A reconfigurable distributed architecture for clock generation in large many-core SoC

Chuan Shan, Dimitri Galayko, François Anceau, Eldar Zianbetov
Laboratoire d'informatique de Paris 6 (LIP6)
University of Pierre and Marie Curie (UPMC), Paris, France
Email: {First name.Last name}@lip6.fr

***Abstract*—This paper focuses on clock generation and distribution in large SoC. After a brief analysis of diverse existed approaches, we propose a distributed architecture based on coupled local clock generators. Three prototypes are presented to demonstrate the feasibility of a large globally synchronous SoC with high reliability by using this approach. Moreover, the reconfigurability feature of this architecture provides a platform for exploring topologies with potentially improved performance.**

I. Introduction

Advances in CMOS technology have led to an exponential increase of chip complexity and transistor numbers (Moore's law). The modern SoC can be regarded as micro-networks allowing different parts of the systems to communicate. Synchronization of these communications becomes a research subject of utmost importance. This study addresses the problem of global clock generation inside complex and large SoC, so to allow a fully-synchronous communication on the chip. Although asynchronous communication techniques in the SOCs have recently gained the ground, fully synchronous operation of the digital system is desirable in many cases, espetially where the reliability of the system is the first priority.

The main problem for implementation of fully sychronous SOCs is a generation of a global clock. Our study is motivated by deficiencies of conventional clock generation techniques. Traditional clock generation in complex circuits uses tree (Fig. 1(a)) or grid structures [1] [2]. The matching between different clock paths is the key point of the clock network design. In large chips designed in advanced CMOS technology, such a global matching is difficult to achieve. Because even if the positions of the receivers are symmetric with regard to the source point of the clock, mismatches between the buffers and the lines introduce uncontrollable skew. In practice, the local clock areas have different sizes and positions, making difficult a perfect clock tree equilibrium. To solve the problem of mismatch, the size of the lines and buffers must be increased so to become less sensitive to fabrication errors, which however makes the solution very expensive, mainly in terms of energy consumption and area.

Possible improvements of the original tree distribution system consist in providing the clock generator with a skew compensation mechanism. One approach using this technique is described in [3]. This microprocessor uses a clock distribution network comprised of two trunks. Each trunk is governed by its own clock tree. The roots of clock trees are

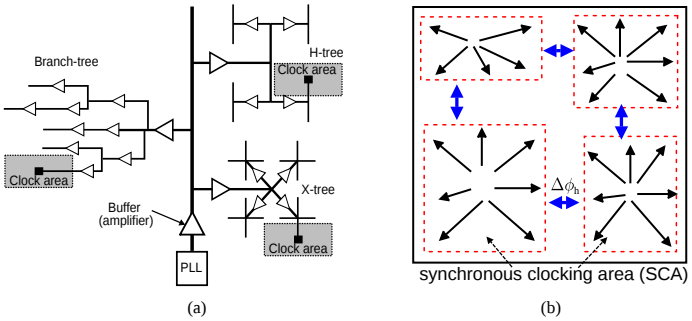

Fig. 1. Different approaches of clock distribution: (a) centralized clock generation (clock tree); (b) partitioned clocking area

connected with a source clock signal by variable delay lines (VDL). It is possible to precisely adjust the clock in each trunk through its VDL. A phase comparator (PC) located between the two trunks compares the arrival time of the clock ticks. In case of imbalance of clock ticks between two trunks, the delay of the distribution chain is adjusted. This approach is particularly effective for compensation of skew associated with interconnections, and process, supply voltage, temperature (PVT) variations. Despite these advantages, this strategy requires global distribution of the high frequency clock, which should be buffered. Hence it requires a lot of power to keep the amplitude of the clock on appropriate level.

To simplify the problem of the global clock generation, large digital chips are often partitioned into local clock areas, as shown in Fig. 1(b), [4] [5]. These areas are also known as isochronous zones [4] or synchronous clocking areas (SCA) [13]. Each zone has its own local clock: The zones are small enough so that the clock distribution inside the zones can easily be achieved by conventional techniques. The problem of global communication is reduced to communications through the borders of the zones.

If all local clocks are synchronized in phase and in frequency, the communications are synchronous (Globally Synchronous systems). The advantage of such an architecture is that the communication between two adjacent areas obey the same rules as it is inside a single clocking area, and the design of whole chip can use traditional and well established digital design techniques.

However, because of the difficulties of global synchronization, SoC engineers orient their choice toward the ar-

chitectures where the communication between SCAs is asynchronous (Globally Asynchronous Locally Synchronous systems, GALS). This is achieved with use of bisynchronous interfaces guaranteeing the signal integrity, at price of design complexity and increased latency. Moreover, in asynchronous circuits, the reliability is difficult to guarantee at design stage for two reasons. First, an asynchronous system is analog, since the time is continuous, and its exaustive verification is impossible. Second, the metastability risk may be worse than usually expected, because the clocks of SCAs are not really fully independent. Indeed, these clocks are derived from a single input clock and distributed to different modules. Their relative phase shifts depend upon fast and slow changing parameters like aging (slow), temperature (medium speed), and voltage (fast). Metastability could occur for certain values of these parameters. During a short period of time, slow parameters do not change, leading the metastability to reoccur repetitively, so increasing the risk of the system failure. For this reason, synchronous circuits may be desirable in applications with high reliability requirement.

