Architectural Improvements Towards an Efficient 16-18 Bit 100-200 MSPS ADC

Author: Francesco Zanini
Academic Supervisor: Dr. Ronan Farrell
Head of Department: Dr. Frank Devitt

Master Engineering Science Thesis

Institute of Microelectronics and Wireless Systems
Department of Electronic Engineering
National University of Ireland, Maynooth
Ireland

10th May 2007

Dedicated to

All the people who helped me to reach this result

Abstract

As Data conversion systems continue to improve in speed and resolution, increasing demands are placed on the performance of high-speed Analog to Digital Conversion systems. This work makes a survey about all these and proposes a suitable architecture in order to achieve the desired specifications of 100-200MS/s with 16-18 bit of resolution. The main architecture is based on paralleled structures in order to achieve high sampling rate and at the same time high resolution. In order to solve problems related to Time-interleaved architectures, an advanced randomization method was introduced. It combines randomization and spectral shaping of mismatches. With a simple low-pass filter the method can, compared to conventional randomization algorithms, improve the SFDR as well as the SINAD. The main advantage of this technique over previous ones is that, because the algorithm only need that ADCs are ordered basing on their time mismatches, the absolute accuracy of the mismatch identification method does not matter and, therefore, the requirements on the timing mismatch identification are very low. In addition to that, this correction system uses very simple algorithms able to correct not only for time but also for gain and offset mismatches.

Declaration

The work in this thesis is based on research carried out at the Institute of Microelectronics and Wireless Systems, department of Electronic Engineering of the National University of Ireland, Maynooth, Ireland.

Acknowledgements

The author would like to thank SFI and CTVR funded centre and also all the people of the institute of Microelectronics and Wireless Systems for their technical support. A special "thank you" goes to Dr. Ronan Farrell for his help during the developing of the project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Nowadays the market of radio frequency communication devices is rapidly expanding. analog to digital converters (ADC) represents one of the bottlenecks of the performance of all these systems. ADCs in fact are key elements that are able to let the digital or the "brain" section able to communicate with the "arms" or the radio frequency analog signals of any telecommunication system. The aim of this thesis is present a method of utilising parallelised groups of lower performance ADCs to achieve the required performance. While theoretically this is possible, practical issues in matching the performance of the individual ADCs results in severely degraded performance. This thesis presents an analysis of these problems and proposes a new method for minimising these mismatch effects and thus allowing the theoretical performance to be full obtained.

1.2 Research goals

This research seeks to address issues of improvements of performance parameters in parallelised analog to digital converter. The aim is to find a new architecture able to achieve performances higher than previous ones possibly with less power consumption and area occupation in order to decrease the price and improve the reliability and the portability of the system. Key research contributions and results

are summarized below:

- Investigated limitations of current ADC architectures with particular reference to the bottleneck of these systems. The linearity limitations of input stages have been analyzed as well as conversion law non-linearities.
- Presented a new architectural solution able to overcome and to solve state of the art performance limitations.
- Investigated the state of the art of solutions for problems and limitations in parallelized architectures.
- Proposed a new solution able to outperform previous ones. The new solution is an advanced randomization method that combines randomization and spectral shaping of mismatches. With a simple low-pass filter the method can improve the SFDR as well as the SINAD.
- Analyzed the efficacy and the ease of implementation for this new method.

1.3 Thesis organization

Chapter 2 provides a brief introduction to Analog to Digital Converters (ADCs). Current state of the art of ADCs and their related problems are described respectively in section 2 and 3, while an introduction is made in section 1. Chapter 3 makes a comparison between state of the art techniques for distortion correction in time-interleaved architectures. An introduction is provided in section 1 while state of the art solutions are presented in section 2, 3 and 4. Section 2 describes spurs prevention techniques, section 3 explains identification&correction ones while section 4 introduces the reader to mismatch shaping algorithms. In chapter 4 the proposed technique for mismatch error correction is presented. Section 1 describes the system focusing only on time mismatches. Subsections from 1 to 4 describe the main idea, make the optimality mathematical proof of the system and describe obtained results. Section 2 describes the system and extends the overall analysis made in previous section also to gain and offset mismatches. Results are shown from different point

of view and perspectives. Appendices show softwares mathematical analysis made in order to be able to simulate the system and some theoretical extensions made. Appendix A shows how to perform a simulation in order to test linearity parameters of the input stage of ADCs while appendix B shows MATLAB®code written in order to simulate the proposed algorithm for mismatch compensation in parallelized architectures. Appendix C makes some considerations in order to improve the T&H section of ADCs.

Chapter 2

Introduction to A/D Converters

In this chapter background information are given about the work presented in following chapters

2.1 Functional description and main performance parameters

Analog to digital converters (ADCs) are ubiquitous, critical components of software radio and other signal processing systems. Every day, every person uses at least one of these devices. Figure 2.1 just shows some applications of these key devices. An analog to digital converter (ADC) is a device that converts real world (analog)

Figure 2.1: some applications of ADCs. (figure from [68])

signals into digital codes used by electronic devices in order to do their specific tasks. These devices represents the gate between the real (analog) world and the computing machines (digital) one. Furthermore these devices allows electronic systems to communicate both to each others and to the real physical world. Considering that both the real and the digital world allows an almost infinite precision at an almost infinite speed, limitations of today electronic devices are given by ADCs.

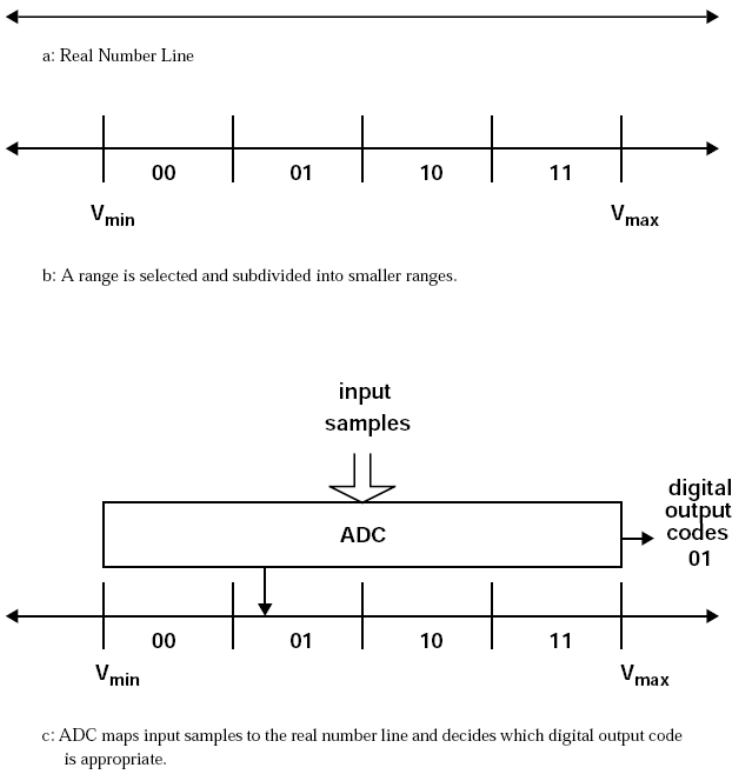


Figure 2.2: function of an ADC. (figure from [69])

Conceptually, an ADC works as shown in 2.2. Analog signals have a continuous range of values as do numbers on the real number line. An ADC takes a range of the real number line and divides it into smaller subranges. The size of each of the subranges is often referred to as the step size. These steps are usually uniform in size, but not always. A companding ADC for codecs is one example of an ADC having nonuniform step sizes. In this case the step sizes follow a logarithmic scale. To each subrange or step a code is assigned. Then, during the conversion process input samples are taken and mapped onto this real number line. The ADC then decides which subrange corresponds to the sample and sends the appropriate digital

code to the output. The key performance parameters of this conversion process are resolution and speed. The higher the two parameters are, the more the ADC allows the digital system to communicate in a better way with the real world.

Resolution

Resolution describes the fineness of the quantization performed by the ADC. A high resolution ADC divides the input range into a larger number of subranges than a low resolution converter. Resolution is usually defined as the base 2 logarithm of the number of subranges the ADC input range is divided into. This quantity is referred to as the number of bits resolved by the ADC. Another way to quantify resolution is the LSB or least Significant Bit and it is equal to the full input range of the ADC divided by the number of steps. Thus, for a fixed full scale input range a high resolution ADC can resolve smaller signals than a low resolution ADC is able to resolve.

Speed or Sampling rate

The sampling rate indicates the number times the input signal is sampled per second and it is expressed as Sample per second (S/s).Because of the Shannon theorem [1], the higher this value, the higher the device input bandwidth is allowing the system to increase its transmission capabilities. Considering the complexity of ADCs, there are several other terms used to characterize ADC performance.

Nonlinearity

Generally, the transfer characteristic of an ideal ADC progresses from low to high in a series of uniform steps. As the resolution increases, the input-output characteristic of the ADC better approximates a straight line. Due to this step nature, nonlinearities are present even in an ideal ADC. The transfer characteristic of a practical ADC contains steps which are not perfectly uniform, and this deviation generally contributes to further nonlinearity. Two measures of nonlinearity are used to characterize this deviation. Differential nonlinearity (DNL) measures how far each of the step sizes deviates from the nominal value of the step size. Integral nonlinearity (INL) is the difference between the actual transfer characteristic and the straight line characteristic which the ADC is intended to approximate. DNL and INL, illustrated in fig. 2.3 are generally expressed in terms of least significant

bits (LSBs) of converter input, or the smallest step size.

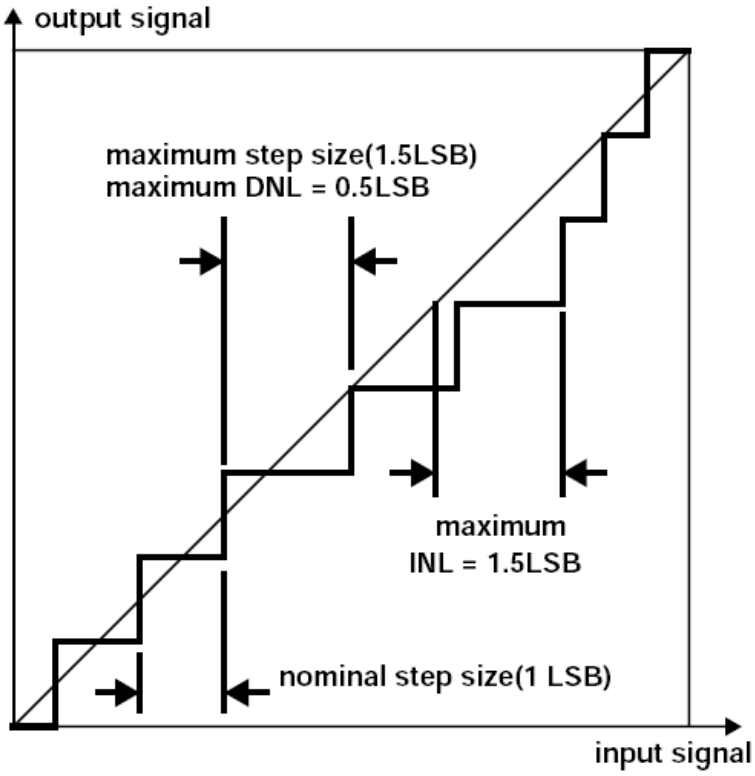


Figure 2.3: example of a transfer characteristics of a 3-bit ADC. (figure from [69])

SNR(signal to noise ratio) and SINAD or SNDR (signal to noise and distortion ratio)

The signal to noise ratio (SNR) is the ratio of signal power to noise power in the output of the ADC. One way to measure SNR is to plot the spectrum of the output of the ADC. The SNR is calculated by measuring the difference between the signal peak and the noise floor and including a factor to adjust for the number of samples used to generate the spectrum as shown below.

$$SNR(dB) = signalpeak(dB) - noisefloor(dB) - 10logN \tag{2.1}$$

The last term in the above equation may be understood as follows. To generate an N point fast Fourier transform (FFT) of a signal, N samples of the signal are taken. Sampling the signal N times increases the signal energy by a factor of and the noise energy by a factor of N. Thus the ratio of signal power to noise power is increased by a factor of N and the signal to noise ratio of the FFT is higher than the signal

to noise ratio in one sample of the signal. The signal to noise ratio improvement in dB is 10logN. Thus, the noise floor in the FFT becomes lower relative to the signal as more samples are taken. This idea is illustrated in fig. 2.4.

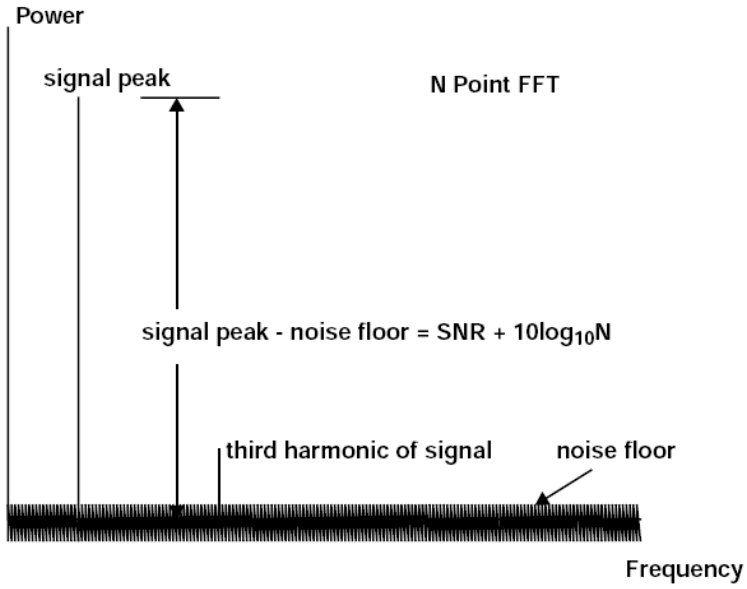


Figure 2.4: Procedure for computing SNR from a N-point FFT. (figure from [2])

The SNR is an alternative way to define the resolution of the conversion system. It is shown in [2] that

$$SNR(dB) = 6.02 \cdot Nbit + 1.76 \tag{2.2}$$

where Nbit is the resolution in bits.

The signal to noise plus distortion ratio measures the degradation due to the combined effect of noise, quantization errors, and harmonic distortion. The SINAD or SNDR of a system is usually measured for a sinusoidal input and is a function of the frequency and amplitude of the input signal. When a sinusoidal signal of a single frequency is applied to a system, the output of the system generally contains a signal component at the input frequency. Due to distortion, the output also contains signal components at harmonics of the input frequency. An ADC usually samples an input signal at some finite rate. As a result, some of the harmonic distortion products are aliased down to lower frequencies. Furthermore, the ADC adds noise to the output, and this noise is generally present, to some degree, at all frequencies. The SNDR of the ADC is defined as the ratio of the signal power in the fundamental

to the sum of the power in all of the harmonics, all of the aliased harmonics, and all of the noise. In a similar way to SNR, this is also an alternative way to define the resolution of the conversion system. It is shown in [2] that

$$SINAD(dB) = 6.02 \cdot ENOB + 1.76 \tag{2.3}$$

where $ENOB$ is the effective number of bits of resolution of the system. The notation 'effective' comes from the fact that this is a more realistic measure of the resolution of a system cause all possible signal degradation factors are considered.

DR(dynamic range) and SFDR(spurious free dynamic range)

Dynamic range is a measure of the range of input signal amplitudes for which useful output can be obtained from a system. Dynamic range can be defined in a number of different ways. One way to define dynamic range for a system is as follows. Apply a sinusoidal input of a single frequency to the system and vary the amplitude. Measure the maximum power obtainable from the system at the input frequency. The dynamic range can be than defined as the ratio of the maximum power at the fundamental frequency to the output power for a minimum detectable input signal. The minimum detectable input signal power is the value of the signal power when the signal to noise ratio is 0 dB. If the noise power is independent of the size of the signal, the dynamic range is equal to the SNR at full scale. However, in some cases the noise power increases as the signal level increases. In these cases, the maximum SNR is less than the dynamic range. The spurious free dynamic range is the ratio of the input signal level for maximum SNDR to the input signal level for 0dB SNDR. This measure of dynamic range is useful because it indicates the amount of dynamic range that can be obtained before distortion becomes dominant over noise. Fig. 2.5 indicates how to determine spurious free dynamic range from a plot of SNDR versus input level.

Intermodulation distortion

Intermodulation distortion (IMD) is a common measure of the linearity for amplifiers, gain blocks, mixers, and so, also for ADCs.

Two tone IMD is measured by applying two spectrally pure sine waves to the ADC at frequencies f1 and f2, usually relatively close together. The amplitude of each tone is set slightly more than 6 dB below full-scale so that the ADC does not clip

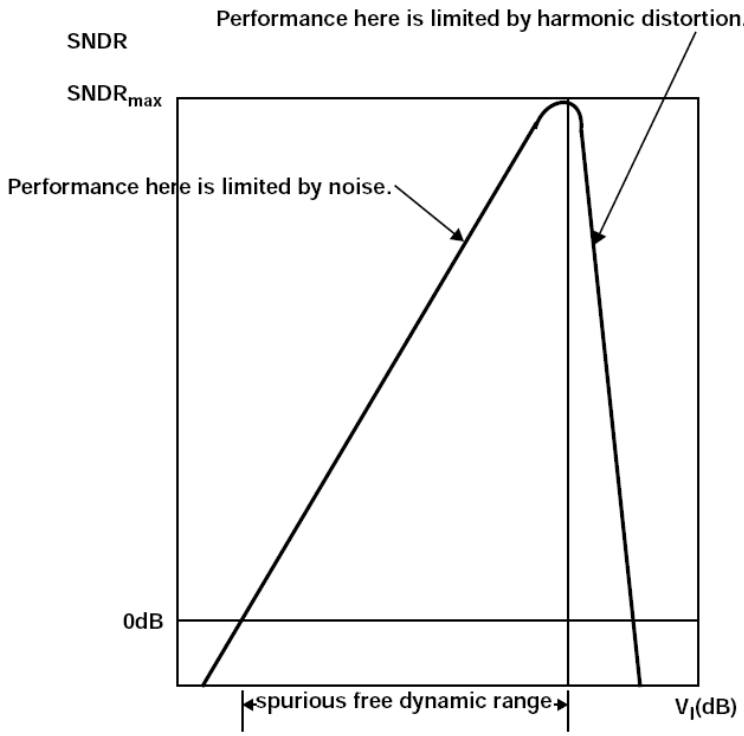


Figure 2.5: procedure for computing SFDR from a SNDR graph. (figure from [69])

when the two tones add in-phase. The location of the second and third-order products are shown in figure 2.6. It's important to notice that the second-order products

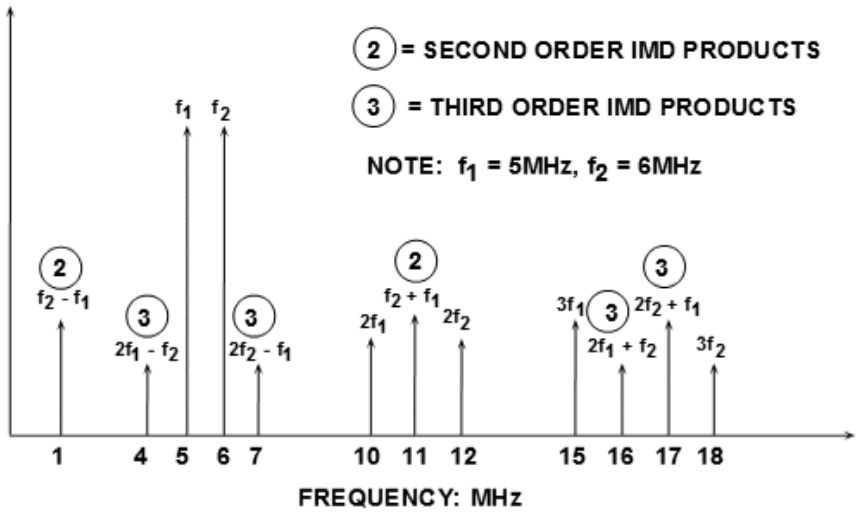


Figure 2.6: Intermodulation distortion. (figure from [2])

fall at frequencies which can be removed by digital filters. However, the third-order products 2f2 f1 and 2f1 f2 are close to the original signals and are almost impossible

to filter. The two-tone IMD usually refers to these "close-in" third-order products. Their value is specified in dBc relative to the value of either of the two original tones. The more the ADC is nonlinear and the more these intermodulation products are relevant.

This measure of non-linearity is very useful in multi-channel communications systems where the channel separation is constant across the frequency band. Third-order IMD products in fact can mask out small signals in the presence of larger ones.

Input signal swing

Input signal swing indicates the allowable range of values for the input. The input signal swing indicates the maximum and minimum values that the input signal may have without driving the ADC out of range or resulting in an unacceptable level of distortion.

2.2 Current state of the art of ADCs

Figure 2.7 shows the functional block diagram of a generalized conversion process. The overall conversion process is here divided into two part: the sampling section

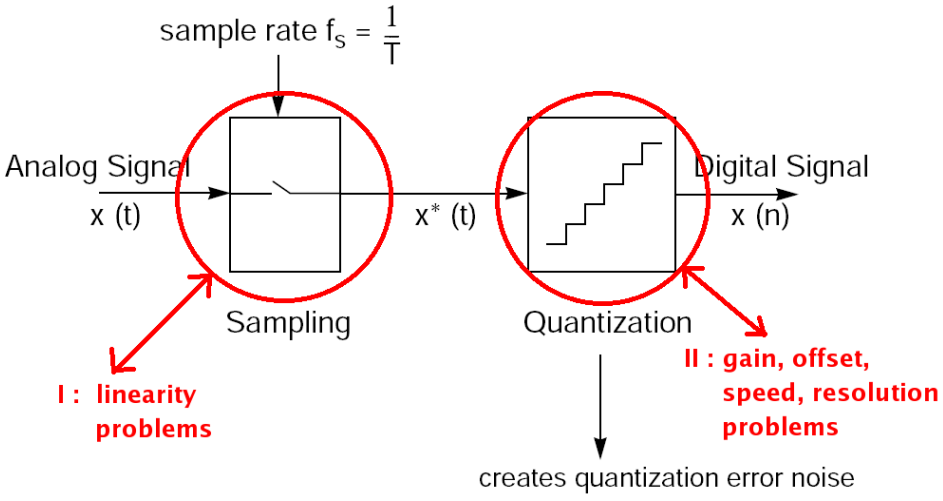


Figure 2.7: generalized A/D conversion process.

and the quantization section. The first one is important and necessary to keep the data fixed and stable during the conversion process. The necessity of such a circuit is easy to see if one considers what would happen if it was not present.

If the input value was permitted to change during this comparison process, the resulting conversion would be inaccurate, and possibly completely unrelated to the true input value. This circuit is used to convert a time-continuous signal into a discrete-time signal at a pre-defined sampling rate. Thus, the voltage comparison and, consequently, the data conversion can be performed on the sampling cycle. The precision of the whole conversion process depends directly on the sample-and-hold precision. The second section 'quantization' just converts the sampled analog input signal into a digital one. Critical limitations due to this section are the linearity of the conversion process that may induce distortion on the sampled signal. Many linearity problems in this section are architecture dependent and solutions normally require architectural modifications. Other errors like gain and offset can be corrected through either analog or digital calibration techniques. These kinds of problems are particularly critical in parallel architectures. Both sections are going to be analyzed in next section and a proposed solution in relation to the sampling section is presented in Appendix C. Problems related to the quantization section are going to be described as well and a new solution is presented in chapter 4.

2.2.1 Sampling section

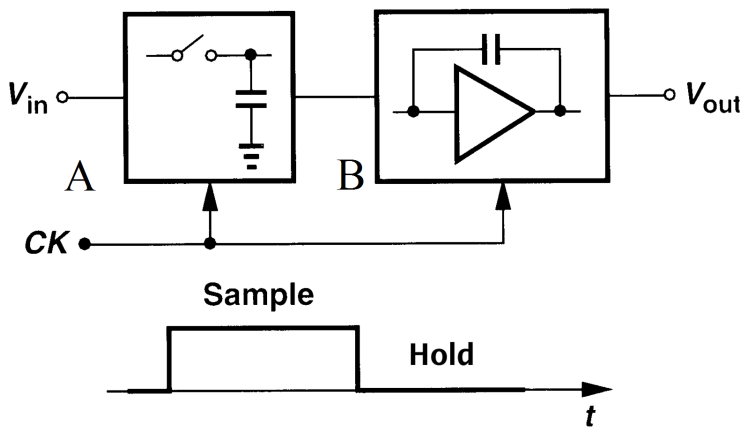


Figure 2.8: functional block diagram of a S/H circuit (figure from [75]).

The operation cycle of the sample-and-hold circuit (shown in figure 2.8) is divided into two distinct phases; these are clearly indicated by their names: the sampling phase and the holding phase. In the sampling phase, the analog signal is sampled,

where the sample-and-hold circuit works as a buffer. In the holding phase, the sampled signal is kept fixed until the next sampling phase, where the sample-and-hold circuit allows the voltage comparison of the signal. At this point, the time-continuous signal has been converted to a discrete time one. The sample and the hold phase are performed by a circuit that can functionally be divided into two main sections. These 2 sections are marked with 'A' and 'B' in the preceding scheme. Section 'A' has the function to sample the input signal and to hold it and it is composed by a switch and a capacitor; the switch samples the input signal while the capacitor memorizes the sampled value of it. Section 'B' has the function to buffer the sampled signal and make it available without any degradation of it caused by non-idealities of following stages. Usually the main element of this block is an operational amplifier that employs feedback to buffer the input signal present on the capacitor.

Analysis of the ideal case

All these building blocks in real implementations have many idealities that will be discussed later. Furthermore, even if all these blocks were ideal, the conversion process would still be affected by errors. This problem is due to the fact that the thermal noise present in the input signal generates fluctuations with zero mean around the correct value and when the switch samples the input signal, it doesn't only samples the wanted signal but the sum of it with the thermal noise. To analyze quantitatively the effects of this noise on sampled signal, the following schematic 2.9 is reported [5]. where R_s is the resistance of the sampling switch, V_{in} the input

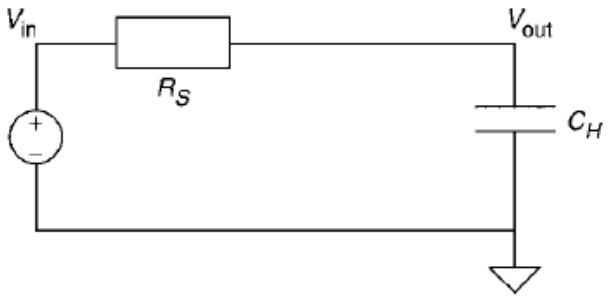


Figure 2.9: S/H circuit in sampling mode. (figure from [5])

signal and C_h the holding capacitor. The total noise power on the capacitor after sampling is given by equation 2.4.

$$P_n = \frac{k_B T(1 - e^{-\frac{2\delta t}{\tau}})}{C_h} \tag{2.4}$$

where $\tau = R_s C_h$ and δt=sampling duration.According to the preceding equation, the accurate resulting thermal noise on the capacitor is as shown in figure 2.10. This

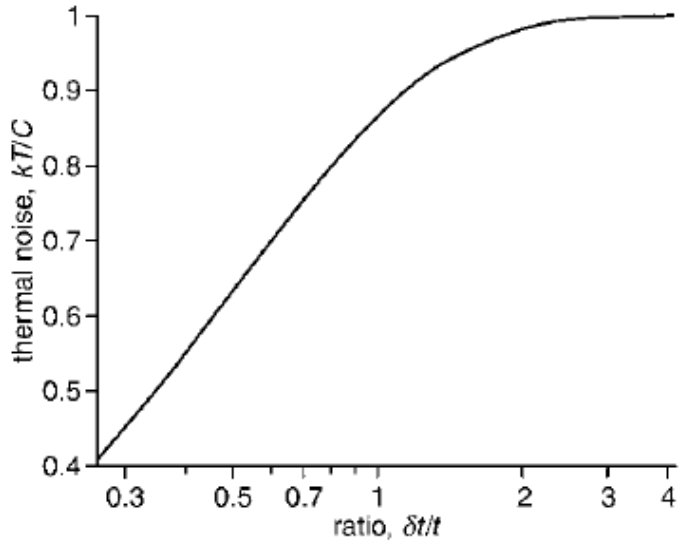


Figure 2.10: output thermal noise against $\delta t/\tau$ (figure from [5]).

shows that the noise for $\delta t/\tau \gg 1$ is approximately equal to KT/C_h, so it is present also for switch resistance equal to 0 (ideal switch). To prevent this problem, the only thing we can do is to proper size the value of Ch to make the power of this error compatible with resolution requirements of the overall converter. Another way to reduce the noise power is to make $\delta t \ll \tau$, but this solution is not a good one since we introduce a distortion in the sampled signal because the switch time constant τ is larger than the sampling duration one.

Analysis of section A: Sampling Process

MOSFET Transistors as switches

In CMOS technology devices like NMOS or PMOS are naturally acting as switches when the control voltage between the gate and the source is higher than Vt. The

more this voltage is high and the better performance can be achieved. Figure 2.11 shows the implementation of a switch with and NMOS device. The main problems

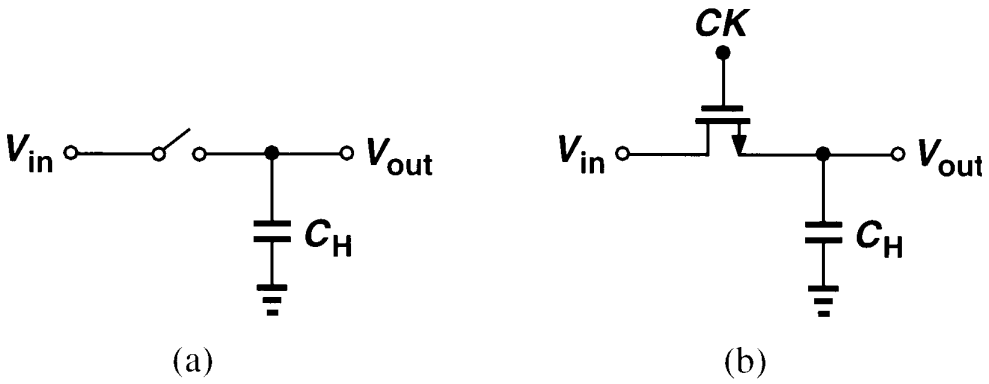


Figure 2.11: (a)simple sampling switch; (b)implementation of the switch by a MOS device. (figure from [75])

of CMOS devices utilized as a switch is that they are good one when the control voltage ck is much more than the drain-source voltage. The main problems are related to the fact that: if $V_{ds} > V_{gs} - Vt$ the MOS is in saturation, while if it is not, the MOS is in the linear region. The problem with this is that there are two different equation governing the current flowing through the device with a tremendous impact on the linearity of it. Another more tragic problem is when we want to charge the capacitor to the same voltage V_{dd} as the control Ck has. The device when the voltage on the capacitor reaches the value of $V_{dd} - V_t$ turns off and V_{out} is not able to grow any more. PMOS devices have the same problems in order not to charge, but to discharge a capacitor. Figure 2.12 plots the on-resistance of these devices acting as switches. From the preceding figure can be noted the strong variation of the input resistance in these devices. If the system had no speed requirements, this would not be a problem, but since the signal has got a bandwidth that extends from the dc value to a frequency f_{max}, changing the resistance value of the switch will change the transfer function of the sampling system in a way depending on the value of the input signal. If this problem is not solved, it will cause a distortion on the sampled signal. Figure 2.13 shows the sampling system and its transfer function. The best technique to avoid distortion is to keep R_{sw} constant. For this reason, the value of

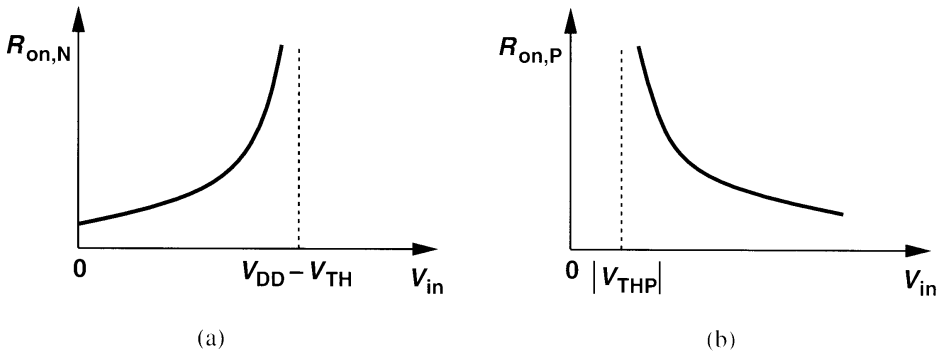


Figure 2.12: on resistance of (a)NMOS and (b)PMOS devices as a function of the input voltage. (figure from [75])

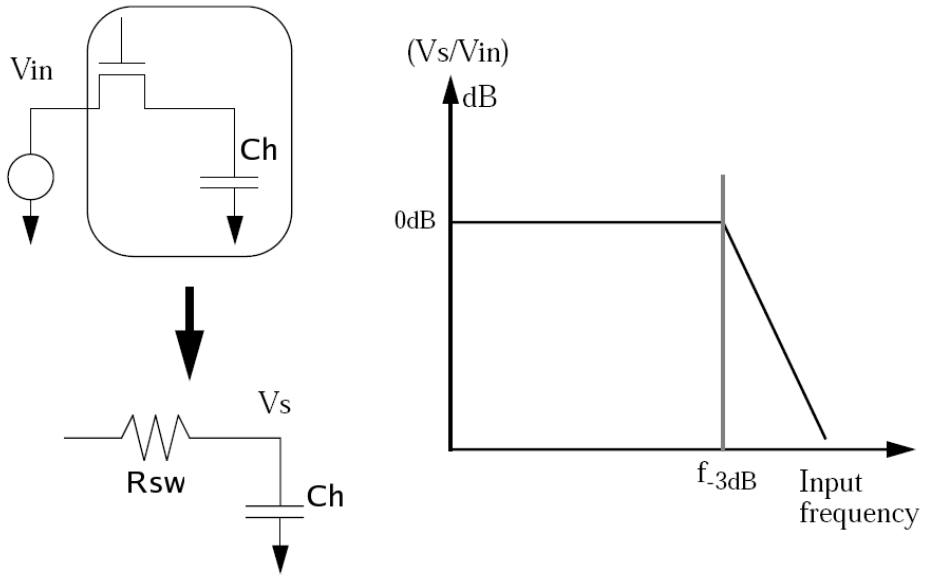


Figure 2.13: input bandwidth of a simple MOS S/H circuit. (figure from [74])

C is normally determined by thermal noise requirements. In order to enhance the signal bandwidth, the switch resistance is usually kept as low as possible. Another mechanism that affects sampling linearity is charge injection error. When a MOS transistor is used as a switch as shown in fig. 2.14, there is a finite amount of charge in the conducting channel, whose magnitude is approximately $C_{ox}(V_{gs} - V_{th})$. When input signal is sampled on a capacitor by turning off the transistor, this charge is pushed out from the channel to either direction, and part of it is dumped on C_S causing an error voltage of $\Delta Q/C_S$. The magnitude of ΔQ is a complex function

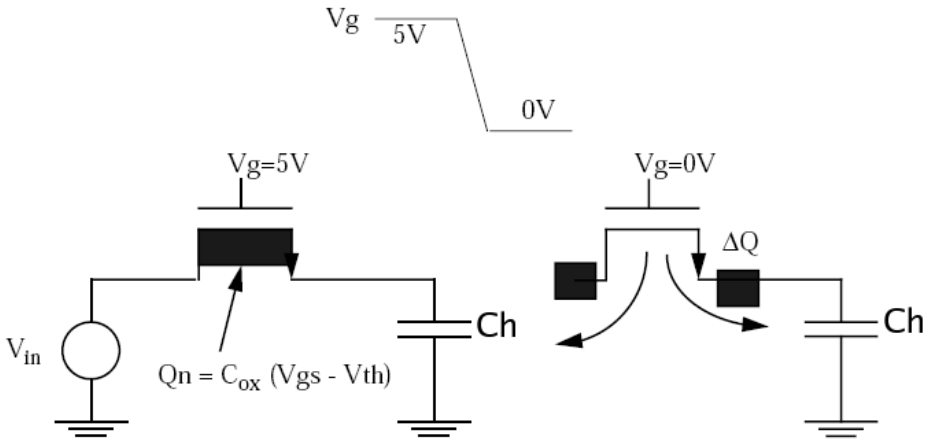


Figure 2.14: charge in the conducting channel of a MOS device. (figure from [74])

of the falling time of the sampling clock edge and impedance level at drain/source, and in the fast switching-off conditions, the transistor conduction disappears quickly and ΔQ approaches 50% of $C_{ox}(V_{gs} - V_{th})$ [6]. This charge injection error can produce a large error in the S/H circuit because the magnitude of the error voltage is signal dependent (ΔQ is proportional to $C_{ox}(V_g - V_{in} - V_{th})$). If the input signal is close to V_g, then less charge is in the conducting channel and less error voltage is observed. However, when V_{in} is much less than V_g, the amount of charge in the channel increases and so the error voltage. How does the charge injection affect the precision? Assuming all of the charge (worst case) is deposited on the capacitor, we express the sampled output voltage as:

$$V_{out} = V_{in}(1 + \frac{WLC_{ox}}{C_h}) - \frac{WLC_{ox}}{C_h}(Vdd - Vth) \tag{2.5}$$

suggesting that the output deviates from the ideal value through two effects: a non-unity gain equal to $1 + WLC_{ox}/C_h$ and a constant offset voltage $-WLC_{ox}(V_{dd} - Vt)/C_h$. In other words, since we have assumed channel charge is a linear function of the input voltage, the circuit exhibits only gain error and dc offset. In the this discussion, it was assumed that Vth is constant. However, for NMOS switches (in an n-well technology), body effect must be taken into account. Since

$$Vth = Vth_0 + \gamma(\sqrt{2\phi_b + V_b s} - 2\phi_b) \tag{2.6}$$

if $V_{bs} = -V_{in}$, we have:

$$V_{out} = V_{in}(1 + \frac{WLC_{ox}}{C_h}) - \frac{WLC_{ox}}{C_h}\gamma\sqrt{2\phi_b + V_b s} - \frac{WLC_{ox}}{C_h}(Vdd - Vth_0 - \gamma\sqrt{2\phi_b}) \tag{2.7}$$

It follows that the nonlinear dependence of Vth upon V_{in} introduces nonlinearity in the input/output characteristic. In summary, charge injection contributes three types of errors in MOS sampling circuits: gain error, dc offsets and nonlinearity. In many applications, the first two can be tolerated or corrected whereas the last cannot. It is instructive to consider the speed-precision trade-off resulting from charge injection. Representing the speed by a simple time constant τ and the precision by the error δV due to charge injection, we define a figure of merit as $F = (\tau \cdot \delta V)^{-1}$, where

$$\tau = R_{on}C_h = \frac{LC_h}{\mu_n C_{ox} W(Vdd - Vin - Vth)} \tag{2.8}$$

$$\delta V = \frac{WLC_{ox}}{C_h}(Vdd - Vin - Vth) \tag{2.9}$$

and

$$F = \frac{\mu_n}{L^2} \tag{2.10}$$

Thus, to first order, the trade-off is independent of the switch width and the sampling capacitor but depends only by technological parameters. Even if many architectural techniques discussed later consent to overcome this technological bound, it is still true that the more one technology is advanced, the better performance can be achieved. In addition to the channel charge injection, a MOS switch couples the clock transitions to the sampling capacitor through its gate-drain or gate-source overlap capacitance. Depicted in fig. 2.15, the effect introduces an error in the sampled output voltage. Assuming the overlap capacitance is constant, we express the error as:

$$\delta V = C_{ck}\frac{WC_{ov}}{WC_{ov} + C_h} \tag{2.11}$$

where C_{ov} is the overlap capacitance per unit width. The error δV is independent of the input level, manifesting itself as a constant offset in the input/output characteristics. Despite of charge injection and clock feed-through this doesn't lead to a trade-off between speed and precision since this is a constant offset that can be

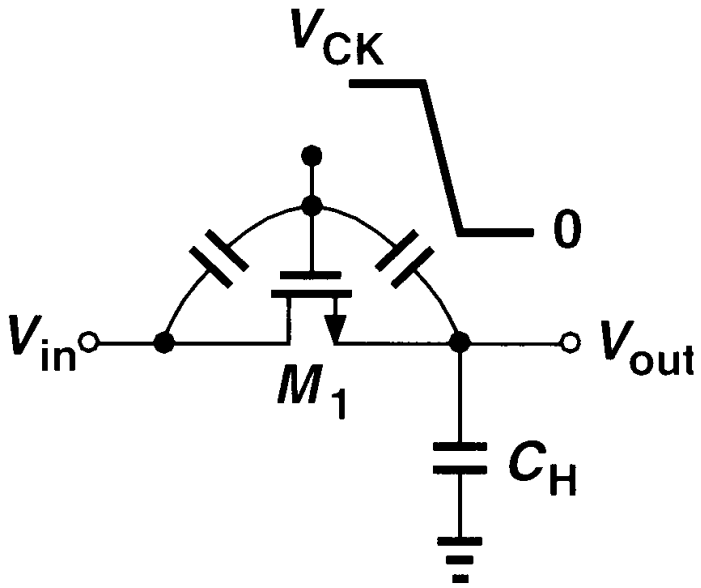


Figure 2.15: clock feed-through in a sampling circuit. (figure from [75])

avoided simply with a differential architecture.

Solutions to encountered problems

The following architecture [7] is the most used solution to previous problems. The main idea is that, if we keep constant the voltage between the gate and the source of a MOS device, it is possible to eliminate all sources of distortion or non-idealities discussed before. This way the on-resistance is perfectly constant and the charge injected is independent on the input signal leaving no distortion at all on the sampled signal.Figure 2.16 shows the conceptual output waveforms of the bootstrap circuit. As the variations in on-resistance due to the body effect, are not eliminated, the following architecture presented in [8] has been proposed. An improvement of these techniques is called bottom-plate sampling [7]. This serves to partially eliminate some of the residual errors in all the preceding techniques; fig. 2.17 shows the functional diagram related to this system.

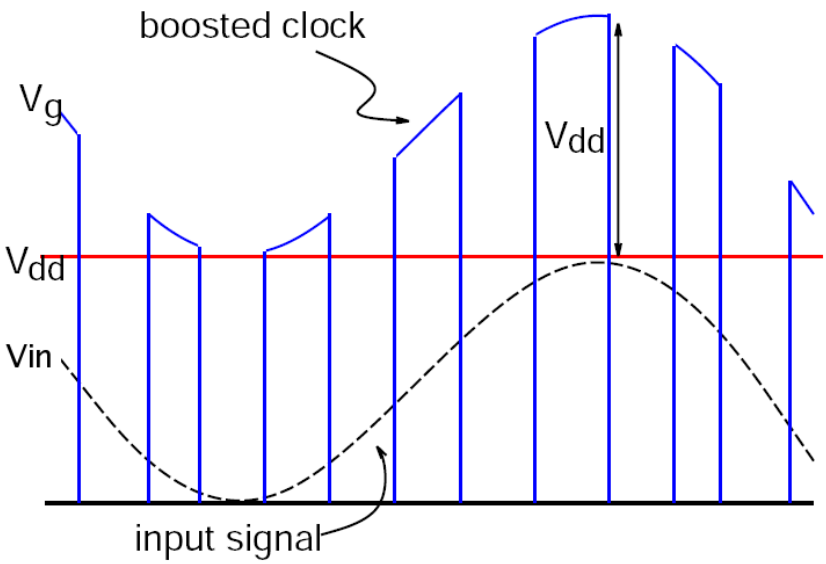


Figure 2.16: conceptual bootstrap circuit output in order to modulate Vg. (figure from [7])

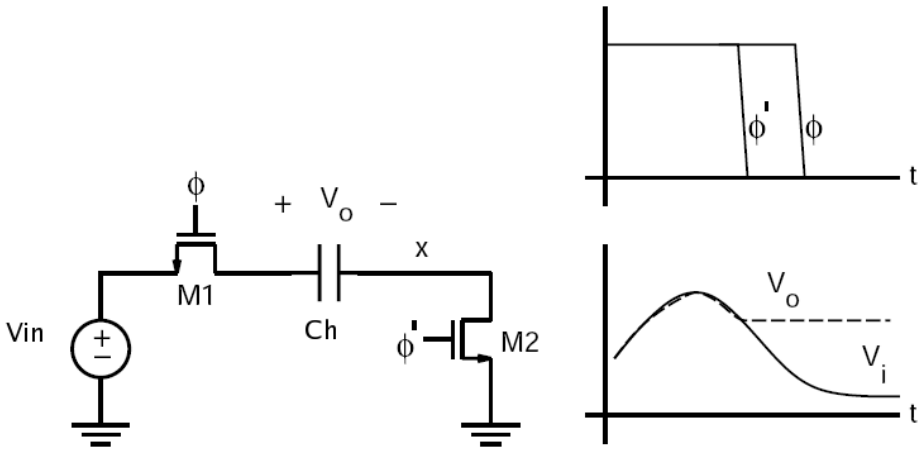


Figure 2.17: bottom plate technique. (figure from [7])

Analysis of section B: Holding Stage

The main function of this section is to buffer the sampled signal allowing later stages to receive a copy of it without affecting sampling resolution or speed. The best way to buffer a signal with a reduced introduction of errors is to create a copy of it using an operational amplifier closed in a feedback loop. This way lots of errors introduced by circuit non-idealities are divided by the op-amp gain. Offset errors are

still present but they are fixed errors that can be compensated during the calibration phase of the converter. Lots of architectures have been proposed during past years. Common features present in all are:

- The use of a differential architecture in order to avoid even-order harmonic distortion in the output spectrum of the sampled signal.

- The use of a negative feedback in order to reduce errors introduced by non-idealities in the closed loop system.

- Changes in feedback configuration between the sampling and the holding phase are performed basing on consideration made in section 'A' in order to minimize charge injection and other effects that may allow distortion in the buffered signal.

- Loop connections are made to obtain a feedback factor as close to 1 as possible in order to reduce GBW requirement and so power consumed by the Op-Amp.

In the next paragraph proposed architectures will be analyzed and discussed. Many other types have been proposed but most of them are derivations of these fundamental ones.The following section is going to analyze main techniques in order to acquire the signal stored into the sampling capacitor. After that different types of buffering architectures will be discussed

Type of data acquisition: Direct Vs Indirect systems.

Direct systems acquire the data stored in the capacitor by simply disconnecting the capacitor from the input and connecting it in a feedback loop with an operational amplifier. This way the current needed to bias the following stages is supplied by the opamp and the charge contained into the hold capacitance is not modified during hold phase. Figure 2.18 gives an example of how this class of converters works. The precision of this architecture is related to the inverse of the opamp voltage gain, while the ideal feed-back factor of this architecture is 1. A possible implementation of this architecture is called the flip-around T&H and it is presented in [7]. However because of some parasitic capacitance M1 does inject charge onto CH. This coupling, as can be seen from fig. 2.19 undermines the overall system. However, even if this

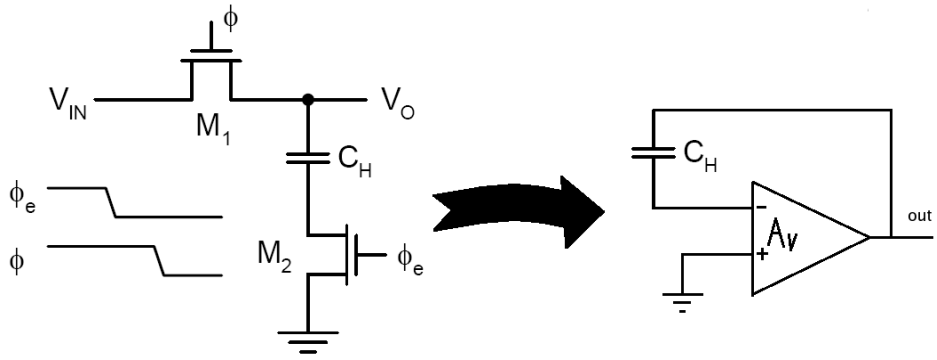


Figure 2.18: direct acquisition system. (figure from [76])

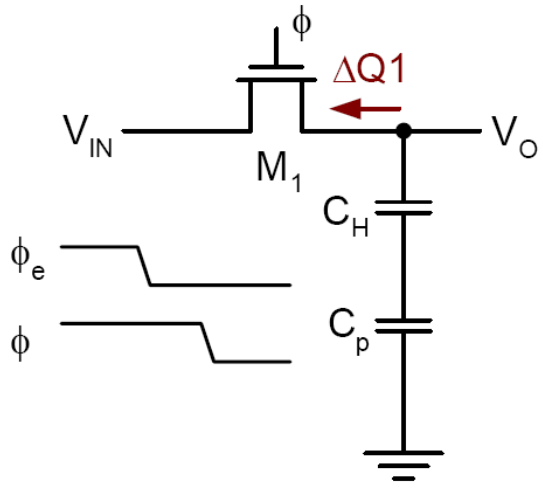


Figure 2.19: parasitic effect of system in fig.**??**. (figure from [78])

is true, an interesting observation should be noted: even if M1 injects some charge onto CH, the total charge at node X cannot change cause of Kirchhoff's 2nd law. It states that the sum of the currents entering any node (i.e., any junction of wires) equals the sum of the currents leaving that node. This concept is shown in figure 2.20. The main idea to improve this system resolution is to Process the total charge at node X instead of looking at voltage across CH like in the previous case. This is the base concept of Indirect data acquisition systems. A possible implementation of this architecture is called the charge-redistribution T&H and is described in [9]. To compare these architectures is quite difficult as on one hand the direct acquisition architecture has noise and power advantages compared to the indirect one, but, on the other hand it has less resolution performance than the other one due

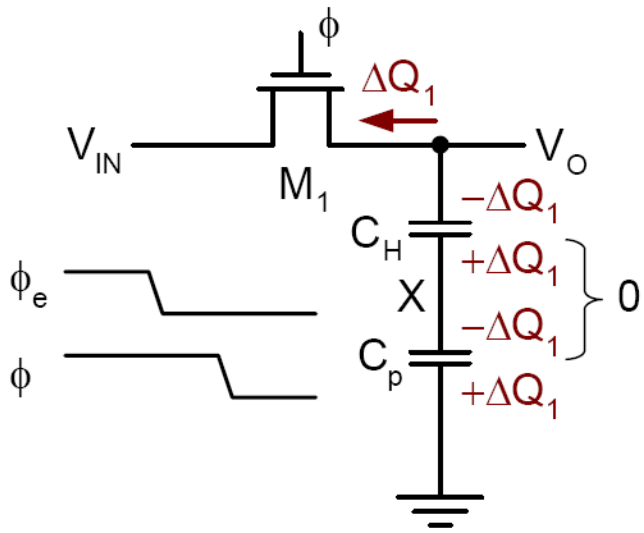


Figure 2.20: charge change at node X. (figure from [78])

to bottom-plate MOS parasitic coupling. Further analysis related to a particular circuital implementation of these techniques may define in a better way which one is more suitable to achieve our target, but this is beyond the purpose of this thesis.

2.2.2 Quantization section

This section is going to describe the system responsible for quantizing the sampled signal. Basing on previously described considerations, many ADC architecture and integrated circuit technologies have been proposed and implemented to date. Each architecture has advantages and disadvantages, and each has a set of applications for which is the best solution. Fig. 2.21 gives a visual representation of the state of the art of these devices, grouping them basing on their functional architecture. The next subsections will present in detail the advantages, disadvantages and improvements of these architectures.

