_j) + \sum_{\forall R_k:\|Fan-in(R_k)\|\geq 1}(A_k - a_k)]$
	subject to
(*i*)	$a_f \geq H_f$
	[Latching-Hold time]
(*ii*)	$A_f \leq T - S_f$
	[Latching-Setup time]
(*iii*)	$d_i \geq a_i + D^i_{DQm}$
	$d_i \geq T - C^L_W + D^i_{CQm}$
	[Synchronization-Earliest time]
(*iv*)	$D_i \geq A_i + D^i_{DQM}$
	$D_i \geq T - C^L_W + D^i_{CQM}$
	[Synchronization-Latest time]
(*v*)	$a_f \leq d_{i_1} + D^{i_1 f}_{Pm} + T_{skew}(i_1, f) - T$
	$\vdots$
	$a_f \leq d_{i_n} + D^{i_n f}_{Pm} + T_{skew}(i_n, f) - T$
	[Propagation-Earliest time]
(*vi*)	$A_f \geq D_{i_1} + D^{i_1 f}_{PM} + T_{skew}(i_1, f) - T$
	$\vdots$
	$A_f \geq D_{i_n} + D^{i_n f}_{PM} + T_{skew}(i_n, f) - T$
	[Propagation-Latest time]
(*vii*)	$A_f \geq a_f$
	[Validity-Arrival time]
(*viii*)	$D_f \geq d_f$
	[Validity-Departure time]
(*ix*)	$A_l = d_l - (D^l_{CQm}$ or $D^l_{DQm}), \forall R_l : \|Fan - in(R_l)\| = 0$
	[Initialization]

equivalent propagation constraints in the LP model are shown in (*vi*). In the LP model, the variable A_f is greater than or equal to the expressions $\left(\left[D_i + D^{if}_{PM} + T_{skew}(i, f)\right] - \phi_{if}\right)$, evaluated for each fan-in path of register R_f. In the formulation, fan-in paths of R_f are indexed by the parameter n.

Unlike other non-linear constraints in the formulation, the propagation constraint on the earliest arrival time a_f is modeled by the min function. In this type of linearization, a_f is set to be less than or equal to each operand of the min function. As shown in (*v*), the expressions $\left[d_i + D^{if}_{Pm} + T_{skew}(i, f) - T\right]$ evaluated for each fan-in path of register R_f are included in the finalized LP model.

In order to illustrate the derivation of the NLP and LP model formulations for a clock period mini-

mization problem, a simple synchronous circuit is investigated. The NLP model formulation of the clock period minimization problem for the sample circuit (shown in Figure A-1) is presented in Appendix A. The non-linear constraints in the NLP problem formulation are linearized using the MBM method described in Section 5.1. The finalized LP model formulation for the clock period minimization problem is presented in Appendix B.

6.0 AN EXAMPLE AND EXPERIMENTAL RESULTS

The circuit network shown in Figure 6.1 is analyzed in order to illustrate the application of the proposed procedure. Without affecting the generality of the solution, zero setup and hold times and zero internal delays are considered ($S_i = H_i = D_{CQ} = D_{DQ} = 0$).

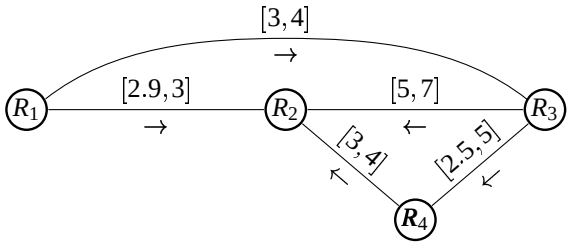


Figure 6.1 A simple synchronous circuit. Note that the minimum clock period with zero skew and using flip-flops is $T = 7$ (time units).

Given single-phase synchronization under zero and non-zero clock skew, the clock period minimization problems of three different synchronous circuits with same circuit topology are formulated. The simpler (in terms of timing analysis) circuit, which is used as the basis of comparison for other circuits, is the *zero clock skew edge-sensitive* circuit. The minimum clock period of a zero clock skew edge-sensitive circuit is defined by the maximum data propagation time in the circuit [12]. Thus, the synchronous circuit network presented in Figure 6.1 has a minimum clock period of $T = D^{32}_{PM} = 7$ (time units) when used with edge-triggered flip-flops.

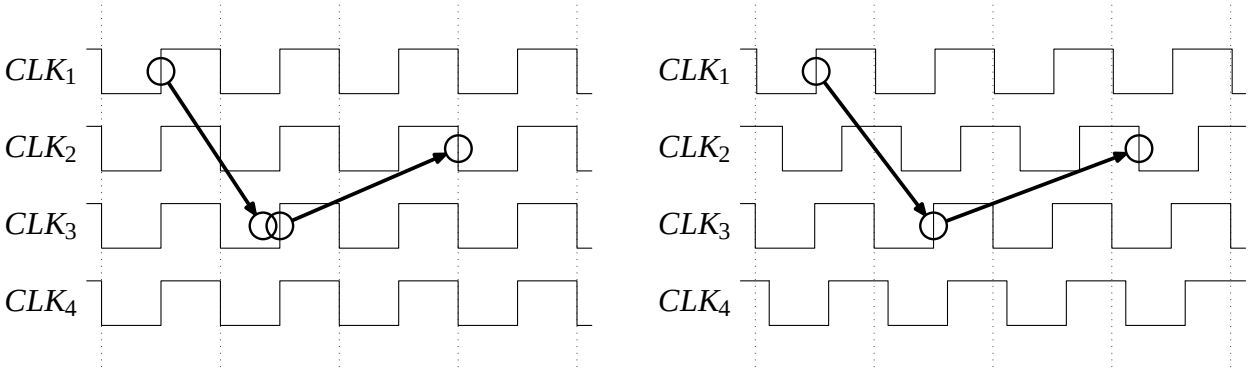
The second synchronous circuit of interest is the *zero clock skew level-sensitive* circuit. In order to design a level-sensitive synchronous circuit, each flip-flop in the given circuit topology is replaced with a level-sensitive latch. Zero clock skew level-sensitive circuits exhibit improved circuit performance due to time borrowing. Finally in the third synchronous circuit, clock skew scheduling is applied. This *non-zero clock skew level-sensitive* circuit exhibits performance improvement due to the simultaneous consideration of time borrowing and clock skew scheduling.

The commercial optimization package CPLEX [10] is used to solve for the clock period minimization problem of the generated synchronous circuits. The generic LP model constituting a fully-linear optimiza-

tion problem for such timing analysis is presented in Table 5.2. In order to solve this type of an LP problem, CPLEX implements optimizers based on simplex algorithms (both primal and dual simplex). In the experiments, the particular optimizer is automatically selected by the solver. The worst case analysis shows that the simplex method and its variants may require exponential number of steps to reach an optimal solution [3]. However, a vast amount of practice has confirmed that in most cases, the number of iterations to reach an optimal solution is a linear function of the number of variables n and a quadratic function of the problem constraints m [3]. Thus, the expected computational effort of the presented procedure is similar to the $O(m^2 n)$ of the simplex method.

Note that the number of problem constraints m is proportional to the number of registers r and the number of local data paths p in the circuit. Let s denote the number of input registers for which the initialization constraints are defined. In the LP model clock period minimization problem shown in Table 5.2, there are eight (8) constraints for each register, two (2) constraints for each local data path, and one (1) constraint for each input register. Thus, the number of constraints in the problem formulation is $m = 8r + 2p + s$. The minimum clock period T is a problem variable. Also, there are five (5) problem variables defined for each register leading to a total number of $n = 5r + 1$ variables in the problem formulation. Thus, the problem complexity is $O\left[(8r + 2p + s)^2(5r + 1)\right]$. Note that the exact computational complexity cannot be determined since the internal presolver, matrix-sparsity checker and large-scale optimizer [3, 10] routines employed within CPLEX are proprietary and unknown.

In the analysis, the minimum clock period for the zero clock skew level-sensitive circuit is calculated as 4.66 (time units), which is a 33% improvement over the zero clock skew edge-sensitive synchronous circuit. Note that the percentage improvement is calculated by the expression $100(T_{old} - T_{new})/T_{old}$. As stated before, clock skew scheduling is applied on the level-sensitive circuit in order to generate the non-zero clock skew level-sensitive circuit. The calculated minimum clock period of 4.05 for the non-zero clock skew level-sensitive circuit is a 13% improvement over the zero clock skew level-sensitive circuit and a 42% improvement over the zero clock skew edge-sensitive circuit. Note that 13% improvement is only due to clock skew scheduling, while 42% improvement is due to time borrowing *and* clock skew scheduling. Further analysis of the time borrowing and clock skew scheduling effects on circuit timing will be presented



$T = 4.66$
Zero clock skew

$T = 4.05$
Non-zero clock skew

Zero Skew	Non-Zero Skew	Critical Path
$A_3 = 1.66 = D_1 + 4 - 4.66$	$A_3 = 2.025 = D_1 + 4 + (0.05 - 0) - 4.05$	$R_1 \rightarrow R_3$
$A_2 = 4.66 = D_3 + 7 - 4.66$	$A_2 = 4.05 = D_3 + 7 + (0 - 0.925) - 4.05$	$R_3 \rightarrow R_2$

Figure 6.2 Zero clock skew and non-zero clock skew clocking schedules for the synchronous circuit in Figure 6.1. The clocking schedule for the zero clock skew circuit is shown on the left, with a minimum clock period of $T = 4.66$. Non-zero clock skew scheduling results with a minimum clock period of $T = 4.05$ is shown on the right. For non-zero clock skew scheduling, the optimal clock signal delays at the register are $t_{R_1} = 0.05$, $t_{R_2} = 0.925$, $t_{R_3} = 0$ and $t_{R_4} = 0.475$. The arrows represent data signal propagation on the respective critical paths. Note that unlike the presented case, the critical paths for zero and non-zero clock skew scheduling need not be identical.

in Section 6.2. The clocking schedules and the data propagation on the critical paths of the circuit in Figure 6.1 in case of zero and non-zero clock skew scheduling are shown in Figure 6.2.

