PLL Phase Noise & Jitter Modeling, for High Speed Serial Links

Klodjan Bidaj

université de BORDEAUX

THÈSE PRÉSENTÉE

POUR OBTENIR LE GRADE DE

DOCTEUR DE

L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE

SPÉCIALITÉ ELECTRONIQUE

Par Klodjan BIDAJ

Modélisation du bruit de phase et de la gigue d'une PLL, pour les liens séries haut débit.

Sous la direction de : Jean-Baptiste BEGUERET, Laboratoire IMS, Bordeaux
Co-directeur : Jerome DEROO, STMicroelectronics, Grenoble

Soutenue le 30/11/2016

Membres du jury :

M.	DALLET, Dominique	Professeur	IPB	Examinateur
M.	MIR, Salvador	Directeur de recherche CNRS	TIMA	Rapporteur
M.	MAZZANTI, Andrea	Professeur	Univ. de Pavia	Rapporteur
Mme.	AZAIS, Florence	Chargé de recherche CNRS	LIRMM	Examinateur
M.	BEGUERET, JB	Professeur	Univ. de Bordeaux	Directeur de Thèse
M.	DEROO, Jerome	Ingénieur	STMicroelectronics	Encadrant industriel
M.	THIES, William	Ingénieur	STMicroelectronics	Invité

Titre : Modélisation du bruit de phase et de la gigue d'une PLL, pour les liens séries haut débit.

Résumé :

La vitesse des liens séries haut débit (USB, SATA, PCI-express, etc.) a atteint les multi-gigabits par seconde, et continue à augmenter. Deux des principaux paramètres électriques utilisés pour caractériser les performances des SerDes sont la gigue transmis à un niveau de taux d'erreur donné et la capacité du récepteur à suivre la gigue à un taux d'erreur donné.

Modéliser le bruit de phase des différents components du SerDes, et extraire la gigue temporelle pour la décomposer, aideraient les ingénieurs en conception de circuits à atteindre les meilleurs résultats pour les futures versions des SerDes. Générer des patterns de gigue synthétiques de bruits blancs ou colorés permettrait de mieux analyser les effets de la gigue dans le système pendant la phase de vérification.

La boucle d'asservissement de phase est un des contributeurs de la gigue d'horloge aléatoire et déterministe à l'intérieur du système. Cette thèse présente une méthode pour modéliser la boucle d'asservissement de phase avec injection du bruit de phase et estimation de la gigue temporelle. Un modèle dans le domaine temporel en incluant les effets de non-linéarité de la boucle a été créé pour estimer cette gigue. Une nouvelle méthode pour générer des patterns synthétiques de gigue avec une distribution Gaussienne à partir de profils de bruit de phase coloré a été proposée.

Les standards spécifient des budgets séparés de gigue aléatoire et déterministe. Pour décomposer la gigue de la sortie de la boucle d'asservissement de phase (ou la gigue généré par la méthode présentée), une nouvelle technique pour analyser et décomposer la gigue a été proposée. Les résultats de modélisation corrèlent bien avec les mesures et cette technique aidera les ingénieurs de conception à identifier et quantifier proprement les sources de la gigue ainsi que leurs impacts dans les systèmes SerDes.

Nous avons développé une méthode, pour spécifier la boucle d'asservissement de phase en termes de bruit de phase. Cette méthode est applicable à tout standard (USB, SATA, PCIe, …) et définit les profils de bruits de

phases pour les différentes parties de la boucle d'asservissement de phase, pour s'assurer que les requis des standards sont satisfaits en termes de gigue. Ces modèles nous ont également permis de générer les spécifications de la PLL, pour des standards différents.

Mots clés : bruit de phase, gigue, taux d'erreur, décomposition de la gigue, liens séries haut débit, SerDes, analyse temps-fréquence, boucle d'asservissement de phase, gigue aléatoire, gigue déterministe, génération de bruit coloré, génération de bruit blanc

Title : PLL Phase Noise & Jitter Modeling, for High Speed Serial Links

Abstract :

Bit rates of high speed serial links (USB, SATA, PCI-express, etc.) have reached the multi-gigabits per second, and continue to increase. Two of the major electrical parameters used to characterize SerDes Integrated Circuit performance are the transmitted jitter at a given bit error rate (BER) and the receiver capacity to track jitter at a given BER.

Modeling the phase noise of the different SerDes components, extracting the time jitter and decomposing it, would help designers to achieve desired Figure of Merit (FoM) for future SerDes versions. Generating white and colored noise synthetic jitter patterns would allow to better analyze the effect of jitter in a system for design verification.

The phase locked loop (PLL) is one of the contributors of clock random and periodic jitter inside the system. This thesis presents a method for modeling the PLL with phase noise injection and estimating the time domain jitter. A time domain model including PLL loop nonlinearities is created in order to estimate jitter. A novel method for generating Gaussian distribution synthetic jitter patterns from colored noise profiles is also proposed.

The Standard Organizations specify random and deterministic jitter budgets. In order to decompose the PLL output jitter (or the generated jitter from the proposed method), a new technique for jitter analysis and decomposition is proposed. Modeling simulation results correlate well with measurements and this technique will help designers to properly identify and quantify the sources of deterministic jitter and their impact on the SerDes system.

We have developed a method, for specifying PLLs in terms of Phase Noise. This method works for any standard (USB, SATA, PCIe, …), and defines Phase noise profiles of the different parts of the PLL, in order to be sure that the standard requirements are satisfied in terms of Jitter.

Keywords : Phase Noise, Jitter, BER, jitter decomposition, high speed serial links, SerDes, time-frequency analysis, PLL, RJ, DJ, colored noise generation, white noise generation

Table of Contents

Chapter 1: High Speed Serial Links

Table of Figures

1.1 Introduction

Integrated Circuits have become essential part of our lives in the last decades. They are found in our daily life applications. Their range of applications is very large (such as smartphones, tablets, computers, cars, TVs, screens, cameras, connected objects, electronic gadgets, etc.) and is growing daily. This increasing number of applications and the all-time increasing quality of service requested by users, results in more components complexity and much more data transferred between them.

In order to make the connection between different devices easier by replacing a multitude of connectors, and simplify software configuration of all devices, the Universal Serial Bus (USB) was developed in the early 1990's. The USB [USB] was designed to standardize the connection of computer peripherals to computers. Since then, it has been used in a lot of other devices (such as smartphones, tablets, video game consoles, etc.).

The reason for creating high speed serial links is to be able to transfer much more data within the same timeframe, and get better quality. For example, as shown in Fig 1-1, the transfer rate necessary in 1990's of 147 Mbps has been increased by 40 times within 20 years, reaching 5.9 Gbps in 2010's, in order to obtain todays FULL HD quality on our PC screens. [Yonsei]

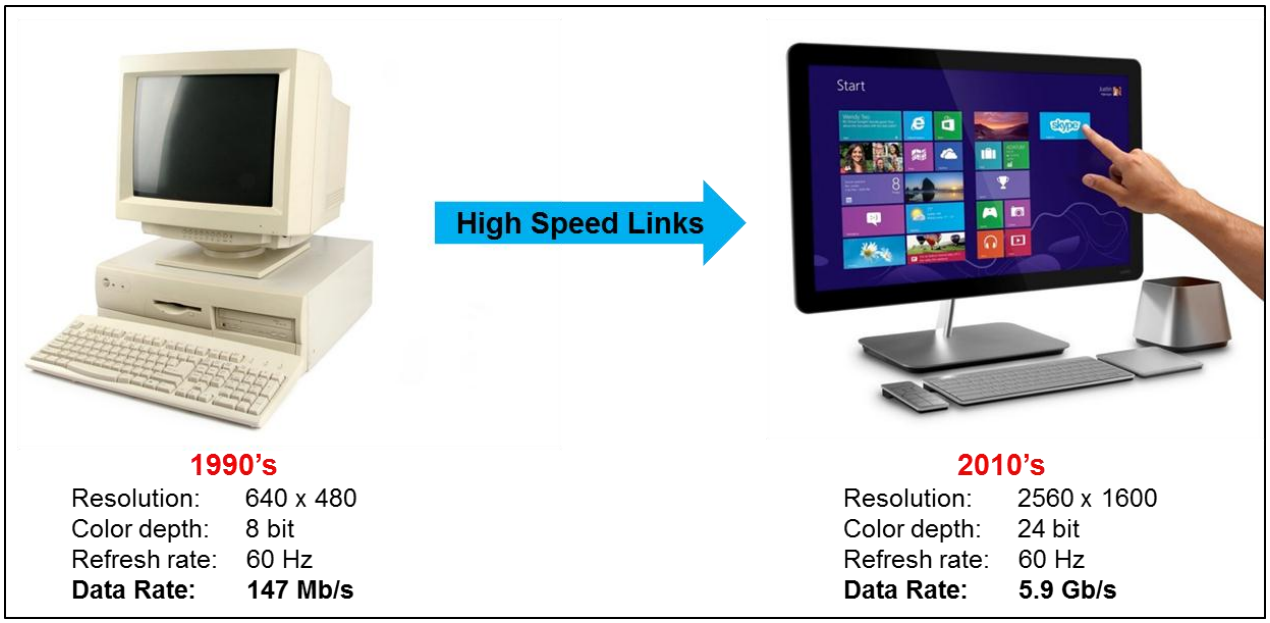


Figure 1-1: Why High Speed Serial Links?

Therefore, the serial communication techniques transferring data within devices (or interconnecting several devices together) have been revolutionized in the last decades to respond the continuously increasing market demand [Hong] - [Kuo].

High Speed Serial Links (HSSLs) are today the “dominant” [Stauffer] communication links of Input/Output for many chips, and reach currently speeds up to tens of Gbps [Hanumolu] – [Ong].

1.2 Applications of High Speed Serial Links

HSSLs are used in most mobile devices (such as smartphones or tablets) to interconnect different components inside them. For example they connect the processor to the camera, to the storage device, to the modem, to another chip, etc… This kind of communications uses channels’ length of up to 10 cm.

HSSLs are also used inside laptops, computers, TVs, screens, etc. They connect board-to-board chips, such as processor-to-memory, or processor-to-processor communications. This kind of communications uses channels up to 50 cm long.

Another utility of HSSLs is to connect different devices together, for example processor-to-peripheral or processor-to-storage devices. Some of the most known HSSLs for these kinds of connections are PCIe, USB, SATA, etc. This kind of communications uses channels up to 3 m long.

In addition to these applications, HSSLs are also used in network communications (such as LAN: Ethernet), which might use channels up to 100 meters long.

As each of the above applications has different requirements, a variety of HSSLs with different protocols are designed to fulfill their needs.

1.3 Challenges of High Speed Serial Links

Two of the main challenges of HSSLs are increasing the data rates and reducing the power consumption.

As shown in Fig 1-2, the data centers have a real challenge looking for about 80% of transfer rate growth each year.

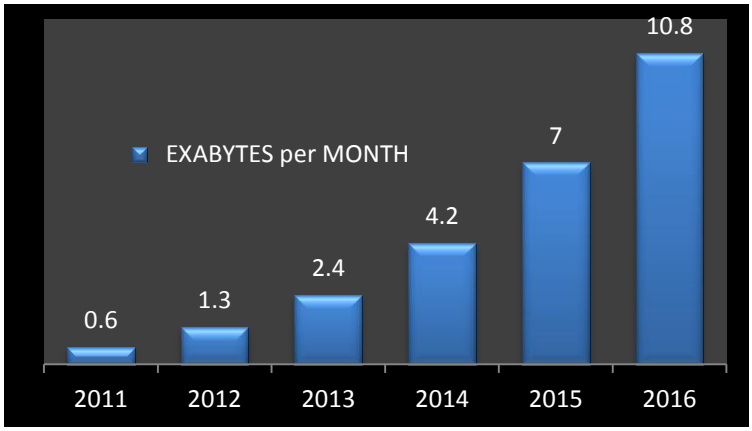


Figure 1-2: Worldwide Data Transferring Traffic - Cisco Visual Networking Index

To answer to the permanent increasing request of bandwidth, Fig 1-3 shows the different standards that double their bandwidths every 4 years. Furthermore, we remark an average period of 4 years between technologies used in industrial market (FC - Fiber Channel), and the standards developed for consumer market (PCI-Express, USB, DP).

Consequently, the expertise capitalized during the design of High Speed serial links might be reused for the design of HSSLs dedicated for consumer market.

As shown in Fig 1-3, the data rates have increased from 480 Mbps to 16 Gbps within the last 15 years.

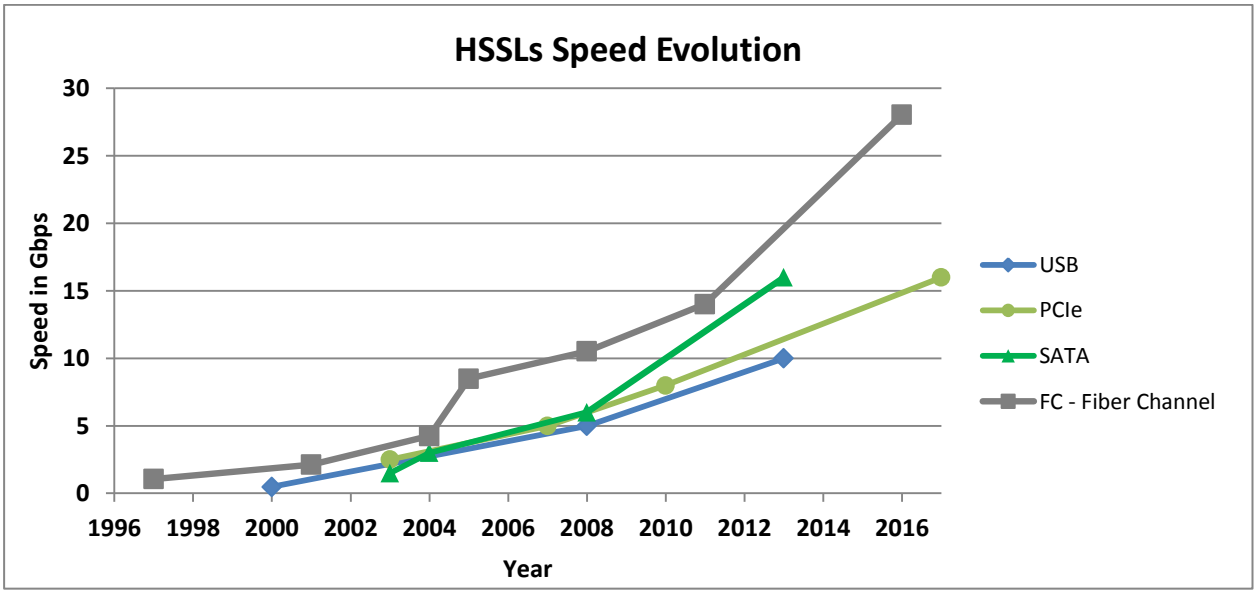


Figure 1-3: HSSLs Speed Evolution over years

Decreasing power consumption has been staying a priority for decades on all integrated circuits. Fig 1-4 shows STMicroelectronics HSSLs circuits IP performances in terms of power consumption in the last 10 years:

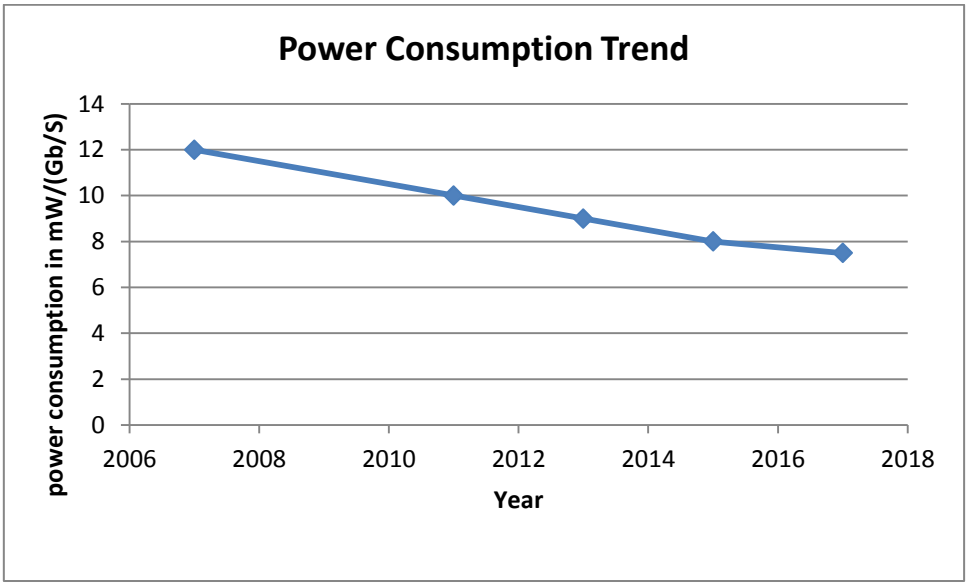


Figure 1-4: Power consumption evolution of STM HSSL circuits.

It is a big challenge to find out the compromise of how much we can decrease the power consumption of each block of the IP without degrading global performances of the system.

1.4 Jitter Role in High Speed Serial Links

Jitter is defined as the phase variation of a timing signal from its ideal positions in time, which corresponds to an early or late transition, as shown in Fig 1-5 [Dou_1] - [HFE] - [Kuo] - [LeCroy] - [Sui] - [Wisetphanichkij].

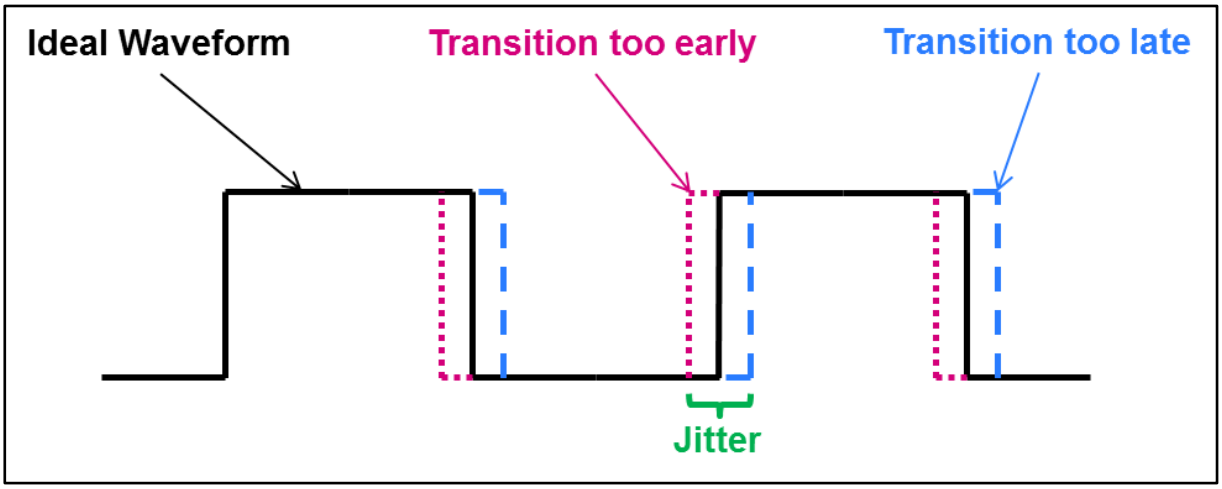


Figure 1-5: Jitter Definition

The amount of jitter on a transmission is related to the BER (Bit Error Rate). If jitter is too high, it can cause transmission errors [Zhang]. The receiving circuit will interpret the bit differently from the transmitted one, as shown in Fig 1-6 [HFE]. Hence, the global transmission performance is degraded.

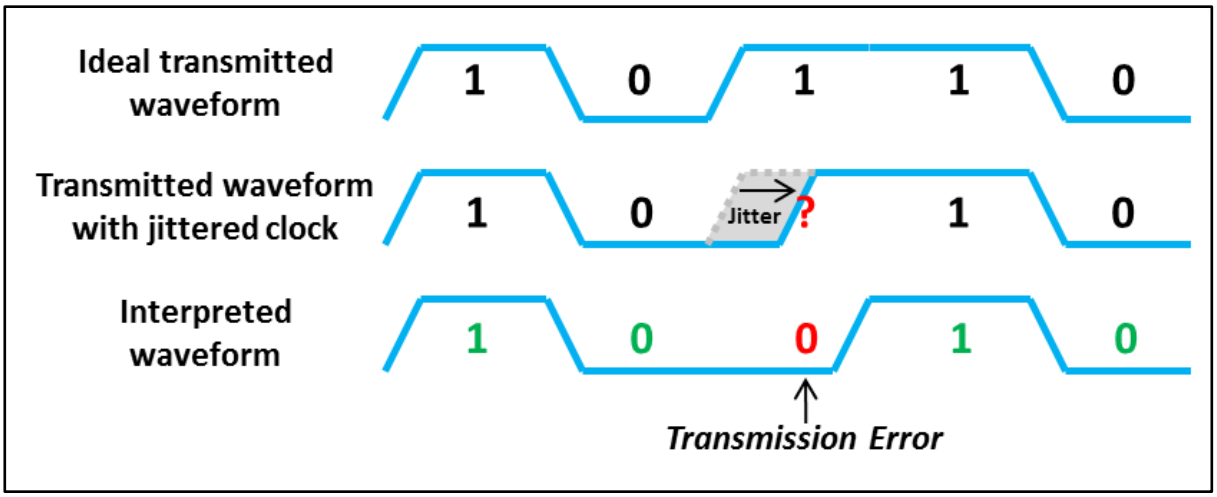


Figure 1-6: Transmission Error due to Jitter

With the growing speeds of HSSLs, it becomes very difficult to maintain high signal quality. The absolute tolerated jitter quantity gets smaller [Tripathi], and the noise level becomes much more sensitive. High Speed Serial Links standards require an expected error rate (BER) of 10^{-12} [LeCroy]. While this represents only 1 bit error every 167 minutes at 100 Mbps, it translates to 1 bit error every ~3 minutes at 5 Gbps and to 1 bit error every 1 minute at 16 Gbps. This is why the capacity to determine and limit jitter is one of the most important elements in maintaining high signal quality in HSSL transmissions [Sharma].

1.5 High Speed Serial Links System Study

In Fig 1-7 we find the principal components of a High Speed Serial Links System. [Hanumolu] - [Palermo] - [Yonsei]

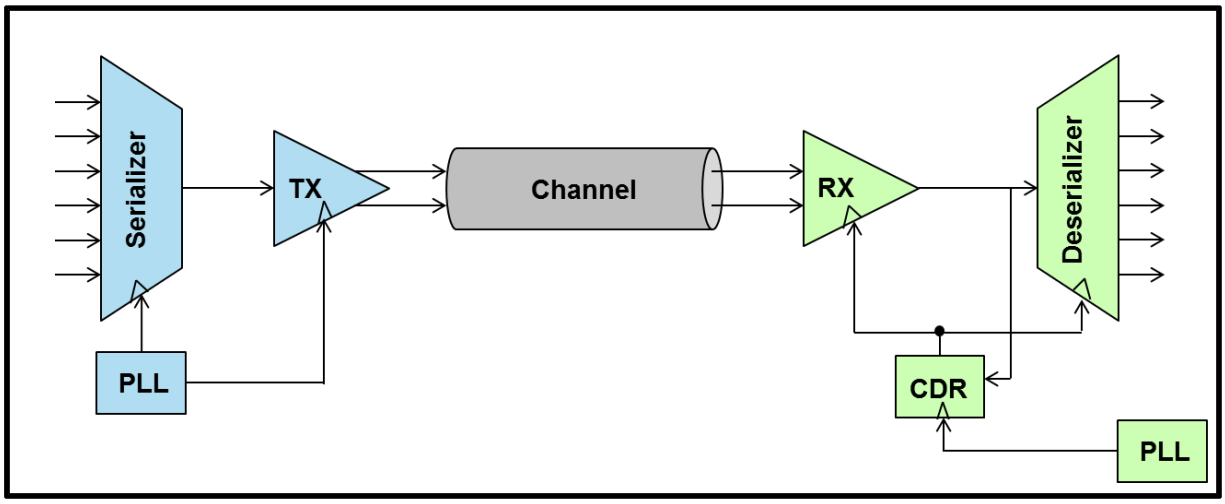


Figure 1-7: High Speed Serial Links System

On the left side of the picture, the "**Serializer**" converts the incoming parallel data into series ones. It serializes data from 10 or 20 bits into 1 bit.

The "**PLL**" generates the different clocks needed for the Transmission and the Reception. ***It is one of the main contributors for clock Random and Periodic jitter***. [Hansel] - [Telba]

The "**TX**" enables the transmission. It takes the digital stream and transforms it to electrical signal (First stage of Signal Amplitude Pre-Equalization is done here). This block *contributes to clock jitter*. It is *the first contributor of ISI jitter* (circuit non linearity).

On the right side of the figure, the receiver block "**RX**" decodes the received signal, and transforms electrical signal into digital stream. This block *contributes to clock jitter* (clock recovery algorithm + analog sampling clock source). Second stage of Signal Post-Equalization is done here: CTLE + DFE (circuit non linearity).

The "**CDR**" (Clock and Data Recovery) is responsible for locking the receiver clock to the transmitter clock by correcting the phase and frequency offsets, and recovering the transmitted data.

The "**Deserializer**" converts the data from series to parallel. It de-serializes data from 1 bit to 10 or 20 bits.

Between the “TX” and the “RX” the “**channel**” has different properties, depending on the transmission medium (Backpannel, Ports, Cables …).

1.6 Motivation of this thesis

Two of the major electrical parameters used to characterize SerDes Integrated Circuit performances are the Transmitter & Receiver clocks jitter at a given BER and the Insertion Loss capabilities.

Modeling the phase noise of the different SerDes components, extracting the time jitter and decomposing it, would help designers to achieve desired FoM for future SerDes versions.

As mentioned in 1.5, the PLL is the main contributor of clock random and periodic jitter inside the HSSLs system. This is the reason why we decided to create a PLL Model with phase noise injection in order to estimate the time domain jitter. To better estimate the time jitter by including the PLL loop nonlinearities, a time domain model is also created.

The Standard Organizations (such as USB or PCIe) specify random and deterministic jitter budgets. In order to decompose the PLL output estimated TIE jitter into random and deterministic jitter, a new technique for jitter analysis and decomposition is proposed.

Modeling simulation results correlate well with measurements and this technique will help designers to properly identify and quantify the sources of deterministic jitter and their impact on the SerDes system.

Furthermore, these models permit us to determine System Specifications for a new IP development, by determining maximum phase noise profile to be respected by designers for each sub-block, in order to satisfy standards in term of Random and Deterministic Jitter.

1.7 Thesis Organization

This thesis is organized as follows:

Chapter 2, "*Jitter and Phase Noise Theoretical Study*" presents the general theoretical study of jitter. The complete study for calculating Jitter in the Time and Frequency domain is detailed here.

Chapter 3, "*PLL Modeling*" is divided into two main sections; the first one presents complete PLL Modeling in the Frequency Domain. The second section presents the chart flow, and the Time Domain PLL Modeling including nonlinearities. Result comparison is done between both models in order to validate the Time Domain Model.

Chapter 4, "Generation of colored noise patterns with Gaussian Jitter distribution" describes a novel method for generating colored noise patterns with Gaussian distributions. PLL has Colored Noise profile properties. In order to analyze SerDes system characteristics, it is important to generate synthetic colored noise patterns with a Gaussian distribution. The patterns will be used to predict impact of jitter on the system performance with time domain simulation during the design verification phase.

Chapter 5, "*RJ/DJ Jitter Decomposition Technique*", presents a new technique for decomposing Random from Deterministic Jitter. Results of Simulations and Measurements are given to verify that this technique works properly.

Chapter 6, "PLL *Specifications*", gives detailed example of specifications in term of Phase Noise and Jitter for PLL. The specifications are given as example based on PCI-Express Standard requirements, using our developed models.

Chapter 7, "*Conclusions and Perspectives*", summarizes the work presented in this thesis and gives future perspectives on algorithm improvements.

Chapter 2: Jitter and Phase Noise Theoretical Study

Table of Figures

Table of Tables

2.1 Introduction to Jitter

Jitter is defined as the phase variation of a timing signal from its ideal positions in time, as shown in Fig 2-1 [Dou_1] - [HFE] - [Kuo] - [LeCroy] - [Sui] - [Zamek] - [Wisetphanichkij].

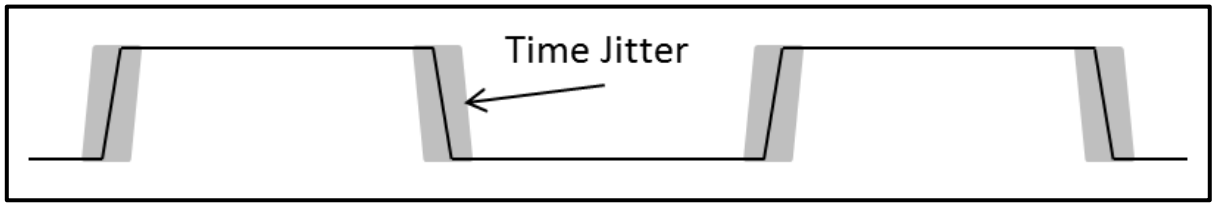


Figure 2-1: Time Jitter

It can be viewed as different forms:

The cumulative jitter corresponds to the difference between the n^{th} measured edge instant 't_n' and the ideal '**nT**' where T corresponds to the ideal signal period [HFE] - [Kuo]. This is the time interval error (TIE), shown in Fig 2-2 and expressed as (2.1):

$$\Phi n = t_n - nT \tag{2.1}$$

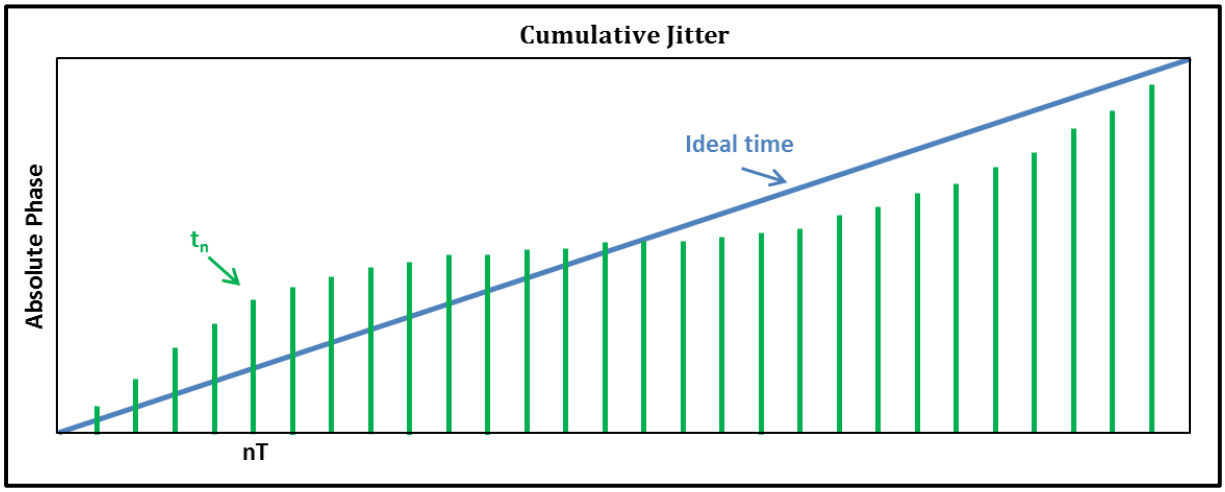


Figure 2-2: Cumulative Jitter

The periodic jitter corresponds to the difference between the measured period and the ideal period. It is shown in Fig 2-3 and expressed as (2.2) [Kuo].

$$\Phi_n{}' = \Phi_n - \Phi_{n-1} \tag{2.2}$$

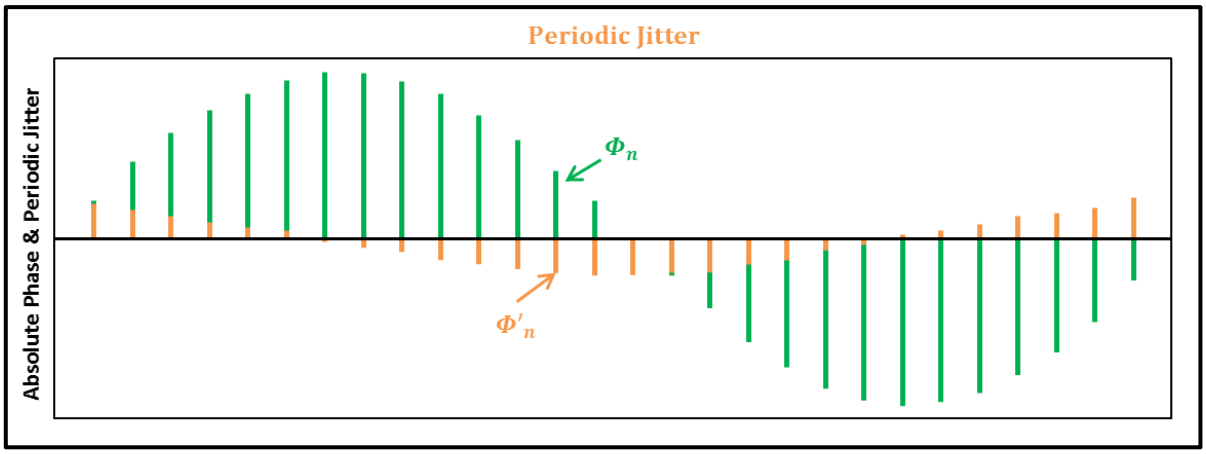


Figure 2-3: Periodic Jitter

The cycle to cycle jitter corresponds to the difference between consecutive bit periods. It is shown in Fig 2-4 and expressed as (2.3) [Kuo].

$$\Phi_n'' = \Phi_n' - \Phi_{n-1}' \quad (2.3)$$

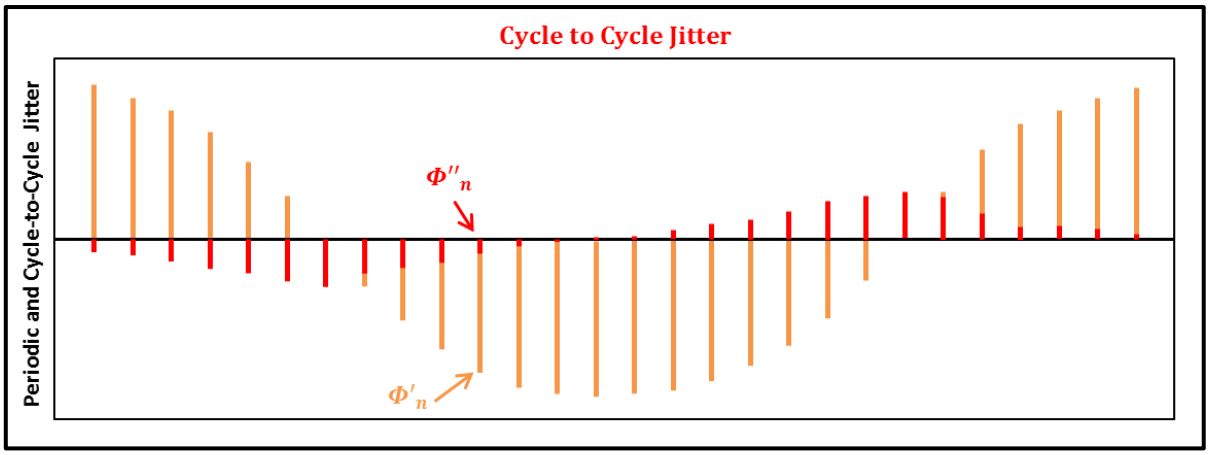


Figure 2-4: Cycle to Cycle Jitter

In this thesis we will study the cumulative TIE jitter. It consists of Random Jitter (RJ), which is unpredictable, unbounded timing noise, and Deterministic Jitter (DJ), which is uncorrelated to data and bounded [Dou_1] - [Hansel] - [Kim] - [Kuo] - [Li] - [Sharma] - [Sui] - [Wisetphanichkij]. The PCI-express standard [PCIe] gives the decomposition of jitter as follow:

- Data Uncorrelated Jitter
  - Unbounded Jitter
    - Random Jitter (T_{TX-RJ})
    - Total Jitter (T_{TX-UTJ})
  - Bounded
    - Deterministic Jitter ($T_{TX-UDJDD}$)

Equation (2.4) describes the total jitter as a function of BER and is related to the RJ and DJ components of jitter. [LeCroy] - [Tektronix]

$$TJ = N(BER) * RJ + DJ \tag{2.4}$$

The Bit Error Rate is the measurement of data reliability; it corresponds to the probability of doing an error while determining a bit level. Another view of jitter is provided by the bathtub plot. It is a graph of BER in log scale versus jitter as shown in Fig 2-5 [HFE] - [Zhang].

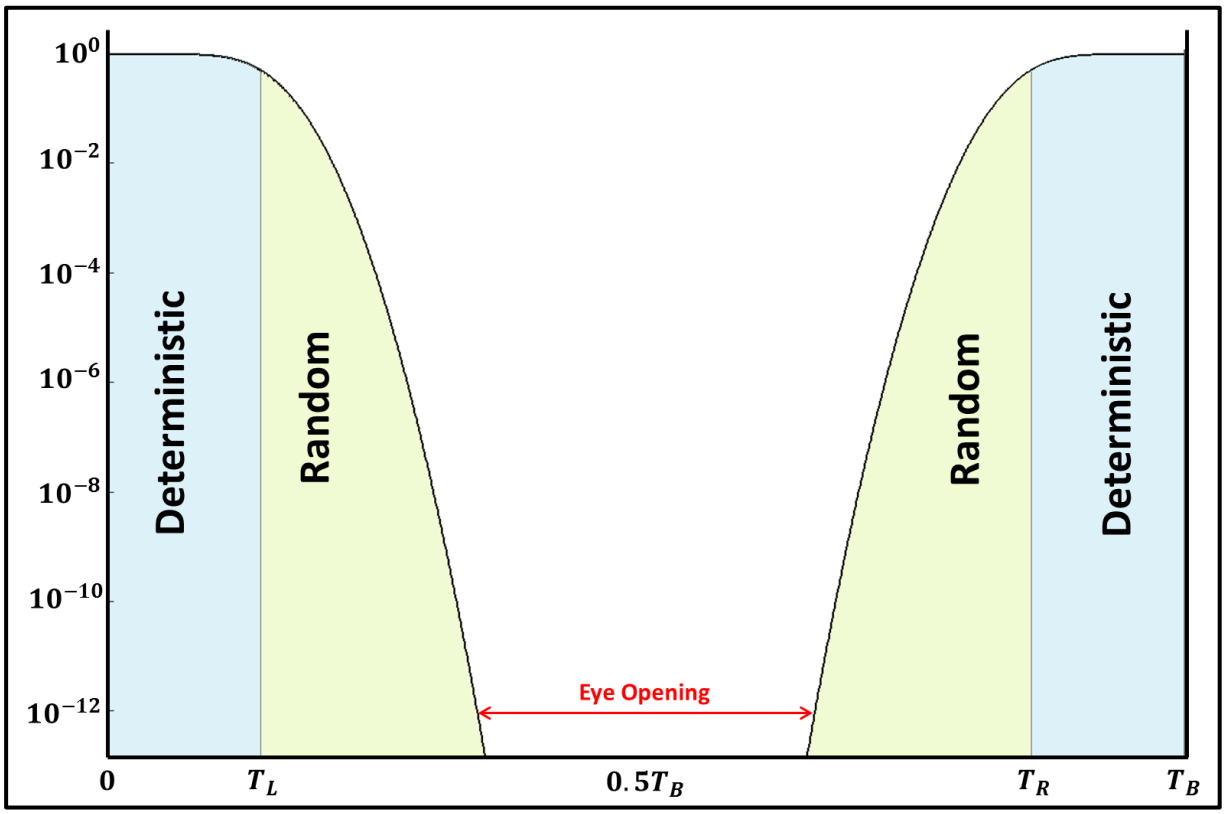


Figure 2-5: BER Graph

In the above bathtub plot, for a given BER, we have an eye-opening which corresponds to the distance between the two curves of the bathtub curve.

The purpose of this chapter is to present the general theoretical study of jitter. The complete study for calculating Jitter in the Time and Frequency domain is given here. This chapter is composed as follows:

In Section 2.2 we provide Jitter fundamentals and equations to calculate it in Time & Frequency domain.

Section 2.3 presents a complete study about Phase & Amplitude noise. In this section we will show that a noise is composed of a Phase and an Amplitude Noise. Study will be done for noisy sinusoidal signals.

Section 2.4 proves that the Amplitude noise has no impact on jitter (under a given Noise Profile Power limit).

Chapter's conclusions are given in Section 2.5.

2.2 Jitter Calculation

In this section, we provide principal equations, for calculating Jitter, in the Time Domain, and in the Frequency Domain.

2.2.1 Jitter Calculation in Time Domain

In the Time Domain, the phase noise modulated signal is expressed as in (2.6) and is shown in Fig 2-6:

Original Signal: $$sig = \alpha \sin(\omega_c t) \quad (2.5)$$

Phase Modulated Signal: $$sig_{noisy} = \alpha \sin(\omega_c t + \beta \sin(\omega_m t)) \quad (2.6)$$

where:

β is the Amplitude of the jitter, $2 * \beta$ is the Jitter peak to peak, and $\omega_m = 2 * \pi * F_m$ the jitter modulation (with F_m the modulating Frequency).

α is the Amplitude of the signal, and $\omega_c = 2 * \pi * F_c$ the carrier modulation (with F_m the carrier Frequency).

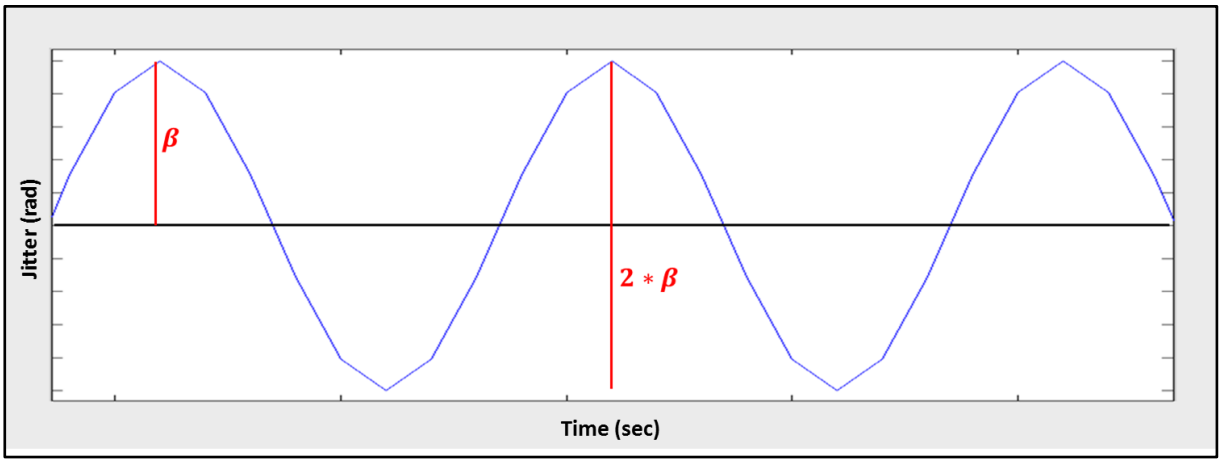


Figure 2-6: Time Domain Phase Modulation

The Jitter peak-to-peak can be expressed in radians (2.7), and in seconds (2.8).

$$Jitter_{pp_{radians}} = 2 * \beta \quad (2.7)$$

$$Jitter_{pp_{seconds}} = \frac{Jitter_{pp_{radians}}}{2*\pi*F_c} = \frac{2*\beta}{2*\pi*F_c} \quad \text{with} \quad F_c = Carrier_{Frequency} \quad (2.8)$$

Let's consider we have a TIE Jitter with ***N*** random $\boldsymbol{x_i}$ values. The Jitter RMS is expressed as in (2.9). It corresponds to the Standard Deviation of the TIE Jitter.

$$Jitter_{RMS} = \sigma(TIE) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \quad \text{where} \quad \mu = \frac{1}{N}\sum_{i=1}^{N} x_i \quad (2.9)$$

2.2.2 Jitter Calculation in Frequency Domain

2.2.2.1 1 Spur Noise

From time domain jitter calculation above, we show how to perform this computation in frequency domain. As previously computed in the time domain (2.8), the equation for the Jitter peak-to-peak is given in (2.10):

$$Jitter_{spur_{pp}} = \frac{2*\beta}{2*\pi*F_c} \quad (2.10)$$

As **β** is not directly available, we need to express this equation with information available from a spectrum analyzer, or a phase noise analyzer.

From **Annexe 8.1**, we can decompose (2.6) equation into the following expression (2.11), with the condition that jitter rms is well below 2π:

$$f(t) \simeq \boldsymbol{\alpha} sin(\omega_c t) + \frac{\boldsymbol{\alpha\beta}}{\mathbf{2}}(-\sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m)) \quad (2.11)$$

The frequency (in absolute value) representation of this modulated signal (2.11) is given in Fig 2-7.

With: $A_n = \frac{\alpha*\beta}{2}$ and $A_c = \alpha$. From which we extract β: $\beta = 2 * \frac{A_n}{A_c}$

So the Jitter Peak-To-Peak equation (2.10) can be expressed as (2.12):

$$Jitter_{spur_{pp}} = \frac{4*\frac{A_n}{A_c}}{2*\pi*F_c} \quad (2.12)$$

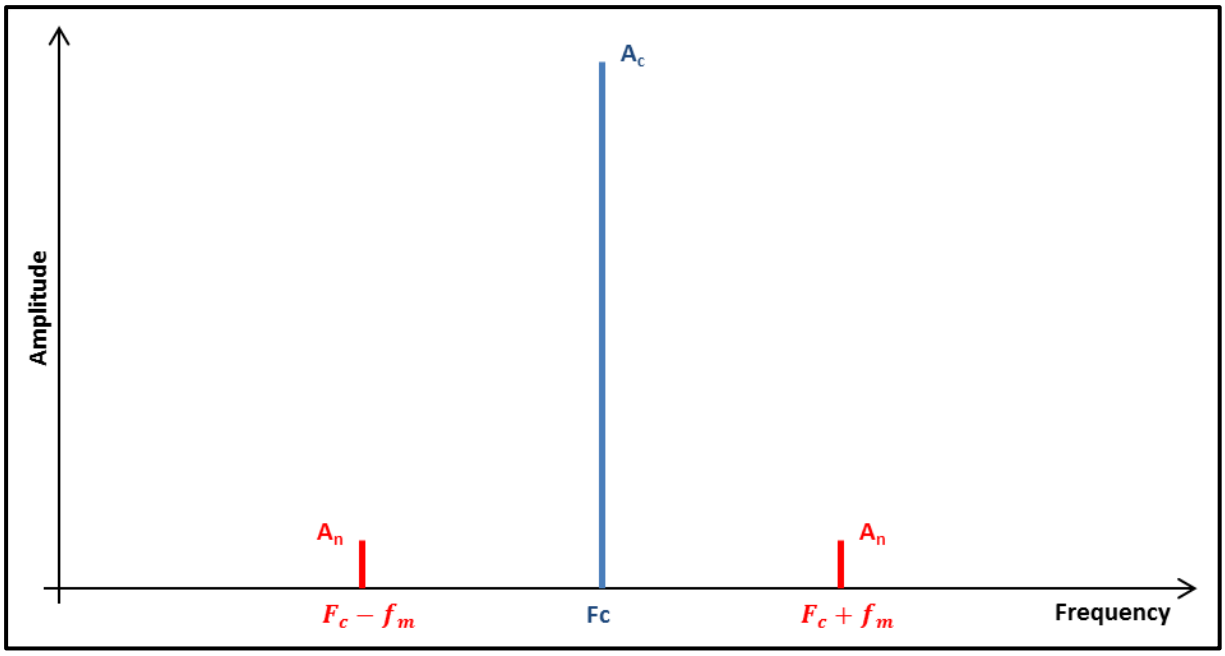


Figure 2-7: Frequency Representation of the modulated signal

The spectrum analyzer provides the Power information, in *dB*. In order to use information provided by spectrum analyzer, or phase noise analyzer, we need to express A_n and A_c into ***dBc***. The Phase Noise Profile is then expressed as in (2.13)

$$L_{spur_{dBc}} = P_{n_{dB}} - P_{c_{dB}} = 10 * \log\left(\frac{P_n}{P_c}\right) = 10 * \log\left(\frac{{A_n}^2}{{A_c}^2}\right) = 20 * \log(\frac{A_n}{A_c}) \quad (2.13)$$

From the equation (2.13), we can extract:

$$\frac{A_n}{A_c} = 10^{\frac{L_{spur_{dBc}}}{20}} = \sqrt{10^{\frac{L_{spur_{dBc}}}{10}}} \quad (2.14) \quad \text{and} \quad Jitter_{spur_{pp}} = \frac{4*\sqrt{10^{\frac{L_{spur_{dBc}}}{10}}}}{2*\pi*F_c} \quad (2.15)$$

2.2.2.2 Multi Spurs Noise Profile

Before calculating RMS Jitter of an area, we consider a carrier frequency Fc with multiple phase modulation sinuses. We have presented in Fig 2-8 the FFT with **|*absolute*|** values:

Any periodic signal, with the Fourier Series, can be written as (2.16):

$$f(t) = \alpha\sin(\omega_c t + \beta_1 \sin(\omega_{m_1} t) + \beta_2 \sin(\omega_{m_2} t) + \cdots + \beta_k \sin(\omega_{m_k} t)) \quad (2.16)$$

With Jitter: $F_{Jitter}(t) = \beta_1 \sin(\omega_{m_1} t) + \beta_2 \sin(\omega_{m_2} t) + \cdots + \beta_k \sin(\omega_{m_k} t)$ (2.17)

The average value of $F_{Jitter} = 0$.

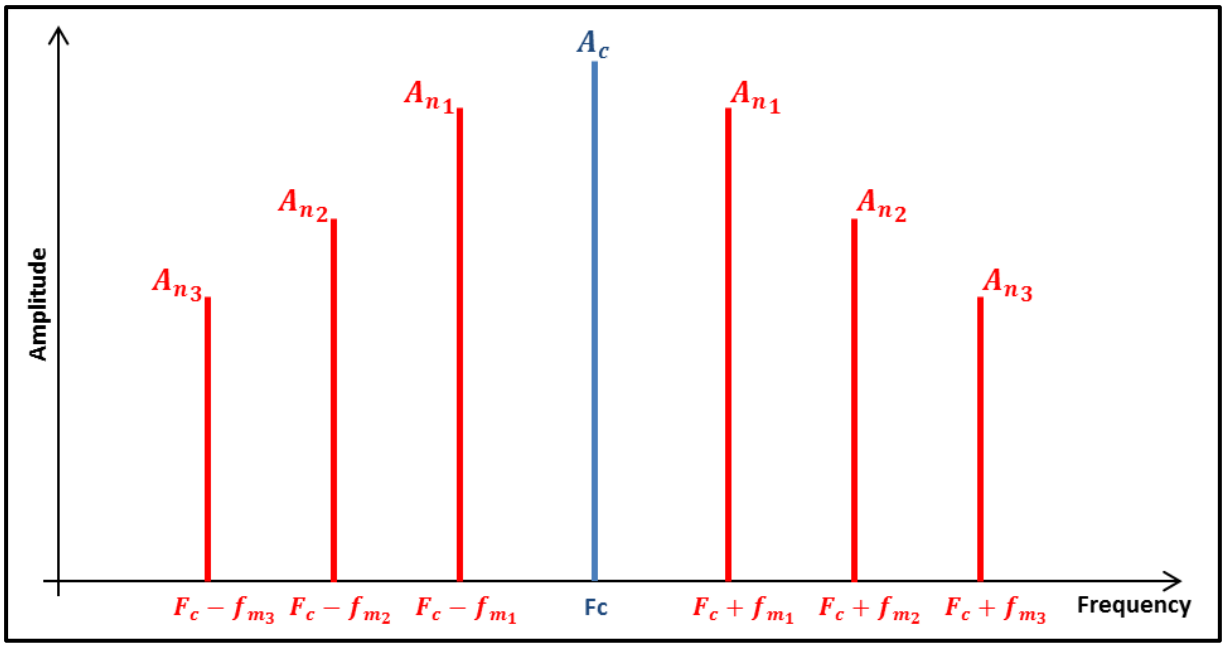


Figure 2-8: Spectral Analysis of multi-spur phase noise modulated signal

From the Parseval Theorem, with conservation of energy, we can express the RMS value of F_{Jitter} (2.18):

$$(F_{jitter_{rms}})^2 = \frac{1}{T}\int_0^T F_{Jitter}(t)^2\,dt = \sum_0^\infty \frac{1}{2}{\beta_1}^2 + \frac{1}{2}{\beta_2}^2 + \cdots + \frac{1}{2}{\beta_k}^2 \quad (2.18)$$

So the Jitter RMS will be calculated as following:

$$RMS_{Jitter_{multispurs}} = \frac{\sqrt{(\frac{1}{2}(\beta_1)^2 + \frac{1}{2}(\beta_2)^2 + \cdots + \frac{1}{2}(\beta_k)^2)}}{2\pi F_c} = \frac{\sqrt{(\frac{1}{2}(2\frac{A_{n_1}}{A_c})^2 + \frac{1}{2}(2\frac{A_{n_2}}{A_c})^2 + \cdots + \frac{1}{2}(2\frac{A_{n_k}}{A_c})^2)}}{2\pi F_c} \quad (2.19)$$

$$RMS_{Jitter_{multispurs}} = \frac{\sqrt{4*\frac{1}{2}*\sum_{k=1}^{m}(\frac{A_{n_k}}{A_c})^2}}{2\pi F_c} = \frac{\sqrt{4*\frac{1}{2}*\sum_{k=1}^{m}\sqrt{10^{\frac{L_{spur_{k\,dBc}}}{10}}}^2}}{2\pi F_c} = \frac{\sqrt{2*\sum_{k=1}^{m}10^{\frac{L_{spur_{k\,dBc}}}{10}}}}{2\pi F_c} \quad (2.20)$$

2.2.2.3 Area Noise Profile

The RMS Jitter, when a given *L(f)* noise profile is continuous (Fig 2-9), and not discrete as given before (Fig 2-8), is calculated with the following formula (2.21) [Analog] - [Feng] - [Maxim] - [Zamek], which is the integration of the whole band of frequencies:

$$RMS_{Jitter_{area}} = \frac{1}{2*\pi*F_c}\sqrt{2 * \int_0^\infty 10^{\frac{L(f)}{10}}df} \quad (2.21)$$

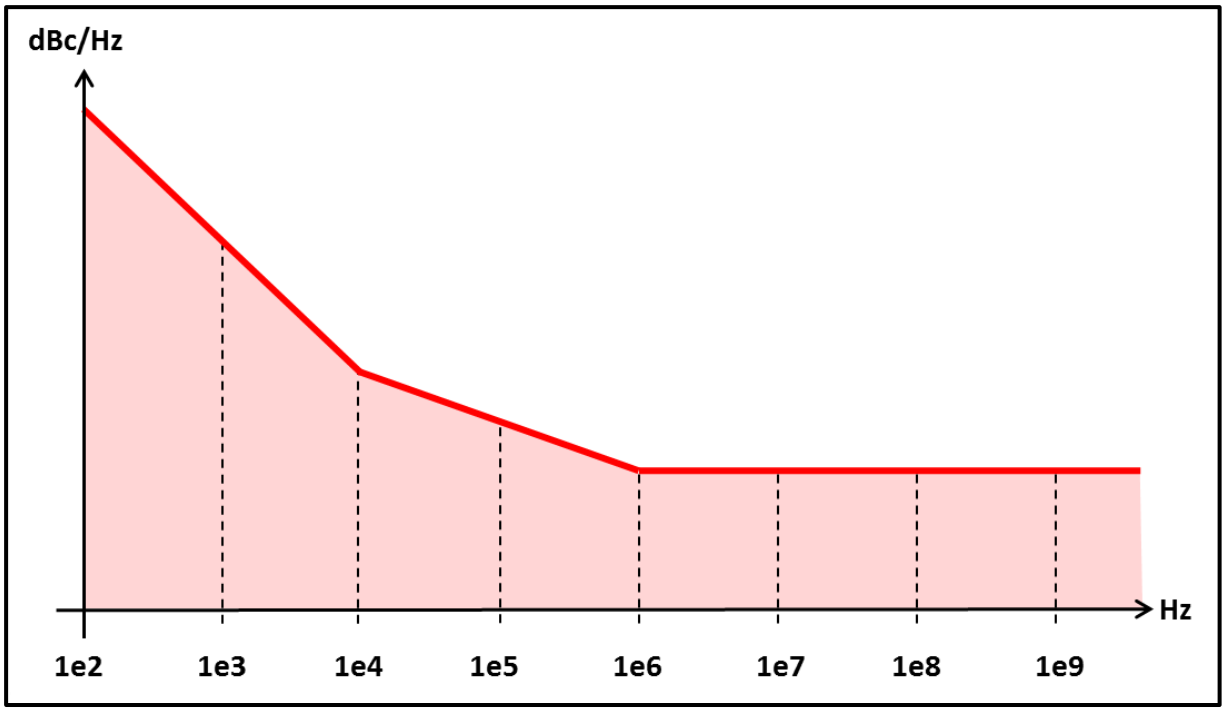


Figure 2-9: Spectral Analysis of Area Profile Phase Noise Modulated Signal

In order to calculate the $RMS_{Jitter_{area}}$, we must integrate the following frequency bandwidths:

- Up to infinite for Sinusoidal Signal
- Up to $2 * F_c$ for Square Signal

The demonstrations of Integrating Bandwidth are given in **Annex 8.4**.