Flash ADC

The flash architecture is capable of the fastest conversion rate and constitutes the basis of many of the other architectures. The functional block diagram and the operation of the converter are shown respectively in fig. 2.22 and fig. 2.23. The input

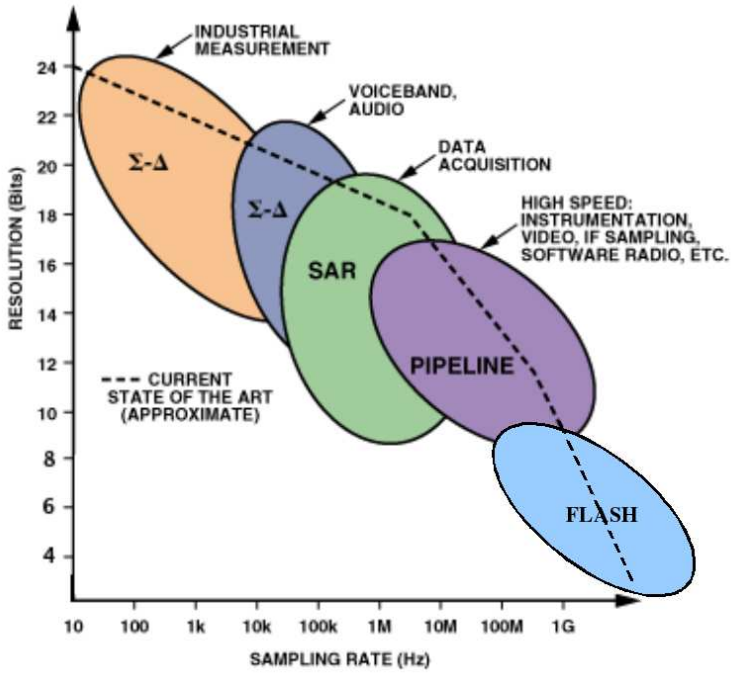


Figure 2.21: state of the Art of ADCs.

signal is sampled and compared using comparators. The output of the comparators is processed by an encoder to provide a binary output. The two primary drawbacks to the flash ADC are the large hardware requirement and so high power dissipation and sensitivity to comparator offset. The required comparator offset voltage for a flash ADC with N bit resolution is less then 2^{-N}. At high resolution, this required comparator offset becomes very small and requires a large number of comparators, therefore ADCs with resolution higher than 10 bits rarely use flash architecture. A further limitation of this architecture include differential phase errors from clock skew, very large circuit area and significant charge blowback.

Pipelined ADC

This architecture is composed by multiple-stage flash converters. This architecture, in fact is composed by many stages. Each stage is a little flash converter with few bits of resolution and they are all connected each others like in fig. 2.24, thus the name 'pipelined'. In these multiple stage devices, each stage includes an input sample-and-hold circuit that allows the various stages to work with different input values. In this

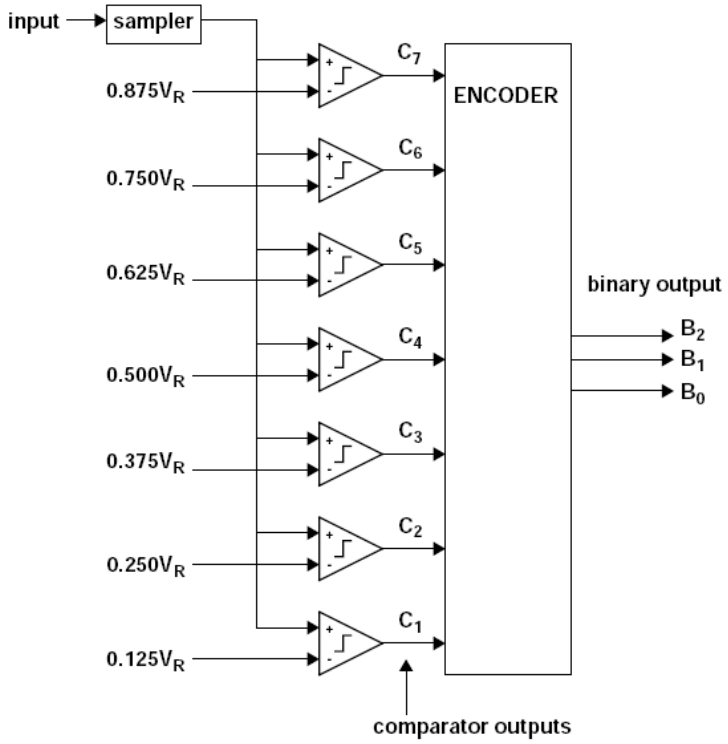


Figure 2.22: flash ADC architecture. (figure from [69])

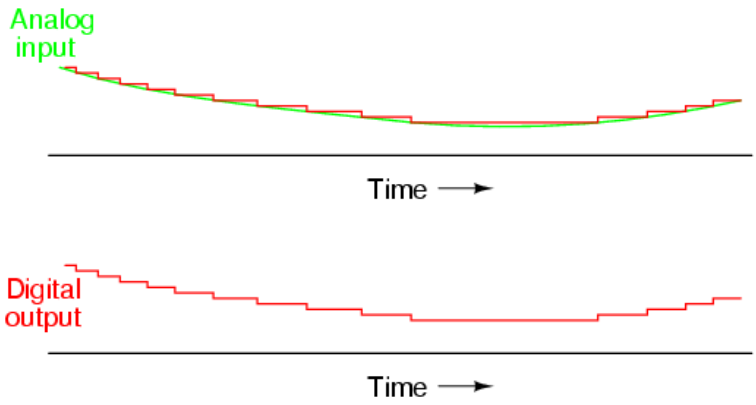


Figure 2.23: flash ADC operation. (figure from [71])

way it is possible to trade circuit complexity with resolution, separating throughput rate from conversion time. Fig. 2.25 shows the pipeline stage architecture. Although the throughput rate is independent of the number of stages in the pipeline, the conversion time for any given sample is proportional to the number of stages in the pipeline. This is true as the signal must work its way through all of the stages before the complete output word is generated. This delay can be an issue if the pipelined

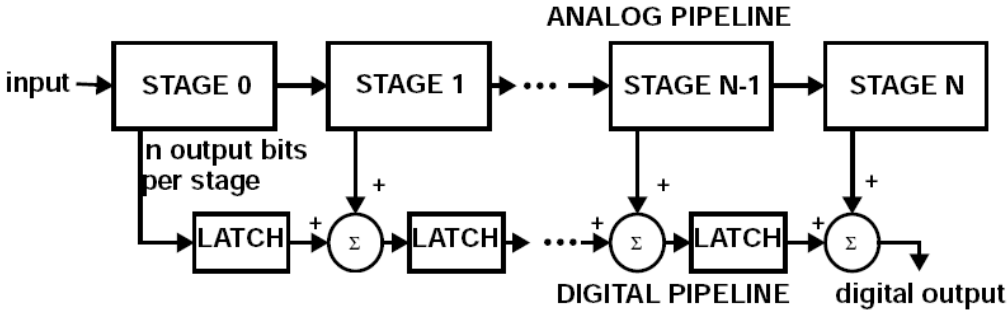


Figure 2.24: pipeline ADC architecture. (figure from [69])

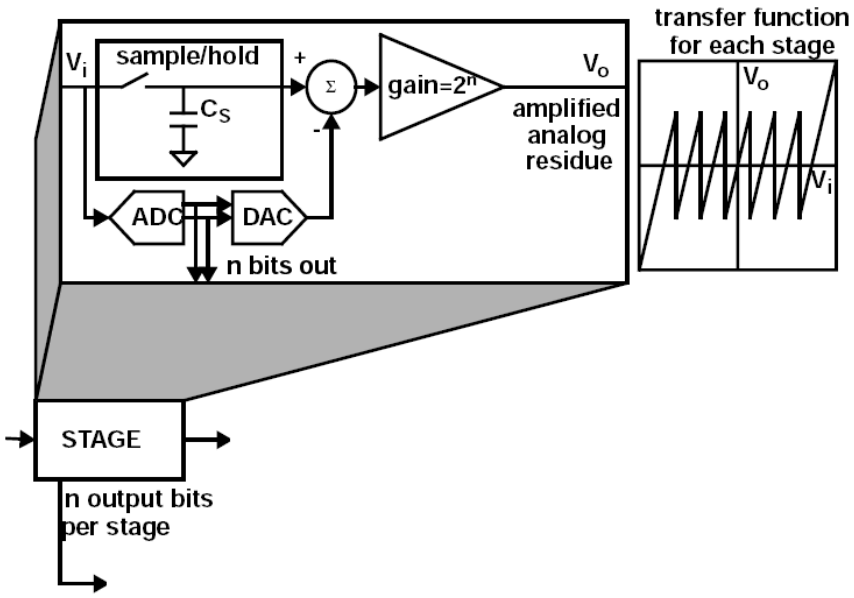


Figure 2.25: pipeline ADC stage architecture (figure from [69]).

ADC is part of a feedback system. Looking at figure 2.25, it can be noted that an amplifier is placed after the flash converter. This gain element is used to amplify the residue(conversion error) before passing it on to the next stage. By doing this, the resolution requirements for the following stages are relaxed. One significant implication of this is that the comparators in the later stages of the pipeline need not be accurate to the full ADC resolution. The disadvantage of adding the gain blocks is that they tend to be the dominant source of power dissipation in the ADC. Due to their tolerance to comparator offsets and the ability of the pipeline stages to operate in parallel, pipelined ADCs are well suited for high resolution applications where high speed is required. Despite of Flash converter architectures, pipelined

ones can achieve a resolution up to 16-bit at a considerable high frequency (around 100Ms/s). In addition to that, this architecture is also able to partially cover the frequency range of flash converters at a cost of a reduced resolution. Some pipelined ADCs (such as the MAX104) are in fact are able to achieve sampling rates of 1GHz. The resolution of pipelined flash converters is mainly limited by differential linearity errors caused when the first error signal is handed off to the second stage of the pipeline, and by clock jitter.

Successive approximation ADC architecture.

This kind of converter was invented to trade area occupation and speed with resolution. This architecture performs the same operation of a pipeline converter, just not in space(pipeline architecture) but in time. A successive approximation ADC, also known as a binary search ADC, is a special type of converter that uses a DAC to produce an analog signal that approximates the input sample. By adjusting the DAC until the DAC output matches the input sample, a digital code representing the analog input can be generated. Fig. 2.26 shows the way the DAC signal approximates the input signal and fig. 2.27 shows the operation of the converter. Successive approximation ADC consists of only one stage containing a sample and hold, a flash ADC, a DAC, and a logic that controls the DAC. An example of a successive approximation ADC is shown in the block diagram in fig. 2.28. In the example shown, the ADC consists of a single comparator. The operation of the successive approximation ADC is as follows. The control logic is initialized, and this initializes the output of the DAC. A sample of the input signal is taken by the sample and hold circuit, and the initial DAC output is subtracted from the input sample. The difference is quantized by the comparator which instructs the control logic to either increase or decrease the DAC output. The new DAC output is again subtracted from the input sample, and the process repeats until the desired accuracy is obtained. This single comparator successive approximation ADC resolves one bit per cycle. With a good feedback DAC, such converters are capable of more than 16-bit resolution, but at least one clock cycle is required for each bit of resolution. Furthermore, an input sample-and-hold circuit is required to preserve the input am-

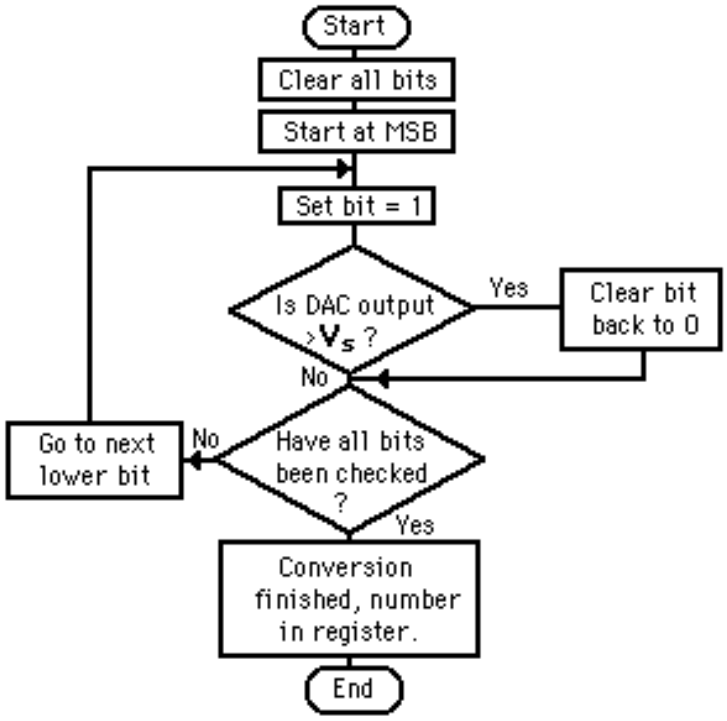


Figure 2.26: successive approximation ADC algorithm (figure from [71]).

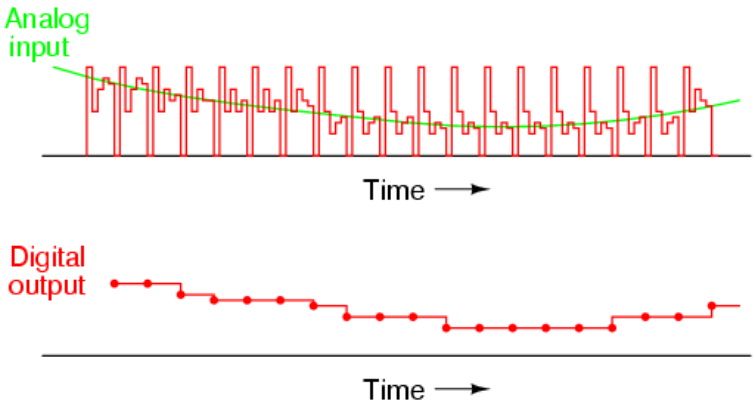


Figure 2.27: successive approximation ADC operation. (figure from [71])

plitude during the time required for the conversion. Droop in the sample-and-hold over the time required for the conversion leads to differential linearity inaccuracies in the converter unless redundant successive approximation steps near the end of the conversion are provided. Another drawback comes from the amplifier: in order to provide high accuracy in these converters, the gain of the amplifier has to be increased on progressive steps. The settling time of the amplifier on high-gain steps is longer and slows the converter.

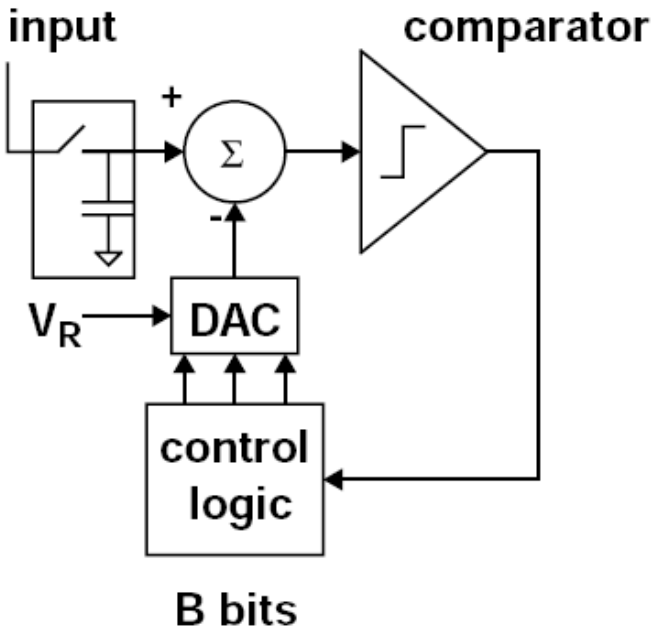


Figure 2.28: successive approximation ADC architecture (figure from [69]).

Sigma-Delta ADC

Figure 2.29 shows the functional block diagram of the converter. Sigma-delta con-

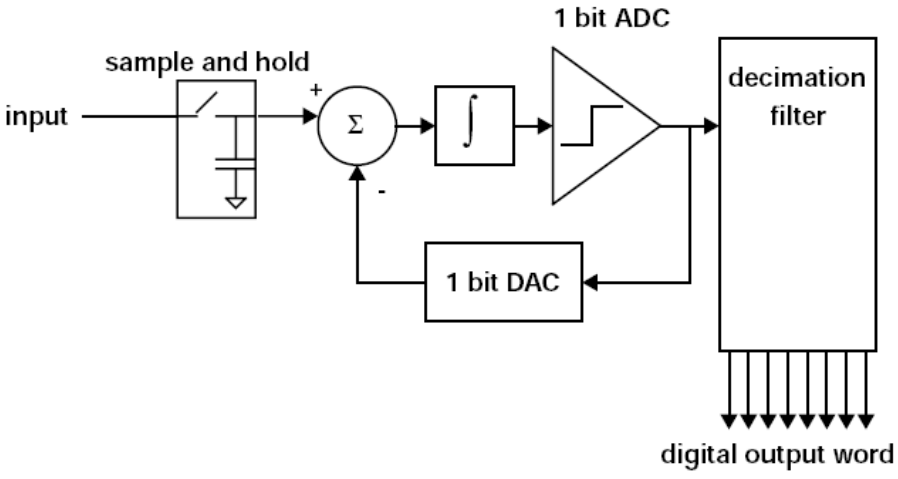


Figure 2.29: sigma-delta ADC architecture. (figure from [69])

verters belongs to the family of oversampling converters. This kind of architecture was invented to trade just sampling speed with resolution. As quantization noise

spreads over the overall allowable input bandwidth, oversampling the input signal just spreads the noise power over a wider band and so lowers the noise floor. If the resulting spectrum is cut with an appropriate low-pass filter, the in-band noise power is reduced improving the SNR. The principle just described is the one of oversampling converters. Sigma-delta converters are an improvement of the oversampling ones. The main difference is that they use information theory to modulate the input signal in order to push the most of quantization noise outside the signal bandwidth. Fig. 2.30 shows how the quantization noise is modulated by different ADCs architectures while fig. 2.31 shows how the input signal is modulated by the system before being filtered by the low-pass filter. Figure 2.30 compares per-

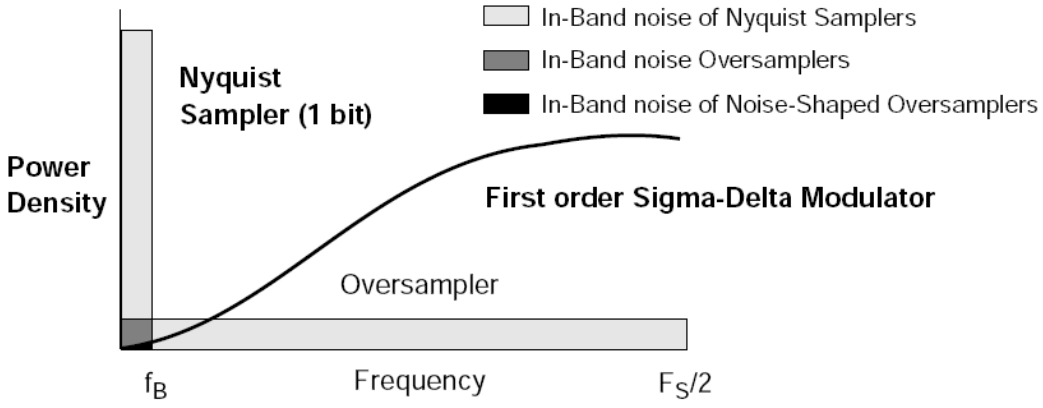


Figure 2.30: comparison of noise shaping performances of various ADCs. (figure from [72])

formances of the oversampling and Nyquist architecture with that of a sigma-delta converter. As can be inferred, sigma-delta modulation technique achieves the best performance. It can be shown [3] that, compared to Nyquist converters, oversampling technique approximately reduces the in-band rms noise by the square root of oversampling ratio (OSR). This ratio is the ratio between the sampling frequency to the Nyquist frequency. Therefore each doubling of the sampling frequency decreases the in-band noise by 3 dB, increasing the resolution by only half a bit. First order sigma-delta modulators exhibit a noise shaping characteristic that allow the in-band noise power to be reduced by 9 dB each doubling of the OSR providing 1.5 bit of extra resolution. The way to increase the resolution of the sigma-delta modulator

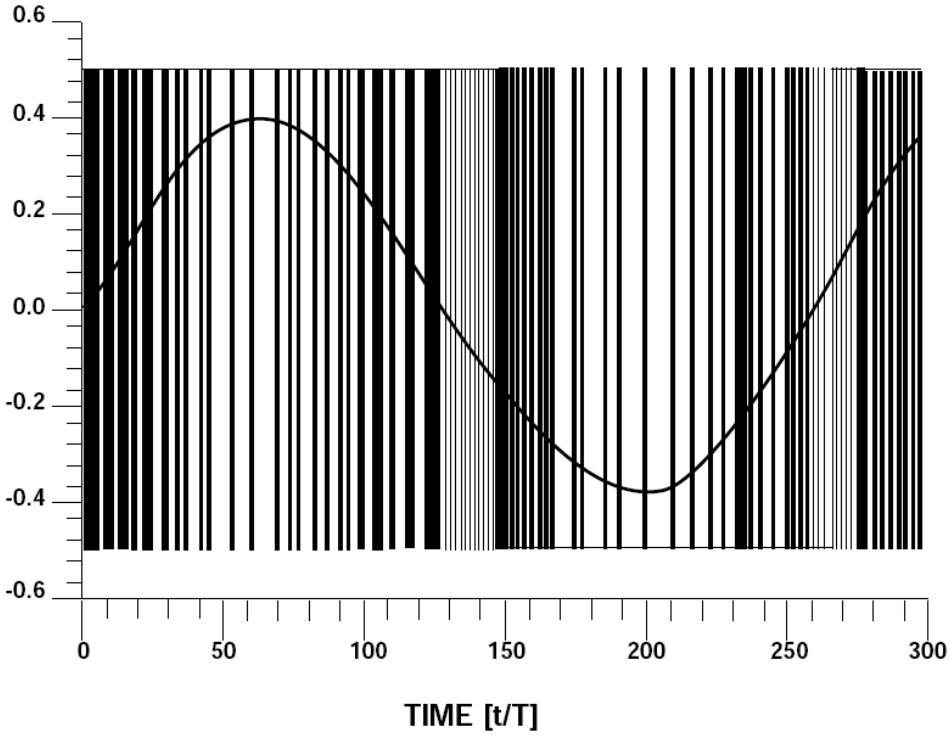


Figure 2.31: input and unfiltered output of a Sigma-Delta modulator. (figure from [72])

without oversampling too much the input signal, is to use a loop filter that performs more aggressive noise shaping functions. Figure 2.32 shows the performance of this kind of systems. Looking at the Figure 2.32, it can be inferred that the higher the

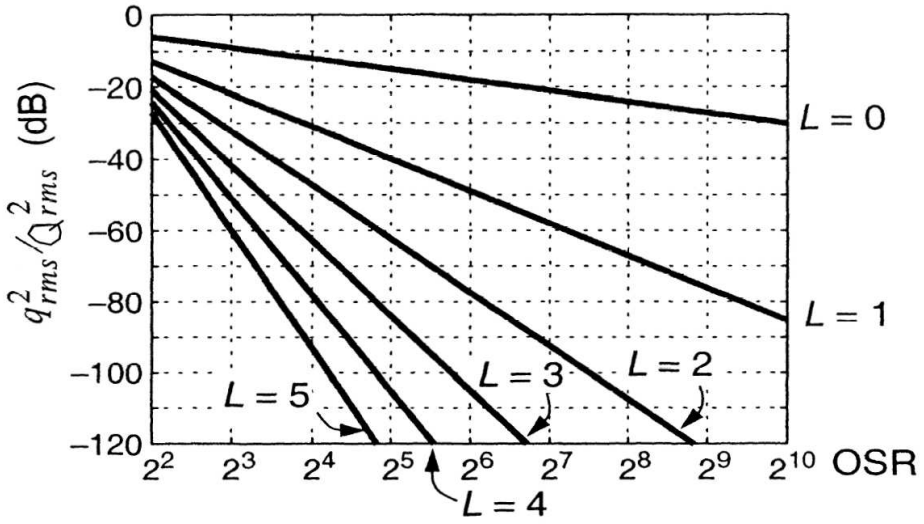


Figure 2.32: performance of higher order noise shaping loops. (figure from [73])

loop order is and the better performances can be achieved. Another advantage of this architecture is its robustness to circuit imperfections. All these non-idealities result in noise and are spread over the overall sampling bandwidth and thus are greatly reduced by the filter. In addition to that, if the coarse quantization is made using a 1-bit converter, linearity is guaranteed [3]. One drawback of the sigma-delta converter is that it requires a very high over-sampling ratio (ratio of sampling rate to Nyquist sampling rate) in order to achieve high resolution. Other drawbacks relate to the order of the converter loop. It is well known, for example, that the order of the signal processing feedback loop may be increased to increase the input signal bandwidth for the same converter resolution. The number of integrating amplifiers provided must be increased, however, in proportion to the order of the required loop filter. There is a well known issue of instability in sigma-delta converters as the order increases, though there are known stable high-order structures. Other well-known limitations of the sigma-delta converter include spurious tones and an output noise spectral density that increases with frequency at a rate that is proportional to the order of the converter loop. To increase the sigma-delta ADC sampling frequency for high-resolution telecommunications applications, designers have replaced the single comparator in the sigma-delta converter with a multi-bit flash converter at the output of the integrating amplifier chain in order to obtain more bits per conversion step. Presently the highest resolution ADCs use oversampling because of the fact that this architecture is able to trade sampling speed with resolution, while being robust to circuit imperfections.

2.3 Improving the state of the art

2.3.1 Main Limitations of "traditional" architectures

Looking at figure 2.21 it can be noted that all architectural topologies, other than Flash, trade sampling speed with resolution. Each ADC family in fact increases performance in terms of conversion accuracy but it looses in speed. For a better understanding of these limitations, figure 2.33 is provided. Aside from quantization noise, three mechanisms limit achieved SNR: input-referred circuit noise (equivalent

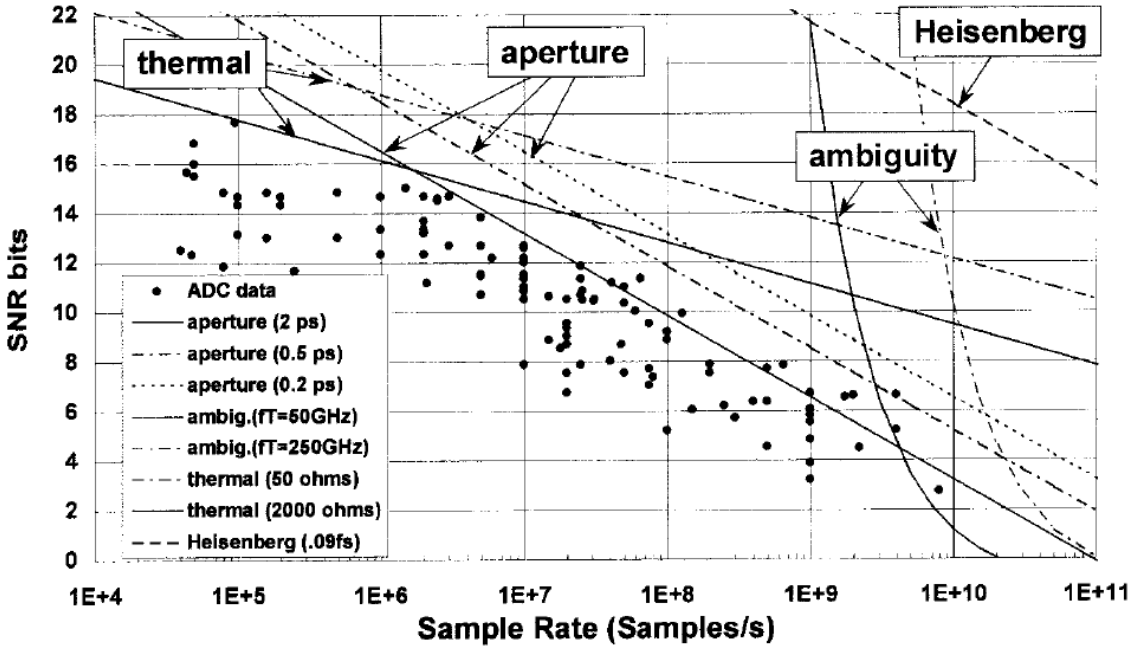


Figure 2.33: current state of the art of ADCs. (figure from [4])

thermal noise), aperture uncertainty, and comparator ambiguity. The first one (thermal noise, Johnson noise, or Nyquist noise) is the noise generated by the thermal agitation of the charge carriers (the electrons) inside an electrical conductor in equilibrium, which happens regardless of any applied voltage. The second one (aperture uncertainty) is the sample-to- sample variation in the sampling process. Aperture uncertainty has two residual effects: the first is an increase in system noise, the second is an uncertainty in the actual phase of the sampled signal itself. The last one (comparator ambiguity) is when the output of a comparator is not able to generate any logical value (0 or 1). This effect is due to the finite speed with which transistors in the comparator are able to respond to a small voltage difference. The equations that calculate the associated maximum achievable resolutions, in SNR-bits, are given in Appendix II of [4]. The ultimate limit to the ADC resolution-sampling rate product may be estimated using the Heisenberg uncertainty principle. The analysis is summarized in [4]. For these reasons, if we consider a fast ADC system, comparator ambiguity is the main limitation. Thermal noise and aperture jitter instead, are the main problems of high resolution systems. All ADCs can therefore be grouped into two main class where: to the first one (A) belongs all those converters suitable

for high frequency but low resolution, and to the other group (B) belongs all low frequency, high resolution converters. Figure 2.34 shows graphically the previous statement while the circle (T) represent our target.

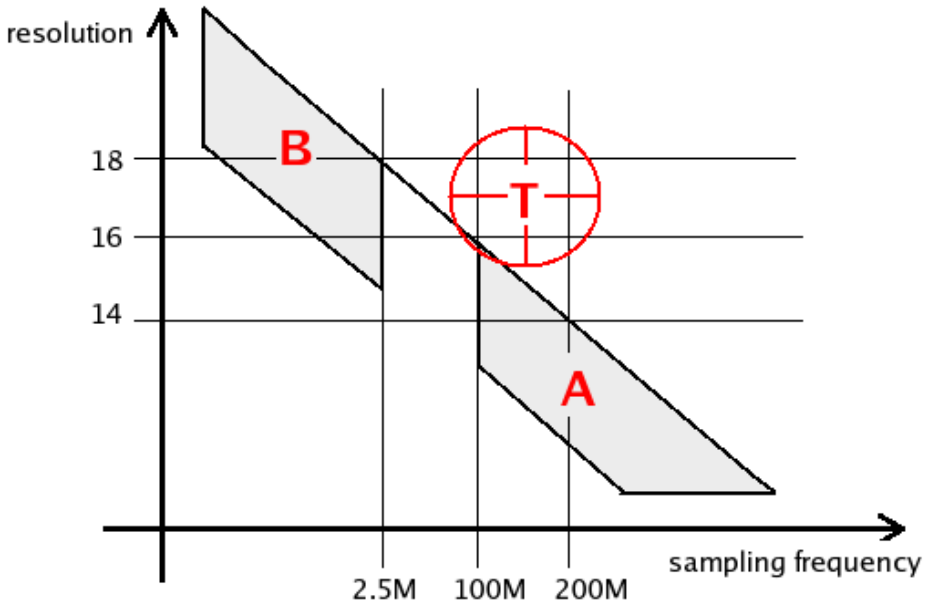


Figure 2.34: current state of the art of ADCs.

The target region (T) has been chosen to be approximately from 100 to 200 MSPS and from 16 to 18 bits of resolution. The reason of this choice is because, as it can be noted from figure 2.34, there is a gap between converters belonging to the groups A and B. The purpose of this work is in fact to investigate a way to find the missing bridge between these two types of classes of ADC architectures. In this thesis we try to do it by studying and improving the state of the art of parallel architectures.

2.3.2 Advanced ADC architectures

For high-speed, high-resolution systems (area marked by T), this is quite difficult to achieve with ”standard” single pass architectures so in recent years, parallel architectures which use multiple ADC’s in parallel have been proposed and analyzed. From a conceptual point of view they all can be grouped into two groups: averaging

and time-interleaving. Signal averaging is used to increase resolution without loss of speed and time interleaving, or to increase sampling rates without loss of resolution. Both of these are going to be analyzed in the next paragraphs.

Averaging architecture

The main idea of this configuration is that signals add directly, while noise from the individual ADCs (assumed to be uncorrelated) sum as the root sum square, so summing improves the overall SNR. Summing the outputs of four ADCs improves the SNR by 6 dB, or 1 LSB. Table 2.1 shows the increased SNR that results from summing the outputs of multiple ADCs. Figure 2.35 shows a possible architectural implementation of this configuration employing 4 ADCs.

Number of ADCs	Increase in SNR (dB)
2	3
4	6
8	9
16	12
32	15

Table 2.1: increase in SNR vs. Number of ADCs. (table from [2])

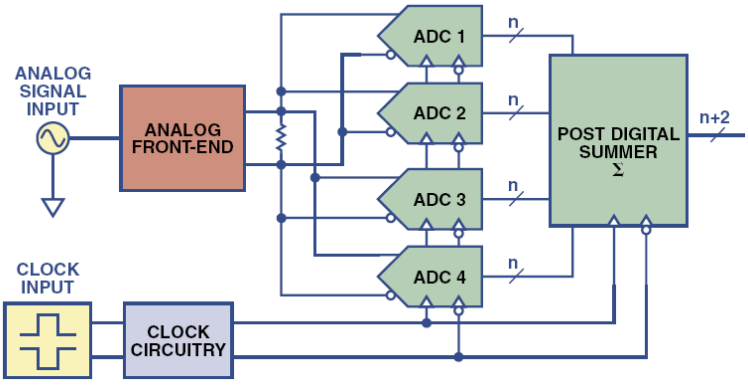


Figure 2.35: averaging technique using four ADCs in parallel. (figure from [2])

Time-interleaving architecture

Time interleaving of M ADCs allows the sample rate to be increased by the factor, M. By properly phasing each ADC's clock signal, the maximum sample rate of any

standard integrated circuit ADC type can be multiplied by the number of ADCs in the system. The proper clock phase required for the ADC number m can be calculated using the following relationship:

$$\phi_m = 2\pi \frac{m-1}{M} \tag{2.3.1}$$

Figure 2.36 shows a possible architectural implementation of this configuration employing 4 ADCs. By filtering the output signal bandwidth of such architecture, it is

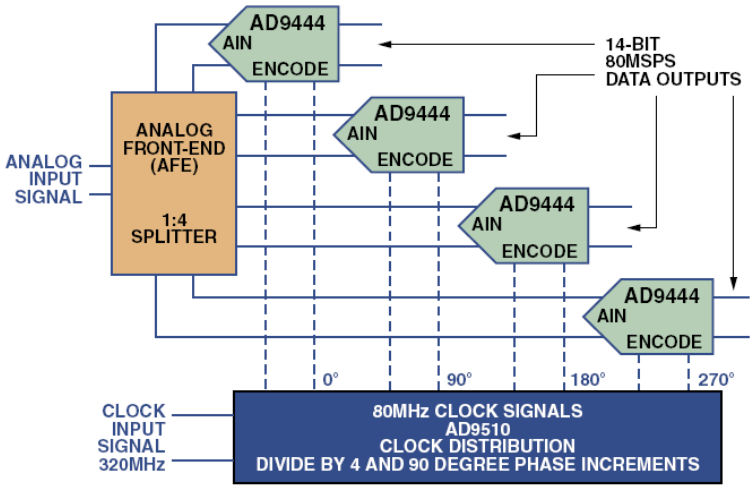


Figure 2.36: time interleaving technique using four ADCs in parallel. (figure from [2])

possible to achieve a resolution improvement. The thermal and quantization noise are spread over the overall output signal bandwidth and reducing bandwidth removes noise as well. Looking at the averaging technique in the frequency domain, it is possible to notice that the resolution is improved in exactly the same way. Averaging in the time domain is equivalent to low-pass filtering in the frequency domain with a sinc function.

Comparing the two configurations, they offer the same performance n area occupation, speed and resolution. However from another perspective, time-interleaving is the more challenging. Given technology related issues, speed improvements are much easier to obtain than improvements in resolution. Over the years, many techniques have been developed and formalized in order to trade the enhanced transistor speed with resolution. In this thesis time-interleaved architectures are going to be

discussed and analyzed in detail. A proposed algorithm is also presented in order to solve some unsolved problems relate to this kind of conversion architecture.

Spurs generation in time-interleaving architectures

The main problem related with architectures employing time interleaving is that if there are mismatches between converters in terms of gain, offset or deterministic sampling time uncertainty, distortion harmonics are generated and the overall SFDR and SNR are greatly worsened. This section is going to quantitatively analyze problems connected with these kind of systems.

A time-interleaved ADC consists of a number of channel ADCs, which have the same sampling rate but different sampling phases, as if they were a single converter operating at an times higher sampling rate [41], [42]. The time-interleaved architecture is illustrated in fig. 2.37, where each channel ADC operates with a sampling frequency of fs/M. Each time a sample is taken, a digital output is produced. Hence, each channel ADC has a sampling period of MTs, whereas the overall time-interleaved system has a sampling period of Ts. Unfortunately, it is a drawback of time-interleaved ADCs that any mismatch between the channel ADCs causes spurious components in the spectrum, degrading the signal-to-noise-and-distortion ratio (SINAD) [43]. Three types of mismatches result in the main degradation of the SINAD: gain, offset, and timing mismatches. Fig. 2.38 shows how all these non-idealities can be modelled. In this model the deviation from the ideal sampling period (i.e., the timing mismatch) is modelled as a time shift of the input signal, which simplifies further calculations. Additionally, each channel ADC has a gain and an offset value. Therefore, the sampled output of one channel ADC becomes:

$$y_l(t) = \sum_{k=-\infty}^{\infty} (g_l x(t - \Delta t_l) + o_l) \cdot \delta(t - kMTs - lTs) \tag{2.3.2}$$

Apart from global offset and gain errors, no mismatch errors would occur if these parameters were identical for all channel ADCs in the system. However, if they differ, spurious tones appear in the spectrum as shown in fig. 2.39, where a sinusoidal input signal has been coherently sampled. For this time-interleaved ADC model,

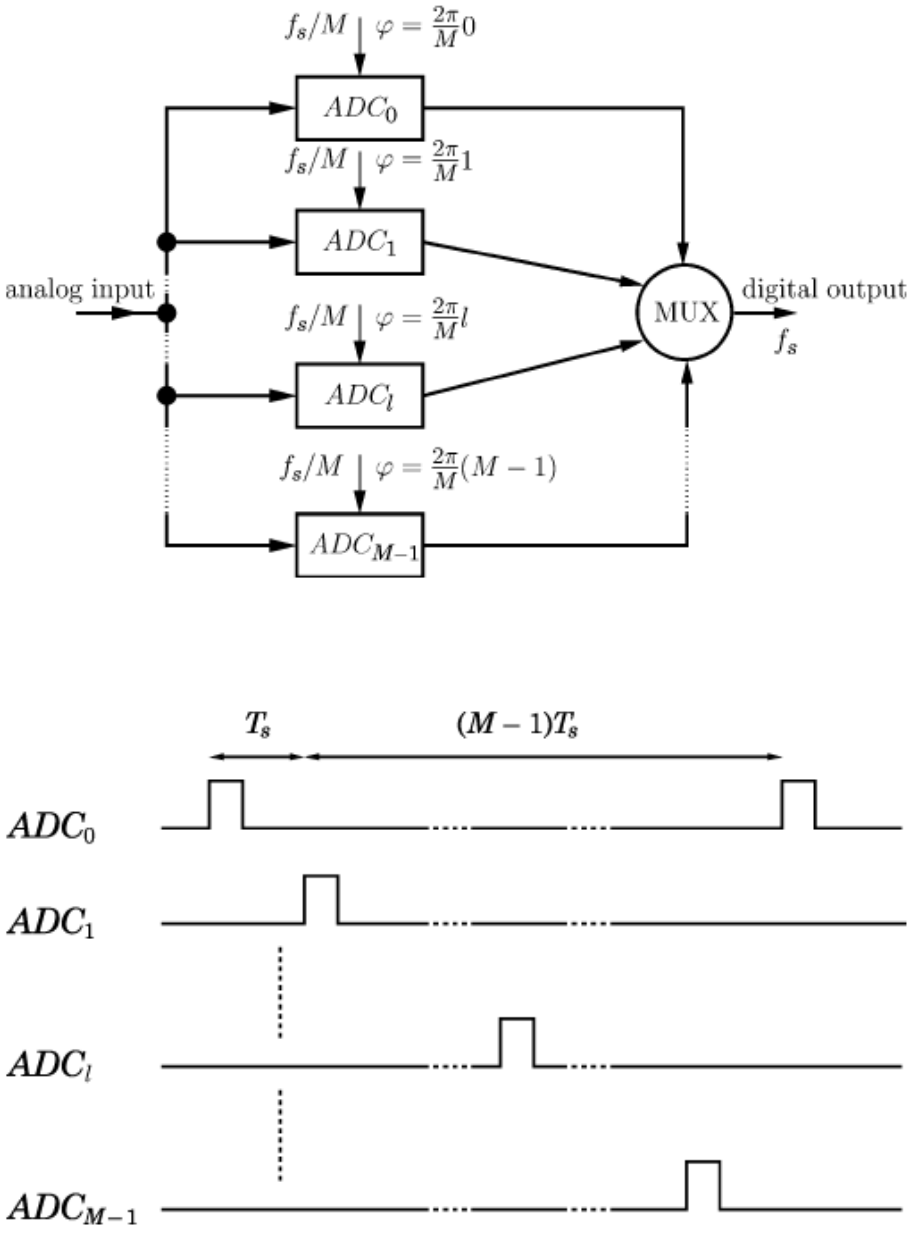


Figure 2.37: time interleaved ADC Architecture with M channels. (figure from [44])

the Fourier transform of the overall sampled output has the form (see [44])

$$Y(j\Omega) = \frac{2\pi}{T_s} \sum_{p=-\infty}^{\infty} \alpha[p]\delta(\Omega - \Omega_0 - p\frac{\Omega_s}{M}) - \alpha^*[M-p] \cdot \\ \cdot\delta(\Omega + \Omega_0 - p\frac{\Omega_s}{M}) + \beta[p]\delta(\Omega - p\frac{\Omega_s}{M}) \tag{2.3.3}$$

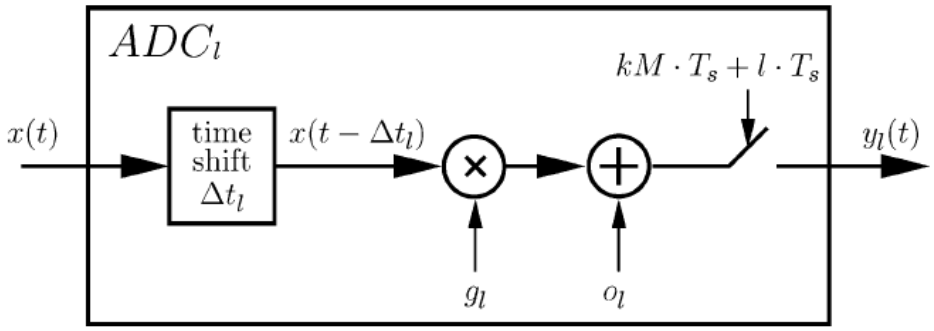


Figure 2.38: mathematical model of a channel ADC. (figure from [44])

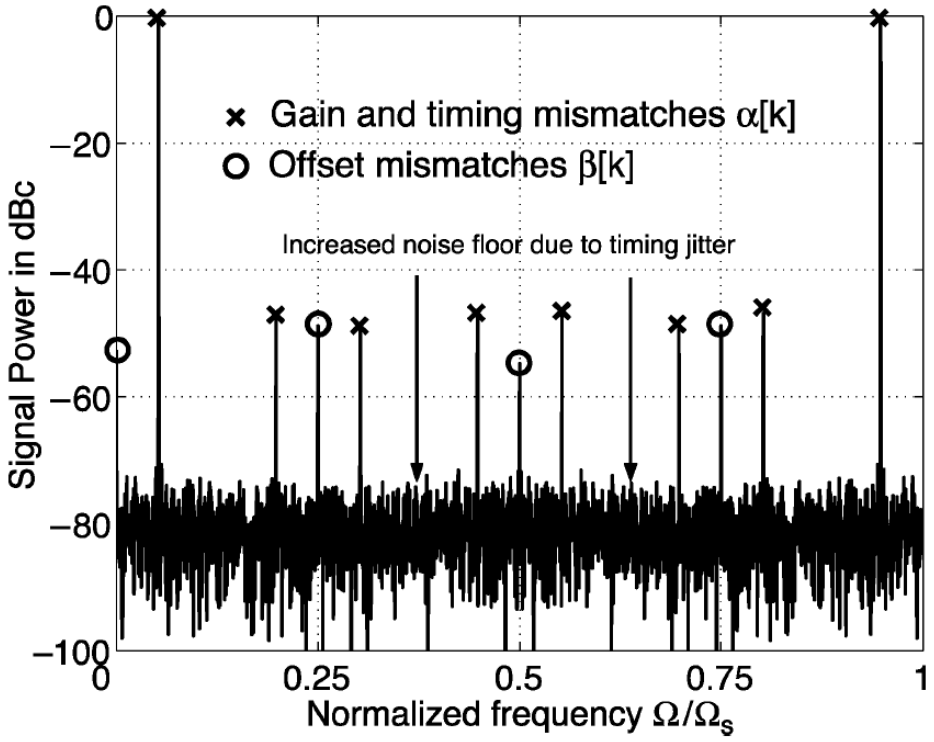


Figure 2.39: example of a spectrum of a Time Interleaved ADC with gain, time and offset mismatches. (figure from [44])

where

$$x(t) = Asin(\Omega_0 t)$$

$$\Omega_s = 2\pi/T_s$$

$$\alpha[p] = \frac{A}{j2M} \sum_{l=0}^{M-1} g_l e^{-j\Omega_0 \Delta t_l} e^{-jpl\frac{2\pi}{M}}$$

$$\beta[p] = \frac{1}{M} \sum_{l=0}^{M-1} o_l e^{-jpl\frac{2\pi}{M}} \tag{2.3.4}$$

The symbol * in 2.3.3 marks complex conjugation, Ω_0 is the input signal frequency and Ω_s is the sampling frequency of the overall architecture. The Fourier transform shows that some additional spurious peaks are centered at $\pm\Omega_0 + p\Omega_s/M$ in the case of gain and timing mismatches, whereas others are centered at $k\Omega_s/M$ in the case of offset mismatches. Where M is the number of ADCs and p is an index varying from 1 to M. If these spurs occur, the SFDR is greatly worsened, while the overall SINAD is given by the equation reported in [44]. Figure 2.40 shows it graphically for Gaussian distributed mismatches errors. Some important consideration can be

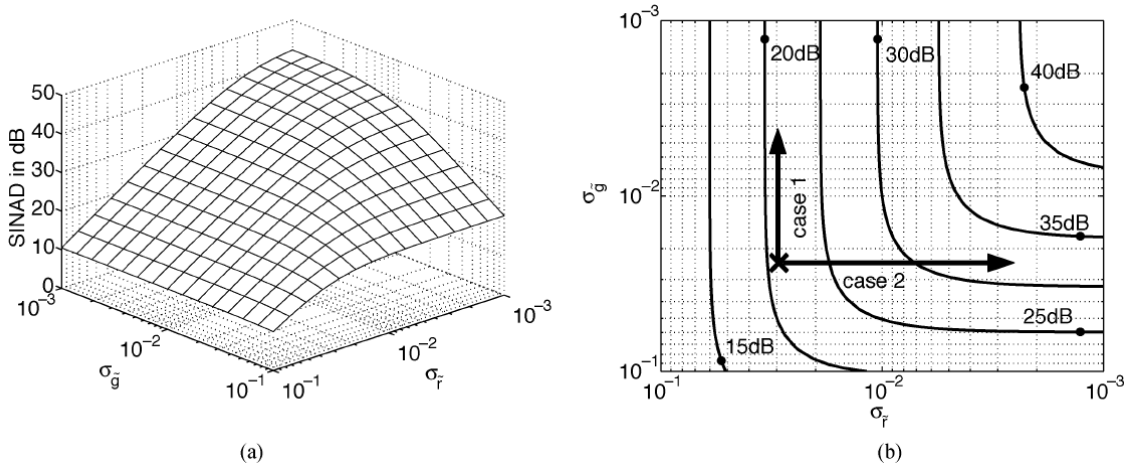


Figure 2.40: SINAD for gaussian distributed gain, offset and timing mismatch errors. a)SINAD for fixed offset standard deviation=0.5% b)Isolines of fig. a). [44]

made in order to design a system able to avoid this negative effect. Assuming that in order to improve the SINAD, it could either be possible to develop some method to reduce the gain mismatch (case 1) or the timing mismatch (case 2). It can be noted from fig. 2.40b that for the first case, there's only a slightly improvement of the expected SINAD no matter how sophisticated our gain mismatch compensation method is. The reason is that usually a little timing mismatch may undermine the performance of the time-interleaved ADC. This result come from a complex analysis carried on in [44]. The analysis takes into account many effects and models them using complex functions that take into account both deterministic and statistical variations of mismatch parameters.

Concluding, basing on previously described considerations, to improve the ex-

pected SINAD a timing mismatch compensation method must be found.

2.4 Summary

In this chapter an overview about ADCs has been given. Main performance parameter are described in section 1. Section 2 makes a functional description about ADCs and describes main issues, limitations and key points about these kind of devices. State of the art solutions and techniques to overcome main problems have been described. Concluding the main point of this chapter is to point out some promising architectural solutions that can be used as starting point to develop a new system able to improve the current state of the art. The outcome of this investigation is that time interleaving architectures offer the best opportunity to achieve the desired goal. A number of unsolved problems have been raised . The next chapter is going to describe state of the art solutions for these issues while a new solution is presented in chapter 4 that addresses timing mismatch issues.

Chapter 3

Comparison of correction techniques for distortion in time-interleaved systems

3.1 Introduction

The easiest and most natural solution to the problem analyzed before is an off-line fixed calibration of the time interleaved ADC. This kind of calibration is made while the system is not working. The problem with this is that even if this technique can be performed in an optimal way, it is effective only in the determined condition in which the system is calibrated. Variables like room temperature, power dissipation and chip temperature affects converter gain, offset and deterministic sampling time uncertainty. In high speed, high resolution systems, the amount of power dissipation and resolution required are both high. Due to this reason, even if the system is perfectly calibrated in a certain condition, even a little increase of the chip temperature can undermine the overall calibration. The way to calibrate these systems is to operate a calibration uninterruptedly while the system is running. This type of calibration is called online or dynamic calibration. In recent years various online techniques have been proposed to correct or avoid these errors. All these techniques can be classified into three main types. To the first group belongs all systems that aims to prevent the formation of distortion harmonics in the output signal. To the

second group belongs all systems that avoid distortion by a former identification of spurs and than applying a correction. The third group contains all systems that shape distortion harmonic noise power in order to improve SFDR and SNR. Fig.3.1 shows a graphical representation of the previous classification. The next sections of this chapter are going to analyze in detail all systems belonging to each group in order to find advantages and disadvantages of the respective techniques. In the next pages main correction techniques are going to be presented and discussed.

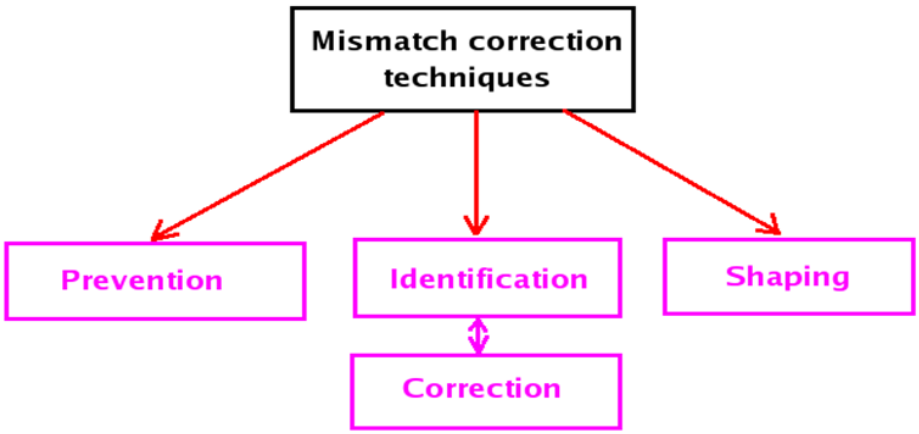


Figure 3.1: Mismatches induced distortion harmonics elimination techniques classification.

