國立中正大學

資訊工程研究所碩士論文

40 奈米製程下開發與設計快速鎖定之全數位鎖相迴路

Design of A Fast Lock-in All-Digital Phase-Locked Loop in 40-nm CMOS Technology

研究生：　　羅啟光

指導教授：　鍾菁哲　博士

中華民國　一零四　年　七　月

國立中正大學碩士班研究生

學位考試同意書

本人所指導
資訊工程學系

研究生 羅啟光 所提之論文

40 奈米製程下開發與設計快速鎖定之全數位鎖相迴路 Design of A Fast Lock-In All-Digital Phase-Locked Loop in 40-nm CMOS technology

同意其提付 碩 士學位論文考試

指導教授 鍾菁哲 簽章

104 年 6 月 11 日

國立中正大學碩士學位論文考試審定書

資訊工程學系

研究生 羅啟光 所提之論文

40 奈米製程下開發與設計快速鎖定之全數位鎖相迴路 Design of A Fast Lock-In All-Digital Phase-Locked Loop in 40-nm CMOS technology

經本委員會審查，符合 碩 士學位論文標準。

學位考試委員會
召集人 李順裕 簽章

委員

林鳳[illegible]　　盛鐸

鍾菁哲

指導教授 鍾菁哲 簽章

中華民國 104 年 7 月 2 日

博碩士論文授權書

103CCU00392037

（本聯請裝訂於論文紙本書名頁前空白處，供學校圖書館做為授權管理用）

ID:103CCU00392037

本授權書所授權之論文為授權人在 國立中正 大學(學院) 資訊工程研究所 系所 ________ 組 103 學年度第 二 學期取得 碩 士學位之論文。

論文題目： 40 奈米製程下開發與設計快速鎖定之全數位鎖相迴路

指導教授： 鍾菁哲,Ching-Che Chung

茲同意將授權人擁有著作權之上列論文全文(含摘要)，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印，此項授權係非專屬、無償授權國家圖書館及本人畢業學校之圖書館，不限地域、時間與次數，以微縮、光碟或數位化方式將上列論文進行重製，並同意公開傳輸數位檔案。

紙本論文：茲同意將授權人擁有著作權之上列論文全文(含摘要)，提供讀者基於個人非營利性質之閱覽或列印，此項授權系非專屬、無償授權國立中正大學圖書館做為編目上架及公開陳列閱覽使用。

□ 校內外立即開放
□ 校內立即開放，校外於 2020 年 12 月 31 日後開放
☑ 校內於 2020 年 12 月 31 日；校外於 2020 年 12 月 31 日後開放
□ 其他

授權人：羅啟光

簽 名： 羅啟光　　　　日期： 104 年 8 月 12 日

摘要

近年來，系統晶片(SoC)被廣泛的應用在生醫電子、展頻電路、植入式醫療裝置以及無線網路等應用，由於在 SoC 中，需要供應不同頻率的工作時脈到不同 I/O 介面，因此，鎖相迴路(PLLs)經常使用在 SoC 上，以產生多組不同的工作時脈。此外，這些應用的需求在於低耗能及低成本。

隨著先進 CMOS 製程的開發，工作電壓不斷的下降，傳統 PLL 在設計上遭遇很大的挑戰。基於 time to market 的考量，為了減少重新設計 PLL 的時間與努力，ADPLL 採用了全數位的設計方式，以及使用標準元件(Standard Cell)實現 ADPLL，不僅加速了設計過程，更增加 PLL 的移植性，相較於傳統 PLL，全數位鎖相迴路(ADPLL)更適合用於 SoC 上。

由於傳統 PLL 的鎖定時間都很長，導致 PLL 需要較長的工作時間。當系統待命時，PLL 的耗能即成為 SoC 待機(Standby)時功率消耗的主要來源。因此，若縮短 PLL 的鎖定時間，即早鎖定頻率及相位，PLL 便可隨需求關閉以減少耗能，快速鎖定的 ADPLL 變成設計的趨勢。

因此，在本文中，我們提出快速鎖定的全數位鎖相迴路，透過校正及補償機制來降低面積成本的消耗，以及提升鎖定中心頻率的精準度，是本論文的主要目的及貢獻。本論文以 40 奈米製程標準元件庫實現，並驗證我們所提出的電路架構。

關鍵字：全數位鎖相迴路、快速鎖定全數位鎖相迴路

Abstract

In recent years, biomedical electronic applications, spread-spectrum clock generators, implantable medical devices, and frequency hopping wireless applications are widely used in system-on-a-chip (SoC). In an SoC, it requires different clock sources for different I/O interfaces. Thus, phase-locked loops play an important role in SoC in order to generate different clock sources. Besides, the primary concern of these applications are low energy and low cost.

While the operation voltage is scaling down with the latest CMOS process, analog PLLs encounter great design challenges. According to time to market, in order to minimize the design time and the design efforts, all-digital phase-locked loops (ADPLLs) are adopted in digital design approaches. In addition, ADPLLs implemented with standard cells can not only speed up the design time, but also improve the portability. As compared with analog PLLs, ADPLLs are more suitable for SoC.

Analog PLLs are usually not to be stopped due to the long lock-in time. When the system is in sleeping mode, the PLL power consumption dominates the standby power consumption of the system. Therefore, if PLLs can lock the frequency and phase quickly, the lock-in time can be reduced so that PLLs can be turned off in low power modes. As a result, fast lock-in ADPLLs become more and more popular.

Therefore, in this thesis, we proposed a fast lock-in ADPLL with a calibration method in order to decrease the chip area and improve the accuracy in frequency estimation. In addition, the test chip is implemented and verified in TSMC 40-nm CMOS process with standard cells.

Keywords：All-Digital Phase-Locked Loop (ADPLL), Fast Lock-in ADPLL

Acknowledgments

I would like to express the deepest gratitude to my advisor, Prof. Ching-Che Chung. He always gives me the right direction and enthusiastic guidance, and helps me to complete my research.

I would like to thank to my partners of Silicon Sensor and System (S3) Lab of National Chung Chen University, too. Without their assistance, I could not overcome many difficulties over two years.

Finally, I would like to thank my family and my girlfriend, who always concern for me. This really makes me to be optimistic every time.

Content

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Why the Feature of Fast Lock-in is Required in ADPLL

In recent years, phase-locked loop (PLL) plays an important role in system-on-chip (SoC). In an SoC, it usually requires several PLLs to provide different clock sources for different I/O interfaces. PLLs generate various frequencies to apply to different applications and it is widely used in mobile communication system, wireless communication system and biomedical electronic.

Traditionally, PLL is implemented by a charge-pump phase-locked loop which faces design challenge of leakage current in advanced CMOS process. Besides, the gain of voltage controlled oscillator (K_{VCO}) becomes larger due to the narrower range of the control voltage. Thus it easily suffers from noise on the control voltage. Traditional PLLs contain some passive components to design an analog loop filter that occupies huge chip area and has higher chip cost. Traditional PLLs also have a long lock-in time and have high power consumption. Therefore, in the era of the nano-scale process, PLLs which consist of a VCO, a charge-pump and an analog loop filter are not suitable for SoC that demands low power, low voltage operation, low chip area and especially requires fast locking [18].

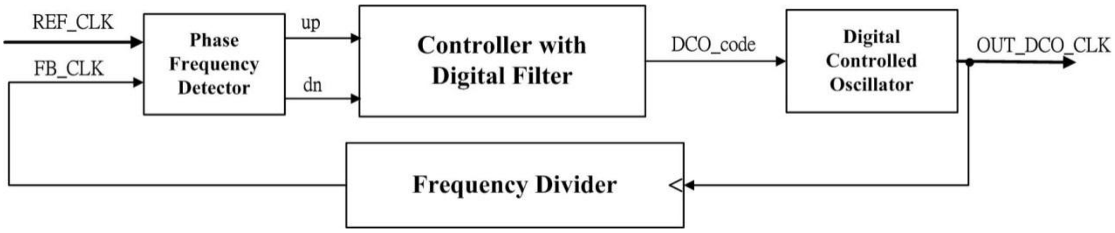


Fig. 1.1: ADPLL Architecture.

An all-digital phase-locked loop (ADPLL) takes the advantages of CMOS process as shown in Fig. 1.1. Compared with the traditional PLL, the ADPLL replaces passive components with the digital circuits, a VCO to a digital-controlled oscillator (DCO), and an analog filter to a digital filter. The chip size is reduced and the chip cost decreased accordingly. As a result, an ADPLL can operate at low voltage and have low chip area due to all digital design techniques.

The evolution of traditional PLLs into digital PLLs, some issues need to conquer such as the efficacy of the DCO. A DCO occupies about half of the PLL area and also consumes most portions of the power of the ADPLL. Therefore, DCO is the most critical component in an ADPLL. In [21], it presents a smooth code jumping technique to reduce output jitter. However it requires two DCOs for comparison and costs huge area. Moreover, there are some papers research on a wide-range, high resolution and monotonic DCO. In [22], a DCO architecture that can be glitch free is presented.

In [23], an interpolating DCO is proposed to improve the architecture of DCO. The sequential bandwidth needs to overlap in order to prevent from the PVT variation. However, the performance of the DCO [23] becomes worse as operating at high frequency, the differential nonlinearity (DNL) is higher than 0.4 LSB. A wide frequency range and low supply sensitivity DCO is proposed in [24], it tunes the transistor sizes in order to compensate the delay variation of delay line due to voltage variations. However, migrating to another CMOS process or with the DC voltage drop, the size of the transistors required to be retuned. In [20], a monotonic DCO with the calibration circuit achieves low power consumption and low jitter performance. As a result, DCOs can reach low power [33], low jitter, fine resolution [18], and wide frequency range [23] that is more suitable than traditional PLLs in SoC applications.

In wire and wireless applications, digital PLLs become popular due to the process scaling down. In [1], a fast-lock digital PLL with a LC DCO that generates eight phase. The LC DCO operates at high frequency and generates a stable clock to the system. However, the LC DCO demands for large area. In order to reduce chip area and device noise, [28] proposes three TDCs and a DCO to replace the LC structure. Although three TDCs and a DCO successfully decrease the chip area, three TDCs encounter the PVT variation that causes poor linearity. On the other hand, [28] abandons the architecture of LC DCO, while operating at high frequency, the device noise must worse than the LC DCO. As a result, wire and wireless application requires for high frequency, ADPLL with LC DCO meets the requirement and provides stable signal to the system.

Furthermore, injection-locked [35]-[39] has been used in PLLs recently to improve the jitter and phase noise. The primary premise of injection-locked PLL is the feedback clock should be continuously output and closes to the reference clock. Therefore, injection-locked PLLs require addition calibration process to make sure the feedback clock is in the locking window with the PVT variation. If the feedback clock exceeds the locking window, the injection-locked PLL will be failed. For instance, in [39], the locking window is designed less than 1MHz. However, injection-locked transfers from VCO to DCO, in [37]-[39], these papers exploit DACs or varactors to achieve digital controlled. As a result, ADPLL based on injection-locked attains low jitter, low power and small area. However the limitation is the narrow locking window.

According to $P = CV^2 f$, reducing the operation supply voltage is an efficient approach to minimize the dynamic power consumption. However, scaling down the operation supply voltage causes design challenges in traditional PLLs. ADPLLs can conquer this situation and work at low supply voltage to achieve the low power consumption [10] [25]-[27] [40]. In [25], an ADPLL with a digital supply regulator (DSR) increases the DCO supply voltage so that the DCO can operate at a higher supply voltage than reference voltage. However, the DSR consumes almost half of the power of overall and occupies lots of chip area.

In [26], a DCO is designed which not runs at full swing so that can minimize the power consumption. Although it can operate at 0.5V supply voltage and optimizes the power consumption, it has poor jitter performance and costs extra chip area. Furthermore, [27] proposes a customized set-reset latch (SR-Latch) that can survive in 0.25V supply voltage and latches the value. However, it is not able to dynamically adjust the multiplication factor and the sub frequency bands of the DCO require being overlapped to overcome the PVT variations. Besides, the lock-in time is increased due to the low supply voltage. In [10], it provides a pulse latch DFF which can operates at low supply voltage and the ADPLL also achieves fast lock-in time. For time to market, an ADPLL is more suitable in SoC design.

Fast lock-in ADPLLs [10] [11] [13] [18] [19] and [29] are widely used in biomedical electronic applications, frequency hopping wireless applications, spread-spectrum clock generators, grid applications, and implantable medical devices. These applications require reducing the standby power consumption and switch to different frequency channels as fast as they can. If an ADPLL has a fast lock-in time, when high-speed clock generation is not required, the ADPLL can be turned off into sleep mode in order to minimize the power consumption. Thus, fast lock-in ADPLLs are important to meet the low-power demand.

1.2 Conventional Fast Lock-in Methods Survey

1.2.1 Binary Search Algorithm

The binary search algorithm (BSA) is a well-known approach used in tracking frequency and phase in [15], [16], [19] and [34], for frequency and phase tracking process. As searching for the target frequency, the search step is arranged in descending order. In [19], after the system reset, the PFD starts detecting the phase and frequency error between the reference clock and the feedback clock. Then, the PFD outputs leads or lags to the ADPLL controller which determines the DCO should speed up or slow down. While the output of the PFD changes from leads to lags (or lags to leads), the polarity is triggered and the frequency gain is divided by two repeatedly until the frequency gain is reduced to the minimum value. In the end, the target frequency is acquired as shown in Fig. 1.2.

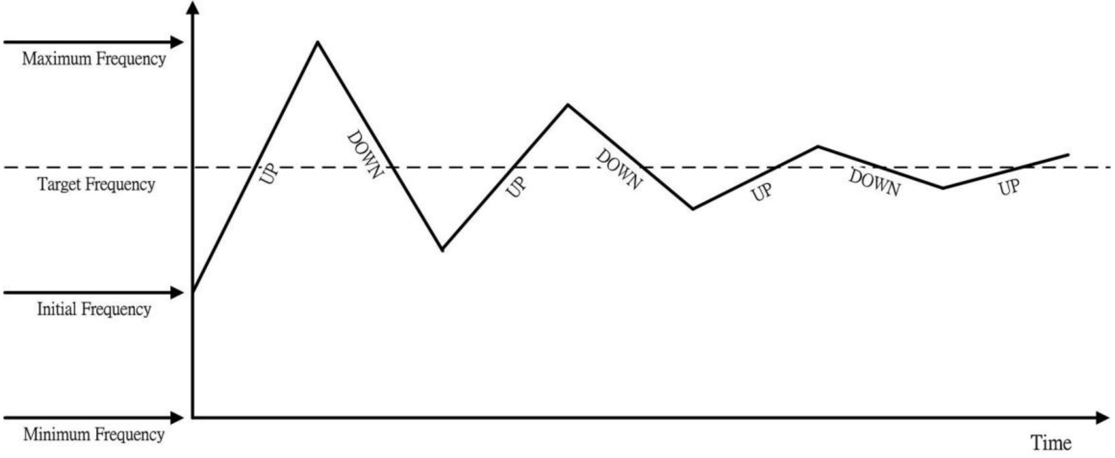


Fig. 1.2: Binary Search Algorithm.

In [15], a frequency comparator which has two frequency gains for adding or subtracting is presented. The frequency comparator detects the relative frequency of the reference clock and the DCO clock, then the frequency comparator outputs fast signal or slow signal that depends on the DCO clock is faster or slower than the reference clock. While the comparator output changes from fast signals to slow signals, the algorithm reduces to a quarter of the frequency gain.

For example, in table 1.1 we initialize the DCO control code at 640; the add gain at 640; the sub gain at 320. In the first cycle, the comparator outputs a slow signal so that the DCO control code adds the value from the add gain to speed up the DCO. Then, the comparator outputs the fast signal, thus the DCO control code is subtracted the value from the sub gain to 960 and the add gain is divided by four to 160. While the slow signal consecutive occurs, the DCO control code is added by the value of the add gain twice and so on. Therefore, when the fast signal and a slow signal are interchanged, it divides the frequency gain value by four. Otherwise, the frequency gain value is preserved and the ADPLL controller keeps tracking the frequency.

Table 1.1: An Example of Binary Search Algorithm in [15].

Frequency Compare	**Initial**	**slow**	**Fast**	**slow**	**slow**	**fast**
Add Gain	640	640	160	160	160	40
Sub Gain	320	320	320	80	80	80
DCO Control Code	640	1280	960	1120	1280	1200

In [16], a modified binary search algorithm is shown in Fig. 1.3. While the PFD output changes from lead to lag (or lag to lead), the modified binary search algorithm records the control code and jumps to a calculated code which is the averaged value of the last record DCO control code and the current DCO control code. Therefore, the modified binary search algorithm not only reduces the lock-in time, but also stabilizes the frequency during the frequency tracking.

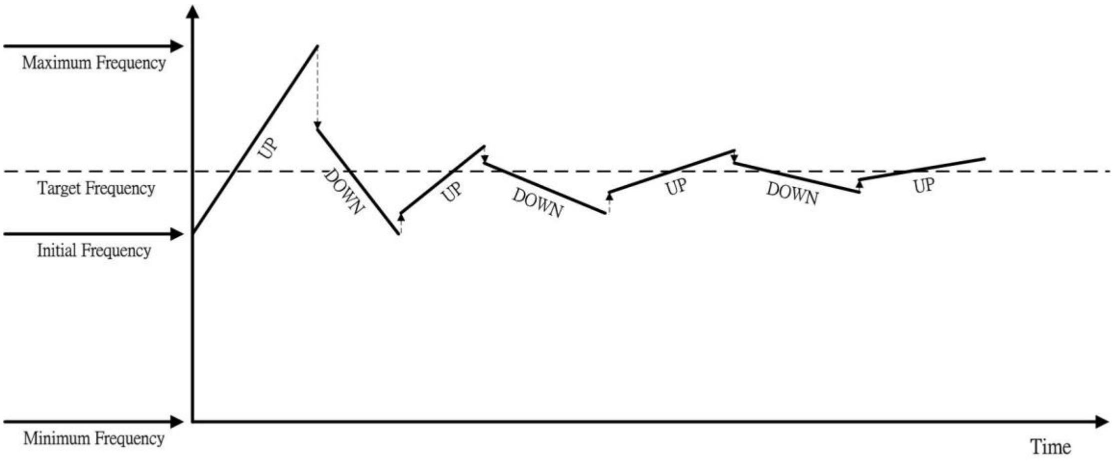


Fig. 1.3: Modified Binary Search Algorithm.

The binary search algorithm is intuitional and has low design complexity. However, it still spends a lot of time in obtaining the target frequency. The initial DCO control code is tricky, if the initial control code is closed to the target frequency, the ADPLL lock-in time can be significant reduced. However we are not able to find the target DCO control code at the beginning. If the initial code is set to the target control code, the output frequency of the ADPLL will be far from the target frequency due to the large gain at first. In addition, the binary search algorithm takes several cycles to relock the frequency. Consequently, the binary search algorithm is not suit for fast lock-in ADPLLs owning to a long lock-in time.

1.2.2 Gear Shifting

The gear-shifting algorithm is proposed in [7], [8] and [30]. The gear-shifting algorithm changes the bandwidth in different modes in order to acquire the target frequency fast and reduce the phase noise.

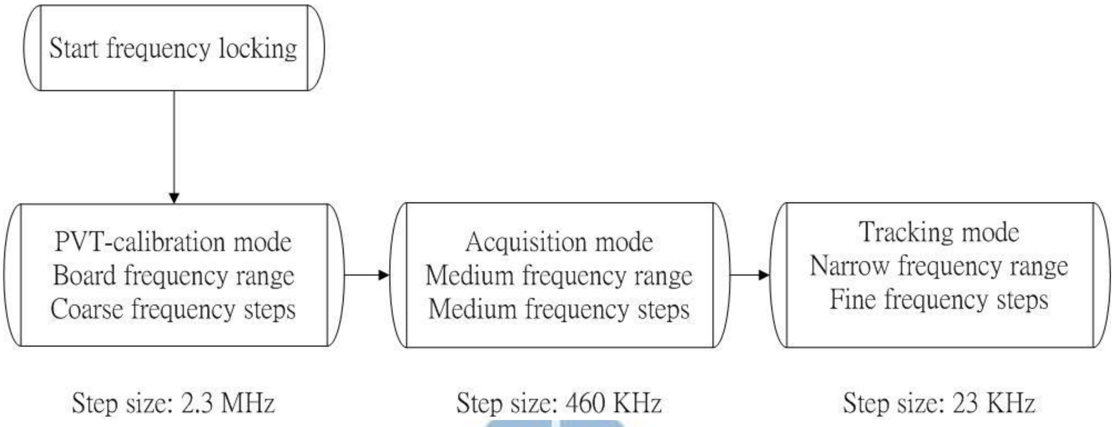


Fig. 1.4: Modes of the Gear-Shifting Algorithm.

The modes of the gear-shifting algorithm are shown in Fig. 1.4. Three DCO banks are switched during the frequency locking. The PVT-calibration mode is used to calibrate the DCO center frequency. As tracking the target frequency, the loop bandwidth is wide to achieve fast lock-in. In [7], it sets the frequency search step to 460 KHz to speed up the PLL in the acquisition mode. The wide bandwidth results in large phase noise. During the phase tracking mode, the gear-shifting algorithm uses a narrower bandwidth to reduce the phase noise and spur after the acquisition mode.

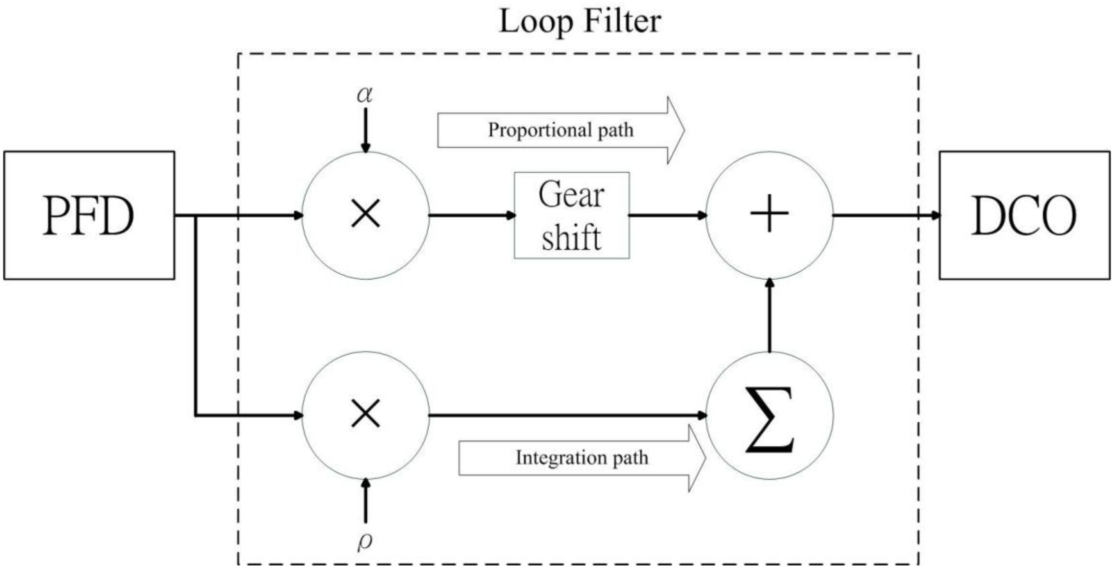


Fig. 1.5: Block Diagram of the Loop Filter.

The digital loop filter is shown in Fig. 1.5, two paths a proportional path and an integration path are changing during the frequency tracking. The proportional path is changed with the current output of the phase frequency detector (PFD); the integration path is accumulated by the history of the PFD output. Each output from the PFD is added in the proportional path to control the DCO. At the same time the integration path is accumulated the phase error. While in different modes, the integration path provides different amount of phase error integration results. As the PLL in the acquisition mode, the proportional path is selected to acquire the target frequency as fast as it can. As the PLL in the tracking mode, the integration path is selected to reduce the control code variations and to stabilize the DCO. At last, the loop filter feds the controlled code to the DCO by adjusting weighting ratios between the proportional path and integration path to achieve fast lock-in.

However, the gear-shifting algorithm [7] uses a narrow bandwidth to stabilize the DCO, it cost less than 50µs (650 reference cycles) to achieve lock-in. Furthermore, the weighting ratios of the integration path and the proportional path are a tradeoff between the lock-in time and the phase noise. As migrating to different CMOS process, the value in the loop filter may be different to achieve fast lock-in. Therefore, the gear-shifting algorithm requires to resimulate in different process.

In [8], a modified bang-bang algorithm which is similar to gear-shifting algorithm is presented. If the PFD outputs n consecutive leadings or laggings, the proportional path of the loop filter outputs β×n as an output code. While we compare with the gear-shifting algorithm, the modified bang-bang algorithm violently speeds up the lock-in time in 15μs (750 reference cycles).

