Integrated Parametric Timing Optimization of Digital Systems[1]

Hong-Yean Hsieh, Wentai Liu, Ralph K. Cavin

Electronics Research Laboratory

Department of Electrical and Computer Engineering

North Carolina State University

Raleigh, NC 27695-7911

Abstract

Many design techniques have been proposed to optimize the performance of a digital system implemented in a given technology. These techniques include retiming, insertion of intentional clock skew, insertion of buffers, transistor sizing, resynthesizing, and wave pipelining. Each of these techniques can be advantageous in particular applications, and they are often applied *individually* to enhance performance. In this paper, a mixed integer linear programming formulation is derived in which the performance of edge-triggered design is optimized by *simultaneously* adjusting clock skew, retiming the circuit, and applying wave pipelining methods. This formulation is applicable to a broad range of digital circuits. We have applied our formulation to the design examples of a RISC microprocessor, a multiplier, and a correlator. In doing so, we were able to reduce the system clock period several fold in each design. A benefit of this approach to timing optimization is that unintentional clock skew tolerance is a by-product of the computation and the resulting designs often exhibit improved margins against it relative to other non-optimal digital circuits.

[1]H. Hsieh and W. Liu were partially supported by a NSF grant under MIP-9212346

1 Introduction

Clock period optimization is crucial in high performance digital systems. Various techniques for clock period reduction have been proposed, including transistor sizing, logic resynthesis, intentional clock skew, cycle stealing with transparent latches, retiming, and wave pipelining.

Among these optimization techniques, wave pipelining [11, 12, 13, 14] is the newest. Implementations of wave pipelining in different technologies have been reported [10, 13]. In conventional pipelining, input data cannot be strobed into the combinational block until the data being processed is clocked out. In contrast, wave pipelining allows input data to be clocked into the combinational block before the current result is clocked out. In other words, it is possible that two or more elements of the input sequence are present simultaneously in a combinational block. This implies that the clock period is not restricted by the longest propagation delay of the combinational block. It is clear that the insertion of intentional clock skew, as proposed by Fishburn [4], is a special type of wave pipelining technique. Most previous works for wave pipelining focused mainly on the design of a single stage combinational block [10, 13]. However, the theory of a wave pipelined system containing feedback loops and reconvergent fanout paths has not been fully explored [11, 12, 13]. Ekroot deals with this problem but with a restriction of equal average multiplexing degree, explained further in section 3, for each combinational block [12]. Other works only consider a single closed loop system [11, 15]. In this paper, we present a more general model for systems including feedback loops and reconvergent fanout paths. In developing the model, we only consider designs with *edge-triggered latches*, hereafter called registers. The treatment of transparent latch-based designs will be discussed in a related work [2].

The propagation delays of different stages of combinational logic can be equalized by relocating registers, which leads to a reduced clock period. Leiserson first reported this retiming technique by relocating registers in zero-skew synchronous edge-triggered designs [5]. Attempts have been made to further reduce the clock period of a system by combining retiming with other optimization techniques. Examples include the combination of retiming with resynthesis [1, 3] and retiming for latch-based designs [7, 8, 9]. In this paper, we present a mixed integer linear programming formulation in which optimization of retiming, intentional clock skew, and wave pipelining can be modeled concurrently.

Optimization of buffer padding is also easily incorporated into our formulation. We have applied our formulation to designs of a RISC microprocessor, a multiplier, and a correlator and reduced the clock period by more than three fold in each case. The formulation also admits the analysis of design robustness. Due to process and environmental variations, an amount of unintentional clock skew could be associated with each register. Robustness, defined as the maximum tolerance of unintentional clock skew, is dependent on the longest and shortest path delays. No matter if there exists a feedback loop in a zero skew design, as the clock period is increased, robustness increases but eventually saturates. For a non-zero skew design with feedback loops, robustness saturates as well, and its upper bound can be derived theoretically. However, robustness for a non-zero skew design is always larger than that of a zero skew design.

This paper is organized as follows. Section 2 defines the terminologies used for retiming and wave pipelining. In Section 3, we discuss in detail the difficulty and complexity incurred by feedback loops and reconvergent fanout paths during optimization of a single phase design via wave pipelining. Section 4 provides a graph model for a pipelined system with fixed locations of registers. When optimizing wave pipelined systems, LP constraints are formulated to preserve these systems' functionality. In Section 5, these constraints are modified to accommodate the retiming process in the optimization formulation. In Section 6, the tolerance to unintentional clock skew is studied for both zero skew and non-zero skew design styles. Experimental data for other circuits is provided in Section 7. Section 8 considers the padding problem. Finally, directions for future research are given.

2 Background

In this section, we first define the proposed timing model and then give a brief literature review for retiming and wave pipelining.

2.1 Timing Model

The longest propagation delay, $t_{max}(v)$, and the shortest propagation delay, $t_{min}(v)$, are defined over every gate or combinational block, which is referred to as a node v in this paper. For a node v, its outputs are generated if all its inputs are stable for at least the minimum amount of time, denoted as $t_{stable}(v)$ [11, 12]. These amounts are assumed to be measured under worst case conditions. Although the timing model assumes the constant longest and shortest propagation delays for all input-output pairs of an individual node, it can be easily generalized to the case in which propagation delays for each input-output pair of a node are not equal.

Figure 1: (a) A System View (b) A Circuit (black bar represents a register.)

2.2 Retiming

Retiming is a technique which can improve circuit performance by register relocation [5]. In a directed graph, a node v represents a combinational block and a directed edge, $e(u,v)$ or $u \xrightarrow{e} v$, represents the connection between u and v. In this case, u is called a *fan-in node* of v, and v is a *fanout node* of u. Edge $e(u,v)$ is a *fan-in edge* of node v and a *fanout edge* of u. The weight, $w(e)$, is the number of registers associated with edge $e(u,v)$. At node v, the retiming process moves an equal number of registers, represented by an integer variable $r(v)$, from every fanout edge to every fan-in edge. After retiming, the number of registers with an edge $u \xrightarrow{e} v$ is $w(e) - r(u) + r(v)$. A valid retiming process must preserve the property of nonnegative edge weight and is represented mathematically by

$$w(e) - r(u) + r(v) \geq 0 \quad \text{for all edges } u \xrightarrow{e} v$$

Clearly along a path, $p = v_0 \xrightarrow{e_0} v_1 \xrightarrow{e_1} \cdots \xrightarrow{e_{n-1}} v_n$, the path weight, $w(p)$, is the sum of the composed edge weights. An important observation is that retiming never changes the difference of two path weights, $w(p)$ and $w(q)$, if they start and end at the same nodes. This also implies that the weight, $w(l)$, of a loop l is invariant in retiming.

2.3 Wave Pipelining

Wave pipelining is a timing methodology used in digital systems to achieve maximal rate operation [4, 11, 12, 13, 14]. Using this technique, new data is applied to the inputs of a combinational block before the previous data is clocked out, thus effectively pipelining the logic and maximizing the utilization of the logic without inserting registers. Experimental designs of an adder [13] and a population counter [10] showed that the clock period can be reduced by a factor of two or three.

Reconvergent fanout paths are defined as simple paths which share both start and end nodes. *Balanced reconvergent fanout paths* exist if there are the same number of registers along each recon-

Original Formulas	Relaxed Formulas
$s(v) = \max_{i=1}^{n} s(u_i) + t_{max}(v)$	$s(v) \geq s(u_i) + t_{max}(v)$ for $1 \leq i \leq n$
$b(v) = \min_{i=1}^{n} b(u_i) + t_{min}(v)$	$b(v) \leq b(u_i) + t_{min}(v)$ for $1 \leq i \leq n$
$g(v) = \max_{i=1}^{n} s(u_i)$	$g(v) \geq s(u_i)$ for $1 \leq i \leq n$
$h(v) = \min_{i=1}^{n} b(u_i)$	$h(v) \leq b(u_i)$ for $1 \leq i \leq n$

Table 1: Original and Relaxed Formulas

vergent fanout path. Otherwise, these paths are called *unbalanced reconvergent fanout paths.* In graph theory, reconvergent fanout paths are called a semi-loop. These two terms, reconvergent fanout paths and semi-loop, are used interchangeably in this paper. A feedback loop can be regarded as a special case of reconvergent fanout paths. If there are no feedback loops or unbalanced reconvergent fanout paths in a pipelined system, it is called a pipelined system *with balanced paths.* Otherwise, it is called a pipelined system *with unbalanced paths.* A pipelined system with unbalanced paths will be discussed in Section 3.

