

Modern Clock Distribution Systems

Simon Tam

Intel Corporation (SC12-408)

2.1 Introduction

Modern clock distribution design continues to face challenges in spite of significant advances in the last decade. We can distinguish three primary challenges. The first is the need to support higher clock frequencies based on the strong correlation between frequency and chip performance. Figure 2.1 shows processor clock frequency trend suggesting a continuous exponential increase in clock frequency with variable rates. Second, process technology scaling allows higher level of integration and larger die size leading to higher clock loading and larger distances the clock network needs to traverse. The final challenge is that technology scaling leads to an increase in on-die variations that may degrade clock performance if not properly addressed.

In order to address these design challenges successfully, it is necessary to understand the fundamental clocking requirements, key design parameters that affect clock performance, different clock distribution topologies and their trade-offs, and design techniques needed to overcome certain limitations. In this chapter, the following topics are presented:

- Definitions and Design Requirements
- Clock Distribution Topologies
- Microprocessor Clock Distributions
- Clock Design for Test and Manufacturing
- Elements of Clock Distribution Circuits
- Clock DFX (Design-for-Test and Design-for-Manufacturing) Techniques
- Multiclock Domain Distributions
- Future Directions

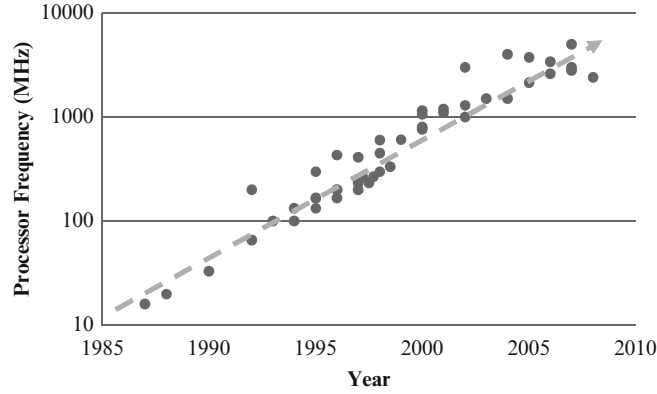


Fig. 2.1. Processor clock frequency trend [1–30]

2.2 Definitions and Design Requirements

Synchronous circuits may be simplified to have two timing limitations: setup (MAX delay) and hold (MIN delay). Setup specifies whether the digital signal from one stage of the sequential structure has sufficient time to travel to and “set-up” before being captured by the next stage of the sequential structure. Hold specifies whether the digital signal from the current state within a sequential structure is immune from contamination by a signal from a future state due to a fast path. Figure 2.2 shows a typical synchronous sequential structure bounded by two flip-flops with a logic circuit that exhibits a circuit delay of value T_d . The sequential elements are clocked by a source clock Ck1 and a destination clock Ck2.

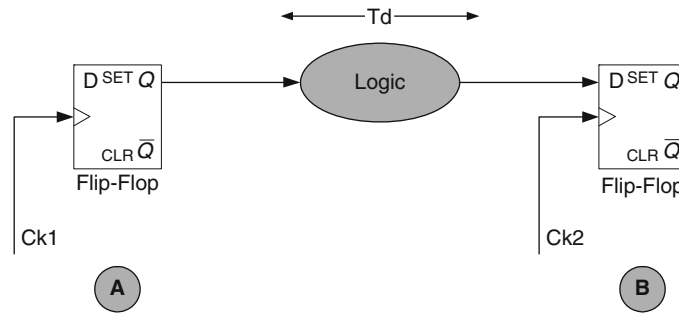


Fig. 2.2. Sequential structure bounded by flip-flops

Clocks Ck1 and Ck2 can be spatially far apart on die as shown in Fig. 2.3. In this illustration, clocks Ck1 and Ck2 have their root at a common point (Clock Gen.) and are routed through the on-die clock distribution before arriving at their respective destinations. Locations A and B constitute the source and destination of the sequen-

tial path. The transit time (clock latency¹) of CK1 and Ck2, their latency difference, their variations, and the design structure to minimize the above are the main topics of discussion in subsequent sections. As will be shown later, the timing uncertainty and the timing differences of Ck1 and Ck2 will play a fundamental role in determining whether the setup and the hold constraints can be robustly met.

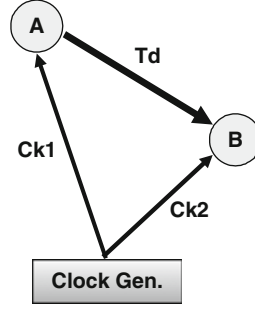


Fig. 2.3. Sequential path showing explicit clock distribution

2.2.1 Setup and Hold Timing Constraints

This section will present a brief outline of the formulation and the key parameters affecting the setup and the hold constraints.

The setup constraint specifies how data from the source sequential stage at cycle N can be captured reliably at the destination sequential stage at cycle $N + 1$. This situation is illustrated in Fig. 2.4 in which the source clock Ck1 is shown to lag behind the receive clock Ck2 due to clock uncertainty. The constraint for the source data to be received reliably by the receiver is defined in inequality 2.1, where T_{d-slow} is the slowest (maximum) data path delay, T_{su} is the setup time for the receiver flip-flop, T_{per} is the clock period, T_{Ck1} and T_{Ck2} are the arrival times for clocks Ck1 and Ck2 (at cycle N) respectively.

$$T_{per} \geq T_{d-slow} + T_{su} + |T_{Ck1} - T_{Ck2}|. \quad (2.1)$$

In the setup constraint situation, the available time for data propagation is reduced by the clock uncertainty defined as the absolute difference of the clock arrival times. This uncertainty $|T_{Ck1} - T_{Ck2}|$ can originate from various sources and their classification will be discussed in subsequent sections. In order to accommodate the clock uncertainty and meet the inequality in (2.1), either clock period must be extended or path delay must be reduced. In either case, power and operating frequency may be affected.

The hold constraint is shown in Fig. 2.5. This case specifies the situation where the data propagation delay is fast, and clock uncertainty makes the problem even

¹ The latency is referenced to the root of the distribution.

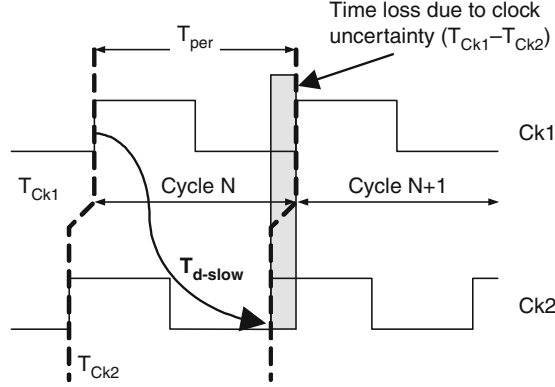


Fig. 2.4. Timing diagram for the setup constraint

worse and the data intended to be captured at cycle $N + 1$ is erroneously captured at cycle N , corrupting the receiver state. In order to ensure that the hold constraint is not violated, the design has to guarantee that the minimum data propagation delay is sufficiently long to satisfy inequality (2.2):

$$T_{d-fast} \geq T_{hold} + |T_{Ck1} - T_{Ck2}|, \quad (2.2)$$

where T_{hold} is the hold time requirement for the receive flip-flop.

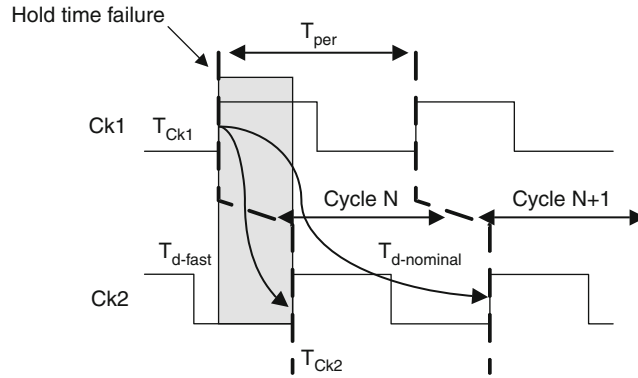


Fig. 2.5. Timing diagram for the hold constraint

In the discussion above, the following relationship is expected to hold (inequality 2.3):

$$T_{d-fast} < T_{d-nominal} < T_{d-slow}. \quad (2.3)$$

Meeting the hold constraint in (2.2) with large clock uncertainty could result in setup violation due to (2.3) since the slowest manifestation of the same path could violate

the delay requirement in (2.1). Such two-sided constraints are not uncommon in modern design if the clock uncertainty is high.

Central to the discussion above is the clock uncertainty defined by the absolute difference of delays T_{Ck1} and T_{Ck2} . A typical clock distribution structure (Fig. 2.6) relies on buffer² stages to amplify the clock from the clock generator to the respective receivers (shown as the sequential elements FF in Fig. 2.6). In general, when measuring the clock arrival time at the end points of a clock distribution, the clock latencies with respect to the distribution common point (T_{DELAY}) will exhibit a statistical distribution as shown in Fig. 2.6. This statistical distribution is attributed to various static or dynamic sources. For example, design mismatches and on-die process variations will result in static delay mismatches. Clock generator (e.g. PLL) jitter or dynamic voltage variations can introduce dynamic clock uncertainties. Minimizing T_{DELAY} will also minimize clock uncertainty and improve setup and hold margins.

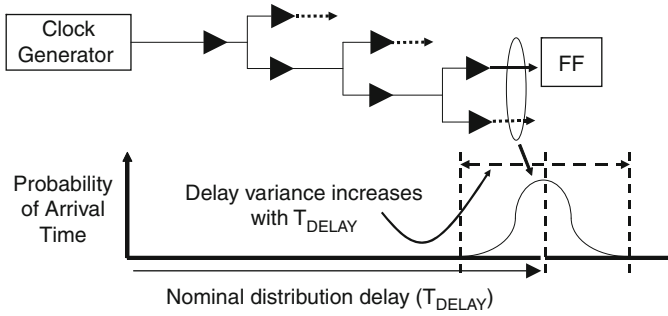


Fig. 2.6. Statistical nature of clock arrival times

2.2.2 Clock Attributes

We use this term to denote clock characteristics that affect the timing constraints described in Sect. 2.2.1. The key attributes are:

1. Clock uncertainties (skew and jitter)
2. Clock distribution latency
3. Clock duty cycle

The first and the last play an explicit role in the timing constraints of a synchronous design. The clock latency by itself does not affect the sequential timing constraints but plays a critical role in determining the other two.

² In this context, a buffer stage could be any gate that exhibits gain.

Static and Dynamic Clock Uncertainties

Clock uncertainties can be classified as static or dynamic. Static uncertainty does not vary or varies very slowly with time. Process variation induced clock uncertainty is such an example. On the other hand, dynamic uncertainty varies with time. Dynamic power supply induced delay variation is an example of a dynamic uncertainty.

In Fig. 2.7, the clock attributes T_{skew} and T_{jitter} are defined on clock waves Ck1 and Ck2. Taking the wave Ck1 as an example, when one of the clock edges is repeatedly sampled with an ideal reference, a timing histogram will result. A timing histogram exists for every clock edge and is characterized by a mean value and a peak-to-peak range (Fig. 2.7). The difference between the mean of two corresponding clock edges (example: between edge A and edge B) is defined as skew (T_{skew}) and is treated as a static uncertainty. The peak-to-peak range of a single edge is specified as the jitter (T_{jitter}) and its character is dynamic.

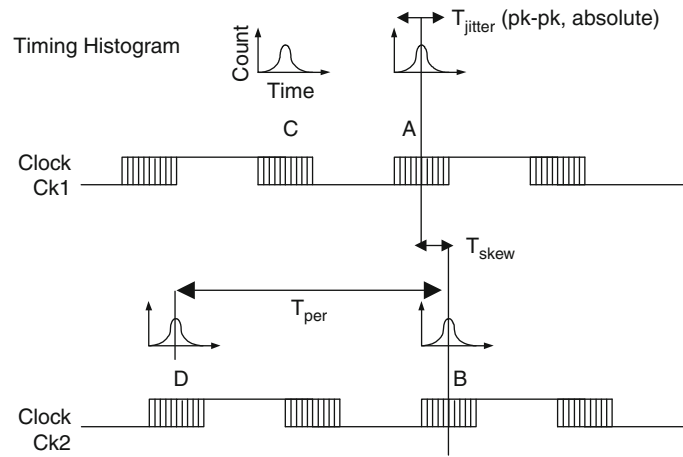


Fig. 2.7. Clock skew and jitter definitions

Table 2.1 highlights the sources of the static and dynamic clock uncertainties. Sources of static clock uncertainties are:

1. Intentional or unintentional design mismatches
2. On-die process variations
3. Loading variations (mismatch) at the intermediate or final stage of the clock distribution

Design mismatches arise because of a number of factors. For example, a nonbalanced clock distribution may be necessary due to floorplan constraints. A poorly chosen distribution topology could lead to structural design mismatches. In other situations, the clock arrival times at certain receivers are intentionally skewed to facilitate time

borrowing across sequential boundaries due to nonuniform data path lengths. On-die device mismatch due to on-die process variations is a significant factor. Additionally, nonuniform clock loading is common in highly integrated designs. All skew sources mentioned above remain constant over time (except through the slow process of transistor aging) and are treated as static. Figure 2.8 shows an empirical breakdown of skew contributors.

Table 2.1. Sources of static and dynamic clock uncertainties

Clock uncertainties	Sources
Static (skew)	Intentional or unintentional design mismatches On-die process variations Final or intermediate loading variations
Dynamic (jitter)	Voltage droop and dynamic voltage variations Temperature gradient due to activity variations Clock generator jitter

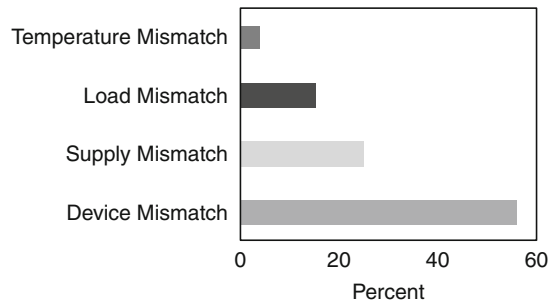


Fig. 2.8. Factors affecting clock skew. Among all the sources, device mismatch is the dominant contributor. Reproduced with permission from [31], ©1998 IEEE

Inherently static clock uncertainties can be corrected either by careful pre-silicon analysis and design or post-silicon adaptive compensation. Accurate pre-silicon analysis can be time consuming and iterative. On the other hand, post-silicon adaptive compensation is flexible and significantly more suited to high volume manufacturing.

Figure 2.9 shows clock skew as a percentage of cycle time vs. processor frequency for a number of recent designs. On average, the trend suggests that the skew as a fraction of the clock cycle time stays at about 4.5–5%. The ability of the trend to continue is attributed to the adoption of clock distribution topologies that are skew tolerant, more robust design flow, and more importantly the incorporation of robust post-silicon compensation techniques.

Clock uncertainties caused by voltage variation, temperature variation, and clock generator jitter are dynamic in nature. We use the term jitter to encompass all

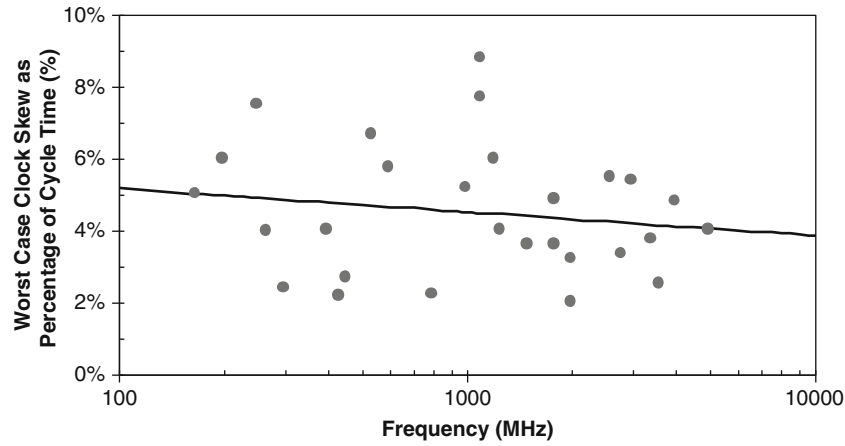


Fig. 2.9. Clock skew as percentage of cycle time vs. processor frequency [8, 15–17, 19–21, 24–26, 28–30, 32–46]

dynamic uncertainties. Voltage variation is the dominant source and it can be due to local switching events affecting specific areas of the clock distribution in a nonuniform fashion. A mathematical model of supply-induced jitter based on additive sinusoidal supply noise is developed in Section 6.8. Global voltage droop and clock generator jitter are common to the entire distribution and contribute to the setup constraint by modulating the cycle time. Clock generator jitter is addressed in detail in Chapter 5. Temperature variation has a long time constant and its impact is usually minor as seen in Fig. 2.8. Figure 2.10 shows the trend of peak-to-peak clock jitter as a fraction of cycle time. The average reduction of effective clock cycle time due to jitter is about 5.5% and minimizing this is critical for performance reasons.