Fig. 1.6 shows the architecture of the modified bang-bang algorithm. A cumulative multiplier which multiplies the proportional path is added before the loop filter. The integration path depends on the history record that works the same as the gear-shifting algorithm. While the PFD outputs three consecutive leadings, the proportional path increases as β, 2β and 3β. Therefore, the ADPLL with the modified bang-bang algorithm can achieve fast lock-in.

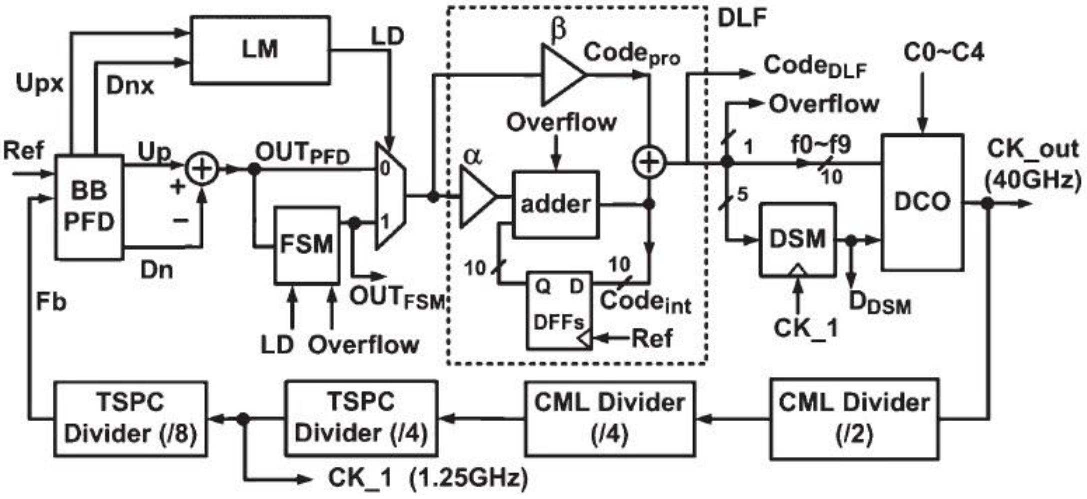


Fig. 1.6: Architecture of Modified Bang-Bang Algorithm-Based ADPLL.

In [31], it discusses the coefficients in different integration paths (K_i) and proportional paths (K_p) and the step response of the ADPLL. As shown in Fig. 1.7, if K_p equals to 1, it can improve the lock-in time. A K_i equals to 500, it can accelerate the loop filter to stable the DCO. Obviously, [31] trades off between K_p and K_i to achieve fast lock-in time and low phase noise. As a result, the gear-shifting algorithm makes a lot of efforts on tuning the value between K_p and K_i. While the ADPLL is ported to different CMOS process, these values are needed to be adjusted.

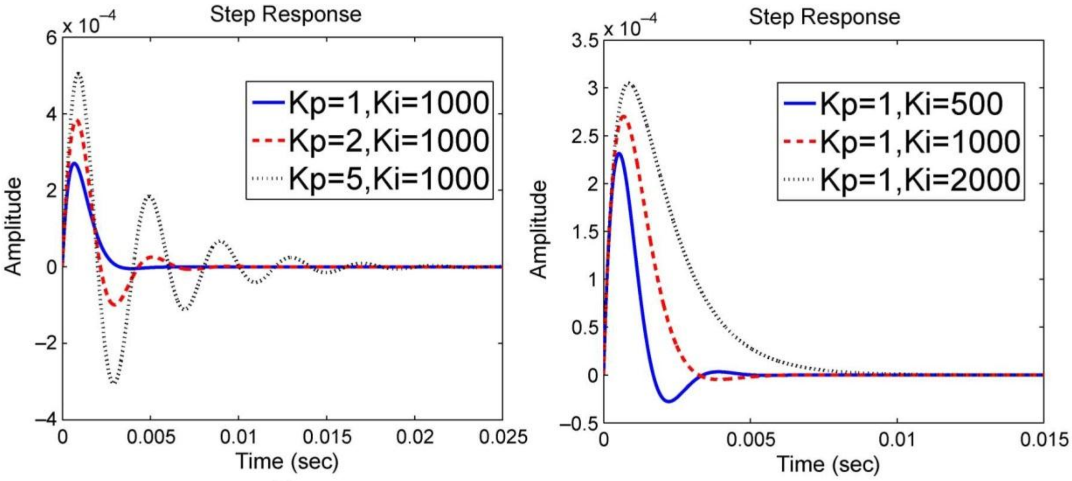


Fig. 1.7: Step Response of the ADPLL under Different Loop Filter Parameters.

1.2.3 Frequency Counter Based Approach

Using the time-to-digital converter (TDC) is another method to achieve fast lock. Recently, TDCs are widely used in ADPLLs design [17] [18] to search the frequency and quantify the phase error between the reference clock and the feedback clock. The TDC converts timing information into digital numbers. While using the TDC to convert the reference clock by the DCO clock, it acts like a counter that inputs a reference clock as data and the counter is triggered by the DCO clock. Then, the output is a ratio between the reference clock and the DCO clock.

In [17], a wide frequency multiplication range with seven-cycle lock-in time is proposed. It utilizes the TDC to acquire the target frequency. The features of [17] are that the ring delay line of the TDC is shared with the DCO delay line. In order to avoid the PVT variation, [17] uses the same delay cells to the TDC and the DCO in order to ensure that they have the same resolution. Fig. 1.8 shows the block diagram of the TDC. Moreover, it consists of an integer part and a fractional part to extend the quantization range of the TDC. In Fig. 1.8, the counter counts the integer part; latch and encoder count the fractional part.

However, the design challenge of [17] is at the TDC resolution which limits the maximum frequency of the input reference clock. In order to avoid the overflow problem in the TDC counter, the resolution of the ring delay line is not good. The worst TDC resolution in [17] is 170ps, thus the ADPLL input frequency can only accept the clock frequency lower than 350 kHz. If the input frequency is higher than 350 kHz, this method will have a large frequency error. Unfortunately, there are few applications work lower than 350 kHz.

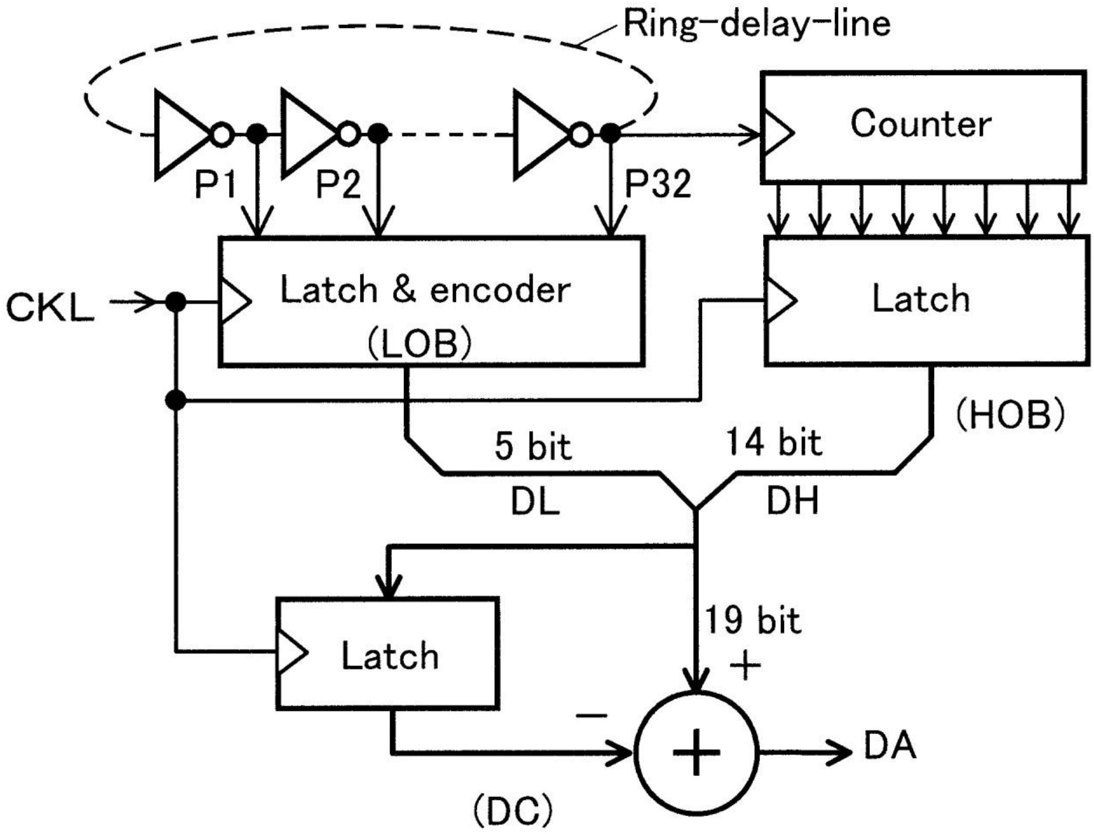


Fig. 1.8: Architecture of TDC [17].

In [18], a 2-level flash TDC to achieve fast lock-in time within two reference cycles is proposed. It has a high resolution and low power consumption due to the 2-level flash TDC which also has a wide quantization range for the TDC as shown in Fig. 1.9. In order to enhance the TDC resolution and reduce the hardware complexity, the TDC only uses 12 D-type flip-flops.

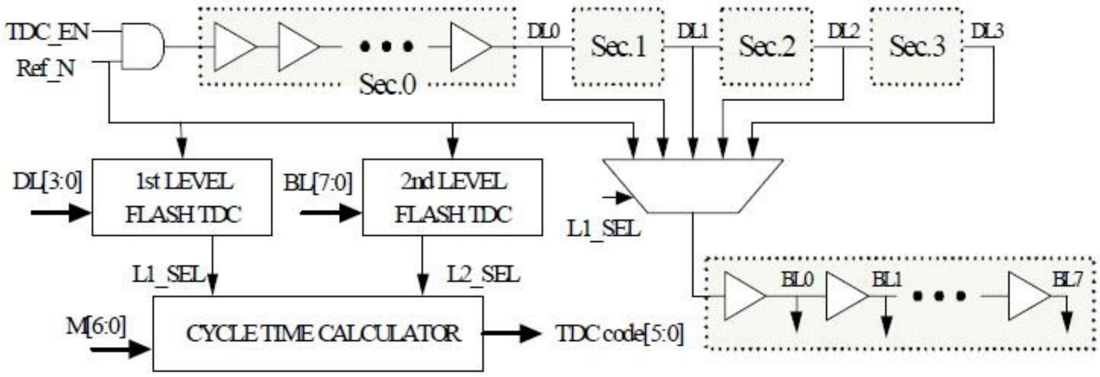


Fig. 1.9: Architecture of 2-level flash TDC.

However, the 2-level flash TDC consists of MUX cells that the intrinsic delay of the MUX cell may worsen the resolution of the TDC. Fig. 1.10 shows the overall architecture of the ADPLL [18]. Designing a fine resolution TDC requires great efforts. For this reason, in this ADPLL, a pre-divider divides the reference clock into a lower frequency before quantizing by the TDC. Therefore, the effective TDC resolution can be improved. In addition the quantization result (TDC_code) of the TDC will record in 6 bits. However the frequency multiplication factor becomes larger, it may have few bits which not been used, thus the TDC quantization error may increase with the frequency multiplication factor.

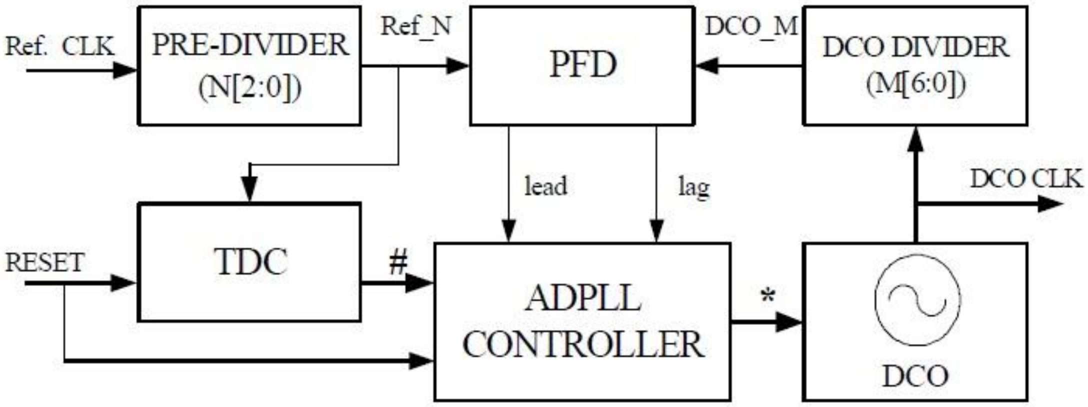


Fig. 1.10: Architecture of ADPLL [18].

1.2.4 Frequency Estimation Algorithm

The frequency estimation algorithm (FEA) can achieve faster lock-in time than the binary search algorithm (BSA). In addition, the frequency counter based approach is proposed in [4], [5], [10] and [13]. The frequency estimation algorithm takes the characteristics of the DCO and calculates the target frequency in a short time. In [4] and [10], they utilize a fine resolution and a high linearity DCO with the monotonic response to estimate the control code for the output frequency. In [5], it uses the Regula Falsi method, therefore it could deal with the DCO with nonlinear or nonmonotonic response. In [13], it proposes techniques that estimate and preset the control code for the ADPLL to reduce the settling time.

In [4], a fast-lock engine (FLE) that achieves two reference cycles lock-in time is proposed. The block diagram of the FLE is shown in Fig. 1.11(a). Fig. 1.11(b) shows the ordinary ADPLL circuit. At the beginning, the ADPLL runs at the minimum frequency and the maximum frequency with two inner DCOs, respectively. Subsequently, two frequency counters quantize the period ratio between the DCO and the reference clock at DCO minimum frequency and DCO maximum frequency into two parameters (i.e. R_{max} and R_{min}), respectively. Consequently, the FLE uses two parameters to calculate the target DCO control code and sends it to the outer DCO. Then the ADPLL can work at the target frequency within two reference cycles.

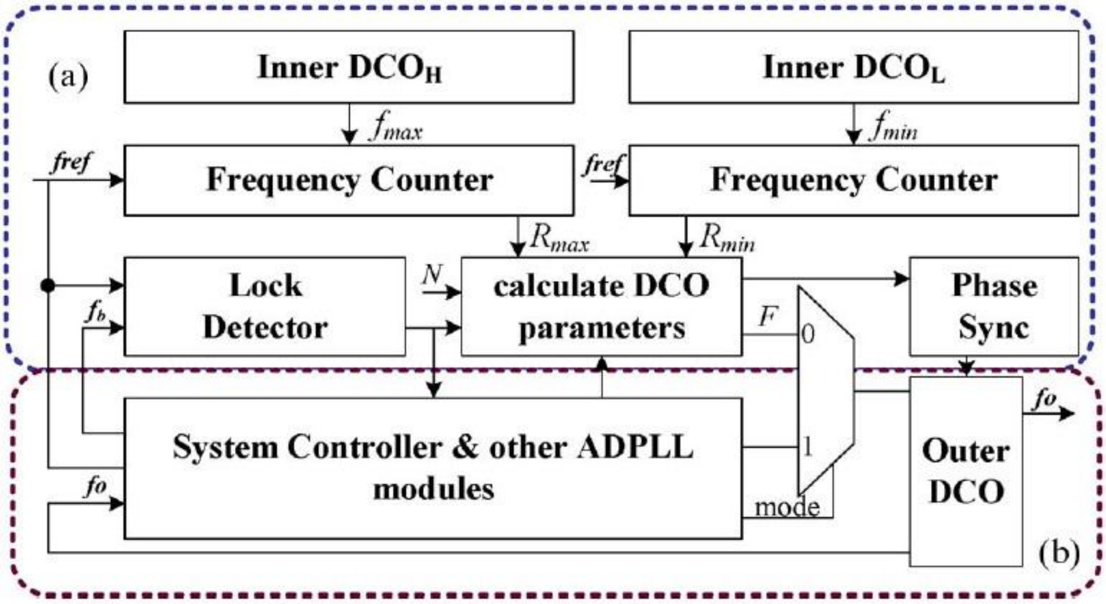


Fig. 1.11: FEA-Based Frequency Synthesizer Architecture.

The frequency estimation algorithm requires a high linearity and a high DCO resolution to acquire a precise target frequency in two reference cycles. However, the differential non-linearity (DNL) of the proposed DCO [4] is larger than one LSB. Hence, the DCO [4] does not have a good linearity characteristic. Furthermore, the FEA-based ADPLL [4] uses two inner DCOs and one outer DCO that cost a large chip area. Besides, on-chip variation can cause these DCOs different that also increases the estimation error in the output frequency.

In [10], another FEA-based fast lock-in ADPLL is proposed by using only one DCO that can achieve fast lock-in within four reference clock cycles. The architecture of the cyclic TDC embedded DCO is shown in Fig. 1.12. Due to the frequency estimation algorithm, the DCO oscillates at the maximum frequency and the minimum frequency to calculate the period ratio in the first cycle and second cycle. Comparing with [4], [10] uses only a DCO and can avoid the on-chip variation problems in [4].

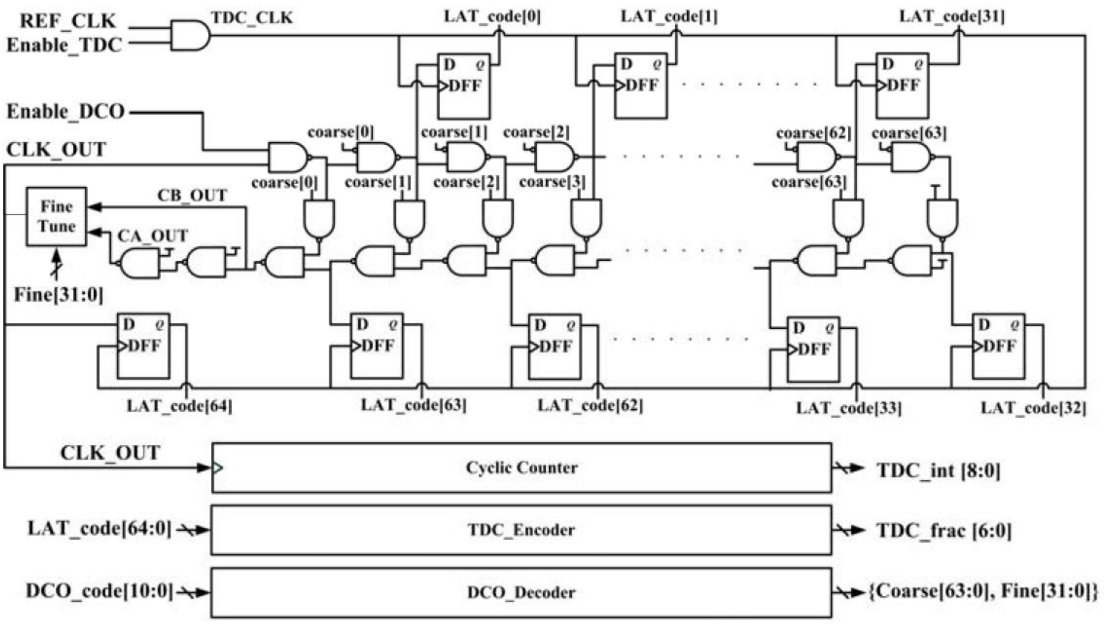


Fig. 1.12: Cyclic TDC embedded DCO.

Even though, the 63-stage D-type flip-flops (DFFs) could quantize the period ratio into a fixed point number to minimize the quantization effect of the TDC. Nevertheless, the cyclic TDC is only used for generating the period ratio at the minimum frequency of the DCO. As the DCO oscillates at the maximum frequency, the TDC can not quantize the period ratio with the fractional part. Consequently, the hardware utilization of the TDC [10] is not sufficient.

On the other hand, the DCO requires oscillating at minimum and maximum frequencies in order to estimate the control code for the target frequency. While the PVT variations are at the best case, the maximum frequency of the DCO may cause the DFFs setup time or hold time violations, for this reason, the operation of the DFFs become unpredictable. Therefore, the frequency finder [10] is not able to calculate the control code for the target frequency correctly in some worst cases.

In [5], a fast lock-in ADPLL by using the Regula Falsi method is proposed. Although the DCO is nonmonotonic or nonlinear, it claims that the ADPLL can lock the target frequency within seven reference cycles. In addition, the lock-in time of the Regula Falsi method is shorter than the binary search algorithm.

As a general frequency estimation algorithm, the Regula Falsi Method also requires the DCO runs at maximum and the minimum frequency at the beginning. If the DCO is linear and monotonic, the first estimation result is the target frequency. Otherwise, they compare the first estimation result with the target N, if the estimation result is larger than the target N, the right bound is replaced by the result. Conversely if the estimation result is less than the target N, the left bound is replaced by the result and keep the right bound. Repeating this action until the result is equal to the target N.

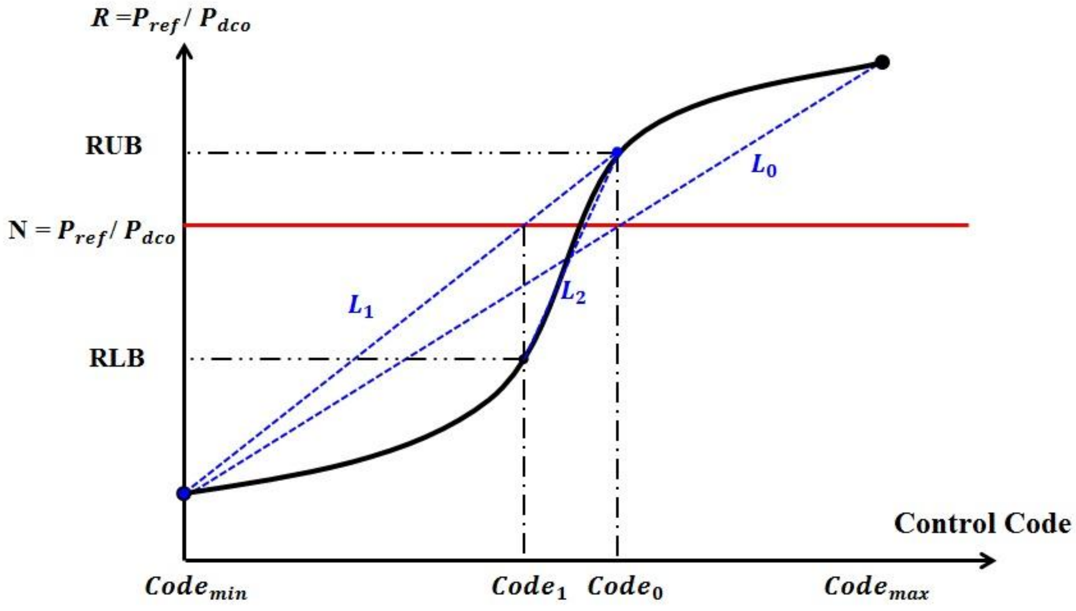


Fig. 1.13: Example of the Regula Falsi Method.

For example, in Fig. 1.13, L_0 is the first line which is used to estimate $Code_0$. However, as running a DCO with $Code_0$, the frequency ratio is larger than the target N. Subsequently, the right bound is replaced by the ratio upper bound (RUB) and keeps the left bound. Next, it utilizes L_1 as a new line to estimate the result again. However the estimation result is less than the target N, it replaces the left bound by ratio lower bound (RLB) and keeps RUB as the right bound. Consequently using L_2 for calculation, the result is equal to the target N. In the end, it outputs the control code to the DCO precisely.

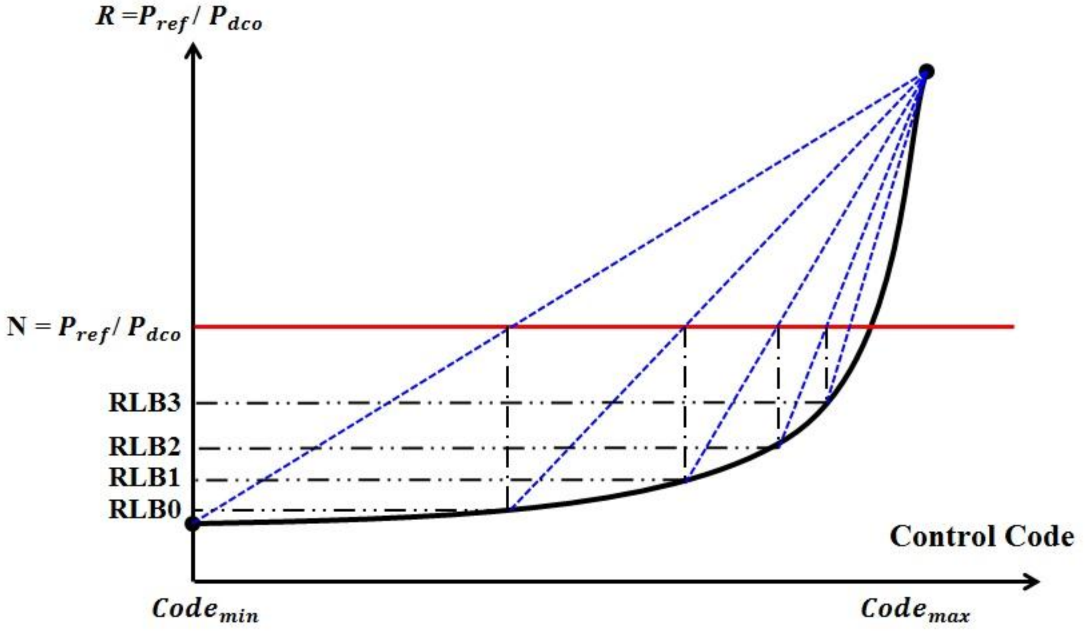


Fig. 1.14: Worst case of the Regula Falsi Method.