3.2 Prevention: Advanced Filter Banks

3.2.1 Main principle

This techniques avoid distortion harmonics preventing the formation of them. The latest and most successful technique in this area is called Advanced Filter Bank (AFB) and was first presented by S.R.Velazquez in his Ph.D. thesis [45]. The AFB architecture is illustrated in fig. 3.2. The AFB employs decomposition splitters, D_k, to divide the wideband analog input signal into M channel signals. By using time-skewed channel clock signals for time-division multiplexing, the decomposition splitter can simply be implemented as a high frequency splitter network. The channel signals are sampled at 1/M the effective sample rate of the system and converted to

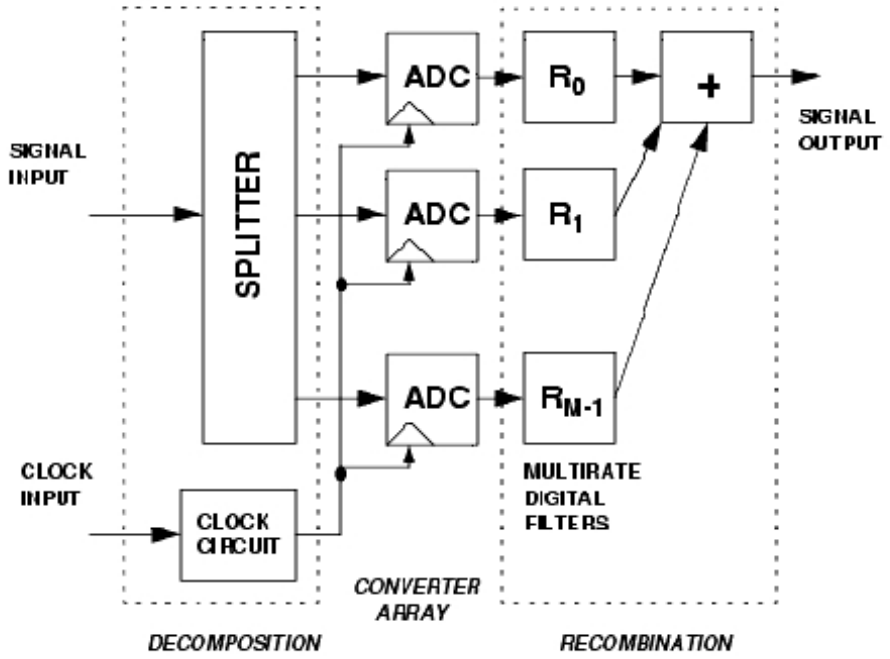


Figure 3.2: Advanced Filter Bank technique. (figure from [45])

digital signals with n-bit ADCs. The digitized channel signals are upsampled by M and reconstructed via digital recombination filters, $R_k(z)$. The recombination filters directly target the alias imaging errors caused by mismatches between converters in the array to insure that they do not limit the resolution of the system. The effective sample rate of the system is M times that of the channel ADCs in the array, and the resolution is n bits, the same as that of the channel ADCs in the array.

The AFB approach can be seen as a generalization of the T.I. architecture. Figures 3.3 and 3.4 compare block diagrams of both techniques. Comparing figures 3.3 and 3.4, it can be noted that T.I. is a special case of AFB with filters chosen to be appropriate delays as shown in fig. 3.4. However, since this architecture doesn't mix high-frequency bands down to baseband, the AFB (like the Time-Interleaved system) does not have to overcome the need for high-speed, high-precision sampling circuitry. Recently manufacturers such as Maxim, Analog Devices and National Semiconductors have introduced off-the-shelf converters whose sampling analog bandwidth exceeds the converter's Nyquist bandwidth by a factor from five to ten, eliminating the need to implement additional sampling circuitry for use in the

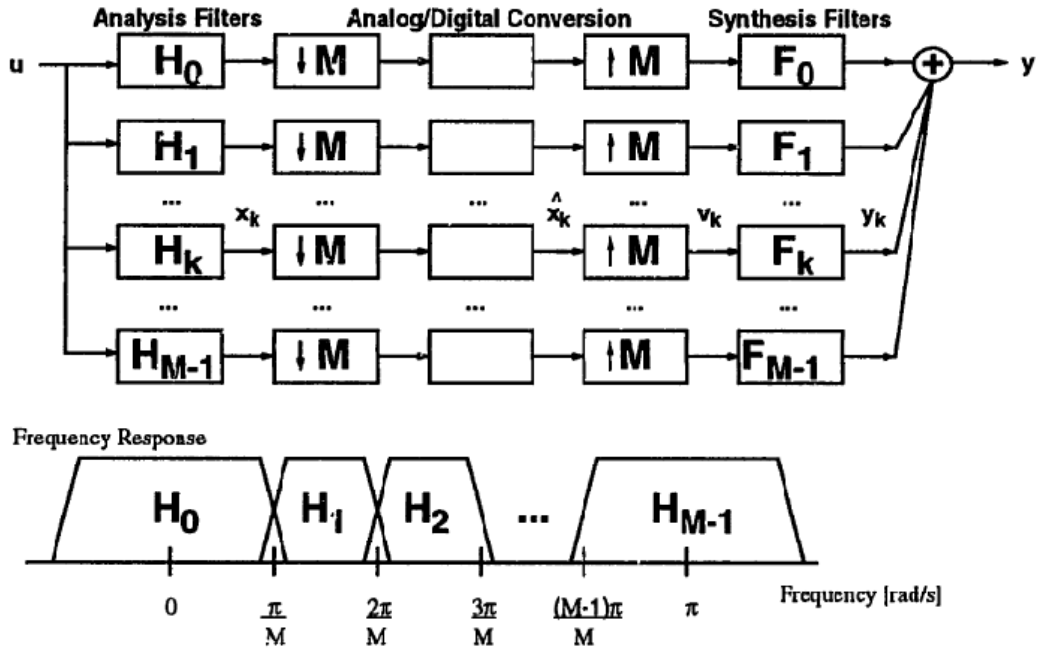


Figure 3.3: M-channel AFB technique. (figure from [45])

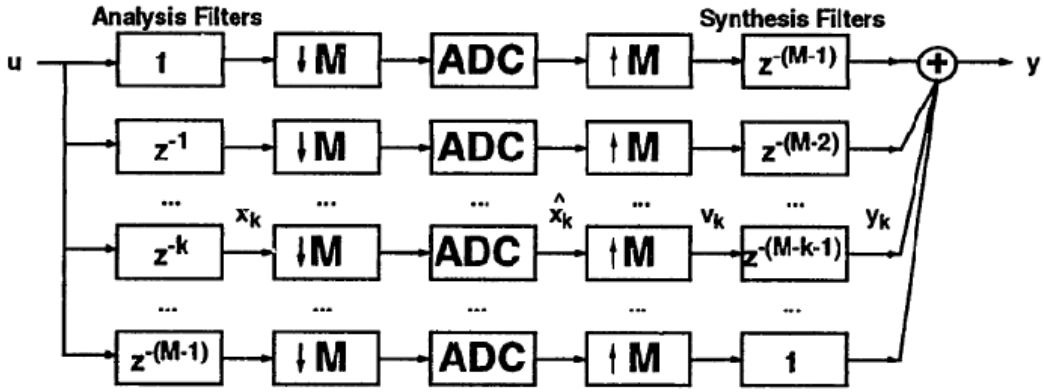


Figure 3.4: M-channel time interleaving technique. (figure from [45])

AFB or Time-Interleaving systems.

3.2.2 Comparison between AFB and a simple time-interleaving system

An accurate mathematical analysis of previous models is made in [45]. A brief summary of these results is presented below.

Gain and phase mismatches introduces in both systems aliasing error signals

which degrade the resolution of the system. The AFB attenuates the aliasing error signals much more than the TI system cause spurs falls into the stop-band of the filter and so they are greatly attenuated. The aliasing error is maximum in the AFB in correspondence of the filter transition frequencies where the filter attenuation is only -3 [dB]. As the T.I. system doesn't isolate the converters from each other, aliasing errors statistically average and decrease as the number of converters in the array, M, increases. However, this decrease is only a modest 3 dB per doubling of M. This averaging effect is not seen in the AFB since the filters isolate the converters from each others, so the maximum aliasing error in the AFB is actually greater than that of the TI case. AFB filters should have the smallest transition bandwidth as possible in order to reduce this Achille's heel of the system.

Another difference between the two systems is that filters can be used to compensate for gain mismatches as they need to be matched only at the transition bands; in TI calibration must be performed on the overall bandwidth.

DC offset errors introduce an error signal which has constant frequency. As the TI system doesn't isolate the converters from each other, DC-offset errors statistically average and therefore decrease as the number of converters in the array, M, increases. As with aliasing error, this decrease is again 3 dB per doubling of M. This averaging effect is not seen in the HFB since the filters isolate the converters from each other. The error signal has frequency content at the crossover frequencies of the filters and signals are therefore not attenuated by the stop-band of the filters, so the DC-offset error in the AFB is actually greater than that of the TI case. Clearly DC-offset errors can be significant and can limit the resolution of the system and should be nulled to zero as part of the calibration of the system.

3.2.3 AFB Design Example

The AD12400 is the first member of a new family of Analog Devices products that leverage both time interleaving and AFB. Its performance will be used to illustrate what can be achieved when state-of-the-art ADC design is combined with advanced digital post-processing technologies. Fig. 3.5 illustrates the AD12400's block diagram and its key circuit functions. The AD12400 employs a unique analog front-end

circuit with 400-MHz input bandwidth, two 12-bit, 200-MSPS ADC channels, and an AFB implementation using an advanced field-programmable gate array (FPGA). It was designed using many of the classical matching techniques discussed above, together with a very low jitter clock distribution circuit. These key components are combined to develop a 12-bit, 400-MSPS ADC module that performs very well over 90% of the Nyquist band and over an 85°C temperature range. It has an analog input bandwidth of 400MHz. The impact of the AFB technique can be seen in fig.

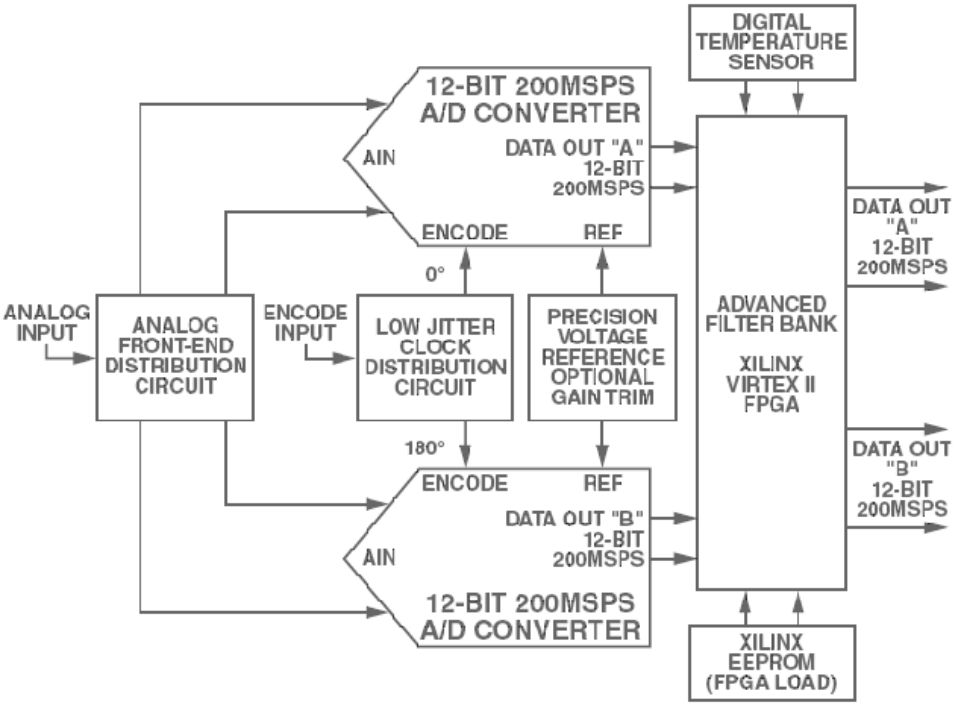


Figure 3.5: AD12400 block diagram. (figure from [2])

3.6.

The first curve in fig. 3.6(a) represents the performance of a 2-channel time-interleaved system that has been carefully designed to provide optimal matching in the layout. The behavior of the image spur in this curve makes it obvious that this system was manually trimmed at an analog input frequency of 128 MHz. A similar observation of fig. 3.6(b) suggests a manual trim temperature of 40°C. Despite a careful PCB layout, tightly matched front-end circuit, tightly matched clock-distribution circuit, and common reference voltages used in the AD12400 ADC, the dynamic range degrades rapidly as the frequency and/or temperature deviates from

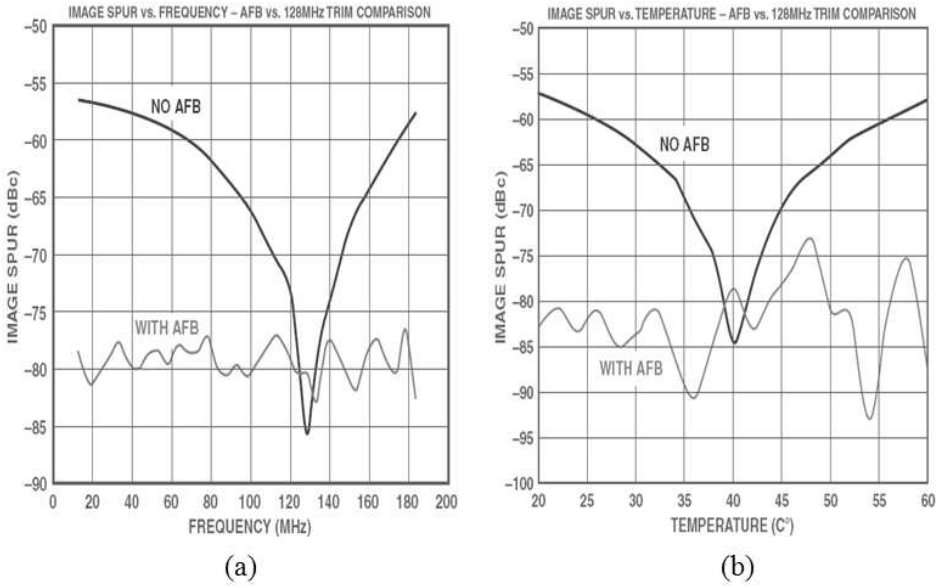


Figure 3.6: performance of a manually trimmed system 'before and after' AFB compensation over the:(a) frequency range, (b) temperature range. (figure from [2])

the manual trim conditions. This rapid rate of degradation can be anticipated in any two-converter time-interleaved ADC system by analyzing some of the sensitive factors affecting this circuit. For example, the gain-temperature coefficient of a typical high-performance, 12-bit ADC is 0.02%°C. In this case, a 10°C change in temperature would cause a 0.2% change in gain, resulting in an image spur of 60 dBc. Considering just this single ADC temperature characteristic, the predicted image spur is 3 dB worse than the 30°C performance displayed in fig. 3.6(b). By contrast, the dynamic range performance shown in these figures remains solid when the AFB compensation is enabled. In fact, the dynamic range performance surpasses the 12-bit level across a bandwidth of nearly 190 MHz and a temperature range of 40°C. Another significant advantage of this approach is that the temperature range can actually be expanded from the 20°C to 60°C range shown to 0°C to 85°C by using additional FIR coefficient sets-as embodied in the AD12400.

3.3 Identification & Correction of Mismatches.

The second technique that can be used to overcome mismatch issues is to identify the mismatches sources and then correct for them. All techniques in this group can be performed dynamically and can continuously recalibrate the system while it is working in order to follow mismatches variations. Fig. 3.7 shows graphically how these techniques works.

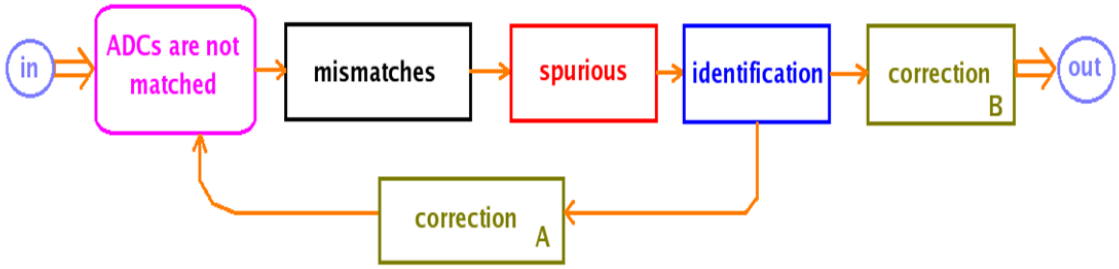


Figure 3.7: block diagram of how identification&correction techniques works.

3.3.1 Identification of Mismatches.

This is the first part of I&C (identification and correction) process , and it is also the most important one. The efficiency of correction techniques is greatly influenced by estimation errors. A system cannot be corrected if error parameters are affected by errors as well. Identification techniques can be grouped into two main groups: techniques that exploit statistical properties of the input signal (blind) and techniques that use reference signals (known signal). All signals in the following equations are referred to fig. 3.8.

Blind estimation:

With this technique, the three different error types can be estimated by minimization of different loss functions. A loss function is a function that can be calculated from the measured data and depends on a set of parameters, here the mismatch error parameters. The loss function should be strictly positive except for the true parameters, where it should be zero. This means that by minimizing the loss function

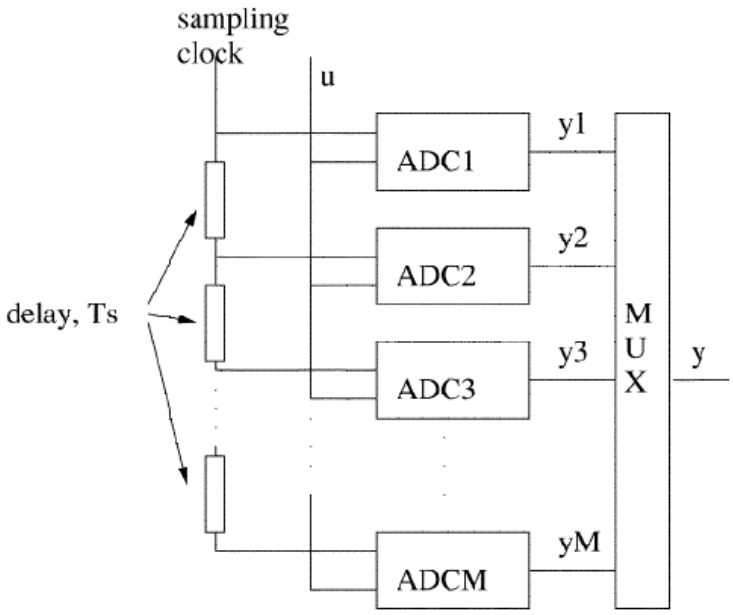


Figure 3.8: M channel time interleaved ADC system. (figure from [80])

we can get an estimate of the parameters. It is also an advantage if the loss function is convex, since it makes numerical minimization more robust. The different errors will first be discussed separately and then together to estimate all errors simultaneously. The dynamic performance of ADC systems is going to be studied. This means that it is not important that for instance the amplitude offset is zero as long as it is the same for all the ADCs in the system. Therefore, the reference level can be chosen arbitrarily and in the following we will assume that all the errors are zero in the first ADC, all the other errors are relative to the offset, gain, and time errors in the first ADC.

Amplitude Offset Error Estimation:

The idea for estimation of the offset errors is that the mean value of the output from each ADC corresponds to the respective offset errors [46]. It is first assumed that the time and gain errors are zero, then the influence of time and gain errors will be discussed. The signal model is:

$$y_i[k] = u((kM + i)T_s) + \Delta_{oi}^o \tag{3.3.1}$$

where Δ_{oi}^o is the real value of offset error. Assuming that $u[k]$ is quasi-stationary and modulo M quasi-stationary then the mean value of the input is the same for all

subsequences. Introducing the amplitude offset loss function:

$$V_o^{(N)}(\Delta_o) = \sum_{i=1}^{M-1}\sum_{j=0}^{i-1}(\frac{1}{N}\sum_{k=1}^{N} z_i^{(\Delta_{oi})}[k] - z_j^{(\Delta_{oj})}[k])^2 \tag{3.3.2}$$

where $z^{(\Delta_{oi})}[k]$ is the output signal y_k reconstructed with the error parameters Δ_o and $z^{(\Delta_{oi})}[k] = z^{(\Delta_{oi})}[k]$. It is shown in [47] that $V_o^{(inf)}(\Delta_o^o) = 0$, $\Delta_o = \Delta_o^o$ is the only minimum and that $V_o^{(inf)}(\Delta_o)$ is quadratic. More details are given in [47]. This mean that the minimizing argument of $V_o^{(N)}(\Delta_o)$ will converge to the true offset parameters. Since there are no local minima, any minimization algorithm will converge to the global minimum.

Gain Error Estimation:

The idea (proposed in [47]) for estimation of gain errors is that the variance of the output from each ADC corresponds to the respective gain of the ADC. It is assumed that the time and offset errors are zero, then the influence of time and offset errors can be discussed. The signal model is now :

$$y_i[k] = (1 + \Delta_{gi}^o)u((kM + i)T_s) \tag{3.3.3}$$

where δ_{gi}^o is the real value of gain error. Assuming that u[k] is quasi-stationary and modulo M quasi-stationary then the mean square value of the input is the same for all subsequences. The gain error loss function can be given as:

$$V_g^{(N)}(\Delta_g) = \sum_{i=1}^{M-1}\sum_{j=0}^{i-1}(\frac{1}{N}\sum_{k=1}^{N}(z_i^{(\Delta_{gi})}[k])^2 - (z_j^{(\Delta_{gj})}[k])^2)^2 \tag{3.3.4}$$

where $z^{(\Delta_{gi})}[k]$ is the output signal y_k reconstructed with the error parameters Δ_g and $z_i^{(\Delta_{gi})}[k] = z^{(\Delta_{gi})}[kM + i]$. It can be shown in [47] that $V_g^{(inf)}(\Delta_g^o) = 0$, and that $V_g^{(inf)}(\Delta_g) > 0$ if $\Delta_g \neq \Delta_g^0$, which means that the minimum at $\Delta_g = \Delta_g^0$ is the global minimum. More details are given in [47]. The gain error loss function is not convex, but simulations still show good performance.

Time Error Estimation:

The basic idea (proposed in [48]) for the time error estimation is to study the mean square difference between the outputs of adjacent ADCs. Assuming that the input signal is band limited to the Nyquist frequency, the signal cannot change arbitrarily fast. If the time interval between two ADCs is shorter than Ts the signal

will change less on average between the samples compared to a time difference of Ts and vice versa if the time interval is no longer than Ts. In figure 3.9 this is illustrated for a dual ADC system. Assuming that the offset and gain errors are zero, then the

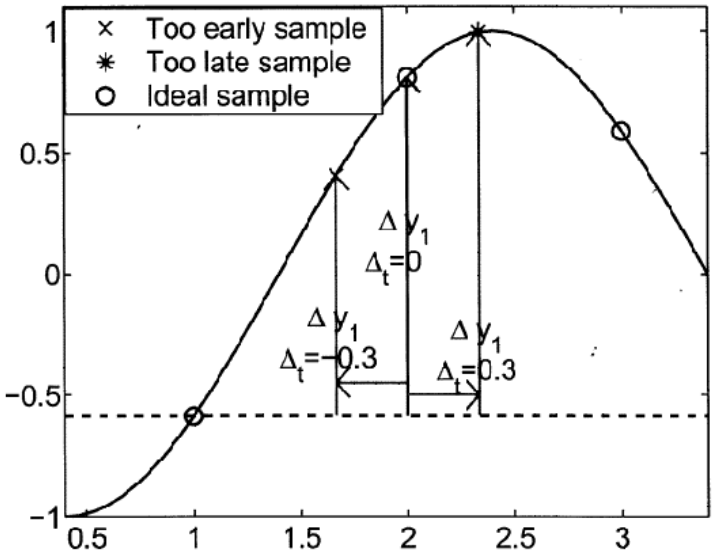


Figure 3.9: proposed idea for the time-error estimation. (figure from [48])

influence of timing mismatch errors can be discussed. The signal model is given by:

$$y_i[k] = u((kM + i)T_s + \Delta_{ti}^o) \tag{3.3.5}$$

where Δ_{ti}^o is the real value of time error. Assuming that u[k] is quasi-stationary and modulo M quasi-stationary and considering that the time error functions are:

$$V_{t,R}^{(N)}(\Delta_t) = \sum_{i=1}^{M-1}\sum_{j=0}^{i-1}(\overline{R}_{z_i z_{i-1}}^{(N),(\Delta_t)}[0] - R_{z_j z_{j-1}}^{(N),(\Delta_t)}[0])^2 \tag{3.3.6}$$

$$V_{t,\sigma}^{(N)}(\Delta_t) = \sum_{i=1}^{M-1}\sum_{j=0}^{i-1}(\frac{1}{N}\sum_{k=1}^{N}(z_i^{(\Delta_{ti})}[k])^2 - (z_j^{(\Delta_{tj})}[k])^2)^2 \tag{3.3.7}$$

where $z^{(\Delta_{ti})}[k]$ is the output signal y_k reconstructed with the error parameters Δ_t and $z_i^{(\Delta_{ti})}[k] = z_i^{(\Delta_{ti})}[kM + i]$. R is the autocorrelation operator more details are given in [48]. Adding two loss functions the obtained time error loss function is:

$$V_t^N(\Delta_t) = V_{t,R}^N(\Delta_t) + V_{t,\sigma}^N(\Delta_t) \tag{3.3.8}$$

It can be shown that $V_t^{(inf)}(\Delta_t) = 0$ and that the minimum at $\Delta_t = \Delta_t^0$ is the global minimum. To evaluate the performance of the mismatch error estimation method, a

time-interleaved ADC system has been simulated. The Cramer-Rao Bound (CRB) [49] is a lower bound, independent of the estimation method, on how good the estimation accuracy can be, given an amount of data. In the following simulations the estimation accuracy is compared to the CRB. It is impossible to reach the CRB since the CRB is calculated assuming known input, but it is still interesting to study how close it is possible to be to the CRB. To compare the estimation accuracy with the CRB, the minimization has been done on one batch of data instead of updating with new data for each iteration. The estimation algorithm has been tested with different input signals and different signal parameters have been varied. Results are shown in figure 3.10. As can be noted from previous figures,

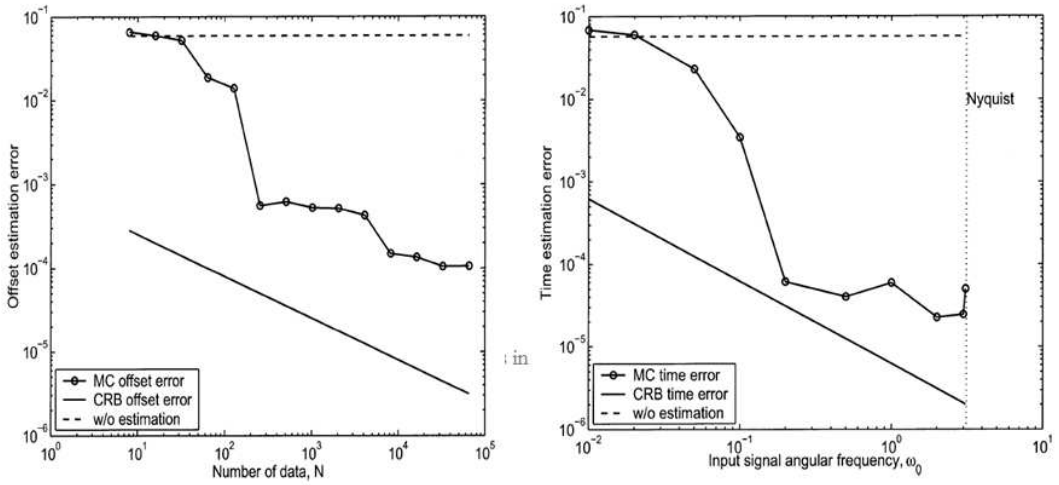


Figure 3.10: estimation accuracy in offset and time errors compared with CRB. (figure from [48])

this estimation technique is very close to CRB bound. Due to the fact that this method assumes that gain and amplitude errors are removed before the estimation of time errors, we need to investigate how an estimation error in one parameter influence the performance of the estimation of others parameters. Simulations have been performed and results are reported in fig. 3.11 in the case of time estimation errors with gain and amplitude offset errors. From previous figures it can be noted that small gain and amplitude errors do not affect the estimation accuracy for most input signals.

A second idea (proposed in [54]) for time error estimation detects the timing

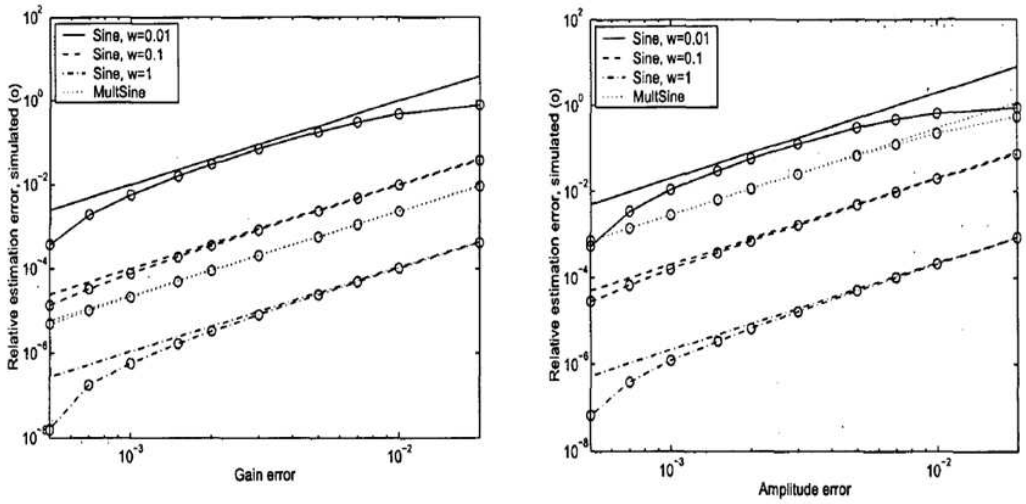


Figure 3.11: comparison between theoretical time estimation error and time estimation error from simulations as a function of gain and offset amplitude error size for four different input signal. (figure from [80])

skews by simply counting the number of zero crossings of input signal among the A/D channels while randomly changing their sampling sequence. The theoretical basis of this method is now explained. Firstly it is assumed that the sampling rate, is larger than the Nyquist sampling frequency for the input, u(t). Since the signal is continuous in time and in amplitude, there is one and only one moment between two consecutive sampling instant that the u(t) crosses over the zero. This happens if the input's values sampled by the corresponding A/D channels, ADC_j and the subsequent ADC_{j+1}, have opposite signs, i.e.: $y_j[k] \cdot y_{j+1}[k] < 0$, where $y_i[k] = u((kM+i)T_s + \Delta_{ti}^o)$. Second, it is assumed that u(t) is a stationary Gaussian process with zero mean. Then, the probability of a zero crossing between ADC_j and ADC_{j+1},$P_{j,j+1}^z$ is a bivariate normal distribution [50], [51], and can be expressed as :

$$P_{j,j+1}^z = \frac{1}{2} - \frac{1}{\pi} sin^{-1} \rho_{j,j+1} \tag{3.3.9}$$

with :

$$\rho_{j,j+1} = \frac{E(u_j u_{j+1})}{\sigma_j \sigma_{j+1}} \tag{3.3.10}$$

where σ_j and σ_{j+1} are the standard deviations of the u_j and u_{j+1} random variables respectively. The $\rho_{j,j+1}$ of 3.3.9 denotes the cross-correlation between u_j and u_{j+1}.

As explained in previous citations, $P^z_{j,j+1}$ is a monotonic function of the timing mismatch $\Delta T_{j,j+1}$ between channel j and j+1 for an input with limited bandwidth. Therefore by looking at zero crossing transition probabilities it is possible to estimate timing mismatches between channels. Simulation and a more detailed analysis are reported in [54].

Known signal Estimation

This approach uses a known signal input to the system and the resultant outputs are analysed to detect errors. The approach is a parallel system is slightly more complicated. The block diagram of the ADC system is shown in figure 3.12. At any

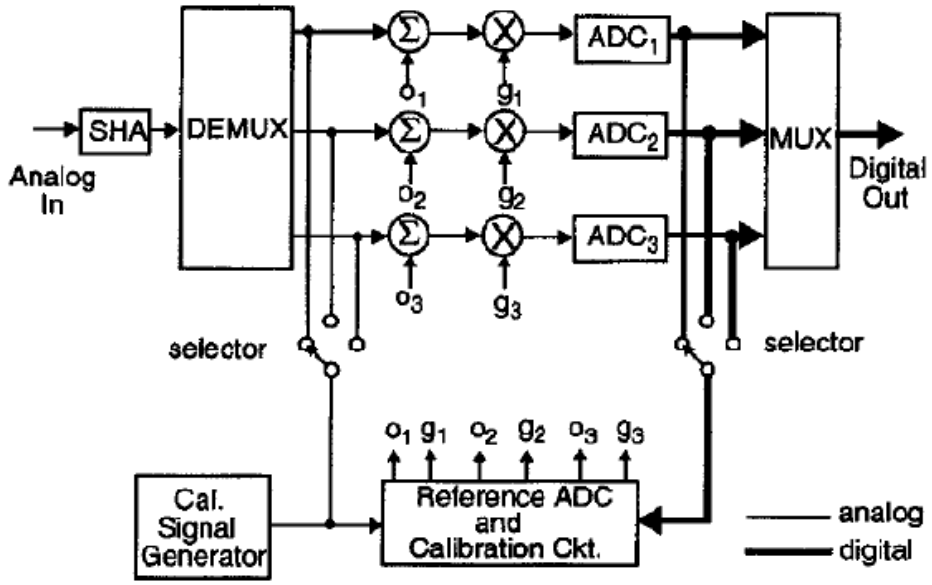


Figure 3.12: generalized known signal calibration system. (figure from [81])

time, two of the three ADCs process the input in a time-interleaved fashion, giving a conversion rate that is double that of each component ADC. The other pipelined ADC is connected in a calibration loop, that adjusts the gain (g) and offset (o) of the selected ADC to match those of the reference ADC. Time mismatches are not considered here but this system can be extended to those errors. Once the calibration cycle is completed, another ADC is selected for calibration, and the most-recently calibrated ADC replaces it in the ping-pong conversion mode. Each

ADC is calibrated periodically in the background while the other ADCs continue the conversion of the input signal. Due to the fact that this system interrupts the conversion flow inside each converter during its calibration phase, this system is not suitable for ADCs that needs past samples to perform the conversion, for example sigma-delta architectures. If one of the modulators is periodically replaced with a calibrated one, a periodic deviation of the single path history is introduced with respect to the ideal one. This causes a discontinuity in the output in the time slots including the switching transient resulting in an unacceptable degradation of the SNDR. A solution for this kind of converters has been proposed in [55] and [56]. This technique is based on the addition of a calibration signal, generated by a pseudo-random number generator, to the ADC input. Both signals are then processed simultaneously by means of an adaptive algorithm, which digitally filters out the output to remove residual tones. In this case there is no need for an extra parallel channel. This on-line method can be applied to any type of ADC, however it exhibits some severe limitations. Firstly, part of the full-scale input range of the ADC is used by the calibration signal. Therefore, the ADC paths require extra resolution to achieve a given dynamic range. A suitable solution has been proposed by [58] and it is shown in fig. 3.13. As shown in fig. 3.13, instead of periodically placing one path in a calibration section, the reference path (CHANNEL) is connected in parallel to the path to calibrate. The parallel connection and can be realized by simply assigning the clock phases of the channel under calibration to the reference path. In the circuit of fig. 3.13 for example, the block called "AVERAGE" tries to eliminate the offset from the channel under calibration by comparing its output with the one of the reference channel. This way it is possible to calibrate every single ADC without interrupting the normal operation of it. Even if this technique sounds quite interesting, in sigma-delta converters there are same other problems to solve. The main important one is called the 'Domino effect'. The domino effect is defined as: "a certain quantizer output connected to another quantizer input via an analog block without any delay". This situation inevitably occurs in time interleaved sigma-delta modulators when consecutive time slot outputs of the modulator are simultaneously generated by the quantizers. The main idea to solve the problem of TI sigma delta

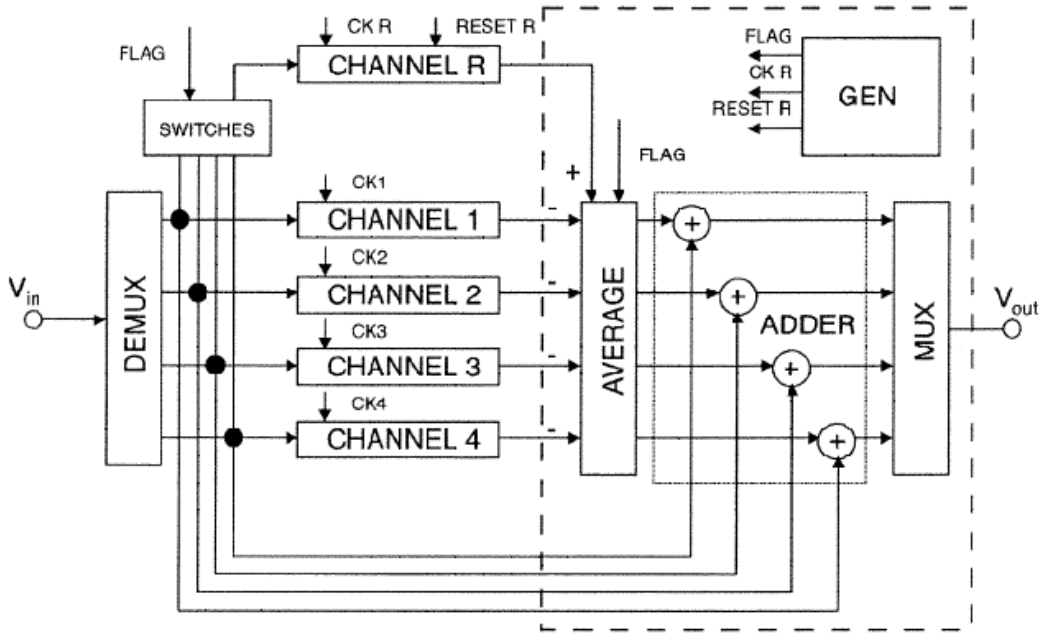


Figure 3.13: block diagram of technique proposed by [58] in relation to offset calibration.

system is to unify the overall TI architecture and to add extra signal path able to eliminate this effect. Figure 3.14 is an example of this. From this figure, the complex feedback network needed to avoid this effect can be seen. This approach has been shown to work well however it is difficult to ascertain the value of the mismatches to a high degree.

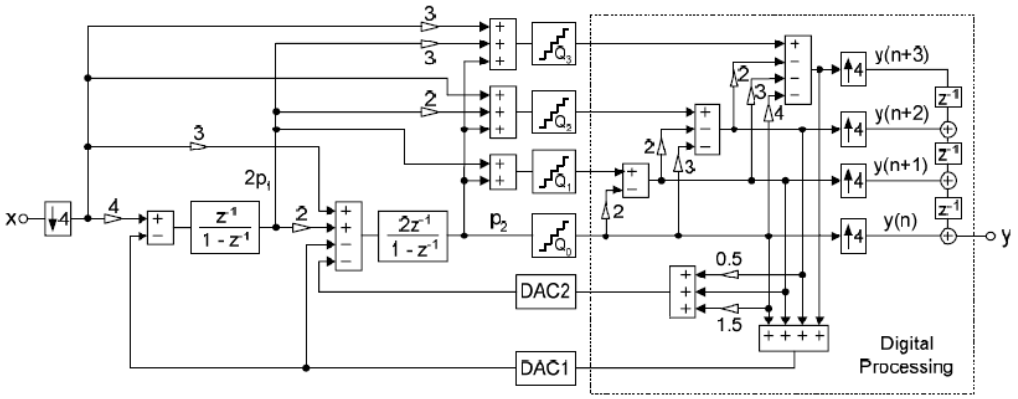


Figure 3.14: block diagram of technique proposed by [57] in relation to domino effect elimination.

3.3.2 Correction of Mismatches.

Once the mismatch error parameters have been identified, the calibration system tries to correct the conversion system in order to reduce these mismatches. This correction could be performed by modifying both each ADC parameter and performing post-processing operations on the sampled signal. To the first group belongs techniques that tries to trim in an analog or digital way each ADC (trimming). To the second one belongs techniques that tries to reconstruct the original signal and re-sample it with in the correct instant with correct gain and offset parameters (reconstruction). Following paragraphs are going to give an overview of these techniques.

Electronic trimming

This kind of trimming acts directly on the converter circuitry. In figure 3.7 it is represented by the block "correction" marked with the letter "A". Its purpose is to modify circuital parameters like capacitance or resistance values in order to correct for time, gain and offset mismatches. This technique is very simple and it is used widely in commercial converters. Figure 3.15 shows an example of a 18-bit, 1.25 MS/s Analog Devices ADC. This system works as follow: it analyzes the digital output of the ADC under calibration and then modifies ADC parameters in order to correct those mismatches. An example for the correction of the timing mismatches is shown in the figures 3.16 and 3.17. From these figures, it can be noted how this architecture modifies ADCs analog parameters on the basis of a digital control signal in order to correct for time, gain and offset mismatches. The problem with this architectural philosophy is that electronic devices used to correct mismatches are affected by mismatches as well. These kind of systems usually offer good performance for resolution usually lower than 18-bits. For systems with higher resolution like sigma-delta digital post-processing techniques (see next section) are used.

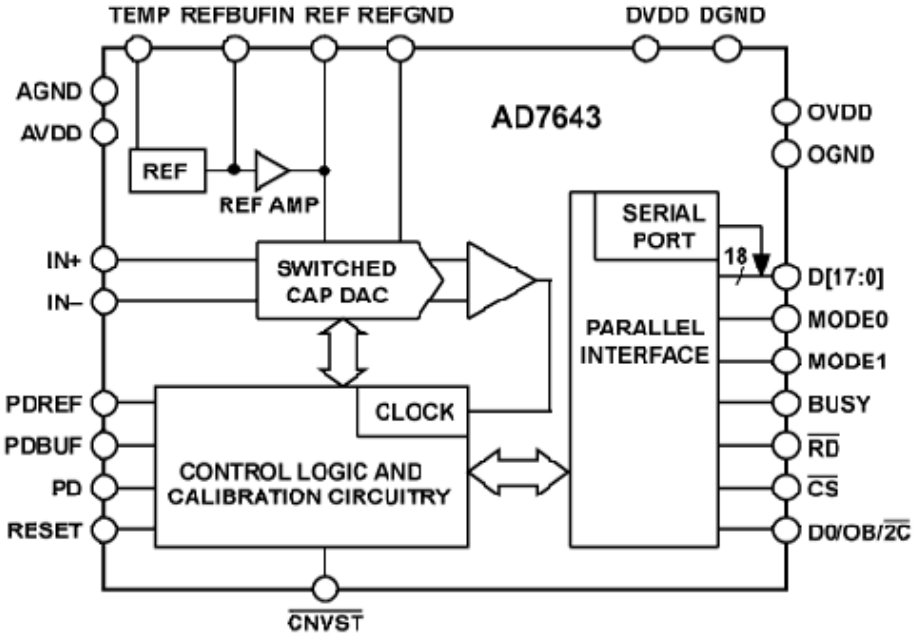


Figure 3.15: AD7643 block diagram. (figure from [2])

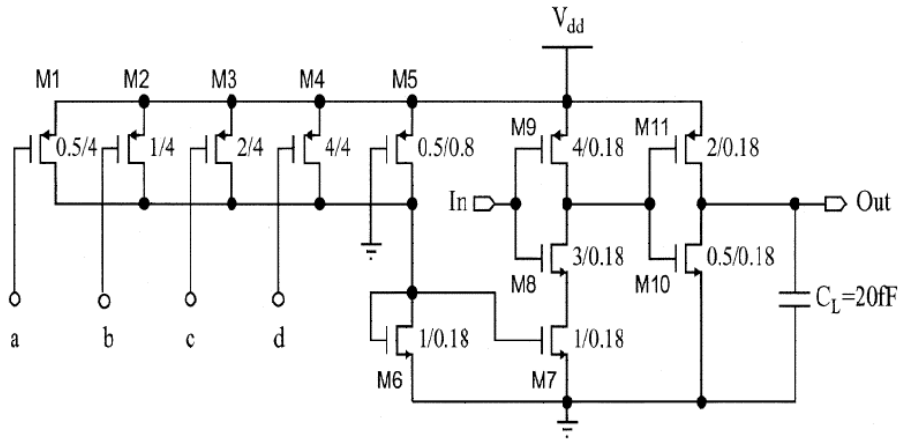


Figure 3.16: digitally programmable delay element proposed in [59].

Reconstruction

This kind of trimming doesn't actually change any analog parameter but adjusts the sampled output data through post-processing. In figure 3.7 it is represented by the block "correction" marked with the letter "B". The major advantage of this technique compared with the previous one is that all the calibration process is carried out using digital systems and no modifications are required to the analog one.

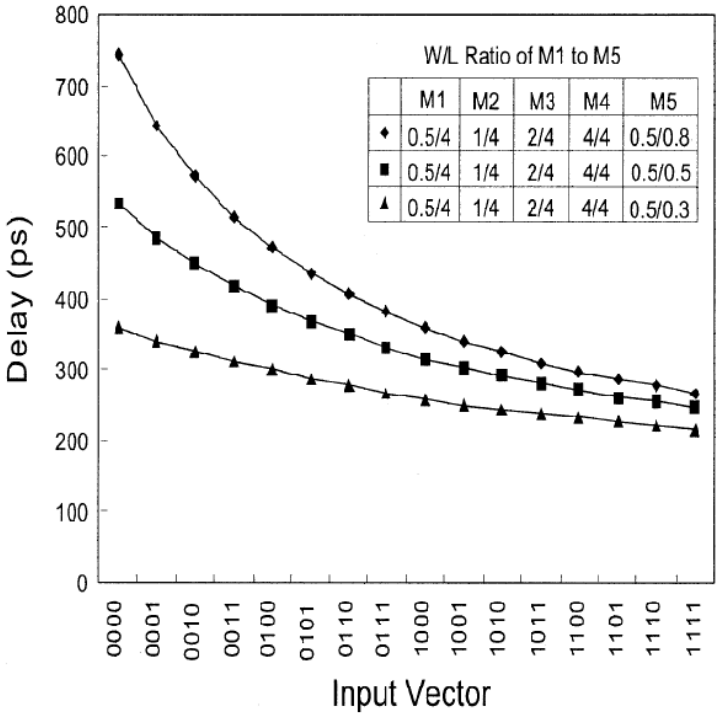


Figure 3.17: previous circuit delay versus input vector for different W/L ratios of transistors M1-M5. (figure from [59])

On the one hand digital post-processing usually is much more reliable than analog trimming as mismatches in a digital architecture don't influence the output results. On the other hand the reconstruction algorithm usually require a large amount of calculus capability not always available in Arithmetic Logic Units of small integrated DSP. This calculus capability depends on the reconstruction algorithm and usually the more accurate it is, the more computationally intense it becomes. This problem constitutes the main trade-off in this kind of reconstruction philosophy between resolution and computational complexity of the post-processing systems. To correct for gain or offset errors is easier as it's easy to multiply or add to the output sample a certain correction factor. Little computational complexity is required. To correct for timing mismatches is difficult and it requires significant computational power. The next section is going to give an overview about state of the art in fractional delay algorithms.

Fractional delay algorithms

Delaying a continuous-time signal xc(t) by an amount td is conceptually simple

cause we know its value in every instant. Delaying a uniformly sampled band-limited (baseband) digital signal must be treated with care. Sampling xc(t), its values is known only on time instants t = nTs, where n is an integer and Ts is the sampling interval. If someone wants to delay xc(t) by an amount of time that is a fraction of Ts, the value of the signal at that precise instant must be known. The only way to know that value is to use band-limited interpolation. The problem can be solved by viewing a delay as a resampling process. The desired solution can be obtained by first reconstructing the continuous band-limited signal and then resampling it after shifting.

Ideal solution

Assuming that the (real-valued) discrete-time signal represents a band-limited baseband signal, the implementation of a constant delay can be considered as an approximation of the ideal discrete-time linear-phase all-pass filter with unity magnitude and constant group delay of the given value D. The corresponding impulse response is obtained via the inverse discrete-time Fourier transform and equals to:

$$h(n) = \int_{-\pi}^{\pi} H(e^{jw}) e^{jwn} dw \quad for\ all\ n \tag{3.3.11}$$

Substituting $H(e^{jw})$ with e^{-jwD} into eq. 3.3.11 where D is the desired delay:

$$h_{id}(n) = \frac{sin[\pi(n-D)]}{\pi(n-D)} = sinc(n-D) \quad for\ all\ n \tag{3.3.12}$$

When the desired delay D assumes an integer value, the impulse response eq. 3.3.12 reduces to a single impulse at n=D, but for non-integer values of D the impulse response is an infinitely long, shifted and sampled version of the sinc function. Fig. 3.18 shows the impulse response of the filter with different delays. Unfortunately, the ideal impulse response is not only infinitely long but also non-causal, which makes it impossible to implement it in real-time applications. Equation 3.3.12 gives an answer to the original problem, i.e., where the delayed signal value should be placed as it cannot be put "between the samples." In the ideal case, it is to be spread over all the discrete-time signal values, weighted by appropriate values of the sinc function. This simple result is of fundamental importance since, whatever method is used, the impulse response of the approximating (real-coefficient) filter must imitate this ideal response in some meaningful sense. It is also evident that with a finite-order causal

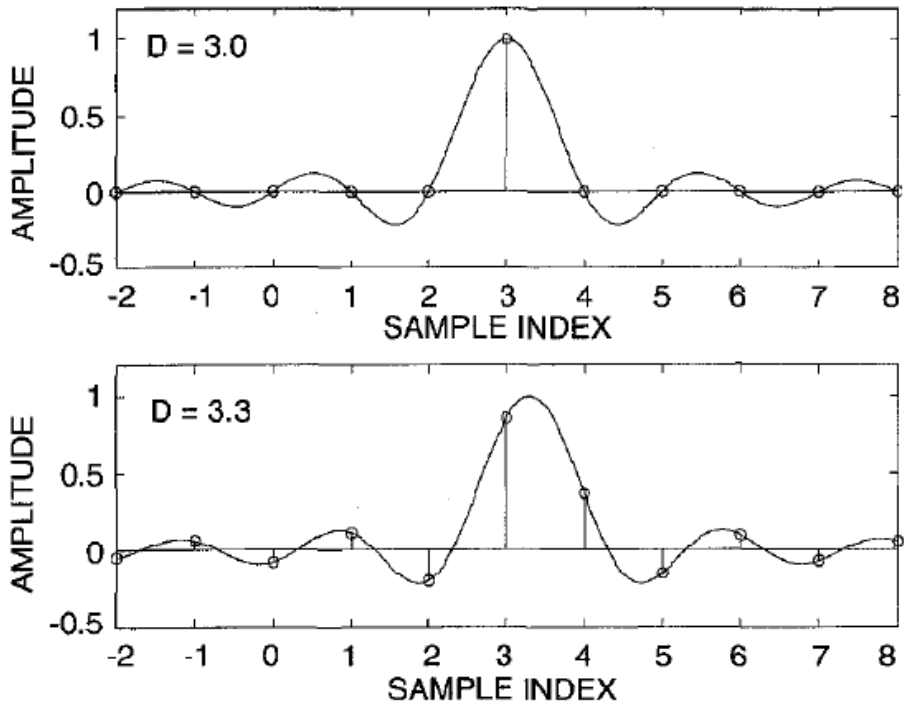


Figure 3.18: impulse response of an ideal delay filter with delay D=3 and D=3.3. (figure from [60])

FIR or IIR filter the ideal response can only be approximated. IIR filters are much more sensitive to numeric approximation problems than FIR ones. For this reason, IIR filters should be only used for high resolution applications where errors due to numeric approximation are small. Furthermore in IIR filters, the phase, phase delay, and group delay are all related to the filter coefficients in a nonlinear manner. This means that one can expect design formulas for IIR filters to be much more complex than the ones for FIR. Therefore FIR filters are preferred to IIR ones.

