INTEGRATION the VLSI journal 58 (2017) 142–154

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

A standard cell phase locked loop design, analysis and high-level synthesis tool (CellPLL)

Yalcin Balcioglu*, Gunhan Dundar

Bogazici University, Department of Electrical and Electronics Engineering, BETA Laboratory, Bebek, Istanbul, Turkey

ARTICLE INFO

Keywords:
All-digital phase locked loop
High-level synthesis
Phase domain model
Phase noise estimation

ABSTRACT

In this paper, we present a new approach that provides a complete design, analysis, and high-level synthesis (HLS) flow for all-digital phase locked loops (ADPLL). CellPLL uses a methodology for direct design of transfer functions given a set of specifications by the user. In order to analyze the estimated phase noise of each design, a flexible phase domain model implementation of ADPLL is incorporated. For automatic design implementation, a new HLS engine with a library parser and ADPLL realization template is used. The flow is applied for four different cases and the results match circuit level simulation results. CellPLL successfully generates ADPLL designs and provides ability to move between production processes.

1. Introduction

Phase locked loops (PLLs) are being extensively used in today's wired and wireless communication products as part of data recovery circuits, clock multipliers, and frequency synthesizers. With the increasingly tougher specifications of today's communication circuits, there is a constant push for developing small and low power PLLs while satisfying strict frequency spectrum specifications [1].

During PLL design, another driving factor is the design cycle time which does not allow the re-design and complete analysis of PLLs in many cases. Hence, it is crucial to develop a framework in which designs are modeled, analyzed for phase noise, and implemented within a guided flow. This allows minor modifications to be implemented rapidly with reduced risk, design transfer between technology nodes, and rapid design space exploration to identify what is possible within a technology.

As a subset of PLLs, the fractional-N All Digital PLLs (ADPLLs) in Fig. 1 are specifically suitable for such a framework because of their standard cell architecture. Standard cell architectures allow reuse, provide flexibility, and ease scaling with technology migration. It is seen that ADPLLs lack a flexible phase model implementation with various phase noise contributors such as:

1. Inherent oscillator noise
2. Digitally controlled oscillator (DCO) quantization noise
3. Quantization noise from digital phase detection
4. Quantization noise from the $\Sigma\Delta$ modulator

Additionally, a methodology for generating the loop transfer functions for given specifications and support of HLS are required for ADPLL design flow.

Similar to many other circuit types, PLLs have traditionally been implemented using analog circuits. However, area hungry circuit components such as the capacitors in the loop filter and poor g_{ds} of transistors pose potential problems against larger scale integration in finer process nodes [1]. This creates the need for digitization of PLLs with cheaper digital resources. Previous work [2–4] on the digitization of PLLs has led to:

- Phase detection, implemented digitally by a TDC
- Voltage-controlled oscillator, replaced by a DCO
- Loop filter,in the digital domain instead of the analog

However for digital PLLs, the digitization has been limited to some of the components in the design, and approaches utilizing only standard cells have just recently been published [5,6].

In [7], a general phase model has been proposed for PLLs; however, it is targeted for analog PLL architectures. This paper targets phase noise of the sub-components of a general fractional-N ADPLL architecture. While the theory behind individual noise components is known, this extended phase noise model brings the components together and presents a flexible implementation for various ADPLL topologies in an effort to reduce the need for time costly transient simulations. Similarly, in [8], an automatic loop design method has been presented, but this method concentrates on analog PLLs while an improved loop generation method for ADPLLs is illustrated in this work.

* Corresponding author.

http://dx.doi.org/10.1016/j.vlsi.2017.04.001
Received 4 February 2016; Received in revised form 23 March 2017; Accepted 5 April 2017
Available online 08 April 2017

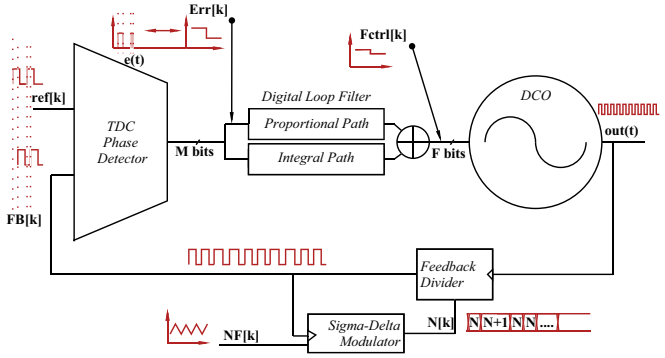

Fig. 1. Fractional-N all-digital PLL.

There are tools for HLS of digital circuit types other than ADPLLs [9,10]. However, CellPLL adds a HLS engine for the first time to the design flow for ADPLLs. HLS gets the loop design parameters from transfer function generator and generates Verilog code together with synthesis scripts for the register transfer level (RTL) synthesizer. During HLS, CellPLL uses the internally embedded ADPLL template [11]; thus, this limits the set of specification space within template's boundaries. However, as the loop design generator and the phase model are independent of the template, any standard cell ADPLL implementation can be embedded in CellPLL in order to support a different subset of possible ADPLL specifications.

