DESIGN, ANALYSIS AND SIMULATION OF A JITTER REDUCTION CIRCUIT (JRC) SYSTEM AT 1GHz

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

by

Run Bin Yu

December 2016

COMMITTEE MEMBERSHIP

TITLE:	Design, Analysis and Simulation of a Jitter Reduction Circuit (JRC) System at 1GHz
AUTHOR:	Run Bin Yu
DATE SUBMITTED:	December 2016
COMMITTEE CHAIR:	Tina H Smilkstein, Ph.D. Associate Professor of Electrical Engineering
COMMITTEE MEMBER:	Dennis Derickson, Ph.D. Associate Professor of Electrical Engineering Department Chair of Electrical Engineering
COMMITTEE MEMBER:	Vladimir Prodanov, Ph.D. Associate Professor of Electrical Engineering

ABSTRACT

Design, Analysis and Simulation of a Jitter Reduction Circuit (JRC) System at 1GHz

Run Bin Yu

The clock signal is considered as the “heartbeat” of a digital system yet jitter which is a variation on the arrival time of the clock edge, could undermine the overall performance or even cause failures on the system. Deterministic jitter could be reduced during the designing process however random jitter during operation is somehow less-controllable and unavoidable. Being able to remove jitter on the clock would therefore play a vital role in system performance improvement.

This thesis implements a 1GHz fully feedforward jitter reduction circuit (JRC) which can be used as an on-chip IP core at clock tree terminals to provide a low jitter clock signal to a local clock network or be used at the clock insertion point to reduce jitter from an off chip signal. It can also be stand-alone and used on PCB designs to reduce jitter on the high-frequency clock signal used on the board. This jitter attenuation circuit is implemented using IBM CMHV7SF 180nm MOSFET process, demonstrates a jitter reduction of at least 8dB at 1GHz with 33ps rms Gaussian random jitter (for a 200ps peak-to-peak randomly changing rising edge input signal).

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Tina Smilkstein for all efforts she has put on this thesis. Tina is best known for her dedication to students, as soon as I come up with any question or challenge she is always around to help.

I'd also like to thank Dr. Derickson and Dr. Prodanov for being the thesis committee members to review my work and provide helpful feedback. I have been in Dr. Derickson and Dr. Prodanov's classes which are also big inspirations to accomplish this work. Huge appreciations to Cal Poly Electrical Engineer department and faculties for all supports during these years.

I want to appreciate my parents, family members and friends for supporting me for years on my path of being educated.

Finally a deep appreciation to Cal Poly CSSA president Shaozhang Wu (Ah-Zhang) and all the members, along with all the memorial moments shared with these lovely homies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Page

Chapter 1

INTRODUCTION

Modern IC design is rapidly moving to a highly integration scope which requires much smaller devices. A huge advantage of using smaller devices is to boost up the speed of the system in other words a higher system clock, yet a faster clock brings a challenge into design. In an ideal case, digital system clock has a constant frequency to have all circuitries be synchronized with an evenly-separated time period. In real cases, the system clock period usually contains some small random variations between each cycle therefore all synchronized devices must be able to tolerance the difference between the actual clock period and its nominal value, which is called jitter. Random jitter is independent of the clock period value so for systems with lower clock frequency, jitter might have less disturbing effects. For example a ±100ps deviation on a nominal 16MHz clock (a typical value for some general purpose microcontrollers) is a ±0.16% clock period variation which the system could tolerance. But the same amount of deviation would be ±10% clock period variation for a 1GHz system (while most of the processors now are running at multiple gigahertz) that could cause series errors. As a consequence while we are pushing up the system speed in our designs, jitter influence becomes more destructive and is worthy of a dedicated solution.

1.1 Jitter definition, metrics and effects

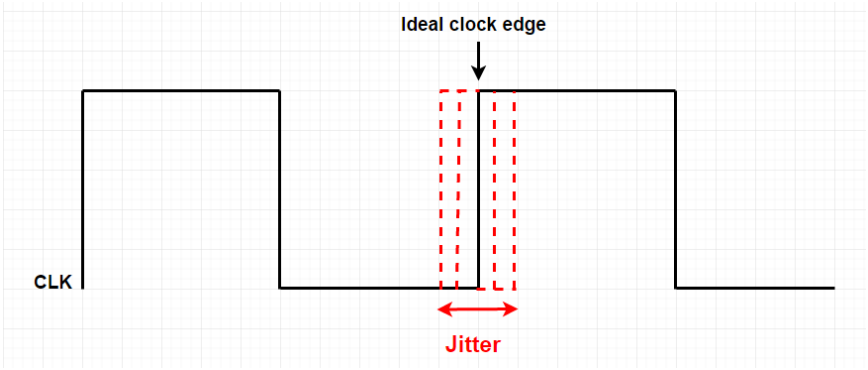

Figure 1.1 Clock rising edge jitter

Clock jitter is the deviation from the true periodicity of a reference clock signal [23], as the variation of the clock rising edge compared to its ideal timing shown in Figure 1.1. There are two major types of clock jitter: deterministic jitter and random jitter [24]. Deterministic jitter is somehow more "predictable" and "controllable" which usually caused by the process variation or design decisions like wire length and/or size of buffers and other devices, yet the random jitter due to the capacitive coupling and interference effect between circuitry modules and wires, as its name implies, is less predictable. In other words we are able to determine how far the actual rising edge is away from its ideal position for the deterministic jitter case, but we cannot predict where the actual rising edge is located for the random jitter cases because it is varying randomly.

To quantify random jitter, we need to use its characteristic of following the Gaussian distribution and describe it with mean μ_{jitter} and standard deviation σ_{jitter} (root mean square, or rms) [24]. Consider the jitter shown in Figure 1.1 as Gaussian random jitter, 99.7% of the possible rising edge deviation from its ideal location falls into the range of $\pm 3\sigma_{jitter}$ with a μ_{jitter} of 0, a typical Gaussian distribution shown in Figure 1.2.

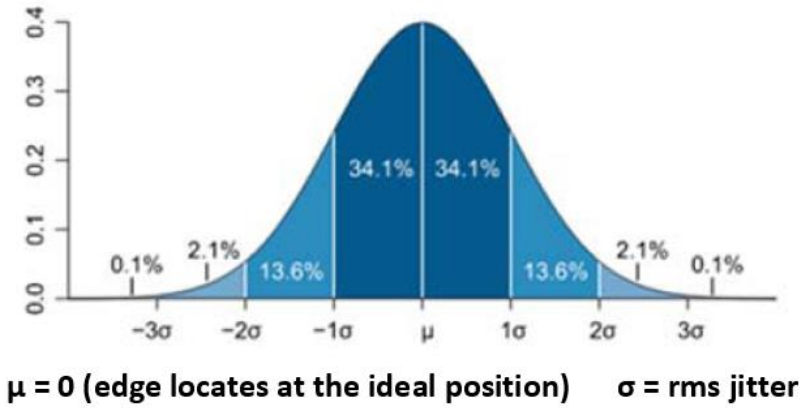

Figure 1.2 Typical Gaussian random jitter distribution

Three commonly used jitter metrics are absolute jitter, cycle-to-cycle jitter and period jitter [24]. Absolute jitter is the absolute difference in the position of a clock's edge from where it would ideally be. Cycle-to-cycle jitter is the difference in duration of any two adjacent clock periods. Period jitter is the difference between any one clock period and the ideal or average clock period. For digital systems with synchronous circuitries, period jitter is the metric that being watched most since the performance is set by the average clock period, and the error-free operation is limited by the shortest possible clock period.

In the simplified digital system model shown in Figure 1.3, the average clock period T must be at least greater than the sum of the propagation delay of the first register $t_{D\rightarrow Q}$, the propagation time t_{d_logic} of the logic circuitry and the setup time of the last register, to ensure the input data has been processed and be ready for the second register to pick up before clock ticks. This determines the minimum clock period of the system, or the maximum frequency to be $f_{max} = \frac{1}{t_{D\rightarrow Q}+t_{d\_logic}+t_{setup}}$ to avoid setup time violation. Yet if jitter presents at the rising edge of the clock, to avoid the setup up time violation, the maximum frequency reduces to $\frac{1}{t_{D\rightarrow Q}+t_{d\_logic}+t_{setup}+t_{jitter}}$ where t_{jitter} is not controllable. Similarly the ideal maximum hold time for the system is $t_{D\rightarrow Q} + t_{d\_logic}$

to ensure the second register being ready to accept new data, but taking jitter into account, it becomes $t_{D\rightarrow Q} + t_{d\_logic} + t_{jitter}$ where t_{jitter} is randomly varying and could cause hold time violation.

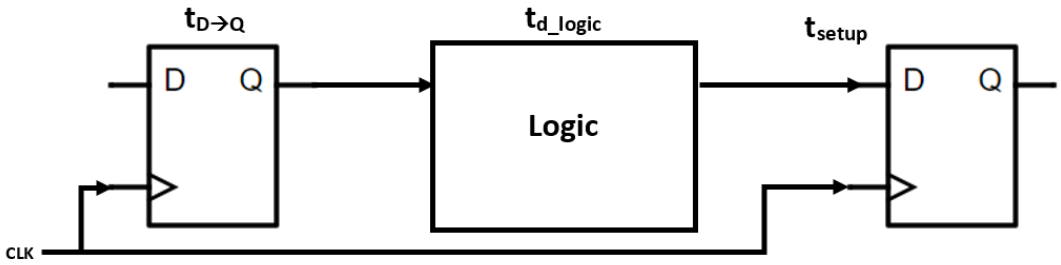

Figure 1.3 Simplified digital system model

In the analog-to-digital process, the analog signal is designed to be sampled at evenly-spaced separation timing points, which determined by the period sampling clock. Every sampled analog signal quantity is taken at a specific rising (or falling) edge of the clock signal. Therefore if the triggering edge arrives randomly, the actual sampled value is not the desired one but something we cannot predict. As a consequence the samples used for later processing miss some useful data but contain some "unwanted" data, and the reconstruction is distorted or completely off.

These two examples of how the period jitter could hurt the synchronous digital system or an ADC. Having a dedicated solution to reduce the period jitter on system's clock is clearly beneficial.

1.2 Jitter removal solutions

1.2.1 Phase-Locked loop (PLL)

Phase-locked loops (PLL) provide some jitter removal due to the frequency control signal pass through a low-pass filter but their main application is to synthesize its output frequency to match reference frequency.

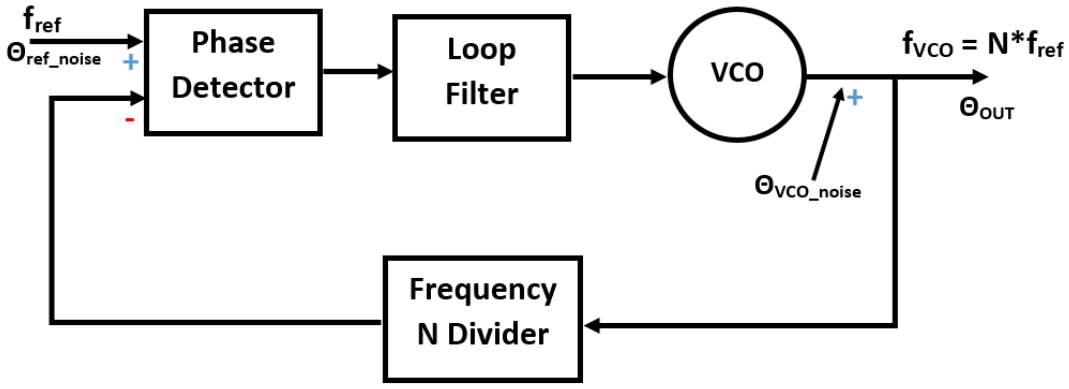

Figure 1.4 Typical phase-locked-loop block diagram

Figure 1.4 shows the typical block diagram of a PLL system, which includes three forward-gain blocks: phase detector, loop filter, voltage-control-oscillator (VCO) and a feedback frequency N-divider block. The phase detector block compares the phase difference between the reference frequency and the divided-by-N version of the output frequency. If the frequencies don't match, it alters a control signal (which controls the frequency out of the VCO). The control signal is generated by averaging (low-pass filtering) the output of the phase detector block and then converting that information on the difference in phase to an appropriate value to control the VCO. This process continues till the VCO output matches the reference frequency with phases aligned. Because of the low pass filter after the phase detector, the output phase noise with respect to the reference input phase noise is also low-pass filtered, and high frequency jitter at the reference input is reduced. If we set the frequency division number N to 1 and connect the jittery 1GHz clock to the reference input, the high-speed rising clock edge jitter is blocked by the filter in the loop. This implies the output clock would be a perfect 1GHz if the loop filter has its bandwidth low enough

to block all frequencies down to 0Hz jitter at the input clock. However the VCO also contains phase noise and the transfer function of the output phase noise with respect to the VCO phase noise, $\frac{\theta_{out\_noise}}{\theta_{VCO\_noise}}$ has high-passing nature because the low-passing loop filter now is included in the feedback gain. As a consequence, any high-frequency jitter produced by the VCO shows up at the output directly. To suppress VCO jitter's appearance at the output, the bandwidth of the low-pass filter after the phase detector needs to be increased. But then more jitter from the input could get through and show up at output. Also another drawback of lowering the loop filter bandwidth is a longer settling time of the PLL, which might be problematic in the cases require fast response during frequency variations. Also, the phase error could possibly be built-up through the feedback loop and addressing that problem complicates the design too. Some other works [6,7] show how complicated it would be to design high-frequency PLL such as 1GHz generation, with lower supply voltages such as 1.8V (compared to some market available discrete PLL chips, which usually have supply higher than 3V). By putting more appropriate constraints on a PLL's output jitter performance [8], the design would become even more complicated due to the amount of effort it would take to optimize the loop parameters.

1.2.2 Jitter Reduction Circuit (JRC)

An alternative jitter removal solution being presented in this thesis, is the Jitter Reduction Circuit system (JRC). Compared to the PLL solution mentioned above, the JRC solution requires less complexity on design, using some basic analog and digital modules. Unlike PLL, the JRC system does not include any oscillator which could contribute jitter on the output and therefore reduces the amount of less-controllable jitter generated by the system. Also, due to the simplicity of the JRC implementation, it could be a good candidate for an on-chip solution, being integrated into the SoC. Another one of its advantages is that it is a purely feedforward design and therefore does not experience a build-up of error.

The JRC system is fully dedicated to jitter reduction and does not maintain the phase alignment between the input and the output clock. Yet, if the JRC is used as the clock correction stage at the top of the clock distribution network, at the clock insertion point or global clock PLL is located, then we do not need the phase alignment anymore.

This design implementation is based on the research by Dr. Tina Smilkstein's of "Jitter Attenuation Circuit" which can be found in [1]. The work done in this thesis uses a different fabrication process and some modifications to the original implementation by Dr. Smilkstein. Chapter 2 introduces the basic theories that the JRC system is based on, and delivers overviews on the system and the sub-blocks. Chapter 3 through Chapter 6 analyze individual sub-block of the system along with design aspects and test results. Chapter 7 includes the system-level simulations and analyses, and Chapter 8 concludes the overall performance of the JRC plus some potential topics for future improvements.

Chapter 2

JRC OPERATION THEORY AND SYSTEM OVERVIEW

2.1 Basic theory

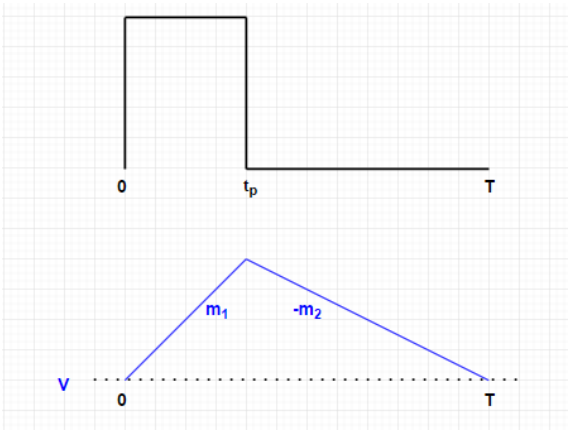

Figure 2.1 Linear integration of a square-wave with pulse width of t_p

The underlying basic of the JRC system described in this work follows the working theories of a "Jitter Attenuation Circuit" by Dr. Tina H. Smilkstein in [1] and "Anti-Jitter Circuit" by Michael J. Underhill in [2,3,4,5].

Figure 2.1 shows a linear integration of a square-wave with a pulse width of t_p and a period of T. During t_p where the square-wave has its high output level, the integration waveform ramps up with a positive slope of m_1. On the other hand as soon as the square-wave has its low output level, the integration waveform ramps down with a negative slope of $-m_2$. By setting m_1 and $-m_2$ to satisfy the relationship that:

$$\mathbf{m_1 \times t_p - m_2 \times (T - t_p) = 0} \qquad \text{(Eqn 2.1.1)}$$

then the integration waveform will have its beginning voltage level same as its ending voltage level during one period of T. In Figure 2.1 the rising pulse starts exactly at the beginning of the period, say t=0. But now let's consider a situation with the pulse starts a bit later than the beginning of the period, say $t=t_d$, and we watch the same duration of one period from t=0 to t=T, as shown in Figure 2.2.

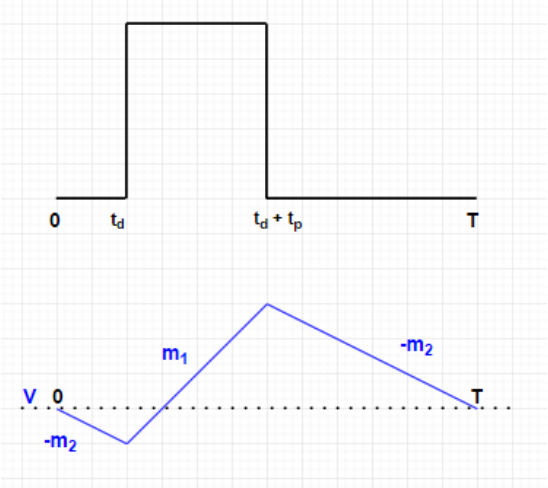

Figure 2.2 Linear integration of a square-wave with t_p starts with a delay t_d

Consider t_d as the delay of the rising edge of the pulse t_p we have seen in Figure 2.1, the total time of low output level is $t_d + [T - (t_d + t_p)] = (T - t_p)$, which is actually same as the case without delay. Therefore Equation 2.1.1 still applies to this case with delay t_d presents:

$$-m_2 \times t_d + m_1 \times t_p - m_2 \times [T - (t_d + t_p)] = m_1 \times t_p - m_2 \times (T - t_p) = 0$$

Hence the integration waveform still has its beginning and ending voltage level equal, at exactly one period from 0 to T. This implies as long as we have the three parameters in Equation 2.1.1, m_1, m_2 and t_p being fixed, then t_d does not affect the beginning and the ending voltage level of the integrating waveform. The delay t_d in fact could be considered as a jitter t_{jitter} on pulse's rising edge, which could be randomly varying between each rising edge. As long as a starting voltage is known, the exact point where a period has ended can be found by looking for that same voltage on its second crossing. Let's fix that voltage and see what problems can occur. Figure 2.3 shows the case with jitter presents at multiple rising edges with all pulses maintaining a width of t_p. The 2nd pulse comes later than the expected location and the 3rd pulse on the other hand comes earlier than it should be. If we choose the beginning voltage level of the 1st pulse as the reference level, trying to verify the statement we just made in the case shown in Figure 2.2, the result fails. This is because

the integration waveform by the 2nd pulse does not cross reach the reference level. Similar missing cross-point failure could happen if the reference level is selected to be too high.

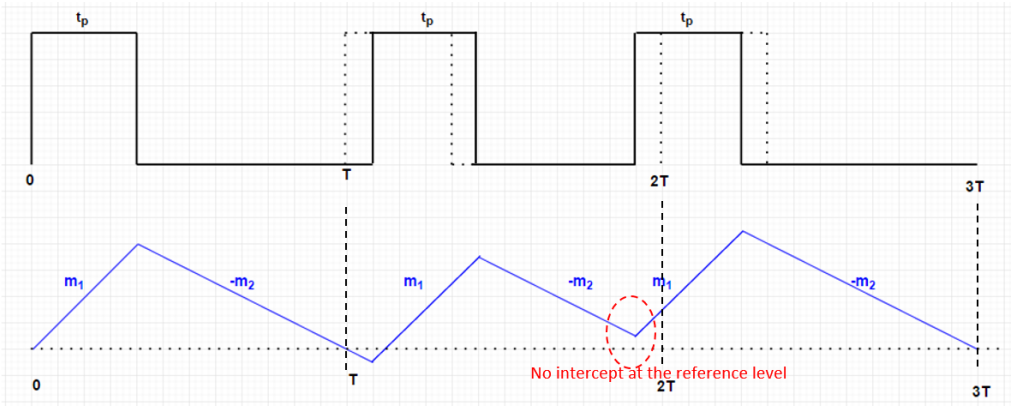

Figure 2.3 Integration waveform with jitter presents, with inappropriate reference level

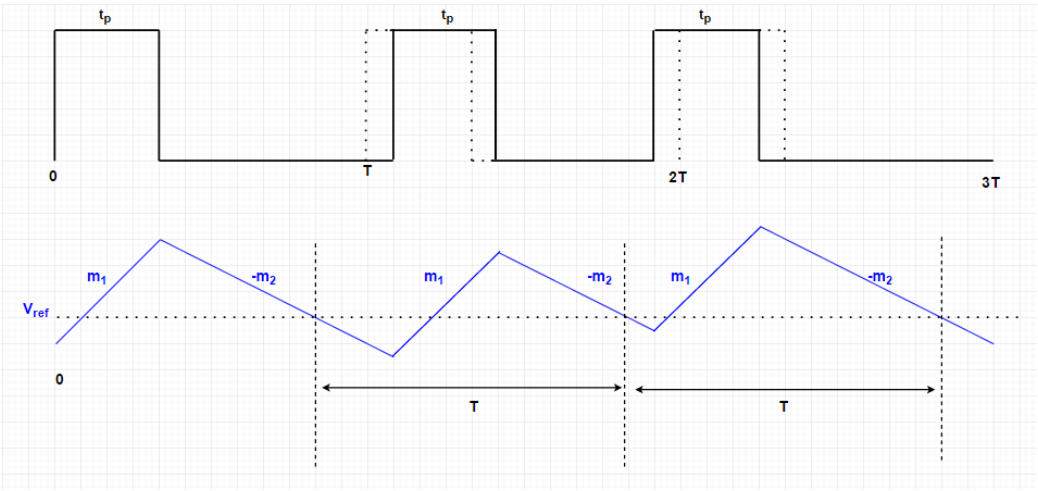

Figure 2.4 Same integration waveform as shown in Figure 2.3, appropriate reference level

If the reference level is adjusted, the problem is fixed as shown in Figure 2.4, and now, all of the $-m_2$ legs cross the reference level. This means as long as an appropriate reference voltage level is set to have the integration waveform by each t_p pulse cross, in other words, as long as the reference, m_1, m_2 and t_p, are set correctly the point where $V_{out} = V_{ref}$ on the $-m_2$ leg is exactly one period T, despite the jitter (with one limitation that will be described below). The V_{ref} value is valid

as long as–m_2 legs consistently cross it and this means that V_{ref} has multiple solutions. For example if we push the V_{ref} up a bit more (but still have the $-m_2$ legs cross over it for each cycle) in Figure 2.4, the new crossing points still have a period of T and the only difference is where the period starts. Because of the randomness of the rising edge jitter and multiple solutions of V_{ref}, it is difficult determine one specific reference voltage needed for period sensing with a single-end signal.

Now let's put a complementary copy of the single-end integrating waveform to be on top of the signal described above, as shown in Figure 2.5. Instead of searching for a "good" reference voltage level, the differential integrating waveforms have predictable crossing points on the $\pm m_2$ legs which can be used as period sensing triggers. These crossing point locations are not affected by the existence of jitter, as long as the jitter is within an allowable range. The calculation of crossing point locations and the allowable jitter range will be covered in the next chapters.

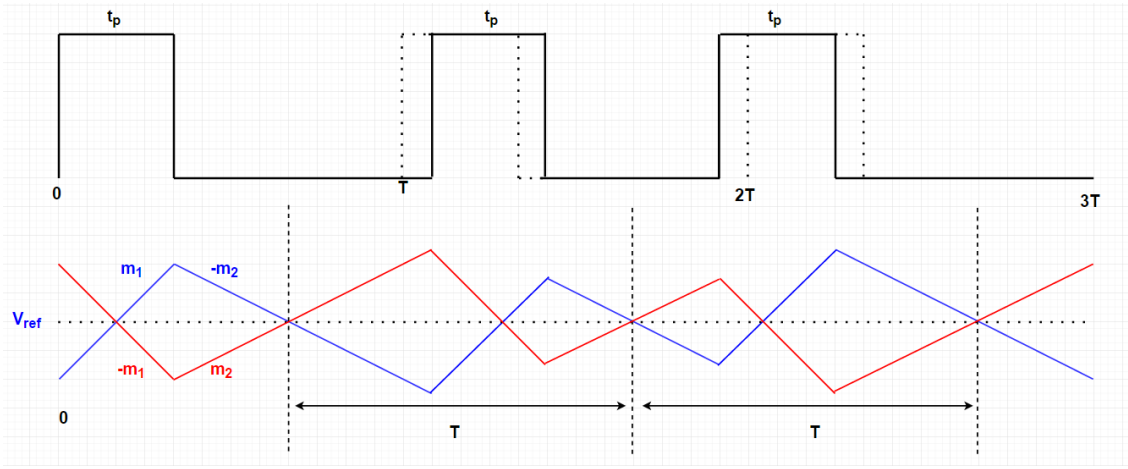

Figure 2.5 Differential integrations with the crossing points for period sensing

2.2 Jitter Reduction Circuit (JRC) system overview

The JRC system is based on the basic theory model established above, including the setup of constant pulse width t_p and the appropriate integrating slopes m_1 and m_2 that satisfy the fundamental equation Eqn 2.1.1. In general, the JRC accepts a jittery clock with a nominal period of T as input and generates a pair of complementary constant-width pulse chains each time it sees a rising edge of the jittery clock. The system then performs the differential integration to set up the crossing points which indicate a period of T. Finally the system senses the crossing point locations and generates the recovered clock.

This process requires four major sub-blocks: a pulse generation block that generates constant-width pulses to represent each clock rising edge it sees; an auto-biasing block that sets up appropriate biasing information to set $\pm m_1$ and $\pm m_2$ for the differential integrating waveforms; a integrating block that accepts the timing information provided by the pulse generation block and the biasing information provided by the auto-biasing block, to produce differential integrating waveforms; and a output reforming block which senses the integration crossing points and regenerates a clock signal with the correct period of T. Figure 2.6 shows the high-level block diagram of the JRC system.

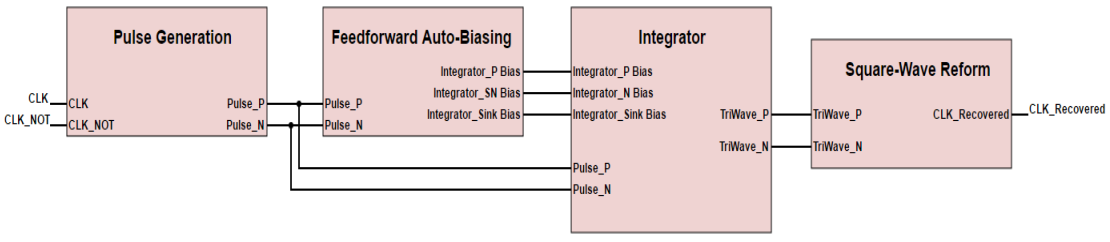

Figure 2.6 JRC high-level block diagram

2.2.1 Pulse generation block description

The constant-width pulses generated by the pulse generation block will be used as switching controls of the differential integration. For a single-ended integration implementation, one pulse chain with constant width t_p is enough. Yet since we have the integration in a differential fashion, it is necessary to have a complementary version of the pulse chain, (which will be call the t_p' in this thesis). While the t_p chain has a constant high level width of t_p, to control one of the differential integrating waveforms, the t_p' chain then must have a constant low level width of t_p to control the complementary integration. In other words, while one integrating waveform is ramping up with a slope of m_1 during t_p, its complement is ramping down with a slope of $-m_1$. Similar relationship applies to t_p pulse in the other direction: one ramps down with the constant slope of $-m_2$ but the other one ramps up with a slope of m_2.

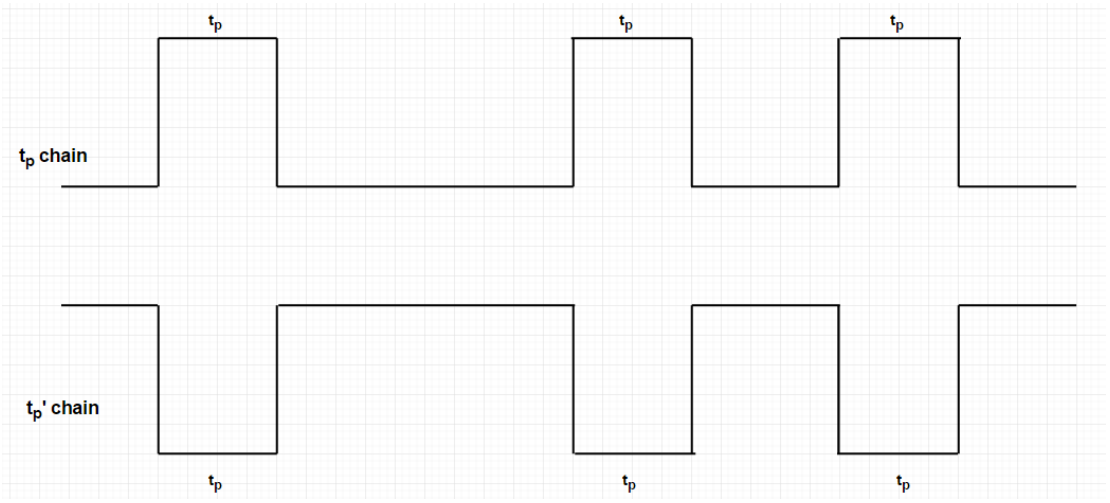

Figure 2.7 Ideal complementary pulse chains by the pulse generation blcok

2.2.2 Feedforward auto-biasing block and integrator block description

The feedforward auto-biasing block is responsible for setting up the biasing and control information needed by the integrator which uses constant currents to charge or discharge capacitors for the integrator output. Constant currents are supplied by the current sources and sinks within the integrator, to charge the capacitor for ramping up voltage integration and to ramp it down. Figure 2.8 shows a general implementation of the integrator.

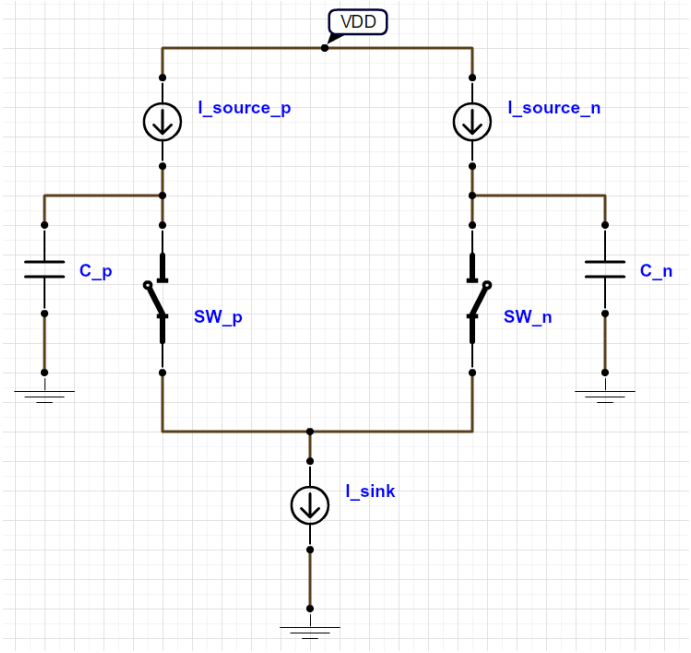

Figure 2.8 Integrator implementation in general

On each branch of the differential integrator, the switch is controlled by pulses from the pulse generation block. Since we have the pulse chain from the generation block being differential, only one of the two switches is conducting at one time. As soon as the left switch is opened (not conducting), the top left current source supplies constant current $I_{charging}$ to charge up the left capacitor. Because of this the voltage at that capacitor ramps up with a slope of m_1. Meanwhile the switch on the right is closed (conducting) and has a current from the supply minus the tail current sink so the voltage ramps down with a slope of $-m_1$. The same thing happens in the other direction

where, when the right capacitor is charged up at the rate of m_2, the left capacitor ramps down as $-m_2$.

