Clock Tree Generation by Abutment in Synchoros VLSI Design

Dimitrios Stathis*, Panagiotis Chaourani*, Syed M. A. H. Jafri*, Ahmed Hemani*
* KTH Royal Institute of Technology, Stockholm, Sweden (e-mail: stathis, pancha, jafri, hemani@kth.se)

ABSTRACT

Synchoros VLSI design style has been proposed as an alternative to standard cell-based design. Standard cells are replaced by synchoros, large grain, VLSI design objects called SiLago (Silicon Lego) blocks. This new design style eliminates the need to synthesise ad hoc wires of any type: functional and infrastructural. SiLago blocks are organised into region instances. Communication among SiLago blocks in a region instance is synchro$\underline{n}$ous and happens over regional NoCs whose fragments are also absorbed into SiLago blocks. Consequently, the regional NoCs get created by abutment of SiLago blocks. The clock tree that is used in a region is called regional clock tree **(RCT)**. Synchoros design style requires that the RCT, like the regional NoCs, be created by abutting its fragments that are absorbed within the SiLago blocks. The RCT created by abutment is not an ad-hoc clock tree but a structured and predictable design with known cost metrics. The design of such an RCT is the focus of this paper. The scheme is scalable, and we demonstrate that the proposed RCT can be generated for valid VLSI designs of ~1.5 million gates. The RCT created by abutment is correct by construction, and its properties are predictable. We have validated the generated RCTs with static timing analysis to validate the correct-by-construction claim. Finally, we show that the cost metrics of the SiLago RCT is comparable to the one generated by commercial EDA tools.

KEYWORDS

CGRA, CTS, EDA, SiLago, VLSI Design, Synchoricity

1 Introduction

This paper presents a clock tree generation scheme for a novel synchoros VLSI design framework. Synchoros VLSI design was proposed as an alternative to the standard cell-based VLSI design [1]. The word synchoros is derived from the Greek word for space – χώρος (khôros). Synchoricity is analogous to synchro$\underline{n}$icity[1]. In synchro$\underline{n}$icity, time is uniformly discretised with clock ticks to simplify the temporal and logical composition. In synchoricity, space is uniformly discretised with a virtual grid to simplify the spatial and electrical composition.

In standard cells based design, the cells themselves are pre-designed, but their placement and wires needed to connect, clock, power, and reset them must be synthesised anew in an ad-hoc fashion. This makes the cost metrics unpredictable for synthesis from higher abstractions. In synchoros design style, all wires – functional and infrastructural – are absorbed in SiLago (Silicon Lego) blocks. SiLago blocks replace standard cells as atomic building blocks. All inter-SiLago block wires are bought to the periphery at the right place and right metal layer to enable composition by abutting valid neighbours. This eliminates the need to synthesise ad-hoc wires and makes the cost-metrics predictable as long as the mapping of functionality to SiLago blocks and its topological arrangement is known. This paper focuses on how one category of wire – the regional clock tree (**RCT**) – is absorbed in the SiLago block, and a *valid* and *predictable* RCT gets created by abutment. In synchoros VLSI design style, there are three levels of hierarchical objects: Local, Regional and Global.

SiLago blocks are at the lowest level of the hierarchy. All wires in SiLago blocks, including the Local Clock Tree (**LCT**), are ad-hoc and synthesised by EDA tools. Regions are composed of SiLago blocks of the same type, Figure 1. Inter-SiLago block communication in a region happens over regional NOCs, whose wires are also absorbed within the SiLago blocks. The RCT drives the LCTs in each SiLago block. The RCT fragments and ancillary circuit to maintain slew and minimise skew are also absorbed in each SiLago block. As stated, this is the focus of this paper. The RCT must maintain synchro$\underline{n}$icity amongst SiLago blocks in a region, as shown in Figure 2, where an inter-SiLago block timing path created by a *regional NoC* is clocked by two local clocks. Region instances communicate with each other over Global NOCs, also composed in terms of SiLago blocks [1]; see Figure 1. Global NOCs have their own Global Clock Tree (**GCT**).

2 RCT by abutment in synchoros VLSI Design

This section presents the requirements imposed by synchoricity on the RCT and how these requirements are fulfilled, including creating a valid and predictable design that factors in process, voltage, and temperature **(PVT)** and On-Chip variations **(OCV)** challenges.

