論文 / 著書情報
Article / Book Information

題目(和文)	A/D変換器の特性解析手法
Title(English)	Characterization and analysis methods of analog-to-digital converters
著者(和文)	源代裕治
Author(English)	Yuji Gendai
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第8715号, 授与年月日:2012年3月26日, 学位の種別:課程博士, 審査員:松澤昭
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第8715号, Conferred date:2012/3/26, Degree Type:Course doctor, Examiner:Akira Matsuzawa
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Characterization and Analysis Methods of Analog-to-Digital Converters

A Doctoral Dissertation by

Yuji Gendai
Department of Physical Electronics
Tokyo Institute of Technology

in partial fulfillment of the requirements for the degree of
Doctor of Engineering

March 12, 2012

To my wife, Kikuko

Acknowledgment

This thesis stands on generous help and support that I have received over the years. I would like to take this chance to acknowledge the many people who have helped me throughout my doctor course.

I am grateful to Prof. Akira Matsuzawa, my principal advisor who has made all of this work possible. He gave me a chance to take part in the laboratory and to accomplish my research. His wisdom and expertise enabled the successful completion of this research. I have learned up-to-date converter techniques, the way to present, the orientations of the research, the approaches to the goal, and lots more. Vague ideas that I held previously come to clear results during the lab years.

I am obliged to Assoc. Prof. Kenichi Okada, my associate advisor. His advices are often to the point. I am frequently inspired from his comments.

I would like to thank my doctor committee: Prof. Kazuya Masu, Prof. Kiyomichi Araki, Prof. Shigetaka Takagi, Visiting Prof. Toshihiko Mori, and Assoc. Prof. Kenichi Ohhata. I appreciate all attentions on my thesis, in spite of their busy schedules. Their questions and comments gave me new insights of my work and helped to improve the quality of my thesis.

I would like to appreciate Asst. Prof. Masaya Miyahara for his helps and advices. I also appreciate Hironori Sakaguchi for his support of CAD tools.

I would like to thank to Yoshino Kasuga and Makiko Tsunashima for handling lots of official works.

I would like to thank to every one at Matsuzawa-Okada lab for all supports and friendship. I enjoyed my time in the lab with you.

I would like to thank to Sony Corporation, chiefs and colleagues. I am trained for converter design in Sony. Works and discussions with Yoji Yoshii, Yoshihiro Komatsu, Masoto Kawata, Kazuhisa Nojima and Norio Shoji have been fruitful. We have tried lots of ideas in developing products. The evaluations also contributed the progress of the design. Those experiences and thoughts formed my theory bases.

I am grateful to my lovely family Kikuko and Hirotaro. Their supports and encouragements are the essential part of my life and research.

Abstract

This thesis deals with the characterization of Analog-to-Digital converters (ADCs). Unlike testing of ADC, whose interests are in selecting good dies with the minimum cost, or unlike the ADC standards, whose interests are in defining the compatible and exchangeable specifications, this thesis analyzes the intrinsic properties of ADC aiming at the design improvements.

Conventionally, an ADC is evaluated by linearity and distortion. Linearity is sensitive as it is, because it can relate the internal module with measured results. Thus the study on it goes to the finer behavior around the code transition level (CT level). Distortion measurement usually does not have such resolutions. The characterization for design requires to obtain the deviation at each conversion for any input. Usual test signals are ramp signals for linearity and sinusoidal signals for distortion. The question is if they are sufficient to characterize an ADC, and if not, how we can obtain deviations for other signals.

In this thesis, the linearity analysis is formulated as the maximum likelihood (ML) estimation problem for a rising ramp input. This framework displays the statistical nature of A-to-D conversion. One result is an explicit formula of the input equivalent noise. The basic tools for distortion analysis are the powerful waveform reconstruction technique that I named “Euclidean reordering” and the difference wave analysis method. Applying the methods into flash ADCs, I have newly identified a distortion pattern that I named “dog-tooth.” A low power bipolar latch design also demonstrates the capability of the methods.

In order to get the deviation for each conversion, we first define the equivalence of signals. Then we need a practical method to estimate the ideal value for the conversion. This procedure is performed in the least mean square basis using convolution methods. I propose two candidates for composite test signals, the “two-tone signal” and the “duplex sine signal.” They characterize ADCs involving the frequency responses that monotone sinusoidal waves cannot.

Tools developed in this thesis are powerful by themselves. The combination of them is not only possible but also necessary for more precise characterization of ADCs. But it remains for the future study.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The noble mission of Analog-to-Digital converter (ADC) is to digitize a signal from its continuous form into a series of discrete numbers. The discrete form of data is processed in digital domain. According to the semiconductor technology progress, relative cost for digital processing becomes lower and lower, so significant part of signal processing has been executed in digital domain. The digital signal processing is an important part of various modern electronic systems such as communication, broadcasting, storage, radar, measurement, and etc. The role of ADC becomes more important since digital signal processing is necessity rather than desirableness, while most signal sources arise in continuous form.

As the range of the applications extends, the evaluation techniques have also made progress. There are lots of papers, articles and books on the subject, *e.g.* [1] [2] [3]. Several standards are available now, *e.g.* IEEE 1057 “IEEE Standard for Digitizing Waveform Recorders” [4], IEEE 1241 “IEEE Standard for Terminology and Test Methods for Analog-to-Digital Converters” [5], and IEC 60748-4 “Semiconductor Devices - Integrated circuits - Part 4-3: Interface integrated circuits - Dynamic criteria for analog-digital converters (ADC)” [6]. By now, many converter professionals deem the measurement methods are well-established.

On the other hand while designing fast ADCs, I felt insufficiency of standard measurement methods for ADC characterization. For example, how do we identify the cause of distortion from the signal-to-noise ratio? Or, how are transistor sizes of circuits optimized from the measurement? In the end, the purpose of standards is not to analyze ADC but to evaluate the ADC performance and to provide the compatible values for ADC users. ADC designers need more minute analyses than those which standards provide. My original motivation is to develop links from various evaluations to the individual design parameters in the circuits. Fig. 1.1 depects the situations a little bit generally. The purpose of the study is to enforce links between the design and the evaluation in both directions.

During the efforts, my focus shifted gradually on the characterization itself. Real ADCs

show various aspects in operation that seems out of the scope of common specifications. An anxiety is whether a pure sine wave that is commonly used as a test signal can fully describe the ADC characteristics. The intermodulation measurement using the two-tone test signal might be a remedy. But this evaluation is rarely used for ADC, at least in baseband applications. Further still the same anxiety remains if it is sufficient.

Viewing the ADC as a black box, analog input and the clock are the action to it. Digital output is the reaction. This input-output response reflects their natures stemming from their internal structures. Then what input and what analysis are appropriate to characterize ADC? This view point shown in Fig. 1.2 becomes the subject of thesis.

In this introductory chapter, we reconsider Analog-to-Digital (A-to-D) conversion process from its foundation. This procedure clarifies what we are doing in conventional measurement and provides us a universal view point of ADC characterization. We see a unique role of ramp and sine signals. Then we review the linearity evaluation using ramp signals and find the necessity of the extension. Next we examine the distortion evaluation using sine waves and clarify the problems. The quantitative contributions of definite causes of errors are evaluated in Section 1.4, which shows the sensibility differences of evaluation methods and the necessities of the finer resolution of analysis. The organization of the thesis is the last section in this chapter.

1.1 Decomposition of the A-to-D conversion process

The A-to-D conversion process consists of two elementary steps. One is scaling and the other is digitizing. This conceptual discrimination is helpful to clarify the subjects.

The scaling is related to the definition of "fidelity." Usually the continuous form of signal is called "analog" and the discrete form is called "digital," but in fact, both expressions should be analogous to the original signal in appropriate interpretations. Generally in signal processing, signals are analogous (or equivalent) if they overlap by three-parameter changes, *i.e.* the amplitude, the time origin and the level. In short, signals have three dimensions of freedom. The signal $v(t)$ is equivalent to the input signal $u(t)$, if the following equation holds.

$$v(t) = a\,u(t - t_0) + b\ . \tag{1.1}$$

The distortion is understood as the deviation of the output signal $w(t)$ from the best fit equivalent input signal $v(t)$.

In general conditions, we have no way to know the true input $u(t)$ so that the equivalent $v(t)$. Real systems for communication, storage or any other, make use of constraints for waveforms to recover the signal. In measurement, we have freedom to select a specific waveform for $u(t)$. Some choices make the ideal signal $v(t)$ estimation fairly easy. One choice is a ramp signal and another is a sine signal. For a line part of the ramp $u(t)$, any equivalent $v(t)$ also has a corresponding line

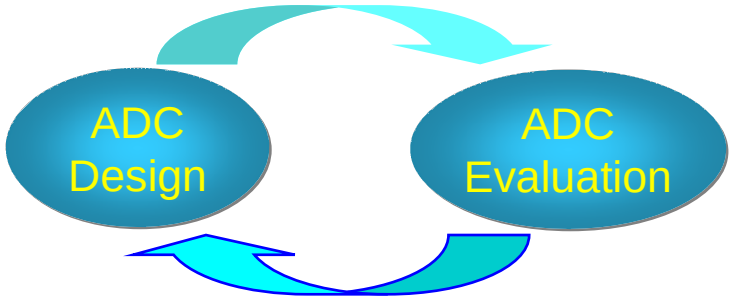

Figure 1.1: Design loop of ADC

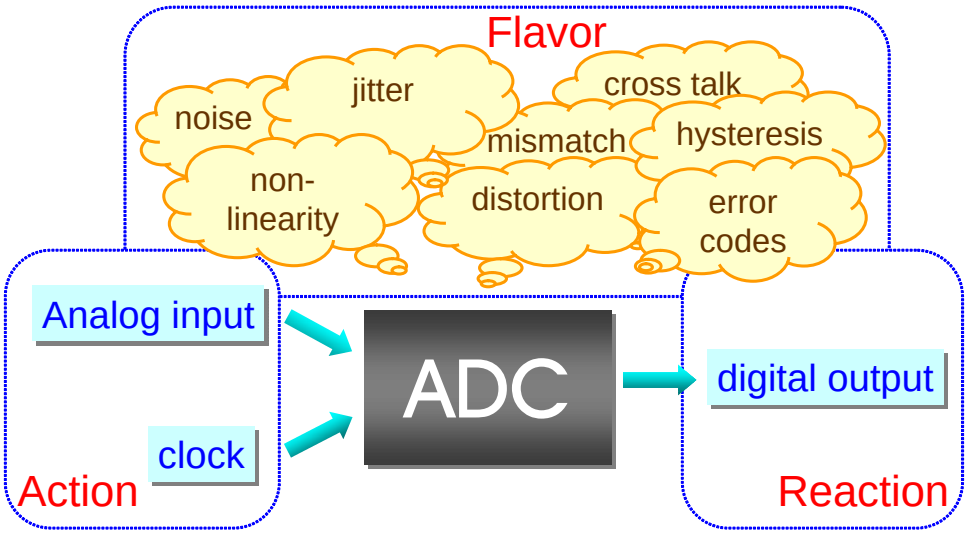

Figure 1.2: Subject of the study: What inputs and what analyses can reveal the ADC characteristics?

part. Thus a line fitting method works for the estimation. For a sine signal, equivalent signals are also a sinusoid with the same frequency. The three parameter estimation method [5] or the FFT method [7] takes the estimation role. The traditional test signals seem to be almost limited to these two choices.

While both signals provide the deviation from the ideal signal, two signals have been used to investigate different aspects of ADC. A ramp signal usually characterizes the linearity, in which deviations are grasped code by code. On the other hand, a sine signal characterizes dynamic performance at high frequency. The distortion is usually grabbed by its relative power.

A very important comment on equation (1.1), three parameters must be constant independent of the input signal $u(t)$. Traditional test methods seem indifferent to the point. The input amplitude is adjusted at each measurement [5, 5.5] and the offset is usually ignored. I devote Chapter 4 to provide a remedy.

The rest of the A-to-D conversion is the digitizing process. It has two dimensions. One is the time direction (sampling) and another is the value direction (quantizing). Both processes are closely related in signal processing, but ADCs treat them rather independently. Fig. 1.3 shows two processes. The performance index of the time direction is the maximum conversion rate. The sampling jitter is the main concern. The performance index of the value direction is the bit-width. Both sampling time and quantization levels are usually uniformly distributed.

The inevitable error of digitizing is called “quantization error.” The definition implies the difference between the continuous wave and the quantized wave in Fig. 1.3. As seen from the figure, the difference between ADC output data and the original wave, which is the definition of digitizing error, should be discriminated from quantization error in principle.

1.2 Linearity evaluation issues

This section depicts traditions on linearity measurement. We see a ramp signal plays a conceptual role as well as the practical importance in linearity measurement. Thus we comprehend the necessity of the expansion of linearity measurement.

Linearity of ADC is a character of its input-output transfer curve. An example of the ideal transfer curve of 3-bit ADC is shown in Fig. 1.4. Two reference voltages, V_{RB} and V_{RT} , are equally separated into 8 levels. The code transfer levels (CT levels) are placed at each center of the reference levels in this case. There is no code transition corresponding to the code 8, since it requires an extra (4-th) bit to express the code. In some converters, the overflow output indicates the CT level.

The nominal interval for one bit is called “LSB.” This term is probably ported from the computer term “least significant bit.” This term originally, with the opposing term the most significant bit or MSB, solely designates a bit position of binary code. In converter community,

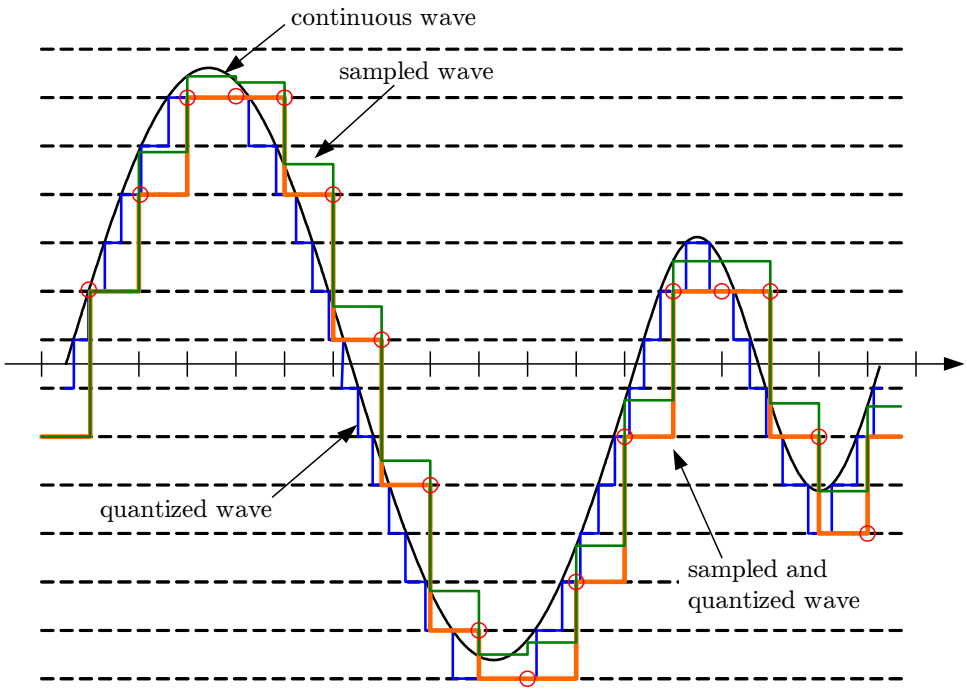

Figure 1.3: Sampling and quantization: Circles indicate the ADC output values

the LSB is used as a unit of the designed resolution of the ADC.

Using this term, the CT level in the transfer curve Fig. 1.4 has a half LSB offset from the reference voltage. This is not mandatory but merely a design matter. From this choice, the output code for the center voltage (V_{RB} +V_{RT}) / 2 becomes the center for the code 4, or the code 0 in two's complement format. This property is often preferable for many applications.

Fig. 1.5 shows a reference level design of this CT level definition. The external reference voltages are divided by internal resisters. The resistive voltage drop by parasitic resisters Rst, Rsb, and external PCB traces, can be compensated by Kelvin feedback [1]. The method senses the target voltages, V_{RT} and V_{RB} in this case, and forces V_{RTF} and V_{RTB} so that the sense voltages become the target voltages. Since the reference resisters can keep their regularity fairly easily within a chip, the reference voltages can be controlled precisely. The status of comparators forms "thermometer code." The comparator number 8 is optional for the overflow bit. It can be placed quite regularly in this design of reference levels.

The output of ADC is a digital code. Fig. 1.4 takes a straight binary code. Two's complement code is also appropriate, then the range of codes becomes -4 to 3. Gray code is still another can-

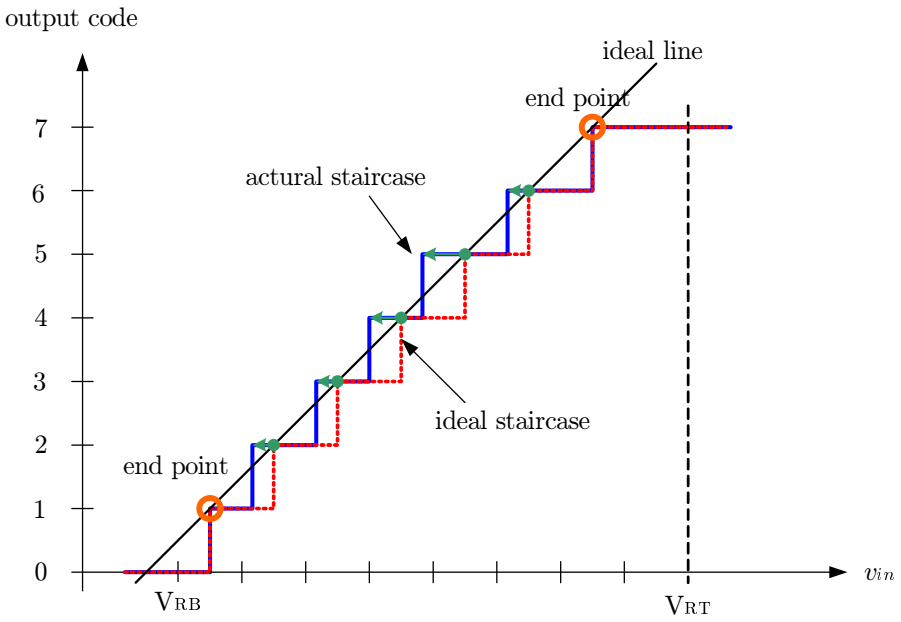

Figure 1.4: Input-output transfer curve

didate and sometimes used. Since the encoding is the back-end process of the A-to-D conversion, it has rather minor effects on the ADC characteristics.

Nonlinearity is a deviation from the ideal line. The problem is the best choice of the ideal line. There are several choices. To define the discussion, we consider the CT point of the CT level exists at the left end of the step. The first candidate is to connect both ends, which are the first CT point from 0 to 1 and the last CT point from 6 to 7 as circled in Fig. 1.4. This definition is called an 'end-point' calibration. Since two ends are obtained by one measurement, this becomes (probably) the most commonly used choice of the ideal line. IEEE Std. 1241 does not treat how to define the ideal line but solely used this definition. A weak point of the definition is there is no decisive reason to give the privilege to both ends. If the ADC is defective at either end, the ideal line will be inappropriate.

The best fit line determined by all CT points using the least mean square criterion is another candidate. This definition is robust for a few malfunctions of CT levels but requires more calculation than end-point calibration. In some applications, some of the CT levels are more important than the rest. Then the more weight should be placed on those important CT levels to determine the ideal line.

Following to the determination of the ideal line, one LSB is defined the distance between

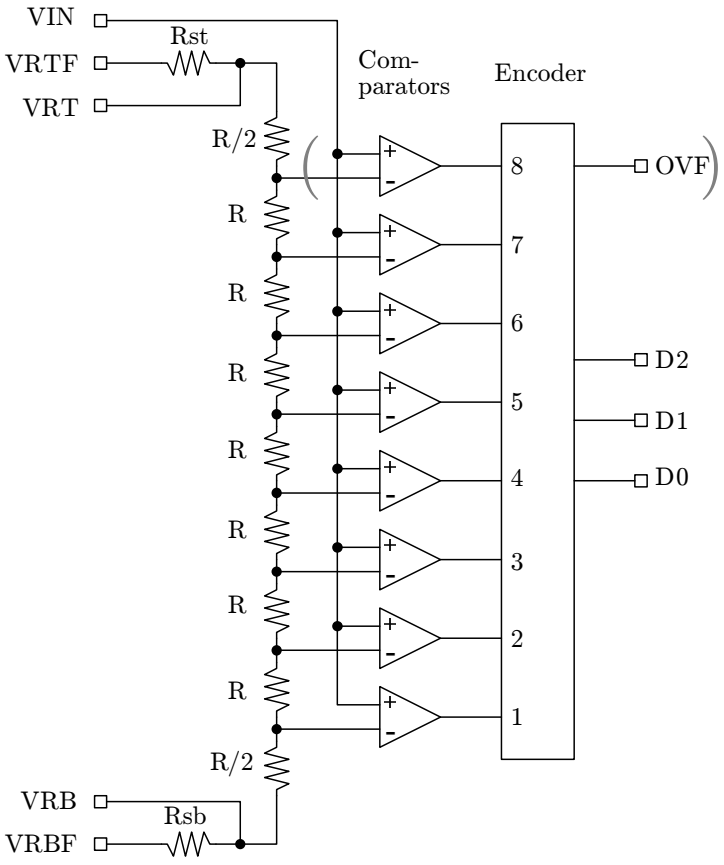

Figure 1.5: Reference level design of a 3-bit flash ADC. Stages (latches) are omitted. Real ADCs place several stages of latches after comparators.

codes, or equivalently as an inverse of the inclination. In end-point calibration,

$$\text{LSB size} = \frac{C_{2^n-1} - C_1}{2^n - 2} \tag{1.2}$$

where n is the bit width of the ADC and C_k is the CT level for the CT from $k-1$ to k.

The linearity error is defined as the distance between CT points and the ideal line. The distance can be measured for x-axis (V_{IN} direction) or y-axis (code direction). If we use the LSB unit for input, both measurements provide the same value. Many literatures, *e.g.* [1] [5] [8], do not care the error direction and do mixed treatments in figures and equations, presumably in this reason. However, because the errors should be evaluated in the converted output code, we

Table 1.1: Binary and Gray codes (3-bit example)

straight binary (decimal interpretation)	two's complement (decimal interpretation)	gray
000 (0)	100 (-4)	000
001 (1)	101 (-3)	001
010 (2)	110 (-2)	011
011 (3)	111 (-1)	010
100 (4)	000 (0)	110
101 (5)	001 (1)	111
110 (6)	010 (2)	101
111 (7)	011 (3)	100

had better deem the errors exist in V_{IN} direction both in theory and in practice. This conceptual difference brings the actual difference if we determine the ideal line using the least square method since the method treats two axis asymmetrically.

Arrows in Fig. 1.4 indicate the linearity error called the integral non-linearity (INL). This terminology should have a historical origin, but I have no source to identify it. We take the direction with the V_{IN} axis from the ideal line to CT levels. So all arrows in Fig. 1.4 aim at the negative direction. INL_1 is the INL for the CT level from 0 to 1. INL_7, or more generally INL_{2^n-1} for n-bit ADC is the INL for the last CT. If we adopt an end-point calibration, INL_{2^n-1} becomes 0.

The difference between adjacent INLs is called the "differential non-linearity" (DNL).

$$DNL_k = INL_{k+1} - INL_k \,. \tag{1.3}$$

The value coincides with the step width (code bin) of the transfer curve. Though INLs contain all information of non-linearity, DNL is defined separately probably because it has different nature form INL. In some applications such as visual signal processing, DNL is more important.

Fig. 1.6 shows a linearity plot of a real measurement. Linearity errors are obtained at each code like this figure. Sometimes the maximums and the minimums of DNL and INL are used as the representative values for linearity specification. But the point is, results for all CT levels are obtained during the processing of the data in linearity measurement. This property makes the linearity test very sensitive for ADC characterization.

There are two topologies for linearity measurement setup, the closed loop and the open loop. Historically, the closed loop integrating servo system in Fig. 1.7 was devised first.

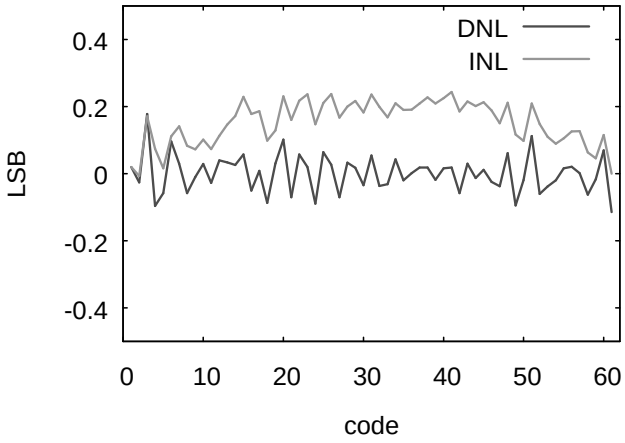

Figure 1.6: Linearity plot (DNL and INL) of 6-bit ADC

The ADC output is compared to the target code. The analog integrator circuit formed by an operational amplifier integrates the comparator output code density. The negative polarity of the feedback loop makes the ADC output codes converge around the target code. The digital voltage meter (DVM) measures the input voltage VIN corresponding to the target code. In a sense, this setup is the direct implementation of the input-output relation of the A-to-D conversion.

The system has problems in accuracy and in speed. The DVM can easily be quite precise, but the input VIN has ripples brought by the loop delay. The ripple effect to the accuracy is difficult to evaluate. The convergence time after the target code change and the measurement time of the DVM are often unendurable for mass production. More over, the clock frequency that the servo loop can operate may be much lower than the ADC conversion frequency. Then full speed linearity measurement is impossible.

In order to reduce the convergence time in servo system, the advanced closed loop system replaces the comparator and integrator into a more sophisticated digital controller block and a precise DAC as shown in Fig. 1.8 [9]. This configuration needs the DAC is much more precise (at least 2-bit, preferably more than 4-bit precise) than the ADC. This precision is achievable since the clock frequency of the DAC can be much slower than the ADC under the test.

The open loop measurement system takes the condition that the test signal is a line, a part of ramp signal or a triangular wave. The test bench is shown in Fig. 1.9. The open loop system is practical for low resolution and fast ADCs. The configuration is also an appropriate theoretical model of linearity measurement as we will see below.

In ramp method, one problem hidden in servo method has appeared. It is the noise of ADC.

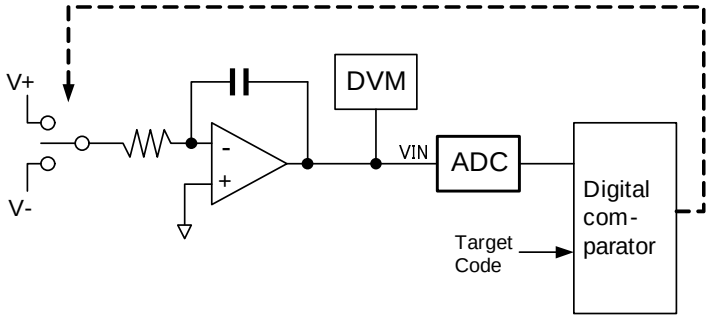

Figure 1.7: Closed loop linearity measurement servo system

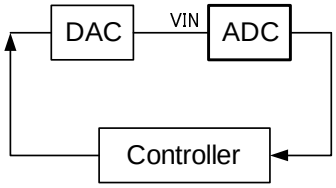

Figure 1.8: Closed loop linearity measurement digital loop system

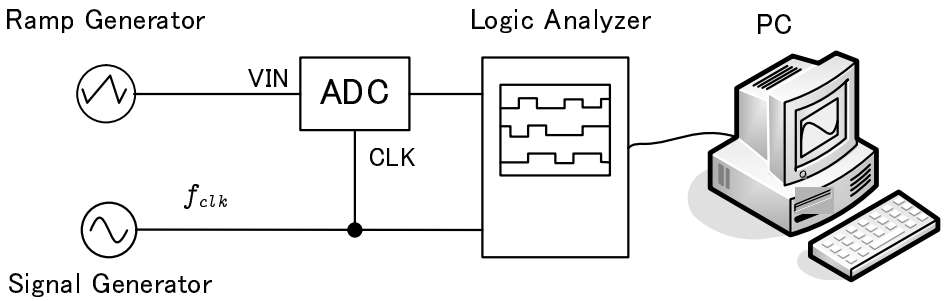

Figure 1.9: Open loop linearity measurement system: Ramp method

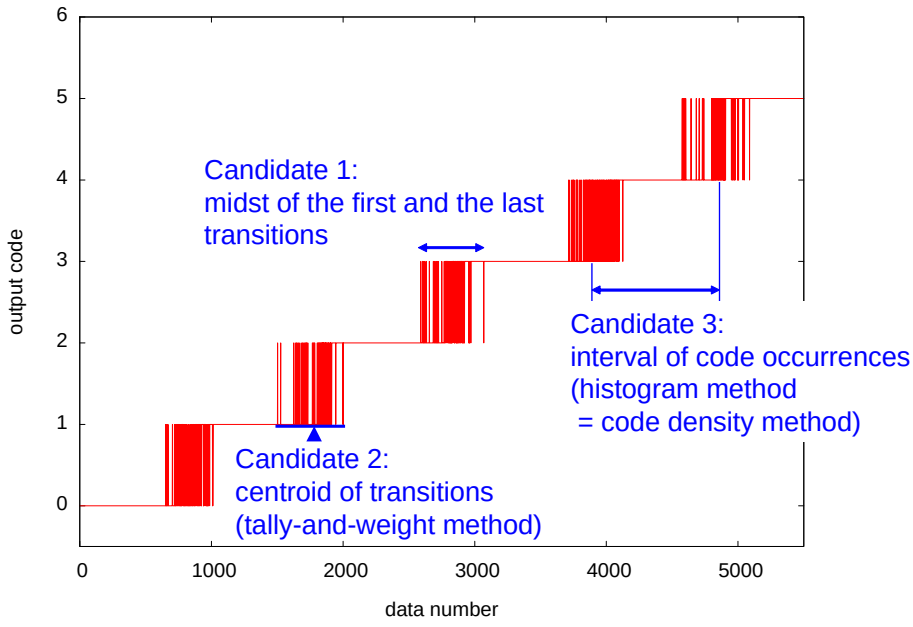

Figure 1.10: Vibrations at code transition: what position is appropriate as the transition level?

The real ADC does not have a one-to-one relation between input and output, but the output code fluctuates around the CT level like shown in Fig. 1.10. In servo method, the noise has automatically been averaged without noticing.

Some primitive ideas to determine the CT level are proposed.

1. midst of the first and the last transitions

2. centroid of transitions (known as “tally-and-weight method”)

3. interval of code occurrences (known as “histogram method” or “code density method”)

Along with the advanced methods trying to determine the CT level at the minimum sampling count, the problem becomes gradually been grasped as a statistical inference “CT level estimation” [9] [10] [11] [12] [13] [14] [15]. IEEE Std. 1241 [5] also takes this framework. The estimation stands on likelihood methods, which have much solider foundation than above primitive ideas. Chapter 2 enhances the methods.

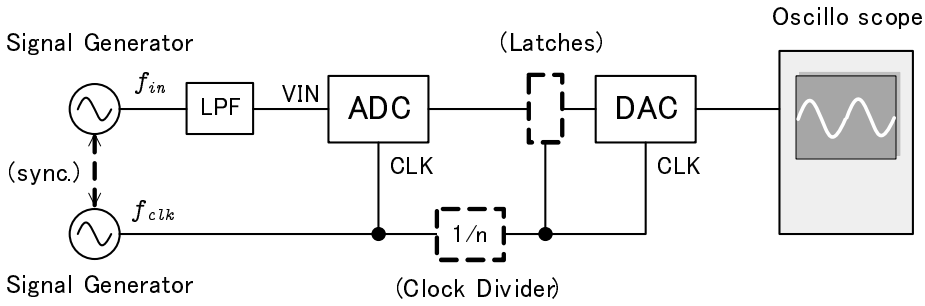

Figure 1.11: Back-to-back test setup for ADC functionality and distortion

The causes of the noise are various, some of them are externally and some of them are kick-back from output changes. Comparators in the ADC also have their intrinsic noise. We cannot eliminate the noise completely and the noise should be treated as one of characteristics of the ADC as well as DNL and INL. To keep the noise level within the system requirements is one important matter in ADC design.

The transfer function of ADC is conceptual because there is no one-to-one relationship between the input and the output. One reason is noise as just mentioned.

Another reason is the transfer curve dependency on the input slew rate. Any ADCs follow the input change with limited speed. If the input slew rate is not negligible to the tracking speed, the conversion results deviated from those of the DC input voltage. Even when the slew rate is rather low, it is possible the transfer curve for rising signal and that for falling signal are different.

The deviations depending on the input frequency have different mechanisms from those determining DC linearity. We can see them as the warping of the transfer curve or high frequency distortion. Anyway we want to characterize them separately. Later in Section 1.3, we see how much the transfer curve affects the distortion.

1.3 Distortion evaluation issues

We start with the history of the distortion measurement. Many parts of current ADC test methods seem still dragging history, rather than derived from principles. On early days, ADCs were usually evaluated by “back-to-back” fashion as shown in Fig. 1.11 [1]. The output of the test ADC under the test is connected to a DAC that is already known functional. This configuration is mainly for testing the functionality of the ADC rather than specify its performance numerically.

The input frequency f_{in} and the clock frequency f_{clk} are often selected as following:

$$f_{in} = \frac{f_{clk}}{n} + \alpha \tag{1.4}$$

where α is a small offset frequency and n is a positive integer. The number n is usually low order powers of two, such as 1, 2, or 4. Other ratios are possible but only simple fractions are practical. When $n = 1$, the condition is called "beat frequency test," because DAC output frequency becomes α instead of $f_{clk}+\alpha$ by aliasing of sampling. Since α is much lower frequency than $f_{clk} + \alpha$, the DAC operation is substantially relaxed. When $n = 2$, the condition is called "envelope test." The clock frequency to the DAC is often decimated by 1/2 at this condition, then the output becomes the frequency α and the DAC operation is further relaxed by the slower clock frequency.

The input does not have to be a sine wave but usually it is because (1) it was the only practical choice for high frequency signal source in the olden days, and (2) the beat wave can be measured by an audio distortion meter if it is a sine wave.

The offset frequency α can be positive and negative. Both selections produce much the same beat wave seemingly. Negative α had often been used in the past. This tradition would have its root in the sampling theorem. It state if the input frequency is lower than the half of the sampling frequency (Nyquist frequency) then the input continuous wave can be recovered perfectly by the sampled discrete data. However the theorem is irrelevant to the measurement. We only need a beat wave. Moreover at a negative α, the phase order of the beat wave is opposite to the original wave. In effect, time axis becomes opposite, from right to left. Falling side of the beat wave is the rising side of the original wave. This effect is noticeable if we can use asymmetric input such as sawtooth waves. Recently positive α is preferably chosen by these reasons.

The DAC output is a reconstructed waveform that is analogous to the ADC input signal and ideally which differs only in frequency and quantization errors. In reality, especially in classic ADCs, the beat waveform often shows a number of deficits of the ADC.

The photograph in Fig. 1.12 is an example of old measurement of an 8-b 300 MSps flash ADC designed in 1980's. The clock frequency f_{clk} is 200 MHz. The input frequency f_{in} is 24.99 MHz. The RF generator output is fed into the ADC analog input VIN through a low pass (or band pass) filter that cut all harmonic distortions of the source as in Fig. 1.11. That was the practical and effective way to ensure the purity of the input wave. The output is decimated by four and the resulting DAC output is the beat sine wave of 10 kHz. The distortion meter indicated -42.2 dB. At this purity, no distortion is noticeable in the displayed beat wave. Another wave in the photograph is the notch filter output from an audio distortion meter. By doing this I have expected the notched wave should magnify the distortion. But to my regret, it was hard to interpret what the waveform means.

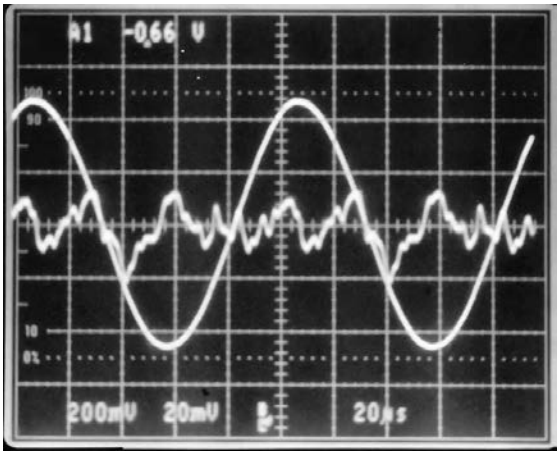

Figure 1.12: Beat wave and notched wave for 8-b 300 MSps flash ADC, f_{in} = 24.99 MHz, f_{clk} = 200 MSps, $\Delta f = -10$ kHz

Another example in Fig. 1.13 is clearer. The photograph is cited from [1]. Warping of 4-cycle ripples in the descending slope (the rising slope of the original wave) is noticeable in the wave. This is the typical distortion pattern for classical 8-bit flash ADCs, since the top-level layout of latched-comparator blocks is folded 4 times following the pioneer work [16]. This internal structure arises 4 cycles of ripples by two mechanisms. The analog input VIN line is separated into 8 lines. The voltage drops from top to bottom by currents for parasitic capacitor charge. The clock distributions are also divided into 8 lines. Then the sampling timing differs slightly from top to bottom. We can separate these causes by several measurements of various input frequencies and magnitudes. This quantitative separation of the distortion cause helps us to improve the ADC design. This is a typical application of ADC characterization.

Expanded view in Fig. 1.13 shows the degradation of DNL. A simple interpretation is the sampling delay variations of latched-comparators. Some variations are systematic that stems from the irregularity of the layout. They should be reduced though it might be difficult to eliminate them completely. The rests of variations are random. They are controlled by size of transistors. They also depend on circuit topologies. We must design the variations within the acceptable level. The reconstructed waveform provides valuable information for redesign.

Another type of error sometimes happens in beat wave as illustrated in Fig. 1.14. This error is called “sparkle.” It is natural and inevitable that sparkles appear at the highest edge of conversion rate, but the problem is it sometimes appears at much lower clock frequency than designated or expected from simulation. Some of sparkles happen regularly at specific positions of beat wave, but troublesome sparkles happen intermittently, say once a million clocks. This rate is called

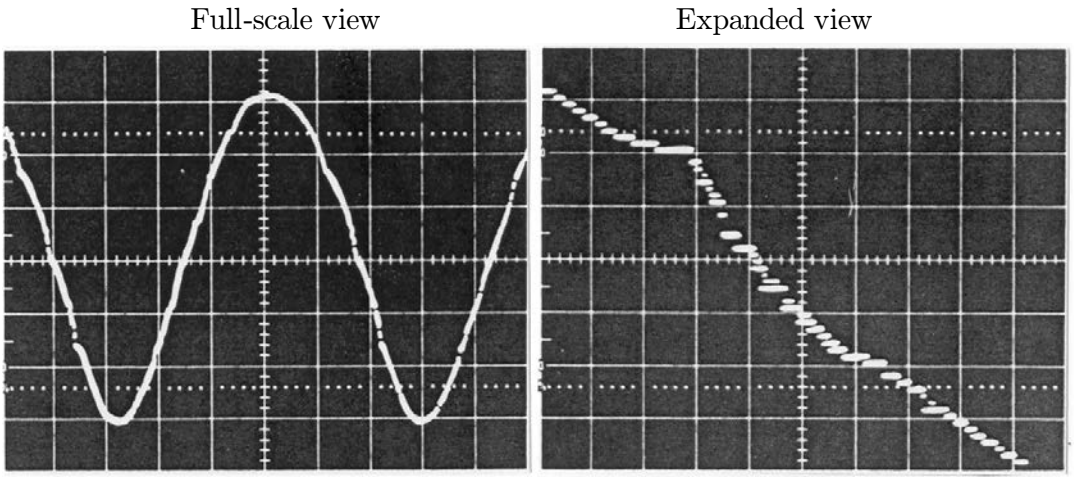

Figure 1.13: Beat wave of 8-b 20 MSps flash ADC, f_{in} = 19.98 MHz, f_{clk} = 20 MSps, $\alpha = -20$ kHz (cited from [1])

10^{-6} TPS (Times-Per-Samples).

Beat frequency test is handy to check the functionality of the ADC, but it is difficult to specify the ADC numerically. The audio distortion meter can provide the signal-to-noise ratio but the results rely on the DAC performance. Thus modern measurement systems replaced the DAC and the oscilloscope by a logic analyzer and a PC system as shown in Fig. 1.15. The audio distortion mater was replaced by an FFT program. That means the post evaluation is done in digital domain. The output signal does not have to be low frequency that relaxes the input frequency condition from (1.4). On the other hand, the intuitive views that beat waves presented had been lost in the frequency domain analysis.

Fig. 1.16 shows a possible ADC output sequence. This is an artificial example, not measured one. I will show later how it is made. There are 1024 samples, where relative frequency f_{clk} = 1024 and f_{in} = 275. I will reveal other input parameters later. The output code ranges from 0 to 63, which simulates an ideal 6-bit ADC. In this high input frequency, we see rather random patterns in the output sequence. If we connect sampled points, which imitates the DAC output in traditional measurement, the situation looks worse.

A First Fourier Transform (FFT) procedure provides a clear picture of the distortion. Fig. 1.18 shows the power spectrum of the data Fig. 1.16. The abscissa is frequency relative to the sampling clock frequency 1024. The spectrum of the fundamental tone is found at 275 as generated. The ordinate is the relative power putting the fundamental tone as 0 dB. The second harmonics appears at 474. This is the aliased frequency $1024 - 275 \times 2$. The minus sign of the calculation

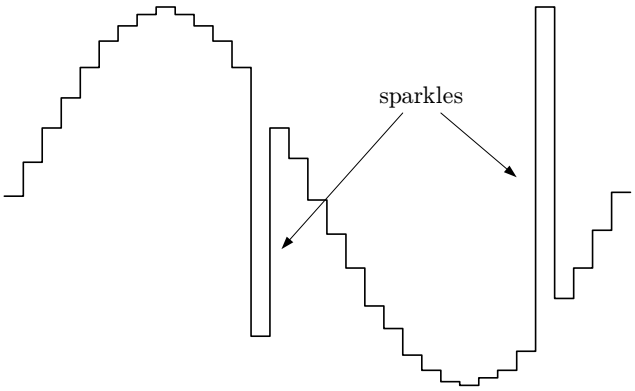

Figure 1.14: An illustration of sparkle errors

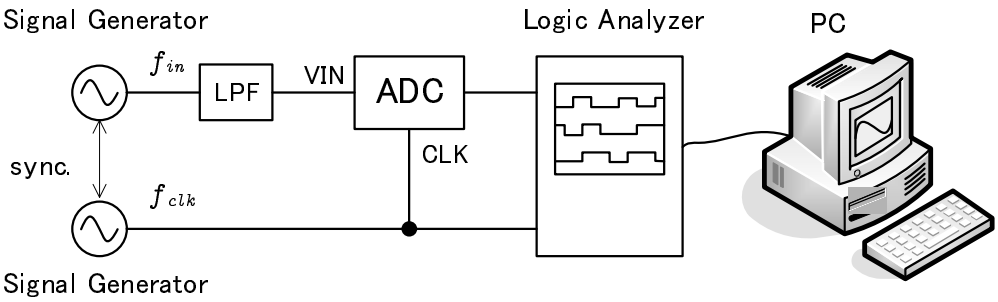

Figure 1.15: Modern distortion measurement system

means that the frequency is phase reversed in fact. The third harmonics appears at $199 = 1024 - 275 \times 3$ and the firth at $76 = 275 \times 4 - 1024$. The powers of harmonics are -43.668 dB (2nd), -42.974 dB (3rd), -45.728 dB (4th) and -44.182 dB (5th), respectively.

Several typical values are obtained from FFT results. To simplify definitions, we denote the power of k-th harmonics as $p(k)$. This is the square of the amplitude of the spectrum. In this definition, the powers of the fundamental tone and the second harmonics are $p(1)$ and $p(2)$ respectively. The index does not correspond to the frequency directly. Only the DC offset power $p(0)$ is independent of the fundamental tone frequency. Take the number N so that $\{p(k)|0 \leq k \leq N\}$ cover all spectral components.