Our research focuses on looking for an approach to synchronize the whole chip in an effective and less expensive way compared with conventional centralized clock distribution. We propose a solution for dynamic and distributed synchronization of local clocks on large synchronous circuit [6] [8]. For that, each area has its local clock generator coupled in phase with the clocks of its neighboring areas. Compared with a centralized clock generator, the distributed clock generator requires shortest paths to carry the synchronization signals, reducing the accumulation of delays and jitters on the distribution lines, and provides a scalable solution, as our study will show.

An innovation of our solution compared with [6] is that the clock generator is fully digital, which makes the methodology compatible with functional blocks in SoC and allows using the digital design flow and EDA tools.

This paper is organized as follows: In Section II, we start by introducing the general architecture and features of the proposed distributed clock generator. In Section III, we present this technique in detail through three prototypes. Section IV presents the use of reconfigurability of the network so to improve the performances of the clock generation.

II. Architecture of Distributed Clock Generator

A. System description

The structure of the clocking network is presented in Fig. 2. The local clocks are generated by digitally controlled oscillators (DCOs). Digital phase-frequency detectors (PFD) measure the timing error between each couple of neighboring DCOs. The network is coupled with the external reference clock through a PFD placed in upper left corner of the network. The digital error signals from PFDs are processed by the digital proportional-integral (PI) loop filters. Each filter processes the weighted sum of the errors from up to 4 PFDs (depends on topological position of the node) and generates the digital control code for the DCO. The control objective of the filter is to maintain the sum of errors close to zero. In such a network

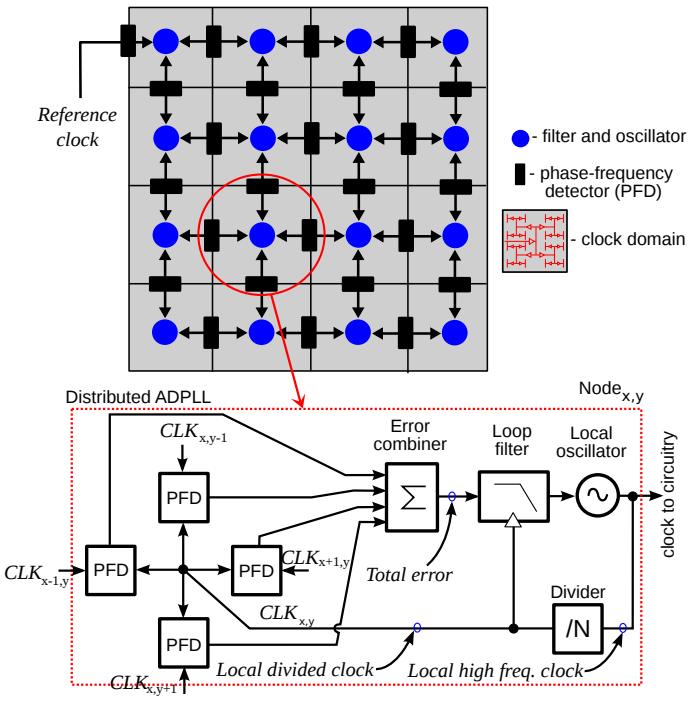

Fig. 2. Architecture of the ADPLL network and of a single node

all oscillators synchronize with the phase of the reference clock, if the system is designed properly.

The PFD is an analog-to-digital converter quantifying the synchronization error into a digital signed number. Fig. 3(b) is the transfer function of a 5-bit PFD, its dynamic range is limited by the boundaries $\pm\Delta\phi_r$ defined by constraints of precision and hardware complexity. The detail block diagram of the PFD is shown in Fig. 3(a). The PFD consists of a bang-bang phase detector (BB) measuring the sign of the phase error, a time-to-digital converter (TDC) for quantification of the absolute time error between two clocks and an arithmetic block combining the outputs of these blocks and producing two binary signed signals (straight and inverted) thereafter used by the local and neighboring nodes.

The implemented DCOs are the ring CMOS oscillators employing width-modulated technique for the digital frequency tuning [8] [9] (Fig. 4). Their structure is based on a 7-stage ring oscillator with a parallel connection of tri-state tuning inverters in each stage of oscillator. The inverters are controlled by thermometer codes obtained from the binary to thermometer decoders. The monotonicity of the code-frequency characteristic is guaranteed by an appropriate choice of the control algorithm. The characteristics of designed in 65 nm CMOS technology oscillator are presented in Table I of Sec. III.A.

The error processing unit is composed of an error combining block and a loop filter. The error combiner receives up to four 2's-complement coded errors. They are passed through four variable gain blocks and then summed using four-input adder. The weighting coefficients of the variable gain $Kw_1 - Kw_4$ are programmable. Each gain can take independently a value

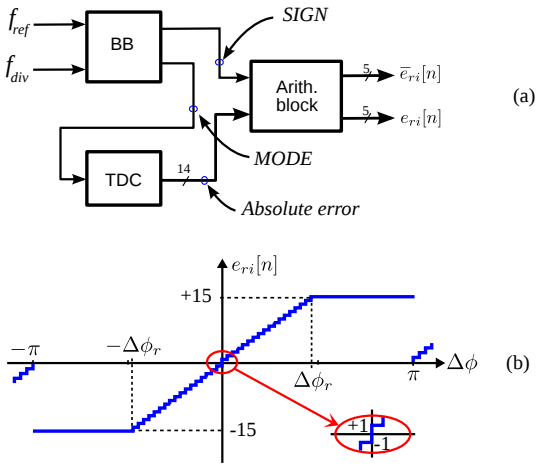

Fig. 3. PFD: (a) block diagram and (b) its transfer function.