6.1 Digital Synchronous Circuit State of Operation

Presence of data path loops (cycles) and transient state errors are two major issues that needs to be identified in the timing analysis of level-sensitive circuits. As discussed in Section 4.2, the iterative solution procedure offered in [18] was shown to suffer from excessive run-times and produce false negative outputs in presence of data path loops [24]. In [24], modifications are offered for the iterative algorithm in order to detect and handle the effects of data path loops in the circuit. Also in [24], it has been shown that synchronous circuits are prone to suffer from transient state errors. The transient state errors occur due to the non-unique solution sets of the problem parameters [24]. In circuits under transient state errors, setup violations occur in certain registers after the system is initiated from a reset state. The arrival and departure times may not

be stable at start-up, in which case these times change during initial clock cycles, constituting the transient state. As circuit operation progresses in time, the arrival and departure times converge to their steady-state values.

There are two major conventions in evaluating the transient errors and determining the steady-state behavior. The first convention overlooks the transient errors and presumes that the departure times converge to the opening edge of the driving clock, which is the expected schedule for the steady-state of operation. The second convention is more strict in that transient state errors are not permitted. The first convention is more widely accepted within the proposed solution algorithms and leads to a generally acceptable solution unless the transient state operation of the level-sensitive circuit is decisive to overall circuit operation. Given that the second convention is adopted, the reset state is preferably extended until the steady state of operation is reached [24].

The LP model proposed in this work assumes the transient-state operation of a level-sensitive circuit to be negligible. The aim of the generated model is to solve for the steady-state timing scheduling problem. The simplex algorithm-based LP solver directs the gradual advancement of parameter values as they are enforced by the LP model (Table 5.2). Previously offered algorithms are vulnerable to potential fallacies caused by data path loops due to their iterative nature. However, in the presented procedure, complications posed by the presence of data path loops are resolved within the mechanics of the LP solver without significantly affecting the run-time or quality of the solution. If the problem remains feasible, the timing parameters for the steady state operation of the circuit are calculated.

In order to illustrate the described phenomenon, the steady-state optimal timing schedule for the IS-CAS'89 benchmark circuit *s27* is presented in Figure 6.3. The circuit *s27* has one input register and a data path loop consisting of two other registers. The data signal departs from input register R_3 and perpetually propagates on the loop between R_1 and R_2. The minimum clock period is calculated to be 4.1, where the calculated propagation delays are indicated on the circuit graph. Note that the propagation delays are calculated to be constants using the authors' generic delay calculation procedure. This fact does not effect the generality of the solution and the inclusion of variable propagation delay in the problem solution is straightforward.

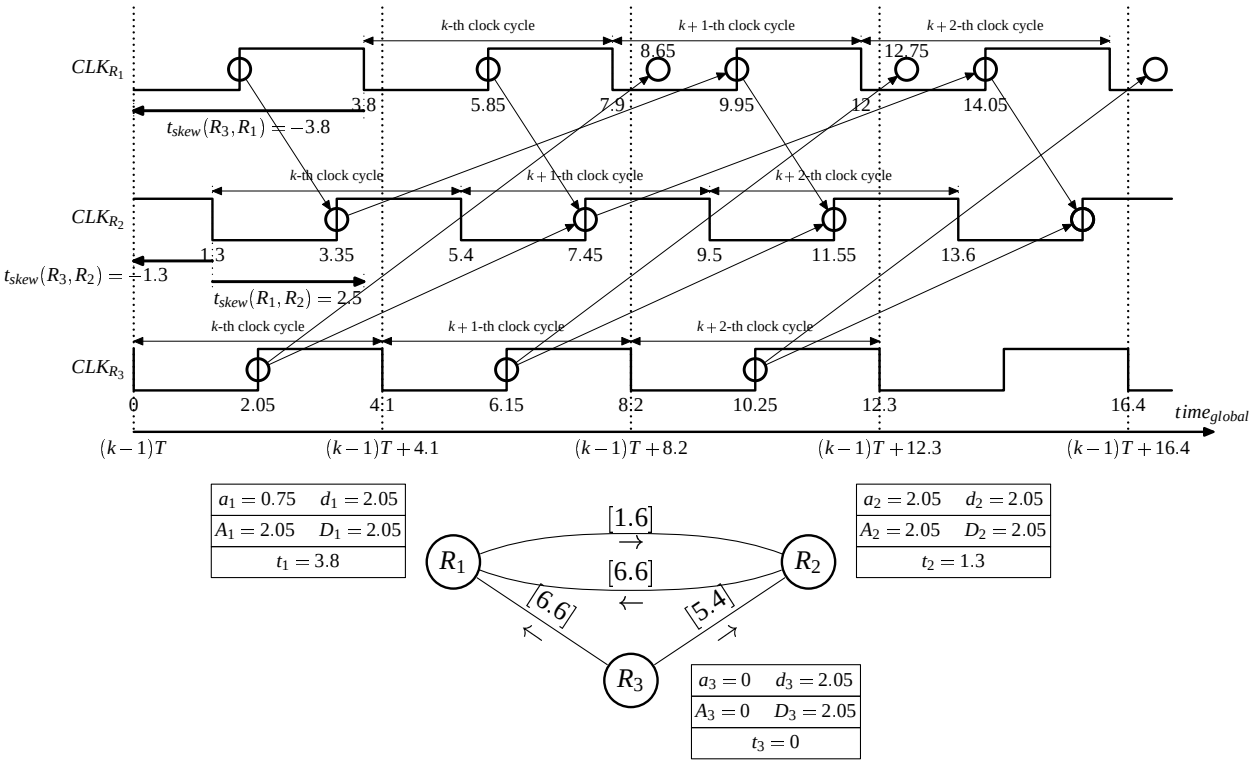


Figure 6.3 The optimized timing schedule for the benchmark circuit *s27* operable with a minimum clock period of $T = 4.1$. Note that $D^{if}_{Pm} = D^{if}_{PM}, \forall R_i \rightarrow R_f$ and $S_i = H_i = D^i_{CQ} = D^i_{DQ} = 0$ are considered.

In Figure 6.3, the data propagation occurring on all data paths of the $s27$ benchmark circuit is analyzed. The clock signals CLK_{R_1}, CLK_{R_2} and CLK_{R_3}, where the subscripts indicate the register being synchronized by the clock signal, build the frame for the analysis. The clock signals may not be completely aligned in time due to the non-identical clock signal delays to the respective registers. The clock signal CLK_{R_3} at the input register R_3 has no delay in time with respect to the clock signal at the clock source $(t_3 = 0)$. Hence, the origin of the clock signal at the source is aligned with the origin of CLK_{R_3}. The clock signals CLK_{R_1} and CLK_{R_2} however, are shifted in time by $t_1 = 3.8$ and $t_2 = 1.3$ relative to the origin of the clock signal at the source. The horizontal axis of Figure 6.3 represents the time, where the beginning $(k-1)T$ of the k-th clock cycle of CLK_{R_3}, is defined as the local time reference, with an assigned value of zero. In Figure 6.3, the numbers associated with the enabling and latching edges of the clock signals label the times with respect to the local time reference. The arrows illustrate the propagation between the registers and are drawn to scale. Illustration of the data propagation on three consecutive clock cycles are sufficient to

analyze the behavior of the data path loop of the benchmark circuit $s27$; the k-th, $(k+1)$-th and $(k+2)$-th clock cycles are selected to illustrate the behavior. The solid arrows represent the data propagation during the selected clock cycles. For instance, the propagation between R_3 and R_1 is represented by the arrows initiating from the CLK_{R_3} row at times 2.05 and 6.15, and concluding at the CLK_{R_2} row at times 8.65 and 12.75, respectively. Data propagation on the data path loop between the registers R_1 and R_2 is visible by the cross-structured arrows initiating and concluding in the corresponding clock signal rows. Note that the calculated nominal arrival and departure times are illustrated on the circuit graph, inside the boxes associated with each node.

