# Design of a digitally pre-corrected DAC with built-in self-measurement

de Meulmeester, J.M.

*Award date:*
2005

TU/e technische universiteit eindhoven

# Design of a digitally pre-corrected DAC with built-in self-measurement

J.M. de Meulmeester

*June 30, 2005*

**Master of Science thesis**

Project period: September 2004 - June 2005

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

Supervisors:
Ir. P.J.A. Harpe
Dr. Ir. J.A. Hegt
Prof. Dr. Ir. A.H.M. van Roermund

# Abstract

This thesis presents an approach for the design of current-steering digital-to-analog converters (DACs), yielding high static performance and reliability.

As a consequence of device mismatch, the actual output value of the converter will mostly deviate from the the nominal output value, thereby decreasing its performance. This problem could be solved by either extensive intrinsic design of the analog core, or by calibration and/or correction techniques. In this thesis, a digital pre-correction technique is presented, which is able to approximate the desired output value, showing near-ideal behavior. This yields improved static performance and reliability. Besides a pre-correction algorithm, a DAC core with built-in redundancy and a measuring circuit are required for the implementation of this correction technique. All required components are discussed in detail.

A design of a digitally pre-corrected 12-bit DAC with built-in redundancy and self-measurement, based on the given approach is presented. Both system-level and circuit-level simulations are performed, in order to verify the achieved performance. The simulation results show that the presented approach is successful, by achieving high static performance while using small, inaccurate unit current cells. Furthermore, the static performance and reliability is improved in comparison to an intrinsic 12-bit binary-weighted DAC.

# Contents

# List of symbols

| Symbol | Description |
|---|---|
| $I_j$ | Output current of the $j$-th current source |
| $\alpha_j$ | Weighting factor of the $j$-th current source |
| $I_{unit}$ | Output current of the unit current source |
| $\frac{\sigma_{unit}}{I_{unit}}$ | Relative spread of the unit current source |
| $\mathbf{I_j}$ | Random variable $\mathbf{I_j}$ |
| $\mu_{\mathbf{I_j}}$ | Mean value of random variable $\mathbf{I_j}$ |
| $\sigma_{\mathbf{I_j}}$ | Standard deviation of random variable $\mathbf{I_j}$ |
| $N$ | Resolution of the D/A converter |
| $k$ | Number of current sources |
| $X$ | Digital input code |

# Chapter 1

# Introduction

*In this chapter, an introduction to the master project is given, including background information and goals of the project. Furthermore, an overview of the remaining chapters is provided.*

## 1.1 Project background

Technology development and trends in system design have led to the development of mixed-signal systems-on-chip (MSSoC). These integrated circuits have a large range of applications, as they combine both analog and digital subsystems on a single chip. Unfortunately, the design of MSSoCs is hard, mostly due to complicated analog design. The complexity of the design may be reduced by relaxing the analog constraints. However, this will adversely affect the performance of the system. This performance drop may be counterbalanced by an appropriate algorithm which improves the system's performance. The implementation of such an algorithm is made affordable by the on-chip digital processing power, as a consequence of MSSoC. As this approach takes full advantage of the integration of analog and digital systems on a single chip, it provides a good starting point for a smart approach. This master project is part of the 'Smart AD/DA' project, in which A/D and D/A converters are designed using a smart approach.

## 1.2 Project goal

The goal of this project is to design a current-steering DAC using a smart approach: the required static accuracy should not be achieved by intrinsic design of the analog core, but by a digital algorithm that optimizes the static properties. In order to succeed, a number of components have to be developed, including a digital pre-correction algorithm, a DAC core with built-in redundancy, and a measurement circuit. The combination of the aforementioned components should yield an improved performance and reliability. The performance of the DAC is verified by simulations on both system and circuit level. Note that this project focusses on the improvement of static performance.

## 1.3 Chapter overview

In the next chapter, the basic principles of digital-to-analog conversion are discussed, including an overview of current-steering DAC architectures. Chapter 3 describes a number of static and dynamic specifications, intended to characterize the performance of a DAC. Chapter 4 describes a number of design issues which are important for the design of accurate D/A converters. In chapter 5, the design approach for D/A converters is presented and the issues of chapter 4 are challenged. Chapter 5 ends with a design example on system level. Chapter 6 presents the circuit level design of a digitally pre-corrected DAC, followed by its simulation results. Finally, conclusions are drawn and recommendations for further research are given.

# Chapter 2

# Current-steering D/A converters

*In this chapter, the basic principles of digital-to-analog converters are discussed, with great emphasis on current-steering architectures.*

## 2.1 Basic principles of digital-to-analog conversion

Digital-to-analog converters (DACs) provide the link between the digital world of signal processing and our analog world. They become increasingly important as signal processing in the digital domain gains in popularity. A typical digital signal processing system is shown in figure 2.1.

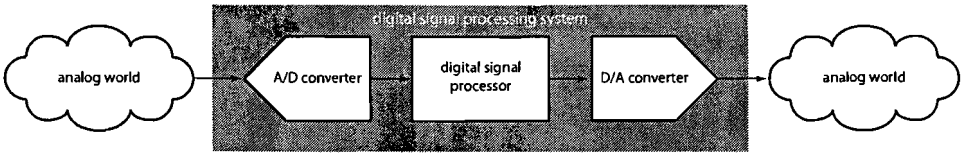



*Figure 2.1: An example of a digital signal processing system*

The basic function of a DAC is to convert a digital input code in a corresponding analog output signal. This can be described by the following equation:

$$A_{out} = X \cdot A_{ref} \tag{2.1}$$

From (2.1) it can be seen that each digital input code $X$ generates a multiple of a reference value $A_{ref}$ at the analog output $A_{out}$. The reference value may be one of the three electrical quantities: voltage, current or charge. The digital input code $X = \{b_{N-1},\ b_{N-2},\ \ldots,\ b_1,\ b_0\}$ is constructed by N digital bits, where $b_{N-1}$ is the most significant bit (MSB) and $b_0$ the least significant bit (LSB). Figure 2.2 shows the transfer characteristic of an ideal 3-bit single-ended DAC.

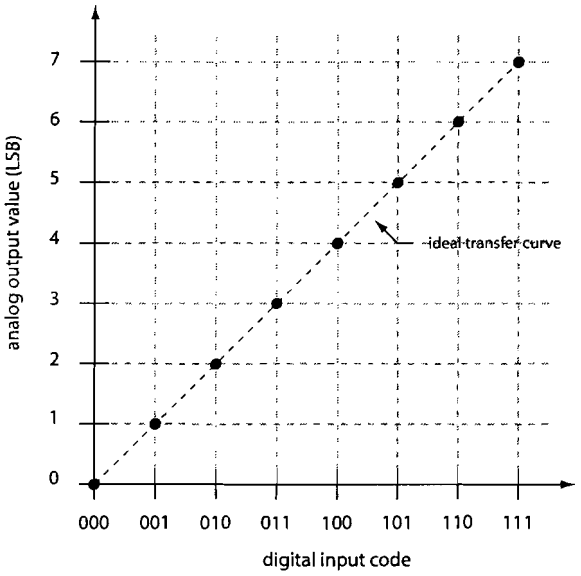



*Figure 2.2: Ideal transfer characteristic of a 3-bit single-ended DAC*

There are several different ways in which a DAC can be implemented. Various DAC architectures are known from literature: R2R ladder based converters, switched capacitor converters, current-steering converters, etc. For this project a current steering architecture is selected. The properties of a current-steering DAC will be discussed in the following section.

## 2.2 Current-steering DAC architectures

Basically, a current-steering DAC consists of a number of weighted current sources, which are combined according to the digital input code. The number of current sources is related to the number of bits $N$. The summation of currents is performed by switches which connect the outputs of the current sources to the converter's output node. The switches are controlled by the input code. This is a straightforward approach, since currents are easy to weight, sum and switch. The main advantage of current-steering DACs compared to other architectures is the achievable speed. Since current-steering DACs can drive resistive loads directly, no high-speed buffer at the output of the converter is required. This is advantageous, because an output buffer will limit the converter's performance with respect to speed. In addition, savings on chip area and design effort can be realized. Various current-steering architectures are known from literature. In the following sections binary-weighted, thermometer-coded and segmented converters are mentioned.

### 2.2.1 Binary-weighted current-steering DAC

In a binary-weighted architecture the DAC is composed of $N$ current sources, where $N$ is its resolution. The output current $I_j$ of the current source which is controlled by bit $b_j$, i.e. the $j$-th current source, is $2^j$ times larger than the output current associated with the LSB. This may be achieved by connecting $2^j$ unit current sources in parallel. The above-mentioned is denoted by:

$$I_j = 2^j \cdot I_{unit} \ , \tag{2.2}$$

where $I_{unit}$ is the output current of a single unit current source. The analog output value is given by:

$$I_{out} = I_{unit} \cdot \sum_{j=0}^{N-1} 2^j \cdot b_j \ , \tag{2.3}$$

where $b_i \in \{-1, 1\}$ , $0 \leq j < N$ considering a differential topology. A differential architecture achieves a better dynamic performance, compensates for a number of imperfections, and reduces the sensitivity for power supply variations. The main advantage of the binary-weighted architecture is its simplicity. Furthermore, it is a compact, area-saving architecture compared to other topologies. However, a binary-weighted DAC is sensitive to device mismatch, which could result in large DNL errors and non-monotonic behavior (see section 3.1.3). Another drawback of this architecture is its large glitch energy. This is a major limitation during high-speed operation. To achieve monotonicity and reduce the influence of glitches, a thermometer-coded architecture or a segmented converter could be used. A circuit diagram of a differential binary-weighted current-steering DAC is shown in figure 2.3.

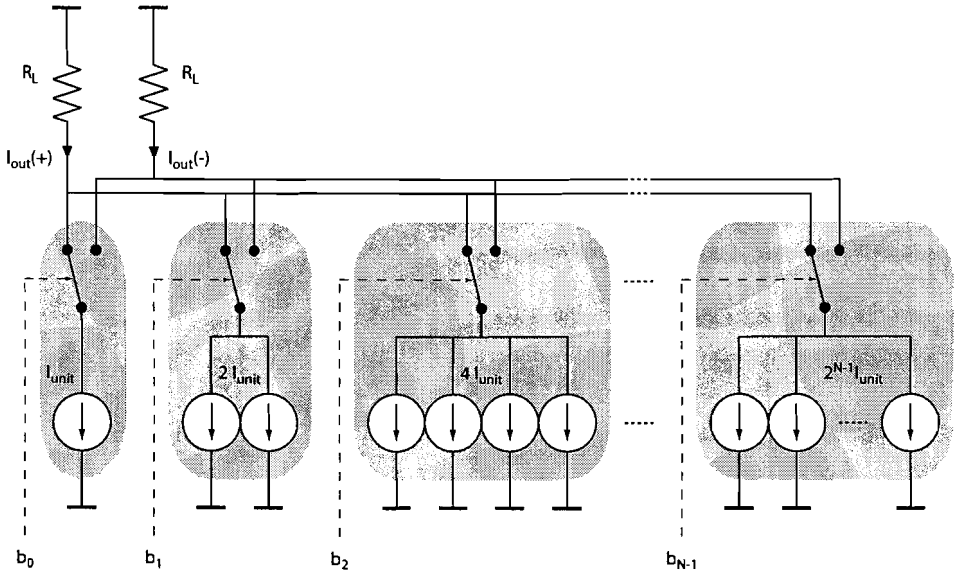



*Figure 2.3: Circuit diagram of a binary-weighted current-steering DAC*

### 2.2.2 Thermometer-coded current-steering DAC

A thermometer-coded current-steering DAC architecture consists of $M = 2^N - 1$ equal-sized independent unit current sources, utilized to generate the analog output signal. The $N$-bits binary input code is converted in a $M$ bits thermometer code, which controls the $M$ independent current sources. The conversion is performed by a so-called binary-to-thermometer converter. Equation (2.4) denotes the analog output value of a thermometer-coded current-steering DAC:

$$I_{out} = I_{unit} \cdot \sum_{j=0}^{M-1} d_j \ , \tag{2.4}$$

where $d_j \in \{-1, 1\}$ , $0 \leq j < M$ are the thermometer-coded bits supposing a differential architecture. A major advantage of the thermometer-coded topology is its guaranteed monotonicity. This follows directly from the converter's transfer function, given by (2.4). When the input code is incremented by one, exactly one current source is added to the set which was generating the foregoing analog output value. Other advantages of the thermometer-coded architecture are low DNL errors and a reduced glitch area compared to a binary-weighted approach. However, a thermometer-coded converter is complex. For example, a 12 bits thermometer-coded DAC requires 4095 unit current sources and 4095 switches. Besides the large, additional amount of switches compared to a binary-weighted architecture, a binary-to-thermometer converter is required. Figure 2.4 shows a circuit diagram of a differential thermometer-coded current-steering DAC.

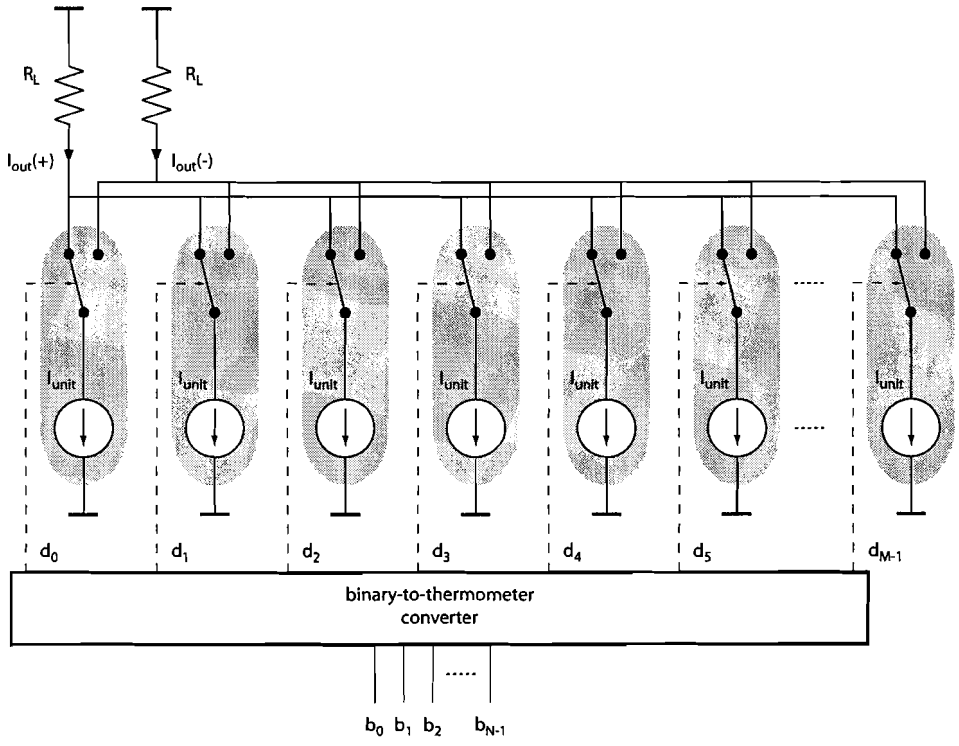



*Figure 2.4: Circuit diagram of a thermometer-coded current-steering DAC*

### 2.2.3 Segmented current-steering DAC

The majority of recent DAC implementations employ the advantages of both binary-weighted and thermometer-coded architectures by combining them in a hybrid structure, often referred to as a segmented DAC. Basically, segmented DACs split the digital input code in two fields: a MSB field and a LSB field. The MSB field, converted by the thermometer-coded segment, is a coarse quantized representation of the input code. The LSB field, which is converted by the binary-weighted segment, takes care of the fine quantization. The proportion between the MSB and LSB fields is indicated by the term segmentation: 0% segmentation stands for a fully binary-weighted converter, whereas 100% segmentation is used for fully thermometer-coded converters. The analog output is the sum of the coarse and fine segments. Equation (2.5) denotes the analog output value of a segmented current-steering DAC:

$$I_{out} = I_{unit} \cdot \left( \sum_{i=0}^{P-1} 2^i \cdot b_i + 2^P \cdot \sum_{j=0}^{Q-1} d_j \right) , \tag{2.5}$$

where $P = N(1-s)$, $Q = 2^{Ns}-1$, $N$ represents the number of bits, and $s$ denotes the segmentation. The selection of the segmentation becomes a trade-off between performance, complexity and chip-area. A circuit diagram of a differential segmented current-steering DAC is shown in figure 2.5.