Windowing the ideal response This approach is based on applying a windowing function, generated by a least square approximation. The performance of a fractional delay filter obtained by truncating the sinc solution is usually not acceptable in practice. A well-known method to reduce the Gibbs phenomenon is to use window functions for time-domain weighting. An example of this effect is shown in the following figure. Instead of truncating the impulse response, a bell-shaped window can be used which puts more emphasis on the middle values of the impulse response

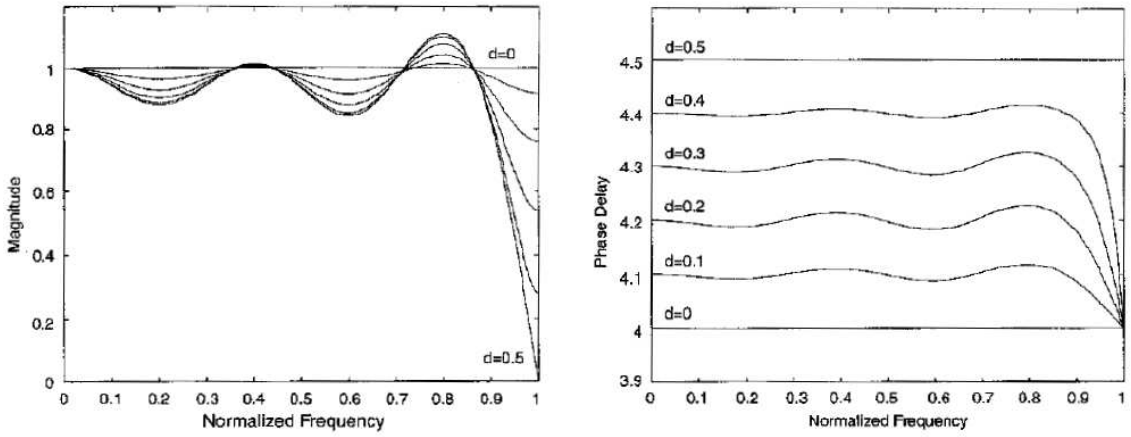


Figure 3.19: truncated ideal response; L=10. (figure from [60])

and reduces the peak magnitude error at the cost of a wider transition band of the filter. The new windowed impulse response is obtained as

$$h(n) = W(n-D)sinc(n-D) \quad for\ 0 \leq n \leq N;\ 0\ otherwise \tag{3.3.13}$$

where the ideal impulse response hid(n) is truncated and shaped by multiplying by W(n-D), which is a length-L (=N+1) window sequence shifted by the appropriate delay value, D. Figure 3.20 shows an example of the resulting transfer function of the filter designed using this technique. Many windows have been proposed; the

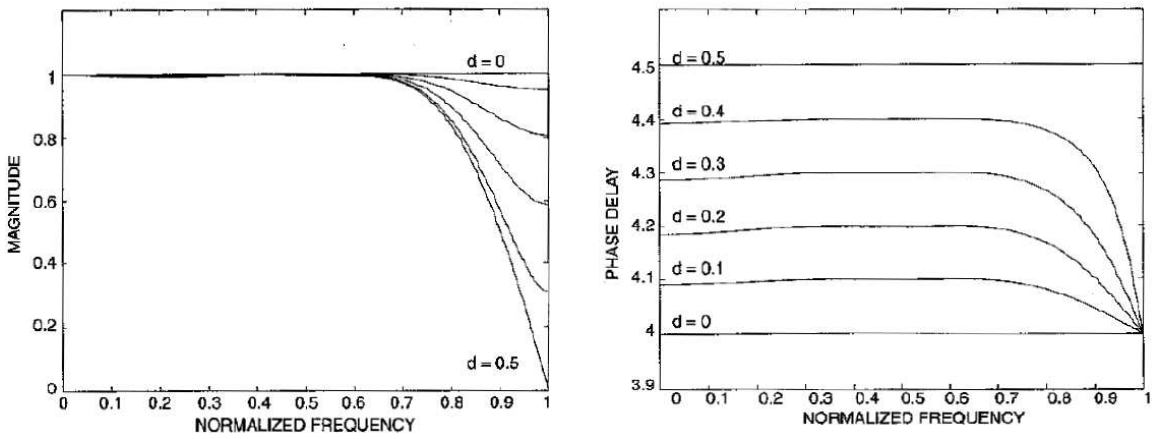


Figure 3.20: ideal response windowed with Dolph-Chebyshev window with 40dB sidelobe ripple for L=10. (figure from [60])

Kaiser window allows the control of the peak ripple using one parameter, while the Dolph-Chebyshev window is optimal in the sense that it has the smallest possible

peak side-lobe ripple, which is given as a parameter. In principle, window-based design is fast and easy. However, in practice it is somewhat difficult to control the magnitude error by adjusting window parameters-this is particularly true for very short filter lengths (L less than 10), which are typical in many applications. However, if the detailed error behavior is not critical, the method is quite suitable for real-time coefficient update. One can either store the window in memory and compute the values of the sinc function on-line or, for example, store enough samples of the windowed sinc function in the memory and compute the filter coefficients using interpolation.

Maximally flat FIR (Lagrange interpolation)

Here a new philosophy has been proposed. Instead of minimizing a least squared error measure like the previous case, the error function can be made maximally flat at a certain frequency, typically at w0=0, so that the approximation is at its best close to this frequency. This means that the derivatives of the frequency-domain error function are set to zero at this point, that is:

$$\frac{d^n E(e^{jw})}{dw^n} = 0 \quad @w = w_0 \quad for\ n = 0, 1, 2, ...N \tag{3.3.14}$$

As discussed in [60], the solution is equal to the classical Lagrange interpolation formula, where the coefficients are obtained by fitting the interpolating polynomial to pass through a given set of data values. The solution can be given in an explicit form as:

$$h(n) = \prod_{K=0;K\neq n} \frac{D-k}{n-k} \quad for\ n = 0, 1, 2, ...N \tag{3.3.15}$$

where N is the order of the filter. Lagrange interpolation has several advantages: easy explicit formulas for the coefficients, very good response at low frequencies, and a smooth magnitude response. The maximum of the magnitude response never exceeds unity when the delay is near to half the filter length. This is important in applications using feedback. On the other hand, the approximation error is often unnecessarily small at low frequencies, at the cost of the performance at higher frequencies. Fig. 3.21 shows an example of the resulting transfer function of the filter designed using this technique. Figure 3.22 shows the performance of this algorithm in terms of SFDR improvement for different value of the input signal

(Rs= Fsampling/Fsignal) and different filter order (pt interpolation).

There are two common techniques for generating variable delays: a variable FIR filter and using a Farrow Structure. These two techniques are discussed in further detail in the following sections.

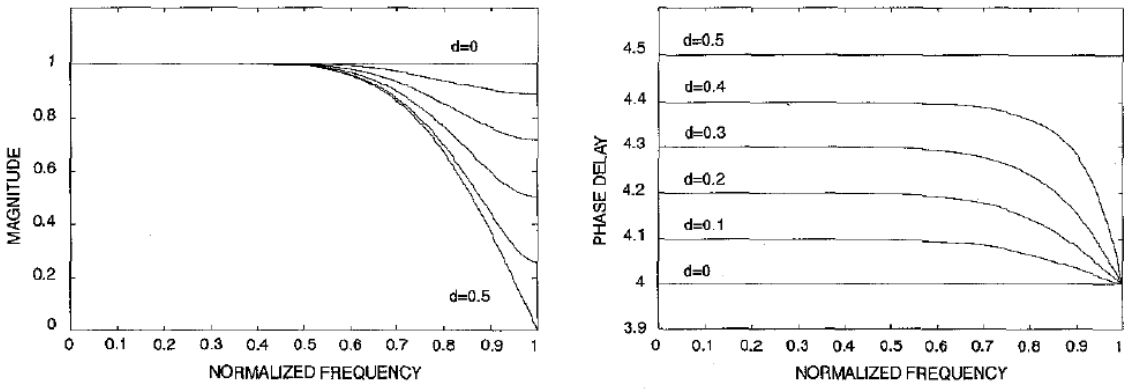


Figure 3.21: maximally flat FIR design magnitude and phase frequency response for L=10. (figure from [60])

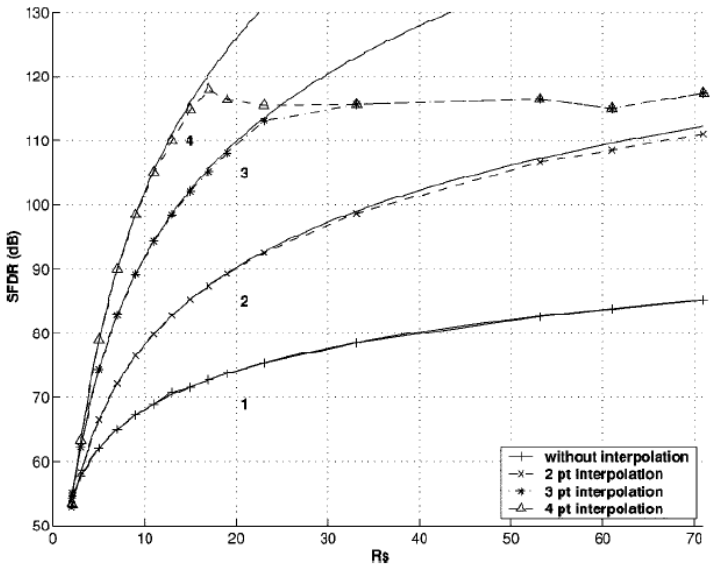


Figure 3.22: analytical and simulated results of Lagrange reconstruction method performance. (figure from [62])

1^{st} method: Fast delay control FIR filters In the previous cases, we have assumed that the fractional delay filter is designed exclusively for producing the desired delay. However, in time-critical applications this may be impossible since an additional

FIR filter always introduces some net delay, as discussed earlier. Instead, it may be more advantageous to control the delay of a FIR filter that is already included in the system. As the FIR filter may be adaptive or variable or for other reasons its coefficients may not be known, it is sometimes essential that the delay control algorithm be independent from the filter coefficients.

It has been shown that the coefficients for the FIR filter can be derived using the Fourier Transform [60]. It was shown that if hp(k) is the impulse response of a generalized FIR filter, the impulse response h(k) of the same filter with a phase response delayed by D is:

$$h(n) = \sum_{k=0}^{N} h_p(K) sinc(n - K - D) \tag{3.3.16}$$

which is seen to be a weighted sum of sinc functions with an infinitely long impulse response and consequently with truncation problems. This can be alleviated in the usual manner by employing a suitable time-domain window to truncate h(n).

2^{nd} method:

Farrow Structure A promising technique for efficient implementation of a continuously variable delay element was proposed by Farrow [61]. The basic idea is to design a set of filters approximating a fractional delay in the desired range (e.g. $0 \leq d \leq 1$) and then to approximate each coefficient as a Pth-order polynomial of d :

$$h_d(n) = \sum_{m=0}^{p} C_m(n) d^m \quad for\ n = 0, 1, 2, ...N \tag{3.3.17}$$

where $c_m(n)$ are real-valued approximating coefficients. The subscript d is now included to emphasize that each coefficient is a function of the fractional delay, d. In [60] it is shown that the transfer function of the filter can be elaborated into the form :

$$h_d(z) = \sum_{m=0}^{p} C_m(z) d^m \tag{3.3.18}$$

where it was defined

$$C_m(z) = \sum_{n=0}^{N} C_m(n) z^{-n} \tag{3.3.19}$$

The form of eq. 3.3.18 immediately suggests an efficient implementation as a parallel connection of fixed filters with output taps weighted by an appropriate power of d

(figure 3.23). The polynomial approach can readily be generalized for other filter

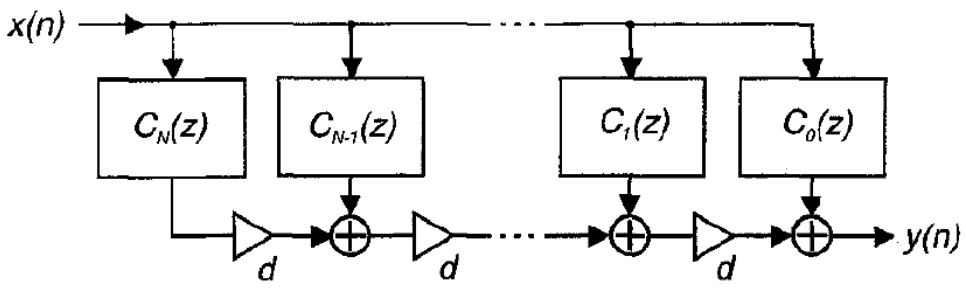


Figure 3.23: farrow structure for implementation of polynomial approximation of filter coefficients. (figure from [60])

design techniques and used in addition to previous methods.

Choosing the right method

We have explored only a few of the key techniques to implement fractional delay FIR filters. There is a plethora of methods available for engineers willing to design fractional delay filters, and one may be confused about which one to pick. The guideline among all these techniques is the trade-off between computational complexity and filter approximation error between the filter itself and the ideal one. To help the reader to understand this, methods have been described starting from the ideal one but most computationally heavy and arriving to the last one: very easy algorithm but not ideal in terms of phase and magnitude response. The last technique in fact offers the advantage of an easy algorithm to change the delay of the filter, but it designs a filter that is an approximation of a filter approximating the ideal one. The Lagrange interpolation method is a compromise; every time someone wants to change the delay of the filter, the filter must be re-designed and FIR filter coefficient must be recalculated. However the procedure is recursive and not particularly computationally complex. An example of system using this method is analyzed in [62]. Despite the advantages of variable delays for correcting mismatches, windowing methods are also very used where complexity needs to be minimised [i.e. [64]].

3.4 Mismatch shaping

The main problem with identification and correction techniques previously analyzed is that the quality of digital compensation algorithms depends on the quality of the timing mismatch identification. Therefore, even when we use an ideal compensation algorithms, the performance is only as good as the identification method. Since no identification technique is perfect and usually to achieve good performance, we need computationally complex algorithms, mismatches compensation systems should not depend upon the identification of them. A solution has been proposed to solve the raised problem: spectral shaping of mismatches noise. This method is an extension of extension of the sigma-delta principle to mismatches in time interleaved modulators. Spectral shaping of error power means shifting the error power to some input signal free region in the frequency band. In order to guarantee some input signal free band, we oversample the input signal and ensure that the signal is band-limited. The shifted error power then, can be easily filtered out. Techniques and algorithms to implement this new design philosophy are going to be described in the following paragraphs.

3.4.1 Randomization (Zero order shaping)

This technique spreads the mismatches noise power over a wider frequency range by randomizing the order in which the ADCs are used in the interleaved structure. The architecture and the principle of a randomized TIADC are shown in figure 3.24. For each sampling instant a new channel ADC is randomly selected, whereas we have some constraint on the selection process. Given a redundant array of (M + ΔM) channel ADCs, that operates with a minimum channel sampling period of MTs, and wanting to build a randomized TIADC with an overall sampling period of Ts, each time instant a channel among ΔM+1 channel ADCs must be chosen without violating the sampling constraint. This is illustrated in fig. 3.24(b) for M=2 and $\Delta M = 1$. From fig. 3.24(a) can be noted that the TIADC is followed by a low-pass filter, which filters out some portion of the uniformly shaped mismatch spectrum in order to increase the SINAD. The spectrum of the output signal from a

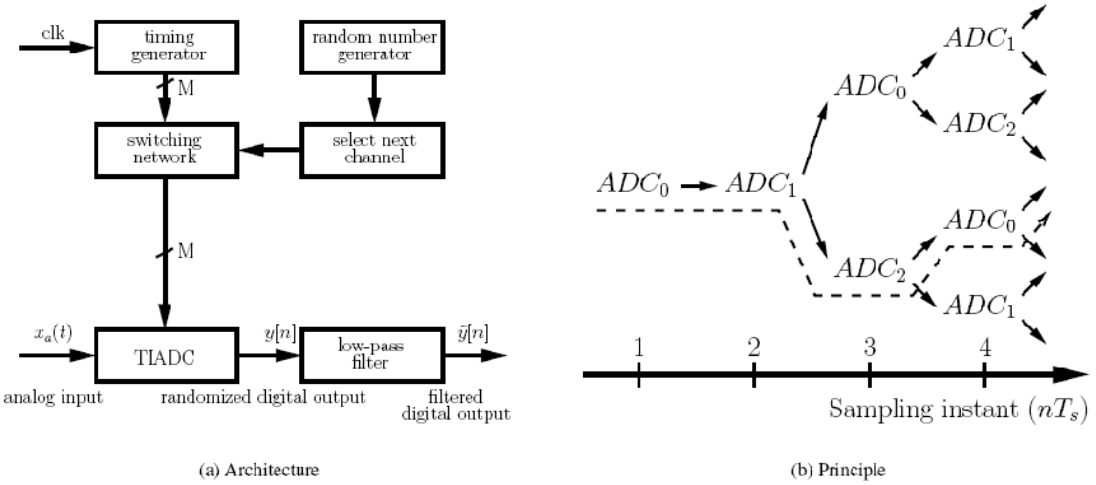


Figure 3.24: concept of randomization technique. (figure from [67])

randomly interleaved ADC system equals to a long and complex expression reported in [65]. A graphical representation of it is shown in figure 3.25. Plot of fig. 3.25

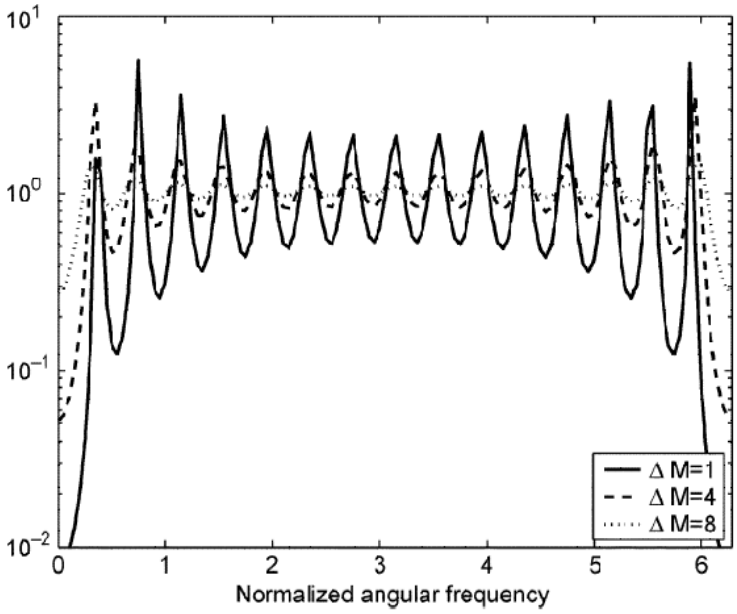


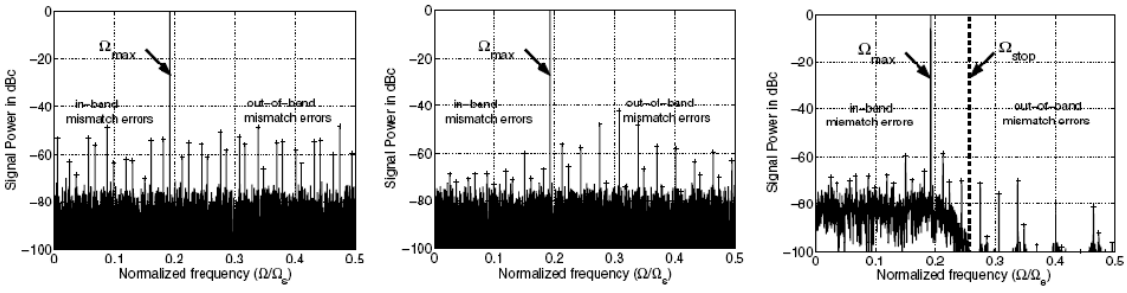
Figure 3.25: mismatch noise spectrum for M=16 and ΔM=1,4,16. (figure from [65])

shows that the oscillations and peak values, decrease when the number of additional ADCs if ΔM(the number of extra ADCs) is increased. This is expected since a higher value of means that there are more ADCs to choose from at each sampling instance, and the errors are then more randomized. When $\Delta M \to \infty$, the mismatch

noise spectrum becomes constant. It's important to note that the total mismatch noise power in a system before and after applying this selection technique is still the same. This is expected since the total amount of distortion cannot be modified by changing the order in which we select the ADCs. However, the shape of the distortion is very different between the fixed interleaved and the randomly interleaved case. In the randomly interleaved case there are no spikes in the output spectrum, so this technique improves only the SFDR and not the SNR of the sampled signal. It's important to note that this technique doesn't require any sort of estimation of mismatches, so it's performance is not affected by non-idealities of estimation algorithms.

3.4.2 Time mismatch noise shaping

According to the analysis made in the previous section, the most perturbing mismatches are timing mismatches. Furthermore they are also difficult to compensate due to the computational complexity of estimation and correction algorithms. Due to this fact, many techniques have been proposed to reduce the impact of these errors. The first proposed technique was randomization. The main problem with it is that only SFDR is improved while SNR is not. A technique has been proposed lately by Vogel to improve both [66]. It aims to shape the mismatch noise power out of the band of interest emphasizing out of band spurs rather than in-band ones. The effect of channel sequence optimization and the required architecture are shown in figure 3.26 and 3.27. The system works as follow: without any optimization



(a) non-optimized (b) optimized (spectrally shaped) (c) optimized and low-pass filtered

Figure 3.26: output spectrum of a TI ADC with 32 channels. (figure from [66])

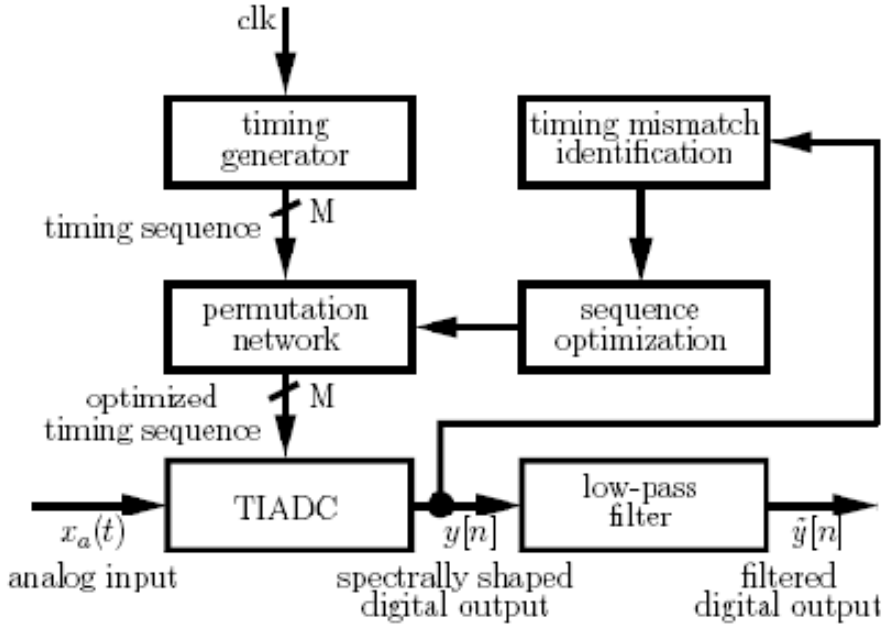


Figure 3.27: timing mismatch error power shaping principle. (figure from [66])

the unfiltered digital output signal appears as in fig. 3.26(a). After the timing mismatches are identified and a new optimized channel sequence is determined (cf. fig. 3.27) the output becomes spectrally shaped, as shown in fig. 3.26(b). The spectrally shaped output spectrum can easily be filtered, which results in the spectrum shown in fig. 3.26(c). While this approach is not new, finding an optimal sequence which maximizes the power of out of band mismatch noise spurs is a difficult combinatorial optimization problem. It can be solved by using different methods such as simulated annealing and genetic algorithms. These optimization methods are computationally complex and are difficult to be implemented on a chip. Furthermore, there is no guarantee for finding the global maximum with such algorithms within a given time span. The algorithm proposed by Vogel in [66], has low complexity and can always determine, at least, a good suboptimal solution. The basic idea in spectral shaping is to focus the mismatch error power onto the highest frequency components. In [66] it is shown that its magnitude is proportional to the difference of the sum of all even indexed timing mismatches and the sum of all odd indexed timing mismatches. Therefore choosing a conversion sequence where this condition is true achieves the desired goal. The next step to increase the spectral shaping is to

maximize the spurious closest to the central one. Going on this way the power of all out of band spurs is maximized. In [66] an algorithm implementing a good suboptimal solution is proposed. This algorithm allows an improvement of SFDR and also of the SNR. One drawback is that time identification is required but requirements are much more relaxed compared with estimation&correction techniques. The purpose in fact of the identification is not to correct for timing mismatches but just to understand the relative order of them. For this reason, the performance of this technique has little dependence on the resolution of the timing mismatch estimation algorithms. Fig. 3.28 show performances of this technique.

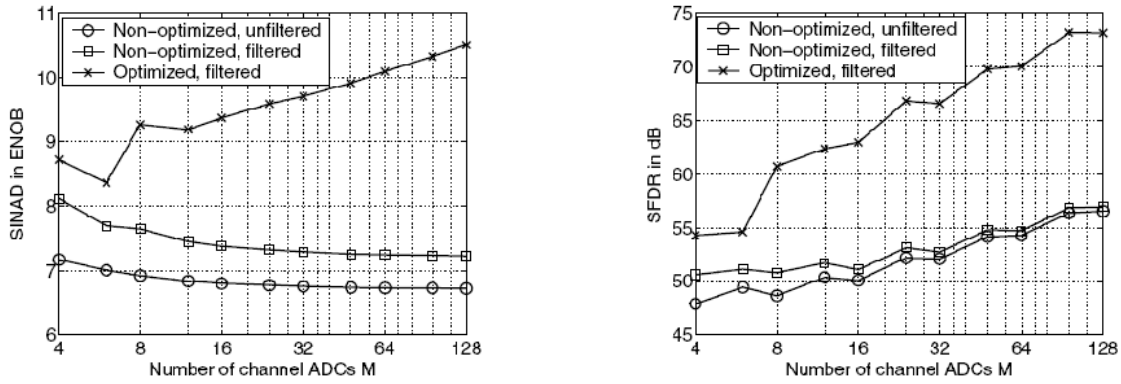


Figure 3.28: performance of this technique in relation to a timing mismatch standard deviation of 0.01Ts. (figure from [66])

3.4.3 Time mismatch in-band noise randomization

The technique described in [67] by Vogel is a variation of the one proposed by the same author in [66] and described in the previous paragraph. As explained before, the aim of both methods is to maximize the magnitude of the highest out of band spurs expressed by the following equation.

$$\alpha(j\Omega) = -\frac{j\Omega}{N}\sum_{n=0}^{\frac{N}{2}-1}(\Delta tv_{2n} - \Delta tv_{2n+1}) \tag{3.4.20}$$

where N is the number of ADCs and Δtv_{2n} is the time error mismatch of the ADC selected at the sample number 2n. Looking at this equation, in order to maximize $\alpha(j\Omega)$, in [67] the following method has been proposed. By assuring that the time

error of every ADCs selected on an even sampling instant is always higher or always lower than the ones selected on an odd sampling instant,eq. 3.4.20 is maximized. However as there is more than one way to generate a selection sequence where the previous statement is true, it is possible to use these degrees of freedom to achieve further improvement. To exploit this, all ADCs should be split into two groups A and B. The first group contains all ADCs with positive timing mismatches and to the other, the ADCS with negative timing mismatches. The zero reference is chosen such as the size of each group is N/2. Substituting all these considerations into eq. 3.4.20 the following equation is obtained :

$$\alpha(j\Omega) = -\frac{j\Omega}{N}\sum_{n=0}^{\frac{N}{2}-1}(\Delta t_m^A - \Delta t_m^B) \tag{3.4.21}$$

From previous equation can be noted that, randomization can be applied in order to further improve the performance of the system while shaping. Picture of fig. 3.29 shows the performance of this kind of system where both randomization and spectral shaping of mismatch noise work together. In fig. 3.29(a) the output spectrum of a TIADC (M(number of ADCs working at the same time)=6 , R(extra ADCs to allow randomized choice selection)=2, N(n. of ADCs)=8) with timing mismatches is shown. Additional aliased spectra of the input signal noticeably distort the whole spectrum. In fig. 3.29(b) conventional randomization to distribute the mismatch power has been used. A low-pass filter at Ωs/4 can reduce the SINAD at most by 3dB. Finally, in fig. 3.29(c) randomization and spectral shaping of timing mismatches have been combined together as suggested in [67]. Comparing the last two figures a decrease in the noise floor can be noted. With a simple low-pass filter the spectrally shaped mismatch power can be filtered out and increase the SINAD and the SFDR compared to fig. 3.29(b).

3.5 Summary

This chapter makes a comparison between correction techniques for distortion in time-interleaved systems. Section 1 gives an overview of the state of the art techniques and classifies them into three main areas: prevention, identification&correction

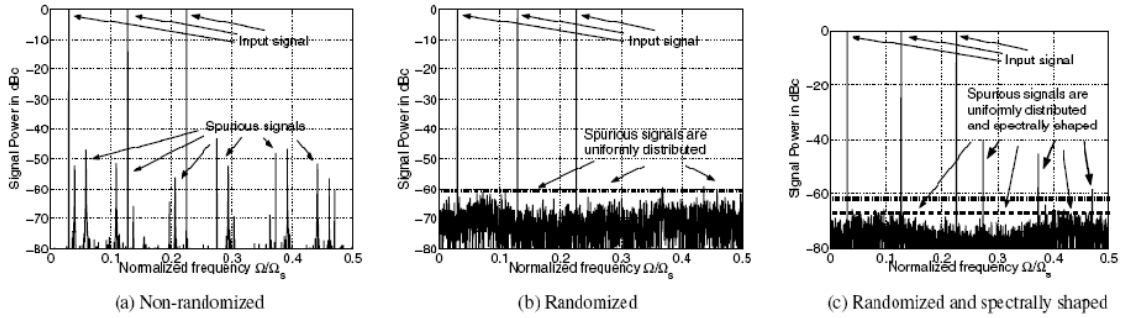


Figure 3.29: effect of spectrally shaped randomization. (figure from [67])

and shaping. Sections from 3.2 to 3.4 make a detailed analysis of these methods.

To make a comparison of all these methods on the matter of performance is quite difficult because the way they are implemented is highly correlated with many parameters that are not related with the method itself. The performance of the first method for example is highly correlated with the amount of offset error in the system that is correlated with the architectural topology and with the offset correction system, but not with the AFB method itself. Given these considerations, the next paragraph tries to express some points on mismatch correction techniques.

The first method (AFB) has some problems with offset mismatch errors and in addition to that doesn't average mismatch error power over the overall bandwidth. Due to this problem, even a small offset error can undermine the resolution of the overall system.

The efficiency of the second method is highly related to identification techniques that are strongly dependent on the statistical properties of the input signal. Therefore, even when we use an ideal compensation algorithms, the performance is only as good as the identification method. Since no identification technique is perfect and usually to achieve good performance, we need computationally complex algorithms, mismatches compensation systems should not depend upon the identification of them.

The third method seems to be the most challenging one since the performance of this correction system is almost unrelated to mismatch identification techniques. This technique however is not perfect and some issues have been identified in previous section.

Next chapter presents a new solution in order to overcome limitations of this method and states a new state of the art for mismatch noise post-processing correction techniques.

Chapter 4

Proposed Methodology for Reducing Timing, Gain and Offset Mismatch Distortion

In the previous chapter an overview was given of many of the state of the art techniques to solve problems related to mismatches between ADCs in time interleaved systems. A new technique is going to be proposed in this chapter. The first section of this chapter describes the new mismatch correction system. Simulation results are going to be presented in section two in order to show the performance of this new proposal in comparison with existing channel sequencing schemes. All software has been developed using MATLAB® environment and are provided in appendix B. The most important performance parameters like SFDR and SNR are reported for system composed by ADCs ranging from 14 to 18 bits. The performance of the proposed methodology will be compared with existing techniques from a number of different perspectives.

4.1 Time mismatches correction

4.1.1 Main idea

Among all proposed systems the spectral shaping of mismatches for high-speed, high-resolution systems seems to be the most promising one. This technique does not require an accurate mismatches identification or computationally complex algorithms. The new technique is a novel selection ordering method for time interleaved analog-to-digital converters (TI ADCs). This method improves the signal to noise and distortion ratio (SINAD) and the spurious free dynamic range (SFDR) by combining randomization of channel ADCs and spectral shaping of timing mismatches. According to the analysis made in the previous chapter, time mismatches most undermine the performance of the system. For this reason, the new technique is focused on timing mismatches, but good performance is obtained also for gain and offset errors. The output spectrum of a non-randomized TI ADC with N channels, offset, gain and timing mismatches Δ_{tm} is given by equations 2.3.3 and 2.3.4. In the

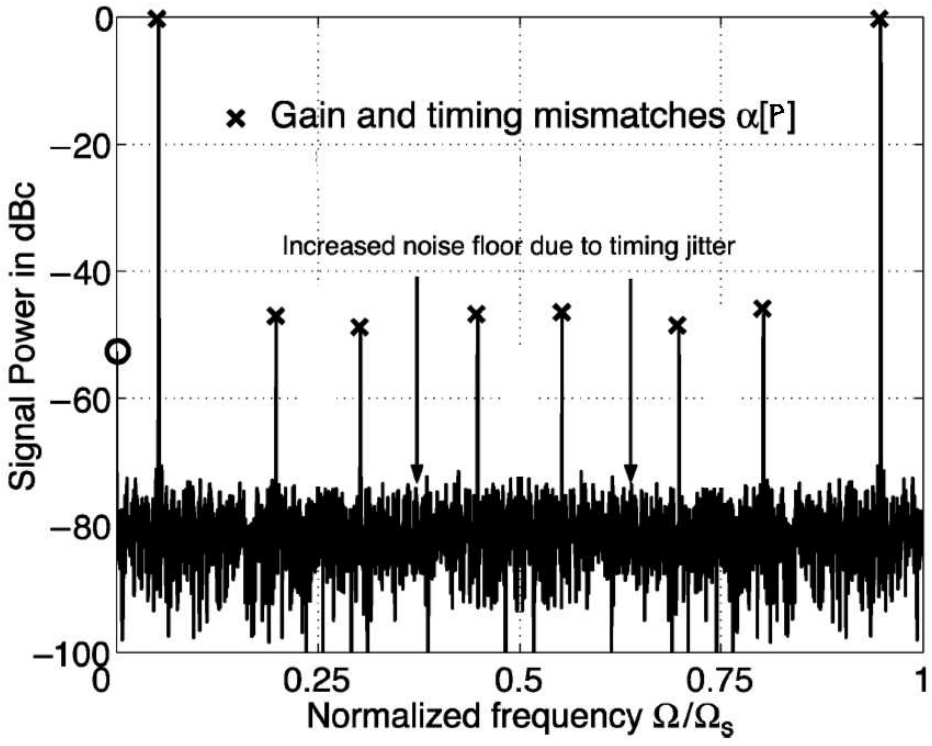


Figure 4.1: example of a spectrum of a time interleaved ADC with only time mismatches.

case of only time mismatches, $g_l = 1$, $o_l = 0$ and previous equations become equal to:

$$Y(j\Omega) = \frac{2\pi}{T_s} \sum_{p=-\infty}^{\infty} \alpha[p]\delta(\Omega - \Omega_0 - p\frac{\Omega_s}{M}) - \alpha^*[M-p] \cdot \\ \cdot\delta(\Omega + \Omega_0 - p\frac{\Omega_s}{M}) \tag{4.1.1}$$

where

$$x(t) = Asin(\Omega_0 t)$$
$$\Omega_s = 2\pi/T_s$$
$$\alpha[p] = \frac{A}{j2M} \sum_{l=0}^{M-1} e^{-j\Omega_0 \Delta t_l} e^{-jpl\frac{2\pi}{M}} \tag{4.1.2}$$

According to previous equations timing mismatches cause weighted aliased spectra of the input signal. Fig. 4.1 gives a graphical representation of previously described equations in the case of a four channels ADC. The weight is given by $\alpha[p]$ while the position by $\pm\Omega_0 + \Omega_s\frac{p}{N}$. The variable p ranges from 0 to N-1 and identifies the order of distortion spurs. It has been shown in previous section that through an appropriate selection order for the channel ADCs, these spectral spurs can be minimized. Random selection attempts to provide a uniform spectral density. However non-random channel selection sequences have shown to provide superior performance.

This section will describe an approach for the development of highly effective selection sequences. In this approach, the objective is to minimize $\alpha[p]$ terms corresponding to in-band frequency spurs, moving the power to out-of-band frequencies that may be subsequently filtered. In order to describe any permutation of the channel ADC sequence, a mapping between the conventional index ordering $< 0, 1, 2, ..., M-1 >$ and the reordered sequence must be defined. To this end, the set of possible index values is defined as {0,1, . . . ,M-1} from where we can choose permutations $v =< v_m >_{m=0}^{M-1}$. The simplest permutation equals $v =< 0, 1, ..., M-1 >$, whereby the conventional sequence of channel ADCs, i.e., $ADC_0, ADC_1, ..., ADC_{M-1}$, does not change. Another example for a selection order is $v =< M-1, M-2, ..., 0 >$, which reverses the conventional order.

We can mathematically explore the effect of these different selection orders by approximating the magnitude of the timing mismatches spurs with the first two terms of a Taylor series expansion, using the just described vector v:

$$\alpha_p = \delta[p] - \frac{\Omega_0}{2N} \sum_{l=0}^{N-1} \Delta t v_l e^{-jpl\frac{2\pi}{N}} \tag{4.1.3}$$

According to Vogel [66], to minimize the in-band frequency components and transfer the power to out-of-band frequencies, we need to maximize the central coefficients of this expansion. However while this shapes the magnitude of the individual frequency spurs, the spurs do persist, reducing our distortion performance. Randomization is effective at removing spurs and providing a uniform spectral distribution. In [67] it is shown that is possible to take a group of channel ADCs and apply a selection sequence to this group according to your selection sequence, while undertaking randomization within that group of ADCs. However, to date, this technique is limited to two groups.

We propose to develop upon this technique and provide guidance on selection schemes using multiple groups, and taking advantage of this ability for more directed spur-minimization.

4.1.2 Optimality proof of the algorithm with time-mismatches only

To allow randomization and spectral shaping to be undertaken at the same time, the N channel ADCs must be divided into G equal-sized groups. Two distinct selection sequences can then be used to perform both randomization and spectral shaping. As randomization occurs within a group, for the purposes of analysis, the effective mismatch for the group can be considered as the mean of the individual mismatches within that group. In 4.1.3 the number p refers to the magnitude of the tone at frequency $\pm\Omega_0 + \Omega_s \frac{p}{N}$ in the output spectrum. To assist in identifying the rules for optimum groups selection and the subsequent sequence, fig. 4.2 shows a graphical evaluation of 4.1.3 for a system with 12 converters (N), 4 groups (A-D) and a fixed selection sequence (A,B,C,D).

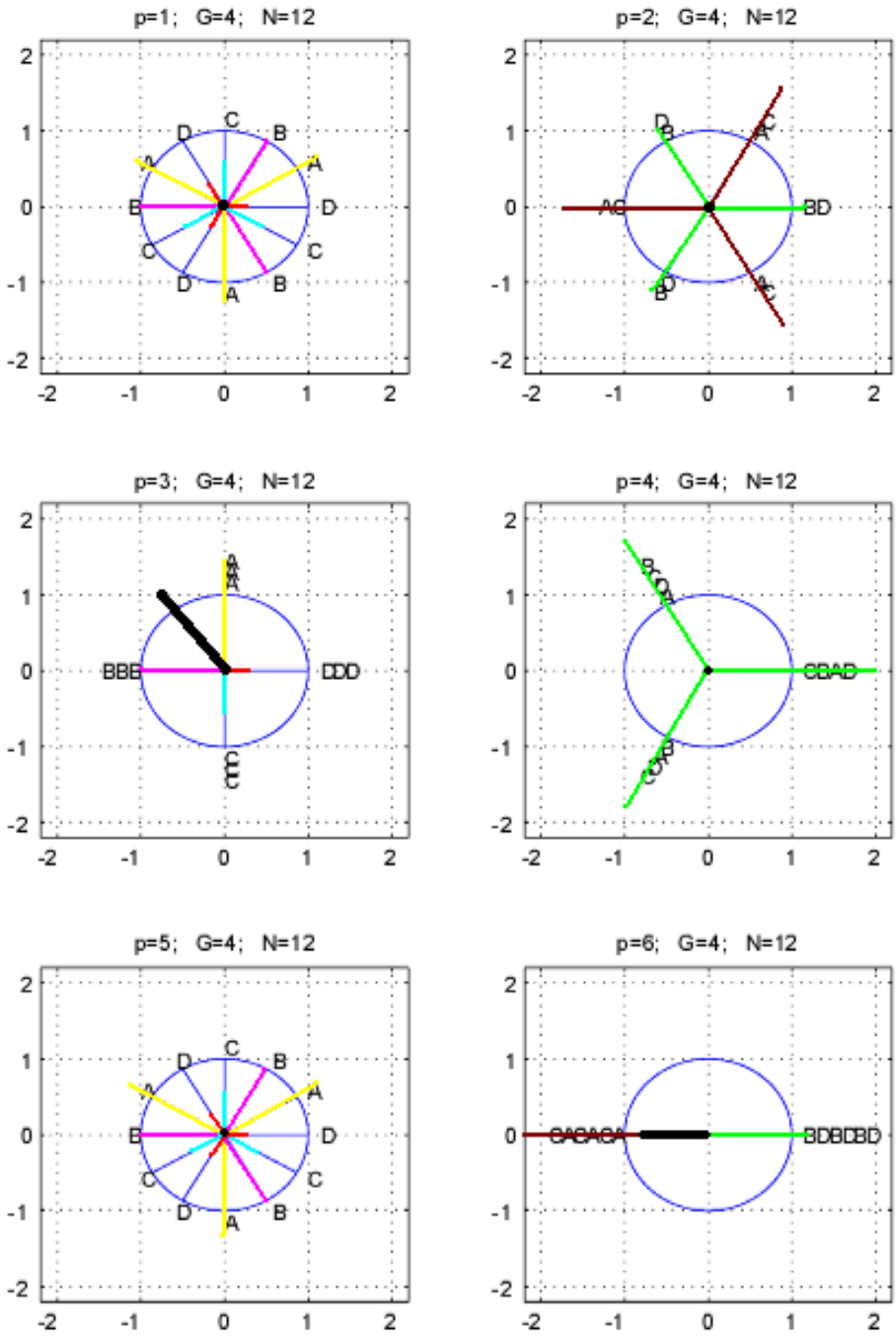


Figure 4.2: graphical representation of 4.1.3 for a system with 12 channel ADCs (N=12) and 4 groups (G=4) (complex numbers, real-imag axes). Vectors are distinguished by color and the black line is the resulting one

Picture 4.2 is very useful to understand how this principle works. There are six plots, one for different values of p. Each vector has a different color. The magnitude

of it is related to the lenght of the colored line. Looking at equation 4.1.3, the magnitude of each vector is proportional to the time mismatch of the corresponding channel ADC. Due to this fact, it is possible to establish a bijective correspondance between the magnitude of each vector(from A to D in the example) and the selection sequence of the time interleaving converter.

The desired target is to push mismatch noise power to higher frequencies maximizing the power of out-of-band spurious tones to be able to both increase sampling speed and resolution. To understand how to do this, the following questions must be answered:
What are grouping criteria that allows the formation of spurs in the output spectrum?
Where should I put the filter cutoff in order to eliminate the most spurious tones ?
How can the grouping criteria maximized in order to maximize out-of-band spurious tones and output signal bandwidth ?.

Looking at the symmetry of the drawings, if vectors A,B,C and D had the same magnitude, no spurious tone would be generated. This is the case of randomization. By properly choosing members of each group, differences in the magnitude of each vector could be obtained. This condition anyway is a necessary one but not a sufficient one. Looking at previous example in fact, even if differences in magnitude between each vector would be allowed, only p=3 and p=6 cases had a resulting vector magnitude different from zero (black line on fig. 4.2). This is due to many factors. First the phase difference between each vector is a constant value. Second the number of the overall rotations made by the N vectors (12 in this example) is an integer number equal to p. Third, even if randomization is performed inside each group, the selection order of them is fixed and all vectors occurs the same number of times equals to N/G. For these reasons, the resulting vector corresponding to most of harmonics equals to zero.

The only condition that added to the previous ones generates spurious tones is the one that the phase amplitude of a complete group selection cycle, where all groups have been selected once is an integer multiple of 2p. The following equation

expresses in an analytical way the previous statement.

$$pG\frac{2\pi}{N} = 2\pi k \tag{4.1.4}$$

where k is a positive integer greater than zero. So, solving for p:

$$p = \frac{N}{G}k \tag{4.1.5}$$

considering that p is bounded from 1 to N/2, consequently reasonable values for k are the ones for whom p assumes values from 1 to N/2. Solving the previous equation for N=12 and G=4, spurs with magnitude different from zero are located at p=3 and p=6. Looking at previous picture it is possible to realize that from these vector configurations a resulting vector may be generated.

In order to maximize these spurs the magnitude of all vectors must be progressively decreasing or increasing with the increasing of the letter value. In this case if $A > B > C > D$ spurious tones that correspond to p=3 and p=6 are maximised as A+C is greater than B+D. This grouping technique is very simple as groups are simply made by ranking converters basing on their relative time mismatches. Figure 4.2 provides a graphical support for this argument

Figure 4.3 shows a typical output spectrum of a TI system using this algorithm, with N=12 converters and G=4 groups. Gain and offset errors are set to zero and time mismatches are randomly generated between ±20% of the overall structure sampling time. It can be noted that only 3^{rd} and 6^{th} order distortion harmonics are present while all the others are just spread over the overall frequency spectrum as quantization noise.

Now, in order to be able to design the system, other important issues must be investigated. The first one is the definition of the stop-band frequency of the digital filter in order to cut spurs and to maximize output signal bandwidth. The second one is related to the optimization in order to maximize the resolution and the signal bandwidth.

The following proof gives an answer to both questions. According to equation 2.3.4 spurious tones generated by time mismatches are located at frequencies $\pm\Omega_0 + \Omega_s\frac{p}{N}$. In order to eliminate this tone the low-pass filter cut-off frequency must be located slightly before the lowest distortion tone. From the previous calculation the

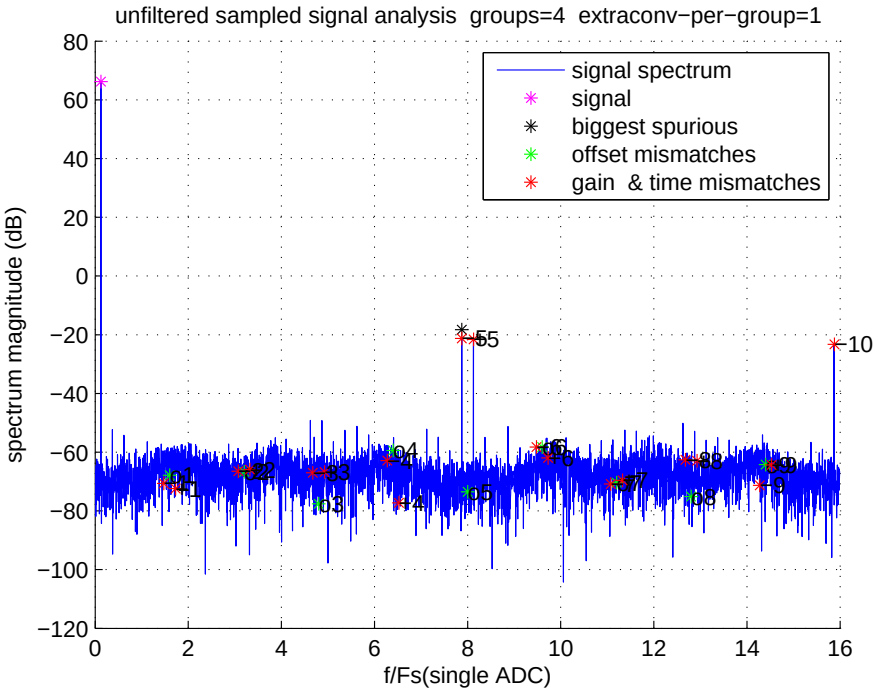


Figure 4.3: output spectrum of the TI system performing the proposed mismatch shaping technique.

lowest value of p is given by 4.1.5 for k=1. Substituting equation 4.1.5 solved for k=1 into equation 2.3.4 (considering only the "-" as a more restrictive case) follows that the highest in-band frequency must be lower than the lowest time-mismatch spurious that is:

$$\Omega_0 \leq \frac{\Omega_s}{2G} \tag{4.1.6}$$

Now it's useful to introduce the over-sampling ratio R defined as the ratio between the sampling frequency and the double of the maximum allowable input signal frequency. The following equation expresses the previous statement.

$$R = \frac{\Omega_s}{2\Omega_0} \tag{4.1.7}$$

Substituting this into 4.1.6:

$$G \leq R \tag{4.1.8}$$

The previous equation states that the number of groups G must be less or at most equal the over-sampling ratio R. In other words, for a fixed number of converters each sampling at a fixed frequency, increasing the number of groups, decreases the

frequency of the first spurious tone causing a reduction of the maximum allowable input frequency. This makes comparisons between systems difficult. In order to compare the performance of this system for fixed group numbers and sampling frequency, simulations based on the following considerations were performed.
If we have N converters all with the same resolution and the same sampling speed and we want to create a system with an overall bandwidth equal to Ni times that of a single converter, the overall number of ADCs N must be equal to:

$$N = N_i R + \Delta_N \tag{4.1.9}$$

where R is the over-sampling ratio of the overall system in relation to its bandwidth Ni. Δ_N is the number of additional ADCs necessary to create randomization in the selection sequence. Using equation 4.1.8 follows that the groups number must be at most equal R. Considering that in this demonstration the number of ADCs used for a fixed bandwidth and groups must be minimized, in equation 4.1.9 R is substituted with G. Thus:

$$N = N_i G + \Delta_N \tag{4.1.10}$$

Defining $\Delta_N = G\Delta_M$ and substituting this into equation 4.1.10, follows that:

$$N = G(N_i + \Delta_M) \tag{4.1.11}$$

where Δ_M stands for the number of extra converters in each group in order to perform the randomization of the selection sequence inside it. Due to this fact, Δ_M must be at least equal to one.
All parameters introduced in the previous derivations to model the number of samples inside a period and the number of converters used are assumed to be integer variables since they both are indivisible elements.
The filter in order to cut out of band mismatches spurious tones is chosen in order to simplify the overall system and reduce its required computational complexity. The filter must also present ripple in the pass-band compatible with the desired system resolution and a stop-band able to attenuate distortion harmonics to a level under the pass-band noise floor. The transition bandwidth is chosen to be as small as possible given a reasonable complexity filter. Since high resolution is required,

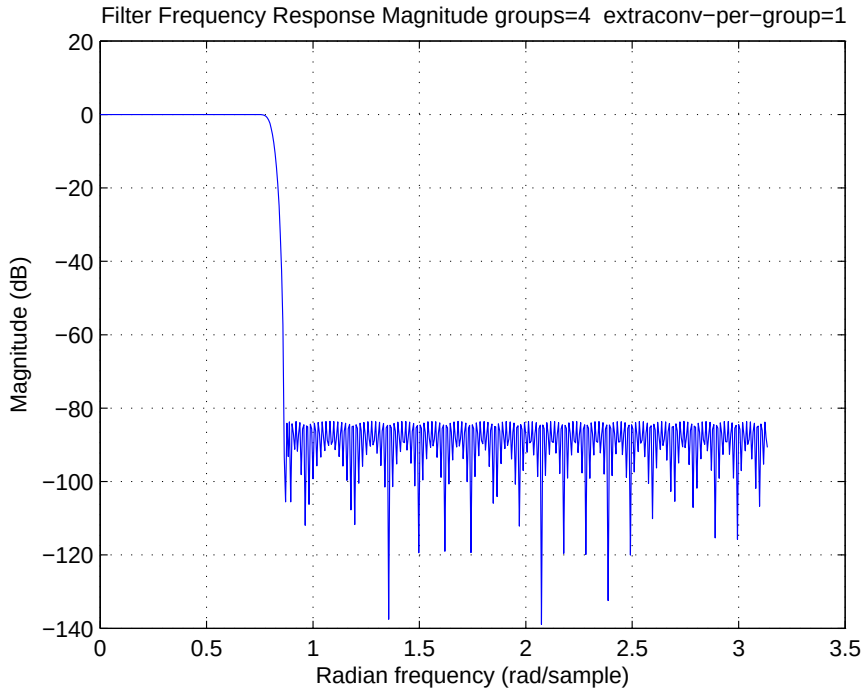


Figure 4.4: equiripple FIR filter with: max pass-band ripple = $10^{-}12$, max stop-band ripple = $10^{-}5$, filter order=641, transition bandwidth = 10% pass-band, cutoff frequency=1/7 filter band.

the filter architecture chosen was a Finite Impulse Response(FIR) structure. FIR architectures in fact presents properties of Linear phase response (useful in many applications of the ADC) and robustness to filter coefficients approximation errors as they are non-recursive. The equiripple (optimum) FIR filter architecture was selected as it most suited our requirements. Picture of figure 4.4 shows an example of the selected filter.