To see how this works out mathematically, replace m_1 and m_2 with the currents that the capacitors see when they are charging and discharging. Replace m_1 and m_2 with $I_{charging}$ and $I_{discharging}$ respectively in the fundamental equation, it becomes $I_{charging} \times t_p - I_{discharging} \times (T - t_p) = 0$. Solving for the ratio of the currents, we have $\frac{I_{charging}}{I_{discharging}} = \frac{T-t_p}{t_p}$, $\frac{I_{charge}}{I_{sink}} = \frac{T-t_p}{T}$ and $\frac{I_{discharge}}{I_{sink}} = \frac{t_p}{T}$. This means the two sourced currents are directly related to the duty cycle of the constant width pulsing signal. This means that the auto-biasing block needs to convert the duty cycle into biasing controls to set up the correct current ratio between the current sources in the integrator. In addition, the biasing block is responsible for biasing the current sink in the integrator so it constantly conducts the sum of the two sourced currents.

2.2.3 Square-wave output reform block description

Square-wave reform block has two major functionalities: differential integration crossing point sensing and generation of the square-wave clock output.

The crossing points of the differential integrating waveforms provides timing information on when to generate a rising edge. An ideal clocking signal needs to be sharp at its rising and falling edges so the system will be synchronized at the same specific moment for each clocking cycle.

The square-wave output reform block takes the differential integrating triangular waveforms as inputs and creates a sharp-rising-edge pulse as soon as it finds a crossing point, and therefore a pulse chain with the corrected period T. Figure 2.9 shows an example of the ideal square-wave output reform block's input-output relationship, with jitter showing up at the input of the JRC system (therefore the pulse generation output pulses contain jitter as shown). V_+ and V-

represent the differential triangular outputs from the integrator, and the square-wave outputted by the reforming block, therefore can be used as the recovered clocking signal.

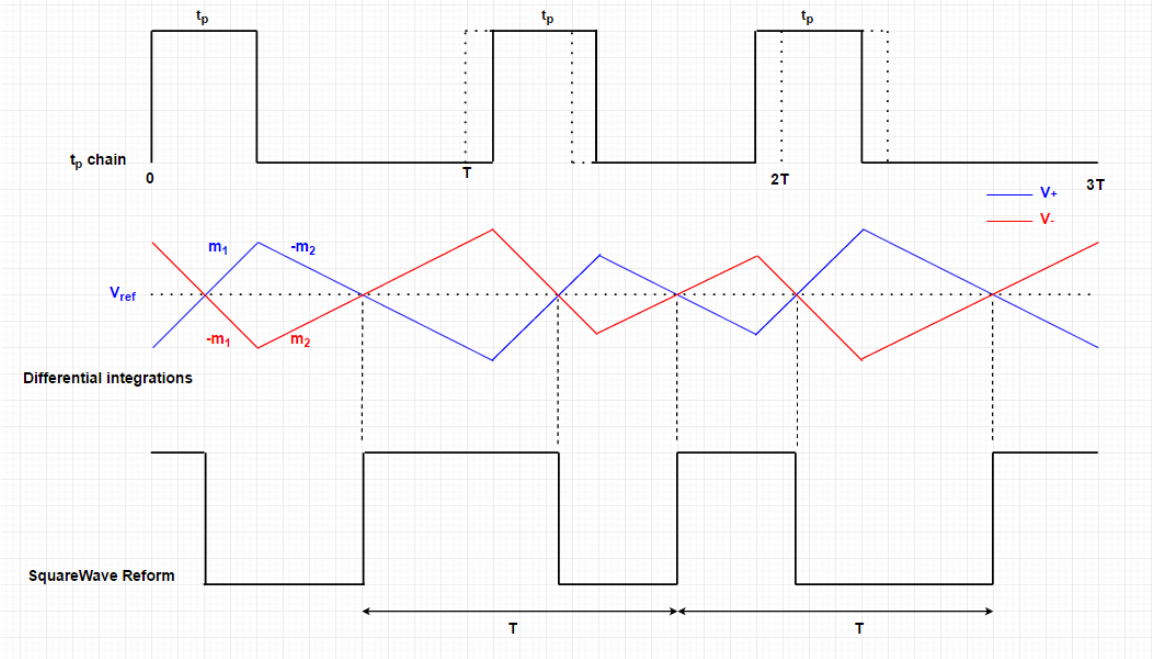

Figure 2.9 Square-wave reform block ideal output

2.3 System overall specifications

The targeted input jittery clock signal needs to be cleaned up by this JRC design is a 1GHz square-wave clocking signal, with maximum 200ps peak-to-peak Gaussian random jitter showing up on the rising edges. This means the rising edge could randomly arrive either earlier or later than its expected time within a variation range of ±100ps. To quantify the input jitter in a more intuitive way, the input jittery clock has an expected average clock period of 1ns, the period jitter has a μ_{jitter} of 0ps and a maximum σ_{jitter_in} ($t_{jitter\_rms\_in}$) of 33ps.

Figure 2.10 shows the targeted jittery clock and the maximum jitter range.

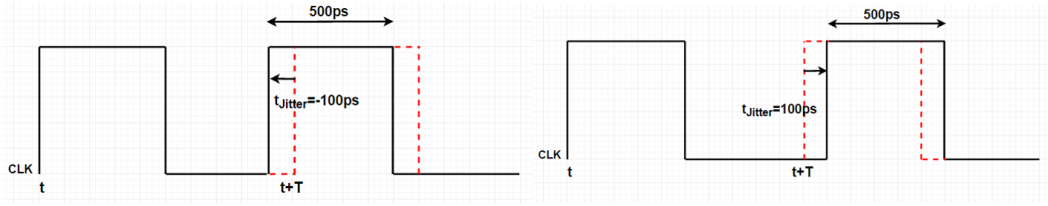

Figure 2.10 Targeted jittery clock with its maximum jitter range

Chapter 3

PULSE GENERATION BLOCK

The first block of the JRC system is the Pulse Generation block, which generates a complementary constant width pulse pair at each rising edge on the input clock. The width of the pulse has to be constant between each cycle in order to make the JRC system work properly though it will slide earlier or later depending on when the clock edge comes.

3.1 Constant pulse generation method and pulse width determination

As discussed in Chapter 2, t_p is the width of the pulse and it is generated at each rising edge of the incoming clock signal. Note that there hasn't been a detailed discussion of the actual pulse width, other than the fact that the pulse has to be constant width. In theory, the actual value of t_p does not matter as long as it is less than the period of the incoming clock signal T and, with jitter, does not extend before or after the period defined by the crossovers of the integrator. If these rules are followed, the equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$ will be true.

Section 3.1.1 will discuss the pulse generation circuit the case without jitter present. Section 3.1.2 discusses how t_p is constrained when jitter exists on the incoming clock.

3.1.1 Constant pulse generation method selection

The functionality of the pulse generation block is to generate a constant width pulse each time it sees a rising edge on the incoming clock signal. First let's consider the case without jitter presenting in the incoming clock signal. One possible way generate a pre-designed length of time is to introduce a well-defined time delay t_{pd} on the original signal and then put the delayed signal and an un-delayed signal into a Set-Reset latch as shown in Figure 3.1.

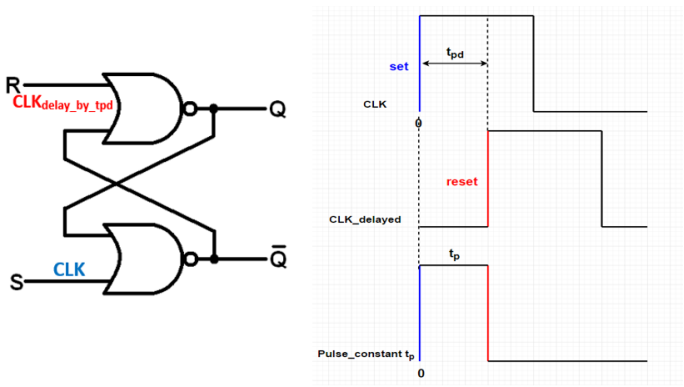

Figure 3.1 Constant pulse generation using NOR gate based SR latch

The original rising edge of CLK sets the latch output Q to high. Q stays high until the delayed clock signal $CLK\_{}_{delay}$ goes high after t_{pd}. When the delayed clock signal goes high, it resets the Q output to low therefore a pulse with width equals to t_{pd} is generated at output Q. As long as the delay t_{pd} is less than the pulse width of the jitter-less input CLK and extend the output waveform further with more clock cycles, a constant width pulse chain with the width t_p equal to t_{pd} is generated as shown in Figure 3.2.

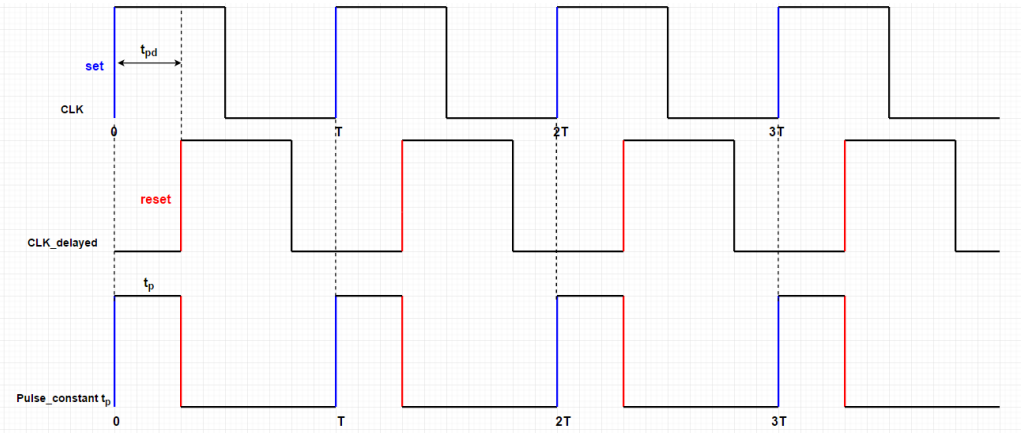

Figure 3.2 Constant pulse generation at each rising edge of the clock signal

So far the SR set-reset method to generate a constant width pulse chain at the Q output, associated with a jitter-less incoming clock CLK, works well as long as t_{pd} is less than CLK's pulse width. Yet recall the fact that the integrator block and the auto-biasing block need a complementary

copy of the Q output pulse chain. Table 3.1 is a typical truth table of a SR latch, and one of the state that might be problematic is the “Invalid” state both S and R with logic high input. In Figure 3.3, Q and Q’ are both reset to output low during the time when CLK and $CLK_{\_delay}$ are both at output high level, which represents the “Invalid” state. In this case Q and Q’ are not complementary and the overlapped logic low duration between the Q and Q’ outputs is considered as a pulse width error.

S	R	Q	Q’	Q State
0	0	Q-	Q’-	Latch
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Invalid

Table 3.1 SR Latch Truth Table

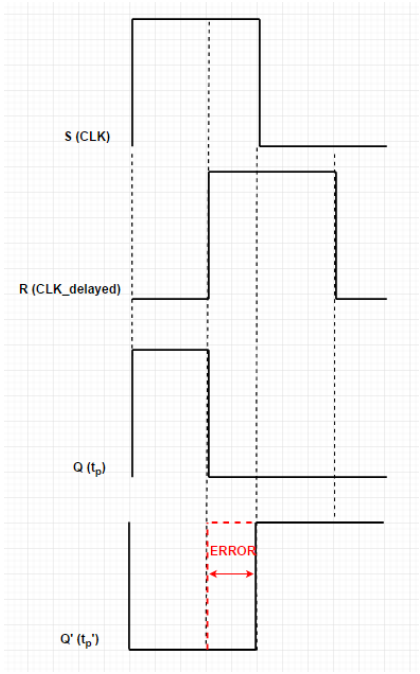

Figure 3.3 SR Latch outputs, using CLK as S input & CLK_delayed as R input

As an alternative, instead of watching the delayed clock signal, we could move to the inverted version of it, say, $\overline{CLK_{delayed}}$ as Figure 3.4 shows below. By maintaining the constraint that t_{pd} stay less than the pulse width of jitter-less CLK, taking logic AND operation between the CLK and $\overline{CLK_{delayed}}$, a constant width $t_p = t_{pd}$ pulse chain is achieved. Likewise, taking a logic NAND operation gives the exact complement of the t_p pulse chain. This AND-NAND method, due to its simplicity, is selected for the constant pulse chains generation.

To have the constant pulse chains generation stay valid with a jittery clock input, t_{pd} must be furtherly constrained by the maximum jitter amount contained by the clock. This leads to the t_p width determination which the next section will cover.

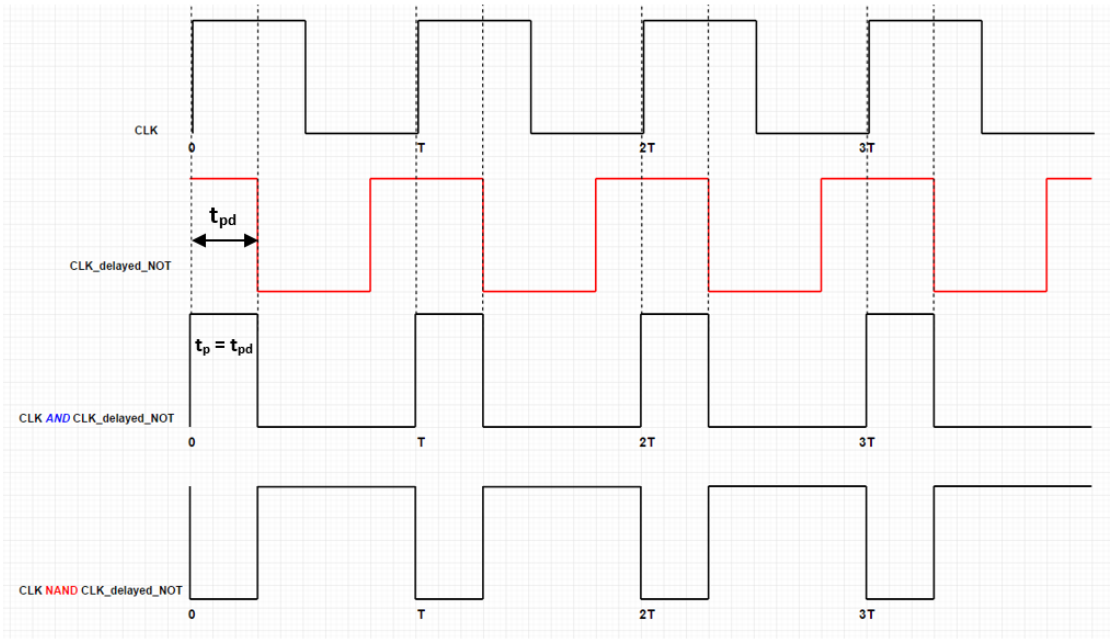

Figure 3.4 Constant pulse generation by the AND-NAND method

3.1.2 Constant pulse width t_p with jitter taken into account

Recall the requirement that each constant pulse width t_p must be bounced within one single period of T to make the equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$ stay true. Let's define the boundaries for t_p to be t and (t + T) as shown in Figure 3.5 which describes a case of jitter-less incoming clock. The highlighted portion of the CLK pulse represents where the constant width pulse t_p (generated by CLK and $\overline{CLK_{delayed}}$ using the AND-NAMD method) locates within the boundaries.

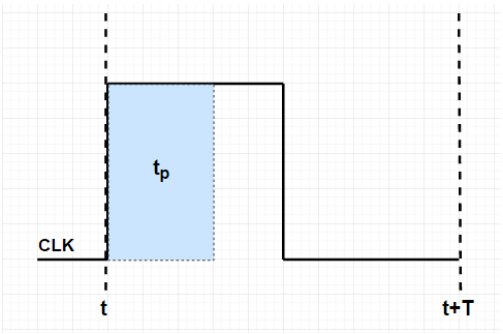

Figure 3.5 t_p pulse bounced by boundaries for a jitter-less incoming clock

Now jitter comes up to push the clock pulse behind by an amount of t_{jitter} as shown in Figure 3.6. As long as the full clock pulse width stays within the t and (t + T) boundaries, the same constant width pulse t_p is generated and represented by the highlighted portion which is guaranteed to be bounced by the boundaries too.

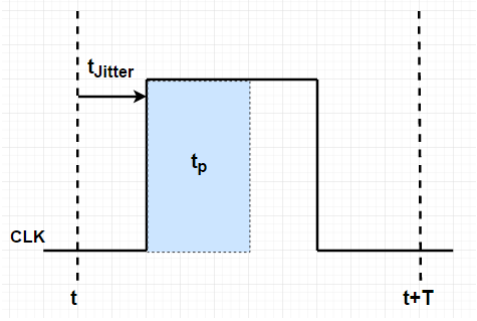

Figure 3.6 t_p pulse bounced by boundaries for a jittery incoming clock

Figure 3.7 shows an extreme case which has the highlighted portion reach the lower boundary of t. Even though part of the incoming clock pulse passes the boundary line, a width equals to t_p is still bounced by the t and (t + T) boundaries. So the same constant width t_p can still be generated by the CLK and $\overline{CLK_{delayed}}$ signals. This represents the case of maximum negative jitter, which equals to the difference between the CLK pulse width and the desired t_p. Once the negative jitter becomes larger than $t_{jitter\_negative\_max}$, the CLK pulse with the boundaries is less than the desired value of t_p and therefore a reduction of t_p within one period from t to (t + T).

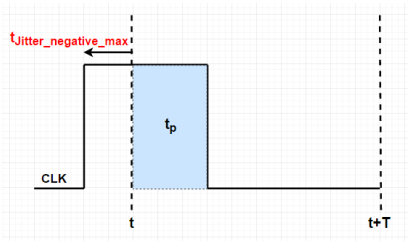

Figure 3.7 Maxmium allowable negative jitter to maintain constant t_p within one period

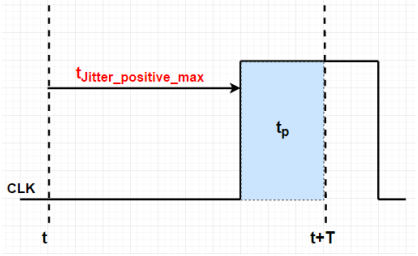

Figure 3.8 Maxmium allowable positive jitter to maintain constant t_p within one period

Similarly for the case with positive jitter as Figure 3.8 shows, the maximum positive jitter pushes a highlighted portion of the CLK pulse with a width of t_p, to reach the higher boundary line of (t + T). Any positive jitter larger than $t_{jitter\_positive\_max}$ will cause a t_p width reduction within the boundaries.

$t_{jitter\_negative\_max}$	$t_{p\_CLK} - t_p$
$t_{jitter\_positive\_max}$	$T - t_p$

Table 3.2 Maximum allowable jitter in each direction

Table 3.2 summarizes the maximum allowable clock jitter in each direction with which the AND-NAND pulse generation method can maintain a desired constant pulse width of t_p. Note that t_{p_CLK} means the pulse width of the incoming clock signal, and it can be different based on the duty cycle of the incoming clock signal.

For all simulations in this thesis, the nominal incoming clock signal is 1GHz with a 50% duty cycle therefore $t_{jitter\_negative\_max}$ can be calculated as $(\frac{T}{2} - t_p) = (500ps - t_p)$, and likewise, $t_{jitter\_positive\_max}$ is $(1000ps - t_p)$. Since the generated constant pulse width is designed to be equal to the delay amount t_{pd} by the AND-NAND method, t_{pd} value is directly related to the allowable jitter range too. Recall the constraint from section 3.1.1 that t_{pd} must be less than t_{p_CLK} in order to have $t_p = t_{pd}$, which can be rewritten as $t_{pd} < t_{p\_CLK} = \frac{T}{2} = 500ps$ based on the clock signal used in this thesis. Also recall that the targeted random Gaussian jitter distribution of ±100ps on the incoming clock, so the pulse generation must be capable for handling at least 100ps jitter on each side of the rising clock edges. This implies that $t_{jitter\_negative\_max} \geq 100ps$ since $t_{jitter\_positive\_max}$ is greater than $t_{jitter\_negative\_max}$. List the constraints as follow:

$$t_{pd} < t_{p\_CLK}$$

$$t_{p\_CLK} - t_{pd} \geq 100ps \rightarrow t_{pd} \leq t_{p\_CLK} - 100ps$$

Replace t_{p_CLK} with 500ps for the incoming clock signal used in this thesis, we get the constraint for t_{pd} to be

$$t_{pd} \leq 400ps$$

Now we have $t_{pd} = 400ps$ as the maximum delay amount to make the constant width pulse generation within the jittery environment. Reducing t_{pd} can accommodate larger jitter, however a longer t_p pulse is beneficial to the integrator stage against t_p variation and this will be included in Chapter 5. For this purpose, t_{pd} is selected to be 400ps and therefore t_p generated has a constant width of 400ps.

3.2 High-level block decomposition

As discussed in section 3.1, the pulse generation block is capable of generating a complementary constant width pulse chain pair to represent each rising edge of the incoming jittery clock. The pulse chain pair is then fed to the integrator stage as the switching controls. The pulse generation block in this JRC design takes differential clock signal as inputs, and then it generates the complementary pulse pair output: Pulse_P has constant output high width, and Pulse_N has constant output low width.

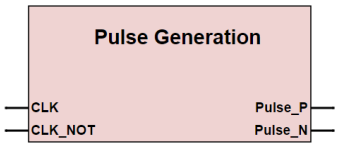

Figure 3.9 Level 1 Pulse Generation block diagram

Recall the functionality break down mentioned in section 3.1, the level 1 pulse generation block could be furtherly divided into two sub-blocks: a differential delay chain block and a differential NAND gate that performs logic NAND and logic AND operations at the same time.

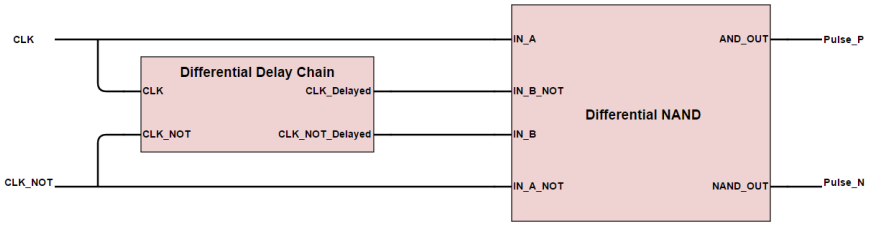

Figure 3.10 Level-2 Pulse Generation block diagram

Figure 3.10 shows the level-2 block diagram along with the sub-block connections and the I/O signals. The differential delay chain block delays the differential clock input signals and feed these delayed copies to the differential NAND gate. The differential NAND gate takes the delayed copies along with the original differential clock signals, and generates Pulse_P = $CLK \cdot \overline{CLK_{delayed}}$ and Pulse_N = $\overline{CLK \cdot \overline{CLK_{delayed}}}$.

3.3 Differential Delay Chain sub-block implementation

In section 3.1.2, we have determined the delay amount t_{pd} to be 400ps which means the differential delay chain introduces 400ps on the differential clock input through the full-chain propagation. Two separated even-numbered CMOS inverter chains is be used to generate the desired 400ps delay for each input of the complementary clock pair.

To determine the number of inverters used in the delay chain, we must first calculate the propagation delay of the individual inverter before stacking them up. In theory [25], to achieve equal rising and falling time therefore same high-to-low and low-to-high delay in a CMOS inverter, the size ratio $\frac{\left(\frac{W}{L}\right)_P}{\left(\frac{W}{L}\right)_N}$ needed to be equal to $\frac{\mu_N}{\mu_P}$ therefore the pull-up and pull-down paths have the same amount of effective resistance R and effective capacitance C at the output node. In the process being used in this thesis, μ_P=116.537cm^2/v/s and μ_N=266.035cm^2/v/s therefore the PMOS is μ_N/μ_P = 2.3 times size of the NMOS in the inverter. Figure 3.11 shows the CMOS inverter with a (1.2μm/180nm) PMOS and a (500nm/180nm) NMOS which matches the size ratio. Choose 0.9V as the voltage threshold which fully turns on/off a NMOS switch, and verify the rise-to-fall and fall-to-rise delays from the simulation results shown in Figure 3.12 and Figure 3.13, both are around 9.5ps.

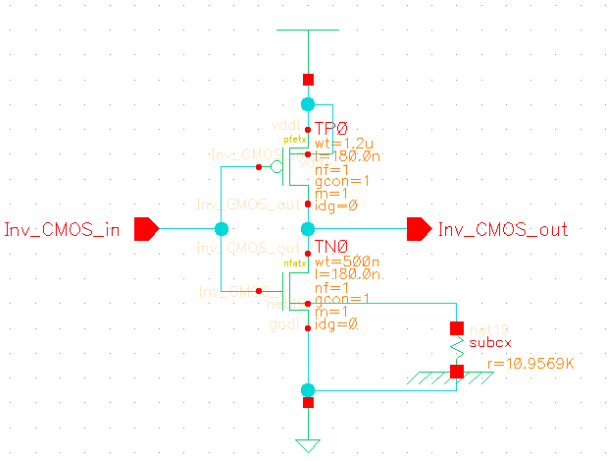

Figure 3.11 Single CMOS inverter

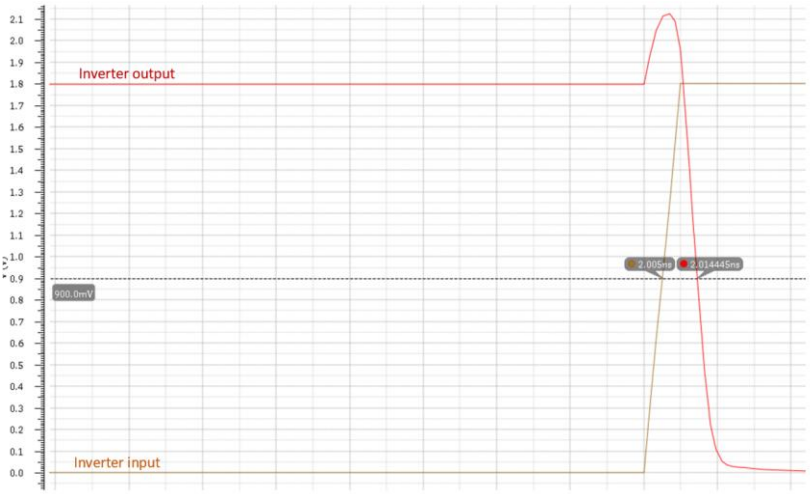

Figure 3.12 CMOS inverter rise-to-fall delay simulation

Figure 3.13 CMOS inverter fall-to-rise delay simulation

Two cascaded single inverters becomes a buffer, yet the buffered delay will not be equal to 2 times of the delay from a single inverter. Figure 3.14 shows the equivalent RC delay models for the single CMOS inverter and the CMOS inverter based buffer. The output Y node of the single inverter contains the drain capacitance C_{D_P} and C_{D_N}, but the same node in the buffer has extra gate capacitances C_{G_P} and C_{G_N} contributed by the second inverter. Due to the increased capacitance load at the output node, a longer delay from each cascaded CMOS inverter is expected.

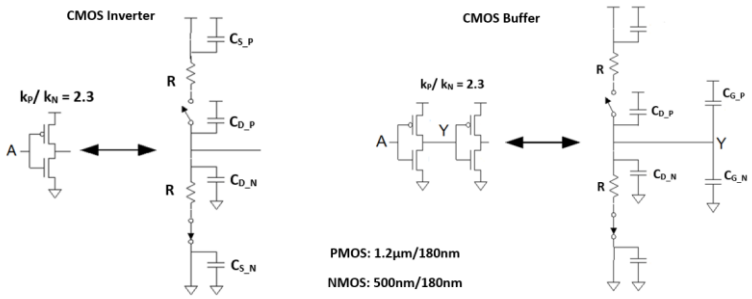

Figure 3.14 RC delay models of the CMOS inverter and buffer

In Figure 3.15, three single CMOS inverters shown in Figure 3.11 are cascaded and the first two inverters in this chain can be treated as a CMOS buffer. Simulation in Figure 3.16 shows the delay generated by the buffer is about 40ps, so ideally 10 of cascaded CMOS buffer can produce a 400ps time delay.

