International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 4 (2015) pp. 10995-11011

http://www.ripublication.com

An Efficient All Digital Phase Locked Loop Using Programmable Delay Line Circuit

[1]N Nagadevi
Department of ECE, Sathyabama University, Chennai-119, Tamilnadu, India.
E-mail: nagadevisaladi@gmail.com
[2]Dr G SUNDARI
Professor, Department of ECE, Sathyabama University, Chennai-119, Tamilnadu, India, E-mail: sundarig2014@gmail.com

Abstract

An efficient All Digital Phase Locked Loop (ADPLL) is presented in this paper. The ADPLL is designed using standard cells and described by Hardware Description Language (HDL). The ADPLL implemented in a 90 nm CMOS process can operate from 10 to 200MHz.The key issues of ADPLL are jitter and noise. The previously proposed ADPLL uses NAND based DCDL which gives glitches at the output, increasing in jitter. And the existing ADPLL structure uses SAR delay search which consumes more power. Hence in order to reduce power consumption and jitter a new ADPLL structure is proposed. Comparison of ADPLL using NAND based DCDL and ADPLL MUX based DCDL is done.

Keywords: Phase locked loop (PLL), All digital phase locked loop **(**ADPPLL), Bang-Bang phase and frequency Detector (BBPFD), Digitally controlled Oscillator (DCO), Digitally Controlled Delay Line (DCDL),MUX based DCDL.

Introduction

PHASE-LOCKED loops are widely used for many applications, such as clock and data recovery (CDR) circuits, frequency synthesizers, telecommunications and on-chip clock generators. PLLs have become indispensable modules in system-on-a-chip (SoC) designs. In clock and data recovery applications such as hard-disk drivers and digital videodisks, the channel characteristics vary over time, the PLL needs to respond accordingly to unpredictable phase fluctuations, instantaneous frequency shifts, and time-varying jitter [2]. But classical PLL implementations rely on analog components which are not suited for usage in such variable environments.

All-Digital phase locked loops have been proposed to overcome those disadvantages of analog Phase Locked Loops. Compared With analog PLLs, the all-digital phase locked loops (ADPLL) has advantages such as robustness, easy-to-process migration, and without a passive loop filter. By using an ADPLL the turnaround time for system integration will also be reduced. One of the key issues that slow down ADPLL locking is that even though the ADPLL already outputs a correct frequency, if its phase error is large, it still needs to adjust this frequency and the process results in increasing jitter. To resolve this issue a digital-controlled delay line (DCDL) is used, which helps to insert optimum delay between the input and output to minimize the phase difference. In previously proposed ADPLL NAND based DCDLs are used which are presenting glitches at the output. This performance leads to more jitter. A glitch free NAND based DCDL and A MUX based DCDL is proposed in this paper in order to reduce the jitter and power consumption in the ADPLL circuit.

Different DCDL topologies have been designed. The classical approach to design a DCDL is using a delay-cells chain and a mux to select the desired cell output. In these chain-based DCDLs, the delay increases with the increase of the number of cells and this result in a tradeoff between the delay range and minimum delay of the DCDL. The large of chain-based DCDLs can be reduced by using a tree-based multiplexer topology. This however results in an irregular structure which complicates layout design and also increases the nonlinearity of the DCDL. Then DCDLs are designed by using three state inverters, which also increasing the delay with increase in delay cells. In order to maintain minimum delay time and resolution DCDLs are designed using NAND gates as delay cells, they provided two times of NAND gate delay as the minimum delay which is satisfactory. But the above topologies of DCDL are unable to remove glitches. In order to remove glitches modified glitch free NAND based DCDLs are designed but results in more power consumption and area. To overcome the above issues, a MUX based DCDL is proposed in this paper.

Existing Adpll Sysytem

The architecture of the existing ADPLL is illustrated in **Error! Reference source not found.** 1, which consists of a bang-bang phase detector (BBPHD), a digital loop filter, a divider, a five-stage digitally controlled oscillator (DCO), a SAR delay-search block that determines the amount of delay needed for phase alignment, and a SAR frequency-search block that determines the initial control code of the Digitally Controlled Oscillator.

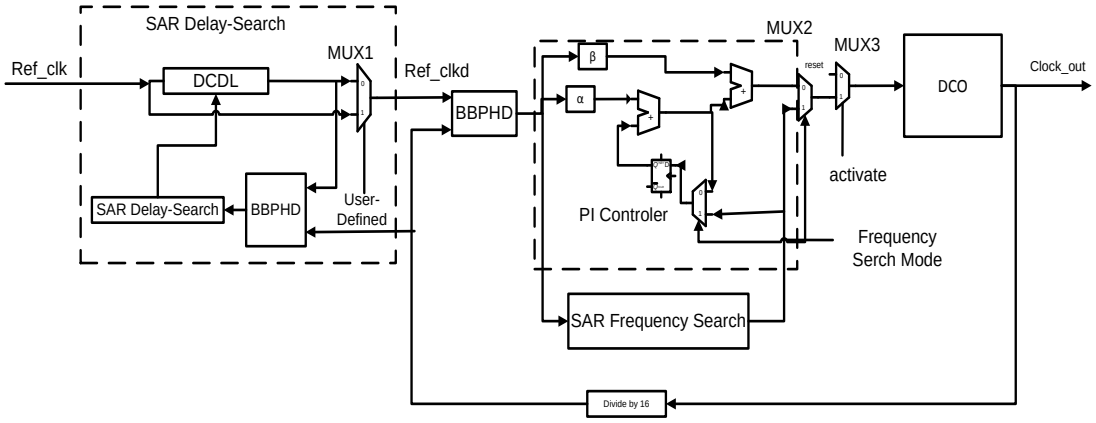

Figure 1: Block diagram of existing ADPLL

A.SAR based Frequency search:

The frequency-search block, firstly, initializes the most significant bit (MSB) of the DCO binary code to 1 and keeps the remaining bits to be 0. When the clock is high, the DCO, consisting of tri-state buffers, is disabled with all buffers, while the divider is also reset to output *Vdd*. At the falling edge of the reference clock, the divider is activated and it starts falling from *Vdd*. MUX2 then passes the DCO thermal code, which also comes from the frequency-search block, and enables the DCO. After that, with divide-by-16 circuitry, the divider output signal will start rising after 8 cycles of DCO output. The BBPHD, by identifying whether divider clock signal rises earlier than the reference, decides whether the MSB should be 1 or 0. When the reference clock is high again, the divider and DCO are reset again; and the frequency-search block sets the MSB based on this BBPHD decision. Starting the next round, the frequency-search block sets the next MSB [MSB-1] to 1 and then goes through the above process to check whether this MSB-1 should be 1 or 0.