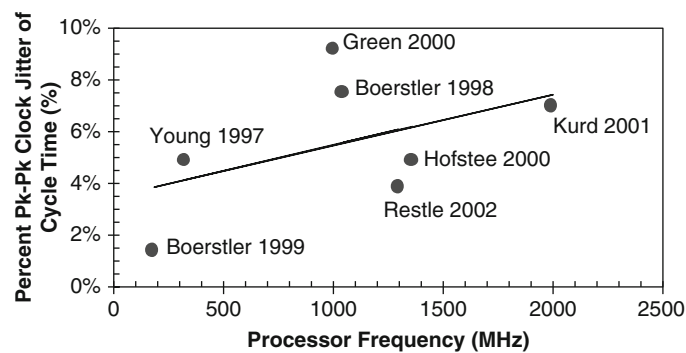


Fig. 2.10. Pk–pk clock jitter as a fraction of clock cycle time vs. processor frequency [6, 17, 18, 36, 47–49]

We now present a simple mathematical skew and jitter model based on clock distribution latency (Fig. 2.11). The figure shows a source clock path with M buffer stages and a receiver clock path with N buffer stages resulting in delays of T_{Ck1} and T_{Ck2} respectively. The point-of-divergence (POD) delay is defined as the sum of the source clock delay and the receiver clock delay measured from a common origin. In Fig. 2.11, the POD delay equals the sum of T_{Ck1} and T_{Ck2} . Assuming T_i is the actual delay at buffer stage i and τ is the average delay per stage, the source clock and the receiver clock delays are:

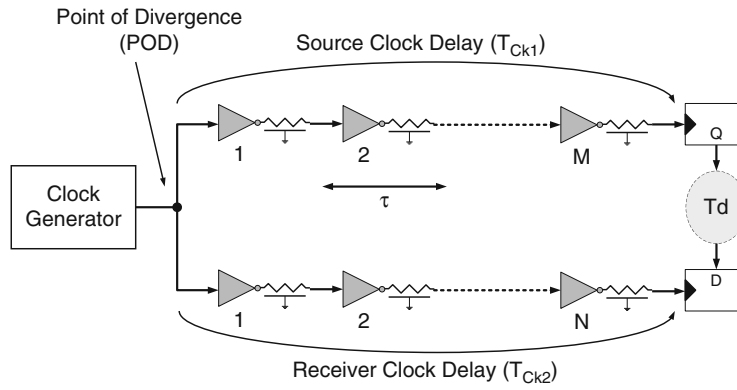


Fig. 2.11. Sample clock distribution for skew and jitter model

$$T_{\text{Ck1}} = \sum_{i=1}^M T_i \cong M\tau, \quad (2.4)$$

$$T_{\text{Ck2}} = \sum_{i=1}^N T_i \cong N\tau. \quad (2.5)$$

The average delay per stage is determined by the drive current of the driver stage (I_d), the output loading capacitance (C_l), and the output voltage swing (V_{CC}):

$$\tau = \frac{C_l V_{\text{CC}}}{I_d}. \quad (2.6)$$

Using a simple linearized model, the change in delay per stage ($\Delta\tau$) can be formulated as:

$$\Delta\tau = \frac{\partial\tau}{\partial V_{\text{CC}}} \Delta V_{\text{CC}} + \frac{\partial\tau}{\partial C_l} \Delta C_l + \frac{\partial\tau}{\partial I_d} \Delta I_d. \quad (2.7)$$

Evaluating the partial derivatives yields:

$$\Delta\tau = \frac{C_l}{I_d} \Delta V_{\text{CC}} + \frac{V_{\text{CC}}}{I_d} \Delta C_l - \frac{C_l V_{\text{CC}}}{I_d^2} \Delta I_d. \quad (2.8)$$

Finally, substituting the expression for τ (2.6) results in the following expression:

$$\Delta \tau = \tau \left(\frac{\Delta V_{CC}}{V_{CC}} + \frac{\Delta C_1}{C_1} - \frac{\Delta I_d}{I_d} \right). \quad (2.9)$$

Equation 2.9 states that the delay variation at each stage can be approximated to be proportional to the stage delay. Additionally, we make the assumption that the delay per stage is a random variable, normally distributed with the following standard deviation:

$$\sigma(\tau) \cong \alpha \tau, \quad (2.10)$$

where α is the proportionality constant predicted by (2.9). We assume that α is on the order of 5%. Under the assumption that each stage delay is independent and identically distributed, the standard deviations of T_{Ck1} , T_{Ck2} , and $|T_{Ck1} - T_{Ck2}|$ become:

$$\sigma(T_{Ck1}) \cong \sqrt{M} \alpha \tau, \quad (2.11)$$

$$\sigma(T_{Ck2}) \cong \sqrt{N} \alpha \tau, \quad (2.12)$$

$$\sigma(|T_{Ck1} - T_{Ck2}|) \cong (\sqrt{M+N}) \alpha \tau. \quad (2.13)$$

The standard deviations of skew and jitter are therefore³:

$$\sigma[T_{\text{skew}}(Ck1, Ck2)] = (\sqrt{M+N}) \alpha_{\text{skew}} \tau, \quad (2.14)$$

$$\sigma[T_{\text{jitter}}(Ck1)] = (\sqrt{M}) \alpha_{\text{jitter}} \tau, \quad (2.15)$$

$$\sigma[T_{\text{jitter}}(Ck2)] = (\sqrt{N}) \alpha_{\text{jitter}} \tau. \quad (2.16)$$

where α_{skew} and α_{jitter} represent the variation coefficients for static and dynamic clock uncertainties, respectively. Typical clock distribution will have $M = N$ and the formulation will be reduced to:

$$\sigma[T_{\text{skew}}(Ck1, Ck2)] = (\sqrt{2M}) \alpha_{\text{skew}} \tau, \quad (2.17)$$

$$\sigma[T_{\text{jitter}}(Ck1)] = (\sqrt{M}) \alpha_{\text{jitter}} \tau, \quad (2.18)$$

$$\sigma[T_{\text{jitter}}(Ck2)] = (\sqrt{M}) \alpha_{\text{jitter}} \tau. \quad (2.19)$$

Equations 2.17–2.19 show that the skew and jitter variations will grow as the square-root of the number of distribution buffering stages and linearly with the nominal delay per stage. This formulation can be generalized for any pair of clocks that share a common point of clock divergence. The sum of the variation coefficients for modern process technology and design is between 5 and 10%.

³ $T_{\text{skew}}(Ck1, Ck2)$ means the skew between clocks Ck1 and Ck2 and $T_{\text{jitter}}(Ck\#)$ is the jitter of Ck#.

Distribution Delay

Equations 2.17–2.19 suggest that the clock distribution delay (latency) is a key component in determining the overall clock uncertainties. In order to handle the final clock loading and to traverse the distances needed to reach the loads, a clock network has to rely on a series of clock buffers for gain and signal propagation. In a typical processor, the number of clock buffer⁴ stages may exceed 20 resulting in clock latency that can approach 1 ns. Minimizing clock distribution latency is a primary design objective irrespective of the distribution topology.

Duty Cycle

Duty cycle (Fig. 2.12) is the relative percentage of the clock high phase time vs. low phase time. Except for special clocking applications such as pulse generators and clocks for dynamic and memory circuits, a 50% clock duty cycle is considered optimal. This is particularly important for a latch-based designs and memory circuits where any offset between the high phase and low phase can lead to phase paths that are more difficult to meet timing constraints. In a phase path, time lost due to duty cycle distortion will subtract directly from the total available phase time. Cycle-based sequential designs using edge-triggered flip-flops are more immune to clock duty cycle distortion. In a clock distribution, duty cycle distortion is introduced when there is asymmetry between rising and falling edge delays.

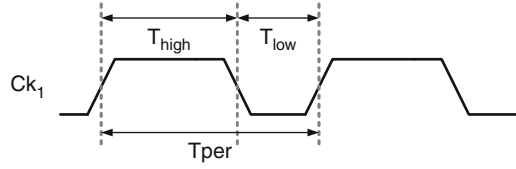


Fig. 2.12. Clock duty cycle

2.2.3 Clock Distribution Power

Power dissipation attributed to the clock distribution has emerged as a critical constraint in multi-GHz multicore processors with large on-die caches. In this section, we will develop a model for clock distribution power. Typically, the total end-of-distribution loading due to sequential elements strongly determines the overall clock network power. Let us consider the switching power of an unconditional clock at the final distribution stage M :

$$Pd_{CK_M} = C_{LM} V_{CC}^2 f, \quad (2.20)$$

⁴ In this context, a clock buffer stage represents one unit-inverter stage.

where $C_{L,M}$ is the stage load that encompasses both gates and interconnect, V_{CC} is the power supply voltage, and f is the clock frequency. Total power will include components for short circuit and leakage, but the dynamic component dominates. If the fan out per stage is k , the clock dynamic power consumed at stage $M - 1$ is:

$$Pd_{Ck,M-1} = C_{L,M-1} V_{CC}^2 f = \frac{C_{L,M}}{k} V_{CC}^2 f. \quad (2.21)$$

Assuming that the fan out is constant across all M stages, the total clock distribution dynamic power is:

$$Pd_{Ck,Total} = \sum_{i=1}^M Pd_{Ck,i} = \sum_{i=1}^M C_{L,i} V_{CC}^2 f, \quad (2.22)$$

$$Pd_{Ck,Total} = V_{CC}^2 f C_{L,M} \left[\frac{1 - \left(\frac{1}{k}\right)^M}{1 - \frac{1}{k}} \right]. \quad (2.23)$$

Let us define the clock load multiplier as the ratio of the total clock distribution load capacitance to the end-of-distribution load capacitance:

$$\text{Clock load multiplier} = \frac{\sum_{i=1}^M C_{L,i}}{C_{L,M}} = \left[\frac{1 - \left(\frac{1}{k}\right)^M}{1 - \frac{1}{k}} \right]. \quad (2.24)$$

Figure 2.13 shows the clock load multiplier vs. the number of distribution stages. Decreasing the stage fan out will lead to higher total network capacitance that approaches 1.5 at a stage fan out of 3. Note that this parameter is not very sensitive to the number of buffer stages in the clock distribution network. Figure 2.14 shows the

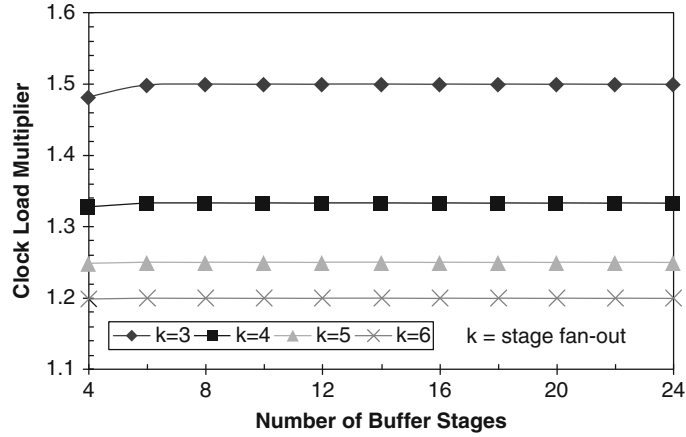


Fig. 2.13. Clock loading multiplier of a clock distribution

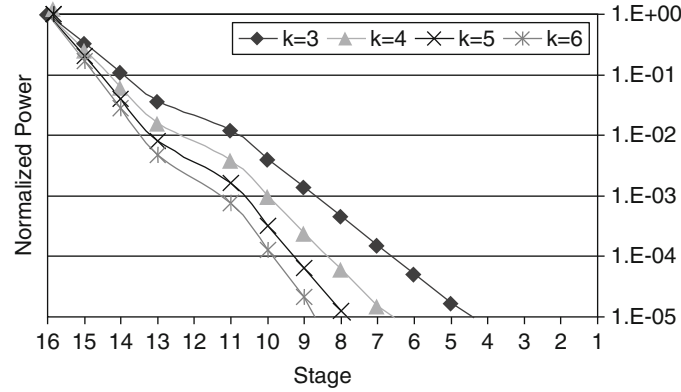


Fig. 2.14. Normalized clock stage power vs. stage number

power dissipation at each of the stage normalized to the power dissipation at the last stage. It can be seen that power dissipation of the clock distribution is dominated by the final end-of-distribution loading and that the last couple of stages in the distribution will account for more than 90% of the total clock power. An implication of this analysis is that the manner in which clock is distributed at the final stages of the distribution will ultimately determine the overall clock power and that the distribution topology upstream will not have a strong impact. In a typical processor, clock distribution (excluding the last stage) should not exceed 10% of total chip power and have a design goal of 5–8%.

2.3 Clock Distribution Topologies

In this section, various clock distribution topologies are described. Table 2.2 lists distribution topologies encountered in modern processors. While discussing these topologies, we will focus on the same attributes described in Sect. 2.2.2. In addition, ease of implementation will be considered. Ease of implementation is subjective and depends heavily on historical design styles and prior art.

2.3.1 Unconstrained Tree

An unconstrained tree style clock distribution is illustrated in Fig. 2.15. It is commonly used in automatic synthesis flows and usually placed with little or no restriction on the number of buffer stages and explicit matching between interconnect delays and the buffer delays. The network design is accomplished with a cost function that minimizes the delay differences across all clock branches. Figure 2.15

Table 2.2. Clock distribution topologies

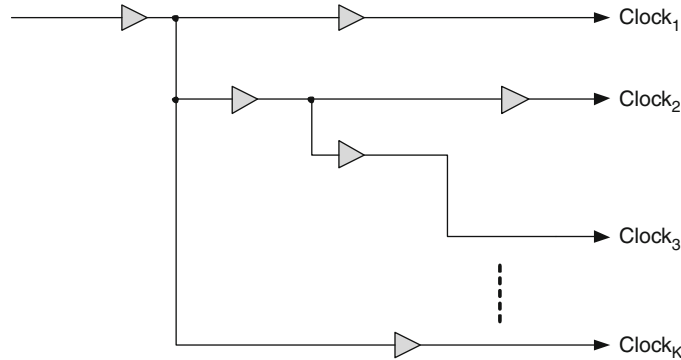
Style	Description
Unconstrained tree	Automated buffer placements with unconstrained trees
Balanced tree	Multiple levels of balanced tree segments H-tree is most common
Central spine	Central clock driver
Spines with matched branches	Multiple central structures with length (or delay) matched branches
Grid	Interconnected (shorted) clock structure
Hybrid distribution	Combination of multiple techniques Common theme is tree + grid or spine + grid

shows an unconstrained clock distribution tree with K branches. A cost function (ϑ) that minimizes the delay differences can be used in the construction of the network:

$$\vartheta = \sum_{i=1}^K (T_{Cki} - T_{Ck_Average})^2, \quad (2.25)$$

$$T_{Ck_Average} = \frac{1}{N} \sum_{i=1}^K T_{Cki}. \quad (2.26)$$

In the primitive form and specifically without explicit structural matching, a clock network with dissimilar buffer and interconnect delay composition may result in radically different branches that will exhibit significant mis-tracking across process, voltage, and temperature variations. More sophisticated optimizing algorithms can be incorporated to improve PVT tracking. Due to this limitation, this style of clock distribution is usually restricted to small functional blocks within a larger design.

**Fig. 2.15.** Unconstrained tree clock network

2.3.2 Balanced Tree

Figure 2.16 shows a balanced H-tree clock topology. Due to the structural symmetry, a balanced tree exhibits identical nominal delay and identical buffer and interconnect segments from the root of the distribution to all branches. If the matching is adhered to, structural skew can be zero. With identical buffer and interconnect segments, an idealized balanced tree clock distribution will exhibit good tracking across PVT compared to the unconstrained network described earlier.

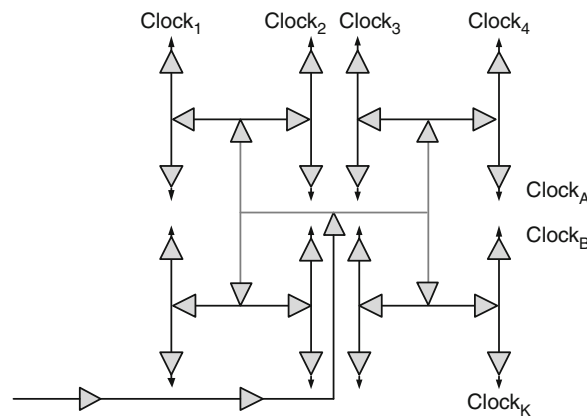


Fig. 2.16. Balanced H-tree clock network

Figure 2.17 shows alternative balanced tree topologies: the X-tree and a tapered H-tree. The X-tree incorporates nonrectilinear clock trunks in the physical implementation but exhibits the same properties as the H-tree. The trunk widths in a tapered H-tree increase geometrically toward the root of the distribution to maintain impedance matching at the T-junctions. One important characteristic of the aforementioned tree structures is that by continuing to expand the buffer hierarchy, balanced trees are capable of delivering the clock to all part of the silicon die. Typically, the clocks at the end-of-distribution branches will serve a small local region. The size and number of the local regions will determine the depth of the tree. A larger number of regions requires a tree with more depth.

Full balanced tree topologies are designed to span the entire die in both the horizontal and vertical dimensions. They are capable of delivering the clock to all regions of the die. A binary tree on the other hand (Fig. 2.18) is intended to deliver the clock in a balanced manner in either the vertical or horizontal dimension.

All branches of a binary tree exhibit identical buffer-interconnect segments, zero structural skew, and similar PVT tracking. In contrast to the H-tree, the buffers in a binary tree can be designed to co-locate in close proximity along a centralized stripe. The closer physical proximity of the buffers in a binary tree can result in

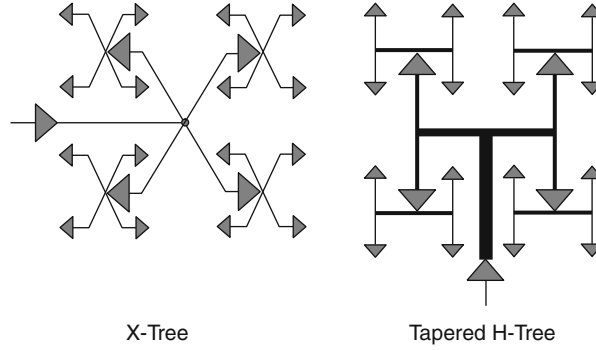


Fig. 2.17. Variations on the balanced tree topology

reduced sensitivity to on-die variation. Moreover, physical placement of the clock buffers in close proximity will minimize floorplan disruptions. On the other hand, the idealized buffer placements associated with an H-tree may be difficult to achieve. Due to these reasons, binary trees are often the preferred structure over an idealized H-tree. Figure 2.19 shows a binary tree distribution with intermediate shorting. The benefits of shorting will be discussed in a later section.

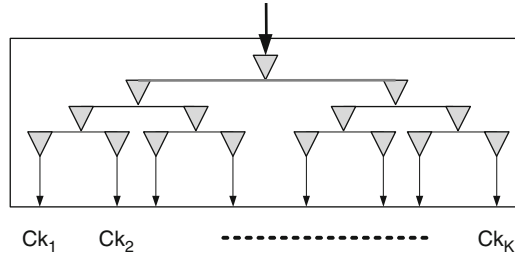


Fig. 2.18. Binary tree clock distribution

A balanced tree will exhibit nonzero clock uncertainty among branches due to nonzero POD delays (2.17–2.19). Let us consider branches Clock_4 and Clock_A in Fig. 2.16. The point-of-divergence is two buffer-interconnect segments apart. In contrast, branches Clock_A and Clock_B are six buffer-interconnect segments apart resulting in higher POD delay and higher skew uncertainty. Therefore, among pairs of equivalent branches in a balanced tree, nonuniform skew uncertainty will result and will depend on point-of-divergence delay. In summary, a balanced tree is capable of delivering the clock from the root to all regions of the die with good structural

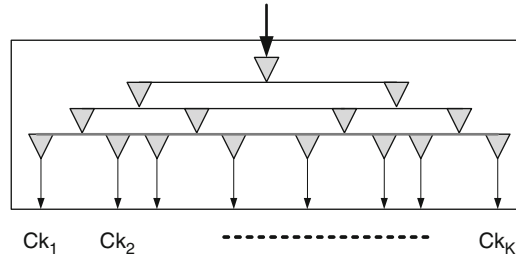


Fig. 2.19. Binary tree clock distribution with intermediate shorting

matching, efficient clock power and low structural latency. On the other hand, it will exhibit nonuniform POD-induced clock skew uncertainty.

Before proceeding it should be noted that a nonsymmetric tree can also be used in this context. A nonsymmetric tree usually maintains the same number of buffer stages but will not have delay matching on a per stage basis. Delay adjustment for overall branch equalization is done in a fashion similar to the unconstrained tree. Intense computational effort usually is needed for this design and its application is less common.

