


Acknowledgments
We thank Glenn Jennings for helpful discussions.

References


B.2.3.2 The Clocking System Implementation.

The complementary clocks. Clock \( \phi \) is distributed globally, and \( \bar{\phi} \) is created locally (e.g., separately for each latch).

Clock signal rout. The chip receives the master clock generated outside it. This clock (of 600 MHz) is divided by two and then routed to the center of the chip. It is then amplified through 6 inverter levels and fanned-out to 24 inverters through a balanced tree to drive the preliminary clock signal. This signal is routed to the final stage clock buffers, located to the right and to the left of the execution core outside edges. Each of these buffers has 4 more levels of inverter before it drives the single wire \( \phi \) signal. The accumulative gate width of the inverter of the 4-th level is 58 centimeters (cm).\(^1\)

The grid layout. Clock \( \phi \) is distributed over grid composed of horizontal and vertical lines, both of a constant pitch, connected at every cross point. Metal levels 3 and 4 were destined especially for the grid, power supplement and the global routing.

Different wire segments (the pieces between two cross-points) have different width. Some segments may be absent at all. This because of the reasons explained in the following section.

B.2.3.3 Algorithm of Reduction of Grid Capacitance

A sophisticated approximation algorithm was developed in order to reduce the capacitance. (However, this algorithm does not aim at a global minimum.) There were two constraints the algorithm ought to satisfy:

- To keep the clock skew below 100 ps.
- The current density was bounded to prevent electro-migration [38].

It is interesting that the issue of inductive reflections, stressed in [3, 35], is completely ignored here.

The initial clock distribution network was a regular grid satisfying the above constraints. In each iteration of the algorithm the widths of some segments are narrowed (even to zero if possible). The algorithm stops when no reduction is possible. Four iterations were run and a capacitance reduction of 9.6% was achieved.

B.2.3.4 The Implementation Results

The Alpha 21164 chip has the best clock skew, both absolute and relative to the clock cycle, among all the high-performance microprocessors presented in the ISSCC’95 and ’96 [14, 13]. The final evaluation of the clock skew, that was done with Spice, is about 2.7 percent of the period — 90 ps from 3.33 ns.

The passive grid causes a great power consumption. Approximately 40 percent of the total chip power is dissipated in the clocking system. The power goes to drive the load of 3.75 nF of terminals and grid capacitance. Over half of this 40% is due to the grid capacitance. The area of the amplifying buffers is about 5% (our estimation) of the total.

---

\(^1\)Such an inverter should be considered as a large number of parallelly connected inverters. In spite that from the electrical point of view there is no difference between the two structures, in our case the inverters are connected by non-ideal wires and there is a skew between them.
Typically, the clocking system is composed of the following components that are usually intermixed:

1. A clock source external to the chip that creates a single master clock of high precision, whose frequency is typically an integral multiple of that of the internal high-speed clock phases.
2. On-chip clock generator that generates the internal clock signals from the master clock.
3. An amplifier that enables the clock signals to drive the numerous terminals.
4. A clock distribution network that delivers the clock signals to their terminals.

Here we present an example of a clocking system of a high-performance microprocessor.

B.2.2 Alpha Processors

The example is the Digital’s line of Alpha microprocessors. They are interesting case in our study from the following two reasons.

- Each Alpha processor has had the highest performance relative to contemporary processors.
- Each Alpha processor has the fastest clocks. This makes the clock skew a more significant fraction of the clock cycle, and as the result sets stronger requirements on the clocking system.

The Alpha processors family include Alpha 21064 working in 200-MHz clock frequency [7], 300-MHz Alpha 21164 [4] and 433-MHz Alpha 21164 [17] which is the scaling down of the 300-MHz version. The clocking systems in all the Alpha processors are similar. We elaborate on one of the family’s representatives — the 300-MHz Alpha 21164.

B.2.3 300-MHz Alpha 21164 Microprocessor

The description of the 300-MHz Alpha 21164 microprocessor was published in the beginning of 1995 [4]. It is designed in the 0.5-micrometer (\(\mu m\)) CMOS technology using four levels of metal. The die measures 16.5 millimeters (mm) by 18.1 mm and contains 9.3 million transistors. We are going to describe the clocking system design as we understood from [4] and [6].

B.2.3.1 The Clocking System Goals.

In the center of the chip there is a rectangular, named the execution core, where almost all the execution units are grouped (but not the caches). The execution core is the critical part of the chip, working with the 300-MHz clock. We, henceforth, focus on the execution core and ignore the rest of the chip.

The clocking system had to deliver two complementary clocks to approximately 700 000 terminals. The terminals are spread irregularly over the chip. Their total capacitance is about 2 nF. In addition, several ad-hock clocks are employed; we do not consider them here.

The predefined design challenge was to limit the clock skew to 100 ps with economical area and power consumption.

\[1\] We do not know how many transistors are in the core, but we assume the terminals to constitute a significant fraction of them since the core heavily uses dynamic logic.
There is much theoretical research aimed at reducing the clock skew [10]. However, this research has no impact on the industry, and we ignore it as well.

It's widely accepted that the clock skew is a cardinal issue in contemporary high-speed VLSI. Every presentation of a high-performance processor on the two past International Solid State Circuits Conferences [14, 13] seriously refers to this issue. The cause of its importance is that most of the modern systems have to increase the clock cycle by twice the clock skew, and this reduces the performance. Moreover, it is accepted that this harm will become more serious in the future with the clock frequencies speed up, since while the clock periods get appreciably shorter, the clock skew values scale down in a smaller proportion.

### B.1 Clock Skew Causes

Theoretically, under ideal conditions (i.e., exact manufacturing process, known operating temperatures, etc.) clock signals can be distributed without clock skew.

VLSI manufacturing processes are imperfect [36]. This leads to the differences in the electrical characteristics of the chip components (wires — e.g., resistivity, dielectric constant; transistors - e.g., threshold voltages, channel mobilities) from their nominal values. The deviations in the clock distribution network are responsible for the variations in the clock signals' arrival times. It's very important that these deviations are not known at the design stage, they create uncertainties to the clock signals' behavior.

Afghahi and Svensson [2] developed formulas giving the distribution function of the clock skew. They also claimed that distortions occur mostly in transistors. In practice, reducing the clock skew in economical fashion is a major design challenge.

### B.2 Contemporary Techniques to Reduce Clock Skew

Presentations of the high-performance processors on the two past International Solid State Circuits Conferences [14, 13] demonstrate the contemporary significance of this issue. All the reports dedicate a great part to present the clocking system design and the clock skew handling. The clocking systems in such processors are very sophisticated. They are designed to stand very tough requirements of the clock skew and slope, power consumption, noise and other aspects. Their developers invest a lot of resources in order to evaluate precisely the clock skew on the design stage. In most designs the extracted clocking system is simulated thoroughly apart from the rest of the logic. The maps of the clock delays are built. The differences in these delays are related as clock skew.

The common points of the clocking system design are summarized below.

#### B.2.1 Clocking System in Contemporary Microprocessors

Synchronous systems are considered to consist of two sub-systems:

- the data system that processes the data but generates no timing signals;
- the clocking system that generates the timing signals that control the data system via its memory elements and dynamic gates.

The physical points of interface between these sub-systems are called terminals. Usually, a terminal is a gate of a transistor.
A.2 Inequalities Imply Strictly Functional

In this section we prove the $\iff$ direction of lemma A.1. We use the following lemma.

Let $\Psi(i, \Pi, t)$ be the following statement:

$$\Psi(i, \Pi, t) = \text{is a latch, } \Pi \text{ is a pulse of } \Phi, \ t \leq \Pi^1 \text{ and } R_i(t) \neq \hat{R}_i(\Pi^1).$$

**Lemma A.4** Let $Y$ be CD=0 and $\Phi$ disciplined. Let $t$ be a time, $R \in R(Y, \Phi)$, $\hat{R} \in R(Y^{PD=0}, \Phi)$, $I^R = I^\hat{R}$ and $R \succeq_{[\hat{R}]} \hat{R}$. Let $(c, \Pi, t)$ satisfy $\Psi(c, \Pi, t)$ and $\Pi^1 \leq t$ and $t \in \Pi_c$. Then there are $(b, \Pi, t')$ s.t. $\Psi(b, \Pi, t')$ and $b \sim c$, $\Pi^1_b \leq \Pi^1_i$ and $t' \in (t - P_D(b,c), t)$.

**Proof.** Assume that for every $x \in ?^\hat{c}$ and $t' \in (t - P_D(x, \hat{c}), t)$: $R_x(t') = \hat{R}_x(\Pi^1_x)$. Then according to lemma 4.9, $R_c(t) = \hat{R}_c(\Pi^1_c)$, contradicting the lemma’s premise. Thus, there exists a latch $b \in ?^\hat{c}$ and $t' \in (t - P_D(b, \hat{c}), t)$, such that $R_b(t') \neq \hat{R}_b(\Pi^1_b)$. We’re going to show that $t'$ satisfies the lemma’s conclusion.

Assume that $b$ is opaque during all the interval $I = [t', \Pi^1_b]$. Hence, $\hat{R}_b(t') = \hat{R}_b(\Pi^1_b)$ and $R_b(t') \neq \hat{R}_b(\Pi^1_b)$. A contradiction to the fact that $R \succeq_{[\hat{R}]} \hat{R}$. As the result, there is a subinterval of $I$, where $b$ is transparent. Moreover, from the *unambiguous bit property* follows that there is a fall of a pulse of $\Phi_b$ during $I$. Let $\Pi_b$ be the last pulse of $\Phi_b$ satisfying $\Pi^1_b \leq \Pi^1_b$. So, $t' \leq \Pi^1_b$ and $R_b(t') \neq \hat{R}_b(\Pi^1_b) = \hat{R}_b(\Pi^1_b)$. \hfill $\square$

Let us now establish the $\iff$ direction of lemma A.1. Assume that $(Y, \Phi)$ is not strictly functional. Hence, there is a system $Y \sim Y$ such that $(Y, \Phi)$ isn’t functional. According to lemma 4.5, there are runs $R \in R(Y, \Phi)$ and $\hat{R} \in R(Y^{PD=0}, \Phi)$, $I^R = I^\hat{R}$, a latch $l$ and a pulse $\Pi$ of $\Phi$ such that $R_l(\Pi^1_l) \neq \hat{R}_l(\Pi^1_l)$. Pick such $l$ and $\Pi$ such that $\Pi^1_l$ is minimal. This implies $R \succeq_{[\hat{R}]} \hat{R}$.

We’re going to construct a trace $T$ of $(Y, \Phi)$, s.t. $R_F(T) < P_D(T)$. To this end, we build a (finite or infinite) sequence of triples $Q = \langle \langle l_1, \Pi_1, t_1 \rangle, \langle l_2, \Pi_2, t_2 \rangle, \ldots \rangle$. There are two rules concerning the triples:

**Rule A.1** For every triple $\langle l_i, \Pi_i, t_i \rangle$: $\Psi(l_i, \Pi_i, t_i)$ holds.

**Rule A.2** $l_{i+1} \sim l_i$, $\Pi^1_{i+1} \leq \Pi^1_i$, and $t_{i+1} \in (t_i - P_D(l_{i+1}, l_i), t_i)$ for all $i$.

The first triple $\langle l_1, \Pi_1, t_1 \rangle = \langle l, \Pi, t \rangle$ satisfies the rule A.1. Assume we’ve built the first $i$ elements of $Q$. If $t_i \not\in \Pi_i$, we terminate $Q$. Otherwise, we add an entry to $Q$ as follows. We apply Lemma A.4 with the last triple of $Q$. The lemma provides us a new triple $\langle b, \Pi', t' \rangle$ which we add to $Q$. Clearly, both rules A.1 and A.2 are kept.

We now prove that $Q$ is finite. Otherwise, since the number of latches and the number of pulses before $t_1$ are finite, there are distinct indexes $k < m$ such that $l_k = l_m$ and $\Pi_k = \Pi_m$. That is, the latches $l_1, l_{i+1}, \ldots, l_m$ lay on a directed cycle $C$. From rule A.2, $\langle \Pi^1_k \rangle$ is monotonic decreasing. Thus, $\Pi^1_k = \Pi^1_{k+1} = \ldots = \Pi^1_m$. Hence, at $\Pi^1_k$ all the latches of $C$ are transparent, which contradicts the *no transparent cycle* property of $\Phi$. So, $Q$ is finite. Let $k = |Q|$. We have:

$$t_1 - t_k = \sum_{i=1}^{k-1} t_i - t_{i+1} \leq \sum_{i=1}^{k-1} P_D(l_{i+1}, l_i) \tag{A.1}$$

We now prove that the inequality of A.1 is strong. Otherwise, all the inequalities are weak. From rule A.2 follows that $t_i - t_{i+1}$ for every $i$. However, $t_k \leq \Pi_k < \Pi^1_k \leq \Pi^1_i = t_i$. A contradiction. So, the inequality of A.1 is strong.

Define $T = \langle \langle l_k, l_{k-1}, \ldots, l_1 \rangle, \langle \Pi_k, \Pi_{k-1}, \ldots, \Pi_1 \rangle \rangle$. By our construction, $t_k \leq \Pi_k$; hence, $R_F(T) = \Pi^1_k - \Pi_k \leq t_1 - t_k < P_D(T)$.

This establishes the $\iff$ direction of lemma A.1.
A Proof of The Trace Inequalities Lemma

In this appendix we prove the Trace Inequalities Lemma:

Lemma A.1 (The Trace Inequalities Lemma) Let $Y$ be $CD=0$ and $PD>0$ and $\Phi$ be disciplined. Then $\langle Y, \Phi \rangle$ is strictly functional iff all the inequalities of $TI^\Phi$ hold.

A.1 Strictly Functional Implies the Trace Inequalities

In this section we prove the $\Rightarrow$ direction of lemma A.1. We use the next lemma that follows from Lemma 4.10.

Lemma A.2 Let $R \in \mathcal{R}(Y, \Phi)$, $Y$ be $PD>0$ and $CD=0$ and $c$ an edge. Let $t', t$ be reals, $t' \leq t$, $i \in \mathcal{I}$, and $R_i$ not stable in $(t' - P_D(i, c), t)$. Then there is a run $R' \in \mathcal{R}(Y, \Phi)$ with $I^{R'} = I^R$ such that $R'_i$ is not stable in $[t', t]$.

We next prove the $\Rightarrow$ direction of lemma A.1 by contradiction and induction on the length of the trace associated with the violated inequality of $TI^\Phi$. As frequently happens, we actually prove a stronger statement as follows:

Lemma A.3 Let $\langle Y, \Phi \rangle$ be strictly functional, $Y$ be $CD=0$ and $PD>0$ and $\Phi$ disciplined. Let $T = \langle \langle i_1, \cdots, i_k \rangle, \langle \Pi_1, \cdots, \Pi_k \rangle \rangle$ be a trace of $\langle Y, \Phi \rangle$, $c = i_k$ and $\Pi_c = \Pi_k$. Let $t \in \Pi_c$ be such that $t - 1'^T < P_D(T)$. Then there are $\Pi' \sim Y$ and $R \in \mathcal{R}(Y', \Phi)$ s.t. $R_c$ is not stable in $[t, \Pi_c]$.

In the special case of $t = \Pi_c$, our lemma implies that if its premise holds then $R_c$ is not stable in $[t, t]$; i.e., $R_c(t) = \frac{1}{2}$. This establish the $\Rightarrow$ of lemma A.1.

Proof. We prove the lemma by induction on $k$. Note that by the trace definition, $k \geq 2$. Let $b = i_{k-1}$, $\Pi_b = \Pi_{k-1}$ and $t = t - P_D(b, c)$. Let $T'$ be the sub-trace of $T$ having the first $k-1$ entries.

According to lemma A.2, it is sufficient to show the existence of a run $\hat{R}$ such that $\hat{R}_b$ is not stable in the interval $I = (t, \Pi_c]$. There are three cases:

- **Case 1**: $t < ^1\Pi_b$. (Note that if $k = 2$, this case holds.) The required $\hat{R}$ exists by the no surplus pulses property of $\Phi$ (definition 4.4) and since $^1\Pi_b, \Pi_b \in I$. (Note that $^1\Pi_b \leq \Pi_b$ due to the trace definition.)

- **Case 2**: $^1\Pi_b \leq t < ^1\Pi_c$. Note that $t - 1'^T < P_D(T')$. Pick $\tau > t$ and close enough to $t$ such that $\tau - 1'^T < P_D(T')$ and $\tau \in \Pi_b$. From the induction assumption with $\langle T', \tau \rangle$ instead of $\langle T, t \rangle$ follows that exists a run $\hat{R}$ such that $\hat{R}_b$ is not stable in $[\tau, \Pi_c] \subseteq I$.

- **Case 3**: $^1\Pi_c \leq t$. This implies $\Pi'_c - 1'^T < P_D(P_s)$. From the induction assumption with $\langle T', \Pi'_c \rangle$ instead of $\langle T, t \rangle$ follows that there is a run $\hat{R}$ s.t. $\hat{R}_c(\Pi'_c) = \frac{1}{2}$. This contradicts our assumption that $\langle Y, \Phi \rangle$ is strictly functional.