2.1 RCT requirements in SiLago Design Flow

The RCT has two types of requirements. One is common to all clock tree synthesis, i.e., to minimise the clock skew and maintain sufficient drive strength to ensure that the slew rate is not violated. The second set of requirements stems from synchoricity; it imposes two additional constraints predictability and validity. An RCT composed of its fragments should be predictable. The predictability requires that the RCT can only be composed of a finite set of pre-characterised fragments absorbed within the

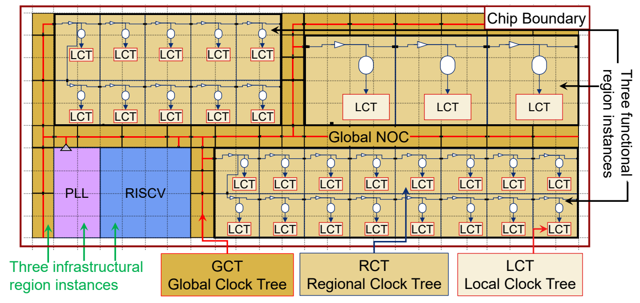
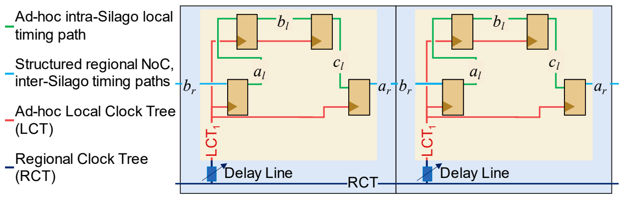


Figure 1. Example synchoros VLSI design instance

[1] Synchoricity is a different concept from synchro$\underline{n}$icity, and not a typo. We highlight the difference between the two words by underlining the $\underline{n}$ in synchro$\underline{n}$icity.

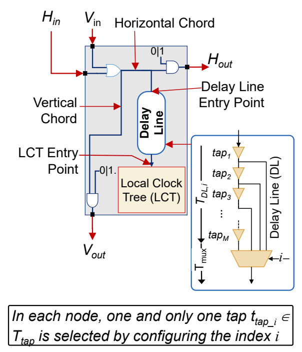

Ad-hoc intra-Silago local timing path
Structured regional NoC, inter-Silago timing paths
Ad-hoc Local Clock Tree (LCT)
Regional Clock Tree (RCT)
b_l
c_l
a_l
b_r
a_r
LCT$_1$
Delay Line
RCT
Figure 2. Local & regional clocks and timing paths.

SiLago blocks. We discuss such a timing model in section 2.3. The parameters of this model factors in PVT and OCV. The RCT generated by abutment should also be valid, i.e., fulfil the technology design rules and be manufacturable.

2.2 RCT Components in each SiLago Block

In this sub-section, we elaborate the components of RCT that are absorbed within each SiLago block. These components enable abutment, maintain slew rate and minimise skew.

1. *Standardised Entry and Exit Points*: Every SiLago block type has a standard entry (H_{in}, V_{in}) and exit (H_{out}, V_{out}) points for the RCT fragment; see Figure 3. Standard implies fixed location on the specific edge of SiLago block and metal layers. The entry and exit points of RCT fragments in neighbouring SiLago blocks abut to create a valid RCT. Since the neighbours can be in horizontal, vertical, or both dimensions, SiLago blocks need entry and exit points on both edges. The RCT can be distributed in top-down and left-right, or bottom-up and right-left. The choice of orientation depends on the corner at which the global clock tree **(GCT)** enters. The GCT entry point depends on the floor planning of the global NoCs during the syntheses from higher abstractions.
2. *Multiplexed and buffered horizontal and vertical chords*: These components select the RCT input and output and maintain the slew rate. Selecting the input implies selecting the H_{in} or $V_{in,}$ as shown in Figure 3. The two inputs are fed to an OR gate. Only one of the inputs can be a clock, and the other is set to zero when configuring the RCT. Selecting the output implies selecting if the RCT is to be propagated to the right exit (H_{out}), or the bottom exit (V_{out}), or both. The unselected exit is grounded using two AND gates. Depending on the two AND gates' configuration, the variants of chord delay, $T_{RCT\_chord,}$ is selected; see Figure 3. These gates also serve as the drivers to maintain the clock's slew.
3. *Programmable Delay Line*: The programmable delay line adjusts the delay to the local clock tree **(LCT)** entry point; see Figure 3.



SiLago Block Delay Notation	
T_{RCT_chord}	$T_{V\_to\_V}$, $T_{V\_to\_H,}$ $T_{H\_to\_H}$, $T_{H\_to\_V}$ Four variants of delay through horizontal and vertical chords
$T_{V\_to\_DL}$, $T_{H\_to\_DL}$	Delay of the wire connecting the V_{in} or H_{in} and the Delay Line Entry Point
T_{DL_i}	Delay imposed by delay line depending on the index i
T_{mux_i}	Propagation delay through the mux depending on the i index
t_{tap_i}	Delay between the V_{in} or H_{in} and the LCT Entry Point associated with the selected tap i of the Delay Line $t_{tap\_i} = [T_{H\_to\_DL} \mid T_{v\_to\_DL}] + T_{DL\_i} + T_{mux\_i}$
T_{tap}	Set of delays associated with taps in the Delay Line $\{t_{tap\_0}, \ldots, t_{tap\_i}, \ldots, t_{tap\_(M-1)}\}$

Figure 3. Components of RCT fragment and their notations.

