Published in IET Circuits, Devices & Systems
Received on 14th May 2013
Revised on 5th September 2013
Accepted on 6th September 2013
doi: 10.1049/iet-cds.2013.0175

ISSN 1751-858X

Development of low-complexity all-digital frequency locked loop as 500 MHz reference clock generator for field-programmable gate array

Sigit Yuwono[1], Seok-Kyun Han[2], Giwan Yoon[2], Han-Jin Cho[3], Sang-Gug Lee[2]

[1]*Department of Information and Communication Engineering, KAIST, Daejeon, Republic of Korea*
[2]*Department of Electrical Engineering, KAIST, Daejeon, Republic of Korea*
[3]*SW-SoC Convergence Research Laboratory, ETRI, Daejeon, Republic of Korea*
E-mail: s. yuwono@kaist.ac.kr

Abstract: The authors report the development of an on-chip 500 MHz reference clock generator as a part of a clock manager for a field-programmable gate array. The generator is implemented in the form of an all-digital frequency locked loop (ADFLL) in architecture of low complexity and high modularity. For the development of the ADFLL, they propose a new circuit that employs two under-sampled 1-bit ΔΣ frequency-to-digital converters to convert a frequency difference into a proportional distributed pulsewidth. By the combination of the proposed circuit with a conventional phase-and-frequency detector, a frequency comparator is implemented and can indicate its two input frequency conditions, that is, (i) equal to, (ii) lower than or (iii) higher than. The ADFLL which adopts the proposed frequency comparator is implemented in a 90 nm CMOS technology. Consuming 2.64 mW from a 1.2 V supply, the ADFLL shows about 50 μs of locking time at the frequency accuracy of 99.2% while operating at 500 MHz and being driven by a 10 MHz reference clock.

1 Introduction

This paper reports the development of an on-chip 500 MHz reference clock generator as a part of a clock manager for a field-programmable gate array (FPGA). To obtain such a high reference frequency, the clock generator functions as a frequency multiplier and is developed based on a coherent indirect frequency synthesiser [1]. In this work, both the FPGA and the clock manager developed are targeted to be implemented in a fully digital CMOS technology where no components such as inductors, metal-insulatar-metal (MIM) capacitors and varactors are available. Therefore the clock generator as a part of the clock manager has to be implemented in the form of either the all-digital phase-locked loop (ADPLL) or all-digital frequency-locked loop (ADFLL); in this paper, both ADPLL and ADFLL are termed all-digital frequency synthesiser (ADFS).

One of the approaches in developing an ADFS is using a direct digitalisation in which the analogue parts of a phase-locked loop (PLL) or a frequency-locked loop (FLL) are replaced by their digital equivalents. Although there have been a lot of computer-aided design tools for designing an ADPLL or an ADFLL based on the direct digitalisation approach, this approach inevitably involves relatively complex arithmetic operations [2].

In this work, the ADFS is developed mainly by adopting a binary search algorithm (BSA) [3], which is considered as a simpler approach. In this approach, the fundamental function of the ADFS can be described with the flowchart shown in Fig. 1*a*. In the figure, the ADFS consists basically of two functional blocks, that is, a frequency comparator (FC) and a control code generator (CCG). The FC is to detect whether the digitally controlled oscillator (DCO) frequency is equal to or different from the target frequency, and the CCG is to generate the control codes to the DCO so that the corresponding frequency change can be made. In addition to those main blocks, a decision making process is added at the bottom of the flowchart to meet the requirement that a loop in a flowchart may not continue endlessly; in the implementation, this block is represented with a power or reset switch.

In the flowchart presented in Fig. 1*a*, the control code for the DCO is assumed to be directly proportional to the DCO frequency so that the locking process of the ADFS can be described as follows: when the DCO frequency is the same as the target frequency, the control code will stay unchanged; when the DCO frequency is lower (higher) than the target frequency, the control code to the DCO will be increased (decreased) so as to bring the DCO frequency to the target frequency. In Fig. 1*a*, to distinguish the two decision-making subblocks in the FC, the subblock that detects whether both frequencies are the same is called a frequency estimator (FE), and the other subblock that detects whether a frequency is lower or higher than the other one is called a frequency detector (FD).

Although the algorithm represented by the flowchart in Fig. 1*a* is fundamental, almost all practical ADFSs are implemented based on a slightly different algorithm whose

flowchart is shown in Fig. 1*b*. In the latter figure, the FC is now supplemented with a functional block that will first process the frequency or phase error before the FC concludes whether the DCO frequency is equal to, lower than or higher than the target frequency. In an ADPLL developed based on direct digitalisation of a PLL, the FC is implemented basically by employing a phase-and-frequency detector (PFD), a time-to-digital converter and a digital filter [4]. An ADFLL developed based on a direct digitalisation of an FLL [5] works almost by the same principle and the FC is implemented basically with a 1-bit ΔΣ frequency-to-digital converter (ΔΣFDC), a frequency-error comparator and a digital filter.

The ADFSs in [2, 3, 6] are developed based on the BSA approach. The FE of the ADFLL in [3] is implemented with a control logic circuit that contains both adding and subtracting circuits for measuring the magnitude of the phase and frequency differences. In [6], the FD of the ADPLL is implemented with a conventional PFD [1, 4], whereas its FE is implemented with a finite state machine (FSM) circuit that processes further the output of the FD to generate the indication of a locked state. The ADPLL in [2] is probably the ADFS that can implement an FE in the simplest way. However, to enable such a simple FE, its FD has to be implemented in a rather complex way than the FE. Moreover, the FC (FD and FE) in [2] must be accompanied by a high frequency-resolution DCO, for example, LC tank-based DCO, to operate as designed.

Based on the above study, we can deduce that in developing an ADFS, the FD, that is, the circuit to detect whether one frequency is lower or higher than the other one, can actually be implemented with quite a simple circuit. Therefore the additional process in the FC of the flowchart shown in Fig. 1*b* is strongly related to the complexity occurring in the implementation of the FE rather than the FD. Therefore, we conclude that if the FE, that is, the circuit to detect a frequency similarity, of an ADFS can be implemented in a very simple way, then the whole architecture of the ADFS will also be very simple.

In this paper, we propose a new and simple FE that enables the implementation of a new ADFS in architecture of low complexity and high modularity. The resulting ADFS architecture strongly resembles the fundamental flowchart shown in Fig. 1*a*. Since the flowchart in Fig. 1*a* processes only frequency but not phase, the resulting ADFS will eventually be a type of ADFLL. Moreover, from the application standpoint, this ADFLL will be fully functional as a 500 MHz reference clock generator for an FPGA.