In steady-state of operation, the departure times of the registers that constitute a data path loop converge to the beginning of their respective clock cycles. The circuit *s27* in Figure 6.3 is scrutinized in order to provide a better insight on how the latest departure times converge to a certain value in the steady-state. Define a variable ϵ, where ϵ is a very small period of time. Suppose that a deviation of ϵ occurs in the departure time of the data signal from R_3. The signal departure from R_3 occurs at time *2.05+*ϵ, delaying the arrival times at R_1 and R_2 by ϵ. The departure from R_2 is gradually delayed by ϵ every turn, which in turn delays the arrival time at R_1. The arrival and departure times cumulatively increase in each turn of the data signal around the loop. Eventually, the signal arrivals at the latches occur during the non-transparent state of the latches. At this point, the signal departure times return to their starting values, which are the latching edges of their respective clock cycles. It is evident that the arrival times will finally be restored to their initial values when the source of the deviation vanishes. Thus, the assignment of the time-varying departure times to the enabling edges of the synchronizing clock signals is referred to as the steady-state of operation for the synchronous circuit.

6.2 Performance Results of the Procedure on the ISCAS'89 Benchmark Circuits

The timing analysis algorithm described in this thesis is applied on the selected suite of ISCAS'89 benchmark circuits in order to derive the performance results and illustrate the efficiency of the presented algorithm. The original ISCAS'89 benchmark circuits are edge-sensitive synchronous circuits and the specific timing information of different circuit elements is not defined. A generic delay calculation procedure

is generated in order to provide timing information for each local data path. The data propagation times $\left(D_P^{if}\right)$ on local data paths are calculated using pre-determined delay times for each logic gate type. The number of fanout branches on each node is also considered effective on the data propagation time.

The level-sensitive circuit is generated by replacing each flip-flop in the original benchmark circuit with a level-sensitive latch as explain in Chapter 6. Note that this procedure does not affect the operation of the original circuit and preserves the circuit topology. In experimentation, 50% duty cycle is selected both for the single phase clock signal. Without affecting the generality of the solution, the setup and hold times and the internal delays are assumed to be zero ($S_i = H_i = D_{CQ} = D_{DQ} = 0$). The consideration of these numeric constants in an actual problem is straightforward. *Edge-sensitive* and *level-sensitive* synchronous circuit implementations are analyzed for *zero* and *non-zero clock skew scheduling* applications. The effects of time borrowing and clock skew scheduling in circuit implementation are investigated. The results of the analyses—computed on a 440MHz Sun Ultra-10 Workstation—are presented in Table 6.1. For each circuit, the following data are listed—the circuit name, the number of registers r and the number of paths p, the clock periods T_{FF}^{noskew} for a zero skew circuit with flip-flops, T_L^{noskew} for a zero skew circuit with latches, T_{FF}^{skewed} for a non-zero skew circuit with flip-flops, T_L^{skewed} for a non-zero skew circuit with latches, and T_L^r for a non-zero skew circuit where the clock delays to I/O registers are restricted to be equal. The subscripts FF, L represent circuit topologies for flip-flop based and latch-based circuits, respectively. The superscripts $noskew, skewed$ indicate zero or non-zero clock skew scheduling. Also listed are the calculation time of T_L^{skewed}, t_L^{skewed}, and the clock period improvements I_L^{TB}, I_{FF}^{CSS} and I_L^{TBCSS}, where the superscripts $TB, CSS, TBCSS$ stand for time borrowing, clock skew scheduling and both, respectively.

The minimum clock periods calculated for the edge-sensitive synchronous circuits under zero and non-zero clock skew scheduling (T_{FF}^{noskew} and T_{FF}^{skewed}, respectively) are borrowed from [12]. It is reported in [12] that, due to clock skew scheduling, an average improvement of 30% is reported in the minimum clock period for the ISCAS'89 benchmark circuits.

The experimental results shown in Table 6.1 represent significant improvements in the minimum clock period for synchronous circuits with level-sensitive latches. In digital synchronous circuits, utilizing latches as memory elements instead of flip-flops may result in up to 33% improvement of the minimum clock

Table 6.1 ISCAS'89 benchmark circuits results showing the number of registers r and paths p (before modification). Optimal clock periods, improvements and calculation time are denoted by T, I and t, respectively. Subscripts FF, L represent circuit topologies for flip-flop based and latch-based circuits, respectively. Superscripts $noskew, skewed, r$ indicate zero or non-zero clock skew and restricted circuit (for clock periods only), and $TB, CSS, TBCSS$ stand for time borrowing, clock skew scheduling and both, respectively.

Circuit Info			**Zero CS**		**I (%)**	**Non-Zero CS**		**I (%)**			**T (sec)**	**R**	**I (%)**
Circuit	r	p	T_{FF}^{noskew}	T_{L}^{noskew}	I_{L}^{TB}	T_{FF}^{skewed}	T_{L}^{skewed}	I_{FF}^{CSS}	I_{L}^{TBCSS}	I_{L}^{CSS}	t_{L}^{skewed}	T_{L}^{r}	I_{L}^{r}
s27	3	4	6.6	5.4	18	4.1	4.1	38	38	24	0.02	4.1	38
s208.1	8	28	12.4	8.6	31	4.9	5.2	60	58	40	0.01	7.6	39
s298	14	54	13	10.6	18	9.4	9.4	28	28	11	0.02	10.6	18
s344	15	68	27	18.4	32	18.4	18.4	32	32	0	0.03	18.4	32
s349	15	68	27	18.4	32	18.4	18.4	32	32	0	0.03	18.4	32
s382	21	113	14.2	10.3	27	8.5	8.5	40	40	17	0.04	8.72	39
s386	6	15	17.8	17.3	3	17.3	17.3	3	3	0	0.03	17.3	3
s400	21	113	14.2	10.4	27	8.6	8.6	39	39	17	0.05	8.8	38
s420.1	16	120	16.4	12.6	23	6.8	7.2	59	56	43	0.04	10.27	37
s444	16	113	16.8	12.4	26	9.9	9.9	41	41	20	0.07	9.9	41
s510	6	15	16.8	14.8	12	14.8	14.3	12	15	3	0.02	14.8	12
s526	21	117	13	10.6	18	9.4	9.4	28	28	11	0.05	10.6	18
s526n	21	117	13	10.6	18	9.4	9.4	28	28	11	0.05	10.6	18
s641	19	81	83.6	66.2	21	61.9	61.9	26	26	6	0.05	63.1	25
s713	19	81	89.2	71.2	20	63.8	63.8	28	28	10	0.05	65	27
s820	5	10	18.6	18.3	2	18.3	18.3	2	2	0	0.01	18.3	2
s832	5	10	19	18.8	1	18.8	18.8	1	1	0	0.01	18.8	1
s838.1	32	496	24.4	20.6	16	8.3	9.1	66	63	56	0.28	15.6	36
s938	32	496	24.4	20.6	16	8.3	9.1	66	63	56	0.31	15.6	36
s953	29	135	23.2	21.2	9	18.3	18.3	21	21	14	0.10	21.2	9
s967	29	135	20.6	17.9	13	16.2	16.6	21	19	7	0.08	17.9	13
s991	19	51	96.4	91.6	5	79.4	79.4	18	18	13	0.02	79.4	18
s1196	18	20	20.8	16	23	10.8	7.8	48	63	51	0.03	16	23
s1238	18	20	20.8	16	23	10.8	7.8	48	63	51	0.01	16	23
s1423	74	1471	92.2	86.4	6	77.4	75.8	16	18	12	1.10	75.8	18
s1488	6	15	32.2	29	10	29	29	10	10	0	0.02	29	10
s1494	6	15	32.8	29.6	10	29.6	29.6	10	10	0	0.01	29.6	10
s1512	57	415	39.6	34.8	12	34.8	34.8	12	12	0	0.28	34.8	12
s3271	116	789	40.3	29.8	26	28.6	28.6	29	29	4	0.69	29	28
s3330	132	514	34.8	23.4	33	17.8	17.8	49	49	24	0.49	23.2	33
s3384	183	1759	85.2	77.4	9	67.4	67.4	21	21	13	1.88	76.2	11
s4863	104	620	81.2	75.4	7	69	69	15	15	8	0.64	69	15
s5378	179	1147	28.4	23.2	18	22	22	23	23	5	1.66	22	23
s6669	239	2138	128.6	124.6	3	109.8	109.8	15	15	12	3.62	109.8	15
s9234	228	247	75.8	64.8	15	54.2	54.2	28	28	16	4.59	59.2	22
s9234.1	211	2342	75.8	64.8	15	54.2	54.2	28	28	16	3.88	59.2	22
s13207	669	3068	85.6	67.4	21	57.1	57.1	33	33	15	14.86	57.1	33
s15850	597	14257	116	92.8	20	83.6	83.6	28	28	10	76.96	83.6	28
s15850.1	534	10830	81.2	71.4	12	57.4	57.4	29	29	20	58.89	57.4	29
s35932	1728	4187	34.2	34.1	0	20.4	20.4	40	40	40	80.03	20.4	40
s38417	1636	28082	69	54.8	21	42.2	42.2	39	39	23	603.49	43	39
s38584	1452	15545	94.2	76.4	19	65.2	65.2	31	31	16	321.74	64.8	31
Average	204	2141	44.7	38.1	15	29.6	32.6	30	27	14	28.01	34.29	23.74

period under zero clock skew. On the ISCAS'89 suite of benchmark circuits for instance, an average of 15% improvement is observed when the flip-flops are replaced by latches (under zero clock skew). The recorded improvement is solely due to time borrowing.

