

FPGA implementation of reconfigurable ADPLL network for distributed clock generation

Chuan Shan, Eldar Zianbetov, Mohammad Javidan, François Anceau, Mehdi Terosiet, Sylvain Feruglio, Dimitri Galayko, Olivier Romain, Eric Colinet, Jérome Juillard

▶ **To cite this version:**

Chuan Shan, Eldar Zianbetov, Mohammad Javidan, François Anceau, Mehdi Terosiet, et al.. FPGA implementation of reconfigurable ADPLL network for distributed clock generation. FTP 2011 - International Conference on Field Programmable Technology, Dec 2011, New Delhi, India. pp.1-4, 10.1109/FPT.2011.6132670 . hal-01053755

HAL Id: hal-01053755
https://hal.sorbonne-universite.fr/hal-01053755

Submitted on 4 Aug 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

FPGA Implementation of Reconfigurable ADPLL Network for Distributed Clock Generation

Chuan Shan [1], Eldar Zianbetov [1], Mohammad Javidan [1], François Anceau [1], Mehdi Terosiet [1], Sylvain Féruglio [1], Dimitri Galayko [1], Olivier Romain [2], Éric Colinet [3], Jérôme Juillard [4]

[1] *LIP6 - UPMC Sorbonne Universités, Paris, France*
[2] *Laboratoire ETIS UMR8051, Universit de Cergy-Pontoise, France*
[3] *Supélec, Gif-suf-Yvette, France,* [4] *CEA-LETI, Grenoble, France*

***Abstract*—This paper presents a FPGA platform for the design and study of network of coupled all-digital phase locked loops (ADPLLs), destinated for clock generation in a large synchronous system on chip (SOC). An implementation of a programmable and reconfigurable 4×4 ADPLL network is described. The paper emphasises the difference between the FPGA and ASIC-based implementation of such a system, in particular, implementation of digitally contolled oscillators and phase-frequency detector. The FPGA-implemented network allows to study complex phenomena related with coupled ADPLL operation, and to exploit stability issues and nonlinear behaviour. A dynamic setup mechanism has been proposed for the network, allowing to select the desirable synchronised state. Experimental results demonstrate the global synchronization of network and performance of the network for different configuraitons.**

I. INTRODUCTION

Clocking is one of the main issues in design of large circuits on chip (SOC) like manycore processors. synchronous communication is still largely used in SOCs, espetially for application requiring high reliability. However, practical implementation of a global clock distribution is very difficult in advanced deeply submicronic CMOS technologies. Our study explores an alternative technique of clocking, consisting in distributed clock generation through a network of all-digital PLLs. Such a network is composed of several oscillators distributed over the chip, which are coupled in the phase domain. Each oscillator generates a local clock. A network properly designed guarantiees that all oscillators generates a signal at the same frequency and with the same phase. The originality of our study is the fact that the network of PLL is fully digital, contrary to previous implementations [1].

A network of coupled digital PLLs is a very complex nonlinear high-degree dynamic system, having different operating modes, among which only some of them are desirable. The selection of the mode is done through the appropriate choice of the network parameters (the ADPLL filter coefficients, the choice of initial conditions, etc.), and requires a solid underlying theory and a prototyping platform. This work presents an implementation of such a prototype on a single FPGA chip. The goal of the prototype is to model as close as possible the behaviour of an ASIC-based ADPLL network, which is being designed in the LIP6 analog/mixed circuit group. The FPGA emulator must have exactly the same architecture with the same parameters values, however, scaled down proportionally in frequency because of the maximum frequency limit of the FPGA board. However, the FPGA emulator must reproduces with high fidelity the functional behaviour of the designed ASIC chip.

However, the FPGA-implementation is different from the implementation of their ASIC counterpart, in particular, in blocks whose operation is based on controllable pure delays, like time-to-digital converter (TDC) and the digitally controlled oscillator (DCO). The paper describes the way to prototype these elements on FPGA, so to model as close as possible the ASIC-based blocks.

The validation of the prototype is done through testing the network synchronisation with different filter coefficient, and by observing nonlinear phenomena predicted theoretically for a PLL network. The method of selection of the synchronised mode presented in [2] has been implemented and tested.

In section II the architecture of network and functional blocks are described. Section III presents the procedure allowing to respect the homothety in dimensionning of the FPGA prototype with repsect to the ASIC prototype. Section IV presents experimental results.

