Optimizing Two-Phase, Level-Clocked Circuitry

(Extended Abstract)

Alexander T. Ishii
NEC C&C Research Laboratories
Princeton, New Jersey 08540

Charles E. Leiserson
Marios C. Papaefthymiou
MIT Laboratory for Computer Science
Cambridge, Massachusetts 02139

Abstract

We investigate two strategies for reducing the clock period of a two-phase, level-clocked circuit: *clock tuning*, which adjusts the waveforms that clock the circuit, and *retiming*, which relocates circuit latches. These methods can be used to convert a circuit with edge-triggered latches into a faster level-clocked one.

We model a two-phase circuit as a graph whose vertex set V is a collection of combinational logic blocks, and whose edge set E is a set of interconnections. Each interconnection passes through 0 or more latches, where each latch is clocked by one of two periodic, nonoverlapping waveforms, or *phases*.

We give efficient polynomial-time algorithms for problems involving the timing verification and optimization of two-phase circuitry. Included are algorithms for

- verifying proper timing: $O(VE)$ time.
- minimizing the clock period using clock tuning: $O(VE)$ time.
- retiming to achieve a given clock period when the phases are symmetric: $O(VE + V^2 \lg V)$ time.
- retiming to achieve a given clock period when either the duty cycle (high time) of one phase or the ratio of the phases' duty cycles is fixed: $O(V^3)$ time.

By characterizing the set of possible clock periods under any retiming of the circuit, we are able to obtain polynomial-time algorithms for clock period minimization by

- retiming when the phases are symmetric: $O(V^2E)$ time.
- retiming when either the duty cycle of one phase is fixed or the ratio of the phases' duty cycles is fixed: $O(V^2E + V^3 \lg V)$ time.
- simultaneous retiming and clock tuning: $O(V^{11})$ time.

Obviously, this last algorithm is not practical. For this problem, however, we give a fully polynomial-time approximation scheme that achieves the optimal period to within any given relative error bound $e > 0$ and runs in $O(V^3(1/e)\lg(1/e) + (VE + V^2 \lg V)\lg(V/e))$ time. We also give fully polynomial-time approximation schemes for the first two of these three period-minimization retiming problems. These two algorithms run in $O((VE + V^2 \lg V)\lg(V/e))$ and $O(V^3 \lg(V/e))$ time, respectively, and can be used to obtain the optimum clock period in the special case where all propagation delays are integers.

We generalize most of the results for two-phase clocking schemes to simple multiphase clocking disciplines, including ones with overlapping phases. Typically, the

algorithms to verify and optimize the timing of k-phase circuits are at most a factor of k slower than the corresponding algorithms for two-phase circuits.

1 Introduction

A VLSI designer often has the choice of whether to use level-clocked latches or the more conventional edge-triggered latches to implement clocked storage elements in his circuit. An edge-triggered latch directly supports the abstraction of a storage element that is synchronized by the ticking of a clock. When the clocking waveform rises (*i.e.*, the clock ticks), an edge-triggered latch instantaneously samples its input and asserts that value on its output.

A level-clocked latch operates somewhat differently. While the clock input to a level-clocked latch is low, the latch output maintains its value from the most recent time that the clock was high. While the clock is high, however, the input flows unimpeded to the output, unsynchronized with either edge of the clock. In order to avoid problems with race conditions, it is common for level-clocked circuits to adopt clocking disciplines which involve multiple clock waveforms, or *phases*.

In a *two-phase clocking scheme* [15], two clocking waveforms, or *phases*, denoted ϕ_0 and ϕ_1, are employed, as is shown in Figure 1. Formally, we denote a two-phase clocking scheme by a 4-tuple $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1 \rangle$ of strictly positive real numbers. In this context, ϕ_0 denotes the *duty cycle* of the first phase, *i.e.*, the length of time during which the phase is high, while ϵ_0 denotes the *gap* of the first phase, *i.e.*, the amount of time between each falling edge of the first phase and the rising edge of the "next" pulse of the second phase, where clock pulses are ordered according to their falling edges. The duty cycle and gap of the second phase are, similarly, denoted by ϕ_1 and ϵ_1, respectively. The ratio $\rho = \phi_1/\phi_0$ is the *duty ratio* of the clocking scheme. The symbol π is overloaded to also denote the sum

$$\pi = \phi_0 + \epsilon_0 + \phi_1 + \epsilon_1, \tag{1}$$

or *period* of the clocking scheme $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1 \rangle$. Observe, that since ϵ_0 and ϵ_1 are required to be strictly greater than zero, phases ϕ_0 and ϕ_1 must be nonoverlapping, *i.e.*, not simultaneously high.

In this paper we investigate methods for verifying the proper timing of circuits that utilize two-phase clocking schemes, and we present algorithms for optimizing their clock periods. The application domain of these methods, however, is not restricted only to the domain of such two-phase, level-clocked circuits. For example, since an edge-triggered latch can be implemented by two back-to-back level-clocked latches, our algorithms also provide an automatic way for speeding-up

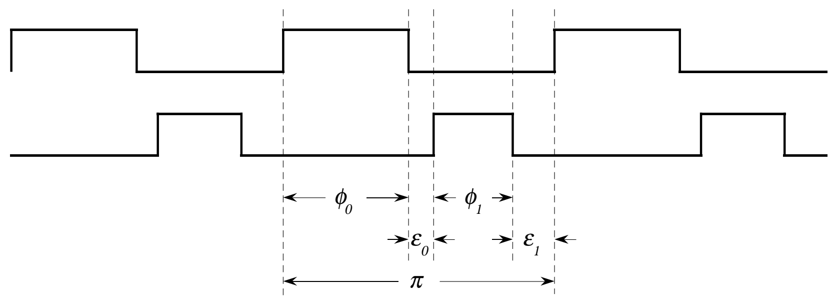

Figure 1: A two-phase, nonoverlapping clocking scheme.

edge-triggered circuits. In addition, most of our algorithms can be generalized to level-clocked circuits utilizing k clock phases.

We model a circuit as a connected graph $G = (V, E)$ in which the vertex set V is a collection of combinational logic elements with associated propagation delays, and the edge set E is a set of interconnections each of which passes through 0 or more latches. Each latch is clocked by either ϕ_0 or ϕ_1. A general framework for the timing verification of level-clocked circuits has been given in [7]. Extensive details of [7], and some details of this work, appear in [6].

An example of a two-phase, level-clocked circuit is given in Figure 2(a). The integers in the vertices signify propagation delays. For simplicity, assume that $\epsilon_0 = \epsilon_1 = 0$. (In practice and in our later analysis for two-phase circuits, we require that ϵ_0 and ϵ_1 be strictly positive; handling the case when either or both are zero simply causes various inequalities to be strict instead of nonstrict.) If the two phases ϕ_0 and ϕ_1 have equal duty cycles, then the circuit cannot be clocked with a clock period shorter than 36 units of time, since the path DEA has propagation delay 54, and intuitively, a datum has at most 3/2 clock periods to propagate from the latch preceding D to the latch succeeding A.

Our first new result is an $O(VE)$-time algorithm that verifies whether a level-clocked circuit is properly timed by a given two-phase clocking scheme. This result improves the $O(E^2)$ bound obtained when the general algorithm from [7] is applied to the special case of two-phase, level-clocked circuits. (The bound in [7] is also $O(VE)$, but the circuit model used in that paper represented both functional elements and latches as vertices, and interconnections between them as edges. Translating their bound to the model presented here yields the $O(E^2)$ bound. The algorithm in [7] applies to more general circuits than two-phase ones, however.)

Our next result deals with modifying, or *tuning*, the clocking scheme of a circuit—that is, providing the circuit with new clocking waveforms. The two clocking schemes in Figure 2(e) and Figure 2(f), for example, illustrate how the clocking scheme of a circuit can be tuned. Observe, that the circuit shown in Figure 2(a) is properly timed by either clocking scheme, but the first clocking scheme has a period of 36, while the second has a clock period of 33. Notice that the notion of clock tuning encompasses more than a simple increase in the frequency of the clock. In particular, clock tuning allows the duty ratio of the clocking scheme to be adjusted as well. Observe also that the circuit cannot be properly timed by any clocking scheme whose period is less than 33, since the delay of the path CDEA is 66 and must be distributed among at most two full clock periods. The clocking scheme shown in Figure 2(f) is thus an *optimal* clock tuning for the circuit in Figure 2(a). We give an $O(VE)$-time algorithm that computes an optimal clock tuning for any given two-phase circuit. Previous algorithms for clock tuning have either addressed alternate types of clocking methodologies [23, 20, 4], or been uncharacterized with respect to worst-case running time [2, 21].