9 Simulations

In order to validate our results we simulated some Domino Express systems. After the simulations were completed, the paper of Harris and Horowitz [19] was published. Surprisingly, our simulations were very similar to their ones.

The first simulation was of a synthetic system — a ring of identical gates. A similar ring with different gates was tested by Harris and Horowitz. This simulation shown the skew-tolerance and time-borrowing abilities of the Domino Express. Moreover, the self-borrowing was discovered with it.

To simulate a real system, we built a shifter stage of 300-Mhz Alpha pipeline [4] both in the two-phase domino technique and Domino Express. (Harris and Horowitz built the adder stage of the same processor.) This stage has to fulfill two tasks. The first is to decode the shift value from the binary to the unary presentation; this is the easy operation. The second is to carry out a shift and to drive the result on the bus; this operation takes much more time. When building in the two-phase domino technique, these operations are fulfilled in the different halves of the period. Since this clocking technique does not enable time-borrowing, this very disbalanced stage causes a loss of the performance. A Domino Express implementation gave us a speed-up of about 10%.
the two right columns we show the maximal skew-tolerance of OTB and Domino Express as the percentage of the optimal clock period.

<table>
<thead>
<tr>
<th>$C_D / P_D$</th>
<th>Skew-Tolerance of OTB</th>
<th>Skew-Tolerance of Express</th>
</tr>
</thead>
<tbody>
<tr>
<td>0%</td>
<td>0%</td>
<td>16.6%</td>
</tr>
<tr>
<td>25%</td>
<td>12.5%</td>
<td>20.8%</td>
</tr>
<tr>
<td>50%</td>
<td>25%</td>
<td>25%</td>
</tr>
<tr>
<td>75%</td>
<td>25%</td>
<td>29.2%</td>
</tr>
</tbody>
</table>

We assume that a $C_D / P_D$ ratio of 25% can be easily achieved. Only experience will reveal the practical limit of this ratio in large VLSI systems.
As the system nominal clock-set we use \( \hat{\Phi} \) which is defined from \( (\Phi^{\text{min}}, \Delta) \) the same way that \( \Phi \) of the previous section was defined from \( (\Phi^{\text{min}}, \Delta) \); i.e., we extend each pulse leftward by \( \hat{\Delta} \).

We show that \( \langle Y, \hat{\Phi} \rangle \) is \( \hat{\Delta} \)-skew-tolerant the same way as in the CD-free case. That is, we show that every run of \( \langle Y, \hat{\Phi} \rangle \) is also a run of the capricious nondet \( Y \) clocked by \( \hat{\Phi} \) — the very same pair used in Lemma 8.4.

Let \( \hat{\Phi'} \in B(\hat{\Phi}, \hat{\Delta}) \). See Figure 8.9. In contrast the CD-free case, there is an unsafe interval, denoted \( U \) in Figure 8.9, of length \( C_D(Y) \) such that all the gates (of all the sections) may be in the evaluate state during all of \( U \) under \( \hat{\Phi'} \). Nevertheless, all the arguments of the proof of Lemma 8.4 hold for \( \langle Y, \hat{\Phi'} \rangle \) except of one difficulty.

The strong version of Claim 4 does not hold; i.e., some edges of \( S_2 \) may strictly increase during \( C'_i \); this because \( C'_i \) does not cover now the uncertainty interval within \( C'_i \). However, an edge of \( X_3 \) (an output of \( S_2 \)) may strictly increase during \( C'_i \) only if a transition traverse all of \( S_1 \) during the unsafe interval \( U \). This is impossible since \( |U| = C_D(Y) \).

### 8.5 Disadvantages of the Domino Express

Here we present a number of relative disadvantages of Domino Express and explain why we believe that in spite of them using Domino Express is worthy.

Domino Express system has to satisfy two structural rules: the domino composition rule (section 7.1) and the demand on fences at the beginning of every section. These rules compel dual-railing more often than latch based clocking techniques. However, contemporary high-speed microprocessors heavily use dual-rail [4, 12] because of other reasons, e.g., noise-tolerance.

Another disadvantage of Domino Express is that it needs at least 3 clocks, while other domino clocking techniques have a single clock. However, the skew-tolerance, time-borrowing and latchlessness of Domino Express can cause a performance benefit in spite of that.

### 8.6 Comparison of Skew-Tolerance between OTB and Domino Express

Since to our opinion, skew-tolerance is the main advantage of Domino Express technique, we want to compare this aspect against the first domino skew-tolerant technique — OTB. In the following table we assume no time-borrowing in a system and zero propagation delays of fences. The expression in the left column is the ratio between the contamination and propagation delays of a section. In
of claim 2 follows from the hold-in-evaluate property of $g$, its zero delay and from our assumption that $e$ is monotonic decreasing during $C'$. 

**Claim 3** In the run $R$, any gate $g \in S_2 - F_2$ behave like a static gate having the same parameters (boolean function and delays) during $C_2$.

Let $g$ be such a gate. $g$ enters its evaluate state at some unknown time $t_1 \in C_2' - C_2$. By claim 1, the input signals of $g$ are “proper” during $[t_1, C_2]$. This implies claim 3.

**Claim 4** For any edge $e$ of $X_3$, $R_e$ is monotonic decreasing during $C_2'$.

We will actually prove this claim for any edge of $S_2$. Let $g$ be the source of such an edge $e$. We first consider the case where $g \in F_2$. Our gate exits its evaluate state at some unknown time $t_2 \in C_2' \cap C_2$. Claim 4 holds during $[t_1, t_2]$ because of our assumption on $S_1$ and the hold-in-evaluate property of $g$. Claim 4 holds in the rest of $C_2'$ since $g$ is in the preset state.

We now prove, by a contradiction, that claim 4 holds for $g \in S_2 - F_2$. Let $g$ be a “leftmost” gate that violates claim 4. All $g$ inputs are monotonic decreasing during $C_2'$, and $g$ has the hold-in-evaluate property. So, $g$ can violate claim 4 only if it was not firm at time $\tau = \downarrow C_2'$.

It follows that, up to time $\tau$, $R$ is a run of $\langle Y, \Phi \rangle$ and at time $\tau$, a gate is not firm while a “downstream” latch closes. By lemma 4.6 (that holds also for capricious systems) $\langle Y, \Phi \rangle$ is not functional. This contradicts lemma 8.3.

Let $\Phi' \in B(\Phi, \Delta)$. By lemma 8.4, $R(Y, \Phi') \subseteq R(Y, \Phi)$. By lemma 8.3, $\langle Y, \Phi \rangle$ is functional. Therefore, $\langle Y, \Phi' \rangle$ is functional. Since this holds for any $\Phi' \in B(\Phi, \Delta)$, $Y$ is $\Delta$-skew-tolerant.

### 8.4.1.2 The CD-based case

Now we show that $Y$ is also $\hat{\Delta}$-skew-tolerant for:

\[
\hat{\Delta} = \frac{P(Y)/3 + C_D(Y) - TB(Y)}{2}
\]  

(8.7)
Figure 8.7: (1) The optimal clock-set with minimal pulse-width; (2) The nominal clock-set; (3) The ball; (4) The clock-set of a capricious static system.

We establish 8.2 via an extension of the spotlight argument. That is, we show that if section \( S_i \) satisfies 8.2 during one of its spotlight intervals, then section \( S_{i+1} \) satisfies 8.2 during its next spotlight interval. Because of symmetry, it suffices to establish the above implication for \( i = 1 \). We prove that via a sequence of claims. In the proof we refer to figure 8.8.

**Claim 1** For any edge \( e \) of \( S_2 \): \( R_e(\Delta C_2') = 0 \) and \( R_e \) is monotonic increasing during \( C_2'' \).

Recall that all our gates compute monotonic increasing boolean functions and during the evaluation state their outputs are monotonic increasing. Hence, their outputs decrease only during preset.

Let the gate \( g \) be the source of such an edge \( e \). At time \( \uparrow C_2'' \), \( g \) has been “long enough” in its preset state and therefore has completed its preset transition; hence, \( R_e(\Delta C_2'') = 0 \). The gate \( g \) enters its evaluation state at some unknown time \( t_0 \in C_2'' - C_2 \). During \( [\uparrow C_2'', t_0] \), \( R_e \) can not decrease since it already reaches its minimum at \( t_0 \). In the rest of \( C_2'' \), \( g \) is in the evaluation state and its output is only increasing.

**Claim 2** In the run \( R \), any gate \( g \in F_2 \) behaves like an open latch during \( C_2 - C_1' \), and like a closed latch during \( C_2 \cap C_1' \).

(Note, however, that \( g \) may hold a non-logical value during \( C_2 \cap C_1' \).)

Let \( e \) be the input edge of such a gate \( g \). The first part of claim 2 follows from the fact that \( R_e \) is monotonic increasing during \( C_2'' - C_1' \); i.e., it is a “proper” input for our gate. The second part
This is achieved with the nominal clock-set with the pulse-width of:

\[ Pw = \frac{P(Y)}{3} + TB(Y) + \Delta + \varphi \]  

(8.5)

It comes out that in the case of \( \varphi = 0 \), \( TB(Y) = 0 \) and not using contamination delay, the maximal skew-tolerance is achieved by pulse-width of \( Pw = P(Y)/2 \).

8.4.1 “Formal” proof

8.4.1.1 The CD-free case

We firstly prove that \( Y \) is skew-tolerant, \textit{without use of contamination delay}, for clock skew of the magnitude:

\[ \Delta = \frac{P(Y)/3 - TB(Y)}{2} \]  

(8.6)

Let clock-set \( \Phi_{\text{min}} \) depicted in figure 8.7(1) be minimal for a Domino Express system \( Y \). As the system \textit{nominal} clock, we use \( \Phi \) depicted in Figure 8.7(2) and generated by extending every pulse of \( \Phi_{\text{min}} \) to the left by \( \Delta \). The radius \( \Delta \) ball of time-sets centered at \( \Phi \) is shown in figure 8.7(3). Note that under clock skew of magnitude \( \Delta \), three sections of the system never evaluate simultaneously. Moreover, the above property is tight – it does not hold for any larger clock skew. The macro-behavior of \( Y \) is defined in this case by the signals on the fences’ outputs at the ends of the nominal (rather than actual) pulses of the previous section; i.e., the values of \( F_i \) are checked at \( \{ II \mid \Pi \text{is a pulse of } \phi_{i-1} \} \). Figure 8.7(4) presents the clock-set \( \Phi \) of the capricious static system \( Y \), where \( \phi_i = \phi_i^{\text{min}} \wedge \phi_i^{\text{min}} \). Note that \( \Phi \) is optimal for \( Y \).

Lemma 8.3 \( \langle Y, \Phi \rangle \) is CD-mild.

Proof Sketch. It is given that \( \langle Y, \Phi \rangle \) is CD-mild. From this one may prove that all the inequalities of \( T I^Y \) hold. (This is the hard part.) Let \( \tilde{Y} \) be a non-capricious variant of \( Y \). By lemma 4.13, \( \langle \tilde{Y}, \Phi \rangle \) is CD-mild. By lemma 4.15, \( \langle Y, \Phi \rangle \) is CD-mild. \( \square \)

Lemma 8.4 Let \( \Phi' \in B(\Phi, \Delta) \) and \( R \) be a run of \( \langle Y, \Phi' \rangle \). Then \( R \) is a run of \( \langle Y, \Phi \rangle \).

Note that \( R \) is not assumed to be a “desired” run of \( \langle Y, \Phi' \rangle \). In fact, we will use this lemma to translate failures from \( \langle Y, \Phi' \rangle \) to \( \langle Y, \Phi \rangle \).

Proof Sketch. We define the \textit{edges of a section} \( S_i \) to be the edges exiting a node of \( S_i \). Consider, for example, section \( S_2 \) and its spotlight interval \( C_2 \) depicted in figure 8.8 (as usual, there is an infinite sequence of such intervals). Outside of \( C_2 \), all the edges of \( S_2 \) are abandoned in \( \langle Y, \Phi \rangle \). Hence, on these edges outside of \( C_2 \), \( R \) is a “legitimate” run of \( \langle Y, \Phi \rangle \). So, we need only to show:

\textbf{Statement 8.1} \( \forall i: R \) is a run of \( \langle Y, \Phi \rangle \) on \( S_i \) edges during \( C_i \).

We actually prove a stronger statement:

\textbf{Statement 8.2} Statement 8.1 holds and under \( R \) all the \( X_{i+1} \) edges (the outputs of \( C_i \)) are monotonic increasing during \( C_i \), and monotonic decreasing during \( C'_i \).
Note that the less the propagation delay of the fences $\phi$, the higher clock skew can be endured. Note as well, that the system do not need contamination delay in order to endure clock skew of almost $17\%$ of clock period. In the next section we show that exploiting contamination delay adds to the skew-tolerance ability.

8.4 Skew-Tolerance and Time-Borrowing of Domino Express

In this section we assume for simplicity that each of the gates of the fences has a single input and its propagation delay is zero ($\phi = 0$). This resembles our model of latches (section 4.2.1) used in nondets.

Assume a Domino Express system $Y$ and the clock-set $\Phi$ depicted in figure 8.6. Assume also that there is no clock skew in the system.

![Figure 8.6: Generating the static capricious 3-phase system $Y$ and the clock-set $\Phi$ from the Domino Express system $Y$' and clock-set $\Phi$.](image)

Look at section $S_i$. Its fence is transparent, when $\phi_i = 1 \wedge \phi_{i-1} = 1$. When $\phi_{i-1} = 0$, $F_i$ is in hold-in-evaluate, i.e., is opaque. When $\phi = 0$, the whole $S_i$ is in preset, but during this interval $S_{i+1}$ is either in hold-in-evaluate or in preset itself; i.e., does not depend on the outputs of $S_i$.

This observation leads us to the definition of the following static capricious 3-phase system $Y$ and a clock-set $\Phi$ (figure 8.6). Each dynamic gate $g$ is replaced by a static gate with the same $f_g$, $P_D(g)$ and $C_D(g)$. $\Phi$ has the clocks $\phi_1$, $\phi_2$ and $\phi_3$, defined by $\phi_i = \phi_i \wedge \phi_{i-1}$. Each $F_i$ is replaced by a latch fence controlled by a clock $\phi_i$.

For the following discussion we need to define the macro-behavior of a Domino Express system. We define it by the values on the fences’ outputs at the ends of the pulses of the nominal clocks of the previous section — i.e., values of the fence $F_i$ are checked when $\phi_{i-1}$ falls.

We assume that $(Y, \Phi)$ is CD-mild; i.e., it uses the contamination delay of $Y$ only to endure clock skew. We’re going to prove in the following section 8.4.1 that $Y$ is $\Delta$-skew-tolerant for:

$$\Delta = \frac{P(Y)/3 + C_D(Y) - TB(Y) - \varphi}{2}$$

(8.4)
We show here by the spotlight argument that the system endures this skew, even if its contamination delay is 0. (The system may endure a clock skew of a greater magnitude without use of contamination delay, via another nominal clock-set. For simplicity we keep the same one, as in the previous section.)

Figure 8.4 presents the ball of radius $\Delta$ centered at $\Phi$ and the possible spotlight intervals. Each $C_j$ is divided into three subsequent subintervals $C^i_j$, $C^s_j$ and $C^d_j$ such that $|C^i_j| = \Delta$ and $|C^d_j| = \varphi$.

![Figure 8.4: The ball and spotlight intervals.](image)

**Definition 8.2** An edge is **semi-stable** in $C_j$ if it is monotonic increasing in $C^i_j$, stable during $C^s_j$ and monotonic decreasing during $C^d_j$.

**Lemma 8.2** Consider $S_1$ in isolation (figure 8.5). Let $X_1$ be semi-stable during a $C_1$ interval. Then $X_2$ is semi-stable and correct during the next $C_2$ interval.

![Figure 8.5: Proof of the one-step property of $S_1$.](image)

**Proof.**

Let $\Phi$ be an actual time-set and $I$ the interval where all the actual timing signals derived from $\phi_1$ equal 1. Since $X_1$ is monotonic increasing during $C^i_1$ and stable during $C^s_1$, at the beginning of $C^d_1$ the outputs of $F_1$ are correct. Since $X_1$ is monotonic decreasing during $C^d_1 \cap I$, $F_1$ is in hold-in-evaluate (section 7.1); i.e., its outputs are stable. Since the whole $S_1$ is in evaluation during $I$, its outputs are monotonic increasing, particularly during the next $C^i_2$ interval.

Subinterval $C^d_2$ is included in $I$ because $\Delta \leq P/6 - \varphi$; hence, since $P_D(R_1) \leq P/3 - \varphi$, the outputs of $S_1$ are correct and stable during $C^d_2$ (i.e., at least $\varphi$ time units). In the interval $C^d_1 \cap C^d_2$ the outputs of $S_1$ may remain stable or decrease due to preset. (Note that the hold-in-evaluate property of the gates whose outputs are the outputs of $S_1$ is used here.) In the interval $C^d_2 \setminus C_1$, $S_1$ is in preset state. As the result its outputs are monotonic decreasing. \qed
Definition 8.1 An edge is semi-stable during $C_i$ if it is stable during $C_i^s$ and monotonic decreasing during $C_i^d$.