The following description is only well suited to a pipelined system with balanced paths. As shown in Fig. 1(a), a host machine applies data, d_i, through *external input registers* to a pipelined system at time $i \cdot c$, where c is the system cycle time, and extracts data from it through *external output registers.* In the following discussion, unless stated explicitly, these external registers will not be shown in figures. Data d_i arrives at node v via different paths. Calculated from the host, variables s, b, g, and h of node v are defined as follows:

$s(v)$ = latest arrival time at the output of node v = longest path delay to its outputs plus $i \cdot c$

$b(v)$ = earliest arrival time at the output of node v = shortest path delay to its outputs plus $i \cdot c$

$g(v)$ = latest arrival time at the input of node v = longest path delay to its inputs plus $i \cdot c$

$h(v)$ = earliest arrival time at the input of node v = shortest path delay to its inputs plus $i \cdot c$

The values for these variables at a node can be easily calculated from its fan-in nodes. For a circuit as shown in Fig. 1(b) where there exists no register in the fan-in edges of node v, the values for variables $s(v)$, $b(v)$, $g(v)$, and $h(v)$ can be calculated using the formulas as shown in the first column of Table 1. It has been proven that these formulas can be transformed to the relaxed ones as shown in the second column of Table 1 such that the solution set for c is neither expanded nor shrunk [6, 12].

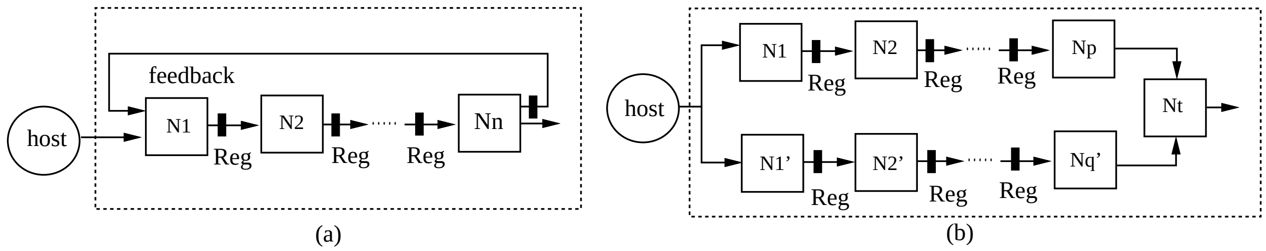

Figure 2: Feedback Loop and Reconvergent Fanout Paths

In order to pass the signal successfully in a wave pipelined design, the following constraint, called the internal node constraint [13], must be satisfied.

$$g(v) + t_{stable}(v) \leq h(v) + c$$

As shown in Fig. 1(b), if data d_i is supposed to be clocked at the output of node v, then the value of $s(v)$ plus the setup time should be smaller than or equal to the clocking time of this register, which is known as the zero clocking constraint [4]. The earliest arrival time at the output of node v is $b(v)$ with respect to d_i. In order to prevent later data from overriding the previous data, the value of $b(v)$ plus the clock period c should be greater than or equal to the sum of the clocking time and the hold time of this register. This is also known as the double clocking constraint [4].

3 Preliminaries

In this section, we focus on single phase pipelined systems with unbalanced paths. Fig. 2(a) shows an example in which there are n registers along the feedback loop. If no skew is allowed at the registers, data d_{i-n} arrives at node 1 at time $i \cdot c$ when it traverses the feedback loop once. Thus both d_{i-n} and d_i are simultaneously available at node 1. Similarly d_{i-n} would be simultaneous with $d_{i+n}, d_{i+2n}, \cdots$. If d_{i-n} interferes with either d_{i+1} or d_{i-1} ($n \neq 1$) after applying wave pipelining, then the system might function incorrectly. In order to preserve the functionality of a wave pipelined system, there should be n "waves" of data in the system. If the system supports $m \cdot n$ waves of data (m is a positive integer), correct operation can be obtained if the system includes an additional $m \times 1$ multiplexer and $1 \times m$ demultiplexer such that primary input/output data are carefully scheduled via multiplexer/demultiplexer. In this paper, m is defined as the *multiplexing degree* of a pipelined system. It is clear that with this new construct, d_i should be paired with $d_{i-m \cdot n}$, instead of d_{i-n} (if $m > 1$).

Fig. 2(b) shows the case of a system with reconvergent fanout paths. Depending on the route, data arrives at node t by traversing either p or q registers (excluding the external one). It is worthwhile pointing out that a feedback loop can be regarded as a special case of reconvergent fanout paths (*i.e.* $p = n$, $q = 0$). In a zero-skew pipelined system, d_i arrives at node t via top route after p cycles meanwhile $d_{i-(q-p)}$ arrives at node t via bottom route or vice versa. In the process of wave pipelining, this property should be taken into account if correct functionality is to be preserved. By the same reasoning used for the feedback loop, in a pipelined system with a multiplexing degree m, d_i should be paired with $d_{i-m(q-p)}$, instead of $d_{i-(q-p)}$.

For a pipelined system with balanced paths, constraints at any node for wave pipelining are easily derived from its fan-in nodes as described in the previous section. However, it becomes more complicated to derive constraints in the case of a pipelined system with unbalanced paths (e.g. for node t). As described previously, there are two alternatives depending on the route taken by d_i. Constraint formulation based on either of these two routes is plausible with the proper manipulation as described in the following paragraph. In other words, if a feasible solution exists for one formulation, then without modifying the amounts of non-zero skew at each register, a feasible solution also exists for the other. Thus constraint formulation can be done by arbitrarily choosing the alternatives.

If d_i reaches node t by traversing the top route, constraints derived along the bottom route for node t are applicable if the values of the variables, $s(q')$, $b(q')$, $p(q')$, and $q(q')$, of node q' are all reduced by the amount of $(q-p)c$. This preserves the constraint in which d_i should be paired with $d_{i-(q-p)}$ at node t when $m = 1$. In a pipelined system with a multiplexing degree m, the amount of reduction is $m(q-p)c$. The amount of $m(q-p)$ is then referred to as the *shift of time origin* for the bottom route. Similarly, if d_i reaches node t by traversing the bottom route, the shift of time origin for the top route should be $m(p-q)$. The definition of the shift of time origin is then defined as follows:

Definition 1 *With reconvergent fanout paths of* $u \xrightarrow{p} v$ *and* $u \xrightarrow{q} v$*, there are* $w(p)$ *registers along path* p *and* $w(q)$ *along path* q *respectively. The shift of time origin for path* p *is* $m(w(p) - w(q))$ *and for path* q *is* $m(w(q) - w(p))$*, where* m *is the multiplexing degree.*

In the graph model of a general pipelined system, at first a spanning tree is found in order to define a pipelined system with balanced paths. Then each of the remaining edges, called chords, in the original graph defines reconvergent fanout paths based on this spanning tree. The shift of time origin associated with each chord can then be calculated accordingly.

It is important to note that in a feedback loop d_i would pair not only with $d_{i-m\cdot n}$ but also with $d_{i-2m\cdot n}, d_{i-3m\cdot n}$, etc. Due to the clock's repetitive nature, it is enough to consider only the constraint due to $d_{i-m\cdot n}$.

4 Clock Optimization via Wave Pipelining

In this section, based on the graph model, we develop constraints required for wave pipelining in a general pipelined system.

4.1 Graph Model for Pipelined Systems

A pipelined system is modeled as a directed graph $G = (V, V_{fo1}, V_{fo2}, V_l, V_{pi}, V_{po}, E, w, t_{max}, t_{min}, t_{stable}, os)$, where

V	set of functional nodes in the pipelined system
V_{fo1}	set of dummy fanout nodes (explained later)
V_{fo2}	set of dummy fanout nodes (explained later)
V_l	set of dummy loop nodes (explained later)
V_{pi}	vertex set as a host which drives primary inputs
V_{po}	vertex set as a host which is driven by primary outputs
E	set of directed edges
w	edge weight (number of registers at each edge)
$t_{max}(v)$	longest propagation delay at each node v of $V \cup V_{fo1} \cup V_{fo2} \cup V_{po}$
$t_{min}(v)$	shortest propagation delay at each node v of $V \cup V_{fo1} \cup V_{fo2} \cup V_{po}$
$t_{stable}(v)$	stable time at each node v of V
$os(v)$	shift of time origin at each node v of V_l if multiplexing degree is 1

A functional node refers to either a gate or a complex module. Vertex sets of V_{pi} and V_{po} depend on the system under study. In a closed system such as Fig. 3(a), they are empty sets. Otherwise, all primary inputs of a system are driven through external input registers by a host node, $v \in V_{pi}$, while all primary outputs are sent through external output registers to a host node, $v \in V_{po}$. It is assumed that all external input registers are clocked without skew among them, and so are external output registers. Therefore, external registers will not be shown in graph model, and for simplicity their propagation delays are assumed to be 0. But the model can easily be extended to the case of