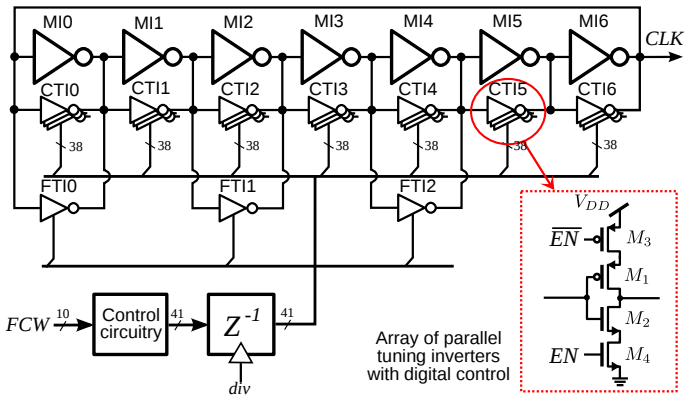

Fig. 4. Architecture of the used DCO.

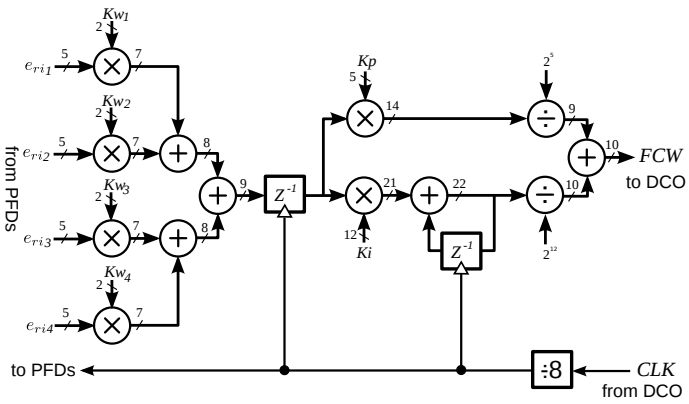

Fig. 5. The error combiner and proportional-integral filter.

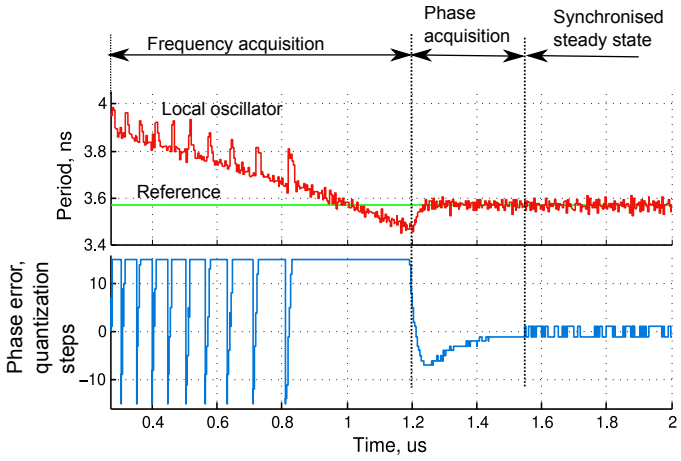

Fig. 6. Typical transient process of a single ADPLL.

from the set {0,1,2,4} and implemented as a binary shift, so introducing the smallest delay. By programming these coefficients, we can control the connectivity between the nodes of the network. It should be mentioned that each node is an auto-sampled system: the filter is sampled with the locally generated clock (divided), whose phase is defined by the output of the filter. The loop filter is implemented as a proportional integral (PI) filter processing the sum of errors. Its coefficients K_p and K_i are programmable.

B. Fundamental properties of the system

1) Stability issues: The paramount question about the stability of a network of coupled oscillators has been addressed in many theoretical studies. It should be noted that our system is mixed time-discrete/time continuous time, is self-sampled, and the levels are quantized. These properties together with high dimension of the system make an exact analysis extremely difficult. Two studies partially addressed the stability of the system, under simplifying hypotheses of absence of the level quantization [11] [12]. The domain of validity of the studies are explained in Fig. 6, which presents a typical transient process for a PLL: the frequency acquisition, the phase acquisition and the synchronized operation. The studies [11] [12] proved formally the stability of the system in the two first modes, and proposed an algorithm of choice of the loop filter parameters guaranteeing the stability and the required performances of the transient process. However, in the synchronized mode, there is a residual synchronization error due to the quantization, and the models used in above mentioned studies are not sufficient to its magnitude. It was empirically shown and experimentally confirmed that this error can be reduced to a level under 3 quantization steps, but a formal demonstration is still lacking.

2) Multiplicity of synchronization modes: The work of Pratt and Nguyen in [6] highlighted a fundamental problem specifically related to the PLL networks. In addition to the main synchronization mode in which all oscillators have the same phase, a PLL network may have several modes in which the local oscillators are synchronized in frequency, but with fixed non-zero phase errors between the oscillators. The multiplicity of synchronization modes is called *mode-locking* in the literature. The multimodality is typical for non-linear systems of high dimension. In the PLL networks it is caused by the cyclic (modular) nature of the phase: the transfer function of the phase comparator is defined modulo 2π. Obviously, for distributed clock generation, only the mode in which the phase errors are zero is suitable. The actual mode established

in the network depends on the initial conditions, which are not controllable *a priori*.