The following definitions try to accord IEEE Std. 1241-2010 [5], but they look simpler mostly

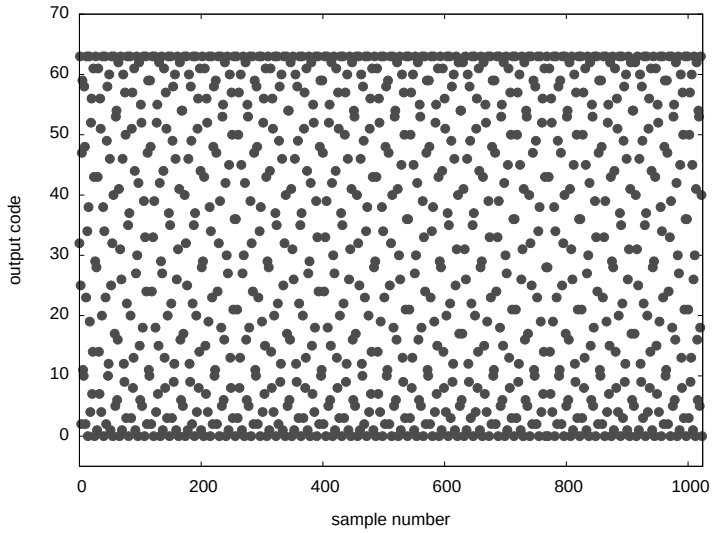

Figure 1.16: An artificial example of the outputs of 6-bit flash ADC

because we use the harmonic order instead of the frequency order.[1]

Signal-to-noise-and-distortion ratio (SINAD or SNDR) This is the ratio of signal power $p(1)$ to noise and distortion (NAD) power.

$$\text{SINAD} = 10 \log_{10} \frac{p(1)}{\sum_{k=2}^{N} p(k)} \tag{1.5}$$

where the denominator is the NAD power.

Total harmonic distortion(THD) This is the ratio of signal power $p(1)$ and low order harmonics.

$$\text{THD} = 10 \log_{10} \frac{p(1)}{\sum_{k=2}^{n} p(k)} \tag{1.6}$$

where n varies depending on specifications or vendors from 5 in [1] to 10 in [5].

[1]There are several variations in specifications commonly used. Moreover, traditional terms are a little bit confusing. 'Signal' in definitions means fundamental tone, 'harmonic' in harmonic distortion means 'low' order harmonics, 'noise' and 'spurious' usually have different meanings but are identical as spectral components.

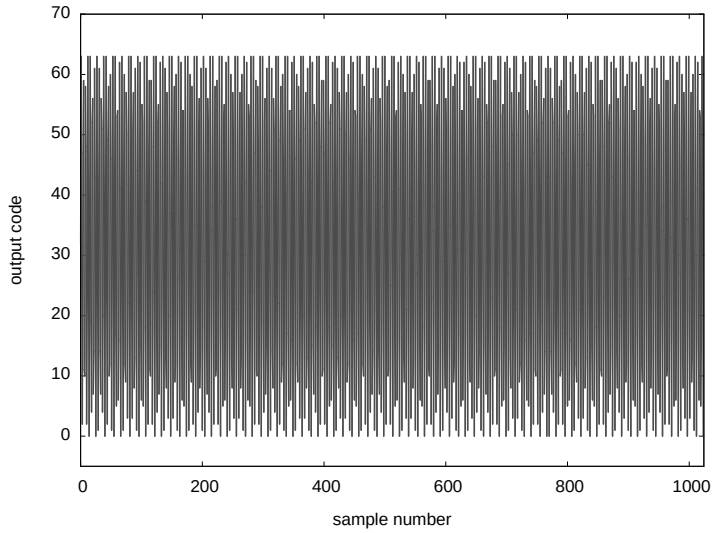

Figure 1.17: An artificial example of the outputs of 6-bit flash ADC, the same data with Fig. 1.16 except they are connected

Signal-to-noise ratio (SNR) . This is the ratio of signal power $p(1)$ and the spectral components other than harmonic distortions.

$$\mathrm{SNR} = 10 \log_{10} \frac{p(1)}{\sum_{k=n+1}^{N} p(k)} \tag{1.7}$$

where n coincides with the THD definition.

Spurious-free dynamic range(SFDR) This is the ratio of signal power $p(1)$ and the largest other harmonic component.

$$\mathrm{SFDR} = 10 \log_{10} \frac{p(1)}{\max_{k \in [2,N]} p(k)} \tag{1.8}$$

The choice of n in above definitions usually does not change values too much because spectra from 6 to 10 are rather small relative to other terms.

One noticeable feature that all representative value definitions in frequency domain have is that the DC level $p(0)$ does not appear. This is probably because signals are traditionally AC

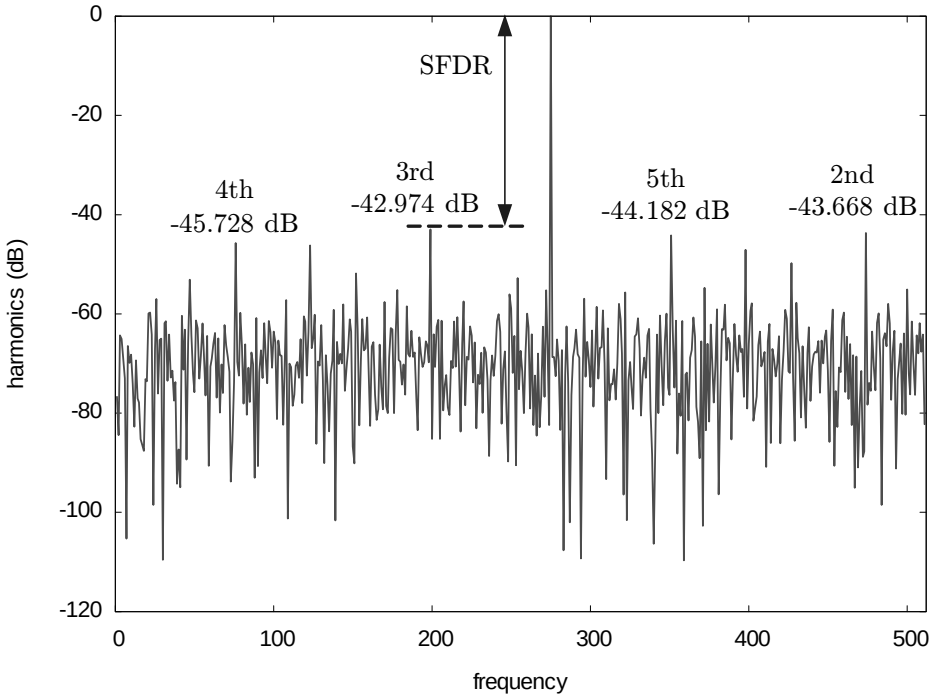

Figure 1.18: Power spectrum of data in Fig. 1.16

coupled fluently in the flow. This is pertinent to monotone signals, but the problem is how much extent this is appropriate for practical input waveforms. If the ADC has varying DC offset depending on the frequency, this ignorance may arise the under estimation of the distortion. I say it as the zeroth order distortion. In the other words, this is the fluctuation of the DC level parameter b in (1.1) depending on the input signal $u(t)$.

Another feature is slightly difficult to notice that they are macro indexes. The values do not correspond to individual codes or samples. From this feature the distortion indexes have less resolution than linearity in ADC analyses. We will examine the property quantitatively in next section.

By these definitions the spectrum in Fig. 1.18 is represented as follows.

SINAD	34.349 dB	
THD	36.541 dB	(38.005 dB)
SNR	38.369 dB	(36.369 dB)
SFDR	42.974 dB	

Parenthetic numbers in THD and SNR indicate the value when $n = 5$ in the definitions while the preceding numbers use $n = 10$. The obtained precision is far beyond the back-to-back measurement system using DAC in Fig. 1.11. This is the conclusive advantage of the digital signal processing. However, the spectrum might have lost a sign of internal deficits of ADCs from its macro nature.

The power spectrum Fig. 1.18 looks very common in real measurement. The fact is it is a quite artificial clipped sinusoid shown in Fig. 1.19. The clipping induces odd number harmonics. The wave also has a small offset in order to induce the second harmonics. This simple distortion is hidden from the ADC outputs in Fig. 1.16 by the intentionally selected sampling ratio. This example demonstrates the deficiency of frequency domain analysis for characterization of distortion. Time domain analysis has more resolution and complements the deficits of frequency domain analysis.

Two difference waves are shown in Fig. 1.19. Both are magnified by 8 for visibility. One is the deviation from the base sinusoid by offsetting and clipping. Another is the distortion given as the residue from the fundamental tone calculated by FFT. The distortion wave has a spectral component of the fundamental tone because the amplitude of the fundamental tone differs from the original sinusoid by clipping. We can say this is the first order harmonic distortion. There is another way of the first order distortion appearance as phase shift. The first order distortion does not matter if the input is sinusoid or alike. However it may matter for real signals, especially if the signal spectrum covers over the ADC input frequency range. In other words, they are fluctuation of parameters a and t_0 in (1.1). We will deal with the methods to analyze them in Chapter 4.

The real measurement also emphasizes the importance of time domain analysis. Fig. 1.20 shows a measured data of a 10-b 75 MSps ADC proto type. This ADC suffers from insufficient performance which SINAD shows only 40.81 dB at 15 MHz input. Since odd number harmonics are dominant, we can guess the clipping like distortion. But it is difficult to say how it distorts the signal from this frequency domain results.

Fig. 1.21 shows the reconstructed waveform from the same measurement in Fig. 1.20. The reconstruction method makes use of a special relation between the input frequency and the clock frequency. It is the substantial extension of the beat condition (1.4) that provides the most flexible frequency setting. The funny numbers reflect the constraint. We will deal with unabridged theory in Chapter 3. We can see the reconstructed waveform as if it is sampled at quite time resolution in one input period. Fig. 1.21 also shows the "difference wave" between the reconstructed waveform and the base tone calculated by FFT from the waveform.[2]

The base tone is nearly invisible since the reconstructed wave is overlapped. The difference wave is magnified by eight for visibility. The distortions over 10 LSB are recognized around the

[2]The adjective 'residual' used in [17] might be more natural than 'difference,' but I use the latter in this thesis according to my convention.

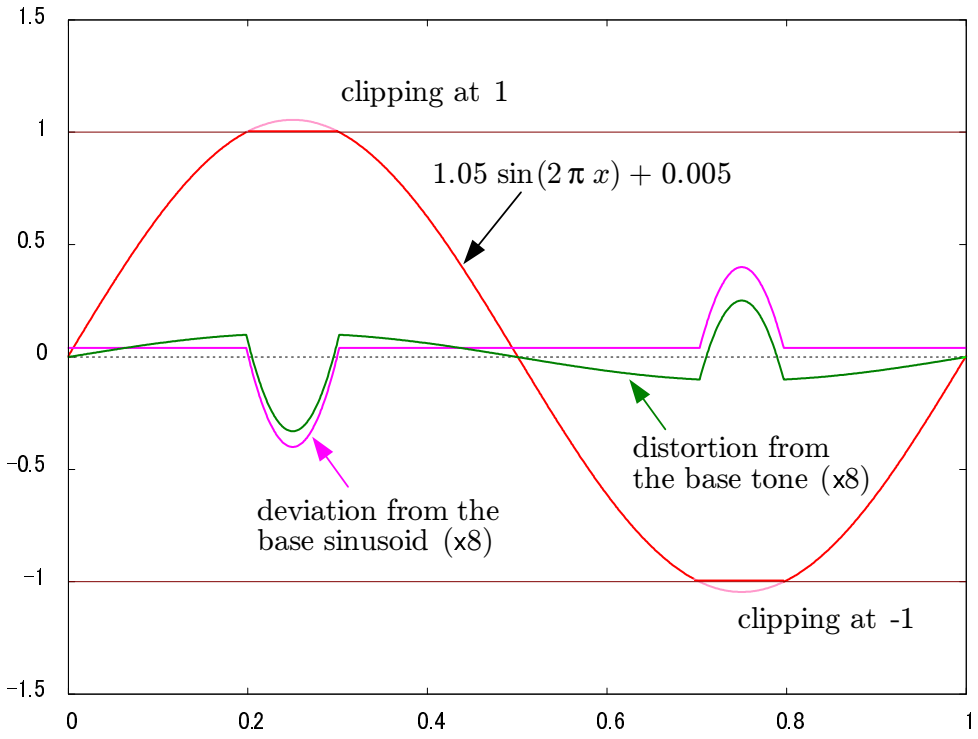

Figure 1.19: Power spectrum of data in Fig. 1.19

peak of the reconstructed signal. After noticed the distortion manner, we can analyze them with the assist of other measurements or simulations. In this particular case, the cause of this distortion is identified as the insufficient comparator speed. It would be hard to reach the solution only by the spectrum in Fig. 1.20.

1.4 Distortion posed by definite causes

The purpose of this section is to demonstrate the importance of resolution of the analysis for ADC characterization. Linearity measurement provides micro indexes that are deviation of individual codes, while distortion measurement provides only macro indexes that cannot be located to codes or sampled data. Concretely speaking, we examine the sensitivity difference by evaluating SINAD by five causes, *i.e.* quantization error, noise, DNL, INL and sparkles. These causes are *definite* since they are basically independent of the input.

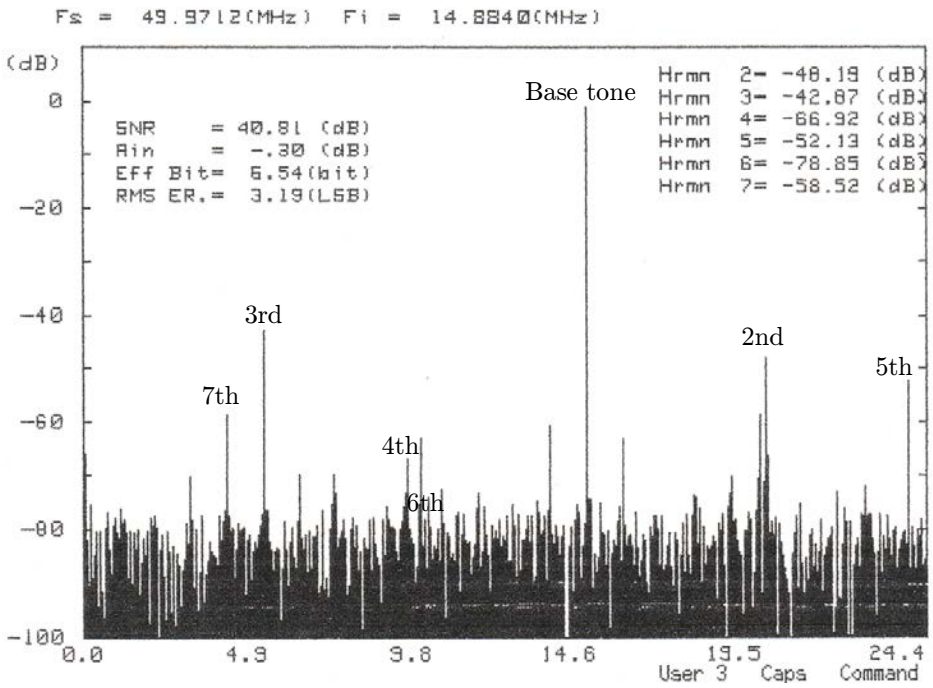

Figure 1.20: Power spectrum measurement of a 10-b ADC prototype: Fi= f_{in} = 14.8840 MHz, Fc= f_{clk} = 49.9712 MSps

Quantization error distributes almost uniformly in one LSB for various input. Then the power p_{qn} of the quantization error is expressed by the voltage average

$$\begin{aligned} p_{qn} &= \int_{-1/2}^{1/2} v^2 dv \\ &= \frac{1}{3}\left[x^3\right]_{-q/2}^{q/2} \\ &= \frac{1}{12} \end{aligned} \tag{1.9}$$

where the power is measured by square of LSB unit. This type of averaging is sometimes called “phase space average” from terminology in ergodic theory. On the other hand, if the input is the full scale sinusoid for *n*-bit ADC, the power of the signal p_{sig} is expressed by the time average

$$p_{sig} = \int_0^1 \left(2^{n-1}\sin(2\pi t)\right)^2 dt$$

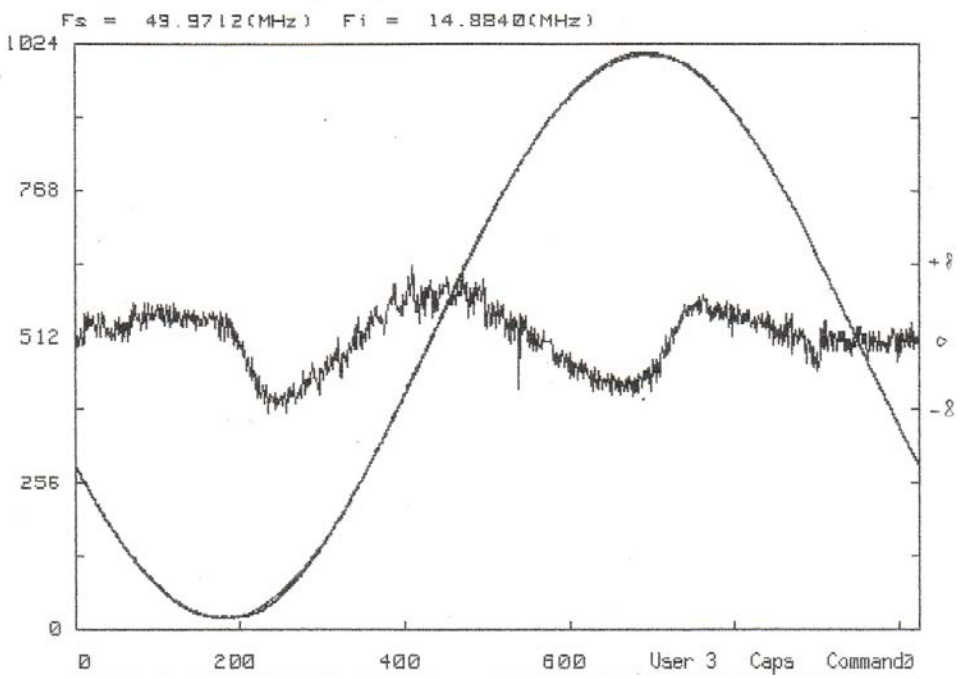

Figure 1.21: Reconstructed waveform and the difference wave of a 10-b ADC prototype: Fi= f_{in} = 14.8840 MHz, Fc= f_{clk} = 49.9712 MSps

$$= \quad 2^{2n-3} \tag{1.10}$$

so we get the well-known expression of SINAD of an ideal ADC.

$$\begin{aligned} \text{SINAD}_{ideal} &= 10\log\frac{p_{sig}}{p_{qn}} \\ &= 10\log\frac{2^{2n-3}}{1/12} \\ &= (20\log 2)\,n - 30\log(2) - 10\log\frac{1}{12} \\ &\approx 6.02n - 9.03 + 10.79 \\ &= 6.02n + 1.76 \end{aligned} \tag{1.11}$$

The idea of the Effective Number of Bits (ENOB) is to solve (1.11) for n. It is calculated by using measured SINAD as

$$\text{ENOB} = \frac{\text{SINAD} - 1.76}{6.02}\,. \tag{1.12}$$

In practice, SINAD is not measured in a full scale input in order to avoid clipping of the signal. As a result, measured SINAD does not reach the theoretical limit. The IEEE Std. 1241-2010 [5, 9.4] prescribes to compensate the amplitude. This stance is consistent with the assumption behind the equation (1.11). On the other hand, low order harmonics, such as second or third, often show the tendency to be smaller for smaller inputs. Amplitude compensation may provide unfairly good results using small input signals. Several books [1] [18] take a position not to compensate the amplitude in the definition.

The most misleading feature of the ENOB will exist in the fact that quantization noise and low order harmonics have different characters in the ADC performance. Those who do not know the term very well tend to assume, for example, a real 12-bit ADC of 10-bit ENOB has the equivalent performance of the ideal 10-bit ADC.

Next we evaluate INL and DNL separately. To consider a sine wave input, we assume the ADC input full range from -1 to 1. As a simple INL model, or as a working assumption, we take a second polynomial expression. Higher order warping is often observed in real ADCs but we will not lose the essence by this restriction.

$$\mathtt{dout}(v_{in}) = 2^{n-1}\, v_{in} + a\,(v_{in}^2 - 1) \tag{1.13}$$

where $\mathtt{dout}(v_{in})$ is the ADC output code from -2^{n-1} to 2^{n-1}. We treat it here as a real number.

$$\begin{aligned} \mathtt{dout}(-1) &= -2^{n-1} \\ \mathtt{dout}(0) &= -a \\ \mathtt{dout}(1) &= 2^{n-1} \end{aligned}$$

so a is the INL_{max} of the n-bit ADC. To be strict, INL is defined for dout, not for v_{in}. But if $|a| << 1$ that ordinary ADCs satisfy, we can reverse the equation (1.13) approximately

$$v_{in} \approx \mathtt{dout} + a\,(1 - \mathtt{dout}^2) \tag{1.14}$$

then we do not have to care this too much. For the sine wave $v_{in} = \sin(2\pi ft)$,

$$\begin{aligned} \mathtt{dout}(\sin(2\pi ft)) &= \sin(2\pi ft) + a\,(\sin(2\pi ft)^2 - 1) \\ &= \sin(2\pi ft) - a\,\cos(2\pi ft)^2 \\ &= \sin(2\pi ft) - \frac{a}{2}\,(1 + \cos(4\pi ft)) \end{aligned} \tag{1.15}$$

The second term of (1.15) is the distortion. The constant part $a/2$ is DC offset of the converted signal. This term is ignored following the definition of SINAD in Section 1.3. Then the power

p_{inl} is averaged over the period

$$\begin{aligned} p_{inl} &= \left(\frac{a}{2}\right)^2 \int_0^1 \cos(2\pi t)^2 dt \\ &= \frac{a^2}{8} \end{aligned} \tag{1.16}$$

The ignorance of the constant part $a/2$ will bring a problem as pointed out along with the SINAD definition in Section 1.3. DC offset for the full scale input is $a/2$ from (1.15), while it approaches to a for smaller input. Then the DC offset is modulated by varying amplitude of the signal. It will contribute to the distortion in the conversion process.

To evaluate the DNL effect, we use the space averaging (the code averaging in this case). Let the code transition level for code k as

$$\mathrm{C}_k = k + X_k \tag{1.17}$$

where X_k is a random variable with zero mean, and we use LSB unit in the expression. The X_k can include the time dependent value as well as a static deviation. The space averaging of the error power p_k is calculated

$$\begin{aligned} p_k &= \int_{-1/2+X_k}^{1/2+X_{k+1}} x^2 dx \\ &= \left[\frac{x^3}{3}\right]_{-1/2+X_k}^{1/2+X_{k+1}} \\ &= \frac{1}{3}\left(\left(\frac{1}{8} + \frac{3}{4}X_{n+1} + \frac{3}{2}X_{n+1}^2 + X_{n+1}^3\right) - \left(-\frac{1}{8} + \frac{3}{4}X_{n+1} - \frac{3}{2}X_{n+1}^2 + X_{n+1}^3\right)\right) \\ &= \frac{1}{12} + \frac{1}{4}(X_{n+1} - X_n) + \frac{1}{2}(X_{n+1}^2 + X_n^2) + \frac{1}{3}(X_{n+1}^3 - X_n^3) \end{aligned} \tag{1.18}$$

Total DNL power is also the average of p_k. In the average, the second and the last term of (1.18) are cancelled individually. So total power of the space average p_{space}

$$p_{space} = \mathbf{E}\left(\frac{\sum_k p_k}{n}\right) \tag{1.19}$$

$$\begin{aligned} &\approx \frac{1}{12} + \mathbf{E}\left(\frac{\sum_k X_k^2}{n}\right) \\ &= \frac{1}{12} + \frac{\sigma_{dnl}^2}{2} + \sigma_n^2 \end{aligned} \tag{1.20}$$

where n is the number of code levels, σ_{dnl} is the static DNL that is double of the variance of CT levels $\sigma_{dnl}^2 = 2\,\sigma_{CT}^2$, and σ_n is the input equivalent noise that is posed on X_k. Thus coefficients of σ_{CT} and σ_n become same. But the effects are different, because σ_{CT} is fixed in each chip while σ_n appears each conversion.

The number n is two to the power of the ADC bit-width. This is large to ignore a single or a few DNL changes in evaluation of p_{space}. For example, if one DNL deviates by 0.1 LSB more, it will be detected by linearity measurement but the p_{space} modification will hide in the measurement error by the division by n. This is the sensitivity difference between the micro index and the macro index.

A single sparkle is almost independent of the input. So the sparkle of size d_s LSB of rate r has the power $p_{sparkle}$

$$p_{sparkle} = d_s^2\, r\,. \tag{1.21}$$

In linearity measurement, the samples are usually very small compared to $1/r$, we expect no sparkle in the data. However if there is one sparkle, its power becomes much larger than the equation (1.21) to

$$p'_{sparkle} = \frac{d_s^2}{m} \tag{1.22}$$

where m is the number of the data used for distortion calculation. If the sparkle is small such as $d_s = 10$ LSB and the data is sufficiently large like $m = 16384$,

$$p'_{sparkle} = 10^2/16384 \approx 0.0061$$

which is negligible compared to the quantization noise $1/12 \approx 0.0833$. On the other hand, if $d_s = 30$, $p'_{sparkle} \approx 0.055$ becomes comparable to the quantization noise. According to the application, 10-bit sparkle does matter, while it cannot detect by distortion measurement. This is one of the reasons we should measure the sparkle rate independently.

The total distortion SINAD_{model} neglecting sparkle noise becomes

$$\begin{aligned}
\mathrm{SINAD}_{model} &= \log \frac{p_{sig}}{p_{space} + p_{inl}} \\
&= 10 \log \frac{2^{2n-3}}{\frac{1}{12} + \frac{\sigma_{dnl}^2}{2} + \sigma_n^2 + \frac{a^2}{8}} \\
&= (20 \log 2)\, n - 30 \log(2) - 10 \log\left(\frac{1}{12} + \frac{\sigma_{dnl}^2}{2} + \sigma_n^2 + \frac{a^2}{8}\right) \\
&\approx 6.02n - 9.03 - 10 \log\left(\frac{1}{12} + \frac{\sigma_{dnl}^2}{2} + \sigma_n^2 + \frac{a^2}{8}\right)
\end{aligned} \tag{1.23}$$

Table 1.2 shows some calculation results for 6-bit ADC. Since DNL is roughly three to five times of σ_{DNL}, σ_{DNL} may around 0.1 LSB. The bottom line of the table is a realistic case, which indicates 2 dB degradation from an ideal ADC.

Table 1.2: Some calculations of static distortion effects

σ_{DNL} (LSB)	σ_n (LSB)	INL (LSB)	SINAD (dB)
0.0	0.0	0.0	37.88
0.1	0.0	0.0	37.39
0.2	0.0	0.0	36.18
0.0	0.2	0.0	36.95
0.0	0.0	0.5	36.50
0.1	0.1	0.5	35.97

1.5 Organization of the thesis

In this introductory chapter, we have reviewed the analog-to-digital conversion process to its principle. We understand how it deviates at the arbitrary input signal is the key to characterize and to analyze the ADC. The conventional methods cover the range roughly described as the following schema.

ramp signal $\longrightarrow$ linearity error
sine signal $\longrightarrow$ distortion

The above two input signals have a common feature that the ideal output is easily determined from the output data. We have observed what the schema implied.

- Linearity measurement is sensitive but it eliminates the statistical nature around the code transition level (CT level).
- Power spectrum in frequency domain is much less sensitive since it does not provide the error at each conversion.
- Distortion by a sine signal will not cover possible distortions in real signals.

Purposes of this work are to overcome these constraints of conventional evaluation methods. The solutions provide more minute information of the ADC characters, which help the ADC performance improvements in turn. These problems require somewhat different theories. They are rather independent, but also they aim at the same direction, further resolution of analysis. They can be applied separately or in combination.

Chapter 2 deals with the enhancement of linearity measurement. We grasp it as a CT level estimation. One principal idea is to break a ramp output into a set of binary sequences. Two maximum likelihood equations (ML equations) for each CT level are derived without assuming a specific noise distribution. Then the logistic noise distribution is introduced to approximate the ML equations. Histogram method is verified from one ML equation. Another ML equation provides an explicit formula to evaluate the input equivalent noise. This is the information that the conventional linearity measurement has eliminated. Several examples demonstrate the usefulness of the input noise evaluation. We see the solutions given on the logistic noise approximation are valid in expectation mean for broader range of noise distributions including Gaussian noise. Variances of the estimation are explicitly provided for both Gaussian and logistic noise. As an application of the noise estimation, a latch circuit design is examined Effects of comparator hysteresis are also treated.

Chapter 3 deals with the general theory of waveform reconstruction for distortion analysis. I call the method "Euclidean reordering." In fact, the method itself is not new at all [17] [19] [20] [21] [22]. My contribution is consolidation of previous works into one theory. The difference wave accompanied with the reconstructed waveform plays an important role in the analysis. This is what provides the error at each conversion. Applying the method into flash ADCs, I have newly identified a distortion phenomenon that I named "dog-tooth." Mechanisms of dog-tooth are analyzed by using and extending the difference wave analysis. A low power bipolar latch circuit is introduced, which demonstrates the ability of the difference wave analysis.

In Chapter 4, we extend the difference wave analysis to general test signals other than sine waves. The least mean square estimation is provided for general signals. Composite signals characterize ADCs in practical signal conditions that monotone signals cannot. I propose two candidates for test signals: "two-tone signal" and "duplex sine signal." I show numerical examples of difference wave for both signals. These preliminary results exhibit the capability of the composite signals for the characterization of ADC.

Chapter 5 provides conclusions and the future work. I included my personal history of the research.

Appendix A introduces a series of flash ADCs. The measured data in the thesis are mainly taken from them. They are featured by strong error suppression capability and its low power, stemmed from their unique architecture.

Chapter 2

Linearity analysis as the code transition level estimation

Linearity is a sensitive measure for ADC characterization. The sensitivity stems from its resolution of each code transition (CT). In chapter 1, I noticed finer resolution will be obtained by analyzing the behavior around the CT level further. In this chapter, we grasp the linearity measurement as the estimation of CT levels. Maximum likelihood (ML) criteria and expectation value calculations provide quite minute results. I show various applications for measurement and design.

2.1 Derivation of the maximum likelihood equations

2.1.1 Decomposition of ADC outputs

First, I illustrate the problem that we deal here. The ramp out in Fig. 2.1 (upper) shows an example of the measured output sequence of an ADC. The test setup in Fig. 1.9 is used. The input is an ascending ramp signal generated by a waveform generator. The abscissa is the sample number of the output sequence. The ordinate is the output code. This ADC is 6-bit and its output code is straight binary from 0 to 63. The details of the design are provided in Appendix A. The input signal starts well-below of the ADC range bottom, so that earlier codes in Fig. 2.1 stick to zero. Along with the input signal rises, we observe ascending steps with vibrations at each code transition. Eventually the input goes over the ADC range top and the output codes will stick to 63, though Fig. 2.1 does not show the region. As easily seen, the output codes around each CT level are vibrating. This is more or less common nature in ADCs. The phenomena are primarily brought by noise, internal or external, of the ADC, because the input changes very slowly, the

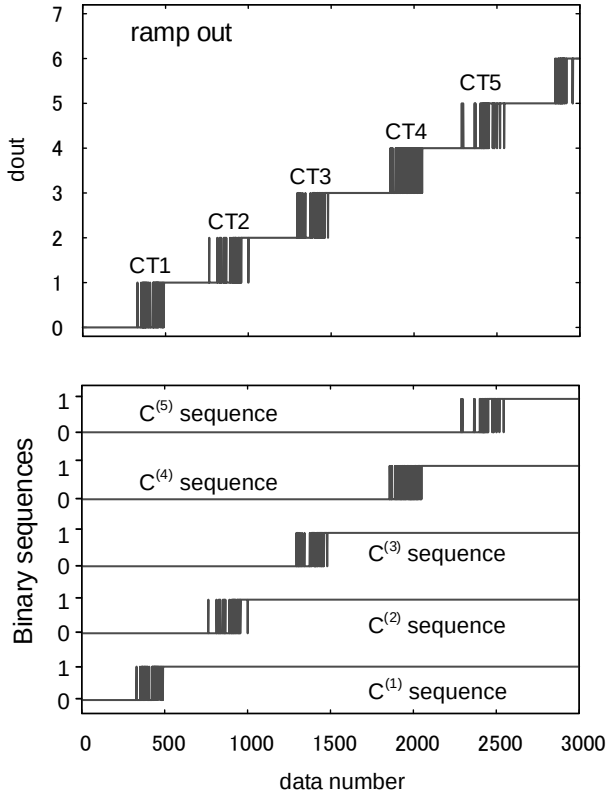

Figure 2.1: Measured output of an ADC of a ramp input (upper) and corresponding binary sequences (lower)

clock jitter does not affect the result. The problem is how we can define the true code transition behavior from the vibrating pattern.

As the input ramp signal rises, the probability of each output code occurrence also varies. Since our concern is not the code value itself but the code transition level, we had better deal each code transition separately. This idea is unique compared to previous works in literature [10] [23] that treat all CT levels as a whole. This decomposition is possible in reasonable conditions as follows.

Generally speaking, each code transition corresponds to one internal state change in the ADC. We can assume there is one and only one comparator corresponding to it, at least virtually. We denote this comparator as $C^{(n)}$, which is corresponding to the code transition from $n-1$ to n. In flash ADCs, the corresponding comparator is actually unique to each transition. In pipeline ADCs, physically the same comparator works several times as a corresponding comparator. That

is to say, by the concept of the corresponding comparator, we view various types of ADC as an equivalent flash ADC.

For normal ADCs, we can expect the following monotonicity condition:
monotonicity: ADC is *monotonic* when for all n,

$$\begin{aligned} C^{(n)} = 0 &\quad \Leftrightarrow \quad \texttt{dout} < n \\ C^{(n)} = 1 &\quad \Leftrightarrow \quad \texttt{dout} \geq n, \end{aligned}$$

where dout is the ADC output data and $\Leftrightarrow$ stands for "equivalent to." It is easy to see the above two conditions are actually contraposition.

The monotonicity in this sense is rather difficult to verify by measurements and it should be guaranteed by design. This way of monotonicity definition differs from IEEE Std. 1241-2010 [5], but conceptually the same idea with [13].

The term *monotonicity* is usually referred to the input-output characteristics [5]. However if noise exits, especially when the noise source is internal, it is difficult to define a monotonic relation between input and output, as the curve in Fig. 2.1 depicts. We avoided this difficulty by introducing internal states.

The monotonicity enables us to neglect other comparators to determine each internal comparator state from detectable output data. In effect by this condition, the ADC degenerates into a set of individual comparators. So the model we have to treat consists of only one comparator. Applying this concept into a ramp out, we have binary sequences corresponding to each CT level. Lower part of Fig. 2.1 depicts the concept. To grasp a picture of what we are doing further, let us focus on the corresponding comparator $C^{(2)}$ as an example. Under the monotonicity condition, if $\texttt{dout} < 2$ then $C^{(2)} = 0$ and if $\texttt{dout} \geq 2$ then $C^{(2)} = 1$. The $C^{(2)}$ sequence in Fig. 2.1 is obtained by this procedure.

2.1.2 Stochastic binary sequence driven by input noise

The next issue is to define the appropriate criterion to determine the CT level from the binary sequence. This work is carried out by parameter fitting of the probability curve. We assume the ADC input is an ascending ramp, so we expect the probability curve increases monotonically as shown in Fig. 2.2.

In order to get the probability transition curve, let us take a model in Fig. 2.3. The noise v_n is assumed to be additive. The origin is not necessarily internal, but we incorporate the external noises also into v_n. The input ramp signal v_{in} is sampled so as the input of k-th sample is $v_{in}[k]$. Restricting the interest range to an ascending slope, we can denote

$$v_{in}[k] = a\,(k - k_0) \;+\; v_{ref} \tag{2.1}$$

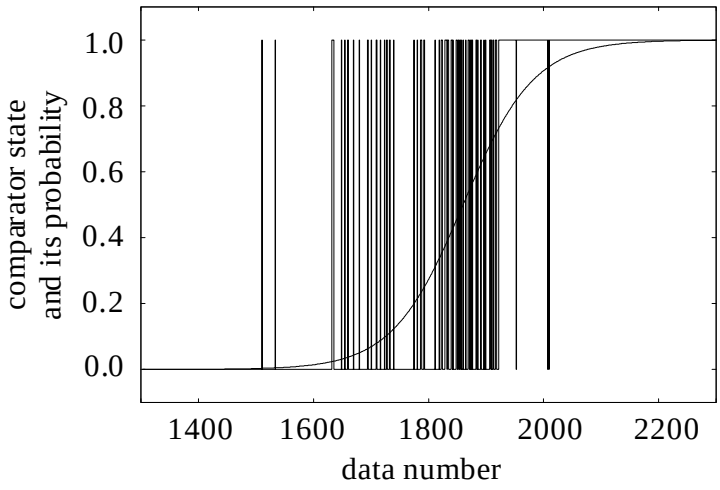

Figure 2.2: Comparator state sequence and an image of its probability curve

where the notation $[k]$ stands for k-th sample value of the continuous signal v_{in}. By this notation, we can use the same name for both discrete and continuous values.

The equation (2.1) implies the transition level of the input v_{in} that crosses the reference voltage v_{ref} is k_0. In the equation (2.1), the slope a is a controllable parameter in practical measurements and it can be treated as a known variable. There are reliable ways to estimated it separately. We do not care the absolute value v_{ref}, neither. Instead, the comparator offset we want to estimate is incorporated into k_0. Thus we rewrite the input signal equation (2.1) neglecting v_{ref}

$$v_{in}[k] = a\,(k - k_0) \tag{2.2}$$

and simplify our problem as k_0 estimation. Generally k_0 is a real variable, but we treat is as an integer assuming the slope a is small enough that an integer can approximate the real value. The comparator output $h[k]$ is a binary sequence that takes a value in $\{0, 1\}$. In textbooks on probability, the binary sequence is called as Bernoulli trials [24] [25].

Let denote the probability density function (PDF) of the noise of variance σ_n^2 as $p_n(v_n/\sigma_n)$. The function p_n itself is normalized and has no offset. So we assume followings:

$$p_n(v) \geq 0 \tag{2.3}$$

$$\int_{-\infty}^{\infty} p_n(v)\,dv = 1 \tag{2.4}$$

$$\int_{-\infty}^{\infty} v\,p_n(v)\,dv = 0 \tag{2.5}$$

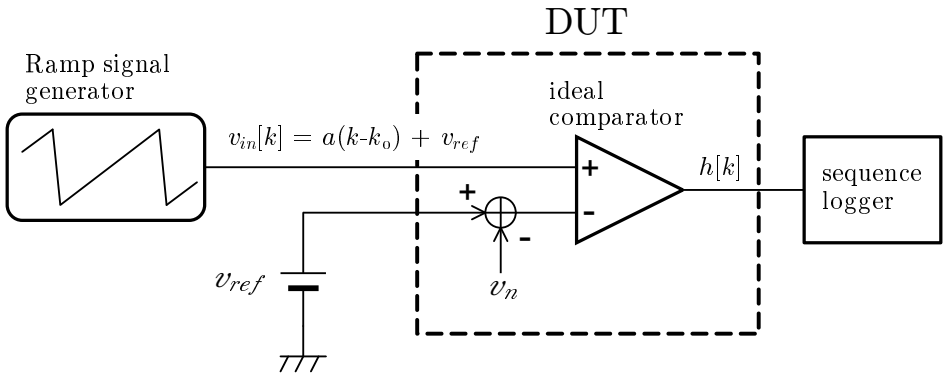

Figure 2.3: ADC noise measurement system on a degenerated comparator model

$$\int_{-\infty}^{\infty} v^2 \, p_n(v) \, dv \quad = \quad 1 \tag{2.6}$$

The $p_h = \text{Prob}[h = 1]$ denotes the probability of $h[k] = 1$, which is

$$p_h(v_{in}/\sigma_n) = \int_{-\infty}^{v_{in}/\sigma_n} p_n(x) dx \tag{2.7}$$

as understood from Fig. 2.4. The equation (2.7) is the cumulative distribution function (CDF) of the noise. In short, noise CDF is the comparator output PDF. Substituting (2.2) to (2.7) for the sampled value,

$$p_h[k] = p_h\left(\frac{a\,(k - k_0)}{\sigma_n}\right). \tag{2.8}$$

2.1.3 Maximization of the likelihood function

From discussions in Section 2.1.2, we have a binary series $\{\cdots, h[k-2], h[k-1], h[k], h[k+1] \cdots\}$ for the designated CT level, where $h[k] \in \{0, 1\}$ is a corresponding comparator value of the k-th data. The $p_h[k]$ stands for the probability $h[k] = 1$ and in reverse the probability $h[k] = 0$ is $1 - p_h[k]$. We assume each occurrence $h[k]$ is independent. Then the possibility that we get a particular sequence forms the following *likelihood function*

$$L = \prod_k p_h[k]^{h[k]}(1 - p_h[k])^{1-h[k]}. \tag{2.9}$$

We omit the range of k unless the explicit expression is necessary.

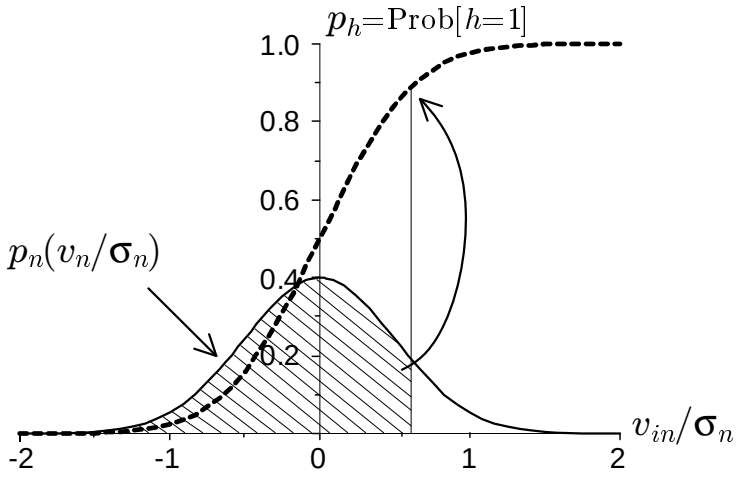

Figure 2.4: Illustrative relation between a noise distribution and a comparator output probability

The likelihood function is proportional to the probability the particular sequence happens, but it is not probability since it is not normalized, i.e. the integral of whole range may not be one. We adopt the criterion that the best estimate of k_0 should maximize the probability of the occurrence, and so the likelihood function.

Taking logarithm of both sides of (2.9), we get the log-likelihood function

$$\log L = \sum_k \left(h[k] \log p_h[k] + (1 - h[k]) \log(1 - p_h[k]) \right) . \tag{2.10}$$

Substituting (2.8) to $p_h[k]$ and differentiating (2.10) by k_0,

$$\begin{aligned} \frac{\partial \log L}{\partial k_0} &= \sum_k \left(\frac{a\, h[k]\, p_h'[k]}{p_h[k]} - \frac{a\,(1 - h[k])\, p_h'[k]}{1 - p_h[k]} \right) \\ &= a \sum_k \frac{p_h'[k]}{(1 - p_h[k]) p_h[k]} \left(h[k] - p_h[k] \right) \end{aligned} \tag{2.11}$$

where

$$p_h'[k] = \left. \frac{dp_h(v)}{dv} \right|_{v = a\,(k - k_0)} \tag{2.12}$$

and similarly differentiating (2.10) by $1/\sigma_n$,

$$\frac{\partial \log L}{\partial\, 1/\sigma_n} = \sum_k \frac{p_h'[k]}{(1 - p_h[k])\, p_h[k]} (k - k_0)(h[k] - p_h[k]) . \tag{2.13}$$

The formulas look simpler by introducing the weight function $W(v)$

$$W(v) = \frac{p_h'(v)}{(1 - p_h(v))\, p_h(v)} \tag{2.14}$$

and weighting coefficients

$$\begin{aligned} w[k] &= W\left(a\,(k - k_0)\right) && (2.15)\\ &= \frac{p_h'[k]}{(1 - p_h[k])\, p_h[k]}\,. && (2.16) \end{aligned}$$

Then at the maximum of the likelihood function L, by requiring (2.11) and (2.13) to be zero, we get the following simultaneous ML equations.

$$\sum_k w[k]\,(p_h[k] - h[k]) = 0 \tag{2.17}$$

$$\sum_k w[k]\,(k - k_0)\,(p_h[k] - h[k]) = 0\,. \tag{2.18}$$

In equations, $h[k]$ is a binary sequence extracted from measured data and k_0 is a variable to be determined.

The noise PDF defines $p_h[k]$ and also $w[k]$, which can be so non-linear that equations are very difficult to solve. Numerical algorithms such as Newton-Raphson seem only practical method as in [10] [23] in other literature. However, we have something to investigate the ML equations before going to a numerical resort.

2.2 The maximum likelihood equations of some specific noise distributions

We have not assumed any specific noise PDFs so far. In this section, we look into some special cases of practical importance.