The rest of this paper is organised as follows. Section 2 presents the development of the proposed FE as well as the new ADFLL for a DCO with fine frequency-resolution. Section 3 describes the further development of the ADFLL to accommodate a DCO with coarser frequency-resolution. Section 4 presents the measurement results of the implemented ADFLL. Finally, Section 5 concludes this paper.

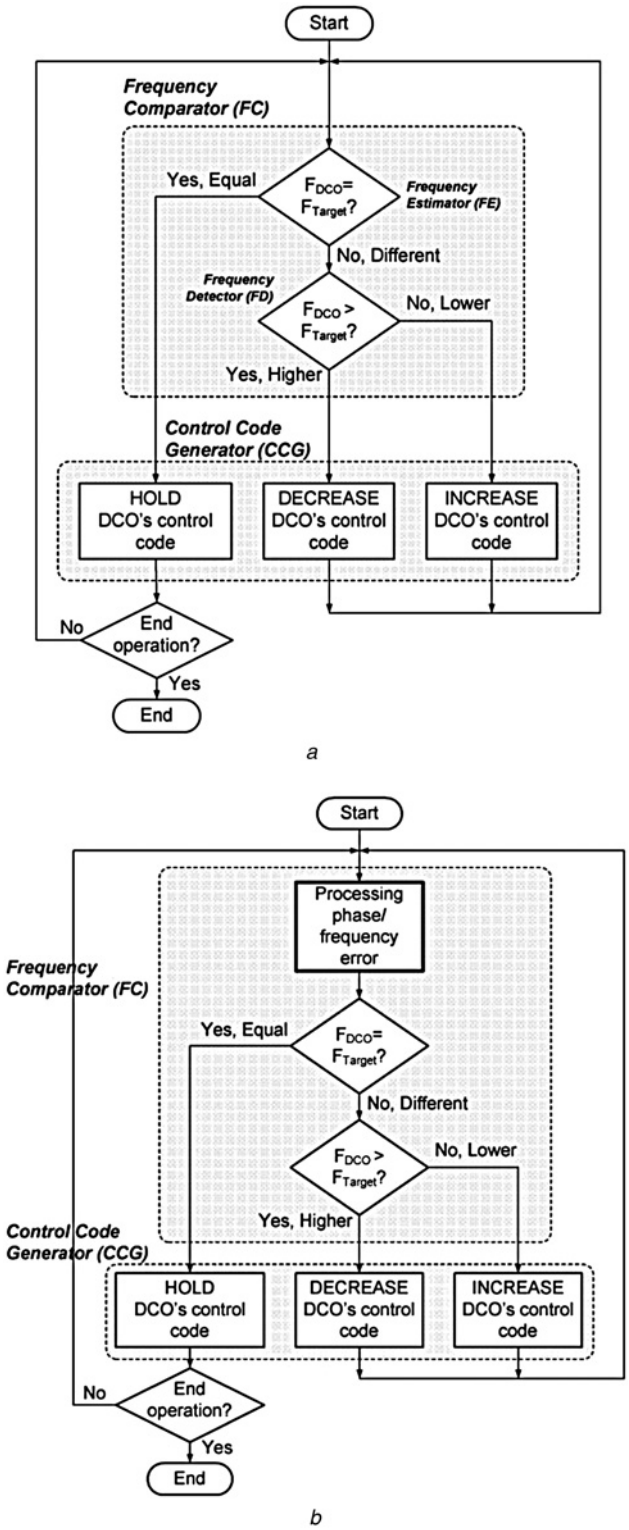


Fig. 1 *Flowcharts describing the working principles of ADFSs*

a Fundamental flowchart of an ADFS
b Flowchart of practical ADFSs

2 Development of the proposed ADFLL

Fig. 2*a* shows the architecture of the proposed ADFLL which is a straightforward translation of the flowchart shown in Fig. 1*a*. In Fig. 2*a*, F_{REF} is the external reference clock frequency which represents the target frequency and F_{FB} is the feedback frequency which basically represents the DCO frequency or the output frequency F_{OUT}, which is programmable. A divider DIV is added into the diagram because the ADFLL will function as a frequency multiplier. The number of the CCG's output bits depends largely on the requirements of the DCO; in this work, it is nine. In the following subsections, we present the circuits adopted to implement the proposed ADFLL, along with the systematic steps in designing the FE suitable for the proposed ADFLL. The design of the FE is the main work in this study.

doi: 10.1049/iet-cds.2013.0175

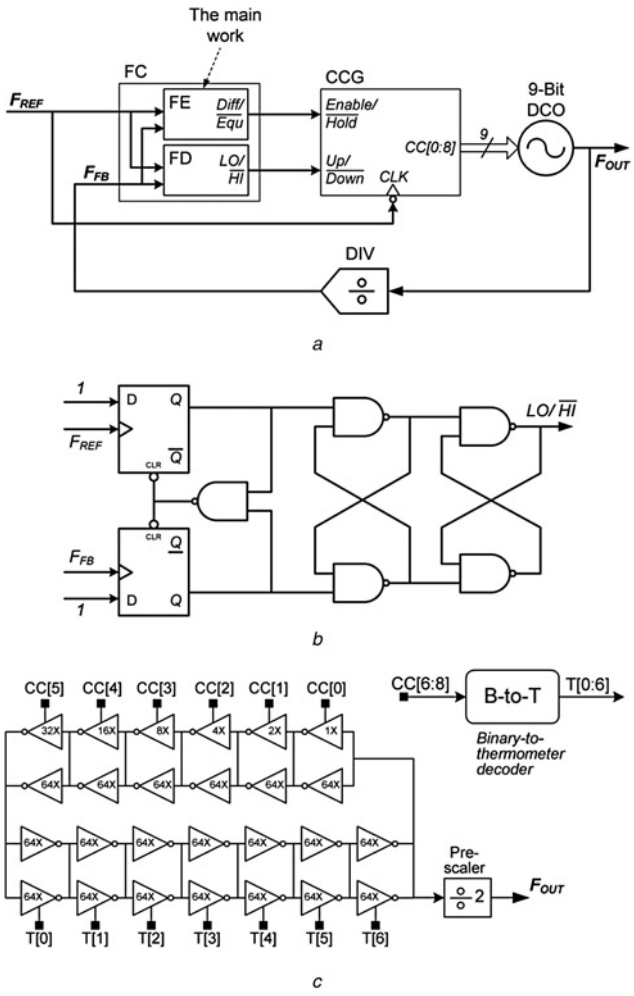


Fig. 2 *Schematic diagram of*

a Proposed ADFLL and the main work (arrow-indicated) in this study
b Implemented FD
c Implemented DCO

2.1 Control code generator

According to the schematic shown in Fig. 2*a*, the 9-bit CCG should be implemented with a circuit that can increase, decrease and hold its output binary code. For this purpose, the CCG is implemented with a conventional JK flip-flop up-down counter (UDC) [7]. The adopted UDC is falling-edge sensitive which changes its output at the falling edges of F_{REF}; this is indicated by the small circle attached to Port CLK of the CCG shown in Fig. 2*a*.