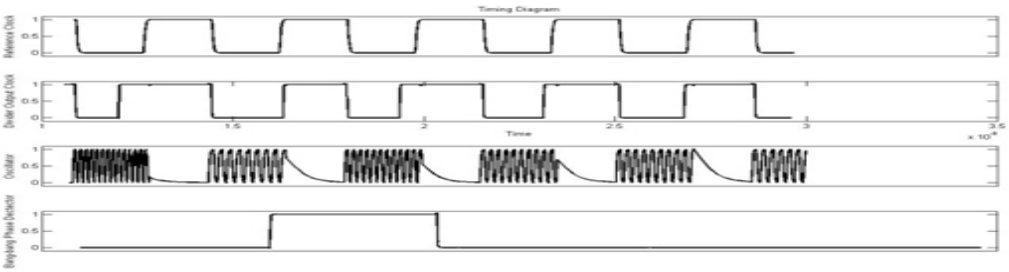

Figure 2: Timing diagram of SAR-based frequency-search DCO is activated when reference clock is low; BBPHD detects the rising edge of divider and reference clock

B. Digitally Controlled Oscillator design:

A five-stage tri-state buffer based ring oscillator is implemented as shown in Figure 3, which consists of 960 tri-state buffers controlled by thermo code. In order to reduce

the complexity of binary-to-thermo decoder, 15 tri-state buffers are grouped into a row controlled by a 64-bit thermo code converted from a 4-bit binary code. And then within the row group, the row decoder controls whether this group should be all on or off or should be partially on based on a 16 bits thermo code. When the ring oscillator is off, all the internal nodes are not driven and may not be able to stay at either *Vdd* or *gnd* due to leakage, which will reduce the DCO start-up speed. Therefore, five extra tri-state buffers are added to drive each node to either 1 or 0 during the reset. The top 4 rows of this DCO are always on after frequency-search, which sets the minimum frequency and worst case resolution. Since the maximum frequency is only limited by the delay of tri-state buffer cells, and the minimum frequency is limited by the always-on buffers in the top row driving all the buffers, this DCO achieves a wide range of frequency (0.42-12GHz).

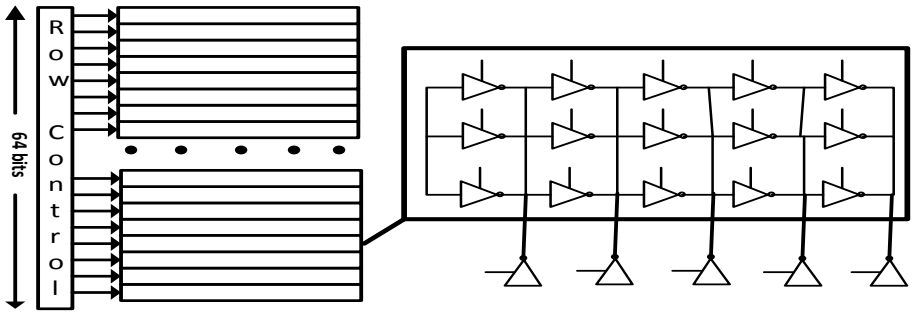

Figure 3: Five state DCO using tri-state buffers

C. Loop Filter:
After the SAR frequency-Search, both frequency and phase errors are small, which allows this ADPLL to use a much smaller proportional path gain and integral path gain to reduce the jitter due to quantization noise. However, the resolution of DCO sets a limit on this quantization noise. In order to achieve stability, the ratio of proportional path gain (β) to integral path gain (α) needs to be greater than a certain number. If α is set to DCO resolution, then β needs to be several times larger, which results in huge quantization noise. If the DCO resolution is kept small, a large number of tri-state buffers need to be on, resulting in huge power consumption. In order to use close to minimum resolution, in this ADPLL, integral path code increment or decrement by 1 when it is a multiple of 4, which creates an equivalent integral path gain of $\alpha=¼$. The proportional path gain is chosen to be $\beta=2$. As a result, the largest step is when the integral path increment by 1 and the proportional path gives a 2, which is $3K_{DCO}$, whereas K_{DCO} is the resolution of DCO.

D. SAR based delay search:
In the implementation, during frequency-search mode, the BBPHD is actually comparing a delayed version of the reference clock, REF_CLKD in **Error! Reference source not found.**4, instead of the reference clock itself, REF_CLK in **Error! Reference source not found.**. Since such delay is sensitive to process and temperature variations,

a digitally controlled delay line (DCDL) is employed to adjust the amount of delay needed to align the falling edge of the reference clock with the divider output clock.

As shown in **Error! Reference source not found.**, this DCDL is implemented with NAND gates, consisting of four different delay paths controlled by SAR-based delay search. Similar to the SAR algorithm used in frequency-search, a conventional SAR delay-search block is implemented here. When the reference clock is high, it sets the MSB to be 1 and resets the divider and oscillator. When the reference clock is low, another detector, BBPHD2, checks the falling edge of the delayed reference clock and the divider clock to determine the MSB. Then it continues to check the remaining bits with the same algorithm and begins the SAR-based frequency-search later when it is done. This extra delay line does generate additional phase noise to the clock, which can somehow be filtered out through a low-pass ADPLL system.

