Reference-Based Clock Distribution Architectures

Atanu Chattopadhyay
McGill University
Department of Electrical and Computer Engineering
Montreal, Canada

Zeljko Zilic
McGill University
Department of Electrical and Computer Engineering
Montreal, Canada

Abstract— **This paper examines the use of clock distribution architectures employing a reference-based skew compensation technique. For each clock domain, a bi-directional clock line is daisy-chained using specially designed switches at each tap in the distribution. Daisy-chaining the clock decreases the clock load by eliminating the redundant paths used to equalize delays in traditional H-tree distributions. Clock skew is accounted for by actively synchronizing each local clock to a position directly between forward and reverse-moving reference clocks. This reference-based clocking strategy achieves a set of skew-tolerant clocks at each tap in a daisy-chain. The design provides simple-to-layout and scalable multi-point skew compensation useful for large designs. The implementation of a reference-based clocking chain is outlined, followed by the description of single clock and multi-clock architectures using this design strategy.**

I. INTRODUCTION

Clock distribution represents an increasingly difficult challenge due to progressively more complex systems, decreased power supply voltages, larger die sizes and higher clock frequencies [1,2]. There are a number of sometimes conflicting characteristics that need to be balanced to create an adequate clock distribution network (CDN). These considerations include clock signal characteristics such as fast transition times, a balanced duty cycle and low clock skew [3]. Traditionally, passive forms of clock skew reduction were used to balance all the leaves in a CDN by a combination of matching wire length and adjusting clock buffer delays [4,5]. In modern systems, clock tree balancing techniques are becoming insufficient to eliminate skew since clock buffer mismatches and in-die process variations have become a limiting factor in maintaining tight skew tolerance [6,7,8]. Floorplanning requirements add additional complication to the clock distribution, since matching clock trace lengths over irregularly shaped domains is difficult. Typical clock skew budgets use a 10% of clock period standard [9,10].

In this paper, we examine a clock distribution scheme for single and multi-clock systems. By actively synchronizing each local clock to a position directly in between forward and reverse-moving reference clocks, we obtain a set of skew-free clocks at each tap of the distribution line. The method is based on an idea first introduced in [11], which provides multi-point skew compensation, with skew bound performance limited by the process variation amongst the independent PLL-based skew compensating buffers. Their scheme uses a 3-wire method with separate raw clock, forward reference line and reverse reference line, and the reference lines tied at the far end of the clock distribution. However, this technique calibrates the clock network for a reference line that is distinct from the line upon which the clock is distributed. As such, [11] is susceptible to skew from wire mismatches between the clock reference lines and the raw clock line which each must be sized differently to reflect their different tasks. The lack of buffering in the clock path limits the total load, the distance between taps and the number of taps that the reference line can spawn. The PLLs used also consume considerable power and area.

Our method improves upon [11], using delays lines instead of PLLs and combining the three reference lines into one bi-directional line. Simulations show that the proposed CDN is scalable, compatible with irregularly shaped distribution areas, and combines low power operation with very tight skew bounds. Our skew-tolerant reference clocking scheme is also easy to lay out since it automatically compensates for process and line length variances in the global clock spine used to distribute the clock to each local tap. While [2] and [4] are based on a similar idea, their use of different forward and reverse clock lines for each tap can lead to in-die process variation and possible trace length discrepancies that can hurt the effectiveness of the skew compensation circuitry. In addition, [2] and [4] perform skew compensation at the source independently for each leaf. This creates a need for long and varying length reference lines from every synchronization point to the source. These feedback lines are subject to the same trace matching discrepancies present with H-trees, introducing error to the skew compensation technique [12]. Our technique uses the same reference line for every tap, so any local process variation present in the line affects all taps equally, and is thus not a source for additional skew.

The paper is organized as follows. Section II describes reference-based clocking and Section III provides several applications scenarios.

1-4244-0173-9/06/$20.00 ©2006 IEEE.

II. REFERENCE-BASED CLOCKING

A reference-based clocking thread is divided into *n* smaller subsections, each of which is connected to a tap. Each of these subsections should be roughly equal in size to help match tap-to-leaf delays for every tap. The clock taps do not need to be distributed in close proximity, nor do they need to be regularly spaced. The structure of an 8-tap clock distribution is shown in Figure 1. A single bi-directional clock line is threaded as needed throughout the clock domain and shared between the forward and reverse clocks during synchronization using specially designed switches. Both the region 1 input node (forward clock) and the region *n* output node (reverse clock) are tied to the global clock. Separate fine and coarse grain delay blocks are used to implement the requisite delay line, as shown in Figure 2. The fine grain delay block is implemented using four smaller delay blocks grouped in pairs of two to span a sufficient delay range to encompass one coarse grain delay increment. The clock distribution network requires three distinct phases to work properly: synchronization, calibration and operation. These are discussed in the following sections.