2.3.3 Central Spine

A central spine clock distribution is a specific implementation of a binary tree. Figure 2.20 shows an idealized central spine implementation with the final branches serving all parts of the die. The binary tree is shown to have embedded shorting at all distribution levels and unconstrained routing to the local loads at the final branches. In this configuration, the clock can be transported in a balanced fashion across one dimension of the die with low structural skew. The unconstrained branches are simple to implement although there will be residual skew due to asymmetry (Fig. 2.20).

2.3.4 Spines with Matched Branches

An extension of the central spine structure can be realized by replacing the unconstrained end-of-distribution branches with delay matched routes as shown in Fig. 2.21. In this implementation, the longest branch determines the delay from the output of the central spine to the end loads. Serpentine routes are added to the shorter branches for delay matching. Figure 2.21 shows a structure with three central spines. Multiple central spines are needed when the routing distance of the local branches is increased. Dividing the chip into several sectors served by multiple spines is a practical topology to ensure small local branch delays.

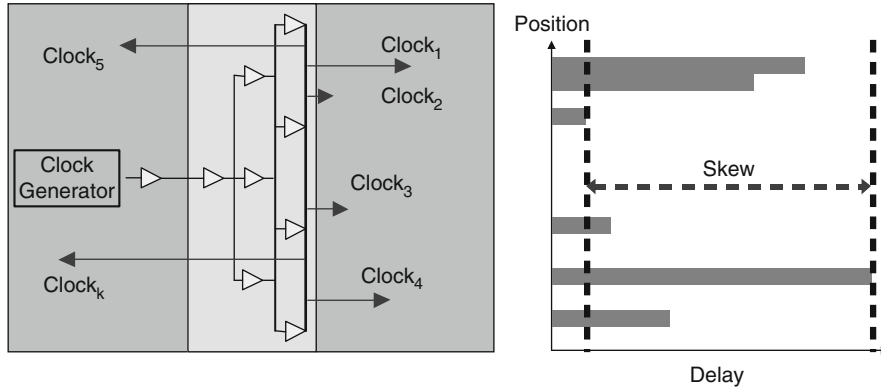


Fig. 2.20. Central clock spine distribution

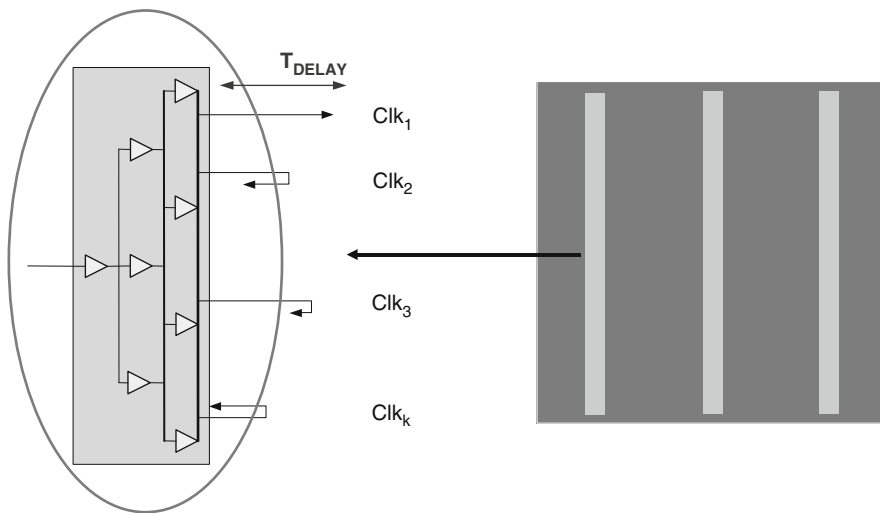


Fig. 2.21. Multiple clock spines with matched branches

2.3.5 Grid

The tree style distributions described in previous sections rely on individual branches to deliver the clock to the local (end-of-distribution) points of clock consumption (e.g. local flip-flops). A processor will have a large number of these local points and will require a large number of branches and therefore a deep distribution tree. A deep distribution tree will exhibit large POD delays and degraded clock performance. Subdividing the die into a smaller number of clock regions and applying a grid to serve each region can be a superior solution.

Figure 2.22 schematically shows a 2-dimensional grid serving one of these clock regions. This clock grid resembles a mesh with fully connected clock tracks in both dimensions and grid drivers located on all four sides. Local loads within a region can be directly connected to the grid. The grid effectively shorts the output of all drivers and helps minimize delay mismatches. Figure 2.22 shows an idealized delay profile of a 2-dimensional grid assuming uniform loading. The shorted grid node helps balance the load nonuniformities and results in a more gradual delay profile across the region. Additionally, since the grid drivers are shorted, the POD delay to all the loads within a region is limited to the interconnect delay of the grid which is typically small and results in lower clock skew uncertainty across the region. Grid drivers may also be placed on two sides leading to a structure and delay profile shown in Fig. 2.23. Critical design parameters for the grid are grid driver locations and pitch in addition to the grid metal pitches and dimensions.

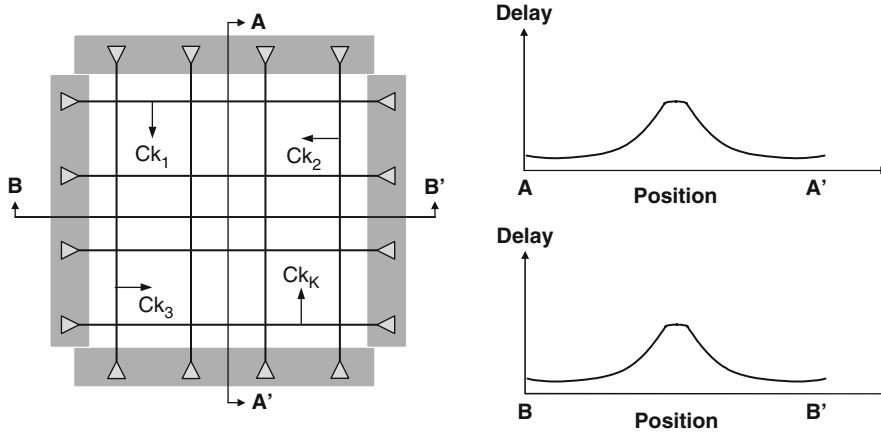


Fig. 2.22. Clock grid with 2-dimensional clock drivers

The recombinant tile structure is an enhancement over the conventional grid structure and incorporates the properties of a balanced tree [29]. Figure 2.24 shows the evolution of the recombinant tile structure from a regular H-tree segment to a tile template and to the final tile assembly. A typical implementation will have uniform interconnect pitches in both the x and y dimensions.⁵ The pitches are determined by the intrinsic interconnect segment delays and the edge rate requirements. The uniformity of the segment pitches allows all intermediate buffers to be placed in predetermined locations.

The following analysis highlights the benefits of shorting intermediate stages in reducing clock uncertainties. Figure 2.25 shows two clock branches Ck_A and Ck_B exhibiting input skew $skew_{IN}$. Let us assume that the two branches are identical

⁵ Note that the x and y pitches do not need to be equal.

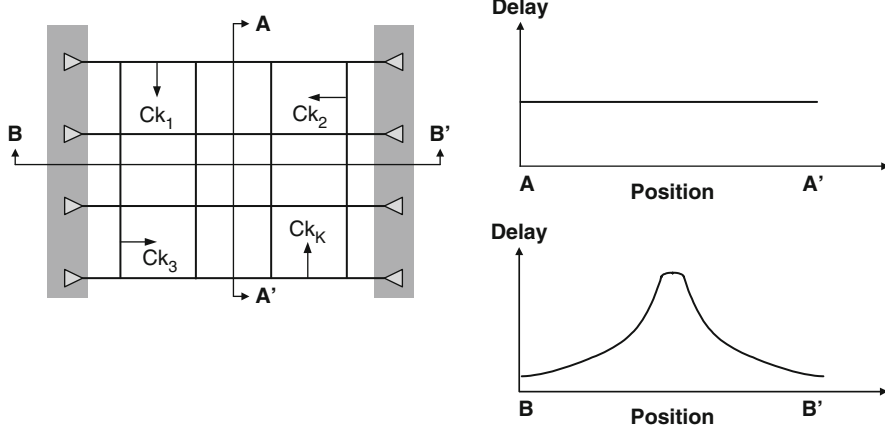


Fig. 2.23. Clock grid with 1-dimensional drivers

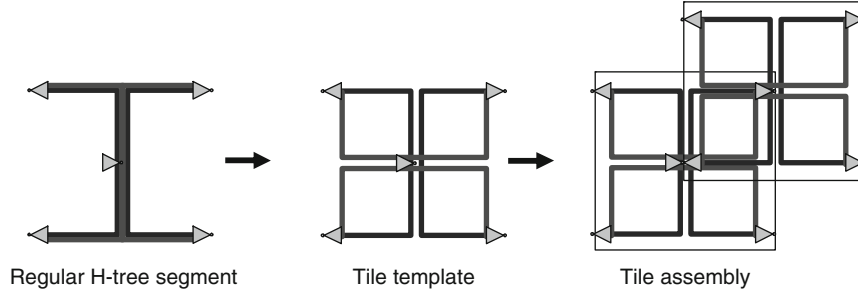


Fig. 2.24. Recombinant tile clock structure. Reproduced with permission from [29], ©2003 IEEE

(A-X-Y and B-U-V) and that R_{SHORT} in Fig. 2.25 is an open circuit ($R_{\text{SHORT}} = \infty$). In this case, the skew between Ck_U and Ck_Y will be the same as the skew between Ck_A and Ck_B . With an ideal short circuit ($R_{\text{SHORT}} = 0\Omega$), the skew between Ck_U and Ck_Y will be zero. Hence, with a non zero and finite R_{SHORT} , the output skew between Ck_U and Ck_Y will be proportional to the incoming skew:

$$\text{skew}(\text{Ck}_U, \text{Ck}_Y) = \gamma \text{skew}(\text{Ck}_A, \text{Ck}_B), \quad (2.27)$$

where γ is a skew averaging coefficient ($\gamma \geq 0$). The averaging coefficient is dependent upon the local POD induced delay mismatch and will scale with the spatial separation between the output nodes. For example, when the shorting resistor R_{SHORT} in Fig. 2.25 exhibits near zero resistance, the skew between Ck_Y and Ck_U will be close to zero suggesting that γ is near 0. As R_{SHORT} approaches an open, γ will be equal to or larger than 1. For example, in a 90nm technology, a typical R_{SHORT} will

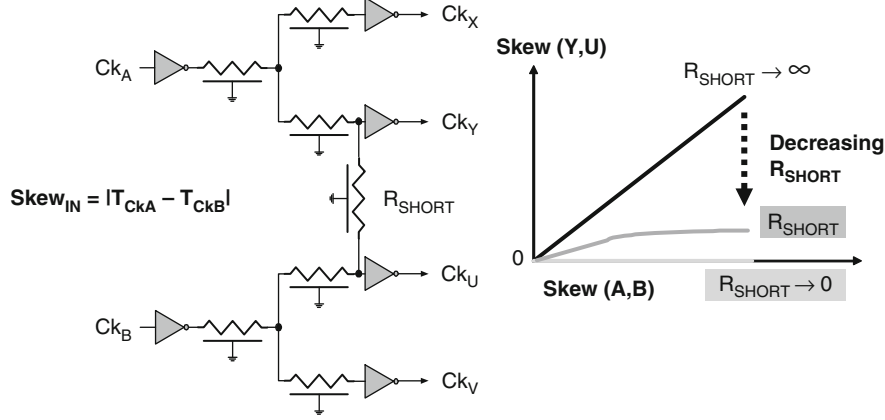


Fig. 2.25. Effect of shorting on clock skew [50]

result in γ in the range of 0.2–0.8 for a few hundred microns of spatial separation [29]. When this structure is cascaded in a clock distribution consisting of S cascaded stages ($S > 20$), the skew at the output of stage S will be (for small γ) [50]:

$$\text{skew}_S \approx \gamma^2(\text{skew}_{\text{IN}}) + \gamma(\text{skew}_{\text{IN}}). \quad (2.28)$$

Equation 2.28 states that the skew at the output of the distribution will stay relatively independent of the number of stages and therefore less sensitive to die size. The recombinant tile structure can be easily scaled to accommodate new designs. Comparing the recombinant tile of Fig. 2.24 to the grid structures of Figs. 2.22 and 2.23, reveals that they share similar skew benefits due to averaging. However, the grid structures require a pregrid clock distribution network to drive the grid or collection of grids. A hybrid clock distribution topology can meet this need.

2.3.6 Hybrid Distribution

A hybrid clock distribution incorporates a combination of earlier described topologies. Common configurations are spines-grid distribution or tree-grid distribution. Figure 2.26 shows the topology of a tree-grid distribution. It employs a multilevel H-tree driving a common grid. Specifically, the multilevel H-tree delivers the clock from the clock generator (PLL in Fig. 2.26) to various regions of the die. Regional buffers (labeled as level 4 buffers in Fig. 2.26) residing at the end of the multilevel H-tree drive a common grid that includes all local loads. As an alternative, there can be multiple regional grids each served by a branch of the pre-grid H-tree. Partitioning the design enables intentional skew rebalancing across the regions.

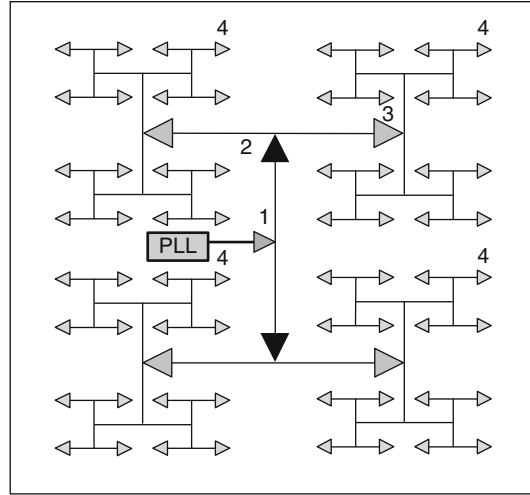


Fig. 2.26. Hybrid clock distribution consisting of balanced H-Tree and Grid

2.4 Microprocessor Clock Distributions

Due to the design complexity and the significant mis-tracking to process, voltage and temperature, a fully unconstrained clock distribution network is rarely (if ever) applied to a processor design.

The closest example is a hybrid combination of symmetric and asymmetric clock trees. Figure 2.27 shows an example of a processor clock distribution with a first level H-tree connected to multiple secondary trees that are asymmetric but delay balanced.

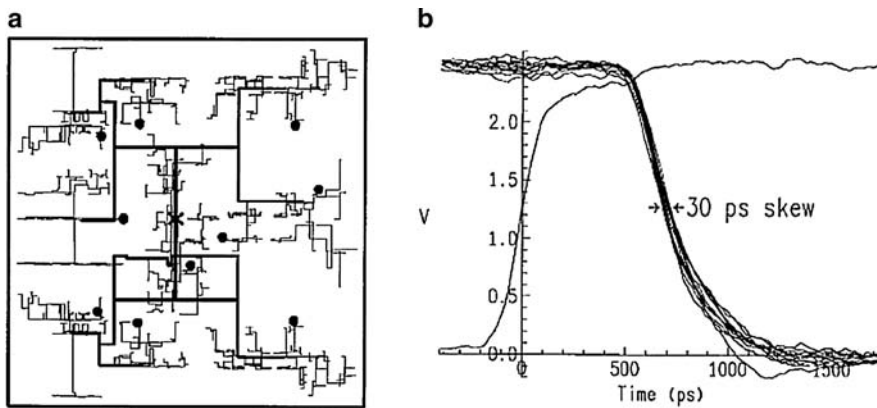


Fig. 2.27. Asymmetric clock tree distribution network based on delay matching. Reproduced with permission from [51], ©1998 IEEE

The construction of the trees was based on a custom methodology that matches the wire delays by tuning the metal widths, spacings, and lengths. Specifically, the first level symmetric H-tree (thick lines in Fig. 2.27a) routes the global clock from the center of the die to 9 sector buffers. The 9 sector buffers rely on multiple delay-matched secondary trees (light lines in Fig. 2.27a) to distribute the clock to 580 global clock receivers. Figure 2.27b shows the measured skew.

Figure 2.28 shows another example of a hybrid multilevel clock tree design [52]. The cache area (un-core) of the processor is partitioned into 13 regional clock zones served by the secondary clock trees. Postlayout extraction-based simulation models were used to perform tree optimization and delay matching. Additional examples of a hybrid multilevel clock tree for processors are found in [10, 12, 13, 43].

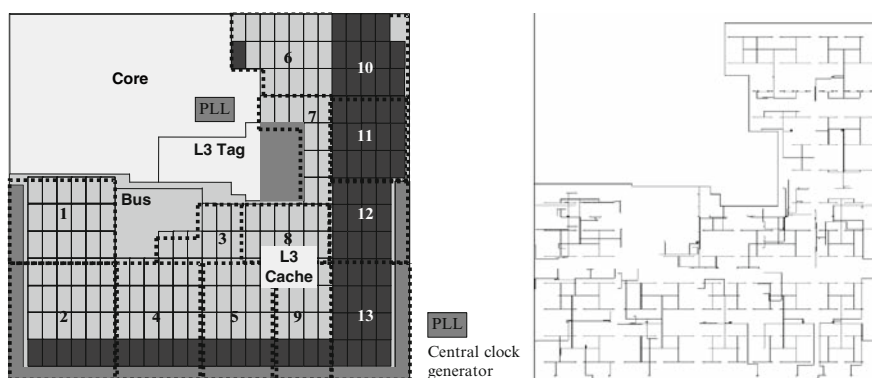


Fig. 2.28. Asymmetric clock tree distribution with multiple regions. Reproduced with permission from [52], ©2005 IEEE

Hybrid clock distributions that consist of multilevel symmetric trees and grids have been applied to a number of processors [17–20]. Figure 2.29 shows an example [18]. In this implementation, the clock from the clock generator is distributed from a central clock buffer through two levels of buffering and three levels of delay tuned H-trees before reaching a main grid covering most of the chip, and two smaller grids covering two units that require delayed clocks.

Figure 2.30 shows another example of a hybrid tree-grid design in [17]. A multilevel tree structure delivers the clock to 64 sector buffers driving a common grid via multiple second level tuned trees.

Figure 2.31 shows the floorplan and the clock skew profile of three generations of Alpha®⁶ processors. These designs followed a common strategy of having one or more centralized clock spines to drive a common grid. The first generation design relied on a single spine to support the entire die whereas the third generation design

⁶ Other names and brands may be claimed as property of others.