Another way to optimize a circuit is by *retiming*: a method for relocating latches within the circuit without affecting the functionality of the circuit. Retiming has been well studied in the context of edge-triggered circuits [9, 12, 19], and has been the subject of study in the context of single-phase, level-clocked circuits [14, 22]. We extend this technique to encompass the optimization of two-phase, level-clocked circuits. We consider three retiming problems.

The first problem asks whether, for a given two-phase circuit G and clocking scheme π, the circuit G can be retimed to be properly timed by π. As an example, consider the circuit in Figure 2(a). If we retime the circuit to be properly timed by the clocking scheme in Figure 2(g), we obtain the circuit in Figure 2(b). For

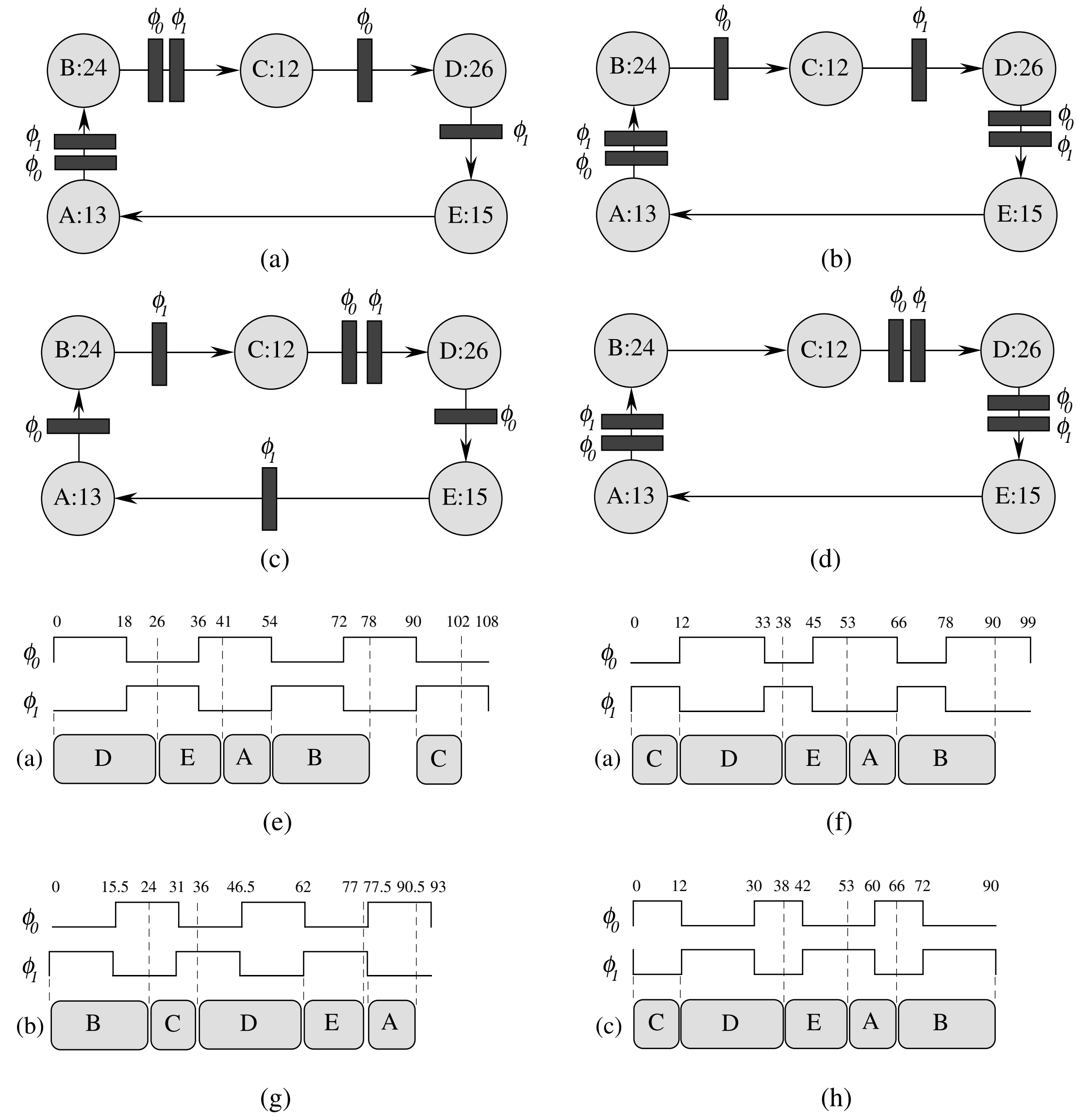

Figure 2: An illustration of the various techniques for optimizing two-phase, level-clocked circuits. Part (a) of the figure shows a level-clocked circuit. When the duty cycles of the phases ϕ_0 and ϕ_1 are equal, as is shown in part (e) of the figure, the clock period of the circuit cannot be made smaller than 36. By tuning the clocks of the circuit to have a duty ratio of 4:7, as shown in part (f), a clock period of 33 can be achieved. Part (b) of the figure shows a retimed version of the original circuit. When clocked with the symmetric clocking scheme shown in part (g), it achieves a clock period of 31, which is optimal for symmetric clocking schemes under any retiming. The optimal combination of retiming and clock tuning for the circuit is shown in part (c). When clocked with the waveforms in part (h), which have a duty ratio of 3:2, the circuit achieves a clock period of 30, which is the optimal for any combination of clocking scheme and retiming. Part (d) shows a typical level-clocked implementation of an edge-triggered circuit which is equivalent under retiming to the other circuits. Each pair of level-clocked latches in the circuit implements an edge-triggered latch. This edge-triggered circuit has a clock period of 36, which is the best that can be obtained by any retiming using edge-triggered latches.

symmetric clocking schemes, such as that in Figure 2(g), in which the two phases are identical and 180 degrees out of phase with each other—and thus, the clocking scheme has the form $\pi = \langle \phi, \epsilon, \phi, \epsilon \rangle$—we provide an algorithm for retiming that runs in $O(VE + V^2 \lg V)$ time. When the clocking scheme π has general form, we provide an algorithm for retiming that runs in $O(V^3)$ time.

The second problem is the *fixed duty-ratio retiming* problem, which asks how a circuit can be retimed to achieve the minimum clock period possible over all clocking schemes with a given duty ratio. For example, with a duty ratio of 1, the optimal retiming of the circuit in Figure 2(a) and tuning of its clock yields the circuit in Figure 2(b) with the clocking scheme in Figure 2(g). We give an algorithm for the fixed duty-ratio retiming problem which runs in $O(V^2E + V^3 \lg V)$ time and produces the optimal retimed circuit. In addition, we give an algorithm that, for any given $e > 0$, runs in $O(V^3 \lg(V/e))$ time and produces a retimed circuit that is properly timed by a clocking scheme whose period is at most $(1 + e)$ times the optimal. If the clocking scheme is required to be symmetric, the algorithms can be improved to run in $O(V^2E)$ time and $O((VE + V^2 \lg V) \lg(V/e))$ time, respectively. All these algorithms can be adapted to optimally retime circuits when one of the two duty cycles of the clock is fixed and the other can be tuned.

The third problem asks how a circuit can be retimed to achieve the minimum clock period possible when no duty ratio is specified. For example, the optimal retiming of the circuit in Figure 2(a) yields the circuit in Figure 2(c) with the clocking scheme in Figure 2(h). We provide an efficient approximation algorithm, but unfortunately, only an impractical polynomial-time algorithm for this *variable duty-ratio retiming* problem. For any given $e > 0$, the approximation algorithm runs in $O(V^3(1/e)\lg(1/e) + (VE + V^2 \lg V)\lg(V/e))$ time and produces a retiming that is properly timed by a clocking scheme whose period is at most $(1 + e)$ times the optimal.

Our optimization techniques cannot only be used to facilitate the design of level-clocked circuits, but also to convert edge-triggered circuits into faster level-clocked circuits. The basis for the conversion is that an edge-triggered latch can be implemented by a pair of level-clocked latches. In Figure 2(d) we illustrate the typical level-clocked implementation of an edge-triggered circuit. This circuit has a clock-period of 36, but the algorithms presented in this paper can automatically produce the optimal level-clocked circuit in Figure 2(c) which, with the clocking scheme in Figure 2(h), is timed properly with a clock period of 30. As a testament to the additional power gained by level-clocking, it should be noted that no edge-triggered retiming of the circuit from Figure 2(d) improves upon its period of 36.