In this section an optimality proof of the algorithm has been given assuming baseband signals. Baseband signals are signals which have a low frequency of 0hz (DC voltage), and extend upwards in the frequency spectrum to a given finite upper bound. The fact that in the presented algorithm the proof is considering only base-band signals is not a limiting assumption, since all limited bandwidth modulated signals can be downconverted into baseband ones.

The only requirement that the signal must have in order to allow the proposed method to work is that it must be limited in band. This way in fact it can be

converted to a baseband one and the previously discussed mathematical process can be applied to it. This restriction doesn't present a serious problem since, also for more practical reasons all signals can be considered as band-limited.
In the case of broadband signals this technique is also feasible since the signal spectrum is still limited. The main difference in fact with baseband ones is that in a baseband transmission, the entire bandwidth is consumed by a single signal. In broadband transmission, signals are sent on multiple frequencies, allowing multiple signals to be sent simultaneously. In order to properly sample a broadband signal with this technique we would only need to oversample more.

4.1.3 Operative description of the algorithm

Summarizing, this grouping technique is simple and requires a simpler level of mismatch identification than existing systems. The proposed system allows control of the size of spurs and the magnitude of their values. For optimal performance the filter cutoff frequency must be located slightly before the first mismatch spur. From a system perspective, the grouping scheme exchanges additional channel ADCS for increased robustness to mismatch. To conclude, the steps needed for the optimum group element selection and sequencing are listed below and shown in fig. 4.5.

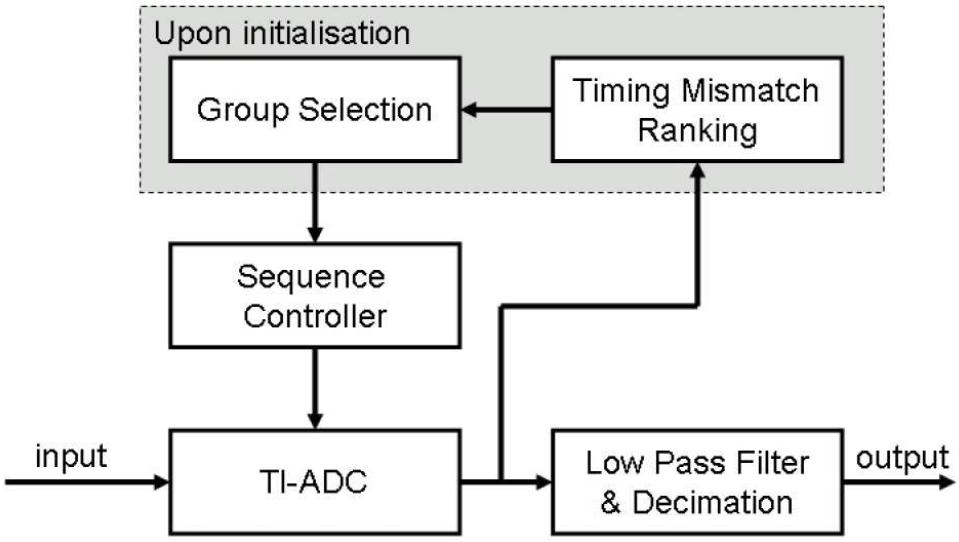


Figure 4.5: Algorithm for proposed channel selection scheme

- Identify and rank the channel ADCs according to their timing mismatch.
- Assign the ADCs to groups according to this ranking.
- Select between the groups in a strict rotation.
- Apply randomization to elements within a group.
- Low-pass filter and decimate the combined ADC output.

4.1.4 Simulation results in the case of time mismatches

In order to simulate the system, a simulation program has been developed (see appendix B1). The proposed technique was tested for a time interleaved system where only time mismatches are present. Gain and offset mismatches are assumed to be zero. The next paragraphs describes these simulations, the output results and compares them with techniques described in the Chapter 3.

Simulations description

These simulations implement the proposed technique for a time interleaved system affected by time error mismatches. The way the overall simulation systems work is described in the following paragraphs.

In this simulation the system is tested for a fixed input signal bandwidth. This value has been kept fixed to Ni times the input bandwidth of a single ADC. The value of Ni for this simulation has been fixed to 4. The resolution of each channel ADC is the same resolution of the overall system. In this case, it equals 14-bits.
As explained before, in order to allow randomization Δ_N extra ADCs must be added to the overall system. The number of extra ADCs, according to 4.1.10 and 4.1.11 must be an integer multiple Δ_M of the number of groups G. This simulation varies these free parameters in order to test them on the performance of the system. The number of groups G is varying from 1 to 8 while the number extra ADCs per group Δ_M is varying from 1 to 4. It's important to notice that when G equals 1 this technique is equivalent to randomization since no grouping is done. When G equals 2 this technique exactly corresponds to Vogel's technique [67] [66]. This way it is

possible to compare the new proposed technique to previous ones.
The parameters of the filter are selected in order that its non-idealities can be considered as negligible compared to the resolution of the system.
The time mismatch between each ADCs is chosen randomly according to a uniform distribution. Its value is given as a percentage of the sampling period of each ADC.

This simulation is an extreme case one for many reasons. First the number of ADCs and so the power of mismatch noise increases with the increasing of the number of groups. Since the total number of ADCs in the system is given by eq.4.1.11, an increase of G corresponds an increase of the number of ADCs N needed by the time interleaved system. Since each ADCs introduces a time mismatch error, the total time mismatch error power of the system increases as well. This is a worst case, since some ADCs are not affected by time error mismatch, or it may be negligible. Second the time mismatch is considered to be a fixed value, independent from the overall architecture. This is conservative as usually the clock delays are better controlled as the clock frequency increases.

A second simulation is also provided in order to show the performance of the system where one assumption is made more close to reality. According to the considerations made during previous sections of this chapter, the more the group number increases, the more the input signal is oversampled. In this simulation, we introduce a new definition of time mismatch. The time mismatch between ADCs here is chosen randomly with a maximum value fixed to a number that is related to the overall structure sampling period and not to the one of the single converter like in the previous case. This way its power is now independent from the number of groups used in the algorithm.

The results of both simulations are going to be described in next subsection.

Analysis of results

The final result of the first simulation described in previous subsection is presented in fig. 4.6. From fig. 4.6 can be noted how this system outperforms previous ones. The SNR and SFDR are progressively improved with the increasing of the number of groups. Following paragraphs are going to make some points about the obtained

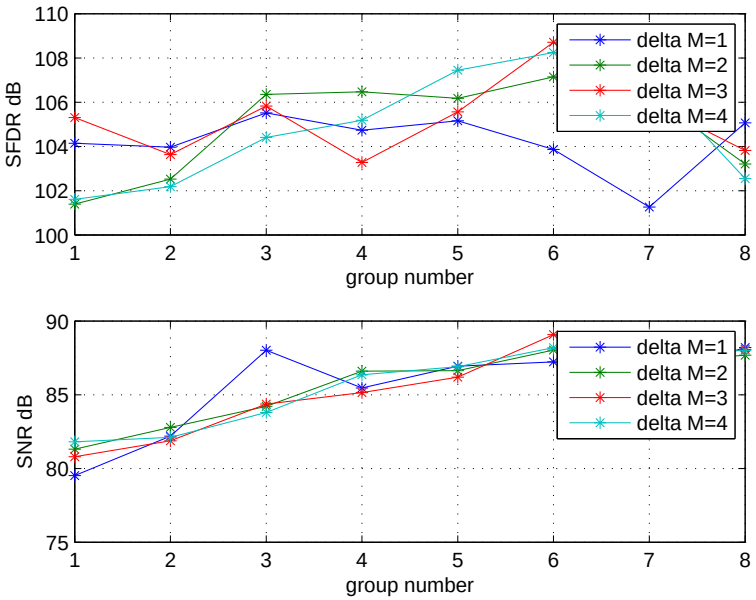


Figure 4.6: 14-bit system; max time error ± 1.5E-3 (referred to a single ADC sampling period Ts). FIR filter with: max pass-band ripple = 1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. Ni=4.

results.

- As the number of groups increases, the noise power in the input bandwidth is lowered.

The reason is due to the fact that, as explained during the mathematical analysis of this method, the more the number of groups increases the more the mismatch noise power is concentrated into high frequency spurs. These spurs are cut by the filter and the remaining noise power is spread over the overall bandwidth due to the random choice of ADCs inside each group.

- How the remaining noise is spread depends on how well the randomization is performed inside elements of each group.

This depends on the number of extra converter per group Δ_M. The higher this number, the better the noise is spread on the overall system output bandwidth. Looking at fig. 4.6 in fact it is possible to note how the line corresponding to $\Delta_M = 1$ presents a rate of change in relation to the groups number that is less regular than

the one corresponding to $\Delta_M = 4$. The standard deviation of the rate of change of performance parameters is shown in fig. 4.7 and 4.8. As can be noted in fact, the

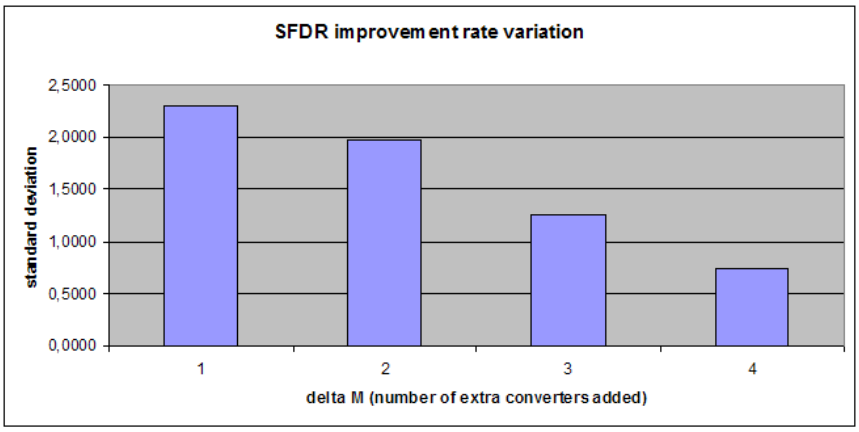


Figure 4.7: standard deviation of the rate of change of SFDR results plotted in fig. 4.6.

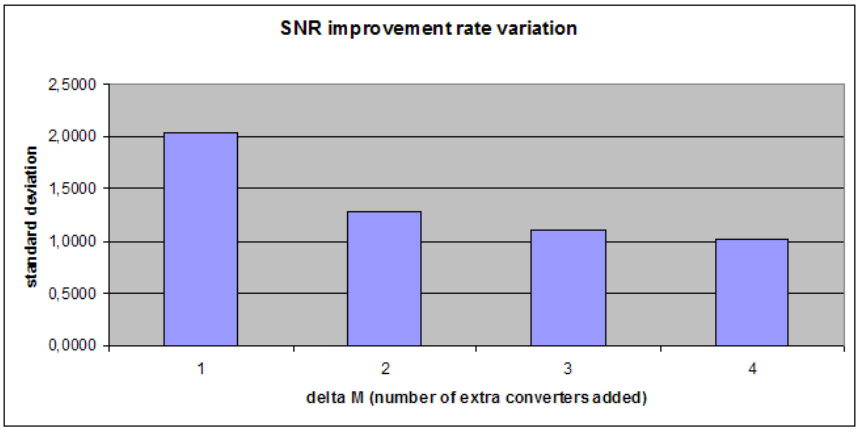


Figure 4.8: standard deviation of the rate of change of SNR results plotted in fig. 4.6.

standard variation of the rate of change of the improvement in SNR and SFDR over the growing of the number of groups is decreasing with the increasing of Δ_M.

Despite the worsening assumptions related to the way the mismatch error is defined, the rate of change of SNR and SFDR versus group number is still increasing. Concluding even from a worst case point of view both SNR and SFDR are improving with the increasing of groups number.

The following results uses the alternative means of describing timing mismatch. From fig. 4.9 can be noted how the rate of change of SNR with the increasing of

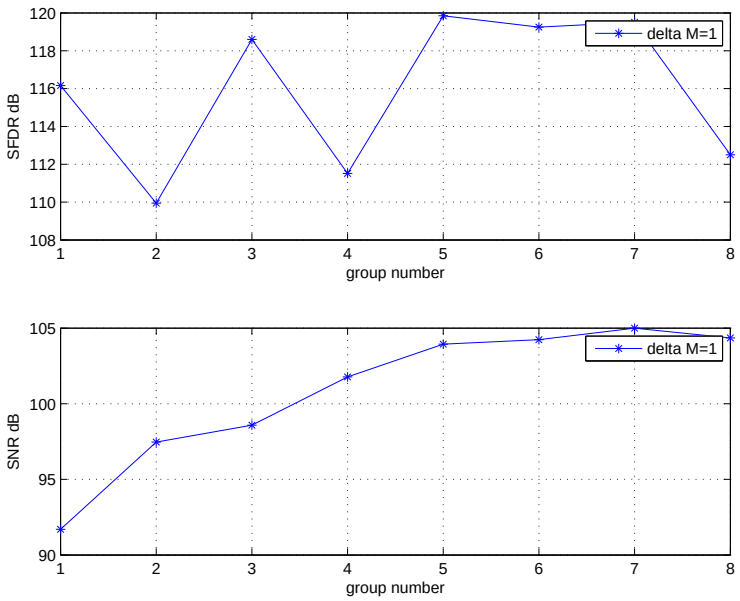


Figure 4.9: 16-bit system; max time error ± 2.5E-3 (referred to Ts of the overall architecture). FIR filter with: max pass-band ripple = 1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. Ni=4.

groups number is higher compared with the one of the previous simulation.

The rate of change of the SFDR in this simulations seems to be very irregular. The reason is due to the fact that here only one extra ADC per group has been used and so the variance of the improvement of performance parameters over the number of groups used is high. However, interpolating the obtained points using a least mean square straight line approximation shows that its rate of change is positive.

Comparison of results

Figures 4.10 and 4.11 shows the performance in relation to 16 and 18 bit systems.

As can be noted from previous graphs, while the rate of change of SNR is quite constant, the SFDR one is quite irregular due to $\Delta_M = 1$ as explained before. Furthermore, looking at the 18-bit example, to achieve 18 ENOB, a SNR of 110.12dB must be achieved. Picture 4.11 shows how with +-7.5E-5 of deterministic time

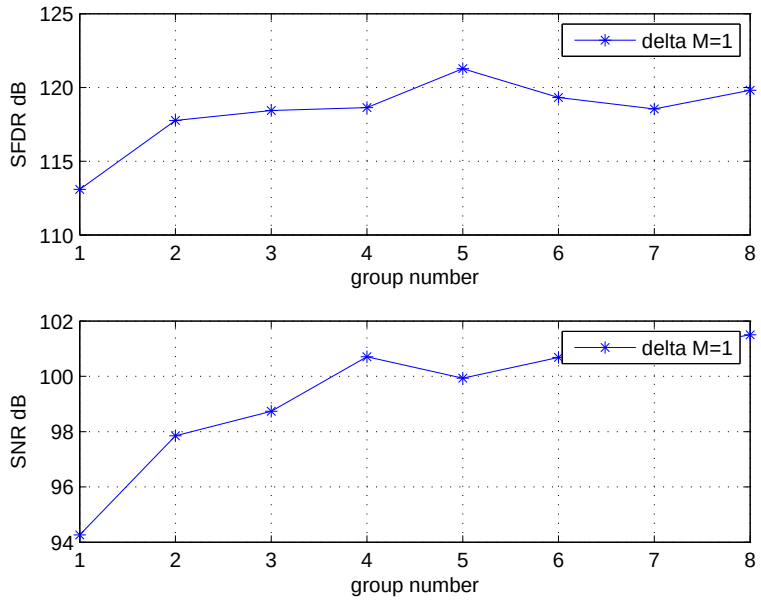


Figure 4.10: 14-bit system; max time error ± 3E-4 (referred to a single ADC sampling period Ts). FIR filter with: max pass-band ripple = 1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. Ni=4.

uncertainty between ADCs (referred to the channel ADCs sampling period), a randomization technique (group number=1) is not able to achieve the desired goal, while the proposed technique with $G > 3$ does. Table 4.1 shows SNR improvements of fig. 4.6,4.10 and 4.11 in relation to the randomization technique.

The only drawback in doing this is that the higher the number of groups, the greater the required accuracy of the time mismatch estimation algorithm, as we must divide the ADCs accurately into the various groups. However as we are still seeking only relative rankings, this remains a significantly simpler problem than experienced by other correction schemes.

The behavior of the system in relation to signals with an input frequency higher than the one of a single ADC is reported in fig. 4.12 and 4.13. As can be noted this technique shows also good results for this kind of signals.

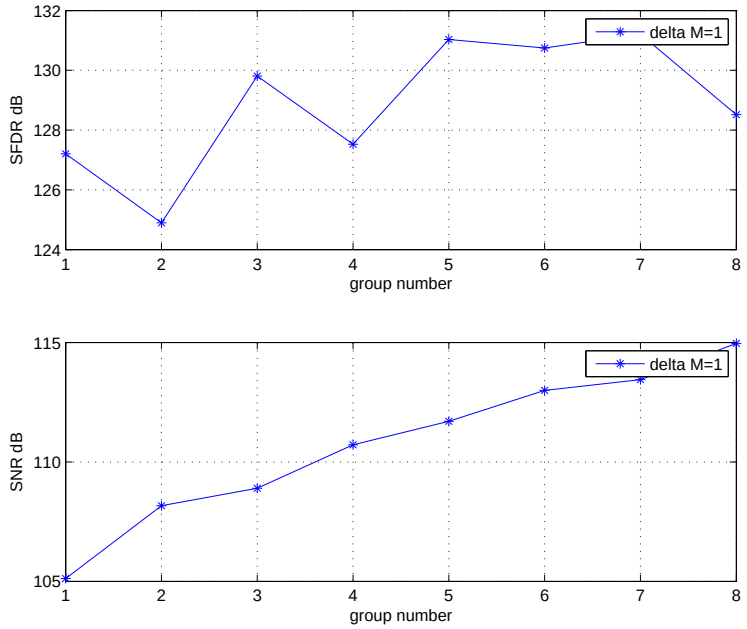


Figure 4.11: 14-bit system; max time error ± 7.5E-5 (referred to a single ADC sampling period Ts). FIR filter with: max pass-band ripple = 1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. Ni=4.

	SNR improvement		
Number of Groups	**14-bit**	**16-bit**	**18-bit**
1:randomization	0 dB	0 dB	0 dB
2	0,3 dB	3,6 dB	3,1 dB
3	2,0 dB	4,5 dB	3,8 dB
4	4,6 dB	6,3 dB	5,6 dB
5	5,1 dB	6,4 dB	6,6 dB
6	6,4 dB	6,5 dB	7,9 dB
7	6,3 dB	6,6 dB	8,3 dB
8	6,4 dB	7,2 dB	9,8 dB

Table 4.1: table indicating improvement of proposed solution over randomization. Data taken from 14($\Delta M = 4$)-16-18 bit systems of fig. 4.6,4.10 and 4.11

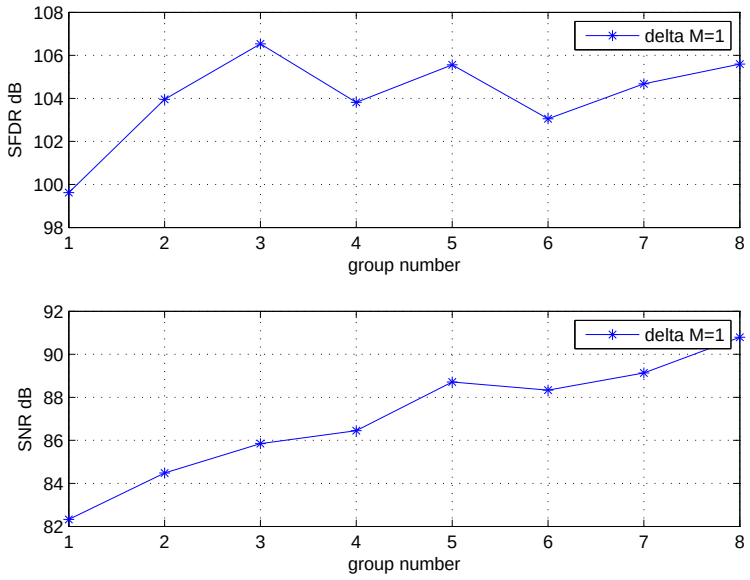


Figure 4.12: 14-bit system; max time error ± 4E-5 (referred to a single ADC sampling period Ts). FIR filter with: max pass-band ripple = 1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. Ni=16.

4.2 Time, gain and offset mismatch correction

4.2.1 Extended optimality proof of the algorithm to gain and offset mismatches

In all previous derivations, gain and offset mismatches have been considered to be not present. Allowing them to be different from zero, what would be the impact of them on the overall performance ?

From equation 2.3.3 and 2.3.4 time, gain and offset spurs magnitude coefficients $\alpha[p]$ and $\beta[p]$ are given by:

$$\alpha[p] = \frac{A}{j2M} \sum_{l=0}^{M-1} g_l e^{-j\Omega_0 \Delta t_l} e^{-jpl\frac{2\pi}{M}}$$
$$\beta[p] = \frac{1}{M} \sum_{l=0}^{M-1} o_l e^{-jpl\frac{2\pi}{M}} \tag{4.2.12}$$

As can be noted, both coefficients are related to mismatches in the same exponential way, only with the difference of a multiplicative factor g_l and o_l. Cause of this, if all the operations made to proof the proposed method are applied at equation

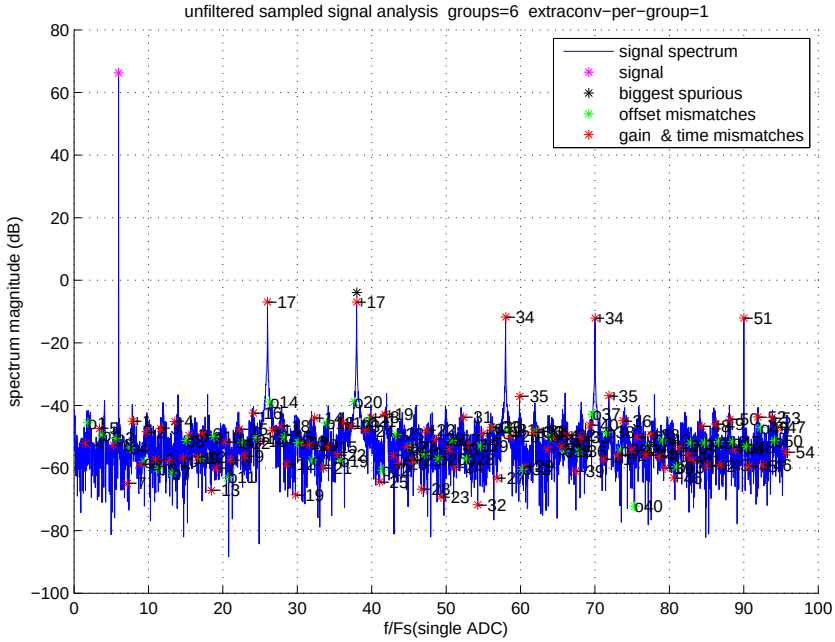


Figure 4.13: 14-bit system; max time error ± 4E-5 (referred to a single ADC sampling period Ts). FIR filter with: max pass-band ripple = 1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. Ni=16. Groups=6.

4.2.12 instead of equation 4.1.3 as did before, it can be proved that the optimality of this method is conserved. Fig. 4.14 shows the unfiltered output of a 16-bit system with gain, time and offset errors while fig. 4.15 shows the same system with only time errors. Comparing both pictures, it is possible to note how fig. 4.14 shows almost the same spurs distribution as in fig. 4.15 without offset and gain errors. The only difference is that in the first figure, between the two gain and time spurs, there is an offset distortion harmonic. This shows the efficacy of this approach to minimizing other mismatch effects. The shaping and subsequent filtering can also remove those spurs arising from the gain and offset mismatches. However, even if the system works perfectly, the achieved resolution (SFDR and SNR) in the first case is lower than the second one. This is due to the fact that more mismatch noise is injected in the system and even with noise-shaping, the in-band noise is higher. This can be easily seen comparing the noise floor of the first and the second figures. In conclusion, the proposed system not only corrects for time error mismatches but also for gain and offset ones.

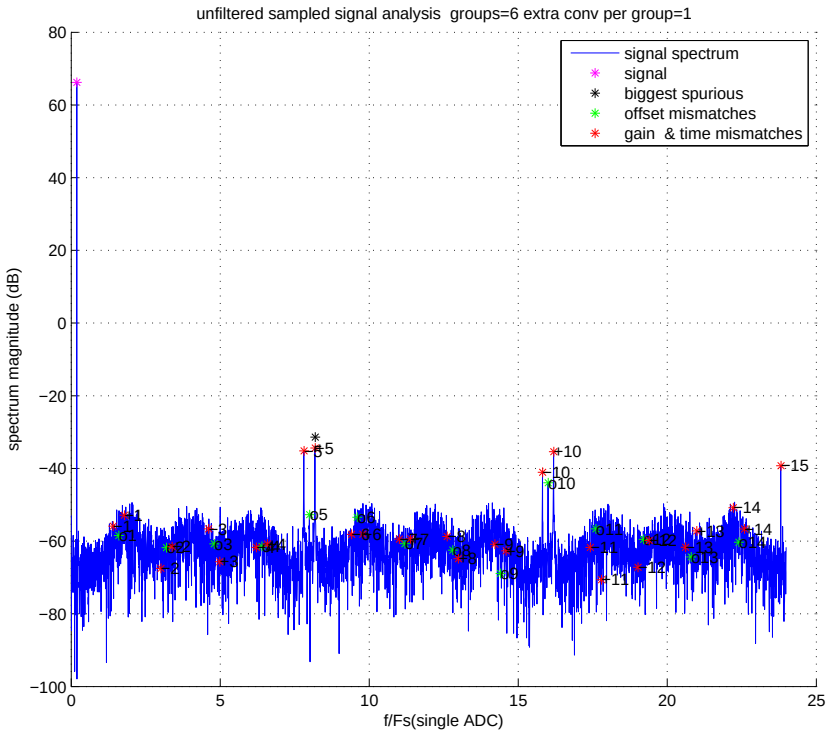


Figure 4.14: 16-bit system, with time, gain and offset mismatches. Proposed method with G=6 groups: signal before the filter.

Many existing systems have problems in correcting for both types of mismatches since the performance of most mismatch characterization algorithms are sensitive to the influence of one mismatch parameter on the detection and quantification of another one. The system proposed in this thesis is not affected by this interdependency as the mismatch estimation systems need only to rank them and not to estimate their actual value. This fact makes this architecture quite independent from the detection algorithms performance and allows a complete correction for all types of mismatches at the same time.

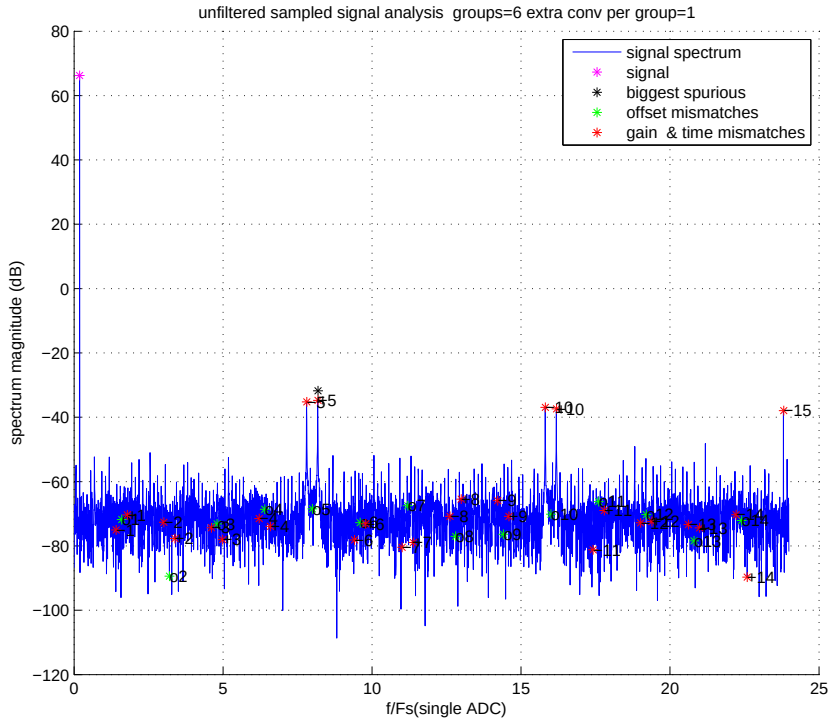


Figure 4.15: 16-bit system, with only time mismatches. Proposed method with G=6 groups: signal before the filter.

4.2.2 Simulation results in case of time, gain and offset mismatches

Simulation description

The following results show the performance of the system with gain and offset error mismatches. The software is reported in appendix B.2. The simulations are similar to the first set of simulations from the previous subsection for the case of time mismatches. The number of groups varies from 1 to 8. The input bandwidth equals Ni times the one of the single channel ADCs with Ni equals 4. The resolution of each channel ADC is the same resolution of the overall system ,14, 16 and 18 bit systems have been simulated. The number Δ_M of extra ADCs per group G equals to 1. This choice has been chosen since we want to minimize the area occupation of the overall system and so all components that are not necessarily needed have been removed from it. The parameters of the filter are selected in order that its non-

idealities can be considered as negligible compared to the resolution of the system. The time mismatch between each ADCs is chosen randomly according to a uniform distribution. It is referred to the sampling period of each ADC in order to test the system under worst case conditions (see simulations made in previous sections for a detailed explanation about this point). Randomly chosen values of gain and offset mismatches has been added to the system.

Analysis and comparison of results

Three figures 4.16,4.17 and 4.18 are presented in order to show how the proposed method works in systems composed by channel ADCs of the same type with respectively 14, 16 and 18 bits.

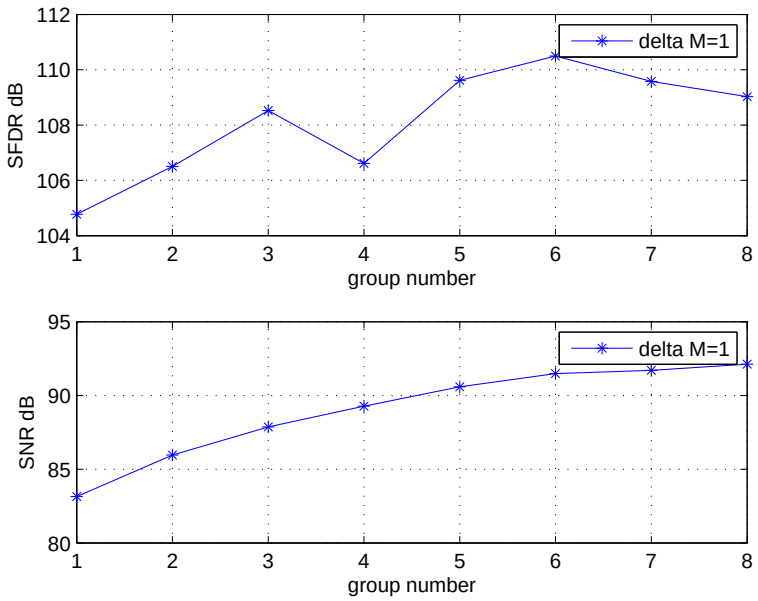


Figure 4.16: 14-bit system; max time error ±3E-4 Ts single ADC, max gain error = ±1E-2 (Full scale), max offset error = ±30%LSB; FIR filter with: max pass-band ripple=1E12, max stop-band ripple=1E-5, transition bandwidth=10% pass-band. Ni=4.

As can be noted from previous graphs, the rate of change of SNR and SFDR is quite irregular due to $\Delta_M = 1$ as explained before. Table 4.2 shows SNR improvements of fig. 4.16,4.17 and 4.18 in relation to the randomization technique.

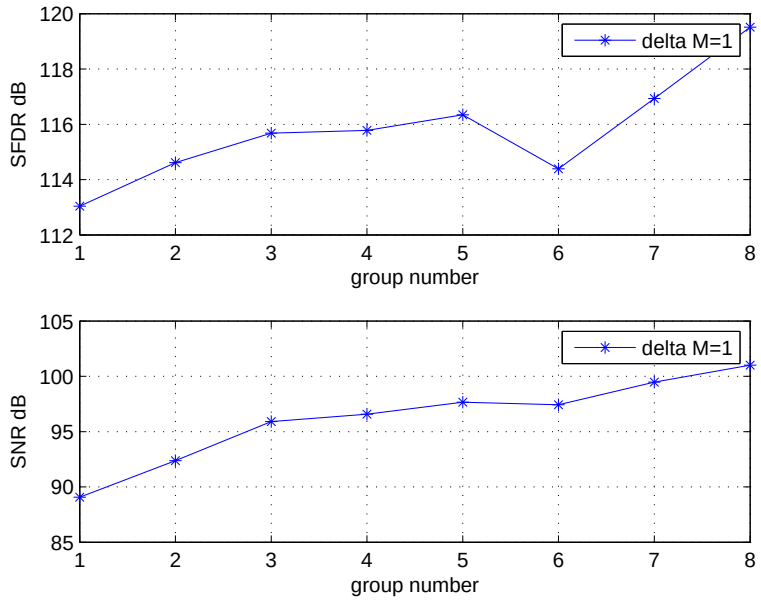


Figure 4.17: 16-bit system; max time error ±7.5E-5 Ts single ADC, max gain error = ±1E-2 (Full scale), max offset error = ±30%LSB; FIR filter with: max pass-band ripple=1E12, max stop-band ripple=1E-5, transition bandwidth=10% pass-band. Ni=4.

It can be noted that the proposed systems outperforms previous ones. The more the number of groups are increased, the more the algorithm is able to correct for mismatch errors.

	SNR improvement		
Number of Groups	**14-bit**	**16-bit**	**18-bit**
1:randomization	0 dB	0 dB	0 dB
2	2,8 dB	3,3 dB	1 dB
3	4,7 dB	6,8 dB	1,1 dB
4	6,1 dB	7,51 dB	3,7 dB
5	7,5 dB	8,6 dB	3,8 dB
6	8,3 dB	8,4 dB	5,1 dB
7	8,6 dB	10,4 dB	5,6 dB
8	9 dB	11,9 dB	5,5 dB

Table 4.2: table indicating improvement of proposed solution over randomization. Data taken from 14-16-18 bit systems of fig. 4.16,4.17 and 4.18

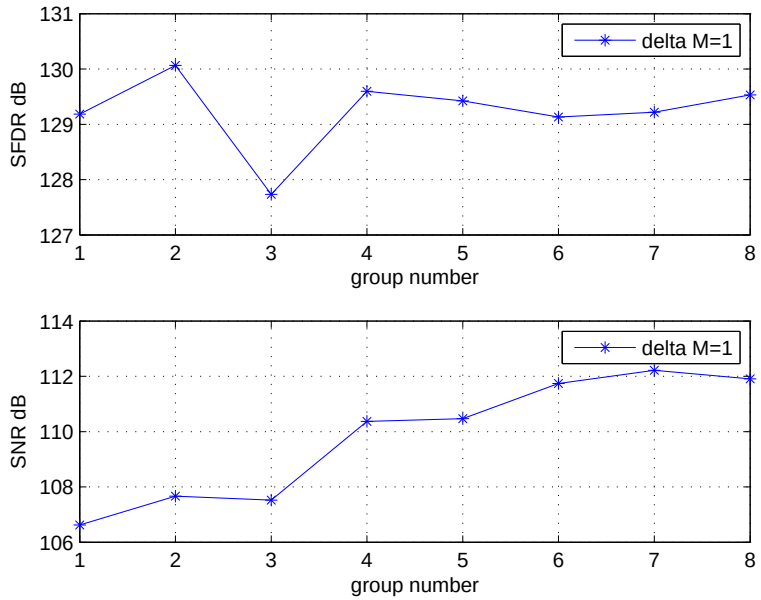


Figure 4.18: 18-bit system; max time error ±3E-5 Ts single ADC, max gain error = ±2E-3 (Full scale), max offset error = ±20%LSB; FIR filter with: max pass-band ripple=1E12, max stop-band ripple=1E-5, transition bandwidth=10% pass-band. Ni=4.

Looking at table 4.2, comparing the 3 simulations, it can be noticed that the rate of improvement changes significantly. This is related to the amount of mismatch error power in the system. Changing it also changes the rate of change of SNR over the group numbers. The more mismatch error power, compared to the final resolution of the system, the better the algorithm works in removing it. This is the main reason why the proposed algorithm seems to work better for the 16-bit system. Looking at figure 4.17, the amount of mismatch noise power is so high that the algorithm needs at least 7 groups in order to reach the desired SNR compatible with the 16-bits specified. In the other examples the minimum number of groups needed is lower, that's why the rate of change of SNR over group number is so high in the 16-bit system.

4.2.3 Simulation results: area occupation point of view

In order to compare different techniques from a different point of view, the following simulation was made. The software is reported in appendix B.3.

Simulation description

This simulation compares the proposed technique to existing ones for a time interleaving architecture having a fixed number of ADCs and a fixed input signal bandwidth. The outcome of this simulation, looking at previous simulations, is not so trivial to forecast. The main limitation is given by the limited number of ADCs that can be used. According to equation 4.1.10, this implies that the higher the number of groups, the lower the number of interleaved ADCs is. In addition, some ADCs are used in order to allow randomization and not to oversample the input signal.

Going more into the detail of the simulation, we note that if the number of ADCs of the overall architecture N is fixed, than, the number of interleaved ADC from eq. 4.1.11 equals to:

$$GN_i = N - G\Delta_M \tag{4.2.13}$$

if we assume $\Delta_M = 1$ and N fixed, GN_i decreases with the increasing of G. If we compare systems with different values of G but with a constant input signal bandwidth, with the increasing of G, the OSR is reduced and so the SNR tend to be worsened. On the other hand, previous results (fig. 4.6-4.9) show that performance parameters, even with worst case assumptions, improve with the increase in group number as more mismatch power noise is spread out of the band of interest. In order to understand the result of this two opposite effects the software reported in appendix B.3 has been developed. The simulations show that the SNR and SFDR of systems having the same number of ADCs and the same input bandwidth for different values of G. The ADCs in this simulation are arranged in order to achieve the highest possible resolution for the given number of groups. The architecture to do this is to oversample the input signal as much as possible and than consider only a little fraction of the overall allowed bandwidth in order to increase SNR. The

input bandwidth of the whole architecture is common for all simulations in order to be able to compare them.

Analysis and comparison of results

Simulations results are reported in picture 4.19 and table 4.3.

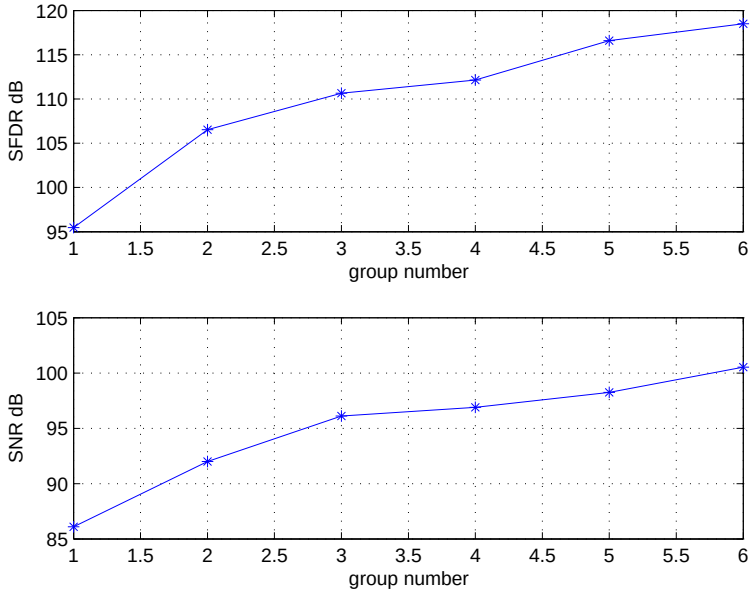


Figure 4.19: 16-bit system; max time error +-1.5E-4 Ts single ADC, max gain error = ±1E-2 (Full scale), max offset error = ±30%LSB. FIR filter with: max pass-band ripple = 1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. N=60.

As can be noted, even from this point of view the best way to arrange a fixed number N of ADCs is to apply the proposed algorithm with as many groups G as possible. Both SFDR and SNR are improved and, with reference to previous simulation, for a random sampling time error deviation of $\pm 1.5E-4$, only the proposed technique with $G > 4$ is able to achieve 16 effective bit of linearity. Table 4.3 provides details on the performance gains achieved in this system. Finally, it can be seen that with this level of timing mismatch, only a configuration using at least four groups can obtain an effective 16 bit resolution. Pictures of fig. 4.20 and 4.21 show a similar behavior in the case of ADCs with 18 and 14 bit of resolution.

Number of Groups	SFDR Improvement	SNR Improvement
1: randomization	0	0
2	+11.1 dB	+ 5.9 dB
3	+15.2 dB	+10.0 dB
4	+16.7 dB	+10.8 dB
5	+21.1 dB	+12.2 dB
6	+23.1 dB	+14.4 dB
Resolution: 16 bits Number of ADCs: 60 System Bandwidth = 9 times individual ADC bandwidth		

Table 4.3: Table indicating improvement of proposed solution over randomization. Data from fig. 4.19.

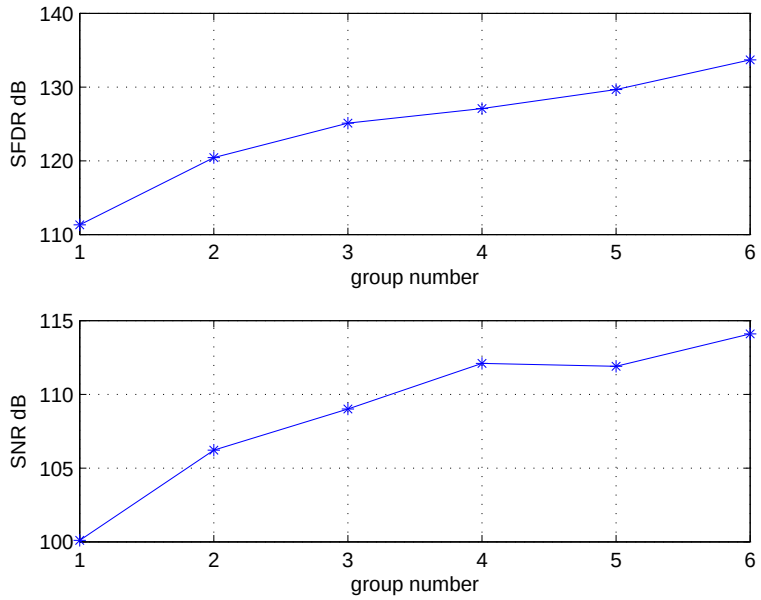


Figure 4.20: 18-bit system; max time error ±3E-5 Ts single ADC, max gain error = ±2E-3 (Full scale), max offset error = ±20%LSB. FIR filter with: max pass-band ripple=1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. N=60.

4.2.4 Simulation results: influence of conversion law non-linearity

A further test of the system could be performed in order to see the performance of it in the case of a random nonlinear conversion law transfer function. The software

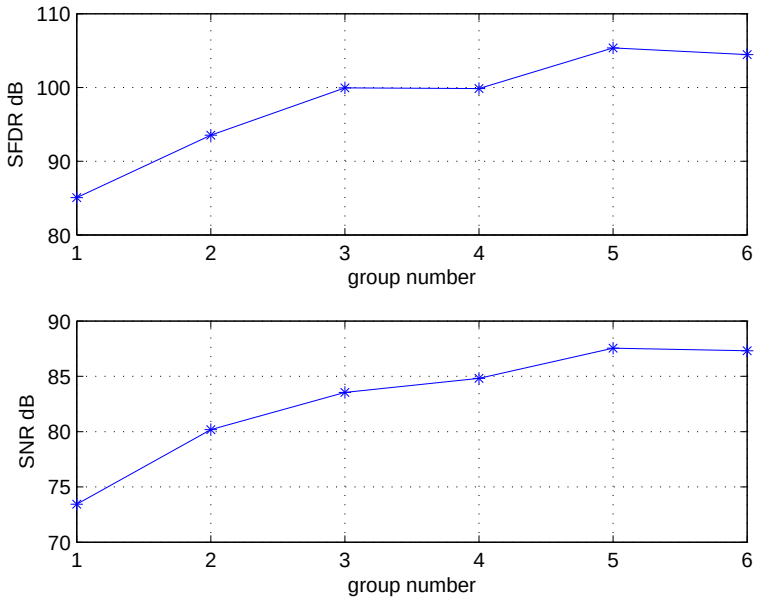


Figure 4.21: 14-bit system; max time error ±6E-4 Ts single ADC, max gain error = ±1E-2 (Full scale), max offset error = ±30%LSB. FIR filter with: max pass-band ripple=1E-12, max stop-band ripple = 1E-5, transition bandwidth = 10% pass-band. N=60.

is reported in appendix B.4.

Simulation description

In all previously described simulations, all ADCs were modelled with an ideal conversion law. In order to get more close to the real world, in this simulation the ideal conversion law has been substituted with a real one with variations from the ideal model. These variations have been chosen randomly according to a uniform distribution. Randomly chosen values of gain and offset mismatches has been added to the system. The number of groups varies from 1 to 4. The input bandwidth equals Ni times the one of the single channel ADCs with Ni equals 2. The resolution of each channel ADC is the same resolution of the overall system that is 10-bit. The total number of ADCs into the system N is fixed to the value of 12 ADCs. The parameters of the filter are selected in order that its non-idealities can be considered as negligible compared to the resolution of the system. The time mismatch between

each ADCs is chosen randomly according to a uniform distribution. It is referred to the sampling period of each ADC in order to test the system under worst case conditions (see simulations made in previous sections for a detailed explanation about this point).

Considering that this simulation wants to quantize the impact of conversion law non-idealities on the efficiency of the algorithm, during the simulation non-linearities have been decreased according with the graph reported in picture 4.22. As it can

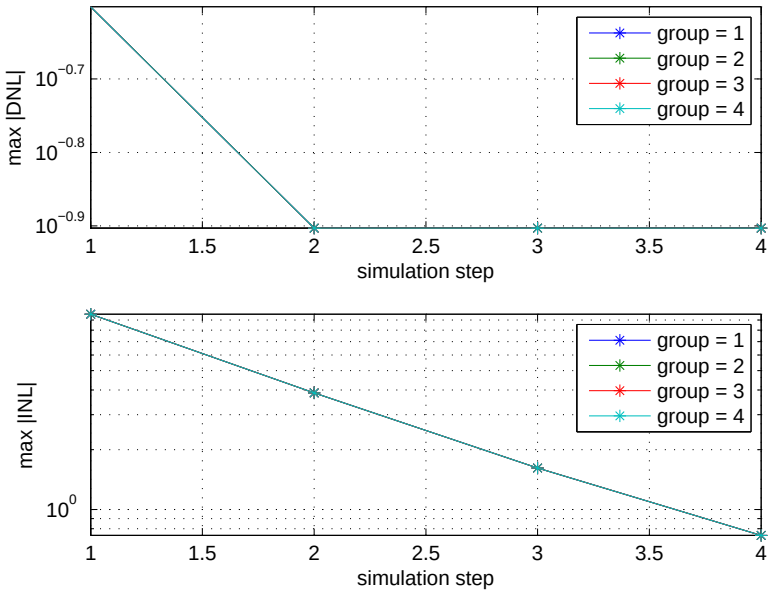


Figure 4.22: 10-bit system; max time error ±1.5E-4 Ts single ADC, max gain error =±1E-3 (Full scale), max offset error =±60%LSB; FIR filter with: max pass-band ripple=1E-12, max stop-band ripple=1E-5, transition bandwidth = 10% pass-band. Ni=2, N=12 ADCs.

be noted, the integral conversion law non-linearity is reduced by 60% at every step. Obtained results are discussed in next paragraphs.

Analysis and comparison of results

Figure 4.23 shows the results of the previously described simulation. As can be noted from this figure, increasing the group number always increases the performance of the system in term of SNR. This shows that the system is able to provide good results

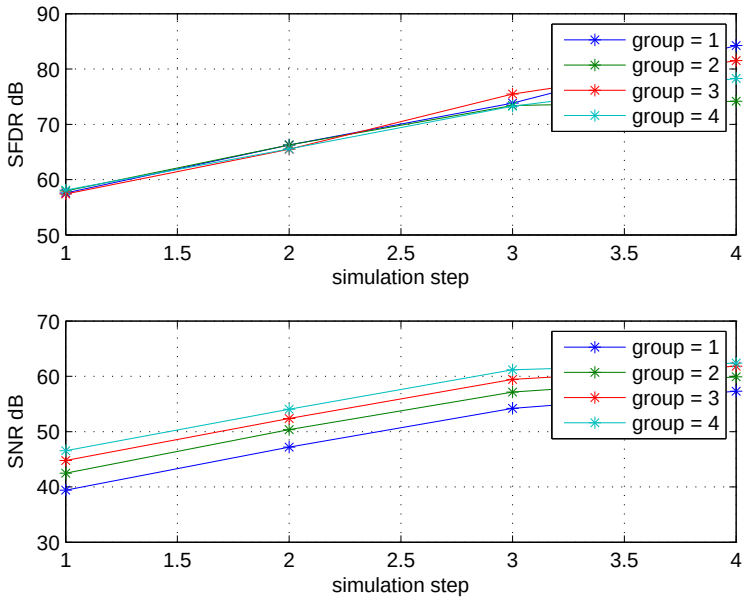


Figure 4.23: 10-bit system; max time error ±1.5E-4 Ts single ADC, max gain error=±10E3(Full scale), max offset error=±60%LSB; FIR filter with: max pass-band ripple=1E-12, max stop-band ripple=1E-5, transition bandwidth=10% pass-band. Ni=2, N=12 ADCs. INL and DNL variation reduction every step according with fig. 4.22.

even in the worsening conditions explained before. A strange behavior affects SFDR. It seems in fact to be that the proposed system doesn't improve it at all. Spurious free dynamic range is the ratio between the signal and the biggest spurious tone in the signal spectrum. This means that the more spurs are high in magnitude, the more SFDR is worsened. In this simulation, the biggest spurs are not caused by mismatches between different ADCs in the architecture, but by the non-linear conversion law. This means that this technique is not able to reduce conversion law non-linearities. Figure 4.24 shows some output signal spectrums demonstrating these effects. In picture a and b it is possible to notice that the SFDR is not limited by the noise floor but by distortion spurs much higher than that level. From both pictures it can be also noticed that even if the spurs due to gain, time and offset mismatches are almost completely eliminated by the algorithm, the ones due to conversion law non-linearities are not and represent the main limitation of the

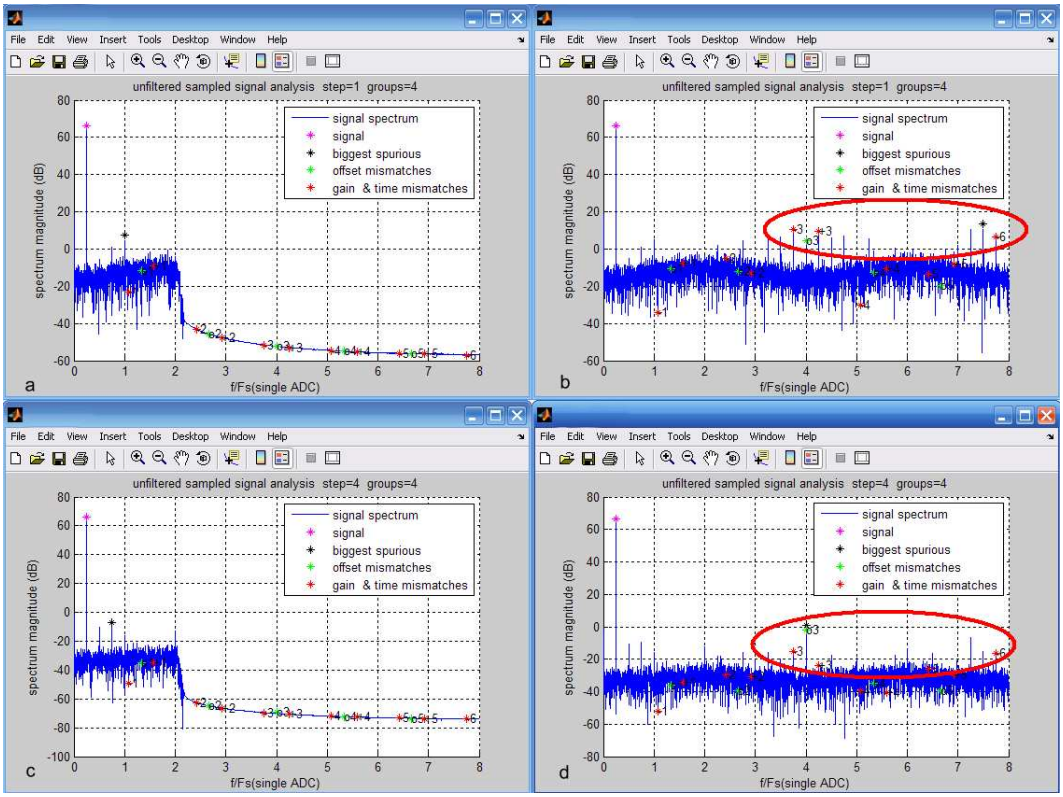


Figure 4.24: some output cases of picture 4.23. Output spectrum of step 1 with 4 groups: a) filtered output, b) unfiltered output. Output spectrum of step 4 with 4 groups: c) filtered output, d) unfiltered output.

resolution of the system. In figure 4.24b, the red circle highlights the fact that the biggest spurious tone (dark *) is not due to time, gain or offset mismatches, but non-linearities in the conversion law.