II. NETWORK ARCHITECTURE

The topology of network is presented in Fig. 1 [2]. It is composed of Phase Frequency Detectors (PFD) and 16 filter/oscillator (FO) blocks. PFDs are placed on each border between two FO blocks, measuring the phase error between each couple of neighboring oscillators. The PFD placed in upper left corner compares the phase of the input reference and the first oscillator in the network. Such a network, if properly designed, will be synchronized with the reference clock both in frequency and in phase.

The structure of a typical network node is presented in Fig. 2 [3]. Each node contains 2-4 PFDs: each PFD detects the phase/frequency difference between the locally-generated clock and a neighboring clock. The number of PFDs depends on the number of the neighbors of a node. A PFD generates

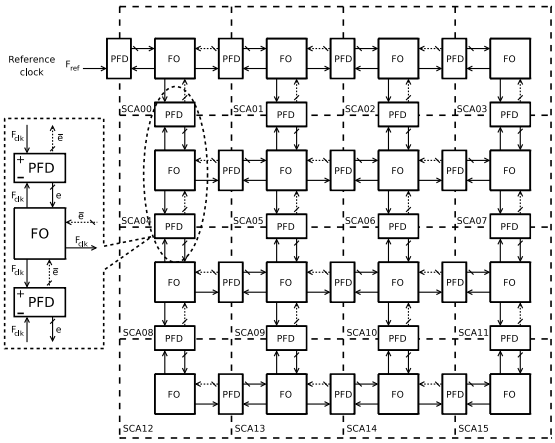


Fig. 1. Topology of network

a 5 bits signed binary code. For each node, the errors with neighbours are added and are processed by the loop filter, so to generate a control word for the digital controlled oscillator (DCO). The loop filter is a proportional-integral (PI) filter similar to one used in classical single ADPLLs. The frequency divider is used to generate a clock with a frequency higher than that at which the error phase information is processed.

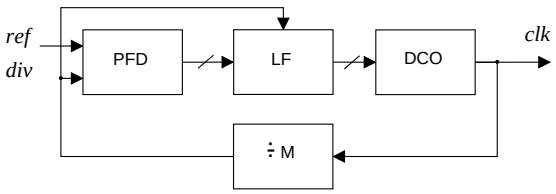


Fig. 2. Architecture of ADPLL

A. PFD

Fig. 3 presents the structure of a PFD. The Bang-Bang phase-frequency detector (BBPFD) detects the sign of phase error ($SIGN$) and the interval of it ($MODE$) [4]. The duration of the signal $MODE$ represents the absolute value of the phase error; it is applied to the input of Time-to-Digital Converter (TDC). The TDC converts the duration of input signal to an unsigned binary code, which is then combined with the $SIGN$ signal by the arithmetic block to form a signed binary code.

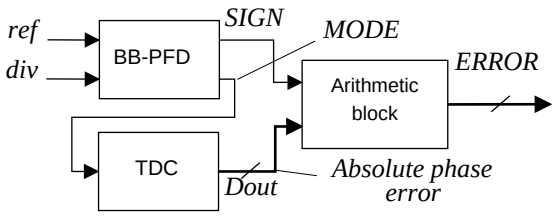


Fig. 3. Architecture of PFD

The BBPFD can be seen as a finite state automaton driven by events (rising edges) at its inpus. For this implementation, the architecture presented in [4], is described at behavioural level for the FPGA synthesis. In an ASIC-based TDC, the $MODE$ is delayed by a tapped delay line [5]. At the falling edge of $MODE$, the thermometer code produced by the delay line is stored in a register and then converted to an unsigned binary code. The sensitivity of PFD (ΔT_{ASIC}) is defined as one stage delay in the delay line. Since a logic gate with defined delay cannot be realized in FPGA, in the FPGA-based network, the TDC is implemented as a digital chronometer with an external clock, counting the number of the clock event included in the measured time interval. The period of the external clock corresponds to the sensitivity of PFD (ΔT_{FPGA}). However, the external clock is desynchronized with the start of the time interval to measure. Hence, even if the time interval is below the TDC time step and if the clock event happens to be inside the interval, the FPGA-based TDC may output 1, whereas the ASIC-based TDC outputs 0, which causes a signal-correlated noise with amplitude of ± 1 unit over the output code; the sign of the noise is the same as the sign of the error. The transfer function is presented in Fig. 4.