Lemma 8.1 Consider $S_1$ in isolation (figure 8.3). Let $X_1$ be semi-stable during a $C_1$ interval. Then $X_2$ is semi-stable and correct during the next $C_2$ interval.

Proof. The behavior of $X_1$ during $C_1$ and the hold-in-evaluate property of $F_1$ presented in section 7.1 ensure that the outputs of $F_1$ are correct and stable during $C_1^d$. Since $P_D(R_1) \leq C_1^d \cap C_2$, the outputs of $S_1$ are correct and stable during $C_1 \cap C_2$ (i.e., at least $\varphi$ time units). In the interval $C_2 \setminus C_1$, $S_1$ is in the preset state; hence, its outputs are monotonic decreasing.

Figure 8.3: Proof of the one-step property of $S_1$.

8.3 Skew-Tolerance of Domino Express

Assume the system with the clock-set $\Phi$ of naive clock period and pulse-width $P_w = P/2$, as presented in the previous section. Assume there is a clock skew in the system of magnitude

$$\Delta = \frac{P}{6} - \varphi$$  \hspace{1cm} (8.3)

\hspace{1cm} i.e., the stable values are correct.
8 Domino Express

This section presents a new domino clocking technique, called *Domino Express*. The same technique was found earlier and independently by Harris and Horowitz [19]. This technique is latchless, enable time-borrowing and is skew-tolerant. As said, most contemporary dynamic clocking techniques do not have this property.

8.1 Three-sections structure

A Domino Express Logic system is built of three sections, \( S_1, S_2, S_3 \), as shown in figure 8.1.

![Figure 8.1: Three-sections structure.](image)

All sections are built of N-Domino logic. (Actually, any known domino logic can be used; the only requirement is that it has the hold-in-evaluate property. We use N-Domino here for the sake of simplicity). Henceforth, a *gate* means a standard N-gate of figure 7.1(1) followed by a static inverter. Note that, with these gates, there is no restriction on their composition.

\( S_i \) has a *fence* \( F_i \) (depicted as a segment at the left part of a section). \( F_i \) is an N-Domino logic such that each path from an input to an output in \( F_i \) has exactly one gate. The *rear* of \( S_i \) standing behind \( F_i \), denoted \( R_i \), can have gate-less paths. For simplicity we assume that all gates in \( F_i \) have the same small propagation delay, denoted by \( \varphi \); i.e., \( P_D(S_i) = \varphi + P_D(R_i) \).

Every section \( S_i \) is controlled by its own clock \( \phi_i \). The clocks are symmetric. In section 8.2 we show that, assuming no clock skew, the system is strictly functional with the naive clock period:

\[
P = 3 \cdot \max\{P_D(S_1), P_D(S_2), P_D(S_3)\}
\]

and pulse-width:

\[
\frac{P}{3} + \varphi \leq P_w \leq 2 \cdot \frac{P}{3}.
\]

I.e., the overlapping of pulses are at least \( \varphi \). (Such period is naive, since Domino Express enables time-borrowing (section 8.4))

8.2 Proof of the One-Step Property in the Non-Skew Case

Figure 8.2 presents a possible clock-set with naive clock period and pulse-width \( P_w = P/2 \). (We assume that \( \varphi \leq P/6 \).) It also shows possible spotlight intervals \( C_1, C_2, \) and \( C_3 \). Each \( C_i \) is divided into two subintervals \( C_i^d \) followed by \( C_i^a \) such that \( |C_i^a| = \varphi \).
Figure 7.1: Standard domino gates. (1) N-gate; (2) P-gate.

those with $\psi = \varphi$. Generally, the geometric sizes of a gate are small, hence, the “internal” clock skew is negligible. However, this problem is keen in a precharge bus. Such a bus is similar to a domino gate in its mode of operation and in its structure. Because of its large size, the “internal” clock skew may be significant. For example, in the 300-MHz Alpha microprocessor [4], the preset phase of a core bus is delayed in order to avoid such a problem.

Non-standard domino gates can be found in Weste and Esghragnian’s type B technique [38], the HS-PDCMOS technique [41] and All-N-Logic [18].

### 7.3 Specific Domino Logic Clock Skew Problems

In section 2.2 we presented the basic forms of failures in static logic systems. Those failures were associated with latches. Thus, in a domino system that use latches all those failures may occur. Besides that, in a domino system there are eight specific basic forms of failures caused by clock skew. All are caused by a single distortion at a single domino gate $g$; had this distortion not occurred the system would have worked correctly.

The $E$ and $P$ expressions give rise to two time intervals. Any distortion, to the left or to the right, of any of the endpoints of these intervals (eight possibilities at all) may cause a failure. For instance, assume $E$ widens. During this interval $g$ is sensitive to its inputs, hence it may perform an erroneous output transition.

Another failure example concerns clocking techniques of Mode 1 (section 7.1). Assume that the left endpoint of $P$ moves to the left. In this case, the preset of $g$ initiates earlier. This may cause a double clocking failure in a succeeding latch. Recall that if a clocking technique is CD-based, it endures double clocking by contamination delay. In such systems, if the described distortion occurs in a last domino gate before a latch fence, one has either to intentionally insert static logic between them or to increase the contamination delay of the latches. That is, the problem cannot be solved by increasing the contamination delay of an existing dynamic logic.
not hard. In the preset stage, all the gates work in parallel\(^1\), in contrast to the evaluate stage where the gates work somewhat in serial. Moreover, the duration of the preset stage can (usually) be made excessive without reducing the performance.

Clocking techniques can be distinguished concerning the issue of when the outputs of the domino logic have to be correct and stable. Denote this interval by \(C\).

- **Mode 1.** The upper ends of \(C\) and \(E\) coincide. This is the case, for example, in the tight version of the two-phase domino, NORA, TSPC and PHIMOS.

- **Mode 2.** \(C\) ends inside \(E\). This is the case, for example, in Weste and Eshraghian’s types A and B techniques.

- **Mode 3.** \(C\) begins at the end of \(E\) and ends at the beginning of the next \(P\); i.e., the hold state is used. This is the case, for example, in the non-overlapping version of the two-phase domino.

There is another mode of domino logic which is different from the above. In this mode, we need an additional property of domino gates, called **hold-in-evaluate**. Assume a gate \(g\) is firm at \(t\), and later some of its inputs change in the direction opposite to their modes, all this during the evaluation state. The gate has the hold-in-evaluate property if the output of \(g\) remains stable during this scenario. Note that the basic property of domino gates ensures the hold-in-evaluate one in the case when \(g\) did a logical transition before time \(t\).

All the known implementations of domino gates have the hold-in-evaluate property. Clocking techniques using it are OTB [20] and Domino Express (section 8). This property is also exploited (via an ad-hoc clock) in the 300-MHz Alpha [4].

### 7.2 Standard Domino Gate

The basic advantage of the domino logic over the static one is the significant reduction in the load capacitance of the gates inputs which leads to improved performance. This made dynamic logic wide-spread in high-speed microprocessors. However, domino gates have electrical disadvantages of charge sharing problems and reduced noise margins that are out of the scope of this study.

The standard implementation of domino gates\(^1\) are shown in figure 7.1. The letters ‘N’ or ‘P’ denote the type of transistors used in the indicated networks. Both gates depicted have \(E = \psi\) and \(P = \neg \psi\). The N-gate (figure 7.1(1)) has a “down” output and “up” inputs. The P-gate (figure 7.1(1)) has an “up” output and “down” inputs. Domino logic built of N-gates and P-gates is called \(NP\)-domino, while this using only N-gates is called \(N\)-domino [38]. (In the latter case, there is a static inverter between any two domino gates.) The noise margin of NP-Domino is significantly lower than that of N-Domino and moreover, it needs more clock signals. Therefore, it is rarely used by the industry [28].

Standard gates may have an electrical problem caused by clock skew. If the actual \(E\) and \(P\) expressions of a gate are not disjoint, then when \(E \land P = 1\), there is an electrical current in the circuit. (The same problem is caused by a slow transition of the clock signal [27].) This causes no functional fault, but leads to a waste of power and excessive heat. The most popular gates are

---

\(^1\)An exception to this parallelism will be discussed shortly.

\(^1\)In order to reduce the gate delay, transistor \(n_1 (p_2)\) can be omitted. This is possible in inner gates only. When a gate is in the preset state, the absence of a current is ensured by the preset outputs of the immediately preceding gates; i.e., the preset is done in serial. A long path of such gates may cause a long preset time accompanied with a high power consumption.
7 Domino Logic

7.1 Domino Gate

A *domino gate* $g$ is controlled by clocks; it is associated with two boolean expressions of its clocks, $E$ and $P$, that determine its state to be one of **preset, evaluate or hold**. In the preset state (when $P = 1$), the output of $g$ is preset to a predetermined value. In the hold state (when $E = P = 0$), the output is constant. The behavior of the gate in the evaluate state (when $E = 1$) is much more subtle. In the following discussion we consider a data-signal as a continuous real function whose range is the real interval $[0, 1]$. A domino gate $g$ is associated with a boolean function $f_g$, and its main task is to compute $f_g$ and transmit it on its output.

The **Basic Domino Property** of a domino gate is that during the evaluate state the output signal is monotonic in the direction from the preset value to the other logical value; i.e., a gate can not undo an output transition. This property gives rise to the Domino Metaphor. During the evaluate state, a gate may fall (perform a complete, logical transition) but it can not rise. Moreover, a fall of a gate may cause the fall of downstream gates, as will become clear shortly.

Because of the Basic Domino Property, a gate is not combinational. It is rather similar to an asynchronous FSM. Since we want it to behave in a combinational fashion, the behavior of its inputs must be restricted.

A *domino logic* is built of domino gates; the $E$ expressions of all of them are equivalent under the nominal clock-set, and the same with the $P$'s. The domino logic should behave like a combinational logic during the $E$ interval; i.e., all the gates should be consistent at the end of the interval. This should hold at least under the assumption that the inputs of the logic are stable during this interval. For the time being, assume that this interval is long enough and there is no clock skew.

Since an input of a domino gate usually come from another domino gate, it is monotonic during the $E$ interval and its direction, denoted "up" or "down", is known. We refer to this direction as the **mode** of the edge. The final output of a gate $g$ must be independent of the order of its inputs transitions (the **Order-Independent** property). This implies that $f_g(x_1, \ldots, x_k)$ is quasi-monotonic; i.e., for each $x_i$, $f_g$ is either monotonic increasing or monotonic decreasing in $x_i$. It follows that the mode of each input edge is an inherent property of the gate; the same input edge can be of both modes (in two instantiations) only in the obscure case where $f_g(x_1, \ldots, x_k)$ does not depend on the corresponding variable. (We henceforth ignore this case.) When two gates are connected, the output mode has to match the input mode. This constitutes the domino composition rule. If the mode does not match, a static inverter can be inserted between these gates.

The inputs of a domino logic are the inputs of domino gates; hence, they may be monotonic (rather than stable as we have assumed) during $E$. Clearly, their permissible directions have to satisfy the domino composition rule.

The above arguments concerning domino gates are also valid for a domino logic. Hence, such a logic can compute only quasi-monotonic functions; e.g., it can not compute XOR. In order to overcome this limitation, one has to use dual-rail.

Static logic can be inserted within a domino one in a very restricted way, and such an insertion does not alleviate the computational limitation above. Hence, the common practice is to insert only static inverters for mode matching, as mention above.

Under clock skew, the gates do not enter the evaluate state simultaneously. This is not a problem thank to the Order-Independent property. However, it is vital that a gate does not start evaluating before all the domino gates immediately preceding it have completed their preset transition. We call this requirement the **Preset Condition**. Maintaining this condition, under a clock skew, is
Despite that the skew-tolerance of static logic was not known, the industry enjoyed it nevertheless. It produced more working high-speed chips than expected.

### 6.4 Self-Borrowing

Assume a static two-phase system (figure 6.2) that performs each cycle either an addition or multiplication. Moreover, assume that these operations are carried out in alternating order: a sum computation follows a product computation and vice versa. It’s known that the propagation delays of logics $CL_1$ and $CL_2$ in a sum computation are 1, and in a product computation — 3. The naive clock period of such a system is 6. However, clearly, it is functional with 4. This property of static systems we call **self-borrowing**.

More formally, let $Y$ be a static two-phase system and $\Phi$ the tight clocks clock-set with a period $P$. Assume no clock skew in the system. Define a sequence of computations (of boolean functions) $\langle X_i, i \geq 0 \rangle$ such that $X_i$ is a computation of $CL_{i(mod\ 2)}$. Define a sequence of durations $\langle D_i, i \geq 0 \rangle$ be such that $D_i$ is the time needed for $CL_{i(mod\ 2)} \ \text{1}$ to complete the computation $X_i$.

**Lemma 6.2** A pair $\langle Y, \Phi \rangle$ is functional if for every $j, k \geq 0$ holds:

$$\sum_{i=0}^{j} D_{j+i} \leq (k + 1) \cdot \frac{P}{2} \quad (6.2)$$

Note that in contrast to all the analyses we have done before, this one is based on our knowledge of the system functionality and data. Hence, exploiting self-borrowing is a hard task which should be considered on the architecture level of design. Note that our NONDET model is not capable to analyze such systems.

Self-borrowing can be used in systems which sometimes work “hard”, meaning that data goes through critical paths, and sometimes work “easy”. Such a system “saves” time during the “easy” steps to be used for the “hard” ones. Such “saving” can increase the system performance. For the sake of illustration, assume the following processor. It executes each instruction in a single cycle; it has more and less difficult instructions; each is associated with a time which takes to execute it. Without self-borrowing, the longest instruction defines the clock period. However, suppose that to work correctly, it is sufficient that every $N$ consequent instructions will need not more than $Q$ accumulative time. A clever compiler can take advantage of this property and schedule the machine instructions accordingly.

We believe that self-borrowing is an interesting and promising topic for a future research.

---

1We assume that $CL_{i(mod\ 2)}$ receives all its inputs simultaneously.
Proof. Consider the minimal clock-set $\Phi^{min}$ with the pulse-width $TB(Y)$ depicted in figure 6.1(1). Figure 6.1(2) presents $\Phi$ — the expansion of $\Phi^{min}$ such that the rise of every pulse of every clock is made earlier by $\Delta$. We use $\Phi$ as the nominal clock-set. Figure 6.1(3) shows $B(\Phi, \Delta)$. Let $\Phi' \in B(\Phi, \Delta)$. We need to show that $\Phi' \simeq_{[Y]} \Phi$. Without loss of generality, any timing signal of $\Phi'$ has exactly one transition in each of its uncertainty intervals. Clearly, $\Phi'$ is a pulse-expansion of $\Phi^{min}$, but not necessarily a modest one. However, from the definition of $\Delta$ follows that there is a time-set $\Phi'' \in B(\Phi, \Delta)$, which is a modest pulse-expansion of $\Phi^{min}$ and $\Phi'$ is a $CD(Y)$-right-expansion of $\Phi''$. Since $\Phi^{min}$ is optimal for $Y$, due to definition 6.1 ($Y, \Phi^{min}$) is CD-mild; hence, we can use the CD=0 Pulse-Expansion Theorem 4.2, which implies $\Phi'' \simeq_{[Y]} \Phi^{min}$. By the CD Pulse-Expansion Theorem 4.1, $\Phi' \simeq_{[Y]} \Phi^{min}$. Hence, $Y$ is $\Delta$-skew-tolerant. 

\[ \Phi^{min} \]
\[ \Phi \]
\[ B(\Phi^{min}, \Delta) \]

Figure 6.1: The pulse-expansion steps.

Note that non-overlapping clocks enable a greater skew-tolerance than tight clocks. So, having tight nominal clocks reduces the number of clocks but also reduces the system ability to endure clock skew.

6.3 Conclusions

Note that even a CD=0 system is skew-tolerant. Also note that the greatest amount of clock skew is endured by systems with a zero time-borrowing. Such a system easily (CD-free) endure clock skew of quarter a period:

\[ \Delta = \frac{P}{4}. \]

As a conclusion, it is worth to make an effort to balance the propagation delays of the logics between the latches; i.e., to get a system without time-borrowing. Such a balancing can be achieved via retiming [29].
6 Time-Borrowing and Skew-Tolerance of Static Systems

In this section we prove that the well-known static N-phase clocking technique, described in section 2.3.1, is skew-tolerant. We also show that in these systems time-borrowing and skew-tolerance are strongly related and “compete” on the same resources.

6.1 Time-Borrowing Definition

In the section 3 we gave a number of examples presenting the time-borrowing concept. Here we give the formal definition.

Definition 6.1 Let Y be a static N-phase system. A clock-set \( \Phi \) is optimal for Y if:

- it obeys the clocking technique; in particular, \( \Phi \) is symmetric;
- \( \langle Y, \Phi \rangle \) is CD-mild (and therefore also strictly functional);
- its period is minimal among those satisfying the above conditions.

Definition 6.2 Let Y be a static N-phase system. The minimal clock-set of Y, denoted \( \Phi^{\text{min}} \), is the optimal clock-set of Y whose pulse-width is minimal.