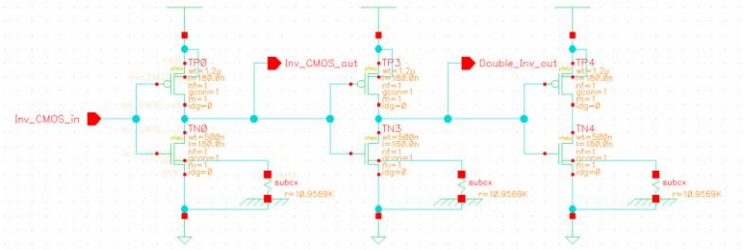

Figure 3.15 3-stage CMOS inverter chain

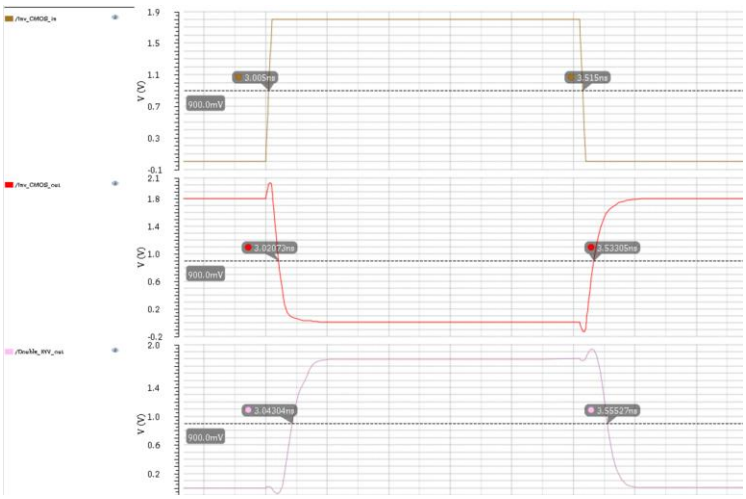

Figure 3.16 Time delay simulation for the CMOS buffer

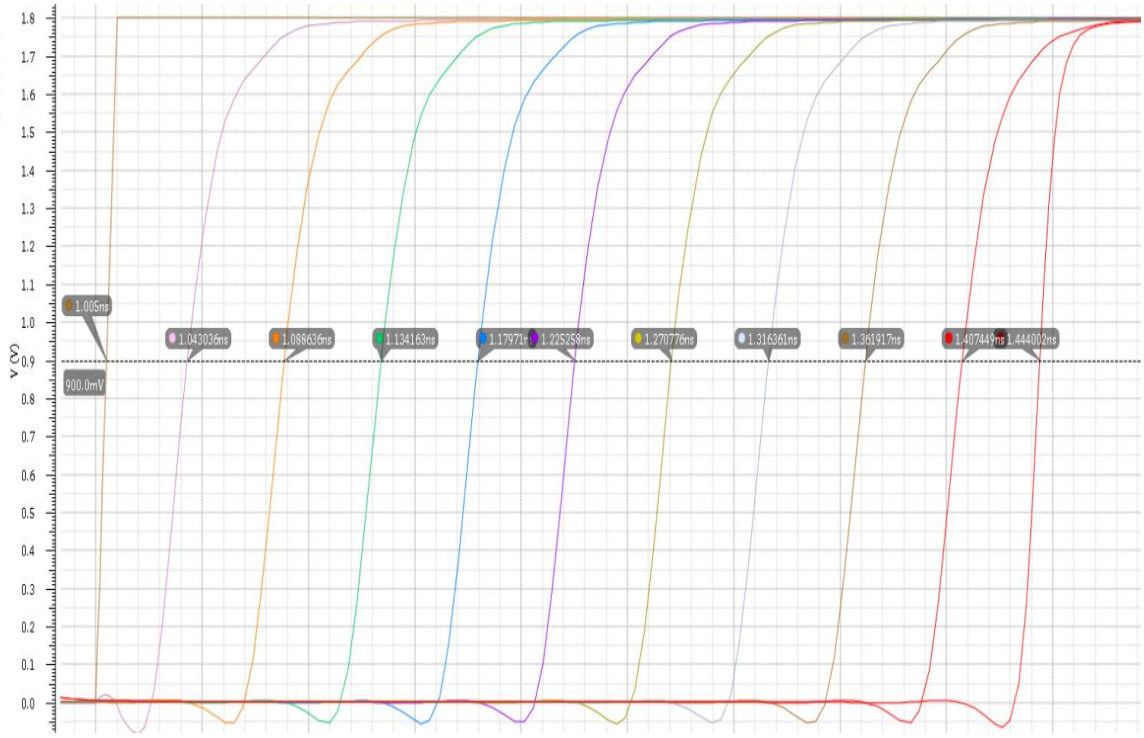

Figure 3.17 Time delay simulation for 10-stage double-inverter delay chain

The simulation in Figure 3.17 shows a total delay of 440ps is produced by 10 cascaded buffers instead of 400ps. The 1st buffer introduces 40ps delay as shown in Figure 3.16, but each of the 2nd through the 9th buffer is actually introducing 44ps delay. This is because the 1st buffer is driven by an ideal source during the simulation, which has infinite driving capability, so a shorter delay through the buffer. Otherwise, buffers except the 1st one are driven by another buffer within the chain which has limited driving capability and produces longer delay. So a delay chain with 9 buffers is enough to produce the intended 400ps delay. Simulation in Figure 3.18 shows a total delay of 396ps from the 9-buffer delay chain for the CLK_{delay} signal, while simulation in Figure 3.19 shows the $\overline{CLK_{delayed}}$ signal is also delayed by the same amount with another identical 9-buffer delay chain.

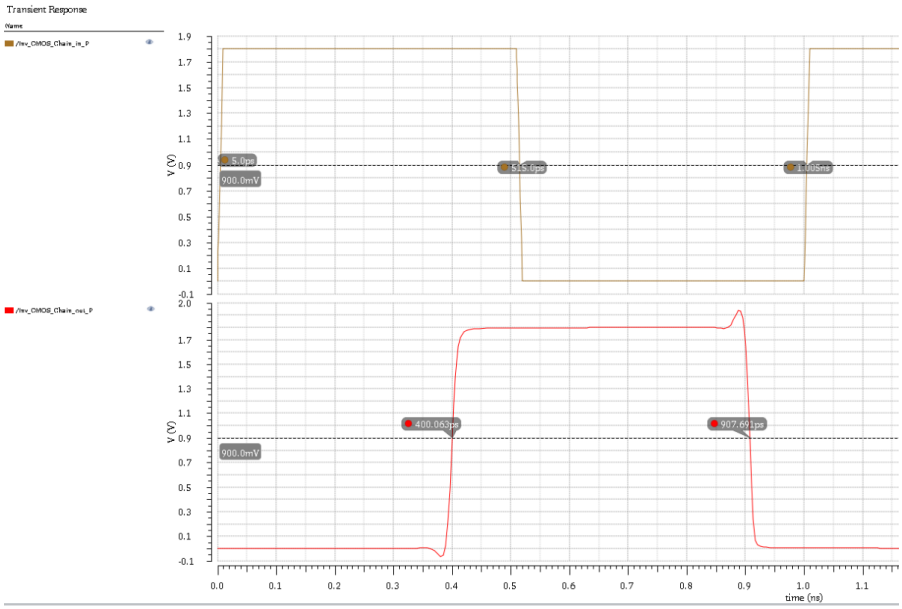

Figure 3.18 CLK_$_{delay}$ delay simulation with a 9-buffer delay chain

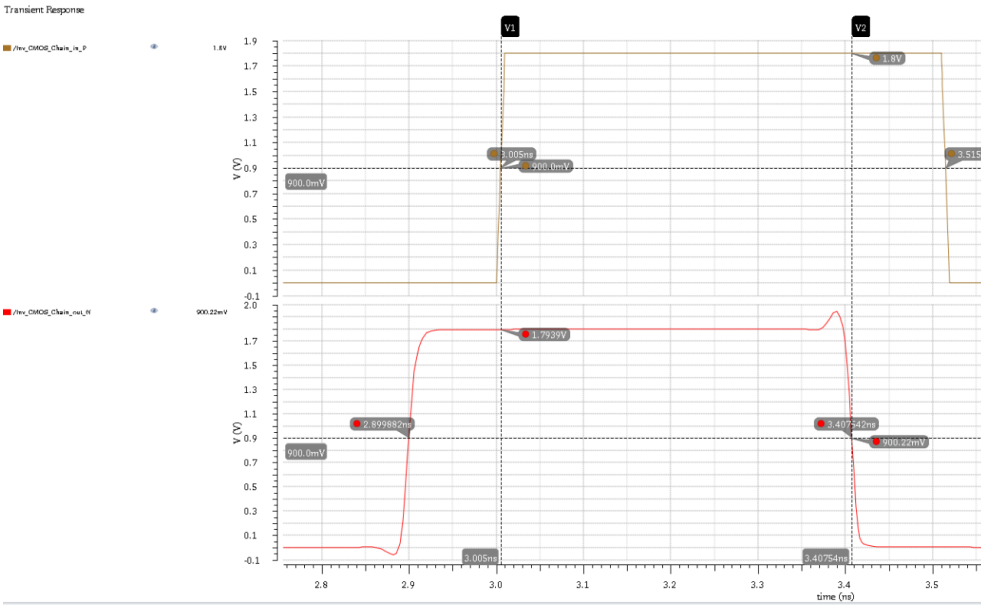

Figure 3.19 CLK_$_{delayed_NOT}$ delay simulation with a 9-buffer delay chain

3.4 Differential NAND gate sub-block implementation

The differential NAND gate takes CLK, $\overline{CLK}$, $CLK_{delayed}$ and $\overline{CLK_{delayed}}$ as inputs, performs logic AND and logic NAND operations, and outputs Pulse_P = $CLK \cdot \overline{CLK_{delayed}}$ and Pulse_N = $\overline{CLK \cdot \overline{CLK_{delayed}}}$. Figure 3.20 shows the two logical operations in parallel. If we implement this circuit in CMOS, it will consists of an AND gate and AND gates are made up of a NAND gate cascaded with an inverter. This introduces one inverter propagation delay difference between two outputs. This delay difference might not be significant in other applications but disrupts the generation of m_1 and m_2 in the JRC. An inverter may have a propagation delay of 10ps and would be present in both high-to-low and low-to-high transition. Since the pulse and the complementary pulse signal control the integrator's NMOS switches, any additional mismatch between the timing of the switching control signals brings in more switching error and that timing error affects the performance of the integrator.

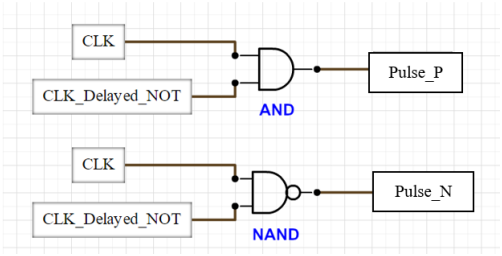

Figure 3.20 A functional break down of the differential NAND gate

Instead of generating the logic NAND and the logic AND outputs in different circuits, the differential NAND gate should carry out both AND and NAND and have differential outputs as well. A logic family that uses differential inputs and outputs is MOS Current Mode Logic (MCML) and the logic gated from this family is the NAND gate. The MCML family has other advantages such as much stronger noise immunity, no switching power consumption because it's a current steering logic family and better energy usage at high frequencies [11, 12]. The drawbacks of MCML is that it needs biasing circuitry which requires more hardware and more complicated

equivalent models. Also MCML logic outputs are not rail-to-rail and the reduced output swing may sometimes show a smaller noise margin. In this design, however, the MCML NAND outputs would be converted to rail to rail by adding CMOS inverters on the outputs. In addition the CMOS inverters at the end of output could sharpen the final pulse output signals, and provide better driving capability.

The MCML NAND gate is the universal MCML logic gate [10, 13] in Figure 3.21, by where CLK is connected to ~A, $\overline{CLK}$ to A, $\overline{CLK_{delayed}}$ is connected to B and $CLK_{delayed}$ is connected to ~B. MN5 in the universal gate improves the symmetry between both branches and the performance in high-speed applications. All NMOS in this universal gate have the same size, and if we eliminate MN5 but at the same time halve the width of MN2, the modified structure would have an equivalent total impedance to the universal gate. Figure 3.21 shows the modified MCML NAND gate with transistor sizing done. To calculate the PMOS load size and tail NMOS size along with the biasing circuity, a complicated analysis on MCML delay model in terms of the bias current, the voltage swing and process-dependent parameters, is needed [10, 13]. This goes beyond the scope of this thesis and would not be included here.

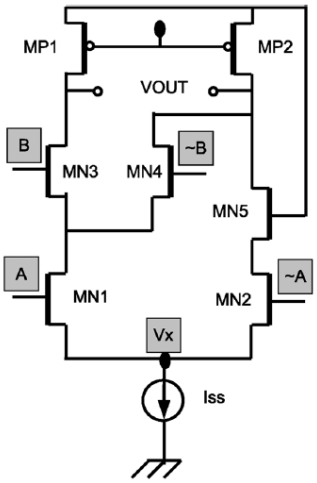

Figure 3.21 Universal MCML gate

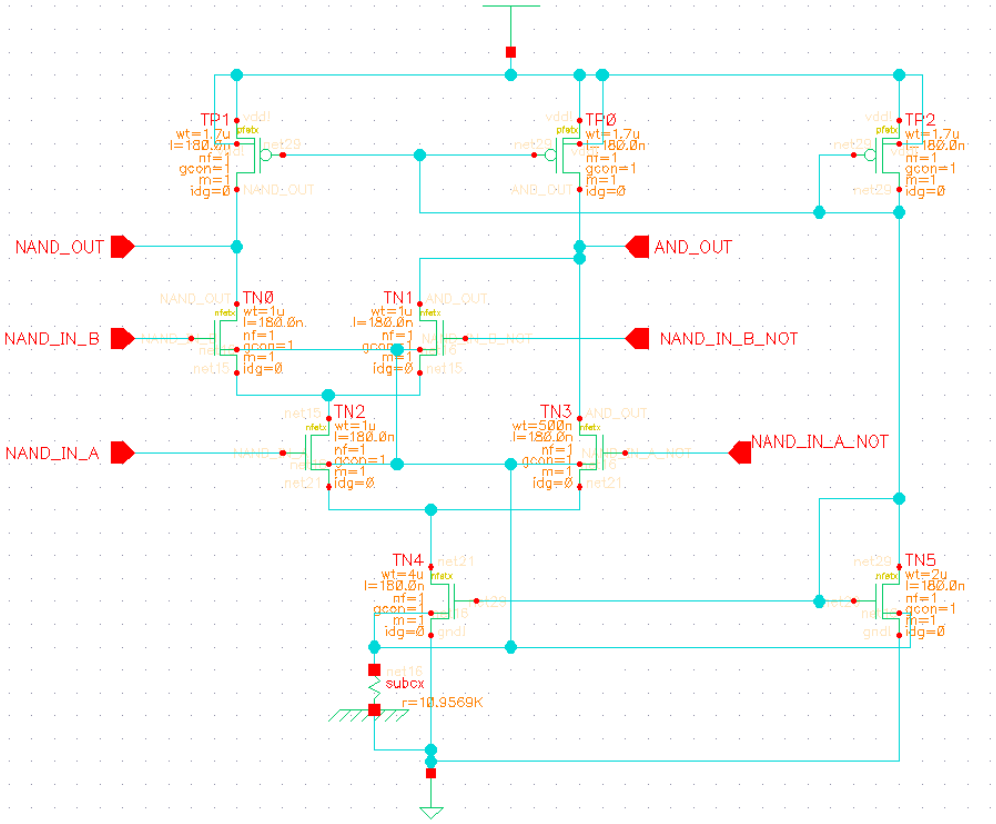

Figure 3.22 MCML NAND gate used in the pulse generation blcok, with sizing

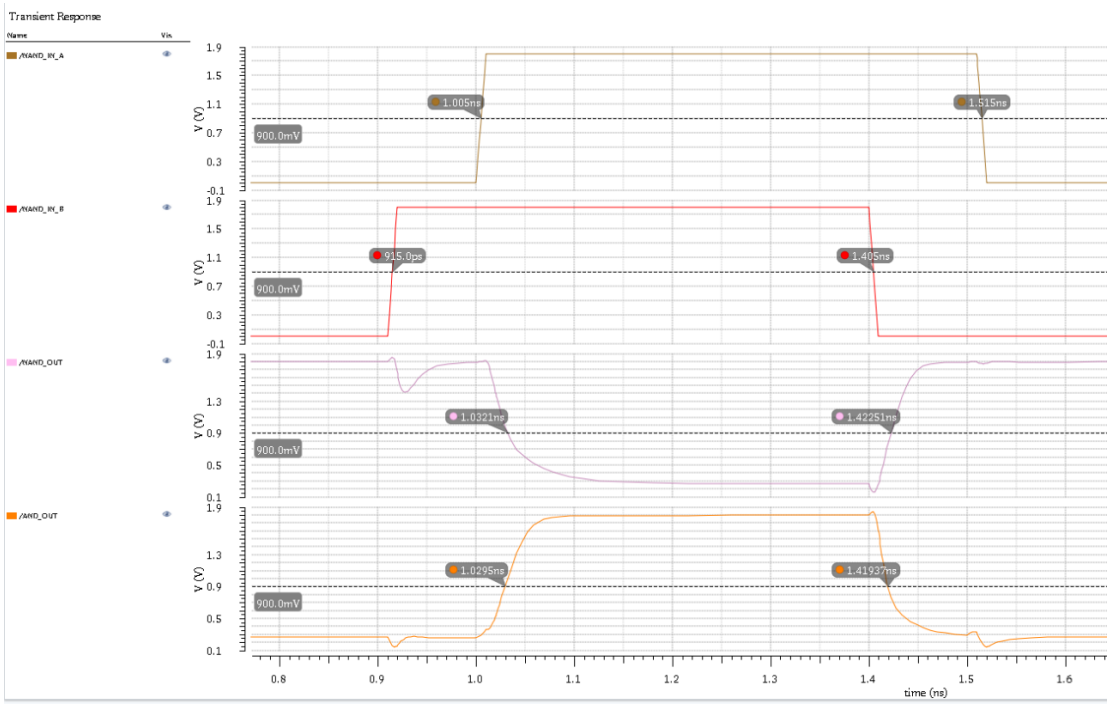

Figure 3.23 NAND simulation using ideal clock inputs

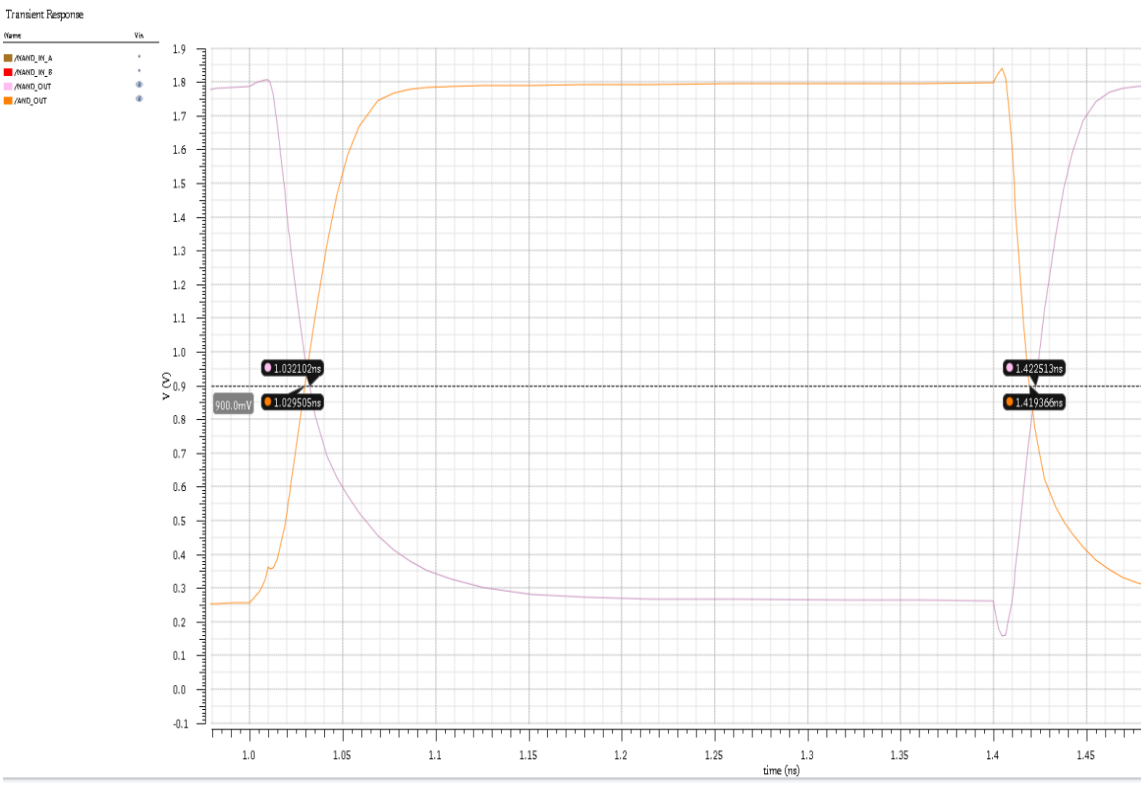

Figure 3.24 NAND gate output intecepts

NAND gate simulation results using ideal complementary clock and delayed clock signals as shown in Figure 3.22 and Figure 3.23. We see the AND output has a pulse width of 390ps as does the NAND. When putting these two output together as Figure 3.24 shows, we see 3ps of overlapped time with both pulses' output level higher than 0.9V on one transition and 3ps overlapped time with both pulses' output level lower than 0.9V on the other transition. In other words the NMOS switches driving by these two pulses would have 3ps with both of them turned on during one switch, and 3ps with both of them off during the opposite switch. Also note that the output logic low level does not go down to ground, and the rising and falling edges are not sharp enough as switching control signal requires. To achieve better driving capability, CMOS inverters are cascaded to the outputs as we will see in the next section.

3.5 Cascaded differential delay chain and differential NAND

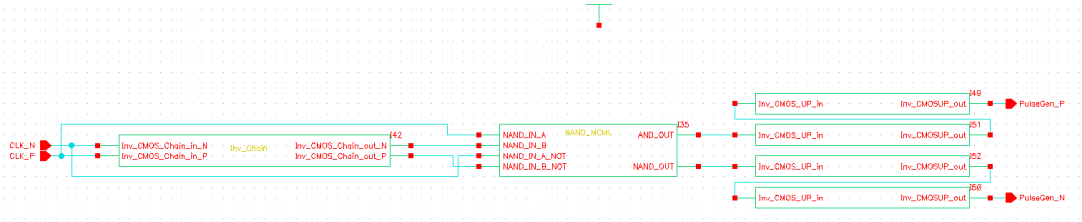

Figure 3.25 The sub-block-cascaded of the Pulse Generation block

To test the cascaded system, first of all an ideal complementary clock pair is fed to the delay chain and the final pulse pair is examined.

Figure 3.26 compares the complementary pulse pair by the MCML NAND gate and the enhanced version by the CMOS buffer, the pulse width is preserved but the rising and falling edges are sharpened. In Figure 3.27, a 396ps width pulse at Pulse_P output is generated for each rising edge of the incoming signal which is an ideal jitter-less 1GHz clock in this case, therefore the period of the constant width pulse chain is same as the input clock has. Similar conclusion applies to the Pulse_N chain.

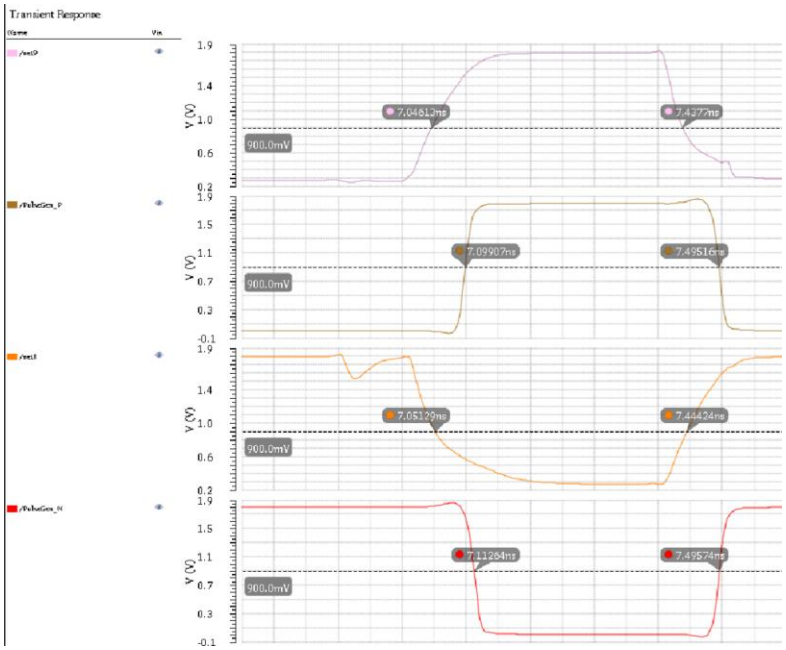

Figure 3.26 MCML NAND output vs CMOS inverter-enhanced outputs

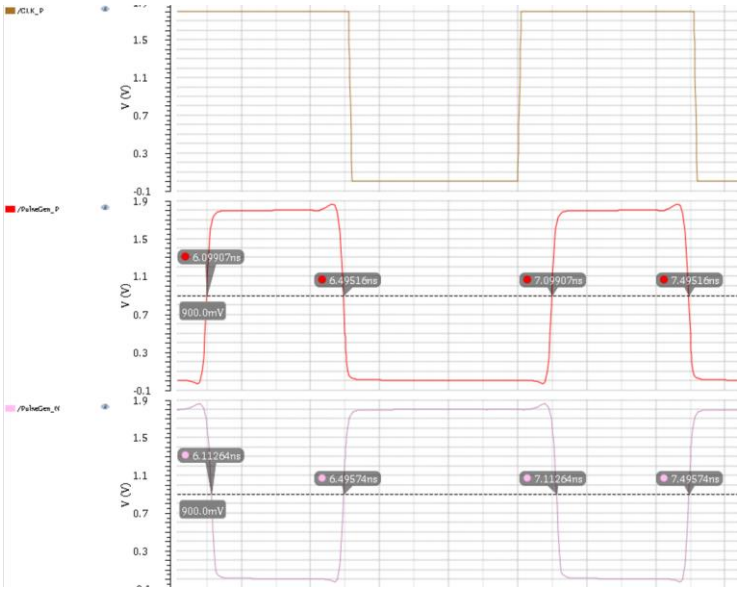

Figure 3.27 Pulse generation outputs with respect to an ideal input clock

Putting Pulse_P and Pulse_N on top of each other as Figure 3.28 shows, we see an overlapped time of about 12ps with both pulses have voltage level higher than 0.9V (both switches on integrator conduct) during one of the transitions but nearly none for the other transition. This could be caused by the output buffer itself or an error transmits and combines from each stage of the cascaded system. This overlapped turn-on time would contribute error to the integrator stage, which will be examined in the Chapter 5.

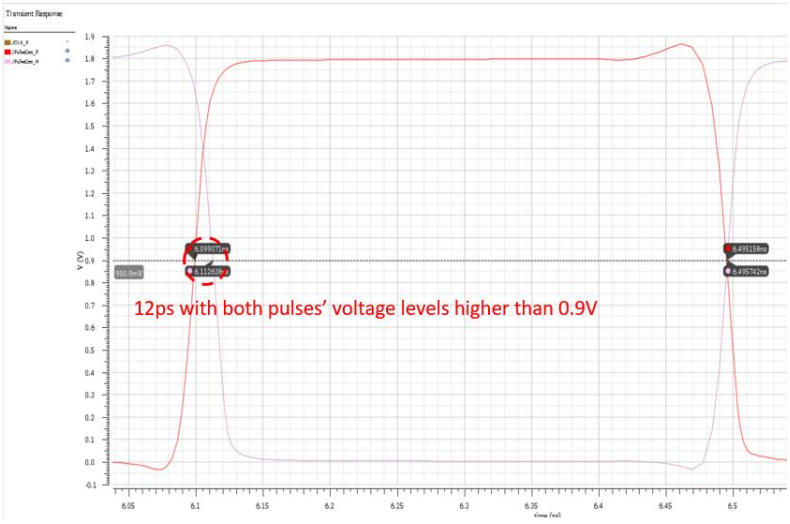

Figure 3.28 Pulse_P & Pulse_N overlap measurement

To verify the performance with jitter present, a Piece-Wise-Linear (PWL) file is generated with a Gaussian random noise (μ_{jitter}=0, σ_{jitter}=33ps) added to the rising edges of a 1GHz clock signal for 150 cycles. With a σ_{jitter} of 33ps, 99.7% of the random jitter falls into the ±99ps range therefore the pulse generation block should be capable to produce a constant 396ps pulse for each jittery rising edge it encounters. Figure 3.29 shows that Pulse_P has same jitter amount as the jittery clock has, but it generates a constant pulse width of 396ps for each jittery clock rising edge. Table 3.3 lists the width measurement result (time duration with pulse voltage level beyond 0.9V) on the generated pulses with the jittery incoming clock signal. 150 out of 150 of the generated pulses have a width of 396ps as intended.

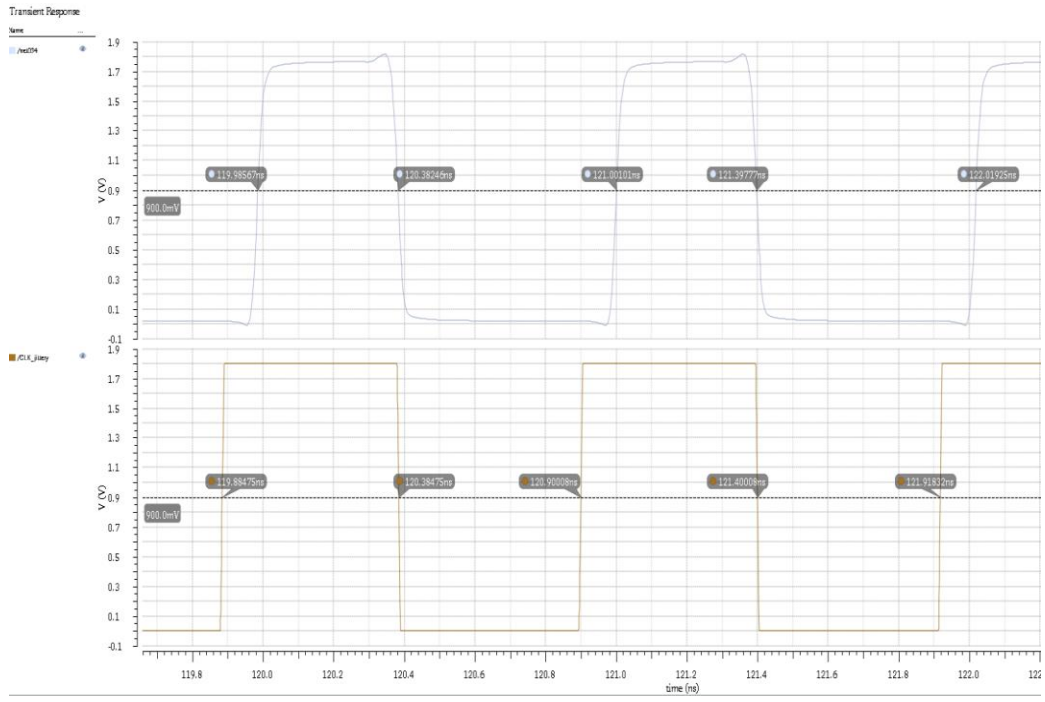

Figure 3.29 Constant pulse generation with jittery input clock at 1GHz

	Total # of pulses	**t_p = 396ps**	**t_p>396ps**	**t_p<396ps**
# of measurements	150	150	0	0
% of total	100%	100%	0%	0%

Table 3.3 Pulse generation test results under jittery input clock

Chapter 4

FEEDFORWARD AUTO-BIASING BLOCK

In the Chapter 3, the pulse generation block was described. Its job is to generate the constant pulse width t_p which is critical in the fundamental equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$. Now we move to the feedforward biasing block which is capable of adjusting m_1 and m_2 automatically in order to make the equation stay true.

There are two important design requirements for the integrator: that the current supply PMOS which charges the integrator's capacitance stays in the saturation mode so it will behave as a constant current source; and that m_1 and m_2 (which are created by the charging and discharging of an integrator capacitor) must be values which make the fundamental equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$ true. As we have seen in Chapter 2, the integrator's tail current sink has its sunk current equal to the sum of the two PMOS sourced currents. We also know that the charging (PMOS) current is exactly the sourced current from the PMOS because the switch disconnects the capacitor from the tail current sink. During the discharging time, the switch connects the charged capacitor to the tail current sink, since the tail current source is larger than the PMOS current, the capacitor is discharged. The net current off of the capacitor is the PMOS current minus the tail current source. Since the tail current source is equal to the sum of the two PMOS currents, when one side's PMOS current is subtracted, the resulting current is the current through the opposite PMOS.

Restating the relationship between the constant PMOS sourced currents and the constant pulse width t_p we have concluded in Chapter 2, as follow:

$$\frac{I_{charge}}{I_{sink}} = \frac{T - t_p}{T}$$

$$\frac{I_{discharge}}{I_{sink}} = \frac{t_p}{T}$$

$$\frac{I_{charge}}{I_{discharge}} = \frac{T - t_p}{t_p}$$

I_{charge} and $I_{discharge}$ therefore is proportional to the negative and the positive duty cycle of the pulse chain. Since the ratios of the duty cycles of the pulse and the inverted pulse give the current ratios, it is important to have appropriate bias voltages to the PMOS current sources and the NMOS tail current sink. This will be achieved by the feedforward biasing block introduced in this chapter.

4.1 Feedforward auto-biasing block decomposition

In general, this feedforward biasing block uses information from the differential constant width pulse chains to keep the two PMOS in the integrator in saturation operating mode and therefore function as constant current sources. The sourced currents must represent the duty cycles of the differential pulse chain. In addition, the biasing block needs to bias the integrator's tail NMOS so it conducts a constant current equal to the sum of the PMOS sourced currents. To get biasing values for currents that have a ratio equal to the duty cycles of the constant width pulses in the pulse train, this feedforward biasing block is divided into three major sub-blocks: a duty cycle to voltage conversion block, a voltage to current conversion block used to generate the PMOS biasing voltages, and a current summing block to sum the PMOS currents and generate the biasing voltage for the NMOS tail current sink.