In the fourth image, SFDR seems to improve as the number of groups increases and this strange behavior seems to disappear. This is because the spurs generated by the non-linear conversion law are not significant compared to the ones generated by time, gain and offset mismatches. This is highlight by the red circle of fig. 4.24 d. The biggest spurious tone in fact (dark *), here is due to offset mismatches. However in picture c, a similar behavior to fig. 4.24 a is shown.

In conclusion, if these spurs due to non-linearities rather than mismatches are below the noise floor, the system works as expected and improves both the SNR and SFDR, otherwise the system performance is limited by the non-linearities of the channel ADCs. This is demonstrated in figure 4.24 where it is possible to notice

that the SFDR is not limited by the noise floor but by significant spurs arising from non-linearities.

4.3 Summary

This chapter describes the proposed system and tests it in order to show its performance under different conditions. To make an absolute comparative analysis of the proposed methods on the matter of performance is quite difficult because different parameters inside the proposed algorithm are highly correlated. Due to this fact, usually a change in the number of groups for example, implies a change in the number of ADCs in the system. This is the reason why results are described and compared with existing methods from different point of view and perspectives.

Section 1 describes the proposed technique for time mismatch correction. Simulation results are also described and compared with existing systems. Section 2 describes the proposed system for gain, time and offset mismatch error correction. Simulation results are described from many perspectives. The area occupation point of view and the influence of conversion law non-linearities have been presented.

The common point of all the different comparisons is that this approach outperforms existing techniques. The randomization technique and the Vogel's method shows worse performance compared with this method. The proposed solution offers an algorithm that is easy to implement, it doesn't require complex computations and it is not limited by any particular technology.

It is also important to notice that this method has been tested under different bandwidth and resolution specifications. In addition to that, performance reported have been tested for input signals with a frequency that is less, equal and even greater than the sampling frequency of each channel ADC.

In relation to this last point, it's important to say that in order to allow the input signal to have a frequency higher than the sampling one of a channel ADC, the ADC must preserve good linearity properties also for signals of this kind inside the overall system bandwidth.

Another limitation of the proposed system is that it has some problems to correct

for conversion law non-linearities, but this is a common unsolved problem for almost all mismatches correction techniques.

Chapter 5

Conclusions and Future work

5.1 Introduction

This chapter summarizes the key research contributions and results, and provides some recommendations for future work. There were two general thrusts of this research: 1) Solutions to overcome limitations in the speed-linearity product of ADC sampling stages were investigated. 2) Mathematical analysis of previous solutions were performed in order to solve the outstanding issues in time interleaved ADCs. Objectives of high levels of integration, area occupation, power and performances issues were also considered when proposing our new methodology. The proposed solution outperforms existing techniques and offers an ease to implement method that is not limited to any particular technology.

5.2 Key research contributions and Results

This research explored the creation of a new architecture for a high-speed, high-resolution Analog to digital Converter able to overcome performances and limitations of previous ones. The new architecture is based on paralleled structures in order to achieve high sampling rate and at the same time high resolution. Parallelization is performed both in time and space. In order to solve problems related to Time-interleaved architectures, an advanced randomization method was introduced. It combines randomization and spectral shaping of timing and compared to conven-

tional randomization algorithms, it improves the SFDR as well as the SINAD. Key research contributions and results are summarized below:

- Investigated limitations of current ADC architectures with particular reference to the bottleneck of these systems. Linearity limitations of input stages have been analyzed as well as conversion law non-linearities.
- Presented a new architectural solution able to overcome and to solve state of the art performance limitations.
- Analyzed and simulated the presented solution with both mathematical equations and computer-based simulations.
- Investigated the state of the art of solutions for problems and limitations in Parallelized architectures.
- Proposed a new solution able to outperform previous ones. The new solution is an advanced randomization method that combines randomization and spectral shaping of mismatches. With a simple low-pass filter the method can improve the SFDR as well as the SINAD.
- Analyzed with both simulations and mathematical analysis from different point of view the efficacy and the easy to implement characteristics of this new method.

5.3 Recommended Future Work

The first part of this thesis focused primarily on the design of a new ADC architecture able to outperform previous ones. Many of the enabling technologies were investigated, and many of these techniques are general to all ADC architectures. The majority of the work however focused on techniques to minimise the effect of mismatches, thus making the construction of a high performance time-interleaved ADC easier. The main advantage of the proposed solution over previous ones is that the requirements on the mismatch identification are very low. Furthermore the correction algorithm is simple and can be implemented on-line and also with a

simple digital logic unit. A highly desirable next steps would be the implementation of the algorithm on a Field Programmable Gate Array with custom designed channel ADCs, with known or controllable characeteristics. In this way empirical measurements could support our theoretical analyses.

A further research area could be the investigation of a possible solution for the issue of minimizing the effects of non-linearities in individual channel ADCs. IF this could be achieved, smaller, lower cost channel ADCs could be designed, utilsing the power of digital circuitry to correct for any errors.

Bibliography

[1] Claude E. Shannon, *A mathematical theory of Communication*, Bell system Technical Journal, vol 27, july-october 1948

[2] Analog Devices, *www.analogdevices.com*

[3] S.R.Norsworthy, R.Schreier, G.C.Temes, *Delta-Sigma Data Converter*, Prentice-Hall, IEEE PRESS, 1997.

[4] Walden, R.H., *Analog-to-digital converter survey and analysis*, IEEE Journal on Selected Areas in Communications, Volume 17, Issue 4, April 1999 Page(s):539 - 550

[5] Xu G., Yuan J., *Accurate sample-and-hold circuit model*, Electronics Letters Volume 41, Issue 9, 28 April 2005 pp. 520-522.

[6] J.Shieh, et al., *Measurement and analysis of charge injection in MOS analog switches*, IEEE J. Solid-State Circuits, vol.22, no.2, April 1987, pp.277-281.

[7] A.Abo,*Design for Reliability of Low-voltage, Switched-capacitor Circuits*, PhD Thesis, University of California, Berkeley, May, 1999.

[8] Chun-Yueh Yang, Chung-Chih Hung, *A low-voltage low-distortion MOS sampling switch Circuits and Systems*, 2005. ISCAS 2005. IEEE International Symposium on 23-26 May 2005 pp. 3131-3134 Vol. 4

[9] Kelly, D.; Yang, W.; Mehr, I.; Sayuk, M.; Singer, L.; *A 3 V 340 mW 14 b 75 MSPS CMOS ADC with 85 dB SFDR at Nyquist*; ISSCC. 2001 IEEE International Solid-State Circuits Conference, 2001. Digest of Technical Papers. 5-7 Feb. 2001 Page(s):134 - 135, 439

[10] Gatti, U.; Maloberti, F.; Palmisano, G.; *An accurate CMOS sample-and-hold circuit*, IEEE Journal of Solid-State Circuits, Volume 27, Issue 1, Jan. 1992 Page(s):120 - 122

[11] K. Watanabe and S . Ogawa, *Clock-feedthrough compensated sample&hold circuits*, Electronics Letters, vol. 24, pp. 1226-1228, Sept. 1988.

[12] Standarovski, D.; Cousineau, M.; Lescure, M.; *Wideband, low-voltage CMOS sample-and-hold amplifier design*; Proceedings of the 21st IEEE Instrumentation and Measurement Technology Conference, 2004. IMTC 04. Volume 3, 18-20 May 2004 Page(s):1765 - 1770 Vol.3

[13] M.Steyaert et al.,*Speed-Power-Accuracy Trade-off in high-speed Analog-to-Digital Converters: Now and in the future...*, Proc. AACD, Tegernsee, April 2000.

[14] Abrishami et al.,*International Technology Roadmap for Semiconductors*, Semiconductor Industry Assoc., 1999.

[15] C.Hu,*Future CMOS Scaling and Reliability*, IEEE Proceedings, vol. 81, no. 5, pp. 682-689, May 1993.

[16] M.Pelgrom et al.,*Matching Properties of MOS Transistors*, IEEE J. of Solid-State Circ., vol 24, no. 5, pp. 1433-1439, Oct. 1989.

[17] Peter Kinget and Michiel Steyaert, *Impact of transistor mismatch on the speed-accuracy-power trade-off of analog CMOS circuits*, Proc. IEEE Custom Integrated Circuit Conference, CICC96, pp.333-336, 1996.

[18] M.Steyaert et al., *Custom Analog Low Power Design: The problem of low-voltage and mismatch*, Proc. IEEECustom Int. Circ. Conf., CICC97, pp.285-292, 1997.

[19] K. Lakshmikumar et al., *Characterization and Modelling of Mismatch in MOS Transistor for Precision Analog Design*, IEEE J. of Solid-State Circ., vol SC-21, no. 6, pp. 1057-11066, Dec. 1986

[20] T. Mizuno et al., *Experimental Study of Threshold Voltage Fluctuation Due to Statistical Variation of Channel Dopant Number in MOSFETs*, IEEE Trans. on Elec. Dev. vol. 41, no.11, pp. 2216-2221, Nov. 1994.

[21] K.McCall et al. *A 6-bit 125 MHz CMOS A/D Converter*, Proc. IEEE Custom Int. Circ. Conf., CICC, 1992.

[22] M.Flynn and D.Allstot, *CMOS Folding ADCs with Current- Mode Interpolation*, IEEE Int. Solid-State Circ. Conf., Dig. Tech. Papers, pp.274-275, Feb. 1995.

[23] F.Paillardet and P.Robert, *A 3.3 V 6 bits 60 MHz CMOS Dual ADC*, IEEE Trans. on Cons. Elec., vol. 41, no. 3, pp. 880-883, Aug. 1995.

[24] J.Spalding and D.Dalton,*A 200MSample/s 6b Flash ADC in 0.6m CMOS*, IEEE Int. Solid-State Circ. Conf., Dig. Tech. Papers, pp.320-321, Feb. 1996.

[25] R.Roovers and M.Steyaert, *A 175 Ms/s, 6b, 160 mW, 3.3 V CMOS A/D Converter*, IEEE J. of Solid-State Circ., vol 31, no. 7, pp. 938-944, July 1996.

[26] S.Tsukamoto et al., *A CMOS 6-b, 200 MSample/s, 3 Vsupply A/D Converter for a PRML Read Channel LSI*, IEEE J. of Solid-State Circ., vol 31, no. 11, pp. 1831-1836, Nov. 1996.

[27] D.Dalton et al., *A 200-MSPS 6-Bit Flash ADC in 0.6-m CMOS*, IEEE Trans. on Circ. and Syst., vol. 45, no. 11, pp. 1433-1444, Nov. 1998.

[28] M.Flynn and B.Sheahan, *A 400-MSample/s 6-b CMOS Folding and Interpolating ADC*, IEEE J. of Solid-State Circ., vol 33, no. 12, pp. 1932-1938, Dec. 1998.

[29] S.Tsukamoto et al., *A CMOS 6-b, 400-MSample/s ADC with Error Correction*, IEEE J. of Solid-State Circ., vol 33, no. 12, pp. 1939-1947, Dec. 1998.

[30] Y.Tamba and K.Yamakido, *A CMOS 6b 500MSample/s ADC for a Hard Disk Drive Read Channel*, IEEE Int. Solid-State Circ. Conf., Dig. Tech. Papers, pp.324-325, Feb. 1999.

[31] K.Yoon et al., *A 6b 500MSample/s CMOS Flash ADC with a Background Interpolated Auto-Zero Technique*, IEEE Int. Solid-State Circ. Conf., Dig. Tech. Papers, pp.326-327, Feb. 1999.

[32] I.Mehr and D.Dalton, *A 500-MSample/s, 6-Bit Nyquist- Rate ADC for Disk-Drive Read-Channel Applications*, IEEE J. of Solid-State Circ., vol 34, no. 7, pp. 912-920, July 1999.

[33] K.Nagaraj et al., *Efficient 6-Bit A/D Converter Using a 1- Bit Folding Front End*, IEEE J. of Solid-State Circ., vol 34, no. 8, pp. 1056-1062, Aug. 1999.

[34] K.Nagaraj et al., *A 700MSample/s 6b Read Channel A/D Converter with 7b Servo Mode*, IEEE Int. Solid-State Circ. Conf., Dig. Tech. Papers, pp.426-427, Feb. 2000.

[35] K.Sushihara et al., *A 6b 800MSample/s CMOS A/D Converter*, IEEE Int. Solid-State Circ. Conf., Dig. Tech. Papers, pp.428-429, Feb. 2000.

[36] K. Bult, *Analog design in deep-submicron CMOS*, in Proc. ESSCIRC,. 2000, pp. 11?17.

[37] Enz C.C.; Temes G.C.;*Circuit techniques for reducing the effects of opamp imperfections: autozeroing, correlated double sampling, and chopper stabilization*, Proceedings of the IEEE Volume 84,Issue 11,Nov. 1996 Page(s):1584-1614.

[38] A. Eshraghi, T. S. Fiez,*A comparative analysis of parallel Delta-sigma ADC architectures*, IEEE transactions on circuits and systems: regular papers, Vol. 51,No. 3, March 2004.

[39] F.Borghetti, C. della Fiore, P. Malcovati, F. Maloberti,*Synthesis of the noise transfer function in n-path sigma delta modulators*, ADDA 2005.

[40] Wei Yu; Subhajit Sen; Leung, B.H.;*Distortion analysis of MOS track-and-hold sampling mixers using time-varying Volterra series*; IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Volume 46, Issue 2, Feb. 1999 Page(s):101-113

[41] W. C. Black, Jr. and D. A. Hodges, *Time-interleaved converter arrays*, IEEE J. Solid State Circuits, vol. SSC-15, pp. 1024-1029, Dec. 1980.

[42] W. C. Black, Jr., *High Speed CMOS A/D Conversion Techniques*, Ph.D. Dissertation, Univ. California, Berkeley, Nov. 1980.

[43] IEEE *Standard for Terminology and Test Methods for Analog-to-Digital Converters*, IEEE Stand. 1241-2000, Jun. 2001.

[44] Vogel, C.; *The impact of combined channel mismatch effects in time-interleaved ADCs*, IEEE Transactions on Instrumentation and Measurement, Volume 54, Issue 1, Feb. 2005 Page(s):415-427

[45] S.R. Velazquez, *Hybrid Filter Banks for Analog/Digital Conversion*, MIT Ph.D. Thesys, 24th July 1997.

[46] J.E.Eklund and F. Gustafsson, *Digital offset compensation of time interleaved ADC using random chopper sampling*, Proc. IEEE Int. Symp. Circuits and Systems, vol. 3, 2000, pp. 447-450.

[47] J. Elbornsson, *Analysis, estimation and compensation of mismatch effects in A/D converters*, Ph.D. dissertation, Dept. of Elect. Eng., Linkoping Univ., Linkoping, Sweden, 2003.

[48] Elbornsson J., Gustafsson F., Eklund J.-E., *Blind adaptive equalization of mismatch errors in a time-interleaved A/D converter system*, IEEE Transactions on Circuits and Systems I: Regular Papers, Volume 51, Issue 1, Jan 2004 Page(s):151 - 158

[49] L. Ljung,*System Identification. Theory for the User*, 2nd ed. Upper Saddle River, NJ: Prentice- Hall, 1999.

[50] H. Stark and J. W. Woods, *Probability Random Processes, and Estimation Theory for Engineers*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[51] J. T. Barnett and B. Kedem, *Zero-crossing rates of functions of Gaussian processes*, IEEE Trans. Inf. Theory, vol. 37, no. 7, pp. 1188-1194, Jul. 1991.

[52] C.-C. Huang and J.-T. Wu, *A background comparator calibration technique for flash analog-to-digital converters*, IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 9, pp. 1732-1740, Sep. 2005.

[53] H. van der Ploeg, G. Hoogzaad, H. A. H. Termeer, M. Vertregt, and R. L. J. Roovers, *A 2.5 V 12-b 54-Msample/s 0.25-m CMOS ADC in 1-mm with mixed-signal chopping and calibration*, IEEE J. Solid-State Circuits, vol. 36, no. 12, pp. 1859-1867, Dec. 2001.

[54] Chung-Yi Wang, Jieh-Tsorng Wu, *A background timing-skew calibration technique for time-interleaved analog-to-digital converters*, IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Volume 53, Issue 4, April 2006 Page(s):299-303

[55] D. Fu, K. C. Dyer, S. H. Lewis, and P. J. Hurst, *A digital background calibration technique for time-interleaved analog-to-digital converters*, IEEE J. Solid-State Circuits, vol. 33, pp. 1904-1911, Dec. 1998.

[56] K. C. Dyer, D. Fu, P. J. Hurst, and S. H. Lewis, *A comparison of monolithic background calibration in two time-interleaved analog-to-digital converters*, in Proc. IEEE Int. Symp. Circuits Systems, May 1998, pp. 13-16.

[57] Kye-Shin Lee, Yunyoung Choi, Maloberti F., *Domino free 4-path time-interleaved second order sigma-delta modulator*, Proceedings of the 2004 International Symposium on Circuits and Systems, 2004. ISCAS '04. Volume 1, 23-26 May 2004 Page(s):I - 473-6.

[58] Ferragina V., Fornasari A., Gatti U., Malcovati P., Maloberti F., *Gain and offset mismatch calibration in time-interleaved multipath A/D sigma-delta modulators*; IEEE Transactions on Circuits and Systems I: Regular Papers, Volume 51, Issue 12, Dec. 2004 Page(s):2365-2373

[59] Maymandi-Nejad M., Sachdev M., *A digitally programmable delay element: design and analysis*; IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Volume 11, Issue 5, Oct. 2003 Page(s): 871-878

[60] Laakso T.I., Valimaki V., Karjalainen M., Laine U.K.,*Splitting the unit delay*,IEEE Signal Processing Magazine, Volume 13, Issue 1, Jan. 1996 Page(s):30-60

[61] C. W. Farrow, *A continuously variable digital delay element*, in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS-88), vol. 3, pp. 2641-2645, Espoo, Finland, June 6-9, 1988.

[62] Huawen Jin, Lee E.K.F., *A digital-background calibration technique for minimizing timing-error effects in time-interleaved ADCs*; IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing; Volume 47, Issue 7, July 2000 Page(s):603-613

[63] [22] Chen H.-H., Lee J., Chen J.-T., *Digital background calibration for timing mismatch in time-interleaved adcs*, Electronics Letters; Volume 42, Issue 2, Jan. 2006 Page(s):74-75

[64] Jamal S.M., Daihong Fu, Chang N.C.-J., Hurst P.J., Lewis S.H., *A 10-b 120-Msample/s time-interleaved analog-to-digital converter with digital background calibration*, IEEE Journal of Solid-State Circuits, Volume 37, Issue 12, Dec. 2002 Page(s):1618-1627

[65] Elbornsson J., Gustafsson F., Eklund J.-E., *Analysis of mismatch effects in a randomly interleaved A/D converter system*,IEEE Transactions on Circuits and Systems I: Regular Papers, Volume 52, Issue 3, March 2005 Page(s):465-476

[66] Vogel C., Draxelmayr D., Kubin G., *Spectral shaping of timing mismatches in time-interleaved analog-to-digital converters*; IEEE International Symposium on Circuits and Systems, 2005. ISCAS 2005, 23-26 May 2005 Page(s):1394-1397 Vol. 2

[67] Vogel C., Pammer V., Kubin G., *A Novel Channel Randomization Method for Time-Interleaved ADCs*, Proceedings of the IEEE Instrumentation and Measurement Technology Conference, 2005. IMTC 2005. Volume 1, 16-19 May 2005 Page(s):150-155.

[68] Global Spec: the engineering search engine,*www.globalspec.com*

[69] F.Zanini,*Current state of the art of ADCs*, N.U.I.M. Professional report, 1st November 2005.

[70] F.Zanini,*Principles of Sigma-Delta modulators*, N.U.I.M. Professional report, 11th December 2005.

[71] Allaboutcircuits.com: this site provides a series of online textbooks covering electricity and electronics,*www.allaboutcircuits.com*

[72] S.Park,*Motorola Digital Signal Processors*, MOTOROLA®APR8/D.

[73] R.Schreier, G.C.Temes, *Understanding Delta-Sigma Data Converters*, IEEE PRESS, 2005,ISBN 0-471-46585-2.

[74] F.Zanini,*Principles of S&H*, N.U.I.M. Professional report, 5th Jenuary 2006.

[75] B.Razavi,*Design of analog CMOS integrated circuits*, preview edition, McGraw Hill, 2000.

[76] F.Zanini,*Design of a suitable S&H architecture compatible with a 100-200MS/s ADC with 16-18 bit of resolution*, N.U.I.M. Professional report, 12th April 2006.

[77] F.Zanini,*Study of generation of distortion harmonics caused by mismatches in parallel ADC architectures and suitable solutions*, N.U.I.M. Professional report, 1st September 2006.

[78] B.Murmann,*Slides from EE315*, Stanford University, 2005.

[79] Bult K.,*Analog design in deep sub-micron CMOS*, Solid-State Circuits Conference, 2000. ESSCIRC '00. Proceedings of the 26th European 19-21 Sept.2000. pp:126-132.

[80] Elbornsson J., Gustafsson F., Eklund J.-E.,*Amplitude and gain error influence on time error estimation algorithm for time interleaved A/D converter system*, IEEE International Conference on Acoustics, Speech, and Signal Processing, Volume 2, 2002 Pp:1281-1284

[81] K. C. Dyer, D. Fu, P. J. Hurst, and S. H. Lewis,*A comparison of monolithic background calibration in two time-interleaved analog-to-digital converters*, in Proc. IEEE Int. Symp. Circuits Systems, May 1998, pp. 13-16.

Appendix A

How to perform a simulation in order to test Track and Hold linearity parameters

A.1 Introduction

The best way to test system linearity parameters is to perform a FFT simulation of the input and output data. If the system is linear, the output spectra is an exact copy of the input one except for a difference in the amplitude scaling factor and a delay in time. The non-linear behavior of a system can be quantified looking at the frequency spectrum of the output signal. Distortion infact generates harmonics that are located at frequencies multiple of the fundamental one. The more the system is distorting the input signal, the more distortion harmonics can be compared in amplitude with the fundamental one. This way is very effective in quantifying distortion and linearity parameters, however it is very difficult to set simulation parameters properly in order to obtain the desired output without any influence by the finite simulation time, finite digit resolution simulator, finite resolution solving algorithm, finite CPU speed etc... The aim of this section is to elaborate a mathematical theory able to allow the user to set FFT simulation parameters in order to estimate the linearity of a system in the best way possible for the given constraints. This mathematical theory gives closed form solutions to all problems a designer has to face while performing

linearity simulations. In order to avoid the negative effect of raised problems, the following conditions must be all true:
(1): The FFT simulation must process 2^q number of samples where x is related to the FFT frequency resolution in bit; this optimizes the solving FFT algorithm.
(2): The processed signal must contain an integer number of input signal periods; this avoids the finite length signal problem.
(3): The signal must be sampled and processed in different points along one period in order to test system linearity in different points of the sinusoid cycle (the phase of the sampled signal module 2π never repeats in 2^x points); this assures to test the linearity behavior in as many points as possible for the given constraints.
(4): Simulator resolution must be properly set as well as the solving algorithm steps; the resolution error should be keep as little as possible in order to not cover distortion harmonics with simulation quantization noise. In the section 5.2 the mathematical theory in order to match all these conditions is developed while in section 5.3 an example of these powerful formulas is presented.

A.2 Development of mathematical theory

A.2.1 Solving for conditions of points (1) and (4)

The fact that the processed signal must contain 2^q number of samples is due to the fact that the FFT algorithm is very efficient when a power of 2 samples are processed. The parameter q in this condition is unknown, to set it, it must be known that it is related to the simulator quantization noise floor. This kind of noise is related to the fact that, every signal processed by a real simulator is made by a finite number of digit. The real value of a pure sinusoidal signal, in a determined point usually is made by an infinite long number of digit. The simulator, approximates the value of the signal with its finite digit resolution and the noise generated by this process has the same characteristic of quantization noise generated by an A/D Converter (in practice it is a conversion process). This noise floor should be kept lower than the linearity requirement of the device under test in order not to cover with it distortion harmonics of the system under test. The power of this quantization noise entirely

depends on the simulator resolution (simulation step, relative tolerance etc...), but the level of the noise floor depends on the parameter q. This is because this power is spread all over the FFT harmonics of the output spectrum. To keep the quantization noise power as low as possible the simulator print step time and solving algorithm time step must be fixed at the same value. This value must be as little as possible and a power of 10 more little of the FFT sampling period. This is in order to sample not interpolated value of the input signal. Considering that the number of harmonics is exactly like the simulation number of points equal to 2^q, the higher this number is and the lower the noise floor is. The ratio between the power of the simulator approximation noise and the FFT quantization noise floor is the FFT quantization noise processing gain $FFT_{QNPGain}$ is equal to:

$$FFT_{QNPGain}(dB) = 10 \cdot LOG_{10}(2^{q-1}) \tag{A.2.1}$$

solving for q:

$$q = \frac{FFT_{QNPGain}(dB)}{10 \cdot LOG_{10}2} + 1 \approx \frac{FFT_{QNPGain}(dB)}{3.01} + 1 \tag{A.2.2}$$

These ideas are clarified by the following example:

Question: 'Given a simulator with an absolute resolution of 2^-10 and a sinusoidal signal with amplitude equal to 1Volt; we want to obtain a noise floor lower than 100db. What is the value of q ?'

Solution: the quantization noise power QNP is given by:

$$QNP \approx 6.02 \cdot (resolutionbits) + 1.76 = 6.02 \cdot 10 + 1.76 = 61.96dB \tag{A.2.3}$$

the needed processing gain is:

$$FFT_{QNPGain}(dB) > 100 - QNP = 100 - 61.96 = 38.04dB \tag{A.2.4}$$

substituting into eq. A.2.2:

$$q > (38.04/3.01) + 1 = 13.64 \tag{A.2.5}$$

with a value of q=14, a -102.89dB noise floor is obtained.

A.2.2 Solving for conditions of points (2) and (3)

To match the conditions that the input signal must be sampled 2^x times in C integer input signal periods T_x and in equispaced time intervals T_s, the following equation must be true:

$$CT_x = 2^x T_s \tag{A.2.6}$$

solving for T_s:

$$T_s = \frac{CT_x}{2^x} \tag{A.2.7}$$

Cause of the fact that there are 2 unknown variables and in only one equation, there is one more degree of freedom that can be used to match the last condition of points (2) and (3). This equation doesn't take into account that the phase of the sampled signal module 2π must never repeats in 2^x points. Now a sampling instant kT_s is considered. Cause of the fact that the signal is periodic with period T_x, the same phase shift of the periodic signal is repeated every m integer number of the signal period T_x. This way the same phase shift point are being sampled every $kT_s + mT_x$ seconds. The missing condition is achieved if the first occurrence of this happens for every value of m greater than C. The preceding sentence is summarized in the following expression:

$$kT_s + nT_s = mT_x + kT_s, \quad only\ with\ m \geq C \tag{A.2.8}$$

substituting eq.A.2.7 into eq.A.2.8:

$$nC = m2^x, \quad only\ with\ m \geq C \tag{A.2.9}$$

this condition is verified if the minimum common multiple (m.c.m.) of C, 2^x equals $C \cdot 2^x$. This is true if C is prime factor composed by numbers different from 2 that simply means that C must be odd. Once matched this condition, the frequency of the input signal is (from A.2.7):

$$F_x = \frac{1}{T_x} = \frac{C}{T_s 2^x} \tag{A.2.10}$$

Even if a closed form for F_x is derived, now the exact value of F_x is still unknown. C in fact can assume all odd value between 1 and infinite. Thanks to this degree

of freedom, there is the possibility to set approximately the input signal frequency value. It is now possible to state that:

$$F_x = kF_s \tag{A.2.11}$$

where $F_s = 1/T_s$.Substituting eq.A.2.11 into eq.A.2.10 and solving for C:

$$C = 2^x k \tag{A.2.12}$$

Even if this equation now sets a determined value for C, there is no guarantee that this value is a decimal number with finite number of digit. This further condition is necessary in order to be able to insert it in the simulator without any approximation error. To match this, if we define T_s and C as:

$$T_s = g \cdot 2^h \cdot 2^y \tag{A.2.13}$$

$$C = J \cdot g \tag{A.2.14}$$

so substituting eq.A.2.12 into A.2.14:

$$J \approx \frac{2^x k}{g} \tag{A.2.15}$$

substituting this into eq.A.2.10:

$$F_x = \frac{Jg}{2^x 2^h 5^y g} = \frac{J}{2^{(x+h)} 5^y} \tag{A.2.16}$$

Looking at previous equation it can be noted that Fx is given by an equation where the denominator is composed by powers of 2 and 5. This assures that Fx is a decimal number with finite number of digit, in fact:

$$F_x = J\frac{2^{(y-x-h)}}{10^y}, \quad if\ y \geq x+h \tag{A.2.17}$$

$$F_x = J\frac{5^{(x+h-y)}}{10^{(x+h)}}, \quad if\ y < x+h \tag{A.2.18}$$

in both cases the numerator is composed by integer numbers and the denominator by a power of 10. This leads to:

$$F_x = J2^{(y-x-h)}10^{-y}, \quad if\ y \geq x+h \tag{A.2.19}$$

$$F_x = J5^{(x+h-y)}10^{-(x+h)}, \quad if\ y < x+h \tag{A.2.20}$$

After all these equations, one link is still missing; what is the relationship between q (conditions 1 and 4) and x (conditions 2 and 3)? The period of observation of the input signal T_0 is given by:

$$T_0 = 2^q T_s \tag{A.2.21}$$

the period T_{nr} during which the the input signal is never sampled twice in the same phase shift points module 2π is:

$$T_{nr} = 2^x T_s \tag{A.2.22}$$

The ratio v between To and Tnr is the number of times the input signal is sampled in the same phase shift points module. This equals to:

$$v = \frac{T_0}{T_{nr}} = \frac{2^q T_s}{2^x T_s} = 2^{(q-x)} \tag{A.2.23}$$

Now the resulting FFT spectrum in both sub-sampling and oversampling case is going to be presented.

(Case 1) Sub-sampling:

The number of spectral lines equals to $2^{(q-1)}$, the position of the spectral line corresponding to F_x in this case is quite difficult to calculate cause of the aliasing effect of this type of sampling. An easy explanation of the solution is provided through the help of the picture of fig.A.1. As it is possible to infer from picture A.1, the sine-wave generated from sampled data looks a sinusoidal signal with a period equals to $T_{xonspectrum}$. $T_{xonspectrum}$ equals to the number of samples sampling it in different phase shift module 2π points multiplied by Ts. In our case, the number of those points is equal to 2^x from condition (2). The following equations unifies last paragraph:

$$T_{xonspectrum} = 2^x T_s \tag{A.2.24}$$

considering that $F_x = 1/T_x$, and that $F_s = 1/T_s$, we have:

$$F_{xonspectrum} = \frac{F_s}{2^x} \tag{A.2.25}$$

using equations A.2.25, A.2.23, and considering that the frequency resolution of the FFT equals to $\delta f = F_s/2^q$, the position P_{fx} of F_x expressed in terms of spectral

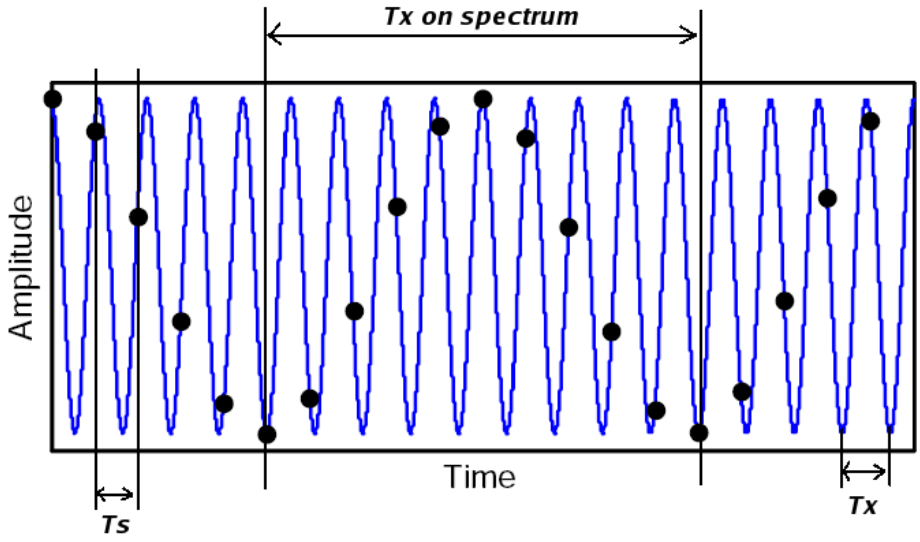


Figure A.1: Input signal, sampling time and aliasing effect. (picture from [76])

lines after the zero is:

$$P_{fx} = \frac{F_{xonspectrum}}{\delta f} = \frac{F_s/2^x}{F_s/2^q} = v \tag{A.2.26}$$

This means that, the number of the spectral lines comprises between 0 and Fx spectral line can be chosen simply by observing the input signal for a time that is v times $2^x T_s$. Picture of fig.A.2 shows preceding expressions results in graphical way. An important remark should be noted on the relative position r of Fx spectral line

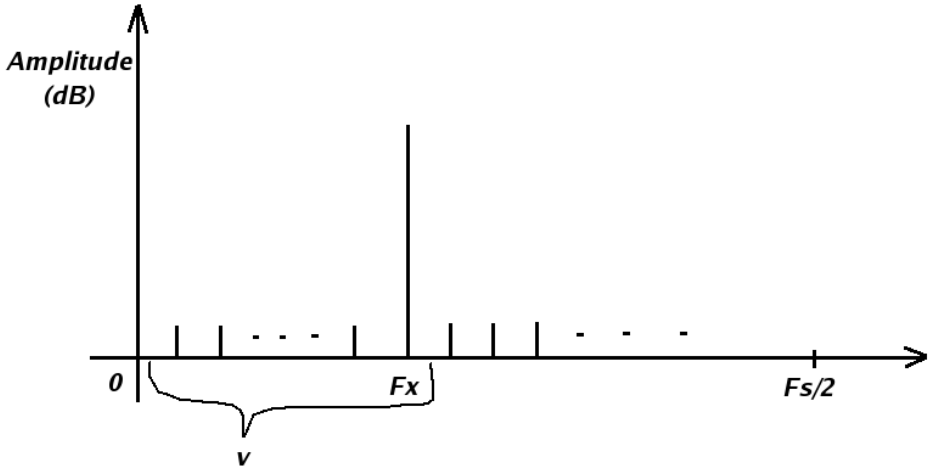


Figure A.2: Output spectrum of an FFT in the case that $Fx > Fs/2$.

over Fs/2. Using equations A.2.26,A.2.23 and the fact that the FFT contains 2^{q-1}

spectral lines:

$$r = \frac{F_{xonspectrum}}{F_s/2} = \frac{v}{v2^{(x-1)}} = \frac{1}{2^{(x-1)}} \tag{A.2.27}$$

As can be noted r depends only on the on the parameter x.

(Case 2) Over-sampling:

This case is easier than the preceding one because aliasing is not present. Using equations A.2.10, A.2.23, and considering that the frequency resolution of the FFT equals to $\delta f = F_s/2^q$, the position $P_f x$ of F_x expressed in terms of spectral lines after zero is:

$$P_{fx} = \frac{F_x}{\delta f} = \frac{CF_s/2^x}{F_s/2^q} = Cv \tag{A.2.28}$$

The relative position r of F_x spectral line over $F_s/2$, using equations A.2.10 and A.2.12 is:

$$r = \frac{F_x}{F_s/2} = \frac{CF_s/2^x}{F_s/2} = C2^{(1-x)} \approx 2k \tag{A.2.29}$$

As can be noted r depends approximately only on the on the parameter k. Picture of fig.A.3 shows preceding expressions results in graphical way.

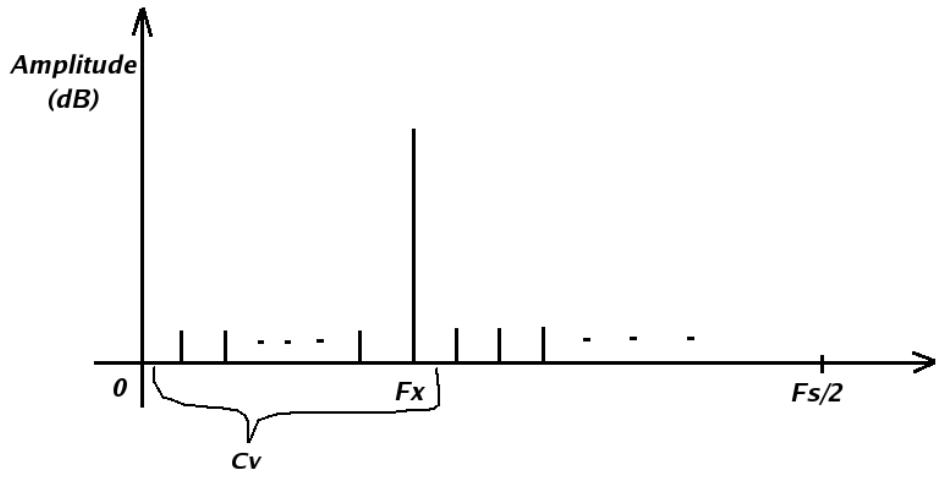


Figure A.3: Output spectrum of an FFT in the case that $Fx < Fs/2$.

A.3 Practical examples

Question: 'basing on the following input data, set FFT simulator parameter in order to test the linearity of a system with frequencies that are one third of the sampling

frequency and 6 times it. Given data : (a) Fs = 10 Mhz; (b) v = 2; (c) x = 10;'

Solution: from (a), (eq.A.2.13) and (eq.A.2.14):

$$10MHz = 10^6 Hz \rightarrow g = 1 \rightarrow J = C \tag{A.3.30}$$

$$T_s = \frac{1}{F_s} = 100 \cdot 10^{-9} s \tag{A.3.31}$$

substituting into (eq.A.2.15) and (A.3.30):

$$J \approx k2^x \approx k2^{10} \tag{A.3.32}$$

from the text of the question it is known that $k_1 = 0.3$ and $k_2 = 6$. Substituting this into past equations and knowing that $C_{1,2}$ must be odd:

$$C_1 \approx 1024 \cdot 0.3 \approx 307.2 \rightarrow 307 \tag{A.3.33}$$

$$C_1 \approx 1024 \cdot 6 \approx 6144 \rightarrow 6145 \tag{A.3.34}$$

substituting into (eq.A.2.10):

$$F_{x1} = \frac{307}{1024 \cdot 100 \cdot 10^{-9}} = 2.998046875MHz \tag{A.3.35}$$

$$F_{x1} = \frac{6145}{1024 \cdot 100 \cdot 10^{-9}} = 60.009765625MHz \tag{A.3.36}$$

using data (c), (b), (eq.A.3.31) and (eq.A.2.21), in both case T_0 equals to:

$$T_0 = 2 \cdot 1024 \cdot 100 \cdot 10^{-9} = 204.8 \cdot 10^{-6} s \tag{A.3.37}$$

The power of this theory is shown in pictures of fig.A.4 and A.5. The simulator print time step is set to 0.1ns and the solving algorithm step is kept constant at the same value. Should be noted that according to past equations the spectral line corresponding to the input signal is located approximately at 60% of the frequency axis. Furthermore, as it is shown in the zoom of picture A.6 that the spectral line corresponding to the input signal is exactly the second one after the zero.

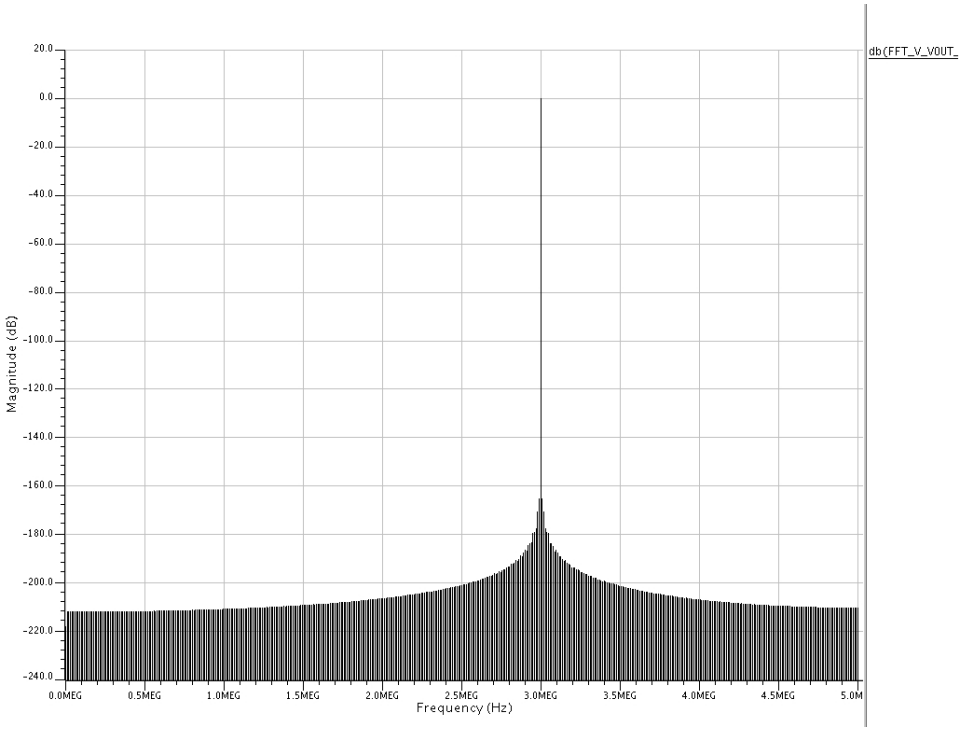


Figure A.4: Output signal spectrum with k=0.3.

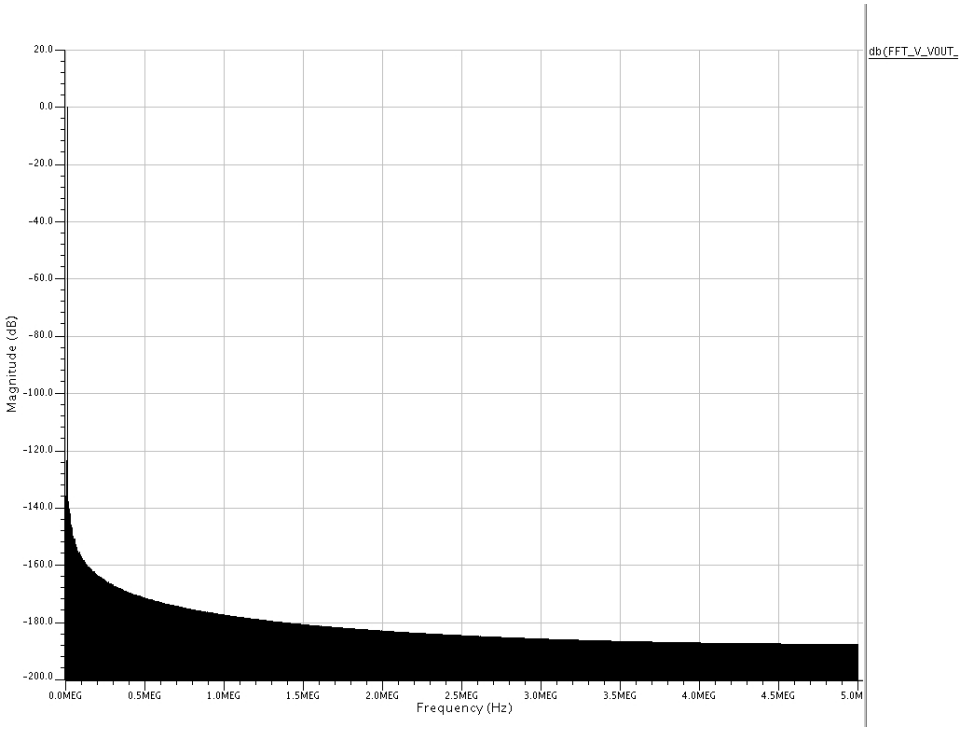


Figure A.5: Output signal spectrum with k=6.

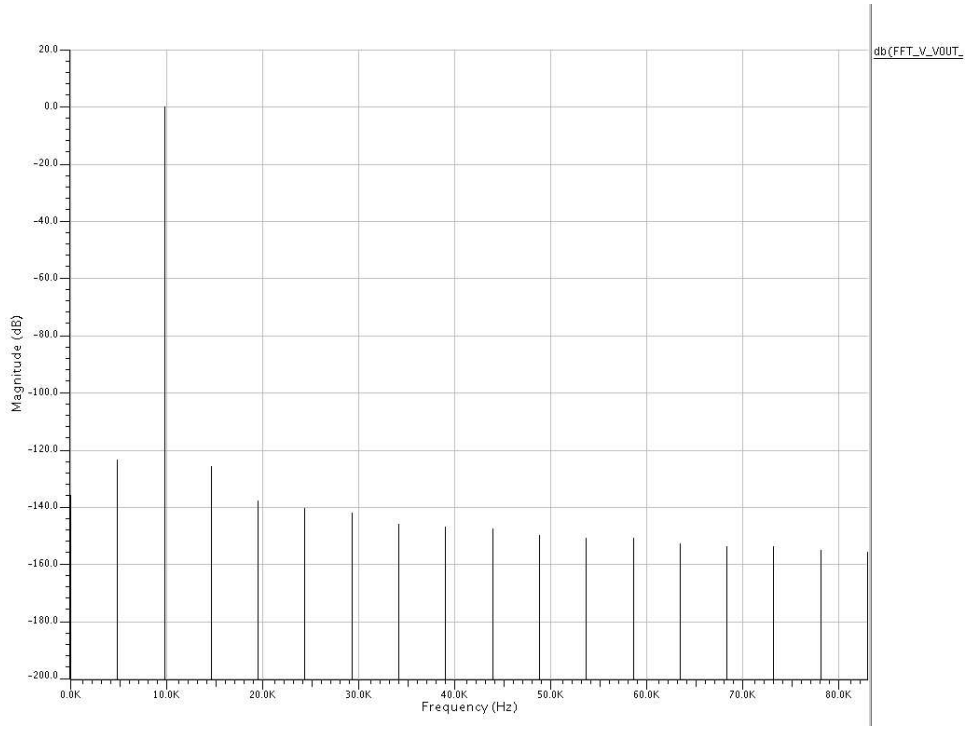


Figure A.6: Zoom of picture A.5 (low frequency spectral lines).