2.2 Frequency detector

In the proposed ADFLL, the FD detects whether the frequency (not phase) of F_{FB} is lower or higher than that of F_{REF}. The circuit to perform this function is adopted from [8] and its diagram is shown in Fig. 2*b*. When the frequency of F_{FB} is lower (higher) than that of F_{REF}, the output will be 'true (false)' or 'logic 1 (0)'.

2.3 Divider (DIV)

For the proposed ADFLL the divider in [9], which is a clock swallowing multi-modulus divider (MMD), is adopted since it gives a 50% duty-cycled output. In this work, the implemented divider is designed to provide integer feedback ratios ranging from 16 to 63.

2.4 Digitally controlled oscillator

Owing to the limitation of the process library and chip area, the DCO construction is decided to be based on a CMOS digital inverter adopted from [10, 11]. The schematic diagram of the implemented DCO is shown in Fig. 2*c*. The DCO totally requires nine bits of binary code for its control code, which are denoted as CC[0:8] in the diagram. Six bits, CC[0:5], directly control the DCO for fine frequency tuning. The remaining three bits, CC[6:8], are first converted, with a binary-to-thermometer decoder, into seven bits of thermometer code, which are denoted as T[0:6] in the diagram. The resulting bits T[0:6] are then used to control the DCO for coarse frequency tuning. The resulting DCO has output frequency directly proportional to the control code.

A true single-phase clock (TSPC) prescaler is added to improve both the frequency resolution and the duty cycle. To accomplish the target specifications of the DCO, that is, a DCO with a centre frequency of 500 MHz and a frequency resolution of 1 MHz, the sizes of the transistors of the inverters are manually adjusted one by one.

2.5 Development of the FE

As emphasised in the introduction, there has not been a simple FE that is suitable for the proposed ADFLL. In the absence of the FE, the final output frequency of the proposed ADFLL will be determined by both the FD and F_{REF} which control the CCG. However, the speed of the FD in detecting which input frequency is lower or higher is much lower than F_{REF}; this speed difference creates an exaggerated time span in which the CCG changes the control code (thus, also F_{OUT}) in the opposite direction to where the control code should be directed. As a result, the ADFLL without the FE will not be able to lock the target frequency.

By considering the architecture of the proposed ADFLL shown in Fig. 2*a*, the above unlocking problem should be solved by the FE, whose output controls Port Enable/$\overline{\text{Hold}}$ of the CCG, by decreasing the effective rate of the CCG in generating the control code as the frequency difference decreases. On combining this requirement with the main function of the FE described by the flowchart in Fig. 1*a*, the required FE should exhibit the following two behaviours: (i) the FE should generate an output whose pulse (logic 1) width decreases as its input frequencies are going closer to each other, and (ii) the FE's output should be completely logic 0 to totally disable the CCG when the FE's input frequencies are exactly the same.

In this work, the required FE is developed by employing a frequency subtracting circuit (FSC) [12] whose output is connected to a clocked level-change detecting circuit so that the pulse or logic 1 only occurs around the rising and falling edges of the FSC output. Since the number of edges is directly proportional to the frequency, the output pulse (logic 1) width of the level-change detecting circuit is also proportional to the frequency difference at the input of the FSC.

In the ADFLL, F_{FB} as the sensed frequency can be either much higher or lower than the reference frequency F_{REF}. To enable the FE to work properly in both conditions, two identical circuits, each consisting of the FSC and the level-change detecting circuit, are implemented as the FE.

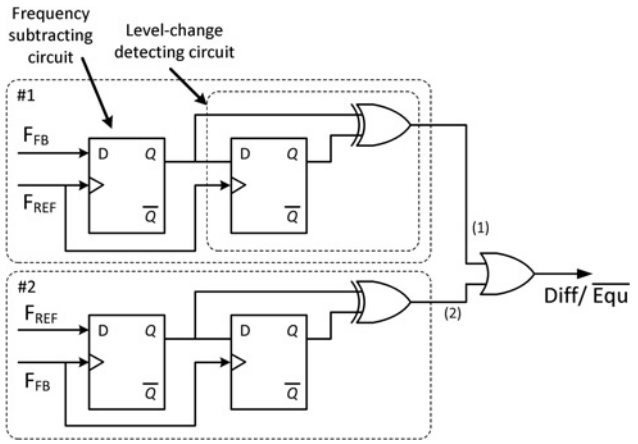


Fig. 3 *Proposed FE that consists of the frequency subtracting circuit and the clocked level-change detecting circuit*

In the first circuit, F_{REF} samples F_{FB}, whereas in the second circuit, F_{REF} is sampled by F_{FB}. Fig. 3 shows the schematic diagram of the proposed FE.

The constituting circuit of the FE shown in Fig. 3 does not look uncommon since it is well known as a ΔΣ modulator [13] or a ΔΣFDC [5, 14]. The differences are only in the applications. The input frequencies of the ΔΣFDC in [5, 13, 14] are largely different since one of them functions as a sampling clock to digitise the other, whereas in the proposed FE, the ΔΣFDC circuit works in the under-sampling mode. Moreover, the output signal Diff/$\overline{Equ}$ of the proposed FE is simply treated as a 1-bit digital signal, whereas the output of the conventional ΔΣFDC is usually treated as serial data that need to be further processed or interpreted.

Fig. 4*a* shows the comparative simulation results of six FEs for the six input frequency differences of the same simulation time while using the same F_{REF}. The decreasing frequency difference is to simulate the condition in the ADFLL when F_{FB} approaches F_{REF} as happened in the locking process. The result proves that when the absolute frequency difference decreases, occurrences of the output pulse (logic 1) are becoming rarer, which means that the total pulsewidth over a certain duration is proportional to the frequency difference. Moreover, based on the property of the FSC [12], we can deduce that when both input frequencies become exactly the same, the output of the FE will be completely equal to logic 0, which conforms to the required behaviour of the FE.

Another important property of the proposed FE is that when its input frequencies (F_{REF} and F_{FB}) are becoming very close to each other, every single pulse (logic 1) in Diff/$\overline{Equ}$ will be almost always as wide as one period of F_{REF}, as partly indicated in the lower row of Fig. 4*b*. Thus, only one falling edge of F_{REF} will occur when the state of Diff/$\overline{Equ}$ is logic 1 or the CCG is enabled. As a result, in this condition, the CCG can increase or decrease its output only as much as one LSB. This condition is expected with high probability in the steady state because the CCG will be able to tune the DCO with a frequency step as fine as the DCO's frequency resolution.

