Title	Generation of a Clocking Signal in Synchronized All-Digital PLL Networks
Authors(s)	Koskin, Eugene; Galayko, Dimitri; Feely, Orla; Blokhina, Elena
Publication date	2018-06
Publication information	IEEE Transactions on Circuits and Systems II: Express Briefs, 65 (6): 809-813
Publisher	IEEE
Item record/more information	http://hdl.handle.net/10197/9685
Publisher's statement	© 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Publisher's version (DOI)	10.1109/TCSII.2018.2798409

Generation of a Clocking Signal in Synchronised All-Digital PLL Networks

Eugene Koskin[1], Dimitri Galayko[2], Orla Feely[1] and Elena Blokhina[1]

[1]School of Electrical and Electronic Engineering, University College Dublin, Belfield, Dublin 4, Ireland
[2]LIP6 Lab, UPMC Sorbonne Universités, 4 place Jussieu, 75252 Paris Cedex 05, France

***Abstract*—In this paper, we propose a discrete-time framework for the modelling and studying of All-Digital Phase-Locked Loop (ADPLL) Networks with applications in clock-generating systems. The framework is based on a set of nonlinear stochastic iterating maps and allows us to study a distributed ADPLL network of arbitrary topology. We determine the optimal set of control parameters for the reliable synchronous clocking regime, taking into account the intrinsic noise from both local and reference oscillators. The simulation results demonstrate very good agreement with experimental measurements of a 65nm CMOS ADPLL network. Our study shows that an ADPLL network can be synchronised both in frequency and phase. We show that for a large Cartesian network the average network jitter increases insignificantly with the size of the system.**

I. Introduction

A Phase-Locked Loop (PLL) network is a spatially distributed array of controlled oscillators interacting with each other through the exchange of error signals. It has been shown in studies [1]–[6] that networks can be used for generating a *distributed clock signal* in Systems-on-a-Chip (SoCs). This approach has become immensely important as the complexity of microchips and microsystems has increased. Thus supplying them with a coherent clock signal, using conventional clock tree routing, has been proven to be less efficient. The major reason for this is that numerous buffers in clock trees consume significant energy and have propagation delays due to internal relaxation processes. This results in signal distortions with skew and jitter, which are the major parasitic effects in clock generating networks.

In the most common case, each oscillator (or PLL) of a network can be allocated as a node on a Cartesian grid as shown in Fig. 1(a). Every oscillator has conventional components such as a voltage (or digitally) controlled oscillator (VCO or DCO respectivly) that generates a signal with a given frequency, a control block (also known as a loop filter) that provides a control signal for the controlled generator and a phase-frequency detector (PFD) that produces an error signal for the control block. The number of PFDs for each oscillator in a network is equal to the number of its neighbours. It has been shown in [3] that when at least one node in the network is connected to an external reference signal, the network can be synchronised with this signal both in frequency and phase.

An ADPLL network has some common features with pulse-coupled oscillators for wireless sensor networks. Both types of networks can be seen as event-driven networks with promising potential for Internet of Things applications [7], [8]. In a simple case, pulse-coupled networks consist of arrays of distributed phase oscillators having identical natural frequencies. The interaction between the oscillators occurs due to exchanging pulses at firing time instances, *i.e.*, when the phase of an oscillator is a multiple of 2π. At the same time, the phase of other topologically interconnected oscillators instantaneously changes by a value determined by the phase response function. If the natural frequencies are non-identical, their phases may not be synchronised and additional optimisation techniques are required [9]. In ADPLL networks, the phase of each oscillator is a continuous function of time, but its frequency changes instantaneously at the beginning of a new clocking cycle with respect to the input control code.