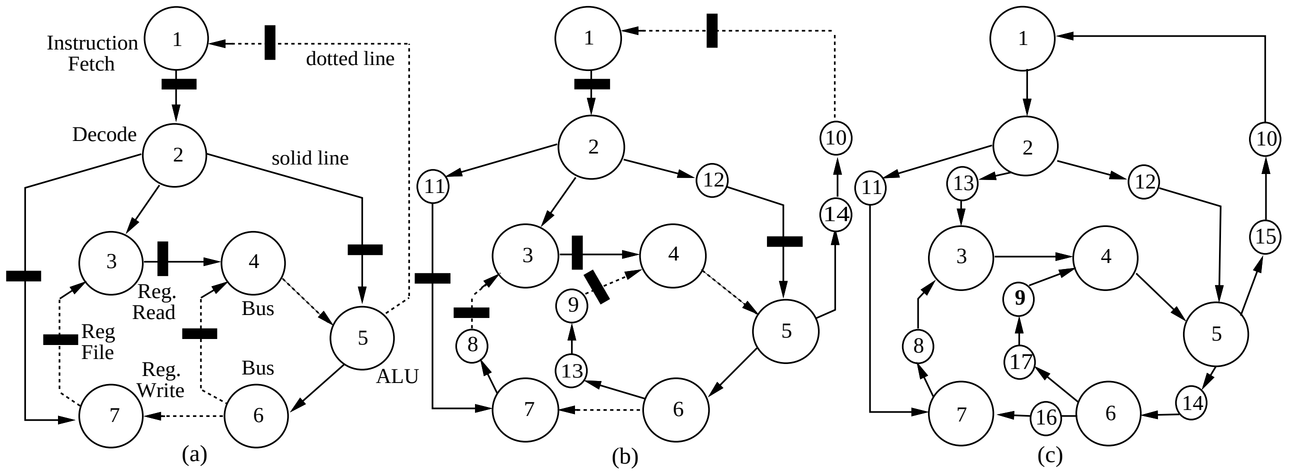

Figure 3: (a) A RISC Design (b) Graph Model (c) Revised Graph Model

different clocking times for each external input/output register. For $v \in V_{po}$, $t_{max}(v)$ and $t_{min}(v)$ are set to 0. Fig. 3(b) illustrates the graph model for the RISC microprocessor shown in Fig. 3(a). Three kinds of dummy nodes are created in the model. A dummy node $v_{f1} \in V_{fo1}$ is inserted in an edge $e(u, v)$ if there is more than one fanout edge for node u and $w(e) \geq 1$. This removes the restriction of applying the same clocking time to the registers at every fanout edge of node u. In this procedure, edge weight $w(e(u, v_{f1}))$ is set to 0 and $w(e(v_{f1}, v))$ is $w(e)$. If more than one register is required to appear at an edge $e(u, v)$, another dummy node $v_{f2} \in V_{fo2}$ can be inserted such that at most one register is on each edge. Edge weight $w(e(u, v_{f2}))$ is set to 1 and $w(e(v_{f2}, v))$ is $w(e) - 1$. Both t_{max} and t_{min} of nodes v_{f1} and v_{f2} are set to 0. As explained in the previous section, a chord, $e(u, v)$, defines a loop or semi-loop in a spanning tree. A dummy loop node v_l is added along the chord if the resulting loop or semi-loop is unbalanced. The *shift of time origin* for node v_l with a multiplexing degree of 1 is $os(v_l)$. In this insertion process, edge weight $w(e(u, v_l))$ is set to zero and $w(e(v_l, v))$ is $w(e)$.

For the example in Fig. 3(b), V is the set of $\{1,2,3,4,5,6,7\}$ and V_{pi}, V_{po}, and V_{fo2} are $\emptyset$. Dummy fanout nodes, 11, 12, 13, and 14 $\in V_{fo1}$, are inserted in the fanout edges $e(2, 7)$, $e(2, 5)$, $e(6, 4)$, and $e(5, 1)$ of nodes 2, 6, and 5. The solid lines and nodes form a spanning tree, and the dotted lines are chords. The loops or semi-loops created by chords $e(7, 3)$, $e(13, 4)$, and $e(14, 1)$ are not balanced, so dummy nodes of 8, 9 and 10 are inserted along these chords respectively. This makes the shift of time origin as follows: 2 for node 8, 1 for node 9, and 3 for node 10.

4.2 Constraints for Wave pipelining

Constraints for valid wave pipelining with a multiplexing degree m are presented in this section. Constraints are classified into the categories of delay constraints, synchronous constraints, loop/semi-loop constraints, wave pipelining constraints, and initiation constraint.

In our model, c is the clock period. Let t_s, t_h, t_{cl}, and t_{cs} denote the setup time, hold time, longest propagation delay, and shortest propagation delay of a register. Variables $s(v)$ and $b(v)$ are the latest and earliest arrival times respectively at the output of node v. Variables $g(v)$ and $h(v)$ are the latest and earliest arrival times respectively at the input of node v. For a node u connected to a register, $s(u) + t_s$ implicitly represents the clocking time of this register.

delay constraints:

$$s(u) + t_{max}(v) \leq s(v) \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo1} \cup V_{fo2} \cup V_{po} \text{ and } w(e) = 0 \qquad (1)$$
$$b(v) \leq b(u) + t_{min}(v) \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo1} \cup V_{fo2} \cup V_{po} \text{ and } w(e) = 0 \qquad (2)$$
$$s(u) \leq g(v) \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = 0 \qquad (3)$$
$$h(v) \leq b(u) \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = 0 \qquad (4)$$

synchronous constraints:

$$s(u) + t_{max}(v) + t_s + t_{cl} \leq s(v) \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo2} \cup V_{po} \text{ and } w(e) = 1 \qquad (5)$$
$$b(v) \leq s(u) + t_{min}(v) + t_s + t_{cs} \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo2} \cup V_{po} \text{ and } w(e) = 1 \qquad (6)$$
$$s(u) + t_s + t_{cl} \leq g(v) \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = 1 \qquad (7)$$
$$h(v) \leq s(u) + t_s + t_{cs} \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = 1 \qquad (8)$$

loop/semi-loop constraints:

$$s(v) = s(u) - m \cdot os(v) \cdot c \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V_l \qquad (9)$$
$$b(v) = b(u) - m \cdot os(v) \cdot c \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V_l \qquad (10)$$

wave pipelining constraints:

$$s(u) + t_s + t_h \leq b(u) + c \qquad \text{for } u \in V_{po} \text{ or for } u \xrightarrow{e} v \text{ and } w(e) = 1 \qquad (11)$$
$$g(v) + t_{stable}(v) \leq h(v) + c \qquad \text{for } v \in V \qquad (12)$$

initiation constraint:

$$b(v) \leq s(v) \qquad v \in V_{pi} \qquad (13)$$

Both delay and synchronous constraints define the timing relationship between node v and its fan-in node u. This depends on whether there exists a register in the fan-in edge $e(u, v)$. Constraint

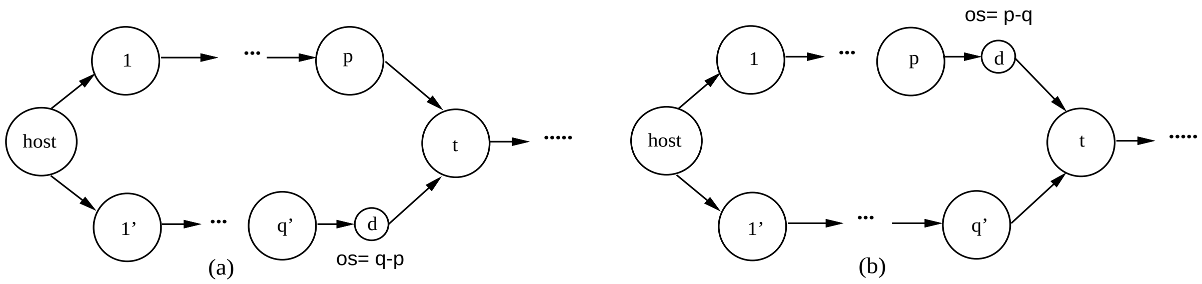

Figure 4: Two Different Spanning Trees

(1) specifies that if $w(e) = 0$, the latest arrival time, $s(v)$, must be greater than or equal to the sum of the latest arrival time, $s(u)$, and the longest propagation delay, $t_{max}(v)$, of node v. If $w(e) = 1$, as shown in constraint (5), the latest arrival time, $s(v)$, should be greater than or equal to the sum of the clocking time, $s(u)+t_s$, $t_{max}(v)$ and the propagation delay, t_{cl}, of the register. Similarly, constraint (2) specifies that if $w(e) = 0$, the earliest arrival time, $b(v)$, must be less than or equal to the sum of the earliest arrival time, $b(u)$, and the shortest propagation delay, $t_{min}(v)$, of node v. If $w(e) = 1$, as shown in constraint (6), the earliest arrival time, $b(v)$, is less than or equal to the sum of the clocking time, $s(u)+t_s$, $t_{min}(v)$ and the propagation delay, t_{cs}, of the register due to the synchronization requirement. Constraints (3), (4), (7), and (8) can be interpreted similarly. Loop/semi-loop constraints incorporate the effects of unbalanced reconvergent fanout paths and feedback loops which have been discussed in the previous section. Three wave pipelining constraints are implicitly or explicitly included in the LP formulations. Since $s(u)$ is the latest arrival time at the output of node u, the zero clocking constraint is satisfied implicitly by the fact that clocking is at time $s(u)+t_s$. Constraints (11) and (12) explicitly express the double clocking constraint and internal node constraint.