To convert pulse duty cycles into voltage, a low-pass filter is used to calculate the average DC value of the square-wave pulse chain from the pulse generation block. The lowest frequency component of a square wave is the square wave's DC average and a LPF can extract that DC value. The lower the corner frequency of the filter, the closer to a DC value will be output by the filter but note that the lower the pole, the longer it takes for the filter to find the DC value. Since the DC values are being used to bias the integrator, the core component of the system, the system won't work until the LPF has arrived at its final value so, when using this circuit, the tradeoff between having the system reject more jitter or be able to adjust to a new frequency quickly needs to be made. Also note that the duty cycle is directly proportional the DC average. A longer duty cycle

will translate to a higher DC value and a shorter duty cycle will translate to a lower value. Once the DC average is generated, it is fed to a linear voltage-to-current conversion block to generate a current. That current is the translated to a voltage by putting it through a diode connect PMOS. The gate voltages of the diode connected PMOSs are then outputted as the bias control of integrator's PMOSs. Finally a circuit sums up the currents from the PMOS sources to an NMOS current sink and then takes the gate voltage for the summed current as the bias control output for the integrator's NMOS tail current sink. Figure 4.1 shows the high-level functional block decomposition of the feedforward biasing block.

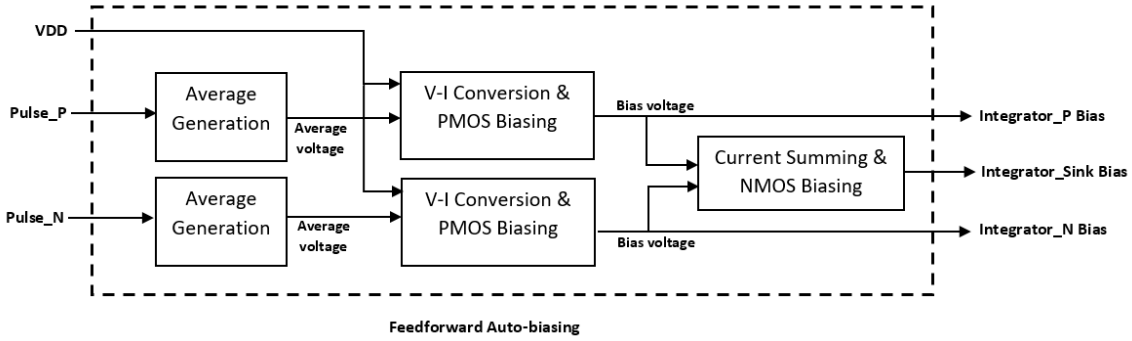

Figure 4.1 Feedforward auto-biasing block decomposition

4.2 Feedforward auto-biasing circuitry sub-block implementation

4.2.1 Averager

A traditional passive RC low-pass filter network is used as an averaging device, and converts the input square-wave pulse chain into its DC average voltage which represents the square wave duty cycle. Figure 4.2.1.1 shows the RC low-pass averaging network. There are some tradeoffs between the averaging voltage output settling time and the output ripple. To achieve a faster start-up time, a low-pass filter with a higher corner frequency c is needed. This produces a DC value more quickly than a lower corner frequency filter would however a larger ripple would present at the average output. On the other hand a more constant DC average output requires a lower cut-off frequency but that causes a longer responding time. Since the point of the auto-biasing block is to provide accurate biasing information, better average output ripple suppression is more likely to be preferred. The low-pass network in Figure 4.2 has a -3dB low-passing cut-off frequency of about 17MHz which provides an output ripple of about 0.5mV$_{pp}$ with a start-up time of about 50nS.

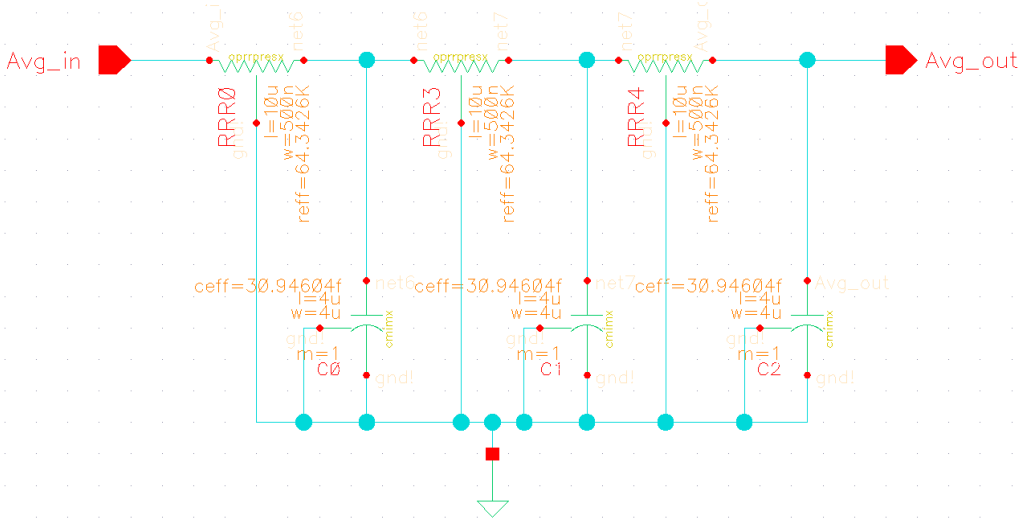

Figure 4.2 Cascaded RC low-passing network for DC average generation

As mentioned in Chapter 3, the pulse generation block generates a differential pulse chain: in the cases without jitter at 1GHz, the pulse generation block provides a pulse width of 396ps and the complement signal has a pulse width of 617ps measured from the 50% mark of the rise and fall times. When the two pulse trains with the widths listed above are fed to the averaging network the accuracy of the averaging circuit can be evaluated.

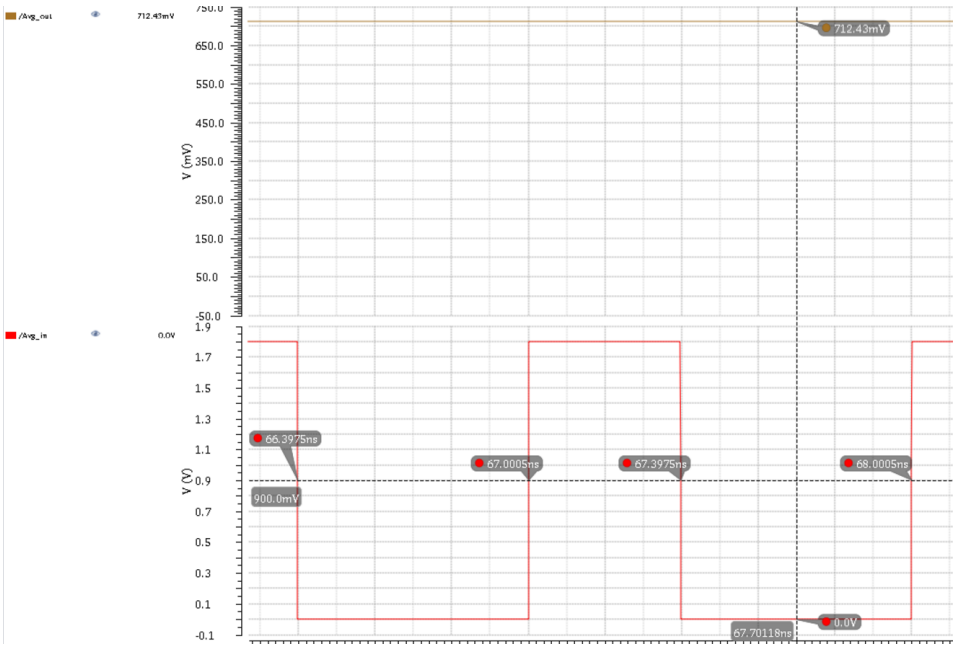

Figure 4.3 Averaging network output with 39.6% duty cycle pulsing

Pulse width	Duty cycle	Ideal average output (1.8V supply)	Actual average output	Actual Duty	Output ripple$_{pk\text{-}to\text{-}pk}$	Start-up time
396ps	39.6%	0.7128V	0.7146V	39.7%	0.5mV	50ns
617ps	61.7%	1.1106V	1.1121V	61.8%	0.5mV	50ns

Table 4.1 Averager performance with pre-designed duties by the pulse generation blcok

Based on the result listed in Table 4.1, the output from the averaging network is about 0.2% higher than the expected value and contains an output ripple of 0.5mV. Start-up time is measured as the time the output takes the output of the filter to rise from ground to its final value, and 50ns means 50 clock cycles (at 1GHz) are needed for the averaging network to have its outputs become valid.

4.2.2 Current source PMOS biasing

The DC average voltage representing duty cycle of the pulse chain is fed to the second stage of the auto-biasing block, which is a linear voltage-to-current converter. In this block the voltage is converted to current and then that current is pushed through a diode connected PMOS to get the correct biasing voltage for that current. Because the current is proportional to the duty cycle, the current is too. Then the PMOS gate terminal voltage is used as the bias current to control the PMOS current source on the integrator.

A gain-boosted linear transconductor topology [15] is used to do the linear voltage-to-current conversion and is shown in Figure 4.4. M_{P1} and M_{N1}, M_{P2} and M_{N2} are configured as high gain amplifiers. M_{P3} and M_{N3} forms a current feedback loop to the node V_D. All of these transistors are in the saturation operating mode, except the tail NMOS M_{SINK} which will be in deep linear operating mode. The circuitry above M_{SINK} is able to maintain a constant voltage at node V_D, regardless of the input voltage at V_{IN}. If V_D is set to be under the saturation voltage V_{D_SAT} of M_{SINK}, then it will operate in the linear mode and behave as a resistor and resistors have a linear $V_{IN} - I_{SINK}$ relationship.

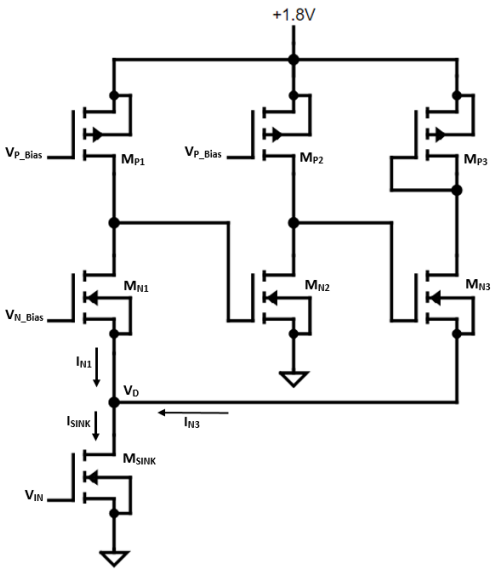

Figure 4.4 Gain-boosted linear transconductor topology

To find out the relationship between V_{IN} and V_D, we can apply Kirchhoff's Current Law at node V_D to get $I_{SINK} = I_{N1} + I_{N3}$. Next apply the fact that only M_{SINK} is in linear mode and all other transistors are in saturation mode and consider the gate voltage of M_{N3} as a function of V_D, the current equation at node V_D could be replaced by the current expression in each mode as follow:

$$k_N \frac{W_{sink}}{L_{sink}} V_D \left(V_{IN} - V_{TN} - \frac{V_D}{2}\right) = \frac{k_N}{2}\frac{W_1}{L_1}(V_{Bias1} - V_D - V_{TN})^2 + \frac{k_N}{2}\frac{W_3}{L_3}\left(V_{G3f(V_D)} - V_D - V_{TN}\right)^2 \quad \textbf{(Eqn 4.2.2.1)}$$

Solving for V_{IN} from Eqn 4.2.1.1 and then taking the derivative with respect to V_D along with using the fact that the M_{N1} and M_{N2} amplifiers have high gain, it is shown in [1] that:

$$\frac{\partial V_{IN}}{\partial V_D} \approx \frac{2}{V_D}(V_{G3f(V_D)} - V_D - V_{TN})(g_{m\,M_{N1}} \cdot r_{O\,M_{N1}} \cdot g_{m\,M_{N2}} \cdot r_{O\,M_{N2}} - 1)$$

Replacing $(V_{G3f(V_D)} - V_D - V_{TN})$ with $V_{D\_SAT\_N3}$, and replacing V_D with $V_{Bias1} - V_{D\_SAT\_N1}$ to represent the minimum value of $\frac{\partial V_{IN}}{\partial V_D}$, and then flipping the expression above gives you the ratio $\frac{\partial V_D}{\partial V_{IN}}$ we are interested in:

$$\frac{\partial V_D}{\partial V_{IN}} \approx \frac{1}{\frac{2}{V_{Bias1} - V_{D\_SAT\_N1}} \cdot V_{D\_SAT\_N3} \cdot (g_{m\,M_{N1}} \cdot r_{O\,M_{N1}} \cdot g_{m\,M_{N2}} \cdot r_{O\,M_{N2}} - 1)}$$

We can see that if the product of gain through M_{N1} and M_{N2} is large enough, V_D does not change even when V_{IN} varies and it could be treated as a constant. As a result, the only variable left in the current equation of M_{SINK} is V_{IN} and the sunk current therefore becomes purely in proportional to V_{IN}.

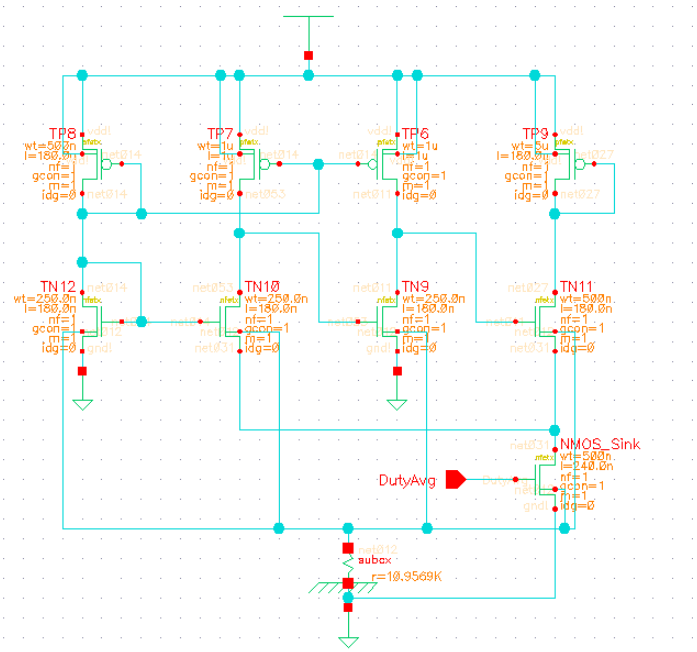

Figure 4.5 Linear transconductor implementation

Figure 4.5 shows the schematic of the linear transconductor. The DC voltage input is swept from 0 to 1.8V, represents a duty cycle change from 0% to 100%. The drain voltage and current of the sink NMOS are shown during a linear input voltage sweep, as shown in Figure 4.6.

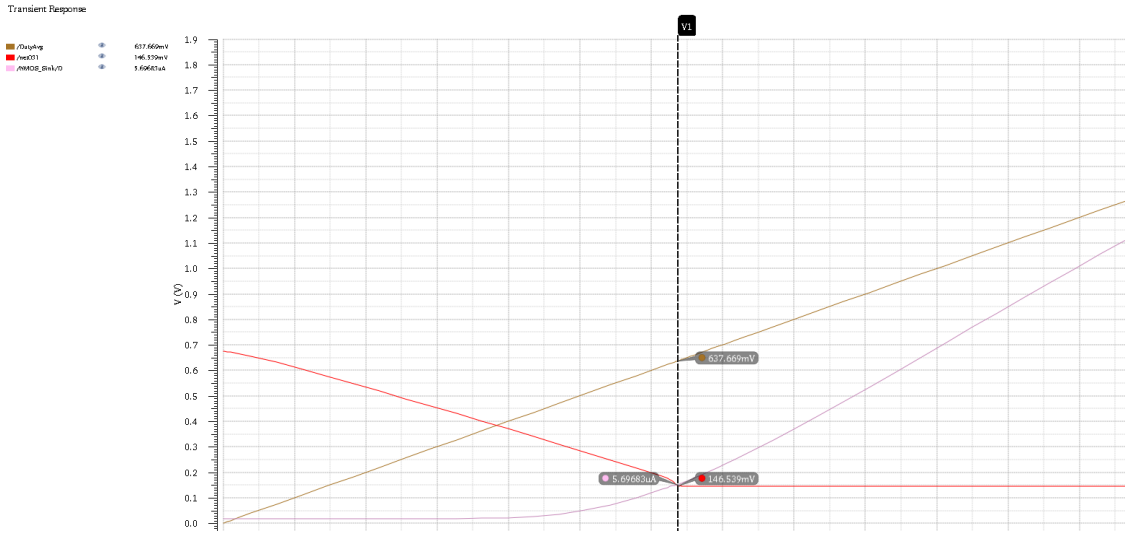

Figure 4.6 Input sweeping simulation of the linear transconductor

Within the region where $V_{IN} < V_{TN}$ (0.3V from the model file) the sink NMOS is in cut-off mode therefore no current is conducted, but we note the drain voltage V_D drops. After going through the saturation operation region where $0 < V_{IN} - V_{TN} < V_D$, the sink NMOS is eventually driven hard enough to have the overdrive voltage raise beyond V_D to enter its linear operation mode. We see that as soon as V_{IN} goes up above 0.638V, V_D is fixed at around 0.146V which is significantly lower than $V_{IN} - V_{TN}$. Therefore we can conclude that the sink NMOS stays in linear mode for V_{IN} over 0.637V and a linear relationship between V_{IN} and I_{SINK} can be achieved. The simulation in Figure 4.6 shows this ramping drain current but note that the ramp looks consistent in a limited range only. Recall the expected DC voltages provided by the averaging network we have mentioned in the section 4.2.1, are about 0.7V and 1.1V with pulse width set by the pulse generator, 396ps and 617ps. The linearity from 0.7V to 1.2V input is examined in Figure 4.7. Input voltage is sweeping at a speed of 1V/s, from 0 to 1.8V.

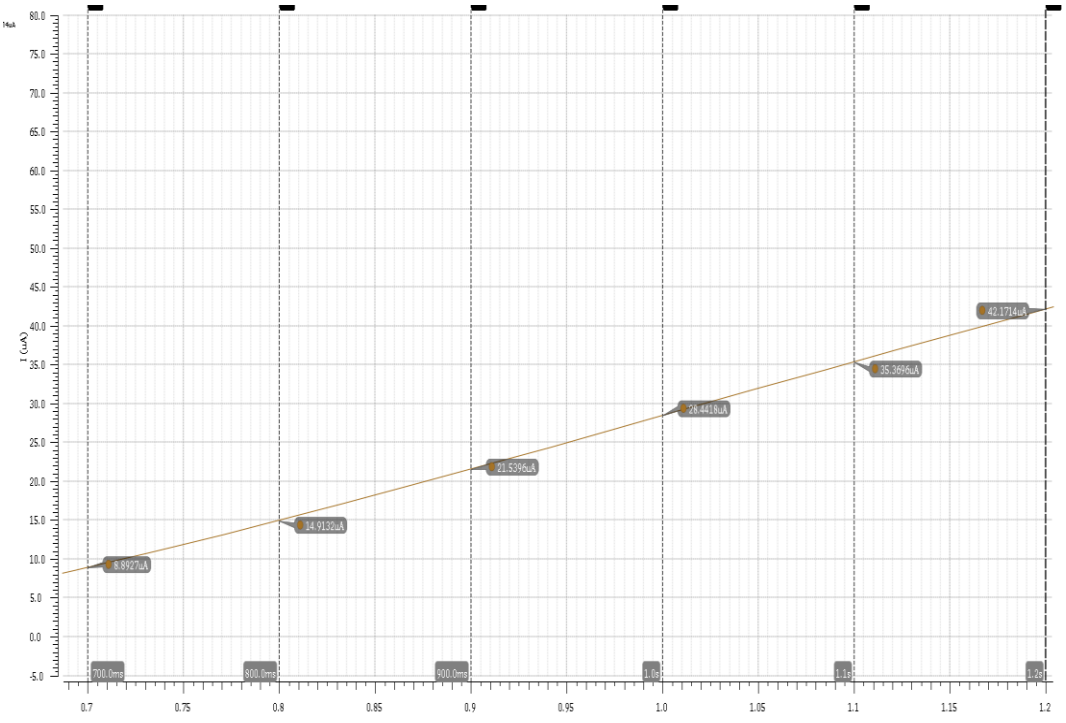

Figure 4.7 I_{sink} linearity between 0.7V and 1.2V input

Range	Current change (µA)	Slope (µA/100mV)
700mV – 800mV	6.02	6.02
800mV – 900mV	6.62	6.62
900mV – 1000mV	6.90	6.90
1000mV – 1100mV	6.92	6.92
1100mV – 1200mV	6.80	6.80
700mV – 1200mV	33.28	6.65

Table 4.2 I_{sink} linearity results within the 0.7V to 1.2V input range

Table 4.2 shows the sink current linearity results of the transconductor within the input range from 0.7V to 1.2V. The input segment from 700mV to 800mV gives the lowest slope which is only 90.5% of the endpoints slope because of the curvy I-V relationship around 700mV input. This means the transconductor is not turned on hard enough to enter its deep linear operation region yet. If it was in deep linear, as the input voltage increases, the sink current would become more linear with a less varying slope.

Convert the sweeping input voltage back to pulse duty cycle (duty = V_{in} / VDD), we get the current-duty relationship of the transconductor within the same 0.7V to 1.2V input range, shown in Figure 4.8. The linear-fit slope of the current-duty curve is 1.2027 µA/1%$_{\_duty}$.

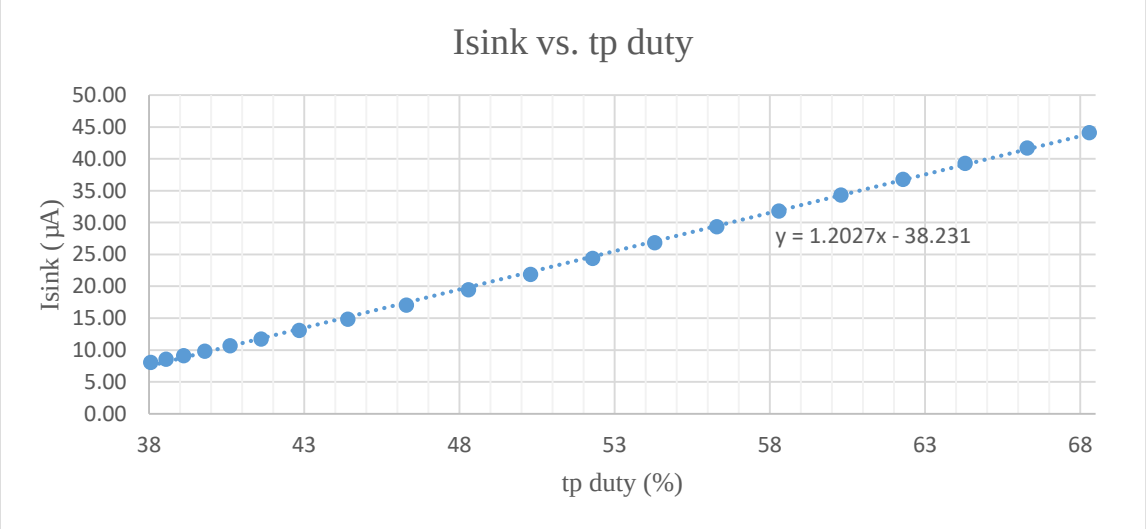

Figure 4.8 I_{SINK} vs t_p duty cycle

The NMOS current I_{SINK}, however, is not the current we are actually interested in since the transistor needed to be biased is the current source PMOS in the integrator block. In fact, the transistor would be used to provide the bias voltage to the integrator is M_{P3} in Figure 4.4, which is diode connected to guarantee that it is operating in the saturation region. M_{P3} conducts the same amount of current as M_{N3} conducts, which we will call I_{N3}. Figure 4.9 shows how I_{N1} and I_{N3} change as I_{SINK} ramps up. We see that I_{N3} is zero before V_D reaches its final fixed value and, therefore, I_{SINK} = I_{N1}. However as soon as V_D is above a voltage that allows MP3 and MN3 to turn on, I_{N1} due to the high gain of MP1/MN1 and MP2/MN2 amplifiers. They change the gate voltage of MN3 a large amount for a small change in voltage and therefore take the majority of the change in the current becomes constant. Now I_{N3} is just an offset from I_{SINK} due to the fixed I_{N1} therefore behaviors the same way I_{SINK} does.

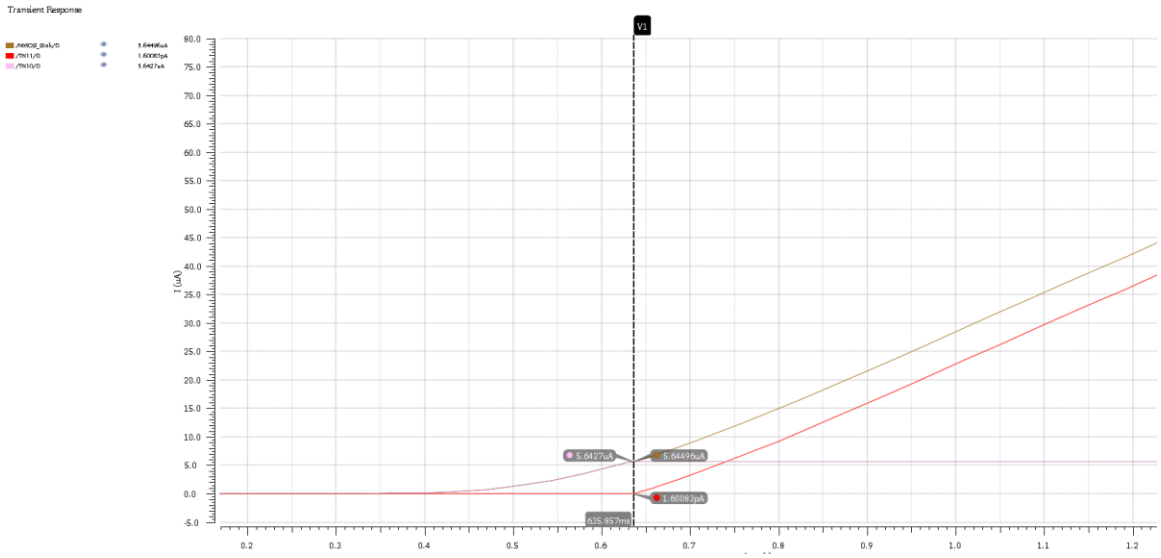

Figure 4.9 I_{sink}, I_{N1} and I_{N3} relationship during input sweep

Though the PMOS conducts current linearly with respect to input voltage in a certain range, one thing to be careful of is the actual relationship between the conducted current and the duty cycle: the charging current is proportional to the discharging duty cycle, and the discharging current

is in proportional to the charging duty cycle. This makes the equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$ stay true. In other words, the charging current conducted within t_p is in proportional to $(T - t_p)$ while the discharging current conducted within $(T - t_p)$ is in proportional to t_p. To achieve such a relationship between current and duty cycle, we could consider adding a subtraction mechanism to the transconductor: $I_{charge} = (I_T - I_{t_P}) \propto (T - t_p)$ and $I_{discharge} = (I_T - I_{(T-t_P)}) \propto T - (T - t_p) = t_p$. I_T represents the full-duty current, which is generated by feeding VDD to the transconductor because a 100% duty signal could be treated as a constant DC at its positive rail.

To perform the current subtraction, an extra branch which mirrors $I_{t_P}{}'$ (I_{N3} in Figure 4.4, which is $I_{Sink} - I_{N1}$) is attached to the transconductor that conducts I_T at node V_D as Figure 4.10 shows. Applying Kirchhoff's Current Law again at node V_D, we get $I_{N3} = (I_T - I_{N1\_T}) - I_{tp}' = (I_T - I_{N1\_T}) - (I_{tp} - I_{N1\_tp})$. If $I_{N1\_T} = I_{N1\_tp}$ then I_{N3} is the desired current $I_{charge} = I_T - I_{tp}$ we are looking for. Figure 4.11 and Figure 4.12 show the simulation results of the combined two linear transconductors whose full schematic is shown in Figure 4.13. The simulation shows them producing the charging and discharging current we are targeting.

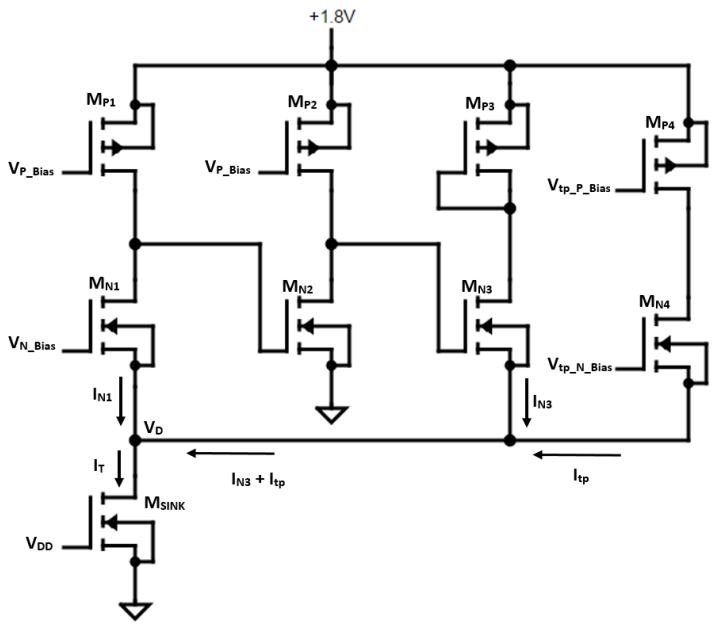

Figure 4.10 Transconductor configuration with current subtraction branch attached

Figure 4.11 shows a simulation of the constant voltage node V_D and the current through M_{N1} for both transconductors. One transconductor has VDD as its input and the other one has the V_{avg_tp} as its input. We see that as soon as V_D settles down, it has the same value for both transconductors. Also V_{GN1} are both at V_{N_Bias} because of the same biasing circuitry is used. And I_{N1} in both transconductors is the same due to the same overdrive voltage. Because of the subtraction, I_{N3} in Figure 4.10 is the current $I_T - I_{tp}$ needed by the current source. Figure 4.12 shows the current through each branch that connects to node V_D in Figure 4.10. Constant I_T = (76.72 – 5.67) = 71.05 µA, using the point the cursor is located at in Figure 4.12 as an example, I_{tp} = 13.54 µA is being conducted due to an 864.5mV input duty cycle average but the actual current that used as reference to the PMOS current source is (71.05 – 13.54) = 57.52 µA.