The common DCO ratio between the reference clock and the DCO clock is illustrated in Fig. 1.14. While the curve of the DCO ratio operates in Fig. 1.14, it is hard to achieve fast lock-in time. In that case, the estimation result is less than the target N in several iterations, it always replaces the left bound and keeps the right bound. In this case, the Regula Falsi Method does not have a chance to adjust the right bound and converge the estimation line, therefore it requires many reference cycles to achieve lock-in in this example.

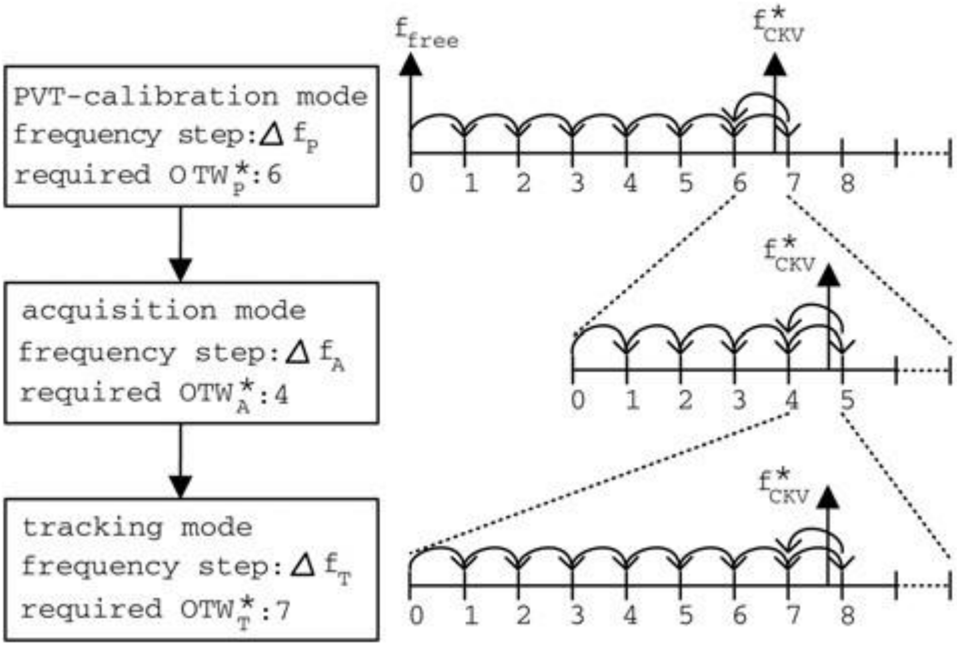


Fig. 1.15: Example of Locking Process in [13].

In [13], an equation that describes the relation between the target frequency and the steps in various modes during acquiring and tracking the frequency is proposed. As shown in Fig. 1.15, this figure shows an example. While tracking the frequency, the PVT calibration mode acquires the frequency toggles between 6 and 7 with Δf_p. Then, the PVT calibration mode locks the code at 6 and the residual frequency error is less than Δf_p. Using Δf_A as the medium frequency step, it acquires the frequency toggles between 4 and 5. Then the acquisition mode locks the code at 4, and then, the residual frequency error is less than Δf_A. In the tracking mode, the PLL uses the finest frequency step Δf_T to track the frequency in the same action as mention before. Therefore, the action of estimating the frequency can be described as Eq. 1.1.

$$f_{target} - f_{initial} = OTW_p \times \Delta f_p + OTW_A \times \Delta f_A + OTW_T \times \Delta f_T + \epsilon \qquad (1.1)$$

$$OTW_P = \left(frequency\ Multiplication - \frac{f_{initial}}{f_{reference}} \right) \times \frac{f_{reference}}{\Delta f_p} \qquad (1.2)$$

Fig. 1.16 shows the flowchart of the locking operation. As the ADPLL power on, it starts self-calibrating the process variations. In this stage, the ADPLL estimates the value of Δf_p, Δf_A and Δf_T owing to the PVT variations which influence the value. Since the ADPLL obtains the value by the self-calibration, the PLL utilizes the value in estimating the control code (OTW) as shown in Eq. 1.2. In Eq. 1.2, firstly, the frequency multiplication subtracts the ratio of the initial frequency and the reference frequency and times the step ratio of step frequency and reference frequency that explains how many times of the step ratio requires to achieve the target frequency. Subsequently the ADPLL starts the normal operation to track the frequency. In the end, the ADPLL is turned off to reduce the power consumption.

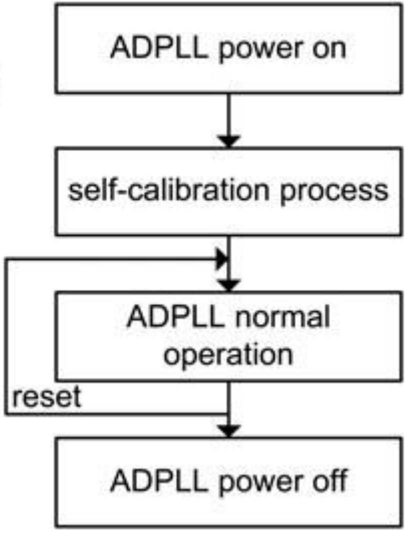


Fig. 1.16: The Flowchart of ADPLL in [13].

Although this method can reduce the settling time by about 50%, the method requires the self-calibration in the beginning that is not efficient enough. On the other hand, the estimation equation requires a high DCO resolution to ensure the self-calibration is precisely enough. If the self-calibration is not accurate, it will generate a large frequency error during the estimation.

1.2.5 Phase Calibration

In order to achieve fast lock, tracking the frequency and phase concurrently is necessary. Due to the fact that the PFD is a sequential logic circuit which continuously outputs leads or lags of comparing the frequency and phase of the reference clock and feedback clock. Therefore, [1]-[4] present a phase calibration scheme to align the phase before entering the PFD in order to reduce the time of locking phase. For design a fast lock-in PLL, it is necessary to align the phase of the feedback clock and the reference clock at the beginning.

In [2], a single-shot phase synchronizer is proposed that synchronizes the reference clock and the feedback clock at the rising edges. The single-shot phase synchronizer combines with delay cells and latches. Each delay cell is arranged in pairs with a latch. Therefore, the feedback clock is fed to the delay cell that the delay cells are lined up in the delay line as shown in Fig. 1.17.

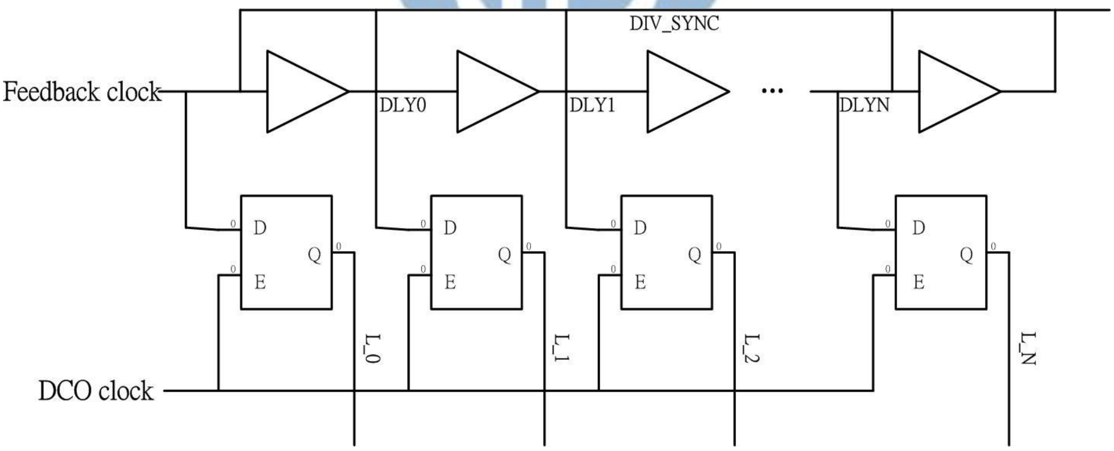


Fig. 1.17: Single-Shot Phase Synchronizer Schematic.

After the DCO is released from reset, the DCO starts working with the first rising edge of the reference clock. Then, the feedback clock is fed to the delay line which generates different phases. At the same time, the single-shot phase synchronizer delays a copy of the reference clock for synchronizing. The feedback clock is delayed by each delay cell and latched the value by the DCO clock.

According to the latch value, the single-shot phase synchronizer can acquire the relative position of the feedback clock and the reference clock.

However, [2] uses the single-shot phase synchronizer at the beginning, the phase is synchronized in the first cycle of the acquisition state before fed into the PFD. In the second cycle, after the PFD detects leads or lags from the feedback clock and the reference clock, the frequency of the feedback clock is changed. As the frequency changes, the phase is not aligned. Besides, the single-shot phase synchronizer requires a multi-phase DCO. Additionally, the resolution of delay cell determines the phase error. Therefore, it is not suitable for high frequency operation.

In [3], a phase error compensation which can accumulate the phase error and adjust the frequency smoothly is proposed. This paper focuses on wireless communication applications which require frequency hopping. If a large initial phase error at the instant of frequency hopping, the PLL is unable to compensate for the large phase error immediately. Therefore, the long settle time is inevitable.

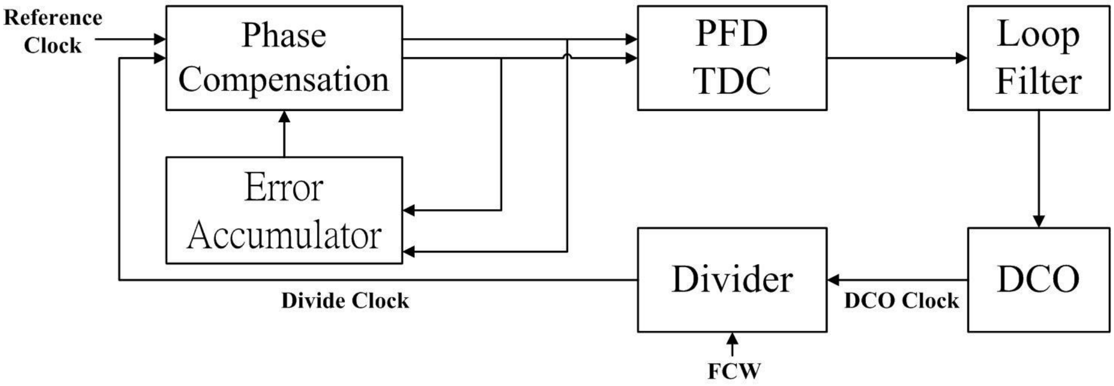


Fig. 1.18: Block Diagram of Phase Error Compensation.

The concept of the phase error compensation is dynamically modified the divider ratio as shown in Fig. 1.18. While the synthesizer jumps to another channel, the divider ratio changes form M to N. The error accumulator accumulates the phase error due to the frequency change. After the phase tracking mode, according to the output of the TDC, the divide ratio may decrease or increase that depends on M < N or M >

N. If M < N, the divide ratio may decrease. In that case, the next edge of the divide clock arrives earlier to compensate for the large phase error as shown in Fig. 1.19. At the same time, the loop filter is adjusted to a wide bandwidth to speed up the locking process. After the locking process, the loop filter is adjusted to a narrow bandwidth to suppress the noise.

However, the FCW (Frequency Controlled word) which controls the divide ratio is inputted from outside the chip. Modifying the value from the outside of the chip is easy to design. For the reason that the ADPLL adjusts the value from outside the chip to reduce the PVT variations, it is more valuable to avoid PVT variations by on-chip circuits. However, adjusting the divide ratio to compensate the phase, the target frequency will drift away easily. Otherwise, it demands on a large divide ratio to decrease the target frequency drift. Besides, although the phase error compensation reduces the phase error continuously, the PLL still takes 25 reference cycles to lock-in due to the narrow bandwidth.

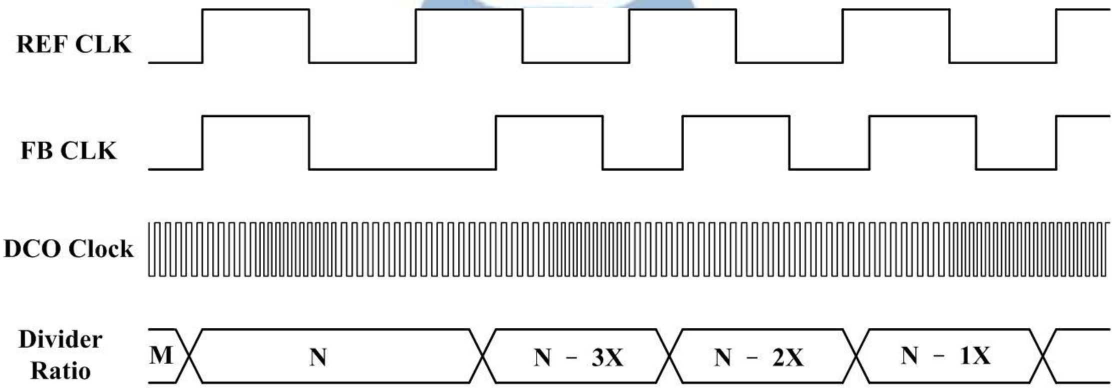


Fig. 1.19: Compensating Timing Diagram.

In [4], they illustrate the phase-synchronized scheme between the reference clock and the DCO clock. In Fig. 1.20, the module “Phase Sync” which accumulates the phase error and outputs a sync signal. When the Phase Sync accumulates to a certain amount of error, the Phase Sync will trigger a sync signal to modify the control code. At the same time, a D flip-flop stops the DCO and waits for the next positive edge of the reference clock to restart the DCO. As shown in Fig. 1.22, the phase is aligned at the next positive edge of the reference clock.

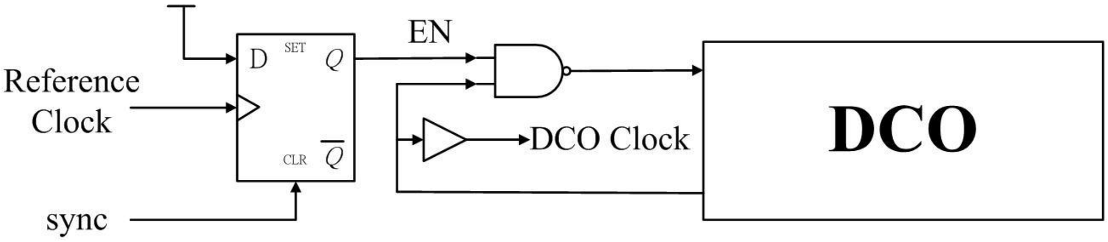


Fig. 1.20: Architecture of the Phase Sync Scheme.

A lock detector (LD) is presented to set the accumulate window which determines when to start tracking the phase. The propagation delay decides the window of the LD that is consists of numbers of the delay buffer stages as shown in Fig. 1.21. M is the number of the buffer stages, and ϵ as the gate delay of the buffer. The accumulate window can express as

$$W_{LD} = M \times \epsilon$$

The application determines the total stages M. Therefore, the M is different in various applications which have different tolerance of the phase error.

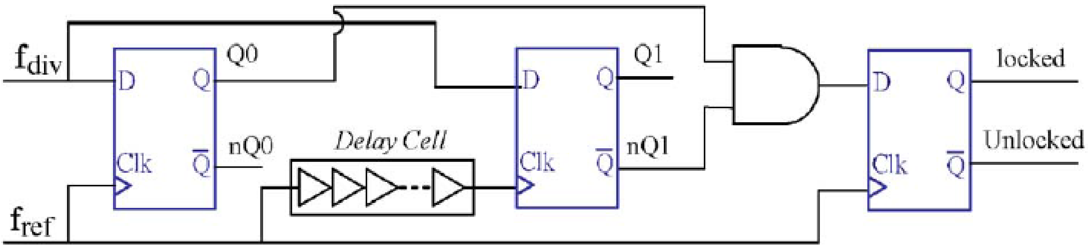


Fig. 1.21: Schematic of Lock Detector.

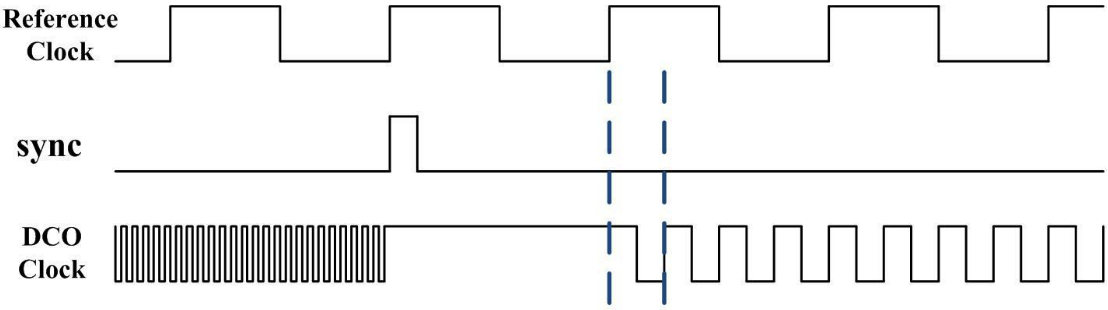


Fig. 1.22: Initializing DCO Timing Diagram.

In [10], this paper also uses an enable signal to turn on or off the DCO. As the enable signal sets to 1, the DCO starts working; as the enable signal sets to 0, the DCO stops working. It is more intuitive to design. Besides, restarting the DCO to initialize the phase error can reduce the costs of area and power consumption. In [4], [5], [9], [10], these papers design to restart the DCO to initialize the phase error. Consequently, the phase error is required to eliminate to achieve fast lock-in. While we compensate for the phase error, restarting DCO at the beginning is more efficient and more suitable.

Table 1.2: Comparison Table of Fast Lock-in Methods

Parameter	Application	Tracking Frequency	Phase Calibration	Gear Shifting	Lock-in Cycles
JSSC 2015 [1]	Global System for Mobile Communication	Binary Search	Phase Calibration State Machine	O	40 (100 ns)
TCAS-II 2012 [2]	Multiprocessor System-on-Chip	Binary Search	Single-Shot Phase Synchronization	X	< 50 (< 1 us)
A-SSCC 2012 [3]	Wireless Communication Industry	Binary Search	Phase Error Compensation	X	25 (5 us)
TCAS-II 2007 [7]	Global System for Mobile Communication	Binary Search	Restart to Initial Phase	O	650 (<50 us)
TCAS-II 2011 [8]	Wireless Transceivers	Binary Search	Modified Bang-Bang Algorithm	X	750 (15 us)
ISCAS 2014 [5]	DDR Controllers USB2.0 USB3.0 SATA2 SATA3	Regula Falsi Method	Restart to Initial Phase	O	7
ICECS 2014 [6]	Wireless Communication Application	Temperature Compensation	Edge Detector	X	45 (3 us)
TCAS-II 2010 [4]	Wireless Communication System	Frequency Estimation Algorithm	Restart to Initial Phase	X	2
NEWCAS 2013 [9]	Wireless Communication Application	Temperature Compensation	Restart to Initial Phase	X	120 (6 us)
TVLSI 2015 [10]	Biomedical Electronic Application	Frequency Estimation Algorithm	Restart to Initial Phase	X	4

In Table 1.2, the table shows that applications apply to different methods which correspond to various lock-in cycles. Fast lock-in methods are widely employed in plenty of applications. According to this table, using the binary search algorithm to track the frequency and phase takes a long lock-in time than using the frequency estimation algorithm. However, although the frequency estimation algorithm can reach fast locking in few reference cycles, the accuracy and frequency error are an important issue required to concern. As a result, fast lock-in ADPLLs becomes a trend in recent years. Besides, how many percentages of the frequency error can be tolerated after frequency tracking is dependent on the applications.

1.3 Summary

There are several ADPLL architectures we have discussed above. In different applications, they have advantages and disadvantages. Therefore, in this thesis, we proposed an FEA-based ADPLL for wearable device that demands on low power and low chip area. The proposed ADPLL is implemented in TSMC 40-nm CMOS process and achieves a fast lock-in time.

The proposed ADPLL improves the frequency estimation algorithm from [10] and [32]. We use the DCO medium frequency and the DCO maximum frequency to quantize the period ratio between the DCO clock frequency and the reference clock frequency. Therefore we can estimate the target DCO control code within 3.5 clock cycles. Then we compensate the estimated value in order to reduce the quantization error within 1 clock cycle. In addition, as the FEA scheme is finished, it outputs the target DCO control code to the DCO within 4.5 clock cycles.

Furthermore, we compare the DCO embedded-cyclic TDC proposed in [10]. Although the cyclic TDC can quantify the period ratio into a fixed point number, it occupies large area. We abandon the cyclic TDC and reduce the quantization error with an equation based on the DCO resolution. On the other hand, we follow the DCO regular placement approach proposed in [32], the regular placement approach improves the DNL of the DCO that makes the DCO more linear.

After the FEA scheme estimates the target DCO control code, the proposed ADPLL adjusts the DCO fine-tuning stage to maintain frequency and phase by the binary search algorithm. As a result, we proposed a fast lock-in ADPLL with low power, low chip area and high accuracy.

1.4 Thesis Organization

In this thesis, the rest of the paper is organized as follow. Chapter 2 presents the proposed ADPLL architecture and the proposed frequency estimation algorithm. In addition we discuss fast lock-in frequency estimation issues. Chapter 3 describes the fast lock-in ADPLL circuit implementation that includes the monotonic digital controlled oscillator, the digital loop filter, the phase and frequency detector and output frequency divider. The specifications and the simulation results of the proposed fast lock-in ADPLL are presented in Chapter 4. Finally, in Chapter 5, we make a conclusion of this thesis and discuss some issues which should be improved in near further.

Chapter 2

Fast Lock-in All Digital Phase Locked Loop

2.1 The Proposed Fast Lock-in ADPLL

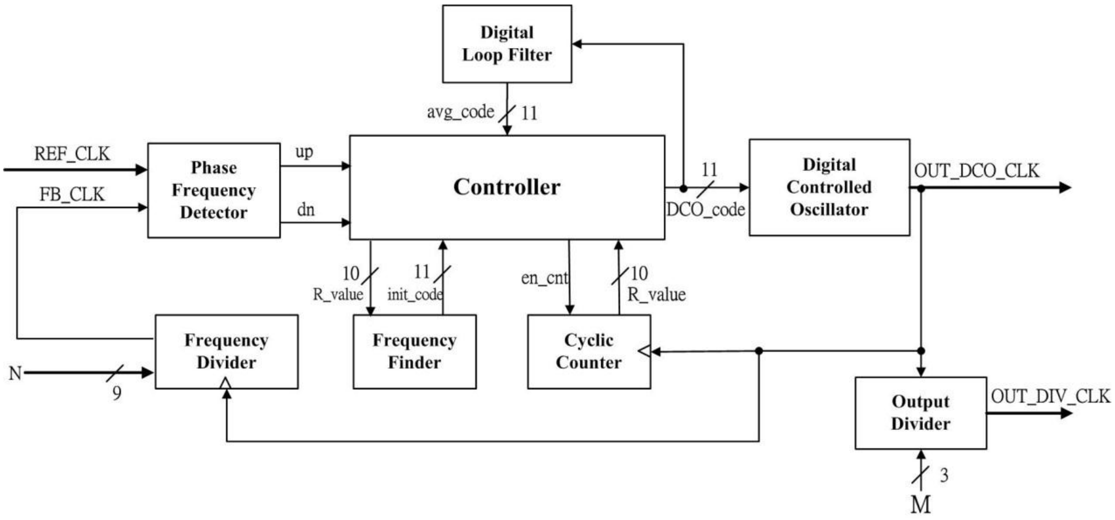


Fig. 2.1: The Block Diagram of the Proposed ADPLL.

A fast lock-in ADPLL approach is proposed in this section and it is implemented in TSMC 40-nm CMOS process. The fast lock-in ADPLL is composed of a phase frequency detector (PFD), an ADPLL controller with a digital loop filter, a monotonic DCO, a frequency divider, a frequency finder, a cyclic counter and an output divider.