2.3 Phase and Amplitude Noise Study

The objective of this section is to show that a given Noise profile can be expressed as a sum of a phase and amplitude noise.

2.3.1 Noise Spur Modulation

Let's consider we have following Carrier Frequency with given Noise. The noise can be described as a phase modulation, with one spur Dirac, as shown in Fig 2-10:

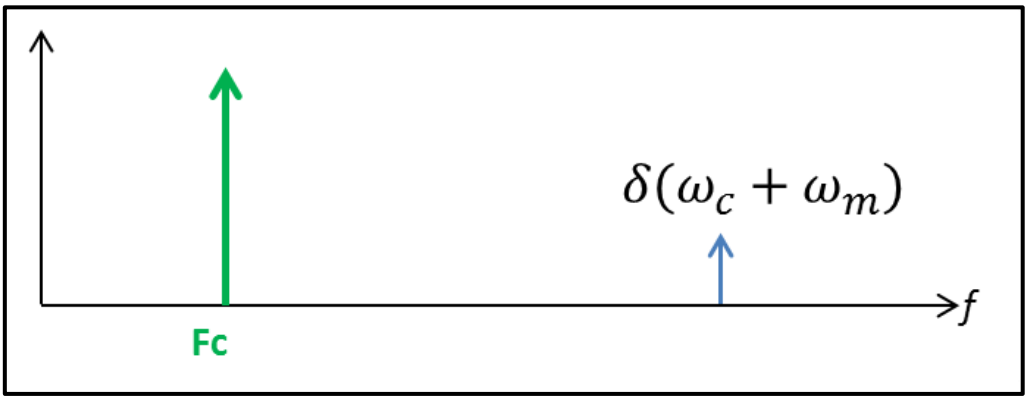


Figure 2-10: Carrier Frequency with given Noise

The phase modulation of Fig 2-10 can be written as (2.22):

$$\boldsymbol{\delta(\omega_c + \omega_m)} = \frac{1}{2}\boldsymbol{\delta(\omega_c - \omega_m)} + \frac{1}{2}\boldsymbol{\delta(\omega_c + \omega_m)} - \frac{1}{2}\boldsymbol{\delta(\omega_c - \omega_m)} + \frac{1}{2}\boldsymbol{\delta(\omega_c + \omega_m)} \quad (2.22)$$

Our hypothesis is that this is expressed with an Amplitude Noise (2.23) and Phase noise (2.24). We will prove these equations in (2.29):

$$\boldsymbol{Amplitude_{Noise}} = \frac{1}{2}\boldsymbol{\delta(\omega_c - \omega_m)} + \frac{1}{2}\boldsymbol{\delta(\omega_c + \omega_m)} \quad (2.23)$$

$$\boldsymbol{Phase_{Noise}} = -\frac{1}{2}\boldsymbol{\delta(\omega_c - \omega_m)} + \frac{1}{2}\boldsymbol{\delta(\omega_c + \omega_m)} \quad (2.24)$$

For equation development details, check **Annex 8.1**.

Equation (2.22) is Equivalent to Fig 2-11:

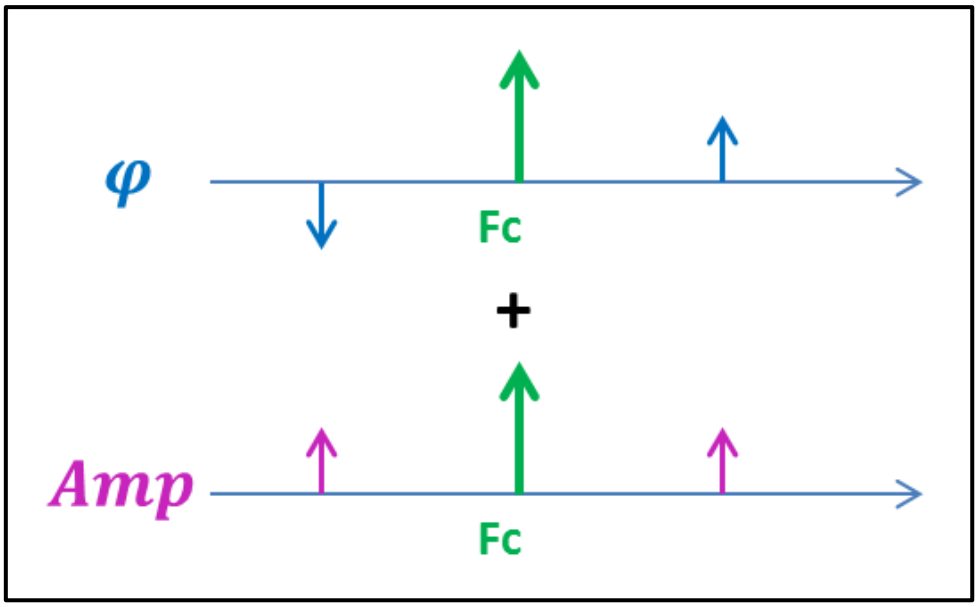


Figure 2-11: Noise decomposition into Amplitude and Phase Noise

So, a given Noise (Fig 2-10) can be expressed as sum of a Phase and an Amplitude Noise (Fig 2-11).

2.3.2 Phase Noise & Amplitude Noise Modulation

Let's consider that we have a phase and amplitude noise Modulation (2.25):

$$\boldsymbol{V(t) = V_0\, sin[(\omega_c t) + \beta sin(\omega_m t + \varphi_m)]\,[1 + \alpha cos(\omega_m t + \varphi_m)]} \quad (2.25)$$

Equation (2.25) can be expressed as (2.26):

$$V(t) = \left[V_0 sin(\omega_c t) + \frac{V_0\beta}{2}(-sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m))\right] * \quad (2.26)$$
$$[1 + \alpha\, cos(\omega_m t + \varphi_m)]$$

If we develop equation (2.26), we obtain (2.27):

$$V(t) = \left[V_0 sin(\omega_c t) + \frac{V_0\beta}{2}(-sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m))\right] + \quad (2.27)$$
$$\left[V_0 sin(\omega_c t) + \frac{V_0\beta}{2}(-sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m))\right] *$$
$$\alpha\, cos(\omega_m t + \varphi_m)$$

As $\alpha^*\beta$ << 1, (2.27) becomes (2.28):

$$V(t) = \left[V_0 sin(\omega_c t) + \frac{V_0\beta}{2}(-sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m))\right] + \quad (2.28)$$

$$V_0 sin(\omega_c t)\, \alpha\, cos(\omega_m t + \varphi_m)$$

Equation (2.28) is also expressed with Phase Noise and Amplitude Noise:

$$V(t) = V_0\, sin(\omega_c t) + \frac{\beta V_0}{2}[-\boldsymbol{sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m)}] \quad (2.29)$$

$$+ \frac{V_0\alpha}{2}[\boldsymbol{sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m)}]$$

Equation (2.29) is represented in Fig 2-12.

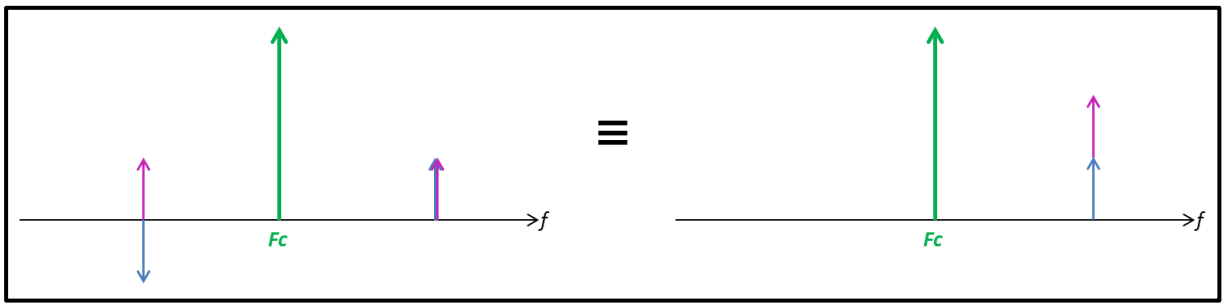


Figure 2-12: Amplitude & Phase Noise Modulation for Sinusoidal Signal

In conclusion, any noise added to the signal, can be represented as a sum of a Phase and an Amplitude noise.

This is also demonstrated for Square Signals. For equations and demonstration of noise added to square signals, check **Annex 8.2**.

The particular case of measuring transition positions at a given offset is given in **Annex 8.3**.

2.4 Phase & Amplitude Noise Impact onto TIE jitter

In this chapter, we will show the impact of Phase and Amplitude noise on TIE jitter:

2.4.1 Translate phase noise into TIE Jitter

As shown in Section 2.2, the TIE Jitter is calculated as follows:

- **In Time Domain:**

$$Jitter_{spur_{pp}} = \frac{2*\beta}{2*\pi*F_c} \tag{2.30}$$

- **In Frequency Domain:**
  - ***1 Spur Noise:***

$$Jitter_{spur_{pp}} = \frac{4*\sqrt{10^{\frac{Lspur_{dBc}}{10}}}}{2*\pi*F_c} \tag{2.31}$$

  - ***Multi Spur Noise:***

$$RMS_{Jitter_{multispurs}} = \frac{\sqrt{2*\sum_{k=1}^{m} 10^{\frac{Lspur_{kdBc}}{10}}}}{2\pi F_c} \tag{2.32}$$

  - ***Area Noise:***

$$RMS_{Jitter_{area}} = \frac{1}{2*\pi*F_c}\sqrt{2 * \int_0^{\infty} 10^{\frac{L(f)}{10}} df} \tag{2.33}$$

We integrate:

- Up to infinite for Sinusoidal Signal
- Up to $2 * F_c$ for Square Signal

2.4.2 Translate amplitude noise into TIE Jitter

2.4.2.1 One spur Amplitude Noise Modulation

The Amplitude Noise Modulation is expressed as follows (2.34):

$$\boldsymbol{V(t) = V_0\, sin(\omega_c t)\,[1 + \alpha\, cos(\omega_m t)]} \tag{2.34}$$

We will have a transition due to amplitude jitter, each time $|\alpha\cos(\omega_m t)| > 1$. This might come at any time, and would deteriorate our TIE Jitter.

In order to prevent this effect, α should be <<1.

If α << 1, then $-1 \ll \alpha\, cos(\omega_m t) \ll 1$ (2.35)

Or, $1 + \alpha\, cos(\omega_m t) \gg 0$ (2.36)

So if α << 1, the sign on V(t) will not change and **the amplitude noise will not impact the TIE Jitter**.

2.4.2.2 Spectrally flat profile Amplitude Noise modulation

In this section, we will define the profile limit, below which the amplitude noise has No impact on the Jitter. This is done following several steps:

- First of all, depending on the samples number ($N_{samples}$), we find how many sigma σ we have:

For example, if we have $N_{samples}$ = 1e6 samples, then, we find the corresponding number of sigma σ (N_{sigma}):

$$N_{sigma} = 2 * \sqrt{2} * erfcinv\left(\frac{1}{N_{samples}} * 2\right) = 2 * \sqrt{2} * erfcinv\left(\frac{1}{10^6} * 2\right) = \mathbf{9.5068\sigma} \quad (2.37)$$

In order to have half of the jitter Peak-to-Peak being below -1 (so we have a transition due to Amplitude Noise), we create a random amplitude noise with twice the σ value:

The value of 2 times the σ is: $\frac{2}{N_{sigma}} = 0.2104.$ (2.38)

$$Amplitude_{noise_{modulation}} = rand(1, N_{samples}) * \frac{2}{N_{sigma}} \quad (2.39)$$

After multiple simulation results on Matlab, we verify that this is the limit, below which amplitude noise will have no impact on TIE Jitter.

The 2nd step is to find the Profile Limit, below which the Amplitude Noise has no Impact.

As in (2.39), in order to have a transition due to Amplitude Noise, we search the limit for twice the value of σ.

We have found it theoretically, and confirmed it through Matlab Simulations.

In conclusion, under a defined profile, the amplitude noise will have no impact on TIE Jitter. This is why we will take only the Phase Noise into account.

2.5 RJ/DJ Jitter Correlation Algorithm

2.5.1 Introduction

The total Random (2.40) and total Deterministic (2.41) jitters are defined as following in the literature [Agilent] - [Kim] - [Tektronix]:

$$RJ_{Total} = \sqrt{RJ_1{}^2 + RJ_2{}^2 + RJ_3{}^2 + \ ...} \quad (2.40)$$

$$DJ_{Total} = DJ_1 + DJ_2 + DJ_3 + \ ... \quad (2.41)$$

The Total Jitter is given as in (2.42):

$$TJ = N(BER).RJ_{Total} + DJ_{Total} \quad (2.42)$$

with N(BER), a multiplicative coefficient depending on the BER value, as given in Table 2-1:

Table 2-1: Multiplying coefficients depending on BER.

BER	10^{-7}	10^{-8}	10^{-9}	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}	10^{-15}
N(BER)	10.399	11.224	11.996	12.723	13.412	14.069	14.698	15.301	15.883

This value is given as a Worst Case value in the standards [PCIe]. For the Total Deterministic Jitter, all DJ variables are independent from each other, so they can be considered as Random variables. Therefore, the total DJ is a convolution between all DJ [Papoulis] - [Tektronix]. The real Total Jitter (TJ) is then smaller than the calculation equation given in (2.42). This is why we are proposing to better estimate the Total Jitter. This would help designers to better constraint design and achieve desired FoM.

2.5.2 Proposed Method Principle

The idea of the new calculation method, with all steps is given in Fig 2-13. Its principle is the following:

Step 1:

As explained above, the total Deterministic Jitter is defined as the convolution of all Deterministic Jitters found in the system, as given in (2.43).

$$DJ_{Total} = DJ_1 * DJ_2 * DJ_3 * \ ... \quad (2.43)$$

The total Random Jitter is defined same as before (2.40).

Step 2:

The Total Jitter corresponds then to the convolution of the total DJ spurs (2.43) to the RJ standard deviation (2.40), and is expressed as in (2.44):

$$TJ = DJ_{Total} * RJ_{Total} \quad (2.44)$$

Step 3:

Equation (2.44) will give the CDF and CCDF of the total Jitter, in a linear scale.

Step 4:

The calculated CDF & CCDF are transposed to a logarithmic scale, and extrapolated to have curve estimation at different BER. The difference between the CDF and CCDF curves at any given BER will give the Total Jitter estimation at this BER.

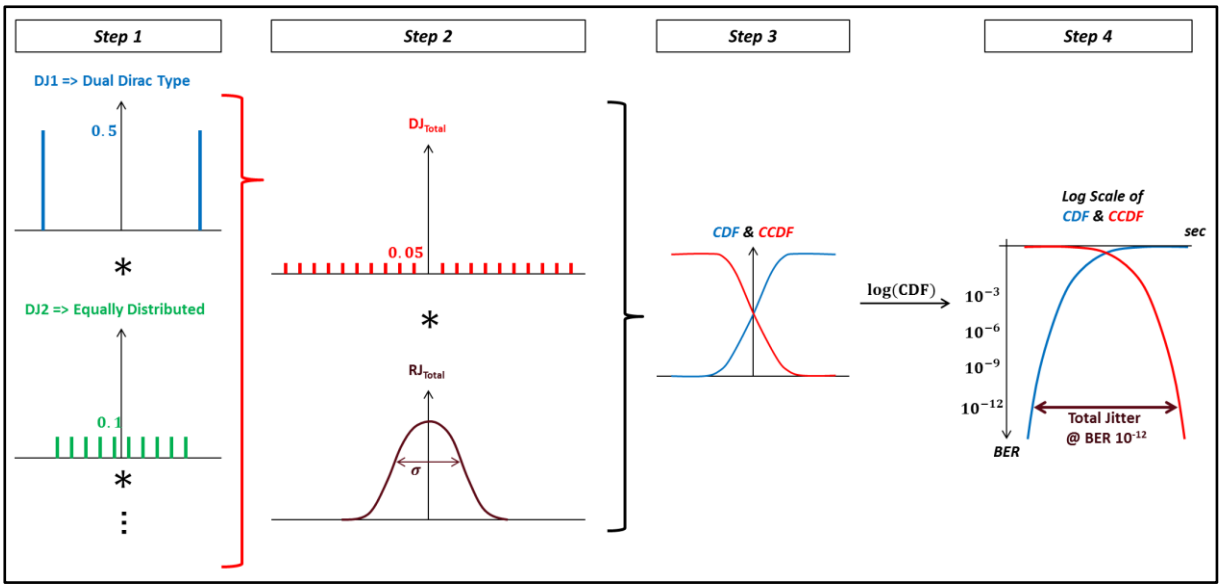


Figure 2-13: Total Jitter Estimation

Some examples of different convolutions are given in **Annex 8.5**. They prove that the developed algorithm which performs the convolutions between all DJ jitters works as expected.

2.6 Chapter Conclusion

In this section we have given all necessary equations to calculate jitter in the time and frequency domain.

We have shown that any given noise can be expressed as sum of a Phase and an Amplitude Noise.

We have demonstrated that the Amplitude Noise has no impact on Jitter under a given profile. Therefore, in this thesis, we will only work on Phase Noise Profiles for the jitter.

Chapter 3: PLL Modeling

Table of Figures

Table of Tables

3.1 Introduction

Modeling the phase noise of the different SerDes components and extracting the time jitter would help designers to achieve desired Figure of Merit for future SerDes versions [Bidaj]. The phase locked loop (PLL) is one of the main contributors of clock random and periodic jitter inside the system [Hansel] - [Telba]. This chapter presents a method for modeling the PLL with phase noise injection and estimating the time domain jitter. A time domain model including PLL loop nonlinearities is created in order to estimate jitter. We present in this chapter a particular PLL architecture, which is also implemented in STMicroelectronics SerDes circuits.

3.2 PLL Architecture

A phase-locked loop is a feedback system where an oscillator-generated signal is phase locked to a reference clock [Soyuer]. Phase-locked loops can be used, for example, to generate stable output high frequency clock from a fixed lower-frequency one.

A PLL is composed of the following building blocks: Phase Detector, Charge Pump, Low-Pass Filter, Voltage Controlled Oscillator (VCO) and Feedback Divider, as shown in Fig 3-1. [Analog] - [Arakali] - [Chu] - [Herzel] - [Nonis] - [Pu] - [Schober] - [Ting] - [Wu] - [Yuan].

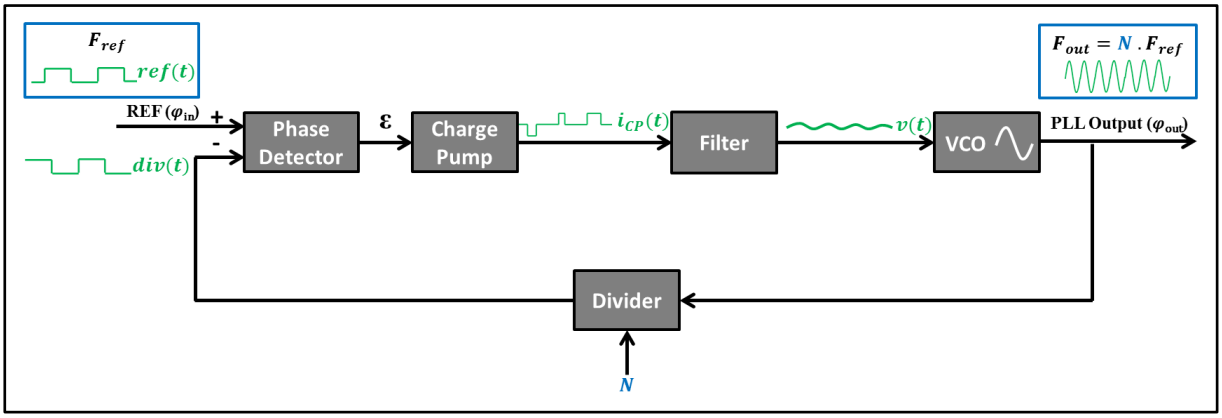


Figure 3-1: PLL Architecture

At the input of the PLL, we have the **Reference Clock (Oscillator)**, which is a periodic signal. For SerDes applications, it is generally a Clock signal, between 20MHz and 100MHz.

3.2.1 Phase Detector

The **Phase-Frequency Detector** (PFD) compares the phase and frequency of the feedback clock generated by the VCO to the ones of the REF clock, and generates an error to adjust the oscillator and keep the phases and frequencies matched. It produces an UP and DOWN output signal [Chu] - [Kennedy] - [Pu] - [Schober] - [Young] - [Yuan], proportional to the phase error when the PLL reaches the locked state (as shown in Fig 3-2).

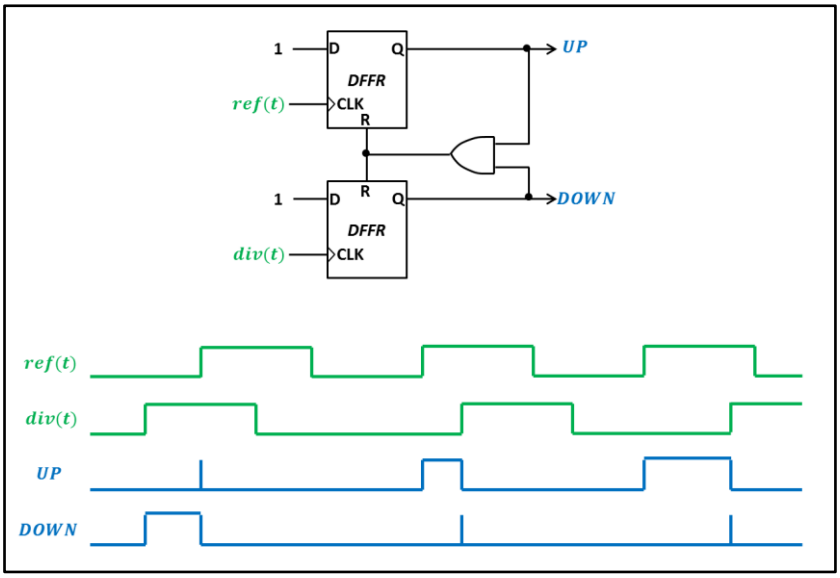


Figure 3-2: Phase-Frequency Detector Output

3.2.2 Charge Pump (CP)

The above UP and DOWN output phase error signals enter the **Charge Pump** as shown in Fig 3-3 [Schober].

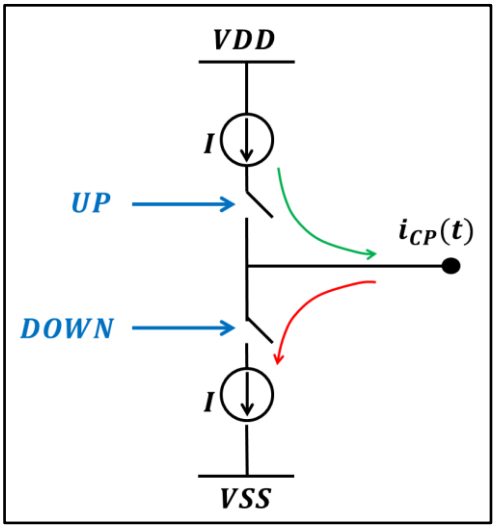


Figure 3-3: Charge Pump schematic

The charge pump generates a current related to the input phase errors, which is proportional to the PFD pulse widths [Chu]. (3.1)

$$CP_{Gain} = \frac{1}{2\pi} * I_{CP}$$

3.2.3 Loop Filter

The Loop **Filter** is an integrator. It generates a voltage to control the VCO from phase detector pulses [Young], and make it go faster/slower depending on the phase error sign. In Fig 3-4 we show a 4th order filter, very common in the PLL for RF applications.

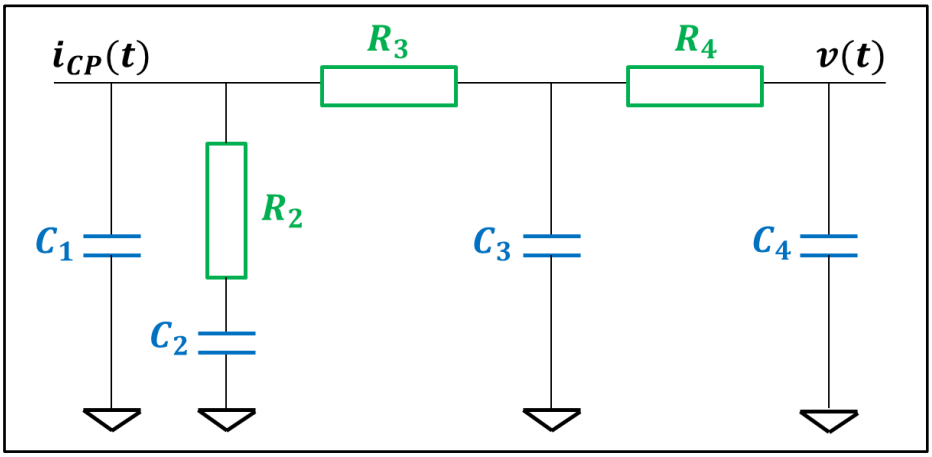


Figure 3-4: Loop Filter Schematic

The C_1 capacitance is used to integrate the current coming from the Charge Pump. The resistance R_2 and the capacitance C_2 add one zero to the Transfer Function to ensure the loop stability. The two first order low-pass filters added afterwards serve to better filter the high frequency noises and the reference frequency spurs.

3.2.4 VCO

A Voltage-Controlled oscillator is an oscillator whose output oscillation frequency is controlled by the filter's output voltage $v(t)$, as shown in Fig 3-5.

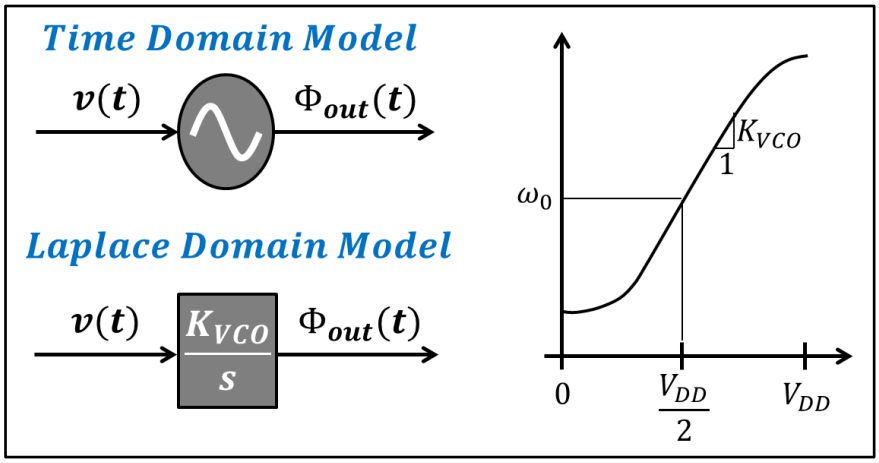


Figure 3-5: VCO

The time-domain phase relationship is given in (3.1).

$$\Phi_{out}(t) = \int \Delta\omega_{out}(t)dt = K_{VCO} \int v(t)dt \quad 3.1$$

with $\omega_{out}(t) = \omega_0 + \Delta\omega_{out}(t) = \omega_0 + K_{VCO}v(t)$ 3.2

3.2.5 Divider

The **Divider** will serve as a frequency divider as shown in Fig 3-6. The loop feedback frequency will be a division of the PLL output frequency.

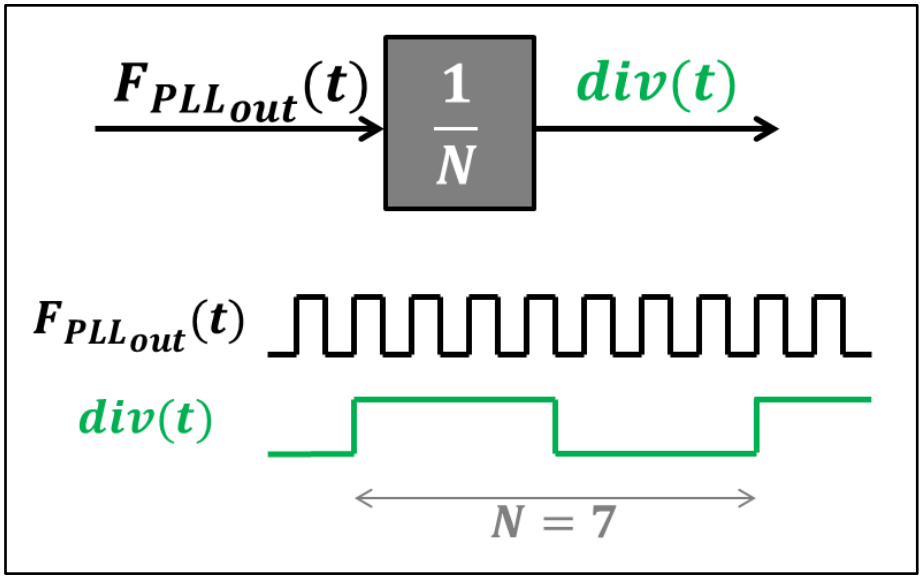


Figure 3-6: Divider

3.2.6 Sigma-Delta Modulator:

A Sigma Delta (SD) modulator is implemented for the Fractional PLL [Kennedy] - [Sadeghi] - [Ye]. The schematic of the PLL with the SD modulator added is given in Fig 3-7.

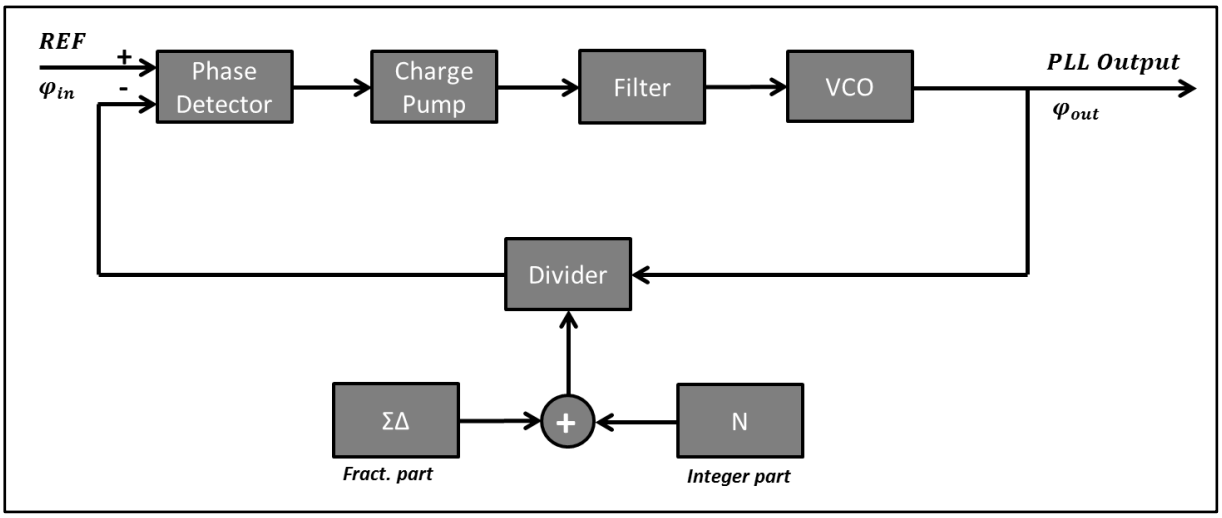


Figure 3-7: PLL schematic with Sigma Delta Modulator

The schematic of the SD modulator (3-TAP Sigma Delta Modulator) is given in Fig 3-8:

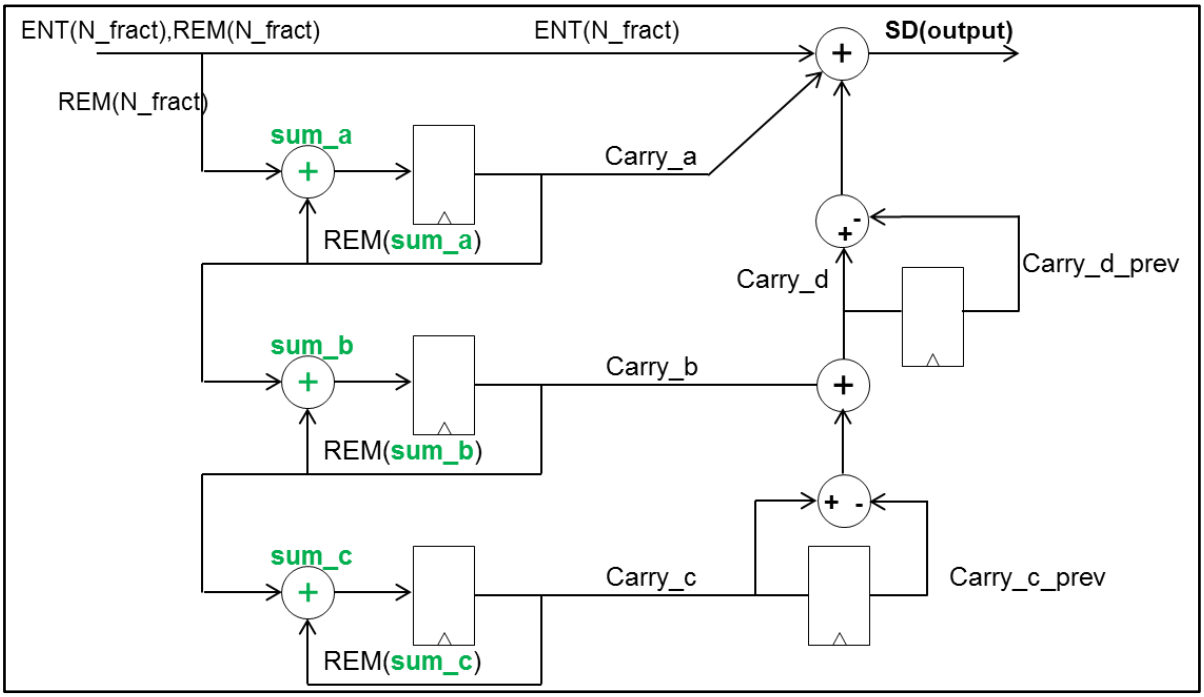


Figure 3-8: Sigma Delta Modulator

A 3-TAP Sigma Delta Modulator is chosen in order to send the noise due to the division in the High Frequencies. This noise will then be filtered by the PLL Low Pass Filter equivalent transfer function.

3.3 Frequency Domain PLL modeling with phase noise injection

3.3.1 Fractional PLL modeling for a small signal analysis

The first step is to model each sub-block of the PLL in MATLAB, with its proper transfer function. A phase noise profile is then injected for each sub-block. The PLL schematic with phase noise injection for each sub-block is given in Fig 3-9.

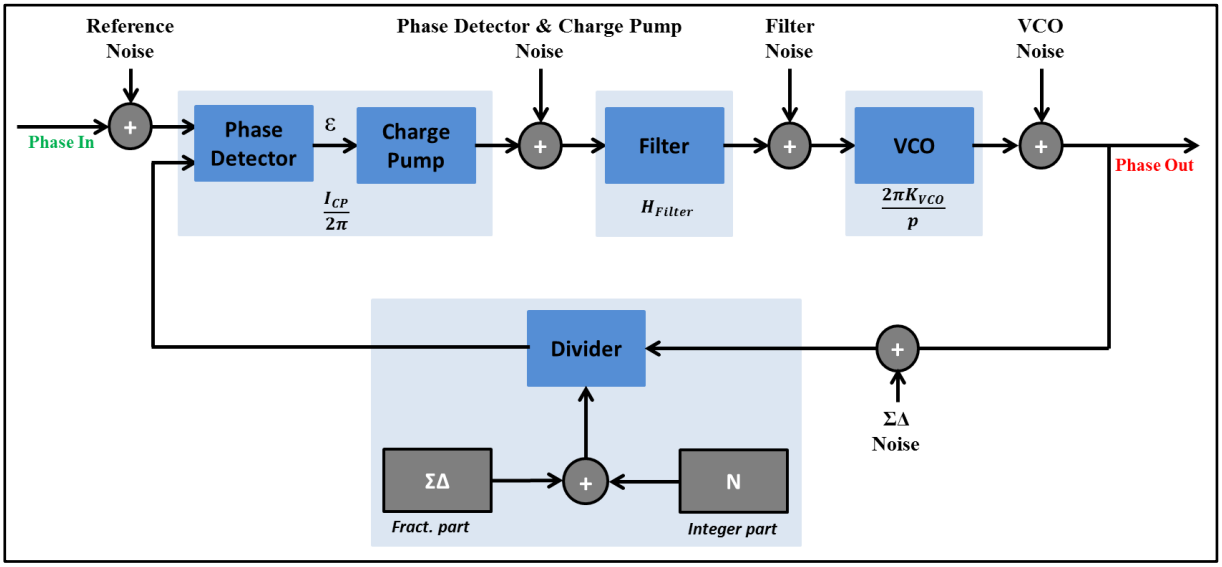


Figure 3-9: PLL Schematic with Phase Noise Injection

The small-signal closed-loop transfer function of the PLL is described in (3.3):

$$TF(s) = \frac{\frac{I_{CP}}{2\pi} H_{Filter}(s)\, 2\pi \frac{K_{VCO}}{s}}{1+\frac{1}{N}\frac{I_{CP}}{2\pi} H_{Filter}(s)\, 2\pi \frac{K_{VCO}}{s}} \tag{3.3}$$

With: I_{CP}: Charge pump current.

$H_{Filter}(s)$: Filter transfer function to ensure loop stability with a cutoff frequency at 150 kHz, and a phase margin of 60°.

K_{VCO}: Gain of the VCO in Hz/V, N: Frequency Division ratio

We give in Fig 3-10 an example of the Noise Transfer Function of each sub-block of the PLL.

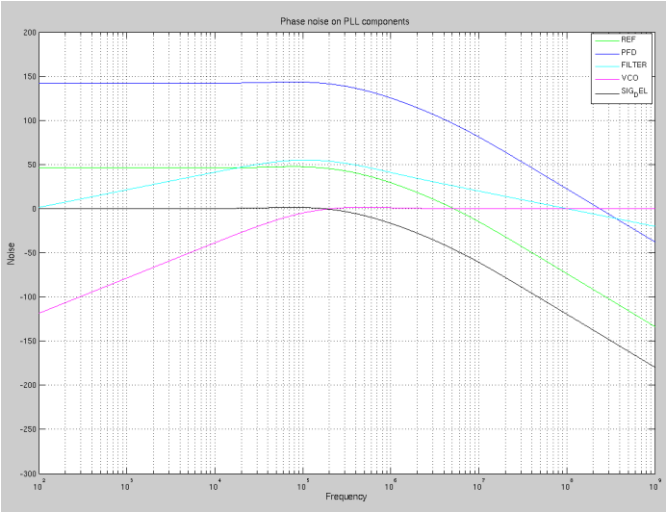


Figure 3-10: Noise Transfer Function for each sub-block of the PLL

3.3.2 Open Loop Phase Noise Profiles

Phase noise profiles for each sub-block of the PLL (except the ΣΔ profile which is a mathematical equation of order 3 ΣΔ) are simulated with ELDO RF. Their open-loop phase noise profiles (in dBc/Hz) are shown in Fig 3-11 (Charge pump noise profile is in $^{dBA}/_{\sqrt{Hz}}$ and Filter loop noise profile is in $^{dBV}/_{\sqrt{Hz}}$).

Simulation conditions might change the different phase noise profiles. We have a realistic distribution of the noise. All profiles are given in single-sideband (SSB) power [Howe]. The VCO and the ΣΔ noise profiles are around the 1st harmonic (H1) of the VCO frequency. The reference profile is around H1 of the reference frequency. The filter and phase detector/charge-pump profiles are around DC.

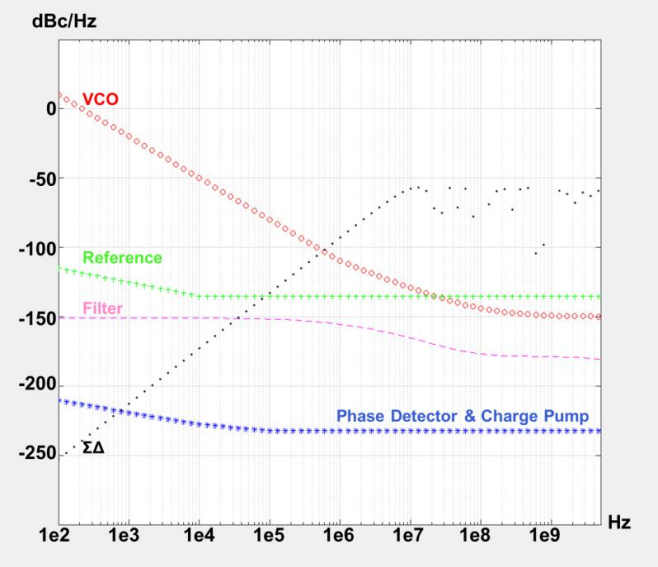


Figure 3-11: Open-loop phase noise profiles

These open-loop phase noise profiles (Fig 3-11) are passed through the closed-loop system response (Fig 3-10) and the output closed-loop noise profiles are given in Fig 3-12:

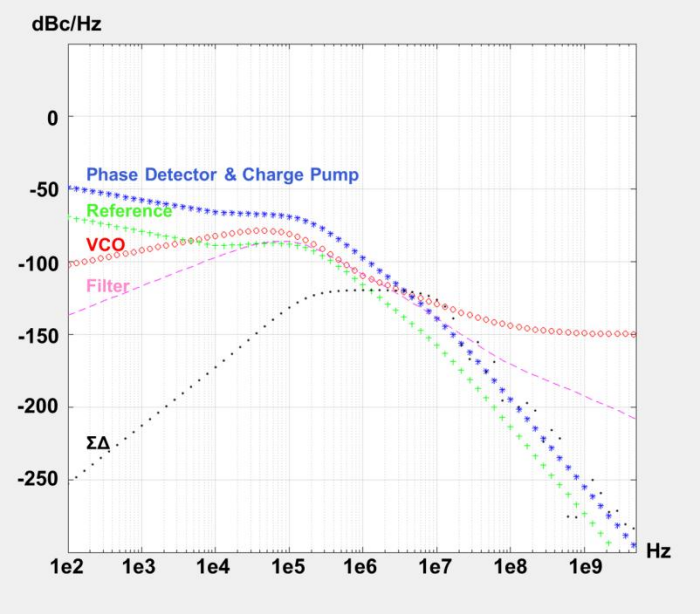


Figure 3-12: Closed-loop phase noise profiles

3.3.3 PLL Simulation in the Frequency Domain

It is important to estimate the jitter present at the output of the PLL (in closed-loop) as the PLL is one of the main contributors of jitter. To do this, the RMS jitter is computed in the frequency domain using the closed-loop **phase noise profiles *L(f)*** presented in Fig 3-12. This is done using the mathematical formula (3.4) with $\boldsymbol{F_C}$ (Carrier Frequency) [Feng] - [Maxim] - [Zamek]. The integration is multiplied by 2 (to include the total power, as the noise profiles are given in SSB).

$$RMS_{Jitter} = \frac{1}{2\pi F_C}\sqrt{2\int_0^{\frac{F_C}{2}} 10^{\frac{L(f)}{10}} df} \qquad (3.4)$$

The frequency domain model gives a global estimation of the RMS jitter. It does not give visibility of the DJ linked to imperfections of the PLL loop, which is possible with time domain model.

In order to characterize SerDes performance in terms of BER including nonlinearities, we need to model and measure the jitter in the time domain.

3.4 Time domain PLL modeling with jitter injection

The time domain PLL simulation was done using the generated open-loop phase noise profiles.

3.4.1 Time jitter generation

The steps to generate the time noise jitter are given below, and are shown in Fig 3-13. This method allows the time jitter to be obtained from the phase noise profile.

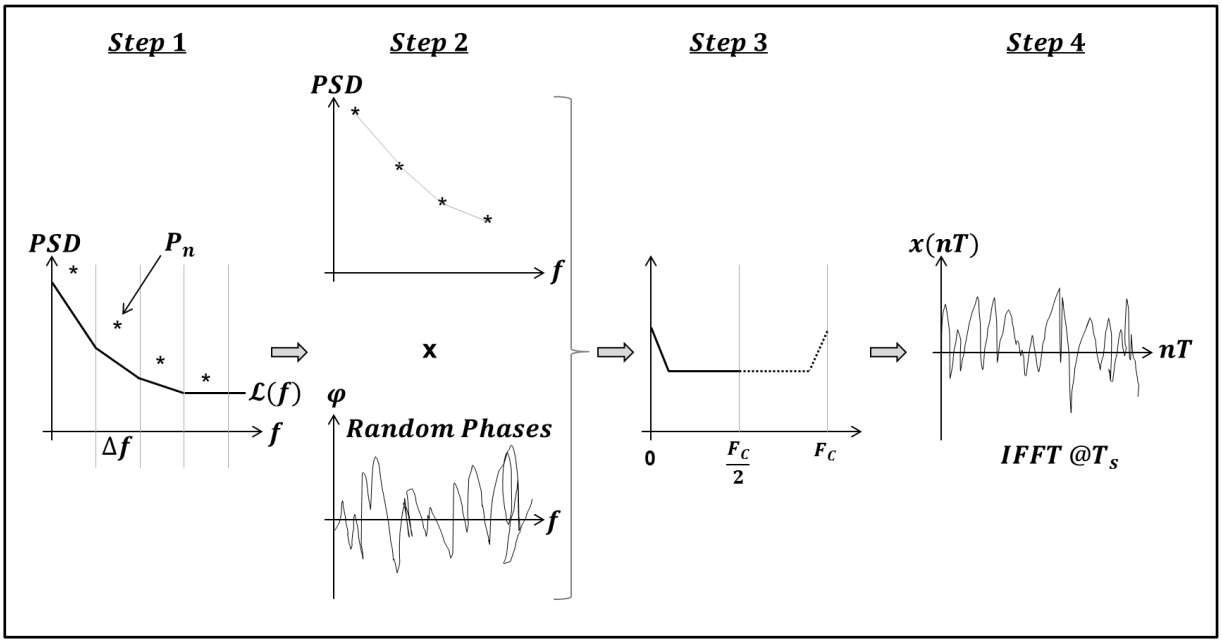


Figure 3-13: Time Jitter Generation Technique

Step 1: First open-loop phase noise profile $\mathcal{L}(f)$ is recovered from the ELDO RF simulation, calculated as in (3.4).

$$\mathcal{L}(f) = 10log_{10}(\frac{A_N(f)^2}{A_C{}^2}) \quad (3.4)$$

where A_N is the phase noise and A_C the carrier amplitude at a given frequency.

The power *P* of a spur *n*, corresponding to *Δf* frequency step, around 'f_n' offset frequency, is calculated in (3.5):

$$P_n = \int_{f_n-\frac{1}{2}\Delta f}^{f_n+\frac{1}{2}\Delta f} \mathcal{L}(f)df \qquad \text{(in dBc/Δf)} \quad (3.5)$$

Step 2: For each value of the power profile P_n, we associate a random phases with equal probability to be from 0 to 2π defined by the formula (3.6).

$$\varphi_n = e^{j\,2\pi\,rand[0:1]} \quad (3.6)$$

By multiplying the power profile defined in (3.5) by the associated random phases as in (3.6), we obtain the phase noise PN_n which is the noise corresponding to frequency offsets from 0 to $F_c/2$, by step of Δf, with F_c carrier frequency (3.7).

$$PN_n = P_n \varphi_n \tag{3.7}$$

Step 3: To generate the time noise jitter from the frequency noise, the inverse FFT is used. To do this, we complete the vector of the FFT corresponding to frequencies $F_c/2 \rightarrow F_c$, with the conjugate and symmetric of the phase noise profile. In this way, performing the IFFT of the frequency vector provides a time vector with real values only.

Step 4: The IFFT is calculated, and the time jitter noise is added to the transition position of the perfect clock signal, on the different sub-blocks of the PLL. This will be shown in next paragraph.

3.4.2 PLL Simulation in the Time Domain

We present here a time domain model (sampled model) which permits to take into account the non-linearity effect of the PLL. Furthermore, this permits to study precisely the time and frequency behavior of the PLL, and predict the different spurs in the frequency spectrum.

The model proposed in this section is a sampled model, which means that only some events inside the PLL are taken into account. This increases the efficiency of the model, because it can reduce the number of points of the simulation, and keep a good precision.

The samples for calculations are taken only on useful edges, as shown in the chronogram of Fig 3-14.

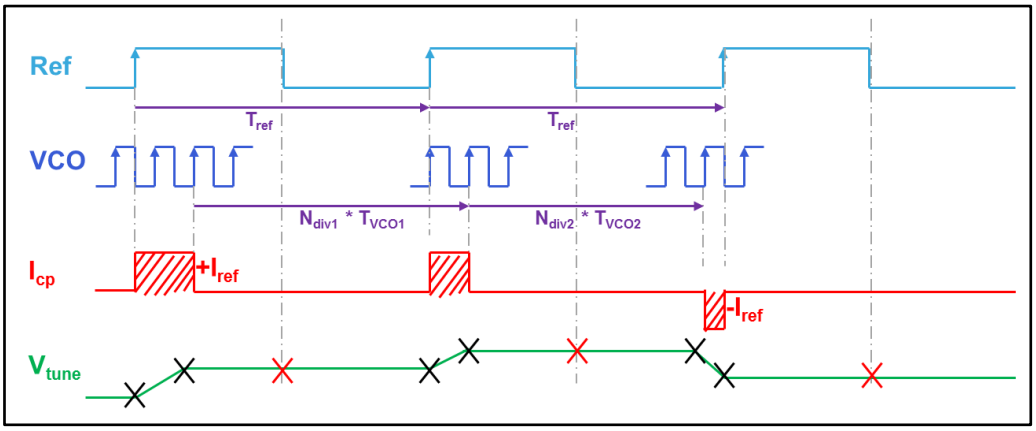


Figure 3-14: Chronogram

The minimum calculation points for the model are represented with a cross on the voltage control V_{tune} in Fig 3-14. They correspond to modifications of state of the system, as for example the change of the charge pump I_{cp} value. This chronogram (Fig 3-14) represents the behavior of the PLL as introduced in previous section 3.3.

Based on Fig 3-14, we can develop the model sampled at T_{ref} and based on the complete flow chart shown in Fig 3-15.