2.2.1 Gaussian distribution

When the noise is Gaussian, the specific form of $p_h(v)$ in (2.7) is $p_{hg}(v)$ below:

$$\begin{aligned} p_{hg}(v/\sigma_n) &= \frac{1}{\sqrt{2\pi}\sigma_n}\int_{-\infty}^{v} \exp\left(-\frac{x^2}{2\sigma_n^2}\right) dx \\ &= \frac{1 + \mathrm{erf}\left(\dfrac{v}{\sqrt{2}\,\sigma_n}\right)}{2} \end{aligned} \tag{2.19}$$

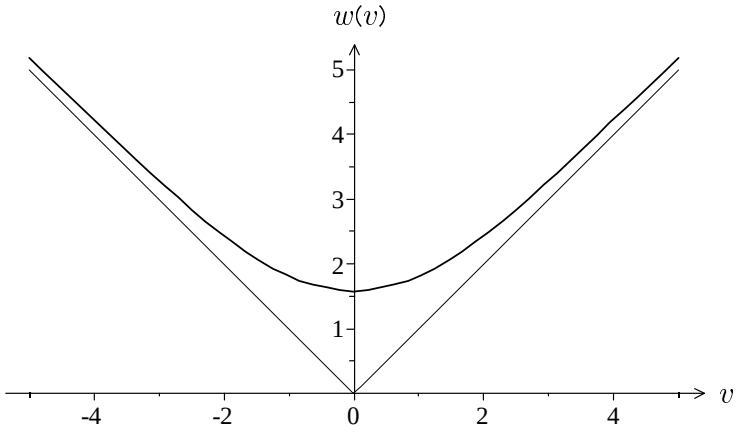

Figure 2.5: Weight function for Gaussian distribution and its asymptotic lines

where erf is the error function [26]. Substituting (2.19) to (2.14), we get the weight function for Gaussian noise

$$W_g(v) = \frac{2\sqrt{2}\exp\left(-\frac{v^2}{2\sigma_n^2}\right)}{\sqrt{\pi}\,\sigma_n\left(1-\mathrm{erf}\left(\frac{v}{\sqrt{2}\,\sigma_n}\right)^2\right)} \tag{2.20}$$

Fig. 2.5 shows the weight function when $\sigma_n = 1$. The curve is concave and symmetrical with respect to $v = 0$. The value at $v = 0$ is

$$W_g(0) = \frac{2\sqrt{2}}{\sqrt{\pi}\,\sigma_n} \approx \frac{1.595769}{\sigma_n} \tag{2.21}$$

The term $p_h[k] - h[k]$ in equations (2.17) (2.18) takes the value around 0.5 with alternative polarities near the code transition. In this region, the weight function for Gaussian noise is roughly constant and takes the minimal value. At the regions apart from the code transition, most of $(p_h[k] - h[k])$'s are close to zero, but if not, the value is close to +1 or -1, and has a large weight. Such the occurrences have a greater affect on the solutions of (2.17) and (2.18) than those around the code transition.

The asymptotic expansion of $W_g(v)$ is

$$\sigma_n\, W_g(v) = \frac{|v|}{\sigma_n} + \frac{\sigma_n}{|v|} - 2\left(\frac{\sigma_n}{|v|}\right)^3 + O\left(\left(\frac{\sigma_n}{|v|}\right)^5\right) \tag{2.22}$$

so $W_g(v)$ approximately proportional to $|v|$ for large $|v|$. Fig. 2.5 also shows the asymptotic lines.

Weight function introduced in [27] is similar to these asymptotic lines, while merely the origin is shifted. The weight function is the basis of the *tally and weight* method discussed in [10]. From ML point of view, the method is optimal only upon an implausible noise distribution far from Gaussian.

2.2.2 Logistic distribution

In the previous subsection, we assumed the Gaussian CDF and then derived the weighting function (2.20). Next here, we assume a constant weighting function first.

$$W(v) \equiv w_0 \,. \tag{2.23}$$

This is the assumption to weigh every output code equally without considering its point of occurrence. Or we can say equivalently this assumption is to ignore the order of occurrence of the data.

To get the explicit formula of the noise PDF, substitute the assumption (2.23) to (2.14), obtaining an ordinary differential equation (ODE)

$$\frac{d\, p_h(v)}{d\, v} = (1 - p_h(v))\, p_h(v)\, w_0 \,. \tag{2.24}$$

This ODE is known as a *logistic equation* [26]. When the noise offset is zero, or equivalently the condition $p_h(0) = 0.5$, the solution of (2.24) gives a *logistic function*

$$p_h(v) = \frac{\exp(w_0\, v)}{1 + \exp(w_0\, v)} \tag{2.25}$$

The σ_n^2 calculated from (2.25) is

$$\sigma_n^2 = \int_{-\infty}^{\infty} v^2\, \frac{d}{dv}\left(\frac{\exp(w_0\, v)}{1 + \exp(w_0\, v)}\right) dv = \frac{\pi^2}{3\, w_0^2} \tag{2.26}$$

so we get the logistic probability curve of the standard deviation σ_n

$$p_{hl}(v/\sigma_n) = \frac{\exp\left(\dfrac{\pi}{\sqrt{3}}\, \dfrac{v}{\sigma_n}\right)}{1 + \exp\left(\dfrac{\pi}{\sqrt{3}}\, \dfrac{v}{\sigma_n}\right)} \,. \tag{2.27}$$

We call the noise with logistic distribution as *logistic noise*.

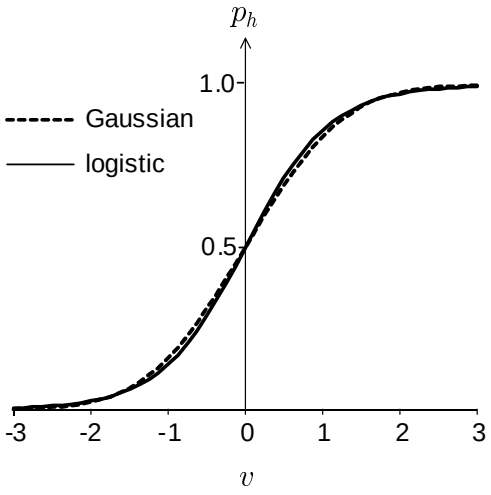

Figure 2.6: Probability curve comparison between a Gaussian noise and a logistic noise. The σ_n is normalized to 1.

The logistic probability curve (2.27) is close to the Gaussian probability curve (2.19), but it is slightly steeper around $v = 0$ and has longer tails where $|v|$ is large. Fig. 2.6 shows both curves of $\sigma_n = 1$.

The ML equations (2.17) (2.18) for logistic noise have the following simple forms.

$$\sum_k (p_h[k] - h[k]) = 0 \tag{2.28}$$

$$\sum_k (k - k_0)\,(p_h[k] - h[k]) = 0\,. \tag{2.29}$$

There are subtle situations in theory and practice. As shown in Fig. 2.5, the occurrence of rare case in Gaussian noise should have more weight than fluent cases. Equations (2.28) and (2.29) are obtained by ignoring the order of the occurrence. On the other hand, the precise noise distribution should not be Gaussian nor logistic. In addition, the rare case affects less on the CT level estimation. Hence it will be acceptable we take (2.28) and (2.29) as approximations of the ML equations (2.17) and (2.18) for practical noise distributions. Later in section 2.5, we will investigate legitimacy of the approximation.

One more comment. If we rewrite formally the equation (2.28) to $\sum_k p[k] = \sum_k h[k]$, we can say both summations of theoretical probabilities and of measured data meet each other. Similarly we can interpret the equation (2.29) as the first order moment equation. The forms (2.28)(2.29) are preferable not to be rewritten because they converge when $k \to \infty$.

2.3 Solutions of the maximum likelihood equations

Even after eliminating the weight function, the approximated ML equations (2.28) (2.29) are still non-linear both for the CT level k_0 and the noise parameter σ_n. But if we select the range of k wide enough around k_0, approximate solutions become quite simple.

2.3.1 Optimal code transition level estimation

If the measurement range is wide enough, early $h[k]$'s are 0 and late $h[k]$'s are 1. So if k_r is large enough and $k_a < k_0 - k_r < k_0 + k_r < k_b$,

$$\sum_{k=k_a}^{k_0-k_r-1} p_k \approx 0 \tag{2.30}$$

$$\sum_{k=k_0+k_r+1}^{k_b} p_k \approx k_b - (k_0 + k_r) \tag{2.31}$$

Since logistic distribution is symmetric

$$p_{hl}(-v) = 1 - p_{hl}(v) \tag{2.32}$$

then

$$\begin{aligned} \sum_{k=k_0-k_r}^{k_0+k_r} p_{hl}[k] &\approx \int_{k_0-k_r}^{k_0+k_r} p_{hl}\,(a(k-k_0))\,dk \\ &= \int_{-k_r}^{k_r} p_{hl}\,(a\,k)\,dk \\ &= \int_{0}^{k_r} (p_{hl}\,(-a\,k) + p_{hl}\,(a\,k))\,dk \\ &= k_r + 1\,. \end{aligned} \tag{2.33}$$

From (2.30) (2.31) (2.33)

$$\begin{aligned} \sum_{k=k_a}^{k_b} p_{hl}[k] &= \left(\sum_{k=k_a}^{k_0-k_r-1} + \sum_{k=k_0-k_r}^{k_0+k_r} + \sum_{k=k_0+k_r+1}^{k_b} \right) p_{hl}[k] \\ &\approx k_r + 1 + k_b - (k_0 + k_r) \\ &= k_b - k_0 + 1\,. \end{aligned} \tag{2.34}$$

Substituting (2.34) into the ML equation (2.28), we get the approximate version of the ML equation

$$k_b - k_0 + 1 - \sum_{k=k_a}^{k_b} h[k] = 0 \tag{2.35}$$

or

$$\begin{aligned} & k_b - k_0 + 1 - \sum_{k=k_a}^{k_b} h[k] \\ = \; & k_b - k_0 + 1 - \left(k_b - k_a + 1 - \sum_{k=k_a}^{k_b} (1 - h[k]) \right) \\ = \; & \sum_{k=k_a}^{k_b} (1 - h[k]) + k_a - k_0 = 0 \end{aligned} \tag{2.36}$$

thus

$$k_0 = k_a + \sum_{k=k_a}^{k_b} (1 - h[k]) \,. \tag{2.37}$$

Since $h[k]$ is measured data and k_a is known, the Eq. (2.37) determines the transition level k_0 for the logistic noise. The sum term of Eq. (2.37) is identical to the count of 0's in the binary sequence. The very important feature of (2.36) is that it does not depend on the parameter σ_n nor a.

2.3.2 Optimal linearity measurement

Here we consider the linearity measurement constructed from (2.36). For the code transition from $n-1$ to n, let's denote the expression k_0 of (2.37) as $k_0^{(n)}$ and $h[k]$ as $h[k]^{(n)}$. Then

$$k_0^{(n)} = k_b - \sum_{k=k_a}^{k_b} h[k]^{(n)} \tag{2.38}$$

and if k_b is large enough also for $k_0^{(n+1)}$

$$k_0^{(n+1)} = k_b - \sum_{k=k_a}^{k_b} h[k]^{(n+1)} \,. \tag{2.39}$$

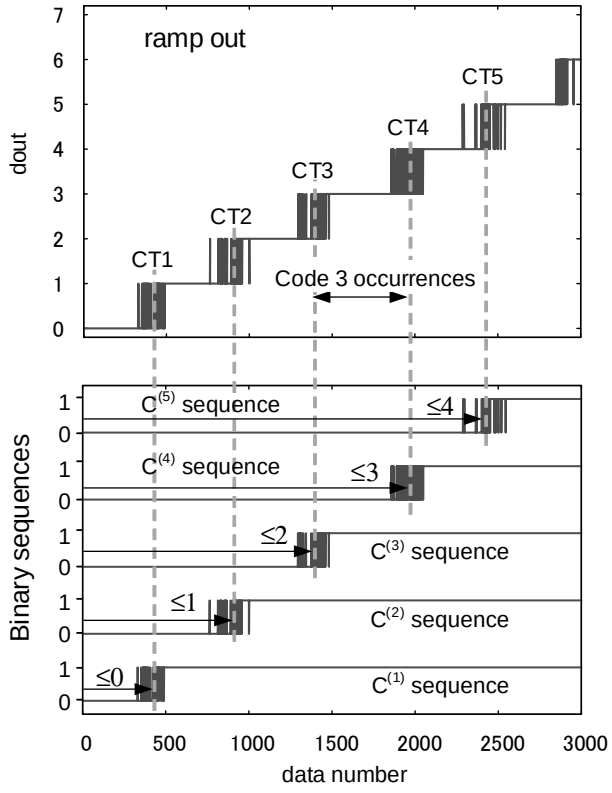

Figure 2.7: Interpretation of CT level estimations, Eq. (2.37): The best CT level estimation for a binary sequence is 0's count, which is the sample number under the CT level. So the difference of adjacent CT levels is the code count, or better known as "histogram" or "code density."

Subtracting (2.38) from (2.39), we get

$$k_0^{(n+1)} - k_0^{(n)} = \sum_{k=k_a}^{k_b} \left(h[k]^{(n+1)} - h[k]^{(n)} \right) . \tag{2.40}$$

Noting that $h[k]^{(n+1)} - h[k]^{(n)}$ is the characteristic function that the k-th output is n, the right side of (2.40) is equal to the code bin count of n. This is the criteria the histogram (or code density) method adopts. Fig. 2.7 depicts the situation.

2.3.3 Optimal input noise variance estimation

Another ML equation (2.29) provides the noise variance σ_n. To see this, calculate the left side of (2.29)

$$\begin{aligned}&\left(\sum_{k<k_0}+\sum_{k>k_0}\right)(k_0-k)\,(p_h[k]-h[k])\\&=\sum_{k<k_0}(k_0-k)\,p_h[k]-\sum_{k<k_0}(k_0-k)\,h[k]\\&+\sum_{k>k_0}(k-k_0)\,(1-p_h[k])-\sum_{k>k_0}(k-k_0)\,(1-h[k])\end{aligned}$$

and from the symmetry condition (2.32)

$$\sum_{k<k_0}(k_0-k)\,p_h[k]=\sum_{k>k_0}(k-k_0)\,(1-p_h[k]) \tag{2.41}$$

then (2.29) turns to

$$2\sum_{k<k_0}(k_0-k)\,p_h[k]=\sum_{k<k_0}(k_0-k)\,h[k]+\sum_{k>k_0}(k-k_0)\,(1-h[k])\,. \tag{2.42}$$

Using (2.27) and (2.8)

$$\begin{aligned}\sum_{k<k_0}(k_0-k)\,p_h[k]&\approx\int_{-\infty}^{0}(-k)p_{hl}\,(ak)\,dk\\&=\frac{\sigma_n^2}{4a^2}\,.\end{aligned} \tag{2.43}$$

Using (2.42) and (2.43) we have the explicit expression of the variance

$$\sigma_n^2=2a^2\left(\sum_{k<k_0}(k_0-k)\,h[k]+\sum_{k>k_0}(k-k_0)\,(1-h[k])\right) \tag{2.44}$$

where a is a known variable that is determined from the inclination either of the terminal-based straight line or of the best-fit straight line [5]. So we can easily calculate (2.44) from measured data.

The first term of the right side is the moment of the occurrences of $h[k]=1$ before the code transition, and the second term is the moment of the occurrences of $h[k]=0$ after the

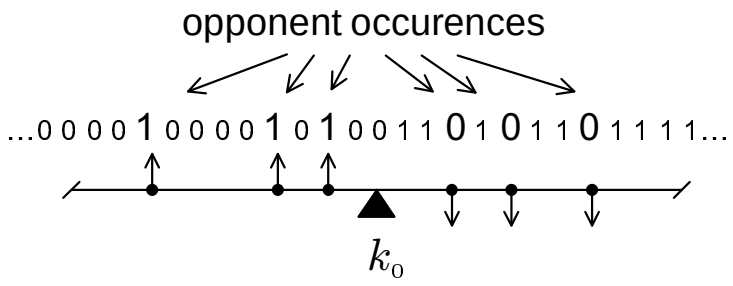

Figure 2.8: Interpretation of σ_n equation: and the first moment of the opponent occurrences around the CT level

code transition. The sum is conceptually the same with the physical moment of the opponent occurrences of the binary sequence given from the measured data. Fig. 2.8 illustrates the physical image of these terms. Thus the formula (2.44) claims that the normalized moment using the mean code bin width a, by the factor $2a^2$, provides the input noise variance σ_n^2. We summarize the equations in Table 2.1.

The σ_n calculated by (2.44) is useful for several ways.

- ADC's static performance is not solely decided by DNL/INL, but also affected by the intrinsic noise. I daringly say the noise variance is the third index of linearity having matching importance with DNL/INL.
- It also includes the information of the measurement accuracy.
- Implementation of the equation (2.44) is straightforward that is easy to use in simulation.
- It will be appropriate to use in calibration, too.

I will show a Perl script of σ_n calculation later in Table 2.3.

Table 2.1: The equations for the CT level estimation and the input noise estimation

CT level estimation	$\hat{k}_0 = \sum_k (1 - h[k])$
Input noise variance	$\hat{\sigma}_n^2 = 2a^2 \left(\sum_{k<k_0} (k_0 - k)h[k] + \sum_{k>k_0} (k - k_0)(1 - h[k]) \right)$

2.4 Ordinary and out of ordinary examples of linearity measurement

The formula (2.44) provides σ_n at each transition. This feature enables us to plot the linearity with σ_n's. Fig. 2.9 illustrates an example of measured linearity of a 6-bit ADC, the same data already used in Fig. 2.1. Small warping of INL is observable in this particular ADC, while the standard deviations look fine.

Fig. 2.10 and Fig. 2.11 are intended to demonstrate the inherent drawback of the code density method. It shows an artificial example that swapped the output code 5 and 6 of the ramp measured in Fig. 2.1. As Max [27] has pointed out, both DNL and INL are the same with Fig. 2.9 because the code bin counts do not change. However, the big difference appears in the calculated standard deviation. When the swapped codes are more apart, the deviation becomes larger. Thus the deviation measure by (2.44) redeems the drawback of the code density method.

Fig. 2.12 shows a block diagram of an ADC that can deceive the code density method by a different manner. This ADC has an ordinal 3-bit core and one bit uniform pseudo random binary sequence (PRBS) generator. Since the code density method does not distinguish a ramp signal from a uniformly distributed random signal, this one-bit extension exhibits very good linearity in the code density method.

Applying the same idea to the original 6-bit data in Fig. 2.1, we get Fig. 2.13. The figure shows a quasi 7-bit ramp out numerically appended a pseudo-random bit. Except noisier, the curve looks similar to the 6-bit one rather than a 7-bit ramp. Fig. 2.14 shows the linearity. Though the INL is doubled by halved LSB, the DNL is very small. The code density method does not detect the trick. We see the standard deviation curve reveals the difference.

The above situation is not solely imaginary. The similar effect may really happen if the ADC noise is very large. DNL errors could be averaged by the noise so that the good linearity might

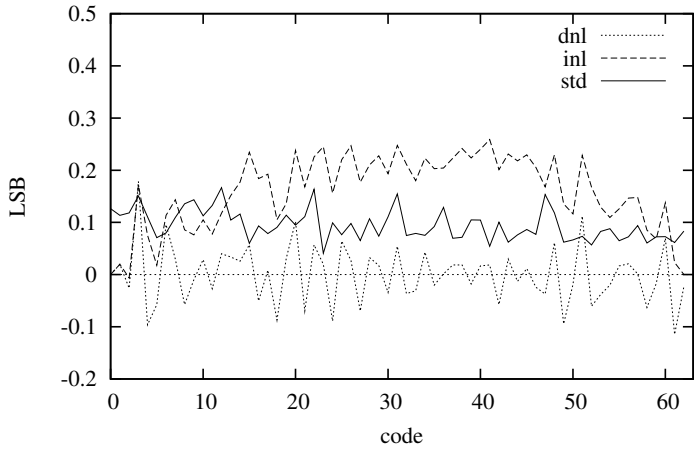

Figure 2.9: An example of linearity plot with standard deviations

be observed by the code density method. In such a case, the standard deviations given by (2.44) become large all in all.

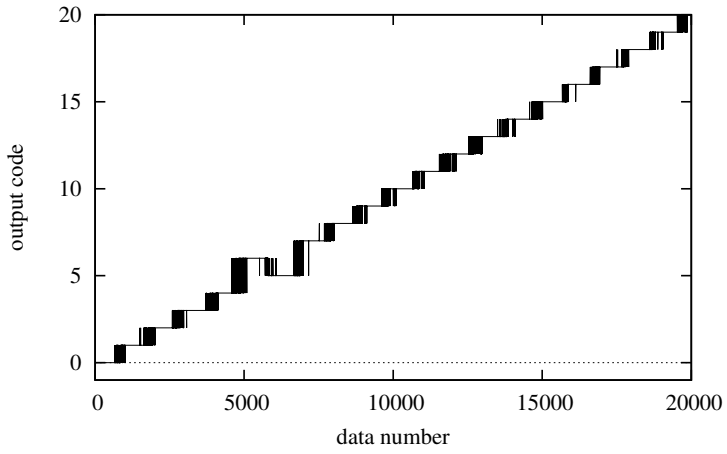

Figure 2.10: Ramp output made by swapping code 5 and 6 from measured data

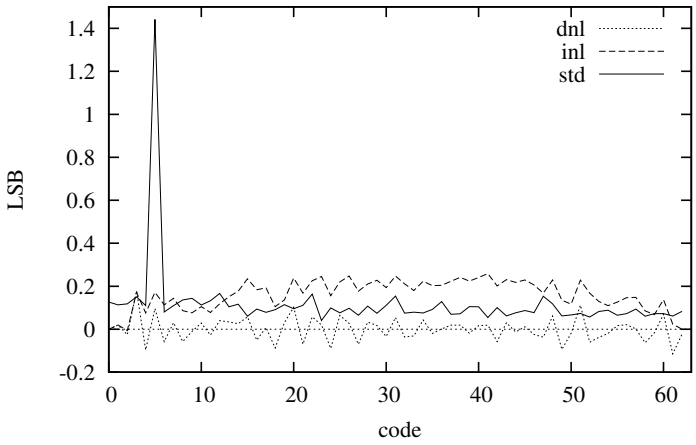

Figure 2.11: Linearity plot of the swapped data

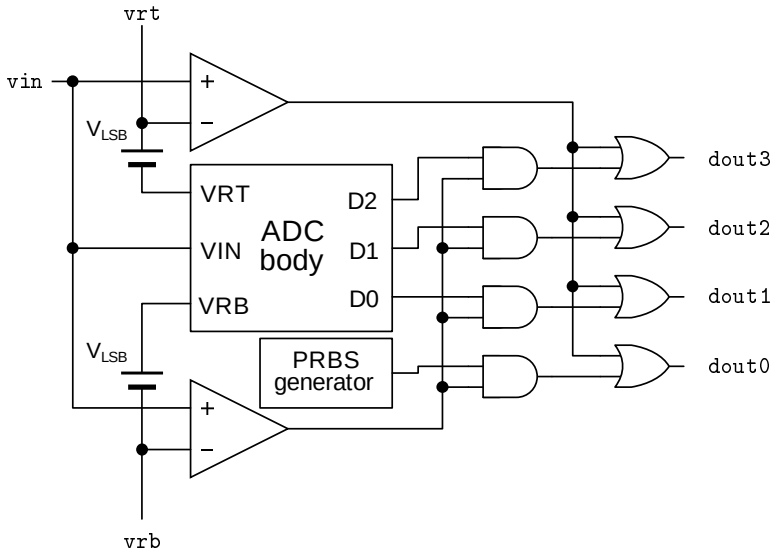

Figure 2.12: One-bit digitally extended ADC

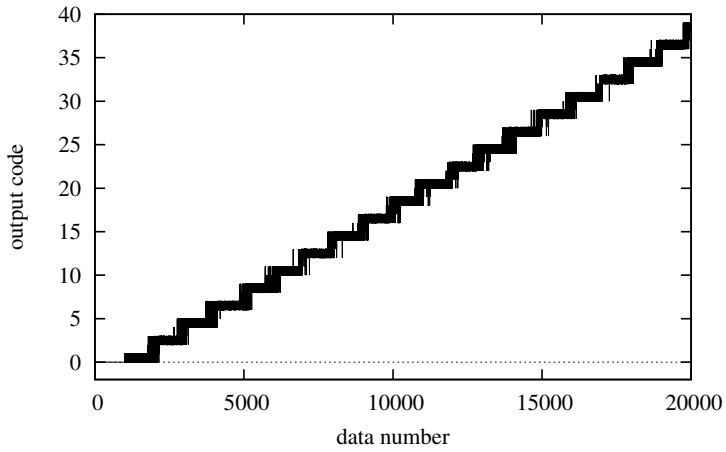

Figure 2.13: Ramp output of quasi-seven-bit ADC

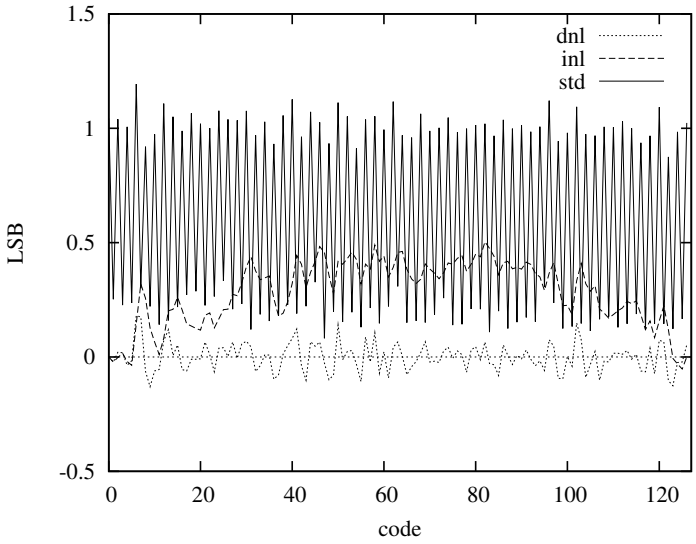

Figure 2.14: Linearity plot of quasi-seven-bit ADC with standard deviations

2.5 Expectation of estimations

The equations of the CT level estimation (2.36) and noise variance estimation (2.44) are derived for logistic noise. This section shows these equations are unbiased estimations for wider range of noise distributions, including Gaussian noise.

Before going forward to main discussion, I explain the subtle situation by a simple example. If the measured data of a single value is a set $\{d_0, d_1, d_2\}$. To estimate the true value d_t, we usually calculate the average a_1

$$a_1 = \frac{d_0 + d_1 + d_2}{3} . \tag{2.45}$$

If the measurement is unbiased, the expectation of each data is

$$\mathbf{E}[d_0] = \mathbf{E}[d_1] = \mathbf{E}[d_2] = d_t \tag{2.46}$$

so

$$\mathbf{E}[a_1] = \frac{\mathbf{E}[d_0] + \mathbf{E}[d_1] + \mathbf{E}[d_2]}{3} = \frac{3d_t}{3} = d_t . \tag{2.47}$$

This result means the equation (2.45) is an unbiased estimation of d_t. The point is there are many other equations that have the same property. For example,

$$a_2 \quad = \quad 0.25d_0 + 0.25d_1 + 0.5d_2$$

$$a_3 \quad = \quad 0.1d_0 + 0.2d_1 + 0.7d_2 \,.$$

Generally, as far as the coefficients satisfy

$$w_0 + w_1 + w_2 = 1 \tag{2.48}$$

The expectation of $a = w_0 d_0 + w_1 d_1 + w_2 d_2$ is unbiased.

$$\mathbf{E}\,[a] = (w_0 + w_1 + w_2)\, d_t = d_t \,. \tag{2.49}$$

This example indicates "unbiased" is a quite loose requirement for the estimation.

Now, we go back to estimations in Table 2.1. They are

$$\hat{k}_0 \quad = \quad \sum_k (1 - h[k]) \tag{2.50}$$

$$\hat{\sigma}_n^2 \quad = \quad 2a^2 \left(\sum_{k<k_0} (k_0 - k) h[k] + \sum_{k>k_0} (k - k_0)(1 - h[k]) \right) \tag{2.51}$$

where $\hat{}$ indicates the estimation of the value. The former equation (2.50) provides the CT level estimation and the latter equation (2.51) provides noise variance. I repeat that they are optimal in ML sense only for logistic noise, while they are valid for much wider noise distribution in expectation sense.

Note since $h[k]$ is binary, the expectation $h[k] = 1$ is its probability.

$$\mathbf{E}\,[h[k]] = p_h[k] \tag{2.52}$$

Here we assume the noise distribution is symmetric

$$p_h[k_0 - k] \equiv 1 - p_h[k_0 + k] \tag{2.53}$$

for all k. This is a discrete version of (2.32). Then the expectation of the right hand side of (2.50) becomes

$$\begin{aligned} \mathbf{E}\left[\sum_k (1 - h[k]) \right] \quad &= \quad \sum_k (1 - p_h[k]) \\ &= \quad \left(\sum_{k<k_0} (1 - p_h[k]) + \sum_{k<k_0} p_h[k] \right) + \left(- \sum_{k<k_0} p_h[k] + \sum_{k>k_0} (1 - p_h[k]) \right) \qquad (2.54) \\ &= \quad \sum_{k<k_0} 1 \qquad (2.55) \end{aligned}$$

The lower limit of the sum in the last formula is the first data number, so the sum provides the data count before k_0, which is the estimation $\hat{k_0}$. In short, we derived the equation (2.50) from the symmetry condition (2.53). Note that not only logistic noise but also Gaussian noise satisfies the symmetry condition.

Next we go to the noise variance estimation (2.51). Using (2.2) and (2.7),

$$\begin{aligned} & \mathbf{E}\left[\sum_{k<k_0}(k_0-k)h[k]+\sum_{k-k_0}(k-k_0)(1-h[k])\right] \\ = & \frac{1}{a}\left(\sum_{k<k_0}\left(-v_{in}[k]\int_{-\infty}^{v_{in}[k]}p_n(x/\sigma_n)dx\right)+\sum_{k_0<k}\left(v_{in}[k]\int_{v_{in}[k]}^{\infty}p_n(x/\sigma_n)dx\right)\right) \\ \approx & \frac{1}{a^2}\left(\int_{-\infty}^{0}(-v)\cdot\left(\int_{-\infty}^{v}p_n(x/\sigma_n)dx\right)dv+\int_{0}^{\infty}v\cdot\left(\int_{v}^{\infty}p_n(x/\sigma_n)dx\right)dv\right) \end{aligned} \tag{2.56}$$

Integration by parts of the first term of the last equation provides

$$\begin{aligned} & \int_{-\infty}^{0}(-v)\left(\int_{-\infty}^{v}p_n(x/\sigma_n)dx\right)dv \\ = & \left[-\frac{v^2}{2}\int_{-\infty}^{v}p_n(x/\sigma_n)dx\right]_{-\infty}^{0}+\frac{1}{2}\int_{-\infty}^{0}v^2p_n(v/\sigma_n)dv \end{aligned} \tag{2.57}$$

where the first term converses to 0 by a CDF property. With the similar calculation of the second term of (2.56), we have

$$\mathbf{E}\left[\sum_{k<k_0}(k_0-k)h[k]+\sum_{k-k_0}(k-k_0)(1-h[k])\right]=\frac{1}{2a^2}\int_{-\infty}^{\infty}v^2p_n(v/\sigma_n)dv \tag{2.58}$$

The integral provides σ_n from (2.6). Thus we have calculated the expectation of noise variance (2.51).

2.6 Variance of the estimation

The ML estimation given by (2.50) is also subject to statistical fluctuations. The estimation error is evaluated as the variance of the estimation. Practical importance of ML estimation is that the variance of the estimation tends to be the minimum asymptotically under *regular* conditions. This property is called *efficiency* in statistics [25]. The efficiency is much more restricting condition than unbiasedness. In stead of dealing with this subject straightly, we focus on the variance of estimations (2.50) and (2.51).

The variance of (2.50) is

$$\begin{aligned}\mathbf{Var}(\hat{k}_0) &= \mathbf{E}\left[\hat{k}_0^2 - \mathbf{E}\left[\hat{k}_0\right]^2\right] \\ &= \mathbf{E}\left[\left(\sum_k (1-h[k])\right)^2 - \left(\sum_k (1-p_h[k])\right)^2\right] \\ &= \mathbf{E}\left[\left(\sum_k h[k]\right)^2\right] - \left(\sum_k p_h[k]\right)^2 \end{aligned} \tag{2.59}$$

The first term becomes

$$\mathbf{E}\left[\left(\sum_k h[k]\right)^2\right] = \mathbf{E}\left[\sum_k h[k]^2\right] + \mathbf{E}\left[\sum\sum\nolimits_{k\neq l} h[k]h[l]\right] \tag{2.60}$$

Using the following properties of binary sequence

$$h[k]^2 = h[k] \tag{2.61}$$

$$(h[k]h[l]=1) \Leftrightarrow (h[k]=1)\&(h[l]=1) \tag{2.62}$$

the equation (2.60) becomes

$$\begin{aligned}\mathbf{E}\left[\left(\sum_k h[k]\right)^2\right] &= \sum_k p_h[k] + \sum\sum\nolimits_{k\neq l} p_h[k]p_h[l] \\ &= \sum_k p_h[k] + \left(\sum_k p_h[k]\right)^2 - \sum_k p_h[k]^2 \end{aligned} \tag{2.63}$$

so (2.59) is calculated as

$$\mathbf{Var}(\hat{k}_0) = \sum_k (p_h[k] - p_h[k]^2)\,. \tag{2.64}$$

Approximating the sum by the integral using (2.8),

$$\begin{aligned}\mathbf{Var}(\hat{k}_0) &\approx \int_{-\infty}^{\infty}\left(p_h\left(\frac{a(k-k_0)}{\sigma_n}\right) - p_h\left(\frac{a(k-k_0)}{\sigma_n}\right)^2\right)dk \\ &= \frac{\sigma_n}{a}\int_{-\infty}^{\infty}\left(p_h(v) - p_h(v)^2\right)dv \end{aligned} \tag{2.65}$$

The integral is a constant depending on the noise distribution. For logistic noise (2.27) we can get an exact value

$$\int_{-\infty}^{\infty}\left(p_{hl}(v) - p_{hl}(v)^2\right)dv = \frac{\sqrt{3}}{\pi} \approx 0.551329 \tag{2.66}$$

For Gaussian noise (2.19), it is

$$\int_{-\infty}^{\infty} \left(p_{hg}(v) - p_{hg}(v)^2 \right) dv = \frac{5}{6\sqrt{\pi}} \approx 0.470158 \tag{2.67}$$

Though the CT estimation (2.50) is efficient only for logistic noise, the variance is smaller in Gaussian noise.

Next, we calculate the variance of the noise variance estimation (2.51). Let define Y

$$Y = \hat{\sigma}_n^2/(4a) \tag{2.68}$$

and let $k_0 = 0$ to simplify the calculation. The result is unchanged.

$$\mathbf{Var}(Y) = \mathbf{E}\left[Y^2\right] - (\mathbf{E}\left[Y\right])^2 \tag{2.69}$$

The first term becomes

$$\begin{aligned} \mathbf{E}\left[Y^2\right] &= \mathbf{E}\left[\left(\sum_{k<0}(-k)h[k] + \sum_{k>0} k(1-h[k])\right)^2\right] \\ &= \mathbf{E}\left[\left(\sum_{k<0}(-k)h[k]\right)^2\right] + 2\,\mathbf{E}\left[\left(\sum_{k<0}(-k)h[k]\right)\left(\sum_{k>0} k(1-h[k])\right)\right] \\ &\quad + \mathbf{E}\left[\left(\sum_{k>0} k(1-h[k])\right)^2\right] \end{aligned} \tag{2.70}$$

Exchanging the order of square and expectation using the similar technique in (2.63),

$$\mathbf{E}\left[Y^2\right] = (\mathbf{E}\left[Y\right])^2 + \sum_k k^2(p_h[k] - p_h[k]^2) \tag{2.71}$$

Substituting (2.71) to (2.69) and using (2.68) we get

$$\mathbf{Var}\left[\hat{\sigma}_n^2\right] = 4a^4 \sum_k k^2(p_h[k] - p_h[k]^2) \tag{2.72}$$

What we want is $\mathbf{Var}\left[\hat{\sigma}_n\right]$, which is not a square root of (2.72). It takes a little bit more steps. Let consider $\hat{\sigma}_n^2$ as a random variable expressed as

$$\hat{\sigma}_n^2 = \sigma_n^2(1 + X) \tag{2.73}$$

where X is also a random variable. If the estimation $\hat{\sigma}_n^2$ is good, $|X| << 1$. Then

$$\begin{aligned}
\mathbf{Var}\left[\hat{\sigma}_n\right] &= \mathbf{Var}\left[\sigma_n\sqrt{1+X}\right] \\
&\approx \sigma_n^2\mathbf{Var}\left[1+\frac{X}{2}\right] \\
&= \frac{\mathbf{Var}\left[\sigma_n^2 X\right]}{4\sigma_n^2} \\
&= \frac{\mathbf{Var}\left[\sigma_n^2(1+X)\right]}{4\sigma_n^2} \\
&= \frac{\mathbf{Var}\left[\hat{\sigma}_n^2\right]}{4\sigma_n^2}
\end{aligned} \tag{2.74}$$

so

$$\mathbf{Var}\left[\hat{\sigma}_n\right] = \frac{a^4}{\sigma_n^2}\sum_k k^2(p[k]-p[k]^2)\,. \tag{2.75}$$

Approximating the sum (2.75) with the integral,

$$\begin{aligned}
\mathbf{Var}\left[\hat{\sigma}_n\right] &\approx \frac{a^4}{\sigma_n^2}\int_{-\infty}^{\infty}\left(\frac{v}{a}\right)^2\left(p_h\left(\frac{v}{\sigma_n}\right)-p_h\left(\frac{v}{\sigma_n}\right)^2\right)\frac{dv}{a} \\
&= a\,\sigma_n\int_{-\infty}^{\infty}\left(p_h(v)-p_h(v)^2\right)dv
\end{aligned} \tag{2.76}$$

The integral in (2.76) is the same within the CT level estimation variance (2.65).

As a result,

$$\mathbf{Var}(\hat{k}_0) \approx \frac{0.5\,\sigma_n}{a} \tag{2.77}$$

$$\mathbf{Var}\left[\hat{\sigma}_n\right] \approx 0.5\,a\,\sigma_n \tag{2.78}$$

are appropriate for both Gaussian and logistic noise. The equation (2.77) looks against the intuition the steeper inclination a gives the lower estimation error. The fact is, depending on a, data count becomes $1/a$ while the standard deviation becomes $1/\sqrt{a}$ so that the accuracy also becomes $1/\sqrt{a}$ in effect.

2.7 Numerical experiments on estimation variance

From (2.77), the CT level estimation error is inversely proportional to the ramp slope a. To confirm the result, I executed numerical experiments. Main body of the experiment is described in Perl script. The comparator model in Fig. 2.4 is one sentence.

Table 2.2: The estimation variance of the CT level and the noise

CT level variance	$\mathbf{Var}(\hat{k}_0) = \dfrac{\sigma_n}{a} K$
Noise variance	$\mathbf{Var}\left[\hat{\sigma}_n\right] = a\,\sigma_n K$
K for noise CDF p_h	$\displaystyle\int_{-\infty}^{\infty} \left(p_h(v) - p_h(v)^2\right) dv$
K for logistic noise	$\dfrac{\sqrt{3}}{\pi} \approx 0.551329$
K for Gaussian noise	$\dfrac{5}{6\sqrt{\pi}} \approx 0.470158$

```
$compout[$j] = ( ( ($i * $slew) - $sigma * nrand() ) < 0) ? 0 : 1;
```

where nrand() is a noise function, which is Gaussian noise in this case. Specifically for Gaussian noise generation, I used "rand()" function in Perl for uniform random numbers in [0, 1) and Box-Muller method. One trivial comment is Box-Muller method has to exclude 0 as a source random number. Perl has a useful expression to avoid rand()==0.

```
do {$x = rand()} while ($x == 1 || $x == 0);
```

though the possibility is very low so that the care might be unnecessary.

For logistic noise, I adopt the transformation method by the inverse of the logistic distribution (2.27)

$$p_{hl}^{-1}(x) = \frac{\sqrt{3}}{\pi} \log\left(\frac{x}{1-x}\right) \tag{2.79}$$

where x is a uniform random number in [0, 1) that rand() function generates.

The Perl script in Table 2.3 is the core part of the data processing. This is a straight implementation of (2.50) and (2.51). The array `$cout` contains a binary sequence of comparator outputs. It can be a measured data of ADC reduced to a binary sequence for each CT. The array `$sum` counts occurrences of 0. The moment to the opposite values are accumulated to `$sumi`. At the exit of the routine, `$sum` has $\hat{k}_0$ and `$sumi` times $2a^2$ becomes $\hat{\sigma}_n^2$.

Fig. 2.15 shows a result for CT level estimations by (2.50). The ordinate shows the CT level estimation variance and abscissa is the slope a. The variance σ_n is fixed to 1. Data counts in each

Table 2.3: Perl script of code transition estimation

```
my  $sum = 0;
my  $sumi = 0;
for $i (0..$#cout) {
    if ($cout[$i]==0) {
        $sum++;
    }
}
for $i (0..($sum - 1)) {
    $sumi += $cout[$i] * ($sum - $i);
}
for $i (($sum + 1) .. $#cout) {
    $sumi += (1-$cout[$i]) * ($i - $sum);
}
```

slope before and after the CT level are $4/a$ respectively. Doing so, the slope covers $\pm 4\sigma_n$ that insures the comparator outputs start at stack-to-0 and end at stack-to-1. Variance is calculated from 1000 times repetitions for each a. The line is the theoretical equation (2.77) that matches experimented data. Fig. 2.16 shows a result for σ_n estimations by (2.51). This result also shows a good coincidence. Gaussian noise has a tendency of slightly smaller estimation error than logistic noise as theoretical calculations (2.66) and (2.67) had shown.

2.8 Comparison with the conventional noise simulation

Let us compare the noise estimation (2.51) with the conventional histogram method. This example becomes also an application of the equation for a comparator design.

Fig. 2.17 is a simulated schematic cited from [28]. This is one of modern dynamic latches with double tails. The first stage is a differential amplifier. While the CLP is low, the amplifier state is reset that clears the hysteresis. The second stage is a comparator to generate logic level signals and the third stage is a SR latch to hold the compared result. The device parameters are typical for TSMC 0.18 μm. The dimension of the input differential NMOS pair is W/L=10 μm / 1 μm, and other transistors are PMOS W/L=3 μm / 0.18 μm and NMOS W/L=1.5 μm / 0.18 μm. These parameters are not tuned so well, only the PMOS loads of the input differential pair are multiplied by two to enhance the reset speed.

