A Reconfigurable Calibration Method for Current-Steering DACs

Gonçalo Nogueira
INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal
Email: goncalo.nogueira@tecnico.ulisboa.pt

Abstract—A new calibration method for current steering DACs based on dynamically rearranging the commutation sequence of thermometer coded current sources, also called switching-sequence post-adjustment (SSPA) is presented. This calibration method conducts a search for the optimal commutation sequence that maximizes any desired performance metric instead of generating it based on ad-hoc or heuristic arguments. An add-on to the calibration method based on the correction of the output current is also proposed. The proposed calibration method improves the INL up to 9 times and the ENOB up to 1.1 bit when compared to another state-of-the-art SSPA calibration method.

Index Terms—Calibration method, current-steering, DAC, optimal commutation sequence, reconfigurable, SSPA.

I. INTRODUCTION

The most common DAC architecture used nowadays is the segmented current-steering architecture, as it presents high-speed and low area, comparatively to other existent DAC architectures. Segmentation is commonly used as it combines the advantages of the binary-weighted architecture and thermometer-coded architecture, reducing the DNL, area and complexity of the circuit [1].

Despite the segmentation advantages, the transistors of the current sources are affected by process variations, which affect the current value of the current sources present in the DAC, depreciating the DNL and INL. A method to decrease the effect of process variations is to select which MSB current sources to turn on at each code instead of choosing them randomly [2]. In order to select the best MSB current sources they are compared one with another and the result is stored, being the data gathered used by an external processing unit that determines the optimal commutation sequence.

The method has been explored by other authors [3],[4], but only commutation sequences based on ad-hoc or heuristic arguments have been used so far. Also, when using this method, it was only measured if the current of a MSB current source was higher than another (1 bit difference quantization).

This paper presents a calibration method for segmented or thermometer coded current steering DACs based on the selection of the optimal commutation sequence that minimizes or maximizes a desired performance. As a proof of concept, part of a 12-bit segmented current steering DAC is designed, using a 6-6 segmentation, and using the minimum area possible, as this is a key feature in DAC design. The DAC can be calibrated as many times as desired and for any performance metric. The calibration method has three steps: (1) sorting the thermometer coded current sources by increasing current value, (2) measuring the current produced by each thermometer coded current source, using more than 1 bit difference quantization, and (3) finding the optimal commutation sequence. An add-on is proposed, consisting on the correction of the output error in realtime, during D/A conversion.

The organization of the paper is as it follows: Section II presents the DAC’s architecture. Section III describes the calibration method in detail, including the correction of the INL error during conversion. The MSB current sources and comparator that were designed in this work are described in Section IV. The simulations performed can be seen in Section V, where the results for the proposed calibration method and the add-on are compared to the results with the calibration method in [4]. Finally, Section V presents the conclusions obtained from this work.

II. ARCHITECTURE

The architecture of the DAC is divided in 6 blocks (see Fig.1): the Control Unit, the Decode/Memory, the MSB and LSB Current Source arrays, the Correction Current Source Array and the Comparator.

![Fig. 1. Block diagram of the DAC system.](image)

The Control Unit is a digital block that controls the operations of the DAC and communicates with the external microprocessor. The Decode/Memory is a writable digital block that stores and decodes information from the Control Unit and that controls the enabling of the current sources. The LSB Array block has the LSB current sources and the MSB Array block has the MSB current sources. The Correction Array block has the correction current sources, which allow the measurement of the current of each MSB current source...
and the correction of the INL during conversion, by adding or subtracting current.

In order to decrease area, the program with the calibration algorithm is stored and ran in the external microprocessor, usually available in embedded systems. If the Decode/Memory is designed with a non-volatile memory, then it is possible to calibrate the DAC upon fabrication, requiring no further calibration. On the other hand, if required, the calibration may be changed at any time to optimize a different performance criterion.

III. CALIBRATION METHOD

In this calibration method the LSB current sources are despised, as the calibration is performed on the MSB current source array, which is thermometer coded. As so, the calibration method is based in thermometer coded DACs. For these DACs, it can be shown, using the IEEE Std 1658-2011 [5], that the DNL of a code is only dependent of the current source that was enabled for that code, considering only process variations. Let the current of a current source be defined as

\[ I_k = \Delta + e_k \]  

(1)

with \( \Delta \) the quantization step and \( e_k \) the error due to process variations. Using the definition of DNL from [5] it is obtained that

\[ \text{DNL}(D_{in}) = e_{D_{in}}. \]  

(2)

with \( D_{in} \) the code at the input of the DAC. Using the definition of DNL and INL from [5] for any DAC it is obtained that

\[ \text{INL}(D_{in}) = \sum_{k=1}^{D_{in}} \text{DNL}(k). \]  

(3)

considering zero offset.