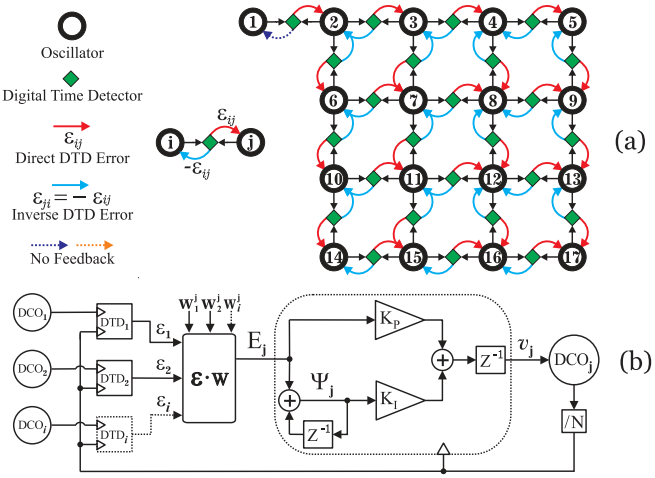

Fig. 1. Schematics of an ADPLL network. (a) The network is driven by a reference clock (DCO #1) on a square Cartesian grid. The interaction between DCOs consist of the exchange of error signals between closest neighbours. (b) The frequency control block that includes digital time detectors, averaging block, PI filter and frequency divider.

ADPLLs are complex systems and the conventional method for their analysis is behavioural modelling using MATLAB Simulink or hardware/mixed-signal description languages. To date there are only a few analytical models that have been suggested: discrete-time models [4], [10], [11] and pulse-event driven models for synchronization in wireless networks [7], [12]–[14].

In this study, we expand our model to the case of ADPLL *networks* having arbitrary topology. Based on this model, we investigate the dynamics of an ADPLL network by varying the parameters of the system. The validation of the mathematical

model is carried out through a number of experiments made on chip that has been proposed in [15].

The paper is organised as follows. Sec. II describes an ADPLL network in general. The description is extended in Sec. III where we use a graph and adjacency matrix to define the topology of a network. The experimental verification is given in Sec. IV, and key results obtained from the model are presented in Sec. V.

II. Statement of the Problem

An ADPLL network is an array of DCOs topologically connected with each other via digital time detectors (DTDs) as shown in Fig. 1(a). In this study we describe a DCO as a single block DCO and its divider. The external reference signal (its period is T^{R}, frequency is F^{R} and $T^{\mathrm{R}} = 1/F^{\mathrm{R}}$) is a part of the network and represented by a reference oscillator (see oscillator 1 in Fig. 1(a)).

The network is described in terms of a directed graph $G = (V, E)$ with vertices V and edges E. Each oscillator is located at a vertex of the graph, and we number the oscillators from 1 to $N_{\mathrm{DCO}} + 1$, where N_{DCO} is the number of DCOs in the network. Every detector in the network we associate with an edge of the graph and number the edges from 1 to N_{DTD}, where N_{DTD} is the number of detectors in the network. We will then refer to a detector using the index k. For example, the detector DTD$[k]$ connects two vertices (oscillators) numbered i and j.

A DTD has two inputs for the comparison of the time interval between clocking events of oscillators it connected to. We assume that the signal coming to DTD$[k]$ from the oscillator whose number is minimal in the pair (i, j) is the reference (with respect to the k^{th} detector) while the signal coming from the other oscillator is local. Every detector updates its *state* (intrinsic set of variables such as memory and error values) only when a rising edge of either the reference or local signal arrives at its input. At DTD$[k]$ the time intervals to the closest local and reference events are denoted as L_k and R_k respectively. The sets of local and reference time events for all time detectors $k = 1, ..., N_{\mathrm{DTD}}$ are presented as $\mathbf{L}$ and $\mathbf{R}$.

The output of a DTD represents the timing error ε_k (also called the error signal in this paper) that characterises the mismatch in the DCO clocking events of adjacent oscillators, as shown in Fig. 1(b). The error signal ε_k controls the state of the local oscillator (see the red arrow in Fig. 1(a)). The same error but with the opposite sign, $-\varepsilon_k$, controls the reference oscillators (see the blue arrow in Fig. 1(a)).

The timing error from several DTDs is averaged, and control of a DCO is done by passing the averaged timing error E_j through a PI controller, with integral and proportional gains K_{i} and K_{p} respectively. As a result, we obtain the control code $v_j = K_{\mathrm{p}} E_j + K_{\mathrm{i}} \Psi_j$, where Ψ_j is the accumulated error.