The mode-lock phenomenon is illustrated on the example of a 4 nodes network Fig. 7, in which the error processing block receives the sum of the errors with two neighbors. As it was explained in [6], the error processing block operates so to keep the *Total error* (on fig. 2) equal to zero. The *Total error* can be zero if :

1) the phases ϕ_1, ϕ_2, ϕ_3 and ϕ_4 of the oscillators are equal

2) the phases have the values given in Fig. 7.

In the first case, all phase errors of the network are obviously zero, their sum is zero as well. For the second case, the phase errors are non-zero, but the *Total error* value is zero. In this way, the sum for all nodes of this network is zero despite the unequal phases of all oscillators. It can be proven that such a state is stable; once acquiring this operation mode, the network remains in it, since the control objective (zeroing the *Total error* quantity) is fulfilled. For larger networks, many undesirable states may exist.

Two questions are associated with this phenomenon: (i) which is the probability of undesired mode-lock given random initial condition, (2) which is the technique allowing an active selection of the desirable mode. Concerning the first question, our study highlighted that the spontaneous (intrinsic) selection of the mode by the network highly depends on the network model. In simulation of a large network with a VHDL model (16 nodes and more), the network nearly always converges to an undesirable mode. However, the same network implemented on a chip (cf. the next section) always converges spontaneously to the desirable mode. A deeper theoretical investigation of these phenomena is required, addressing the noise issue and the topological structure of the basin of attraction of different synchronization modes.

These considerations makes the second question of paramount importance. Several studies were done previously on the problem of active selection of the desired synchronization mode [6]. These solutions were based on use of a phase detector with a particular characteristics. In section IV we will present solutions which makes use of digital nature and of reconfigurable properties of our system.

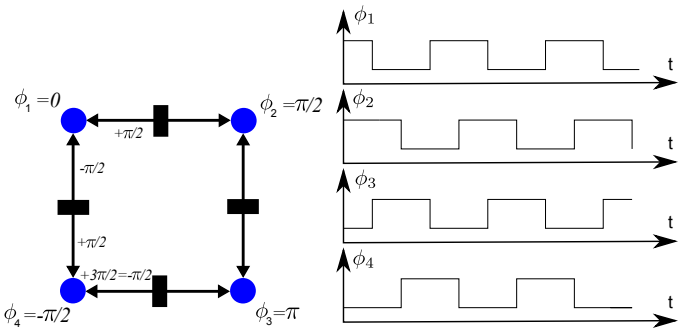

Fig. 7. Illustration of the mode-locking phenomenon in a 2×2 mesh network

III. Prototype implementation

Based on the principle presented in the previous section, several prototypes have been developed: an ASIC prototype of a 4×4 network (fabricated and tested), a 10×10 network FPGA prototype on Altera Cyclone II platform (implemented and tested), and an ASIC prototype of a 10×10 network (ongoing). These work validate experimentally the proposed clocking solution.

A. 4×4 network ASIC prototype design

A prototype of the distributed clock generator with 16 nodes has been designed and manufactured in 65 nm CMOS technology [14]. It has an area of ≈2 mm^2 where the clock network itself occupies 0.8×0.9 mm^2 (Fig. 8). Besides the clocking network, the chip includes design-for-test block and individual blocks of the network for their characterization. The microphotograph of the fabricated silicon prototype is presented in Fig. 8. The goal of this prototype is proof of feasibility of the network of all-digital PLLs for synchronization of several clock sources.

Fig. 9 presents the waveforms captured off-chip local clocks when the network is synchronized. The 1.6 GHz clock frequencies are divided by 16, to comply with the limits of the available pads. The observed timing errors between neighboring clocks were in the range of 30-60 ps. This is less than 10% of the clock period. These figures should be seen as pessimistic, because of the off-chip measurement. The residual synchronization error can be reduced by increasing the phase detector resolution. As predicted by theory, the error is a zero-centered random process, i.e. the average skew is zero.

Fig. 10 shows the transient process in the one of the oscillators. A perturbation was introduced in a network at t =8 μs, in order to study the robustness of the synchronization. After this perturbation, the clocking network returns to the reference frequency and phase after 17 μs. The frequency acquisition

