# An All-Digital Phase-Locked Loop Compiler with Liberty Timing Files

Ching-Che Chung[1], Duo Sheng[2], and Chen-Han Chen [1]
[1] Department of Computer Science and Information Engineering, National Chung Cheng University,
No. 168 University Rd., Min-Hsiung, Chia-Yi, Taiwan
[2] Department of Electrical Engineering, Fu Jen Catholic University,
No. 510 Chung-Cheng Rd. Hsin-Chung, Taipei, Taiwan.
Email: wildwolf@cs.ccu.edu.tw

***Abstract* — In this paper, an all-digital phase-locked loop (ADPLL) compiler with liberty timing files (.lib) is presented. The proposed digitally controlled oscillator (DCO) frequency range estimation algorithm can accurately compute the frequency range of the DCO with only liberty timing files. Therefore, the proposed ADPLL compiler can generate a wide frequency range and monotonic response DCO circuit according to the user input specifications. The generated DCO circuit is designed with standard cells. Thus, the design turnaround time for the ADPLL can be greatly reduced. The proposed ADPLL compiler is verified with SPICE circuit simulator. The maximum frequency estimation error is smaller than 5.92% in 90nm or 65nm CMOS processes.**

## I. INTRODUCTION

Phase-locked loop (PLLs) and delay-locked loops (DLLs) are widely used in a system-on-a-chip (SoC). PLL with a loop filter have better reference jitter tolerance than the DLL-based frequency synthesizer, and thus, PLLs are usually used for on-chip high-speed clock generation applications. Traditionally, PLLs are usually designed by the analog charge-pump based architecture. However, analog PLLs need to be redesigned when the process technology is changed. Moreover, the leakage current of transistors in advanced CMOS process makes it difficult to design a voltage controlled loop at a low supply voltage.

In contrast to analog PLLs, all-digital phase-locked loops (ADPLLs) [3] -[12] use the digital design approaches and does not use any passive component which allows it to be easily integrated with other digital circuits into the systems in advanced CMOS process. In a SoC, it often requires several PLLs for different I/O interface. In order to reduce the design time and design efforts when processes or specifications are changed, ADPLLs implemented with standard cells can have best portability and are more suitable for the SoC design than analog PLLs.

Among the functional blocks of the ADPLL, DCO is the most critical component. Because the DCO usually occupies the most portions of the ADPLL's area and consumes the relative large power consumption than the other blocks of the ADPLL. Furthermore, DCO dominates the major performance of the ADPLL, such as the output frequency range, and output jitter. According to different design requirements for realizing a DCO for various applications, such as spread-spectrum clock generator (SSCG), fast settling ADPLL, an automatic design flow for the ADPLL is demanded.

A parameterized ADPLL compiler [6] is proposed to support easy process migration. In [6], the DCO is transformed into the equivalent model, and then the DCO timing analyzer will use the timing information provided by liberty timing files (.lib) to compute the frequency range of the DCO. After that, the ADPLL compiler [6] can generate a suitable DCO circuit to meet the frequency range requirements. However, due to the non-linearity of the proposed DCO architecture, there has a large frequency estimation error between the estimation results calculated by the ADPLL compiler [6] and the SPICE circuit simulation results. As a result, for different process, many SPICE circuit simulations should be performed for calibration before the ADPLL compiler [6] can be used.

The non-linear delay model used in the liberty timing files (.lib) will use four neighboring points which simulated by SPICE circuit simulator to compute the delay time of the logic gate with the specified input transition time and the specified output loading. However, the DCO architecture [6] has a high input transition time and a large output loading during looking up the liberty timing files (.lib). As a result, there has a large frequency error in the DCO timing analyzer [6].

In this paper, the proposed ADPLL compiler uses a linear DCO architecture to avoid the problems mentioned above. Thus we can obtain more accurate timing information as compared to the SPICE circuit simulation results. The proposed ADPLL compiler also uses cell timing information provided by the liberty timing files (.lib) to calculate the period information of the DCO. As a result, the ADPLL compiler does not need to perform many SPICE circuit simulations before porting to a new CMOS process which makes the proposed ADPLL compiler more suitable for the design automation of the ADPLL.

The rest of the paper is organized as follows: Section II describes the proposed DCO architecture and the frequency range estimation algorithm of the proposed ADPLL compiler. Experimental results and circuit implementation results are discussed in Section III. Finally, Section IV concludes with a summary.