In the existing architecture shown in Figure 1, MUX1 is used to select whether the input clock should be the original reference clock or its delayed version after both delay-search and frequency-search. If the ADPLL system bandwidth is large and CPU clock jitter requirement is strict, then the original reference clock should be selected as the input clock. In this situation, the ADPLL still needs several cycles to compensate the phase error due to divider and oscillator delay before it resolves the remaining frequency error. On the other hand, if the ADPLL system bandwidth is small and CPU demands fast locking property of ADPLL during frequency scaling to improve its power performance, then the delayed version of the reference clock should be selected because in this case, once the frequency search is done, there is almost negligible phase error between the divider and the reference clock, and the ADPLL can adjust the remained frequency error right away.

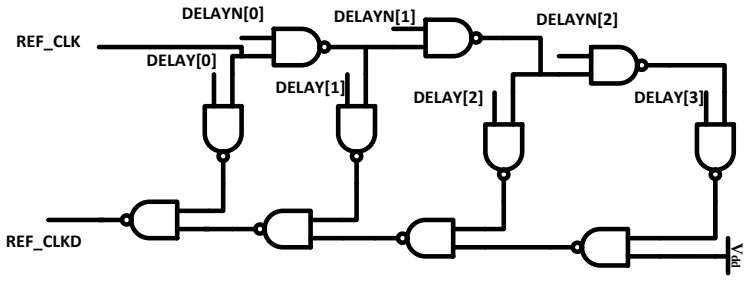

Figure 4: NAND based DCDL

Proposed Adpll System

The proposed ADPLL is shown in figure 5 which consists of a digital PFD, a digital loop filter, DCO, a frequency divider, a DCDL, and a digital controller. The SAR delay search is time taking process and the registers consume more power and area, two new programmable delay line circuits are proposed with a digital controller to control the delay. By using the proposed DCDL the glitches are avoided at the output

and hence the jitter is reduced. And the PI controller is replaced with a digital loop filter to improve the efficiency.

The operation of the proposed ADPLL is, at first the Digital controller makes the decision of either increasing or decreasing in delay by comparing the delayed reference clock and the feedback divided clock. Then the DCDL delays the reference clock according to delay control code generated by digital controller and the PFD compares the delayed reference clock signal with a feedback clock signal from a frequency divider on either positive or negative edges. Decrement signals will be generated if the edges of lead the edges of delayed reference signal. On the other hand, increment signal will be generated if the edges of feedback signal fall behind those of delayed reference signal. And then the digital loop filter generates a control signal to the DCO, based on the input from the PFD. According to the control signal, a feedback signal is generated from the DCO and passes through the frequency divider which reduces the frequency by N times. The frequency-divided feedback signal will then is compared With delayed reference signal as a routine, to continuously modify the voltage level of the control signal from the DCO until the frequency and phase difference between and frequency divided signal is minimized. When the locking is effective, either the decrement or the increment signals from PFD shall be Zero.

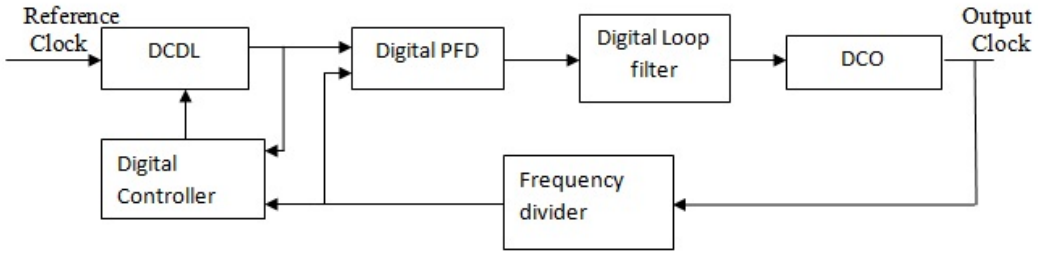

Figure 5: Proposed ADPLL structure

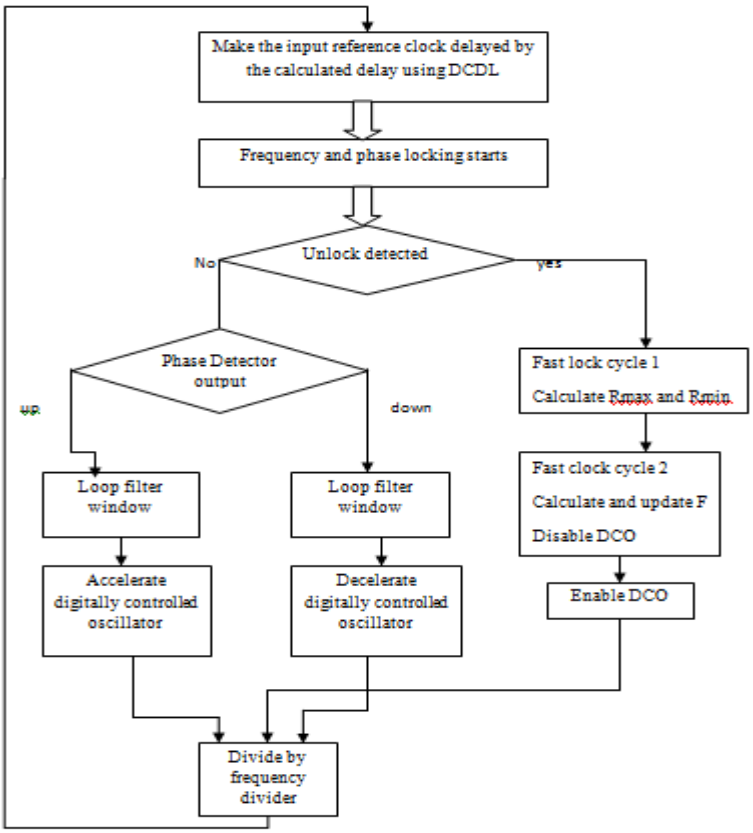

Figure 6: Flow chart for proposed ADPLL structure

A. Proposed Dcdls For Adpll

1. Glitch free NAND based DCDL:

The structure of proposed DCDL is shown in Fig. 7. In this figure "A" denotes the fast input of each NAND gate. Gates marked with "D", represents dummy cells added for load balancing. Two sets of control-bits S_i and T_i, control the DCDL. The S_i bits encode the control-code by using a thermometric code: $S_i = 0$ for $i < c$ and $S_i = 1$ for $i \geq c$. The bits encode again by using a one-cold code: $T_{c+1} = 0$, $T_i = 1$ for $i \neq c+1$. According to the chosen control-bits encoding, each delay-element (DE) can be in one of three possible states.