Figs. 5*a* and *b* plot simulation results showing the behaviours of the proposed ADFLL after adopting the proposed FE in locking the target frequency. The data plotted in Fig. 5*a* are the control codes observed at the CCG output, CC[0:8]. It indicates that the proposed FE can effectively help the ADFLL to lock the target frequency precisely.

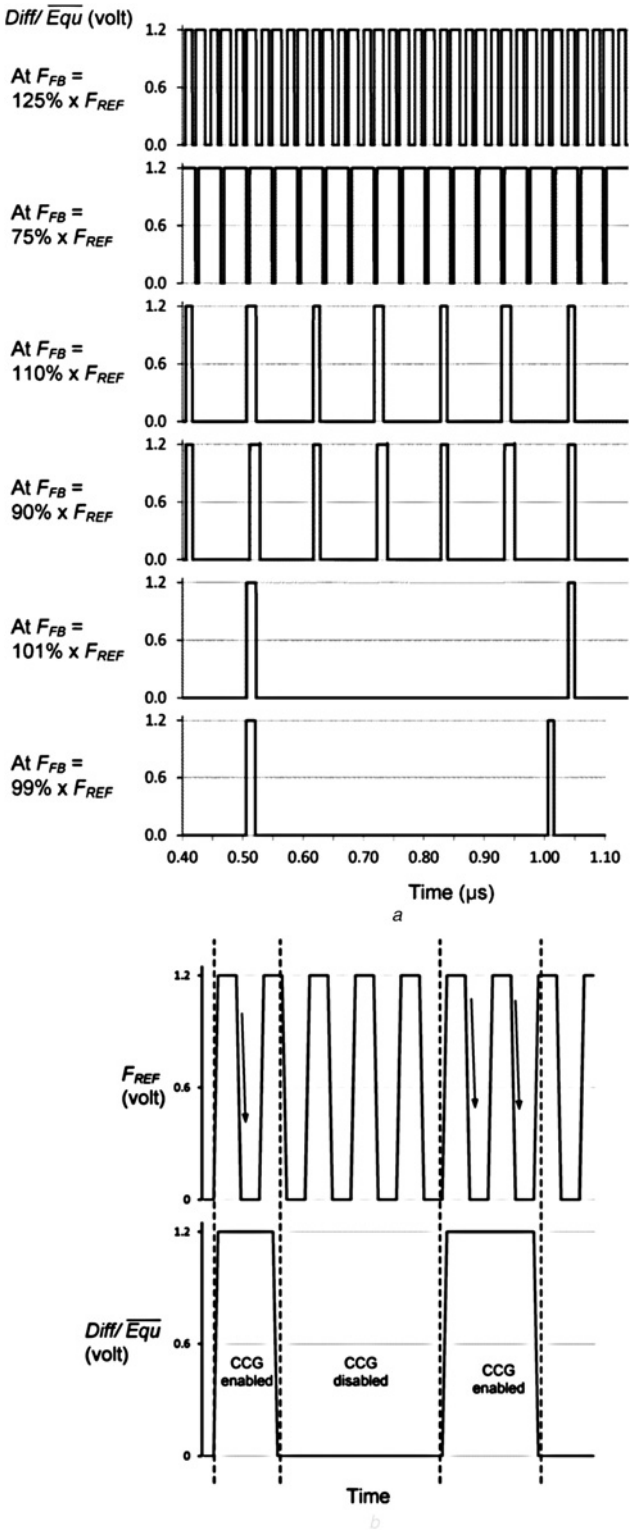


Fig. 4 *Simulation results on the proposed FE*

a Various waveforms at the output of the FE, Diff/$\overline{Equ}$, with decreasing input frequency differences

b Time-aligned waveforms of F_{REF} and Diff/$\overline{Equ}$ at $F_{FB} = 0.9 \times F_{REF}$

3 Development of an intermittently tracking ADFLL

In Fig. 5*a*, we can still observe the alterations of the control code in the steady state. According to the previous discussion, regarding the simulation result of the FE plotted in Fig. 4*b*, the control code changes only between two

doi: 10.1049/iet-cds.2013.0175

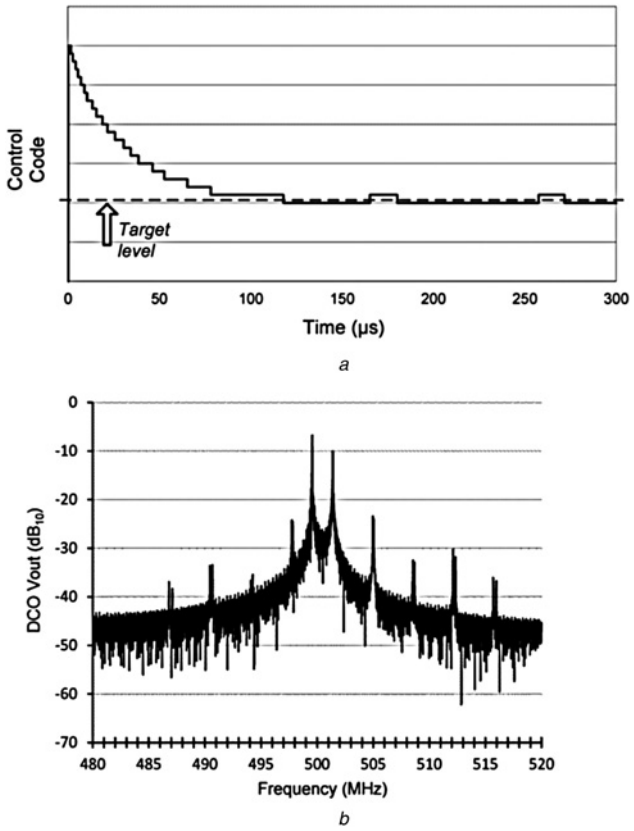


Fig. 5 *Simulation results of the ADFLL that shows*

a Effectiveness of the proposed FE in locking the target frequency
b F_{OUT}'s frequency spectrum along with two main frequency peaks in the steady state

values whose difference is only one LSB. This means that in the steady state, the output frequency of the ADFLL, F_{OUT}, moves back and forth between two frequencies that are slightly higher and lower than the target frequency, respectively, and the difference of these two frequencies is equal to the DCO's frequency resolution.

If the implemented DCO has a very fine frequency resolution, for example, similar to an LC tank-based DCO, then the difference of those two frequencies would be very small. However, in this work, the implemented DCO shows only a frequency resolution of about 1 MHz at the working frequency of 500 MHz. The frequency jump because of this relatively poor frequency resolution, as indicated in Fig. 5*b*, will manifest as considerable deterministic jitters and may deteriorate the performance of the ADFLL. To handle such a problem, we decide to adopt a mechanism that opens the loop of the ADFLL during the steady state and intermittently closes it when the output frequency needs to be corrected [15].