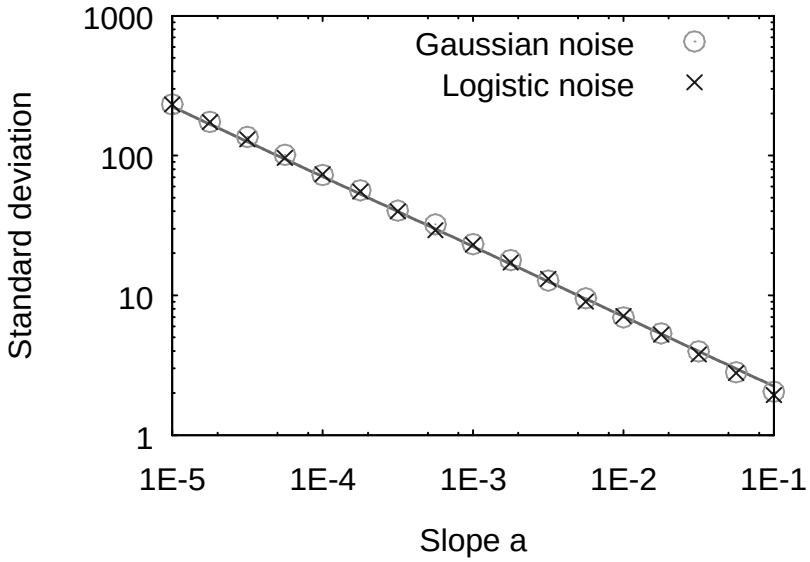

Figure 2.15: Transition point estimation error vs. slope *a*

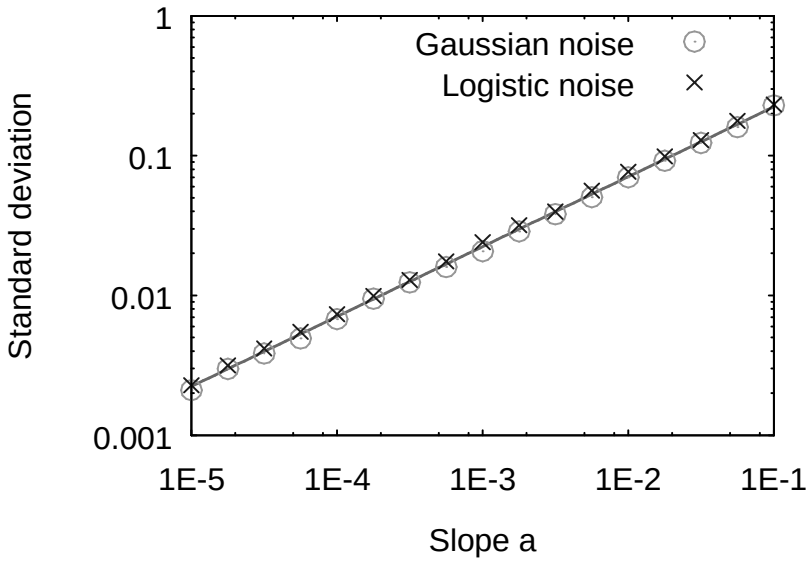

Figure 2.16: Noise estimation error vs. slope *a*

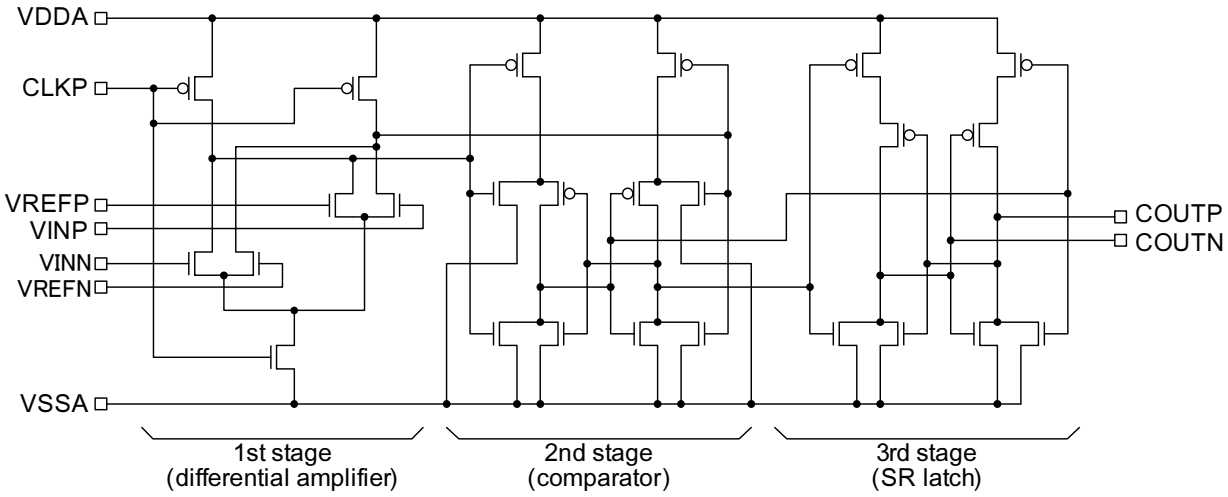

Figure 2.17: Schematics of the simulated latch

The transient simulation with noise option is executed. Fig. 2.18 shows a result. The clock frequency is 500 MHz. The noise parameters are noisefmax=10G, noisefmin=1, and noisetmin=0.5n. The test bench is scripted by Verilog-A and the comparator outputs are collected also by the Verilog-A script as a series of binary data.

Voltages are VDDA=2.5 V, VREFP=VREFN=1.0 V. The input is a differential ramp signal and the common voltage is 1.0 V. The initial values are VINP=0.9995 V and VINN=1.0005 V. The value exchanges after 20000 clocks, so the slew rate $a = 0.1\,\mu$V/clk and there are 10000 clocks each before and after crossing. The ramp condition is decided after several trials. The comparator outputs are 0's at first and 1's finally. Under these boundary conditions, the vibrations around CT level have better be as many as possible.

Post processing is done by a Perl script similar as described in the previous section. The result is $\sigma_n = 187.128031\,\mu$Vrms. From the equation (2.78), $\mathbf{Var}\,[\hat{\sigma}_n] \approx 0.5 \cdot 0.1 \cdot 187.128031\,\mu\text{V}^2/\text{clk}$, then $\hat{\sigma}_n \approx 3.1\,\mu\text{V}/\sqrt{\text{clk}}$. Thus we can expect the effective digit of this σ_n estimation is about two.

Points in Fig. 2.19 shows simulated data with traditional histogram method that counts the occurrence of comparator output being high for each reference level. The same range with the ramp method is divided into 20 levels. At each level, the comparator is simulated for 1000 clocks. So the total data count is set to same with the ramp method. The curve in Fig. 2.19 is a theoretical Gaussian noise of $\sigma_n = 187.128031\,\mu$Vrms. It looks a good coincidence.

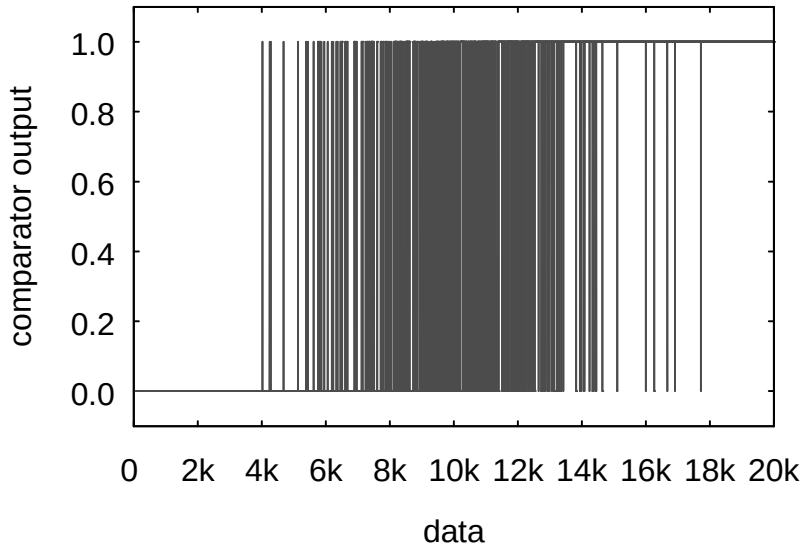

Figure 2.18: Transient noise simulation output of a latch output sequence

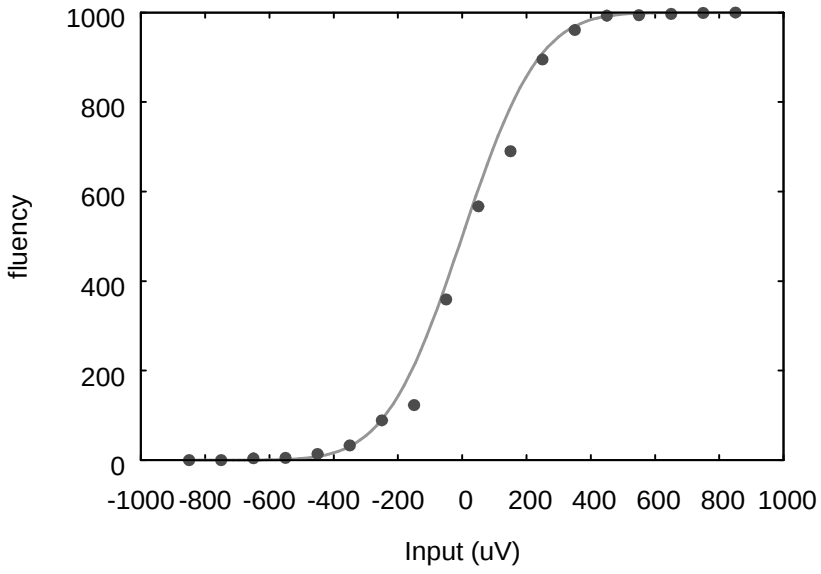

Figure 2.19: Histogram of comparator output vs. input voltage

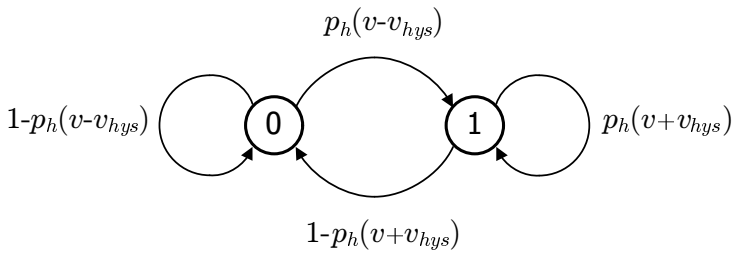

Figure 2.20: Comparator state change diagram

2.9 Noise estimation with comparator hysteresis

Modern CMOS comparators have very small hysteresis so that the necessity to care about hysteresis has been reduced recently. However there exists an unavoidable hysteresis mechanism as far as the comparator has to drive the load latch and to invert its state according to the comparison result. The mechanism tends to become outstanding at the highest rate of conversion rate. If hysteresis exists, results in this chapter have to be modified.

The hysteresis effect is expressed as the reference level shift. The probability $p_h(v)$ that the comparator output is 1 with hysteresis $\pm v_{hist}$ is expressed as

$$p_h(v) = \begin{cases} p_h(v + v_{hys}) & \text{if the previous state is 0} \\ p_h(v - v_{hys}) & \text{if the previous state is 1} \end{cases} \tag{2.80}$$

where v_{hys} is normalized by noise σ_n. Fig. 2.20 depicts the comparator state changes more comprehensively. The circled 0 and 1 stand for the comparator state. Arrows from and into them indicate the state change. Each equation attached to the arrow is the probability the state change happens.

At the equilibrium state, which we can expect when the input changes slowly enough, the entrance rate into 1 and the exit rate from 1 are equal, so the probability of 1-state $\bar{p_h}(v)$ satisfies

$$(1 - \bar{p_h}(v))p_h(v - v_{hys}) = \bar{p_h}(v)(1 - p_h(v + v_{hys}))\,. \tag{2.81}$$

Solving (2.81) for $\bar{p_h}(v)$, we have the explicit expression of the equivalent transfer probability.

$$\bar{p_h}(v) = \frac{p_h(v - v_{hys})}{1 - p_h(v + v_{hys}) + p_h(v - v_{hys})}\,. \tag{2.82}$$

The $\bar{p_h}(v)$ keeps the properties of a distribution function. They are:

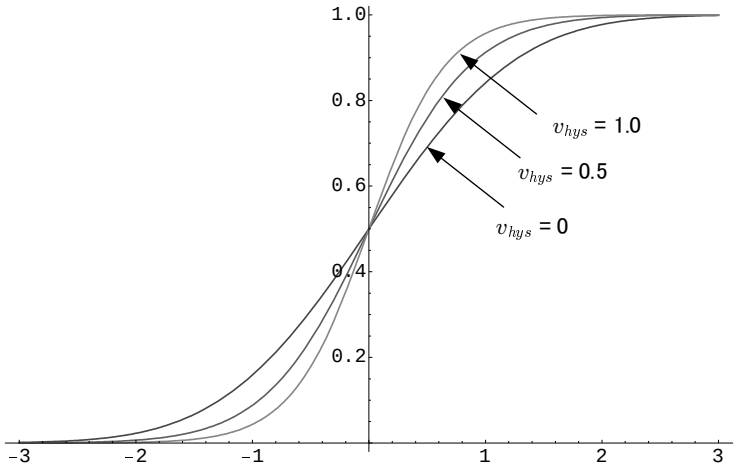

Figure 2.21: Some distribution functions with hysteresis for Gaussian noise $\sigma_n = 1$

- Non-decreasing
  If $v_1 < v_2$ then $\bar{p}_h(v_1) \leq \bar{p}_h(v_2)$

- Limit values

$$\lim_{v \to -\infty} \bar{p}_h(v) = 0$$

  and

$$\lim_{v \to \infty} \bar{p}_h(v) = 1$$

- Right-continuous

$$\lim_{v \to +v_1} \bar{p}_h(v) = \bar{p}_h(v_1)$$

Fig. 2.21 shows some distribution functions with hysteresis for Gaussian noise $\sigma_n = 1$ calculated by the equation (2.81). It looks the effective noise variance getting smaller as the hysteresis becomes larger. Logistic noise provides much the same curves. Fig. 2.22 supports this observation.

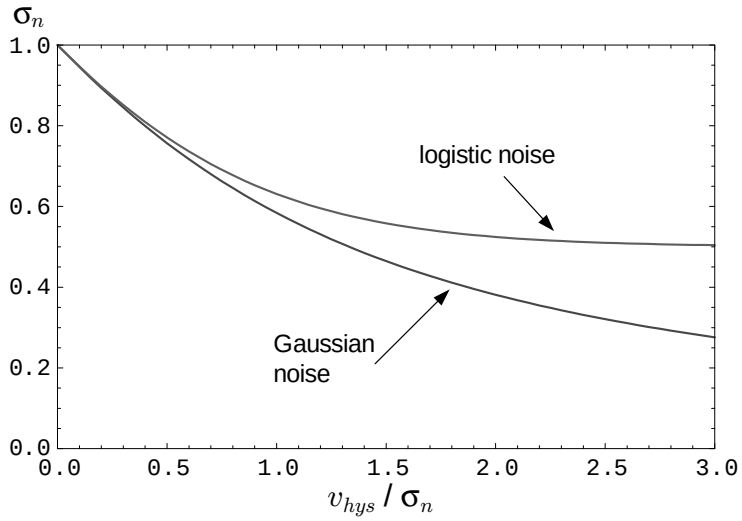

Figure 2.22: Noise variance vs. hysteresis: normalized as $\sigma_n = 1$

2.10 Summary

This chapter described the linearity analysis as a CT level estimation. The results are summarized as follows.

1. Breaking a ramp sequence into binary sequences of individual CT levels, we have derived two maximum likelihood equations (ML equations).

2. ML equations include weight functions determined by the noise distribution. Gaussian distribution has larger weight for rare case events. On the other hand, we have derived the logistic distribution that has even weight, or equivalently, ignoring the order of the sampled data.

3. For logistic noise, the ML equations are solvable analytically. In other words, the logistic approximation makes the ML equations solvable. Table 2.1 shows solutions.

4. One solution validates a histogram test for the CT level estimation. This seems natural since the histogram test ignores the sample order.

5. Another solution of the ML equation provides the explicit expression of the input equivalent noise.

6. The noise expression reflects the behavior around the CT level. It has a right to be the third index (sub-micro index) of linearity, following DNL (micro index) and INL (macro index).

7. Several applications are shown. They demonstrate the noise expression complements the drawback of the histogram method that ignores the sample order.

8. The equations obtained by the logistic approximation are valid for much wider class of noise, including Gaussian noise, in expectation sense.

9. Variances of the estimation are provided explicitly. Exact constants both for logistic noise and Gaussian noise are obtained (Table 2.2). These formulas provide the confidence level of the estimation.

10. The noise expression is applied to a latch circuit noise simulation and compared to the conventional method. Good coincidence is obtained.

11. The fact that hysteresis of comparators diminish the input equivalent noise is examined.

The method in this chapter is developed for a ramp input here. Theoretically, we can extend the input to any signal as far as we can pose the probability on each sample. The signal can be high frequency sine wave. Combined with the reconstruction technique unrolled in Chapter 3, we will be able to analyze the noise properties quite minutely at high frequencies. This extension is the future work.

Chapter 3

Distortion analysis with the use of difference waves

Distortion is usually carried out in frequency domain. However as we have seen in Chapter 1, it has rather poor sensitivity for ADC analysis. Time domain analysis has much finer resolution by providing deviations of individual samples. The “waveform reconstruction” technique is a basic tool for time domain analysis. In this chapter, we establish the most general reconstruction method that I named “Euclidean reordering.” The technique has very old roots, so as it is difficult to identify a specific origin. I compose the most thorough description of the theory. Combining the Euclidean reordering method and the frequency domain analysis, we introduce the “difference wave analysis.” Applications of the methods with subsidiary analysis techniques demonstrate the ability.

3.1 Theory of the waveform reconstruction

Permutation methods are the method that reorders the series of sampled data taking advantage of special relations between input and clock frequencies. As such, coherent sampling is appropriate, though it is not mandatory. Fig. 3.1 shows an example. The input sine wave is sampled at a constant interval in time order. Fifteen clock periods match three input periods exactly. Numbers at sampling points in the figure show the phase order of the input. If we reorder the samples in these numbers, we have fifteen samples in an input period. This is three times finer resolution compared to the original that has roughly four samples in a period. The permutation methods are individuated by the setting of input and clock frequencies and the attendant algorithm.

In coherent sampling, the ratio of the input frequency f_{in} and the clock frequency f_{clk} is

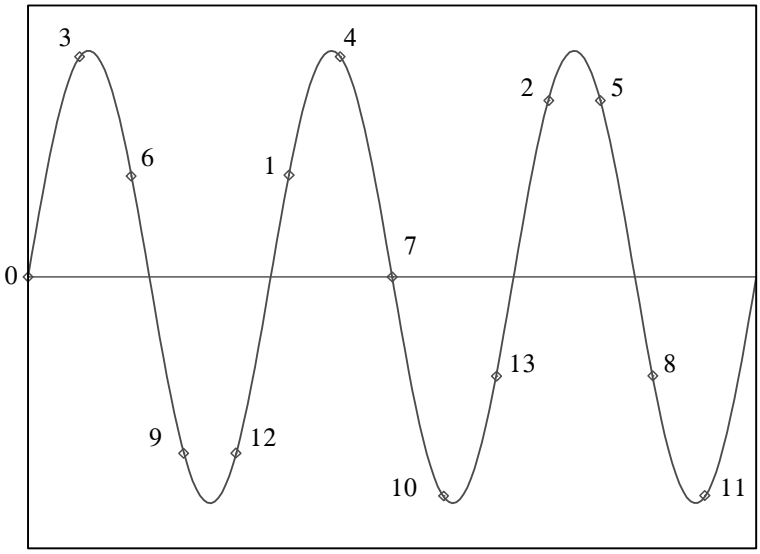

Figure 3.1: An example of sampling: 3 input periods for 14 samplings

expressed as multiples of the common base frequency f_0,

$$f_{in} = n_{in}\, f_0 \tag{3.1}$$
$$f_{clk} = n_{clk}\, f_0\,, \tag{3.2}$$

where n_{in} and n_{clk} are positive integers. These numbers are more essential than frequencies in the theory. Let us call them as the *input frequency number* and the *clock frequency number* respectively.

IEEE Std 1241-2010 [5] provides two specific permutation methods in "Section 4.4 Equivalent-time sampling and undersampling." The standard presents two more techniques in "Annex D (informative) Presentation of sine-wave data." One is "modulo-time plot" [17] and the other is used by Blair [21]. They are different in appearance and they have different limitations. However they can be treated uniformly in one theory.

In one permutation method in IEEE Std 1241-2010, the clock frequency number n_{clk} in (3.2) is about a multiple of the input frequency number n_{in} in (3.1) but smaller just by one, that is $n_{clk}/n_{in} = (LD-1)/D$ where L and D are integers. In practice, D is set to n_{in}. Then we have the condition that n_{in} and n_{clk} satisfy:

$$n_{in}\, L = n_{clk} + 1. \tag{3.3}$$

The condition of Fig. 3.1 is $n_{in} = 3$ and $n_{clk} = 14$. So $L = 5$ satisfies the equation (3.3).

For reordering, we look in Fig. 3.1 and find that first samples at every period form a phase order. The second samples have the same property. This property will be more easily seen by Table 3.1. Sampled data number are placed sequentially from left to write in each raw and then from top to bottom. Reordering takes the data from top to bottom in each column and then from left to right.

Table 3.1: A sampling rectangle: 14 samples in 3 periods

0	3	6	9	12
1	4	7	10	13
2	5	8	11	(0)

In an interpretation, the equation (3.4) is a rectangular condition that the length is n_{in} and the height is L. The area becomes $n_{clk} + 1$. The difficulty in this method to find appropriate n_{in} and n_{clk} that suit for the evaluation targets, thought it is not very difficult.

The equation (3.3) can be written as

$$L\, n_{in} - n_{clk} = 1. \tag{3.4}$$

This form is convenient as a standard form in this section.

Another extraction method in IEEE Std. 1241-2010 is equivalent to the envelope test in essence. In this case, the condition is

$$-n_{in} + n_{clk} = 1 \tag{3.5}$$

or it is also possible that

$$n_{in} - n_{clk} = 1. \tag{3.6}$$

We do not have to reorder the sampled data in these cases. The sampled waveform looks same for sine wave input in (3.5) and (3.6). The fact is the order is opposite. Since the sine wave is time-reflection symmetric, we do not notice the difference. Equation (3.6) is a special case of (3.4) where $L = 1$.

"Modulo time plot" [17] does not reorder the measured data in narrow sense. Instead, it plots the measured data by residue of the input phase as x-coordinate. Inherently the data are not arranged in the sampled order and the resulting plot consists of points that are disconnected. This method is implemented in MATLAB [29] and available through the Internet.

The k-th sample phase ϕ_k is

$$\phi_k = \frac{f_{in}}{f_{clk}} k + \phi_0 = \frac{n_{in}}{n_{clk}} k + \phi_0 \tag{3.7}$$

where the period is normalized as 1. We can put $\phi_0 = 0$ without losing generality as far as the phase difference matters. So the residue of ϕ_k is expressed using a fraction $\frac{r}{n_{clk}}$

$$\frac{f_{in}}{f_{clk}}k + v = \frac{r}{n_{clk}} \tag{3.8}$$

or

$$k\,n_{in} + v\,n_{clk} = r \tag{3.9}$$

where r and v are integers and $0 \le r < n_{clk}$. The equation (3.9) is rewritten as

$$k\,n_{in} = -v\,n_{clk} + r \tag{3.10}$$

which is a standard integer division formula. The dividend is $k\,n_{in}$, the divisor is n_{clk}, the quotient is $-v$ and the residue is r. Equivalently modulo time plot calculates r and reordering is executed when the data is plotted according to r.

Inspired by the method, Pápay [19] [20] proposed a direct reordering method. He called it "uniform permutation" of samples following the literature [30] in 1977. His method is identical with our method that we call "Euclidean reordering." We have already used it for over twenty years without knowing prior works. The same method is commercially available, *e.g.* the ADC toolbox by Synopsis [22].

"Euclidean reordering" is named from Euclidean algorithm in elementary number theory. As we soon show in (3.22), the frequency condition in Euclidean reordering relaxes the equation (3.4) essentially to the limit for the permutation method.

It is well-known that if n_{in} and n_{clk} are relatively prime, sampling time is uniformly spaced over an input period [5, 5.4] [17] [21]. The term 'uniformly spaced' seems ambiguous. We state it precisely in Theorem 1.

Theorem 1 (Maximum different phase theorem) *When sampling a periodical signal of the input frequency number n_{in} by the clock of n_{clk}, and let n_0 be the greatest common divider (gcd) of n_{in} and n_{clk},*

$$n_0 = \gcd(n_{in}, n_{clk}) \tag{3.11}$$

then successive n_{clk}/n_0 samples have all different phases of the input signal, while no further sample has a different phase. As a special case if n_{in} and n_{clk} are relatively prime, all successive n_{clk} samples have different phases of the input signal.

Proof 1 *Since the initial phase does not change how many different phases are sampled at given ratio of n_{in} and n_{clk}, we assume the first sample phase is zero. Then, the phase of the k-th sampled data p_k is*

$$p_k = \frac{2\pi k\,n_{in}}{n_{clk}}. \tag{3.12}$$

From periodical nature of the input signal, only the residue of $k\,n_{in}$ by n_{clk} matters. That is to say, if the residue is different, the phase is different, and vice versa. In other words, all samples can be tagged by integers from 0 to $n_{clk}-1$ of the phase order. Understanding this principle, the rest of the proof is a standard in elementary number theory.

Consider the minimum integer $k' > 0$ that satisfies

$$k'\,n_{in} \equiv 0 \pmod{n_{clk}} \tag{3.13}$$

where $\equiv$ is a notation of congruence, c.f. [31].

Existence of k' is easy to see by translating (3.13) into the following Diophantine (integer) equation

$$k\,n_{in} = q\,n_{clk} \tag{3.14}$$

which has infinite number of solutions, particularly $k = n_{clk}$ and $q = n_{in}$.

All elements in $\{k\,n_{in} \pmod{n_{clk}} \mid k = 0, 1, 2, \cdots, k'-1\}$ are different, because if

$$k_1\,n_{in} \equiv k_2\,n_{in} \pmod{n_{clk}} \tag{3.15}$$

for $0 \le k_1 < k_2 < k'$, then

$$(k_2 - k_1)\,n_{in} \equiv 0 \pmod{n_{clk}} \tag{3.16}$$

that $k_2 - k_1$ satisfies equation (3.13) and $0 < k_2 - k_1 < k'$, which contradicts the minimality of k'. On the other hand from (3.13),

$$k' n_{in} \equiv 0\,n_{in} \pmod{n_{clk}} \tag{3.17}$$

so $k = k'$ and $k = 0$ provide the same congruence classes. As a result, there are just k' elements in $\{k\,n_{in} \pmod{n_{clk}} \mid k \in \mathbf{N}\}$, where $\mathbf{N}$ is a set of whole integers.

From the assumption (3.11),

$$n'_{in} = \frac{n_{in}}{n_0} \tag{3.18}$$

$$n'_{clk} = \frac{n_{clk}}{n_0} \tag{3.19}$$

are relatively prime integers. Substituting (3.18)(3.19) into (3.14)

$$k\,n'_{in} = q\,n'_{clk} \tag{3.20}$$

so k is a multiple of $n'_{clk} = n_{clk}/n_0$. Among those k's, k' is determined as

$$k' = n_{clk}/n_0 \tag{3.21}$$

by the minimality requirement of k'.

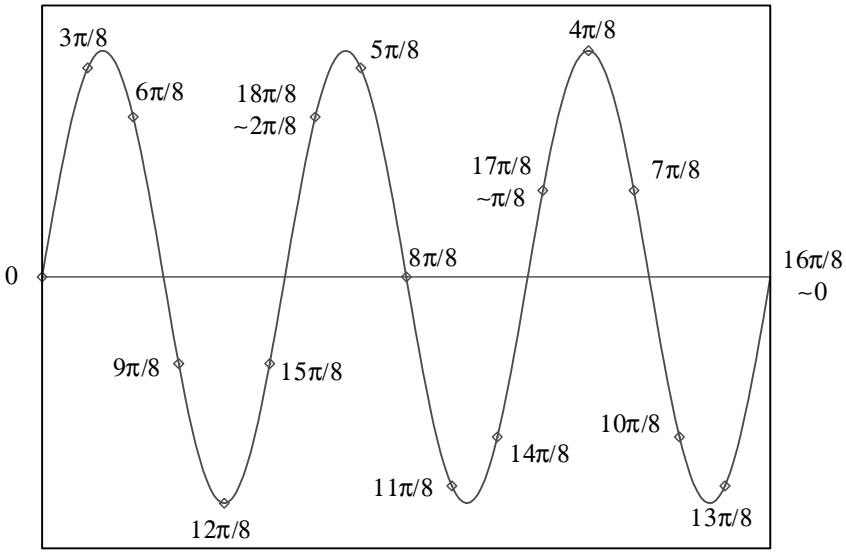

Figure 3.2: An example of the sampling phases, $n_{in} = 3$, $n_{clk} = 16$

Fig. 3.2 shows an example of $n_{in} = 3$, $n_{clk} = 16$. In a similar way as we assumed in the previous proof, the first sample is taken at the input phase 0. The sampling step is $2\pi n_{in}/n_{clk} = 3\pi/8$. The phase sequence $\{p_k\}$ is tagged by the following integer sequence.

$$\begin{aligned} & 0,\ n_{in},\ 2\,n_{in},\ 3\,n_{in},\ 4\,n_{in},\ 5\,n_{in},\ \cdots \pmod{n_{clk}} \\ \equiv\ & 0, 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, \\ & 39, 42, 45\ \cdots \pmod{16} \\ \equiv\ & 0, 3, 6, 9, 12, 15, 2, 5, 8, 11, 14, 1, 4, 7, 10, 13, 0, \cdots \end{aligned}$$

This sequence includes all integers from 0 to 15 ($= n_{clk} - 1$) in a period as proved.

A special case of interest that satisfies the maximum different phase theorem is the condition that n_{clk} is a power of two and n_{in} is odd. This condition provides easy and flexible frequency settings compared to the IEEE method. In addition, this sample count is convenient for FFT post-processing.

Unlike the IEEE method, the general reconstruction process for the condition in Theorem 1 is not so obvious. In fact, the procedure is more like picking up samples periodically rather than sorting. This is the operation that literature [30] [19] called “uniform permutation.” Our term is “cyclic pickup” of the sampled data sequence of the length n_{clk}. The word ‘cyclic’ means to treat the sequence as a loop. In other words, the first data in the sequence is deemed to be the next data of the last data. Theorem 2 provides the foundation of the procedure.

Theorem 2 (Cyclic pickup) *Let the input frequency number n_{in} and the clock frequency number n_{clk} be relatively prime. Then there is an integer u that cyclic pickup of every u-th data provides the data in the input phase order. The number u is determined as a solution of the Diophantine equation*

$$u\, n_{in} + v\, n_{clk} = 1. \tag{3.22}$$

All solutions of (3.22) are equivalent as a cyclic pickup. If u is negative, the u-th pickup means the reverse direction pickup.

Proof 2 *As mentioned in the proof of Theorem 1, we can label the sampled sequence as an ordered set of integers*

$$D = \{k\, n_{in} \pmod{n_{clk}} \,|\, k = 0, 1, 2, \cdots, n_{clk} - 1\}. \tag{3.23}$$

Each value of the element in D is the phase order of samples, assuming the first sampling phase is zero. Since n_{in} and n_{clk} are relatively prime, D includes all integers from 0 to $n_{clk} - 1$. So there is an integer u that satisfies

$$u\, n_{in} \equiv 1 \pmod{n_{clk}} \tag{3.24}$$

from Theorem 1. The equivalent translation of (3.24) provides the equation (3.22). Counting the first data as 0, u-th data in D has the first non-zero phase of the input.

The next u-th 'cyclic' pickup from D satisfies

$$2\, u\, n_{in} \equiv 2 \pmod{n_{clk}} \tag{3.25}$$

that is the second phase of the input. Thus the repetition of this operation provides the rearrangement of the sampled data in phase order.

If the first sampling phase is not zero, the resulting data sequence starts at the non-zero phase, but it is arranged in phase order in cyclic mean nevertheless.

If (u, v) is a solution of (3.22), it is easy to see $(u + n_{clk}, v - n_{in})$ is also a solution of (3.22). Repeatedly, we can make infinite number of solutions of the equation (3.22). However from the nature of the cyclic pickup, u-th reverse pickup is equivalent to the $(u + n_{clk})$-th forward pickup. Thus we see the solutions of (3.22) provide a unique cyclic pickup and we do not have to worry about the polarity of u.

When $n_0 = \gcd(n_{in}, n_{clk}) > 1$, obviously n'_{in} and n'_{clk} in (3.18)(3.19) takes the same role in Theorem 2. That is, any frequency ratio is permissible, but effective samples are diminished depending on the frequency ratio.

In the condition of Fig. 3.2, that is $n_{in} = 3$, $n_{clk} = 16$, a set of solutions of the equation (3.22) is $u = 11$ and $v = -2$. As Theorem 2 declares, the phase difference in eleven sample interval, $3 \cdot 11 \cdot \pi/8 \equiv 17\pi/8 \equiv \pi/8$, is the unit phase distance in the reconstructed waveform.

The equation (3.22) includes the IEEE Std condition (3.4) as a special case. In this meaning, Euclidean reordering is a true generalization of the IEEE method. The modulo time plot equation (3.9) is the multiple of (3.22) by r. The condition for modulo time plot is identical with Euclidean reordering. The method directly calculates the phase number r instead of finding the pickup interval u.

The Euclidean algorithm is a well-known algorithm that provides a solution of the equation (3.22). We provide a compact proof in a convenient form for implementation.

Theorem 3 (Euclidean algorithm) *Let $a_0 = n_{in}$ and $a_1 = n_{clk}$. Repeat the calculation of the quotient q_k and the remainder $0 \le a_{k+1} < a_k$, which satisfies*

$$a_{k-1} = q_k a_k + a_{k+1} \quad \text{for } k = 1, 2, \cdots \tag{3.26}$$

until $a_{n+1} = 0$. Then u and v that satisfy the equation (3.22) are provided as

$$\begin{pmatrix} u & v \\ * & * \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_n \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -q_{n-1} \end{pmatrix}\cdots\begin{pmatrix} 0 & 1 \\ 1 & -q_1 \end{pmatrix} \tag{3.27}$$

where asterisks denote the elements that we do not matter and that are usually different.

Proof 3 *The matrix form of the equation (3.26) is*

$$\begin{pmatrix} a_k \\ a_{k+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_k \end{pmatrix}\begin{pmatrix} a_{k-1} \\ a_k \end{pmatrix} \tag{3.28}$$

This operation keeps the gcm of a_{k-1} and a_k, so the gcm of $a_0 = n_{in}$ and $a_1 = n_{clk}$. Since the sequence $\{a_k\}$ forms strictly reducing positive integers multiplied by this common gcm, it must reach zero value. If $a_{n+1} = 0$, then $a_n = \mathrm{gcm}(n_{in}, n_{clk}) = 1$. That is

$$\begin{aligned} \begin{pmatrix} 1 \\ 0 \end{pmatrix} &= \begin{pmatrix} 0 & 1 \\ 1 & -q_n \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -q_{n-1} \end{pmatrix}\cdots \\ &\quad \begin{pmatrix} 0 & 1 \\ 1 & -q_1 \end{pmatrix}\begin{pmatrix} n_{in} \\ n_{clk} \end{pmatrix} \\ &= \begin{pmatrix} u & v \\ * & * \end{pmatrix}\begin{pmatrix} n_{in} \\ n_{clk} \end{pmatrix} \\ &= \begin{pmatrix} u\,n_{in} + v\,n_{clk} \\ * \end{pmatrix}. \end{aligned} \tag{3.29}$$

The second equation is obtained from the matrix multiplication from left to right that provides u and v as in (3.27). A part of the last equation is the equation (3.22).

The calculated u often becomes negative. If we prefer forward order picking-up, we can replace it by $u + n_{clk}$.

List in Table 3.2 is an implementation of Euclidean reordering. The Euclidean algorithm part is imported from an elegant embodiment in [31, Figure 6.1]. This is the multiplication of the matrices in (3.29) from right to left written in recursive form.

The input frequency number and the clock frequency number are stored in $a and $b beforehand respectively.

It is worth notifying that the Nyquist frequency has little role in ADC evaluation. Euclidean reordering works fine for any frequencies satisfying the equation (3.22), even if the input frequency is over the Nyquist frequency.

3.2 Reconstructed waveform and difference wave analysis

This section provides practical applications of the Euclidean reordering.

Data count in real analysis should be large enough to analysis. In simulation, samples may be limited by 256 through 1024 due to the CPU time. The number is usually enough for distortion analysis. In measurement, much larger samples are preferable for clearer view of the ADC properties. In modern measurement systems, 65536 samples are practical. This number seems sufficient for 6-bit ADC analysis. Larger number will become appropriate in future with further storage and faster processors. Accuracy of the measurement instrument is another bound. Jitter in the analog input and the clock will make further data meaningless.

Fig. 3.3 is a measured example of the reconstructed waveform of a 6-bit ADC, $f_{in} = 1.01133$ MHz and $f_{clk} = 100.27008$ MHz. The base frequency f_0 is 1.53 kHz, $n_{in} = 661$ and $n_{clk} = 2^{16} = 65536$. The output code of this ADC is binary from -31 to 31. The input amplitude is adjusted to slightly smaller than the full scale monitoring the logic analyzer display. The phase of the reconstructed waveform is not adjusted. As a result, the phase is randomly reconstructed. The phase will be easily adjusted.

The reconstructed waveform in Fig. 3.3 looks fine and it is difficult to recognize the distortion. A remedy is the “difference wave,” which shows the difference between the reconstructed waveform and the ideal wave. The point of the idea is the ideal wave. What is it and how do we get it? We will treat the question later in Chapter 4, but here we simply define the ideal wave as the best fit sinusoid in the least mean square (LMS) sense. The sinusoid is obtained by three parameter estimation method [5], or as a byproduct of FFT calculation [7]. Since there are numerous references on FFT, I describe the story briefly. Let the reconstructed waveform $w_r(\tau)$ where τ is the phase number from 0 to $n_{clk} - 1$. The discrete Fourier transform (DFT) theory

Table 3.2: Perl script of Euclidean reordering

```
########## Euclidean algorithm ###################################
    my  $m = 1;
    my  $g = $a;
    my  $v = 0;
    my  $w = $b;

    my  $t;
    my  $q;
    my  $s;

    while ($w != 0 ) {
        $q = int($g / $w);      # quotient
        $t = $g - $q * $w;      # remainder
        $s = $m - $q * $v;
        ($m, $g) = ($v, $w);
        ($v, $w) = ($s, $t);
    }

    if ($m < 0) { $m += $b; }
    my  $y = ($g - $a * $m) / $b;

    if ($g != 1) {
        printf(STDERR "Error: nin and nclk are not mutual primitive.\n");
        exit 1;
    }

########## reordering ########################################
    my  @indat = <>;

    if ($#indat+1 < $b) {
        printf(STDERR "Error: samples are too short. %g\n", $#indat);
        exit 2;
    }

    my  $i = 0;
    my  $ii = 0;
    do {
        do {
            print $indat[$ii + $i];
            $i = ($i+$m) % $b;
        } until ($i == 0);
        $ii += $b;
    } until ($#indat+1 < $ii + $b);
```

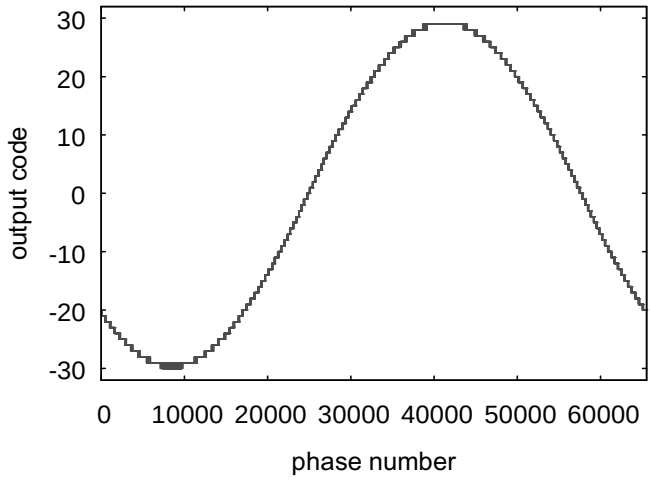

Figure 3.3: A reconstructed waveform of measured data: f_{in} = 1.01133 MHz and f_{clk} = 100.27008 MHz

states there is an expression of $w_r(\tau)$

$$w_r(\tau) = a_0 + \sum_{k=1}^{n_{clk}/2-1} (a_k \cos(2\pi k\tau/n_{clk}) + b_k \sin(2\pi k\tau/n_{clk})) . \tag{3.30}$$

The FFT is the algorithm that provides the DFT coefficients quite efficiently. The ideal wave $w_i(\tau)$ is the fundamental tone with the offset.

$$w_i(\tau) = a_0 + a_1 \cos(2\pi\tau/n_{clk}) + b_k \sin(2\pi\tau/n_{clk}). \tag{3.31}$$

From the general theory of orthogonal systems [32, 16.4] [33, Volume I, Chapter I], the coefficients a_0, a_1 and b_1 provide the LMS estimate of $w_r(\tau)$. Three parameter estimation provides the same coefficients by different way. The power of k-th spectrum $p(k)$ is

$$p(k) = a_k^2 + b_k^2 \quad k \in [0, n_{clk}/2 - 1] \tag{3.32}$$

where $b_0 = 0$. From the definition, the rest of the series (3.30) coincides the difference wave $w_d(\tau)$ in integer τ.

$$w_d(\tau) = \sum_{k=2}^{n_{clk}/2-1} (a_k \cos(2\pi k\tau/n_{clk}) + b_k \sin(2\pi k\tau/n_{clk})) . \tag{3.33}$$

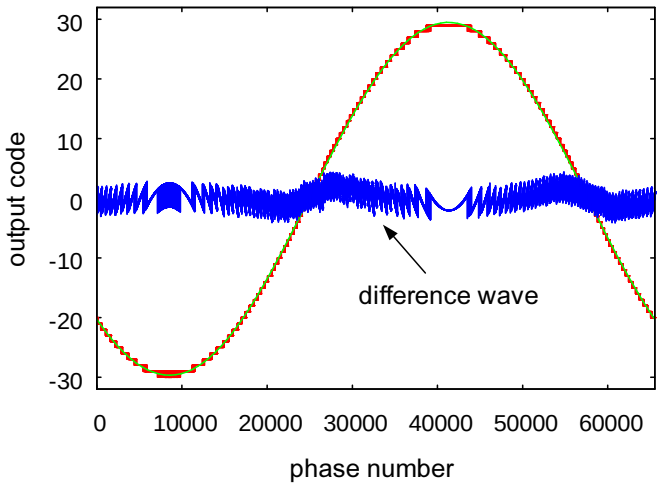

Figure 3.4: A reconstructed waveform with the difference wave: f_{in} = 1.01133 MHz and f_{clk} = 100.27008 MHz

The difference wave provides very valuable information. The analysis of the wave is a key in time domain analysis. The wave provides not only the distortion amplitude but also distortion form and phase relative to the input signal. In other words, it indicates how the ADC distorts. Thus the difference wave shows internal properties of the ADC.

Fig. 3.4 appended the ideal sinusoid and the difference wave in Fig. 3.3. The magnitude of the difference wave is magnified by 4 for visibility. Form this figure, we can recognize a warping in the difference wave.

One tip for analysis is that the difference wave thickness is roughly one LSB size. The distortion is gauged by this unit. From this tip, the warping in the difference wave is about 1 LSB peak-to-peak. Since the warping happens at low frequency for the ADC, the linearity error is suspected. The linearity plot in Fig. 3.5 shows 1-LSB warping in INL, which is consistent with the difference wave distortion. In fact, this ADC has a function to bend the linearity at code ±8. There was the design error that could not turn off the function completely.

More often, the warping happens at higher input frequency. Waves in Fig. 3.6 show an example. They are dated measured data of a 10-bit flash ADC. Samples in a period are only 1024. At low frequency input, the ADC provides fairly good performance. The SINAD in f_{in} = 0.5124 MHz is 57.49 dB. The steps in the difference wave perceptible at the peak and the bottom of the reconstructed waveform are dog-tooth we examine in Section 3.3. This is the first appearance of the phenomena but it had not identified yet at that time. Another character most

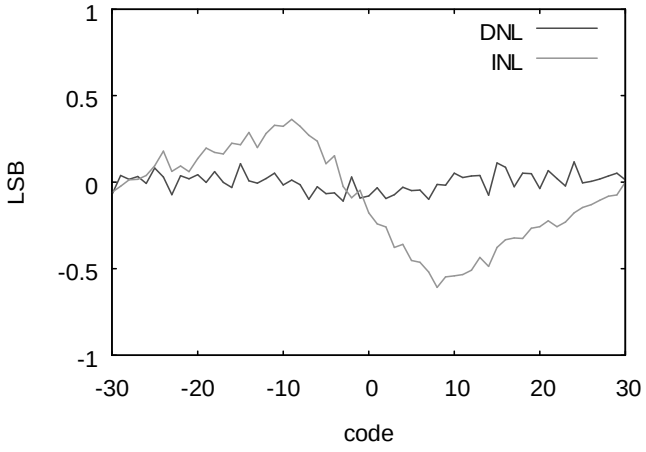

Figure 3.5: Linearity plot of the 6-bit ADC in Fig. 3.4

clearly seen in Fig. 3.6(d) is that the distortion is not symmetrical at rising slope and falling slope of the input. In this specific ADC, the cause of the asymmetry is identified, by further analysis assisted by theoretical calculations and simulation, as the internal coupling in the chip between the analog input line and the base line of the current sink of the comparators.

Sparkles found in Fig. 3.6(f) stem from delay mismatch in internal latched-comparator blocks. Another tip is, since one period of the reconstructed waveform corresponds to the input signal period $1/f_{in}$, the sampling period is equivalent to $1/(n_{clk}\ f_{in})$. From the timing error in Fig. 3.6(f) looks about 5 samples, we can evaluate the timing error as $5/(n_{clk}\ f_{in}) \approx 200$ psec.

Sometimes, timing jitter spoils SINAD. Fig. 3.7 shows an example. This ADC has a problem in its test clock system. Jitter is a noise of time direction. The feature of jitter is that the width of the difference wave becomes larger where the input slew rate is large. We can identify the tendency in Fig.3.6 (f). In a different point of view, while the thickness of the difference wave is much the same, we can reason that jitter of the measurement system is negligible.

On the other hand, additive noise appends a flat thickness on the difference wave as shown in Fig. 3.8. This ADC has a CMOS 6-bit flash ADC with self-calibration. Pursuing the highest performance, the comparator size is intentionally small and the calibration should compensate the errors. However several incidents, not fully identified, brought the failure of the compensation mechanism, resulting in such a large noise. Fig. 3.9 is the same data of discrete dots. This way of plot enables to see the ADC does not have missing codes (outputs of all codes in between) but codes are overlapped roughly four LSB.