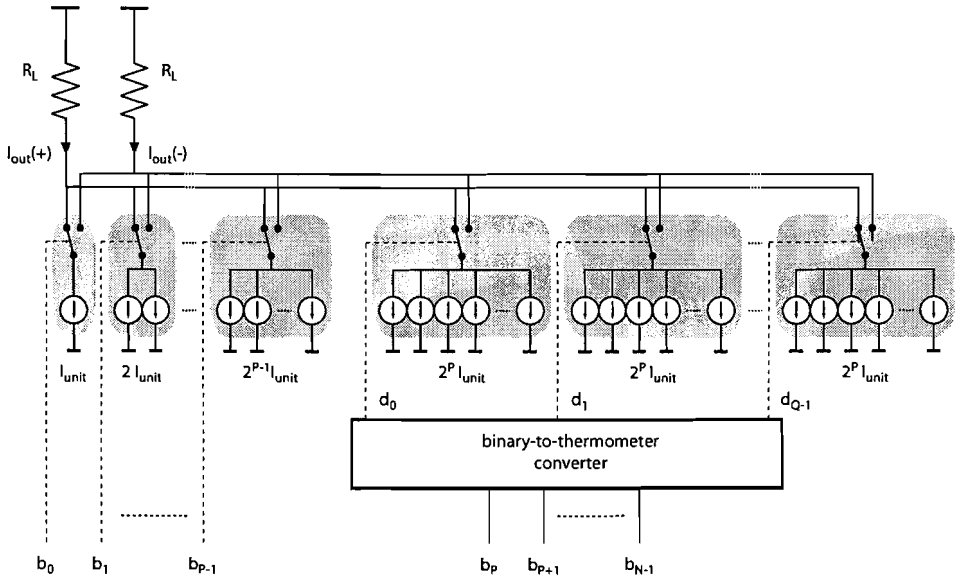



*Figure 2.5: Circuit diagram of a segmented current-steering DAC*

# Chapter 3

# Converter specifications

*In this chapter, a number of static and dynamic specifications are introduced, in order to characterize the performance of a D/A converter.*

## 3.1 Static specifications

Static properties are derived from settled output values, and are often too optimistic to determine the true performance of the converter. Static performance can be specified by several parameters. In this thesis, the following metrics are considered: offset error, gain error, integral non-linearity (INL) error and differential non-linearity (DNL) error. In all discussions a single-ended D/A converter is assumed.

### 3.1.1 Offset error

Basically, all practical converters suffer from an offset voltage and/or offset current, which is caused by the finite matching of components. The offset results in a non-zero output value although a zero input signal is applied to the converter. The offset error, as shown in figure 3.1, is defined as the global shift of the transfer characteristic with respect to the ideal transfer curve. It is expressed in LSB units, where 1 LSB is defined as the nominal full-scale output range divided by the number of feasible values minus one.

### 3.1.2 Gain error

The gain error characterizes divergence of the actual transfer characteristic slope from the ideal slope, resulting in a deviation of the nominal output value when a full-scale input signal is applied. The gain error, as shown in figure 3.2, is defined as the difference between the nominal and actual end points on the transfer characteristic after the offset error has been corrected to zero. Like the offset error, the gain error is expressed in LSB units.

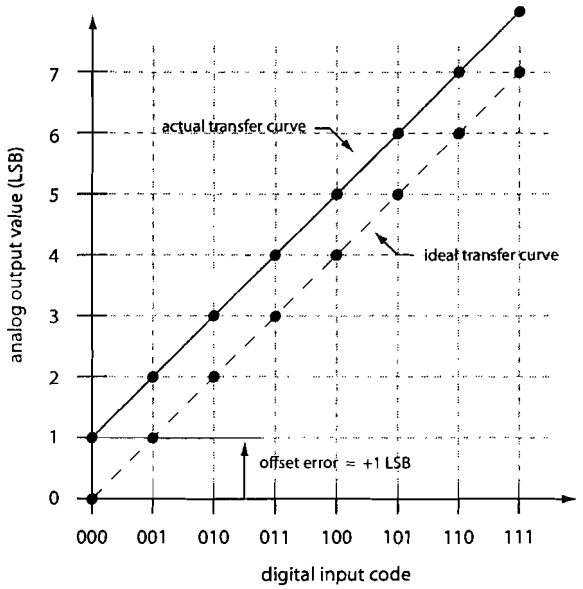



*Figure 3.1: Offset error*

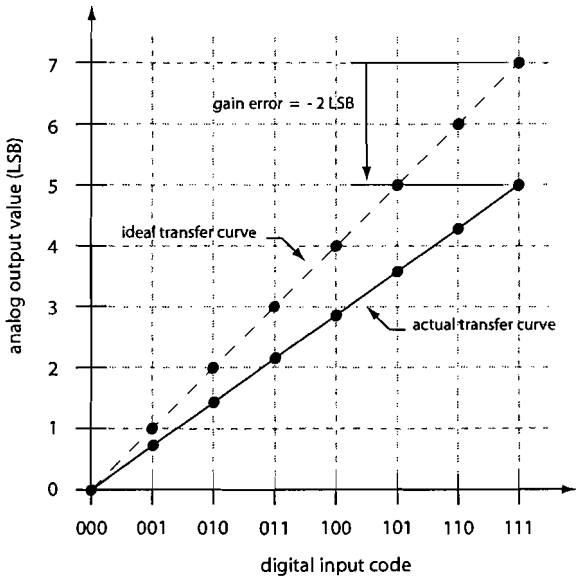



*Figure 3.2: Gain error*

### 3.1.3 Differential non-linearity (DNL) error

Differential non-linearity (DNL) expresses the irregularities between consecutive output levels, after offset and gain errors are nullified (see figure 3.3). The DNL is evaluated in

LSBs and denoted by:

$$DNL(X) = A_{out}(X) - A_{out}(X-1) - 1 \text{ LSB} \quad [\text{LSB}] \,, \tag{3.1}$$

where $DNL(X)$ is the diffential non-linearity error for input code $X$, and $A_{out}(X)$ is the analog output value corresponding to input code $X$. Basically, the worst case DNL error is specified. This is expressed by:

$$DNL_{max} = \max\{|DNL(X)|\} \quad [\text{LSB}] \,, \qquad X \in [1, 2^N - 1] \tag{3.2}$$

where $N$ is the number of bits. Sometimes, negative steps between two consecutive codes are experienced. This large DNL error, called non-monotonicity, deserves special attention as it may cause serious problems in circuits where a converter closes a feedback loop. Non-monotonicity is similar to a local sign change of the transfer characteristic slope, jeopardizing stability.

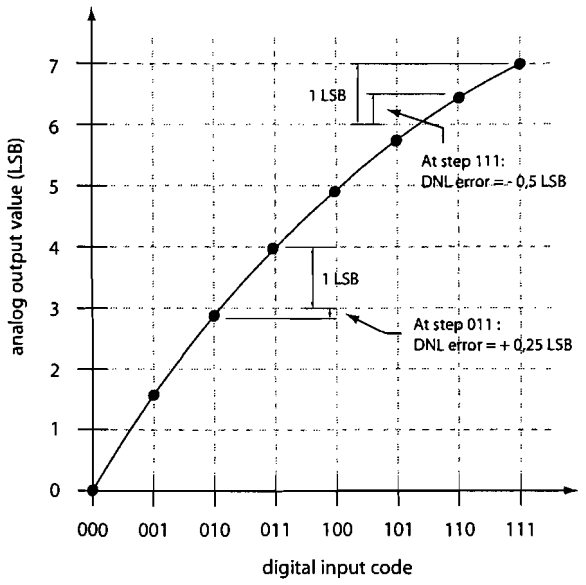



*Figure 3.3: Differential non-linearity (DNL) error*

### 3.1.4 Integral non-linearity (INL) error

Integral non-linearity (INL) expresses the total deviation of the actual output value in relation to the nominal output, after offset and gain errors are nullified. Therefore, the difference between the actual transfer characteristic and the ideal transfer curve is measured. The ideal transfer characteristic may be either the straight line joining the first and last points, as shown in figure 3.4, or the linear regression of the transfer curve. In this discussion the former option is considered. The INL error is expressed in LSB units according to the following equation:

$$INL(X) = A_{out}(X) - A_{out}(0) - X \text{ LSB} \quad [\text{LSB}] \,, \tag{3.3}$$

where $INL(X)$ is the integral non-linearity error for input code $X$, $A_{out}(X)$ is the analog output value corresponding to input code $X$. The worst case INL error is denoted by:

$$INL_{max} = \max\{|INL(X)|\} \quad [\text{LSB}] \,, \qquad X \in [0, 2^N - 1] \tag{3.4}$$

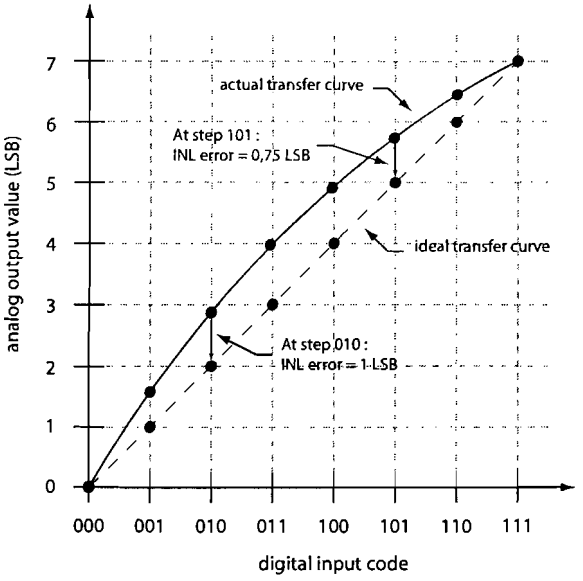



*Figure 3.4: Integral non-linearity (INL) error*

## 3.2 Dynamic specifications

Dynamic properties are derived from the complete signal, including settling times, glitches, charge feedthrough, etc. Therefore, especially during high-speed operation, it is mostly the dynamic performance that determines the real performance of the converter. There are many dynamic specifications known from literature. Here, Signal-to-Noise and Distortion Ratio (SNDR) and Spurious Free Dynamic Range (SFDR) are considered.

### 3.2.1 Signal-to-Noise and Distortion Ratio

The signal-to-noise and distortion ratio (SNDR), usually expressed in decibels, is defined as the ratio between the power of the fundamental and the total noise and distortion power. The SNDR is denoted by:

$$SNDR = 10 \log_{10} \frac{P_s}{P_n + \sum_{k=2}^{\infty} P_k} \quad [\text{dB}] \,, \tag{3.5}$$

where $P_s$ is the signal power, $P_n$ is the noise power, and $P_k$ is the power of the k-th harmonic.

### 3.2.2 Spurious Free Dynamic Range

The spurious free dynamic range (SFDR) is defined as the ratio between the signal power and the power of the largest spurious spectral component within the specified frequency band (see figure 3.5). The SFDR is usually expressed in dB:

$$SFDR = 10 \log_{10} \frac{P_s}{P_x} \quad [\text{dB}] \,, \tag{3.6}$$

where $P_s$ is the signal power and $P_x$ is power of the largest spurious component.

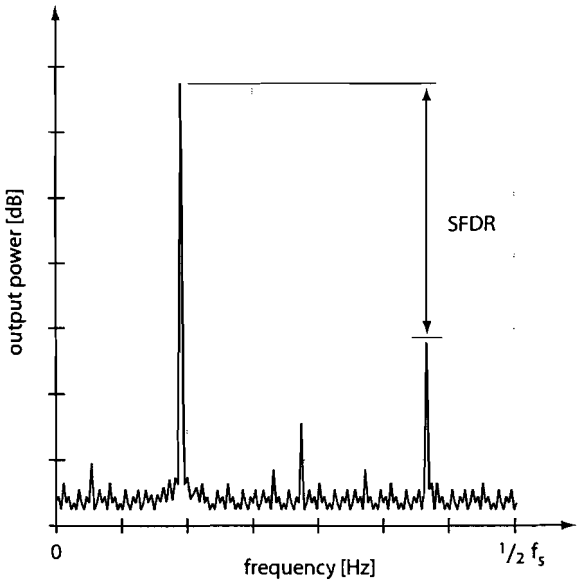



*Figure 3.5: Spurious free dynamic range (SFDR)*

# Chapter 4

# Design issues in current-steering DACs

*In this chapter, the influence of a number of non-idealities on the performance of the converter is discussed. The discussion encloses mismatch errors and the finite output resistance of the current sources.*

## 4.1 Current source mismatch

In the presented design, the unit current sources are realized by NMOS transistors. These devices always suffer from mismatch errors. According to [8], mismatch is defined as the process that causes time-independent random variations in physical quantities of identically designed devices. When applied to transistors, mismatch may be interpreted as statistical device differences between pairs of identical designed and identically used transistors.

Basically, the mismatch error between a pair of transistors is caused by local effects and/or the placing of the transistors on the silicon wafer. The latter is due to various gradients on the wafer. By using proper layout techniques, the effect of these mismatches can be minimized to a negligible level.

Local effects lead to random errors, which are characterized by a normal distribution. They are the result of a number of causes, including edge effects, implantation and surface-state charges, oxide effects and mobility effects. In 1989, Pelgrom [8] proposed a methodology that predicts technology induced mismatches affecting the threshold voltage $V_T$ and the $\beta$ factor ($\mu C_{ox} W/L$). When applied to unit current sources, the following expression is obtained:

$$\frac{\sigma_{unit}}{I_{unit}} = \sqrt{\frac{\frac{1}{2} \cdot \left( A_\beta^2 + 4A_{V_T}^2/(V_{GS} - V_T)^2 \right)}{WL}}, \tag{4.1}$$

where $\frac{\sigma_{unit}}{I_{unit}}$ is the relative spread of the unit current sources, $A_\beta$ and $A_{V_T}$ are technology constants which denote the deviation of respectively $\beta$ and $V_T$ as a function of the

active area, and $(V_{GS} - V_T)$ is the difference between the actual gate-source voltage and threshold voltage, the so-called gate overdrive voltage. According to (4.1), mismatch is not only a function of the size of the transistor, but also of the gate overdrive voltage. Therefore, matching effects gain importance as process dimensions decrease and voltage headroom lowers.

Mismatch errors are a limiting factor in analog systems, especially in high performance D/A converters. The static properties are affected adversely by mismatch errors, since the mismatch between unit current sources highly affects the DNL and INL error. Given that the accuracy requirements concerning the DNL error are related to the architecture of the DAC, the worst-case constraints are used to illustrate the influence of mismatch errors. For that reason, a binary-weighted topology is considered. According to [2], the $\mathrm{DNL}_{max}$ error as a function of the relative spread for a $3\sigma$ confidence level is denoted by:

$$DNL_{max} = 3\sqrt{2^N - 1} \cdot \frac{\sigma_{unit}}{I_{unit}} \quad [\mathrm{LSB}] \,, \tag{4.2}$$

The $\mathrm{INL}_{max}$ error is independent of the chosen topology. The $\mathrm{INL}_{max}$ error as a function of the relative spread for a $3\sigma$ confidence level is given by:

$$INL_{max} = 3\sqrt{2^{N-1}} \cdot \frac{\sigma_{unit}}{I_{unit}} \quad [\mathrm{LSB}] \,, \tag{4.3}$$

where $N$ is the resolution of the converter expressed in bits. Mismatch errors have also an adverse effect on the dynamic specifications. In 1999, Wikner [12] presented the SNDR and SFDR as functions of the relative spread $\frac{\sigma_{unit}}{I_{unit}}$. According to this work, the SNDR is denoted by:

$$SNDR \approx 6{,}02N + 1{,}76 - 10\log_{10}\left(1 + 3\left(\frac{\sigma_{unit}}{I_{unit}}\right)^2 2^{N+1}\right) \quad [\mathrm{dB}] \tag{4.4}$$

The SFDR as function of the relative spread is expressed by:

$$SFDR \approx 3(N+3) - 10\log_{10}\left(\frac{\sigma_{unit}}{I_{unit}}\right)^2 , \quad [\mathrm{dB}] \tag{4.5}$$

The above-mentioned relations illustrate that mismatch errors are a serious problem, as they determine the performance of the D/A converter. Therefore, appropriate solutions which eliminate or decrease the influence of mismatch errors are required.

Expression (4.1) shows that the relative spread is inversely proportional to the square root of the dimensions of the transistor. Therefore, device mismatch seems to be manageable by a proper dimensioning, at the cost of chip area. In other terms, the analog components are designed with an accuracy sufficient to meet the required specifications. D/A converters, which are designed using this approach, are known as intrinsic D/A converters. Basically, these DACs require a large chip area, dominated by the unit current sources. An example is shown in figure 4.1, which contains a die-photo of an intrinsic 12-bit 6/6 segmented current-steering DAC (see [3]). The most striking part of the die is the current source array and its biasing (region D) which takes credit for more than one third of the whole active chip area.

Another approach to decrease the influence of device mismatches is the use of calibration and/or correction techniques. This approach is often used for converters aiming at resolutions above 10-bit. Such high resolutions are accomplished by either improving the effective matching of individual devices by self-adjustment (calibration), or correcting the overall transfer characteristic (correction). However, the use of calibration

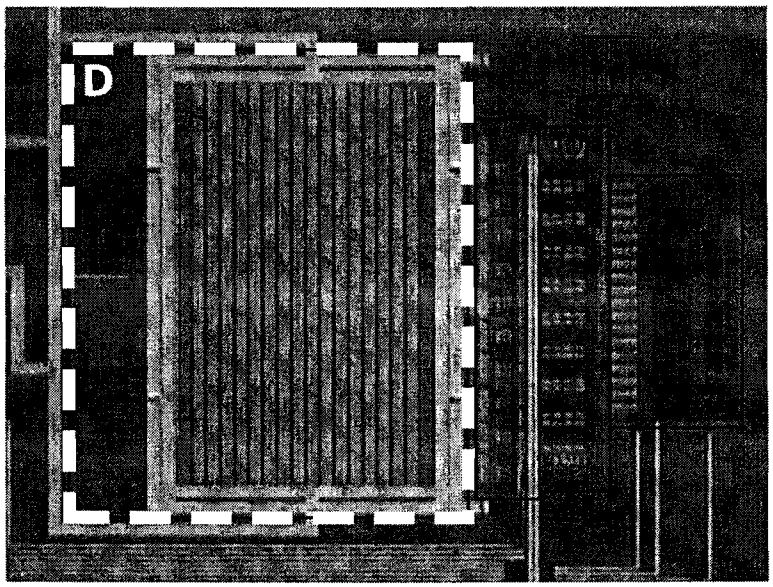



*Figure 4.1: An intrinsic 12-bit 6/6 segmented current-steering DAC*

and/or correction techniques complicates the design, since necessary add-ons have to be implemented. The amount of chip-area occupied by these add-ons has to be minimized. Therefore, still new calibration and correction techniques are proposed, claiming increased performance in combination with less chip-area. For a comprehensive overview of calibration techniques for D/A converters [10] is recommended. A digital pre-correction technique for D/A converters is presented in chapter 5.