To adopt this mechanism, the initially-proposed ADFLL needs to be supplemented with the following two circuits, that is, one circuit to open the ADFLL's loop when the ADFLL is in the steady state, which is called a loop-opening circuit (LOC), and the other circuit to re-close the ADFLL's loop when the ADFLL's output frequency needs to be corrected, which is called a loop-closing circuit (LCC).

3.1 Loop-opening circuit

According to the previous discussion, when the loop of the ADFLL is opened during the steady state, the ADFLL frequency will be locked to a frequency whose distance to the target frequency is not more than the frequency resolution of the DCO. In this design, the opening of the loop is controlled by the LOC and the key information used to develop the LOC is the output of the FD that changes its state whenever the polarity of its input frequency difference is inverted, that is, from the states of '$F_{FB} > F_{REF}$' to '$F_{FB} < F_{REF}$' or vice versa. The polarity inversion means that the DCO frequency just crossed the target frequency and indicates that the ADFLL has entered the steady state.

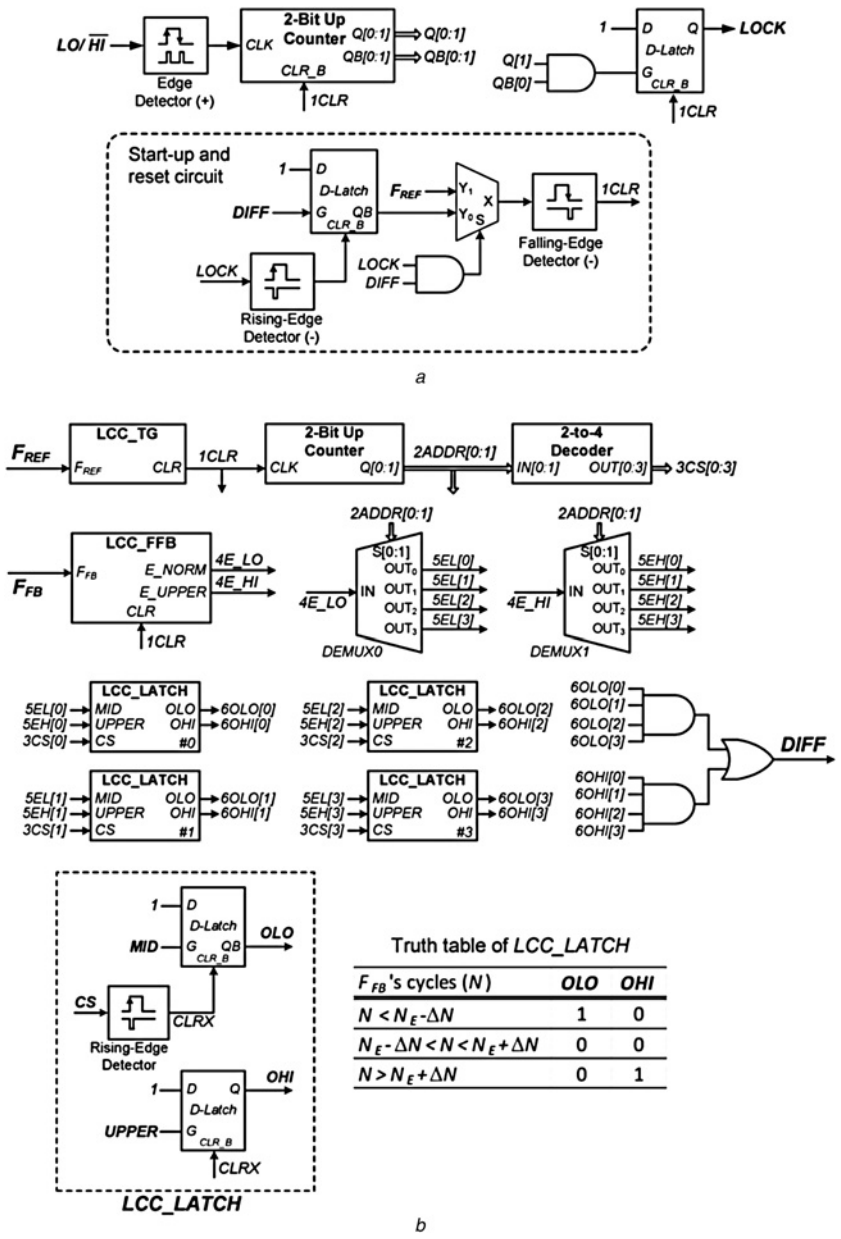
The schematic diagram of the LOC is shown in Fig. 6*a*, and it works as follows: at the beginning of the locking process, the up-counter output is reset to zero; the counter then counts the occurrence of the state change of the FD output, LO/$\overline{\text{HI}}$; when the number of the occurrences reaches two, the output signal of the LOC, LOCK, will go to logic 1 and disable the CCG so that the output control code of the CCG will stop changing and the loop is considered as being open.

3.2 Loop-closing circuit

The LCC function is to re-close the loop by resetting the LOC whenever the output frequency of the ADFLL, F_{OUT}, needs correction. Port DIFF of the LOC as shown in Fig. 6*a* is dedicated for this purpose. The first step in designing the LCC is defining the range of the frequency deviation, called Err_{Frq}, in which F_{OUT} may still be considered in the locked state. The value of Err_{Frq}, should be higher than the relative frequency resolution of the DCO, that is, the ratio between the DCO's frequency resolution and the working frequency so that, in this design, Err_{Frq} should be higher than 1 MHz/500 MHz or 0.2%. If Err_{Frq} is set lower than this value, whenever the LOC opens the ADFLL's loop, the LCC will immediately generate the reset signal DIFF to the LOC and the loop will be closed again. In this design, Err_{Frq} is set to around 0.8% in order to provide some error margin because of the performance of the implemented DCO which is based on a digital inverter and usually has more coarse frequency resolution and larger jitter compared with those of an LC tank-based DCO or a CMOS ring-based DCO [5, 16].

To implement the value of Err_{Frq} into the LCC, we first define the valid duration in which the frequency deviation is measured. This duration is called the gating time T_G and its length is equal to a certain number of periods of F_{REF} defined as N_G. Later, the value of Err_{Frq} is converted into a fraction which is equal to $\Delta N/N_E$; where N_E is the expected number of the periods of F_{FB} that occurs in one T_G, and $N_E + \Delta N$ is less than N_G. Afterwards, in the operation, the LCC will count the number of periods of F_{FB}, simply called N, in one T_G. As long as N is between $N_E + \Delta N$, the loop is kept open or there is no generation of the reset signal DIFF to the LOC.