The calibration method uses these formulas to estimate the DNL and INL values at each code, depending on the commutation sequence used.

A. MSB Current Source Sorting

Initially the MSB current sources of the Decode/Memory block are not sorted and present random values. As so, inside the external microprocessor a pointer array is created to address the MSB current sources using the Decode/Memory block.

The comparator block in Fig.1 compares between two current sources which one has the higher current value. The result is transmitted to the Control Unit, and therefore, to the external microprocessor. Using the Comparator block and the quicksort algorithm [6] the pointer array is sorted by increasing current value. The quicksort algorithm is used as it in-place, that is, it uses a small auxiliary stack, and has \( O(N\log(N)) \) complexity with \( N \) the number of elements to sort.

In [2] the authors design the MSB current source array with small but extra current sources (higher process variation), selecting only the best ones for the sorting algorithm. This decreases the area spent with the MSB current source array but maintains the DNL of the array. In this work this is also done, being the MSB current source array designed with 2 extra current sources and reducing the area by 18.5%.

B. MSB Current Source Current Measurement

With the pointer array sorted, the MSB current source with the median current value is defined as the master current source and is used as a standard to measure the current of each MSB current source. To do this, an MSB current source is compared with the master current source in parallel with current sources from the Correction Array block. This block produces current in steps of \( 2^{-5} \) LSB, allowing to measure the difference between the current produced by the master current source and by another MSB current source. Each MSB current source’s difference to the master current source is measured in \( d_{\text{step}} \) and stored in a difference array in the external microprocessor with a discrete value. Each \( d_{\text{step}} \) corresponds to \( 2^{-5} \) LSB of current.

The current generated by each MSB current source is presented in Fig.2, following the sorted pointer array sequence, being the master current source marked as a filled circle and the excluded MSB current sources marked with a cross. The calibration method is designed with 128 intervals, although only 16 are represented (lines).

\[ d_{\text{est}}[k] = \frac{n_{\text{dif,smaller}}[k] + 1}{n_{\text{dif}}[k] + 1} + d_{\text{mes}}[k] \]  

(4)

Fig. 2. Plot of the current generated by each MSB current source, following the pointer array sequence, and difference interval limits.

C. Optimal Commutation Sequence Search

In this step of the calibration method, all the calculations are done exclusively by the external microprocessor.

Although the difference between the current produced by the master and MSB current sources has been measured, it is needed to estimate these values, passing them from discrete to analog values. This is made taking into account the number of MSB current sources with the same difference value. The following expression is used to estimate the difference to the master current source.
where \(d_{ext}\) is the array with the estimated difference values, \(d_{mes}\) is the difference array, \(n_{dif}[k]\) is the number of current sources with the measured difference value \(d_{mes}[k]\) and \(n_{dif}[k]\) is the number of current sources with the measured difference value \(d_{mes}[k]\) with smaller current production than the current source \(k\) (this is verified with the pointer array). For the estimation of the difference, the master current source does not enter into account, being the value of this in \(d_{ext}\) equal to zero.

After this, the DNL associated to each MSB current source is estimated. To do this it is necessary to calculate the difference of current between the current produced by a MSB current source and a MSB quantization step, as seen in (1) and (2). A MSB quantization step corresponds to \(2^{N_{LSB}} \Delta\), with \(N_{LSB}\) the number of LSB bits of the DAC and \(\Delta\) the quantization step of the DAC. The MSB quantization step is estimated as being equal to the sum of the elements of the quantization step of the DAC. The pairing of MSB current sources is done by distance to the current source

\[\text{pair if they were added individually. The DNL stops being}\]

\[\text{randomly between the two [4]. In this way, the DNL of}\]

\[\text{expected noise sources, is easy to compute, being chosen as}\]

\[\text{metric to reduce. It is possible to show that in a thermometer-}\]

\[\text{coded current-steering DAC the quantization noise power is}\]

\[\text{given by}\]

\[P_{quant,N} = \frac{\Delta^2}{\sqrt{12}} + \left( \frac{\sum_{D_{in}=1}^{n} \text{INL}(D_{in})^2}{n} \right)\]

(5)

where \(n\) is the number of current sources, considering that \(\text{INL}(0) = 0\) and \(\text{INL}(n) = 0\). The objective function used to improve the ENOB calculates the sum of the square of the INL of all codes returning a value which is desired as small as possible. This is done using two accumulators, one for the calculation of the INL and the other for the value returned.