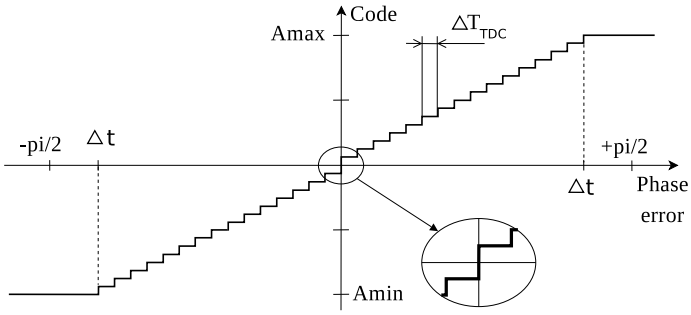


Fig. 4. Transfer function of PFD

B. Loop Filter

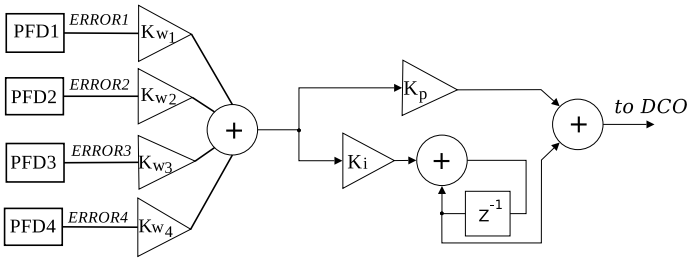


Fig. 5. Architecture of the loop filter

A proportional-integral (PI) filter is implemented as the loop filter. Its transfer function is described as follows:

$$H(z) = K_p + \frac{K_i}{1 - z^{-1}} \tag{1}$$

where K_p and K_i are gain coefficients of the proportional and integral paths respectively. Their values are programmable. The filter has up to 4 inputs (Fig. 5), which are weighted with

programmable coefficients K_{w_i}. The number of inputs can be regulated by programming the coefficients K_{w_i} to one or zero.

The programming of the filter coefficients is designed so that all the programmable values of the network can be changed in parallel, at the same time moment. This is achievied with a simple serial-to-parallel register used for the programming value loading. This allows to test scenarii of dynamic reconfiguration of the network.

C. DCO

Because of the limits of FPGA, a ring oscillator with programmable delay is difficult to be implemented on FPGA. Instead, the DCO is implemented as a pre-loaded N_c-bit counter which defines the DCO clock period, where N_c is decided by the number of bit of command signal. The DCO based on counter uses an external clock with a high frequency ($f_{DCO\ clk}$). When the counter saturates, an output event is generated, and at the same time the counter will be reloaded with the DCO input code. Hence, the period of the clock ($T_{DCO\ FPGA}$) generated by DCO for a some input code C_{in} is defined as

$$T_{DCO\ FPGA}(C_{in}) = (2^{N_c} - C_{in}) \times T_{DCO\ clk} \qquad (2)$$

where $T_{DCO\ clk}$ is the period of the external clock.

The frequency step of DCO in the FPGA prototype ($\Delta f_{DCO\ FPGA}$) is defined as

$$\Delta f_{DCO\ FPGA} = \frac{1}{T_{oFPGA}} - \frac{1}{T_{oFPGA} + T_{DCO\ clk}} \qquad (3)$$

where T_{oFPGA} is the period value corresponding to the nominal frequency of FPGA-based DCO (f_{oPFGA}).

D. Scaling of FPGA prototype parameters

Due to the limit of maximum frequency in FPGA device, all the temporal parameters in the prototype have to be scaled down propotionally respecting the following relation

$$\begin{aligned} &\Delta f_{DCO\ ASIC} : f_{oASIC} : \frac{1}{\Delta T_{ASIC}} \\ =&\Delta f_{DCO\ FPGA} : f_{oFPGA} : \frac{1}{\Delta T_{FPGA}} \\ =&M_1 : M_2 : M_3 \end{aligned} \qquad (4)$$

where f_{oASIC} and $\Delta f_{DCO\ ASIC}$ are respectively the nominal frequency and the frequency step of the ASIC-based DCO. M_1, M_2, and M_3 are three constants used to simplify the derivation.