Generally speaking, identification of malfunction mechanism is often challenging and cre-

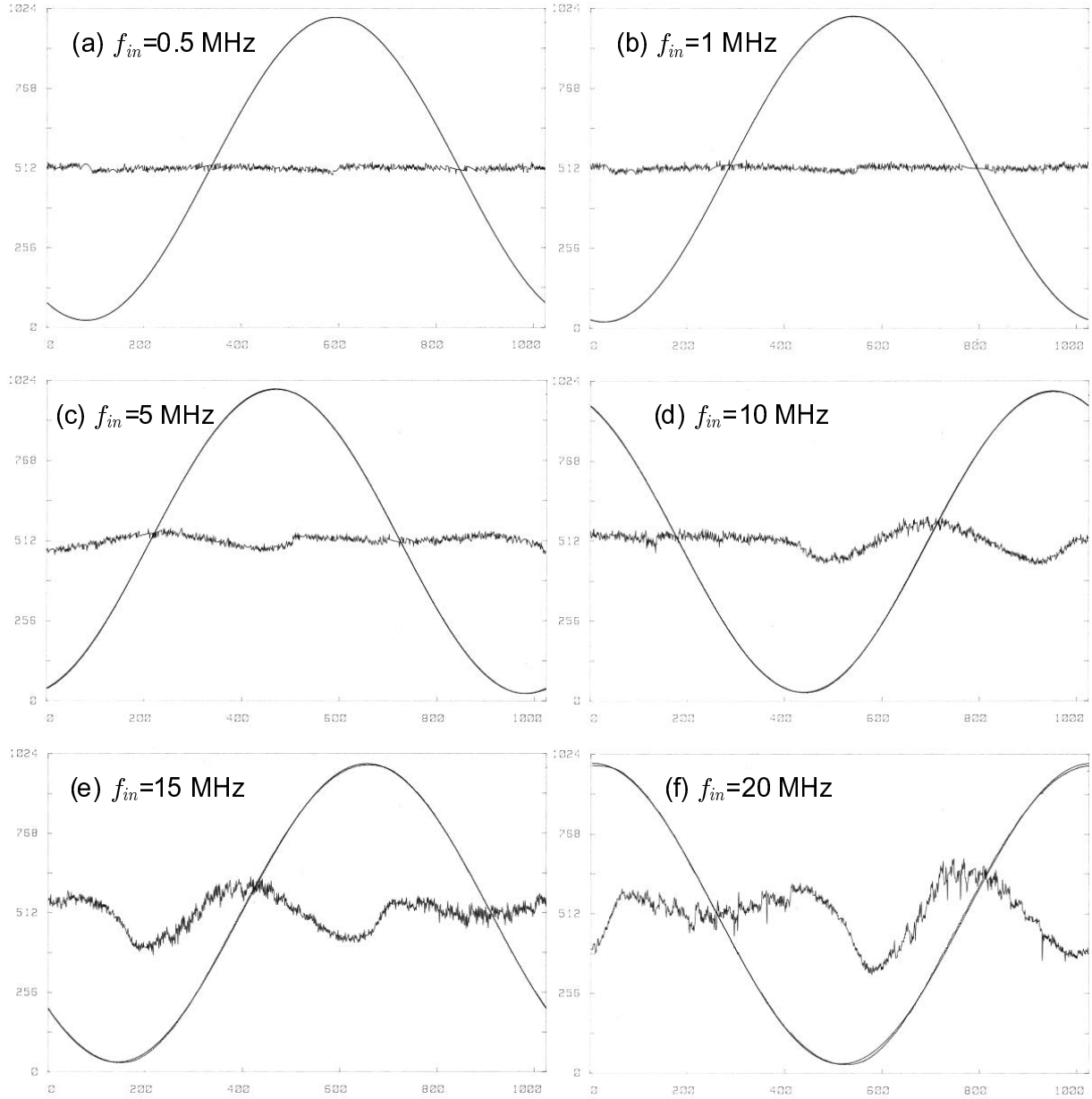

Figure 3.6: Input frequency dependency of the difference wave in a 10-bit flash ADC. Precisely, (a) f_{in} = 0.5124 MHz, (b) f_{in} = 1.0980 MHz, (c) f_{in} = 5.0508 MHz, (d) f_{in} = 10.0284 MHz, (e) f_{in} = 15.0060 MHz, (b) f_{in} = 19.9386 MHz, and thoroughly f_{clk} = 74.9568 MHz.

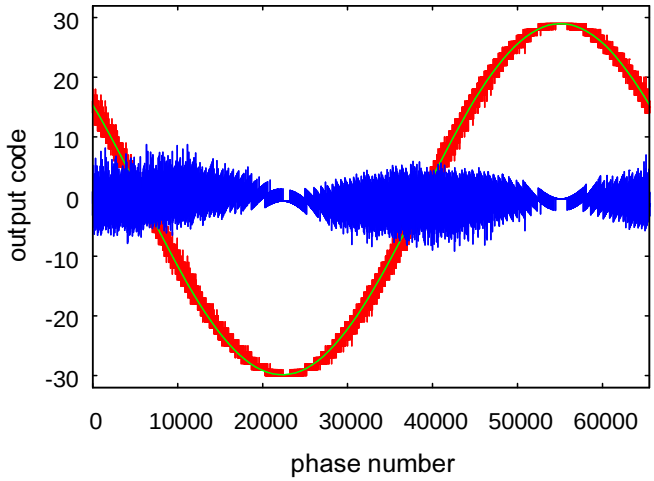

Figure 3.7: Difference wave affected by large jitter: f_{in} = 49.999640 MHz and f_{clk} = 99.614720 MHz

ative task. A single method including the difference wave analysis will not do all, even how strong it may be. Measurements in various conditions, such as the input frequency, the clock frequency, supply voltages, temperature and so on are important. The way to process the measured data is also important. Good assumptions are often critical to the solution. Simulation is the auxiliary method that works in many cases. Sometimes, physical analyses of the chip are required. Even so, I am sure the reconstruction technique here is powerful technique to characterize the ADC.

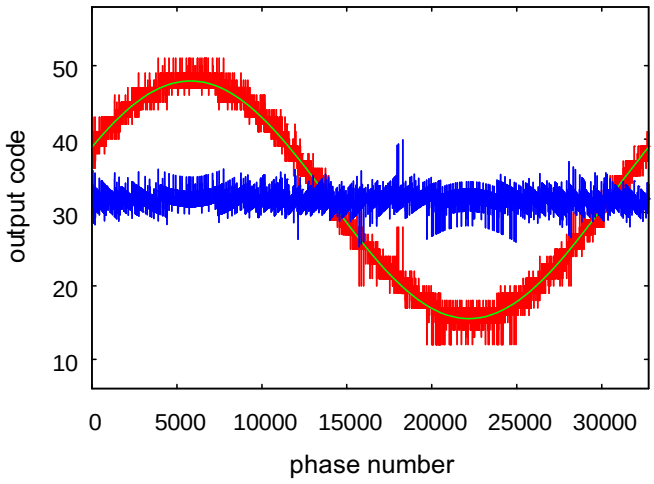

Figure 3.8: Difference wave affected by large additive noise: $f_{in} \approx 50$ MHz and $f_{clk} = 200$ MHz

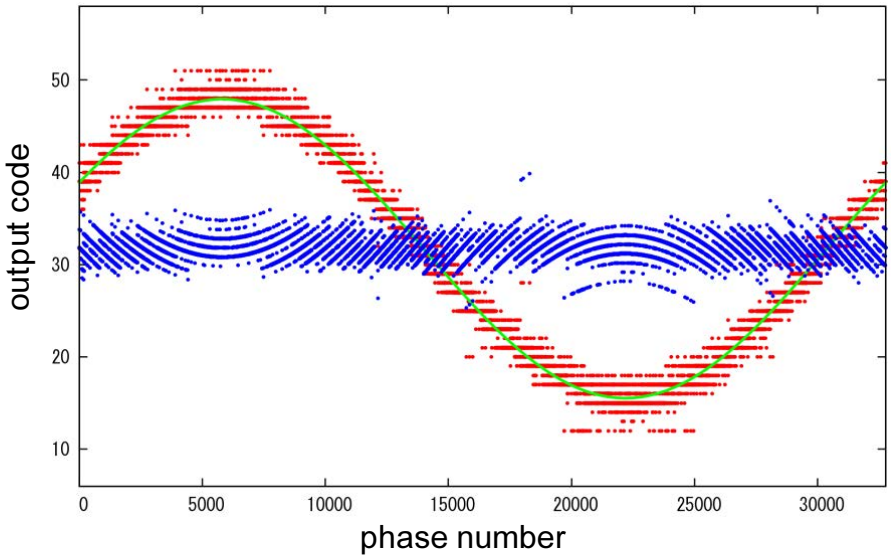

Figure 3.9: Difference wave affected by large additive noise (disconnected points)

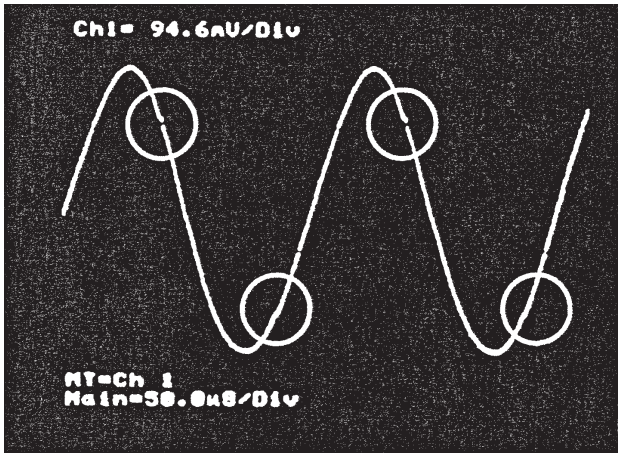

Figure 3.10: Gaps observed in a modified beat test: $f_{in} = 125\,\mathrm{MHz} + \alpha$, $f_{clk} = 500\,\mathrm{MHz}$

3.3 Dogtooth-like distortion

As a good application of the reconstruction technique, this section aims to identify the error phenomena that we call "dog-tooth." While investigating the cause of the phenomena I propose an additional analysis technique, the "previous wave plot." I also devise an extra simulation technique that reorders the outputs of a comparator with the inputs. The technique helps to design a comparator circuit without constructing a whole ADC.

3.3.1 Observation of the phenomena

Pursuing low power design of flash ADCs, I have gradually noticed there seems a general distortion pattern in reconstructed waveforms. More specifically, the error looks like a snag that appears at about the same delay after "extremum," *i.e.* a peak or a bottom. The phenomena are first observed a 10-bit flash ADC prototype as already shown in Fig. 3.6. The same phenomenon is also found in Matsuzawa [34, Figure 6.22]. Since the design is completely different, it should have common nature in the distortion. I cite the figure in Fig. 3.10. This is an oscilloscope screen shot measured in envelope frequency test. In the test, an 8-bit flash ADC (device-under-the-test) and a DAC are back-to-back connected as Fig. 1.11. The input is sine wave of $f_{in} = 125\mathrm{MHz} + \alpha$ and the ADC operates at $f_{clk} = 500\mathrm{MHz}$. Clock to the DAC are decimated by 4 so the DAC output forms a beat wave of frequency α.

To identify the phenomena, we have measured a 6-bit 500 MSps flash ADC. It is a bipolar flash ADC with a number of improvements over classic ADCs. The base architecture and circuits

are explained in Appendix A. The key advantage is its immunity against two major error mechanisms, *bubbles* and *metastability* [35]. Bubbles are anomaly of the internal thermometer code of flash ADCs. They may bring large encoding errors, or sparkles. Metastability is the occasional inability of a front-end latched-comparator to resolve a small differential input into a valid logic level. In this ADC, bubbles within 7 LSB do not bring a large sparkle error. Metastable errors are suppressed within one LSB. From the requirement by the target system, this ADC is designed to operate at double sampling mode, and the input bandwidth is designed to be 100 MHz, well below the Nyquist frequency. In order to reduce the power consumption, we have reduced currents of the front-end latched comparators accordingly. This limited bandwidth has been beneficial for dog-tooth measurements as we see eventually.

In this particular measurement, due to the speed limit of our logic analyzer, ADC outputs are decimated by four. In this case, the clock frequency number n_{clk} in the equation (3.2) has to be modified to

$$f_{clk} = 4\, n_{clk}\, f_0. \tag{3.34}$$

This modification keeps most of the reconstruction procedure, only to use the new n_{clk}.

Fig. 3.11 and Fig. 3.12 illustrate reconstructed waveforms. The target f_{clk} in both measurements is 500 MHz, while target f_{in} in Fig. 3.11 is 100 MHz and f_{in} in Fig. 3.12 is 150 MHz. The value $n_{clk} = 2^{16} = 65536$ is chosen for enough resolution and handy data size. The base frequency f_0 is around

$$f_0 = f_{clk}/n_{clk}/4 \approx 500\,\mathrm{MHz}/65536/4 \approx 1.90735\,\mathrm{kHz}$$

from the conditions (3.34). Considering minimum resolution of signal generators, f_0 is truncated as 1.9 kHz. So that $f_{clk} = 498.0736$ MHz. Since n_{in} only has to be an odd number as commented just after the formula (3.2), f_{in} step becomes very fine, only $2\, f_0 = 3.8$ kHz. Selected f_{in} values are $99.9989\,\mathrm{MHz} = 52631 \times 1.9\,\mathrm{kHz}$ in Fig. 3.11 and $1499.993\,\mathrm{MHz} = 78947 \times 1.9\,\mathrm{kHz}$ in Fig. 3.12.

The measured data in Fig. 3.11 show slight bumps, which are clearer in Fig. 3.12. They coincide with the phases where the input signal crosses the codes 8, 16, 24, etc., or borders of equally divided of the ADC full scale by eight. This property reflects the internal structure of the ADC, which is built up by eight blocks [36].

At the higher input frequency in Fig. 3.12, new bumps appear in the difference wave, which we call “dog-teeth.”

Fig. 3.13 shows the power spectrum of the reconstructed waveform. Resulting from reordering, the power spectrum is naturally sorted by harmonic order. Fundamental harmonic frequency corresponds to the input frequency and its power spectrum is normalized as 0 dB. The rest of harmonics are equivalent to the power spectrum of the difference wave due to the orthogonality of Fourier series.

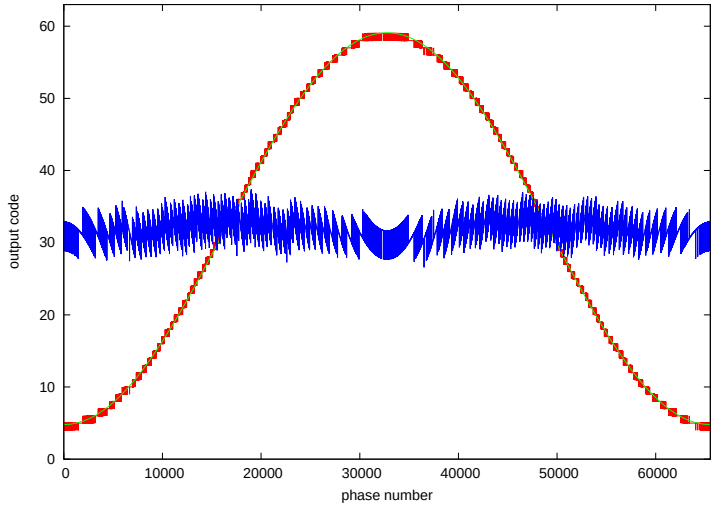

Figure 3.11: Reconstructed waveform: $f_{in} = 99.9989\,\text{MHz}$, $f_{clk} = 498.0736\,\text{MHz}$

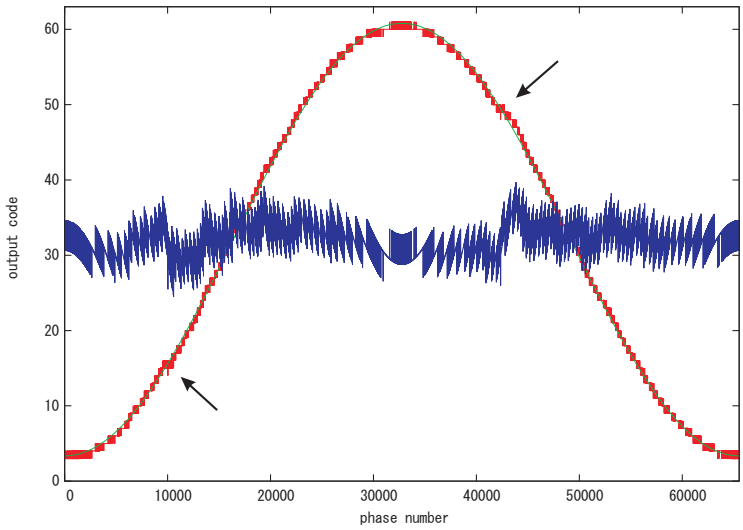

Figure 3.12: Reconstructed waveform: $f_{in} = 149.9993\,\text{MHz}$, $f_{clk} = 498.0736\,\text{MHz}$

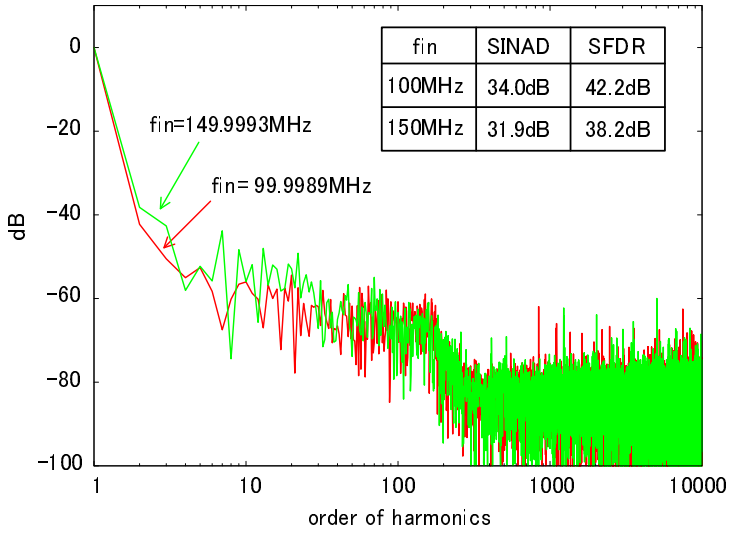

Figure 3.13: FFT results of data in Fig. 3.11 and Fig. 3.12, $f_{clk} = 498.0736\,\mathrm{MHz}$

Compared to the difference wave in time domain, power spectrum plot lost various features of the distortion.[1] For one thing, the SINAD of the data in Fig. 3.12 is 31.9 dB, not so degraded from 34.0 dB in Fig. 3.11 but dog-teeth appear at this f_{in}. For the second, a slight warping of the difference wave in Fig. 3.11 is hard to recognize in frequency domain.

Knowing the result in Fig. 3.12 solely, dog-teeth seemed to occur just after extrema of the input signal. So a mechanism was suspected that comparators at around extrema do not swing fully, resulting the sampling delay variation. The variance must be recovered after each extremum. There might be a discontinuity at that recovery. If this assumption is correct, dog-teeth cannot be far apart from extrema.

In order to determine the places the dog-teeth appear, we measured at different input frequencies. Fig. 3.14 shows results. In the figure, initial phases of the reconstructed waveforms have been intentionally shifted for visibility. Obviously dog-teeth move later steadily as f_{in} gets higher. They even move across 90 degrees at $f_{in} = 300\,\mathrm{MHz}$, where no longer *just* after the extremum. This observation clearly contradicts the above assumption.

One noticeable feature in Fig. 3.14 is that dog-teeth after peak and after bottom occur at the same delay from the extrema. Since comparator circuits in different reference levels usually do not operate so symmetrically, especially for single (not complimentary) circuit this ADC adapts,

[1] In fact, some information in Fourier transform is really lost since the power spectrum plot ignores the phase information.

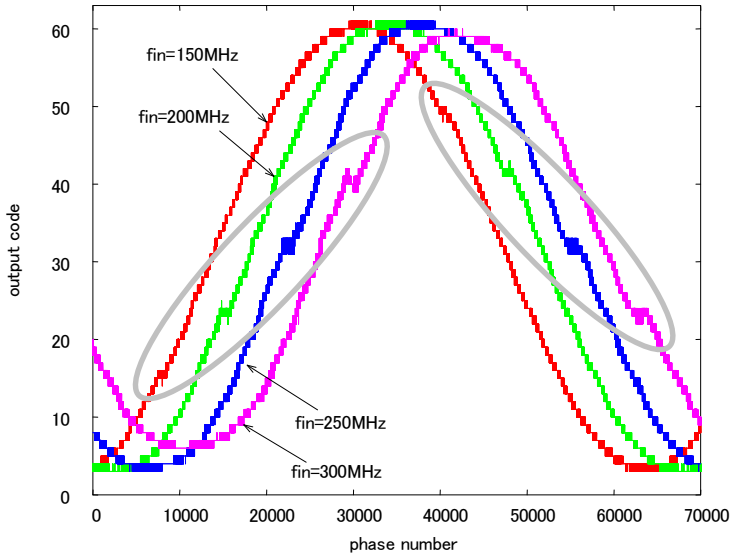

Figure 3.14: Input f_{in} dependency of dog-teeth at $f_{clk} = 498.0736\,\mathrm{MHz}$

this observation suggests the mechanism is more global than local functions of comparators.

Fig. 3.15 shows the f_{clk} dependency at the same $f_{in} \approx 200\,\mathrm{MHz}$. Phases of the reconstructed waveforms have been intentionally shifted so as dog-teeth take the same abscissa. We see the dog-tooth place shifts earlier as the clock frequency becomes higher. Since the input frequency is unchanged in those measurements, this shift should not be attributed to individual comparators in the flash ADC. This is another clue that contradicts the above assumption.

Fig. 3.16 shows the reconstructed waveforms of various amplitudes. The results indicate the dog-tooth place is independent of the input slew rate.

To investigate the f_{in} dependency more minutely, we drew the dog-tooth place vs. f_{in}/f_{clk} plot in Fig. 3.17. The clock frequency f_{clk} is 498.0736 MHz in all data. The dog-tooth places are measured after the extremum and in unit of the clock period. Explicitly there exists a linear relation in Fig. 3.17. Its slope is $1/2$. Actually, phases in Fig. 3.15 are shifted according to this speculation, where the parameter is f_{clk} instead of f_{in}.

We summarize above observations:

- Dog-tooth error appears at a definite phase that is determined only by the ratio of f_{in}/f_{clk}. More specifically, the place is $\pi f_{in}/f_{clk}$ radian after each extremum of the input.
- Dog-tooth place does not depend separately on f_{in} or f_{clk}. Only the ratio affects.
- Dog-tooth place does not depend on input magnitude nor slew rate.

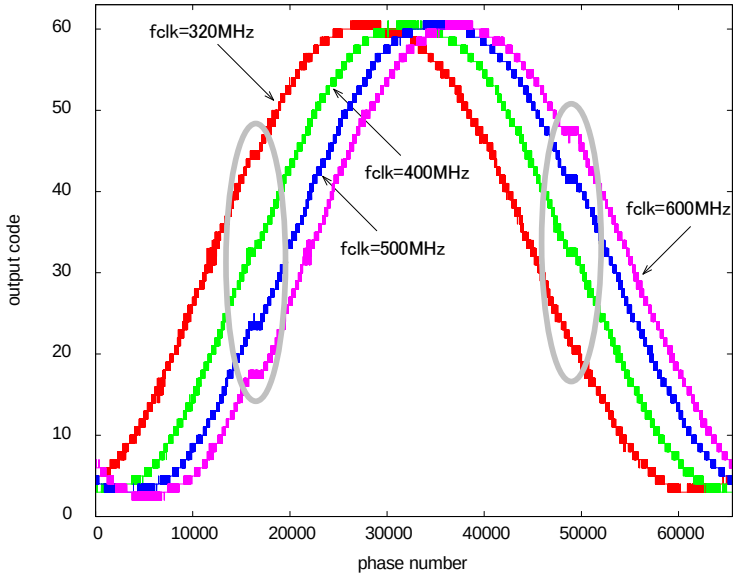

Figure 3.15: Input f_{clk} dependency of dog-teeth at $f_{in} \approx 200\,\text{MHz}$. Precisely ($f_{in}$, f_{clk}) = (200.0007 MHz, 319.81568 MHz), (200.00008 MHz, 398.45888 MHz), (199.9997 MHz, 498.07360 MHz), (199.99932 MHz, 597.68832 MHz).

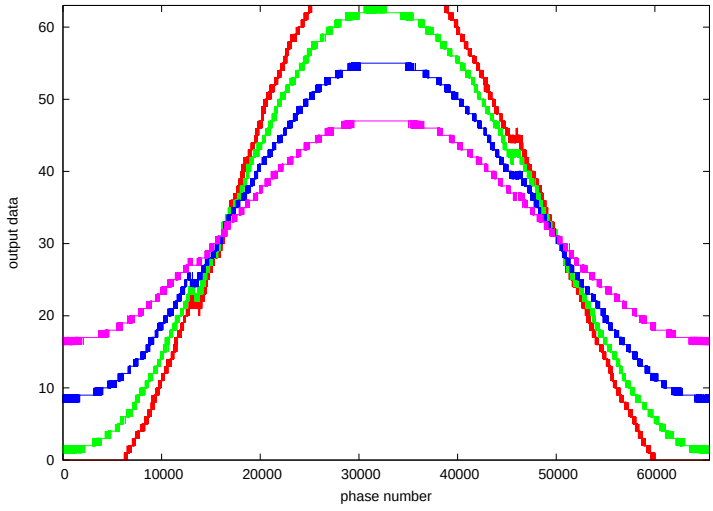

Figure 3.16: Input amplitude dependency of dog-teeth: $f_{in} = 199.9993\,\text{MHz}$, $f_{clk} = 498.0736\,\text{MHz}$

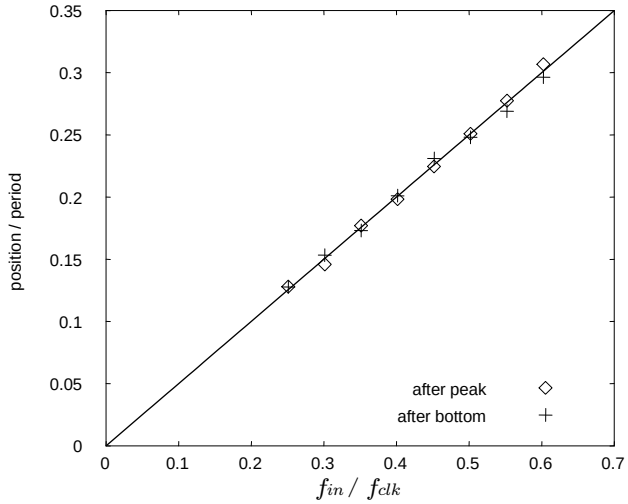

Figure 3.17: Dog-tooth place dependency on f_{in}/f_{clk}, where $f_{clk} = 498.0736\,\mathrm{MHz}$

Various documents provide further data if knowing the phenomena. The one-quarter beat wave in Fig. 3.10 shows dog-teeth where they are expected at 45 degrees after extrema. The reference ADC in Appendix A also shows dog-teeth in Fig. A.12. Though it has similar design in this section, it is implemented in different process. We can find more data in the envelope test condition, where $f_{in} \approx f_{clk}/2$. Then dog-teeth will appear at the midst (zero-crossing) of the reconstructed sine wave. We observe them in [37, Fig. 14], [38, Fig. 6] and [39]. Especially, Figure 6 in [39] is quite clear with no other deficits in envelope waves. Though dog-teeth do not always appear in measurements, we expect that they happen generally and occasionally in many flash ADCs.

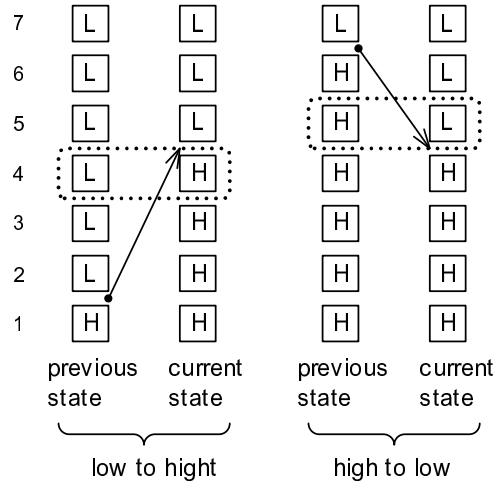

Figure 3.18: Two types of the state change of latches that determine the border of a thermometer code

3.3.2 An assumption of the error mechanism and investigations

Observations in the previous section suggest that the dog-tooth should stem from very nature of flash ADCs, rather than characteristics of individual circuits or implementations.

To investigate the mechanism, we now go back to its principle. A flash ADC consists of a row of comparator-and-latches, *c.f.* a textbook [18]. One of two inputs of each comparator is commonly connected to an analog input, while another input is connected to each reference level individually. Sampling clock holds the state of comparators to latches. The latches form a thermometer code whose border reflects the input level and then determines the ADC output.

Two types of state changes occur at the border. One is a change from high to low and the other is from low to high. Fig. 3.18 shows both examples. Boxes in the figure show individual latch states. It will be natural to assume that two changes have different characteristics.

To know which type of state changes occur at each sampling phase, we need to know the one-clock previous state of latches. The problem is how the previous state is depicted in the reconstruction waveform plot. Since the reconstruction operation changed the sampling order, some inference is required.

However, the reordering operation does not destroy the previous state completely. Let consider a sequence of sampled data. If we put one previous data before the current sequence and remove the last one we get a new sequence of sampled data with the same length. This operation is shown Fig. 3.19. The obtained sequence is the previous state sequence. It advances the cur-

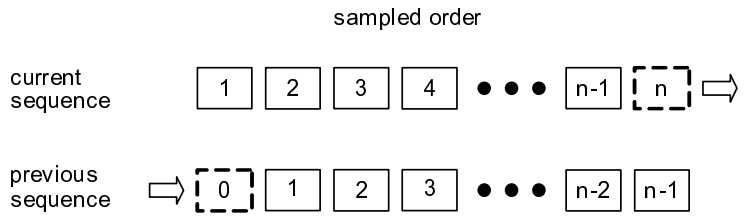

Figure 3.19: Current and previous state sequences

rent sequence by one clock. We can reconstruct the previous wave from this new data set. The obtained waveform is very similar with the original reconstructed waveform. Only the phase is advanced by one clock.

One clock period corresponds to $(f_{in}/f_{clk})n_{clk}$ samples. If there is no clock decimation in the measurement, this value is equal to n_{in} from (3.1)(3.2). Our idea is, instead of reordering, simply to overlay the one-clock previous fundamental tone on the reconstructed waveform plot. Fig. 3.20 is an example. The figure shows a reconstructed waveform of $f_{in} = 199.99932\,\mathrm{MHz}$ and $f_{clk} = 597.68832\,\mathrm{MHz}$. The jagged wave is the reconstructed wave and the smooth one is the derived previous wave.

In a flash ADC, both waves show the border of the thermometer code at that time. Comparator states above the wave are low and below the wave are high. So the reconstructed wave below the previous wave results from high-to-low state change, and the waveform above the previous wave results from low-to-high stage change. At the intersections of two waveforms, different types of state changes cross over. The previous wave proceeds by $2\pi f_{in}/f_{clk}$ radian than the reconstructed waveform and the intersections reside at the midst of corresponding extrema. They are just where we observed dog-teeth.

In Fig. 3.20, the upper part of the reconstructed waveform divided at dog-teeth is pulled down and the lower part is pulled up. If this is the case as it looks, a straightforward assumption to explain the phenomena is hysteresis of latched comparators. Fig. 3.21 illustrates this new assumption.

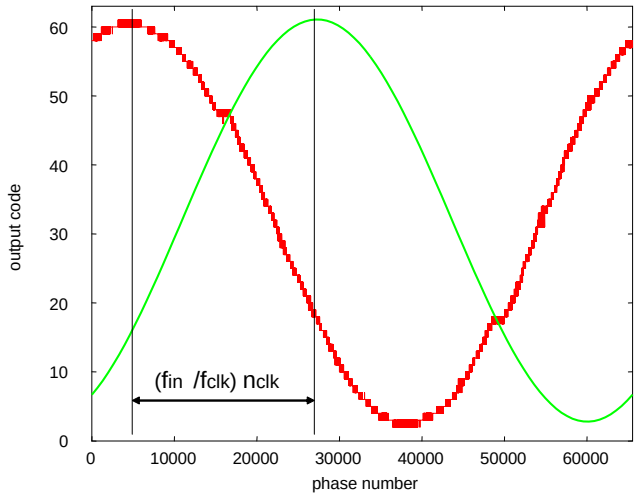

Figure 3.20: Reconstructed waveform of f_{in} = 199.99932 MHz with one-clock previous sine wave, where f_{clk} = 597.68832 MHz

In order to verify the assumption, authors wrote a simple Verilog-A script that implements this block diagram as a 6-bit quantizer. The main body of the script is as follows:
The value of `clock` is 0 or 1. The rising edge of `clock` is detected by `cross` function. The hysteresis is posed on `vin` instead of `vref` (not used explicitly) for easier implementation. The variable `lsb` holds one LSB value and `vrb` holds the bottom reference value.

Fig. 3.22 is an overlaid plot of the two reconstructed waveforms. Dotted one is the measurement data. Connected one is simulated results by the Verilog-A script. The dog-tooth places coincide very well. Since our model can determine the place but it cannot determine the magnitude of the hysteresis, the magnitude is adjusted to the measured data.

We think bubble codes tend to occur around the dog-teeth. There would be sparkles there, if the ADC has less capability to suppress the bubble code errors.

Table 3.3: Verilog script of 6-bit ADC with hysteresis

```
analog begin
    @( cross(V(clock) - 0.5, 1) ) begin
        vprev = vcurr;      // save the quantized value previously
        vcurr = V(vin);

        if (vcurr > vprev) begin      // hysteresis
            vcurr = vcurr - vhys;
        end else begin
            vcurr = vcurr + vhys;
        end

        code = (vprev - vrb)/lsb;   // quantization
        if (code < 0) begin           // clipping
            code = 0;
        end else if (code > 63) begin
            code = 63;
        end
        $fstrobe(fout, "%2d", code);
    end

    V(dout) <+ transition(code, 0, 1n, 1n);

end
```

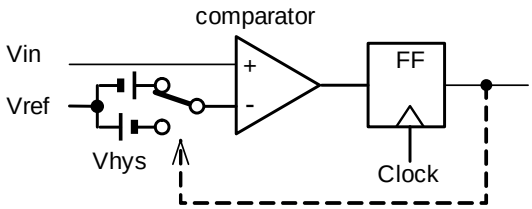

Figure 3.21: Hysteresis model of one latched-comparator of a flash ADC

3.3.3 Simulation of dog-tooth distortion using a sole latch

The origin of hysteresis is another problem. Plural mechanisms will contribute to the effect. We can simulate the mechanism caused by comparator itself fairly easily. With assist by Euclidean ordering technique, we can reconstruct dog-tooth distortions using only one latch simulation. The simulation provides the size estimation of dog-tooth.

We take a CMOS latch [28] in Fig. 2.17 that is also used for linearity analysis.

An idea we present here is to simulate the latched comparator with stepping the reference voltage, instead of constructing a whole ADC. In this simulation, the result is a bit stream of comparator outputs. We can append the reference level information to it by saving the reference value as 0 and the slightly higher level as 1. The Euclidean reordering works fine with the tuples as well as single values, so we can treat the analog input and comparator outputs as a set. A good point of this idea over conventional treatments of ADC, we can juxtapose the comparator outputs and the analog input waveform. Then we can visualize the comparison delay and the magnitude change. Fig. 3.23 shows a result.

We can generate the ADC output-like waveform from Fig. 3.23 by taking the maximum reference level that the comparator output is high. By slight shift, 0.5 LSB equivalently, the result can directly be comparable with the input. Fig. 3.24 shows the reconstructed waveform plot with the input wave. The difference wave is formed for the fundamental tone as usual, not for the input wave. We can find dog-teeth of 4 mV where the theory expects. Over all shape of the difference wave is quite similar with the measured data in Fig. 3.12. We think this simple circuit simulation can explain major part of the dog-tooth mechanism.

Generally speaking, the hysteresis mechanism is common to all latched comparators so that any types of ADCs may suffer from the same cause of distortion. A flash ADC shows the simplest form of the error that we call dog-tooth.

In two-step ADCs, the previous wave plot will be as helpful as in flash ADCs for the performance analysis, since the one clock previous state that the previous wave plot clarifies affects the conversion result in direct manner. In other types of ADCs, such as pipeline, successive

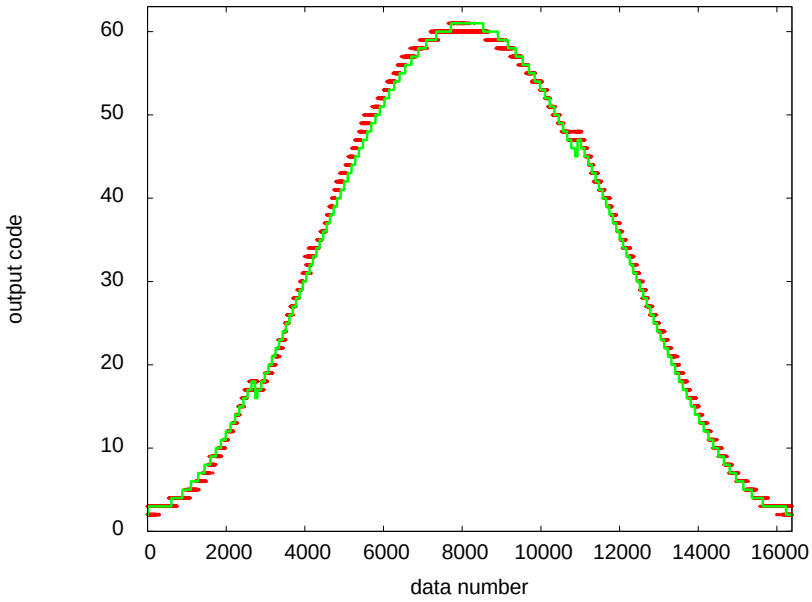

Figure 3.22: Overlaid plot of reconstructed waveforms: the hysteresis model results (connected) and measured data (dotted) at $f_{in} = 199.99932\,\mathrm{MHz}$, $f_{clk} = 597.68832\,\mathrm{MHz}$

approximation, or ΣΔ ADCs, this error mechanism appears differently. It will not form a dog-tooth any more. But still, authors expect that methods we have presented, which are developed for the dog-tooth analysis, will play an important role to analyze the dynamic performance of comparators.

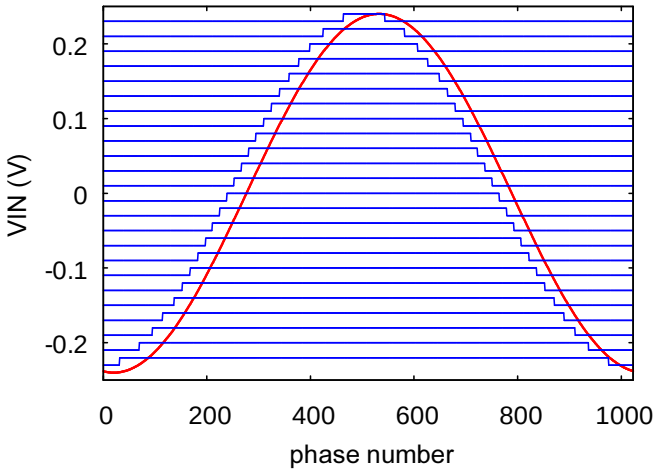

Figure 3.23: Reconstructed input waveform and comparator outputs with changing reference voltages: $f_{in} = 355.725\,\text{MHz}$ ($n_{in} = 255$) and $f_{clk} = 1428.48\,\text{MHz}$ ($n_{clk} = 1024$)

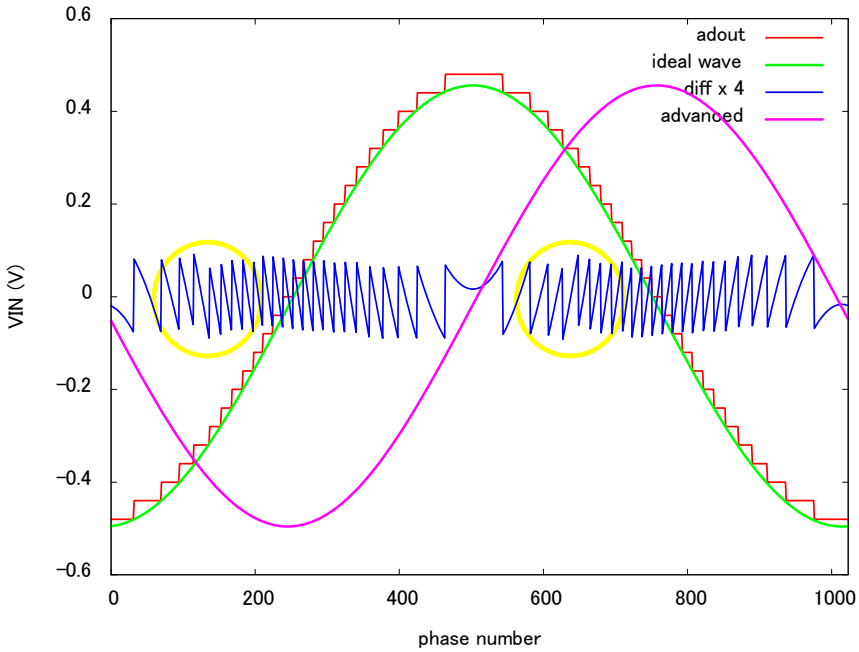

Figure 3.24: Reconstructed waveform of the simulated data in Fig. 3.23

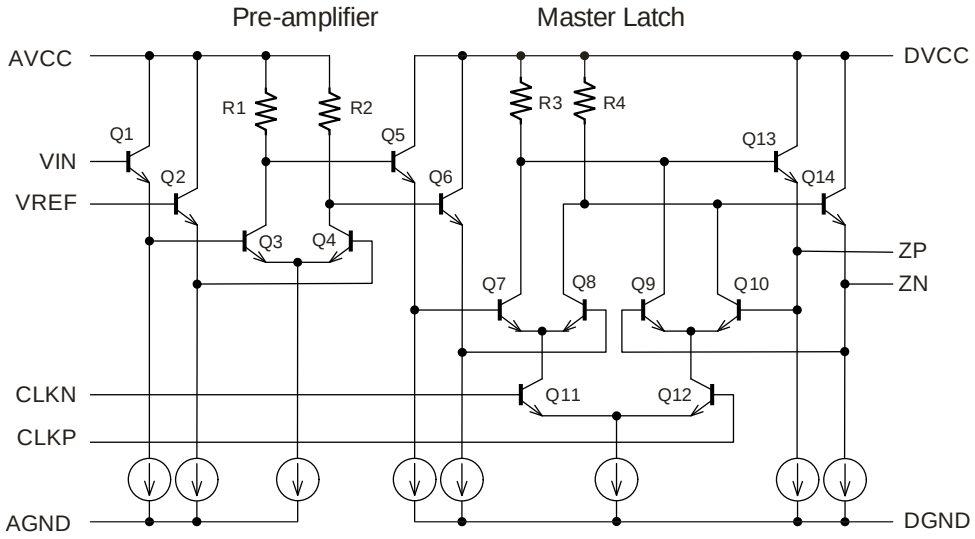

Figure 3.25: Conventional bipolar latch

3.4 An application of the difference wave method to a bipolar latch design

In this section, we demonstrate a capability of the difference wave method to a circuit design. The reference flash ADC in Appendix A adopted a new architecture for the latch [40]. The circuit contributes to make the ADC power lower but it is against the common knowledge at the time. If the difference wave method were not available, it had been too risky to adopt it.

Fig. 3.25 shows a conventional bipolar latched-comparator circuit [41]. The cell repeats $2^n - 1$ times in an n-bit flash ADC. All VINs are connected to one analog input terminal and each VREF is connected to the individual reference voltage as shown in Fig. 1.5. The circuit consists of a pre-amplifier and a master latch. The pre-amplifier shifts the signal levels to those defined by VCC, which keeps the operation range for the master latch. Since the pre-amplifier has wider common mode range than the master latch, this configuration is useful to expand the analog input range. Another expectation to the pre-amplifier is to reduce the kick-back noise. Aiming at further noise reduction, the power supplies as well as grounds for the pre-amplifier and the master latch are separated in order to diminish the interferences by common impedances.