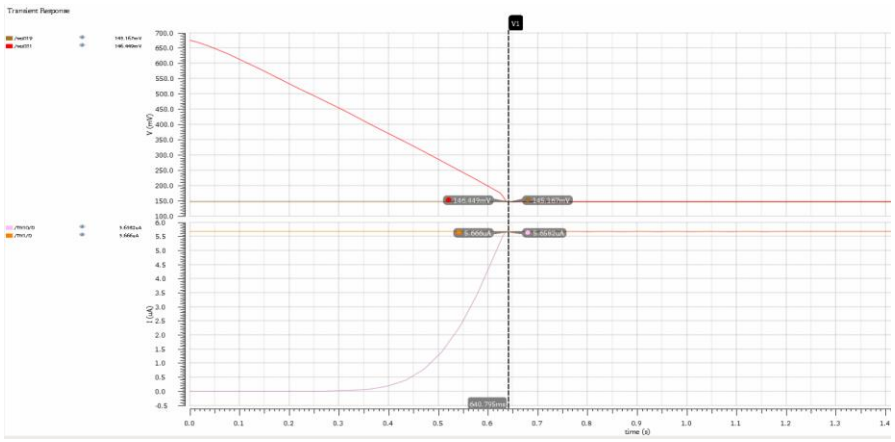

Figure 4.11 V_D and I_{MN1} comparsion between the transconductors in subtraction circuitry

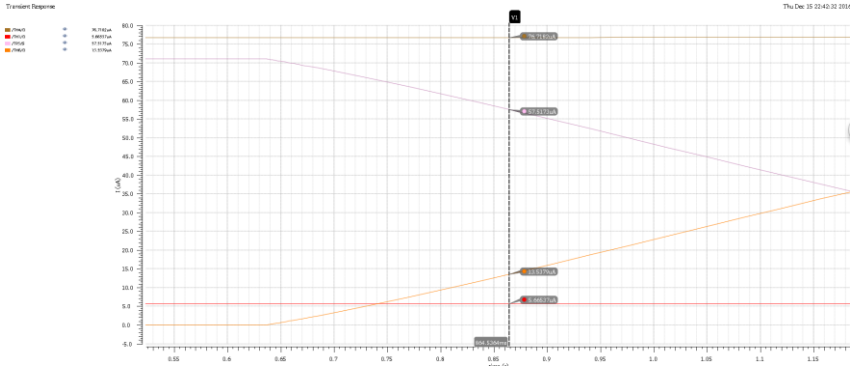

Figure 4.12 Current relationshp at node V_D of the transconductor with subtraction branch

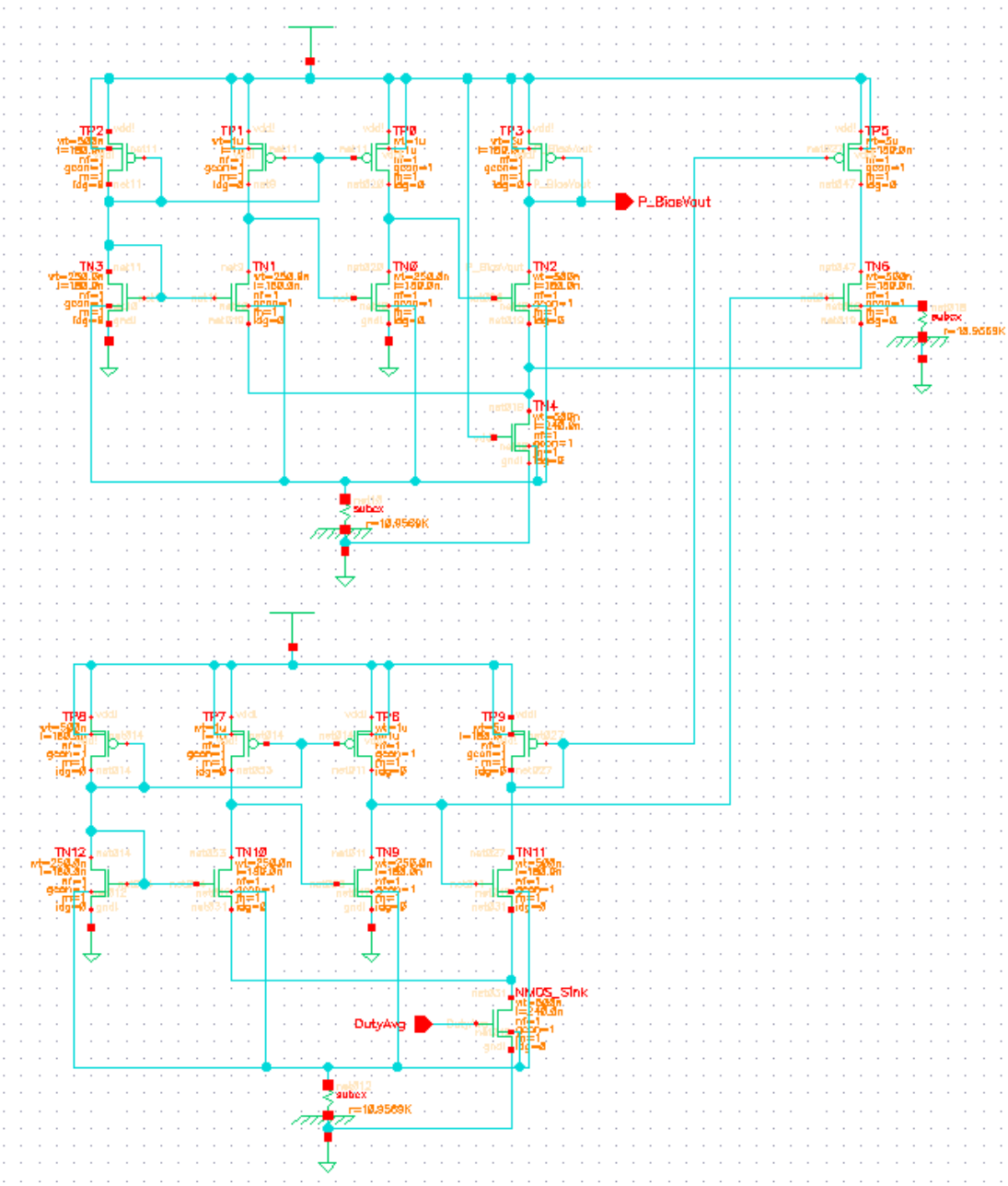

Figure 4.13 Full schematic of the linear transconductor used for current source biasing

P_BiasVout, the gate voltage at TP3 shown in Figure 4.13, is the voltage used for the PMOS source current control voltage in the integrator. And additionally, it is also used in the current summing circuitry which generates the NMOS tail current sink bias voltage for the integrator also.

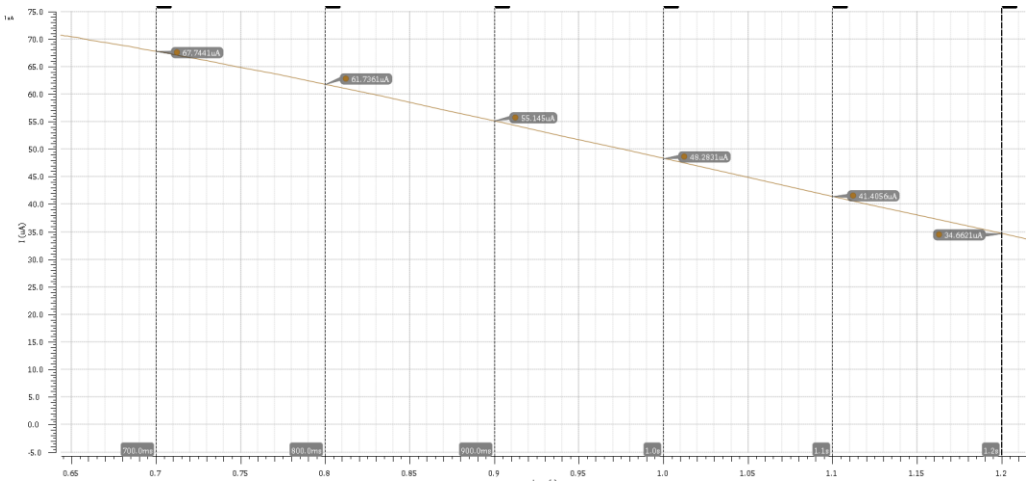

Figure 4.14 Linearity check of I_{TP3} of the combined V-I, within input range 0.7V to 1.2V

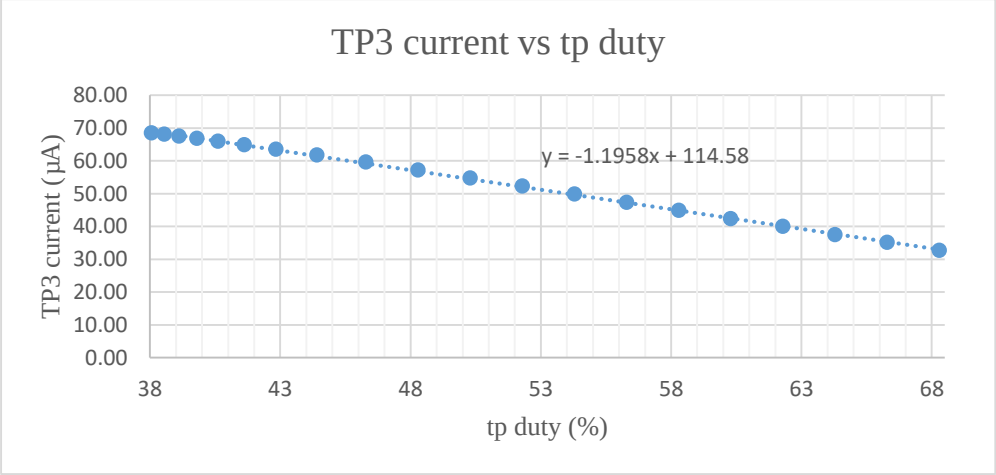

Figure 4.15 I_{TP3} vs t_p duty cycle

Figure 4.15 shows the current-duty relationship of the PMOS TP3 which has a linear-fit line slope of -1.1958 µA/1%$_{\_duty}$. And compared to the 1.2027 µA/1%$_{\_duty}$ current-duty slope of the NMOS_Sink, these two slopes match each other in magnitude but different signs. This is actually due to the relationship that $I_{charging} = I_{(T-tp)} = I_T - I_{tp}$, which means a better linearity on I_{tp} generation leads to a better linearity on $I_{charging}$ generation.

A simulation is done using the average voltage values listed in Table 4.1, 0.7146V for 39.7% duty and 1.1121V for 61.8% duty, and the currents are measured. The results are shown in Figure

4.14 and in Table 4.3. In the ideal case with the complementary pulse pair of 396ps and 604ps pulse width relative duty cycles, a changing/discharging current ratio of 1.523 is desired. Isolating the error passed down from the pulse generation block and treating the 396ps and 617ps width pulse pair as errorless inputs, the 'ideal' charging/discharging current ratio is 1.554. The actual measured ratio, however is 1.649 due to the imperfect linearity of the transconductor. It is 6.1% off from the perfect linear transconductor, and is 8.3% off from a perfect cascaded pulse generator and the perfect PMOS reference current generation transconductor.

	t_p	$T-t_p$	I_{charge}	$I_{discharge}$	$(T-t_p)/ t_p$	$I_{charge}/I_{discharge}$
Ideal	396ps	604ps	N/A	N/A	604/396=1.523	617/397=1.554
Actual	397ps	617ps	66.92 µA	40.59 µA	617/397=1.554	66.92/40.59=1.649

Table 4.3 Simulated current ratio compared to the ideal current ratio

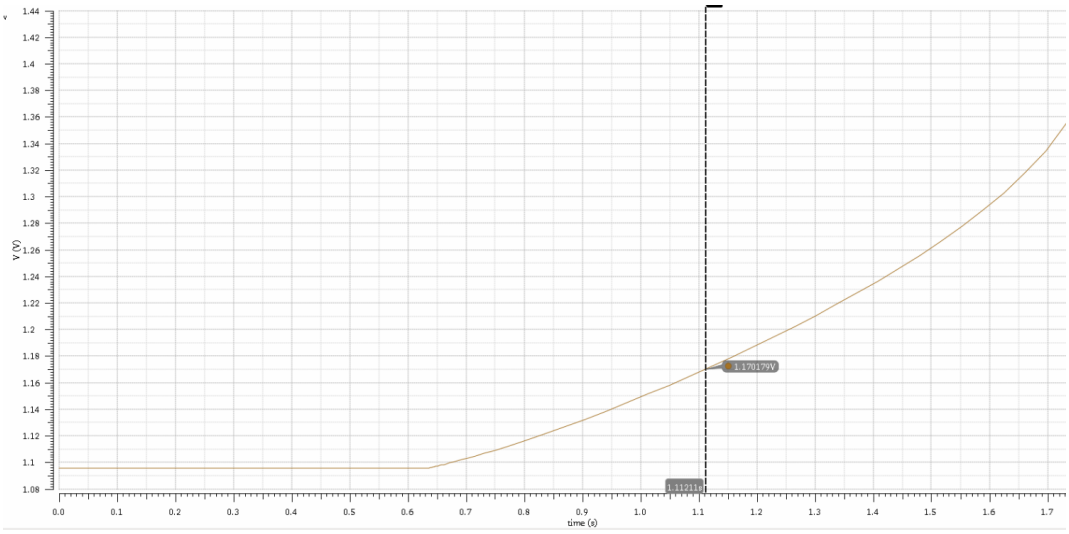

Figure 4.16 PMOS bias voltage P_BiasVout under the input sweep simulation

P_BiasVout is the bias voltage that drives the current source PMOS on the integrator, the output range of this bias voltage constrains the maximum output level of the integrator as we will

see in Chapter 5. Since the current source PMOS on the integrator has to stay in saturation mode, V_{DS} which is (VDD – V_{OUT}), must be higher than the overdrive voltage which is (VDD – $V_{P\_BiasVout}$ – $|V_{TP}|$). For the branch on the integrator with the PMOS responsible for the 61.7% duty cycle, $V_{P\_BiasVout}$ = 1.17V from Figure 4.16 and $|V_{TP}|$ is 0.41V based on the model file data, yielding a maximum V_{OUT} of $V_{P\_BiasVout}$ + $|V_{TP}|$ = 1.58V. In other words, anytime the integration output voltage goes above 1.58V, the PMOS will fall out of the saturation mode and then will not conduct a constant current source anymore. Similar analysis on $V_{P\_BiasVout}$ of 1.10V (39.7% duty) gives a maximum integrator output swing level of 1.51V. Consider the fact that ideally two differential integrating waveforms in the integrator are on top of each other, theoretically they should have to same maximum voltage swing level. The worst case (smaller swing) output voltage must be selected to keep the PMOS in saturation and that voltage is 1.51V.

4.2.3 Current summing and NMOS current sink bias

Recalling the current relationship equation $I_{SINK} = I_{charging} + I_{discharging}$, a circuit is needed to combine the charging and the discharging PMOS currents and generate the bias control voltage for the NMOS current sink on the integrator. A configuration similar to the integrator stage is capable of such a functionality and is shown in Figure 4.17. The two top PMOS transistors are biased by the bias reference voltages associated with the charging and the discharging current. The diode connected NMOS in each branch forces itself into the saturation mode and sinks the sum of whatever the top PMOSs are conducting which is $I_{charging} + I_{discharging}$.

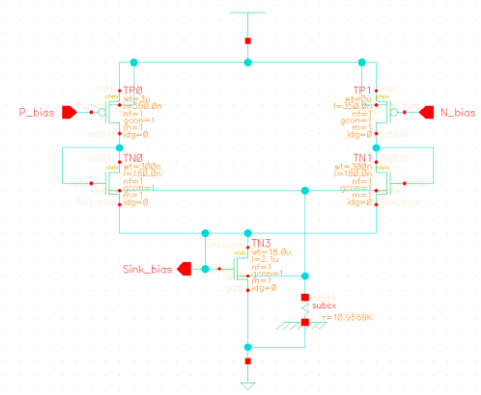

Figure 4.17 Current summing and NMOS sink bias generation circuitry

The presence of the diode connected NMOS and the two PMOS has a second function which is to make this current summing circuitry similar in structure to the integrator. Similar DC voltages at the drain nodes of the current source PMOS and the tail sink NMOS should show similar channel length modulation affects and should, therefore, provide more appropriate currents than a unlike structure might. It is important have similar DC bias voltage at these nodes between this current summing circuitry and the integrator. Furthermore, transistor sizing should be matched between the summing circuit and the integrator in order to keep them conducting the same amount of current. The actual conducted source current in the summing circuit and the integrator can be different from the reference current conducted in the linear transconductor depending on the design.

Though the actual charging and discharging source currents could be different from the references, the ratio should be preserved. In the design shown in Figure 4.17, and in the design in Figure 4.13 note that the PMOS used has a W/L ratio of 5/0.18 = 27.78, while the PMOS in the current summing circuit has a W/L ratio of 3/0.35 = 8.57. Since the ratios are equal, the currents should also be alike. PMOSs are in the saturation mode and if channel length modulation can be ignored, the current in the summing circuit will be scaled down by 27.78/8.57 = 3.24 times. But since the lengths of the PMOS in the transconductor are small, channel length modulation should not be ignored and it can introduce some mismatches.

Simulation results in Table 4.4 show the deviation between the expected value and the simulated result. We see that the simulated results are lower than the expected, primarily caused by channel length modulation due to the drain voltages being different. The PMOS drain voltage is lower in the transconductor, introducing a higher drain-source difference and therefore a higher current. This is predictable since the PMOS lengths used in this case are small at only 180nm and 350nm. However, in fact, what really matters is the ratio between the charging and the discharging current, which manages to keep $I_{charging}/I_{discharging} = (T-t_p)/t_p$ stays true. The transconductor generates a current ratio of 1.65, compared to an expected ratio of 1.55 with the case if perfect linearity is achieved. Yet, the current summing circuit has a better outcome for the current ratio of 1.59 and it will be the ratio being seen by the integrator.

	Expected	**Simulated**
$I_{charging}$ @ current summing	66.92/3.24 = 20.65 µA	15.09 µA
$I_{discharging}$ @ current summing	40.59/3.24 = 12.58 µA	9.46 µA
$I_{charging}/I_{discharging}$ @ current summing	66.92/40.59 = 1.65	15.09/9.46 = 1.59
$I_{charging}/I_{discharging}$ vs $(T-t_p)/t_p$	617ps/397ps = 1.55	1.59

Table 4.4 Conducted PMOS current in the current summing circuit

This feedforward auto-biasing block provides DC bias voltages for the current source and current sink transistors in the integrator, and, therefore, does not contribute jitter directly to the JRC system. However its performance is crucial to the system. First of all, the linearity of the transconductor and the accuracy of the current summing circuit could limit the actual charging/discharging current ratio, which is the quantity we must maintain as close to the target ratio as possible. Once the ratio is off, the triangular differential integrating waveforms from the integrator begin to diverge. In worst cases, the triangular waveforms do not cross each other therefore no recovery clock would be generated. On the other hand the low-pass averaging module affects the start-up time of the JRC system, and it has a possibility of loading the differential pulsing chain from the pulse generation block and therefore distorting the pulse signal. Feeding the distorted pulse control to the integrator brings in significant errors because the switching behavior becomes less predictable and non-reliable.

Chapter 5

INTEGRATOR BLOCK

5.1 Integrator operation theories

The integrator is responsible for the differential triangular waveform generation, which has the pre-designed linear slope m_1 and m_2. The integrator is implemented by using a differential pair as its core as shown in Figure 5.1. A differential pair has two current sources on top followed by two switches controlled by the pulse generation block, and then a tail current sink which constantly conducts a current equal to the sum of the two current sources. At the two nodes between current source and the switch, two single capacitors are attached and used as integrating devices. As current is pushed on or pulled off from the capacitor over certain amount of time, it integrates the net current through and translates that into voltage output. As long as the net current flows through the integrating capacitor is constant for a specific amount of time, the voltage output is a linear ramp for that duration. A constant current means a linear slope as shown in Eqn 5.1.1.

$$Q = CV \rightarrow \frac{dQ}{dt} = C\frac{dV}{dt} \rightarrow \frac{dQ}{dt} = I \rightarrow I = C\frac{dV}{dt} \rightarrow \frac{dV}{dt} = \frac{I}{C} \quad \text{(Eqn 5.1.1)}$$

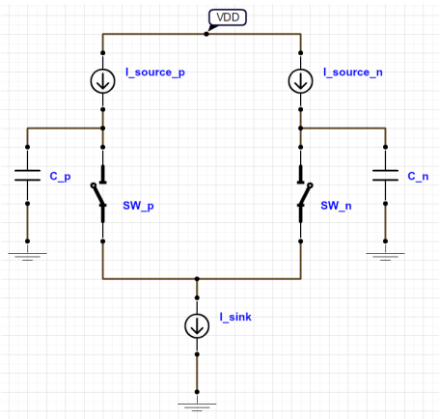

Figure 5.1 Integrator implementation

I_{source_p} and I_{source_n} in Figure 5.1 represent two constant current sources with different current values. Two switches sw_p and sw_n are driven by the complementary pulse train from the

pulse generation block and therefore only one switch is closed at a time while the other one is opened. The tail current sink is biased to conduct the sum of the sourced currents I_{source_p} and I_{source_n} all the time no matter what status the switches are. The branch with its switch opened (no current) has the conducting current charging the capacitor. All of the current from the source goes onto the capacitor. As soon as the switch closes (starts to conduct), the source is still pushing the same amount of current onto the node that the capacitor is connected to however, now, the connected tail sink is also connected to that node and it pulls a larger amount of current off that node so current is pulled off of the capacitor to make up for difference in the sourced and sunk current. Since the tail current sink maintains a current equals to the sum of the sourced currents, the net current being pulled out from the capacitor has an equivalent value of the sourced current on the other branch. For example, when sw_p opens, sw_n is closed. That means that I_{source_p} is charging C_p while $I_{sink} - I_{source\_n} = I_{source\_p}$ is the current being pulled off of C_n. After flipping the switches, sw_p is closed while sw_n is opened and now I_{source_n} is being pulled off of C_p and I_{source_n} is being pushed onto C_n.

By setting I_{source_p}, I_{source_n}, I_{sink} and the complementary switching controls appropriately, integrator's voltage output satisfies the fundamental equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$. In fact as we have discussed in the previous chapters, I_{source_p}, I_{source_n}, and I_{sink} are biased by the feedforward auto-biasing block to maintain the correct charging/discharging current ratio, while the differential switching controls are provided by the pulse generation block.

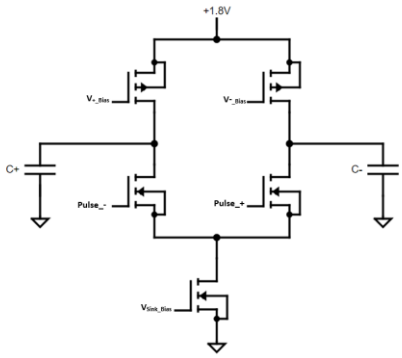

Figure 5.2 Integrator implementation schematic

Figure 5.2 shows the implementation of the integrator with two PMOSs as the constant current sources, two NMOS as the switches and a large NMOS as the tail current sink. The bias voltages V_{+_Bias}, V_{-_Bias} and V_{Sink_Bias} come from the auto-biasing block discussed in Chapter 4, and the switching controls pulse_- and pulse_+ are from the pulse generation block. The biasing voltages keep the PMOS current sources constant and in the saturation mode as does the biasing to the tail NMOS. Pulse_+ and pulse_- turn the NMOS switches on and off in order to steer the charging/discharging current to the integrating capacitors. Smaller-sized NMOS should be used as switches to increase the switching speed. Smaller sizes also mean that the pulse generation block is loaded less.

The voltage change on the capacitor during integration can be written as $\Delta V = \frac{\Delta Q}{C}$ where ΔQ is the change of charge during a specific amount of time. The slope of integration, $\frac{\Delta V}{\Delta t}$ therefore could be written as $\frac{\Delta Q}{C \cdot \Delta t} = \frac{I}{C}$, where I is the net current onto or off of the capacitor. Due to the linear integrating characteristics of a capacitor, the voltage output swing is directly related to the net current and the capacitance. As we have stated above, the two current sourcing PMOS must stay in saturation and this requires their $|V_{DS}|$s to stay above their $|V_{GS} - V_T|$. If V_{OUT} swings too high, in other words, the capacitors keep charging even after $V_{DS} = V_{GS} - V_T$, the PMOS will be forced into linear and it won't be conducting the required current anymore. The maximum output level can be found by subtracting V_{Dsat} from VDD where:

$$V_{Dsat} = \sqrt{\frac{I_{DS}}{(\frac{k'}{2} \cdot \frac{W}{L})}}$$

Because the k', W and L are the same for both PMOS, it is the current that determines how large or small V_{Dsat} is. There are some tradeoffs here. A small V_{Dsat} (a small current) means that the PMOS will stay in the saturation region for higher values of V_{out}. Also, for the same swing, a smaller current means you need to use smaller capacitors which means less area used by the circuit.

The tradeoff is that with smaller currents the circuit is more susceptible to feedthrough, noise and charge sharing. In this case charge sharing is not a problem but noise and feedthrough could still cause variations in the shape of the triangle wave. Feedthrough would come from the pulses from the pulse generator block. No other signals are moving into the integrator block. A falling edge would lower the value on the capacitor that shares a node with the transistor receiving the pulse and a rising edge would raise it. An approximation of the degree to which the node is affected by the pulse can be found by hand but it will not be as accurate as we need so a decision was made at this point to use simulation to investigate these problems instead of doing detailed calculations by hand.

Another value that needs to be well controlled is the bias voltage to the gate of each PMOS from the auto-biasing block. Also, in this process, V_T is affected by both the W and L of the transistor. Because the V_T of the transistor is part of the edge of saturation condition, the threshold of the PMOS needs to be considered as well. This was done in Chapter 4 while we were examining the PMOS bias output range and 1.51V was chosen as the maximum output swing level. As a result the selection of I and C should include this constraint which will be covered later in this chapter.

Note that pulse_- which represents the t_p' pulse chain (a constant width of t_p with output level low) is connected to the left branch shown in Figure 5.2, therefore the left switch would be opened with a duration of t_p and the capacitor is charged for that amount of time. For the rest of the time, the left switch is closed, causing the capacitor to discharge.

5.2 Differential integration characteristics under non-idealities

In this section the sources of integration error will be explored individually and then total error summarized.

5.2.1 Mismatched charging/discharging current ratio

The fundamental equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$ requires $I_{charging}/I_{discharging} = (T - t_p)/t_p$. If for some reasons this requirement is disturbed, for example the mismatching or channel length modulation between the PMOS current sources of the integrator, the fundamental equation is not satisfied anymore and the differential waveforms won't share the same average.

Imagine a situation that the one of the current source PMOS in the integrator has a larger size than the other one due to process variation or that V_{DS} is greater on one side and λ is not negligible. Then that side would conduct a larger charging current than the expected value set by the feedforward auto-biasing block. Consider the correct charging and discharging currents as $I_{charging}$ and $I_{discharging}$, the extra conducted charging current as Δi. The modified equation for one of the differential waveform becomes

$$\frac{I_{charging} + \Delta i}{C} \times t_p - \frac{I_{discharging}}{C} \times (T - t_p) = I_{charging} \times t_p - I_{discharging} \times (T - t_p) + \Delta i \times t_p = \Delta i \times t_p$$

where $I_{charging} \times t_p - I_{discharging} \times (T - t_p)$ remains 0. Similarly the other differential waveform becomes

$$-\frac{I_{charging} + \Delta i}{C} \times t_p + \frac{I_{discharging}}{C} \times (T - t_p) = -I_{charging} \times t_p + I_{discharging} \times (T - t_p) - \Delta i \times t_p = -\Delta i \times t_p$$

It shows that for one of the differential waveforms, there is a net voltage increment of $\Delta i \times t_p$ after each integrating clock cycle, while the other waveform has a net voltage decrement of $-\Delta i \times t_p$. Figure 5.3 shows the behavior of such a pair of differential waveforms. Since the differential voltage levels at the ending point of each clock cycle move to the opposite directions, two waveforms diverge from each other therefore there may be no crossing point.

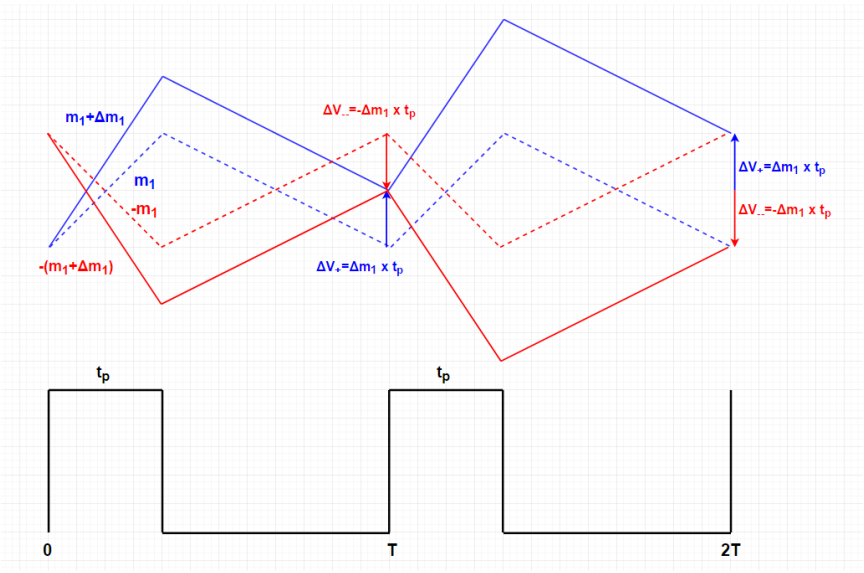

Figure 5.3 Integration diverging due to current ratio error

The same analysis could be applied to the case with a discharging current that is higher than what it should be. This would cause the same diverging behavior but with the divergence direction is swapped from the one in Figure 5.3.

As a consequence it is very important to improve the accuracy of current ratio and reduce the influence due to mismatching, channel-length modulation, error and noise.

In this section the error introduced from mismatches in the current sources was evaluated. In the next section the error from a varying t_p is explored.

5.2.2 Variation on t_p

Constant pulse width t_p is critical to maintain the fundamental equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$, any variation on t_p displaces the crossing points on the differential waveforms therefore introduces timing error to the output clock directly.

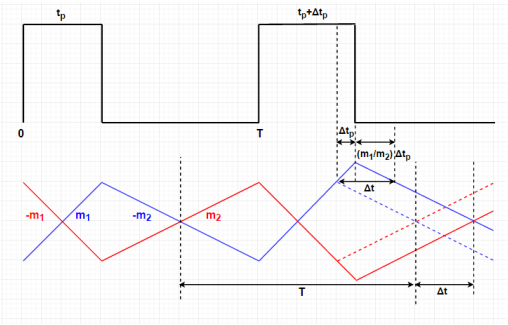

Figure 5.4 Integration crossing point timming error due to t_p variation

In Figure 5.4, the second pulse width has a positive variation Δt_p as a result it is longer than the expected value. Due to the extra charging time Δt_p, V_+ is charged up more by the amount of $m_1 \times \Delta t_p$ and it takes $\frac{m_1}{m_2} \times \Delta t_p$ extra discharging time in order to have V_+ and V- cross again. The total time variation Δt between the ideal and the actual crossing $\pm m_2$ point is therefore

$$\Delta t = \Delta t_p + \frac{m_1}{m_2} \times \Delta t_p = (1 + \frac{m_1}{m_2}) \times \Delta t_p$$

Recall that $\frac{m_1}{m_2} = \frac{I_{charging}}{I_{discharging}} = \frac{T - t_p}{t_p} = \frac{T}{t_p} - 1$, plug this into Δt equation, we have

$$\Delta t = \frac{T}{t_p} \times \Delta t_p \qquad \textbf{(Eqn 5.1)}$$

For the case with a negative pulse width variation, Δt equation still applies with Δt_p now being negative. Eqn 5.1 means a steeper m_2 helps to reduce the timing error caused by Δt_p, and this requires a longer t_p. Variation on t_p could be caused by the pulse generation block itself, but the threshold variation on integrator's NMOS switches could also affects the effective t_p length. A reduced threshold on the switch is equivalent to a longer pulse width, while an increased threshold could be considered as a reduction on pulse width. One way to minimize mismatch between the NMOS switches is to have them being near each other with the same orientation.

