MASTER

A pre-correction method for improved static linearity using parallel DACs

van den Hoven, R.A.T

Award date:
2006

Link to publication
A pre-correction method for improved static linearity using parallel DACs

R.A.T. van den Hoven

March 20, 2006

Master of Science thesis

Project period: April 2005 – March 2006

Department of Electrical Engineering
Capacity group: Information and Communication Systems
Chair: Mixed-signal Microelectronics

Supervisors:
Ir. G.I. Radulov
Dr. Ir. J.A. Hegt
Prof. Dr. Ir. A.H.M. van Roermund

The Department of Electrical Engineering of the Eindhoven University of Technology accepts no responsibility of the contents of M.Sc. theses or practical training reports
Abstract

This thesis presents a fully integrated self-pre-correction method for current-steering Digital to Analog Converters (DACs). The method improves the static performance. It is based on a novel flexible architecture that uses parallel sub-DACs.

The self-pre-correction method uses a self-measurement to determine the INLs of the sub-DACs. It is not possible to measure the INLs directly because this requires an ideal reference DAC. Therefore, the DNLs are measured. The INLs are constructed through the accumulated sum of the DNLs. Knowing the non-linearity of the sub-DACs, an optimal combination for the digital input-word can be found that minimizes the DAC non-linearity errors. The optimal combination of the sub-DACs is stored for every code in a look-up table as a pre-processing block. The algorithm is implemented as a Finite State Machine (FSM), so it can be implemented on-chip in future works.

This thesis contains two main parts. The first main part starts with the introduction of D/A conversion specifications. A parallel DAC architecture with flexibility of the performance is introduced. The parallel DAC architecture is designed with fixed building blocks. These building blocks can be controlled and connected in such a way that they form one or multiple DACs. The combinations change the properties of the D/A conversion. Therefore, one D/A converter-chip can be used for a broad range of applications.

The second main part presents a method to improve the static linearity. The parallel DAC architecture of the first part is used. The linearity can be improved by reducing the effect of the device mismatch errors. The input-code is distributed in an optimized way to the multiple sub-DACs, so that the errors due to transistor mismatch is reduced. All required extra resources are discussed in detail. Transistor level simulations of the most critical parts are performed. A design with four 10-bit DACs with built-in self-measurement and self-correction is presented. Also the on-chip implementation of the parallel architecture in a FPGA is discussed.
Contents

1 Introduction .............................................................................................................. 9
  1.1 Project Background ......................................................................................... 9
  1.2 Project Goal ..................................................................................................... 9
  1.3 Overview of the Chapters ............................................................................... 11

2 Conventional current-steering D/A Converters ...................................................... 13
  2.1 Basic principles .............................................................................................. 13
  2.2 Static Performance ......................................................................................... 13
    2.2.1 Transfer characteristic ............................................................................ 13
    2.2.2 Mismatch error ....................................................................................... 16
  2.3 Dynamic Performance .................................................................................... 17
  2.4 DAC architectures .......................................................................................... 19
    2.4.1 Binary DAC architecture ....................................................................... 19
    2.4.2 Thermometer DAC architecture ............................................................. 21
    2.4.3 Segmented DAC architecture ................................................................. 22
    2.4.4 Area resources ....................................................................................... 23
  2.5 Brief literature investigation ........................................................................... 25
  2.6 Generalized mapping ...................................................................................... 26

3 Parallel DAC Architecture and flexibility ............................................................ 27
  3.1 Operation modes ............................................................................................ 28
    3.1.1 Multiple independent converters ............................................................ 28
    3.1.2 Resolution improvement ......................................................................... 28
    3.1.3 Linearity improvement ........................................................................... 28
    3.1.4 Other operation modes ........................................................................... 29

4 Linearity improvement .......................................................................................... 31
  4.1 Transfer characteristics of the sub-DACs ....................................................... 31
  4.2 Finding the optimal combinations ................................................................... 34
  4.3 Finding an optimal combination with higher resolutions ............................... 42
  4.4 Dynamic performance due to static linearity ................................................ 47

5 Self-measurement and self-correction .................................................................. 49
  5.1 Ideal self-measurement ................................................................................... 49
  5.2 Self-correction of the measurement errors .................................................... 56
    5.2.1 Measurement error by 1 LSB current source .......................................... 56
    5.2.2 Measurement error by quantization and non-linearity ............................. 56
    5.2.3 Averaging multiple measurements .......................................................... 58

6 Current scaling .................................................................................................... 61
  6.1 Simulation current scaling .............................................................................. 62
  6.2 Scaling effect in the measurement .................................................................. 65
  6.3 Conclusions ..................................................................................................... 67
List of abbreviations

DAC  Digital to Analog Converter
DNL  Differential Non-Linearity
FPGA Field Programmable Gate Array
INL  Integral Non-Linearity
LSB  Least Significant Bit
MSB  Most Significant Bit
SFDR Spurious Free Dynamic Range
SN(D)R Signal-to-Noise-(and Distortion)-Ratio
a.o. amongst others
e.g. exempli gratia, “for example”

List of symbols

\( C \) Number of Combinations
\( F \) Full scale \( (2^N) \)
\( I_j \) Output current of source \( j \)
\( k \) Input code (of one sub-DAC)
\( k_T \) Input code of the total digital range
\( L \) Length of the transistor channel
\( N \) Resolution of the D/A converter
\( M \) Number of sub-DACs
\( u \) Unit-element
\( V_{ox} \) Offset voltage
\( V_{Th} \) Threshold voltage
\( W \) Width for the transistor channel
\( \sigma_u \) Standard deviation of random variable \( u \)
1 Introduction

This chapter introduces the master project. The chapter gives background, defines the project goal and gives an overview of the thesis.

1.1 Project Background

The majority of the modern technologies like telecommunication, computers and video processing applications are digital. Examples are: digital video encoding, 256 QAM transmissions, closed-loop control systems, etc. Digital to Analog Converters (DACs) convert the digital signals into analog signals. Each application has its own purpose and specifications for the DAC, e.g.

- Resolution,
- Accuracy,
- Speed,
- Power,
- Chip area.

These are important properties of the DACs. Nowadays, it is necessary to use different DACs for applications with different properties.

1.2 Project Goal

The aim of the project is to introduce flexibility in the D/A conversion. As a result of this the end-user can use one DAC for a broad range of applications. The basic idea behind flexibility is to design a DAC architecture with fixed building blocks. These building blocks can be controlled and connected in such a way that they form one or multiple DACs. The combinations change the properties of the D/A conversion.

![Parallel DAC Architecture](image-url)

Figure 1.1 Parallel DAC Architecture
The main parts of these fixed building blocks are separate DACs (sub-DACs) implemented in one chip. This architecture with multiple sub-DACs in one chip we call a parallel DAC architecture. Figure 1.1 shows the architecture with multiple sub-DACs and an extra digital control part. The end user can select the operation mode. The control-part receives the digital input code and the operation-mode. The operation mode selects the way that the control-part distributes the code to the sub-DACs. The outputs of the sub-DACs working together are connected off-chip.

One of the operation modes to investigate is a combination of some sub-DACs to improve the overall static performance. In this way a higher resolution and improved linearity is obtained.

Improved linearity can be achieved by distributing the digital input code to the sub-DACs in an optimized way. This distribution of the input code over the sub-DACs causes a reduction of the deviation (caused by the mismatch errors) in the combined output [Doris 2004, Deveugele 2004].

An example of a transfer characteristic of a DAC is shown in Figure 1.2. The actual transfer curve is the digital input transferred to an analog output. This curve deviates from the ideal transfer curve due to the non-linearity [Jespers 2001].

![Figure 1.2 D/A transfer characteristic](image)

The output current becomes larger when the sub-DACs work together than when used separately. The output currents have to be decreased when an equal maximal output current is required for each operation mode. The output currents have to be decreased by the same factor as the number of used sub-DACs (see Chapter 6).

This novel approach is especially interesting when it is combined with Field-Programmable Gate Arrays (FPGA). These FPGAs are especially popular for

---

*Figure 1.2 D/A transfer characteristic*
prototyping integrated circuit designs [Gulak 1995]. For different applications they require different DACs that are connected to their outputs. With the novel approach the DACs can be integrated on the same FPGA-chip because their performance can be adapted to the required specifications. The FPGA also has digital resources that can be used for the digital control of the D/A converters [Xilinx-1]. Therefore, it only needs a limited amount of extra resources for the flexible parallel DAC architecture and can be integrated on the same chip. Figure 1.3 shows an example of a conventional FPGA that requires external DACs and an FPGA with the internal parallel DACs. Another advantage is that less I/O ports are used for the DACs.

![Figure 1.3 FPGA plus DACs and a novel FPGA chip](image)

### 1.3 Overview of the Chapters

In the next chapter the basic principles and parameters of the D/A converter are discussed. Chapter 3 discusses the principle of the parallel DAC architecture and several operation modes. Chapter 4 explains the theory of the novel approach with the linearity improvement method. Chapter 5 presents the self-measurement method, followed by Chapter 6 with the current scaling used with the self-measurement. Chapter 7 presents the transistor simulations. Chapter 8 presents the extra resources required for the linearity improvement. Finally the conclusions and recommendations are presented.
2 Conventional current-steering D/A Converters

This chapter presents the basic principles, performance and architectures of Digital to Analog Conversion.

2.1 Basic principles

Current-steering DACs are systems composed of three main parts [Deveugele 2006]. These parts are shown in Figure 2.1 as the

- Digital control,
- Analog unit-elements,
- Mixed-signal switches.

The digital input word is processed in the control part. This controls the switches in the mixed signal part. The analog part contains the unit-elements that are combined to form the current-source. This is the static part of the DAC. In the mixed signal part the switches are connected to the unit-elements. This is the dynamic part of the DAC. The switches switch the current sources to construct the analog output.

![Figure 2.1 A high level current-steering DAC architecture](image)

2.2 Static Performance

The static performance describes the quality of the D/A conversion for static signals.

2.2.1 Transfer characteristic

The DAC converts the digital codes into analog values. Non-linearity should be avoided because it generates harmonic distortions, accuracy errors, etc. Each step in the digital code gives a related step in the analog value. In the ideal case, all current...
steps are the same size and the transfer curve is linear [Jespers 2001]. The ideal current step is

\[ I_{\text{LSB}} = \frac{I_{\text{out}}[2^N - 1] - I_{\text{out}}[0]}{2^N - 1} \]  

(Eq 2.1)

where \( I_{\text{LSB}} \) is the ideal/average current difference with one code step, \( I_{\text{out}}[0] \) is the analog output value at code 0, \( I_{\text{out}}[2^N - 1] \) is the analog output value at full-range, \( N \) is the number of digital input bits.

In current-steering DACs there are multiple unit-elements that are designed nominally identical. However in reality they are different and this mismatch error depends mainly on the area of the current source transistor. When the sizes of the current sources increase, the mismatch errors decrease [Pelgrom 89, Bastos 1998].

This mismatch error in the current sources causes the steps to be different. The Differential Non-Linearity (DNL) is the deviation between the ideal and the actual step and is expressed in LSB. The DNL\(_k\) is the step (DNL) error between two following codes

\[ DNL_k = \frac{I_{\text{out}}[k-1] - I_{\text{out}}[k]}{I_{\text{LSB}}} - 1 \]  

(Eq 2.2)

where \( k \) being the code \((k \in 1, ..., 2^N - 1)\), \( "I_{\text{out}}[k-1] - I_{\text{out}}[k]" \) is the actual analog current step and \( I_{\text{LSB}} \) the ideal current step.

The transfer characteristic is monotonic when the analog output increases with every step of the digital input. When the DNL\(_k\) is negative and larger than 1 LSB the transfer characteristic is non-monotonic. For example, this may cause serious problems when a converter is used in a closed feedback loop [Jespers 2001].

![Figure 2.2 Digital to Analog transfer characteristic](image)
Since there are mismatch errors, the analog output at a code $k$ can be higher or lower than the ideal output value. Figure 2.2 shows an example of an actual and an ideal curve. The ideal transfer curve is a straight line between the minimal and the maximal output value. The difference between the actual and ideal curve is the Integral Non-Linearity (INL). Therefore, the INL is zero at the beginning and at the end of the digital code range. $INL_k$ is the difference at code $k$ and is the accumulation of the $DNL_k$ errors. The $INL_k$ definition is

$$INL_k = \frac{I_{out}[k] - I_{out\_ideal}[k]}{I_{LSB}}$$

(Eq 2.3)

$$INL_k = \sum_{j=1}^{k} DNL_j$$

, where $I_{out}[k]$ is the actual and $I_{out\_ideal}[k]$ is the ideal analog value at code $k$.

The INL and DNL are expressed in LSBs. The INL and DNL without the transfer curve give better information about the non-linearity. Figure 2.3 shows the DNL at each digital input code.

![Figure 2.3 DNL of the transfer characteristic](image)

Figure 2.4 shows the INL at each digital input code. The values are discrete. However, a line drawn between the values gives a better view of the DNL and INL error behaviour.

![Figure 2.4 INL of the transfer characteristic](image)

In reality offset and gain errors can occur [Jespers 2001]. However they are linear errors, therefore by definition, they must be excluded from the non-linear errors. An
offset occurs when the ideal transfer curve shifts up or down and the gain error is a change of the angle of the ideal transfer curve. However, they do not exist when Equation 2.1 is applied for each curve. This is because the ideal/average step (LSB) is calculated from the maximal input and output range. This will compensate the offset and gain error.

### 2.2.2 Mismatch error

The mismatch error of a unit-element is a time-independent variation from the average of the identically designed unit-elements. The unit-elements of current steering DACs can be presented as the sum of the designed current and an error current.

$$I_u = \bar{I} + \delta I$$  \hspace{1cm} (Eq 2.4)

where $I_u$ is the current of the unit-element, $\bar{I}$ the designed current and $\delta I$ is the error current due to the mismatch [Radulov 2004].

The mismatch error that causes the DNL and INL can be modelled as a process with a normal probability density function. [Bosch 2004]. The integral of the function $p(x)$ is equal to one,

$$\int_{-\infty}^{\infty} p(x) \, dx = 1$$  \hspace{1cm} (Eq 2.5)

where $p(x)$ is the normal distribution of the unit elements.

![Normal distribution of the unit elements](image)

The normal distribution and standard deviation $\sigma$ of the unit elements are show in Figure 2.5. The probability density function is mathematically defined by

$$p(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-(x-\mu)^2 / 2\sigma^2}$$  \hspace{1cm} (Eq 2.6)

where $\mu=\bar{I}$ the expected value, $x$ a particular value and $\sigma$ is the standard deviation.
The INL is zero at the beginning and the end of the transfer curve (see Equation 2.1 and Figure 2.5) and has a symmetrical behaviour. Therefore, the highest probability of maximal INL error of a DAC can be expected at half-scale, [Bastos 1998, Bosch 2001]. With an $N$-bit converter this is at code $2^{N-1}$. The variance of the DAC at half-scale is the sum of the variances ($\sigma^2_u$) of the combined unit-elements.

$$\sigma^2_{\text{midscale}} = (2^{N-1} - 1) \cdot \sigma^2_u$$  \hspace{1cm} (Eq 2.7)

, with $N$ being the DAC resolution.

Note that this theory is not exactly true. When most mismatch errors are positive or negative then the average changes (caused by gain error). This is a linear error and therefore, the mismatch error $\delta$ becomes smaller. However it is a formula that is easy to use and because of this we use it as a representation of the mismatch error.

The variance of the DAC at half-scale is the sum of the variances ($\sigma^2_u$) of the combined unit-elements.

The standard deviation of the $(2^{N-1}-1)$ unit-elements is

$$\sigma_{\text{midscale}} = \sqrt{2^{N-1} - 1} \cdot \sigma_u$$  \hspace{1cm} (Eq 2.8)

Normalizing this to 1 LSB = $I_u$

$$\frac{\sigma_{\text{midscale}}}{I_u} = \sqrt{2^{N-1} - 1} \cdot \frac{\sigma_u}{I_u}$$  \hspace{1cm} (Eq 2.9)

At mid-scale we can model the random process with Equation 2.9 and

$$\mu = (2^{N-1} - 1) \cdot I_u$$  \hspace{1cm} (Eq 2.10)

The maximal mismatch error at midscale includes 99.73% of the errors at $3\sigma$.

$$\epsilon_{\text{midscale}} = 3 \sqrt{2^{N-1} - 1} \cdot \frac{\sigma_u}{I_u} \text{ LSB}$$  \hspace{1cm} (Eq 2.11)

, with $\epsilon_{\text{midscale}}$ as the maximal mismatch error at midscale.