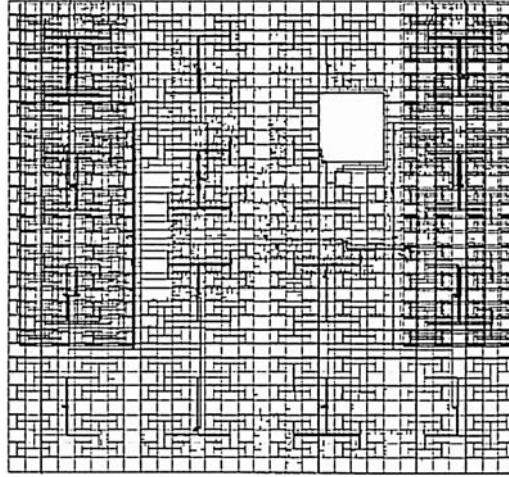


Fig. 2.29. Multilevel symmetric H-Tree distribution. Reproduced with permission from [18], ©2000 IEEE

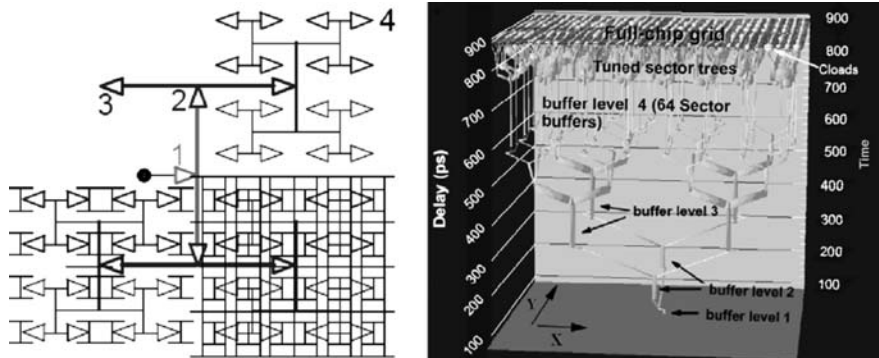


Fig. 2.30. Delay characteristics of a multilevel tree-grid distribution. Reproduced with permission from [17], ©2002 IEEE

utilized 16 central spines organized in a 2-dimensional fashion to drive the common grid. By partitioning the die into smaller regions, the third generation design reduced the clock skew across the grid.

Figure 2.32 shows the application of recombinant tiles to a multi-GHz IA processor fabricated in 90nm [29].⁷ The buffers needed for the recombinant tiles are embedded in eight central clock stripes. The recombinant tile distribution consists of

⁷ Other names and brands may be claimed as property of others.

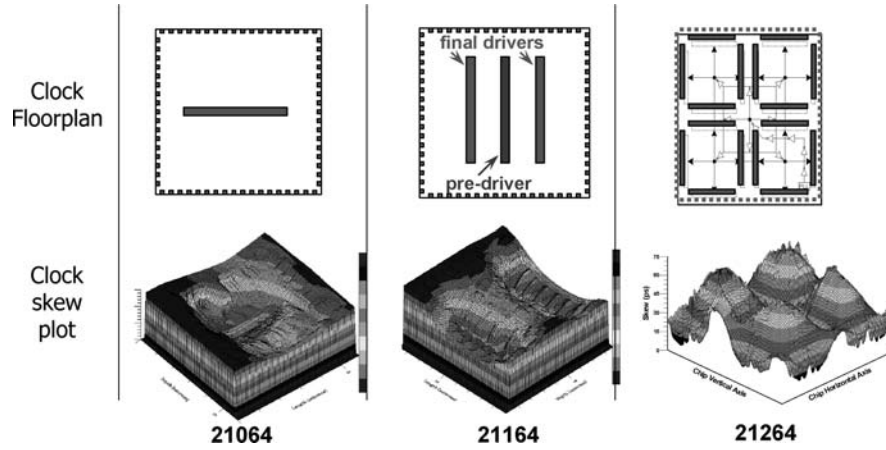


Fig. 2.31. Centralized clock drivers with grids on three generations of the Alpha® microprocessor. Reproduced with permission from [53], ©1998 IEEE

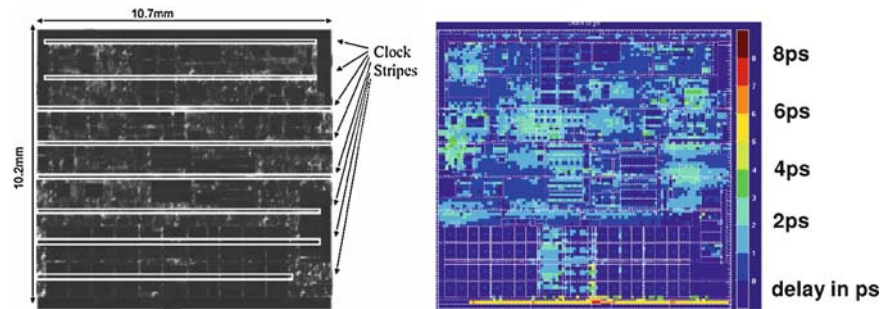


Fig. 2.32. Recombinant clock tiles on a 90nm processor. Reproduced with permission from [50], ©2003 IEEE

27 inversion stages and a total of 1,474 grid drivers (each driver is an inverter). An automated grid driver sizing flow was used to minimize grid driver oversizing for power efficiency. The simulated delay profile is shown in Fig. 2.32. A global skew of less than 10ps was achieved with this design.

An example of centralized spines with delay-matched branches is the clock distribution of the 180nm Pentium® 4 processor [8]⁸. Binary distribution trees embedded in three central clock spines drive the local loads with delay-matched branches. The binary trees embedded in middle spine buffer the clock from the central PLL and deliver it in a balanced fashion to the other spines. The final clock drivers use matched

⁸ Other names and brands may be claimed as property of others.

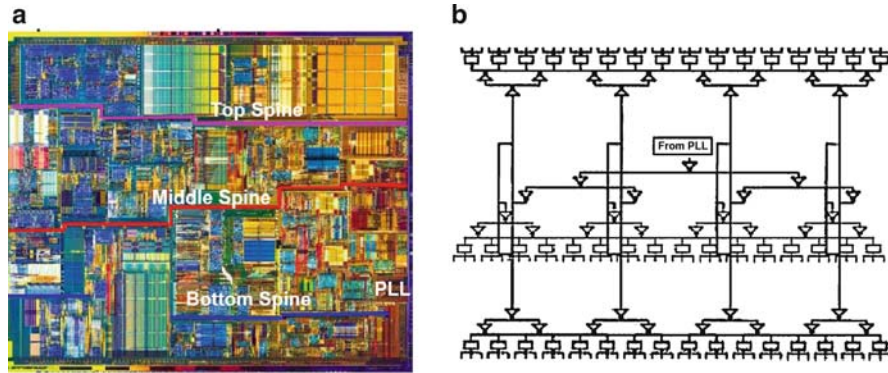


Fig. 2.33. Pentium® 4 processor clock distribution using centralized spines with delay matched final branches. Reproduced with permission from [49], ©2001 IEEE

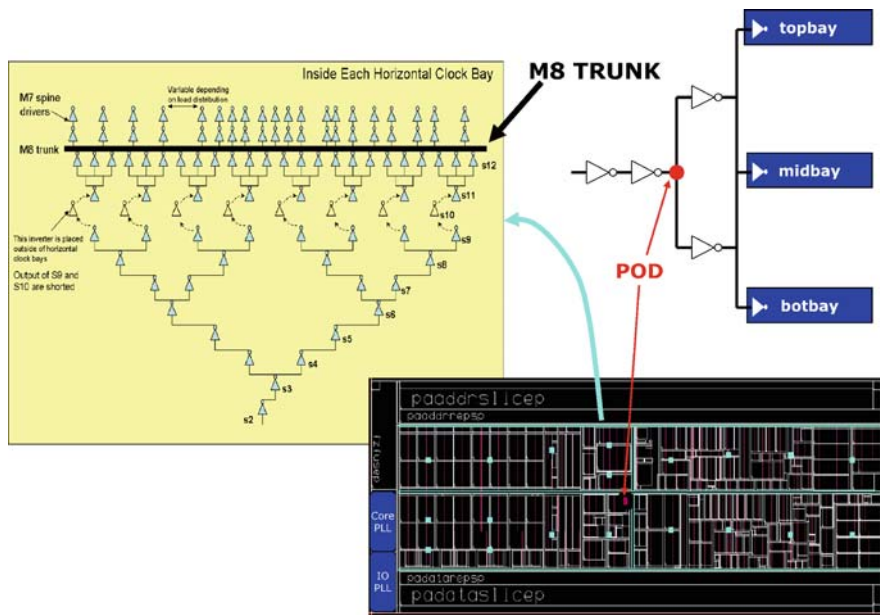


Fig. 2.34. Clock distribution of a low power IA processor consisting of binary trees embedded in the centralized spines. Reproduced with permission from [55], ©2008 IEEE

branches to support the local loads. The final drivers incorporate delay tuning capability to further optimize the skew via post-silicon compensation. Without engaging compensation, the measured skew is $\pm 32\text{ps}$.

The low power IA processor in [54] is another example of the application of the clock distribution topology consisting of centralized spines and delay-matched branches. The global binary tree with limited clock recombination is embedded in the spines. The spine output drivers are shorted with a high layer metal (M8) to equalize the driver delays. The highly selective application of clock recombination and other power saving schemes enables this design to achieve total clock power dissipation that is less than 10% of the total processor power.

The 65nm dual-core Xeon® processor employed a hybrid spine-grid clock distribution topology to support the multicore and uncore clock domains [45].⁹ Figure 2.35 shows the multiple clock domains and the distribution design of this processor. The processor has two cores operating at the high frequency MCLK. The uncore is supported by the SCLK at half the MCLK frequency and the ZCLK dedicated for the I/O circuits at 4 times the system clock frequency. Binary trees embedded in the horizontal and vertical clock spines in the uncore distribute the clocks to the SCLK and ZCLK grids. The core employs the recombinant tile clock distribution similar to that described in [29]. Operational flexibility is achieved by keeping core and uncore clock regions independent.

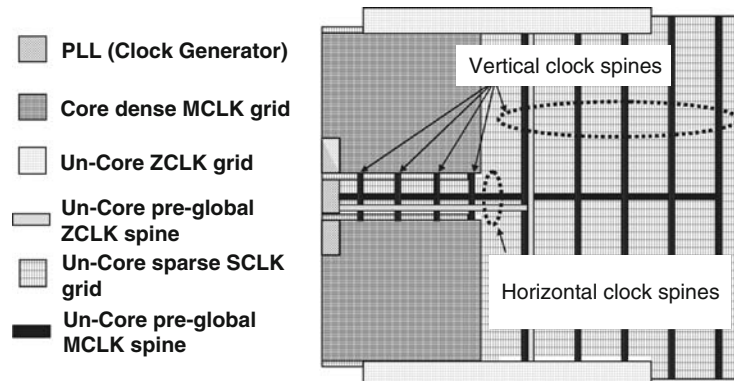


Fig. 2.35. Hybrid spine-grid clock distribution in a dual core processor. Reproduced with permission from [45], ©2006 IEEE

Figure 2.36 shows the details of the uncore grid implementation and the local clocking in [45]. The preglobal clock and the final grid clock driver are embedded in the vertical clock spines. A common SCLK grid covers the entire uncore area to serve the local logic units. Local clocking consists of two buffer stages featuring clock gating and delay tuning. The local clock buffers are placed inside the local logic unit with direct connection to the overlying global grid. Support for multiple local clock flavors is achieved with a family of local clock buffer macros.

⁹ Other names and brands may be claimed as property of others.

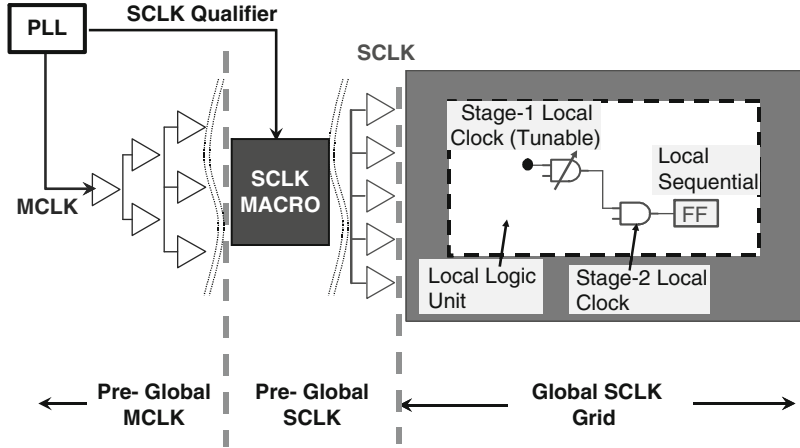


Fig. 2.36. Local clock distribution of the hybrid spine-grid clock distribution. Reproduced with permission from [45], ©2006 IEEE

2.5 Clock Design for Test and Manufacturing

2.5.1 Global and Local Clock Compensations

It should be obvious from the earlier discussions that the primary objective of the aforementioned clock distribution topologies is to deliver the clock to all corners of the die with low skew. For example, the application of averaging in the recombinant tile clock distribution will result in a scalable network that exhibits very low skew. However, implementation of the recombinant tile network will encounter floorplan constraints leading to nonideal driver placements and loss of performance. Floorplan constraints will also affect other topologies such as the H-tree distribution. Moreover, in many situations, intentional clock skew between specific regions of the global clock network is needed to rebalance the path timings. Therefore, a clock distribution network with adaptive delay compensation¹⁰ is superior to the conventional design that has fixed delays, even if the adaptive design may exhibit higher initial skew. Additionally, adaptive delay compensation will adjust to skew caused by process variations and will overcome difficulties related to the construction of a pre-silicon design model and design flow that accurately and exhaustively accounts for all process effects (i.e. SOI dynamic switching effects). As an example, the evolution of the processor clock distribution designs in [17–19] eventually incorporated adaptive clock compensation in the latest implementation [20]. By allowing for slightly high initial skew, the physical design resource needs (e.g. metal tracks, floorplan

¹⁰ The terms “adaptive delay compensation,” “active deskew,” and “skew rebalancing” are used interchangeably in this discussion.

restrictions, pre-silicon analysis, etc.) for a clock network with adaptive compensation are expected to be lower. In the following sections, we discuss adaptive global and local clock compensation architectures.

2.5.2 Global Clock Compensation Architecture

Figure 2.37 shows a dual-zone adaptive deskew clock distribution architecture implemented in a 450MHz microprocessor [32]. The global clock distribution of this processor is partitioned into two planes supported by the “left” and the “right” clock spines. Binary clock trees embedded in the spines deliver the global clock from the clock generator to the spine drivers. Two digital delay lines with 17 delay adjustment steps and approximately 12ps average delay step size reside near the root of their respective clock spines. A phase detector (PD in Fig.2.37) strategically placed in the microprocessor core compares the phase difference of the clocks between the left and the right planes. A digital control logic unit (Control FSM and Delay SR) interprets the phase detector output and makes adjustments to the delay lines. The construction of the delay line is shown in Fig.2.38. It consists of two cascaded inverters with switchable load capacitors at each stage. The delay shift register (Delay SR) generates a thermometer-coded delay adjustment and sequentially enables the load capacitors between the two stages. A 60ps skew was reported with adaptive deskewing disabled and 15ps with the mechanism engaged.

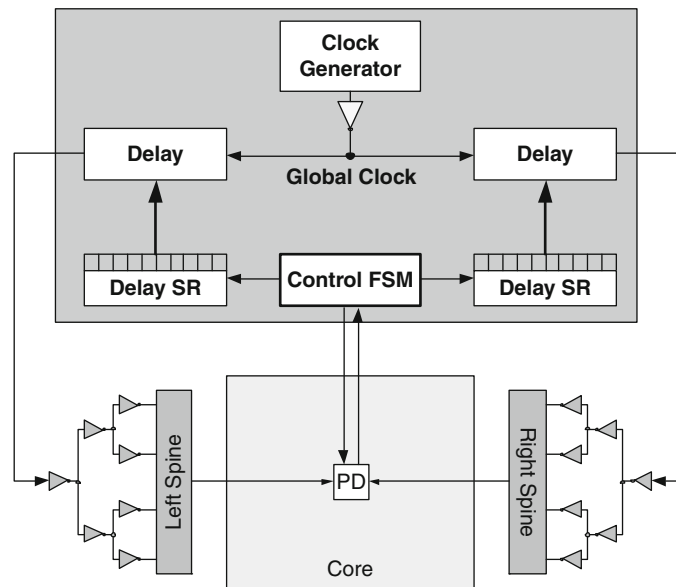


Fig. 2.37. Dual-zone deskew architecture. Reproduced with permission from [32], ©1998 IEEE

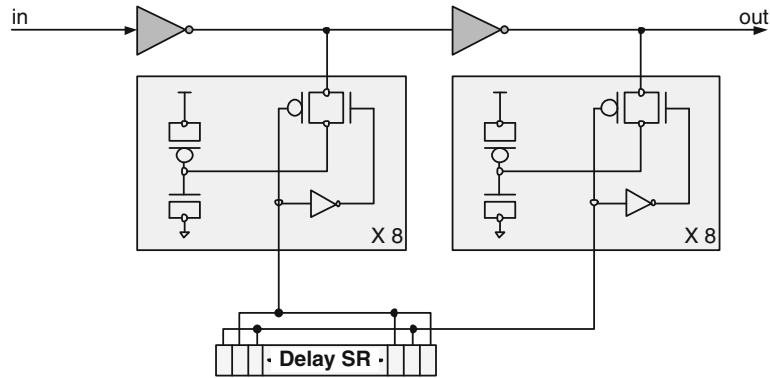


Fig. 2.38. Deskew delay line structure. Reproduced with permission from [32], ©1998 IEEE

The previous implementation [32] embodied only two independent deskew regions. Figure 2.39 shows a deskew architecture in the Itanium® processor¹¹ that supports 30 independent deskew regions [36, 56]. An H-tree distributes the global clock from the central PLL to eight clusters of deskew buffers (DSK in Fig. 2.39) serving 30 independent deskew regions. Each DSK cluster may contain up to four independent deskew buffers.

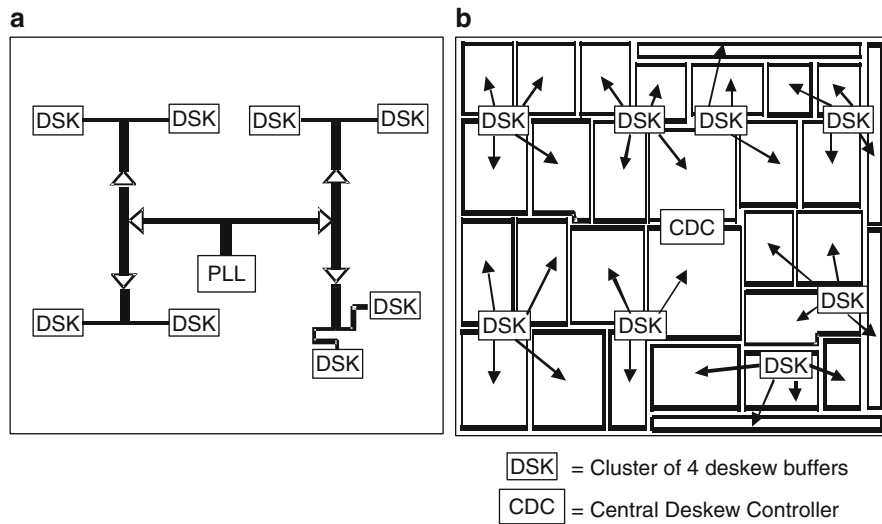


Fig. 2.39. Deskew zones in the itanium® processor. Reproduced with permission from [56], ©2000 IEEE

¹¹ Other names and brands may be claimed as property of others.