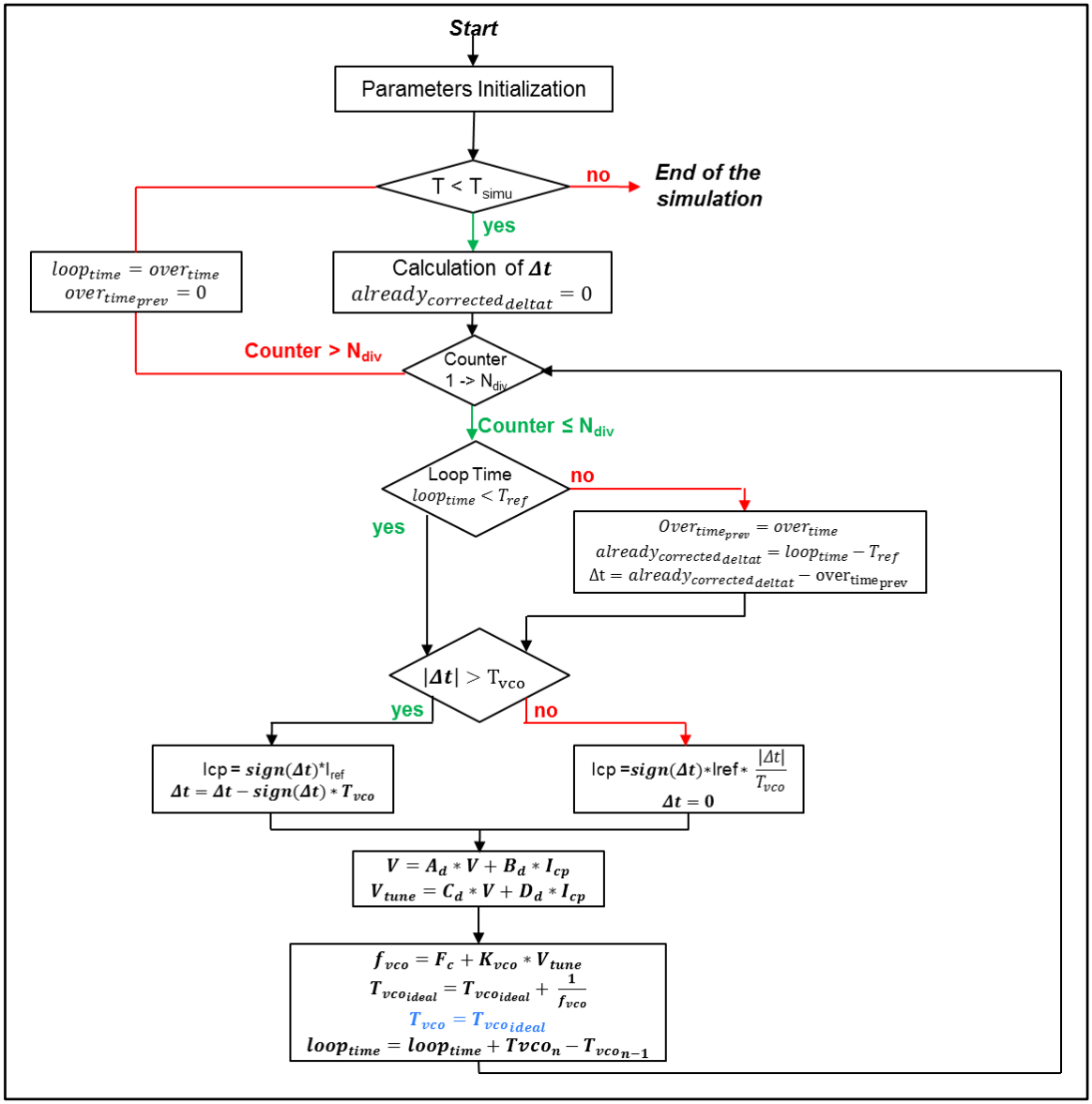


Figure 3-15: Flow Chart

First of all it is important to initialize the simulation parameters (simulation duration, sampling frequency, …), and the different PLL parameters (reference frequency, division ratio, VCO gain, …).

Then the phase difference (or delay) Δt between the reference edges and the division feedback are calculated. This corresponds to the modeling of the Phase Detector. A counter up to N_{div} is used, in order to simulate the VCO, N_{div} times faster than the reference. The Loop time is calculated, in order to update Δt value.

Then depending the Δt sign and value, the I_{cp} is calculated. This corresponds to the Charge Pump generated current. The loop filter converts the generated I_{cp} into the V_{tune} voltage. At the end, the VCO generates the f_{vco} frequency output depending on the V_{tune} voltage input.

The next step is to inject the jitter corresponding to each block, into the Time Domain PLL Model. This is done as shown in the Flow Chart in Fig 3-16. We add the jitter corresponding to the different blocks of the PLL, in specific places at the flow chart.

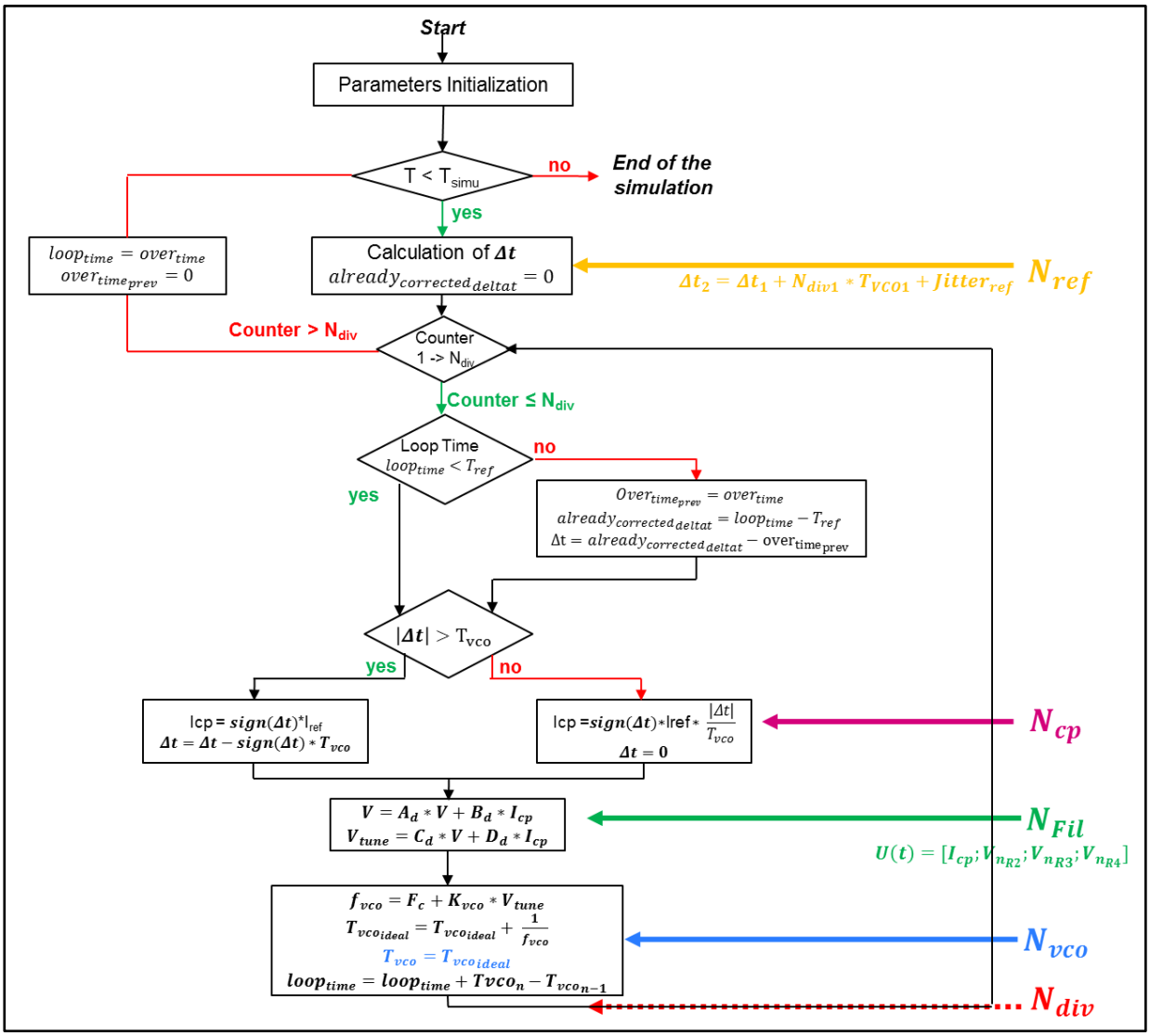


Figure 3-16: Flow Chart with Jitter injection

At the end of the time simulation of the PLL in closed-loop we measure the standard deviation, σ, of the TIE which corresponds to the closed-loop RMS jitter and the jitter due to circuit model nonlinearities.

3.5 Frequency-Time domain comparison

We simulated Phase Noise of each sub-block of the PLL in both models and compared them to see if they correlate in terms of Phase Noise and RMS Jitter.

We show in Fig 3-17 one example of simulation results comparing VCO Phase Noise at the Output of the PLL, for both models. In red color, we have the Frequency Domain, and in blue color, the Time Domain simulation results. We remark that both models correlate very well.

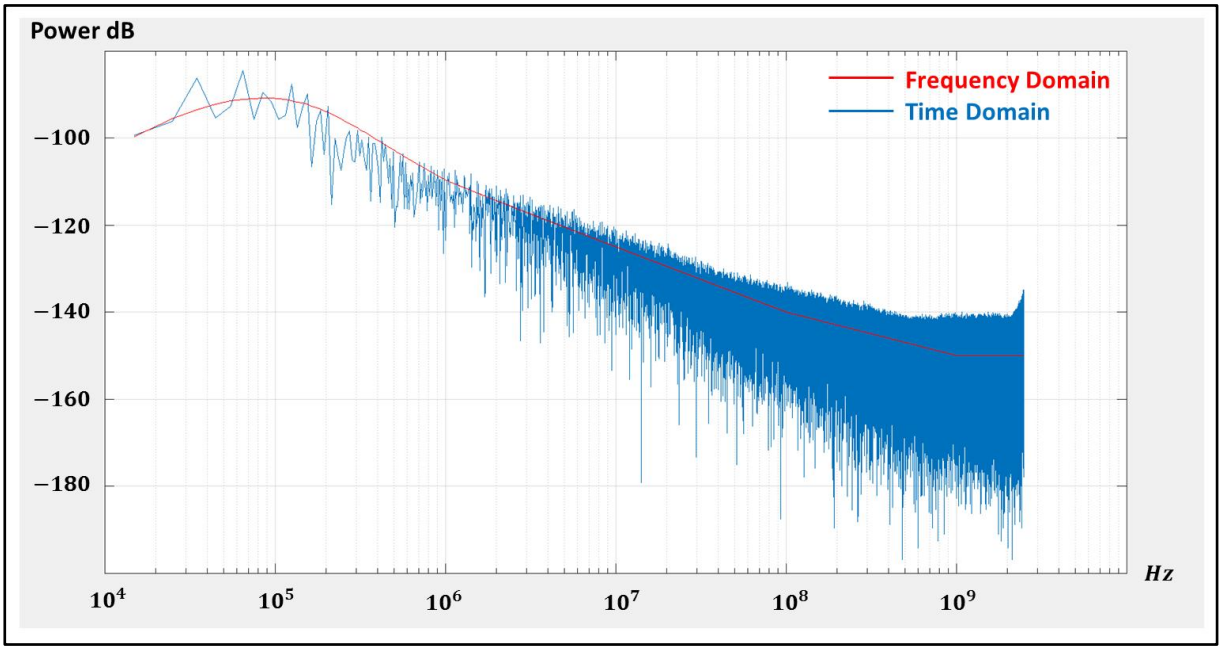


Figure 3-17: VCO Phase Noise at the PLL Output for Frequency & Time Domain Models

Furthermore, Table 3-1 shows one example of jitter results in terms of RMS jitter for each sub-block of the PLL, and the comparison between frequency domain simulations as defined in 3.3 and time domain simulations as defined in 3.4. For the comparison, in the time domain simulation, we have not introduced nonlinearity defaults, but have taken into account only the jitter due to Phase Noise profiles.

Table 3-1: Correlations between Frequency & Time Domain Simulations

PLL Sub-Blocks	Frequency Domain RMS Jitter	Time Domain RMS Jitter	Correlation % Error
VCO	4.57 ps	4.52 ps	1.1 %
REF	0.82 ps	0.81 ps	1.2 %
CP	0.62 ps	0.63 ps	1.6 %
FILTER	1.06 ps	1.07 ps	0.9 %
ΣΔ	1.23 ps	1.21 ps	1.6 %

Table 3-1 shows accurate correlation between both simulation methods, allowing RMS jitter to be modelled within 2% of error between the time and the frequency domain analysis.

In the time domain we have loop imperfections (mainly due to phase error discretization) which lead to additional jitter at TIE output [Kieffer]. The time model was created in order to estimate the time jitter of the system.

Chapter 4: Generation of colored noise patterns with Gaussian Jitter distribution

Table of Figures

Table of Tables

4.1 Introduction

In this chapter we will concentrate our study only on unbounded & uncorrelated jitter, also called Gaussian random noise or Random Jitter (RJ).

RJ can have different noise properties. It can have a white (spectrally flat) or colored (spectrally curved) noise profile. Many approaches have been developed based on white noise generation [Endo] - [Kafadar]. Nevertheless theoretical studies on colored noise generation are less common as it depends on dedicated requirements for a specific application [Chow] - [İspir] - [Murch] - [Kasdin].

From laboratory measurements of a transmitter output (TX of a SerDes) we may observe two characteristics:

- From Phase Noise Analyzer measurement (Agilent (Keysight) E5052B) in Fig 4-1 we observe that the jitter has a profile that is spectrally colored.

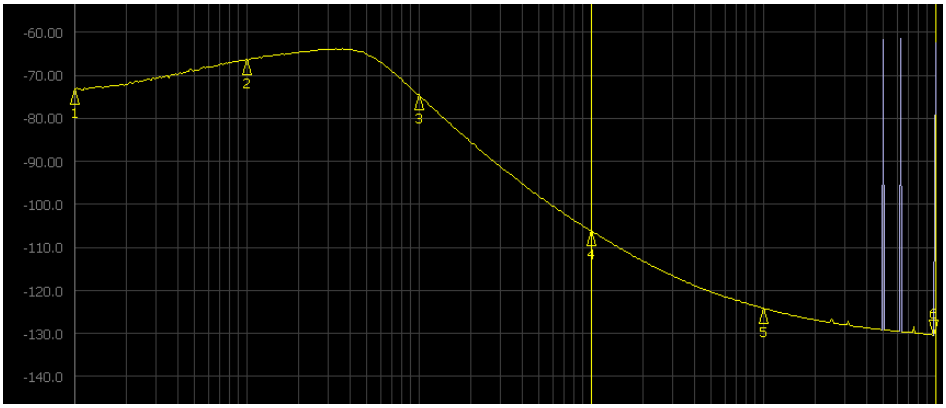


Figure 4-1: Phase Noise Analyzer Measurement

- From J-BERT measurement (N4903B 12.5Gbps Agilent (Keysight) Serial Bert) in Fig 4-2 we observe the unbounded tails of timing jitter which show its Gaussian distribution properties.

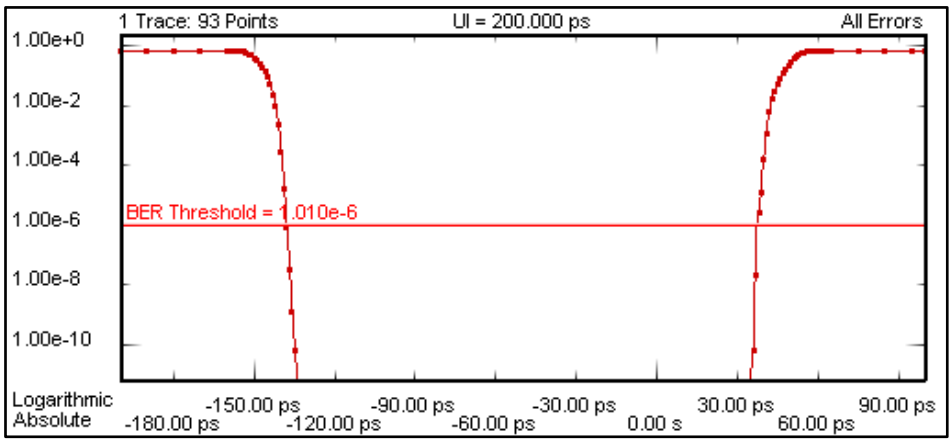


Figure 4-2: J-BERT Measurement

In order to analyze SerDes system characteristics, it is important to generate synthetic colored noise patterns with a Gaussian distribution. The patterns will be used to predict impact of jitter on the system performances with time domain simulation during the design verification phase.

In this chapter we present a different method for generating synthetic patterns with Gaussian distribution from colored noise PSD profiles.

In section 4.2 of this chapter we describe our criteria to decide whether or not a generated white or colored noise pattern has a Gaussian distribution.

In section 4.3, we generate white noise patterns with a Gaussian distribution, with the method explained in 3.4.1. We will also validate our criteria used to decide whether or not the generated noise patterns have Gaussian distribution properties.

In section 4.4 we describe several methods from the literature for generating colored noise patterns. We will determine, based on our criteria, if these methods match our expectations in term of noise power profile and Gaussian jitter distribution.

Section 4.5 describes our novel method for generating colored noise patterns with Gaussian distributions.

Results and conclusions are given in section 4.6.

4.2 Definition of criteria for colored noise with Gaussian distribution

In this section we describe different criteria for determining whether or not the generated noise patterns have Gaussian distribution properties. This will be done in two steps:

The first step is to process the Power Spectral Density (PSD) of the TIE jitter in order to define the type of noise profile (white or colored).

The second step evaluates the maximum acceptable error between a Cumulated Density Function (CDF) from a synthetic pattern with Gaussian jitter distribution (obtained from a signal with white noise PSD) and the CDF from the ideal Gaussian error function (4.1). This maximum error will then be used to define the template to determine whether or not a synthetic pattern has Gaussian distribution properties.

$$Error_{fct} = \frac{1}{2} + \frac{1}{2}\text{erf}(\frac{x}{\sigma\sqrt{2}}) \quad (4.1)$$

With $erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$ (4.2)

Step 1:

Once a TIE jitter pattern is generated, the first step is to calculate its PSD using (4.3).

$$\Phi_{lin}(\omega) = \int_{-\infty}^{+\infty} \varphi(\tau)e^{-j\omega t}dt \quad (4.3)$$

The Power Spectral Density $\boldsymbol{\Phi(\omega)}$ of the signal $\boldsymbol{x(t)}$, is the Fourier Transform of the autocorrelation function $\boldsymbol{\varphi(\tau)}$.

The power of the PSD is converted to dB using (4.4).

$$\Phi_{dB}(\omega) = 10 * log_{10}(\frac{|\Phi_{lin}(\omega)|}{\Phi_{1ps}}) \quad (4.4)$$

with $\sqrt{\int_0^{+\infty}(\Phi_{1ps})^2 d\omega} = 1ps$ (4.5)

This allows the noise profile to be better visualized and to determine whether it is white or colored noise. An example of a white and colored noise profile is shown in Fig 4-3.

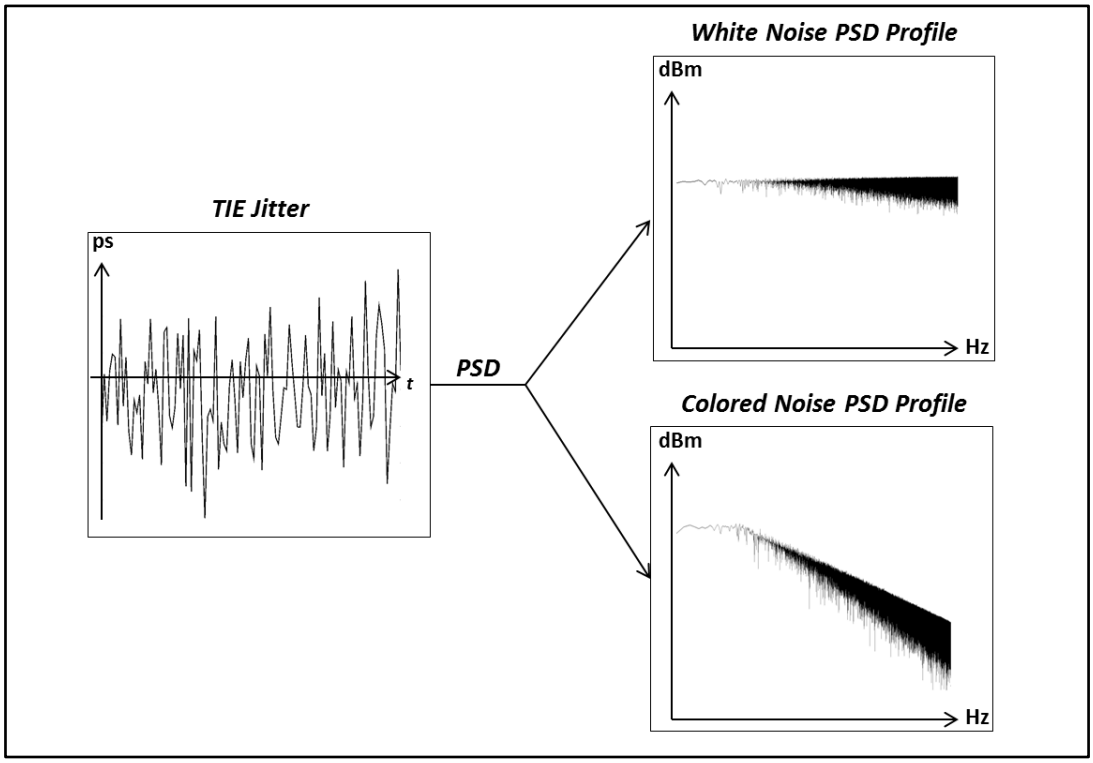


Figure 4-3: Example of a white and colored noise profile

Step 2:

The next step is to calculate the CDF & the Complementary Cumulated Density Function (CCDF) of the TIE jitter. In order to calculate the CDF & the CCDF of the TIE, we need to plot its Probability Density Function (PDF), as shown in Fig 4-4.

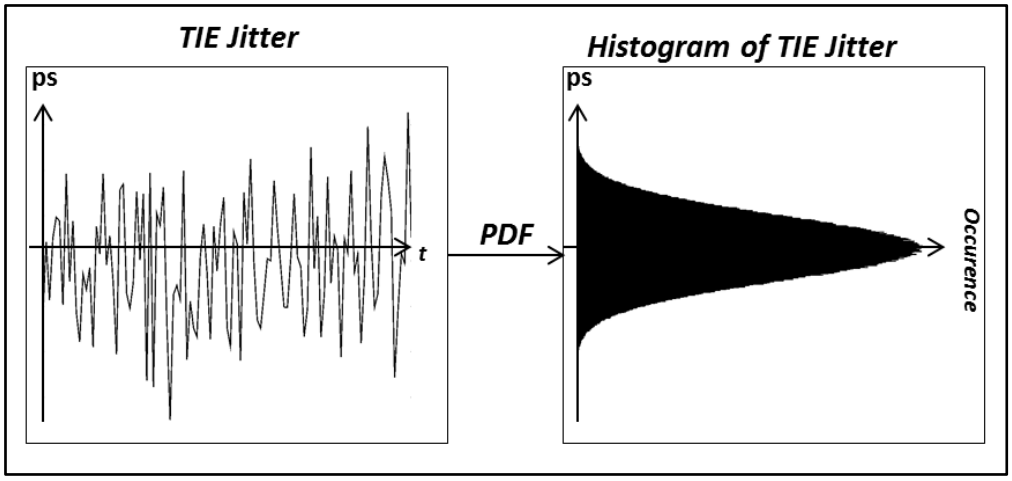


Figure 4-4: Histogram of PDF of the TIE Jitter

From the PDF we can calculate the CDF & CCDF of the Jitter. The CDF is calculated by integrating the PDF curve, as shown in Fig 4-5.

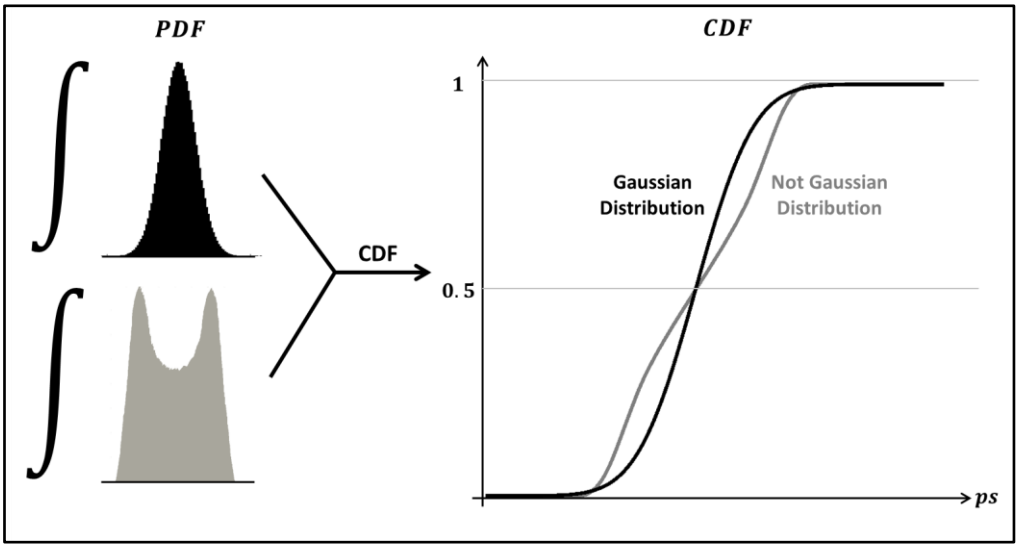


Figure 4-5: CDF Calculation from PDF

The CCDF is calculated using (4.6).

$$CCDF = 1 - CDF \tag{4.6}$$

The linear scale allows the global distribution to be properly represented. Nevertheless, it is there difficult to compare the tails of the distribution on a linear scale.

This is why the calculated CDF and CCDF are transposed to a logarithmic scale, as shown in Fig 4-6. The logarithmic scale allows the tails of the distribution to be properly compared.

The CDF curve (plotted using both linear and logarithmic scales) and the CCDF curve (plotted using the logarithmic scale only) of the TIE jitter are compared to the template based on the Gauss error function (4.1) in order to determine whether or not it has Gaussian distribution properties.

If the CDF and the CCDF of the TIE jitter from the generated synthetic pattern lies within the template (Fig 4-6.a) then we determine that the synthetic pattern has Gaussian distribution properties. Otherwise, the synthetic pattern is set as non-Gaussian distribution, as shown in Fig 4-6.b.

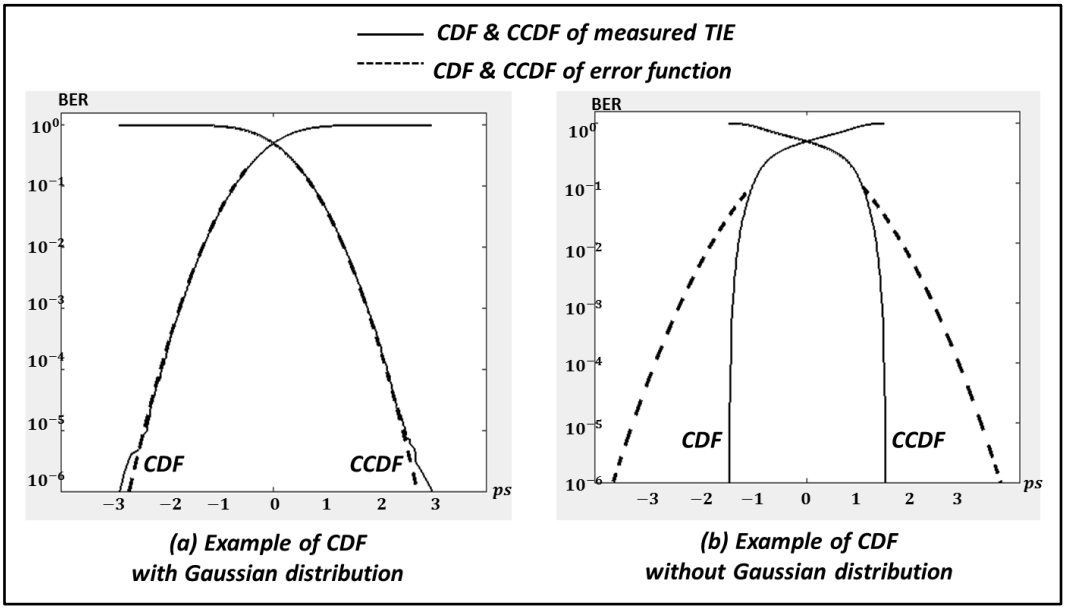


Figure 4-6: Log. scale CDF & CCDF example of a Gaussian and not Gaussian distribution

To define the template used to determine whether or not a CDF has Gaussian distribution properties, we generated synthetic pattern with 10^6 values using pseudo random pattern generation techniques. From [Kang] - [Lee], we may claim that the pseudo random sequence generation we use for simulation has Gaussian distribution properties and white PSD profile.

For the CDF on a linear scale, we define the maximum error used for the template between 1% and 99% of the CDF.

For the CDF on a logarithmic scale we define the maximum error used for the template between BER from 10^{-2} to 10^{-6}.

The standard deviation of the error σ_{error} (in fs) is calculated for a total random jitter with a standard deviation of $1ps$. This level is chosen arbitrarily, then as the error σ_{error} is calculated from this value, it can be scaled to any other RMS jitter values.

In order to estimate properly σ_{error} between a simulated CDF and its ideal Gaussian error function (4.1), we perform 1000 simulations. A template of $\pm 4\sigma_{error}$ is then defined around the Gauss error function. Consequently, 99.99% of synthetic patterns with Gaussian distribution properties will be detected positively by our criteria. For the sake of clarity, we provide the template range of $\pm 4\sigma_{error}$ for a limited set of values in Table 4-1.

Table 4-1: Maximum error in fs to have Gaussian distribution

Linear Scale CDF (Fig 4-6.a)		**Logarithmic Scale CDF (Fig 4-6.b)**	
Percentage	$\pm 4\,\sigma_{error}$	***BER***	$\pm 4\,\sigma_{error}$
20 %	± 5.40 fs	10-2	± 14.04 fs
40 %	± 5.04 fs	10-3	± 35.64 fs
50 %	± 4.96 fs	10-4	± 104.80 fs
60 %	± 5.08 fs	10-5	± 288.80 fs
80 %	± 5.36 fs	10-6	± 996.00 fs

These templates are illustrated with grey regions in the following CDF plots for a linear (Fig 4-7.a) and a logarithmic scale (Fig 4-7.b). We have a Gaussian distribution only if the CDF and CCDF curves of the TIE jitter lie within limits shown in grey.

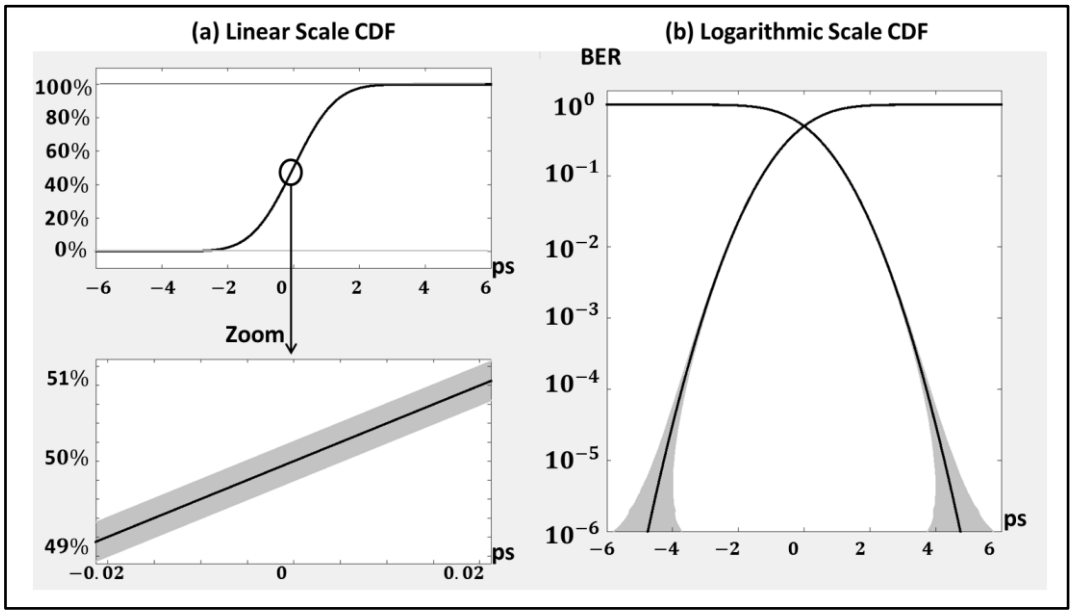


Figure 4-7: CDF Templates to ensure Gaussian distribution for the generated pattern

4.3 White noise generation with Gaussian distribution

4.3.1 White noise patterns generation

In this section we will generate white noise patterns of Gaussian distribution. The steps to generate the time noise jitter (based on 3.4.1) are given below, and are shown in Fig 4-8. This method allows the time jitter to be obtained from the phase noise profile.

Step 1: We first define a white noise profile, $\mathcal{L}(f)$, calculated using (4.7).

$$\mathcal{L}(f) = 10log_{10}(\frac{{A_N}^2}{{A_C}^2}) \tag{4.7}$$

where A_N is the phase noise and A_C the carrier amplitude at a given frequency.

The power *P* of a spur *n*, corresponding to a *Δf* frequency step, around 'f_n' offset frequency, is calculated using (4.8):

$$P_n = \int_{f_n-\frac{1}{2}\Delta f}^{f_n+\frac{1}{2}\Delta f} \mathcal{L}(f)df \qquad \text{(in dBc/Δf)} \tag{4.8}$$

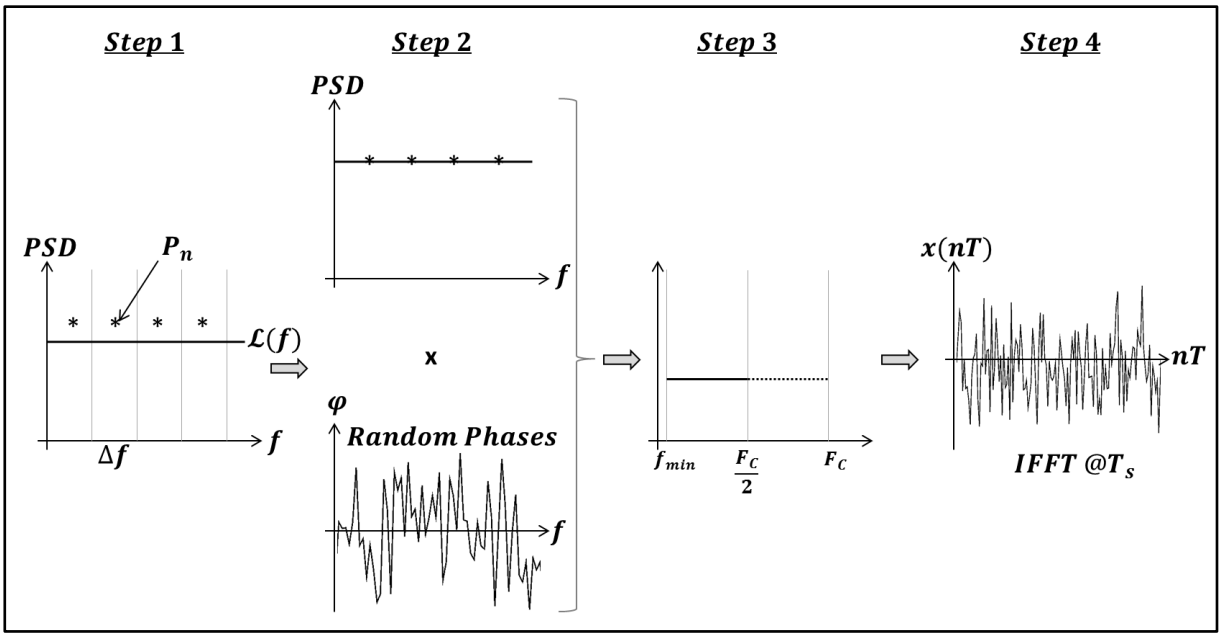


Figure 4-8: Time jitter generation from phase noise profiles.

Step 2: For each value of the power profile, P_n, we associate a random phase with equal probability of lying between 0 and 2π, defined by the formula (4.9).

$$\varphi_n = e^{j\,2\pi\,rand[0:1]} \tag{4.9}$$

By multiplying the power profile defined in (4.8) with the associated random phases in (4.9), we obtain the phase noise, PN_n using (4.10), which is the noise

corresponding to frequency offsets from 0 to $F_c/2$, by step of Δf, with F_c carrier frequency.

$$PN_n = P_n . \varphi_n \quad (4.10)$$

Step 3: To generate the time noise jitter from the frequency noise, the inverse FFT is used. To do this, we complete the vector of the FFT corresponding to frequencies $F_c/2 \rightarrow F_c$, with the conjugate and symmetric of the phase noise profile. In this way, performing the IFFT of the frequency vector results in a time vector with only real parts.

Step 4: The time jitter noise is obtained by calculating the IFFT. Using our criteria we will verify that the generated white noise pattern has Gaussian distribution.

4.3.2 Verifying if generated white noise pattern has Gaussian distribution based on our criteria.

As described in Section 4.2, we have defined several criteria to verify whether or not the generated pattern is a Gaussian distribution. In Fig 4-9 we plot the PSD of the generated time jitter using the above technique. We verify that the generated jitter corresponds to a white noise profile, since its power is spectrally flat.

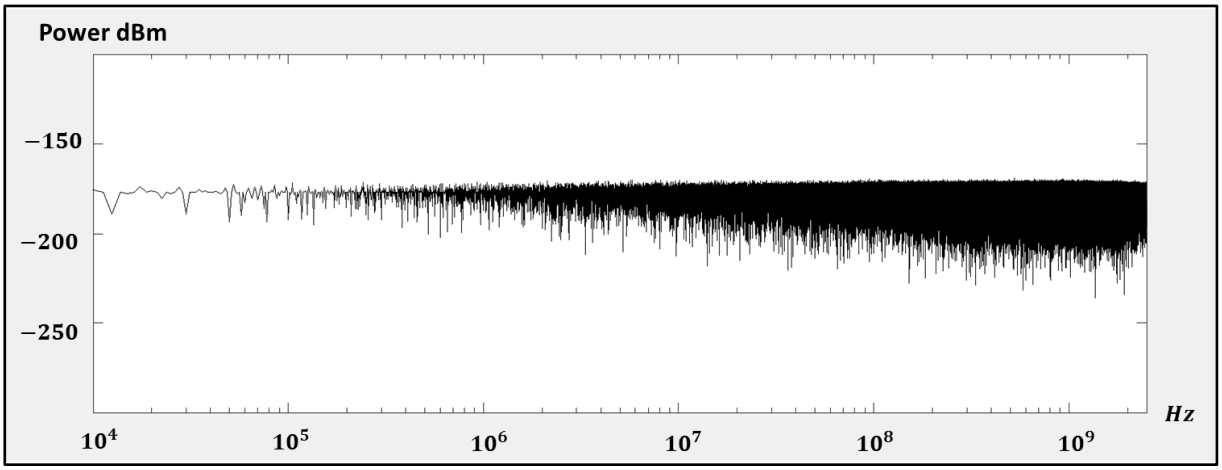


Figure 4-9: White noise profile generated Power

Our criteria are applied after processing its CDF & CCDF on a linear (Fig 4-10.a) and a logarithmic scale (Fig 4-10.b). The template defined in Section 4.2 confirms the Gaussian distribution. Based on our criteria, defined in Section 4.2, the generated white noise pattern with this method has Gaussian distribution properties.

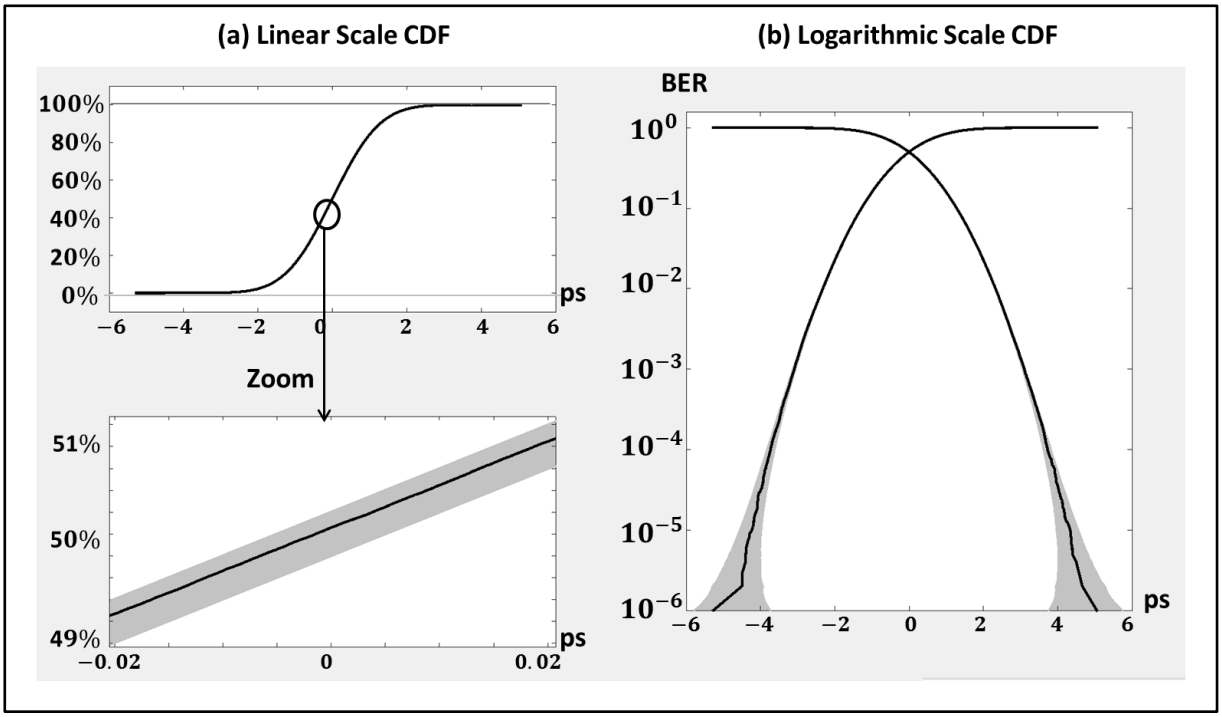


Figure 4-10: CDF & CCDF curves of the generated white noise pattern.

In conclusion, in order to ensure a Gaussian distribution on the tails, a generated pattern should satisfy the following conditions:

- The CDF of the TIE fits the CDF of the Error Function on a linear scale within defined maximum error for a given percentage.
- The CDF & CCDF of the TIE fits the CDF & CCDF of the Error Function on a logarithmic scale within defined maximum error for a given BER.

4.4 Methods from literature for generating colored noise patterns

In this paragraph we study different existing methods for generating colored noise patterns.

4.4.1 Filtering generated white noise pattern

Various approaches in the literature are based on white noise filtering to obtain colored noise patterns [Ferdi] - [Park] - [Zao].

In this section we show that, based on our criteria, such techniques degrade the Gaussian distribution properties of the pattern.

To demonstrate this, we created a -40dB/dec filter in order to filter the white noise pattern with Gaussian distribution that we generated in Section 4.3.1. The Transfer Function of the filter is given in (4.11).

$$TF_{Filter} = \frac{\omega^2}{(\omega+s)^2} \tag{4.11}$$

The filter response is given in Fig 4-11.

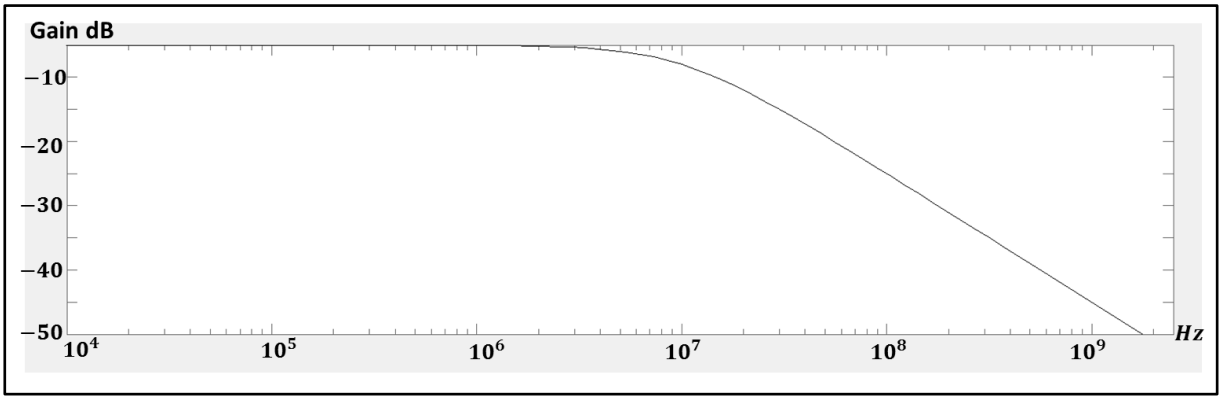


Figure 4-11: Filter Response for filtering white noise

In Fig 4-12 we plot the filtered noise PSD (in dB) and verify that the generated jitter corresponds to a colored noise profile (a slope of -40dB/dec from 10MHz to 2.5GHz).

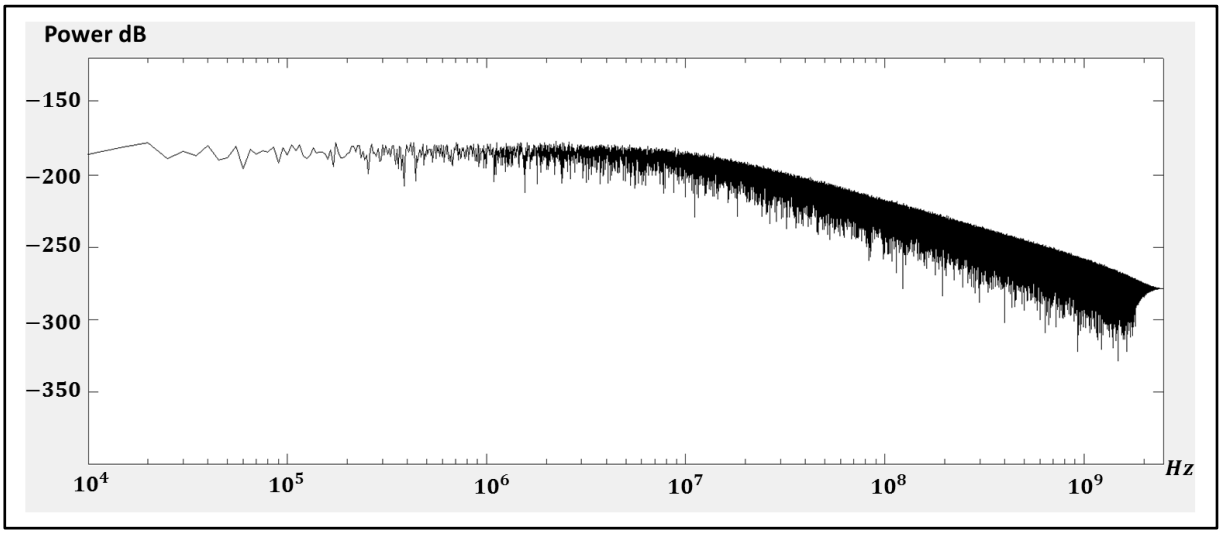


Figure 4-12: Filtered noise power profile.

The CDF & CCDF of the generated Colored Noise pattern are processed and transposed to a linear (Fig 4-13.a) and a logarithmic scale (Fig 4-13.b). The criteria confirming the Gaussian distribution, defined in Section 4.2, are also shown as grey shading in Fig 4-13. We observe that the colored noise generated by filtering the

white noise does not have a Gaussian distribution. The CDF & CCDF curves of the TIE do not fit within the defined limits.

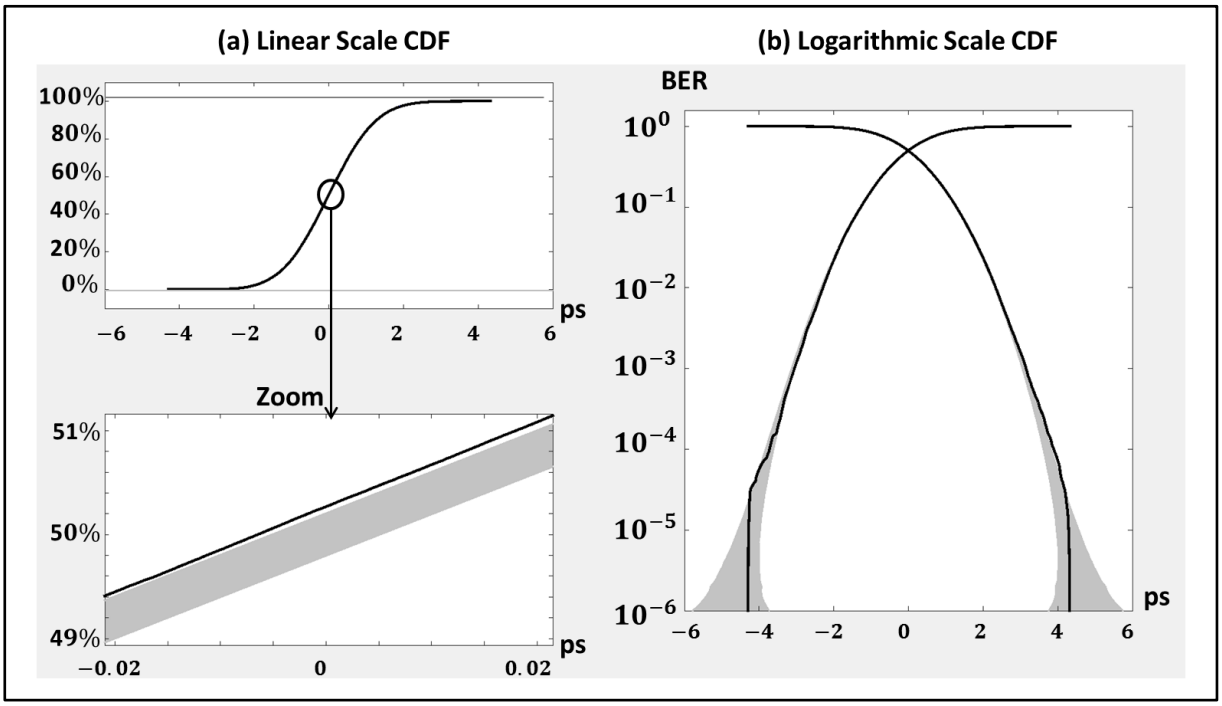


Figure 4-13: CDF & CCDF of Colored Noise generated by Filtering

Since the filtering technique does not meet our criteria for Gaussian distribution colored noise patterns, we go on to explore other methods from the literature.

4.4.2 IIR filtering method

We have tested and implemented the "*IIR Filtering Method*", given in [Kasdin]. Its transfer function (TF) for filtering the random jitter and generating a colored noise profile is given in (4.12):

$$H(z) = \frac{1}{1-\frac{\alpha}{2}z^{-1}-\frac{\frac{\alpha}{2}\left(1-\frac{\alpha}{2}\right)}{2!}z^{-2}+\cdots} \tag{4.12}$$

with α determining the noise power profile of $1/f^{\alpha}$ law.

This method has some specific limitations. It is limited in generating colored noise profiles, for slopes from -20dB/dec to +20dB/dec. This may be sufficient for many applications, but for our SerDes system, the PLL noise profile has a slope of -40dB/dec. This method cannot therefore be used.

Another drawback is that, in increasing slope noise, it adds deterministic spurs to the generated random noise, as shown in the PSD in Fig 4-14. This is due to the numerous poles in the transfer function.

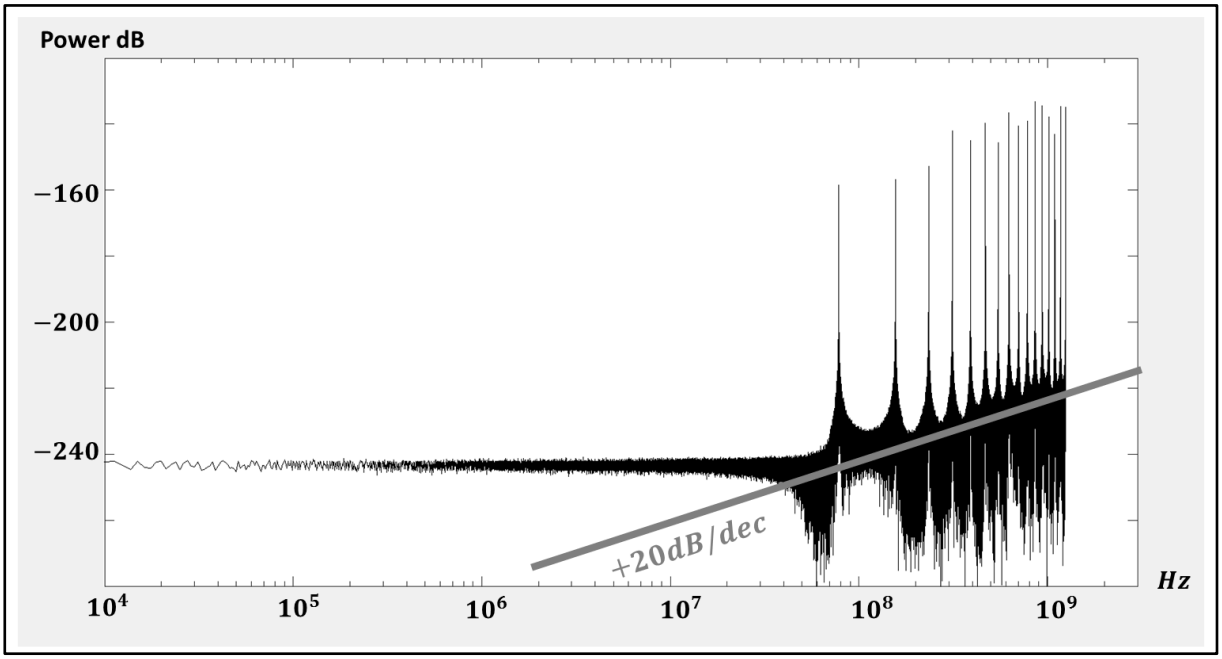


Figure 4-14: DJ Spurs added to random colored noise

Furthermore, for +20dB/dec and -20dB/dec noise profiles, the generated random pattern is bounded on the tails. Its CDF and CCDF are not included within the template limitations defined in Section 4.2, as shown in Fig 4-15.