The remainder of this paper is organized as follows. Section 2 describes the necessary and sufficient conditions for a two-phase, level-clocked circuit to be properly timed. Section 3 describes the $O(VE)$ algorithm for verifying the proper timing of a two-phase, level-clocked circuit. Section 4 presents the $O(VE)$ algorithm for tuning clocking schemes. This algorithm adjusts the duty cycles of the two phases, so that the clock period is minimized. In Section 5, we show how to retime a circuit to achieve a specified clocking scheme π. We describe the $O(VE + V^2 \lg V)$-time algorithm for retiming when the duty cycles are symmetric. This algorithm is based on the reduction of the retiming problem to an efficiently solvable mixed-integer linear program. We also describe the $O(V^3)$-time algorithm for retiming to achieve an arbitrary clocking scheme π. Both algorithms also determine when the given clocking scheme cannot be achieved by any retiming. In Section 6 we present algorithms for retiming that achieve the optimal clock period to within a relative error of $e > 0$. Specifically, we give an $O((VE + V^2 \lg V)\lg(V/e))$-time algorithm for the case when duty cycles are symmetric, and an $O(V^3 \lg(V/e))$-time algorithm for the case when either the duty cycle of one of the phases is fixed, or the duty ratio is fixed. When the duty ratio is not fixed, we present an $O(V^3(1/e)\lg(1/e)+(VE+V^2\lg V)\lg(V/e))$-time algorithm that retimes the circuit to achieve the optimal period to within a relative error of e. In Section 7 we describe fully-polynomial algorithms for the problems of optimal retiming. Specifically, we give an $O(V^2E)$-time algorithm for optimal retiming when duty cycles are symmetric, an $O(V^2E+V^3\lg V)$-time algorithm for optimal retiming when either the duty cycle for one of the phases is fixed or the duty ratio of the clocking scheme is fixed, and an $O(V^{11})$-time algorithm for optimal retiming combined with clock tuning. Section 8 extends many of our techniques to circuits that utilize more than two phases. The algorithms for k-phase circuits are generally at most a factor of k slower than the corresponding algorithms for two-phase circuits. Finally, Section 9 presents some concluding remarks. Due to space limitations, most proofs in this extended abstract have been omitted. These will appear in the final version of the paper.

In independent work, Lockyear and Ebeling [13] have also obtained algorithms for retiming multiphase, level-clocked circuits. Their results include an $O(V^3)$-time algorithm to determine whether a given clock period can be achieved by retiming when all clock phases are symmetric. They use this algorithm as a subroutine for determining a retiming that achieves the minimum clock period possible when phases are symmetric. They also determine a set of constraints for retiming with asymmetric clock phases, and they sketch a Bellman-Ford-like algorithm for solving the constraints.

2 Preliminaries

In this section we give necessary and sufficient conditions for a two-phase, level-clocked circuit to be properly timed by a given clocking scheme. The set of conditions, or *timing constraints*, is based on the general formulation from [7], but is substantially simplified by the additional structure inherent in two-phase, level-clocked circuits.

A level-clocked circuit is a set of interconnected functional elements, and level-clocked latches. A functional element (*i.e.*, combinational logic block) is characterized by its maximum propagation delay, while a level-clocked latch is characterized by the phase that controls it. A level-clocked circuit is *two-phase* level-clocked if it

1. utilizes a two-phase clocking scheme,
2. contains no latch-free cycles, and
3. contains no combinational paths between latches controlled by the same phase,

where a path is *combinational* if the first and last components in the path are the only path components that are not functional elements. Observe that every cycle in a two-phase circuit must contain at least two latches—one for each of the two different clock waveforms.

A fundamental property of two-phase, level-clocked circuits is that it is typically possible to associate clock waveforms with each circuit component. In particular, if a circuit has no sources or sinks, *i.e.*, if the output of every component is an input to some component and each input to a component is the output of some component, then for any functional element B, the latches from which B can be reached via combinational paths must all be controlled by the same clock phase ϕ. The clock ϕ is the *input phase*, or simply, the *phase* of the functional element B. One can also associate with B the unique clock waveform ϕ' which controls all latches that can be reached from B via a combinational path. The clock ϕ' is the *output phase* of B.

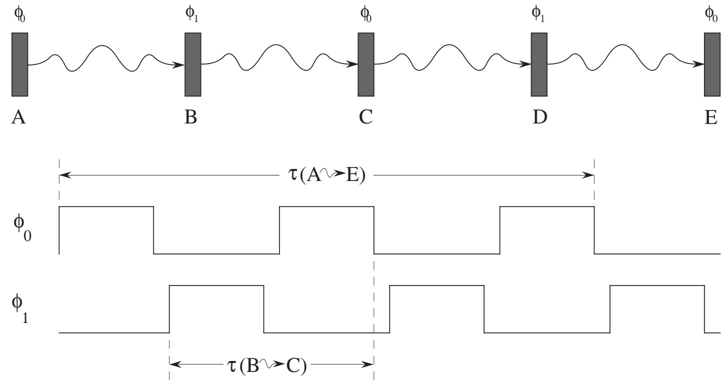

Figure 3: The rise-to-fall time for the path $A \rightsquigarrow E$ is $\phi_0 + 2(\epsilon_0 + \phi_1 + \epsilon_1 + \phi_0) = \phi_0 + 2\pi$. The rise-to-fall time for the path $B \rightsquigarrow C$ is $\phi_1 + \epsilon_1 + \phi_0$. Each of the squiggly lines in the top part of the figure represents a path of zero or greater combinational delay.

It is easy to confirm that the input phase of a functional element must be different from the output phase. For succinctness, we use the function χ to indicate the (input) phase of a component B. If $\chi(B) = 0$, then the phase of B is ϕ_0. Similarly, if $\chi(B) = 1$, then the phase of B is ϕ_1. We generally denote the phase of B as $\phi_{\chi(B)}$. Using this notation, the output phase of B can be denoted as $\phi_{1-\chi(B)}$. If a circuit contains sources or sinks, then the existence of well-defined input phases and output phases must be guaranteed by adding an explicit condition to the definition of a two-phase, level-clocked circuit.

Intuitively, a level-clocked circuit is properly timed if whenever a latch holds a value (*i.e.*, whenever its clock input is low), it holds the same value it would in an identical circuit in which all functional elements have zero propagation delay. Ishii and Leiserson [7] give a set of "Δ-constraints" that serve as a necessary and sufficient set of conditions for the proper timing of a general class of level-clocked circuits. Although this formulation is somewhat complicated by the generality of the circuits which it handles, for two-phase circuits, it leads to a set of necessary and sufficient conditions that can be expressed in a simple manner.

Let p be any path, not necessarily simple, from a latch A to a latch B in the circuit. The *rise-to-fall* time $\tau(p)$ of p is the time it takes in a zero-delay circuit for the effect of a rising edge of A's phase to propagate along p and be stored in B by the falling edge of B's phase. For example, the rise-to-fall time of path $A \rightsquigarrow E$ in Figure 3 is $\phi_0 + 2(\epsilon_0 + \phi_1 + \epsilon_1 + \phi_0) = \phi_0 + 2\pi$. The rise-to-fall time of path $B \rightsquigarrow C$ is $\phi_1 + \epsilon_1 + \phi_0$. The following proposition states the conditions for the proper timing of a circuit. Its proof will be given in the full version of the paper.

Proposition 1 *A two-phase circuit is properly timed if and only if for all latches A and B in the circuit, the propagation delay $d(p)$ along any path p from A to B is no greater than the rise-to-fall time $\tau(p)$ of the path.*

It should be noted that, as in [7, 6], the notion of proper timing assumed in this paper is *structural*, in the sense that we require that a circuit be properly timed regardless of the functions computed by the functional elements. This requirement avoids potential difficulties with computational intractability.

A two-phase, level-clocked circuit is formally represented by a connected, directed multigraph $G = \langle V, E, d, w, \chi \rangle$, where the *delay* function d is a mapping from V to the nonnegative real numbers, the *weighting* function w is a mapping from E to the nonnegative integers, and the *phase* function χ is a mapping from V to the set $\{0, 1\}$. For convenience, we use the terms "graph" and "circuit" interchangeably. A vertex $v \in V$ corresponds to a functional element whose maximum propagation delay is equal to $d(v)$. Level-clocked latches are not represented as vertices, but rather, are represent implicitly by the weights $w(e)$ of edges $u \xrightarrow{e} v \in E$. A value $w(e) = 0$ indicates that a direct connection exists between the output of the functional element represented by u and an input of the functional element represented by v. A value $w(e) > 0$ indicates that the connection between the two functional elements consists of $w(e)$ level-clocked latches connected in series. One can verify that a given graph $G = \langle V, E, d, w, \chi\rangle$ corresponds to a two-phase, level-clocked circuit if and only if, for all paths $u \overset{p}{\leadsto} v$ in G, the value $w(p) - \chi(u) + \chi(v)$ is an even number.