$\epsilon_{\text{midscale}}$ can be seen as the representation of INL (and the gain error, see note Equation 2.7). This region with 99.73% of all mismatch errors is called the $3\sigma$ confidence level. When the area of the designed current source is increased, the $\sigma$ becomes smaller. This principle is used with intrinsically accurate current sources.

### 2.3 Dynamic Performance

The dynamic performance describes the quality of the D/A conversion for dynamic signals. This includes a.o.

- Settling times,
- Glitches,
- Charge feed-through,
- Quantization noise.
The Signal to Noise and Distortion Ratio (SNDR) shows how many times the power of the desired signal is stronger than the noise and harmonic distortion components. The SNDR is expressed in decibels:

\[ SNDR = 10 \cdot \log \frac{P_s}{P_n + \sum_{k=2}^{\infty} P_k} \]  

(Eq 2.12)

where \( P_s \) is the main signal power (fundamental), \( P_n \) is the noise power and \( P_k \) is the power of the \( k^{th} \) harmonic in the frequency band of interest.

Figure 2.6 shows an example of such a frequency spectrum. The fundamental represents the main signal, a converted sine-wave. The power of the fundamental is the reference level. The signal harmonics and the noise are shown. The horizontal axis shows the frequencies from DC to half the sample frequency (Nyquist) and the vertical axis shows the power of the signal in dB.

The relation between the power of the fundamental and the power of all quantization noise is the Signal to Noise Ratio (SNR) [Jespers 2001]. Some other types of noise like the power of the thermal noise is also in the SNR, however it is normally much smaller than the quantization noise. The formula of the SNR is

\[ SNR = 6.02N + 1.76 \]  

(Eq 2.13)

where \( N \) is the number of bits of the converter.

The Spurious Free Dynamic Range (SFDR) is the difference between the fundamental and the largest harmonic, shown in Figure 2.6. This is perhaps the most important dynamic specification.

\[ SFDR = 10 \log \frac{P_s}{P_{\text{max}}} \]  

(Eq 2.14)

where \( P_s \) is the power of the fundamental and \( P_{\text{max}} \) is the power of the largest harmonic.
2.4 DAC architectures

The digital input word is transformed by a DAC into an analog output value, shown in Figure 2.2. In theory the current-steering DAC has an array of nominally identically designed unit-elements. These unit-elements are grouped according to the weight of the digital input code. The number of unit-elements is almost equal to the number of digital input codes. The digital input-code is binary and the number of unit-elements is

$$N_{\text{elements}} = 2^N - 1$$  \hspace{1cm} (Eq 2.15)

with $N_{\text{elements}}$ being the number of unit-elements, $N$ being the resolution of the DAC. The minus one in the equation is because $2^N-1$ elements can create $2^N$ codes.

E.g. one unit-element can be switch on (1) or off (0) and can create two values.

2.4.1 Binary DAC architecture

For a single ended binary architecture the unit elements are grouped to $N$ current sources, shown in Figure 2.7. The $N$ switches can add the currents together to create the output current. The least significant input bit only switches one unit-element (shown in Figure 2.7, switch B0) and every next bit controls a group that is twice as large as the previous group. The current of group unit elements switched by bit $j$ is

$$I_j = 2^j \cdot I_{\text{unit}}$$  \hspace{1cm} (Eq 2.16)

with $I_{\text{unit}}$ being the current of one unit element.

The output value $I_{\text{out}}$ is the combination of the switched groups. The formula is

$$I_{\text{out}} = I_{\text{unit}} \cdot \sum_{j=0}^{N-1} 2^j \cdot B_j$$  \hspace{1cm} (Eq 2.17)

with $N$ being the number of input bits, $j$ the bit number and $B_j$ being the $j^{th}$ bit of the digital input word with value 1 or 0 (high or low).

The maximum output current is the current from all unit-elements together. The formula is

$$I_{\text{out, Max}} = I_{\text{unit}} \cdot \sum_{j=0}^{N-1} 2^j = I_{\text{unit}} \cdot (2^0 + \cdots + 2^{N-1}) = I_{\text{unit}} \cdot (2^N - 1)$$  \hspace{1cm} (Eq 2.18)
The majority of the current-steering DACs have a differential architecture, shown in Figure 2.8. The current from the unit elements can be switched to the positive or negative output. This means the positive output has a counterpart and the differential output current can be positive or negative. The outputs are differential currents, which minimizes even-order harmonics and common noise.

The Equations 2.16-18 remain the same. However, $B$ of Equation 2.17 can be 1 or -1.

The group of the Most Significant Bit (MSB) has $2^{N-1}$ of unit-elements. The remaining $2^{N-1}-1$ unit-elements are controlled by the other bits. This is the main disadvantage of the binary architecture because the unit-elements have mismatch errors. At mid-scale $2^{N-1}$ unit-elements are switched off and replaced with the $2^{N-1}$ remaining unit-elements. Therefore, statistically the maximal DNL error is expected at mid-scale [Bastos 1998]. The first $2^{N-1}-1$ unit-elements (switched off) have a variance of

$$\sigma^2_{2^{N-1}-1} = (2^{N-1} - 1) \cdot \sigma_u^2$$

(Eq 2.19)

The second $2^{N-1}$ unit-elements (switched on) have a variance of

$$\sigma^2_{2^{N-1}} = (2^{N-1}) \cdot \sigma_u^2$$

(Eq 2.20)

The variance and deviation at mid-scale are

$$\sigma^2_{\text{mid-scale}} = \sigma^2_{2^{N-1}-1} + \sigma^2_{2^{N-1}} = (2^{N-1} + 2^{N-1} - 1) \cdot \sigma_u^2$$

$$\sigma_{\text{mid-scale}} = (2^{N-1} - 1) \cdot \sigma_u$$

(Eq 2.21)
The $DNL_{\text{max}}$ with a $3\sigma$ confidence level is

$$DNL_{\text{max}} \equiv 3 \cdot \frac{\sigma_{\text{midscale}}}{I}$$

$$DNL_{\text{max}} \equiv 3\sqrt{2^N - 1} \cdot \left( \frac{\sigma_m}{I} \right) \text{ LSB}$$

(Eq 2.22)

The $DNL_{\text{max}}$ can become larger than 1, when the DNL is negative it causes non-monotonic behaviour.

An advantage of the binary architecture is that it requires only $N$ pairs of switches, drivers and interconnections for an $N$ bit input. Therefore, it requires a smaller silicon area for the digital and mixed-signal part, shown in Figure 1.1. A disadvantage of this architecture is that the mismatch errors are able to create a large $DNL_{\text{max}}$ and can cause non-monotonicity.

### 2.4.2 Thermometer DAC architecture

The thermometer architecture has a similar architecture, however every unit-element is controlled with a separate switch. Therefore, the largest DNL error has the same size as the largest mismatch error of the unit elements,

$$DNL_{\text{max}} \equiv 3 \cdot \left( \frac{\sigma_m}{I} \right) \text{ LSB}$$

(Eq 2.23)

Thermometer architectures only add unit-elements for each code and do not replace/swap currents. The transfer characteristic always has a monotonic behaviour. However, it requires a decoder to transform the binary code to a thermometer code and a switch for each unit-element, shown in Figure 2.9.

![Figure 2.9 Thermometer Architecture](image_url)

An advantage of the thermometer architecture is that it has small DNL errors. However a disadvantage is that it needs a switch pair and latches for each unit element. It also requires a binary to thermometer converter. Therefore, it requires a larger silicon area for the control and mixed signal part, shown in Figure 1.1.
2.4.3 Segmented DAC architecture

The majority of the DACs are implemented with a segmented architecture [O'Sullivan 2004, Hyde 2003], since it has the advantages of the binary and thermometer architecture. Segmentation is a way to divide the array of unit-elements into groups, in order to achieve specific advantages. The segmented DAC uses a thermometer code for the MSBs and binary for the LSBs, as shown in Figure 2.10. With this architecture the MSBs do not cause a large DNL error compared to a full binary architecture and it does not require switches for all unit-elements. This creates a balance between the accuracy of the thermometer code and the required silicon area. However it has a higher accuracy requirement for the mismatches than the thermometer architecture and it needs decoding logic. It also uses more silicon area as the pure binary architecture.

With an $N$-bit DAC, there are $B$ LSBs and $N-B$ MSBs. The largest DNL is expected where the $B$ LSBs are replaced by one large segment (one of the MSBs). The DNL equation would be the same as for Equation 2.22, however $N$ is replaced with $B+1$.

The segmented architecture is a compromise between the advantages (and disadvantages) of the binary and thermometer architecture.

There are also other ways for (pseudo) segmentation. The unit-elements are grouped to achieve a specific performance.
2.4.4 Area resources

An array of unit-elements can be used for different architectures. The number and size remain the same and only the manner they are grouped is different. However some architectures require more silicon area for the extra switches, latches, decoders and wires. The array of unit-elements is the analog-part. The others are in the digital and mixed-signal part of the DAC, shown in Figure 1.1.

Figure 2.11 shows an example of a micrograph of a DAC chip [Doris 2005]. The current sources in this design require a large chip area. A design with a method that reduces the effect of the mismatch errors, requires a smaller area for the current sources. However, it does require area for the extra resources, e.g. CalDACs, latches and switches, see Appendix C.

Figure 2.11 A micrograph of a DAC chip.
Figure 2.12 shows examples of a unit-elements array, used in different architectures. These architectures are

a) A thermometer architecture that uses the unit-elements separately,
b) A binary architecture uses the unit elements in groups, which increase by a factor two at each bit,
c) A conventional segmented architecture uses a grouping of the unit-elements like the binary one for the LSB bits and a thermometer one for the MSBs,
d) Novel approach that also uses a segmentation of the unit-elements. However, the segmentation is dividing them as multiple conventional sub architectures. We call this a parallel DAC architecture.

**Figure 2.12** Silicon area and grouping of the unit elements for
a) Thermometer architecture
b) Binary architecture
c) Conventional segmented architecture
d) Novel parallel approach.
2.5 Brief literature investigation

The presented work is focused on the design of a high (static linearity) performance current-steering D/A converter architecture. Other new technologies have already been investigated and designed by other authors. This section presents a selection of these papers about current-steering DACs.

The following papers are examples of state-of-the-art DACs with different techniques to reduce the non-linearity effects. The examples are representative for state-of-the-art chip designs.

The paper of [Chan 2006] describes a 14-bit DAC design, constructed with two binary 12-bit DACs. The sampling frequency is 100 Ms/s. The DAC is designed for a 0.18\textmu{}m, 1.8V CMOS process and occupies 3.18 mm$^2$ of active silicon area. The power consumption is 54 mW for the analog part and 96.3 mW for the digital part. The two DACs are used for Dynamic Element Matching (DEM) and not for flexibility. The pseudo random distribution of the input code to the sub-DACs does not require memory. Randomising the combined output code causes it to become uncorrelated. Thereby, it avoids harmonic distortion. The SFDR was improved from 52.4 dB to 76.2 dB.

The paper of [Deveugele 2006] describes a 10-bit DAC design with method for reducing the segmentation. The sampling rate is 250 MS/s The DAC is designed in 0.18\textmu{}m, 1.8V CMOS and the active silicon area is less than 0.35 mm$^2$. The power consumption is 22mW. The INL and DNL are below 0.1 LSB.

The paper of [Hyde 2003] describes a 14-bit DAC design with calibration and has a [5 9] segmentation. The sampling rate is 300 MS/s. The DAC is designed in 0.25 and 0.18\textmu{}m 1.8V CMOS logic processes and occupies 0.44 mm$^2$ of active silicon area. The power consumption is 53 mW. The calibration uses on-chip electrical trimming. The INL is 0.3 LSB and the DNL is 0.4 LSB.

The paper of [O'sullivan 2004] describes a 12-bit DAC design with [5LSB-7MSB] segmentation. The sampling rate is 320 MS/s. The DAC is designed in 0.18\textmu{}m, 1.8V CMOS processes and occupies 0.44 mm$^2$ of active silicon area. The power consumption is 82 mW. The INL is 0.4 LSB and the DNL is 0.3 LSB. The increased switching noise associated with a high degree of segmentation has been reduced by a new latch architecture.

The paper of [Schofield 2003] describes a 16-bit DAC design with segmentation and calibration. The segmentation is [7 MSB, 4 ISB and 5 LSB]. The LSB are binary, the ISB (Inter Mediate Bits) use 15 groups of 32 unit-elements and the MSB use 127 groups of 512 unit-elements. The sampling rate is 400 MS/s and the DAC is designed in 0.25\textmu{}m, 3.3/2.5V CMOS. The power consumption is 400 mW. The active area is 1.95mm$^2$ The INL is 0.7 LSB and DNL 0.3 LSB with 16 bit resolution.
Table 2.1 shows the key-features of the converters, such as resolution, INL/DNL, power and area.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Chan 2006</td>
<td>14</td>
<td>0.1</td>
<td>0.1</td>
<td>100</td>
<td>150.3</td>
<td>3.18</td>
</tr>
<tr>
<td>Deveugle 2006</td>
<td>10</td>
<td>0.3</td>
<td>0.4</td>
<td>250</td>
<td>22</td>
<td>0.35</td>
</tr>
<tr>
<td>Hyde 2003</td>
<td>14</td>
<td>0.4</td>
<td>0.3</td>
<td>300</td>
<td>53</td>
<td>0.44</td>
</tr>
<tr>
<td>O'sullivan 2004</td>
<td>12</td>
<td>0.7</td>
<td>0.3</td>
<td>320</td>
<td>82</td>
<td>0.44</td>
</tr>
<tr>
<td>Schofield 2003</td>
<td>16</td>
<td>0.4</td>
<td>0.3</td>
<td>400</td>
<td>400</td>
<td>1.95</td>
</tr>
</tbody>
</table>

Table 2.1 State-of-the-art DAC summary

2.6 Generalized mapping

A technique to reduce the effect of non-linearity is discussed in [Doris 2004]. The method uses generalized mapping of the unit elements. The DAC uses current sources with the same weight, like the thermometer architecture, shown in Figure 2.12a. The current sources can be controlled separately in the digital domain. Therefore, there is redundancy in combinations of current sources for each output. Information about the errors of each current source is required. Selecting the current-sources in a specific way (mapping) creates an output current with a minimal error.

The linearity improvement method of Chapter 4 has redundancy in the code distribution to the sub-DACs. This method uses the code combinations of the sub-DACs that provide an output with a minimal error. The method does not require a full control of each current source. The weights of the current sources do not have to be the same because it uses groups instead of separate current sources. Therefore, also other (sub) DAC architectures can be used. For example, the sub-DACs have a binary architecture, shown in Figure 2.12d.
3 Parallel DAC Architecture and flexibility

The structure and performance of the DAC architectures discussed in Chapter two can not be changed after the design and production. Since they can not be changed anymore, we call them fixed architectures. A novel approach would be a DAC that can change its performance, e.g. resolution, accuracy, speed, power, etc.

Our approach uses an architecture with multiple fixed sub-DACs, shown in Figure 3.1. The sub-DACs can be controlled and connected in parallel. Therefore, we call this a parallel architecture (Appendix D, [Hoven 2005]). The extra digital pre-processing controls the sub-DACs. The sub-DACs can be used in different operation modes. Hence, the resolution and accuracy are no longer fixed but flexible. Note that the flexibility in the digital part affects the whole DAC performance, without a need for major changes in the analog and mixed-signal parts.

![Figure 3.1 Parallel Architecture](image)

The outputs of the sub-DACs can be connected to each other at the outside of the chip. The required specification of the architecture is selected by the operation mode. The internal control-part distributes the digital input to the sub-DACs. The equally designed $M$ number of sub-DACs have an $N$-bit resolution.