Fig. 6*b* shows the complete schematic diagram of the LCC. In the figure, the LCC_TG is the circuit to define the duration of T_G. Later, the number of F_{FB}'s cycles in one T_G is counted by the LLC_FFB that contains the predefined values of $N_E - \Delta N$ and $N_E + \Delta N$. To make the LCC more immune to the generation of false outputs and also to provide a level of certainty to its outputs, the LCC evaluates four consecutive T_Gs before generating the reset signal DIFF. The evaluation results of the four T_Gs are stored in four LCC_LATCHs, whose individual diagram is shown in the inset of Fig. 6*b*. The truth table of the LCC_LATCH attached to the figure describes the states of the LCC_LATCH output, that is, OLO and OHI, as a function of the number of F_{FB}'s cycles



F_{FB}'s cycles (N)	OLO	OHI
$N < N_E - \Delta N$	1	0
$N_E - \Delta N < N < N_E + \Delta N$	0	0
$N > N_E + \Delta N$	0	1

Fig. 6 *Schematic diagram of*

a LOC that both opens the loop and locks the final output frequency
b LCC that re-closes the loop for correcting F_{OUT}

in one T_G. The final output of the LCC is annotated as DIFF; its state is logic 0 when the frequency deviation of F_{OUT} is still in the tolerable range, and its state becomes logic 1 when the deviation is beyond the boundaries. The rising edge of this DIFF will reset the LOC through its input port DIFF.

3.3 Final architecture of the implemented ADFLL

Fig. 7*a* shows the final architecture of the implemented ADFLL after adopting the LOC and LCC. Fig. 7*b* plots the simulation result to verify the functionalities of the LOC and LCC. In the simulation, the frequency deviation at F_{OUT} is emulated by alternating the feedback ratio, which is represented with the state of the MMD's $P[0]$ plotted in the upper row of Fig. 7*b*. The alternations of the feedback ratio cause F_{FB} to exhibit frequency changes. Later, these frequency changes are detected by the LCC and subsequently trigger the LOC to re-close the loop for correcting F_{OUT}; this sequence of events is confirmed with the regions marked with (i) in Fig. 7*b*. After both the frequencies of F_{OUT} are updated and the ADFLL reaches the new steady state, the LOC disables the CCG so that the control code stays unchanged in the steady state; the effectiveness of the LOC is confirmed with the regions marked with (ii) in Fig. 7*b*.

4 Experimental verifications

The proposed ADFLL has been implemented as a part of an FPGA's clock manager in a 90 nm CMOS technology, consuming an effective area of 200 μm × 100 μm as shown in Fig. 8. The figure also shows the micrograph of the clock manager (lower left) and the test board (lower right) on which the clock manager chip is attached. The clock manager consists of the ADFLL, a delay locked loop (DLL) and a phase shifter. The whole test board of the clock

doi: 10.1049/iet-cds.2013.0175

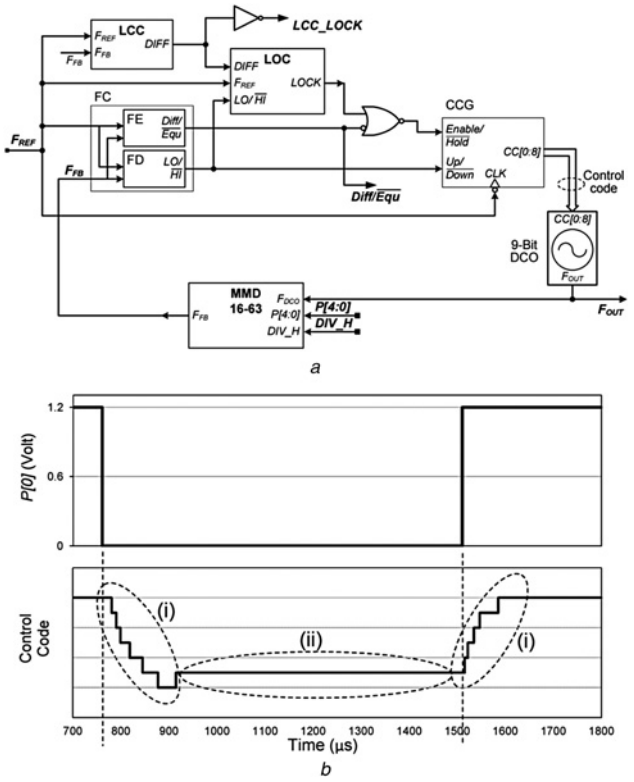


Fig. 7 *Implemented ADFLL*

a Final architecture
b Simulation result to verify the functionalities of the LCC and LOC

manager operated from a single 1.2 V supply with a quiescent current consumption of 2.2 mA at the working frequency of 500 MHz.

The first specification we verified in the measurement was the operating frequency of the implemented DCO; it ranged approximately from 496 to 623 MHz which covered the main target frequency of 500 MHz. Fig. 9*a* shows the frequency spectrum of F_{OUT} when the ADFLL is locked at the target frequency of 500 MHz. Fig. 9*a* also shows that the reference spur is around −66 dBc which is relatively decent [17]. Ideally, in the locked stated in which the loop of the ADFLL is open, the DCO should generate a spurless output. The occurrence of the spurs is very likely because of the use of single power supply for the whole clock manager without separating the power supply for the DCO. In the next experiment, the quality of F_{OUT} in the time domain was also verified. A screen capture of this measurement result is presented in Fig. 9*b*, showing that the rms jitter is around 25 ps, which is a relatively common value for a DCO based on a CMOS digital inverter [11, 18].

The locking time of the ADFLL was examined through the following experiment. F_{REF} was set to 10 MHz and the feedback ratio was initially set to 51. This setup results in the target frequency of 510 MHz. Later, the feedback ratio was repeatedly alternated between 51 and 50. This alternation made F_{OUT} move back and forth between 510 and 500 MHz or as far as one F_{REF}. Although the feedback ratio was being alternated, the locking time was observed through Port LCC_LOCK of the ADFLL which is the inverted output of the LCC as depicted in Fig. 7*a*.