A. Synchronization

During the synchronization phase, each clock tap is sequentially calibrated, starting with the region 1 tap and ending with the region *n* tap. The switches are configured so that a tap sees both the forward and reverse clocks during its synchronization interval. Since the clock distribution line has a constant delay (*K*) over its entirely length, if the delay of the forward clock is δ, then the delay of the reverse clock will be K-δ. By synchronizing each local clock to a position directly in between the forward and reverse clocks - the average between their clock edges - the resulting clock is located at:

$$\frac{(K-\delta)+\delta}{2}=\frac{K}{2} \tag{1}$$

for all the taps because the delta terms cancel out. The forward and reverse delays are all taken modulo the clock period to account for the periodic nature of the signals.

Synchronization for each clock thread can be achieved by predicting what each source-to-tap delay might be and hard-coding the delay line setting required to compensate for it into the system. This method has a number of benefits including zero synchronization time and reduced circuitry overhead by eliminating the reverse path of the clock thread, the phase detector and the control circuitry. However, it does not compensate for any process variation induced skew, so it is only useful for distributing irregularly shaped clock domains where clock skew is not a great concern. To fully utilize reference-based clocking, online skew calibration needs to be included.

Each tap in this configuration consists of a 2:2 switch, a delay line, a phase detector and control circuitry. The forward clock is delayed using two identically set delay lines - a source delay line and a local delay line - to align with the reverse clock. Removing the source delay line from the clock path provides the midpoint between the forward and reverse reference lines clock at this tap. Each local delay line is required to save the delay setting determined during synchronization. The clock distribution is configured so that the source delay line is reused among all of the clock taps so only need *n*+1 delay lines are needed. One possible run-time synchronization approach would be to, beginning with the closest tap to the clock source, verify each coarse grain setting from fastest to slowest, using the longest fine grain setting for each test. The phase detector is designed to indicate when the delay line setting is correct, too slow (*UP* signal is asserted) or too fast (*DOWN* signal is asserted). The coarse grain settings can be set to increment if an *UP* signal is found. The coarse grain setting is finalized once the first *DOWN* signal is found. To synchronize the fine grain delay, an appropriate strategy would be to use a binary search to traverse the fine grain delay settings using the same *UP* and *DOWN* signals to control the search.

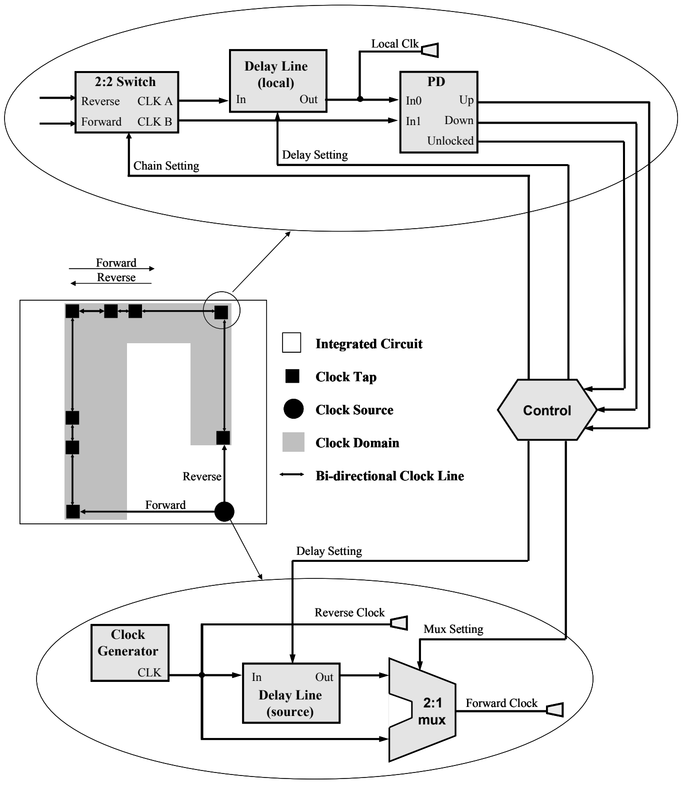

Figure 1. Proposed clock distribution network for 8 clock taps.