Let Y be a static N-phase system and denote the period of its optimal clock-set by \( P(Y) \). Any simple path \( H \) in Y is a zero path. (Note that a cycle cannot be a hog). Let the number of its internal latches be \( L \). Then the “fair share” of \( H \) is \( \mathcal{FS}(H) = (L + 1) \cdot (P(Y)/N) \). Clearly, \( H \) “consumes” its propagation delay – \( P_D(H) \).

Definition 6.3 The magnitude of time-borrowing in Y is:

\[
TB(Y) = \max\{P_D(H) - \mathcal{FS}(H) \mid H \text{ is a simple path in } Y\}
\]

Note that \( TB \) cannot be negative, since \( P(Y) \) is the period of an optimal clock-set. This and the next lemma follow from the Trace Inequalities Lemma (4.13).

Lemma 6.1 \( TB(Y) \) is the pulse-width of the minimal clock-set of Y.

6.2 Skew-Tolerance of the Static Two-Phase Systems

Let Y be a static two-phase system and \( \Phi \) its clock-set. Clearly, \( \Phi \) has the unambiguous bit and no transparent cycle properties. This system also satisfies the no surplus pulses property in a strong sense. That is, there is a run of a similar system \( Y' \) such that at every pulse every latch will change its value. For example, such a \( Y' \) has one inverting and one non-inverting latch fences; all the gates are OR-gates; in the initial state, all signals are 0.

Recall that \( C_D(Y) \) is the minimal contamination delay of a two-latches path in Y.

Theorem 6.1 Let Y be a static two-phase system, \( C_D(Y) \leq P(Y)/2 - TB(Y) \) and

\[
\Delta = \frac{P(Y)/2 + C_D(Y) - TB(Y)}{2}.
\]

Then Y is \( \Delta \)-skew-tolerant.
Figure 5.1: (1) Nominal clock-set $\Phi (x, y \in Z)$; (2) presentation of $A(\Phi, \Delta)$ and $B(\Phi, \Delta)$; (3) $\Phi'$ is a member of $A(\Phi, \Delta)$; (4) $\Phi''$ is a uniform translation of $\Phi'$.

5.2.1 An Engineering Dilemma in VLSI

Imagine one knows at the design stage the actual time-set, and this time-set is irregular. Should he take advantage of this knowledge and complicate the design or assume a regular clock-set that approximates the actual one? The first approach is addressed in [8, 34]. The second one creates an artificial clock skew.

5.3 Quantifying the Skew-Tolerance

**Definition 5.2** Let $Y$ be a system obeying some clocking technique and $\Delta > 0$. $Y$ is $\Delta$-skew-tolerant if there is a clock-set $\Phi$ with minimal period obeying the clocking technique such that $\Phi' \equiv [Y] \Phi$ for every $\Phi' \in B(\Phi, \Delta)$. 
Note that by this definition a uniform translation is a zero clock skew; this is the right definition for systems having no I/O.

Our definition of the magnitude of the clock skew agrees with most of the literature and is equivalent to the definition given (in a different form) in [5].

Clearly, all time-sets \{\Phi\} such that their clock skews are at most \(\Delta\) constitute the ball \(B(\Phi, \Delta)\). One may assume that, in the case of a static system where all terminals are latches, only the clock skew between two adjacent latches is relevant; hence the \(d\) in definition 5.1 should be replaced by \(d'(\Phi, \Phi') = \max_{x \in \mathbb{Z}} \{ d_{\text{in}, z}(\Phi, \Phi') \}\). This assumption is wrong, as demonstrated by the Trace Inequalities Lemma (4.13).

Clearly, the clock skew can be quantified in a more detailed form; e.g., by the \(d_{\text{in}, z}\) function.

However, in this study we quantify the clock skew by only a single number. This obviously hides an information which probably could be used to advantage. This issue calls for further research.

5.1 Graphical Representation of Clock Skew

To depict a clock skew of magnitude \(\Delta\) (relative to the nominal clock-set \(\Phi\)) we actually need to depict the set of all possible time-sets under this skew; i.e., \(B(\Phi, \Delta)\). Figure 5.1(2) is such a representation. This representation is subtle and should be understood as follows.

Consider the drawing denoted \(A_x\) in figure 5.1(2) (\(x\) is a member of \(\mathbb{Z}\)). This drawing represents a set \(A_x\) of timing signals as follows: \(f : \mathbb{R} \rightarrow \{0, 1\}\) is in \(A_x\) iff \(f\) agrees with the depicted function on the “white” intervals, and in each “gray” interval \(f\) has exactly one transition. The gray intervals are called uncertainty intervals.

In this drawing, an uncertainty interval of length \(\Delta\) is attached to each transition of each \(\Phi_z\), and the interval is left-aligned with the transition. This gives rise to the sets \(A_z, z \in \mathbb{Z}\).

Figure 5.1(2) can be understood in two ways. First, as a representation of the set \(A(\Phi, \Delta) = \prod_{z \in \mathbb{Z}} A_z\). Under this interpretation, it represents any time-set \(\Phi'\) such that for each \(z\), \(\Phi'_z\) has exactly one transition in each uncertainty interval, and otherwise \(\Phi'_z\) agrees with \(\Phi_z\). (Note that \(A(\Phi, \Delta) \subseteq B(\Phi, \Delta)\).)

Under the second (and correct) interpretation, figure 5.1(2) depicts \(B(\Phi, \Delta)\). That is, the set of all time-sets \(\Phi'\) such that \(\Phi'\) is a uniform translation of a member of \(A(\Phi, \Delta)\).

Because of this last closure, the uncertainty intervals do not need to be left-aligned. Each of them must include its transition and all should be aligned the same way; i.e., the distance from the start of the interval to its transition has to be uniform.

Note that this representation method can not depict a skew that is equal or greater than the length of pulses and negative pulses. We are never interested in such a skew since no system can endure it.

In the following discussions, whenever we consider a member \(\Phi'\) of \(B(\Phi, \Delta)\), we also assume that \(\Phi' \in A(\Phi, \Delta)\). This assumption is permitted since our systems have no I/O and therefore their behavior is preserved under a uniform time translation.

5.2 Clock Skew as Uncertainty and Artificial Clock Skew

Like most of the authors, we consider clock skew as an uncertainty at the design stage. It is a challenge that the designers should overcome in order to produce a reliable system. In this sense, clock skew is, of course, never useful. Some authors [21] consider clock skew as a known deviation from a non-optimal clock-set; this skew clearly may be useful.
5 Quantifying the Clock Skew

This section presents the quantifying of the clock skew.

Let \( Z \) be the set of timing terminals of a data sub-system; i.e., the terminals through which it receives the clocking signals. (In practice, each terminal is a transistor gate). We reuse the term time-set of definition 4.4 to denote an association of a timing signal to each member of \( Z \); however, there are no restrictions of continuity from the left and initialization on the timing signals. Note that a nominal clock-set (of some clocking technique) is a special case of a time-set.

We model the clock skew phenomenon by a time-distortion \( \delta \), an association of a strictly monotonic increasing bijection from \( \mathbb{R} \) to \( \mathbb{R} \), denoted \( \delta_z \), for each terminal \( z \).

Given a time-set \( \Phi \) and a time-distortion \( \delta \), we say that \( \delta \) distorts \( \Phi \) into the time-set \( \delta(\Phi) \), defined by:

\[
\delta(\Phi)(t) = \Phi_z(\delta_z(t))
\]

The norm of \( \delta \) is defined by:

\[
||\delta|| = \sup \{ |(\delta_z(t_1) - \delta_z(t_2)) - (t_1 - t_2)| : z_1, z_2 \in Z, t_1, t_2 \in \mathbb{R} \}
\]

Note that \( ||\delta|| = 0 \) iff \( \delta \) is a uniform translation; i.e., \( \delta_z(t) = t + c \) for some constant \( c \) and all \( z \) and \( t \). This express our understanding that a uniform translation is actually not a distortion.

Given two time-sets \( \Phi \) and \( \Phi' \) for \( Z \), define the distance between them by:

\[
d(\Phi, \Phi') = \inf \{ ||\delta|| : \Phi' = \delta(\Phi) \}
\]

(If the above set is empty, define \( d(\Phi, \Phi') = \infty \).)

Note that \( d(\Phi, \Phi') = d(\Phi', \Phi) \) and \( d \) satisfies the triangle inequality\(^1\). Henceforth, let \( \Phi' = \delta(\Phi) \), where \( ||\delta|| \) is minimal. The following equation is an alternative definition of the distance.

\[
d(\Phi, \Phi') = \sup \{ |(\delta_z'(t_1) - \delta_z'(t_2)) - (t_1 - t_2)| : z_1, z_2 \in Z, t_i \text{ is a transition of } \Phi_z' \}
\]

Note that \( d(\Phi, \Phi') = 0 \) iff \( \Phi' \) is a uniform translation of \( \Phi \); i.e., \( \Phi'_z(t) = \Phi_z(t + c) \) for some constant \( c \) and all \( z \) and \( t \).

Define the closed ball of radius \( r \) centered at \( \Phi \) by:

\[
B(\Phi, r) = \{ \Phi' : d(\Phi, \Phi') \leq r \}
\]

Note that even if \( \Phi \) is a clock-set, \( B \) includes time-sets which are not clock-sets.

Define the distance between \( \Phi \) and \( \Phi' \) relative to the terminals \( z_1 \) and \( z_2 \) by:

\[
d^{z_1, z_2}(\Phi, \Phi') = \sup \{ |(\delta_z'(t_1) - \delta_z'(t_2)) - (t_1 - t_2)| : t_i \text{ is a transition of } \Phi_z' \}
\]

Note that:

\[
d(\Phi, \Phi') = \max_{z_1, z_2 \in Z} \{ d^{z_1, z_2}(\Phi, \Phi') \}
\]

A distortion of a nominal time-set results in an actual time-set \( \Phi' \).

**Definition 5.1** Let \( \Phi \) and \( \Phi' \) be nominal and actual time-sets for \( Z \). Define the clock skew of \( \Phi' \) (relative to \( \Phi \)) by \( \delta = d(\Phi, \Phi') \).

\(^1\) Actually, \( \{\Phi, d\} \) is almost a metric space; the only violation is that \( d(\Phi, \Phi') \) may be 0 or \( \infty \) for \( \Phi \neq \Phi' \).
4.12 The Static Model and Real Latches

Real latches are, of course, more complex than the latch of our model. Here we present the subtle properties of real latches and show how our model can be adjusted so that our main theorems hold.

Real latches are usually controlled by several clocks like the simple latch presented in section 2.1. Even if a latch is governed by a single nominal clock, this clock control several transistors, as in the TSPC latches presented in section 2.3.3. Since we model actual signals rather than nominal ones, our latch should have several timing signals even in this case.

Without loss of generality, we may assume that the real latch in question is transparent when all its timing signals are 1 and opaque when all are 0; its behavior during other states (determined by these signals and called intermediate states) is idiosyncratic.

Our ability to handle this latch hinges on the following question. Assume that the input of the real latch is stable during a transit from the transparent state to the opaque one (possibly via some intermediate states). Is the output hazard-free (i.e., stable) during this transition?

If the answer is negative we have two serious difficulties. Firstly, the Trace Inequalities Lemma (4.13) no longer holds. Secondly, a system based on these latches is not skew-tolerant. Hence, we have no interest in such latches.

If the answer is positive, as is the case with all known latches, we can extend our model to fit the real latches without disturbing our main theorems.

A related issue concerns the output behavior of a real latch when the input is stable and equals the initial output value during a transit from the opaque to the transparent state. By definition, the original latches of our model are hazard-free in this scenario; however, we never use this property in our proofs and our theorems hold without it. Note that TSPC latches depicted in figure 2.11 actually have this hazard even under an ideal clock.

Another deviation of our model from real systems is that the timing signals of the latter have non-instantaneous transitions. We can model such a signal by two of our signals; one will mark the beginning of a transition and the other its end.
Proof of the CD=0 pulse-expansion theorem. Since \( \langle Y, \Phi \rangle \) is CD-mild, \( \langle Y^{CD=0}, \Phi' \rangle \) is strictly functional. Hence, according to the Trace Inequalities Lemma (4.13), all the inequalities of \( T Y^{CE=0, \Phi} \) hold. By lemma 4.14, all the inequalities of \( T Y^{CE=0, \Phi} \) hold. Using lemma 4.13 in the other direction, we conclude that \( \langle Y^{CD=0}, \Phi' \rangle \) is strictly functional and since \( Y \leq Y^{CD=0}, \langle Y, \Phi' \rangle \) is strictly functional.

Since \( \Phi \leq_m \Phi' \), lemma 4.12 implies \( \Phi \equiv Y^{CE=0} \Phi' \). Thus, according to lemma 4.11, \( \Phi \equiv Y \Phi' \).

Note that in the above proof we need \( \leq_m \) rather than \( \leq \) to apply lemmas 4.14 and 4.12. The theorem itself does not hold for \( \leq \) instead of \( \leq_m \).

4.11 A Capricious Nondet

In this section we introduce a variant of our model that is much more non-deterministic than the regular one. Hence, by our definition of ‘is functional’, the new model is much more failure prone. Our motivation to study this model concerns dynamic VLSI systems. As will be discussed in section 7, the behavior and failures of dynamic systems are idiosyncratic and we do not wish to study them directly. Instead, we will map a (real) dynamic system \( Y \) into a capricious nondet \( Y' \) in such a way that \( Y' \) is even more failure prone than \( Y \). Then the functionality of \( Y \) will follow from the functionality of \( Y' \).

**Definition 4.16** Let \( e \) be an edge. We say that \( e \) is abandoned at \( t \) if there is a time \( t' \geq t \) such that the two following conditions hold.

- no latch \( l \) s.t. \( e \sim l \) closes during \([t, t']\).
- There is a latch \( l \sim e \) that opens at \( t' \).

**Definition 4.17** A capricious nondet is a nondet where there are no restrictions on the signals of the abandoned edges.

That is, if \( e \) is abandoned at \( t \) and is an output of a latch (gate) then the Latch Rule, Def. 4.1, (the Gate Rule, Def. 4.2) does not hold. The value of the edge signal at time \( t \) can be anything.

We consider the macro-behavior of a capricious nondet to be the behavior of the latches output when they are not abandoned and the latch is opaque. Actually, it is enough to look at the moments when the latches become closed. The concepts of macro-equivalence, “being functional” and “being strictly functional” are defined accordingly.

**Lemma 4.15** Let \( Y \) be a (regular) nondet, \( Y' \) its capricious variant and \( \Phi \) a disciplined time-set. Then \( \langle Y, \Phi \rangle \) is CD-mild iff \( \langle Y', \Phi \rangle \) is CD-mild.

Note that “CD-mild” cannot be replaced by “functional”, as demonstrated in example 4.1. For similar reason, “CD-mild” cannot be replaced by “strictly functional”.

**Proof Sketch.** The \( \Rightarrow \) direction is based on the observation that the values of abandoned edges are irrelevant. This follows from the Trace Inequalities Lemma (4.13); in its proof (Appendix A) we never use the “old” value of an abandoned edge.

The \( \Leftarrow \) direction follows from \( \mathcal{R}(Y, \Phi) \subseteq \mathcal{R}(Y', \Phi) \).

Capricious systems satisfy a strong variant of theorem 4.2. In this variant, not only \( \Phi \equiv Y \Phi' \) but also \( \mathcal{R}(Y, \Phi') \subseteq \mathcal{R}(Y, \Phi) \). This extra property much simplifies the proof which does not need the Trace Inequality Lemma. We will not use this stronger theorem, and therefore we do not prove it.
Consider a latch \( b \in \gamma \). Since \( \Phi' \) is disciplined, \( b \) either closes or is closed at \( \tau' \) under \( \Phi' \). By the minimality of \( \tau' \) and \( l_i \), \( R_{i_k}(\tau') = R_{i_k}(\tau') \). Since \( \Phi \) is disciplined and \( \Phi' \) a modest expansion, \( b \) is closed in \( \{ \tau, \tau' \} \) under \( \Phi \). Hence, \( R_{i_k}(\tau) = R_{i_k}(\tau') \). The last two equations imply: \( R_i(\tau) = R_i(\tau') \).

Since this holds for any \( b \in \gamma \), by Lemma \text{4.7}, \( R_i(\tau) = R_i(\tau') \). Since \( l \) is open in each of the appropriate time-sets, \( R_i(\tau) = R_i(\tau') \). Under \( \Phi \), \( l \) is closed in \( \{ \tau, \tau' \} \); hence, \( R_i(\tau) = R_i(\tau') \). The last two equations imply: \( R_i(\tau) = R_i(\tau') \). A contradiction. \( \square \)

### 4.10 The CD=0 Pulse-Expansion Theorem.

In this section we establish the next theorem whose proof will be given shortly.

**Theorem 4.2 (The CD=0 Pulse-Expansion Theorem)** Let \( \langle Y, \Phi \rangle \) be CD-mild, \( Y \) be PD>0, \( \Phi \trianglelefteq_m \Phi' \) and \( \Phi \) and \( \Phi' \) disciplined. Then \( \Phi \equiv_{[Y]} \Phi' \).