Utilizing non-zero clock skew, an even higher improvement is possible: up to 63% improvement—over flip-flop based synchronous circuit with zero clock skew—is observed. The average improvement in the minimum clock period in this case is calculated to be 27%. The recorded improvement is due to simultaneous application of clock skew scheduling and consideration of time borrowing. Note that the functional characteristics of any synchronous circuit must not be affected by replacing the flip-flops with level-sensitive latches. In addition, the clock skew distribution of a circuit block must be performed considering additional application-specific constraints such as global layout design restrictions on placement and routing.

As mentioned earlier, the improvement in the minimum clock period for non-zero clock skew level-sensitive circuits is due to simultaneous consideration of time borrowing and application of clock skew scheduling. The improvement due to time borrowing is 15% and the improvement due to clock skew scheduling is 14%. It is interesting to note that the improvements achieved through time borrowing and clock skew scheduling are not fully additive in defining the overall improvement. Time borrowing and clock skew scheduling are contradictory effects in performance improvement, thus leading to the degradation in the overall improvement. There is a limited amount of slack propagation time on the critical paths and a circuit where time borrowing is abundantly realized, cannot benefit as much from clock skew scheduling. It has been shown however, that even though time borrowing and clock skew scheduling are battling effects, dramatically shorter clock periods are achievable through the collaboration of both effects.

The zero clock skew level-sensitive circuit implementation is analogous to the circuits previously analyzed in $[1, 25]$, which also solve for the clock period minimization problem. Note that unlike the unit-delay-per-gate approach used in $[1, 25]$, the combinational logic delay is calculated by assuming different delay times for each logic gate type and considering effects of fanout on the propagation time. Thus, the obtained results are not *directly* comparable to the previously published algorithm results. However, presuming the accuracy and correctness of both procedures, the listed results for T_L^{noskew} present the state of improvement achieved through previous work in the literature.

It is evident that a fair comparison of the run times for the previous and presented procedures is not feasible due to the differences in the problem formulations. Without any formal proof, assuming that the procedure described in this thesis and previously published algorithms are equivalent formulations for the clock period minimization problem, the calculated minimum clock periods T_L^{noskew} are used in order to compare the improvements achieved through both approaches. As will be discussed shortly, simultaneous consideration of time borrowing and clock skew scheduling in the proposed procedure results in higher improvements (I_L^{TBCSS}) compared to consideration of time borrowing and constant clock skew in the previously published algorithms. Therefore, the procedure presented in this work is superior in terms of circuit improvement. A comparison of algorithm run-times is not rational, however, as the problem formulations are nonidentical.

6.3 Verification and Interpretation of Results

Certain synchronous circuits are inoperable with level-sensitive latches or fail to satisfy the timing constraints due to predetermined circuit or clock tree topologies. In such circuits, the minimum clock period problem is infeasible. The proposed timing analysis procedure easily detects the infeasibility of a problem and provides diagnostics messages. The slack and excess values associated with each constraint can be examined in the sensitivity analysis output provided by the LP solver. Even though the details will not be discussed here, careful interpretation of the sensitivity output leads to the identification of the necessary modifications on the circuit topology to achieve the desired operating frequency. The sensitivity analysis output of the LP solver CPLEX for the timing analysis discussed in Appendices A and B is presented in Appendix C.

The interpretation of the timing schedule for a synchronous circuit presents a model to investigate the effects of zero and non-zero clock skew scheduling on synchronous circuit operation. In the rest of this section, the timing schedules generated for the synchronization of the ISCAS'89 benchmark circuit *s938* with zero and non-zero clock skew scheduling are analyzed. The analyses include the data distributions for various parameters, which are presented in Section 6.3.1. The verification of clock skew values is discussed

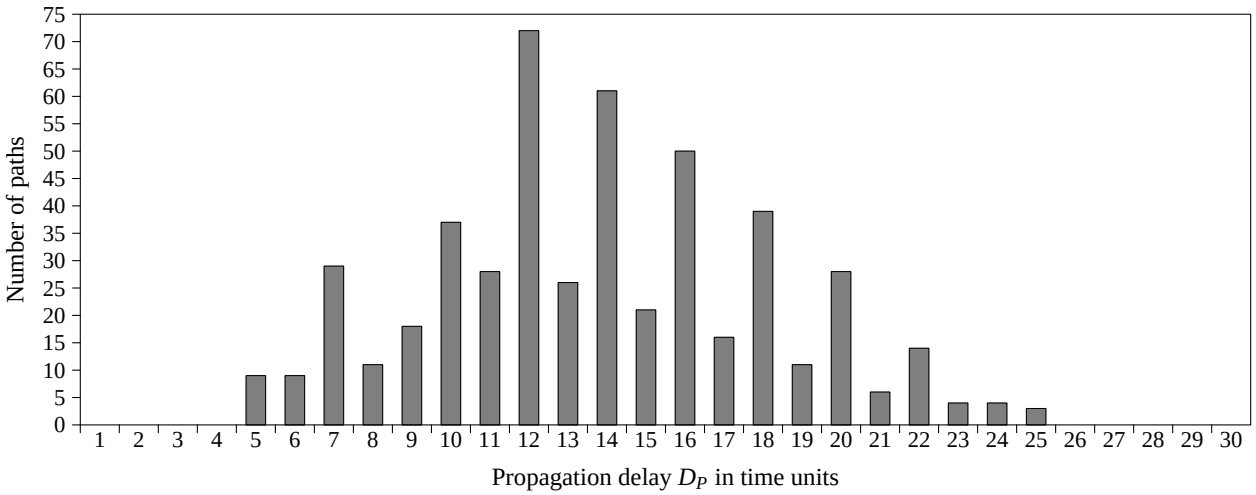


Figure 6.4 Distribution of data propagation times for *s938* with $r = 32$ registers and $p = 496$ data paths. The height of each bar corresponds to the number of paths within a given delay range. For example, there are nine (9) paths with delays between 4 and 5 time units.

in Section 6.3.2. In Section 6.3.2, the skew constraints of Section 4.1.4 are used to derive lower and upper bounds on clock skew.

6.3.1 Parameter Data Distributions

In Section 3.2, data propagation time D_P^{if} is defined as the period of time the data is processed in the combinational logic block of a local data path $R_i \rightarrow R_f$. Without loss of generality, an empirical calculation method is used to calculate the data propagation times of each local data path of a circuit (a simple fan-out delay model is used as timing data is not included in the ISCAS'89 benchmark circuits). The distribution of the calculated data propagation times for the ISCAS'89 benchmark circuit $s938$ is illustrated in Figure 6.4.

Define the *effective path delay* [12] as the time period between the departure of the data signal from the initial register and the arrival of the same data signal at the final register. The effective path delay of a local data path differs from data propagation delay, because of the additional propagation time provided by clock skew and the time borrowing property of level-sensitive synchronous circuits. Note that in level-sensitive synchronous circuits, the effective path delay is defined within a permissible range instead of a fixed value, as the arrival and departure times are indeterminate. The nominal effective path delay is determined when the arrival and departure times are realized in run-time as certain values in the permissible ranges $[a_f, A_f]$

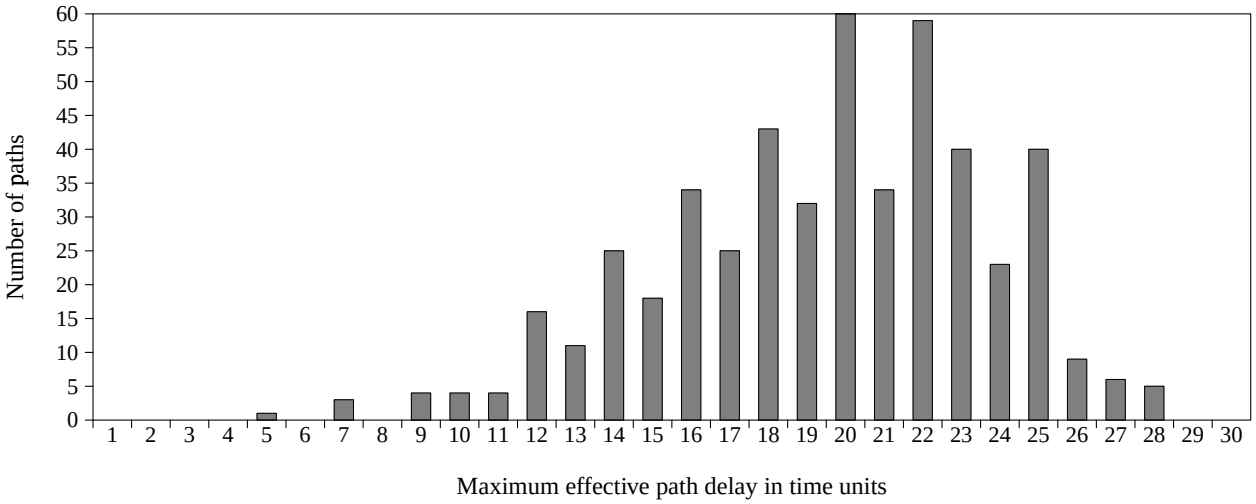


Figure 6.5 Distribution of the maximum effective path delays in data paths of *s938* for zero clock skew. The target clock period is T = 20.6. The height of each bar corresponds to the number of paths with an effective path delay within a given range.