Figure 2.40 shows the detailed clock distribution architecture of [36] that encompasses the H-tree global distribution, the grid structure for the regional distribution, and the local buffers for the local distribution. In this design, the grid drivers (RCD in Fig. 2.40) are located at the top and bottom of grid. In addition to the global clock (main clock in Fig. 2.40), a dedicated and tightly matched reference clock is routed from the central clock generator to the eight DSK clusters (reference clock in Fig. 2.40). The purpose of the reference clock is to act as the reference to deskew the global clock.

Figure 2.41 shows the details of the deskew buffer architecture and the delay circuit design. The delay circuit design is similar to the previous design consisting of two inverter stages with switchable capacitor loads using a 20b thermometer code with a tri-state controllable output stage. In this implementation, the total skew is 28ps with deskew turned on. The skew increases by a factor of 4 with the deskew mechanism disabled. Additional details of this active deskew architecture are discussed in Sect. 7.3.2 in the context of addressing variations in the clock network.

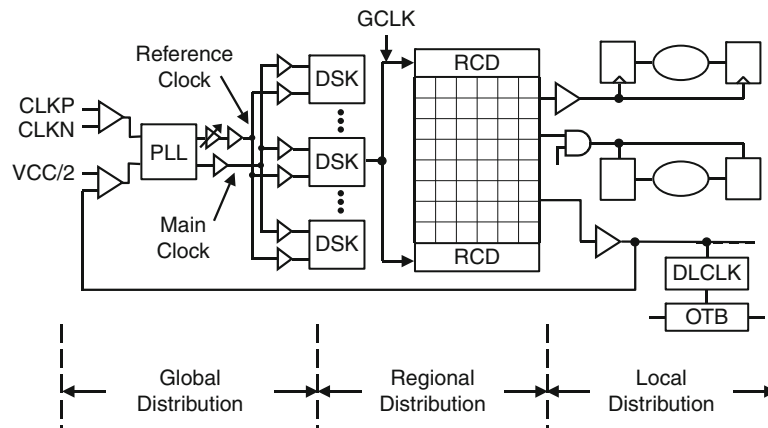


Fig. 2.40. Clocking architecture of the first itanium® processor. Reproduced with permission from [56], ©2000 IEEE

The 180nm Pentium® 4 processor¹² clock distribution described in Sect. 2.4 employs a hierarchical deskew architecture consisting of 47 adjustable clock zones, and 3 levels of deskew hierarchy with 46 phase detectors [49]. The left panel in Fig. 2.42 shows the deskew system architecture whereas the right panel shows the details of the deskew hierarchy. Phase detectors are placed in between the deskew hierarchies. For example, phase detectors between the single primary reference and the secondary references will detect the delay differences between the reference clock zone and the secondary zones. The clock latencies to the secondary zones are adjusted via corresponding deskew buffers (DB2–DB47). Application of hierarchical

¹² Other names and brands may be claimed as property of others.

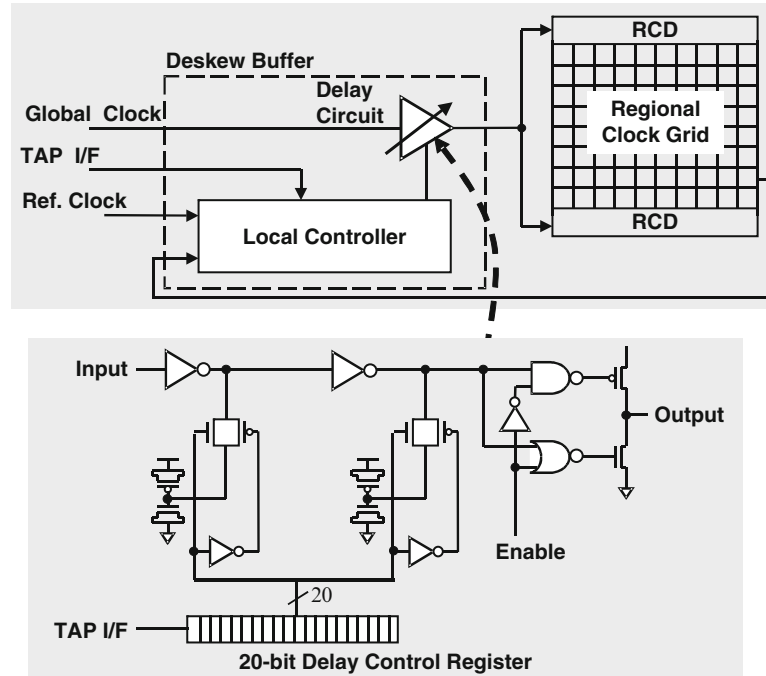


Fig. 2.41. Deskw controller and deskw buffer design. Reproduced with permission from [56], ©2000 IEEE

deskw eliminated the need for a tightly matched reference clock as in [56]. The deskw hierarchy depth and the phase detector quantization error should be kept low to ensure that there is no excessive accumulation of residual mismatches between the primary reference and the final clock zones. In this implementation, the depth of the deskw hierarchy is 3. Figure 2.43 shows the skew profile before and after

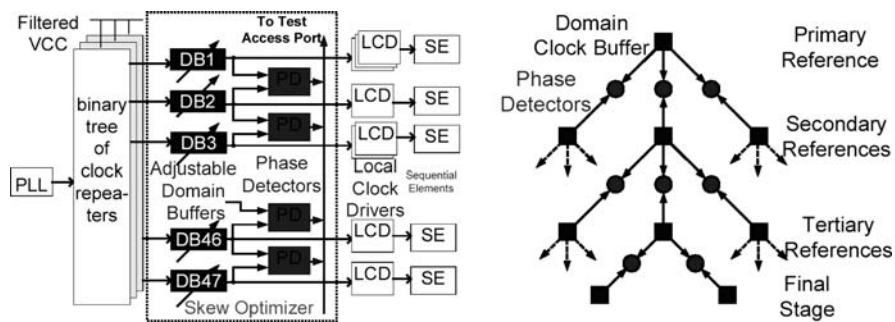


Fig. 2.42. Pentium® 4 processor deskw architecture. Reproduced with permission from [49], ©2001 IEEE

deskew. With deskew enabled, the in-die skew was adjusted to $\pm 8\text{ps}$ and limited by the resolution of the adjustable delay elements. The preadjustment skew was at about 64ps . A hierarchical deskew architecture with deeper hierarchy was used in the 90nm dual-core Itanium® processor [43]¹³ and is shown in Fig. 2.44.

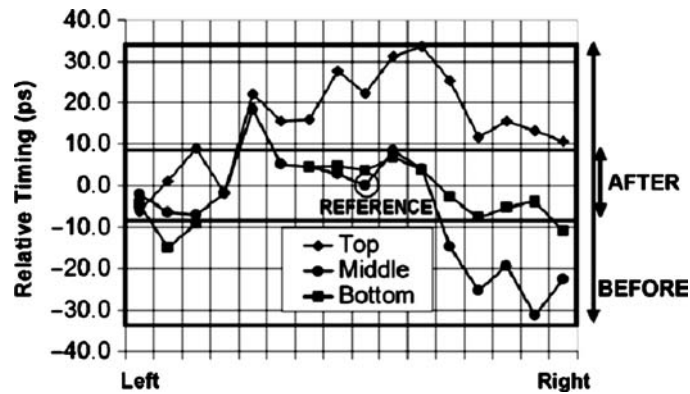


Fig. 2.43. Before and after skew profile of the Pentium® 4 processor. Reproduced with permission from [49], ©2001 IEEE

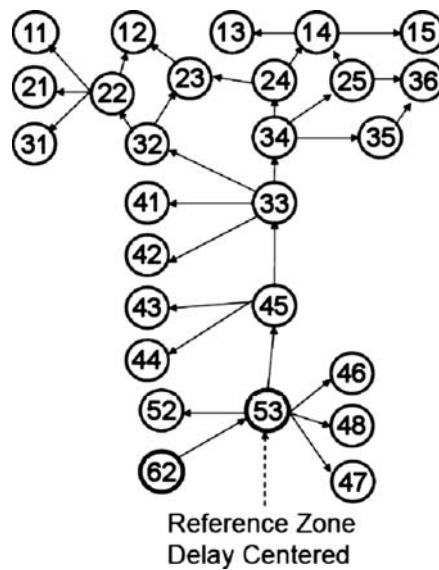


Fig. 2.44. Hierarchical deskew architecture of a dual-core processor. Reproduced with permission from [43], ©2005 IEEE

¹³ Other names and brands may be claimed as property of others.

The hierarchical deskew scheme can be generalized to cover any neighboring clock zones in an H-tree style implementation (Fig. 2.45) or a mesh style implementation (Fig. 2.46) [57]. In the H-tree topology shown in Fig. 2.45, deskew is accomplished hierarchically. For example, the level-1 phase detector placed at the boundary between zone D and zone H will determine which of these two clocks is early and the control logic associated with this phase detector will adjust the clock delays to reduce the skew to within some predetermined guard-band. Once the zones associated with the level-1 phase detectors are brought in phase, deskew will continue in zones associated with the level-2 phase detectors. The procedure will continue until it reaches the level-4 phase detector. There are drawbacks associated with the H-tree deskew topology. First, a deskew buffer must be inserted in all the branches associated with a phase detector leading to longer clock distribution delay and higher jitter. Second, there could be large accumulated guard-bands in zones that are physically adjacent but hierarchically far apart (example: zone B and zone C).

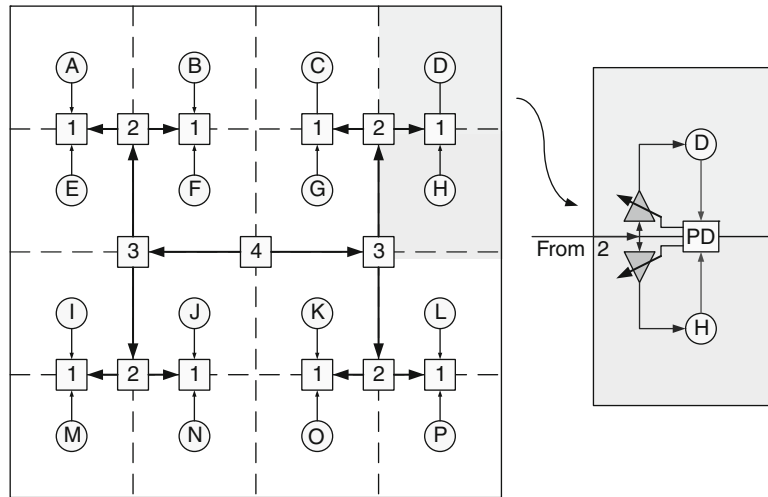


Fig. 2.45. H-tree deskew topology (PD = Phase detector) [57]

The mesh deskew topology addresses the drawbacks of the H-tree deskew topology. In the mesh deskew topology, deskew will be performed on all adjacent zones via averaging. For example, region C in Fig. 2.46 is deskewed with respect to zones B, D, & G simultaneously by averaging their respective phase detector outputs. To ensure stability, a mesh deskew algorithm has been proposed that takes into account potential conflicts and the impact due to guard-band accumulations [57].

Table 2.5.2 summarizes clock distribution characteristics of various commercial processors. The prevalence of adaptive skew compensation techniques is evident.

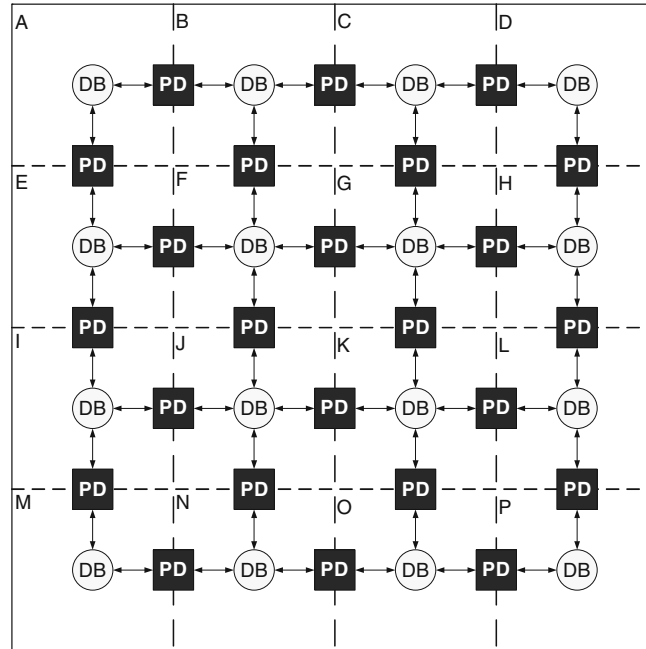


Fig. 2.46. Mesh deskew topology (PD = Phase detector, DB = Deskew buffer) [57]

Table 2.3. Clock distribution characteristics of commercial processors

Name	Ref.	Frequency (MHz)	skew (ps)	Technology (nm)	Clock Dist. style	Deskew
Merom	[30]	3,000	18	65	Tree/Grid	Yes
Power6®	[20]	5,000	8	65	Sym. H-Tree/Grid	Yes
Quad-Core Opteron™	[16]	2,800	12	65	Tree/Grid	
Xeon® processor	[45]	3,400	11	65	Tree/Grid	Yes
Itanium® 2 processor	[43]	>2,000	10	90	Asymmetric tree	Yes
Power5®	[19]	>1,500	27	130	Sym. H-Tree/Grid	No
Pentium® 4 processor	[29]	3,600	7	90	Recombinant tile	Yes
Itanium® 2 processor	[42]	1,500	24	130	Asymmetric tree	Yes
Power4®	[17]	>1,000	25	180	Tree/Grid	No
Itanium® 2 processor	[35]	1,000	52	180	Asymmetric tree	No
Pentium® 4 processor	[8]	>2,000	16	180	Spine/Grid	Yes
Itanium® processor	[36]	800	28	180	H-Tree/Grid	Yes

Note: Other names and brands may be claimed as property of others

2.5.3 Local Clock Compensation Architecture

The concept of global clock compensation outlined in the last section and aimed at optimizing clock skews (intentional or unintentional) among global clock zones can

be easily extended to the local level to address path specific clock timings. Specifically, locating critical paths (LCP) buffers with adjustable delays can replace regular local buffers with fixed delays [7, 30, 45]. Figure 2.47 shows an implementation example [30]. The local clock buffers are replaced by LCP buffers to create the LCP domains. They are controlled by the LCP control chain and the chain setting is usually determined post-silicon. Since each LCP buffer is targeted at a limited fan out, the number of LCP domains could be too many to manage. To overcome this difficulty, a processor implementation will cluster the LCP domains resulting in hundreds of LCP zones. The LCP technique is highly effective in resolving clock timing marginalities post-silicon. For example, a MAX timing path between “A” and “B” in Fig. 2.47 can be compensated by intentionally delaying the LCP buffer at the receiver (B). On the contrary, a MIN path marginality between “A” and “B” can be resolved by delaying the LCP buffer at domain A. The highly distributed nature of the LCP methodology permits fine-grain post-silicon clock timing adjustments in contemporary processor implementations.

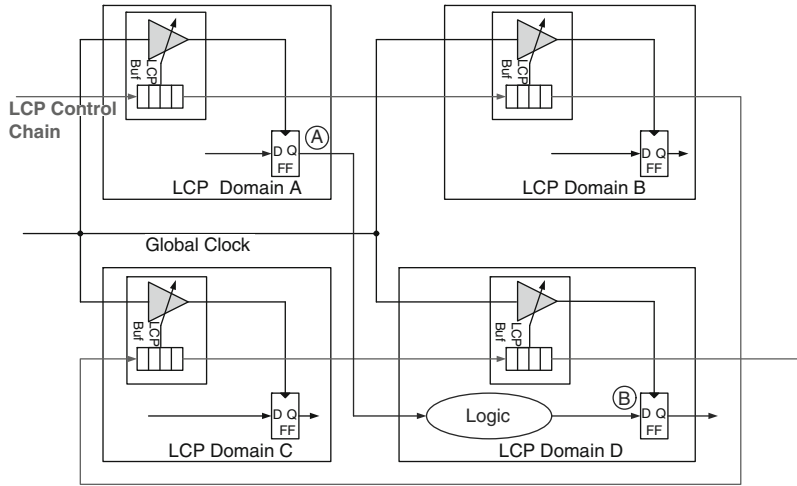


Fig. 2.47. Local Clock Compensation [30]

2.6 Elements of Clock Distribution Circuits

2.6.1 Clock Duty Cycle

Maintaining the clock duty cycle as close to 50% as possible is important and particularly critical for high performance processors. Specifically, high performance processors will have many phase paths in which any duty cycle distortion will unnecessarily penalize the design. Let us assume a clock period of T_{per} with a duty cycle error of $Q\%$. The corresponding high and low phase times become:

$$T_{\text{phH}} = T_{\text{per}} \times \frac{(50 + Q)}{100}, \quad (2.29)$$

$$T_{\text{phL}} = T_{\text{per}} \times \frac{(50 - Q)}{100}. \quad (2.30)$$

Let $F_{50\%}$ be the maximum operating frequency achievable with a 50% duty cycle. Then, the corresponding high and low phase paths will correspond to effective frequencies F_{maxH} and F_{maxL} :

$$F_{\text{maxH}} = F_{50\%} \times \frac{100}{(100 + 2Q)}, \quad (2.31)$$

$$F_{\text{maxL}} = F_{50\%} \times \frac{100}{(100 - 2Q)}. \quad (2.32)$$

Figure 2.48 shows the effective F_{max} increase necessary for a phase path to meet timing due to duty cycle distortion. The effective maximum frequency increase is approximately twice that of the percentage duty cycle offset from 50%.

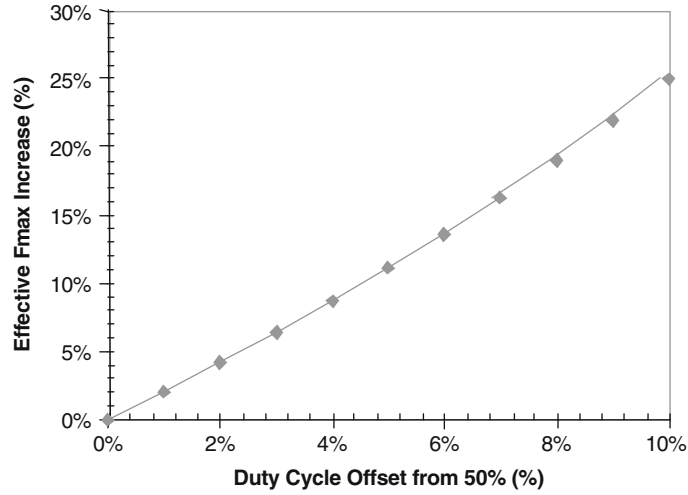


Fig. 2.48. F_{max} shift caused by duty cycle distortion in a phase-path dominated design

Clock distribution induced duty cycle error is mainly attributed to asymmetry in the clock distribution repeaters. Figure 2.49 compares a buffer-based (two inverters) clock distribution design vs. an inverter-based clock distribution. In a buffer-based design, an incoming clock edge undergoes asymmetric rise and fall edges: Two fall edges experience gate loading only whereas two rise edges experience interconnect loading. In contrast, in the inverter-based implementation, both positive and negative edges experience similar loads. By having loading symmetry between rise and fall edges, an inverter-based clock distribution network is more robust in maintaining duty cycle fidelity.