Fig. 2.1 shows the block diagram of the proposed ADPLL. In the beginning, the PFD and frequency divider are stopped waiting for the frequency finder to estimate the target DCO control code (init_code). Subsequently, the ADPLL controller acquires the DCO control code and sends to the DCO. We apply a similar FEA-based [10] [4] fast lock-in method in the ADPLL controller. However, [4] requires a fine resolution and a high linear characteristic of the DCO. Otherwise the frequency estimation error becomes large. Therefore, after estimating target frequency, [4] requires extra several cycles to track the target frequency with a large frequency estimation error. Besides, using a frequency counter without any calibration methods cause the large quantization error. Furthermore, three DCOs occupy huge chip area and cause the on chip variations. [10] exploits an embedded cyclic-TDC in order to minimize the quantization error. However, it also costs a lot of area due to the cyclic-TDC.

Therefore, we proposed a calibration algorithm to reduce the quantization error in [10] without a TDC. The calibration algorithm is based on a DCO resolution. We utilize the monotonic characteristic of the DCO to find out the relation between the DCO control code and the ratio between the DCO clock and the reference clock. Finally, the frequency finder estimates a calibration value and calculates the DCO control code within 4.5 cycles.

Subsequently, the PFD and the frequency divider are enabled. The PFD continuously detects the phase and the frequency relation between the feedback clock and the reference clock. At the same time, the PFD updates the DCO control code in order to avoid the phase error or frequency drift. Then, the ADPLL controller receives the DCO control code and communicates to the digital loop filter. The digital loop filter generates the baseline DCO control code in order to reduce the reference jitter and stabilize the DCO output clock.

2.2 The Proposed Frequency Estimation Algorithm

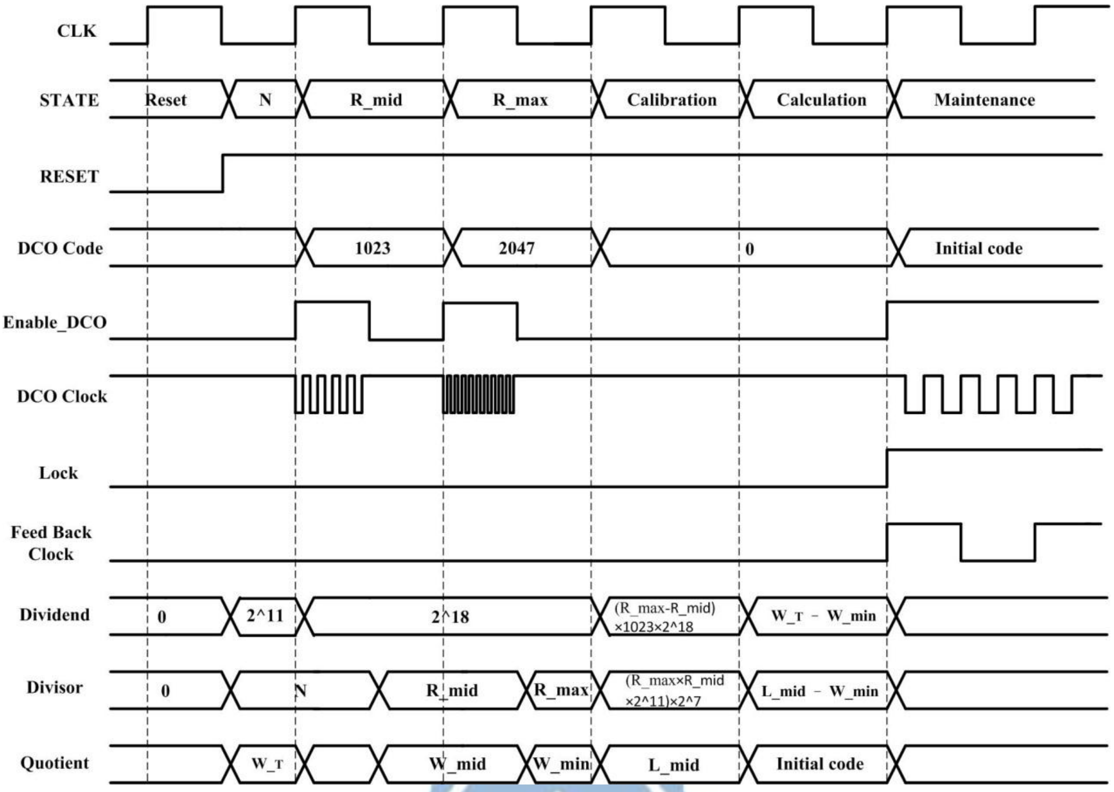


Fig. 2.2: Timing Diagram of the Proposed ADPLL.

The timing diagram of the frequency estimation algorithm with the proposed calibration method is shown in Fig. 2.2. After reset, the frequency finder computes the reciprocal of the multiplication factor (N) within a half cycle. When the DCO control code is set to the medium value of the DCO control code (DCO Code = 1023), the DCO operates at the median frequency with the median period, denoted as P_{mid}. Additionally, when the DCO control code is set to the maximum value of the DCO control code (DCO Code = 2047), the DCO operates at the maximum frequency with the minimum period, denoted as P_{min}.

Subsequently, we utilize the median period (P_{mid}) and the minimum period

(P_{min}) to calculate the R_{mid} and the R_{max} by using the cyclic counter. The definition of the R_{mid} and the R_{max} is the ratio between the reference clock period (P_{ref}) and the DCO clock period at the median frequency and the minimum frequency respectively. The mathematical formula to describe R_{mid} and R_{max} is expressed in Eqs. 2.1 and 2.2. The R_{mid} is the reference clock period divided by the medium clock period and the R_{max} is the reference clock period divided by the minimum clock period.

$$R_{mid} = \frac{P_{ref}}{P_{mid}} \tag{2.1}$$

$$R_{max} = \frac{P_{ref}}{P_{min}} \tag{2.2}$$

The reciprocal of the R value is defined as W value. The definitions of W_{mid} and W_{min} are expressed in Eqs. 2.3 and 2.4.

$$W_{mid} = \frac{2^{11}}{R_{mid}} \tag{2.3}$$

$$W_{min} = \frac{2^{11}}{R_{max}} \tag{2.4}$$

After R values and W values are defined, their relation is shown in Fig. 2.3. This figure shows the R value and W value corresponding to the DCO control code (DCO_code). However, in this figure, the R value is assumed to be estimated by the cyclic TDC proposed in [10]. Thus R value is a fixed-point number. Obviously, the function of R value with DCO code is a hyperbola; the function of the W value with DCO code is a straight line. In [4], it exploits the R code function to estimate the DCO control code as a line that is not accurate enough.

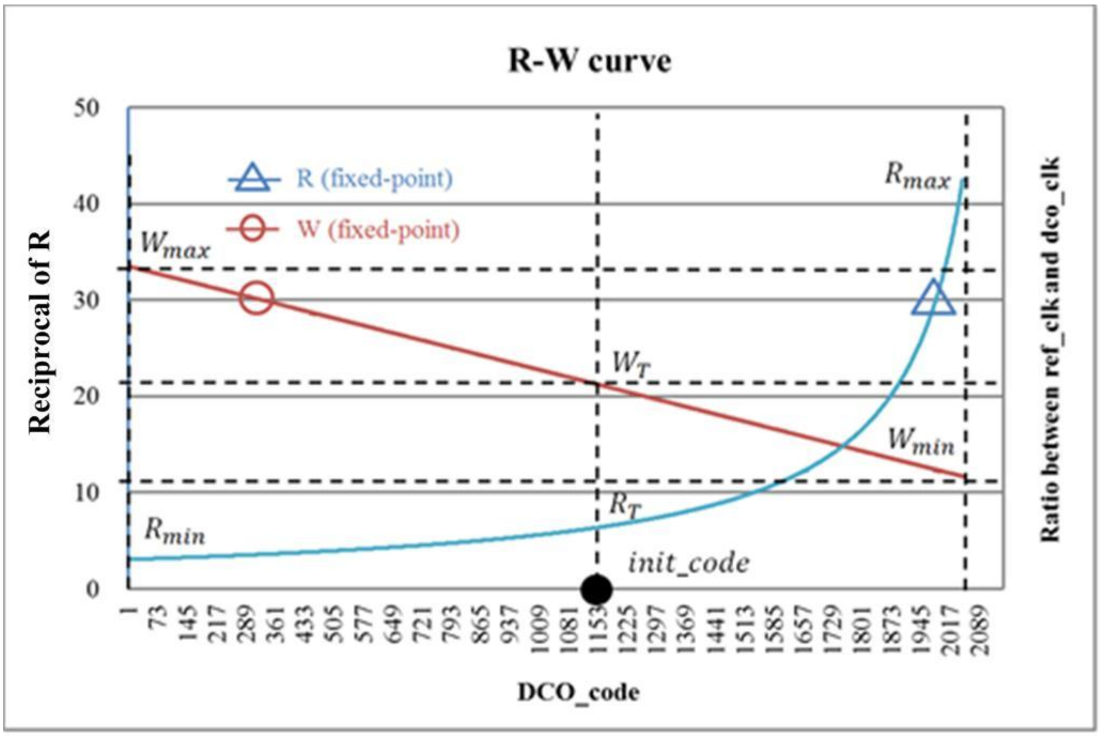


Fig. 2.3: The relationship of *R* value and *W* value.

However, [10] defines and utilizes the *W* code function to estimate the DCO control code precisely with a fixed-point *R* value. As the *W* code function is approximate to a straight line, we can use the linear equation to calculate the target DCO control code. Therefore, the equation of the *W* (code) can be expressed as Eq. 2.5. The interval of W_{min} and W_{mid} times the target code and plus the W_{min} is the target code.

$$W(\text{code}) = W_{min} + \frac{(W_{mid} - W_{min}) \times 2}{2^{11} - 1} \times code \qquad (2.5)$$

However, the *W* code function in Fig. 2.3 is quantified by the cyclic TDC [10]. When we remove the TDC, since *R* value becomes an integer value, the straight *W* line becomes serrated. If we estimate the DCO control code with a serrated line, the quantize error will significantly affect the accuracy of the estimation result. Fig. 2.4

shows the relation between the *W* code function and the DCO control code in three process corners according to the simulation result with integer *R* values. First of all, the *W* line becomes trapezoid, so that is not able to use the linear equation any more. Besides, the quantization error is large especially when the target DCO control code is small and at the SS corner.

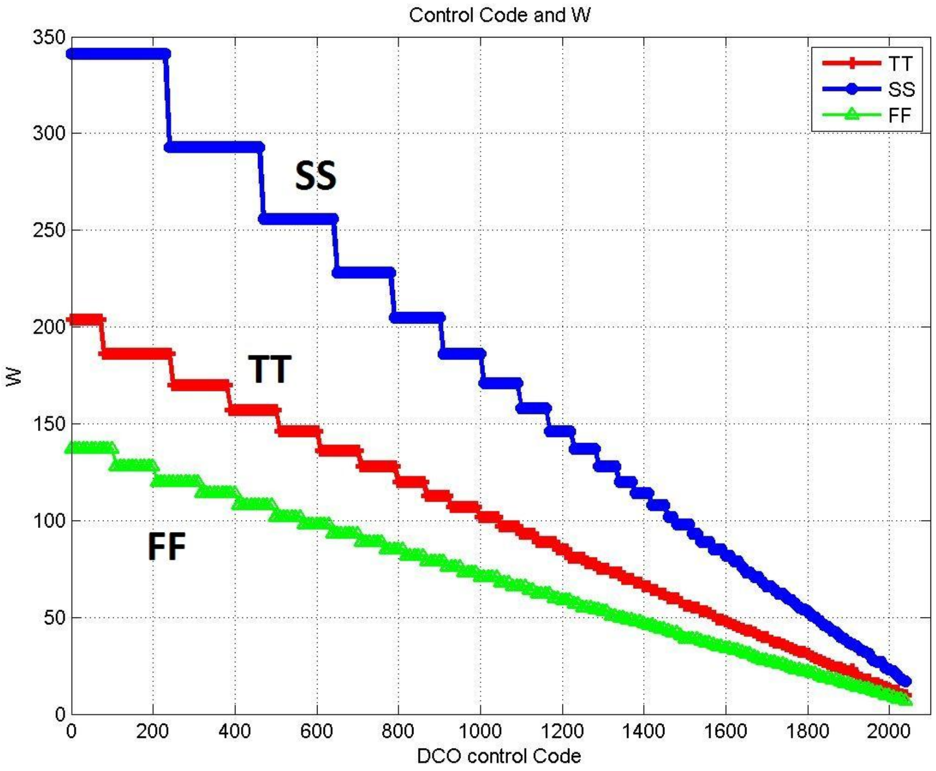


Fig. 2.4: The Relationship of the DCO Control Code and the *W* Value.

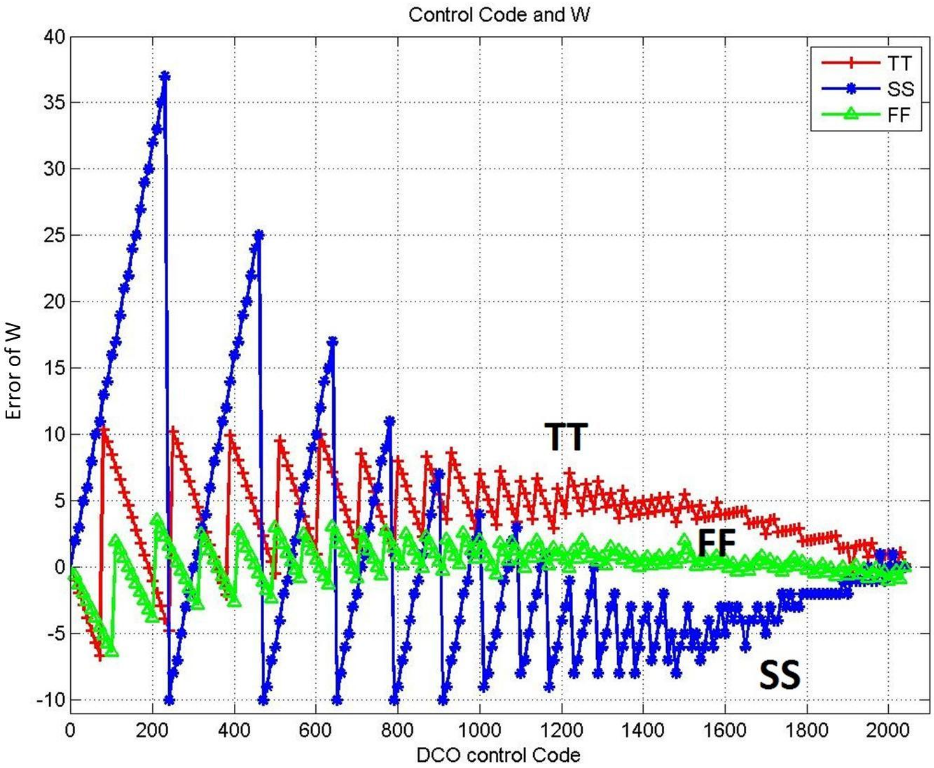


Fig. 2.5: The Quantization Error of the *W* Value without TDC.

According to the simulation in Fig. 2.4, we use the minimum value and maximum value of the *W* value to create a straight line, and then the quantization error is shown in Fig. 2.5. Therefore, the quantization error is small at the FF corner or with the larger DCO control code as shown in Fig. 2.5. In other words, while the DCO operates at the higher frequency, the quantization error to compute the integer *R* value is gradually decreased. As a result, we decide to exploit the medium and maximum frequency of the DCO as both ends of the line, so that can minimize the quantization error.

However only utilize the medium and maximum frequency of the DCO is not accurate enough, we provide a calibration method to reduce the quantization error of the *W* line. We define three parameters as follow.

T_0 : The minimum clock period of the DCO.

Δ : Fine-tuning resolution.

T : The period of the reference clock.

First, we explain the R value in another way. As we mention before, the R value is the period ratio between the DCO clock and the reference clock. The R_{max} is the number of how many minimum DCO clock periods in a reference clock period. We can describe R_{max} in the mathematical formula as Eq. 2.6. In this thesis, we only quantize one half cycle of the DCO minimum clock period and denoted as R_{max}. Then R_{mid} can be described as Eq. 2.7, and R_{mid} is the period ratio when the DCO control code is set to 1023.

$$R_{max} = \frac{\frac{T}{2}}{T_0} \tag{2.6}$$

$$R_{mid} = \frac{\frac{T}{2}}{T_0+1023\varDelta} \tag{2.7}$$

According to Eq. 2.6 and Eq. 2.7, we express the Δ and T_0 in terms of R_{mid} and R_{max}. First, Eq. 2.6 is derives Eq. 2.8. Then T_0 is expressed in terms of R_{max} and T. According to Eq. 2.7, we substitute Eq. 2.8 into Eq. 2.7 and derive Eq. 2.9. Subsequently, from Eqs. 2.9 to 2.13, the fine-tuning resolution (Δ) can be derived in terms of T, R_{mid} and R_{max}.

$$T_0 = \frac{\frac{T}{2}}{R_{max}} \tag{2.8}$$

$$R_{mid} = \frac{\frac{T}{2}}{\frac{{}^{T}/_{2}}{R_{max}}+1023\varDelta} \tag{2.9}$$

$$\frac{\frac{T}{2}}{R_{max}} + 1023\varDelta = \frac{\frac{T}{2}}{R_{mid}} \tag{2.10}$$

$$1023\varDelta = \frac{\frac{T}{2}}{R_{mid}} - \frac{\frac{T}{2}}{R_{max}} \tag{2.11}$$

$$1023\mathit{\Delta} = \frac{\frac{T}{2}\times(R_{max}-R_{mid})}{R_{mid}\times R_{max}} \tag{2.12}$$

$$\Delta= \frac{\frac{T}{2}\times(R_{max}- R_{mid})}{1023 \times R_{max} \times R_{mid}} \tag{2.13}$$

The general formula of the period ratio R (DCO_code) can be expressed in Eq. 2.14. T_0 in Eq. 2.14 can be substituted by Eq. 2.8, and Δ can be substituted by Eq. 2.13, and then Eq. 2.15 can be derived. Subsequently, Eq. 2.15 and 2.16 are derived.

$$R(DCO\_code) = \frac{\frac{T}{2}}{T_0 + DCO\_code\times\mathit{\Delta}} \tag{2.14}$$

$$R(DCO\_code)= \frac{\frac{T}{2}}{\frac{\frac{T}{2}}{R_{max}} + DCO\_code\times\frac{\frac{T}{2}\times(R_{max}-R_{mid})}{1023\times R_{max}\times R_{mid}}} \tag{2.15}$$

$$R(DCO\_code) = \frac{1023\times R_{max}\times R_{mid}}{1023\times R_{mid}+DCO\_code\times(R_{max}-R_{mid})} \tag{2.16}$$

Since *W(DCO_CODE)* is the reciprocal of $R(DCO\_code)$, and Eq. 2.17 can be easily derived. Eq. 2.17 shows the compensated *W* lines equation. Fig. 2.6 shows the DCO control code is set to 0 to 1200 because of a large quantization error in this portion. Obviously, the trapezoid line (Simulation W) is now close to the ideal *W* line (Conpensation W). As a result, we obtain a relation between *W* value and the DCO control code precisely by the calibration method. We can estimate the desire DCO control code with the linear equation expressed in Eq. 2.17.

$$W_{DCO\_code} = \frac{1023\times R_{mid}+DCO\_code\times(R_{max}-R_{mid})}{1023\times R_{max}\times R_{mid}}\times 2^{11} \tag{2.17}$$

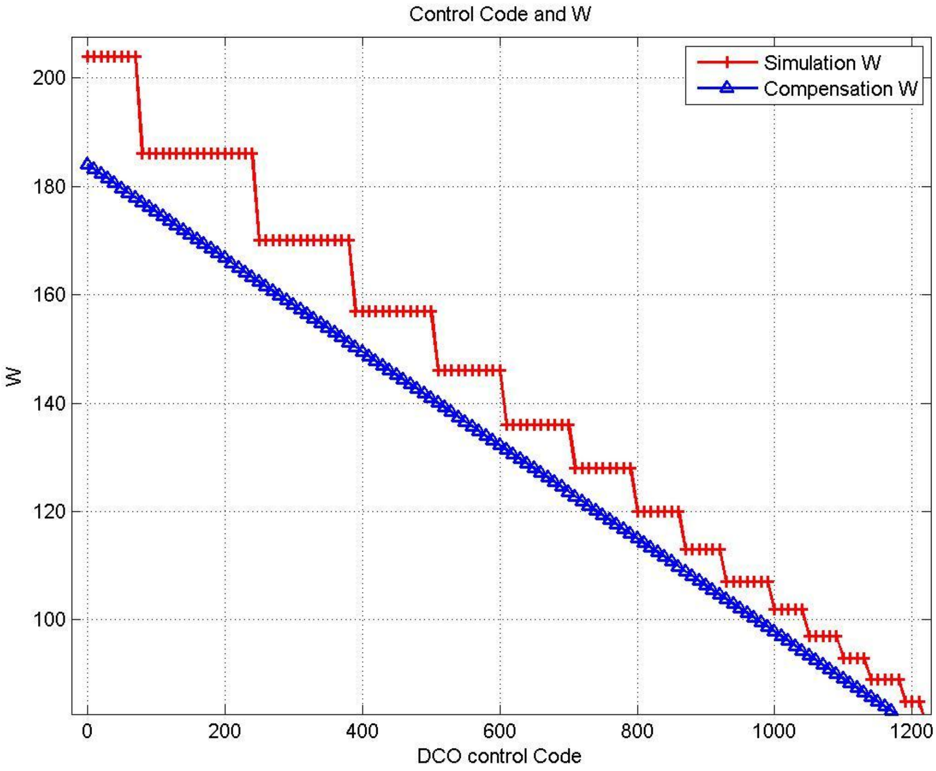


Fig. 2.6: The Relationship Between the Trapezoid Simulation *W* and the Compensate *W* line.

$$L_{mid} = \frac{1023\times R_{mid}+1023\times(R_{max}-R_{mid})}{1023\times R_{max}\times R_{mid}} \times 2^{11} \qquad (2.18)$$

$$L_{mid} = W_{min} + \frac{R_{max}-R_{mid}}{R_{max}\times R_{mid}} \times 2^{11} \qquad (2.19)$$

Consequently, we set DCO_code to 1023 into Eq. 2.17 and calculate the calibration value that is defined as L_{mid} and expressed in Eq. 2.18. After simplifying Eq. 2.18, we obtain Eq. 2.19 which is easy to be implemented by hardware. From W_{min} and L_{mid} of the compensation W line, the equation of the line can be expressed in Eq. 2.20. The frequency multiplication factor of the ADPLL is the ratio between the reference clock period and the target DCO clock period. Hence the target period ratio (R_T) is equal to the frequency multiplication factor (N); and thus W_T is equal to 1/N. Therefore, we can estimate the target DCO control code (*init_code*) by Eq. 2.20.

$$init\_code = 2^{10} \times \frac{W_T - W_{min}}{L_{mid} - W_{min}} \tag{2.20}$$

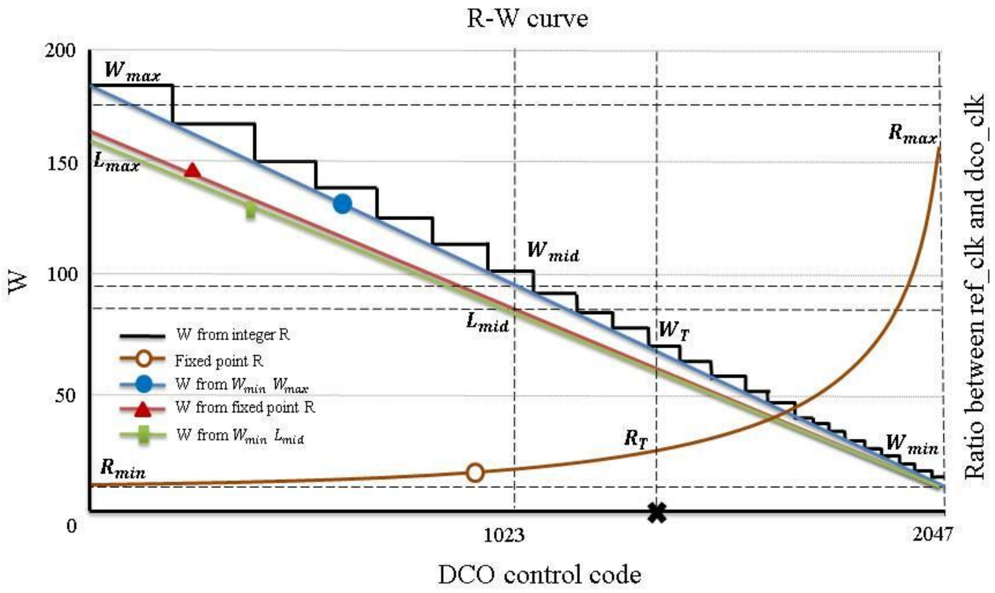


Fig. 2.7: The *W* Value Before Calibration and After Calibration.