## 4.2 Finite output resistance

An ideal current source is a device whose current is independent of the voltage across its terminals. These devices do not suffer from leakage currents, since they have an infinite output resistance. In practical current-steering DAC designs, non-ideal transistors are employed for the implementation of the unit current sources. These devices always have a finite output resistance.

A non-ideal current source can be modeled by a resistance in parallel with an ideal current source. Figure 4.2 shows a simple model of a non-ideal differential thermometer-coded DAC. From this figure, it can be noticed that the current through the load resistance is adversely affected by the finite output resistances of the unit current cells. Furthermore, the combined output current generated by the unit current sources, the common output resistance and thus the DAC's output current are a function of the input code. This is denoted by:

$$I_{diff}(X) = \frac{I_{unit}(X)}{1 + R_{load}/R_{unit}(X)} - \frac{I_{unit}(\overline{X})}{1 + R_{load}/R_{unit}(\overline{X})} , \tag{4.6}$$

where $I_{diff} = (I_{out+} - I_{out-})$ represents the differential output current of the DAC, $I_{unit}(X)$ is the combined output current generated by the unit current sources, $R_{load}$

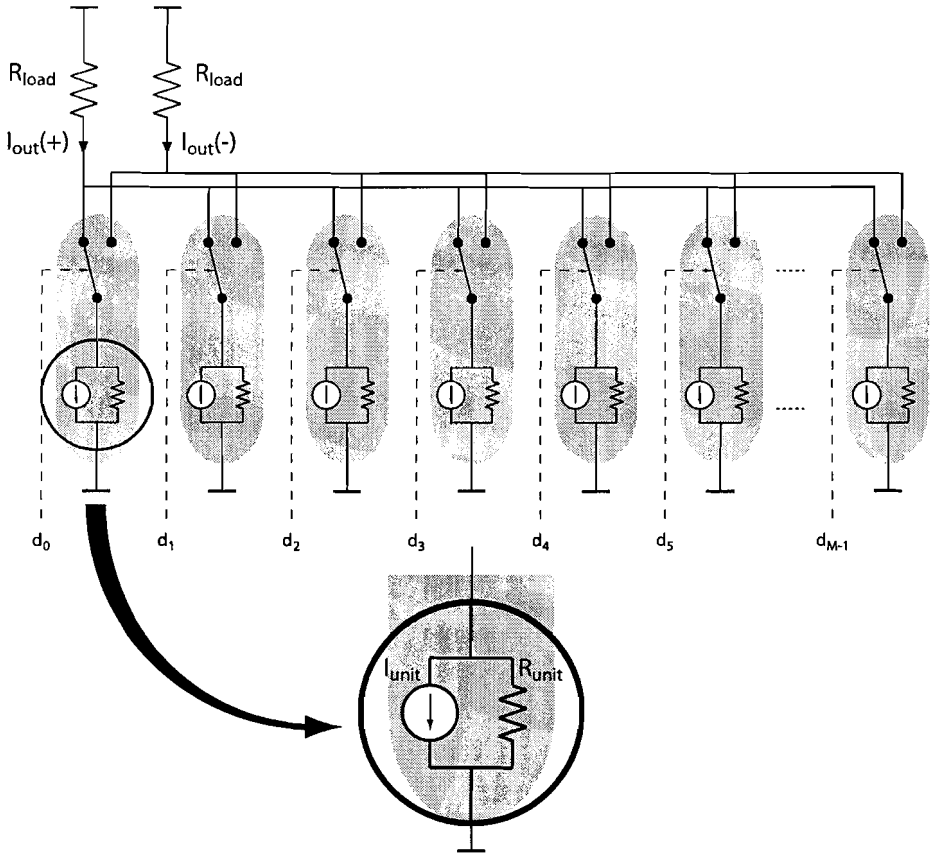


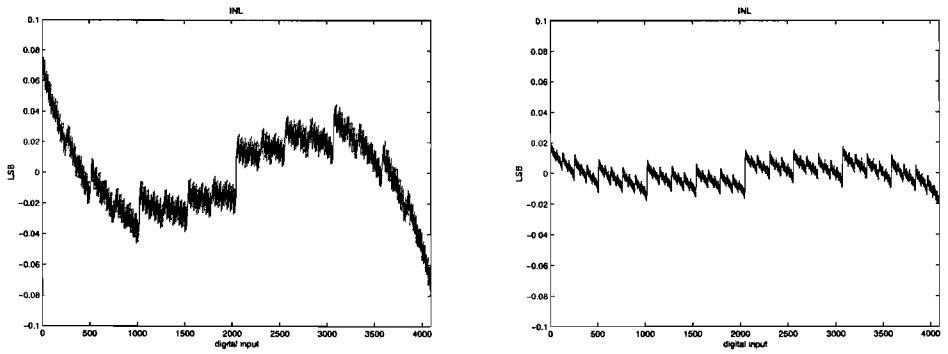

*Figure 4.2: Modeling of the finite output resistance of unit current sources in current-steering DACs*

is the load resistance, $R_{unit}(X)$ is the common output resistance of the unit current sources, $X$ is the digital input code, and $\overline{X}$ is the inverted digital input code.

The signal-dependency of the output current results in non-linear behavior of the converter. However, when the common output resistance of the unit current source is sufficiently large, this non-linearity error could be neglected. As the output resistance is related to the dimensions of the transistor, this erroneous behavior might become a problem if sub-micron current sources are employed. In that case, a correction technique which suppresses the unwanted behavior should be implemented. Note that only static properties are considered in this discussion. The influence of non-ideal current sources on the dynamic performance is disregarded.

As the simplified model of a differential current-steering DAC (see figure 4.2) is not accurate enough to predict the influence of the finite output resistance on the static performance of a differential converter, transistor level simulations are performed. For this purpose, a 12-bit binary-weighted DAC designed for the UMC $0.18\mu m$ CMOS process is being used. Its current sources are implemented as cascoded current sources. Simulations are performed by Cadence, taking no mismatch in account. A ramp over the full input range is applied to the converter, resulting in a full transfer curve. From

this data, the best-fitted INL is derived by Matlab. The simulation is performed twice, using small transistors ($W/L = 200nm/400nm$) (see figure 4.3(a)) and relatively large transistors ($W/L = 3.2\mu m/6.98\mu m$) (see figure 4.3(b)) for the implementation of the current sources. In both cases, the biasing and output currents are maintained. As




(a) $W/L = 200nm/400nm$ (b) $W/L = 3.2\mu m/6.98\mu m$

*Figure 4.3: INL error caused by the finite output resistance of current sources*

expected, the finite output resistance of the current sources adversely affects the static performance of the converter. The non-linear effect is increased when smaller transistors are employed for the implementation of the unit current sources. However, the observed error is small, which makes the implementation of a correction technique or the use of large current sources unnecessary. Therefore, the influence of the finite output resistance on the static performance is neglected. As a technology trend of decreasing dimensions can be observed, a correction technique for the non-linear effect caused by a finite output resistance might be necessary in the near future. Especially, when the cascode transistors have to be removed, as a consequence of a decreased voltage headroom.

# Chapter 5

# Correction methods

*In this chapter a design approach for D/A converters is presented, taking full advantage of new technology trends, and improving performance and reliability.*

## 5.1 Introduction

Basically, a conventional DAC design has to comply with several rules of thumb, like (4.2) in the case of a binary-weighted converter. This results in stringent constraints concerning the relative spread of transistors which are operating as unit current sources. Given that the relative spread of a transistor is related to its size, large chip area is required for high accuracy. As discussed in section 4.1, calibration and/or correction techniques are able to relax the analog design constraints, thereby maintaining or even improving the performance of the converter. However, the improvement in accuracy is limited to only a few bits and the use of calibration techniques is complicating the design.

In this thesis, a design approach for high performance DACs is presented, which does not require accurate current sources in contrast to the conventional approach. Therefore, less chip-area is required for the implementation of the current sources.

To make this approach successful, three components are required:

1. **DAC core with built-in redundancy**, preventing the occurrence of missing output values, caused by the stochastic spread of the unit cells. This can be realized by weighting the current sources by a sub-binary radix, i.d. a radix smaller than two. In the presented approach, a varying sub-binary radix is used, which is determined by an iterative procedure taking the relative unit cell spread into account. As a consequence of utilizing sub-binary radices, overlap is introduced into the transfer curve. This leads to a number of problems including non-monotonicity and full-scale output range reduction. The former problem could be solved by using digital pre-correction. The latter can be solved by adding a number of current sources, counterbalancing the reduction of the full-scale output range.

2. **Digital pre-correction circuit**, re-mapping every digital input code in an appropriate new code representing the desired output value. The re-mapping of the input code is performed online, using a successive approximation algorithm which starts at the MSB. For a successful re-mapping, the output values of the current sources must be available in the digital domain. Therefore, built-in self-measurement is employed.

3. **Built-in self-measurement circuit**, measuring the output value of each current source including its mismatch error. In order to determine the output value of a specific source, a combination of previously measured sources is defined. This combination of sources approximates the output value of the current source under test. As a consequence of this approach, the sources are measured in a relative sense. Whenever the output value of a current source is determined, it is being digitized and saved to digital memory. The presented technique is an iterative procedure, starting at the LSB. The whole measurement procedure is performed during start-up time of the converter.

In the following sections, redundancy, digital pre-correction, and built-in self-measurement are discussed in detail. A system overview is given in figure 5.1.

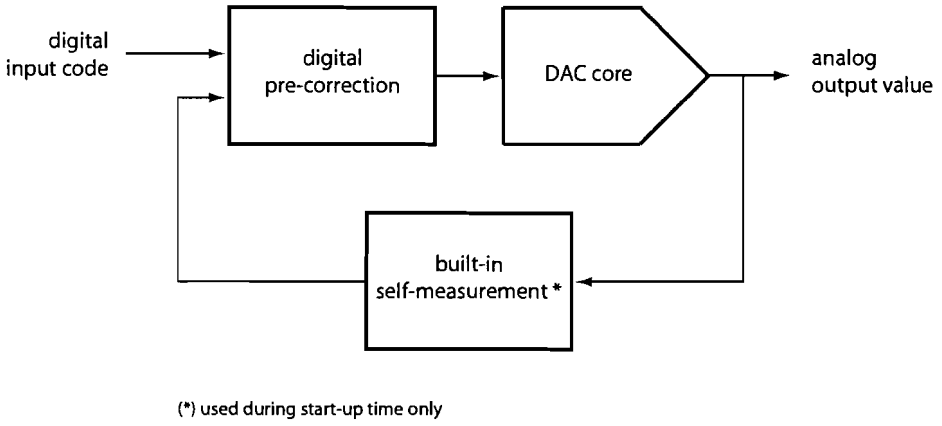



*Figure 5.1: System overview of a digitally pre-corrected DAC with built-in self-measurement*

## 5.2 Redundancy

### 5.2.1 Introduction

The actual output value of a current source mostly deviates from its nominal output value, as a consequence of mismatch errors (see section 4.1). This deviation can be either positive or negative. Both types of mismatch errors are illustrated by figure 5.2, where a binary-weighted DAC with a mismatch error at its MSB is considered. Note that a comparable situation can occur with the other current sources of the DAC.

A positive deviation from the nominal output value may lead in missing output values, when the other sources are considered to be ideal (see figure 5.2 (left)). Missing output values are required output levels that cannot be generated by the DAC, as there is no suitable combination of current sources available. They create a 'gap' in the transfer curve. This situation has to be avoided, as there is no digital pre-correction possible.

A negative deviation from the nominal output value of the current source may result in non-monotone behavior of the converter and leads to a decrement of the full-scale range, when the other sources are considered to be ideal (see figure 5.2 (right)). However, digital pre-correction is able to cancel non-monotonicity and the reduction of the full-scale range is easily counterbalanced by adding more current sources.

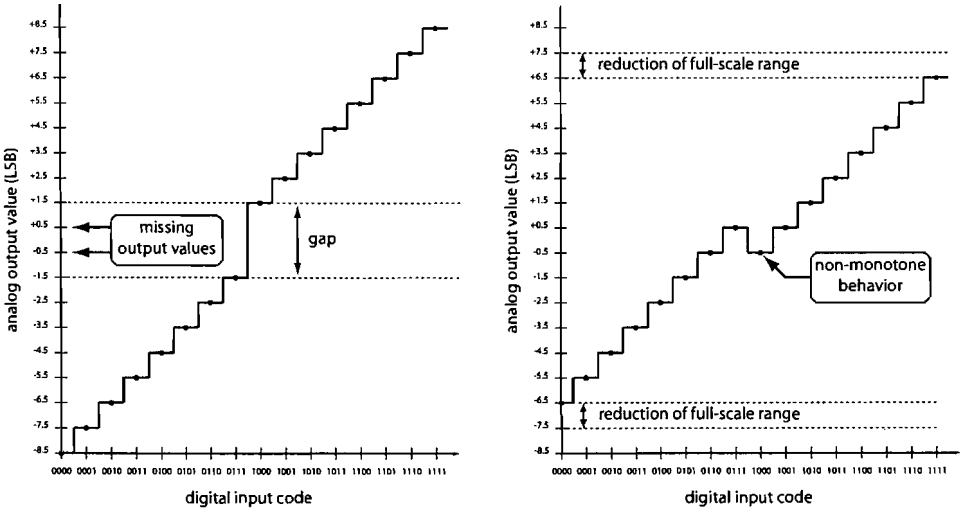



*Figure 5.2: Possible effects of mismatch of the MSB: a positive (left) and a negative (right) deviation of the MSB*

With redundancy, a remedy against missing output values is realized, as 'gaps' in the transfer curve will no longer occur. The more redundancy is added, the more severe deviations due to mismatch errors can be compensated by using pre-correction. However, also note that the more redundancy, the more sources have to be employed as the redundancy reduces the full-scale range of the converter. Nevertheless, redundancy in combination with digital pre-correction and built-in self-measurement results in improved performance and reliability. Figure 5.3 illustrates the effect of redundancy and pre-correction to the DAC.

### 5.2.2 Implementation of redundancy

A normal $N$-bit binary-weighted DAC is composed of $N$ current sources, basically implemented by connecting a number of unit cells in parallel. This is denoted by:

$$I_j = \alpha_j \cdot I_{unit} \ , \tag{5.1}$$

where $I_j$ is the output current of the $j$-th current source, $I_{unit}$ is the output current of a unit cell, and $\alpha_j$ is the number of combined unit cells to achieve the desired out-

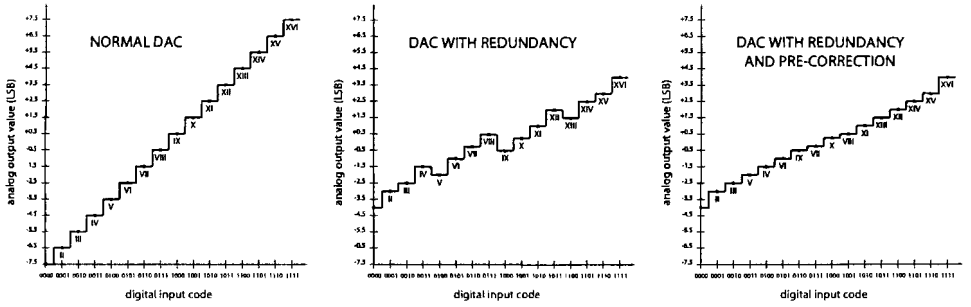



*Figure 5.3: Different transfer characteristics: a normal DAC (left), a DAC with added redundancy without pre-correction (middle), and a DAC with added redundancy and pre-correction (right)*

put current, also referred to as the weighting factor. In the case of a binary-weighted converter, the sources are weighted by a binary radix. For example,

$$\begin{aligned} I_0 &= 1 \cdot I_{unit} \\ I_1 &= 2 \cdot I_{unit} \\ I_2 &= 4 \cdot I_{unit} \\ I_3 &= 8 \cdot I_{unit} \\ I_4 &= \ldots, \text{etc.} \end{aligned}$$

Thereby, the difference between two consecutive levels is exactly 1 LSB. Considering a differential architecture, this is denoted by:

$$\left(I_j - \sum_{i=0}^{j-1} I_i\right) - \left(-I_j + \sum_{i=0}^{j-1} I_i\right) = 1 \text{ LSB} \qquad \text{for } 0 \leq j < N \tag{5.2}$$

Assuming that 1 LSB equals 2 $I_{unit}$, equation (5.2) is simplified by:

$$I_j = \sum_{i=0}^{j-1} I_i + I_{unit} \qquad \text{for } 0 \leq j < N \tag{5.3}$$

As discussed in section 4.1, the output levels of the converter will mostly deviate from their nominal output level. This is a serious problem for positive deviations, which may lead to missing output values. However, this problem can be avoided by adding redundancy to the converter. Therefore, the current sources are weighted by a sub-binary radix, i.d. a radix less than two. As an example, a series of current sources suitable for this approach is given below. For the origin of these numbers is referred to section 5.2.3, where this series is derived.

$$\begin{aligned} I_0 &= 0.74 \cdot I_{unit} \\ I_1 &= 1.31 \cdot I_{unit} \\ I_2 &= 2.42 \cdot I_{unit} \\ I_3 &= 4.57 \cdot I_{unit} \\ I_4 &= \ldots, \text{etc.} \end{aligned}$$

Note that the difference between two consecutive levels is now intentionally smaller than 1 LSB. This is expressed by:

$$I_j > I_{j-1} \quad \wedge \quad I_j < \sum_{i=0}^{j-1} I_i + I_{unit} \qquad \text{for } 1 \leq j < k \ , \tag{5.4}$$

where $k$ is the number of current sources. Relation (5.4) bounds the $j$-th current source by the value of the foregoing source and the sum of all smaller sources. For the special case of $j = 0$ applies $I_0 > 0$ and $I_0 < I_{unit}$.