4.3 Correctness of the formulation

In this section, the correctness of the LP formulation is proven by examining the pipelined system as shown in Fig. 2(b). Since any reconvergent fanout paths can be regarded as such a structure, the proof does not lose its generality. For such a pipelined system, modeling could result in two different graph representations as shown in Fig. 4. Two lemmas are presented. These lemmas imply that the designs could be verified by the constraints formulated with an arbitrary spanning tree instead of verifying all possible spanning trees.

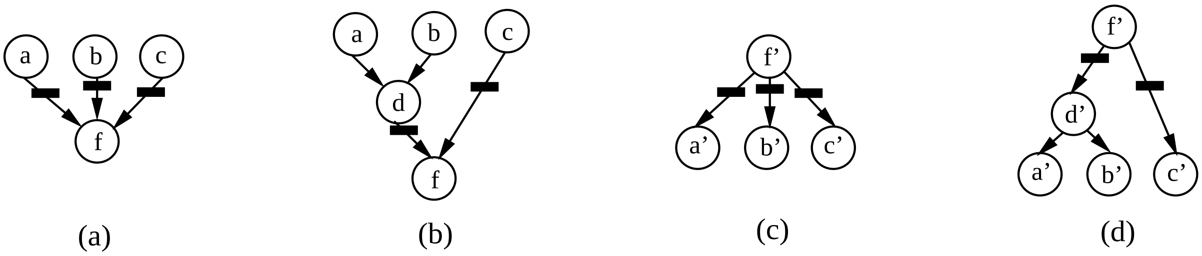

Figure 5: Special Considerations

Lemma 1 *If there exists a feasible solution for clock period c for the graph representation as shown in Fig. 4(a), then at node t, data* d_i *traversing along the top path can meet data* $d_{i-m(q-p)}$*, which traverses along the bottom path. Moreover,* d_i *would not interfere with any other data.*

Sketch of proof: It can be shown that the latest arrival times of data d_i traversing along the top path and of data $d_{i-m(q-p)}$ which traverses along the bottom path at node t are smaller than or equal to the earliest arrival time of data $d_{i-m(q-p)+1}$ traversing along the bottom path. Similarly, the earliest arrival times of data d_i traversing along the top path and of data $d_{i-m(q-p)}$ traversing along the bottom path at node t are greater than or equal to the latest arrival time of data $d_{i-m(q-p)-1}$ which traverses along the bottom path. $\Box$

Lemma 2 *If there exists a feasible solution for the constraints formulated in terms of the graph representation as shown in Fig. 4(a) for clock period c, then there must exist a feasible solution for the constraints formulated for the graph representation as shown in Fig. 4(b) if the same intentional clock skews are maintained.*

Sketch of proof: The feasible solution can be obtained by adding $m(q-p)c$ to the variables, $s(t), b(t), g(t), h(t)$, of node t and its descendant nodes in the spanning tree. And the values of variables of the other nodes are not changed. This transformation results in a feasible solution for the constraints formulated for the graph model as shown in Fig. 4(b). $\Box$

4.4 Special Considerations

A designer may require that registers in some fan-in/fanout edges of a node be clocked without skew among them. The graph model can be modified for such cases. Consider the example shown in Fig. 5(a), in which the registers at edges $e(a, f)$ and $e(b, f)$ are required to be clocked without skew between them. A dummy node, d, can then be inserted as shown in Fig. 5(b). Similarly, in the example shown in Fig. 5(c), the registers at edges $e(f', a')$ and $e(f', b')$ are required to be clocked without skew between them. A dummy node d' is inserted as shown in Fig. 5(d). Other considerations might be required for the preservation of the temporal equivalence after applying wave pipelining [12, 18]. This can be ensured by the following constraint, where k is the number of registers along the spanning tree from $s(u)$ to $s(v)$.

$$s(v) + t_s - s(u) = (k+1) \cdot m \cdot c \qquad u \in V_{pi} \text{ and } v \in V_{po} \tag{14}$$

5 Clock Optimization via Retiming

In the previous section, a LP is formulated for clock period minimization if the locations of the registers are fixed. Retiming reduces the clock period by means of re-positioning registers. Modification of the constraints is needed in the LP formulation to allow retiming and wave pipelining to be applied simultaneously. This optimization formulation becomes a mixed integer linear programming problem.

5.1 Mixed Integer LP Formulation

Since no advanced information about registers' position is available after applying retiming, the graph model must be modified accordingly. For a node with multiple fanout edges, a dummy fanout node is inserted at each fanout edge. For example, Fig. 3(c) illustrates the modified graph for the system as shown in Fig. 3(a). V_{fo1} now becomes the set of {11,12,13,14,15,16,17}, and both $t_{max}(v)$ and $t_{min}(v)$ for every node $v \in V_{fo1}$ are set to zero. However, the shift of time origin for each node of V_l is invariant in retiming. If more than one register is allowed at an edge, a dummy node $v \in V_{fo2}$ should be inserted along the edge. In this example, at most only one register will appear at each edge.

For an edge $u \xrightarrow{e} v$, retiming sets the number of registers at edge e to $w(e) - r(u) + r(v)$, which must be a nonnegative integer. These retiming constraints are expressed as follows:

$$w(e) = r(u) - r(v) \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V_{fo1} \cup V_l \tag{15}$$

$$w(e) \geq r(u) - r(v) \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{16}$$

$$w(e) \leq r(u) - r(v) + 1 \qquad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{17}$$

As described in Section 4.1, node $v \in V_{fo1}$ is used to remove the restriction of applying the same clocking time to the registers on every fanout edge of node u, and node $v \in V_l$ is used to subtract the amount of the shift of time origin with the chord. Therefore, by definition, the number of registers at the fan-in edge of node $v \in V_l \cup V_{fo1}$ is set to 0 after retiming as shown in constraint (15). Constraints (16) and (17) restrict the number of registers at an edge to either zero or one. However, there are some special cases, such as $w(e(7,3))$ in Fig. 3(a), which must be greater than or equal to one since the *register file* is a system functional block.

Some constraints formulated in the previous section are modified by including these new integer variables $r(v)$. Thus the constraint set becomes as follows:

delay constraints:

$$s(u) + t_{max}(v) \leq s(v) \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo1} \cup V_{fo2} \cup V_{po} \text{ and } w(e) = r(u) - r(v) \tag{18}$$

$$b(v) \leq b(u) + t_{min}(v) \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo1} \cup V_{fo2} \cup V_{po} \text{ and } w(e) = r(u) - r(v) \tag{19}$$

$$s(u) \leq g(v) \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = r(u) - r(v) \tag{20}$$

$$h(v) \leq b(u) \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = r(u) - r(v) \tag{21}$$

synchronous constraints:

$$s(u) + t_{max}(v) + t_s + t_{cl} \leq s(v) \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo2} \cup V_{po} \text{ and } w(e) = r(u) - r(v) + 1 \tag{22}$$

$$b(v) \leq s(u) + t_{min}(v) + t_s + t_{cs} \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo2} \cup V_{po} \text{ and } w(e) = r(u) - r(v) + 1 \tag{23}$$

$$s(u) \leq g(v) + t_s + t_{cl} \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = r(u) - r(v) + 1 \tag{24}$$

$$h(v) \leq s(u) + t_s + t_{cs} \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = r(u) - r(v) + 1 \tag{25}$$

wave pipelining constraints:

$$s(u) + t_s + t_h \leq b(u) + c \quad \text{for } u \in V_{po} \text{ or for } u \xrightarrow{e} v \text{ and } w(e) = r(u) - r(v) + 1 \tag{26}$$

$$g(v) + t_{stable}(v) \leq h(v) + c \quad \text{for } v \in V \tag{27}$$

Given a clock period c, if these constraints are satisfied by variables s, b, g, h, and r, then the clock period is a feasible solution. However, solution is impeded by the qualifying clauses in constraints (18)-(26). By appropriate transformations, the optimization with these constraints becomes a mixed integer LP problem. Such a transformation is only valid if $t_h \geq t_{cs}$. Nevertheless, this condition could always be satisfied by increasing the value of t_h such that it is equal to t_{cs}.