We proposed a calibration method to eliminate the cyclic TDC in [10] and calibrate the W_{mid} to the L_{mid} as shown in Fig. 2.7. The L_{mid} is lower than the W_{mid} due to the integer *R* value. The reciprocal of the fixed-point *R* value is less than the reciprocal of the *R* integer. The proposed calibration method compensates the quantization error of *W* line. Furthermore, by adjusting the W_{mid} to the L_{mid}, we improve the accuracy of the FEA. Therefore, we can achieve a fast lock-in ADPLL precisely with the frequency estimation algorithm and the calibration method.

As shown in Fig. 2.2, in the first cycle, after reset, we compute the reciprocal of the frequency multiplication factor (N) into the W_T. Subsequently, the R_{mid} is calculated by the cyclic counter; the W_{mid} is calculated by the frequency counter in the same cycle (R_mid state). Consequently, the R_{max} and the W_{min} are calculated (R_max state). After we obtain the R_{max} and the W_{min}, we start to compensate the quantization error and generate a calibration value L_{mid} in the third cycle (Calibration state). Then, the ADPLL controller can calculate the target DCO control code by Eq. 2.20 (Calculation state). Finally, the ADPLL controller can output the target DCO control code after 4.5 cycles.

2.3 Summary

In Fig. 2.8, the frequency track of the ADPLL is shown, and the target frequency is 260MHz. In the first 4.5 cycles, the proposed ADPLL estimates the frequency closed to the target frequency. Subsequently, the binary search algorithm and the loop filter are applied to fine-tune the output frequency and keep maintaining the frequency and phase of the output clock. Therefore, after 4.5 cycles, the lock signal is pulled up, and the proposed ADPLL can provide the stable frequency output during fine-tuning.

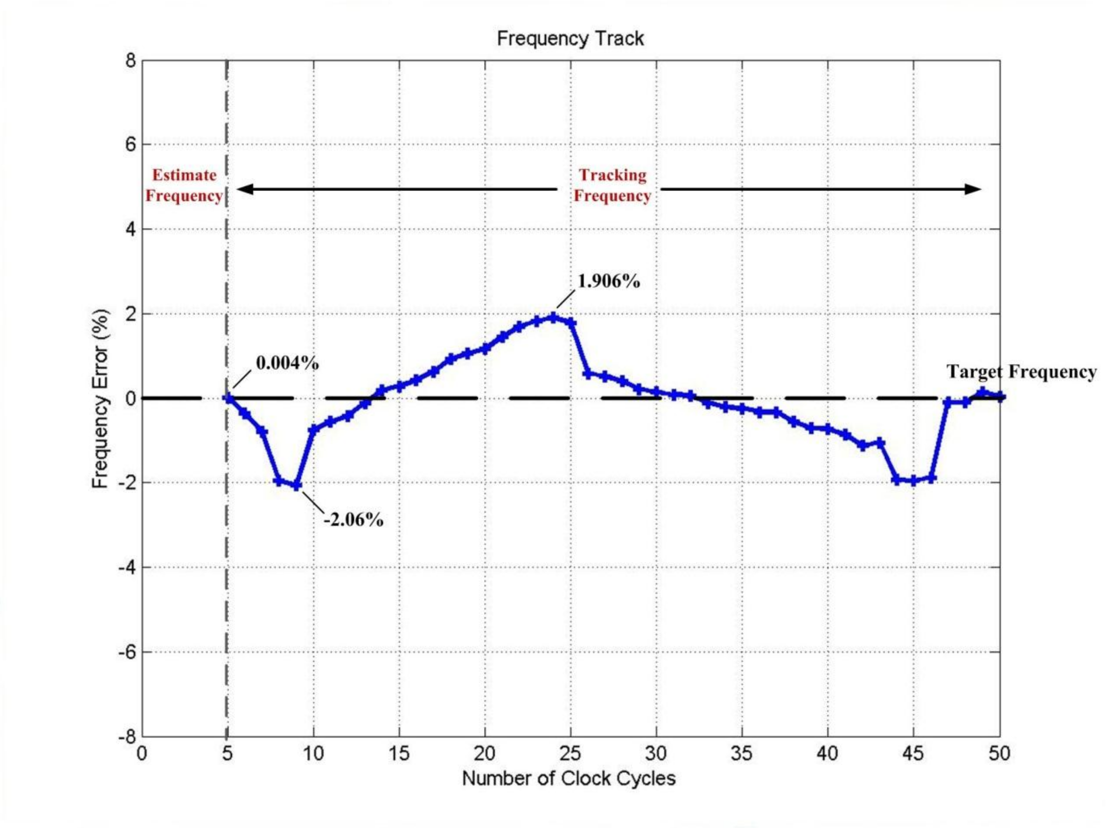


Fig. 2.8: The Frequency Analysis at 260MHz.

As compares to [4] and [10], the proposed the calibration method based frequency estimation algorithm has a better hardware utilization and a precise prediction of the target frequency. Comparing with [4], although it can achieve fast lock within two cycles, the quantization leads to a large estimation error after two cycles. Besides three DCOs also cause the on-chip variation, it brings out each independent variable into the frequency estimation algorithm which requires relevant values.

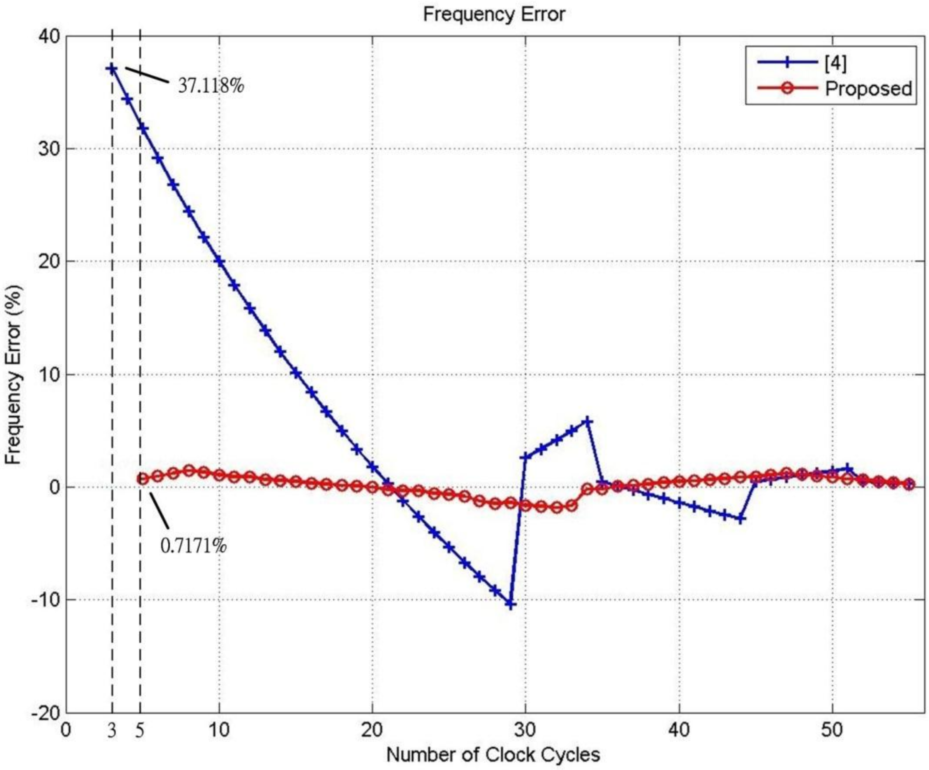


Fig. 2.9: The frequency error Analysis with 150MHz.

We compare the frequency error between [4] and the proposed ADPLL as shown in Fig. 2.9. We set the multiplication factor in 30 and the target frequency is 150MHz. Obviously, after two cycles, [4] still remains a large frequency error which is almost 40%. Due to the large frequency error, [4] needs to track the target frequency with the binary search algorithm. Thus, actually it takes more than 50 reference cycles to achieve lock. However the proposed ADPLL requires 4.5 cycles to estimate the target frequency, the frequency error keeps under 1% after the 5th reference clock cycle.

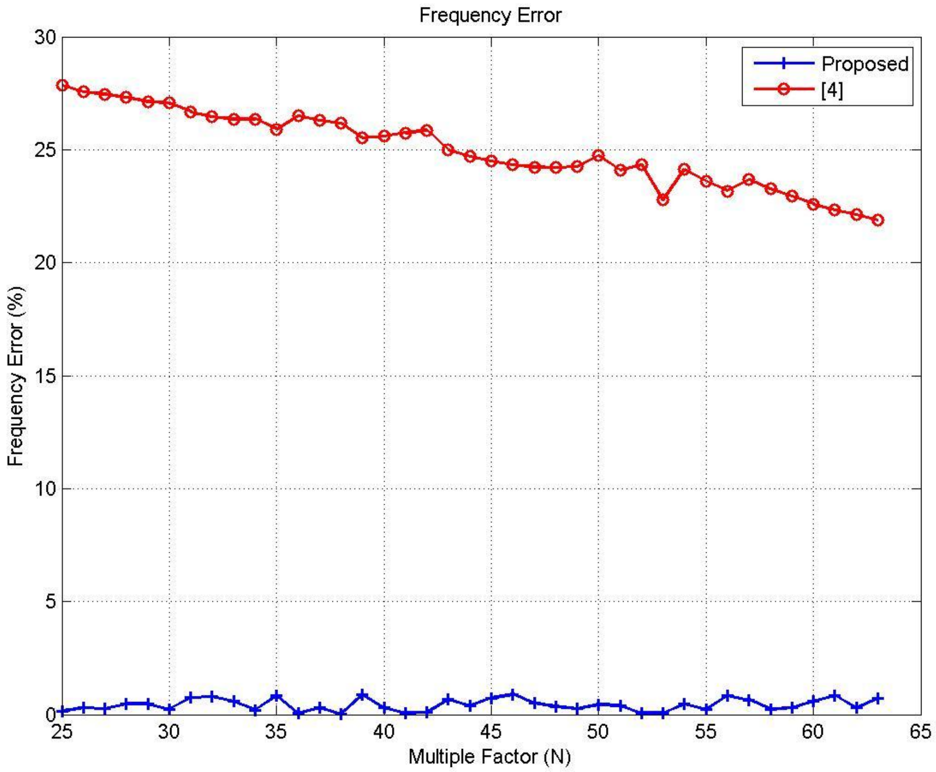


Fig. 2.10: The initial frequency error between the proposed ADPLL and [4].

In Fig. 2.10, after estimating the target frequency, we compare the frequency error between [4] and the proposed ADPLL at the fifth reference clock with different N value. The frequency error of [4] is very large after the frequency estimation. However, while the multiplication ratio increases, the frequency error can be decreased.

In [10], a cyclic TDC is applied to minimize the quantization error, it improves the accuracy of frequency estimation indeed. It utilizes a DCO to avoid on-chip variations in [4]. However, the cyclic TDC [10] occupies large chip area and the efficiency of hardware utilization is not satisfied. In Fig. 2.11, this figure shows the frequency error in different multiple factors after the frequency estimation. We compare the proposed ADPLL with [10] with a 5MHz reference clock. Although [10] solves the issue in [4], the proposed ADPLL further increases the utilization of the hardware and improves the initial frequency error under 1%.

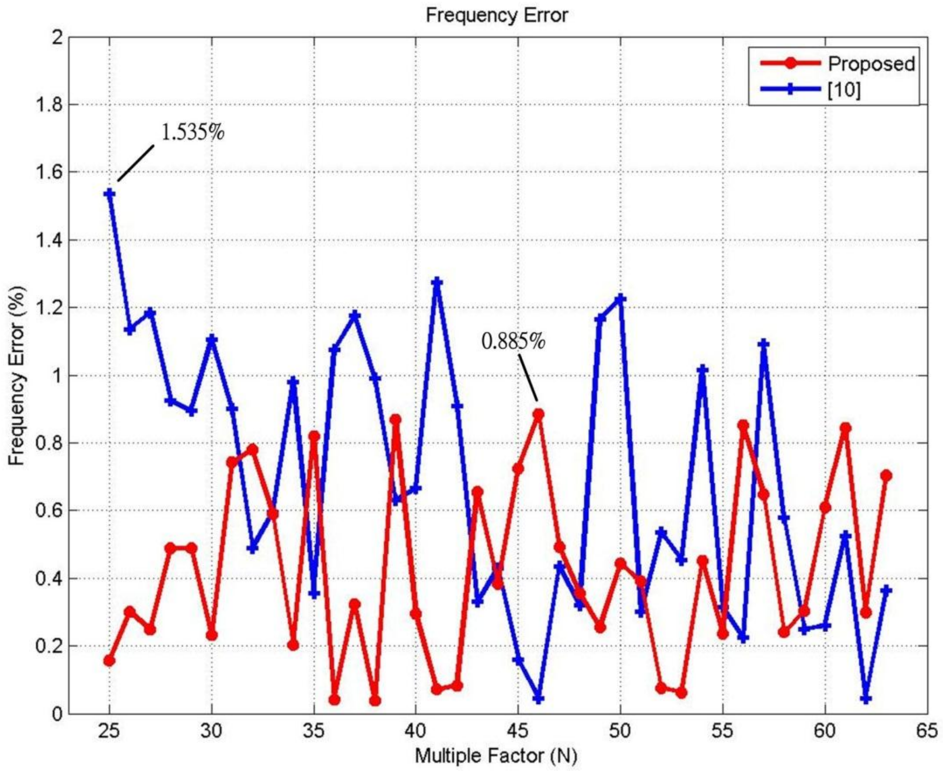


Fig. 2.11: The initial frequency error between proposed and [10].

Chapter 3

Circuit Design and Implementation of ADPLL

In this chapter, we describe the circuit design in the proposed fast lock-in ADPLL in 40-nm CMOS process that contains a monotonic DCO, a digital loop filter, a phase and frequency detector and a frequency detector.

3.1 Monotonic Digital Controlled Oscillator Design

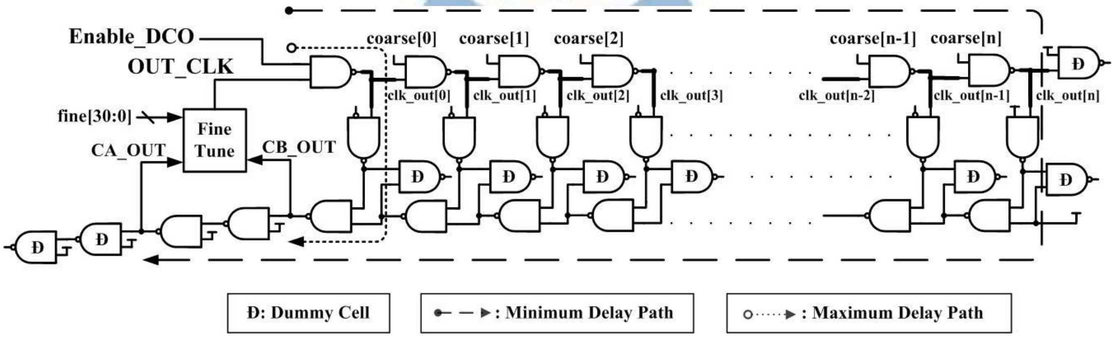


Fig. 3.1: The Proposed DCO.

In Fig. 3.1, the architecture of the monotonic DCO is shown. The proposed DCO is composed of 63 coarse-tuning stages [10] and 31 fine-tune stages [12].

3.1.1 Coarse-Tuning Stage

The coarse-tuning stages are applied in NAND-based architecture [10] as shown in Fig. 3.2. Four NAND gates constitute a coarse-tuning stage. The 1st NAND gate is operated like a switch that controls this coarse-tuning stage to be on or off. The 2nd and the 3rd NAND gate provide the delay time, thus, the coarse-tuning resolution is dependent on the 2nd and the 3rd NAND gates. The 4th NAND gate is used to balance the rise time and the full time due to that each input node has equal capacitance loading. While the 4th NAND gate balances the coarse-tuning rise time and full time, the duty cycle will not be distorted by the unbalanced rise time and full time, as shown in table 3.1.

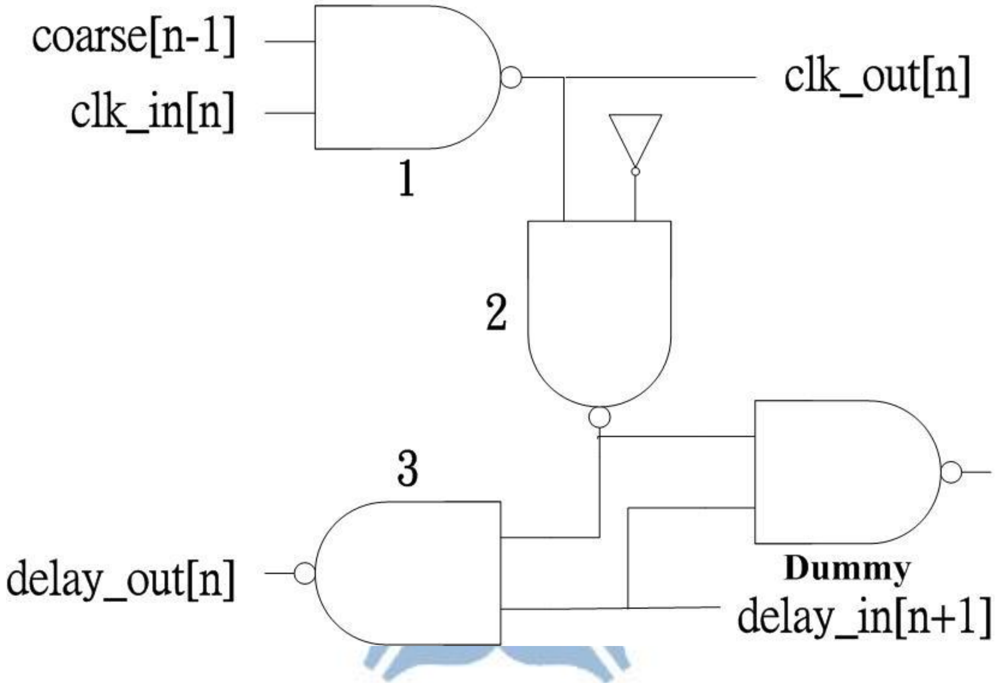


Fig. 3.2: The Proposed Coarse-Tuning Stage.

Table 3.1: The DCO Duty Cycle in 40-nm CMOS Process.

Parameter	129MHz	1.47GHz
Duty Cycle	51.11%	47.27%

The coarse-tuning stage is controlled by the coarse control code (coarse[n]). We assume that the coarse control code is 63’hffff_ffff_ffff_fffc (coarse [0] ~ coarse [1] = 0, coarse [2] ~ coarse [62] = 1) as shown in Fig. 3.3, the delay path pass through the group of coarse [2]. Therefore, as the delay cells are unused, they can be gated by the coarse control code that can reduce the dynamic power consumption of the DCO.

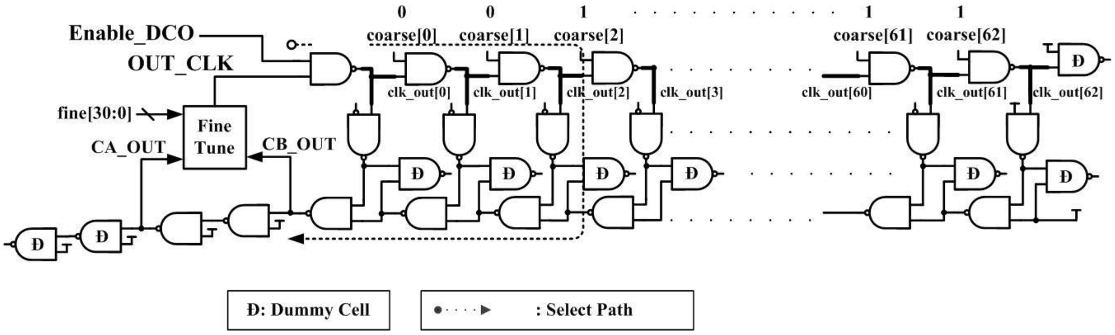


Fig. 3.3: The Coarse-Tuning Stage Delay Path.

The coarse-tuning stage structure has high linearity between the coarse control code and the output period due to the regular placement approach [32]. In Fig.3.1, the minimum delay path is the shortest delay path of the coarse-tuning stage; the maximum delay path is the longest delay path of the coarse-tuning stage. As a result, the proposed coarse-tuning stages can provide a wide frequency range in various coarse control codes.

3.1.2 Fine-Tuning Stage

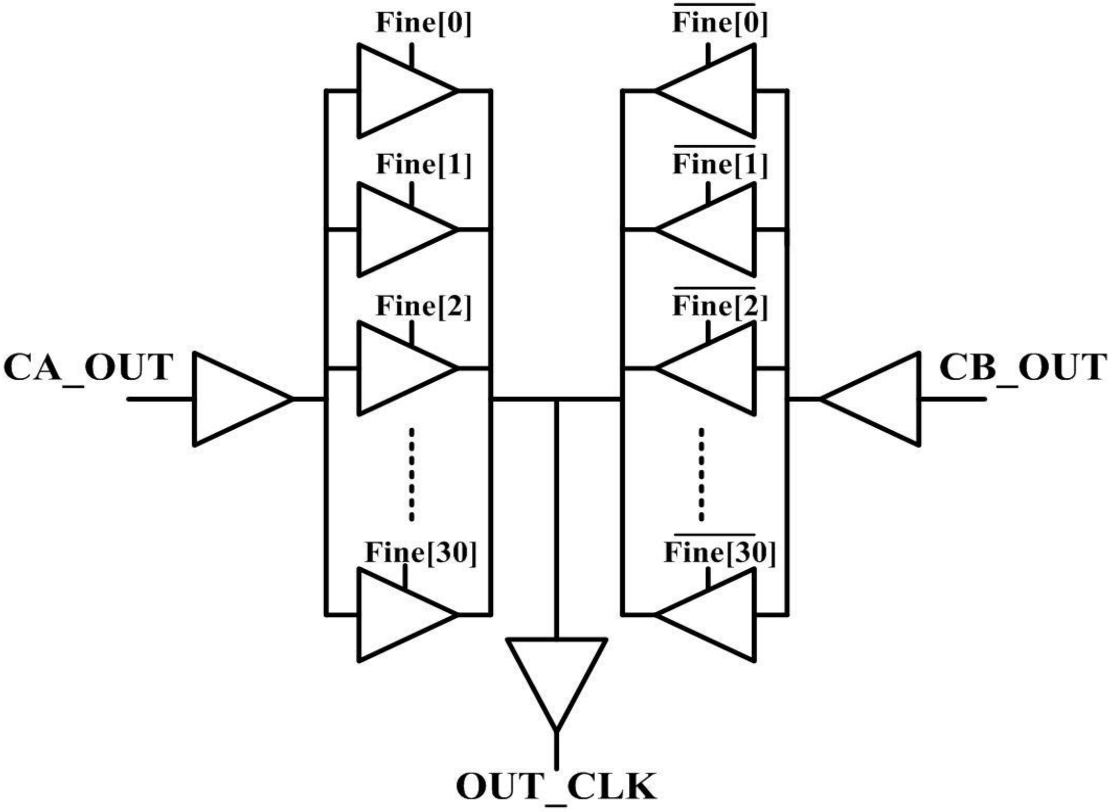

Fine[0]
$\overline{\text{Fine}[0]}$
Fine[1]
$\overline{\text{Fine}[1]}$
Fine[2]
$\overline{\text{Fine}[2]}$
CA_OUT
CB_OUT
Fine[30]
$\overline{\text{Fine}[30]}$
OUT_CLK

Fig. 3.4: The Fine-Tuning Stage of the Proposed DCO.

The fine-tuning stage architecture of the DCO is shown in Fig. 3.4. The fine-tuning stage is composed of two parallel connected tri-state buffer arrays [12]. The two parallel connected tri-state buffer arrays operate as an interpolator circuit which is controlled by the fine control code. We use 31 tri-state buffers to interpolate the signal CA_OUT and the signal CB_OUT. The different between the CA_OUT and the CB_OUT is a coarse-tuning stage. Utilizing the fine control code (Fine[n]), the total controllable range is always equal to a coarse-tuning stage.

The traditional fine-tuning stage of the DCO requires overlapping the coarse-tuning stage. As the PVT variation affects the DCO, the traditional fine-tuning stage without overlap may lose some frequencies. Besides, while switching the coarse control code, the cycle-to-cycle jitter of the DCO becomes worse. Therefore, we utilize the two parallel connected tri-state buffer arrays as the fine-tuning stage not only improves the resolution of the DCO, but also ensure the total fine-tuning range is a coarse-tuning stage.