A screen capture of this experimental result is presented in Fig. 10; the upper row shows the waveform of the signal injected into the MMD's *P*[0] to change the feedback ratio, whereas the lower row shows the waveform of LCC_LOCK. As indicated in the figure, the change in the feedback ratio triggers the LCC to momentarily change the state of LCC_LOCK. The duration of this momentary change is equal to the locking time of the ADFLL and the

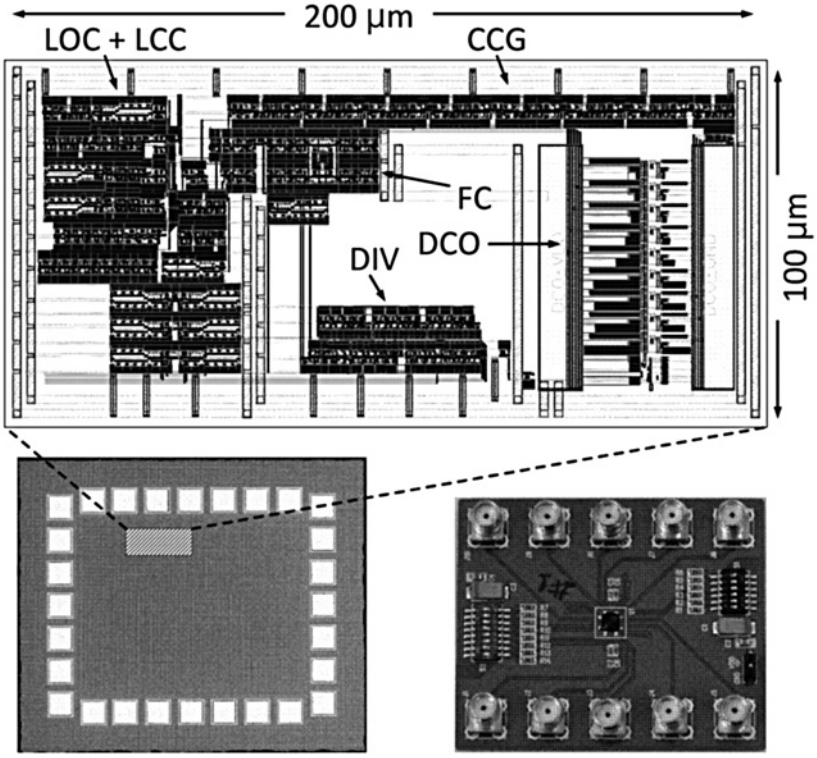


Fig. 8 *Layout of the ADFLL, the micrograph of the clock manager (lower left), and the test board (lower right) for measurement; the shaded area on the micrograph is the effective area of the ADFLL*

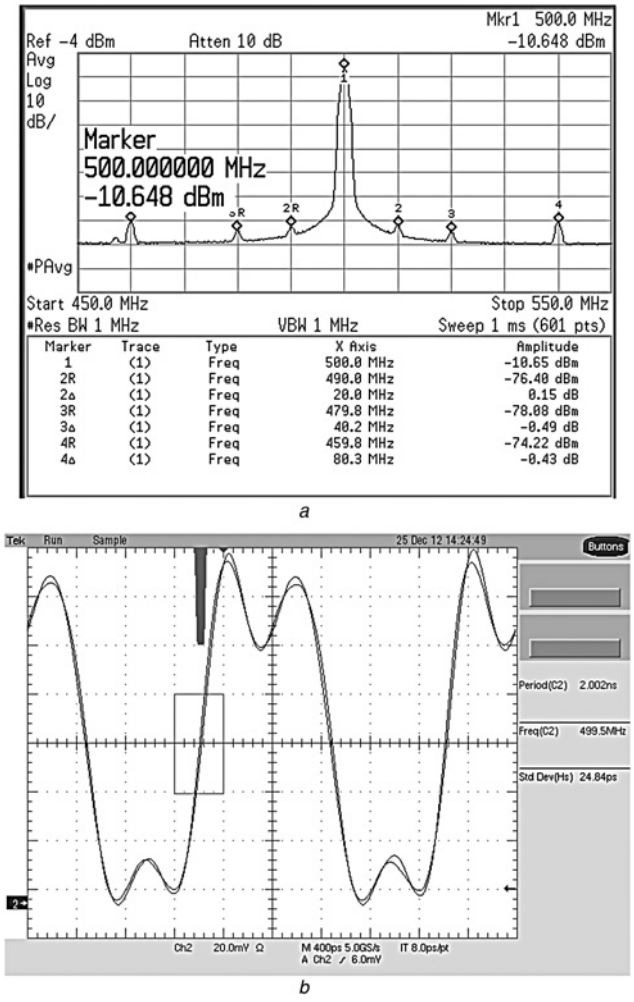


Fig. 9 *Measurement of the ADFLL output F_{OUT} at the working frequency of 500 MHz*

a Frequency spectrum and the reference spurs
b Time-domain waveform and the jitters

measured value was around 50 µs. The monotonous logic 1 in LCC_LOCK indicates that F_{OUT} is already in the range of 99.2% of the target frequency, as specified in the design of the LCC.

In the measurements, the effectiveness of the LOC and LCC were only indirectly observed. The fact that the ADFLL can lock the target frequencies as shown in Figs. 9*a*, *b* and Fig. 10 proves that the LOC worked properly. Fig. 10 also shows that the ADFLL can correct its output frequency which implies that the LCC also worked properly.

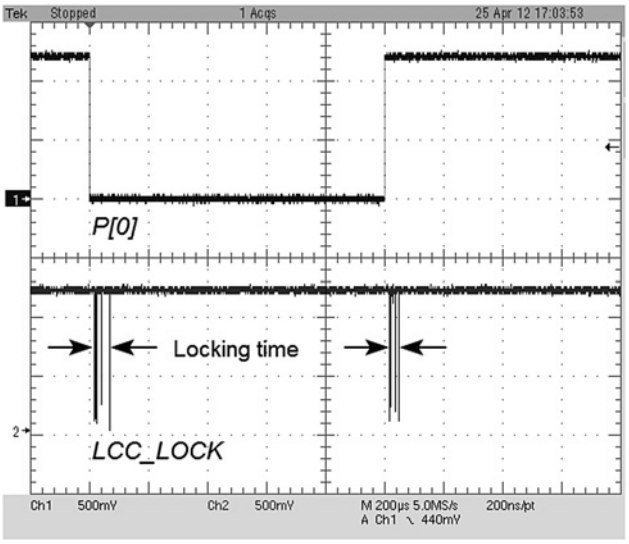


Fig. 10 *Locking time observed through LCC_LOCK (lower row) whereas the feedback ratio alternation is represented by P[0] (upper row)*

Table 1 presents the performance summary of the implemented ADFLL and four published ADFSs for comparison purpose. From the table, it can be found that the settling time of the proposed ADFLL is considerably short, especially when it is compared with the other ADFLL [5]. Our ADFLL's performance can also be considered superior in terms of the area and reference jitters. Moreover, we can also see that although the rms jitter of the implemented ADFLL is quite high, it is still on par with those of the other ADFSs whose DCOs are based on a CMOS digital inverter [11, 18]. On the other hand, the power consumption of the proposed ADFLL is found to be little more than twice that of the ADFLL in [5] for the same output frequency. The reason is that the implemented DCO, as shown in Fig. 2*c*, actually works at twice the frequency of F_{OUT} or at 1 GHz, which consequently degrades significantly the score of the figure of merit (FOM) [18]. Besides, the measured power is consumed not only by the ADFLL but also by the whole clock manager, which also contains both a DLL and a phase shifting circuit.