From Eq.(3) and Eq.(4), the ratio between the external clock frequency of FPGA-based TDC ($f_{TDC\ clk}$) and that of FPGA-based DCO ($f_{DCO\ clk}$) can be derived (Eq.(5)). $f_{TDC\ clk}$ and $f_{DCO\ clk}$ are respectively the inverse of the sensitivity of PFD (ΔT_{FPGA}) and the inverse of $T_{DCO\ clk}$. The two clocks are generated respectively by two PLLs located in two corners of the device, thus are uncorrelated with each other.

$$\frac{f_{TDC\ clk}}{f_{DCO\ clk}} = \frac{M_3}{M_2} \times \frac{1}{M_2/M_1 - 1} \qquad (5)$$

Using the ASIC parameters given in TABLE I, the ratio can be calculated ($f_{TDC\ clk}/f_{DCO\ clk} \simeq 0.1068$). If $f_{DCO\ clk}$ uses the largest frequency value of device, the other temporal parameters could be calculated by using the ratio equation (Eq.4). The result is shown in TABLE I.

TABLE I
MAIN CHARACTERISTICS OF SYSTEM

	ASIC	**FPGA**
PFD resolution (ΔT)	30 ps	149.88 ns
DCO gain (Δf)	200 KHz/LSB	40.03 Hz/LSB
Nominal frequency (f_o)	250 MHz	50 KHz
DCO clock frequency (f_{DCOclk})		62.5 MHz

Since the DCO in prototype has the same number of frequency steps as the one in ASIC has, and the center code of the binary control code range corresponds to the nominal frequency of DCO, the value of N_c could be calculated from Eq.(2).

In this way, the FPGA emulator is designed as a propotionally scaled down prototype of the ASIC system, and it could operate with the same filter coeffients as those designed for ASIC.

III. EXPERIMENTAL RESULTS

A. System stability and performance

A FPGA emulator for a network with 16 clock generators is implemented on ALTERA CYCLONE II EP2C70 platform. Tests are done to compare the stability and performance of system with different filter coefficients. The phase error between the first and second clock generators in the network is observed. Fig. 6 presents the results of three tests with different groups of coefficients. The upper plot demonstrates a system with good performance. The maximum error is only ± 2 units when it converges, where one unit corresponds to the resolution of PFD ($\Delta T = 149.88ns$). The middle plot shows a system which converges rapidly, while with worse performance when it is stable. The maximum error is about ± 4 units. With some other coefficient values, the frequencies oscillate wildly and the system is no longer stable, like shown in the lower plot. With the help of this FPGA prototype, designers could choose the coefficients according to the specification of system and observe the results easily.

B. Prevention of undesired stable states

Due to cyclic (modular) nature of phase and a large number of degrees of freedom in the complex system, the system could have more than one stable states. In some of the states, all oscillators have equal frequency, but may have fixed non-zero phase difference compared with their neighbors. The stable state that the system will enter despends on the initial condition, thus is not controllable. Therefore, a configuration mechanism is necessary to garantee the actual setteled state of the system is that at which all the oscillators are synchronized both in frequency and in phase.

Fig. 7 is a figure captured by the oscillator showing the clock generated by one node ($node11$) and those generated

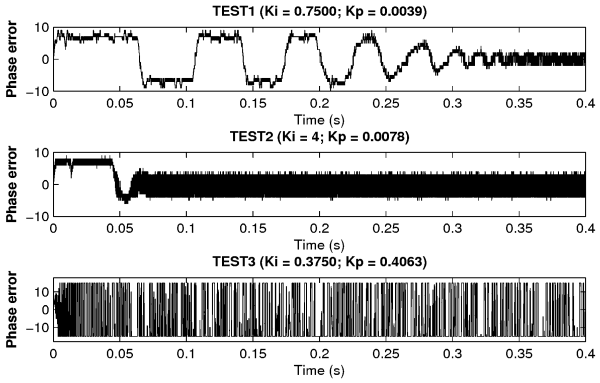


Fig. 6. Phase error between node 1 and node 2

by its 4 neighbors ($node7, 10, 12, 15$). In this case, when the network is stable, the rising edge of $clock11$ is about 1.8 us ahead of those of $clock7$ and $clock10$ but about 1.8 us late compared to edges of $clock12$ and $clock15$. 1.8 us is 12 times the resolution of PFD, which is a big value, while due to a nealy zero average value (one-fourth of $Total_Err$ presented by the lowest plot in Fig. 7), the frequency of $clock11$ keeps unchanged, and similar phenomenon happens in other nodes. The network falls into an undesired stable state.