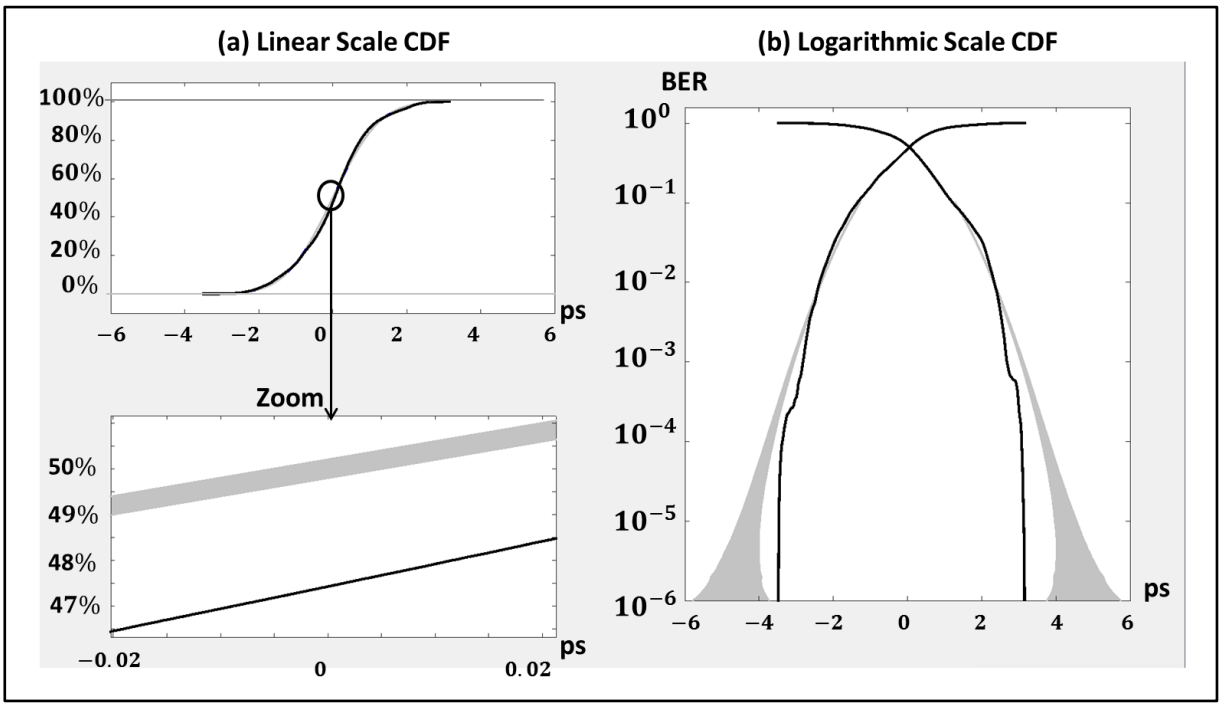


Figure 4-15: Colored noise generated with IIR filtering

We conclude that the distribution of the generated colored noise pattern with this method does not meet our criteria for a Gaussian distribution and may not behave well in time domain simulations.

4.4.3 Colored noise generation with IFFT

We also tested and implemented the "*Colored noise generation with IFFT*" given in [İspir].

We defined a colored noise profile, with a slope of -20dB/dec. We aim to generate a time jitter using an IFFT from a given phase noise profile. We used the method in 3.4.1 to generate the time jitter from the defined profile of -20dB/dec slope. The results of the PSD of the generated noise using this method are shown in Fig 4-16.

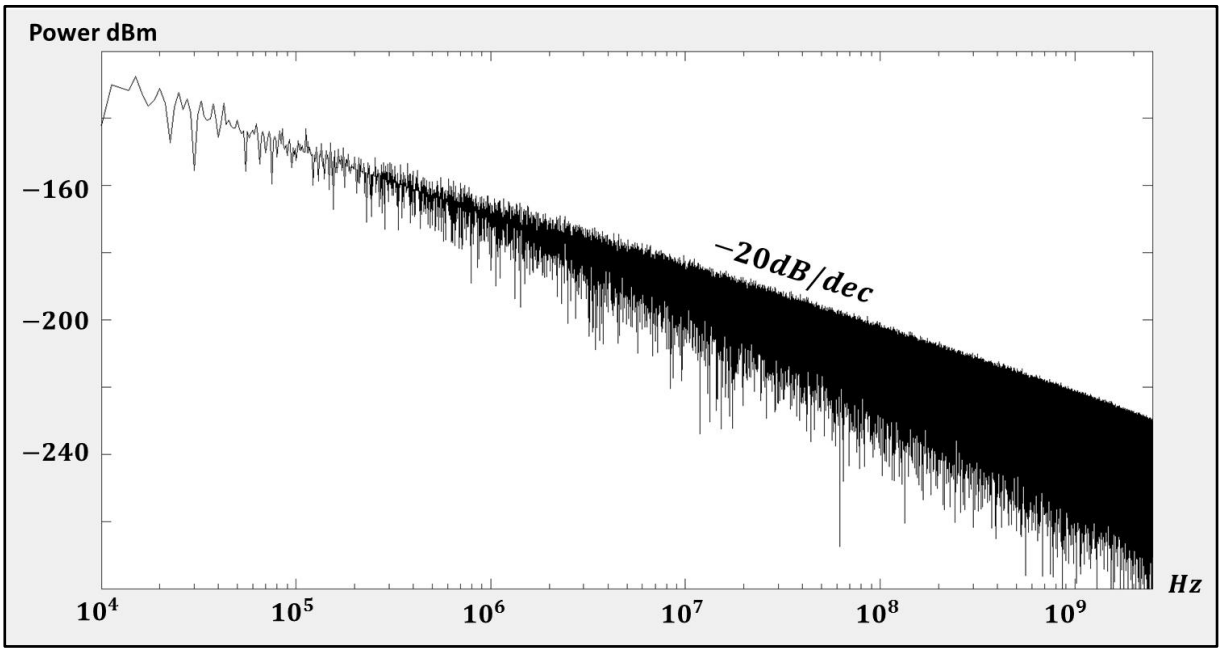


Figure 4-16: PSD of a -20dB/dec generated noise profile

The CDF and CCDF are transposed onto a linear (Fig 4-17.a) and a logarithmic scale (Fig 4-17.b). We observe that the CDF & CCDF of the generated TIE jitter do not fall within the template limitations, defined in Section 4.2.

We therefore conclude that the noise pattern generated with this method does not have a Gaussian distribution either.

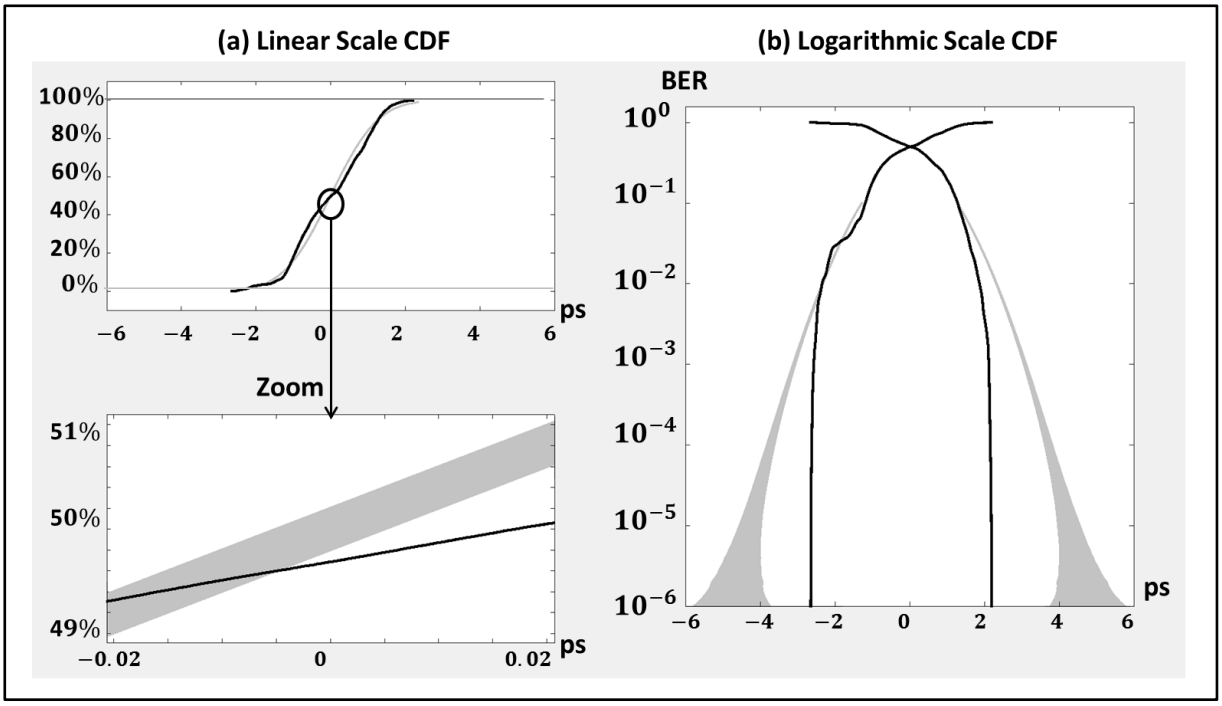


Figure 4-17: Colored noise generated with IFFT

We showed that these three methods do not permit to generate colored noise patterns with a Gaussian distribution. This is why we propose a novel method for generating colored noise with Gaussian distribution.

4.5 New method for generating colored noise with Gaussian distribution

This section is separated into two parts.

The first explains the reasons why we have a bounded effect on the CDF & CCDF tails of the generated TIE jitters with the different methods explained in section 4.4.

The second part describes our method for generating Gaussian distribution colored noise patterns.

The analysis has been performed on low pass filtering PSD profiles (some examples are given in Fig 4-18). We claim that similar analysis and results could be obtained for band pass filtering PSD noise profiles.

Part 1:

First, it is important to understand why we have a bounded effect on the tails instead of a Gaussian distribution, from the time noise jitter obtained using the method described in Section 3.4.1

We generated noise patterns with different colored noise profiles. For each pattern, the total RMS power is kept unchanged. Three examples are shown in Fig 4-18.

We started by setting the filter cut-off frequency (F_{cut}) to 100kHz (Fig 4-18.a), 1MHz (Fig 4-18.b) and 10MHz (Fig 4-18.c).

For each of these profiles, we applied a decreasing slope noise, from -1dB/dec to -40dB/dec, after the cut-off frequencies. Two examples of slopes of -10dB/dec and -30dB/dec are shown in Fig 4-18.

Then, for each of the three F_{cut} frequencies, simulations were performed (with noise slopes from -1dB/dec to -40dB/dec). The RMS power corresponding to low frequencies (4.13) (below the cut-off frequency, shown in black in Fig 4-18) and the RMS power corresponding to high frequencies (4.14) (above the cut-off frequency shown in grey in Fig 4-18) were computed.

$$RMS_{Power\ Low\ Freq} = \int_0^{F_{cut}} 10^{\frac{\mathcal{L}(f)}{10}} \tag{4.13}$$

$$RMS_{Power\ High\ Freq} = \int_{F_{cut}}^{\frac{F_c}{2}} 10^{\frac{\mathcal{L}(f)}{10}} \tag{4.14}$$

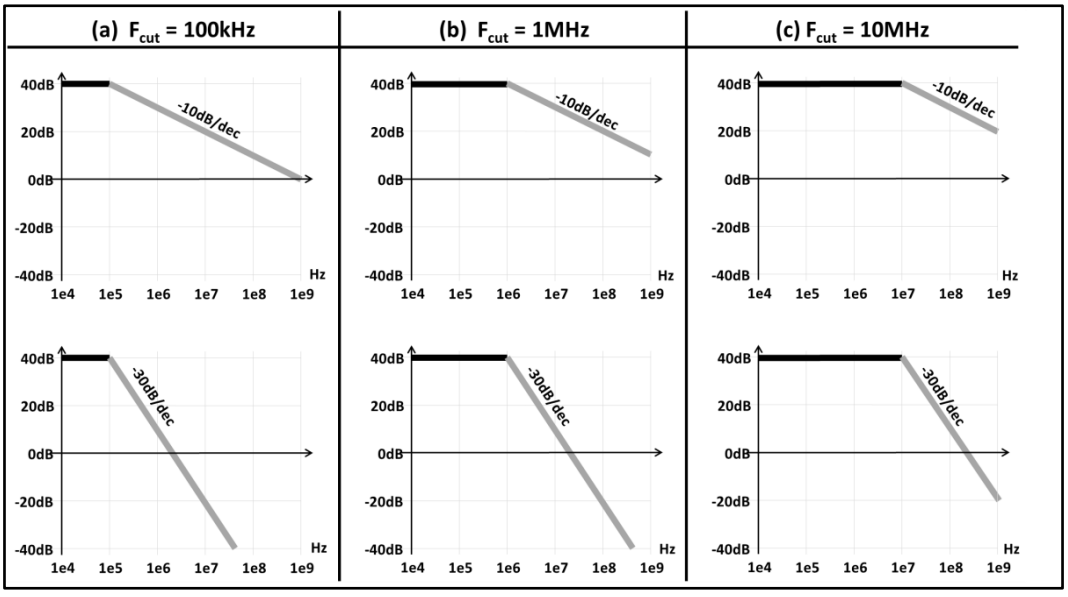


Figure 4-18: Generated types of colored noise profiles.

The RMS powers of low and high frequencies are plotted in Fig 4-19 for the 3 examples shown in Fig 4-18.

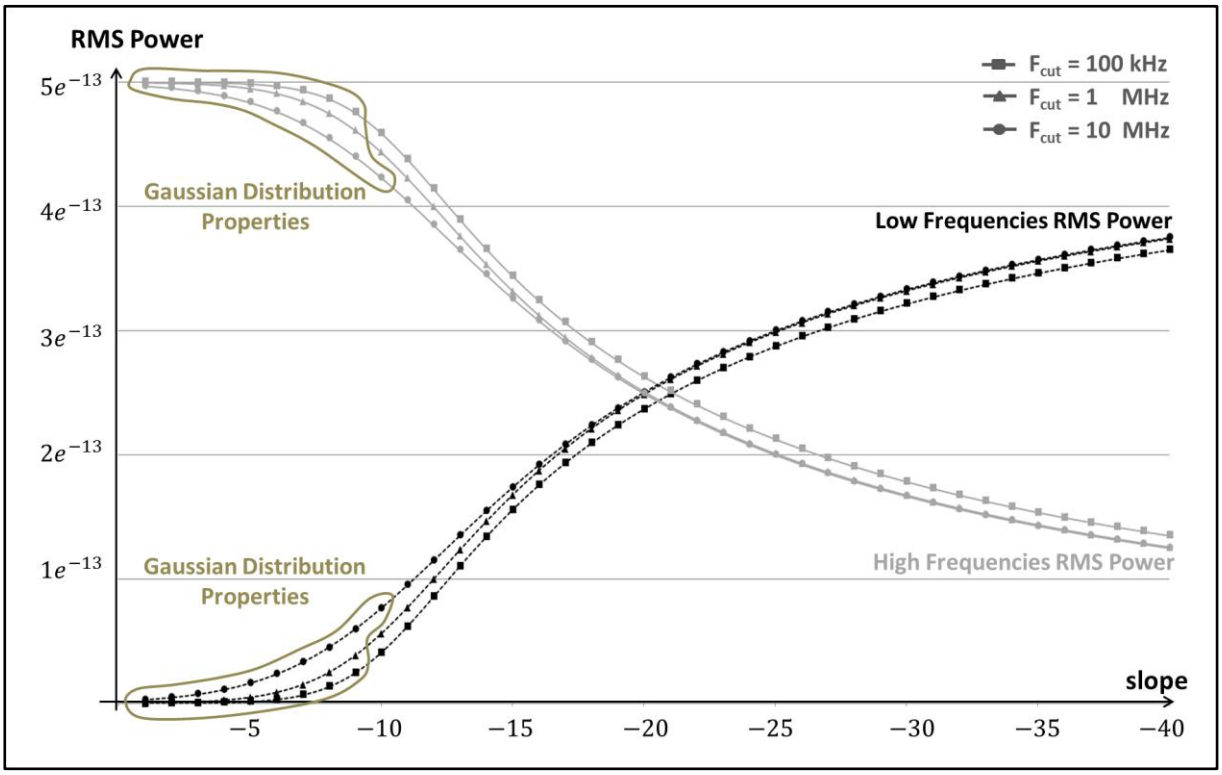


Figure 4-19: RMS power of low and high frequencies

The x-axis of Fig 4-19 represents the slopes from -1dB/dec to -40dB/dec.

Furthermore, Fig 4-19 indicates all the points where synthetic pattern compliance with the properties of a Gaussian distribution has been detected, according to criteria defined in section 4.2.

We remark that we lose the Gaussian distribution properties when the RMS power in the low frequencies starts occupying more than 10% of the total power while the low frequencies occupies less than 1% of the total signal.

Part 2:

Based on previous results and analysis, we propose to increase the noise power over all frequencies by adding a white noise pattern with a power σ_{floor_noise}. We then tune σ_{floor_noise} until we reach the limit condition where our generated pattern is detected to have Gaussian distribution properties.

This method for generating colored noise synthetic patterns with Gaussian distribution properties comprises two steps.

Step 1:

The first step consists in verifying whether or not the generated colored noise pattern has Gaussian distribution, based on our criteria of Section 4.2.

If the generated colored noise pattern does not have Gaussian distribution properties, then Step 2 is performed.

Step 2:

We start by generating two different patterns.

The first one corresponds to a colored noise profile (an example profile with a slope of -40dB/dec is given in Fig 4.20.a).

The second pattern corresponds to a white noise profile (Fig 4.20.b).

For each profile we generate the corresponding time jitter pattern using the method explained in 3.4.1.

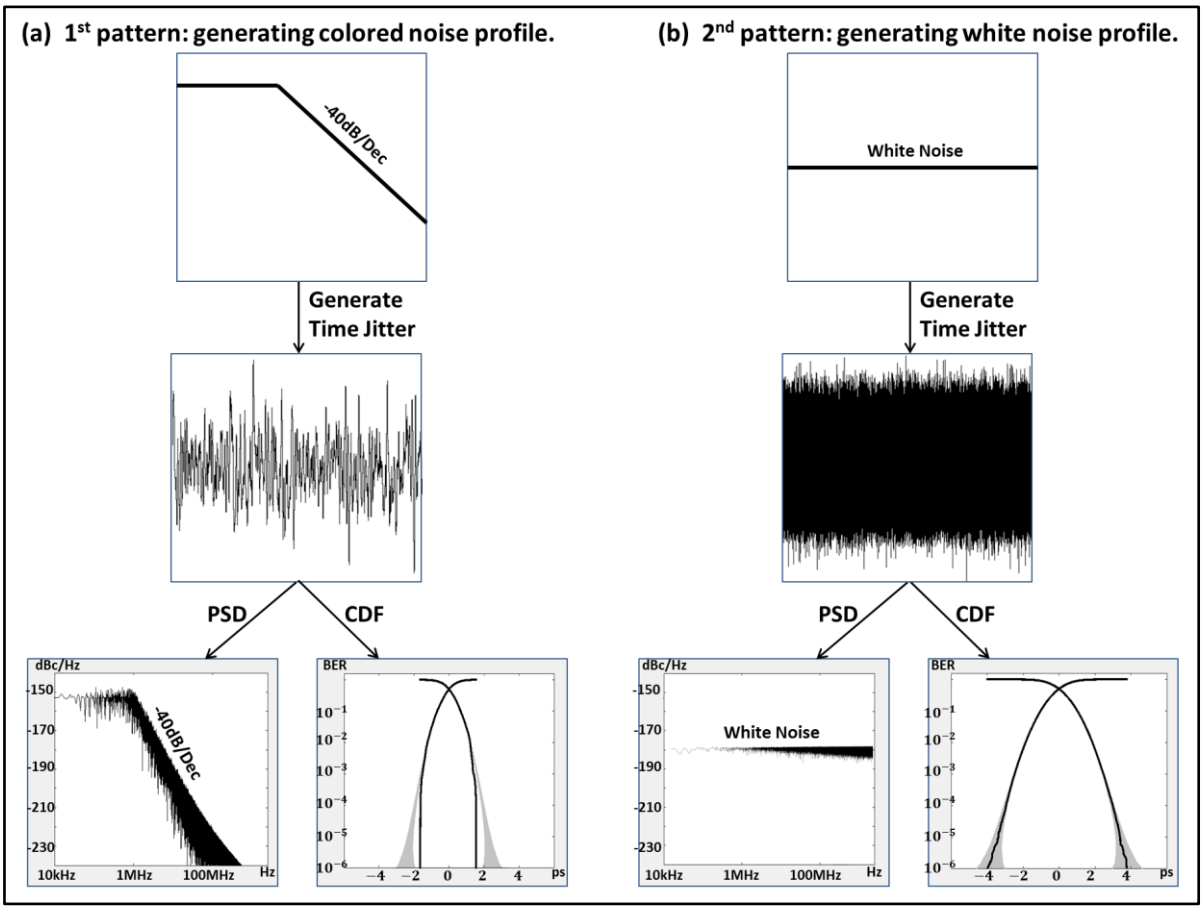


Figure 4-20: Step 1 - Generating white and colored noise profiles.

Then the two generated time jitter patterns are summed, as shown in Fig 4.21. By doing so, we obtain a colored noise profile, with a slope of -40dB/dec and limited by a white noise floor, as shown in the PSD in Fig 4.21.

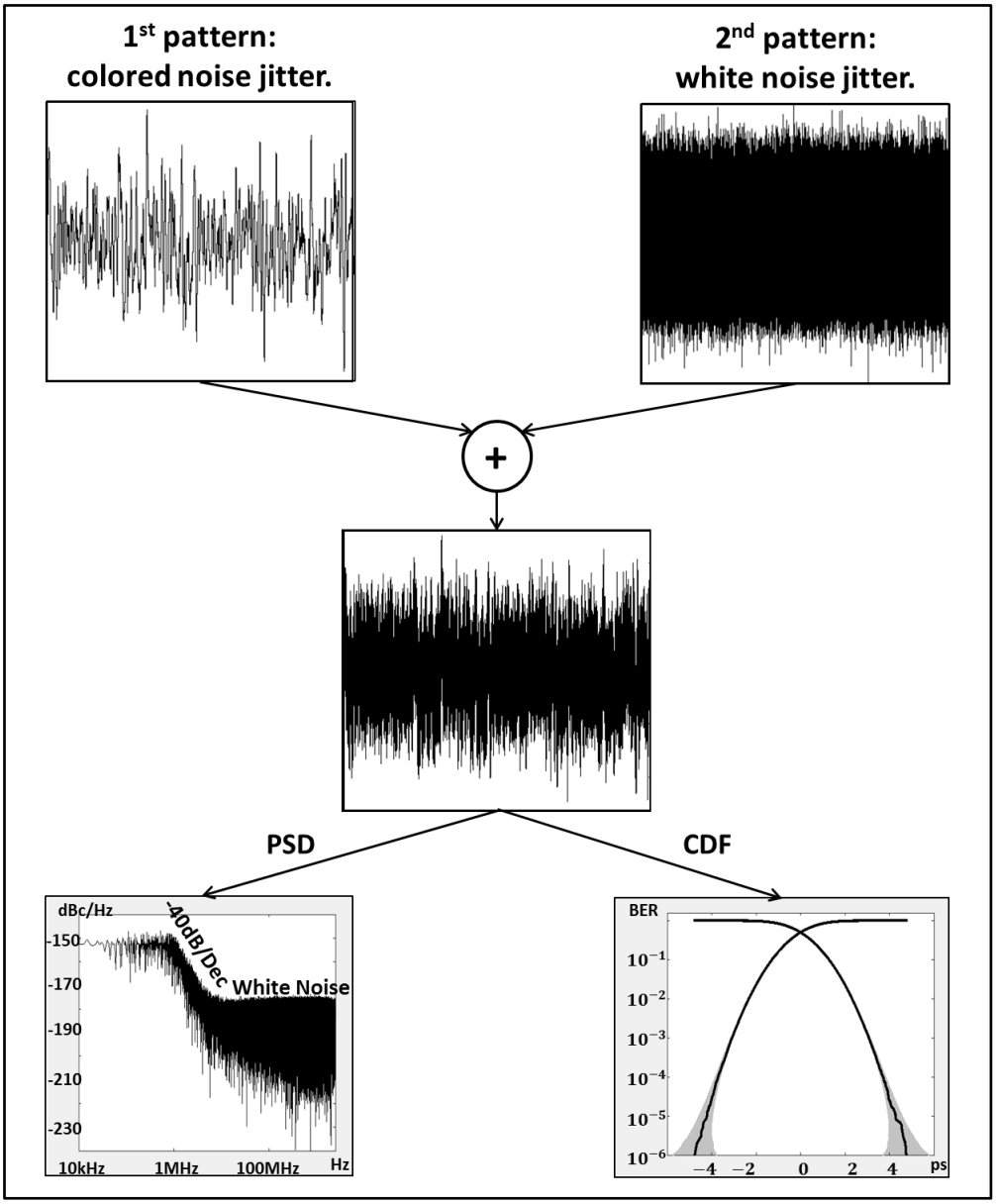


Figure 4-21: Step 2 - Generating colored noise profiles with Gaussian distribution.

The minimum necessary power of the white noise floor is automatically computed, through a dichotomy-like algorithm. This computed minimum white noise power allows Gaussian distribution properties for the generated pattern to be achieved.

From the CDF plot in Fig 4.21, we observe that the jitter corresponding to the new generated colored noise profile is a Gaussian distribution. It fits very well within the limits defined in Section 4.2.

4.6 Results - generating colored noise patterns with Gaussian distribution.

To validate our novel method, several colored noise patterns with different slopes were generated. For each of the generated jitter patterns, we calculated its CDF and CCDF to verify whether or not it has Gaussian distribution properties.

We provide two examples of the generated phase noise PSD profiles (-20dB/dec (Fig 4.22.a) and -40dB/dec (Fig 4.22.b)) with their respective CDF and CCDF shown to lie within our defined template, confirming that both synthetic patterns have Gaussian distribution properties based on our criteria of Section 4.2. Nevertheless this has been done by limiting the colored noise floor profile as seen in the PSD of Fig 4.22.

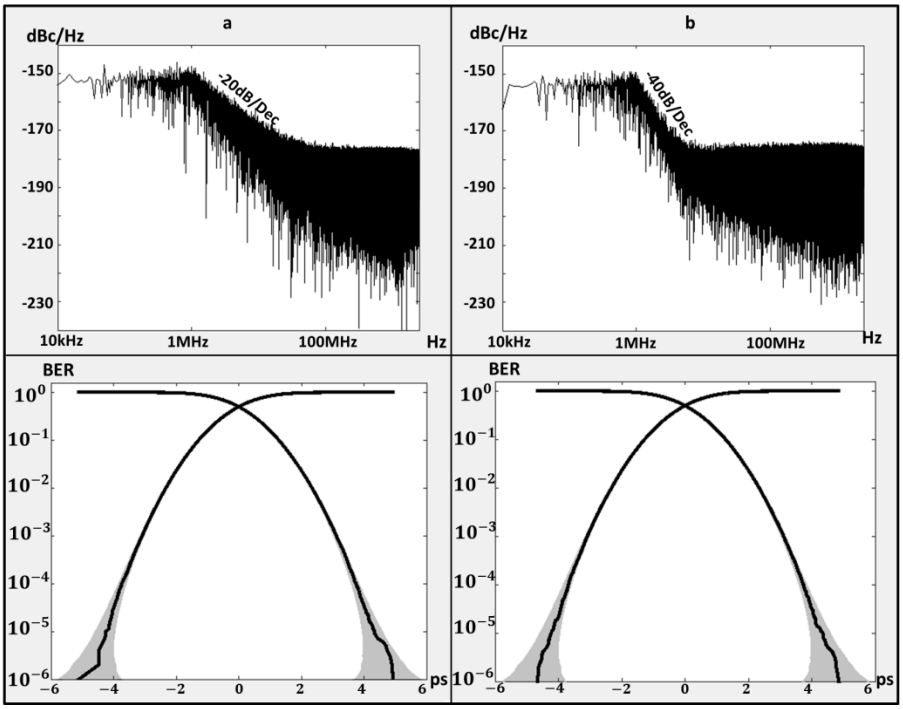


Figure 4-22: Generated colored noise profiles with Gaussian distribution.

4.7 Chapter's conclusion

We propose a novel method for generating colored noise patterns, by computing the optimal trade-off between the noise profile and a Gaussian distribution. Specific criteria are defined to verify whether or not a generated noise pattern has Gaussian distribution properties. The generated colored noise patterns will be used to predict the impact of jitter on SerDes performances during the design verification phase, using time domain simulation. This will allow desired Figure of Merit to be achieved.

Chapter 5: RJ/DJ Jitter Decomposition Technique

Table of Figures

Table of Tables

5.1 Introduction

The TIE can be separated into two major jitter categories: random and deterministic jitter. Random jitter (RJ) is unpredictable timing noise, which typically follows a Gaussian distribution [Dou_1] - [Mistry]. Deterministic jitter (DJ) is predictable and the Peak-to-Peak (PP) value of this jitter is bounded [HFE] - [Sharma].

The standard organizations give specific budgets for RJ and DJ [USB] - [PCIe]. It is then important to be able to properly identify and quantify each type of jitter in the pattern, by decomposing the jitter into RJ and DJ.

As given in Section 4.2, RJ can have different noise properties. It can have a white (spectrally flat) [Endo] - [Kafadar] or colored (spectrally curved) [İspir] - [Murch] - [Kasdin] noise profile.

In order to decompose the TIE jitter and cover all kind of noise types, we introduce a new algorithm, for decomposing jitter into RJ and DJ. The proposed algorithm is to be validated by generated white and colored noise patterns, and laboratory measurements.

Various jitter analysis and decomposition methods have been developed for diagnosing and testing high speed interfaces [Aleksić] - [Dou_2] - [Erb] - [Hong] - [Kho] - [Li] - [Mistry] - [Tzou] - [Wisetphanichkij]. The principal objective of these methods is to have an estimation of the quantity of each type of jitter (RJ / DJ) in the system.

In section 5.2 of this chapter we will compare some of the existing techniques for jitter decomposition, with their limitations.

Section 5.3 explains step by step the novel technique of jitter decomposition developed for model noise estimations.

Section 5.4 gives jitter decomposition results, from generated white and colored noise patterns with Gaussian Jitter Distribution for the Random jitter.

Conclusions and future perspectives are given in section 5.5.

5.2 Various existing RJ/DJ decomposition techniques

In this paragraph we study different existing techniques for jitter decomposition into RJ and DJ.

5.2.1 Time lag correlation technique

The Time lag correlation (TLC) method [Dou_2] is an efficient technique for jitter decomposition, as each component of jitter evolves with time. As RJ is a random variable, it has small correlation with its lagged time series while sinusoidal jitters (SJ) have strong correlation. Decomposition of duty cycle distortion (DCD), RJ and multiple SJ is then possible using this method. However, this technique becomes very complex when total jitter (TJ) contains more jitter components, and will cost longer simulation time to obtain accurate results.

5.2.2 Tail fitting method

The Tail fitting method [Li] is used to correctly identify the three unknown Gaussian model parameters, which are the mean μ (μ_L & μ_R), the standard deviation σ (σ_L & σ_R), and the amplitude A (A_L & A_R) of tail area for left and right distribution as given in Fig 5-1. This method is based on extracting the RMS value (σ) of the RJ from the 2 tails of the Probability Density Function (PDF) and defining the PP value of the DJ by the difference between the two peaks in the PDF. This method does not work properly when the RMS value is comparable to the PP value of DJ since there is only one peak on the PDF.

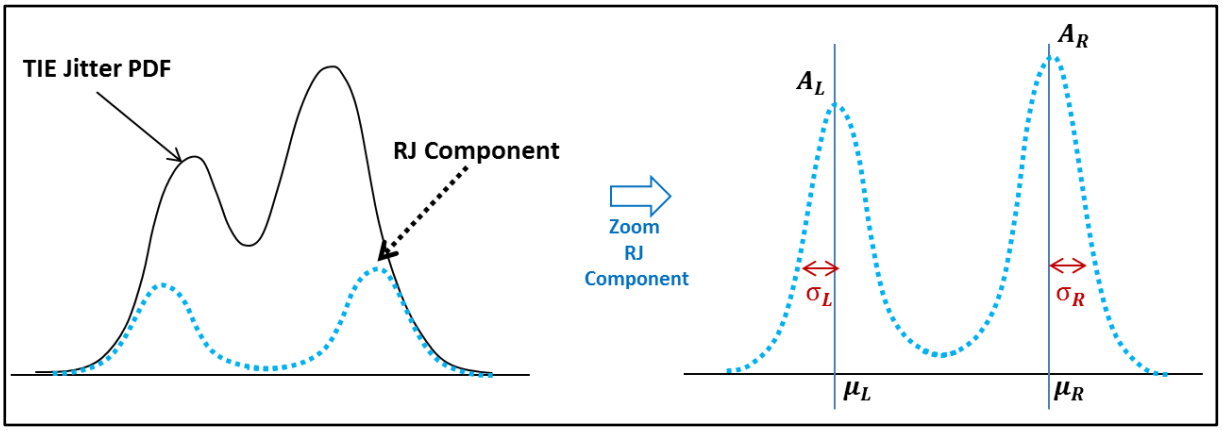


Figure 5-1: Tail Fitting Method for Jitter Decomposition

5.2.3 Dual Dirac Decomposition Method

The Dual-Dirac decomposition method is actually the most frequently used jitter decomposition technique [Aleksić] - [Hong]. This method is similar to the Tail Fitting method [Li]. It is a very quick method for estimating RJ and DJ. It assumes that jitter is separated into RJ which has Gaussian distribution and DJ which is bounded and consists of two Dirac delta functions, as shown in Fig 5-2. [Agilent] - [Li] - [Mistry] - [Tektronix]

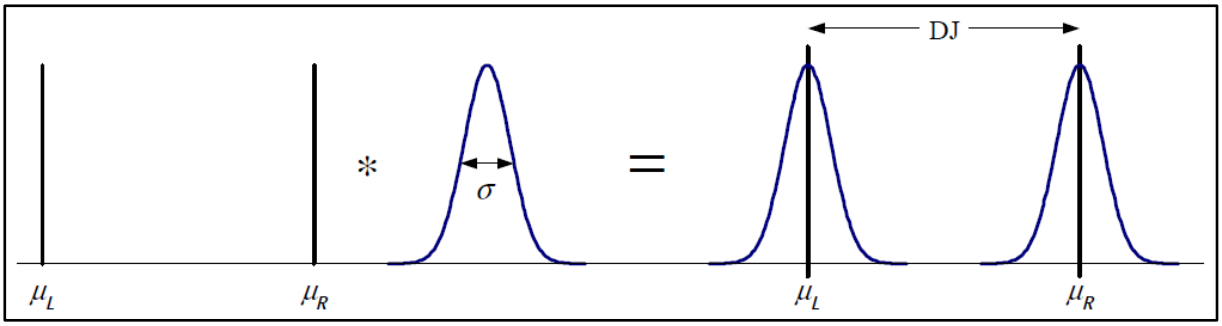


Figure 5-2: Dual-Dirac Method for Jitter Decomposition

Therefore, the dual-Dirac model is a Gaussian approximation to the outer edges of the jitter distribution displaced by $DJ(\delta\delta)$. Once the standard deviation and $DJ(\delta\delta)$ are measured the eye closure at any BER value can be estimated with (5.1) [Agilent] - [Mistry] - [Tektronix]

$$TJ(BER) \cong 2Q_{BER}.\sigma + DJ(\delta\delta) \quad (5.1)$$

where Q_{BER} is calculated from the complementary error function, as given in Table 5-1. [Agilent] - [Tektronix]

Table 5-1: Q_{BER} values, multiplicative constant determining eye closure due to RJ

BER	10^{-10}	10^{-11}	10^{-12}	10^{-13}	10^{-14}
Q_{BER}	6.4	6.7	7.0	7.3	7.6

Nevertheless, the estimated Dual-Dirac deterministic jitter DJ(δδ) with this method is smaller than the real Peak-to-Peak deterministic jitter DJ(p-p) [Tektronix], as shown in Fig 5-3 [Agilent]. This method only permits to have a fast estimation of global decomposition, without enough precision.

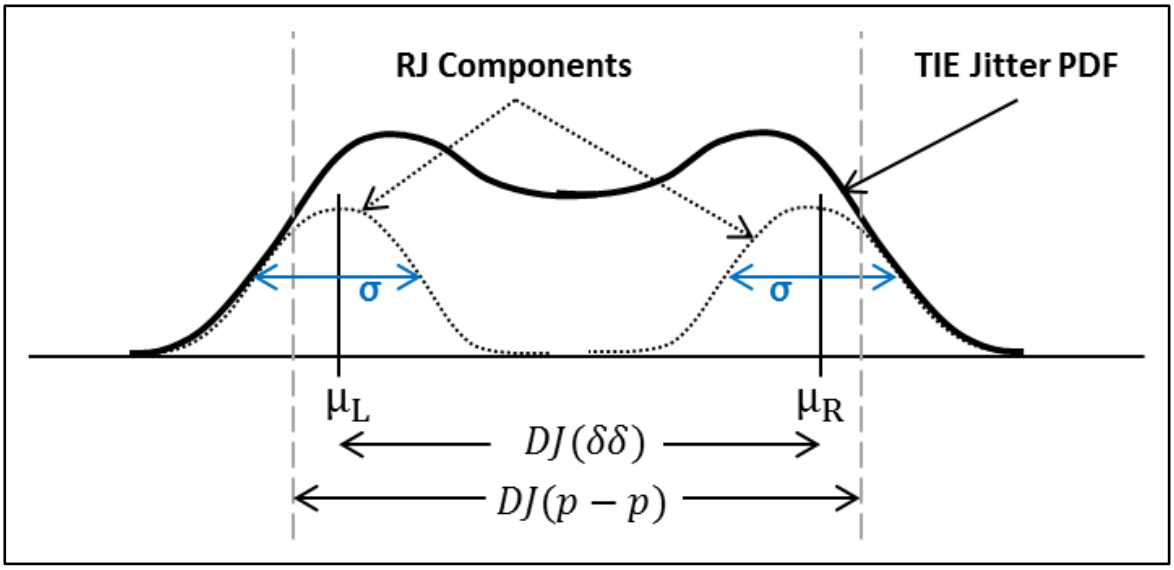


Figure 5-3: Dual-Dirac DJ is smaller than real Peak-to-Peak DJ

5.3 Extended Jitter Decomposition Technique

In order to optimize jitter decomposition results we propose a novel method for decomposing the total TIE jitter into RJ and DJ.

Based on the dual-Dirac method, we multiplied the number of Dirac tones derived from the deterministic jitter estimation, which are convoluted with the Random Jitter as shown in Fig 5-4. This is done in order to better approach the real jitter distribution. The total jitter TIE is described by the sum of all these convolutions.

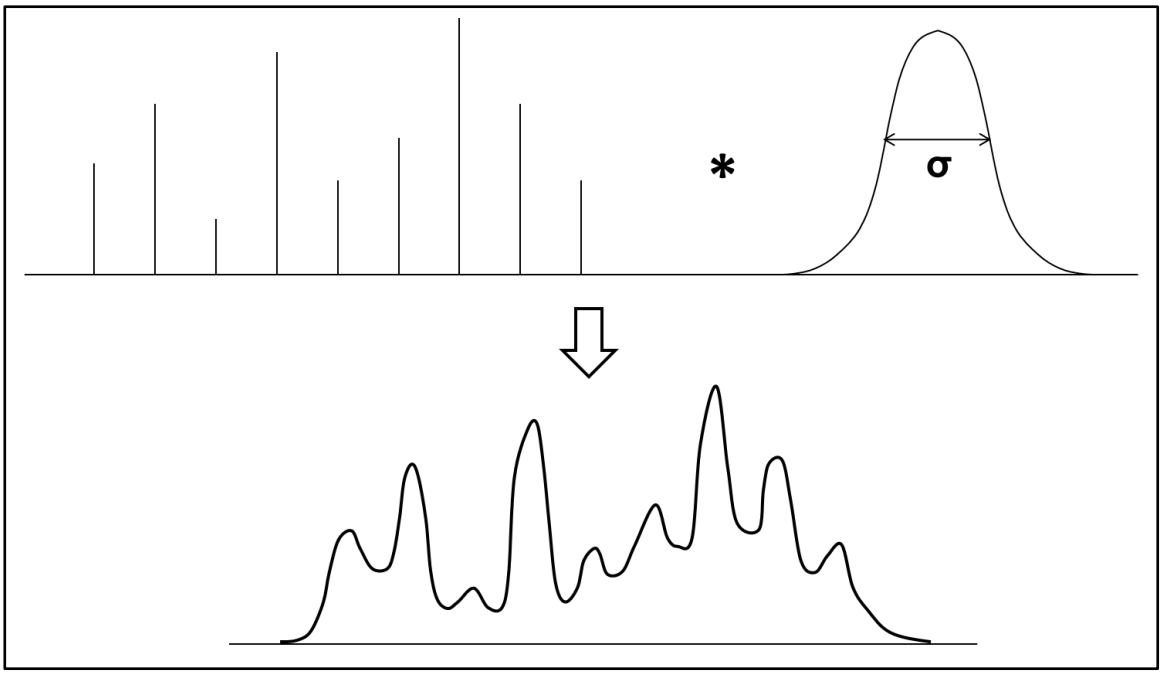


Figure 5-4: Jitter Decomposition proposed method

The principle of the proposed algorithm for jitter decomposition is shown in Fig 5-5. The idea of the method firstly consists in identifying all possible DJ spurs by applying a threshold, which is calculated with a sliding window over all frequencies. Then, from the possible DJ spurs, the algorithm considers more or less spurs as DJ and the others as RJ, and with loop iterations, finds the best match for RJ/DJ separation.

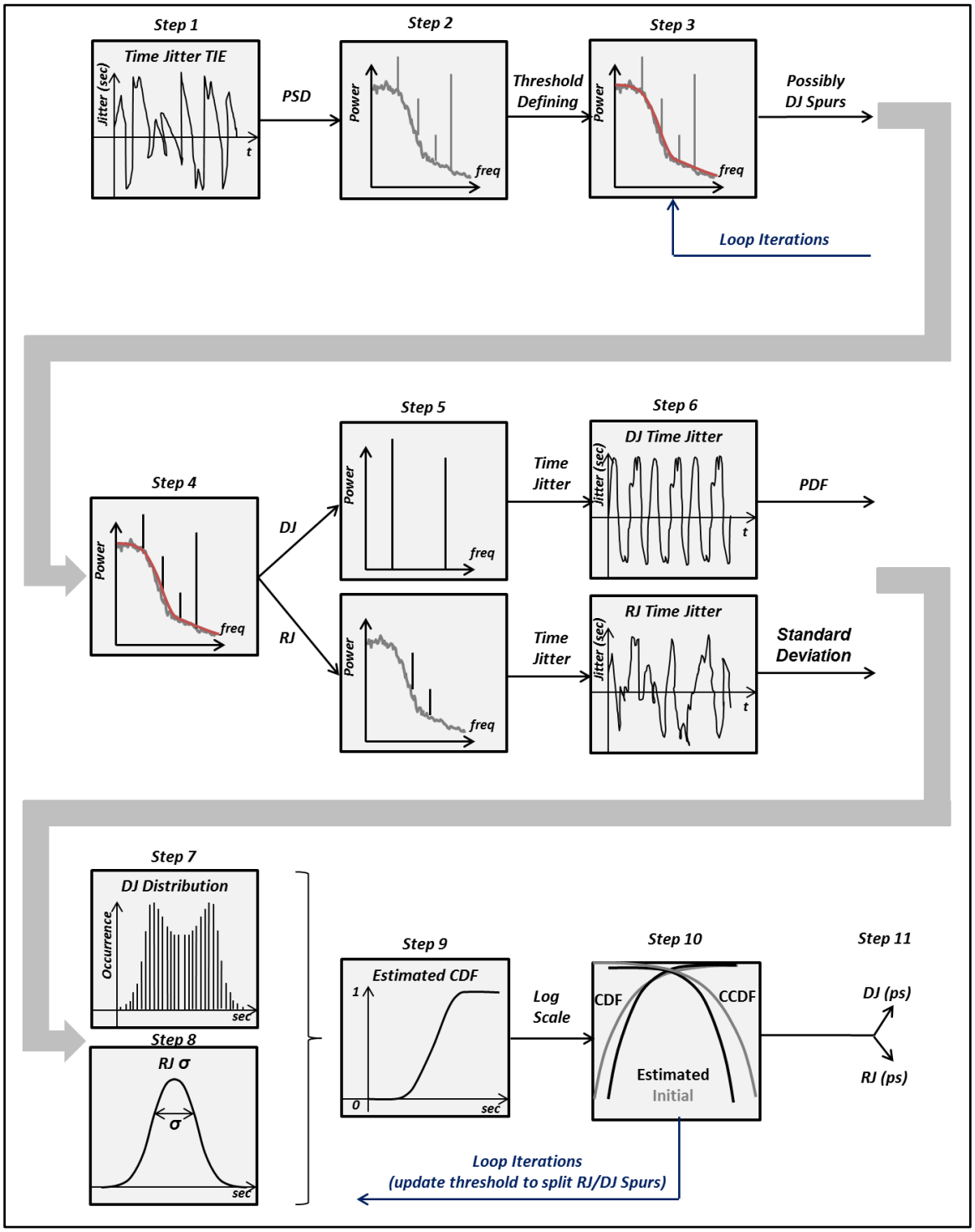
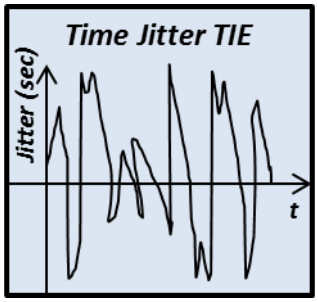


Figure 5-5: Jitter Decomposition Algorithm

There are several steps detailed below, which are applied in order to estimate RJ and DJ quantities of the TIE jitter.

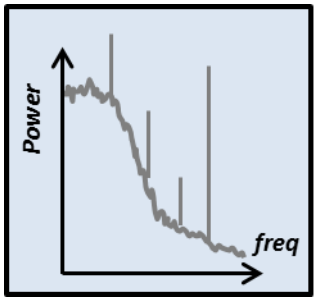
Step1:



First of all, the TIE jitter that will be decomposed is recovered from simulation results (generated noise patterns), or measurements in laboratory. The Initial CDF and CCDF of this TIE are calculated.

They will be compared in Step 10 with the Estimated CDF and CCDF.

Step2:



The PSD of the TIE time jitter is calculated. This is done with (5.1).

$$\Phi_{lin}(\omega) = \int_{-\infty}^{+\infty} \varphi(\tau) e^{-j\omega t} dt \qquad (5.1)$$

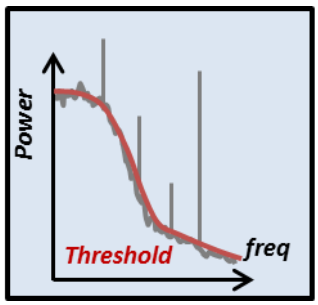
The Power Spectral Density $\Phi(\omega)$ of the signal $x(t)$ is the Fourier Transform of the autocorrelation function $\varphi(\tau)$.

The power of the PSD is converted to dB using (5.2).

$$\Phi_{dB}(\omega) = 10 * log_{10}\left(\frac{|\Phi_{lin}(\omega)|}{\Phi_{1ps}}\right) \qquad (5.2)$$

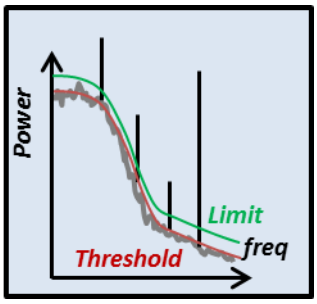
with $\sqrt{\int_0^{+\infty} (\Phi_{1ps})^2 d\omega} = 1ps$ (5.3)

Step3:



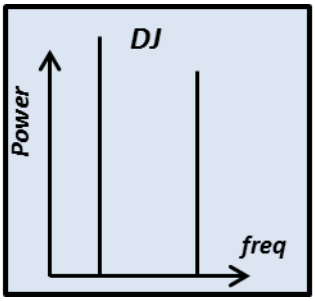
A sliding window is applied over all the frequencies, to define a **threshold**. The threshold corresponds to the 95% of the power value of the noise spurs inside the window.

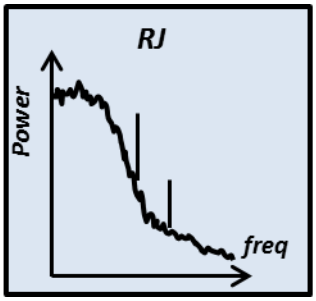
Step4:



To limit possible RJ Spurs, the algorithm defines a **Limit** of 6dB and starts by defining as possible DJ spurs all spurs above this limit. The latter is automatically changed and adapted (increased or decreased) by the algorithm, in order to obtain best match of jitter decomposition, with Loop iterations.

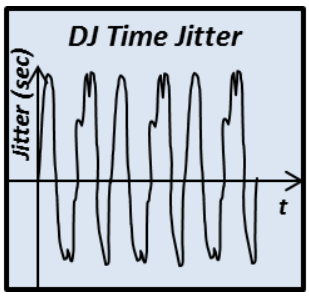
Step5:

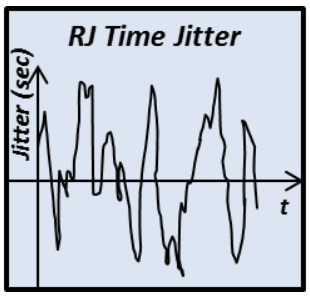




From the possible deterministic spurs, the strongest spurs corresponding to DJ are selected first, and this number is automatically increased or decreased for each loop iteration. The rest of the spurs (smallest ones), are defined as possible random noise spurs.

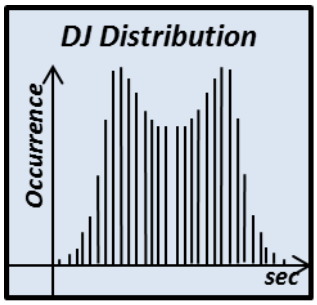
Step6:





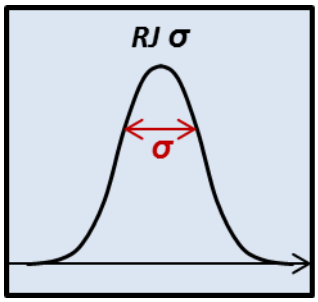
The time jitter is calculated for the spurs considered as DJ, and for the spurs related to RJ. This is done to have the time jitter corresponding only to DJ, and the time jitter corresponding only to RJ.

Step7:



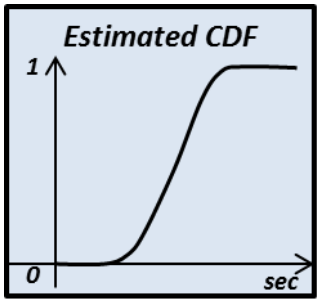
The DJ time noise PDF is calculated from the time jitter corresponding to DJ. This gives the dispersion of the spurs, with power probability.

Step8:



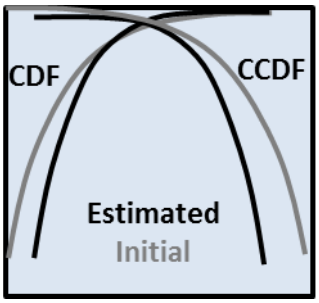
For the RJ time noise, the standard deviation (σ) of the time jitter corresponding to RJ is calculated.

Step9:



By convolution between estimated DJ Diracs and estimated RJ standard deviation (as shown in Fig 5-6), the algorithm calculates the estimated CDF and CCDF.

Step10:



The estimated and initial CDF and CCDF are transposed to a logarithmic scale. The logarithmic scale allows the tails of the distribution to be properly represented. The error between the CDF and CCDF curves of the initial generated TIE and the estimated CDF and CCDF curves is calculated for each of the iterations. Then, the algorithm returns by loop iterations to Step4 to update the limit power value above the threshold.

Step11:

Loop iterations end when all possible DJ spurs are selected as DJ. The result having the minimum error calculated in Step10 over all the loop iterations, gives the best estimation between RJ and DJ.

In Fig 5-6, we give an example of a colored noise profile, with DJ spurs injected with random amplitudes and at random modulation frequencies. We remark that the algorithm has successfully detected all injected DJ spurs.