Due to device mismatch, each unit cell can be described by a gaussian random variable $\mathbf{I_{unit}}$, with mean value $\mu_{\mathbf{I_{unit}}} = I_{unit}$ and variance ${\sigma_{\mathbf{I_{unit}}}}^2 = {\sigma_{unit}}^2$. As a consequence of relation (5.1), the output value of current source $I_j$ is directly affected by the stochastic nature of the unit cells. Therefore, each current source is described by a gaussian random variable $\mathbf{I_j}$ with statistical parameters $\mu_{I_j} = \alpha_j \cdot I_{unit}$, and ${\sigma_{I_j}}^2 = \alpha_j \cdot {\sigma_{unit}}^2$. To guarantee that all required output levels can be produced with sufficient accuracy, the relations from (5.4) have to be fulfilled, taking the stochastic spread of the current sources into account. Therefore, a suitable fixed sub-binary radix system could be selected like the approach proposed in [7]. However, this approach results in a surplus of redundancy at the higher bits of the converter, thereby decreasing the full-scale range unnecessarily. This problem is avoided by implementing exactly as much redundancy as needed. Therefore, a new gaussian random variable $\mathbf{\Delta_j}$ is introduced, which denotes the irregularities between consecutive output levels. This random variable is defined by:

$$\mathbf{\Delta_j} = \sum_{i=0}^{j-1} \mathbf{I_i} - \mathbf{I_j} + I_{unit} \qquad \text{for } 0 \leq j < k \ , \tag{5.5}$$

where its statistical parameters are given by:

$$\mu_{\mathbf{\Delta_j}} = \Big(1 - \alpha_j + \sum_{i=0}^{j-1} \alpha_i\Big) \cdot I_{unit} \tag{5.6}$$

$${\sigma_{\mathbf{\Delta_j}}}^2 = \Big(\sum_{i=0}^{j} \alpha_i\Big) \cdot {\sigma_{unit}}^2 \tag{5.7}$$

Note that $\mathbf{\Delta_j}$ should be positive in order to prevent a 'gap' between two consecutive output levels. This is expressed by $\mathbf{\Delta_j} > 0$ for all $0 \leq j < k$, which has to be fulfilled by a specific level of certainty. Here, a $4\sigma$ confidence level is considered, which equals 99.997% of certainty. This is denoted by:

$$P\{\mathbf{\Delta_j} > 0\} = 1 - \frac{1}{2}\,\mathrm{erfc}\Big(\frac{4}{\sqrt{2}}\Big) = 0.99997 \qquad \text{for } 0 \leq j < k \tag{5.8}$$

Take into account that the level of certainty for the prevention of a 'gap' in the whole transfer characteristic will be less than a $4\sigma$ confidence level. In order to obtain a more practical relation, the left hand side of expression 5.8 is written as:

$$P\{\mathbf{\Delta_j} > 0\} = 1 - \frac{1}{2}\,\mathrm{erfc}\Big(\frac{\mu_{\mathbf{\Delta_j}}/\sigma_{\mathbf{\Delta_j}}}{\sqrt{2}}\Big) \tag{5.9}$$

The combination of (5.8) and (5.9) yields an expression related to the mean value $\mu_{\mathbf{\Delta_j}}$ and the standard $\sigma_{\mathbf{\Delta_j}}$ deviation only:

$$\mu_{\mathbf{\Delta_j}} - 4\sigma_{\mathbf{\Delta_j}} = 0 \qquad \text{for } 0 \leq j < k \ , \tag{5.10}$$

Substitution of (5.6),(5.7) in (5.10) and subsequently solving for $\alpha_j$ leads to an iterative procedure for the derivation of $\alpha_j$ ($j = 0, 1, 2, \ldots, k-1$). Thereby, the exact amount of redundancy which is required to meet (5.8) can be determined. The procedure is given by:

$$\alpha_j = \frac{x_j - \sqrt{x_j^2 - 4y_j}}{2} \quad , \tag{5.11}$$

where $x_j$ and $y_j$ are given by:

$$\begin{aligned} x_j &= 2\left(1 + \sum_{i=0}^{j-1} \alpha_i\right) + 4^2 \left(\frac{\sigma_{unit}}{I_{unit}}\right)^2 \\ y_j &= \left(1 + \sum_{i=0}^{j-1} \alpha_i\right)^2 - 4^2 \left(\frac{\sigma_{unit}}{I_{unit}}\right)^2 \sum_{i=0}^{j-1} \alpha_i \end{aligned}$$

An example of the presented approach is given in the next section.

### 5.2.3 Example of redundancy

As an example, the nominal output current for a sequence of 16 current sources is calculated. Later on, it turns out that this number of sources is required to achieve 12-bit accuracy. A relative spread $\frac{\sigma_{unit}}{I_{unit}}$ of 7.5% is assumed. First, the value of the smallest current source $I_0$ with relative size $\alpha_0$ is derived from (5.1) and (5.11):

$$\left.\begin{aligned} x_0 &= 2.09 \\ y_0 &= 1.00 \end{aligned}\right\} \quad \Rightarrow \quad \alpha_0 = 0.74 \quad \Rightarrow \quad I_0 = 0.74 \cdot I_{unit}$$

The next step is to derive the value of source $I_1$, yielding:

$$\left.\begin{aligned} x_1 &= 3.57 \\ y_1 &= 2.97 \end{aligned}\right\} \quad \Rightarrow \quad \alpha_1 = 1.31 \quad \Rightarrow \quad I_1 = 1.31 \cdot I_{unit}$$

Likewise, the values of all other current sources can be derived. Table 5.1 shows the results for all 16 sources. These results are verified by system level simulations in section 5.5, where a design of a digitally pre-corrected DAC with built-in redundancy and self-measurement is presented.

### 5.2.4 Discussion

Redundancy is proven to be a successful remedy against missing output values, thereby allowing small unit current sources and thus a small chip-area. Furthermore, the reliability of the converter is improved. A number of drawbacks are also known: a DAC core with built-in redundancy requires pre-correction to achieve accurate output values, additional current sources are required to counterbalance the reduction of the output range, and the final saving on silicon is unknown yet. Nevertheless, the approach looks very promising, since a technology trend of decreasing dimensions is observed. Note that the employment of redundancy is not limited to current-steering architectures, as redundancy is not restricted to a certain electrical implementation.

| j | Nominal value of source $I_j$ |
|---|---|
| 0 | $0.74 \cdot I_{unit}$ |
| 1 | $1.31 \cdot I_{unit}$ |
| 2 | $2.42 \cdot I_{unit}$ |
| 3 | $4.57 \cdot I_{unit}$ |
| 4 | $8.78 \cdot I_{unit}$ |
| 5 | $17.05 \cdot I_{unit}$ |
| 6 | $33.39 \cdot I_{unit}$ |
| 7 | $65.78 \cdot I_{unit}$ |
| 8 | $130.16 \cdot I_{unit}$ |
| 9 | $258.34 \cdot I_{unit}$ |
| 10 | $513.87 \cdot I_{unit}$ |
| 11 | $1023.78 \cdot I_{unit}$ |
| 12 | $2041.97 \cdot I_{unit}$ |
| 13 | $4076.02 \cdot I_{unit}$ |
| 14 | $8140.84 \cdot I_{unit}$ |
| 15 | $16265.84 \cdot I_{unit}$ |

*Table 5.1: Values of 16 current sources, assuming $\sigma_{unit} = 7.5\%$ mismatch and $4\sigma$ confidence level*

## 5.3 Digital pre-correction

### 5.3.1 Introduction

As discussed in the previous section, redundancy improves the reliability by preventing the occurrence of missing output values. However, a DAC with built-in redundancy is non-monotonic and totally imprecise. Therefore, it should not be used as a conventional DAC. These problems can be solved by the implementation of digital pre-correction.

The pre-correction algorithm should determine an appropriate combination of current sources, approximating the desired output level as good as possible. For this goal, the $N$-bit digital input code is re-mapped in a $k$-bits code, representing a suitable combination of sources. The $k$-bit code controls the current cells of the DAC, which finally generate the desired output current. Note that the new code contains $k - N$ bits more than the input code, as $k-N$ extra current sources are added to counterbalance the decrement in full-scale range. For a proper working of the pre-correction circuit, the actual values of the current sources should be known and stored in digital memory. Therefore, a built-in self-measurement circuit is employed, which measures the actual value of each current source $j$ $(0 \leq j < k)$, represents it with a digital code $\omega_i$, and stores it in memory (see section 5.4).

An obvious implementation of pre-correction would be an algorithm, which examines all $2^k$ realizable combinations of sources in order to pick the combination which minimizes the difference between the corresponding output current and the desired output current. This approach guarantees the best performance available. However, this algorithm is very time-consuming, becoming a bottleneck during high-speed operation. A look-up table may solve this problem at the cost of chip-area, since the implementation of a look-up table suitable for high performance DACs requires a significant amount of

digital memory, due to the large number of feasible combinations.

A simple, efficient pre-correction algorithm is developed, solving the aforementioned problems. The algorithm shows near-ideal correction behavior, while minimizing the required computational power, since only digital sign-detectors, adders and subtractors are employed. Whenever an input code is applied to the input terminal of the converter, the pre-correction algorithm is executed and a new code, approximating the desired output level is determined. The algorithm can be implemented in a pipe-lined architecture. Another possibility is to store all feasible codes together with their corresponding output levels in a look-up table. However, this will significantly increase the required amount of chip area, which is rather unwanted. The pre-correction algorithm is discussed in the following section.

### 5.3.2 Pre-correction algorithm

In this section, a successive-approximation algorithm is proposed which is able to approximate the desired output level showing near-ideal behavior. Starting with $\omega_j$, the preceding values $\omega_{j-1}$ down to $\omega_0$ are either added to or subtracted from the residual value, such that this value is minimized. This residual value equals the input code at the start of the algorithm. Assuming a differential DAC architecture, the output current of source $j$, represented by $\omega_j$, is contributed to the positive or negative output node, corresponding to this subtraction or addition. The residual signal is used as input for the next iteration in the algorithm, which is shown in figure 5.4.

```
residue ← 'new input code'
for j = k − 1 down to 0
        if residue ≥ 0
                select I_j positive
                residue = residue − ω_j
        else
                select I_j negative
                residue = residue + ω_j
        end if
end for
```

*Figure 5.4: Pre-correction algorithm*

The above algorithm can be completely executed in the digital domain, requiring neither feedback nor intervention with analog components. Furthermore, the algorithm is relatively simple as it requires sign-detection, addition and subtraction only, minimizing the required computational power. An example of the presented algorithm is given in the next section. Simulation results are given in section 5.5.

### 5.3.3 Example of digital pre-correction

This section gives an example of the pre-correction algorithm. Consider a converter with eight sources, having measured values $\omega_j$ $(0 \leq j < 8)$, equal to the values of the current sources as given in table 5.1. Take into account that these values are only

employed to illustrate the working of the algorithm. In a practical situation, digitized measurement results are used. For example, consider the pre-correction of input code 63. Applying the algorithm from figure 5.4, current source 7 will contribute its output current $I_7$ to the positive output of the DAC, as $residue = 63 \geq 0$. The new value of *residue* becomes $residue = 63 - \omega_7 = -2.78$. In the second iteration, *residue* is negative, hence $I_6$ is contributed to the negative output, and the new *residue* becomes $residue = -2.78 + \omega_6 = 30.61$, and so on. Table 5.2 summarizes the iterations of the algorithm, also showing the final DAC output level (63.16), and the quantization error (0.16). Figure 5.5 illustrates the working of the algorithm.

| Iteration | *Residue* | *Residue* $\geq 0$ ? | Weight Contribution | | New code |
|---|---|---|---|---|---|
| 1 | +63.00 | y | $\omega_7$ | +65.78 | 1 |
| 2 | −2.78 | n | $\omega_6$ | −33.39 | 10 |
| 3 | +30.61 | y | $\omega_5$ | +17.05 | 101 |
| 4 | +13.56 | y | $\omega_4$ | +8.78 | 1011 |
| 5 | +4.78 | y | $\omega_3$ | +4.57 | 10111 |
| 6 | +0.21 | y | $\omega_2$ | +2.42 | 101111 |
| 7 | −2.21 | n | $\omega_1$ | −1.31 | 1011110 |
| 8 | −0.90 | n | $\omega_0$ | −0.74 | 10111100 |
| | −0.16 | | $\Sigma\omega$ | +63.16 | 10111100 |

*Table 5.2: Example of digital pre-correction for input code 63*

### 5.3.4 Discussion

A simple, efficient pre-correction algorithm is presented, showing near-ideal correction behavior while minimizing the required computational power, as for the implementation of the algorithm only digital sign-detectors, adders and subtractors are required. Note that it is unknown yet, which amount of chip-area is actually required to implement the algorithm. This could be a serious drawback, as in comparable implementations this seems to dominate the overall chip-area [7].

## 5.4 Built-in self-measurement

### 5.4.1 Introduction

The built-in self-measurement circuit determines the actual values of the current sources. It is an essential component in the presented DAC design, since the actual output currents have to be known to take advantage of the previously discussed correction techniques: redundancy and digital pre-correction.

The measurement circuit has to meet a number of hard constraints, since it has to be reliable, small, and realizable on-chip. Besides that, adverse influences of the measurement method on the overall accuracy and reliability of the DAC have to be minimized. Also note that, as the pre-correction algorithm is implemented in the digital domain, the measurement method should provide the measurement results in a digitized representa-

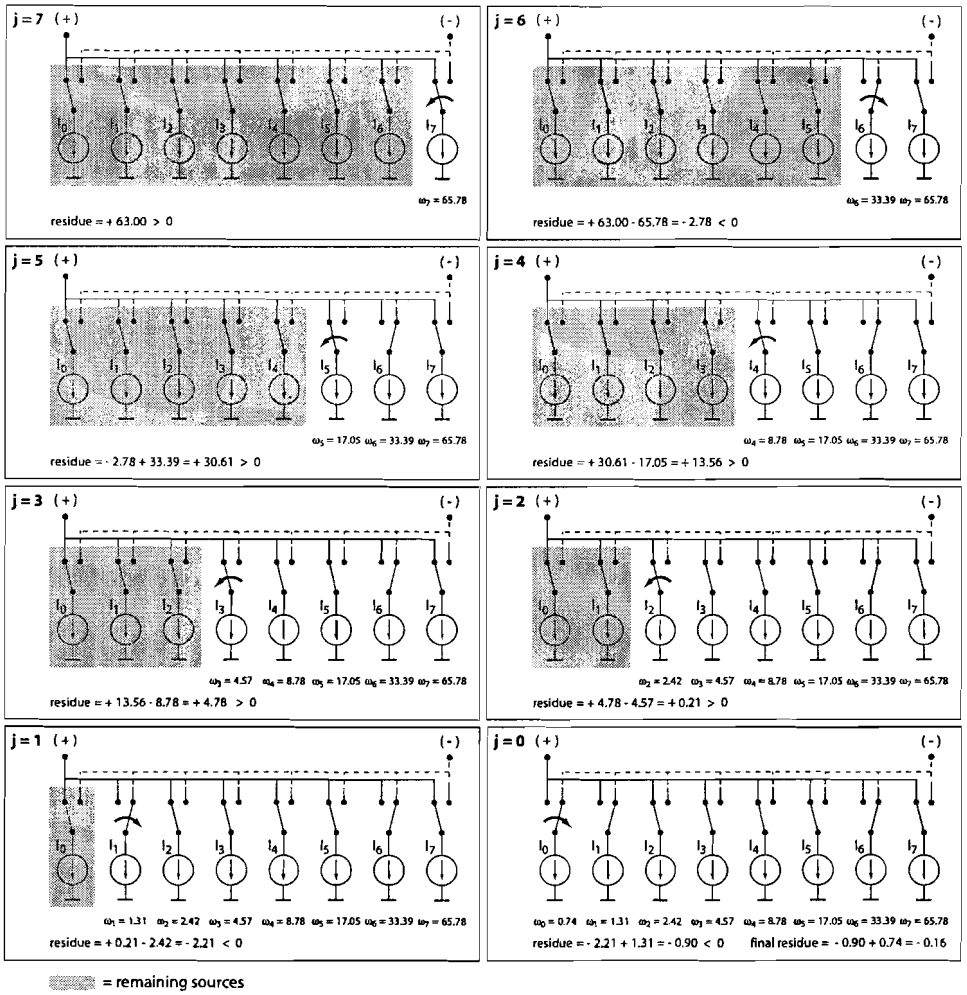



*Figure 5.5: Example of digital pre-correction for input code 63*

tion. A solution suitable for differential architectures, meeting all these constraints, is proposed in the next section.