## II. PROPOSED ADPLL COMPILER

### A. PROPOSED DCO ARCHITECTURE

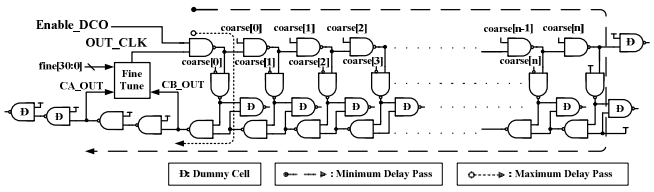


FIGURE 1. THE PROPOSED DCO ARCHITECTURE.

Fig.1 shows the proposed DCO architecture. The DCO is composed of a coarse-tuning stage [1] and a fine-tuning stage [2],[3]. The coarse-tuning stage consists of n coarse-tuning delay cells (CDCs). Thus the coarse-tuning resolution of the proposed DCO is two NAND gate delay time. In addition, unused CDCs can be gated for reducing the power consumption of the DCO at high frequency operation.

Fig. 2 shows the fine-tuning stage of the DCO. The fine-tuning stage [2],[3] is composed of two parallel connected tri-state buffer arrays operating as an interpolator circuit that are controlled by the fine-tuning control code (Fine[31:0]). The total delay controllable

This work was supported in part by the National Science Council of Taiwan, under Grant NSC102-2221-E-194-063-MY3.

978-1-4799-2776-0/14/$31.00 ©2014 IEEE

range of the fine-tuning stage is always equal to one coarse-tuning resolution. When more left-hand side tri-state buffers are turned on, the output clock is more close to the CA_OUT. Oppositely, when more right-hand side tri-state buffers are turned on, the output clock is more close to the CB_OUT. As a result, the proposed DCO will have a monotonic response in output clock period versus the DCO control code which can reduce the jitter of the output clock during coarse-tuning control code switching.

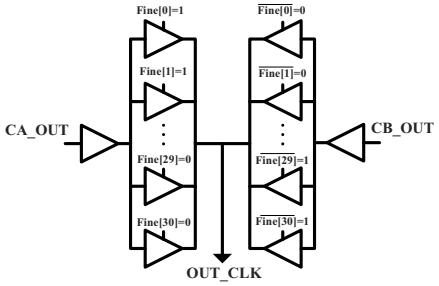


FIGURE 2. THE PROPOSED FINE-TUNING STAGE.

In the proposed DCO, the coarse-tuning stage has a regular structure with a good linearity in output period versus the coarse-tuning control code. Therefore, it can achieve a wide frequency range by increasing the number of the coarse-tuning delay cells. The proposed flexible DCO architecture is suitable for the ADPLL compiler to generate a DCO circuit with user input specifications.

## B. FREQUENCY RANGE ESTIMATION

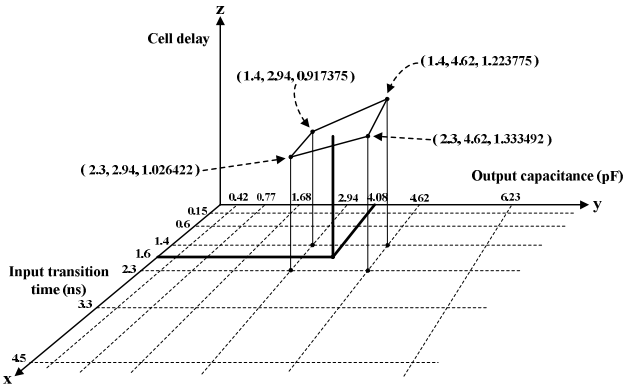


FIGURE 3. THE NON-LINEAR DELAY MODEL IN LIBERTY TIMING FILE.

The proposed ADPLL compiler uses the cell timing information in the liberty timing files (.lib) to estimate the period of the DCO. To compute the period of the DCO, the rise delay and fall delay of each delay cell in the DCO are calculated and accumulated. Hence, we need to look up the liberty timing files (.lib) of the delay cells.