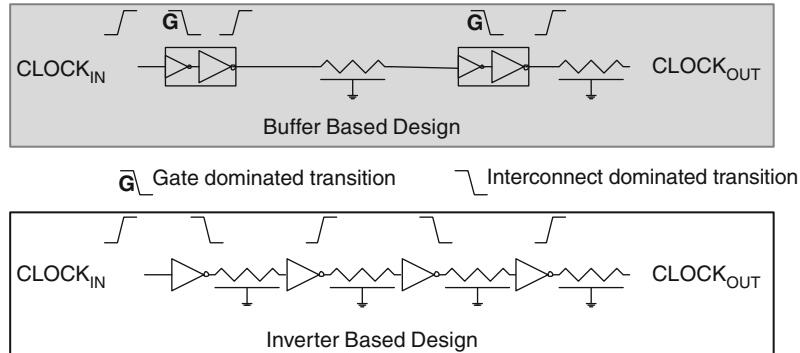


Fig. 2.49. Duty cycle distortion due to asymmetric edge propagation between a buffer-based clock distribution and an inverter-based clock distribution

In applications such as double data rate circuits where attaining a near perfect 50% duty cycle is critical, a differential clock distribution is the preferred solution. Sending the clock differentially improves noise immunity and duty cycle fidelity. Moreover, cross coupled stages can be added to further reduce duty cycle error by introducing constraints between true and complement clock signals. Finally, active duty cycle correction can be added to dynamically adjust the duty cycle across process, voltage, and temperature variations. Figure 2.50 shows an active duty cycle correction system with the corrector and detector circuits shown in Fig. 2.51. In a differential clocking environment, the duty cycle can be easily determined by sensing the clock common mode and making the appropriate adjustments.

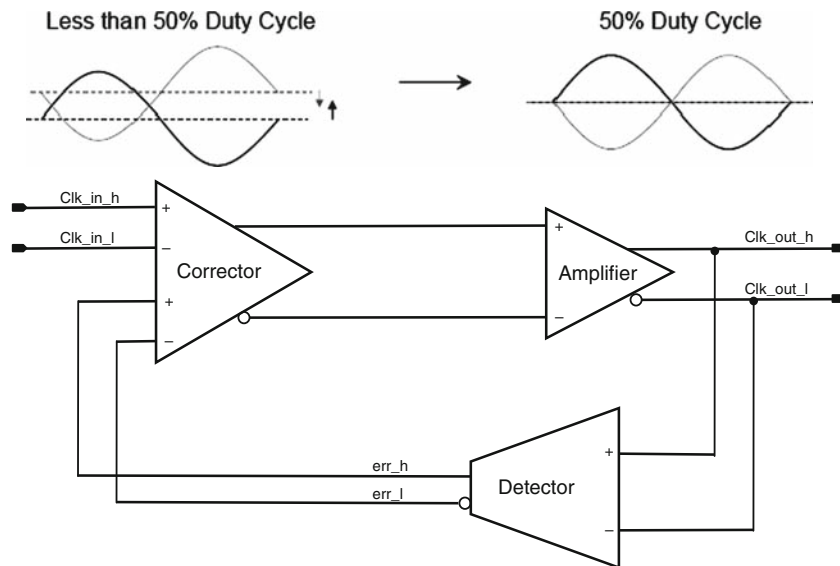


Fig. 2.50. Duty cycle corrector. Reproduced with permission from [13], ©2008 IEEE

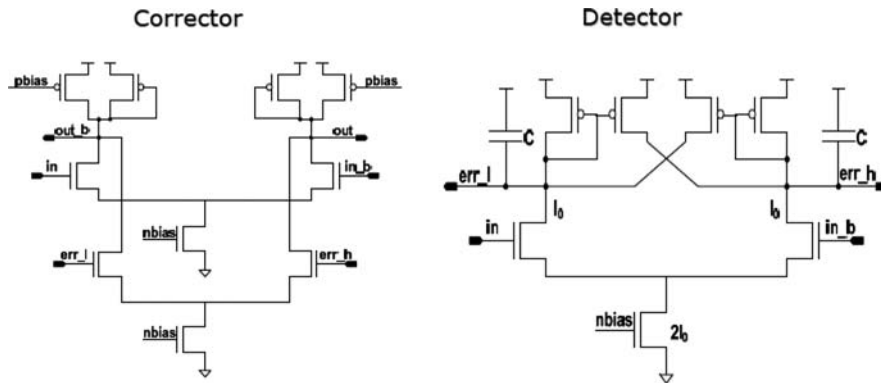


Fig. 2.51. Duty cycle corrector circuits. Reproduced with permission from [13], ©2008 IEEE

2.6.2 Power Supply

In (2.9), it was noted that the clock distribution delay variation is proportional to the distribution latency of which the delay sensitivity to power supply variation is a key factor. Power supply droop due to di/dt and power supply noise due to switching are two significant sources.

Power supply decoupling capacitors are used extensively to reduce the power supply droop and noise effects. Wong et al. [58] studied improving the processor timing margin by enhancing its immunity to power supply noise by compensating the clock delay against the data delay. In addition to on-die decoupling capacitors, on-die power supply filters can reduce the impact of power supply noise. Figure 2.52 shows two possible implementations [16, 49]. The implementation in [49] demonstrated a $5\times$ reduction in power supply noise with filtering (Fig. 2.53). It should be noted that the circuits shown in Fig. 2.52, if not designed with care, can experience excessive DC power supply drop leading to delay degradations.

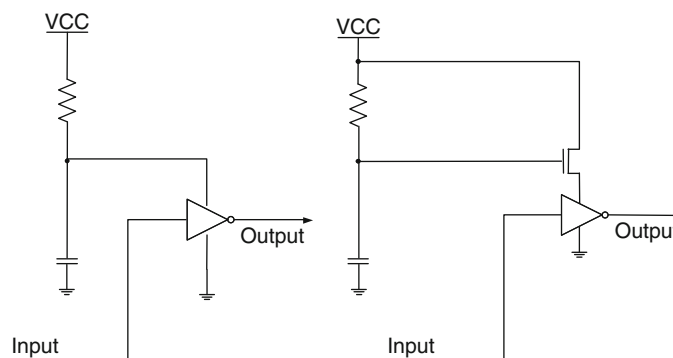


Fig. 2.52. Clock buffer design with power-supply filters. Reproduced with permission in a form similar to that in [49] and [16], ©2001, 2007 IEEE

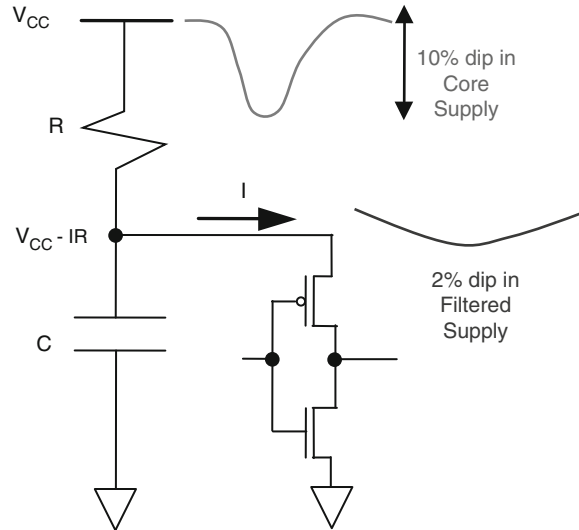


Fig. 2.53. On-die clock tree filter circuit. Reproduced with permission from [49], ©2001 IEEE

The reader is encouraged to review Sect. 6.8 for an analytical model of supply noise induced jitter as a function of noise frequency and amplitude and clock network latency.

2.7 Clock DFX Techniques

Clock DFX refers to the design-for-test or design-for-manufacturing features. DFX capabilities are critical for the design of the clock distribution network. For example, methods to measure on-die clock skew and jitter are needed for post-silicon validation of the design. Additionally, the ability to manipulate the clock edges at some specific clock cycle is a critical timing debug feature.

2.7.1 Optical Probing

Due to the increased number of metal layers and flip-chip packaging, optical probing from the back-side of the die is a standard technique to probe die internal nodes [59]. This technique is quite suitable for on-die clock measurement due to the periodic nature of the clock. The back-side optical probing technique monitors the infrared photons that are emitted by switching transistors. Figures 2.54 and 2.55 illustrate the emission mechanism and measurement methodology [60]. The photon emission intensity is proportional to the transistor switching current and is at a maximum during the switching transient. Therefore, the clock transition edge can be determined by correlating the photon emission peak with a time base. By monitoring the clock

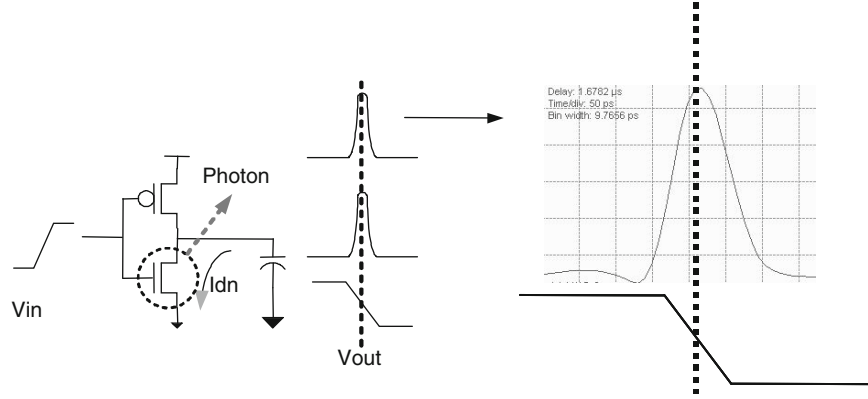


Fig. 2.54. Back-side optical probing technique (a). Reproduced with permission from [60], ©2004 IEEE

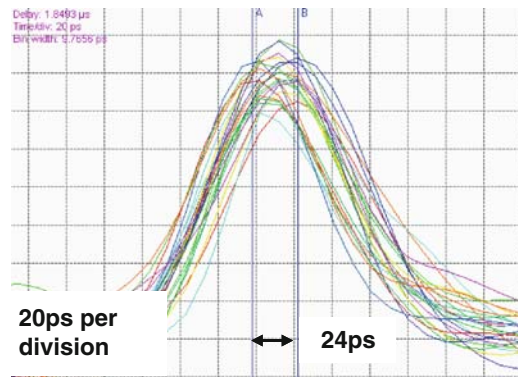


Fig. 2.55. Back-side optical probing technique (b). Reproduced with permission from [60], ©2004 IEEE

drivers in various die locations, this technique provides static clock skew measurement capability. Limitations of this technique are instrumentation complexity, timing resolution and its inability to measure dynamic clock uncertainties (jitter) due to the intrinsic averaging effect of the measurement.

2.7.2 On-Die Measurement

On-die skew and jitter measurement schemes are more suitable for high volume data collection when compared to the optical back-side probing technique outlined in the last section. Figure 2.56 shows a circuit capable of measuring both skew and jitter [61]. The circuit contains an inverter based delay line (inverters A_1 to A_{K+1}) and a

chain of flip-flops configured to sample the taps along the delay line. The inverters are designed to be nominally identical and with low fan out. The length of the delay line and the sampling chain K is designed to be sufficiently long so that the total delay will fully contain at least one clock period at the lowest frequency of interest. If τ is the nominal unit delay for the inverter and $T_{\text{per,max}}$ is the maximum clock period, then $K\tau > T_{\text{per,max}}$. In [61], K is set to 128 and the inverter has a nominal delay of 8ps. When Ck_A is selected as the input to the delay line (via the MUX in “Ck_{Sample1}” in Fig.2.56), the resulting pattern from the sampling flip-flops is shown in Fig.2.57 with the clock transition edges denoted by two consecutive 0s or two consecutive 1s. If the number of inverter stages between the clock transition edges is N_x , the phase time T_{ph} is bounded by:

$$N_x \tau \leq T_{\text{ph}} \leq (N_x + 1) \tau. \quad (2.33)$$

If there is no jitter on the incoming clock, the stage count N_x and the pattern will be the same in every cycle. With nonzero clock jitter, the sample pattern will not be identical in every cycle and the sampled locations of the clock edges will vary.

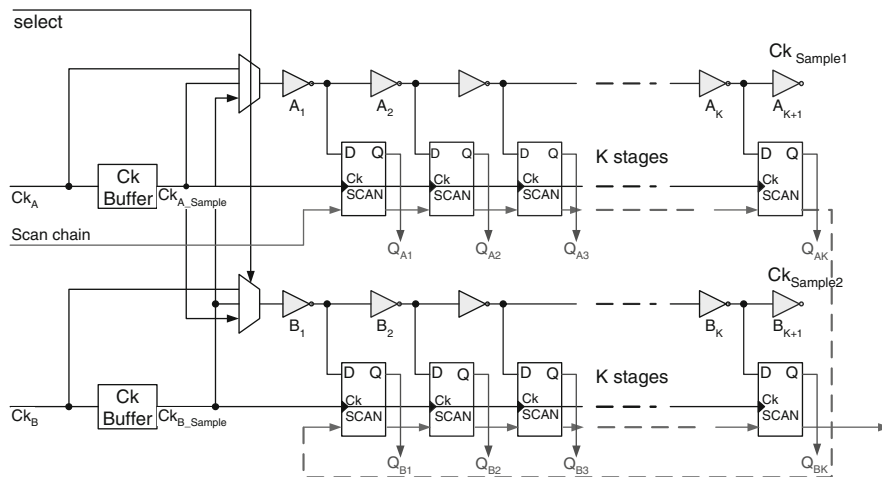


Fig. 2.56. Skew and jitter measurement circuit. Reproduced with permission from [61], ©2004 IEEE

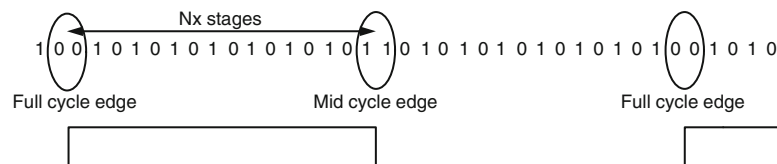


Fig. 2.57. Sampled delay pattern of the skew and jitter measurement circuit

To measure the clock skew between clock Ck_A and Ck_B , the coupled configuration shown in Fig. 2.56 is used. Clock Ck_{B_Sample} is multiplexed to the delay chain in circuit “ $Ck_{Sample1}$ ” and Ck_{A_Sample} is multiplexed to the delay chain in circuit “ $Ck_{Sample2}$.” If there is no skew between Ck_A and Ck_B , the captured patterns from the two circuits will be identical. With finite skew, the difference in the position of the sampled clock edges is the measured skew.

Shortcomings of the aforementioned technique are the limited measurement resolution due to the discrete unit delay of the inverter and the delay variation among the inverters in the delay line [62, 63]. A vernier delay line (VDL) as shown in Fig. 2.58 can be used to improve the measurement resolution. With $\tau_A \neq \tau_B$ (Fig. 2.58), the timing resolution becomes $\delta = |\tau_A - \tau_B|$. Although this scheme can in principle achieve higher resolution, delay variation among the stages will impose a limitation on the results.

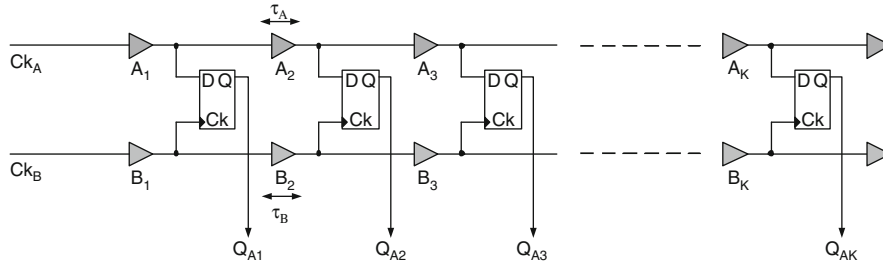


Fig. 2.58. Vernier delay line

Figure 2.59 shows a timing diagram of the Vernier delay line with Ck_A and Ck_B skewed by 3δ where δ is the delay difference between τ_A and τ_B . The sampled

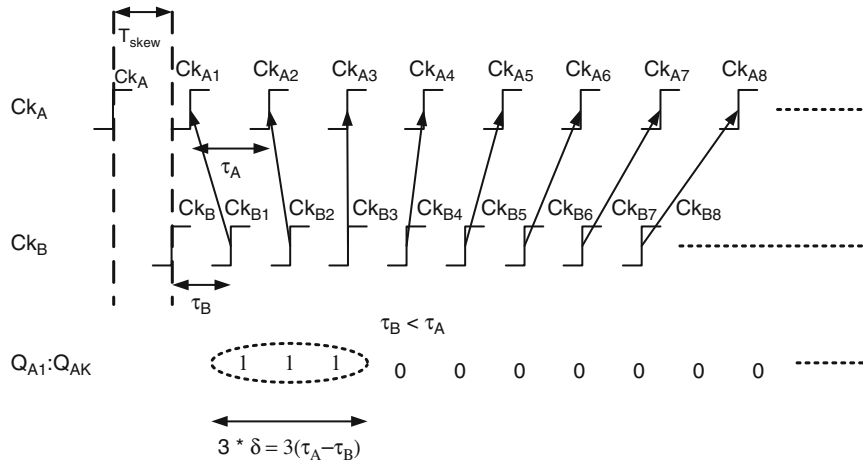


Fig. 2.59. Vernier delay line timing example

pattern Q_{A1} to Q_{AK} will show three consecutive ones representing the skew between the two clocks. In general, if the skew between the clock inputs to the VDL is T_{skew} and the stage delays are τ_A and τ_B respectively, the number of ones (N_y) in the observed pattern will be:

$$N_y = \left\lfloor \frac{T_{\text{skew}}}{\delta} \right\rfloor, \quad (2.34)$$

$$\delta = |\tau_A - \tau_B| \quad (2.35)$$

The limitation of the VDL technique is attributed to the stage-to-stage delay variations making measurement resolutions in the sub-picosecond range difficult to achieve without incorporating additional mismatch compensation schemes.

2.7.3 Locating Critical Path

In addition to being an important post-silicon (local) clock compensation technique, the locating critical path (LCP) scheme is also a valuable post-silicon timing and clock debug tool [7, 30]. By intentionally skewing the clock arrival times between local clock zones, physical locations of marginal timing paths can be identified for further analysis. When coupled with the on-die clock shrink method (Sect. 2.7.4), the LCP technique has been shown to be a critical post-silicon clock and timing debug tool.