Given \( \langle Y, \Phi \rangle \), we’re going to define a *trace*, which is informally a traverse of a data bit along a path, annotated with timing information. That is:

**Definition 4.15** Let \( Y \) be a nondet and \( \Phi \) a time-set. A **trace** \( T \) of \( \langle Y, \Phi \rangle \) is a pair of two sequences, \( T = \langle \{ l_1, l_2, \cdots, l_k \}, \{ \Pi_1, \Pi_2, \cdots, \Pi_k \} \rangle \), s.t. the \( l_i \) are latches, \( l_i \sim l_{i+1} \), \( l_i \) is a pulse of \( \Phi_i \) and \( \Pi_i \leq \Pi_{i+1} \). The rise and fall times of a trace \( T \) are: \( t^R = \Pi_1 \) and \( t^F = \Pi_1 \). The rise-to-fall time of \( T \) is defined by: \( RF(T) = T^F - T^R \). The delay of \( T \) is \( P_D(T) = \sum_{i=1}^{k-1} P_D(l_i, l_{i+1}) \).

Note that not every trace describes the traverse of a bit. It does so if \( \Phi \) is disciplined and for every \( i \): \( \Pi_i \) is the last pulse of \( \Phi_i \) such that \( \Pi_i = \Pi_{i+1} \). (And this because of the “unambiguous bit” property). Having “fake” traces will not hinder us.

For \( \langle Y, \Phi \rangle \), define the following set of inequalities, called the **trace inequalities**:

\[
T^Y, \Phi = \{ RF(T) \geq P_D(T) \mid T \text{ is a trace of } \langle Y, \Phi \rangle \}.
\]

**Lemma 4.13 (The Trace Inequalities Lemma)** Let \( Y \) be CD=0 and PD>0 and \( \Phi \) be disciplined. Then \( \langle Y, \Phi \rangle \) is strictly functional iff all the inequalities of \( T^Y, \Phi \) hold.

The Trace Inequalities Lemma is proven in Appendix A. The following claim is an “axiom” of digital systems: “If a combinational logic does not have enough time to complete its computation, then the system fails.” As shown in example 4.1, this axiom is not correct in the general case. The \( \Rightarrow \) direction of lemma 4.13 is this axiom in a very restricted case. This direction seems to be trivial, but its proof is not easy.

This lemma is similar to lemmas presented in [22, 23, 24, 25]. In [24] there is a special case of lemma 4.13 where \( Y \) is restricted to a two-phase system. In [23], theorem 4.2 looks very similar to our one. However, it uses the “functional” concept instead of “strictly functional” (the last one was introduced only in [24]). Moreover, the inequalities in that theorem depend on the system behavior, while our inequalities come only from its structure and the time-set.

Let \( \Phi \trianglelefteq \Phi' \) and \( T' = \langle \{ l'_1, \cdots, l'_k \}, \{ \Pi'_1, \cdots, \Pi'_k \} \rangle \) be a trace of \( \langle Y, \Phi' \rangle \). Define the **source** of \( T' \) by:

\[
(T')^\Phi = \langle \{ l'_1, \cdots, l'_k \}, \{ \Pi'_1, \cdots, \Pi'_k \} \rangle,
\]

where \( l_i \) is the source of \( \Pi'_i \). The next lemma is immediate.

**Lemma 4.14** Let \( \Phi \) and \( \Phi' \) be disciplined time-sets of \( Y \), \( \Phi \trianglelefteq_m \Phi' \) and \( T' \) a trace of \( \langle Y, \Phi' \rangle \). Then \( (T')^\Phi \) is a trace of \( \langle Y, \Phi \rangle \) and \( RF(T') \geq RF(T) \).
The “unambiguous bit” property implies that a pulse which violates the “no surplus pulses” property is actually surplus.

As discussed in 6.2, the static 2-phase and k-phase clocking techniques obey the above restrictions.

**Lemma 4.12** Let $\Phi$ and $\Phi'$ be disciplined time-sets of a PD=0 nondet $Y$ and $\Phi \preceq_m \Phi'$. Then $\Phi \simeq_Y \Phi'$.

**Remark.** If $\preceq_m$ is replaced by $\preceq$, the lemma does not hold. For example, let $b \sim c$ and $\Phi, \Phi'$ be as depicted in figure 4.3. The bit stored in $c$ when $\Phi'$ falls can be determined by a new bit that passes through $b$ during its second pulse. In this case, $\Phi \not\simeq_Y \Phi'$.

**Proof.** By an above note, $\Phi$ and $\Phi'$ are comparable. Since $Y$ is PD=0 and $\Phi, \Phi'$ disciplined, lemma 4.4 implies that both $\langle Y, \Phi \rangle$ and $\langle Y, \Phi' \rangle$ are strictly functional.

Let $Y$ be a nondet similar to $Y$. Let $R \in \mathcal{R}(Y, \Phi), R' \in \mathcal{R}(Y, \Phi')$ and $I^R = I^{R'}$. In order to prove that for every latch $l$ and every time $t$, $\Phi_l(t) = \Phi'_l(t) = 0 \Rightarrow R_l(t) = R'_l(t)$, it is enough to show this equality at every fall of $\Phi'_l$. That is, if $\Pi$ is a pulse of $\Phi'_l$, then $R_l(\Pi) = R'_l(\Pi)$.

We prove it by contradiction. Assume that the above equality does not hold for some latches and pulses. Define: $\tau' = \min\{\Pi^l = \tau : \Pi^l \text{ is a pulse of } \Phi'_l \text{ and } R_l(\Pi^l) \neq R'_l(\Pi^l)\}$. The minimum exists because the above set is discrete and bounded from below. Define the set of latches $L = \{l \mid l \text{ is a latch, } \Pi \text{ a pulse of } \Phi'_l, \tau = \Pi^l \text{ and } R_l(\tau') \neq R'_l(\tau')\}$. Now, let $G'$ be the graph generated from $G$, the graph of $Y$, by deleting all the latches that are opaque at $\tau'$ under $\Phi'$. From the no transparent cycle property of $\Phi'$ follows that $G'$ is a acyclic. By definition, $L \subseteq G'$. Pick $l \in L$ which is minimal in the partial order induced by $G'$. Let $\Pi'$ be the pulse of $\Phi'_l$ such that $\Pi^l = \tau'$; let $\Pi$ be the source of $\Pi'$ and $\tau = \Pi^l$. 

![Figure 4.2: The relative pulses positions.](image)

![Figure 4.3: Non-modest expansion.](image)
that a distortion of a time-set (i.e., clock skew) does not disturb the functionality of the nondet under certain conditions.

Given a timing signal $\psi$, $\Pi$ is a negative pulse of $\psi$ if $\Pi$ is a maximal interval of reals over which $\psi$ is 0. Two timing signals $\psi$ and $\psi'$ are comparable if for every negative pulse $\Pi$ in $\psi$ there is a non-disjoint negative pulse $\Pi'$ in $\psi'$ ($\Pi \cap \Pi' \neq \emptyset$) and vice versa. Two time-sets $\Phi$ and $\Phi'$ of a nondet $Y$ are comparable if for every latch $l$: $\Phi_l$ and $\Phi'_l$ are comparable. Note that if $\Phi \preceq \Phi'$, then $\Phi$ and $\Phi'$ are comparable.

**Definition 4.13** Given time-sets $\Phi$ and $\Phi'$ of a nondet $Y$, we say that $\Phi$ and $\Phi'$ are $Y$-equivalent ($\Phi \cong_{[Y]} \Phi'$) if the following three conditions hold:

1. $\Phi$ and $\Phi'$ are comparable;
2. both $\langle Y, \Phi \rangle$ and $\langle Y, \Phi' \rangle$ are strictly functional;
3. for every nondet $Y$, similar to $Y$, and for every two runs, $R \in \mathbb{R}(Y, \Phi)$ and $R' \in \mathbb{R}(Y, \Phi')$ with $I^R = I^{R'}$, the following holds: $\Phi_l(t) = \Phi'_l(t) = 0 \Rightarrow R_l(t) = R'_l(t)$ for every latch $l$ and every time $t$.

**Lemma 4.11** Let $Y$ be CD=0, $\langle Y, \Phi \rangle$ and $\langle Y, \Phi' \rangle$ strictly functional and $\Phi \cong_{[Y, \Phi = 0]} \Phi'$. Then $\Phi \cong_{[Y]} \Phi'$.

**Proof.** Clearly, conditions 1 and 2 of definition 4.13 hold. Concerning the third condition, let $Y$ be a nondet similar to $Y$. Let $R \in \mathbb{R}(Y, \Phi)$, $R' \in \mathbb{R}(Y, \Phi')$, $\hat{R} \in \mathbb{R}(Y^{PD=0}, \Phi)$, $\hat{R}' \in \mathbb{R}(Y^{PD=0}, \Phi')$ be runs and $I^R = I^{R'} = I^{\hat{R}} = I^{{\hat{R}}'}$. Since $\langle Y, \Phi \rangle$ and $\langle Y, \Phi' \rangle$ are strictly functional, $\langle Y, \Phi \rangle$ and $\langle Y, \Phi' \rangle$ are functional. Hence, according to lemma 4.5, $R \cong_{[\Phi]} \hat{R}$ and $R' \cong_{[\Phi']} \hat{R}'$. Since $\Phi \cong_{[Y, \Phi = 0]} \Phi'$, for every latch $l$ and every time $t$ such that $\Phi_l(t) = \Phi'_l(t) = 0$: $\hat{R}_l(t) = \hat{R}'_l(t)$, and as the result, $R_l(t) = R'_l(t)$.

To establish stronger statements, we need to restrict the time-sets to disciplined ones as follows.

**Definition 4.14** A time-set $\Phi$ of $Y$ is disciplined if it satisfies the additional requirements:

- **Unambiguous bit property.** Let $b$ and $c$ be latches, $b \sim c$, $\Pi_b$ and $\Pi_c$ pulses of $\Phi_b$ and $\Phi_c$ respectively and $\uparrow \Pi_b \prec \Pi_c \downarrow$. Then: $\Pi_b \downarrow \Pi_c \downarrow$.

- **No surplus pulses.** Let $l$ be a latch and $\Pi$ a pulse of $\Phi_l$. Then there are $Y' \sim Y$ and $R' \in \mathbb{R}(Y', \Phi)$ s.t. $R'_l(\uparrow \Pi) \neq R'_l(\downarrow \Pi)$.

- **No transparent cycle.** As defined above, at any moment any directed cycle of $Y$ has a closed latch.

**Remarks**. The above requirements make no sense for systems that are heavily based on contamination delay like Wave Pipelining [40]. Recall that we focus on systems that use contamination delay only for enduring clock-skew, if at all.

Even a nondet of interest may be functional with time-sets that violate the “unambiguous bit” property. We require this restriction since it simplifies the analysis of the nondet’s behavior. Look at figure 4.2. Let $x$ and $y$ be the bits that latches $b$ and $c$ hold after the falls of their timing signals. In cases (1), (2) and (3), it is clear that bit $x$ “participates” in the computation of $y$ in the “correct operation” of the nondet. In case (4) we do not know whether $x$, or any previous value passing through $b$ during the $\Phi_b$ pulse, “participates” in the computation of $y$.
4.8 Being Strictly Functional

The CD Pulse-Expansion Theorem is only a secondary mechanism for enduring clock-skew since some clocking techniques do without it. The main mechanism, which is indispensable, is formalized by the CD=0 Pulse-Expansion Theorem. This mechanism is much more subtle. Under it, \( \mathcal{R}(Y, \Phi) \neq \mathcal{R}(Y, \Phi') \); nevertheless, both systems are equivalent at the digital level of abstraction. We are able to formulate this mechanism only under restricted conditions discussed below.

A major difficulty is that not much can be deduced from the fact that a nondet is functional, since this may be due to the wrong reason as illustrated by the next example.

**Example 4.1** Consider a (combinational) logic \( A \) leading to a logic \( B \) and all are surrounded by latches (figure 4.1). Assume that \( B \) hasn’t enough time to complete its computation, but \( A \), which has enough time, computes a constant function. Thus the whole nondet is functional, despite its “invalid” timing. Obviously, such a system is neither practical nor interesting.

![Figure 4.1: A system working because of the wrong reason](image)

Because of this deficiency of the “functional” concept, we use the *strictly functional* concept introduced by Ishii, Leiserson and Papaeftymiou in [24]. (They called it *proper timing*).

**Definition 4.12** Two static nondets \( Y_1 \) and \( Y_2 \) are similar \( \langle Y_1 \sim Y_2 \rangle \) if they are identical in all respects, except for the boolean functions associated with the gates and the latches. (Recall that a latch may invert its input.) A pair \( \langle Y, \Phi \rangle \) is strictly functional if for every \( Y' \sim Y \): \( \langle Y', \Phi \rangle \) is functional.

In other words, \( \langle Y, \Phi \rangle \) is strictly functional if it is functional, and moreover this property does not depend on the gates’ boolean functions or the actual data processed by the system.

A pair \( \langle Y, \Phi \rangle \) is called **CD-mild** if \( \langle Y^{CD=0}, \Phi \rangle \) is strictly functional. Note that if \( \langle Y, \Phi \rangle \) is CD-mild then by lemma 4.2 it is strictly functional, and this functionality is not based on the contamination delay of the system. We use the term “CD-mild” rather than “CD-free” since the contamination delay may help \( \langle Y, \Phi \rangle \) to endure clock skew, as demonstrated by Theorem 4.1. As said, the focus of this paper is on CD-mild systems.

4.9 Time-Sets Equivalence

We will next introduce the concept of “equivalent” time-sets, meaning that replacing one by another does not change the nondet’s macro-behavior. In later sections, we will use this equivalence to show
4.7 The CD Pulse-Expansion Theorem

We present here the CD Pulse-Expansion Theorem. This Theorem formalizes the industry contemporary technique of applying contamination delay to prevent double clocking failures due to clock skew. In a nutshell, assume that \( (Y^{CD=0}, \Phi) \) is functional. This imply, by lemma 4.1, that \( (Y, \Phi) \) is functional, and moreover, its functionality is not based on its contamination delay. The latter attribute of \( (Y, \Phi) \) enables it to endure a restricted distortion of \( \Phi \) as formalized below.

**Definition 4.9** A timing signal \( \psi' \) is a pulse-expansion of a timing signal \( \psi \) (\( \psi \preceq \psi' \)) if it is obtained by the expansion of some pulses in \( \psi \). That is:

- for every pulse \( \Pi \) of \( \psi \), there exists exactly one pulse \( \Pi' \) of \( \psi' \), called the image of \( \Pi \) such that \( \Pi \subseteq \Pi' \).

- for every pulse \( \Pi' \) of \( \psi' \), there exists exactly one pulse \( \Pi \) of \( \psi \), called the source of \( \Pi' \) such that \( \Pi \subseteq \Pi' \).

**Definition 4.10** Given time-sets \( \Phi \) and \( \Phi' \) of a nondet \( Y \), we say that \( \Phi' \) is a pulse-expansion of \( \Phi \) (\( \Phi \preceq \Phi' \)) if for every latch \( l \), \( \Phi_l \preceq \Phi'_l \). We say that \( \Phi' \) is a modest pulse-expansion of \( \Phi \) (\( \Phi \preceq_m \Phi' \)) if \( \Phi \preceq \Phi' \) and no new intersections between pulses are created. That is, let \( \Pi_i \) and \( \Pi_c \) be disjoint pulses of \( \Phi_i \) and \( \Phi_c \) respectively. Let \( \Pi'_i \) and \( \Pi'_c \) be their images. Then \( \Pi'_i \) and \( \Pi'_c \) are disjoint.

**Definition 4.11** A time-set \( \Phi' \) is a right-expansion of \( \Phi \) if \( \Phi \preceq \Phi' \) and the pulses of \( \Phi' \) are expanded only to the right. That is, let \( \Pi' \) be the image of \( \Pi \), then

\[ ^1\Pi = ^1\Pi' \text{ and } \Pi^1 \leq \Pi'^1. \]

For a real \( \Delta \geq 0 \), \( \Phi' \) is a \( \Delta \)-right-expansion of \( \Phi \) if, in addition, \( \Pi'^1 \leq \Pi^1 + \Delta \).

A two-latch path is a latchless path whose two endpoints are latches. Define the *contamination delay* of a nondet \( Y \) by:

\[ C_D(Y) \triangleq \min \{ C_D(p) \mid p \text{ is a two-latch path} \}. \]

**Theorem 4.1 (The CD Pulse-Expansion Theorem)** Let \( (Y^{CD=0}, \Phi) \) be functional and \( \Phi' \) a \( C_D(Y) \)-right-expansion of \( \Phi \). Then \( \Re(Y, \Phi) = \Re(Y, \Phi') \).

**Remark.** The expansion is not necessarily modest and there are no additional restrictions on \( \Phi \) and \( \Phi' \) as in the CD=0 Theorem.

**Proof.** Assume, for a contradiction, that \( R \) is a run of \( Y \) under exactly one of \( \Phi \) or \( \Phi' \). Hence, \( R \) must violate the Latch Rule (Def. 4.1) under the other time-set. It follows that there is a latch \( l \), a pulse \( \Pi \) of \( \Phi \), and its image \( \Pi' \) s.t. \( R \) is not stable in \([\Pi^1, \Pi'^1]\).