Although the parallel architecture uses multiple (sub) DACs, it does not require more unit-elements than a conventional architecture, shown in Figure 2.11d.
3.1 Operation modes

The end-user can select an operation (op) mode for the required DAC specification. The three main goals of specific op-modes are

- Independent converters,
- Improving the static performance, such as
  - Resolution,
  - Linearity,
- Improving the dynamic performance.

3.1.1 Multiple independent converters

One of the op-modes is to use the sub-DACs as independent converters. The outputs of the DACs are not connected to each other. This op-mode allows the user to use multiple DACs at once for different applications. For example, to supply the (red-green-blue) outputs for video decoding. However the parameters of these DACs are not improved. Each sub-DAC requires an \( N \)-bit input.

3.1.2 Resolution improvement

The operation mode to improve the resolution, "stacks" the sub-DACs. The output currents of the sub-DACs are added together and since the multiple sub-DACs have more unit-elements, the resolution is increased. The architecture works as a pseudo-segmented architecture. The first sub-DAC converts the first \( 2^N \) codes and the second sub-DAC the codes from \( 2^N+1 \) to \( 2^{N+1} \), etc. The sub-DACs are stacked by the control-part. The output currents are added together. The DNL_{max} of the combined DAC has same size as the maximal DNL_{max} of the separate DACs. Therefore, the differential linearity is not improved. The resolution of the combined DAC, with \( M \) \( N \)-bit sub-DACs, is

\[
R = N + \log_2(M) \quad \text{(Eq 3.1)}
\]

with \( R \) being the resolution in bits, \( N \) being the resolution of each sub-DAC and \( M \) being the number of sub-DACs.

3.1.3 Linearity improvement

The operation mode to achieve a higher linearity uses a pre-correction method. The current-sources have deviations caused by the mismatch errors. When these are known, they can be used to reduce the non-linearity in the output. Improving the linearity can be achieved by distributing the digital input code to the sub-DACs in an optimized way. This method uses the deviations (mismatch errors) of each output to
reduce/cancel each other in the combined output current. When the mismatch errors are known, an algorithm can calculate the optimal combination of the sub-DACs for each input code. This combination of the sub-DAC is stored for each code. When the DAC receives an input code, the algorithm retrieves the combination from the memory and sends those to the sub-DACs. This op-mode also has a higher resolution because of the combination of sub-DACs.

The linearity improvement is the main operation mode of this thesis.

### 3.1.4 Other operation modes

There are other possible operation modes that are not explored in the master project. Examples of these op-modes are

- Improving the dynamic performance by
  - Cancelling harmonics, by phase-shifting the input for the sub-DACs [Mensink 2004],
  - Cancelling the image band, This by time interleaving the input for the sub-DACs [Deveugele 2003],
  - Reducing the effect of glitches [Deveugele 2006]
- Reducing the power consumption, by switching off un used digital resources and sub-DACs.
- Using multiple op-modes at the same time.

There are no restrictions for the number of sub-DACs in the parallel architecture. It is possible that the architecture operates with multiple op-modes at the same time. For example (M-3) sub-DACs are used for an improved linearity, two being a separate DAC and one switched off.

When the sub-DACs work together in a conventional way, the maximal output current is higher than when they are used separately. When it is required that the maximal output current is limited, then the output currents of each sub-DAC have to be reduced, see Chapter 6.
4 Linearity improvement

This chapter concentrates on the linearization operation mode. The proposed method reduces the negative effect of mismatch errors on the output value. This is achieved through distributing the input digital word among the sub-DACs in an optimized way.

The method improves the linearity by an explicit reordering of the mismatch errors. The unit-elements are grouped as a pseudo-segmentation into sub-DACs. Two groups of unit-elements with opposite mismatch errors are combined into one larger group. The combined mismatch error of this group is smaller than those of each separate group. The output errors of the sub-DACs are required to be known before they can be used to compensate each other.

4.1 Transfer characteristics of the sub-DACs

Before the linearity can be improved, the non-linearity of the sub-DAC transfer characteristics has to be known. Figure 4.1 shows an example of four sub-DAC transfer characteristics. These are the sub-DACs A, B, C, D and the ideal curve of a parallel DAC architecture. The sub-DACs will work together in this op-mode. Therefore, the ideal characteristic is drawn between the average of the minimal and the maximal output currents of the sub-DACs. The difference between the maximal outputs is caused by the mismatch errors. The differential transfer characteristic is shown in Appendix C.

The deviation from the ideal curve is shown in Figure 4.2. In Chapter 2, the gain error caused by the mismatch errors was removed from the INL, see Equation 2.1 and 2.3. Therefore, the INL was zero at the beginning and end of the scale. However, with this op-mode the sub-DACs work together and the gain is different for each sub-DAC. Therefore, it is a part of the non-linearity and will affect output deviation of the combined DACs.
To achieve a higher resolution, the sub-DAC outputs are stacked. Figure 4.3a shows the stacked transfer curves of Figure 4.1.
Stacking of the sub-DACs means only one sub-DAC is active for a part of the input code range. The sub-DAC A is active during the first $F$ codes, after code $F$ sub-
DAC.A stays fully on. Between code $F$ and $2F-1$ DAC B is active, etc. In the combined DAC are the transfer characteristics connected to each other. The INL of the combined DAC is shown in Figure 4.3b below the transfer characteristic.

![Figure 4.3 Combined transfer characteristic (a) and INL (b)](image)

The code $F$ stands for the “Full-scale” of one sub-DAC ($F=2^N$). On the horizontal axis are the digital input codes, the range goes from 1 to $4F-3$.

In the example of Figure 4.3, at code $2F$ the first two sub-DAC (A and B) are fully used (maximal output) and one output-code from sub-DAC C. The INL$_{2F}$ is 2.2 LSB. However to improve the linearity a better combination can be found. For example, with DAC A and D plus code one of sub-DAC B, the INL$_{2F}$ is only 0.1 LSB. The positive error of DAC A is compensated by the negative error of DAC D. This combination has to be stored in a memory and recalled when the input code is $2F$.

For each code an optimal combination has to be found and stored in the memory as a look-up table. When the DAC is used, it “looks” with each input-code in the memory to retrieve the codes for the sub-DACs.
4.2 Finding the optimal combinations

The optimal combination of the sub-DAC outputs construct a combined-output with the minimal error compared to the ideal output. The (combined) DAC is constructed with the outputs of the sub-DACs. With a single DAC there is only one output combination for each input code. When two DACs are used, the number of output combinations is increased. Except at the beginning and end of the code-range, where all sub-DACs are fully off or on. For example, two sub-DACs are used. The analog output value is the sum of the two sub-DAC outputs. The formulas for the input and output values are

\[ k_T = k_1 + k_2 \]

\[ I_{out_T}(k_T) = I_{out_{subDAC1}}(k_1) + I_{out_{subDAC2}}(k_2) \]  

(Eq 4.1)

, with the input code \( k_T \) being the sum of the sub-DACs codes \( k_1 \) and \( k_2 \) \((k_1, k_2 \in 0, \ldots, (2^N-1))\), \( I_{out_T} \) the output of the combined DAC and \( I_{out_{subDAC1}} \) and \( I_{out_{subDAC2}} \) the sub-DAC outputs.

Table 4.1 shows the combinations of \( k_1 \) and \( k_2 \) to construct the analog output corresponding to the digital code \( k_T \) with two parallel 8-bit sub-DACs.

<table>
<thead>
<tr>
<th>Input ((kT))</th>
<th>Combinations to distribute the codes</th>
<th>( C_{kT} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0+0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0+1, 1+0</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>0+2, 1+1, 2+0</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>0+3, 1+2, 2+1, 3+0</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>0+254, 1+253, ..., 253+1, 254+0</td>
<td>255</td>
</tr>
<tr>
<td></td>
<td>0+255, 1+254, ..., 254+1, 255+0</td>
<td>256</td>
</tr>
<tr>
<td>255</td>
<td>255+0</td>
<td>256</td>
</tr>
<tr>
<td>256</td>
<td>1+255, 2+254, ..., 254+2, 255+1</td>
<td>255</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>508</td>
<td>253+255, 254+254, 255+253</td>
<td>3</td>
</tr>
<tr>
<td>509</td>
<td>254+255, 255+254</td>
<td>2</td>
</tr>
<tr>
<td>510</td>
<td>255+255</td>
<td>1</td>
</tr>
<tr>
<td>511</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.1. Combinations with two 8-bit sub-DACs.

\( C_{kT} \) is the number of combinations in which way the codes can be distributed to the sub-DACs. Note that for every extra sub-DAC one input code will be lost. This is because every sub-DAC has \( 2^N \) codes and only \( 2^N-1 \) unit-elements. With two sub-DACs, \( 2 \times (2^N-1) \) unit-elements and it has only \( 2^{N+1} - 1 \) codes.

The number of input codes for the combined DAC with \( MN \)-bit sub-DACs is:

\[ Codes_{in} = M \times 2^N - (M - 1) \]  

(Eq 4.2)

, with \( Codes_{in} \) being the number of input codes, \( M \) the number of sub-DACs and \( N \) the resolution.
The linearity improvement method tries all combinations that would be correct without a mismatch error and checks if the errors reduce each other. See Equation 4.1. If it found an INL combination that is smaller than the stored INL\textsubscript{\(kT\)} it stores this value and the sub-DAC combination for code \(k_T\). When all these combinations are checked, the optimal combinations are found and stored. For each input code it can retrieve the sub-DAC combinations and construct the optimal output characteristic.

However, it is possible that the errors do not reduce but are added to obtain another valid output. For example, two output values have a positive mismatch:

\[
\begin{align*}
I_{\text{out}_{\text{subDAC1}}}(k_1) &= k_1 \cdot I_{\text{Isb}} + \Delta I_{k_1} \\
I_{\text{out}_{\text{subDAC2}}}(k_2) &= k_2 \cdot I_{\text{Isb}} + \Delta I_{k_2} \\
\end{align*}
\]

(Eq 4.3)

\(k_T = k_1 + k_2\)

\[
I_{\text{out}}(k_T) = (k_1 + k_2) \cdot I_{\text{Isb}} + \Delta I_{k_1} + \Delta I_{k_2}
\]

The algorithm checks if \(\Delta I_{k_1} + \Delta I_{k_2} \approx 0\) and if it is, the combination will improve the linearity. When \(\Delta I_1 + \Delta I_2 = I_{\text{Isb}}\) this would also be a good combination for \(k_T + 1\), however the algorithm does not check on this.

The number of sub-DACs affects the number of combinations (it does check) for each code. For example, when \(M\) DACs are used, code 1 can be constructed with \(M\) combinations.

![Figure 4.4 Combinations 2, 3 and 4 sub-DACs](image)
The number of combinations for each code, increases to half-scale and then decreases again. Figure 4.4 shows the number of combinations for each code with 2, 3 and 4 sub-DACs. In this example, the combined DAC has a 9-bit resolution and has 255 unit-elements. The unit-elements are divided by the $M$ sub-DACs.

With each extra sub-DAC the number of combinations increases. At midscale with two sub-DACs there are “only” 256 combinations, shown in Figure 4.4a and Table 4.1. With three sub-DACs (4.4b) there are over 20,000 and with four sub-DACs there are over 1.4 million combinations (4.4c) for each code.

The total number of possible combinations for all codes with $M$ sub-DACs is

$$C_M = \sum_{k=0}^{M} C_{kT} = (2^N)^M = 2^{M*N}$$

(Eq 4.4)

, with $C_M$ being the total number of combinations, $kT$ being a code on the total (combined) range, $C_{kT}$ the number of combinations at code $kT$, $M$ the number of sub-DACs and $N$ the resolution.

To construct the combined DAC transfer characteristic two sub-DACs have $2^{2N}$ combinations to choose from, for $2^{N+1.1}$ codes (see Equations 4.2 and 4.4). When for each code the combination with the best linearity is selected, the INL$_k$ (the INL error for code k) is equal or smaller than with a random/fixed combination. With each additional sub-DAC the number of combinations, and hence the chance to find a better one increases, shown in Figure 4.4b and 4.4c. The total number of combinations for 3 and 4 $N$-bit sub-DACs are $2^{3N}$ and $2^{4N}$ (Equation 4.3) and there are only $3*2^N-2$ and $4*2^N-3$ codes to construct (Equation 4.1).

Note that the INL improvement will be less at the ends of the transfer characteristic, because there the number of possible combinations is smaller. At half-scale there are the most possible combinations and hence the possibility to create one for a better linearity is higher.

The best combined transfer characteristic can be constructed by using the INL$_k$ of the sub-DACs to find the minimal INL$_{kT}$. The best overall INL will be constructed when all the minimal INL$_{kT}$s are found. To find the minimal INL$_{kT}$ all combinations of the sub-INL$_{k}$s must be evaluated. Every time it finds an INL$_{kT}$ that is smaller than the INL$_{kT}$ stored in the memory (at code $kT$), the stored INL$_{kT}$ will be replaced with the new INL$_{kT}$. For this code $kT$ also the sub-codes for the sub-DACs will be stored in the memory as a look-up-table. The distributed sub-codes are combined the digital input-code. For example with four sub-DACs

$$k_T = k_1 + k_2 + k_3 + k_4$$

(Eq 4.5)

, with $k_T$ being the digital input code and $k_1$, $k_2$, $k_3$ and $k_4$ being the input codes of the sub-DACs.
The INL\textsubscript{kT} of the combined code is the sum of the INLs and gain of the sub-DACs

\[
\text{INL}_{kT} = \text{INL}_{g_{k1}} + \text{INL}_{g_{k2}} + \text{INL}_{g_{k3}} + \text{INL}_{g_{k4}} \quad \text{(Eq 4.6)}
\]

, with \text{INL}_{kT} being the INL of the overall analog output and \text{INL}_{g_{k1-k4}} being the INL and the gain-error of the sub-DACs.

The different linearity with a fixed and flexible architecture are shown in Figures 4.5 to 4.8. To show the improvement, 700 Monte-Carlo simulations are drawn in one figure. For each simulation a DAC is constructed with \( M \) sub-DACs. In Figures 4.5 and 4.8, the first figure (a) has a fixed control for the sub-DACs, i.e. the distribution of the digital codes is not optimized for improved linearity. The sub-DACs are stacked. The simulations are created with three different sub-DAC architectures, the thermometer, binary and segmented (50% segmentation) architecture. Like in the normal DACs, the highest statistical possibility of locating \( \text{INL}_{\text{max}} \) is at mid-scale. The second figure (b) shows the INL of the same DAC but with full flexibility in the control-part and linearity optimization through choosing the best code combination to generate an as small INL error as possible. Because the \text{INL}_{kT} errors become smaller with multiple sub-DACs, the same figure is zoomed in and placed at the bottom (c).

To improve the linearity of the D/A transfer characteristic, the transfer characteristics of the sub-DACs are acquired. The best combinations of \text{INL}_{g8} can reduce the nonlinear effect caused by the mismatch. This will improve the overall INL, as shown in the example with two sub-DACs in Figure 4.5.

Figure 4.5 INL improvement with two sub-DACs
Within each plot: the “blue/dark” one is the binary architecture, “red/grey” simulation is the thermometer architecture and “cyan/light grey” is the segmented architecture. The simulations with these architectures will show that there is no significant difference in the reduction of INL_{max}.

In this example, the INL_{max} of the improved characteristic, created with the best combinations of two sub-DACs is 18% compared to the non-improved INL_{max}. However, in some of the worst cases it is 32%. It can not find a better combination. For example, when both sub-DACs have positive or negative INL_{k} up-to code k, none of the errors can compensate each other to create code k. However, still the combination with the smallest error will be selected.

When more sub-DACs are used, not only the number of combinations (shown in Figure 4.4) is increased but also the combinations itself are (most likely) better. With three sub-DACs there is a high probability that one of the three sub-DACs has an INL_{k} with an opposite polarity (negative when the other two are positive or visa versa). Therefore, also the probability that combinations will be found that do compensate each other is increased. This is shown in the three following figures.

In Figure 4.6, three sub-DACs are used to construct the overall DAC. The improvement is larger and the non-linear errors are especially at the beginning and end of the scale. At the beginning and end of the scale there are less INL_{k} combinations that reduce each other, shown in Figure 4.4b.