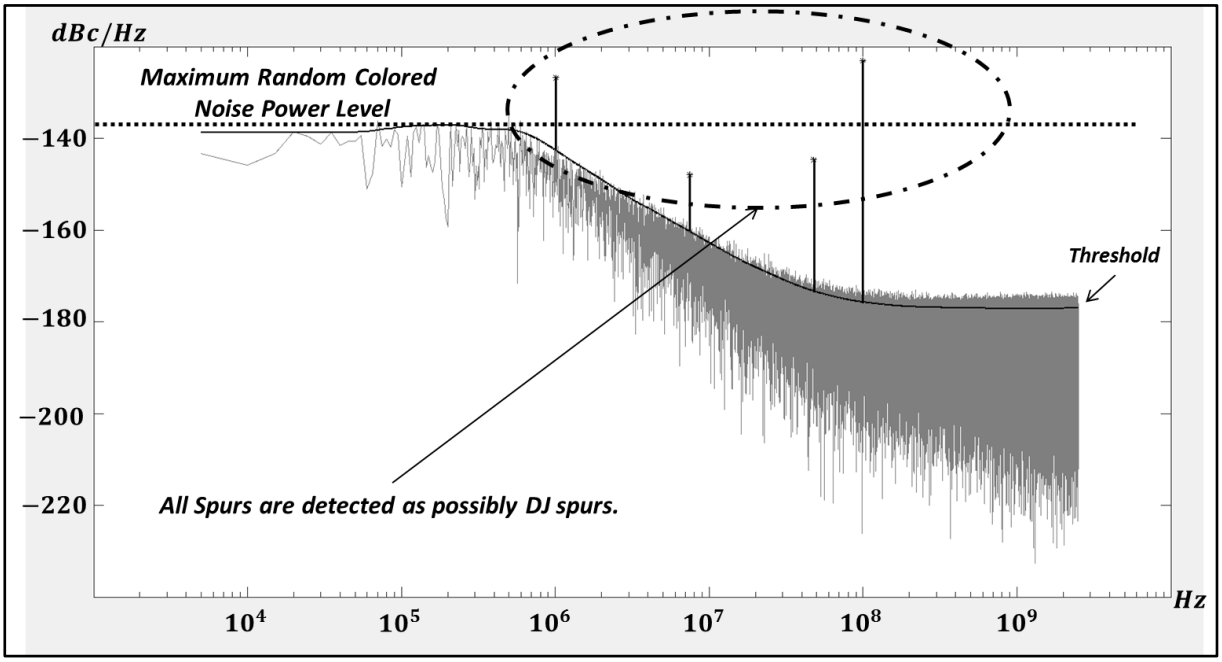


Figure 5-6: All DJ spurs detected by algorithm

5.4 Results of the generated noise patterns

In order to validate the extended algorithm, we generated noisy signals, with different noise profile types for the random jitter: white noise (spectrally flat), and colored noise profiles (from -1dB/Dec to -40dB/Dec slopes).

The synthetic noise patterns were generated with the method explained in Chapter 4.

The noisy patterns are generated with exactly known RJ and DJ quantities. Random number of spurs, amplitudes, and frequencies were used for the DJ. The noise patterns were then used by our algorithm in order to decompose the jitter.

Algorithm worked very well on separating RJ and DJ for any generated white or colored noise. For sake of clarity, we will give below results for one of the noise profiles (-20dB/Dec noise slope).

Different noises, with different RMS values for the RJ (0.1ps, 1ps, 2ps or 3ps), and different PP values for the DJ (0ps, 3ps, 6ps, 9ps, 12ps, 15ps) were generated.

Some results of estimating the RJ RMS quantities are given in Fig 5-7. Others of estimating the DJ PP quantities are given in Fig 5-8.

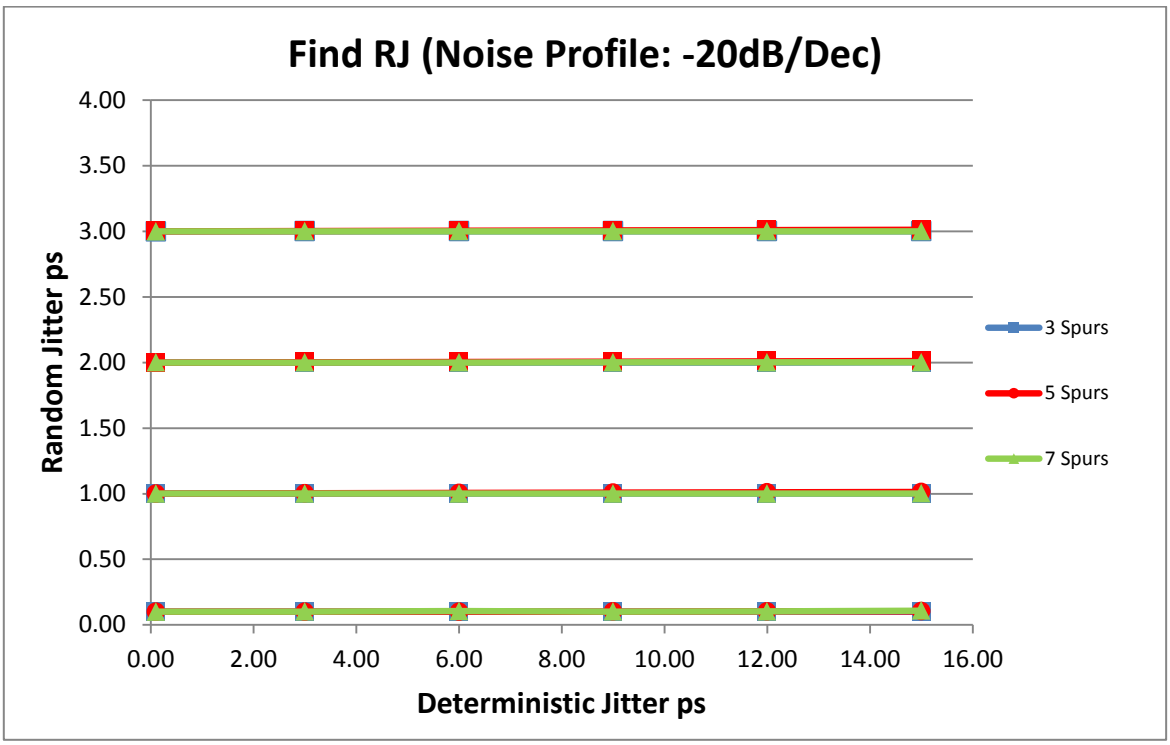


Figure 5-7: RJ for different quantities of generated RJ & DJ Jitter

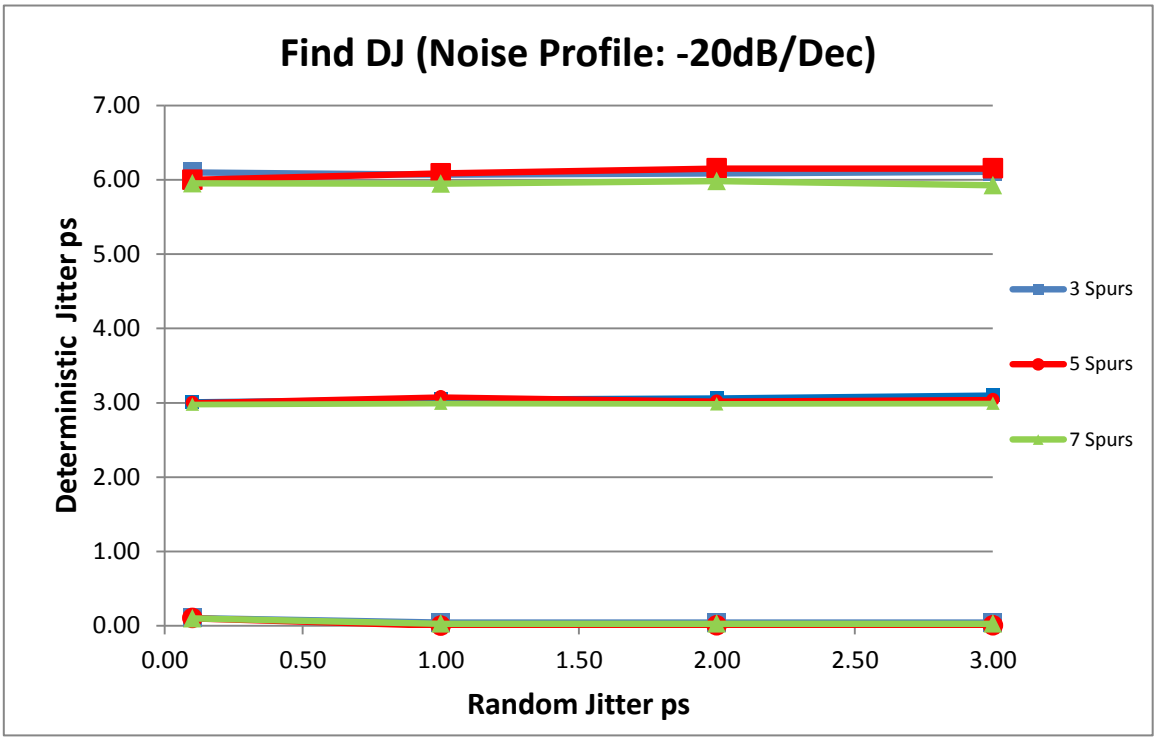


Figure 5-8: DJ for different quantities of generated RJ & DJ Jitter

We remark that the algorithm estimates with precision the generated RJ RMS quantities (Fig 5-7) and DJ PP values (Fig 5-8). This is done for any RJ & DJ quantity injected, and any spur number composing the DJ (examples of 3, 5 and 7 spurs are given in each of the figures).

5.5 Results of the Laboratory Measurements

Laboratory measurements and correlation with the simulation model allow the validity of the algorithm to be confirmed. We used a J-BERT instrument (N4903B 12.5Gbps Agilent Serial Bert) to create a noisy signal. It injects known RJ and DJ to the signal. The waveform is captured by a 50Gsamples/s Tektronix Oscilloscope (DSA720004B).

The samples are used by our algorithm in order to separate RJ and DJ. Some results of injected jitter by the J-BERT and estimated jitter by our algorithm are given in Table II.:

Table 5-2: Correlation between Injected and Estimated Jitter

J-BERT (Jitter Injection)		MODEL (Jitter Estimation)	
Injected RJ	**Injected DJ**	**Estimated RJ**	**Estimated DJ**
0 ps	0 ps	0.50 ps	2.49 ps
0 ps	18 ps	0.50 ps	22.40 ps
0 ps	36 ps	0.51 ps	41.00 ps
0.5 ps	0 ps	0.76 ps	2.59 ps
0.5 ps	18 ps	0.75 ps	22.41 ps
0.5 ps	36 ps	0.75 ps	41.46 ps
1 ps	0 ps	1.19 ps	2.29 ps
1 ps	18 ps	1.15 ps	22.59 ps
1 ps	36 ps	1.14 ps	41.48 ps
1.5 ps	0 ps	1.65 ps	2.56 ps
1.5 ps	18 ps	1.64 ps	22.98 ps
1.5 ps	36 ps	1.60 ps	41.72 ps

We see that jitter estimated by our algorithm correlates with jitter injected by the J-BERT. Our model is “stable” for random jitter estimation, for any value of injected deterministic jitter.

Our algorithm estimates ~0.5ps of RJ when no RJ is injected by the J-BERT. This is J-BERT and oscilloscope natural random jitter.

For example, when ~0.5ps of RJ is injected by the J-BERT, the algorithm estimates ~0.7ps due to Gaussian noise addition in (5.4):

$$Estimated_{RJ} = \sqrt{Injected_{RJ}{}^2 + JBERT_{RJ}{}^2} = \sqrt{0.5^2 + 0.5^2} = 0.7\ ps \quad (5.4)$$

Our algorithm also estimated more DJ than the J-BERT was supposed to have injected. From the biggest spurs that the algorithm considered as DJ, we extracted spurs at frequencies corresponding to the injected spur frequency (100 MHz), and its harmonics. The DJ corresponding to these spurs explains most of the above difference.

We also remarked that some spurs permanently correspond to DJ for any DJ or RJ injected value. These spurs exist at different frequencies and correspond to ~2ps of permanent deterministic jitter. One example of these spurs is given in Fig 5.. This corresponds to the case of 0ps of DJ and 0ps of RJ generated. The DJ spurs correspond to ~2ps of DJ.

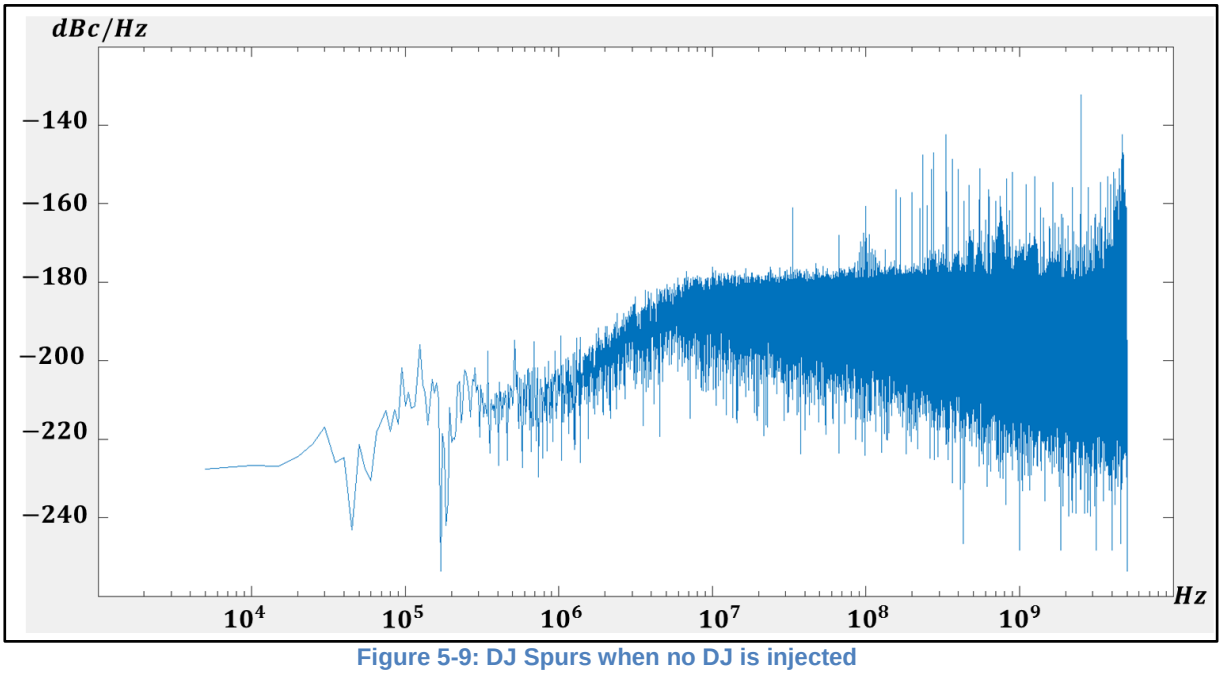


Figure 5-9: DJ Spurs when no DJ is injected

Moreover, we can observe that whatever the RJ injected for a given DJ, the estimated DJ value by the algorithm are close, which proves that our algorithm is well suited for RJ/DJ estimation and extraction.

5.6 Conclusion

We succeeded in proposing a jitter separation method, which estimates with accuracy RJ and DJ quantities in the TIE, for any type of noise profiles (White or Colored Noise Profiles). The algorithm was validated by generated synthetic noise patterns, with exactly known Random and Deterministic jitter.

This method will be useful for designers during design phase, as it will help them to properly constraint the design and better estimate jitter contribution of their blocks. This method decomposes jitter into RJ & DJ and also allows identifying the sources causing DJ to improve overall solution and optimize the performances of the circuit for future versions.

Chapter 6: PCI-Express PLL Specifications

Table of Figures

Table of Tables

6.1 Introduction

In this chapter we will specify a PLL design that has to fit within PCI-Express specifications. This will be done based on the PCI-Express (PCIe) standard requirements. We will propose a method of specifying the PLL, and show a complete example of how we define the PLL bandwidth (BW) and the different dBc/Hz phase noise profiles for each part of the PLL.

In our proposed method, the RMS jitter will be calculated for the 3 different principal noise parts of the PLL, which are shown in Fig 6-1:

- The InBand Noise
- The -20dB/dec slope Noise
- The Floor Noise

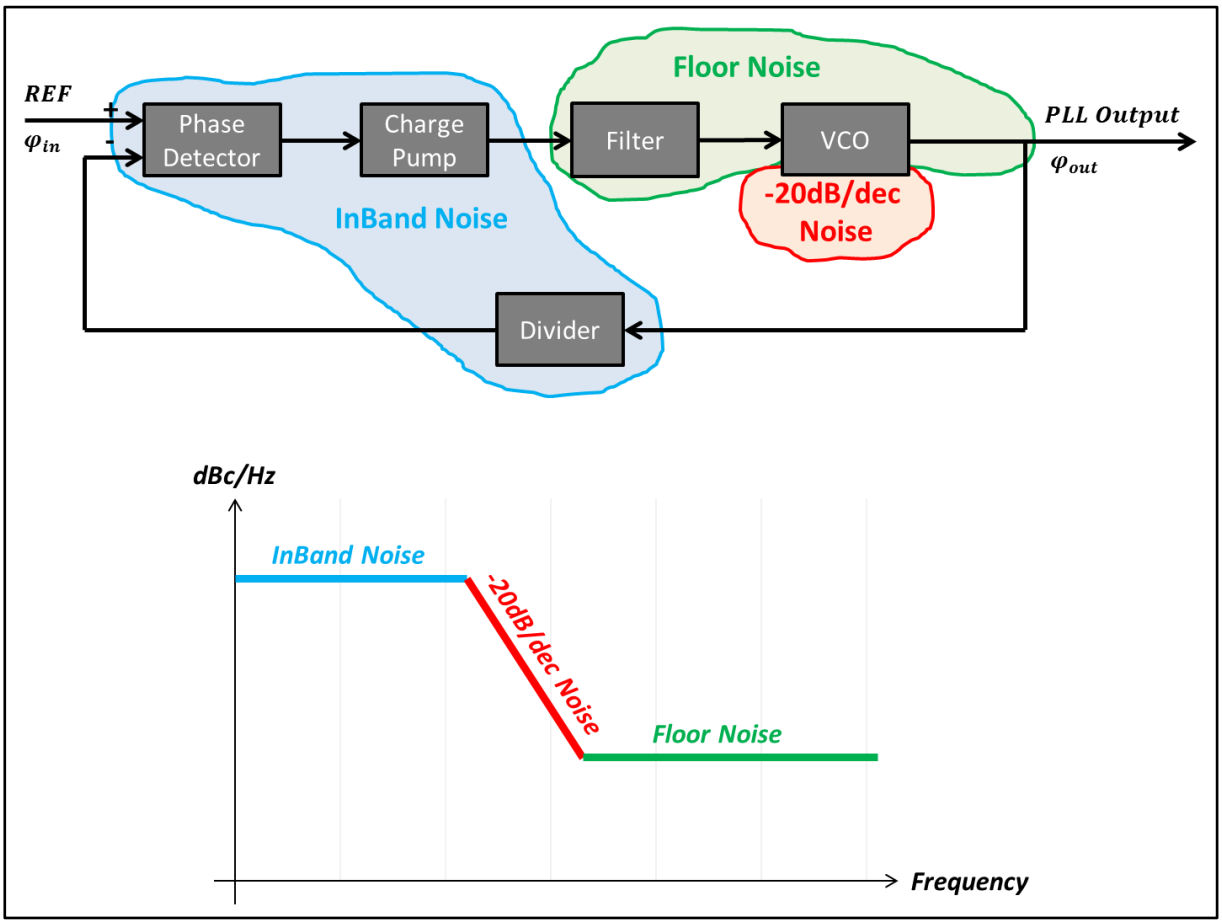


Figure 6-1: PLL Specification Schematic

6.2 PCIe Standard Specifications

The specifications used as starting point are the PCIe Standard Specifications [PCIe]. The targets to reach on the PCI-Express Gen4 Specs are given in Table 6-1. These targets are taken from "*PCI Express® Base Specification Revision 4.0, page 1007, table 9.5*". We remark that we have a total of 0.89ps RMS jitter budget.

Table 6-1: PCI-Express Specifications

Symbol	Parameter Description	16Gbps	Units
T_{TX-UTJ}	Tx uncorrelated total jitter	12.5 (max)	ps PP
T_{TX-RJ}	Tx Random jitter	0.89 (max)	ps RMS

Based on the total RMS jitter budget, we can attribute as example 50% of the total jitter budget to the PLL, and the other 50% to the TX and RX.

So **the Total RMS Jitter attributed to the PLL on these specifications example is 0.445 ps**.

The PCI-Express has a Common clock scenario (which means we have a High Bandwidth PLL). In this case the PC distributes the Reference (Ref) clock (Clk) everywhere, as shown in Fig 6-2.

As defined in PCIe Standards [PCIe], the SSC modulation is embedded within the reference clock. Therefore, the Root Cpx and the End point share same reference clock, with same SSC Modulation of reference clock.

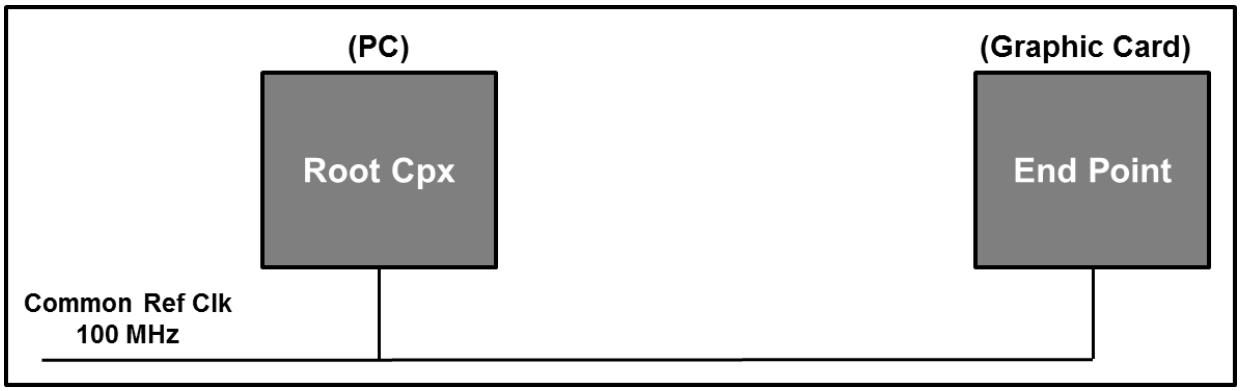


Figure 6-2: PCIe Common Clock

The PCIe subsystem defines the PLL bandwidth limit that each SerDes must comply with:

- To guarantee Root Cpx & End Point are frequency aligned

- To minimize low frequency Phase Noise inside system

  The PLL must respect the Bandwidth parameters given in Table 6-2.

These data are taken from "***PCI Express® Base Specification Revision 4.0, page 1006, table 9.5***"

Table 6-2: PCIe Standards - PLL BW Parameters

Symbol	Parameter description	16Gbps	Units
$BW_{TX-PKG-PLL1}$	Tx PLL banwidth corresponding to $PKG_{TX-PLL1}$	4.0 (max) 2.0 (min)	MHz
$BW_{TX-PKG-PLL2}$	Tx PLL banwidth corresponding to $PKG_{TX-PLL2}$	5.0 (max) 2.0 (min)	MHz
$PKG_{TX-PLL1}$	Tx PLL peaking corresponding to $BW_{TX-PKG-PLL1}$	2.0 (max)	dB
$PKG_{TX-PLL2}$	Tx PLL peaking corresponding to $BW_{TX-PKG-PLL2}$	1.0 (max)	dB

For the High PLL Bandwidth scenario, based on the above specifications, we conclude that the PLL bandwidth should be comprised between 2MHz and 5MHz, and the peaking can be up to 2dB maximum.

6.3 PLL Loop Filter Calculations

The different Loop Filter possibilities given in Table 6-2 were studied, in order to cover all BW and peaking possibilities. The idea is to choose the best PLL Bandwidth, and minimize global Phase Noise impact. The loop filter is the one given in Fig 3-4. The different BW frequencies and peaking used are the following:

Peaking: 0.5 dB, 1.0 dB, 1.5 dB, 2.0 dB

BW Frequency: 2 MHz, 3 MHz, 4 MHz, 5 MHz

In order to create the Transfer Functions with above properties, in Open Loop, we make sure that at 0dB, we have the maximum Phase Margin, as shown in Fig 6-3:

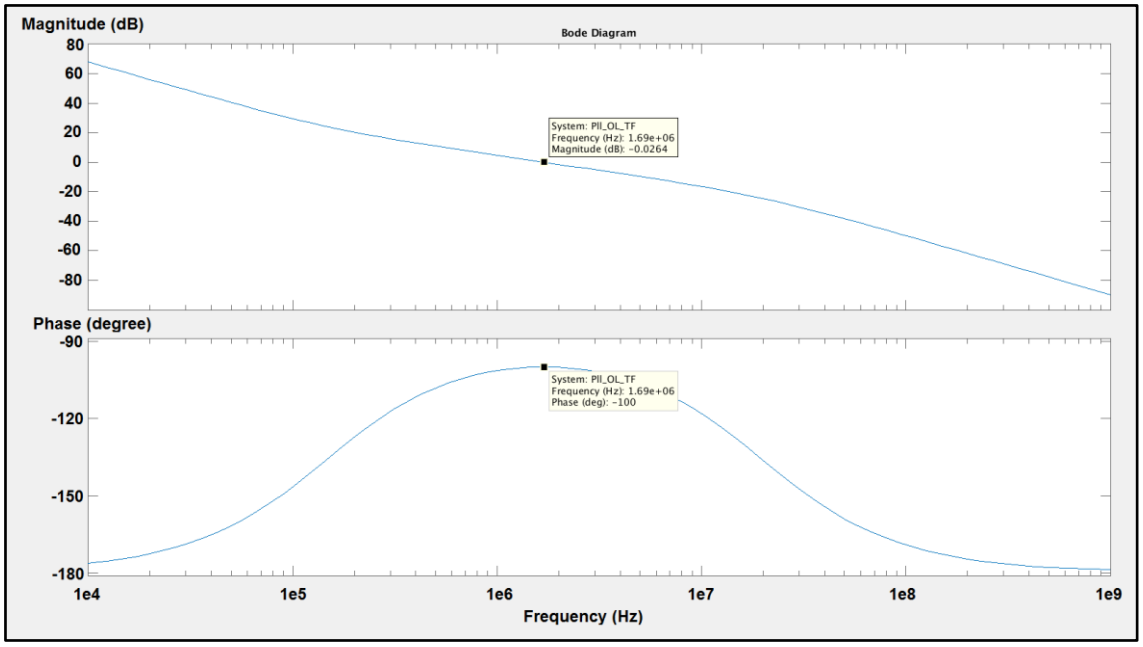


Figure 6-3: Open Loop Transfer Function

Then we check Peaking Value and the Frequency @ -3dB (cut off frequency is -3dB at Nyquist Frequency), in Closed Loop, as shown in Fig 6-4:

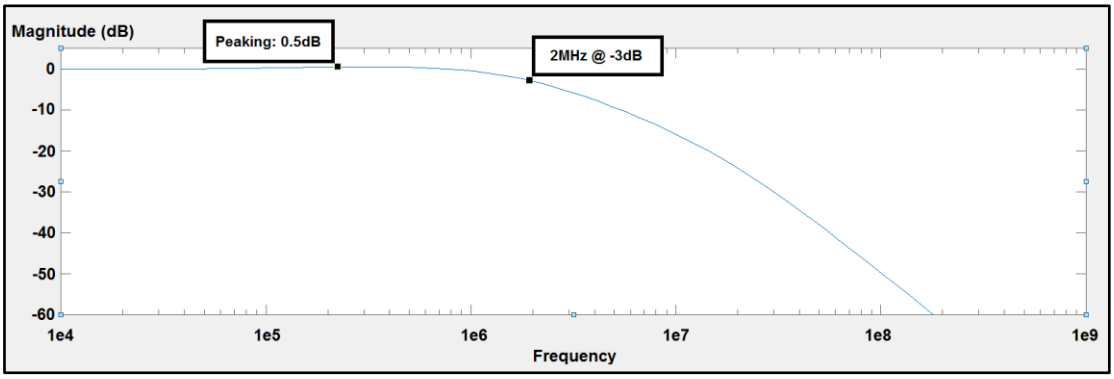


Figure 6-4: Closed Loop Response

In Table 6-3, we show the different simulated values, for the different Transfer Functions that we generated:

Table 6-3: PLL Bandwidth Possibilities

Peaking (@CL)	BW Freq (@-3dB in CL)	0dB OL (Frequency)	Phase Margin (@ OL, @ 0dB)
0.5dB	**2MHz**	1.7 MHz	80°
	3MHz	2.5 MHz	80°
	4MHz	3.3 MHz	80°
	5MHz	4.2 MHz	80°
1.0dB	**2MHz**	1.5 MHz	72°
	3MHz	2.3 MHz	72°
	4MHz	3.0 MHz	72°
	5MHz	3.8 MHz	72°
1.5dB	**2MHz**	1.2 MHz	64°
	3MHz	2.0 MHz	64°
	4MHz	2.7 MHz	64°
	5MHz	3.3 MHz	64°
2.0dB	**2MHz**	1.3 MHz	58°
	3MHz	1.9 MHz	58°
	4MHz	2.6 MHz	58°
	5MHz	3.3 MHz	58°

6.4 CDR equivalent Transfer Function

As given in the standards, we have chosen a CDR Type I, 1st order CDR (-20dB/dec). The transfer function of the CDR is given in (6.2):

$$H(s) = \frac{s}{s+\omega_n} \tag{6.2}$$

With: $\omega_n = \omega_{3dB} = 2\pi * 10MHz$, we have a cut-off frequency of 10 MHz at -3dB.

These parameters were taken from specifications "*PCI Express® Base Specification Revision 4.0, page 999, paragraph 9.3.5.5".*

The Transfer Function of the CDR is given in Fig 6-5.

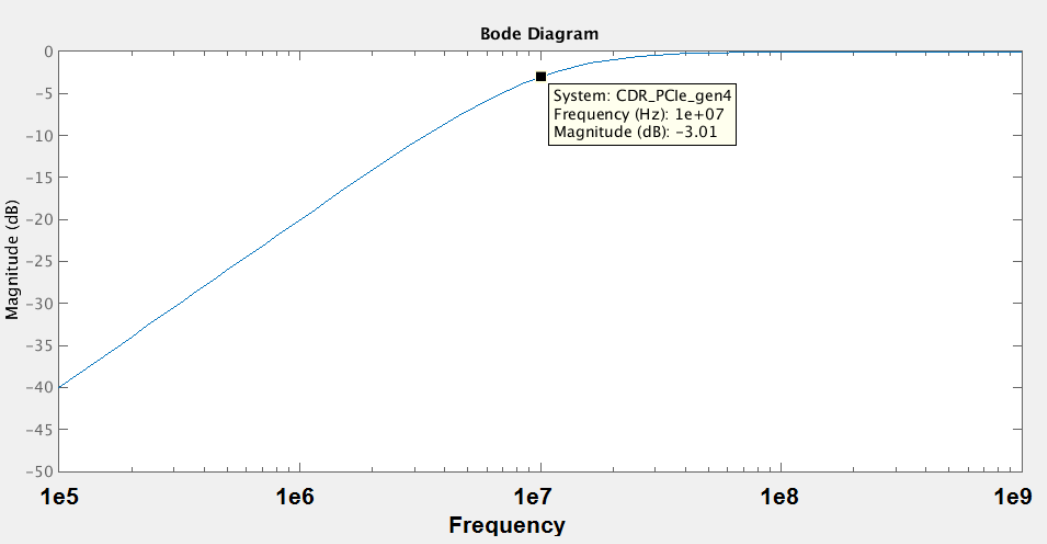


Figure 6-5: CDR Transfer Function

6.5 RMS Jitter Calculation

The RMS jitter will be calculated for the 3 different principal noise parts of the PLL, as shown in Fig 6-1.

The details of each one of these parts are given below:

Inband Noise

This corresponds to the noise (spectrally flat) which is filtered through the PLL Low Pass Filter equivalent TF.

The PLL blocks composing this profile are the followings:

- Reference Clock
- Charge Pump
- Divider output

-20dB/dec Noise

This corresponds to the noise (spectrally colored -20dB/dec) which is filtered through the PLL High Pass Filter equivalent TF (PLL Output)

The PLL blocks composing this profile are the followings:

- VCO Thermal Noise

Floor Noise

This corresponds to the noise (spectrally flat) which is filtered through the PLL High Pass Filter equivalent TF (PLL Output)

The PLL blocks composing this profile are the followings:

- VCO Phase Noise
- Correction Filter Noise

As given in 6.2, the maximum total RMS Jitter for the PLL is defined as 0.445ps. So the equation of the total RMS jitter would be equal to (6.3)

$$\sqrt{\boldsymbol{Noise_{InBand}}^2 + \boldsymbol{Noise_{-20dB/dec}}^2 + \boldsymbol{Noise_{Floor}}^2} = 0.445ps \quad (6.3)$$

Let's consider the 3 noise parts composing the PLL noise profile have equal proportion of jitter. Then, (6.3) would become (6.4):

$$\sqrt{3.\boldsymbol{Noise_{part}}^2} = 0.445ps \quad (6.4)$$

and $\boldsymbol{Noise_{part}} = \boldsymbol{0.257ps}$.

So the maximum authorized RMS Jitter for each of the defined noise profiles would be 0.257ps.

We will define below the maximum noise profiles in dBc/Hz for each of the defined profiles, in order to have a maximum RMS jitter of 0.257ps for each part.

6.5.1 InBand Noise

The RMS Jitter coming from noise (spectrally flat) that passes through the PLL Low Pass Filter, was calculated for different possible phase noise profiles and for the different PLL Bandwidths. Results are given in Table 6-4. The defined maximum RMS Jitter budget is 257fs. In the table in blue color are highlighted all the jitter values below 0.257ps.

Table 6-4: InBand Noise RMS Jitter

		Profile (dBc/Hz)								
Peaking	**BW Freq**	**-110**	**-115**	**-120**	**-125**	**-130**	**-135**	**-140**	**-145**	**-150**
0.5dB	**2MHz**	1.86e-12	1.04e-12	5.87e-13	3.30e-13	1.86e-13	1.04e-13	5.87e-14	3.30e-14	1.86e-14
	3MHz	2.81e-12	1.58e-12	8.88e-13	4.99e-13	2.81e-13	1.58e-13	8.88e-14	4.99e-14	2.81e-14
	4MHz	3.72e-12	2.09e-12	1.18e-12	6.62e-13	3.72e-13	2.09e-13	1.18e-13	6.62e-14	3.72e-14
	5MHz	4.70e-12	2.64e-12	1.49e-12	8.35e-13	4.70e-13	2.64e-13	1.49e-13	8.35e-14	4.70e-14
1dB	**2MHz**	1.43e-12	8.04e-13	4.52e-13	2.54e-13	1.43e-13	8.04e-14	4.52e-14	2.54e-14	1.43e-14
	3MHz	2.36e-12	1.33e-12	7.48e-13	4.21e-13	2.36e-13	1.33e-13	7.48e-14	4.21e-14	2.36e-14
	4MHz	3.18e-12	1.79e-12	1.00e-12	5.65e-13	3.18e-13	1.79e-13	1.00e-13	5.65e-14	3.18e-14
	5MHz	4.08e-12	2.29e-12	1.29e-12	7.25e-13	4.08e-13	2.29e-13	1.29e-13	7.25e-14	4.08e-14
1.5dB	**2MHz**	1.19e-12	6.71e-13	3.78e-13	2.12e-13	1.19e-13	6.71e-14	3.78e-14	2.12e-14	1.19e-14
	3MHz	1.98e-12	1.11e-12	6.27e-13	3.53e-13	1.98e-13	1.11e-13	6.27e-14	3.53e-14	1.98e-14
	4MHz	2.79e-12	1.57e-12	8.82e-13	4.96e-13	2.79e-13	1.57e-13	8.82e-14	4.96e-14	2.79e-14
2dB	**2MHz**	1.12e-12	6.28e-13	3.53e-13	1.98e-13	1.12e-13	6.28e-14	3.53e-14	1.98e-14	1.12e-14
	3MHz	1.88e-12	1.06e-12	5.95e-13	3.35e-13	1.88e-13	1.06e-13	5.95e-14	3.35e-14	1.88e-14
	4MHz	2.68e-12	1.51e-12	8.48e-13	4.77e-13	2.68e-13	1.51e-13	8.48e-14	4.77e-14	2.68e-14

Let's target a PLL Bandwidth of 3MHz, with 1.5dB of peaking. For a phase noise profile of -135dBc/Hz, even if we have a Bandwidth modification due to Kvco variation, we should still satisfy targeted jitter value.

In order to be sure that we satisfy standards, for the Targeted PLL BW (1,5dB of peaking and 3MHz of BW), we study the jitter impact due to Kvco variation. We vary **Kvco** from -50% to +50%, and calculate RMS Jitter, as given in Table 6-5:

Table 6-5: Kvco variation impact on InBand RMS Jitter

Kvco	0dB OL Frequency	Phase Margin (@OL @0dB)	-3dB CL Frequency	Peaking CL	Gain Margin	RMS Jitter (sec)
0.50*Kvco	1.09MHz	58.90°	1.59MHz	2.37dB	-120dB	5.78e-14
0.75*Kvco	1.55MHz	62.90°	2.26MHz	1.82dB	-116dB	8.49e-14
1.00*Kvco	2.00MHz	64.08°	2.98MHz	1.50dB	-113dB	1.11e-13
1.25*Kvco	2.45MHz	63.96°	3.73MHz	1.30dB	-112dB	1.37e-13
1.50*Kvco	2.89MHz	63.17°	4.49MHz	1.15dB	-110dB	1.63e-13

We remark that even in worst case (where **Kvco** gets 50% higher), we have a RMS Jitter due to InBand noise of 0.163ps, which is still smaller than the maximum defined RMS Jitter of 0.257ps.

Therefore the **InBand Noise Profile** is set to **-135dBc/Hz** maximum, and $\boldsymbol{Noise_{InBand} = 0.163ps}$ in worst case.

We can also define the **External Clock Noise Profile** which is generally set to 10dB below the **InBand Noise Profile.** So we can define it to **-145dBc/Hz** maximum, and $\boldsymbol{Noise_{ExtRef} = 0.035ps}$ in worst case.

In Fig 6-6 is shown the reason why for same frequency, the noise decreases when the peaking increases.

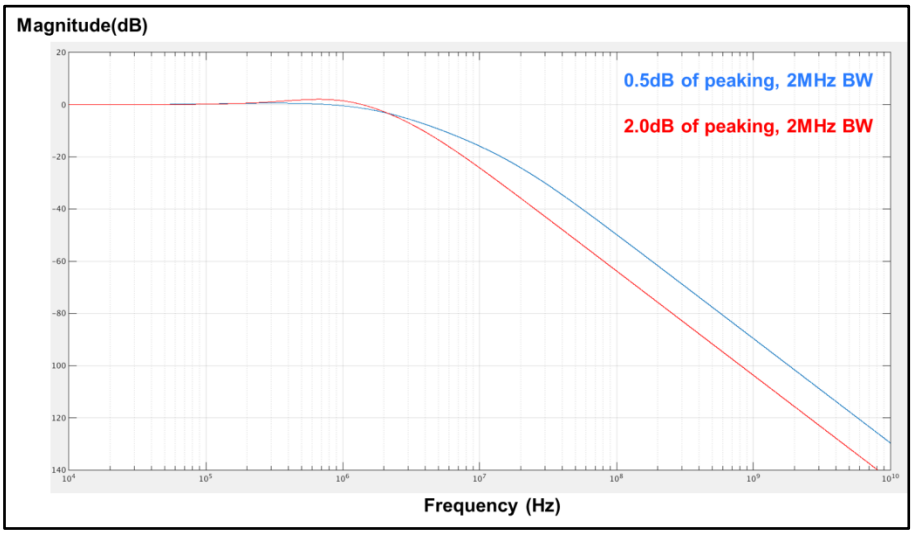


Figure 6-6: Comparision between 2 different PLL BW

We have shown as example 2 different PLL BW, Frequencies fixed at 2MHz, but in blue 0.5dB of peaking, and in red 2dB of peaking. We remark that the 2nd one cuts a little bit faster the high frequencies, which explains a better result of RMS Jitter, for same BW frequency, but higher peaking.

6.5.2 -20dB/dec Noise

Table 6-6 gives the RMS jitter (in s), for the different phase noise profiles (in dBc/Hz) of the different “Slopes of -20dB/dec”, passing at “X” dBc/Hz value @ 1MHz, and for different PLL bandwidths. We have colored in blue color all RMS jitter below 0.257ps. We remark that this is reached for any noise profile and PLL Bandwidth.

Table 6-6: -20dB/dec Noise RMS Jitter given for different profiles

		Slope Profile (dBc/Hz) passing at X value @ 1MHz						
Peaking	**BW Freq**	**-100**	**-105**	**-110**	**-115**	**-120**	**-125**	**-130**
0.5dB	**2MHz**	5.47e-14	3.08e-14	1.73e-14	9.73e-15	5.47e-15	3.08e-15	1.73e-15
	3MHz	5.34e-14	3.00e-14	1.69e-14	9.49e-15	5.34e-15	3.00e-15	1.69e-15
	4MHz	5.19e-14	2.92e-14	1.64e-14	9.24e-15	5.19e-15	2.92e-15	1.64e-15
	5MHz	5.05e-14	2.84e-14	1.60e-14	8.97e-15	5.05e-15	2.84e-15	1.60e-15
1dB	**2MHz**	5.65e-14	3.18e-14	1.79e-14	1.01e-14	5.65e-15	3.18e-15	1.79e-15
	3MHz	5.57e-14	3.13e-14	1.76e-14	9.90e-15	5.57e-15	3.13e-15	1.76e-15
	4MHz	5.48e-14	3.08e-14	1.73e-14	9.74e-15	5.48e-15	3.08e-15	1.73e-15
	5MHz	5.36e-14	3.02e-14	1.70e-14	9.54e-15	5.36e-15	3.02e-15	1.70e-15
1.5dB	**2MHz**	5.76e-14	3.24e-14	1.82e-14	1.02e-14	5.76e-15	3.24e-15	1.82e-15
	3MHz	5.74e-14	3.23e-14	1.81e-14	1.02e-14	5.74e-15	3.23e-15	1.81e-15
	4MHz	5.68e-14	3.20e-14	1.80e-14	1.01e-14	5.68e-15	3.20e-15	1.80e-15
	5MHz	5.61e-14	3.16e-14	1.77e-14	9.98e-15	5.61e-15	3.16e-15	1.77e-15
2dB	**2MHz**	5.83e-14	3.28e-14	1.84e-14	1.04e-14	5.83e-15	3.28e-15	1.84e-15
	3MHz	5.84e-14	3.29e-14	1.85e-14	1.04e-14	5.84e-15	3.29e-15	1.85e-15
	4MHz	5.82e-14	3.27e-14	1.84e-14	1.04e-14	5.82e-15	3.27e-15	1.84e-15
	5MHz	5.78e-14	3.25e-14	1.83e-14	1.03e-14	5.78e-15	3.25e-15	1.83e-15

For the Targeted PLL BW (1,5dB of peaking and 3MHz of BW), if we vary Kvco from -50% to +50%, we have following values, given in Table 6-7:

Table 6-7: Kvco variation impact on -20dB/dec slope RMS Jitter

Kvco	**50% Kvco**	**75% Kvco**	**100% Kvco**	**125% Kvco**	**150% Kvco**
RMS Jitter (sec)	5.47e-14	5.47e-14	5.47e-14	5.47e-14	5.47e-14

We remark that there is no impact on this jitter, due to Kvco variation. Therefore the **-20dB/dec Noise Profile** is set to **-100dBc/Hz** maximum at **1MHz**, and $\boldsymbol{Noise_{-20dB/dec} = 0.0547ps}$ in worst case.

We can now calculate the proportion of RMS jitter left to the Noise Floor Profile:

In equation (6.3) we had:

$$\sqrt{Noise_{InBand}{}^2 + Noise_{-20dB/dec}{}^2 + Noise_{Floor}{}^2} = 0.445ps$$

Therefore, we can find the jitter proportion left to $Noise_{Floor}$ with (6.5):

$$Noise_{Floor} = \sqrt{0.445^2 - (Noise_{InBand}{}^2 + Noise_{-20dB/dec}{}^2)} \quad (6.5)$$

and $Noise_{Floor_{max}} = 0.410ps$

6.5.3 Noise floor

Table 6-8 gives the RMS jitter (in s) for Floor noise, when passing through high pass filter PLL. The RMS jitter values are independent of peaking or PLL BW frequency. This is because the PLL high pass filter max BW Frequency is set to 5MHz. Compared to the 8GHz frequency we integrate, it is negligible (about 0.03%).

Table 6-8: Floor Noise RMS Jitter given for different profiles

Profile (dBc/Hz)								
-110	**-115**	**-120**	**-125**	**-130**	**-135**	**-140**	**-145**	**-150**
5.62e-12	3.16e-12	1.78e-12	1.00e-12	5.62e-13	3.16e-13	1.78e-13	1.00e-13	5.62e-14

We defined above $Noise_{Floor_{max}} = 0.410ps$, so the global budget of the **floor phase noise** is **-135dBc/Hz** at maximum based on Table 6-8.

For the targeted PLL BW (1,5dB of peaking and 3MHz of BW), if we vary Kvco from -50% to +50%, we have following values, given in Table 6-9:

Table 6-9: Kvco variation impact on Floor RMS Jitter

Kvco	**50% Kvco**	**75% Kvco**	**100% Kvco**	**125% Kvco**	**150% Kvco**
RMS Jitter (sec)	3.1629e-13	3.1630e-13	3.1630e-13	3.1631e-13	3.1632e-13

There is no impact on this jitter, due to Kvco variation. Therefore the **floor noise profile** is set to **-135dBc/Hz** maximum, and $Noise_{Floor} = 0.316ps$ in worst case.

6.6 Resuming Results

In conclusion, for our example, the targeted PLL bandwidths will be as given in Table 6-10:

Table 6-10: Targeted PLL Bandwidths

Clock Scenario:	Peaking	PLL BW
Common Clock (High BW)	1.5 dB	3 MHz

The Targeted Phase Noise Profiles are the followings:

External REF => -145dBc/Hz

InBand Noise => -135dBc/Hz

Slope Noise (-20db/dec) => -100dBc/Hz @ 1MHz

Noise Floor => -135dBc/Hz

The results in term of RMS Jitter are given in Table 6-11.

Table 6-11: Resuming Results in terms of RMS Jitter

	Ref Clock Scenario	External Ref -145dBc/Hz	InBand Noise -135dBc/Hz	-20dB/dec Noise -100dBc/Hz @ 1MHz	Floor Noise -135dBc/Hz	Total RMS Jitter
Standards						**0.445ps (max)**
Simulated Jitter with Targetted PLL Parameters	**High BW PLL**	**0.084ps**	**0.163ps**	**0.055ps**	**0.316ps**	**0.369ps**

We conclude that in this example, for the chosen phase noise profiles and PLL BW specifications, we satisfy standard requirements in terms of RMS Jitter.

Furthermore, The Jitter Correlation Algorithm method explained in 2.5 for estimating the total jitter by convolving all DJ jitters helped us optimize the total jitter of our PLL block in terms of jitter budget:

- Minimum of 3ps of total jitter (in a worst case) at BER 10^{-12}, when all DJ Jitters are considered as Dual-Diracs.
- Minimum of 8ps of total jitter (in a normal case) at BER 10^{-12}., when DJ jitters are considered as equally-distributed.

Chapter 7: Conclusions and Perspectives

As explained in the beginning of this thesis, two of the major electrical parameters used to characterize SerDes integrated circuit performances are:

- The transmitter voltage level & transmitted jitter at a given BER
- The receiver maximum insertion loss capabilities & capacity to track jitter at a given BER.

Modeling the phase noise of the different SerDes components, extracting the time jitter and decomposing it, would help designers to achieve desired figure of merit for future SerDes versions.

The PLL is one of the contributors of clock random and periodic jitter inside the system. This is the reason why we decided to model the PLL with phase noise injection and estimate the RMS jitter.

There were 3 main goals for our study:

- Our 1st target was to obtain a strong knowledge of jitter, phase noise, and time/frequency analysis.
- The 2nd target was to provide tools for the designers during the design phase, so they can properly constraint the design and estimate the jitter contribution of each block of the PLL. Furthermore, provide tools to analyze effect of non-linearity in the system.
- The 3rd target was to provide tools for the designers during the IC validation phase to decompose jitter into RJ & DJ, and also identify the sources causing DJ to improve overall solution.

We have started by doing a deep theoretical study of jitter and phase noise. We understood how they are related, and how to analyze them in both domains (time and frequency).

Then we have developed a frequency domain PLL model. The phase noise profiles corresponding to each sub-block of the PLL are obtained from harmonic balance simulations. This is a small signal model which works with transfer functions, and estimates the RMS jitter found in the system, at the output of the PLL.

In order to include loop nonlinearities, and estimate the time jitter, we have created a time-domain PLL model. The characteristics of this model are the followings:

- First of all it converts the phase noise profiles into time jitter. (The time jitter can also be obtained from laboratory measurements)
- The time jitters corresponding to each sub-block of the PLL are injected in the time domain PLL loop
- The model will estimate the time jitter of the system, at the output of the PLL.
- The RMS jitter can then be estimated by calculating the standard deviation of the time jitter.

The frequency-domain model allowed us to validate the time-domain model in terms of RMS jitter, by comparing their results, with a maximum of 1.6% of difference.

From laboratory measurements of a transmitter output (TX of a SerDes) we may observe two characteristics:

1) The jitter has a profile that is spectrally colored.

2) The jitter has Gaussian distribution properties.

In order to analyze SerDes system characteristics, it is important to generate synthetic colored noise patterns with a Gaussian distribution. The patterns will be used to predict impact of jitter on the system performance with time domain simulation during the design verification phase. They will also be used to define the budgets in terms of jitter, for the different blocks of SerDes system.

To our knowledge, in the literature there is no previous study proposing a method for generating colored noise patterns, with Gaussian distribution properties for the time jitter. We have proposed a novel method for generating colored noise patterns with Gaussian distributions properties.

The standard organizations (such as USB or PCIe) specify random and deterministic jitter budgets. In order to decompose the PLL output estimated TIE jitter into random and deterministic jitter, we proposed a new technique for jitter analysis and decomposition. We realized laboratory measurements to verify the algorithm. Jitter extraction simulation results correlate well with measurements and this

technique will help designers to properly identify and quantify the sources of deterministic jitter and their impact on the SerDes system.

Furthermore, these models helped us developing a new method, for System Specifications. We did PLL specifications in terms of phase noise, for different standards and different PLL properties. All generated specifications satisfy standards in terms of random and deterministic jitter.

In perspective, it would be interesting to model the full chain of the SerDes, by modeling the TX, RX and the channel. There is a real interest in modeling the CDR as it will quantify the jitter removed after CDR filtering.

Chapter 8: Annexes

Table of Figures

Table of Tables

8.1 Sinusoidal Signal, Phase & Amplitude Noise

8.1.1 Sinusoidal Signal Phase Modulation Equations

In this sub-section, we give equation demonstration of decomposition of a phase modulation formula.

We have a **phase modulation equation (8.1)**:

$$\boldsymbol{V(t) = V_0 sin[(\omega_c t) + \beta sin(\omega_m t + \varphi_m)]} \quad (8.1)$$

Expressing the equation with exponentials as in (8.2):

$$V(t) = V_0 \frac{e^{j\omega_c t + j\beta sin(\omega_m t + \varphi_m)} - e^{-j\omega_c t - j\beta sin(\omega_m t + \varphi_m)}}{2j} \quad (8.2)$$

We note: $a = \omega_c t$ and $b = \beta sin(\omega_m t + \varphi_m)$ (8.3)

So the above exponential equation (8.2) becomes (8.4):

$$V(t) = V_0 \frac{e^{ja+jb} - e^{-ja-jb}}{2j} \quad (8.4)$$

We have the following exponential development (8.5):

$$e^{jx} \simeq 1 + x + \frac{(jx)^2}{2!} + \ldots + \frac{(jx)^n}{n!} + \varepsilon \quad (8.5)$$

For x<<1, the exponential development (8.5) is approximated to: $e^{jx} \simeq 1 + x$ (8.6)

We rewrite the equation:

$$V(t) = \frac{V_0}{2j}[e^{ja}(1 + jb) - e^{-ja}(1 - jb)] \quad (8.7)$$

$$V(t) = \frac{V_0}{2j}[(e^{ja} - e^{-ja}) + jb(e^{ja} + e^{-ja})] \quad (8.8)$$

$$V(t) = \frac{V_0}{2j}(e^{ja} - e^{-ja}) + \frac{V_0}{2} b(e^{ja} + e^{-ja}) \quad (8.9)$$

$$V(t) = V_0 sin(a) + \frac{V_0}{2}\beta sin(\omega_m t + \varphi_m)(e^{ja} + e^{-ja}) \quad (8.10)$$

We note: $c = \omega_m t + \varphi_m$ (8.11)

$$V(t) = V_0 sin(a) + \frac{V_0}{2}\beta sin(c)(e^{ja} + e^{-ja}) \quad (8.12)$$

$$V(t) = V_0 sin(a) + \frac{V_0}{2}\beta \frac{e^{jc} - e^{-jc}}{2j}(e^{ja} + e^{-ja}) \quad (8.13)$$

$$V(t) = V_0 sin(a) + \frac{V_0 \beta}{2*2j}(e^{jc} - e^{-jc})(e^{ja} + e^{-ja}) \quad (8.14)$$

$$V(t) = V_0 sin(a) + \frac{V_0\beta}{2*2j}(e^{j(c+a)} + e^{j(c-a)} - e^{-j(c-a)} - e^{-j(c+a)}) \quad (8.15)$$

$$V(t) = V_0 sin(a) + \frac{V_0\beta}{2}(\frac{e^{j(c+a)} - e^{-j(c+a)}}{2j} + \frac{e^{j(c-a)} - e^{-j(c-a)}}{2j}) \quad (8.16)$$

$$V(t) = V_0 sin(a) + \frac{V_0\beta}{2}(sin(c+a) + sin(c-a)) \quad (8.17)$$

$$V(t) = V_0 sin(a) + \frac{V_0\beta}{2}(sin(a+c) - sin(a-c)) \quad (8.18)$$

$$V(t) = V_0 sin(\omega_c t) + \frac{V_0\beta}{2}(sin(\omega_c t + \omega_m t + \varphi_m) - sin(\omega_c t - \omega_m t - \varphi_m)) \quad (8.19)$$

$$V(t) = V_0 sin(\omega_c t) + \frac{V_0\beta}{2}(-sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m)) \quad (8.20)$$

This is represented with Fig 8-1:

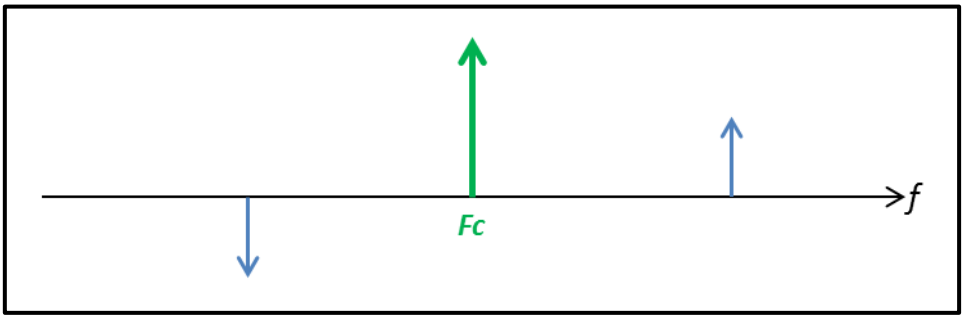


Figure 8-1: Phase Noise Modulation for Sinusoidal Signal

8.1.2 Sinusoïdal Signal Amplitude Modulation Equations

In this sub-section, we give equation demonstration of decomposition of an **Amplitude Modulation**.