This work implements a general phase model and loop generation algorithm for ADPLLs together with a HLS engine. Sections 2 and 3 explain architecture, phase noise model, and HLS support for Digitally Controlled Oscillator (DCO) and Time-to-digital Converter (TDC), respectively. Section 4 details closed and open loop transfer function generation, loop design, phase noise model, and synthesis support for the top-level of the ADPLL. In Section 5, various use cases with different specifications and process nodes are generated. Moreover, correlation results with circuit level simulations and other tools used in the industry are reported. Finally, we draw conclusions in Section 6.

2. Digitally-controlled oscillator (DCO)

A DCO is an oscillator type where the frequency tuning control is done with a digital control word F_{ctrl}. While a DCO internally does not have to be digital for a digitally assisted PLL (DPLL), this work uses the architecture of the previously designed standard cell DCO [12,13] in order to be able to demonstrate HLS, modeling, and generate a standard cell ADPLL.

2.1. Architecture

The DCO in Fig. 2 incorporates N ring oscillators. This architecture is selected in order to provide a standard cell based flexible design with a large frequency tuning range of 0.65 – 1.35 GHz at the expense of larger power and area consumption due to the multi ring architecture. Every ring oscillator uses the same number of three, five or seven programmable delay cells rather than basic inverters for creating an oscillator loop. An offline calibration algorithm is deployed for use before the ADPLL starts using the DCO. During calibration, an externally provided clock source is used to measure the free running oscillation frequency for five delay cells in a ring while using the center frequency control word. If the oscillator frequency is slow due to process, voltage or temperature (PVT), the delay cell count in the rings are reduced to three. Similarly, if the oscillation frequency is initially too fast, the rings are programmed to use seven delay cells. Depending on the calibration result, unwanted delay cells are bypassed using multiplexers and the desired number of delay cells is connected to create a ring.

Each ring has tri-state buffers at each delay element output and all of the rings are connected in parallel at the output of delay elements. Each ring has a unique and one bit drive enable signal that enables all of the delay elements. The nodes driven by multiple drivers create the main time constant for each delay stage as the capacitance from every active or inactive ring's driver and next stage input is summed. Frequency tuning is achieved by changing the effective resistance at each high time constant node by enabling more or less rings while the capacitance is the same. Tri-stated rings work as capacitive load; otherwise, when their drivers are active they increase the driving strength, thereby increasing the output frequency of the loop by decreasing the time constant at the output node of every delay element. By adjusting how many of the rings are active, coarse frequency tuning is obtained. Furthermore, each delay cell has a unique fine frequency control ($FCW[3:0]$) that allows the delay of each delay cell to be adjusted in fine steps. There are seven FCW signals connected to each delay element in a ring and this signal is shared in all rings. Except "0000", all FCW values can be used to provide slightly tuned delay variations using the inherent propagation delay difference between the inputs of the gates. LSB bits of the linear F_{ctrl} binary vector are mapped to the non-linear FCW signals of delay cells for each delay cell in the ring separately. Combining coarse and fine frequency control mechanisms provides the tuning control for the DCO. [12,13].

2.2. Phase noise modeling

The DCO has been modeled in the frequency domain as given in Fig. 3a. Bit vector frequency control input F_{ctrl} goes through gain blocks with a gain of $K_v[Hz/bits]$ and 2π. The resulting signal is the instantaneous frequency of oscillation in rad/sec which is subsequently integrated to get the continuous time phase of the output clock in rad. The DCO open loop noise is modeled using a random number generator (RNG) with a normal distribution to generate a white noise spectrum in the frequency domain. This instantaneous frequency noise is integrated as in Eq. (1) with the reference clock period T_s as the sampling time. The resulting phase noise seen in Fig. 3b is added to the output phase of the DCO.

$$\Phi_{DCO_{out}} = DCO_{PN} + \sum 2\pi K_v T_s F_{ctrl_{in}}[k] \tag{1}$$

This noise power spectral density (PSD) rolls off with $-20\ dB/dec$ and its magnitude at zero offset frequency is determined by the variance of the RNG. With Eq. (2) provided in [14], the magnitude of the phase noise at the specified frequency by the user is calculated. F_{osc} is the center frequency where V_{DD} is the supply voltage, k is the Boltzmann constant, M is the number of delay stages, and C is the load capacitance.

$$f_{osc} = \frac{I}{CMV_{DD}}$$
$$L(f)_w = \frac{2kT}{I}\left(\frac{\gamma_N + \gamma_P}{V_{DD} - V_t} + \frac{1}{V_{DD}}\right)\frac{f_{osc}^2}{f_{offset}^2} \tag{2}$$

In order to use this equation, some parameters are extracted from the synthesis libraries and some are calculated. The DCO's internal phase noise parameters such as supply voltage V_{DD}, temperature T, threshold voltage V_t, and noise factor γ_N γ_P are used from the library. However, the oscillation frequency f_{osc}, pull-up and pull-down current I, and the stage load capacitance value C which changes with the drive strength of the cells in the rings needs to be estimated. After HLS is run for the DCO, the strengths of the cells, load capacitance per stage C, and the oscillation frequency are identified with the help of static timing analysis. [14] assumes that the pull-up and pull-down currents are the same. With similar assumption, the pull current I is calculated by dividing the total charge $V_{DD}C$ by the average propagation delay t_p. Having all of the parameters of Eq. (2) allows the phase noise ($DCO_{PN}@f_{offset}$) at the desired offset frequency f_{offset} to be calculated. The desired maximum phase