Theorem 1 *A pipelined system is defined by $G = (V, V_{fo1}, V_{fo2}, V_l, V_{pi}, V_{po}, E, w, t_{max}, t_{min}, t_{stable}, os)$. If G has a multiplexing degree of m, and $t_h \geq t_{cs}$ for all registers in G, then G correctly operates at clock period c if and only if there exists a feasible solution for the following mixed integer LP constraints. In the constraints, R, T, P, Q are real variables and r is an integer variable.*

retiming constraints:

$$r(u) - r(v) = w(e) \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V_{fo1} \cup V_l \tag{28}$$

$$r(u) - r(v) \leq w(e) \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{29}$$

$$r(u) - r(v) \geq w(e) - 1 \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{30}$$

delay constraints:

$$R(u) - R(v) + r(u) - r(v) \leq -\frac{t_{max}(v)}{c} \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo1} \cup V_{fo2} \cup V_{po} \tag{31}$$

$$T(v) - T(u) \leq w(e) + \frac{t_{min}(v)}{c} \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo1} \cup V_{fo2} \cup V_{po} \tag{32}$$

$$R(u) - P(v) + r(u) - r(v) \leq 0 \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \tag{33}$$

$$Q(v) - T(u) \leq w(e) \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \tag{34}$$

synchronous constraints:

$$R(u) - R(v) \leq -\frac{t_{max}(v)}{c} - \frac{t_s}{c} - \frac{t_{cl}}{c} - w(e) + 1 \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{35}$$

$$T(v) - R(u) - r(u) + r(v) \leq \frac{t_{min}(v)}{c} + \frac{t_s}{c} + \frac{t_{cs}}{c} \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{36}$$

$$R(u) - P(v) \leq -\frac{t_s}{c} - \frac{t_{cl}}{c} - w(e) + 1 \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \tag{37}$$

$$Q(v) - R(u) - r(u) + r(v) \leq \frac{t_s}{c} + \frac{t_{cs}}{c} \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \tag{38}$$

loop/semi-loop constraints:

$$R(v) - R(u) = -m \cdot os(v) \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V_l \tag{39}$$

$$T(v) - T(u) = -m \cdot os(v) \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V_l \tag{40}$$

wave pipelining constraints:

$$R(u) - T(u) \leq -\frac{t_s}{c} - \frac{t_h}{c} + 1 \qquad \textit{for } u \in V \cup V_{fo1} \cup V_{fo2} \cup V_l \cup V_{po} \tag{41}$$

$$P(v) - Q(v) \leq -\frac{t_{stable}(v)}{c} + 1 \qquad \textit{for } v \in V \tag{42}$$

initiation constraint:

$$T(v) - R(v) \leq 0 \qquad v \in V_{pi} \tag{43}$$

Proof: For each constraint, there exists a corresponding constraint in the original qualified LP formulation. These mixed integer LP constraints can be obtained by respectively substituting $c(R(v)+r(v))$ for $s(v)$, $c(T(v)+r(v))$ for $b(v)$, $c(P(v)+r(v))$ for $g(v)$, and $c(Q(v)+r(v))$ for $h(v)$. It can be proven that if and only if constraints in the qualified LP formulation are satisfied by the values of variables s, g, b, h, and r at clock period c, then the corresponding constraints are also satisfied by the values of variables R, T, P, Q, and r at clock period c. The values of variables R, T, P, and Q are obtained from the values of variables s, g, b, h, and r by the above transformation equations, and vice versa. The details are given in the appendix. $\square$

5.2 Optimal Clock Period

For a given clock period c, the mixed integer LP constraints can be tested to see if it is a feasible solution. In a single closed loop l with $w(l)$ registers, the longest path delay along the loop is $T_{max}(l)$. The feasible clock period is bounded below by $\frac{T_{max}(l)}{m \cdot w(l)}$ for a multiplexing degree m [13], and the lower bound of the feasible clock period is equal to $\max_{\forall\ l} \frac{T_{max}(l)}{m \cdot w(l)}$. This serves as a starting point for the search of an optimal clock period. In fact, this is a *maximum-ratio-cycle* problem which can be solved in polynominal time by Hartmann [8]. He shows an algorithm with a running time of $O(T(|E| + |V| \log |V|))$, where T is the sum of all node delays.

Designers, however, usually care about a system with a multiplexing degree of 1. The following theorem facilitates the search for the optimal clock period.

Theorem 2 *For a pipelined system with a multiplexing degree of 1, if clock period c is not feasible, any clock period $< c$ will not be feasible.*

Sketch of proof: The feasible solution set of any LP problem must be convex. If there exists at least one feasible solution for a given configuration, it can be proven that the solution set must be unbounded above. The feasible solution set of the clock period for the above mixed integer LP can be regarded as the union of solution sets for all configurations of a system. This means that the feasible solution set of c is continuous and unbounded above. $\square$

Theorem 2 suggests a way to obtain the optimal clock period for a system with multiplexing of one. The optimal clock period can be obtained by a binary search over possible clock periods.

5.3 Design Example: RISC Microprocessor

In this section, we demonstrate the application of our formulation to a RISC microprocessor design as shown in Fig. 6(a). The numbers inside the circles indicate the longest and shortest propagation delay for each node. For simplicity, the setup time, hold time, and propagation delay of a register are

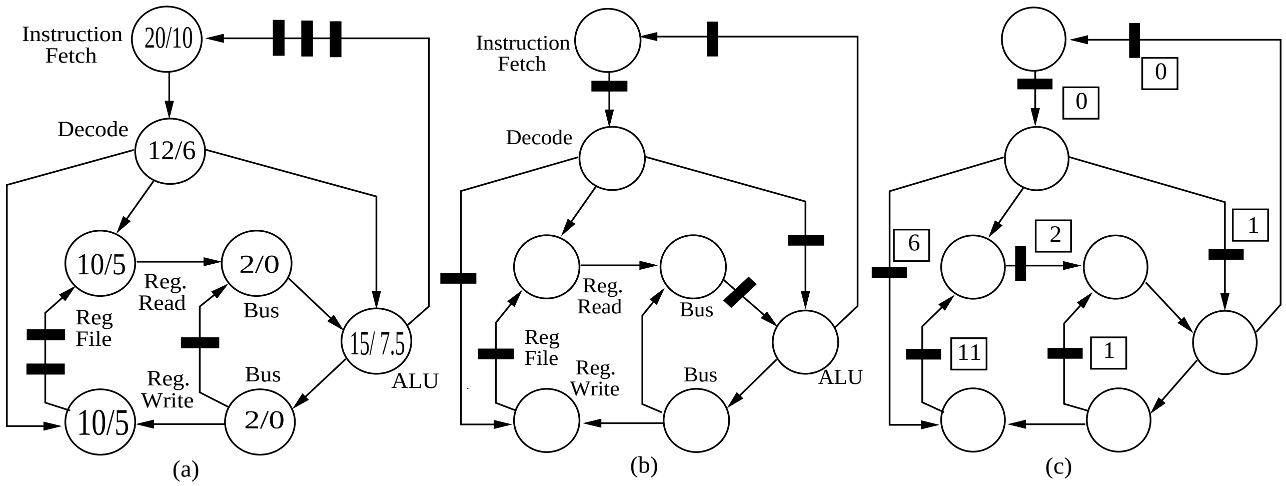

Figure 6: Optimal Configurations for Zero and Non-zero Skew Designs

set to 0. This design was previously analyzed by Lockyear [8]. There are two fundamental differences between our model and Lockyear's. First, our model is based on edge-triggered registers rather than latches due to the stringent synchronization requirement of multi-staged wave pipelining. Second, in our model intentional skews at registers are variables and can be optimized. In contrast, in Lockyear's model intentional skew at every latch is simply fixed at a given value.

Once the set of constraints are captured, optimal solution is obtained by LINDO, a mixed integer linear programming solver. Without retiming, the cycle time of the given system is $71ns$ if no skew is allowed at each register (zero skew design). With retiming, the system's optimum cycle time becomes $27ns$. The result is shown in Fig. 6(b). According to our formulation, if non-zero skews are allowed at registers, the optimal clock period is further reduced to $20ns$ as shown in Fig. 6(c). The intentional clock skew for each register is shown by the number inside the rectangle box. It is worth pointing out that if each combinational block is carefully designed by path balance [10], the cycle time can be further reduced.