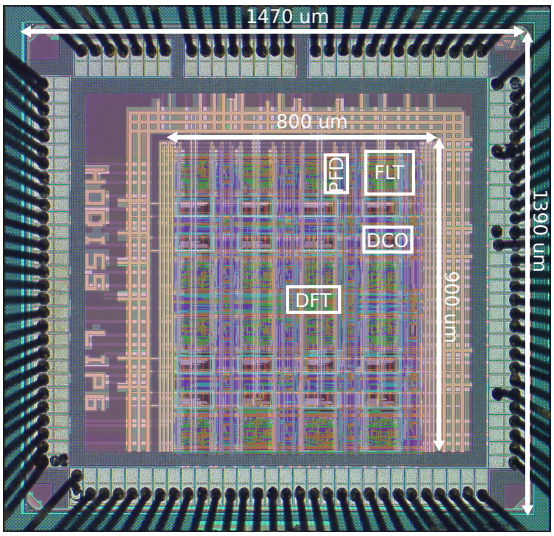

Fig. 8. Die microphotograph

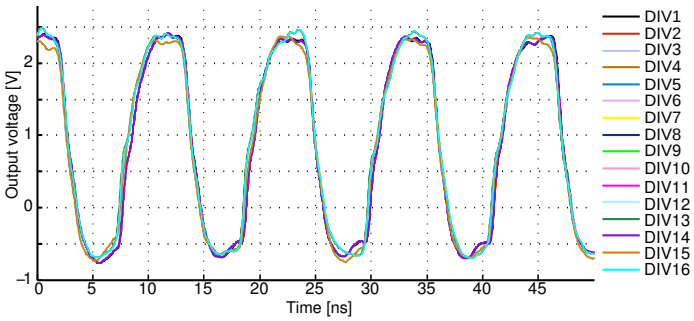

Fig. 9. Captured phase locked divided clocks

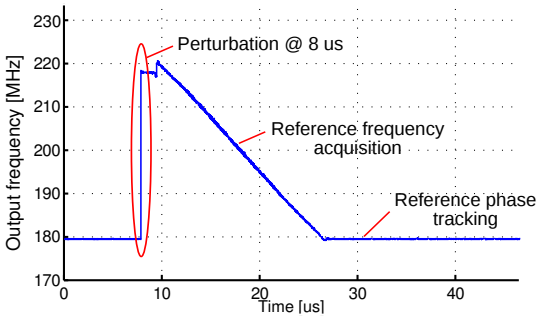

Fig. 10. Transitional process in Node 11

speed may be increased by employing special techniques more efficient than a simple PI correction.

Table I shows a performance summary of the measured results and comparison with existing implementation of the distributed clock generator.

TABLE I
NETWORK TEST CHIP MEASUREMENTS SUMMARY AND COMPARISON

Parameter	**[7]**	***This work***
Number of nodes	16	16
Central frequency of the SCA, MHz	1200	1744
Frequency range, MHz	1100∼1300	1100∼2380
Timing error, ps	30	$< 60^{*}$
Power consumption, mW	390**	186.2***
Technology, nm	350	65
Clocking core area, mm^2	-	≈0.72
Chip area, mm^2	≈9	≈2.04
Circuitry nature	analog	digital

* between neighbor nodes
** F_{clk} = 1.2 GHz
*** mixed and digital @ F_{clk} = 1.6 GHz

B. 10×10 network design

After a successful implementation of a small 4x4 oscillator network, the main question concerned the scalability of the solution: how the residual synchronization error increases with the size of the network. For that, we studied and designed a 10x10 network, containing 100 nodes. Priorly to an ASIC implementation of the network, we studied the architecture on an FPGA platform.

1) FPGA prototype of 10x10 network: The FPGA model represented truthfully the studied system on the FPGA model, with the following exception: (i) all time/frequency parameters of the network were scaled proportionally so to comply with the maximum frequency limit of the FPGA platform, (ii) the time of the FPGA prototype was discretized with a step corresponding to the frequency of the internal FPGA clock. For this reason, the analog/digital blocks of the ADPLL (the digitally controlled oscillator and the phase-frequency detector) have a particular implementation adapted to the FPGA limits [16].

DCO implementation. A DCO is a digital-to-analog converter converting a digital code into the oscillation frequency. Since analog functions cannot be implemented in FPGA (e.g., programmable delay), in this work the DCO is implemented as a pre-loaded N-bit counter whose overflow signal provides the output DCO clock signal. The counter uses an external clock with a high frequency (period=$T^{fpga}_{clk\ DCO}$). When the counter saturates, an output event is generated, and at the same time the counter is reloaded with the filter output code. Hence, the period of the clock generated by DCO (T^{fpga}_{DCO}) for a certain input code K is given by

$$T^{fpga}_{DCO} = T^{fpga}_{clk\ DCO}(2^N - K) \tag{1}$$

where N is the number of bits of the counter.

PFD implementation. The bang-bang detector was implemented with the same architecture as in ASIC (cf. section III-A). However, the TDC for FPGA prototyping is a digital chronometer counting the number of internal FPGA high frequency clock cycles during the interval to be measured. The interval length is specified by the MODE pulse duration (cf. fig. 3). The pulse is applied to a counter ENABLE input. The counter output increments till the end of the MODE pulse. The counter output values gives the measured interval in internal clock period units.

The only time parameter of the TDC is the time quantization step τ^{fpga}_{TDC}. In order to truthfully represent the dynamic of a network on the ASIC, this parameter must be related to the ASIC TDC quantization step τ^{asic}_{TDC} by the same scaling factor α as the DCO time/frequency parameters of both ASIC and FPGA systems:

$$\tau^{fpga}_{TDC}/\tau^{asic}_{TDC} = F^{asic}_{n}/F^{fpga}_{n}. \tag{2}$$

Table II summarize the parameters of the TDC and of the DCO implemented for the FPGA prototype of the ADPLL network.

This FPGA implementation of a 10x10 network allowed a study of the space distribution of the synchronization errors, in particular: errors between the neighboring nodes, and errors between distant nodes. For that we measured the phase difference between each local clock of the diagonal nodes and the reference clock (as shown in Fig. 11). Then we drew the curve of phase error RMS value in function of the distance between the node and the reference clock. It should be noted that the *distance* mentioned here is not the physical distance in microns, but the Manhattan distance the reference phase

TABLE II
PARAMETERS OF THE FPGA IMPLEMENTATION AND ASIC SPECIFICATION

Parameter	ASIC	FPGA
F_n	265.516 MHz*	77.93 kHz
ΔF_n	330 kHz*	97.05 Hz
F_{min}	232.2 MHz*	67.28 kHz
F_{max}	295.9 MHz*	92.73 kHz
τ_{TDC}	20 ps	68.065 ns
$F^{fpga}_{clk_{DCO}}$	–	62.5 MHz
$F^{fpga}_{clk_{TDC}}$	–	14.7 MHz