We employ a DTD introduced in [4] which consists of the combination of a finite-state machine and time-to-digital converter. It transforms the time difference $\tilde{\tau}_k$ between the rising edges of the reference and local signals into the error $\tilde{\tau}_k \overset{\mathrm{H}}{\longmapsto} \varepsilon_k$ represented as an integer number in the range $[-7; -1] \cup [1; 7]$, where H is a quantization function of the DTD. A DCO generates a rectangular clock signal with the frequency $f_j = f_0 + \Delta f \, v_j$, where f_0 is the initial frequency and Δf is the frequency gain. The frequency lies within a range bounded by minimum and maximum values. The range depends on the integrator bit resolution and the frequency gain. If the control block is unable to provide a frequency value beyond the control range, it resets its integrator to zero and sets the DCO frequency to its initial value. The model of such an ADPLL was developed in [16]. It describes self-sampling, event-based frequency tuning, quantization of control signals and noise effects. We will use this model here, keeping in mind that the main focus of this study is the investigation of the network. An interested reader can find the equations in the above reference.

III. Defining the Topology of an ADPLL Network and Simulation Algorithm

In this section we will represent the graph of an ADPLL network graph through an adjacency matrix $\mathbf{A}$, where $A_{i,j} = 1$ if the i^{th} DCO affects the j^{th} DCO, otherwise $A_{i,j} = 0$. In our case the first column is all-zero due to the presence of an external reference signal, whilst for a purely bi-directional graph, the matrix is symmetric. Each DCO does not interact with itself, therefore, all diagonal elements of the matrix are equal to zero.

Having determined the adjacency matrix, we define a set $\mathbf{D}$ of DTDs in a straightforward manner. For every nonzero element in $\mathbf{A}$ we create a 2-tuple with corresponding matrix indexes sorted in ascending order. Next, we create the array $\mathbf{D}$ based on the tuples excluding repeated elements. The length of $\mathbf{D}$ is equal to the number of DTDs in the network N_{DTD}. Each D_k contains two values $D_k = (i, j)$ where $i < j$. According to the definition of the reference and local oscillators in Sec. II, the first element in D_k corresponds to the index of its reference oscillator while the second element corresponds to the local oscillator.

Now we assign the matrix $\mathbf{W}$ of weight coefficients shown in Fig. 1(b). Here we assume that each DCO is tuned by the average error from its neighbours. The total number of neighbours for the j^{th} oscillator is determined by the adjacency matrix: $g_j = \sum_{i=1}^{N_{\mathrm{DCO}}+1} A_{i,j}$. Thus, the weight matrix is introduced as follows: if $g_j \neq 0 \Rightarrow W_{i,j} = A_{i,j}/g_j$, otherwise $W_{i,j} = 0$. Note that as it is costly to divide by 3 in digital implementation, the prototype presented in this study uses division by 4.

Algorithm 1: GetWeightedError finds the averaged error supplied to the PI control block of a DCO.

Input: Weight matrix $\mathbf{W}^{(N_{\mathrm{DCO}}+1)\times(N_{\mathrm{DCO}}+1)}$, a set of DTDs $\mathbf{D}^{N_{\mathrm{DTD}}\times 2}$, a set of DTD errors $\varepsilon^{N_{\mathrm{DTD}}}$

Output: Set of weighted errors for each DCO $\mathbf{E}^{N_{\mathrm{DCO}}+1}$

$\mathbf{E} \leftarrow 0^{N_{\mathrm{DCO}}+1}$

for $k \leftarrow 1$ ***to*** N_{DTD} **do**

  $i \leftarrow D_{k,1}, \quad j \leftarrow D_{k,2}$

  $E_j \leftarrow E_j + W_{i,j}\varepsilon_k, \quad E_i \leftarrow E_i - W_{j,i}\varepsilon_k$

return E

Below, we describe the structural algorithm for the time evolution of the ADPLL network, where the average error E_i that is supplied to the PI controller of a DCO in the network is calculated using algorithm 1.