Assume, w.l.o.g., that the above \( l \), \( \Pi \) and \( \Pi' \) are these where \( \Pi^1 \) is minimal. Hence, in the interval \( (-\infty, \Pi^1] \), \( R \) is a run under both \( \Phi \) and \( \Phi' \).

By Lemma 4.1, \( R \) is a run of \( (Y^{CD=0}, \Phi) \) in \(( -\infty, \Pi^1] \). By Lemma 4.6, \( Y^{\Phi} \) is firm at \( \Pi^1 \) under \( R \) and \( Y^{CD=0} \). Clearly, it is also firm under \( R \) and \( Y \).

By Lemma 4.8, \( Y_{\Phi} \) is stable in \([\Pi^1, \Pi^1 + C_D(Y)] \). But \([\Pi^1, \Pi^1] \subset [\Pi^1, \Pi^1 + C_D(Y)] \). A contradiction. \( \square \)
Let \( e \) be an edge and \( x \) a vertex (which may be one of \( e \) endpoints). We say that \( x \) **precedes** \( e \) (\( x \sim e \)) if there is a latchless path starting at \( x \) and ending at \( e \). Similarly, \( e \sim x \) means that there is a latchless path starting at \( e \) and ending at \( x \). (These paths may be just the edge \( e \).)

For a directed path \( p = \langle v_1, v_2, \ldots, v_n \rangle \), define its propagation delay by \( P_D(p) = \sum_{i=1}^{n} P_D(v_i) \), and its contamination delay by: \( C_D(p) = \sum_{i=1}^{n} C_D(v_i) \). (The propagation and contamination delays of latches are zero.)

For two vertices, \( x \) and \( y \), define: \( P_D(x, y) = \max \{0, P_D(p) \mid p \text{ is a latchless path from } x \text{ to } y\} \); similarly define: \( C_D(x, y) = \min \{\infty, C_D(p) \mid p \text{ is a latchless path from } x \text{ to } y\} \); For an edge \( e = (v \rightarrow u) \) and a vertex \( x \), define: \( P_D(x, e) = P_D(x, v) \) and \( C_D(x, e) = C_D(x, v) \).

For an edge \( e \), define the **fan-in cone** of \( e \) as the set of gates that precedes \( e \); that is: \( \Upsilon_e = \{ x \mid x \sim e, x \text{ a gate}\} \); define the **fan-in border** of \( e \) as the set of latches that precedes \( e \); that is: \( ?_e = \{ l \mid l \sim e, l \text{ a latch}\} \). Note that \( \Upsilon_e \) may be empty while \( ?_e \) is never empty. For a latch \( l \) let \( \hat{l} \) denote the edge entering \( l \).

**Lemma 4.6** Let \( \langle Y, \Phi \rangle \) be functional and \( Y \) be CD=0. Let \( l \) be a latch, \( \Pi \) a pulse of \( \Phi \), and \( R \in \mathcal{R}(Y, \Phi) \). Then \( \Upsilon^<_l \) is firm\(^1\) at \( \Pi^1 \) under \( R \).

**Proof.** Assume, for a contradiction, that \( \Upsilon^<_l \) is not firm at \( t = \Pi^1 \). Let \( g \in \Upsilon^<_l \) be a gate that is not firm at \( t \). Since \( Y \) is CD=0, one can modify \( R \) into \( R' \) such that \( R'(t) = \frac{1}{2} \) on a path from \( g \) to \( t \). Particularly, \( R'_l(t) = \frac{1}{2} \). As the result, \( l \) will hold the non-logical value when it closes, which contradicts the definition of being functional. \( \square \)

The following four lemmas relate to \( \Upsilon_e \) and \( ?_e \); all can be proved by induction on the depth of \( \Upsilon_e \).

**Lemma 4.7** Let \( Y \) be PD=0, \( R \in \mathcal{R}(Y, \Phi) \), \( R' \in \mathcal{R}(Y, \Phi') \), \( e \) an edge and \( t, t' \) times. Let \( R_i(t) = R'_i(t') \neq \frac{1}{2} \) for every \( i \in ?_e \). Then \( R_e(t) = R'_e(t') \neq \frac{1}{2} \).

Define the contamination delay of a fan-in cone \( \Upsilon_e \) by: \( C_D(\Upsilon_e) = \min \{ C_D(i, e) \mid i \in ?_e \} \). The next lemma will enable us to use contamination delay for expanding the pulses of a working system.

**Lemma 4.8** Let \( R \in \mathcal{R}(Y, \Phi) \), \( e \) an edge and \( \Upsilon_e \) be firm at \( t \) under \( R \). Then \( \Upsilon_e \) is stable in \([t, t + C_D(\Upsilon_e)]\).

**Lemma 4.9** Let \( R \in \mathcal{R}(Y, \Phi) \), \( t \) a time and \( e \) an edge. Assume that \( R_i \) is stable in \( (t - P_D(i, e), t) \) for each \( i \in ?_e \). Then \( \Upsilon_e \) is firm (and consistent) at \( t \) under \( R \).

The next lemma complements the previous one.

**Lemma 4.10** Let \( R \in \mathcal{R}(Y, \Phi) \), \( Y \) be PD\(>0 \) and CD=0, \( e \) an edge and \( t \) a time. Let \( l \in ?_e \), and \( R_l \) not stable in \( (t - P_D(l, e), t) \). Then there is a run \( R' \in \mathcal{R}(Y, \Phi) \) s.t. \( I^{R'} = I^R \) and \( R'_l(t) = \frac{1}{2} \).

Note that this lemma does not hold without the PD>0 condition; e.g., if \( \Upsilon_e \) is a path of several gates starting at \( g \) and \( P_D(g) = 0 \), then \( g \) may “stop” the propagation of a transition.

\(^1\)i.e., all the gates of \( \Upsilon^<_l \) are firm.
4.5 Being Functional

**Definition 4.8** A pair \((Y, \Phi)\) is functional if the following conditions hold:

- all its runs having the same initial state are \(\Phi\)-macro-equivalent;
- for any run \(R\) and any latch \(l\): whenever \(l\) is opaque, it holds a logical value.

In other words, a nondet is functional if its macro-behavior is deterministic and logical\(^1\). Only under this condition, the digital abstraction of the system is meaningful.

The next lemma follows immediately from Lemma 4.1.

**Lemma 4.2** If \(Y \subseteq Y'\) and \(\langle Y', \Phi \rangle\) is functional, then \(\langle Y, \Phi \rangle\) is functional.

A time-set \(\Phi\) of \(Y\) has the no transparent cycle property if at any moment any directed cycle of \(G\) has a closed latch.

A nondet \(Y\) is \(CD=0\) if \(C_D(g) = 0\) for every gate \(g\). If a \(CD=0\) nondet \(Y\) is clocked by time-set \(\Phi\) which violates the “no transparent cycle” property, then there exists a directed cycle \(c\) in \(Y\) and a time \(t\) such that all the latches on \(c\) are transparent. Thus, we can construct a run \(R\) s.t. at \(t\) all the edges on \(c\) are set to the non-logical value \(\frac{1}{2}\), and this remains thereafter. This implies the next lemma.

**Lemma 4.3** Let \(\langle Y, \Phi \rangle\) be functional and \(Y\) be \(CD=0\). Then \(\Phi\) has the “no transparent cycle” property.

A nondet \(Y\) is \(PD=0\) if \(P_D(g) = 0\) for every gate \(g\). (Note that such a nondet is also \(CD=0\).) A nondet is \(PD>0\) if \(P_D(g) > 0\) for every gate \(g\).

**Lemma 4.4** Let \(Y\) be \(PD=0\) and \(\Phi\) have the “no transparent cycle” property. Then \(\langle Y, \Phi \rangle\) is functional. Moreover, it is deterministic; i.e., it has exactly one run for each initial state.

The lemma follows from the observation that in such a system if no latch opens during an interval \(I\) then all the data-signals are stable during \(I\).

Given a nondet \(Y\), define the \(CD=0\) nondet \(Y^{CD=0}\) to be identical to \(Y\) in all respects, except that the contamination delays of all \(Y^{CD=0}\) gates are zero. The \(PD=0\) nondet \(Y^{PD=0}\) is defined similarly; i.e., the propagation (and contamination) delays of all \(Y^{PD=0}\) gates are zero. Clearly, if \(Y\) is \(CD=0\) then \(Y^{PD=0}\) \(\subseteq Y\). Hence, Lemma 4.1 implies:

**Lemma 4.5** Let \(Y\) be \(CD=0\) and \(\Phi\) a time-set. Then \(\langle Y, \Phi \rangle\) is functional iff for every run \(R \in \mathcal{R}(Y, \Phi)\) and \(\hat{R} \in \mathcal{R}(Y^{PD=0}, \Phi)\): \(I^R = I^{\hat{R}} \Rightarrow R \simeq_{[x]} \hat{R}\).

4.6 Timing Lemmas

We establish here several timing lemmas that will be used later.

Let \(Y\) be a nondet. A latchless path is a directed path in \(Y\) whose internal vertices are gates (but the endpoints may be latches). We say that a vertex \(x\) precedes a vertex \(y\) (\(x \rightsquigarrow y\)) if there is a latchless path from \(x\) to \(y\).

---

\(^1\)An actual system can, of course, be useful even if some of the latches sometimes hold invalid data. Our insistence that they always hold valid data follows the timing discipline introduced by Ward and Halstead [37].
Definition 4.4 Given a nondet \( Y \), we define a time-set \( \Phi \) of \( Y \) as an association of a timing signal, denoted \( \Phi_l \), for each latch \( l \).

Henceforth, \( \Phi, \Phi' \), etc. will be used to denote time-sets of the nondet in question.

4.4 Run of a System

The micro-behavior of every gate in a nondet is non-deterministic, hence the micro-behavior of the whole nondet is non-deterministic. To wit, a pair \( (Y, \Phi) \) has many possible behaviors which we call runs.

Definition 4.5 A run \( R \) of \( (Y, \Phi) \) is an association of a data-signal, denoted \( R_e \), to each edge \( e \), such that:

- If \( e_1 \) and \( e_2 \) are two outputs of a vertex, then: \( R_{e_1} = R_{e_2} \); (Recall that the set of edges exiting a vertex represent the same, single net driven by this vertex.)
- The latches’ outputs satisfy the Latch Rule (Def. 4.1);
- The gates’ outputs satisfy the Gate Rule (Def. 4.2);
- In the interval \((-\infty, 0]\) the signals are stable.

The last condition serves for the system initialization. It implies that all the gates are firm in the interval \((-\infty, 0]\), and consequently, are consistent. Given a run \( R \), the initial state of \( R \), denoted \( I^R \), is the function from the edges to \( \{0, 1\} \) defined by: \( I^R(e) = R_e(0) \). We consider the system initialization, not because we care about it, but because we do not know to define “working correctly” without it.

For any node \( v \), denote by \( R_v \) the signal \( R_e \), where \( e \) is some output of \( v \). (Note that the signals of all the outputs are equal.)

Given a pair \( (Y, \Phi) \), define the following set of runs:

\[
\mathcal{R}(Y, \Phi) \triangleq \{ R \mid R \text{ is a run of } (Y, \Phi) \}.
\]

We define a relation “\( \leq \)” for nondets. Given two nondets \( Y \) and \( Y' \), \( Y \leq Y' \) if they are identical in all respects, except of the propagation and contamination delays of the gates, and for every gate \( g \) the following inequalities hold:

\[
P_D(g) \leq P_D'(g) \\
C_D(g) \geq C_D'(g)
\]

The next lemma is immediate.

Lemma 4.1 Let \( Y \) and \( Y' \) be nondets and \( Y \leq Y' \). Then \( \mathcal{R}(Y, \Phi) \subseteq \mathcal{R}(Y', \Phi) \).

We consider the macro-behavior of a nondet to be the behavior of the latches when they are opaque\(^1\), and therefore two runs are \( \Phi \)-macro-equivalent if they are equal in that respect. That is:

Definition 4.6 Two runs \( R, R' \in \mathcal{R}(Y, \Phi) \) are \( \Phi \)-macro-equivalent (\( R \simeq_{[\Phi]} R' \)) if for every latch \( l \) and for any time \( t \): if \( l \) is opaque at \( t \), then \( R_l(t) = R'_l(t) \).

Definition 4.7 Two runs \( R, R' \in \mathcal{R}(Y, \Phi) \) are \( \Phi \)-macro-equivalent up to time \( \bar{t} \) (\( R \simeq^*_{[\Phi]} R' \)) if Def. 4.6 holds for any \( t \leq \bar{t} \).

\(^1\)Note that this resembles the spotlight argument (section 2) where we examine the outputs of sections only during selected intervals.
That is, the latch is transparent (open) when the clock is high. The output changes instantly with the input. Otherwise, the latch is opaque (closed), and we say that the latch holds the value \( s_o(t') \). Note that in the definition we use the fact that timing signals are continuous from the left.

The above latch is a non-inverting one; the model has also a variant of this latch that inverts its input.

### 4.2.1.1 Gate.

Let \( g \) be a vertex of \( G \) which is a gate. This \( g \) is associated with a \( k \)-placed boolean function \( f_g \) where \( k \) is the in-degree of \( g \). The role of \( g \) is to compute \( f_g \) and transmit it on its outputs. A gate has at least one input and at least one output. Recall that our gate is static, hence it is not controlled by any timing signal.

Generally, the gate’s behavior is non-deterministic; i.e., the input signals do not uniquely determine the output signal. Only under certain conditions, defined shortly, the output value at time \( t \) is determined by the input signals. A gate has two parameters that are associated with these conditions: the propagation delay, denoted \( P_D(g) \) and the contamination delay\(^1\) \( C_D(g) \), satisfying \( P_D(g) \geq C_D(g) \geq 0 \). The semantics of these parameters will be described shortly.

Given a gate \( g \) with inputs \( i_1, i_2, \ldots, i_k \) and an output \( o \), we say that \( g \) is consistent at time \( t \) if all its inputs are logical at that time and: \( f_g(s_1(t), s_2(t), \ldots, s_k(t)) = s_o(t) \). Note that this does not mean that \( s_o(t) \) is the causal result of the \( s_i(t) \); only that it is consistent with them.

For any reals \( x \) and \( y \) define \( (x, y) \triangleq (x, y] \cup \{y\} \); i.e., the only different between \( (x, y) \) and \( (x, y] \) is that the latter always contains \( y \) while the former may be empty. Given a gate \( g \) and a time \( t \), if the inputs of \( g \) have been stable in the interval \( (t - P_D(g), t) \), then \( g \) is firm at \( t \). Note that in this definition we need ‘\( \} \)’ rather than ‘\( ] \)’ only for the special case of \( P_D(g) = 0 \).

**Definition 4.2 (The Gate Rule)** The parameters \( P_D(g) \) and \( C_D(g) \) restrict the behavior of the gate \( g \) as follows:

- If \( g \) is firm at \( t \), then it is consistent at \( t \).
- If \( g \) is firm at \( t \), then \( s_o \) is stable in the interval \([t, t + C_D(g)]\).

In any other time \( t' \), \( s_o(t') \) can be any one of \( \{0, 1, \frac{1}{2}\} \).

The above conditions imply that if the inputs of \( g \) are stable in the interval \( (t - P_D(g), t + \frac{1}{2}) \), then \( g \) is consistent during \([t, t + \frac{1}{2}]\) and its outputs are stable during \([t, t + 1 + C_D(g)]\).

### 4.3 Nondet and Time-Set.

**Definition 4.3** A nondet \( Y \) constitutes a directed graph \( G = (V, E) \) annotated with additional information. The vertices of \( G \) are latches and gates. The graph \( G \) has no directed cycle\(^2\) without latches. Every gate \( g \) is associated with a boolean function \( f_g \) and propagation and contamination delays \( P_D(g) \) and \( C_D(g) \). Each latch \( l \) is controlled by its own timing signal. (There is no edge in the graph associated with this signal).

---

\(^1\) Contamination delay is also called minimal delay.

\(^2\) A directed cycle of a graph \( G \) is a sequence of vertices \((v_1, v_2, \ldots, v_n)\) s.t. \( G \) has an edge leading from \( v_i \) to \( v_{i+1} \) for each \( i \) and \( v_1 = v_n \).
**data signal** is a function: \[ s : \mathbb{R} \rightarrow \{0,1,\frac{1}{2}\}, \]

where 0 and 1 are logical values and \( \frac{1}{2} \) is the non-logical one. We say that a signal is **stable** in the time interval \( I \) if it is **logical** and constant during this interval. Non-continuity points of a signal are called **transitions**.

### 4.1.2 Timing Signals.

A **timing signal** is a function: \( \psi : \mathbb{R} \rightarrow \{0,1\} \), satisfying:

- \( \psi \) is continuous from the left;
- in the interval \( (-\infty, 0] \), \( \psi = 0 \) (this is used for initialization);
- in any finite interval \( \psi \) has only a finite number of transitions.

**Remarks.** A timing signal is not necessarily periodic, and may have only a finite number of transitions. Therefore, we do not call it ‘clock signal’. We assume instantaneous transitions of the timing signals. Our requirement of continuity from the left is, of course, meaningless in real systems; however, this will simplify our mathematical arguments.