Emitter followers are placed at every gain stages to diminish the inter-stage coupling. The input source follower Q1 and Q2 reduce the effect of the pre-amplifier state and the kick back. Since the input capacitance of Q3 changes whether it is active or inactive, the input capacitance

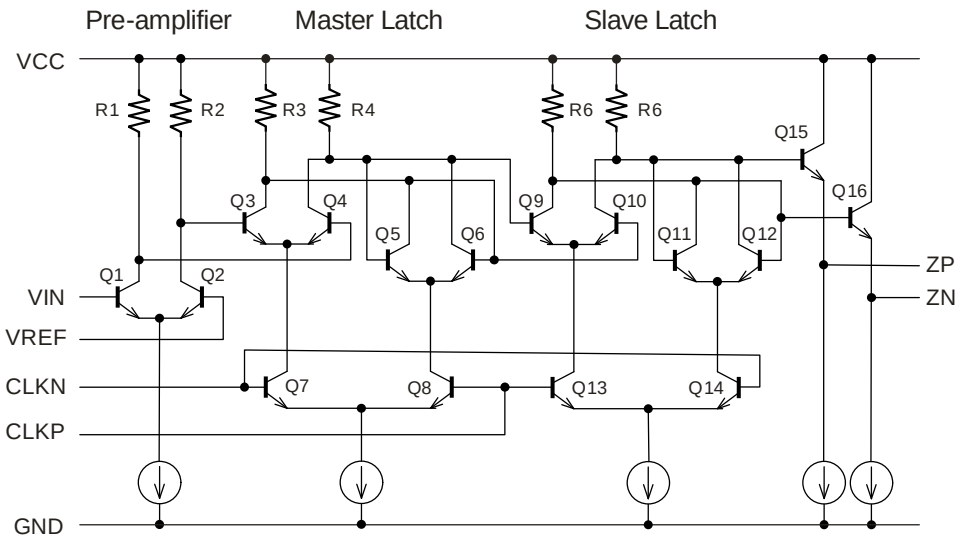

Figure 3.26: Low power latch by eliminating source followers

of the ADC changes depending on the VIN level without the emitter follower. And the VREF voltage is modulated by the base current of Q4, INL will also be affected. These become additional distortion causes. The emitter follower Q5 and Q6 reduces the kick-back from the master latch. The emitter follower Q13 and Q14 enhance the driving capability of the master latch. They are also expected to enhance the latch performance driving Q9 and Q10 with lower impedance.

A simple idea to reduce the power is to remove source followers in the circuit. Fig. 3.26 shows the schematic. The slave latch is appended. The distortion by the elimination of the input source followers is evaluated and compensated to some extent. A good point is this elimination enhances the common mode range by one V_{BE} which realizes the wider input range of the ADC. The output emitter follower Q15 and Q16 remain caring the kick back from the load. The latch loop of Q11 and Q12 is formed without the emitter followers. From my study, the performance difference is subtle but slightly better without emitter followers in the loop. Further, I think the latch loop should be isolated from the output loads as far as possible.

The source follower elimination between the pre-amplifier and the master latch posed a problem, which is *dog-tooth*. Fig. 3.27 and Fig. 3.28 show reconstructed waveforms from simulation results of the ADC top schematic. Parasitic capacitors are not extracted but 10 fF is added uniformly at every node. When the input frequency is 133 MHz, dog-teeth are about 1 LSB. At 201 MHz, they look over 2 LSB. Since these dog-teeth almost disappeared when the source followers are inserted, the cause are identified the kick back from the master latch.

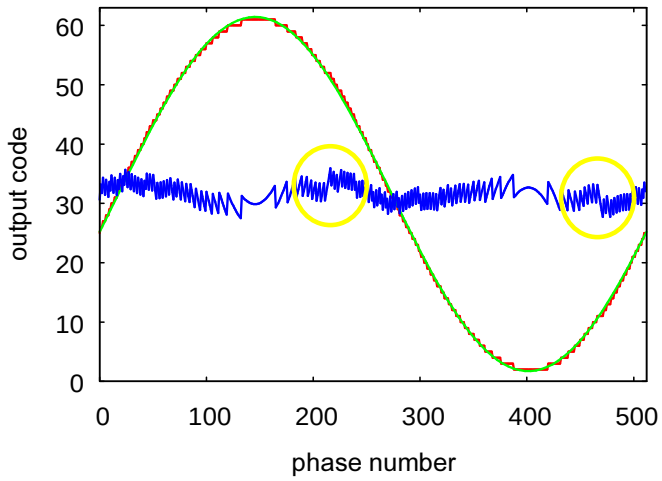

Figure 3.27: Schematic simulation results of the low power latch: f_{in} = 133 MHz and f_{clk} = 512 MHz, SINAD=34.17dB

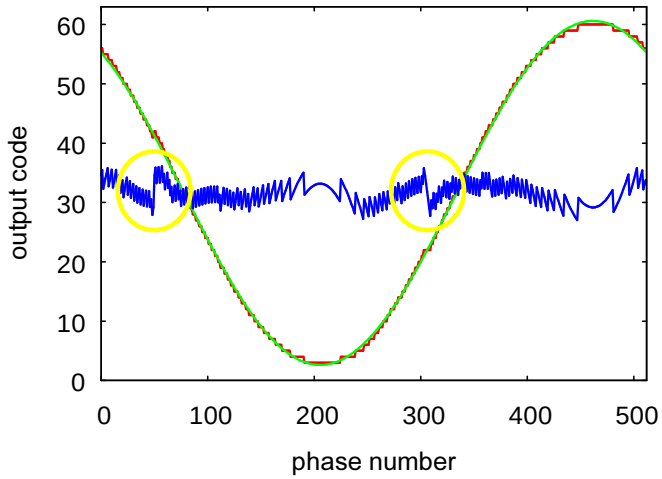

Figure 3.28: Schematic simulation results of the low power latch: f_{in} = 201 MHz and f_{clk} = 512 MHz, SINAD=33.16dB

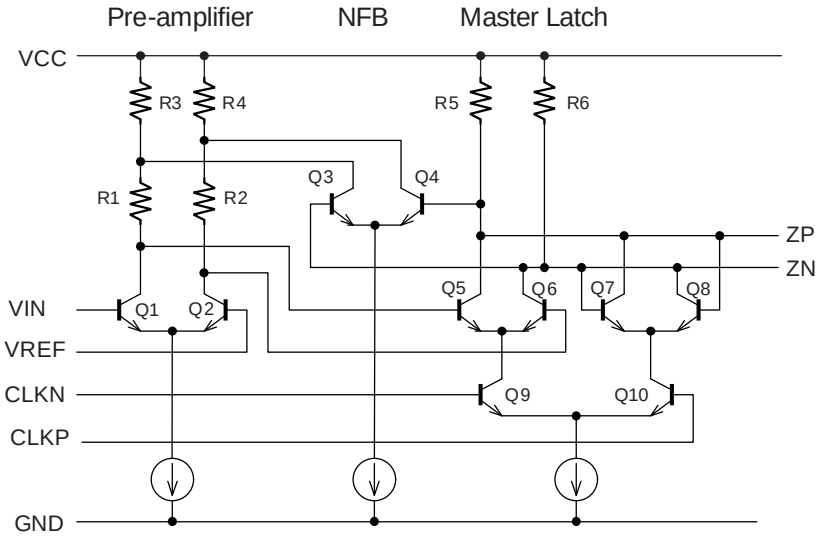

Figure 3.29: Low power latch with the negative feedback

Here comes an idea to diminish the kick back, “negative feedback from the latch.” Because the origin of the dog-tooth must be the base currents through Q3 and Q4, which are necessary to invert the latch state, the negative feedback may compensate the effect. Fig. 3.29 shows the idea. The slave latch is omitted in the figure but necessary. This configuration couples the pre-amplifier and the master latch, which is against the previous knowledge, but worked fine.

In order to reduce the current for the NFB amplifier, I modified the circuit further in Fig. 3.30. In stead of genuine negative feedback, the common resister R3 and R4 are used. The presumption of the configuration is the transition of the master latch is much wilder than the pre-amplifier so that the coupling will work as the feedback rather than the bilateral cross-coupling. This presumption is proved by simulations in Fig. 3.31 and Fig. 3.32. In addition to the dog-tooth reduction, warping in the difference wave is also diminished. We can control the warping and dog-tooth adjusting several parameters like R1 to R6, the tail current of the pre-amplifier and that of the master latch.

This latch circuit shown in Fig. 3.30 is low power. It has only two tail currents. This is one-quarter of the conventional latch in Fig. 3.25. This reduction contributes to reduce the total power of the ADC.

Parasitic capacitors tend to worse the distortion . Fig. 3.33 and Fig. 3.35 show the simulation results after layout. They correspond to Fig. 3.31 and Fig.3.32 respectively. Measured data at the same conditions are placed in Fig. 3.34 and Fig. 3.36. Measured SINADs are slightly worse than

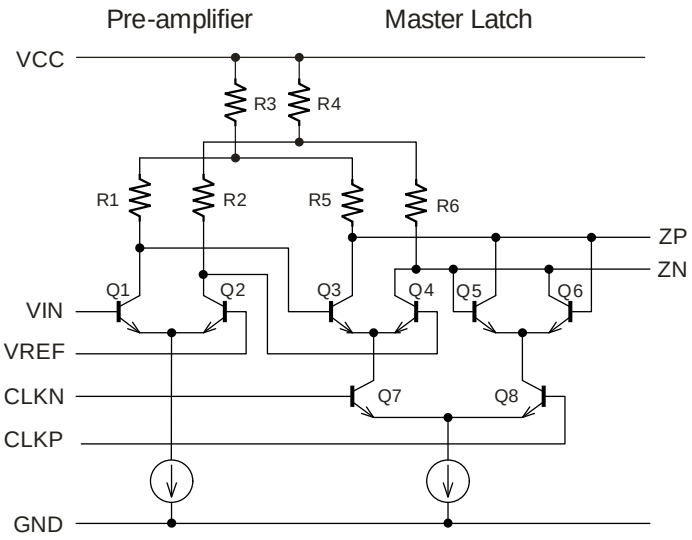

Figure 3.30: Low power latch with the common resisters (The slave latch is not shown but exists.)

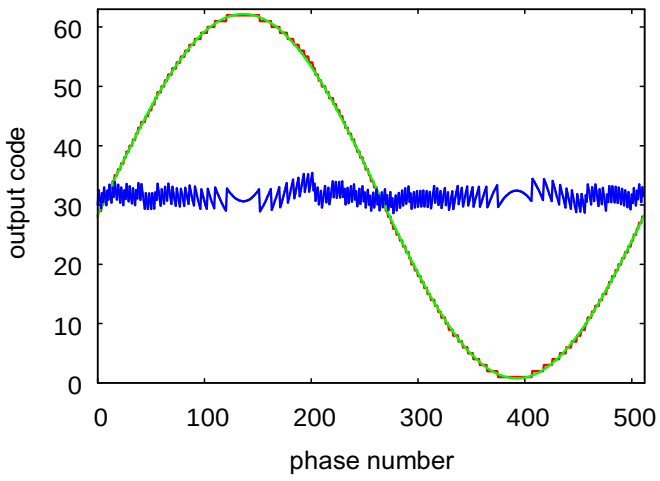

Figure 3.31: Schematic simulation results of the common resister latch: f_{in} = 133 MHz and f_{clk} = 512 MHz, SINAD=36.62 dB

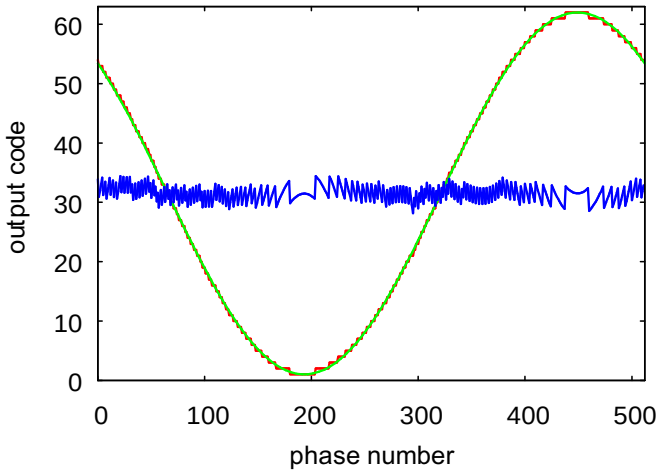

Figure 3.32: Schematic simulation results of the common resister latch: f_{in} = 201 MHz and f_{clk} = 512 MHz, SINAD=36.49 dB

the simulation but the tendency looks similar. The difference seems from micro structures of the ADC rather than the warping or dog-teeth. We could have lessened the dog-teeth if we adopted more feedback gain. Since the ADC was the first of this radical latch, I have chosen the moderate coupling than the minimum distortion by the simulation.

3.5 Summary

This chapter described the distortion analysis at time domain. Summaries are:

1. An unabridged theory of the waveform reconstruction technique is presented. The theory is based upon Euclidean algorithm. The theory provides not only the most flexible frequency conditions for reordering but also an efficient practical method.

2. The difference wave analysis method for a sine wave is provided. I dare say this is the electrocardiogram of ADC.

3. A specific distortion pattern of flash ADCs is identified using the difference wave analysis. We call it “dog-tooth.” The distortion power is so small that the frequency domain analysis has lost to find it.

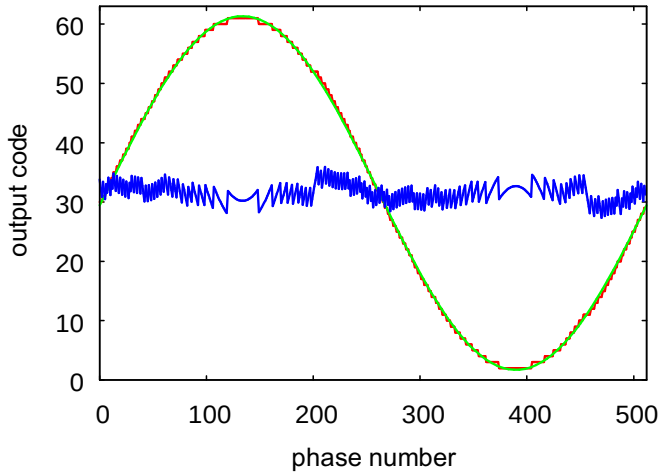

Figure 3.33: Simulation results of the common resister latch with parasitic capacitors: f_{in} = 133 MHz and f_{clk} = 512 MHz, SINAD=34.5 dB

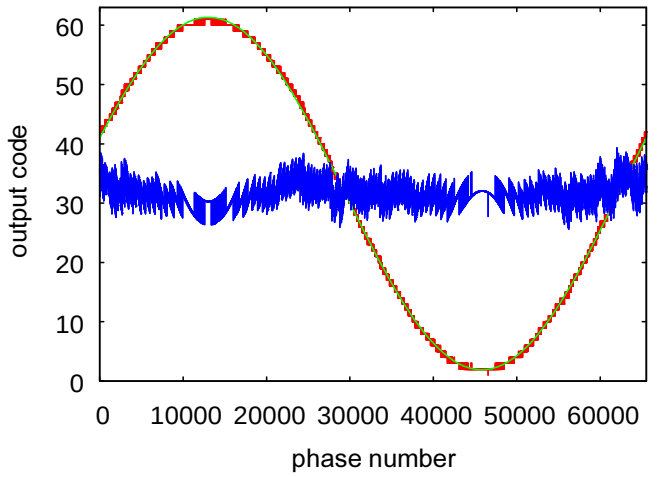

Figure 3.34: Measurements results of the common resister latch: f_{in} = 132.99975 MHz and f_{clk} = 511.18080 MHz, SINAD=32.9 dB

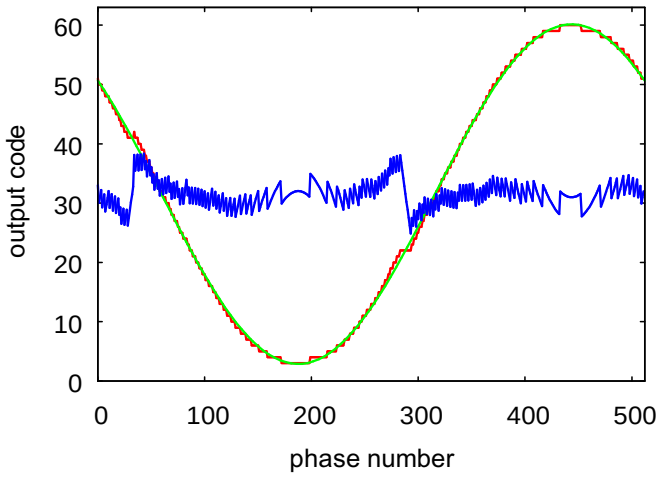

Figure 3.35: Simulation results of the common resister latch with parasitic capacitors: f_{in} = 201 MHz and f_{clk} = 512 MHz, SINAD=31.5 dB

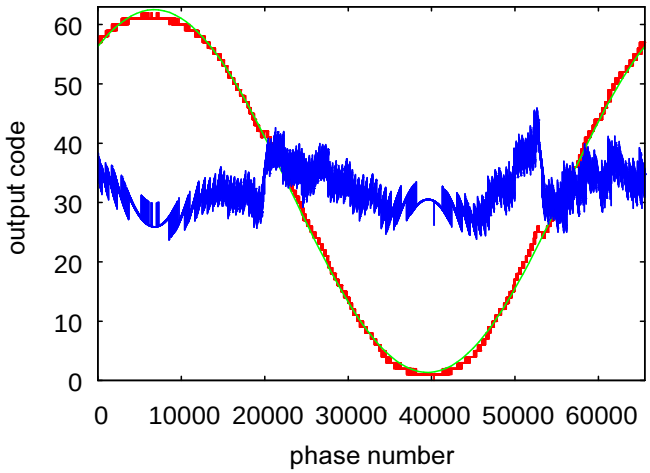

Figure 3.36: Measurements results of the common resister latch: f_{in} = 201.00015 MHz and f_{clk} = 511.18080 MHz, SINAD=27.7 dB

Table 3.4: Tips on the difference wave analysis

- The thickness of the difference wave is about one LSB.
- One period of the reconstructed waveform is one period of input. The division by samples provides the time resolution.
- Additive input noise makes the difference wave thicker uniformly. Timing jitter makes the difference wave thicker proportionally to the input slew rate.
- Warping in the difference wave of the same frequency with the reconstructed waveform indicates the amplitude estimation error caused by clipping-like (3rd order) distortion.
- A one-clock previous wave plot provides the information of the internal state change of the ADC.

4. To analyze the cause of the dog-tooth, we devised the “previous wave plot” that is accompanied with the reconstructed waveform. The plot designates the dog-tooth places. The positions suggest its cause is the hysteresis of latched-comparators.
5. We also devised a simulation method of latch hysteresis. It is a new tool of the latch circuit design.
6. We introduced a new bipolar latch circuit. The topology is featured by its low power. The difference wave analysis played an essential role since the topology is against the common knowledge at that time.

I have summarized useful tips on the difference wave analysis in Table 3.4.

Chapter 4

Composite signal analysis

The input signal of the difference wave analysis developed in Chapter 3 was limited to a sine wave. In this case, the ideal wave is estimated easily by FFT or three parameter estimation. Two methods provide the identical estimation since both use the same least mean square (LMS) error criterion. A pure sine wave has another merit that it is easy to generate. On the other hand, a sine wave input has intrinsic drawbacks as I have noticed in Chapter 1. In this chapter, we further examine them and cultivate the possibilities of composite waves as a test signal. We see the LMS criterion is also applicable for various signals as well as for sine waves.

4.1 A generalization of the distortion definition

Signals are transferred and stored in various form. Generally speaking, the absolute value at a time means nothing. Instead, its shape contains the information. Let us formalize this situation. Signals $u(t)$ and $w(t)$ are equivalent when the following relation holds.

$$w(t) = a\,u(t - t_0) + b\,. \tag{4.1}$$

There are three parameters a, b and t_0 of freedom. The magnification, the level shift and the time shift of the waveform do not change the information that the signal contains. We call these operations M-transform, L-transform and T-transform respectively.

Actual signal processors, *e.g.* [42], have these three functions. Fig. 4.1 shows the conceptual diagram of the front end. The variable gain amplifier (VGA) is a part of automatic gain control that takes on the M-transform. The offset control loop realizes the L-transform. The T-transform is implemented indirectly by a phase locked loop (PLL) that consists of a phase detector, a loop filter, a voltage controlled oscillator (VCO), and an ADC. consequently it will be appropriate to define the distortion of measured data as the residue after three transforms of admissible inputs.

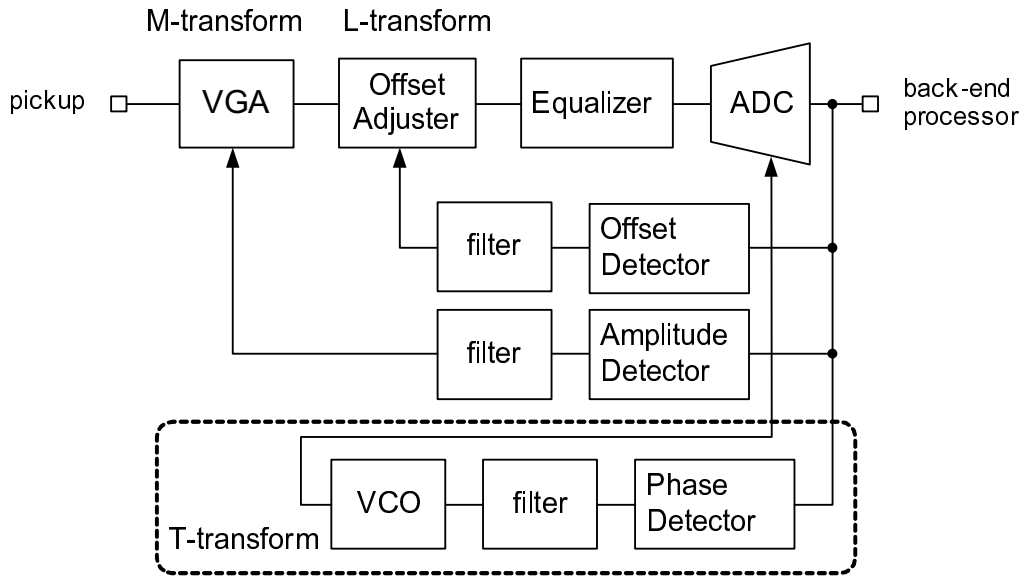

Figure 4.1: Conceptual diagram of an analog front end

The determination of the true input among admissible inputs is a hard work in real systems. As a matter of fact, the signal processing system exists for it. In measurement, we can use a known waveform according to our purposes. Then we can adjust three-parameters to obtain the ideal signal.

This situation is shown in Fig. 4.2. The distortion $e(t)$ of the obtained signal $w(t)$ is defined using the ideal waveform $\hat{w}(t)$ among the equivalent signals of the input signal.

$$e(t) = w(t) - \hat{w}(t)\,. \tag{4.2}$$

The problem here is to find the best estimation $\hat{w}(t)$ from $w(t)$. To be quite general, we can formalize the problem in Hilbert space. In this scenario, we first define an appropriate norm of waves $\|f(t)\|$ and then define a subspace W that the equation (1.1) generates and find a function $\hat{w}(t) \in W$ that satisfy

$$\min \|w(t) - \hat{w}(t)\| \tag{4.3}$$

or using orthogonal projection P onto the subspace W, the estimation $\hat{w}(t)$ is defined as

$$\hat{w}(t) = P(w(t))\,. \tag{4.4}$$

Let us take the signal power as the norm here. Then the equation (4.3) turns to be the LMS criterion that is consistent with the conventional sine wave measurements.

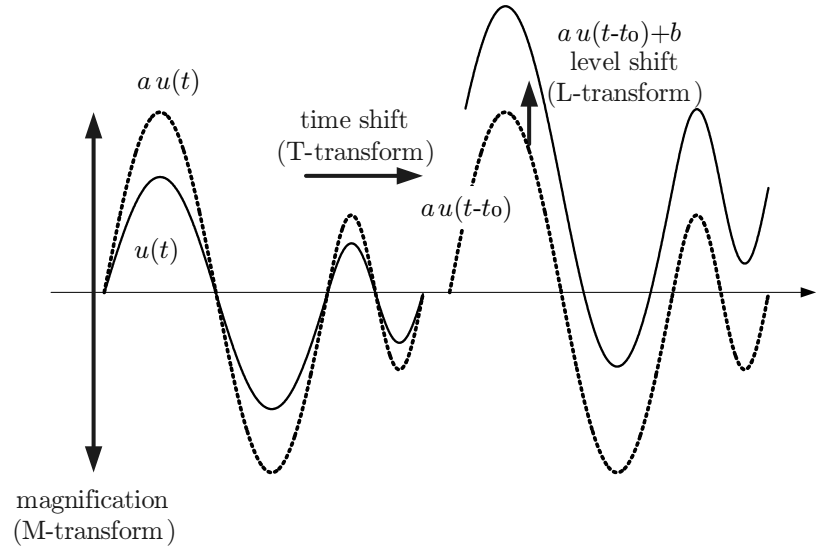

Figure 4.2: Equivalence of a signal

The very important point is three parameters a, t_0 and b in (4.5) must be constant during the conversion. It must not vary depending on the waveform. Pure sine waves cannot tell the dependency. Fig. 4.3 depicts the situation. The input signal magnitude is adjusted at each input frequency f_{in}. The appropriate cutoff LPF is selected according to f_{in}. It changes the signal amplitude, too. Finally, the ADC may have different offset depending on f_{in}. This dependency is completely ignored in definitions of SINAD or SFDR as we have discussed in Section1.4.

In order to examine these anxieties, let use a simple model of the practical ADC in Fig. 4.4. The frequency response of the ADC is implemented by the first order low-pass filter of the cut-off frequency 50 MHz. The INL is posed as 0.5 LSB parabola. Other non-idealities in real ADCs such as higher order distortion, DNL, noise and hysteresis are omitted.

Simulation results are shown in Fig. 4.5. The clock frequency f_{clk} is 102.4MHz and 1024-point data are used for FFT. At each input frequency f_{in}, the input magnitudes are adjusted manually to fit the ADC full scale. As afraid, the SINAD is quite flat for f_{in} in spite of the frequency response implemented in the ADC model.

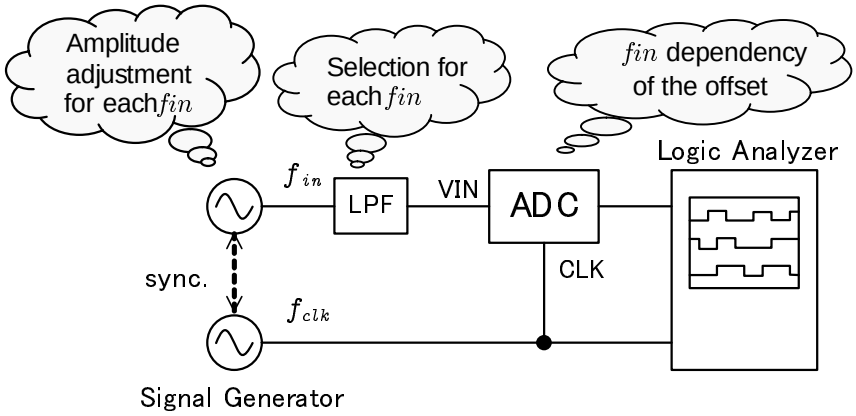

Figure 4.3: Operations in distortion measurement using sine wave

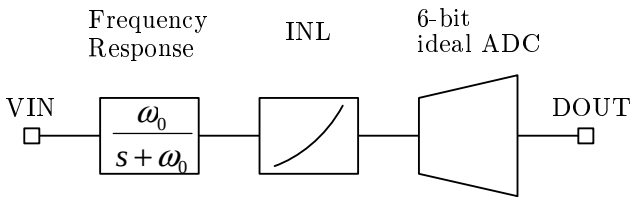

Figure 4.4: A simple model of an ADC

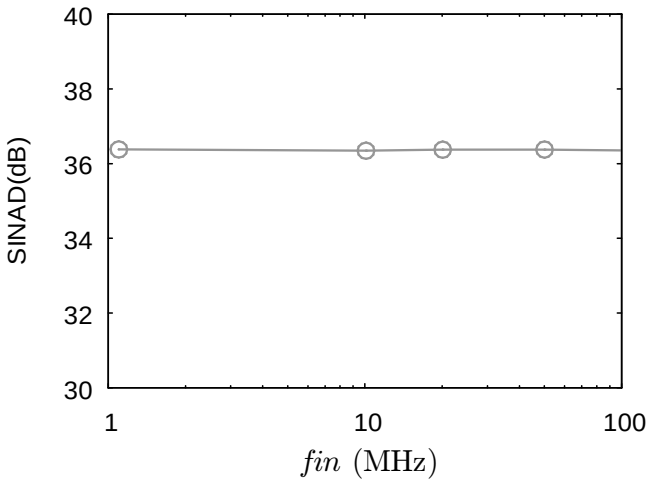

Figure 4.5: Simulated SINAD of sine input by adjusting the amplitude at each f_{in}

4.2 Estimation of the ideal signal

Let $u(t)$ be a known input signal. It is transformed by three unknown parameters in equation.

$$w(t) = au(t - t_0) + b + n(t) \tag{4.5}$$

where $n(t)$ includes all distortions and noises introduced by the ADC. Distortions will depend on the input signal generally, but we approximate it is independent here. Further we restrict the type of $u(t)$ to be cyclic with a period T, zero mean and power p_u. We do not discriminate equivalent waves of $u(t)$, whether it is the input signal or it is the ideal waveform of the ADC output.

$$\frac{1}{T}\int_0^T u(t)\,dt = 0 \tag{4.6}$$

$$\frac{1}{T}\int_0^T u(t)^2 dt = p_u \tag{4.7}$$

We assume noise average and variance in a period are 0 and σ_n^2 respectively. Further we assume it is independent of $u(t)$.

$$\frac{1}{T}\int_0^T n(t)\,dt = 0 \tag{4.8}$$

$$\frac{1}{T}\int_0^T n(t)^2 dt = \sigma_n^2 \tag{4.9}$$

$$\frac{1}{T}\int_0^T n(t)\,u(t-\tau)\,dt = 0 \quad \text{for any } \tau \tag{4.10}$$

Let $\hat{w}(t)$ be the equivalent wave of $w(t)$.

$$\hat{w}(t) \stackrel{\text{def}}{=} \hat{a}\,u(t - \hat{t}_0) + \hat{b}\,. \tag{4.11}$$

The LMS estimation of $w(t)$ is provided as follows using assumptions (4.8), (4.9) and (4.10).

$$\begin{aligned} & \frac{1}{T}\int_0^T (w(t) - \hat{w}(t))^2 dt \\ = \; & \frac{1}{T}\int_0^T (a\,u(t - t_0) - \hat{a}\,u(t - \hat{t}_0))^2\,dt + (b - \hat{b}(t))^2 + \frac{1}{T}\int_0^T n(t)^2\,dt \\ \geq \; & \sigma_n^2\,. \end{aligned} \tag{4.12}$$

where the equation holds when $\hat{a} = a$, $\hat{t}_0 = t_0$ and $\hat{b} = b$.

Then we determine three parameters respectively. The offset b is expressed by averaging $w(t)$.

$$\begin{aligned} \frac{1}{T}\int_0^T w(t)\,dt &= \frac{a}{T}\int_0^t u(t-t_0)\,dt + b \\ &= b\,. \end{aligned} \tag{4.13}$$

In order to get a formula of a, we consider the convolution of $w(t)$ and $u(-t)$, or also known as the cross correlation between $w(t)$ and $u(t)$. Using the symbol $*$ for convolution

$$\begin{aligned} w(t) * u(-t) &\stackrel{\text{def}}{=} \int_0^T w(\tau)\,u(\tau - t)\,d\tau \\ &= a\int_0^T u(t-t_0)\,u(\tau-t)\,d\tau + b\int_0^T u(\tau-t)\,d\tau + \int_0^T n(\tau)\,u(\tau-t)\,dt \\ &= a\int_0^T u(\tau + (t-t_0))\,u(\tau)\,d\tau \\ &\leq a\int_0^T u(\tau)^2\,d\tau\,. \end{aligned} \tag{4.14}$$

The last transformation is the Cauchy-Schwarz inequality. The equality holds when $t = t_0$. Using this value as $\hat{t}_0$, we determine

$$\hat{a} = \frac{\max_{t_0}\left(w(t) * u(-t)\right)}{\displaystyle\int_0^T u(t)^2 dt} \tag{4.15}$$

$$\hat{b} = \frac{1}{T}\int_0^T w(t)\,dt\,. \tag{4.16}$$

The calculation cost of a convolution is reasonable in modern computers because it becomes a simple product after Fourier transform (Wiener-Khinchin theorem).

4.3 Candidates for composite signals

Theoretical requirements for the test signal $u(t)$ in the previous section is rather general. Any cyclic signal will work fine. Then are there any criteria to select a 'good' test signal? It had better be expressed in a simple equation for generation. The waveform should be also simple for visual recognition. I am not sure what it is, but this section proposes two candidates. One

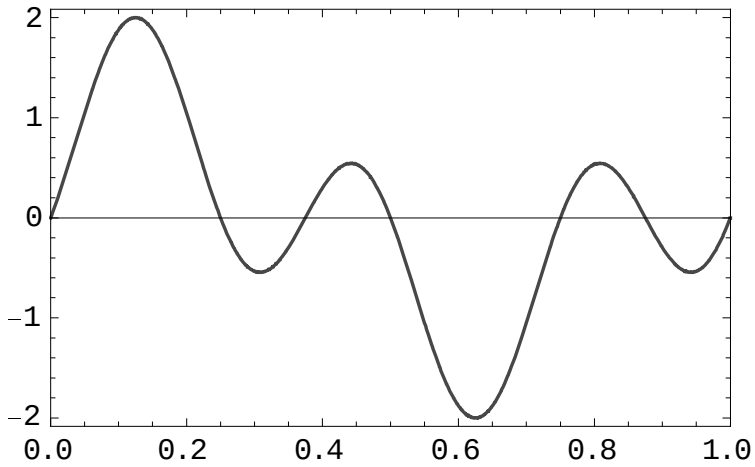

Figure 4.6: A two-tone signal: $\sin(2\pi(t + 0.125)) + \sin(2\pi(3t - 0.125))$

is two-tone and another is duplex sine. The latter may have no common name so that I named it. Both of them are suitable for simulation, while they may be currently difficult to generate in practical measurement.

A general two-tone signal $u_{tt}(t)$ is a sum of two sine waves.

$$u_{tt}(t) = a_1 \sin(2\pi(f_1 t + p_1)) + a_2 \sin(2\pi(f_2 t + p_2)) + b \tag{4.17}$$

Two-tone signal is a signal commonly used for intermodulation measurement. For the purpose, f_1 and f_2 are relatively close and the initial phase difference does not matter. On the other hand for waveform reconstruction, the phase difference should be controlled for a designated value. The frequencies should have a simple fractional ratio so that they are usually quite different. An example of the waveform is shown in Fig. 4.6. For this specific function, the auto-correlation $u_{tt}(t) * u_{tt}(-t)$ has an explicit form

$$\frac{1}{2}\cos(2\pi t) + \cos(6\pi t)\,,$$

that is shown in Fig. 4.7. According to the Cauchy-Schwarz inequality, the auto-correlation function has its maximum value at $t = 0$ as expected.

The “duplex sine”, or sind I named, is a juxtaposition of two sine waves of different frequen-

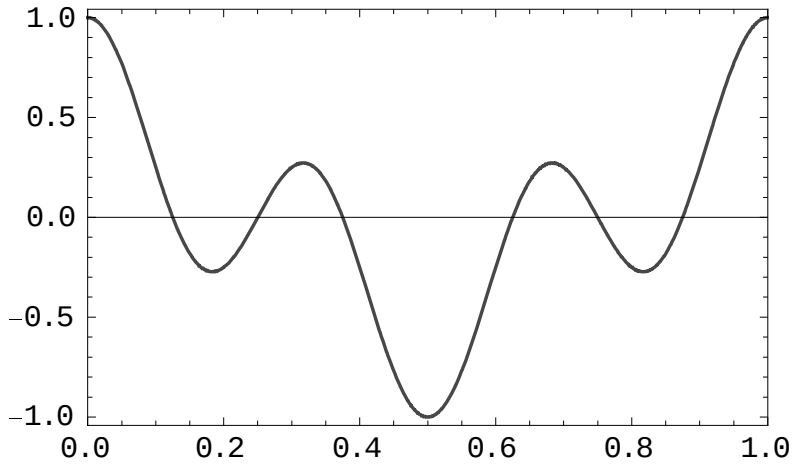

Figure 4.7: Auto-correlation of a two-tone signal: $\sin(2\pi(t + 0.125)) + \sin(2\pi(3t - 0.125))$

cies. It is expressed in equations below.

$$\mathrm{sind}(x, a) \stackrel{\text{def}}{=} \begin{cases} \sin\left(2\pi \dfrac{\tilde{x}}{\lambda}\right) & \text{when } \tilde{x} \leq a \\ \dfrac{1-a}{a} \sin\left(2\pi \dfrac{\tilde{x}-\lambda}{1-\lambda}\right) & \text{when } a < \tilde{x} \end{cases} \tag{4.18}$$

where $\tilde{x} = x - \lfloor x \rfloor$. For positive x, $\tilde{x}$ is the decimal fraction of x. The magnitude of the second sine wave is selected as the waveform is smooth both at $\tilde{x} = 0$ and at $\tilde{x} = a$. In mathematical term, this property is called a class C^1.

Since duplex sine signal has only one additive parameter from a monotone sine wave, the waveform looks much simpler than two-tone signal. Another difference is its power spectra spread for a few harmonics from the beginning. Still they are concentrated for low order harmonics, so Fig. 4.9 shows spectra till 6th order. The ordinate is linear scale of power to grasp a quantitative image. Though the frequency of the higher component becomes higher as λ gets larger, the second and third harmonics become smaller.

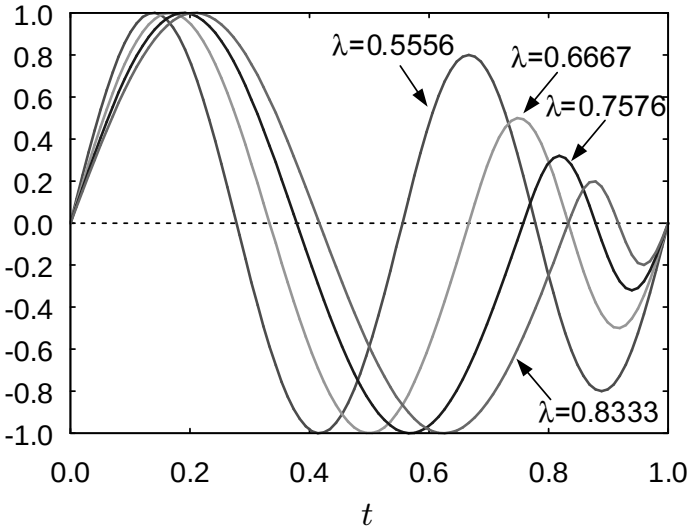

Figure 4.8: Duplex sine signals

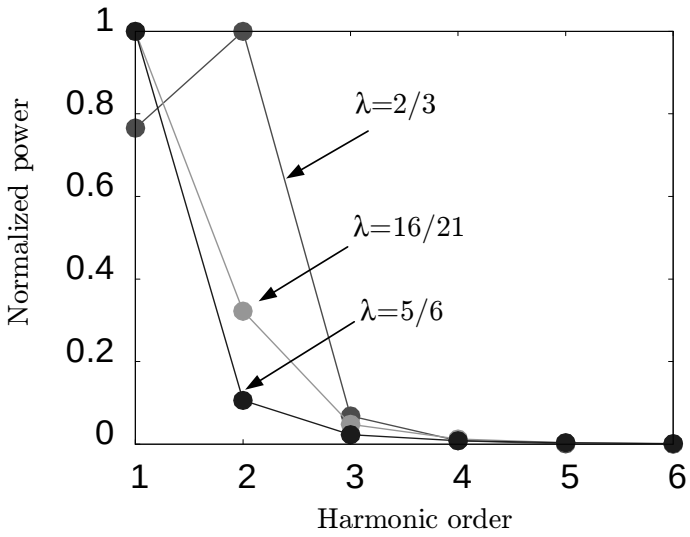

Figure 4.9: Low order spectra of duplex sine signals: sind$(t, 2/3)$, sind$(t, 21/16)$ and sind$(t, 5/6)$

4.4 Reconstruction of two-tone signals

The application of two-tone signal to practice poses a problem how to determine the two frequencies. There are additional requirements for the ratio for reconstruction. In practice, it will be preferable to select the base tone f_1 for first and then to choose the second tone f_2 around the admissible frequencies. One sufficient condition for two tone signal reconstruction is the greatest common divider (GCD) of two tones is odd, roughly speaking. This sufficient condition seems sufficient enough for ADC evaluation.

More precise expressions as follows. We take powers of two as the clock frequency number n_{clk}. Then the input frequency number n_{in} becomes odd for reconstruction. If two frequencies f_1 and f_2 have also integer ratios to the base frequency f_0,

$$f_1 = n_1 f_0 \tag{4.19}$$
$$f_2 = n_2 f_0 , \tag{4.20}$$

we can take the input frequency number

$$n_{in} = \mathrm{GCD}(n_1, n_2) . \tag{4.21}$$

The second tone f_2 is calculated

$$f_2 = \frac{m_2}{m_1} f_1$$

where m_2/m_1 is a irreducible fraction. Then following equations hold.

$$n_1 = m_1 n_{in} \tag{4.22}$$
$$n_2 = m_2 n_{in} \tag{4.23}$$

A simple ratio is preferable for m_2/m_1 since then n_{in} and n_1 become so different that the reconstructed waveforms has lot of short cycles and spoils the visibility of the distortion.

The following Perl script provides f_1 and f_2 from the target frequency f_1 and the base frequency f_0 assuming m_2 and m_1 are relatively prime.

```
$nin = (floor($f1 / $f0 / 2 / $m1 + 0.5) * 2  + 1 );
$f1 = $m1 * $nin * $f0;
$f2 = $m2 * $nin * $f0;
```

As a numerical example, we take the following values for the target $f_{clk} = 100$ MHz and $f_1 = 10$ MHz.

$$\begin{aligned} n_{clk} &= 1024 \\ f_0 &= 0.1\,\mathrm{MHz} \\ m_1 &= 1 \\ m_2 &= 3 \end{aligned}$$

Calculated values are

$$\begin{aligned} f_{clk} &= 102.4\,\text{MHz} \\ n_{in} &= 101 \\ f_1 &= 10.1\,\text{MHz} \\ f_2 &= 30.3\,\text{MHz} \end{aligned}$$

Let use a simple ADC model in Fig. 4.4 here again. Fig. 4.10 shows the sampled data of 20 μsec simulation. Last 1024 samples are used from just over 2,000 samples (clocks) in simulation in order to avoid initial settling of the simulation. Fig. 4.11 (left) shows the reconstructed waveform with the input two-tone amplified by 10 for visibility. The parameters of the input two-tone are $a_1 = a_2 = 1$, $f_1 = 1$, $f_2 = 3$, $p_1 = 0.125$, and $p_2 = -0.125$ in equation (4.17). The reconstructed waveform has offset -0.370117 due to INL of the model ADC. Fig. 4.11 (right) shows the cross correlation of two waves in left of the Figure. From the equation (4.15), the peak position (954, 15.5087) means right rotation (T-transform) by 954 samples of the base two-tone. The quotient of 15.5087 by the power of the base two-tone provides the multiplicand for M-transform. Since the power of the base two-tone happens to be one, the value coincides with the multiplicand. With the offset adjustment (L-transform) by -0.370117, we get the estimated wave in Fig. 4.12. As a summary, we get the best estimation of the ideal wave using (4.16) and (4.15) to (4.17).

$$\hat{w}(x) = \frac{15.5087}{1.0}\left(\sin\left(2\pi\left(\frac{x-954}{1024}+\frac{1}{8}\right)\right)+\sin\left(2\pi\left(\frac{3(x-954)}{1024}-\frac{1}{8}\right)\right)\right)-0.370117\,. \qquad (4.24)$$

The calculated SINAD from the difference wave is 22.65 dB. This is much worse than 36.50 dB calculated INL distortion in Table 1.2.