In this example, the average INL_{max} with the improved method is 4.1% of the average INL_{max} of the stacked sub-DACs. Figure 4.7 shows that the INL improvement with four sub-DACs is better and the average INL_{max} is only 2.78% of the non improved
INL$_{\text{max}}$. Just as with the three sub-DACs, the non-linearity is mainly in the beginning and end of the code-range.

![Graph a) INL improvement with 4 sub-DACs](image)

Figure 4.7 INL improvement with 4 sub-DACs

Figure 4.8 shows an example of the effect with five sub-DACs. On a normal scale the non-linearity is not visible and therefore only the zoomed version is shown. The average improved INL$_{\text{max}}$ is 1.94% compared to the average normal INL$_{\text{max}}$.

![Graph a) INL improvement with 5 sub-DACs](image)

Figure 4.8 INL improvement with 5 sub-DACs
The maximal INL for each simulation is

\[ \text{INL}_{\text{max}} = \max \left( \text{INL}_{kT} \right), \quad kT \in \{0, 1, \ldots, (2^N - 1)\} \]  

(Eq 4.7)

with \( \text{INL}_{\text{max}} \) being the maximal \( \text{INL}_{kT} \) for this simulation, \( \text{INL}_{kT} \) being the INL for code \( kT \), and \( kT \) being all codes from 0 to \( 2^N - 1 \).

Figure 4.9 shows the distribution of the maximal INLs with two sub-DACs. These \( \text{INL}_{\text{max}} \)'s are from the 700 simulations, shown in Figure 4.5. There are two curves, the first one is the distribution of the improved \( \text{INL}_{\text{max}} \), the maximal deviation for each simulation, shown in Figure 4.5b. The second is the distribution of the \( \text{INL}_{\text{max}} \) of the stacked/non-improved method, maximal deviation for each simulation, shown in Figure 4.5a. The \( \text{INL}_{\text{max}} \) samples of the improved method are in a smaller range and therefore, the density of the distribution is higher.

The \( \text{INL}_{\text{max}} \) of the non-improved simulations were 99% in a 1.8 LSB range. The improved method has 99% of the \( \text{INL}_{\text{max}} \) in the range of 0.65 LSB.

Figure 4.10 shows the distribution of the maximal INLs with three sub-DACs. These \( \text{INL}_{\text{max}} \)'s are from the 700 simulations, shown in Figure 4.6. The \( \text{INL}_{\text{max}} \) of the non-improved simulations were 99% in a 1.8 LSB range. The improved method has 99% of the \( \text{INL}_{\text{max}} \) in the range of 0.24 LSB.

Figure 4.11 shows the distribution of the maximal INLs with four sub-DACs. These \( \text{INL}_{\text{max}} \)'s are from the 700 simulations, shown in Figure 4.7. The \( \text{INL}_{\text{max}} \) of the non-improved simulations were 99% in a 1.8 LSB range. The improved method has 99% of the \( \text{INL}_{\text{max}} \) in the range of 0.09 LSB.

Figure 4.12 shows the distribution of the maximal INLs with four sub-DACs. These \( \text{INL}_{\text{max}} \)'s are from the 170 simulations, shown in Figure 4.8. The \( \text{INL}_{\text{max}} \) of the non-improved simulations were 99% in a 1.8 LSB range. The improved method has 99% of the \( \text{INL}_{\text{max}} \) in the range of 0.05 LSB.

There is a relation between the average \( \text{INL}_{\text{max}} \) and the 99.73% range of the improved and stacked sub-DAC method. When the average \( \text{INL}_{\text{max}} \) and 99.73% range of the non-improved method goes up (or down) the \( \text{INL}_{\text{max}} \) and 99.73% range of the improved method goes up (or down) as well with the same factor.
Two sub-DACs have an average remaining \( \text{INL}_{\text{max}} \) of 17.8\% of the non-improved \( \text{INL}_{\text{max}} \). The best combination of three sub-DACs has an average remaining \( \text{INL}_{\text{max}} \) of 4.1\%, four has 2.8\% and five has 1.9\% remaining \( \text{INL}_{\text{max}} \). The average reduction after and before improvement \( \text{INL}_{\text{max}} \) and between the \( \text{INL}_{\text{max}} \), 3\( \sigma \) level of the optimized and non-optimized method is shown in Figure 4.13 for each number of sub-DACs.

The non optimal combination is the stacked DAC combination, is just improving the resolution. The linearity with the optimized combination is improved with each extra sub-DAC. The unit-element array can be pseudo segmented into more sub-DACs. This method of checking all options is not very practical to use for large amounts of sub-DACs. The number of combinations will grow, see Equation 4.3. When the resolution (unit-elements) is increased, the number of combinations will also grow.
4.3 Finding an optimal combination with higher resolutions

The total number of sub-DAC combinations to evaluate with four sub-DACs is $2^{4N}$, according to Equation 4.3. For example, with four 10-bit sub-DACs there are $2^{40}$ combinations. With a process that can even evaluate 100,000 combinations per second, the initialization of this DAC would still take 4 months. A faster algorithm has to evaluate fewer combinations.

Therefore, a method with a reduced number of combinations is required. However we still want to find “optimal” combinations that are almost as good as the best INL_{kT} combinations. Therefore, four sub-DACs are used. With four sub-DACs there is a high probability that the INL_{k} of one DAC will reduce the INL_{k} of the other. Therefore, it is expected that with “well chosen” combinations a close to the optimal INL_{kT} will be found.

By only actively using two of the four sub-DACs, $2^{2N}$ combinations can be evaluated (see Equation 4.4 and Figure 4.13, Part1). It is clear that with only two active sub-DACs not all of the four sub-DAC current-sources are used. The combined DAC can only reach half of the entire code range. These are $2F - 1$ codes, instead of the full-range with $4F - 3$ codes ($F = 2^N$). To solve this problem the outputs of the two passive sub-DACs can be switched to zero or maximal and the active sub-DACs will evaluate the combinations again to find the small INL_{k}s. The middle of the range options are only used at $1/4$ of the total code range, shown in Figure 4.14, Part 2. The first part is the range from code 0 to $1.5F - 1$, the second from code $1.5F$ to $2.5F - 1$ and the third from code $2.5F$ to $4F - 3$, shown in Figure 4.14 with Part1, Part2 and Part3.

![Figure 4.14 Combinations with two active sub-DACs](image)

In Figure 4.14, the horizontal axis is the digital-code-range and the vertical axis is the number of possible combinations. Part one is the same as Figure 4.4a. In Part1 only 2 DACs are active. In Part2, one passive DAC is switched fully on and the other two DACs make the combinations. In Part3, both passive DAC are switched fully on and the two active DACs make the combinations.
The number of combinations in the first, second and third part for only two active DACs are

\[
\text{Comb}_{\text{scale part1}} = \frac{1}{2}(2F \cdot F) - \frac{1}{2}\left(2F - \frac{1}{2}F\right) \cdot \frac{1}{2}F = F^2 - \frac{1}{8}F^2 = \frac{7}{8}F^2
\]

\[
\text{Comb}_{\text{scale part2}} = F^2 - 2 \cdot \frac{1}{8}F^2 = \frac{3}{4}F^2
\]

\[
\text{Comb}_{\text{scale part3}} = \text{Comb}_{\text{scale part1}} = \frac{7}{8}F^2
\]

\[
\text{Comb}_{\text{Total}} = \text{Comb}_{\text{scale part1+2+3}} = \frac{7}{8}F^2 + \frac{3}{4}F^2 + \frac{7}{8}F^2 = 2.5F^2 = 2.5 \cdot 2^{2N}
\]

(Eq 4.8)

, with \(\text{Comb}_{\text{scale part1,2,3}}\) being the number of combinations in the first, second and third part of the entire scale-range. \(\text{Comb}_{\text{scale part1,2,3}}\) is the total number of combinations with only two active sub-DACs, shown in Figure 4.14.

By only using two active and two passive sub-DACs, it is possible that the mismatch errors of the combined sub-DACs do not compensate each other. With four sub-DACs there are six options to choose the two active DACs. These six options are the combinations of sub-DACs “1&2, 1&3, 1&4, 2&3, 2&4 and 3&4”. When all combinations of those two-combined-DACs are evaluated, it is expected to find combinations with a small INL_kT. At the middle of the range (Part2) either the first or the second passive sub-DAC can be switched on. This will cause that the number of options to combine the sub-DACs with becomes twice as large at the middle of the scale. The options to combine the active and passive sub-DACs are there twelve instead of six times. The combinations with the full-range of this method is

\[
\text{Comb}_{\text{Total}} = 6 \cdot \text{Comb}_{\text{scale part1}} + 12 \cdot \text{Comb}_{\text{scale part2}} + 6 \cdot \text{Comb}_{\text{scale part3}}
\]

\[
\text{Comb}_{\text{Total}} = 6 \cdot \left(\frac{7}{8}F^2\right) + 12 \cdot \left(\frac{3}{4}F^2\right) + 6 \cdot \left(\frac{7}{8}F^2\right) = 19.5 \cdot F^2 = 19.5 \cdot 2^{2N}
\]

(Eq 4.9)

, with \(\text{Comb}_{\text{Total}}\) being the total number of combinations the algorithm checks, 6 being the number of combine-options of four sub-DACs, At the range of midscale (Part2) there are 12 combine-options of the four sub-DACs. The simulated results with this algorithm are still good (shown in Figure 4.16b).

This algorithm has a little “mistake” and does check some combinations three times. It does not see a difference between an active sub-DAC with the sub-code 0 or full-range and the same sub-DAC being a passive sub-DAC that is switched totally on or off. This can be avoided with an algorithm that does not check the combinations with the active sub-DAC codes 0 and full-scale. The amount of extra checked values \((4\cdot 6 \cdot 2^N)\) is negligible compared with the total checked combinations.

Figure 4.15 shows an example with the number of combinations for each code with 7-bit sub-DACs. The method with using all combinations in Figure 4.15a and the method with reduced combinations in Figure 4.15b. Figure 4.15b is the same as Figure 4.14, however with 12 times the combinations in Part2 and 6 times the combinations in Part1 and Part3. The difference in total combinations with just 7-bit
sub-DACs is more than a factor 840 (Equation 4.4/4.9: $2^{2N}/19.5$). With 10-bit sub-DACs the factor is almost 54,000.

![Graph showing combinations with 2 and 4 active sub-DACs](image)

Figure 4.15 Combinations with four 7-bit sub-DACs

This method of reduced combinations is used for a simulation with four 10-bit sub-DACs with a thermometer architecture. The simulation clearly shows peaks at the position where no good $\text{INL}_{\text{IT}}$ can be found (Figure 4.16b). These are at beginning, the end and at the middle of the digital code range. The missing combinations are the main reason of the extra peaks at the middle. It is expected that using an algorithm with more combinations can solve the problems with the peaks at the middle. It is possible to reduce some of the peaks at the beginning and end by using there all combinations of four DACs. Note that the improved version, Figure 4.16b has an INL scale from 0.2 to -0.2 LSB, with the same scale being the non optimized INL, the error spikes will be hardly visible.

![Graph showing fixed and flexible architecture](image)

Figure 4.16 Four 10-bit sub-DACs with combination reduction
In this example, shown in Figure 4.16, the average INL\textsubscript{max} of the improved method is 2.5\% of the non-improved INL\textsubscript{max}. Note that in this example the measurement method of Chapter 5 was used.

It is expected that the errors caused by the measurement-DACs will have more impact than the missing combinations in the algorithm. The peaks in the middle are caused by 8 of the 200 simulations. It is expected that on average the missing combinations can decrease the performance by 0.5\%.

Figure 4.17 shows an example of a parallel architecture is used with segmented 10-bit sub-DACs. The segmentation of this DAC is 2 LSB and 8 MSB. The linearity improvement method is not depended a segmentation. In Figure 4.17a the non-improved INL\textsubscript{max} is 1.7 LSB. There are no measurement errors, the improve INL\textsubscript{max} is 0.01 LSB. The INL\textsubscript{max} was found at the beginning or end of the digital code range.

![Figure 4.17 INL before and after improvement in theory](image)

In Figure 4.17a, the INLs can be shown being the stacked INLs of sub-DAC A, B, C and D. The improved version shown in Figure 4.17b is also shown in Figure 4.17a, this is the line in the middle that looks like the horizontal axis.

Figure 4.17b does not show what part of a sub-DAC contributes to this output-code. To give an idea what the output currents of the sub-DACs are at each code, there is a representation of the sub-DAC currents shown in Figure 4.18. These are the currents to construct the output used to construct the best combination in Figure 4.17b.

Figure 4.19 shows a zoomed-in version of Figure 4.18.
Combined Output of the sub-DACs

Figure 4.18 Sub-DAC outputs for the optimal combination

Combined Output of the sub-DACs

Figure 4.19 Sub-DAC outputs for the optimal combination, of code 3000 to 3100.
4.4 Dynamic performance due to static linearity

The linear improvement will reduce the harmonic distortion in the frequency spectrum. A digital sinewave is converted with an improved and non-improved DAC transfer characteristic. Figure 4.20 shows examples of the frequency spectrum before and after the linearity improvement.

![Spectral content, before and after linear improvement](image)

The spectral content is created by sampling the digital sinewave. The simulation was performed in Matlab. The number of samples per sinewave is

\[ \text{Samples}_{\text{sinewave}} = \frac{F_s}{F_{\text{signal}}} \]  

(Eq 4.10)

, with \( \text{Samples}_{\text{sinewave}} \) being the number of samples per sinewave, \( F_s \) the sampling frequency and \( F_{\text{signal}} \) the frequency of the sinewave.

Figure 4.21 shows the SFDR result of multiple spectral simulations. The \( \text{INL}_{\text{max}} \) (3\( \sigma \) level) of the non-improved characteristic is on the horizontal axis. The sinewaves of the spectral simulations are sampled with more than 2 samples/period (Nyquist sampling frequency) and more than 50 periods.

![SFDR in relation with 3\( \sigma \) INL\(_{\text{max}} \), before and after improvement](image)
5 Self-measurement and self-correction.

This chapter presents a method to measure the INLₖ of a parallel architecture with four sub-DACs. The goal is that the self-measurement method and extra resources can be implemented in the same chip. For this purpose the INLₖ of the sub-DACs have to be determined. INLₖ can not be directly measured because there is no ideal transfer characteristic to compare with. The self-measurement determines the DNLₖ of the sub-DACs and calculates the INLₖ.

5.1 Ideal self-measurement

In the theoretical approach of Chapter 4, the INLs of the sub-DACs were determined by subtracting the actual characteristic from the ideal characteristic. In practice, there is no real ideal characteristic that we can use. Therefore, a different approach is needed.

Our example of a parallel architecture has four sub-DACs. An example of four sub-DAC INL characteristics is shown in Figure 5.1.

![Figure 5.1 INL sub-DAC characteristics](image)

An approach which does not require an ideal characteristic to determine the INLₖs is measuring the DNLₖs and from this constructs the INLₖs.

\[
INLₖ = \sum_{j=1}^{k} DNL_j
\]  

(Eq 5.1)

, with INLₖ being the sum of all DNLₖs.
When it is possible to measure one $DNL_k$, the same approach can be used for all $DNL_k$s of all sub-DACs.

Our method to determine the $INL_k$s of the sub-DACs in the parallel architecture uses some extra resources, being

- $M$ Sub-DACs with scalable currents,
- One 1-bit ADC,
- One 1 LSB current source,
- Some extra switches,
- Digital resources,
  - Algorithm,
    - To measure the $DNL_k$s,
    - To construct the $INL_k$s,
    - To self-correct errors of the constructed $INL_k$s,
    - To temporarily store the $INL_k$s,
  - Memory,
    - To temporarily store the $INL_k$s.

When the $INL_k$s are known, the other resources required to find the optimal combinations, see Chapter 4.3, are

- Digital resources,
  - Algorithm,
    - To find the optimal combinations to minimize of the overall $INL_k$s,
  - Memory,
    - To store the look-up-table.

In Figure 5.2, the sub-DAC A is shown being the sub-DAC that has to be measured. It is unpractical to store the analog output of a sub-DAC in the analog domain, when we want an on-chip implementation. The $DNL_k$ of sub-DAC A can not be measured directly. This is because the sub-DAC can not generate both outputs $I_{aout k}$ and $I_{aout k+1}$ at the same time.