### 5.4.2 Implementation of built-in self-measurement

A straight-forward approach to determine the actual values of the sources is to measure them in an absolute sense. However, this approach requires accurate, bulky components. The proposed technique measures the current sources relatively to each other, requiring a simple voltage comparator and a digital feedback only. These components are small and realizable on-chip, as a consequence of the mixed-signal system-on-chip (MSSoC) technology trend. Furthermore, the measurement results are automatically digitized by the comparator. Figure 5.6 shows an overview of this approach.

The aim of the measurement technique is to find for each current source $j$ a combination

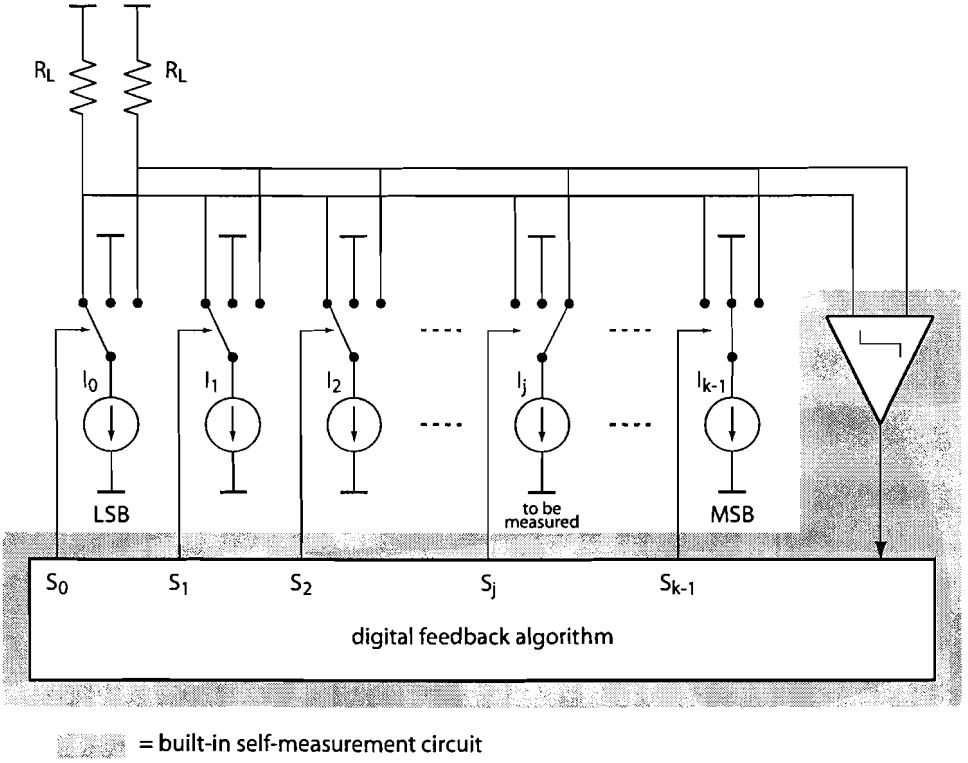



*Figure 5.6: Built-in self-measurement setup using a digital feedback algorithm, and a comparator*

of smaller current sources $j-1$ down to 0 of which the combined output value $I_{sum}$ approximates the output value of source $j$, being $I_j$ as good as possible. Therefore, the difference between $I_j$ and $I_{sum}$ is minimized, where $I_j$ remains larger than $I_{sum}$. This condition is obvious for conventional DACs, since the difference between current source $I_j$ and the sum of all smaller sources equals exactly 1 LSB. In case of a DAC with redundancy, the fulfillment of this condition is not guaranteed. On the other hand, this enables the measurement of the actual output values relatively to each other. An appropriate combination composing $I_{sum}$ is found by using an analog voltage comparator and a feedback algorithm implemented in the digital domain. The voltage comparator is able to determine the sign of $I_j - I_{sum}$ since they are converted in voltages by the load resistances. Utilizing a voltage comparator is advantageous as it could stay connected to the output nodes of the converter during normal operation, without affecting the performance adversely. Note that in order to determine the sign of $I_j - I_{sum}$, $I_j$ and $I_{sum}$ have to be contributed to the opposite output nodes of the DAC. During the measurement phase of source $j$, the value of $I_{sum}$ is gradually increased by combining more sources; starting at the largest source $j-1$, sources $j-2$ down to 0 are added until $I_j - I_{sum}$ becomes negative. The source which causes the sign change is disabled, whereupon the measurement process is continued until all sources have been examined. The combined value $I_{sum}$ is expressed as:

$$I_{sum} = \sum_{i=0}^{j-1} S_i I_i \ , \tag{5.12}$$

where $I_i$ is the actual value of $i$-th current source, and $S_i$ is 0 when the source is not used and 1 when the source is used.

The measurement process starts by determining a combination of sources which approximate current source 1. Then, the same algorithm is repeated for current source 2, 3, ... up to $k-1$. It is not possible to approximate the value of the smallest source 0, as there is no combination of smaller sources available. As a consequence of the above, the algorithm has to be executed $k-1$ times in order to determine the values of all current sources. The measurement of a single source $j$ requires $j-1$ iterations ($0 \leq j < k$), according to the feedback algorithm illustrated in figure 5.7. Note that this algorithm works in a similar manner as a successive approximation algorithm.

```
for j = 1 up to k − 1
        Select source j positive (I_j)
        Turn off all sources (I_sum = 0)
        for i = j − 1 down to 0
                Select source i negative (I_sum = I_sum + I_i)
                if I_sum > I_j
                        S_i = 0
                        Turn off source i (I_sum = I_sum − I_i)
                else
                        S_i = 1
                end if
        end for
end for
```

*Figure 5.7: Feedback algorithm required for the implementation of built-in self-measurement*

In the digital domain, the actual value of source $j$ is represented by a digital code $\omega_j$, and can be derived from the values $S_i$, as determined by the measurement algorithm and the values of $\omega_i$, as determined in the preceding measurement phases:

$$\omega_j = \sum_{i=0}^{j-1} S_i \omega_i \qquad \text{for } 1 \leq j < k \tag{5.13}$$

For the special case of $j = 0$, $\omega_0$ cannot be approximated, as there are no smaller sources than source 0 available. Therefore, the value of $\omega_0$ is set to 1. All digitized measurement values $\omega_j$ ($0 \leq j < k-1$) are stored in digital memory, in behalf of the digital pre-correction algorithm. An example of the self-measurement method is given in the next section. Simulation results are presented in section 5.5.

### 5.4.3 Example of built-in self-measurement

For illustration, the presented measurement algorithm will be performed on a converter with 16 current sources. Suppose that the actual values of the sources of this converter correspond to the values given in table 5.1.

First, to initialize the recursive measurement algorithm, $\omega_0$, the digital value representing source 0, is set to 1. Now, source 1 with value $I_1 = 1.31 \cdot I_{unit}$ can be measured using

the algorithm from fig 5.7 with $j = 1$. The only source that can be used to compose $I_{sum}$ is source 0. Turning this source on yields $I_{sum} = I_0 = 0.74 \cdot I_{unit}$. As $I_{sum} \leq I_0$ the algorithm results in $S_0 = 1$, and thus the digital representation of source 1 becomes $\omega_1 = 1$ (according to (5.13)).

In the next step, the algorithm is repeated for source 2 with value $I_2 = 2.42 \cdot I_{unit}$. First, $I_{sum}$ is set to $I_1$. As $I_{sum}$ is smaller than $I_2$, $S_1$ becomes 1. Then, $I_0$ is added to $I_{sum}$. As $I_{sum}$ is still smaller than $I_2$, $S_0$ becomes also 1, and $\omega_2 = \omega_1 + \omega_0 = 2$ is yielded.

Likewise, the algorithm is repeated for the other sources composing the converter, resulting in the measured values $\omega_j$ as given in table 5.3. In addition to the above, figure 5.8 illustrates the working of the algorithm for the measurement of source 5 ($j = 5$).

| j | Nominal value of source $I_j$ | Approximation code $S_i$ , $i \in [j-1, 0]$ | Measurement result $\omega_j$ |
|---|---|---|---|
| 0 | $0.74 \cdot I_{unit}$ | - | 1 |
| 1 | $1.31 \cdot I_{unit}$ | 1 | 1 |
| 2 | $2.42 \cdot I_{unit}$ | 11 | 2 |
| 3 | $4.57 \cdot I_{unit}$ | 111 | 4 |
| 4 | $8.78 \cdot I_{unit}$ | 1101 | 7 |
| 5 | $17.05 \cdot I_{unit}$ | 11101 | 14 |
| 6 | $33.39 \cdot I_{unit}$ | 111101 | 27 |
| 7 | $65.78 \cdot I_{unit}$ | 1111010 | 53 |
| 8 | $130.16 \cdot I_{unit}$ | 11111000 | 105 |
| 9 | $258.34 \cdot I_{unit}$ | 111110101 | 209 |
| 10 | $513.87 \cdot I_{unit}$ | 1111110000 | 415 |
| 11 | $1023.78 \cdot I_{unit}$ | 11111101000 | 827 |
| 12 | $2041.97 \cdot I_{unit}$ | 111111011101 | 1650 |
| 13 | $4076.02 \cdot I_{unit}$ | 1111111001110 | 3293 |
| 14 | $8140.84 \cdot I_{unit}$ | 11111110111000 | 6577 |
| 15 | $16265.84 \cdot I_{unit}$ | 111111110011010 | 13141 |

*Table 5.3: Nominal values and measurement results of 16 current sources*

As the measuring circuit does not measure current sources in an absolute sense, but only relatively to each other, the numbers from table 5.1 and table 5.3 look quite different at first sight. However, when the measured values are normalized to $I_{15}$ (multiplied with $\alpha_{15}/\omega_{15}$), the resemblance becomes apparent (table 5.4). Note that this normalization is performed only to demonstrate the correctness of the measurement algorithm.

From the normalized measurement results, it can be seen that the sources are measured correctly, except for the smallest ones. However, the maximum measurement error is around 0.5, while the LSB of this converter is roughly $\alpha_{15}/2^{N-1} \approx 8$, assuming a $N = 12$ bit accuracy. Therefore, the INL/DNL errors introduced by the measurement errors are relatively small. Extensive simulations (section 5.5) also show that the INL and DNL (including measurement errors) remain within the targeted specification of 0.5 LSB.

| j | Actual value of source $I_j$ | Normalized measurement result $\hat{\omega}_j$ |
|---|---|---|
| 0 | $0.74 \cdot I_{unit}$ | 1.24 |
| 1 | $1.31 \cdot I_{unit}$ | 1.24 |
| 2 | $2.42 \cdot I_{unit}$ | 2.48 |
| 3 | $4.57 \cdot I_{unit}$ | 4.95 |
| 4 | $8.78 \cdot I_{unit}$ | 8.66 |
| 5 | $17.05 \cdot I_{unit}$ | 17.33 |
| 6 | $33.39 \cdot I_{unit}$ | 33.42 |
| 7 | $65.78 \cdot I_{unit}$ | 65.60 |
| 8 | $130.16 \cdot I_{unit}$ | 129.97 |
| 9 | $258.34 \cdot I_{unit}$ | 258.70 |
| 10 | $513.87 \cdot I_{unit}$ | 513.68 |
| 11 | $1023.78 \cdot I_{unit}$ | 1023.65 |
| 12 | $2041.97 \cdot I_{unit}$ | 2052.36 |
| 13 | $4076.02 \cdot I_{unit}$ | 4076.05 |
| 14 | $8140.84 \cdot I_{unit}$ | 8140.97 |
| 15 | $16265.84 \cdot I_{unit}$ | 16265.84 |

*Table 5.4: Actual values and normalized measurement results of 16 current sources*

### 5.4.4 Discussion

The proposed built-in self-measurement circuit is simple, and realizable on-chip. However, built-in self-measurement has a number of disadvantages also. A major drawback is that the presented approach dominates the static performance of the converter, as shown by system level simulations (see section 5.5). This is caused by the inaccuracy by which the current sources are measured. As this problem concerns the used algorithm, it is considered as a problem on an algorithmic level. In addition to the aforementioned problem, two problems on an implementation level can be mentioned. The first drawback is formed by the alterations to the DAC core, since the current sources must be able to be switched off. This operation is not feasible in an ordinary differential architecture, as the sources contribute their values to either the positive or negative output node. Secondly, a DC offset in the comparator influences the measurement adversely and limits the accuracy of the measurement algorithm directly. For example, consider the measurement process for source 2. The sum of $I_1$ and $I_0$ is smaller than $I_2$, whereby $\omega_2$ becomes $\omega_1 + \omega_0 = 1 + 1 = 2$ (see section 5.4.3). Assume that the sum of $I_1$ and $I_0$ becomes larger than $I_2$, due to an offset error of the comparator. In that case $\omega_2$ would equal $\omega_1 = 1$, adversely affecting all subsequent measurement results. Further investigation has yielded an extension of the presented technique, canceling the last two drawbacks (see [4] and [5]). The discussion of this approach is beyond the scope of this thesis.

## 5.5 Design example

In this section, the design of a digitally pre-corrected DAC with built-in redundancy and self-measurement is presented, in order to verify the design approach described

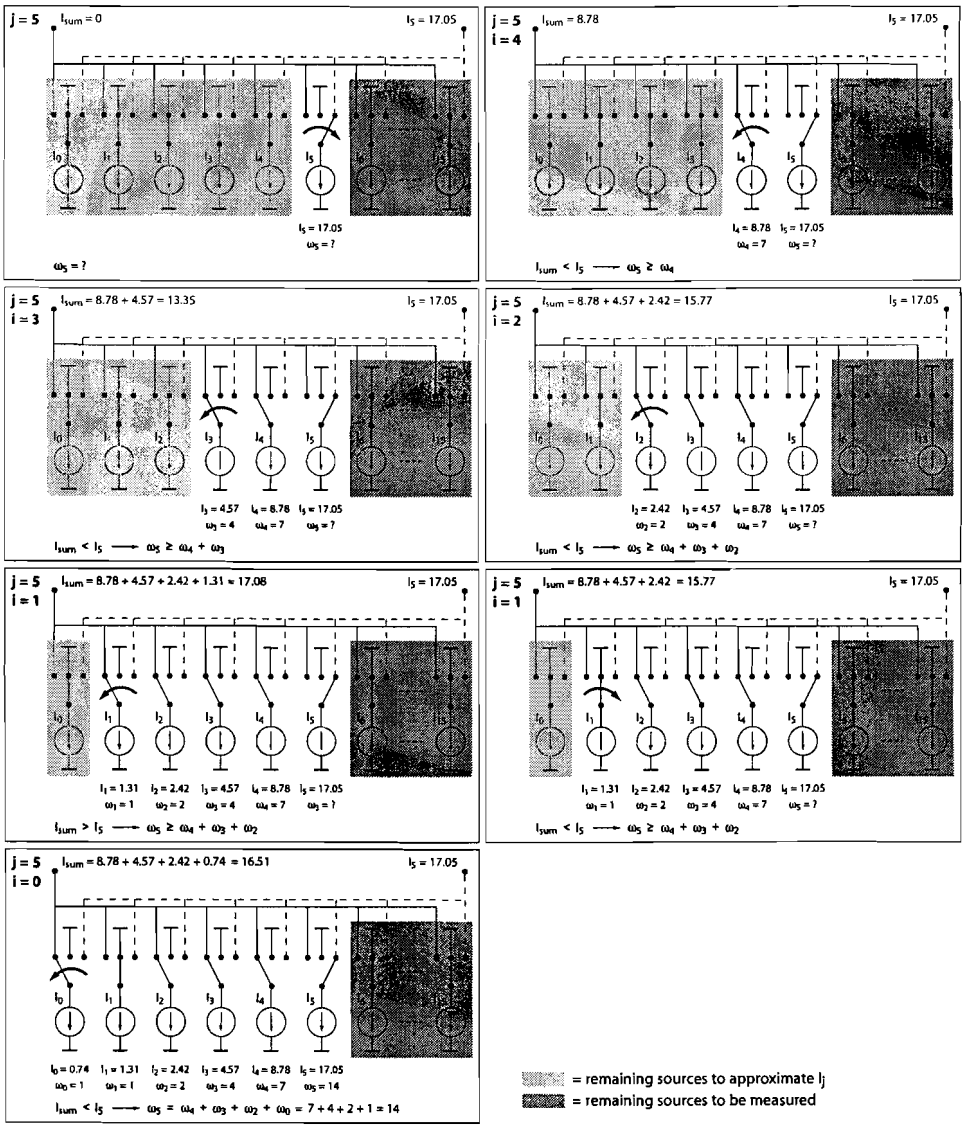



*Figure 5.8: Example of built-in self-measurement technique for $j = 0$ up to 2*

throughout chapter 5. The static properties of the converter are derived by system level simulations, thereupon compared to the performance of a normal binary-weighted DAC.