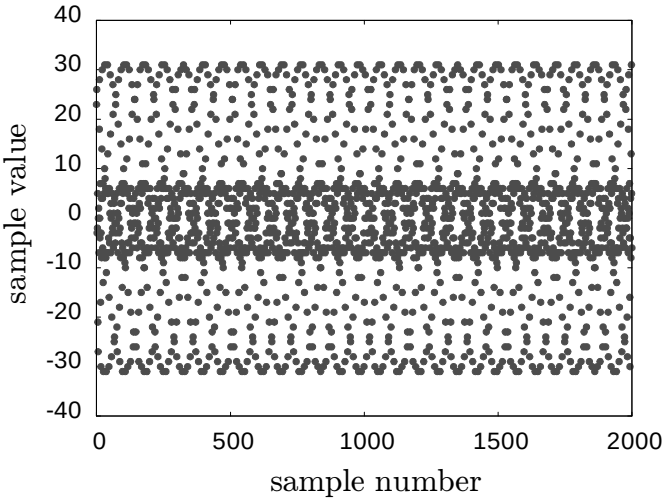

Figure 4.10: Simulated sampled sequence for two-tone signal: f_1 = 10.1 MHz, f_2 = 30.3 MHz and f_{clk} = 102.4 MHz

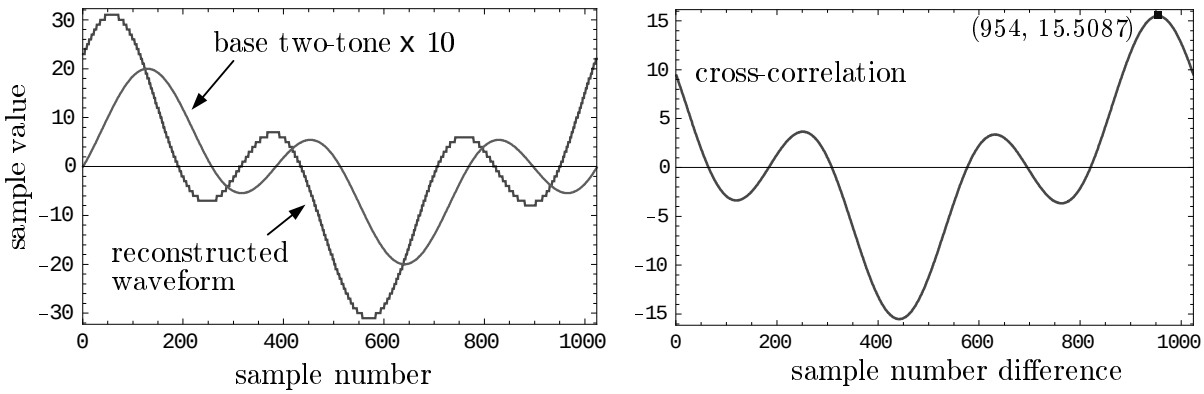

Figure 4.11: The reconstructed waveform of two-tone signal (left) and the cross-correlation with the base wave (right)

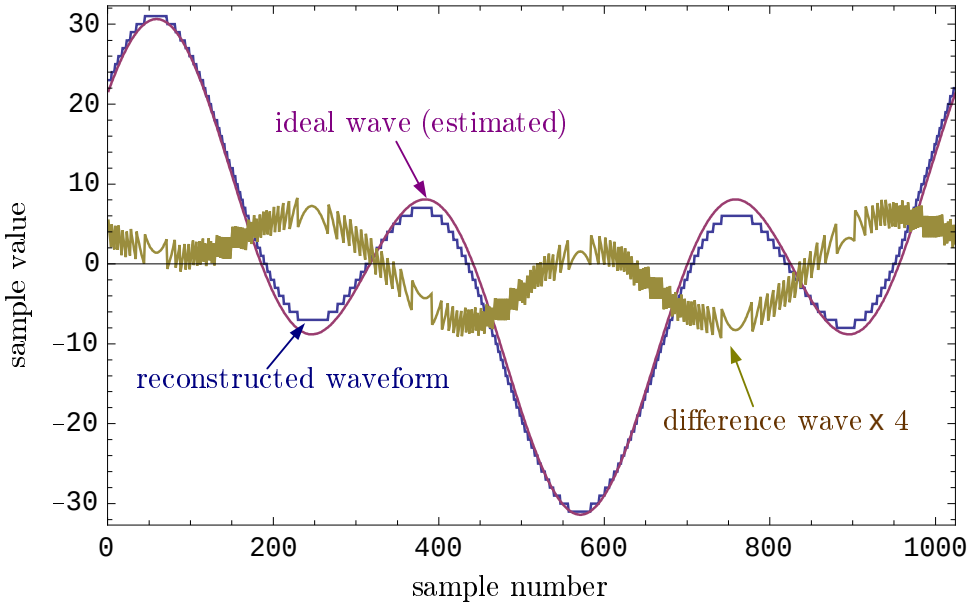

Figure 4.12: The reconstructed waveform of two-tone signal with the difference wave

4.5 Reconstruction of duplex sine signals

The calculation for duplex sine signal goes parallel with two-tone signal. Only the condition for two frequencies is different. The frequency f_{in} of a duplex sine signal is a combination of f_1 and f_2.

$$\frac{1}{f_{in}} = \frac{1}{f_1} + \frac{1}{f_2} \tag{4.25}$$

The relations with the parameter λ are

$$\lambda = \frac{f_2}{f_1 + f_2} = \frac{\mu}{1+\mu} \tag{4.26}$$

$$f_1 = \frac{f_{in}}{\lambda} \tag{4.27}$$

$$f_2 = \frac{f_{in}}{1-\lambda} \tag{4.28}$$

$$\mu = \frac{f_2}{f_1} = \frac{\lambda}{1-\lambda} \tag{4.29}$$

where μ is the frequency ratio of two components. If we consider the frequency resolution of signal sources, both (4.27) and (4.28) should be within the selection of the source. If we allow 3-digit extension from f_{in} resolution, we require the condition that both $1000/\lambda$ and $1000/(1-\lambda)$ are integer. The choices are limited as shown in Table 4.1, but it looks enough for evaluation. In simulation, we have enough resolution so that we can use calculated frequencies.

Table 4.1: Admissible λ's of duplex sine within 3-digit extension, and less than 10-times frequency ratio. Two center columns show a fractional expression of λ.

$\lambda = n/m$			$\mu = f_2/f_1$
0.555555555556	5	9	1.250
0.615384615385	8	13	1.600
0.666666666667	2	3	2.000
0.714285714286	5	7	2.500
0.757575757576	25	33	3.125
0.800000000000	4	5	4.000
0.833333333333	5	6	5.000
0.862068965517	25	29	6.250
0.888888888889	8	9	8.000
0.909090909091	10	11	10.000

The following Perl script calculates the test settings.

```
$f0 = floor($fc / $nclk / $fres + 0.5) * $fres;
$fc = $f0 * $nclk;
$nin = floor( $f1 * $f2 / ($f1 + $f2) / $f0 / 2 ) * 2 + 1;
$lambda = $f2 / ($f1+$f2);
$mu = $f2 / $f1;
$f1 = $nin * $f0 / $lambda;
$f2 = $nin * $f0 / (1-$lambda);
```

where inputs are

```
$nclk   clock frequency number
$fres   frequency resolution (before extension)
$fc     target clock frequency
$f1     target f1 frequency
$f2     target f2 frequency
```

and outputs are, overwriting target frequencies,

```
$f0     base frequency
$nin    input frequency number
$fc     calculated clock frequency
$f1     calculated f1
$f2     calculated f2
```

and λ and μ for the following processing. In practice, μ will be previously selected using or without using Table 4.1.

As a numerical example, we take the following values for the target $f_{clk} = 100$ MHz and $f_1 = 10$ MHz.

$$\begin{aligned} f_{res} &= 0.01\,\text{MHz} \\ n_{clk} &= 1024 \\ \mu &= 2 \end{aligned}$$

Calculated values are

$$\begin{aligned} f_0 &= 0.1\,\text{MHz} \\ f_{clk} &= 102.4\,\text{MHz} \\ n_{in} &= 67 \\ f_1 &= 10.05\,\text{MHz} \\ f_2 &= 20.10\,\text{MHz} \end{aligned}$$

We use an ADC model in Fig. 4.4 again. Fig. 4.13 shows the sampled data of 20 μsec simulation. Last 1024 samples are used for reconstruction. Fig. 4.14 shows the reconstructed waveform and the cross correlation to the original duplex sine signal. The ideal wave is expressed from the peak coordinate (979, 11.5642) of the cross correlation in Fig. 4.14 (right) and the signal average -0.314453 as

$$\hat{w}(x) = \frac{11.5642}{0.375}\,\text{sind}\left(\frac{x-979}{1024}, \frac{2}{3}\right) - 0.314453\,. \tag{4.30}$$

where the denominator 0.375 of the first factor is the power of the input duplex sine signal. The signal adjustment provides Fig. 4.15. Calculated SINAD is 33.88 dB. The value itself is much better than 22.65 dB in two-tone test in Fig. 4.12.

In conventional measurement, the input amplitude is adjusted at each frequency [5, 5.5]. This adjustment changes the M-transform parameter. In a simple ADC model in Fig. 4.4, frequency response of the ADC disappears by this operation. On the other hand, by using duplex sine signal,

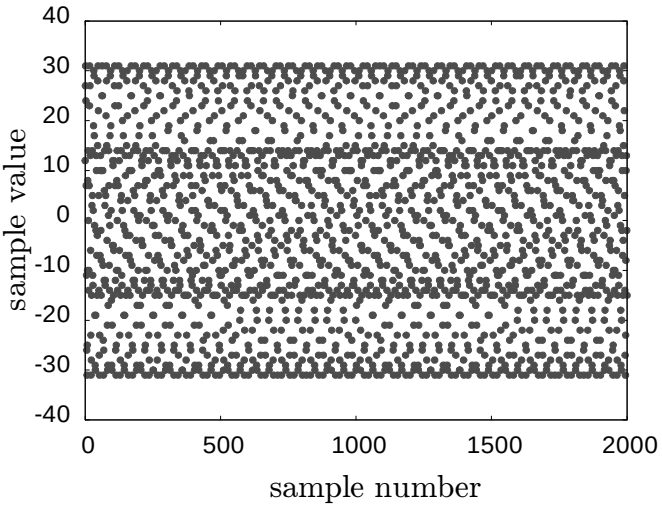

Figure 4.13: Simulated sampled sequence for duplex singe signal: f_1 = 10.05 MHz, f_2 = 20.10 MHz and f_{clk} = 102.4 MHz

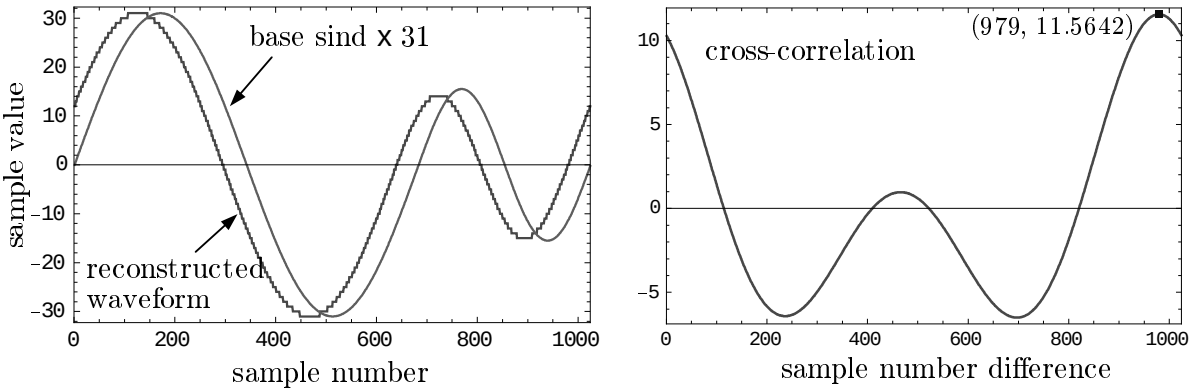

Figure 4.14: The reconstructed waveform of duplex sine signal (left) and the cross-correlation with the base wave (right)

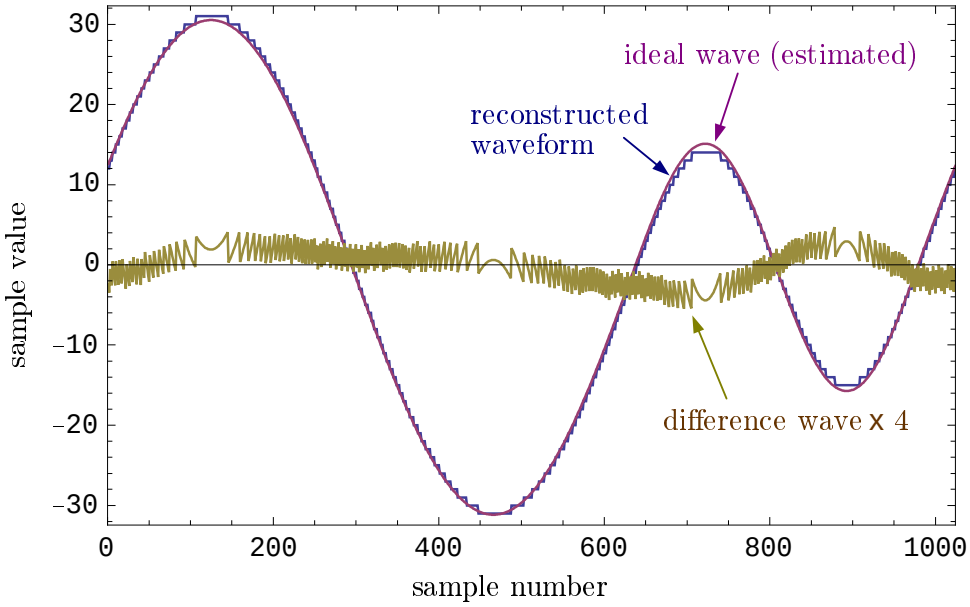

Figure 4.15: The reconstructed waveform of duplex sine signal with the difference wave

the magnitude adjustment at each frequency remains the frequency response. Fig. 4.16 compares SINAD of pure sine signals and duplex sine signals. All input amplitudes are adjusted to the full scale of the ADC in Fig. 4.4. The frequency ratio of duplex sine signal is just 2 ($\lambda = 2/3$) in all simulations. Since the M-transform parameter varies also in duplex sine signal, the SINAD may not be fully appropriate, but still, the SINAD values seem more practical performance indexes. Fig. 4.17 shows the difference wave changes at various input frequencies.

4.6 Summary

This chapter extended the difference wave analysis method to general test signals other than sine waves. This extension is essential to insure the independence of three transforms in A-to-D conversion of the input, *i.e.* "M-transform" for magnitude, "T-transform" for time shift, and "L-transform" for level shift. The summaries are:

1. The least mean square estimation method for generalized test signals is derived.

2. Two candidates for test signal are proposed: the "two-tone signal" and the "duplex sine signal."

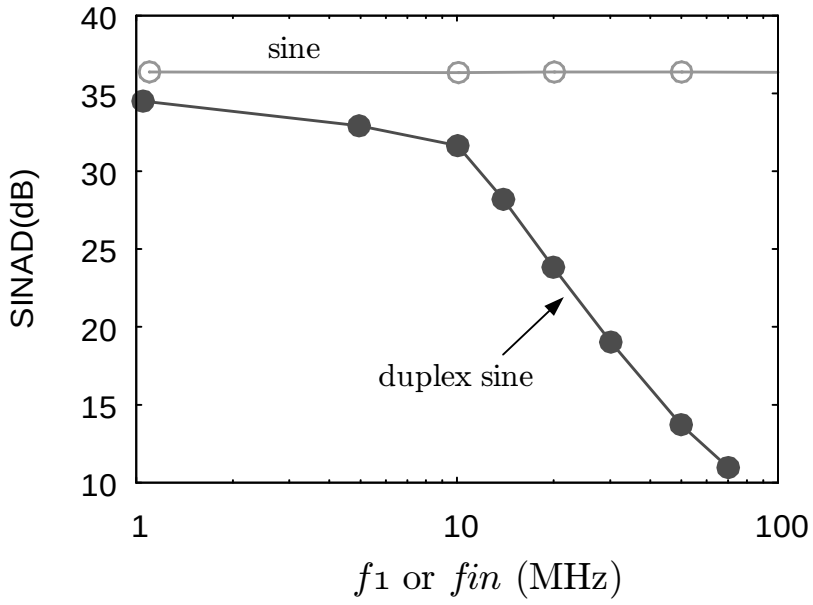

Figure 4.16: SINAD vs. input frequency: comparison between sine and duplex sine ($\lambda = 2/3$)

3. Additional conditions for waveform reconstruction for these two signals are derived.

4. Numerical examples for two signals are investigated. Calculated SINADs are input frequency dependent, while pure sine signal is not.

These results are preliminary but they demonstrate the effectiveness of the composite waves for the characterization of ADC.

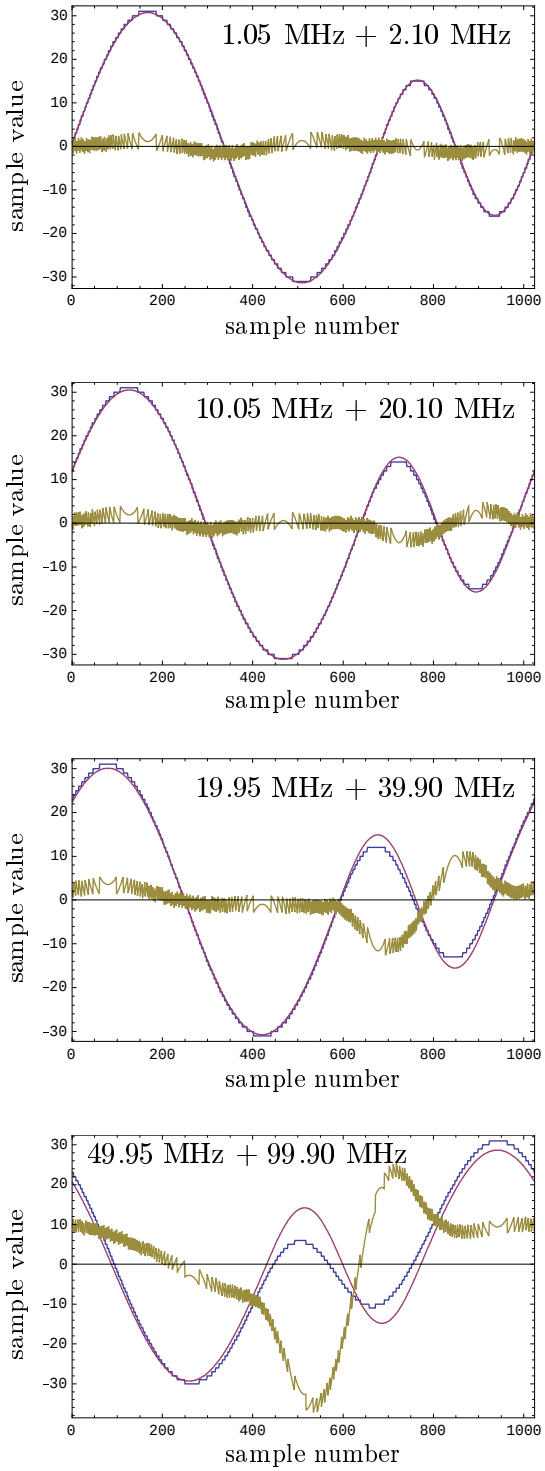

Figure 4.17: The reconstructed duplex sine waveforms ($\lambda = 2/3$) of several frequencies

Chapter 5

Conclusion

I conclude the thesis briefly as follows.

The research originally started to improve the performance ADCs. The purpose required much finer resolution of the ADC characterization. It should be fine enough to determine the transistor sizes and currents of internal circuits. I get to understand the characterization of an ADC is, in the end, to get the deviation from the ideal value at each conversion of arbitrary inputs.

The deviation is composed of three elements: the input dependent distortion, the conversion history dependent distortion and the noise. The former two elements are decisive and the last one is random. Only the first one is usually evaluated, but solely using sine waves. Composite signal analysis in Chapter 4 is required to obtain the deviations for arbitrary inputs. The simplest form of the conversion history dependency is the dog-tooth distortion of a flash ADC studied in Chapter 3. The random components append the statistical nature to the conversion. It is grasped first in linearity measurement as analyzed in Chapter 2, but it exists at any conversion.

Then let me conclude a little bit lengthier with what I wanted and what we have got.

5.1 History of research and main results

This thesis is an achievement of my efforts for ADC design since 1989. Results in the thesis are reordered by its theoretical structure. I want to state here relating to my personal history.

The first project I engaged was to develop very fast ADCs. I found the design methodology at that time was quite inadequate. For example, there was no knowledge to determine the comparator current. There were lots of unexpected phenomena, such as sparkles at much lower frequency than expected, fixed pattern distortions, kick-backs, etc. We needed the development of tools to

identify causes of those errors. A little while later, I have gradually noticed the test method itself was also deficient. The largest anxiety is the test method ignores the frequency dependency of ADC, since the input sine wave amplitude is adjusted at each frequency setup.

Some years later I got involved in design of various read channel ICs such for HDDs, tape streamers and optical discs. I became a user-designer of ADC. The experience fed me the knowledge of signal processing, and also strengthened the above anxiety. So the characterization of ADC became my subject.

When I look back old days, notes and measurement data, I am impressed how I was groping the way. I feel the present theory looks natural, or even obvious, but it is a pearl of my struggles.

In Chapter 1, we formalized the characterization of ADC. The analog-to-digital conversion process is decomposed into scaling and digitizing. The scaling is the adjustment of three degrees of freedom of signal, which are magnitude, time origin, and level. The scaling procedure requires the estimation of the ideal wave. If the signal is linear or sinusoid, the estimation is easy. Traditional test methods use these signals fairly exclusively. The proposed problem is the choice of test signals should not be enough.

The linearity measurement by ramp signal introduces the statistical view of the A-to-D conversion. The distortion measurement by sine wave has indicated the necessity of further resolution of analysis to use the measurement for ADC design.

In Chapter 2 has dealt with the statistical nature of the conversion. Specifically, linearity measurement by rising ramp signal is studied using the maximum likelihood method (ML method). The logistic noise is derived as approximations of the ML equations.

In fact, the order of my personal history is opposite. The theory was first inspired while I was brooding to overcome a deficit in aperture jitter measurement known as locked histogram test [1]. In its principle, the test setup feeds the same signal into the analog input and the clock input. The sampling timing is adjusted by a delay line. The ADC output should become a constant in the condition. In practice, the sampled data fluctuate and the obtained histogram is used to calculate the jitter. The problem in the method is the output data are often limited to a few codes. Then the sampling timing changes the code distribution and the measurement results substantially. DNLs at those particular codes may also affect the results.

The idea came to mind was to use a slightly different frequency for the analog input. Then the beat wave appears in the output. We can focus on a designated code, usually the midst of the ADC range, and we only have care of the output is larger or smaller than the code. It is a binary sequence around the designated code transition.

The next idea was needed how to calculate the jitter from the binary sequence. The logistic function (I did not know the name at that time) came to mind to approximate the normal distribution (my Japanese presentation C-487 in the Institute of Electronics, Information and Communication Engineers Society Conference 1995). I devised the formula as the much easier function to handle than the Gaussian distribution. After looking for literatures, I found the func-

tion name. It is the function frequently used in pattern recognition field, occasionally by more general sigmoid forms.

After founding the logistic function I have noticed the same idea is applicable to linearity measurement (my Japanese presentation C-12-45 in the Institute of Electronics, Information and Communication Engineers General Conference 1998). Naturally, my method breaks down the ramp signal into binary sequences from the beginning. The method is also related to the waveform reconstruction technique from its origin. Figures in Section 3.2 touch the capability of the difference wave analysis for jitter measurement.

After derived ML equations with a weight function, I found the logistic equation could be derived from the constant weight assumption. It was surprising to me. The logistic distribution is not a mere shortcut of Gaussian noise but has a solid position to verify the histogram method in ML sense.

The difference and the similarity between Gaussian noise and logistic noise have been my long concern. Many formulas derived for logistic noise are found to be appropriate for Gaussian noise, too. I have managed to calculate variations of estimation of code transition levels and input equivalent noise of both noise distributions in Section 2.6. The explicit expressions in Table 2.2 provide the confidence interval of the CT level and the noise estimation.

Chapter 3 deals with distortion analysis. The theory of the waveform reconstruction method that I named “Euclidean reordering” is provided. The difference wave analysis associated with the reconstruction method is a powerful tool both for analysis and design. Using the method, I have identified the dog-tooth distortion. A latch hysteresis simulation technique is also developed. A new topology of a bipolar latch is developed applying the difference wave method.

The difference wave illustrates the ADC characteristics much more precisely than the power spectrum. I hope it will become a common practice that ADC papers present the difference wave as a measured result.

The waveform reconstruction method is not my idea at all. I am inherited the difference wave method from my predecessors from the very beginning of my experience of ADC. In spite of my intensive search of the origin, I could not find the root. Pápay [19] shows the same idea later than us, but he cites older references. He did not say the method is his idea, either. I looked into some of his references, but to me they dealt with different issues.

The original method I inherited reorders the sampled data literally using working arrays in the program. I could not understand details of the source code, instead, I managed to make the cyclic pickup algorithm by myself. It worked fine, but at first, the method was understood only intuitively to me. Several years were taken for me to establish the Euclidean reordering theory.

The “dog-tooth” distortion was first observed in a 10-bit flash ADC prototype. To reduce the power, I removed emitter followers in the latched-comparator as possible as shown in Fig. 3.26. For years I have assumed the cause of dog-tooth as comparator speed change at around extrema. The invention (JP P2001-198279) of the common resister latch in Fig. 3.30 was in a later project

while pursuing better SINAD. There was no connection to the dog-tooth distortion at first. But the improvement of the dog-tooth by the new latch was recognized immediately after the invention, since the difference wave method had been established already. More later while I was engaged in the CMOS comparator design, I suddenly noticed the cause of dog-tooth must be hysteresis. Data shown in Chapter 3 were taken to verify the insight.

In Chapter 4, we extend the difference wave method to use much general test signals other than sine wave. After reconstructed the waveform, the LMS estimation of the ideal wave is not only possible but also cost effective. Two-tone signal and duplex sine signal are investigated for promising candidates of composite signal analysis. Sweeping the frequency of the duplex sine signal showed the SINAD degradation of a simple modeled ADC that the sine wave test signal did not.

The drawbacks of the pure sine wave have been recognized over years personally, but I had no guideline to choose a specific composite signal. For years, I looked for a set of signals in vain that forms a closed orthogonal system and is easy to treat. It was difficult since the time-shift of a signal generates quite different spectra in non-harmonic systems. In 2011, I noticed the LMS estimation of general class of signals is possible. It is a quite popular technique in signal processing that I have already known. I should have noticed it before the struggle. My unique idea is to apply the Euclidean reordering for those composite signals, such as two-tone or duplex sine signals. Admissible conditions of Euclidean reordering for those composite signals are obtained. The difference wave analysis as well as SINAD calculation is also shown possible.

5.2 Items for the future study

Thought various knowledge and understanding of ADCs have been provided from this study, the study lefts a lot of rooms to cultivate.

Linearity analysis combined with Euclidean reordering will extend the horizon of linearity noise analysis. Any repetitive signal, including saw-tooth, triangular, sine or other composite signals can be sorted in phase order, and then becomes a target of the CT level estimation. The noise seems to have some interference with the input. The combination of methods will provide much finer view of the ADC than mere sight of distortion.

Noise affects the linearity results. I have preliminary data that the variations of repetitive linearity measurements are not always consistent with the noise evaluations of the ramp signal. Hysteresis of the comparator may affect the inconsistency but this item needs for further study

Composite signal analysis introduced in Chapter 4 is in its preliminary level. It is an open question what composite signals are appropriate for standard signals. Combinations of composite signals will be necessary to characterize an ADC fully. The practical generation methods for experiments are wanted to be developed.

Noise and hysteresis are important concern of a latch design. They should have the input dependency but the relations between static characters and dynamic characters are also left for further study.

One goal of the characterization of ADCs will be a modeling. A good model provides accurate results over various situations with reasonable number of parameters. Noise and hysteresis of comparators should be incorporated in the model. The model should show good coincidence over composite signals. If it becomes a standard as SPICE model, it will be beneficial for researchers, designers, and users of ADCs.

My original motivation, to improve ADC design through analysis, is always under the study. Distortion mechanisms, noise mechanisms and speed limit mechanisms are evaluated and identified through this study on characterization of ADCs.

I hope my methods will be applied to numerous ADC designs and help the improvements of them.

Bibliography

[1] W. Kester, *Data Conversion Handbook (Analog Devices)*. Analog Devices Inc., 2005, ISBN: 0-7506-7841-0, [Online] Available: http://www.analog.com/library/analogDialogue/archives/39-06/data_conversion_handbook.html.

[2] R. J. van de Plassche, *Integrated analog-to-digital and digital-to-analog converters*. Kluwer Academic Publishers, 1994, ISBN: 0-7923-9436-4.

[3] T. E. Linnenbrink, J. Blair, S. Rapuano, P. Daponte, E. Balestrieri, L. D. Vito, S. Max, and S. J. Tilden, "ADC testing," *IEEE Instrumentation & Measurement Magazine*, vol. 9, no. 2, pp. 37–47, Apr. 2006.

[4] Waveform Measurement and Analysis Technical Committee, *IEEE Standard for digitizing waveform recorders*, IEEE Std. 1057-2007, Apr. 2008.

[5] ——, *IEEE Standard for terminology and test methods for analog-to-digital converters*, IEEE Std. 1241-2010, Jan. 2011.

[6] *IEC 60748-4-3 Semiconductor Devices - Integrated circuits - Part 4-3: Interface integrated circuits - Dynamic criteria for analog-digital converters (ADC)*, International Electrotechnical Commission Std., Rev. First Edition, Aug. 2006.

[7] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 2nd ed. Prentice-Hall, 1999, ISBN:0-13-754920-2.

[8] R. J. Baker, *CMOS Circuit Design, Layout, and Simulation*, 3rd ed., ser. IEEE Press Series on Microelectronics Systems, S. K. Tewksbury and J. E. Brewer, Eds. John Wiley & sons, 2010, ISBN:978-0-470-88132-3.

[9] S. Max, "Optimum measurement of ADC code transitions using a feedback loop," in *IEEE Instrumentation and Measurement Technology Conference*, vol. 3, May 1999, pp. 1415–1420.

[10] P. D. Capofreddi and B. A. Wooley, "The efficiency of methods for measuring A/D converter linearity," *IEEE Transactions on Instrumentation and Measurement*, vol. 48, no. 3, pp. 763–769, Jun. 1999.

[11] P. N. Variyam and V. Agrawal, "Measuring code edges of ADCs using interpolation and its application to offset and gain error testing," in *IEEE International Test Conference*, Oct. 2000, pp. 349–357.

[12] S. Cherubal and A. Chatterjee, "Optimal INL/DNL testing of A/D converters using a linear model," in *IEEE International Test Conference*, Oct. 2000, pp. 358–366.

[13] L. Jin, K. Parthasarathy, T. Kuyel, D. Chen, and R. L. Geiger, "Accurate testing of analog-to-digital converters using low linearity signals with stimulus error identification and removal," *IEEE Transactions on Instrumentation and Measurement*, vol. 54, no. 3, pp. 1188–1199, Jun. 2005.

[14] L. Balogh, B. Fodor, A. Sárhegyi, and I. Kollár, "Maximum likelihood estimation of ADC parameters from sine wave test data," in *15th IMEKO TC4 Symposium on Novelties in Electrical Measurements and Instrumentation: 12th Workshop on ADC Modeling and Testing*, Sep. 2007, pp. 85–90. [Online]. Available: http://home.mit.bme.hu/ kollar/papers/TC4-2007.pdf

[15] G. Cavone, A. D. Nisio, N. Giaquinto, and M. Savino, "A maximum likelihood estimator for ADC and DAC linearity testing," in *13th Workshop on ADC modeling and testing*, Sep. 2008.

[16] J. G. Peterson, "A monolithic, fully parallel, 8b A/D converter," in *International Solid-State Circuits Conference*, 1979, pp. 128–129.

[17] F. H. Irons and D. M. Hummels, "The modulo time plot – a useful data acquisition diagnostic tool," *IEEE Transactions on Instrumentation and Measurement*, vol. 45, no. 3, pp. 734–738, Jun. 1996.

[18] B. Razavi, *Principles of data conversion system design*. IEEE press, 1994, ISBN: 0-7803-1093-4.

[19] Z. Pápay, "Comments on 'The Modulo Time Plot: A Useful Data Acquisition Diagnostic Tool'," *IEEE Transactions on Instrumentation and Measurement*, vol. 45, no. 6, p. 959, Dec. 1996.

[20] ——, "Correction to comments on 'The Modulo Time Plot: A Useful Data Acquisition Diagnostic Tool'," *IEEE Transactions on Instrumentation and Measurement*, vol. 46, no. 3, p. 739, Jun. 1997.

[21] J. J. Blair, "Selecting test frequencies for sinewave tests of ADCs," *IEEE Transactions on Instrumentation and Measurement*, vol. 54, no. 1, pp. 73–78, Feb. 2005.

[22] Synopsys, *SpiceExplorer and WaveView Analyzer User Guide, A-2008.03*, 2008.

[23] L. Balogh, I. Kollár, and A. Sárhegyi, "Maximum likelihood estimation of ADC parameters," in *Instrumentation and Measurement Technology Conference (I2MTC)*. Dept. of Meas. & Inf. Syst., Budapest Univ. of Technol. & Econ., Budapest, Hungary, May 2010, pp. 24–29. [Online]. Available: http://mycite.omikk.bme.hu/doc/84802.pdf

[24] W. Feller, *An introduction to probability theory and its applications*, 3rd ed. John Wiley & sons, 1968, vol. I, ISBN: 0-471-25711-7.

[25] R. V. Hogg, J. W. McKean, and A. T. Craig, *Introduction to mathematical statistics*, 6th ed. Pearson Prentice Hall, 2005, ISBN:0-13-122605-3.

[26] E. W. Weisstein, *CRC concise encyclopedia of mathematics*. Chapman & Hall/CRC, 1998, ISBN: 0-8493-9640-9.

[27] S. Max, "IEEE Std 1241: the benefits and risks of ADC histogram testing," in *IEEE Instrumentation and Measurement Technology Conference*, vol. 1, May 2001, pp. 704–709.

[28] M. Miyahara, Y. Asada, D. Paik, and A. Matsuzawa, "A low-noise self-calibrating dynamic comparator for high-speed ADCs," in *IEEE Asian Solid-State Circuit Conference*, Nov. 2008, pp. 269–272.

[29] J. Márkus and I. Kollár, "Standard framework for IEEE-STD-1241 in MATLAB," in *IEEE Instrumentation and Measurement Technology Conference*. IEEE, May 2001, pp. 1847–1852.

[30] S. C. Kak and N. S. Jayant, "On speech encryption using waveform scrambling," *Bell system technical journal*, vol. 56, no. 5, p. 5, May 1977.

[31] J. H. Silverman, *A friendly introduction to number theory*, 3rd ed. Pearson Education Inc., 2005, ISBN: 0-13-186137-9.

[32] A. N. Kolmogorov and S. V. Fomin, *Introductory real analysis*, R. A. Silverman, Ed. Dover, 1975, ISBN:0-486-61226-0.

[33] A. Zygmund, *Trigonometric Series*, 3rd ed. Cambridge, 2003, ISBN:0-521-89053-5.

[34] A. Matsuzawa, "超高速・高精度 A/D 変換 LSI に関する研究 (A study of ultra fast accurate A/D conversion LSI)," Ph.D. dissertation, Tohoku University, 1997, in Japanese.

[35] C. W. Mangelsdorf, "A 400-MHz input flash converter with error correction," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 1, pp. 184–191, Feb. 1990.

[36] Y. Gendai, "A 6-bit 340 Msps BiCMOS ADC of 1.8 V single power supply adopting folding logic," *IEICE Transactions on Electronics*, vol. E85-C, no. 8, pp. 1546–1553, Aug. 2002.

[37] D. Dalton, G. J. Spalding, H. Reyhani, T. Murphy, K. Deevy, M. Walsh, and P. Griffin, "A 200-MSPS 6-bit flash ADC in 0.6-μm CMOS," *IEEE Transactions on Circuits and Systems—Part II: Analog and Digital Signal Processing*, vol. 45, no. 11, pp. 1433–1444, Nov. 1998.

[38] R. Petschacher, B. Zojer, and W. Luschnig, "New methods improving static and dynamic performance of an 8-bit/120-MHz A/D converter," in *11th European Solid-State Circuits Conference*, 1985, pp. 40–44.

[39] Datel, "ADC-207 7-bit, 20MHz, CMOS flash A/D converters," http://www.murata-ps.com/data/ads/adc-207.pdf.

[40] Y. Gendai, "Comparator circuit," U.S. Patent 6 710 733, Mar. 23, 2004.

[41] Y. Akazawa, A. Iwata, T. Wakimoto, H. Nakamura, and H. Ikawa, "A 400MSPS 8b flash AD conversion LSI," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb. 1987, pp. 98–99, 357.

[42] K. Okamoto, T. Morie, A. Yamamoto, K. Nagano, K. Sushihara, H. Nakahira, R. Horibe, K. Aida, T. Takahashi, M. Ochiai, A. Soneda, T. Kakiage, T. Iwasaki, H. Taniuchi, T. Shibata, T. Ochi, M. Takiguchi, T. Yamamoto, T. Seike, and A. Matsuzawa, "A fully-integrated 0.13μm CMOS mixed-signal SoC for DVD player applications," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, 2003, pp. 38–476.

[43] *IEEE standard hardware description language based on the Verilog(R) hardware description language*, IEEE Std. 1364-1995, Oct. 1996.

[44] B. Razavi, Y. Ota, and R. G. Swartz, "Design techniques for low-voltage high-speed digital bipolar circuits," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 3, pp. 332–339, Mar. 1994.

[45] H. Taub and D. Schilling, *Digital Integrated Electronics*. McGraw-Hill Book Company, 1977, ISBN:0-0708-5788-1.

[46] Y. Nejime, M. Hotta, and S. Ueda, "An 8b ADC with over-nyquist input at 300Msps conversion rate," in *Proc. IEEE 1990 Bipolar Circuits and Technology Meeting*, Sep. 1990, pp. 210–213.

[47] H. Ammo, H. Ejiri, S. Kanematsu, H. Kikuchi, M. Yano, and H. Miwa, "A complementary BiCMOS technology for low power wireless telecommunication applications," in *Proc. of 29th European Solid-State Device Research Conf.*, vol. 1, Sep. 1999, pp. 444–447.

Appendix A

A reference design of a flash ADC

This appendix describes a reference design of a 6-bit flash ADC. In fact, several implementations of the same architecture are used in this thesis. Types cover 6-bit and 7-bit resolutions, from 100 MSps to 1 GSps, in bipolar and in BiCMOS. They are mass-produced by several millions as a whole.

I have reported an implementation in 2002 [36]. It is fabricated in bipolar based BiCMOS. It is designed for 1.8-V operation. The measurement revealed that it operates up to 340 MSps at 1.26 V power supply consuming 36 mW. The conversion rate per power performance index of 1.65 pJ/conv was among the best in fast 6-bit ADCs reported at the time.

To operate at relatively low supply voltage for a bipolar ADC, we developed a novel encoder scheme together with the unique layout, which improved sparkle error rate substantially. The encoder circuit was synthesized in a new logic topology that we named “folding logic.” This original logic topology is not only suitable for low-voltage operation but also fast operation.

A.1 Conventional flash ADC topology and obstacles to fast and low voltage operation

Figure A.1 shows a simplified schematic of a conventional flash ADC. The principle of operation is to compare the analog input V_{IN} voltage with reference voltages, concurrently, using an array of comparators. For n-bit width, $2^n - 1$ comparators are necessary. The reference voltages are usually generated from a so-called “resistor string” which equally divides between the reference voltage top V_{RT} and the reference voltage bottom V_{RB} .

In the operation the row of comparators is divided into two regions depending on those states. One region is formed by high comparators and the other is by low state comparators. This

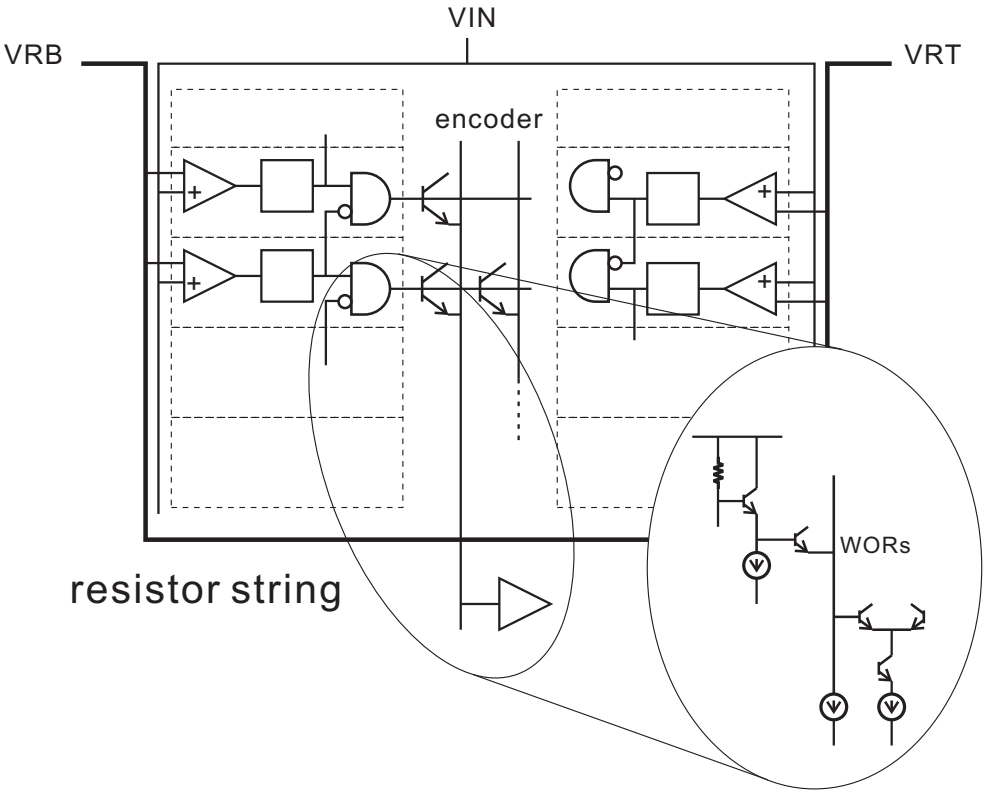

Figure A.1: Conventional ADC layout and its stacked series gating

division forms a so-called “thermometer code.” The boundary swings according to the input voltage. The subsequent encoder stage converts the thermometer code into the binary code.

The encoder scheme usually adopts a combination of a row of differentiators and an encoder ROM. Differentiators detect the boundary of the comparators’ states, where one side is high and the other side is low. In normal operation, only one differentiator outputs high. The encoder ROM adopts a wired-OR (WOR) scheme to generate an n-bit binary output code.

This configuration assumed the supply voltage of 5 V or above, so we have to review totally the design for 1.8 V operation. Among many problems, the most difficult obstacle for low voltage operation is the encoder WORs. In Fig. A.1, the encoder is incorporated into the series gating of four NPN transistors, the first for the differentiator output, the second for encoder WORs, the third and the fourth for a latch. Additionally some headroom voltage for tail current sources is required. At 5V operation, this series gating is not only possible but also advantageous both in speed and power because this is the critical timing path in the conventional flash ADC. However

since V_{BE} is about 0.7 V, this configuration is impossible for 1.8 V operation. We are forced to leave the classical encoder circuit topology.

A combination of logic gates is the inevitable choice, but the direct conversion of the conventional encoder logic may cancel the power reduction of the low voltage operation by requiring too much additional current. Besides, the conventional encoder inherently introduces sparkle error (digital error) mechanisms. The new encoder scheme had better overcome these problems.

Dynamic range of the ADC, i.e. V_{RT} and V_{RB} selection, and the comparator topology are other design issues for 1.8 V operation. The supply voltage limits the choice of the comparator design and this ADC adopted a simple differential pair of bipolar transistors. The bases are directly connected to the analog input and reference voltages. In this form, V_{RT} can be equal to the supply voltage V_{CC}, which is the highest voltage level in the ADC. Wider dynamic range, or lower V_{RB} , is preferable to reduce differential non-linearity (DNL), but it is limited by the comparators' operating region. The lowest reference voltage V_{RB} must be greater than the base-emitter voltage V_{BE} plus headroom for the tail current source. Using a MOS transistor as the source, 1.2 V will be minimum and preferably higher than 1.4 V, when low temperature operation is considered. The good pair balance of bipolar transistors is helpful to satisfy this constraint. We will discuss a specific implementation later in section A.5.

A.2 Encoder scheme and the ADC layout

The principle we have chosen to suppress the meta-stable effects is to place as many latch stages as possible before the critical section in which a meta-stable (threshold level) input condition may generate sparkle errors. In order to reduce the total number of gates, we also have to try to reduce the number of latches gradually at stages.

The first idea is to apply the Gray coding directly without a differentiator. Let's name the comparator outputs as C_1 to C_{63} from the bottom to the top. The direct Gray encoder for 3 most significant bits (MSBs) E_3 to E_5 is

$$\begin{aligned} E_5 &= C_{32} \\ E_4 &= C_{16} \,\&\, \tilde{}C_{48} \\ E_3 &= (C_8 \,\&\, \tilde{}C_{24}) \,|\, (C_{40} \,\&\, \tilde{}C_{56}) \end{aligned} \tag{A.1}$$

where the notation for logic expressions follows Verilog-HDL standard [43], specifically ˜ for negation, & for AND, | for OR, and ∧ for exclusive-OR (XOR).