With only one sub-DAC available at a time, the measurement device should be able to measure the entire sub-DAC output range and has to be very accurate. This accuracy is required to reduce the accumulation of quantization and non-linear errors.

To reduce the range of the measurement device, sub-DAC B is used being a static point of reference. The $DNL$ of the sub-DAC is determined as the difference between two measurements $M1$ and $M2$, shown in Figure 5.2. The first measurement is between the output of the sub-DAC A at code $k$ and the static reference. The static reference is sub-DAC B at code $k$.

The measurement is performed by DAC C. DAC C and the 1-bit ADC operate together as an approximation A/D converter. DAC C adds an (scaled) output step until the 1-bit ADC switches, from 0 to 1, or from 1 to 0. The value of DAC C is the representation of the measured value. The second measurement is between the output of the sub-DAC at code $k+1$ and the static reference plus the 1 LSB current source.
The output of DAC C is scaled. This means the output at each code is smaller than the output of DAC A or B. With the 1-bit-ADC it can be detected when the output currents of DAC B+C are sufficiently equal to DAC A. The value of DAC C will be stored in a temporary memory $M1$. Next, the code at DAC A will be incremented by one, code B stays the same and the extra 1-LSB current source will be switched on, shown in Figure 5.2 and 5.3.

The difference between the outputs of sub-DAC A and B will be determined again by adding steps to DAC C. This difference $M2$ and the value in the temporary memory $M1$ are subtracted and is a representation of the DNL size. Based on this the INL can be calculated, see Equation 5.1. This will be done for each code and sub-DAC. The INLs are stored in a memory, see Chapter 8. An algorithm will use the stored INLs to find the best sub-DAC combination for each input code.
The measurement-DAC C should determine the difference between the output of DAC A and B with enough accuracy. Therefore, the steps of DAC C have to be smaller than this difference. However, DAC C also has to be able to determine the largest possible difference. For example, when the mismatch errors in each sub-DAC are able to cause an INL of +/- 2 LSB, DAC C has to be able to measure a maximal difference of +/- 4 LSB. Since in this example single-ended-DACs are used, the negative measurements are done by adding DAC C to DAC A, instead of to DAC B. The differential DAC does not require an extra switch since it directly supplies the positive or negative current. The INL and quantization errors of DAC C are also accumulated with this method, see Equation 5.1. When the output of DAC C becomes smaller, the accumulation also becomes smaller.

In the simulation, shown in Figure 5.8, DAC C is constructed with two current-downscaled-DACs, shown in Figure 5.4. This is required because it has to be able to measure the maximal range between DAC A and B and has requires small steps to reduce the quantization error accumulation. The maximal output of the first “rough” measure-DAC can be downscaled to 16 normal LSBs. The maximal output of the second “fine”-DAC can be maximally downscaled to the DNL of the rough-DAC. The worst case DNL is with a binary architecture and is below 8 LSB, shown in Figure 5.2. The relationship between the minimal measure DAC LSB-currents are

\[ I_{\text{LSB}\text{rough}} = \frac{8}{2^N} \cdot I_{\text{LSB}\text{normal}} = 2^{(N-3)} \cdot I_{\text{LSB}\text{normal}} \]

(Eq 5.2)

\[ I_{\text{LSB}\text{fine}} = \frac{I_{\text{LSB}\text{rough}}}{2^{N-3}} = 2^{(2N-6)} \cdot I_{\text{LSB}\text{normal}} \]

, with \( I_{\text{LSB}\text{normal}} \) being the current of one normal DAC LSB, \( I_{\text{LSB}\text{rough}} \) being the current of 1 rough DAC LSB and \( I_{\text{LSB}\text{fine}} \) being the current of 1 fine measurement DAC LSB.

The output current of the rough-measure-DAC is added to the output current of DAC B and approaches the output of DAC A. The 1-bit ADC detects when the current from the rough DAC becomes larger or smaller than the output difference between DAC A
and B. The remaining difference will be determined by the fine-measure-DAC. In the simulation, shown in Figure 5.6, 10-bit sub-DACs are used to construct one 12-bit DAC. The rough-measure-DAC is compared to the normal DAC $2^2$ times downscaled and the fine-measure-DAC $2^{10}$ times. Only $2^{10}$ times because with binary/segmented architectures the non-linearity of the measure DAC appears to be larger error than the quantization noise. Therefore, a larger resolution (by downscaling) does not improve the measurement.

The maximal current of the fine DAC can be smaller than the non-linearity deviation of the rough DAC. Therefore, this method is not useful for a normal use of high resolution DACs. This kind of downscaling is used for very low resolution (2-bit) DACs in [Chow 1994]. However, the rough DAC is used being a static reference point that does not change often. When the rough DAC stays at the same code, the non-linearity of the rough DAC does not affect the measurement. Therefore, the measurement only uses a small part of the transfer curve of the combined measurement DAC. The binary architecture is the worst because by the large DNL steps, the rough-DAC is not always a static reference point and is active with the measurement.

The measurement error can be reduced when the difference between A & B/C/D is measured multiple times with changing the rough and fine DAC, see Chapter 5.2.3.

The measurement circuits of Figure 5.2 and 5.3 use four sub-DACs. This is the method used to construct the INL$_k$s and is the main reason why this thesis concentrates at an architecture with four sub-DACs. With Figure 5.1b the measurement of 1 DNL$_k$ is explained, with the next algorithm the measurement of all codes for the four DACs is explained, shown in Figure 5.6.

The sub-DACs are used in a different way for each measurement. For each measurement, this is

- One sub-DAC is the DAC to be measured. The INL$_k$s of this DAC will be measured. This measured DAC is called DAC$_A$.
- One sub-DAC is the follow DAC$_B$, this one is only used as a fixed reference point for each DNL$_k$ measurement. At the beginning of the INL$_k$ measurement this DAC$_B$ follows the code of DAC$_A$.
- The other two DACs will be downscaled and used as the measuring DAC$_C$. DAC$_C$ measures the deviation between the measured DAC$_A$ and follow DAC$_B$.

When more than four sub-DACs are used, this does not require more resources, other than the extra sub-DACs. However, when less than four sub-DACs are implemented, this measurement method is not an option.

The algorithm to construct all the INL$_k$s is shown in Figure 5.5. The algorithm can be implemented on-chip as a Finite State Machine [Radulov 2005].

The state diagram gives an overview how the INL$_k$s of all DACs are determined. Each block is a state and has operations that are processed before the algorithm can go to the next state.
At initialisation the algorithm receives information like the operation-mode and how many $M$ DACs it has, etc. $x$ is the number of the sub-DAC that has to be measured, and is set on zero. At the end of the algorithm $x$ is $M$ and all sub-DACs are measured.

At the second block, $x$ is increased and $DACA$ is sub-DAC$1$, $DACA$ is the sub-DAC that has to be measured. $DACB$ is the follow DAC. It does not matter which DAC it is, as long as it is not the same sub-DAC as for $DACA$.

The current source is turned off and in the next block the difference between DAC A and B is measured. This measurement is done with DAC C and the result is put in memory $M1$.

The current source is switched on and added to DACB. The input code of $DACA$ is increased by 1.

Now the difference between $DACA$ and $DACB$ plus LSB is measured again with DACC and the value (steps) is stored in $M2$.

The INL$_{k+1}$ for DAC$_{A}$ at code $k+1$ is the INL$_{k}$ plus the measured representation of the DNL$_{k}$ "$M1-M2$".

If not all INL$_{s}$ are constructed yet, the algorithm repeats the last 5 states until all INL$_{s}$ are constructed.

When this is done and not all the sub-DACs are measured, the algorithm repeats the last 6 steps until they are. When all INLs are constructed a self-correcting method of the measurement errors can be applied, see Chapter 5.2. After the self-correction of the measurement errors, the INLs are constructed and can be used for finding/constructing the optimal output combination, see Chapter 4.3.

---

**Figure 5.5 Self-measurement algorithm**
This algorithm was implemented and tested in Matlab. For four 10-bit sub-DACs this construction cost 4 seconds. For comparison, the algorithm to find the option options (see Chapter 4) for all codes, takes 70 seconds.

It is expected that for each DNL it requires two measurements (for $M_1$ and $M_2$) and 100 steps/measurements of $DAC_C$. To reduce measurement and noise errors (see Chapter 7.2) the measurement is repeated three times. In this example there are four sub-DACs and the total number of measurement steps of $DAC_C$ is

\[
 Nr_{Total \ steps} = M \cdot 2^N \cdot 2 \cdot Nr_{Steps\ Measure} \cdot 3
\]

\[
 Nr_{Total \ steps} = 4 \cdot 2^{10} \cdot 2 \cdot 100 \cdot 3 = 2,457,600
\]  

(Eq 5.3)

With a relatively slow DAC measurement of 100,000 samples per second, this costs 24.6 seconds. This measurement method only has to be done once after the production. Provided the mismatch errors stay the same after the chip is produced. For example, when the temperature of the chips changes, the area of the transistors and therefore the mismatch errors change. However, all mismatches change and are relative to each other, they stay approximately the same. Therefore, with the optimal combination, the mismatch errors still compensate each other.
5.2 Self-correction of the measurement errors

Some errors are introduced with this measurement method. These are the accumulation of the
- Mismatch error of the 1 LSB current source
- Quantization error of the measurement
- Non-linearity of the measurement

5.2.1 Measurement error by 1 LSB current source

This self-measurement method uses an extra 1-LSB current-source which is not equal to the average of the DAC current sources. This will introduce an error that is accumulated and stored with the sub-INLₖₛ. This error is linear and can be determined because the sum of the $M$ INLₖₛ at full-scale should be zero, shown in Figure 5.6. If it is not zero, there is a gain error in the measurement. This gain error affects all the measured INLₖₛ and the stored INLₖₛ have to be adjusted with this calculated gain.

![Figure 5.6 INLs and gain of the four sub-DACs](image)

5.2.2 Measurement error by quantization and non-linearity

Improving the resolution of the measurement DAC can reduce the accumulation of the quantization errors.
The effect of the non-linearity of the measurement DAC is related to the architecture of the sub-DACs. With thermometer architectures it has almost no effect, because all DNLₖ steps between the measured values are small. Therefore, the step itself and the measurement DAC have small DNLₖₛ. However with binary sub-DAC architectures the DNLₖₛ are larger. At a large DNLₖ the measurement DAC has to measure a large
range. Even for small steps the measurement DAC also has large a DNLₖ itself. If in the first output value of the measurement DAC was an error of 1 LSB and in the second output an error of -1 LSB, the measurement error between those two codes is 2 LSB. When there are a lot of large DNLₖs, the measurement errors do not have to compensate each other. Therefore, with this method the measurement errors are worse for the binary architecture than for the thermometer architecture.

This self-correction method can compensate a part of the quantization and non-linearity errors.

The ideal characteristic is a straight line drawn between the minimal and maximal output of the combined sub-DACs. Therefore, the sum of the sub-DAC's INLₖs at zero and full-scale are zero, shown in Figure 2.4 and Equation 2.3. Also the difference between the sub-DAC A, B, C, and D can be measured at zero and at full-scale. This measurement has only one time the quantization and non-linearity error. The difference between the calculated and the measured INLₖ (at full-scale) can be corrected. Therefore, the measurement error in the INL at full-scale is very small. The other codes have an accumulation of quantisation and non-linearity errors. The other stored code values are corrected by this deviation as a gain error. This also automatically compensates the error caused by the mismatch of the 1 LSB current source.

Figure 5.7a shows an example of four 10-bit sub-DACs with a conventional [8 MSB 2 LSB] segmentation, integrated in a parallel architecture.

The theoretical measurement of the sub-DACs is done with the method described in Chapter 5.1. In this example, the downscaled measurement DACs have the same characteristic as in the normal state. The measurement errors that are accumulated in the INLₖs are shown in Figure 5.7b.

Figure 5.8a shows the INLs of the sub-DACs in stacked state and Figure 5.8b with the optimized method.

![Figure 5.7 sub-DAC INL and measurement error](image)
The difference between Figure 5.8b and Figure 4.17b is caused by the measurement errors. The measurement errors cause directly an error at the code of the optimized INLₜ.

### 5.2.3 Averaging multiple measurements

The measurements can become more accurate when multiple measurements are averaged. For a second measurement the rough and fine DAC can be swapped and the values of the first and second measurement being averaged. Hence, the measurement errors can decrease with almost 30% (see Appendix A, Figure A.4 and A.5). An example is shown in Figure 5.9. This figure shows a decrease of measurement errors and Figure 5.10 shows the direct effect of this in the best combination.

There is also another way to achieve an averaging of the measurement errors. Already explained in Chapter 5.2.1 is that the sub-DAC outputs are compared to each other at code zero and full-scale. This to compensate a gain error caused by the measurement errors. This method can be applied at every code. However it only gives an indication of the INLₐₛ compared to each other. The sum of the INLₐₛ is not zero. The calculated (accumulated) and measured indication of the INLₜ can be averaged again. This method can be applied after the INLₐₛ are already determined. This last method is not applied in the measurements of this thesis. This, because the results of the method with just one or multiple DNL measurements is sufficient.
In Chapter 4, the passive sub-DACs are only used as fully on or off. This choice was not random. This, because there is no accumulation of the measurement errors at these codes. Therefore, the calculated best result has maximally only two values with accumulated measurement errors.

Multiple Monte-Carlo simulations have been performed without the averaging method and the sub-DACs downscaled as the measurement DACs (see Appendix A). In these simulations the optimizing method with 10-bit linear accurate sub-DACs has an average improvement of more than 4 bit.
6 Current scaling

This chapter describes the current scaling of the sub-DACs. Firstly a short introduction is given why the sub-DAC currents should be scalable, followed by how this can be achieved and what the side effects are.

In the parallel architecture, there are multiple operation modes. In order to be able to use the same sub-DACs for all operation modes, the output current has to be scalable. This is because:

- The output current of the combined $M$ sub-DAC would otherwise be $M$ times larger.
- It eliminates the need for an extra/external measurement ADC or DAC.

The outputs of a DAC are connected to two load resistors of $25\Omega$. The load resistors are required to convert the output current to an output voltage. The resistor value (of $50\Omega$ differential) is standard to terminate reflections.

For example, a sub-DAC (working as a separate DAC) has a maximal output current of 20 mA. This current causes a maximal voltage over a resistor of 0.5 Volt (1 Volt differential). The unit-element, cascode transistor and switch need a minimal output voltage of approximately 1.1 Volt, shown in Figure 6.1. The DAC works properly when the resistors are connected to 1.8 Volt (this is typical for 0.18μm CMOS) and therefore the output is minimal 1.3 Volt. When multiple sub-DACs are used in parallel, the current would increase and also the voltage over the resistor. For example with two sub-DACs the maximal output current becomes 40 mA and the output voltage is 0.8 Volt.

![Figure 6.1 Switched current cell circuit diagram](image)
Therefore, the maximal output current is required to be equal when multiple DACs work together. When $M$ DACs are used, the current supplied by each sub-DAC has to be reduced $M$ times. When the chip is specially designed for the linearity improvement method, the unit-elements can be designed to supply a lower current.

Another reason for flexible scalability is, to use the sub-DACs together with the 1-bit ADC to measure the DNL₄s. The scaling is required for an on-chip implementation of the self-measurement. The DNL₄s are small and therefore, the analog output step-size of the measurement DAC has to be small. The small steps are also required to reduce the effect of the quantization error accumulation. In Chapter 5.2 was shown that a downscaling of 1024 times was enough to determine the INL₄s of the sub-DACs accurately enough.

### 6.1 Simulation current scaling

Without changing major parts of the DAC design, the current supplied by the DAC can be reduced at two positions. These are at the output of the DAC and at the unit-elements.

Extra components at the output will decrease the performance of the DAC in normal operation. For example, the components introduce extra impedance and power consumption.

The currents of the unit-elements can be reduced by changing the $V_{gs}$ of the current source transistor. The $V_{gs}$ to $I_u$ characteristic of an unit-element is shown in Figure 6.2. At a $V_{gs}$ of 840 mV each unit-element has an output current of 5 $\mu$A.