Given Proposition 1, we can formulate a set of conditions for the proper timing of a two-phase, level-clocked circuit. Specifically, we have the following lemma.

Lemma 2 *Let $G = \langle V, E, d, w, \chi\rangle$ be a circuit that utilizes a clocking scheme $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1\rangle$. Then G is properly timed if and only if, for every path $u \overset{p}{\leadsto} v$ in G, we have*

$$d(p) \leq \pi\left(\frac{1 + w(p)}{2}\right) + \phi_{1-\chi(v)} \tag{2}$$

if $\chi(u) \neq \chi(v)$, and

$$d(p) \leq \pi\left(\frac{2 + w(p)}{2}\right) - \epsilon_{1-\chi(v)} \tag{3}$$

if $\chi(u) = \chi(v)$. □

The reader should note that the lemma requires *all* paths in the circuit to be considered, not just simple ones.

3 Verifying proper circuit timing

In this section we describe an $O(VE)$-time algorithm to verify whether a level-clocked circuit is properly timed by a given two-phase clocking scheme. This bound is better than the $O(E^2)$ bound that one obtains by applying the general verification algorithm in [7] to two-phase circuits.

The verification algorithm VERIFY is shown in Figure 4. The algorithm determines whether a circuit is properly timed by checking certain constraints which can be shown to be equivalent to the path constraints in Lemma 2.

In order to prove the correctness of Algorithm VERIFY, we first show that Step 1 computes a new circuit G' with at most 3 latches per wire and such that G is properly timed if and only if G' is properly timed. The reason for performing this transformation is that the running time of Steps 2 and 3 depends on the number of latches on each wire of the circuit. Thus, reducing this number to at most 3 results in a more efficient verification algorithm. Note that the new circuit G' does not in general compute the same function as G.

Lemma 3 *Let $G = \langle V, E, d, w, \chi\rangle$ be a circuit, and let $G' = \langle V, E, d, w', \chi\rangle$ be the circuit obtained by replacing the $w(e)$ latches on each edge $e \in E$ with $w'(e)$ latches, where*

$$w'(e) = \begin{cases} w(e) & \text{if } w(e) \leq 3 \ , \\ (w(e) \bmod 2) + 2 & \text{if } w(e) \geq 4 \ . \end{cases}$$

VERIFY(G, π) $\quad \triangleright G = (V, E, d, w, \chi)$; $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1 \rangle$

1. Generate $G' = (V, E, w', d, \chi)$ by replacing $w(e)$ on each edge $e \in E$ with $w'(e)$, where
$$w'(e) = \begin{cases} w(e) & \text{if } w(e) \leq 3 \;, \\ (w(e) \bmod 2) + 2 & \text{if } w(e) \geq 4 \;. \end{cases}$$
2. Compute $D(v, i)$ for all $v \in V$ and $i = 0, 1, 2, \ldots, 3|V|$, from the recurrence
$$D(v, i) = d(v) + \max\left\{D(u, i - w'(e)) : u \xrightarrow{e} v \text{ and } i \geq w'(e)\right\}.$$
3. Check the following constraints for each vertex $v \in V$ and $i = 0, 1, \ldots, 3\,|V|$:
$$D(v, i) \leq \pi\left(\frac{1+i}{2}\right) + \phi_{1-\chi(v)} \quad \text{if } i \text{ is odd;}$$
$$D(v, i) \leq \pi\left(\frac{2+i}{2}\right) - \epsilon_{1-\chi(v)} \quad \text{if } i \text{ is even.}$$
If any constraint is violated, then return **no**.
4. Determine whether the graph G with weight $w(e) - 2d(u)/\pi$ on each edge $u \xrightarrow{e} v$ has a negative-weight cycle. If a negative-weight cycle exists, then return **no**; otherwise, return **yes**.

Figure 4: Pseudocode for the $O(VE)$ verification algorithm.

Then for any clocking scheme $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1 \rangle$, G is properly timed by π if and only if G' is properly timed by π. □

The following lemma, whose proof relies on Lemma 2, justifies Steps 2, 3, and 4 of the algorithm. The lemma shows that we can substitute $O(V^2)$ inequalities for the infinitely many path-constraints (2) and (3) that describe the conditions for proper circuit timing. This simpler set of inequalities motivates the computation of the various $D(v, i)$ values.

Lemma 4 *Let $G = \langle V, E, d, w, \chi \rangle$ be a circuit that uses a clocking scheme $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1 \rangle$. Then G is properly timed if and only if for every vertex $v \in V$ and $i = 0, 1, \ldots, 3\,|V|$, we have*
$$D(v, i) \leq \pi\left(\frac{1+i}{2}\right) + \phi_{1-\chi(v)} \tag{4}$$
if i is odd, and
$$D(v, i) \leq \pi\left(\frac{2+i}{2}\right) - \epsilon_{1-\chi(v)} \tag{5}$$
if i is even; and for every simple cycle p, we have
$$d(p) \leq \pi\left(\frac{w(p)}{2}\right) \;. \tag{6}$$
□

Inequalities (4) and (5) directly justify Steps 2 and 3 of the algorithm. The values $D(v, i)$, which are computed in Step 2 and used in Inequalities (4) and (5), are defined for $v \in V$ and $i = 1, 2, \ldots, 3|V|$ by

$$D(v, i) = \max\{d(p) : u \overset{p}{\rightsquigarrow} v, \text{ and } w(p) = i\} ,$$

where, by convention, $D(v, i) = 0$ if there are no paths with exactly i latches to v from any vertex. Intuitively, for a given vertex v and integer i, a value $D(v, i)$ embodies the largest left-hand side of any constraint described by Inequalities (2) and (3). This formulation has the advantage of reducing the number of constraints to be checked, since every path between two vertices need not be considered.

Inequality (6) justifies Step 4 as follows. Sum the edge weights around a cycle. If the cycle has a negative weight, then Inequality (6) is violated. Conversely, if there is no negative-weight cycle, then every cycle in the graph satisfies the inequality. It follows that Algorithm VERIFY correctly determines whether a circuit is properly timed by a given two-phase clocking scheme.

Algorithm VERIFY terminates in $O(VE)$ steps. The circuit transformation of Step 1 can be computed in $O(E)$ time. In Step 2, the algorithm performs $3|V| + 1$ passes over the edge set E. In each pass, it first considers the edges with $w'(e) > 0$, and then the edges with $w'(e) = 0$ in topological sort order. Since a topological sort can be computed in $O(E)$ time [1], Step 2 terminates in $O(VE)$ time. Step 3 checks $O(V^2)$ constraints in $O(V^2)$ time. Finally, Step 4 can be performed in $O(VE)$ time using the Bellman-Ford algorithm [1]. Thus, the total is $O(E) + O(VE) + O(V^2) + O(VE) = O(VE)$.

4 Period minimization by clock tuning

In this section we describe an $O(VE)$-time algorithm which, given a circuit G and a clocking scheme $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1 \rangle$, computes a clocking scheme $\pi^* = \langle \phi_0^*, \epsilon_0, \phi_1^*, \epsilon_1 \rangle$ that properly times G and whose period π^* cannot be further reduced without reducing either ϵ_0 or ϵ_1.

The basic idea behind the period minimization algorithm is to view π, ϕ_0, and ϕ_1 as variables. By Equation (1), $\phi_1 = \pi - \epsilon_0 - \epsilon_1 - \phi_0$, and thus, the constraints defined by (4), (5), and (6) can be viewed as inequalities in the two variables π and ϕ_0. In fact, they form a linear program in which the objective is to minimize π. This program can be described by a set of lines in the (ϕ_0, π) plane. We distinguish three kinds of lines corresponding to Inequalities (4), (5), and (6). Let us look more closely at each of these three sets of constraints that form the linear program.