In each cell timing model, there are four delays can be looked up from the liberty timing files (.lib). There are cell rise transition time, cell fall transition time, cell rise delay time, and cell fall delay time. To use the non-linear cell delay model, the specified input transition time and output loading are needed to compute these four delays of a delay cell. Thus for one delay cell, the rise transition time and fall transition time of the previous delay cell, and the total loading capacitance followed by the delay cell will be the input transition time and the output capacitance to lookup the cell timing model.

$$Z = A + Bx + Cy + Dxy \quad (1)$$

Fig. 3 illustrates how to use the non-linear delay model to compute the cell delay time. If the input transition time and the output capacitance are 1.4 ns and 2.94 pF, respectively, the cell delay can be directly looked up from the timing model and will be 0.917375ns. However, if the input transition time and the output capacitance are 1.6 ns and 4.08 pF, respectively, the cell delay can only be computed by the neighboring four points which simulated by the SPICE circuit simulation in cell library characterization. Eq. 1 can be used to compute the cell delay. In Eq. 1, Z is the cell delay time, x is the input transition time, and y is the output capacitance, and A, B, C, D are coefficients needed to be solved. The delay time values in the neighboring four points can be substituted into the Eq. 1 to obtain the coefficients of A, B, C, and D. As a result, for the specified input transition time and output loading, the delay time of the delay cell can be computed from the cell timing model. Thus, the total delay of the coarse-tuning stage can be computed by the above mentioned approach.

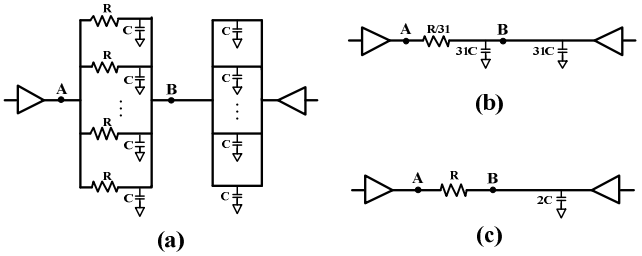


FIGURE 4. EQUIVALENT MODEL OF THE FINE-TUNING STAGE.

In the fine-tuning stage of the proposed DCO, there are parallel connected tri-state buffers. In [4], [5], a delay model for parallel connected tri-state buffer arrays is proposed. The delay model [4], [5] shows that the delay time of the parallel connected tri-state buffer array is related by the number of conductive tri-state buffers. However, the linear model [4], [5] is not accurate in advanced CMOS process. A transformed equivalence model [6] is proposed for parallel connected tri-state buffer arrays. In [6], the parallel connected tri-state buffer arrays are transformed into an equivalent RC model to represent the changes of the driving strength and the capacitance loading. Then, the driving strength of the driving cell is normalized to R, and we can use the equivalent C value to look-up the timing information from the cell timing model.

According to [6], Fig. 4 shows the equivalent model for computing the maximum delay of the fine-tuning stage. Fig. 4(a) shows the proposed fine-tuning stage is transformed into resistors and capacitors. Fig. 4(b) shows the combination values of transformed RC circuit. The final normalization result shown in Fig. 4(c). As a result, we can use two times the output capacitance of one tri-state buffer to look up the cell delay of the fine-tuning delay stage.

## C. AUTOMATIC DCO GENERATION FLOW

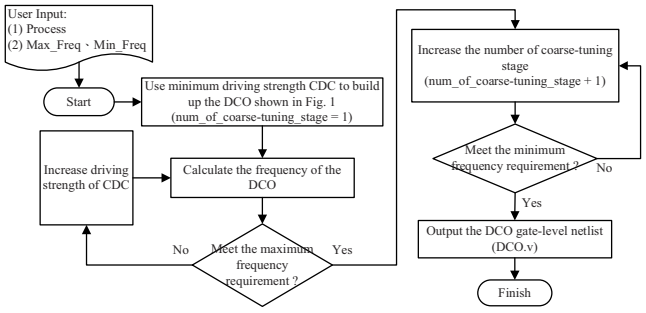


FIGURE 5. FLOWCHART OF AUTOMATIC DCO GENERATION.