To reduce the current 1000 times, the $V_{gs}$ will be lowered. By switching the $V_{gs}$ to 246 mV the output current is reduced to approximately 5 nA.

![Figure 6.2 Current scaling of the unit element](image)
In the simulation program (Cadence), the transistors are designed identically and do not have a mismatch. The sub-DAC works with the Vgs below the threshold voltage $V_{Th}$ (sub-threshold). The unit-elements are designed in 0.18μm CMOS. When we scale the current by reducing the Vgs below $V_{Th}$, the mismatch of the currents increases [Lin 1998, Razavi 2001, Gray 1984].

When the Vgs is above the threshold voltage $V_{Th}$, the behavior of the transistor curve is approximately linear. Below the threshold voltage $V_{Th}$ the behavior of the curve is approximately exponential.

Mismatches cause that the threshold voltage of each unit-element is different. The standard deviation of the threshold can be calculated with the formula derived from [Pelgrom 1998]:

$$\sigma(\Delta V_{Th}) = \frac{A_{V_{Th}}}{\sqrt{L \cdot W}}$$  \hspace{1cm} (Eq 6.1)

The unit-element for the simulation is designed in 0.18μm CMOS. The width is 840 nm and a length is 5 μm. For 0.18μm CMOS transistors, a typical value for $A_{V_{Th}}$ is approximately 6 mV μm. With these values, the $\sigma(\Delta V_{Th})$ is

$$\sigma(\Delta V_{Th}) = \frac{6}{\sqrt{5 \cdot 0.84}} = 2.93 \text{ mV}$$  \hspace{1cm} (Eq 6.2)

, with $\sigma(\Delta V_{Th})$ being the standard deviation of the offset at $V_{Th}$, $L$ the length, $W$ the width.

This deviation can be seen as an offset voltage added to the threshold voltage $V_{Th}$, $\Delta V_{Th} = V_{os}$, shown in Figure 6.3. It can be assumed that the deviation from the transform curve below threshold is not larger than the deviation of the threshold voltage $V_{Th}$. Therefore, the deviation of $V_{os}$ is maximal 3 mV. With this deviation and transistor characteristics it can be calculated what the effect for the current mismatch will be. Figure 6.4 shows the effect with +/- $V_{os}$ of 3 mV for the current mismatch error.

![Figure 6.3 Offset in the transistor](image_url)
The graph in Figure 6.4 is derived from Figure 6.2 with a +/- $V_{oa}$ of 3 mV

$$\frac{\sigma(\Delta I_{\text{vgs}})}{I_{\text{vgs}}} = \frac{(I_{\text{vgs}+3mV} - I_{\text{vgs}-3mV})}{2 \cdot I_{\text{vgs}}} \cdot 100\% \quad (\text{Eq 6.3})$$

, with $I_{\text{vgs}}$ being the current of the unit-element at $V_{gs}$, $I_{\text{vgs}+3mV}$ the current with $V_{gs} + \Delta V_{oa}$ and $I_{\text{vgs}-3mV}$ the current with $V_{gs} - \Delta V_{oa}$.

The voltages where the transistor has an output of 5 nA and 5 $\mu$A are shown in Figure 6.4.

Table 6.1 shows the values of the current reduction with a factor 1, 4, 1000 and 4000.

<table>
<thead>
<tr>
<th>Scaling factor</th>
<th>$V_{gs}$ [mV]</th>
<th>$I_u$ [nA]</th>
<th>$\sigma(\Delta I)/I$ [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>850</td>
<td>5000</td>
<td>1.13</td>
</tr>
<tr>
<td>4</td>
<td>580</td>
<td>1250</td>
<td>2.23</td>
</tr>
<tr>
<td>1000</td>
<td>246</td>
<td>5</td>
<td>8.27</td>
</tr>
<tr>
<td>4000</td>
<td>199</td>
<td>1.25</td>
<td>9.20</td>
</tr>
</tbody>
</table>

The transistors were designed with an overall mismatch $\sigma$ of 1.5%. This overall current mismatch error is the combination of the threshold mismatch and the area mismatch error. The measurement scaling factor is 1000 times and the actual mismatch $\sigma$ of the measurement DAC is expected to be 9%.

The current of the unit-elements is scaled through $V_{gs}$. This increases the mismatch and hence the INL of the scaled measure DAC is increased. Because of this, the measurement errors will increase as well.
6.2 Scaling effect in the measurement

The sub-DACs have to be measured in the same state (factor) as they will be used to combine the main DAC, shown in Figure 6.5a. Due to the extra mismatch errors, the accuracy of the measurement DAC will be reduced. This extra error will be accumulated and will produce an error in the measured INL. Figure 6.5b shows the difference between the actual INL\(_{\text{real}}\) and the determined INL\(_{\text{measured}}\) of the measurement method.

The measurement error is small at the beginning and end of the scale. This because of the self-correction of the measurement errors, explained in Chapter 5.2.2.

The determined INL\(_{\text{measured}}\) is used to find the optimal combination. However, the truly optimal combinations are not recognized due to the measurement errors. The optimal combinations are found with the INL\(_{\text{measured}}\) of the sub-DACs. Therefore, the optimal combination will have the sum of the measurement errors of the used INL\(_{\text{measured}}\)s. The real INL with the optimal combination of the determined INL\(_{\text{measured}}\)s is shown in Figure 6.6b.

Figure 4.17b was constructed with the INL\(_{\text{real}}\)s. When both figures are compared, it is clear that the extra deviations in Figure 6.6b are caused by the measurement errors.

The sub-DACs have a 8-2 segmentation and the unit-elements have a mismatch distribution (\(\sigma\)) of 1.5%. The measurement is conducted with the same sub-DAC segmentation however with a mismatch distribution of 9%.
The measurement errors can be reduced by using multiple measurements and averaging those. See Chapter 5.2.3 and Appendix A. In the example of Figure 6.7, six measurements are taken by switching the follow, rough and fine measure DAC. In this example the measurement is improved by 60%, from +/- 0.1 LSB to +/- 0.04.
The errors in the measurement are reduced, therefore also the errors in the optimal combinations are reduced. The maximal \( \text{INL}_k \) in the example of Figure 6.8b is 0.07 LSB instead of 0.17 LSB, shown in Figure 6.6b.

![Figure 6.8 Improvement with two-measurements averaging method](image)

6.3 Conclusions

The current of the unit-elements are downscaled by changing the \( V_{gs} \). The \( V_{gs} \) of the scaled unit-elements is below \( V_{Th} \). This increases the mismatch and hence, the linearity of the DAC while being downscaled is worse. The extra measurement errors are accumulated in the determined \( \text{INL}_k \). The measurement errors can be reduced by using multiple measurements. The (remaining) measurement errors directly reduce the accuracy of the optimized combinations.
7 Transistor level simulations with the parallel DAC architecture

This chapter describes the transistor level simulations of the parallel sub-DACs. Important simulations results are presented. These include static and dynamic DAC performance with and without the improvement method. Furthermore, a system noise expectation of the combined DAC is presented.

7.1 Transistor level simulation

In this section, transistor level simulations of the parallel DAC architecture are presented. The simulations are performed with four 10-bit sub-DACs, designed in 0.18μm CMOS. The segmentation of the sub-DACs is 8 LSB and 2 MSB. Note that main parts of the DAC design were taken from an existing DAC. The designed transistor model of the DAC has no mismatch. In each sub-DAC a static mismatch error was added to the current-sources. With these mismatch-devices, the effect of the optimal distribution is simulated.

The self-measurement and the self-optimizing methods were not simulated in Cadence. This would cost too much time, so they were conducted in Matlab. The digital control-part is created in VerilogA, (see Appendix B). It receives the input signal and distributes it to the sub-DACs. Two ways of distribution are used, stacked and the optimized distribution of the input code.

7.1.1 Static performance

The design in the simulation program uses four sub-DACs. Figure 7.1 shows the INL of the combined DAC without any mismatch errors. Without the mismatches in the unit-elements, the transfer curve is still non-linear. The non-linearity is caused by the finite output resistance of the transistors and the output voltage. The behaviour of the combined INL curve does not change due to the code distribution. When the mismatch of the unit-elements is 100% reduced, this non-linearity still remains.

The finite output resistance non-linearity of each separate sub-DAC is smaller than of the combined DAC. Only one sub- DAC is used and the output voltage is 4 times smaller than with a combined DAC. Therefore, the effect on the INL is smaller, shown in Figure 7.2. When the output voltage is kept at the same potential during the self-measurement, the INL due to the finite output resistance will be negligible.

The $\text{INL}_{\text{max}}$ of the combined DAC and sub-DAC are +/- 0.1 LSB and +/- 0.019 LSB respectively.
The current mismatches are added in the simulation by a mismatch generator. The INL of the DAC is the sum of the INLs caused by the finite resistance and the current mismatch. Figure 7.3 and 7.4 show simulations that use the created current mismatches from the simulation shown in Figure 5.8. The mismatch of the unit-elements is larger than the mismatch of the finite output resistance, shown in Figure 7.3.

In Figure 5.8b the linearity is improved by the optimized code distribution. The same code distribution is used in the transistor level simulation. Figure 7.4 shows a maximal INL of +/- 0.12 LSB due to the finite resistance mismatch instead of 0.04 LSB caused by the unit-element mismatch.
The applied self-measurement and the optimization method do not determine the INL behaviour due to the finite output resistance. However, after the DAC is designed and the typical behaviour is known (by measurement, calculation or simulation) it can be reduced.

The typical curve behaviour can be added to the algorithm to find the optimal combination. The new optimal combinations are based on the combination that cancels both the current mismatch error and the average typical INL behaviour. In this example the INLmax of the finite output resistance is 0.1 LSB and was not compensated.

The static linear accuracy achieved with the stacked method (shown in Figure 7.2) is 10-bit. The simulation with the improved method has a linear accuracy of more than 14 bit.
7.1.2 Dynamic performance due to static errors

The harmonic distortions that are caused by the static non-linearity are the main interest in the dynamic performance simulations.

Figure 7.5 shows the frequency spectrum of a converted sinewave with the stacked non-improved sub-DACs. The harmonics are larger than the noise level before the improvement. The sample frequency was 106 MHz and the signal frequency was 5.1 MHz.

![Before improvement](image)

Figure 7.5 Frequency spectrum of signal with the stacked method

Figure 7.6 shows the frequency spectrum of a converted sinewave with the linearity improvement method. After the improvement, the harmonics are reduced and sometimes masked by the noise. The sample Frequency was 106 MHz and the signal frequency 5.1 MHz.

![After improvement](image)

Figure 7.6 Frequency spectrum of the signal with the optimized code distribution.
Figure 7.7 shows the SFDR measurements with the simulations of the improved and non-improved method.

![SFDR Cadence](image)

Figure 7.7 SFDR in related to the frequency

The spectral simulations are performed from 1 to 22 MHz. The sample frequency is 106 MHz. Above 16 MHz the main harmonics are masked by the noise and more difficult to determine. On average a SFDR improvement of 13.5 dB was achieved.

### 7.2 Circuit noise effect on the self-measurement.

Each transistor of a DAC has (thermal) noise. The sum of all transistor noise is the DAC circuit noise. Normally the circuit noise is smaller than the quantisation noise. However, with the self-measurement method the currents of a sub-DAC are downscaled. The noise can deteriorate the measurement. For this reason, it is required to know the order of the circuit noise.

The sub-DAC circuit noise can be simulated with Cadence. The program gives the size of the noise power density in the sub-DAC outputs in $V^2/Hz$. The noise power up to a frequency is the integration (sum) of all the noise power between zero and the frequency. Assumed is that the internal impedances of the sub-DACs will filter the noise above 1 GHz.

Figure 7.8 shows the (intergraded) noise level created by two normal scale sub-DACs. One would be the measured DAC and one the follow DAC. The noise of the downscaled DACs and the ADC are expected to be smaller than those of the normal scaled sub-DACs.
The simulated integrated differential noise power at 1 GHz is 258 pV^2. To compare this with the current of the downscaled DAC, the maximal differential noise current is

\[
I_{\text{noise}} = \frac{U_{\text{differential}}}{R_{\text{differential}}} = \sqrt{\frac{P_{\text{noise}}}{R_{\text{differential}}}} = \sqrt{\frac{2.58 \cdot 10^{-10}}{50}} = 321 \text{ nA}
\]  
(Eq 7.1)

The differential current of \(I_{\text{measure}}\) is 10 nA and therefore the noise is 32.1 LSB\(_{\text{measure}}\) (=LSB\(_{\text{fine}}\) is 0.032 LSB\(_{\text{normal}}\)). This will have an effect on the self-measurement.

The circuit noise is a normal distributed process. With multiple measurements the actual value can be determined [Razavi 2001]. The measurement with DAC-C and the ADC is an approximating measurement. This means DAC C adds a small step to the current of DAC B, until this value becomes larger than the value of DAC A. The ADC switches from “0” to “1”. In Chapter 5.1 the algorithm stops and the value of DAC C was stored in memory \(M1\) or \(M2\). Since there is noise, the ADC could be switched to 1 because of the added noise. Therefore, the self-measurement will do more measurements. For example, when the ADC switches from “0” to “1”, DAC C keeps adding steps until the ADC output has stayed at “1” for 10 steps. Each time the ADC output was “1”, a counter was increased. When the ADC has no “0” output anymore, the value of DAC-C is subtracted with the value in the counter. This will function as an averaging window of the measured ones and zeros, shown in Figure 7.9. To acquire an even more accurate measurement, DAC C can decrease the steps and count the zeros, etc. Since the fine, rough and follow DAC can replace each other, this measurement can be done 6 times. This also reduces the non-linearity in the measurement, see Chapter 5.2.3.
7.3 Conclusions

The performance of the linearity improvement method is verified by simulations on circuit level. The optimal distribution improved the linear accuracy with more than 4 bit. The system noise is expected to be maximal 33 LSB of the measure DAC (≈ 0.033 LSB). The system noise is random and the effect is expected to be eliminated with multiple measurements. The multiple measurements will be achieved by modifying the self-measurement algorithm.
8 Extra resources

This chapter is about the extra resources required for the pre-correction, self-measurement and self-correction algorithms.

8.1 Memory Method

The algorithm of Chapter 4 evaluates the combinations to construct the combined transfer characteristic. The optimal sub-DAC combinations are stored in a memory, e.g. a lookup-table for the digital pre-processing. The control-part retrieves for each input-code the corresponding sub-codes from the look-up-table and sends these to the sub-DACs, shown in Figure 1.1 and Figure 8.1. With four 10-bit sub-DACs, there are \( (4 \times 2^{10} - 3) = 4093 \) input codes, see Equation 4.2. The size of the memory for this lookup table with four active sub-DACs is

\[
\text{Memory_{lookup}} = \text{Input}_{codes} \cdot M \cdot N = \left(2^{12} - 3\right) \cdot 4 \cdot 10 = 163.84 \text{ Kbit} \quad \text{(Eq 8.1)}
\]

, with \( \text{Memory}_{lookup} \) being the total lookup-table pre-correction memory, \( \text{Input}_{codes} \) the number of possible input codes, \( M \) being the number of sub-DACs and \( N \) the sub-DAC resolution.

Figure 8.1 shows the architecture with the memory blocks. For each digital input code, the corresponding sub-DAC codes are stored in the memory. When the DAC is operational and receives the digital input code, the corresponding sub-codes are sent to the sub-DACs.

![Figure 8.1 Required look-up-table memory](#)
The fourth sub-DAC code can be calculated by subtracting the other sub-DAC codes from the input-code. In this way, the size of the memory can be decreased by 25%. The fourth sub-DAC code is

\[ k_4 = k_T - (k_1 + k_2 + k_3) \]  

(Eq 8.2)

, with \( k \) being the input code of the sub-DAC and \( k_T \) as the digital input code.

The algorithm of Chapter 4 has only two active DACs. The sub-code of the two passive DACs is zero or maximal, depending on the input-code. The sub-codes of the second active sub-DAC can be calculated from the first one and the input-code. This since the third and fourth sub-code is also calculated from the input-code.