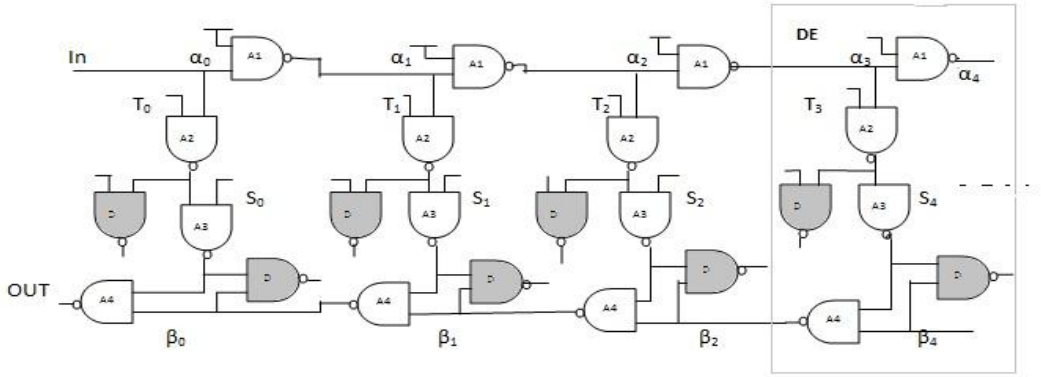

Figure 7: Glitch free NAND based DCDL (inverting topology)

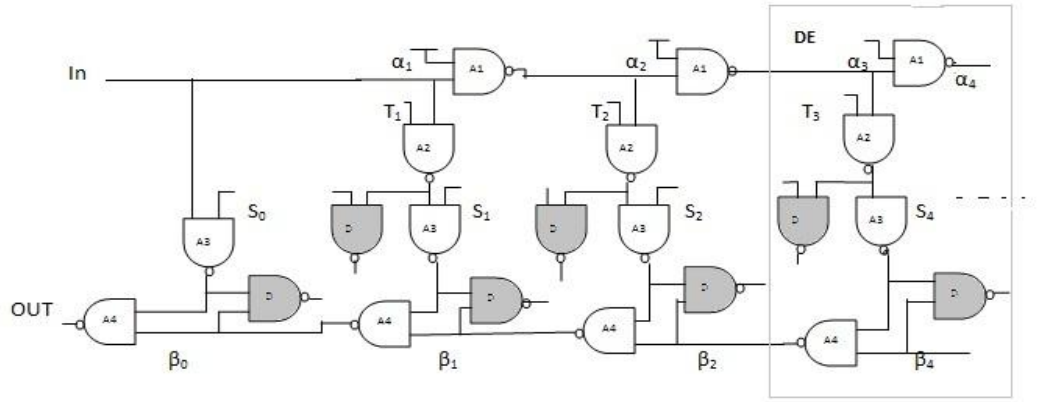

Figure 8: Glitch free NAND based DCDL (non-inverting topology)

Table 1: Logic-States of each DE in glitch free NAND based DCDL

S_i	T_i	DE State
0	**1**	**Pass**
1	**1**	**Turn**
1	**0**	**Post-turn**

The DEs with i < c are in pass-state. In this state the NAND "3" output is equal to 1 and the NAND "4" allows the signal propagation in the lower NAND gates chain. The DE with i=c is in turn-state. In this state the upper input of the DE is passed to the output of NAND "3". The next DE is in post-turn-state. In this state DE the output of the NAND "4" is stuck-at 1, by allowing the propagation, in the previous DE (which is in turn-state), of the output of NAND "3" through NAND "4". All remaining DEs

(for i > c+1) are again in turn-state. The three possible DE states of proposed DCDL and the corresponding and values are summarized in Table I.

In the proposed DCDL the state of all α_i and β_i and signals depends on the input with the only exception of β_c, which is stuck-at 1. The glitch-free switching property of the proposed DCDL is conceptually simple to demonstrate. Let us assume a switching of the delay control-code from c=k to c=h. In the initial state of the line, $\alpha_{2i} = \beta_{2i}$ = In and $\alpha_{2i+1} = \beta_{2i+1}$ = inverted In, with the exception of β_k, which is stuck-at 1. Let us suppose to first switch the k+1th DE from the post-turn-state to the turn-state. By looking to Fig. 6 it can be observed that, in these conditions, switches from 1 to α_k. The signal β_k is the input of the NAND"4" gate of kth DE. The switching of β_k is glitch-free since the other input of this gate is stuck-at α_k, therefore the NAND "4" output remains equal to α_k. After the k+1th DE switching, all cells are either in pass-state or in turn-state. In these conditions it is possible to freely change the state of DEs from pass-state to turn-state, since this change does not affect the logic state of signals α_i and β_i. After this phase the h+1th DE can be switched from turn-state to post-turn-state. This switching is again glitch free, since only β_h signal switches from α_h to 1. This procedure has the drawback to require a three-step switching of the DCDL.

The circuit of Fig. 7 is an inverting DCDL. In this circuit it is interesting to observe that the first DE is never in post-turn state, therefore T_0 is always 1 (see Table I). This observation allows constructing a non-inverting DCDL by modifying only the first DE, as shown in Fig. 8. In this circuit the NAND gates "1" and "2" of the first DE have been deleted, together with signal. The signal α_i of the second DE is now equal to *In*, therefore the whole behavior of the DCDL is non-inverting. This topology maintains the same resolution of previous solution, while it can easily verified that the minimum delay is now. The non-inverting DCDL of Fig. 8, therefore, maintains the same performances of the NAND-based DCDL of previous papers while avoiding its glitching problem.