and $[d_i, D_i]$, respectively. Specifically, the shortest effective path delay occurs when the data signal departs at its latest time D_i from the initial register R_i and arrives at its earliest arrival time a_f at the final register R_f. The longest effective path delay is realized by the earliest departure d_i of the data signal from R_i and latest arrival A_f at R_f. Hence, the interval for the effective path delay of level-sensitive synchronous circuits can be defined as:

$$a_f - D_i - T_{skew}(i,f) + T \leq \text{Effective path delay} \leq A_f - d_i - T_{skew}(i,f) + T. \qquad \text{(6-1)}$$

In this work, the longest effective path delay is investigated in order to illustrate the effects of clock skew and time borrowing on data propagation. The aim is to observe the increase in the effective path delay of a circuit, which in turn leads to a higher operating frequency, by replacement of flip-flops with latches and introducing non-zero clock skew. Observe that the distribution of the propagation delays for the $s938$ benchmark circuit presented in Figure 6.4 is exactly the same as the distribution of the effective path delay of the same benchmark circuit $s938$, when operational with flip-flops (under zero-clock skew). In circuits with flip-flops, the effective path delays are determinate $\left[D_P^{if} - T_{skew}(i,f)\right]$ as the data departures occur at the active transition of the clock signal.

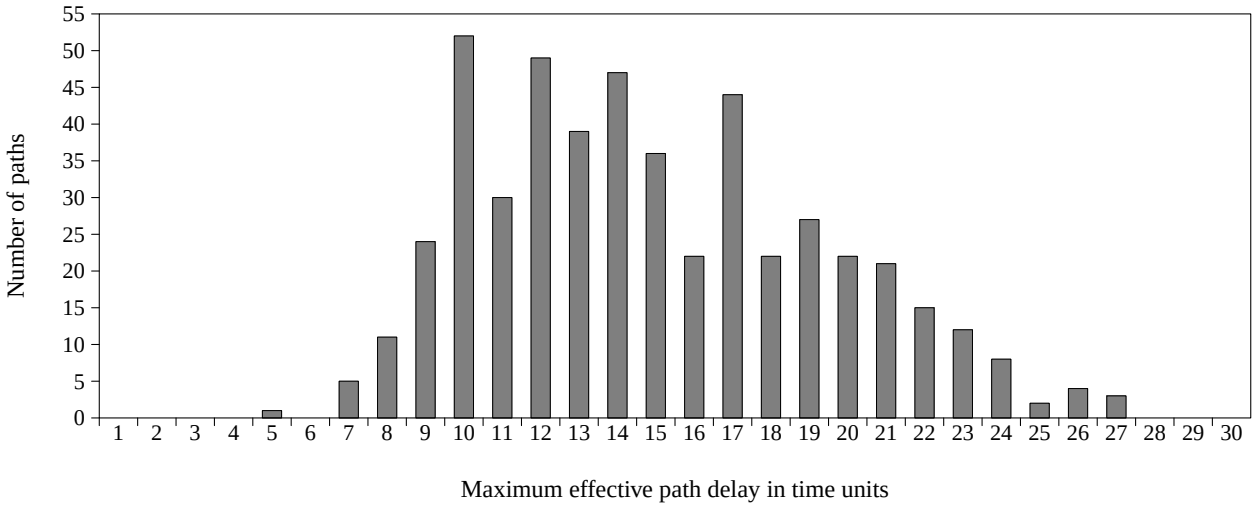


Figure 6.6 Distribution of the maximum effective path delays in data paths of *s938* for non-zero clock skew. The target clock period is T = 9.085714. The height of each bar corresponds to the number of paths with an effective path delay within a given range.

The distribution of the maximum effective path delays of the level-sensitive *s938* circuit with zero clock skew scheduling is shown in Figure 6.5. Note that the maximum effective path delay is calculated by the expression $[A_f - d_i - T_{skew}(i,f) + \phi_{if}]$. It is observed by comparing Figures 6.4 and 6.5 that the maximum effective path delays are *increased* in the level-sensitive circuit, as well as providing a smaller minimum clock period $\left(T_{FF}^{noskew} = 24.4 \text{ v.s. } T_L^{noskew} = 20.6\right)$. The increase in the effective path delays is due to time borrowing. Cumulation of effective path delay values slightly below or above the minimum operating clock period $T = 20.6$ is visible. Note that the effective path delay having larger values than the minimum clock period is a sufficient but not a necessary condition for time borrowing. Thus, local data paths where the effective path delay is calculated to be smaller than $T = 20.6$ may still benefit from time borrowing. Furthermore, it can be observed that certain data paths in the circuit benefit more from time borrowing, realizing an effective path delay close to the theoretical limit of $\left[\phi_{if} + C_W^L - T_{skew}(i,f)\right]$.

6.3.2 Skew Analysis

As discussed throughout this thesis, non-zero clock skew scheduling in synchronous circuits permits smaller clock periods. Note that in presence of non-zero clock skew, the effective path delay for the data signal over a data path most likely gets *smaller* compared to its value observed in zero clock skew schedul-

ing. This fact is directed by Eq. (6-1) (T gets smaller). However, as the minimum clock period T gets smaller, the percentage of the data paths, on which the effective path delay exceeds the minimum clock period, significantly increases (see Figure 6.6). The effect of clock skew on improving the minimum clock period is visible by comparing the histograms presented in Figures 6.5 and 6.6.

The skew constraints [Eqs. (4-7), (4-8) and (4-9)] introduced in Section 4.1.4 can be included in the LP model (Table 5.2) in order to ensure the correctness of the solution. The skew constraints not only constitute an extra measure to check for the feasibility of the solution but are also used in collecting statistical data on clock skew values. Interpretation of Eqs. (4-11) and (4-13) lead to the upper and lower bound definitions for the clock skew. In order to generate an expression for the upper bound, Eq. (4-11) rewritten as:

$$D_i + D_{PM}^{if} - \phi_{if} + T_{skew}(i,f) \leq T - S_f. \tag{6-2}$$

In Eq. (6-2), the earliest possible time is assigned to D_i in order to realize the upper bound on clock skew. The earliest possible time that a data signal departs from a latch is D_{CQ} later than the leading edge of the clock signal, $\left(T - C_W^L + D_{CQ}\right)$. Reordering the expression gives the upper bound on clock skew:

$$T_{skew}(i,f) \leq \phi_{if} + C_W^L - D_{PM}^{if} - D_{CQ} - S_f. \tag{6-3}$$

The lower bound on the clock skew is derived similarly from Eq. (4-13), which leads to:

$$a_f + D_{Pm}^{if} \geq \phi_{if} - T_{skew}(i,f) + H_f. \tag{6-4}$$

In order to derive the lower bound, the data arrival time at R_f must be considered to occur at its latest possible time. The latest data arrival time is the setup time S_f earlier than the trailing edge of the clock signal, $T - S_f$. Thus, the lower bound on the clock skew is:

$$T_{skew}(i,f) \geq \phi_{if} - T - D_{Pm}^{if} + S_f + H_f. \tag{6-5}$$

Combining Eq. (6-3) and Eq. (6-5), the theoretical limits on clock skew is expressed as follows:

$$\phi_{if} - T - D_{Pm}^{if} + S_f + H_f \leq T_{skew}(i,f) \leq \phi_{if} + C_W^L - D_{PM}^{if} - D_{CQ} - S_f. \tag{6-6}$$

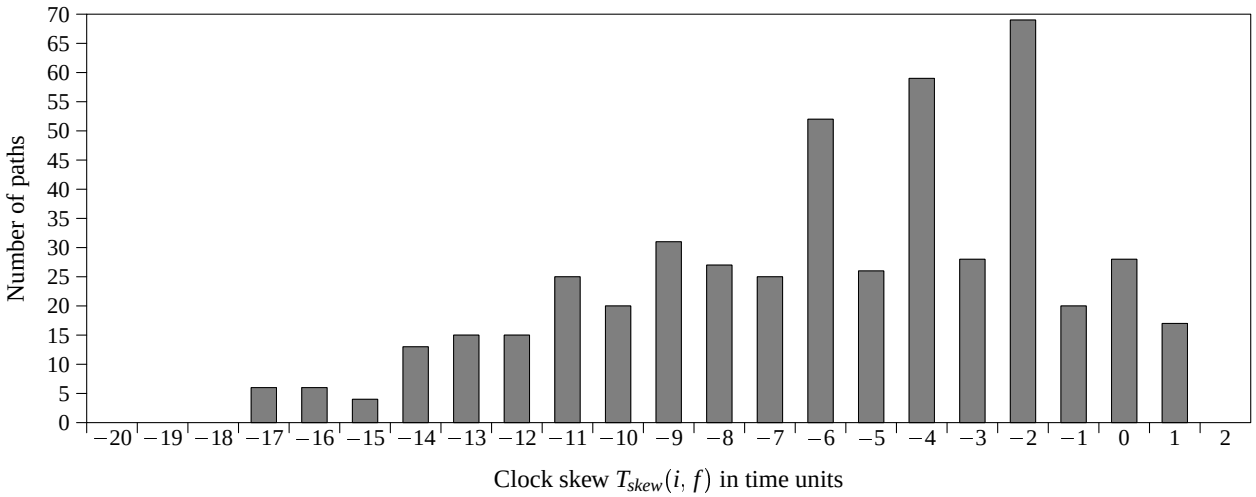


Figure 6.7 Distribution of the clock skew values of the non-zero clock skew case for *s938*. The target clock period is $T = 9.085714$. The height of each bar corresponds to the number of paths formed by sequentially adjacent pair of registers which have a clock skew within the given range.