6 Design Robustness

Timing constraints are formulated so that at a feasible clock period a system is immune to zero and double clocking [4]. When a system is implemented, an amount of unintentional clock skew exists at each register due to manufacturing variations. In other words, registers are not clocked exactly at the specific moment for which they are designed. Generally, the exact moment is not predictable because of process variations, environmental and temperature fluctuation, different propagation delay variations along routing wires, etc. These uncertainties prevent a system from operating at its theoretically optimal clock period. In order to accommodate these uncertainties, a conservative approach is to operate the system at a larger clock period. However we are interested in finding the maximal amount of *safety margin*, denoted by the variable δ, by which the clock occurrence can deviate from the design specification time t. In other words, the system still operates correctly as long as the clock is delivered in the interval of $[t - \delta,\ t + \delta]$. In this way, the allowable clock skew becomes a continuous range, instead of a single discrete value. Constraints (22)-(27) must be modified as follows:

synchronous constraints:

$$s(u) + t_{max}(v) + t_s + t_{cl} + 2\delta \leq s(v) \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo2} \cup V_{po} \text{ and } w(e) = r(u) - r(v) + 1 \tag{44}$$

$$b(v) \leq s(u) + t_{min}(v) + t_s + t_{cs} + 2\delta \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \cup V_{fo2} \cup V_{po} \text{ and } w(e) = r(u) - r(v) + 1 \tag{45}$$

$$s(u) + t_s + t_{cl} + 2\delta \leq g(v) \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = r(u) - r(v) + 1 \tag{46}$$

$$h(v) \leq s(u) + t_s + t_{cs} + 2\delta \quad \text{for } u \xrightarrow{e} v \text{ and } v \in V \text{ and } w(e) = r(u) - r(v) + 1 \tag{47}$$

wave pipelining constraints:

$$s(u) + t_s + t_h + 4\delta \leq b(u) + c \quad \text{for } u \in V_{po} \text{ or for } u \xrightarrow{e} v \text{ and } w(e) = r(u) - r(v) + 1 \tag{48}$$

$$g(v) + t_{stable}(v) + 2\delta \leq h(v) + c \quad \text{for } v \in V \tag{49}$$

For a node u which is connected to a register, the clocking time of this register now becomes $s(u) + t_s + 2\delta$. The additional amount of 2δ is needed to prevent zero clocking in the presence of non-zero unintentional clock skew. This can happen when its input registers are clocked too late and its output registers are clocked too early. Conversely, when input registers are clocked too early and output registers are clocked too late, or when some input registers are clocked too early and some are too late, constraints (48) and (49) prevent the later data from overriding the previous data by the safety margin of δ. Robustness is thus defined as the maximal amount of safety margin for a given clock scheme. In a non-zero design style, for a given clock period, the robustness can be obtained by maximizing the safety margin of δ.

Theorem 3 *A pipelined system is defined by $G = (V, V_{fo1}, V_{fo2}, V_l, V_{pi}, V_{po}, E, w, t_{max}, t_{min}, t_{stable}, os)$. If G has a multiplexing degree of m, a safety margin δ, and $t_h + 2\delta \geq t_{cs}$ for all registers in G, then G operates correctly at clock period c if and only if there exists a feasible solution for the mixed* *integer LP constraints described as follows. In the constraints, R, T, P, Q are real variables and r is an integer variable.*

retiming constraints:

$$r(u) - r(v) = w(e) \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V_{fo1} \cup V_l \tag{50}$$
$$r(u) - r(v) \leq w(e) \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{51}$$
$$r(u) - r(v) \geq w(e) - 1 \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{52}$$

delay constraints:

$$R(u) - R(v) + r(u) - r(v) \leq -\frac{t_{max}(v)}{c} \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo1} \cup V_{fo2} \cup V_{po} \tag{53}$$
$$T(v) - T(u) \leq w(e) + \frac{t_{min}(v)}{c} \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo1} \cup V_{fo2} \cup V_{po} \tag{54}$$
$$R(u) - P(v) + r(u) - r(v) \leq 0 \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \tag{55}$$
$$Q(v) - T(u) \leq w(e) \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \tag{56}$$

synchronous constraints:

$$R(u) - R(v) \leq -\frac{t_{max}(v)}{c} - \frac{t_s}{c} - \frac{t_{cl}}{c} - \frac{2\delta}{c} - w(e) + 1 \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{57}$$
$$T(v) - R(u) - r(u) + r(v) \leq \frac{t_{min}(v)}{c} + \frac{t_s}{c} + \frac{t_{cs}}{c} + \frac{2\delta}{c} \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \cup V_{fo2} \cup V_{po} \tag{58}$$
$$R(u) - P(v) \leq -\frac{t_s}{c} - \frac{t_{cl}}{c} - \frac{2\delta}{c} - w(e) + 1 \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \tag{59}$$
$$Q(v) - R(u) - r(u) + r(v) \leq \frac{t_s}{c} + \frac{t_{cs}}{c} + \frac{2\delta}{c} \qquad \textit{for } u \xrightarrow{e} v \textit{ and } v \in V \tag{60}$$

loop/semi-loop constraints: same as shown in Theorem 1

wave pipelining constraints:

$$R(u) - T(u) \leq -\frac{t_s}{c} - \frac{t_h}{c} - \frac{4\delta}{c} + 1 \qquad \textit{for } u \in V \cup V_{fo1} \cup V_{fo2} \cup V_l \cup V_{po} \tag{61}$$
$$P(v) - Q(v) \leq -\frac{t_{stable}(v)}{c} - \frac{2\delta}{c} + 1 \qquad \textit{for } v \in V \tag{62}$$

initiation constraint: same as shown in Theorem 1

Proof: Due to similarity with the proof of Theorem 1, the proof of Theorem 3 is omitted here. □

For a feedback loop which contains n registers, an upper bound of the safety margin can be calculated by Theorem 4. $T_{max}(i)$ and $T_{min}(i)$ are defined as the longest and shortest path delays of each stage i where $1 \leq i \leq n$.

Theorem 4 *A general loop containing n registers is operated at a clock period c and a multiplexing degree m. The safety margin of δ is limited by the following three inequalities:*

$$\delta \leq \frac{1}{2}(m \cdot c - \frac{\sum_{i=1}^{n} T_{max}(i)}{n} - t_s - t_{cl}) \tag{63}$$

$$\delta \leq \frac{1}{2}((1-m)c + \frac{\sum_{i=1}^{n} T_{min}(i)}{n} + t_{cs} - t_h) \tag{64}$$

$$\delta \leq \min_i \{\frac{1}{4}(c - T_{max}(i) + T_{min}(i) + t_{cs} - t_s - t_h - t_{cl})\} \tag{65}$$

Proof: The edge pointed from stage n to stage 1 is selected as a chord. Constraints (44) for such a spanning tree can be formulated as follows:

$$s(i) + T_{max}(i+1) + t_s + t_{cl} + 2\delta \leq s(i+1) \qquad 1 \leq i \leq n-1$$

Loop/semi-loop constraint (9) and constraint (44) for the chord can be written together as follows:

$$s(n) - n \cdot m \cdot c + T_{max}(1) + t_s + t_{cl} + 2\delta \leq s(1)$$

Summing all these inequalities leads to constraint (63).

Constraints (45) and (48) for the spanning tree can be written as follows:

$$b(i+1) \leq s(i) + T_{min}(i+1) + t_s + t_{cs} + 2\delta \qquad 1 \leq i \leq n-1$$

$$s(i+1) + t_s + t_h + 4\delta \leq b(i+1) + c \qquad 1 \leq i \leq n-1$$

Loop/semi-loop constraints, constraint (45) and constraint (48) for the chord can be formed together as follows.

$$\begin{aligned} b(1) &\leq s(n) - n \cdot m \cdot c + T_{min}(1) + t_s + t_{cs} + 2\delta \\ s(1) + t_s + t_h + 4\delta &\leq b(1) + c \end{aligned}$$

Again summation of the left and right sides of these inequalities gives constraint (64).

For each stage i, constraints (44), (45), and (48) can be expressed by the following inequalities.

$$s(i) + T_{max}(i+1) + t_s + t_{cl} + 2\delta \leq s(i+1)$$

$$b(i+1) \leq s(i) + T_{min}(i+1) + t_s + t_{cs} + 2\delta$$

$$s(i+1) + t_s + t_h + 4\delta \leq b(i+1) + c$$

Summation of these inequalities gives constraint (65). $\square$

c	20.0	22.0	24.0	25.0	26.0	27.0	28.0	29.0	30.0	31.0	32.0	33.0	34.0	35.0
d1	-	-	-	-	-	0.0	0.5	0.75	1.5	1.75	2.5	2.5	2.5	2.5
d2	0.0	0.5	1.0	1.25	1.5	1.75	2.0	2.25	2.5	2.75	3.0	3.25	3.35	3.35

Table 2: Robustness at Various Clock Periods (c: clock period, d1: zero skew design, d2: non-zero skew design.)

If the multiplexing degree is 1, the right side of inequality (64) is $\frac{1}{2}(\frac{1}{n}\sum_{i=1}^{n} T_{min}(i) + t_{cs} - t_h)$, which is not dependent on the clock period. It can be seen that for a general loop the robustness no longer increases when the clock period is increased to a threshold value that is dependent on the longest and shortest path delays.

6.1 Robustness for Design Examples

Depending on the skew at each register, a system could be either a zero-skew design or non-zero skew one. In this section, the robustness for both styles are studied for the design example in Section 5.3. For simplicity, the setup time, hold time and propagation delay of registers in both designs are zero. In a zero-skew design, the robustness is calculated by the following formula where $T_{max}(i)$ and $T_{min}(i)$ represent the longest and shortest propagation delay of combinational stage i.

$$\delta = \frac{1}{2}(\min_{\forall i}\{c - T_{max}(i) - t_s - t_{cl}, T_{min}(i) + t_{cs} - t_h\}) \tag{66}$$

The minimum clock cycle period is $27ns$ if it is zero skew design style, but unfortunately, the robustness is zero. In contrast, for this clock period the robustness is $1.75ns$ if non-zero skew is employed. The solution for a non-zero skew design is a by-product of LINDO. Table 2 shows the robustness of both designs. It is important to note that the robustness for zero skew design saturates at $c = 32ns$, while it saturates at $c = 34ns$ if non-zero skew is employed. One way to further increase the safety margin is to equalize the path delays. This is usually achieved through careful padding of the shortest paths.