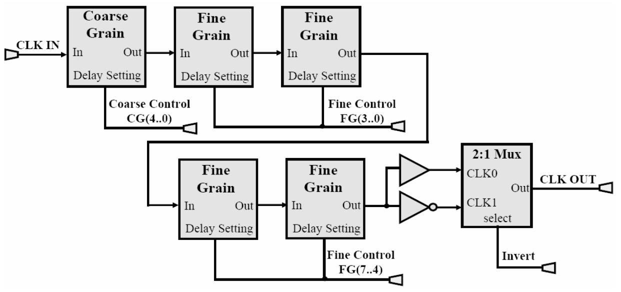

Figure 2. Variable delay cell capable of periodic delays.

B. *Calibration*

Once all the taps have been synchronized, the source delay element is disabled, leaving the undelayed forward clock to propagate through the entire forward path of the clock chain for the calibration and operation phases. Because of the nature of the synchronization method, we end up with 2 possibilities for each local synchronized clock: either the local clock is correct or it is inverted, depending on the relative phase of the reference clocks at the tap. Some circuits, such as those that operate on both edges of a clock, may not require a calibration phase. To perform the polarity adjustment should it be required, the system could examine each pair of adjacent clocks individually and invert the higher order (number) clock, if needed, starting with the output of taps 1 and 2 and ending with taps n-1 and n. Another method would be to propagate a fixed logic level through the clock chain and invert the output of every tap that does not generate the expected polarity.

C. *Operation*

Following calibration, the components of the clock distribution that are unused during operation such as the phase detection and calibration circuitry are disabled to save power. Adding a constant delay to every tap in the CDN does not change the synchronization of the network. As such, any delay mismatch between the delayed and undelayed modes of the source clock simply delays or advances all of the clocks by a constant amount, but the taps remain synchronized. Sharing a single wire for distributing forward and reverse clocks eliminates any wire-related process variation. Assuming locally negligible intra-die process variation assures constant forward path and reverse path delays through a 2:2 switch. Any chip-to-chip variation is not an issue since each CDN is calibrated independently. The primary source of skew for our method is the in-die process variation that can occur between the source and local delay lines. However, this is a single instance of variation which is acceptable when one considers that a conventional H-tree has variances that accumulate as a signal propagates through the distributed buffers within the CDN.

The circuits required to create this reference-based clock distribution have been designed for TSMC's 0.18 μm P-well process using the Cadence Virtuoso design environment and simulated with SpectreS and the Analog Artist simulation tool. The performance of the system is implementation-dependent; with our particular circuit implementation using the delay line in Figure 2 capable of distributing frequencies between 456 MHz and 1.90 GHz, with a worst-case skew of under 4%, or +/-10 ps, for all frequencies. Simulations show that for a typical 8-tap clock distribution, power consumption at the maximum frequency is 33.2 mW during the phase alignment cycle and 18.0 mW during run time. Figure 3 shows the output clocks for an 8-tap distribution operating at 891 MHz. In our implementation, there are approximately 300 transistors per tap.

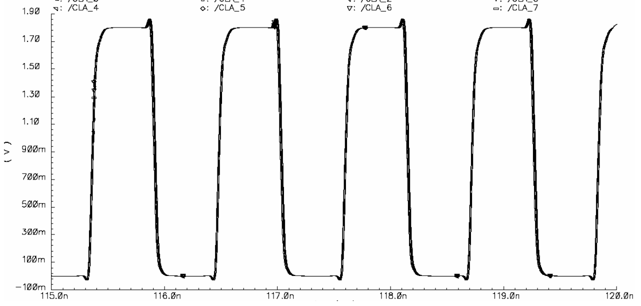

Figure 3. Resulting low-skew output clocks for an 8-tap system.

III. Using Our Clock Distribution

A. *Components*

A small library of components is required to implement this distribution. A delay line, phase detector, bi-directional clock routing switch and other basic components such as logic gates and a 2:1 multiplexer are needed. These can be designed and laid out for a given technology and replicated as needed like standard cells. Getting good delay line resolution is important since the skew bound achieved is roughly equal to the delay increment. Equal low-to-low and high-to-high delay times through the delay line are critical for correct operation of the circuitry and should be verified throughout the delay setting range.