The delay is adjusted according to the SiLago block's position in a region instance with respect to where the GCT enters the region. The objective of adjusting the delay is to minimise the skew of the clock's arrival at the LCT entry points. The delay is adjusted by selecting a tap, with index i, in the delay line. The selected tap i introduces a delay t_{tap_i} between the SiLago Block's RCT entry point, i.e., H_{in} or V_{in}, and the LCT entry point.

The RCT fragments' design in terms of the pieces described above is identical for all SiLago block types. However, the dimension of the three components depends on the type of SiLago block. The entry and exit points of the SiLago blocks are like the Lego studs. These points' positions are a standard offset from top-left and bottom-right corners, irrespective of the type of SiLago blocks. The standard position makes it possible to abut SiLago blocks of different types. By having standardised offsets from the corners, the spatial composition of blocks becomes feasible.

The length of the horizontal and vertical chords is adjusted to match specific SiLago block types' dimensions. The drive strength of the three gates used for configuring the propagation path is also dimensioned to match their respective loads. These loads are not arbitrary but known a priori. This is true because a SiLago block can have a finite number and types of neighbours. When a SiLago block type is designed, its connection to other possible block types is characterised. The characterisation ensures that the electrical connection between them is valid.

2.3 Regional Clock Tree Delay Model

The RCT delay model enables post-layout accurate predictability of electrical and temporal properties of an arbitrary RCT created by the abutment of its fragments. This model is used to decide how large a region instance can be and select the taps in the delay line to balance each SiLago block's skew.

The delay model captures the RCT latency from the entry point of a region instance to the LCT entry point of the SiLago blocks in a region. Figure 4 presents an example of how the delay model is applied. It illustrates a region instance with 8 SiLago blocks, where the RCT entry point is on the top left corner. Each block will have a unique RCT arrival time, notated as $T_{LCT\_x,i}$. The subscript x identifies the block id, and subscript i identifies the programmable delay line's selected tap index. The programmable line aims to make $T_{LCT\_x,i,}$ as equal as possible for every block x. The $T_{LCT\_x,i}$ has two components, EQ. 2:

a. One is the propagation delay in the intervening RCT chords between the RCT entry point in a region instance and the entry to a SiLago block x. This delay is denoted by T_{prop_x}, where x is SiLago block id (EQ. 1). In Figure 4, x=*7* and $T_{prop.x}$ is shown as the thick red line, colouring the chords in blocks 1, 5, and 6.
b. The second component is the delay imposed on the RCT by the delay line in the SiLago block x. This delay is represented by $t_{tap\_i,x}$, where i is the tap index. In Figure 4 this delay component is shown as $t_{tap\_i,7}$ in a thick golden line in block 7. The $t_{tap\_i,x}$ is further divided into three sub-components shown in Figure 3.

$$T_{prop\_x} = \sum_{nodes\ previous\ to\ x} T_{RCT\_chord} \qquad \text{EQ. 1}$$

$$T_{LCT\_x,i} = T_{prop\_x} + t_{tap\_i,x} \qquad \text{EQ. 2}$$

The SiLago design flow involves two phases. In the first, the blocks are designed, verified, and characterised as a one-time effort. A partial list of outputs that are produced in this phase are:

1. The RCT fragments are analysed and characterised to extract the parameters used in the RCT timing model presented above.
2. The fragments of regional NoCs are also analysed and characterised in a manner like the RCT fragments. Examples of these fragments are shown as a_r and b_r in Figure 2. The timing analysis for such inter-SiLago flop-to-flop timing paths is done for each SiLago block type in conjunction with valid neighbour block types. Such timing analysis is used to characterise the wire fragments' delays. The validation of such timing paths cannot be done until RCT is created when a design is assembled in terms of the SiLago blocks during the second phase. The inter-SiLago timing path example shown in Figure 2 spans to the nearest neighbours. In practice, such timing paths can exist in a small contiguous cluster of SiLago blocks in a region instance. The size of the cluster depends on the NoC that is used to connect the SiLago blocks. The SiLago blocks do not communicate over ad hoc wires but over a structured NoC based interconnect, for example [2].
3. The intra-SiLago, i.e., local timing paths (flop-to-flop), are analysed by the static timing analysis **(STA)** that is part of the standard cell-based design flow. Examples of these paths are shown as a_l, b_l, and c_l in Figure 2; the subscript l stands for local.