Given a timing signal \( \psi \), \( II \) is a **pulse of** \( \psi \) if \( II \) is a maximal interval of reals over which \( \psi \) is 1. We define the rising edge and the falling edge of a pulse (or an interval) \( II \) by \( ^\uparrow II \triangleq \inf(II) \) and \( ^\downarrow II \triangleq \sup(II) \), respectively.

### 4.2 The Graph and Its Elements

We assume that a VLSI system is composed of gates and latches connected by wires. In our model, such a system is represented by a directed graph \( G \). The vertices of \( G \) represent the gates and the latches. A directed edge from \( x \) to \( y \) represents a wire connecting the output of \( x \) to an input of \( y \). The set of edges exiting a vertex \( x \) represents the net driven by \( x \). (So, our representation is not the bipartite graph one.) We refer to the edges entering a vertex \( x \) as the **inputs of** \( x \) and to those exiting \( x \) as the **outputs of** \( x \).

In a running system, an edge \( e \) is associated with a data-signal \( s \), that the edge deliver without distortion or delay. All the edges exiting the same vertex carry identical signals.

### 4.2.1 Latch.

Our latch has one input and at least one output, has no propagation delay, and is controlled by a single timing signal. Each latch \( i \) has its own timing signal named \( \Phi_i \). Let \( i \) be the latch input and \( o \) one of the outputs. Let \( s_i \) and \( s_o \) be their signals. The latch’s behavior is as follows:

**Definition 4.1 (The Latch Rule)**

\[
  s_o(t) = \begin{cases} 
    s_i(t) & \Phi_i(t) = 1 \\
    s_o(t'), t' = \max \{ t'' | t'' < t \land \Phi_i(t'') = 1 \} \cup \{0\} & \Phi_i(t) = 0 
  \end{cases}
\]
4 The Non-Deterministic Static Model

In this section we present a novel model, called NONDET (NON-DETerministic), for studying the analog behavior of static VLSI systems. The main novelty of our model is that the analog behavior of the system is non-deterministic. Clearly, the digital behavior of a functional system is deterministic. Hence, we will apply this non-determinism to reason whether a system in question is functional.

Our model assumes nothing about the methodology and clocking technique employed in the system design. In the model, each latch has its own timing signal and essentially there is no restriction on these signals (e.g., they are not periodic). Hence, although the model does not explicitly incorporate clock skew, it considers the actual behavior of a system under actual (rather than nominal) clocks.

We present the concept of ‘macro-behavior’ of a system in contrast to its ‘micro-behavior’. A system is functional (works correctly) if its macro-behavior is deterministic. The main contribution of our model are two Pulse Expansion Theorems (Theorems 4.1 and 4.2) stating that under certain conditions a system has the same macro-behavior under two different time-sets. In latter sections, we use these theorems to establish that a limited distortion (i.e., a clock skew) of a clock-set, under certain conditions, does not disturb the functionality of the system.

Some lemmas presented under this model are implicit “axioms” of the theory of digital systems. Another contribution of our model is the exact formulation of these “axioms” and their rigorous proof.

We then present a ‘capricious’ variant of our model that is much more non-deterministic then the first one, and therefore more failure prone. This variant will serve as our major tool to study dynamic VLSI systems. On the one hand, it is simpler then dynamic VLSI, yet on the other hand more failure prone. Hence, it will be used to show that some dynamic systems work.

Both our models are based on a directed graph with latches and gates as its vertices and some additional information. Our first model is an extension of a model presented by Ishii, Leiserson and Papaefthymiou [22, 23, 24, 25]. The main extensions are as follows.

- Their model is limited to systems whose gates have zero contamination delay. We model gates whose contamination delay is nonnegative.
- Their gates are deterministic, while ours are not. Hence, our model is more realistic.
- Our lemma 4.5 is actually their definition of “working correctly”. That definition is not readily extended to the case of non-zero contamination delay.

Our model and theirs are applied in a completely different fashion. They focus on a single system and ask the question: “does the system work correctly”. We focus on two systems and ask the question: “are the systems functionally equivalent”. In the following chapters, we will use this equivalence in the context where one system is derived from the other by clock skew.

4.1 Signals in the model.

4.1.1 Data Signals.

Actual signals are continuous functions from the time axis to analog voltage values. However, in our study, it is sufficient to restrict the analog values to three discrete ones. So, in our model, a

\[1\text{i.e., the system gates are static rather then dynamic.}\]
Figure 3.4: General sectioning.

inequality $P/2 \geq P_D(S_1)$ follows that at the beginning of the next $C_2$ interval $X_2$ are correct. Since during $C_2$ the $\phi_1$-latches are opaque, $X_2$ are stable.

From this lemma follow all the claims stated in 3.1 and 3.2. Actually, the system structure assumed in lemma 3.1 encompasses the general case of time-borrowing and it can provide a definition of the magnitude of time-borrowing in a given system, except for one problem. We also need to consider virtual sectioning where some gates are cut, one part belongs to one section, and the other to the second one. We present another definition of this magnitude in section 6.1.

Note that dynamic clocking techniques which are not skew-tolerant, like NORA, do not enable time-borrowing. It follows from the fact that a domino logic may evaluate only when $E = 1$; and in those clocking techniques the $E$ expressions of different sections are disjoint.
Figure 3.2: Time-borrowing on a long path. The hog is the dashed rectangular.

The term “time-borrowing” is misleading by implying that a hog seizes time from another logic having some time to spare. This, however, is wrong. Figure 3.3 depicts a hog in a time-borrowing system that gets its extra time from nobody.

Figure 3.3: Nobody to borrow from.

The presented sectioning and lemma 3.1 ensure that the system works with the clock-set of $P = 2$ and $Pw = 1$. The hog gets 1 extra time unit but there is no logic with time to spare.

3.3 General Sectioning

Consider the sectioning of a static two-phase system presented in 3.4. $A_1$ ($A_2$) denotes the logic in section $S_1$ ($S_2$) before the fence and $B_1$ ($B_2$) — the logic after the fence.

Define the symmetric clock-set $\Phi$ by $P = 2 \cdot \max\{P_D(S_1), P_D(S_2)\}$ and $Pw = \max\{P_D(A_1), P_D(A_2)\}$. The spotlight intervals $C_1$ and $C_2$ are chosen to coincide with the pulses.

Lemma 3.1 Consider section $S_1$ of figure 3.4 in isolation. Let $X_1$ be stable during a $C_1$ interval. Then $X_2$ is stable during the next $C_2$ interval.

Proof. From the inequality $|C_1| \geq P_D(A_1)$ follows that at the fall of $\phi_1$ the outputs of the $\phi_1$ latches are correct. After that the latches are opaque, and their outputs are stable. From the
3 Time-Borrowing

In this section we present the phenomenon of time-borrowing [20, 30] in the context of the static two-phase technique (with symmetric clocks). Informally, time-borrowing is the phenomenon that a part of the system, called a hog, “consumes” more time than its “fair share”. It comes out (section 6) that there is a strong association between this phenomenon and skew-tolerance.

3.1 Example

Consider the static two-phase system presented in figure 3.1. The given numbers denote the propagation delay of the corresponding logic. Its naive clock period is 6. This system has a directed cycle of (combinational) logics and latches that crosses each fence once. The total delay over the cycle is $2 + 1 + 1 = 4$, and the data has to traverse the cycle in a clock period. Clearly, the period must be at least 4.

Consider the (non-natural) sectioning depicted there. It will follow from this sectioning and lemma 3.1 that the system works with the symmetric clock-set of $P = 4$ and $P_w = 1$. The hogs in this case are the dashed rectangulars; each of them “consumes” 3 time units while their “fair share” is 2.

Figure 3.1: Example of time-borrowing. The numbers denote the propagation delays. The hogs are the dashed rectangulars.

3.2 Time-Borrowing Through a Number of Sections

Generally, a hog can be a long path that crosses the fences several times. Consider the static two-phase system presented in figure 3.2(1) (figure 3.2(2) is a “standard” presentation of the system). Its naive clock period is 20. Consider the sectioning depicted there. It will follow from this sectioning and lemma 3.1 that the system works with the symmetric clock-set of $P = 14$ and $P_w = 4$.

1Actually, it is hard both to formulate and to prove the above statement; section 6 addresses this issue; the above statement, as written, is wrong; the period may be much shorter.
Weste and Eshraghian’s Type A Technique. This technique ([38], pp. 164-165) is a four-phase one that uses four clocks (two couples). It’s built of regular dynamic gates and simple latches. It is CD-free.

Weste and Eshraghian’s Type B Technique. This latchless clocking technique([38], pp. 165-166) is a two-phase one that uses four clocks (one couple and two singles). It is CD-based.

The HS-PDCMOS Technique. High-Speed Precharge-Discharge CMOS system [41] is a two-phase one using four clocks (one couple and two singles). It’s built of special dynamic gates, is latchless and is CD-free.

The TSPC Technique. A True Single-Phase-Clock (TSPC) system [42, 43, 1] is actually a two-phase system using a single clock $\phi$. The structure of a TSPC system is similar to that of two-phase domino except of the following. It uses latches controlled by a single clock: one type is transparent when $\phi = 1$, the other when $\phi = 0$. Several implementations of these latches were presented. One of them is shown in figure 2.11.

![TSPC latches](image)

Figure 2.11: TSPC latches: (1) transparent when $\phi = 1$; (2) transparent when $\phi = 0$.

A faster variant of TSPC is the All-N-Logic [18] that uses sophisticated gates and latches in Sect.2.

The PHIMOS Technique. A PHIMOS [32] system is a two-phase one having a single clock like TSPC. It uses asynchronous SR-“latches”, not controlled by the clock. Its special dynamic gates work in dual-rail. Its structure is similar to this of figure 2.10 except that the static logics before the latches ($S_1$ and $S_2$) are omitted. This technique is CD-based.

The OTB Technique. The Opportunistic Time-Borrowing Domino Logic [20] is a two-phase system having no latches and no static logic. It has four clocks (one couple and two singles). This is the first dynamic clocking technique that enables time-borrowing (section 3). It is skew-tolerant only if each section has a significant contamination delay. (The inventors, however, failed to notice this capability and to require this delay.) It is CD-based.

The Domino Express Technique. This novel skew-tolerant clocking technique is presented in section 8. It is a three-phase one having three clocks (three singles), is latchless and enables time-borrowing. It is skew-tolerant even without contamination delay. I.e., it is CD-free.
In the presented sectioning of a system, Sect₁ is clocked by the quadruple (φ₁, φ₂, φ₁, φ₂). φ₁ and φ₂ control the latches which are transparent when φ₁ = 1. Each dynamic gate of D₁ is controlled by one phase of each couple; it evaluates during φ₁, presets during φ₂ and holds otherwise.

Section Sect₂ is clocked in the same manner by the quadruple (φ₂, φ₁, φ₂, φ₁), which is a translation of Sect₁ quadruple by half a cycle.

**The NORA Technique.** A NORA [16] system is similar to a tight two-phase domino system. The clocks are called φ and φ̅. It endures, without a contamination delay, a very limited type of clock skew. In this type, all the nodes receive the φ signal simultaneously, and the same with φ̅; so, the only possible skew is an inter-clock one or a distortion of the pulse-width; moreover, clock transitions must be instantaneous. To endure this limited clock skew, NORA uses latches of enhanced functionality¹ (e.g., C²MOS latches), and the structural rules severely restrict the logic composition. In spite of these measures, a NORA system must have contamination delay to endure any realistic clock imperfections.

NORA is not skew-tolerant even for that limited type of clock skew defined above. Consider a NORA system where all the static logics (S₁₁, S₁₂, S₂₁ and S₂₂ of figure 2.10) are empty. (This is permissible in NORA.) Clearly, the duration of the evaluation state of D₁ must be at least \( P_D(D₁) \). Hence, the pulse-width of the actual φ must be at least that much. The same holds for D₂ and φ̅. To guarantee this, the nominal pulse-width must be enlarged at least by the magnitude of the clock skew; i.e., the clock period must be increased by at least twice the clock skew.

¹The additional functionality of this NORA latch is as follows. The analog output signal is monotonic increasing (decreasing) when \( \psi = 0 \) (\( \psi = 1 \)).
Dynamic Clocking Techniques

In this section we present a number of contemporary dynamic clocking techniques. Among them, only the last two, OTB (under some conditions) and Domino Express, are skew-tolerant. We also show, as an example, that NORA is not skew-tolerant.

The major advantage of dynamic logic over the static one is a significant reduction of the gates input capacitance leading to improved performance. This made the use of dynamic logic wide-spread in high-speed CMOS microprocessors.

The dynamic clocking techniques use dynamic combinational logic, called domino logic, composed, according to restrictive rules (section 7), of dynamic gates. The clocks determine the state of a domino logic which is one of: preset, evaluate or hold.

- In the preset state, each edge is preset to a predetermined value.
- In the evaluate state, the logic computes the associated boolean function. Each input must be stable or change in a very restrictive way.
- In the hold state, the output is constant. Due to this state, a dynamic gate can also serve as a memory element, enabling latchless clocking techniques. Clearly, the clocking technique can dictate that the duration of this state is zero and this is actually the dominant variant.

Figure 2.8 presents the notation of a domino logic, where $E$ and $P$ are boolean expressions of the clocks. The logic evaluates during $E = 1$, presets during $P = 1$ and holds otherwise. The clocking technique dictates that always $E \land P = 0$. This notation does not specify what clocks are actually connected to the dynamic gates.

The Two-Phase Domino Technique. The Two-Phase Domino is presented in [26, 11, 38]; its structure is shown in figure 2.10. The latches are simple ones (section 2.1). It has two variants similar from all aspects (number of clocks, use of contamination delay) to the static two-phase technique. The tight variant (and its sub-variants) is dominant in contemporary high-speed CMOS VLSI.

\[1\] Usually, each of these expressions involves a single clock; e.g., $\phi_i$ or $\neg \phi_i$.  

Figure 2.8: 6-phase timing. (1) non-overlapping clocks; (2) tight clocks.
If the lemma’s premise is once satisfied then the whole system works correctly thereafter by induction. We refer to this reasoning as the **spotlight argument**.

The strength of this method is as follows:

- It considers a section in isolation rather than the whole system.
- It considers a fraction of the clock cycle rather than the whole cycle.
- It is irrelevant when each edge of $X_i$ firstly becomes stable and when it stops being so. In fact, we do not care what these edges are doing when not in the spotlight.
- The issue of when the combinational logic actually “works” is irrelevant.

Resolving the last two issues is hard in the complex cases. The advantage of this economy of concern will be magnified in the complicated cases.

### 2.3.2 The Static N-phase Technique

This clocking technique (figure 2.7) is a straightforward generalization of the two-phase one. The 3-phase variant is advocated in [15].

![N-phase static system](image)

Figure 2.7: N-phase static system.

It uses $2 \cdot N$ clocks ($N$ couples). Like the special two-phase case, we have two variants: non-overlapping and tight clocks. In both of them, $Pw = P(N - 1)/(N) - g$, where the gap $g$ is not negative. In the non-overlapping case $g > 0$, and in the tight one $g = 0$. Figure 2.8 presents examples of clock-sets for $N = 6$. Note that the tight variant is attractive only in the special case of $N = 2$; otherwise, it provides no reduction in the number of clocks.

In the non-overlapping variant for every interval $I$, $|I| < g$, there is a fence of latches opaque during all of $I$. This variant is CD-free. In the tight case for every $\epsilon > 0$ there is an interval $I$, $|I| = \epsilon$, where every fence is transparent sometime in $I$. Therefore, a clock skew of magnitude $\epsilon$ may make all the fences transparent during $I$. Hence, this variant is CD-based.

In the non-overlapping variant, a number of fences may be opaque simultaneously. However, only the fence closed last holds significant data; other fences hold data that is no longer relevant.

We show in section 6 that this clocking technique is skew-tolerant.
**Natural Sectioning**

Figures 2.5 presents two natural ways to partition a two-phase system into sections.

![Diagram of two natural sectionings](image)

Figure 2.5: Two natural sectionings.

Let $P_D(CL_i)$ denote the propagation delay of $CL_i$. Clearly, the system works correctly with the *naive* clock period: $P = 2 \cdot \max\{P_D(CL_1), P_D(CL_2)\}$ and arbitrary pulse-width\(^1\). This period is not necessarily optimal, as illustrated in section 3.

Figure 2.6 presents a possible clock-set (whose period is somewhat greater than $P$) and possible spotlight intervals for both sectionings. In the first case the length of $C_1$ and $C_2$ can be arbitrary short. In the second one: $|C'_1| = P_D(CL_2)$ and $|C'_2| = P_D(CL_1)$.

![Diagram of spotlight intervals](image)

Figure 2.6: Spotlight intervals in natural sectionings.

Establishing the one-step property for both sectionings is similar, so we consider only the second one.

**Lemma 2.1** Consider $S'_1$ in isolation. Let $X'_1$ be stable during a $C'_1$ interval. Then $X'_2$ is stable during the next $C'_2$ interval.