The constraints defined by Inequality (4) depend on both π and ϕ_0. We can separate these constraints into two sets depending on the value of $\chi(v)$. If $\chi(v) = 1$, then the inequality becomes

$$\pi \geq \frac{D(v, i) - \phi_0}{(i + 1)/2} , \tag{7}$$

which defines a half-plane of feasible (ϕ_0, π) points for each fixed i. If $\chi(v) = 0$, the inequality becomes

$$\pi \geq \frac{D(v, i) + \epsilon_0 + \epsilon_1 + \phi_0}{(3 + i)/2} , \tag{8}$$

which once again defines a half-plane of feasible (ϕ_0, π) points for each fixed i, since ϵ_0 and ϵ_1 are constants. Each inequality holds for each $v \in V$, and for each odd i in $1 \leq i \leq 3|V|$. The $O(V^2)$ constraints defined by Inequality (4) can be determined in $O(VE)$ time by computing the values $D(v,i)$, as in Step 2 of Algorithm VERIFY. These constraints can be reduced in $O(V^2)$ time to an equivalent set of $O(V)$ constraints by selecting, for each odd i in the range $1 \le i \le 3|V|$, the particular constraint for which $D(v,i)$ is maximized.

Constraints defined by Inequality (5) depend only on π, and not on ϕ_0. Consequently, these constraints determine a lower bound on π^* which is independent of the duty cycle of either phase. A single lower bound on π^* can thus be determined from these $O(V^2)$ inequalities in $O(V^2)$ time, after having computed the values of $D(v,i)$ in $O(VE)$ time.

Constraints defined by Inequality (6) depend only on π, and not on ϕ_0, and they also determine a lower bound on π^* which is independent of the duty cycles. This lower bound can be found in $O(VE)$ time by solving a "tramp steamer" problem. The *tramp steamer problem* (also known as the *minimum cost-to-time ratio cycle problem*) was formulated in [3] as follows. Let $G = \langle V, E, c, t\rangle$ be a directed graph in which each edge (u,v) has an integer cost $c(u,v)$ and an integer transit time $t(u,v)$, such that for any cycle p in G, we have $\sum_{(u,v)\in p} t(u,v) > 0$. For any cycle p in G, define the cost-to-time ratio of the cycle by

$$R(p) = \frac{\sum_{(u,v)\in p} c(u,v)}{\sum_{(u,v)\in p} t(u,v)} .$$

The problem is to find the minimum value $R(G)$ of the cost-to-time ratio over all cycles in the graph.

Lemma 5 *Let $G = (V, E, d, w, \chi)$ be a circuit, and let $\tilde{G} = \langle V, E, c, t\rangle$ be the graph obtained by assigning to each edge $(u,v) \in E$ a cost $c(u,v) = -d(u)$ and a transit time $t(u,v) = w(u,v)$. Let $R(G)$ be the minimum cost-to-time ratio of any cycle in G. Then Inequality (6) is satisfied if and only if*

$$\pi \ge 2\left|R(\tilde{G})\right| . \tag{9}$$

□

Using the algorithm for the tramp steamer problem given in [5], the cycle constraints can be checked in $O(VE)$ time. In order to obtain this running time, we must guarantee that the transit time of any path with $|V|$ edges is $O(V)$. Indeed, this requirement is met, since from Lemma 3 the number of latches on any edge can be reduced to at most 3.

Besides these three kinds of constraints, we additionally require the following:

$$\pi \ge \epsilon_0 + \epsilon_1 + \phi_0 . \tag{10}$$

This inequality guarantees that $\phi_1 > 0$, and thus that only (ϕ_0, π) combinations representing valid clocking schemes are considered.

By linear programming theory [17], an optimal (ϕ_0^*, π^*) combination can be obtained at a point of intersection of the constraints defined by Inequalities (5), (7), (8), (9), and (10). As we have mentioned, Inequalities (7) and (8) can be reduced to a set of $O(V)$ constraints, and the remaining three inequalities can each be reduced to a single constraint. Megiddo's algorithm [16] can solve such a "two-dimensional" linear program in $O(V)$ time. Alternatively, one can simply compute the $O(V^2)$ intersections between the lines defined by Inequality (7) and the lines defined by Inequalities (8) and (10). If (ϕ_0', π') is the intersection point of maximum π value, and if π'' is the lower bound derived from Inequalities (5) and (9), then an optimal point (ϕ_0^*, π^*) is given by $(\phi_0', \max\{\pi', \pi''\})$. In either case, the overall running time is $O(VE)$.

5 Retiming

In this section, we show how to determine whether a circuit can be retimed so that it is properly timed by a given clocking scheme. We present two algorithms that take a circuit $G = \langle V, E, d, w, \chi \rangle$ and a clocking scheme $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1 \rangle$, and either compute a retiming of G that is properly timed by π, or determine that no such retiming exists. The first algorithm is optimized for the special case of a symmetric clocking scheme, where $\phi_0 = \phi_1$ and $\epsilon_0 = \epsilon_1$. Like other previous retiming results [10, 22] the algorithm casts retiming for a symmetric clocking scheme as a mixed-integer linear program. The overall running time of the algorithm is $O(VE + V^2 \lg V)$. The second algorithm can be applied to arbitrary two-phase clocking schemes and relies on a new technique we call *integer monotonic programming.* The second algorithm runs in $O(V^3)$ time.

Adopting the notation from [10], a circuit $G_r = \langle V, E, d, w_r, \chi_r \rangle$ is a *legal retiming* of another circuit $G = \langle V, E, d, w, \chi \rangle$ if, for some *retiming function* $r : V \rightarrow \mathbf{Z}$, we have

$$w(e) + r(v) - r(u) \geq 0 \tag{11}$$

for every edge $u \xrightarrow{e} v \in E$, and

$$\chi_r(v) = \begin{cases} \chi(v) & \text{if } r(v) \text{ is even }, \\ 1 - \chi(v) & \text{if } r(v) \text{ is odd }, \end{cases}$$

for all $v \in V$. The variable $r(v)$ counts the *lag* of vertex v, and indicates the number of latches that the retiming process moves from the output of v to the input of v. By definition, the number of latches $w_r(e)$ in the retimed circuit G_r is given by

$$w_r(e) = w(e) + r(v) - r(u) \tag{12}$$

for every edge $u \xrightarrow{e} v \in E$. If a given circuit $G = \langle V, E, d, w, \chi \rangle$ is not properly timed by its clocking scheme π, then the *retiming problem* is to either compute an appropriate retiming function r such that Inequalities (2) and (3) hold for every path in the retimed circuit G_r, or determine that no such retiming exists. If such a retiming function does exist, then π is *achievable* by G under retiming, and the retiming function r is said to *achieve* the clocking scheme.

Symmetric clocking schemes

If the clocking scheme utilized by a circuit is symmetric, *i.e.*, of the form $\pi = \langle \phi, \epsilon, \phi, \epsilon \rangle$, then Inequalities (2) and (3) can be simplified as follows. A circuit $G = \langle V, E, d, w, \chi \rangle$ is properly timed if and only if, for every path $u \overset{p}{\rightsquigarrow} v$, we have

$$d(p) \leq \pi \left(\frac{w(p)}{2} \right) + (\pi - \epsilon) \ . \tag{13}$$

By Definition (12), a retiming $G_r = \langle V, E, d, w_r, \chi_r \rangle$ of a given two-phase circuit $G = \langle V, E, d, w, \chi \rangle$ is properly timed if for every path $u \overset{p}{\rightsquigarrow} v$ in G_r, we have

$$\begin{aligned} d(p) &\leq \pi \left(\frac{1}{2} w_r(p) \right) + (\pi - \epsilon) \\ &= \frac{\pi}{2} \left(w(p) + r(v) - r(u) \right) + (\pi - \epsilon) \ . \end{aligned}$$

This last inequality can be rewritten as the following simple linear constraint over the variables $r(u)$ and $r(v)$,

$$\begin{aligned} r(u) - r(v) &\leq w(p) - \frac{2}{\pi} d(p) + \left(2 - \frac{2\epsilon}{\pi}\right) \\ &= \sum_{i \xrightarrow{e} j \in p} \left(w(e) - \frac{2}{\pi} d(j)\right) + \left(2 - \frac{2}{\pi}(d(u) + \epsilon)\right) \ , \end{aligned} \tag{14}$$

which in turn can be used to prove the following theorem.