Any characteristic known beforehand, like, for instance, the finite output impedance of the MSB current sources, can be added to the calibration method, making the expected output current adapt to that characteristic. If this characteristic is used in the objective function, the characteristic is not permuted, as it is only code dependent.

The more vertices there are in the source queue and the bigger the number of transpositions between each, the better the graph will be searched, as it starts from “distant” vertices, having more chances of converging to better local optima. The source vertices used in the algorithm are shifts of the pointer array. The optimal queue has the vertices that present the best values for the desired performance, in ascending order, having been designed with 6 positions. The algorithm starts with the worst value possible in every position of the optimal queue and no vertices.

In the beginning of the search, the following loop is run: a vertex is removed from the source queue and marked, it’s value is calculated using the objective function, and it is checked if the value obtained is better or worse than the first position of the optimal queue. If it is better, then the vertex is inserted in the queue and the vertices with worse values are shifted, being the worst, this is, the first of the queue, removed. Otherwise the vertex is not put on the queue. After this, the values of the adjacent vertices of the source vertex are calculated, being put in the optimal queue only the vertices that have the best values. These vertices, which are unmarked, are put onto the optimal queue in the same way as the source vertices. The loop only ends when the source queue is empty. In Fig.3 a flowchart with this algorithm can be seen.

After this, the following loop is run: the first unmarked vertex of the optimal queue is marked. Then, the unmarked adjacent vertices to this vertex are checked and put onto the queue, the same way as the source vertices, if they present a better value than the first vertex of the optimal queue. This loop is repeated until all the vertices in the optimal queue are marked. In Fig.4 a flowchart with this algorithm can be seen.
The permutation that leads to optimal commutation sequence is the one corresponding to the vertex in the last position of the queue. This permutation is applied to the pair pointer array, that is split back into the original pairs, being the resultant commutation sequence the one used by the DAC.

An add-on used on this calibration method is the use of the current produced by the Correction Array block during D/A conversion to adjust the output current in realtime, as this block produces current in steps of $2^{-5}$ LSB that can be added or subtracted.

In order to do this, before pairing MSB current sources, to each MSB current source of the estimated DNL array is added a multiple of $d_{\text{step}}$ is added, such that after, the estimated DNL value is inferior or equal to $0.5d_{\text{step}}$. The multiple of $d_{\text{step}}$ added to each MSB current source is stored in a correction array. This may turn the sum of the array with estimated difference values different from zero, but this is corrected during the search, as the objective function when the add-on is used implements correction.

The current sources are then paired as explained previously and the search starts. During the search, when the objective function calculates an INL value, it verifies if adding or subtracting a $d_{\text{step}}$ would reduce the INL. If it does then it is performed, being the change recorded. The objective function not only returns the value of the performance but also an array with the $d_{\text{step}}$ added.

When the search is over, to the pair pointer array is applied the optimal permutation. This array is split back into the commutation sequence and the array with the multiple of $d_{\text{step}}$ is permuted to make correspond each multiple of $d_{\text{step}}$ with the correspondent MSB current source. After this, if the objective function of the permutation resulting from the search returned an array with any $1$ or $-1$ values (the only values possible different from zero), these values are added to the MSB current source of the correspondent pair with the smallest or highest multiple of $d_{\text{step}}$, respectively. The last step consists on taking the cumulative values added at the output during D/A conversion. Therefore, to all the values in the correction array must be added the sum of all previous $d_{\text{step}}$ pair values. For instance, if the array at the second to last step is $\{-2, 1; -4, 5\}$, with the ';' separating the pairs, the final array is $\{-2, 1; -5, 4\}$, as the multiple of $d_{\text{step}}$ accumulates ($-1$ is added to the second pair).