Appendix B

Simulation softwares

B.1 Software simulating the TI architecture performance for fixed input bandwidth and group number

(This is directly related to picture of fig. 4.6)

```
clear all; clc; close all;

%%% copyright information
fprintf('\n\n Software simulating the TI architecture performance');
fprintf('\n for fixed input bandwidth and group number');
fprintf('\n N.U.I. Maynooth');
fprintf('\n Author: Francesco Zanini \n Supervisor: Ronan Farrell \n -
-Date: 28/8/2006 \n');
fprintf('\n\n Simulation started ... \n');
%%%%%%

%%% constants
A=0.5;
ps=2500;       % margine per 0 phase delay filter
sl=2^13;
f=2^-7;
fs=2;
```

```
nbit=14;
ni=4;          % ratio between the input bandwidth of single -
-ADC and overall structure. dtmax=30;
Pe=10^-4;      % time error weight constant
errfix=1;      % 0=% in the overall structure (Ts); 1=% in one -
-ADC (Tss)
bkst=1;
bk=4;
vst=1;
v=8;
htb=0.1;       % filter transition bandwidth % of passband
deltap=10^-12; % max passband ripple (filter response)
deltas=10^-5;  % max stopband ripple (filter response)
ap=1;as=0;     % filter gain in passband and stopband
ncoeff=512;    % filter spectrum visualization points
vfilt=1;       % show filter spectrum (1=yes, 0=no)
SNRt=[]; SFDRt=[];
%%%%%

%%% main simulation cycle
for vbk=bkst:bk for vsim=vst:v

    fprintf('\n\n group=%d   delta M=%d ...\n',vsim,vbk);
    dni=vbk; % number of extra converters each group.
    group=vsim; % number of groups
    N=sl+2*ps; % number input signal samples analyzed;
    ncon=group*(ni+dni); % here it is supposed that OSR=group
    nused=ni*group;
    nc=ni+dni;
    gain=zeros(1,ncon);  % gain and offset errors already been -
    -corrected
    off=zeros(1,ncon);
    seqt=round(rand(1,ncon)*dtmax-dtmax/2); % random time mismat-
    -ches sequence

    dt=sort(seqt); % sorting of time mismatches from lowest to -
    -highest: main algorithm principle
```

```
%%% Randomized sequence + mismatch shaping "seqrand" deter-
-mination.
seq=[ncon];ncc=0;kc=0;
while kc<N
    stp=length(seq)-nused+2;
    if (stp<1) stp=1;end;
    set=seq(stp:length(seq));
    j=randint(1,1,nc)+1+nc*ncc;
    if (find(set==j)>0)
    else
        kc=kc+1;
        seq=[seq j];
        ncc=ncc+1;
        if (ncc==group) ncc=0;end;
    end;
end;

% Time Interleaving algorithm
dm=1;
in=[];
outd=[];
outdf=[];
digm=[];
fil=zeros(ncon,1);
mh=zeros(ncon,1);
ind=zeros(1,ncon);
for j=1:N       % time interleaving virtual multiplexing
    cu=seq(j);
    ind(cu)=ind(cu)+1;
    if (ind(cu)==dm)
        fil=[fil mh];
        dm=dm+1;
    end;
    if (errfix)
        fil(cu,ind(cu))=A*sin(2*pi*f/fs*(j+Pe*nused*dt(cu-
        -))); % fissa 1 ADC
```

```
    else
        fil(cu,ind(cu))=A*sin(2*pi*f/fs*(j+Pe*dt(cu))); % -
        -overall stucture
    end;
end;
for i=1:ncon    % quantization algorithm
        in=fil(i,:);
        in=in.*(1+(gain(i)/100))+(off(i)/100)*(2^-nbit)+A;
        digm(i,:)= ((2^-nbit).*(round(in./(2^-nbit))))-A;
end;
ind=zeros(1,ncon);
for j=1:N      % time interleaving virtual demultiplexing
    cu=seq(j);
    ind(cu)=ind(cu)+1;
    in(j)=fil(cu,ind(cu));
    outd(j)=digm(cu,ind(cu));
end;
%%%%%%

%%% calculations on the converted signal
outdef=outd(ps:(ps+sl-1));
Yout=fft(outdef);
gt=abs(Yout).^2;
ss=2*max(gt);
sigpows=sum(gt);
SNR=10*log10(ss/(sigpows-ss));
datafft2=20*log10(abs(Yout));
h=sl/2;
h1=linspace(0,nused,h);
datafft2=datafft2(1:h);

%%% plot of fft spectrum with some highlighting in it.
figure;hold on;grid on;
plot(h1,datafft2);
[i,j]=max(datafft2);
plot(h1(j),i,'m*');
peak=[];
```

```
val=[];
k=3;
while (k<length(datafft2)-4)
    k=k+1;
    if ((datafft2(k)>datafft2(k-3)+3) && (datafft2(k)>data-
    -fft2(k+3)+3))
        [i1,j1]=max(datafft2(k-3:k+3));
        if (((k-4+j1)~=j) && (h1(abs((k-4+j1)-j))>0.002))
            peak=[peak (k-4+j1)];
            val=[val datafft2(k-4+j1)];
        end;
        k=k+3;
    end;
end;
[i1,j1]=max(val);
sip=h1(j);si=i;
spp=h1(peak(j1));sp=i1;
SFDR=i-i1;
plot(h1(peak(j1)),i1+3,'k*');
for k=1:ncon-1
    o=round(h*2*k/ncon)+1;
    if (o<=h) plot(h1(o),datafft2(o),'g*');text(h1(o),data-
    -fft2(o),['o' num2str(k)]);end;
    g=round(h*2*k/ncon)+j;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text-
    -(h1(g),datafft2(g),['+' num2str(k)]);end;
    g=round(h*2*k/ncon)-j+2;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text-
    -(h1(g),datafft2(g),['-' num2str(k)]);end;
end;
legend('signal spectrum','signal','biggest spurious','offset-
- mismatches','gain  & time mismatches');
xlabel('f/Fs(single ADC)');ylabel('spectrum magnitude (dB)');
title(['unfiltered sampled signal analysis  groups=' num2-
-str(vsim) '  extraconv-per-group=' num2str(vbk)]);
hold off;
fprintf('\n Unfiltered signal:');
```

```
fprintf('\n signal tone rilevated at %6f (f/Fs(single ADC)) -
-\n with an intensity of %6f dB ',h1(j),i);
fprintf('\n biggest distortion tone rilevated at %6f (f/Fs(sin-
-gle ADC)) \n with an intensity of %6f dB',h1(peak(j1)),i1);
fprintf('\n SFDR rilevated : %6f dB',SFDR);
fprintf('\n SNR unfiltered: %6f dB',SNR);

%%% digital filtering of converted signal to cut noise and-
- harmonics

%%% calculus of equiripple filter to cut spurious distortion-
- harmonics
if (vsim>1)
    fcut=1/vsim;        % filter cutoff frequency
    fpf=fcut*(1-htb);
    fsf=fcut*(1+htb);
    [M,ff,m,w]= firpmord([fpf fsf],[ap as],[deltap deltas],fs);
    fprintf('\n Filter order: %6d ',M);
    [b, delta]= firpm(M,ff,m,w);

    if (vfilt)  % filter visualization
        [hr,w]=freqz(b,1,ncoeff);
        figure;
        plot(w,20*log10(abs(hr)));
        title(['Filter Frequency Response Magnitude groups=' -
        -num2str(vsim) '  extraconv-per-group=' num2str(vbk)]);
        hold on;grid on;
        xlabel('Radian frequency (rad/sample)');ylabel('Ma-
        -gnitude (dB)');
    end;

    % filtering
    outdf=filter(b,1,outd);
    outdf=outdf(ps:(ps+sl-1));

    %%% calculations on the converted signal
    Yout=fft(outdf);  % spettro
```

```
gt=abs(Yout).^2;
sigpow=10*log10(sum(gt));
sigpows=sum(gt);
Yout=Yout(1:h);
datafft2=20*log10(abs(Yout));
ss=2*max(gt);
SNR=10*log10(ss/(sigpows-ss));

%%% plot of filtered signal spectrum with some highligh-
-ting in it.
figure;hold on;grid on;
plot(h1,datafft2);
[i,j]=max(datafft2);
plot(h1(j),i,'m*');
peak=[];
val=[];
k=3;
while (k<length(datafft2)-4)
    k=k+1;
    if ((datafft2(k)>datafft2(k-3)+3) && -
    -(datafft2(k)>datafft2(k+3)+3))
        [i1,j1]=max(datafft2(k-3:k+3));
        if (((k-4+j1)~=j) && (h1(abs((k-4+j1)-j))>0.002))
            peak=[peak (k-4+j1)];
            val=[val datafft2(k-4+j1)];
        end;
        k=k+3;
    end;
end;
[i1,j1]=max(val);
SFDR=i-i1;
sip=h1(j);si=i;
spp=h1(peak(j1));sp=i1;
plot(h1(peak(j1)),i1+3,'k*');
legend('signal spectrum','signal','biggest spurious');
xlabel('f/Fs(single ADC)');ylabel('spectrum magnitude -
-(dB)');title(['filtered signal spectrum;  groups=' num-
```

```
        -2str(vsim) '  extraconv-per-group=' num2str(vbk)]);
        hold off;
        fprintf('\n Filtered signal:');
        fprintf('\n signal tone rilevated at %6f (f/Fs(single -
        -ADC)) \n with an intensity of %6f dB ',h1(j),i);
        fprintf('\n biggest distortion tone rilevated at %6f -
        -(f/Fs(single ADC)) \n with an intensity of %6f dB',h1-
        -(peak(j1)),i1);
        fprintf('\n SFDR rilevated : %6f dB',SFDR);
        fprintf('\n SNR filtered: %6f dB\n',SNR);
    end;
    %%%
    SNRt=[SNRt SNR];
    SFDRt=[SFDRt SFDR];
    fprintf('\n\n calculation ended \n');
end; end;

%%% main plot procedure
vtmp=vst:v;
k=1;
SFDRtp=[];
SNRtp=[];
vect=[];
for j=1:length(SFDRt)
    SFDRtp(j-((k-1)*(v-vst+1)),k)=SFDRt(j);
    SNRtp(j-((k-1)*(v-vst+1)),k)=SNRt(j);
    if (j==(k*(v-vst+1)))
        k=k+1;
        vect=[vect;vtmp];
    end;
end;
testo=[];
for i=bkst:bk
    files=['delta M=' num2str(i)];
    testo=[testo;files];
end;
figure;
```

```
subplot(2,1,1);
plot(vect',SFDRtp,'-*');
hold on;
grid on;
ylabel('SFDR dB');
xlabel('group number');
legend(testo);
hold off;
subplot(2,1,2);
plot(vect',SNRtp,'-*');
hold on;
grid on;
ylabel('SNR dB');
xlabel('group number');
legend(testo);
hold off;
fprintf('\n\n Simulation ended. \n');
%%% end
```

B.2 Software simul. the TI arch. perf. with gain and offset error mismatches for fixed input bandwidth and group number

(This is directly related to picture of fig. 4.17)

```
clear all; clc; close all;

%%% copyright information
fprintf('\n\n Software simulating the TI architecture performance');
fprintf('\n with gain and offset error mismatches');

fprintf('\n for fixed input bandwidth and group number');
fprintf('\n N.U.I. Maynooth');
```

```
fprintf('\n Author: Francesco Zanini \n Supervisor: Ronan Farrell-
- \n Date: 28/8/2006 \n'); fprintf('\n\n Simulation started ... \n');
%%%%%%

%%% constants
A=0.5;
ps=2500;       % margine per 0 phase delay filter
sl=2^13; f=2^-7; fs=2; nbit=16;
ni=4;          % ratio between the input bandwidth of single -
-ADC and overall structure. dtmax=30;
Pe=5*10^-6;    % time error weight constant
errfix=1;      % 0=% in the overall structure (Ts); 1=% in -
-one ADC (Tss) bkst=1; bk=1; vst=1; v=8;
htb=0.1;       % filter transition bandwidth % of passband
deltap=10^-12; % max passband ripple (filter response)
deltas=10^-5;  % max stopband ripple (filter response)
ap=1;as=0;     % filter gain in passband and stopband
ncoeff=512;    % filter spectrum visualization points
vfilt=0;       % show filter spectrum (1=yes, 0=no)
SNRt=[]; SFDRt=[]; gmax=10^-2; omax=30;
%%%%%%

%%% main simulation cycle
for vbk=bkst:bk for vsim=vst:v

    fprintf('\n\n group=%d   delta M=%d ...\n',vsim,vbk);
    dni=vbk; % number of extra converters each group.
    group=vsim; % number of groups
    N=sl+2*ps; % number input signal samples analyzed;
    ncon=group*(ni+dni); % here it is supposed that OSR=group
    nused=ni*group;
    nc=ni+dni;

    gain=(randint(1,ncon,(gmax*2*10^10))-(gmax*10^10))/(2*10^10);
    off=randint(1,ncon,(2*omax))-omax;

    seqt=round(rand(1,ncon)*dtmax-dtmax/2); % random time mis-
```

```
-matches sequence

dt=sort(seqt); % sorting of time mismatches from lowest to -
-highest: main algorithm principle

%%% Randomized sequence + mismatch shaping "seqrand" deter-
-mination.
seq=[ncon];ncc=0;kc=0;
while kc<N
    stp=length(seq)-nused+2;
    if (stp<1) stp=1;end;
    set=seq(stp:length(seq));
    j=randint(1,1,nc)+1+nc*ncc;
    if (find(set==j)>0)
    else
        kc=kc+1;
        seq=[seq j];
        ncc=ncc+1;
        if (ncc==group) ncc=0;end;
    end;
end;

% Time Interleaving algorithm
dm=1;
in=[];
outd=[];
outdf=[];
digm=[];
fil=zeros(ncon,1);
mh=zeros(ncon,1);
ind=zeros(1,ncon);
for j=1:N       % time interleaving virtual multiplexing
    cu=seq(j);
    ind(cu)=ind(cu)+1;
    if (ind(cu)==dm)
        fil=[fil mh];
        dm=dm+1;
```

```
    end;
    if (errfix)
        fil(cu,ind(cu))=A*sin(2*pi*f/fs*(j+Pe*nused*dt(cu)));-
        - % fissa 1 ADC
    else
        fil(cu,ind(cu))=A*sin(2*pi*f/fs*(j+Pe*dt(cu))); % -
        -overall stucture
    end;
end;
for i=1:ncon    % quantization algorithm
        in=fil(i,:);
        in=in.*(1+(gain(i)/100))+(off(i)/100)*(2^-nbit)+A;
        digm(i,:)= ((2^-nbit).*(round(in./(2^-nbit))))-A;
end;
ind=zeros(1,ncon);
for j=1:N      % time interleaving virtual demultiplexing
    cu=seq(j);
    ind(cu)=ind(cu)+1;
    in(j)=fil(cu,ind(cu));
    outd(j)=digm(cu,ind(cu));
end;
%%%%%

%%% calculations on the converted signal
outdef=outd(ps:(ps+sl-1));
Yout=fft(outdef);
gt=abs(Yout).^2;
ss=2*max(gt);
sigpows=sum(gt);
SNR=10*log10(ss/(sigpows-ss));
datafft2=20*log10(abs(Yout));
h=sl/2;
h1=linspace(0,nused,h);
datafft2=datafft2(1:h);

%%% plot of fft spectrum with some highlighting in it.
figure;hold on;grid on;
```

```
plot(h1,datafft2);
[i,j]=max(datafft2);
plot(h1(j),i,'m*');
peak=[];
val=[];
k=3;
while (k<length(datafft2)-4)
    k=k+1;
    if ((datafft2(k)>datafft2(k-3)+3) && (datafft2(k)>data-
    -fft2(k+3)+3))
        [i1,j1]=max(datafft2(k-3:k+3));
        if (((k-4+j1)~=j) && (h1(abs((k-4+j1)-j))>0.002))
            peak=[peak (k-4+j1)];
            val=[val datafft2(k-4+j1)];
        end;
        k=k+3;
    end;
end;
[i1,j1]=max(val);
sip=h1(j);si=i;
spp=h1(peak(j1));sp=i1;
SFDR=i-i1;
plot(h1(peak(j1)),i1+3,'k*');
for k=1:ncon-1
    o=round(h*2*k/ncon)+1;
    if (o<=h) plot(h1(o),datafft2(o),'g*');text(h1(o),data-
    -fft2(o),['o' num2str(k)]);end;
    g=round(h*2*k/ncon)+j;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text(h1-
    -(g),datafft2(g),['+' num2str(k)]);end;
    g=round(h*2*k/ncon)-j+2;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text(h1-
    -(g),datafft2(g),['-' num2str(k)]);end;
end;
legend('signal spectrum','signal','biggest spurious','offset-
- mismatches','gain  & time mismatches');
xlabel('f/Fs(single ADC)');ylabel('spectrum magnitude (dB)');
```

```
title(['unfiltered sampled signal analysis  groups=' num2-
-str(vsim) ' extra conv per group=' num2str(vbk)]);
hold off;
fprintf('\n Unfiltered signal:');
fprintf('\n signal tone rilevated at %6f (f/Fs(single ADC)) -
-\n with an intensity of %6f dB ',h1(j),i);
fprintf('\n biggest distortion tone rilevated at %6f (f/Fs-
-(single ADC)) \n with an intensity of %6f dB',h1(peak(j1)),i1);
fprintf('\n SFDR rilevated : %6f dB',SFDR);
fprintf('\n SNR unfiltered: %6f dB',SNR);

%%% digital filtering of converted signal to cut noise -
-and harmonics

%%% calculus of equiripple filter to cut spurious distor-
-tion harmonics
if (vsim>1)
    fcut=1/vsim;        % filter cutoff frequency
    fpf=fcut*(1-htb);
    fsf=fcut*(1+htb);
    [M,ff,m,w]= firpmord([fpf fsf],[ap as],[deltap deltas],fs);
    fprintf('\n Filter order: %6d ',M);
    [b, delta]= firpm(M,ff,m,w);

    if (vfilt)  % filter visualization
        [hr,w]=freqz(b,1,ncoeff);
        figure;
        plot(w,20*log10(abs(hr)));
        title(['Filter Frequency Response Magnitude groups=-
        -' num2str(vsim) '  extraconv-per-group=' num2str(vbk)]);
        hold on;grid on;
        xlabel('Radian frequency (rad/sample)');ylabel('Magn-
        -itude (dB)');
    end;

    % filtering
    outdf=filter(b,1,outd);
```

```
outdf=outdf(ps:(ps+sl-1));

%%% calculations on the converted signal
Yout=fft(outdf);  % spettro
gt=abs(Yout).^2;
sigpow=10*log10(sum(gt));
sigpows=sum(gt);
Yout=Yout(1:h);
datafft2=20*log10(abs(Yout));
ss=2*max(gt);
SNR=10*log10(ss/(sigpows-ss));

%%% plot of filtered signal spectrum with some high-
-lighting in it.
figure;hold on;grid on;
plot(h1,datafft2);
[i,j]=max(datafft2);
plot(h1(j),i,'m*');
peak=[];
val=[];
k=3;
while (k<length(datafft2)-4)
    k=k+1;
    if ((datafft2(k)>datafft2(k-3)+3) && (datafft2(k)>-
    -datafft2(k+3)+3))
        [i1,j1]=max(datafft2(k-3:k+3));
        if (((k-4+j1)~=j) && (h1(abs((k-4+j1)-j))>0.002))
            peak=[peak (k-4+j1)];
            val=[val datafft2(k-4+j1)];
        end;
        k=k+3;
    end;
end;
[i1,j1]=max(val);
SFDR=i-i1;
sip=h1(j);si=i;
spp=h1(peak(j1));sp=i1;
```

```
        plot(h1(peak(j1)),i1+3,'k*');
        legend('signal spectrum','signal','biggest spurious');
        xlabel('f/Fs(single ADC)');
        ylabel('spectrum magnitude (dB)');
        title(['filtered signal spectrum  groups=' num2-
        -str(vsim) '  extraconv-per-group=' num2str(vbk)]);
        hold off;
        fprintf('\n Filtered signal:');
        fprintf('\n signal tone rilevated at %6f (f/Fs(single-
        - ADC)) \n with an intensity of %6f dB ',h1(j),i);
        fprintf('\n biggest distortion tone rilevated at %6f -
        -(f/Fs(single ADC)) \n with an intensity of %6f dB',-
        -h1(peak(j1)),i1);
        fprintf('\n SFDR rilevated : %6f dB',SFDR);
        fprintf('\n SNR filtered: %6f dB\n',SNR);
    end;
    %%%
    SNRt=[SNRt SNR];
    SFDRt=[SFDRt SFDR];
    fprintf('\n\n calculation ended \n');
end; end;

%%% main plot procedure
vtmp=vst:v; k=1; SFDRtp=[]; SNRtp=[]; vect=[];
for j=1:length(SFDRt)
    SFDRtp(j-((k-1)*(v-vst+1)),k)=SFDRt(j);
    SNRtp(j-((k-1)*(v-vst+1)),k)=SNRt(j);
    if (j==(k*(v-vst+1)))
        k=k+1;
        vect=[vect;vtmp];
    end;
end; testo=[];
for i=bkst:bk
    files=['delta M=' num2str(i)];
    testo=[testo;files];
end; figure; subplot(2,1,1); plot(vect',SFDRtp,'-*');
hold on;grid on; ylabel('SFDR dB');xlabel('group number');
```

```
legend(testo); hold off; subplot(2,1,2);
plot(vect',SNRtp,'-*'); hold on;grid on; ylabel('SNR dB');
xlabel('group number'); legend(testo); hold off;
fprintf('\n\n Simulation ended. \n');
%%% end
```

B.3 Software simul. the TI arch. perf. for fixed input bandwidth, group number and ADCs number

(This is directly related to picture of fig. 4.19)

```
clear all; clc; close all;

%%% copyright information
fprintf('\n\n Software simulating the TI architecture performance');
fprintf('\n for fixed input bandwidth, group number and ADCs number');

fprintf('\n N.U.I.Maynooth');
fprintf('\n Author: Francesco Zanini \n Supervisor: Ronan Farrell -
-\n Date: 28/8/2006 \n');
fprintf('\n\n Simulation started ... \n');
%%%%%%

%%% constants
A=0.5;
ps=2500;       % margin for 0 phase delay filter
sl=2^13; f=2^-7; fs=2; nbit=16; dtmax=30;
Pe=10^-5;    % time error weight constant
errfix=1;      % 0=% in the overall structure (Ts); 1=% in one -
-ADC (Tss)
vst=1; v=6;

htb=0.1;       % filter transition bandwidth % of passband
```

```
deltap=10^-12; % max passband ripple (filter response)
deltas=10^-5;  % max stopband ripple (filter response)
ap=1;as=0;     % filter gain in passband and stopband
ncoeff=512;    % filter spectrum visualization points
vfilt=0;       % show filter spectrum (1=yes, 0=no)
SNRt=[]; SFDRt=[];
%%%%%%

%%% main simulation cycle
ncon=1;       % number of converter calculus
for i=2:v
    ncon=lcm(ncon,i);
end;

for vsim=vst:v
    fprintf('\n\n group=%d ',vsim);
    group=vsim; % number of groups
    N=sl+2*ps; % number input signal samples analyzed;
    nimin=((ncon/v)-1);
    nc=ncon/group;
    dni=1;
    ni=nc-dni;
    nused=group*ni;

    gain=zeros(1,ncon);  % gain and offset errors already been-
    - corrected
    off=zeros(1,ncon);
    seqt=round(rand(1,ncon)*dtmax-dtmax/2); % random time mis-
    -matches sequence

    dt=sort(seqt); % sorting of time mismatches from lowest to -
    -highest: main algorithm principle

    %%% Randomized sequence + mismatch shaping "seqrand" deter-
    -mination.
    seq=[ncon];ncc=0;kc=0;
    while kc<N
```

```
    stp=length(seq)-nused+2;
    if (stp<1) stp=1;end;
    set=seq(stp:length(seq));
    j=randint(1,1,nc)+1+nc*ncc;
    if (find(set==j)>0)
    else
        kc=kc+1;
        seq=[seq j];
        ncc=ncc+1;
        if (ncc==group) ncc=0;end;
    end;
end;

% Time Interleaving algorithm
dm=1;
in=[];
outd=[];
outdf=[];
digm=[];
fil=zeros(ncon,1);
mh=zeros(ncon,1);
ind=zeros(1,ncon);
for j=1:N       % time interleaving virtual multiplexing
    cu=seq(j);
    ind(cu)=ind(cu)+1;
    if (ind(cu)==dm)
        fil=[fil mh];
        dm=dm+1;
    end;
    if (errfix)
        fil(cu,ind(cu))=A*sin(2*pi*f/fs*(j+Pe*nused*dt(cu)));-
        - % fissa 1 ADC
    else
        fil(cu,ind(cu))=A*sin(2*pi*f/fs*(j+Pe*dt(cu))); % o-
        -verall stucture
    end;
end;
```

```
for i=1:ncon    % quantization algorithm
        in=fil(i,:);
        in=in.*(1+(gain(i)/100))+(off(i)/100)*(2^-nbit)+A;
        digm(i,:)= ((2^-nbit).*(round(in./(2^-nbit))))-A;
end;
ind=zeros(1,ncon);
for j=1:N      % time interleaving virtual demultiplexing
    cu=seq(j);
    ind(cu)=ind(cu)+1;
    in(j)=fil(cu,ind(cu));
    outd(j)=digm(cu,ind(cu));
end;
%%%%%%

%%% calculations on the converted signal
outdef=outd(ps:(ps+sl-1));
Yout=fft(outdef);
gt=abs(Yout).^2;
ss=2*max(gt);
sigpows=sum(gt);
SNR=10*log10(ss/(sigpows-ss));
datafft2=20*log10(abs(Yout));
h=sl/2;
h1=linspace(0,nused,h);
datafft2=datafft2(1:h);

%%% plot of fft spectrum with some highlighting in it.
figure;hold on;grid on;
plot(h1,datafft2);
[i,j]=max(datafft2);
plot(h1(j),i,'m*');
peak=[];
val=[];
k=3;
while (k<length(datafft2)-4)
    k=k+1;
    if ((datafft2(k)>datafft2(k-3)+3) && (datafft2(k)>data-
```

```
    -fft2(k+3)+3))
        [i1,j1]=max(datafft2(k-3:k+3));
        if (((k-4+j1)~=j) && (h1(abs((k-4+j1)-j))>0.002))
            peak=[peak (k-4+j1)];
            val=[val datafft2(k-4+j1)];
        end;
        k=k+3;
    end;
end;
[i1,j1]=max(val);
sip=h1(j);si=i;
spp=h1(peak(j1));sp=i1;
SFDR=i-i1;
plot(h1(peak(j1)),i1+3,'k*');
for k=1:ncon-1
    o=round(h*2*k/ncon)+1;
    if (o<=h) plot(h1(o),datafft2(o),'g*');text(h1(o),data-
    -fft2(o),['o' num2str(k)]);end;
    g=round(h*2*k/ncon)+j;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text(h1-
    -(g),datafft2(g),['+' num2str(k)]);end;
    g=round(h*2*k/ncon)-j+2;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text(h1-
    -(g),datafft2(g),['-' num2str(k)]);end;
end;
legend('signal spectrum','signal','biggest spurious','off-
-set mismatches','gain  & time mismatches');
xlabel('f/Fs(single ADC)');ylabel('spectrum magnitude (dB)');
title(['unfiltered sampled signal analysis;  groups=' num2-
-str(vsim)]);
hold off;
fprintf('\n Unfiltered signal:');
fprintf('\n signal tone rilevated at %6f (f/Fs(single ADC)) -
-\n with an intensity of %6f dB ',h1(j),i);
fprintf('\n biggest distortion tone rilevated at %6f (f/Fs-
-(single ADC)) \n with an intensity of %6f dB',h1(peak(j1)),i1);
fprintf('\n SFDR rilevated : %6f dB',SFDR);
```

```
fprintf('\n SNR unfiltered: %6f dB',SNR);

%%% digital filtering of converted signal to cut noise and -
-harmonics

%%% calculus of equiripple filter to cut spurious distortion -
-harmonics
%%%
    fcut=nimin/nused;             % filter cutoff frequency
    fpf=fcut*(1-htb);
    fsf=fcut*(1+htb);
    [M,ff,m,w]= firpmord([fpf fsf],[ap as],[deltap delt-
    -as],fs);
    fprintf('\n Filter order: %6d ',M);
    [b, delta]= firpm(M,ff,m,w);

    if (vfilt)  % filter visualization
        [hr,w]=freqz(b,1,ncoeff);
        figure;
        plot(w,20*log10(abs(hr)));
        title(['Filter Frequency Response Magnitude gro-
        -ups=' num2str(vsim)]);
        hold on;grid on;
        xlabel('Radian frequency (rad/sample)');ylabel('Ma-
        -gnitude (dB)');
    end;

    % filtering
    outdf=filter(b,1,outd);
    outdf=outdf(ps:(ps+sl-1));

    %%% calculations on the converted signal
    Yout=fft(outdf);  % spettro
    gt=abs(Yout).^2;
    sigpow=10*log10(sum(gt));
    sigpows=sum(gt);
    Yout=Yout(1:h);
```

```
datafft2=20*log10(abs(Yout));
ss=2*max(gt);
SNR=10*log10(ss/(sigpows-ss));

%%% plot of filtered signal spectrum with some highligh-
-ting in it.
figure;hold on;grid on;
plot(h1,datafft2);
[i,j]=max(datafft2);
plot(h1(j),i,'m*');
peak=[];
val=[];
k=3;
while (k<length(datafft2)-4)
    k=k+1;
    if ((datafft2(k)>datafft2(k-3)+3) && (datafft2(k)>-
    -datafft2(k+3)+3))
        [i1,j1]=max(datafft2(k-3:k+3));
        if (((k-4+j1)~=j) && (h1(abs((k-4+j1)-j))>0.002))
            peak=[peak (k-4+j1)];
            val=[val datafft2(k-4+j1)];
        end;
        k=k+3;
    end;
end;
[i1,j1]=max(val);
SFDR=i-i1;
sip=h1(j);si=i;
spp=h1(peak(j1));sp=i1;
plot(h1(peak(j1)),i1+3,'k*');
legend('signal spectrum','signal','biggest spurious');
xlabel('f/Fs(single ADC)');ylabel('spectrum magnitude -
-(dB)');title(['filtered signal spectrum  groups=' num-
-2str(vsim)]);
hold off;
fprintf('\n Filtered signal:');
fprintf('\n signal tone rilevated at %6f (f/Fs(single -
```

```
        -ADC)) \n with an intensity of %6f dB ',h1(j),i);
        fprintf('\n biggest distortion tone rilevated at %6f -
        -(f/Fs(single ADC)) \n with an intensity of %6f dB',h1-
        -(peak(j1)),i1);
        fprintf('\n SFDR rilevated : %6f dB',SFDR);
        fprintf('\n SNR filtered: %6f dB\n',SNR);
    %%%
    SNRt=[SNRt SNR];
    SFDRt=[SFDRt SFDR];
    fprintf('\n\n calculation ended \n');
end;

%%% main plot procedure
vtmp=vst:v; k=1; SFDRtp=[]; SNRtp=[]; vect=[];
for j=1:length(SFDRt)
    SFDRtp(j-((k-1)*(v-vst+1)),k)=SFDRt(j);
    SNRtp(j-((k-1)*(v-vst+1)),k)=SNRt(j);
    if (j==(k*(v-vst+1)))
        k=k+1;
        vect=[vect;vtmp];
    end;
end;
figure; subplot(2,1,1);
plot(vect',SFDRtp,'-*');
hold on;grid on;
ylabel('SFDR dB');xlabel('group number');
hold off; subplot(2,1,2); plot(vect',SNRtp,'-*');
hold on;grid on; ylabel('SNR dB');xlabel('group number');
hold off; fprintf('\n\n Simulation ended. \n');
%%% end
```

B.4 Software simul. the TI arch. perf. for fixed input bandwidth and group number with non-linear random conversion law

(This is directly related to picture of fig. 4.22 and 4.23)

```
clear all; clc; close all;

%%% copyright information
fprintf('\n\n Software simulating the TI architecture performance');
fprintf('\n for fixed input bandwidth and group number with');

fprintf('\n non-linear random conversion law');
fprintf('\n N.U.I. Maynooth');
fprintf('\n Author: Francesco Zanini \n Supervisor: Ronan Farrell -
-\n Date: 28/8/2006
\n'); fprintf('\n\n Simulation started ... \n');
%%%%%%

%%% constants
A=0.5;
ps=2500;       % margine per 0 phase delay filter
sl=2^13; f=2^-5; fs=2; nbit=10;
ncon=12;       % ratio between the input bandwidth of single -
-ADC and overall structure.
ni=2; dtmax=30;
Pe=10^-5;      % time error weight constant
errfix=1;      % 0=% in the overall structure (Ts); 1=% in one -
-ADC (Tss) bkst=1; bk=4; vst=1; v=4;
htb=0.1;       % filter transition bandwidth % of passband
deltap=10^-12; % max passband ripple (filter response)
deltas=10^-5;  % max stopband ripple (filter response)
ap=1;as=0;     % filter gain in passband and stopband
ncoeff=512;    % filter spectrum visualization points
difflin=800;   % percent LSB deviation from ideal conversion law
sconlaw=0;     % show conversion function (1=yes, 0=no)
```

```
vsi=0;         % detailed input & output signal of each conver-
-ter (1=yes,0=no)
vfilt=0;       % show filter spectrum (1=yes, 0=no)
gmax=10^-3;    % gain and offset mismatch error max variation
omax=60; SNRt=[];SFDRt=[];dmt=[];imt=[]; deformpar=-30;
%%%%%%

%%% main simulation cycle

gain=(randint(1,ncon,(gmax*2*10^10))-(gmax*10^10))/(2*10^10);
off=randint(1,ncon,(2*omax))-omax;

seqt=round(rand(1,ncon)*dtmax-dtmax/2); % random time mismat-
-ches sequence
dt=sort(seqt); % sorting of time mismatches from lowest to -
-highest: main algorithm principle

%%% conversion law
    % random interpolation points of the conversion law function
    v1=[];v2=[];
    v1i=linspace(-2*A,2*A,10);
    for i=1:ncon v1=[v1;v1i]; end;
    LSB=2^-nbit;
    deviation=(LSB/100)*difflin;
    a=(rand(ncon,length(v1i))-0.5)*deviation/0.5;
    v2=v1;
    v2=v2+a;
    v2(:,1)=-2*A;
    v2(:,length(v1i))=2*A;
    v1old=v1;
    v2old=v2;

for vbk=bkst:bk
    v1=v1old;v2=v2old;
for vsim=vst:v

    fprintf('\n\n group=%d, n ADCs = 12, deform step=%d ...\n',-
```

```
-vbk,vsim);
group=vbk; % number of groups
N=sl+2*ps; % number input signal samples analyzed;

nused=ni*group;
nc=ncon/group;
dni=nc-ni;

% conversion law, INL and DNL calculation
dmtemp=0;imtemp=0;
tc=[];
for ty=1:ncon
    xx = linspace(-2*A,2*A,2^(nbit+4));
    yy = spline(v1(ty,:),v2(ty,:),xx);
    l2=2^nbit;
    vi=round(l2.*yy);
    k=find(diff(vi))+1;
    c=xx(k);
    c=[-2*A c 2*A];
    if (sconlaw)
        figure;
        subplot(3,1,1);
        grid on;hold on;
        title(['converter #' num2str(ty) ':  step ' num2-
        -str(vsim) ', group ' num2str(vbk)]);
        xlabel('analog input');ylabel('digital output');
        plot(v1(ty,:),v2(ty,:),'mo',xx,yy,'m-',xx,xx);
        plot(xx,(vi./l2),'r');
        hold off;
    end;
    % INL,DNL calculation
    a=diff(c);
    dnl=(a./LSB)-1;
    dnl=dnl(2:length(dnl)-1);
    dm=max(abs(dnl));
    vi=0;
    for i=2:(length(a)-1)
```

```
        vi=vi+a(i-1);
        inl(i)=((vi+(a(i)/2))/LSB)-(i-1);
    end;
    im=max(abs(inl));
    if (sconlaw)
        subplot(3,1,2);hold on;grid on;
        plot(linspace(min(c),max(c),length(dnl)),dnl);
        xlabel('analog input');ylabel('dnl');
        hold off;
        subplot(3,1,3);grid on;hold on;
        plot(linspace(min(c),max(c),length(inl)),inl);
        xlabel('analog input');ylabel('inl');
        hold off;
    end;

    dmtemp=max(dmtemp,dm);
    imtemp=max(imtemp,im);
    if (ty==1) tc=c;
    else tc=[tc;c];end;
end;
fprintf('\n max|DNL|=%d; max|INL|=%d \n',dmtemp,imtemp);
dmt=[dmt dmtemp];
imt=[imt imtemp];

for q=1:ncon  % deformation procedure
    x=v1(q,:);y=v2(q,:);
    for i=1:length(y);
        vw=deformpar/100*abs(x(i)-y(i));
        if (x(i)>y(i))
            if (x(i)+vw<=max(x) && x(i)+vw>=min(x)) x(i)-
            -=x(i)+vw;end;
            if (y(i)-vw>=min(x) && y(i)-vw<=max(x)) y(i)-
            -=y(i)-vw;end;
        end;
        if (x(i)<y(i))
            if (x(i)-vw>=min(x) && x(i)-vw<=max(x)) x(i)-
            -=x(i)-vw;end;
```

```
            if (y(i)+vw<=max(x) && y(i)+vw>=min(x)) y(i)-
            -=y(i)+vw;end;
        end;
    end;
    v1(q,:)=x;v2(q,:)=y;
end;

% Randomized sequence + mismatch shaping "seqrand" deter-
-mination.
seq=[ncon];ncc=0;kc=0;
while kc<N
    stp=length(seq)-nused+2;
    if (stp<1) stp=1;end;
    set=seq(stp:length(seq));
    j=randint(1,1,nc)+1+nc*ncc;
    if (find(set==j)>0)
    else
        kc=kc+1;
        seq=[seq j];
        ncc=ncc+1;
        if (ncc==group) ncc=0;end;
    end;
end;

% Time Interleaving algorithm
dm=1;
in=[];
outd=[];
outdf=[];
digm=[];
fil=zeros(ncon,1);
mh=zeros(ncon,1);
ind=zeros(1,ncon);
for j=1:N       % time interleaving virtual multiplexing
    cu=seq(j);
    ind(cu)=ind(cu)+1;
    if (ind(cu)==dm)
```

```
            fil=[fil mh];
            dm=dm+1;
        end;
        if (errfix)
            fil(cu,ind(cu))=A*sin(2*pi*f/fs*(j+Pe*nused*dt-
            -(cu))); % fissa 1 ADC
        else
            fil(cu,ind(cu))=A*sin(2*pi*f/fs*(j+Pe*dt(cu))); -
            -% overall stucture
        end;
    end;
    for i=1:ncon    % quantization algorithm
        lc=length(tc(i,:));
        fil(i,:)=fil(i,:).*(1+(gain(i)/100))+(off(i)/100)*LSB;
        for t=1:length(fil(i,:))
            for i2=1:(lc-1)
                if (fil(i,t)>=tc(i,i2) && fil(i,t)<tc(i,i2+1))
                    digm(i,t)=(((i2-1)-(2^nbit))*LSB);
                end;
            end;
        end;
        if (vsi)
            figure;clf;hold on;grid on;
            title(['converter #' num2str(i) '  step=' num2str-
            -(vsim) ', groups=' num2str(vbk)]);
            xlabel('sampling instant');
            ylabel('signal value');
            plot(fil(i,:));
            plot(digm(i,:),'r');
            legend('input signal','output signal');
            hold off;
        end;
    end
    ind=zeros(1,ncon);
    for j=1:N      % time interleaving virtual demultiplexing
        cu=seq(j);
        ind(cu)=ind(cu)+1;
```

```
    in(j)=fil(cu,ind(cu));
    outd(j)=digm(cu,ind(cu));
end;
%%%%%

%%% calculations on the converted signal
outdef=outd(ps:(ps+sl-1));
Yout=fft(outdef);
gt=abs(Yout).^2;
ss=2*max(gt);
sigpows=sum(gt);
SNR=10*log10(ss/(sigpows-ss));
datafft2=20*log10(abs(Yout));
h=sl/2;
h1=linspace(0,nused,h);
datafft2=datafft2(1:h);

%%% plot of fft spectrum with some highlighting in it.
figure;hold on;grid on;
plot(h1,datafft2);
[i,j]=max(datafft2);
plot(h1(j),i,'m*');
peak=[];
val=[];
k=3;
while (k<length(datafft2)-4)
    k=k+1;
    if ((datafft2(k)>datafft2(k-3)+3) && (datafft2(k)>data-
    -fft2(k+3)+3))
        [i1,j1]=max(datafft2(k-3:k+3));
        if (((k-4+j1)~=j) && (h1(abs((k-4+j1)-j))>0.002))
            peak=[peak (k-4+j1)];
            val=[val datafft2(k-4+j1)];
        end;
        k=k+3;
    end;
end;
```

```
[i1,j1]=max(val);
sip=h1(j);si=i;
spp=h1(peak(j1));sp=i1;
SFDR=i-i1;
plot(h1(peak(j1)),i1+3,'k*');
for k=1:ncon-1
    o=round(h*2*k/ncon)+1;
    if (o<=h) plot(h1(o),datafft2(o),'g*');text(h1(o),data-
    -fft2(o),['o' num2str(k)]);end;
    g=round(h*2*k/ncon)+j;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text(h1-
    -(g),datafft2(g),['+' num2str(k)]);end;
    g=round(h*2*k/ncon)-j+2;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text(h1-
    -(g),datafft2(g),['-' num2str(k)]);end;
end;
legend('signal spectrum','signal','biggest spurious','offset -
-mismatches','gain  & time mismatches');
xlabel('f/Fs(single ADC)');ylabel('spectrum magnitude (dB)');
title(['unfiltered sampled signal analysis  step=' num2str-
-(vsim) '  groups=' num2str(vbk)]);
hold off;
fprintf('\n Unfiltered signal:');
fprintf('\n signal tone rilevated at %6f (f/Fs(single ADC))-
- \n with an intensity of %6f dB ',h1(j),i);
fprintf('\n biggest distortion tone rilevated at %6f (f/Fs-
-(single ADC)) \n with an intensity of %6f dB',-
-h1(peak(j1)),i1);
fprintf('\n SFDR rilevated : %6f dB',SFDR);
fprintf('\n SNR unfiltered: %6f dB',SNR);

%%% digital filtering of converted signal to cut noise -
-and harmonics

%%% calculus of equiripple filter to cut spurious distor-
-tion harmonics
if (group>1)
```

```
fcut=1/group;        % filter cutoff frequency
fpf=fcut*(1-htb);
fsf=fcut*(1+htb);
[M,ff,m,w]= firpmord([fpf fsf],[ap as],[deltap delt-
-as],fs);
fprintf('\n Filter order: %6d ',M);
[b, delta]= firpm(M,ff,m,w);

if (vfilt)  % filter visualization
    [hr,w]=freqz(b,1,ncoeff);
    figure;
    plot(w,20*log10(abs(hr)));
    title(['Filter Frequency Response Magnitude step=' -
    -num2str(vsim) ', groups=' num2str(vbk)]);
    hold on;grid on;
    xlabel('Radian frequency (rad/sample)');ylabel('Ma-
    -gnitude (dB)');
end;

% filtering
outdf=filter(b,1,outd);
outdf=outdf(ps:(ps+sl-1));

%%% calculations on the converted signal
Yout=fft(outdf);  % spettro
gt=abs(Yout).^2;
sigpow=10*log10(sum(gt));
sigpows=sum(gt);
Yout=Yout(1:h);
datafft2=20*log10(abs(Yout));
ss=2*max(gt);
SNR=10*log10(ss/(sigpows-ss));

%%% plot of filtered signal spectrum with some high-
-lighting in it.
figure;hold on;grid on;
plot(h1,datafft2);
```

```
[i,j]=max(datafft2);
plot(h1(j),i,'m*');
peak=[];
val=[];
k=3;
while (k<length(datafft2)-4)
    k=k+1;
    if ((datafft2(k)>datafft2(k-3)+3) && (datafft2(k)>-
    -datafft2(k+3)+3))
        [i1,j1]=max(datafft2(k-3:k+3));
        if (((k-4+j1)~=j) && (h1(abs((k-4+j1)-j))>0.002))
            peak=[peak (k-4+j1)];
            val=[val datafft2(k-4+j1)];
        end;
        k=k+3;
    end;
end;
[i1,j1]=max(val);
SFDR=i-i1;
sip=h1(j);si=i;
spp=h1(peak(j1));sp=i1;
plot(h1(peak(j1)),i1+3,'k*');
for k=1:ncon-1
    o=round(h*2*k/ncon)+1;
    if (o<=h) plot(h1(o),datafft2(o),'g*');text(h1(o),da-
    tafft2(o),['o' num2str(k)]);end;
    g=round(h*2*k/ncon)+j;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text-
    (h1(g),datafft2(g),['+' num2str(k)]);end;
    g=round(h*2*k/ncon)-j+2;
    if (g>=0 && g<=h) plot(h1(g),datafft2(g),'r*');text-
    (h1(g),datafft2(g),['-' num2str(k)]);end;
end;
legend('signal spectrum','signal','biggest spurious','of-
fset mismatches','gain  & time mismatches');
xlabel('f/Fs(single ADC)');
ylabel('spectrum magnitude (dB)');
```

```
        title(['unfiltered sampled signal analysis  step=' num2-
        str(vsim) '  groups=' num2str(vbk)]);
        hold off;
        fprintf('\n Filtered signal:');
        fprintf('\n signal tone rilevated at %6f (f/Fs(single -
        -ADC)) \n with an intensity of %6f dB ',h1(j),i);
        fprintf('\n biggest distortion tone rilevated at %6f -
        -(f/Fs(single ADC)) \n with an intensity of %6f dB',-
        -h1(peak(j1)),i1);
        fprintf('\n SFDR rilevated : %6f dB',SFDR);
        fprintf('\n SNR filtered: %6f dB\n',SNR);
    end;
    %%%
    SNRt=[SNRt SNR];
    SFDRt=[SFDRt SFDR];
    fprintf('\n\n calculation ended \n');

end; end;

%%% main plot procedure
vtmp=vst:v; k=1; SFDRtp=[]; SNRtp=[]; vect=[];
for j=1:length(SFDRt)
    SFDRtp(j-((k-1)*(v-vst+1)),k)=SFDRt(j);
    SNRtp(j-((k-1)*(v-vst+1)),k)=SNRt(j);
    dmtp(j-((k-1)*(v-vst+1)),k)=dmt(j);
    imtp(j-((k-1)*(v-vst+1)),k)=imt(j);
    if (j==(k*(v-vst+1)))
        k=k+1;
        vect=[vect;vtmp];
    end
end; testo=[];
for i=bkst:bk
    files=['group = ' num2str(i)];
    testo=[testo;files];
end;
figure;
subplot(2,1,1);
```

```
plot(vect',SFDRtp,'-*');
hold on;
grid on;
ylabel('SFDR dB');
xlabel('simulation step');
legend(testo);
hold off; subplot(2,1,2);
plot(vect',SNRtp,'-*');
hold on;
grid on;
ylabel('SNR dB');
xlabel('simulation step');
legend(testo);
hold off;
figure;
subplot(2,1,1);
semilogy(vect',dmtp,'-*');

hold on;
grid on;
ylabel('max |DNL|');xlabel('simulation step');
legend(testo);
hold off;
subplot(2,1,2);
semilogy(vect',imtp,'-*');
hold on;
grid on;
ylabel('max|INL|');
xlabel('simulation step');
legend(testo);
hold off;
fprintf('\n\n Simulation ended. \n');
%%% end
```

Appendix C

Improving the T&H stage

This section is going to make some consideration about the structure of a high speed, high resolution T&H system. The desired target is to create an architecture compatible with a 100-200 MS/s ADC with 16-18 bit of resolution. In order to do this, next sections are going to analyze and design each part of the system from the technology used to the high level behavior of the system.

C.1 Choice of technology

Process parameters are continuously in evolution. They changed a lot during the past and the will do the same in the future. Table C.1 gives an overview of 14 different processes, ranging from 3.0u down to 0.07u, of which only the last two (0.1u and 0.07u) are predictions (data from [13]-[16]). The Supply Voltage Vdd, Oxide Thickness Tox, Threshold Voltage Vth and Matching parameter AVth of these 14 processes are plotted on a log-log scale in fig. C.1. In [17] and [18] it has been shown that in order to determining the low-end of the dynamic range, matching plays the most important role, rather than noise. Matching is reported to scale with oxide thickness $A_{Vth} = \gamma Tox$ [16], [19], [20], which is visible in fig.C.1. As suggested by [79], the Dynamic Range (DR) is defined as the ratio between Vsig and Vmismatch and $Vmismatch = nA_{Vth}/sqrtWL$ where n is the number of sigma's necessary for a certain yield. It is possible to find that:

$$DR = \frac{\eta_{vol} vdd \sqrt{WL}}{n\gamma Tox} \tag{C.1.1}$$

with $\eta_{vol} = Vsig/Vdd$ being the voltage efficiency. Fig. C.2(a) depicts the DR and the terms it consists of, where $\sqrt{WL}$ (height of lower shaded area) is adjusted such that a constant DR is obtained over all processes. Using the inverse of C.1.1:

$$WL = \frac{n^2 DR^2 (n\gamma Tox)^2}{(\eta_{vol} Vdd)^2} \tag{C.1.2}$$

the gate capacitance may be derived:

$$C_{gate} = \frac{T_{tech} n^2 DR^2 (n\gamma Tox)^2}{(\eta_{vol} Vdd)^2} \tag{C.1.3}$$

#	Lmin	Vdd	Tox	Vth	AVth
1	3.0	5.0	700	1.5	35 *
2	2.5	5.0	600	1.2	30
3	2.0	5.0	400	1.1	25
4	1.5	5.0	250	1.0	22 *
5	1.2	5.0	250	1.0	21 *
6	1.0	5.0	250	0.95	20
7	0.8	5.0	200	0.85	13
8	0.5	3.3	135	0.73	11
9	0.35	3.3	100	0.59	9.0
10	0.25	2.5	60	0.52	6.0
11	0.18	1.8	50	0.42	4.2
12	0.12	1.2	42	0.32	3.8
13	0.10	1.2	36	0.31	3.2 *
14	0.07	0.9	30	0.30	2.5 *
* these parameters have been extrapolated					

Table C.1: Different processes comparison. (data from [13]- [16])

where $T_{tech} = \epsilon_0 \epsilon_r Tox\gamma^2$, a constant depending purely on technology. This capacitance is the gate capacitance of the transistor with a matching requirement to support a certain Dynamic Range (DR). Assuming a maximum signal frequency Fsig, the Slew-Rate current Isr to support a swing of Vsig at frequency Fsig is:

$$Isr = 2\pi C_{gate} VsigFsig = \frac{2\pi T_{tech} n^2 DR^2 Fsig}{\eta_{vol} Vdd} \tag{C.1.4}$$

If this current is delivered by a driver with an efficiency $\eta_{cur} = Isr/Idd$, than the power dissipation P follows:

$$P = \frac{2\pi T_{tech} n^2 DR^2 Fsig}{\eta_{vol}\eta_{cur}} \tag{C.1.5}$$

As suggested by [79], expression C.1.5 consists of 4 separate terms. The first term is process dependent only and is mainly dependent on oxide thickness. The second term is the product of voltage efficiency η_{vol} and current efficiency η_{cur} and reflects circuit 'smartness'. The third term is the result of yield requirements and the last term depicts the system needs. These components are also visible in fig. C.2(b). If constant circuit smartness, yield and system requirements are assumed, power scales down with oxide thickness. As a test to the above derivation of circuit performance versus technology, data was gathered from 15 different 6-bit ADCs [21]- [35]. A figure of merit for an ADC can be defined as:

$$F_{merit} = \frac{fsig}{A_{layout} P} \tag{C.1.6}$$

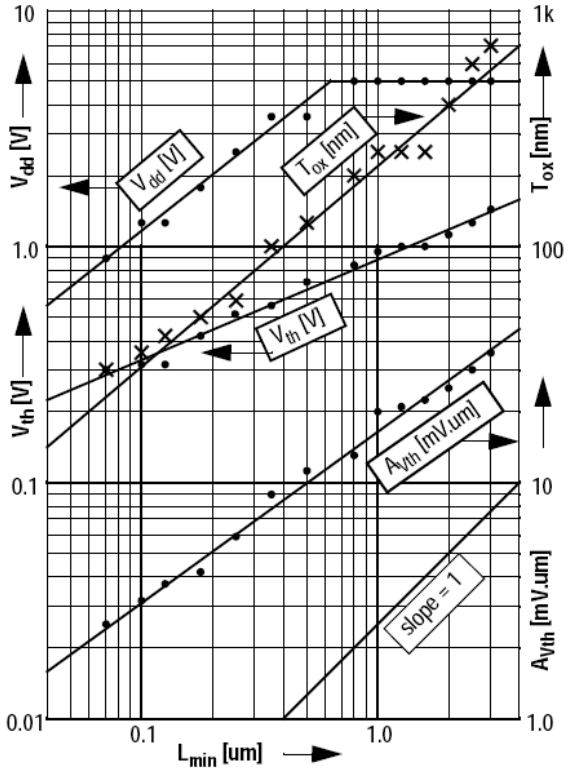


Figure C.1: Different processes comparison graphs. (data from [13]- [16])

and is plotted against technology Lmin in fig.C.3. Assuming the majority of the layout scales with $Lmin^2$ (i.e. source and drain diffusion area's, contacts and wiring) and C.1.5 predicting the power P to scale with Lmin, F_{merit} is expected to scale with $1/Lmin^3$, as is shown clearly in fig.C.3. As suggested by [79], from this point of view the future of analog design in deep submicron does not seem so dark. The crux of the above assumption obviously lies in maintaining a constant product ($\eta_{cur}\eta_{vol}$). Various techniques have been published to overcome this problem. The state of the art of those ones is described in [36] and [37]. Even if those techniques are good in solving low-voltage problems, they are only mitigating and not solving those issues. In high resolution A/D conversion systems some of those problems the more the technology is scaled the more these become unsolvable. The top 3 critical issues related to voltage scaling are now listed:

(1st): the gate overdrive voltage of CMOS sampling/floating switches is becoming too small to achieve fast settling, and it could reach the point where the floating switch would fail to turn on as the supply voltage becomes less than the sum of the NMOS and PMOS threshold voltages.

(2nd): high gain opamps, one of the most important building blocks in SC circuits, are very difficult to design at ultra-low-voltage supplies because most commonly used gain boosting techniques are no longer feasible.

(3rd): it is more difficult to achieve a good signal-to-noise ratio (SNR) due to reduced signal range and increased noise coupling in high density mixed-signal integrated circuits (ICs).