The principle of the two parallel connected tri-state buffer arrays is utilized the driving strength of the tri-state buffer arrays to generate various delay time. While the more left-hand side tri-state buffers are turned on, the output clock (OUT_CLK) is closer to the CA_OUT. Conversely, while the more right-hand side tri-state buffers are turned on, the output clock is closer to the CA_OUT. We suppose that the fine control code is 5’d29, the fine control code should encode as 31’h7fff_ffff (Fine [0] ~ Fine [28] = 1, Fine [29] ~ Fine [30] = 0). The delay time between the CA_OUT and the CB_OUT can be considered as divided into 31 portions as shown in Fig.3.5. Therefore, the output clock is more close to the signal CA_OUT.

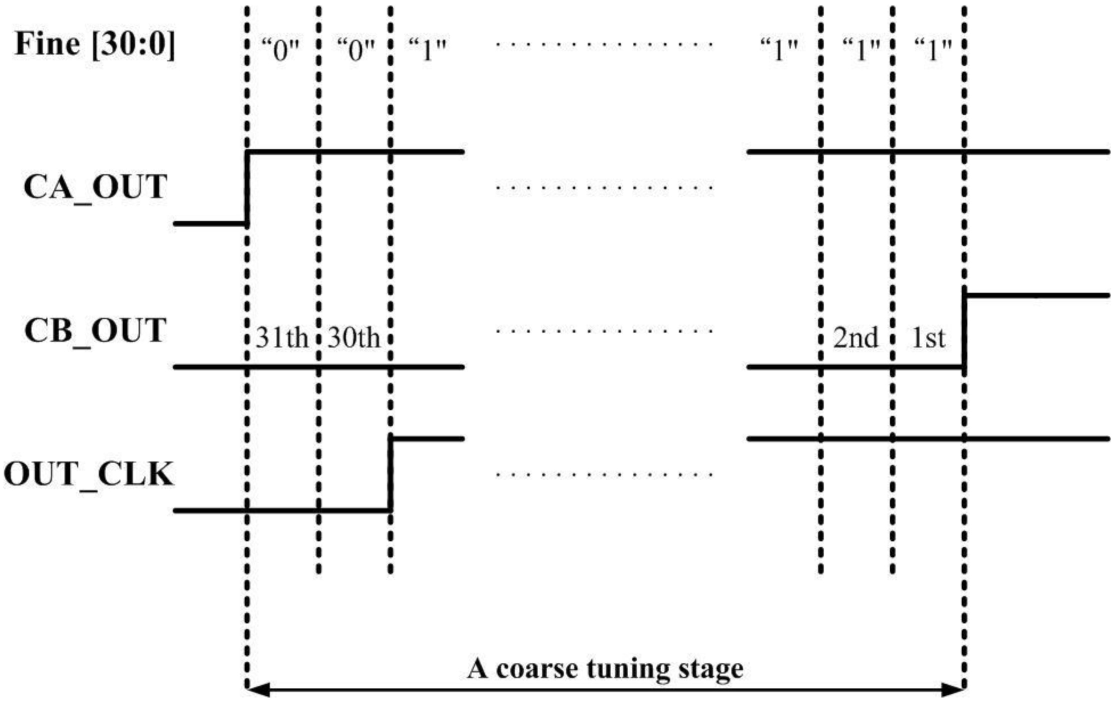


Fig. 3.5: The Timing Diagram of the Fine-Tuning Interpolation Signal Illustration.

Besides, the DCO control code has 11 bits. A DCO Encoder encodes the 11 bits DCO control code into the coarse-tuning and fine-tuning control thermometer codes. The dco_code [10:5] and the dco_code [4:0] encodes into coarse [62:0] and fine [31:0], respectively.

Furthermore, the proposed DCO can operate in a wide range during low voltage. The frequency range of various operating voltage in 40-nm is shown in table 3.2. The normal operating voltage in 40-nm is 0.9V. As we demand for low operating voltage to decrease power consumption, the proposed DCO can still operate in wide frequency range. However, the results are the pre-layout simulation. As the process is scaling down to the deep submicron CMOS process, the interconnection delay has great impact on the total delay. Therefore, in pre-layout simulation, the DCO frequency range is not accurate enough. After automatic placement and routing (APR) is done, delay cells are connected by wires. Subsequently, the wire delay is considered in post-layout simulation, we should depend on the post-layout simulation to analyze the DCO frequency range.

For example, in the pre-layout simulation at 0.9V supply voltage, the DCO maximum frequency is adjusted at 5.9GHz, after APR, the DCO maximum frequency in post-layout simulation is only 2GHz. In the pre-layout simulation at 0.5V supply voltage, the DCO maximum frequency is adjusted at 857MHz, after APR, the DCO maximum frequency is 264MHz.

As a result, the proposed DCO with the fine-tuning stage and the coarse-tuning stage can achieve wide frequency range and monotonic response in the output clock versus the DCO control code. In addition the proposed DCO minimizes the jitter during the coarse control codes switching.

Table 3.2: The DCO Frequency Range in Different Operating Voltage in Pre-Layout Simulation and Post-Layout Simulation

Operating Voltage (V)		0.5	0.6	0.7	0.8	0.9
Frequency Range (MHz)	Pre-Layout Simulation	65 ~ 857	142 ~ 1869	238 ~ 3154	341 ~ 4566	441 ~5896
	Post-Layout Simulation	33 ~ 264	76 ~ 606	134 ~ 1123	200 ~ 1519	256 ~2024

3.1.3 The Proposed Regular Placement Method

As the process technology is scaling down to the deep submicron CMOS process, the interconnection delay has great impact on the total delay connection. Therefore, the relative locations between delay cells during the automatic placement and routing (APR) are controlled by the regular placement approach [32].

The wire lengths between the delay cells of the DCO are determined by the APR tools. Once the area constraints and the cell placement results are changed, the routing results will be changed accordingly. Hence, the DCO regular placement approach is presented to avoid the large wire length variations in different DCO placement cases.

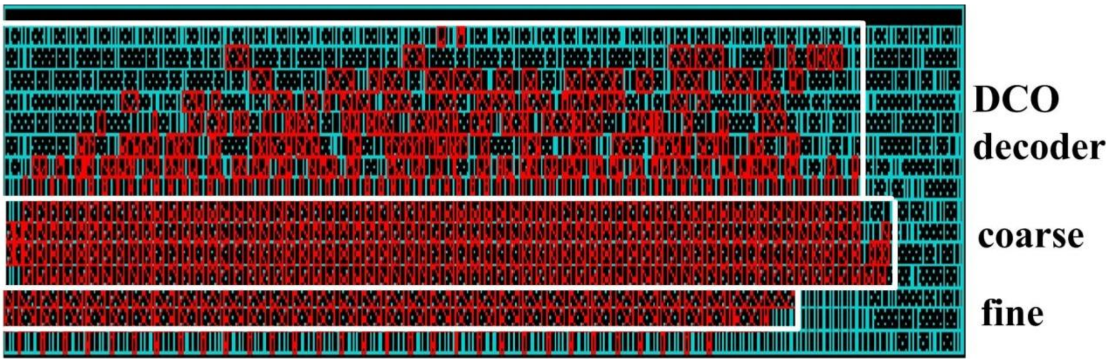


Fig. 3.6: The Regular Placement of the DCO

Fig. 3.6 shows the layout of the DCO including the coarse-tuning stages and the fine-tuning stages. When the coarse-tuning stages are placed closer, the wire length between the pins of the delay cells becomes shorter. Besides, the wire length between two neighboring coarse-tuning stages can be controlled almost the same which also improves the differential nonlinearity (DNL) of the coarse-tuning stage.

Fig. 3.7 shows the DNL comparison results between the regular placement and random placement by the APR tools in 150 × 80 μm^2. In addition, the serrated edge of the DNL results is due to well taps and stripes. The maximum DNL of the proposed DCO regular placement is lower than 0.3 LSB and higher than -0.2 LSB. Therefore, the proposed DCO has a monotonic response.

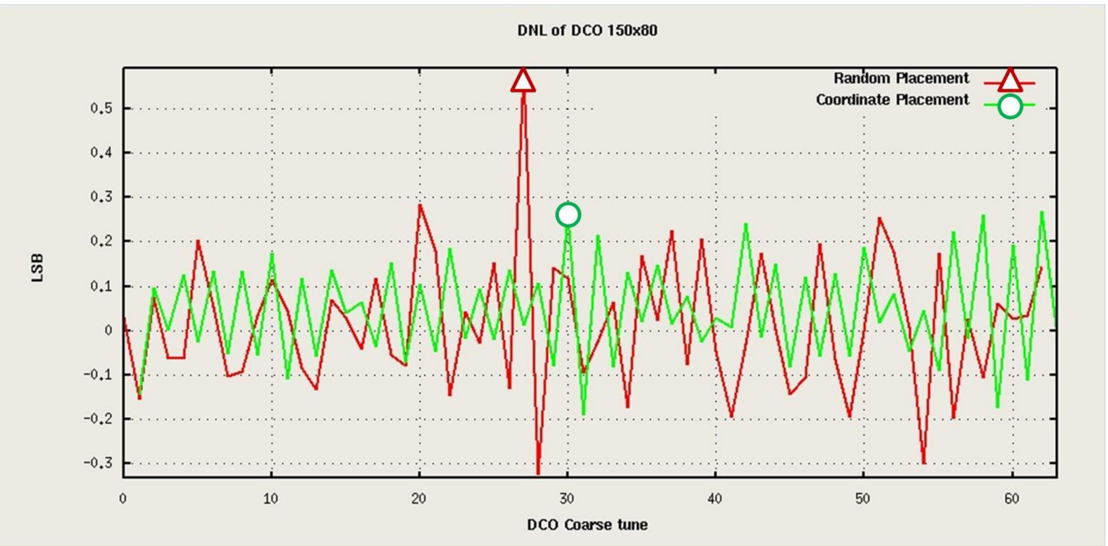


Fig. 3.7: DNL Comparison of the DCO in 150 × 80 μm^2.

3.2 Phase and Frequency Detector

We adopt the architecture of the cell-based bang-bang phase and frequency detector (PFD) which is proposed in [11]. The PFD outputs the UP signals or the DN signals between the reference clock (REF_CLK) and the feedback clock (FB_CLK). In Fig. 3.8, the proposed PFD consists of buffers, inverters, NAND gates and DFFs which depends the dead zone of the PFD. The dead zone is the smallest detectable phase error of the PFD. As we compare with the phase detector, the phase detector can only detect the phase error. According to the continuous UP signals or DN signals, the PFD outputs to the ADPLL controller to adjust the DCO output frequency.

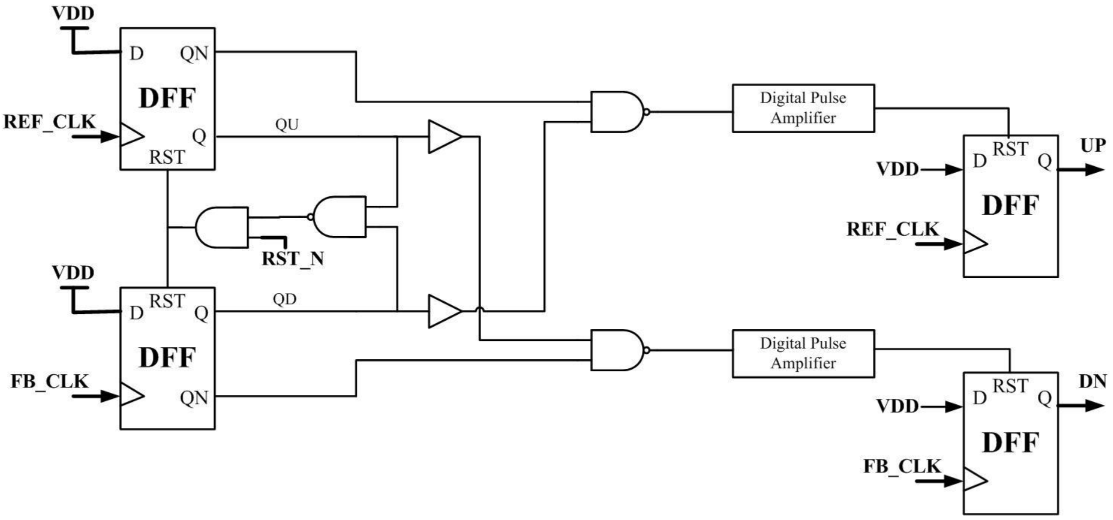


Fig. 3.8: The Proposed phase and frequency detector.

However, the digital pulse amplifier [11] is the indispensable part in promoting the performance of the PFD. If the phase error between the reference clock and the feedback clock is very small, the narrow pulse width is not able to trigger the DFF reset pin (RST). Therefore, the digital pulse amplifier extends the phase error between the reference clock and the feedback clock so that the pulse width is wide enough to reset the DFF. The architecture of the digital phase amplifier is shown in Fig. 3.9. It utilizes the cascade two-input AND gates architecture to increase the narrow pulse width for RST pin.

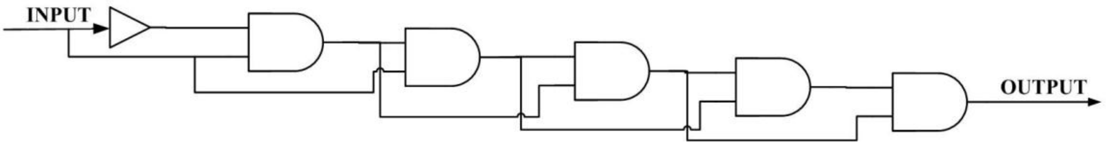


Fig. 3.9: The Digital Pulse Amplifier Architecture.

As a result, the proposed PFD is able to output the UP signals and the DN signals continuously. It can not only detect the phase error, but also the frequency error. Besides, in 40-nm, the dead zone of the PFD is 17ps so that it can detect the slight phase error which the reference clock is very close to feedback clock.

3.3 Test Chip Implementation

The proposed frequency divider is designed for chip measurement at high frequency operation. Due to the clock rate on I/O pads, signals operate at high frequency are unable to transmit through I/O pads in TSMC 40nm cell-library.

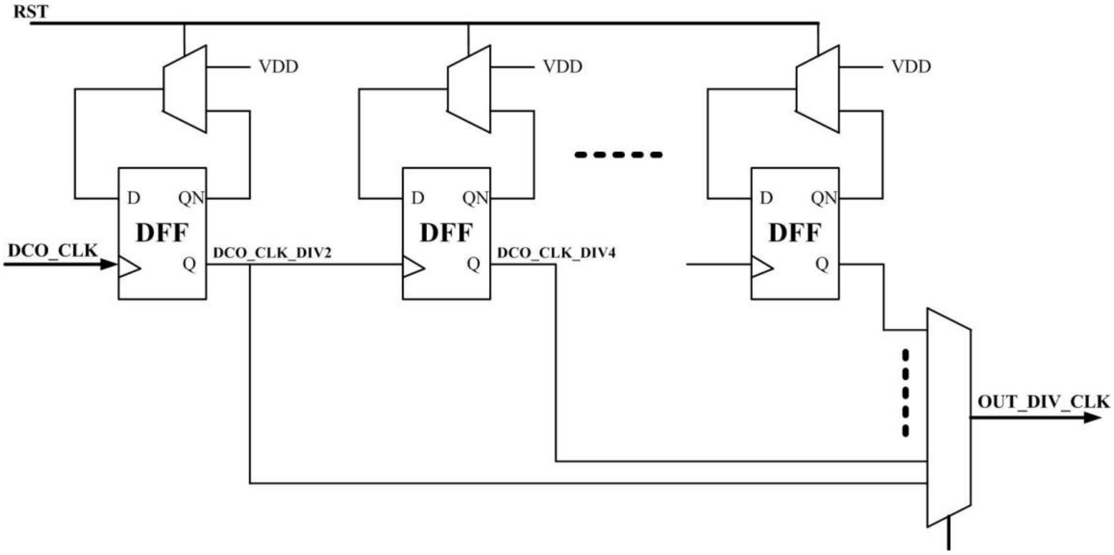


Fig. 3.10: Frequency Divider Architecture.

In Fig. 3.10, the frequency divider is adopted the cascaded divide-by-2 frequency divider architecture. The proposed frequency divider is consisted of multiplexers and four DFFs. The multiplexers are used to select the signals in various division ratios. The DFFs are asynchronous binary counter clocked by the input signal (DCO_CLK). The frequency of the DCO_CLK_DIV2 signal is half of the DCO_CLK frequency and so on the DCO_CLK_DIV4 signal is a quarter of the DCO_CLK frequency. Each stage divides signals by 2. Therefore, the proposed frequency divider can divide the DCO_CLK signal to the OUT_DIV_CLK signal by 2^m, where m is the stage number.

The feature of the cascaded divide-by-2 frequency divider is that it can deal with a very fast signal input. However, the cascaded divide-by-2 frequency divider can only be used for power-of-2 integer division.

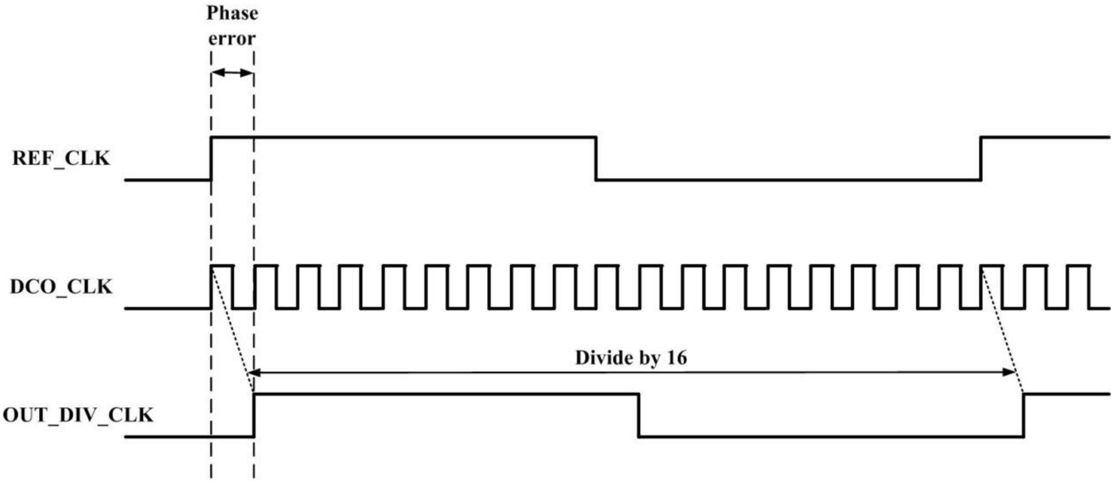


Fig. 3.11: The Phase Error between REF_CLK and OUT_DIV_CLK Illustration.

In the standard cell library, due to the worse transition time and rise delay of the DFFs, the frequency divider produces a clock skew between the REF_CLK and the OUT_DIV_CLK. Fig. 3.11 shows the phase error between the REF_CLK and the OUT_DIV_CLK.

3.4 Digital Loop Filter

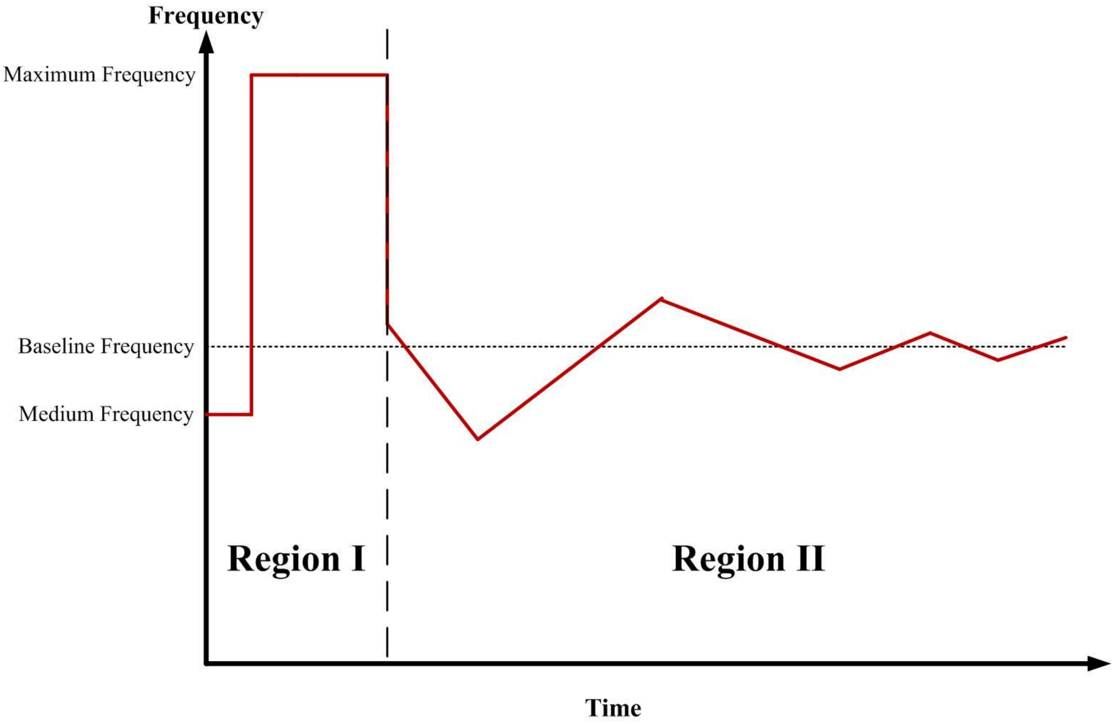


Fig. 3.12: ADPLL Frequency Tracking Procedure.

The ADPLL continues tracking the frequency and the phase of the reference clock by the DCO control code. The digital loop filter [19] [43] is required to stabilize the DCO control code and maintain the output frequency. Fig. 3.12 describes the tracking process of the ADPLL. In Region I, the ADPLL estimates the target frequency. After entering Region II, the output frequency of the DCO is close to the target frequency. However, it still has small frequency error and phase error. When the ADPLL suffers from the reference clock jitter or supply noise, the output frequency of the DCO becomes unstable in Region II. Therefore, the digital loop filter can overcome the reference clock jitter or supply noise and stabilize the loop.

The digital loop filter structure [43] is shown in Fig. 3.13. The digital loop filter obtains the DCO control code from the ADPLL controller output. After the target DCO control code sends to the DCO, the digital loop filter stores the DCO control code in each reference cycle. In addition, the digital loop filter stores eight DCO control codes to generate a baseline DCO control code. If eight DCO control codes are already stored in the digital loop filter, an ff_init_ok signal is triggered to generate the baseline DCO control code. Subsequently, every two new DCO control codes are sent to the digital loop filter, the digital loop filter sorts the DCO control codes. Consequently, the digital loop filter removes both the minimum and the maximum stored DCO control codes, and the digital loop filter generates the baseline DCO control code by averaging the rest of the DCO control codes ($C_0, C_1, \ldots, C_7$). Therefore, the baseline DCO control code is updated by the digital loop filter. As a result, the proposed digital loop filter updates the baseline DCO control code in a short time and stabilizes the output frequency of the DCO.