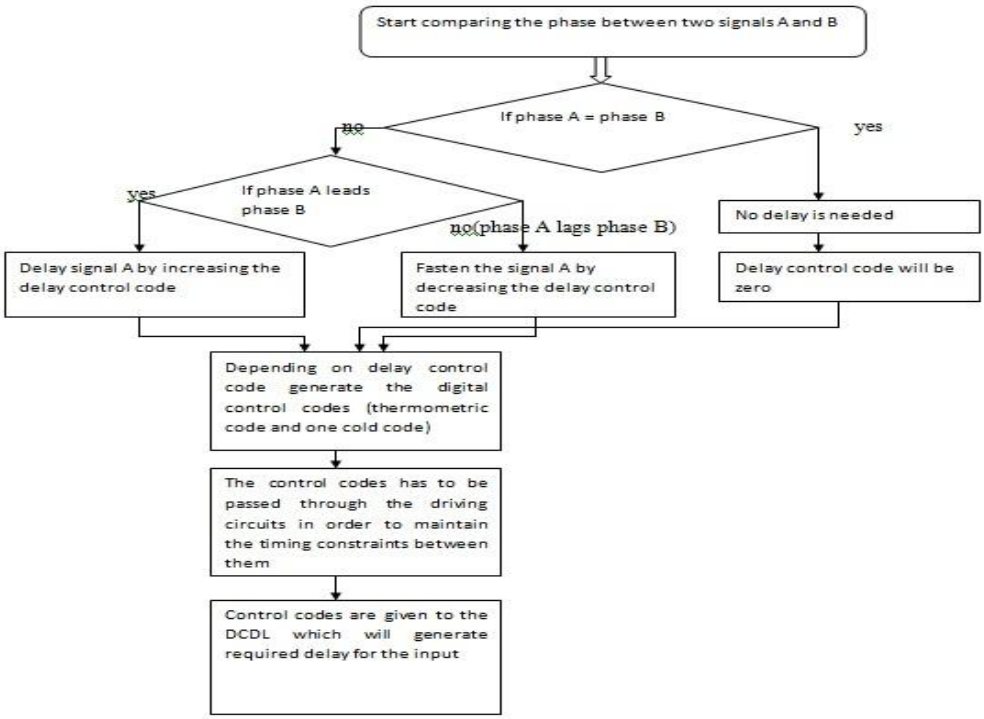

Figure 9: Flow chart of Proposed Glitch free NAND based DCDL working

1.1. Glitch-free switching of proposed DCDL and control-bits driving circuits:
In the previous section we have seen that the glitch-free operation of the proposed DCDL can be obtained with a three-step switching mechanism: for a switching from a delay control code c=k to a delay control code c=h, first, the k+1th DE is switched from post-turn-state to the turn-state; next all DE are switched from pass to turn-state (or vice versa) and finally the h+1th DE is switched to post-turn-state. This switching mechanism presents the drawback of being slow and can result in a not simple driving circuit for the DCDL control-bits.

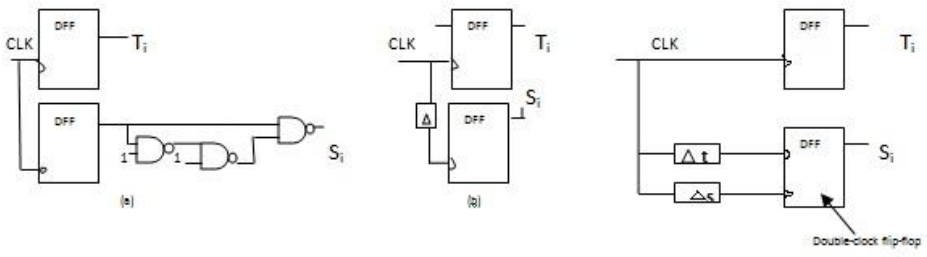

Figure 10: Possible driving circuits for the control-bits of proposed DCDL: (a) signals delayed with different LH/HL delays by using a NAND-based circuit; (b) signals delayed using clock-tree delay (c) Si delayed with different LH/HL delays by using clock-tree delay and double-clock flip-flops

B. MUX based DCDL:

The switching mechanism of above DCDL presents the drawback of being slow and can result in a not simple driving circuit for the DCDL control-bits. And as the driving circuits are implemented using flip-flops the area and power consumption will be more.