The second part of the look-up-table contains information about which DAC are active and passive. The first part has the information of the sub-codes for the first active DAC. Chapter 4 explains that there are 6 and 12 options to combine the active and passive sub-DACs for \( \frac{3}{4} \) and \( \frac{1}{4} \) of the code range. The 6 options cost 3 bit and 12 options cost 4 bit. The memory of the lookup table with two active sub-DACs is

\[ Memory_{Table} = \text{Input \_ codes} \cdot \left( N + \left( \frac{3}{4} \cdot 3 + \frac{1}{4} \cdot 4 \right) \right) \]  

(Eq 8.3)

\[ Memory_{Table} = (2^{12} - 3) \cdot (13.25) = 54.23 \text{ Kbit} \]

, with \( Memory_{Table} \) being the size of the total lookup-table memory by using this retrieving method and \( \text{Input \_ codes} \) the number of possible digital input codes.

This method decreases the required memory by two-third, see Equation 8.1 and 8.3. The algorithm sends the sub-codes to the sub-DACs after it retrieves those from the memory and input-code, shown in Figure 8.2.

There is also an extra temporary memory required. This to store the \( \text{INL}_{ak} \) of the four sub-DACs before they can be used to calculate an optimal combination. This extra temporary memory is

\[ Memory_{Temp} = 2^N \cdot M \cdot N = 2^{10} \cdot 4 \cdot 10 = 40.96 \text{ Kbit} \]  

(Eq 8.4)
Figure 8.2 Reduced memory block for the look-up-table

Figure 8.3 shows the building blocks for the self-measurement method algorithm and the extra temporary memory. The self-measurement algorithm stores the $INL_k$ values in the temporary memory. When this algorithm has measured and self-compensated these values, the linearity improvement algorithm will calculate the optimal values. The sub-DACs codes belonging to these optimal values are (compressed and) stored in the look-up table.

Figure 8.3 Measure algorithms and memory

When both algorithms are executed, the temporary memory can be released and reused for other processes. Only the memory of the look-up-table remains, shown in
Figure 8.1. When the first algorithm if finished, the sub-DACs can already be used (for a non-optimized op-mode).

The calculated $INL_{kT}$ of the combined DAC is also temporarily stored. It is possible to examine for one input code all valid combinations of the sub-DACs. When all combinations are examined, the algorithm can proceed with the next input-code. The same 12-bit memory can be reused to store the value of $INL_{kT}$, this size is negligible in practice. With Matlab simulations, the $INL_{kTs}$ were stored in a matrix.

The total required amount of memory with this approach is 95 Kbit. When the lookup table of the DAC is initialized, the 41 Kbit temporary memory is not used any more and can be reused.

The physical size of the memory will become smaller in the future. The new generations FPGAs already have up to 50.9 Mbit internal memory [Xilinx-1]. When parallel DACs are added to these FPGAs, the internal memory and digital resources can be used. However, it is possible that the FPGA introduces extra noise.

The linearity improvement method in this master project uses four sub-DACs. This is because of the measurement method requirements of Chapter 5. When more sub-DACs are used, the maximal amount of required memory increase, see Equation 8.1. The extra sub-DACs do not require being down-scalable, because the measurement can be accomplished with the four down-scalable sub-DACs. Therefore, beside the extra sub-DACs and memory, there are not more extra required resources.

### 8.2 Extra unit-element.

The measurement algorithm requires one extra unit-element that can be switched on and off by the measure algorithm, see Chapter 5. The mismatch error of the extra unit-element is accumulated being a linear gain error and can be fully corrected.

If the measurement gain errors in the $INL_{48}$ are not compensated, they can become non-linear errors in the combined DAC. Due to the distribution of the input code, the $INL_{48}$ of one sub-DAC are combined with the $INL_{48}$ of other sub-DACs, see Equation 4.5.

In theory, it would be useful to design one very accurate unit-element. A theoretical current-source of exactly $I_{LSB}$ has no error and does not accumulate with the measurement method. However the other unit-elements combined still deviate from the designed value. $2^{11}$ unit-elements with a mismatch $\sigma$ of 1.5\% have approximately a combined deviation ($3\sigma$ level) of +/-2 $I_{LSB}$. Therefore, it is not useful to design a very accurate $I_{LSB}$ current if the average $I_{LSB}$ deviates. Even without other errors, see Chapter 5, the self-measurement algorithm has to compensate the measurement gain error.
8.3 1-bit ADC

The self-measurement method requires a 1-bit ADC as a current comparator. A 1 bit ADC uses only a few components and therefore requires a small silicon area and can be implemented on-chip, see [Radulov 2004]. In the simulations of the project the 1-bit ADC was assumed to be ideal. It did not create extra noise and did not have an offset. Assumed is that the noise of the 1-bit ADC is negligible compared to the noise from the sub-DACs. The offset does not cause a problem with differential DACs because all measurements are relative.

The self-measurement method can also be used by single-ended DACs. The ADC offset can cause a measurement error with single-ended DACs. The polarity of the relative measured value can change and the measure DAC_c should be added to the other DAC_A to subtract the value (shown in Figure 5.2). The offset will add a measurement error. The polarity change can be avoided by selecting a smaller or larger value for DAC_B. Another method is to measure the offset with DAC_c and use this to compensate the possible offset error in the measurement. The offset measurement can be achieved by measuring the value (with DAC_c) between A and B and after this, switch the ADC inputs and measure the value again. The difference between these two measurements is two times the offset.

8.4 Speed limitations of the linearity improvement method

With the linearity improvement method, the speed of the D/A conversion is limited by some factors. These are the

- Speed of the digital circuits, receiving the input word, retrieving the data from look-up-table and sending the codes to the sub-DACs,
- Extra glitches with switching the sub-DAC currents between codes,
- Dynamic performance of the sub-DACs.

8.4.1 Memory

Not only the availability but also the speed of the memory is important. The on-chip memory of the FPGA has a speed of 450MHz with 64 bit/clock, see the webpage of FPGA manufacture [Xilinx-2]. The architecture of Figure 8.1 requires 40 bit/clock. The data of the memory must be sent directly to the sub-DACs. The sub-DACs can work with a sampling frequency of 450MHz.

With the memory reduction method of Figure 8.3 the decoding “algorithm” requires extra clocks. Note that the decoding “algorithm” shown in Figure 8.3 can be implemented in normal digital building blocks and then only requires 1 clock-sample. With one extra clock sample, the sampling frequency of the main DAC is reduced to maximal 225 MHz.
8.4.2 Extra glitches

Extra glitches are caused by the switching of the current sources. With the linearity improvement method it is possible that two following codes are created by two different sub-DAC currents. This also happens at half-scale with a binary architecture, when the LSB are replaced by one MSB current. However with the binary architecture this only happens at that transmission. With the optimized method this can appear more than once. In Cadence with the sinewave simulation, these glitches have the same size as the glitches with the stacked version. The width of glitch is 2 ps and the size is appears to be maximally 25 LSB. Due to the relatively small width of the glitch, it hardly seems to affect the frequency spectrum.

If in reality, these glitches do reduce the performance (with small or slow signals), it can be useful to use another optimization. This optimization checks not only if a combination is still acceptable (small INL) but also if it does (not) change much with the last combination.

The expected measurement error has to be known to determine what an acceptable deviation is. For example, the best combination for four sub-DACs is shown in Figure 4.17b. Since the maximal INL is 0.01 LSB, this would not be an acceptable deviation. However, Figure 5.8 shows the same DACs with an optimal combination with measurement errors and 0.01 LSB is acceptable. DACC has a downscaling of 2^10, an acceptable deviation of 0.01 LSB would be +/- 10 steps. A positive side effect is that when the accuracy is relaxed, the algorithm to find an optimal combination can be faster. The algorithm does not have to find the best combination. It can already abort the search when it found an acceptable combination. This combination should have almost the same combination it has used with last code.

8.4.3 Technology of the sub-DACs

The combined-DAC output is the sum of the sub-DAC outputs. Therefore, the dynamic performance of the combined DAC is (almost) the same as the dynamic performance of the sub-DACs. Note that the output of the sub-DACs can affect each other, see Chapter 7.1. However, this is expected to be the same as the unit-elements were combined in one DAC.

The linearity improvement method only requires that the sub-DACs are able to be downscaled. However, the performance of the sub-DACs, such as the settling time, charge feed-through and the glitches, do affect the overall performance.
9 Conclusions

This thesis described a new pre-processing method for improving the static linearity of DACs. The improvement is based on explicit reordering of the static DAC errors. The DAC digital input word is optimally distributed among the parallel sub-DACs, which construct the main DAC. The optimal distribution of the input word makes that the mismatch errors of the current sources cancelling each other as much as possible. Simple additional resources and algorithms were used so that full on-chip integration is possible.

The performance of the theoretical linearity improvement method increases with every extra sub-DAC. Four parallel 10-bit sub-DACs were combined to achieve 12-bit resolution. The initial accuracy was at 10-bit level. After optimizing the distribution of the mismatch errors, the accuracy was improved to higher than 14-bit level. A practical method, reducing the required memory resources was developed. Transistor level simulations with optimised distributions were presented. A self-measurement method was presented that used the same sub-DACs with a down-scaled current, an extra 1-bit ADC and 1 LSB current source. The increased mismatch due to the sub-threshold downscaling and the effect of the noise on the measurement accuracy were also examined.

Our approach is especially attractive in combination with FPGAs. The already existing internal digital resources can be used for pre-processing and only a few extra resources are required. Most of the resources can be reused. Our approach will allow DACs with physically smaller current sources, i.e. lower accuracy, because the mismatch errors can be optimally combined and the combined mismatch will be reduced.
10 Recommendations

Memory reduction
The pre-correction method uses a look-up table. A method is presented to reduce the amount of required memory. However, it is likely that the linearity improvement method will only be applied when the additional resources are already present. Further investigation is required to achieve a significant reduction of the amount of required memory. For example, the memory look-up-table only stores the combinations for the even $k_I$s. The odd $k_I$s can be constructed with the even $k_I$s combinations and one optimal 1-lsb source from a sub-DAC. Another example is to only store the error of each current (binary/segmented) source and store them in an increasing order. An algorithm can combine two sources, one with a positive DNL and one with a negative DNL. It is expected that both examples will have a trade-off between the required memory, speed and accuracy.

Self-Measurement
In the transistor level simulations it is shown that the output currents can be reduced adjusting the $V_{gs}$ of the unit-elements. The mismatch errors of the unit-elements functioning in sub-threshold will be increased. In an actual design, it is not sure that the linear accuracy will be sufficient for this self-measurement method. Further investigation is recommended to attain information about the DAC functioning in the sub-threshold range. The presented self-measurement method is able to construct the INLs for all DAC segmentations. The number of measurements with binary and segmented architectures can be reduced by only measuring the DNL of each current-source. Further investigation is recommended to see the affect of this method.

Resolution
An $N$-bit DAC has $2^N-1$ unit-elements and $2^N$ input codes. This principle is applied with the presented linearity improvement method. However, there is mismatch and with $M$ sub-DACs there was a large number of combinations (shown in Figure 4.7a and 4.4c). It is expected that codes can be formed between $k$ and $k+1$. Hence, it is possible that the resolution can be increased. However, at the beginning and the end of the code range, the size of the mismatches and the number of combinations are small. Therefore, the increased resolution may have a larger INL at the beginning and end of the range. Further investigation is recommended.

Randomize the distribution
The linearity improvement method uses an optimized distribution to reduce the mismatch errors. This method requires memory. A pseudo random distribution of the input code to the sub-DACs does not require memory. Randomising the combined output code causes it to become uncorrelated. Thereby it is avoiding harmonic distortion. This might be useful for a different operation mode. The paper of [Chan 2006] describes how this method is used with two 12 bit sub-DACs to create a 14 bit 100Ms/s Nyquist-rate DAC. Further investigation is recommended to apply this method with parallel sub-DACs.
Bibliography


[Doris 2005] K. Doris, J. Briaire, D. Leenaerts, M. Vertregt, A. van Roermund “A 12B 500MS/S DAC WITH >70DB SFDR UP TO 120MHZ IN 0.18µM CMOS”, ISSCC 2005, Session 6, Feb. 2005


Appendix A

Results Matlab simulations
This Appendix presents a small overview of non-linearity of the measure DAC affecting the average INL$_{\text{max}}$ reduction after the improvement method.
The average remaining INL$_{\text{max}}$ compared to the INL$_{\text{max}}$ before the measurement is in the right corner of the figure.

The sub-DAC are measured with a:
- Ideal measurement with ideal characteristic reference
- Quantized measurement
- Downscaled normal DAC (sigma 1.5%) measurement
- Downscaled DAC (sigma 9%) measurement
- Downscaled DAC (sigma 9%) measurement (averaged 2 times).

![Figure A.1 Using the actual INL](image1)
![Figure A.2 Measurement without non-linearity](image2)
![Figure A.3 measurement sigma 1.5%](image3)
![Figure A.4 Measurement sigma 9%](image4)
![Figure A.5 Measurement sigma 9% and 1 averaging](image5)
![Figure A.6 Monte Carlo, sigma 1.5%](image6)
Appendix B

Cadence figures

The pictures in Appendix B are from the simulation program Cadence. Figure B.1 shows a representation of a 1-bit DAC. The main components for one of the current sources of the 10 bit sub-DAC are placed together.

Figure B.1 Schematic 1-bit DAC
Parallel DAC architecture simulated in Cadence.

Figure B.2 shows a parallel DAC architecture.

The input code is distributed to the sub-DACs and the outputs are connected to each other.
Appendix C

Micrograph of a DAC Chip with smaller current sources.

Figure D.1 shows an example of a micrograph of a DAC chip [Radulov 2005]. The unit-elements are grouped into current-sources M1. Note that this architecture uses calibration and due to this, the size of the unit-element array is reduced.

Figure D.1 Micrograph of a DAC chip
This appendix contains the ProRisc Paper published in November 2005.

A parallel current-steering DAC architecture for flexible and improved performance.
A parallel current-steering DAC architecture for flexible and improved performance


Mixed-signal Microelectronics Group, Eindhoven University of Technology, Electrical Engineering, Eindhoven, The Netherlands, Email: HovenR@gmail.com

Abstract— This paper presents a new current-steering DAC architecture for flexible and improved performance. This flexible DAC architecture is based on fixed entities: sub-DACs. They are nominally identical and operate in parallel, which results in improved and flexible performance, delivered in several modes of operation (OP). One OP mode is using the sub-DACs as independent converters. Another option is using them together for higher conversion resolution and accuracy. This paper concentrates on a particular OP mode, which through distributing the input digital word among the parallel sub-DACs, achieves cancellation of the mismatch errors. This technique leads to improved static linearity, whereas the improvement depends on the occupied pre-processing resources. The proposed technique can be fully integrated on-chip, as it relies on a 1bit ADC and makes reuse of already existing resources.

Index Terms—Digital-to-Analog converters, Flexibility

I. INTRODUCTION

Current-steering DACs are systems composed of three main parts. These are the digital control part, the analog unit elements, and the switches in the mixed signal part, see Figure 1. In conventional approaches, all these three parts are fixed. Thus, the performance is fixed and unique. Once a conventional DAC is manufactured, it has a certain resolution and accuracy. Its non-linearity is unique, because the mismatch errors of its current sources and their distribution are unique.

The work is sponsored by Stichting Technische Wetenschappen.

The first two approaches reduce the mismatch errors, while the third approach alters the distribution of the errors. Important advantages of digital pre-processing include that:

a) the D/A process is intact;

b) the needed extra resources are in the digital domain.

Our approach uses an architecture based on parallel sub-DACs, shown in Figure 2, [1-3]. Extra digital pre-processing controls the fixed sub-DACs. The sub-DACs can be used in different OP modes and hence the resolution and accuracy are no longer fixed but flexible. Note that the flexibility in the digital part affects the whole DAC without a need for major changes in the analog and mixed-signal parts.

In this paper, a technique is presented to improve the linearity of the parallel DAC system. In section II the definition of a parallel DAC architecture and the principles for linearity improvement are presented. Section III presents a self-measurement and self-correction method. The algorithms and required memory are explained in section IV and V. Finally, simulation results and conclusions are given in Section VI and VII.