1) Generate an ADPLL network with a set of given initial conditions and system parameters.
2) Find the minimum time to the next clocking event among all detectors: $t_{\min} = \min(\mathbf{R} \cup \mathbf{L})$.
3) Make time shift for all detectors: $\mathbf{R} \leftarrow \mathbf{R} - t_{\min}\mathbb{1}$, $\mathbf{L} \leftarrow \mathbf{L} - t_{\min}\mathbb{1}$, where $\mathbb{1}$ is an all-ones vector. If $R_k = 0$ or $L_k = 0$ then corresponding reference/local oscillator starts a new cycle.
4) Find indices of oscillators which start a new cycle.
5) Recalculate average error E_i for all oscillators using algorithm 1.
6) Update frequencies and controller states of selected oscillators from step 4. That requires four stages:
   a) Set a frequency according to a control code $f_i = f_0 + \Delta f(K_{\mathrm{p}}E_i + K_{\mathrm{i}}\Psi_i)$.
   b) If new value appears to be outside generating range then reset its value to the initial DCO frequency f_0 and set integrator to zero $\Psi_i = 0$.
   c) Generate frequency jitter by multiplying it with a *log-normal* random number: $f_i \leftarrow f_i \exp(N(0, \sigma))$. Here $N(0, \sigma)$ is the Gaussian normal distribution with zero average and standard deviation σ.
   d) Update clocking time for detectors connected to selected oscillators: $R_k \leftarrow 1/f_i$ or $L_k \leftarrow 1/f_i$.
   e) Update selected integrators: $\Psi_i \leftarrow \Psi_i + E_i$.
7) Update state of all detectors and simulating time.
8) Go to step 2 while current simulation time is less than the target value.

In this work we assumed that all control and model parameters such as K_{p}, K_{i}, σ, f_0, Δf, etc., are the same for every DCO.

IV. Experimental Validation and Comparison With the Model

The aim of the experiment is to validate the framework proposed in Sec. III.

The validation consists of experimentally measured transient responses to a periodical frequency change of the reference signal in an ADPLL network. The experiment has been performed using the prototype of a 4×4 (can be reconfigured as 2×2) ADPLL network fabricated in 65nm CMOS technology. The architecture of the microchip is described in [5].

The parameters of the ASIC implementation of an ADPLL network are: V_{DD} = 1.0 V, Δf = 150 kHz, $F_{\min\ \mathrm{DCO}}$ = 135 MHz, $F_{\max\ \mathrm{DCO}}$ = 175 MHz, resolution time of a time-to-digital converter in a DTD τ_{TDC} = 20 ps, number of levels in DTD N_{D} = 7.

In order to observe a transient response, the reference signal generator was programmed to produce a rectangular signal, whose frequency was swapping between two values. This was achieved by the periodic changing of the generator's division coefficient (with period ≈ 17 μs) between two values chosen from the set $\{6, 6.5, 7\}$. The main frequency of the generator in the experiments (before applying the division coefficient) was either 1000 MHz or 950 MHz.

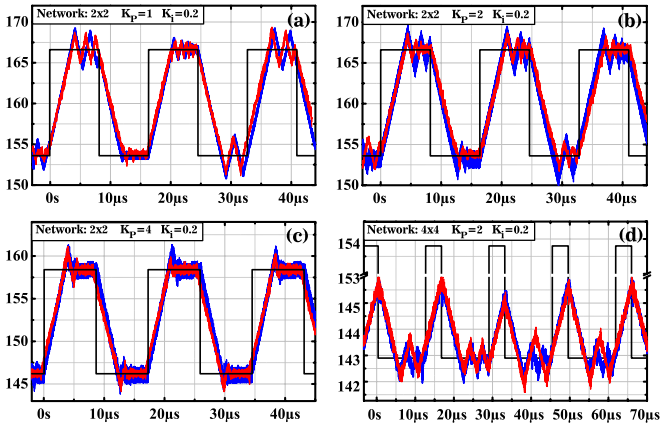

Fig. 2. Measured and simulates results: the dependence of frequency (MHz) on time (μs) for ADPLL networks. The measured signals are taken from oscillators 1 (reference, black), 2, 3 and 7 (blue) in Fig. 1(a). The simulated dynamics is shown by red. Because all the DCOs are synchronised, their waveforms overlap and cannot be clearly distinguished in the figure. The reference frequency is switched between $1000/6$ MHz and $1000/6.5$ MHz with 50% duty cycle in cases (a) and (b) for a 2×2 network, then between $950/6$ MHz and $950/6.5$ MHz with 50% duty cycle in case (c) for a 2×2 network and then between $1000/7$ MHz and $1000/6.5$ MHz with 75%-25% duty cycle in case (c) for a 4×4 network.