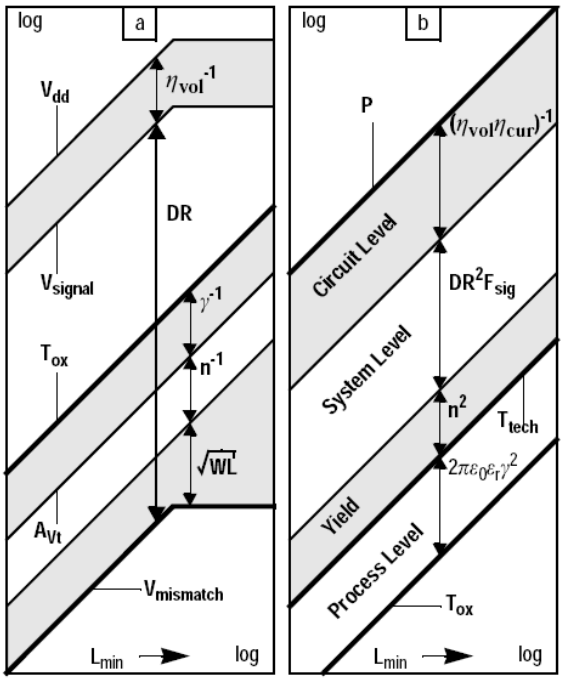


Figure C.2: (a)Dynamic Range (DR) versus Lmin derived from Vdd and process parameters.(b)Power versus Lmin derived from process parameters, yield, system level parameters and circuit level parameters. (picture from [79])

Thus, in order to achieve high performance, a scaled technology is used, but one in which high scaling effect like gate current, low off state channel resistance or reduced Vdd are lightly present. Based on the preceding assumptions a 0.18u CMOS technology is used.
Is it better to use a 3.3V-0.18u CMOS technology or a 1.8V 0.18u one ?
The maximum drain current flowing through these devices in saturation for a given area occupation (Vgs = 3.3V or 1.8V and (W=L=1u)), both technologies shows almost identical performance. This means that for our project specifications, it' s preferable to use a 3.3V technology, in order to allow greater input signal and so reach easily the desired resolution.

C.2 Input stages

This subsection is going to analyze and design an input stage suitable for the desired target. The generalized diagram of a passive sampling input stage is shown in picture of fig. C.4. It is composed by a voltage storage system (capacitor) and a switch able to repeatedly connect and disconnect the last one from the input signal. This switch is represented with a varying resistor controlled by a sampling signal. Theoretically, to avoid any form of input signal distortion, the holding capacitor should be infinite (to suppress thermal noise) and should show a constant frequency behavior

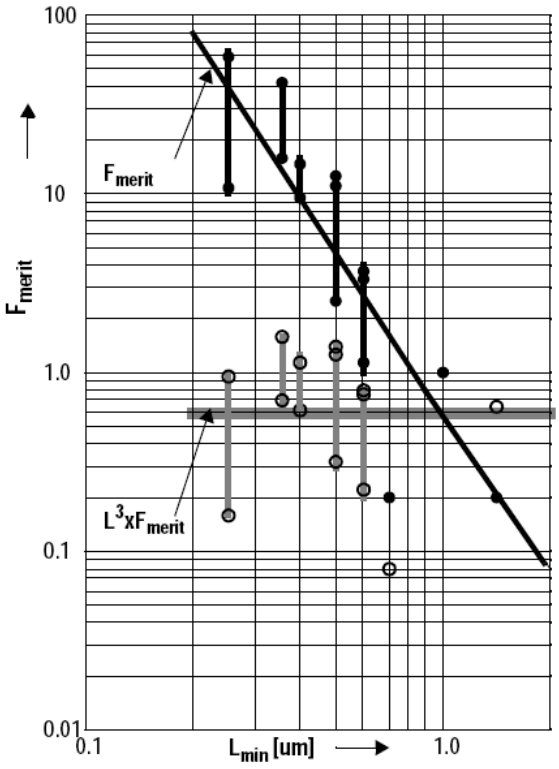


Figure C.3: Figure-of-Merit (F_{merit}) of 14 different 6-bit ADC designs versus Lmin. (picture from [79])

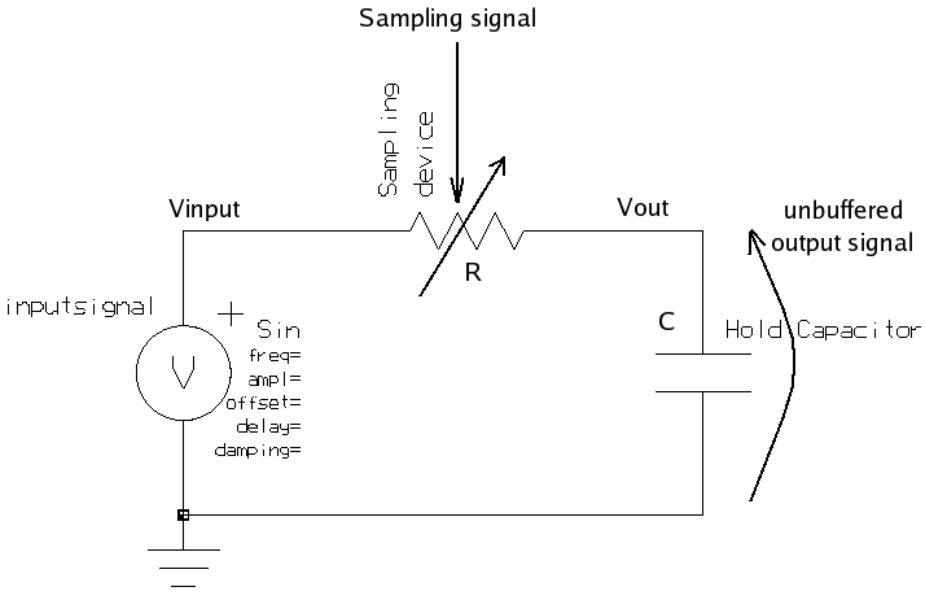


Figure C.4: General passive sampling network architecture.

while the input switch should have infinite resistance when open and zero resistance when closed. Furthermore the transition between these two states must take 0 seconds (to avoid Jitter noise). In real implementations no one of these assumptions is true. In real implementations of such circuits a finite value for both the capacitor and the resistor is needed. Usually, the value of the capacitor is fixed by thermal noise (its power infact is equal to KT/Chold) requirements while the value of the switch resistance can be modeled by the IC designer. If we consider this value in track mode finite but constant, the sampling circuit is described by the differential equation:

$$V_{input}(t) = V_{out}(t) + \frac{\tau dV_{out}}{dt} \tag{C.2.7}$$

where

$$\tau = RC \tag{C.2.8}$$

The complete solution of equation C.2.7 given a sinusoidal input of the form:

$$V_{input}(t) = A \cdot cos(wt) \tag{C.2.9}$$

after some calculations is:

$$V_{out}(t) = (V_{out}(0) - \frac{A}{1+w^2\tau^2})e^{-\frac{t}{\tau}} + \frac{A}{1+w^2\tau^2}cos(wt-\phi) \tag{C.2.10}$$

where

$$\phi = arctg(w\tau) \tag{C.2.11}$$

As simple as this system is, with perfectly finite but linear elements, it is still nonlinear. To linearize the circuit, τ must be minimized sufficiently to incrementally meet the desired linearity. Consider the exponential term in equation C.2.10. Assume that the sampling time duration equals $k\tau$, with k measuring the number of time constants required to attenuate the exponential term by the amount SFDR (decibels) below the maximum of the acquired signal at this frequency. Since $w^2\tau^2 \ll 1$ and A=1, using eq. C.2.10 we have that, to obtain a determined SFDR, k must be:

$$k > \frac{SFDR}{20}ln(10) + ln(2) \tag{C.2.12}$$

Each 6dB change in SFDR thus requires 0.7 more time constants. For example, SFDR=90 dB requires $k > 11$ but SFDR=105 requires $k > 12.8$. From this perspective, maximizing the sampling bandwidth is crucial.

If we replace the ideal capacitor with a real one, other sources of distortion are added to this circuit. A generalized circuit network to model a real RF-MIM capacitor is shown in picture of fig. C.5. In the preceding model, C is the ideal capacitor while all the others are parasitic elements. C_1,C_2 models bulk parasitic coupling, R the wire resistance while L the inductance associate with it. Considering all these effects, the overall capacitance C_{eq} seen between terminal 1-2 shows a value that changes with the frequency. Solving infact the network we obtain a value for C_{eq} that is:

$$C_{eq} = \frac{S^3RLCC_1C_2 + S^2LCC_1 + sRC(C_1+C_2) + C}{S^3RLC_2(C+C_1) + S^2L(C+C_1) + sR(C+C_1+C_2) + 1} \tag{C.2.13}$$

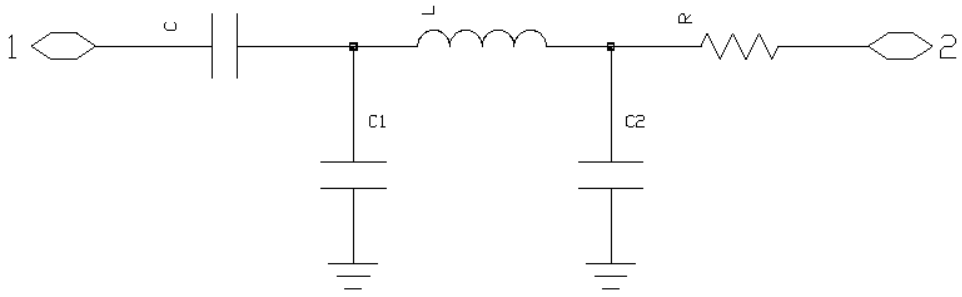


Figure C.5: General real RF-MIM capacitor model.

From the equation can be noted that for:

$$lim_{s\to 0} C_{eq=C} \tag{C.2.14}$$

$$lim_{s\to \infty} C_{eq} = C//C_1 \tag{C.2.15}$$

In picture of fig. C.2.9, the frequency behavior of a real RF-MIM capacitor of 4pF is shown. Looking at the graph can be noted that the relative variation in value of the capacitor between 0 and 100MHz is approximately 159E-6. In picture of fig. C.2.10 an FFT simulation is shown for an input signal of approximately 97MHz. From the picture it is possible to recognize 2nd and 3rd distortion harmonics caused by the capacitor non-ideal behavior. These non-idealities anyway are negligible compared to 18-bit linearity required by project specifications. Step by step the model of the system is getting more and more close to the real implementation of such device. A this stage replacing the ideal switch with a MOSFET, other distortion sources are added to the system. Of all these non-idealities, the main important ones have been explained before. Current solutions based on constant gate-source voltage bootstrapping, bottom-plate sampling or bulk-voltage controlling for body-effect compensation are all quite suitable to achieve desired performance. Supposing that all these techniques worked almost perfectly, at the same time like in the circuit of fig.C.8, we would obtain performance as in fig.C.9. Picture of fig.C.9 shows an harmonic distortion in differential configuration (no second harmonic) of 110.2dB that correspond to approximately 18 effective bit of linearity.

Are all these techniques necessary in order to achieve this linearity?
Only by introducing the body-effect like in circuit of fig.C.10, the achieved linearity falls down to 96.9dB as shown in C.11, allowing only 15.8 effective bit of linearity. This means that those techniques are all essential to achieve the desired resolution at the given frequency.

The last ideal assumption in previous circuits is related to transition time between the sampling and the holding time. Cause of finite rise and fall time of clock signals, the transistor may not sample the signal every Ts seconds and this effect can cause distortion harmonics in the output spectrum of the sampled signal. The following picture explains how the sample time may be dependent on the input signal. If the clock signal is directly applied at the gate of the switching

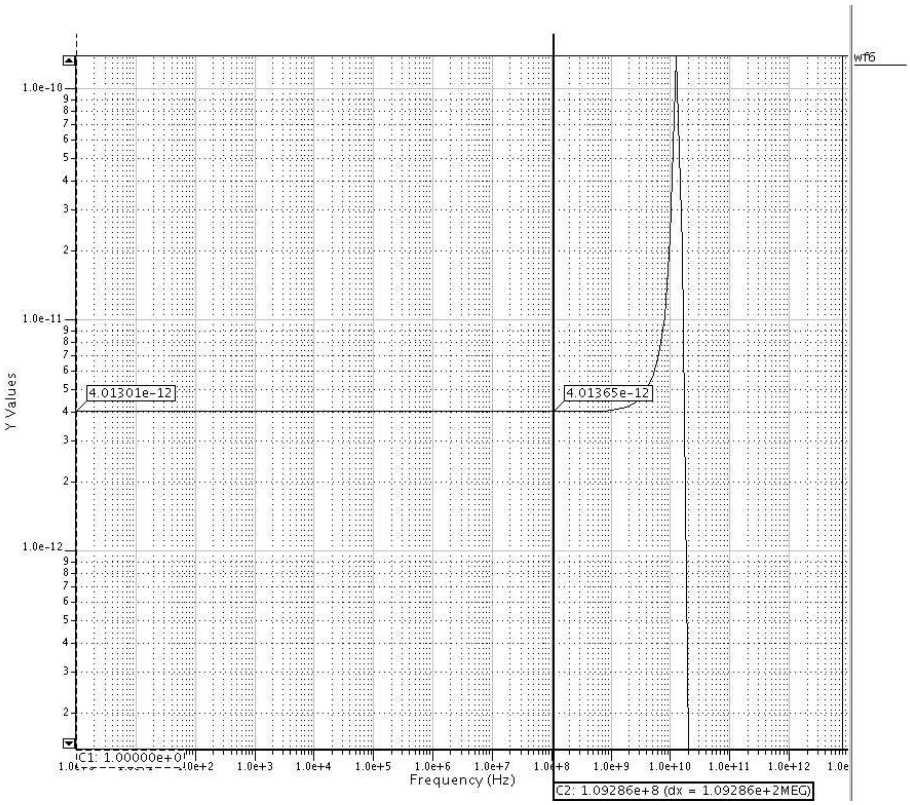


Figure C.6: Frequency behavior of the value of a real RF-MIM capacitor model.

device, the Vgs voltage of it depends on the input signal, and so the sampling time is input-dependent. The bootstrap technique helps in preventing this kind of effect. For the jitter not to cause the sample to be greater than half an LSB away from the correct value it must be:

$$\sigma_j < \frac{10^{-\frac{SNR}{20}}}{2\pi F_s} \tag{C.2.16}$$

Consequently, from the required SNR of 110 dB with a sampling frequency F_s of 100MS/s, $\sigma_j < 5.035E-15s \approx 5fs$ which is considerably stringent even using bootstrapping techniques.

C.3 Input signal acquisition: single Vs differential

In this subsection the best way to acquire the input signal from an external signal source is going to be identifyed. During past decades, two main ways have been proposed in order to do that: single ended acquisition and differential acquisition.

In a single-ended configuration all signals are referenced to a common ground. Figure C.13 shows a simplified example of a track-and-hold (T/H) input of a single-ended ADC. The differential architecture is shown in fig. C.14. In this architecture all signals are not referenced to a common

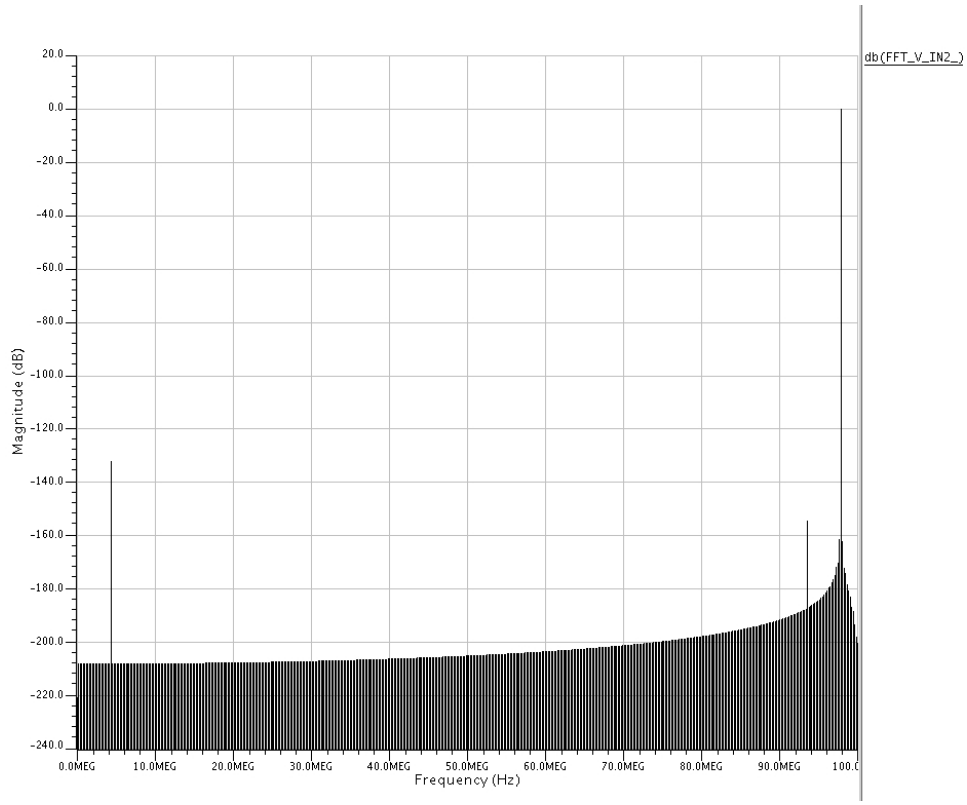


Figure C.7: FFT harmonics generated by a real RF-MIM capacitor model.

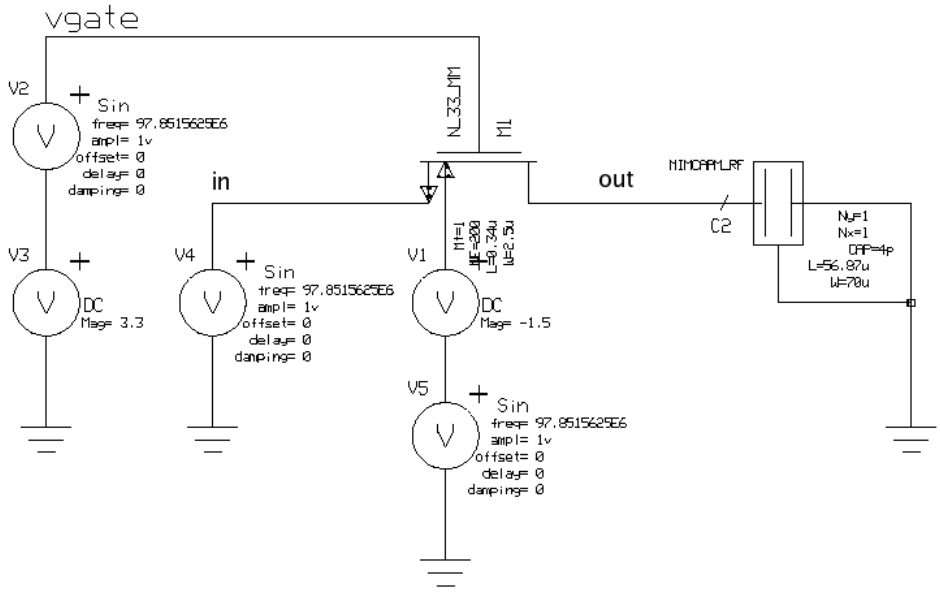


Figure C.8: Schematic circuit to test performance of an ideal bootstrap and body effect compensation tech-nique (the bottom-plate sampling circuit is not shown here because we are testing now tracking linearity).

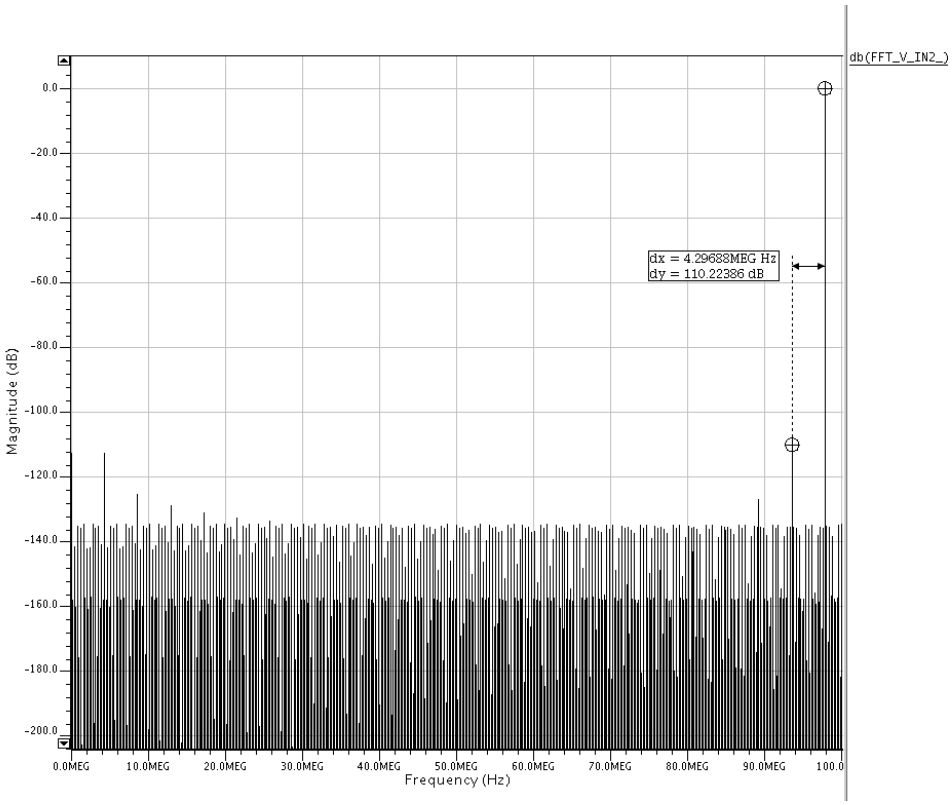


Figure C.9: FFT linearity simulation of circuit of fig.C.8.

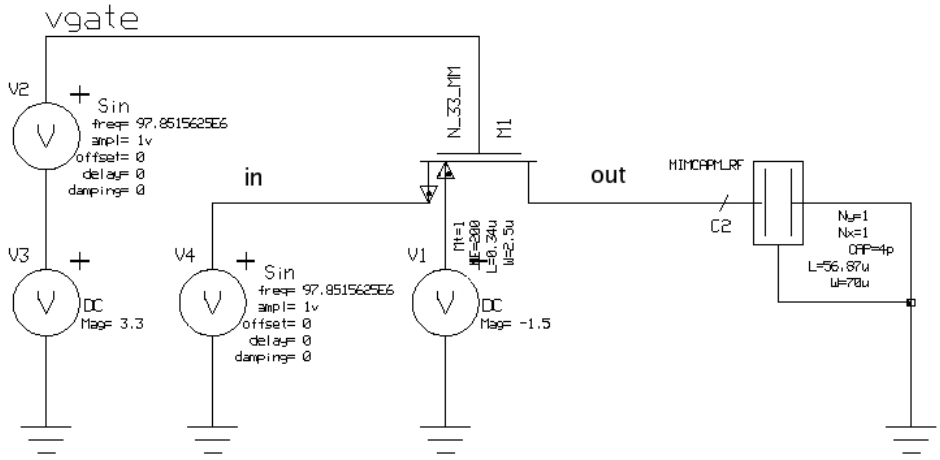


Figure C.10: Schematic circuit to test performance of an ideal bootstrap without the body effect compensation technique (the bottom-plate sampling circuit is not shown here for the same reason as in fig. C.8).

ground; but each signal is split into 2 symmetric signals. The symmetry axis is a fixed voltage (i.e. the ground voltage) and the difference of this two signals gives the original signal. This way no information is lost and the common mode input signal now is fixed and it's not varying like

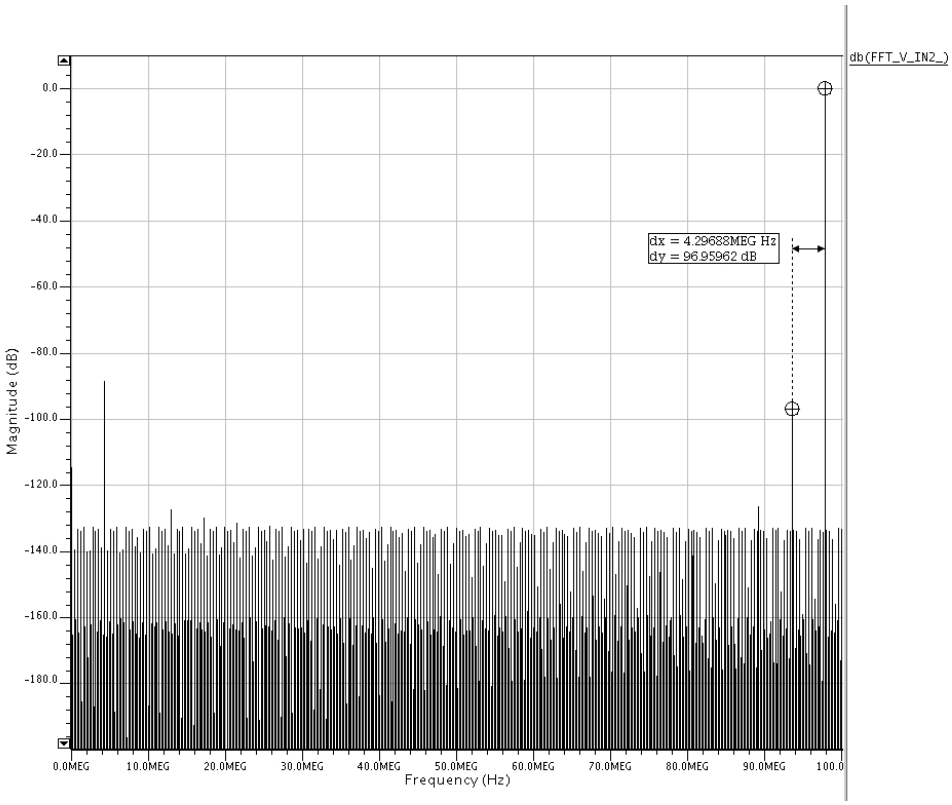


Figure C.11: FFT linearity simulation of circuit of fig.C.10.

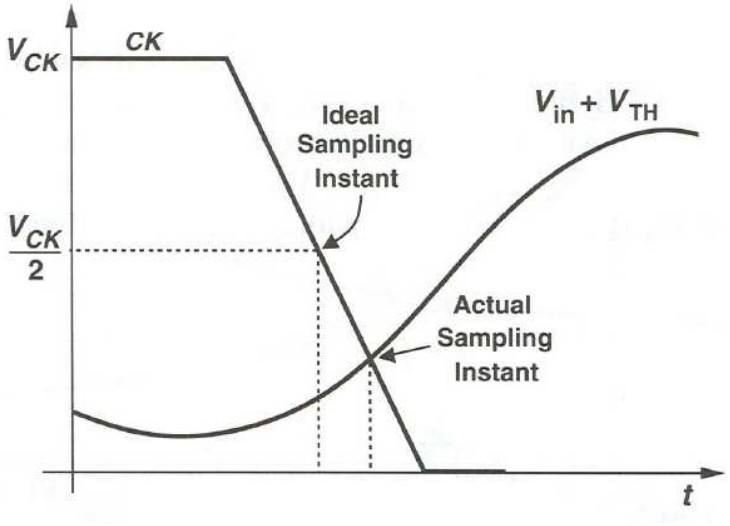


Figure C.12: Input signal induced Jitter noise. (picture from [75])

in the single-ended configuration. Next paragraph is going to analyze input stage signal to noise ratio performance in relation to different architectures in order to understand which one is more suitable for our project specifications.

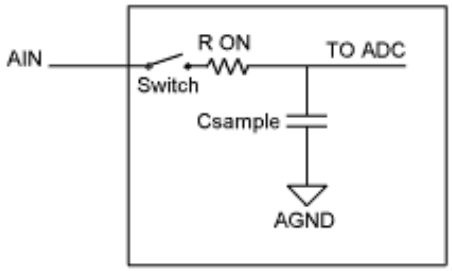


Figure C.13: Single ended T&H stage configuration.

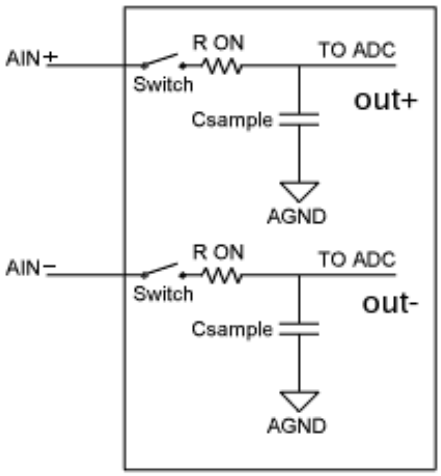


Figure C.14: Differential T&H stage configuration.

In single-ended architectures, given a signal of the form:

$$S_{single} = V_p cos(wt) \tag{C.3.17}$$

the thermal noise power N_{single} integrated on the $R_{on}C_{sample}$ pass bass filter band is:

$$N_{single} = \frac{kT}{C_{sample}} \tag{C.3.18}$$

where k is the Boltzmann constant and T the absolute temperature. The signal to thermal noise ratio SNRsingle is given by:

$$SNR_{single} = \frac{\frac{V_p^2}{2}}{\frac{kT}{C_{sample}}} = \frac{V_p^2 C_{sample}}{2kT} \tag{C.3.19}$$

In a differential configuration, if the voltage swing is kept constant on each input branch and equal in magnitude to the one allowed in the single-ended architecture as in eq. C.3.17, the effective input signal over the whole branches is:

$$S_{diff} = 2V_p cos(wt) \tag{C.3.20}$$

that is exactly the double of the single-ended case. The thermal noise power N_{diff} integrated on the two $R_{on}C_{sample}$ pass bass filter band is:

$$N_{diff} = \frac{2kT}{C_{sample}} \tag{C.3.21}$$

that is exactly the double of the single-ended case cause now we have 2 independent branches with two resistors. The signal to thermal noise ratio SNR_{diff} is given by:

$$SNR_{diff} = \frac{\frac{(2V_p)^2}{2}}{\frac{2kT}{C_{sample}}} = \frac{V_p^2 C_{sample}}{kT} = 2 \cdot SNR_{single} \tag{C.3.22}$$

As can be noted from the previous equation, the SNR of a differential configuration improves by 3dB compared to the one of a single ended one. Another advantage is that with a differential configuration also the linearity is improved cause the input common mode is fixed (this helps sampling devices) and all even distortion harmonics are avoided. In order to have a SNR compatible with B-bit of resolution, it must be:

$$\frac{2kT}{C_{sample}} < \frac{(\frac{4V_p}{2^{(B+1)}})^2}{12} \tag{C.3.23}$$

solving for C_{sample}, it can be obtained:

$$C_{sample} > \frac{3kT2^{(2B+1)}}{V_p^2} \tag{C.3.24}$$

the maximum current in each branch required to charge and discharge this capacitor at a frequency fmax using eq.C.3.24 is given by:

$$I_{max} = C_{sample}\frac{\partial\ V_p cos(2\pi f_{max}t)}{dt} = C_{sample}2\pi f_{max}V_p > \frac{3kT2^{2(B+1)}}{V_p} \tag{C.3.25}$$

From previous equations can be inferred that Csample and Imax are exponentially proportional to the resolution and inversely proportional to the voltage amplitude. If B=18 $Fmax = 100MHz$ and T=300Kelvin, than:

$$C_{sample} > \frac{1.708E-9}{V_p^2} \tag{C.3.26}$$

$$I_{max} > \frac{1.073}{V_p} \tag{C.3.27}$$

Using a 3.3V technology, as suggested by previous subsection, and allowing Vp to be as high as 1.4 V (almost rail to rail operation) the minimum value for C_{sample} and I_{max} are:

$$C_{sample} \approx 871pF \tag{C.3.28}$$

$$I_{max} \approx 766mA \tag{C.3.29}$$

These value of capacitance and current are very high and could be a problem to design an IC with required resolution able to drive such load. Next subsection is going to develop an architecture able to trade this requirements with area occupation through the use of parallel architectures.

C.4 Architectural topology

Is it possible to trade requirements of equations C.3.28 and C.3.29 with speed or area occupation through the use of parallelized structures ? Lots of parallelized architectures have been analyzed in order to achieve design performance gains. The simplest and more generic one is going to be studied. It is applicable to any type of passive elements sampling input stage converters. A generalized model is shown in picture of fig.C.15; this model consider both time (interleaving) and space (repetition of the same structure many times) parallelization.

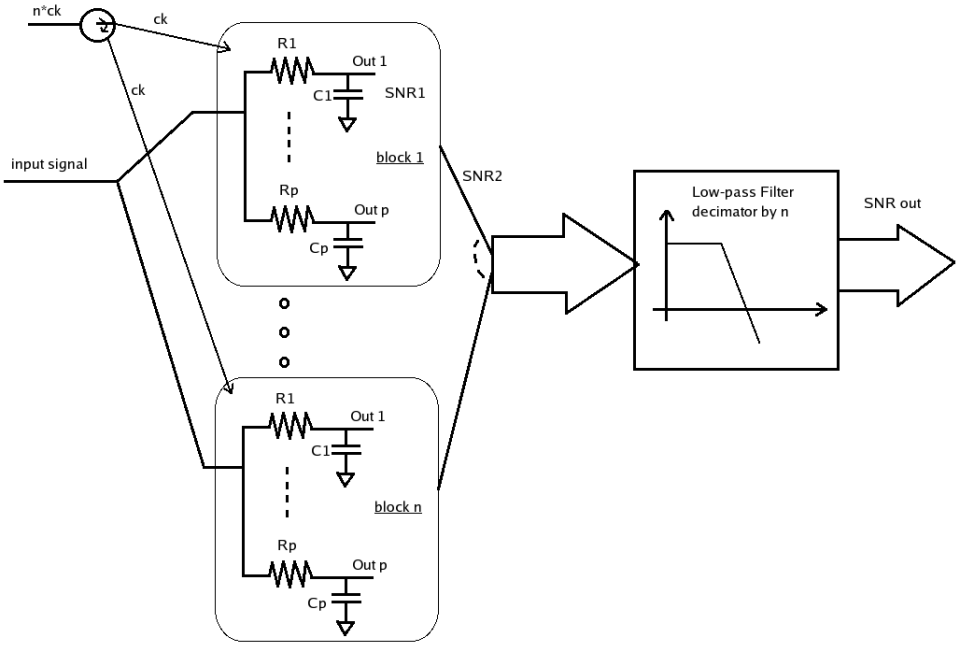


Figure C.15: Generalized parallel architecture with passive input sampling stages.

The generalized model works as follow: the input signal is sampled with a frequency f_{max} by n identical blocks. From one block to another the sampling instant is delayed by Ts/n where Ts is the sampling period. This cause the overall system to sample the input signal with a frequency n-times higher. Each block is composed by p identical passive input stages sampling the input signal at the same time. After being sampled, the signal is decimated and filtered by a low-pass filter in order to trade oversampling with resolution. This structure is going to be analyzed quantitatively in order to see if it is possible to obtain some benefits through parallelization.

In the overall following presentation we are implicitly referring to a differential architecture for benefits just raised. For simplicity all values of voltages and capacitances are referred to one of the two single branches of a differential configuration. If C1=C2=...=Cp=C, the signal to noise ratio in each branch (SNR1) is:

$$SNR1 = \frac{V_p^2 C}{kT} \tag{C.4.30}$$

Sampled data on each branch are going to be combined in order to achieve a higher resolution

averaging thermal noise. The overall block1 signal to noise ratio SNR2 is:

$$SNR2 = pSNR1 = \frac{pV_p^2C}{kT} \tag{C.4.31}$$

The signal to noise ratio SNRout after the low-pass decimator filter is:

$$SNR_{out} = nSNR2 = npSNR1 = \frac{npV_p^2C}{kT} \tag{C.4.32}$$

It's important to note that, each doubling of p or n increases SNR by 3 db and so it brings 0.5 extra bit of resolution. The total capacitive input load seen at the input every sampling instant is equal to pC, the SNR_{base} of the system without any type of parallelization is:

$$SNR_{base} = \frac{pV_p^2C}{kT} \tag{C.4.33}$$

that is exactly the same as eq. C.4.31, so from this point of view there are no particular advantages in parallelizing 'in space' sampling input branches inside each block. However there is an advantage in parallelizing 'in time' each sampling block by oversampling. This advantage brings a reduction by a factor n on the total input capacitive load for a fixed value of SNR. However, employing all the total on-chip capacitive load (equal to pnC) in only one S/H, the total SNR would be equal to the one of eq. C.4.32. From this point of view, no gain is obtained. However, if every block was able to sample the input signal with a frequency higher than clock's one, even the on chip total input capacitance would be reduced by a factor n. Picture C.16 shows this modified architecture. A practical implementation of conceptual diagrams shown before is the one of fig. C.17: This

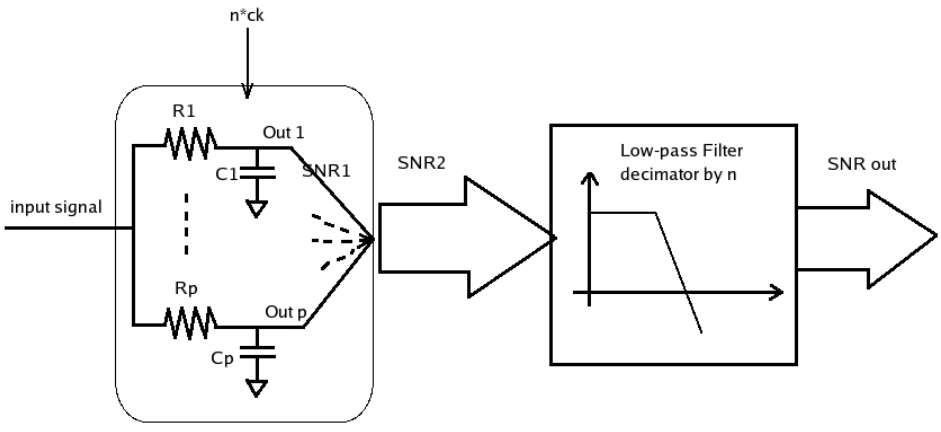


Figure C.16: Parallel architecture with passive input sampling stages and oversampling without interleaving.

architecture trades both sampling speed and area occupation with resolution. The signal here is sampled by k identical blocks. The introduction of oversampling is due to the A/D conversion time requirements in converting the sampled data. Each block samples and converts the input signal with a frequency that is n/k*ck. The digital output signal of each block is than filtered and

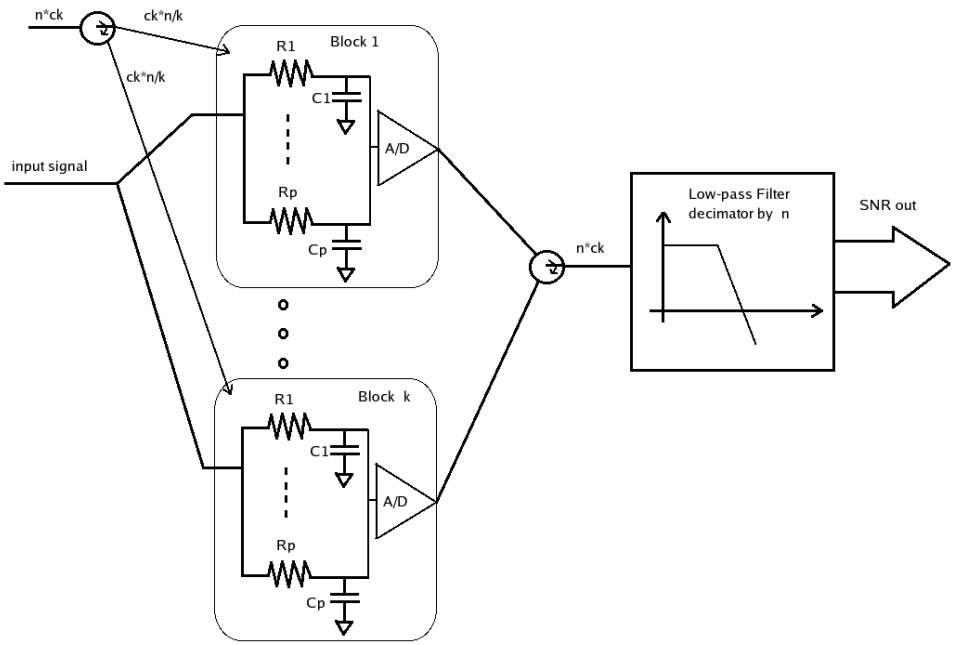


Figure C.17: Proposed architecture with passive input sampling stages.

decimated with a digital filter. Each block is composed by p identical sampling branches. Basing on fig. C.17, the capacitive input load C_{input} is given by:

$$C_{input} = pC \tag{C.4.34}$$

According with the previous analysis, C_{sample}, the equivalent input capacitance of a system without any sort of parallelization (i.e. oversampling and passive input branch stage) is given by :

$$C_{sample} = nCinput = npC \tag{C.4.35}$$

solving for C:

$$C = C_{sample}/np \tag{C.4.36}$$

substituting eq.C.3.24 into eq.C.4.36:

$$C > \frac{3kT2^{(2B+1)}}{npV_p^2} \tag{C.4.37}$$

Considering that basing on the previous analysis there's no gain in parallelizing input branches inside each block, are they essential in the architecture of fig. C.17 ? The gain arises from real devices limitations. To understand this, non-idealities of techniques used to obtain high linearity from passive input branches are going to be greatly increased. In addition to that, parasitic capacitive coupling have been added to the circuit. The resulting sampling input branch network is shown in fig. C.18. As it is possible to see from the previous picture, the gate-source bootstrap voltage technique is absolutely not present, furthermore, additional 2 parasitic capacitors Cj and

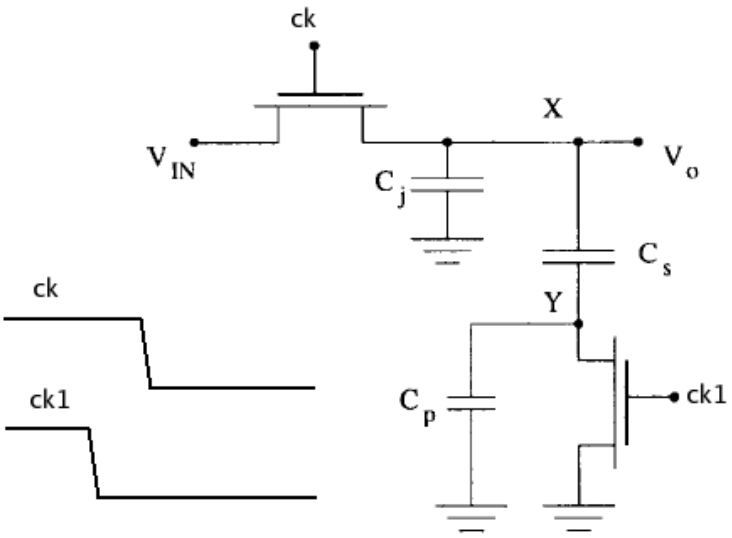


Figure C.18: Input sampling branch (unreal worst case).

Cp are present. Solving the network of fig. C.18 using Volterra series analysis in order to obtain HD3 (the ratio between the third distortion harmonic and the fundamental) [40] it can be obtained:

$$HD3 = \frac{C_s}{(C_s + C_p)} \frac{V_p^2}{4} jw \frac{(C_s + C_j)}{k_1 (V_G - Vt)^3} + \frac{C_p}{(C_s + C_p)} \frac{3(V_p w T_f)^2}{32 V_G^2} \tag{C.4.38}$$

where $V_p cos(wt)$ is the input signal amplitude, V_G is the voltage applied to the gate of the input transistor, T_f the clock falling time and k_1 contains MOS parameters: $K_1 = \mu_n C_{ox} W/L$. Considering the falling time to be very little, C_p and $C_j \ll C_s$ and V_G approximately equal to V_{gs}, HD3 can be approximated as:

$$HD3 = \frac{V_p^2 w C_s}{4(V_g s - Vt)^3 k_1} \tag{C.4.39}$$

The denominator of eq. C.4.39 for technological reasons can't be more than a certain constant A. In addition to that Vp can't be smaller than another constant B for current and area requirements. Basing on this assumptions, HD3 is limited to be no more that:

$$HD3 \approx \frac{BwC_s}{A} \tag{C.4.40}$$

Eq. C.4.37 relates C value with resolution requirements. Thinking on fig. C.18 like a branch of a differential architecture, than C=Cs. Substituting eq.C.4.37 into eq. C.4.40:

$$HD3 \approx \frac{3kT2^{(2B+1)}Bw}{npV_p^2 A} \tag{C.4.41}$$

from the previous equation can be noted that the designer can reach the desired value of distortion for a fixed input signal level, oversampling ratio, and resolution, simply increasing p. In relation the desired target, equation C.4.41 expresses the concept that, even if techniques to increase sampling switch linearity sometimes are not effective in reaching our desired target, putting together many

sampling systems of that kind in parallel can improve those techniques and so the achieved linearity. As shown in figure C.9, if sampling switch linearization techniques are used, it is possible to achieve 110 dB of linearity with a capacitance of 4pF. The architecture shown in figure C.17 solves problems related to sampling speed and thermal noise reduction trading these performances with area occupation. This can allow us to achieve the desired speed and resolution performances. Simulation of figure C.9 assumes that all these linearisation techniques are ideal. Simulations of the non-ideal behaviour of these have been performed using the circuit of figure C.19. This

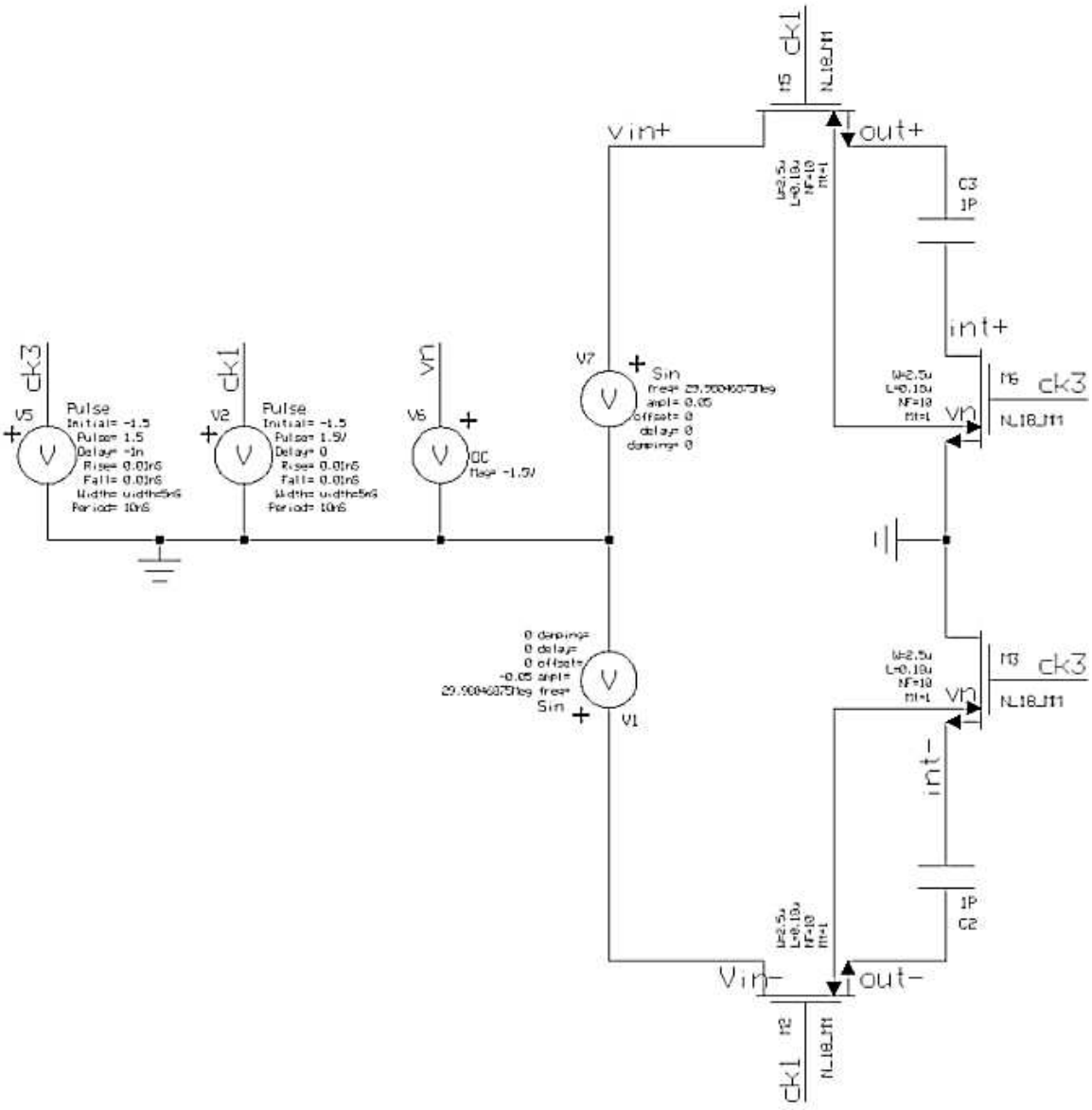


Figure C.19: Non-ideal behavior of sampling device linearisation techniques simulation circuit.

circuit simulates gate voltage bootstrap and body effect compensation techniques non-idealities expressed in an oscillation from the ideal value of 50mV. Taking advantage of previous derivations,

these problems can be solved reducing device and capacitance size allowing third order distortion harmonic to be approximately -120dB. Picture of figure C.20 shows the simulation output. Another advantage of this shrinking is that the more sampling MOS is little, the more its linearisation techniques are close to the ideal case. MOS capacitance variations infact and its parasitic couplings are reduced and so they influence less linearising circuits.

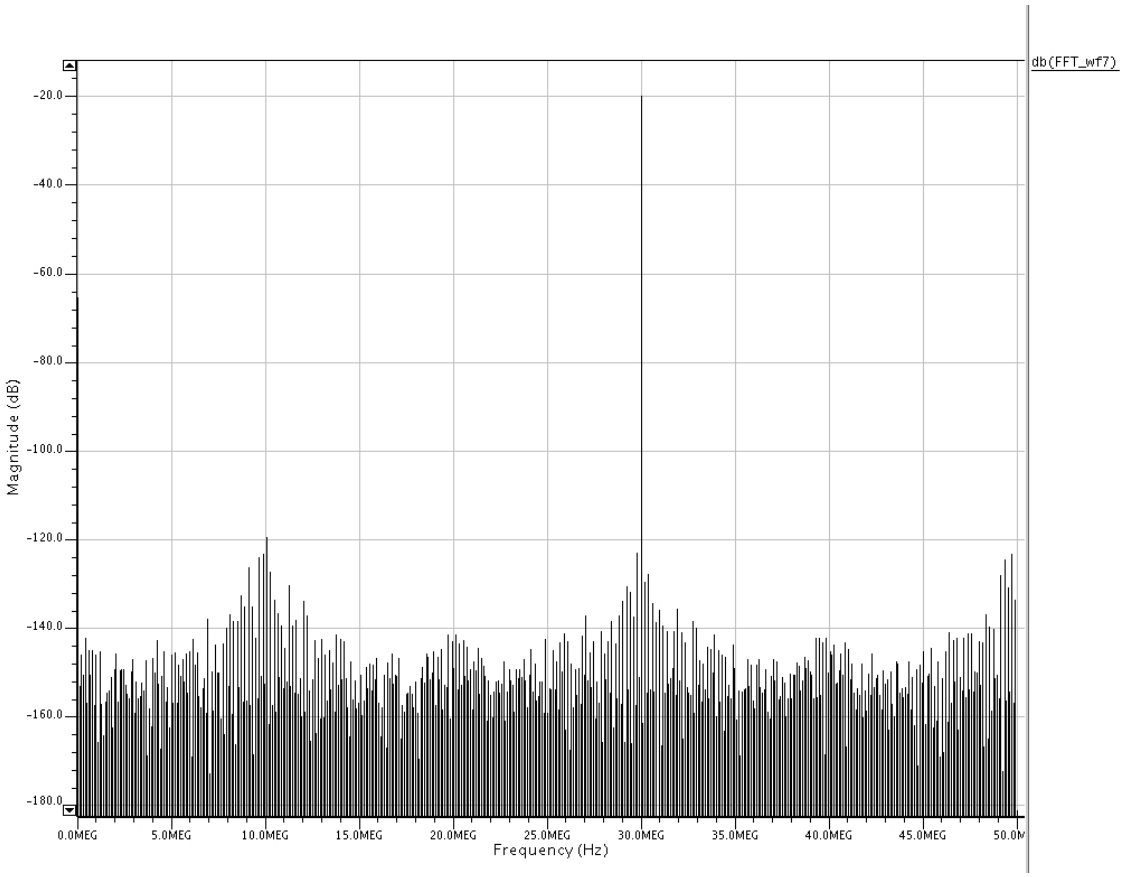


Figure C.20: FFT analysis of fig. 15 circuit output.