5.3 Integrator circuit implementation

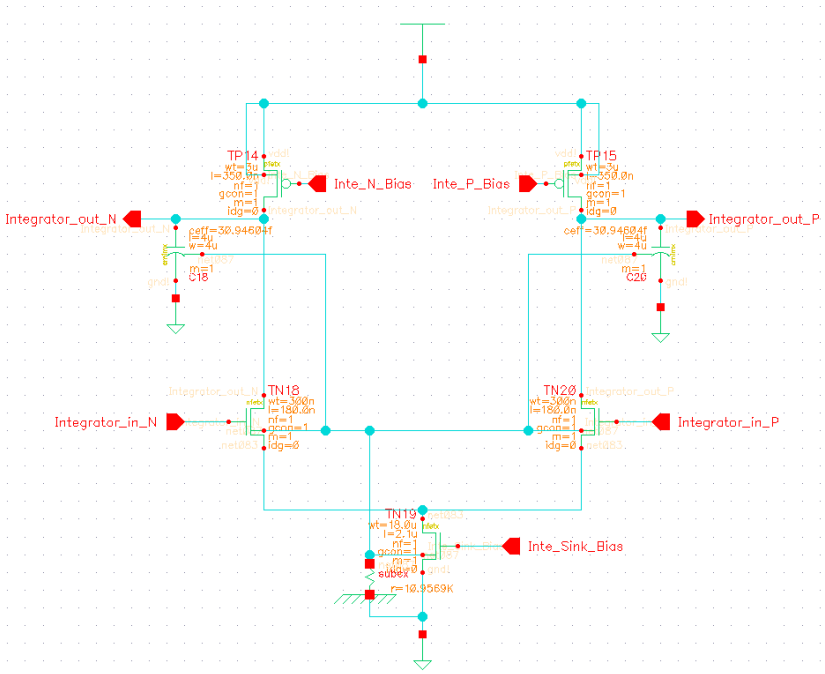

Figure 5.5 Integrator full schematic

Figure 5.5 shows the implementation of the integrator. The NMOS switches are small compared to the other transistors in the integrator, in order to achieve better switching response by reducing the input capacitance. The PMOS current sources have a large W/L ratio in order to decrease channel-length modulation and the outputs are connected to the square-wave reform block. As mentioned before, the PMOS must stay in the saturation region to be considered as a constant current source and it is controlled by the auto-biasing block. A larger W/L ratio reduces the minimum source-drain voltage required for saturation mode, which is $V_{SD\_Sat} = \sqrt{\frac{I_{SD}}{\frac{\mu_p C_{OX}}{2}\frac{W}{L}}}$. A reduced V_{SD_Sat} keeps the PMOS further away from its linear operation region while the drain terminal of the PMOS, which is moving during the operation therefore V_{SD_PMOS} is altering.

Even if the PMOS is biased to stay in saturation, there are other non-idealities that could affect the current. Channel length modulation could still kick in to alter the conducted current.

When the output node voltage moves, the VDS across the PMOS changes and channel-length modulation will alter the current. Recall the current equation in saturation mode:

$$I_{SD} = \frac{1}{2}\frac{W}{L}\mu_p C_{OX}(V_{SG} - V_{TP})^2(1 + \lambda V_{SD})$$

where λ is the channel length modulation coefficient, and $\lambda = \alpha\frac{\Delta L}{L}$. Because ΔL is independent of transistor sizing and is completely dependent on the drain voltage, it is clear that a larger length L reduces the channel length variation ratio $\frac{\Delta L}{L}$ and therefore reduces the dependence between I_{SD} and V_{SD}.

Combining the requirements stated above, a larger W/L device with a longer L gives a better constant current sourcing performance. However as the PMOS is sized up and lengthened, it adds extra transistor-contributed capacitance to the output node. The capacitance contributed by the transistor is voltage-dependent, it changes when the voltages on the transistor change. This means the total capacitance at the output node is a combination of a fixed value determined by the capacitor and a varying value contributed by the transistors connected to that node. Therefore as the PMOS is sized up more, the output capacitance varies more as output voltage changes.

Use the results from Table 4.4, which shows the simulated $I_{charging}$ and $I_{discharging}$ for 3μm/350nm PMOS current sources biased by the feedforward auto-biasing block, with t_p = 396ps.

	Simulated result
$I_{charging}$ by 3μm/350nm PMOS current source	15.09 μA
$I_{discharging}$ by 3μm/350nm PMOS current source	9.46 μA

Table 5.1 3μm/350nm PMOS current by auto-biasing for t_p=396ps case

To achieve a 200mV output swing (396ps charge time), the integrating capacitor value could be calculated as:

$$\frac{I_{charging}}{C} = \frac{V_{swing}}{t_p} \quad \rightarrow \quad C = \frac{V_{swing}}{t_p} = \frac{I_{charging} \cdot t_p}{V_{swing}} = \frac{15.09\mu A \cdot 396ps}{200mV} = 30fF$$

The requirement of large W/L ratio with longer L also applies to the tail NMOS current sink, and since it is only responsible for conducting a constant DC current through the operation, neither the large capacitance nor the long channel of the tail current source would affect the integrator output. Therefore an 18 µm/2.1 µm NMOS is used to accommodate low V_{DS_Sat} and low $\frac{\Delta L}{L}$ which keeps the tail current source NMOS transistor in saturation mode.

5.4 Integrator performance analysis

A fully functional integrator needs biasing voltages and switching controls, and, therefore, for testing purposes, the feedforward auto-biasing block and the pulse generation block will be used to provide these necessary inputs. However errors from these two blocks are passed to the integrator and affect its outputs. For a better analysis on integrator's performance we first include the auto-biasing block but use ideal switching signals. Then as a second analysis, the ideal switching signals will be replaced with the pulses from the pulse generation block.

5.4.1 Ideal switching control, no jitter

Based on the discussion in Chapter 3, the ideal differential switching control pulses should have a constant width of 396ps and 604ps. This ideal switching pair is fed to the auto-biasing block to generate the appropriate biasing voltages, and it also drives the NMOS switches of the integrator to steer the current flow.

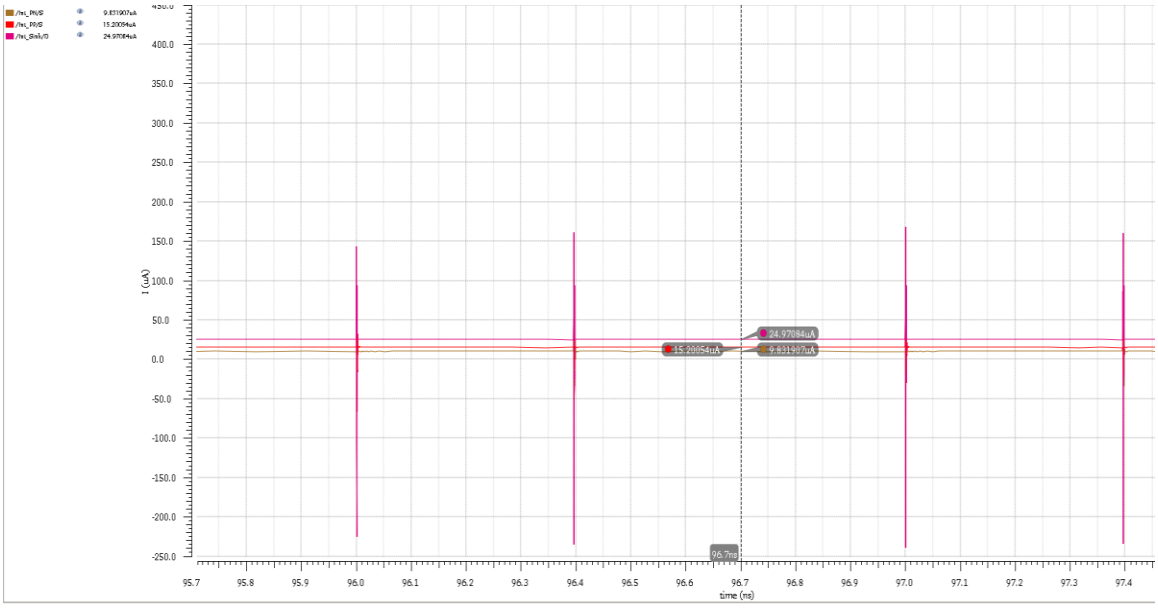

Figure 5.6 Integrator sourced currents and sunk current with ideal switching, no jitter

Figure 5.6 shows the sourced current from the two PMOSs and the sunk current through the tail NMOS. The PMOS which charges the capacitor during t_p sources a nearly constant 15.2 µA, while the other one sources 9.83 µA. This yields a charging/discharging current ratio of 1.55 compared to the expected value of (604ps/396ps) = 1.53. The tail current sink conducts 24.97 µA, and this matches the sum of the PMOS currents which is (15.2+9.83) = 25.03 µA.

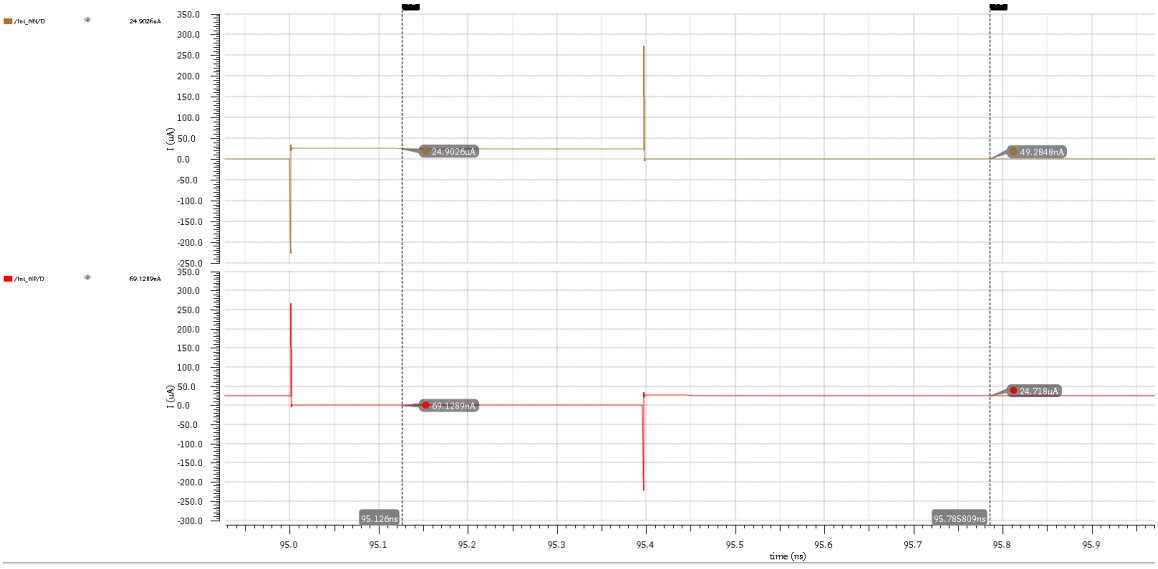

Figure 5.7 Integrator NMOS-switch currents with ideal switching, no jitter

Figure 5.7 shows the current through the NMOS switches during the operation. We see that only one of the switches is conducting at a time, with the current equals to the sunk current by the tail NMOS. However we also notice the conducted currents for each switch while they are on, are slightly different (24.7 µA for V_+ branch and 24.9 µA for V_- branch) which indicates a slightly varying sunk current. And, there is leakage current shows up when the NMOS switches are turned off (49nA for V_+ branch and 69nA for V_- branch). Due to such a slightly varying sunk current, capacitor current mismatches are expected and shown in Figure 5.8.

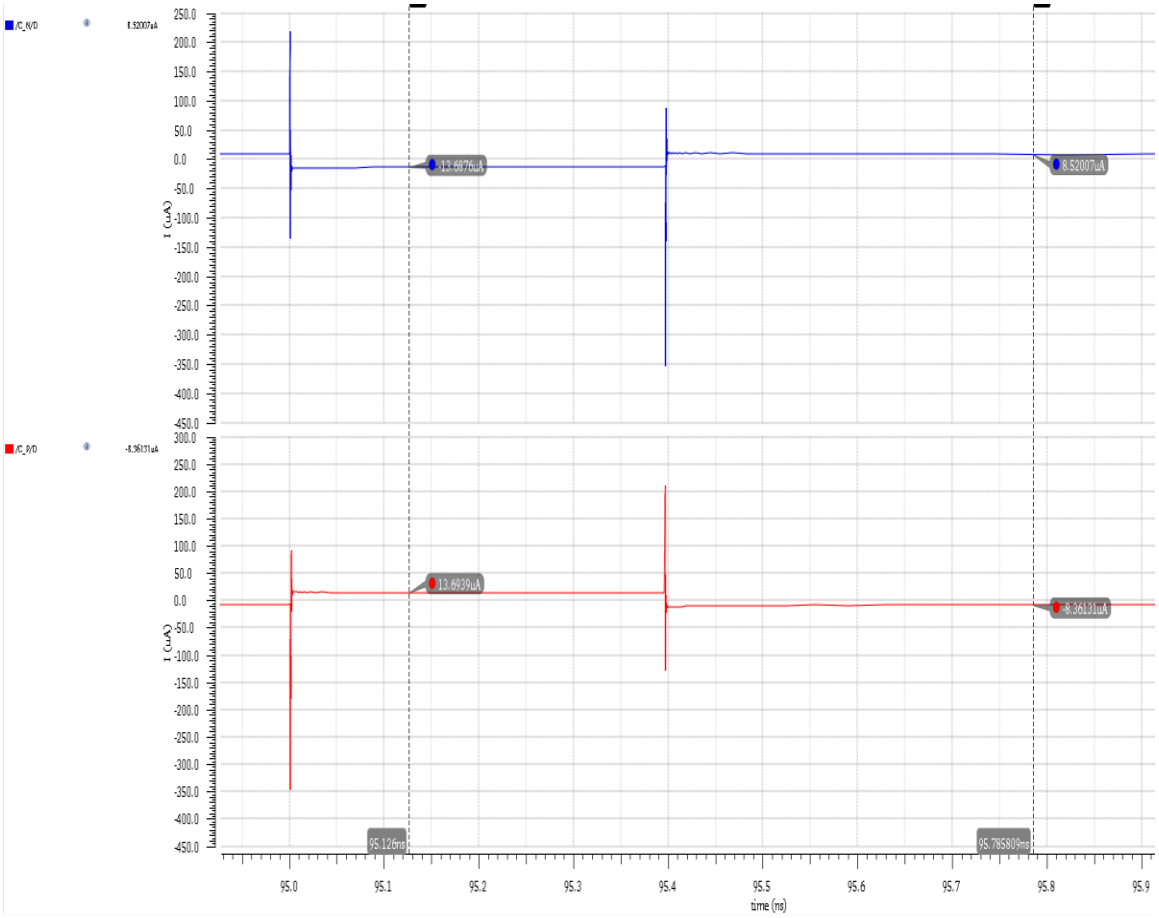

Figure 5.8 Integrating capacitor currents with ideal switching, no jitter

During t_p the V_+ capacitor C_+ is being charged with 13.69 µA while C is also being discharged with 13.69 µA. After switching, C_+ discharges with 8.36 µA while C- discharges with 8.52 µA. The actual charging/discharging currents through the capacitor are lower than the sourced currents by the PMOS, due to leakages. However as pointed out previously, what really matters is the charging/discharging current ratio. For C_+ the current ratio is 13.69/8.36 = 1.63, and a ratio of 13.69/8.52 = 1.61 for C-. But the tiny current variation between two capacitors implies the triangular waveforms might not be on top of each other perfectly. Figure 5.9 shows the adjacent $\pm m_2$ crossing points are 1ns apart however the differential triangular waves do not have the averages on top of each other.

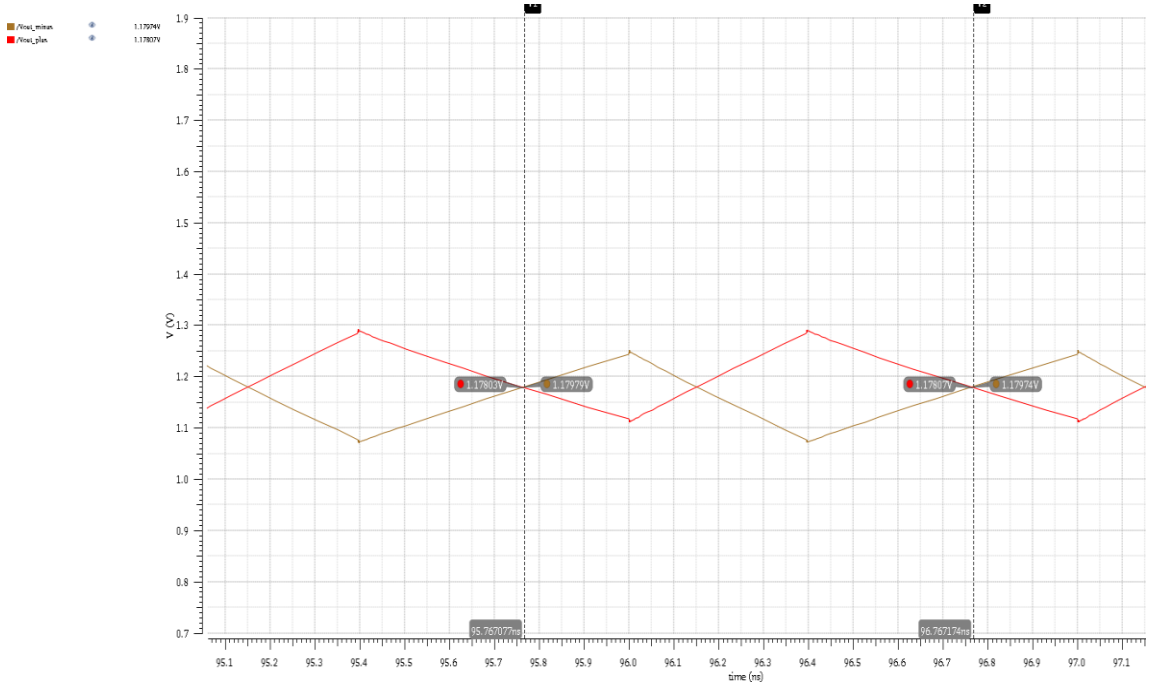

Figure 5.9 Differential integration crossing point period with ideal switching, no jitter

5.4.2 Switching controls by Pulse Generation Block, no jitter

Replacing the ideal switching control by the pulse generation block developed in Chapter 3, will allow us to see how the non-ideal switching signal affects integrator's performance. Most of the analyses in this section will be based on observations from simulation results since the non-idealities bring in significant difficulty in coming up with a mathematical description of the behavior.

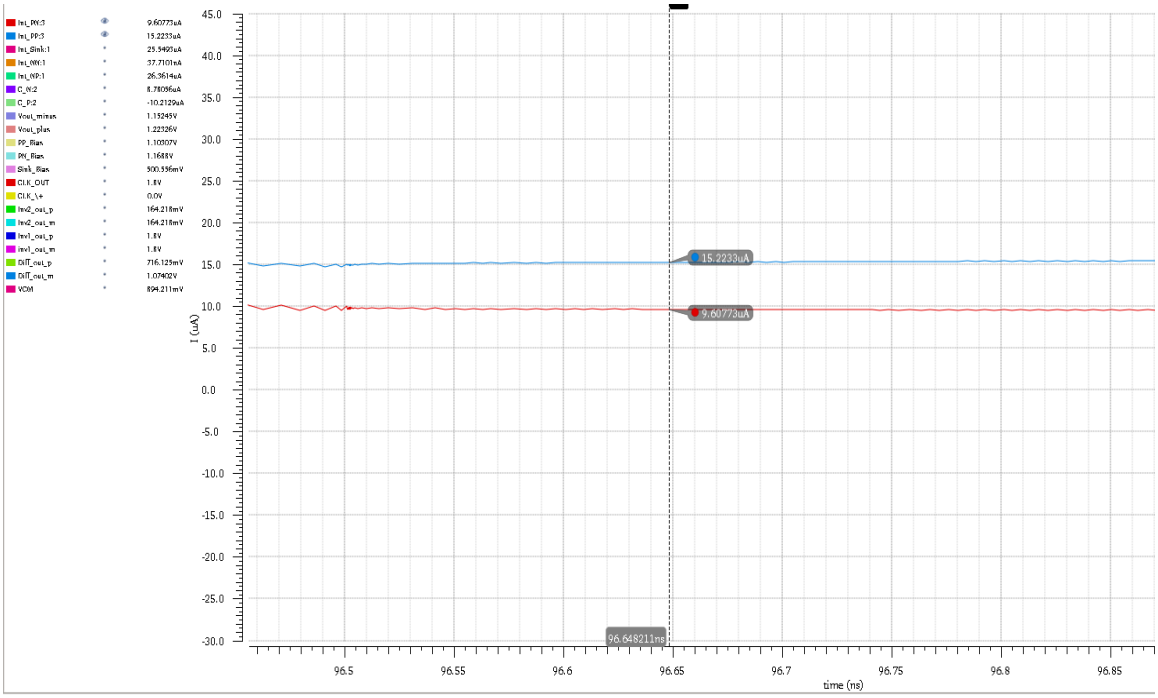

Figure 5.10 Integrator sourced currents with actual switching, no jitter

Figure 5.10 shows the currents conducted by the PMOS current sources during the operation. We see the currents have tiny variations that follow the trend of the integrating output voltages (which is not shown in the plot) due to the channel length modulation. However the variation amplitude is 0.05 µA peak-to-peak (for a 200mV peak-to-peak output voltage swing), therefore we can conclude that a 350nm channel length works pretty well to suppress the current variation due to the varying V_{SD}. The averages for the charging and the discharging current are 15.223 µA and 9.608 µA, which yields a ratio of 15.223/9.608 = 1.58. Recall that the widths of the pulse from the pulse generation block are 396ps and 617ps which suggests a current ratio of 1.56.

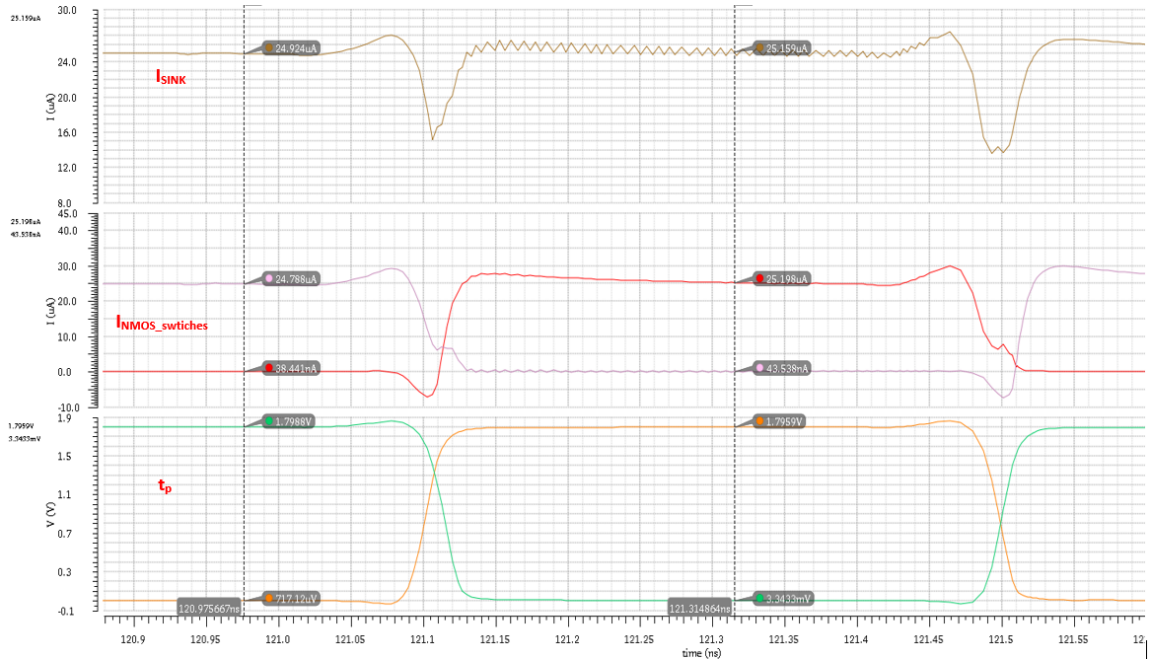

Figure 5.11 Integrator tail and NMOS-switch currents with actual switching, no jitter

In Figure 5.11 we see that the tail maintains a pretty solid sunk current of 25 µA outside the switching region. However unlike the case with ideal instantaneous switching event, current drops show up while the NMOS are switching. This is because the two switches are not fully turned off when the other switch turns on due to the non-zero rise and fall time of the switching control signals from the pulse generation block. Ideally in each switching event, one switch fully turns on and conducts whatever the tail is sinking while the other switch shuts off completely with no current flowing through at exactly the same time with no time when both are on. In addition there is a timing mismatch between the rising and the falling edge therefore each switch reaches the threshold level in different time. Furthermore, the turn-on and turn-off thresholds for the same NMOS are not necessary the same. Also notice that the feedthrough spikes at edges of the switching control show up as positive and negative current spikes at the beginning of each switching event. The tail sinks current equal to the sum of the currents through the NMOS switches at all times, and therefore the combination of non-idealities mentioned above make the sunk current drop while switching. In

Chapter 3, 0.9V was selected as the switching threshold therefore we expect to see a very little drop for the second switching event shown in Figure 5.11 because the crossing between both edges is almost on top of the 0.9V level. However that drop is deeper and wider than the first dip which with a larger "active-time" overlapped between two switches.

Due to the sink current variation, the slope of the integrating waveforms at each switching event is different than what it ideally should be. Recall that $I_{discharging} = I_{Sink} - I_{charging}$ and when I_{Sink} changes its value at the switching edges, the resultant current is off which results in the slope $m = I_{discharging}/C$ also being off. As a consequence, we can predict rounded turning points of the triangular integrating waveform, which could alter the locations of the $\pm m_2$ crossing point and introducing deterministic jitter to the output.

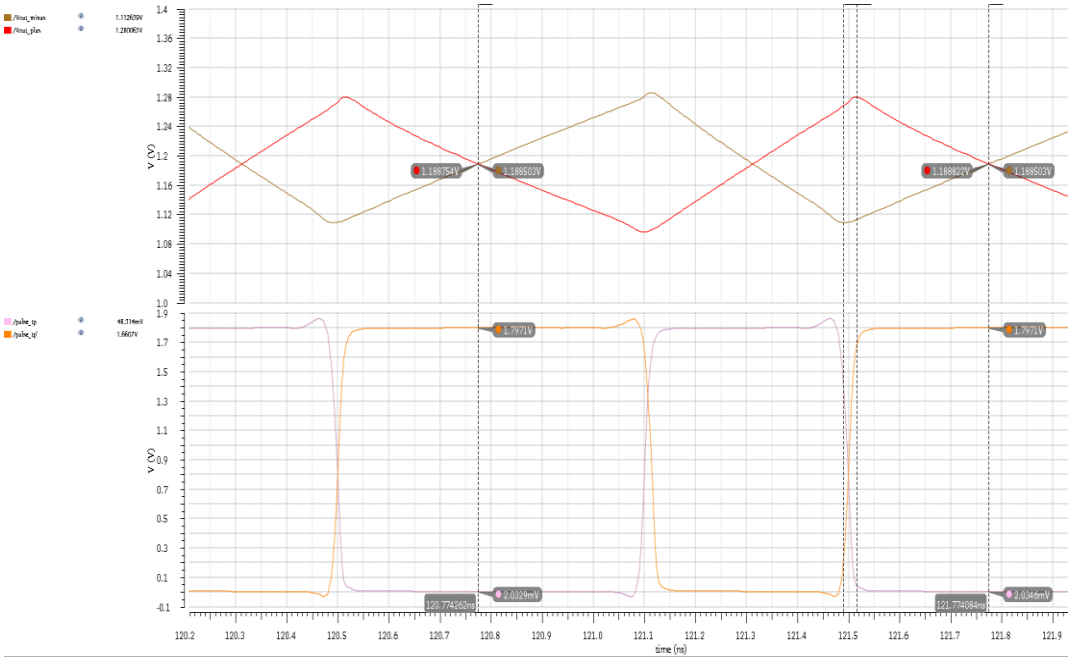

Figure 5.12 Differential integration crossing point period with actual switching, no jitter

Figure 5.12 shows the final differential triangular integrating outputs. The two waveforms, as predicted, have rounded turning points and some slope variations around the peaks.

Despite of the imperfections shows up on the output waveforms, time period in-between adjacent $\pm m_2$ crossing points remains approximately 1ns with sub-nanosecond variations. The variation implies system-contributed jitter exists, and this will be measured in Chapter 7. The crossing points stay at a voltage level of 1.19V.

5.4.3 Switching controls by Pulse Generation Block, with jitter

In Chapter 3 we tested the pulse generation block with a noisy input clock that contained Gaussian-distributed jitter, with $\mu_{jitter} = 0$ps and $\sigma = 33$ps and proved that the pulse generation block produces constant width pulse on each rising clock edge. The same jittery clock is reused as the input of the cascaded system here, to examine the jitter reduction performance.

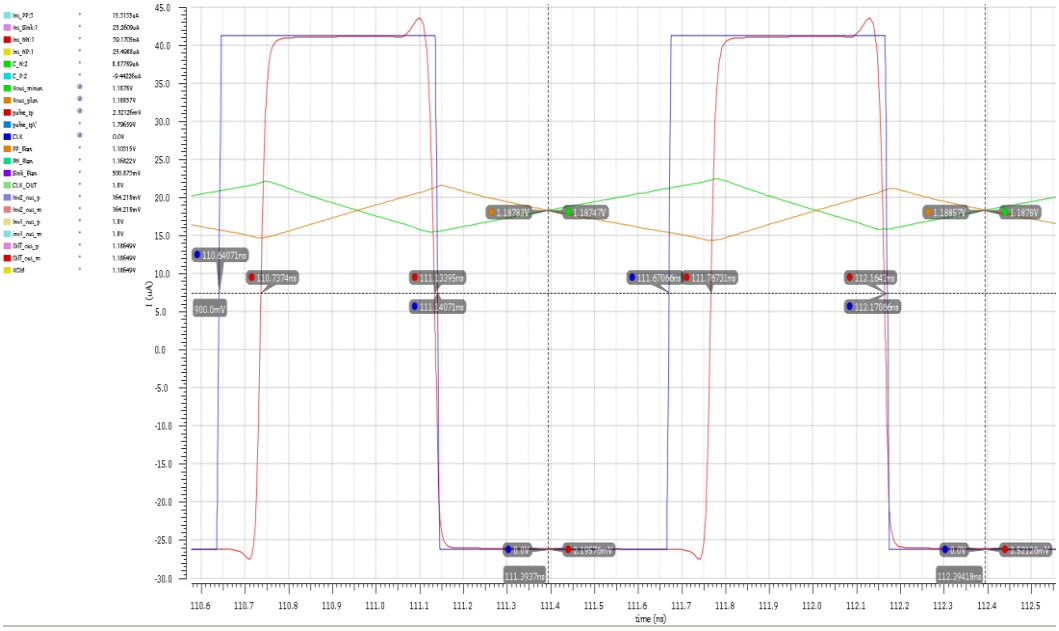

Figure 5.13 Integration crossing point period with 30ps input jitter

The clock cycle shown in Figure 5.13 has a positive jitter of $(t_2 - t_1 - T_{reference})$ = (111.670ns – 110.640ns) – 1ns = 30ps. Constant 397ps pulse is generated to represent each rising edge. The time difference between the associated $\pm m_2$ crossing points is 112.39418ns – 111.3937ns =

1.00048ns, a jitter of 0.48ps which yields a jitter reduction of -35.9dB on that clock cycle. In Figure 5.14, -29ps jitter shows up on input clock's rising edge but jitter measured on the associated $\pm m_2$ crossing points is only 1ps, which is a reduction of -29.2dB on that clock cycle.