Table 1 Performance summary and comparison

Parameter	[5]	[11]	[16]	[18]	This work
type	fract.-N ADFLL	Int.-N ADPLL	Int.-N ADPLL	Int.-N ADPLL	Int.-N ADFLL
CMOS process	0.18 µm	65 nm	0.18 µm	0.18 µm	90 nm
size, mm^2	0.113	0.03	0.47	0.32	0.02
operating freq., MHz	400–410	300–5000	900–1250	33–1040	495–625
ref. freq., MHz	11–26	125 at 500 MHz	60	—	10–34
voltage, V	1.8	0.4 at 500 MHz	1.8	1.8	1.2
power, mW	1.26 at 405 MHz	1.34 at 500 MHz	12.24 at 960 MHz	15.7 at 1.04 GHz	2.64 at 500 MHz
DCO type	RO	digital inverter RO	RO	digital inverter RO	digital inverter RO
settling time, µs	110	—	—	—	50
RMS-jitter, ps	—	16.4 at 500 MHz	5.03 at 960 MHz	13.8 at 950 MHz	25
ref. Spurs, dBc	– 55	—	—	—	– 66
FOM	0.44	4.98	0.44	7.03	0.22

FOM = F_{MAX}[GHz] × ((F_{MAX})/(F_{MIN})) ÷ (power[mW] ÷ (VDD^2 × F_{Power}[GHz])) [18]

doi: 10.1049/iet-cds.2013.0175

5 Conclusion

In this work, we report a novel ADFLL as a 500 MHz reference clock generator for an FPGA. The key circuit in the ADFLL is a frequency-difference to pulsewidth converter (the FE) which consists of two 1-bit ΔΣFDCs that are designated to work in the under-sampling mode. The adoption of the proposed FE has enabled the development of an ADFLL that has low complexity and high modularity, consumes small chip area and exhibits short locking time. Moreover, the adoption of the intermittently-tracking mode has probably contributed in significantly suppressing the reference spurs.

6 Acknowledgment

This work was supported by IDEC, the IT R&D program of MKE, Republic of Korea [KI002168, Development of Configurable Device & SW Environment], the Ministry of Education and Science Technology (MEST) and Korea Institute for Advancement of Technology (KIAT) through the Human Resource Training Project for Regional Innovation.

Sigit Yuwono would like to thank Joo-Myoung Kim particularly for his schematic and layout designs of the implemented DCO.

7 References

1 Manassewitsch, V.: 'Frequency synthesizers: theory and design' (Wiley-Interscience, New York, 1980, 2nd edn.), pp. 22–31
2 Zhuang, J., Du, Q., Kwasniewski, T.: 'A 4 GHz low complexity ADPLL-based frequency synthesizer in 90 nm CMOS'. Proc. IEEE CICC, San Jose, CA, USA, September 2007, pp. 543–546
3 Gothandaraman, A., Islam, S.K.: 'An all-digital frequency locked loop (ADFLL) with a pulse output direct digital frequency synthesizer (DDFS) and an adaptive phase estimator'. Proc. RFIC Symp., Philadelphia, PA, USA, June 2003, pp. 303–306
4 Kratyuk, V., Hanumolu, P., Moon, U., Mayaram, K.: 'All-digital phase-locked loops based on a charge-pump phase-locked-loop analogy', *IEEE Trans. Circuits Syst. II, Exp. Briefs*, 2007, **54**, (3), pp. 247–251
5 Khalil, W., Shashidharan, S., Copani, T., *et al.*: 'A 700-μA 405-MHz all-digital fractional-N frequency- locked loop for ISM band applications', *IEEE Trans. Microw. Theory Tech.*, 2011, **59**, (5), pp. 1319–1326
6 Wang, C.-C., Huang, C.-C., Tseng, S.-L.: 'A low-power ADPLL using feedback DCO quarterly disabled in time domain', *Microelectron. J.*, 2008, **39**, (5), pp. 832–840
7 TAMS: 'Synchronous up/down counter (JK flipflops)', http://tams-www.informatik.uni-hamburg.de/applets/hades/webdemos/30-counters/40-updown/updown.html, accessed August 2012
8 Okada, T., Endo, A.: 'Digital frequency comparator circuit'. U.S. Patent 3 987 365, 19 October 1976
9 Yang, Y.-C., Yu, S.-A., Wang, T., Lu, S.-S.: 'A dual-mode truly modular programmable fractional divider based on a 1/1.5 divider cell', *IEEE Microw. Wirel. Compon. Lett.*, 2005, **15**, (11), pp. 754–756
10 Olsson, T., Nilsson, P.: 'A fully integrated standard-cell digital PLL', *IEE Electron. Lett.*, 2001, **37**, pp. 211–212
11 Tierno, J., Rylyakov, A., Friedman, D.: 'A wide power supply range, wide tuning range, all static CMOS all digital PLL in 65 nm SOI', *IEEE J. Solid-State Circuits*, 2008, **43**, (1), pp. 42–51
12 Javeri, R.J., Grove, E.: 'Frequency substractor'. U.S. Patent 4 683 437, 28 July 1987
13 Hovin, M., Olsen, A., Lande, T.S., Toumazou, C.: 'Delta-sigma modulators using frequency modulated intermediate values', *IEEE J. Solid-State Circuits*, 1997, **32**, pp. 13–22
14 Hovin, M., Saether, T., Wisland, D.T., Lande, T.S.: 'A narrow-band delta-sigma frequency-to-digital converter'. Proc. IEEE ISCAS, Hong Kong, June 1997, vol. 1, pp. 77–80
15 Mesgarzadeh, B., Alvandpour, A.: 'A low-power digital DLL-based clock generator in open-loop mode', *IEEE J. Solid-State Circuits*, 2009, **44**, (7), pp. 1907–1913
16 Lee, S., Seo, Y., Park, H., Sim, J.: 'A 1 GHz ADPLL with a 1.25 ps minimum-resolution sub-exponent TDC in 0.18 μm CMOS', *IEEE J. Solid-State Circuits*, 2010, **45**, (12), pp. 2874–2882
17 Banerjee, D.: 'PLL performance, simulation, and design' (Dog Ear Publishing, Indianapolis, 2006, 4th edn.), pp. 64–72
18 Choi, K.-H., Shin, J.-B., Sim, J.-Y., Park, H.-J.: 'An interpolating digitally controlled oscillator for a wide-range all-digital PLL', *IEEE Trans. Circuits Syst. I, Reg. Pap.*, 2009, **56**, (9), pp. 2055–2063