Fig. 5 shows the flowchart of automatic DCO circuit generation. After the user inputs the specifications of the ADPLL, such as process, and the frequency range (i.e. maximum output frequency and minimum output frequency), the ADPLL uses the proposed frequency range estimation algorithm to compute the range of the DCO. In the beginning, the DCO is created by minimum driving strength CDCs with one coarse-tuning delay stage. Then, the maximum output frequency of the DCO can be computed by the proposed frequency range estimation algorithm from the cell timing library. If the current DCO configuration does not meet the maximum output frequency requirement, we will increase the driving strength of the CDCs until the maximum output frequency requirement is satisfied. Then, the number of the coarse-tuning stage is increased until the minimum output frequency requirement is met. Finally, the DCO gate-level netlist in hardware description language (HDL) format is output for ADPLL integration.

The other modules of the ADPLL except for the phase and frequency detector (PFD) are designed with register transfer level (RTL) HDL codes. Thus the gate-level netlist of these modules can be generated after logic synthesis. By using the proposed ADPLL compiler, the design turnaround time of the ADPLL can be greatly reduced.

## III. Experimental Results

TABLE I. FREQUENCY RANGE ESTIMATION ERROR IN 90NM CMOS

| | | 90nm CMOS process | | |
|---|---|---|---|---|
| | | Estimation | HSPICE (MHz) | Error (%)* |
| CLKNAND2X2 | FF | 176.80 ~ 3209.99 | 177.37 ~ 3041.17 | 0.32 ~ -5.26 |
| | TT | 117.07 ~ 2092.14 | 121.35 ~ 2049.34 | 3.7 ~ -2.05 |
| | SS | 70.94 ~ 1260.25 | 74.97 ~ 1257.91 | 5.54 ~ -0.19 |
| CLKNAND2X4 | FF | 176.59 ~ 3315.35 | 171.92 ~ 3128.97 | -2.64 ~ -5.62 |
| | TT | 117.25 ~ 2160.17 | 117.67 ~ 2104.46 | 0.36 ~ -2.58 |
| | SS | 71.49 ~ 1303.08 | 72.72 ~ 1290.20 | 1.72 ~ -0.99 |
| CLKNAND2X8 | FF | 178.70 ~ 3355.70 | 173.27 ~ 3160.33 | -3.04 ~ -5.82 |
| | TT | 118.79 ~ 2186.80 | 118.20 ~ 2117.80 | -0.5 ~ -3.16 |
| | SS | 72.93 ~ 1322.58 | 73.13 ~ 1299.90 | 0.27 ~ -1.71 |
| CLKNAND2X12 | FF | 176.42 ~ 3335.53 | 172.90 ~ 3145.40 | -2 ~ -5.7 |
| | TT | 116.92 ~ 2170.96 | 118.05 ~ 2110.70 | 0.97 ~ -2.78 |
| | SS | 71.41 ~ 1310.73 | 72.97 ~ 1289.80 | 2.19 ~ -1.6 |

* Error (%) = ($Freq_{HSIPCE}$ - $Freq_{Estimation}$) / $Freq_{HSPICE}$ * 100%

To verify the accuracy of the proposed frequency range estimation algorithm, the ADPLL compiler is verified with standard cells in 0.18µm, 90nm, and 65nm CMOS processes. Table I shows the estimation frequency range in 90nm CMOS process as compared with HSPICE simulation results under process, voltage, and temperature (PVT) variations. The DCO in the Table I has 63 coarse-tuning stages. As shown in Table I, the maximum frequency error as compared to the HSPICE simulation results with different CDCs is 5.82%.

Table II shows the frequency range estimation error in 0.18µm, 90nm, and 65nm CMOS processes. The maximum frequency estimation error in 90nm and 65nm CMOS process is 5.82% and 5.92%, respectively. However, the maximum frequency estimation error in 0.18um is 13.95%. In 0.18µm CMOS process, the input capacitance of standard cells is much larger than in 90nm and 65nm CMOS process. Therefore, the spacing of the neighboring four points in the non-linear table is also larger than 90nm and 65nm CMOS process in typical liberty timing files (.lib). As a result, there has a larger error in the delay time calculation. Since the proposed ADPLL compiler depends on the accuracy of the liberty timing files (.lib), and thus, the accuracy of the frequency range estimation can be improved by re-characterize some delay cells of the 0.18µm cell-library.