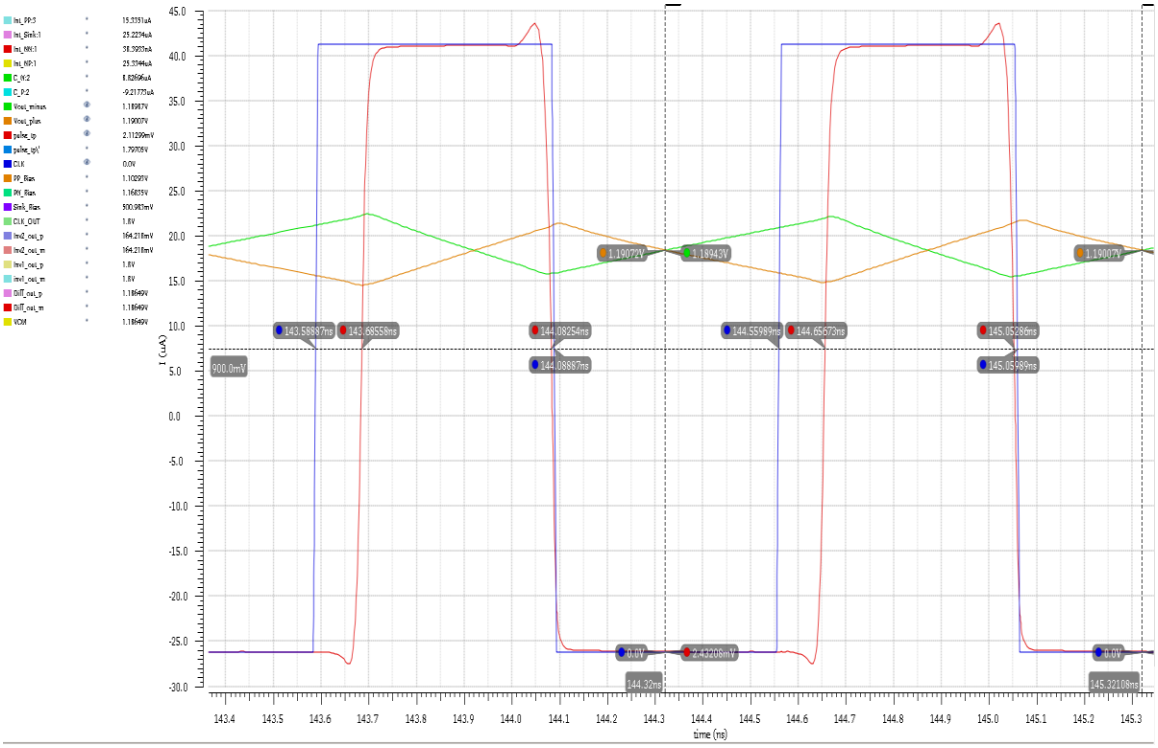

Figure 5.14 Integration crossing point period with -29ps input jitter

Recall that as long as the fundamental equation $m_1 \times t_p - m_2 \times (T - t_p) = 0$ stays true, each of the differential integrating waveforms has the same voltage level as it has at exact one targeted period T before. In other words, the $\pm m_2$ crossing points of the differential waveforms are maintained at a certain voltage level. Figure 5.15 shows the crossing point intercepts of the differential waveforms of the same jittery input clock used above. As seen, after the cascaded system has settled down, the differential waveforms have their $\pm m_2$ slope crossing points at the same voltage level of 1.19V, which implies that the fundamental equation is met.

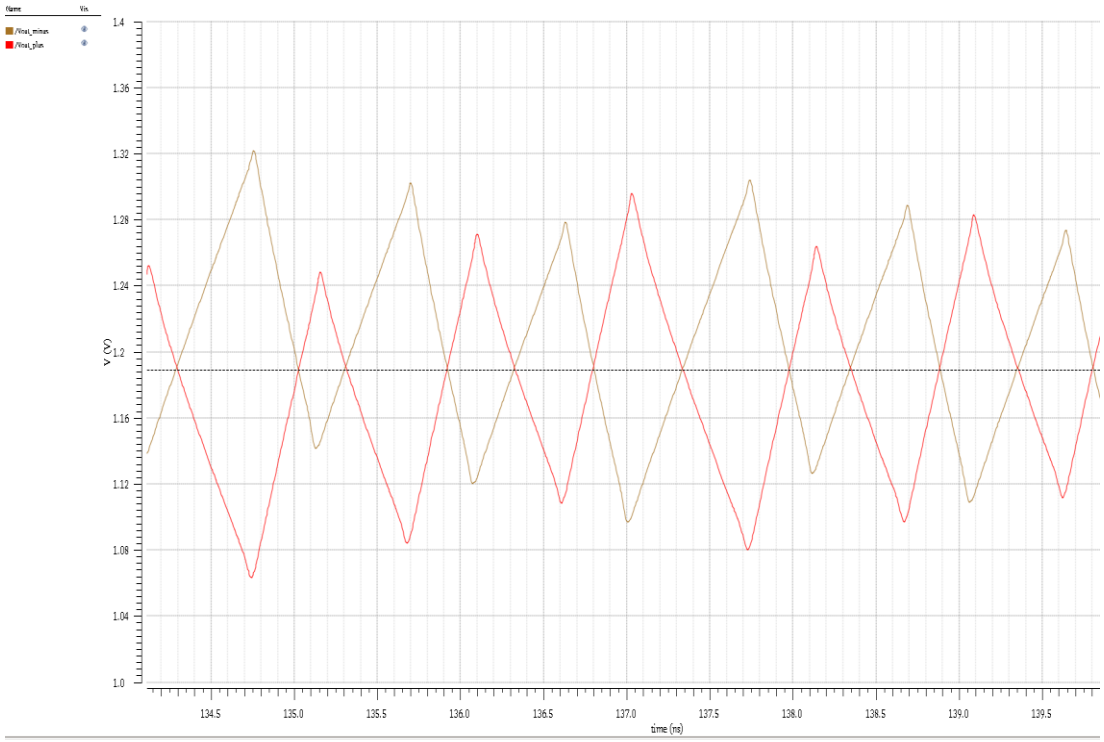

Figure 5.15 Integration crossing point output level with input jitter

Figure 5.16 Integrator period jitter reduction simulation plot

Figure 5.16 shows the plot of period jitter measurement for each incoming clock's rising edge and the period jitter measurement of the $\pm m_2$ crossing points at 1.19V level. The random jitter on the original jittery clock has a range from 100ps to -80ps as shown in the plot, and the jitter of the crossing points by the integrator is bounced within a range from 30ps to -30ps. Table 5.2 lists the rms value of the input/output period jitter and the jitter reduction in dB scale.

$\mathbf{t_{period\_jitter\_rms\_IN}}$	32.47ps
$\mathbf{t_{period\_jitter\_rms\_OUT}}$	12.76ps
$\mathbf{20log(t_{jitter\_OUT} / t_{jitter\_IN})}$	-8.11dB

Table 5.2 Integrator jitter reduction performance result

The output square-wave reform block will be cascaded to the system we have constructed so far. This reform block must preserve the reduction ratio provided by the integrator, which means the reform block should not contribute extra jitter on the output.

Chapter 6

SQUARE-WAVE OUTPUT REFORM BLOCK

This section discusses the final block which takes the output triangle wave of the integrator block and turns it into a square wave. This block will be different depending on what type of system the clock will be used in. For example, if the logic that will be using the clock sign al is based on MCML logic, a full swing is not needed and a simple MCML inverter will be enough to generate the needed clock signal. If the logic using the clock needs a full swing clock signal then the clock will need to have an additional block to create a rail to rail signal. Other possible output clock requirements may include a single-ended instead of differential clock, a clock that is centered around a different voltage and/or a clock with a fixed duty cycle.

6.1 Square-wave reform output block operation theory

Square-wave reform output block of the JRC system is responsible for the crossing point detection and the square-wave output clock generation. It takes the differential triangular waveforms from the integrator as inputs, senses the $\pm m_2$ crossing time and generates sharp square-wave rising edges at each crossing point detected. As a consequence the output square-wave has its rising edges representing the recovered clock signal with jitter attenuated.

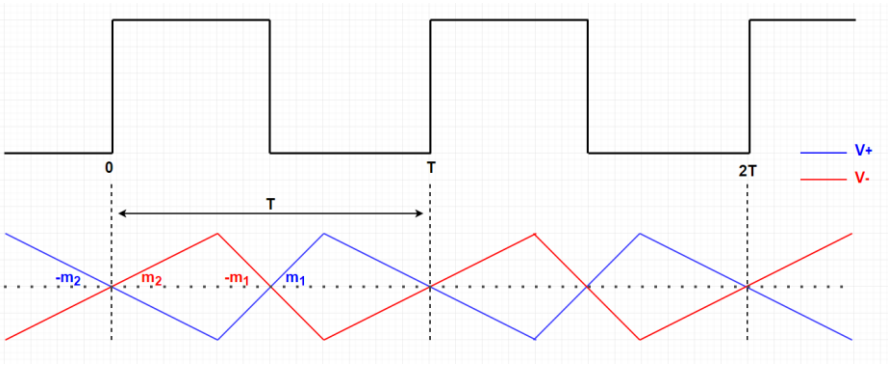

Figure 6.1 Ideal square-wave reforming output

Figure 6.1 shows an example of the ideal square-wave reformed output which has its rising edges aligned with the $\pm m_2$ crossing points on integrator's differential triangular outputs. One important fact used for the crossing detection is that as soon as the $\pm m_2$ slope legs cross each other, V_+ becomes less than V-. Therefore $\pm m_2$ and $\pm m_1$ crossing point detection could be translated to a task of comparing V_+ and V- . As soon as V_+ goes below V- a rising edge is triggered, and the falling edge occurs when V_+ becomes less than V-. Because of this, as jitter occurs, the width of the output square wave will not have a constant duty cycle. A voltage comparator is the best candidate and simply compares V- to V_+ and outputs logic high while V- is greater than V_+.

A comparator is usually implemented by high-gain amplification and what will be used here. Because of the small swing on the output of the integrator, MCML is an appropriate way to implement the comparator. MCML does not swing from rail to rail so an additional stage is needed to convert the smaller MCML swing to a rail to rail swing.

Due to the delay of using the MCML and rail to rail block, the final reformed square-wave has delay, or, in other words, a phase shift. However the JRC system itself does not maintain phase alignment between the input clock and the output clock. Its sole job is to remove jitter.

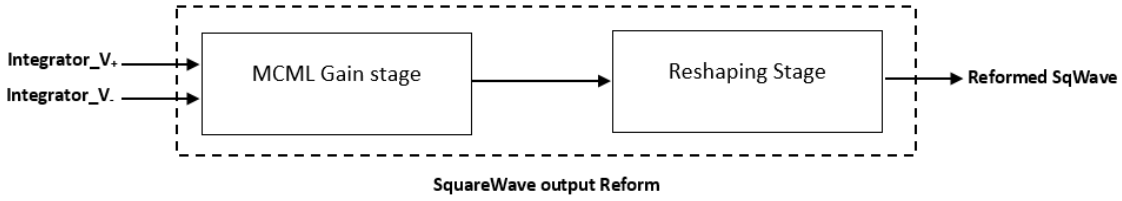

Figure 6.2 Square-wave output reform block decomposition

6.2 Square-wave output reform sub-block implementation

6.2.1 MCML gain stage (comparator)

Two identical MCML inverter (differential pair) are cascaded and used as the comparator as shown in Figure 6.3. It is true that other amplification topologies such as folded cascaded and telescopic amplifier give much higher gain, but they require more transistors being stacked together but since the integrator operates using voltage appropriate for MCML gates, an MCML solution will match the output of the integrator better. Also, with a telescopic or folded op-amp solution, the increase in stacked transistors means more components and a more complicated design, but, even more importantly, more biasing voltage drops on each stacked transistor, reducing the amplification output swing.

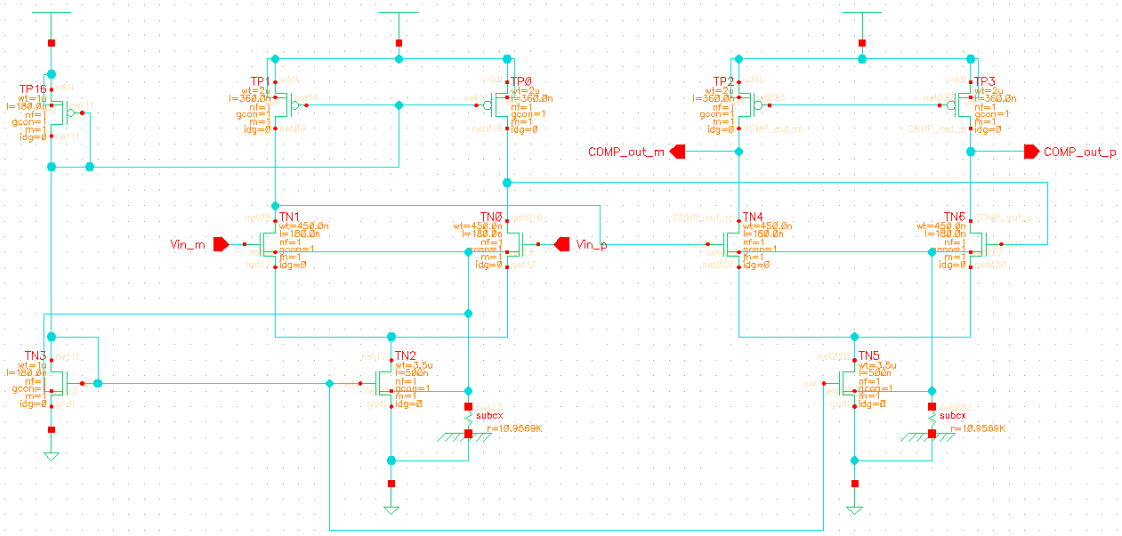

Figure 6.3 Comparator using 2 cascaded MCML inverters

The comparator compares the differential triangular waveforms V_+ and V- coming from the integrator, provides a logic high at COMP_out_p output when V- is greater than V_+ and a logic low at COMP_out_p when V- is less than V_+. Figure 6.4 shows the simulation of this comparator, with ideal 1GHz jitter-less differential integration waveforms (1ns period on the $\pm m_2$ crossing points) as inputs.

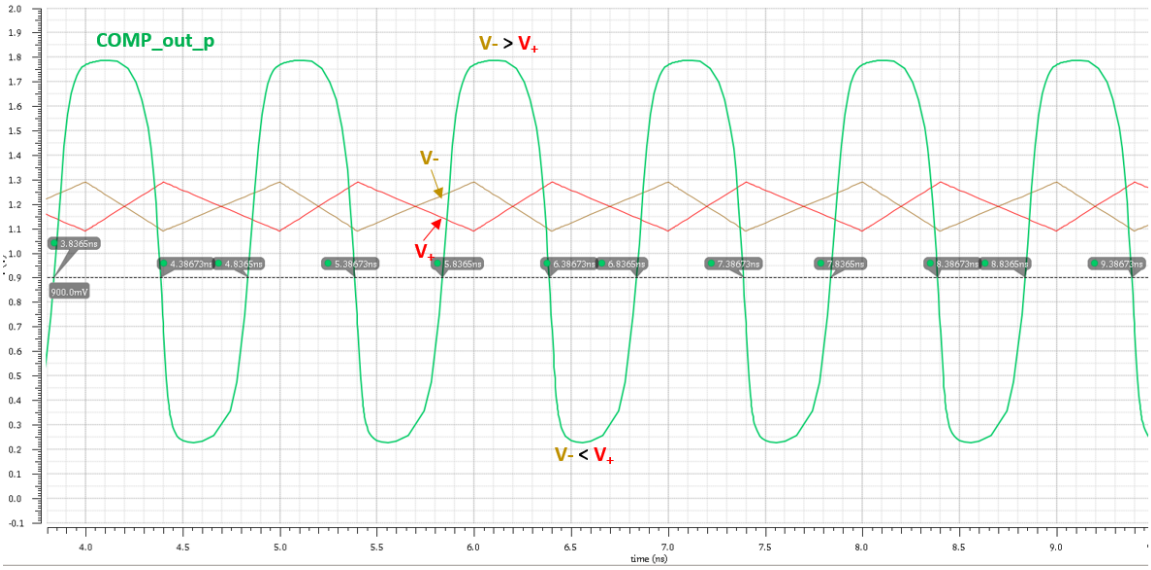

Figure 6.4 Comparator output for the ideal integration waveforms inputs

1ns period on the rising edges of the comparator output is verified from Figure 6.4, which means the comparator preserves the $\pm m_2$ crossing point time separation. In ideal case, the comparator should have instantaneous output logic level change as soon as V- goes either greater or less than V_+. But due to the limited gain from the diff pair in this implementation, the output edges have rising/falling time of about 300ps. This large rising/falling time can be reduced by adding output reshaping block which is used to sharpen the edges. In addition for applications that require rail-to-rail clock output levels, the output reshaping block can serve as the MCML swing to rail-to-rail swing converter.

6.2.2 Output reshaping stage (CMOS buffer)

A CMOS buffer in Figure 6.5 is used as the output reshaping stage, cascaded to the comparator. The reshaped output from this CMOS buffer should have sharpened rising/falling edges with rail-to-rail swing.

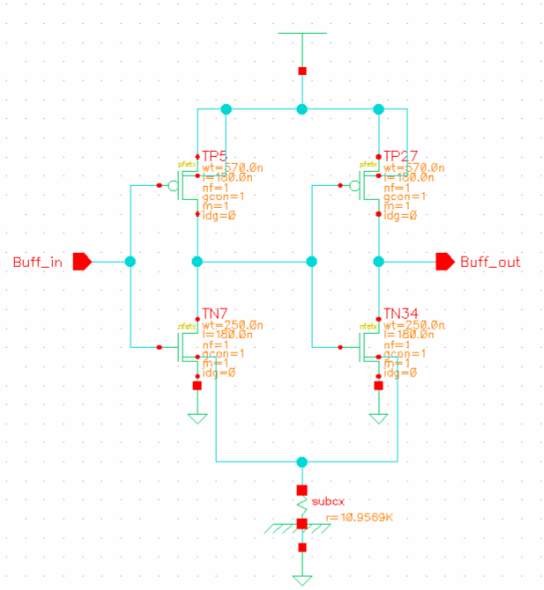

Figure 6.5 CMOS buffer as the output reshaping stage

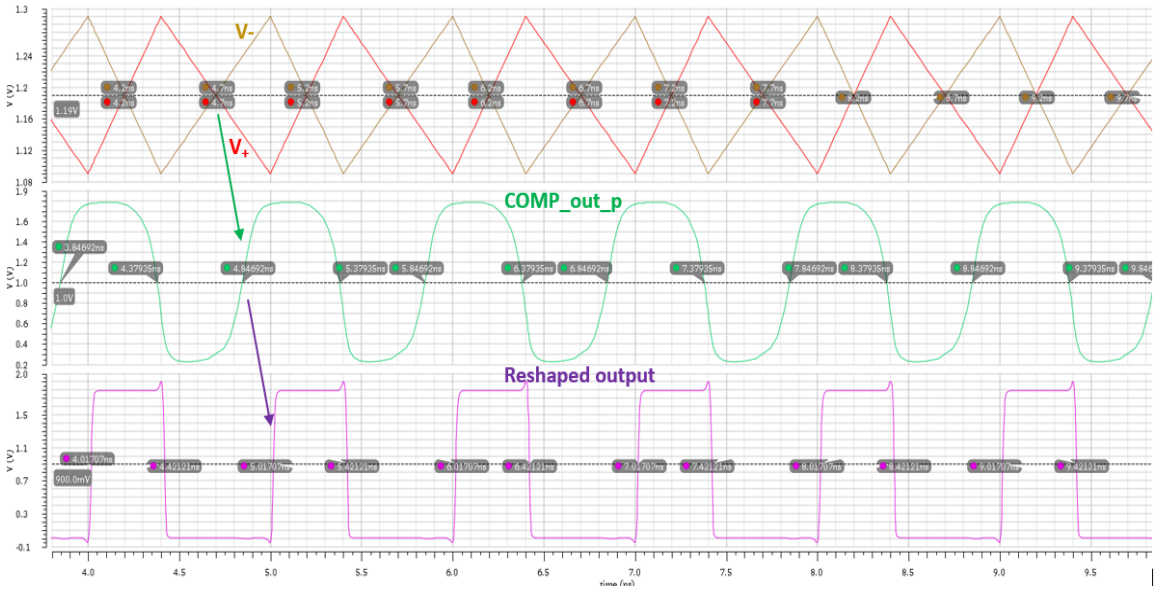

Figure 6.6 Reshaping stage output

Figure 6.6 shows the output of the reshaped comparator output by the reshaping stage. The rising time and falling time on reshaped edges are 27ps compared to the 300ps rising/falling time on comparator's output. Also note that the reshaped output swing is rail-to-rail with the output logic level being well defined (VDD for logic high output and GND for logic low output), and the period on rising edges is 1ns.

The simulation results verify that the output reform block works as desired with ideal differential integration waveforms for a jitter-less incoming clock signal. In next section, the ideal differential integration waveforms will be replaced by the actual waveforms from the integrator, in order to investigate the square-wave output reform block's performance in real cases.

6.3 Square-wave output reform block performance analysis

Nest the square-wave reform block along with the pulse generation block, the feedforward auto-biasing block and the integrator, under the case of perfect clock input and jittery input clock is investigated.

6.3.1 Jitter-less 1GHz 50% duty input clock

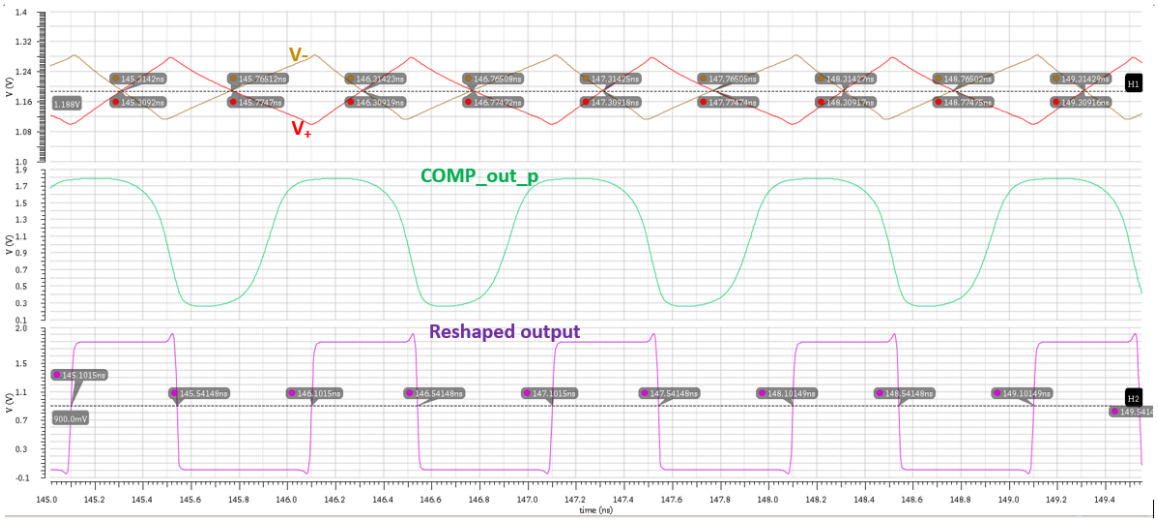

Figure 6.7 Square-wave reform block output of a jitter-less input clock

Figure 6.7 shows the simulation results of the case with a jitter-less 1GHz clock as input. As discussed in Chapter 5, the differential triangular outputs from the integrator has rounded switching corners and some curvy behaviors near the corners due to short circuit currents, non-instantaneous switching of inputs and channel-length modulation. Jitter measurement shows a 0.823ps rms period jitter on the reformed square-wave rising clock edges, and this is considered as system-contributed jitter. It is added to the attenuated jitter, to form the total jitter seen at the output of the system.

6.3.2 Jittery 1GHz clock input

The same jittery 1GHz clock with Gaussian-distributed random jitter (μ_{jitter}=0, σ_{jitter}=33ps) which was used in previous simulations is used here to test the JRC system's performance overall on jitter reduction.

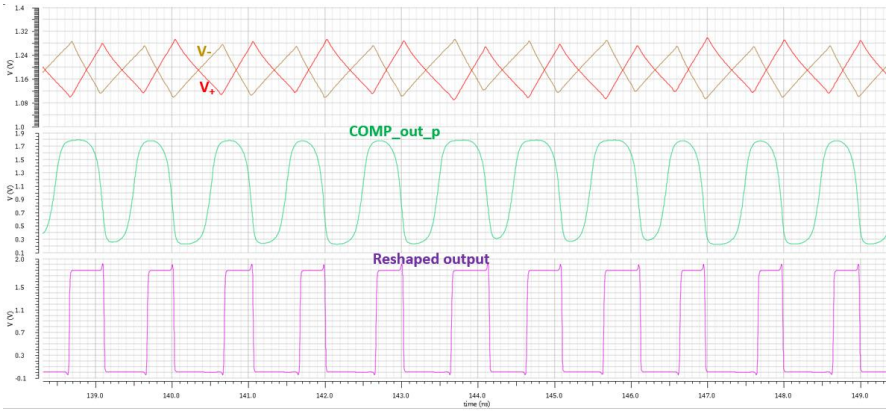

Figure 6.8 Square-wave reform block output of a jittery input clock

In Figure 6.8 we see that the duty cycle of the output recovered clock is not constant since the $\pm m_1$ crossing locations are not constant due to input jitter. This means for applications require specific clock duty cycle, one extra block that can generate a fixed pulse for each rising edge is needed.

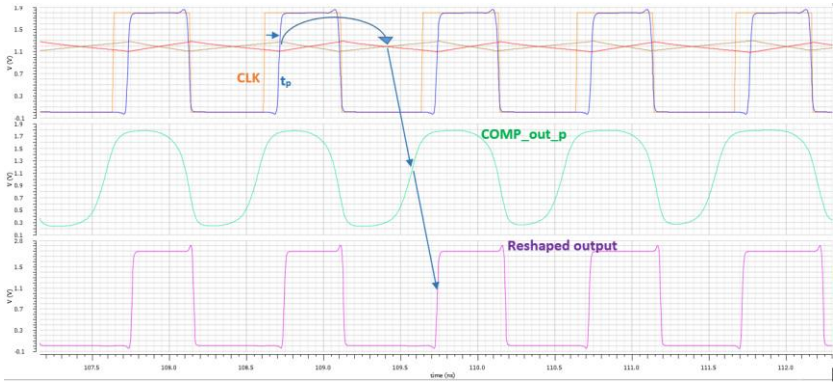

Figure 6.9 JRC propagation time

Figure 6.9 shows the delay between the input jittery clock and the output recovered clock. The delay between the jittery clock's rising edge and the constant pulse's rising edge, the time it

takes for the integrator's V_+ output reaches the $\pm m_2$ point, and the time needed to convert the crossing point to the rising edge of the reformed square-wave output (due to the delay of the comparator and the buffer) are added up to be the "propagation time" of the jitter correction. In this design, the propagation time for each clock edge is around 1ns. So when we compare the cycle to cycle jitter between the input clock and the reformed clock as shown in Figure 6.10, the corrected rising edge on the output clock should be about one period behind the input jittery rising edge. The output clock rising edge period jitter measurement reported by the simulation ranges from -31ps to +38ps, with an rms value of 12.85ps. This yields a jitter reduction of $20 \cdot \log(12.85/32.47) = -8.05$dB.

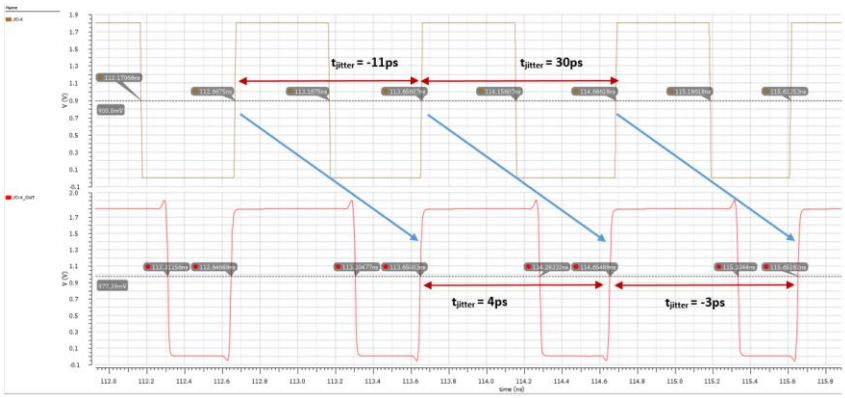

Figure 6.10 Recovered output clock period vs. jittery input clock period

	Integrator	**JRC system**
$\mathbf{t_{period\_jitter\_rms\_IN}}$	32.47ps	32.47ps
$\mathbf{t_{period\_jitter\_rms\_OUT}}$	12.76ps	12.85ps
$\mathbf{20log(t_{jitter\_OUT} / t_{jitter\_IN})}$	-8.11dB	-8.05dB

Table 6.1 JRC system attenuation performance on rms jitter

Table 6.1 compares the jitter reduction performance between the integrator alone (gathered in Chapter 5) and the system with the square-wave reforming block. We see the JRC system gives a jitter reduction of -8.05dB when the cascaded system without the reform block gives -8.11dB reduction. This means the reform block does contribute jitter on output, by (12.85 – 12.76) = 0.09ps rms.

Chapter 7

SYSTEM-LEVEL SIMULATION ANALYSIS

In addition to the simulations within the chapters of individual blocks, more simulations are done in this chapter, with JRC system-level analyses with variation on input jitter amount, environment temperature and supply noise. All simulations in this analysis are taken under the "typical-typical" corner condition, simulation under "fast-fast", "slow-slow", "fast-slow" and "slow-fast" corners will be left for future works.

7.1 Jitter reduction performance at room temperature (27℃)

Noisy clock piece-wise files used in the simulations are generated in Matlab, containing random jitter on rising edges. Recall that the input jitter range was targeted for the JRC system was 200ps peak-to-peak so multiple jittery 1GHz clocks with different jitter distribution bounced within ±100ps are examined.

$t_{jitter\_input\_rms}$	$t_{jitter\_input\_max}$	$t_{jitter\_input\_min}$	$t_{jitter\_output\_rms}$	$t_{jitter\_output\_max}$	$t_{jitter\_output\_min}$	**Jitter attenuation**
0	0	0	0.823ps	1.555ps	-1.802ps	N/A
5.07ps	18.08ps	-14.67ps	2.743ps	6.237ps	-6.506ps	-5.334dB
9.82ps	25.78ps	-28.84ps	3.596ps	8.731ps	-9.528ps	-8.726dB
16.35ps	49.47ps	-41.5ps	5.694ps	19.829ps	-20.524ps	-9.162dB
21.82ps	65.8ps	-63.46ps	8.161ps	24.325ps	-25.621ps	-8.542dB
25.41ps	66.28ps	-73.78ps	9.641ps	30.543ps	-23.759ps	-8.418dB
32.47ps	97.89ps	-96.24ps	12.850ps	40.647ps	-33.252ps	-8.052dB
39.93ps	126.34ps	-122.89ps	24.608ps	64.826ps	-59.128ps	-4.204dB

Table 7.1 Jitter attenuation with different Gaussian random jitter distribution

Table 7.1 and Figure 7.1 conclude how the JRC system respond to different Gaussian random jitter input. As seen in Table 7.1, with a perfect 1GHz clock input there is an rms value of 0.8ps of jitter that shows up at the output. As mentioned in the previous chapters, this is due to the intrinsic noise, channel-length modulation and is considered as system-contributed jitter which is

part of the total output jitter. Reduction of 5ps input jitter, as shown in the table, is about -5.36dB. Compared to the cases with higher input jitter, the lower jitter input case seems less impressive but it is explainable by noting that the system-contributed jitter takes a significant portion within the total output jitter. Otherwise the JRC maintains over -8dB attenuation for random jitter in the range within ±100ps, with its best performance at about 16ps rms input jitter. As the input jitter rms approaches its maximum value of 33ps, the attenuation is reduced since there are more values close to the allowable boundary. Also larger input jitter causes the integration waveform swing increase more which can reduce the source-drain voltage difference V_{SD} on the PMOS current sources. A reduced V_{SD} can potentially push the PMOS into its linear region. Meanwhile the increase V_{SD} can trigger channel-length modulation since the PMOS current sources do not have a large enough length L. All these influence the charging currents provided by the current sources and affect the slope of the integration waveforms.