In the second phase, the library of characterised and abutment ready SiLago blocks are used to create more complex designs for algorithms, applications, and systems. This is automated by the higher abstraction synthesis tools. In this phase, the inter-SiLago block timing paths are analysed. The RCT timing model plays a central role in this higher abstraction timing analysis by predicting the skew between the RCT entry points into each SiLago block in a region instance. This skew is the only unknown when composing a design in terms of the SiLago blocks. An algorithm, presented in the next sub-section 2.4, is used to minimise the skew. Knowing a) the clock period, b) the delay over the inter-SiLago regional NoC timing paths from its pre-characterised fragments like a_r and br and delays in the flops, and c) the skew predicted by the RCT timing model, it is possible to validate the inter-SiLago timing paths as timing clean. The RCT model can predict the skew as accurately as the static timing analysis. This claim is validated in sec. 3.4. The end-user never uses the RCT timing model explicitly. However, it is used by the higher abstraction synthesis tools to a) decide the taps in the delay lines, b) validate the inter-SiLago timing paths, and c) the largest feasible size of region instance. The RCT timing components' characterisation and the inter-SiLago timing paths are like the delay properties of standard cells and wires in a specific technology. The end-user does not explicitly use these properties, but they empower logic and physical synthesis tools.

2.4 Minimisation of the skew

Once a region instance is composed in terms of SiLago blocks by the higher abstraction synthesis tools, the next step is to select the tap delay in each SiLago block. The problem of tap selection is formulated as an optimisation problem: what is the assignment of the tap index in each node x, that would minimise the absolute difference among $T_{LCT\_x,i}$. Node IDs are $1 \ldots N$, and tap indices are $1 \ldots M$. The first tap, $i=1$, imposes the minimum delay, and the last tap, $i=M$, the maximum delay. The ID of the node where the RCT enters a region instance is by convention 1 and has $T_{prop\_1}=0$. The node ID N is reserved for the furthest node, i.e., $T_{prop\_N}= max\ (T_{prop\_x})$. The tap index in node N is fixed to t_{tap_1} to have minimal latency, $T_{LCT\_N,1}$. The aim is to minimise the insertion delay to the blocks and the number of buffers used in each delay line. The cost function, L in EQ. 3, quantifies the mean of the absolute differences between $T_{LCT\_x,i}$ and $T_{LCT\_N,1}$, where $x=1 \ldots N-1$.

$$L = \frac{1}{N-1} \sum_{x=1}^{x=N-1} \left| T_{LCT\_x,i} - T_{LCT\_N,1} \right| \qquad \text{EQ. 3}$$

L's minimality can only be guaranteed if each term is minimal since L is a sum of absolutes. The minimality can be guaranteed by visiting each of the $1 \ldots N-1$ nodes and sweeping through the M taps to find the index that gives the minimal absolute difference, with respect to the reference node N, annotated I in EQ. 4. To conclude, L is minimal if we replace $T_{LCT\_x,i}$ with $T_{LCT\_x,k}$ where $k=I(x)$. The complexity of this algorithm is $O(N \cdot M)$, as is evident from EQ. 3.

$$\begin{aligned} I = \{ i \mid & 1 \le i \le M \qquad AND \\ & 1 \le x \le N-1 \qquad AND \\ T_{LCT\_x,i} - T_{LCT\_N,1} &= \min_{1 \le j \le M, j \ne i} \left(T_{LCT_x,j} - T_{LCT_N,1} \right) \} \end{aligned} \qquad \text{EQ. 4}$$

2.5 Maximum feasible size of region-instances

The RCT model introduced in section 2.3 is also used to decide the region instance's maximum size. The number of taps and their ability to compensate for the monotonous increase in T_{nat_x} dictates the maximum size of region instances allowed. This is formalised in EQ. 5. The value of x that fulfils this inequality decides the furthest node, N, from the RCT entry point and thereby the maximum feasible dimension of the region instance.

$$max(t_{tap\_i}) - min(t_{tap\_i}) \le max(T_{nat\_x}) \qquad \text{EQ. 5}$$

2.6 On-Chip Variations

On-chip variations **(OCV)** is a well-known challenge that impacts timing. There are three main approaches in use today for the standard cell based design flows [3], [4]: a) single deration per PVT (Process, Voltage, and Temperature) point, b) advanced OCV, and c) parametric OCV. All these methods rely on standard cell

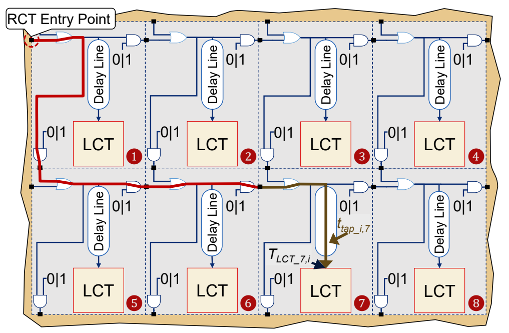


Figure 4. RCT Delay Model components in an example region instance.

libraries to define how much variation can be expected during manufacturing. The EDA tools take the variations into account during the STA to ensure that each timing path is designed and validated with sufficient margins to ensure no violations can occur in any process or operating point corner.