Theorem 6 *Let $G = \langle V, E, d, w, \chi \rangle$ be a circuit, and let $\pi = \langle \phi, \epsilon, \phi, \epsilon \rangle$ be a clocking scheme. Then there exists a retiming $r : V \longrightarrow \mathbf{Z}$ of G such that G_r is properly timed by π if and only if there exists an assignment of a real value $R(v)$ and an integer value $r(v)$ to each vertex $v \in V$ such that the following conditions are satisfied:*

$$\begin{array}{rcll} r(u) - r(v) & \leq & w(e) & \text{for all } u \xrightarrow{e} v \in E \\ R(v) - r(v) & \leq & 0 & \text{for all } v \in V \\ R(u) - R(v) & \leq & w(e) - \frac{2}{\pi} d(v) & \text{for all } u \xrightarrow{e} v \in E \\ r(v) - R(v) & \leq & 2 - \frac{2}{\pi}(d(v) + \epsilon) & \text{for all } v \in V \ . \end{array} \tag{15}$$

□

The constraints defined in Theorem 6 form a mixed-integer linear program of the special form MI from [11]. Problem MI can be solved in $O(VE + V^2 \lg V)$ time by applying Algorithm MILP from [11], which gives us an $O(VE + V^2 \lg V)$ algorithm for retiming when the clocking scheme utilized by the circuit is symmetric.

General two-phase clocking schemes

If the clocking scheme utilized by a circuit has a general form, *i.e.*, is not symmetric, the retiming problem can still be cast as a feasibility problem of a special form. There are two cases to consider for a path $u \overset{p}{\rightsquigarrow} v$ in G. First, if $\chi(u) \neq \chi(v)$, we have

$$\begin{aligned} &\left\lfloor \frac{1}{2} r(v) \right\rfloor \pi + (r(v) \bmod 2)\left(\epsilon_{\chi(u)} + \phi_{1-\chi(u)}\right) + \pi \left(\frac{1 + w(p)}{2}\right) - d(p) + \phi_{\chi(u)} \\ &\quad \geq \left\lfloor \frac{1}{2} r(u) \right\rfloor \pi + (r(u) \bmod 2)\left(\phi_{\chi(u)} + \epsilon_{\chi(u)}\right) \ , \end{aligned} \tag{16}$$

where $w(p)$ denotes the number of latches on p. Second, a similar inequality (omitted in this extended abstract) must also hold if $\chi(u) = \chi(v)$. Following the rationale of Section 3, we can rewrite these inequalities as a set of constraints between pairs of vertices. Although there is potentially an exponential number of constraints, they can be reduced to a set of $O(V^2)$ constraints by observing that for each pair of vertices u and v, the constraint corresponding to the shortest path between u and v, with respect to the distance measure $w(e) - 2d(j)/\pi$ for each edge $i \xrightarrow{e} j$, is the strongest among all the paths from u to v. We can compute these constraints by applying Johnson's $O(VE + V^2 \lg V)$-time algorithm for all-pairs shortest-paths [1].

The problem of retiming to achieve a general, two-phase clocking scheme, as described by the set of inequalities (11) and (16), is an instance of a general feasibility problem that we call an *integer monotonic program.* This problem is defined as follows.

Problem IMP *Let S be a set of constraints in which the kth constraint has the form $f_k(x_i) \geq g_k(x_1, x_2, \ldots, x_n)$, where each f_k is monotonically increasing with respect to x_i, and each g_k is monotonically increasing with respect to each x_j, for $j = 1, 2, \ldots, n$. Find a vector $x = (x_1, x_2, \ldots, x_n)$ of nonnegative integers satisfying S, or determine that no feasible vector exists.*

In our retiming problem, the retiming variables $r(v)$ are the unknowns x_i, the left-hand sides of Inequality (16) form the functions f_k, and the right-hand sides of Inequality (16) form the functions g_k.

We can solve an integer monotonic program by setting all the unknowns x_i equal to 0 and then performing a sequence of relaxations over the set of constraints. At each relaxation step, we consider a violated constraint, and we increase the argument of f_k by as much as it is required to satisfy the constraint.

One can prove that for integer monotonic programs arising from Inequality 16, no variable $r(v)$ can attain a value larger than $3|V|$ if the constraint system is satisfiable. Consequently, if the algorithm has not terminated after $O(V^2)$ relaxations, no legal retiming exists. If the clocking scheme is achievable, however, the algorithm can be made to generate a legal retiming. The algorithm can be made to run in $O(V^3)$ time.

6 Approximation schemes for minimum-period retiming

This section presents fully polynomial-time approximation schemes for computing a "minimum-period" retiming of a given two-phase, level-clocked circuit G. A retiming G_r is a *minimum-period* retiming if it achieves a clocking scheme whose period is minimal over some specified set of achievable clocking schemes.

The first algorithm computes, for a given G and a fixed duty ratio $\rho = \phi_1/\phi_0$, a retimed circuit G_r and a clocking scheme $\pi = \langle \phi_0, \epsilon_0, \rho\phi_0, \epsilon_1 \rangle$ such that G_r is properly timed by π, and the period of π is at most $(1 + e)$ times the period of any clocking scheme that can be achieved by G under retiming and whose duty ratio is ρ. The running time of this algorithm is $O(V^3 \lg(V/e))$, but the running time can be reduced to $O((VE + V^2 \lg V) \lg(V/e))$ if only symmetric clocking schemes are utilized. In addition, if all functional elements have integer propagation delays, then the algorithm can be adapted to compute retimings which achieve exactly the minimum period possible.

The second algorithm lifts the fixed duty-ratio requirement and computes in $O(V^3(1/e) \lg(1/e) + (VE + V^2 \lg V) \lg(V/e))$ time a retiming G_r and a clocking scheme $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1 \rangle$ such that G_r is properly timed by π, and the period of π is at most $(1 + e)$ times the period of any clocking scheme that can be achieved by G under retiming.

The full paper will contain a complete description of the two algorithms. We now sketch each.

The basic algorithm for fixed duty-ratio retiming is a binary search over a space of possible clock periods. Since only fixed duty-ratio clocking schemes are considered, each possible clock period corresponds to a unique clocking scheme. We can check in $O(V^3)$ time whether a particular period is achievable using the general retiming algorithm of Section 5. The only difficulties are determining the space of possible clock periods and determining when the search has isolated a sufficiently accurate approximation of the true minimum. Inequality (2) specifies that the period of an achievable clocking scheme must be greater than the maximum delay $d_{\max}$ of any individual functional element. In addition, Inequalities (4), (5) and (6) guarantee that a period of $|V|d_{\max} + \epsilon_0 + \epsilon_1$ is achievable, regardless of the duty-ratio.

Therefore, the minimum period must be in the range from $d_{\max}$ to $|V|d_{\max}+\epsilon_0+\epsilon_1$. Moreover, since $d_{\max}$ is a lower bound on the minimum achievable clock period, a clock period that exceeds a lower bound of the optimal clock period by at most $e\, d_{\max}$ is guaranteed to be less than $(1+e)$ times the period of any fixed duty-ratio clocking scheme that is achievable under retiming. Algorithm correctness follows from the fact that the inability to achieve a clock period of π guarantees the inability to achieve any clock period $\pi' < \pi$.

As in the algorithm for optimal retiming with fixed duty-ratios, the algorithm for the general problem uses the $O(V^3)$-time algorithm from Section 5 to determine whether a particular combination of clock period and duty-ratio can be achieved by retiming.

The algorithm proceeds in two stages. Let π^* denote the actual minimum clock period. The first stage assumes that the retimed circuit must utilize a symmetric clocking scheme, and uses the minimum-period retiming algorithm for the symmetric clocking scheme case to compute an approximate minimum clock period π^+. The second stage then lifts the assumption of a symmetric clocking scheme and isolates a general scheme of approximately minimum clock-period by binary searching over the range $[\pi^+/2, \pi^+]$ of potential clock periods. The fact that $\pi^* \geq \pi^+/2$ follows from the observation that the clock of any circuit with period π can be tuned to become a symmetric clocking scheme with period at most 2π; one simply increases the shorter of the two duty cycles until it is equal to the longer duty cycle. Using $\pi^+ e$ as an increment of resolution, there are only $O(1/e)$ potential minimum clock periods, and thus, the binary search performs $O(\lg(1/e))$ iterations.

Whether a particular potential clock period π is achievable is determined by searching a set of $1/e$ potential values for ϕ_0. The search is necessary, because the retiming algorithm of Section 5 cannot be applied unless a ϕ_0 value is also supplied. The task is thus to determine whether the interval $(0, \pi/2)$ contains a ϕ_0 value such that the clocking scheme $\langle \phi_0, \epsilon_0, \pi - \phi_0 - \epsilon_0 - \epsilon_1, \epsilon_1 \rangle$ is achievable. Since "achievability" is not monotonic in ϕ_0, however, a simple binary search of the interval $(0, \pi/2)$ is not possible, and the algorithm resorts to a linear search. Fortunately, it can be shown that searching the interval at increments of $\pi^* e$ is sufficient to accurately discern the achievability of periods up to a resolution of $\pi^* e$. Thus, since each π is bounded above by π^+, and since π^* is bounded below by $\pi^+/2$, only $O(1/e)$ points in the interval $(0, \pi/2)$ need be checked.