The performance of the system is dependant on good matching between the delay lines. Since each tap is replicated, there is little variation between the delay lines due to layout. Care needs to be taken in laying out the phase detector since any variation in trace length within the detector may alter the result. For the 2:2 switches, the layout should be constructed to obtain equal left-to-right and right-to-left delays for the symmetric element. For all the other components, only equal high-to-low and low-to-high propagation delays are important, since any other variances are compensated for by the skew-reduction hardware. Finally the clock traces need to be kept short enough to preserve full swing clocks. Should "clipping" occur in the clock signal, the duty cycle of the local clocks will not be ideal unless high and low clipping is carefully balanced.

B. *Architecture*

1) *Single clock*

To implement a single clock distribution on an ASIC, few changes are required to the underlying design. When floorplanning, the clock domains need to be divided into roughly equally sized sub-regions, each of whom will require a traditional H-tree to distribute the clock from tap-to-leaf. Due to much smaller distances from synchronized source-to-leaf, this represents a much more skew-tolerant signal. As such, our method provides designers with a more forgiving approach to clock distribution. Sizing these sub-regions will represent a compromise between area and skew tolerance: the smaller the area, the more taps will be required, but the more skew-tolerant the leaves will be due to closer proximity to the skew-adjusted source.

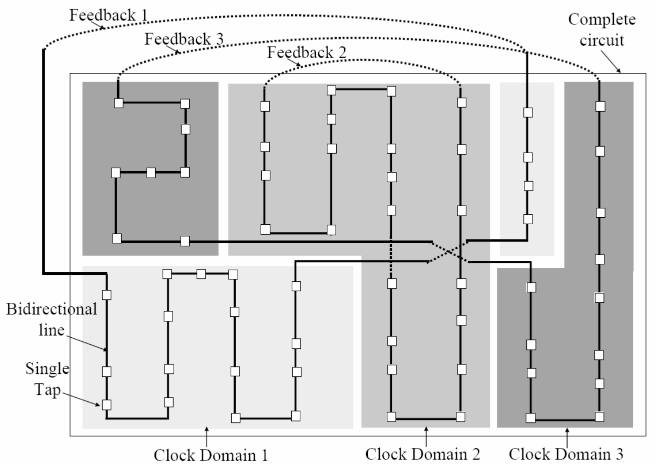

Figure 4. A 3 clock domain reference-based distribution.

2) *Multi-clock*

Distributing multiple clocks requires multiple instances of these threads. Figure 4 shows a 3-clock domain distribution. These threads can cross other clock domains easily and are very easy to lay out since the taps do not need to be placed at regular intervals. Irregularly-shaped clock regions like the ones shown in Figures 1 and 4 are much easier to implement using our method since adjacent areas are all connected linearly. This is also true for disjointed distributions, which can simply be connected with a wire with little attention paid to the wire length. This trait is convenient for pin-limited designs where it may not be desirable to locate a large sub-circuit along a chip perimeter. Instead a small block can be placed near the edge to control information flow and associated to a large block elsewhere on the chip, while still maintaining one skew-tolerant synchronous clock domain. Irregularly shaped clock regions would be much more difficult to implement using an H-tree, requiring additional care to match wire length, but it is performed automatically by our reference-based architecture.

3) *Wire Length*

Because of the flexibility of these multi-clock architectures, complex circuits can be created with little thought towards clock layout. Instead, circuits can be placed as is convenient for the designer and connected together using our skew-tolerant daisy-chains. Each domain also needs to be divided into roughly equally sized sub-regions that each becomes associated to a single tap in the distribution line. Implementing a multi-clock reference-based clock distribution is not much more complicated than a single clock distribution. Each component in the clock thread is simply replicated as needed, including an extra global clock generator for each domain. The clock generators can either be placed in close proximity in one area of a chip and suitably routed to and back from the clock thread, or spread out over the chip. A single clock generator can also be connected to a series of dividers (or multipliers) to drive all the clock lines if the required frequencies are suitable matched. The only constraint is that the clock must have 50% duty cycles and be connected to both the head and the

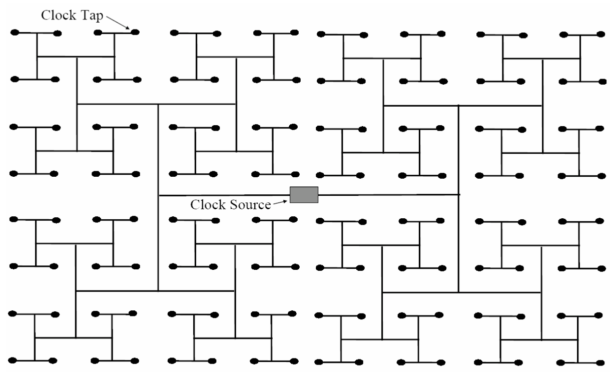

Figure 5. A 128-tap H-tree.