The implementation of the add-on in the sorting algorithm comes at the expense of more area in the Decode/Memory block as more bits are needed to address the correction current sources at each code and more area in the Correction Array block as more current sources are needed.
IV. Designed Circuits

In the work developed a MSB current source array and comparator were designed. These are explained in subsections IV-A and IV-B respectively. The DAC circuit is designed in a 130 nm CMOS technology with a 1.2 V supply voltage, an output swing of 0.5 V and using a 200Ω output resistor.

A. MSB current source

The MSB current source designed is the one in Fig.5. The transistor $PM_1$ and $PM_2$ are the current source and cascode transistors, respectively. $PM_3 - PM_6$ are the switch transistors, that commute the current between branches and $PM_7 - PM_{10}$ are a cascode to the switch transistors.

Four branches are used, one for the output, other for calibration and the other two as dummies. Only one of the branches of the MSB current source is enabled every time, being the current produced by $PM_1$ commuted from one branch to another when required, which is faster than enabling or disabling the current source cell. A cascode transistor ($PM_7 - PM_{10}$) for the switches ($PM_3 - PM_6$) is used as this decreases the distortion of the output signal [7].

The transistors $PM_1$ and $PM_2$ are implemented with high speed transistors, as these present the best matching and output resistance values. The transistors used for $PM_3 - PM_{10}$ are high speed low $V_{th}$ transistors as these present the smallest parasitic capacitances, resulting in a faster circuit.

The transistors $PM_1$ and $PM_2$ are biased with a cascoded current mirror, and provide a high output resistance. The transistors $PM_7 - PM_{10}$ are biased with ground, which puts them in triode region. The latter transistors have a smaller output impedance by being in triode than in saturation region, nevertheless, this problem is solved with the addition of the current characteristic of the MSB current source into the objective function.

Due to the number of MSB current sources, the output swing of 0.5 V and the 200Ω resistor to convert the current, each MSB current source produces 39.06 µA. From the analysis performed with the extra current sources, in order to have the MSB current sources with a deviation less than 2 LSB with a 99% yield, the current sources have to be designed with $\sigma/I = 1.15\%$. The finite output impedance of each MSB current source does not affect the INL more than 2 LSB. This was measured by calculating the average output current characteristic as a function of the output voltage out of a 500 runs Monte Carlo simulation. After this, the difference between the average and ideal output current characteristics was calculated and at each code multiplied by the number of MSB current sources enabled at that code. The result can be seen in the simulated characteristic in Fig.6. This characteristic is used in the simulations and minimized by the calibration method.

After the design, each MSB current source occupies 12.9 µm², which multiplied by 65 equals 838 µm².

B. Comparator

Inside the comparator block of Fig.1 there is a resistor which converts the current from a MSB current source to 0.5 V. This value is sampled and hold in a capacitor. There are two capacitor that sample and hold voltages originated in different current sources, being the comparison executed when the sampling of the two has been finished. This comparison is realized in a comparator, whose design can be seen in Fig.7. The architecture of this comparator is based on the one presented in [8], with the difference that the circuit is built with and for a PMOS differential input pair and capacitors are added in the second stage.

The comparator is designed to have the smallest area possible, an input offset of 50 µV and an input referred noise voltage of 200 µV.

In order to decrease the noise of the comparator the transistors used for $M_{2P}$, $M_{2N}$, $M_{6P}$ and $M_{6N}$ are thick oxide transistors, as these present a higher threshold voltage than the high speed transistors and put $M_{2P}$ and $M_{2N}$ in weak inversion during a comparison [8], [9]. The capacitors added to the second stage are used to decrease the input referred noise power as the gain provided by the first stage is not enough to decrease the noise of the second stage to an insignificant level.

The calibration of the input offset of the comparator is done with the use of variable capacitors, based on [10], being composed of four adjustable capacitor cells that are depicted in Fig.8. These cells change the input offset of the comparator.

Fig. 5. Designed MSB current source cell.

Fig. 6. Estimated output current average characteristic of the DAC due to the finite impedance of the current sources as a function of the input code.
Fig. 7. Schematic of the circuit of the comparator.

from 50 µV up to 800 µV. The comparator circuit is designed to, upon fabrication, have an input offset inferior to 800 µV with a 2.75σ yield.

In order to reduce the noise sensitivity of the comparator, when comparing two currents the comparison is done 21 times, being chosen as the highest current the current that has been selected more times as higher.