* for DIV clock

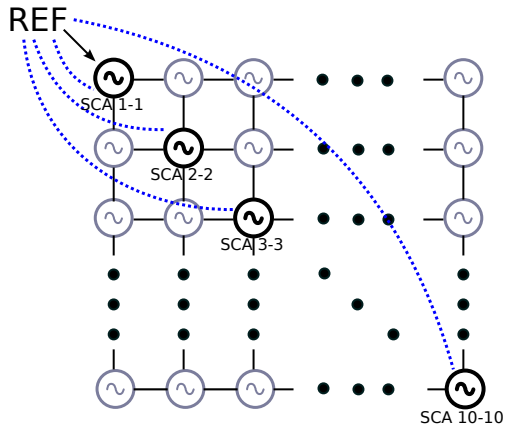

Fig. 11. Experiment principle diagram : bidirectional configuration, initial frequencies of nodes are different

information has to travel before arriving at a local node. For instance, the shortest distance in Fig. 11 is from the reference and SCA1-1, which equals to 1, and the longest one is to SCA10-10, which equals to 19.

Fig. 12 traces the maximum absolute value of the synchronisation error in function of the distance of the node from the reference, for two network topologies. We can observe that in unidirectional network (cf. section IV-A) the phase error is accumulated as the reference phase information travels further, just like in a conventional clock tree. While in bidirectional network, which is the configuration at which the network works at steady state, phase errors between all the nodes in the network and the reference are well constrained within ± 3 times PFD quantification steps. This proofs the scalability of the synchronization solution we propose.

2) 10×10 network ASIC prototype design: On the base of successful test of the FPGA prototype of a large network, a design and fabrication of this architecture in 65 nm technology is ongoing. Here we present the main issues related with this design work.

Since the network has a regular mesh structure and the local clock generator in different isochronous zone possesses the same components: DCO, PFD, filter, etc., the physical implementation of the network bases on an IP (Intelligence Property) reuse hierarchical strategy. A mixed signal IP block named NODE is designed as the basic node element of the

Fig. 12. Maximal value of phase error in function of distance to the reference clock

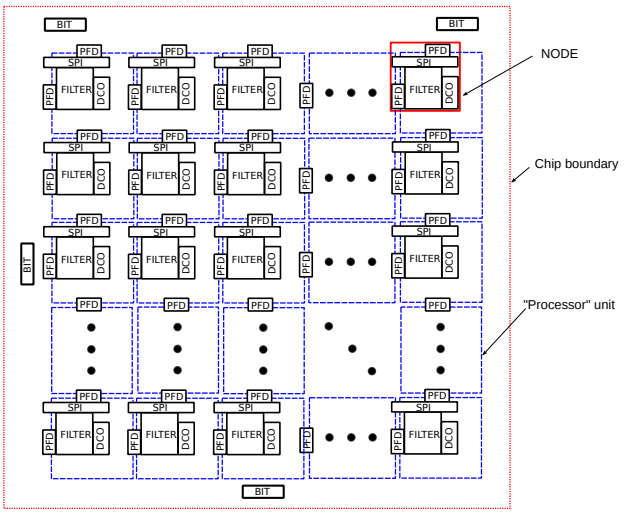

Fig. 13. Preliminary floorplan of the test chip

network. It contains the following blocks: serial programming interface (SPI), loop filter, DCO, and two PFDs. These two PFDs are shared with its neighbors when we combine several NODEs together. Since the mesh is two dimensional, two PFDs are enough: one for vertical neighbors, the other for horizontal ones. The chip is basically composed of NODE blocks in the core for the clock generation and synchronization, and four built-in test (BIT) circuits around the core for phase error characterization [15]. A preliminary floorplan of the implemented circuit is given in Fig. 13. The core area is partitioned into SCAs where the NODEs are placed.

The size of the chip is 2734×2756.8 μm^2 where clocking network core occupies 1805×1771 μm^2. In this particular case, since the chip size is I/O constrained, the free space is filled with fillers, decoupling capacitance. In real conditions, the remaining space inside the NODE and the routing channel would be used by the functional circuit.

IV. EXPLORATION OF TOPOLOGIES AND CONFIGURATIONS

The reconfigurability feature of proposed system consists in the following two aspects:

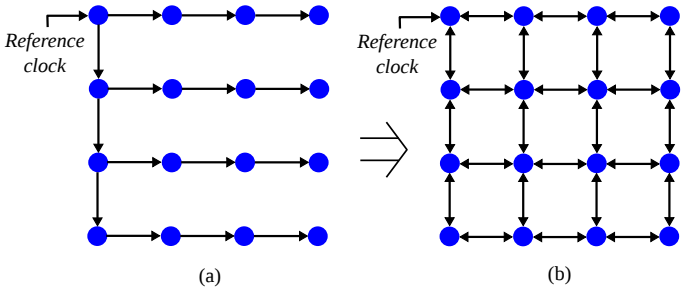

Fig. 14. Mode-lock elimination technique: (a) power up in unidirectional configuration, (b) bidirectional configuration activated after establishment of unidirectional network.

- Programmable filter coefficients. This allows choosing different loop correction strategies (ex. minimum settling time oriented or minimum phase error oriented).
- Programmable connectivity and weight of different links between PFDs and local loop filter. This allows configuring network into different topologies (ex. unidirectional mode, bidirectional mode, etc.).