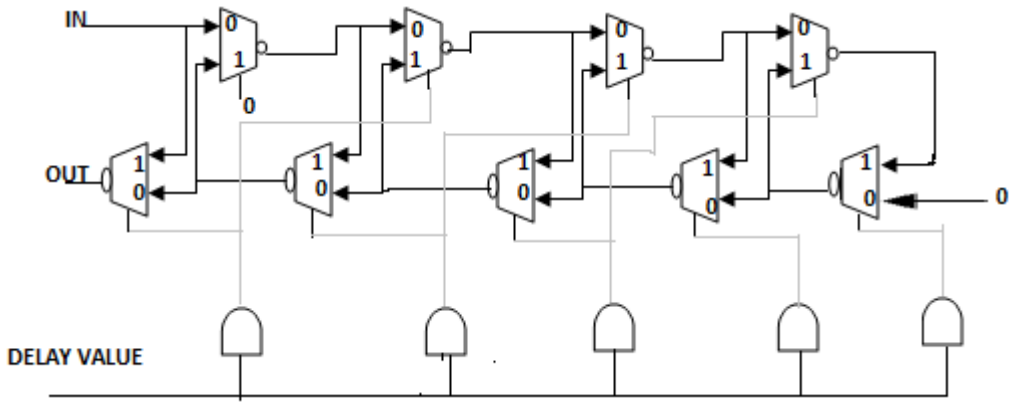

Figure 11: MUX based DCDL

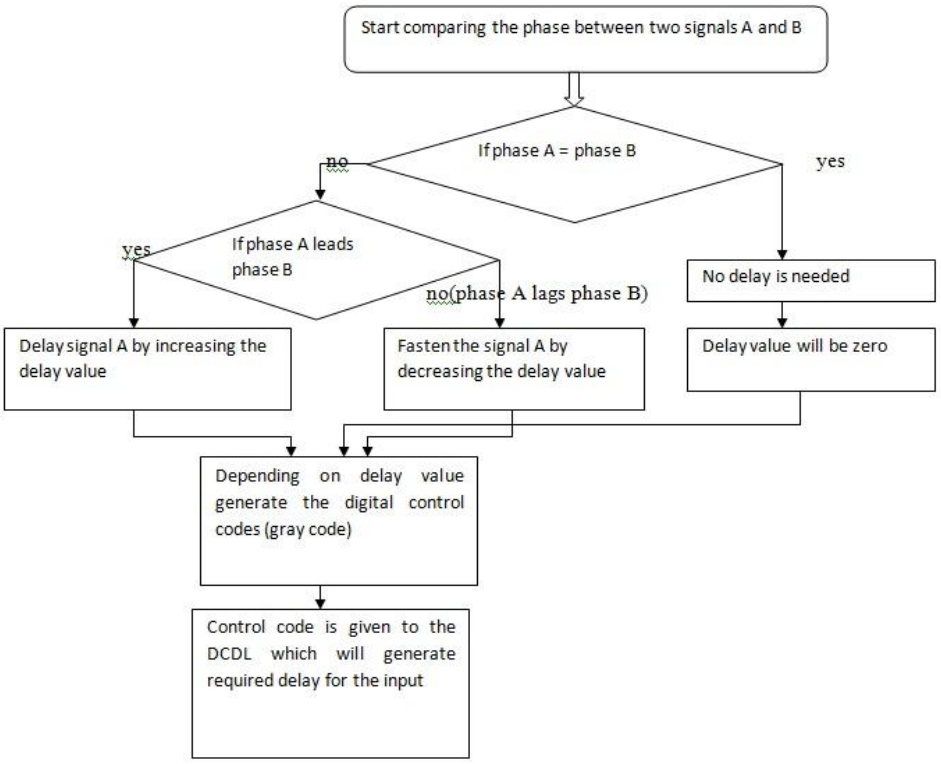

Figure 12: Flow chart of MUX based DCDL working

To reduce the larger area henceforth the larger power consumption the proposed methodology with multiplexers using gray code is proposed. This method also eliminates the use of driving circuits i.e., the NAND gates followed by D flip-flops. This architecture replaces 6 NAND gates (4 active + 2 dummy) with just 2 multiplexers+1[2] for providing the turn state. One of the inputs of the last MUX is kept at '0' ensuring that the n-1th state will always carry the data present at the input '1' of the nth bit. The control bits are based on the gray code.

In MUX based DCDL, 2:1 multiplexer's acts as delay elements. Each delay cell consists of two multiplexers and an AND gate. The delay of the DCDL is controlled by delay value which is generated by delay code controller. The delay value is converted into gray code and given to the DCDL. Depending upon the gray code the delay elements are selected and thus the delay of the DCDL is controlled by the delay control code. In the above figure 8, the DCDL composed of five delay cells chain. Each multiplexer is provided with an input terminal, an output terminal, and a selection line terminal. The delay control code is passed through the AND gate and then given to the respective multiplexers. The above DCDL can take the decimal values from 0 to 10 and by using its respective gray code, the delay is produced.

Simulation Results

A. In this below simulation, the output of NAND based DCDL is shown. The NAND based DCDL is producing glitches a the output due to switching of delay control codes

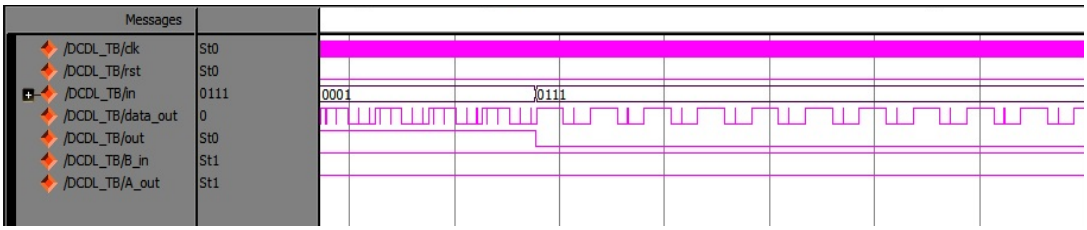

Figure 13: Simulation result of NAND based DCDL in existing system

*B.*The below simulation shows tha output of Glitch free NAND based DCDL. The output of proposed DCDL is glitch free even during the switching of delay control codes. Here 'in' is the input and 'out' is the output. S and T are the control codes.

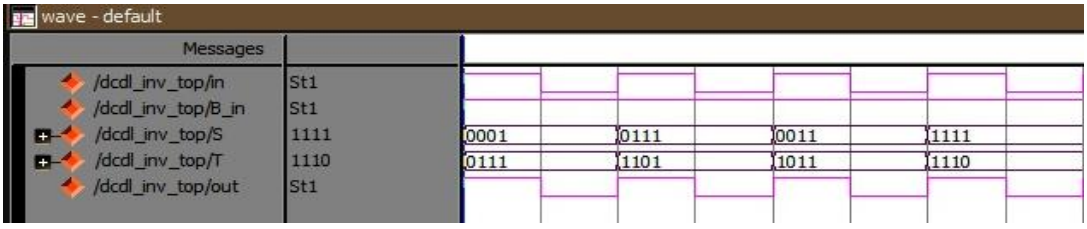

Figure 14: Simulation result of Glitch free NAND based DCDL

C. In figure 12, simulation result of MUX based DCDL is shown. Here 'in' is the input of the system and 'out' is the output of the system where as S is the control code. In MUX based DCDL only one control code is used and produces glitch free output.