Combining the time bounds of the various stages and searches, an approximate minimum clock period with unconstrained duty ratio can be computed in $O(V^3(1/e)\lg(1/e) + (VE + V^2 \lg V)\lg(V/e))$ time.

7 Polynomial-time algorithms for minimum-period retiming

In the full paper, this section will describe polynomial-time algorithms that yield exact solutions to the minimum-period retiming problem. When the duty ratio of the candidate clocking schemes is fixed to a given ρ, we can compute in $O(V^2E)$ time a set $\Pi(\rho)$ of $O(V^3)$ potential clock periods that is guaranteed to contain the *exact* minimum clock period π^* achievable by any retimed circuit that utilizes a clocking scheme with duty ratio ρ. A binary search over $\Pi(\rho)$ yields π^* in $O(V^2E + V^3 \lg V)$ time, where the binary search uses the $O(V^3)$ retiming algorithm from Section 5 to test particular elements of $\Pi(\rho)$. If only symmetric clocking schemes are considered, then we can use algorithm MILP for retiming, and the overall algorithm can be made to run in $O(V^2E)$ time. When the duty-ratio of the candidate clocking schemes is unrestricted, we can compute a set Π of potential clock-period/duty-cycle pairs which is guaranteed to contain a pair (π, ϕ_0) such that π is the minimum achievable clock period. The method for computing Π is similar to the method for isolating the optimal clock period by tuning. Unfortunately, the best bound we have on the size of Π is $O(V^8)$. Using the retiming algorithm from Section 5 to check each such point, (π, ϕ_0) can be found in $O(V^{11})$ time—polynomial, but impractical.

8 Multiphase clocking

Many of the results for two-phase clocking can be generalized to "simple" multiphase clocking. For example, a straightforward generalization of the algorithm for verifying the proper timing of two-phase circuits runs in $O(kVE)$ time, where k is the number of clock phases employed. Clock tuning can also be performed, but may require the solution of a general linear programming problem. The algorithms for retiming to achieve a given clocking scheme can be generalized to run in $O(VE+V^2 \lg V)$ time when the given clocking scheme is symmetric, and in $O(kV^3)$ time when the given clocking scheme is not symmetric. The fully polynomial-time approximation schemes for minimum-period retiming with fixed duty ratio can also be generalized to run in $O((VE+V^2 \lg V)\lg(kV/e))$ time and $O(kV^3 \lg(kV/e))$ time for symmetric and asymmetric clocking schemes, respectively. The approximation scheme for the simultaneous clock tuning and retiming of two-phase circuits can be generalized, but the resulting polynomial running time is impractically large. We now sketch the formal multiphase clocking framework and the generalizations of the two-phase algorithms for timing verification and optimization.

A $2k$-tuple of real numbers $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1, \ldots, \phi_{k-1}, \epsilon_{k-1} \rangle$ is a *simple k-phase clocking scheme* if it satisfies the following constraints:

- $\pi = \sum_{i=0}^{k-1} (\phi_i + \epsilon_i)$;
- $0 < \phi_i < \pi$ for $i = 0, 1, \ldots, k-1$;
- $\epsilon_i + \phi_{i+1} > 0$ for $i = 0, 1, \ldots, k-1$;
- $\phi_i + \epsilon_i + \phi_{i+1} < \pi$, for $i = 0, 1, \ldots, k-1$;

where we assume here and henceforth that addition in subscripts is performed modulo k. In this formulation, we allow overlapping phases, that is, the ϵ_i values can be negative. The fourth constraint guarantees that the circuit contains no unsynchronized loops; for two-phase circuitry, this constraint reduces to $\epsilon_i > 0$ for $i = 0, 1$. We say a simple multiphase clocking scheme is *symmetric* if $\phi_i = \phi_j$ and $\epsilon_i = \epsilon_j$ for all $0 \leq i, j \leq k-1$.

A level-clocked circuit is *simple k-phase* level-clocked if

1. it utilizes a simple k-phase clocking scheme,
2. it contains no latch-free cycles, and
3. the latches surrounding any purely combinational path are controlled by successive clock phases, *i.e.*, the path "begins" with a latch controlled by ϕ_i and "ends" with a latch controlled by ϕ_{i+1}.

We denote such a *simple multiphase* circuit by $G = \langle V, E, d, w, \chi \rangle$, where χ now maps each vertex to a number in $\{0, 1, \ldots, k-1\}$ corresponding to the input phase of the vertex.

Clock period constraints. For simple k-phase clocking, the timing constraints analogous to Inequalities (2) and (3) are somewhat more complicated than for two-phase clocking. Let $G = \langle V, E, d, w, \chi \rangle$ be a simple k-phase circuit which uses a clocking scheme $\pi = \langle \phi_0, \epsilon_0, \phi_1, \epsilon_1, \ldots, \phi_{k-1}, \epsilon_{k-1} \rangle$. Then G is properly timed if and only if, for every path $u \overset{p}{\rightsquigarrow} v$ in G,

$$d(p) \le \pi \left\lfloor \frac{w(p)+1}{k} \right\rfloor + \psi(\chi(u), \chi(v)) \ , \tag{17}$$

where

$$\psi(i,j) = \phi_i + \epsilon_i + \phi_{i+1} + \cdots + \epsilon_j + \phi_{j+1} \ .$$

Verifying proper timing. To check whether a simple multiphase circuit operates properly, we can use the constraints defined by Inequality (17) to derive a set of cyclic constraints identical to those described by Inequality (6), except with a k in the denominator instead of a 2. These cyclic constraints can be checked in $O(VE)$ time as in Step 4 of Algorithm VERIFY. If the cyclic constraints are met, the path constraints (17) need only be satisfied by simple paths. The $D(v,i)$ values in Algorithm VERIFY must now be computed for $i = 0, 1, \ldots, (2k-1)|V|$, and the entire verification algorithm runs in $O(kVE)$ time.

Clock tuning. Clock tuning for simple multiphase circuits can be done in much the same way as for two-phase circuits, but general linear programming techniques must be used, because the problem no longer succumbs to a simple, special-case algorithm. Each of the ϕ_i and ϵ_i become variables in the linear program. In the two-phase tuning problem, the ϵ_i values were essentially adjusted in advance to be as small as possible. Depending on the design methodology, however, there may be different technology constraints governing what legal values the ϵ_i may take on. If these technology constraints are linear, as one would typically expect, they can be incorporated into the linear programming problem. Of course, certain special cases can be handled without resorting to a general linear program. For example, a circuit with a three-phase, nonoverlapping clock can be tuned in $O(VE)$ time using the three-dimensional linear programming algorithm of Megiddo [16].

Retiming with symmetric clocks. When clock phases are symmetric, simple multiphase circuits can be retimed to achieve a given clock period in $O(VE + V^2 \lg V)$ time. The following constraints are the natural analog of the two-phase constraints (15):

$$\begin{array}{rcll} r(u) - r(v) & \le & w(e) & \text{for all } u \xrightarrow{e} v \in E \\ R(v) - r(v) & \le & 0 & \text{for all } v \in V \\ R(u) - R(v) & \le & w(e) - \frac{k}{\pi} d(v) & \text{for all } u \xrightarrow{e} v \in E \\ r(v) - R(v) & \le & \frac{k}{\pi}(\phi - d(v)) + 1 & \text{for all } v \in V \ . \end{array}$$

As with the two-phase constraints, these constraints can be solved using the algorithm MILP from [11].

Retiming with unequal phases. Even when clock phases are not symmetric, we can retime to achieve a given simple k-phase clocking scheme in $O(kV^3)$ time. For simple multiphase circuits, inequalities analogous to Inequality (16) are easily formulated as simple summations. In particular, if one assumes that $\phi_i + \epsilon_i \ge 0$, for all $0 \le i \le k-1$, then Inequality (17) can be rewritten as

$$\sum_{i=\chi(v)}^{\chi(v)+r(v)} (\epsilon_i + \phi_{i+1}) - \left(\epsilon_{\chi(v)} + \phi_{\chi(v)+1}\right) + \pi \left\lfloor \frac{w(p)+1}{k} \right\rfloor + \psi(\chi(u), \chi(v)) - d(p)$$

$$\geq \sum_{i=\chi(u)-1}^{\chi(u)-1+r(u)} (\phi_i + \epsilon_i) - \left(\phi_{\chi(u)-1} + \epsilon_{\chi(u)-1}\right) \qquad (18)$$

for any retimed circuit. These constraints can be solved using the integer monotonic programming algorithm. By maintaining the two summations dynamically, the claimed running time can be obtained, with the additional k-factor stemming from the fact that some $r(v)$ can now be as high as $O(kV)$.