TABLE II. ESTIMATION ERROR IN DIFFERENT PROCESSES

| | | Frequency Range Error (%) | | |
|---|---|---|---|---|
| | | 65nm | 90nm | 0.18µm |
| NAND2XL | FF | -1.8 ~ 5.8 | 0.32 ~ -5.26 | 2.53 ~ -1.41 |
| | TT | -1.68 ~ 5.92 | 3.7 ~ -2.05 | 7.07 ~ -7.86 |
| | SS | -1.36 ~ 5.52 | 5.54 ~ -0.19 | 8.91 ~ -6.07 |
| NAND2X1 | FF | -2.84 ~ 4.76 | -2.64 ~ -5.62 | -1.97 ~ -2.06 |
| | TT | -2.71 ~ 5.2 | 0.36 ~ -2.58 | 3.47 ~ -13.95 |
| | SS | -1.57 ~ 4.58 | 1.72 ~ -0.99 | 5.93 ~ -11.74 |
| NANDX2 | FF | -3.34 ~ 4.8 | -3.04 ~ -5.82 | 0.19 ~ -11.99 |
| | TT | -2.33 ~ 5 | -0.5 ~ -3.16 | 6.31 ~ -4.69 |
| | SS | -2.13 ~ 4.75 | 0.27 ~ -1.71 | 8.61 ~ -2.88 |

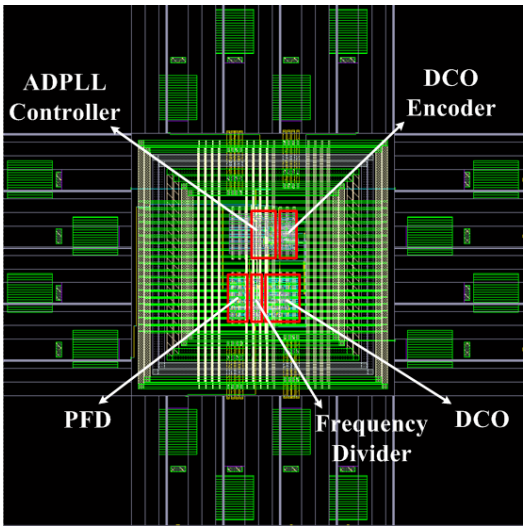


FIGURE 6. LAYOUT OF ADPLL TEST CHIP.

To verify the proposed ADPLL compiler, the proposed ADPLL is implemented in a standard performance 90nm CMOS process with a 1.0V power supply. The layout of the ADPLL test chip is shown in Fig. 6. The active area is 180µm×180µm, and the chip area including I/O pads is 760µm×760µm. The architecture of the ADPLL is similar to the ADPLL [3] without the fast lock-in feature.

Table III shows the comparisons with prior researches. The proposed ADPLL is composed of a phase and frequency detector (PFD) [7], an ADPLL controller [8], a frequency divider, and a DCO which is automatic generated by the proposed ADPLL compiler with 63 coarse-tuning stages. The proposed ADPLL has a fine DCO resolution and smallest chip area, and it also has a wide frequency range.

## IV. Conclusion

In this paper, an all-digital phase-locked loop (ADPLL) compiler with liberty timing files (.lib) is presented. The proposed ADPLL compiler uses a flexible DCO architecture with the proposed frequency range estimation algorithm to automatically generate the DCO circuit which meets the user input specifications. In addition, the proposed ADPLL is designed in an all-digital approach with

TABLE III. PERFORMANCE COMPARISONS

| Parameter | Proposed | [8] JSSC'11 | [9] TCAS-I'09 | [10] TVLSI'11 | [11] JSSC'03 | [12] TCAS-I'12 |
|---|---|---|---|---|---|---|
| Category | All-Digital PLL | All-Digital PLL | All-Digital PLL | All-Digital PLL | All-Digital PLL | All-Digital PLL |
| Process | 90nm | 65nm | 0.18μm | 0.18μm | 0.65μm | 90nm |
| Supply Voltage | 1.0V | 1.0V | 1.8V | 1.8V | 5V | 1.0V/1.2V |
| Input Frequency (MHz) | 50 | 0.036 ~12.5 | N/A | 0.0303~100 | 0.011 ~ 0.339 | 60 |
| Multiplication Factor | 2 ~ 256 | 16 ~ 5600 | 16 | 1 ~ 2046 | 4 ~ 1022 | 64 |
| Time Resolution | 4.1 ps | 16.2 ps | N/A | 8.8 ps | 170 ps | 20 kHz/LSB |
| Power Consumption | 7.74mW @2.17GHz | 1.81 mW @520MHz | 15.7 mW @1.04GHz | 26.7 mW @600MHz | N/A | 8.48 mW@1.0V 10.08 mW@1.2V |
| Lock-in Time | 38 cycles | N/A | N/A | N/A | 7 cycles | < 45 cycles |
| Output Frequency (MHz) | 123.28~ 2168.25 | 90 ~ 527 | 33 ~ 1040 | 62 ~ 616 | 0.045 ~ 61.3 | 4080 |
| Area ($mm^2$) | 0.0324 | 0.07 | 0.32 | 0.14 | N/A | 0.34 |