After each switching, the reference frequency remains constant for enough time to observe the frequency acquisition by the network. The frequency acquisition process is limited by the switching period (and does not always settle to a synchronous mode). The transient process lasts till the next switching. Because we apply a periodic signal of the modulated reference frequency, we observe a sequence of repeating similar transient waveforms.

This setup was designed to verify the framework from Section III for modelling the ADPLL network. The results are shown in Fig. 2, where we display very different transient responses that network experiences when we change (i) PI controller coefficient K_{i} and K_{p}; (ii) the frequency of the reference signal $F^{\mathrm{R}} = 1/T^{\mathrm{R}}$; (iii) the duty cycle of the reference signal modulation and (iv) the number of PLLs in the network. Each plot in this figure contains the measured signals and simulation results. The four different examples shown in the figure demonstrate the ability of the model to predict correctly the behaviour of the network and, most importantly, its transient dynamics in very different situations. We deliberately show transient processes and the frequency acquisition by the network since these provide more information and are more valuable for the verification of the model. Note that the signals are almost identical and cannot be distinguished in the figure. This is due to the fact that all initial conditions for DCOs in the network differ only in phases while starting frequencies are the same. Hence, phase synchronisation between DCOs occurs quickly, just after the network is switched on. After that, the network behaves as a single oscillator whose frequency moves slowly toward the reference frequency.

We note that Fig. 2(a)-Fig. 2(c) display the results for a

2×2 ADPLL network, and Fig. 2(d) shows the result for a 4×4 ADPLL network. Because its behaviour is qualitatively the same as in the 2×2 network case, we have chosen one experiment to demonstrate frequency dynamics. Also in this case we applied a "non-symmetrical" duty cycle of reference frequency modulation (75% / 25%) to prove that the model will capture the behaviour of the network even in this cases.

The comparison presented Fig. 2 allows us to conclude that the model is verified and describes various dynamics, transient processes and synchronised modes. It is also extremely accurate in the prediction of the frequency acquisition rate F'. There is indeed a difference between the envelope width in steady state. This is clearly due to larger supply noise which negatively affects the jitter measured from the chip (since the output pad buffers are also supplied by noisy V_{DD}). The contribution of V_{DD} is difficult to quantify precisely, given the complexity of the microchip. However, the fact that the model predicts consistent dynamical behaviour in transient mode is an indication of its correctness.

In addition, the figure gives rise to the hypothesis that the frequency acquisition rate likely does not depend on K_{p}, and it decreases with the size of the network if other conditions remain the same. In the next section, we use our framework to show that this is indeed true. Moreover, we show the parameters domain where synchronization is possible and suggest how the overall network performance could be optimised.

V. Performance of the Network

In this section we consider dynamics of ADPLL networks based on the framework proposed in Sec. III. The simulation parameters correspond to the experimental setup described in Sec. IV. We note that these parameters have not been chosen through a fitting, but are taken "as is" from the experiment.

Fig. 3 shows a typical synchronisation pattern that appears in ADPLL networks given the proper choice of control parameters, such as K_{p} and K_{i}. From the picture all basic dynamical features of a PLL are clearly seen: phase-frequency acquisition, phase locking and phase tracking. We can see that for random phase and frequency initial conditions all DCOs in the network synchronise between themselves, and later they synchronise with the external reference signal both in frequency and phase.