The synchoros VLSI methodology implicitly factors in OCV. The incorporation happens when the SiLago blocks are designed, analysed, and characterised using the OCV aware standard cell based design flows. The SiLago blocks are characterised at all delay corners to a) ensure that all its local intra-SiLago timing paths are timing clean in these corners and b) extract timing parameters for the fragments of wires that abut to create inter-SiLago wires; when a SiLago based design is assembled. These timing parameters are bounded by given OCV ranges to ensure their validity in the various corners. As a result, the OCV ruggedised parameters are used for the RCT timing models and the inter-SiLago NoC fragments when doing the inter-SiLago timing path analysis to ensure that all the timing paths are analysed to the same standard as the conventional EDA flows, albeit at a higher abstraction.

3 Experiments and results

This section presents the experimental results to validate the claim that the RCT generated by abutment is: a) valid, i.e., it is guaranteed to be timing and design rule check **(DRC)** clean, and b) predictable, i.e., properties of the generated RCT are known with post layout accuracy without having to do STA at the physical level, as it is done in the standard cells design flow.

Three experiments were performed to validate these claims. The first experiment reports the RCT delay model results and its properties, as discussed in sec. 2.3. The second experiment uses the RCT delay model to predict the properties of the RCT. The predicted values are validated against the values analysed by commercial EDA tools. This experiment's side effect is that RCT created by abutment is validated to be timing and DRC clean by the commercial EDA tools. We also demonstrate that the predictability is scalable. The third experiment benchmarks the RCT properties generated by abutment against a functionally equivalent RCT generated by the EDA tools. The results are used to prove that the synchoricity and abutment have negligible overheads in terms of typical cost metrics for clock tree: area, wire length, switching capacitance, skew, and average trunk slew.

3.1 Experimental setup

All experiments have been done in a 40 nm technology node, and the results have been validated using commercial EDA tools. These tools have been used for: a) to implement synchoros SiLago blocks, including the RCT fragments. b) validating the claims that the RCT generated by abutment are predictable, and timing and DRC clean and c) demonstrate that the benefits of synchoricity and abutment do not degrade the quality of RCT. EDA tools for points *b* and *c* above are solely used for research purposes to validate the claims and are not part of the regular synchoros design flow.

The proposed RCT by abutment scheme and the state-of-the-art hierarchical EDA flow is applied to the same experimental design. The design is a composite region instance of *two* different types of SiLago blocks. The one called Dynamically Reconfigurable Resource Array **(DRRA)** [5], and a second fabric called Distributed Memory Architecture **(DiMArch)** [6], Figure 5.

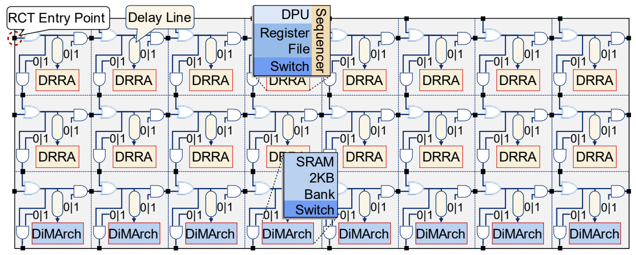


Figure 5. The experimental region-instance.

The region instance has 24 SiLago blocks that correspond to ~1.5 million NAND gates and 16 kBs of SRAM or 4 mm^2 in 40 nm node. The design's size is compatible with the expected typical size of synchronous region instance, for which the RCT is generated by abutment. To establish the method's scalability, we also tested a ~4 mil. gate design and show that the predictability is unaffected.

3.2 RCT Model

An RCT fragment with 32 taps in the delay line was incorporated into the DRRA and DiMArch SiLago blocks. These blocks were hardened to be synchoros and abutment ready. The RCT model parameters were extracted using static timing analysis **(STA)** from the post layout data and factoring in PVT and OCV. We remind this STA is a one-time engineering effort to characterise SiLago blocks and not seen by the end-user. The SiLago blocks on the edges have slightly different values of T_{RCT_chord} compared to the ones in the middle. These differences exist because there are minor differences in the interconnect and the layout of them. The 32 taps in the delay line can insert delays from 1.07 ns to 6.2 ns, with a step size of 0.16ns. The maximum size of the region that can be supported is 9 columns. A delay line with more taps or a larger step size can support larger regions. The RCT timing model is based on the extraction of timing using EDA tools that factor in PVT variations.