The proof is as follows. From $|C'_1| = P_D(CL_2)$ it follows that $X'_2$ are correct at the fall of $\phi_1$. Exactly at that time the $\phi_1$-latches close, keeping $X'_2$ stable during the next $C'_2$ interval. (Note that the pulse-width is irrelevant.)

---

\(^1\)Actually, the minimal pulse-width is determined by the propagation delay of the latches. Recall that in our case this is 0.
skew must be known at the logic design time. In contrast, CD-free systems can endure a clock skew of any magnitude by only modifying the clocks (and increasing the period if necessary).

Among the skew-tolerant systems there are both CD-based and CD-free ones. If a clocking technique is not skew-tolerant, then, by definition, the period must be enlarged to endure clock skew. In CD-based systems this solves only the zero clocking problem, while in CD-free ones it solves both of them.

Some clocking techniques (e.g., wave-pipelining [40, 33]) use contamination delay, not only to endure clock skew, but also to increase the performance; these techniques are out of the scope of this study. The other techniques either do not use contamination delay or use it only to endure clock skew; we call these techniques CD-mild.

2.3 Survey of Clocking Techniques

We give here a survey of the known high-speed clocking techniques, starting with the static clocking techniques (that use static gates) and ending with the dynamic ones. Following some of the conventions of the literature, we use the term $N$-phase system to refer exclusively to a system having $N$ sections; it has nothing to do with the number of clocks.

2.3.1 The Static Two-Phase Technique

The structure of a static two-phase system is depicted in figure 2.4; $CL_1$ and $CL_2$ are static combinational logics (acyclic networks of static gates). It uses four clocks: $\phi_1, \bar{\phi}_1, \phi_2, \bar{\phi}_2$ (two couples), where $\langle \phi_2, \bar{\phi}_2 \rangle$ is a translation of $\langle \phi_1, \bar{\phi}_1 \rangle$ by $P/2$, where $P$ is the period. Let $Pw$ denote the pulse-width of $\phi_1$ and $\phi_2$.

![Figure 2.4: Static Two-Phase system.](image)

There are two variants of the timing rules; in both of them $Pw = P/2 - g$, where the gap $g$ is not negative. Hence, there is always a fence of opaque latches.

- The non-overlapping clocks, popularized by Mead and Conway [31], have $g > 0$. This gap is used for enduring the clock skew. This variant is CD-free.

- The tight clocks, popularized by Weste and Eshraghian [38], have $g = 0$. In this case $\phi_1 = \bar{\phi}_2$, hence there are only two distinct clocks (one couple) — a great advantage over the non-overlapping case. This variant is CD-based.

We show in section 6 that this clocking technique is skew-tolerant.

\footnote{Nevertheless, our formal definition (4.8) of “working correctly” applies to these systems as well.}
2.2 Clock Imperfections and their Endurance

The clocks specified by the clocking technique are \textit{nominal} clocks; they always have instantaneous transitions. \textit{Actual} clocks, however, have two types of imperfections: clock skew and non-instantaneous transitions. Both types may cause a similar failure, and the clocking techniques \textit{endure} both of them actually in the same manner (this does not mean that they are skew-tolerant). Henceforth, we will refer only to the clock skew problem.

In a static system, there are four basic forms of failures caused by clock skew. (Additional forms, specific to dynamic systems, are addressed in section 7.3.) All are caused by a single distortion at a single latch; had this distortion not occurred the system would have worked correctly. The following terms are refinements of those of Fishburn [8]:

- \textit{zero clocking}. A latch closes too early, capturing contamination rather than data.
- \textit{double clocking}. A latch closes too late, capturing new contamination rather than data.
- \textit{indirect zero clocking}. A latch opens too late, causing a ‘zero clocking’ failure in a succeeding latch.
- \textit{indirect double clocking}. A latch opens too early, causing a ‘double clocking’ failure in a succeeding latch.

We distinguish clocking techniques by how they endure the double clocking problem. Those that depend on \textit{contamination delay} \footnote{also called \textit{minimal delay.}} \cite{37} for this end are called \textit{CD-based}, and the others \textit{CD-free}\footnote{CD-free systems are sometimes called \textit{race-free}.}. (Contamination delay never alleviates the zero clocking problem.) The CD-based case has three sub-cases. It may be that the fastest latches and gates of the current technology have enough contamination delay to endure the expected clock skew. Otherwise, contamination delay may be added to all the latches; this will increase the period. Another and more laborious approach is to selectively add contamination delay to critical paths. In all these cases, the magnitude of the clock
decided before the logic design; at this stage symmetric clocks are a natural choice. Like most of
the scientific literature, we focus on symmetric clock-sets.

For each $X_i$ there is a spotlight interval $C_i$, defined relatively to the clocks and according to
logic parameters like propagation delays. These intervals appear in the following alternate order:
$C_0, C_1, \ldots, C_{N-1}, C_0, \ldots$, and the period of these appearances equals the clocks’ period. We say
that $X_i$ is in the spotlight during $C_i$; at other times we do not care about its behavior.

In all the clocking techniques of interest, the structural and timing rules ensure the following
one-step property: if $X_i$ is stable\(^1\) during a $C_i$ interval, then $X_{i+1 (mod N)}$ is stable and correct\(^2\)
during the next $C_{i+1}$. This property implies, by induction, that once an $X_i$ is stable during its
spotlight, the whole system works correctly thereafter.

Note that on the one hand a section behaves in a combinational fashion — its output depends
only on its input. On the other hand, it has features of memory — its outputs are stable later than
its inputs.

All the above properties are functional. The only structural property common to all the relevant
techniques is that each section is acyclic.

### 2.1 The Latch

Most of our clocking techniques use latches as memory elements. The simple latch (of minimal
functionality) has one input\(^3\) and one output and is controlled by two clocks: $\psi$ and $\overline{\psi}$. (The
clocking technique dictates that these complement each other.) The latch is transparent or open
(out = in) when $\psi = 1$ (and $\overline{\psi} = 0$) and is opaque or closed (out is constant) otherwise. For
simplicity, the propagation delay of the latch is assumed to be zero\(^4\).

Figure 2.2 presents the popular dynamic\(^5\) latch, an implementation of the simple inverting latch.
A fence of latches transparent when $\psi = 1$ is depicted in figure 2.3.

---

\(^1\)“Stable” also means that the signals are logical; in complicated cases, “stable” will be substituted by other
conditions.

\(^2\)I.e., the $X_{i+1 (mod N)}$ values are equal to $f_i$ applied to the stable values of $X_i$.

\(^3\)Usually, a latch has several inputs and computes a boolean function; for simplicity, we ignore that.

\(^4\)This implies that the width of the setup/hold window is zero as well.

\(^5\)I.e., it fails if the period is too large; henceforth, we ignore this aspect.
2 VLSI Clocking Techniques

A synchronous digital system is a system whose temporal behavior is governed by periodic timing signals, called clocks, that control its memory elements and dynamic gates.

A clocking technique is a method of applying these clocks. In this paper we study only VLSI high-speed clocking techniques, e.g., those used in the critical core of high-performance microprocessors. Moreover, we focus on computation intensive rather than IO intensive systems; thus, we assume that our systems have no IO. Several such techniques are presented in literature, and all have the following features.

- They do not use flip-flops.
- They employ a small number of free-running clocks, all of the same period, called phases.
- They are comprehensive and also dictate the structure of the system.

A clocking technique is composed of two sets of rules:

- **Structural rules** that present the set of the basic elements (gates, latches, etc.), ways of their composition and what clocks are connected to what elements.
- **Timing rules** that constrain the form of the clock signals.

An important aspect of VLSI that is, however, beyond the scope of the clocking techniques is the implementation of the basic elements; the techniques specify only their functionality.

We consider a digital system as a directed network whose nodes are the basic elements (gates, latches, etc.). In all the clocking techniques of interest, the network can be partitioned into a small number \((N \geq 2)\) of sections: \(S_0, S_1, \ldots, S_{N-1}\), having the properties described below. (Such a partitioning is not necessarily unique.)

The outputs of section \(S_{i-1} (\mod N)\), denoted \(X_i\), are the inputs of section \(S_i\) (figure 2.1). Each section \(S_i\) is associated with a boolean function \(f_i\) and its main task is to compute \(f_i(X_i)\) and transmit this vector over \(X_{i+1} (\mod N)\).

Each section \(S_i\) is controlled by a small number of clocks: \(\phi_i^0, \phi_i^1, \ldots, \phi_i^k\), the same \(k\) for each section. The structural and timing rules for every section are identical except for the names of the clocks. In addition, there is a sequence of strictly positive reals, \(\Lambda_0, \Lambda_1, \ldots, \Lambda_{N-1}\) such that \(\sum \Lambda_i\) equals the period and the clocks of \(S_{i+1}\) are a forward translation of the clocks of \(S_i\) by \(\Lambda_i\). It seems that such a system has \(k \cdot N\) distinct clocks, but this is usually wrong; the clocking technique typically specifies that many of these clocks are identical. The association of periodic timing signals to these clocks constitutes a clock-set. Two complementary clock signals are called a couple. We will usually note how many distinct couples and singles a clock-set has. We use \(P\) to denote the period of the clock-set in question.

A clock-set is symmetric if all the \(\Lambda_i\) are equal. Symmetric clock-sets are preferable for two reasons: They are simpler to understand and to design with. Furthermore, the clocks are usually

---

1 In this section we use, without a definition, several technical terms; these terms are applied with their accepted meaning and they are defined in later sections.
2 A timing signal is periodic if the span of any two consequent rises is constant and equals the span of any two consequent falls; this span is the period.
3 Flip-flop based techniques are also irrelevant to our study, since they are clearly not skew-tolerant [9].
4 There are two exceptions to this statement. Contemporary systems employ ad-hoc clocks — a derivative of one of the main clocks that is tailored to the requirements of a small logic [39]. Recently, it is prevalent in low-power systems to stop the clocks of idle subsystems. Both these exceptions are out of the scope of this study.
The contents of this paper is as follows. Section 2 gives a survey of contemporary high-speed clocking techniques and presents the spotlight argument. In section 3 we introduce the concept of time-borrowing. Section 4 presents the static model and establishes the pulse-expansion theorems. In section 5 we formally define the magnitude of a clock skew and the concept of skew-tolerance. In section 6 we prove that the standard clocking technique of static VLSI is skew-tolerant. We also present there the self-borrowing technique. Section 7 addresses the issue of domino logic and considers its specific failure modes due to clock skew. In section 8 we present the Domino Express technique and prove that it is skew-tolerant. We also make a comparison between skew-tolerance capabilities of OTB and Domino Express. Finally, section 9 briefly describes the simulation we carried out in order to validate Domino Express.

In Appendix A we prove the trace inequalities lemma, which is an implicit “axiom” of the theory of digital systems. It’s used to prove the pulse-expansion theorems. In Appendix B we present how contemporary high-speed microprocessors handle the clock skew and elaborate on the clock distribution system of the 300-MHz Alpha microprocessor.
1 Introduction

In this paper, clock-skew is the temporal deviation of actual clocks' transitions from their expected ones. This deviation may be both location and time dependent; i.e., the deviations at two physical points (even if connected by a wire) may differ, and the deviations at the same point may vary over time.

A great effort is invested by the industry and the research community to reduce the clock-skew of synchronous VLSI systems. However, the subject of reducing the penalty caused by a clock-skew has not been well investigated. Our paper addresses this neglected subject.

It is commonly held that the clock period must be enlarged to endure clock-skew. In popular VLSI books we find the following statements.

Bakoglu ([3], page 353) writes:

... clock skew directly adds to the cycle time.

Glasser and Dobberpuhl ([15], page 345) write:

Uncertainties in our ability to exactly predict clock timing lowers the performance of a VLSI system.

It turns out that this common wisdom is incorrect. There are systems, called skew-tolerant\(^1\), which endure a significant clock-skew without increasing the clock period. The clocking techniques employed in such systems are the subject of this paper.

The major contribution of this work is the invention of a new clocking technique for domino logic, named Domino Express, which is skew-tolerant. (The same technique was found earlier and independently by Harris and Horowitz [19]). This invention is important since all contemporary domino techniques are not skew-tolerant, except for the Opportunistic Time Borrowing (OTB) technique [20] that is skew-tolerant under additional conditions. (Its inventors, however, failed to notice this property and to require these conditions).

Another contribution of this work is the surprising discovery that the static 2-phase clocking technique, which has been used for decades, is skew-tolerant.

We present two novel reasoning methods for studying the behavior of synchronous VLSI systems. The first, named the spotlight argument, focuses on the behavior of selected edges during selected time intervals. The second method is composed of two pulse-expansion theorems stating that, under certain conditions, the functionality of a system is not disturbed when the pulses of its clocks are expanded in an irregular fashion. These methods are used to reason about systems' behavior in general and particularly under clock-skew. The above two theorems are established under a novel model of VLSI systems. Some lemmas presented under this model are implicit “axioms” of the theory of digital systems. Another contribution of our model is the exact formulation of these “axioms” and their rigorous proof.

This work reveals that there is a tight linkage between time-borrowing and skew-tolerance. Most of the clocking-techniques we know either have both or have neither of these properties. In the former case, the magnitude of the skew-tolerance of a system is inverse to the magnitude of the time-borrowing it has; it seems that both properties compete over the same resource. The OTB technique is an exception to this rule, since it enables a time-borrowing of half a clock-cycle but its skew-tolerance is completely based on contamination delay and may be zero.

This work also presents a promising new technique, called self-borrowing, which is a derivative of time-borrowing that may increase the performance of future VLSI systems.

---

\(^1\)The term ‘skew-tolerant’ was coined by Harris and Horowitz [19].
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Sections</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Domino Logic</td>
<td>7.1 Domino Gate, 7.2 Standard Domino Gate, 7.3 Specific Domino Logic Clock Skew Problems</td>
</tr>
<tr>
<td>8</td>
<td>Domino Express</td>
<td>8.1 Three-sections structure, 8.2 Proof of the One-Step Property in the Non-Skew Case, 8.3 Skew-Tolerance of Domino Express, 8.4 Skew-Tolerance and Time-Borrowing of Domino Express (8.4.1 “Formal” proof), 8.5 Disadvantages of the Domino Express, 8.6 Comparison of Skew-Tolerance between OTB and Domino Express</td>
</tr>
<tr>
<td>9</td>
<td>Simulations</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>Proof of The Trace Inequalities Lemma</td>
<td>A.1 Strictly Functional Implies the Trace Inequalities, A.2 Inequalities Imply Strictly Functional</td>
</tr>
<tr>
<td>B</td>
<td>Clock Skew in Contemporary High-Speed VLSI</td>
<td>B.1 Clock Skew Causes, B.2 Contemporary Techniques to Reduce Clock Skew (B.2.1 Clocking System in Contemporary Microprocessors, B.2.2 Alpha Processors, B.2.3 300-MHz Alpha 21164 Microprocessor), B.2.3.1 The Clocking System Goals, B.2.3.2 The Clocking System Implementation, B.2.3.3 Algorithm of Reduction of Grid Capacitance, B.2.3.4 The Implementation Results</td>
</tr>
</tbody>
</table>
Skew-Tolerant Clocking Techniques

Ami Litman               Yuri Miroshnik
Computer Science Department, Technion
Haifa 32000, ISRAEL
email: [litman,mirosh]@cs.technion.ac.il

Abstract

A great effort is invested by the industry and the research community to reduce the clock-skew of synchronous VLSI systems. However, the subject of reducing the penalty caused by a clock-skew has not been well investigated. Our paper addresses this neglected subject.

It is commonly held that the clock period must be enlarged to endure clock-skew. It turns out that this common wisdom is incorrect. There are systems, called skew-tolerant, which endure a significant clock-skew without increasing the clock period. The clocking techniques employed in such systems are the subject of this paper.

The major contribution of this work is the invention of a new clocking technique for domino logic, named Domino Express, which is skew-tolerant. This invention is important since all contemporary domino techniques are not skew-tolerant, except for OTB that is skew-tolerant under additional conditions. (Its inventors, however, failed to notice this property).

Another contribution of this work is the surprising discovery that the static 2-phase clocking technique, which has been used for decades, is skew-tolerant.

We present two novel reasoning methods for studying the behavior of synchronous VLSI systems. The first, named the spotlight argument, focuses on the behavior of selected edges during selected time intervals. The second method is composed of two pulse-expansion theorems stating that, under certain conditions, the functionality of a system is not disturbed when the pulses of its clocks are expanded in an irregular fashion. These methods are used to reason about systems’ behavior in general and particularly under clock-skew. The above two theorems are established under a novel model of VLSI systems. Some lemmas presented under this model are implicit “axioms” of the theory of digital systems. Another contribution of our model is the exact formulation of these “axioms” and their rigorous proof.

This work reveals that there is a tight linkage between time-borrowing and skew-tolerance. Most of the clocking techniques we know either have both or have neither of these properties. (The OTB technique is an exception.) In the former case, the magnitude of the skew-tolerance of a system is inverse to the magnitude of the time-borrowing it has; it seems that both properties compete over the same resource.

This work also presents a promising new technique, called self-borrowing, which is a derivative of time-borrowing that may increase the performance of future VLSI systems.