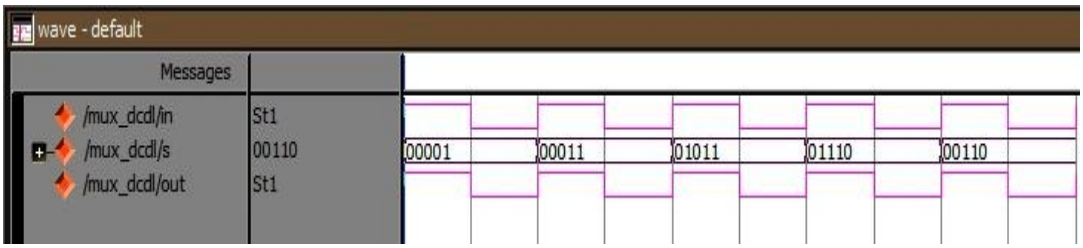

Figure 15: Simulation result of MUX based DCDL

D. The below simulation result shows the output of an proposed ADPLL using Glitch free NAND based DCDL as the programmable delay circuit.

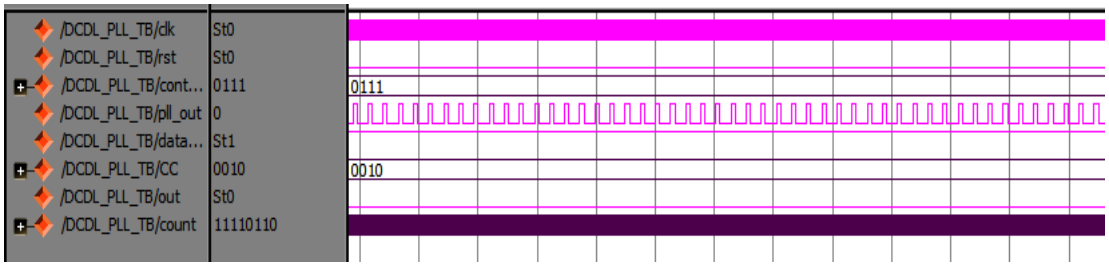

Figure 16: Simulation result of ADPLL using Glitch free NAND based DCDL

E. The below simulation result shows the output of an proposed ADPLL using MUX based DCDL as the programmable delay circuit.

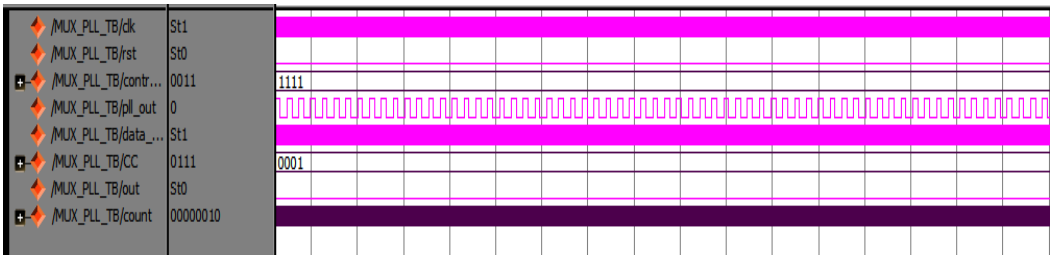

Figure 17: Simulation result of ADPLL using MUX based DCDL

Performance Analysis

In table II, the area, power consumption and total number of transistors used per one delay cell of proposed DCDLs is presented. In table III, the number of logic elements and maximum frequency of implemented ADPLLs is presented.

Table 2: Comparison Of Glitch Free Nand Dcdl And Mux Based Dcdl

Proposed DCDLS	**Power Consumption (MW)**	**Total Delay (NS)**	**No of Loigic Transisitors Per Delay Cell**
Glitch free NAND based DCDL	67	15.97	24
MUX based DCDL	46	12.03	18

Table 3: Comparison of Adpll Using Glitch Free Nand Based Dcdl And Adpll Using Mux Based Dcdl

Adplls	**No Of Logic Elements Used**	**Maximum Frequency (Mhz)**
ADPLL using GLITCH free NAND based DCDL	815	69.71
ADPLL Using MUX based DCDL	760	74.46

Performance Analysis Chart

A. *Comparison of area and power consumption between NAND based DCDL and MUX based DCDL:* The area has been reduced by 5% and power has been reduced by 3.33% than the existing NAND based delay lines.

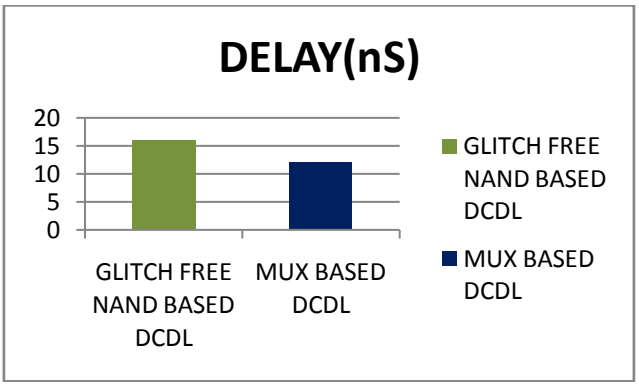

Figure 10: Power (mW) comparison for NAND and MUX based DCDL

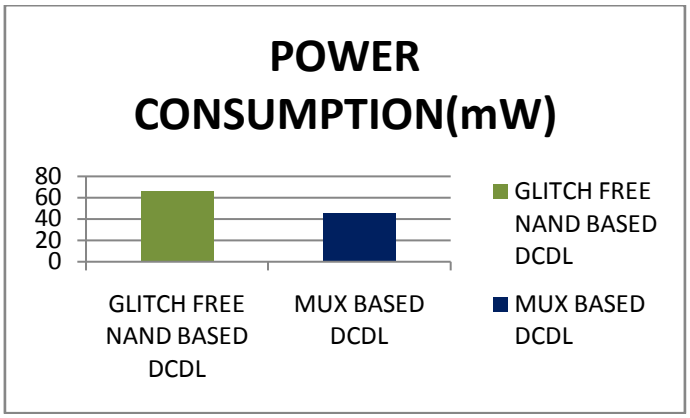

Figure 11: Comparison of power consumption between glitch free NAND based DCDL and MUX based DCL

B. Comparison between ADPLL using glitch free NAND based DCDL and ADPLL using MUX based DCDL: From the chart, it is clear that ADPLL using MUX based DCDL requires less area and less power consumption compared to other ADPLL.