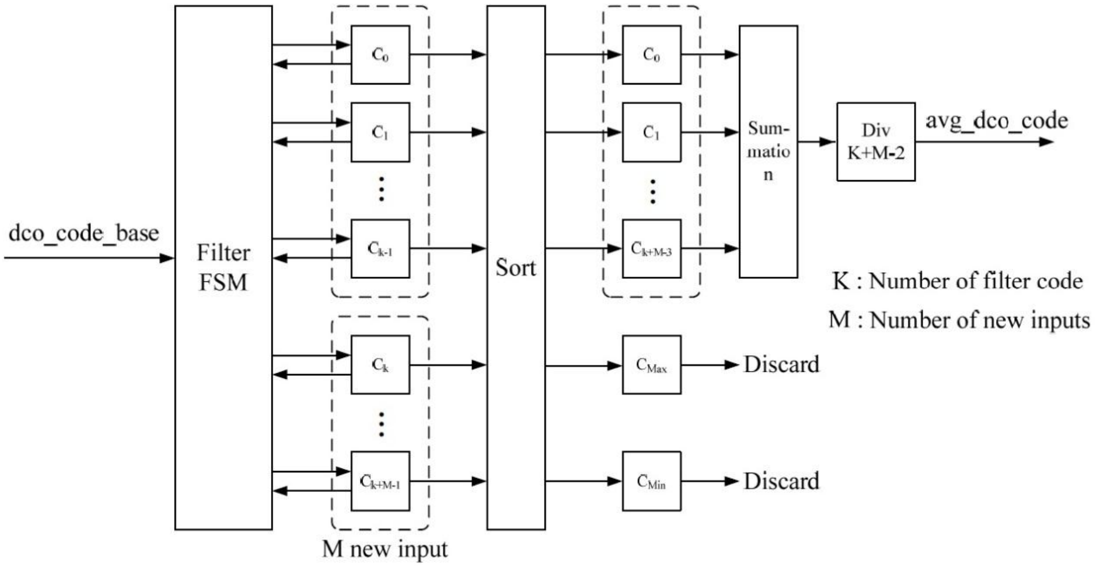
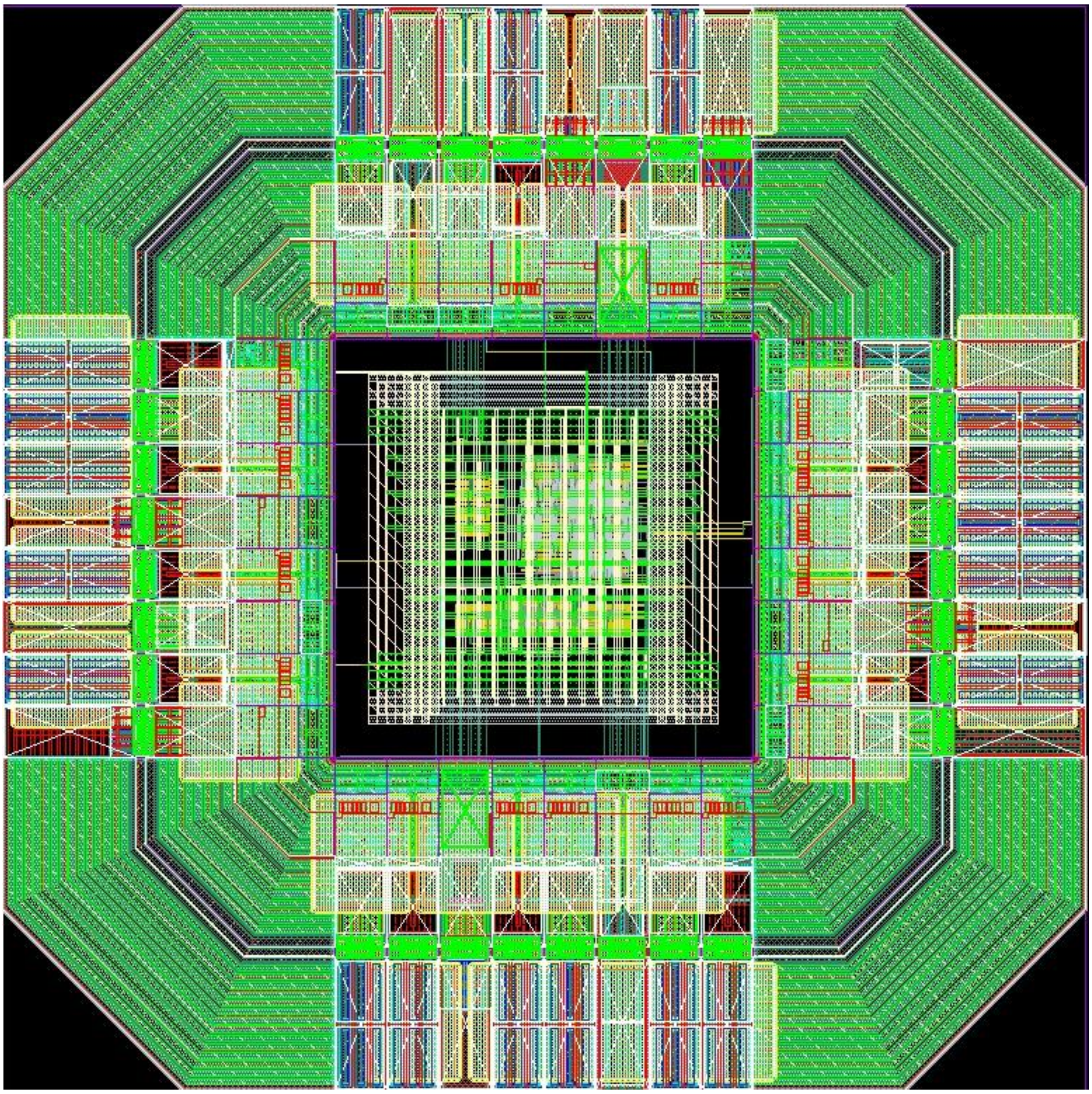


Fig. 3.13: The Digital Loop Filter Structure [43].

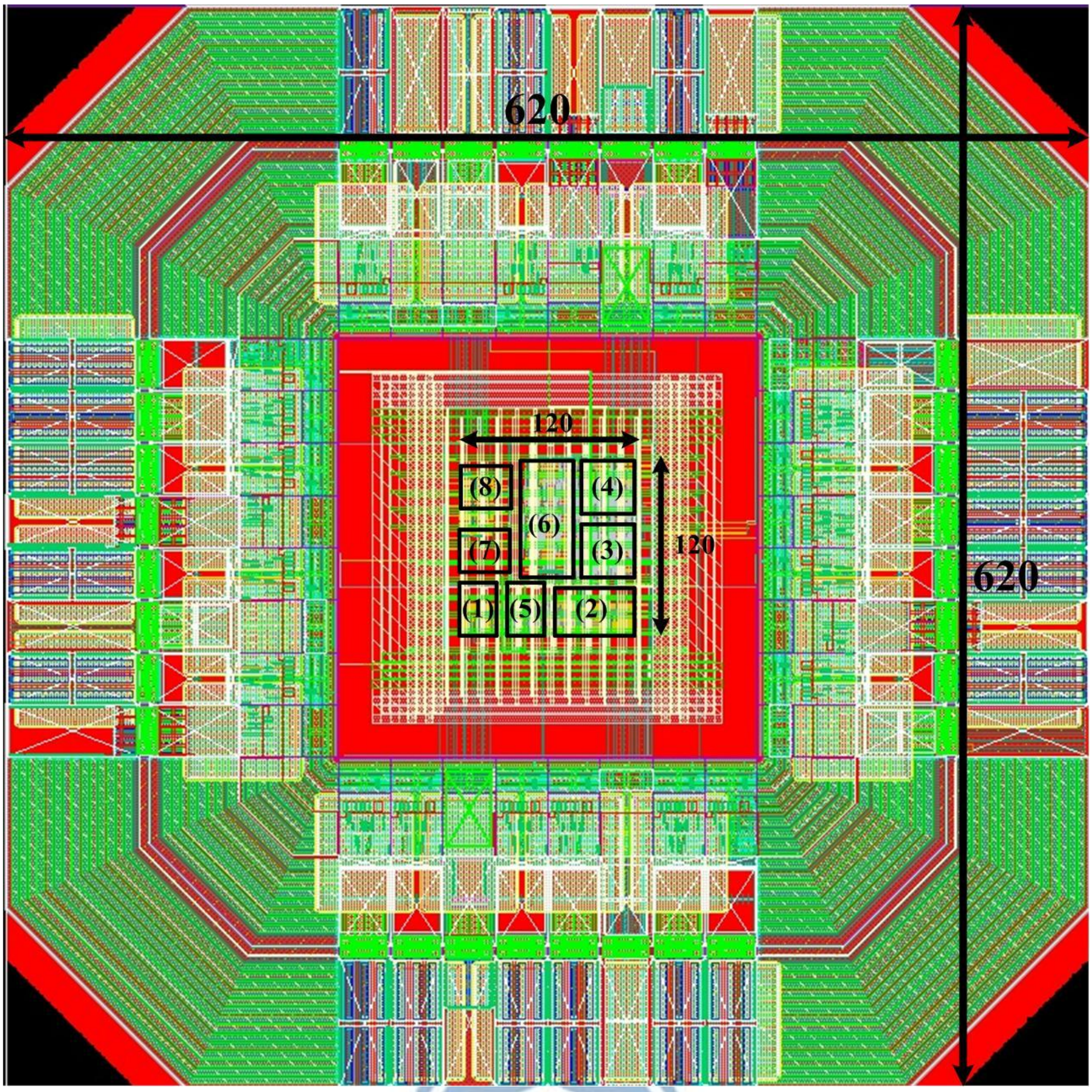
Chapter 4

Full Chip Experimental Results

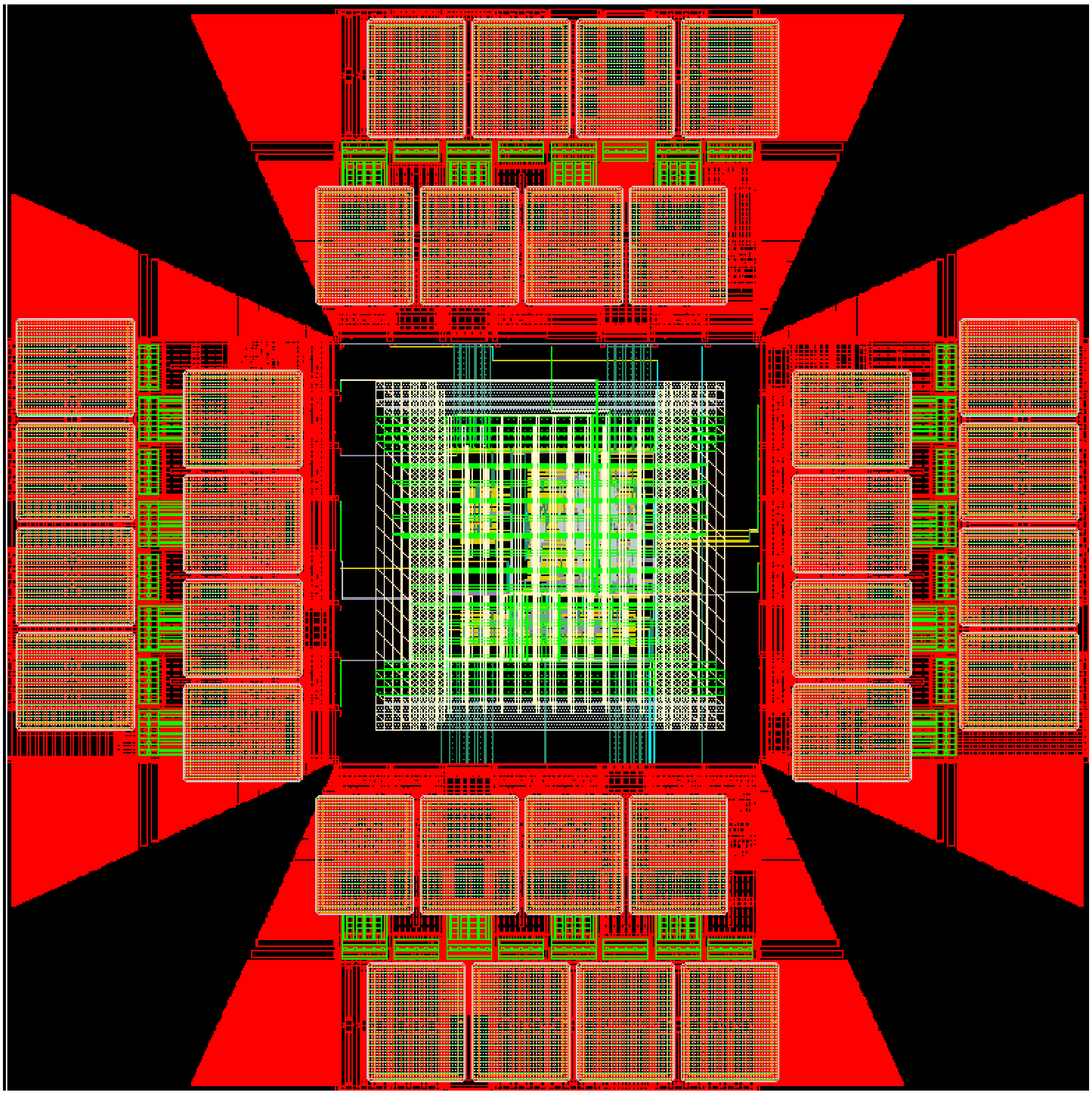
4.1 Specifications



(a)



(b)



(c)

Fig. 4.1: (a) Layout of the Test Chip without the Dummy cell. (b) Layout of the Test Chip with the Dummy cell. (c) Layout of the Test Chip with the Bonding Pad

Table 4.1: Block Module Name.

Block Number	Module Name
(1)	Phase and Frequency Detector (PFD)
(2)	ADPLL Controller

(3)	Frequency Finder
(4)	Cyclic Counter
(5)	Digital Loop Filter (DLF)
(6)	Digital Controlled Oscillator (DCO)
(7)	Output Divider
(8)	Frequency Divider

The proposed fast lock-in ADPLL is implemented in TSMC 40-nm UTM CMOS process. The complete chip layout is shown in Fig. 4.1(a). In addition, the chip layout with the dummy cells is shown in Fig. 4.1(b). Fig. 4.1(c) shows the chip layout with the bonding pads. The active area is 120 × 120 μm^2 and the chip area including I/O pads is 620 × 620 μm^2. In Table 4.1, the block module names are listed. The chip consists of the ADPLL and the test chip circuit. The gate count on the chip is about 6,965. The simulated power consumption is 0.511mW at the test mode = 1 and the test n = 5 with 0.9V supply voltage.

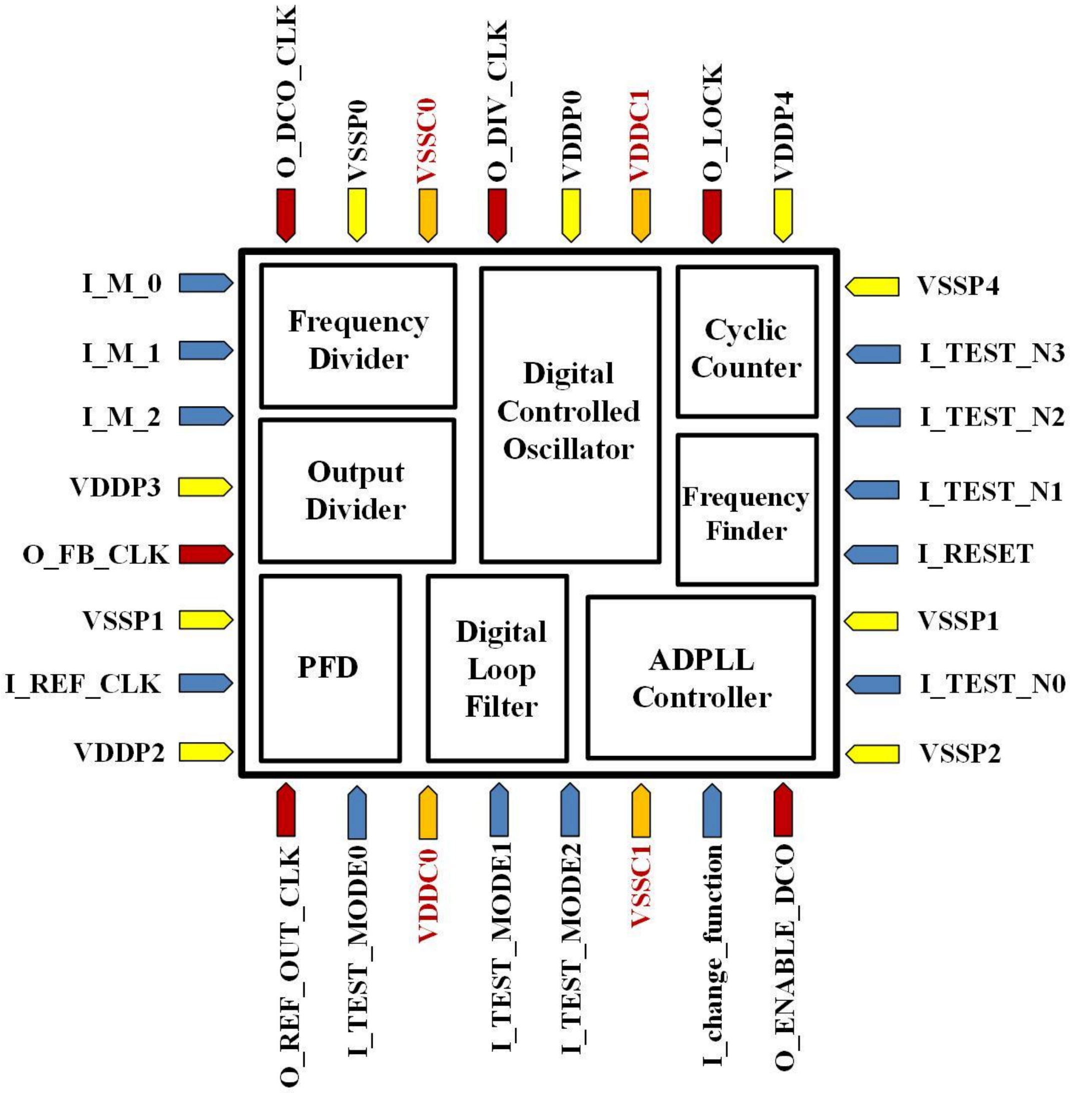


Fig. 4.2: Chip Floorplan and I/O Plan.

The chip I/O planning and the floorplanning of the proposed ADPLL are described in Fig. 4.2. The proposed ADPLL test chip is designed with 13 input pins, 6 output pins, and 13 power pins. The detail I/O pads information are shown in Table 4.2.

Table 4.2: I/O PAD Description.

Input	Bits	Function
REF_CLK	1	Reference input clock
RESET	1	Initial the chip at 0

I_change_function	1	**Value**	**Function**
		1’b0	Original Function
		1’b1	Improving Function
Test_N	4	Set the multiplication factor of ADPLL	
		Value	**Factor code**
		4’d0	2
		4’d1	4
		4’d2	8
		4’d3	16
		4’d4	32
		4’d5	64
		4’d6	128
		4’d7	256
		4’d8	7
		4’d9	13
		4’d10	19
		4’d11	23
		4’d12	47
		4’d13	73
		4’d14	131
		4’d15	257
M	2	Divide frequency of the output clock	
		Value	**Function**
		2’d0	Divide factor = 2 (test chip)
		2’d1	Divide factor = 4 (test chip)

		2’d2	Divide factor = 8 (test chip)	
		2’d3	Divide factor = 16 (test chip)	
Test_Mode	3	Set the test mode of the chip		
		Value	**Function**	
		3’d0	Block DCO_CLK	Fixed DCO at minimum frequency
		3’d1		Tracking frequency
		3’d2		Fixed DCO at first calculation code
		3’d3		Fixed DCO at maximum frequency
		3’d4	Block DIV_CLK	Fixed DCO at minimum frequency
		3’d5		Tracking frequency
		3’d6		Fixed DCO at first calculation code
		3’d7		Fixed DCO at maximum frequency

In Table 4.2, we design a 3-bit Test_Mode. Before measuring the chip performance, we require to figure out whether the chip is in FF corner, SS corner or TT corner. Therefore, we have four modes testing the minimum frequency and the maximum frequency of the chip. Besides, if two I/O pads output signals simultaneously, two signals may interfere with each other. Therefore, we block a clock signal during measuring the other clock signal.

4.2 Simulation Results

4.2.1 Test Mode Simulation

The post-layout simulation result of the proposed ADPLL in various test modes is shown in Fig. 4.3. The dco_clk or the div_clk are blocked in different test modes. In addition, the div_clk is operated at test modes 1 to 3; the dco_clk is operated at test modes 4 to 7. The detail functions of each mode are shown in table 4.2. Therefore, as we measure the clock signals, the clock signals may not be interfered by each other.

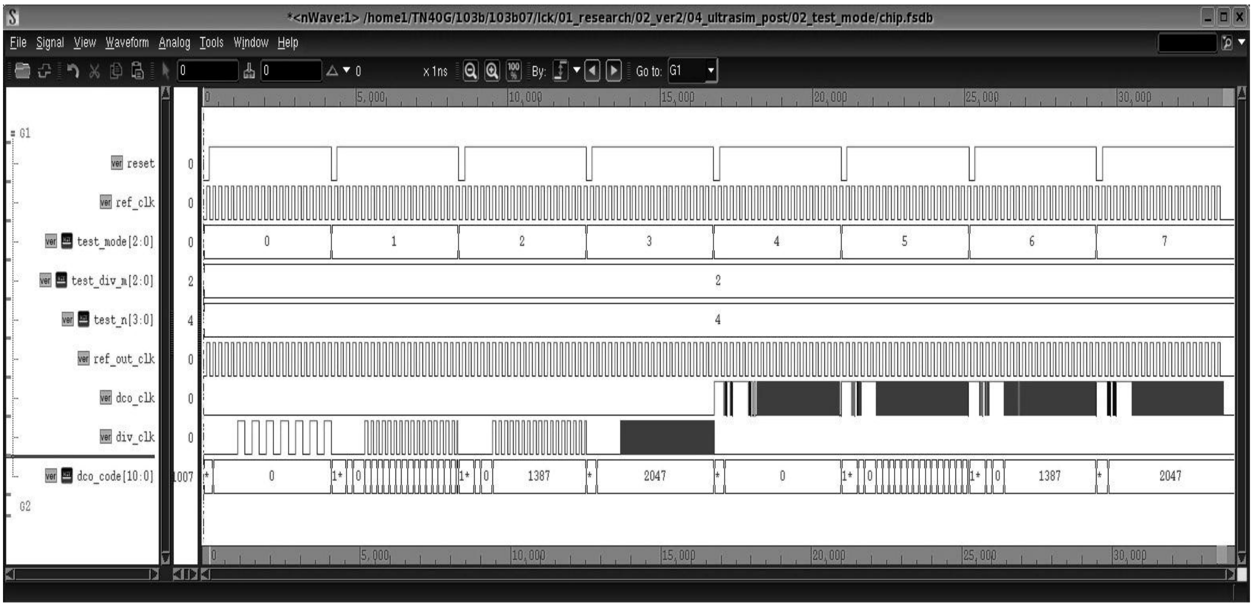
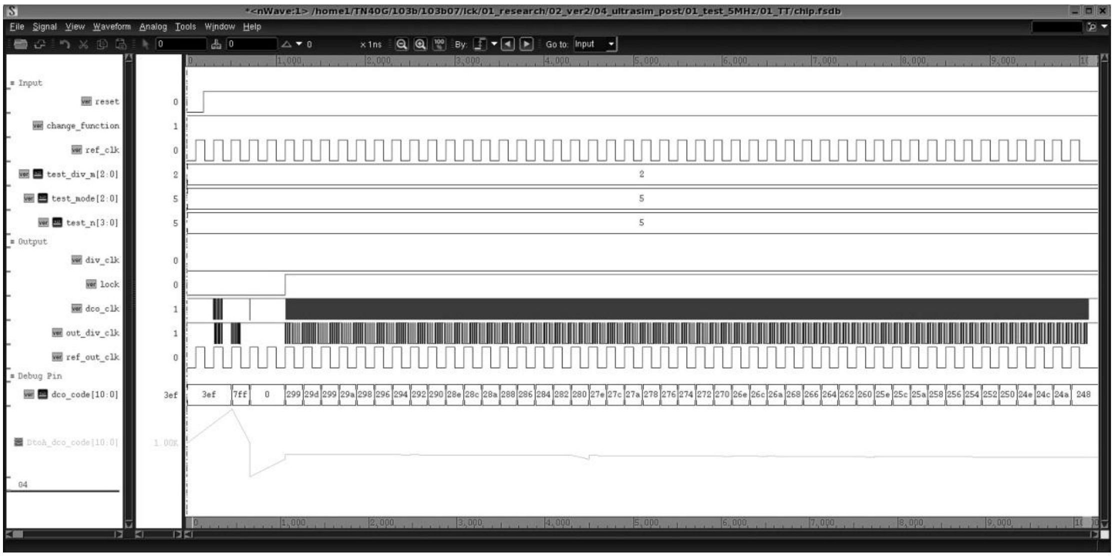
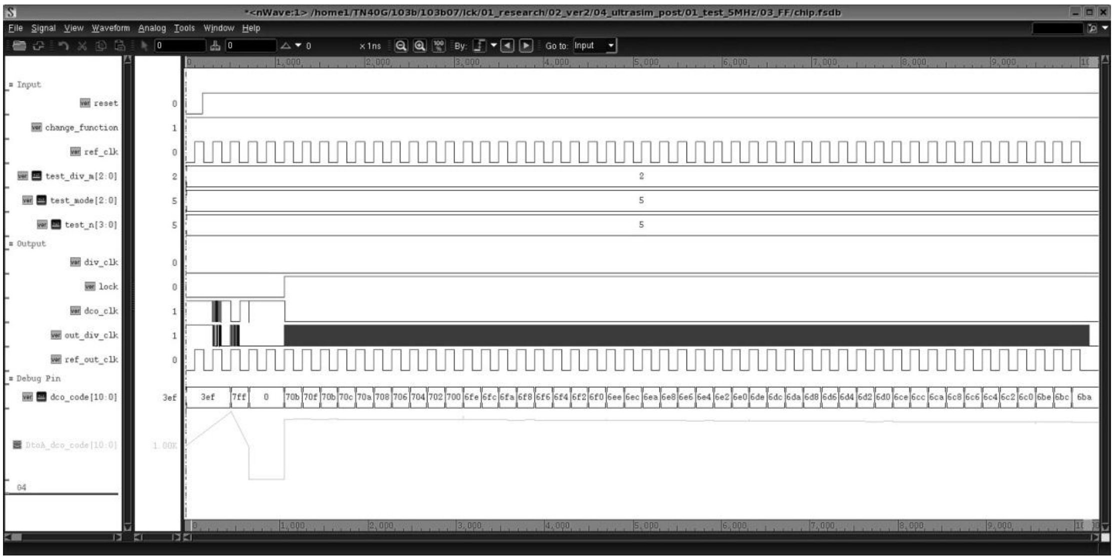


Fig. 4.3: System Simulation of the proposed ADPLL with various test modes.