We have an amplitude modulation equation:

$$\boldsymbol{V(t) = V_0\, sin(\omega_c t)\,[1 + \beta\, cos(\omega_m t)]} \tag{8.21}$$

$$V(t) = V_0 \frac{e^{j\omega_c t} - e^{-j\omega_c t}}{2j}\left[1 + \beta \frac{e^{j(\omega_m t+\varphi_m)} + e^{-j(\omega_m t+\varphi_m)}}{2}\right] \tag{8.22}$$

$$V(t) = V_0 \frac{e^{j\omega_c t} - e^{-j\omega_c t}}{2j} + V_0\beta \frac{e^{j\omega_c t} - e^{-j\omega_c t}}{2j} * \frac{e^{j(\omega_m t+\varphi_m)} + e^{-j(\omega_m t+\varphi_m)}}{2} \tag{8.23}$$

$$V(t) = V_0 sin(\omega_c t) + \frac{V_0\beta}{2} \frac{e^{j(\omega_c t+\omega_m t+\varphi_m)} + e^{j(\omega_c t-\omega_m t-\varphi_m)} - e^{-j(\omega_c t-\omega_m t-\varphi_m)} - e^{-j(\omega_c t+\omega_m t+\varphi_m)}}{2j} \tag{8.24}$$

$$V(t) = V_0 sin(\omega_c t) + \frac{V_0\beta}{2}\left[\frac{e^{j(\omega_c t+\omega_m t+\varphi_m)} - e^{-j(\omega_c t+\omega_m t+\varphi_m)}}{2j} + \frac{e^{j(\omega_c t-\omega_m t-\varphi_m)} - e^{-j(\omega_c t-\omega_m t-\varphi_m)}}{2j}\right] \tag{8.25}$$

$$V(t) = V_0 sin(\omega_c t) + \frac{V_0\beta}{2}\left[\frac{e^{j(\omega_c t+\omega_m t+\varphi_m)} - e^{-j(\omega_c t+\omega_m t+\varphi_m)}}{2j} + \frac{e^{j(\omega_c t-\omega_m t-\varphi_m)} - e^{-j(\omega_c t-\omega_m t-\varphi_m)}}{2j}\right] \tag{8.26}$$

$$V(t) = V_0 sin(\omega_c t) + \frac{V_0\beta}{2}\left[sin(\omega_c t - \omega_m t - \varphi_m) + sin(\omega_c t + \omega_m t + \varphi_m)\right] \tag{8.27}$$

This is represented with Fig 8-2:

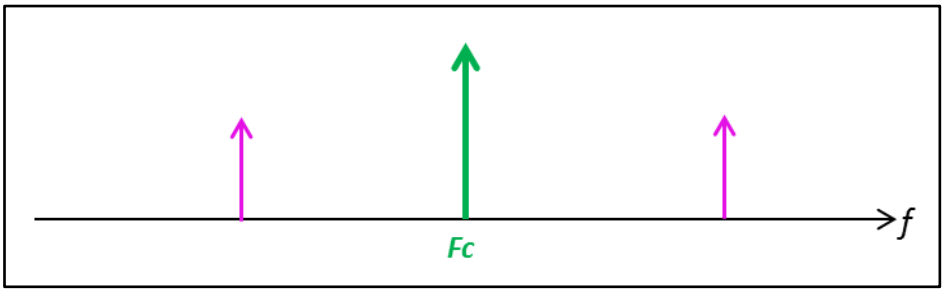


Figure 8-2: Amplitude Noise Modulation for Sinusoidal Signal

8.2 Square Signal, Phase & Amplitude Noise

8.2.1 Square Signal, Phase Modulation Equations

In this sub-section, we give equation demonstration of decomposition of a **Phase Modulation** for a **Square Signal**.

Let us consider a sinusoidal signal, with F_c frequency:

$$\sin_{signal}(t) = \sin(\omega_c t) \tag{8.28}$$

As explained in **Annexe 8.1**, the phase modulation of the sinusoidal signal is expressed as in ():

$$\mathbf{sin}_{\boldsymbol{signal}_{noisy}}(\boldsymbol{t}) = \boldsymbol{V_0}\,\mathbf{sin}(\boldsymbol{\omega_c t} + \boldsymbol{\beta}\mathbf{sin}(\boldsymbol{\omega_m t})) \tag{8.29}$$

For a square signal, a phase modulation corresponds to a displacement of *Δt* in time. This might be showed with the Fig 8-3:

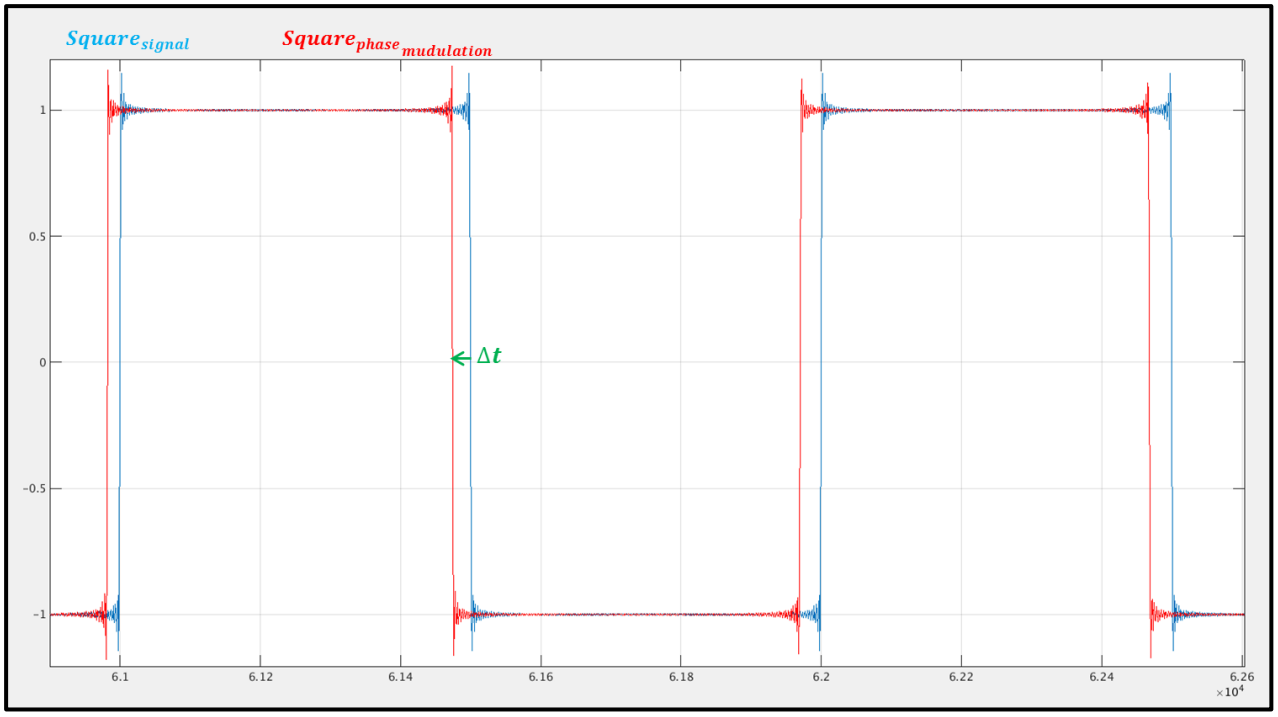


Figure 8-3: Phase Noise Modulation for Square Signal (Time Domain)

The decomposition with the Fourier Series of a time square signal with same Fc frequency can be noted as: $\text{square}_{signal}(t) = V_0 \frac{4}{\pi} \sum_{k=1}^{\infty} \frac{\sin((2k-1)\omega_c t)}{(2k-1)}$ (8.30)

For a sinusoidal signal, the displacement in time is expressed as:

$$sin_{signal_{noisy}}(t) = V_0\, sin(\omega_c t + \ Phase_{Noise}(t)) \tag{8.31}$$

$$sin_{signal_{noisy}}(t) = V_0\, sin\left(\omega_c(t + \frac{Phase_{Noise}(t)}{\omega_c})\right) \tag{8.32}$$

$$sin_{signal_{noisy}}(t) = V_0\, sin(\omega_c(t + \ \Delta t)) \tag{8.33}$$

In a square signal, the displacement in time is expressed as:

$$\text{square}_{signal_{noisy}}(t) = V_0\ \frac{4}{\pi}\sum_{k=1}^{\infty}\frac{\sin((2k-1)\omega_c(t-\Delta t))}{(2k-1)} \tag{8.34}$$

Let's denote $\alpha = \ (2k-1)$ (8.35)

If we develop the expression (8.34) (for 1 harmonic only):

$$\text{square}_{signal_{noisy}}(t) = \ \frac{\mathrm{V_0}\sin(\alpha*\omega_c(t-\Delta t))}{\alpha} = \ \frac{\mathrm{V_0}\sin(\alpha*(\omega_c t-\omega_c\Delta t))}{\alpha} \equiv \ \frac{\mathrm{V_0}\sin(\alpha*(\omega_c t+\beta sin(\omega_m t)))}{\alpha} \tag{8.36}$$

So a phase modulation of a square signal can be expressed as:

$$\mathbf{square}_{\boldsymbol{signal_{noisy}}}(\boldsymbol{t}) = \boldsymbol{V_0}\frac{\boldsymbol{4}}{\boldsymbol{\pi}}\sum_{\boldsymbol{k=1}}^{\boldsymbol{\infty}}\frac{\boldsymbol{sin}((\boldsymbol{2k-1})[\boldsymbol{\omega_c t+\beta\, sin(\omega_m t)}])}{(\boldsymbol{2k-1})} \tag{8.37}$$

$$x_{square_{noise}} = V_0\ \frac{4}{\pi} * \frac{e^{j\omega_c t+j\beta\, sin(\omega_m t)}\ - e^{-j\omega_c t-j\beta\, sin(\omega_m t)}}{2j}\ + \tag{8.38}$$

$$V_0\ \frac{4}{3\pi} * \frac{e^{3j\omega_c t+3j\beta\, sin(\omega_m t)}\ - e^{-3j\omega_c t-3j\beta\, sin(\omega_m t)}}{2j} +$$

$$V_0\ \frac{4}{5\pi} * \frac{e^{5j\omega_c t+5j\beta\, sin(\omega_m t)}\ - e^{-5j\omega_c t-5j\beta\, sin(\omega_m t)}}{2j} + \cdots$$

We note: $a = \omega_c t$ and $b = \beta sin(\omega_m t)$ (8.39)

$$x_{square_{noise}} = \ V_0\frac{4}{\pi} * \frac{e^{ja+jb}\ - e^{-ja-jb}}{2j} \quad + \ V_0\frac{4}{3\pi} * \frac{e^{3ja+3jb}\ - e^{-3ja-3jb}}{2j} + \tag{8.40}$$

$$V_0\frac{4}{5\pi} * \frac{e^{5ja+5jb}\ - e^{-5ja-5jb}}{2j} + \cdots$$

We have the following exponential development:

$$e^{jx} = 1 + jx + \ \frac{(jx)^2}{2!} + \ \ldots + \ \frac{(jx)^n}{n!} \tag{8.41}$$

For n*x<<1, the exponential development is approximated to: $e^{jnx} = 1 + jnx$ (8.42)

This expression (8.42) can be considered as valid, because only the first harmonics count.

We rewrite the equation (8.40):

$$x_{square_{noise}} = V_0 \frac{4}{2j\pi} * \left[e^{ja}(1+jb) - e^{-ja}(1-jb)\right] + \quad (8.43)$$
$$V_0 \frac{4}{3*2j\pi} * \left[e^{3ja}(1+3jb) - e^{-3ja}(1-3jb)\right] +$$
$$V_0 \frac{4}{5*2j\pi} * \left[e^{5ja}(1+5jb) - e^{-5ja}(1-5jb)\right] + \cdots$$

$$x_{square_{noise}} = V_0 \frac{4}{2j\pi} * \left[\left(e^{ja} - e^{-ja}\right) + jb\left(e^{ja} + e^{-ja}\right)\right] + \quad (8.44)$$
$$V_0 \frac{4}{3*2j\pi} * \left[\left(e^{3ja} - e^{-3ja}\right) + 3jb\left(e^{3ja} + e^{-3ja}\right)\right] +$$
$$V_0 \frac{4}{5*2j\pi} * \left[\left(e^{5ja} - e^{-5ja}\right) + 5jb\left(e^{5ja} + e^{-5ja}\right)\right] + \cdots$$

$$x_{square_{noise}} = V_0 \frac{4}{2j\pi} * \left(e^{ja} - e^{-ja}\right) + V_0 \frac{4}{2\pi} * b\left(e^{ja} + e^{-ja}\right) + \quad (8.45)$$
$$V_0 \frac{4}{3*2j\pi} * \left(e^{3ja} - e^{-3ja}\right) + V_0 \frac{4}{2\pi} * b\left(e^{3ja} + e^{-3ja}\right) +$$
$$V_0 \frac{4}{5*2j\pi} * \left(e^{5ja} - e^{-5ja}\right) + V_0 \frac{4}{2\pi} * b\left(e^{5ja} + e^{-5ja}\right) + \cdots$$

$$x_{square_{noise}} = V_0 \frac{4}{2j\pi} * \left(e^{ja} - e^{-ja}\right) + V_0 \frac{4}{3*2j\pi} * \left(e^{3ja} - e^{-3ja}\right) + \quad (8.46)$$
$$V_0 \frac{4}{5*2j\pi} * \left(e^{5ja} - e^{-5ja}\right) + \cdots +$$
$$V_0 \frac{4}{2\pi} * b\left(e^{ja} + e^{-ja}\right) + V_0 \frac{4}{2\pi} * b\left(e^{3ja} + e^{-3ja}\right) +$$
$$V_0 \frac{4}{2\pi} * b\left(e^{5ja} + e^{-5ja}\right) + \cdots$$

$$x_{square_{noise}} = V_0 * \frac{4}{\pi} * \left[\frac{\left(e^{ja} - e^{-ja}\right)}{2j} + \frac{1}{3} * \frac{\left(e^{3ja} - e^{-3ja}\right)}{2j} + \frac{1}{5} * \frac{\left(e^{5ja} - e^{-5ja}\right)}{2j} + \cdots\right] + \quad (8.47)$$
$$V_0 \frac{4}{2\pi} * \left[b\left(e^{ja} + e^{-ja}\right) + b\left(e^{3ja} + e^{-3ja}\right) + b\left(e^{5ja} + e^{-5ja}\right) + \cdots\right]$$

$$x_{square_{noise}} = x_{square} + \tag{8.48}$$

$$V_0 \frac{4}{2\pi} * [b\left(e^{ja} + e^{-ja}\right) + b\left(e^{3ja} + e^{-3ja}\right) + b\left(e^{5ja} + e^{-5ja}\right) + \cdots]$$

$$x_{square_{noise}} = x_{square} + \tag{8.49}$$

$$V_0 \frac{4}{2\pi} * [\beta\sin(\omega_m t)\left(e^{ja} + e^{-ja}\right) + \beta\sin(\omega_m t)\left(e^{3ja} + e^{-3ja}\right) +$$

$$\beta sin(\omega_m t)\left(e^{5ja} + e^{-5ja}\right) + \cdots]$$

We note: $c = \omega_m t$ (8.50)

$$x_{square_{noise}} = x_{square} + \tag{8.51}$$

$$V_0 \frac{4}{2\pi} * [\beta\sin(c)\left(e^{ja} + e^{-ja}\right) + \beta\sin(c)\left(e^{3ja} + e^{-3ja}\right) +$$

$$\beta sin(c)\left(e^{5ja} + e^{-5ja}\right) + \cdots]$$

$$x_{square_{noise}} = x_{square} + \tag{8.52}$$

$$V_0 \frac{4}{2\pi} * [\beta\sin\left(\frac{e^{jc}-e^{-jc}}{2j}\right)\left(e^{ja} + e^{-ja}\right) + \beta\sin\left(\frac{e^{jc}-e^{-jc}}{2j}\right)\left(e^{3ja} + e^{-3ja}\right) +$$

$$\beta sin\left(\frac{e^{jc}-e^{-jc}}{2j}\right)\left(e^{5ja} + e^{-5ja}\right) + \cdots]$$

$$x_{square_{noise}} = x_{square} + \tag{8.53}$$

$$V_0 \frac{4}{2j2\pi} * [\beta\sin\left(e^{jc} - e^{-jc}\right)\left(e^{ja} + e^{-ja}\right) + \beta\sin\left(e^{jc} - e^{-jc}\right)\left(e^{3ja} + e^{-3ja}\right) +$$

$$\beta sin\left(e^{jc} - e^{-jc}\right)\left(e^{5ja} + e^{-5ja}\right) + \cdots]$$

$$x_{square_{noise}} = x_{square} + \tag{8.54}$$

$$V_0 \frac{4\beta}{2j2\pi} * [\left(e^{j(c+a)} + e^{j(c-a)} - e^{-j(c-a)} - e^{-j(c+a)}\right) +$$

$$\left(e^{j(c+3a)} + e^{j(c-3a)} - e^{-j(c-3a)} - e^{-j(c+3a)}\right) +$$

$$(e^{j(c+5a)} + e^{j(c-5a)} - e^{-j(c-5a)} - e^{-j(c+5a)}) + \cdots]$$

$$x_{square_{noise}} = x_{square} + \tag{8.55}$$

$$V_0 \frac{4\beta}{2\pi} * [\left(\frac{e^{j(c+a)} - e^{-j(c+a)}}{2j} + \frac{e^{j(c-a)} - e^{-j(c-a)}}{2j}\right) +$$

$$\left(\frac{e^{j(c+3a)} - e^{-j(c+3a)}}{2j} + \frac{e^{j(c-3a)} - e^{-j(c-3a)}}{2j}\right) +$$

$$\left(\frac{e^{j(c+5a)} - e^{-j(c+5a)}}{2j} + \frac{e^{j(c-5a)} - e^{-j(c-5a)}}{2j}\right) + \cdots]$$

$$x_{square_{noise}} = x_{square} + \tag{8.56}$$

$$V_0 \frac{4\beta}{2\pi} * [(sin(c+a) + sin(c-a)) +$$

$$(sin(c+3a) + sin(c-3a)) +$$

$$(sin(c+5a) + sin(c-5a)) + \cdots]$$

$$x_{square_{noise}} = x_{square} + \tag{8.57}$$

$$V_0 \frac{4\beta}{2\pi} * [(sin(a+c) - sin(a-c)) +$$

$$(sin(3a+c) - sin(3a-c)) +$$

$$(sin(5a+c) - sin(5a-c)) + \cdots]$$

$$x_{square_{noise}} = x_{square} + \tag{8.58}$$

$$V_0 \frac{4\beta}{2\pi} * [sin(\omega_c t + \omega_m t) - sin(\omega_c t - \omega_m t) +$$

$$(sin(3\omega_c t + \omega_m t) - sin(3\omega_c t - \omega_m t)) +$$

$$(sin(5\omega_c t + \omega_m t) - sin(5\omega_c t - \omega_m t)) + \cdots]$$

This is represented with Fig 8-4:

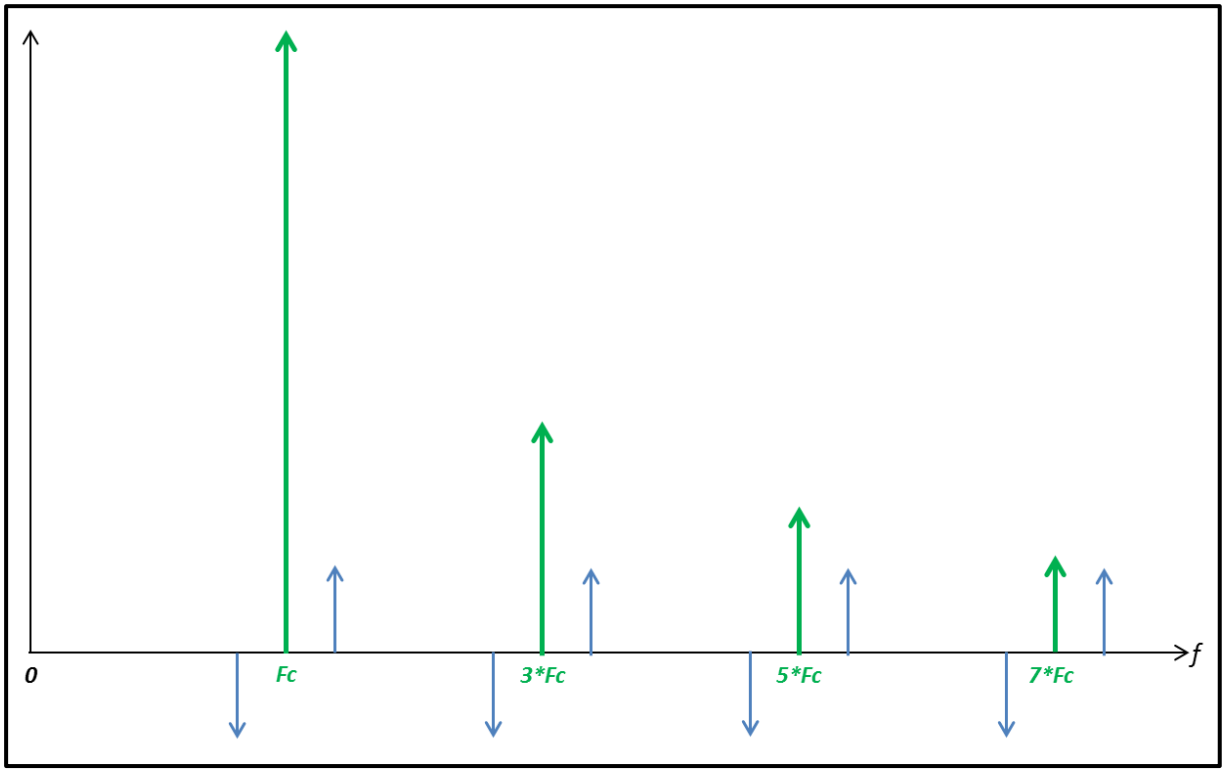


Figure 8-4: Phase Noise Modulation for Square Signal (Frequency Domain)

We remark that phase noise is constant, and reported to all carriers.

In order to verify above equations, we created in Matlab the 4 signals:

- $\sin_{signal}(t)$
- $\text{square}_{signal}(t)$
- $\sin_{signal_{noisy}}(t)$
- $\text{square}_{signal_{noisy}}(t)$

First of all, we verified that Square noisy signal, corresponds or not to the sinusoidal noisy signal.

The time simulation is shown in Fig 8-5 for $\beta = 0.1$.

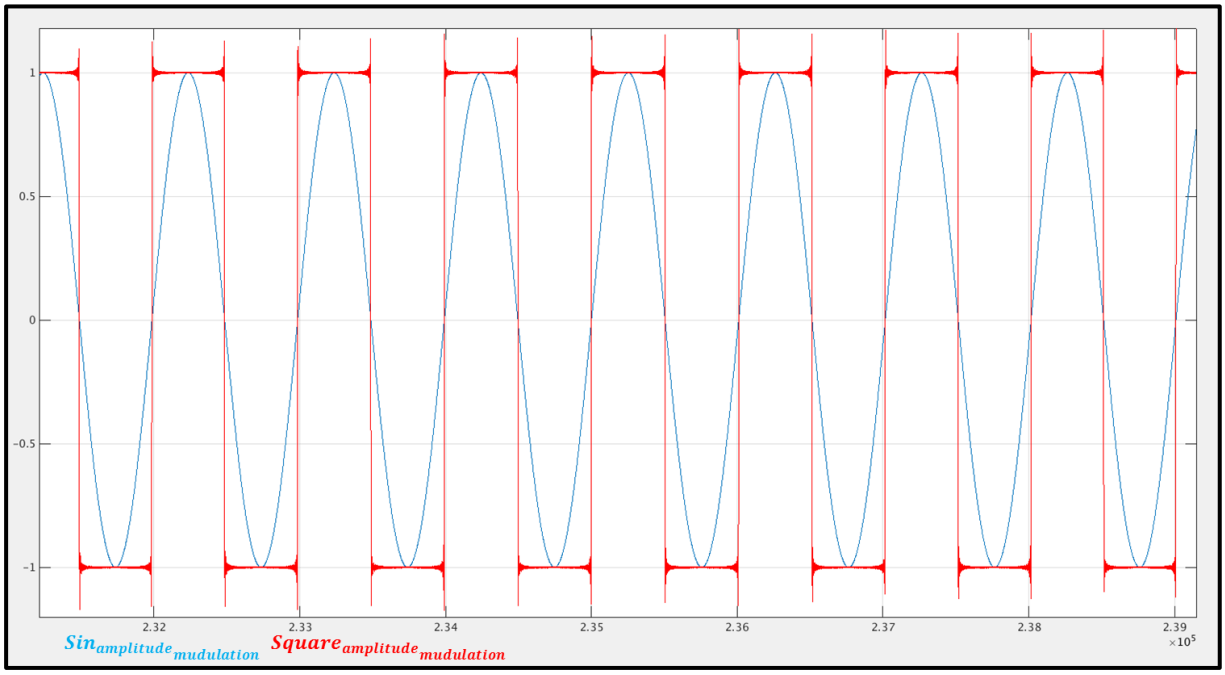


Figure 8-5: Time simulation of Phase Noise Sinusoidal & Square signal

We remark that noisy square signal follows perfectly noisy sinusoidal signal.

We calculate the FFT of the noisy square signal in order to verify if it correlates with Fig 8-4. Noise Spurs are given in absolute values. The results are given in Fig 8-6:

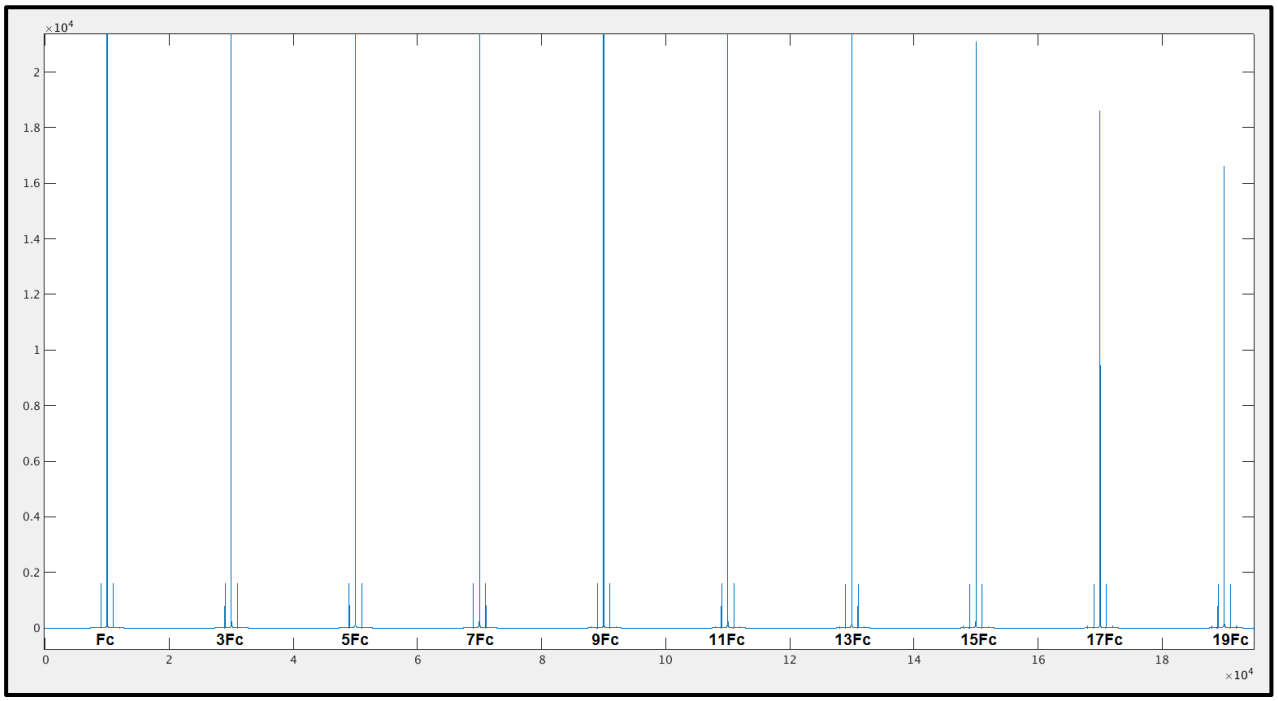


Figure 8-6: Phase Noise Modulation for Square Signal – Result of FFT

This correlates exactly with spurs in Fig 8-4.

Phase noise spurs stay constant, harmonics spurs decrease. This proves that the theoretical decomposition is correct. We lose 20log(n) dBc to each 'n'-th harmonic from the 1st one.

8.2.2 Square Signal, Amplitude Modulation Equations

In this sub-section, we give equation demonstration of decomposition of an **Amplitude Modulation** for a **Square Signal**.

Let us consider a sinusoidal signal, with F_c frequency: $\sin_{signal}(t) = \sin(\omega_c t)$ (8.59)

As explained before, the amplitude modulation of the sinusoidal signal is:

$$\mathbf{sin}_{\boldsymbol{signal}_{\boldsymbol{noisy}}}(\boldsymbol{t}) = \mathbf{sin}(\boldsymbol{\omega_c t})\,[\mathbf{1} + \boldsymbol{\beta}\mathbf{cos}(\boldsymbol{\omega_m t})] \quad (8.60)$$

For a square signal, an amplitude modulation corresponds to a displacement of ΔV in time. This might be shown with the Fig 8-7:

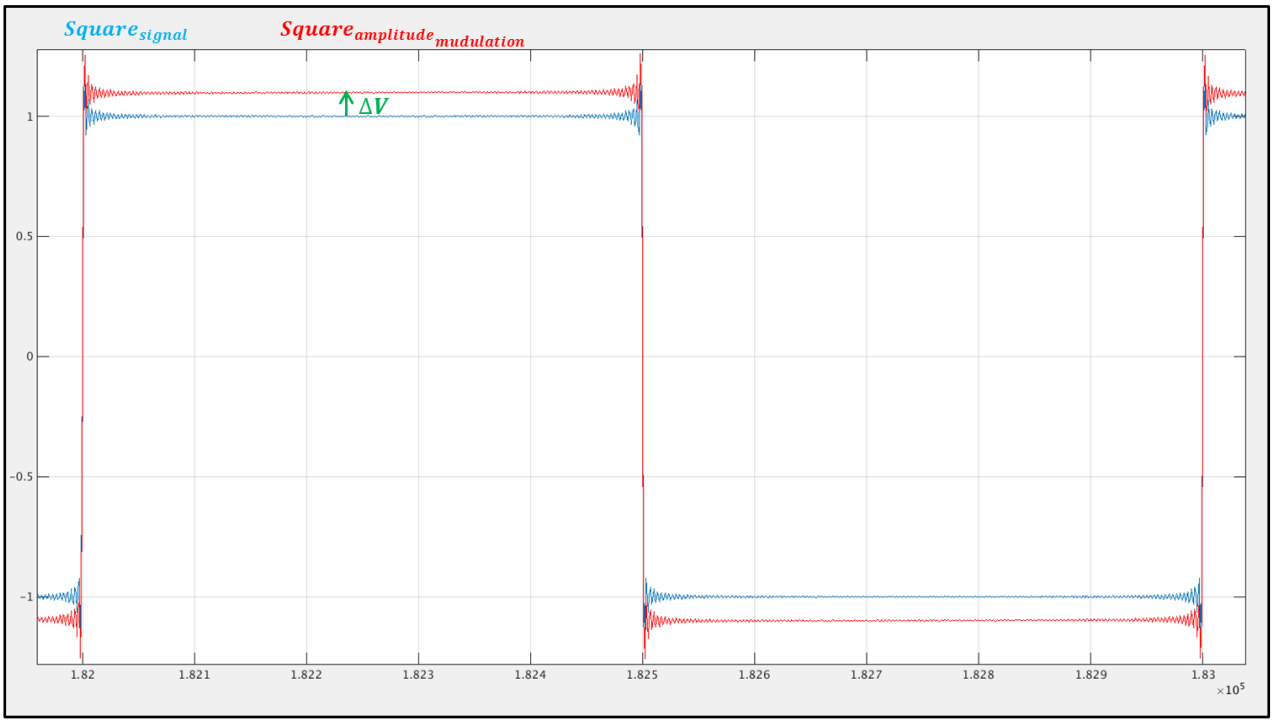


Figure 8-7: Amplitude Noise Modulation for Square Signal

The decomposition with the Fourier Series of a time square signal with same Fc frequency can be noted as: $\text{square}_{signal}(t) = \frac{4}{\pi}\sum_{k=1}^{\infty}\frac{\sin((2k-1)\omega_c t)}{(2k-1)}$ (8.61)

An amplitude modulation for a square signal is expressed as:

$$\mathbf{square}_{\boldsymbol{signal_{noisy}}}(\boldsymbol{t}) = \frac{\mathbf{4}}{\boldsymbol{\pi}}\sum_{\boldsymbol{k=1}}^{\infty}\frac{\boldsymbol{sin}((\mathbf{2k-1})\boldsymbol{\omega_c t})}{(\mathbf{2k-1})}[\mathbf{1} + \boldsymbol{\beta}\, \boldsymbol{cos}(\boldsymbol{\omega_m t})] \quad (8.62)$$

$$square_{signal_{noisy}}(t) = \frac{4}{\pi} * \frac{e^{j\omega_c t} - e^{-j\omega_c t}}{2j}\left[1 + \beta\frac{e^{j(\omega_m t+\varphi_m)} + e^{-j(\omega_m t+\varphi_m)}}{2}\right] + \quad (8.63)$$

$$\frac{4}{\pi} * \frac{e^{j3\omega_c t} - e^{-j3\omega_c t}}{3*2j}\left[1 + \beta\frac{e^{j(\omega_m t+\varphi_m)} + e^{-j(\omega_m t+\varphi_m)}}{2}\right] +$$

$$\frac{4}{\pi} * \frac{e^{j5\omega_c t} - e^{-j5\omega_c t}}{5*2j}\left[1 + \beta\frac{e^{j(\omega_m t+\varphi_m)} + e^{-j(\omega_m t+\varphi_m)}}{2}\right] + \cdots$$

$$square_{signal_{noisy}}(t) = \frac{4}{\pi} * \frac{e^{j\omega_c t} - e^{-j\omega_c t}}{2j} + \frac{4}{\pi} * \frac{e^{j3\omega_c t} - e^{-j3\omega_c t}}{3*2j} + \frac{4}{\pi} * \frac{e^{j5\omega_c t} - e^{-j5\omega_c t}}{5*2j} + \cdots + \quad (8.64)$$

$$\frac{4}{\pi}\beta * \frac{e^{j\omega_c t} - e^{-j\omega_c t}}{2j}\frac{e^{j(\omega_m t+\varphi_m)} + e^{-j(\omega_m t+\varphi_m)}}{2} +$$

$$\frac{4}{\pi}\beta * \frac{e^{j3\omega_c t} - e^{-j3\omega_c t}}{3*2j}\frac{e^{j(\omega_m t+\varphi_m)} + e^{-j(\omega_m t+\varphi_m)}}{2} +$$

$$\frac{4}{\pi}\beta * \frac{e^{j5\omega_c t} - e^{-j5\omega_c t}}{5*2j}\frac{e^{j(\omega_m t+\varphi_m)} + e^{-j(\omega_m t+\varphi_m)}}{2} + \cdots$$

$$square_{signal_{noisy}}(t) = \frac{4}{\pi} * \left[\frac{\sin(\omega_c t)}{1} + \frac{\sin(3\omega_c t)}{3} + \frac{\sin(5\omega_c t)}{5} + \cdots\right] + \quad (8.65)$$

$$\frac{4\beta}{2\pi} * \frac{e^{j(\omega_c t+\omega_m t+\varphi_m)} + e^{j(\omega_c t-\omega_m t-\varphi_m)} - e^{-j(\omega_c t-\omega_m t-\varphi_m)} - e^{-j(\omega_c t+\omega_m t+\varphi_m)}}{2j} +$$

$$\frac{4\beta}{2*3*\pi} * \frac{e^{j(3\omega_c t+\omega_m t+\varphi_m)} + e^{j(3\omega_c t-\omega_m t-\varphi_m)} - e^{-j(3\omega_c t-\omega_m t-\varphi_m)} - e^{-j(3\omega_c t+\omega_m t+\varphi_m)}}{2j} +$$

$$\frac{4\beta}{2*5*\pi} * \frac{e^{j(5\omega_c t+\omega_m t+\varphi_m)} + e^{j(5\omega_c t-\omega_m t-\varphi_m)} - e^{-j(5\omega_c t-\omega_m t-\varphi_m)} - e^{-j(5\omega_c t+\omega_m t+\varphi_m)}}{2j} + \cdots$$

$$square_{signal_{noisy}}(t) = square_{signal}(t) + \tag{8.66}$$

$$\frac{4\beta}{2\pi} * \frac{e^{j(\omega_c t+\omega_m t+\varphi_m)} - e^{-j(\omega_c t+\omega_m t+\varphi_m)}}{2j} + \frac{e^{j(\omega_c t-\omega_m t-\varphi_m)} - e^{-j(\omega_c t-\omega_m t-\varphi_m)}}{2j}$$

$$\frac{4\beta}{3*2\pi} * \frac{e^{j(3\omega_c t+\omega_m t+\varphi_m)} - e^{-j(3\omega_c t+\omega_m t+\varphi_m)}}{2j} + \frac{e^{j3(\omega_c t-\omega_m t-\varphi_m)} - e^{-j(3\omega_c t-\omega_m t-\varphi_m)}}{2j}$$

$$\frac{4\beta}{5*2\pi} * \frac{e^{j(5\omega_c t+\omega_m t+\varphi_m)} - e^{-j(5\omega_c t+\omega_m t+\varphi_m)}}{2j} + \frac{e^{j(5\omega_c t-\omega_m t-\varphi_m)} - e^{-j(5\omega_c t-\omega_m t-\varphi_m)}}{2j}$$

$$square_{signal_{noisy}}(t) = \boldsymbol{square_{signal}(t)} + \tag{8.67}$$

$$\frac{4\beta}{2\pi} * \begin{bmatrix} sin(\omega_c t + \omega_m t + \varphi_m) + sin(\omega_c t - \omega_m t - \varphi_m) + \frac{1}{3}sin(3\omega_c t + \omega_m t + \varphi_m) \\ +\frac{1}{3}sin(3\omega_c t - \omega_m t - \varphi_m) + \frac{1}{5}sin(5\omega_c t + \omega_m t + \varphi_m) + \frac{1}{5}sin(5\omega_c t - \omega_m t - \varphi_m) \end{bmatrix}$$

This is represented with Fig 8-8:

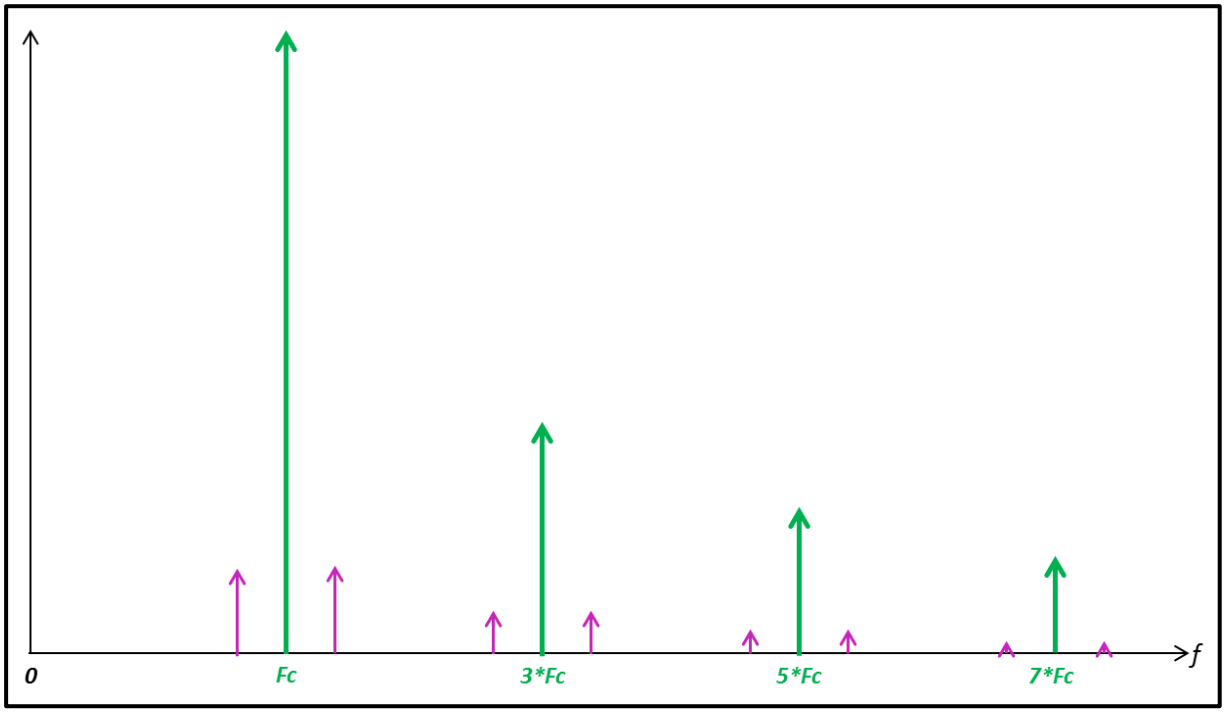


Figure 8-8: Amplitude Noise Modulation for Square Signal (Frequency Domain)

In order to verify above equations, we created in Matlab the 4 signals:

- $\sin_{signal}(t)$
- $\text{square}_{signal}(t)$
- $\sin_{signal_{noisy}}(t)$
- $\text{square}_{signal_{noisy}}(t)$

First of all, we verified if a Square noisy signal corresponds or not to the sinusoidal noisy signal. The time simulation (with $\beta = 0.1$) is shown in Fig 8-9:

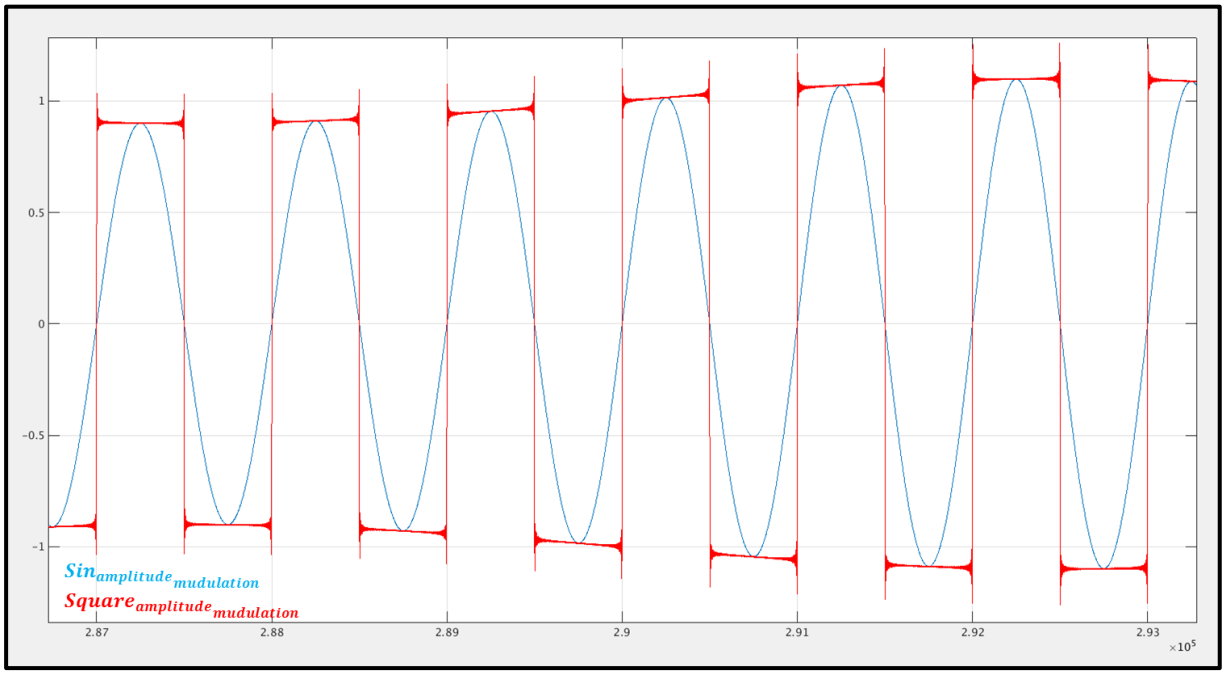


Figure 8-9: Time simulation of Amplitude Noise Sinusoidal & Square signal

We remark that noisy square signal follows perfectly noisy sinusoidal signal.

We calculate the FFT of the noisy square signal in order to verify if it correlates with Fig 8-8. Noise Spurs are given in absolute values. The results are given in Fig 8-10:

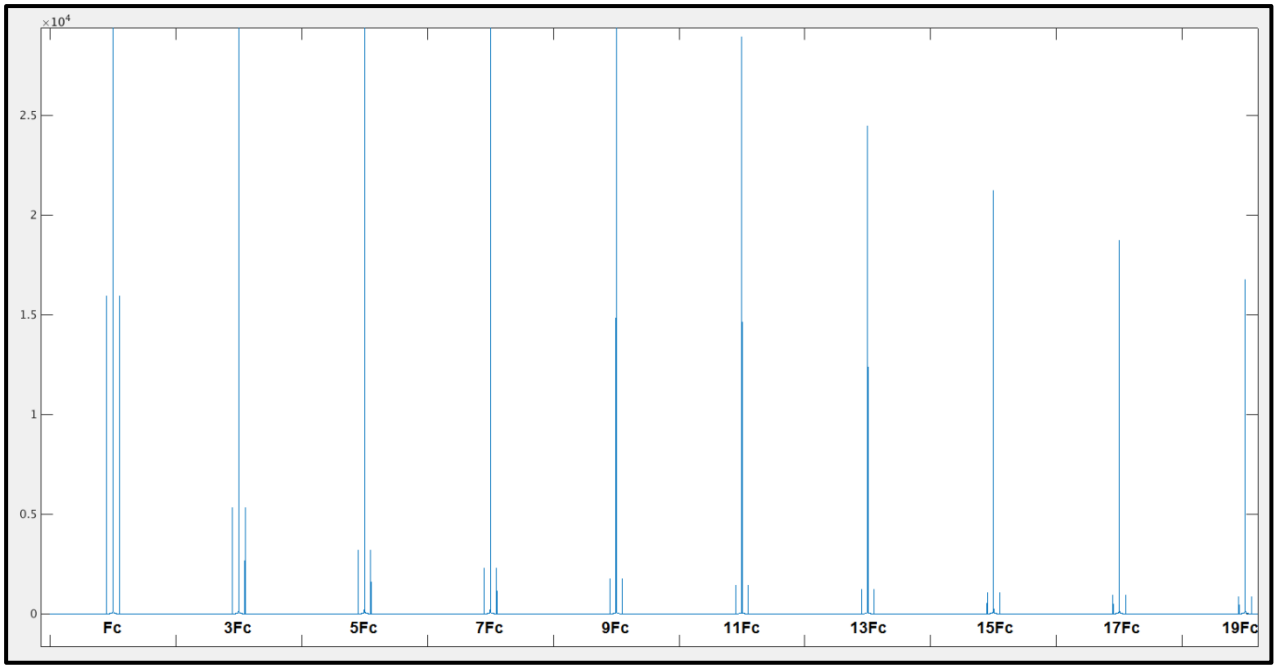


Figure 8-10: Amplitude Noise Modulation for Square Signal – Result of FFT

This correlates exactly with spurs in Fig 8-8.

Amplitude noise spurs decrease proportionally with Frequency Harmonics amplitudes 1/n.

8.3 Measuring Transition Position at a given Offset.

In this section, we will show the fact that adding an offset plays the same role as we had a Phase & an Amplitude Noise. An Offset, is added as a DC, so **@0Hz of frequency**.

If we add an offset @0Hz, normally we should find a Phase & Amplitude Noise @ 2*Fc, as shown in Fig 8-11:

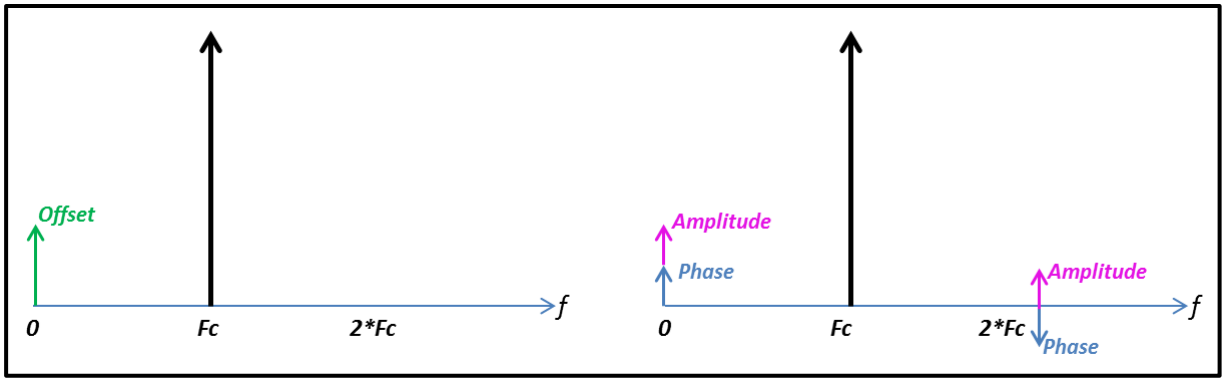


Figure 8-11: Adding an offset to the signal

We add an **Offset** directly to the **signal**, with the following equation:

$$V(t) = \boldsymbol{V_0\, sin}[(\boldsymbol{\omega_c t + \varphi_c})] + \boldsymbol{A_n\, sin(\omega_0 t + \varphi_k)} \quad (8.68)$$

Equation (8.68) can be written as:

$$V(t) = \boldsymbol{V_0\, sin}[(\boldsymbol{\omega_c t + \varphi_c})] + \boldsymbol{A_n sin(\omega_c t - \omega_m t + \varphi_c + \varphi_k)} \quad (8.69)$$

$$V(t) = V_0\, sin[(\omega_c t + \varphi_c)] + 2\frac{A_n}{2} sin(\omega_c t - \omega_m t + \varphi_c + \varphi_k) + \quad (8.70)$$

$$\frac{A_n}{2} sin(\omega_c t + \omega_m t + \varphi_c - \varphi_k) - \frac{A_n}{2} sin(\omega_c t + \omega_m t + \varphi_c - \varphi_k)$$

$$V(t) = V_0\, sin[(\omega_c t + \varphi_c)] + \quad (8.71)$$

$$\frac{A_n}{2}[sin(\omega_c t - \omega_m t + \varphi_c + \varphi_k) + sin(\omega_c t + \omega_m t + \varphi_c - \varphi_k)] +$$

$$\frac{A_n}{2}[sin(\omega_c t - \omega_m t + \varphi_c + \varphi_k) - sin(\omega_c t + \omega_m t + \varphi_c - \varphi_k)]$$

Equation (8.71) can be represented with **Amplitude Noise** and **Phase Noise**.

$$V(t) = V_0 \sin[(\omega_c t + \varphi_c)] + \tag{8.72}$$

$$\frac{A_n}{2}[\boldsymbol{sin(\omega_c t - \omega_m t + \varphi_c - \varphi_k) + \sin(\omega_c t + \omega_m t + \varphi_c + \varphi_k)}] +$$

$$\frac{A_n}{2}[sin(\omega_c t - \omega_m t + \varphi_c + \varphi_k) - sin(\omega_c t + \omega_m t + \varphi_c - \varphi_k)]$$

In our case, $\boldsymbol{\omega_m = \omega_c}$, so the (8.72) equation becomes:

$$V(t) = V_0 \sin[(\omega_c t + \varphi_c)] + \frac{A_n}{2}[\boldsymbol{sin(\omega_0 t + \varphi_c - \varphi_k) + \sin(2\omega_c t + \varphi_c + \varphi_k)}] + \tag{8.73}$$

$$\frac{A_n}{2}[sin(\omega_0 t + \varphi_c + \varphi_k) - sin(2\omega_c t + \varphi_c - \varphi_k)]$$

Equation (8.73) shows that an Offset, is also decomposed as a sum of a Phase & an Amplitude Noise.

8.4 Integration Band for Sin and Square Signal

The objective of this section is to define the integration band for a sinusoidal signal, and the integration band for a square signal.