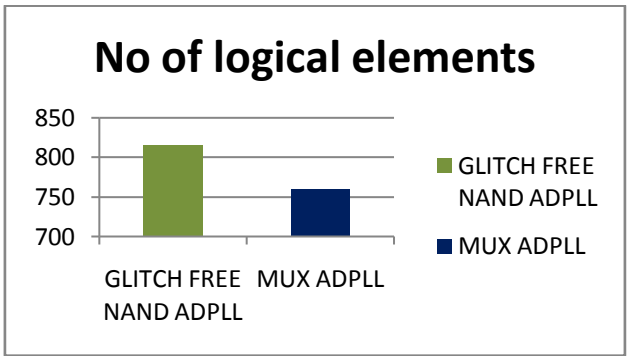

Figure 12: No of logical elements comparison for MUX ADPLL and NAND ADPLL

Conclusion

In this paper a new all digital phase locked loop is presented. Two new programmable delay circuits are proposed to provide glitch free outputs and implemented in ADPLL separately. When the reference clock has large jitter, the proposed digital loop filter can eliminate the reference clock jitter effects and reduces the period jitter of the output. Each proposed DCDL is analyzed and simulation results are presented. The analysis of proposed ADPLL with proposed programmable delay line circuits is presented. The proposed ADPLL can perform a precise clock generation from a very noisy and low frequency reference clock.

References

[1]. Dian Huang and Ying Qiao "A Fast-Locked All-Digital Phase-Locked Loop for Dynamic Frequency Scaling" *IEEE Journal of SolidState Circuits*, vol. 40, pp. 2469-2482, Dec. 2012

[2]. Davide De Caro, Senior Member, IEEE, "Glitch-Free NAND Based Digitally Controlled Delay-Lines" IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 21, No. 1, January 2013.

[3]. Jean Barbier, Montpellier, "Programmable Delay Line Circuit With Glitch Avoidance", Patent published on sep.30,2010.

[4]. K. Sungjoon, K. Lee, Y. Moon, D.-K. Jeong, Y. Choi, and H. K. Lim, "A 960-Mb/s/pin interface for skew-tolerant bus using low jitter PLL," *IEEE J. Solid-State Circuits*, vol. 32, no. 5, pp. 691–700, May 1997.

[5]. Kuo-Hsing,W.-B. Yang and C.-M. Ying, "A dual-slope phase frequency detector and charge pump architecture to achieve fast locking of phase-locked loop," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 50, no. 11, pp. 892–896, Nov. 2003.

[6]. H. Eisenreich, C. Mayr, S. Henker, M. Wickert, and R. Schüffny "A Programmable Clock Generator HDL Softcore"

[7]. "wide frequency range delay locked loop" United States Patent Application Publication 2010.

[8]. L.Wang, L. Liu, and H. Chen, "An implementation of fast-locking and wide-range 11-bit reversible SAR DLL," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 6, pp. 421–425, Jun. 2010.

[9]. J. S. Wang, C. Y. Cheng, J. C. Liu, Y. C. Liu, and Y. M. Wang, "A duty cycle distortion tolerant half delay line low-power fast lock in all digital delay locked loop," *IEEE J. Solid-State Circuits*, vol. 45, no. 5, pp. 1036–1047, May 2010.

[10]. S. Damphousse, K. Ouici, A. Rizki, and M. Mallinson, "All digital spread spectrum clock generator for EMI reduction," *IEEE J. Solid- State Circuits*, vol. 42, no. 1, pp. 145–150, Jan. 2007.

[11]. D. D. Caro, C. A. Romani, N. Petra, A. G. M. Strollo, and C. Parrella, "A 1.27 GHz, all digital spread spectrum clock generator/synthesizer in 65 nm CMOS," *IEEE J. Solid-State Circuits*, vol. 45, no. 5, pp. 1048–1060, May 2010.

[12]. K. H. Choi, J. B. Shin, J. Y. Sim, and H. J. Park, "An interpolating digitally controlled oscillator for a wide range all digital PLL," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 56, no. 9, pp. 2055–2063, Sep. 2009.

[13]. C. C. Hung et al, "A 40-GHz Fast-Locked All-Digital Phase-Locked Loop Using a Modified BangBang Algorithm," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 58, pp. 321-325, June 2011.

[14]. D.-S. Kim et al. "A 0.3-1.4 GHz All-Digital Fractional-N PLL With Adaptive Loop Gain Controller," *IEEE Journal of Solid State Circuits*, vol. 45, pp. 2300-2311, Nov.2011

[15]. C.-C. Chung et al. "A Fast Tracking ADPLL for Video Pixel Clock Generation in 65nm CMOS Technology", *IEEE Journal of Solid State Circuits*, vol. 40, pp. 2300-2311, Oct.2011

[16]. R. J. Yang and S. I. Liu, "A 40–550 MHz harmonic-free all digital delay locked loop using a variable SAR algorithm," IEEE J. Solid-State Circuits, vol. 42, no. 2, pp. 361–373, Feb. 2007.

[17]. Mohammad Maymandi-Nejad and Manoj Sachdev,"A Digitally Programmable Delay Element:Design and Analysis" IEEE transactions on very large scale integration (vlsi) systems, vol. 11, no. 5, october 2003.