3.3 Predictability, Validity, and Scalability

Here, we describe the experiments that we did to establish that the RCT created by abutment is predictable and valid. The RCT property that we predict is the arrival times of RCT at each LCT entry point, the $T_{LCT\ x,i}$. We measure this property using two methods. The first method uses the RCT delay model and the higher abstraction timing analysis to analyse the inter-SiLago timing paths in terms of pre-designed and pre-characterised wire fragments that are absorbed in the SiLago blocks. This higher abstraction timing analysis for RCT is expressed as EQ. 2. The second method is to use EDA tools to measure the same delay. The results of these two methods are compared to establish the accuracy of the RCT model's prediction with EDA tools-based measurements as the benchmark. This experiment's side-effect is that the RCT created by abutment gets certified as being timing and DRC clean by the commercial EDA's analysis tools.

The worst-case skew, the difference between two LCT entry points, is 129 ps. This difference is small enough to be easily absorbed by the slack margin with which the SiLago blocks are synthesised. We remind here that the SiLago blocks do not communicate over ad-hoc wires synthesised for every design instance and iteration, as is the case with standard cells-based designs. The inter-SiLago communication in a region instance happens over regional NoCs. The RCT delay model's predicted values are almost identical to the one analysed by the EDA tools. The worst-case error compared to the EDA tools is 1.5 ps, and the RMS is 0.0005ps. This difference comes from the fact that our experimental setup does not have an infinite ground plane. This results in different design parts experiencing different coupling with long signals, like the reset and the ground plane. The result proves that the RCT delay model is sufficiently accurate that the higher abstraction synthesis tools do not need to do static timing analysis at the physical design level. To establish the RCT's scalability with increasing complexity, we experiment with a larger design of 20 columns. The predictability of the larger design was as good as the one for the smaller design.

3.4 SiLago RCT compared to EDA RCT

Here we present results that show RCT generated by abutment has comparable cost metrics to an ad-hoc RCT generated by EDA tools. To generate a functionally equivalent RCT by commercial EDS tools, we harden each DRRA cell as a leaf node. This hardening phase includes the synthesis of the local clock tree. The next step is to floorplan the design in terms of leaf nodes and synthesise the ad-hoc wires to connect and clock these leaf nodes. The clock that is synthesised in this step to drive the local clock trees is functionally equivalent to the RCT generated by abutment.

The commercial EDA tools then analysed both designs to compare the two functionally equivalent RCTs' properties. The results are shown in Table 1. As can be seen, the values of critical parameters are comparable. The standard cell area refers to standard cells dedicated to the RCT components shown in Figure 3, including MUX/AND/OR gates. Do note that the RCT compared to the SiLago block's LCT capacitance takes up ~2.5% of the total. The EDA generated RCT introduces a capacitance overhead of 0.4%. The overhead as a percentage of the total clock distribution (RCT + Local Clock Tree-**LCT**) is ~2%.

The SiLago RCT achieves a comparable arrival time of RCT at LCT entry points compared to the commercial EDA RCT. The SiLago RCT has a slightly better slew rate at LCT entry. The average and absolute difference in slew at different points in trunks for the entire (RCT+LCT) clock tree is also comparable and within the limits of the technology rules. The above experiment and the reported results establish that RCT generated by abutment has comparable quality as the one generated by the commercial EDA tools. The difference is that the EDA tool generated RCT is ad-hoc and synthesised anew for each design instance and iteration. Notice that the EDA RCT's irregularity in Figure 6 resulting from the attempt to factor and reuse the buffers. In contrast, the synchoros RCT is regular. The ad-hoc nature of the EDA RCT and its irregularity violates the requirements that the synchoros design style places on RCT, as discussed in section 2.

The synchoros RCT by abutment is regular, as shown in Figure 6b. It has three main branches corresponding to two DRRA and one DiMArch rows. Each branch has eight leaf nodes, and each leaf node has the same RCT structure. The regularity of the SiLago RCT enables abutment. Its stable structure, together with its absorption in the SiLago blocks as a pre-synthesised and characterised structure, enables predictability. The proposed method's main drawback is the clock latency, which is higher than the EDA RCT, as shown in Figure 6. The increased variation implied by the higher latency is factored in the PVT variations. The higher latency is defended by its potential to improve design productivity by enabling a predictable RCT and a composition by abutment scheme. We observe that when we transitioned from full-custom to standard-cell based designs, we accepted significant overheads in the interest of improved design productivity.