2.7.4 On-Die-Clock Shrink

In conjunction with the LCP technique, the on-die clock shrink (ODCS) method [56] is an effective clock edge manipulation technique that has been used extensively in post-silicon timing debug. The main concept behind ODCS is the capability to manipulate specific clock edges for phase or cycle expansion or contraction. Figure 2.60

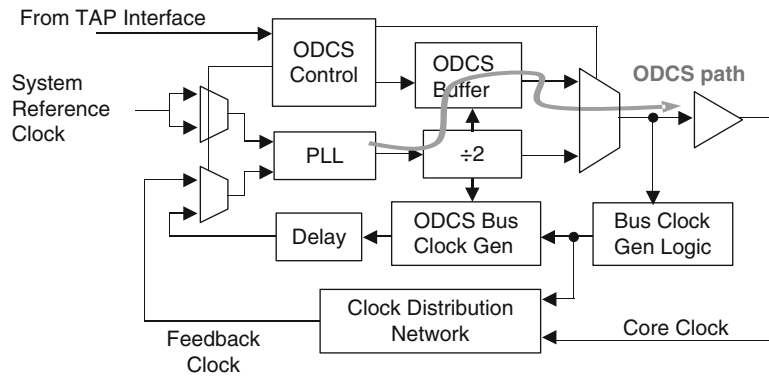


Fig. 2.60. On-die-clock shrink architecture. Reproduced with permission from [56] ©2000 IEEE

shows the clock generation architecture capable of supporting this feature. An ODCS buffer is inserted in series with the core clock generation path. The ODCS buffer is designed to be able to stretch or shrink the phase or cycle time.

Figure 2.61 shows an example of the ODCS operating principle. The input to the ODCS buffer is assumed to have equal high and low phase times. The initial rise and fall default setting are equal (10 arbitrary units in Fig. 2.61). With this setting, the default ODCS buffer output will also have equal high and low phase times. The settings in cycles $N+1$, $N+2$, and $N+3$ shown in Fig. 2.61 will shrink the low phase at cycle $N+1$ and then restore the clock waveform to the default.¹⁴ Figure 2.62 summarizes the overall capabilities of the ODCS technique. Implementation of ODCS requires the incorporation of ODCS rise and fall registers and a state machine to cycle through these registers in a deterministic manner.

In an actual experiment to locate the source of the timing failure during timing debug, the default operating frequency of the device under test is first relaxed and the ODCS phase or period shrinkage is applied systematically across a specific range of tester cycles to locate the actual cycle of failure and the failure timing margin. In this manner, the failure pattern can be correlated to a specific test cycle to identify root cause.

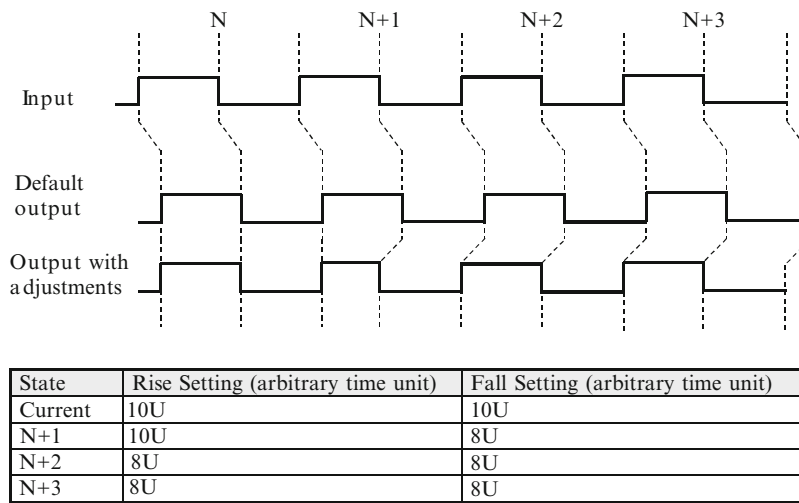


Fig. 2.61. ODCS clock waveform. Reproduced with permission from [56] ©2000 IEEE

¹⁴ In this implementation, the $N+1$ “Rise” setting actually affects the rising edge at cycle $N+2$ since this setting is applied after the fall edge of cycle $N+1$. The $N+1$ “Fall” setting affects the falling edge at cycle $N+1$ because it is also applied after the rising edge of cycle $N+1$.

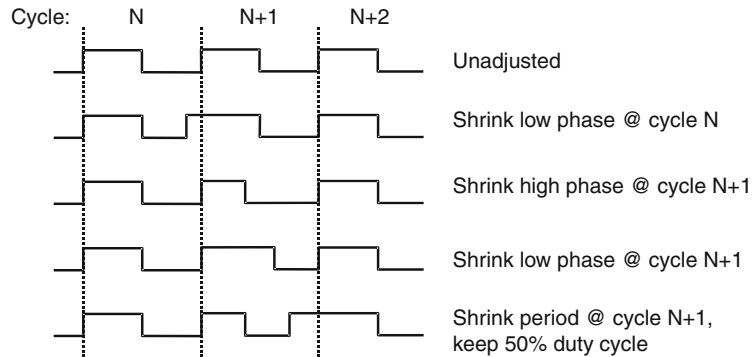


Fig. 2.62. ODCS capabilities. Reproduced with permission from [56] ©2000 IEEE

2.8 Multiclock Domain Distributions

As digital designs move towards multicore and systems-on-chip (SOC) architectures, multiple clock domains will become a prevalent design style and the clock distribution schemes will need to be enhanced to fulfill this need. A multiclock domain design typically embodies multiple islands operating synchronously and served by independent clocks within the domain and with dedicated interfaces to manage the inter-domain communications. In a multicore processor, each of the cores can be a separate clock domain. Similarly, different functional units in an SoC can be on separate clock domains.

In multicore processors and SoC designs, it is advantageous to implement multiple clock domains because this scheme provides functional flexibility for each of the domains to operate at the optimal frequency and to minimize the complexity and power associated with distributing a low skew clock to the entire die. The multidomain clock distribution architectures for multicore processors and SoCs belong to a class of designs called globally asynchronous and locally synchronous (GALS) [64]. Table 2.4 summarizes synchronization categories within the GALS class.

Table 2.4. Clock synchronization categories

Type	Characteristics of distribution
Synchronous	Single distribution point-of-divergence (POD) with known static delay offsets among all the branches and single operating frequency. Encompasses all of the clock distribution styles described in Sects. 2.3 and 2.4
Mesochronous	Single distribution POD but with nonconstant delay offset among the branches
Plesiochronous	Multiple distribution PODs but with nominally identical frequency among all the domains
Heterochronous	Multiple distribution PODs with nominally different operating frequencies among the domains

Figure 2.63 is a generic illustration of the GALS design style. Multiple clock domains are embedded in a single silicon die. The chip may receive multiple copies of the system clock and multiple PLLs are used to generate the clocks for each of the synchronous units. Clocks Ck1, Ck2, ..., Ck4 may have different phases or frequencies. The clock distribution within Each synchronous unit will rely on any of the topologies described in Sect. 2.3 to achieve low skew and fully synchronous operation.

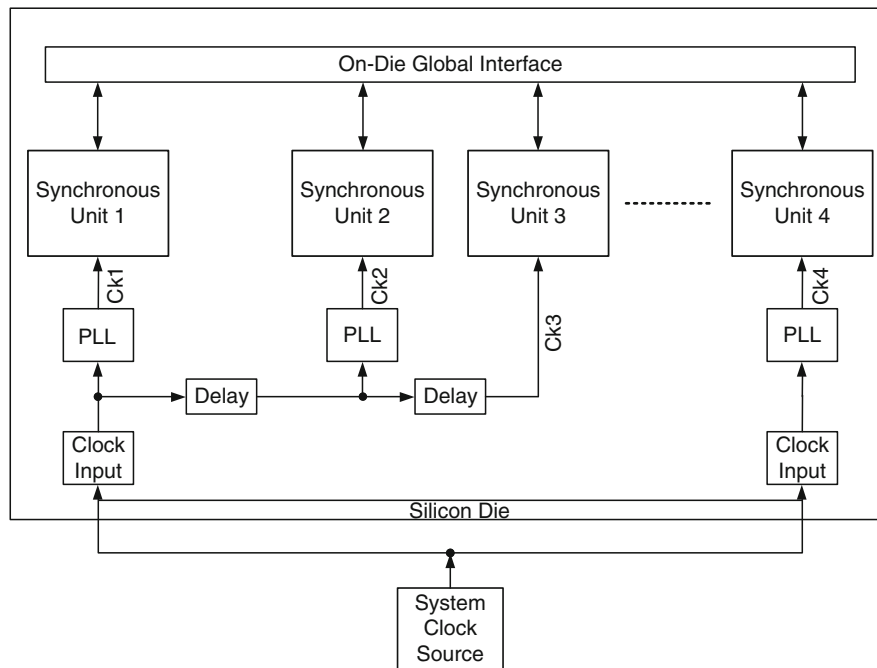


Fig. 2.63. Globally synchronous and locally synchronous architecture. Reproduced with permission from [64], ©2007 IEEE

Since each of the synchronous units can operate at different frequency and phase from the rest, the on-die global interfaces will be responsible for the data transfer among the domains. Many modern multicore processors and SoCs only adopt the loosely synchronous styles of Table 2.4 to avoid the significant complexity associated with truly asynchronous design.

2.8.1 Multicore Processor Clock Distribution

A plesiochronous clock distribution involves multiple distribution PODs but with a nominally identical operating frequency. The 65nm dual core processor described

in [45] is an example. Figure 2.64 illustrates this scheme which consists of two domains for the two cores and the uncore and *I/O* domain with the interface operating at the same frequency. The system clock input to the three clock generators (PLLs) is routed in the package. There are three independent distribution PODs for the cores and the uncore. A recombinant style distribution for low skew is used in the core whereas the uncore employs a hybrid spine-grid structure for lower power and simpler implementation. Data communication between the cores and the uncore is accomplished via the pipelined deskew logic (PDSL) interfaces operating at the same frequency. Introducing independent clock domains for the cores permitted a modular design.

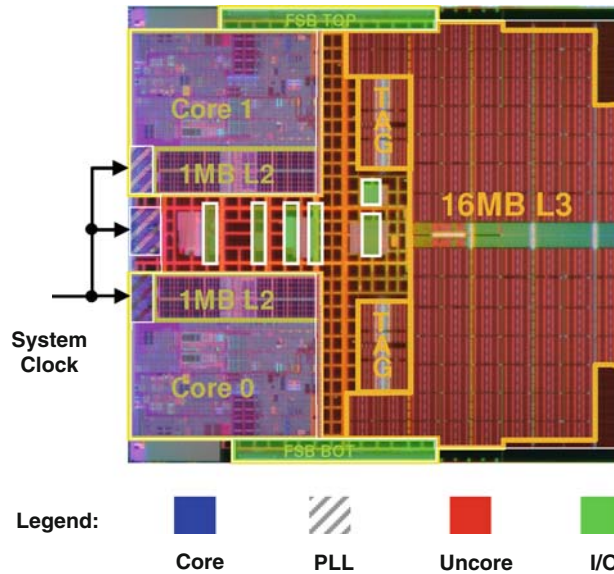


Fig. 2.64. Multidomain clocking in a dual-core processor. Reproduced with permission from [45], ©2006 IEEE

A mesochronous clock distribution has a single distribution POD while exhibiting nonconstant delays to the branches. The 80-tile processor design in [65, 66] is one example of a mesochronous clock distribution. This processor has a single PLL and therefore a single distribution POD (Fig. 2.65). Distributing the global clock to the respective tiles is achieved using chains of clock buffers embedded within each tile. The daisy-chaining of the global clock buffers suggests nonequal clock latencies to each processor tile. The systematic clock skews across the processor tiles help spread the clock switching power over an entire cycle. Interprocessor-tile communication is achieved with a skew insensitive FIFO interface [66]. Within each processor tile, a balanced H-tree is responsible for the distribution and all circuits

within each processor tile operate synchronously. Figure 2.65 (left) shows details of the distribution scheme. Figure 2.65 (right) shows the relative arrival times of the clocks at the processor tiles. In a processor design having a large number of processor cores operating at the same frequency, this clock distribution style has the potential of reducing the impact of simultaneous switching on the on-chip power delivery network.

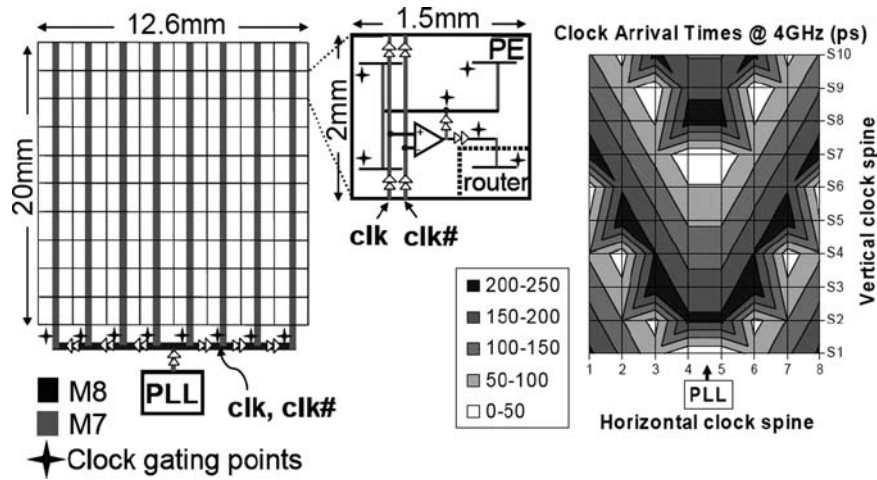


Fig. 2.65. Clock distribution of an 80-tile processor design. Reproduced with permission from [66], ©2008 IEEE

A heterochronous clock distribution has multiple distribution PODs and can support nominally different operating frequencies across the domains [12, 13, 16, 67]. The 90nm 2-core Itanium® processor [12] and the 65nm quad-core Itanium® processor [13] have adaptive core frequency switching implemented with multiple PLLs. Low failure probability FIFOs are placed at the interface of the corresponding clock domains. Similarly, [16] and [67] have independent PLLs for the cores, uncore, and the *I/O* that are capable of operating at different frequencies. Clock domain crossing is accomplished with low-latency FIFO schemes.

Despite the potential design benefits of the loosely synchronous methods described above, the requirement of a deterministic domain crossing will impose higher design complexity at the interfaces. FIFO-like structures are used at the cost of additional transfer latency in order to compensate for the higher crossdomain clock uncertainties. The PDSL interface in [45] incurs two additional cycles of latency when compared to a single-cycle transfer in a fully synchronous design. Similarly, [66] incorporates a 4-deep FIFO to ensure robust crossdomain transfer. Better partitioning that reduces the crossdomain traffic will reduce the performance penalty due to the increased synchronization latency.

2.9 Future Directions

Alternative clock distribution methods have been proposed in the literature. Among them are optical clock distribution [68, 69], package level clock distribution [70], clock network with distributed PLLs [71] and resonant clocking [72, 73].

The main advantage of an optical clock distribution scheme vs. a conventional one is the potential for higher performance and short-term jitter reduction. This is due to the ability of an optical distribution network to have fewer repeaters and be first-order immune to on-chip power supply noise. A significant disadvantage is the design complexity and performance limitations (e.g. power & electrical robustness) of the signal conversion from optical to electrical and vice versa. Another limitation is due to the process complexity involved in the formation of the on-die waveguides.

In a package level clock distribution, high speed package routes replace the on-die clock network. The package interconnect will have comparatively much lower *RC* delay. The main limitation is due to the interface between the package network and the on-die receivers. Additional skew and jitter at the on-die receivers (due to buffer design and ESD requirements) and testing complexities in prepackaged parts are the most important implementation impediments.

Application of on-die resonant clocking is a very promising clock distribution alternative due to lower power dissipation and jitter reduction. A significant hurdle of on-die resonant clocking is in the physical limitation of the network and the naturally narrow frequency operating window. Numerous enhancements to resonant clocking have been proposed and some of these techniques can become practical. A resonant clock distribution network was applied to a 90nm processor by adding a new top layer metal for the *LC* tank inductors and capacitors [74]. This design demonstrated basic functionality and power savings at frequencies above 3GHz. A detailed discussion of resonant clocking techniques is presented in Chap.4.

All of the alternatives above try to address limitations of electrical clock interconnect such as power and noise. With the strong emergence of multidomain clock distribution in multicore processors and SOCs, constraints due to the very large on-die electrical interconnect are significantly alleviated. Since the industry is moving away from a single large distribution network with long latency and large number of clock buffers, on-die electrical clock distribution will continue to be the predominant clocking technology in the near future.

2.10 Conclusion

In this chapter, the requirements for synchronous clock distributions have been described. Numerous clock distribution topologies have been described with spine-grid and tree-grid topologies emerging as the dominant ones. On-die deskew has become common practice in order to contain skew without excessive design complexity and use of interconnect resources. Various clock debug features have been summarized that encompass circuits for skew and jitter measurement, design for test and critical path location. Additionally, the chapter has also outlined the basics of multidomain

clock distribution topologies that are becoming the industry norm. Finally, the chapter concludes by mentioning three alternative clock distribution techniques that may demonstrate practical design benefits in the future.