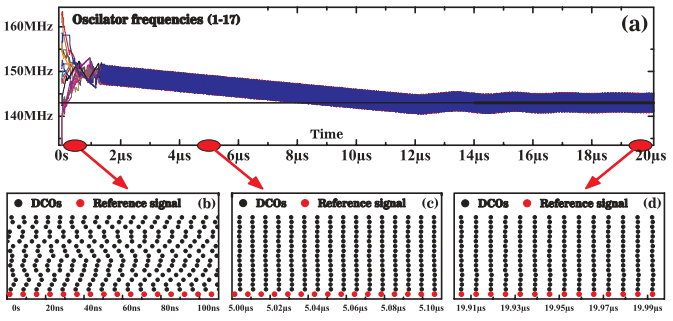

Fig. 3. Simulation results for a 4×4 ADPLL network with random initial phase and frequency conditions: (a) Frequency dynamics, (b)-(d) event dynamics. Fig. (d) demonstrates phase synchronicity. Each dot in Figs. (b)-(d) corresponds to a rising edge of an oscillator with the corresponding number. The parameters of the network were taken from the experiment in Fig. 2(e).

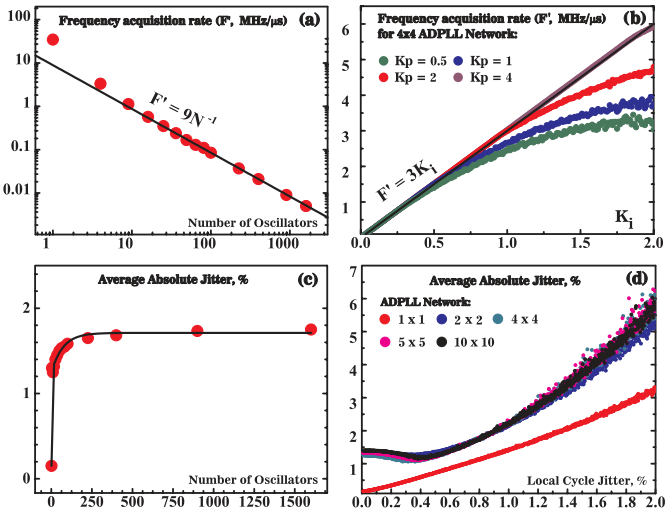

Fig. 4. The dependence of the frequency acquisition rate on: (a) DCO number in a square Cartesian ADPLL network, (b) integral gain factor for 4×4 network. The dependence of relative average network jitter on: (c) DCO number in a square Cartesian ADPLL network, (d) DCO intrinsic noise.

In order to quantify the synchronisation in ADPLL networks we shall use the *averaged network jitter* $\langle|\Delta\tau|\rangle$ as the measure of synchronisity:

$$\langle|\Delta\tau|\rangle = N_{\mathrm{DTD}}^{-1} \sum_{k=1}^{N_{\mathrm{DTD}}} \lim_{t\to\infty} t^{-1} \int_0^t |\tilde{\tau}_k(t')|\, dt'$$

This measure represents the average through all the detectors and time error $\tilde{\tau}_k(t)$. It is equal to zero when the network operates synchronously and all rising edges come at the same time. In addition to this, we shall consider the *relative average jitter* that is normalised to the reference period: $\langle|\Delta\tau|\rangle / T^{\mathrm{R}} \cdot 100\%$.

Frequency acquisition. Fig. 4(a) shows the dependence of the frequency acquisition rate F' on the number of DCOs in a network. Here, we have neglected the intrinsic cycle jitter of the reference signal and DCOs. We set the PI control parameters to be $K_{\mathrm{p}} = 2$ and $K_{\mathrm{i}} = 0.2$ (as in the experiment in Fig. 2(c) to ensure a synchronisation in the network). We can see that F' decreases following a *power law* as the number of DCOs in the network increases. We shall give a brief intuitive interpretation of this phenomenon, but a more precise explanation is an issue for future study.

When the frequencies of all DCOs significantly differ from the reference frequency, the average frequency acquisition rate is predominantly defined by the error signal between the reference clock and the first oscillator. As we add more oscillators to a network, this "driving" signal propagates within the whole network with a finite speed that depends on the DCO frequencies. For larger networks, it takes more time for the signal to propagate from the reference clock to the network's edges. Therefore, the average frequency acquisition rate decreases.