Since the number of bits to be programmed is large (20 bits or more per node), a serial programming interface (SPI) is used for the programming sequence transmission [16]. The SPI protocol is basically implemented as a shift register with equal length in each NODE (as displayed in Fig. 13). These shift registers are cascaded serially in the network, which makes the programming strategy extensible.

Thanks to the reconfigurability of proposed architecture, we can solve several problems related to PLL networks: the undesirable mode selection and stabilisation of the network through particular configuration of the network edges, so to limit the propagation of perturbations/waves in the network.

A. Desirable mode selection

The proposed method is based on an on-fly dynamic reconfiguration of the network [13]. The reconfiguration procedure is performed during the start-up and consists of two steps:

Step 1. The clocking network is powered up and programmed into a unidirectional configuration. This is achieved by programming the feedback links between nodes; by disabling or enabling them. For example, each node receives the information about errors from upper or left neighbors Fig. 14(a). In such a mode information about reference frequency and phase propagates from the left upper node to the lowest right corner of the network. This mode excludes the cycles of propagation of information, hence eliminates the possibility of undesired locking. However, in such an operation mode the suppression of perturbations is weak (cf. fig. 12 and [13]) and clocking network has the accumulative errors. They increase with the distance from the reference point and introduce an undesired skew.

Step 2. Once the network is synchronized with certain timing errors, it is programmed into a bidirectional configuration Fig. 14(b). In this mode the reverse links are activated, and the network operates in a fully synchronous mode with distributed feedback (coupling) maintaining the synchronization.

Thus, when the required synchronization mode is selected, the timing errors between neighboring local clocks are close to zero.

As an example, Fig. 15 shows the four phase errors after weight coefficients (Kw_i) multiplication between one node in a network and its neighboring nodes during the entire experimental process on the FPGA prototype. During the time period 0ms - 74ms, the network is configured as unidirectional mode, only the phase difference between the node and its left neighbor is applied to the input of the filter. Then the reprogramming is finished within 50 μs. During this time, the network still works in unidirectional mode. Since 74.05 ms, the network switches to bidirectional mode, all the four errors are taken into consideration by the filter. We can observe that during the unidirectional mode the phase error may be as large as three TDC measurement steps. However, in the bidirectional mode the maximal error is reduced to ± 2 TDC steps: that corresponds to the desired synchronized operation of the network.

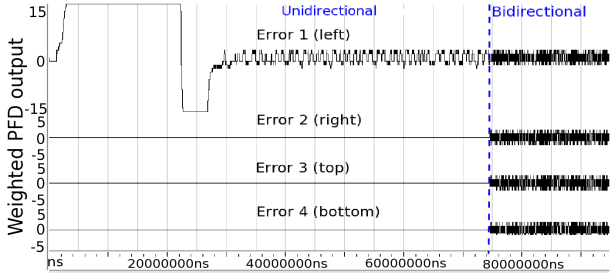

Fig. 15. Phase errors between one node and its neighbors (after K_{w_i} multiplication)

B. Control of the network edges: a "swimming-pool" model

Through a theoretical study [17] we find that the transient process of an unlimited ADPLL network in phase domain can be seen as analogous to the wave movement in a vast expanse of water. In equilibrium, the whole water surface is flat. Similarly, when the ADPLL network is synchronized, all the locally generated clocks are in phase. However, on the boundary of a limited network, the error wave reflects on the border and increase the perturbation inside the network.

To suppress this reflection, we can change connections at the border by removing the link from boundary nodes to inside nodes. As shown in Fig. 16, the isolated border distributes its clock unidirectionally to the kernel surface in which the nodes are fully connected in bidirectional mode.

All ADPLLs (border and kernel) are used to generate local clocks. The network border can be regarded as an independent and synchronous ring exciting the inner kernel as a membrane. This ring, with the reference clock signals at its 4 corners, produces a reference for ADPLLs in the kernel and absorbs the error waves. In the "Swimming pool"-like analogy, the

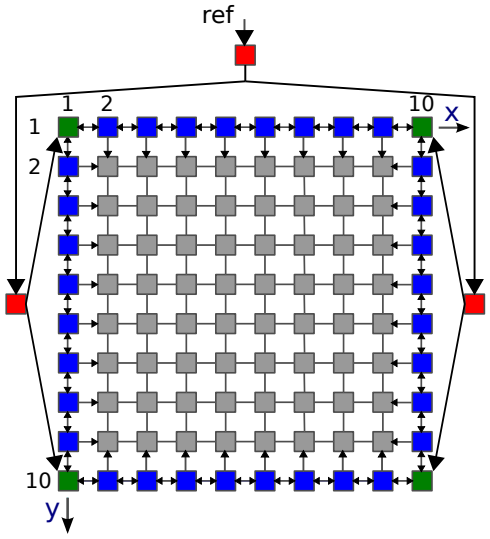

Fig. 16. Proposed network topology (→: unidirectional; ↔, −: bidirectional)

ring of the ADPLL network acts as the overflow channels of a pool.

To observe the transient process of error attenuation, phase errors of local clocks with respect to the reference clock are sampled each cycle and used to create a 3-D animation of the transient process. Fig. 17 shows the phase error surface before the whole network is in phase. We observe that the border is very stable with a relatively low amplitude of errors, while the kernel acts like a membrane fluctuating up and down with an amplitude smaller and smaller until the whole network gets in phase.