7 Other Design Examples

Besides the study of the RISC microprocessor, our formulation has been applied to two additional examples: a correlator [5] and a multiplier [8]. Our method improves the performance of both designs if both retiming and wave pipelining are applied simultaneously.

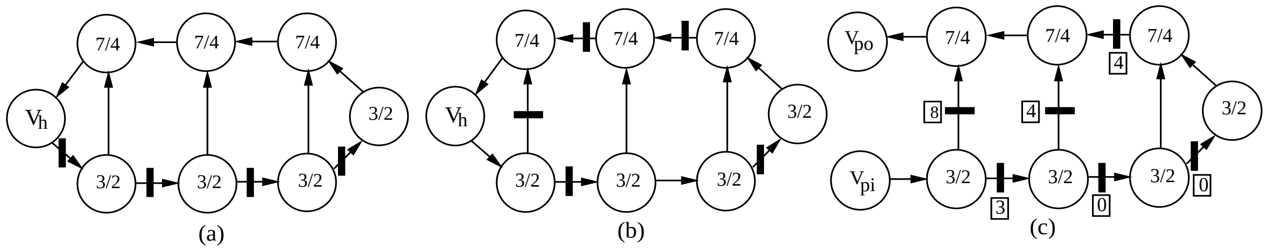

Figure 7: Correlator Design (numbers in circles denote longest and shortest propagation delays for each node.)

c	6.0	8.0	10.0	12.0	13.0	14.0	16.0	18.0	20.0	22.0	24.0	26.0	28.0
d1	-	-	-	-	0.0	0.5	1.0	1.0	1.0	1.0	1.0	1.0	1.0
d2	0.0	0.5	1.0	1.5	1.75	2.0	2.33	2.67	3.0	3.33	3.75	4.25	4.71

Table 3: Robustness at Various Clock Period (c: clock period, d1: zero skew design, d2: non-zero skew design.)

7.1 Correlator

For the zero-skew design as shown in Fig. 7(a), the clock period is $24ns$ before applying any optimization technique. Retiming could reduce the clock period to $13ns$ as shown in Fig. 7(b). Further reduction is possible if wave pipelining and retiming techniques are applied simultaneously. In this example, the optimal clock period is $6ns$. Fig. 7(c) shows the locations of registers and their associated intentional skews. The skew of external input registers is 0, while the skew of external output registers is $12ns$. Note that by using non-zero skew in the design as shown in Fig. 7(b), the minimum clock period is only $7ns$, which gives a sub-optimal solution. The robustness for both the zero skew and non-zero skew designs is shown in Table 3.

7.2 Multiplier

Fig. 8(a) is a serial-parallel multiplier with a minimal clock period of $6ns$. The effects of the shortest paths and register positions on the clock period are studied by the designs as shown in Fig. 8(b) and 8(c). Table 4 shows that an equal amount of the shortest propagation delays is set to nodes v_1, v_2, v_3, and v_4 as shown in the top entry of each column. Also, an equal amount of the shortest propagation delays is set to nodes v_5, v_6, v_7, and v_8 as shown in the leftmost column of Table 4. The optimal clock period, however, is independent of the shortest delays at nodes v_9, v_{10}, v_{11}, and v_{12}. The minimum clock period for the design shown in Fig. 8(b) appears above the slash and for the design

	0.0	0.2	0.4	0.6	0.8	1.0
0.0	4.0/3.0	4.0/3.0	4.0/3.0	4.0/3.0	4.0/3.0	4.0/3.0
0.1	3.6/3.0	3.6/2.9	3.6/2.8	3.6/2.7	3.6/2.7	3.6/2.7
0.2	3.2/3.0	3.2/2.9	3.2/2.8	3.2/2.7	3.2/2.6	3.2/2.5
0.3	3.0/3.0	2.9/2.9	2.8/2.8	2.8/2.7	2.8/2.6	2.8/2.5
0.4	3.0/3.0	2.9/2.9	2.8/2.8	2.7/2.7	2.6/2.6	2.5/2.5
0.5	3.0/3.0	2.9/2.9	2.8/2.8	2.7/2.7	2.6/2.6	2.5/2.5
1.0	3.0/3.0	2.9/2.9	2.8/2.8	2.7/2.7	2.6/2.6	2.5/2.5

Table 4: Dependency of Optimal Clock Period on Multipliers and Shortest Delays

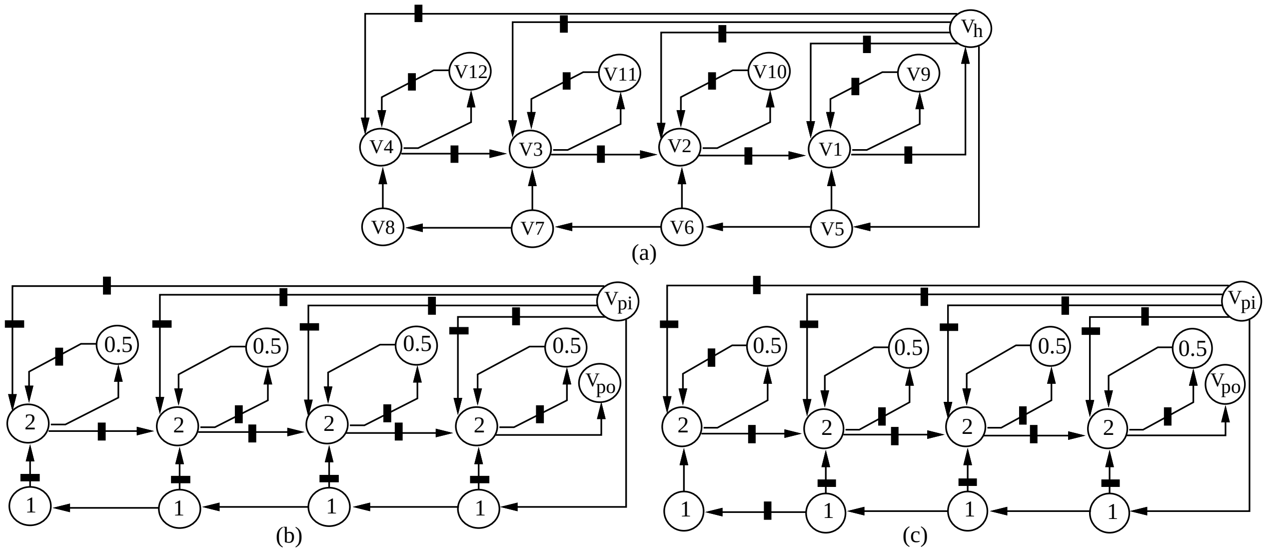

Figure 8: Multipliers (numbers in circles are longest delays for each node.)

shown in Fig. 8c below the slash. It is worth pointing out that the optimal clock period is reduced when the shortest delays for these nodes are prolonged. However, the clock period eventually saturates as shown in Table 4. Thus, it is no longer beneficial to pad the short paths. Another point is that different configurations need different paddings to achieve the same clock period. For the design shown in Fig. 8(c), a clock period of $2.7ns$ is achieved by padding $0.1ns$ for the shortest path of node v_5, v_6, v_7, and v_8 as well as $0.6ns$ for nodes v_1, v_2, v_3, and v_4. In contrast, the design, shown in Fig. 8(b), needs $0.4ns$ for the shortest path of node v_5, v_6, v_7, and v_8 to achieve the same clock period.

8 Optimal Padding

When wave pipelining is applied to a combinational block, the minimum clock period can be reduced by padding short paths [10]. When padding a pipelined system for a specific clock period, two considerations should be taken in order to minimize the area. The first one is to determine which combinational block should be padded. For the multiplier design, the clock period is not reduced by padding nodes v_9, v_{10}, v_{11}, and v_{12}. Also the same clock period could be achieved by padding different nodes in different configurations. However, the area cost incurred by padding each node and by re-positioning registers may be different. Therefore, a global optimization should be proposed in advance, instead of as a postprocessing step. The second consideration is to determine by how much the short paths of a combinational block should be padded. As in the multiplier example, the clock period will eventually saturate such that padding a node past a threshold value is of no use.

It has been shown that padding the short paths in a combinational block can be formulated as a mixed integer LP problem [16]. Area versus delay of the combinational block's shortest path can be approximated as a piecewise linear function by calculating area in terms of the different shortest path delays [17]. The global optimization of area can be formulated as follows.