Recall that in experimentation, the parameters D_{DQ}, D_{CQ}, S_f, H_f are considered zero and 50% duty cycle is selected for the single-phase synchronization clock signal. In order to evaluate the upper and lower bounds on clock skew in this simplified case, the parameters are substituted in Eq. (6-6):

$$-D_{Pm}^{if} \leq T_{skew}(i, f) \leq 1.5T - D_{PM}^{if}. \tag{6-7}$$

Specifically on the ISCAS'89 benchmark circuit *s938*, the clock skew bounds are verified using the experimental values shown in Figure 6.4. For the benchmark circuit *s938* with a minimum clock period of 9.09, the minimum and maximum propagation delays are calculated to be 5 and 24.4, respectively. Thus, the value set for the clock skew variable on the data paths of *s938* is constrained by $-24.4 \leq T_{skew}(i, f) \leq 8.64$.

The distribution of the clock skew values of *s938*, when operable with a minimum clock period of 9.09, is presented in Figure 6.7. The calculated clock skew values are within the derived limits, most of which are negative. Negative clock skew between registers help improve the minimum clock period of the synchronous circuit due to the additional time it provides for data signal propagation. The data paths, on which positive skew is recorded, most likely occur due to two reasons. The first reason is the presence of data path loops within the circuit. The second reason are the—faster—paths which provide extra time for neighboring critical paths.

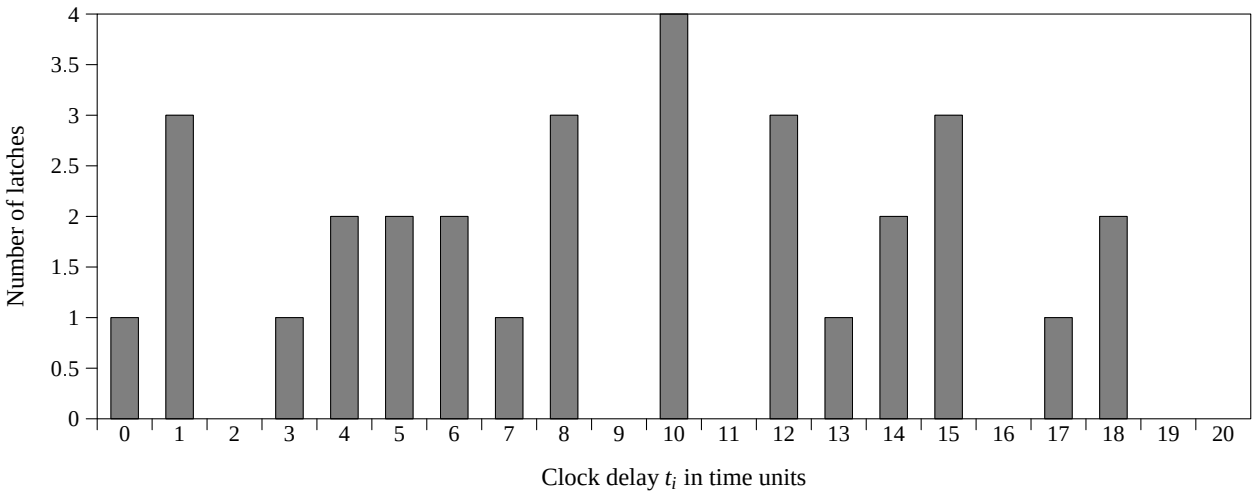


Figure 6.8 Distribution of the clock delay values of the non-zero clock skew case for *s938*. The target clock period is $T = 9.085714$. The height of each bar corresponds to the number of latches being driven by a clock signal with a time delay within the given range.

The distribution of the clock delays to each register presented in Figure 6.8. The distribution is significantly wide-spread, ranging from 0 to 19 (time units), where the minimum clock period is $T = 9.09$. If the clock tree network of the synchronous circuit is implemented to accommodate for these nominal clock delays, operation at the target minimum clock period is achieved.

6.4 Further Considerations

The presented LP model formulation is demonstrated to be effective for the static timing analysis of synchronous circuits. In the analyses, classical circuit implementations are investigated, such that, no additional timing dependencies between the registers of a synchronous circuit, other than the dependencies leading to the predefined timing constraints, are prescribed. As the application-specific integrated circuit (ASIC) design techniques become wide-spread and with the growing impact of secondary effects on the operation of sub-micron devices, the need for a timing analysis model, merited to accommodate for application-specific constraints, becomes essential. Unlike the previous work, the presented formulation is highly amenable to such modifications, constituting a well-defined timing analysis framework.

The following describes a potential problem in the timing analysis of SOC designs. In an ASIC/SOC implementation, the clock signal distribution between different IP blocks (or clock domains) are subject to consideration as well as the distribution of the clock signal within an IP block. The clock delays to the *I/O* registers of a synchronous IP block are less flexible compared to the clock signal delays to the *internal* registers. It is likely that the timing analysis will be performed on individual IP blocks by the vendors, without a priori information of the application environment. Therefore, timing violations may occur on outgoing (non-local, intra-block) data paths, as the clock skew on these data paths will be unaccounted for in the initial computation. A simple solution to avoid timing violations between the IP blocks is to equalize all I/O register clock delay values. In the presented framework, additional timing constraints enforcing the equality of the clock signal delays can easily be integrated into the constraints set of the LP model problem presented in Table 5.2.

Another commonly encountered design constraint is to implement a predetermined—possibly non-optimal—clock tree network for the synchronous circuit. If the clock tree topology is predetermined, the minimum clock period problem must be solved with known clock delays to each register. The LP model presented in Table 5.2 can easily be modified to account for such changes, by assigning the given clock signal delays to the respective clock delay variables.

7.0 CONCLUSIONS AND FUTURE WORK

The timing analysis and optimization of synchronous circuits are subject to non-zero clock skew (intentional or not) and other effects of process parameter variations. A novel timing analysis procedure is introduced, which considers the simultaneous application of time borrowing and clock skew scheduling to improve the performance of level-sensitive synchronous circuits. The described procedure is the first to integrate non-zero clock skew scheduling in the timing analysis of level-sensitive circuits. The procedure is based on a stand-alone LP model formulation (to be solved by any standard LP solver) which constitutes a novel timing analysis framework for level-sensitive synchronous circuits. The timing framework is formulated for a single-phase synchronization scheme. The optimal clocking and timing schedules for data propagation between registers are computed as a result of the timing analysis.

In this thesis, the described timing analysis framework is used to automate the clock period minimization problem of level-sensitive synchronous circuits. Accurate timing analyses of relatively bigger benchmarks are performed by using the MBM method in order to generate equivalent LP model problems. The generated LP model formulation is sufficiently general and can be modified to accommodate application-specific constraints and timing properties. The stand-alone nature of the presented timing framework supports easy adaptation to various timing optimization problems such as the clock period verification problem and statistical timing analysis.

The work presented in this thesis has been published in [26–28]. In [26, 27] and the thesis, single-phase synchronization of level-sensitive circuits is analyzed. The proposed formulation (Section 4.3) and solution (Section 5) procedures can be modified so that these procedures apply to multi-phase synchronized circuits. The formulation of the timing analysis of level-sensitive circuits for multi-phase synchronization is addressed in [28]. Other enhancements on the formulation of the timing analysis of synchronous circuits for multi-phase synchronization are among future directions of research.

A potential direction of research is to develop an algorithm to determine the optimal number of clock phases for any given level-sensitive circuit. It is shown in [28] that the optimal number of clock phases for any two level-sensitive circuits need not be identical. The optimal number of clock phases in a multi-phase

synchronization scheme is fully dependent on the specific design. In general, as the number of clock phases increases, the maximum operating frequency of the circuit increases at the expense of circuit area. It may be possible to formulate an optimization problem to solve for the optimal number of clock phases for any given level-sensitive circuit, by formulating the trade-off between the increase in circuit area and the improvement in the maximum operating frequency.