TABLE 1
EDA RCT VS SILAGO RCT

	SiLago RCT	EDA RCT	% Change
Wire Length [µm]	422932.1	415263.9	1.8 %
Standard Cell Area [µm^2]	19472.6	18276.5	6.5 %
Average Trunk Slew [ns]	0.062	0.090	31.1 %
Total Capacitance [pf]	129.9	127.2	2.1 %
Avg. Diff. in Arrival Time (LCT Entry Point) [ns]	0.04299	0.0337	27.6%

3.5 Analysis of Experimental Results

The experiments and their results presented in the previous sub-sections prove the following: i) the feasibility of generating RCT by abutment, ii) the generated RCT is a valid VLSI design, iii) the generated RCT is predictable, iv) the composition by abutment is scalable with complexity and v) the RCT generated by abutment has negligible overhead when compared to functionally equivalent RCT generated by the state-of-the-art design flow.

The key benefit of the composition of RCT by abutment is that it contributes to raising the physical design level to RTL; for both logic and wires. This makes the VLSI design space exponentially smaller, composable and predictable to enable automated synthesis from higher abstractions, as argued in [1].

4 State of the Art

The CTS (Clock Tree Synthesis) has been researched since the earliest days of VLSI. The CTS research's main objective has been to optimise the clock tree's cost metrics, i.e., smaller switching capacitance, minimising skew, maintaining an edge, etc.[7], [8]. These approaches are based on sophisticated heuristic algorithms.

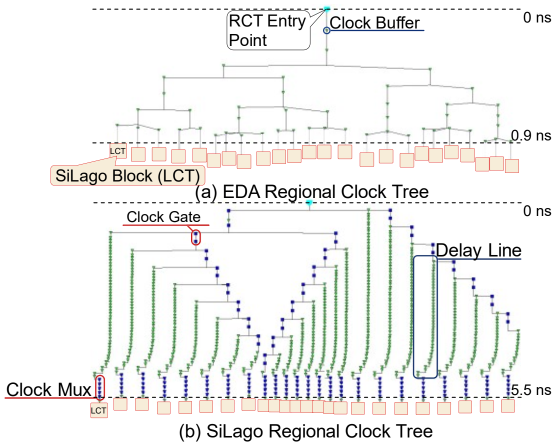


Figure 6. EDA generated RCT vs SiLago RCT

Most of these methods are based on the van Ginneken dynamic programming algorithm for buffer insertion and sizing, and the delay model used is the Elmore delay model [9]. An excellent survey of CTS techniques is presented in [10]. In [7], [11] and [12] the authors use a post-clock-tree-synthesis optimization. The proposed methods optimise the clock tree by altering an already synthesised tree. In contrast, our goal is to balance the clock's arrival time at the local clock trees' source points.

The key difference between the research proposed in this paper and the ad hoc CTS research reviewed above is that in synchoros VLSI design style, composing a larger design in terms of SiLago blocks does not involve creating new wires of any type, including the clock. Whereas in the state-of-the-art, clock wires and buffers are synthesised as part of physical synthesis as a post logic synthesis step. In many other respects, the RCT generation method proposed in this paper does not compete directly with the research reviewed above; they are complementary. What we propose is an alternative to the clock tree generation at a higher level, which has relatively insignificant capacitance but profoundly affects the engineering cost and predictability.

CTS research has also focussed on regular topologies of clock trees like H-tree and Mesh clock structures [9], [13]. The regularity of these structures makes them seem a good match for the synchoros VLSI design style. These structures do not fulfil the requirements of creating RCT by abutment. H-Tree is a hierarchical structure, and its depth would depend on the size of the region instance. For this reason, it is absorbing an H-Tree as RCT fragments and being able to create an arbitrary H-Tree by abutment that is dependent on the size and shape of the region instance is not feasible. Mesh-based clock tree poses a challenge in terms of the predictability requirement. Since the clock tree mesh creates an equipotential surface, it creates cyclic graphs that are impossible to analyse with STA [9], [13]. This is an even bigger problem for the RCT by abutment scheme because it requires the iron-clad timing models to predict the RCT properties with sufficient accuracy. The standardised entry points and propagation path of the SiLago RCT make it possible to have a trustworthy timing model.