References

- [1] J. Schutz, "A CMOS 100MHz microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1991)*, 1991, pp. 90–91, 294.
- [2] J. Schutz, "A 3.3V 0.6 μ m BiCMOS superscalar microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1994)*, 1994, pp. 202–203.
- [3] R. Colwell and R. Steck, "A 0.6 μ m BiCMOS processor with dynamic execution," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1995)*, 1995, pp. 176–177, 361.
- [4] M. Choudhury and J. Miller, "A 300 MHz CMOS microprocessor with multimedia technology," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1997)*, 1997, pp. 170–171, 450.
- [5] J. Schutz and R. Wallace, "A 450MHz IA32 P6 family microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1998)*, 1998, pp. 236–237.
- [6] P. Green, "A 1GHz IA-32 architecture microprocessor implemented on 0.18 μ technology with aluminum interconnect," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2000)*, 2000, pp. 98–99, 449.
- [7] J. Schutz and C. Webb, "A scalable x86 CPU design for 90nm process," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2004)*, 2004, pp. 62–63, 513.
- [8] N. Kurd, J. Barkatullah, R. Dizon, T. Fletcher, and P. Madland, "Multi-GHz clocking scheme for Intel® Pentium® 4 microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2001)*, 2001, pp. 404–405.
- [9] G. Singer and S. Rusu, "The first IA-64 microprocessor: a design for highly-parallel execution," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2000)*, 2000, pp. 422–423.
- [10] S. Naffziger and G. Hammond, "The implementation of the next-generation 64 b Itanium™ microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2002)*, vol. 1, 2002, pp. 344–345, 472.
- [11] J. Stinson and S. Rusu, "A 1.5GHz third generation Itanium® processor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2003)*, 2003, pp. 252–253, 492.
- [12] S. Naffziger, B. Stackhouse, and T. Grutkowski, "The implementation of a 2-core multi-threaded Itanium family processor," in *Digest of Technical*

- Papers IEEE International Solid-State Circuits Conference (ISSCC 2005)*, 2005, pp. 182–183, 592.
- [13] B. Stackhouse, B. Cherkauer, M. Gowan, P. Gronowski, and C. Lyles, “A 65nm 2-billion-transistor quad-core Itanium® processor,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2008)*, 2008, pp. 92–93, 598.
 - [14] S. Rusu, S. Tam, H. Muljono, D. Ayers, and J. Chang, “A dual-core multi-threaded Xeon® processor with 16MB L3 cache,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2006)*, 2006, pp. 315–324.
 - [15] M. Golden, S. Arekapudi, G. Dabney, M. Haertel, S. Hale, L. Herlinger, Y. Kim, K. McGrath, V. Palisetti, and M. Singh, “A 2.6GHz dual-core 64bx86 microprocessor with DDR2 memory support,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2006)*, 2006, pp. 325–332.
 - [16] J. Dorsey, S. Searles, M. Ciraula, S. Johnson, N. Bujanos, D. Wu, M. Braganza, S. Meyers, E. Fang, and R. Kumar, “An integrated quad-core Opteron processor,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2007)*, 2007, pp. 102–103.
 - [17] P. Restle, C. Carter, J. Eckhardt, B. Krauter, B. McCredie, K. Jenkins, A. Weger, and A. Mule, “The clock distribution of the Power4 microprocessor,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2002)*, vol. 1, 3–7 Feb. 2002, pp. 144–145.
 - [18] P. Hofstee, N. Aoki, D. Boerstler, P. Coulman, S. Dhong, B. Flachs, N. Kojima, O. Kwon, K. Lee, D. Meltzer, K. Nowka, J. Park, J. Peter, S. Posluszny, M. Shapiro, J. Silberman, O. Takahashi, and B. Weinberger, “A 1GHz single-issue 64b PowerPC processor,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2000)*, 2000, pp. 92–93.
 - [19] J. Clabes, J. Friedrich, M. Sweet, J. Dilullo, S. Chu, D. Plass, J. Dawson, P. Muench, L. Powell, M. Floyd, B. Sinharoy, M. Lee, M. Goulet, J. Wagoner, N. Schwartz, S. Runyon, G. Gorman, P. Restle, R. Kalla, J. McGill, and S. Dodson, “Design and implementation of the POWER5™ microprocessor,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2004)*, 2004, pp. 56–57.
 - [20] J. Friedrich, B. McCredie, N. James, B. Huott, B. Curran, E. Fluhr, G. Mittal, E. Chan, Y. Chan, D. Plass, S. Chu, H. Le, L. Clark, J. Ripley, S. Taylor, J. Dilullo, and M. Lanzerotti, “Design of the POWER6 microprocessor,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2007)*, 2007, pp. 96–97.
 - [21] C. Webb, C. Anderson, L. Sigal, K. Shepard, J. Liptay, J. Warnock, B. Curran, B. Krumm, M. Mayo, P. Camporese, E. Schwarz, M. Farrell, P. Restle, I. Averill, R.M., T. Slegel, W. Houtt, Y. Chan, B. Wile, T. Nguyen, P. Emma, D. Beece, C.-T. Chuang, and C. Price, “A 400-MHz S/390 microprocessor,” *IEEE Journal of Solid-State Circuits*, vol. 32, no. 11, pp. 1665–1675, Nov. 1997.

- [22] G. Northrop, R. Averill, K. Barkley, S. Carey, Y. Chan, Y. Chan, M. Check, D. Hoffman, W. Huott, B. Krumm, C. Krygowski, J. Liptay, M. Mayo, T. McNamara, T. McPherson, E. Schwarz, L. Siegel, C. Webb, D. Webber, and P. Williams, "600MHz G5 S/390 microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1999)*, 1999, pp. 88–89.
- [23] T. McPherson, R. Averill, D. Balazich, K. Barkley, S. Carey, Y. Chan, Y. Chan, R. Crea, A. Dansky, R. Dwyer, A. Haen, D. Hoffman, A. Jatkowski, M. Mayo, D. Merrill, T. McNamara, G. Northrop, J. Rawlins, L. Sigal, T. Slegel, D. Webber, P. Williams, and F. Yee, "760 MHz G6 S/390 microprocessor exploiting multiple V_t and copper interconnects," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2000)*, 2000, pp. 96–97.
- [24] D. Dobberpuhl, R. Witek, R. Allmon, R. Anglin, D. Bertucci, S. Britton, L. Chao, R. Conrad, D. Dever, B. Gieseke, S. Hassoun, G. Hoeppepner, K. Kuchler, M. Ladd, B. Leary, L. Madden, E. McLellan, D. Meyer, J. Montanaro, D. Priore, V. Rajagopalan, S. Samudrala, and S. Santhanam, "A 200-MHz 64-b dual-issue CMOS microprocessor," *IEEE J. Solid-State Circuits*, vol. 27, no. 11, pp. 1555–1567, Nov. 1992.
- [25] B. Benschneider, A. Black, W. Bowhill, S. Britton, D. Dever, D. Donchin, R. Dupcak, R. Fromm, M. Gowan, P. Gronowski, M. Kantrowitz, M. Lamere, S. Mehta, J. Meyer, R. Mueller, A. Olesin, R. Preston, D. Priore, S. Santhanam, M. Smith, and G. Wolrich, "A 300-mhz 64-b quad-issue CMOS RISC microprocessor," *IEEE J. Solid-State Circuits*, vol. 30, no. 11, pp. 1203–1214, Nov. 1995.
- [26] P. Gronowski, W. Bowhill, D. Donchin, R. Blake-Campos, D. Carlson, E. Equi, B. Loughlin, S. Mehta, R. Mueller, A. Olesin, D. Noorlag, and R. Preston, "A 433-MHz 64-b quad-issue RISC microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 11, pp. 1687–1696, Nov. 1996.
- [27] D. Bailey and B. Benschneider, "Clocking design and analysis for a 600-mhz Alpha microprocessor," *IEEE J. Solid-State Circuits*, vol. 33, no. 11, pp. 1627–1633, Nov. 1998.
- [28] T. Xanthopoulos, D. Bailey, A. Gangwar, M. Gowan, A. Jain, and B. Prewitt, "The design and analysis of the clock distribution network for a 1.2 GHz Alpha microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2001)*, 2001, pp. 402–403.
- [29] N. Bindal, T. Kelly, N. Velastegui, and K. Wong, "Scalable sub-10ps skew global clock distribution for a 90nm multi-GHz IA microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2003)*, 2003, pp. 346–347, 498.
- [30] N. Sakran, M. Yuffe, M. Mehalel, J. Doweck, E. Knoll, and A. Kovacs, "The implementation of the 65nm dual-core 64b Merom processor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2007)*, 2007, pp. 106–107, 590.

- [31] G. Geannopoulos and X. Dai, "An adaptive digital deskewing circuit for clock distribution networks," in *Visuals Supplement IEEE International Solid-State Circuits Conference (ISSCC 1998)*, 1998, pp. 326–327, 468–469.
- [32] G. Geannopoulos and X. Dai, "An adaptive digital deskewing circuit for clock distribution networks," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1998)*, 1998, pp. 400–401.
- [33] A. Chamas, A. Dalal, P. deDood, P. Ferolito, B. Frederick, O. Geva, D. Greenhill, H. Hingarh, J. Kaku, L. Kohn, L. Lev, M. Levitt, R. Melanson, S. Mitra, R. Sundar, M. Tamjidi, P. Wang, D. Wendell, R. Yu, and G. Zyner, "A 64 b microprocessor with multimedia support," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1995)*, 1995, pp. 178–179.
- [34] G. Konstadinidis, K. Normoyle, S. Wong, S. Bhutani, H. Stuimer, T. Johnson, A. Smith, D. Cheung, F. Romano, S. Yu, S.-H. Oh, V. Melamed, S. Narayanan, D. Bunsey, C. Khieu, K. Wu, R. Schmitt, A. Dumlaio, M. Sutura, J. Chau, K. Lin, and W. Coates, "Implementation of a third-generation 1.1-GHz 64-bit microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 11, pp. 1461–1469, 2002.
- [35] F. Anderson, J. Wells, and E. Berta, "The core clock system on the next generation Itanium™ microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2002)*, vol. 1, 2002, pp. 146–147, 453.
- [36] S. Rusu and S. Tam, "Clock generation and distribution for the first IA-64 microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2000)*, 2000, pp. 176–177.
- [37] M. Golden, S. Hesley, A. Scherer, M. Crowley, S. Johnson, S. Meier, D. Meyer, J. Moench, S. Oberman, H. Partovi, F. Weber, S. White, T. Wood, and J. Yong, "A seventh-generation x86 microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 11, pp. 1466–1477, 1999.
- [38] D. Draper, M. Crowley, J. Holst, G. Favor, A. Schoy, J. Trull, A. Ben-Meir, R. Khanna, D. Wendell, R. Krishna, J. Nolan, D. Mallick, H. Partovi, M. Roberts, M. Johnson, and T. Lee, "Circuit techniques in a 266-MHz MMX-enabled processor," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 11, pp. 1650–1664, 1997.
- [39] G. Gerosa, M. Alexander, J. Alvarez, C. Croxton, M. D'Addeo, A. Kennedy, C. Nicoletta, J. Nissen, R. Philip, P. Reed, H. Sanchez, S. Taylor, and B. Burgess, "A 250-MHz 5-w PowerPC microprocessor with on-chip L2 cache controller," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 11, pp. 1635–1649, 1997.
- [40] E. Cohen, J. Ballard, J. Blomgren, C. Brashears, V. Moldenhauer, and J. Pattin, "A 533 MHz BiCMOS superscalar microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 1997)*, 1997, pp. 164–165, 448.

- [41] E. Fayneh and E. Knoll, "Clock generation and distribution for intel Baniyas mobile microprocessor," in *Proceedings of Digest of Technical Papers VLSI Circuits 2003 Symposium*, 2003, pp. 17–20.
- [42] S. Tam, U. Desai, and R. Limaye, "Clock generation and distribution for the third generation Itanium® processor," in *Proceedings of Digest of Technical Papers VLSI Circuits 2003 Symposium*, 2003, pp. 9–12.
- [43] P. Mahoney, E. Fetzer, B. Doyle, and S. Naffziger, "Clock distribution on a dual-core, multi-threaded Itanium®-family processor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2005)*, 2005, pp. 292–293, 599.
- [44] D. Pham, S. Asano, M. Bolliger, M. Day, H. Hofstee, C. Johns, J. Kahle, A. Kameyama, J. Keaty, Y. Masubuchi, M. Riley, D. Shippy, D. Stasiak, M. Suzuoki, M. Wang, J. Warnock, S. Weitzel, D. Wendel, T. Yamazaki, and K. Yazawa, "The design and implementation of a first-generation CELL processor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2005)*, 2005, pp. 184–185, 592 Vol. 1.
- [45] S. Tam, J. Leung, R. Limaye, S. Choy, S. Vora, and M. Adachi, "Clock generation and distribution of a dual-core Xeon® processor with 16MB L3 cache," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2006)*, 2006, pp. 1512–1521.
- [46] J. Hart, K. Lee, D. Chen, L. Cheng, C. Chou, A. Dixit, D. Greenley, G. Gruber, K. Ho, J. Hsu, N. Malur, and J. Wu, "Implementation of a fourth-generation 1.8-GHz dual-core SPARC V9 microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, pp. 210–217, 2006.
- [47] D. Boerstler, "A low-jitter PLL clock generator for microprocessors with lock range of 340–612 MHz," *IEEE J. Solid-State Circuits*, vol. 34, no. 4, pp. 513–519, April 1999.
- [48] D. Boerstler and K. Jenkins, "A phase-locked loop clock generator for a 1 GHz microprocessor," in *Proceedings of Digest of Technical Papers VLSI Circuits 1998 Symposium*, 11–13 June 1998, pp. 212–213.
- [49] N. Kurd, J. Barkarullah, R. Dizon, T. Fletcher, and P. Madland, "A multigigahertz clocking scheme for the Pentium® 4 microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 11, pp. 1647–1653, Nov. 2001.
- [50] N. Bindal, T. Kelly, N. Velastegui, and K. Wong, "Scalable sub-10ps skew global clock distribution for a 90nm multi-GHz IA microprocessor," in *Visuals Supplement IEEE International Solid-State Circuits Conference (ISSCC 2003)*, 2003, pp. 281–284.
- [51] P. Restle, K. Jenkins, A. Deutsch, and P. Cook, "Measurement and modeling of on-chip transmission line effects in a 400MHz microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 4, pp. 662–665, April 1998.
- [52] J. Chang, S. Rusu, J. Shoemaker, S. Tam, M. Huang, M. Haque, S. Chiu, K. Truong, M. Karim, G. Leong, K. Desai, R. Goe, and S. Kulkarni, "A 130-nm triple- V_t 9-MB third-level on-die cache for the 1.7-GHz Itanium® 2 processor," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 1, pp. 195–203, Jan. 2005.

- [53] P. Gronowski, W. Bowhill, R. Preston, M. Gowan, and R. Allmon, "High-performance microprocessor design," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 5, pp. 676–686, May 1998.
- [54] G. Gerosa, S. Curtis, M. D'Addeo, B. Jiang, B. Kuttanna, F. Merchant, B. Patel, M. Taufique, and H. Samarchi, "A sub-1w to 2w low-power IA processor for mobile internet devices and ultra-mobile PCs in 45nm Hi-K metal gate CMOS," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2008)*, 2008, pp. 256–257, 611.
- [55] G. Gerosa, S. Curtis, M. D'Addeo, B. Jiang, B. Kuttanna, F. Merchant, B. Patel, M. Taufique, and H. Samarchi, "A sub-1w to 2w low-power IA processor for mobile internet devices and ultra-mobile PCs in 45nm Hi-K metal gate CMOS," in *Visuals Supplement IEEE International Solid-State Circuits Conference (ISSCC 2008)*, 2008.
- [56] S. Tam, S. Rusu, U. Nagarji Desai, R. Kim, J. Zhang, and I. Young, "Clock generation and distribution for the first IA-64 microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 11, pp. 1545–1552, Nov. 2000.
- [57] C. Dike, N. Kurd, P. Patra, and J. Barkatullah, "A design for digital, dynamic clock deskew," in *Proceedings of Digest of Technical Papers VLSI Circuits 2003 Symposium*, 12–14 June 2003, pp. 21–24.
- [58] K. Wong, T. Rahal-Arabi, M. Ma, and G. Taylor, "Enhancing microprocessor immunity to power supply noise with clock-data compensation," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 4, pp. 749–758, April 2006.
- [59] S. Rusu, S. Seidel, G. Woods, D. Grannes, H. Muljono, J. Rowlette, and K. Petrosky, "Backside infrared probing for static voltage drop and dynamic timing measurements," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2001)*, 2001, pp. 276–277, 454.
- [60] S. Tam, R. Limaye, and U. Desai, "Clock generation and distribution for the 130nm Itanium® 2 processor with 6-MB on-die L3 cache," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 4, pp. 636–642, April 2004.
- [61] P. Restle, R. Franch, N. James, W. Huott, T. Skergan, S. Wilson, N. Schwartz, and J. Clabes, "Timing uncertainty measurements on the Power5 microprocessor," in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2004)*, 2004, pp. 354–355.
- [62] P. Dudek, S. Szczepanski, and J. Hatfield, "A high-resolution CMOS time-to-digital converter utilizing a Vernier delay line," *IEEE J. Solid-State Circuits*, vol. 35, no. 2, pp. 240–247, Feb. 2000.
- [63] T. Hashimoto, H. Yamazaki, A. Muramatsu, T. Sato, and A. Inoue, "Time-to-digital converter with vernier delay mismatch compensation for high resolution on-die clock jitter measurement," in *Proceedings of IEEE Symposium on VLSI Circuits*, 18–20 June 2008, pp. 166–167.
- [64] P. Teehan, M. Greenstreet, and G. Lemieux, "A survey and taxonomy of GALs design styles," *IEEE Design & Test of Computers*, vol. 24, no. 5, pp. 418–428, Sept.–Oct. 2007.
- [65] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, and N. Borkar,

- “An 80-tile 1.28TFLOPS network-on-chip in 65nm CMOS,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2007)*, 2007, pp. 98–99, 589.
- [66] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar, “An 80-tile sub-100-w TeraFLOPS processor in 65-nm CMOS,” *IEEE Journal of Solid-State Circuits*, vol. 43, no. 1, pp. 29–41, Jan. 2008.
- [67] N. Kurd, J. Douglas, P. Mosalikanti, and R. Kumar, “Next generation Intel® micro-architecture (Nehalem) clocking architecture,” in *Proceedings of IEEE Symposium on VLSI Circuits*, 18–20 June 2008, pp. 62–63.
- [68] A. Mule, S. Schultz, T. Gaylord, and J. Meindl, “An optical clock distribution network for gigascale integration,” in *Proceedings of IEEE 2000 International Interconnect Technology Conference*, 5–7 June 2000, pp. 6–8.
- [69] A. Mule, E. Glytis, T. Gaylord, and J. Meindl, “Electrical and optical clock distribution networks for gigascale microprocessors,” *IEEE Transactions of VLSI Systems*, vol. 10, no. 5, pp. 582–594, Oct. 2002.
- [70] Q. Zhu and S. Tam, “Package clock distribution design optimization for high-speed and low-power VLSIs,” *IEEE Transactions on Components, Packaging, and Manufacturing Technology, Part B: Advanced Packaging*, vol. 20, no. 1, pp. 56–63, Feb. 1997.
- [71] V. Gutnik and A. Chandrakasan, “Active GHz clock network using distributed PLLs,” *IEEE Journal of Solid-State Circuits*, vol. 35, no. 11, pp. 1553–1560, Nov. 2000.
- [72] F. O’Mahony, C. Yue, M. Horowitz, and S. Wong, “A 10-GHz global clock distribution using coupled standing-wave oscillators,” *IEEE Journal of Solid-State Circuits*, vol. 38, no. 11, pp. 1813–1820, Nov. 2003.
- [73] S. Chan, P. Restle, K. Shepard, N. James, and R. Franch, “A 4.6ghz resonant global clock distribution network,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2004)*, 2004, pp. 342–343, Vol. 1.
- [74] S. Chan, P. Restle, T. Bucelot, S. Weitzel, J. Keaty, J. Liberty, B. Flachs, R. Volant, P. Kapusta, and J. Zimmerman, “A resonant global clock distribution for the cell broadband-engine™ processor,” in *Digest of Technical Papers IEEE International Solid-State Circuits Conference (ISSCC 2008)*, 2008, pp. 512–513.



<http://www.springer.com/978-1-4419-0260-3>

Clocking in Modern VLSI Systems

Xanthopoulos, T. (Ed.)

2009, XXV, 320 p., Hardcover

ISBN: 978-1-4419-0260-3