The schematic used to create jitter is given in Fig 8-12:

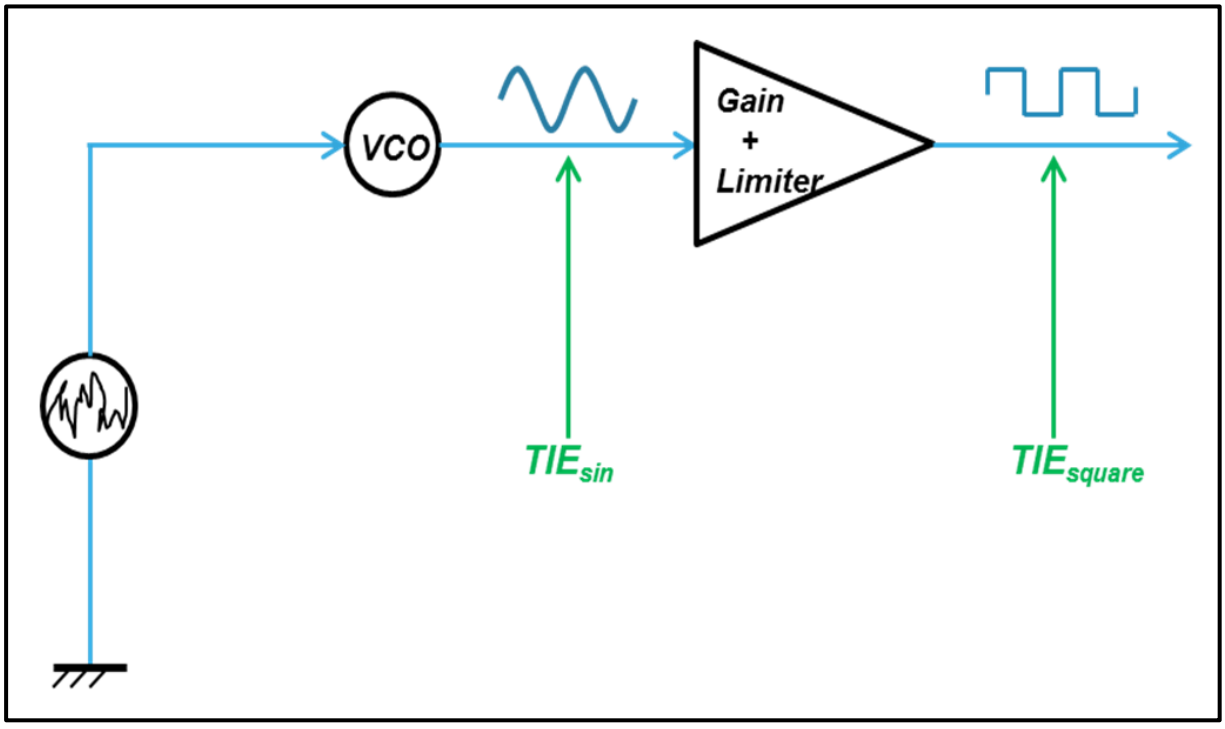


Figure 8-12: Convert Noisy Sinusoidal signal to Noisy Square Signal

The TIE jitter is the same for SIN & SQUARE signals. We know that for a sinusoidal signal we integrate the noise up to infinite. We will define the integration band of a square signal, so the power of the noise (and the RMS Jitter) included in the integration band is same as the one of the sinusoidal signal.

We make the hypothesis that for a square signal, we integrate up to 2*Fc.

In this section, we will try to demonstrate that our hypothesis is exact, and that for a square signal, we should integrate the noise from **0** up to **2*Fc**.

Simulations on ELDO, VERILOG-A and MATLAB are given in **Annexe 8.4.5** to support the different mathematical results.

8.4.1 Let's consider a phase noise @ ω_{m1}, with $0 < \omega_{m1} < F_c$

First of all we take a noise found at an offset of ω_1 from the carrier frequency. We apply it to a sinusoidal and to a square signal.

The noisy sinusoidal signal is expressed as:

$$V(t) = V_0 sin(\omega_c t) + \frac{V_0 \beta}{2}(-sin(\omega_c t - \omega_{m1} t - \varphi_m) + sin(\omega_c t + \omega_{m1} t + \varphi_m)) \qquad (8.74)$$

The Carrier amplitude is: $A_c = V_0$ The Noise amplitude is: $A_{n1} = \frac{V_0 \beta}{2}$

RMS Jitter is found from $-\infty$ to ∞. The frequency representation of the sinusoidal modulated signal is given in Fig 8-13.

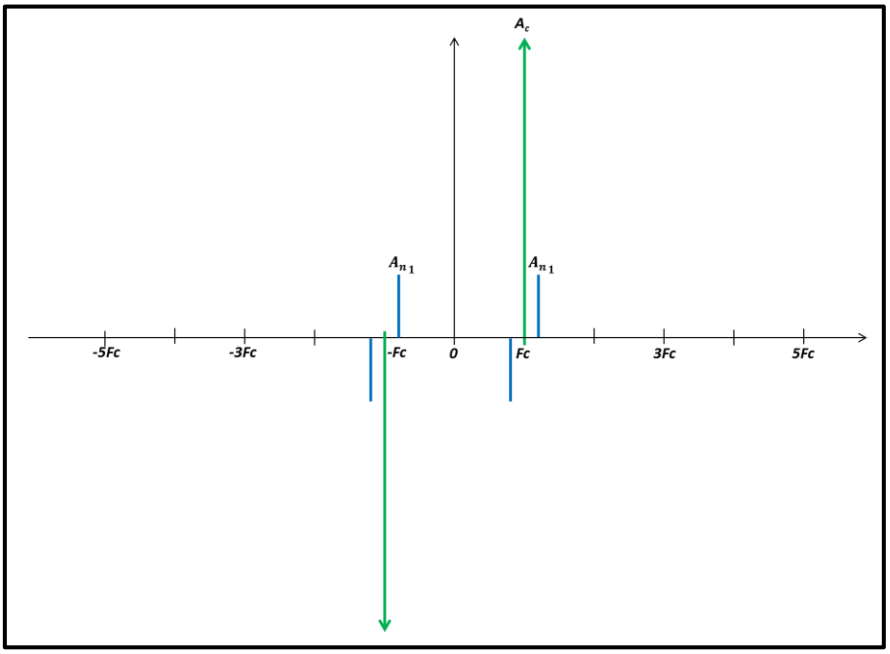


Figure 8-13: Frequency representation of the sinusoidal modulated signal

The jitter RMS of this noise is:

$$RMS_{Jitter_{multispurs}} = \frac{\sqrt{[(\frac{A_{n_1}}{A_c})^2+(\frac{-A_{n_1}}{A_c})^2+(\frac{A_{n_1}}{-A_c})^2+(\frac{-A_{n_1}}{-A_c})^2\ ]}}{2\pi F_c} = \frac{\sqrt{4*\frac{{A_{n_1}}^2}{{A_c}^2}}}{2\pi F_c} = \frac{2*\frac{A_{n_1}}{A_c}}{2\pi F_c} = \frac{2*\frac{\beta}{2}}{2\pi F_c} = \frac{\beta}{2\pi F_c} \quad (8.75)$$

The noisy square signal is expressed as:

$$x_{square_{noise}} = V_0\frac{4}{\pi}(sin(\omega_c t)) + V_0\frac{4\beta}{2\pi}(sin(\omega_c t + \omega_{m1}t) - \sin(\omega_c t - \omega_{m1}t)) + \quad (8.76)$$

$$\frac{V_0}{3}\frac{4}{\pi}(sin(3\omega_c t)) + V_0\frac{4\beta}{2\pi}(sin(3\omega_c t + \omega_{m1}t) - sin(3\omega_c t - \omega_{m1}t)) +$$

$$\frac{V_0}{5}\frac{4}{\pi}(sin(5\omega_c t)) + V_0\frac{4\beta}{2\pi}(sin(5\omega_c t + \omega_{m1}t) - sin(5\omega_c t - \omega_{m1}t)) + \cdots$$

Check **Annexe 8.2** for more information.

The frequency representation of the square modulated signal is given in Fig 8-14.

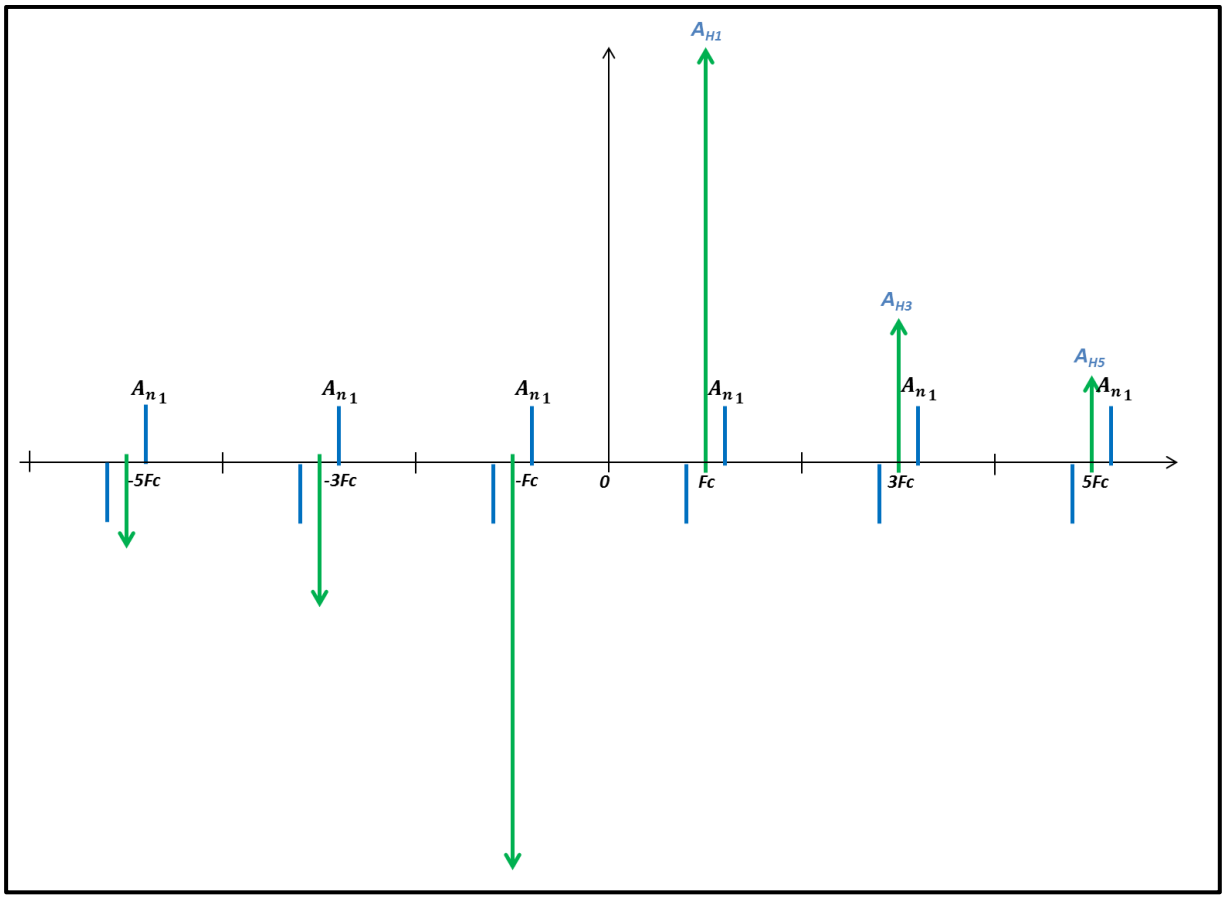


Figure 8-14: Frequency representation of the square modulated signal

The RMS Jitter (TIE) of the square signal should be same as the one of the sinusoidal signal. We will try to define integration band for which these 2 noises are equal.

For Harmonic 1

First of all we calculate the noise around Harmonic 1, with integration band from 0 to 2*Fc frequency.

The Carrier amplitude is: $A_c = V_0 \frac{4}{\pi}$ The Noise amplitude is: $A_{n1} = V_0 \frac{4\beta}{\pi 2}$

The frequency representation of the square modulated signal with the integration band is given in Fig 8-15.

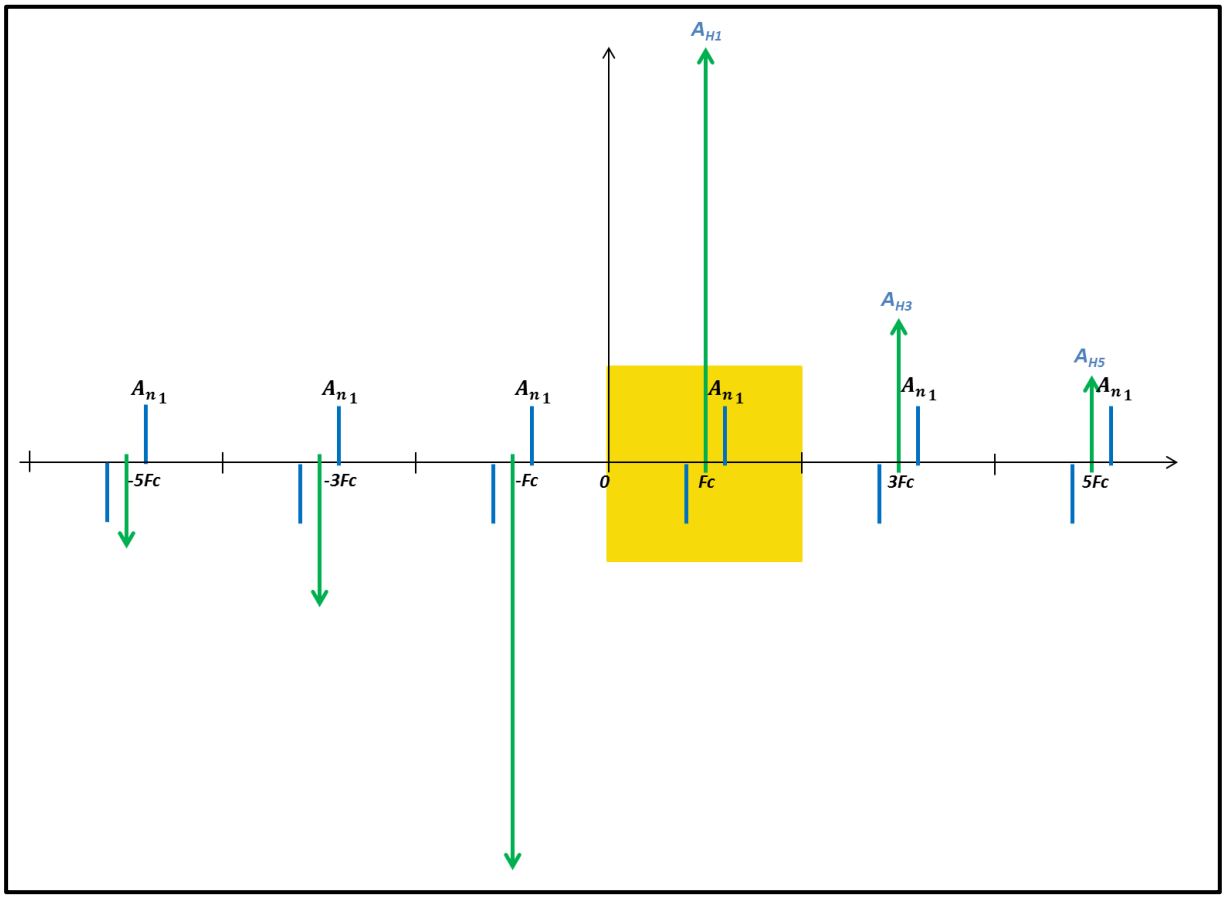


Figure 8-15: Freq. rep. of square modulated signal with Integration band 0->2*Fc

The jitter RMS found from 0 to 2*Fc:

$$RMS_{Jitter_{multispurs}} = \frac{\sqrt{2*[(\frac{A_{n1}}{A_{H1}})^2+(\frac{-A_{n1}}{A_{H1}})^2]}}{2\pi F_c} = \frac{2*\frac{A_{n1}}{A_{H1}}}{2\pi F_c} = \frac{2*\frac{\beta}{2}}{2\pi F_c} = \frac{\beta}{2\pi F_c} \quad (8.77)$$

The multiplication by '2', is done in order to take in account negative frequencies.

We remark that when "$0 < \omega_{m1} < F_c$" the global jitter of sinusoidal signal (8.75) is equal to the jitter included around H_1 for the square signal, from 0 to 2Fc (8.77). The example of a phase noise @ ω_2, with $\omega_{m2} > F_c$, is given in **Annexe 8.4.4**.

8.4.2 Explaining why all noise spurs (with different ω_m) are found between 0 and 2*F_c frequencies.

The objective of this section, is to show that a noise at any offset, is aliased between 0 and 2*Fc frequency.

Let's consider we have a square signal, with a noise spur ω_m. Let's consider that ω_m is found between (n-1)*ω_c and n*ω_c frequency. The noisy square signal is expressed as:

$$x_{square_{noise}} = V_0 \frac{4}{\pi}(sin(\omega_c t)) + V_0 \frac{4\beta}{2\pi}(sin(\omega_c t + \omega_m t) - \sin(\omega_c t - \omega_m t)) + \quad (8.78)$$

$$\frac{V_0}{3}\frac{4}{\pi}(sin(3\omega_c t)) + V_0 \frac{4\beta}{2\pi}(sin(3\omega_c t + \omega_m t) - sin(3\omega_c t - \omega_m t)) + \cdots +$$

$$\frac{V_0}{n}\frac{4}{\pi}(sin(n * \omega_c t)) + V_0 \frac{4\beta}{2\pi}(sin(n * \omega_c t + \omega_m t) - \boldsymbol{sin(n * \omega_c t - \omega_m t)}) + \cdots$$

If "$\mathbf{(n-1) * \omega_c < \omega_m < n * \omega_c}$":

In Fig 8-16 we show only the noise spur, corresponding to the H1 Frequency:

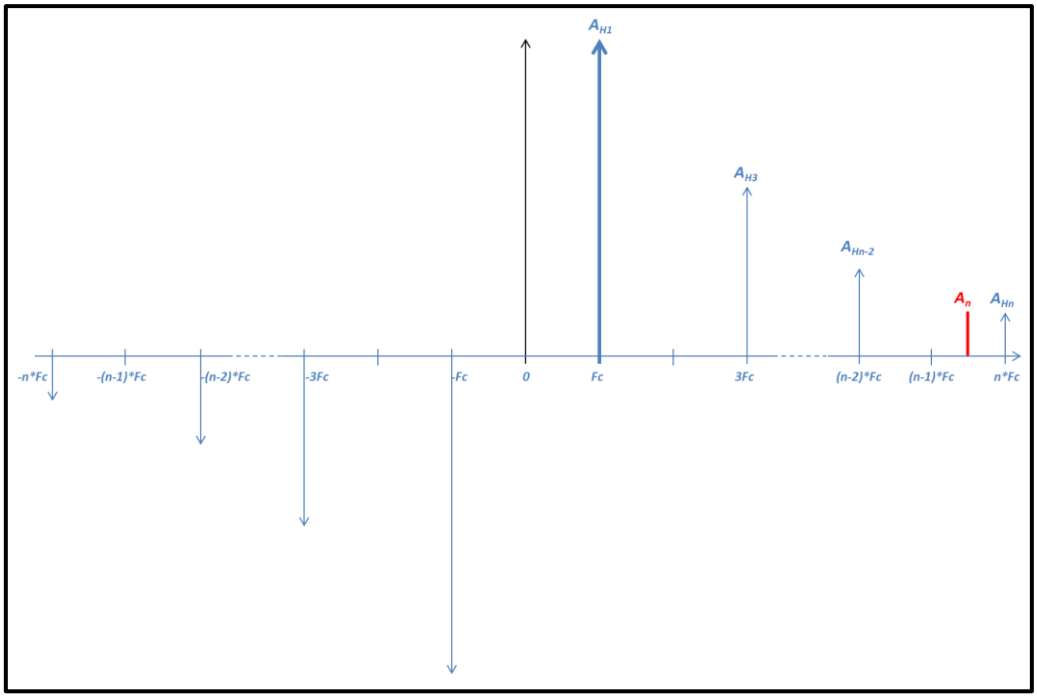


Figure 8-16: Frequency representation – Noise spurs corresponding to H1.

Then, from (8.78) we remark that for the n'th harmonic, the corresponding noise is expressed as:

$$\frac{V_0}{n}\frac{4}{\pi}(sin(n*\omega_c t)) + V_0\ \frac{4\beta}{2\pi}(sin(n*\omega_c t\ + \omega_m t) - \boldsymbol{sin(n*\omega_c t - \omega_m t)}) + \cdots \quad (8.79)$$

We find the aliasing frequencies for 1 of the 2 spur noises of (8.80):

$$\boldsymbol{-sin(n*\omega_c t - \omega_m t))} \equiv -\sin((n*\omega_c - \omega_m)*t) \quad (8.80)$$

We know that $(n-1)*\omega_c < \omega_m < n*\omega_c$, (8.81)

so $\omega_m = (n-1)*\omega_c + \Delta\omega_m$ with "$0 < \Delta\omega_m < F_c$"

$$\boldsymbol{-sin(n*\omega_c t - \omega_m t))} \equiv -sin((n*\omega_c - ((n-1)*\omega_c + \Delta\omega_m))*t) \quad \text{with "}0 < \Delta\omega_m < F_c\text{"} \quad (8.82)$$

$$\equiv -sin((\omega_c + \Delta\omega_m))*t) \quad \text{with "}0 < \Delta\omega_m < F_c\text{"}$$

So one of the noise spurs corresponding to an **n**'th harmonic (8.79), is aliased as a noise between 0 and 2*Fc (8.82).

In Fig 8-17 we show the aliased spur noise:

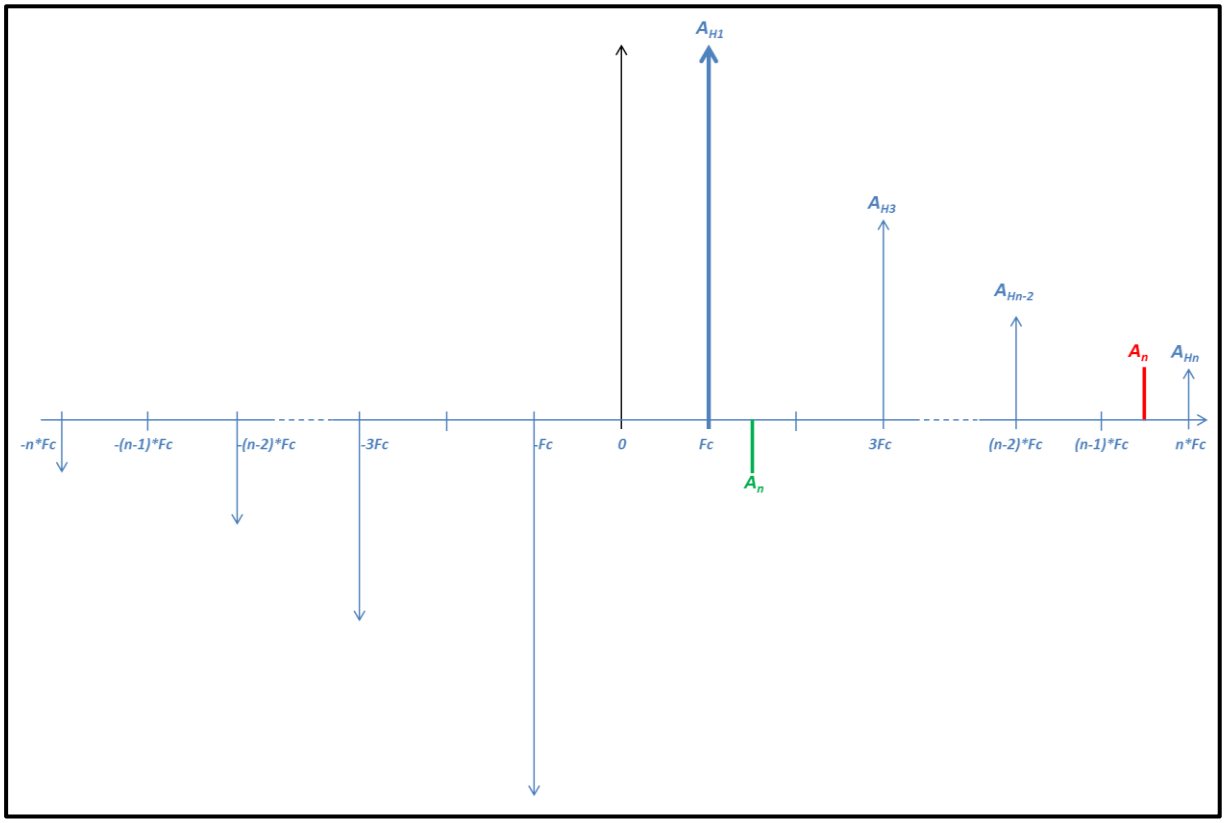


Figure 8-17: Frequency representation – Noise spurs aliased from 0 to 2*Fc.

8.4.3 Let's consider a spectrally flat phase noise profile (up to n*Fc)

The objective of this section, is to show that for a noise profile from $-\infty$ to $+\infty$, the integration band on the square signal $\int_0^{2F_c} Noise_{square}$, is equal to the integration band on the sinusoidal signal $\int_{-\infty}^{+\infty} Noise_{sin}$.

The noisy sinusoidal signal:

The noise on the sinusoidal signal is presented in Fig 8-18:

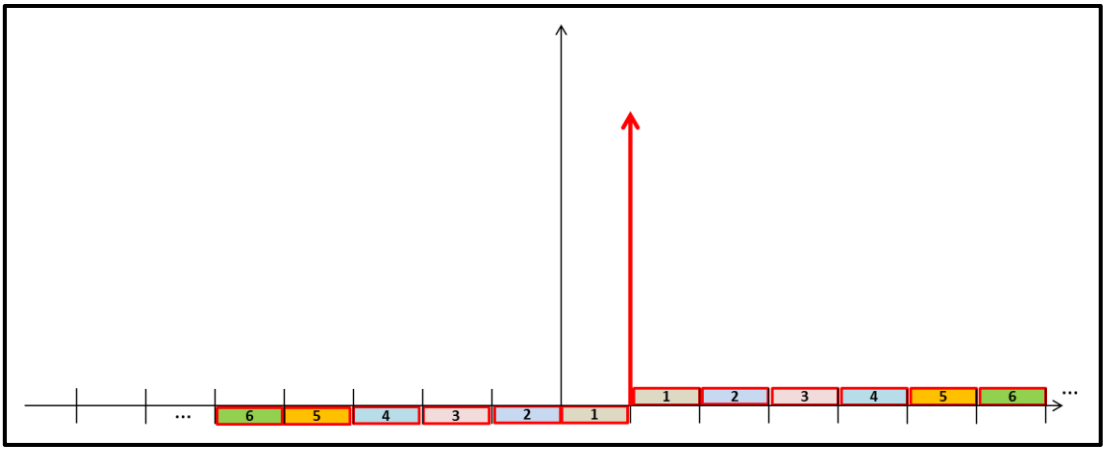


Figure 8-18: Frequency representation – Spectrally Flat Noise on Sinusoidal Signal.

If we present negative frequencies also, we will get Fig 8-19:

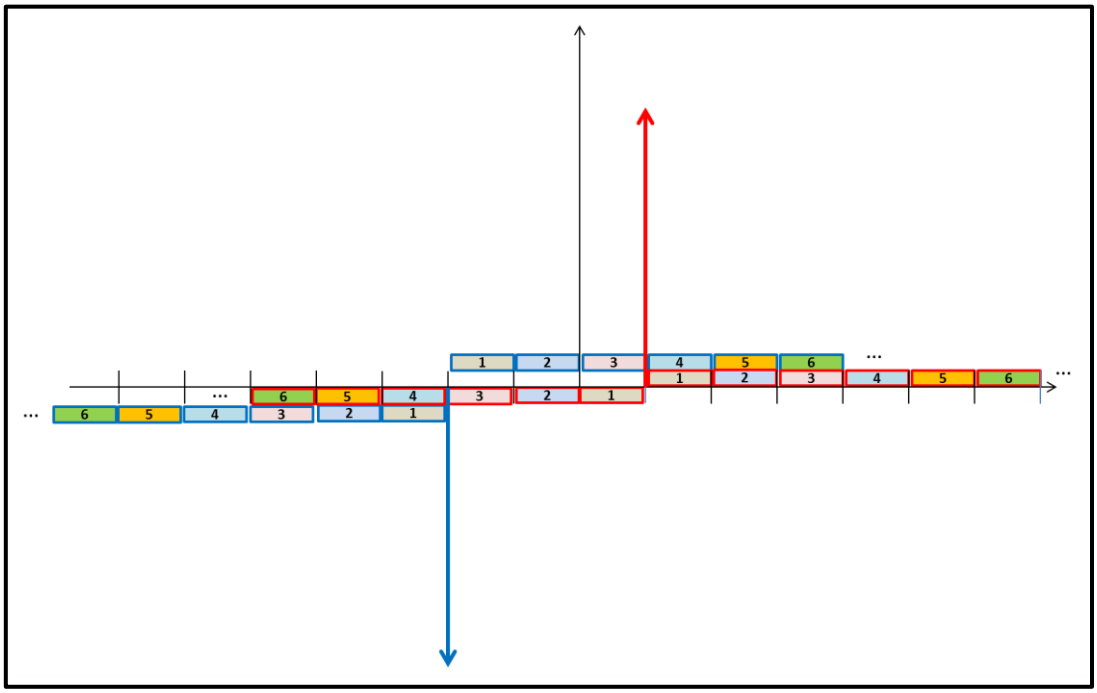


Figure 8-19: Freq. representation of flat noise on Sin. Signal with negative Freq.

So the total RMS jitter, of the sinusoidal noisy signal will be:

$$Jitter_{RMS} = \frac{\int_{-\infty}^{+\infty} Noise_{sin}}{2\pi F_c} = \frac{\sqrt{\sum_{k=1}^{\infty} Noise_k{}^2}}{2\pi F_c} = \frac{\sqrt{4*(n_1{}^2+n_2{}^2+n_3{}^2+n_4{}^2+n_5{}^2+n_6{}^2+\cdots)}}{2\pi F_c} \quad (8.83)$$

The noisy square signal:

The noise on the square signal is presented as in Fig 8-20:

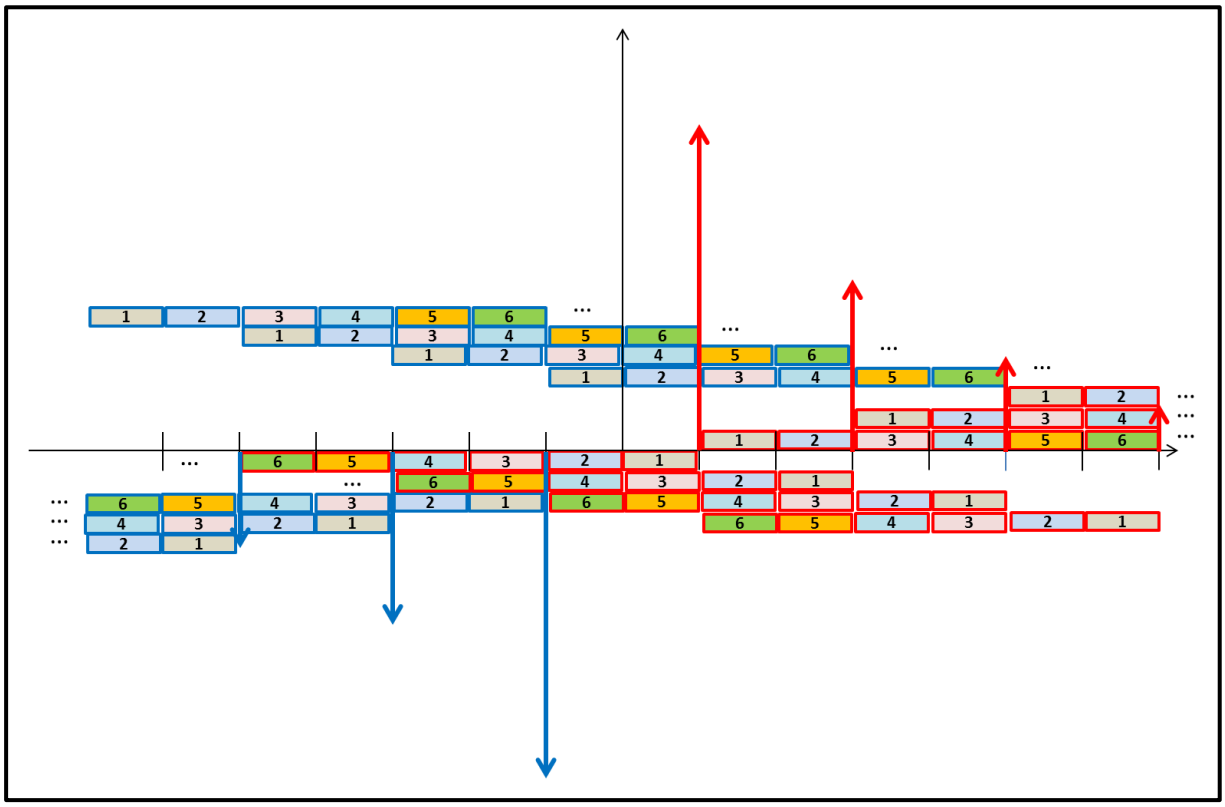


Figure 8-20: Freq. representation of flat noise on Square Signal with negative Freq.

So the total RMS jitter, of the sinusoidal noisy signal will be:

$$Jitter_{RMS} = \frac{\int_0^{2F_c} Noise_{square}}{2\pi F_c} = \frac{\sqrt{2*\sum_0^{2F_c} Noise^2}}{2\pi F_c} = \frac{\sqrt{2*(2*(n_1{}^2+n_2{}^2+n_3{}^2+n_4{}^2+n_5{}^2+n_6{}^2+\cdots))}}{2\pi F_c} \quad (8.84)$$

$$Jitter_{RMS} = \frac{\sqrt{4*(n_1{}^2+n_2{}^2+n_3{}^2+n_4{}^2+n_5{}^2+n_6{}^2+\cdots)}}{2\pi F_c} \quad (8.85)$$

We remark that (8.85), is equal to (8.83).

This proves that our hypothesis is correct. The noise for a square signal should be integrated from 0 to 2*Fc.

8.4.4 Phase noise @ ω_{m2}, with $\omega_{m2} > F_c$

Now we will consider a noise found at an offset of ω_2 from the carrier frequency. We apply it to a sinusoidal and to a square signal.

The noisy sinusoidal signal is expressed as:

$$V(t) = V_0 sin(\omega_c t) + \frac{V_0\beta}{2}(-sin(\omega_c t - \omega_{m2} t - \varphi_m) + sin(\omega_c t + \omega_{m2} t + \varphi_m)) \quad (8.86)$$

The Carrier amplitude is: $A_c = V_0$ The Noise amplitude is: $A_{n2} = \frac{V_0\beta}{2}$

RMS Jitter is found from $-\infty$ to ∞. This is represented with Fig 8-21:

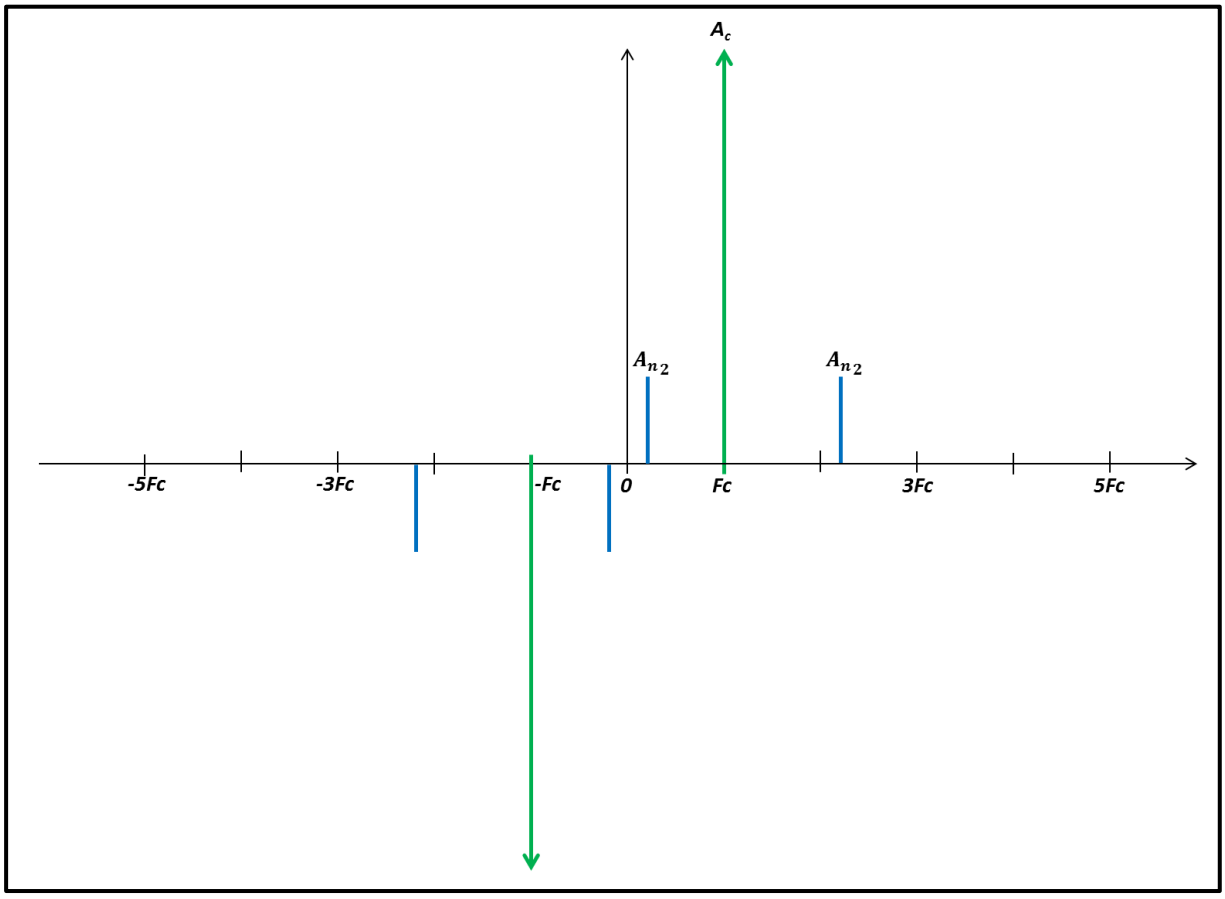


Figure 8-21: Phase Noise Modulation for Sinusoidal Signal

The jitter RMS of this noise is:

$$RMS_{Jitter_{multispurs}} = \frac{\sqrt{[(\frac{A_{n2}}{A_c})^2 + (\frac{-A_{n2}}{A_c})^2 + (\frac{A_{n2}}{-A_c})^2 + (\frac{-A_{n2}}{-A_c})^2]}}{2\pi F_c} = \frac{\sqrt{4 * \frac{{A_{n2}}^2}{{A_c}^2}}}{2\pi F_c} = \frac{2 * \frac{A_{n2}}{A_c}}{2\pi F_c} = \frac{2 * \frac{\beta}{2}}{2\pi F_c} = \frac{\beta}{2\pi F_c} \quad (8.87)$$

The noisy square signal is expressed as:

$$x_{square_{noise}} = V_0\frac{4}{\pi}(sin(\omega_c t)) + V_0\frac{4\beta}{2\pi}(sin(\omega_c t + \omega_{m2}t) - \sin(\omega_c t - \omega_{m2}t)) + \quad (8.88)$$

$$\frac{V_0}{3}\frac{4}{\pi}(sin(3\omega_c t)) + V_0\frac{4\beta}{2\pi}(sin(3\omega_c t + \omega_{m2}t) - sin(3\omega_c t - \omega_{m2}t)) +$$

$$\frac{V_0}{5}\frac{4}{\pi}(sin(5\omega_c t)) + V_0\frac{4\beta}{2\pi}(sin(5\omega_c t + \omega_{m2}t) - sin(5\omega_c t - \omega_{m2}t)) + \cdots$$

This is represented with Fig 8-22:

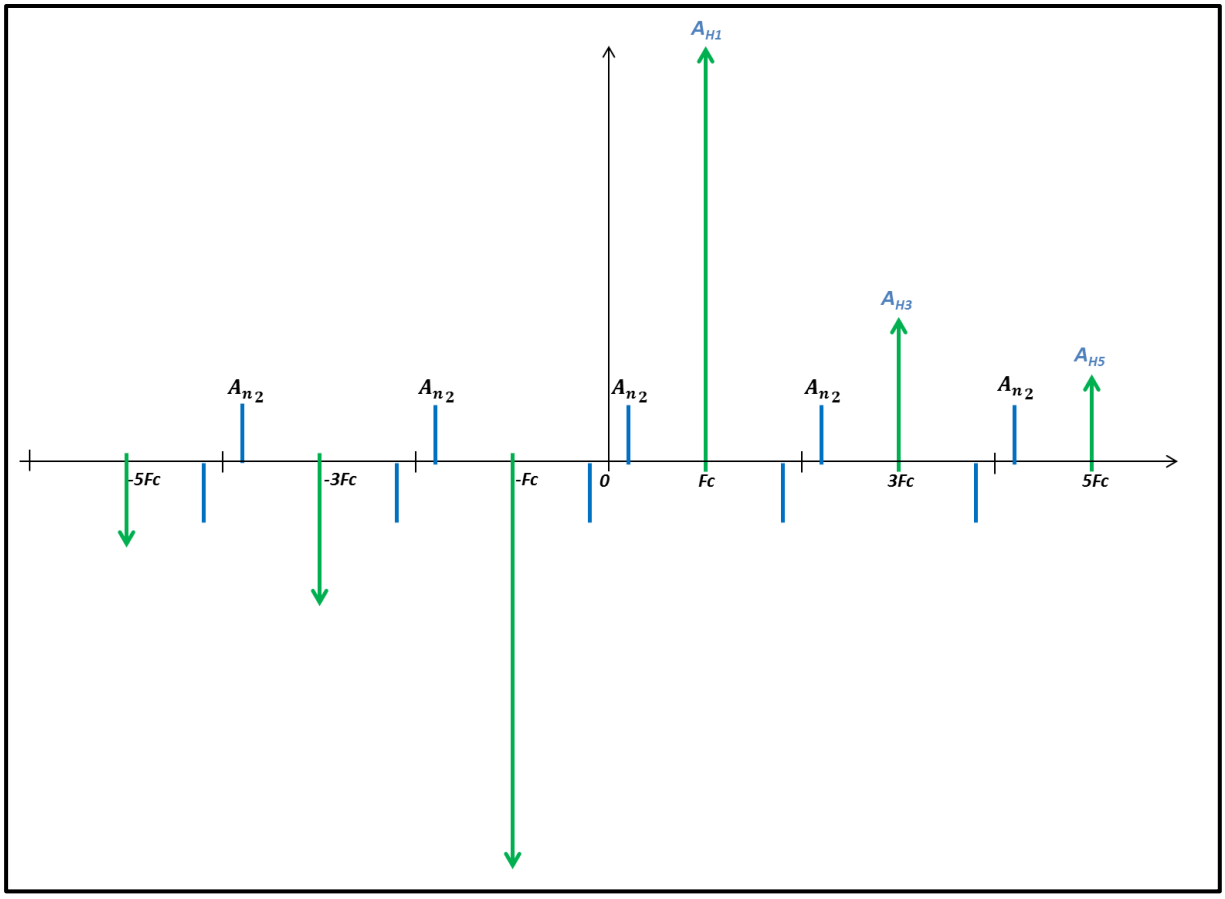


Figure 8-22: Phase Noise Modulation for Square Signal

The RMS Jitter (TIE) of the square signal should be same as the one of the sinusoidal signal. We will try to define integration band for which these 2 noises are equal.

For Harmonic 1

The Carrier amplitude is: $A_c = V_0 \frac{4}{\pi}$ The Noise amplitude is: $A_{n2} = V_0 \frac{4\beta}{\pi 2}$

RMS Jitter is found from 0 to 2*Fc. This is represented with Fig 8-23:

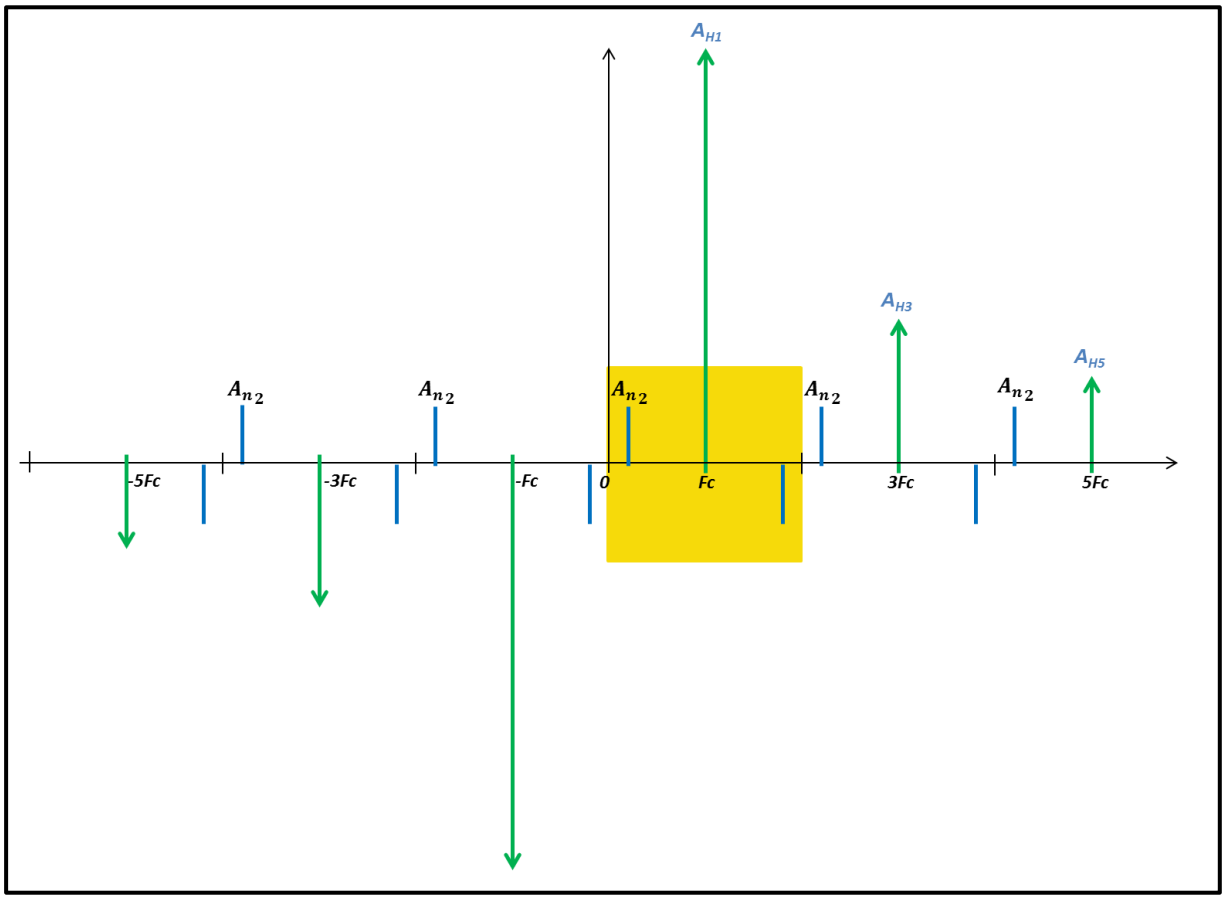


Figure 8-23: Phase Noise Modulation for Square Signal – Integration Band 0->2*Fc

The jitter RMS is found from 0 to 2*Fc:

$$RMS_{Jitter_{multispurs}} = \frac{\sqrt{2*[(\frac{A_{n2}}{A_{H1}})^2+(\frac{-A_{n2}}{A_{H1}})^2]}}{2\pi F_c} = \frac{2*\frac{A_{n2}}{A_{H1}}}{2\pi F_c} = \frac{2*\frac{\beta}{2}}{2\pi F_c} = \frac{\beta}{2\pi F_c} \quad (8.89)$$

We remark that when "$\omega_{m2} > F_c$" the global noise of sinusoidal signal is equal to the noise included around H_1 for the square signal, from 0 to 2Fc.

This demonstrates that for any ω_m, the noise spur is aliased from 0 to 2*Fc.

8.4.5 Simulations of noisy Sinusoidal / Square signal

The objective of this part is to study noise impact in sinusoidal and square signals, and confirm the integration bandwidths of sinusoidal/square signals, through different simulations.

3 types of simulations were done:

- Eldo simulations.
- Verilog-A simulations.
- Matlab simulations.

8.4.5.1 Eldo Simulation

The following simulation schematic Fig 8-24 was done in Eldo:

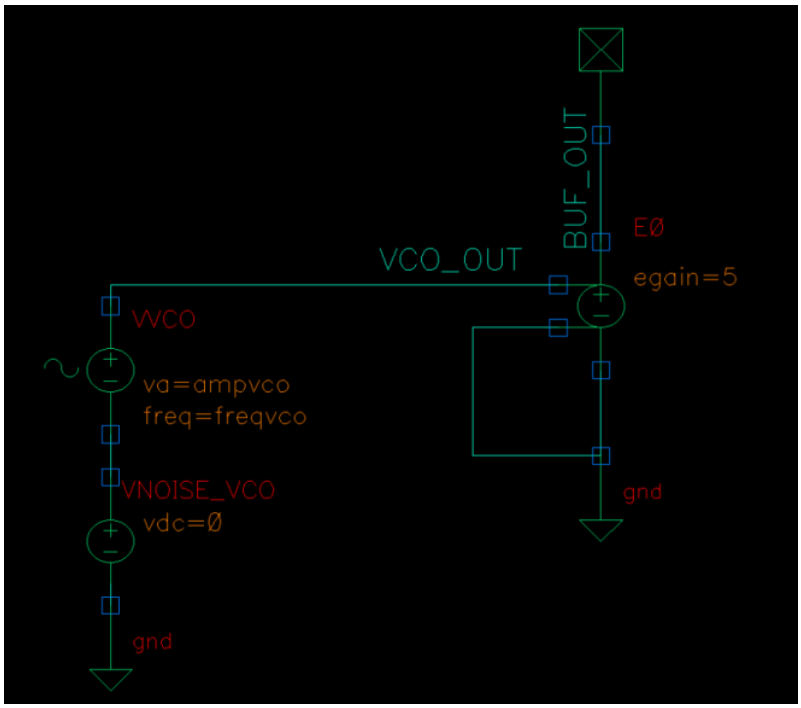


Figure 8-24: Eldo Simulation Schematic

The Oscillator block (VCO or crystal) generates a sinusoidal signal with a phase noise. The signal is sent to the transmitter thanks to a buffer. This buffer converts the sinusoidal signal into a square one. This schematic is equivalent to Fig 8-12. The TIE jitter for the sinusoidal signal and for the square signal should be same.

As seen in (2.22), we consider that the noise is constituted of Phase Noise & Amplitude Noise (50% - 50%)

We show on same graph, Fig 8-25, Phase & Amplitude Noise, for Sinusoidal and Square signal.

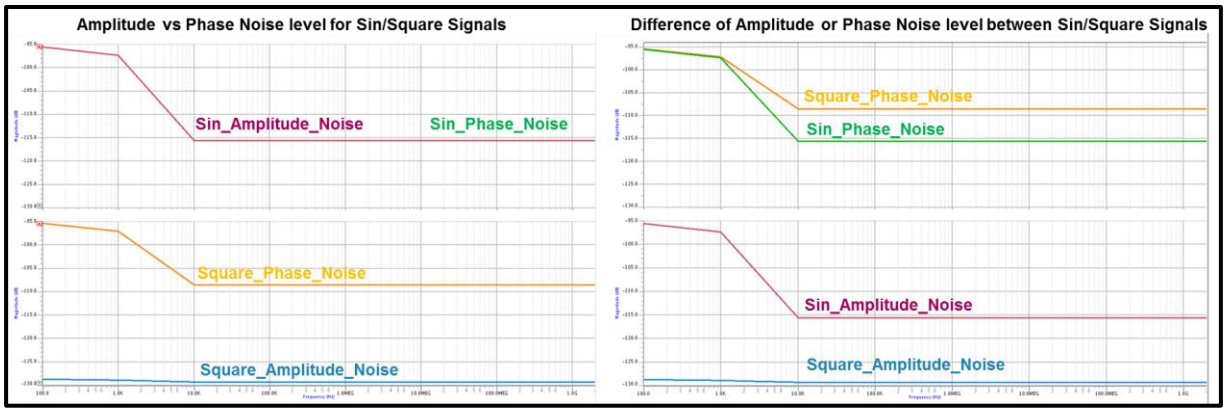


Figure 8-25: Amplitude & Phase noise for Sinusoidal/Square signal

We see in Fig 8-25 that we have the same level of amplitude & phase noise for sinusoidal signal (50%-50%). This is normal, and is in correlation with (2.22).

For square signal, phase noise profile is “higher” than the one of the sinusoidal signal and amplitude noise profile much lower (and decreasing). This is in correlation with explications in **Annexe 8.2**.

Above results are in correlation with **Annexe 8.4.3**. Because of the aliasing, the phase noise floor increases for the square signal.

Explain the fact that the phase noise floor increases for square signal.

The Noise Transfer function of a square signal is given in Fig 8-26:

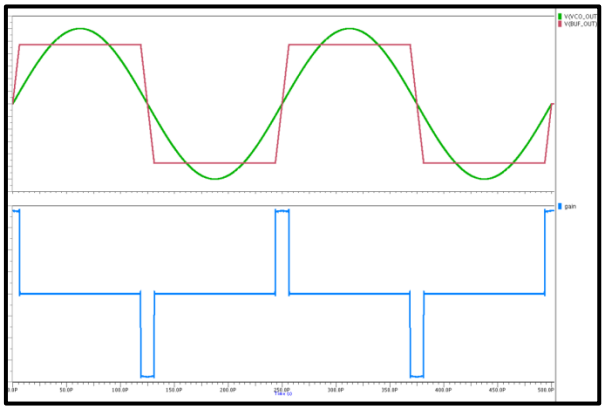


Figure 8-26: Noise Transfer Function of a square signal

If we calculate its FFT, and convolute with Phase Noise profile, we will have the impact of Fig 8-27:

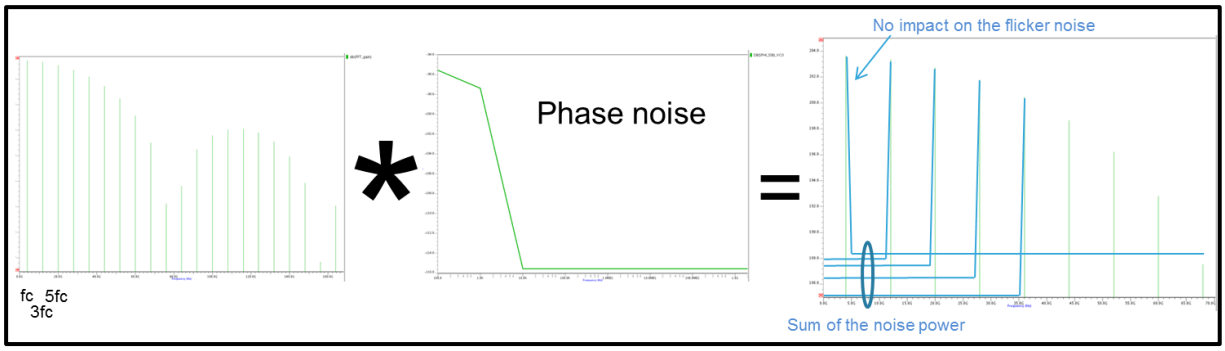


Figure 8-27: Convolute Phase noise profile with Noise Transfer Function FFT

As shown in Fig 8-27 we suppose that the quadratic sum of the aliasing noise powers will increase the phase noise profile for the square signal.