In the case with input rms jitter of 39.93ps with some individual jitters pass the ±100ps limit, as shown in Table 7.1, a lot of timing errors due to the non-constant pulse width are added to the output as extra jitter, the attenuation drops rapidly.

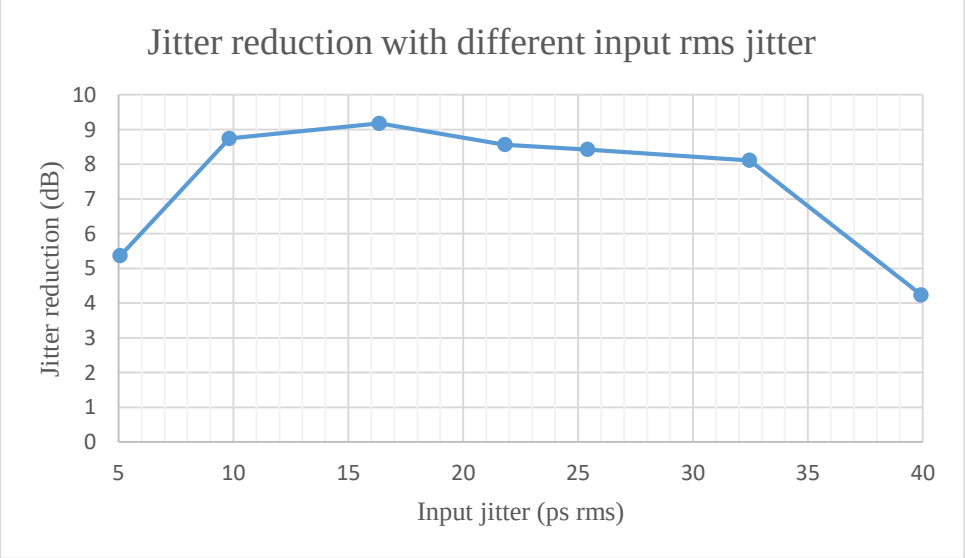

Figure 7.1 Jitter attenuation performance with different input jitter amount

7.2 Jitter reduction performance with different temperature

Thermal noise, threshold voltages and leakage current of MOSFET are temperature-dependent and a rise in operating temperature could affect JRC system's performance. Jitter reduction ratio under different temperatures is examined with the jitter range still constrained to ±100ps and an rms jitter value of 36.46ps. The results are shown in Table 7.2.

T	$t_{jitter\_input\_rms}$	$t_{jitter\_output\_rms}$	$t_{jitter\_output\_max}$	$t_{jitter\_output\_min}$	Jitter attenuation
27℃	36.46ps	14.23ps	37.75ps	-35.63ps	-8.172dB
35℃		15.12ps	39.93ps	-38.51ps	-7.645dB
40℃		14.31ps	39.88ps	-29.63ps	-8.124dB
50℃		14.71ps	40.81ps	-30.52ps	-7.884dB
60℃		14.65ps	40.29ps	-32.53ps	-7.920dB
70℃		14.87ps	40.45ps	-43.26ps	-7.790dB

Table 7.2 Jitter attenuation performance under different temperatures

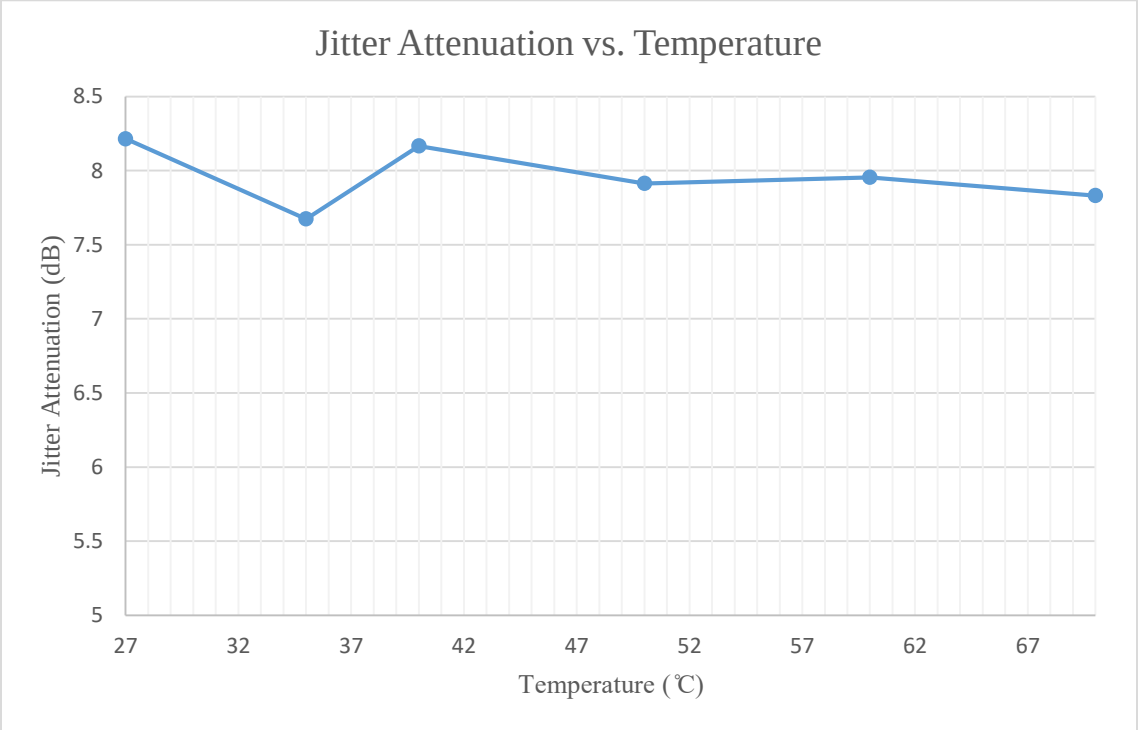

Figure 7.2 Jiiter attenuation performance under different temperatures

The output jitter attenuation is maintained at around -8dB through 27℃ to 70℃, but it generally decreases as temperature goes up to 70℃. One reason of the attenuation drop is the increased MOSFET thermal noise degrades JRC's performance by introducing more system-contributed jitter.

T	t_p	Ideal t_p' (no jitter case)	Measured t_p' (no jitter case)	Ideal $I_{charge}/I_{discharge} = t_p'/t_p$	Measured $I_{charge}/I_{discharge} = t_p'/t_p$	Current ratio error
27℃	396ps	604ps	617ps	1.53	1.56	1.96%
35℃	402ps	598ps	613ps	1.48	1.52	2.70%
40℃	407ps	593ps	611ps	1.46	1.50	2.74%
50℃	414ps	586ps	606ps	1.42	1.46	2.82%
60℃	422ps	578ps	601ps	1.37	1.42	3.65%
70℃	429ps	571ps	596ps	1.33	1.39	4.51%

Table 7.3 Constant-width pulse variation under different temperature

Table 7.3 shows the variation of the constant-width pulse t_p and its complement t_p' generated by the pulse generation block, under the changing temperature but no jitter presents in the input clock. The measured t_p width increases as temperature goes up when t_p' has it width reduce. In an ideal case the changing rate of the t_p and t_p' should be equal however the measured results show that t_p increases at about 7.5ps/10℃ when t_p' decreases with a rate of about -5ps/10℃. Therefore as temperature goes up, the deviation between the ideal value and the actual value of the current ratio $\frac{I_{charging}}{I_{discharging}} = \frac{t_p'}{t_p}$ increases, from 1.96% at 27℃ to 4.51% at 70℃. This causes error on the fundamental relationship $I_{charging} \times t_p - I_{discharing} \times (T - t_p) = 0$ increases, and the system-contributed jitter increases as a consequence.

The mismatched changing rate between t_p and t_p' also increases the overlapped turn-on time seen by the NMOS switches on the integrator. As mentioned in the integrator discussion, during the time where both switches on the integrator are on, the crucial current relation that one capacitor sees $I_{charging}$ while the other one sees $-I_{charging}$, is not valid. Distortion therefore shows up at the turning points on the triangular waves, and curvy behavior is added causing crossing point timing errors.

7.3 Jitter attenuation performance with power supply ripples

Let's say the JRC is included in a large digital CMOS logic system synchronized with the 1GHz clocking signal. For each clock cycle part of the circuitry is pulling current from the supply rail which, in fact, is dumping electrons to the supply; on the other hand, part of the circuitry is pushing current into the ground rail which in fact pumping electrons from the ground. Dumping electrons to the supply rail makes the supply less positive and pulling electrons from the ground rail makes the ground less negative with respect to the supply rail [20, 21]. Furthermore the current rushing on and off the supplies creates a magnetic field which can cause ringing on VDD and GND. Therefore we would like to see how the noisy supply rails affect JRC's performance.

Since we assume the system is switching at 1GHz with electrons being pushed and pulled from the supply and ground for some time, here we could simulate this variations with a sinusoidal ripple on the supply as shown in Figure 7.3. This makes both of the rails move from the expected level. The frequency is set to 1GHz, to match the switching behavior of the system. Even though this model is not accurate, it provides a general view showing how the JRC would be affected by supply noise.

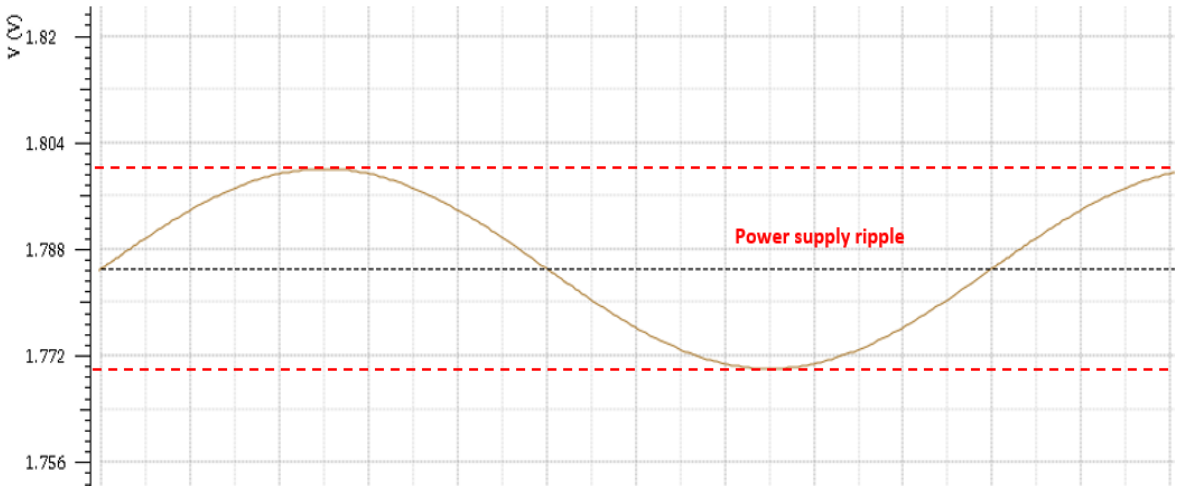

Figure 7.3 Supply ripple simulation model

Output jitter		Input jitter						
	Supply ripple	0ps	9.82ps		16.35ps		36.46ps	
	0	0.823ps	3.592ps	-8.736dB	5.692ps	-9.165dB	14.22ps	-8.178dB
	$10mV_{pp}$	1.220ps	3.711ps	-8.452dB	5.772ps	-9.044dB	15.09ps	-7.663dB
	$20mV_{pp}$	1.362ps	3.762ps	-8.334dB	5.825ps	-8.964dB	16.02ps	-7.143dB
	$30mV_{pp}$	1.392ps	3.842ps	-8.151dB	6.147ps	-8.497dB	16.06ps	-7.121dB

Table 7.4 Jitter attenuation performance under different supply ripples

Table 7.4 compares the jitter attenuation performance with different supply ripples for different input jitter amounts. For the case with a jitter-less input clock, the system-contributed jitter increases from 0.823ps to 1.392ps with a $30mV_{pp}$ VDD supply ripple (1.77V to 1.8V). For the cases with jitter presents in the input clock, the jitter reduction is also reduced. As the input jitter goes higher, the drop on the reduction ratio become more obvious as ripple increases.

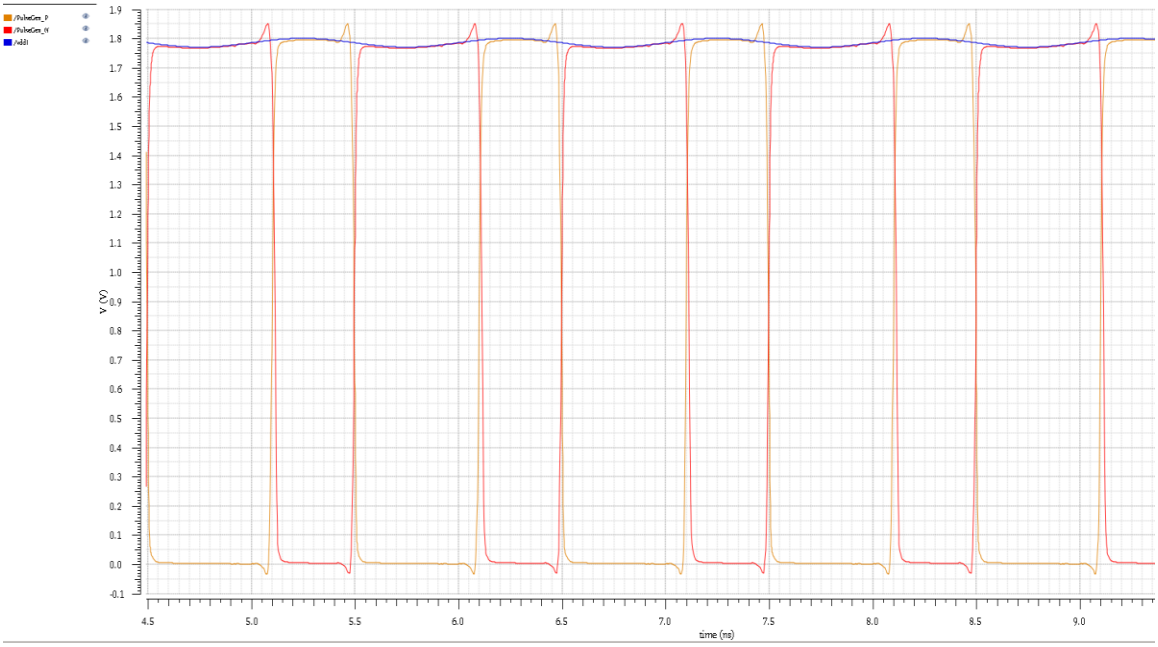

Figure 7.4 Pulse generation outputs with supply ripple presents

The stage which suffers from supply noise the most is the pulse generation block, which contains two individual CMOS inverter delay chains in this thesis. Figure 7.4 shows the differential pulses from the pulse generation block with supply noise applied. $30mV_{pp}$ supply noise causes $29.4mV_{pp}$ ripple on pulses being generated, which yields a power supply rejection ratio (PSRR) of $20 \cdot \log\left(\frac{\Delta V_{supply}}{\Delta V_{out}}\right) = 20 \cdot \log\left(\frac{30mV_{pp}}{29.4mV_{pp}}\right) = 0.18dB$. The 0.18dB PSRR implies that the noise shows up in VDD will show up on the output pulses. Since the feedforward auto-biasing block uses these pulses' average and the supply rail to set up the charging/discharging current ratio for the integrator, the ripple directly introduces errors to the currents conducted within the integrator. For a full investigation, PSRR analysis for each stage and the JRC system is needed. This topic is left for the future work.

In addition, the power supply noise affects the source-gate voltage difference of the PMOS current sources in the integrator. The PMOS current sources have their gate voltage biased by the auto-biasing block, however the source voltage is tied to VDD. A noisy VDD leads to a noise V_{SG} to the PMOS current source, which can introduce noise to $I_{charging}$ and $I_{discharging}$. Similarly, a noisy GND rail affects the V_{GS} of the NMOS current sink and can introduce sink current noise.

To improve the performance under supply ripples, large capacitor must be connected between the rails to work as a bypass capacitor and filter out the ripple. A bypass capacitor serves as a local storage of charge to supply charge when charge is pulled off of a node and to sink current when current is dumped onto a node.

Chapter 8

CONCLUSION AND FUTURE WORKS

8.1 Conclusion

The Jitter Reduction Circuit is a system which takes jittery input clocking signal as input, and produces a jitter-reduced clock output with the same frequency as the input. The advantage of this system is its simplicity compared to other jitter reduction solutions. The design developed in this work contains only four major blocks: constant pulse generation block to generate differential switching pulses which start with the input clock's rising edges and have a constant width; a feedforward auto-biasing block to set up appropriate biasing voltages required by the integrator; an integrator block which accepts the input biasing block's biasing voltages and the constant width pulse generator's pulses, and the square-wave reform block which senses the crossing point locations and coverts this timing information into square-wave-like output clock.

The design in this work is done using the IBM cmhv7sf 180nm technology, on Cadence Virtuoso tools and simulated with Spectre. The targeted jittery clock to be cleaned is a 1GHz ±10%, which means that the input clock has Gaussian random jitter ranging from ±100ps presents at the rising edges. Simulation shows the JRC has at least -8dB jitter attenuation at room temperature. Simulation with temperature at 70℃ shows a reduced jitter suppression down to -7.8dB which is predominantly due to the timing mismatches between the complementary pulses t_p and t_p'. Simulation with power supply noise shows the performance is also degraded as the ripple increases. Through simulation it was observed that the switching pulses have a common mode output level which seems to follow the ripple directly.

The constant pulse width generation block, as seen in the simulation analysis, is the block that introduces most of the errors and the errors are passed down to the cascaded blocks to undermine the overall performance. Therefore a better pulse generating block implementation is needed. This would be discussed in the future works section.

8.2 Future works

8.2.1 Fully differential delay chain

In this design, two individual CMOS inverter delay chains are used in the pulse generation block to create differential delayed copies of the input clock. The independent chains do not have a sense on how the other is functioning so if any variation or mismatch between these two chains occurs, the outputs from them are not perfectly complementary and the mismatched timing directly affects the quality of the pulse generation.

One solution is to use a MCML differential inverter chain to replace the independent CMOS inverter chain. Using an MCML inverter chain guarantees the delayed differential signals from each inverter sees any external common variation at the same time and the signals will respond in the same way. For example the variation by power supply ripple could be suppressed by the high PSRR of the differential inverter. Also the threshold variation due to temperature has the same effect on each half of the MCML inverter therefore same effect on the differential outputs.

8.2.2 Square-wave reform output duty cycle correction

The reformed square-wave output of the JRC system has the frequency of the rising edge of the output square wave match the average frequency of the input clock's, yet the duty cycle is not constant. As we have discussed before, the duty cycle of the output square-wave is determined by the timing distance of the $\pm m_2$ crossing point with respect to the $\pm m_1$ crossing point for each integrating cycle, which is varying as soon as jitter comes up at the input. To have the output duty cycle be constant, an extra block is needed. This block should take the reproduced clock output from the square-wave reform block as input, and triggers a fixed-width pulse as soon as it sees an incoming rising edge.

One candidate of such a block is the block that was used as the constant width pulse generator for the input of the JRC. However the square-wave reform output cannot guarantee each of its output has a duty cycle width greater than the delay of the delay chain which is a requirement for the constant width pulse generator used at the input of the JRC to work. Therefore another kind of "one-shot" circuitry is needed, with the pulse width determination being totally independent of the duty cycle of the triggering input. Examples of such circuits can be found in [1].

8.2.3 Additional simulations and analysis

All simulations are taken under the typical-typical process corner condition in this thesis. However to verify the robustness of the design, fabrication process variation must be taken into account and must be simulated with different corner conditions: fast-fast, slow-slow, fast-slow and slow-fast are the conditions needed to be included in future works.

The current PSRR analysis is incomplete due to time constraint, so another analysis needed to be included in the future work is a full PSRR investigation on each stage of the JRC system. After implementing the pulse generation block in a fully MCML fashion, a PSRR comparison between the current JRC system and the improved JRC can be done.

BIBLIOGRAPHY

[1] Smilkstein, Tina Harriet. “Jitter Reduction on High-Speed Clock Signals.” University of California, Berkeley, Fall, UC Berkeley EECS , 6 Aug. 2007, www2.eecs.berkeley.edu/Pubs/TechRpts/2007/EECS-2007-96.pdf.

[2] Underhill, M.j. “The Adiabatic Anti-Jitter Circuit.” *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, vol. 48, no. 3, 2001, pp. 666–674.

[3] Underhill, M.j. “The Anti Jitter Circuit for the Suppression of Wideband Phase-Noise.” *IEEE Colloquium on Microwave and Millimetre-Wave Oscillators and Mixers*, pp. 1–14,1998.

[4] Underhill, M.j. “The Noise and Suppression Transfer Functions of the Anti-Jitter Circuit.” *IEEE International Frequency Control Sympposium and PDA Exhibition Jointly with the 17th European Frequency and Time Forum, 2003. Proceedings of the 2003*.

[5] Underhill, M.j., and J. Brodrick. “The Performance of the Anti-Jitter Circuit with Enhanced Feedback ('EF-AJC').” *Proceedings of the 2004 IEEE International Frequency Control Symposium and Exposition, 2004.*

[6] Zaziabl, Adam. “A 800μW 1GHz Charge Pump Based Phase-Locked Loop in Submicron CMOS Process.” *International Journal of Electronics and Telecommunications*, vol. 56, no. 4, Jan. 2010.

[7] Mansuri, M., and Chih-Kong Ken. “Jitter Optimization Based on Phase-Locked Loop Design Parameters.” *IEEE Journal of Solid-State Circuits*, vol. 37, no. 11, 2002, pp. 1375–1382.

[8] “Jitter Attenuation-Choosing the Right Phase-Locked Loop Bandwidth.” Silicon Labs Inc., June 2010.

[9] Li, Zhiqun et al. “Design of a High Performance CMOS Charge Pump for Phase-Locked Loop Synthesizers.” *Journal of Semiconductors*, vol. 32, no. 7, 2011, p. 075007.

[10] Musa, Osman, and Maitham Shams. “An Efficient Delay Model for MOS Current-Mode Logic Automated Design and Optimization.” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 8, 2010, pp. 2041–2052.

[11] Shapiro, Alexander, and Eby Friedman. “MOS Current Mode Logic Near Threshold Circuits.” *Journal of Low Power Electronics and Applications*, vol. 4, no. 2, Nov. 2014, pp. 138–152.

[12] Heim, Marcus, and Tina H Smilkstein. “Analysis of MOS Current Mode Logic (MCML) and Implementation of MCML Standard Cell Library for Low-Noise Digital Circuit Design.” *California Polytechnic State University*.

[13] Abdulkarim, Osman Bakri Musa. “Design and Optimization of MOS Current-Mode Logic Circuits.” Carleton University, 2006, curve.carleton.ca/system/files/etd/92f8fe64-3349-454d-a99c-ca74e8b9145f/etd_pdf/1b4e0e48cbf7fa4d6ad8a5df6e56f79f/abdulkarim-designandoptimizationofmoscurrentmodelogic.pdf.

[14] Galan, Juan Antonio Gómez et al. “Low-Voltage Tunable Pseudo-Differential Transconductor with High Linearity.” *ETRI Journal*, vol. 31, no. 5, May 2009, pp. 576–584.

[15] Torralba, A. et al. “Low-Voltage Transconductor with High Linearity and Large Bandwidth.” *Electronics Letters*, vol. 38, no. 25, 2002, p. 1616.

[16] Lo, Tien-Yu, and Chung-Chih Hung. “A 1 GHz Equiripple Low-Pass Filter With a High-Speed Automatic Tuning Scheme.” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 2, 2011, pp. 175–181.

[17] Czarnul, Z. et al. “Common-Mode Feedback Circuit with Differential-Difference Amplifier.” *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 41, no. 3, 1994, pp. 243–246.

[18] Krishnapura, Nagendra. “Analog Integrated Circuit Design: Common Mode Feedback Circuits.” Apr. 2006.

[19] Tauro, A. et al. “Common Mode Stability in Fully Differential Voltage Feedback CMOS Amplifiers.” *10th IEEE International Conference on Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003*, vol. 1, 1 June 2004, pp. 288–291.

[20] Rusu, Stefan. “Clock Generation and Distribution for High-Performance Processors.” Intel Corporation, 2004.

[21] Sharama, Chetan. “Comparison between Low Power Clock Distribution Schemes in VLSI Design.” *Journal of Global Research in Computer Science*, vol. 2, no. 2, February 2011.

[22] Mansuri, Mozhgan. “Low-Power Low-Jitter on-Chip Clock Generation.” *University of California Los Angeles*, 2003.

[23] “Understanding and Characterizing Timing Jitter.” Tektronix, Inc., 2012.

[24] “Understanding Jitter Units, Application Note-815 Rev A.” Integrated Device Technology, Inc, 2014.

[25] Oliver, John. “EE523_Lecture2:CMOS Inverter.” California Polytechnic State University, 2015.

[26] Gray, Paul R. *Analysis and Design of Analog Integrated Circuits*. New York, Wiley, 2009.

[27] Jaeger, Richard C., and Travis N. Blalock. *Microelectronic Circuit Design*. Boston, McGraw-Hill Higher Education, 2008.

[28] Wolpert, David, and Paul Ampadu. *Managing Temperature Effects in Nanoscale Adaptive Systems*. New York, Springer, 2012, pp.15-33.

APPENDIX A JITTERY 1GHz CLOCK PWL FILE GENERATION

```
% Jittery 1GHz Clock PWL generation Matlab script
%
% Run Bin Yu
% Cal Poly at San Luis Obispo
% 2016
%
% +/- 100ps random jitter appears at the rising clock edges
% but pulse width remains 500ps

close all
clear all

% Gaussian distribution parameters & # of clock cycles
JITTER_U=0;
JITTER_SIGMA=33e-12;
CLOCK_CYCLE=350;

% Initialization
T_NoJitter=[0 10e-12 500e-12 510e-12 1000e-12]';
voltage=[0 1.8 1.8 0 0]';
t_previousCycle=T_NoJitter;
T=T_NoJitter(1:4);
V=voltage(1:4);
T_ideal=T_NoJitter(1:4);
Jitter_array=[0];

% Radom jitter generation iteration loop
for i=1:CLOCK_CYCLE
   jitter=normrnd(JITTER_U,JITTER_SIGMA);
   Jitter_array=vertcat(Jitter_array,jitter);
   t_currentCycle=t_previousCycle(5)+jitter+T_NoJitter;
   T=vertcat(T,t_currentCycle(1:4));
   V=vertcat(V,voltage(1:4));
   t_previousCycle=t_currentCycle;
   T_ideal=vertcat(T_ideal,T_ideal(1:4)+i*T_NoJitter(5));
end

% Result display
plot(T,V,'r')
hold on
plot(T_ideal,V,'b--')
T_V=horzcat(T,V);
figure
hist(Jitter_array,100)
xlabel('Jitter Bin in seconds');
ylabel('# of jitters within the bin');
disp(['Jitter_rms = ',num2str(rms(Jitter_array)),'s'])
disp(['Jitter_max = ',num2str(max(Jitter_array)),'s'])
disp(['Jitter_min = ',num2str(min(Jitter_array)),'s'])
```

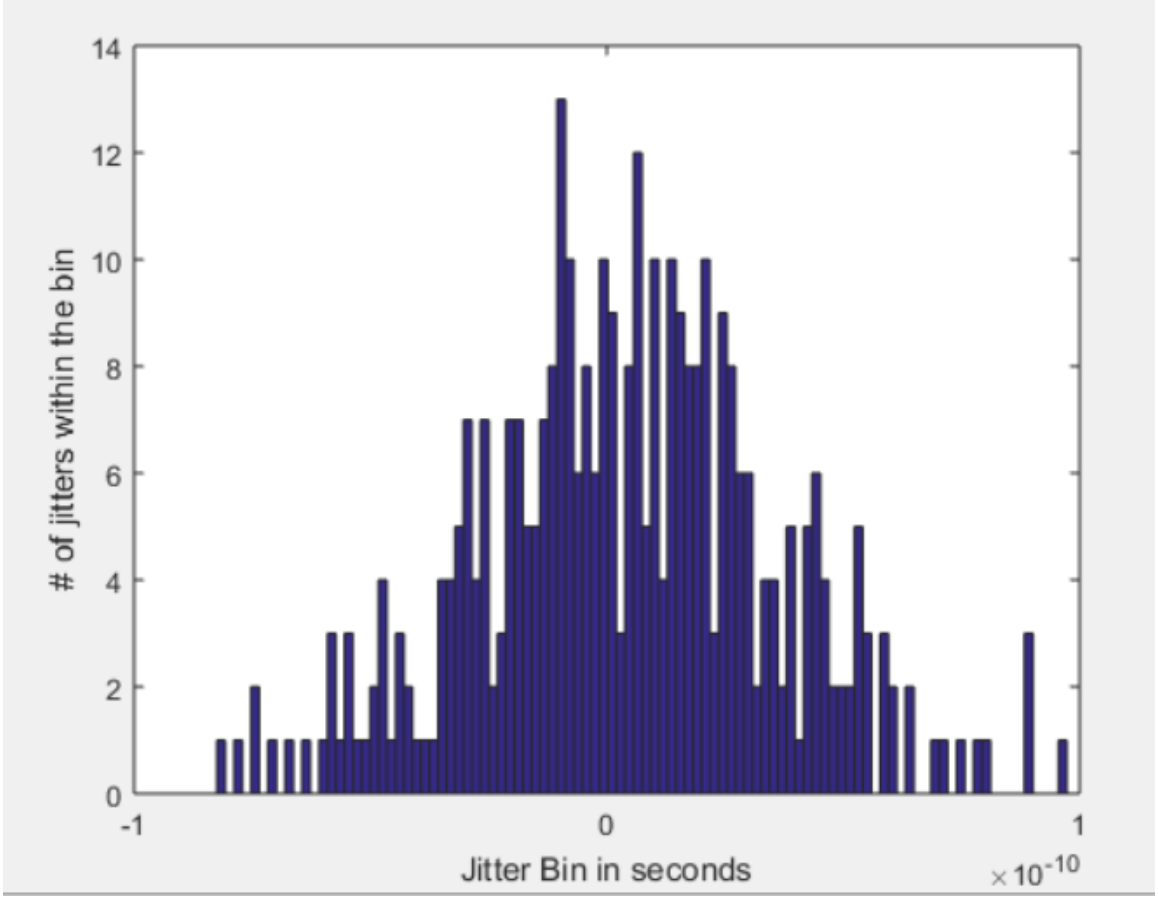

Figure A.1 Gaussian Random Jitter Distribution Histogram example

```
Jitter_rms = 3.0742e-11s
Jitter_max = 9.5292e-11s
Jitter_min = -8.1038e-11s
```

Figure A.2 Gaussian random jitter parameters