Minimum-period retiming. Polynomial-time approximation schemes for the minimum-period retiming of simple k-phase level-clocked circuits can be obtained in a straightforward manner. The algorithms for retiming a multiphase circuit to achieve a given clock period can be employed in practical fully-polynomial approximation schemes for fixed duty-ratio minimum-period retiming. Straightforward generalizations also yield an approximation scheme for the simultaneous clock tuning and retiming of multiphase circuits. Unless significant restrictions are placed on the extent to which clock tuning can be performed, however, the running time of the approximation scheme is likely to contain a e^{-k} factor, which is prohibitively large for most relative error bounds e. We have not yet investigated polynomial-time algorithms for the exact minimum-period retiming of simple multiphase circuits.

The results for simple k-phase level-clocked circuits are optimal for circuits whose functional elements have minimum propagation delays of 0. In particular, such circuits can only be properly timed if they contain no unsynchronized loops, *i.e.*, at every point in time, every loop contains a latch whose controlling phase is low. The formulation above guarantees this property by specifying that $\phi_i + \epsilon_i + \phi_{i+1} < \pi$, for $i = 0, 1, \ldots, k-1$. It is possible to show that this specification does not restrict optimality with respect to minimum clock period, *i.e.*, for any circuit that has no unsynchronized loops and achieves a clock period of π, there exists another circuit that meets the specification and also achieves the period π. Similarly, the assumption that $\phi_i + \epsilon_i \geq 0$, for $i = 0, 1, \ldots, k-1$, also does not restrict optimality with respect to minimum clock period. In fact, this assumption can even be lifted at the expense of complicating Inequality (18).

9 Conclusion

Our algorithms for verifying and optimizing level-clocked circuits have their limitations. For example, as is the case with most work on retiming, it is hard to incorporate data-dependent propagation delays in the framework without making most of the interesting questions NP-hard. There are, however, several topics which appear more amenable to efficient algorithmic solutions.

The first of these is the incorporation of functional elements where the propagation delay may differ for different input-output pairs (the "nonuniform propagation delay" model from [12]). Incorporating such elements changes the time complexities of our algorithms, but the essential algorithms remain unchanged.

Another generalization of the circuit model involves the incorporation of minimum propagation delays (sometimes called "contamination" delays). In this model, the output of a functional element does not become invalid until some specified minimum amount of time after an input changes. We believe that many of our algorithms can be generalized to handle such circuits. This is a topic of current research.

Three generalizations of our work which seem more problematic are the handling of gated clocks, precharged circuits, and nonsimple multiphase clocking disciplines. When logic circuits involve the clock signals themselves (so-called gated clocks), it is possible to show, in the general case, that many timing verification and optimization problems are NP-hard. Nevertheless, we suspect that by making conservative assumptions regarding the behavior of such circuits, many of these problems become tractable. In a precharged circuit, combinational paths may exist between latches controlled by the same phase. Optimizations, such as retiming and clock tuning, may have efficient algorithmic solutions, but there are many subtleties that arise in the formulation of the constraints. Nonsimple multiphase circuits also exhibit many subtleties. A path in such circuits may pass through latches in an arbitrary order, rather than the canonical order assumed in a simple multiphase circuit. Though the proper timing of such circuits can be verified using the analysis and algorithms from [7], the timing optimization of such circuits is possibly more complex than even precharged circuits. Whether these problems have efficient solutions is a topic for further research.

Most of the algorithms described in this paper have been programmed using the *C* programming language, although we have not typically used the most asymptotically efficient implementations. We are currently in the process of applying these algorithms to actual circuits in order to evaluate their performances. An interesting direction for further research in this area is to develop algorithms which provide interactive feedback to designers. For example, rather than simply reporting the minimum clock period of a circuit, an algorithm that performs a "sensitivity analysis" could report the extent to which noncritical propagation delays can be increased without affecting the clock period. We believe that the area of algorithms for the interactive analysis of circuit timing is a rich and largely unexplored field.

Acknowledgements

This research was supported in part by the Defense Advanced Research Projects Agency under Grant N00014-91-J-1698.

References

[1] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. McGraw-Hill, MIT Press, 1990.

[2] M. R. Daganais and N. C. Rumin. Automatic determination of optimal clocking parameters in synchronous mos vlsi circuits. In *Advanced Research in VLSI: Proc. of the 5th MIT Conference*, pages 19–33, 1988.

[3] G. B. Dantzig, W. Blattner, and M. R. Rao. Finding a cycle in a graph with minimum cost to times ratio with application to a ship routing problem. *Theory of Graphs*, 1967. P. Rosenstiehl, editor, Dunod, Paris, and Gordon and Breach, New York, pp. 77–84.

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

[5] M. Hartmann and J. Orlin. Finding minimum cost to time ratio cycles with small integral transit times. Technical Report UNC/OR/TR/91-19, University of North Carolina, Chapel Hill, October 1991.

[6] A. T. Ishii. *Timing in Level-Clocked Circuits*. PhD thesis, Massachusetts Institute of Technology, 1991. Available as MIT/LCS/TR-522.

[7] A. T. Ishii and C. E. Leiserson. A timing analysis of level-clocked circuitry. In *Advanced Research in VLSI: Proceedings of the Sixth MIT Conference*, pages 113–130, April 1990.

[8] D. B. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the Association for Computing Machinery*, 24(1):1–13, January 1977.

[9] C. E. Leiserson, F. M. Rose, and J. B. Saxe. Optimizing synchronous circuitry by retiming. *3rd Caltech Conference on VLSI*, 1983. R. Bryant, ed., pp. 87-116.

[10] C. E. Leiserson and J. B. Saxe. Optimizing synchronous systems. *Journal of VLSI and Computer Systems*, 1(1):41–67, 1983.

[11] C. E. Leiserson and J. B. Saxe. A mixed-integer programming problem which is efficiently solvable. *Journal of Algorithms*, 9:114–128, 1988.

[12] C. E. Leiserson and J. B. Saxe. Retiming synchronous circuitry. *Algorithmica*, 6(1), 1991. Also available as MIT/LCS/TM-372.

[13] B. Lockyear and C. Ebeling. Optimal retiming of multi-phase level-clocked circuits. Technical Report TR 91-10-01, Dept. of Comp. Sci and Eng., University of Washington, Seattle, October 1991.

[14] S. Malik, E. Sentovich, R. K. Brayton, and A. Sangiovanni-Vincentelli. Retiming and resynthesis: Optimizing sequential networks with combinational techniques. In *Proceedings of the Hawaii International Conference on System Sciences*, 1990.

[15] C. A. Mead and L. A. Conway. *Introduction to VLSI Systems*. Addison-Wesley, Reading, Massachusetts, 1980.

[16] N. Megiddo. Partitioning with two lines in the plane. *Journal of Algorithms*, 6:430 – 433, 1985.

[17] C. Papadimitriou and K. Steiglitz. *Combinatorial Optimization, Algorithms and Complexity*. Prentice-Hall, Inc., 1982.

[18] M. C. Papaefthymiou. On retiming synchronous circuitry and mixed-integer optimization. Master's thesis, Massachusetts Institute of Technology, September 1990. Available as MIT/LCS/TR-486.

[19] M. C. Papaefthymiou. Understanding retiming through maximum average-weight cycles. *3rd ACM Symposium on Parallel Algorithms and Architectures*, July 1991.

[20] N. Park and A. C. Parker. Theory of clocking for maximum execution overlap of high-speed digital systems. *IEEE Transactions on Computers*, 37(6):678–690, June 1988.

[21] K. A. Sakallah, T. N. Mudge, and O. A. Olukotun. Optimal clocking of synchronous digital circuits. In *Proceedings of the 1990 IEEE International Conference on CAD*, pages 552–555, November 1990.

[22] N. Shenoy, R. K. Brayton, and A. Sangiovanni-Vincentelli. Retiming of circuits with single phase level-sensitive latches. In *ICCD-91*, 1991. To appear.

[23] S. H. Unger and C. J. Tan. Clocking schemes for high speed digital systems. *IEEE Transactions on Computers*, C-35(10):880–895, October 1986.