II. PARALLEL DAC ARCHITECTURE

Some of the OP modes combine the sub-DAC outputs to construct one single higher resolution DAC, as shown in Figure 2.

Fig. 1. DAC with grouped unit elements.

Popular approaches to improve this unique static performance include:

a) intrinsically accurate current sources;

b) calibration;

c) digital pre-processing.
When the end-user requires higher accuracy, our flexible DAC can distribute the digital input code among its sub-DACs, in such a way that their individual mismatch errors cancel each other. Note that the outputs of the sub-DACs have to be combined at the outputs of the chip.

With a single DAC there is only one output combination for each input code. When two DACs are used the number of output combinations increases. For example, with two sub-DACs there are three combinations to construct digital code 2, etc., Table 1 shows the possible combinations to construct the analog output for all digital codes in the case of two parallel 11 bit sub-DACs.

The number of combinations with two sub-DACs (shown in Table 1 and Figure 3a) for digital code \( k \), with \( k \) from zero to a half-scale is:

\[ C_{k2} = k + 1 \]  

(2)

The number of combinations with \( k \) from a half- to full-scale is:

\[ C_{k2} = (\text{fullscale} - k) + 1 \]  

(3)

The total number of possible combinations with "M" multiple N-bit sub-DACs is:

\[ C_M = \sum_{k=0}^{M} C_k = (2^N)^M = 2^{M \cdot N} \]  

(4)

To construct the combined DAC transfer characteristic two sub-DACs have \( 2^{2N} \) combinations to choose from for \( 2^{N+1} \) codes (according to equations 4 and 1). When for each code the combination with the best linearity is selected, the INL_{max} (the INL error for code \( k \)) is equal or smaller than with a random/fixed combination. This is shown for four sub-DACs in Figure 4b.

With each additional sub-DAC the number of combinations, and hence the chance to find a better one increases, as shown in Figure 3b and 3c. The total number of combinations for 3 and 4 N-bit sub-DACs are \( 2^{3N} \) and \( 2^{4N} \) (Equation 4) and there are only \( 3 \cdot 2^{N-2} \) and \( 4 \cdot 2^{N-3} \) codes to construct (Equation 1). Note that the INL improvement will be less at the ends of the transfer characteristic, because there the number of possible combinations is smaller. At half-scale there are the most possible combinations and thus the possibility to create one for a better linearity is higher.

The difference in linearity with a fixed and flexible architecture is shown in Figure 4. For each simulation an 8-bit DAC is constructed with four 6-bit sub-DACs. In Figure 4a, the control of the sub-DACs is fixed, i.e. the distribution of the digital codes is not optimized for improved linearity. Like in the normal DACs, the highest statistical possibility of locating INL_{max} is at mid-scale. Figure 4b shows the INL of the same 8-bit DAC but with full flexibility in the control-part and linearity optimization through choosing the best code combination to generate as small INL error as possible. With four sub-DACs a combination can be found that creates small INL_{k} errors.

With the 700 (Monte Carlo) simulations shown in Figure 4b this happened only once around code 220.

To improve the linearity of the D/A transfer characteristic the transfer characteristics of the sub-DACs are acquired. The best
combinations of “sub-INLk’s” can reduce the nonlinear effect caused by the mismatches. This will improve the overall INL, as shown in Figure 4b.

Fig. 4. INL improvement with four sub-DACs in a fixed and flexible architecture.
(a) INLs of fixed DACs, (b) INLs of flexible DACs with the same unit elements as the fixed DACs, (c) Number of possible combinations for each code.
Monte Carlo simulation.

III. PARALLEL SELF-MEASUREMENT AND SELF-CORRECTION

Our parallel architecture has four sub-DACs, one extra current source, a 1-bit ADC and some extra switches. The INLk's of the sub-DACs have to be determined and stored before a good combination of the sub-DACs can be made. The INLk's can not be directly measured because there is no ideal transfer characteristic to compare with. That is why the DNLk's are measured and then INLk's are determined as:

\[ INL = \sum_{j=1}^{k} DNL_k \]  \hspace{1cm} (5)

SELF-MEASUREMENT

To determine the INL of sub-DAC A its DNLk's are measured. DAC A and B will start with the same code and the difference between the outputs will be “measured” with the DAC C, shown in Figure 5.

The output of DAC-C is downscaled; this means that the output at each code is smaller than the output of DAC A or B. With the 1-bit-ADC it can be detected when the output currents from DAC B+C are sufficiently equal to DAC A. The value of DAC C will be stored in a temporary memory. Next, the code at DAC A will be increased by one, code B stays the same and the extra 1-LSB current source will be switched on, see Figure 5. The difference between the outputs of sub-DAC A and B will be determined again. This difference and the value in the temporary memory are subtracted. Based on this the INLk can be calculated, see (5). This will be done for each code and sub-DAC. The INLk's are stored in a memory and with this an algorithm (in Section IV) will calculate the best combination of the sub-DACs for each output code.

This self-measurement method uses an extra 1-LSB current-source and this one is not always equal to the average of the DAC current sources. This will introduce an error that is accumulated and affects the stored sub-INLk's. This error can be determined because the INLk's at full-scale of each sub-DAC should together be zero. If not zero, there is an offset. This offset will affect all the measured sub-INLk's and the stored INLk's will be adjusted with the calculated offset.

The measurement-DAC C determines the difference between the output of DAC A and B. Therefore the steps of DAC C have to be smaller than this difference. However DAC C also has to be able to determine the largest possible difference. For example, when the mismatches in each sub-DAC can cause an INL of +/- 2 LSB, DAC C has to be able to measure a maximal difference of +/- 4 LSB. Since in this example single-ended-DACs are used, the negative measurements are done by adding DAC C to DAC A, instead of to DAC B. The INL and quantization errors of DAC C are also accumulated with this method (Equation 5). When the output of DAC C becomes smaller, the accumulation also becomes smaller.

In the simulation, shown in Figure 8, DAC C is constructed with two smaller downscaled-DACs and is able to measure the difference of +/- 4 LSB and has small steps, shown in Figure 6. The first “rough” measure-DAC has 2^N steps and the maximal output is downscaled to 8 normal LSBs. The second “fine”-DAC can be maximally downscaled to 2 LSB of the rough-DAC:

\[ LSB_{rough} = \frac{8}{2^N} = 2^{-(N-3)} LSB_{normal} \]
\[ LSB_{fine} = \frac{LSB_{rough}}{2^{N-1}} = 2^{-(N-4)} LSB_{normal} \]  \hspace{1cm} (6)

The measurement-DAC C determines the difference between the output of DAC A and B. Therefore the steps of DAC C have to be smaller than this difference. However DAC C also has to be able to determine the largest possible difference. For example, when the mismatches in each sub-DAC can cause an INL of +/- 2 LSB, DAC C has to be able to measure a maximal difference of +/- 4 LSB. Since in this example single-ended-DACs are used, the negative measurements are done by adding DAC C to DAC A, instead of to DAC B. The INL and quantization errors of DAC C are also accumulated with this method (Equation 5). When the output of DAC C becomes smaller, the accumulation also becomes smaller.
The output current in the simulation of the rough-measure-DAC is added to the output current of DAC B and approaches the output of DAC A. The ADC detects when this current becomes larger or smaller than the output of DAC A. The remaining difference will be determined by the fine-measure-DAC. In the simulation, shown in Figure 8, 10-bit sub-DACs are used to construct one 12-bit DAC. The rough-measure-DAC is compared to the normal DAC $2^7$ times downscaled and the fine-measure-DAC $2^{14}$ times.

![Fig. 6. Measure DAC-C constructed with a rough and fine sub-DAC.](image)

The sub-DACs in a flexible parallel architecture can be used as multiple separate DACs or all/some combined together. When M sub-DACs work together the maximal -combined- output must be the same as the maximal output of a separately working sub-DAC. The individual output currents of the M combined sub-DACs must be M times smaller than normal. When the scalability of the sub-DACs is flexible, the sub-DACs can be used as the rough and fine-measurement-DACs as well. The extra acquired components for this method are the 1-bit ADC, 1-LSB current source, some switches and of course the memory and digital resources for the algorithms.

IV. ALGORITHMS TO FIND THE BEST COMBINATION

The best transfer characteristic of the combined DAC can be constructed by using the INLk’s of the sub-DACs to find the minimal INL. The best overall INL will be constructed when all the minimal INLk’s of the overall code are found. To find the minimal overall INL (INL_{overall}) all combinations of the sub INLk’s have to be evaluated and every time it finds a INL_{overall} that is smaller than the INL_k stored in the memory (at code kT), the stored INL_k will be replaced with the new INL_{overall}. Also for each input code kT the sub-codes of the sub-DACs will be stored in the memory as a lookup table. The total number of sub-DAC combinations to evaluate with four sub-DACs is $2^{4N}$, according to (4).

For example with four 10-bit sub-DACs there are $2^{40}$ combinations. With a process that can even evaluate 100,000 combinations per second; the initialization of this DAC would still take 4 months.

A faster algorithm has to evaluate fewer combinations. If some combinations are not evaluated and a few of these combinations would be the best one, these will not be found. However it is expected that with “well chosen” combinations a close to optimal INL_{overall} will be found. By only actively using two of the four sub-DACs, $2^{2N}$ combinations can be evaluated (Equation 4). With four sub-DACs there are six options to couple them. These six options are the combinations of sub-DACs “1&2, 1&3, 1&4, 2&3, 2&4 and 3&4”. When all combinations of those two-coupled-DACs are evaluated, it is expected to find combinations with a small INL_{overall}. It is clear that with only two active sub-DACs not all of the four sub-DAC current-sources are used. The combined DAC can only use half of the entire code range. To solve this problem the outputs of the passive sub-DACs can be switched to zero or maximal and the active sub-DACs will evaluate the combinations again to find the small INL_{overall}. At the middle of the scale either the first or the second passive sub-DAC can be switched on. This will cause that the number of options to combine the sub-DACs with becomes twice as large at the middle of the scale. The options to combine the sub-DACs are there twelve instead of six times. The middle of the scale options are only used at ¼ of the total code range. The average coupling options for the total code range is:

$$\text{Options}_{\text{coupling}} = 6 \cdot \frac{3}{4} + 12 \cdot \frac{1}{4} = (6 \cdot 1\frac{1}{4}) = 7.5 \quad (7)$$

The simulated results with this algorithm are still good (shown in Figure 8b) and the total number of combinations for the full range is:

$$C = \text{Options}_{\text{coupling}} \cdot \text{Comb}_{\text{DACcouple}} = 15 \cdot 2^{2N} \quad (8)$$

![Fig. 7. Number of combinations with four 7bit sub-DACs](image)

The simulation, clearly shows peaks at the position where no good INL can be found (Figure 8b). These are at beginning, the end and at the middle. The missing combinations are the main reason of the extra peaks at the middle. It is expected that using an algorithm with more combinations can solve the problems with the peaks at the middle. It is possible to reduce some of the peaks at the beginning and end by using all combinations of four DACs.
With this method the memory is decreased by two-thirds. There is also a temporary memory to store the INL's of the four sub-DACs (See Section IV),

\[ M_{\text{INL,sub}} = \text{subcodes} \cdot n_{\text{DACs}} \cdot N_{\text{subDACs}} \]

\[ M_{\text{INL,sub}} = 2^{10} \cdot 4 \cdot 10 = 40.96\text{kbit} \]  

(11)

The calculated INL's of the combined DAC should also be stored temporarily. However with the two active sub-DACs method it is possible to evaluate all the combinations for each code \( k \) before going to the next code. The same 14-bit memory can be reused and this size is negligible.

The total amount of memory this approach requires is 95kbit. When the lookup table of the DAC is initialized, the 41kbit temporary memory is not needed any more and can be reused.

The size of memory will only become smaller in the future. The new generations FPGAs already have up to 50Mbit internal memory. When parallel DACs are added to these FPGAs, the internal memory and digital resources can be used for the algorithms.

V. PRE-PROCESSING SELF-CORRECTION

The algorithm of Section IV evaluates the combinations to construct a transfer characteristic and stores the best sub-DAC combinations in the memory as a lookup-table. For each digital input-code the control-part (shown in Figure 2) gets the corresponding sub-codes from the lookup-table and sends these to the sub-DACs.

With four 10-bit sub-DACs, there are \((4 \cdot 2^{10} - 3) = 4093\) input codes (Equation 1). For each input-code the best sub-code combination of the sub-DACs is stored in a memory, this is called the pre-correction lookup table. The memory for the lookup table with four active sub-DACs is:

\[ M_4 = \text{Inputcodes} \cdot M_{\text{subDACs}} \cdot N_{\text{subDACs}} \]

\[ M_4 = 4093 \cdot 4 \cdot 10 = 163.7\text{kbits} \]  

(9)

The size of the memory can already be decreased by 25% by calculating the fourth sub-DAC code from the input-code and subtracting the other sub-DAC codes. The algorithm of Section IV has only two active DACs and the sub-code of the two passive DACs are zero or maximal, depending on the input-code. The sub-codes of the second active sub-DAC can be calculated from the first one and the input-code, since the third and fourth sub-code are also calculated from it. The lookup table has to remember what are the active and passive DACs are and the sub-codes of the first active DAC. For all input codes there are maximal 12 and on average 7.5 options to combine the active and passive sub-DACs, (according to equation 7). The memory of the lookup table with two active sub-DACs is:

\[ M_2 = \text{Inputcodes} \cdot (n_{\text{ActiveDACs}} - 1) \cdot (N + \log_2 \text{options}) \]

\[ M_2 = 4093 \cdot (2 - 1) \cdot (10 + 3) = 54\text{kbit} \]  

(10)

With this method the memory is decreased by two-thirds. There is also a temporary memory to store the INL's of the four sub-DACs (See Section IV),

\[ M_{\text{INL,sub}} = \text{subcodes} \cdot n_{\text{DACs}} \cdot N_{\text{subDACs}} \]

\[ M_{\text{INL,sub}} = 2^{10} \cdot 4 \cdot 10 = 40.96\text{kbit} \]  

(11)

The calculated INL's of the combined DAC should also be stored temporarily. However with the two active sub-DACs method it is possible to evaluate all the combinations for each code \( k \) before going to the next code. The same 14-bit memory can be reused and this size is negligible.

The total amount of memory this approach requires is 95kbit. When the lookup table of the DAC is initialized, the 41kbit temporary memory is not needed any more and can be reused.

The size of memory will only become smaller in the future. The new generations FPGAs already have up to 50Mbit internal memory. When parallel DACs are added to these FPGAs, the internal memory and digital resources can be used for the algorithms.

VI. SIMULATION RESULTS

The simulation with the technique that has two active and static sub-DACs has an average improvement of 97.5% (shown in Figure 8) with a thermometer sub-DAC architecture and 96.5% with a binary sub-DAC architecture. The 1% difference is caused by the larger INL accumulation of the measure DAC with the measurement of the binary sub-DACs. Even with smaller average improvement of the binary architecture, the linear accuracy improvement is:

\[ L_{\text{improvement}} = \log_2 \left( \frac{100}{100 - 96.5} \right) = 4.8\text{bit} \]  

(12)

When unit elements with a certain mismatch are used in a fixed architecture, this would have a 10-bit linear accuracy.

The unit elements with the same mismatch used in a flexible architecture, a linear accuracy of more than 14-bit can be achieved.

The technique that uses only two active sub-DACs has reduced the total number of combinations more than \(2^{17}\) times.

VII. CONCLUSION

This paper showed how DAC static linearity can be improved through explicit reordering of the mismatch errors. Parallel sub-DACs are used and the digital input word is optimally distributed among them, so that the mismatch errors cancel each other as much as possible. Simple additional resources and algorithms were used so that a fully integrated solution is possible.
An example of the main concept was shown. Four parallel 10-bit sub-DACs were combined to achieve 12-bit resolution. The initial accuracy was at 10-bit level. After optimizing the distribution of the mismatch errors, the accuracy was improved to higher than 14-bit level.

Our approach is especially attractive in combination with FPGAs. The already existing internal digital resources can be used for pre-processing and only a few extra resources are required. Most of the resources are reused. Our approach will allow DACs with physically smaller current sources, i.e. lower accuracy, because the mismatch errors can be optimally combined and will cancel each other as much as possible.

REFERENCES