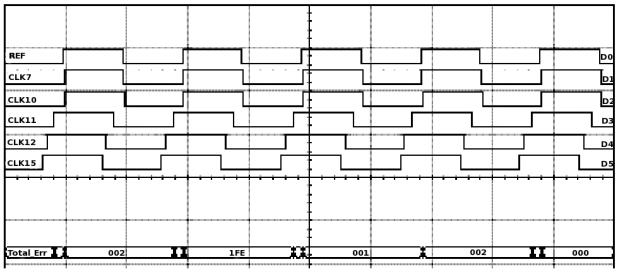


Fig. 7. clock of $node11$ and its 4 neighbors (configuration: bidirectional)

It is known that these undisired modes can be eliminated if the network is unidirectional [1], which means each node receives the phase error information only from upper and left neighbors. Thus, the information is transmitted in one direction from the upper-left corner of the network to the lower-right corner. However, it has a drawback: Just like a traditional clock tree, any perturbation appearing in early nodes will propagate through the entire network. The clock in the area far from the reference clock will have a relatively poor quality compared with clocks near the reference. This drawback could be critical and could degrade the system performance if the network has a high order.

A solution has been presented in [2]. The network is configured dynamically and works in two phases. In the first phase, the network is unidirectional configured in order to avoid undesirable stable states. When the phase errors are corrected and all the nodes are synchronized in phase, the network is reconfigured quickly as bidirectional mode. All the links are activated: This is the second phase. In this phase, each clock is coupled with its four neighboring clocks so that perturbations will be suppressed. The switching of the connectivity is implemented by programming dynamically the coefficients K_{w_i} in the loop filters presented in Fig. 5. A link can be activated or deactivated simply by assigning 1 or 0 to the corresponding coefficient K_{w_i}.

Fig. 8 shows the four phase errors after weight coefficients (K_{w_i}) multiplication between node 11 and its neighboring nodes. During the period $0ms \sim 30ms$, the network is unidirectional configured, only the phase differences between the node and its left ($node10$) and upper node ($node7$) are considered. After $30ms$, the network switches to bidirectional mode, all the four errors are taken into consideration.

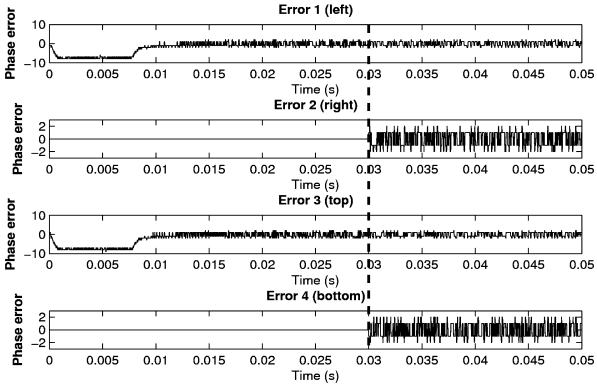


Fig. 8. phase errors between $node11$ and its neighboring nodes after weight coefficients multiplication

IV. CONCLUSION

A reconfigurable and adaptive ADPLL network is developed and implemented on FPGA. The network for distributed clock generation solved jitter and metastability problems of traditional synchronization approaches. A solution for modelocks is proposed. The feasibility and performance of architecture is verified by experimental results. The FPGA emulator could be used as a effective tool in future multi-core SoC design.

V. ACKNOWLEDGMENT

This work has been funded by the French National Agency of Research (ANR) under grant ANR-07-ARFU-05.

REFERENCES

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

[2] 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

[3] E. Zianbetov et al., *Design and VHDL modeling of all-digital PLLs*, 8^{th} IEEE international NEWCAS conf., 2010, Montreal, QC, pp. 293-296

[4] 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.

[5] P. M. Levine et al., *A high-resolutino flash time-to-digital converter and calibration, proceeding of International Test Conference*, pp. 1148-1157, 2004.