Finally, the problem definition can be improved by targeting the *statistical* timing analysis of level-sensitive circuits instead of the *static* timing analysis. The timing analysis procedures offered in this thesis defines static timing variables in order to model circuit timing. The difference between a static timing variable and a statistical timing variable is that, a static timing variable identifies the permissible range for a variable, but does not identify the probability of that variable having a particular value in the given permissible range. While the static timing analysis identifies the circuit operation at the operating frequency margins, it fails to provide a probabilistic profile for different states of circuit operation. The statistical timing analysis of a synchronous circuit is performed in order to derive the probabilities of a circuit operating at any feasible frequency (or more generally, at any feasible timing schedule). It may be possible to use the proposed formulation as a template or a well-defined starting point in order to derive a novel problem formulation for the statistical timing analysis of level-sensitive circuits. In very deep sub-micron (VDSM) circuits, the uncertainty in circuit operation including the uncertainty in the precision of circuit timing significantly increases. This fact leads to the growing attention of the digital circuit designers to statistical timing analysis approaches for synchronous circuits. Statistical timing analysis is worth exploring in future research.

APPENDIX A

APPENDIX A

NONLINEAR PROBLEM FORMULATION

This appendix demonstrates the Nonlinear Programming (NLP) model problem formulation of the clock period minimization problem. The circuit network shown in Figure 6.1 is investigated for the clock period minimization problem and the NLP model problem formulation is demonstrated. The circuit network in Figure 6.1 is presented in Figure A-1 for convenience.

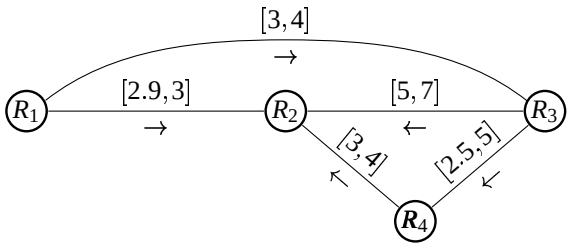


Figure A-1 A simple synchronous circuit.

(Obj) $Min \quad T$

such that

(i) Latching Constraints - Hold Time

$a_1 \geq 0$ $\qquad$ $a_2 \geq 0$

$a_3 \geq 0$ $\qquad$ $a_4 \geq 0$

(ii) Latching Constraints - Setup Time

$A_1 - T \leq 0$ $\qquad$ $A_2 - T \leq 0$

$A_3 - T \leq 0$ $\qquad$ $A_4 - T \leq 0$

(iii) Synchronization Constraints - Earliest Time

$$d_1 = \max(a_1, 0.5T) \qquad d_2 = \max(a_2, 0.5T)$$

$$d_3 = \max(a_3, 0.5T) \qquad d_4 = \max(a_4, 0.5T)$$

(iv) Synchronization Constraints - Latest Time

$$D_1 = \max(A_1, 0.5T) \qquad D_2 = \max(A_2, 0.5T)$$

$$D_3 = \max(A_3, 0.5T) \qquad D_4 = \max(A_4, 0.5T)$$

(v) Propagation Constraints - Earliest Time

$$a_2 = \min\left[(d_1 + 2.9 + t_1 - t_2 - T), (d_3 + 5 + t_3 - t_2 - T), (d_4 + 3 + t_4 - t_2 - T)\right]$$

$$a_3 = d_1 + 3 + t_1 - t_3 - T$$

$$a_4 = d_3 + 2.5 + t_3 - t_4 - T$$

(vi) Propagation Constraints - Latest Time

$$A_2 = \max\left[(D_1 + 3 + t_1 - t_2 - T), (D_3 + 7 + t_3 - t_2 - T), (D_4 + 4 + t_4 - t_2 - T)\right]$$

$$A_3 = D_1 + 4 + t_1 - t_3 - T$$

$$A_4 = D_3 + 5 + t_3 - t_4 - T$$

(vii) Validity Constraints - Arrival Time

$$A_1 - a_1 \geq 0 \qquad A_2 - a_2 \geq 0$$

$$A_3 - a_3 \geq 0 \qquad A_4 - a_4 \geq 0$$

(viii) Validity Constraints - Departure Time

$$D_1 - d_1 \geq 0 \qquad D_2 - d_2 \geq 0$$

$$D_3 - d_3 \geq 0 \qquad D_4 - d_4 \geq 0$$

(ix) Initialization Constraints

$$A_1 = d_1$$

APPENDIX B

APPENDIX B

LP PROBLEM FORMULATION

This appendix demonstrates the Linear Programming (LP) model problem formulation of the clock period minimization problem. The circuit network shown in Figure 6.1 is investigated for the clock period minimization problem and the LP model problem formulation[1] is derived. The circuit network in Figure 6.1 is presented in Figure B-1 for convenience.

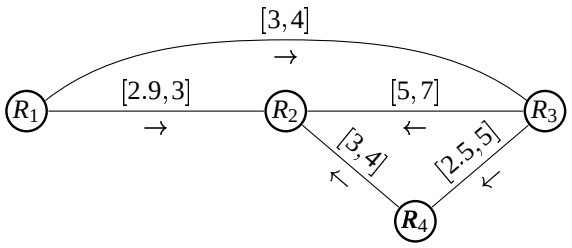


Figure B-1 A simple synchronous circuit.

(Obj) $Min \quad T + 1000d_1 + 1000d_2 + 1000d_3 + 1000d_4 + 1000D_1 + 1000D_2 + 1000D_3 + 1000D_4 + 1000A_2 + 1000A_3 + 1000A_4 - 1000a_2 - 1000a_3 - 1000a_4$

such that

(i) Latching Constraints - Hold Time

$c1 : a_1 \geq 0$ $\qquad$ $c2 : a_2 \geq 0$

$c3 : a_3 \geq 0$ $\qquad$ $c4 : a_4 \geq 0$

[1]The constraints are labeled c1–c43 in order to improve the output readability.

(ii) Latching Constraints - Setup Time

$c5 : A_1 - T \leq 0$ $c6 : A_2 - T \leq 0$

$c7 : A_3 - T \leq 0$ $c8 : A_4 - T \leq 0$

(iii) Synchronization Constraints - Earliest Time

$c9 : d_1 - a_1 \geq 0$ $c10 : d_1 - 0.5T \geq 0$

$c11 : d_2 - a_2 \geq 0$ $c12 : d_2 - 0.5T \geq 0$

$c13 : d_3 - a_3 \geq 0$ $c14 : d_3 - 0.5T \geq 0$

$c15 : d_4 - a_4 \geq 0$ $c16 : d_4 - 0.5T \geq 0$

(iv) Synchronization Constraints - Latest Time

$c17 : D_1 - A_1 \geq 0$ $c18 : D_1 - 0.5T \geq 0$

$c19 : D_2 - A_2 \geq 0$ $c20 : D_2 - 0.5T \geq 0$

$c21 : D_3 - A_3 \geq 0$ $c22 : D_3 - 0.5T \geq 0$

$c23 : D_4 - A_4 \geq 0$ $c24 : D_4 - 0.5T \geq 0$

(v) Propagation Constraints - Earliest Time

$c25 : a_2 - d_1 - t_1 + t_2 + T \leq 2.9$ $c26 : a_3 - d_1 - t_1 + t_3 + T \leq 3$

$c27 : a_2 - d_3 - t_3 + t_2 + T \leq 5$ $c28 : a_4 - d_3 - t_3 + t_4 + T \leq 2.5$

$c29 : a_2 - d_4 - t_4 + t_2 + T \leq 3$

(vi) Propagation Constraints - Latest Time

$c30 : A_2 - D_1 - t_1 + t_2 + T \geq 3$ $c31 : A_3 - D_1 - t_1 + t_3 + T \geq 4$

$c32 : A_2 - D_3 - t_3 + t_2 + T \geq 7$ $c33 : A_4 - D_3 - t_3 + t_4 + T \geq 5$

$c34 : A_2 - D_4 - t_4 + t_2 + T \geq 4$

(vii) Validity Constraints - Arrival Time

$c35 : A_1 - a_1 \geq 0$ $c36 : A_2 - a_2 \geq 0$

$c37 : A_3 - a_3 \geq 0$ $c38 : A_4 - a_4 \geq 0$

(viii) Validity Constraints - Departure Time

$c39 : D_1 - d_1 \geq 0$ $c40 : D_2 - d_2 \geq 0$

$c41 : D_3 - d_3 \geq 0$ $c42 : D_4 - d_4 \geq 0$

(ix) Initialization Constraints

$c43 : A_1 - d_1 = 0$

APPENDIX C

APPENDIX C

LP PROBLEM SOLUTION - CPLEX OUTPUT

This appendix includes the solution of the LP model problem describing the clock period minimization problem of the circuit network shown in Figure B-1. The LP model problem shown in Appendix B is solved using the industrial solver CPLEX [10] and the results are shown below. In the results, `SECTION 1 - ROWS` section presents the optimal solution for each constraint and `SECTION 2 - COLUMNS` section presents the optimal results for each variable.