Fig. 4(b) shows the dependence of the frequency acquisition rate on the integral gain factor for different proportional gain

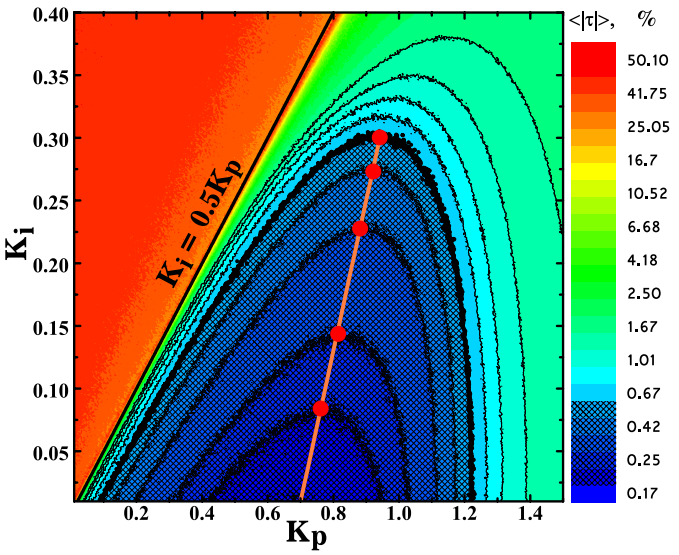

Fig. 5. The dependence of relative network jitter on the PI control parameters for 4×4 ADPLL network in a steady-state. The DCOs' intrinsic jitter is set to be 0.1%, and $F^{\mathrm{R}} = 167$ MHz. The orange line shows the optimal set of K_{p} that minimises the average network jitter for a given K_{i}. The dashed area corresponds to $\approx 0.5\%$ network jitter. The plane contains 400×400 points.

values. We can see that for small K_{i} ($K_{\mathrm{i}} < 0.5\, K_{\mathrm{p}}$) F' depends on K_{i} linearly, and does not depend on K_{p}.

Network jitter. Fig. 4(c) shows the relative average jitter in a square Cartesian network. The intrinsic cycle jitter of both reference and local oscillators is set to zero and PI control parameters are $K_{\mathrm{p}} = 2$ and $K_{\mathrm{i}} = 0.2$. We can see that $\langle|\tau|\rangle$ changes insignificantly when the DCOs number is large, therefore, the network tends to be stable when the number of oscillators increases. This behaviour is similar to the dynamics of a network of noisy Kuramoto-like oscillators, where, under certain conditions, the order parameter stays still close to unity even though the number of oscillators approaches to infinity.

Fig. 4(d) shows the dependence of network jitter on the intrinsic noise of DCOs in a steady state. We can see that the dependence tends to be the same for any large network. The detailed analysis of the asymptotic behaviour of large ADPLL networks could be an interesting theoretical challenge for a future work.

Plane of parameters. Fig. 5 shows the relative averaged network jitter in the plane spanned by the PI control parameters. From this figure, we conclude that the system experiences dynamical instability when the integral gain K_{i} exceeds $\approx\ 0.5K_{\mathrm{p}}$. We can see that in this case, the relative network jitter equals $\approx\ 50\%$. This can be the case when all rising edges of DCOs in the network are independently and uniformly distributed in time within one period of the reference clock being desynchronized in phase. Moreover, for every K_{i} there is a value K_{p} where $\langle|\tau|\rangle$ is minimal. Due to the fact that the frequency acquisition rate is proportional to K_{i} (see Fig. 4(b)), it appears to be possible to choose an optimal proportional gain K_{p} that minimises the network jitter for given frequency acquisition rate.

VI. Conclusions

Using the analysis developed in the paper we have proved the feasibility of ADPLL networks to generate a synchronous clocking signal for parallel computations, for example, in application to ASIC design. We have shown that the network synchronises in phase and frequency for a given set of control parameters. No mode-locking was observed during experiments or numerical simulations. We have highlighted the possibility to minimise network jitter for a given frequency acquisition rate by the optimal choice of the proportional gain factor. We have shown that for large networks the average jitter increases insignificantly with respect to the oscillator's number. However, the frequency acquisition rate decreases in accordance to the power law. Finally, due to the discreet nature of the model, it works faster than existing behavioral models.