standard cells. Thus it can be easily ported to different process in a short time. Most of all, the proposed frequency range estimation algorithm can accurately estimate the frequency range of the DCO without performing SPICE circuit simulation.

## ACKNOWLEDGMENT

The authors would like to thank their colleagues in the Silicon Sensor and System (S3) Laboratory of National Chung Cheng University for many fruitful discussions. The EDA tools supported by National Chip Implementation Center (CIC) are acknowledged as well.

## REFERENCES

[1] Rong-Jyi Yang and Shen-Iuan Liu, "A 40-550 MHz harmonic-free all-digital delay-locked loop using a variable SAR algorithm, " *IEEE Journal of Solid-State Circuits*, vol. 42, no. 2, pp. 361-373, Feb. 2007.

[2] Duo Sheng, Ching-Che Chung, and Jhih-Ci Lan, "A monotonic and low-power digitally controlled oscillator using standard cells for SoC applications," *in Proceedings of International Asia Symposium on Quality Electronic Design (ASQED),* Jul. 2012, pp. 123-127.

[3] Ching-Che Chung, Duo Sheng, and Wei-Siang Su, "A 0.5V/1.0V fast lock-in ADPLL for DVFS battery-powered devices, " *in Proceedings of International Symposium on VLSI Design, Automation, and Test (VLSI-DAT)*, Apr. 2013.

[4] Terng-Yin Hsu, Bai-Jue Shieh, and Chen-Yi Lee, "An all-digital phase-locked loop (ADPLL)-based clock recovery circuit, " *IEEE Journal of Solid-State Circuits*, vol. 34, no. 8, pp. 1063-1073, Aug. 1999.

[5] Terng-Yin Hsu, Chung-Cheng Wang, and Chen-Yi Lee, "Design and analysis of a portable high-speed clock generator, " *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 48, no. 4, pp. 367-375, Apr. 2001.

[6] Chao-Wen Tzeng, Shi-Yu Huang, and Pei-Ying Chao, "Parameterized all-digital PLL architecture and its compiler to support easy process migration, " *in press, IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Sep. 2013.

[7] Ching-Che Chung and Chen-Yi Lee, "An all-digital phase-locked loop for high-speed clock generation, " *IEEE Journal of Solid-State Circuits*, vol. 38, no. 2, pp. 347-351, Feb. 2003.

[8] Ching-Che Chung and Chiun-Yao Ko, "A fast phase tracking ADPLL for video pixel clock generation in 65nm CMOS technology, " *IEEE Journal of Solid-State Circuits*, vol. 46, no. 10, pp. 2300-2311, Oct. 2011.

[9] Kwang-Hee Choi, Jung-Bum Shin, Jae-Yoon Sim, and Hong-June Park, "An interpolating digitally controlled oscillator for a wide-range all-digital PLL," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 9, pp. 2055-2063, Sep. 2009.

[10] Hsuan-Jung Hsu and Shi-Yu Huang, "A low-jitter ADPLL via a suppressive digital filter and an interpolation-based locking scheme," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 1, pp. 165-170, Jan. 2011.

[11] Takamoto Watanabe and Shigenori Yamauchi, "An all-digital PLL for frequency multiplication by 4 to 1022 with seven-cycle lock time, " *IEEE Journal of Solid-State Circuits*, vol. 38, no. 2, pp. 198-204, Feb. 2003

[12] Ja-Yol Lee, Mi-Jeong Park, Byung-Hun Min, Seongdo Kim, Mun-Yang Park, and Hyun-Kyu Yu, "A 4-GHz all digital PLL with low-power TDC and phase-error compensation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 8, pp. 1706-1719, Aug. 2012.