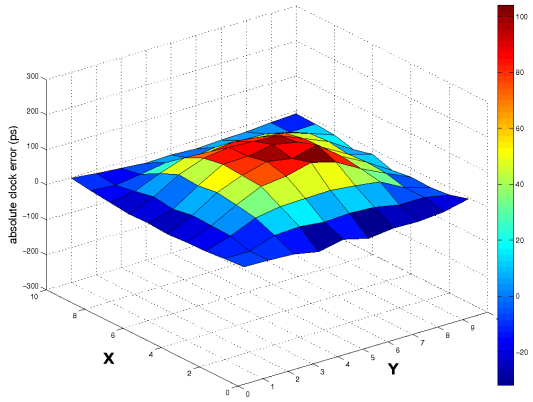

Fig. 17. Phase errors of all clocks in the network at a given moment

V. Conclusion

A distributed clock generator has been proposed in this paper for synchronization in large many-core SoC. Its architecture is based on a network of coupled in phase oscillators using ADPLL technique. The advantage of the proposed approach is its compatibility with digital environment, its flexibility of reconfiguration and scalability to larger dimension topology. Three prototypes and one "swimming pool" like topology have also been presented, which demonstrate the reliability of proposed technique. However, the proposed architecture is a complex system, mixing discrete and continuous time, autosampled and quantized. Although the experimental study proved the operation and functionality of the solution, several theoretical issues remains uncleared. In particular, deeper study is required in order to link the residual synchronization error with the network parameters, and to study the topology of the basin of attraction related with the multiple synchronized modes.

Acknowledgment

This work has been funded by the French National Agency of Research (ANR) under grant ANR-10-SEGI-014-01.

References

[1] A. Abdelhadi et al., *"Timing-driven variation-aware nonuniform clock mesh synthesis"*, Proceedings of the 20th symposium on Great lakes symposium on VLSI, 2010, pp. 15-20

[2] V. Tiwari et al., *"Reducing power in high-performance microprocessors"*, Proceedings of the 35th annual Design Automation Conference, 1998, pp. 732-737

[3] V. Tiwari et al., *"A 600 MHz IA-32 microprocessor with enhanced data streaming for graphics and video"*, 1999 IEEE International Solid-State Circuits Conference, 1999, pp. 98-101

[4] Anceau, Francois, *"A synchronous approach for clocking VLSI systems"*, Solid-State Circuits, IEEE Journal of, vol. 17, n. 1, 1982, pp. 51-56.

[5] S. Rusu et al., *"A 45 nm 8-core enterprise Xeon processor"*, IEEE Journal of Solid-State Circuits, vol. 45, n. 1, 2010, pp. 7-14.

[6] G. A. Pratt et al., *"Distributed Synchronous clocking"*, IEEE transaction on parallel and distributed systems, vol. 6, n. 3, march 1995, pp. 314-328.

[7] V. Gutnik and A. P. Chandrakasan. *"Active GHz clock network using distributed PLLs."* IEEE JSCC, vol. 35, no. 11 (2000): 1553-1560.

[8] E. Zianbetov et al., *"All-digital PLL array provides reliable distributed clock for SOCs"*, IEEE international ISCAS conf., 2011, Rio de Janeiro, pp. 2589-2593

[9] J. A. Thierno et al., *A Wide Power Supply Range, Wide Tuning Range, All Static CMOS All Digital PLL in 65 nm SOI*, IEEE JSSCC, vol. 43, no. 1, January 2008.

[10] Nikishkov, Gennadiy and Nishidate, Yohei, *"Water Surface Animation using Damped Wave Equation and CUDA Acceleration"*, integration, vol. 1, pp. 1-7.

[11] A. Korniienko et al., *"Control Law Synthesis for Distributed Multi-Agent Systems: Applications to Active Clock Distribution Network"*, Automatic and Control Conference, San Francisco, CA, 2010

[12] J. M. Akre, J. Juillard, D. Galayko and E. Colinet. *"Synchronization analysis of networks of self-sampled all-digital phase-locked loops."* Circuits and Systems I, IEEE Trans. on 59, no. 4 (2012): 708-720.

[13] M. Javidan et al., *"All-digital PLL array provides reliable distributed clock for SOCs."* IEEE ISCAS conf., pp. 2589-2592, 2011.

[14] Zianbetov, E and Galayko, D and Anceau, F and Javidan, M and Shan, C and Billoint, O and Korniienko, A and Colinet, E and Scorletti, G and Akrea, JM and others *"Distributed clock generator for synchronous SoC using ADPLL network"*, Custom Integrated Circuits Conference (CICC), 2013 IEEE, pp. 1-4

[15] Shan, Chuan and Galayko, Dimitri and Anceau, Francois *"On-chip clock error characterization for clock distribution system"*, 2013 IEEE Computer Society Annual Symposium on VLSI, Natal, Brazil, 2013, pp. 102-108

[16] Shan, Chuan and Zianbetov, Eldar and Yu, Weiqiang and Anceau, Francois and Billoint, Olivier and Galayko, Dimitri *"FPGA Prototyping of Large Reconfigurable ADPLL Network for Distributed Clock Generation"*, International Conference on Reconfigurable Computing and FPGAs (ReConFig), Cancun, Mexico, 2013, pp. 1-6

[17] Shan, Chuan and Anceau, Francois and Galayko, Dimitri and Zianbetov, Eldar*""Swimming pool"-like distributed architecture for clock generation in large many-core SoC"*, IEEE ISCAS conf., Melbourne, Australia, 2014