The total area occupied by the designed comparator block is 95.1 µm².

V. SIMULATIONS AND RESULTS

Four simulations using a MATLAB model were done to test the calibration method explained in section III. These are compared to the sorting algorithm described in [4], which performs the pairing of the MSB current sources and calculates the commutation sequence using an heuristic sorting algorithm.

In the simulations performed the current sources are affected by process variations and the comparator has a 50 µV input offset. The simulation runs for 5 different number of intervals used to measure the difference of current between the MSB current source and the master current source (64, 128, 256, 512 and 1024). Nevertheless, to simplify, the results shown are only for the simulation obtained when using 128 error intervals. Each simulation generates 100000 MSB current source arrays and after a calibration method has been applied, the resultant INL and ENOB are calculated. The results are presented as a function of the input referred noise voltage of the comparator.

In the first two simulations the output impedance of the MSB current sources used is infinite while in the last two simulations the finite output impedance characteristic of the designed MSB current sources is used, being added the characteristic in Fig.6 to the output. In the first three simulations the objective function that improves the INL is used, being the objective function that improves the ENOB used in the fourth. High frequency effects are not simulated.

The first simulation compares the INL obtained with the proposed calibration method, with the calibration method in [4] and with no calibration method. The INL results for the first two are in Fig.9 where the INL for the proposed calibration method is marked with crosses, obtaining an INL between 0.3 LSB and 0.33 LSB, and the calibration method in [4] is marked with circles obtaining an INL between 0.62 LSB and 0.65 LSB. The INL obtained with no calibration method was 4.7 LSB.

Fig. 9. INL for the proposed calibration method of this work, and for the calibration method in [4] as a function of the input referred noise voltage of the comparator (crosses and circles, respectively).

The second simulation compares the INL obtained with the proposed calibration method with the add-on and the calibration method in [4] adapted with the add-on. The INL results are in Fig.10 where the INL for the proposed calibration method of this work and for the calibration method in [4] are marked with squares and asterisks, respectively. The results show that these two INL characteristics nearly match, varying

![Fig. 8. Circuit used as variable capacitor in Fig.7 (C₂).](image)
from 0.18 LSB up to 0.21 LSB, which means that if the add-on is used, the commutation sequence used becomes irrelevant. that reaches an ENOB of 11.6 bit and lastly the calibration method in [4] reaching an ENOB of 10.8 bit.

Fig. 10. INL for the proposed calibration method with add-on, and for the calibration method in [4] adapted with add-on, as a function of the input referred noise voltage of the comparator (squares and asterisks, respectively).

In the third simulation the proposed calibration method with and without the add-on and the calibration method in [4] are implemented. The INL results for the latter can be seen in Fig.11 where they are marked with squares, crosses and circles, respectively. The calibration method in [4] does not take the deviation characteristic of the MSB current sources into account, so it is expected for its INL to round 2 LSB, as the INL deviation caused by the finite output resistance of the MSB current source reaches 1.8 LSB. The proposed calibration method of this work without add-on does not achieve very good results for this simulation as it cannot follow the deviation characteristic. This is due to the pairing technique that decreases the DNL characteristic of the DAC significantly.

The fourth simulation is similar to the third, targeting the ENOB instead of the INL and calculating it at low-frequency, that is, not taking high frequency effects into account. A fullscale sinusoidal signal with 4 periods was generated using the current sources. From this signal the ENOB was calculated. The results for the ENOB in this simulation are in Fig.12 where they are marked with squares, crosses and circles, respectively. The calibration method in [4] adapted with add-on, presenting better results for the former case but with the penalty of more die area at the Decode/Memory block, being advisable to use the add-on in case the specifications desired are not reached. The results presented show that the INL and the ENOB achieved by the calibration method with and without add-on are significantly better than those obtained for the calibration method presented in [4], one of the best SSPA calibration methods known.

VI. CONCLUSION
A calibration method for current steering DACs that is reconfigurable and can be used to improve any performance metric is presented. The calibration method can be implemented with or without the proposed add-on, presenting better results for the former case but with the penalty of more die area at the Decode/Memory block, being advisable to use the add-on in case the specifications desired are not reached. The results presented show that the INL and the ENOB achieved by the calibration method with and without add-on are significantly better than those obtained for the calibration method presented in [4], one of the best SSPA calibration methods known.

REFERENCES