Note that the optimal objective function value is not completely relevant to the clock period minimization problem. Obtaining the minimum value for the clock signal period is the main objective of the clock period minimiation problem and the minimum clock period is presented in `SECTION 2` ($T = 4.05$). Likewise, the optimal values for the data signal arrival and departure times are presented in `SECTION 2`, constituting the optimal clocking and timing schedules for the synchronous circuit under investigation. For detailed information about CPLEX operation and output formatting, see [10].

```
PROBLEM NAME     fig7.lp
 DATA    NAME
 OBJECTIVE VALUE  26254.05
 STATUS           OPTIMAL SOLN
 ITERATION        27

 OBJECTIVE        obj                 (MIN)
 RHS
 RANGES
 BOUNDS

 SECTION 1 - ROWS

  NUMBER .ROW...   AT  .ACTIVITY...  SLACK ACTIVITY  .LOWER LIMIT.  .UPPER LIMIT.  .DUAL ACTIVITY

       1  obj      BS      26254.05       -26254.05           NONE           NONE               1
       2  c43      EQ             0               0              0              0              -0
       3  c1       BS             0               0              0           NONE               0
       4  c5       BS        -2.025           2.025           NONE              0              -0
       5  c9       BS         2.025          -2.025              0           NONE               0
       6  c10      BS             0              -0              0           NONE               0
```

7	c17	BS	0	-0	0	NONE	0
8	c18	LL	0	0	0	NONE	-2000
9	c35	BS	2.025	-2.025	0	NONE	0
10	c39	LL	0	0	0	NONE	-2500.5
11	c26	UL	3	0	NONE	3	1000
12	c31	LL	4	0	4	NONE	-3500.5
13	c25	UL	2.9	0	NONE	2.9	2500.5
14	c32	BS	6.95	-3.95	3	NONE	0
15	c4	BS	0	0	0	NONE	0
16	c8	BS	-1.55	1.55	NONE	0	-0
17	c15	BS	2.025	-2.025	0	NONE	0
18	c16	LL	0	0	0	NONE	-1000
19	c23	LL	0	0	0	NONE	-1000
20	c24	BS	0.475	-0.475	0	NONE	0
21	c38	BS	2.5	-2.5	0	NONE	0
22	c23	BS	0.475	-0.475	0	NONE	0
23	c29	BS	2.475	0.525	NONE	3	-0
24	c34	BS	6.05	-2.05	4	NONE	0
25	c2	BS	0	0	0	NONE	0
26	c6	UL	0	0	NONE	0	500.5
27	c11	BS	2.025	-2.025	0	NONE	0
28	c12	LL	0	0	0	NONE	-1000
29	c19	LL	0	0	0	NONE	-1000
30	c20	BS	2.025	-2.025	0	NONE	0
31	c36	BS	4.05	-4.05	0	NONE	0
32	c40	BS	2.025	-2.025	0	NONE	0
33	c3	BS	1.025	-1.025	0	NONE	0
34	c7	BS	-2.025	2.025	NONE	0	-0
35	c13	BS	1	-1	0	NONE	0
36	c14	BS	0	-0	0	NONE	0
37	c21	LL	0	0	0	NONE	-2500.5
38	c22	LL	0	0	0	NONE	-2000
39	c37	BS	1	-1	0	NONE	0
40	c41	LL	0	0	0	NONE	-1000
41	c27	BS	2.95	2.05	NONE	5	-0
42	c28	UL	2.5	0	NONE	2.5	2000
43	c32	LL	7	0	7	NONE	-2500.5
44	c33	LL	5	0	5	NONE	-2000

SECTION 2 - COLUMNS

NUMBER	.COLUMN.	AT	.ACTIVITY...	..INPUT COST..	.LOWER LIMIT.	.UPPER LIMIT.	.REDUCED COST.
45	T	BS	4.05	1	0	NONE	0
46	D1	BS	2.025	1000	0	NONE	0
47	BD1	BS	2.025	1000	0	NONE	0
48	A4	LL	0	-1000	0	NONE	1000
49	BA4	BS	2.5	1000	0	NONE	0
50	D4	BS	2.025	1000	0	NONE	0
51	BD4	BS	2.5	1000	0	NONE	0
52	A2	LL	0	-1000	0	NONE	1500.5
53	BA2	BS	4.05	1000	0	NONE	0
54	D2	BS	2.025	1000	0	NONE	0
55	BD2	BS	4.05	1000	0	NONE	0
56	A3	BS	1.025	-1000	0	NONE	0
57	BA3	BS	2.025	1000	0	NONE	0
58	D3	BS	2.025	1000	0	NONE	0
59	BD3	BS	2.025	1000	0	NONE	0
60	BA1	BS	2.025	0	0	NONE	0
61	A1	LL	0	0	0	NONE	0
62	T1	BS	0.05	0	0	NONE	0
63	T3	LL	0	0	0	NONE	0
64	T2	BS	0.925	0	0	NONE	0
65	T4	BS	0.475	0	0	NONE	0

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] T. M. Burks, K. A. Sakallah, and T. N. Mudge. Critical paths in circuits with level-sensitive latches. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 3(2):273–291, June 1995.

[2] M. R. Dagenais and N. C. Rumin. On the calculation of optimal clocking parameters in synchronous circuits with level-sensitive latches. *IEEE Transactions on Computer-Aided Design*, CAD-8(3):268–278, March 1989.

[3] S.-C. Fang and S. Puthenpura. *Linear Optimization and Extensions: Theory and Algorithms.* AT&T. Prentice Hall, 1993.

[4] J. P. Fishburn. Clock skew optimization. *IEEE Transactions on Computers*, C–39(7):945–951, July 1990.

[5] W. Ford and W. Topp. *Data Structures with C++.* Prentice Hall, 1996.

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

[7] P. Gronowski and W. Bowhill. Dynamic logic and latches ii. *IEEE VLSI Circuits Workshop*, 1996.

[8] D. Harris and M. Horowitz. Skew-tolerant domino circuits. *IEEE Journal of Solid-State Circuits*, 32(11):1702–1711, November 1997.

[9] http://public.itrs.net/. International technology roadmap for semiconductors (itrs 2002). Technical report, 2002.

[10] ILOG. *CPLEX 7.1 User's Manual*, 2001.

[11] I. S. Kourtev and E. G. Friedman. A quadratic programming approach to clock skew scheduling for reduced sensitivity to process parameter variations. In *Proceedings of the 1999 IEEE ASIC/SOC Conference*, 1999.

[12] I. S. Kourtev and E. G. Friedman. *Timing Optimization Through Clock Skew Scheduling.* Kluwer Academic Publishers, 2000.

[13] J. Lee, D. T. Tang, and C. K. Wong. A timing analysis algorithm for circuits with level-sensitive latches. *IEEE Transactions on Computer-Aided Design*, CAD-15(5):535–543, May 1996.

[14] I. Lin, J. A. Ludwig, and K. Eng. Analyzing cycle stealing on synchronous circuits with level-sensitive latches. *Proceedings of the 29th ACM/IEEE Design Automation Conference*, pages 393–398, June 1992.

[15] B. Lockyear and C. Ebeling. Optimal retiming of level-clocked circuits using symmetric clock schedules. *IEEE Transactions on Computer-Aided Design*, CAD-13(9):1097–1109, Sep 1994.

[16] D. A. Pucknell and K. Eshraghian. *Basic VLSI Design.* Prentice Hall, 1994.

[17] J. M. Rabaey. *Digital Integrated Circuits.* Prentice Hall, 1996.

[18] K. A. Sakallah, T. N. Mudge, and O. A. Olukotun. $checkT_c$ and $minT_c$: Timing verification and optimal clocking of synchronous digital circuits. *Proceedings of the IEEE/ACM International Conference on Computer–Aided Design*, pages 552–555, November 1990.

[19] K. A. Sakallah, T. N. Mudge, and O. A. Olukotun. Analysis and design of latch-controlled synchronous digital circuits. *IEEE Transactions on Computer-Aided Design*, CAD-11(3):322–333, March 1992.

[20] A. S. Sedra and K. C. Smith. *Microelectronic Circuits.* Oxford University Press, 1998.

[21] N. Shenoy, R. K. Brayton, and A. L. Sangiovanni-Vincentelli. Graph algorithms for clock schedule optimization. *Proceedings of the IEEE/ACM International Conference on Computer–Aided Design*, pages 132–136, November 1992.

[22] B. Stroustrup. *The C++ Programming Language.* Addison Wesley, 2000.

[23] Synopsys Inc. *Synopsys Online Documentation*, 2002.

[24] T. G. Syzmanski and N. Shenoy. Verifying clock schedules. In *Proceedings of the IEEE/ACM International Conference on Computer–Aided Design*, 1992.

[25] T. G. Szymanski. Computing optimal clock schedules. *Proceedings of the 29th ACM/IEEE Design Automation Conference*, pages 399–404, June 1992.

[26] B. Taskin and I. S. Kourtev. Linear timing analysis of soc synchronous circuits with level-sensitive latches. In *Proceedings of the Fifteenth Annual IEEE ASIC/SOC Conference*, pages 358–362, 2002.

[27] B. Taskin and I. S. Kourtev. Performance optimization of single-phase level-sensitive circuits using time borrowing and clock skew scheduling. In *ACM/IEEE International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, pages 111–118, 2002.

[28] B. Taskin and I. S. Kourtev. Linearization of the timing analysis and optimization of level-sensitive circuits. In *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, in submission.

[29] J. Uyemura. *Introduction to VLSI Circuits and Systems.* Wiley, 2002.

[30] W. L. Winston. *Operations Research Application and Algorithms.* PWS-Kent Publishing Company, second edition, 1991.

[31] H. Zhou. Clock schedule verification crosstalk. In *ACM/IEEE International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, pages 78–83, 2002.