For the design of the digitally pre-corrected DAC, 16 current sources are required in order to achieve 12-bit accuracy after pre-correction and self-measurement. Here, a relative unit cell spread of 7.5% and a $4\sigma$ confidence level are assumed. For the implementation of a normal 12-bit binary-weighted DAC, 12 sources with a relative unit cell spread of 0.37% are required in order to achieve an INL error less than $\pm 0.5$ LSB (see expression (4.3)). The nominal output values of the required current sources of both designs are summarized in table 5.5. Note that the unit cell employed for the binary-weighted DAC is not the same as used for the pre-corrected DAC, since the full-scale output current

should be similar in both cases. The static performance of both designs is derived

| j | Nominal value of source $I_j$ |
|---|---|
| 0 | $1 \cdot I^*_{unit}$ |
| 1 | $2 \cdot I^*_{unit}$ |
| 2 | $4 \cdot I^*_{unit}$ |
| 3 | $8 \cdot I^*_{unit}$ |
| 4 | $16 \cdot I^*_{unit}$ |
| 5 | $32 \cdot I^*_{unit}$ |
| 6 | $64 \cdot I^*_{unit}$ |
| 7 | $128 \cdot I^*_{unit}$ |
| 8 | $256 \cdot I^*_{unit}$ |
| 9 | $512 \cdot I^*_{unit}$ |
| 10 | $1024 \cdot I^*_{unit}$ |
| 11 | $2048 \cdot I^*_{unit}$ |
| - | - |
| - | - |
| - | - |
| - | - |

| j | Nominal value of source $I_j$ |
|---|---|
| 0 | $0.74 \cdot I_{unit}$ |
| 1 | $1.31 \cdot I_{unit}$ |
| 2 | $2.42 \cdot I_{unit}$ |
| 3 | $4.57 \cdot I_{unit}$ |
| 4 | $8.78 \cdot I_{unit}$ |
| 5 | $17.05 \cdot I_{unit}$ |
| 6 | $33.39 \cdot I_{unit}$ |
| 7 | $65.78 \cdot I_{unit}$ |
| 8 | $130.16 \cdot I_{unit}$ |
| 9 | $258.34 \cdot I_{unit}$ |
| 10 | $513.87 \cdot I_{unit}$ |
| 11 | $1023.78 \cdot I_{unit}$ |
| 12 | $2041.97 \cdot I_{unit}$ |
| 13 | $4076.02 \cdot I_{unit}$ |
| 14 | $8140.84 \cdot I_{unit}$ |
| 15 | $16265.84 \cdot I_{unit}$ |

*Table 5.5: Nominal values of the current sources required for the design of a conventional binary-weighted DAC (left) and a digital pre-corrected DAC with built-in redundancy and self-measurement (right)*

by system level simulation, using software written in the C-programming language. From the simulation results, a comparison can be made between the conventional design approach and the approach described throughout this chapter. In addition, different configurations of the digitally pre-corrected DAC are examined, in order to determine the performance of the individual components. All configurations used are summarized below:

1. **Normal binary-weighted current-steering DAC**, containing 12 current sources with a relative unit cell spread of 0.37%, aiming at 12-bit accuracy (see figure 5.9).
2. **Digitally pre-corrected DAC, with built-in redundancy and self-measurement**,
   (a) containing 16 current sources with a relative unit cell spread of 7.5%, aiming at 12-bit accuracy after digital pre-correction and built-in self measurement. The digital pre-correction and built-in self-measurement are implemented using the simplified algorithms as proposed in section 5.3 and 5.4 respectively (see figure 5.10).
   (b) as 2(a), but using ideal pre-correction, by inspecting all realizable output levels in order to select the best solution available (see figure 5.11).
   (c) as 2(a), but using ideal self-measurement, by copying the actual values of the current sources into digital memory directly (see figure 5.12).

Monte-Carlo simulations are performed on 10000 samples of each configuration, deriving maximum INL and DNL for each sample. Figures 5.9, 5.10, 5.11 and 5.12 show the results. From the comparison between figure 5.9 and 5.10, it can be concluded that the digitally pre-corrected DAC yields improved static performance and reliability. The

simplified algorithm for the digital pre-correction achieves (near-)ideal results as shown by figure 5.10 and 5.11. Therefore, the proposed pre-correction algorithm is successful. The source of concern of this approach is the built-in self-measurement, by dominating the static performance of the converter (see figure 5.12). An improved measurement circuit is required in order to exploit the advantages of this approach further.

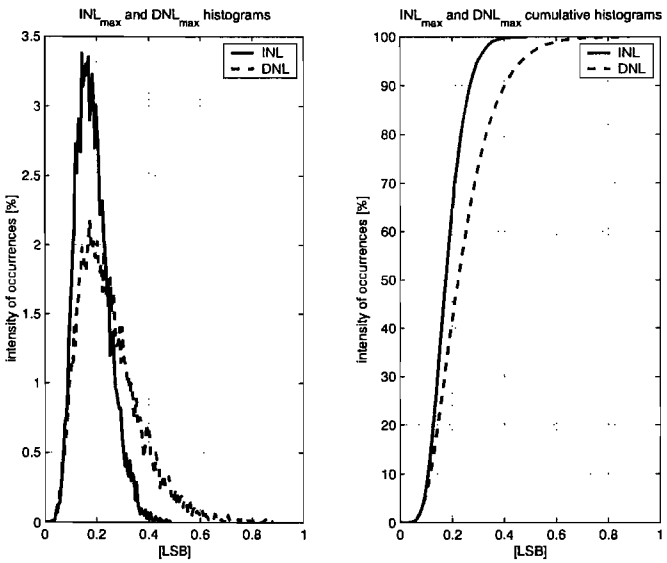



*Figure 5.9: Maximal INL and DNL for 10000 binary-weighted DACs (configuration 1: no pre-correction)*

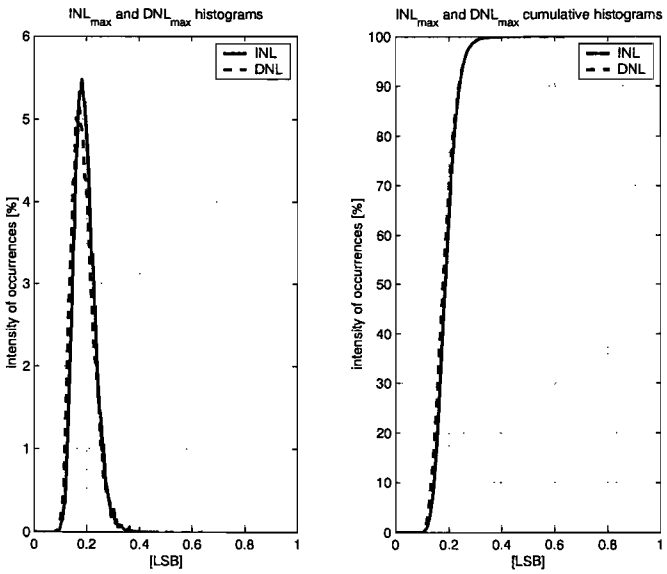



*Figure 5.10: Maximal INL and DNL for 10000 digitally pre-corrected DACs (configuration 2a: proposed digital pre-correction and proposed self-measurement)*

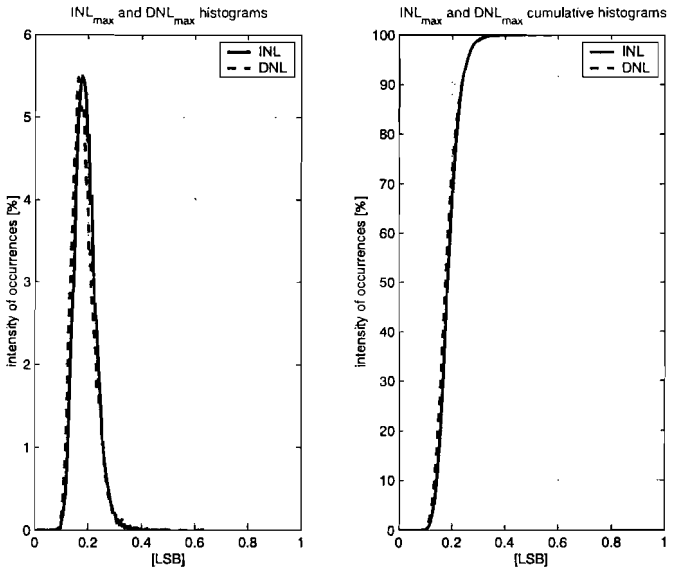



*Figure 5.11: Maximal INL and DNL for 10000 digitally pre-corrected DACs (configuration 2b: ideal digital pre-correction and proposed self-measurement)*

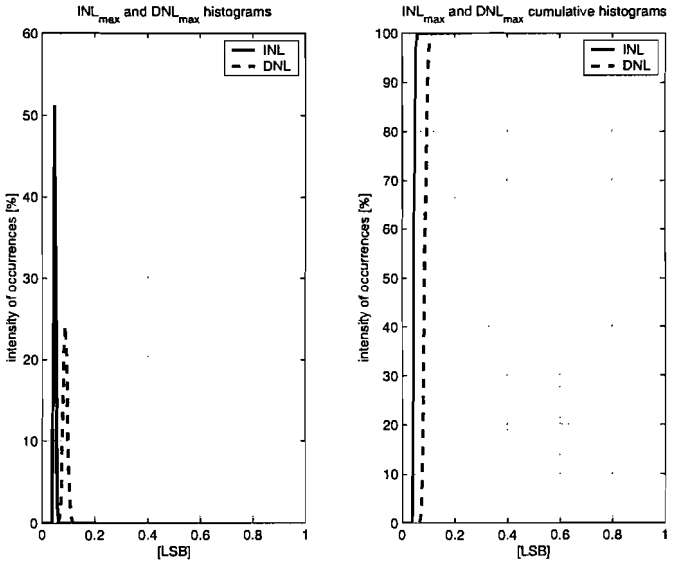



*Figure 5.12: Maximal INL and DNL for 10000 digitally pre-corrected DACs (configuration 2c: proposed digital pre-correction and ideal self-measurement)*

# Chapter 6

# Transistor level design of a digitally pre-corrected DAC

*In this chapter, the circuit level design of a digitally pre-corrected DAC with built-in redundancy is presented. Its performance is verified by simulations on circuit level.*

## 6.1 Design goal and constraints

In chapter 5, an approach for the design of high-performance DACs is presented, yielding improved performance and reliability. This was verified by simulations on system level (see section 5.5). DACs based on this approach contain digital pre-correction, an analog core with built-in redundancy, and self-measurement.

In this chapter, a circuit level design of a digitally pre-corrected DAC with built-in redundancy is presented. This design enables the verification of the aforementioned statements by simulations on circuit level. DC simulations are suitable, since the presented technique corrects static errors only. The simulations are performed by Cadence, taking current source mismatches into account. The design is based on an existing DAC design of Radulov, resulting in less design effort as most components can be reused.

Note that self-measurement is not implemented in the design, which makes alterations to the DAC core unnecessary. The lack of built-in self-measurement does not affect the simulation results, since the measurements are performed during the start-up phase only. The measured values of the current sources, which are required for a proper working of the digital pre-correction, are obtained by Matlab simulations using the simplified self-measurement algorithm (see section 5.4).

An overview of design specifications is given in table 6.1.

| Specification | Value |
|---|---|
| Resolution | 12 bits |
| $INL_{max}$ | ± 0.5 LSB |
| $DNL_{max}$ | ± 0.5 LSB |
| Full-scale current | ± 20 mA |
| Load resistance | 25 Ω |
| Signal swing | ± 0.5 V |
| Technology | CMOS 0.18 $\mu$m |
| Supply voltage | 1.8 V |

*Table 6.1: Overview of design specifications*

## 6.2 DAC architecture

The DAC architecture is fully differential, containing digital pre-correction and a DAC core with built-in redundancy. Figure 6.1 shows the block diagram of this architecture.

After a 12-bit input code is applied to the input terminal of the digital pre-correction, it is re-mapped in a 16-bit code. This code represents a combination of sources, approximating the desired output current. It is directed directly to the analog DAC core, which is composed of latches, drivers, and switched current cells. The latches and drivers are required to obtain well-shaped, synchronized waveforms, driving the switched current cells. Each current cell contributes its output value to either the positive or negative output node of the DAC, in order to generate an output current corresponding to the digital input code. The full-scale output current of the DAC equals approximately 20 mA, with a signal swing around 0.5 V as the converter is terminated by two 25Ω load resistances. In the following sections, the above-mentioned building blocks are discussed briefly.

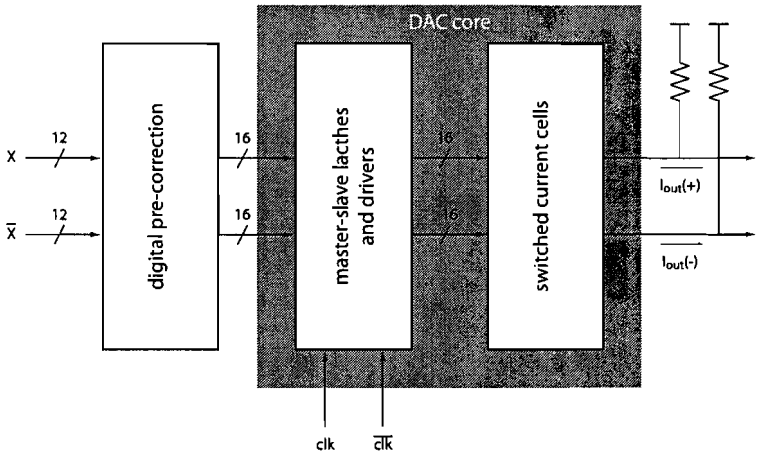



*Figure 6.1: Block diagram of the DAC architecture*

## 6.3 Design of the digital pre-correction

In section 5.3 an efficient and simple algorithm is presented, re-mapping the input codes in order to generate the corresponding output values. According to this algorithm, the measured values $\omega_j$ $(0 \leq j < k)$ are either added to or subtracted from the input code $X$, such that the residual value *residue* is minimized. In order to obtain correct results, the input codes and measurement results should be normalized to the same range. The algorithm shown in figure 6.2, re-maps 12-bit input codes in 16-bit codes and performs the above-mentioned normalization process. The structure and behavior of this algorithm are described in Verilog-A (see appendix A). Using this model in combination with an ideal ADC, a variety of circuit level simulations can be set up easily, since waveforms can be applied without interaction of another simulation tool (Matlab).

$residue \leftarrow (X - \overline{X})/(2^{12} - 1)$
for $j = 15$ down to 0
    if $residue \geq 0$
        select $I_j$ positive
        $residue = residue - (\omega_j/\omega_{FS})$
    else
        select $I_j$ negative
        $residue = residue + (\omega_j/\omega_{FS})$
    end if
end for

$X$ = non-inverted input code
$\overline{X}$ = inverted input code
$\omega_{FS} = \sum_{i=0}^{15} \omega_i$

*Figure 6.2: Extended digital pre-correction algorithm suitable for a 12-bit DAC with built-in redundancy*

## 6.4 Design of the analog DAC core

### 6.4.1 Master-slave latches and drivers

In the presented design, a master-slave latch configuration is used as an interface between digital pre-correction and switched current cells, in order to shape and synchronize the data signals before driving the switches. Figure 6.3 shows this configuration, consisting of two latches and corresponding drivers, which are implemented in current mode logic (CML). The latches are clocked with a differential clock signal.

After the digital input code has been pre-corrected, it is applied to the input terminal of the master latch. The master latch synchronizes and shapes the incoming data, thereby removing most disturbances. At the next clock phase, the signal is directed to the slave latch. The slave latch attenuates any remaining disturbances of the data signal, resulting in a well-shaped, precisely synchronized waveform. Note that the master-slave

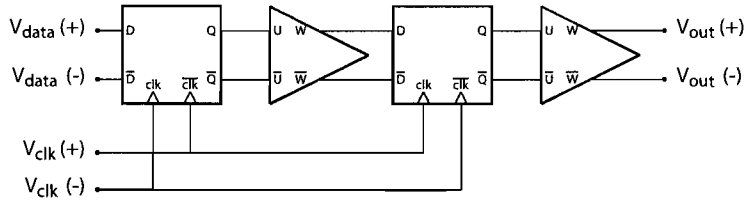



*Figure 6.3: Master-slave latch configuration*

configuration performs a coarse-fine conditioning of the signal. Both latches are based on the same topology, which is shown in figure 6.4(a).

Each latch is followed by a driver that establishes a correct voltage swing and filters disturbances caused by latch transitions. Both drivers are implemented as normal differential pairs, as shown in figure 6.4(b).