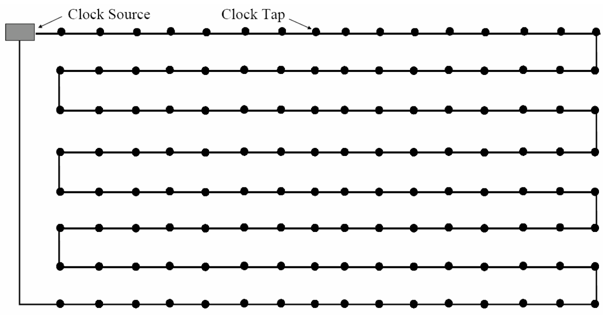

Figure 6. A 128-tap reference based clock distribution.

tail of the clock thread. Before entering the thread, clock skew is not an issue since it is compensated for within the architecture, so clock buffering can be used here as required. Synchronization and calibration can either be performed simultaneously for every domain using dedicated controllers, or sequentially using a single controller. Because the reference-based distribution can simply route a clock in the shortest possible path to each tap, there is significant wire savings with respect to a traditional H-tree that requires redundant wires to match the path length from source to every tap. Reference-based clock distribution can contain an arbitrary number of nodes, whereas ideally constructed H-trees require a power of 2 to be properly balanced. Figures 5 and 6 show how a 128-tap H-tree distribution compares with our reference-based one. A decrease in wire length will reduce clock load. Replacing a 3-level H-tree containing 64 taps with our reference-based distribution can save 16.7% in wire length. Table 1 shows a summary of the wire lengths (in units) that can be achieved between different depths of square H-trees and comparably-sized referenced-based clocking solutions assuming 1 unit length spacing between each of the clock taps. The wire length numbers assume 2^{n-1} length wires for each level of the tree [13]. These numbers ignore the tap-to-leaf distributions, which will be identical for both implementations. The wire length is governed by Equation 2 for reference-based clocks and Equation 3 for H-tree clocks.

$$Length_{reference-based} = (2^n - 1)(2^n + 2) \qquad (2)$$

$$Length_H = 3 \cdot 2^{n-1} \cdot (2^n - 1) \qquad (3)$$

TABLE I. WIRE LENGTH COMPARISON

Largest H-tree level	Number of taps	Wire length (H-tree)	Wire length (reference)	Savings (%)
1	4	3	4	-33.33
2	16	18	18	0.00
3	64	84	70	16.67
4	256	360	270	25.00
5	1024	1488	1054	26.81
6	4096	6048	4158	31.25
7	16384	24384	16510	32.29
8	65536	97920	65790	32.81
9	262144	392488	262654	33.08
10	1048576	1571328	1049598	33.20

C. Control

There are many approaches to designing a control system for our reference-based clocking system. The only essential piece of hardware is memory to retain the delay setting for each delay line: this is on the order of one or two bytes for each tap. This delay line setting is used for both local and source delay lines to achieve the required forward clock delay (see Equation 1, and Figure 1). A controller also needs to access the 2-bit direction control of each 2:2 switch during synchronization. These control lines will change in a regular pattern, with a reverse direction initially, a synchronization state (when both forward and reverse clocks propagate to the tap) next, and a forward direction finally. So, any controller has to have access to the delay line settings of the tap, the direction control of the 2:2 switches in the chain, the result bits from the phase detector and the multiplexer setting bit for the clock source. By propagating the control signals produced by the phase detector and not the reference clocks themselves, our solution eliminates another source of skew when compared to other solutions like [2]. The controller can be constructed using any combination of hardware or software-based methods. The controller delay can be as fast as one clock cycle for a completely hardware-based solution. Look-up tables represent the easiest way to traverse the delay setting possibilities. However, some designers might prefer to use software-based synchronization with a microprocessor or other controller for which the delay would be controller-dependant. Since the controller should operate at a low frequency to allow for adequate resolution time in the phase detector, skew is much less of a concern for the control circuitry's clock. Alternative implementations include using a completely asynchronous controller solution or using an independent clock during programming.