In this encoding, each comparator appears only once. This property is important because if one comparator output is meta-stable, and if the state propagates to the next stage, only one code bit from E_3 through E_5 becomes meta-stable. Since one code bit change in Gray code means a

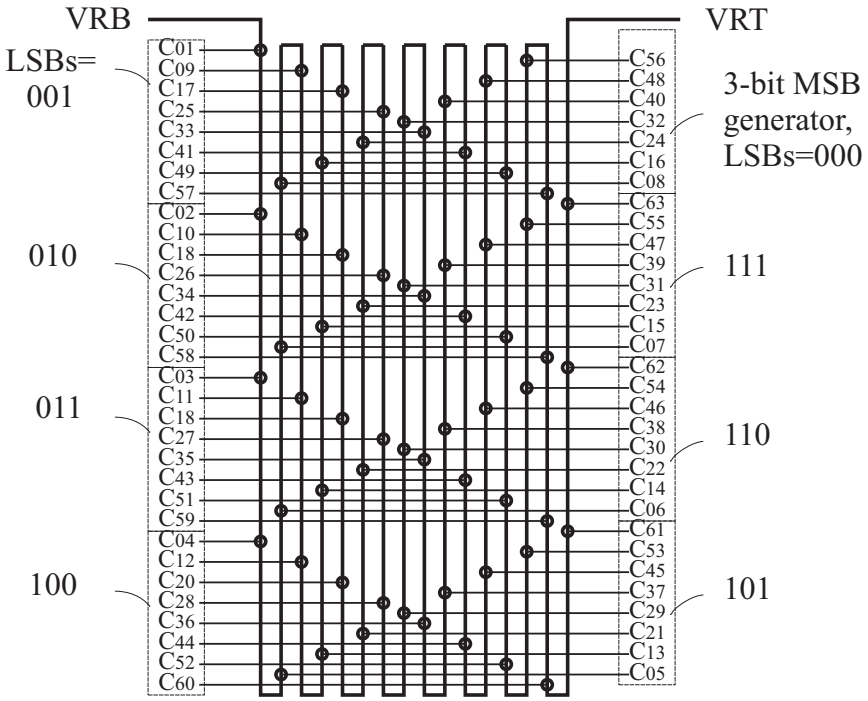

Figure A.2: Layout of the resistor string and comparators

mere shift to the adjacent code, the decision will not matter too much. So we can postpone the critical section after Gray coding, reducing the number of latches from 7 to 3.

Comparators in Eq. (A.1) are spread over the chip in the traditional placement of Fig. A.1. These comparators can be placed adjacently by folding the resistor string. Figure A.2 illustrates the comparator layout. This folding also brings out a natural placement of comparators for the least significant bits (LSBs). Comparators are grouped into 8 clusters. Each cluster represents one unique combination of three-bit LSBs. In other words, comparators in each cluster have the same 3-LSB value. The right upper corner cluster generates 3-bit MSBs, whose LSBs are binary "000." In this thesis, we call this architecture the "folded resistor string and grouped comparator architecture," or FRSGCA in short.

It should be noted that the small offset variation of the bipolar comparator enables arbitrary placement of the comparators, which is helpful for the FRSGCA implementation. In contrast, the interpolation technique common in recent MOS ADCs.

The comparator clusters, other than the MSB, have the same internal encoder and produce an intermediate one-bit code each. The first LSB encoder is:

$$\begin{aligned} F_i = \quad & (C_i \,\&\, \tilde{}C_{i+8}) \,|\, (C_{i+16} \,\&\, \tilde{}C_{i+24}) \\ & |\, (C_{i+32} \,\&\, \tilde{}C_{i+40}) \,|\, (C_{i+48} \,\&\, \tilde{}C_{i+56}) \end{aligned} \tag{A.2}$$

where $i = 1$ to 7 that corresponds to 3-bit LSBs.

The MSBs of the final two's complement output are given by the following Gray-binary converter

$$\begin{aligned} D_5 &= \tilde{}E_5 \\ D_4 &= E_4 \wedge E_5 \\ D_3 &= E_3 \wedge E_4 \wedge E_5 \end{aligned} \tag{A.3}$$

We can generalize the converter for wider bits but it requires more terms of XOR, which is difficult to speed up. This is why we have restricted the MSB width to 3 bits. The following final LSB encoder generates the binary LSBs.

$$\begin{aligned} D_2 &= F_4 \wedge D_3 \\ D_1 &= (F_2 \wedge F_4) \,|\, (F_6 \wedge D_3) \\ D_0 &= (F_1 \wedge F_2) \,|\, (F_3 \wedge F_4) \\ & \quad |\, (F_5 \wedge F_6) \,|\, (F_7 \wedge D_3) \end{aligned} \tag{A.4}$$

Figure A.3 depicts the latch stages of the ADC. The index of each latch shows the number of latches. Encoders from Eq. (A.1) to Eq. (A.4) are the combination logics placed between latches. The final LSB encoder uses D'_3 instead of D_3 in Eq. (A.4), which is one latch stage earlier than D_3, to keep the stages consistent.

The LSB encoder does not have the Gray coding advantage. However, the critical section, most sensitive to meta-stability, comes after five latch stages in this ADC. The probability of a meta-stable state propagating this far is negligible.

On the other hand, the lower bit encoding scheme is not so immune from the bubble error, but the sparkle amplitude is limited within 3-bit width LSBs. For example, assume a bubble near the center of the input range. Specifically, assume C_1 through C_{30} and C_{32} are one and others including C_{31} are zero, then $E_5 = 1,\ E_4 = 1,\ E_3 = 1,\ F_7 = 1$, and $F_i = 0$ for $i = 1, \ldots, 6$. The output binary code "000001" is close to the true value.

There are two possible critical timing paths in this FRSGCA encoder. One is the Gray-binary converter for D_3 in Eq. (A.3) and the other is the path from D'_3 to D_0 output in Fig. A.3. These paths are restricted in area and the power consumption occupies only a fraction of the total power. However they can be the dominant bottleneck of the method. We have to make the paths as fast as possible. The invention of fast low voltage OR and XOR gates adopted in the encoder was a key to this ADC development.

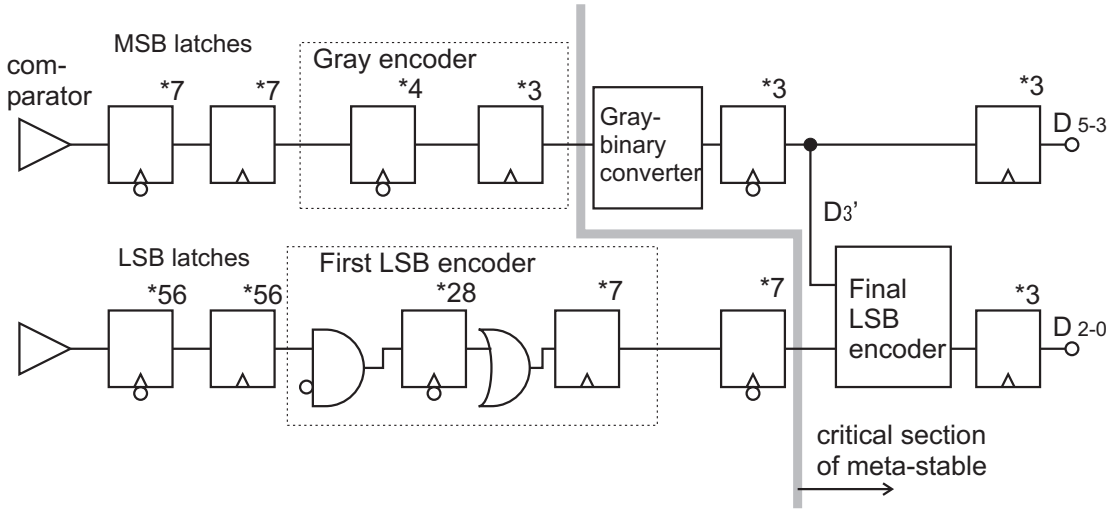

Figure A.3: Latch stages of the ADC: the Gray encoder implements Eq.(A.1), the First LSB encoder implements Eq.(A.2), the Gray-binary converter implements Eq.(A.3), and the Final LSB encoder implements Eq.(A.4).

A.3 Logic circuit for 1.8-V operation

There are various issues on low voltage operation of bipolar logic circuits. They are well discussed in [44]. Several ideas to cope with different situations are required to achieve the best total design.

The base-emitter voltage V_{BE} is the first thing to consider, since this is the parameter we cannot change easily. It is given as

$$V_{BE} \approx V_T \ln \frac{I_C}{I_S} \tag{A.5}$$

where $V_T = kT / q$, I_C is the collector current, and I_S is the saturation current. From its logarithmic nature, V_{BE} is rather constant over transistor parameters and operation currents. On the other hand, V_T and I_S heavily depend on the junction temperature, V_{BE} varies from 0.6 V at high temperature to 0.9 V at low temperature. At 1.8 V operation and at low temperature, a common circuit such as an emitter follower driving a differential pair leaves no headroom for the tail current source.

A simple substitute for the emitter follower is a level shifter utilizing a common impedance and a differential pair. Figure A.4 shows a practical embodiment. R_5, R_8 and R_9 in the figure provide this level shift. After sufficient level shift, various logic functions can be realized.

Preventing transistors from entering heavy saturation is another consideration for low voltage operation. The circuit in Fig. A.4 must be carefully designed from this point. The requirement for non-saturation is to keep

$$V_{CE} > V_{SAT} \tag{A.6}$$

where V_{CE} is the collector-emitter voltage and V_{SAT} is the saturation voltage around 0.2 V. For Q_3, this requirement becomes

$$V_{CC} - V_{R5} - V_{R3} > V_{CC} - V_{BE} + V_{SAT} \tag{A.7}$$

then

$$V_{BE} - V_{SAT} - V_{R3} > V_{R5} = V_{\text{shift}} \tag{A.8}$$

At high temperature, V_{BE} becomes 0.6 V and if the signal swing $V_{\text{swing}} = V_{R3}$ is around 0.2 V, V_{shift} is restricted to 0.2 V in one stage.

After two stages of level shifting, the operating level of the Q_7 and Q_8 pair is lower by 0.4 V than that of the Q_9 and Q_{10} pair. During transient operation, the level difference is diminished by the common emitter voltage swing of the Q_9 and Q_{10} pair, roughly one half of the V_{swing}, leaving little margin to keep Q_7 from saturating.

On the other hand at low temperature, Q_7 emitter level is as low as $V_{CC} - 2V_{\text{shift}} - V_{BE} - V\text{swing}/2 \approx 1.8 - 0.4 - 0.9 - 0.1 = 0.4$ V. This is the voltage left for the current source, which is smaller for bipolar transistors to operate stably. MOS transistors are appropriate for this purpose because we can design the drain source saturation voltage $V_{DS\,(\text{sat})}$ of MOS transistors as low as 0.4 V fairly easily.

As a result, assuming a BiCMOS process, the two-stage level shifter satisfies all requirements for 1.8 V operation with a narrow margin. Our ADC uses this topology extensively, especially for clock drivers of latches.

The drawbacks of the circuit in Fig. A.4 are 1) relatively low driving capability and 2) delay of the level shifter. Improvements to this basic topology were made so that large capacitive loads could be driven. The idea starts from a simple folding diode shown in Fig. A.5.

The emitter-follower-like transistor Q_1 can drive a load with a fairly large parasitic capacitance C_p, and the folding transistor Q_2 recovers the signal level. The circuit looks like a differential pair, but it is not because Q_2 does not switch. This folding driver more resembles an emitter follower and it is particularly suitable for long distance signal transfer. For example in our ADC, it is utilized for the transfer between latches from the first LSB encoder to the final LSB encoder in Fig. A.3.

Folding circuits usually show gain loss due to current changes in the folding transistor. If the gain loss is a problem, it is recovered by adding R_2 between the collector and the base of Q_2 as shown in Fig. A.6. Since R_2 does not affect Q_2 current, the swing is multiplied by $(1 + \frac{R_2}{R_1})$ times.

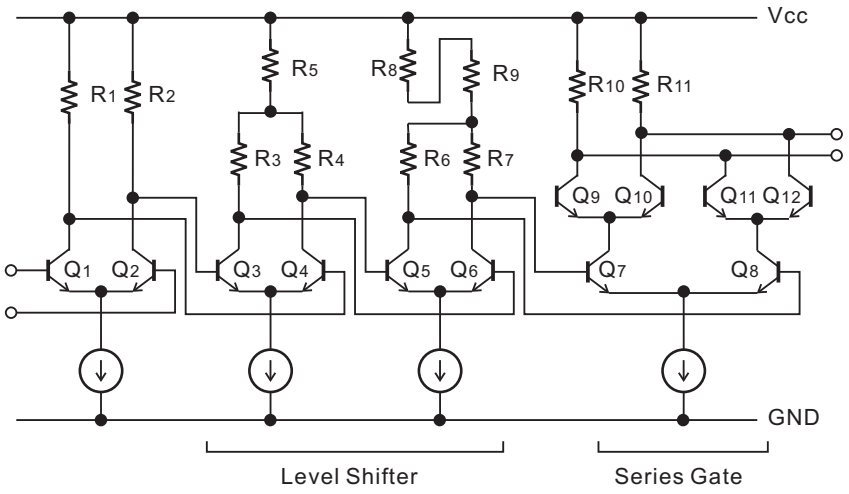

Figure A.4: Level shifting for a low voltage series gate

In order to implement various logic functions, we can combine the basic circuit in Fig. A.5 with wired-OR and collector-dot. In spite of both are very common to ECL design, so to our surprise, this idea gave birth to a general logic family, which we named "folding logic."

Figure A.7 shows the idea. In the figure, emitters of Q_1 and Q_2 form a WOR, as well as the Q_3 and Q_4 pair. Folding transistors Q_5 and Q_6 form a collector-dot. As a whole, the circuit realizes an AND-OR tree. Generally speaking, folding logic can realize a so-called standard product of sums [45] in one stage.

Folding logic looks like conventional Diode-Transistor Logic (DTL) especially at the collector-dot part. Actually while DTL belongs to a saturation logic family, no transistor in folding logic saturates, similar to ECL. This property makes the folding logic much faster than the DTL.

We usually adopt a complementary configuration of folding logic. Figure A.8 and Fig. A.9 show some practical implementations, an OR gate and an XOR gate respectively. This XOR circuit is extensively used in this ADC to realize Eq. (A.3) and Eq. (A.4).

The XOR operation of three terms that is required for D_3 generation in Eq. (A.3) can be realized in one stage applying a technique of the standard product of sums. Propagation delays are roughly 0.6 ns per differential pair and 0.4 ns per folding logic. The time taken for the three term XOR operation is 1.0 ns.

On the other hand, a XOR stage using the level shifter in Fig. A.4 requires at least three differential pair stages without counting the input buffer, two for level shift and one for series gating. The XOR operation of three terms cannot be implemented in one series gating with the 1.8 V power supply limit. It therefore requires two stages of XOR gating which will take 6 $\times$

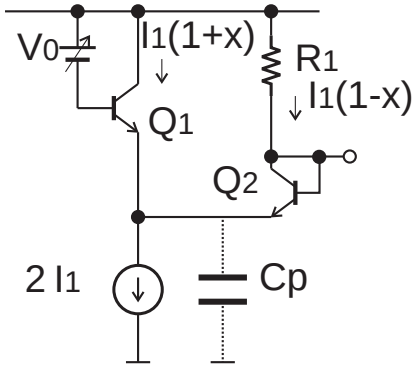

Figure A.5: Folding driver: the basic form of a folding logic

0.6 ns, or 3.6 ns, over three times slower than folding logic.

Since the output level of a folding logic gate is almost the same as the input level, we can directly connect some stages of folding logic without input buffers. This technique further improves the speed. We implemented the XOR-OR operation in Eq. (A.4) by this technique.

Using feedback, folding logic is even able to realize a latch. Figure A.10 shows an implementation. The minimum operating supply voltage can be as low as 1.5 V, i.e. V_{BE} + signal swing + headroom voltage for a tail current source. The limit is comparable to the lowest ever-reported in the reference [44]. Full complementary configuration provides more stable operation than the multi-level operation in [44]. However this example is shown only to demonstrate the ability of folding logic. We did not use the latch in this ADC since the circuit requires too many current sources and elements. As a contrast, a conventional series gating type latch we used [46] requires only one current source, while a level shifting gate for the clock driver is common to many latches.

A.4 An implementation

I show one implementation. The process adopted is a bipolar based BiCMOS process [47]. Table A.1 and Table A.2 show the main characteristics of this process. This process emphasizes the bipolar transistor characteristics at the expense of the CMOS transistor characteristics to provide a cost effective BiCMOS solution. Due to the relatively poor performance of the CMOS part, its major usage in the ADC is limited to DC operations like current sources.

The pair balance of bipolar transistors in the process can be 0.2 mV standard deviation only by stretching the emitter size moderately. This deviation is roughly one digit of magnitude better

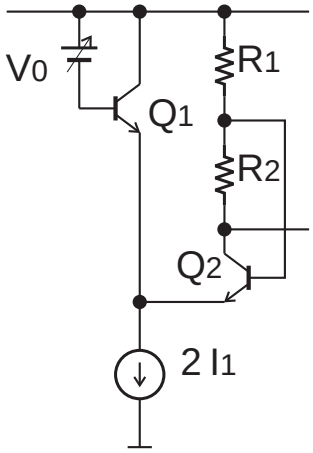

Figure A.6: Folding driver with swing recovery

than the ordinary MOS transistor pair offsets. We have chosen 0.7 μm by 3.1 μm for the emitter size.

This good pair balance allows a simple differential pair to satisfy all constraint required for the input comparator.

The full scale voltage $V_{RT} - V_{RB}$ is traditionally determined by the comparator offset. Ten times of the offset variation will be enough for one LSB, so that the full scale becomes 128 mV (= 0.2mV $\times$ 10 $\times$ 2^6) from this constraint. If $V_{RT} = V_{CC}$ = 1.8 V, then $V_{RB} < V_{RT} - 0.128$ V = 1.672 V, making room for the requirement $V_{RB} > 1.4$ V that we have discussed in section A.1.

On the other hand, bipolar comparators take input current which adversely affects the INL. The current depends heavily on various parameters like h_{fe} or Tj. Traditionally we have compensated for this effect by feeding currents to the resistor string that match the comparator input currents. However, the compensation circuit requires its own operation voltage V_{compen}. Roughly speaking, the condition is $V_{RT} < V_{CC} - V_{\text{compen}}$. There seems to be no circuit capable of making V_{compen} small enough.

Since the amplitude of INL warping by comparator input currents is independent of the full scale, wider dynamic range reduces relative INL. We have selected $V_{RT} = V_{CC}$ and $V_{RB} = V_{CC} - 0.3$ V from these considerations.

As an additional remedy, each reference point in the resistor string is positioned with an appropriate offset that compensates for the nominal input currents of comparators. Applying this technique, pitches between adjacent reference points becomes gradually smaller from V_{RB} to V_{RT}. This technique cannot cope with device variations but roughly halves the maximum INL.

Figure A.11 shows the chip micrograph. The area shown in the micrograph is 1.0 mm by

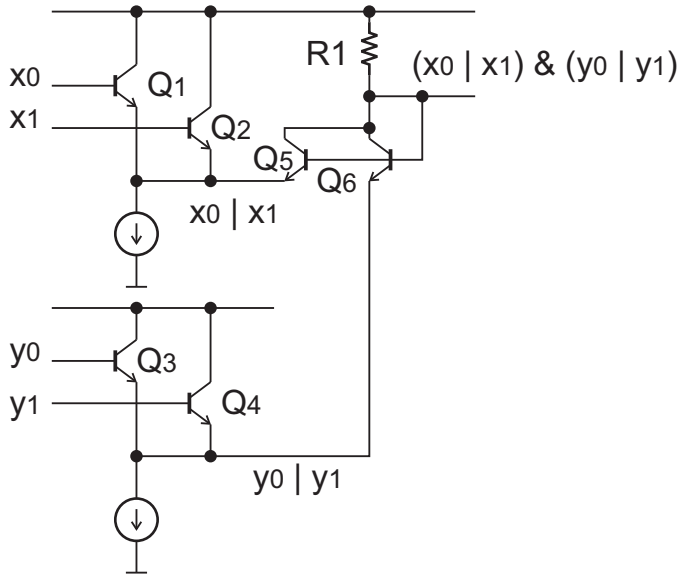

Figure A.7: General form of folding logic

1.6 mm. The layout is analogous to the drawing in Fig. A.2. The resistor string is the first metal seen as the slender, whitish rectangular area near the center. The outline of the folded resistor string is tapered, corresponding to the pitch reduction, and is visible at both upper and lower edges.

Eight clusters of comparators surround the resistor string. The first LSB encoders are also included in each cluster. The analog input comes from the upper area and the output goes from the right edge to the bottom.

A.5 Measured results

Figure A.12 shows the reconstructed waveform of a sinusoidal input to the BiCMOS ADC in the previous section. The clock frequency is about 330 MHz and the input frequency is about 82 MHz. We can identify dog-teeth in the figure. Figure A.13 shows the SINAD dependency on the input frequency, measured at the same clock and V_{CC} conditions with Fig. A.12.

Conversion error is measured for the other 6-bit ADC. The test setup is much the same with the distortion measurement in Fig. 1.15. No LPF is required. The input frequency satisfies $f_{in} = f_{clk}/12 + 10\text{kHz}$. This specific ADC internally decimates outputs by three so that we have four envelopes all of 10 kHz. The input amplitude is slightly over the ADC input range

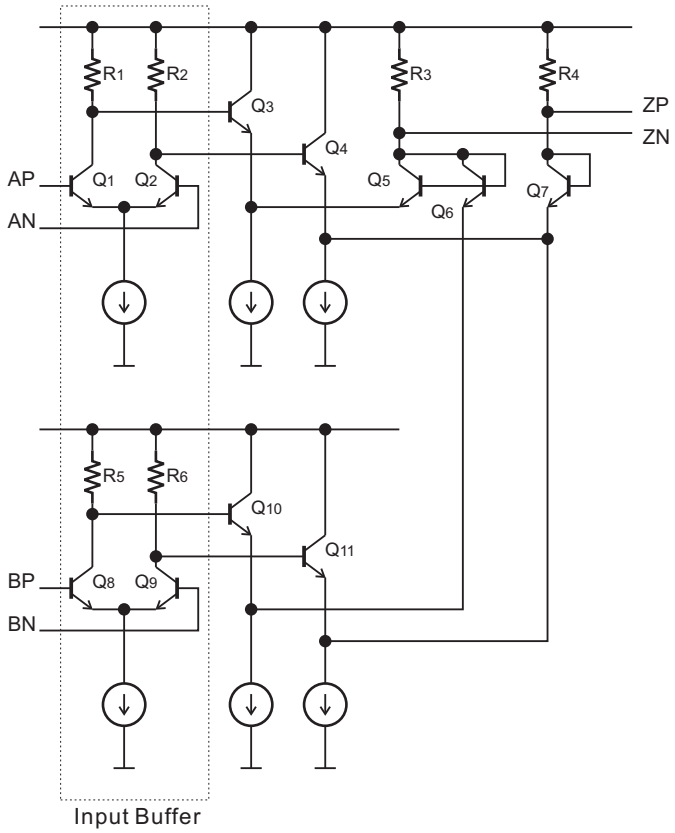

Figure A.8: A folding OR gate

as both the top and the bottom are clipped by the range and all comparators switch during the measurements. Since the clock frequency f_{clk} is high enough over 300 MHz in the measurement, every fourth data is very close that may change 1 LSB at most. We can detect sparkle error as the abnormal value than the expectation. In the following measurement, the average of total eight samples taken before and after at 4-step.

Figure A.14 shows the measured error rate. Since errors tend to be affected directly by the timing, the abscissa is the clock period instead of the clock frequency. The ordinate is a log-scale error rate. Errors are counted at two magnitudes, over 4 LSBs and over 10 LSBs. Error rates are roughly three digit separate in the measured conversion rate. Errors over 10 LSBs should stem from the MSB generator that the encoder design did not expected.

If the error rate changes exponentially to the clock period, the results should be on a line. Instead, they look S-shaped and seem divided into three regions. Mid-range is between 2.2 ns

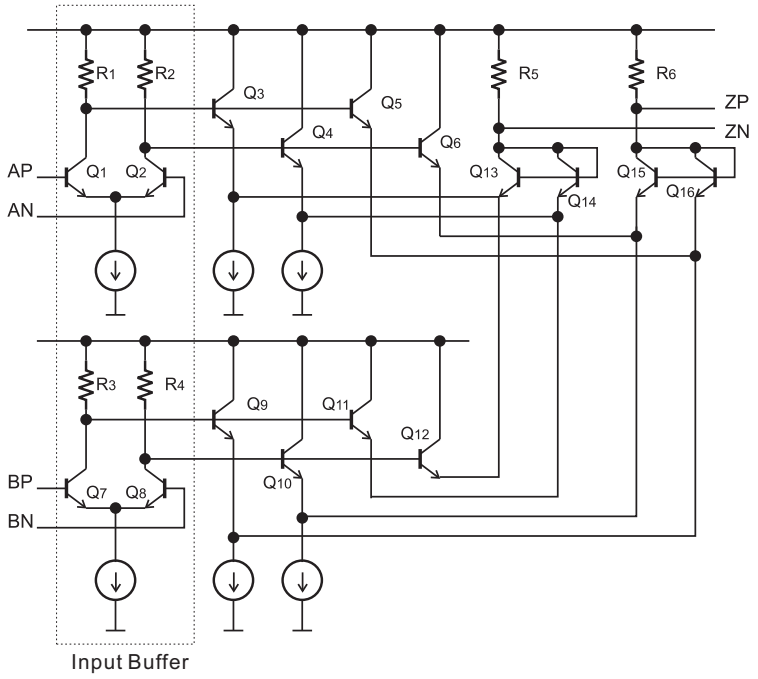

Figure A.9: A folding XOR gate

and 2.7 ns, and higher and lower ranges are in both sides. At shorter clock period, the output begins to collapse. On the other hand, when the clock period becomes longer, error rate rapidly goes too low to measure in reasonable time in our setup. At this region, it seems more appropriate to see the margin by clock frequency than by error rate.

Figure A.15 shows the error pattern at f_{clk} = 360 MSps (2.78 ns), using the same measurement data of Figure A.14. The abscissa is the expected value and the ordinate is the actual value. Two's complement code is used in the figure. The output code that differs more than 4 LSBs from the expected value is counted as an error. Sizes of circles are proportional to the counts.

Figure A.15 implies that MSB latches are associated with most of sparkle errors. To see the fact, we have measured the error rate at a small analog input. Results in Figure A.16 are obtained by the sinusoidal input with the average of the ADC range center and the amplitude of ±4 LSB. This means the input crosses only one MSB comparator (C_{32}) and that happens roughly one-tenth of the total sample count.

The total error rate in Figure A.16 looks almost independent of the input frequency. That is the input slew rate is irrelevant to the error rate. The value around 10^{-6} is consistent with the different input amplitude measurement in Figure A.15. As the MSB switching time equals to the

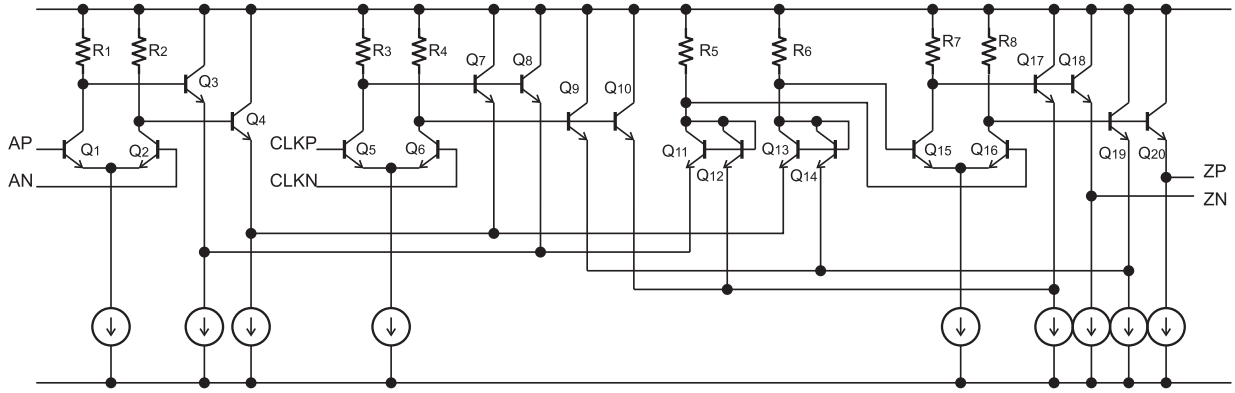

Figure A.10: A folding latch

count of the output code -1, dividing error count by -1 code count provides the malfunction rate of the MSB latch.

From the uniform design of comparator-latches, LSB latches should have the same malfunction rate. Then we can proclaim that intrinsic error rate of the ADC at 330 MSps is 10^{-5} but the encoder scheme reduces the error rate to 10^{-6}. However, if the input is small and it swings only around a MSB comparator, error rate jumps 10 times, while if the input swings avoiding MSB comparators, errors will disappear.

Table A.1: Characteristics of bipolar transistors

	NPN	PNP	unit
Emitter Size	0.2 x 0.6	1.0 x 1.0	μm^2
hfe	90	150	-
V_A	15	8.8	V
$f_T$max	25(V_{CE}=1V)	4.2(V_{CE}=-3V)	GHz
fmax	30(V_{CE}=1V)		GHz
BV_{CEO}	4.2	5.0	V
Cje	2.7	4.1	fF
Cjc	2.1	8.5	fF
Cjs	10	26	fF

Table A.2: Characteristics of MOS transistors

	NMOS	PMOS	unit
W/L	10/1.0	10/1.1	μm
Vth	0.5	0.5	V
Ids	2.2	1.0	mA
β	1100	370	$\mu A/V^2$
BVds	11.8	11.3	V

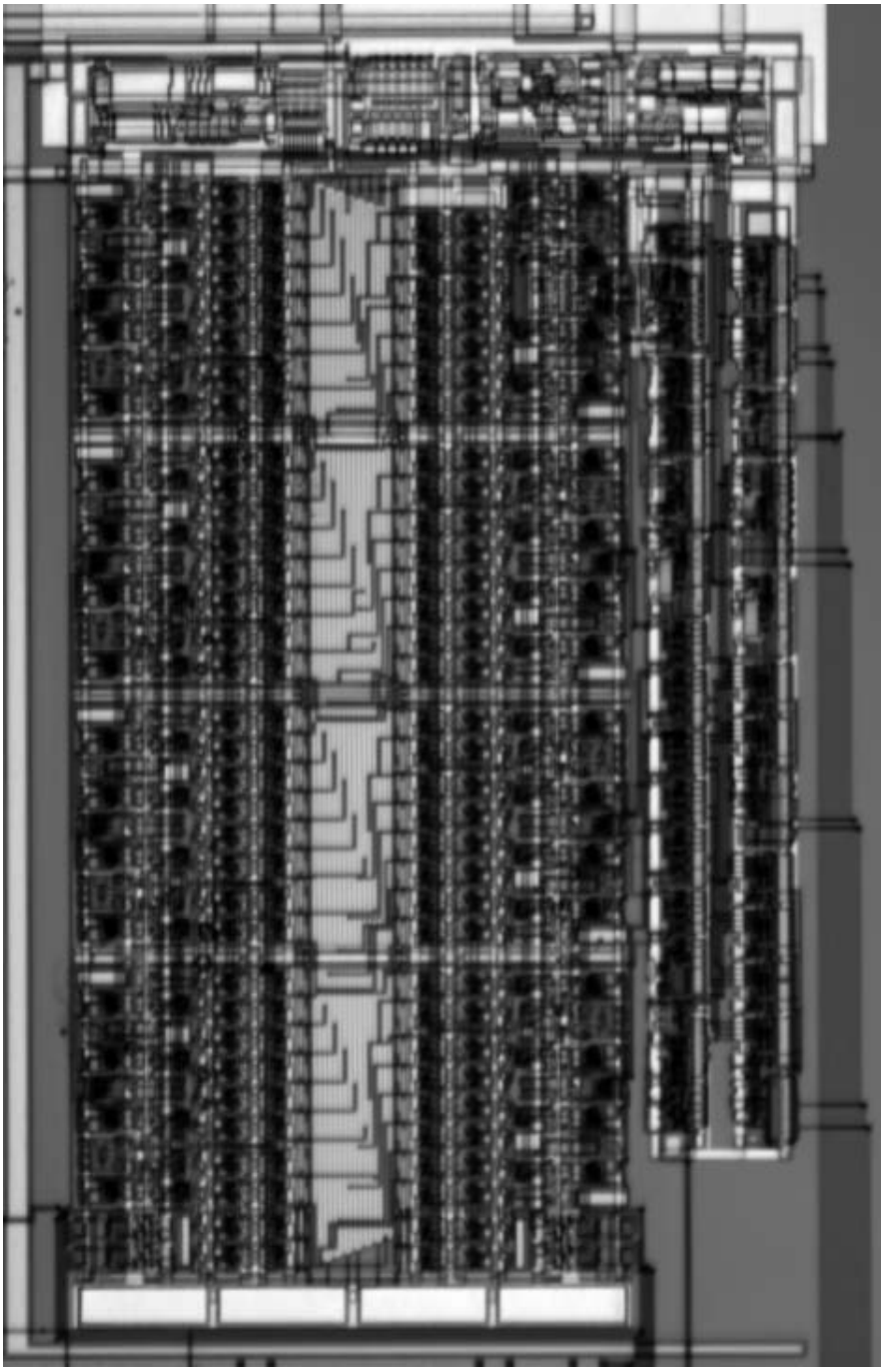

Figure A.11: Chip micrograph of the ADC: 1.0 mm by 1.6 mm

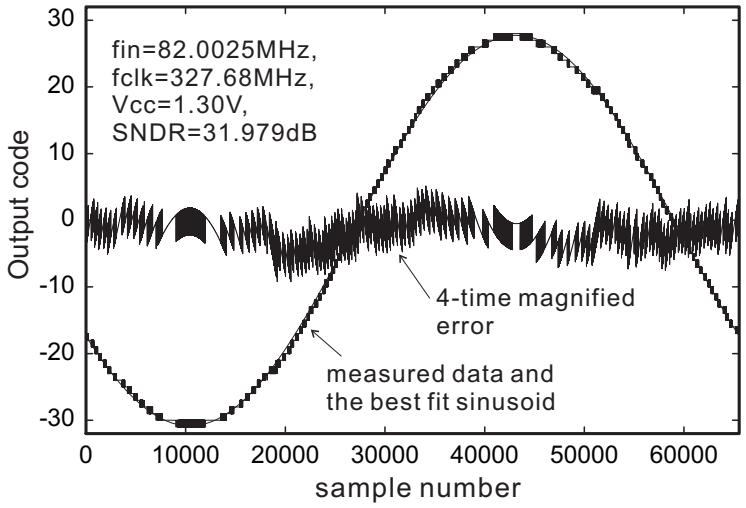

Figure A.12: Reconstructed waveform and the difference wave

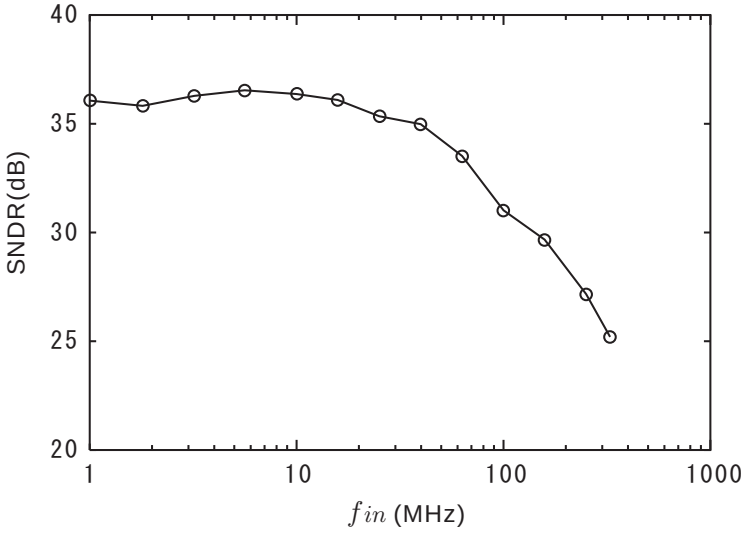

Figure A.13: Signal to Noise + Distortion Ratio vs. Input frequency: Clock frequencies 327.68MHz

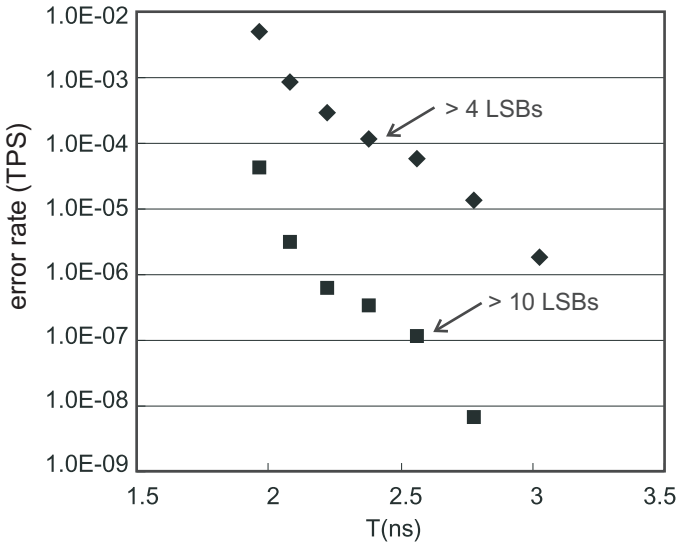

Figure A.14: Measured error rate vs. clock period

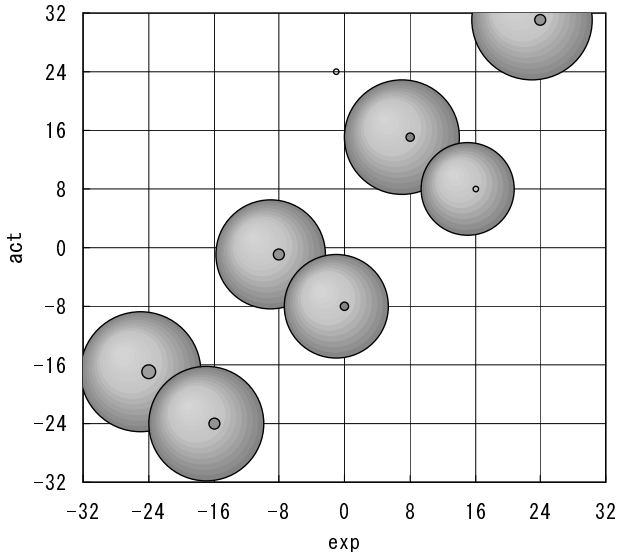

Figure A.15: Measured error pattern: f_{in} = 30.010 MHz and f_{clk} = 360 MHz

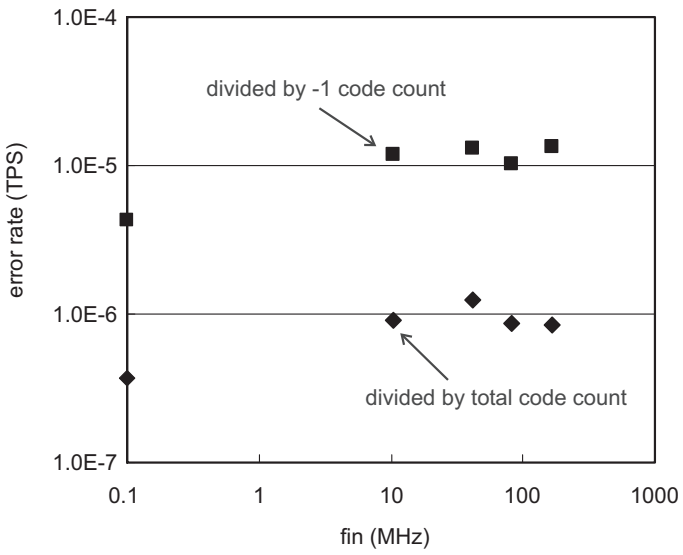

Figure A.16: Small signal error rate at $f_c = 330$ MSps. The V_{IN} swings ±4 LSB around the center of the ADC range. The code -1 count indicates how many times the center comparator is involved in the output code.

Appendix B

List of Published Papers

B.1 Journal Papers

- **Yuji Gendai**, “A 6-Bit 340 Msps BiCMOS ADC of 1.8 V single Power Supply Adopting Folding Logic,” *IEICE Transactions on Electronics*, Vol. E85-C, No. 8, pp. 1546–1553, Aug. 2002.

- **Yuji Gendai**, “The maximum-likelihood noise magnitude estimation in ADC linearity measurements,” *IEEE Transactions on Instrumentation and Measurement*, Vol. 59, No. 7, pp. 1746–1754, Jul. 2010.

- **Yuji Gendai**, Akira Matsuzawa, “A Specific Distortion Pattern of Flash ADCs Identified by Discriminating Time Domain Analysis,” *IEEE Transactions on Instrumentation and Measurement*, Vol. 61, No. 2, pp. 316–325, Feb. 2012.

B.2 International Conference and Workshop

- **Yuji Gendai**, Yoshihiro Komatsu, Shinya Hirase, Masoto Kawata, “An 8b 500MHz ADC,” *ISSCC Digest of Technical Papers*, pp. 172–173, 311, Feb. 1991.

B.3 Domestic Conferences

- **源代裕治**, 小松禎浩, 平瀬紳也, 川田政人, “8b 500MHz A/D 変換器,” 信学技報 ICD91-85, Vol. 91, No. 192, pp. 29–35, Aug. 1992.

- **源代裕治**, “ADC を例にした入力容量測定の S11 パラメータ法の提案,” 電子情報通信学会総合大会, C-622, Mar. 1995.
- **源代裕治**, “A/D コンバータのアパーチャジッタ測定の新手法の提案,” 電子情報通信学会エレクトロニクスソサエティ大会, C-487, Sep. 1995.
- **源代裕治**, “A/D コンバータの微分非直線性の要因分析手法,” 電子情報通信学会総合大会, C-584, Mar. 1996.
- **源代裕治**, “ADC リニアリティ測定のランプ法,” 電子情報通信学会総合大会, C-12-45, Mar. 1998.
- **源代裕治**, “ADC のスパークルエラーパターンの測定法と測定例,” 電子情報通信学会エレクトロニクスソサエティ大会, C-12-14, p. 75, Sep. 2006.
- **源代裕治**, “Verilog-A をテストベンチに用いた AD コンバータのサンプリングディレイ入力依存性検証法,” 電気学会電子回路研究会, Vol. ECT-07-46〜58, No. ECT-07-51, pp. 25–30, Jun. 2007.
- **源代裕治**, 松澤昭, “ランプ信号を用いた比較器雑音計測,” 電子情報通信学会総合大会, C-12-27, Mar. 2011.
- **源代裕治**, 松澤昭, “比較器入力換算ノイズの単一ランプによる測定手法,” 電気学会電子回路研究会, Vol. ECT-11-035〜046, No. ECT-11-038, pp. 17–22, Mar. 2011.

B.4 Patents

- Yoshihiro Komatsu, **Yuji Gendai**, “Full flash analog-to-digital converter, ” US patent 5119908, Jun. 2, 1992.
- Yoshihiro Komatsu, **Yuji Gendai**, “Collector dot and circuit with latched comparator,” US patent 5170079, Dec. 8, 1992.
- **Yuji Gendai**, “Analog to digital converter,” US patent 5548287, Aug. 20, 1996.
- **Yuji Gendai**, “Flash type analog-to-digital converter,” US patent 6480135, Nov. 12, 2002.
- **Yuji Gendai**, “Logic circuit,” US patent 6492842, Dec. 10, 2002.
- **Yuji Gendai**, “Comparator circuit,” US patent 6710733, Mar. 23, 2004.

- **源代裕治**, “並列型Ａ－Ｄ変換器,” 特許公報 第 2844806 号, 平成 10 年 (1998)10 月 30 日.
- **源代裕治**, “並列型Ａ－Ｄ変換器,” 特許公報 第 2844819 号, 平成 10 年 (1998)10 月 30 日.
- **源代裕治**, “並列型Ａ－Ｄ変換器,” 特許公報 第 2844830 号, 平成 10 年 (1998)10 月 30 日.
- **源代裕治**, “論理回路,” 特許公報 第 2990785 号, 平成 10 年 (1999)10 月 15 日.
- **源代裕治**, “比較回路,” 特許公報 第 3867849 号, 平成 18 年 (2006)10 月 20 日.
- **源代裕治**, “アナログ／ディジタル変換回路測定装置,” 特許公報 第 4022978 号, 平成 19 年 (2007)10 月 12 日.