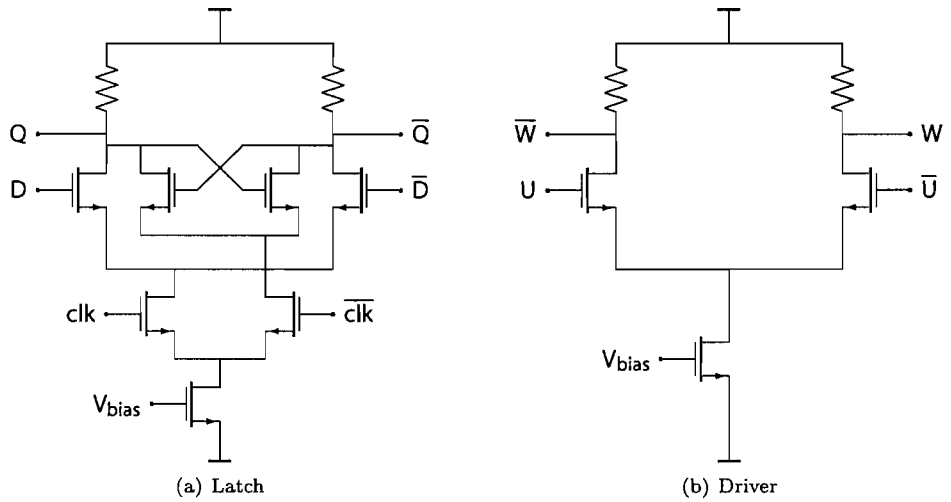



*Figure 6.4: Circuit diagram of a latch and a driver*

### 6.4.2 Switched current cell

The switched current cell consists of a cascoded current source and a cascoded differential switch pair, as shown in figure 6.5. Each current cell contributes its output value to either the positive or negative output node of the converter, in order to generate an output current corresponding to the digital input code. The switched current cell is an essential component in current-steering DAC designs, which highly affects the static performance. In the following sections the cascoded current source and differential switch pair are described.

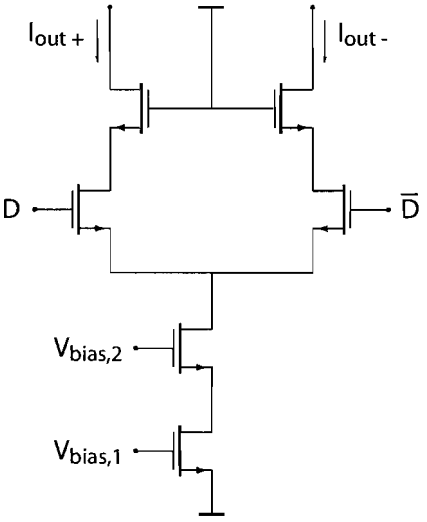



*Figure 6.5: Switched current cell circuit diagram*

### Current source

Practical current sources are non-ideal components, partly due to their finite output resistance which limits the static performance of the converter. The output resistance of the current cell is made as large as possible, in order to decrease this drop in performance to a negligible level (see also section 4.2). Here, cascoded current sources are employed implemented by NMOS transistors. Note that these components are always subject to device mismatch, resulting in differences between pairs of identically designed and biased current sources. This highly affects the DNL and INL error, thereby limiting the static performance of the DAC (see also section 4.1).

As a consequence of the above, conventional DAC designs have to meet stringent constraints in order to achieve a specific accuracy. Considering a $3\sigma$ confidence level, a relative spread of only 0.37% is tolerated for the design of a 12-bit binary-weighted current-steering DAC (see expression (4.3)). To meet this constraint, a relatively large unit cell area is required. In the original design of Radulov, transistors with an area of more than $200\mu m^2$ ($W/L = 16\mu m/14\mu m$) are used for the implementation of the unit current cells.

Using the approach described in chapter 5, 12 bit accuracy can be achieved using unit current cells with an extra-ordinary large relative spread of 7.5%. A $4\sigma$ confidence level is considered for the prevention of missing values between two consecutive output levels. Therefore, the overall yield of the converter will be less than a $4\sigma$ confidence level. According to expression (4.1), the area required to achieve a relative spread of 7.5% as a function of the gate overdrive voltage is given by:

$$WL_{min} = \frac{A_\beta^2 + 4A_{V_T}^2/(V_{GS} - V_T)^2}{2(\frac{\sigma_{unit}}{I_{unit}})^2} , \tag{6.1}$$

where $\frac{\sigma_{unit}}{I_{unit}}$ is the relative spread of a unit current cell, $A_\beta$ and $A_{V_T}$ are technology constants, and $(V_{GS} - V_T)$ is the gate overdrive voltage. As the biasing of the original

design is maintained, an overdrive voltage of 0.4 V is applied, yielding a unit cell area around $0.24\mu m^2$ ($W/L = 450nm/550nm$) [1]. This leads to a huge reduction of the unit cell area. Note that this reduction does not equal the final savings on silicon, since several other factors (layout rules, wiring, etc.) have to be taken into account and the size of the other building blocks is maintained.

#### Switch pair

The differential switch pair is an essential part of the switched current cell, as it directs the output current of the current cell to either the positive or negative output node of the converter. Important design issues are charge feedthrough, signal dependent modulation of the common source node and timing precision. All these issues are related to the dimensions of the switch pair, which scale with the output current of the current cell. The addition of cascode transistors has a positive effect on the performance, by reducing the charge feedthrough effect. The differential switch pair is reused from Radulov's design.

### 6.4.3 Unit-cell approach

Basically, the current sources are implemented by connecting a number of unit cells in parallel, also referred to as the unit-cell approach. The application of this approach on conventional current-steering DAC designs is a straight-forward operation, since the nominal output values of the current sources are always multiples of the unit cell. As shown by table 5.1, this is not the case using the design approach of chapter 5. Therefore, the iterative procedure of section 5.2.2 is adjusted in order to make it suitable for a unit-cell approach; every $\alpha_j$ is rounded down to the nearest multiple of unit-cells, before $\alpha_{j+1}$ is determined. This results in a little surplus of redundancy and an extra, but small decrement of the full-scale output range. However, by doing so a very large number of unit cells is required, thereby increasing the complexity of the design (around 36000 current cells, generating an output current of only $0.55\mu A$ each). A compromise was found by defining source $I_4$ as the unit cell, so that the 12 most significant current sources are implemented using a unit-cell approach. The four least significant sources ($I_0$ up to $I_3$) become fractions of $I_4$. Using this approach, around 3000 unit cells generating an output current of $6.6\mu A$ are required, by which the amount of unit cells required is comparable to a conventional design ($2^{12} - 1 = 4095$ unit cells). Moreover, the system remains accurate enough to meet the constraints of table 6.1. The five least significant sources are determined using the iterative procedure of section 5.2.2:

$$\alpha_j = \frac{x_j - \sqrt{x_j^2 - 4y_j}}{2} \qquad \text{for } 0 \leq j < 5 \,, \tag{6.2}$$

where $x_j$ and $y_j$ are given by:

$$x_j = 2\left(1 + \sum_{i=0}^{j-1} \alpha_i\right) + 4^2\left(\frac{\sigma_{unit}}{I_{unit}}\right)^2$$

[1] It is unknown if Pelgrom's methodology is also suitable for sub-micron technologies in combination with extra-ordinary large mismatches. Therefore, the obtained results should be handled with care.

$$y_j = \left(1+\sum_{i=0}^{j-1}\alpha_i\right)^2 - 4^2\left(\frac{\sigma_{unit}}{I_{unit}}\right)^2\sum_{i=0}^{j-1}\alpha_i \tag{6.3}$$

The 11 most significant sources are expressed by:

$$\alpha_j = \left\lfloor \frac{x_j - \sqrt{x_j^2 - 4y_j}}{2\cdot\alpha_4} \right\rfloor \cdot \alpha_4 \qquad \text{for } 5 \le j < k \tag{6.4}$$

Note that the choice for $I_4$ as unit cell is rather arbitrary; other alternatives may lead to better performance and/or smaller chip-area. Therefore, further investigation is recommended. Table 6.2 shows the nominal output values of all 16 current sources.

| j | Old nominal value of source $I_j$ | New nominal value of source $I_j$ |
|---|---|---|
| 0 | $0.74 \cdot I_{unit}$ | $0.08 \cdot I_4$ |
| 1 | $1.31 \cdot I_{unit}$ | $0.15 \cdot I_4$ |
| 2 | $2.42 \cdot I_{unit}$ | $0.28 \cdot I_4$ |
| 3 | $4.57 \cdot I_{unit}$ | $0.52 \cdot I_4$ |
| 4 | $8.78 \cdot I_{unit}$ | $1 \cdot I_4$ |
| 5 | $17.05 \cdot I_{unit}$ | $2 \cdot I_4$ |
| 6 | $33.39 \cdot I_{unit}$ | $3 \cdot I_4$ |
| 7 | $65.78 \cdot I_{unit}$ | $6 \cdot I_4$ |
| 8 | $130.16 \cdot I_{unit}$ | $12 \cdot I_4$ |
| 9 | $258.34 \cdot I_{unit}$ | $24 \cdot I_4$ |
| 10 | $513.87 \cdot I_{unit}$ | $48 \cdot I_4$ |
| 11 | $1023.78 \cdot I_{unit}$ | $95 \cdot I_4$ |
| 12 | $2041.97 \cdot I_{unit}$ | $190 \cdot I_4$ |
| 13 | $4076.02 \cdot I_{unit}$ | $379 \cdot I_4$ |
| 14 | $8140.84 \cdot I_{unit}$ | $757 \cdot I_4$ |
| 15 | $16265.84 \cdot I_{unit}$ | $1512 \cdot I_4$ |

*Table 6.2: Old and new values of 16 current sources suitable for a unit-cell approach, assuming $\sigma_{unit} = 7.5\%$ and $4\sigma$ confidence level*

## 6.5 Simulation results

In this section, the simulation results of a digitally pre-corrected 12-bit DAC with built-in redundancy are presented. The DAC consists of a digital pre-correction circuit and an analog core with built-in redundancy. The digital pre-correction algorithm is implemented by a Verilog-A model (see appendix A). The analog DAC core is designed for the UMC 0.18$\mu$m CMOS process. Note that most building blocks are reused from a previous DAC design. The only major difference is the implementation of the unit current sources, which are realized by NMOS transistors with a $W/L$ of only $450nm/550nm$. In the following sections, the simulation results are presented.

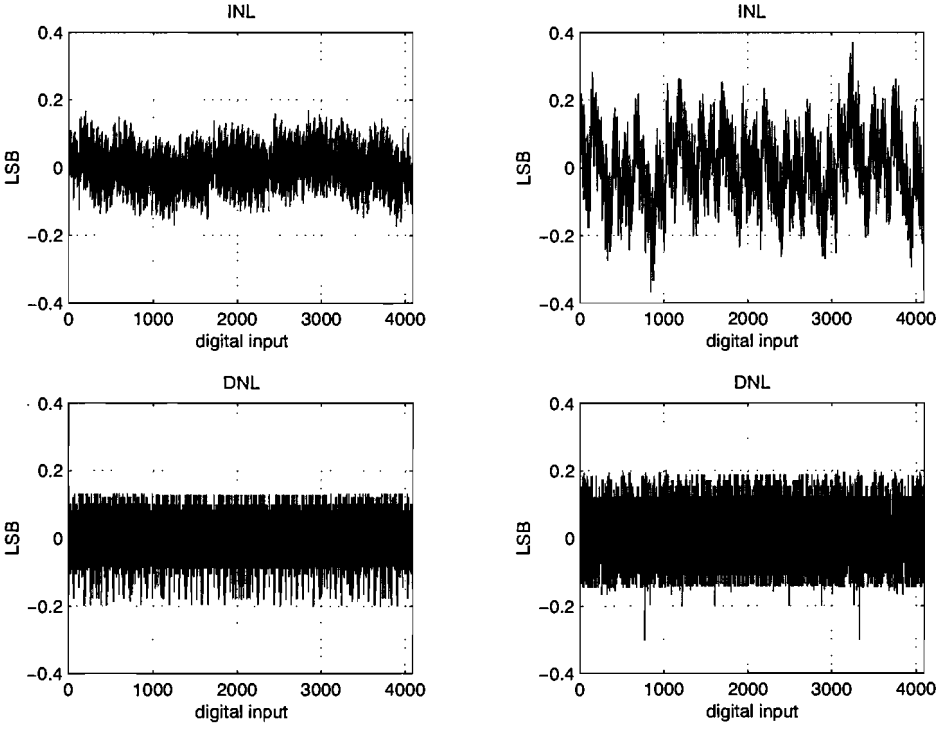
### 6.5.1 Static performance

The static performance of the design is evaluated in terms of INL and DNL. A differential ramp signal (code 0 up to 4095) is applied to the differential input terminals of the converter, in order to obtain a full transfer characteristic of the DAC. Subsequently, the data is processed for the derivation of the DNL and best-fitted INL, where 1 LSB is defined as the actual full-scale output range divided by 4095. Two different configurations are considered:

1. **Ideal current sources**, assuming ideally matched current sources. In that case, the actual output current will equal the nominal output current.

2. **Non-ideal current sources**, taking 7.5 % of relative unit cell spread into account, yielding an output current which deviates from the nominal value.

The simulation results are presented in figures 6.6(a) and 6.6(b). From these figures it can be concluded that the accuracy constraints of table 6.1 are fulfilled, even when a relative unit cell spread of 7.5% is taken into account. Monte-Carlo simulations could not be performed due to lack of time, since these simulations are very time-consuming.




(a) Ideal current sources

(b) Non-ideal current sources

*Figure 6.6: INL and DNL plots of digitally pre-corrected DAC with built-in redundancy*

### 6.5.2 Dynamic performance

In this section, the dynamic performance of the design is evaluated in terms of SFDR and SNDR. Dynamic properties are very important in order to determine the overall performance of the converter (see section 3.2). However, a detailed discussion about the dynamic properties is beyond the scope of this thesis. For that reason, only a limited number of simulations is performed.

The output spectrum of the converter is derived by applying a full-scale digitized sine-wave to the differential inputs of the DAC. The input signal has a frequency of 29 MHz, while the converter is clocked at 500 MHz. Three different configurations are considered:

1. **Binary-weighted DAC**, a normal 12-bit binary-weighted current-steering DAC is employed in order to make a comparison between the dynamic performance of a binary-weighted DAC and a digitally pre-corrected DAC. The DAC is also based on the design of Radulov. Ideally matched current sources are assumed.
2. **Pre-corrected DAC with ideal current sources**, a digitally pre-corrected 12-bit DAC with built-in redundancy, assuming ideally matched components.
3. **Pre-corrected DAC with non-ideal current sources**, as configuration 2a, but 5 % of relative unit cell spread taken into account.

As seen by figures 6.7, 6.8, and 6.9, the digitally pre-corrected DAC achieves a better SFDR in comparison to the binary-weighted DAC. This is due to the third harmonic, which is highly reduced in the pre-corrected DAC design. This suppression might be caused by the decreased unit cell area, as this effect is also observed after transistor level simulations of two normal binary-weighted current-steering DACs, employing small and large unit cells respectively. As a consequence of the reduced third harmonic, the pre-corrected DAC shows also an improved SNDR. It seems that current source mismatch errors do not affect the dynamic performance, as the spectra of figure 6.8 and 6.9 are almost similar. Another explanation could be the major influence of the other components on the dynamic performance of the DAC. This might dominate the dynamic performance, thereby masking the effect of the introduced mismatch errors. The reduced parasitic capacitance of the unit current cells may be profitable for the dynamic properties. Novel correction techniques may exploit this advantage, which might result in improved dynamic performance. Therefore, further investigation is recommended on this topic.

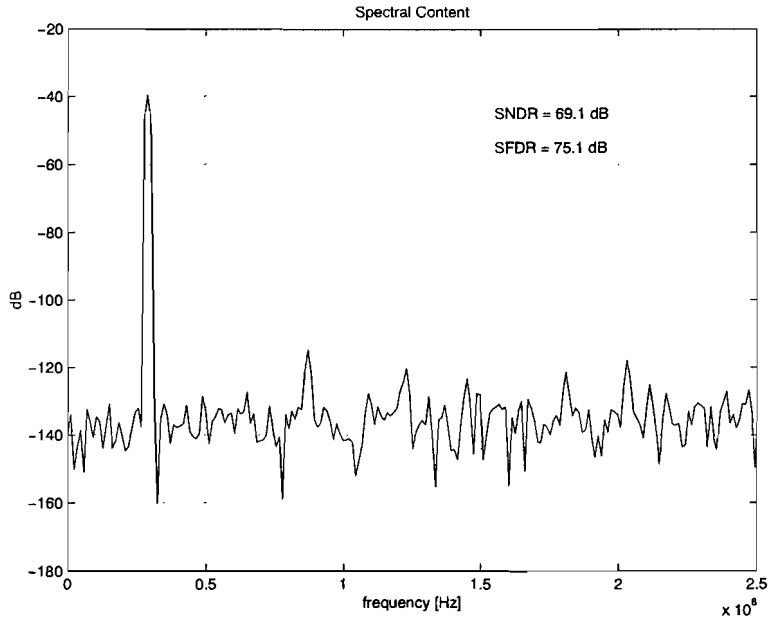



*Figure 6.7: Spectrum of a normal 12-bit binary-weighted current-steering DAC, operating at a clock frequency of 500 MHz. A sine-wave input signal with a frequency of 29 MHz is applied (configuration 1)*