References

[1] G. Pratt and J. Nguyen, "Distributed synchronous clocking," *IEEE Trans. Parallel Distrib. Syst.*, vol. 6, no. 3, pp. 314–328, 1995.

[2] E. G. Friedman, "Clock distribution networks in synchronous digital integrated circuits," *Proc. IEEE*, vol. 89, no. 5, pp. 665–692, 2001.

[3] V. Gutnik and A. Chandrakasan, "Active GHz clock network using distributed PLLs," in *IEEE J. Solid-State Circuits*, 2000, pp. 174–175.

[4] J.-M. Akre, J. Juillard, D. Galayko, and E. Colinet, "Synchronization analysis of networks of self-sampled All-Digital Phase-Locked Loops," *IEEE Trans. Circuits Syst. I*, vol. 59, no. 4, pp. 708–720, 2012.

[5] C. Shan, D. Galayko, F. Anceau, and E. Zianbetov, "A reconfigurable distributed architecture for clock generation in large many-core SoC," in *in Proc. IEEE Int. Symp. Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC)*, 2014, pp. 1–8.

[6] A. Agrawal and A. Natarajan, "2.2 a scalable 28GHz coupled-PLL in 65nm CMOS with single-wire synchronization for large-scale 5G mm-wave arrays," in *in Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, 2016, pp. 38–39.

[7] F. Núñez, Y. Wang, and F. J. Doyle, "Synchronization of pulse-coupled oscillators on (strongly) connected graphs," *IEEE Trans. Autom. Control*, vol. 60, no. 6, pp. 1710–1715, June 2015.

[8] F. Núñez, Y. Wang, A. R. Teel, and F. J. Doyle, III, "Synchronization of pulse-coupled oscillators to a global pacemaker," *Systems & Control Letters*, vol. 88, pp. 75 – 80, 2016.

[9] Y. Wang and F. J. Doyle, III, "Optimal phase response functions for fast pulse-coupled synchronization in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5583–5588, Oct 2012.

[10] N. Da Dalt, "A design-oriented study of the nonlinear dynamics of digital bang-bang PLLs," *IEEE Trans. Circuits Syst. I*, vol. 52, no. 1, pp. 21–31, 2005.

[11] C.-Y. Zhang, Y.-b. Li, H.-M. Li, Q. Wang, and Y.-l. Peng, "The implementation and analysis of a new self-sampling pi control All Digital Phase-Locked Loop," in *in Proc. IEEE Int. Conf. Machine Learning and Cybernetics*, 2006, pp. 241–246.

[12] A. V. Proskurnikov and M. Cao, "Synchronization of pulse-coupled oscillators and clocks under minimal connectivity assumptions," *IEEE Trans. Autom. Control*, vol. PP, no. 99, pp. 1–8, 2016.

[13] R. Gentz, A. Scaglione, L. Ferrari, and Y. W. P. Hong, "Pulsess: A pulse-coupled synchronization and scheduling protocol for clustered wireless sensor networks," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1222–1234, Dec 2016.

[14] D. Kannapan and F. Bullo, "Synchronization in pulse-coupled oscillators with delayed excitatory/inhibitory coupling," *SIAM J. Control Optim.*, vol. 54, no. 4, pp. 1872–1894, 2016.

[15] C. Shan, E. Zianbetov, F. Anceau, O. Billoint, and D. Galayko, "A distributed synchronization of all-digital PLLs network for clock generation in synchronous SOCs," in *in Proc. IEEE Int. Conf. New Circuits and Systems (NEWCAS)*, June 2015, pp. 1–4.

[16] E. Koskin, E. Blokhina, C. Shan, E. Zianbetov, O. Feely, and D. Galayko, "Discrete-time modelling and experimental validation of an all-digital PLL for clock-generating networks," in *in Proc. IEEE Int. Conf. New Circuits and Systems (NEWCAS)*, June 2016, pp. 1–4.