Like synchoros VLSI designs, FPGAs are regular structures and naturally invite comparison of their clock tree schemes. The fundamental difference is that the clock tree routing is pre-designed for each FPGA. The FPGA wires and LUTs can be configured differently, but the silicon will not change. In contrast, depending on functionality and constraints, the higher abstraction syntheses will create region instances of different size, aspect ratio, and entry points. This will result in different RCT dimensions and topology. All the wires in synchoros VLSI design are regular and structured but can be connected and organised in various ways to create endless variations of structures.

Further, FPGAs do require an STA as a post-synthesis step. In synchoros VLSI design, the regions are pre-characterised to work up to a clock speed, as long as the RCT infrastructure can deliver the clock to the LCT entry points inside a known skew margin. This is why the synchoros VLSI design requires no physical level STA. The region instances of arbitrary size and domain-specific functionality are guaranteed to be correct-by-construction.

Some methods have been proposed for post-fabrication clock deskewing [14], [15]. These methods use configurable buffers and delay lines to correct the skew after fabrication and are still designed ad-hoc for each design. Our method's innovation is the structured design of the RCT that makes it predictable and not the use of the delay line. The structured and predictable design enables higher-level synthesis, as argued in [1]

5 Conclusion and Future Work

We have presented a regional clock tree generation method based on abutting fragments of RCT that are absorbed SiLago blocks as a one-time effort. We are working on several enhancements to the RCT method and the overall synchoros VLSI design framework. The programmable delay line is constructed from qualified standard cells, and its atomic delay decides the resolution to which we can minimise the skew. We are working on a more advanced programmable delay line that will allow a finer adjustment of delay that would be needed for higher frequencies. In this improved delay line, the number of taps would increase logarithmically with the delay. Such a delay line will be constructed with weighted positional taps much like a fixed-point number. We are also in the process of making the delay of each tap in the delay line adaptive [16], [17] to make the design more robust and enable dynamic voltage frequency scaling.

REFERENCES

[1] A. Hemani *et al.*, "Synchoricity and NOCs could make Billion Gate Custom Hardware Centric SOCs Affordable," in *IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, 2017, pp. 1–10.

[2] M. A. Shami *et al.*, "Partially reconfigurable interconnection network for dynamically reprogrammable resource array," in *International Conference on ASIC*, Oct. 2009, pp. 122–125.

[3] A. B. Chong, "ASIC design margin methodology," in *IEEE 2013 Tencon - Spring*, Apr. 2013, pp. 165–173.

[4] A. Elzeftawi *et al.*, "Addressing Process Variation and Reducing Timing Pessimism at 16nm and Below," 2016.

[5] M. A. Shami, "Dynamically Reconfigurable Resource Array," KTH - ICT, 2012.

[6] M. A. Tajammul *et al.*, "TransMem: A memory architecture to support dynamic remapping and parallelism in low power high performance CGRAs," in *PATMOS*, 2016, pp. 92–99.

[7] C. Deng *et al.*, "Fast synthesis of low power clock trees based on register clustering," in *International Symposium on Quality Electronic Design*, Mar. 2015.

[8] P. J. Restle *et al.*, "A clock distribution network for microprocessors," *IEEE J. Solid-State Circuits*, vol. 36, no. 5, pp. 792–799, May 2001.

[9] L. Lavagno *et al.*, *Electronic design automation for IC implementation, circuit design, and process technology: circuit design, and process technology*, 2nd ed. Boca Raton, FL: CRC Press USA, 2016.

[10] A. Rajaram *et al.*, "Robust chip-level clock tree synthesis," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 30, no. 6, pp. 877–890, 2011.

[11] T. J. Wang *et al.*, "Top-level activity-driven clock tree synthesis with clock skew variation considered," *IEEE Int. Symp. Circuits Syst.*, 2016.

[12] J. Lu *et al.*, "Post-CTS clock skew scheduling with limited delay buffering," *Midwest Symp. Circuits Syst.*, pp. 224–227, 2009.

[13] A. Abdelhadi *et al.*, "Timing-driven variation-aware nonuniform clock mesh synthesis," *ACM Gt. Lakes Symp. VLSI*, pp. 15–20, 2010.

[14] A. Chattopadhyay *et al.*, "Reconfigurable Clock Distribution Circuitry," in *International Symposium on Circuits and Systems*, May 2007, pp. 877–880.

[15] Yuko Hashizume *et al.*, "A novel clock deskew method by linear programming," in *Midwest Symposium on Circuits and Systems*, Aug. 2007.

[16] C. C. Kao, "Clock Skew Minimisation in Multiple Dynamic Supply Voltage with Adjustable Delay Buffers Restriction," *J. Signal Process. Syst.*, vol. 79, no. 1, 2014.

[17] K. Park *et al.*, "Mixed allocation of adjustable delay buffers combined with buffer sizing in clock tree synthesis of multiple power mode designs," in *DATE*, 2014.