IV. CONCLUSION

We have designed a skew-free multi-point clock distribution. By using delay lines instead of PLLs and with a single reference and distribution line, our method is efficient enough to be replicated for multi-clock applications, using one reference-based clocking chain per clock. The use of a single common forward and reverse clock reference line to cut down on in-die process variation skew is unique. Clock domains can be irregularly shaped and intertwined; leading to maximum flexibility in the way a design can be floorplanned onto an integrated circuit. Using a daisy-chain approach minimizes the required clock load, resulting in power savings. The system can also be used to provide beneficial skew between each of the local taps in the IC [14]. Any reduction in clock skew obtained by our clock distribution is extremely useful since this time can be added directly to the available cycle time within a clock period [6].

REFERENCES

[1] R. L. Aguiar and D. M. Santos, "Wide-area clock distribution using controlled delay lines," Proc. of IEEE Intl. Conf. on Electronics, Circuits & Systems (ICECS 1998), vol. 2, 63-66.

[2] A. Kapoor, N. Jayakumar and S. P. Khatri, "A novel clock distribution and dynamic de-skewing methodology," Proc. of IEEE/ACM International Conference on Computer Aided Design (ICCAD 2004), 626-631.

[3] C.-Y. Yang and S.-I Liu. "A one-wire approach for skew-compensating clock distribution based on bidirectional techniques," IEEE Journal of Solid-State Circuits, vol. 36, issue 2 (Feb. 2001), 266-272.

[4] H. Lee, H. Q. Nguyen and D.W. Potter, "Design self-synchronized clock distribution networks in an SoC ASIC using DLL with remote clock feedback," Proc. of IEEE Intl. ASIC/SOC Conference (ASICSOC 2000), 248-252.

[5] D.E. Brueske and S.H.K. Embabi, "A dynamic clock synchronization technique for large systems," IEEE Trans. on Components, Packaging, and Manufacturing Technology, Part B: Advanced Packaging, vol. 17, issue 3 (Aug. 1994), 350-361.

[6] S. Tam, S. Rusu, U. Nagarji Desai, R. Kim, Ji Zhang and I. Young, "Clock generation and distribution for the first IA-64 microprocessor," IEEE Journal of Solid-State Circuits, vol. 35, issue 11 (Nov. 2000), 1545-1552.

[7] S. Zanella, A. Nardi, A. Neviani, M. Quarantelli, S. Saxena and C. Guardiani, "Analysis of the impact of process variations on clock skew," IEEE Trans. on Semiconductor Manufacturing, vol. 13, issue 4 (Nov. 2000), 401-407.

[8] E.G. Friedman, "Clock distribution networks in synchronous digital integrated circuits," Proceedings of the IEEE, vol. 89, issue 5 (May 2001), 665-692.

[9] M. Omana, D. Rossi and C. Metra, "Fast and low-cost clock deskew buffer," Proc. of IEEE Intl. Symposium on Defect and Fault Tolerance in VLSI Systems (DFT 2004), 202-210.

[10] V. Varghese, T. Chen and P. Young, "Stability analysis of active clock deskewing systems using a control theoretic approach," Proc. American Control Conf. (ACC 2005), vol. 3, 1758-1763.

[11] W.D. Grover, J. Brown, T. Friesen and S. Marsh, "All-digital multipoint adaptive delay compensation circuit for low skew clock distribution," Electronics Letters, vol. 31, issue 23 (9 Nov. 1995), 1996-1998.

[12] M. Saint-Laurent, M. Swaminathan and J.D. Meindl, "On the micro-architectural impact of clock distribution using multiple PLLs," Proc. of Intl. Conf. Comp. Design (ICCD 2001), 214-220.

[13] I. Chanodia and D. Velenis, "Effects of parameter variations and crosstalk on H-tree clock distribution networks," Proc. of 48th Midwest Symposium on Circuits and Systems (MWSCAS 2005), 547- 550.

[14] Hong-Yean Hsieh, Wentai Liu, M. Clements and P. Franzon, "Self-calibrating clock distribution with scheduled skews," Proc. Intl. Symp. Circuits Systems (ISCAS 1998), 470-473.