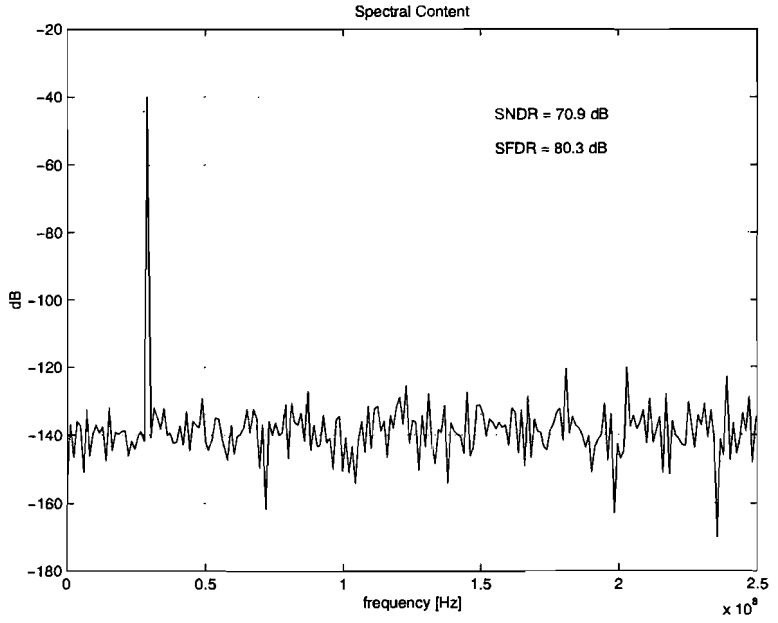



*Figure 6.8: Spectrum of a digitally pre-corrected 12-bit DAC with built-in redundancy, operating at a clock frequency of 500 MHz. A sine-wave input signal with a frequency of 29 MHz is applied (configuration 2)*

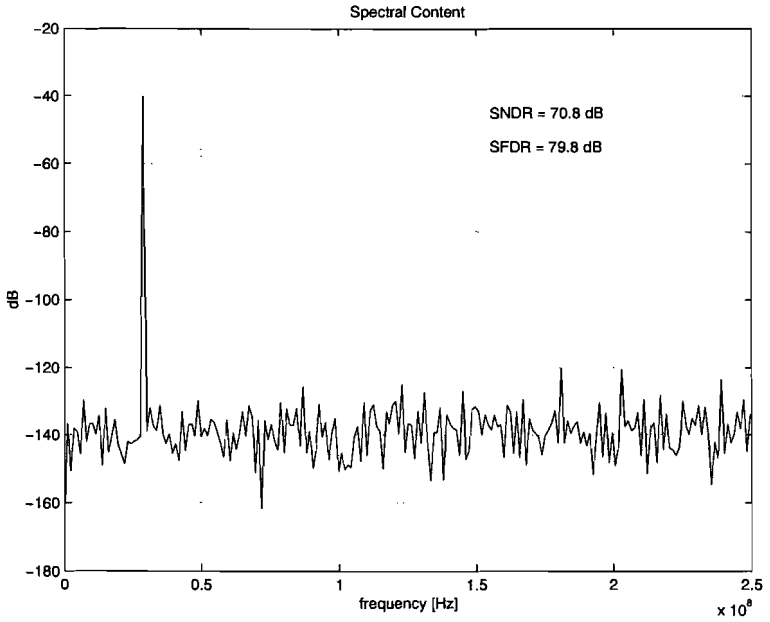



*Figure 6.9: Spectrum of a digitally pre-corrected 12-bit DAC with built-in redundancy, operating at a clock frequency of 500 MHz. A sine-wave input signal with a frequency of 29 MHz is applied. A relative unit cell spread of 5% is taken into account (configuration 3)*

# Chapter 7

# Conclusion

A design approach for current-steering DACs was presented, yielding high static performance and reliability. The required static accuracy is achieved by a combination of digital pre-correction, built-in redundancy and self-measurement, instead of by intrinsic design of the analog core.

A simple, efficient digital pre-correction algorithm was developed showing near-ideal behavior, while the required computational power was minimized. An iterative procedure to obtain exactly as much redundancy as needed was derived, thereby preventing the occurrence of missing output values. A simple self-measurement circuit was developed, which is realizable on-chip. However, this self-measurement circuit becomes the source of concern in the presented approach, by dominating the static performance. Nevertheless, the combination of digital pre-correction, built-in redundancy and self-measurement obtains sufficient performance to meet the design constraints.

Simulation results show that digitally pre-corrected DACs achieve high static performance while using small, inaccurate unit current cells. Furthermore, they achieve improved static performance and reliability in comparison to normal binary-weighted current-steering DACs, as was verified by simulations on both system level and transistor level. For this purpose, a digitally pre-corrected 12-bit D/A converter with built-in redundancy was designed in CMOS technology.

# Chapter 8

# Recommendations

### Design of the DAC core

The iterative procedure, determining the required amount of redundancy was adjusted, in order to implement the current sources using a unit-cell approach. For this purpose, the output value of a unit cell is redefined, in which the new definition is rather arbitrary. Further investigation is recommended, in order to obtain a solution that achieves optimal performance and/or minimal chip-area.

### Dynamic performance

The dynamic performance of the converter might be affected by the reduced parasitic capacitance of the unit cells and/or by mismatch errors. However, these topics require further investigation. Depending on the results, the digital pre-correction algorithm could be extended, in order to gain dynamic performance. In that case, state-of-the-art converters can be achieved, combining both high resolution and high speed.

### Field of application

It would be interesting to investigate whether other architectures can take advantage of built-in redundancy as well. This will increase the field of application of the presented approach.

# Bibliography

[1] Allen, P.E. and D.R. Holberg
CMOS ANALOG CIRCUIT DESIGN
New York: Oxford University Press, 2002

[2] Bastos, J. CHARACTERIZATION OF MOS TRANSISTOR MISMATCH FOR ANALOG DESIGN
Ph.D. dissertation, Katholieke Universiteit Leuven, 1998

[3] Doris, K. HIGH SPEED D/A CONVERTERS: FROM ANALYSIS AND SYNTHESIS CONCEPTS TO IC IMPLEMENTATION
Ph.D. dissertation, Eindhoven University of Technology, 2004

[4] Harpe, P.J.A. and J.M. de Meulmeester, J.A. Hegt, A.H.M. van Roermund
NOVEL DIGITAL PRE-CORRECTION METHOD FOR MISMATCH IN DACS WITH BUILT-IN SELF-MEASUREMENT
In: Proceedings of the IEE ADDA 2005, Limerick, July 25-27, 2005

[5] Harpe, P.J.A. and J.M. de Meulmeester, J.A. Hegt, A.H.M. van Roermund
RELIABLE DESIGN OF DIGITAL-TO-ANALOG CONVERTERS USING PRE-CORRECTION AND EMBEDDED SELF-TEST
In: Proceedings of the IEEE IMSTW 2005, Cannes, June 27-29, 2005

[6] Jespers, P.G.A
INTEGRATED CONVERTERS: D TO A AND A TO D ARCHITECTURES, ANALYSIS AND SIMULATION
New York: Oxford University Press, 2001

[7] Pastre M. and M. Kayal HIGH-PRECISION DAC BASED ON A SELF-CALIBRATED SUB-BINARY RADIX CONVERTER
In: Proceedings of the 2004 International Symposium on Circuits and Systems, Vancouver, May 23-26, 2004
IEEE, 2004. Vol. 1, p. 341-344

[8] Pelgrom, M.J.M. and A.C.J. Duinmaijer, A.P.G. Welbers
MATCHING PROPERTIES OF MOS TRANSISTORS

IEEE Journal of Solid-State Circuits, Vol. 24 (1989), No. 5, p. 1433-1440

[9] Plassche, R. van de
CMOS INTEGRATED ANALOG-TO-DIGITAL AND DIGITAL-TO-ANALOG CONVERTERS, 2nd ed.
Dordrecht: Kluwers Academic Publishers, 2003

[10] Radulov, G.I.
DESIGN OF A CALIBRATED 12-BIT CURRENT-STEERING DIGITAL-TO-ANALOG CONVERTER
PDEng Thesis, Eindhoven University of Technology, 2004

[11] Razavi, B.
PRINCIPLES OF DATA CONVERSION SYSTEM DESIGN
Piscataway: IEEE Press, 1995

[12] Wikner, J.J. and Nianxiong Tan
MODELING OF CMOS DIGITAL-TO-ANALOG CONVERTERS FOR TELECOMMUNICATION
IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Vol. 46 (1999), Iss. 5, p. 489-499

# Appendix A

# Description of digital pre-correction in Verilog-A

*This appendix contains the Verilog-A description of the digital pre-correction circuit (see section ).*

### External view

In this section, the external view of the digital pre-correction model is presented. The model has two differential input terminals: $X$ and $\overline{X}$. The differential outputs $Q$ and $\overline{Q}$ are divided in two parts: MSB nodes, which are implemented using the unit cell approach, and LSB nodes, which are fractions of the unit cell. The model is shown in figure A.

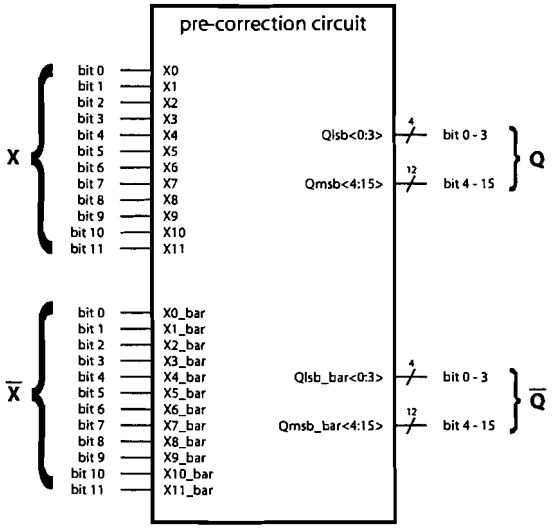



*Figure A.1: External view of the digital pre-correction model*

### Internal view

The Verilog-A description of the digital pre-correction circuit is given below:

```
`include "constants.h"
`include "discipline.h"

module PC(X0, X0_bar, X1, X1_bar, X2, X2_bar, X3, X3_bar, X4, X4_bar,
          X5, X5_bar, X6, X6_bar, X7, X7_bar, X8, X8_bar, X9, X9_bar,
          X10, X10_bar, X11, X11_bar, Qlsb, Qlsb_bar, Qmsb, Qmsb_bar);

  input X0, X0_bar, X1, X1_bar, X2, X2_bar, X3, X3_bar, X4, X4_bar,
          X5, X5_bar, X6, X6_bar, X7, X7_bar, X8, X8_bar, X9, X9_bar,
          X10,   X10_bar, X11,X11_bar;

  output [0:3] Qlsb; output [0:3] Qlsb_bar;
  output [4:15] Qmsb; output [4:15] Qmsb_bar;

  electrical X0, X0_bar, X1, X1_bar, X2, X2_bar, X3, X3_bar, X4, X4_bar,
          X5, X5_bar, X6, X6_bar, X7, X7_bar, X8, X8_bar, X9, X9_bar,
          X10, X10_bar, X11, X11_bar;

  electrical [0:3] Qlsb; electrical [0:3] Qlsb_bar;
  electrical [4:15] Qmsb; electrical [4:15] Qmsb_bar;

  parameter real Trise = 0 from [0:inf);
  parameter real Tfall = 0 from [0:inf);
  parameter real Tdelay = 0 from [0:inf);
  parameter real VlogicHigh = 1.8;
  parameter real VlogicLow = 0;
  parameter real I0 = 1.0000;
  parameter real I1 = 1.5000;
  parameter real I2 = 2.2500;
  parameter real I3 = 3.3750;
  parameter real I4 = 5.0625;
  parameter real I5 = 7.5938;
  parameter real I6 = 11.3906;
  parameter real I7 = 17.0859;
  parameter real I8 = 25.6289;
  parameter real I9 = 38.4434;
  parameter real I10 = 57.6650;
  parameter real I11 = 86.4976;
  parameter real I12 = 129.7463;
  parameter real I13 = 194.6195;
  parameter real I14 = 291.9293;
  parameter real I15 = 437.8939;

  real Imeas[0:15];
  real Out[0:15];
  real Out_bar[0:15];
  real Xin[0:15];
  real Xin_bar[0:15];
  real Residue, FullScale, X;
  integer In, i, j;
```

```
analog begin

    Imeas[0] = I0;
    Imeas[1] = I1;
    Imeas[2] = I2;
    Imeas[3] = I3;
    Imeas[4] = I4;
    Imeas[5] = I5;
    Imeas[6] = I6;
    Imeas[7] = I7;
    Imeas[8] = I8;
    Imeas[9] = I9;
    Imeas[10] = I10;
    Imeas[11] = I11;
    Imeas[12] = I12;
    Imeas[13] = I13;
    Imeas[14] = I14;
    Imeas[15] = I15;
        FullScale = 0;

    for (i = 0; i <= 15 ; i = i + 1) begin
        FullScale = FullScale + Imeas[i];
    end

    Xin[0] = V(X0); Xin_bar[0] = V(X0_bar);
    Xin[1] = V(X1); Xin_bar[1] = V(X1_bar);
    Xin[2] = V(X2); Xin_bar[2] = V(X2_bar);
    Xin[3] = V(X3); Xin_bar[3] = V(X3_bar);
    Xin[4] = V(X4); Xin_bar[4] = V(X4_bar);
    Xin[5] = V(X5); Xin_bar[5] = V(X5_bar);
    Xin[6] = V(X6); Xin_bar[6] = V(X6_bar);
    Xin[7] = V(X7); Xin_bar[7] = V(X7_bar);
    Xin[8] = V(X8); Xin_bar[8] = V(X8_bar);
    Xin[9] = V(X9); Xin_bar[9] = V(X9_bar);
    Xin[10] = V(X10); Xin_bar[10] = V(X10_bar);
    Xin[11] = V(X11); Xin_bar[11] = V(X11_bar);
    X = 0; X_bar = 0; Residue = 0;

    generate i (11,0) begin
    if ( Xin[i] >= (VlogicHigh+VlogicLow)/2 ) begin
            X = X + (1 << i);
    end else begin
        X = X - (1 << i);
    end
    if ( Xin_bar[i] < (VlogicHigh+VlogicLow)/2 ) begin
            X_bar = X_bar + (1 << i);
    end else begin
        X_bar = X_bar - (1 << i);
    end
    end

    Residue = (X - X_bar) / 4095;

    generate j (15,0) begin
    if ( Residue >= 0 ) begin
        Residue = Residue - (Imeas[j]/FullScale);
```

```
            Out[j] = VlogicHigh;
            Out_bar[j] = VlogicLow;
        end else begin
            Residue = Residue + (Imeas[j]/FullScale);
            Out[j] = VlogicLow;
            Out_bar[j] = VlogicHigh;
        end
        end

    V(Qlsb[0]) <+ transition( Out[0], Tdelay, Trise, Tfall );
    V(Qlsb[1]) <+ transition( Out[1], Tdelay, Trise, Tfall );
    V(Qlsb[2]) <+ transition( Out[2], Tdelay, Trise, Tfall );
    V(Qlsb[3]) <+ transition( Out[3], Tdelay, Trise, Tfall );
    V(Qmsb[4]) <+ transition( Out[4], Tdelay, Trise, Tfall );
    V(Qmsb[5]) <+ transition( Out[5], Tdelay, Trise, Tfall );
    V(Qmsb[6]) <+ transition( Out[6], Tdelay, Trise, Tfall );
    V(Qmsb[7]) <+ transition( Out[7], Tdelay, Trise, Tfall );
    V(Qmsb[8]) <+ transition( Out[8], Tdelay, Trise, Tfall );
    V(Qmsb[9]) <+ transition( Out[9], Tdelay, Trise, Tfall );
    V(Qmsb[10]) <+ transition( Out[10], Tdelay, Trise, Tfall );
    V(Qmsb[11]) <+ transition( Out[11], Tdelay, Trise, Tfall );
    V(Qmsb[12]) <+ transition( Out[12], Tdelay, Trise, Tfall );
    V(Qmsb[13]) <+ transition( Out[13], Tdelay, Trise, Tfall );
    V(Qmsb[14]) <+ transition( Out[14], Tdelay, Trise, Tfall );
    V(Qmsb[15]) <+ transition( Out[15], Tdelay, Trise, Tfall );

    V(Qlsb_bar[0]) <+ transition( Out_bar[0], Tdelay, Trise, Tfall );
    V(Qlsb_bar[1]) <+ transition( Out_bar[1], Tdelay, Trise, Tfall );
    V(Qlsb_bar[2]) <+ transition( Out_bar[2], Tdelay, Trise, Tfall );
    V(Qlsb_bar[3]) <+ transition( Out_bar[3], Tdelay, Trise, Tfall );
    V(Qmsb_bar[4]) <+ transition( Out_bar[4], Tdelay, Trise, Tfall );
    V(Qmsb_bar[5]) <+ transition( Out_bar[5], Tdelay, Trise, Tfall );
    V(Qmsb_bar[6]) <+ transition( Out_bar[6], Tdelay, Trise, Tfall );
    V(Qmsb_bar[7]) <+ transition( Out_bar[7], Tdelay, Trise, Tfall );
    V(Qmsb_bar[8]) <+ transition( Out_bar[8], Tdelay, Trise, Tfall );
    V(Qmsb_bar[9]) <+ transition( Out_bar[9], Tdelay, Trise, Tfall );
    V(Qmsb_bar[10]) <+ transition( Out_bar[10], Tdelay, Trise, Tfall );
    V(Qmsb_bar[11]) <+ transition( Out_bar[11], Tdelay, Trise, Tfall );
    V(Qmsb_bar[12]) <+ transition( Out_bar[12], Tdelay, Trise, Tfall );
    V(Qmsb_bar[13]) <+ transition( Out_bar[13], Tdelay, Trise, Tfall );
    V(Qmsb_bar[14]) <+ transition( Out_bar[14], Tdelay, Trise, Tfall );
    V(Qmsb_bar[15]) <+ transition( Out_bar[15], Tdelay, Trise, Tfall );

end
endmodule
```