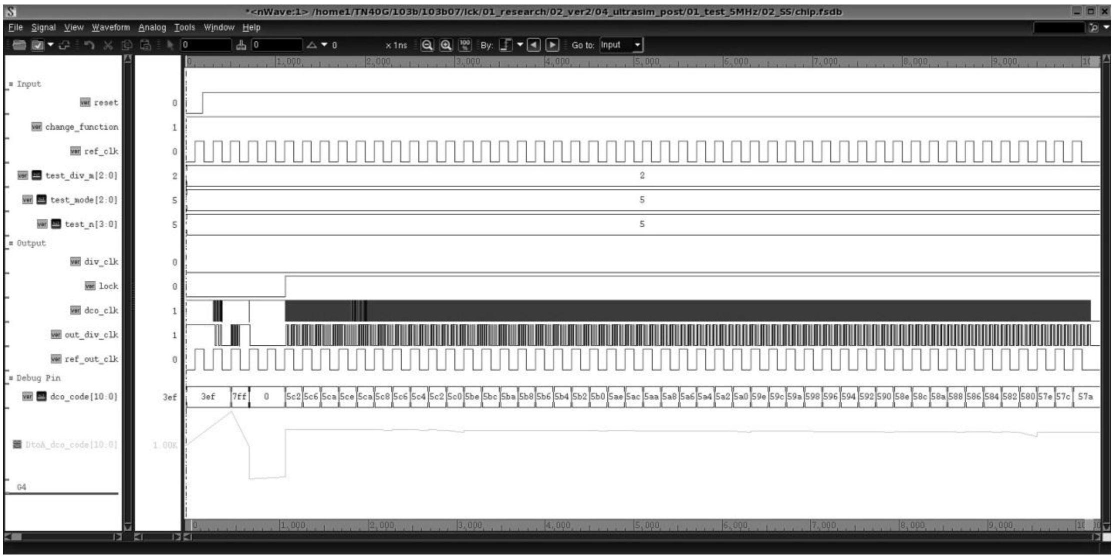
In Fig. 4.4(a), Fig. 4.4(b) and Fig. 4.4(c), we simulate the proposed ADPLL in TT corner, FF corner and SS corner, respectively. We set the test mode at 1 so that the dco_clk is disable output. The proposed fast lock-in ADPLL can lock correctly in different corner simulation.


*<nWave:1> /home1/TN40G/103b/103b07/ick/01_research/02_ver2/04_ultrasim_post/01_test_5MHz/01_TT/chip.fsdb
File Signal View Waveform Analog Tools Window Help
Go to: Input
Input
reset 0
change_function 1
ref_clk 0
test_div_m[2:0] 2
test_mode[2:0] 5
test_n[3:0] 5
Output
div_clk 0
lock 0
dco_clk 1
out_div_clk 1
ref_out_clk 0
Debug Pin
dco_code[10:0] 3ef
3ef 7ff 0 299 29d 299 29a 298 296 294 292 290 28e 28c 28a 288 286 284 282 280 27e 27c 27a 278 276 274 272 270 26e 26c 26a 268 266 264 262 260 25e 25c 25a 258 256 254 252 250 24e 24c 24a 248
Dtoa_dco_code[10:0] 1.00K
04
1,000 2,000 3,000 4,000 5,000 6,000 7,000 8,000 9,000

(a)


*<nWave:1> /home1/TN40G/103b/103b07/ick/01_research/02_ver2/04_ultrasim_post/01_test_5MHz/03_FF/chip.fsdb
File Signal View Waveform Analog Tools Window Help
Go to: Input
Input
reset 0
change_function 1
ref_clk 0
test_div_m[2:0] 2
test_mode[2:0] 5
test_n[3:0] 5
Output
div_clk 0
lock 0
dco_clk 1
out_div_clk 1
ref_out_clk 0
Debug Pin
dco_code[10:0] 3ef
3ef 7ff 0 70b 70f 70b 70c 70a 708 706 704 702 700 6fe 6fc 6fa 6f8 6f6 6f4 6f2 6f0 6ee 6ec 6ea 6e8 6e6 6e4 6e2 6e0 6de 6dc 6da 6d8 6d6 6d4 6d2 6d0 6ce 6cc 6ca 6c8 6c6 6c4 6c2 6c0 6be 6bc 6ba
Dtoa_dco_code[10:0] 1.00K
04
1,000 2,000 3,000 4,000 5,000 6,000 7,000 8,000 9,000

(b)


*<nWave:1> /home1/TN40G/103b/103b07/ick/01_research/02_ver2/04_ultrasim_post/01_test_5MHz/02_SS/chip.fsdb
File Signal View Waveform Analog Tools Window Help
Go to: Input
Input
reset 0
change_function 1
ref_clk 0
test_div_m[2:0] 2
test_mode[2:0] 5
test_n[3:0] 5
Output
div_clk 0
lock 0
dco_clk 1
out_div_clk 1
ref_out_clk 0
Debug Pin
dco_code[10:0] 3ef
3ef 7ff 0 5c2 5c6 5ca 5ce 5ca 5c8 5c6 5c4 5c2 5c0 5be 5bc 5ba 5b8 5b6 5b4 5b2 5b0 5ae 5ac 5aa 5a8 5a6 5a4 5a2 5a0 59e 59c 59a 598 596 594 592 590 58e 58c 58a 588 586 584 582 580 57e 57c 57a
Dtoa_dco_code[10:0] 1.00K
04
1,000 2,000 3,000 4,000 5,000 6,000 7,000 8,000 9,000

(c)

Fig. 4.4: System Simulation of the proposed ADPLL with PVT Variation.

(a) Post-Sim at 0.9V, Ref_clk = 5MHz, TT corner, Temperature 25℃

(b) Post-Sim at 0.99V, Ref_clk = 5MHz, FF corner, Temperature -40℃

(c) Post-Sim at 0.81V, Ref_clk = 5MHz, SS corner, Temperature 125℃

4.2.2 Chip Summary

Table 4.3: Chip Summary.

Process	40-nm CMOS
Core Area	0.0144 mm^2
Chip Area	0.3844 mm^2
Lock-in Time	4.5 cycles
Supply Voltage	0.9V
Time Resolution	3.43ps
Operating Range	129.8MHz~1.47GHz
Power Consumption	0.3749mW @ 129.8 MHz 0.981mW @ 1.47GHz

The chip summary is shown in Table 4.3. The core area and the chip area is 0.0144mm^2 and 0.3844mm^2, respectively. The operating frequency range of the proposed ADPLL is about 130MHz to 1.47GHz. In addition, the power consumption of the proposed ADPLL at 129.8MHz is 0.3749mW with 0.9V supply voltage. Besides, the DCO dominates the power consumption of the ADPLL. The power consumption of the DCO is 0.2674mW and 0.8918mW at 129.8MHz and 1.47GHz as shown in Table 4.4. In addition, the power consumption simulation results of the

ADPLL with PVT variations are shown in Table 4.5. The power consumption at 320MHz is 0.511mW, 1.1mW and 0.343mW in typical case, best case and worse case, respectively.

Table 4.4: The DCO Simulation with Different Frequencies.

Parameter	DCO @129.8MHz	DCO @ 1.47GHz
Power Consumption	0.2674mW	0.8918mW

Table 4.5: Simulation with PVT Variations.

Parameter	TT, 0.9V, 25℃	FF, 0.99V, -40℃	SS, 0.81V, 125℃
Power Consumption	0.5108mW @320MHz	0.9938mW @320MHz	0.3432mW @320MHz

4.2.3 Table of Comparison

Table 4.6: Performance Comparison Table with Prior Design.

Parameter	Proposed		[10]	[32]
Process	40-nm	90-nm	90-nm	90-nm
Core Area (μm×μm)	120×120	180×180	300×300	200×200
Category	ADPLL	ADPLL	ADPLL	ADPLL
Lock-in Time (cycles)	4.5	4.5	4	4
Supply Voltage (V)	0.9	1.0	1.0	1.0
Time Resolution	3.43ps	4.911ps	41ps@0.52V 6.6ps@1.0V	6.109ps
Input Frequency (MHz)	5	5	5	5~20
Output Frequency (MHz)	129.8~1465	93~1428	40~160@0.52V 160~460@1.0V	80~1606
Multiplication Factor	1~511	1~511	2~128	1~63
Power Consumption	0.3749mW @0.9V, 129 MHz 0.981mW @0.9V, 1.47GHz 0.031mW @0.5V, 80MHz	0.2759mW @ 93MHz 1.6278mW @ 1428MHz 0.9045mW @ 600MHz	37μW @0.52V, 120MHz 0.92mW @1.0V, 600MHz	0.3037mW @80MHz 1.73mW @1606MHz

Table 4.6 shows the comparison table with our prior researches. For fair comparison, we also implement the proposed ADPLL in TSMC 90-nm CMOS process. The proposed ADPLL has relatively fine DCO resolution and small chip area. Table 4.7 shows the comparison table with previous researches. Although [4] claims to lock-in in two clock cycles, it still has a large frequency error after two cycles. In addition, the frequency error comes from the quantization effects of the frequency counter.

Table 4.7: Performance Comparison Table.

Parameter	Proposed	[2] TCAS-II ’13	[26] ASSCC ’12	[1] JSSC ’15	[6] ICECS ’14	[5] ISCAS ’14	[4] TCAS-II ’10
Process	40-nm	28-nm	40-nm	40-nm LP	65-nm	90-nm	180-nm
Core Area (μm×μm)	120×120	52×45	350×140	316×316	360×270	323×200	250×300
Category	ADPLL	ADPLL	ADPLL	Digital LC PLL	ADPLL	ADPLL	ADPLL
Lock-in Time (cycles)	4.5	<50 (1μs)	42	40	45	<7	2
Supply Voltage (V)	0.9	1.0	0.5	0.9	1.0	1.2	1.8
Time Resolution	3.43ps	0.3~2.4ps	351ps	0.21fs	5fs	3.923ps	8.8ps @446MHz
Input Frequency (MHz)	5	5	1	390	15	3~10	0.22~8
Output Frequency (MHz)	129.8~1465	83~2000	10~100	21400~25100	1500~2800	460~6117	223~446
Multiplication Factor	1~511	50	10~100	16	100~187	50~2000	45~128
Frequency Error	0.43% @ Multiplication Factor=44	N/A	N/A	0.6% @ 25GHz	N/A	N/A	1% @ Multiplication Factor>=45
Power Consumption	0.3749mW @0.9V, 129 MHz 0.981mW @0.9V, 1.47GHz 31μW @0.5V, 80MHz	0.64mW @2GHz	4.7μW @10MHz 0.045mW @100MHz	64mW	8.85mW	9.237mW @6GHz	14.5mW @446MHz

Chapter 5

Conclusion and Future Works

5.1 Conclusion

In this thesis, an all-digital phase-locked loop (ADPLL) with an improving frequency estimation algorithm is presented. The frequency estimation algorithm requires the period ratio information that is calculated from the cyclic counter. In addition, the target DCO control code can be calculated by the frequency finder within 4.5 cycles.

Although the cyclic counter only can calculate the period ratio into integer numbers, we propose a calibration method to reduce quantization error. The calibration method can not only estimate the period ratio into fixed-point numbers, but also optimize the hardware utilization as compare to previous design with a cyclic TDC. Therefore, the proposed ADPLL can achieve fast lock-in time in 4.5 cycles.

Besides, the proposed interpolator-based fine-tuning architecture effectively solves the non-monotonic response problem of the DCO at 0.9V supply voltage. In addition, the interpolator-based fine-tuning architecture also improves the accuracy of the frequency estimation algorithm.

Therefore, in order to minimize the power consumption and reduce the chip area, the test chip is implemented in TSMC 40-nm CMOS process.

5.2 Future Works

In this thesis, there are some drawbacks in our ADPLL. The first issue is the DCO power consumption consumes most of the power consumption of the ADPLL. If we can reduce the power consumption of the DCO, we can minimize the ADPLL's total power consumption. Therefore, design a low-power DCO is required to further reduce the power consumption of the ADPLL.

There is another issue about frequency estimation algorithm. In section 1.2.4, we discuss the frequency estimation algorithm with and without the cyclic TDC. Although the cyclic TDC reduces the quantization error, the cyclic TDC cost large area. We assumed that the large area of the cyclic TDC is not concern. If we use the cyclic TDC and the proposed calibration method to calculate the period ratio, maybe we can further improve the accuracy of the initial target frequency.

Besides, there are some factors that may affect the estimation error in the frequency estimation algorithm such as jitter and duty cycle of the reference clock. If the reference clock duty cycle is not close to 50%, as we quantify the half of the reference clock, the quantization error may significantly increase. Therefore, if we quantify a clock period of the reference clock, the challenge of the duty cycle may be solved. However, the proposed frequency estimation algorithm should take at least 5 reference cycles to lock the target frequency.

Besides, as an ADPLL is operated at an extremely worse case, the reference clock jitter becomes large. In addition, the large jitter may significantly increase the estimation error. Therefore, after the frequency estimation algorithm, maybe we can increase the value of the step in binary search algorithm to reduce the lock-in time.

References

[1] Mohammad Hekmat, Farshid Aryanfar, Jason Wei, Vijay Gadde, and Reza Navid, “A 25GHz fast-lock digital LC PLL with multiphase output using a magnetically-coupled loop of oscillators,” *IEEE Journal of Solid-State Circuits*, vol. 50, no. 2, pp.490-502, Feb. 2015.

[2] Sebastian Höppner, Stefan Haenzsche, Georg Ellguth, Dennis Walter, Holger Eisenreich, and René Schüffny, “A fast-locking ADPLL with instantaneous restart capability in 28-nm CMOS technology,” *IEEE Transactions on Circuits and System II: Express Briefs*, vol. 59, no. 10, pp. 673-677, Oct. 2013.

[3] Yun-Chen Chuang, Sung-Lin Tsai, Cheng-En Liu, and Tsung-Hsien Lin, “An all-digital phase-locked loop with dynamic phase control for fast locking, ” *in Digest of Technical Papers, IEEE Solid-State Circuits Conference (A-SSCC)*, pp. 297-300, Nov. 2012.

[4] Chia-Tsun Wu, Wen-Chung Shen, Wei Wang, and An-Yeu Wu, “A two-cycle lock-in time ADPLL design based on a frequency estimation algorithm,” *IEEE Transactions on Circuits and System II: Express Briefs*, vol. 57, no. 6, pp.430 - 434, Jun. 2010.

[5] Yi-Wei Chen and Hao-Chiao Hong, “A fast-locking all-digital phase locked loop in 90nm CMOS for gigascale systems,” *in Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1134-1137, Jun. 2014.

[6] Keisuke Okuno, Kana Masaki, Shintaro Izumi, Toshihiro Konishi, Hiroshi Kawaguchi and Masahiko Yoshimoto, “A 2.23ps RMS jitter 3us fast settling ADPLL using temperature compensation PLL controller,” *in Proceedings of*

IEEE International Conference on Electronic, Circuits and System (ICECS), pp. 68-71, Dec. 2014.

[7] Robert Bogdan Staszewski and Poras T. Balsara, "All-digital PLL with ultra fast settling," *IEEE Transactions on Circuits and System II: Express Briefs*, vol. 54, no. 2, pp.181-185, Feb. 2007.

[8] Chao-Ching Huang and Shen-Iuan Liu, "A 40-GHz fast-locked all-digital phase-locked loop using a modified bang-bang algorithm," *IEEE Transactions on Circuits and System II: Express Briefs*, vol. 58, no. 6, pp.321-325, Jun. 2011.

[9] Keisuke Okuno, Shintaro Izumi, Toshihiro Konishi, Song Dae-Woo, Masahiko Yoshimoto, and Hiroshi Kawaguchi, "Temperature compensation using least mean squares for fast settling all-digital phase-locked loop," *in Proceedings of IEEE International New Circuits and Systems Conference (NEWCAS)*, Jun. 2013.

[10] Chin-Che Chung, Wei-Siang Su, and Chi-Kuang Lo, "A 0.52V/1.0V fast lock-in ADPLL for supporting dynamic voltage and frequency scaling," *in press, IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Feb. 2015.

[11] Chin-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.

[12] Dou Sheng, Chin-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)* , pp.123-127, Jul. 2012.

[13] G. Yu, Y. Wang, H. Yang, and H. Wang, "Fast-locking all-digital phase-locked loop with digitally controlled oscillator tuning word estimating and presetting," *IET Circuits, Devices & Systems*, vol. 4, no. 3, pp. 207-217, May. 2010.

[14] Ching-Che Chung, Duo Sheng, Chia-Lin Chang, Wei-Da Ho, Yang-Di Lin, and Fang-Nien Lu, “An all-digital large-N audio frequency synthesizer for HDMI applications,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, no. 7, pp. 424-428, Jul. 2012.

[15] Jim Dunning, Gerald Garcia, Jim Lundberg, and Ed Nuckolls, “An all-digital phase-locked loop with 50-cycle lock time suitable for high-performance microprocessors,” *IEEE Journal of Solid-State Circuits*, vol. 30, no. 4, pp. 412-422, Apr. 1995.

[16] 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.

[17] 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.

[18] Duo Sheng, Ching-Che Chung, and Chen-Yi Lee, “A fast-lock-in ADPLL with high-resolution and low-power DCO for SoC applications,” *in Proceedings of IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 105-108, Dec. 2006.

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

[20] Ching-Che Chung, Chiun-Yao Ko, and Sung-En Shen, “Built-in self-calibration circuit for monotonic digitally controlled oscillator design in 65-nm CMOS technology,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 58, no. 3, pp. 149-153, Mar. 2011.

[21] Pei-Ying Chao, Chao-Wen Tzeng, Shi-Yu Huang, Chia-Chieh Weng, and Shan-Chien Fang, “Process-resilient low-jitter all-digital PLL via smooth code-jumping,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 12, pp. 2240-2249, Dec. 2013.

[22] Davide De Caro, “Glitch-free NAND-based digitally controlled delay-lines,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 1, pp. 55-66, Jan. 2013.

[23] 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.

[24] Seong-Young Seo, Jung-Hoon Chun, Young-Hyun Jun, Seok Kim, and, Kee-Won Kwon, “A digitally controlled oscillator with wide frequency range and low supply sensitivity,” *IEEE Transactions on Circuits and System II: Express Briefs*, vol. 58, no. 10, pp. 632-636, Oct. 2011.

[25] Kuo-Hsing Cheng, Jen-Chieh Liu, and Hong-Yi Huang, “A 0.6-V 800-MHz all digital phase-locked loop with a digital supply regulator,” *IEEE Transactions on Circuits and System II: Express Briefs*, vol. 59, no. 12, pp. 888-892, Dec. 2012.

[26] Yasuyuki Hiraku, Isamu Hayashi, Hayun Chung, Tadahiro Kuroda, and Hiroki Ishikuro, “A 0.5V 10MHz-to -100MHz 0.47uW/MHz power scalable AD-PLL in 40nm CMOS,” *in Proceedings of IEEE Asian Solid-State Circuits*, pp. 33-36, Nov. 2012.

[27] Yingchieh Ho, Yu-Sheng Yang, ChiaChi Chang, and Chauchin Su, “A near-threshold 480MHz 78µW all-digital PLL with a bootstrapped DCO, ” *IEEE Journal of Solid-State Circuits*, vol. 48, no. 11, pp. 2805-2814, Nov. 2013.

[28] Minyoung Song, Inhwa Jung, Sudhakar Pamarti, and Chulwoo Kim, “A 2.4

GHz 0.1-Fref-Bandwidth All-Digital Phase-Locked Loop with Delay-Cell-Less TDC," *IEEE Transactions on Circuits and System-I: Regular Papers*, vol. 60, no. 12, pp. 3145-3151, Dec. 2013.

[29] Ching-Che Chung, Duo Sheng, and Wei-Da Ho, "A low-cost low-power all-digital spread-spectrum clock generator," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 5, pp. 983-987, May 2015.

[30] Song-Yu Yang, Wei-Zen Chen, and Tai-You Lu, "A 7.1 mW, 10 GHz all digital frequency synthesizer with dynamically reconfigured digital loop filter in 90 nm CMOS technology," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 3, pp. 578-586, Mar. 2010.

[31] Hua Geng, Jianbo Sun, Shuai Xiao, and Geng Yang, "Modeling and implementation of an all digital phase-locked-loop for grid-voltage phase detection," *IEEE Transactions on Industrial Information*, vol. 9, no. 2, pp. 772-780, May 2013.

[32] Ching-Che Chung, Duo Sheng, and Chen-Han Chen, "An all-digital phase-locked loop compiler with liberty timing files," *in Proceedings of International Symposium on VLSI Design, Automation, and Test (VLSI-DAT)*, Apr. 2014.

[33] Mohammad hassan shabaany and Mohsen saneei, "A 0.7-to-1.1GHz all-digital phase-locked loop with a new phase frequency detector and controlled oscillator with body-biasing," *in Proceedings of Computer Architecture and Digital System (CADS)*, pp. 54-59, Apr. 2014.

[34] Liaangge Xu, Kari Stadius, and Jussi Ryynanen, "An all-digital PLL frequency synthesizer with an improved phase digitization approach and an optimized frequency calibration technique," *IEEE Transactions on Circuits and System-I: Regular Papers*, vol. 59, no. 11, pp. 2481-2494, Dec. 2012.

[35] Ahmed Musa, Wei Deng, Teerachot Siriburanon, Masaya Miyahara, Kenichi Okada, and Akira Matsuzawa, “A compact, low-power and low-jitter dual-loop injection locked PLL using all-digital PVT calibration,” *IEEE Journal of Solid-State Circuits*, vol. 49, no. 1, pp. 50-60, Jan. 2014.

[36] Pyoungwon Park, Jaejin Park, Hojin Park, and SeongHwan Cho, “An all-digital clock generator using a fractionally injection-locked oscillator in 65nm CMOS,” *in Digest of Technical Papers, IEEE Solid-State Circuits Conference (ISSCC),* pp. 336-337, Feb. 2012.

[37] Wei Deng, Dongsheng Yang, Tomohiro Ueno, Teerachotchot Siriburanon, Satoshi Kondo, Kenichi Okada, and Akira Matsuzawa, “A fully synthesizable all-digital PLL with interpolative phase coupled oscillator, current-output DAC, and fine-resolution digital varactor using gated edge injection technique,” *IEEE Journal of Solid-State Circuits*, vol. 50, no. 1, pp. 68-80, Jan. 2015.

[38] I-Ting Lee, Kai-Hui Zeng, and Shen-Iuan Liu, “A 4.8-GHz dividerless aubharmonically injection-locked all-digital PLL with a FOM of – 252.5 dB,” *IEEE Transactions on Circuits and System II: Express Briefs*, vol. 60, no. 9, pp. 547-551, Sep. 2013.

[39] Joonsung Bae, Kiseok Song, Hyungwoo Lee, Hyunwoo Cho, and Hoi-Jun Yoo, “A low-energy crystal-less double-FSK sensor node transceiver for wireless body-area network,” *IEEE Journal of Solid-State Circuits*, vol. 47, no. 11, pp. 2678-2692, Nov. 2012.

[40] M. Souri, M. B. Ghaznavi-Ghoushchi, and A. Abadian, “A low-power dual band all digital PLL with precision dual mode DCO and digital linearization control circuits,” *in Proceedings of 22nd Iranian Conference on Electrical Engineering (ICEE)*, May. 2014, pp. 284-289.

[41] Feng-Wei Kuo, Ron Chen, Kyle Yen, Hsien-Yuan Liao, Chewn-Pu Jou,

Fu-Lung Hsueh, Masoud Babaie, and Robert Bogdan Staszewski, “A 12mW all-digital PLL based on Class-F DCO for 4G Phones in 28 nm CMOS,” *in Digest of Technical Papers, Symposium on VLSI Circuits,* pp. 1-2, Jun 2014.

[42] Tsung-Hsien Tsai, Min-Shueh Yuan, Chih-Hsien Chang, Chia-Chun Liao, Chao-Chieh Li, and Robert Bogdan Staszewski, “A 1.22ps integrated-jitter 0.25-to-4GHz fractional-N ADPLL in 16nm FinFET CMOS,” *in Digest of Technical Papers, IEEE Solid-State Circuits Conference (ISSCC),* pp. 22-26, Feb. 2015.

[43] Chen-Yi Lee and Ching-Che Chung, “Digital Loop Filter for All-Digital Phase-Locked Loop Design,” US patent 7,696,832 B1, Apr.13, 2010.