Min A
subject to:
$A = A_1 + A_2$
$A_1 =$ area cost incurred by prolonging combinational blocks
$A_2 =$ area cost incurred by re-positioning registers
timing constraints for a given clock period c
$\vdots$
constraints of area cost versus delay of the shortest path for each combinational block
$\vdots$

9 Conclusion

We have first developed a graph model for digital systems and then proved that timing optimization is a mixed integer linear programming problem. The model is general enough because feedback loops and reconvergent fanouts are allowed. This mathematical formulation is capable of performing an optimization on a given system by *simultaneously* applying the techniques of adjusting clock skew, retiming the system, and wave pipelining. Our formulation has been applied to three design examples and has significantly improved each one if all three techniques are used concurrently We have also defined and investigated the issue of design robustness. As the clock period is increased, robustness increases but eventually saturates no matter if there exists a feedback loop in a zero skew design. For a non-zero skew design with a feedback loop, robustness saturates as well, and its upper bound can be derived before executing a LP solver. Further research is needed to determine if there exists an upper bound for a non-zero skew design without feedback loop. As shown in Section 6, a non-zero clock skew design is more robust than a zero clock skew one. In Section 5, we have also presented a procedure for obtaining the optimal solution for the mixed integer linear programming formulation. However, further research is needed to unveil an efficient algorithm for larger problems. Finally, this paper also addressed the problem of area minimization when both padding and retiming are used.

Appendix

Proof of Theorem 1: Due to similarities between all the constraints, we only prove constraints (31) and (32). If constraints for the qualified LP formulation can be satisfied by the values of variables s, b, g, h, and r, then the corresponding constraints for the mixed integer linear formulation can be satisfied by the values of variables R, T, P, Q, and r obtained by the given transformation rules.

Constraint (1) $\Rightarrow$ Constraint (31):

$$\begin{aligned} & R(u) - R(v) + r(u) - r(v) + \tfrac{t_{max}(v)}{c} \\ = & \tfrac{s(u)}{c} - r(u) - \tfrac{s(v)}{c} + r(v) + r(u) - r(v) + \tfrac{t_{max}(v)}{c} \\ = & \tfrac{1}{c}(s(u) - s(v) + t_{max}(v)) \leq 0 \end{aligned}$$

The final inequality is valid and independent of $w(e) = 0$ or $w(e) = 1$.

Constraint (1) $\Leftarrow$ Constraint (31):

$$\begin{aligned} & s(u) - s(v) + t_{max}(v) \\ = & c(R(u) + r(u)) - c(R(v) + r(v)) + t_{max}(v) \\ = & c(R(u) - R(v) + r(u) - r(v) + \tfrac{t_{max}(v)}{c}) \leq 0 \end{aligned}$$

Constraint (2) $\Rightarrow$ Constraint (32):

$$\begin{aligned} & T(v) - T(u) - \tfrac{t_{min}(v)}{c} - w(e) \\ = & \tfrac{b(v)}{c} - r(v) - \tfrac{b(u)}{c} + r(u) - \tfrac{t_{min}(v)}{c} - w(e) \\ = & \tfrac{1}{c}(b(v) - b(u) - t_{min}(v)) - (w(e) - r(u) + r(v)) \end{aligned}$$

When $w(e) = r(u) - r(v)$, (*i.e.* there is no register at edge e), expression $b(v) - b(u) - t_{min}(v)$ is smaller than or equal to 0. The above expression must therefore be smaller than or equal to 0. If $w(e) = r(u) - r(v) + 1$, (*i.e.* there exists one register at edge e), expression $\frac{1}{c}(b(v) - b(u) - t_{min}(v))$ is smaller than or equal to 1 when $t_h \geq t_{cs}$. This can be shown as follows:

$$\begin{aligned} & b(v) - b(u) - t_{min}(v) \\ \leq & s(u) + t_s + t_{cs} + t_{min}(v) - b(u) - t_{min}(v) \\ \leq & s(u) - b(u) + t_s + t_{cs} \\ \leq & b(u) + c - t_s - t_h - b(u) + t_s + t_{cs} \\ \leq & c - t_h + t_{cs} \end{aligned}$$

The second inequality is derived from constraint (23) and the fourth one is from constraint (26). Since expression $b(v)-b(u)-t_{min}(v)$ is smaller than or equal to c, expression $\frac{1}{c}(b(v)-b(u)-t_{min}(v))$ is smaller than or equal to 1, and $-(w(e)-r(u)+r(v))$ is equal to -1. Expression $T(v)-T(u)-\frac{1}{c}t_{min}(v)-w(e)$ is consequently smaller than or equal to 0 and independent of $w(e)=0$ or 1.

Constraint (2) $\Leftarrow$ Constraint (32):

$$
\begin{aligned}
& b(v)-b(u)-t_{min}(v) \\
= \quad & c(T(v)+r(v))-c(T(u)+r(u))-t_{min}(v) \\
= \quad & c(T(v)-T(u)-\tfrac{t_{min}(v)}{c}-r(u)+r(v))
\end{aligned}
$$

When $w(e)=r(u)-r(v)$, the final expression becomes $c(T(v)-T(u)-\frac{1}{c}t_{min}(v)-w(e))$, which must be smaller or equal to 0.

References

[1] S. Malik, E. M. Sentovich, R. K. Bryton, and A. Sangiovanni-Vincentelli, "Retiming and Resynthesis: Optimizing Sequential Networks with Combinational Techniques," *IEEE Trans. on Computer-Aided Design,* vol. 10, pp. 74-84, Jan. 1991.

[2] H. Hsieh, W. Liu, and R. Cavin, "Framework for Parametric Timing Optimization for Latch-Based Digital Systems," Department of Electrical and Computer Engineering, North Carolina State University, Aug. 1994.

[3] K. Bartlett, G. Borriello, and S. Raju, "Timing Optimization of Multiphase Sequential Logic," *IEEE Trans. on Computer-Aided Design,* vol. 10, pp. 51-62, Jan. 1991.

[4] J. P. Fishburn, "Clock Skew Optimization," *IEEE Trans. on Computers,* vol. 39, pp. 945-951, July, 1990.

[5] C. E. Leiserson and F. M. Rose, "Optimization Synchronous Circuitry by Retiming," *Proceedings of the Third Caltech Conference on VLSI,* pp. 87-116, 1983.

[6] K. A. Sakallah, T. N. Mudge, and O. A. Olukotun, "Analysis and Design of Latch-Controlled Synchronous Digital Circuits," *IEEE Trans. on Computer-Aided Design,* vol. 11, pp. 322-333, Mar. 1992.

[7] A. T. Ishii and C. E. Leiserson, "Optimizing Two-Phase Level-Clocked Circuitry," *Advanced Research in VLSI and Parallel Systems: Proceedings of the 1992 Brown/MIT Conference,* pp. 239-264, 1992

[8] B. Lockyear and C. Ebeling, "The Practical Application of Retiming to the Design of High-Performance Systems," *Proceedings of the ICCAD*, pp. 288-295, 1993

[9] N. Shenoy and R. K. Brayton, "Retiming of Circuits with Single Phase Transparent Latches," *International Conference on Computer Design,* pp. 86-89, 1991.

[10] D. Wong, G. De Micheli, and M. J. Flynn, "Design High-Performance Digital Circuits Using Wave Pipelining: Algorithms and Practical Experiences," *IEEE Trans. on Computer-Aided Design,* vol. 12, pp. 25-46, Jan. 1993.

[11] D. A. Joy and M. J. Ciesielski, "Clock Period Minimization with Wave Pipelining," *IEEE Trans. on Computer-Aided Design,* vol. 12, pp. 461-472, Apr. 1993.

[12] B. C. Ekroot, "Optimization of Pipelined Processors by Insertion of Combinational Logic Delay," Ph.D. dissertation, Stanford University, Sept. 1987.

[13] C. T. Gray, W. Liu, and R. Cavin III, Wave Pipelining: Theory and CMOS Implementations, Kluwer Academic Publisher, Oct. 1993.

[14] W. K. C. Lam, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Valid Clocking in Wave Pipelined Circuits," *Proceedings of the ICCAD*, Nov. 1992.

[15] K. A. Sakallah, T. N. Mudge, T. M. Burks, and E. S. Davidson, "Synchronization of Pipelines," *IEEE Trans. on Computer-Aided Design,* Vol. 12, pp. 1132-1146, Aug. 1993.

[16] N. V. Shenoy, R. K. Brayton, A. Sangiovanni-Vincentelli, "Minimum Padding to Satisfy Short Path Constraints," *Proceedings of the ICCAD*, pp. 156-161, Nov. 1993.

[17] J. C. Shyur, H. P. Chen, and T. M. Parng, "Achieve Timing Error-Free Wave Pipelining for Latch Based CMOS Synchronous Systems," *TAU'93* , Sept. 1993.

[18] N. Shenoy, K. Singh, R. K. Brayton, and A. Sangiovanni-Vincentelli, "On the Temporal Equivalence of Sequential Circuits", *Proceedings of the IEEE/ACM Design Automation Conference,* pp. 405-409, 1992.