8.4.5.2 Verilog-A Simulation

To be sure we have only phase noise on the square signal, we use a Verilog-A VCO. We do a noisetran simulation to extract the TIE and an eye diagram. We use a resistor to generate the noise, as the noisetran doesn't support the noise table definition.

The results of the time simulations and the FFT of sin/square signals are given in Fig 8-28:

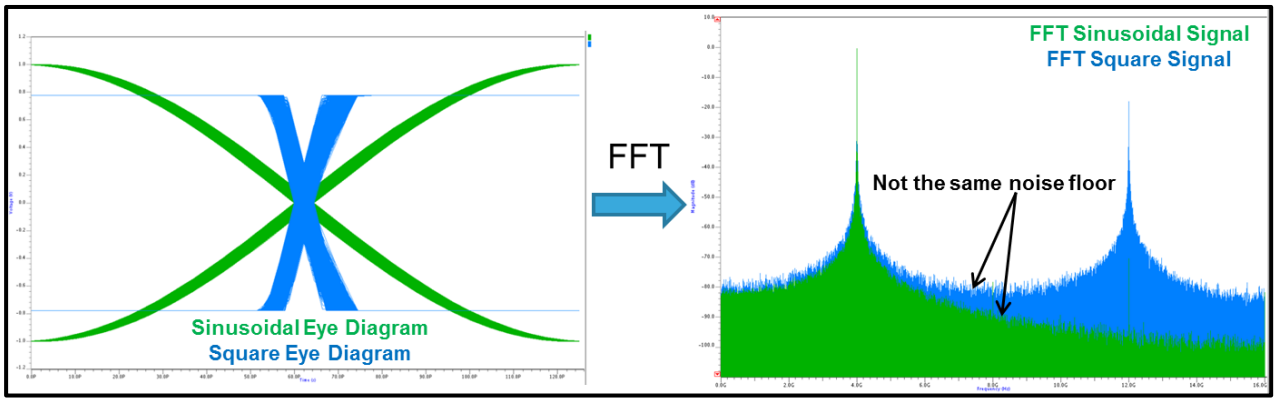


Figure 8-28: Sin/Square Signals Eye Diagrams & FFT

We remark in Fig 8-28 that noise floor is higher for square signal. (same conclusion as with Eldo Simulation of Fig 8-25).

The next step is to verify that the TIE Jitter is the same for sinusoidal and square signal. We show in Fig 8-29 the results of the TIE jitter for Sinusoidal and Square signal:

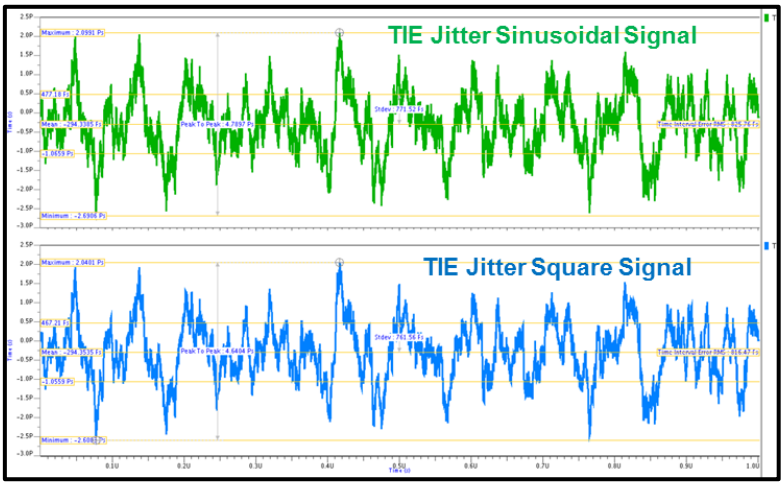


Figure 8-29: Sin/Square Signals TIE Jitter

We remark that the TIE Jitter is same, for sinusoidal & square signal. So the fact that the phase noise profile increases for the square signal, will not increase the TIE Jitter.

8.4.5.3 How to explain the fact that jitter is the same with sinusoidal & square signal?

The easiest explication is given with the graph of Fig 8-30:

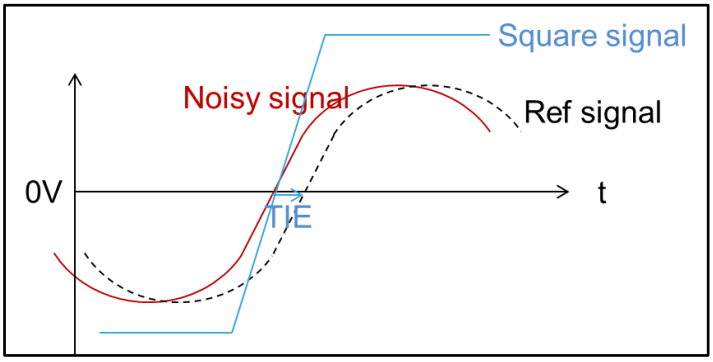


Figure 8-30: TIE for SIN/SQUARE signal

If we multiply the noisy sinusoidal signal by 5: 0V x 5 gives always 0V. The square signal will cross the 0V at the same time than the noisy sinusoidal signal. So there is no reason to change the TIE value.

8.4.5.4 Matlab Simulation

We will try to verify by Matlab simulations, that for square signal, noise is integrated up to 2*Fc. This is done in several steps:

- We create a square signal
- We create Phase Noise (limited within Fmin and Fmax)
- We realize frequency domain jitter calculations
- We realize time domain jitter calculations (independent to the frequency domain ones)
- Comparison of results is done.

8.4.5.4.1 Creating Square Noisy Signal:

As shown in ***2.4.2***, the amplitude noise has no impact in TIE jitter. We will create a square signal from a phase noise modulated sinusoidal signal, with a given Gain & Threshold, as shown in Fig 8-31:

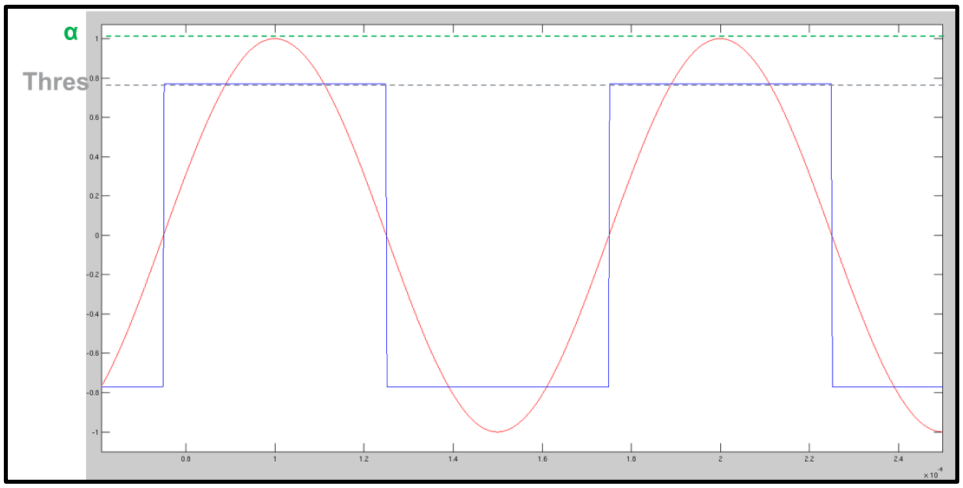


Figure 8-31: Creating Square Signal from Sinusoidal signal

In order to transform noisy Sinusoidal Signal into Square one, we used following equations:

$$Sinusoidal_{signal} = a * \cos(2\pi F_c t + Phase_{modulation}(t)) \quad (8.90)$$

$$Square_{signal} = Gain * Sinusoidal_{signal} \quad \text{(limited @ ±Threshold)} \quad (8.91)$$

In next step, we will check that we have the same noise power for sinusoidal or square signal, inside the integration bands we made hypothesis on.

- For sinusoidal signal we will integrate noise up to infinite.
- For square signal we will integrate noise up to $\frac{F_C}{2}$.

8.4.5.4.2 Creating Frequency Limited Phase Noise:

In order to verify the Integration band, we do following steps:

- First of all we create time random phase noise.
- We calculate FFT of these phase noise, to have their frequency position.
- Then we filter the phase noise frequencies, in order to have the noise only included between a defined minimum and maximum frequency band, as shown in Fig 8-32.

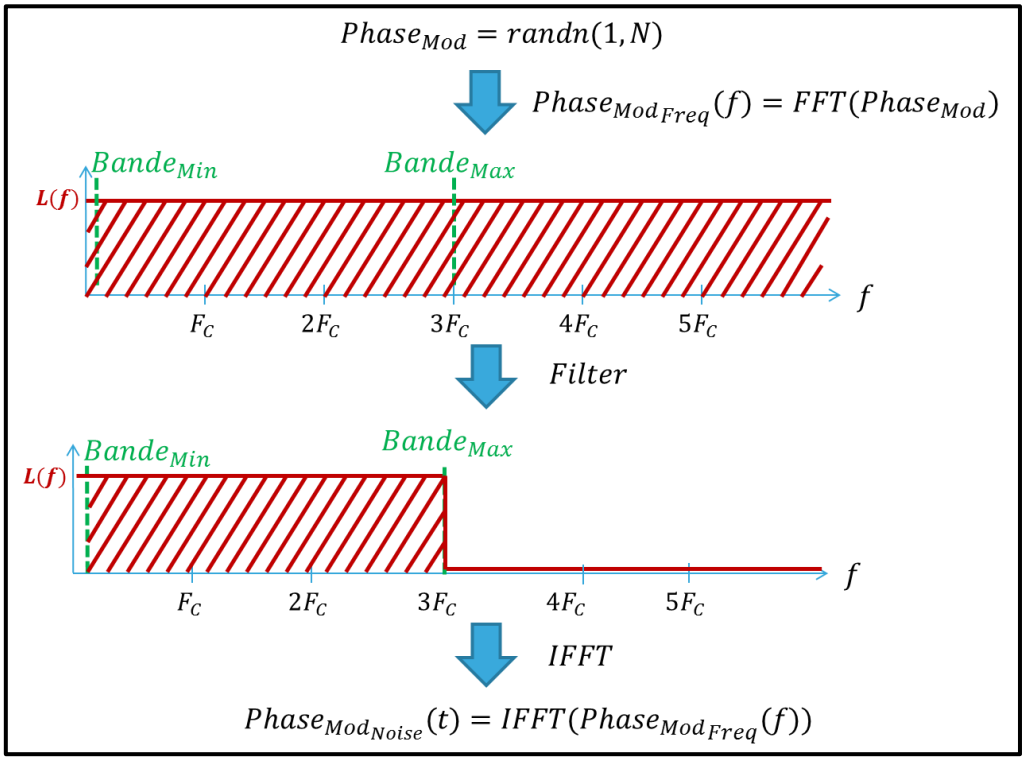


Figure 8-32: Creating frequency filtered phase noise

- At the end IFFT is calculated, to have the time phase noise values, included between F_{min} and F_{max}.

- Then the time sinusoidal signal is modulated with above calculated phase noise:

$$Sinusoidal_{noisy_{signal}} = \alpha \sin(\omega_c t + Phase_{Mod_{Noise}}(t)) \quad (8.92)$$

- Next step is to calculate square signal. This is done same as in (8.91).

At this point we have a sinusoidal signal, and a square signal, both phase modulated with a defined specific phase noise.

In order to calculate the power of each signal, we calculate the FFT of the sinusoidal & square signal.

The result is given in Fig 8-33. As seen In **Annexes 8.4.5.1**, we remark that the profile of phase noise of the square signal is higher than the one of the sinusoidal signal.

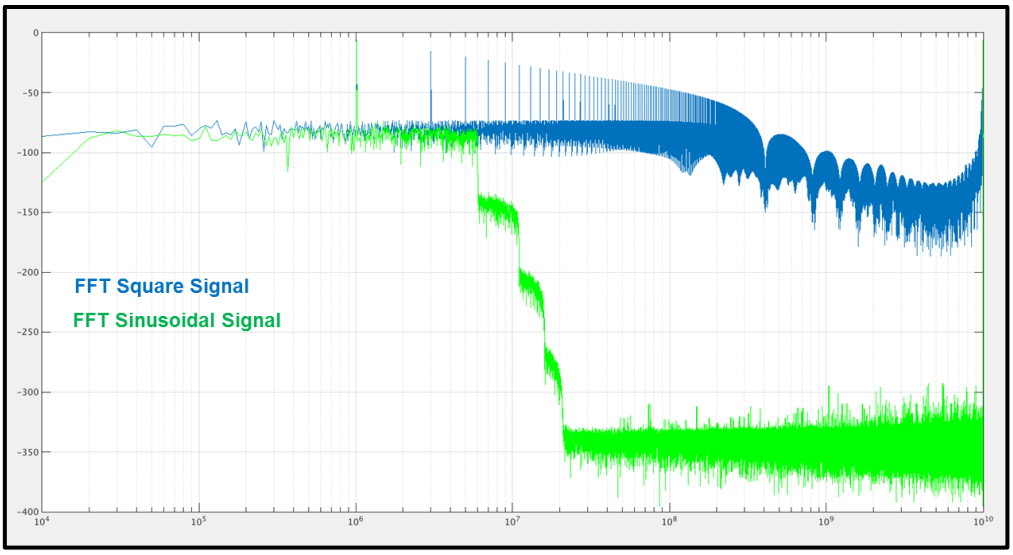


Figure 8-33: FFT of Sinusoidal & Square Noisy Signal - Matlab

This simulation is done for different noise frequency bands, in order to verify that the integration hypothesis is correct.

As example, we filtered phase noise for bands of noise from 0 to 10*Fc.

In all simulation tests, we measured power of noise for sinusoidal signal with integration band $-\infty \rightarrow +\infty$ (including negative frequencies, not shown on the graph), and for square signals, with integration band $0 \rightarrow 2 * Fc$ as in Fig 8-34.

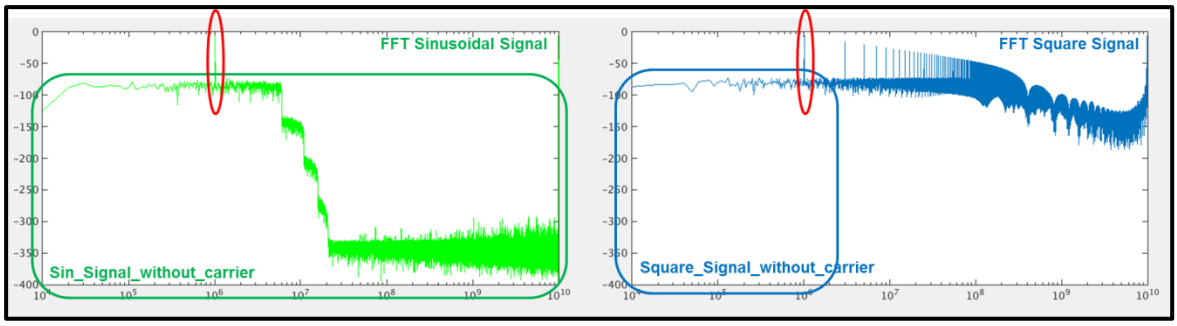


Figure 8-34: FFT of Sinusoidal & Square Noisy Signal - Matlab

Sinusoidal signal power in dBc is calculated with equation (8.93).

$$Power_{Sinusoidal_{Noise}}(dBc) = \sqrt{\frac{2*\sum_0^{\infty} Sin_{Signal_{NoiseWithoutCarrier_{Power}}}}{Sin_{Signal_{Carrier_{Power}}}}} \quad (8.93)$$

Square signal power in dBc is calculated with equation (8.94).

$$Power_{Square_{Noise}}(dBc) = \sqrt{\frac{2*(2*\sum_0^{2*Fc} Square_{Signal_{NoiseWithoutCarrier}})}{Square_{Signal_{Harmonic1_{Power}}}}} \quad (8.94)$$

In all simulations the power of noise was the same with sinusoidal/square signals.

This is a first proof that our hypothesis is correct.

The next step will be to calculate jitter in time/frequency domain, and compare them in order to see if we find the same jitter or not.

8.4.5.4.3 Frequency Domain Calculations:

The FFT of both signals is calculated, and given in Fig 8-35:

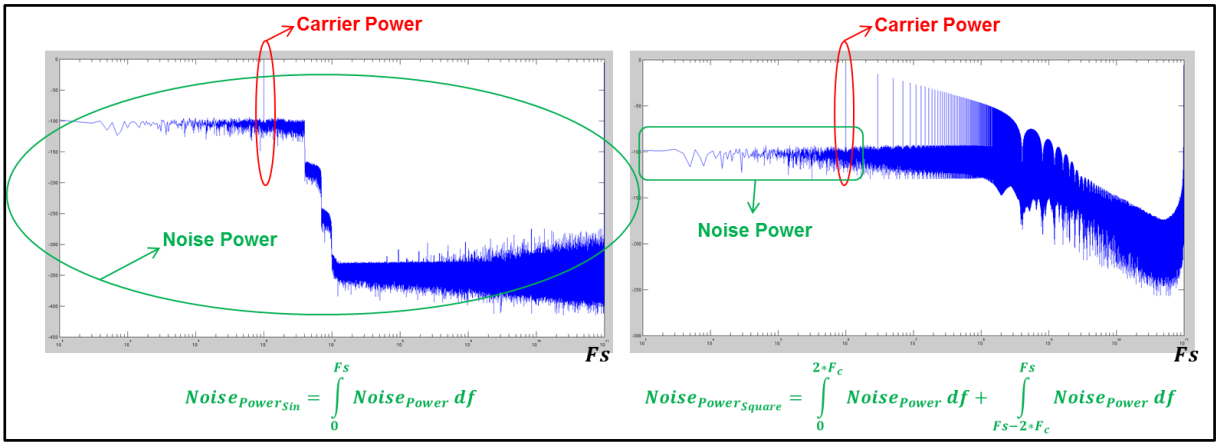


Figure 8-35: FFT of Sinusoidal & Square Noisy Signal - Matlab

For the sinusoidal signal we integrate up to infinite. For the square signal, up to Fc offset from Fc frequency (so up to 2*Fc).

Then we use equations to calculate Jitter:

$$Noise_{SinSig} = \sqrt{\frac{2*Noise_{Power_{SinSig}}}{Noise_{Power_{Carrier}}}} \qquad Noise_{SquareSig} = \sqrt{\frac{2*Noise_{Power_{SquareSig}}}{Noise_{Power_{Carrier}}}}$$

$$Jitter_{RMS_{SinSig}} = \frac{Noise_{SinSig}}{2\pi F_c} = 123ps \qquad Jitter_{RMS_{SinSig}} = \frac{Noise_{SinSquare}}{2\pi F_c} = 123ps$$

So we remark that the results are equal.

8.4.5.4.4 Time Domain Calculations:

We measure the TIE of time sine & square signals. This is done by measuring the transition positions. Then we measure the standard deviation of the TIE of both signals. Results are given as follow:

$$\sigma_{SinSig} = Std(Sin_{TIE}) = 123ps \qquad \sigma_{SquareSig} = Std(Square_{TIE}) = 123ps$$

8.4.5.5 Resuming Table:

We see on following Table 8-1, that for different noise filtering profiles, results between SIN or SQUARE RMS Jitter are same:

Table 8-1: Square Signal vs Time Signal Jitter

			FREQUENCY DOMAIN				TIME DOMAIN	
Test	Fmin	Fmax	Noise_SIN_dbc	Noise_SQUARE_dbc	Jitter_rms_SIN	Jitter_rms_SQUARE	Jitter_SIN	Jitter_SQUARE
1	0	0.5*Fc	1.00E-07	1.00E-07	5.03E-11	5.03E-11	5.04E-11	5.04E-11
2	0	Fc	1.93E-07	1.93E-07	6.99E-11	6.99E-11	6.99E-11	6.99E-11
3	0	1.5*Fc	2.90E-07	2.86E-07	8.57E-11	8.51E-11	8.51E-11	8.51E-11
4	0	2*Fc	3.85E-07	3.79E-07	9.88E-11	9.80E-11	9.82E-11	9.82E-11
5	0	2.5*Fc	4.90E-07	4.96E-07	1.11E-10	1.12E-10	1.12E-10	1.12E-10
6	0	3*Fc	5.82E-07	5.85E-07	1.21E-10	1.22E-10	1.22E-10	1.22E-10
7	0	3.5*Fc	6.84E-07	6.64E-07	1.32E-10	1.29E-10	1.30E-10	1.30E-10
8	0	4*Fc	8.03E-07	2.23E-07	1.43E-10	1.44E-10	1.45E-10	1.45E-10
9	0	4.5*Fc	9.07E-07	8.97E-07	1.52E-10	1.51E-10	1.51E-10	1.51E-10
10	0	5*Fc	1.00E-06	1.01E-06	1.60E-10	1.60E-10	1.61E-10	1.61E-10
11	0	6*Fc	1.20E-06	1.20E-06	1.74E-10	1.75E-10	1.75E-10	1.75E-10
12	0	7*Fc	1.38E-06	1.36E-06	1.87E-10	1.86E-10	1.87E-10	1.87E-10
13	0	8*Fc	1.56E-06	1.53E-06	1.99E-10	1.98E-10	1.98E-10	1.98E-10
14	0	9*Fc	1.78-e6	1.77E-06	2.12E-10	2.11E-10	2.12E-10	2.12E-10
15	0	10*Fc	1.98E-06	1.94E-06	2.24E-10	2.21E-10	2.22E-10	2.22E-10
16	3*Fc	5*Fc	4.10E-07	3.85E-07	1.02E-10	9.90E-11	9.90E-11	9.90E-11
17	3*Fc	6*Fc	5.91E-07	5.99E-07	1.22E-10	1.23E-10	1.24E-10	1.24E-10
18	3*Fc	7*Fc	8.16E-07	8.36E-07	1.44E-10	1.45E-10	1.46E-10	1.46E-10
19	3*Fc	8*Fc	1.01E-06	1.01E-06	1.60E-10	1.60E-10	1.61E-10	1.61E-10
20	3*Fc	9*Fc	1.18E-06	1.16E-06	1.73E-10	1.71E-10	1.72E-10	1.72E-10
21	3*Fc	10*Fc	1.43E-06	1.40E-06	1.90E-10	1.88E-10	1.89E-10	1.89E-10
22	6*Fc	7*Fc	1.97E-07	1.95E-07	7.06E-11	7.03E-11	7.06E-11	7.06E-11
23	6*Fc	8*Fc	4.00E-07	4.06E-07	1.00E-10	1.01E-10	1.02E-10	1.02E-10
24	6*Fc	9*Fc	6.19E-07	6.10E-07	1.25E-10	1.24E-10	1.25E-10	1.25E-10
25	6*Fc	10*Fc	8.00E-07	8.19E-07	1.43E-10	1.44E-10	1.45E-10	1.45E-10

The Table 8-1 confirms the theoretical equations.

We must integrate:

- Up to infinite for sinusoidal signal
- Up to 2Fc for square signal

8.5 Convolution of different Diracs:

In this section we give different examples of convolutions between different.

8.5.1 Convolution of two deterministic jitters:

The first step of our method (2.5.2) consists in defining the total deterministic jitter. This is done by convolving all DJ together. An example of the convolution between two DJ spurs is given in Fig 8-36.

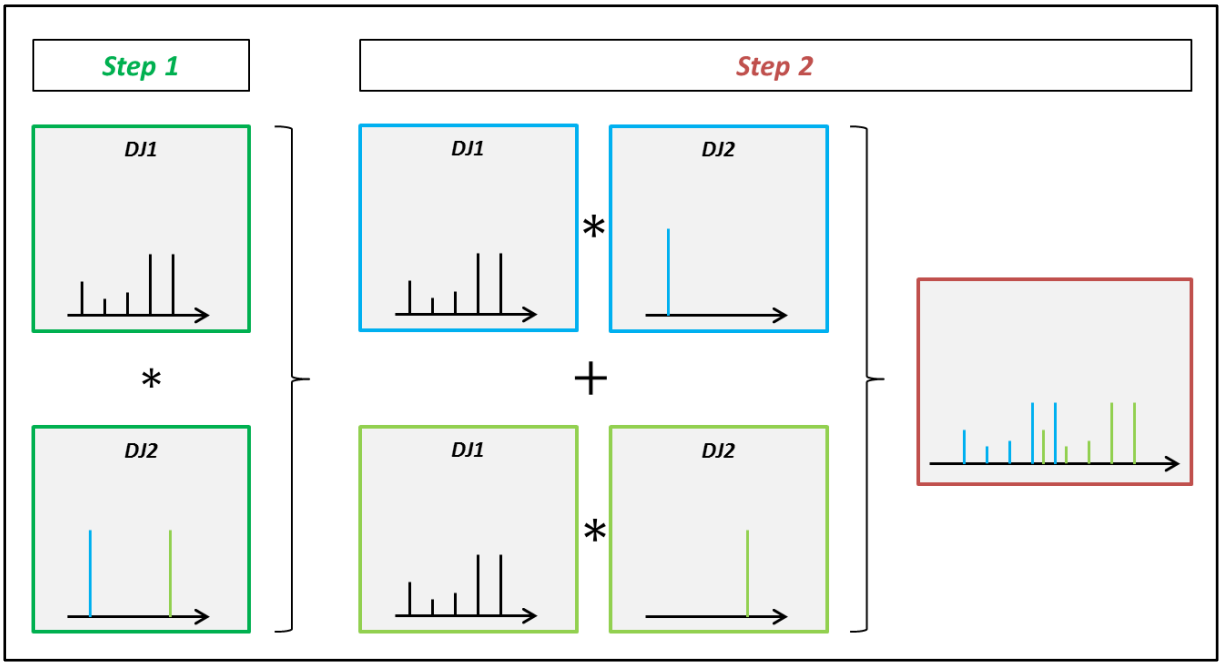


Figure 8-36: Convolution of two DJ Jitters

Step 1 consists in defining the Diracs of the two DJ jitters who will be convolved. All Jitter Diracs are normalized at 1, as given in (8.95).

$$\sum Diracs = 1 \tag{8.95}$$

Step 2 convolves each Dirac of DJ2 with all Diracs of DJ1.

- The X-axis position of the convolved vector will be the sum of X-positions of DJ1 and X-positions of DJ2.
- The Y-axis will be the multiplication of Y values of DJ1 and Y values of DJ2.

8.5.2 Convolution of 1*1 Diracs:

8.5.2.1 Example 1

Here we convolve 2 Dirac spurs with each other:

The 1st Dirac is found at 0ps.

The 2nd Dirac is found at 7ps.

The convolved Dirac is supposed to be found at 7ps (as given in Section 8.5.1, its X position is the sum of X-positions of the 2 other Diracs. So 0ps + 7ps = 7ps).

The result of the convolution is given in Fig 8-37.

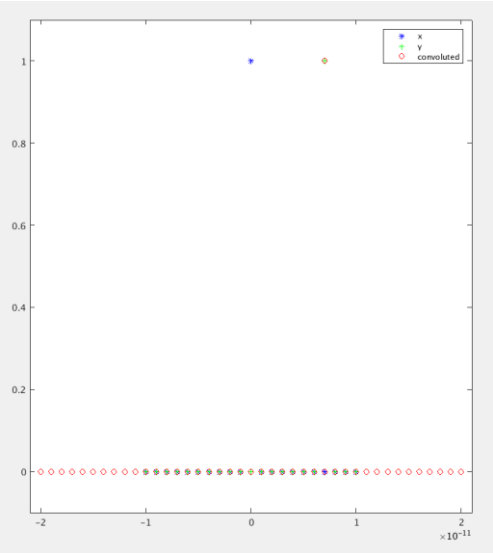


Figure 8-37: Example 1 - Convolution of 1*1 Dirac

As expected, we remark that the convolved Dirac result is found at 7ps of X axis position.

8.5.2.2 Example 2:

Here we convolve 2 Dirac spurs with each other:

The 1st Dirac is found at -5.5ps.

The 2nd Dirac is found at 7ps.

The convolved Dirac is supposed to be found at 1.5ps (X position is the sum of X-positions of the 2 other Diracs. So -5.5ps + 7ps = 1.5ps).

The result of the convolution is given in Fig 8-38.

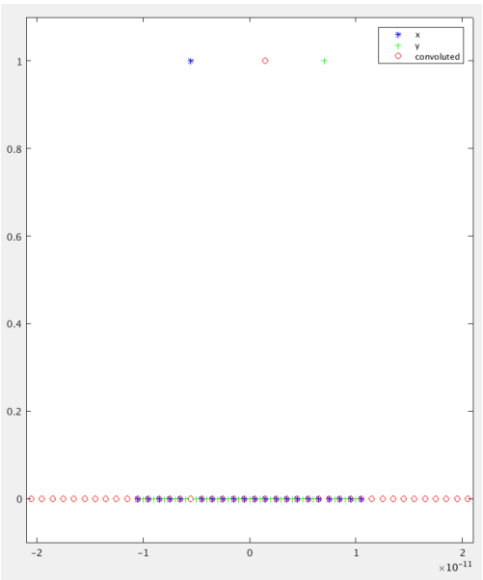


Figure 8-38: Example 2 - Convolution of 1*1 Dirac

As expected, we remark that the convolved Dirac result is found at 1.5ps of X axis position.

8.5.3 Convolution of multi*1 Diracs:

8.5.3.1 Example 1

Here we convolve 2 Dirac spurs with each Other:

The 1st Dirac of DJ1 is found at -7.5ps.

The 2nd Dirac of DJ1 is found at -5.5ps.

The 1st Dirac of DJ2 is found at 7ps.

The convolved Diracs are supposed to be found at -0.5ps and 1.5ps (its X Position is the sum of X-positions of the 2 other Diracs. So -7.5ps + 7ps = -0.5ps and -5.5ps + 7ps = 1.5ps).

The result of the convolution is given in Fig 8-39.

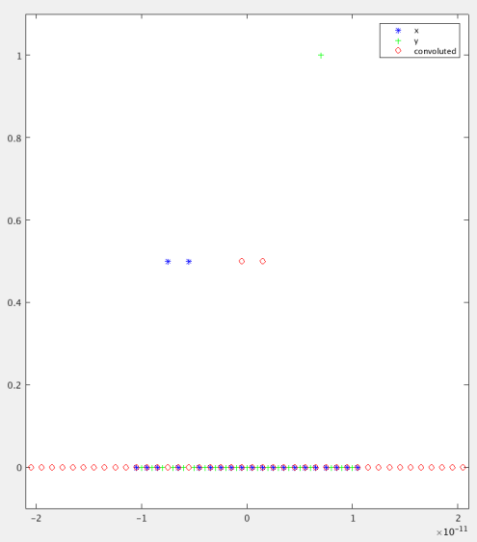


Figure 8-39: Example 1 - Convolution of multi*1 Dirac

As expected, we remark that the convolved Dirac results are found at -0.5ps and 1.5ps of X axis positions.

8.5.4 Conclusion

These examples prove that convolutions are done correctly between all Dirac spurs of the different DJ jitters.

List of acronyms

BER	Bit Error Rate
BW	Bandwidth
CDF	Cumulated Density Function
CCDF	Complementary Cumulated Density Function
CDR	Clock and Data Recovery
CL	Closed Loop
CTLE	Continuous Time Linear Equalization
DCD	Duty Cycle Distortion
DJ	Deterministic Jitter
DFE	Decision Feedback Equalization
FFT	Fast Fourier Transform
FoM	Figure of Merit
H1	1st Harmonic
HSSLs	High Speed Serial Links
IFFT	Inverse Fast Fourier Transform
IIR	Infinite Impulse Response
LPF	Low Pass Filter
OL	Open Loop
PCIe	PCI-Express
PDF	Probability Density Function
PFD	Phase-Frequency Detector
PLL	Phase Locked Loop
PP	Peak-to-Peak
PSD	Power Spectral Density
REF	Reference Clock

RF	Radio Frequency
RJ	Random Jitter
RMS	Root Mean Square
RX	Receiver
SD	Sigma-Delta
SJ	Sinusoidal Jitter
SSB	Single Side Banded
SSC	Spread Spectrum Clocking
TCL	Time Lag Correlation
TF	Transfer Function
TIE	Time Interval Error
TJ	Total Jitter
TX	Transmitter
USB	Universal Serial Bus
VCO	Voltage Controlled Oscillator

Bibliography

[Agilent] Agilent Technologies, Inc., “Jitter analysis: The dual-Dirac model, RJ/DJ, and Q-Scale”, White Paper

[Aleksić] M. Aleksić, "Extraction of jitter parameters from BER measurements," Electrical Performance of Electronic Packaging and Systems (EPEPS), 2011 IEEE 20th Conference on, San Jose, CA, 2011, pp. 63-66. doi: 10.1109/EPEPS.2011.6100187

[Analog] Analog Devices, Walt Kester, “Converting Oscillator Phase Noise to Time Jitter”, Tutorial

[Arakali] A. Arakali, S. Gondi and P. Kumar Hanumolu, "Low-Power Supply-Regulation Techniques for Ring Oscillators in Phase-Locked Loops Using a Split-Tuned Architecture," in IEEE Journal of Solid-State Circuits, vol. 44, no. 8, pp. 2169-2181, Aug. 2009. doi: 10.1109/JSSC.2009.2022916

[Bidaj] K. Bidaj, J. B. Begueret, N. Houdali, J. Deroo and S. Rieubon, "Time-domain PLL modeling and RJ/DJ jitter decomposition," 2016 IEEE International Symposium on Circuits and Systems (ISCAS), Montréal, QC, Canada, 2016, pp. 185-188. doi: 10.1109/ISCAS.2016.7527201

[Chow] D. Chow, M. Shimanouchi and M. P. Li, "Theory, model, and applications of non-Gaussian probability density functions for random jitter/noise with non-white power spectral densities," 2013 IEEE International Test Conference (ITC), Anaheim, CA, 2013, pp. 1-8. doi: 10.1109/TEST.2013.6651910

[Chu] A. Chu, N. Deo, W. Ahmad, M. Törmänen and H. Sjöland, "An ultra-low power charge-pump PLL with high temperature stability in 130 nm CMOS," New Circuits and Systems Conference (NEWCAS), 2015 IEEE 13th International, Grenoble, 2015, pp. 1-4.

[Dou_1] Q. Dou and J. A. Abraham, "Jitter Decomposition in High-Speed Communication Systems," 2008 13th European Test Symposium, Verbania, 2008, pp. 157-162. doi: 10.1109/ETS.2008.35

[Dou_2] Qingqi Dou and J. A. Abraham, "Jitter decomposition by time lag correlation," Quality Electronic Design, 2006. ISQED '06. 7th International Symposium on, San Jose, CA, 2006, pp. 6 pp.-530. doi: 10.1109/ISQED.2006.78

[Endo] T. Endo and J. Yokota, "Generation of White Noise by Using Chaos in Practical Phase-Locked Loop Integrated Circuit Module," Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on, New Orleans, LA, 2007, pp. 201-204. doi: 10.1109/ISCAS.2007.378311

[Erb] S. Erb and W. Pribyl, "Comparison of jitter decomposition methods for BER analysis of high-speed serial links," Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2010 IEEE 13th International Symposium on, Vienna, 2010, pp. 370-375. doi: 10.1109/DDECS.2010.5491751

[Feng] Feng Tan; Xianhe Huang; Wei Wei; Wei Fu, "Analysis of Phase Noise and Timing Jitter in Crystal Oscillator," in Communications, Circuits and Systems, 2007. ICCCAS 2007. International Conference on , vol., no., pp.1103-1106, 11-13 July 2007

[Ferdi] Y. Ferdi, A. Taleb-Ahmed and M. R. Lakehal, "Efficient Generation of Noise Using Signal Modeling Techniques," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 55, no. 6, pp. 1704-1710, July 2008. doi: 10.1109/TCSI.2008.918173

[Hansel] G. Hansel, K. Stieglbauer, G. Schulze and J. Moreira, "Implementation of an economic jitter compliance test for a multi-gigabit device on ATE," Test Conference, 2004. Proceedings. ITC 2004. International, 2004, pp. 1303-1312. doi: 10.1109/TEST.2004.1387405

[Hanumolu] P. K. Hanumolu, B. Casper, R. Mooney, Gu-Yeon Wei and Un-Ku Moon, "Jitter in high-speed serial and parallel links," Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on, 2004, pp. IV-425-8 Vol.4. doi: 10.1109/ISCAS.2004.1329031

[Herzel] F. Herzel, S. A. Osmany and J. C. Scheytt, "Analytical Phase-Noise Modeling and Charge Pump Optimization for Fractional- N PLLs," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 57, no. 8, pp. 1914-1924, Aug. 2010. doi: 10.1109/TCSI.2009.2039832

[HFE] High Frequency Electronics, Johnnie Hancock, April 2004, Summit Technical Media, LLC: “Jitter – Understanding it, Measuring It, Eliminating It”

[Hong] D. Hong and K. T. Cheng, "An Accurate Jitter Estimation Technique for Efficient High Speed I/O Testing," 16th Asian Test Symposium (ATS 2007), Beijing, 2007, pp. 224-229. doi: 10.1109/ATS.2007.77

[Howe] Howe, D.A.; Tasset, T.N., "Clock jitter estimation based on PM noise measurements," in Frequency Control Symposium and PDA Exhibition Jointly with the 17th European Frequency and Time Forum, 2003. Proceedings of the 2003 IEEE International , vol., no., pp.541-546, 4-8 May 2003

[İspir] M. İspir, Ö Özdil and A. Yıldırım, "Coloured noise generation with IFFT," Intelligent Signal Processing Conference 2013 (ISP 2013), IET, London, 2013, pp. 1-4. doi: 10.1049/cp.2013.2062

[Kafadar] K. Kafadar, "Gaussian white-noise generation for digital signal synthesis," in IEEE Transactions on Instrumentation and Measurement, vol. IM-35, no. 4, pp. 492-495, Dec. 1986. doi: 10.1109/TIM.1986.6499122

[Kang] Minsu Kang, "FPGA implementation of Gaussian-distributed pseudo-random number generator," Digital Content, Multimedia Technology and its Applications (IDC), 2010 6th International Conference on, Seoul, 2010, pp. 11-13.

[Kasdin] N. J. Kasdin, "Discrete simulation of colored noise and stochastic processes and 1/fα power law noise generation," in Proceedings of the IEEE, vol. 83, no. 5, pp. 802-827, May 1995. doi: 10.1109/5.381848

[Kennedy] M. P. Kennedy, H. Mo, Z. Huang and J. P. Lana, "A method to quantify the dependence of spur heights on offset current in a CP-PLL," 2016 IEEE International Symposium on Circuits and Systems (ISCAS), Montr¿¿al, QC, Canada, 2016, pp. 1658-1661.

[Kho] J. Kho and T. Y. Ling, "Fast and accurate technique to decompose jitter for very long pattern length waveform," 2014 IEEE Electrical Design of Advanced Packaging & Systems Symposium (EDAPS), Bangalore, 2014, pp. 93-96. doi: 10.1109/EDAPS.2014.7030815

[Kieffer] Kieffer, J.; Houdebine, M.; Dedieu, S.; Novakov, E., “Nonlinear PLL Model for Simulation of Noise and Modulation Mixing in Polar Architecture,” Xiangtan, China. ICCECT 2013, Dec 2013, pp.1-4

[Kim] Kyung Ki Kim, Jing Huang, Yong-Bin Kim and F. Lombardi, "On the modeling and analysis of jitter in ATE using Matlab," 20th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'05), 2005, pp. 285-293. doi: 10.1109/DFTVS.2005.52

[Kuo] C. Y. Kuo and J. L. Huang, "A period tracking based on-chip sinusoidal jitter extraction technique," 24th IEEE VLSI Test Symposium, Berkeley, CA, 2006, pp. 6 pp.-405. doi: 10.1109/VTS.2006.10

[LeCroy] LeCroy Corporation, Michael Schnecker, Jitter Measurements in Serial Data Signals, Whitepaper.

[Lee] Dong-U Lee, W. Luk, J. D. Villasenor, Guanglie Zhang and P. H. W. Leong, "A hardware Gaussian noise generator using the Wallace method," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 13, no. 8, pp. 911-920, Aug. 2005. doi: 10.1109/TVLSI.2005.853615

[Li] M. P. Li, J. Wilstrup, R. Jessen and D. Petrich, "A new method for jitter decomposition through its distribution tail fitting," Test Conference, 1999. Proceedings. International, Atlantic City, NJ, 1999, pp. 788-794. doi: 10.1109/TEST.1999.805809

[Maxim] Maxim Integrated, Application Note 3359: Clock (CLK) Jitter and Phase Noise Conversion

[Mistry] Mistry, D.; Joshi, S.; Agrawal, N., "A novel jitter separation method based on Gaussian mixture model," in Pervasive Computing (ICPC), 2015 International Conference on , vol., no., pp.1-4, 8-10 Jan. 2015

[Murch] A. R. Murch and R. H. T. Bates, "Colored noise generation through deterministic chaos," in IEEE Transactions on Circuits and Systems, vol. 37, no. 5, pp. 608-613, May 1990. doi: 10.1109/31.54997

[Nonis] R. Nonis, N. Da Dalt, P. Palestri and L. Selmi, "Modeling, design and characterization of a new low-jitter analog dual tuning LC-VCO PLL architecture," in IEEE Journal of Solid-State Circuits, vol. 40, no. 6, pp. 1303-1309, June 2005. doi: 10.1109/JSSC.2005.848037

[Ong] Chee-Kian Ong, Dongwoo Hong, Kwang-Ting Cheng and L. C. Wang, "A scalable on-chip jitter extraction technique," VLSI Test Symposium, 2004. Proceedings. 22nd IEEE, 2004, pp. 267-272. doi: 10.1109/VTEST.2004.1299253

[Palermo] Prof. Palermo, Texas A&M University, ECEN 720: High Speed Serial Links and Systems, Lecture.1 http://www.ece.tamu.edu/~spalermo/ecen720.html

[Papoulis] Athanasios Papoulis, S. Unnikrishna Pillai, “Probability, Random Variables and Stochastic Processes”, McGraw-Hill 4th edition

[Park] Jongsun Park, K. Muhammad and K. Roy, "Efficient modeling of 1/f/sup /spl alpha// noise using multirate process," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 25, no. 7, pp. 1247-1256, July 2006. doi: 10.1109/TCAD.2005.855953

[PCIe] http://pcisig.com/specifications - PCI Express Base Specification Revision 3.1.pdf

[Pu] X. Pu, A. Kumar and K. Nagaraj, "Area-Efficient Low-Noise Low-Spur Architecture for an Analog PLL Working From a Low Frequency Reference," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 59, no. 6, pp. 331-335, June 2012.

[Sadeghi] V. S. Sadeghi, H. Miar Naimi and M. P. Kennedy, "The Role of Charge Pump Mismatch in the Generation of Integer Boundary Spurs in Fractional-N Frequency Synthesizers: Why Worse Can Be Better," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 60, no. 12, pp. 862-866, Dec. 2013. doi: 10.1109/TCSII.2013.2285968

[Schober] S. Schober and J. Choma, "A 1.25mW 0.8–28.2GHz charge pump PLL with 0.82ps RMS jitter in all-digital 40nm CMOS," 2015 IEEE International Symposium on Circuits and Systems (ISCAS), Lisbon, 2015, pp. 549-552. doi: 10.1109/ISCAS.2015.7168692

[Sharma] V. K. Sharma, J. N. Tripathi, R. Nagpal, S. Deb and R. Malik, "A comparative analysis of jitter estimation techniques," Electronics,Communication and Computational Engineering (ICECCE), 2014 International Conference on, Hosur, 2014, pp. 125-130. doi: 10.1109/ICECCE.2014.7086645

[Soyuer] M. Soyuer and R. G. Meyer, "High-frequency phase-locked loops in monolithic bipolar technology," in IEEE Journal of Solid-State Circuits, vol. 24, no. 3, pp. 787-795, Jun 1989. doi: 10.1109/4.32041

[Stauffer] D. R. Stauffer, J. T. Mechler, M. Sorna, K. Dramstad, C. R. Ogilvie, A. Mohammad, J. Rockrohr, "High Speed Serdes Devices and Applications", Springer

[Sui] C. Sui, D. G. Beetner, T. Zhu and C. Cheng, "A new tail-fit method design for jitter decomposition," 2014 IEEE International Symposium on Electromagnetic Compatibility (EMC), Raleigh, NC, 2014, pp. 423-427. doi: 10.1109/ISEMC.2014.6899009

[Tektronix] Tektronic, Ransom Stephens (October 2006), Jitter 360° Knowledge Series, Part 2: "What the Dual-Dirac Model is and What it is Not".

[Telba] A. Telba, J. M. Noras, M. Abou El Ela and B. Almashary, "Simulation technique for noise and timing jitter in phase locked loop," Microelectronics, 2004. ICM 2004 Proceedings. The 16th International Conference on, 2004, pp. 501-504. doi: 10.1109/ICM.2004.1434709

[Ting] S. K. Ting, D. Cabric and A. H. Sayed, "An Algorithm to Compensate the Effects of Spurious PLL Tones in Spectrum Sensing Architectures," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 59, no. 5, pp. 1093-1106, May 2012.

[Tripathi] J. N. Tripathi, V. K. Sharma, H. Advani, P. N. Singh, H. Shrimali and R. Malik, "An analysis of power supply induced jitter for a voltage mode driver in high speed serial links," 2016 IEEE 20th Workshop on Signal and Power Integrity (SPI), Turin, 2016, pp. 1-4. doi: 10.1109/SaPIW.2016.7496259

[Tzou] N. L. Tzou, D. Bhatta, S. W. Hsiao and A. Chatterjee, "Periodic jitter and bounded uncorrelated jitter decomposition using incoherent undersampling," Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013, Grenoble, France, 2013, pp. 1667-1672. doi: 10.7873/DATE.2013.337

[USB] http://www.usb.org/developers/docs - Universal Serial Bus Revision 3.1 Specification (.zip file) - USB_3_1_r1.0.pdf

[Wisetphanichkij] S. Wisetphanichkij and K. Dejhan, "Jitter decomposition by derivatived Gaussian wavelet transform," Communications and Information Technology, 2004. ISCIT 2004. IEEE International Symposium on, 2004, pp. 1160-1165 vol.2. doi: 10.1109/ISCIT.2004.1413901

[Wu] T. Wu, P. K. Hanumolu, K. Mayaram and U. K. Moon, "Method for a Constant Loop Bandwidth in LC-VCO PLL Frequency Synthesizers," in IEEE Journal of Solid-State Circuits, vol. 44, no. 2, pp. 427-435, Feb. 2009. doi: 10.1109/JSSC.2008.2010792

[Ye] Zhipeng Ye and M. P. Kennedy, "Noise reduction in fractional-N frequency synthesizers with multiphase VCO," Research in Microelectronics and Electronics Conference, 2007. PRIME 2007. Ph.D., Bordeaux, 2007, pp. 173-176. doi: 10.1109/RME.2007.4401840

[Yonsei_1] Yonsei University, High-Speed Circuits and Systems Lab., Lect.1 http://tera.yonsei.ac.kr/class/2013_1_2/lecture/Lect1%20Introduction.pdf

[Young] I. A. Young, J. K. Greason and K. L. Wong, "A PLL clock generator with 5 to 110 MHz of lock range for microprocessors," in IEEE Journal of Solid-State Circuits, vol. 27, no. 11, pp. 1599-1607, Nov. 1992. doi: 10.1109/4.165341

[Yuan] Boqun Yuan and Huihua Liu, "A generalized low power and lower jitter charge pump PLL," Microwave and Millimeter Wave Circuits and System Technology (MMWCST), 2013 International Workshop on, Chengdu, 2013, pp. 471-474. doi: 10.1109/MMWCST.2013.6814554

[Zamek] Zamek, I.; Zamek, S., "Definitions of jitter measurement terms and relationships," in Test Conference, 2005. Proceedings. ITC 2005. IEEE International , vol., no., pp.10 pp.-34, 8-8 Nov. 2005

[Zao] L. Zao and R. Coelho, "Generation of coloured acoustic noise samples with non-Gaussian distributions," in IET Signal Processing, vol. 6, no. 7, pp. 684-688, September 2012. doi: 10.1049/iet-spr.2011.0216

[Zhang] Jianmin Zhang, David J. Pommerenke, UMR EMC Laboratory, “Jitter”: http://bbs.hwrf.com.cn/downpcbe/Jitter-11095.pdf

Publications related to this work

Conferences

- ***"Time-Domain PLL Modeling and RJ/DJ Jitter Decomposition"***

  IEEE International symposium on circuits and systems, ISCAS 2016, Montréal.

- **"RJ/DJ Jitter Decomposition for High Speed Links"**

  IEEE International Conference on Electronics Circuits and Systems, ICECS 2016, Monte Carlo, Monaco.

Journals

- ***"Generation of Colored Noise Patterns with Gaussian Distribution"***

  Submitted, Notification Pending

  IEEE Transactions on Instrumentation and Measurement.

Résumé de la thèse:

La vitesse des liens séries haut débit (USB, SATA, PCI-express, etc.) a atteint des débits de plusieurs gigabits par seconde, et continue à augmenter. Deux des principaux paramètres électriques utilisés pour caractériser les performances des SerDes sont le jitter transmis à un niveau de taux d'erreur donné et la capacité du récepteur à suivre le jitter à un taux d'erreur donné.

Modéliser le bruit de phase des différents composants du SerDes et extraire le jitter temporel pour le décomposer, aideraient les ingénieurs en conception de circuits à atteindre des meilleurs résultats pour les futures versions des SerDes. De plus, générer des patterns de jitter correspondant à des bruits blancs ou colorés permettrait de mieux analyser ces effets dans le système pendant la phase de vérification.

La boucle d'asservissement de phase (PLL) est un des contributeurs principaux du jitter d'horloge aléatoire et déterministe à l'intérieur du système. Pendant les 3 années de recherche j'ai développé une méthode pour modéliser la PLL (en temporel et en fréquentiel) avec injection du bruit de phase et estimation du jitter.

Nous avions 3 objectifs principaux pour notre étude :

- Notre 1er objectif était d'acquérir une connaissance solide sur le jitter, le bruit de phase et l'analyse temps/fréquence.
- Notre 2ème objectif était de fournir aux designers des outils pendant la phase de design, pour contraindre le système proprement et estimer le jitter de chaque sous-bloc de la PLL.
- Notre 3ème objectif était de fournir aux designers des outils pendant la phase de validation du circuit pour décomposer le jitter en aléatoire (RJ) et déterministe (DJ), et identifier les sources de ce dernier afin d'améliorer les performances du système.

J’ai donc développé un modèle de la PLL dans le domaine fréquentiel. Les profils de bruits de phase correspondant à chacun des sous-blocs de la PLL sont obtenus à partir de simulations ELDO. Le modèle fréquentiel est un modèle petit signal qui fonctionne avec des fonctions de transfert, et estime le jitter RMS du système, à la sortie de la PLL.

Pour pouvoir inclure les non-linéarités de la boucle, et estimer le jitter temporel, j’ai créé un modèle temporel de la PLL. Les caractéristiques de ce modèle sont les suivantes :

- il convertit les profils de bruit de phase en jitter temporel.
- ces jitters temporels correspondant à chacun des sous-blocs de la PLL sont injectés dans la boucle de la PLL.
- Le modèle estime le jitter temporel à la sortie de la PLL.
- Le jitter RMS peut être calculé avec l’écart-type du jitter temporel.

Le modèle fréquentiel de la PLL nous a permis de valider le modèle temporel en termes de jitter RMS, en comparant les résultats, avec une différence maximale de 1.6%.

A partir de mesures en laboratoire d’un transmetteur (TX d’un SerDes) nous pouvons observer deux caractéristiques :

- Le jitter a un profil spectral coloré.
- Le jitter a des propriétés de distribution Gaussienne.

Pour pouvoir analyser les caractéristiques d’un système SerDes, il est important de générer des patterns de jitter colorés avec une distribution Gaussienne. Ces patterns seront utilisés pour prédire l’impact du jitter sur les performances du système avec la simulation dans le domaine temporel pendant la phase de vérification. Ils seront également utilisés pour définir les budgets en termes de jitter, pour les différents blocs du système SerDes.

Nous avons proposé pour la 1ère fois dans la littérature une nouvelle méthode pour générer des patterns synthétiques de jitter avec une distribution Gaussienne à partir de profils de bruit de phase coloré.

Les standards spécifient des budgets séparés pour le RJ et DJ. Pour décomposer et analyser le jitter de la sortie de la PLL en RJ & DJ, une nouvelle technique a été mise au point. Les résultats de modélisation corrèlent bien avec les mesures et cette technique aidera les ingénieurs de conception à identifier et quantifier proprement les sources du jitter ainsi que leurs impacts dans les systèmes SerDes.

Grace à ces modèles, nous avons développé une méthode pour spécifier la PLL en termes de bruit de phase. Cette méthode est applicable à tout standard (USB, SATA, PCIe, …) et définit les profils de bruit de phase pour les différentes parties de la PLL, pour s'assurer que les requis des standards sont satisfaits en termes de jitter.